



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



ЗБОРНИК РАДОВА ФАКУЛТЕТА ТЕХНИЧКИХ НАУКА

Едиција: Техничке науке - зборници

Година: XXXIX

Број: 3/2024

Нови Сад

Едиција: „Техничке науке – Зборници“

Година: XXXIX

Свеска: 3

Издавач: Факултет техничких наука Нови Сад

Главни и одговорни уредник: проф. др Срђан Колаковић, декан Факултета техничких наука у Новом Саду

Уредништво:

Проф. др Срђан Колаковић
Проф. др Александар Купусинац
Проф. др Борис Думнић
Проф. др Дарко Стефановић
Проф. др Себастиан Балоиш
Проф. др Дејан Лукић
Проф. др Јован Дорић
Проф. др Мирослав Кљајић
Проф. др Немања Тасић
Проф. др Дејан Убавин

Проф. др Милан Видаковић
Проф. др Мирјана Дамњановић
Проф. др Јелена Атанацковић Јеличић
Проф. др Игор Пешко
Проф. др Драган Јовановић
Проф. др Небојша Ралевић
Доц. др Сања Ожват
Проф. др Немања Кашиковић
Проф. др Теодор Атанацковић

Редакција:

Проф. др Дарко Стефановић, главни уредник
Проф. др Жељен Трповски, технички
уредник

Проф. др Драгољуб Новаковић
Проф. др Иван Пинђер
Бисерка Милетић

Језичка редакција:

Бисерка Милетић, лектор
Софија Рацков, коректор
Мр Марина Катић, преводилац

Савет за библиотечку и издавачку делатност ФТН,
проф. др Стеван Станковски, председник.

Штампа: ФТН – Графички центар ГРИД, Трг Доситеја Обрадовића 6, Нови Сад

CIP-Каталогизација у публикацији
Библиотека Матице српске, Нови Сад

378.9(497.113)(082)
62

ЗБОРНИК радова Факултета техничких наука / главни и одговорни уредник
Срђан Колаковић. – Год. 7, бр. 9 (1974)-1990/1991, бр.21/22 ; Год. 23, бр 1 (2008)-. – Нови Сад : Факултет
техничких наука, 1974-1991; 2008-. – илустр. ; 30 цм. –(Едиција: Техничке науке – зборници)

Месечно

ISSN 0350-428X

COBISS.SR-ID 58627591

ПРЕДГОВОР

Поштовани читаоци,

Пред вама је трећа овогодишња свеска часописа „Зборник радова Факултета техничких наука“.

Часопис је покренут давне 1960. године, одмах по оснивању Машинског факултета у Новом Саду, као „Зборник радова Машинског факултета“, а први број је одштампан 1965. године. Након осам публикованих бројева у шест година, пратећи прерастање Машинског факултета у Факултет техничких наука, часопис мења назив у „Зборник радова Факултета техничких наука“ и 1974. године излази као број 9 (VII година). У том периоду у часопису се објављују научни и стручни радови, резултати истраживања професора, сарадника и студената ФТН-а, али и аутора ван ФТН-а, тако да часопис постаје значајно место презентације најновијих научних резултата и достигнућа. Од броја 17 (1986. год.), часопис почиње да излази искључиво на енглеском језику и добија поднаслов «Publications of the School of Engineering». Једна од последица нарастања материјалних проблема и несрећних догађаја на нашим просторима јесте и привремени прекид континуитета објављивања часописа двобројем/двогодишњаком 21/22, 1990/1991. год.

Друштво у коме живимо базирано је на знању. Оно претпоставља реорганизацију наставног процеса и увођење читавог низа нових струка, као и квалитетну организацију научног рада. Значајне промене у структури високог образовања, везане за имплементацију Болоњске декларације, усвајање нове и активне улоге студената у процесу образовања и њихово све шире укључивање у стручне и истраживачке пројекте, као и покретање нових мастер и докторских студија, доносе потребу да ови, веома значајни и вредни резултати, постану доступни академској и широј јавности. Оживљавање „Зборника радова Факултета техничких наука“, као јединственог форума за презентацију научних и стручних достигнућа, пре свега студената, обезбеђује услове за доступност ових резултата.

Због тога је Наставно-научно веће ФТН-а одлучило да, од новембра 2008. год. у облику пилот пројекта, а од фебруара 2009. год. као сталну активност, уведе презентацију најважнијих резултата свих мастер радова студената ФТН-а у облику кратког рада у „Зборнику радова Факултета техничких наука“.

Поред студената мастер студија, часопис је отворен и за студенте докторских студија, као и за прилоге аутора са ФТН или ван ФТН-а.

Зборник излази у два облика – електронском на веб сајту ФТН-а (www.ftn.uns.ac.rs) и штампаном, који је пред вама. Обе верзије публикују се сваки месец, у оквиру промоције дипломираних мастера.

У овом броју штампани су радови студената мастер студија, сада већ мастера, који су радове бранили у периоду од 27.09.2023. до 27.10.2023. год., а који се промовишу 22.03.2024. год. То су оригинални прилози студената са главним резултатима њихових мастер радова.

Известан број кандидата објавили су радове на некој од домаћих научних конференција или у неком од часописа. Њихови радови нису штампани у Зборнику радова.

Велик број дипломираних инжењера–мастера у овом периоду био је разлог што су радови поводом ове промоције подељени у четири свеске.

У овој свесци, са редним бројем 3. објављени су радови из области:

- електротехнике и рачунарства и
- инжењерства информационих система.

У свесци са редним бројем 2. објављени су радови из области:

- машинства,
- грађевинарства,
- саобраћаја,
- графичког инжењерства и дизајна и
- инжењерства заштите на раду и заштите животне средине.

У свесци са редним бројем 4. објављени су радови из области:

- архитектуре,
- мехатронике,
- математике у техници,
- геодезије и геоматике и
- планирања и управљања регионалним развојем.

У свесци са редним бројем 5. објављени су радови из области:

- инжењерског менаџмента,
- биомедицинског инжењерства и
- анимација у инжењерству.

Уредништво се нада да ће и професори и сарадници ФТН-а и других институција наћи интерес да публикују своје резултате истраживања у облику регуларних радова у овом часопису. Ти радови ће бити објављивани на енглеском језику због пуне међународне видљивости и проходности презентованих резултата.

У плану је да часопис, својим редовним изласком и високим квалитетом, привуче пажњу и постане довољно препознатљив и цитиран да може да стане раме-уз-раме са водећим часописима и заслужи своје место на СЦИ листи, чиме ће значајно допринети да се оствари мото Факултета техничких наука:

„Високо место у друштву најбољих“

Уредништво

SADRŽAJ

STRANA

Radovi iz oblasti: Elektrotehnika i računarstvo

1. Andrej Ivičiak, PROGNOZA POTROŠNJE ELEKTRIČNE ENERGIJE PRIMENOM ARMA METODE.....	301-304
2. Aleksa Bu, CMOS TEMPERATURNI SENZOR MALE POTROŠNJE OD 60 nW	305-308
3. Damjan Banjac, EFIKASNO UPRAVLJANJE PODACIMA U VEB APLIKACIJAMA UPOTREBOM REACT QUERY BILBLIOTEKE	309-312
4. Dimitrije Stojanov, PRIMENA KONVOLUCIONIH NEURONSKIH MREŽA ZA DETEKCIJU BOLESTI PNEUMONIJE NAD PACIJENTIMA	313-316
5. Tamara Jančić, SISTEM ZA ZAKAZIVANJE ČASOVA I PRAĆENJE RUTA U AUTO-ŠKOLI	317-320
6. Elena Stojkovski, PRIMENA FOTOAKUSTIČNE SPEKTROSKOPIJE ZA ANALIZU I MERENJE GASOVA	321-324
7. Jelena Cupać, ARHITEKTURA ETHEREUM 2.0	325-328
8. Данило Дробњак, МОДЕЛОВАЊЕ КОНСТРУКЦИОНИХ ОСОБЕНОСТИ АСИНХРОНЕ МАШИНЕ ПРИМЕНОМ СОФТВЕРСКИХ АЛАТА.....	329-332
9. Nemanja Milić, HARDVERSKA IMPLEMENTACIJA RISC-V PROCESORA SA PROTOČNOM OBRADOM KOJA PODRŽAVA RV32IM SKUP INSTRUKCIJA	333-336
10. Oliver Krezović, Dejan Jerkan, UTICAJ RADA SOLARNIH ELEKTRANA NA PARAMETRE KVALITETA NAPONA U DISTRIBUTIVNOJ MREŽI	337-340

	STRANA
11. Теодора Недић, Јелена Сливка, ПРЕПОЗНАВАЊЕ ЉУДСКИХ АКТИВНОСТИ УПОТРЕБОМ МОДЕЛА МАШИНСКОГ УЧЕЊА	341-344
12. Dejan Dokić, RAZVOJ ČET BOTA SA DUGOROČNIM PAMĆENJEM I RAZUMEVANJEM KONTEKSTA	345-348
13. Milica Poparić, PRETPLATNIČKI MODELI U SPOTIFAJ MUZIČKOM STRIMING SERVISU: ANALIZA DIZAJNA I UTICAJA NA KORISNIČKO ISKUSTVO	349-352
14. Александра Цвјетићанин, ПРИМЕНА ЈОТ ТЕХНОЛОГИЈЕ ЗА БОЉЕ РАЗУМЕВАЊЕ И УПРАВЉАЊЕ ОКОЛИНОМ	353-356
15. Никола Ђуровић, ДЕТЕКЦИЈА БОЛЕСТИ КОВИД-19 НА СНИМЦИМА ПЛУЋА УПОТРЕБОМ КОНВОЛУЦИОНИХ НЕУРОНСКИХ МРЕЖА.....	357-360
16. Željka Matović, DIGITALNA PLATFORMA ZA UPRAVLJANJE KVAROVIMA I PLANIRANJE REMONTA U ELEKTROENERGETSKIM SISTEMIMA	361-364
17. Nikola Dodig, DOŽIVLJAJ IGRE KROZ INTERAKCIJU I KONTROLNE MECHANIZME	365-368
18. Isidora Stanić, RAZVOJ RAČUNARSKE IGRE SA VOĆNIM PROTAGONISTOM	369-372
19. Младен Васић, ИЗАЗОВИ ПРИ РАЗВОЈУ МОТО КРОС ВИДЕО ИГРЕ	373-376
20. Milica Brkušanić, Neven Kovački, PRORAČUN TOKOVA SNAGA SA UVAŽENOM UDALJENOM REGULACIJOM NAPONA.....	377-380
21. Младен Алексић, ВИРТУЕЛНА ПАРАМЕТРИЗАЦИЈА ПРОЦЕСНЕ КОНТРОЛЕ У ПОГОНУ ЗА ПАСТЕРИЗАЦИЈУ МЛЕКА	381-384
22. Марија Кљештан, ДЕЦЕНТРАЛИЗОВАНИ СОФТВЕРСКИ СИСТЕМ ЗА ЗАШТИТУ ДИГИТАЛНОГ САДРЖАЈА ПРИМЈЕНОМ КОНЦЕПТА BLOCKCHAIN ТЕХНОЛОГИЈЕ	385-388
23. Pavle Janjić, Zoran Ivanović, PRIMENA UDS PROTOKOLA U AUTOMOBILSKOJ INDUSTRIJI	389-392
24. Никола Мајсторовић, FUZZ ТЕСТИРАЊЕ DJANGO REST API-JA.....	393-397
25. Vuko Jovičić, ANALIZA GEOMETRIJE MANIFOLDA U AUTOENKODERIMA	398-401
26. Никола Ђорђевић, ИМПЛЕМЕНТАЦИЈА ПОДРШКЕ ЗА ВИШЕСТРУКО ИНСТАНЦИРАЊЕ АКТИВНОСТИ У ПРОЦЕСНОМ ОКРУЖЕЊУ NEW WAVE.....	402-405
27. Nikola Rončević, OBRADA VELIKIH KOLIČINA METEOROLOŠKIH PODATAKA.....	406-409
28. Tamara Macura, BEŽIČNO UPRAVLJANJE I KONTROLA PROGRAMABILNOG VOZILA PUTEM VEB I ANDROID APLIKACIJE	410-413
29. Dejan Šorgić, PODRŠKA ZA PYFLIES JEZIK U VS CODE EDITORU.....	414-417

Radovi iz oblasti: Inženjerstvo informacionih sistema

1. Bojana Nešković, UTICAJ ERP SISTEMA I INTEGRACIONIH ALATA U INDUSTRIJI 4.0	418-421
--	---------

	STRANA
2. Petar Maksimović, SISTEM ZA PODRŠKU RADA JAVNE UPRAVE U PROCESU IZDAVANJA PASOŠA I LIČNE KARTE	422-425
3. Nađa Jovanović, INTEGRISANI INFORMACIONI SISTEM ZA REZERVACIJU LEKARSKIH PREGLEDA.....	426-429

PROGNOZA POTROŠNJE ELEKTRIČNE ENERGIJE PRIMENOM ARMA METODE LOAD FORECASTING USING ARMA METHOD

Andrej Iviciak, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu su opisani tipovi prognoza potrošnje električne energije. Opisane su neke metode za prognozu potrošnje i faktori koji su od značaja i utiču na njenu preciznost. U praktičnom delu rada, razvijena je veb aplikacija koja primenom ARMA metode vrši prognozu potrošnje električne energije na osnovu istorijske potrošnje električne energije. U radu će biti prikazani rezultati prognoze potrošnje za neke slučajeve, gde će se videti kako odabir vrednosti parametara i interval pretrage utiče na preciznost prognoze potrošnje električne energije.

Ključne reči: Prognoza potrošnje električne energije, ARMA

Abstract – The paper describes the types of load forecasts. Some load forecast methods and important factors that affect accuracy are described. In the practical part of the work, a web application was developed that uses the ARMA method for load forecasting based on historical electricity consumption. The paper will present results of load forecast for some cases, where it will be possible to see how the selection of parameters and search interval values affects the load forecast's precision.

Keywords: Load forecast, ARMA

1. UVOD

Prognoza potrošnje električne energije je od velikog značaja za svaku državu ili elektroenergetsko preduzeće. Pomoću te informacije se računa i određuje koliko će biti potrebno da se proizvede električne energije i ukoliko je količina van kapaciteta sistema, koliko treba da se kupi. Tokom godina su se razvile razne metode za prognozu potrošnje električne energije. Međusobno se mogu razlikovati po tome koje faktore primenjuju da bi povećali preciznost prognoze, za koji period su najefikasnije, itd. Postoji veliki broj faktora koji utiču na preciznost prognoze potrošnje električne energije. Neki od najvažnijih su istorijski podaci o potrošnji električne energije, jačina vetra, temperatura ili drugim rečima vremenski podaci, vremenske prilike, itd. Prognozu potrošnje električne energije moguće je izvršiti za različite vremenske periode, shodno tome moguće ih je podeliti na:

- kratkoročne (od 1 sata do 1 nedelje)
- srednjoročne (od 1 nedelje do 1 godine)
- dugoročne (od 2 godine pa nadalje)

2. FAKTORI ZA PROGNOZU POTROŠNJE ELEKTRIČNE ENERGIJE

Postoje brojni faktori koji se koriste i utiču na preciznost prognoze potrošnje električne energije, neki su od većeg značaja od drugih ali svi faktori doprinose da se preciznost poveća što je više moguće.

Kod kratkoročnih metoda za prognozu potrošnje neki od faktora sa najvećim značajem su:

- vremenski faktori (dan u nedelji, period u godini, itd.)
- podaci o vremenu (meteorološki podaci)
- potrošačke klase

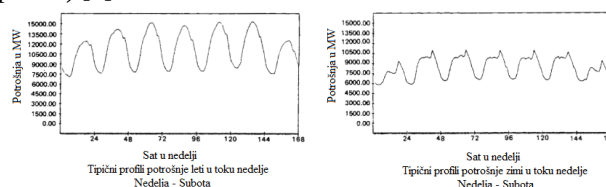
Za srednjoročnu i dugoročnu prognozu postoji veliki broj faktora koji su od značaja, neki od njih su:

- istorijska potrošnja električne energije
- istorijski podaci o vremenu
- broj potrošača po različitim kategorijama
- uređaji u okolini
- ekonomski i demografski podaci, i njihove prognoze

2.1 Vremenski faktori

Značajnu ulogu u uticaju na obrasce potrošnje električne energije, imaju sledeća tri vremenska faktora: sezonski efekti, nedeljno-dnevni ciklusi i zakonski i verski praznici. Sezonske promene određuju da li je isporuka električne energije na vrhuncu tokom leta ili zime. Određene promene u obrascu opterećenja nastaju postepeno kao odgovor na sezonske varijacije kao što su broj sati svetlosti u danu i promena temperature.

Sa druge strane, postoje sezonski događaji koji donose naglu ali važnu strukturalnu modifikaciju u obrascu potrošnje električne energije. To su promene koje donosi letnje računanje vremena (smanjenje ili povećanje broja sati dana), promene u strukturi stope (doba dana, sezonski zahtevi), početak školske godine i značajno smanjenje aktivnosti tokom perioda odmora (Božićni, Novogodišnji period) [1].



Slika 1. Tipični nedeljni obrasci za leto i zimu [1]

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Aleksandar Selakov.

Slika 1. prikazuje tipične potrošačke obrasce letnje i zimske nedelje za leto i zimu kada je isporuka električne energije na vrhuncu [1].

2.2 Vremenske prilike

Jedan od najuticajnijih faktora za prognoze potrošnje električne energije su i podaci o vremenskim prilikama. U obzir se uzimaju razne promenljive. Najčešće korišćene varijable su temperatura i vlažnost vazduha. Indeks temperature i vlažnosti (engl. *Temperature-Humidity index – THI*) i indeks hlađenja vetra (engl. *Wind-Chill index – WCI*) su dve složene vremenske prilike promenljive funkcije, koje su u širokoj upotrebi od strane komunalnih preduzeća kao što su elektrodistribucije. *THI* je mera za nelagodnost od letnje vrućine i tome slično, dok predstavlja *WCI* stres od hladnoće [2].

2.3 Potrošačke klase

Svaka potrošačka klasa ima drugačiji obrazac potrošnje električne energije.

Kada je reč o klasnim razlikama potrošača, njih je moguće podeliti na sledeće potrošačke klase:

- Kućna (kuće, zgrade, itd)
- Komercijalna (prodavnice, bolnice, restorani, hoteli, poslovni prostori, itd)
- Industrijska (proizvodnja, prerada hrane, itd)

Potrošački obrazci se razlikuju između potrošača koji pripadaju drugim klasama, dok su slični oni koji pripadaju istoj klasi. Stoga, većina elektrodistributivnih preduzeća razlikuju potrošnju električne energije od klase do klase [2].

2.4 Faktori za srednjoročnu i dugoročnu prognozu

Kako se period za srednjoročnu i dugoročnu prognozu znatno razlikuje od kratkoročne, istorijski podaci su od najvećeg značaja kako bi prognoza bila što preciznija.

Istorijski podaci koji se koriste su podaci o potrošnji električne energije i podaci o vremenu (o vremenskim prilikama). Što je veća količina istorijskih podataka to je bolje, jer je tada moguće referencirati duži vremenski period i shodno tome postići precizniju prognozu.

Pošto različite kategorije (domaćinstava, ustanove, itd.) troše različitu količinu električne energije, u cilju je da njihov broj bude što tačniji tokom prognoze potrošnje električne energije.

Postoji veliki broj uređaja koji se koriste, svaki od njih troši neku količinu električne energije, neki manju, neki veću. Mnogi faktori utiču na njihovu potrošnju, tu spadaju kvalitet, ekonomičnost, itd.

3. METODE ZA PROGNOZU POTROŠNJE ELEKTRIČNE ENERGIJE

O važnosti predviđanja potrošnje električne energije u određenim područjima, za duži ili kraći vremenski period, svedoči veliki broj metoda koje su u tu svrhu razvijene. Sledi pregled važnijih metoda koje će biti bliže objašnjene u daljem tekstu.

Neke od metoda za kratkoročnu prognozu:

- regresioni modeli (engl. *regression models*)
- vremenske serije (engl. *time series*)
- neuronske mreže (engl. *neural networks*)
- statički algoritmi učenja

Metode za srednjoročnu i dugoročnu prognozu:

- metode koje se oslanjaju na krajnje potrošače (engl. *end-use*)
- ekonometrijski pristup (engl. *econometric approach*)

Predviđa se da će stalan razvoj, unapređenje i pronalazak novih matematičnih alata, dovesti do razvoja (velikog broja) preciznijih tehnika za predviđanje potrošnje električne energije, kao i do novih pristupa i metoda.

3.1 Kratkoročni pristupi prognoze električne energije

Postoji veliki broj, kako pristupa tako i metoda za prognozu kratkotrajne potrošnje električne energije, oni su najčešće bazirani na statistici, matematičkim modelima ili veštačkoj inteligenciji.

3.1.1 Regresioni modeli (engl. *regression models*)

Regresioni pristup je jedan on najšire korišćenih pristupa, koji se zasniva na statističkoj tehnici. Regresione metode najčešće se koriste za modeliranje modela odnosa između potrošnje električne energije i drugih faktora kao što su vremenske prilike, tip dana i potrošačka klasa, za prognozu električne potrošnje [2].

3.1.2 Vremenske serije (engl. *time series*)

Metode vremenskih serija zasnivaju se na pretpostavci da podaci imaju internu strukturu kao što su autokorelacija, trend ili sezonska varijacija. Vremenske serije pronalaze i istražuju takve strukture [2].

Najčešće korišćene metode su:

- ARMA (*autoregressive moving average*)
- ARIMA (*autoregressive integrated moving average*)
- ARMAX (*autoregressive moving average with exogenous variables*)
- ARIMAX (*autoregressive integrated moving average with exogenous variables*)

ARMA modeli se najčešće koriste za stacionarne procese dok je ARIMA predstavlja poboljšanje ARMA za nestacionarne procese. ARMA i ARIMA koriste kao ulazne parametre samo vreme i električnu potrošnju, pošto električna potrošnja zavisi od vremenskih pojava i doba dana, dok se ARIMAX najčešće koristi među klasičnim modelima vremenskih serija [2].

3.1.3 Neuronske mreže (engl. *neural networks*)

Neuronske mreže i veštačka inteligencija postaju sve zastupljene tokom godina. Veštačke neuronske mreže (engl., *artificial neural network* - ANN ili *neural network* - NN) se široko koriste za prognozu potrošnje električne energije još od 1990. godine [2].

Izlazne vrednosti od ANN predstavljaju linearne ili nelinearne matematičke transformacije, ulaznih parametara. Ulazne vrednosti kod ANN mogu biti izlazi nekih drugih mreža. U praksi se najčešće koristi mali broj mrežnih elemenata koji su međusobno povezani kroz različite slojeve, kao mrežni ulazi i izlazi. Postoji više arhitektura koje se koriste za prognozu potrošnje električne energije, kao što su *Hopfield*, *Boltzmann machine*).

Najpopularnija arhitektura za prognozu električne energije koristi propagacije unazad (*back propagation*).

Propagacija unazad koristi kontinuirane vrednosne funkcije i učenje pod nadzorom [2].

3.1.4 Statički algoritmi učenja

Među statičke algoritme učenja spadaju metode potpornih vektora (engl. *support vector machines* - SVMs), SVMs je moćna i među skorijim tehnikama (algoritmima), koja rešava klasifikacione i regresione probleme. Za razliku od ANN, čiji cilj je da definišu kompleksne funkcije ulaznih obeležja, SVMs izvršavaju nelinearno preslikavanje (koristeći tzv. kernel funkcije) podataka u prostor više dimenzije (osobine). Zatim SVMs koriste proste linearne funkcije za kreiranje linearnih granica za odlučivanje u novoj dimenziji (prostoru). Dok je kod ANN problem bio odabir arhitekture, za SVMs problem predstavlja odabir odgovarajućeg kernela [2].

3.2 Srednjoročni i dugoročni pristupi prognoze potrošnje električne energije

Za srednjoročnu i dugoročnu prognozu potrošnje električne energije se najčešće koriste metode koje se oslanjaju na krajnje potrošače i ekonometrijsko modelovanje ili njihova kombinacija.

3.2.1 Modeli koji se oslanjaju na krajnje potrošače (engl. *end-use models*)

Kao što im ime na to i ukazuje, u pitanju su modeli koji se oslanjaju na podatke o potrošnji krajnjih potrošačkih uređaja, odnosno, na podatke sledećeg tipa: koji uređaji su u pitanju, ko su krajnji potrošači, koje su njihove potrebe, njihova starosna dob, veličina stambenog objekta, itd. Ovi podaci zajedno sa njihovim promenama koje se dinamički dešavaju, su osnova prognoze potrošnje električne energije kod modela koji se oslanjaju na krajnje potrošače.

3.2.2 Ekonometrijski pristup (engl. *econometric approach*)

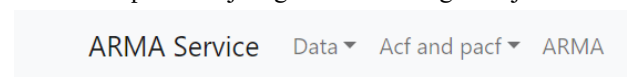
Ekonometrijski pristup kombinuje ekonomiju i statističke tehnike za prognozu potrošnje električne energije. Procenjuju se veze između same potrošnje električne energije kao i faktora koji utiču na njenu potrošnju. Veze se procenjuju metodom najmanjeg kvadrata ili vremenskom serijom [2].

Moguće je integrisati ekonometrijski model u model krajnje upotrebe tako što će se predstaviti dodatne komponente za praćenje ponašanja unutar jednačina krajnje upotrebe [2].

4. KORISNIČKI MENI APLIKACIJE

Razvijena je aplikacija koja primenom ARMA metode vrši prognozu potrošnje električne energije za određeni period. Ona je realizovana kao veb aplikacija, serverska strana je implementirana pomoću Django *framework*-a.. Klijentska strana je implementirana pomoću React.js biblioteke kao SPA (engl. *Single Page Application*) aplikacija, koja prikazuje podatke koji se dobiju od strane servera kao grafikoni, za čije iscrtavanje se koristila Chart.js biblioteka. Podaci koji se koriste za prognozu potrošnje električne energije su istorijski podaci električne energije. Korišćeni podaci se nalaze unutar baze podataka koja predstavlja csv datoteku i sadrži istorijsku potrošnju električne energije za svaki sat u period 2016. – 2019. godine

Na Slici 2. prikazan je izgled korisničkog menija.



Slika 2. Korisnički meni

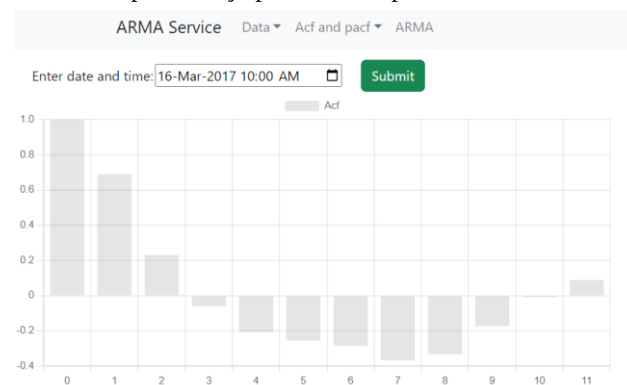
4.1 ACF

Autokorelacija (ACF – engl. *Autocorrelation function*), poznata kao i serijska korelacija u slučaju diskretnog vremena, je korelacija signala sa odloženom kopijom tog signala kao funkcija kašnjenja. Neformalno, to je sličnost između posmatranja slučajne promenljive kao funkcije vremenskog odmaka između njih [3].

Autokorelacija se kod regresionih analiza tipično koristi za modelovanje autoregresionih modela (AR), modela pokretnog proseka (MA), njihova kombinacija ARMA ili ARIMA modela koji je proširen model ARMA modela.

U proceni modela pokretnog proseka (MA), funkcija autokorelacije se koristi za određivanje odgovarajućeg broja termina greške sa kašnjenjem koji će biti uključeni.

Na Slici 3. prikazan je primer ACF prikaza.

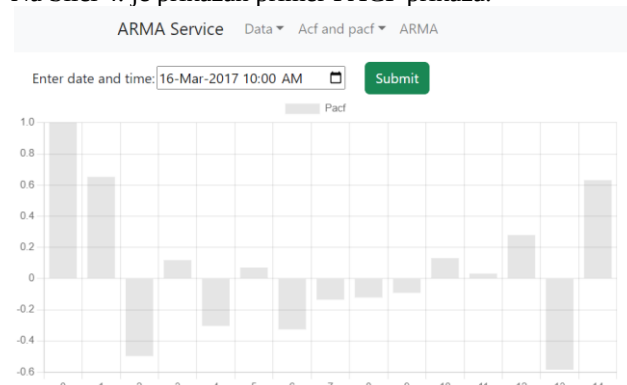


Slika 3. Acf prikaz

4.2 PACF

U analizi vremenskih serija, funkcija parcijalne autokorelacije (PACF - engl. *Partial autocorrelation function*) daje delimičnu korelaciju stacionarne vremenske serije sa sopstvenim zaostalim vrednostima, regresionim vrednostima vremenske serije na svim kraćim kašnjenjima. To je u suprotnosti sa funkcijom autokorelacije, koja ne kontroliše druga zaostajanja. Ova funkcija igra važnu ulogu u analizi podataka koja ima za cilj da identifikuje obim zaostajanja u modelu autoregresije (AR) [4].

Na Slici 4. je prikazan primer PACF prikaza.



Slika 4. Pacf prikaz

4.3 ARMA

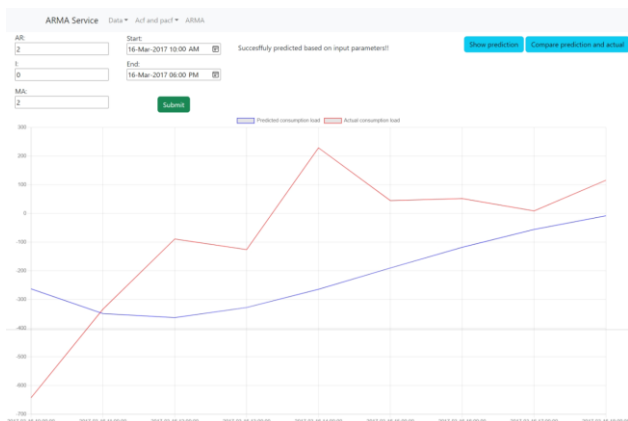
Auto Regressive Moving Average ili ukratko ARMA je klasa modela koja analizira datu vremensku seriju na osnovu njenih vrednosti iz prošlosti, njenih kašnjenja i grešaka prognoze i koristi se da predvidi dalje ponašanje date vremenske serije [5].

ARMA model se može karakterizovati sa dva pojma p i q, gde p je red za AR pojam a q je red za MA pojam.

AR i MA su modeli pomoću kojih se definiše ARMA model.

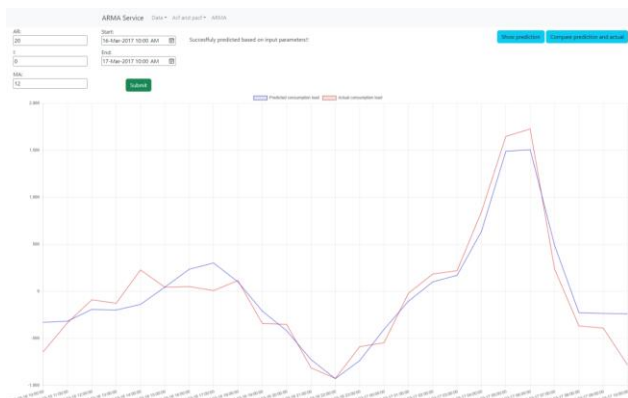
ARMA model:

$$X_t = \varepsilon_t + \sum_{i=1}^p \beta_i X_{t-i} + \sum_{i=1}^q \theta_i \varepsilon_{t-i}$$



Slika 5. ARMA(2,2) očekivane i prave vrednosti za 8h

Na Slici 5. je prikazan rezultat prognoze potrošnje za ARMA(2, 2) i prave vrednosti koje su dobijene iz baze podataka.



Slika 6. ARMA (20,12) očekivane vrednosti i prave vrednosti za 24h

Na Slici 6. je prikazan rezultat prognoze potrošnje za ARMA(20, 12) i prave vrednosti koje su dobijene iz baze podataka. Upoređivanjem Slike 5. i Slike 6. se može primetiti kako izbor različitih redova za ARMA proračun, dobije se različita prognoza.

5. ZAKLJUČAK

Prognoza potrošnje električne energije je od velikog značaja za svaku državu ili preduzeće. Postoji više vrsta i više podela prognoza električne energije, po trajnosti, to su kratkoročne, srednjeročne i dugoročne.

Postoji veliki broj faktora koji mogu da utiču na faktor prognoze električne energije, neki od njih su vremenski faktori koji prate potrošnju energije u zavisnosti od dana, godišnja doba, tu su vremenski uslovi koji se često koriste i od velikog su značaja, kao što su temperature, jačina vetra, vlažnost vazduha, itd. Potrošačke kategorije u kombinaciji sa ekonomskim stanjem domaćinstava zajedno sa kvalitetom uređaja koji se koriste, su direktno povezati i utiču na prognozu potrošnje električne energije.

U ovom radu se koristio ARMA model za prognoziranje potrošnje električne energije. ARMA model sa sastoji od dva modela, AR modela i MA modela. Gde se AR i MA modelima dodeljuju vrednosti, npr. ARMA (20, 12), gde 20 predstavlja red koji se vezuje za model AR a 12 red koji se vezuje za model MA.

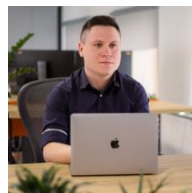
AR model radi sa istorijskim vrednostima dok MA model radi sa greškama i odstupanja u proračunu, a zajedno čine ARMA model. Dok AR model pokušava da prognozira potrošnju na osnovu istorijskih vrednosti, MA model pokušava da popravi napravljene greške da bi krajnji rezultat bio što tačniji.

Izbor redova ARMA modela u velikoj meri utiče na tačnost samog modela, AR deo se bira na pomoću PACF funkcije a MA pomoću ACF funkcije. Ukoliko se izabere pogrešan red u prethodnom poglavlju se moglo videti koliko to loše utiče na tačnost prognoze.

6. LITERATURA

- [1] G. Gross, F. D. Galiana, Short-term forecasting, *Proceedings of the IEEE*, December 1987
- [2] Eugene A. Feinberg, Dora Genethliou, Load Forecasting, State University of New York, 2005
- [3] <https://en.wikipedia.org/wiki/Autocorrelation>, *Autocorrelation*, pristupljeno: jul 2023.
- [4] https://en.wikipedia.org/wiki/Partial_autocorrelation_function, *Partial autocorrelation function*, pristupljeno: jul 2023.
- [5] https://en.wikipedia.org/wiki/Autoregressive%E2%80%93moving-average_model, *Autoregressive – moving – average model*, pristupljeno: jul 2023.

Kratka biografija:



Andrej Ivičiak rođen je u Novom Sadu 1996. godine. Osnovne akademske studije na Fakultetu tehničkih nauka Univerziteta u Novom Sadu upisao je 2015. godine. Diplomirao je 2019. godine i iste godine upisuje master akademske studije.

Kontakt: andrejftniviciak@gmail.com

CMOS TEMPERATURNI SENZOR MALE POTROŠNJE OD 60 nW
A CMOS ON-CHIP 60 nW LOW-POWER ON-CHIP TEMPERATURE SENSOR

Aleksa Bu, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Tema ovog rada jeste projektovanje CMOS temperaturnog senzora zasnovanog na MOSFET uređajima. Senzor je projektovan u UMC 0.18 μm tehnologiji. Projektovani senzor postiže nisku potrošnju od 60 nW i zauzima malu površinu od 0.023 mm^2 sa maksimalnom greškom merenja od $+1.8^\circ\text{C}/-1.3^\circ\text{C}$ u temperaturnom opsegu od $20\text{--}80^\circ\text{C}$.

Ključne reči: CMOS, temperaturni senzor, MOSFET

Abstract – This paper proposes an all-MOSFET CMOS temperature sensor. The sensor was designed in UMC 0.18 μm technology. The implemented sensor achieves low-power consumption of 60 nW, while requiring a small area equal to 0.023 mm^2 with a maximum measurement error equal to $+1.8^\circ\text{C}/-1.3^\circ\text{C}$ in the temperature range of $20\text{--}80^\circ\text{C}$.

Keywords: CMOS, temperature sensor, all-MOSFET

1. UVOD

Razvoj različitih senzorskih sistema, kao što su sistemi sa radio-frekvencijskom identifikacijom (RFID), zahteva temperaturne senzore sa veoma malom potrošnjom. Tradicionalni temperaturni senzori izrađeni u integrisanim tehnologijama koriste bipolarne tranzistore i nemaju dovoljno dobre osobine za ove primene, pošto njihova potrošnja može biti reda 10 μW [1]. Usled toga su razvijeni senzori zasnovani na MOSFET uređajima. Ovi senzori mogu postići potrošnju manju od 1 μW , sa greškom merenja manjom od 2°C [2-4].

Neke od tehnika kojim se može realizovati temperaturni senzor su korišćenje oscilatora sa linearnom temperaturnom zavisnošću [3],[5], korišćenje temperaturno zavisnih kola za generisanje temperaturno zavisnog napona ili struje [6],[7], korišćenje temperaturno zavisnih linija za kašnjenje [4],[8]. Generalno, manja potrošnja se dobija na račun manje preciznosti; senzor iz rada [6] postiže malu grešku merenja od $\pm 0.6^\circ\text{C}$ i ima potrošnju jednaku 6 μW , dok senzor iz rada [2] ima potrošnju od 71 nW, sa greškom od $\pm 1.5^\circ\text{C}/-1.4^\circ\text{C}$.

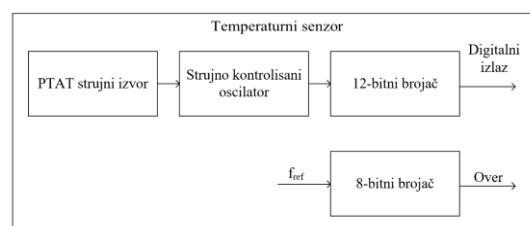
Temperaturni senzor predstavljen u ovom radu koristi MOSFET tranzistore, sa ciljem realizovanja senzora koji ima potrošnju manju od 100 nW, i grešku merenja manju od 2°C u temperaturnom opsegu od $20\text{--}80^\circ\text{C}$. Senzor je projektovan i analiziran uz pomoć Cadence programskog paketa u UMC 0.18 μm tehnologiji.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Mirjana Damnjanović, red.prof.

2. PROJEKTOVANJE TEMPERATURNOG SENZORA

Blok šema predloženog senzora je prikazana na slici 1. Izabran je napon napajanja od 1 V, sa ciljem postizanja što manje potrošnje.

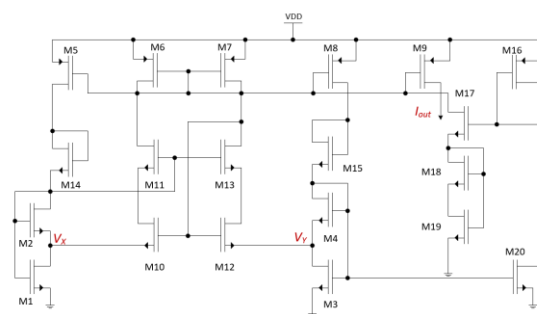


Slika 1. Blok dijagram predloženog temperaturnog senzora

Prvi blok, koji se može smatrati osnovom temperaturnog senzora, je strujni izvor čija je struja direktno proporcionalna temperaturi. Ovaj blok je označen kao PTAT strujni izvor (eng. *proportional to absolute temperature*). Zavisnost izlazne struje PTAT izvora od temperature mora biti izrazito linearna za ispravno funkcionisanje senzora. Izlazna struja PTAT izvora služi kao ulazna struja strujno kontrolisanog oscilatora, čija frekvencija oscilovanja veoma slabo zavisi od temperature. Usled toga, na izlazu oscilatora se dobija signal čija je frekvencija linearno zavisna od temperature. Za merenje frekvencije oscilovanja oscilatora koriste se dva asinhrona brojača. Na ulaz jednog brojača je vezan izlazni signal oscilatora i taj brojač daje digitalni izlaz čitavog senzora. Na ulaz drugog brojača je vezan signal referentne, temperaturno nezavisne, frekvencije. Referentni brojač je osmobaritni, a drugi brojač je dvanaestobaritni [2].

2.1. Projektovanje PTAT strujnog izvora

Implementirani strujni izvor je modifikacija strujnog izvora predloženog u [9]. Šematik predloženog kola je prikazan na slici 2.



Slika 2. Električna šema predloženog PTAT strujnog izvora koji daje temperaturno zavisnu struju I_{out} srazmernu naponu V_Y

Novi strujni izvor je izabran sa ciljem postizanja što manje potrošnje, što veće linearnosti, i da pri tome ima što manju površinu.

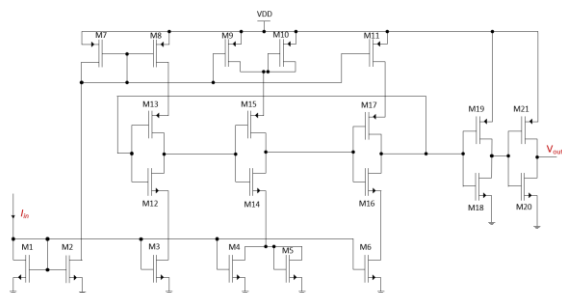
Rad strujnog izvora je zasnovan, pre svega, na kaskodnim MOSFET tranzistorima M_1 i M_2 , kao i kaskodnim tranzistorima M_3 i M_4 , koji rade u režimu ispod praga provođenja. Tranzistori M_3 i M_4 generišu napon V_Y , koji je linearno zavisn od temperature. Taj napon se skoro identično preslikava na napon V_X , između tranzistora M_1 i M_2 . Kako je struja drejna tranzistora M_1 i M_2 linearno zavisna od napona V_X , ostvareno je da njihova struja drejna bude temperaturno zavisna (tzv. PTAT ponašanje). Usled toga je linearnost čitavog PTAT strujnog izvora veoma zavisna od linearnosti napona V_Y .

PMOS tranzistori M_5 - M_9 imaju ulogu strujnog ogledala koje osigurava da su struje u svim granama kola jednake. Zbog toga moraju da imaju iste dimenzije. Deo kola između dve kaskode, sastavljen od tranzistora M_{10} - M_{13} , zajedno sa tranzistorima M_6 - M_7 , formira strujno ogledalo koje ima ulogu preslikavanja napona V_Y na V_X . Tranzistori M_{14} i M_{15} su iskorišćeni da bi se smanjio uticaj procesnih odstupanja, za koje se očekuje da predstavljaju značajan problem. Svi MOSFET uređaji su iz istog razloga odabrani sa velikim dužinama i širinama kanala. Izlazna struja I_{out} se dobija kao struja drejna MOSFET tranzistora M_9 . Kolo je dimenzionisano tako da izlazna struja ima vrednost nekoliko nA za čitav temperaturni opseg. Opseg vrednosti struja iznosi od 2.7 nA (na 20°C) do 5.4 nA (na 80°C).

Projektovano kolo ima dve mirne radne tačke. Jedna mirna radna tačka je nepoželjna i ima sve struje jednake nuli. Da bi se taj slučaj izbegao, potrebno je koristiti *startup* kolo, koje obezbeđuje odgovarajući režim rada. Ovo kolo je sačinjeno od tranzistora M_{16} - M_{20} . *Startup* kolo obezbeđuje da se na gejtovima tranzistora M_5 - M_9 nađe naelektrisanje potrebno za prelazak u provodni režim pri uključenju kola. Dimenzije tranzistora strujnog izvora su prikazane u tabeli 1.

2.2. Projektovanje strujno kontrolisanog oscilatora

Strujno kontrolisani prstenasti oscilator koji je prikazan u ovom radu (sl. 3) predstavlja modifikaciju oscilatora [10]. Ovo kolo se sastoji od strujno kontrolisanih invertora (eng. *current-starved inverter*), čime se postiže mala potrošnja. Ulazna struja oscilatora I_{in} je ujedno izlazna struja PTAT strujnog izvora. Struja I_{in} kontroliše frekvenciju oscilovanja prstenastog oscilatora, tako što se kašnjenje svakog od invertorskih stepena smanjuje povećavanjem struje I_{in} . Tranzistori M_1 , M_2 i M_7 prosleđuju ulaznu struju I_{in} do invertorskih stepena.



Slika 3. Električna šema strujno kontrolisanog prstenastog oscilatora

Oscilator sadrži tri strujno kontrolisana invertorska stepena. Prvi je sastavljen od tranzistora M_3 , M_{12} , M_{13} , M_8 , drugi od M_4 , M_5 , M_{14} , M_{15} , M_9 , M_{10} i treći od M_6 , M_{16} , M_{17} , M_{11} . Minimalan broj invertorskih stepena (tri) osigurava najmanju moguću zauzetu površinu i potrošnju. Nakon njih su povezana dva standardna invertora (sačinjena od tranzistora M_{18} , M_{19} i M_{20} , M_{21}). Dva standardna invertora na izlazu su neophodna da bi se dobio pravougaoni izlazni signal V_{out} , pune amplitude i velike strmine, kao i da frekvencija oscilovanja ne zavisi od kapacitivnosti koja je vezana na izlaz oscilatora. Minimalna frekvencija oscilovanja, na 20°C je 7 kHz, a maksimalna, na 80°C je 13.4 kHz. Kao i kod strujnog izvora, korišćeni su tranzistori sa velikim dimenzijama da bi se smanjio uticaj procesnih promena (tabela 2).

2.3. Projektovanje digitalnih brojača

Za dobijanje digitalnog izlaza senzora, korišćena su dva asinhrona brojača, jedan osmobitni i jedan dvanaestobitni (slika 1). Na ulaz dvanaestobitnog brojača se dovodi izlaz prethodnog stepena, oscilatora, tj. napon V_{out} . Frekvencija signala V_{out} određuje brzinu kojom taj brojač broji. Osmobitni brojač pruža vremenski interval u kojem se merenje izvršava, čija je dužina određena referentnom frekvencijom f_{ref} . Kako se izlazni signal dobija sa dvanaestobitnog brojača, signal na izlazu je dvanaestobitni binarni broj, koji mora biti mapiran na vrednost temperature. Taj postupak se izvršava nakon proizvodnje senzora. Brojači su realizovani pomoću D flipflopova sa asinhronim resetom, koji koriste MOSFET tranzistore sa minimalnim dimenzijama.

Da bi merenje bilo započeto, oba brojača treba da budu resetovana, nakon čega počinju sa brojanjem od nultog stanja. Merenje traje sve dok referentni osmobitni brojač ne stigne do svog poslednjeg stanja, kada signal *Over* ima vrednost jedan. Tada se očitava vrednost sa izlaza dvanaestobitnog brojača. Frekvencija i broj bita referentnog brojača moraju biti izabrani tako da nikada ne dođe do prekoračenja opsega drugog brojača, a da rezolucija i vreme potrebno za merenje budu dovoljno dobri [2]. Frekvencija referentnog brojača je 5 kHz, pa je vremenski interval merenja jednak 51.2 ms (pošto je brojač osmobitni).

Tabela 1. Dimenzije PTAT strujnog izvora

MOSFET	W/L [μm]
M_1	2/40
M_2	2/16.5
M_3	13/18.3
M_4	25/9.1
M_{5-9}	10/10
$M_{10,12}$	2.6/7.5
$M_{11,13}$	2.5/5
$M_{14,15}$	2.5/10
M_{16}	0.5/27
M_{17}	0.5/20
$M_{18,19}$	0.5/10
M_{20}	10/5

Tabela 2. Dimenzije strujno kontrolisanog oscilatora

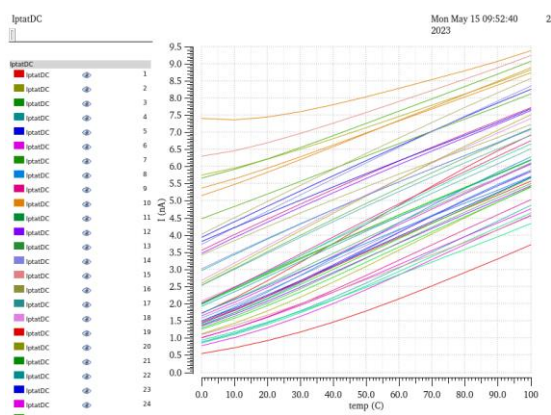
MOSFET	W/L [μm]
M_1 , M_{11}	4/4
M_{12} , M_{13} , M_{16} , M_{17}	2/2
M_{14}	6.5/7.5
M_{15}	5/5
M_{18} , M_{20}	1/10
M_{19} , M_{21}	1/5

3. REZULTATI SIMULACIJA

3.1. Rezultati simulacija za PTAT strujni izvor

Najvažnija osobina koju treba ispitati kod strujnog izvora je uticaj procesnih odstupanja na linearnost zavisnosti struje od temperature. Na slici 4 su prikazani rezultati 50 Monte Karlo simulacija za strujni izvor, urađenih u Cadence programskom paketu, koje imaju za cilj da ispituju tu osobinu. Veći deo karakteristika pokazuje dobru linearnost u temperaturnom opsegu od 0 do 100°C, dok jedan deo karakteristika ima veliko odstupanje na početku i kraju tog opsega. Usled toga je izabran baš temperaturni opseg od 20-80°C. Nakon smanjenja opsega, odstupanje od idealne karakteristike u najgorem slučaju je manje od 10%.

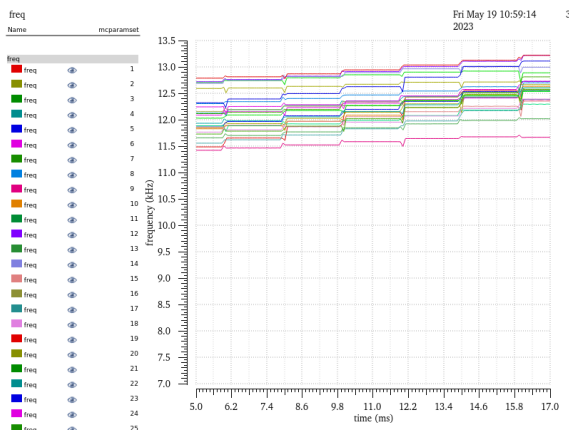
U slučaju bez procesnih varijacija, odstupanje od idealne linearne karakteristike je manje od 1%, a potrošnja na sobnoj temperaturi iznosi 29 nW.



Slika 4. Analiza uticaja procesnih promena preko Monte Karlo simulacija za strujni izvor za zavisnost izlazne struje od temperature u opsegu 0°C-100°C

3.2. Rezultati simulacija za strujno kontrolisani oscilator

Kod strujno kontrolisanog oscilatora, najvažnija osobina koju treba ispitati je zavisnost frekvencije oscilovanja od temperature. Kao i kod PTAT strujnog izvora, biće iskorišćena Monte Karlo analiza da bi se ovo ponašanje posmatralo, uzimajući u obzir i procesne varijacije. Na slici 5. su prikazani rezultati 50 Monte Karlo simulacija urađenih u Cadence programskom paketu.



Slika 5. Analiza uticaja procesnih promena preko Monte Karlo simulacija za oscilator za zavisnost frekvencije od temperature u opsegu od 20°C do 80°C

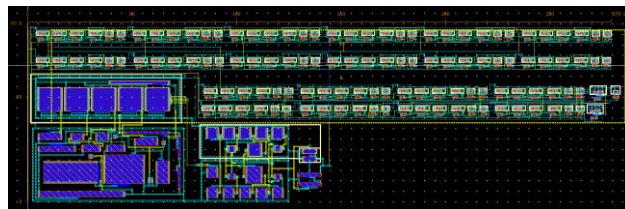
Temperatura je menjana za 10°C svake 2 ms, i rezultati su prikazani u opsegu od 20°C do 80°C. Oblik karakteristika je sličan u svim slučajevima, što je poželjno. Ovaj oblik karakteristika, sa malom nelinearnošću pri povećavanju frekvencije sa temperaturom, izabran je da bi se smanjila nelinearnost strujnog izvora.

3.3. Rezultati simulacija za PTAT strujni izvor i oscilator zajedno

Naredni korak je posmatranje ponašanja sistema koji nastaje kada se prethodna dva kola povežu. Kada se kolo simulira bez procesnih odstupanja, potrošnja iznosi 43 nW na sobnoj temperaturi, dok je minimalna frekvencija oscilovanja 7 kHz (na 20°C) i maksimalna frekvencija 13.4 kHz (na 80°C).

3.4. Rezultati simulacija za kompletan senzor

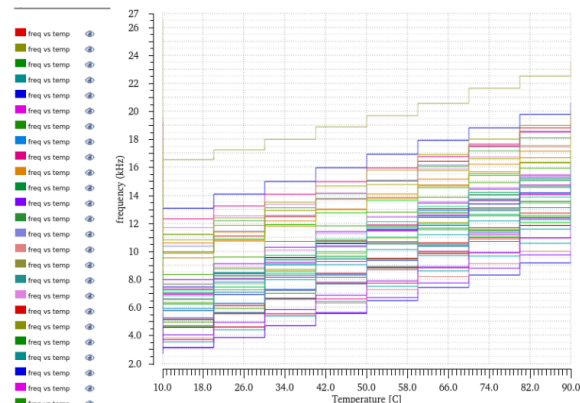
Kompletan senzor je posebno simuliran nakon što je projektovan njegov lejaut, koji je prikazan na slici 6. Površina senzora je oko 0.023 mm² (83.5 μm×279.2 μm). Potrošnja senzora na sobnoj temperaturi iznosi oko 60 nW. Vreme potrebno za jedno merenje iznosi 51.2 ms, što znači da je energija koja se potroši za jedno merenje jednaka 3.1 nJ.



Slika 6. Lejaut predloženog senzora

4. ANALIZA REZULTATA

Posebno važan parametar za senzor je njegova preciznost, tj. maksimalna vrednost greške merenja. Da bi se ispitala vrednost ovog parametra, urađena je Monte Karlo simulacija za strujni izvor i oscilator zajedno (slika 7). Nakon toga, izabrane su proizvoljno dve temperature (30°C i 70°C) i za svaku od dobijenih karakteristika određene su vrednosti frekvencija. Na osnovu ovih parova tačaka temperatura-frekvencija, dobijene su linearne karakteristike (tj. prave). Pomoću tih karakteristike se dobijene vrednosti frekvencije mapiraju na temperaturu. Ovaj postupak odgovara kalibraciji u dve tačke (eng. *two-point calibration*).

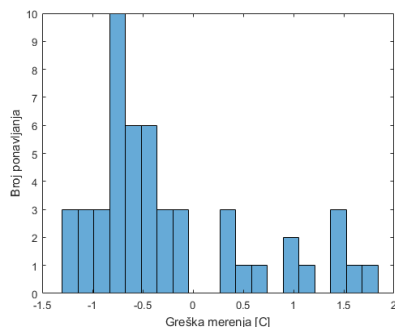


Slika 7. Analiza uticaja procesnih promena preko Monte Karlo simulacija za strujni izvor i oscilator zajedno, za zavisnost frekvencije od temperature u opsegu 10°C-90°C

Tabela 3. Poređenje predloženog senzora sa drugim CMOS temperaturnim senzorima

Parametar	Predloženi senzor	[2]	[11]	[4]	[6]	[5]
Tehnologija	0.18 μ m	0.18 μ m	0.13 μ m	0.18 μ m	0.13 μ m	0.18 μ m
Potrošnja	60nW	71nW	150nW	112nW	6 μ W	92nW
Površina	0.023mm ²	0.09mm ²	0.0014mm ²	0.0125mm ²	0.00916mm ²	0.021mm ²
Napon napajanja	1V	1.2V	1.1V	0.5V	1.3V	1V
Temperaturni opseg	20 – 80°C	0-100°C	-60-40°C	-10-20°C	-20-85°C	-30-70°C
Nepreciznost	+1.8°C/-1.3°C	+1.5°C/-1.4°C	2°C (3 σ value)	+0.3°C/-0.1°C	+0.6°C/-0.6°C	+0.6°C/-0.7°C
Rezolucija	0.18°C	0.3°C	0.5°C	N/A	0.0123°C	0.15°C
Vreme konverzije	51.2 ms	30 ms	1000 ms	N/A	2 ms	N/A

Koristeći ovaj postupak, izračunata je maksimalna greška za svako od 50 merenja prikazanih na slici 7. Rezultati su prikazani na slici 8. Maksimalna moguća greška merenja na tom skupu od 50 merenja iznosi +1.8°C/-1.3°C.



Slika 8. Očekivana maksimalna greška merenja

Potencijalna poboljšanja predloženog senzora uključuju kolo za kalibraciju, koje eliminiše potrebu za spoljašnjom kalibracijom, zatim kolo za naponsku regulaciju, koje smanjuje osetljivost kola na napon napajanja, kao i kontrolnu logiku senzora i generator referentne frekvencije osmokitnog brojača.

Pomenute komponente mogu biti realizovane i na čipu i van njega. Poboljšanje senzora se može postići i prelaskom u nižu tehnologiju, jer se tako dobija brojač sa manjom potrošnjom i manjom površinom. Konačno, predloženi senzor je upoređen sa drugim relevantnim CMOS temperaturnim senzorima (tabela 3).

5. ZAKLJUČAK

U ovom radu je predložen temperaturni senzor sačinjen od PTAT strujnog izvora, strujno kontrolisanog oscilatora i asinhronih brojača. Senzor postiže izuzetno malu potrošnju od 60 nW, zauzima malu površinu na čipu od 0.023 mm². Senzor je usled toga dobar izbor za primene gde je potrebna izuzetno mala potrošnja. Mala površina koju senzor zauzima omogućava postizanje daljeg poboljšanja sa dodatnim komponentama, kao što su kolo za kalibraciju, kolo za regulaciju napona napajanja i kolo za generaciju referentne frekvencije.

6. LITERATURA

- [1] F. Sebastiano, L. J. Breems, K. A. A. Makinwa, S. Drago, D. M. W. Leenaerts, B. Nauta, "A 1.2V 10 μ W NPN-based temperature sensor in 65nm CMOS with an inaccuracy of $\pm 0.2^\circ\text{C}$ (3 σ) from -70°C to 125°C " 2010 IEEE International Solid-State Circuits Conference, San Francisco, USA, 2010, pp. 312-313.
- [2] S. Jeong, Z. Foo, Y. Lee, J. -Y. Sim, D. Blaauw, D. Sylvester, "A Fully-Integrated 71 nW CMOS Temperature Sensor for Low Power Wireless Sensor

Nodes", IEEE J. Solid-State Circuits, vol. 49, no. 8, pp. 1682-1693, Aug. 2014.

- [3] Z. Shenghua, W. Nanjian, "A novel ultra low power temperature sensor for UHF RFID tag chip", 2007 IEEE Asian Solid-State Circuits Conference, Jeju, Korea (South), 12-14 Nov. 2007, pp. 464-467.
- [4] R. Dastanian, E. Abiri, M. Ataiyan, "A 0.5 V, 112 nW CMOS temperature sensor for RFID food monitoring application", 24th Iranian Conf. on Electrical Engineering (ICEE), Shiraz, Iran, 10-12 May 2016, pp. 1433-1438.
- [5] Deng F, He Y, Li B, Zhang L, Wu X, Fu Z, Zuo L. „Design of an Embedded CMOS Temperature Sensor for Passive RFID Tag Chips“, Sensors, vol. 15, no. 5, pp:11442-11453, 2015.
- [6] A. Ballo, G. Bruno, A. D. Grasso, M. Vaiana, "A Compact CMOS Temperature Sensor for On-Chip Thermal Monitoring", 2020 Inter. Conf. on Electrical, Communication, and Computer Engineering, Istanbul, Turkey, 12-13 June 2020, pp. 1-5.
- [7] C. Zhao, Y. -T. Wang, D. Genzer, D. Chen, R. Geiger, "A CMOS on-chip temperature sensor with -0.21°C 0.17°C inaccuracy from -20°C to 100°C ", 2013 IEEE Inter. Symp. on Circuits and Systems, Beijing, China, 19-23 May 2013, pp. 2621-2625.
- [8] G. Popov, F. C. Dualibe, V. Moeyaert, P. Ndungidi, H. García-Vázquez, C. Valderrama, "A 65-nm CMOS battery-less temperature sensor node for RF-powered wireless sensor networks", in Proc. IEEE Wireless Power Transfer Conf., Aveiro Municipality, Portugal, 5-6 May 2016, pp. 1-4.
- [9] E.M. Camacho-Galeano, J.Q. Moreira, M.D. Pereira, A.J. Cardoso, C. Galup-Montoro, M. Schneider, „Temperature performance of sub-1V ultra-low power current sources“, 2008 IEEE International Symp. on Circuits and Systems, Seattle, USA, 18-21 May 2008, pp. 2230-2233.
- [10] F. Deng, H. Yigang, B. Li, L. Zuo, X. Wu, Z. Fu, „A CMOS Pressure Sensor Tag Chip for Passive Wireless Applications“, Sensors (Basel, Switzerland), vol. 15, no. 15, pp: 6872-6884, 2015.
- [11] W. Yang, H. Jiang, Z. Wang, "A 0.0014 mm² 150 nW CMOS Temperature Sensor with Nonlinearity Characterization and Calibration for the -60 to $+40^\circ\text{C}$ Measurement Range", Sensors, vol. 19, no. 8, pp: 1777, 2019.

Kratka biografija:



Aleksa Bu rođen je u Novom Sadu 1999. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Mikroelektronika odbranio je 2023. god.

**EFIKASNO UPRAVLJANJE PODACIMA U VEB APLIKACIJAMA UPOTREBOM
REACT QUERY BIBLIOTEKE****EFFICIENT DATA MANAGEMENT IN WEB APPLICATIONS USING THE REACT
QUERY LIBRARY**

Damjan Banjac, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu predstavljena je biblioteka React Query. Naglasak je na njenim osnovnim principima, mogućnostima primene, prednostima i nedostacima. Razmatrani su neki od glavnih aspekata ove biblioteke međukojima je, pre svega, omogućavanje efikasnog izvršavanja HTTP zahteva za dobijanje podataka sa servera. Njegova jednostavna i intuitivna sintaksa olakšava upravljanje asinhronim operacijama, a ugrađena podrška za keširanje podataka na klijentskoj strani omogućava brzo pristupanje i ažuriranje podataka uklanjanjem nepotrebnih zahteva ka serveru.

Ključne reči: React Query, veb aplikacija, upravljanje podacima, stanje aplikacije

Abstract – This paper introduces the React Query library, focusing on its fundamental principles, practical applications, advantages, and disadvantages. It discusses some of the key aspects of this library, highlighting its capability to efficiently execute HTTP requests for fetching data from the server. Its simple and intuitive syntax makes managing asynchronous operations easier, and built-in support for client-side data caching enables quick access and updates to data without the need for unnecessary server requests.

Keywords: React Query, web application, data management, application state

1. UVOD

U poslednjim decenijama veb aplikacije postaju sve važniji deo našeg svakodnevnog života. One nam pružaju različite usluge i sadržaj. S obzirom na rast kompleksnosti i količine podataka koje veb aplikacije obrađuju, performanse i efikasnost postaju ključni faktori za korisničko iskustvo. U isto vreme, upravljanje stanjem aplikacije, odnosno kako se podaci sinhronizuju između servera i klijenta, predstavlja izazov zbog potrebe za brzim i konzistentnim prikazom ažuriranih informacija.

Do sada primenjivani pristupi upravljanju stanjem, kao što su REST (*Representational State Transfer*) interfejs i biblioteke za upravljanje stanjem poput Redux biblioteke, suočavaju se sa ograničenjima i problemima u vezi sa performansama i složenošću koda.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Dunja Vrbaški, docent.

REST programski interfejsi često su dovodili do problema preuzimanja suviše mnogo ili premalo podataka (eng. *over-fetching*, *under-fetching*), što je rezultiralo nepotrebnim opterećenjem mreže i dužim vremenom odziva aplikacije.

Da bi se rešili ovih problema, pojavila se biblioteka React Query kao moćno i jednostavno rešenje za optimizaciju performansi i upravljanje stanjem u React aplikacijama. React Query koristi tehnike keširanja podataka, paginacije i ranijeg preuzimanja kako bi se izbeglo nepotrebno preuzimanje i obrada podataka sa servera. Ova biblioteka takođe pruža koncizan API za slanje zahteva serveru i upravljanje stanjem čime se smanjuje nepotrebnih ponavljanja kod i povećava efikasnost aplikacije.

U ovom radu se istražuje i analizira verzija 3 TanStack React Query biblioteke, pri čemu se ističu njeni ključni prednosti i nedostaci u poređenju sa tradicionalnim pristupima upravljanju stanjem i performansama. Demonstrirane su praktične funkcionalnosti i primene ove biblioteke u stvarnom okruženju.

2. TEORIJSKA OSNOVA

Veb aplikacija se sastoji iz dva ključna dela, takozvani: *front-end* i *back-end*. Prednji deo se odnosi na korisnički interfejs, uključujući HTML, CSS i JavaScript, koji korisnici vide i koriste za interakciju. Zadnji deo, sa druge strane, je skrivena infrastruktura koja radi iza scene. Ona se sastoji od serverskih komponenta i programa napisanih u jezicima kao što su: Python, Ruby, PHP, Java i Node.js. On obrađuje zahteve korisnika, komunicira sa bazom podataka i štiti podatke.

Stanje aplikacije, koje podrazumeva *client state* i *server state*, je ključni koncept u upravljanju podacima u veb aplikacijama. Stanje na klijentskoj strani, obuhvata podatke čuvane na uređaju korisnika, poput kolačića (eng. *cookies*) ili lokalnog skladišta (eng. *local storage*), za pamćenje preferencija i sesija. Stanje na serverskoj strani podrazumeva podatke na serveru ili u bazi podataka, uključujući informacije o korisnicima, proizvodima i drugim aplikacijskim podacima [1].

Stanje u veb aplikacijama obuhvata informacije i podatke koji se održavaju tokom interakcije korisnika s aplikacijom, ključno utičući na ponašanje i usluge aplikacije.

HTTP (*Hypertext Transfer Protocol*) je protokol za razmenu podataka između klijenata i veb servera putem

interneta. Radi na principu zahteva i odgovora gde klijent šalje zahtev za resursima, a server odgovara statusnim kodom, zaglavljima i telom odgovora. HTTP često koristi TCP/IP sloj za pouzdanu komunikaciju između klijenta i servera (Slika 1).



Slika 1. Princip rada HTTP [11]

3. UPOZNAVANJE SA REACT QUERY BIBLIOTEKOM

U nastavku su ukratko predstavljene funkcionalnosti biblioteke React Query.

3.1. Instalacija React Query

React Query biblioteka se može instalirati korišćenjem npm ili yarn komandi:

- npm install react-query,
- yarn add react-query [2].

3.2. Osnovni koncepti React Query biblioteke

React Query je biblioteka za upravljanje stanjem i za dobavljanjem podataka u React aplikacijama. Osnovni koncepti ove biblioteke su:

- Upit (*Query*) predstavlja zahtev za dobijanje podataka sa servera.
- Mutacije (*Mutation*) su zahtevi za promenom podataka na serveru, kao što su kreiranje, ažuriranje ili brisanje podataka [3].
- Funkcija upita (*Query Function*) je funkcija koja definiše logiku za izvršavanje upita.
- *UseQuery hook* se koristi u React komponentama kako bi se izvršio upit i dobili podaci.
- *UseMutation hook* se koristi za izvršavanje mutacija.
- Keširanje podataka je tehnika čuvanja kopije podataka u brzom i privremenom skladištu, odnosno u keš memoriji.
- Invalidacija keša se odnosi na automatsko osvežavanje podataka u keš memoriji nakon izvršavanja mutacija kako bi se odražavale promene na serveru [2].

3.3. Razmatranje ključnih funkcionalnosti

React Query pruža različite mogućnosti za izvođenje zahteva ka serveru i keširanje podataka kako bi se poboljšala efikasnost i performanse upravljanja podacima u React aplikacijama.

Najbitnije funkcionalnosti koje nudi React Query su izvođenje zahteva prema serveru, paginacija i vraćanje unapred.

Izvođenje zahteva prema serveru se može praktikovati pomoću takozvanih udica (eng. *hook*). Neke od tih udica su: *useQuery Hook*, *useMutation Hook* i *useInfiniteQuery Hook* [4].

Operacija za keširanje podataka koje podržava React Query su: automatsko keširanje, kontrola keša i invalidacija keša. React Query automatski kešira podatke preuzete putem *useQuery* i *useInfiniteQuery*. To znači da će svaki sledeći put, kada se isti upit izvrši, podaci biti brzo preuzeti iz keša umesto slanja novog zahteva serveru. Pod terminom kontrola keša se misli na različite opcije za kontrolu keš memorije i njenih podataka. Na primer, može se postaviti opcija *staleTime* čime se određuje koliko dugo podaci ostaju važeći pre nego što se ponovo preuzimaju sa servera. Takođe, može se koristiti i opcija *refetchInterval* kako bi se podaci u keš memoriji automatski osvežavali u redovnim intervalima. Invalidacija keša je proces ručnog osvežavanja podataka kako bi se odražavali satnje i preuzeli novi podaci sa servera. React Query omogućava invalidaciju keša pozivom metode *queryClient.invalidateQueries()* [4].

Paginacija, straničenje, je tehnika koja omogućava prikazivanje velikog broja podataka u sekcijama ili delovima, odnosno stranicama. U React Query biblioteci paginacija se može izvršiti pomoću *useInfiniteQuery* i *usePaginatedQuery* [5].

Preuzimanje unapred (eng. *prefetching*) je tehnika u veb aplikacijama koja omogućava preuzimanje podataka sa servera pre nego što ih korisnik eksplicitno zatraži. Ova tehnika ima za cilj poboljšanje performansi i korisničkog iskustva smanjenjem vremena čekanja na podatke tokom interakcije sa aplikacijom [6]. Iako je ovo korisna tehnika sa potencijalom za značajno poboljšanje rada i funkcionalnosti veb aplikacija, neki od potencijalnih problema koje treba razmotriti su nepotrebno opterećenje servera i mreže, povećana potrošnja resursa, zastareli podaci i složenost implementacije.

4. POREĐENJE SA DRUGIM BIBLIOTEKAMA

Kada se pričalo o upravljanju stanjem podataka u veb aplikacijama, pre pojave React Query biblioteke, najpopularnija rešenja za ovaj problem nudili su: Redux i Apollo Client.

U poređenju sa Redux bibliotekom, React Query predstavlja bolji alat za upravljanje stanjem podataka, a glavne prednosti su to što ima manje šablonskog, ponavljajućeg koda, što implicira da je jednostavniji za integraciju u projekte. Takođe, ima mogućnost automatskog keširanja za razliku od Redux biblioteke koja ne podržava tu opciju [7].

Kada je u pitanju Apollo Client, on je napravljen za rad isključivo sa GraphQL projektima, pa konkretno zbog toga React Query predstavlja multifunkcionalniju i više prilagodljivu opciju za bilo koju vrstu projekta [8].

5. OPTIMIZACIJA PERFORMANSI

Automatska optimizacija upita u React Query ima za cilj da pojednostavi i poboljša performanse rada sa asinhronim podacima, omogućavajući programerima da se fokusiraju na razvoj komponenta umesto na kompleksno upravljanje stanjem i keširanjem. Pod automatske optimizacije upita u React Query se podrazumevaju: automatsko keširanje i osvežavanje, poništavanje keša nakon mutacija, optimističko ažuriranje i optimizacija preuzimanje novih strana [9].

Takođe, React Query nudi mogućnost ručnog podešavanja keširanja za specifične upite ili mutacije ako to zahteva aplikacija. Keš memorija se može ručno podesiti pomoću metoda: *Invalidate*, *Refetch* i *Manual Query Creation*.

Metoda *Invalidate* je moćna metoda koja omogućava ručno poništavanje keširanih podataka za određeni upit [10]. Metoda *Refetch* omogućava ručno ponovno izvršavanje upita i osvežavanje keširanih podataka. Metoda *Manual Query Creation* omogućava precizno ručno podešavanje opcija keš memorije.

6. REAKTIVNOST I PRILAGODLJIVOST

Praćenje stanja upita i promena u React Query biblioteci je ključno za razumevanje trenutnog statusa izvršavanja upita i reagovanje na promene u trenutnom stanju. Stanja se prate na nekoliko načina korišćenjem različitih komandi:

- Korišćenjem: *isLoading*, *isError*, *isSuccess* [4];
- Korišćenjem: *onSuccess*, *onError* i *onSettled* opcija;
- Upotrebom: *useQueryClient* za praćenje promena
- Komandom: *useIsFetching* za praćenje aktivnih upita

Rad sa podacima u realnom vremenu uz React Query biblioteku omogućava brzo osvežavanje podataka. Koristi se za prikazivanje ažuriranih podataka bez ručnog osvežavanja stranice. React Query podržava ovo kroz invalidaciju keša i procesima udruživanja (eng. *polling*).

Korišćenje parametara za dinamičke upite u React Query biblioteci predstavlja koristan mehanizam za prilagodljivo i fleksibilno preuzimanje podataka sa servera ili drugih izvora. Umesto statičkog definisanja upita sa fiksnim vrednostima, ovaj pristup omogućava generisanje upita na osnovu dinamičkih parametara čime se omogućava višestruka upotreba upita u različitim situacijama.

Dodatno, upiti koji su parametrizovani olakšavaju održavanje koda jer, umesto kreiranja više upita za svaku moguću varijaciju, može se koristiti isti upit uz promenu vrednosti parametara. To omogućava ponovno iskoristive i prilagodljive upite, smanjuje dupliranje koda i olakšava prilagođavanje promenama u zahtevima.

Upravljanjem sa više paralelnih upita u React Query biblioteci omogućava efikasno koordiniranje i izvršavanje više nezavisnih upita ka serveru ili drugim izvorima podataka istovremeno čime se postiže brži prikaz informacija i bolje performanse aplikacije. Ova funkcionalnost je od velikog značaja u situacijama gde su podaci međusobno nezavisni i mogu se preuzimati nezavisno jedan od drugog čime se smanjuje ukupno vreme čekanja.

7. INTEGRACIJA SA DRUGIM TEHNOLOGIJAMA

Integracija React Query biblioteke sa React bibliotekom i komponentama omogućava efikasno upravljanje stanjem i preuzimanjem podataka unutar React aplikacija. Ova integracija omogućava: prilagođavanje performansi, unapređenje korisničkog iskustva i jednostavnij dalji razvoj.

Korišćenje React Query biblioteke sa React Router komponentom omogućava dalju, bolju implementaciju i upravljanje stanjem kao i preuzimanje podataka unutar aplikacije koja koristi rutiranje. Integracija ovih dveju biblioteka pruža mogućnost dinamičkog upravljanja podacima na osnovu trenutne rute omogućavajući još precizniji prikaz i ažurnije informacije. Kroz upotrebu *useQueryClient* i *useNavigate* hukova, koje pruža React Router, mogu se efikasno povezati akcije korisnika sa upitima i promenama u keširanim podacima.

Autentikacija i autorizacija uz pomoć React Query biblioteke pružaju koristan način za upravljanje sigurnošću i pristupom u React aplikacijama. React Query omogućava efikasnu implementaciju mehanizama autentikacije i autorizacije kako bi se zaštitili resursi i omogućio pristup određenim elementima aplikacije samo ovlašćenim korisnicima.

8. PRIMENA REACT QUERY U REALNIM APLIKACIJAMA

U nastavku su navedeni primeri tipova aplikacija u kojima se mogu primeniti funkcionalnosti React Query biblioteke čime se mogu poboljšati njihove performanse.

U aplikacijama za društveno umrežavanje se React Query može primeniti za:

- prikaz informacija na vremenskoj liniji,
- upravljanje profilima korisnika,
- obaveštenja u realnom vremenu,
- pretragu i filtriranje sadržaja
- beskonačno listanje
- optimizaciju performansi.

U sličnom kontekstu, kao i kod aplikacija za društveno umrežavanje, React Query nudi i podršku za aplikacije iz oblasti e-trgovine, i to za sledeće aktivnosti:

- dinamičko preuzimanje podataka o proizvodima,
- pretragu i filtriranje proizvoda ili usluga,
- straničenje za pregled kataloga,
- kreiranje korpe i liste želja,
- personalizovane predloge proizvoda.

Veoma velik potencijal za primenu React Query ima i kod aplikacija za rezervaciju putovanja jer kroz kombinaciju brzog preuzimanja i keširanja informacija o destinacijama, smeštaju, transportu i cenama, React Query omogućava aplikacijama da pruže korisnicima trenutni i tačan pregled dostupnih opcija za putovanja.

9. ZAKLJUČAK

React Query omogućava efikasno izvršavanje zahteva za preuzimanje i ažuriranje podataka sa servera. Ugrađena podrška za keširanje podataka na klijentskoj strani omogućava brz pristup i smanjenje broja i veličine nepotrebnih zahteva prema serveru. Time se smanjuje opterećenje na serveru i poboljšava se brzina odziva aplikacije. Karakteristike React Query biblioteke, istražene i testirane kroz ovaj rad, ukazuju na to da React Query može biti ključni alat za razvoj modernih veb aplikacija.

10. LITERATURA

- [1] Florin Software Consulting, „Server State vs Client State“, <https://www.florinsoftwareconsulting.com/blog/server-vs-client-state> (pristupljeno u avgustu 2023.)
- [2] Edidiong Etok, „Core Concepts of React Query“, <https://blog.devgenius.io/core-concepts-of-react-query-567c91035331> (pristupljeno u avgustu 2023.)
- [3] Tanstack Query official documentation, „Mutation“, <https://github.com/TanStack/query/blob/main/docs/react/guides/mutations.md> (pristupljeno u avgustu 2023.)
- [4] Tanstack Query official documentation <https://tanstack.com/query/v3/docs/react/guides/> (pristupljeno u avgustu 2023.)
- [5] Blitz, „usePaginatedQuery“, <https://blitzjs.com/docs/use-paginated-query> (pristupljeno u avgustu 2023.)
- [8] React Query Firebase, „Prefetching“, <https://react-query-firebase.invertase.dev/firestore/prefetching>, (pristupljeno u avgustu 2023.)
- [9] Tien Nguyen, „React Query vs Redux: A Detailed Comparison for Developers“, <https://www.frontendmag.com/insights/react-query-vs-redux-comparison/> (pristupljeno u septembru 2023.)
- [10] Tien Nguyen, „React Query vs Apollo Client: Which One Should You Use?“, <https://www.frontendmag.com/insights/react-query-vs-apollo-client/> (pristupljeno u septembru 2023.)
- [11] HyperText Transfer Protocol https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/http_basics.html

Kratka biografija:



Damjan Banjac rođen je 14. novembra 1997. godine u Novom Sadu. Godine 2016. upisao je Fakultet tehničkih nauka, odsek Računarstvo i automatika. Oktobra 2020. godine je diplomirao. Iste godine upisao je master studije na Fakultetu tehničkih nauka u Novom Sadu, odsek Računarstvo i automatika, studijski program Elektronsko poslovanje. Master rad odbranio je 2023. godine.

PRIMENA KONVOLUCIONIH NEURONSKIH MREŽA ZA DETEKCIJU BOLESTI PNEUMONIJE NAD PACIJENTIMA**APPLICATION OF CONVOLUTIONAL NEURAL NETWORKS FOR DETECTION OF PNEUMONIA DISEASE IN PATIENTS**

Dimitrije Stojanov, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSTVO I AUTOMATIKA

Kratak sadržaj – U radu je prikazana primena konvolucionih neuronskih mreža u detekciji bolesti pneumonije kod pacijenata. Objašnjene su osnove neuronskih mreža i kako one funkcionišu. Detaljno je prikazan proces i tehnologije koje se koriste u obradi i analizi slika. Skup podataka, javno dostupan, sastoji se od rendgen snimaka grudnog koša i korišćen je u istraživanju. U radu su prikazane dve arhitekture konvolucionih neuronskih mreža i njihovi rezultati klasifikacije upoređeni su sa rezultatima drugih radova.

Ključne reči: Konvolucione neuronske mreže, veštačka inteligencija, obrada i analiza slika, klasifikacija slika.

Abstract – This paper shows the application of convolutional neural networks in detection of pneumonia in patients. Basics of neural networks and how they work are explained. The process and technologies used in image processing and analysis are shown in detail in this paper. The dataset, publicly available, consists of chest x-rays and was used in the research. The paper presents two architectures of convolutional neural networks and their classification results are compared with the results of other research.

Keywords: Convolutional neural networks, artificial intelligence, image processing and analysis, image classification.

1. UVOD

Pneumonija je zapaljenje tkiva u plućima uzrokovanom bakterijskom infekcijom. Svake godine milioni ljudi širom sveta završe u bolnici zbog pneumonije. Srećom, ova bolest može biti izlečena upotrebom antibiotika i antivirusnih lekova. Važno je da se pneumonija rano dijagnostikuje i leči kako bi se izbegle komplikacije koje mogu dovesti do smrti pacijenta. Rendgenski snimci pluća su najčešći način dijagnostikovanja pneumonije, ali čak i stručni radiolozi često imaju izazove u identifikaciji bolesti na tim snimcima. To je zbog toga što slika pneumonije može biti nejasna, lako se može pomešati s drugim bolestima i može izgledati kao benigna abnormalnost. Zbog ovih nedoslednosti, dijagnoza pneumonije varira među radiolozima.

Zato je potrebno razviti sistem podrške koji bi pomogao radiolozima u dijagnostikovanju pneumonije na osnovu rendgenskih snimaka pluća. Razvoj dubokog učenja,

posebno konvolucionih neuronskih mreža (CNN), pokazao je veliki uspeh u klasifikaciji slika. Glavna ideja iza CNN mreža je stvaranje veštačkog modela sličnog ljudskom mozgu. CNN mreže imaju sposobnost da izdvoje značajne karakteristike iz celokupne slike. One su se pokazale korisnim ne samo u rešavanju problema kompjuterskog vida, već i u medicini. Na primer, CNN mreže su uspešno korišćene za prepoznavanje raka dojke, detekciju tumora mozga, dijagnostikovanje Alchajmerove bolesti, klasifikaciju kožnih lezija i drugo. Implementacija ovakvog sistema mogla bi pomoći lekarima u dijagnostikovanju pneumonije. Trenutno, ovi modeli ne mogu zameniti ljudski faktor u dijagnozi, ali imaju potencijal da to jednog dana učine.

U ovom radu je prikazana primena konvolucionih neuronskih mreža za detekciju bolesti pneumonije kod rendgen snimaka pacijenata. Objašnjene su teorijske osnove neuronskih mreža i kako one funkcionišu. Detaljno su prikazane tehnologije i proces obrade i analize slika. U radu su prikazane dve arhitekture konvolucionih neuronskih mreža. Krenirani modeli su klasifikatori koji bi na osnovu priuštenih rendgen snimaka davali odgovor da li je pacijent oboleo od bolesti pneumonije ili je zdrav. Rezultati klasifikacije poređeni su sa rezultatima iz radova koji se bave istom ovom temom.

2. TEORIJSKI POJMOVI I DEFINICIJE**2.1. Veštačke neuronske mreže**

Neuronske mreže (NN) su efikasni sistemi za informaciono procesiranje koji se koriste za rešavanje problema poput aproksimacija funkcija, klasifikacije oblika, optimizacije i klasterizacije. Veštačke neuronske mreže (ANN) se neprestano razvijaju i poboljšavaju kako bi povećale tačnost i efikasnost u raznim primenama. Bez obzira na oblik, ulaz u ANN predstavlja niz vrednosti koje se obrađuju, a u daljem tekstu će biti objašnjeni osnovni koncepti dubokog učenja i arhitekture ANN [3].

2.2. Osnovne komponente veštačkih neuronskih mreža

Osnovni elementi neuronskih mreža su neuroni, veze, težine i funkcija propagacije.

Veštačke neuronske mreže simuliraju biološke neurone u mozgu i sastoje se od neurona koji primaju ulaze i prenose informacije drugim neuronima. Ulazi mogu biti podaci poput slika ili teksta, a izlaz poslednjeg neurona predstavlja krajnji rezultat kao klasifikacija ili prepoznavanje. Ulazni signali se množe sa težinama veza i obrađuju aktivacionom funkcijom kako bi se dobio izlaz.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila prof. dr Jelena Slivka.

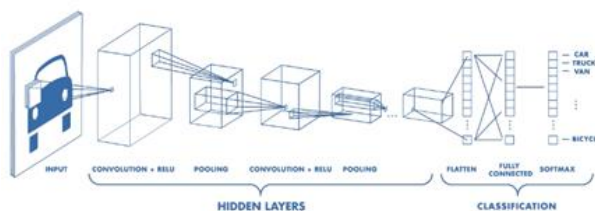
2.3. Vrste neuronskih mreža

Veštačke neuronske mreže koriste različite organizacije neurona kako bi se rešavali različiti zadaci. To uključuje različite strukture konekcija između neurona, poput veza sa svim neuronima u sledećem sloju ili samo sa određenim neuronima. Postoji više popularnih arhitektura, kao što su višeslojni perceptron, konvolutivne neuronske mreže, rekurentne neuronske mreže, LSTM, GRU.

2.4. Konvolucione neuronske mreže

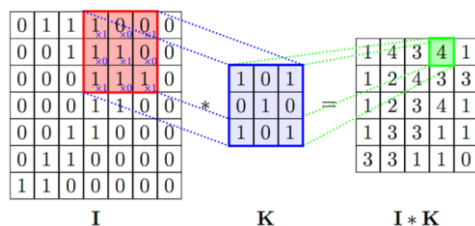
Arhitekture neuronskih mreža mogu biti potpuno povezane, gde su svi neuroni jednog sloja povezani sa svim neuronima sledećeg sloja, ili lokalno povezane, gde se koristi princip lokalne povezanosti neurona. Konvolucione neuronske mreže (CNN) su specifična vrsta mreže koja se često koristi u obradi i analizi slika. Ona omogućava efikasno prepoznavanje karakteristika i šablona u podacima, jer koristi organizaciju slojeva za tu svrhu. Kroz proces treniranja, težine konekcija se prilagođavaju kako bi mreža mogla da prepozna slike ili šablone koje joj se prezentuju.

Konvolucione neuronske mreže (CNN) se sastoje od nekoliko slojeva koji uključuju konvolucione slojeve, slojeve za udruživanje, potpuno povezane slojeve i izlazni sloj, slika 1. Ova arhitektura može se razlikovati u rasporedu i kombinaciji ovih slojeva, a to je ključna razlika u odnosu na višeslojne perceptrone [4].



Slika 1. Prikaz modela konvolucione neuronske mreže [8]

Konvolucionni sloj u neuronskoj mreži koristi lokalnu obradu podataka u dvo-dimenzionalnom formatu. Skriveni neuroni u ovom sloju koriste konvolucione kernele (filtre) koji se primenjuju na mali deo podataka iz prethodnog sloja. Ovo omogućava stvaranje mapa karakteristika za različite filtere u konvolucionom sloju.



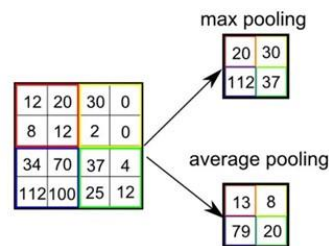
Slika 2. Prikaz konvolucionog sloja CNN mreže

Slojevi za udruživanje (pooling) koriste se za smanjenje dimenzionalnosti tako što grupišu susedne ćelije mape karakteristika [5]. Uobičajene operacije udruživanja uključuju maksimalno ili prosečno uzimanje vrednosti. Efekti pooling operacija prikazani su na slici 3.

3. METODOLOGIJA

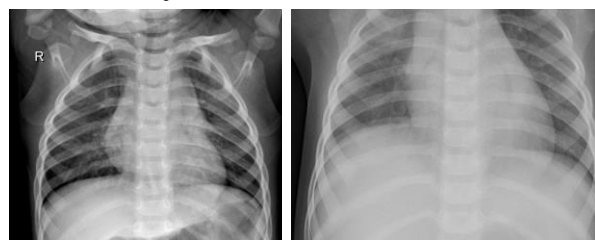
3.1. Skup podataka

Skup podataka sastoji se od 5856 slika raspoređenih u grupe za obuku, testiranje i validaciju.



Slika 3. Prikaz operacije pooling sloja CNN

Skup za obuku sadrži 5216 slika od kojih 3834 su renden snimci pluća pacijenata koji boluju od pneumonije a 1300 snimaka zdravih pacijenata. Skup podataka preuzet je sa sajta kaggle.com pod nazivom Chest X-Ray Images (Pneumonia). Podaci u obliku slika su različitih dimenzija. Slike su tipa JPEG fajla. Veličina fajlova varira od 50 KB do 2350 KB. Primer slika korišćenik za obuku i testiranje može se videti na slici 4.



Slika 4. Zdrava pluća pacijenta naspram pluća pacijenta obolelog od pneumonije

Primećuje se neuravnoteženost podataka između dve navedene klase tako da će se augmentacijom podataka, kasnije navedenom u radu, skup podataka za obuku izbalansirati. Navedene slike su različitih rezolucija i predprocesiranjem dovedene su sve na istu rezoluciju. Ulaz sistema predstavljaju trodimenzionalne matrice $224*224*3$. Za implementaciju sistema korišćena je konvoluciona neuronska mreža. Sistem je obučen za binarnu klasifikaciju slika, odnosno klasifikuje da li je pacijent čiji je rendgen snimak stigao na obradu obolio od bolesti pneumonije ili nije.

3.2. Augmentacija podataka

Augmentacija podataka je moćan alat koji se koristi za povećanje robustnosti modela, posebno kada su skupovi podataka neuravnoteženi. Ona uključuje primenu različitih transformacija nad originalnim slikama, kao što su rotacija, pomeranje, flipovanje, promena svetlosti i zumiranje. Klasa ImageDataGenerator u okviru Keras biblioteke omogućava jednostavno i brzo generisanje augmentiranih slika, čime se obezbeđuje veća varijabilnost u skupu podataka i bolja generalizacija modela. Glavna prednost ove klase je što pruža augmentaciju podataka u realnom vremenu tokom faze obuke, obezbeđujući da model dobije nove varijacije slika u svakoj epohi, a istovremeno štedeći memoriju učitavanjem slika u grupama.

3.3. Arhitektura neuronske mreže

Za implementaciju rešenja korišćena je konvoluciona neuronska mreža. Konvolucione neuronske mreže (CNN-ovi) postale su najnaprednija tehnologija za prepoznavanje slika i uspešno se primenjuju u širokom spektru zadataka kompjuterske vizije. Ključne prednosti upotrebe CNN-ova koje su dovele do odabira ove

arhitekture su: prostorna invarijantnost, hijerarhijsko predstavljanje, smanjenje parametrizacije, transfer učenje i visoka tačnost [7].

3.4. Funkcija gubitka

Funkcija gubitka koja je korišćena u istraživanju je Binary Cross-Entropy Loss. U mašinskom učenju, funkcija gubitka je matematička funkcija koja meri razliku između predviđenog izlaza modela i stvarnog izlaza. Cilj funkcije gubitka je da minimizuje ovu razliku tokom procesa obuke. Binary Cross-Entropy Loss je funkcija gubitka koja se koristi u problemima binarne klasifikacije.

3.5. Model

Konvolucione neuronske mreže se ističu kod obrade podataka koji imaju prostornu strukturu. Zato se najčešće koriste kod obrade i analize slika. U istraživanju korišćene su dve arhitekture konvolucione neuronske mreže. Obe arhitekture sačinjene su od ulaznih slojeva, konvolucionih slojeva (Conv2D), pooling sloja (MaxPooling2D), Flatten, Dense i Dropout sloja. Dve arhitekture se razlikuju u svojoj kompleksnosti.

4. PREGLED STANJA U OBLASTI

U rešenju [1] korišćen je isti skup podataka koji će se upotrebiti za izradu rešenja u ovom radu. Korišćene su već istrenirane konvolucione neuronske mreže, Xception i Vgg16 koje su obučavane na poznatom skupu podataka ImageNet. Kao metrike za evaluaciju korišćeni su tačnost, osetljivost, specifičnost, povrat, preciznost i F1 mera. Rezultati ove dve mreže prikazani su u tabeli 1.

Tabela 1. Rezultati istraživanja rada [1]

	Accuracy	Sensitivity	Specificity	Precision	Recall	F1 score
Xception	0.82	0.85	0.76	0.83	0.86	0.83
VGG16	0.87	0.82	0.91	0.89	0.88	0.88

U radu [2] takođe je korišćen isti skup podataka koji je korišćen za izradu rešenja u ovom radu. Karakteristike iz slike se izdvajaju korišćenjem modela neuronskih mreža unapred obučanih pomoću ImageNet. Pripremljeno je pet različitih modela i analizirane su njihove performanse, gde je peti model kombinacija rezultata svih prethodno obučanih modela. Obučeni modeli su dali sledeće rezultate, AlexNet je postigao tačnost od 92.86%, dok je ResNet18 pokazao najveće performanse od 94.23%. DenseNet121 (tačnost 92.62%), InceptionV3 (tačnost 92.01%) i GoogLeNet (tačnost 93.12%) takođe su pokazale odlične rezultate. Kombinovanjem ovih modela postignuta je najveća tačnost od 96.39%.

5. EKSPERIMENTI

5.1. Eksperiment 1

Prvi model CNN1 sačinjen je od ulaznog sloja, praćenim sa kombinacijom dva konvolucionna sloja i jednim pooling slojem. Konvolucionni slojevi sadrže po 64 filtera dimenzija 3x3 i aktivacionu funkciju relu.

Hiper-parametri u pooling sloju su postavljeni na 2, 2. Veličina 2x2 pomaže u smanjenju prostornih dimenzija ulaznih mapa obeležja za faktor 2, dok se i dalje zadržavaju najvažnije informacije. Na ova četiri sloja nadovezana su još dva konvolucionna sloja praćena sa još jednim pooling slojem. Konvolucionni slojevi ovog puta sadrže po 128 filtera takođe dimenzija 3x3 i aktivacionom

funkcijom relu. Pooling sloj ima veličinu 2x2. Sledeći korišćeni sloj je Flatten a potom kombinacija Dense i Dropout slojeva. Prvi Dense sloj sadrži 512 neurona sa aktivacionom funkcijom relu. Sledi Dropout sloj sa stopom opadanja od 0.7, što zači da postoji 70% šanse da neuron bude ispušten tokom obuke. Drugi Dense sloj se sastoji od 256 neurona sa takođe relu aktivacionom funkcijom. Dropout sloj koji sledi ima stopu padanja od 0.5 i nadovezuje se na poslednji Dense sloj sa 2 neurona i aktivacionom funkcijom softmax.

```
input_img = Input(shape=train_batch.image_shape, name='ImageInput')
x = Conv2D(64, (3, 3), activation='relu', padding='same')(input_img)
x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)
x = MaxPooling2D((2, 2))(x)
x = Conv2D(128, (3, 3), activation='relu', padding='same')(x)
x = Conv2D(128, (3, 3), activation='relu', padding='same')(x)
x = MaxPooling2D((2, 2))(x)
x = Flatten(name='flatten')(x)
x = Dense(512, activation='relu')(x)
x = Dropout(0.7)(x)
x = Dense(256, activation='relu')(x)
x = Dropout(0.5)(x)
x = Dense(2, activation='softmax')(x)
model = Model(inputs=input_img, outputs=x)
```

Slika 5. Arhitektura prvog modela CNN

5.2. Eksperiment 2

Drugi model CNN2 ima jednostavniju strukturu. Ulazni sloj praćen kombinacijama konvolucionih slojeva i pooling slojeva. Prvi konvolucionni sloj sastoji se od 32 filtera, veličinom kernel od 3x3 i aktivacionom funkcijom relu. Sledi MaxPooling2D sloj veličine 2x2. Drugi konvolucionni sloj sadrži 64 filtera i ima istu veličinu i aktivacionu funkciju kao i prvi sloj. Drugi pooling sloj identičan je prvom. Poslednji Conv2D sloj takođe sadrži 64 filtera veličinu kernela od 3x3 i relu aktivacionu funkciju. Sledi Flatten sloj i dva Dense sloja. Prvi Dense sloj sadrži 64 neurona sa relu aktivacionom funkcijom. Drugi Dense sloj se sastoji od dva neurona i softmax kao aktivacionu funkciju. Primer druge arhitekture konvolucione neuronske mreže prikazan je na slici 6.

```
input_img = Input(shape=train_batch.image_shape, name='ImageInput')
x = Conv2D(32, (3, 3), activation='relu', padding='same')(input_img)
x = MaxPooling2D((2, 2))(x)
x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)
x = MaxPooling2D((2, 2))(x)
x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)
x = Flatten(name='flatten')(x)
x = Dense(64, activation='relu')(x)
x = Dense(2, activation='softmax')(x)
model = Model(inputs=input_img, outputs=x)
```

Slika 6. Arhitektura drugog modela CNN

5.3. Evaluacija

Korišćene su metrike koje rade evaluaciju rezultata tačnosti klasifikatora. Da bi pravilno izvršili evaluaciju modela uvode se četiri pojma koji karakterišu rezultate klasifikacije: stvarni pozitiv, stvarni negativ, lažni pozitiv, lažni negativ. Pomoću navedenih izračunatih vrednosti dobijamo neke od metrika koji se koriste za evaluaciju modela za klasifikaciju. Korišćeni metriki u ovom radu su: tačnost, preciznost, povrat i F1 mera [6].

6. REZULTATI I DISKUSIJA

Rezultati metrika evaluacije binarnog klasifikatora prikazani su u tabelama ispod. U tabeli 2 izdvojena je tačnost dva modela nad podacima iz skupa za obuku i nad podacima skupa za evaluaciju. Tabela 3 predstavlja rezultate metrika poput preciznosti, povrata i F1 mere.

Tabela 2. Tačnost dva modela obučavana u radu

	Test data		Validation data	
	Accuracy	Loss	Accuracy	Loss
Model 1	0.80	0.12	0.87	0.03
Model 2	0.82	0.22	0.92	0.08

Tabela 3. Vrednosti preciznosti, povrata i F1 mere

	Precision	Recall	F1 score
Model 1	0.73	0.98	0.83
Model 2	0.80	0.97	0.87

Model sa jednostavnom arhitekturom pokazao je veću tačnost na skupu podataka za testiranje i nad validacionim podacima. Kompleksniji CNN modeli obično pokazuju bolje performanse u klasifikaciji. Složeniji modeli mogu naučiti složenije šablone u podacima. Oni imaju veći kapacitet da detektuju suptilnije karakteristike i varijacije među različitim podacima ali su skloniji natprilagođenju što pretpostavljamo da je dovelo do lošijih rezultata kod prvog modela. Jednostavnije arhitekture CNN-a obično imaju manje parametara i manju sposobnost za učenje složenih šablona. Manje parametara pomaže oko sprečavanja prenaučivosti i tako poboljšava performanse sistema.

Tačnost klasifikacije test podataka i validacionih podataka pokazala se kao veoma uspešna. Loss vrednost se pokazala niskom za ova dva modela, ispod 0.1 validacione podatke i oko 0.2 za test podatke. Ove vrednosti pokazuju da istrenirani modeli nisu preprilagođeni ili pretrenirani. U radu [1] za model Xception vrednost preciznosti iznosi 0.83, povrata 0.86 i F1 mere 0.83. Za model Vgg16 vrednost za preciznost iznosi 0.89, povrata 0.88 i F1 mere iznosi 0.88. Tačnost Xception modela iznosi 0.82 a Vgg16 modela 0.87. Poredeći navedene rezultate sa rezultatima dobijenim u ovom radu vidimo dosta sličnosti. Vgg16 model je pokazao uspešnijim od dva trenirana modela dok je Xception pokazao slične performanse. Rezultati u radu [2] pokazali su mnogo bolje performanse nego dva modela korišćena u ovom radu.

Svih 5 navedenih modela postigli su tačnost od preko 90%. Najveću tačnost je imao ResNet18 od 94.23% dok je GoogleNet imao preko 93% a AlexNet, DenseNet121 i InceptionV3 imali su oko 92%. Visoka tačnost u klasifikaciji može se objasniti korišćenjem već obučanih neuronskih mreža pomoću ImageNet skupa podataka. Kombinacija ovih pet pređa koja je takođe objašnjena u radu [2] dala je tačnost od preko 96%.

Rezultati klasifikacije dobijeni iz modela koji prelaze 90% mogu se smatrati veoma uspešnim i kao takve mogu se koristiti i u polju medicine. Modeli obučeni u ovom radu ne pokazuju dovoljno dobre performanse klasifikacije ovakvih slika da bi potpuno zamenili ljudski faktor ali mogli bi se smatrati podrškom lekarima tokom dijagnostikovanja. Bolje obučanim modelima treniranim na najsavremenijem hardveru mogu se postići bolji rezultati i smanjiti ljudska greška prilikom pregleda pacijenata i dijagnostikovanja raznih bolesti. Modeli obučavani u ovom radu pokazali su određeni stepen greške u klasifikaciji.

Analizirajući skup podataka može se primetiti dosta abnormalnosti kod pacijenata koji su oboleli od pneumonije. Ove razlike među podacima iste klase

klasifikator može tretirati kao šum i da to dovede do pogrešne klasifikacije. Naš cilj je da se što više smanji broj lažno negativnih primera klasifikacije jer su oni najveća pretnja za zdravlje pacijenta.

7. ZAKLJUČAK

U ovom radu predstavljen je sistem za klasifikaciju rendgen snimaka grudnog koša pacijenata. Klasifikator je na osnovu snimka davao rezultat, da li je pacijent oboleo od bolesti pneumonije. Ovakvi modeli pokazuju veliki potencijal u medicini kao način dijagnostikovanja ne samo bolesti pneumonije već i mnogih drugih bolesti. Sistemi predstavljeni u ovom radu oslanjaju se na modele konvolucionih neuronskih mreža.

U budućnosti, sa boljim hardverom i bogatijim skupovima podataka, rezultati klasifikacije mogu se još više unaprediti. Daljim napretkom veštačke inteligencije i medicinske tehnologije, očekuje se da će neuronske mreže postati još snažniji alat za podršku zdravstvenim stručnjacima u dijagnostikovanju različitih bolesti.

8. LITERATURA

- [1] Ayan, Enes, and Halil Murat Ünver. "Diagnosis of pneumonia from chest X-ray images using deep learning." *2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT)*. IEEE, 2019.
- [2] Chouhan, Vikash, et al. "A novel transfer learning based approach for pneumonia detection in chest X-ray images." *Applied Sciences* 10.2 (2020): 559.
- [3] Sze, Vivienne, et al. "Efficient processing of deep neural networks: A tutorial and survey." *Proceedings of the IEEE* 105.12 (2017): 2295-2329.
- [4] Sang, Jonghee, Soomyung Park, and Junwoo Lee. "Convolutional recurrent neural networks for urban sound classification using raw waveforms." *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018.
- [5] Hijazi, Samer, Rishi Kumar, and Chris Rowen. "Using convolutional neural networks for image recognition." *Cadence Design Systems Inc.: San Jose, CA, USA* (2015): 1-12.
- [6] Goutte, Cyril, and Eric Gaussier. "A probabilistic interpretation of precision, recall and F-score, with implication for evaluation." *European conference on information retrieval*. Springer, Berlin, Heidelberg, 2005.
- [7] Alzubaidi, Laith, et al. "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions." *Journal of big Data* 8 (2021): 1-74
- [8] <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

Kratka biografija:



Dimitrije Stojanov rođen je 23.02.1998. godine u Novom Sadu. Osnovne akademske studije završio je 2020 godine na Fakultetu tehničkih nauka nakon kojih upisuje master akademske studije na usmerenju Inteligentni isistemi, studentski program Primjenjene računarske nauke. kontakt: mitadima@gmail.com

SISTEM ZA ZAKAZIVANJE ČASOVA I PRAĆENJE RUTA U AUTO-ŠKOLI

SYSTEM FOR APPOINTMENT BOOKING AND ROUTE TRACKING FOR DRIVING SCHOOLS

Tamara Jančić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Tema ovog rada jeste implementacija veb i mobilne aplikacije za auto-škole, čije su glavne funkcionalnosti zakazivanje termina časova i ispita, kao i praćenje ruta na časovima vožnje.

Ključne reči: veb, mobilna aplikacija, C#, .NET, React, React Native

Abstract – The thesis deals with development of web and mobile application for driving schools, with the focus on scheduling of driving lesson and exam appointments as well as tracking routes during driving lessons.

Keywords: web, mobile application, C#, .NET, React, React Native

1. UVOD

Veb i mobilne aplikacije predstavljaju odgovor modernog doba na svakodnevne probleme i pomoć pri izvršavanju brojnih procesa. Cilj opisanih aplikacija jeste olakšanje komunikacije i podele informacija između studenta, instruktora i ispitivača prilikom obavljanja njihovih dužnosti u auto-školi. Veb aplikacija omogućava zakazivanje časova vožnje i ispita, kad su ostvareni uslovi za to, dok zahvaljujući mobilnoj aplikaciji student u svakom momentu ima pristup istoriji ruta kojima je vozio tokom časova. U okviru rada će biti opisane specifikacije ovog sistema, tehnologije koje su korišćene, kao i moguće ideje za dalji razvoj.

2. PREGLED SLIČNIH SISTEMA

Pre detaljnijeg navođenja funkcionalnosti i implementacije opisanog sistema, biće izvršena analiza sličnih sistema. Glavni kriterijumi na osnovu kojih se izdvajaju navedeni sistemi su mogućnost zakazivanja časova, pregled napretka studenta, praćenje ruta tokom časova vožnje, obaveštenja, kao i tehnički detalji aplikacije kao što je dostupnost na različitim platformama. Imajući u vidu da tržište aplikacija za auto-škole nije veliko, kompanije retko prave aplikacije namenjene auto-školama, već im je isplativija opcija da naprave sistem šire namene, koji se može prilagođavati različitim oblastima. Jedan od takvih sistema je Teachworks [1], dizajniran prvenstveno za obrazovne ustanove. Iako postoji verzija prilagođena auto-školama,

nije integrisana mapa ni praćenje ruta studenata. Još jedan od sistema pogodan za zakazivanje termina časova i ispita, ali takođe bez praćenja ruta je Drive Scout [2]. Za razliku od dva opisana softvera, Samsara [3] je softver čija je primarna uloga praćenje ruta, međutim više je fokusiran na brigu o vozilima i sigurnost učesnika u saobraćaju. Zakazivanje termina je deo koji nedostaje ovom sistemu da bi bio odgovarajući izbor za potrebe auto-škole.

3. OPIS KORIŠĆENIH TEHNOLOGIJA

Za implementaciju serverskog dela aplikacije korišćen je programski jezik C# [4] uz pomoć .NET [5] okruženja. Skladištenje podataka je odrađeno uz pomoć MSSQL (Microsoft SQL Server) objektno-relacionog sistema za upravljanje bazama podataka. Klijentski deo veb aplikacije razvijen je koristeći React [6] biblioteku JavaScript jezika za izradu interaktivnih veb aplikacija. Mobilna aplikacija implementirana je uz pomoć React Native [7] biblioteke, takođe bazirane na JavaScript jeziku, koja omogućava razvoj mobilne aplikacije koristeći sličan kod kao kod veb aplikacija.

3.1 C#

C# je objektno-orijentisani jezik, razvijen od strane kompanije Microsoft, koji se izvršava u okviru .NET okruženja. Podržan je na različitim platformama uz pomoć .NET Core i .NET Framework radnih okvira. C# spada u familiju jezika i programeri mogu primetiti njegove sličnosti sa drugim jezicima kao što su C, C++ i Java.

3.2 .NET

Dok je C# programski jezik, .NET predstavlja okruženje koje izvršava kod napisan u C# jeziku. Osim koda napisanog u C#-u, .NET može izvršavati kod napisan i u drugim jezicima, dok je C# napravljen da bude izvršavan u .NET okruženju. S obzirom da je .NET razvijen od strane Microsoft kompanije, prvo je razvijen .NET Framework kao platforma namenjena za Windows operativni sistem. Kasnije, 2016. godine, razvijena je alternativna platforma .NET Core koja je kompatibilna i sa drugim operativnim sistemima. Trenutna verzija .NET 6 ujedinjuje sve mogućnosti ovih platformi i omogućava razvijanje više tipova aplikacija za različite platforme.

3.3. React

React je JavaScript biblioteka za razvoj klijentskih aplikacija koja obezbeđuje pregled podataka zapisanih u

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Vidaković, red. prof.

HTML-u. Razvoj aplikacija se zasniva na upotrebi komponenti i smanjenju zavisnosti u kodu.

3.4 React Native

React Native je radni okvir za razvoj mobilnih aplikacija baziran na JavaScript kodu. Bitna karakteristika ovog radnog okvira je mogućnost pisanja samo jedne verzije koda koja može biti upotrebljena i na Android i na iOS operativnim sistemima. Zbog svoje sličnosti React-u, na kome je i baziran, stekao je veliku popularnost među programerima.

4. SPECIFIKACIJA

4.1 Veb aplikacija

Prvi deo zadatka obuhvata izradu veb stranice sa brojnim korisnim funkcionalnostima namenjenim članovima auto-škole. Kao najvažniji proces, izdvaja se mogućnost zakazivanja časova vožnje i termina polaganja.

Svaki korisnik može videti sve auto-škole iz sistema, pretražiti ih po imenu i prikazati njihove detaljnije informacije koje uključuju i listu instruktora te škole. Neregistrovan korisnik ima mogućnost prijave na sistem ili registracije kao student, ukoliko već nema svoj profil. Prilikom registracije, korisnik bira auto-školu i kategoriju za koju će biti obučavan. Nakon traženja zahteva za registraciju, potrebno je da verifikuje profil uz pomoć imejla koji mu je stigao na njegovu adresu i nakon toga može da se prijavi na sistem. Instruktor i ispitivač na sistem dodaje admin. Dodavanje i brisanje auto-škole je posao admina. Takođe, on može i da ažurira podatke svake auto-škole.

Svi prijavljeni korisnici mogu da pristupe stranici koja prikazuje njihov profil i na tom mestu mogu da ažuriraju svoje podatke. Admin je jedini korisnik koji ima pristup informacijama svih korisnika. On može i da obriše svakog korisnika iz sistema. Student na svom profilu, pored standardnih informacija, može videti i informacije o rezultatu ispita teorije, broj preostalih časova vožnje i broj izlazaka na polaganje ispita vožnje. Student na svom profilu bira instruktora, iz liste instruktora koji rade u auto-školi čiji je on član i koji obučavaju za kategoriju za koju on vrši obuku. On može i promeniti svog instruktora, biranjem novog iz iste liste.

Zakazivanje časova je moguće za svakog studenta koji je položio teoriju i izabrao svog instruktora. Svaki korisnik ima pristup kalendaru koji prikazuje relevantne informacije za njega. Student vidi raspored njegovog instruktora sa zauzetim i slobodnim terminima. Zakazivanje časa u terminu kad već postoji neki drugi čas je onemogućeno, kao i zakazivanje časa ili ispita u prošlosti. Izborom slobodnog termina i tipa časa, student kreira zahtev koji instruktor može da odobri ili odbije. Tip časa vožnje može biti poligon, grad, autoput, motoput ili slobodna vožnja.

Instruktor na kalendaru vidi sve svoje časove, potencijalne i odobrene. On može da odobri ili odbije čas. Ukoliko je čas odobren, termin tog časa na kalendaru menja boju i postaje uočljiviji, radi lakšeg pregleda. Ukoliko je čas odbijen, tada je taj termin oslobođen, odbijeni čas nestaje sa prikaza. Student i instruktor mogu

u svakom momentu do početka termina, da ga ponište. Tada je taj termin oslobođen, isto kao i kod odbijanja časa. Uvedena su obaveštenja putem imejla, student je obavešten prilikom odobravanja časa o vremenu održavanja, a prilikom odbijanja, obavešteni su i student i instruktor. Nakon odrađenih obaveznih 40 časova vožnje, student ima opciju zakazivanja dodatnih časova, ako smatra da je to potrebno.

Ispitivač može da zakaže termin za polaganje ispita. Prilikom zakazivanja termina, bira auto-školu, a zatim studenta iz liste studenata te škole koji su ostvarili uslove za izlazak na ispit, a to je odrađivanje svih preostalih časova vožnje. Nakon zakazivanja ispita, student može da odobri ili odbije termin. Student može da zakazuje časove vožnje sve do termina ispita. U slučaju da student nije položio ispit, on ponovo dobija mogućnost zakazivanja časova i ispitivač može ponovo da mu zakaže ispit. I pri potvrđivanju termina ispita, student je obavešten putem imejla sa detaljnim informacijama. Odobren termin može biti uklonjen od strane studenta i ispitivača i u tom slučaju novi termin može biti zakazan.

Još jedna važna uloga ispitivača jeste unošenje rezultata ispita. Nakon što termin ispita prođe, ispitivač je dužan da unese rezultat. Student je obavešten putem imejla i prikazom detaljnih informacija o ispitu na kalendaru. Nakon što student položi ispit vožnje, on završava obuku i samim tim više nema mogućnost zakazivanja časova.

4.2 Mobilna aplikacija

Drugi deo zadatka predstavlja izradu mobilne aplikacije baziran na praćenju ruta kojima student vozi na časovima. Mobilna aplikacija je namenjena instruktorima koji beleže rute i studentima koji te rute mogu da pregledaju.

Određene funkcionalnosti veb aplikacije dostupne su i na mobilnoj, kao što su prijava na sistem, registracija, pregled profila korisnika i pregled auto-škola i njihovih detaljnih informacija.

Ono šta izdvaja mobilnu aplikaciju je mogućnost beleženja ruta. Instruktor je zadužen da označi početak časa i tako započne beleženje putanje. On to radi tako što izabere studenta koji trenutno ima čas, iz liste svojih studenata, otvara mapu i klikće na dugme koje označava početak časa. Tokom pomeranja, na mapi se isertava putanja u realnom vremenu. Nakon što je čas završen, student beleži kraj časa i tada putanja ostaje zabeležena, zajedno sa datumom i vremenom tokom kog je čas trajao. Pregled odrađenih ruta je moguć od strane studenta i instruktora. Instruktor ima mogućnost pregleda ruta svih svojih učenika, dok svaki student vidi samo svoje rute.

4.3 Nefunkcionalni zahtevi

Imajući u vidu da je ovo sistem koji može sadržati osetljive podatke, kao što su lični podaci svih korisnika, potrebno je obezbediti određen nivo sigurnosti. Autentifikacija i autorizacija su neophodan deo svake aplikacije. Autentifikacija je proces validacije korisnika, a autorizacija je proces provere korisnikovih permisija u aplikaciji. .NET ima biblioteke koje omogućavaju njihovu implementaciju i korišćene su u ovom radu.

Obaveštenja korisnicima, koja se šalju na imejl adresu su implementirana uz pomoć spoljašnjeg API-ja SendGrid.

Neophodan deo mobilne aplikacije, prikazivanje i beleženje puta, omogućeno je integracijom sa Google mapama. Integracija je moguća uz pomoć biblioteke React radnog okvira, namenjenih za rad sa mapama.

Dostupnost kalendara na kom je moguće dodati termine i prikaz istih je neophodan deo opisane veb aplikacije. Ovaj zahtev je ostvaren upotrebom „*react-big-calendar*“ komponente.

Rad mobilne aplikacije na različitim operativnim sistemima (Android i iOS), omogućen je zahvaljujući React Native-u, koji podržava izvršenje istog koda na njima. Isto tako, bilo je neophodno obezbediti rad veb aplikacije na različitim operativnim sistemima i pretraživačima, što se podrazumeva za aplikacije rađene uz pomoć React-a.

5. IMPLEMENTACIJA

Kao što je već navedeno u tekstu, sistem se sastoji iz tri celine, serverske aplikacije i dve klijentske aplikacije: veb i mobilne. Serverski deo sistema je implementiran u C# programskom jeziku uz pomoć .NET radnog okvira. Za klijentske aplikacije korišćene su biblioteke JavaScript jezika, React i React Native. Radi lakše, brže i stabilnije implementacije, korišćene su i brojne biblioteke .NET-a, React-a i React Native-a, kao i neki spoljašnji API-ji i biblioteke.

5.1 Serverski deo

Serverski deo aplikacije podeljen je u više slojeva, po uzoru na MVC (Model-View-Controller) stil arhitekture sistema. Ovaj način organizacije odlikuje boljom preglednošću i smanjenjem zavisnosti u kodu. Aplikacija se sastoji iz sledećih glavnih slojeva:

- model – klase koje opisuju model podataka,
- repozitorijum – klase koje sadrže metode za pristup i komunikaciju sa bazom podataka,
- servis – vrši se manipulacija podacima, biznis logika aplikacije,
- DTO (Data Transfer Object) – objekti koji se prenose preko mreže i
- kontroler – sloj klasa za komunikaciju sa klijentom.

Obe klijentske aplikacije koriste jednu serversku aplikaciju, slanjem HTTP zahteva API-jima koji se nalaze u kontrolerima. Kontroleri prosleđuju dalje podatke servisima, koji vrše biznis logiku i pozivaju repozitorijume radi ažuriranja baze podataka i vraćaju istim redosledom nazad HTTP odgovor klijentu.

Repozitorijumi i servisi su implementirani u vidu interfejsa i klasa koje ih nasleđuju i implementiraju. Na taj način obezbeđena je manja zavisnost u kodu i lakše izmene.

Pored navedenih, osnovnih slojeva, većina aplikacija po potrebi ima i neke dodatne klase i slojeve, koji, u slučaju opisanog sistema, služe za implementaciju autentifikacije i autorizacije, obradu grešaka, podešavanja baze podataka i čuvanja razne konfiguracije.

Entiteti sistema, predstavljeni klasama na serverskom delu koje preslikavaju tabele baze podataka, su User, Admin, Student, Instructor, Examiner, School, Route i Appointment. Najbitniji podaci za veb aplikaciju, termini časova i ispita, sačuvani su u bazi u Appointments tabeli sa svim neophodnim informacijama, kao što su datum i vreme početka i kraja termina, učesnicima, tipom časa, rezultatom ispita i još nekim korisnim informacijama. Svaka ruta kojom je student vozio čuvana je kao lista koordinata koje formiraju putanju, uz vreme održavanja časa i podacima o učesnicima.

5.2 Klijentski deo – veb aplikacija

Veb aplikacija je odgovorna za prikaz podataka. Ona dobija podatke sa servera i prezentuje ih korisniku, upotrebom komponenti i njihovih stanja.

Najzanimljiviji deo opisane veb aplikacije čini implementacija kalendara i njegovih funkcionalnosti. Biblioteka „*react-big-calendar*“ na jednostavan način omogućava vizuelni prikaz kalendara i svih događaja u prikazanom vremenskom periodu. Potrebno je samo da server prosledi podatke o događajima, koji moraju uključivati datum i vreme izvršenja i spomenuta biblioteka će izgenerisati ceo prikaz kalendara i obeleženih događaja. Izgled kalendara se može menjati i prilagođavati svim vrstama aplikacija.

Još jedna bitna biblioteka koja je doprinela uniformnom i lepšem izgledu veb aplikacije je MaterialUI. Kao Google-ova biblioteka korišćena za bolji izgled veb aplikacija, sadrži veliki broj komponenti predefinisano izgleda koje su spremne za upotrebu bez dodatnog menjanja.

5.3 Klijentski deo – mobilna aplikacija

Mobilne aplikacije imaju identičnu ulogu kao veb aplikacije, a to je prikaz podataka čuvanih u bazi, dobijenih putem komunikacije sa serverom. Glavna razlika pri implementaciji mobilne aplikacije u odnosu na veb jeste korišćenje komponenti za mobilne aplikacije umesto HTML koda.

Glavni deo ove aplikacije, prikaz mape, implementiran je uz pomoć „*react-native-maps*“ biblioteke. Potrebno je samo instalirati biblioteku i iskoristiti njenu komponentu za prikaz mapa. Na Android uređajima automatski se prikazuje Google mapa, dok je na Apple uređajima prikazana Apple mapa, ali moguće je podesiti mapu po izboru, kao što je odrađeno u ovoj aplikaciji izborom Google mape.

Operativni sistemi raznih uređaja, u cilju zaštite svojih korisnika, su sve strožiji povodom sigurnosti ličnih podataka. Jedan od najosetljivijih podataka jeste lokacija korisnika i zbog toga je potrebno dobiti dozvolu za podelu lokacije sa aplikacijom. Implementiran je zahtev kojim se traži korisnikova saglasnost o podeli lokacije pri prvom pristupu mapi. Nakon što korisnik pristane, mapa učitava trenutnu lokaciju i prikazuje je na mapi.

Prilikom beleženja rute, putanja se iscrtava na mapi crvenom bojom na osnovu koordinata koje klijentska aplikacija prikuplja u toku pokreta. Nakon što je putanja sačuvana, svaki sledeći prikaz putanje zahteva dobavljanje sačuvanih koordinata iz baze podataka.

Spomenuta biblioteka za rad sa mapama omogućava sve manipulacije nad mapom, kao što su iscrtavanje putanje, označavanje trenutne lokacije, približavanje mape, rotacija i još mnoge druge koje možda nisu ni korišćene u ovom sistemu, a mogu biti značajne za razvoj nekih drugih aplikacija.

6. ZAKLJUČAK

Osnovni zadatak rada je bio razvoj sistema za auto-škole i ideja je bila da sistem bude podeljen na dve aplikacije, veb i mobilnu.

Omogućavanjem zakazivanja časova i ispita u auto-školama putem veb aplikacije, studentima bi put do vozačke dozvole bio brži, a ispitivačima bi način poslovanja bio prijatniji. Beleženje ruta bi značajno pomoglo i studentima i instruktorima u praćenju napretka učenika i smišljanju novih ruta i izazova.

Aplikacija je razvijana u trenutno savremenim tehnologijama koje se proširuju i napreduju i ima dobru osnovu za dalji razvoj. Ove funkcionalnosti mogu postati osnova za razvoj većeg sistema koji pomaže u svakodnevnim procesima auto-škola. Dodavanje opcije za unos godišnjih odmora i neradnih dana instruktora, implementacija servisa za plaćanje, dodavanje mogućnosti izbora i rezervacije vozila za čas su samo neke od ideja koje bi ovu aplikaciju unapredile i napravile ozbiljan sistem namenjen auto-školama.

7. LITERATURA

- [1] <https://teachworks.com/> (pristupljeno u septembru 2023.)
- [2] <https://drivescout.com/> (pristupljeno u septembru 2023.)
- [3] <https://www.samsara.com/> (pristupljeno u septembru 2023.)
- [4] <https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/> (pristupljeno u septembru 2023.)
- [5] <https://www.sam-solutions.com/blog/dot-net-vs-c/> (pristupljeno u septembru 2023.)
- [6] [https://en.wikipedia.org/wiki/React_\(software\)](https://en.wikipedia.org/wiki/React_(software)) (pristupljeno u septembru 2023.)
- [7] <https://www.netguru.com/glossary/react-native> (pristupljeno u septembru 2023.)

Kratka biografija:



Tamara Jančić rođena je u Novom Sadu 1997. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Računarstvo i automatika, odbranila je 2023.god. kontakt: tamaraa.jancic@gmail.com

PRIMENA FOTOAKUSTIČNE SPEKTROSKOPIJE ZA ANALIZU I MERENJE GASOVA

APPLICATION OF PHOTOACOUSTIC SPECTROSCOPY FOR GAS ANALYSIS AND MEASUREMENT

Elena Stojkovski, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu opisan je fotoakustični efekat, njegova istorija i teorijska pozadina efekta. Objasnjene su dve metode fotoakustike: fotoakustična spektroskopija i fotoakustično snimanje. Praktičan deo rada predstavlja detektovanje koncentracije CO_2 u zatvorenoj prostoriji u različitim uslovima putem CO_2 3 Click pločice koja se bazira na principu fotoakustične spektroskopije.

Ključne reči: Fotoakustika, fotoakustična spektroskopija, fotoakustično snimanje

Abstract – This paper presents photoacoustic effect alongside with its history and theoretical background. Two methods of photoacoustics are explained: photoacoustic spectroscopy and photoacoustic imaging. Practical part of this paper considers measuring CO_2 concentration in a room in different conditions using CO_2 3 Click board which is based on photoacoustic spectroscopy.

Keywords: Photoacoustics, photoacoustic spectroscopy, photoacoustic imaging.

1. UVOD

Fotoakustika (FA) predstavlja neinvanzivnu metodu za detekciju ili određivanje koncentracije željene supstance, kao i za kreiranje visokokvalitetnih slika (mapa) bioloških tkiva u zavisnosti od koncentracije merene supstance [1]. Fotoakustika, poznata i kao optoakustika, se zasniva na fotoakustičnom efektu. Odnosi se na metodu generisanja akustičnih talasa pomoću svetlosnih talasa ili nekog drugog vida radijacione energije [2]. Metoda je primenjiva na materijale u tečnom, čvrstom i gasovitom stanju.

Fotoakustika najveću primenu nalazi u analizi gasova i u medicini. Senzori za detektovanje su izuzetno precizni, imaju dobru rezoluciju i ponovljivost, i kao takvi su idealni za određivanje koncentracije primesa (gasova) u vazduhu.

Zbog osobine da merenja nisu invanzivna, fotoakustika sve veću primenu nalazi u medicini za razne detekcije i snimanja bez potrebe za hirurškim zahvatima [3]. Zračenja koja se koriste nisu jonizujuća i ne mogu imati negativan uticaj na zdravlje.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Jovan Bajić, vanr. prof.

Rad se sastoji od teorijskog dela praćenog praktičnim testiranjem radi bolje demonstracije rada sa fotoakustičnim senzorima i određivanjem kvaliteta merenja.

2. OSNOVE FOTOAKUSTIKE

2.1. Kratak pregled istorije fotoakustičnog efekta

Nakon otkrića telefona 1876. godine, Aleksandar Gram Bel počinje da istražuje kako bi mogao da primeni svetlosnu energiju u prenosu govora i time kreira bežični telefon. Prva istraživanja su se bazirala na fotosenzitivnim svojstvima selen koji, pri osvetljavanju modulisanim izvorom svetlosti (zračenjem sa periodičnim fluktacijama intenziteta) emituje zvuk čiji ton zavisi od frekvencije modulacije izvora [4].

Bel je prvi fotofon predstavio 1880. godine. Fotofon se sastojao od predajnika i prijemnika, i služio je za bežičnu komunikaciju na bazi prenosa svetlosne energije i njenim detektovanjem i konvertovanjem u akustični signal pomoću selen.

Nakon nekoliko publikacija od strane par naučnika, daljih istraživanja na temu fotoakustike skoro pa i nije bilo do 1938. godine kada je u Lenjingradu naučnik Veignerov opisao metodu analize gasa baziranu na fotoakustičnom efektu. Američki fizičar August Herman Pfund je 1939. godine opisao identičan uređaj kao Veignerov koji je služio za detekciju CO i CO_2 u malim koncentracijama. Prvo diferencijalno merenje fotoakustičnom metodom ostvareno je 1943. godine, čime je povećana osetljivost i smanjeni su šumovi [1].

2.2. Fotoakustični efekat



Slika 1. Dijagram principa nastajanja FA efekta

Na slici 1 dat je dijagram principa nastajanja FA efekta. Snop svetlosti ili nekog vida elektromagnetskog zračenja usmeri se u materijal. Materijal apsorbuje snop, što kao posledicu ima zagrevanje materijala, te dolazi do termičkog širenja. S obzirom da je snop modulisan ili pulsirajući, naizmenično dolazi do zagrevanja i hlađenja, to jest, do fluktuacija pritiska. Usled promena pritiska nastaju vibracije koje proizvode akustičan signal [5]. Signal se može detektovati mikrofonom, piezoelektričnim i pieoelektričnim materijalima, fotodiodama... Izborom talasne dužine izvora vrši se izbor analita koji se detektuje, obzirom da različiti analiti apsorbuju različite talasne dužine svetlosti.

2.3. Teorijska pozadina FA efekta

Pri laserskoj pobudi dolazi do promene temperature. U tom procesu se menja i zapremina. Definiše se odnos $\Delta V/V$ kao:

$$\frac{\Delta V}{V} = -\kappa p + \beta T \quad (1)$$

gde β predstavlja termalni koeficijent širenja zapremine, p promenu pritiska, T promenu temperature, dok je κ izotermalna kompresibilnost. Laseri koji se koriste kao izvori imaju termičko ograničenje i ograničenje naprezanja, te usled toga odnos $\Delta V/V$ ima zanemarljivu vrednost, ukoliko se to uvrsti u formuli (1) dobije se:

$$p_0 = \frac{\beta T}{\kappa} \quad (2)$$

Dodatne formule neophodne za izvođenje su:

$$T = \frac{\eta_{th} A_e}{\rho C_v} \quad (3)$$

$$\Gamma = \frac{\beta}{\kappa \rho C_v} \quad (4)$$

$$A_e = F \mu_a \quad (5)$$

U formuli (3) izražena je temperatura T preko procenta svetlosti konvertovanog u toplotu η_{th} , specifične optičke apsorpcije A_e , gustine ρ i specifične toplotne konstante pri nepromenljivoj zapremini C_v . Γ u formuli (4) predstavlja Gruneisenov parametar. Ukoliko se specifična optička apsorpcija A_e izrazi preko optičke gustine protoka F i optičkog koeficijenta apsorpcije μ_a dobija se formula (5). Ubacivanjem formula (3), (4) i (5) u formulu (2) dobija se krajnja formula za porast pritiska do kojeg dolazi tokom fotoakustičnog efekta:

$$p_0 = \Gamma \eta_{th} \mu_a F \quad (6)$$

Iz formule (6) može se primetiti da porast pritiska zavisi od Gruneisenovog parametra Γ (koji direktno zavisi od temperature), efikasnosti pretvaranja svetlosti u toplotu η_{th} , optičkog koeficijenta apsorpcije μ_a i optičke gustine protoka F .

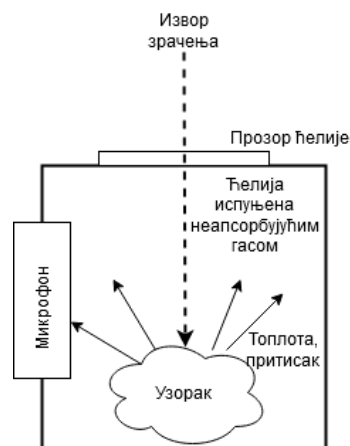
3. FOTOAKUSTIČNA SPEKTROSKOPIJA

Spektroskopija definiše nauku interakcije energije sa materijom i koristi se za kvantitativne i kvalitativne analize [6]. Najstariji oblik spektroskopije je optička spektroskopija. Mane konvencionalnih metoda optičke

spektroskopije prevaziđene su fotoakustičnom spektroskopijom.

Fotoakustična spektroskopija se zasniva na fotoakustičnom efektu. Na slici 2 dat je šematski prikaz ćelije za FA spektroskopiju. Tokom merenja neophodno je uzorak koji se analizira postaviti u zatvorenu ćeliju, dok se ostatak ćelije ispunjava nekim neapsorbirajućim gasom. U ćeliji se nalazi i mikrofoni koji služi za detekciju akustičnog signala. Akustični talas koji se generiše proporcionalan je apsorbovanoj energiji. FA signal zavisi od konstante ćelije, optičke snage izvora i koeficijenta apsorpcije [7]. Koeficijent apsorpcije zavisi od koncentracije gasa i poprečnog preseka. Iz ovoga sledi da porastom koncentracije analita raste koeficijent apsorpcije, a samim time dobija se i porast FA signala koji se detektuje. Na ovom principu zasnivaju se merenja u FA spektroskopiji.

Prednosti primene FA spektroskopije su da je merenje neosetljivo na rasejanje svetlosti, metoda nije destruktivna, nije potrebna priprema uzorka za analizu, osetljivost merenja se povećava povećanjem intenziteta izvora svetlosti...

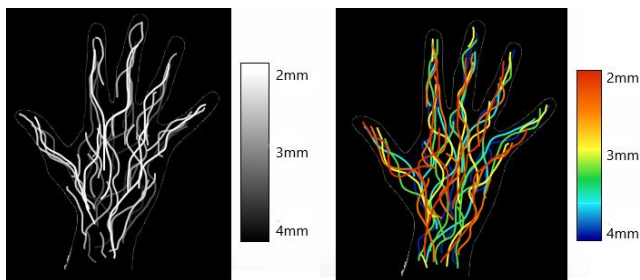


Slika 2. Šematski prikaz ćelije za FA spektroskopiju

4. FOTOAKUSTIČNO SNIMANJE

Fotoakustično snimanje omogućava mnogo bolju rezoluciju od drugih metoda optičkog snimanja koje se primenjuju u medicini. Bazira se na generisanju ultrazvučnog zračenja pomoću lasera radi dobijanja slika visoke rezolucije. Obezbeđuju *in vivo* ili *in vitro* snimanja i analize [8]. Ovom metodom moguće je dobiti slike poprečnog preseka, ali i trodimenzionalne slike.

Skice trodimenzionalne slike krvnih sudova u šaci dobijenih FA metodom prikazane su na slici 3. Krvni sudovi se prikazuju različitim intenzitetom jedne boje ili različitim bojama. Intenzitet boje (ili boja, zavisno od načina prikaza) zavisi od dubine na kojoj je krvni sud snimljen. Uz sliku obično stoji skala koja govori vezu između intenziteta (ili boje) i dubine snimanja. Kod snimaka poprečnog preseka se slike takođe prikazuju na jedan od ova dva načina, gde intenzitet ili boja daju informaciju o koncentraciji analita umesto o dubini na kojoj je došlo do detekcije. Snimanje nije ograničeno na krvne sudove, već je moguće detektovati i druge molekule i strukture od interesa.



Slika 3. Skica trodimenzionalnih slika dobijenih FA snimanjem

U FA snimanju primenjuju se nejonzirani elektromagnetski i ultrazvučni talasi kombinovano radi ostvarivanja veće dubine prodiranja za snimanje slika bez mrlja sa visokim elektromagnetskim kontrastom i visokom ultrazvučnom rezolucijom [9].

Dodatna prednost FA snimanja je ta što je moguće vršiti terapije zračenjem paralelno sa snimanjem, čime se šteti na vremenu i utrošku energije jer ne zahteva odvojene procedure za snimanje i terapiju.

Kontrast je jedan od parametara sa najvećim značajem u FA snimanju. Razlike u bojama i / ili intenzitetu pri snimanju stvaraju kontrast, omogućavajući bolju vidljivost detalja na slici. Veći kontrast omogućava bolju segmentaciju slike, pogotovo pri reprezentaciji slike putem boja. Kontrast nije uvek konstantan već zavisi od interakcije izvora svetlosti sa uzorkom. FA slike se mogu tumačiti kao mape koje prikazuju raspoređenost porasta pritiska p_0 u tkivu. Po formuli (6) pritisak zavisi od koeficijenta apsorpcije.

Koeficijent apsorpcije je dominantni parametar koji određuje mehanizam kontrasta u FA snimanju. Ovo dovodi do zaključka će da analiti koji imaju mali koeficijent apsorpcije davati manji kontrast i biće teži za uočavanje [10]. Radi poboljšanja kontrasta kod snimanja ovih analiza, u tkivo se injektuju kontrastni agensi koji povećavaju kontrast, ali i rezoluciju i dubinu prodiranja.

5. TESTIRANJE SENZORA NA BAZI FOTOAKUSTIČNE SPEKTROSKOPIJE ZA MERENJE KONCENTRACIJE CO₂

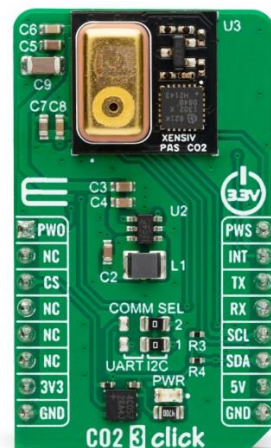
Radi demonstracije rada FA senzora, za svrhe ovog rada testirana je CO₂ 3 Click pločica (Slika 4) proizvedena od strane firme Mikroelektronika sa sedištem u Beogradu. Na pločici se nalazi XENSIV™ PASCO2V01BUMA1 senzor koji služi za visoko precizna merenja koncentracije CO₂ bazirana na fotoakustičnoj spektroskopiji. Pločica je korišćena zajedno sa Fusion for ARM v8 razvojnim sistemom, dok je programiranje i praćenje rezultata rađeno u softverskom alatu NECTOStudio.

Razvojni sistem i softverski alat su takođe razvijeni od strane firme Mikroelektronika. Merenja su rađena u kućnim uslovima zbog ograničenja na desktop računar. Meren je nivo CO₂ u maloj prostoriji sa otvorenim prozorima, nakon čega se merio porast koncentracije u istoj prostoriji ali sa zatvorenim prozorima.

Merenja su vršena sa jednom, dve i tri osobe prisutne u prostoriji. Poslednje merenje demonstrira uticaj duvanskog dima u neposrednoj blizini senzora na porast koncentracije.

Dobijeni rezultati prikazani su u tabeli 1. Svako od ovih 5 merenja nalazi se u novom redu tabele. Merenja su označena slovima. Opis merenja po slovima je sledeći:

- A: Koncentracija u prostoriji sa otvorenim prozorima, u prostoriji je boravila jedna osoba
- B: Koncentracija 10 minuta nakon zatvaranja prozora, u prostoriji je boravila jedna osoba
- C: Koncentracija 20 minuta nakon zatvaranja prozora, u prostoriji su boravile tri osobe
- D: Koncentracija 30 minuta nakon zatvaranja prozora, u prostoriji su boravile dve osobe
- E: Koncentracija 40 minuta nakon zatvaranja prozora, u prostoriji su boravile dve osobe
- F: Koncentracija 10 minuta nakon otvaranja prozora, u prostoriji su boravile dve osobe



Slika 4. CO₂ 3 Click pločica [10]

Svi rezultati prikazani su u jedinici ppm (eng. parts-per million, delova u milion). Sva merenja izvršena su 8 puta u razmacima od nekoliko sekundi radi utvrđivanja ponovljivosti uređaja, te u tabeli postoji 8 kolona sa rezultatima.

Tabela 1. Rezultati merenja koncentracije CO₂

	1	2	3	4	5	6	7	8
A	442	444	444	443	445	446	445	446
B	500	499	497	495	495	495	497	498
C	685	681	681	681	676	676	675	677
D	799	798	799	798	798	797	795	796
E	876	874	872	870	869	868	865	862
F	551	549	550	551	552	552	552	552

Iz tabele se može uočiti da su u svakom redu izmerene vrednosti približno jednake. Za svako merenje je računata srednja vrednost. U odnosu na srednje vrednosti, najmanja vrednost maksimalnog odstupanja dobijena je u merenju F i iznosi 1,125ppm. Najveća vrednost maksimalnog odstupanja od srednje vrednosti merenja dobijena je u merenju E i iznosi 7,5ppm. Ova odstupanja su minimalna u odnosu na izmerene vrednosti, te se može reći da senzor meri sa malim odstupanjima i dobrom ponovljivošću.

Nakon 40 minuta merenja koncentracija CO₂ u prostoriji se udvostručila (Tabela 1, red E). 10 minuta nakon otvaranja prozora koncentracija drastično smanjila

(Tabela 1, red *F*) i skoro dostigla vrednost dobijenu pre zatvaranja prozora (Tabela 1, red *A*). Kao zaključak sledi da je često otvaranje prozora neophodno radi održavanja kvaliteta vazduha u prostoriji.

Dobijeni rezultati su vezani za uslove u kojima je vršeno merenje. Porast koncentracije zavisi od veličine prostorije, broja osoba u prostoriji, ali i zdravstvenog stanja i aktivnosti osoba u prostoriji (ukoliko osoba, na primer, trenira u prostoriji, zbog ubrzanog disanja bi koncentracija CO₂ brže rasla, dok sporije disane rezultuje manjim porastom koncentracije). Na opadanje koncentracije provetravanjem, osim pomenutih faktora koji utiču, treba uzeti u obzir i koliki su prozori i koliko prozora je otvoreno. Dodatni faktori koji takođe utiču na merenja jesu postojanje mehanizma koji utiče na koncentraciju (klima, ventilacija, aspirator...), kao okruženje u kojem se nalazi prostorija (selo / grad, količina saobraćaja, količina zelenila...).

Kao što je i očekivano, koncentracija CO₂ je bila u porastu tokom celog merenja. U tabeli 2 upisane su izmerene vrednosti porasta koncentracije u prostoriji u zavisnosti od broja osoba u njoj. Vidi se da je koncentracija najmanje rasla kada je jedna osoba boravila u prostoriji, dok je najveći porast zabeležen kada su u prostoriji boravile tri osobe, što je teorijski bilo očekivano i potvrđeno je i u praksi ovim merenjem.

Tabela 2. Porast koncentracije CO₂ u prostoriji u zavisnosti od broja osoba u njoj

Broj osoba	1	2	3
Porast	52,625	118,5	182

Poslednje merenje pratilo je porast koncentracije CO₂ u zavisnosti od količine duvanskog dima u neposrednoj blizini senzora. Rezultati su prikazani u tabeli 3. Prvo merenje zabeleženo je pre paljenja cigarete u zatvorenom prostoru, drugo je zabeleženo nakon jednog dima od cigarete usmerenog direktno u pravcu senzora, treće nakon dva dima... Koncentracija CO₂ u okolini senzora se udvostručila nakon šestog dima usmerenog na senzor. U prostoriji bi se, nakon nekog vremena, ova koncentracija rasporedila, te ona ne bi bila toliko povećana, praćen je samo porast pri direktnom usmeravanju dima u senzor.

Tabela 3. Porast koncentracije CO₂ uzrokovan duvanskim dimom u neposrednoj blizini senzora

Br. dim.	0	1	2	3	4	5	6
Koncentr.	578	608	781	964	1086	1127	1184

6. ZAKLJUČAK

U ovom radu opisani su fotoakustični efekat, fotoakustična spektroskopija i fotoakustično snimanje. Objasnjen je princip nastajanja fotoakustičnog signala i date su teorijske osnove neophodne za razumevanje nastajanja signala i uticaja drugih parametara na signal koji se generiše. Teorijski deo rada praćen je i praktičnim testiranjem korišćenjem senzora baziranog na fotoakustičnoj spektroskopiji. Svi rezultati su bili u skladu sa očekivanjima i potvrdili su očekivanja dobijena teorijski. Senzor je pokazao izuzetno dobru ponovljivost u

merenjima. Osetljivost senzora je na visokom nivou i sa lakoćom detektuje i male promene koncentracije CO₂.

7. LITERATURA

- [1] Allan Rosencwaig, "Photoacoustics and Photoacoustic Spectroscopy", Robert E. Krieger Publishing Company, Malabar, Florida, 1990.
- [2] Andrew C. Tam, "Applications of photoacoustic sensing techniques", Review of Modern Physics. Vol. 58, iss. 2, pp.381-431, 1990.
- [3] Слободан З. Тодосијевић, "Развој фотоакустичног мерног система за термичку карактеризацију танких узорака", Електротехнички факултет, Универзитет у Београду, 2017.
- [4] Srirang Manohar and Daniel Razansky. "Photoacoustics: a historical review", Advances in Optics and Photonics, Vol. 8, iss. 4, pp.586, 2016.
- [5] Syahril Siregar, "Application of Carbon Nanotubes for Photoacoustic Imaging Contrast Agent", Doctor dissertation, School of Biomedical Engineering, Tohoku University, Sendai, 2018.
- [6] Allan Rosencwaig, "Photoacoustic Spectroscopy", Ann. Rev. Biophys. Bioeng, Vol 9, pp. 31-54, 1980.
- [7] Christoph Haisch, "Photoacoustic spectroscopy for analytical measurements", Measurement Science and Technology, Vol. 23, 2012.
- [8] A. Dolet, R. Ammanouil, V. Petrilli, C. Richard, P. Tortoli, D. Vray, F. Varray, "In Vitro and In Vivo Multispectral Photoacoustic Imaging for the Evaluation of Chromophore Concentration", Sensors, 2021.
- [9] Lihong V. Wang, "Photoacoustic Imaging and Spectroscopy", 1st Edition, CRC Press, Boca Raton, Florida, SAD, 2017.
- [10] Paul Beard, "Biomedical Photoacoustic Imaging", Interface Focus, Vol. 1, No. 4, pp. 602-631, 2011.
- [11] Mikroelektronika. "CO2 3 Click", доступно на интернет страници <https://www.mikroe.com/co2-3-click>, (последњи пут приступљено: 29.09.2023.).

Kratka biografija:



Elena Stojkovski rođena u Vršcu 1999. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Primenjena elektronika odbranila je 2022. godine. kontakt: elenakjm9@gmail.com

ARHITEKTURA ETHEREUM 2.0**ARCHITECTURE ETHEREUM 2.0**Jelena Cupać, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U radu su opisane faze ažuriranja Ethereum 1.0 mreže na Ethereum 2.0 mrežu. Radi demonstracije načina na koji se mogu inspektovati podaci na Ethereum mreži, kao što su izvršene transakcije i kreirani blokovi, korišćeni su istraživači blokova. Za kreiranje novčanika iskorišten je MetaMask, dok je kao istraživač blokova korišten EtherScan. Transakcije su izvršene i pregledane na testnoj mreži Sepolia.

Ključne reči: *blokčejn, ethereum 1.0, ethereum 2.0, mehanizam dokaza o radu, Sepolia*

Abstract – The paper describes the phases of updating the Ethereum 1.0 network to the Ethereum 2.0 network. To demonstrate how data on the Ethereum network can be inspected, such as performed transactions and created block, block explorers are used. MetaMask was used to create the wallet, while EtherScan was used as a block explorer. Transactions were executed and reviewed on the Sepolia test network.

Keywords: *blockchain, ethereum 1.0, ethereum 2.0, proof of stake, Sepolia*

1. UVOD

Potreba za decentralizovanim novcem je više predstavljala teorijski koncept, sve dok u prethodnoj deceniji ovaj koncept postaje održiv, i to zahvaljujući Satoši Nakamotu i njegovom radu 2008. godine u kojem predstavlja Bitcoin i blokčejn tehnologiju. Bitcoin i blokčejn tehnologija su počeli da oblikuju i definišu nove aspekte u domenu računarstva i informacionih tehnologija.

Vremenom, počeli su da se razvijaju i drugi projekti zasnovani na blokčejnu, jedan od njih je i Ethereum. Danas, Ethereum je decentralizovana mreža čvorova koja održava zajednički pogled na globalno stanje i automatski izvršava programski kod kada su prethodno definisani uslovi ispunjeni – pametni ugovori. Kako je broj korisnika Ethereum mreža u stalnom porastu, postojala je potreba za stabilnijom i skalabilnijom mrežom. Upravo to je dovelo do ažuriranja mreže koje će se odigrati kroz tri faze.

Cilj rada jeste upoznavanje sa tri faze ažuriranja Ethereum 1.0 mreže na Ethereum 2.0 mrežu, kao i demonstracija načina inspekcije podataka na mreži korišćenjem istraživača blokova.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Srđan Vukmirović, red. prof.

2. LANAC BLOKOVA - BLOKČEJN

Potreba da se dokumentima utvrdi autentičnost postoji odavno. Jedan od načina da se utvrdi da je dokument kreiran baš u onom trenutku u kojem autor tvrdi jeste oslanjanje na centralni autoritet u koji svi imaju poverenja. Drugi način ne zahteva centralni autoritet, već samo javno objavljivanje. Iako je moguće lažirati pečate, otvoriti dokument tako da se izbegne fizička zaštita, često je to nepraktično i zahteva vreme, što napadaču ne biva isplativo. Međutim, u digitalnom svetu kreiranje identičnih kopija i lažiranje dokumenata je gotovo trivijalno.

Problem autentičnosti je rešen kriptografijom i digitalnim potpisima. Kako bi se utvrdilo da je digitalni dokument objavljen u određenom trenutku, moguće je imati centralni autoritet i verovati njemu, kao što je slučaj kod digitalnih sertifikata ili je, pak, ovo moguće postići i bez centralnog autoriteta, i to najčešće ulančavanjem kriptografskih heš vrednosti dokumenata i javnim objavljivanjem ovako kreiranog lanca. Blokčejn spaja tehnologije digitalnih potpisa i metode za utvrđivanje hronologije izmena čime se formira poverljiv javni skup podataka.

2.1. Terminologija blokčejna**2.1.1. Distribuirani sistem**

Distribuirani sistem je kolekcija autonomnih računarskih elemenata koji su krajnjem korisniku predstavljeni kao jedan koherentni sistem [1]. Elementi se nazivaju čvorovi deluju koordinisano kako bi ispunili zajedničke ciljeve.

2.1.2. Peer-to-peer sistem

Peer-to-peer (P2P) sistemi pripadaju decentralizovanom tipu sistema. Svi čvorovi u ovakvom sistemu su jednaki, zapravo, svaki čvor je istovremeno i klijent i server (engl. *servant*). Posao je raspoređen po čvorovima umesto da je za obavljanje posla zadužen svega jedan server što P2P sisteme čini otpornijim budući da nema jedinstvene tačke otkaza.

2.1.3. Blokčejn

Postoji nekoliko definicija blokčejna. Blokčejn je distribuirani deljeni registar transakcija. Blokčejn je jezgro peer-to-peer sistema koji je kriptografski bezbedan, nepromenljiv - u koji se podaci mogu samo dodavati, a ne mogu se brisati, niti menjati. Sa poslovnog stanovišta, blokčejn se može definisati kao platforma gde čvorovi mogu da razmenjuju dobra koristeći transakcije bez postojanja centralnog tela kojem strane veruju [2].

2.1.4. Adresa

Adresa je jedinstveni identifikator koji služi da se označi primalac i pošiljalac u transakciji. Najčešće su to javni ključevi.

2.1.5. Transakcija

Transakcija je osnovna jedinica blokčejna, transfer vrednosti sa jedne adrese na drugu. Transakcije se organizuju u blokove.

2.1.6. Blok

Više transakcija su organizovane zajedno u jedan blok. Blokovi se povezuju kako bi kreirali krajnji lanac blokova – blokčejn. Pored grupe transakcija, blok sadrži i vrednost koja se samo jednom upotrebljava (*nonce*), vremensku odrednicu, kao i heš vrednost prethodnog bloka (*hash pointer*).

2.1.7. Čvorovi

Čvorovi su jaki računari povezani na blokčejn mrežu u cilju obrade i verifikacije transakcija. Čvorovi u mreži mogu da obavljaju različite funkcije u zavisnosti od njihove uloge – predlažu/validiraju transakcije i sl.

2.2. Konsenzus mehanizmi

Konsenzus je proces dogovaranja oko konačnog stanja podataka između nepoverljivih čvorova. Postoji nekoliko konsenzus algoritama, od kojih su dva korištena u *Ethereum* mreži – mehanizam dokaza o radu (engl. *proof of work*) i mehanizam dokaza o udelu tj, posedovanju valute (engl. *proof of stake*).

2.2.1. Proof of work (PoW)

Mehanizam dokaza o radu je konsenzus algoritam koji se oslanja na dokaz da je dovoljno računarskih resursa utrošeno pre predlaganja bloka za dodavanje u blokčejn. Ovaj algoritam se naziva i rudarenje. Rudarenje je proces validiranja transakcija rešavajući jednačinu. *Nonce* je broj koji se vrti u rasponu od 0 do 232 u pokušaju da se nađe onaj za koji će heš bloka imati odgovarajuću vrednost (nižu od zadate). Upravo je ovo jednačina koju rudari treba da reše kako bi validirali blok. Heš funkcija koja se koristi je SHA256.

Kako su heš funkcije jednosmerne funkcije, rudari nemaju način da pretpostave kakvi su im ulazni podaci potrebni da bi dobili rezultat koji žele. Zato moraju da isprobavaju redom različite vrednosti *nonce* broja, jednu po jednu, da bi dobili željeni rezultat. Stoga je bitno imati jak hardver, jer se tako može više pokušaja napraviti za kraće vreme i time povećava šansu da baš on bude taj koji će prvi kreirati validan blok. Da bi se našli potrebni ulazni podaci koji daju odgovarajući rezultat neophodan je izuzetno veliki broj pokušaja, a da bi se taj veliki broj pokušaja odradio za kratko vreme, neophodna je velika procesorska snaga i velika količina rada. Zbog toga se ovaj algoritam za rudarenje naziva mehanizam dokaza o radu, jer je rudar morao da uloži veliku količinu rada da bi došao do rezultata.

Pre nego što se u blokčejn doda blok koji je rudar kreirao i koji zadovoljava potrebne uslove, on ga šalje ostatku

mreže na proveru validnosti. Oni proveravaju da li se za ulazne podatke koje je on imao zaista dobija heš koji on tvrdi da je dobio i, ako se pokazalo da je bio u pravu, taj blok zvanično postaje sastavni deo blokčejna i rudar dobija svoju nagradu i proviziju od transakcija. Ako blok nije validan, mreža će odbiti taj blok.

2.2.2. Proof of stake (PoS)

Mehanizam dokaza o udelu, tj. posedovanju valute menja način na koji se verifikuje i dodaje novi blok, te se umesto potrebe za obavljanjem računarskog posla i rešavanja kriptografske jednačine primenjuje ulaganje, odnosno vlasnici ulažu svoj novac kako bi dobili priliku da kreiraju nove blokove i budu nagrađeni. Što više novca biva uloženo, veća je verovatnoća da će algoritam izabrati baš tog validatora za kreiranje novog bloka. Ukoliko je validator maliciozan, a doda transakciju u blokčejn koju drugi validatori smatraju neispravnom, može izgubiti deo sredstava koje je uložio. Blokove proverava više validatora, a kada određeni broj validatora potvrdi ispravnost bloka, blok se smatra finalnim.

3. ETHEREUM 1.0

Ethereum je decentralizovana blokčejn platforma koju pokreće kriptovaluta pod nazivom *Ether*, koja omogućava vršenje transakcija, korišćenje i čuvanje nezamenljivih tokena (NFT), igranje igrice, korišćenje društvenih mreža i još mnogo toga. Platforma podržava i decentralizovane aplikacije (DApps), decentralizovane finansije (DeFi), decentralizovane razmene (DEX).

3.1. Ethereum nalog

U *Ethereum* mreži postoje dva tipa naloga, oni koji su u eksternom vlasništvu (kontrolisani privatnim ključem) i ugovorni nalozi kontrolisani pametnim ugovorom. Nalozi u eksternom vlasništvu su npr. *Metamask* ili *Coinbase* novčanici. Ovi računi se identifikuju preko javnog ključa koji predstavlja njihovu adresu. Kod ugovornih naloga adresa se dobija od adrese kreatora naloga i *nonce* vrednosti. Svaki *Ethereum* nalog je sačinjen iz 4 dela [3]:

- *Nonce* – broj transakcija koje su poslate sa određenog naloga ili broj kreiranih ugovora određenog naloga. Garantuje da se svaka transakcija može precesuirati samo jednom.
- *Ether* balans – broj *Wei*-a koji se poseduje na nalogu.

Wei je najmanja jedinica *Ether* kriptovalute.

- Heš kod – heš vrednost EVM bajt koda naloga, to je kod koji se izvršava kada adresa računa dobija poruke. Koristi se Keccak-256 heš funkcija.
- *Storage root* – 256-bitna heš vrednost korena *Merkle Patricia* stabla koje predstavlja sadržaj naloga.

3.2. Transakcije

Transakcija je kriptografski potpisan set instrukcija. Ove instrukcije mogu da prenose *Ether* sa jednog računa na drugi ili da interaguju sa pametnim ugovorima na blokčejnu. Svaka transakcija se sastoji iz pet delova [3]:

- Primalac transakcije
- Količina *Ether*-a koja se šalje transakcijom
- *Nonce* – broj transakcije poslate od strane tog

pošiljaoca

- *Gaslimit* – maksimalan gas koji može da se potroši za izvršenje transakcije
- *Gas price* - broj *wei*-a po jedinici gasa koje pošiljalac plaća
- Potpisani podaci kojima se identifikuje pošiljalac.

3.3. Konsenzus mehanizam

Ethereum 1.0 koristi PoW algoritam za validiranje transakcija. Čvorovi u mreži se takmiče ko će pre rešiti jednačinu i kreirati novi blok. Kao heš funkcija koristi se *Ethash* heš funkciju, dok se kao glavni resursi za rudarenje koriste grafičke procesorske jedinice - GPU.

4. ETHEREUM 2.0

Ethereum kao platforma omogućava korisnicima da kreiraju pametne ugovore. Pametni ugovori su sporazumi između dve strane koji treba da izvrše transakciju kada su uslovi koji su definisani na početku ispunjeni. Ovi ugovori se smatraju izuzetno bezbednima budući da se čuvaju na blokčejn mreži, nakon što su odobreni, repliciraju se po mreži što ih štiti od zlonamernih akcija i menjanja. To je doprinelo razvoju okruženja i to u pogledu korisnika, *Etherscan* procenjuje da postoji preko 231 miliona jedinstvenih adresa, aplikacija, i to se prognozira da će do kraja 2023. godine postojati oko 500 hiljada decentralizovanih aplikacija, i u pogledu ukupnog korišćenja, pri čemu broj transakcija dostigne 1.2 miliona na dnevnom nivou. Zbog svega navedenog, potreba za stabilnijom i skalabilnijom mrežom se samo povećavala. Stoga su programeri koji stoje iza *Ethereum*-a, zajedno sa Vitalikom Buterinom, konačno odredili datume za dugo očekivano ažuriranje glavne mreže.

Ažuriranje *Ethereum* 1.0 mreže je *Ethereum* 2.0, poznat i pod formalnim nazivom *Serenity* i bio je u planu od skoro samog nastanka *Ethereum*-a. Buterin je već na samom početku shvatio da će mali broj transakcija koje je *Ethereum* 1.0 sposoban da obradi u sekundi postati problem pre ili kasnije. Ovaj broj je iznosio svega 15 transakcija po sekundi, a to nije bilo dovoljno da se zadovolji vizija *Ethereum*-a kako bi podržao mnoštvo decentralizovanih aplikacija.

Ažuriranje na *Ethereum* 2.0 je predviđeno da se izvede u nekoliko faza, tačnije u tri, od kojih su dve već realizovane. Najvažnije promene predviđene 2.0 verzijom su orijentisane na poboljšanje skalabilnosti kako bi mreža mogla da podrži još veći broj korisnika i transakcija. Među promenama je i prelazak sa PoW mehanizma konsenzusa na PoS, u kojem će korisnici zalagati ETH kako bi postali validatori, umesto da se učešće u kreiranju blokova zasniva na procesorskoj moći sistema koji korisnik koristi. Još jedna promena bi bila i tranzicija sa jednog lanca blokova na 64 lanca koji se paralelno izvršavaju.

4.1. Faza 0

Faza 0 podrazumeva lansiranje *Beacon* lanca i prelazak sa PoW mehanizma konsenzusa na PoS, započeta je 1.12.2020. godine. U ovoj fazi potrebno je bilo prikupiti 524288 ETH (16384 validatora, ovi brojevi su odlučeni

kako bi se obezbedila dovoljna sigurnost i decentralizacija) za lansiranje *Beacon* lanca – blokčejn lanca koji koristi PoS mehanizam konsenzusa. Ovaj lanac je kreiran kako bi se osiguralo da je logika PoS mehanizma održiva pre nego što bi se omogućila na *Ethereum*-ovoj glavnoj mreži tzv. *mainnet*-u. Uporedo sa ovim lancem postojao je originalni lanac koji koristi PoW mehanizam konsenzusa sve do trenutka spajanja, tzv. *The Merge* događaja, što je zapravo i sledeća faza - faza 1. *Beacon* lanac nije mogao da procesuirati transakcije, izvršava pametne ugovore i hostuje decentralizovane aplikacije.

4.1.1. Gasper algoritam

PoS mehanizam konsenzusa koji *Ethereum* 2.0 koristi se oslanja na Gasper algoritam. Gasper algoritam definiše kako se validatori nagrađuju i kažnjavaju, odlučuju koje blokove da prihvate i odbace i koju granu blokčejna da nadograđuju, tj. dodaju nove blokove. Gasper algoritam je dizajniran da zameni energetske intenzivan proces rudarenja sa energetske znatno efikasnijim procesom validacije. U novom sistemu korisnici, da bi postali validatori, moraju uložiti minimalno 32 ETH u pametni ugovor na originalnom *Ethereum* lancu. Zatim se identična količina ETH kreira na *Ethereum* 2.0 lancu i korisnici je mogu iskoristiti da postanu validatori. Bitno je napomenuti da ETH koji je kreiran na *Ethereum* 2.0 mreži se ne može konvertovati nazad u originalni lanac. Alternativno, verifikaciju i validaciju blokova izvršavaju validatori koji su odabrani prema uloženom depozitu. Drugim rečima, moć glasanja svakog validatora će biti određena količinom ETH koju su založili. Takođe, uloženi depozit se može izgubiti ukoliko čvorovi nisu poštteni u predlaganju ili validaciji blokova ili su neaktivni.

U svakom slotu (period u trajanju od 12 sekundi) nasumično se bira validator koji treba da predloži blok. Nasumičnost se postiže korišćenjem algoritma RANDAO koji kombinuje heš vrednost predlagača bloka sa *seed*-om koji se ažurira u svakom slotu. Izbor validatora je fiksiran dve epohe unapred (svaka epoha iznosi 32 slot, tj. 6.4 minute) kao način zaštite od manipulacije *seed*-om. Validatori se stalno dodaju u RANDAO, ali se globalna RANDAO vrednost ažurira samo jednom po epohi [4]. Validatori grupišu transakcije, izvršavaju ih, umotavaju u blok i šalju drugim validatorima. Validatori kada dobiju novi blok ponovo izvršavaju transakcije kako bi se uverili da li se slažu sa predloženom izmenom stanja. Ukoliko je blok validan, dodaju ga u svoju bazu podataka.

4.2. Faza 1

Faza 1 se još naziva i *The Merge* i odigrala se 15. septembra 2022. godine kada je *Beacon* lanac spojen sa *mainnet*-om. Dobijeni lanac izvršavao je PoS konsenzus mehanizam, rudarenje više nije bilo sredstvo za kreiranje novih blokova. Ovo ažuriranje mreže nazvano je Pariz. Svrha spajanja bila je eliminacija PoW algoritma, budući da je jedan lanac (*beacon*) koristio PoS, a drugi lanac (*Mainnet*) PoW. Sveukupno gledano, prelazak na PoS učinio je *Ethereum* znatno efikasnijim i decentralizovanim – što više korisnika učestvuje u mreži ona postaje bezbednija i decentralizovanija.

Korišćenje PoS algoritma je osnovna komponenta za siguran, ekološki prihvatljiv i skalabilan *Ethereum* koji danas postoji.

4.3. Faza 2

Faza 2 ima za cilj implementaciju šardova. Prvobitno, plan je bio da se na šardovanju poradi pre samog *Merge*-a, kako bi se rešio problem skalabilnosti. Međutim, sa rešenjem za skaliranje drugog sloja, prioriteta su se izmenili, te se faza 2 još nije ni odigrala. Šardovanje je vid particionisanja podataka, gde se velika baza podataka deli na manje kako bi se lakše upravljalo manjim delovima i povećala celokupna efikasnost. Na primeru blokčejna, šardovanje se odnosi na deljenje lanca na više manjih čime se stvara struktura mreže. Ovo doprinosi smanjenju radnog opterećenja validatora koji skladište i upravljaju samo jednim delom mreže umesto celim blokčejnom. Drugi sloj je zajednički naziv za rešenja za skaliranje rukovanjem transakcija van glavne mreže *Ethereum*-a uz korišćenje prednosti glavne mreže. Većina rešenja se odnosi na server ili klaster servera. Transakcije se podnose čvorovima na drugom sloju, umesto direktno na prvi sloj. Glavna ideja koja stoji iza drugog sloja jeste povećanje protoka transakcija, ali bez žrtvovanja decentralizacije i bezbednosti.

5. ISTRAŽIVAČI BLOKOVA

Podaci blokčejna su javno dostupni. Istraživači blokova (engl. *block explorer*) omogućavaju pristup podacima vezanim za transakcije na blokčejnu, uključujući iznos transakcije, njen status, pošiljaoca i primaoca. Podaci koji su dostupni za pregled su podeljeni u dve grupe: podaci izvršavanja i konsenzus podaci. Podaci izvršavanja se odnose na transakcije koje su izvršene u okviru specifičnog bloka ethorg. Konsenzus podaci se odnose sa blokove i validatora koji su ih predložili.

Za pregledanje transakcija i blokova u radu je korišćen *EtherScan* istraživač bloka. Za potrebe kreiranja transakcije preduslov je bio imati novčanik, stoga je iskorišćena *MetaMask* brauzer ekstenzija za kreiranje novčanika, a kako bi se dobili ETH i inicirala transakcija iskorišćen je *Sepolia faucet*. Za realizaciju transakcije korišćena je *Ethereum*-ova testna mreža – *Sepolia*.

Nakon uspešne ETH transakcije, na *Etherscan*-u su prikazani njeni podaci. Kako je korišćena *Sepolia* testna mreža, neophodno je bilo koristiti i *Etherscan* namenjen ovoj testnoj mreži. Radi pretrage transakcija koristi se heš same transakcije koji se može pronaći u okviru *MetaMask* novčanika. Nakon unosa heša transakcije, detalji koji su prikazani za transakciju su sam heš transakcije, njen status, broj bloka kojem ona pripada, kada je izvršena, javne adrese primaoca i pošiljaoca, kao i vrednost koja je poslata, gas (jedinica koja meri količinu računarskog napora potrebno za izvršenje operacije na *Ethereum* mreži) i nadoknada za transakciju. Ukoliko transakcija nije bila uspešna, umesto statusa *success/successful* može da se nađe još opcija:

- poruka o greški – *bad instruction/out of gas*
- status na čekanju – transakcija čeka da bude potvrđena/obrađena
- nije pronađena - ili transakcija nije prošla ili još

uvek nije prikazana u istraživaču bloka

- vraćena – greška korisnika pametnog ugovora, neophodno je još jednom proveriti detalje transakcije
- Moguća je i inspekcija samog bloka kojem data transakcija pripada, ukoliko se klikne na prikazani broj bloka. Uz to, ukoliko se u polje za pretragu unese javna adresa računa, moguće je vršiti inspekciju aktivnosti računa i njegovo stanje. *Etherscan*, pored navedenih primena, ima i mnoge druge, kao što je inspekcija pametnih ugovora (može se pregledati i njihov izvorni kod), nezamisljivih tokena, može se pratiti statistika mreže (*Topstat* stranica), mogu se uključiti notifikacije za praćenje aktivnosti određenog računa/adrese i još mnogo toga.

6. ZAKLJUČAK

U radu su opisane tri faze ažuriranja *Ethereum* mreže. Objasnjen je koncept blokčejna, i uz to su detaljno opisani konsenzus mehanizmi koji mreže *Ethereum* 1.0 i 2.0 koriste. Korišćenjem *MetaMask* novčanika i *Sepolia* testne mreže su izvršene transakcije, a zatim pomoću *Etherscan*-a je izvršen i uvid u detalje same transakcije na blokčejnu.

Ažuriranje *Ethereum* mreže donelo je mnogo prednosti, smanjeni su hardverski zahtevi budući da se više ne koristi hardverski zahtevan PoW konsenzus mehanizam. Iz istog razloga, sada je *Ethereum* više ekološki prihvatljiv jer je smanjena potrošnja električne energije. Još jedna prednost su i niže transakcijske nadoknade, kao i povećanje propusnosti mreže na kojem je ideja i da se ubuduće radi.

7. LITERATURA

- [1] Maarten van Steen, Andrew S. Tanenbaum, *Distributed Systems*, 2018
- [2] Imran Bashir, *Mastering Blockchain*, 2017
- [3] Dejan Vujičić, Siniša Randić, *Blockchain technology, bitcoin, and Ethereum: A brief overview*, 2018
- [4] <https://ethereum.org/en/developers/docs/consensus-mechanisms/pos> (pregledano avgusta 2023.)

Kratka biografija:



Jelena Cupać rođena je 03.02.1999. u Somboru. U Svcu je završila Osnovnu školu „20. oktobar“ 2013. godine kao nosilac Vukove diplome i zvanja Đaka generacije. Iste godine upisuje gimnaziju „Veljko Petrović“ u Somboru, prirodno- matematički smer. Godine 2017. završava gimnaziju kao nosilac Vukove diplome. Nakon završene gimnazije, upisuje Fakultet tehničkih nauka Univerziteta u Novom Sadu, smer Softversko inženjerstvo i informacione tehnologije. Sve ispite polaže i studije završava u roku, 2021. godine. Iste godine, upisuje master studije na Fakultetu tehničkih nauka, smer Elektronsko poslovanje. Sve ispite polaže i studije završava 2023. godine. kontakt: cupac.r230.2021@uns.ac.rs

МОДЕЛОВАЊЕ КОНСТРУКЦИОНИХ ОСОБЕНОСТИ АСИНХРОНЕ МАШИНЕ ПРИМЕНОМ СОФТВЕРСКИХ АЛАТА

MODELING OF STRUCTURAL CHARACTERISTICS OF INDUCTION MACHINE USING SOFTWARE TOOLS

Данило Дробњак, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – У овом раду, моделоване су конструкционе особености асинхроне машине применом софтверских алата. Рађена је студија случаја асинхроне машине санге 11 kW са различитим навојним кораком секције. Упоредјену су различите карактеристике за различите навојне кораке секције.

Кључне речи: Асинхрона машина, намотаји, навојни корак, секција

Abstract – In this paper, the construction features of the asynchronous machine were modeled using software tools. A case study of an 11 kW asynchronous machine with different throw of the section was made. Different characteristics were compared for different throw of the section.

Keywords: Asynchronous machine, windings, throw, section

1. УВОД

За формирање обртног момента у циљу покретања радних кола машина данашња технологија нуди неколико могућности – пнеуматски погон, хидраулички погон, СУС погон и онај за који се слободно може рећи да је највише заступљен, електрични. Претварање електричне енергије у механичку поверено је уређајима које обједињено називамо електричним моторима.

Електричне машине су од свих поменутих начина за формирање обртног момента погодне и у том смислу што поседују могућност реверзибилности односно могућност да ток енергије кроз њих може бити двосмеран односно да могу претварати електричну енергију у механичку али и обрнуто, да могу вршити конверзију електричне у механичку.

Асинхрона машина у савременој индустрији представља најважнији уређај за извршење механичког рада. Коришћење асинхроне машине је економски исплативо и оправдано, одржавање је јефтино, отпорна је на грубе манипулације и једноставна је за коришћење, поуздана је и сигурна током рада, није великих габарита. У примени најчешће се сусреће као трофазни мотори.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био др Дејан Јеркан.

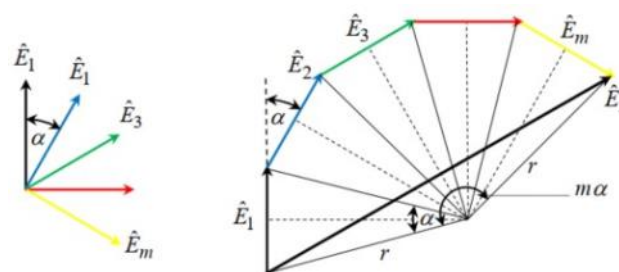
2. ОСОБЕНОСТИ И МАТЕМАТИЧКА ПРЕСТАВА АСИНХРОНЕ МАШИНЕ

Од првих изведби асинхроних машина па до данашњих изведби, асинхроне машине су претрпеле читав низ унапређења и измена у циљу побољшања њиховог рада, али и низ компромиса између економских аспеката експлоатације машина и постизања њихове учинковитости.

2.1. Навојни сачинилац

2.1.1. Појасни навојни сачинилац

Више секција намотаја везаних на ред чине једну полно фазну групу. Ако се претпостави да се полно фазне групе састоји од m секција везаних на ред, оне су смештене у $2m$ жлебова, односно у две зоне са по m жлебова. Векторским сабирањем фазора индукованих напона у појединим проводницима који припадају једној зони добија се укупни напон зоне E_z .



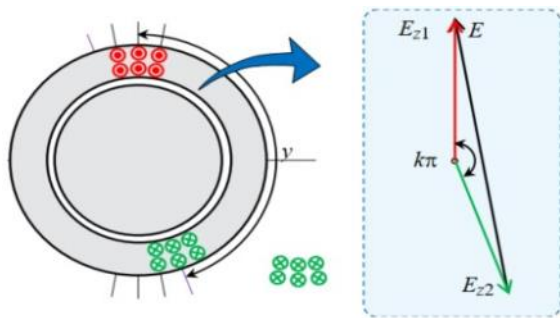
Слика 1 - Фазори индукованих електромоторних сила у проводницима једне зоне [1]

Појасни навојни сачинилац се дефинише се као однос индукованог напона једне зоне E_z , и алгебарског збира напона свих проводника у зони, $m \cdot E_1$:

$$k_p = \frac{E_z}{m \cdot E_1} = \frac{\sin\left(m \frac{\alpha}{2}\right)}{m \cdot \sin\left(\frac{\alpha}{2}\right)} \quad (2.1)$$

2.1.2. Тетивни навојни сачинилац

Често се израђују секције намотаја код којих је навојни корак мањи од полног корака τ_p , то су секције са скраћеним, навојним кораком. Жлебови у које се смешта секција са скраћеним кораком се налазе на тетиви кружнице па одатле назив тетивна секција, које се могу користити само код двослојног намотаја.



Слика 2 - Електромоторна сила зоне скраћеног намота [1]

Укупан индуковани напон редне везе тетивних секција, односно напон индукован у једној полно фазној групи је векторска разлика фазора електромоторних сила зона, и он износи:

$$E = 2E_z \sin\left(\frac{y}{\tau_p} \cdot \frac{\pi}{2}\right) \quad (2.2)$$

Однос тог укупног напона E и алгебарског збира електромоторних сила зона $2E_z$ представља тетивни навојни сачинилац:

$$k_t = \frac{E}{2E_z} = \sin\left(\frac{y}{\tau_p} \cdot \frac{\pi}{2}\right) \quad (2.3)$$

2.1.3. Резултатни навојни сачинилац

Резултатни навојни сачинилац k једнак је производу појасног и тетивног навојног сачиниоца. Вредности навојних сачинилаца су углавном врло блиски јединици.

$$k = k_p \cdot k_t \quad (2.4)$$

2.2. Обртно поље

Типични представник електричних машина је трофазни асинхронни мотор. На статору постоје три расподељена намотаја који су просторно померени за по $2\pi/3$. Кроз три фазна намотаја протичу наизменичне струје истих амплитуда и кружне учестаности ω_s , које су фазно померене за по $2\pi/3$. Те струје стварају вектор магнетнопобудне силе у правцу осе намотаја. Амплитуда магнетнопобудне силе се мења на ободу зазора у функцији времена. Оваква магнетнопобудна сила се назива пулсирајућа магнетнопобудна сила. У таквој машини са три намотаја кроз које теку дате наизменичне струје постоје три пулсирајуће магнетнопобудне силе, односно три пулсирајућа поља.

Резултатна магнетнопобудна сила у сваком тренутку времена је збир магнетнопобудних сила од појединачних намотаја, има константну амплитуду и ротира у простору синхроним брзином ω_s . Овој обртној магнетнопобудној сили одговара и обртно поље, познатије као Теслино обртно поље. С обзиром да кроз све три фазе статорског намотаја протичу наизменичне, фазно померене струје, оне узрокују резултатну магнетнопобудну силу у зазору, која се понаша као да је настала од сталних магнета [3],[4].

2.3. Губици

Преображај енергије из једне врсте у друг неизбежно је везан са губицима. Према томе, да би асинхронни мотор могао да даје корисну механичку снагу P , његова утрошена електрична снага P_s , коју узима из мреже на коју је прикључен, мора да буде већа од корисне снаге за износ свих губитака у мотору.

Од утрошене снаге један део се троши у самом намоту статора на Џулове губитке P_{Cus} , друг део снаге троши се на губитке у гвожђу статора услед хистерезе и вртложних струја P_{Fe} . Остатак утрошене снаге преноси се електромагнетним путем помоћу обртног поља на ротор, стога се тај остатак утрошене снаге назива снага обртног поља.

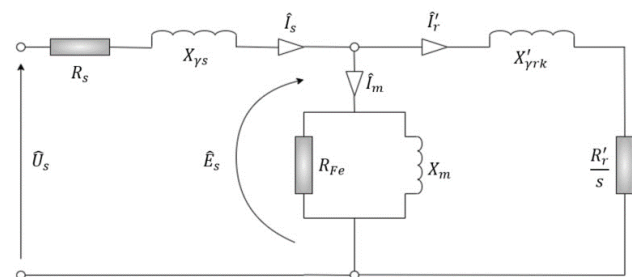
Од снаге обртног поља део се троши на Џулове губитке у баку ротора P_{Cur} , део на механичке губитке P_{gmeh} на трење у лежиштима и вентилацији усед трења ротора о ваздух, део на додатне губитке P_d . Према томе корисна снага мотора једнака је:

$$P = P_s - (P_{Cus} + P_{Fes} + P_{Cur} + P_d + P_{gmeh}) \quad (2.5)$$

2.4. Еквивалентна шема

Како би се лакше моделовале, асинхроне машине се представљају заменском, еквивалентном шемом. За устаљена стања, шема се може одредити аналогно трофазном трансформатору, при чему је статор представљен као примар трансформатора, а ротор је у улози кратко спојеног секундера.

Да би се добила јединствена еквивалентна шема статорског и роторског намотаја, неопходно је да се сведе коло ротора на статорску страну, при чему се користи релација електромоторних сила. Наиме, веза електромоторних сила ротора и статора се може записати преко преносног односа $m = E_s/E_{kr}$, што еквивалентну шему асинхронног мотора чини једнаком трансформаторској шеми.



Слика 3 - Еквивалентна шема асинхронног мотора

3. СТУДИЈА СЛУЧАЈА АСИНХРОНЕ МАШИНЕ СНАГЕ 11 kW

Рађена је студија случаја кавезне асинхроне машине снаге 11 kW са различитим навојним кораком секције. У наставку су приказани основни подаци машине, карактеристике машине, а затим су упоређене различите карактеристике за различите навојне кораке секција. За студију случаја је коришћен софтверски алат, компаније ANSYS, за пројектовање електричних машина, Motor-CAD.

3.1. Подаци машине

Параметри статора су следећи:

- Број жлебова: 36
- Пречник лим пакета: 210 mm
- Пречним отвора статора: 137 mm
- Ширина зупца: 6.05 mm
- Дубина жлеба: 18 mm
- Радиус угла на дну жлеба: 4.1 mm
- Ширина отвора жлеба: 2 mm
- Угао отвора жлеба: 30°



Слика 4 – Изглед жлеба и једног дела лима статора

Параметри ротора су следећи:

- Број жлебова/штапова: 26
- Број полова: 4
- Ширина отвора штапа: 1 mm
- Дубина отвора горњег дела штапа: 0,4142 mm
- Ширина горњег дела зупца: 8,8415 mm
- Дубина горњег дела шипке: 6 mm
- Ширина доњег отвора шипке: 1 mm
- Дубина доњег отвора шипке: 2,7 mm
- Ширина доњег дела зупца: 8,64 mm
- Дубина отвора доњег дела штапа: 13 mm
- Ваздушни зазор: 0,5 mm
- Пречник вратила: 57 mm
- Пречник отвора на вратилу: 0 mm



Слика 5 – Изглед жлеба и једног дела лима ротора

Намотај:

- Трофазни
- Двослојни
- Број навојака по секцији: 13
- Навојни корак: 7,8,9
- Број паралелних грана: 1
- Веза намотаја: троугао

3.2. СТУДИЈА СЛУЧАЈА

3.2.1. Клизање и брзина

На самом почетку прилико задавања вредности синхроне брзине од 1500 o/min и фреквенције од 50 Hz се може константовати да се за различите навојне кораке добија различита номинална брзина машине, као и различите вредности клизања. У случају када је полни корак тр једнак навојном кораку секције у, 9,

клизање износи 4,59% и номинална брзина је 1431,13 o/min. Када се ради о тетивним секцијама, тј. секцијама са скраћеним навојним кораком добија се мање клизање и самим тим већа номинална брзина. У случају када навојни корак износи 8 вредност клизања је 4,37%, а номинална брзина износи 1434 o/min. У случају кад је скраћене још веће, ондосно када је навојни корак 7, вредност клизања се смањује на 3,94% и номинална брзина повећава на 1441 o/min.

3.2.2. Хармоници

У случају када је навојни корак у једнак полном кораку тр, може се рећи да су 3., 5., 15., 17. и 19. хармоник магнетнопобудне силе (мпс) истакнути. Основни хармони мпс достиже вредност 47,5 A/завоја.

У наредно случају при скраћењу навојног корака, у=8, трећи хармоник има мало пригушење у односу на случај у=тр. Пети има значајно пригушење, што је и циљ скраћења навојног корака секције. Изражени остају 15., 17. и 19., али они немају значајнијег утицаја на рад машине. Такође, примећује се да је вредност основног хармоника смањена на 47A/завоја.

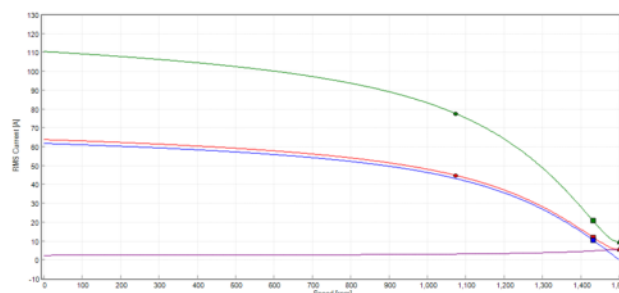
У последњем случају, у=7, трећи хармоник добија знатно пригушење у односу кад је у=тр. Највећи ефекат са овим скраћењем навојног корака је тај што је пети хармоник у потпуности пригушен и више нема никаквог утицаја. У овом случају вредност основног хармоника је 44,5 A/завоја.

Што се тиче резултатног навојног сачиниоца за случај у=тр, значајно су истакнути 17. и 19. хармоник и истакнути су 3.,15. и 21. хармоник. Скраћењем навојног корака су знатно пригушени 3.,15. и 21. хармоник, док истакнути остају 17. и 19. хармоник.

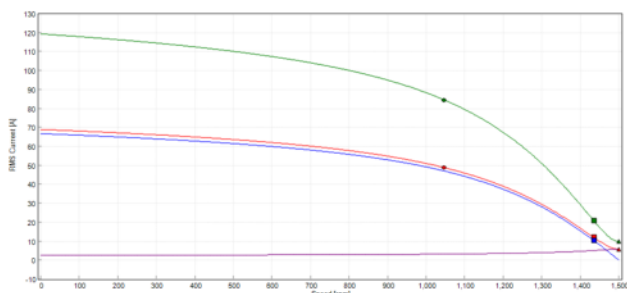
У свим варијантама навојног корака секције вредност основног хармоника резултатног навојног сачиниоца се креће у опсегу од 0,95 до 0,9 што је у складу са очекивањем.

3.2.3. Струје

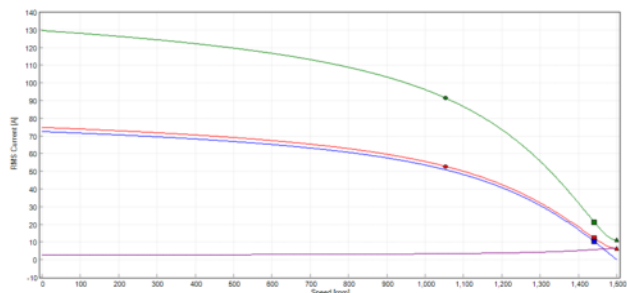
На сликама 6,7,8 зеленом бојом је означена линијска струје, црвеном бојом фазна струја, плавом бојом референта струја ротора, а љубичастом бојом струја магнећења.



Слика 6 – Карактеристика струје у зависности од брзине обртања за случај када је у=9



Слика 7 – Карактеристика струје у зависности од брзине обртања за случај када је $u=8$



Слика 8 – Карактеристика струје у зависности од брзине обртања за случај када је $u=7$

Са приказаних карактеристика на сликама 6,7,8 се јасно уочава да како се смањује навојни корак секције, тј. када је навојни корак мањи од полног корака полазне струја се повећава за око 10 A са сваким наредним скраћењем навојног корака. Струје при превалном моменту се са скраћењем секције повећавају за око 8 A. Номиналне струје су врло уједачене разликују се за 0,03 A, а струје магнетисања су повећавају за око 0,3 A са сваким скраћењем u .

3.2.4. Обртни моменат вратила

Када је $u=\tau_r$, полазни моменат износи 134,16 Nm, превални моменат, тј. највиши моменат које се развија, има вредност 198,52 Nm при брзини од 1074 o/min, а номинални моменат је 73,99 Nm.

За случај $u=8$, може се на основу преходне анализе струје закључити да ће полазни и превални моменат бити већи. Полазни моменат у том случају има вредност 151,75 Nm, превални моменат је 215,67 Nm при брзини од 1047 o/min, а номинални моменат је нешто мањи и има вредност 73,23 Nm.

У последњем случају $u=7$, моменат се мења као и у преходном случају $u=8$ у односу на $u=\tau_r$. Полазни моменат је 163,25 Nm, превални моменат износи 233,44 Nm при брзини од 1053 o/min, а номинални моменат је поново нешто мањи и има вредност 72,90 Nm.

3.2.5. Фактор снаге и ефикасност

За кавезну асинхронну машину се вредност фактора снаге $\cos \vartheta$ креће у опсегу 0,8 до 0,9. У случају испитане машине вредности су врло приближне, за $u=\tau_r$ и $u=8$ износи 0,85, док је за случај када је $u=7$ фактор снаге мало мањи и износи 0,84.

Ефикасност машине је у складу са очекивањима и расте са скраћењем корака намотавања. Када је полни

корак једнак навојном, ефикасност је 88,45 %, затим у случају $u=8$ ефикасност износи 88,81 % и на крају за $u=7$ ефикасност се повећава на вредност 89,34%.

4. ЗАКЉУЧАК

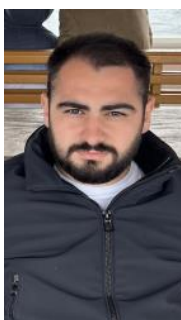
У раду су приказане предности и мане када је навојни корак секције мањи од полног корака. Најзначајнија добит од скраћивања навојног корака је та што се хармоници магнетнобудне силе пригушују, посебно што се пети хармоник може потпуно пригушити. Са друге стране јавља се проблем са повећањем полазне струје која је већа када се користи скраћен навојни корак него када је навојни корак секције једнак полног кораку. Полазне струје су у односу на номинале, за испитивану асинхронну машину снаге 11 kW, веће чак пет до шест пута.

Најнапредније савремено решење за решење проблема великих полазних струја и пуне контроле брзине представља претварач учестаности. Данас у свим модерним, потпуно аутоматизованим погонима стандардно се користе асинхроне машине са претварачем учестаности. Без обзира на његове могућности да добро искористи асинхроне машине, пуна контрола брзине је скоро увек основни захтев сваког погона. Користећи претварач учестаности добијено је низ предности. Неке од најбитнијих су уштеда енергије, оптимизација процеса, мањи трошкови одржавања и побољшано радно окружење.

5. ЛИТЕРАТУРА

- [1] Веран Васић, „Скрипта увод у електричне машине“, Нови Сад 2021.
- [2] Бранко Митраковић, Никола Љ. Николић, „Асинхроне машине“, Научна књига, Београд, 1991.
- [3] Емил Леви, Владан Вучковић, Владимир Стрезорски, „Основи електроенергетике – Електроенергетски претварачи“, ФТН Издаваштво, Нови Сад, 2013
- [4] Веран Васић, Борис Думнић, „Електроенергетски претварачи“, ФТН Издаваштво, Нови Сад, 2022.
- [5] Слободан Н. Вукосавић, „Електричне машине“, Академска мисао, Београд, 2010.
- [6] Веран Васић, Електричне машине 2, материјал са предавања, Факултет техничких наука у Новом Саду

Кратка биографија:



Данило Дробњак рођен је 1998. године у Новом Саду. Основне студије је успешно завршио на Факултету техничких наука у Новом Саду, из Електротехнике и рачунарства, на усмерењу Електричне машине и енергетска електроника. Мастер рад на истом факултету и усмерењу је одбранио 2023. године.

**HARDVERSKA IMPLEMENTACIJA RISC-V PROCESORA SA PROTOČNOM
OBRADOM KOJA PODRŽAVA RV32IM SKUP INSTRUKCIJA****A HARDWARE IMPLEMENTATION OF A RISC-V PROCESSOR WITH PIPELINE
PROCESSING THAT SUPPORTS THE RV32IM INSTRUCTION SET**

Nemanja Milić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj rad ukratko opisuje implementaciju RV32IM skup instrukcija koji podržava pet faza protočne obrade uz detekciju hazarda. Korištene su uobičajene faze protočne obrade: IF, ID, EXE, MEM i WB. Polazna tačka rada je bila inicijalna verzija RV32I procesora. Nastavilo se sa implementiranjem kompletnog I skupa instrukcija. Nakon toga je dodat i M skup instrukcija. Za implementaciju M skupa instrukcija, uzeta su u obzir dva načina pristupa realizacije. Prvi način koristi Booth-4 algoritam, a drugi DSP blokove sa Zybo ploče za instrukcije množenja. Oba pristupa daju iste logičke rezultate, a razlika je u performansama. Rad se završava sa upoređivanjem performansi procesora RV32I sa RV32IM (Booth-4 algoritam i DSP).

Ključne reči: RV32I, RV32IM, Booth-4 algoritam, DSP, Zybo

Abstract – This paper briefly describes an implementation of the RV32IM instruction set that supports five stages of pipeline processing with hazard detection. The usual pipeline stages were used: IF, ID, EXE, MEM and WB. The starting point of the work was the initial version of the RV32I processor. The implementation of the complete I set of instructions followed. After that, the M instruction set was added. For implementation of the M instruction set, two ways of approaching the realization are taken into account. Both approaches produce the same logical results, the difference being in performance. The first method uses the Booth-4 algorithm, and the second uses DSPs from the Zybo board for multiplication instructions. The paper ends with a comparison of the performance of the RV32I processor with the RV32IM (Booth-4 algorithm and DSP).

Keywords: : RV32I, RV32IM, Booth-4 algorithm, DSP, Zybo

1. UVOD

Arhitektura RISC-V procesora nastala je 2010. godine na Univerzitetu Berkley u cilju učenja i razvoja znanja iz oblasti digitalnih blokova.

Pre pomenute poboljšane verzije, 1990. godine nastala je verzija za edukativne svrhe u vidu RISC skup instrukcija DLX. Prvobitni cilj je bio da procesor podržava osnovne

operacije, da se pomoću odgovarajućeg programskog jezika nađe najjednostavniji način za implementaciju istog, da cena takvog rešenja bude što niža, a da performanse budu što bolje [1].

Do danas, različite arhitekture su našle široku primenu i mogu se kategorisati u dve glavne grupe na sledeći način:

RISC – Ovaj pristup koristi brojne jednostavne instrukcije za izvršavanje programa, pri čemu se ove instrukcije izvršavaju unutar jednog mašinskog ciklusa. Proces dekodiranja instrukcija je brz (zahvaljujući fiksnoj širini kodiranja uputstva), a implementacija obrade toka je jednostavna. Dizajn procesora zahteva manje tranzistora. Međutim, rezultujući kod ima tendenciju da bude veći i zauzima znatnu količinu RAM-a [2].

CISC – U navedenoj paradigmi, manja količina složenih instrukcija se koristi za izvršavanje programa, obuhvatajući više ciklusa takta. Dekodiranje i implementacija ovih instrukcija je vrlo složena (najvećim delom zbog promenljive širine kodiranja instrukcija) i zahteva veći broj tranzistora. Kao rezultat prethodno navedenog, veličina koda je smanjena [2].

CISC procesori su izvršavali manje instrukcija po programu u odnosu na RISC, ali je RISC procesorima u proseku trebalo mnogo manje taktova za izvršavanje instrukcija (CPI – Clocks Per Instruction). Tokom dugogodišnjeg rada sa pomenutim procesorima otkriveno je da CISC procesori izvršavaju otprilike upola manje instrukcija od RISC procesora, ali je njihov CPI bio u proseku šest puta veći [3].

I skup instrukcija predstavlja osnovni skup instrukcija koji svaki procesor podržava. Iako je osnovni skup, podržava širok opseg instrukcija. Instrukcije koje spadaju pod ovim skupom jesu one koje ispunjavaju aritmetičke logičke operacije, upisuju rezultate u registre, rade sa skokovima grana i memorijama.

M skup instrukcija podržava operacije množenja i deljenja. U nastavku će biti prikazane dve verzije implementacije. Prva podrška Booth-4 algoritam za množenje, dok druga verzija izvodi množenje pomoću DSP blokova sa Zybo ploče.

2. REALIZACIJA

Sve korištene komponente implementirane su u VHDL jeziku korišćenjem Vivado Design Suite alata. Prvo će biti prikazana realizacija I skupa instrukcija, a potom i M

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Ivan Mezei, red. prof.

skupa instrukcija koji je bio realizovan na dva načina. Dizajn je podeljen na datapath i controlpath celine.

2.1 I skup instrukcija

Pod I skupom pripadaju instrukcije koje ispunjavaju promenu vrednosti registra unutar registerske banke na osnovu poređenja između dve vrednosti. Takođe dolaze u obzir i instrukcije za čije izvršenje je neophodno korišćenje ALU jedinice. Što se tiče instrukcija za poređenje, one obično upisuju vrednost 1 u registar ako je uslov ispunjen, odnosno 0 u slučaju da nije. Koriste se za implementaciju if-else instrukcija i nekih drugih uslovnih operacija. U slučaju da nije ispunjen uslov, može se postići efikasnije izvršavanje koda. U nastavku će biti prikazana implementacija unapređene verzije procesora koji podržava I skup instrukcija. U tabeli 1 se može videti detaljan prikaz formata instrukcija I skupa.

Tabela 1. I skup instrukcija

imm[31:12]			rd	"0110111"	LUI	
imm[31:12]			rd	"0010111"	AUIPC	
imm[20:10:1][11:19:12]			rd	"1101111"	JAL	
imm[11:0]	rs1	"000"	rd	"1100111"	JALR	
imm[12:10:5]	rs2	rs1	"000"	imm[4:1:11]	BEQ	
imm[12:10:5]	rs2	rs1	"001"	imm[4:1:11]	BNE	
imm[12:10:5]	rs2	rs1	"100"	imm[4:1:11]	BLT	
imm[12:10:5]	rs2	rs1	"101"	imm[4:1:11]	BGE	
imm[12:10:5]	rs2	rs1	"110"	imm[4:1:11]	BLTU	
imm[12:10:5]	rs2	rs1	"111"	imm[4:1:11]	BGEU	
imm[11:0]	rs1	"000"	rd	"0000011"	LB	
imm[11:0]	rs1	"001"	rd	"0000011"	LH	
imm[11:0]	rs1	"010"	rd	"0000011"	LW	
imm[11:0]	rs1	"100"	rd	"0000011"	LBU	
imm[11:0]	rs1	"101"	rd	"0000011"	LHU	
imm[11:5]	rs2	rs1	"000"	imm[4:0]	SB	
imm[11:5]	rs2	rs1	"001"	imm[4:0]	SH	
imm[11:5]	rs2	rs1	"010"	imm[4:0]	SW	
imm[11:0]	rs1	"000"	rd	"0010011"	ADDI	
imm[11:0]	rs1	"010"	rd	"0010011"	SLTI	
imm[11:0]	rs1	"011"	rd	"0010011"	SLTIU	
imm[11:0]	rs1	"100"	rd	"0010011"	XORI	
imm[11:0]	rs1	"110"	rd	"0010011"	ORI	
imm[11:0]	rs1	"111"	rd	"0010011"	ANDI	
"0000000"	shamt	rs1	"001"	rd	"0010011"	SLLI
"0000000"	shamt	rs1	"101"	rd	"0010011"	SRLI
"0100000"	shamt	rs1	"101"	rd	"0010011"	SRAI
"0000000"	rs2	rs1	"000"	rd	"0110011"	ADD
"0100000"	rs2	rs1	"000"	rd	"0110011"	SUB
"0000000"	rs2	rs1	"001"	rd	"0110011"	SLL
"0000000"	rs2	rs1	"010"	rd	"0110011"	SLT
"0000000"	rs2	rs1	"011"	rd	"0110011"	SLTU
"0000000"	rs2	rs1	"100"	rd	"0110011"	XOR
"0000000"	rs2	rs1	"101"	rd	"0110011"	SRL
"0100000"	rs2	rs1	"101"	rd	"0110011"	SRA
"0000000"	rs2	rs1	"110"	rd	"0110011"	OR
"0000000"	rs2	rs1	"111"	rd	"0110011"	ANDI

Unapređena aritmetičko logička jedinica sada podržava operacije ADD, SUB, SLT, SLTU, XOR, ILI, I, SLL, SRL i SRA. Immediate blok je unapređen kako bi podržavao sve nove instrukcije koje sadrži u sebi imm polje. Instrukcije koje sadrži imm polje su I, S, B, U i J tipa.

Branch control blok je implementiran u datapath delu, tačnije u ID fazi. Svrha ovog bloka jeste u cilju implementacije instrukcija B tipa. Instrukcije koje spadaju pod B tipom jesu BEQ (*Branch if equal*), BNE (*Branch if not equal*), BLT (*Branch if less than*), BGE (*Branch if greater or equal*), BLTU (*Branch if less than unsigned*) i BGEU (*Branch if greater or equal unsigned*).

Load Store blok je takođe implementiran u datapath celini. Postoji dve instance ovog bloka. Jedna instanca je u MEM dok se druga instanca nalazi u WB fazi. Cilj ovog

bloka jeste potreba instrukcija za skladištenje podataka u memoriji, kao i čitanje iz iste. Instrukcije za skladištenje podataka su SB (*Store byte*), SH (*Store half word*) i SW (*Store word*). Instrukcije za čitanje podataka iz memorije su LB (*Load byte*), LH (*Load half word*), LW (*Load word*), LBU (*Load byte unsigned*) i LHU (*Load half word unsigned*).

U slučaju bajt instrukcija, uzima se najnižih 8 bita, dok će na prvih 24 bita biti podatak predstavljen isti kao što je sedmi bit, nakon toga sledi konkatencija ta dva dela podatka. Tako se dobija 32-bitni izlazni podatak. U slučaju half word instrukcije, slična je priča samo što se sad uzima 16 nižih bita i spaja sa podatkom od 16 bita, koji će biti predstavljen kao sve nule ili jedinice u zavisnosti od 15. bita. U slučaju word instrukcije, koristi se ceo 32-bitni podatak koji je došao na ulaz bloka. Za neoznačene (unsigned) instrukcije konkatencija se vrši uvek sa nulama. Ovde je potrebno napomenuti ulogu bloka memorija podataka. Unutrašnja struktura je podeljena na 8-bitnu ćeliju kako bi se omogućilo pisanje ili čitanje jednog bajta, pola reči (dva bajta) ili cele reči (četiri bajta).

2.2 M skup instrukcija

Na postojeću, unapređenu implementaciju RV32I procesora je dodat M skup instrukcija. Instrukcije koje se podrazumevaju pod ovim skupom jesu instrukcije množenja i deljenja. Najveća promena u ovom dodatku jeste implementacija množača i deljenika koji se nalaze u sklopu ALU jedinice. U nastavku će biti prikazane dve verzije implementacije. Prva podržava Booth-4 algoritam za množenje, dok druga verzija izvodi množenje pomoću DSP blokova. U tabeli 2 je moguće videti instrukcije koje spadaju pod M skupom.

Tabela 2. M skup instrukcija

"0000001"	rs2	rs1	"000"	rd	"0110011"	MUL
"0000001"	rs2	rs1	"001"	rd	"0110011"	MULH
"0000001"	rs2	rs1	"010"	rd	"0110011"	MULHSU
"0000001"	rs2	rs1	"011"	rd	"0110011"	MULHU
"0000001"	rs2	rs1	"100"	rd	"0110011"	DIV
"0000001"	rs2	rs1	"101"	rd	"0110011"	DIVU
"0000001"	rs2	rs1	"110"	rd	"0110011"	REM
"0000001"	rs2	rs1	"111"	rd	"0110011"	REMU

2.2.1 Booth-4 algoritam verzija

Booth-4 algoritam je algoritam množenja koji se koristi za efikasno izvođenje binarnog množenja celih brojeva sa ili bez predznaka. Smanjuje broj operacija sabiranja i oduzimanja potrebnih za množenje korišćenjem oblika predstavljanja sa predznakom.

Standardni Booth-4 algoritam je algoritam za binarne brojeve sa znakom, koji se često naziva i Butov algoritam množenja. Postoji i varijanta poznata kao Butov algoritam za neoznačene binarne brojeve. Obe verzije dele osnovnu ideju optimizacije množenja [4].

Množač je implementiran u sklopu ALU jedinice. Ovaj blok je neophodan zbog realizacije instrukcija MUL (*Multiply*), MULH (*Multiply High*), MULHSU (*Multiply High Signed Unsigned*) i MULHU (*Multiply High Unsigned*). Za samu implementaciju se koristi Booth-4 algoritam. Prednost je smanjenje kašnjenja u poređenju sa

normalnim procesom množenja. Za klasičan proces množenja kao što je već objašnjeno, potrebno je onoliko množenja koliko množilac ima bita.

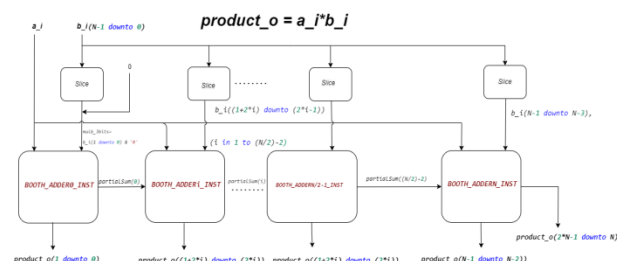
Za svaku dobijenu parcijalnu sumu potrebno je izvršiti sabiranje, što zahteva mnogo hardvera i može biti sporo.

U tabeli 3 prikazane su vrednosti Booth-4 algoritma prema kojima je implementiran poboljšani množač.

Tabela 3. Booth-4 algoritam encoding

X_{i+1}	X_i	X_{i-1}	Radix-4 Booth encoding
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	2
1	0	0	-2
1	0	1	-1
1	1	0	-1
1	1	1	0

Na slici 1 je prikazan blok dijagram množača koji koristi pomenuti algoritam.



Slika 1. Množač koji koristi Booth-4 algoritam

Množač se sastoji od 17 Booth sabirača. Treba obratiti pažnju na desnu stranu tabele Booth-4 algoritma. Brojevi 0, 1 i 2 se mogu predstaviti sa 2 bita (00, 01 i 10), zbog toga svaki Booth sabirač ima izlaz product_o od dva bita. Konkatencijom svakog product_o signala od svih 17 sabirača dobija se podatak od donjih 34 bita, a parcijalna suma poslednjeg sabirača (product_o(2*N-1 downto 0)) predstavlja gornjih 34 bita proizvoda množača.

Svi pomenuti biti u celini predstavljaju izlaz množača od kog se uzimaju donji ili gornji biti u zavisnosti koja je instrukcija u pitanju. B operand odlučuje koji će se koeficijent za množenje biti izabran.

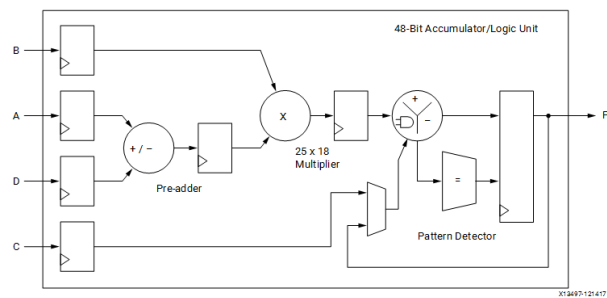
2.2.2 DSP verzija

Najsloženiji blok za izračunavanje na Xilinx FPGA ploči je DSP48 blok koji se može videti na slici 2. DSP48 blok je aritmetičko-logička jedinica (ALU) ugrađena u strukturu FPGA i sastoji se od lanca sačinjena od tri različita bloka.

Pomenuti lanac u DSP48 sadrži blok za sabiranje/oduzimanje povezan sa množačem koji je dalje povezan sa konačnim blokom za sabiranje/oduzimanje/akumulaciju. Pomenuti lanac omogućava jednoj jedinici DSP48 da implementira funkcije oblika [5]:

$$P = B \times (A + D) + C \quad (1)$$

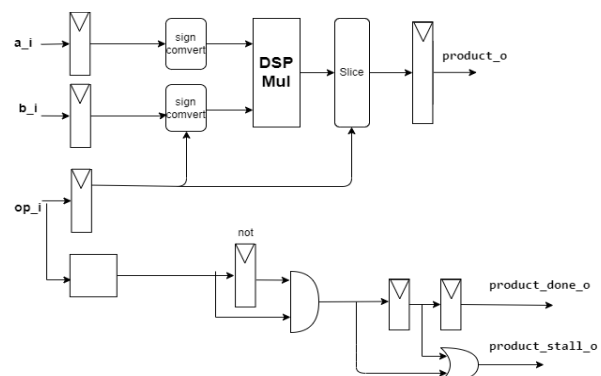
$$P += B \times (A + D) \quad (2)$$



Slika 2. Xilinx DSP48 [5]

Ideja ovog množača jeste što minimalnija upotreba hardverskih resursa u cilju dobijanja tačnih rezultata množenja. Sa korišćenjem DSP48 blokova na Zybo ploči nije potreban dodatni hardver, čime se direktno utiče na povećanje radne frekvencije procesora.

Ulazni operandi širine 32 bita se koriste za množenje. Rezultat je širine 64 bita. Potrebna su 2 DSP modula za svaku MUL instrukciju. Pošto će množač da podržava 4 MUL instrukcije, u rezultatima se može primetiti da se koristi 8 DSP modula. DSP48 podržava akumulaciju od 48 bita, dok je u ovom slučaju potrebna podrška za 64 bita. Iz tog razloga je potrebno 2 DSP modula po jednoj MUL instrukciji. Na slici 3 prikazana je blok šema pomenutog množača.



Slika 3. Blok šema množača koji koristi DSP

Ulazi u blok predstavljaju operandi a i b širine 32 bita, kao i op koji predstavlja jednu od četiri instrukcije. Izlazi množača jesu 32-bitni rezultat množenja kao i signali koji signaliziraju završetak operacije množenja, odnosno stall signal za množenje u slučaju da operacija nije završena.

Ovde se može primetiti činjenica da se operacija množenja ne izvršava samo u jednom taktu tokom EXE faze, nego u dva kloka. Naime, dodavanjem dva uzastopna registra utiče se na povećanje frekvencije. Propagaciono kašnjenje signala preko Bulove logike između dva uzastopna registracija u nizu ograničavanje brzine takta.

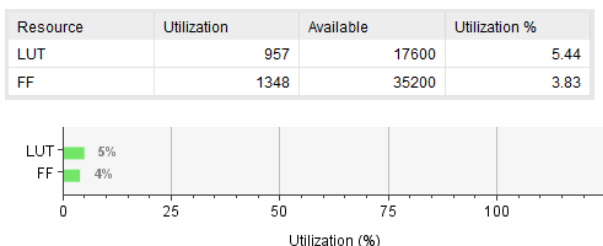
3. REZULTATI

U ovoj sekciji prikazani su rezultati tri verzije procesora:

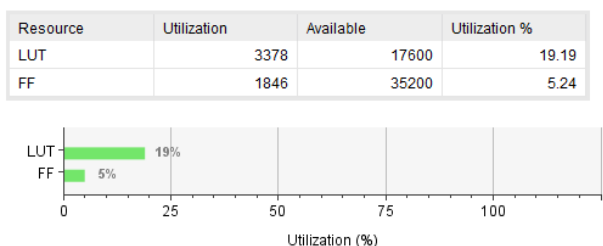
- Inicijalna verzija procesora RV32I
- RV32IM koji koristi Booth-4 algoritam za množenje
- RV32IM koji koristi DSP sa Zybo ploče za množenje

3.1 Iskorišćenost resursa

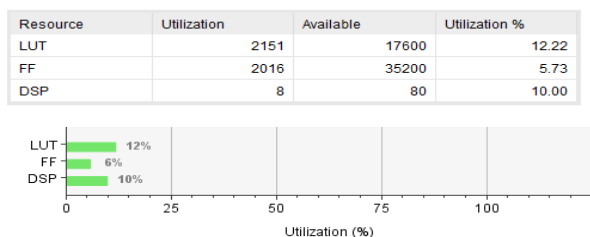
U ovoj sekciji će biti prikazan uvid koliko je bilo iskoršćeno resursa za svaku verziju procesora. Na slikama 4, 5 i 6 se redom može videti iskorišćenost resursa za procesore RV32I, RV32IM procesor koji koristi Booth-4 algoritam kao i za RV32IM procesor koji koristi DSP.



Slika 4. Iskorišćenost resursa RV32I



Slika 5. Iskorišćenost resursa RV32IM koji koristi Booth-4 algoritam za množenje



Slika 6. Iskorišćenost resursa RV32IM koji koristi DSP sa Zybo ploče za množenje

3.2 Maksimalna učestanost rada

Za sve tri verzije procesora je uzeta klok perioda od 10ns sa pretpostavkom da će raditi na 100MHz. Oduzimanje vremena Worst Negative Slack od pomenute klok periode dobija se konačna vrednost vremena koja se koristi za dobijanje maksimalne učestanosti.

$$f = 1/T \quad (3)$$

Inicijalna verzija procesora RV32I:
 $1/13.1 = 76.3\text{MHz}$

RV32IM koji koristi Booth-4 algoritam za množenje:
 $1/47.6 = 21\text{MHz}$

RV32IM koji koristi DSP sa Zybo ploče za množenje:
 $1/8.978 = 112.3\text{MHz}$

Očigledno je da inicijalna verzija procesora, zahteva najmanje hardvera. Razlog je što podržava samo 4 instrukcije iz I skupa, pa samim tim i implementacija nije previše zahtevna. RV32IM procesor koji koristi Booth-4 algoritam, zahteva najviše hardvera od sve tri verzije. Razlog tome leži u implementiranju algoritma koji je suviše zahtevan. Poslednja verzija procesora koristi

dodatni DSP hardver za množenje, ali i dalje ne zahteva toliki kapacitet kao druga verzija procesora.

Što se tiče radne učestalosti, inicijalna verzija iako nije previše zahtevana, nije dostigla postavljenih 100MHz. Najlošije rezultate je pokazala druga verzija sa Booth-4 algoritmom. Velika iskorišćenost resursa, kao i dugačke kritične putanje su dovele do loših rezultata.

Može se zaključiti da najbolje rezultate daje poslednja verzija procesora koja koristi DSP množenje. Razlog počiva na činjenicama da DSP resursi već postoje na korišćenju Zybo ploči, stoga nema potrebe za dodatnim hardverom, što dovodi do velikih ušteda i povećanja radne učestalosti. Ova verzija procesora je ispunila očekivanje i početnu pretpostavku.

4. ZAKLJUČAK

Glavni zadatak rada koji je autor trebalo da ispuni, uspešno je odrađen. Implementirani su I i M skup instrukcija za RISC-V arhitekturu. Poslednja verzija dizajna u potpunosti podržava arhitekturu RV32IM.

Implementacija M skup instrukcija je izvedena na dva načina. Prva verzija podržava Booth-4 algoritam za operaciju množenja, dok druga verzija za operaciju množenja koristi DSP blokove koji se nalaze na razvojnoj Zybo ploči. Rezultat rada se ogleda u poređenju iskorišćenosti resursa kao i maksimalne radne učestanosti između pomenute dve implementirane verzije sa inicijalnom verzijom procesora. DSP verzija RV32IM procesora daje rezultate u najbržem mogućem roku. Nakon nje sledi RV32I verzija procesora, dok Booth-4 verzija RV32IM procesora omogućava rezultate u najsporijem roku u poređenju sa prethodne dve verzije. Što se tiče potrebnog hardvera za realizaciju procesora, po tom pitanju je najmanje zahtevna RV32I verzija, a prate je DSP i na kraju Booth-4 verzija RV32IM procesora.

5. LITERATURA

- [1] <https://en.wikipedia.org/wiki/RISC-V>, pristupljeno septembar 2023.
- [2] https://www.elektronika.ftn.uns.ac.rs/napredni-mikroprocesorski-sistemi/wp-content/uploads/sites/103/2018/03/Vezba1_Uvod_u_RIS-C-V_arhitekturu.pdf, pristupljeno septembar 2023.
- [3] D. Patterson, "Reduced Instruction Set Computers Then and Now Computer, vol. 50, no. 12, pp. 10-12, 2017.
- [4] https://en.wikipedia.org/wiki/Booth%27s_multiplication_algorithm, pristupljeno avgust 2023.
- [5] https://www.xilinx.com/htmldocs/xilinx2017_4/sdacce1_doc/uwa1504034294196.html, pristupljeno jul 2023.

Kratka biografija:



Nemanja Milić rođen je u Novom Sadu 1994. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i telekomunikacije odbranio je 2023.god.
 Kontakt: nemanja.milic.ns@gmail.com

UTICAJ RADA SOLARNIH ELEKTRANA NA PARAMETRE KVALITETA NAPONA U DISTRIBUTIVNOJ MREŽI**THE INFLUENCE OF THE OPERATION OF SOLAR POWER PLANTS ON THE VOLTAGE QUALITY PARAMETERS IN THE DISTRIBUTION NETWORK**

Oliver Krezović, Dejan Jerkan, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu su prikazani rezultati merenja parametara kvaliteta napona distributivne mreže na mestima priključena male solarne elektrane na teritoriji operatora distributivnog sistema ZP „Elektrokrajina“ a.d. Banja Luka. Merenje je izvršeno uređajem MI (Metrel) 2292, dok je obrada podataka izvršena korišćenjem programa PowerLink prema standardu EN 50160.

Ključne reči: parametri kvaliteta napona, distributivna mreža, male solarne elektrane, MI (Metrel) 2292

Abstract The paper presents the results of measuring the voltage quality parameters of the distribution network at the places where small solar power plants are connected to the territory of the DSO ZP "Elektrokrajina" a.d. Banja Luka. The measurement was performed with the device MI (Metrel) 2292, while the data processing was performed using the PowerLink program according to the EN 50160 standard.

Keywords: voltage quality parameters, distribution network, small solar power plants, MI (Metrel) 2292

1. OBNOVLJIVI IZVORI ENERGIJE

S obzirom na prirodnu obnovljivost, izvori energije se dele na neobnovljive izvore energije i obnovljive izvore energije. Neobnovljivi izvori energije (fosilna goriva) su izvori energije koji se ne mogu regenerisati niti ponovo proizvesti. Njihovim sagorevanjem nastaju čađ, prašina, buka, štetne hemijske reakcije koji prouzrokuju kisele kiše, a najviše doprinose emisije štetnih gasova u atmosferu koji su posledica klimatskih promena.

Pod pojmom „obnovljivi izvori energije“ (OIE) podrazumeva se energija koja je stvorena iz prirodnih izvora. OIE danas proizvode samo mali deo ukupne svetske energije. Neobnovljivih izvora energije ima sve manje i njihove rezerve su na izmaku. Njihov štetni uticaj na životnu sredinu je sve izraženiji u poslednjih nekoliko godina.

Samim tim, udeo OIE u ukupnoj proizvodnji energije će u budućnosti morati da se poveća. Pet glavnih tipova OIE su: sunce, vetar, biomasa, geotermalna energija i voda (hidroenergija) [1] [2].

NAPOMENA:

Ovaj rad proistekao je iz master rada, čiji mentor je bio dr Dejan Jerkan, docent.

1.1. Solarna energija i solarne elektrane

Energija sunčevog zračenja (solarna energija) je energija elektromagnetnog zračenja sunca, koja dospeva na Zemljinu površinu. Ona predstavlja osnovu gotovo svih drugih izvora energije na zemlji. Kao akumulisani oblik solarne energije javljaju se fosilna goriva. Kao indirektni oblik pojave sunčeve energije na zemlji postoji energija vetra, energija vodenih tokova i energija biomase. Za proizvodnju toplotne energije iz sunčevog zračenja obično se koriste solarni kolektori za pripremu tehničke tople vode, dok se proizvodnja električne energije se najjednostavnije obavlja uz pomoć solarnih (fotonaponskih) panela, i takav vid transformacije je praktično postao sinonim za solarne elektrane.

Svaki fotonaponski sistem, tj. solarna elektrana se sastoji od različitih elemenata. Osnovni element svakog sistema predstavljaju solarni (fotonaponski) paneli, a pored njih prisutne su i pomoćne komponente – regulatori, baterije, invertori, noseće strukture i sl.

Za pretvaranje sunčeve energije u električnu energiju potrebna je solarna (fotonaponska) ćelija koja je osnovna komponenta svakog fotonaponskog panela. U solarnoj ćeliji se sunčeva energija pretvara direktno u jednosmerni napon (12 V). Princip rada solarne ćelije (solarnih panela) se zasniva na fotonaponskom efektu, pri kojem poluprovodnici pretvaraju osvetljaj u električnu struju. Postoje tri preovlađujuća tipa solarnih ćelija i to: monokristalne ćelije, polikristalne ćelije i ćelije od amornog (nekristalnog) silicijuma. Odnos proizvedene električne snage i snage kojim sunce zrači na solarnu ćeliju naziva se stepen korisnog dejstva. Ova veličina, za standardne ćelije na tržištu, iznosi 12-18%, u zavisnosti od tipa izrade.

Invertor (Pretvarač) predstavlja najosetljiviji deo solarne elektrane. Primarna funkcija mu je pretvaranje jednosmernog napona od 12 V, koji proizvode solarni paneli, u stabilni naizmenični napon, sinusnog oblika, sinhronizovan sa distributivnom mrežom [3] [4].

2. KVALITET ELEKTRIČNE ENERGIJE

Električna energija je proizvod koji se po svojim specifičnim karakteristikama razlikuje od bilo kog drugog proizvoda. Najčešće se proizvodi daleko od mesta potrošnje, a pored toga do mesta potrošnje dolazi preko kilometarskih nadzemnih i kablovskih vodova prenosne i distributivne mreže prošavši pri tome niz transformacija s jednog na drugi naponski nivo. Prolaskom kroz taj put može doći do narušavanja kvaliteta električne energije

koji je neophodan zbog kvalitetnog rada električnih prijemnika/uređaja.

Snabdevanje električnom energijom u Republici Srpskoj vrši se u skladu sa Zakonom o električnoj energiji, Opštim uslovima za isporuku i snabdevanje električne energije, Tarifnim sistemom za prodaju električne energije i ugovorom koji zaključuju snabdevač i kupac. Snabdevanje električnom energijom podrazumeva isporuku i prodaju električne energije.

Kvalitet električne energije u distributivnim mrežama zavisi od:

- karakteristika uređaja kupaca priključenih na mrežu, posebno od industrijskih potrošača sa pogonima u kojima dominiraju energetske pretvarače, pogona sa velikim udarnim opterećenjima, uređaja koji svoju funkciju zasnivaju na dejstvu električnog luka, snažnih jednofaznih prijemnika;
- topologije i karakteristika same mreže (dužina i presek vodova, stepen opterećenja vodova i transformatora, nesimetričnost opterećenja po fazama);
- naponskih prilika na vezi sa prenosnom mrežom kao i elektranama priključenim na mrežu.

Merenje kvaliteta električne energije provodi se prikupljanjem merenih veličina (napona, struje, frekvencije) unutar određenog vremenskog intervala. Prikupljeni podaci se, uz raspoloživu programsku podršku, računarski obrađuju, pri čemu se rezultati izvršenih merenja upoređuju sa standardnim propisanim referentnim vrednostima i utvrđuje nivo njihovog odstupanja od vrednosti koje su standardima definisane kao dopuštena.

Evropski komitet za standardizaciju u elektrotehnici CENELEC doneo je standard EN 50160 – Karakteristike napona u javnim elektrodistributivnim mrežama, kojom se određuju karakteristike napona na NN i SN mreži i koja je opšteprihvaćena u Evropskoj Uniji. Ovaj standard definiše bitne veličine distributivnog napona na mestu predaje potrošaču u javnim NN i SN mrežama pri normalnim pogonskim uslovima. Svrha ovog standarda je da utvrdi i opiše karakteristike distributivnog napona u pogledu frekvencije, veličine, talasnog oblika, simetrije triju napona faznih provodnika [5] [6].

3. PRIKLJUČENJE DISTRIBUIRANIH ENERGETSKIH RESURSA NA DISTRIBUTIVNU MREŽU REPUBLIKE SRPSKE

Pod pojmom „Distribuirana proizvodnja smatra se proizvodnja i skladištenje električne energije koju obavljaju različiti mali uređaji povezani na distributivnu mrežu koji se nazivaju distribuirani energetske resursi (DER). Pod ovim pojmom podrazumevaju se sva postrojenja za proizvodnju električne energije iz OIE (elektrane) [7].

Distributivnom mrežom Republike Srpske operativno upravlja pet operatora distributivnog sistema (ODS). DER-ovi u Republici Srpskoj se na distributivnu mrežu priključuju spram „Pravilnika o uslovima za priključenje elektrana na elektrodistributivnu mrežu Republike Srpske“. Ovaj pravilnik objedinjuje zahteve važećih zakona, tehničkih propisa, pravilnika, preporuka i priznatih svetskih standarda u ovoj oblasti, definišući minimalne uslove koje mora ispuniti Proizvođač kako bi projektovao,

izgradio, priključio i vodio u pogonu elektranu paralelno sa distributivnom mrežom nazivnog napona 0,4 kV, 10 kV, 20 kV ili 35 kV, s ciljem da se izbegnu, odnosno ograniče negativna povratna dejstva na distributivnu mrežu i postojeće korisnike.

Pravilnik se odnosi na priključenje svih tipova elektrana koje proizvode električnu energiju iz OIE na distributivnu mrežu. Njime su propisani osnovni kriterijumi za priključenje elektrane, funkcionalni zahtevi za rad elektrane, osnovni tehnički zahtevi za izvođenje priključka elektrane, tehnički zahtevi za merno mesto, zaštitni uređaji i uslovi za pogon elektrane. Takođe, propisan je i redosled postupaka za izdavanje dokumenata za priključenje elektrane na distributivnu mrežu [8].

Zaključno sa 01.07.2023. godine, na distributivnu mrežu Republike Srpske priključeno je ukupno 353 elektrane, ukupne instalisane snage 146 MW, i to:

- 54 MHE, ukupne instalisane snage 101,5 MW,
- 296 MSE, ukupne instalisane snage 42,4 MW i
- 2 KPB i 1 BGE, ukupne instalisane snage 2,1 MW [9].

4. MERENJE KVALITETA ELEKTRIČNE ENERGIJE U MH “ELEKTROPRIVREDA REPUBLIKE SRPSKE” NA MESTIMA PRIKLJUČENJA DISTRIBUIRANIH ENERGETSKIH RESURSA

Za potrebe analiza u okviru ovog rada merenja su izvršena u mestu priključenja MSE „BLC“ na teritoriji ODS-a ZP „Elektrokrajina“ a.d. Banja Luka. Merna lokacija je odabrana sa ciljem analize uticaja MSE na kvalitet električne energije u distributivnoj mreži.

Generalno, uticaj koji mogu imati DER-ovi na distributivnu mrežu sledeći:

- Zbog sposobnosti generatora da proizvode i troše reaktivnu energiju, u distributivnim mrežama može doći do trajno povišenih ili sniženih napona koji ugrožavaju rad ostalih korisnika na mreži, ali mogu izazvati i prorađu zaštitnih uređaja, što za posledicu ima preklide u napajanju potrošača;
- Priključenje na mrežu ili isključenje generatora sa mreže može izazvati propade ili poraste napona u mreži;
- DER-ovi u distributivnim mrežama izazivaju pojavu flikera – dugotrajnih fluktuacija napona, koji takođe imaju negativan uticaj na rad opreme;
- DER-ovi i njihova prateća oprema u distributivnoj mreži izazivaju harmonijsku distorziju napona;
- Priključenjem DER-ova povećava se snaga kratkog spoja distributivne mreže, što pozitivno utiče na kvalitet električne energije, ali ukoliko je nova snaga kratkog spoja veća od one na koju je oprema dimenzionisana, izaziva dodatne troškove;
- Ako se na mrežu spoji više jednofaznih generatora koji nisu izbalansirani po fazama, dolazi do pojave nesimetrije u mreži sa negativnim uticajem na ostale korisnike.

Pored navedenih uticaja DER-ova na distributivnu mrežu, loš kvalitet električne energije u mreži ima negativan uticaj na rad DER-ova, što može dovesti do neželjenih ispada generatora sa mreže.

Merenje je izvršeno u periodu 17.07.2023-24.07.2023. godine mernim uređajem MI (Metrel) 2292 (Slika 1), dok

je obrada podataka izvršena korišćenjem programa PowerLink prema standardu EN 50160.



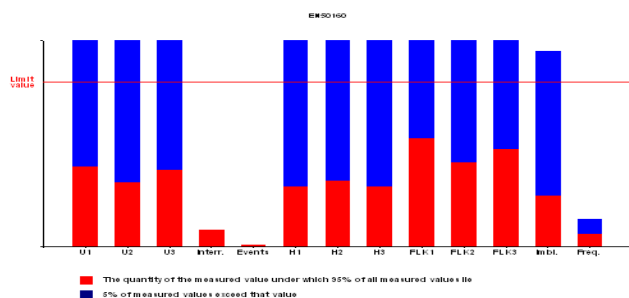
Slika 1. MI (Metrel) 2292 [10]

4.1. Analiza rezultata za mernu lokaciju – mesto priključenja MSE „BLC“

MSE „BLC“ nalazi se u Banja Luci, na objektu visokoškolske ustanove „Banja Luka Koledž“. Instalirana snaga ove MSE je 20 kW, a elektrana je priključena na NN distributivnu mrežu 2014. godine i predstavlja jednu od prvih elektrana koje su priključene na distributivnu mrežu Republike Srpske.

Energiju koju proizvede ova elektrana, oko 21 MWh godišnje, otkupljuje Operator sistema podsticaja u sklopu MH ERS Matično preduzeće.

Na Slici 2. grafički je dat sumarni prikaz rezultata merenja.



Slika 2. Grafički sumarni prikaz rezultata merenja [9]

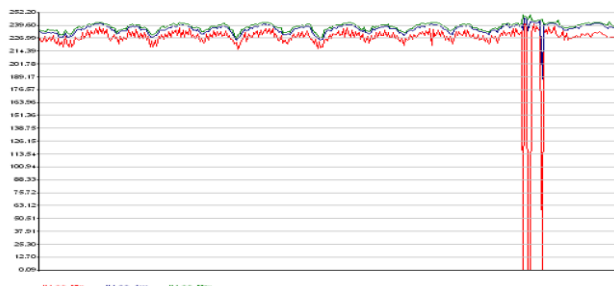
Crvena linija predstavlja graničnu vrednost parametara prema standardu EN 50160, koja mora biti zadovoljena u 95% mernog perioda (7 dana). Crveni stubići prikazuju vrednosti koje su bile postignute tokom 95% posmatranog vremena, u odnosu na preporučenu vrednost. Plavi stubići prikazuju vrednosti koje su bile zabeležene u ostalih 5% posmatranih vremenskih intervala.

Na bazi grafičkog sumarnog prikaza rezultata merenja na mestu priključenja ove MSE zaključeno je da kvalitet napona, tokom sedmodnevnog perioda posmatranja, u potpunosti zadovoljava zahteve propisane standardom EN 50160.

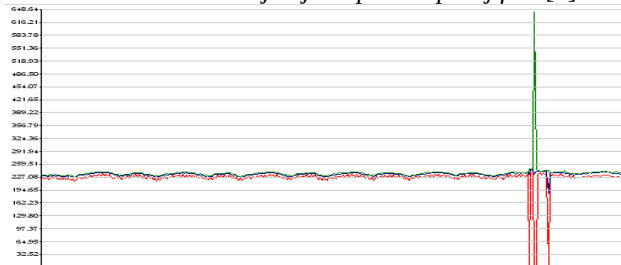
U nastavku je analiziran svaki od parametara kvaliteta napona pojedinačno.

Dijagrami minimalne, srednje i maksimalne efektivne vrednosti napona po fazama prikazani su na slikama 3, 4 i 5, pri čemu minimalne i maksimalne veličine predstavljaju najmanje i najveće efektivne vrednosti napona zabeležene u desetominutnom intervalu.

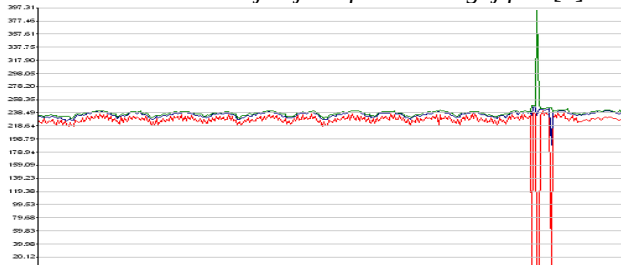
Za ocenu kvaliteta, potrebno je poznavati srednje vrednosti napona u desetominutnom mernom intervalu, pri čemu te vrednosti trebaju biti u granicama od 207 V do 253 V.



Slika 3. Prikaz varijacija napona u prvoj fazi [9]



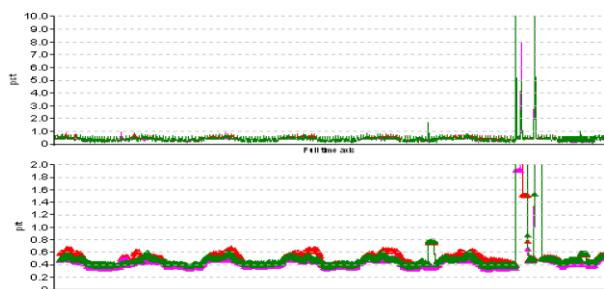
Slika 4. Prikaz varijacija napona u drugoj fazi [9]



Slika 5. Prikaz varijacija napona u trećoj fazi [9]

Posmatrajući prethodno prikazana tri dijagrama, može se uočiti da su varijacije napona značajnije izražene, ali i da je napon po pitaju varijacija tokom perioda posmatranja u 95% intervala bio u propisanim granicama.

Takođe, sa dijagrama se može uočiti da su 23.07.2023. godine zabeleženi naponski događaji koji nisu uticali na dozvoljene granice u 95% intervalu propisanom standardom EN 50160. Napon koji je uređaj zabeležio bio je izuzetno nizak, te se ovaj događaj može okarakterisati kao kratkotrajan naponski prekid trajanja reda minute. Tokom perioda posmatranja nije bilo propada napona.



Slika 6. Dugotrajni i kratkotrajni flikeri (po fazama) [9]

Na Slici 6. prikazani su indeksi kratkotrajne (Pst) i dugotrajne (Plt) jačine treperenja (flikeri). Prema standardu EN 50160 indeks dugotrajne jačine treperenja ne smije prelaziti vrednost $Plt = 1$ u 95% vremenskog intervala.

Na grafičkom prikazu sa Slike 5. može se videti da su vrednosti indeksa dugotrajne Plt i kratkotrajne Pst jačine treperenja po fazama, u periodu kada su se dogodili ranije pomenuti kratkotrajni prekid i preveliki porasti napona,

znatno veće od granične vrednosti 1. Prema raznim preporukama u pogledu praćenja kvaliteta napona za regulatorne svrhe, predlaže se markiranje intervala u kojim se dogodio naponski događaj (prekid napajanja, naponski propad ili porast napona), tako da se vrednosti dobijene u ovim intervalima ne koriste u statistici prilikom poređenja sa graničnim vrednostima propisanim standardom EN 50160. U skladu sa standardom, ranije pomenute vrednosti indeksa $Plt > 1$ se izbacuju iz statistike zbog markiranja naponskog prekida/porasta.

Iz programskog paketa se mogu pročitati sledeće vrednosti indeksa Plt u 95% vremenskom intervalu: faza $L1=0,66$; $L2=0,51$ i faza $L3=0,59$, te se nakon završenog markiranja može izvesti zaključak da su indeksi kratkotrajnog i dugotrajnog treperenja zadovoljili standard EN 50160, odnosno u granicama su standardom dopuštenih vrednosti.

Kada je reč o nesimetriji napona, korišćenjem programa PowerLink, moguće je očitati prosečnu vrednost tokom posmatranog perioda. Analizom prikupljenih podataka o merenim veličinama na mestu priključenja ove MSE u 95% mernom intervalu, nesimetrija napona iznosi 0,61%. Prema standardu EN 50160, dozvoljena granična vrednost koja se ne sme preći iznosi 2%, te je za ovu mernu lokaciju zahtev standarda zadovoljen.

Najveća vrednost THD faktora u 95% perioda merenja jednaka 2,81% Un ($L2$), što je ispod dozvoljene vrednosti od 8% Un (vrednost THD faktora u 100% perioda merenja jednaka je 3,05% Un , takođe manje od 8% vrednosti nominalnog napona) [9].

5. ZAKLJUČAK

U radu su opisani obnovljivi izvori energije, sa posebnim osvrtom na solarnu energiju. Uz to, objašnjen je i način dobijanja električne energije iz solarne energije uz pomoć fotonaponskih sistema (solarnih elektrana).

Takođe, akcenat je dat i na kvalitet električne energije koji se zahteva od strane svih krajnjih korisnika ovog proizvoda. Prikazan je i uticaj koji distribuirani energetske resursi mogu imati na parametre kvaliteta električne energije.

Na kraju su prikazani rezultati obavljenog sedmodnevnog merenja parametara kvaliteta napona za mernu lokaciju – mesto priključenja MSE „BLC“ u Banja Luci uz pomoć mernog uređaja MI (Metrel) 2292. Obrada podataka je izvršena korišćenjem programa PowerLink prema standardu EN 50160. Analizom rezultata merenja zaključeno je da kvalitet napona, tokom sedmodnevnog perioda posmatranja, u potpunosti zadovoljava zahteve propisane ovim standardom.

6. LITERATURA

- [1] Neobnovljivi izvori energije
https://hr.wikipedia.org/wiki/Neobnovljivi_izvori_energij
- [2] Obnovljivi izvori energije
www.industrija.rs/vesti/clanak/oie-energija-buducnosti
- [3] Solarna energija
http://www.zelenaenergija.pks.rs/ZelenaEnergija.aspx?id=48&p=3&#_ftn1
- [4] Ispitni materijal za pripremu ispita „Obnovljivi izvori energije“; prof. dr Vladimir Katić, Fakultet tehničkih nauka Novi Sad
- [5] „Mjerenje kvaliteta električne energije“ MH „Elektroprivreda Republike Srpske“ a.d. Trebinje
- [6] Standard EN 50160 „Karakteristike napona u javnim elektrodistributivnim mrežama“
- [7] Distribuirani energetske resursi
https://en.wikipedia.org/wiki/Distributed_generation
- [8] „Pravilnik o uslovima za priključenje elektrana na elektrodistributivnu mrežu Republike Srpske“
- [9] MH „Elektroprivreda Republike Srpske“ Matično preduzeće a.d. Trebinje
- [10] MI (Metrel) 2292
<http://blutel.net/products-page/loggers-snimaci-parametara-elektricnih-mreza/metrel-mi-2292-power-plus/>

Kratka biografija:

Oliver Krezović je rođen u Pljevljima 1987. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektroenergetika – Distribuirani energetske resursi i pametne mreže odbranio je 2023.god.

Dejan Jerkan je vanredni profesor na Fakultetu tehničkih nauka u Novom Sadu, na Katedri za Energetsku elektroniku i pretvarače. Oblast interesovanja su mu modelovanje i dijagnostika električnih mašina, kao i metoda konačnih elemenata.

**ПРЕПОЗНАВАЊЕ ЉУДСКИХ АКТИВНОСТИ УПОТРЕБОМ МОДЕЛА
МАШИНСКОГ УЧЕЊА****HUMAN ACTIVITY RECOGNITION USING MACHINE LEARNING ALGORITHMS**

Теодора Недић, Јелена Сливка, *Факултет техничких наука, Нови Сад*

Област – РАЧУНАРСТВО И АУТОМАТИКА

Кратак садржај – Проблем препознавања људских активности се користи за дефинисање образаца понашања човека. За препознавање се користе сигнали уграђених сензора паметних телефона или носивих уређаја. Сирове податке потребно је претпроцесирати применом филтера за шум и затим узорковати помоћу фиксираних клизајућих прозора. Из сваког прозора, вектор обележја се добија рачунањем статистичких променљивих из домена времена и фреквенције. У овом раду су, над тако обрађеним подацима, обучени статистички модели машинског учења и конволуционе неуронске мреже. Најбоље резултате постигли су модели насумичне шуме (F -мера 0.93) и конволуциона неуронска мрежа (F -мера 0.94). На основу добијених резултата и радова других аутора, сматра се да ефикасност различитих класификатора у највећој мери зависи од самог скупа података, због чега се у литератури може наћи велики број различитих техника за решавање овог проблема које дају сличне резултате.

Кључне речи: препознавање људске активности, конволуционе неуронске мреже, метод насумичне шуме, машинско учење.

Abstract – The problem of recognizing human activities is used to define patterns of human behavior. Signals from built-in sensors of smartphones or wearable devices are used for recognition. The raw data must be preprocessed using a noise filter and sampled using fixed sliding windows. A feature vector is obtained from each window by computing statistical variables from the time and frequency domain. In this paper, traditional machine learning and convolutional neural network models are trained on the preprocessed data. The best results were achieved using the Random Forest classifier (0.93 F -measure) and the convolutional neural network (0.94 F -measure). Based on the obtained results and the surveyed literature, the efficiency of different classifiers largely depends on the used data set, which is why many different techniques yield similar results.

Keywords: activity recognition, CNN, Random Forest, machine learning

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је била др Јелена Сливка, ванр. проф.

1. УВОД

У овом раду описан је проблем препознавања активности човека (енг. *Human Activity Recognition*, HAR) на основу података прикупљених помоћу сензора паметних мобилних телефона. Доступност сензора у мобилним платформама је омогућила развој низа практичних апликација за неколико научних области као што су здравство, *IoT* и паметни градови, безбедност и превоз. Сензори носивих уређаја (акцелерометар, жирокоп, оријентациони сензори) прикупљају сирове сигнале временских серија.

У овом раду је описано решење за проблем препознавања активности човека. Циљ је детектовати шест активности: лежање, седење, стајање, ходање, ходање уз степенице и ходање низ степенице. Решење је засновано на спајању три јавно доступна скупа података која се типично користе за дати проблем. Над прикупљеним подацима је обучено више модела машинског учења у циљу одређивања најефикаснијег модела. Коначно, прикупљеним подацима је придодат скуп података креиран од стране аутора овог рада.

Прегледом постојећих решења, примећује се употреба и неуронских мрежа и традиционалних метода машинског учења. Најбољи резултати пронађени у литератури су следећи: конволуциона неуронска мрежа (*Convolutional Neural Network*, CNN) је постигла скоро савршене резултате за покретне активности (99.66%), највише на класама као што су ходање уз степенице и низ степенице. Међутим, најлошији резултат исте мреже је постигнут за активност лежања (87.71%). Са друге стране, употребом метода потпорног вектора (*Support Vector Machine*, SVM) постигнут је бољи резултат за стационарне активности (94.91%) [1].

Новина овог рада у односу на претходна решења представља скуп података, који је комбинација више најчешће коришћених скупова. Над таквим скупом, најбољи резултат су постигли насумичне шуме (0.93 F -мера), и конволуциона неуронска мрежа (0.94 F -мера). Остали модели су имали приближне перформансе (> 0.9 F -меру).

У поглављу 2 биће дат преглед постојећих решења. Детаљи решења и имплементације система су описани у поглављу 3, а поглавље 4 садржи анализу добијених резултата.

Последње, 5. поглавље, представља закључак рада.

2. ПРЕТХОДНА РЕШЕЊА

Претраживањем литературе везане за област HAR, дошли смо до неколико корисних радова описаних у наставку.

У раду [2] предложена је конволуциона неуронска мрежа за класификацију, након експеримената са пет модела: CNN, LSTM, BLSTM, MLP и SVM. Експерименти су извршени над два јавно доступна скупа података: UCI HAR [3] и PAMAP2 [4]. Циљна варијабла је активност, која је ограничена на 6 могућих вредности (*walking, walkingupstairs, walkingdownstairs, sitting, standing, laying*). Подаци су подељени на тренинг и тест скуп у односу 7:3. Додатно, подаци су прочитани коришћењем методе клизећег прозора са преклапањем 50%. Због другачије фреквенције узорковања за два скупа података, коришћена је *downsampling* метода да би подаци били конзистентни и лаки за поређење. Сви модели су тренирани са *learning rate*-ом 0.001 у 200 итерација. За метрике евалуације коришћени су: а метрике евалуације коришћени су: *accuracy, precision, recall, F1* мера и *confusion* матрице. CNN модел је постигао најбољи резултат са 93.2% тачности на UCI скупу података и 91% на PAMAP2 скупу.

У раду [5] коришћене су традиционалне методе машинског учења: стабло одлуке, метод насумичне шуме, ET, метод *K* најближих суседа, логистичка регресија, SVM. Предложена метода је логистичка регресија са постигнутом тачношћу од 98.4%. Скуп података је јавно доступни скуп UCI HAR као и у раду [2]. Из рада [5] закључено је да су постигнути бољи резултати у односу на CNN, па самим тим не смемо у старту одбацити статистичке моделе машинског учења.

У раду [1] описани су експерименти са дубоким неуронским мрежама. Показано је да је мала величина филтера сажимања ($1 \times 2 - 1 \times 3$) погоднија. CNN је такође постигао скоро савршену класификацију у покретним активностима. Закључено је да су CNN бољи од *state-of-the-art* техника истраживања података у HAR-у за скуп података прикупљен од 30 волонтера (UCI HAR), постижући тачност од 94,79% на тест скупу са сировим подацима сензора, и 95,75% са додатним информацијама временске брзе Фуријеове трансформације. Подаци су стандардизовани и подељени у односу 7:3. У раду је детаљно описан експериментални приступ. Величина инпут вектора је 128, величина филтера је 1×3 , величина *pooling* филтера је 1×3 , активациона функција је *ReLU*, *learning rate* је 0.01, *dropout* вероватноћа 0.8, *minibatch* величина је 128, максималан број епоха 5000. За подешавање хипер-параметара користили су *greedy-wise* тренирање, где су подешавали број слојева, број *feature* мапа, величине конволуционих филтера и *pooling* филтера. CNN је постигла скоро савршене резултате за покретне активности (99.66%), највише на класама као што су *walking upstairs* и *walking downstairs*. Најлошији резултат је постигнут за *laying* (87.71%). Са друге стране, употребом SVM је постигнут бољи резултат за стационарне активности (94.91%), али, као већина других класификатора, није добро разликовао међусобно сличне активности, као што су ход уз и низ

ступенице. Из овог рада је преузет начин подешавања хипер-параметара ради постизања бољих резултата.

3. МЕТОД

У наредним поглављу су описани скупови података коришћени у раду, начин на који је извршена експлоративна анализа података и архитектура испробаних модела за предикцију.

3.1. Скупови података

У пројекту су спојени следећи скупови података:

UCI HAR [3] обухвата мерење шест различитих дневних активности које је извршило 30 учесника користећи *Samsung Galaxy S II* телефон. Скуп садржи 560 вредности које представљају различите карактеристике активности, укључујући временске и фреквенцијске променљиве. Садржи укупно 10 299 инстанци.

UCI HAR AAL [6] скуп података представља додатак претходном, са циљем побољшања тачности алгоритама за препознавање понашања. Подаци су прикупљени од 30 учесника који су носили уређај за мерење око струка, а њихова старосна група се кретала од 22 до 79 година.

PAMAP2 [4] скуп података обухвата 3 850 505 инстанци различитих физичких активности, укључујући и шест основних активности које се налазе и у првом скупу података. Подаци су прикупљени од стране 9 учесника који су носили три уређаја за мерење на различитим деловима тела. Садржи 54 карактеристике, од којих су најзначајније оне прикупљене помоћу акселерометра и жироскопа.

Ручно креиран скуп података је настао мерењем сигнала апликацијом *Physics Toolbox Suite*[7] док су учесници изводили шест различитих активности, свака у трајању од пет минута. Телефон је био постављен у пределу струка, а након мерења, апликација је генерисала CSV датотеке са прикупљеним подацима. На основу тих датотека, креиран је скуп података који је прошао кроз исти процес претпроцесирања као и претходни скуп података (PAMAP2).

3.2. Обрада података

Процес претпроцесирања података захтевао је низ корака којим се из сирових вредности сигнала који се састоји из три осе, добија 560 обележја у домену времена и фреквенције. У зависности од скупа података, процес је измењен како би се прилагодио подацима.

Прво су избачене непотребне колоне, тј. оне колоне које се не постоје у прва два скупа (временско обележје, притисак, вредности уређаја који су ношени око зглоба ноге и око руке). Затим су избачени и редови који не садрже неке од шест активности, као и редови са недостајућим вредностима јер их није било много. Након тога смањена је фреквенција са 100 на 50 Hz, избацивањем сваког другог реда. Затим су подаци филтрирани помоћу *median* филтера и *low-pass Butterworth* филтера трећег реда са угаоном фреквенцијом 20 Hz како би се уклонио шум.

Сигнал убрзања је раздвојен на два сигнала, убрзање тела и гравитационо убрзање употребом другог *low-*

pass Butterworth филтера са угаоном фреквенцијом 0.3 Hz. Након тога, сигнал убрзања тела и угаона брзина су коришћени за добијање сигнала трзаја (енг. *Jerk* сигнал).

Магнитуда ових сигнала израчуната је помоћу Еуклидске удаљености. На крају, FFT (енг. *Fast Fourier Transform*) је примењен на неке од добијених сигнала.

Над претходно описаним скуповима података извршена је експлоративна анализа и обрада. Овај процес је врло често круцијалан у поступку предикције, јер је то једини начин да сирови подаци постану употребљиви. Први корак представља анализа и обрада сваког од скупова података понаособ.

3.3. Архитектура решења

У наставку су описане архитектуре коришћених метода и мрежа које су примењиване над спојеним скупом података.

Прва испробана метода је метода потпорних вектора (енгл. SVM - *Support Vector Machine*) која показује добре резултате на високо-димензионалним скуповима података, што одговара проблему који решавамо. Испробане су две верзије SVM алгорита, *Linear* и *Kernel*.

У случају употребе *Linear* SVM алгорита, као класификатор коришћена је *LinearSVC* метода из *sklearn* библиотеке са следећим параметрима:

- *C* у опсегу од 1-12 са кораком два,
- *tolerancy* (критеријум заустављања) са вредношћу 0.00005.

Изабрана вредност параметра *C* показала је оптималне перформансе на нашем скупу података. Подешавање хипер-параметара одрађено је применом *RandomizedSearchCV* методе у шест итерација.

Визуелизацијом обележја у 2D простору, утврђено је да подаци нису линеарно сепарабилни, те је из тог разлога испробана и *kernel* SVM метода. Како немамо предзнање о подацима, коришћен је RBF кернел, са следећим параметрима:

- *gamma* = 0.0078125
- *C* = 8

На нашем примеру, најбољи резултат дало је мало гама са малом толеранцијом на грешке.

Следећи експеримент је био са моделом насумичне шуме. Коришћена је *RandomForestClassifier* класа *sklearn* библиотеке са следећим параметрима:

- *n_estimators* = 201
- *max_depth* = 30,

где *n_estimators* представља број стабала.

Конволуциона мрежа је препоручена за временски или просторно повезане податке, па је испробана у последњем експерименту. CNN 1D мрежа се састоји из 2 конволуциона и 2 *fully connected* слоја. Између првог и другог је употребљен један *max pooling* слој. Након сваког слоја је примењена *batch* нормализација и *dropout* са параметром 0.5. Активациона функција је *ReLU*, осим у последњем слоју где је коришћен *Softmax*. Архитектура мреже приказана је на слици 1.

Model: "sequential_5"

Layer (type)	Output Shape	Param #
conv1d_12 (Conv1D)	(None, 138, 96)	1152
batch_normalization_10 (Batch Normalization)	(None, 138, 96)	384
dropout_14 (Dropout)	(None, 138, 96)	0
max_pooling1d_5 (MaxPooling1D)	(None, 68, 96)	0
conv1d_13 (Conv1D)	(None, 68, 32)	15392
batch_normalization_11 (Batch Normalization)	(None, 68, 32)	128
dropout_15 (Dropout)	(None, 68, 32)	0
flatten_4 (Flatten)	(None, 2176)	0
dense_8 (Dense)	(None, 4096)	8916992
dropout_16 (Dropout)	(None, 4096)	0
dense_9 (Dense)	(None, 6)	24582
Total params: 8,958,630		
Trainable params: 8,958,374		
Non-trainable params: 256		

Слика 1. Архитектура CNN

4. РЕЗУЛТАТИ И ДИСКУСИЈА

Над скуповима података предвиђана је људска активност, сврстана у једну од следећих категорија: лежање, седење, стајање, ходање, ходање уз степенице и ходање низ степенице.

Експеримент је постављен поделом на тренинг, валидациони и тест скуп у размери 70:20:10. Подаци су подељени насумичним стратификованим узорковањем. За утврђивање тачности модела коришћене метрике су: *F*-мера (*F-measure*), тачност (*accuracy*), прецизност (*precision*) и одзив (*recall*).

У табели 1 су приказани сумирани резултати експеримената над тест подацима. Као најбољи од модела машинског учења показао се *Random Forest* алгорита. Разлог његове примене било је превазилажење недостатака *Decision Tree* алгорита. Оптимизацијом *Random Forest* алгорита, добијена је тачност 93.8%, што је чини овај модел скоро подједнако добрим као и CNN.

	макро F-мера	тачност
<i>Decision Tree</i>	0.8690	0.8692
<i>Linear</i> SVM	0.9126	0.9047
<i>Kernel</i> SVM	0.9264	0.9161
<i>Random Forest</i>	0.9368	0.9382
CNN	0.9407	0.9305

Табела 1. Резултати експеримената

Конволуциона неуронска мрежа постигла је најбоље резултате, што је очекиван резултат на основу литературе прегледане у поглављу 2. Резултати постигнути на тест скупу показују приближну тачност за све три коришћене метрике, од 94%. У табели 2 се налази преглед резултата по класама.

	прецизност	одзив	F-мера
walking	0.9490	0.9411	0.9450
walking_downstairs	0.9384	0.9366	0.9375
walking_upstairs	0.9432	0.9351	0.9391
laying	0.9465	0.9499	0.9482
sitting	0.9245	0.9164	0.9204
standing	0.9153	0.9072	0.9112

Табела 2. Резултати по класама за CNN модел

Можемо приметити да су модели најлошије препознавали активност стајања. Претпоставља се да је разлог томе сличност ове класе са класом седење.

Резултати по класама су у великој мери међусобно слични, док у постојећим решењима постоје веће варијације између класа. Претпоставка је да ово запажање може бити последица употребе више јавно доступних скупова података који су мерени различитим уређајима и у условима који нису потпуно исти. Такође, применом истог модела само над ручно прављеним скупом података, добијена је висока тачност од 99%, што даје могућност за будуће проширење скупа, укључивањем више учесника у процес мерења и повећањем броја активности.

Претпоставља се да би се ограничења могла превазићи и обучавањем комплекснијих модела неуронских мрежа, које захтевају много више ресурса у контексту хардвера.

5. ЗАКЉУЧАК

У раду је описан решавањем проблем препознавања људских активности на основу података измерених помоћу сензора акцелерометра и жирокопа.

Мотивацију за решавање овог проблема налазимо у могућности практичне употребе. Решење описано у овом раду заснива се на спајању три јавно доступна скупа података који се користе за дати проблем, где је за спајање било потребно претпроцесирати један од скупова како бисмо добили одговарајућа обележја. Затим следи обучавање неколико модела машинског учења како бисмо одредили оптималан модел и урадили поређење резултата. Након тога је ручно направљен скуп података, по узору на поступак за јавно доступне скупове. Најбољи добијени резултат је постигнут коришћењем CNN модела, где F-мера мерена на тест скупу износила 0.94. Одмах за њим је и модел насумичне шуме са F-мером 0.93. Додавањем ручно прављеног скупа података успешно су побољшани укупни резултати за 0.01 F-мере.

Квантитативно поређење HAR приступа омета чињеница да сваки систем ради са различитим скупом података [8]. Предност решења описаног у раду у односу на постојећа је процес спајања више извора података, који су мерени различитим уређајима и у условима који нису потпуно идентични. Претпоставља се да су због тога резултати нешто нижи у односу на најбоље резултате пронађене у литератури, али је модел истрениран тако да је робуснији и може се користити за препознавање активности из сва три коришћена скупа података. За сада је решаван проблем препознавања шест активности, али, узимајући у обзир добре постигнуте резултате, може се у даљим истраживањима додати и низ других људских активности.

6. ЛИТЕРАТУРА

- [1] Ronao, C. A., & Cho, S. B. (2016). Human activity recognition with smartphone sensors using deep learning neural networks. Expert systems with applications, 59, 235-244.
- [2] Wan, S., Qi, L., Xu, X., Tong, C., & Gu, Z. (2020). Deep learning models for real-time human activity recognition with smartphones. Mobile Networks and Applications, 25(2), 743-755.
- [3] UCI HAR dataset [Reyes-Ortiz, Jorge, Anguita, Davide, Ghio, Alessandro, Oneto, Luca & Parra, Xavier. (2012). Human Activity Recognition Using Smartphones. UCI Machine Learning Repository.]
- [4] PAMAP2 dataset [A. Reiss and D. Stricker. Introducing a New Benchmarked Dataset for Activity Monitoring. The 16th IEEE International Symposium on Wearable Computers (ISWC), 2012.]
- [5] Minarno, A. E., Kusuma, W. A., Wibowo, H., Akbi, D. R., & Jawas, N. (2020, June). Single triaxial accelerometer-gyroscope classification for human activity recognition. In 2020 8th International Conference on Information and Communication Technology (ICoICT) (pp. 1-5). IEEE.
- [6] Smartphone Dataset for Human Activity Recognition (HAR) in Ambient Assisted Living (AAL) Data Set
- [7] Vieyra, Chrystian, Vieyra, Rebecca. Physics Toolbox Suit
- [8] Lara, O. D., & Labrador, M. A. (2013). A Survey on Human Activity Recognition using Wearable Sensors. IEEE Communications Surveys & Tutorials, 15(3), 1192-1209. doi:10.1109/SURV.2012.110112.00192

Кратка биографија:



Теодора Недић рођена је 1998. године у Врбасу, Србија. Основне академске студије завршила је 2021. године на Факултету техничких наука, на ком брани и мастер рад 2023. године из области Електротехнике и рачунарства – Софтверско инжењерство и информационе технологије. контакт: nedicteodora1@gmail.com

RAZVOJ ČET BOTA SA DUGOROČNIM PAMĆENJEM I RAZUMEVANJEM KONTEKSTA**DEVELOPMENT OF A CHAT BOT WITH LONG-TERM MEMORY AND CONTEXT UNDERSTANDING**

Dejan Dokić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIČKO I RAČUNARSKO INŽENJERSTVO

Kratak sadržaj – U ovom radu će biti predstavljen razvoj čet bota koji je u stanju da dugoročno pamti i razume kontekst započete prepiske. Rad uključuje opis korišćenih tehnika i načina na koji su povezani zarad dobijanja sistema koji simulira ljudsku interakciju.

Ključne reči: čet botovi, vektorske baze podataka, dugoročna memorija, jezički modeli

Abstract – This work will present the development of a chatbot capable of long-term memory and understanding the context of initiated correspondence. The work includes a description of the techniques used and the way they are connected to achieve a system that simulates human interaction.

Keywords: chatbots, vector databases, long-term memory, language models

1. UVOD

Čet botovi predstavljaju poseban tip digitalnih sistema osmišljenih sa ciljem da simuliraju ljudsku interakciju putem tekstualnog ili glasovnog kanala. Funkcionišu kao agenti pružajući konstantnu podršku u bilo koje doba dana, i na taj način doprinose smanjenju intenziteta potrebne radne snage kao i operativnih troškova. U stanju su da istovremeno obrade veliki broj upita, komuniciraju sa eksternim servisima ako se javi potreba za time i u svakom momentu pruže doslednu informaciju.

Prvobitno, čet botovi su bili ograničeni time što su imali predefinisani skup odgovora i na osnovu korisničkog unosa, najrelevantniji odgovor iz skupa svih odgovora bi bio vraćen korisniku. Vremenom se javila potreba za složenijim i preciznijim odgovorima, što je zahtevalo primenu veštačke inteligencije (engl. *Artificial Intelligence*, skraćeno *AI*), obrade prirodnog jezika (engl. *Natural Language Processing*, skraćeno *NLP*) i algoritama mašinskog učenja (engl. *Machine Learning*, skraćeno *ML*). Na taj način je postignuta personalizovana komunikacija u realnom vremenu sa mogućnošću učenja iz same interakcije i neprestanog poboljšanja kvaliteta samog odgovora.

Iako je samo dobijanje doslednog odgovora isključivo na osnovu pitanja postavljenog od strane korisnika veliki

uspeh, još precizniji i kvalitetniji odgovor bi bio moguć ako bi čet bot bio u mogućnosti da u proces kreiranja odgovora uzme u obzir što je moguće veći nivo razumevanja konteksta, baziranih na osnovu prethodnih poruka. Na ovaj način, korisniku bi bilo omogućeno da u više različitih iteracija komunikacije sa čet botom zadrži kontekst prethodnih dopisivanja, što ujedno prouzrokuje tačnijim i približnijim odgovorom samom korisniku. Kreiranje takvog čet bota koji je u mogućnosti da dugoročno pamti i razume kontekst dosadašnje prepiske predstavlja veoma važan korak i ujedno najveći izazov u daljem razvoju ove oblasti.

2. KORIŠĆENE TEHNOLOGIJE

Zarad razumevanja načina funkcionisanja samog čet bota, neophodno je pre svega objasniti pojmove: *GPT* modela, *Pinecone* vektorske baze podataka, *LangChain* integracionog okvira i *Zep* dugoročne memorije. Ovo poglavlje ima za cilj da sa teorijske strane objasni prethodno navedene koncepte sa naglaskom na komponente svake tehnologije koje će se koristiti u procesu implementacije čet bota.

2.1. Generativni Unapred Obučeni Transformator (GPT)

Generativni unapred obučeni transformator (*GPT*) predstavlja jezički model¹ (engl. *language model*, skraćeno *LM*) razvijen od strane OpenAI-a² koji je sposoban da proizvede odgovor u vidu teksta koji se gotovo ne razlikuje u odnosu na prirodni ljudski jezik. Koncepti koji stoje iza generativnog unapred obučenog transformatora se usavršavaju kroz proces iz dva koraka. Prvi korak je generativna nenadzirana predobuka koristeći neobeležene podatke, dok je drugi korak diskriminativno, nadgledano fino podešavanje (engl. *fine-tuning*) zarad poboljšanja performansi. Tokom prve faze, model uči prirodno, simulirajući kako bi i osoba učila u novom okruženju, dok u drugoj fazi fino podešavanje uključuje više usmereno i struktuirano usavršavanje od strane kreatora [2].

Ključno postignuće u procesu razvoja jezičkih modela predstavlja *InstructGPT*. Ovaj radni okvir omogućava

¹ Jezički model je komponenta za obradu prirodnog jezika i odnose se na sisteme koji su obučeni nad velikim korpusom podataka [1].

² OpenAI je istraživačka laboratorija usmerena u razvoj AI tehnologija i koja je objavila niz proizvoda za mašinsko učenje kao što su: *ChatGPT* i *DALL-E* [2].

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dušan Gajić, vanr. prof.

fino podešavanje instrukcija prethodno obučenog jezičkog modela zasnovano na učenju pojačanjem iz ljudskih povratnih informacija (engl. *reinforcement learning from human feedback*, skraćeno *RLHF*). Korišćenje ljudskih povratnih informacija doprinosi tome da se jezički model uspešno koristi za rešavanje širokog spektra zadataka u oblasti obrade prirodnog jezika, čineći ga veoma raznovrsnim i fleksibilnim. [3].

2.2. Pinecone vektorska baza podataka

Kako bi čet botovi pružili koristan odgovor na zadati korisnikov upit potrebno je razumeti kontekst samog upita, njegovu semantičku vrednost i na osnovu toga generisati najprikladniji odgovor. Kako bi željeni efekat bio ostvariv, neophodno je koristiti vektorske ugradnje (engl. *vector embeddings*), odnosno tip predstavljanja podataka koji omogućava da sami podaci u sebi sadrže semantičke vrednosti koje su kasnije ključne za rad AI modela u svrhu razumevanja konteksta i održavanja dugoročne memorije [4].

Prilikom vektorizacije većeg broja podataka, primetno je da se u okviru jednog vektorskog prostora distanca između vektora razlikuje, odnosno da su neki bliži jedan drugom, dok su drugi udaljeniji [5]. Ovaj efekat je izuzetno bitan, jer grupisani vektori, odnosno vektori bliži jedni drugima imaju sličnije semantičko značenje u odnosu na one druge koji su dosta udaljeniji od njih. Vektori sami po sebi predstavljaju idealnu strukturu podataka za algoritme mašinskog učenja jer su procesorske jedinice i grafičke kartice optimizovane za obavljanje matematičkih operacija za njihovu obradu. Proces transformacije podataka u vektor je zahtevna operacija, jer je neophodno zadržati prvobitno značenje podataka. Da bismo očuvali značenje samih podataka neophodno je razumeti proces vektorizacije takav da odnosi između vektora imaju smisla.

Za postizanje ovako nečega koristi se model ugradnje (engl. *embedding model*) i to propuštanjem velike količine označenih podataka kroz neuronsku mrežu. Obučavanje neuronske mreže se vrši metodom nadgledanog učenja (engl. *supervised learning*), davajući mreži veliki skup trening podataka sastavljenih od parova ulaza (engl. *input*) i označenih izlaza (engl. *labeled output*). Svakom iteracijom obučavanja, neuronska mreža modifikuje aktivacije u svakom mrežnom sloju i to sve sa ciljem da se na kraju obučavanja izlazna oznaka može predvideti na osnovu datog ulaza, čak i za ulaz koji nije bio deo testnog skupa.

Model ugradnje je zapravo ovako obučena neuronska mreža ali bez poslednjeg sloja, kako bi se umesto označenog izlaza, dobila vektorska ugradnja [5].

Dobijene vektorske ugradnje je potrebno prikladno čuvati. Za to se koristi vektorska baza podataka (engl. *vector database*), u konkretnom slučaju korišćena je *Pinecone* vektorska baza podataka, radi optimizovanog skladištenja vektorskih ugradnji i kasnijim upitima nad istim. Ovakav tip baza podataka je specijalno dizajniran za rukovanje ovom vrstom podataka nudeći visoke performanse, skalabilnost i fleksibilnost pri radu sa vektorskim podacima. Obezbeđena je pretraga na semantičkom nivou, kao i dugoročno čuvanje podataka [4].

2.3. LangChain

LangChain služi kao radni okvir dizajniran za kreiranje aplikacija zasnovanih na velikim jezičkim modelima. Njegov primarni cilj je da omogući pogodan način integracije različitih izvora podataka i da olakša integraciju sa drugim aplikacijama. U procesu razvoja aplikacija kao što su čet botovi, ključno je da sistem poseduje svest o kontekstu, što znači da treba da poveže jezički model sa različitim izvorima konteksta [6]. Inkorporacija samog konteksta je od presudnog značaja za modele da shvate i efikasno integrišu informacije. Štaviše, radni okvir se oslanja i na jezički model zbog samog rasuđivanja, uzimajući u obzir kako generisati odgovor na osnovu datog konteksta i odrediti akcije koje je potrebno izvršiti.

U aplikacijama zasnovanim na jezičkom modelu, svakako najvažnija celina jeste sam model. *LangChain* pruža interfejs i integraciju za dva tipa modela: velike jezičke modele (engl. *Large Language Model*, skraćeno *LLM*) i konverzijske modele (engl. *Chat Models*). Ova dva tipa modela se suptilno, ali opet značajno razlikuju. Veliki jezički model se u okviru *LangChain*-a odnosi na model generisanja teksta, prihvatajući tekst na ulazu i isto tako izbaciti generisani tekst na izlazu. *GPT-3* model razvijen od strane OpenAI-a je implementiran kao veliki jezički model. Konverzijski model sa druge strane podržava veliki jezički model, ali je dodatno prilagođen za razgovore. Ovo podrazumeva da na samom ulazu ne primaju čist tekst, već listu poruka, posebno označenih ulogom kojoj pripadaju (obično se koriste: „*System*“, „*AI*“ i „*Human*“), dok je izlaz iz ovakvog tipa modela poruka obeležena oznakom „*AI*“. *GPT-4* model je jedna od implementacija koja koristi ovaj tip [7].

LangChain je skup gradivnih elemenata, od kojih je najvažniji svakako sam lanac (engl. *chain*). Lanac obično kombinuje veliki jezički model zajedno sa *prompt*-om, čineći zasebnu celinu, ali koja se dalje može kombinovati sa drugim gradivnim elementima i na taj način proširiti kako bi se postigao željeni niz operacija nad korisničkim upitom. Najjednostavniji tip lanca predstavlja onaj koji ima jedan ulaz i jedan izlaz. Više lanaca mogu biti pokrenuti jedan za drugim, gde bi izlaz jednog lanca predstavljao ulaz sledećeg. Više različitih lanaca mogu biti spojeni primenom *Simple Sequential Chain*-a. Postoje i drugi tipovi lanaca. Nešto složeniji u odnosu na prethodni je *Sequential Chain*, koji može da ima više ulaza, ali samo jedan izlaz. Poprilično uobičajen scenario je korišćenje više lanaca postavljenih tako da je usmeravanje zavisno od toga šta se nalazi na samom ulazu. Na primer, ako imamo nekoliko usko specijalizovanih lanaca za određen tip podataka, možemo ispred njih imati jedan *Router Chain*, koji će odlučiti u kojem pravcu će informacija dalje ići, odnosno kojem lancu će predati tok [8].

Jezički model je sam po sebi bez stanja (engl. *stateless*), što znači da nije svestan bilo kakve interakcije (razgovora) koji se vodio do sada. Svaki poziv ka jezičkom modelu je nezavisan od predašnjih. Kako bi se ostvario efekat pamćenja, neophodno je uvrstiti komponentu memorije (koja čuva prethodnu prepisku) u jezički model prilikom narednog upita. *LangChain* nudi

niz različitih načina upravljanja memorijom, a najkorišćeniji su [9]:

a) **ConversationBufferMemory.**

ConversationBufferMemory funkcioniše na taj način što čuva čitavu prepisku i prosleđuje je kao kontekst u model prilikom svakog narednog upita. Ovaj pristup može biti koristan u određenim scenarijima, ali se smatra kao izuzetno skup, iz razloga što kada se konverzacija oduži, velike količine informacija se prosleđuju pri svakom sledećem upitu.

b) **ConversationBufferWindowMemory.**

ConversationBufferWindowMemory predstavlja tip memorije, ali specifičan po tome što je moguće definisati koliko maksimalno poslednjih interakcija³ je moguće ubaciti u kontekst prilikom narednog korisničkog upita. Ovim se postiže to da je memorija limitirana, odnosno da je rast ograničen.

c) **ConversationTokenBufferMemory.**

ConversationTokenBufferMemory je dosta sličan prethodnom tipu memorije. Kao što i sam naziv govori, razlika je u tome što se umesto broja interakcija, definiše limit na nivou broja tokena⁴. Pamćenjem N poslednjih tokena, memorija je takođe limitirana.

d) **ConversationSummaryMemory.**

ConversationSummaryMemory predstavlja specifičan tip memorije. Za razliku od prethodnih tipova, u ovom pristupu se koristi jezički model radi sumarizacije prethodne prepiske. Tako sumarizovan kontekst se kasnije uzima u obzir pri generisanju novog odgovora.

Za razliku od lanaca, kod kojih je redosled akcija unapred određen, agenti predstavljaju fleksibilnije rešenje u pogledu da je odlučivanje koje akcije će se izvršiti i kojim redosledom prepušteno samom jezičkom modelu. Agenti se još zovu i lanci koji su odgovorni za određivanje narednih koraka u procesu izvršavanja. Pored samog jezičkog modela i korisničkog unosa, agentima je neophodna informacija o nizu alata (engl. Tools⁵) kojima raspolaze u prethodno izvršenim koracima. Kako bi agenti uspešno koristili ponuđene alate, neophodno je dati opis u kojim situacijama bi određeni alat bio od koristi, što će kasnije služiti jezičkom modelu prilikom odabira narednog koraka [11].

2.4. Zep dugoročna memorija

Zep funkcioniše kao platforma otvorenog koda (engl. *open source*) dizajnirana tako da pruži efikasan rad sa aplikacijama baziranih na jezičkom modelu. Ovu platformu odlikuje sposobnost asinhrona sumarizacije,

³ Interakcija predstavlja spoj korisničkog unosa i dobijenog odgovora iz jezičkog modela

⁴ Token je predstavljen kao deo reči, njegova dužina je oko 4 karaktera engleskog jezika [10].

⁵ Tool predstavlja funkciju koja se poziva od strane *LangChain* agenata zadužena za rešavanje specifičnih operacija [11].

ugrađivanja, kao i obogaćivanja interakcija u okviru prepiske, i to na način da ne narušavaju korisničko iskustvo prilikom korišćenja samog čet bota. *Zep* ima svoju bazu podataka, gde se vrši čuvanje podataka, obezbeđujući skalabilnost potrebnu u bilo kom trenutku [12].

3. IMPLEMENTACIJA „AUTO CALL CENTER“ ČET BOTA

„Auto Call Center“ čet bot predstavlja sistem zadužen da korisnicima pruži pomoć prilikom kupovine novog automobila. Čet bot raspolaze sa informacijama o tome koja su vozila dostupna u salonu, imajući dodatne informacije o svakom. Pored samog informisanja o stanju ponude, čet bot je u mogućnosti da pruži bilo kakav uopšteni vid informacije vezan za automobil. Uspešno savetuje prilikom nedoumice o konkretnim markama automobila i njihovim modelima kao što su potrošnja, istorija itd. Takođe sposoban je da reši naprednije matematičke računice, koje su potrebne prilikom izračunavanja poreza, prilikom povrata nakon uspešne kupovine. I pored svega prethodno opisanog, čet bot ima mogućnost pamćenja same konverzacije sa korisnikom, što pruža još preciznije odgovore, približnije samom korisniku.

Serverska strana pisana u Python [13] programskom jeziku, dok je za klijentsku stranu korišćen Telegram [14]. U nastavku sledi opis komponenti na serverskoj strani koje se koriste u procesu rezonovanja prilikom generisanja odgovora:

Prva bitna komponenta jeste *Pinecone* vektorska baza podataka. Predstavljena u obliku alata korišćenog od strane *LangChain* agenta za dobijanje informacija o ponudi samih vozila kojima salon raspolaze. U okviru ove baze podataka, unete su vektorske ugradnje nastale na osnovu konkretnih primeraka automobila. Svaka vektorska ugradnja predstavlja vektor dimenzije 1536 i kasnije je za pretragu ovih entiteta korišćena kosinusna (engl. *cosine*) razdaljina. Agent pristupa ovom alatu isključivo u situacijama kada je potrebna informacija o raspoloživosti automobila, konkretnoj ponudi i potrebi da odgovori na pitanja vezana za cenu izlistanih modela.

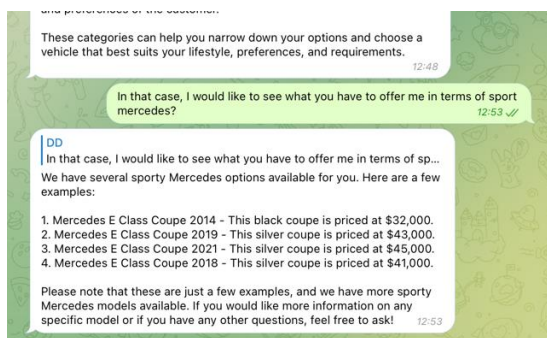
Druga bitna komponenta je *Zep* dugoročna memorija. Kao i *Pinecone* vektorska baza podataka, *Zep* je definisan kao poseban alat korišćen od strane *LangChain* agenta za dobijanje informacija o detaljima iz dosadašnje prepiske. Ova baza sadrži čitavu konverzaciju između korisnika i čet bota. Za svakog korisnika se kreira jedinstvena sesija koja se kasnije koristi za grupisanje svih poruka i sumarizacija za konkretnog korisnika. Najčešća primena ovog alata jeste za pamćenje preferencija korisnika koje se odnose na željeni automobil.

LangChain agent pored prethodno opisanih alata, koristi još dva manja, i to: *MathTool* i *WikiTool*. *MathTool* se koristi za rešavanje matematičkih računica, u ovom slučaju najčešće za izračunavanje poreza u slučajevima kada je moguć povrat istog, dok sa druge strane *WikiTool* služi za informisanje o samim markama i modelima automobila.

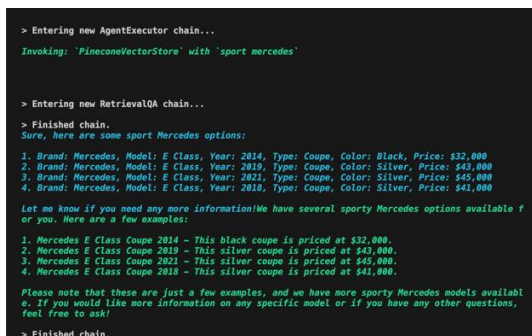
Za kraj je ostao sam *LangChain* agent, koji koristi sve prethodno navedene alate i opise za njihovo korišćenje, kao i jezički model kako bi doneo odluku o tome koji od ponuđenih izvora informisanja je najprikladniji za generisanje odgovora na osnovu korisničkog unosa.

4. REZULTATI RADA

„Auto Call Center“ čet bot predstavlja sistem zadužen da korisnicima pruži pomoć prilikom kupovine novog automobila. Zbog prirode problema koji rešava, demonstracija njegovog rada će biti praćena kroz primer konverzacije između čet bota i potencijalnog kupca. Upravo na Slici 1 je prikazan takav scenario, gde korisnik iznosi informaciju da je zainteresovan za sportska vozila brenda Mercedes i od čet bota traži listu vozila koja ispunjavaju dati kriterijum. Čet bot pruža odgovor u vidu liste, čime ispunjava korisnikov zahtev, a sam odgovor je zasnovan na korišćenju vektorske baze podataka, jer se radi o konkretnim vozilima iz ponude salona, što je upravo i definisano u samom alatu agenta. Rezonovanje od strane agenta je prikazano na Slici 2.



Slika 1. Prikaz konverzacije



Slika 2. Prikaz rezonovanja agenta

5. ZAKLJUČAK

U ovom radu dat je opis čet bot sistema koji simulira ljudsku interakciju putem tekstualnog kanala. Glavna odlika prethodno opisanog čet bota jeste sposobnost dugoročnog pamćenja i razumevanja konteksta čime su poruke koje generiše još približnije samom korisniku, odnosno postignuta je personalizovana komunikacija u realnom vremenu. Detaljno su objašnjeni osnovni tehnološki koncepti poput: vektorske ugradnje i vektorskih baza podataka, *Zep* dugoročne memorije, *LangChain* radnog okvira i svih gradivnih komponenti koje obuhvata, kao i samog *GPT* modela. U nastavku je objašnjen način na koji je implementiran čet bot, kao i iz

kojih komponenti se sastoji, na koji način su te komponente povezane i na koji deo sistema se oslanja u zavisnosti od date situacije. Takođe pružen je uvid u konkretan primer upotrebe čet bota, ispraćen kratkom konverzacijom i putem rezonovanja koji prođe od prihvatanja poruke, do generisanja smislenog odgovora koji će ispuniti potrebe korisnika.

6. LITERATURA

- [1] Brown, Hannah, Katherine Lee, Fatemehsadat Mireshghallah, Reza Shokri, and Florian Tramèr. "What does it mean for a language model to preserve privacy?." In Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency, pp. 2280-229.
- [2] Lund, Brady D., and Ting Wang. "Chatting about ChatGPT: how may AI and GPT impact academia and libraries?." Library Hi Tech News 40, no. 3 (2023): 26-29.
- [3] Liu, Yiheng, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He et al. "Summary of chatgpt/gpt-4 research and perspective towards the future of large language models." arXiv preprint arXiv:2304.01852 (2023).
- [4] <https://www.pinecone.io/learn/vector-database> (pristupljeno u septembru 2023.)
- [5] <https://www.pinecone.io/learn/vector-embeddings-for-developers> (pristupljeno u septembru 2023.)
- [6] https://python.langchain.com/docs/get_started/introduction (pristupljeno u septembru 2023.)
- [7] https://python.langchain.com/docs/modules/model_io/models (pristupljeno u septembru 2023.)
- [8] Topsakal, Oguzhan, and Tahir Cetin Akinci. "Creating Large Language Model Applications Utilizing LangChain: A Primer on Developing LLM Apps Fast." In International Conference on Applied Engineering and Natural Sciences, vol. 1, no. 1, pp. 1050-1056. 2023.
- [9] <https://learn.deeplearning.ai/langchain/lesson/3/memory> (pristupljeno u septembru 2023.)
- [10] <https://help.openai.com/en/articles/4936856-what-are-tokens-and-how-to-count-them> (pristupljeno u septembru 2023.)
- [11] <https://python.langchain.com/docs/modules/agents> (pristupljeno u septembru 2023.)
- [12] <https://github.com/getzep/zep> (pristupljeno u septembru 2023.)
- [13] <https://www.python.org> (pristupljeno u septembru 2023.)
- [14] <https://telegram.org> (pristupljeno u septembru 2023.)

Kratka biografija:



Dejan Dokić rođen je u Novom Sadu 1997. god. Diplomski rad na Fakultetu tehničkih nauka u Novom Sadu, smer softversko inženjerstvo i informacione tehnologije odbranio je 2020.god.

kontakt: dejan.dokic.ns@gmail.com

**PRETPLATNIČKI MODELI U SPOTIFAJ MUZIČKOM STRIMING SERVISU:
ANALIZA DIZAJNA I UTICAJA NA KORISNIČKO ISKUSTVO****SUBSCRIPTION MODELS IN SPOTIFY MUSIC STREAMING SERVICE: DESIGN
ANALYSIS AND USER EXPERIENCE IMPACT**

Milica Poparić, *Fakultet tehničkih nauka, Novi Sad*

**Oblast – SOFTVERSKO INŽENJERSTVO I
INFORMACIONE TEHNOLOGIJE**

Kratak sadržaj – Ovaj rad sadrži analizu Spotifaj platforme u cilju pronalaska glavnih faktora koji utiču na odluku korisnika o pretplati. Kvalitet premium funkcionalnosti analiziran je upotrebom Nilsenovih heuristika. Odrađena je analiza adiktivnosti platforme preko standardnih negativnih dizajn obrazaca kao što su maglovite funkcionalnosti, ekstremno odbrojanje, odlaganje gašenja, živi pesak pažnje i pristrasnost. Korisničko iskustvo testirano je na fenomene zečijih rupa, totalnog otupljivanja i vrteški. Istraživanje obuhvata i analizu psiholoških faktora kao što su brend, identitet i ekspresivnost, socijalizacija i kolekcionarstvo. Rad sadrži i pregled stavova različitih grupa korisnika, baziran na prethodnim eksperimentima iz ove oblast.

Ključne reči: pretplatnički modeli, striming servis, Spotifaj.

Abstract – This paper contains an analysis of the Spotify platform with the aim of identifying the main factors influencing user decisions regarding subscription. The quality of premium functionality has been analyzed using Nielsen heuristics. An analysis of platform addictiveness has been conducted through standard negative design patterns such as feature fog, extreme countdown, switchoff delay, attention quicksand, and bias grind. User experience has been tested for phenomena such as rabbit holes, gorking out, and carousels. The research includes an analysis of psychological factors such as brand, identity, and expressiveness, socialization, and collectivism. The paper also includes an overview of the attitudes of different user groups based on previous experiments in this field.

Keywords: subscription model, streaming servis, Spotify.

1. UVOD

Muzički striming (engl. *streaming*) servisi ili striming platforme predstavljaju najpopularniji oblik distribucije muzike. Njihova popularnost bazirana je na besplatnom i neograničenom pristupu milionskim muzičkim bibliotekama putem Interneta, što oslikava ideju toka [1]. Tako se biznis model striming servisa temelji na postojanju hibridnog modela: korisnici mogu besplatno uživati u korišćenju servisa, koje ima ograničene

funkcionalnosti i reklame (frimium nalog), ili uplatiti pretplatu i otključati brojne druge funkcionalnosti i ukinuti prikaz reklama (premium nalog). Strategija hibridnog modela je da besplatni nivo privuče korisnike i utiče na njihovu odluku da se pretplate za kvalitetnije korišćenje aplikacije. To se postiže uvođenjem probnih perioda, u okviru kojih korisnik može besplatno isprobati sve funkcionalnosti premium naloga pre nego što se odluči za kupovinu istog [2]. Problematika ovog biznis pristupa ogleda se u dobrom balansu između frimium i premium naloga.

U ovom radu urađena je analiza dizajna i funkcionalnosti probnog perioda jednog od vodećih muzičkih servisa: Spotfaja, sa ciljem da se utvrde dobre prakse iz HCI (engl. *Human Computer Interaction*) oblasti koje utiču na razvoj profitabilnog poslovnog modela. Rad odgovara na pitanje šta se krije iza pejvola i koje funkcionalnosti i dizajn prakse dovode do tranzicije korisnika.

Metodologija se oslanja na istraživanje prethodnih radova iz HCI domena koji se bave razvojem muzičkih striming platformi kao i ličnom analizom Spotifaj platforme. Rad obuhvata anлізу kvaliteta premium funkcionalnosti, analizu korisničkog iskustva i analizu psiholoških faktora korisničkog interfejsa.

Struktura rada je sledeća. U narednom poglavlju dat je pregled srodnih istraživanja. Treće poglavlje sadrži pregled metodologije i osnovnih koncepata. Četvrto poglavlje obuhvata rezultate i diskusiju. Peto poglavlje je zaključak rada.

2. SRODNA ISTRAŽIVANJA

Rad [3] istražuje kako se otkrivanje sadržaja na platformama oblikuje putem interfejsa i rezultirajućih korisničkih iskustava, kao i uticaj rezultata na korisnike, kreatora sadržaja i softver. Rezultat ovog rada je predstavljanje analitičkog okvira koji se sastoji od tri ključne dimenzije procesa otkrivanja sadržaja na platformama: 1) dizajn i upravljanje izborom u interfejsima platformi, 2) putevi koje korisnici biraju za pronalaženje sadržaja i efekti tih izbora, i 3) rezultirajuća iskustva koja proizilaze iz ovih elemenata.

Rad [4] ukazuje na postojeće negativne dizajn obrasce muzičkih striming servisa, koji predstavljaju striming paradoks - širina izbora muzike dovodi do sužavanja raznovrsnosti slušanja iste i pojave mejnstrima (engl. *mainstream*). Rad pruža uvid u ključnih šest faktora koji utiču na manipulaciju muzičkim izborom i pojavu

NAPOMENA:

Ovaj rad je proistekao iz master rada čiji mentor je bio dr Dragan Dinu, vanr. prof.

mejnstrima, a to su: jačanje značaja početne stranice, jačanje značaja novog sadržaja, bazirano na događajima, produžavanje toka, naglašavanje konteksta.

Autori u [5] ukazuju na važnost dizajna i korisničkog iskustva pri razvoju softvera muzičkih platformi. Kroz intervju sa koricinima autori istražuju kvalitet striming servisa po Nilsenovim heuristikama. Rezultati pokazuju da verni korisnici prastaju greške, kad se jednom naviknu na servis dovoljno je da pruža dovoljno dobre usluge. Zaključak je takođe da različite ličnosti korisnika preferiraju različite dizajn funkcionalnosti što ukazuje na mogućnost unapređivanja postojećih platformi tako da budu prilagodljive različitim grupama korisnika.

Pored već pomenutih važnih dobrih praksi pri dizajniranju korisničkog interfejsa, rad [6] upozorava na negativne dizajn obrasce u striming servisima, koji loše utiču po psihičko zdravlje korisnika i stvaraju zavisnost. Fokus istraživanja jesu video striming platforme, rezultati ukažu na štetnost sledećih obrazaca: maglovite funkcionalnosti ekstremno odbrojavanje, odlaganje gašenja, živi pesak pažnje i pristrasnost.

Psihološki aspekt muzičkih striming servisa prikazan je u radu [7] koji uvodi dve faze korišćenja servisa: faza primamljivanja i faza zavisnosti (okida pretplatu). Sprovedeno je istraživanje nad 244 studenta iz Amerike od kojih su 73 pretplaćeni korisnici, dok ostatak koristi frimium nalog. Istraživanje pokazuje da se prva faza primamljivanje temelji na ponašanju, socijalnom uticaju i poverenju, dok drugu fazu zavisnosti karakteriše dobar korisnički interfejs i komunikacija kontrole. Rad ukazuje na značaj bihevioralne psihologije kao jednog od alata pri razvoju muzičkih softvera.

3. PREGLED METODOLOGIJE I OSNOVNIH KONCEPATA

Metodologija ovog rada bazirana je na ličnom istraživanju dizajna i funkcionalnosti muzičkih striming platformi kao i prethodnim rešenjima. Analiza servisa podeljena je na tri ključna aspekta: korisnički interfejs primium funkcionalnosti, korisničko iskustvo i psihološki faktori. Dobre prakse u razvoju korisničkog interfejsa poredene su po Nilsenovim heuristikama. Ove heuristike predstavljaju široko prihvaćen set preporuka koji se koristi kao standard za „dobar“ korisnički doživljaj [2].

U nastavku sledi definicija analiziranih Nilsenovih heuristika: povratna informacija-vidljivost statusa sistema, navigacija-kontrola i sloboda korisnika, doslednost-doslednost i standardi, prevencija-prevencija grešaka, efikasnost-fleksibilnost i efikasnost upotrebe, dizajn - estetski i minimalistički dizajn, obnova - pomoć korisnicima da prepoznaju, dijagnostikuju i oporave se od grešaka [5].

Korisničko iskustvo i psihološki aspekti analizirani su i kroz prizmu negativnih dizajn obrazaca koji dovode do adiktivne upotrebe servisa, u te obrasce spadaju: maglovite funkcionalnosti, ekstremno odbrojavanje, odlaganje gašenja, živi pesak pažnje i pristrasnost.

Obrazac maglovite funkcionalnosti odnosi se na dizajn korisničkog interfejsa koji namerno umanjuje korisnikovu svest i kontrolu nad praćenjem vremena koje provodi na

platformi što dovodi do nekontrolisanog i prekovremenog korišćenja servisa i zavisničkog ponašanja [6].

Ekstremno odbrojavanje se odnosi na obrasce korisničkog interfejsa koji uključuju tajmer i automatski izvršavaju određene radnje ako se ne reaguje pre isteka tajmera, stvarajući pritisak na brzinu odluke, dok ujedno smanjuju autonomiju korisnika za donošenjem svesnih odluka zbog automatizovane reprodukcije sledećeg strima [6].

Odlaganje gašenja odnosi se na obrasce korisničkog interfejsa koji namerno skrivaju određene funkcionalnosti u podrazumevanom interfejsu kao što su odjavljivanje iz servisa ili gašenje naloga, što dovodi do nenameravanog produžavanja sesije korišćenja [6].

Živi pesak pažnje odnosi se na obrasce korisničkog interfejsa koji odmah privlače pažnju korisnika i odvlače ga od predviđenog ponašanja bez potrebe za njegovim svesnim pristankom ili izborom [6]. Primer je automatska reprodukcija na prelazak miša, gde korisnik gleda sadržaj iako nije svojevlasno izabrao da ga pogleda.

Pristrasnost se odnosi na obrasce korisničkog interfejsa koji nesrazmerno naglašavaju i preopterećuju korisnika i njegove interese [6]. Primer obrasca jeste beskonačno dugo skrolovanje/listanje preporuka na osnovu korisnikove prethodne istorije strimovanja.

Pored pozitivnih i negativnih praksi istraženi su i koncepti koji oblikuju korisničko iskustvo kao što su: zečije rupe, totalno otupljivanje i vrteške.

Zečije rupe odnose se na sekvencijalne različite slučajeve korišćenja uzrokovane ljudskom radoznalošću kroz mrežu međusobnih veza i korelacija [3]. Primer jesu povezani, slični ili predloženi strimovi koji intrigiraju korisnika i pobuđuju njegovu radoznalost dovodeći time do povećanog korišćenja servisa.

Totalno otupljivanje je termin koji se u američkoj medicini odnosi na terminalno bolesne pacijente čiji mozak nije funkcionalan, pa se telesne funkcije održavaju isključivo upotrebom mehaničkih uređaja i hranljivih rastvora [3]. Analogno tome, rad striming servisa može uzrokovati istu moždanu atrofiju prilikom donošenja odluka, rezultujući tako pasivnim korišćenjem.

Vrteške se odnose se na iskustvo u kojem platforma korisnicima plasira sadržaj na osnovu njihovih društvenih i ličnih preferenci [3]. Cilj je da stvori osećaj bliskosti i emotivne veze kroz korisnički interfejs.

Psihološki faktori istraženi su kroz korisničko iskustvo uzrokovano gore navedenim dizajn obrascima, takođe izdvojene su i posebne psihološke kategorije: brend, identitet i ekspresivnost, socijalizacija kao i kolekcionarstvo kao čovekov ukorenjeni poriv.

Analizom postojeće literature, istražen je korisnički stav o primium nalogima.

4. REZULTATI I DISKUSIJA

Analiza kvaliteta primium funkcionalnosti uz pomoć Nilsenovih heuristika ukazuje na visok nivo dizajna i funkcionalnosti Spotifaj platforme. Platforma pruža korisnicima dobru povratnu informaciju za većinu

funkcionalnosti, dizajn je minimalistički i intuitivan, tolerancija na greške je velika i olakšava učenje što opravdava manjak prevencije greške u nekim slučajevima. Poštovanje doslednosti i standarda povećava intuitivnost i takođe olakšava razumevanje funkcionalnosti i korišćenje aplikacije. Navigacija kroz sistem je dobro pokrivena, vitalni ekrani su dostupni gotovo u svakom momentu, fluidnost navigacije obezbeđena je svepristupnom opcijom navigiranja u sekvenci (napred/ nazad). Dobra implementacija svih primium funkcionalnosti ispunjava obećanja primium naloga, i ne postoji velika razlika između dektop i mobilnih aplikacija.

Platforma je analizirana na negativne dizajn obrasce, a rezultati ukazuju na to da postoje indikacije koje izazivaju zavisnost/prekomerno korišćenje aplikacije, ali da oni nisu toliko agresivni kao što je to slučaj kod platformi za strimovanje video zapisa poput Netflixa (engl. *Netflix*) ili Jutjuba (engl. *Youtube*). Tako na primer, prisustvo indikatora koliko je vremena ostalo do kraja numere/podkasta pruža korisnu informaciju koja utiče na kognitivne odluke korisnika. Sa druge strane odustvo statistike koliko vremena korisnik provodi u aplikaciji, potpomaže gubitak percepcije vremena koji kao za rezultat ima preterano korišćenje servisa. Kako je reč o audio sadržaju, cela ideja strimovanja jeste kontinualni tok. Kontinualni tok audio zapisa ne krade podjednako korisnikovu pažnju kao video zapis koji je bogatiji sadržajem i zahteva drugačiju interakciju korisnika. Tako posmatrano šablon maglovite funkcionalnosti se pojavljuje u Spotifaj servisu, ali može biti opravdan ulogom i korišćenjem servisa. Ekstremno odbrojavanje takođe ne izostaje potpuno iz Spotifaj aplikacije. Ne postoje tajmeri koji stvaraju osećaj pristiska za donošenjem odluke, korisnik podešava aplikaciju na autoplej što rezultuje uzastopnom i neprestanom reprodukcijom sadržaja, ukoliko ne želi korisnik može ugasiti tu opciju, i onda je svaka sledeća interakcija sa novim sadržajem koji se reprodukuje čisto voljna. Spotifaj sa druge strane, ne ograničava korisnike da se odjave iz aplikacije, i dozvoljava im da obrišu svoj nalog, što čini platformu otpornom na obrazac odlaganje gašenja. Pored toga, živi pesak pažnje takođe nije preterano zastupljen u aplikaciji, korisnici nisu opterećeni sadržajem koji im namerno odvlači pažnju prilikom strimovanja. Spotifaj preporuke na početnoj stranici mogu se posmatrati kroz prizmu šablona pristrasnosti, tako uslovno rečeno veliki izbor sadržaja može dovesti neke korisnike do paralize izbora. Važno je napomenuti da ovo nije previše naglašeno, sve preporuke su dobro labelirane, horizontalni i vertikalni skrol je konačan. Sve ovo ukazuju na moguću preteranu upotrebu servisa, ali kako postoji balans, i obrasci ne idu u ekstrem, postoji mogućnost da korisnik nije svestan pojave istih i uživa u korisničkom iskustvu i kontrolama koje ima.

Rezultati analize koncepata kao što su zečije rupe, totalno otupljivanje i vrteške takođe pokazuju da postoji mogućnost preterane upotrebe aplikacije. Spotifaj stavlja akcenat na istraživanje, kontrolu toka i nesmetano slušanje muzike, tako pojava koncepata vrteški i totalnog otupljivanja nije uslovljena dizajnom aplikacije već pre korisnikovim navikama slušanja. Zečije rupe sa druge strane stavljaju akcenat na algoritme za preporuku novog

sadržaja i baziraju se na ljudskoj radoznalosti. Spotifaj ima veliki broj korisnika i dosta metapodataka preko kojih se treniraju algoritmi za preporuku [8]. Dobar interfejs i rad algoritama za preporuku garantuju pozitivno korisničko iskustvo i angažovanje za korišćenje servisa.

Od analiziranih psiholoških faktora, najveći značaj imaju brend, identitet i ekspresivnost. Brendirano korisničko iskustvo igra veliku ulogu u korišćenju aplikacije. Pozitivno korisničko iskustvo garantovano dobrim radom algoritama za preporuku, u saradnji sa kontinualnim i prolongiranim korišćenjem aplikacije stvara lojalnost kod korisnika. Lojalni korisnici imaju dobar stav o aplikaciji i potencijalno dovode nove korisnike. Ekspresivnost izražena kroz socijalni aspekt (deljenje muzičkog sadržaja na drugim društvenim mrežama) za platformu predstavlja dobru reklamu. Pokazalo se sa druge strane da socijalizacija sama po sebi nije faktor od presudne važnosti za tranziciju korisnika. Takođe socijalizacija nije primarna funkcionalnost Spotifaj platforme, akcenat je na slušanju i otkrivanju muzike ne i na deljenju i socijalnoj komponenti. Ekspresivnost i reklamiranje aplikacije dolaze spontano kada je korisnik zadovoljan sadržajem, a opcija za deljenje sadržaja na druge mreže je u današnjem vremenu podrazumevana mogućnost svih servisa. Kolekcionarstvo ima svoju ulogu da poboljšava korisničko iskustvo, omogućavajući korisnicima posjedovanje muzike. Pojava striming servisa promenila je način konzumacije muzike, fizičke kolekcije zamenjene su digitalnim plejlistama dostupnim onlajn. Oflajn slušanje muzike predstavlja dobru funkcionalnost koja je pandan nekadašnjem kolekcionarstvu, ali nije se pokazala presudnom za tranziciju korisnika. Međutim značaj kolekcionarstva ogleda se u ostanku korisnika. Prethodna istraživanja pokazuju da korisnici ne žele da napuste servis nakon dugog korišćenja, iz razloga što ne žele da izgube kreirane/sačuvane plejliste.

Zaključno, korišćenje servisa deli se u dve faze, faza privlačenja, koja predstavlja iskustvo potrošača sa uslugom pre donošenja odluke o pretplati, i faza zadržavanja, koja predstavlja iskustvo potrošača sa uslugom nakon što je doneta odluka o pretplati [7]. Adiktivni dizajn obrasci ključni su za prvu fazu primamljivanja jer uzrokuju češće korišćenje aplikacije, koje stvara zavisnost i povezanost sa aplikacijom. Algoritmi za preporuku pomažu u vezivanju korisnika jer povećavaju poverenje u servis, takođe aktivno angažovanje na platformi pobuđuje kolekcionarstvo, što rezultuje lojalnošću. Ograničene funkcionalnosti onda predstavljaju dobru odskočnu dasku za odluku o pretplati. Kada već postoje poverenje i angažovanje, kvalitetnije korisničko iskustvo obećano primium nalogom deluje primamljivije korisnicima. Time se iz prve faze prelazi postepeno u drugu, što u ovom radu posmatrano kao probni period korišćenja aplikacije. Kako primium nosi drugačije korisničko iskustvo, očekivanja i stav korisnika se takođe menjaju prelaskom na primium nalog. Olakšano korišćenje, puna kontrola toka, zadovoljstvo primium funkcionalnostima igraju ključnu ulogu u ostanku korisnika na aplikaciji. Ranije je već potvrđeno da veće angažovanje povećava lojalnost, otključavanjem primium funkcionalnosti poput pune kontrole toka i oflajn slušanja,

platforma još jače utiče na investiranje korisnika, što rezultuje ostanom. Ranija istraživanja pokazala su da cena ne igra ključnu ulogu kada je korisnik već pretplaćen na servis [9], tako da je ostanak na servisu direktna posledica uživanja korišćenja aplikacije kao i prevelike investiranosti da bi se servis napustio.

5. ZAKLJUČAK

U ovom radu odrađena je analiza dizajna i funkcionalnosti Spotifaj platforme, sa ciljem da se pronađu faktori koji utiču na tranziciju korisnika na pretplaćeno korišćenje servisa. Motivacija za analizu ogleda se u utvrđivanju dobrih prakse iz HCI oblasti koje utiču na razvoj profitabilnog poslovnog modela.

Rezultat ovog istraživanja ukazuje na to da je kvalitet funkcionalnosti od presudnog značaja za zadovoljstvo korisnika koji već imaju premium nalog, ili su trenutno u probnom periodu korišćenja. Umereno prisustvo negativnih dizajn obrazaca ima za posledicu povećano korišćenje aplikacije, što stvara osećaj bliskosti. Kvalitetne premium funkcionalnosti u saradnji sa negativnim/adiktivnim dizajn obrascima, podstiču aktivnije korišćenje aplikacije i veće korisničko angažovanje.

Pozitivno korisničko iskustvo predstavlja rezultat dobrog rada algoritama za preporuku. Ovi algoritmi garantuju precizniju personalizaciju koja izaziva poverenje i utiče na psihološke faktore. Tako personalizovani sadržaj pruža brendirano korisničko iskustvo i garantuje bolju ekspresivnost korisnika, koja za sobom vuče socijalizaciju odnosno deljenje muzičkih iskustava sa prijateljima. Povećano korišćenje aplikacije i veće angažovanje na platformi, stvaraju jaku vezu između korisnika i servisa koja rezultuje lojalnošću. Sve navedeno najviše je primetno u premium funkcionalnosti koja obezbeđuje punu kontrolu toka, i koja je onda ključna za tranziciju korisnika koji su već u probnom periodu korišćenja. Ostale premium funkcionalnosti potpomažu stvaranje pozitivnog korisničkog iskustva.

Sa druge strane tranzicija korisnika sa premium naloga na probni period predstavlja teži zadatak, uslovljena je kulturološkim razlikama. Tako su neki korisnici adaptirani na prisustvo reklama dok su reklame nekoj drugoj kulturološkoj grupi dovoljan razlog za pretplatu. U radu je potvrđeno na koje načine platforma zadržava i premium i free korisnike, i koji su kvaliteti premium naloga ključni u pretplatničkom modelu korišćenja.

Naredna istraživanja mogla bi iskoristiti metodologiju ovog rada za kompletan eksperiment nad različitim grupama korisnika koji bi preciznije odgovorio na pitanje zbog čega korisnici probaju probni period korišćenja.

6. LITERATURA

- [1] MORRIS, Jeremy Wade; POWERS, Devon. Control, curation and musical experience in streaming music services. *Creative Industries Journal*, 2015, 8.2: 106-122.
- [2] KIM, Jiwhan; NAM, Changi; RYU, Min Ho. What do consumers prefer for music streaming services?: A comparative study between Korea and US. *Telecommunications Policy*, 2017, 41.4: 263-272.
- [3] MCKELVEY, Fenwick; HUNT, Robert. Discoverability: Toward a definition of content discovery through platforms. *Social Media+ Society*, 2019, 5.1: 2056305118819188.
- [4] MAASØ, Arnt; SPILKER, Hendrik Storstein. The streaming paradox: Untangling the hybrid gatekeeping mechanisms of music streaming. *Popular Music and Society*, 2022, 45.3: 300-316.
- [5] LEE, Jin Ha; PRICE, Rachel. User experience with commercial music services: An empirical exploration. *Journal of the Association for Information Science and Technology*, 2016, 67.4: 800-811.
- [6] CHAUDHARY, Akash, et al. "Are You Still Watching?": Exploring Unintended User Behaviors and Dark Patterns on Video Streaming Platforms. In: *Designing Interactive Systems Conference*. 2022. p. 776-791.
- [7] NAKAYAMA, Makoto. Are You Hooked on Paid Music Streaming?. *International Journal of E-Business Research*, 2018, 14.1.
- [8] MAASØ, Arnt; HAGEN, Anja Nylund. Metrics and decision-making in music streaming. *Popular Communication*, 2020, 18.1: 18-31.
- [9] THEADORA, Clarissa, et al. How does involvement build loyalty towards music-streaming platforms? A multi-analytical SEM-ANN technique. *Journal of Product & Brand Management*, 2022, ahead-of-print.

Kratka biografija:



Milica Poparić rođena je u Arandelovcu 5. novembra 1998. godine. Osnovne akademske studije završila je 2021. godine na Fakultetu tehničkih nauka. Iste godine upisuje master akademske studije na studijskom programu Softversko inženjerstvo i informacione tehnologije.

**ПРИМЕНА IoT ТЕХНОЛОГИЈЕ ЗА БОЉЕ РАЗУМЕВАЊЕ И УПРАВЉАЊЕ
ОКОЛИНОМ****UTILIZING IOT TECHNOLOGY FOR ENHANCED UNDERSTANDING AND
MANAGEMENT OF THE ENVIRONMENT**Александра Цвјетићанин, *Факултет техничких наука, Нови Сад***Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО**

Кратак садржај – У раду је изложена имплементација система који су базирани на Internet of Things технологији. Први пројекат се односи на управљање радом DHT22 сензора путем корисничког интерфејса, док се други пројекат базира на модификацији рада Sonoff POWR2 паметне утичнице, при чему је омогућено праћење параметара енергетске потрошње. Добијени подаци се у оба пројекта приказују на платформи ThingSpeak.

Кључне речи: Internet of Things, микроконтролер ESP32, IoT платформа ThingSpeak, паметна утичница Sonoff POWR2

Abstract – The master thesis represents the implementation of systems based on Internet of Things technology. The first project involves managing the operation of DHT22 sensors through a user interface, while the second project is based on modifying the operation of the Sonoff POWR2 smart socket, enabling the monitoring of energy consumption parameters. The collected data in both projects are displayed on the ThingSpeak platform.

Keywords: Internet of Things, microcontroller ESP32, IoT platform ThingSpeak, smart switch Sonoff POWR2

1. Увод

Internet of Things (IoT) представља револуционарну парадигму која трансформише начин на који свет функционише, као и огроман потенцијал за даљи економски раст. Ова технологија омогућава повезивање физичких уређаја са интернетом, што резултује могућношћу даљинског праћења и анализе различитих аспеката свакодневног живота.

Први пројекат се фокусира на управљање сензором DHT22 кроз веб интерфејс. Праћење временских услова у окружењу је могуће посредством платформе ThingSpeak.

Други пројекат је базиран на коришћењу Sonoff паметне утичнице и ефикасном праћењу енергетске потрошње путем платформе ThingSpeak.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био др Јован Бајић, ванр. проф.

**2. ИНТЕГРАЦИЈА DHT22 СЕНЗОРА И
ПЛАТФОРМЕ ThingSpeak****2.1. Основни принцип рада система**

Пројекат се темељи на повезивању сензора DHT22 са ESP32 микроконтролером, при чему се за имплементацију кода користи окружење Arduino IDE. Након спуштања кода, корисницима је омогућен приступ веб графичком интерфејсу преко одређене IP адресе. Дизајн интерфејса је креиран применом HTML и CSS технологије, док је функционалност остварена писањем скрипте у JavaScript језику. Корисник има могућност управљања над системом притиском на тастере који одговарају следећим акцијама:

- активирање система и константно слање података
- деактивирање система
- слање једног податка температуре и влажности
- подешавање температурног прага и приказ температуре при одређеном услову
- подешавање прага влажности и приказ података при одређеном услову

Команде се шаљу контролном каналу Control ESP32 у оквиру ThingSpeak платформе у виду целобројног броја од 0 до 5. Микроконтролер ESP32 преузима команде и извршава жељену функционалност захваљујући коду написаном у оквиру Arduino окружења. Подаци који се прослеђују Cloud платформи се чувају у оквиру канала за праћење под називом Temperature & Humidity.

2.2. Хардверске компоненте

За реализацију описаног система, коришћене су следеће хардверске компоненте:

ESP32 [1] представља микроконтролер који има велики број могућности, као што су WiFi, Bluetooth модул и WebServer захваљујући којима је нашао примену у Internet of Things апликацијама. Ову плочицу одликује ниска потрошња, многобројни GPIO пинови и велика количина RAM меморије, која је кључна за обраду података и извршавање захтевних апликација.

DHT22 сензор [2] је дигитални сензор температуре и влажности који има примену у пројектима, као што су мерење временских услова у околини, аутоматизација

стамбених простора, контрола влажности у стакленицима и надзор. Овај сензор пружа прецизно читавање температуре и влажности и једноставан за употребу.

Неопходно је на правилан начин повезати описане компоненте. Пин са ознаком „+“ представља *VCC* и користи се за напајање сензора. Пин означен са „-“ представља *GND* пин повезује се са масом. За слање читаних података о температури и влажности са сензора ка микроконтролеру, користи се пин са ознаком „out“ и повезује се на пин са ознаком *GPIO16*.

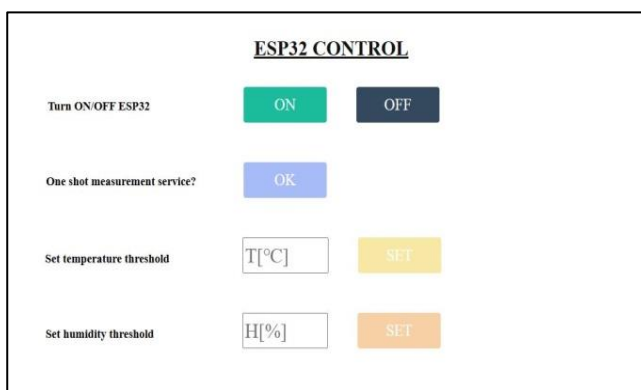
2.3. Софтверски алати и потребна подешавања

За имплементацију програмског кода, коришћено је развојно окружење *Arduino IDE (Arduino Integrated Development Environment)*. Овај софтверски алат користи једноставну синтаксу, што омогућава корисницима ефикасно програмирање. Поседује велики број библиотека које олакшавају рад са различитим компонентама и периферијским уређајима. За приказ података користи се визуелни монитор и график. Додатне погодности овог софтверског алата се односе на могућност коришћења низа готових примера. Конкретно у овом пројекту је коришћена *FireBeetle-ESP32* верзија плочице.

ThingSpeak [3] представља *Cloud* платформу која омогућава корисницима да прикупљају, анализирају и визуализују податке са коришћених периферних уређаја. Након успешне регистрације на платформу, процес рада започиње креирањем канала, на коме се складиште прикупљени подаци. За упис и читање података са платформе се користе кључеви, и то *Write API key* и *Read API key*. У оквиру овог пројекта су реализована два канала. Канал *Control ESP32* поседује поља *Control ESP32*, *Temperature threshold* и *Humidity threshold*. Други канал *Temperature & Humidity* има два поља *Temperature* и *Threshold* и служи за приказ и праћење временских параметара.

2.4. Графички интерфејс апликације

Управљање системом је омогућено посредством графичког интерфејса [4], за чије креирање су коришћене неке од основних технологија које се користе за развој веб страница и апликација – *HTML*, *CSS* и *JavaScript*. На слици 1 приказан је дизајн креираног графичког интерфејса. Креиране су четири различите секције за манипулацију над системом:



Слика 1. Графички интерфејс

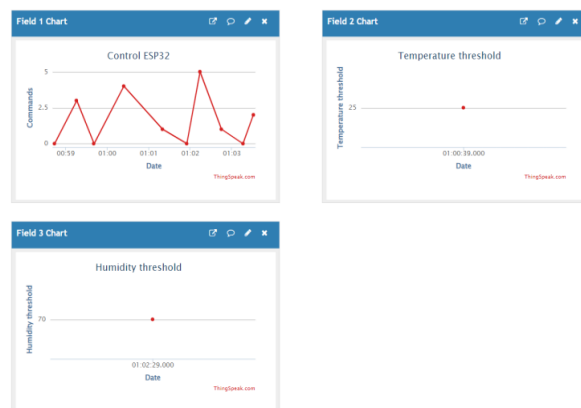
- константно мерење и слање температуре на платформу *ThingSpeak* или деактивирање система
- слање једне вредности температуре и влажности
- подешавање температурног прага
- подешавање прага влажности

2.5 Опис програмског кода

Код је написан у оквиру *Arduino* скрипте. У функцији *setup()* је извршена основна иницијализација система и повезивање *ESP32* модула на *WiFi* мрежу. Након покретања сервера, на одређеној локалној *IP* адреси се приказује графички интерфејс са слике 1, што је интегрисано у склопу функције *sendHTML()*. Основна функционалност система је имплементирана у оквиру *loop()* бесконачне петље. Она се заснива на читању команди са *Control ESP32* канала, које се уписују након притиска на неки од тастера са графичког интерфејса. У зависности од прочитане команде, извршава се жељена функционалност, читањем и слањем података на платформу *ThingSpeak*. Након извршавања сваке команде, прослеђује се нулта вредност на контролни канал, како би се систем вратио у нулто стање, при чему се спречава двоструко читање истих команди.

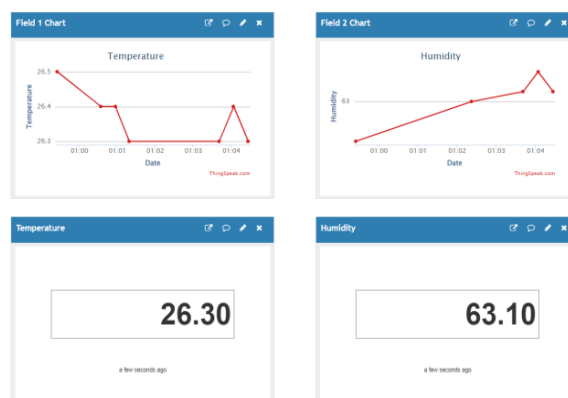
2.6 Резултати

На слици 2 приказан је изглед канала *Control ESP32*.



Слика 2. Канал *Control ESP32*

За праћење вредности температуре и влажности, користи се канал приказан на слици 3.



Слика 3. Канал *Temperature & Humidity*

Графички приказ *Temperature* служи за складиштење вредности температуре, док се график *Humidity* односи на приказ влажности. Могуће је прочитати и вредност последњих параметара на елементима испод графика.

3. ИНТЕГРАЦИЈА УТИЧНИЦЕ Sonoff POWR2 И Платформе ThingSpeak

3.1 Паметна утичница Sonoff POWR2

При реализацији овог пројекта, коришћена је паметна утичница Sonoff POWR2 [5]. Утичница подржава рад на стандардном напону од 110V или 220V, у зависности од региона.

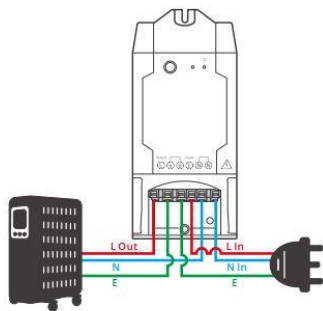
Радна струја износи 15A. Рад са овом утичницом нуди низ могућности, као што су даљинско управљање, праћење енергетске потрошње, компатибилност са гласовним асистентима и друго.

3.2 Инсталација паметне утичнице

За могућност рада са паметном утичницом Sonoff POWR2 неопходно је обезбедити њено правилно повезивање, што је приказано на слици 4.

Црвене жице представљају фазне жице, што значи да носе струју и у нашем случају су означене са *L Out* и *L In*. Плаве жице означене са *N* и *N In* представљају неутралне жице и одговорне су за враћање електричне струје назад до извора.

Две жуто-зелене жице са ознаком *E* представљају заштитне уземљиваче. Оне служе као безбедносна мера и обезбеђују да се струја преусмери у случају кратког споја или других електричних проблема.



Слика 4. Инсталација паметне утичнице

3.3 Повезивање утичнице са мобилном апликацијом

Апликација *eWeLink* је популарна апликација за управљање паметним уређајима путем паметних телефона, као и за приказ параметара, као што су струја, снага и напон. Доступна је за *Android* и *iOS* уређаје.

За успешно повезивање паметне утичнице са апликацијом, неопходно је пратити одговарајуће инструкције дате од стране произвођача.

Додатна могућност јесте интеграција апликације са говорним асистентима.

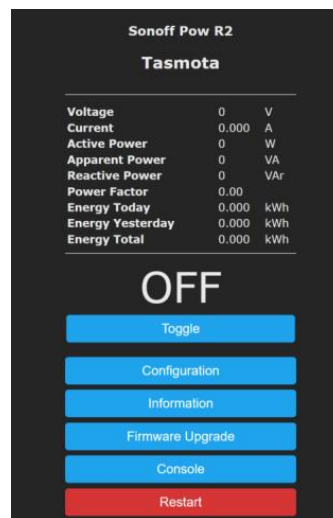
У оквиру пројекта је коришћен *Google Assistant* за активирање и деактивирање утичнице путем гласовне команде.

3.4 Модификација утичнице коришћењем фирвера Tasmota

Како је основна идеја овог пројекта праћење параметара потрошње путем платформе *ThingSpeak*, дошло је до одређених модификација. За успешну реализацију система, коришћен је фирмвер *Tasmota*, који представља *open-source* фирмвер, који је често присутан код паметних уређаја. Њега је могуће убацити у паметну утичницу применом самосталне апликације *Tasmotizer* [6], коју је прво потребно инсталирати. За убацивање фирмвера, коришћен је *USB to TTL* конвертор, који је са једне стране повезан са портом лаптоп-а, док се друга страна путем краткоспојника повезује на одговарајуће отворе на штампаној плочи паметне утичнице. Притиском на тастер *Tasmotize* у оквиру интерфејса апликације, у утичницу се пребацује описани фирмвер. Након повезивања уређаја на *WiFi* мрежу, у оквиру опције *Get IP* се може прочитати вредност локалне адресе паметне утичнице.

3.5 Веб интерфејс

Када се добијена *IP* адреса унесе у било који веб претраживач, отвара се интерфејс новог *Tasmota* фирмвера, који је приказан на слици 5.



Слика 5. Интерфејс Tasmota фирмвера

Поред основне функционалности за укључивање, искључивање утичнице и праћења тренутне потрошње повезаних уређаја, овде је доступан и низ других подешавања, углавном у оквиру картице *Configuration*.

3.6 Конфигурација MQTT комуникационог протокола

MQTT (*Message Queuing Telemetry Transport*) [7] протокол је базиран на клијент-сервер архитектури. Централни део представља *MQTT* брокер који прихвата податке од клијената и има задатак да их усмери на одговарајући начин. Кључни концепт у *MQTT* протоколу је тема (*topic*). *MQTT* користи модел *publish-subscribe*. Идеја је да међу клијентима постоји

такозвани *Publisher*, односно уређај који објављује податке на одређене теме, као и *Subscriber*, односно уређај који може да се претплати на тему и са ње преузме жељене податке.

За потребе овог пројекта, коришћен је јавно доступан MQTT брокер *broker.emqx.io* који је доступан на порту 1883. Клијент који шаље податке на брокер јесте паметна утичница. За коришћење овог протокола, неопходно је одрадити подешавања приказана на слици 6.



Слика 6. Конфигурација MQTT протокола

Праћење података који се шаљу је могуће у оквиру картице *Console*. Подаци се шаљу у JSON формату.

3.7 Програмирање ESP32 модула

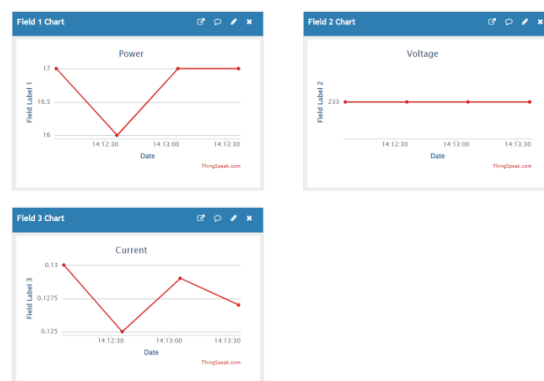
Писање програмског кода је извршено у оквиру окружења *Arduino IDE*. За преузимање параметара енергетске потрошње коришћен је ESP32. Овај микроконтролер је други MQTT клијент, односно *Subscriber*, чија се улога односи на претплаћивање на тему и преузимање жељених података.

Како су преузети подаци написани у JSON, коришћењем инсталираних библиотека су издвојени жељени подаци – снага, напон и струја. Ови параметри се шаљу на *ThingSpeak* канал *Sonoff*, на ком је могуће пратити пристигле податке о енергетској потрошњи уређаја.

3.8 Резултати

За приказ параметара потрошње је коришћена IoT платформа *ThingSpeak*. Слика 7 представља графички приказ параметара енергетске потрошње у одређеном временском интервалу. Графици представљају поља у оквиру креираног канала *Sonoff* и садрже информације о потрошњи енергије за основна три аспекта: снагу, напон и струју.

С обзиром на приказане податке о снази, напону и струји, ова слика нам пружа важне информације о ефикасности и употреби енергије у нашем систему. На основу ових података можемо да проучимо и оптимизујемо нашу енергетску потрошњу и донесемо одлуке о управљању енергетским ресурсима.



Слика 7. Канал *Sonoff*

4. ЗАКЉУЧАК

Реализовани пројекти дају увид у потенцијал и велики значај *Internet of Things* технологије и истичу способност технологије да промени, унапреди и олакша наш начин живота и рада. На основу приказане документације, може се закључити да су пројекти успешно реализовани, односно да је задовољена жељена функционалност оба система. Унапређења у овим пројектима могућа су у виду додатних функција, већег броја сензора и бољих алгоритама за анализу података.

5. ЛИТЕРАТУРА

- [1] <https://www.alldatasheet.com/datasheetpdf/pdf/1148023/ESPRESSIF/ESP32.html>, април 2023.
- [2] <https://www.alldatasheet.com/view.jsp?Searchword=DHT22&sField=4>, април 2023.
- [3] <https://thingspeak.com/>, април 2023.
- [4] <https://www.electronicwings.com/users/shravanprajapati122/projects/228/led-control-via-thingspeak-server-using-nodemcu-esp8266>, мај 2023.
- [5] <https://sonoff.tech/product-document/diy-smart-switches-doc/powr2-doc/>, мај 2023.
- [6] https://www.youtube.com/watch?v=liEpVW9INsY&ab_channel=G6TechnologyServices, мај 2023.
- [7] <https://mqtt.org/>, мај 2023.

Кратка биографија:



Александра Цвјетићанин рођена је у Новом Саду, 1999. Године. Дипломски рад на Факултету техничких наука из области Примењене електронике је одбранила 2022. године.
контакт: cvjetanina99@gmail.com

**ДЕТЕКЦИЈА БОЛЕСТИ КОВИД-19 НА СНИМЦИМА ПЛУЋА УПОТРЕБОМ
КОНВОЛУЦИОНИХ НЕУРОНСКИХ МРЕЖА****DETECTION OF COVID-19 VIRUS ON CT SCANS USING CONVOLUTIONAL NEURAL
NETWORKS**Никола Ђуровић, *Факултет техничких наука, Нови Сад***Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО**

Кратак садржај – Пандемија КОВИД-19 вируса изазвала је глобалну кризу са великим бројем случајева и смртних исхода. У раду се истражује употреба снимка плућа добијених употребом компјутерске томографије (ЦТ) за брзу детекцију вируса, користећи дубоко учење и различите моделе конволуционих неуронских мрежа. Рад анализира претходна решења, теорију конволуционих неуронских мрежа, имплементацију модела и презентује добијене резултате.

Кључне речи: машинско учење, конволуционе неуронске мреже, дубоко учење, класификација

Abstract – The COVID-19 pandemic has triggered a global crisis with a high number of cases and fatalities. This paper explores the use of lung images obtained through computer tomography (CT) for the rapid detection of the virus, employing deep learning and various convolutional neural network (CNN) models. The paper analyzes previous solutions, convolutional neural networks theory, describes model implementation, and presents the obtained results.

Keywords: machine learning, convolutional neural networks, deep learning, classification

1. УВОД

Пандемија COVID-19 вируса представља глобалну кризу са великим бројем потврђених случајева и смртних исхода. Вирус се брзо шири и изазива симптоме као што су кашаљ, грозница и респираторне проблеме. Поред PCR и антигенских тестова, за откривање болести се користи и прецизнија компјутерска томографија (ЦТ) иако се појављује ризик од зрачења. Упркос потенцијалним ризицима, ЦТ се сматра бржом и поузданијом техником за дијагнозу КОВИД-19 вируса [1]. Метода захтева присуство радиолога, што може бити изазов. Аутоматизација анализе ЦТ снимака постаје пожељна како би се омогућила брза дијагноза болести. Проблеми у анализи медицинских слика често произилазе из недостатка јавно доступних ЦТ снимака због заштите приватности пацијената.

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Владимир Бугарски, доцент.

Зато се често користи метода пренесеног учења, користећи већ обучене моделе. Дубоко учење се истиче као ефикасан метод за класификацију медицинских слика и дијагнозу болести. У овом истраживању су предложени модели за откривање КОВИД-19 на основу ЦТ снимака, користећи различите мрежне архитектуре. Истражене су и различите технике подешавања параметара и обогаћивања скупа података. Рад пружа преглед постојећих решења, теоријски оквир неуронских мрежа и детаље имплементације предложених модела.

2. ПОСТОЈЕЋИ МОДЕЛИ

У овом раду користе се подаци са *Kaggle* платформе [2], који су једни од ретких јавно доступних скупова података са ЦТ снимцима плућа за анализу КОВИД-19. Скуп података се састоји од 1.252 КОВИД-19 снимка и 1.229 снимака здравих плућа, димензија 224x224x3 пиксела.

У раду [3], исти скуп података се користи за испитивање ефикасности различитих модела дубоког учења и пренесеног учења. *InceptionV3* модел постиже највишу тачност од 99,59%, док је *VGG16* најмање ефикасан са тачношћу од 69,08%.

У [4], користе се подаци прикупљени из четири института у Кини, са 346 КОВИД снимака и 463 снимака здравих плућа. Разматране су различите архитектуре, укључујући *CTnet-10*, *VGG-19*, *Inception V3*, *DenseNet-169* и *VGG-16*. Најбољи резултат постиже *VGG-19* са тачношћу од 94,12%, док *Inception V3* има најнижу тачност од 53,14%. *CTnet-10* показује добре резултате са тачношћу од 82,1%, а обучавање овог модела траје најкраће (130 секунди).

У истраживању [5], користи се највећи скуп података из Кине са додатном техником обогаћивања података. Развијен је модел за вишеструку класификацију са тачношћу од 99%, способан за детекцију КОВИД-19, других облика пнеумоније и здравих плућа.

У раду [6], користи се велики скуп података који није јавно доступан, са укупно 63.849 снимака. Разматране су архитектуре као што су *Xception*, *ResNet50* и модел развијен од аутора овог рада за ову сврху. Модел који комбинује *Xception* и *ResNet50* постиже тачност од 98%, уз примену различитих техника обогаћивања података.

Ова истраживања показују различите приступе и modele за детекцију КОВИД-19 на основу ЦТ снимака плућа.

3. НЕУРОНСКЕ МРЕЖЕ

Вештачке неуронске мреже су један од алгоритама учења са надзором. Класификација улазних података, на основу њихових карактеристика, у једну од класа је задатак истренираног модела. Вештачки неурон је основни градивни елемент сваке вештачке неуронске мреже, које представљају основ за разумевање конволуционих неуронских мрежа.

3.1. Конволуционе неуронске мреже

Конволуционе неуронске мреже (КНН) су посебне врсте неуронских мрежа које су намењене за обраду визуелних података као што су слике и видео записи. Основна сврха ових мрежа је идентификација облика, карактеристика и шаблона у овим подацима.

КНН користе низ слојева за постизање овог циља:

- **Конволуциони слојеви:** Ови слојеви примењују филтере на улазне податке како би открили различите карактеристике, као што су ивице, углови и текстуре. Ово омогућава мрежи да научи репрезентације слика на различитим нивоима апстракције.
- **Слојеви за сажимање (pooling):** Ови слојеви служе за смањење димензионалности података тако што задржавају најважније информације из претходних слојева. Најчешће се користи сажимање максимумом, где се узима максимална вредност из одређених региона.
- **Слој равнања:** Задужен је да вишедимензионалне низове који се добијају као излази из претходних слојева трансформише у једнодимензионални вектор
- **Нормализациони слојеви:** Додавање нормализационих слојева се користи када улазне вредности неке неуронске мреже нису у истом распону, те се помоћу ових слојева све вредности скалирају на исти опсег, што се одражава на време трајања тренинга или конвергирања мреже.
- **Потпуно повезани слојеви:** Након издвајања карактеристика у претходним слојевима, потпуно повезани слојеви се користе за класификацију или регресију. Ови слојеви донесу крајње одлуке на основу научених карактеристика.

Конволуционе неуронске мреже су широко коришћене у разним задацима обраде слика, као што су класификација објеката, детекција лица, сегментација слика и друге апликације у области компјутерске визије. Њихова архитектура омогућава ефикасно препознавање облика и структура у визуелним подацима, чинећи их кључним алатом у домену дубоког учења.

3.2. Параметри мреже

Развој и обука конволуционих неуронских мрежа је комплексан процес који захтева пажљиву селекцију и конфигурацију различитих компоненти како би се постигао висок ниво тачности и генерализације модела. Неколико кључних аспеката овог процеса су:

Подела података

Први корак је подела доступних података на три основна скупа:

- **Тренинг скуп:** Овај скуп се користи за обуку мреже. Обично чини већину доступних података (нпр. 60% - 80%). Модел се прилагођава овим подацима како би минимизирао грешку и унапредио своју способност класификације.
- **Валидациони скуп:** Валидациони скуп се користи током тренинга како би се пратила перформанса модела на подацима које није видео током тренинга. То је важно за детекцију претераног прилагођавања и оптимизацију параметара.
- **Тестни скуп:** Овај скуп се користи за коначну евалуацију перформанси модела након што је тренинг завршен. Модел се тестира на овим подацима како би се проценило његово понашање на потпуно новим подацима.

Параметри и активационе функције

Параметри и активационе функције играју кључну улогу у архитектури КНН-а. Одабир функција губитка, броја скривених слојева, броја кернела, величина слојева за сажимање и иницијалне тежине директно утичу на перформансе модела. Активационе функције као што су ReLU, хиперболички тангенс и софтвакс играју кључну улогу у преносу информација кроз мрежу и брзини учења.

Регуларизација

Да би се спречило претерано прилагођавање модела на тренинг подацима, користе се различите регуларизационе технике. То укључује:

- **Искључивање (Dropout):** Ово је техника која насумично искључује неуроне током тренинга како би се повећала робусност и генерализација модела.
- **L2 и L1 регуларизација:** Додавање казних израза у функцију губитка како би се тежине држале мале и спречило преприлагођавање.
- **Рано заустављање (Early stopping):** Праћење разлике у вредностима функције губитка током тренинга и прекидање тренинга ако се перформансе на валидационом скупу погоршавају.
- **Обогаћивање података:** Када су доступни подаци ограничени, примена трансформација на улазне слике, као што су ротација, обртање, померање, зумирање и др. може помоћи у повећању количине доступних података и унапређењу генерализације.

Пренесено учење

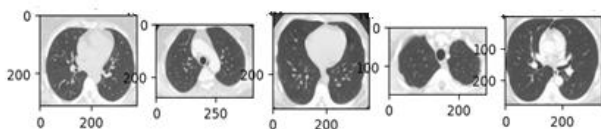
У области компјутерске визије, где се често користе велики скупови података, пренесено учење је моћна техника. Ово подразумева коришћење већ обучених модела (нпр. VGG-16, ResNet) као основе за обуку нових модела за специфичне задатке. Ово може значајно убрзати процес тренинга и побољшати перформансе. Одабир архитектуре модела, као што су VGG-16, VGG-19, ResNet50, InceptionV3, MobileNet и Xception, такође игра кључну улогу. Свака од ових

архитектура има своје специфичне карактеристике и предности.

У суштини, развој и обука КНН-а захтева балансирање многих фактора како би се постигле високе перформансе и генерализација на циљаном задатку. Правилно конфигурисање ових компоненти је кључно за развој поузданог и ефикасног модела неуронске мреже.

4. РЕЗУЛТАТИ И ДИСКУСИЈА

За обуку модела коришћен је *Google Colaboratory* алат, који је бесплатна платформа за извршавање *python* кода на *cloud-u*. Овај алат пружа интерактивно окружење за писање и извршавање кода користећи Јупитер свеске, а такође омогућава приступ графичким и тензор процесорским јединицама (ГПУ и ТПУ) за убрзање рачунања, што је посебно важно код дубоког учења. Коришћени подаци су прикупљени у болници у Сао Паолу, у Бразилу, са циљем развоја модела за рано откривање вируса. Примери снимка из скупа података приказани су на слици 1.



Слика 1. Примери ЦТ снимка плућа из скупа података

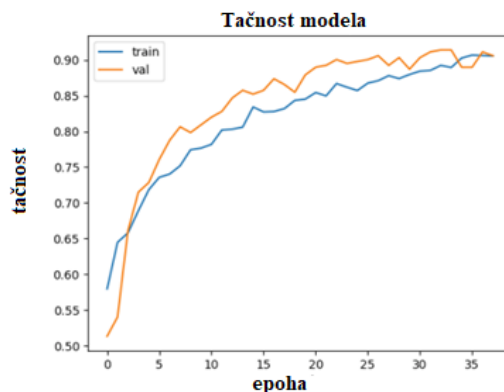
У истраживању су размотрене различите архитектуре КНН модела, укључујући једноставније моделе са неколико конволуционих слојева и слојевима за сажимање, али и комплексније моделе. Разматране су различите технике за постизање бољих перформанси модела, као што су слојеви искључивања, Л2 регуларизација, рано заустављање и обogaћивање података. Као резултат овог истраживања, развијен је модел који постиже тачност од **90%** у детекцији КОВИД-19 вируса на ЦТ снимцима плућа. Осим тога, истраживање упоређује овај модел са другим моделима базираним на пренесеном учењу (*Inception V3*, *VGG-19* и *Xception*), приказујући сличне резултате у тачности детекције.

Развој архитектуре за детекцију КОВИД-19 на ЦТ снимцима плућа почео је са основном конфигурацијом неуронске мреже. Та архитектура укључивала је само 2 конволуциона слоја, слојеве за сажимање и потпуно повезане слојеве. Међутим, током експериментисања са овом почетном конфигурацијом, појавило се неколико проблема. Први проблем је претерано прилагођавање модела подацима за тренинг, што значи да је модел имао веома добре перформансе на тренинг скупу, али је на валидационом и тест скупу лоше генерализовао. Примећене су и осцилације у перформансама модела током тренинга, што указује на нестабилне резултате. У циљу решавања ових проблема вршени су експерименти са различитим техникама и конфигурацијама. Прва промена укључивала је додавање регуларизационих техника, попут *batch* нормализације и слојева искључивања. Ове технике су помогле да се

смањи претерано прилагођавање, али осцилације су и даље биле присутне. Следећа измена је била повећање броја филтера у конволуционим слојевима и додавање Л2 регуларизације, што је делимично побољшало перформансе, али осцилације су и даље биле проблем. Кључни тренутак у побољшању архитектуре дошао је када су постављена 2 уместо 1 неурона на излазном слоју мреже. То је довело до преласка са бинарне на категоријску унакрсну ентропију када је у питању функција губитка, и коришћење софтверских активационих функција. Ово је значајно побољшало тачност и смањило претерано прилагођавање.

Додатно је прилагођена стопа учења и подешени су параметри на слоју искључивања како би се смањиле осцилације и добили стабилнији резултати током тренинга. Коначно, најбољи резултати су постигнути са финалном архитектуром која се састојала од дванаест слојева, од којих су по 2 конволуциона и *batch* нормализациона слоја, 2 слоја сажимања, 3 слоја искључивања и по 1 потпуно повезани, слој равнања и излазни слој.

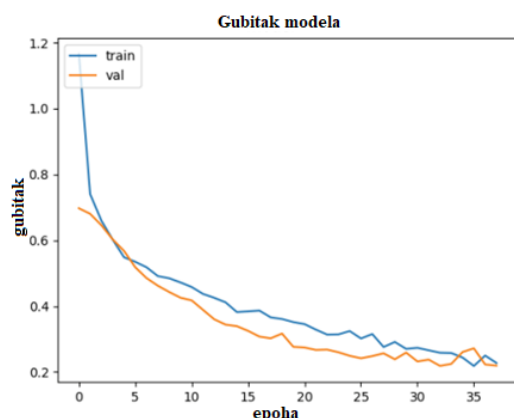
Коришћена је и техника раног заустављања како би се елиминисале осцилације. Постигнута је највиша тачност од **90%** и најмањи губитак од **0,2** на свим скуповима података. Број епоха је био 50, величина *batch-a* 64, а стопа учења 0,0001. Изабран је АДАМ тип оптимизатора. Параметар на прва два слоја искључивања је подешен на 0,2, а на трећем слоју, који се налази непосредно иза излазног слоја, је 0,5. На слици 2 графички је приказана тачност, а на слици 3 губитак на овом моделу. Ова архитектура је се показала као стабилна и добро је генерализовала нове податке, елиминишући претходне проблеме.



Слика 2. Тачност модела на финалној архитектури

Када је у питању пренесено учење, резултати који су постигнути код одређених модела пренесеног учења су сумирани у табели 1. *VGG-19* модел достигао је тачност од 93,79%. Коришћен је АДАМ оптимизатор, а максималан број епоха је 30. Стопа учења је постављена на 0,0001, а величина *batch-a* на 32. Коришћени су исти параметри као и у раду [5]. *Inception V3* архитектура је постигла тачност од 99,19%. Максималан број епоха је 100, а коришћен је АДАМ оптимизатор и стопа учења од 0,001 и величина *batch-a* 16. Коришћена је и регуларизациона техника раног заустављања, а примењени су параметри из рада [4]. *Xception* модел је показао тачност од 97,55%. Максималан број епоха је 50,

величина *batch-a* је 16, а оптимизатор АДАМ. Стопа учења је подешена на 0,0001. Сви параметри су узети из рада [10].



Слика 3. Губитак модела на финалној архитектури

Табела 1. Резултати на моделима пренесеног учења

Модел	Тачност
VGG-19	93,79%
Inception V3	99,19%
Xception	97,55%

Предности модела у разматраним радовима [3, 4, 5, 6] у односу на моделе који су развијани у овом раду, огледају се у пажљивијем бирању параметара мреже, али и у много већим и разноврснијим скуповима података.

5. ЗАКЉУЧАК

У овом раду, истраживање је спроведено са циљем развијања ефикасног система за детекцију КОВИД-19 вируса на ЦТ снимцима плућа применом конволуционих неуронских мрежа. Након пажљиве анализе најновијих достигнућа у овој области, примењене су неке нове архитектуре како би се показала њихова ефикасност код решавања оваквог медицинског проблема.

Кључни допринос овог рада огледа се у развоју иновативне архитектуре неуронске мреже која је пројектована да ефикасно препознаје и класификује присуство КОВИД-19 вируса на ЦТ снимцима плућа. Кроз систематично експериментисање и оптимизацију параметра, постигнута је тачност од **90%**.

Резултати овог рада наглашавају важност примене дубоког учења и конволуционих неуронских мрежа у медицинској дијагностици, посебно у контексту глобалне пандемије која је трајала до маја 2023. године. Употреба ЦТ снимака плућа као улазних података омогућила је брзу, неинвазивну и поуздану детекцију вируса, чиме се медицинском особљу пружа моћан алат у борби против ове болести.

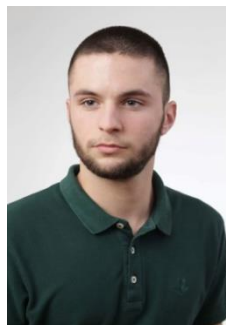
Ипак, треба напоменути да даље истраживање и валидација резултата на већем скупу података треба да се спроведе како би се потврдила генерализација и поузданост предложене архитектуре. Такође, потребно је истражити могућности за унапређење ефикасности оваквог система, као и интеграцију у клиничку праксу ради шире примене и помоћи докторима за лакшу и бржу детекцију болести.

Овај рад је дао допринос у пољу дијагностике КОВИД-19 вируса применом конволуционих неуронских мрежа на ЦТ снимцима плућа. Резултати и предложене архитектуре отварају врата за даље истраживање и развој напредних метода детекције овог вируса, која би могла бити од кључне важности за брзу и прецизну дијагнозу.

6. ЛИТЕРАТУРА

- [1] A. Bernheim, X. Mei, M. Huang, Y. Yang, Z.A. Fayad, N. Zhang, K. Diao, B. Lin, X. Zhu, K. Li, "Chest CT findings in coronavirus disease 2019 (COVID-19): Relationship to duration of infection." *Radiology*, Vol. 295, pp. 685–691, 2020.
- [2] <https://www.kaggle.com/datasets/plameneduardo/sarscov2-ctscan-dataset> (pristupljeno u avgustu 2023.)
- [3] M.M. Sufian, E.G. Moun, M.H.A. Hijazi, F. Yahya, J.A. Dargham, A. Farzamnia, F. Sia, N.F. Mohd Naim, "COVID-19 Classification through Deep Learning Models with Three-Channel Grayscale CT Images." *Big Data Cogn. Comput.*, 7, 36, 2023.
- [4] V. Shah, R. Keniya, A. Shridharani, M. Punjabi, J. Shah, N. Mehendale, "Diagnosis of COVID-19 using CT scan images and deep learning techniques." *Emerg Radiol*, Vol. 28(3), pp. 497-505, 2021.
- [5] W. Zhao, W. Jiang, X. Qiu, "Deep learning for COVID-19 detection based on CT images." *Sci Rep*, 11, 14353, 2021.
- [6] M. Rahimzadeh, A. Attar, S.M. Sakhaei, "A fully automated deep learning-based network for detecting COVID-19 from a new and large lung CT scan dataset." *Biomed Signal Process Control*, 68, 102588, 2021.

Кратка биографија:



Никола Ђуровић уписао је Факултет техничких наука, смер рачунарство и аутоматика. 2021. године завршава основне академске студије са просечном оценом 9,22, а исте године уписује мастер студије на истом смеру. Положио је све испите на мастер академским студијама предвиђене планом и програмом са просечном оценом 10.
контакт: ndjurovic53@yahoo.com

DIGITALNA PLATFORMA ZA UPRAVLJANJE KVAROVIMA I PLANIRANJE REMONTA U ELEKTROENERGETSKIM SISTEMIMA**DIGITAL PLATFORM FOR FAILURE MANAGEMENT AND REPAIR PLANNING IN POWER SYSTEMS**

Željka Matović; Fakultet tehničkih nauka, Novi Sad

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj rad istražuje primjenu digitalne platforme za upravljanje kvarovima i planiranje remonta u elektroenergetskim sistemima u cilju njihovog poboljšanja. Izvršena je analiza ključnih prednosti aplikacije, uključujući brži odgovor na kvarove, bolju koordinaciju timova za održavanje i optimizaciju resursa.

Ključne reči: Elektroenergetski sistemi, kvarovi, remont

Abstract – This paper investigates the application of a digital platform for failure management and repair planning in power systems in order to improve them. An analysis of the application's key benefits was performed, including faster failure response, better coordination of maintenance teams, and resource optimization.

Keywords: Power systems, failures, repairs

1. UVOD

Električna energija je jedna od najvažnijih komponenti održivog društva i od suštinskog je značaja za rad domova, industrije i komercijalnih preduzeća. Upotreba električne energije i njena efikasna distribucija imaju ključnu ulogu u obezbjeđivanju stabilnosti i produktivnosti kompanije i društva uopšte. Elektroenergetski sistemi su kompleksni inženjerski entiteti koji obezbjeđuju neprekidno snabdijevanje različitih sektora električnom energijom, od domaćinstava i industrije do komercijalnih postrojenja.

Međutim, u elektroenergetskim mrežama mogu se pojaviti različiti kvarovi i problemi koji mogu uzrokovati prekid u isporuci električne energije, gubitke i ometanja svakodnevnih aktivnosti.

Kada je u pitanju efikasno upravljanje i rad elektroenergetskih sistema, imperativ je upotreba odgovarajućih komponenti i implementacija tehnoloških rješenja. Ovaj rad se fokusira na razvoj i implementaciju digitalne aplikacije koja olakšava proces prijavljivanja kvarova i planiranja remonta u elektroenergetskim sistemima. Aplikacija ima potencijal da unaprijedi efikasnost detekcije kvarova, brze intervencije, te planiranja preventivnih radova, što će rezultirati povećanjem pouzdanosti i smanjenjem prekida u snabdijevanju električnom energijom.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Bošković, vanr. prof.

2. ELEKTROENERGETSKI SISTEMI**2.1. Struktura elektroenergetskog sistema**

Elektroenergetski sistem (EES) je tehničko-ekonomski sistem u okviru koga se izvode sve elektroenergetske transformacije iz ili u električnu energiju, unutrašnje transformacije električne energije, njen prenos, distribucija i neposredna potrošnja [1].

Osnovni zadaci koji se postavljaju elektroenergetskim sistemima jesu da potrošačima obezbijede dovoljno snabdijevanje, tj. potrebnu snagu i dovoljnu količinu električne energije, i da snabdijevanje potrošača bude sigurno, kvalitetno i ekonomično, a sve to uz racionalnu upotrebu električne energije i minimalne troškove.

Kao složeni dinamički sistem, EES se može razdvojiti na četiri funkcionalno nezavisna, ali međusobno spregnuta i veoma zavisna podsistema [2]. Povezanost tih podsistema određena je smjerom toka električne energije (slika 1). Ti funkcionalno odvojeni podsistemi jesu:

- podsistem proizvodnje – u kojem se u elektranama proizvodi električna energija,
- podsistem prenosa – obuhvata prenosnu mrežu i služi da prenese električnu energiju od elektrana do distributivne mreže,
- podsistem distribucije – obuhvata distributivne mreže i distributivne transformatore u kojem se električna energija dobijena iz prenosne mreže distribuira do potrošača,
- podsistem potrošnje – u kojem se troši električna energija.



Slika 1. Prikaz toka energije u EES

2.2. Kvarovi u elektroenergetskim sistemima

EES su podložni različitim vrstama kvarova koji mogu imati ozbiljne posljedice kako po društvo, tako i po privredu i bezbjednost. Razumijevanje prirode, uzroka i posljedica kvarova u EES je od suštinskog značaja za efikasno upravljanje ovim sistemima.

Kvarove u EES možemo svrstati u nekoliko grupa:

- Električni kvarovi – kratki spojevi, otvorena kola, preopterećenja, naponske fluktuacije i sl.
- Mehanički kvarovi – povezani su sa mehaničkim komponentama EES. Ovi kvarovi često mogu biti posljedica starenja i oštećenja opreme.
- Prirodne katastrofe – oluje, poplave, zemljotresi, požari i sl.
- Ljudski faktori – ljudske greške, nepravilno rukovanje opremom, loše održavanje i sabotaze.

Posljedice navedenih kvarova u EES mogu biti ozbiljne i imati širok spektar negativnih efekata na rad elektroenergetskog sistema. Neke od posljedica su prekid u snabdijevanju, bezbjednosni rizici, gubici i troškovi, poremećaji u električnim mrežama, itd.

Prevenција i upravljanje kvarovima su od ključnog značaja kada je u pitanju očuvanje pouzdanosti i efikasnosti EES. Ovo uključuje: redovno održavanje, praćenje i dijagnostiku, kao i brzu intervenciju.

2.3. Izrada plana remonta i održavanja termičkih jedinica

Remonti i održavanje objekata EES-a spadaju među najvažnije aktivnosti u eksploataciji. Posebno su osjetljivi problemi održavanja proizvodnih objekata jer se, iz ekonomskih razloga, savremeni EES-i razvijaju tako da rade sa što je moguće manjom rezervom.

Plan remonta i održavanja termičkih jedinica sastoji se u određivanju perioda i redoslijeda remonata pojedinih agregata, uz minimalne pogonske troškove i uz ograničenje da potrebe svih potrošača u sistemu budu zadovoljene [3]. Heuristička procedura za izradu optimalnog plana remonata sastoji se u računarskoj simulaciji rada sistema za svaku sedmicu ukupnog dozvoljenog perioda održavanja, pod pretpostavkom da su određene jedinice u remontu, a da su sve ostale jedinice raspoložive sa ukupnom snagom jednakom zbiru svojih nominalnih snaga, pomnoženim sa očekivanom raspoloživošću.

Planiranje remonta u EES je kompleksan proces koji zahtijeva pažljivo razmatranje više faktora kako bi se osigurao optimalan i efikasan postupak održavanja. Ključni elementi u procesu planiranja remonta uključuju:

- Pristup rizicima i problemima – identifikacija potencijalnih rizika i određivanje prioriteta remonta različitih komponenta EES.
- Troškovi i budžetiranje – utvrđivanje finansijskih resursa koji su neophodni za sprovođenje remonta i pravilno budžetiranje kako bi se obezbjedila sredstva za planirane aktivnosti.
- Vrijeme i raspoloživost – definisanje perioda za vrijeme kojeg je moguće izvršiti remont komponente EES i obezbjeđivanje radne snage i materijala koji su potrebni tokom tog vremenskog perioda.

- Ograničenja i norme – poštovanje relevantnih industrijskih standarda i zakonskih propisa tokom procesa remonta, uključujući sigurnosne norme i zahtjeve za zaštitu životne sredine.
- Kontinuirano praćenje i ažuriranje – proces planiranja remonta nije statičan i zahtijeva kontinuirano praćenje i ažuriranje kako bi se reagovalo na promjene i nepredviđene okolnosti.

Nakon planiranja remonta slijedi faza implementacije. Ova faza podrazumijeva precizno izvršenje planova remonta uz striktno poštovanje svih sigurnosnih i tehničkih procedura.

Kada je remont izvršen, pristupa se detaljnoj provjeri i testiranju opreme kako bi se u potpunosti osiguralo njeno ispravno funkcionisanje. Rezultati remonta uključuju poboljšane performanse, povećanu pouzdanost sistema i smanjenje rizika od kvarova.

3. METODOLOGIJA

Upravljanje kvarovima i planiranje radova (remonta) u elektroenergetskim sistemima predstavljaju ključne komponente kada je u pitanju stabilnost, pouzdanost i efikasnost elektroenergetskih mreža.

Razvoj metodologije za efikasno rješavanje izazova koje ove komponente donose zahtijeva pažljivo planiranje, organizaciju i koordinaciju različitih aktivnosti. Metodologija o kojoj će biti riječi u nastavku pruža detaljan uvid u korake i procese koji su od ključnog značaja za uspješno upravljanje kvarovima i cjelokupno održavanje EES-a.

Na slici 2 je prikazan dijagram slučajeva korišćenja koji ilustruje različite slučajeve korišćenja aplikacije za upravljanje kvarovima. Ovaj dijagram pomaže u razumijevanju ključnih interakcija između korisnika i sistema, pružajući osnovu za dalje razumijevanje arhitekture i implementacije aplikacije.

3.1. Prijavljivanje kvarova

Prijavljivanje kvarova predstavlja prvi korak u procesu efikasnog upravljanja EES. Korisnici, poput domaćinstava ili preduzeća, koriste aplikaciju za prijavljivanje kvarova. Kvarovi se prijavljuju sa opisom problema koji se desio, kao i preciznom lokacijom na kojoj se kvar nalazi. Ovaj korak omogućava korisnicima da aktivno učestvuju u identifikaciji problema i komunikaciji u elektroenergetskim mrežama, pružajući osnovu za brz i precizan odgovor na potencijalne prekide ili kvarove.

3.2. Analiza i klasifikacija kvarova

Nakon izvršenog prijavljivanja kvara, tim koji je zadužen za praćenje i upravljanje kvarovima (administratorski tim) analizira svaku prijavu. Klasifikacija kvarova se vrši prema različitim kriterijumima, u koje spadaju ozbiljnost problema, hitnost intervencije, kao i potencijalni uticaj na elektroenergetski sistem. Ova analiza je od velikog značaja za naredni korak, odnosno za prioritizaciju kvarova.

3.3. Prioritetizacija kvarova

Na osnovu izvršene analize i klasifikacije kvarova, sledeći korak u metodologiji je njihova prioritizacija. Ozbiljni kvarovi koji za posledicu mogu imati ugrožavanje bezbjednosti korisnika ili narušavanje snabdijevanja električnom energijom se stavljaju na listu kvarova sa najvišim stepenom

prioriteta. Proces prioritizacije nije statičan. U zavisnosti od razvoja situacije i potencijalnog prijavljivanja novih kvarova, može doći do izmjene u prioritetima. Prioritizacija u velikoj mjeri pomaže efikasnijem rasporedu resursa i bržem odgovoru na hitne situacije.

3.4. Planiranje remonta

U cilju efikasnog upravljanja kvarovima, razvija se plan remonta. Planiranje remonta uključuje određivanje resursa koji su neophodni za popravku kvara, kao i utvrđivanje optimalnog vremenskog okvira za intervenciju koju je potrebno izvršiti.

Nakon kreiranja plana remonta, administrator sistema zakazuje popravku. Kada je popravka kvara zakazana, korisnicima se šalje obavještenje o datumu i lokaciji popravke.

3.5. Komunikacija i koordinacija

Kontinuirana komunikacija i koordinacija između različitih timova i aktera su jedan od ključnih elemenata metodologije. Ovaj korak metodologije omogućava elektrodistributivnim organizacijama da obezbijede pravilnu i brzu razmjenu informacija unutar sistema. Korisnicima se pružaju informacije i uvid u status prijavljenih kvarova, dok se timovima na terenu pruža detaljan raspored remonta. Ovo znatno utiče na poboljšanje efikasnosti upravljanja resursima i smanjuje vrijeme potrebno za popravke.

3.6. Praćenje i evaluacija

Nakon izvršenog remonta, vrši se praćenje i evaluacija kako bi se utvrdilo da li su popravke izvedene u skladu sa planom i da li su prijavljeni kvarovi otklonjeni. Ovaj segment metodologije omogućava organizacijama da aktivno prate performanse sistema, identifikuju potencijalne probleme i kontinuirano unapređuju svoje procese. Prvi korak u praćenju i evaluaciji je identifikacija i usvajanje ključnih pokazatelja performansi (KPI) za EES. Ovi KPI-ji obuhvataju različite aspekte, kao što su pouzdanost napajanja, efikasnost mreže, iskorišćenost resursa i sigurnost. Praćenje i evaluacija čine proces upravljanja EES-om dinamičnim i adaptivnim.

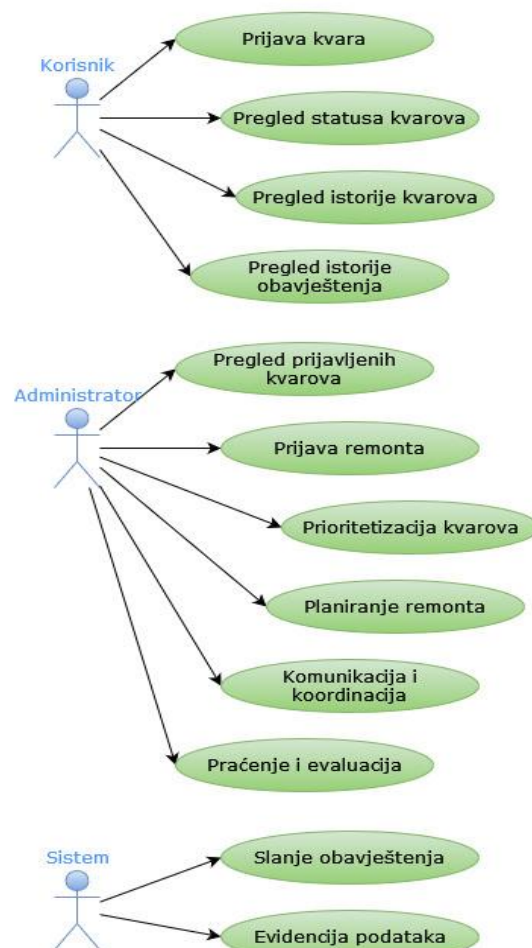
3.7. Unapređenje procesa

Ovaj segment metodologije fokusira se na analizu postojećih procesa i procedura kako bi se identifikovali prostori za optimizaciju i unapređenje. Prvi korak u unapređenju procesa je identifikacija svih ključnih procesa koji su od suštinskog značaja za funkcionisanje EES-a. Ovo uključuje sve korake u procesima održavanja, intervencije u slučaju kvarova, planiranje remonta i komunikacije.

Nakon identifikacije slijedi detaljna analiza svakog procesa u kojoj se identifikuju sve potencijalne slabosti i prostori za poboljšanje.

Na osnovu izvršene analize, definišu se ciljevi za unapređenje i pristupa se dizajniranju novih procesa ili optimizaciji postojećih.

Unapređenje procesa je iterativni proces. Ovaj korak metodologije čini EES sposobnim da se u kontinuitetu prilagođava novim izazovima, postiže viši nivo efikasnosti i da ispunjava očekivanja korisnika.



Slika 2. Slučajevi korišćenja

4. REZULTATI I DISKUSIJA

Poboljšanje upravljanja EES-om se ogleda u efikasnosti i pouzdanosti aplikacije za prijavu kvarova i planiranje remonta. Efikasnost aplikacije podrazumijeva njenu sposobnost obezbjeđivanja brze reakcije na prijave kvarova, kao i automatsko obavješćavanje korisnika i administratora sistema. To značajno smanjuje vrijeme potrebno za identifikaciju i rješavanje problema.

Pouzdanost se odnosi na tačnost informacija koje se pružaju korisnicima. Aplikacija omogućava korisnicima da prate status svojih prijavi kvarova, što doprinosi transparentnosti i povjerenju korisnika u aplikaciju. Intuitivnost i jednostavnost aplikacije za korišćenje omogućava korisnicima da brzo i lako prijave kvarove.

Smanjenje vremena potrebnog za reakciju i popravku kvara je od velikog značaja za održavanje pouzdanosti EES i obezbjeđivanje neprekidnog snabdijevanja korisnika električnom energijom. Uvođenje aplikacije za prijavu kvarova i planiranje remonta značajno doprinosi ostvarenju ovih ciljeva na više načina, u koje spadaju: brža detekcija kvarova u odnosu na dosadašnju, automatizovana obavještenja, bolje planiranje resursa i njihovo efikasnije korišćenje, prioritizacija kvarova, olakšana koordinacija timova, praćenje i evaluacija performansi.

Planiranje remonta u EES predstavlja važnu kariku u održavanju EES, smanjenju vremena prekida u snabdijevanju električnom energijom i produženju životnog vijeka opreme. Implementacija aplikacije može značajno unaprijediti ovaj ključni aspekt upravljanja EES. Aplikacija omogućava sakupljanje podataka o stanju elektroenergetske infrastrukture i procjenu potrebnih resursa i vremenskih okvira, što doprinosi efikasnosti planiranja remonta. Olakšava komunikaciju između terenskih ekipa i centralne administracije. Omogućava planiranje remonta unaprijed, kao i njihov kalendarski pregled. Osim toga, aplikacija može doprinijeti efikasnijem korišćenju resursa i time uticati na smanjenje troškova održavanja i budžet elektroenergetskih preduzeća.

Osim što aplikacija za prijavu kvarova može poboljšati sigurnost i pouzdanost EES, ona značajno može doprinijeti zadovoljstvu korisnika i administracije. Korisnici će cijeniti bržu reakciju na prijavljene kvarove, kao i uvid u njihov status. Korisnici mogu pristupiti historiji svojih prijava i popravki, kao i historiji svih obavještenja, što im omogućava bolje razumijevanje rješavanja njihovih problema tokom vremena. Precizno planiranje i prioritizacija remonta znači da će nepotrebne smetnje korisnicima biti izbjegnute, što doprinosi njihovom zadovoljstvu i minimalnom uticaju na svakodnevne aktivnosti.

Administracija ima bolju kontrolu nad procesom prijave kvarova i planiranja remonta, čime je omogućena efikasna alokacija resursa i bolja koordinacija timova. Preciznim planiranjem remonta, administracija može bolje predvidjeti troškove i izbjeći one koji su nepotrebni. Administrativni procesi, kao što su praćenje i izvještavanje, su poboljšani i unaprijeđeni, čime se štedi kako vrijeme, tako i resursi. Na kraju, zadovoljni korisnici često znače manji pritisak na administraciju, što doprinosi boljem radnom okruženju i opštem zadovoljstvu zaposlenih.

Buduće mogućnosti unapređenja aplikacije za prijavu kvarova i planiranje remonta u EES mogu donijeti dodatne koristi i unapređenja. Jedna od ključnih sekcija koje se mogu istraživati i razvijati u cilju unapređenja aplikacije je vještačka inteligencija i napredna analitika. Njihova implementacija može omogućiti prediktivno održavanje i identifikaciju potencijalnih kvarova prije nego što se oni dese. Vještačka inteligencija može analizirati velike količine podataka kako bi identifikovala obrasce i rizike, čime se smanjuje potreba za reaktivnim intervencijama. Upotreba *Internet of Things* (IoT) senzora za praćenje stanja opreme može automatski generisati prijave kvarova kada senzori otkriju odstupanja ili probleme u elektroenergetskoj mreži, što omogućava još bržu reakciju na probleme. Takođe, razvoj mobilne aplikacije bi omogućio terenskim ekipama lakši pristup informacijama o kvarovima i evidenciji popravki. *Blockchain* tehnologija je još jedan od načina unapređenja aplikacije koji može osigurati sigurnost i transparentnost u procesima prijave kvarova i evidencije popravki. Ova tehnologija ima posebnu važnost u situacijama koje zahtijevaju konstantnu verifikaciju i evidenciju.

Buduće mogućnosti unapređenja ove aplikacije su brojne, a njihova implementacija će doprinijeti efikasnijem i pouzdanijem elektroenergetskom sistemu, čime će se obezbjediti bolja usluga korisnicima i optimizacija resursa.

5. ZAKLJUČAK

U ovom radu istražen je razvoj i primjena digitalne aplikacije za prijavu kvarova i planiranje remonta u EES. Cilj ovog istraživanja bio je unapređenje efikasnosti i pouzdanosti elektroenergetskih mreža, što je od suštinskog značaja za obezbjeđivanje stabilnog snabdijevanja električnom energijom.

Kroz analizu problema u EES, razvijena je metodologija koja omogućava efikasno upravljanje kvarovima i održavanjem. Ova metodologija uključuje korake prijavljivanja kvarova, analize i klasifikacije kvarova, prioritizacije, planiranja remonta, komunikacije i koordinacije, praćenja i evaluacije, kao i unapređenja procesa. Implementacija metodologije u korelaciji sa odgovarajućim tehnologijama i resursima može dati značajan doprinos stabilnosti i pouzdanosti elektroenergetskih mreža i EES u cjelini.

Takođe, rad takođe ukazuje na buduće mogućnosti unapređenja, kao što su integracija vještačke inteligencije i bolje analitike, zatim IoT senzori i *blockchain* tehnologija. Navedene tehnologije mogu dodatno unaprijediti efikasnost i sigurnost EES.

Ovaj rad ukazuje na važnost uloge digitalizacije u modernizaciji EES. Njegova primjena pored toga što doprinosi boljem praćenju i upravljanju kvarovima i remontima, takođe igra važnu ulogu u većem zadovoljstvu korisnika i administracije.

6. LITERATURA

- [1] Sreten Škuletić, Marina Braletić, "Uvod u energetiku", Podgorica, 2022.
- [2] Vladimir C. Strezoski, "Osnovi elektroenergetike", Novi Sad, 2014.
- [3] Miloš Milanković, Dragoslav Perić, Ivana Vlajić-Naumovska, "Osnovi elektroenergetike", Beograd, 2016.

Kratka biografija:



Željka Matović rođena je u Trebinju 1997. godine. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – *Primenjeno softversko inženjerstvo* odbranila je 2023. godine.

kontakt: matoviczeljkaa@gmail.com

DOŽIVLJAJ IGRE KROZ INTERAKCIJU I KONTROLNE MEHANIZME

EXPERIENCING THE GAME THROUGH INTERACTION AND CONTROL MECHANISMS

Nikola Dodig, *Fakultet tehničkih nauka, Novi Sad*

Oblast – RAČUNARSKA GRAFIKA

Kratak sadržaj – Analiza par komercijalnih video igara i identifikacija mehanizama koji najviše utiču na doživljaj igre i na osnovu toga razvijanje igre koja ilustruje zaključke nalize.

Ključne reči: *Gejming, Interfejs, Mehanizmi, doživljaj igre.*

Abstract – *Analysis of a couple of commercial video games and identification of the mechanisms that have the most impact on the game experience and, based on that, the development of a game that illustrates the conclusions of the analysis.*

Keywords: *Gaming, Interface, Mechanisms, Game experience.*

1. UVOD

Video igre su nastale u suštini krajnje logično, mada pomalo i (ne)ozbiljno. Prva video igra, koja tehnički i nije video igra, nastala je iz potrebe za simulacijom i predviđanjem ozbiljnih događaja, ali bez stvarnih posledica. U pitanju su bila predviđanja putanji raketa i pokušaj njihovog obaranja. Rakete su se pojavljivale na radaru, i korisnik bi pokušao da ih obori. U pitanju je video igra Cathod ray amusement device. Igra se sastojala od računara koji je težio oko tonu i displeja koji se sastojao od 12 lampica. Pomeranjem ručke, korisnik je trebao da predvidi i da pomera svetlo i pokuša da pogodi drugo svetlo, dok mu je drugo svetlo "bežalo" [2].

Za potrebe ovog rada kreirana je RPG video igra pod imenom "BookWorm", žanr igre u kojoj igrač kontroliše akcije jednog ili više karaktera uronjenih u dobro definisan svet, obično uključujući neki oblik razvoja likova tokom vremena [1].

Takođe za potrebe zadatka bilo je potrebno ispitati na koji način kontrolni mehanizmi i interakcija sa igrom utiču na sam doživljaj igre. Kontrolni mehanizmi predstavljaju sve moguće komande kroz koje korisnik može da utiče na promene u igri. Postoji više vrsta kontrolnih mehanizama, u zavisnosti od platforme za koju je igrice namenjena. Za potrebe ovog rada predviđeni su kontrolni mehanizmi miša i tastature.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivetić, red. prof.

2. GEJMPLEJ

Kroz različite nivoe igrač treba da savlada izazove na drugačije načine i tako pređe sve nivoe. Nije tačno određen sled događaja koje igrač mora da ispuni kako bi došao do kraja igre već je sva sloboda za rešavanje problema prepuštena njemu.

U poslednjem nivou zadatak je ukombinovati najbolje iz dva sveta, sveta u kojem igrač uči i koristi magije i sveta u kojem koristi oružje. Kroz različite osobine karaktera izbegava se monotonija. Ovo uključuje novu dimenziju uživanja ali opet ne zahteva ponovno učenje komandi jer one ostaju iste kroz sve nivoe.

Gejmpler je skup aktivnosti koje mogu izvoditi igrači tokom spontanog i neusmerenog inskustva igre, kao i ostali subjekti koji pripadaju vrtuelnom svetu, kao odgovor na akcije igrača i/ili kao autonomni pravac delovanja koji doprinose živosti vrtuelnog sveta [3].

3. MEHANIZMI IGRE

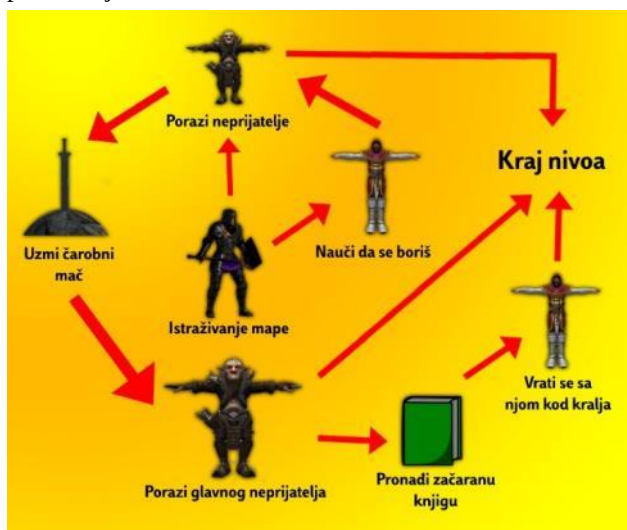
Interaktivnost i aktivnost su dva ključna koncepta koja leže u osnovi prethodne definicije gejmplera. Ali, kako igrači komuniciraju sa igrom? Da bi uopšte preduzeli bilo koji aktivnost, igrači moraju da komuniciraju sa igračkama. Svaka spontana i neusmerena aktivnost podrazumeva reakciju sa konkretnim apostraktnim objektima. Takvi predmeti se takođe obično nazivaju igračkama, a njihova manipulacija zahteva nivo stručnosti koji se postiže kroz proces učenja. Polazna tačka je istraživanje karakteristika igračaka koji se koriste u igri samo kada igrač smatra da je razumeo njihova svojstva. Upotreba igračaka i njihovih odnosa regulisani su pravilima, koja organizuju skup spontanog i neusmerenog aktivnosti i pretvaraju ga u koherentnu igru. Neke od ovih igračaka nativaju se „Mehanizmi igre“ [3].

Shvativši važnost mehanizama igre i kako se nositi sa njima kako bi se održalo interesovanje igrača, još jedno važno pitanje je da li se svi mehanizmi igre tretiraju podjednako? Odgovor je ne. Kada se analizira igranje igre, igrači implicitno očekuju da vide neku vrstu hijerarhije između mehanike igre zasnovanu na njihovom značaju. Važno je ovo razumeti da bi se znalo gde staviti naglasak, čime se pravilno projektuje igranje i mehanizmi igre [3].

Interakcije igrača sa vrtuelnim svetom posredovano je najvažnijim mehanizmom bilo koje igrice – igračkim žetonom. Takav žeton može biti pravi avatar, taji da predstavlja otelotvorenje igrača u vrtuelnom svetu(kao

što se dešava u igrama Quake i Tomb Raider), ili može biti nevidljiva ruka(kao što se to dešava u igrama kao što je Tetris ili Sims). U oba slučaja, igrački žeton prima ulazni signal od igrača i posreduje u svim pokušajima interakcije sa ostatkom virtualnog sveta. Pravila koja regulišu žeton igrača određuju šta igrač može da uradi sa njim i sa ostatkom virtuelnog sveta kroz njega. Drugim rečima igrački žeton je ključ za ono što se u žargonu dizajna igara naziva “Core gejmplej” i stoga je najviše važan od svih mehanizama igre [3].

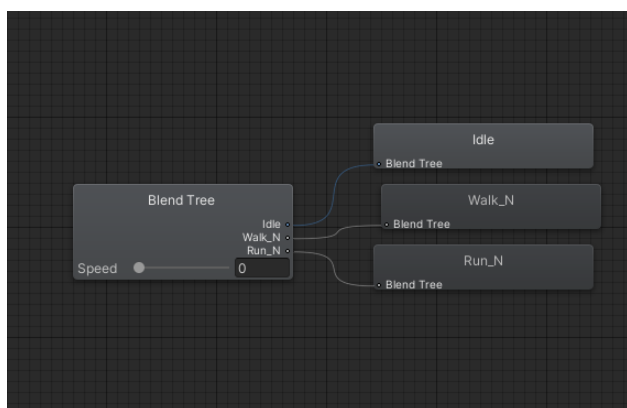
Prikaz core gejmpleja drugog nivoa igre “BookWorm” prikazan je na slici 1.



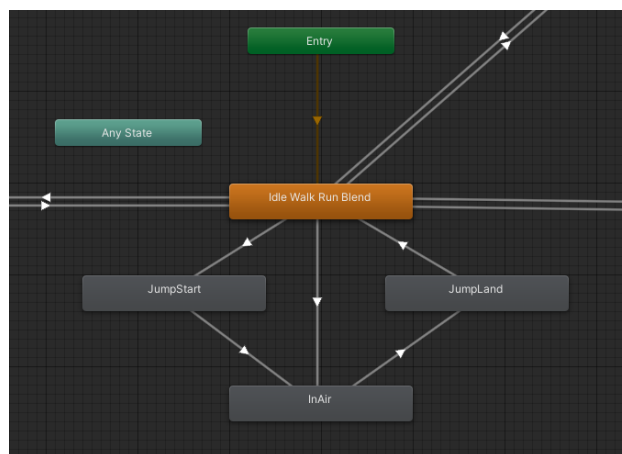
Slika 1. Dijagram core gejmpleja drugog nivoa

Uvek prateći pristup usređen na igrača, možemo definisati core gejmplej kao skup aktivnosti koje će igrač najčešće preduzimati tokom doživljaja igranja igre i koji su neophodni za pobjedu u igri. Mehanička igre koja omogućava izvođenje core aktivnosti igranja nativaju se “core mehanizmi” i shodno tome, najvažniji su u igri pošto će igrač trebati da se nosi sa njima tokom celog doživljaja u igranju [3].

Core mehanizmi razlikuju se u odnosu na nivo koji je trenutno aktivan. U prvom nivou dostupni su mehanizmi za hodanje, trčanje, skakanje i mirovanje. Ovi mehanizmi takođe se nalaze u svim ostalim nivoima i istovetni su.. Mehanizmi kretanja, trčanja i stojanja u mestu blendovani su u jedan “Blend Tree”, a skakanje izvedeno je u osnovnom “Base layer” sloju, slika 2. i 3. respektivno.



Slika 2. Mehanizam za kretanje prvi sloj



Slika 3. Mehanizam za kretanje drugi sloj

Promena oružja sa vratrene moći na mač i obrnuto, vrši se tako što korisnik pritisne na jedno od dva dugmeta 1 ili 2 na tastaturi, u zavisnosti koje oružje želi da koristi, in a taj način izabere željeno oružje.

Ovo je izvedeno pomoću jedne animacije za odlaganje mača na leđa. Ako igrač trenutno koristi vatrenu moć i pritisne na taster 2, da koristi mač, vatrena moć nestaje iz ruke (postaje onesposobljena) pokreće se animacija i avatar pometa ruku u pravcu leđa na mesto gde mu se nalazi mač.

Animacija se završava kada avatar dotakne mač. Tada se mač koji je na leđima takođe onesposobljava, a mač koji mu je u ruci, ali je onesposobljen, se osposobljava i time prikazuje.

Nakon toga izvršava se ista animacija ali u suprotnom smeru i na taj način postiže se doživljaj izvlačenja mača slika 4.

Na isti način implementirana je povratna promena sa mača na moć.

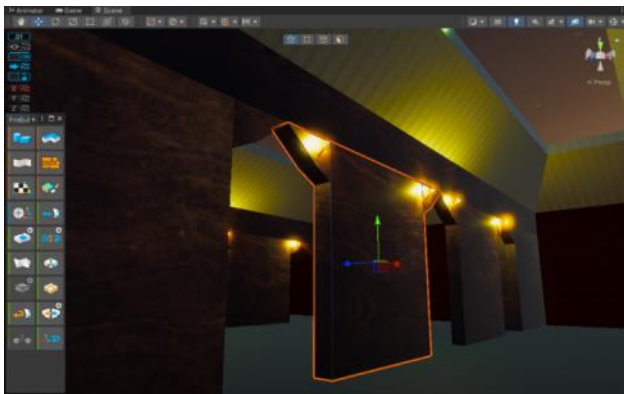


Slika 4. Vizuelni prikaz dešavanja prilikom promene trenutnog oružja sa moći na mač

4. DIZAJN NIVOVA

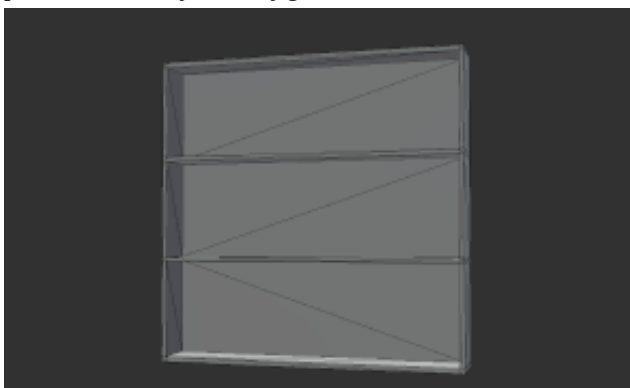
Prvi nivo ima za cilj da korisnika zbliži sa osnovnim karakterom Džonijem koji se našao u zalčaranoj biblioteci. Iz nje ne može izaći ali može čitati njige koje se nalaze u njoj. Kada priđe polici sa knjigama, pojavljuje mu se opcija za čitanje knjiga. Pritiskom na dugme “E” prelazi se u naredni nivo.

Kako se sva radnja u prvom nivou nalazi u biblioteci, potrebno je napraviti istu. Za to je potreban Unity alat ProBuilder. Cilj je napraviti što realističniji model biblioteke koja bi odgovarala tematici igre. Nakon toga potrebno je dodati materijale ta pod, zidove, krov, staklo i vrhu slika 5.

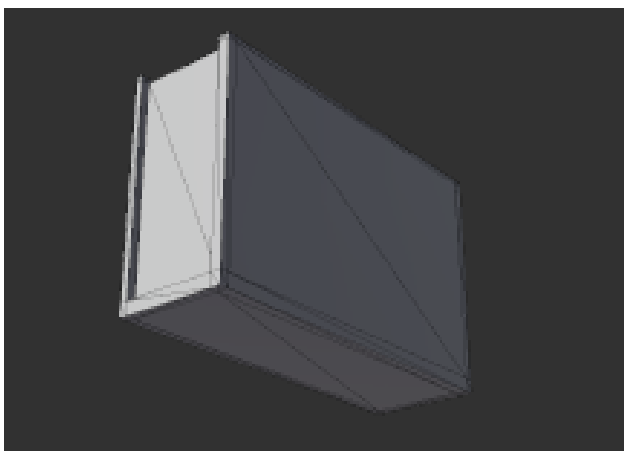


Slika 5. *Ptavljenje biblioteke pomoću ProBuildera*

Nakon što je završena biblioteka, potrebno ju je ispuniti knjigama i policama. Za te potrebe napravljena je jedna polica, slika 6. i jedna knjiga, slika 7.



Slika 6. *Polica napravljena pomoću ProBuildera*

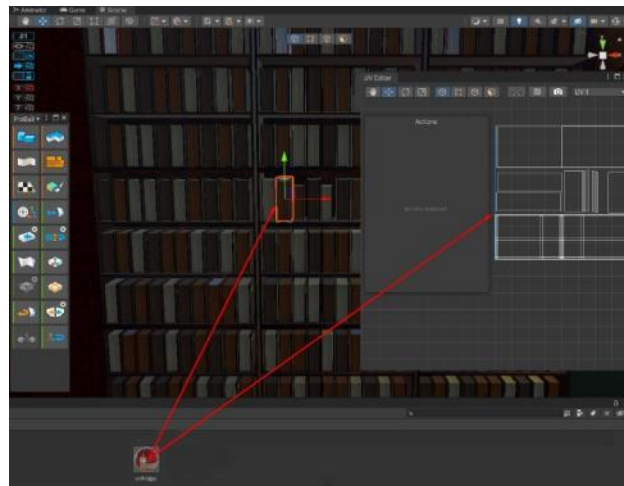


Slika 7. *Knjiga napravljena pomoću ProBuildera*

Nakon toga od knjige je napravljen prefabrikovan model koji se koristio za pravljenje drugih prefabrikovanih knjiga sa drugačijim dimenzijama materijalima i ostalim osobinama. Za svaku knjigu potrebno je napraviti posebnu UV mapu i dodati je kao poseban materijal, slike 8 i 9.

5. INTERFEJS

Kao deo samog izgleda igre, takođe je bitno i da se korisniku omogući jasan pregled informacija koji su relevantni za njega u toku igranja igre kao i mogućnost da se lako navigira kroz menije. Kreiranjem korisničkog interfejsa koji ovo omogućava znatno povećava kvalitet same igre.



Slika 8. *Izgled Pozicionirane UV mape knjige*



Slika 9. *Primer materijala različitih knjiga*

Ono što je potrebno da korisnički interfejs ispunjava u ovom slučaju jeste da poseduje glavni Meni sa svim dostupnim opcijama, HUD koji služi za prikaz informacija o životnim poenima junaka i neprijatelja, snagu odnosno manu junaka i ubliženi deo mape na kojoj se junak trenutno nalazi, meni za pauzu, meni za inventar i meni sa mapom.

Meni za mapu koristi skriptu koja postavlja sve potrebne objekte koji se nalaze na mapi tačno na mestima na kojima se nalaze naspram stvarnog sveta igre, slika 10.



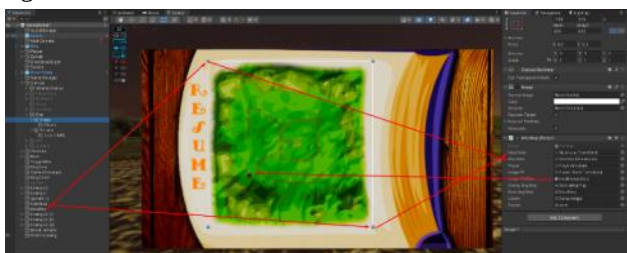
Slika 10. *Meni za mapu*

Svaki put kada korisnik udje u meni za mapu izvršava se rasporedjivanje ikonica predmeta čije informacije treba da se nadju na mapi, slika 11.



Slika 11. Meni za mapu nakon pomeranja po mapi

Mapa je implementirana tako što je u glavni Canvas dodat meni Map sa osnovnom slikom stola i njige, sa podobjektima tipa dete koji se sastoje od dugmeta za navigaciju unazad i slike mape čiji podobjekti su prefabrikovani modeli koji se učitavaju i dodaju dinamički za svaki potrebni objekat u igri Slika 68 i Listing 1. NulaNula i MaxMax predstavljaju objekte sa X i Y koordinatom početne (NulaNula) i krajnje (MaxMax) tačke terena, tj. mape na kojoj se igrač nalazi. Pošto je mapa rotirana za 90 stepeni suprotno od pravca kazaljke na satu NulaNula predstavlja tačku u donjem desnom uglu dok MaxMax predstavlja tačku u gornjem levom uglu, slika 12.



Slika 12. Implementacija menija za mapu

Ovo se može primeniti na bilo koju mapu.

6. ZAKLJUČAK

U današnje vreme sve je više video igara, na televizijama je sve više kanala koji u sebi imaju reč E-Sports što predstavlja vrstu kanala za elektronske sportove tj. za igrice. Sve većom dostupnošću novih tehnologija kao što su pametni telefoni, tableti, laptopovi, računari pa čak i konzole sve je veći broj ljudi koji igraju video igre.

Imajući na umu gorenavedene informacije kao i analizu tržišta iz drugog poglavlja projekta možemo zaključiti da u video igrama leži veliko tržište puno potrošača. Kako Unity pogon ima mogućnost za izgradnju igrica za bilo koju gorenavedenu platformu uz neke promene vezane za mehaniku i grafiku to govori da ovaj projekat može biti proširen za celo gaming tržište. Mehanika na svim uređajima treba da bude što istovetnija kako bi osećaj pri igranju na bilo kojoj platformi bio isti, a opet poseban na način na koji ga platforma pruža.

Izrada projekta u delovima, od priče i skica do kodiranja, snimanja i obrade zvuka, importovanja animacija i karaktera, pravljenja terena i objekata zahteva širok spektar znanja iz oblasti koje to pokrivaju. Kako znanje iz tih oblasti nije bilo kompletno, bilo je potrebno učiti i ovladati novim tehnikama iz tih oblasti. Kao rezultat dobija se ne samo gotov rad već i novo znanje koje će služiti da se rad na narednim projektima lakše razume, postane brži i efikasniji kao i da sam rad bude kvalitetniji.

7. LITERATURA

[1] Wikipedia, "Role-playing video game", https://en.wikipedia.org/wiki/Role-playing_video_game, Članak o RPG video igrama. – datum pristupa 14.10.2023

[2] Raf, "Istorija video igara", <https://raf.edu.rs/citaliste/clanci/svastara/istorija-video-igara/>, Članak o istoriji video igara. – datum pristupa 14.10.2023

[3] OECD, "GAMEPLAY AND GAME MECHANICS DESIGN: A KEY TO QUALITY IN VIDEOGAMES", <https://www.oecd.org/education/ceri/39414829.pdf>, Rad na temu ključnosti mehanizama igre za kvalitet video igre. – datum pristupa 16.10.2023

Kratka biografija:

Nikola Dodig rođen je u Vršcu 1998. god. Master rad na Fakultetu tehničkih nauka iz oblasti Računarske grafike odbranio je 2023.god.
kontakt: dodimircewow@gmail.com

RAZVOJ RAČUNARSKE IGRE SA VOĆNIM PROTAGONISTOM DEVELOPMENT OF A COMPUTER GAME WITH A FRUIT PROTAGONIST

Isidora Stanić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – Ovaj članak opisuje razvoj računarske igre sa voćnim protagonistom, „Go Bananas“. Zadatak projekta bio je razviti računarsku igru od samog početka, od ideje do krajnje realizacije računarske igre i trejlera. Za realizaciju projekta korišćeni su Unity, Blender, Paint.NET i Clipchamp. Ciljevi igre su spasiti prijatelje protagoniste i vratiti im njihove kapice. Kao rezultat, početni deo igre je uspešno realizovan, dok je ostatak igre osmišljen i pripremljen za dalju realizaciju.

Ključne reči: „Go Bananas“, 3D Platformska igra, Avantura, Dizajn, Gejملهj, Trejler

Abstract – This article describes the development process of the computer game with a fruit protagonist, „Go Bananas“. The project task was to develop a computer game from scratch, from the initial idea to the final realization and trailer. Unity, Blender, Paint.NET and Clipchamp were used for the project implementation. The objectives of the game are to save the protagonist's friends and return their caps. As a result, the initial part of the game was successfully implemented, while the remainder of the game has been conceptualized and prepared for further development.

Keywords: „Go Bananas“, 3D Platformer, Adventure, Design, Gameplay, Trailer

1. UVOD

Cilj projekta jeste da se razvije računarska igra sa voćnim protagonistom, od početne ideje do konačne realizacije. Pored igre, napravljen je i trejler, koji služi za promociju igre. Za realizaciju projekta korišćeni su Unity, Blender, Paint.NET i Clipchamp.

Nakon što su njegovi prijatelji otišli od strane zlih Snikera, Nana Boj mora da se podvrgne intenzivnom treningu i postane dovoljno snažan da se suoči sa izazovima koji ga čekaju. Na svom putu će se suočiti sa brojnim preprekama, boriti se protiv neprijatelja i otkrivati tajne tropskih ostrva. Da li će upornost i snaga prijateljstva biti dovoljne?

„Go Bananas“ je avanturistička 3D platformska igra, dizajnirana za PC računar. Igra je smeštena na tropskim ostrvima, gde igrači preuzimaju ulogu Nana Boja, mladog protagoniste koji se upušta u avanturu da spasi svoje prijatelje.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivetić, red. prof.

Ciljna grupa za igru su deca i mlađi adolescenti od 8 do 14 godina. Ova starosna grupa je posebno privučena svetlim, jarkim bojama, jednostavnim stilom likova i okruženja, što igru čini vizualno privlačnom i lako razumljivom. Zabavni zvučni efekti i vesela muzika dali bi igri veselo raspoloženje, što bi dodatno privlačilo ovu demografsku grupu. Interaktivnost okruženja, koja omogućava igračima da manipulišu objektima i istražuju svet oko sebe, pruža dodatnu dubinu i složenost igri, dok istovremeno podstiče kreativnost i radoznalost. Priča o avanturi mladog protagoniste, koji se suočava sa svojim najvećim nedostacima, pruža igračima priliku da se poistovete sa likom. Pored zabave, igra pruža igračima i pozitivne poruke o hrabrosti, upornosti i prijateljstvu. Sve ove karakteristike čine igru idealnom za ovu ciljnu grupu.

2. LIKOVI

Jedna od ključnih komponenti koja čini osnovu svake igre su likovi, koji učestvuju u igri. Definisanje likova je od suštinske važnosti jer postavlja ton i kontekst u kojem se igra odvija, obogaćuje igračko iskustvo i doprinosi imerziji.

U svetu igre „Go Bananas“, gde sunce retko izlazi i zalazi, vreme se meri u danima, koji su mnogo duži nego u stvarnom svetu. Starost likova se određuje u danima. Prosečan životni vek stanovnika Bananosa je 10 dana.

2.1. Protagonista

Nana Boj (engl. *Nana Boy*) je dečak sa ostrva Bananos, iz istoimenog sela. Tek mu je 2 dana, i pred njim je ceo život od bar još 8 dana. Budući da su svi njegovi drugari za dan zreliji od njega, on je najsporiji kada se trkaju. Njegove sposobnosti su udarac nogom i gađanje kokosima. Kako igra napreduje, Nana Boj će postajati sve brži i snažniji.

Nana Boj izgleda kao banana, ali ima ruke i noge (Slika 1). Ima dobroćudno i vedro lice. Na glavi nosi kapicu, kao i svi ostali u njegovom selu, koja ga štiti od sunčanice i produžava mu život. Njegovi drugari izgledaju isto kao on, ali su malo viši.

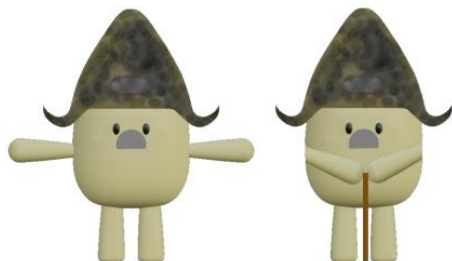


Slika 1. *Nana Boy*, protagonista

2.2. Mentor

Gremps (engl. *Gramps*) je mudar i strog deda od Nana Boja. Star je 8 dana. Pre 4 dana bio je najsposobniji u selu, a sada pokušava da spremi svog unuka za nevolje koje ga čekaju. Uvek ima neki mudri savet da da svom unuku, i ne popušta mu dok trenira.

Izgleda kao banana, ali ima ruke i noge. Ima vedro lice sa sedim brkovima. Na glavi nosi kapicu, koja je posle toliko dana prilično potamnela (Slika 2).



Slika 2. *Gremps, mentor*

2.3. Antagonisti

Snikeri (engl. *Sneakers*) su podanici glavnog antagoniste, kralja Stinka. Kralj im je naredio da kidnapuju svu omladinu Bananosa i dovedu ih u glavnu palatu. Imali su savršen plan, ali im je promakao Nana Boj. Dobili su žestoku grdnju čim je kralj saznao za to. Zavisno od odreda, sposobnosti Snikera variraju od snažnog udarca iz blizine do brzog gađanja sa visine.

Snikeri izgledaju kao gorile. Imaju zlonamerno namršteno lice (Slika 3). Zavisno od odreda, mogu biti iste visine kao Nana Boj ili duplo veći.



Slika 3. *Sniker, antagonista*

Kralj Stink (engl. *King Stink*) je glavni antagonista u igri. On je tiranin koji vlada kraljevstvom Snikeros. Većina Snikera ga bezuslovno sluša jer ga se plaše. Za vreme svoje vladavine prekršio je nekoliko dogovora sa raznim ostrvima, među kojima je i dogovor sa Bananosom o međusobnom nenapadanju. Njegovi napadi se sastoje od gađanja raznim predmetima i jakih udaraca.

Kralj Stink na glavi nosi krunu, a na ramenima svečane epolette (Slika 4), kao simbol svoje moći.



Slika 4. *Kralj Stink, glavni antagonista*

3. PRIČA

Radnja priče počinje na ostrvu Bananos, gde se Nana Boj trka i igra sa svojim prijateljima. Sunčan je dan i Nana Boju baš teško pada da trči, zato kaska za ostalima. Još je on mlad da ih sve prestigne.

Dok se bori za dah u poslednjoj petini staze, gleda kako poslednji prijatelj prolazi cilj. Tek što je trepnuo, video je kako svi njegovi prijatelji upadaju u zamku Snikera. Uхватili su ih u veliku mrežu. Nana Boj se brzo sakrio iza najbližeg drveta da osmotri celu situaciju. Jedan Sniker je skinuo kapicu jednom od njih i bacio je u šumu.

Snužden, Nana Boj odlazi kući, gde ga čeka njegov deda. U očaju mu je ispričao šta se desilo. Bio je siguran da ne može ništa da učini.

Deda ga je uverio u suprotno. Ispričao mu je o svojoj mladosti, kako je i on nekada kaskao za ostalima. Velikim trudom i vežbanjem je osnažio, i postao najsposobniji u selu. Gremps je odlučio da ga trenira. Trčao je, radio je čučnjeve i sklekove, dok konačno nije bio spreman da spasi prijatelje.

Posle napornog treninga, Nana Boj je krenuo na put. Na putu je nailazio na prepreke i izazove koje je morao da premosti. Putovao je od ostrva do ostrva dok nije konačno stigao do ostrva Snikeros.

Na ostrvu Snikeros ga je dočekaio zamak naoružan čuvarima koji su ga napali. Nana Boj se odbranio i krenuo ka kralju, koji je počeo da ga gađa kokosima i udara izbliza. Nana Boj je uspeo da se odbrani i nanese veliku štetu kralju. Pobjeda je bila na dohvat ruke.

Međutim, kralj je pokupio zarobljene prijatelje u mrežu i počeo da beži. Nana Boj ga mora stići da bi ih spasio. Preskače razne prepreke i sakuplja munje da bi dobio ubrzanje. Na kraju ga stiže, zadaje mu poslednji udarac i spašava svoje drugare.

Svi prijatelji su srećni i zahvalni što su konačno na slobodi. Jedan od njih se pita gde su im kapice. Zavisno od broja sakupljenih kapica, kraj će se razlikovati. Ukoliko Nana Boj nije sakupio sve kapice, neće moći da ih vrati, te su mu prijatelji zahvalni, ali tužni jer nemaju svoje kapice. Ukoliko sakupi sve kapice, moći će da ih vrati, i svi će se srećni vratiti kući.

4. GAMEPLAY

Gejmplej ili proces igranja (engl. *gameplay*) je ključni element video igara, koji direktno utiče na igračko iskustvo. Postoje razne definicije gejmplaja, ali kada se gleda iz perspektive igrača, gejmplaj se može definisati kao skup aktivnosti koje igrač može da obavlja tokom ludičkog iskustva, kao i aktivnosti drugih entiteta koji pripadaju virtuelnom svetu, kao odgovor na akcije igrača i/ili kao autonomne radnje koje doprinose živosti virtuelnog sveta [1].

U igri „Go Bananas“, igrači mogu da se kreću kroz 3D prostor. To uključuje hodaње, trčanje i skakanje. Pored kretanja, igrači takođe mogu da napadaju koristeći udarac nogom ili gađanje kokosima. Ove akcije služe za borbu protiv neprijatelja.

Tokom igre, igrači mogu sakupljati različita unapređenja. To uključuje kokose, munje i šlag. Kokosi služe kao municija koju igrači mogu koristiti da gađaju neprijatelje.

Munja povećava brzinu kretanja na određen vremenski period, dok šlag čini lika neuništivim na određeno vreme.

Na svakom nivou, igrači mogu pronaći skrivenu kapicu. Sakupljanje ovih kapica je opciono. Ukoliko igrači sakupe sve kapice, na kraju će ih predati spašenim prijateljima. Ukoliko ih ne sakupe, neće im biti vraćene kapice.

Cilj igre je da se spasi kidnapovani drugari. Da bi stigao do glavnog neprijatelja, igrač mora doći okolnim putem do neprijatelja. Na tom putu nailazi na podanike koje mora da pobeđi kako bi nastavio dalje. Svaki običan nivo se završava kada igrač pobeđi čuvare mosta za sledeći nivo. Kada stigne do kralja, nakon duže borbe on će pokušati da pobjegne, te je uslov pobeđe da ga igrač stigne. Igrač će biti poražen ukoliko mu kralj pobjegne ili ukoliko izgubi sve živote.

5. DIZAJN NIVOA

Svaki nivo u igri predstavlja jedno ostrvo po kojem se igrači kreću. Počevši od Bananosa, igrači kreću na put kroz različite terene i okruženja. Svaki nivo je dizajniran kao ostrvo, sa specifičnim putem koji bi igrači trebalo da prate, ali postoji i prostor za istraživanje i otkrivanje skrivenih puteva i tajni.

Na svakom nivou, igrači se suočavaju sa raznim izazovima i preprekama. To mogu biti neprijatelji koje treba pobediti ili fizičke prepreke koje treba prevazići. Na nekim nivoima će prepreke biti uništene ili neprohodni putevi, koje igrač treba da zaobiđe ili popravi da bi došao do kraja nivoa. Kako igrači napreduju kroz nivoe, težina igre se postepeno povećava. Izazovi postaju teži, a neprijatelji snažniji i češći.

Igrači mogu pronaći različite nagrade i unapređenja. Mogu sakupiti kokose, munje i šlagove. Takođe, na svakom nivou se može pronaći po jedna skrivena kapica koju igrači mogu da sakupe kao dodatni izazov.

5.1 Prvi nivo, Bananos

Prvi nivo igre „Go Bananas“ je dizajniran tako da bude jednostavan, da bi olakšao fazu učenja mehanike (Slika 5). Nana Boj kreće iz svog sela, jednim putem koji postoji. Na raskrsnici puteva nalazi se putokaz koji usmerava igrača napred, da bi stigao do ostrva Snikeros.



Slika 5. Prvi nivo, Bananos

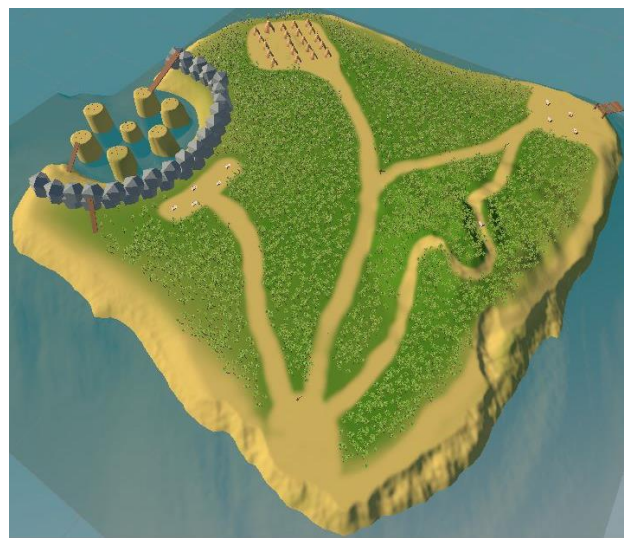
Međutim, kad dođe do mosta, igrač saznaje da je on uništen, zbog čega mora da se vrati nazad i proveriti ostale puteve da bi našao drugi most. Ispred oba mosta se nalaze čuvari. Zadatak Nana Boja je da ih pobeđi sve.

Jedan od preostalih puteva vodi do mesta na kom su postavljena skoro sva unapređenja za sakupljanje. U šumi se krije kapica koju igrač mora da sakupi ukoliko želi da na kraju vrati kapice kidnapovanim drugarima. Da nivo ne bi bio previše lak, te potencijalno dosadan, igrač mora da istraži mapu da bi našao pravi izlaz. Drugi put vodi ka mostu za sledeće ostrvo, čime se prelazi na sledeći nivo. Prvi nivo ima prirodno i mirno okruženje sa vedrim bojama. Ovo stvara prijateljsku i pozitivnu atmosferu koja je prikladna za prvi nivo igre.

5.2 Drugi nivo, Kamentos

Nana Boj počinje od mesta gde se završava most sa prethodnog nivoa, na ostrvu Kamentos. Ovo ostrvo ima više puteva i putokaza od prethodnog.

Ostrvo nije ravno kao Bananos, već blago brdovito (Slika 6). Na ovom ostrvu postoje tri glavna puta. Putokaz pokazuje da treba ići pravo da bi se stiglo do sledećeg ostrva. Ukoliko se ide ovim putem, kasnije se nailazi na račvanje, gde opet stoji putokaz koji pokazuje da treba ići desno. Na kraju puta se nalazi most, kojeg čuva veća grupa Snikera. Nivo se završava kada Nana Boj pobeđi čuvare mosta.



Slika 6. Drugi nivo, Kamentos

Kamentos je mnogo veće ostrvo od Bananosa, te ima i više tajni i prostora za istraživanje. Kada se na početku ode levim putem, nailazi se na veliki kameni zid i veliku grupu Snikera koji ga čuvaju. Iza zida se nalaze kokosi, raspoređeni po uzanim i visokim ostrvcima koja se nalaze u sred vode. Do ovih ostrvaca je moguće doći na više načina. Moguće je proći između kamenja zida, a zatim se popeti preko daske na jedno od ostrvaca. Drugi način je penjanjem preko daske na vrh zida, pa ići po zidu dok se ne dođe do drugog kraja zida na kom se nalazi daska koja se spušta do jednog ostrvca. Moguće je zaleteti se i preskočiti do sledećih ostrvaca kad se dođe do prvog. Visoka ostrvca su poredana u krug oko nižeg ostrvca na kom se nalazi skrivena kapica. Na svakom od ostrvaca su postavljena po četiri kokosa i na nekim se mogu naći i munje.

Kada se na srednjem putu na račvanju skrene levo, nailazi se na napušteno selo koje čuva jedan Sniker. Igrač ima opciju da se sukobi sa njim ili da ga izbegava, dok sakuplja kokose koji se nalaze ispod šatora. U glavnom šatoru se nalazi munja.

Ukoliko se na početku krene desnim putem, nailazi se na visoko brdo sa krivudavim putem. Na ovom brdu se krije jedan Sniker. On čuva skriveni šlag koji bi Nana Boju dao privremenu neuništivost. Nakon što se prođe brdo, ovaj put se spaja sa glavnim putem do mosta.

6. TREJLER

Trejler za video igru predstavlja kratak i sažet prikaz onoga što igrači mogu očekivati kada se upuste u svet igre. Za uspeh video igre, trejler je od ključnog značaja, jer služi kao prva tačka kontakta publike sa igrom. Njegova svrha je da zaintrigira, informiše i ubedi igrače da isprobaju ili kupe konkretnu igru.

Ciljna grupa za igru „Go Bananas“ su deca i mlađi adolescenti. Privlačenje ove grupe zahteva da trejler bude pun veselih i dinamičnih scena jarkih boja, uz pratnju vedre i živahne muzike koja bi im držala pažnju. Pošto igra već sama po sebi ima jarke boje, jednostavan i dopadljiv stil likova i okoline, dovoljno je uraditi dobru montažu snimaka nivoa i dodatnih scena uz živahnu muziku.

Odabrani format za trejler je MP4 format. Za kreiranje trejlera korišćeni su snimci gejملهja prvog nivoa i specijalnih scena kreiranih u Unity radnom okruženju (engl. *Unity game engine*) samo za tu svrhu. Za kreiranje likova korišćen je Blender, program za 3D modelovanje i animaciju. Za montažu trejlera korišćen je Clipchamp, a za kreiranje teksta Paint.NET.

Postoje tri scene koje se koriste u trejleru: scena pregleda nivoa, prvi nivo igre i scena bežanja od neprijatelja.

Scena pregleda nivoa se sastoji od okruženja prvog nivoa i potencijalnog drugog nivoa. Glavni lik se ne nalazi na ovoj sceni. Glavna kamera može da se upravlja tako da podseća na preletanje drona nad okolinom. Ideja iza ove scene je bila da se prikažu razni predeli koji se nalaze u svetu igre „Go Bananas“.

Iz prvog nivoa su odabrani najupečatljiviji delovi u kojima se lik uglavnom bori protiv neprijatelja ili sakuplja zanimljiva unapređenja.

Scena u kojoj glavni lik beži od grupe neprijatelja je kreirana da doda dozu napetosti i živahnosti trejleru. Ovaj deo služi kao najdinamičnija i najzanimljivija tačka trejlera, koja bi potencijalno ubedila gledaoce da probaju igru.

Za muziku u pozadini je odabrana interpretacija finala klasika „Uvertira Viljem Tel“ od Đoakina Antonija Rosinija, „William tell Overture Remix“ [2]. Ovu kompoziciju karakteriše brza i živahna melodija, koja je postala sinonim za jurnjavu i avanturu.

Rađen je blagi prelaz (engl. *fade*) između pojedinih scena i tekstova, da naglase početak jedne faze i ulazak u sledeću. U najintenzivnijem delu, kad neprijatelji jure protagonistu, prelasci između isječaka su brži. Snimci su sečeni tako da se uklapaju u muziku i sam kontekst

trejlera. Osim početka i kraja, muzika nije sečena, niti je izmenjena.

Trejler traje 1 minut, što je dovoljno dugo da se pokaže šta igra sve sadrži, a dovoljno kratko da se održi pažnja publike. Dobijeni rezultat je trejler [3] koji je dinamičan, te drži pažnju publike.

7. ZAKLJUČAK

Projekat je uspešno realizovan. Cilj je bio dizajnirati igru i realizovati deo igre sa voćnim protagonistom, zajedno sa trejlerom. Rezultat je početni deo igre „Go Bananas“, koji uključuje prva dva nivoa. Prvi nivo služi da uvede igrača delom ili u potpunosti u mehaniku igre, mogućnosti i izazove. Drugi nivo je kompleksniji i veći od prvog, kako bi se igrači pripremili za dalje nivoe, koji će vremenom postajati sve teži.

Najizazovniji deo bilo je adresiranje programskih bagova i nepredviđenih problema. U nekim situacijama, rešenja su pronađena brzo i efikasno, dok su drugi problemi zahtevali dublje istraživanje i više vremena. Bilo je i situacija gde je odlučeno da se određeni aspekti rada odlože, uzimajući u obzir potrebno vreme za njihovu implementaciju.

U poređenju sa konkurencijom, igra „Go Bananas“ se ističe jedinstvenom i osvježavajućom pričom i dizajnom. Međutim, prepoznato je da u nekim segmentima, poput mehanike ili tehničke realizacije, postoji prostor za poboljšanja u odnosu na veće naslove u industriji. Ova spoznaja može poslužiti kao putokaz za buduće modifikacije i usavršavanja.

Postoji dosta prostora za dalji razvoj video igre „Go Bananas“. Igra je kreirana za PC računare, pa bi mogla da se razvije i za mobilne uređaje. Potrebno je implementirati više nivoa, koji bi priču doveli do kraja. Takođe bi moglo uvesti nove likove i prepreke, koji bi obogatili igru. Razni efekti poput potresa kamere bi mogli da poboljšaju imerziju igrača.

8. LITERATURA

- [1] C. Fabricatore, „Gameplay and game mechanics design: a key to quality in videogames“, *OECD-CERI Expert Meeting on Videogames and Education*, Октобар, 2007.
- [2] <https://youtu.be/P7svUxY2Gns?si=lefAGnkTMFZzqH0o> (pristupljeno u oktobru 2023.)
- [3] <https://youtu.be/buxRiGtgXJc?si=j6vqxmyLU4FDuBh> (pristupljeno u oktobru 2023.)

Kratka biografija:



Isidora Stanić rođena je u Novom Sadu 1999. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Softversko inženjerstvo i informacione tehnologije odbranila je 2023. god.
kontakt: isidora.stanic@uns.ac.rs

ИЗАЗОВИ ПРИ РАЗВОЈУ МОТО КРОС ВИДЕО ИГРЕ**CHALLENGES IN THE PROCESS OF DEVELOPING A MOTOCROSS VIDEOGAME**

Младен Васић, Факултет техничких наука, Нови Сад

Област – СОФТВЕРСКО ИНЖЕЊЕРСТВО И ИНФОРМАЦИОНЕ ТЕХНОЛОГИЈЕ

Кратак садржај – Тема овог рада јесте опис мото крос видео игре. Видео игра је инспирисана другим видео играма сличне тематике. Рад представља причу и главне ликове у овој игри, механизме играња и типове нивоа. Алати употребљени при развоју су Unity, Blender, Paint.NET, и Audacity.

Кључне речи: Развој видео игре, Мото крос, Иммерсивност, Гејмплеј, Прича, Изазов

Abstract – The subject of this paper is description of a motocross videogame. The videogame is inspired by other videogames of similar theme. This paper presents the story and main characters in this game, gameplay mechanics and types of level present. Tools used in the development process are Unity, Blender, Paint.NET and Audacity.

Keywords: Videogame development, Motocross, Immersion, Gameplay, Story, Challenges

1. УВОД

Индустрија видео игара је званично настала 1971. године, појавом *Computer space*, прве аркадне видео игре, уједно и прве комерцијалне видео игре икада. Следеће године компанија Атари избацује прву комерцијално успешну видео игру, *Pong*, са 19000 продатих аркадних кабинета 0. Те исте, 1972. године се појављује и прва кућна конзола за видео игре, *Magnavox Odyssey*. Међу многим клоновима игре *Pong* се појављује и прва тркачка видео игра *Gran Trak 10*. Ову тркачку игру је направила компанија Атари за аркадне машине 1974. године. Кабинет је имао на себи волан за управљање аутомобилом и мењач за мењање брзина, а видео игра је имала приказ целе стазе из птичије перспективе.

1976. године компанија Сега избацује прву тркачку игру са мотоциклима *Moto-cross*. Ова видео игра је такође имала назив *Man T.T.* и *Fonz*. Кабинет ове игре је имао псеудо-3D приказ и гуверналу мотоцикла којом су играчи управљали мотоциклом. Ова видео игра је прва икада која је користила вибрације да дочара силе које делују на мотоцикл током вожње 0.

Средином 1980-их, компанија Сега популаризује аркадне кабине које у себи садрже хидрауличне симулаторе покрета кроз своју серију таикан видео игара. Таикан значи телесни осећај на јапанском језику. Прва Сегина таикан видео игра је била *Hang-on* из 1985. године. Аркадни кабинет ове видео игре је имао реплику мотоцикла повезану са екраном на којој би играч седео и управљао мотоциклом. Реплика мотоцикла је у себи имала уграђене моторе који би симулирали силе које делују на мотоцикл при скретању. Ова видео игра је једна од првих игара које су користиле 16-битну графику и Сегину *Super Scaler* технологију која је омогућавала псеудо-3D скалирање спрајтова уз велик број фрејмова (eng. *Frames per second*). *Hang-on* је постала најпопуларнија аркадна игра у САД 1986. године, такође једна од најпопуларнијих аркадних игара у Јапану и Лондону.

Током 1990. многи напреси у технологији који су омогућили видео игре са потпуно 3D графиком и квалитетним звучним записима. Неке од значајнијих видео игара са мотоциклима овог доба су *Road Rash*, *Moto racer* и *Moto extreme*. Наведене видео игре су биле доступне на различитим платформама, међутим све три су биле присутне на РС платформама. Поменуте игре су истакнуте као неке од првих видео игара са мотоциклима и полигоналном 3D графиком.

У овом раду ће бити представљена мото крос видео игра под називом „Оверајдин“. Ова видео игра је настала као резултат проучавања других видео игара сличне тематике. Прво ће бити дати основни подаци о видео игри, њена премиса, циљна група и удица. У наредном поглављу ће бити изложена прича и главни актери у причи. Након приче ће бити дата структура механизма ове видео игре, као и структуре различитих типова нивоа који су присутни. На крају овог рада ће бити дати неки од праваца у којима би ова видео игра могла да се даље развија.

2. ВИДЕО ИГРА „ОВЕРАЈДИН“

Ово поглавље садржи основне податке о видео игри коју овај мастер рад излаже. Прво је дата премиса ове видео игре, затим њен наслов, жанр, циљне платформе и концепт. Затим је наведена циљна група играча, удица и на крају је дата кратка анализа конкуренције, то јест описане су видео игре које су сличне игри из рада.

2.1 Основни подаци

Наслов игре: Оверајдин' (eng. *Overridin'*)

Жанр игре: Аркадна тркачка игра

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Драган Иветић, ред. проф.

Циљане платформе: РС

Концепт игре: Оверајдин' је 3Д тркачка игра намењена широком спектру играча (7 и више година), мада првенствено играчима који воле изазовније игре. Играчи преузимају контролу над Рајдером (eng. *Ryder*) на мотоциклу и њихов задатак је да прођу различите полигоне који су постављени пред њима.

2.2 Премиса

Прича прати Рајдера, фанатика мотоцикала који једног дана одлази да посети локални мото скуп у природи и пролазећи кроз масу, он примећује једну дивну младу даму, Мишел.

Рајдер долази на идеју да импресионира Мишел тако што ће да се такмичи на различитим полигонима у нади да ће га Мишел приметити из масе.

2.3 Циљна група и удица

Како би се привукло што више играча, није дефинисана конкретна циљна група. Игра је намењена свакоме ко воли да игра видео игре, међутим због тежине игре би доња старосна граница била 7 година. Уметнички правац ове игре диктира реалистична окружења са уверљивим звучним ефектима, како би се побољшала имерсивност ове видео игре.

Само управљање мотоциклом није толико реалистично као њен изглед како оно не би одвратило превише играча од играња ове видео игре. Прича игре је генерално једноставна, мада има један изненадан преокрет при крају који би могао оставити утисак на велик број играча. Сама прича такође даје благи друштвени коментар.

3. ПРИЧА

Ово поглавље ће се фокусирати на излагање приче присутне у позадини ове видео игре. Прво ће бити представљени главни актери и споредни ликови који се појављују у причи.

Затим ће бити описана прича у целости по хронолошком редоследу и биће приказан *storyboard* приче.

3.1 Ликови



Слика 1 Рајдер

Рајдер

Улога: Лик којим играч управља током већег дела игре

Кратак опис: Ћутљив и повучен тип. Обожава мотоцикле, први пут је сео на мотоцикл са 7 година.

Физички изглед: Носи свој црвено-зелени комбинезон, можда и превише често.

Способности: Веома брз и спретан на мотоциклу.



Слика 2 Мишел

Мишел

Улога: Главна мотивација за Рајдера

Кратак опис: Лепа млада дама која посећује исти мото скуп као и Рајдер. Њен дечко, Колин, наступа на изазовима на мото скупу и она му је дошла као подршка.

Физички изглед: Плавокоса је и са плавим очима, носи ружичасту мајицу, беле хеланке и црне патике.



Слика 3 Колин

Колин

Улога: Дечко од Мишел. Након преокрета у причи, играч преузима контролу над њим

Кратак опис: У вези је са Мишел већ дуже време и планира да је запроси. Воли да ради на себи и воли добар изазов.

Физички изглед: Има смеђу косу и смеђе очи, носи плаво-зелени комбинезон

Способности: Подједнако добар на мотоциклу као и Рајдер.

3.2 Ток приче

Рајдер живи у предграђу Спрингфилда, у САД. Слушајући радио једног дана, Рајдер сазнаје да ће се ускоро одржати мото скуп недалеко изван града. Рајдер се спрема да иде на мото скуп. Рајдер стиже на мото скуп и иде околу и разгледа све догађаје. Пролазећи поред једног полигона, он примећује Мишел како са стране посматра исти полигон.

Рајдер се заљубљује у њу истог момента, и учиниће све да придобије њену пажњу. Под претпоставком да ће је импресионирати својим умећем са мотоциклом, Рајдер се пријављује за учешће на полигонима. Током такмичења, Рајдер често наилази на Колина, заправо на сваком полигону. Мишел изгледа да је заинтересована за Рајдера, пошто је међу масом која посматра сваки од његових наступа. Након свих полигона, Рајдер одлучује да приђе Мишел и да јој изјави своју љубав. Пошто Мишел саслуша шта Рајдер има да јој каже, она га учтиво одбија и саопштава му да је заузета. Рајдер је сломљен овом изјавом јер је под утиском да је сав његов напор

отишао узалуд, али одбија да прихвати да је Мишел ван његовог домета. Том приликом долази и Колин, Мишел га упућује у ситуацију и он покушава да се договори са Рајдером. Обојица ће се тркати у обрачуну један на један, пред свим посетиоцима мото скупа. Ако Рајдер изгуби, он више не сме узнемиравати Мишел на било који начин, а ако победи онда ће моћи да изведе Мишел на један излазак. Креће коначна трка између Колина и Рајдера. Колин је и пре трке био сигуран у своју вештину и то се и показало током трке јер је већи део трке био у благом вођству. Међутим, Рајдер је одлучио да игра прљаво и извадио је ланац који је сакрио у свој комбинезон. Рајдер је тим ланцем ударио Колина и оборио га са мотоцикла.

Иако је Рајдер победио, Мишел није желела да испоштује договор који су постигли он и Колин, јер је Рајдер varaо. Из дубоког очајања, Рајдер одлучује да киднапује Мишел на мотоциклу. Сада је на Колину да стигне Рајдера како би спасио своју девојку и притом је Колин свестан прљавих трикова на које је спреман Рајдер. Колин успева да стигне Рајдера и да спаси своју девојку, а Рајдер одлази потпуно осрамоћен, и више се никад није појавио на мото скупу.

4. ГЕЈМПЛЕЈ

Гејмплеј је скуп активности које могу бити извршене током лудичког (lat. *Ludo, ludere* - играти) искуства од стране играча или као одговор ентитета, који припадају виртуалном свету, на играчеве акције и/или као самосталне токове акција које доприносе живости виртуалног света 0.

У овом погављу биће изложена логичка структура игре, затим ће бити описана правила игре, њени механизми, изазови, награде и казне, структура нивоа и звукови.

Постоје два мода, прича (eng. *Story mode*) и аркадни мод (eng. *Arcade mode*). У прича моду играч контролише Рајдера (и Колина) и игра кроз причу описану у поглављу 3.2. Прича мод може да се паузира и настави на истом нивоу који је претходно био започет.

Аркадни мод је збирка више различитих нивоа које не повезује никаква прича, већ служе као додатни изазов за играча након што заврши причу или као бржи начин да играч одмах ускочи у акцију, без икакве експозиције или туторијала. Играч сакупљањем поена од вратоломија може да купи додатне моделе мотоцикала или додатне комбинеzone за возача и њих може да користи искључиво у аркадном моду. За сваки ниво постоји *online leaderboard* на ком се може видети који играч држи најбоље време или ко је скупио највише поена.

4.1 Правила игре

Постоје две врсте нивоа, полигонски (eng. *checkpoint*) изазови и стилски изазови. Код полигонских изазова, циљ играча јесте да прође мотоциклом кроз све капије у нивоу за задато време. Уколико играч претера и мотоцикл падне на земљу или уколико играч удари у објекте из околине, он губи у том нивоу и мора га покушати испочетка. Потребно је проћи кроз све

капије у задатом редоследу и обавезно је проћи кроз све.

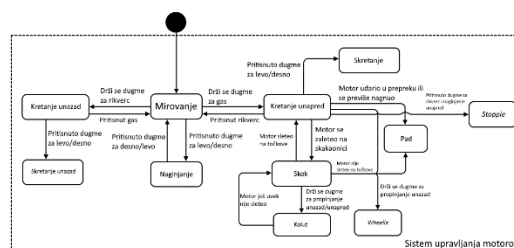
Код стилских изазова, циљ је сакупити довољно бодова у задатом времену. Играч добија поене тако што изводи вратоломије као што су колут уназад, колут унапред, пропињане на задњи точак (eng. *Wheelie*) и пропињање на предњи (eng. *Stoppie*). За разлику од полигонских изазова, играч овде има на располагању пространу област са доста прилика да изведе вратоломију, то јест има много неравнина, скакаоница и правина. Играч може слободно да се креће по овој области.

4.2 Механизми игре

Са становишта система игре, механизми игре се користе да опишу како играчи могу да интерагују са правилима игре и осталим формалних својствима као што су циљеви, акције играча, стратегије и стања игре. Ејвдон и Сатон-Смит су 1971. године први смислили формалну структуру игара и фиксних принципа (токови акција, начини играња, процедура акције), који одређују играње и понашање игре. Бјорк и Холопајнен су 2005. године дефинисали механизме као шаблоне правила у било ком делу система правила једне игре који покривају јединствен скуп интеракције током игре, било да су специфичне или опште. Фулerton и његови сарадници су 2004. године дефинисали механизме игре као акције или начини играња игре ограничени правилима, који креирају интеракције и усмеравају играчево понашање. Ги 2003. године претпоставља да механизми у играма чине поновљено увежбавање кроз све изазовније нивое игре. Мноштво литературе открива да не постоји једногласно прихваћена дефиниција механизма у играма 0.

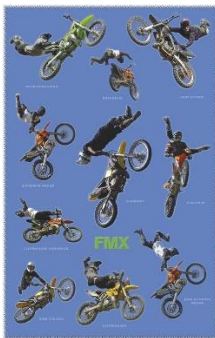
У овом поглављу ће бити наведени неки од сржних (eng. *core*) механизма и сателитских механизма. Сржни механизми представљају основне акције без којих играч не може испунити циљеве играња као што је нпр. кретање. Сателитски механизми су споредни механизми који служе да обогате играчево искуство и додатно га награде ако успешно изврши њихове акције.

4.2.1 Механизми игре



Слика 4 Сржни механизми

Основни механизам ове игре јесте кретање мотоцикла. Мотоцикл има команду за гас и рикверц и може да се нагне лево или десно. У ваздуху може и да се нагиње унапред или узназад. На слици 10 је илустровано како се ове команде компонују заједно и чине систем управљања мотоциклом.



Слика 5 Трикови на мотоциклу

Поред кретања, играч може да изводи различите трикове док је у ваздуху. Играч добија поене ако успешно изведе трик. Да би успешно извео трик, играч мора да га изведе пре него што слети на земљу. Уколико играч успешно уланча више трикова у једном скоку, добиће умножак поена једнак броју изведених трикова. На слици 11 могу се видети неки од могућих трикова.

4.3 Изазови, награде, казне

Изазове у овој видео игри представљају нивои које је потребно прећи у причи. Завршавањем једног нивоа се откључава следећи. Током играња, могуће је изводити трикове и у полигонским и стилским изазовима. Поени зарађени триковима се акумулирају и чувају глобално унутар видео игре за касније, јер се уз помоћ њих могу купити други модели мотоцикала или комбинезони за возача у аркадном моду.

Постоји више казни током играња. Уколико истекне све време које је предвиђено за полигонски ниво, играч мора играти исти испочетка. Уколико играч не скупии довољно поена предвиђено за стилски ниво, играч такође мора играти испочетка. Уколико се играч превише нагне и додирне тло мотоциклом, он ће пасти. Уколико мотоцикл удари у неку препреку (дрво, камен и слично), он ће такође пасти. Када играч падне, он мора почети ниво испочетка. Ако играч промаши капију у полигонском нивоу или покуша да је намерно прескочи, мораће да се до исте врати јер неће моћи да заврши ниво.

4.4 Дизајн нивоа

У овом поглављу ће бити описана генерална структура нивоа који се могу пронаћи у овој видео игри. Такође ће бити дат један пример изгледа нивоа.



Слика 6 Изглед једног нивоа

Терени нивоа направљен је уз помоћ *Tom's terrain tools* асета.

Полигонски нивои су линеарне природе, имају стриктно један почетак и крај. Поред препрека на терену, временско ограничење је постављено на сваки

полигонски ниво да би се исти отежао за играча да пређе.

Стилски нивои су типично отворене природе, што их чини различитим од полигонских нивоа. Немају обавезну путању коју је потребно проћи, играч сам бира како ће да испуни границу поена која је одређена за овакав ниво. И ови нивои као и полигонски нивои имају временско ограничење, како би представило додатни изазов за играча.

5. ЗАКЉУЧАК

У уводу је било речи о историји видео игара укратко. Затим су наведене неке од најзначајнијих видео игара у контексту овог рада. Игра је жанра тркачке игре са неким *multiplayer* карактеристикама као што су *online leaderboard* у аркадном моду. „Оверајдин“ има два главна мода, прича и аркадни мод. У моду приче, играч прелази нивое који прате причу Рајдера, Мишел и Колина. Поред мода приче, играч такође има приступ аркадном моду, у којем може безброј пута да игра било који од нивоа из листе, неvezано за причу.

У овој игри су присутна два типа нивоа, полигонски и стилски изазови. Циљ полигонског изазова јесте проћи кроз све капије за задато време у одговарајућем редоследу. Циљ стилских изазова је извести што више трикова како би играч сакупио довољно поена да пређе границу нивоа.

Постоји више праваца у којима би ова видео игра могла да се развија. Тренутно је предвиђена само за РС рачунаре, мада би могла бити и портована на андроид и iOS платформе уз прилагођавање контрола. Уместо бирања нивоа преко менија, у моду приче могла би бити имплементирана област по којој играчи могу да се крећу пешке, контролишући Рајдера. Овакав приступ би додатно побољшао имерсивност играча у свет видео игре.

6. Литература

- [1] Wikipedia, "Video game industry", https://en.wikipedia.org/wiki/Video_game_industry, објашњава појам индустрије видео игара,
- [2] Wikipedia, "Racing game", https://en.wikipedia.org/wiki/Racing_game, објашњава развој тркачких игара
- [3] C. Fabricatore, "Gameplay and game mechanics design: a key to quality in videogames", OECD-CERI Expert Meeting on Videogames and Education, Октобар, 2007.
- [4] T. Lim, S. Louchart et al., "Strategies for Effective Digital Games Development and Implementation", Cases on Digital Game-Based Learning, 12, 168–198, 2013.

Кратка биографија:



Младен Васић рођен је у Шапцу 1999. год. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства – Софтверско инжењерство и информационе технологије – модул Мултимедијални системи и рачунарске игре одбранио је 2023.год.
контакт: mladenvasic170@yahoo.com

PRORAČUN TOKOVA SNAGA SA UVAŽENOM UDALJENOM REGULACIJOM NAPONA**POWER FLOW CALCULATION CONSIDERING REMOTE VOLTAGE REGULATION**Milica Brkušnin, Neven Kovački, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U ovom radu je predstavljeno proširenje klasičnog modela tokova snaga sa ciljem da se uvaži efekat udaljene regulacije napona. Udaljena regulacija napona podrazumeva da se u svakoj zoni elektroenergetskog sistema definiše po jedan pilot čvor u kome se reguliše napon tako što se menjaju proizvodnje reaktivnih snaga generatora u datoj zoni. Proširenje klasičnog modela tokova snaga se sastoji u uvođenju dva nova tipa čvorova: PQV i P. Predstavljeni model tokova snaga je verifikovan na primeru mreže od 30 čvorova, a rezultati su prikazani i komentarisani u radu.

Ključne reči: Proračun tokova snaga, Udaljena regulacija napona, Pilot čvorovi, Newton-Raphsonova metoda.

Abstract – This paper presents extension of power flow model to consider the effect of remote voltage regulation. Remote voltage regulation is conducted such that in each zone of the power system a pilot bus is defined, in which the voltage is regulated by adjusting the reactive power production on generators in the considered zone. The extension power flow model involves introducing of two new types of nodes: PQV and P. Remote voltage regulation has been verified on test system with 30 buses, and the results are presented and discussed in the paper.

Keywords: Power flow calculation, Remote voltage regulation, pilot buses, Newton-Raphson method.

1. UVOD

Proračun tokova snaga je jedan od najvažnijih proračuna u analizi elektroenergetskih sistema (EES). Rezultati ovog proračuna se široko koriste u planiranju i eksploataciji EES. Bilansne jednačine aktivnih i reaktivnih snaga koje čine klasični model tokova snaga su nelinearne, zbog čega se za njegovo rešavanje najčešće koriste iterativne metode za rešavanje sistema nelinearnih jednačina. U ovom radu se za to primenjuje Newton-Raphson metoda.

S obzirom na veliki značaj ovog proračuna, od velikog značaja je da se u njemu na adekvatan način uvaži efekat regulacije napona u EES. Regulacija napona hijerarhijski se može podeliti na primarnu, sekundarnu i tercijarnu regulaciju. U okviru sekundarne regulacije napona posebno je važno uvažiti efekat udaljene regulacije napona.

NAPOMENA:

Ovaj članak proistekao je iz master rada čiji mentor je bio docent dr Neven Kovački.

Udaljena regulacija napona podrazumeva da se u jednom ili više odabranih pilot čvorova (eng. pilot bus) reguliše napon proizvodnjom reaktivne snage u generatorima koji su više ili manje udaljeni od pilot čvorova. Napon pilot čvor postaje konstantan u proračunu tokova snaga, dok je u čvoru koji reguliše pilot čvor (kontrolni generatorski čvor) napon varijabilan. U ovom radu je predstavljeno proširenje klasičnog modela tokova snaga sa ciljem da se uvaži efekat udaljene regulacije napona.

2. KLASIČNI MODEL TOKOVA SNAGA

U ovom poglavlju predstavljen je klasični model tokova snaga. Pri tome je opisana Netwon-Raphsonova metoda za proračun tokova snaga.

Svaki čvor k opisuje se sa šest promenljivih [4]:

- aktivna i reaktivna snaga proizvodnje P_{G_k} i Q_{G_k} ,
- aktivna i reaktivna snaga potrošnje P_{P_k} i Q_{P_k} ,
- modul i fazni stav fazora napona V_k i θ_k .

Injektirana snaga se predstavlja kao [4]:

$$\underline{S}_k = P_k - jQ_k, \quad (2.1)$$

gde je k čvor u elektroenergetskom sistemu, $\underline{S}_k$ injektirana kompleksna snaga, P_k injektirana aktivna, Q_k injektirana reaktivna snaga čvora k .

Aktivna i reaktivna injektirana snaga čvora k se dobija razlikom aktivne, odnosno reaktivne snage proizvodnje i potrošnje [4]:

$$P_k = P_{G_k} - P_{P_k}, \quad (2.2)$$

$$Q_k = Q_{G_k} - Q_{P_k}. \quad (2.3)$$

Injektirana snaga čvora k je jednaka nuli, ako u čvoru nije priključen nijedan generator, ni potrošač.

Čvorovi u elektroenergetskim sistemima mogu se podeliti na: referentno-balansni čvor (tip θV – poznati su V_k i θ_k), potrošačke čvorove (tip PQ – poznati su P_k i Q_k) i generatorske čvorove (tip PV – poznati su P_k i V_k).

Relacija bilansa kompleksnih snaga u čvorovima elektroenergetskog sistema [4]:

$$\underline{S}_k = \underline{V}_k^* \underline{I}_k, \quad (2.4)$$

$$\underline{I}_k = \sum_{i=1}^n \underline{Y}_{ki} \underline{V}_i, \quad (2.5)$$

gde je $\underline{V}_k$ fazor napona u čvoru k, $\underline{I}_k$ injektirana kompleksna struja čvora k, dok $\underline{Y}_{ki}$ predstavljaju elemente matrice admitansi.

Kombinujući relacije (2.1), (2.4) i (2.5) dobija se:

$$P_k - jQ_k = \underline{V}_k^* \sum_{i=1}^n \underline{Y}_{ki} \underline{V}_i. \quad (2.6)$$

Relacije za bilanse aktivnih i reaktivnih snaga u čvorovima elektroenergetskog sistema dobijaju se iz relacije (2.6) [1]:

$$P_k = \text{Re} \left\{ \underline{V}_k^* \sum_{i=1}^n \underline{Y}_{ki} \underline{V}_i \right\}, \quad (2.7)$$

$$Q_k = -\text{Im} \left\{ \underline{V}_k^* \sum_{i=1}^n \underline{Y}_{ki} \underline{V}_i \right\}, \quad (2.8)$$

gde sa leve strane čine injektirane aktivne i reaktivne snage, a desne strane odgovarajuće snage koje iz tih čvorova granama otiču u sistem.

Proračun tokova snaga svodi se na rešavanje sistema nelinearnih jednačina koga čine relacije (2.7) za svaki PQ i PV čvor, i relacije (2.8) za sve PQ čvorove. U svakoj iteraciji Newton-Raphsonove metode rešava se sledeći linearizovani sistem jednačina [5]:

$$\begin{bmatrix} \Delta P^{(m)} \\ \Delta Q^{(m)} \end{bmatrix} = -J \Delta x^{(m+1)}, \quad (2.9)$$

$$\begin{bmatrix} \Delta P^{(m)} \\ \Delta Q^{(m)} \end{bmatrix} = -J \begin{bmatrix} \Delta \theta^{(m+1)} \\ \Delta V^{(m+1)} \end{bmatrix}, \quad (2.10)$$

$$\Delta P = [\Delta P_{PV} \quad \Delta P_{PQ}]^T, \quad (2.11)$$

$$\Delta Q = [\Delta Q_{PQ}]^T, \quad (2.12)$$

$$\Delta x = [\Delta \theta_{PV} \quad \Delta \theta_{PQ} \quad \Delta V_{PQ}]^T, \quad (2.13)$$

gde je m broj iteracije, ΔP razlika poznate aktivne snage i injektiranih aktivnih snage u tekućoj iteraciji, ΔQ razlika poznatih reaktivnih snaga i injektiranih reaktivnih snaga u tekućoj iteraciji, J matrica Jakobijana, Δx – priraštaj nepoznatih varijabli. Iterativni postupak se ponavlja do ispunjenja zadatih kriterijuma konvergencije.

Dimenzije Jakobijana su [5]:

$$J = \begin{bmatrix} J_1 & J_3 \\ J_2 & J_4 \end{bmatrix}, \quad (2.14)$$

gde je:

J_1 (n - 1) · (n - 1) matrica, a ulaz predstavlja $\partial P / \partial \theta$,

J_2 (n - 1) · n_{PQ} matrica, a ulaz predstavlja $\partial P / \partial V$,

J_3 n_{PQ} · (n - 1) matrica, a ulaz predstavlja $\partial Q / \partial \theta$,

J_4 n_{PQ} · n_{PQ} matrica, a ulaz predstavlja $\partial Q / \partial V$,

pri tome je n ukupan broj čvorova, n_{PQ} broj PQ čvorova.

3. REGULACIJA NAPONA I REAKTIVNIH SNAGA

Regulacija napona je složen proces koji zahteva pažljivo upravljanje reaktivnom snagom, korišćenje regulacionih resursa i primenu odgovarajućih tehnika. Cilj je održavanje željenih nivoa napona unutar propisanog tolerancijskog opsega kako bi se obezbedilo stabilno i pouzdano snabdevanje električnom energijom.

Prema regulacionim dejstvima hijerarhijska organizacija upravljanja naponima i reaktivnim snagama se deli na primarnu, sekundarnu i tercijalnu.

Sekundarna regulacija napona se postiže tako što se u svakom delu, tj. zoni sistema zadaje po jedan čvor u kome se napon reguliše prema zadatoj vrednosti. Ovi čvorovi se obično nazivaju pilot čvorovi. Pilot čvorovi se obično biraju tako da njihov napon oslikava celokupni naponski profil u određenoj zoni. Pilot čvorovi se u modelu tokova snaga predstavljaju kao PQV čvorovi (P, Q, V je poznato, θ je nepoznato).

Zadati naponi u pilot čvorovima se regulišu injektiranjem reaktivne snage generatora u datoj zoni. Pri tome može se koristiti jedan generator, ili više generatora koji su priključeni u različitim čvorovima. Generatori obično nisu priključeni u samim pilot čvorovima, nego u čvorovima koji su više ili manje udaljeni od pilot čvora i zato se ovaj postupak naziva još i udaljena regulacija napona.

Čvorovi u kojima su priključeni generatori koji se koriste za udaljenu regulaciju napona se u modelu tokova snaga predstavljaju kao P čvorovi (P je poznato, Q, V i θ je nepoznato).

Svakom PQV čvoru se dodeljuje ili jedan P čvor ili više P čvorova. Ako se zadaje više P čvorova, onda govorimo o distribuiranom P čvoru i zadaju se faktori učešća za svaki od P čvorova koji u tome učestvuju.

4. PROŠIRENJE MODELA TOKOVA SNAGA

Klasifikacija čvora se proširuje sa dva nova tipa čvora PQV i P. Čvor koji sadrži generator je po klasičnom modelu tokova snaga čvor tipa PV, dok se u ovoj glavi predstavlja kao čvor P za slučaj regulacije udaljenih pilot čvorova.

Čvor koji je kontrolisan je po klasičnom modelu tokova snaga čvor tipa PQ, takav pilot čvor zbog dodate regulacije napona postaje PQV čvor, pošto je PQV čvoru poznata aktivna snaga, reaktivna snaga i modul napona [3].

Injektirana reaktivna snaga za k-ti čvor se proračunava kao [3]:

$$Q_k = -\text{Im} \left\{ \underline{V}_k^* \sum_{i=1}^n \underline{Y}_{ki} \underline{V}_i \right\} - \sum_{i \in PQV} Q_{ki}^P, \quad (4.1)$$

$$Q_{ki}^P = p\text{Faktor}_{ki} \cdot \Delta DQ_i \quad (4.2)$$

gde je k čvor tipa P, $p\text{Faktor}_{ki}$ faktor učešća za svaki P čvor, ΔDQ_i reaktivna snaga koja se distribuira u P čvor. Suma svakog faktora učešća mora biti jednaka jedinici.

Proračun tokova snaga Newton-Raphsonovom metodom za relaciju (2.9) se proširuje na sledeći način:

$$\Delta P = [\Delta P_{PV} \quad \Delta P_{PQ} \quad \Delta P_P \quad \Delta P_{PQV}]^T, \quad (4.3)$$

$$\Delta Q = [\Delta Q_{PV} \quad \Delta Q_{PQV} \quad \Delta Q_P]^T, \quad (4.4)$$

$$\Delta X = [\Delta \theta_{PV} \quad \Delta \theta_{PQ} \quad \Delta \theta_P \quad \Delta \theta_{PQV} \quad \Delta V_{PQ} \quad \Delta V_P \quad \Delta DQ]^T. \quad (4.5)$$

Dimenzije Jakobijana iz relacije (2.14) su proširene na sledeći način [5]:

J_1 $(n - 1) \cdot (n - 1)$ matrica, a ulaz predstavlja $\partial P / \partial \theta$,

J_2 $(n - 1) \cdot (n_{PQ} + n_P + n_{DQ})$ matrica, a ulaz predstavlja $\partial P / \partial V$ i $\partial P / \partial DQ$,

J_3 $(n_{PQ} + n_P + n_{PQV}) \cdot (n - 1)$ matrica, a ulaz predstavlja $\partial Q / \partial \theta$,

J_4 $(n_{PQ} + n_P + n_{PQV}) \cdot (n_{PQ} + n_P + n_{DQ})$ matrica, a ulaz predstavlja $\partial Q / \partial V$ i $\partial Q / \partial DQ$,

pri tome je n ukupan broj čvorova, n_{PQ} broj PQ čvorova, n_P broj P čvorova, a n_{PQV} je ukupan broj PQV čvorova.

ΔDQ se proračunavaju na sledeći način:

$$\partial P_i / \partial DQ_j = 0, \quad (4.6)$$

$$\partial Q_i / \partial DQ_j = -pFaktor_{ij}, \quad (4.7)$$

$$\partial P_i / \partial DQ_i = 0, \quad (4.8)$$

$$\partial Q_i / \partial DQ_i = 0, \quad (4.9)$$

gde je:

i – indeks čvora (1, 2...n),

j – indeks čvora (1, 2...n),

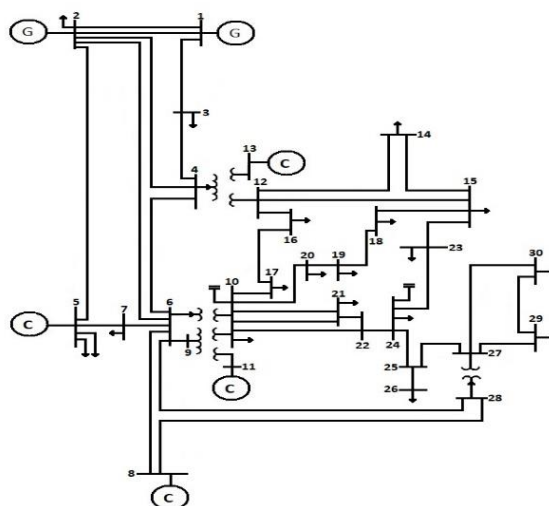
$i \neq j$.

5. VERIFIKACIJA PREDSTAVLJENOG MODELA

Na slici 1. prikazan je model mreže koji se sastoji od 30 čvorova, od kojih su 5 generatorskih čvorova, 24

potrošačka čvora, i 1 referentno-balansni čvor. Model je deo američkog elektroenergetskog sistema, koji je dostupan elektroenergetskoj industriji kao standardni testni slučaj za evaluaciju različitih analitičkih metoda i računarskih programa za rešavanje problema elektroenergetskog sistema. Proračun se vrši nad klasičnim modelom tokova snaga (poglavlje 2), i nad testnim primerima 1, 2 i 3 koji su dobijeni proširenim modelom tokova snaga (poglavlje 4). Razlika između test primera 1 i 2 jeste ispad voda indeksa 2. i 4. kod test primera 2. Test primer 3 ima iste ulazne vrednosti kao test primer 1, osim razlike u vrednostima napona čvora 4 (PQV čvor), gde je postavljen povišen napon.

U tabeli 1. prikazani su najvažniji rezultati proračuna tokova snaga primenom Newton-Raphsonove metode. U cilju potvrde tačnosti proračuna, rezultati za klasičan model tokova snaga su upoređeni sa vrednostima literature iz reference [7]. Ovaj postupak analize rezultirao je zaključkom da su proračunate vrednosti precizno reprodukovane i dosledno usaglašene sa vrednostima navedenim u [7]. Ovaj rezultat pruža potvrdu o tačnosti računskih metoda primenjenih u okviru istraživanja.



Slika 1. Model mreže

Tabela 1. Rezultati proračuna

	Čvor 4		Čvor 2			Grana 2 - 4		Broj iteracija Newton- Raphsonove metode
	tip	V[r.j.]	tip	V[r.j.]	Q_G [MVar]	P[MW]	Q[MVar]	
Klasičan model tokova snaga	PQ	1.01285	PV	1.043	48.8221	45.7118	2.70487	3
Test primer 1	PQV	1.01285	P	1.043	48.8226	45.7119	2.70495	3
Test primer 2	PQV	1.01285	P	1.12513	290.032	/	/	3
Test primer 3	PQV	1.02	P	1.06758	130.486	46.1188	13.3576	3

U test primeru 1, vrednost PQV čvora je postavljena na vrednost dobijenu proračunom klasičnog modela tokova snaga, dok je ulazni podatak za P čvor postavljen na vrednost 1[r.j.]. Analizom klasičnog modela tokova snaga sa test primerom 1, može se primetiti da su rezultati približni. Razlika se pojavljuje zbog zaokruživanja vrednosti modula napona na petu decimalu u test primeru 1, na čvoru 4 (PQV čvoru).

Test primer 2 ima iste ulazne parametre kao test primer 1, sa jedinom razlikom u ispadu voda između čvorova 2 i 4, kod test primera 2. Ovo je rezultiralo povećanjem napona na P čvoru, sa njim i povećanje reaktivne snage. U ovom test primeru, zbog ispada voda na čvorovima 2. i 4., došlo je do provere mogućnosti udaljene regulacije generatora za slučaj da ne postoji direktna veza tj. grana između P i PQV čvora. Na slici 1. se može videti da postoji samo jedan vod između čvora indeksa 2. i čvora indeksa 4., a u prikazanim rezultatima se zaključuje da se udaljena regulacija generatora može izvršiti i za ovaj slučaj.

Test primer 3 ima iste ulazne podatke kao test primer 1, sa jedinom razlikom u povećanju napona na čvoru 4 (PQV čvoru). Ovo je rezultiralo povećanjem napona na P čvoru, sa njim i povećanje reaktivne snage, što utiče na naponsko-reaktivnu stabilnost elektroenergetskog sistema

Proveravala se naponska-reaktivna zavisnost na mreži sa generatorskom udaljenom regulacijom napona. Vrednost napona na PQV čvoru se povećavala i smanjivala unutar granica nominalnog napona. Nakon analize rezultata, utvrđeno je da povećanje napona na PQV čvoru rezultira povećanje reaktivne snage generatora u P čvoru, što dalje doprinosi povećanju napona na tom čvoru. S druge strane, kada se moduo napona na PQV čvoru smanjuje, rezultira se smanjenjem reaktivne snage generatora u P čvoru, što dovodi do smanjenja napona na tom čvoru.

Proračun je konvergirao nakon tri iteracije u sva četiri slučaja, nije došlo do povećavanja vremena izvršavanja proračuna u odnosu na klasični model tokova snaga.

6. ZAKLJUČAK

Proverom rezultata Newton-Raphsonovim proračunom za klasični i prošireni model tokova snaga u mreži sa 30 čvorova, dolazi se do zaključka da uvođenje PQV i P čvorova omogućuje uvažavanje udaljene regulacije napona u klasičnom modelu tokova snaga.

Rezultati test primera su pokazali da se PQV i P čvorovi ponašaju sa očekivanim rezultatima tokova snaga i naponskih stanja, tj. kao klasičan model tokova snaga proračun sa PQ i PV čvorom.

Sa povećanjem zadatog modula napona u kontrolisanom PQV čvoru, raste proizvodnja reaktivne snage u odgovarajućem P čvoru, čime se napon u PQV čvoru održava na zadatoj vrednosti.

U radu je korišćena samo jedna metodologija uvažavanja udaljene regulacije napona. U budućim istraživanjima može se analizirati uvažavanje udaljene regulacije napona bez uvođenja novih tipova čvorova.

Proračun Newton-Raphsona metodom bi u tom slučaju pretrpeo promene tako što bi generatorski čvor postao PQ tipa, a udaljeni regulisani čvor PV tipa. Nakon svake iteracije proračuna, reaktivna snaga bi se distribuirala iz PV čvora u PQ čvor uz uvažavanje faktora učešća.

7. LITERATURA

- [1] V. Strezorski, „Analiza elektroenergetskog sistema“, Fakultet tehničkih nauka, Novi Sad, 2011/2012.
- [2] E. Acha, C.R. Fuerte-Esquivel, H. Ambriz-Pérez, C. Angeles-Camacho, „FACTS – Modelling and Simulation in Power Networks“, John Wiley & Sons, LTD, 2004.
- [3] D. Popović, V. Levi, „Extension of load flow model with remote voltage control by generator“, Electric Power Systems Research, Volume 25, Issue 3, 1992.
- [4] M. Čalović, A. Sarić, „Osnovi analize elektroenergetskih mreža i sistema“, Akademski misao, Beograd, 2004.
- [5] N. Qin, „Voltage Control in Future Power Transmission Systems“, PHD thesis, Aalborg University, Aalborg, Denmark, 2016.
- [6] V. Strezorski, „Osnove elektroenergetike-sistemi, Fakultet tehničkih nauka“, Novi Sad, 1996.
- [7] H. Saadat, „Power System Analysis“, Milwaukee School of Engineering, 1999.
- [8] M. Čalović, A. Sarić, P. Štefanov, „Eksplotacija elektroenergetskih sistema u uslovima slobodnog tržišta“, Tehnički fakultet, Čačak.
- [9] M. Čalović, „Regulacija elektroenergetskih sistema; Tom 2: Regulacija napona i reaktivna snaga“, Elektrotehnički fakultet, Beograd, 1997.

Kratka biografija



Milica Brkušanić rođena je u Novom Sadu 1992. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva– Elektroenergetski sistemi odbranila je 2016. godine.



Neven Kovački rođen je u Zrenjaninu 1987. godine. Diplomirao je, magistrirao i doktorirao na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva, redom 2010, 2011. i 2018. godine.

ВИРТУЕЛНА ПАРАМЕТРИЗАЦИЈА ПРОЦЕСНЕ КОНТРОЛЕ У ПОГОНУ ЗА ПАСТЕРИЗАЦИЈУ МЛЕКА**VIRTUAL PARAMETERIZATION OF PROCESS CONTROL IN A MILK PASTEURIZATION FACILITY**

Младен Алексић, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОТЕХНИКА

Кратак садржај – Процесна контрола представља распрострањен начин управљања у индустријским погонима. Имплементација овог вида управљања у производном погону може довести до економских губитака у виду неадекватно обрађеног производа. Кроз овај рад је изложено једно од решења при којим се одговарајући параметри регулације подешавају помоћу виртуелног модела стварног погона са циљем смањења економских губитака. Овакав метод се назива “дигиталним близанцем” и примењен је на погону за пастеризацију млека.

Кључне речи: Процесна контрола, параметризација, PID регулатор, OPC UA комуникација

Abstract – Process control is a common method of regulation in industrial facilities. Implementing this type of control in a production facility can lead to economic losses in the form of inadequately processed products. This paper presents one of the solutions in which the relevant regulation parameters are adjusted using a virtual model of the real facility in order to reduce economic losses. This method is called a „digital twin“ and has been applied in a milk pasteurization facility.

Keywords: Process control, parametrization, PID regulators, OPC UA

1. УВОД

Кроз кратак осврт на историју је приказано како је се прерада обраде млека и млечних производа одвијала некада. Процес аутоматизације је обухватио и ову грану прехранбене индустријске производње. Процес аутоматизације означава процес у коме се све радње извршавају по унапред предодређеним правилима при чему се остварује оптимална ефикасност производног погона. Подешавање опреме којом се врши аутоматизација често може довести до економских губитака у виду неадекватно обрађеног производа који се не може искористити. У ретким случајевима може изазвати и оштећење коришћене опреме. Да би се смањили економски губици и систем подесио за адекватно управљање у што краћем временском периоду примењују се иновативна технолошка решења.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био доц. др Никола Вукајловић.

Ова решења обухватају тестове на детаљним виртуелним моделима стварних система од којих ће кроз овај рад покушано формирање једног таквог иновативног начина методом “дигиталног близанца”.

2. ПРОЦЕСНА КОНТРОЛА**2.1. Дефиниција и значај**

Процесна контрола има велики значај у модерним индустријским постројењима. Имплементира се у одређеним деловима процеса обраде са циљем олакшавања процеса надзора и извршења одређених операција.

Увођењем процесне контроле се постиже да финални производ увек има исте карактеристике.

Основа процесне контроле су задата вредност од стране корисника (енгл. Set Point value - SV), контролна излазна вредност (енгл. Control Variable - CV) и стварна вредност процеса (енгл. Process Value - PV). Приликом реализације процесне контроле мора се водити рачуна на стабилност система, време одзива, време смирења, и грешки у стационарном стању.

Могућа је реализација процесне контроле која се реализује у погону у зависности од природе процеса који се контролише. Могуће је формирати систем који ради у отвореној спрези, затвореној спрези, спрези са “anti-windup” дејством, спрези са “feed-forward” дејством.

2.1. Индустријски примери употребе процесне контроле

Процесна контрола се примењује у индустријским процесима у којима се обављају континуирани циклуси производње без прекида. Заступљена је у производњи електричне енергије, третмана пречишћавања воде, прехранбеној индустрији, производњи папира, фармацеутској и медицинској индустрији, аутомобилској индустрији.

Детаљније је објашњена процесна контрола током производње електричне енергије у нуклеарним електранама и током раздвајања нафте од воде и гаса у сепараторима рафинеријских постројења.

3. ПРОЦЕС ТРЕТИРАЊА И ОБРАДЕ МЛЕКА

3.1. Прикупљање и прерада млека

Процес производње млека започиње на фарми где се мора прикупљати и складиштити у посебним условима све до момента прикупљања. Транспорт се обавља у бурадима или помоћу цистерни које се израђују да поседују већи број преграда. Пријем и складиштење у млекарни се мора спроводити пазећи на одржавање температуре и тестирање сировог млека ради утврђивања квалитета производа као и детекције присутности бактерија. Потом се млеко води кроз делове погона за хомогенизацију, филтрирање, пастеризацију, сепарацију, хлађење, сушење, паковање итд.

3.2. Пастеризација млека

Пастеризација сировог млека се мора вршити да би се из млека уклониле штетне бактерије које у неким случајевима могу изазвати и теже болести. Приликом процеса пастеризације се мора водити рачуна да млеко које се обрађује не изгуби своја хранљива својства. Успешно неутралисање бактерија се може постићи различитим зависностима температуре загревања и времена током којег је ту температуру одржавати. Стандардизовани начини пастеризације су приказани у табели 3.1

Процес	Температура	Временски период
Термизација	63 – 65 °C	15 s
LTLT пастеризација	63 °C	30 min
HTST пастеризација	72 – 75 °C	15 - 20s
УНТ пастеризација	125 – 138 °C	2 – 4 s
УНТ проточна пастеризација	135 – 140 °C	неколико секунди
Стерилизација паковања	115 – 120 °C	20 – 30 min

Табела 3.1 Процеси термичке обраде млека

Током овог рада ће се обрађивати поступак LTLT (енгл. Low Temperature Long Time) и HTST (енгл. High Temperature Short Time) процеса пастеризације.

3.3. Индустриска опрема за пастеризацију млека

За потребе процеса индустриске пастеризације млека неопходни су елементи попут резервоара, измењивача топлоте, агитатора, сензора притиска, сензора протока, сензора температуре, сензора висине од којих су најчешће коришћени приказани током овог рада.

3.4. Опрема моделована у овом раду

Пошто је потребно моделовати пастеризатор који у стварности постоје и користи се за ове потребе бира се резервоар мањег капацитета који је приказан на слици и измењивач топлоте са слике .Моделовани

резервоар је произвођача “Huasheng-Europe“ и има максималну запремину од 100 литара. Приказан је на слици 3.2. У овај резервоар се са стране ушрафљава грејач у виду измењивача топлоте који се загрева помоћу водене паре. Слика овог измењивача је приказана на слици 3.3.



Слика 3.2 “Huasheng-Europe“ пастеризатор



Слика 3.3 Изглед “Armstrong THS-412-2-1” грејача

4. СОФТВЕРСКА РЕАЛИЗАЦИЈА

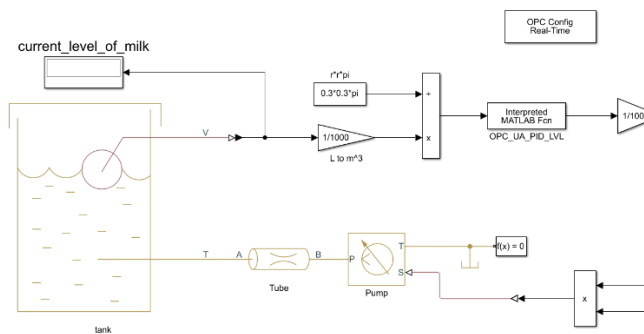
Циљ софтверске реализације је креирати модел “дигиталног близанца”. “Дигиталан близанац” је начин креирања детаљног модела који се преплиће са стварним системом. Тестирањем производа или производног погона током неког временског периода са дигиталним близанцем, може довести до увида о понашању самог производа и система тако што ће се уочене грешке уочити и отклонити.

4.1. Matlab/Simulink модел

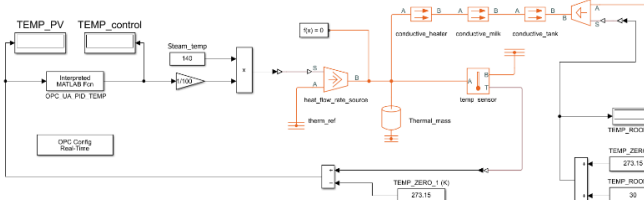
Креирани су и објашњени модели пуњења резервоара и процеса пастеризације у Matlab/Simulink окружењу помоћу Simscape блокова. Модел пуњења резервоара је приказан на слици 4.1 док је модел процеса пастеризације приказан на слици 4.2.

4.2.1 Верификација Matlab/Simulink модела

Претходно пуштању у рад Matlab/Simulink модела претходило је њихово постепено формирање и тестирање од једноставнијих начина управљања попут отворене спреге до финалних модела који ће се користити приликом симулације.



Слика 4.1 Модел пуњења резервоара



Слика 4.2 Модел процеса пастеризације

4.2. Аутоматизација процеса пуњења резервоара и процеса пастеризације млека

Процес управљања врши PLC контролер који се мора адекватно подесити да управља стандардизованим процесима пастеризације и пуњења на адекватан начин. Овај процес је услед употребе Siemens контролера извршен у TIA V16 порталу. Изложени су сви неопходни програмабилни блокови оваквог система управљања као и све потребне функције којима се врши прорачун променљивих вредности.

4.3. Параметрисање функције PID Compact V2

Улогу регулатора током овог процеса преузима Siemens-ов PID Compact V2 блок чије је оптималне параметре потребно пронаћи. Поступак подешавања овог блока се мора изводити на препоручени начин. Овом методом су успешно пронађени параметри који су искоришћени за потребе регулатора загревања код модела пастеризације.

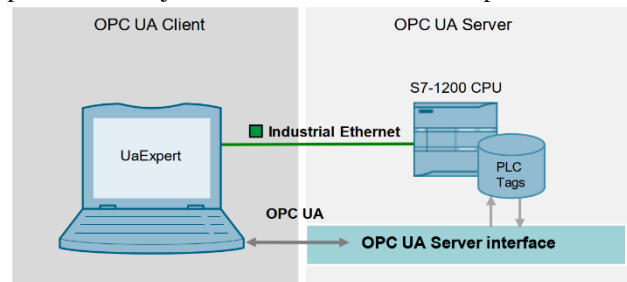
4.3.1 Псеудо-хеуристички приступ подешавању регулатора на примеру пуњења резервоара

Пошто се до оптималних параметара регулатора не може доћи увек пратећи препоруке произвођача или неком од заступљенијих хеуристичких метода попут "Ziegler-Nichols" неопходно је понекад применити псеудо-хеуристичке методе ручног подешавања. Једна оваква метода је описана до које је се дошло пратећи промене током неколико корака подешавања параметара регулатора и тестирањем на моделу пуњења резервоара.

4.4 Комуникација отвореног протокола

На слици 4.3 је блок шема остваривања комуникације OPC UA протоколом коју је потребно остварити да би стварни Siemens-ов контролер и Matlab/Simulink модел могли да успешно размењују податке. Неопходно је извршити подешавање са стране

контролера у виду имплементирања софтверских подешавања чиме се омогућује да он ради као OPC UA Server. Потребно је и имплементирати код са Matlab стране да би модел могао радити као OPC UA Client. Софтвер помоћу кога ће се подаци размењивати је "Unified Automation UaExpert".



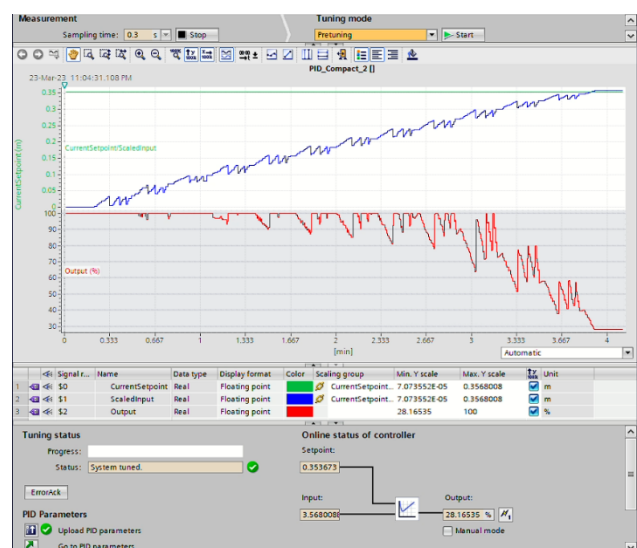
Слика 4.3 Принциуска шема комуникације OPC UA протоколом

5. ЕКСПЕРИМЕНТАЛНА ВЕРИФИКАЦИЈА

Након што је извршено повезивање OPC UA клијента и сервера потребно је извршити експериментално тестирање рада формираног модела са подешеним контролером и верификовати успешност система регулације приликом пуњења резервоара и загревања млека. Комуникација се извршава у реалном времену. За потребе ове верификације формиран су и одговарајући HMI панели са циљем једноставнијег покретања управљања и надзирања процеса од стране корисника.

5.1. Процес пуњења резервоара

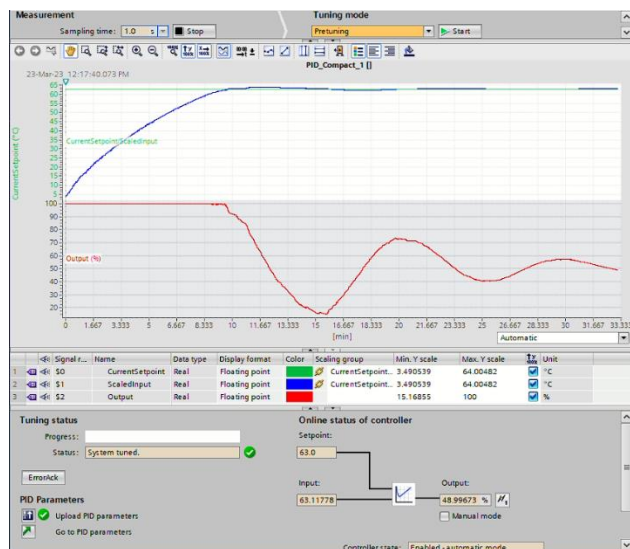
Преносна карактеристика током експерименталне верификације процеса пуњења резервоара је приказана на слици 5.1. Процес је дао задовољавајуће резултате са присутним пребачајем веома мале вредности који не излази из дозвољених граница одступања.



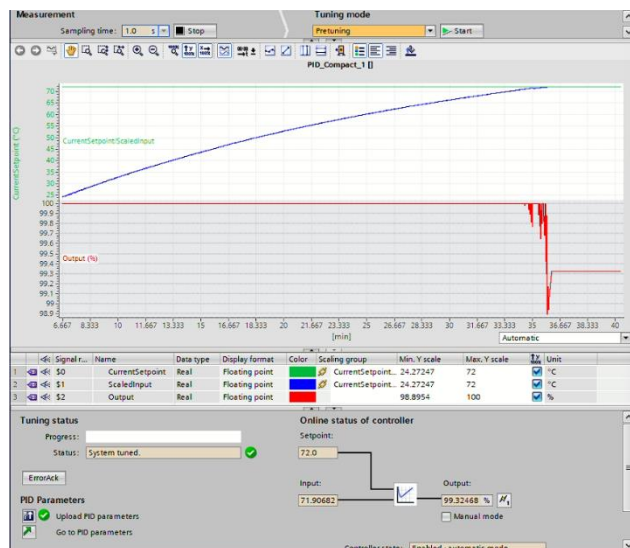
Слика 5.1 Карактеристика процеса пуњења резервоара

5.2. Процес загревања

Преносна карактеристика током експерименталне верификације процеса пастеризације је снимљена за LTLT и HTST процесе пастеризације. Карактеристика LTLT процеса приказана је на слици 5.2 док је карактеристика HTST процеса приказана на слици 5.3. Оба процеса су успешно извршена. LTLT процес је имао благе осцилације током уласка у стационарно стање које су опсегу дозвољених граничних вредности процеса.



Слика 5.2 Карактеристика LTLT процеса пастеризације



Слика 5.3 Карактеристика HTST процеса пастеризације

6. ЗАКЉУЧАК

Након детаљног формирања модела процеса пуњења резервоара и пастеризације млека и успешно је постигнута аутоматизација процеса на контролеру. Изведено је и одговарајуће проналажење параметара PI регулатора што је постигнуто на два различита начина од којих је први препоручен од стране произвођача а други псеудо-хеуристички.

Подешавањем OPC UA конекције остварена је комуникација између модела и контролера у реалном времену и успешно су снимљене преносне карактеристике моделованих процеса.

Даља верификација добијених одговарајућих параметара могла би се извести на реалном физичком систему на коме би се тестирали резултати добијени методом “дигиталног близанца”.

7. ЛИТЕРАТУРА

- [1] Gösta Bylund, “Dairy processing handbook Tetrapack”, Tetra Pak Processing Systems AB, 1995. Доступно на: <https://dairyprocessinghandbook.tetrapak.com>
- [2] Siemens - Digitalization with TIA Portal: Virtual Commissioning with SIMATIC and Simulink - Use case 2: Connecting Simulink models to SIMATIC PLCSIM Advanced via OPC UA. “109749187_DIGI_Usecases_OPC_DOC_V20_en”. Септембар 2023. Доступно на: <https://support.industry.siemens.com/cs/document/109749187/digitalization-with-tia-portal-virtual-commissioning-with-simatic-and-simulink?dti=0&lc=en-RS>
- [3] Siemens - Closed-Loop Control with "PID_Compact" V2.2. “79047707_PidCompactV2_2_DOC_V2_0_1_en”. Септембар 2023. Доступно на: https://cache.industry.siemens.com/dl/files/707/79047707/att_915339/v2/79047707_PidCompactV2_2_DOC_V2_0_1_en.pdf
- [4] “Armstrong fluid technology” tank heater. Септембар 2023. Доступно на: <https://armstrongfluidtechnology.com/en/products/tank-heaters>
- [5] “Huasheng-Europe” pasteurizers. Септембар 2023. Доступно на: <https://www.huasheng-europe.com/product/calf-milk-pasteurizers>

Кратка биографија:



Младен Алексић рођен је 1995. године у Ужицу. Основне студије Енергетске електронике и електричних машина је завршио 2022. године након чега уписује мастер студије на истом усмерењу.

Контакт: mladena217@gmail.com

**ДЕЦЕНТРАЛИЗОВАНИ СОФТВЕРСКИ СИСТЕМ ЗА ЗАШТИТУ ДИГИТАЛНОГ
САДРЖАЈА ПРИМЈЕНОМ КОНЦЕПТА BLOCKCHAIN ТЕХНОЛОГИЈЕ****DECENTRALIZED SOFTWARE SYSTEM FOR SECURING DIGITAL ASSETS USING
THE CONCEPTS OF BLOCKCHAIN TECHNOLOGY**

Марија Кљештан, Факултет техничких наука, Нови Сад

Област – РАЧУНАРСТВО И АУТОМАТИКА**2. ТЕОРИЈСКЕ ОСНОВЕ**

Кратак садржај – У овом раду описана је имплементација децентрализованог система који омогућава корисницима да обезбједе свој дигитални садржај креирајући незамјенљиви токен на Ethereum платформи, а чији метадаци су смјештени на IPFS дистрибуираном систему. Поред тога, описани су основни концепти blockchain технологије.

Кључне ријечи: децентрализовани системи, блокчејн, Ethereum, незамјенљиви токени

Abstract – This paper describes the implementation of a decentralizes system that allows users to secure their digital content by creating NFT on Ethereum blockchain, which metadata is stored on IPFS. Also, this paper gives introduction to the fundamentals of distributed systems and blockchain technology.

Keywords: decentralized systems, blockchain, Ethereum, non-fungible tokens

1. УВОД

Blockchain технологија стекла је велику популарност након увођења биткоина 2009. године. Још од тада, много корисника посматра блокчејн технологију искључиво из перспективе биткоина и система плаћања.

Међутим, blockchain као дистрибуирана дијељена главна књига, која обезбјеђује интегритет и непромјенљивост сачуваних података, без учешћа посредничких страна, нашла је широку примјену како у области финансија, пољопривреде, здравства, туризма, образовања и сличних области, тако и у умјетности, гдје су од посебне важности незамјенљиви токени [1].

Овај рад бави се безбједним чувањем и утврђивањем власништва дигиталог садржаја. Чување се врши на IPFS дистрибуираном систему, а историја власништва је трајно сачувана на Ethereum blockchain-у употребом незамјенљивих токена. У раду су детаљно описане теоријске основе свих концепата примијењених у имплементацији система.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био др Горан Сладић, ред. проф.

2.1. Blockchain

Blockchain је дистрибуирана, дијељена главна књига (енг. ledger) која складишти трансакције и која не може бити измијењена након што се трансакција верификује и дода у књигу [1]. Трансакције се групишу у блокове, при чему је свака трансакција обезбјеђена криптографским методама и валидирана од стране сваког овлашћеног члана мреже, коришћењем консензус алгоритама, тј. скупа правила за валидацију. Трансакција коју нису потврдили сви чланови мреже се не додаје у базу података. Свака трансакција је везана за претходну трансакцију у секвенцијалном редослиједу, стварајући ланац трансакција, односно блокова. Трансакција се не може избрисати или измијенити, једини начин за измјену је додавање друге трансакције у ланац. Осим тога, у blockchain системима елиминише се потреба за постојањем централизованог ентитета, који имају контролу над цијелим системом и којем сви морају вјеровати [2].

2.2. Ethereum

Ethereum представља децентрализовани систем који покреће рачунар под називом Ethereum Виртуелна Машина (EVM). Свако ко учествује у Ethereum мрежи (сваки Ethereum чвор) чува копију стања овог рачунара. Поред тога, сваки учесник може да емитује захтјев да овај рачунар изврши произвољно израчунавање. Кад год се такав захтјев емитује, други учесници на мрежи верификују, потврђују и извршавају прорачун. Ово извршење изазива промјену стања у EVM-у, која се пропагира кроз цијелу мрежу. Захтјеви за израчунавање се називају захтјеви за трансакције, а евиденција о свим трансакцијама и тренутном стању EVM-а се чува на blockchain-у [3].

Сваки рачунар (чвор) у мрежи мора да се сложи око додавања сваког новог блока у ланац. Чворови осигуравају да сви који ступају у интеракцију са blockchain-ом имају исте податке. Како би се ово постигло, Ethereum користи Proof of Stake (PoS) консензус механизам [4].

Ether (Eth) је криптовалута која се користи на Ethereum мрежи.

Ethereum стање састоји се од објеката који се зову налози. Налог представља мапирање између адресе и стања налога. То је ентитет који посједује Ether (Eth)

биланс и стога може да обавља трансакције на *Ethereum* мрежи. Постоје 2 типа налога - налози који су контролисани од стране корисника и налози паметних уговора [5].

2.3. Паметни уговори

Увођењем паметних уговора, *Ethereum* платформа је омогућила корисницима да поред обављања трансакција и трговања криптовалутама, користе и чувају незамјенљиве токени, креирају децентрализоване апликације и још много тога због чега се *Ethereum* сматра представником 3. генерације веба [3].

Паметни уговори су рачунарски програми који се чувају на *blockchain*-у и извршавају се на *Ethereum* виртуелној машини. Сваки паметни уговор се састоји од кода који специфицира унапријед одређене услове који, када се испуне, покрећу њихово извршавање. Покретањем на децентрализованој платформи, умјесто на централизованом серверу, паметни уговори омогућавају да више страна дођу до заједничког резултата на прецизан, благовремен и безбједан начин, без учешћа централног администратора [6].

Паметни уговори се углавном пишу у програмским језицима високог нивоа, од којих је најпопуларнији *Solidity*. С обзиром на чињеницу да *EVM* може да извршава програме написане у језицима ниског нивоа, паметни уговори писани у језицима вишег нивоа преводе се у *EVM Bytecode* који посједује посебан скуп инструкција за интерпретирање кода паметних уговора.

2.4 Незамјенљиви токени

Попут физичког новца, криптовалуте су обично замјенљиве, што значи да се њима може трговати и да се могу размјењивати једна за другу. Такође, могу се дијелити на мање фракције.

Незамјенљиви токени (енг. *Non-Fungible Token, NFT*) мијењају крипто парадигму чинећи сваки токен јединственим и незамјенљивим, односно чинећи немогућим да један незамјенљив токен буде „једнак“ другом токenu. Они су дигитална репрезентација средстава и могу се упоредити са дигиталним пасошима, јер сваки токен садржи јединствени, непреносиви идентитет који га разликује од других токена. Осим тога, ови токени су недјелјиви, односно не могу се подијелити на мање фракције [7].

NFT-ови су направљени у складу са стандардом *ERC-721 (Ethereum Request for Comment#721)*, који диктира како се власништво токена преноси, методе за валидацију трансакција и како апликације рукују безбједним трансферима токена [7].

Поменути стандард се односи и на метаподатке токена. Када је ријеч о складиштењу метаподатака, они се попут самог токена, могу складиштити на *blockchain*-у. Међутим, у том случају долази до проблема перформанси и складиштења података *blockchain* система. Због тога се метаподаци чешће складиште ван *blockchain*-а (енг. *offchain*) на

централизованим серверима или дистрибуираним системима, као што је *IPFS*, који је кориштен и у овом раду [8].

2.5 Децентрализоване апликације и Web 3.0

Децентрализована апликација (*DApp*) је апликација изграђена на децентрализованој мрежи, која комбинује паметне уговоре и кориснички интерфејс [9].

Децентрализована апликација има свој позадински код, у виду паметних уговора, који се извршавају на децентрализованој мрежи, насупротив традиционалним апликацијама, у којима се позадински код извршава на централизованим серверима.

DApp може да има клијентску страну написану на било ком језику (као и у случају централизоване апликације) која упућује позиве свом позадинском дијелу.

Уводећи децентрализацију у све аспекте веб апликација, као што су складиштење, размјена порука и слично, децентрализоване апликације су преусмјериле развој веба. Термин који се користи да опише нови правац у еволуцији веба јесте *web3*, означавајући при томе трећу "верзију" веба [6].

2.6 IPFS

IPFS (енг. *Inter Planetary File System*) је *peer-to-peer (P2P)* дистрибуирани систем за складиштење, приступ и дијелење датотека, вебсајтова, апликација и података [8].

Кључна разлика између централизованог и децентрализованог веба је у томе како се подаци идентификују и преузимају. У централизованом вебу, корисници зависе од поузданих ентитета који чувају њихове податке и приступају им помоћу јединствених локатора ресурса (*URL*-ова) заснованих на локацији.

Насупрот томе, *IPFS* мрежа користи систем адресирања садржаја, гдје сам садржај има кључну улогу у омогућавању корисницима да пронађу оно што траже. У *IPFS*-у, сваки дио садржаја је идентификован јединственим хешом који се зове *CID (Content Identifier)*. То значи да се садржај чува и преузима на основу његовог хеша, а не његове локације, што га чини много тежим за цензурисање или манипулацију.

3. МОДЕЛ СИСТЕМА

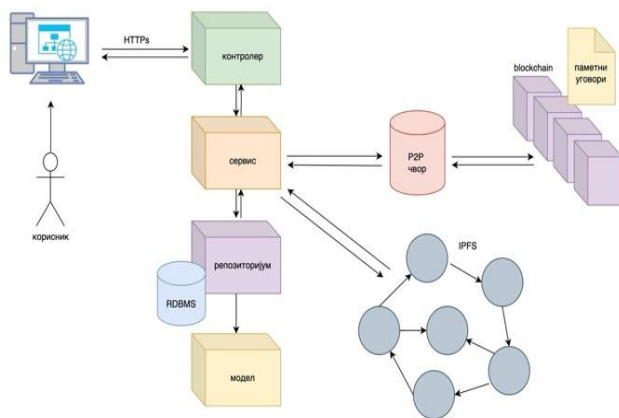
Намјена система којим се бави овај рад јесте да омогући корисницима да сачувају свој дигитални садржај од вриједности на сигуран начин, тако што корисник путем корисничког интерфејса унесе свој дигитални садржај у облику датотеке, која се чува на дистрибуираном систему, а затим се креира *NFT* позивом паметног уговора који се извршава на *blockchain* мрежи, као и да корисник има преглед својих *NFT*-ова везаних за одређени дигитални новчаник.

Систем је развијен као децентрализована веб апликација, са одвојеним клијентским и серверским дијелом, а дијаграм њене архитектуре представљен је на сл. 1.

Клијентска страна представља апликацију која се извршава у прегледачу (енг. *browser*). Серверску страну чини монолитна апликација која се састоји од модула, тако да подржава вишеслојну архитектуру.

Ова апликација комуницира са *IPFS* мрежом, ради складиштења датотека које је корисник унио, као и са ентитетима *blockchain* мреже, како би позивала функције паметног уговора за креирање и добављање корисникових *NFT*-ова.

Иако садржи елементе који су карактеристични за традиционалне веб апликације, функционалности које су од суштинског значаја као што су складиштење дигиталног садржаја и управљање *NFT*-овима, имплементирани су примјеном принципа *blockchain* технологије и дистрибуираних система, чиме се ова апликација може сматрати децентрализованим софтверским рјешењем.



Слика 1. Дијаграм архитектуре система

4. ИМПЛЕМЕНТАЦИЈА

За имплементацију клијентске стране кориштена је *React.js* библиотека, заједно са *ether.js* скупом библиотека које омогућавају интеграцију са дигиталним крипто новчаницима, као што је *MetaMask*, новчаник уграђен у веб претраживач, а омогућава да корисници повежу свој *Ethereum* налог са налогом на систему и на тај начин уз пар корака креирају нове *NFT*-ове.

Серверску страну апликације чине паметни уговори који су писани у *Solidity* програмском језику, који представља објектно оријентисани програмски језик, специфично намијењен за имплементацију паметних уговора. За развој и поставку паметних уговора на *Ethereum* мрежу коришћен је *Truffle* радни оквир. Паметни уговори су уз помоћ овог окружења постављени на *Sepolia* тестну *Ethereum* мрежу.

За складиштење докумената и *NFT* метаподатака кориштен је *IPFS* чиме је обезбјеђено *offchain* складиштење података, које доприноси очувању ефикасности и перформанси *blockchain* мреже. Комуникација са *IPFS* и блокчејн мрежом врши се преко монолитне веб апликације имплементираних у *Golang* програмском језику.

4.1 Повезивање са *blockchain* мрежом

Како би могле да користе услуге *Ethereum* платформе, децентрализоване апликације морају да се повежу на *peer-to-peer* мрежу, што може довести до успореног развоја апликације још у почетној фази развоја. Када би се покретао нови чвор у мрежи, његова синхронизација са *Ethereum blockchain*-ом би могла потратајати сатима, па чак и данима.

Ове проблеме рјешава *Infura* сервис, платформа која нуди инфраструктуру и алате потребне за брже, лакше и ефикасније повезивање децентрализованих апликација са *Ethereum* мрежом. Употребом тих алата, развојни тимови не морају да брину о синхронизацији или сложенем конфигурисању чворова како би се повезали на *Ethereum* мрежу.

Ethereum API који пружа *Infura* нуди инстант приступ *Ethereum blockchain* мрежи преко *WebSocket*-а и *HTTPS*-а, што представља једну од највећих предности употребе овакве платформе.

Поред главне *Ethereum* мреже (енг. *Mainnet*), постоје и тестне мреже. Ове мреже користе углавном развојни тимови приликом имплементације паметних уговора, како би их тестирали у окружењу сличном продукцијом, прије постављања на главну мрежу.

Sepolia мрежа је препоручена подразумијевана тестна мрежа за развој апликација и кориштена је приликом имплементације овог система.

4.2 Паметни уговор *NFT*

За поставку овог паметног уговора на *Sepolia* тестну мрежу, кориштен је *Truffle* радни оквир, који омогућава бржи развој и поставку паметних уговора.

С обзиром да се ради о *NFT* колекцији, за основне операције над токенима које су прописане стандардом, наслијеђени су паметни уговори из *OpenZeppelin* библиотеке, што се може видјети на листингу 1.

```
contract NFT is ERC721URIStorage, Ownable {
    using Counters for Counters.Counter;
    Counters.Counter private tokenIds;

    constructor() ERC721("NFT", "NFT collection") {}

    function mintNFT(address recipient, string memory tokenURI)
        public onlyOwner
        returns (uint256)
    {
        _tokenIds.increment();
        uint256 newItemId = _tokenIds.current();
        _mint(recipient, newItemId);
        _setTokenURI(newItemId, tokenURI);
        return newItemId;
    }
}
```

Листинг 1. Функција за генерисање *NFT*-а

Функција *mintNFT* представљена на листингу 1 служи за креирање новог токена и има 2 параметра. Први

параметар је *Ethereum* адреса налога који је иницирао трансакцију за креирање токена, а други параметар представља адресу метаподатака *NFT*-а, смјештених на *IPFS*-у. Приликом креирања токена, користи се бројач који се инкрементира за 1, а затим се позива `_mint` функција која генерише токен у власништву корисника са прослијеђеном адресом. Затим се новокреирани токен повезује за адресом метаподатака позивом функције `_setTokenURI`.

4.3 Интеграција са крипто новчаником

MetaMask је дигитални крипто новчаник који представља интерфејс за интеракцију са чворовима *blockchain* мреже. Подразумијevano се повезује на *Infura* чвор, али се може повезати и на чвор који је покренут локално.

Приликом иницирања трансакције за креирање *NFT*-а, апликацији је потребна адреса корисниковог *Ethereum* налога, како би се одредило у чијем власништву је креирани токен на *Ethereum* мрежи. Најбезбједнији, и за корисника најједноставнији, начин добављања његове адресе јесте увезивање апликације са *MetaMask* новчаником корисника.

JavaScript библиотека која омогућава интеграцију децентрализованих апликација са *Ethereum* мрежом јесте *Ether.js*. Ова библиотека омогућава и интеграцију апликација са крипто новчаницима. На листингу 2 представљен је дио кода из клијентске апликације који врши добављање тренутног биланса корисниковог налога.

```
function connectToMetaMaskBtnHandler() {
  if (window.ethereum) {
    window.ethereum
      .request({ method: "eth_requestAccounts" })
      .then((res) => {
        accountChangeHandler(res[0])
      });
  } else {
    alert(" please, install metamask extension!!");
  }
}

const getbalance = (address) => {
  window.ethereum
    .request({
      method: "eth_getBalance",
      params: [address, "latest"]
    })
    .then((balance) => {
      setData({
        Balance: ethers.utils.formatEther(balance),
      });
    });
};
```

Листинг 2. Интеграција са *MetaMask* новчаником

5. ЗАКЉУЧАК

Blockchain технологија представља радикалан и нови, а притом безбједан и транспарентан начин обављања свих врста трансакција преко интернета, свима

доступног медија за пренос података. Појава *bitcoin*-а и *blockchain*-а довела је до многих промјена у свијету финансија, али и пољопривреде, медицине, туризма, логистике, спорта, па чак и умјетности.

NFT-ови, или незамјенљиви токени, представљају токене засноване на *blockchain* технологији, који омогућавају дигиталну репрезентацију физичких или дигиталних средстава. Ова релативно нова технологија пружа могућност праћења власништва над имовином и провјере њене аутентичности, омогућавајући већу транспарентност. Као резултат тога, имовином се може релативно лако трговати, а процес преноса власништва је неупоредиво бржи због изостанка посредничких страна.

Овај рад се бавио примјеном концепата *blockchain* технологије и незамјенљивих токена за рјешавање проблема безбједног и ефикасног чувања дигиталних средстава корисника. Поред *blockchain*-а, кориштена је и сродна технологија заснована на концептима дистрибуираних система - *IPFS*.

Рад може да послужи као основа за упознавање са концептима *blockchain* технологије, незамјенљивих токена, децентрализованих апликација изграђених на *Ethereum* платформи, али и система попут *IPFS*-а.

6. ЛИТЕРАТУРА

- [1] H. Natarajan, S. Krause, H. Gradstein, „*Distributed Ledger Technology and Blockchain*“, *FinTech Note*, 2017.
- [2] R. Shaan, „*Blockchains: The technology of transactions*“, Towards Data Science, 2018.
- [3] D. Vujicic, D. Jagodic, S. Randic, „*Blockchain technology, bitcoin, and Ethereum: A brief overview*“, 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), pp. 1-6, 2018.
- [4] J. Wi, „*Blockchain without Waste: Proof-of-Stake*“, The Review of Financial Studies, т. 34, бр. 3, pp. 1156-1190, 2021.
- [5] <https://ethereum.org/en/developers/docs/nodes-and-clients/> (приступљено у септембру 2023.)
- [6] V. Buterin, „*A next-generation smart contract and decentralized application platform*“, white paper, т. 3, бр. 37, 2014.
- [7] R. Sharma, „*Non-Fungible Token (NFT): What It Means and How It Works*“, 2023.
- [8] J. Benet, „*IPFS - Content Addressed, Versioned, P2P File System*“, Cornell University, 2014.
- [9] <https://ethereum.org/en/developers/docs/dapps/> (приступљено у септембру 2023)

Кратка биографија:



Марија Кљештан рођена је у Лозници 1999. године. Мастер рад на Факултету техничких наука из области Рачунарство и аутоматика – Примењене рачунарске науке и информатика одбранила је 2023. године.

контакт: marijakljestan@gmail.com

PRIMENA UDS PROTOKOLA U AUTOMOBILSKOJ INDUSTRIJI

APPLICATION OF UDS PROTOCOL IN AUTOMOTIVE INDUSTRY

Pavle Janjić, Zoran Ivanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U radu je predstavljen UDS protokol kao primarno sredstvo dijagnostičke komunikacije u savremenoj automobilske industriji. Objašnjen je princip rada UDS protokola i dat pregled najbitnijih servisa koje sadrži. Takođe, predložen je ISO transportni protokol kao podloga za dijagnostičku komunikaciju na transportnom nivou. U praktičnom delu data je potvrda pomenutih koncepata kroz konkretan primer dijagnostičke komunikacije izveden u Canoe programskom okruženju.

Ključne reči: UDS protokol, Dijagnostika, ISO transportni protokol, Automobilna industrija

Abstract – This paper introduces the UDS protocol as the primary method of diagnostic communication in the automotive industry. It describes the operational approach of the UDS protocol and the most important diagnostic services contained within it. Additionally, the ISO transport protocol is presented as the foundation for diagnostic communication at the transport level. The practical part involves the verification of the aforementioned through an example of diagnostic communication created in the Canoe environment.

Keywords: UDS protocol, diagnostics, ISO transport protocol, Automotive industry

1. UVOD

Od svih tehničkih inovacija u automobilske industriji najznačajniju predstavlja kompjuterizacija automobilske komponente. Tokom godina, današnji moderni automobili evoluirali su od čisto mehaničkih naprava do naprednih sistema koji se u mnogome oslanjaju na softver kako bi mogli pravilno da funkcionišu. Počevši od sedamdesetih godina prošlog veka broj elektronskih uređaja unutar jednog vozila je neprestano rastao od jednog centralnog računara do oko 100 elektronskih kontrolnih jedinica (engl. *electronic control unit* - ECU) koliko poseduje jedan moderan današnji automobil. Sve kontrolne jedinice zajedno sa svojim senzorima, međusobno povezane preko CAN (engl. *controller area network*) mreže, kontrolišu većinu podsistema unutar vozila, te se njihov skup može smatrati i „mozgom“ jednog današnjeg modernog automobila.

Kako bi se osiguralo da celokupan sistem funkcioniše bez grešaka neophodne su redovne dijagnostičke provere.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Zoran Ivanović, vanr. prof.

Dijagnostika vozila podrazumeva metodologiju uključenu u identifikaciju i evaluaciju problema koji mogu negativno uticati na normalan rad vozila. Ona može da obuhvata širok spektar tehnika i alata u rasponu od površnih fizičkih provera do složenijih, kompjuterski zasnovanih analiza. S obzirom na ponekad skrivenu prirodu problema, efikasna dijagnostika je ključna za uspešne popravke i nesmetan rad svakog vozila. Sa usložnjavanjem kompjuterskih sistema u vozilima broj potencijalnih problema se eksponencijalno povećavao, ali samo njihovo otkrivanje je postalo značajno jednostavnije. Svako današnje vozilo dolazi opremljeno sa dijagnostičkim konektorima, te mehaničar uz pomoć odgovarajućeg instrumenta može da prikupi podatke o greškama iz svih kontrolnih jedinica u automobilu. Redovne dijagnostičke kontrole omogućavaju preventivno otkrivanje problema te se na ovaj način često mogu izbeći kvarovi na putu, ugrožavanje života putnika i skupe popravke.

Usled usložnjavanja procesa dijagnostike, stanje vozila tokom dijagnostičkih provera postaje sve značajnije. Različiti dijagnostički parametri mogu imati drugačije vrednosti u zavisnosti od toga da li je vozilo u pokretu ili miruje u garaži. Kako bi se osigurala maksimalna bezbednost putnika i ispravan rad u svakom trenutku dijagnostiku možemo podeliti na dve grane:

- Dijagnostika na tabli (engl. *on-board diagnostic*)
- Dijagnostika van table (engl. *off-board diagnostic*)

2. OBD PROTOKOL

Za potrebe standardizacije načina komunikacije između dijagnostičkog alata i samog vozila nastao je OBD protokol koji definiše izgled i sadržaj poruka kojima se komunikacija odvija kao i sam tip konektora koji se koristi. Uprkos svim dobrim stranama, nedostatak OBD2 protokola predstavlja njegova fokusiranost na pogon i emisije te pomoću njega nije moguće očitati korisne podatke iz ostalih kontrolnih jedinica kao što su status sigurnosnog pojasa ili vazdušnih jastuka. Takođe, protokol dozvoljava samo očitavanje podataka, ali nije omogućeno njihovo upisivanje, te ne može biti od pomoći pri samom otklanjanju greške nego isključivo prilikom detektovanja.

3. UDS PROTOKOL

Usled povećanja broja ECU-a koji se ne odnose na pogon i emisije, nedostaci OBD2 protokola postaju sve očigledniji. Porastom broja problema kojih kontrolne jedinice koje ne pripadaju pomenutoj grupi mogu izazvati, nastaje potreba za standardizovanjem načina njihove

komunikacije sa dijagnostičkim alatima. Zbog toga, internacionalna organizacija za standardizaciju – ISO (engl. *International organization for standardization*) razvija komunikacioni protokol namenjen dijagnostičkoj analizi rada i konfiguraciji kontrolnih jedinica u automobilske industrije nazvan Ujedinjeni dijagnostički servisi – UDS (engl. *Unified Diagnostic Services*). Sam UDS protokol specificiran je u ISO14229 standardu i predstavlja protokol višeg reda zasnovan na klijent – server topologiji, gde je u ulozi klijenta dijagnostički alat povezan najčešće na standardizovani OBD2 konektor u vozilu.

Čitava komunikacija zasniva se na principu zahteva od strane klijenta (mehaničar) i odgovora od strane servera (ECU). UDS standard definiše: sve dostupne servise, formate poruka zahteva, formate poruka odgovora, vremenske parametre tokom komunikacije, kao i sam način korišćenja svih definisanih servisa. Propisan je standardan format i funkcija ali ne i način na koji implementacija mora biti izvršena te ga iz tog razloga mnogi nazivaju i UDS interfejsom.

Za razliku od OBD2 protokola koji omogućava isključivo mogućnost iščitavanja podataka, UDS obezbeđuje takođe i interfejs za njihov upis. Pored standardnih opcija pronalaženja i analize grešaka sačuvanih u EEPROM memoriji mikrokontrolera UDS nudi i niz drugih značajnih mogućnosti kao što su:

- Transfer velike količine podataka. Najčešće se koristi u slučaju kada je potrebno spustiti novi softver na ECU
- Kontrola ulaza i izlaza mikrokontrolera
- Pokretanje predefinisanih rutina
- Čitanje brojnih podataka iz memorije mikrokontrolera i upis podataka u istu

3.1. Format dijagnostičkog zahteva i odgovora

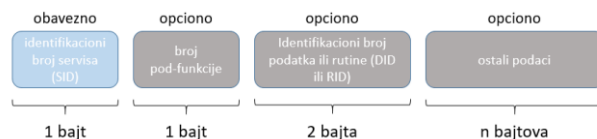
UDS protokol predstavlja kolekciju servisa od kojih svaki poseduje svoj jedinstveni identifikacioni broj (engl. *service identifier* - SID). Identifikacioni broj je veličine jednog bajta i uvek se nalazi na početku zahteva.

Nakon identifikacionog broja servisa, drugi bajt zahteva namenjen je za pod-funkciju (engl. *sub-function*). Njime se određuje koju tačno funkcionalnost datog servisa je neophodno izvršiti. Pojedini servisi, ne poseduju nijednu pod-funkciju te je samim tim ovo polje opciono u formatu zahteva.

Kako UDS protokolom klijent i server mogu da komuniciraju samo putem brojeva svakom podatku koji klijent želi pročitati ili upisati dodeljen je određen identifikacioni broj. Taj broj se naziva identifikacioni broj podatka iliti skraćeno DID (engl. *data identifier*), veličine je dva bajta i u samom zahtevu nalazi se posle pod-funkcije.

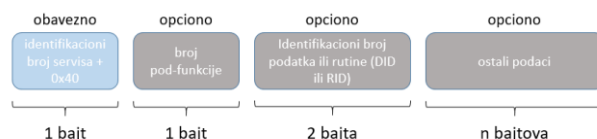
Slično DID-ovima, ukoliko je zahtevan servis za kontrolu rutine, kako bi se specificiralo koja tačno rutinu je potrebno izvršiti, koristi se identifikacioni broj rutine ili skraćeno RID (engl. *routine identifier*). Pošto se DID-ovi i RID-ovi ne koriste u svakom servisu oni predstavljaju opcione bajtove u poruci zahteva.

Nakon DID-a ili RID-a u poruci zahteva se nalaze svi ostali podaci (engl. *data record*). Šta će se nalaziti u ovom polju kao i sama njegova veličina i postojanje zavisi od servisa do servisa. U skladu sa tim i samo polje je opciono.



Slika 1. Format dijagnostičkog zahteva

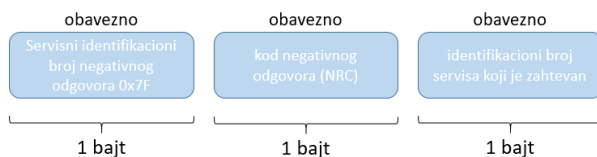
Po pristizanju zahteva ECU najpre proverava njegovu validnost (da li je podržan dati servis ili pod-funkcija, da li je zahtev odgovarajuće dužine, da li su svi preduslovi za ispunjenje zahteva ispunjeni itd.). U slučaju da su sve provere uspešno sprovedene server prelazi na izvršavanje zahteva. U trenutku kada je zahtev uspešno izvršen server vraća testeru pozitivan odgovor. Prvi bajt pozitivnog odgovora sadrži servisni identifikacioni broj odgovora koji se preračunava uvećavanjem identifikacionog broja servisa zahteva za 0x40 (šesti bit setovan). Ukoliko polje pod-funkcije postoji u zahtevu, na mestu drugog bajta će se naći broj pod-funkcije koja je zahtevana (identična vrednost kao u zahtevu). Nakon pod-funkcije slede DID ili RID (takođe opciona polja u zavisnosti od njihovog postojanja u zahtevu). Na kraju dolazi opcioni prostor namenjen za ostale podatke čije se postojanje, veličina i upotreba razlikuje od servisa do servisa.



Slika 2. Format pozitivnog odgovora

Ukoliko bilo koja od provera zahteva nije bila uspešna ili ukoliko server iz nekog razloga nije u mogućnosti da izvrši dati zahtev uslediće negativan odgovor. Svaki negativan odgovor se sastoji od 3 bajta i započinje sa predefinisanim vrednošću servisnog identifikacionog broja negativnog odgovora 0x7F.

Nakon njega sledi kod negativnog odgovora iliti skraćeno NRC (engl. *negative response code*) veličine jednog bajta. On uzima jednu od niza predefinisanih vrednosti u zavisnosti od razloga zbog koga zahtev nije mogao da bude ispunjen. Uz pomoć njega klijent može naslutiti zbog čega je došlo do greške. Vrednosti svih NRC-ova i njihova konkretna namena definisani su u ISO 14229-1 dokumentu [1]. Poslednji bajt negativnog odgovora namenjen je za identifikacioni broj servisa koji nije uspešno izvršen (identična vrednost kao u poruci zahteva).



Slika 3. Format negativnog odgovora

3.2. Dijagnostički servisi

UDS standard u sebi sadrži definisan veliki broj servisa [1] među kojima se kao najbitniji ističu:

- *Diagnostic Session Control* (SID 0x10) – Servis namenjen promeni dijagnostičkih sesija. Pod pojmom sesije se podrazumeva stanje ECU-a koje određuje šta je dostupno za dijagnostiku. Određene dijagnostičke funkcionalnosti dostupne su samo u pojedinim sesijama.
- *ECU Reset* (SID 0x11) - Servis namenjen izvršavanju reseta mikrokontrolera. Željena vrsta reseta se odabira uz pomoć pod-funkcije zahteva.
- *Security Access* (SID 0x27) – Servis namenjen otključavanju potrebnog nivoa sigurnosnog pristupa. Čitav sigurnosni koncept je zasnovan na principu semena (eng. *seed*) i ključa (eng. *key*), a željeni nivo sigurnosnog pristupa se odabira vrednošću pod-funkcije.
- *Read Data By Identifier* (SID 0x22) – Servis namenjen iščitavanju podataka iz memorije. Traženi podatak je određen vrednošću DID-a prosleđenog u zahtevu.
- *Write Data By Identifier* (SID 0x2E) - Servis namenjen upisu podataka u memoriju. Podatak koji će biti upisan odabira se vrednošću DID-a prosleđenog u zahtevu.
- *Routine Control* (SID 0x31) – Servis namenjen pokretanju, zaustavljanju i prikupljanju rezultata rutina. Rutina u smislu dijagnostike predstavlja predefinisani niz instrukcija kojim se izvršava određena funkcionalnosti.
- *Read DTC Information* (SID 0x19) – Servis namenjen iščitavanju različitih informacija o sačuvanim kodovima greški (eng. *diagnostic trouble code* - DTC) unutar mikrokontrolera.

Celokupan aplikativni sloj UDS-a, uključujući sve prethodno navedene servise, može se ravnopravno i neizmerno koristiti bez obzira na protokol nižeg reda koji je zastupljen na fizičkom nivou. S obzirom na rasprostranjenost primene nastavak rada će biti fokusiran na primenu UDS-a koja se oslanja na CAN mrežu na fizičkom nivou.

4. ISO TRANSPORTNI PROTOKOL

Ograničenje u maksimalnoj dužini poruke predstavlja značajan nedostatak CAN komunikacije. Ponajviše u pogledu same dijagnostike, pojedini zahtevi mogu značajno premašivati maksimalnih 8 ili 64 bajta koliko može da sadrži jedna CAN ili CAN FD poruka. Kako bi se ovaj nedostatak nadomestio međunarodna organizacija za standardizaciju u svom ISO 15765-2 daje predlog transportnog protokola (ISO TP) namenjenog više-paketnom prenosu podataka veličine do 4095 bajtova.

Kako bi komunikacija putem transportnog protokola bila moguća u deo CAN poruke namenjen smeštanju podataka, a pre samih podataka, umetnuto je zaglavlje koje sadrži dodatne protokolne informacije. ISO transportni protokol [2] razlikuje 4 različita tipa poruka:

- Pojedinačna poruka (eng. *single frame*)
- Prva poruka (eng. *first frame*)

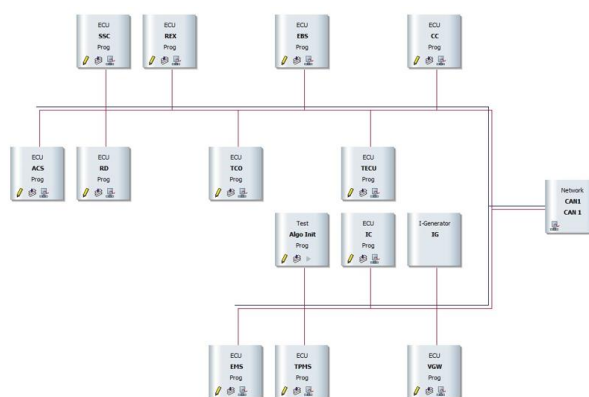
- Uzastopna poruka (eng. *consecutive frame*)
- Poruka za kontrolu toka (eng. *flow control frame*)

Ukoliko je potrebno poslati podatke dužine do 7 bajtova za to će biti korištena pojedinačna poruka. Sa druge strane, ako je potrebno poslati podatke duže od 7 bajtova koristi se više-paketni prenos. Njega otpočinje pošiljalac slanjem prve poruke u kojoj naglašava veličinu podataka koje želi poslati. Primalac je dužan da odgovori sa porukom kontrole toka u kojoj će obavestiti pošiljaoca da li je spreman za prijem i proslediti potrebne komunikacione parametre. Nakon prijema potvrde dalji prenos se nastavlja uzastopnim porukama.

5. CANOE SOFTVERSKI ALAT

U današnje doba, elektronske kontrolne jedinice preuzimaju vodeću ulogu u upravljanju radom jednog modernog vozila te u skladu sa tim od uspešnog rada njihovog softvera umnogome zavisi ugodnost tokom vožnje, a u nekim slučajevima i život samih putnika. Zbog toga je nad svakim ECU-om neophodno sprovesti čitav niz provera kako bi se osiguralo da sve funkcioniše kako što je i planirano. Kako bi rezultati bili merodavni potrebno je ECU dovesti u okruženje što približnije okruženju u stvarnom vozilu. Rešenje pomenutog problema nudi kompanija Vector, sa svojim programskim alatom nazvanim Canoe [3]. Uz pomoć Canoe-a moguće je simulirati kompletnu komunikacionu mrežu jednog vozila, a samim tim i realno okruženje ECU-a za potrebe testiranja. Pored simulacije, Canoe pruža platformu za praćenje kompletnog mrežnog saobraćaja u realnom vremenu, kao i za detekciju i analizu greški nastalih tokom komunikacije.

U konfiguracionom prozoru Canoe programa moguće je kreirati željenu topologiju komunikacionog sistema. Izgled jedne praktične realizacije CAN mreže dat je na slici 4. Prikazani čvorovi simuliraju slanje svih potrebnih poruka koje bi unutar vozila bile slate od strane drugih elektronskih kontrolnih jedinica.



Slika 4. Realizacija CAN mreže u Canoe okruženju

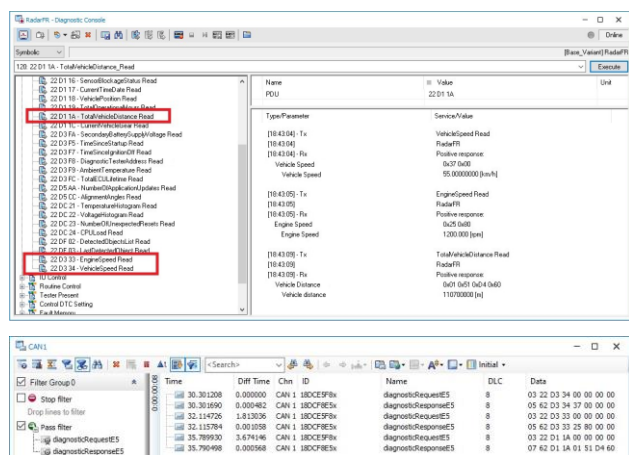
Usled velikog značaja, obezbeđivanje ispravnog rada dijagnostike nalazi se visoko na listi prioriteta svakog proizvođača motornih vozila. U skladu sa tim, Canoe programsko okruženje sadrži i alat za dizajn, razvoj, testiranje i simulaciju rada dijagnostičkih funkcionalnost jednog ECU-a. Za upotrebu dijagnostičkog sloja Canoe programa potrebno je standardnoj konfiguraciji pridodati i

dijagnostički opis, najčešće u vidu CDD fajla (engl. *CANdeli Diagnostic Description* - CDD). Dijagnostički opis se može posmatrati kao vrsta baze podataka za dijagnostičke zahteve koja u sebi sadrži definiciju i uputstvo za interpretaciju svih, od strane ECU-a podržanih, dijagnostičkih servisa i njihovih parametara (kako zahteva tako i odgovora). Na osnovu ubačenog dijagnostičkog opisa, Canoe automatski generiše dijagnostičku konzolu koja predstavlja intuitivni grafički interfejs namenjen jednostavnom slanju dijagnostičkih zahteva i simboličkoj interpretaciji dobijenih odgovora. Nakon odabira iz liste definisanih, zahtev se šalje dvoklikom miša, a dobijeni odgovor biva ispisan u samoj konzoli. Dijagnostička konzola se može smatrati softverskom realizacijom dijagnostičkog alata koji koristi server.

6. DIJAGNOSTIKA U CANOE OKRUŽENJU

U ovom poglavlju kroz će kroz Canoe simulaciju ECU-a radarskog sistema i njegovog okruženja, biti data verifikacija prethodno objašnjenih dijagnostičkih koncepta.

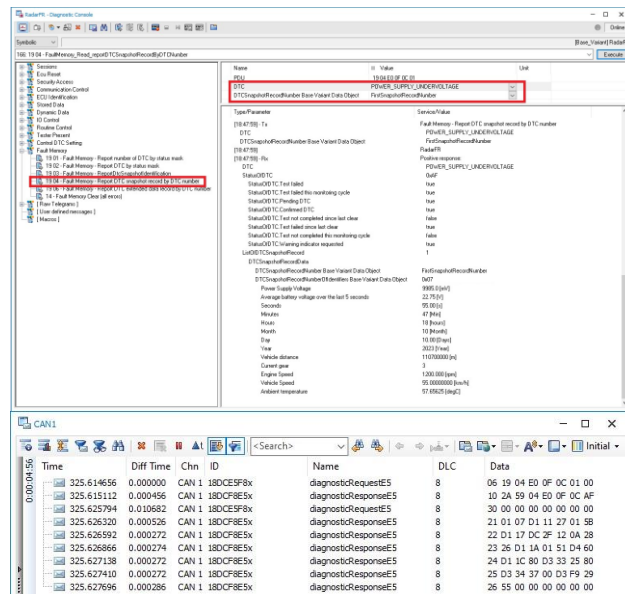
Simulacija je podešena tako da je trenutna brzina vozila postavljena na 55 km/h, broj obrtaja motora na 1200 obrtaja u minuti, a pređena kilometraža na 110.700 km. Sve za sam rad bitne informacije, uključujući i prethodno navedene, ECU dobija preko poruka na CAN magistrali ili sa svojih internih senzora. Trenutne vrednosti pomenutih veličina možemo očitati i sa samog ECU-a preko dijagnostike ukoliko ih zahtevamo preko servisa 0x22 uz pomoć odgovarajućih DID-ova (u ovom slučaju 0xD333, 0xD334 i 0xD11A) što je i prikazano na slici 5. Pored ispisa iz dijagnostičke konzole dat je i pregled CAN komunikacije koja se odigrala u pozadini. Na slici se jasno može uočiti da su za komunikaciju korišćeni pojedinačne poruke. Takođe, može se uvideti ogroman značaj dijagnostičkog alata (u ovom slučaju konzole) koji niz bajtova uz pomoć konverzionih formula prevodi u korisniku čitljive podatke.



Slika 5. Očitavanje stanja sistema putem dijagnostike

Ukoliko se u ovom trenutku u sistem unese greška spuštanjem napona napajanja simuliranog ECU-a ispod dozvoljene granice ona će rezultovati odgovarajućim kodom greške 0xE00F0C. Sada je moguće zatražiti i informacije o stanju sistema u trenutku inicijalnog nastanka greške uz pomoć servisa 0x19 njegovom pod-funkcijom

0x04, što je i prikazano na slici 6. Može se primetiti da se sačuvano stanje trenutka nastanka greške poklapa sa podacima koje su očitani neposredno pre unošenja greške u sistem. Kako je odgovor duži od 7 bajtova za njegovo slanje je morao biti korišten više-paketni prenos.



Slika 6. Stanje sistema u trenutku nastanka greške

7. ZAKLJUČAK

Evolucijom automobilske industrije elektronske kontrolne jedinice preuzimaju kontrolu nad svim bitnijim funkcijama jednog savremenog vozila. Za proveru ispravnosti rada, analizu greški, i parametrizaciju samih kontrolnih jedinica ključnu ulogu ima dijagnostika. U ovom radu predstavljen je UDS protokol kao osnovno sredstvo dijagnostičke komunikacije u automobilske industriji. Protokol je verifikovan kroz praktične primere izvedene u Canoe okruženju.

8. LITERATURA

- [1] International Organization for Standardization, ISO 14229-1 - Unified diagnostic services (UDS) - Part 1: Specification and requirements, 2013
- [2] ISO 15765-2 - Diagnostic communication over Controller Area Network (DoCAN) - Part 2: Transport protocol and network layer services, 2016
- [3] https://cdn.vector.com/cms/content/products/canoe/canoe/docs/Product%20Informations/CANoe_ProductInformation_EN.pdf (pristupljeno u oktobru 2023.)

Kratka biografija:



Pavle Janjić rođen je u Zrenjaninu 1997. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2023.god.

kontakt: pavlejanjic97@gmail.com

FUZZ ТЕСТИРАЊЕ DJANGO REST API-JA**FUZZ TESTING OF DJANGO REST API**

Никола Мајсторовић, Факултет техничких наука, Нови Сад

Област – РАЧУНАРСТВО И АУТОМАТИКА

Кратак садржај – У овом раду представљено је fuzz тестирање Django REST API-ја над апликацијом која представља друштвени веб сајт за међусобну комуникацију и дељење садржаја. У оквиру овог рада описана је целокупна архитектура система за тестирање Ђанго апликације, коју чини алат OWASP ZAP - подржава Fuzz тестирање и Elastic stack (Filebeat, Elasticsearch и Kibana) - колекција три различита пројекта намењена за анализу резултата тестирања. Детаљно је описано конфигурирање свих алата, платформе и радног оквира неопходног за fuzz тестирање предметне Django апликације. Након тога, извршена су fuzz тестирања Django апликације. Анализирају су резултати тестирања.

Кључне речи: логовање догађаја, тестирање система, отклањање рањивости

Abstract – This paper presents fuzz testing of the Django REST API on an application that represents a social website for mutual communication and content sharing. This paper describes the entire architecture of the Django application testing system, which consists of the OWASP ZAP tool - supports Fuzz testing and the Elastic stack (Filebeat, Elasticsearch and Kibana) - a collection of three different projects intended for the analysis of test results. The configuration of all the tools, platform and framework necessary for fuzz testing the Django application is described in detail. After that, fuzz tests of the Django application were performed. The test results were analyzed.

Keywords: Event logging, system testing, vulnerability remediation

1. УВОД

Тестирање софтверског система представља процес провере и евалуације предметног софтверског система како би се осигурало да комплетан предметни систем задовољи дефинисане захтеве, функционалности, перформансе, сигурности, скалабилности, компатибилности и отпорности на грешке. Могу се примењивати различите технике и методе, укључујући ручно тестирање, аутоматизовано тестирање, симулације, испитивање оптерећења и стрес тестирање [1]. Тестирање представља интегрални део процеса развоја софтвера. Поред исправног функционисања софтвера неопходно је обезбедити и одређени степен

сигурности корисника и система који софтверски производ користе. Отклањање сигурносних пропуста у софтверу путем тестирања и у току развоја софтвера у том случају постаје императив. Кључни део заштите информационог система је спречавање недозвољене елевације привилегија, односно заобилажење система контроле приступа.

Једна од метода за проналажење сигурносних пропуста је fuzz тестирање [2].

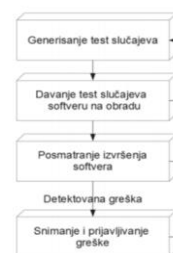
Предмет овог рада је примена fuzz тестирања софтвера – Django апликације, при чему треба истаћи да се овај метод тестирања користи за откривање грешака, рањивости и непредвиђених понашања у софтверским апликацијама базираним на REST API протоколу. Ова техника се често користи за побољшање сигурности и стабилности софтвера [3].

2. FUZZ ТЕСТИРАЊЕ

Сврха fuzz тестирања је прослеђивање података помоћу аутоматизованих или полу-аутоматизованих техника и тестирање система на различите изузетке. Услед високог степена аутоматизације, овај вид тестирања не захтева много експертског времена као ни претходно познавање софтвера [4].

Састоји од генерисања тест случајева улазних података и праћења извршења тестираног софтвера у току обраде.

Постоји више различитих приступа fuzz тестирању, а најзначајнија су: fuzz тестирање базирано на мутацији и fuzz тестирање засновано на генерисању. Кораци при fuzz тестирању приказани су на слици 2.1 [2].



Слика 2.1. Кораци при fuzz тестирању [2]

2.1. Методе тестирања софтвера

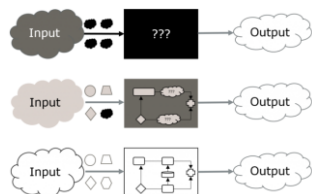
Метода тестирања софтвера обухвата стратегију и планирање тестирања целог софтверског производа. Метода тестирања укључује одабир одговарајућих техника тестирања за различите делове софтвера, постављање приоритета тестирања, идентификацију ресурса потребних за тестирање и дефинисање циљева тестирања [5].

При тестирању, систем за тестирање (енг. System Under Test, SUT) може да се посматра као црна, бела и сива кутија [6] као што је приказано на слици 2.2.

НАПОМЕНА:

Овај рад произтекао је из мастер рада чији ментор је био др Горан Сладић, ред. проф.

- Стратегија беле кутије, унутрашња структура система за тестирање је позната.
- Стратегија црне кутије, систем се посматра као црна кутија и не зна се ништа о унутрашњој структури система или коду.
- Стратегија сиве кутије је техника за тестирање апликације са ограниченим познавањем интерног рада апликације и знањем о основним аспектима система.



Слика 2.2 Fuzz тестирање методом црне, беле и сиве кутије

2.2. Технике тестирања софтвера

Техника тестирања софтвера се односи на специфичне методе или приступе које користимо за извођење тестирања софтвера [5].

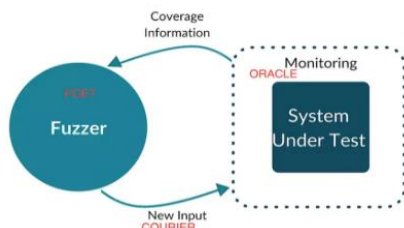
Постоје две основне технике које се примењују код fuzz тестирања. То су dumb fuzz и smart fuzz техника [7].

1. Dumb fuzz техника подразумева приступ код ког се потпуно насумично измењени подаци уносе у апликацију и посматра се реакција апликације на овако насумично измењене податке.
2. Smart fuzz технике уместо креирања потпуно насумичних улазних података, ради се промена специфичних вредности и на тај начин се утиче на основни формат и/или очекивано понашање.

2.3. Начин функционисања fuzz тестирања

Fuzz тестирање се састоји из три кључне компоненте као што је описано на слици 2.3 [8]:

1. “Poet” који креира деформисане улазе или тест случајеве заснованог на генерисању или мутацији у оквиру одређеног протокола, у зависности од тога да ли су улази генерисани од нуле или је потребно модификовање постојећих улаза
2. “Courier” испоручује тест случајеве циљном софтверу. Он може бити: Network protocol fuzzing, File fuzzing, API fuzzing, User interface fuzzing
3. “Oracle” непрекидно врши праћење рада система као и непрекидну процену стања система током одређеног временског периода у односу на његове активности и одређује да ли је тест прошао или није.



Слика 2.3. Опис рада fuzzing-а

2.4. Животни циклус fuzz тестирања

Животни циклус fuzz тестирања се може поделити на следеће кораке:

1. Одабир дела система над којим се врши тестирање
2. Одабир улаза
3. Генерисање fuzz података
4. Извршење тестова коришћењем fuzz података
5. Анализа понашања система
6. Евидентирање проблема

Кораци у fuzz тестирању биће кроз примере објашњени у поглављу 5.5 –Примена fuzz тестирања

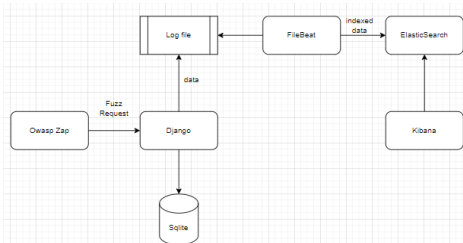
3. МОДЕЛ СИСТЕМА

Предметни систем треба да омогући корисницима тестирање Django REST API-ја, као и визуелни приказ резултата, ради увида у понашање поменутог API-ја.

Архитектура система као што је приказано на слици

3.1. састоји се од:

1. OWASP ZAP - алат који подржава fuzz тестирање
2. Filebeat – алат за доставу различитих врста података из лог фајла у ELK (ElasticSearch, Logstash, Kibana) stack ради анализе
3. ElasticSearch – дистрибуирани *multitenant search engine* за претрагу текстуалног садржаја заснован на *Lucene* библиотеци
4. Kibana је софтвер за визуализацију података за *Elasticsearch*
5. Django - веб апликација која је развијена у Django радном окружењу
6. SQLite – увезана библиотека која представља систем за управљање базама података који је уграђен у апликацију



Слика 3.1. Архитектура система

На слици 3.1. описан је концепт fuzz тестирања. Након сваког захтева Django логије одговор у фајл. Filebeat ослушкује промене у фајлу, прикупља податке које затим структурира и прослеђује у ElasticSearch који их индексира.

Kibana користи ElasticSearch индексе како би омогућио брз приступ и визуализацију података.

4. КОРИШЋЕНЕ ТЕХНОЛОГИЈЕ

У овом поглављу описани су сви алати, платформа и радни оквир неопходни за fuzz тестирање Django апликације.

4.1. OWASP ZAP

OWASP ZAP (скраћено *Zed Attack Proxy*) је open-source алат за сигурност веб апликација који је развијен од стране OWASP-a (*Open Web Application Security Project*). Алат се користи за тестирање

сигурности веб апликација и проналажење рањивости како би се побољшала њихова отпорност на нападе. OWASP ZAP је алат отвореног кода намењен да се користи у циљу тестирања безбедности апликација.

4.2. Filebeat

Filebeat је један од многих лаких „достављача података“ који су доступни као део Elastic Stack-а познатог као Beats. Beats пошиљачи података имају једну намену и дизајнирани су да буду инсталирани на машини која генерише податке без да имају било какав утицај на перформансе машине. Filebeat чита логове засноване на тексту и прослеђује их било директно у Elasticsearch или у Logstash [10].

4.3. Elasticsearch

ElasticSearch је дистрибуирана платформа за претраживање и анализу података. ElasticSearch је изграђен на пројекту Apache Lucene, који пружа моћне могућности за индексирање и претраживање великих количина података у реалном времену [11].

4.4. Kibana

Kibana је алат за управљање, истраживање, визуализацију и приказивање податке у реалном времену, развијен од компаније која је такође створила Elasticsearch, Logstash и Filebeat (ELK stack) чији је концепт приказан на слици 4.4.

Kibana укључује широк избор додатака и алата који омогућавају корисницима да претражују, врше преглед и анализу индексираних података. Kibana омогућава корисницима да интерактивно истражују, анализирају и визуализирају податке сачуване у Elasticsearch-у. Овај моћан алат се често користи уз Elasticsearch како би се добио увид у податке и идентификовале релевантне информације [12].

4.5. Django framework

Django је популарни open-source веб радни оквир (framework) написан у Python који олакшава развој веб апликација. Развијен је од стране Django Software Foundation-а и има богат скуп алата и функционалности који омогућују програмерима брзо стварање скалабилних, сигурних и сложених веб апликација [13].

4.6. Sqlite

SQLite је софтверски пакет јавног домена који обезбеђује систем за управљање базом података или RDBMS. Системи релационих база података се користе за складиштење кориснички дефинисаних записа у великим табелама. Поред складиштења и управљања подацима, систем за управљање базом података може да обрађује сложене команде упита које комбинују податке из више табела за генерисање извештаја и података [14].

5. ИМПЛЕМЕНТАЦИЈА СИСТЕМА

У овом поглављу биће приказани и описани сви делови система неопходни за fuzz тестирање Django апликације, као и конфигурација апликације, алата и engine-а.

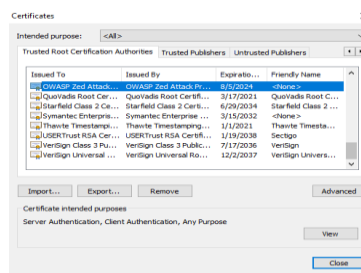
5.1. Конфигурација Owasp Zap алата

Да би се применило fuzz тестирање са OWASP ZAP алатом, потребно је инсталирати OWASP на систем.

Треба нагласити да ZAP захтева Java 11+ да би могао да се покрене.

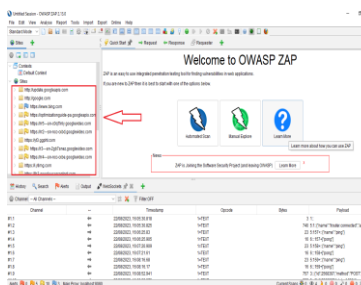
Пошто OWASP не приказује саобраћај локалног домаћина подразумевано у својој историји, потребно је подесити Chrome, или прегледач који ће се користити за тестирање, да користи ZAP проху, тако што се у проху подешавању у прегледачу упише локална IP адреса и порт који користи OWASP.

Након поставке, да би OWASP могао да пресреће HTTPS захтеве, потребно је изгенерисати сертификат од стране OWASP-а који увозимо у претраживач, као што је приказано на слици 5.1.



Слика 5.1. Сертификат потписан од OWASP ZAP-а

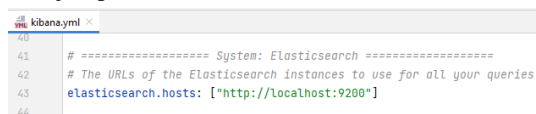
Након поставки и покретања прегледача, у OWASP-у биће приказана историја саобраћаја, као што је приказано на слици 5.2.



Слика 5.2. Историја саобраћаја

5.2. Конфигурација Kibane

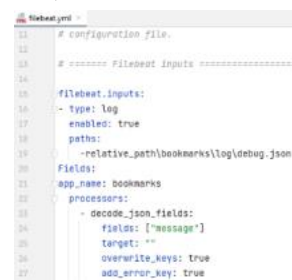
Да би Kibana интегрисали са Elasticsearch-ом, потребно је изменити kibana.yaml фајл који се налази у директоријуму config у Kibana директоријуму који је потребно преузети, и откоментарисати линију 43, као што је приказано на слици 5.3.



Слика 5.3. kibana.yaml

5.3. Конфигурација Filebeat-а

Да би Filebeat могао да региструје промене које се дешавају у Django-вом лог фајлу потребно је додати следеће линије у filebeat.yaml фајл, као што је приказано на слици 5.4.



Слика 5.4. filebeat.yaml

Као и линије кода, које су приказане на слици 5.5.

```
140 # ----- Outputs -----
141 # Configure what output to use when sending the data collected by the beat.
142 # ----- Elasticsearch Output -----
143 output.elasticsearch:
144   # Array of hosts to connect to.
145   hosts: ["localhost:9200"]
```

Слика 5.5. *filebeat.yml*

Додавањем линије изнад, Filebeat добија информацију на који излаз шаље трансформисане податке.

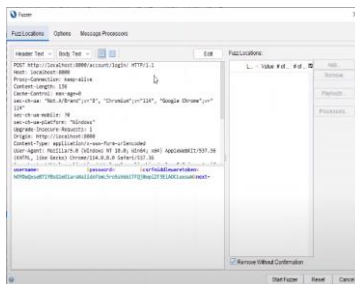
5.4. Конфигурација Django лог фајла

Да би Filebeat могао да прати логове у апликацији, лог фајл мора бити правилно структуриран у JSON формату у ком Filebeat може да га испарсира. Поља која су значајна за анализирање резултата у Kibana или лог фајлу, приказана су на листингу 5.1. као што је: *agent.id* - означава који је корисник послао захтев; *http.request.method* – означава тип захтева; *http.response.status_code* - означава да ли је одговор успешан; *url.path* представља *endpoint* за приказани одговор; *duration* представља дужину трајања захтева и може да указује на потенцијална уска грла (*bottleneck*), итд.

5.5. Примена fuzz тестирања

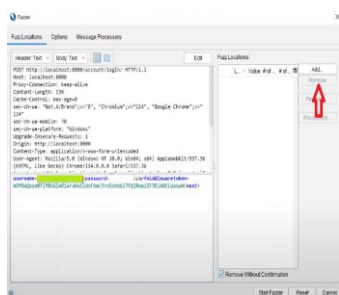
Да би тестирали API, потребно је да OWASP пресретне захтев помоћу прегледача и да на левом прозору буде приказана историја саобраћаја као што је приказано на слици 5.2.

Одабиром одређеног *endpoint*-а и бирањем функције *attack/Fuzz*, отвара се нови прозор који приказује детаље изабраног API-ја, као што је приказано на слици 5.6.



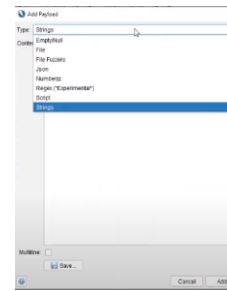
Слика 5.6. Детаљи API-ја који је OWASP снимио

Селектовање одређеног улаза, као што је приказано на слици 5.7, могуће је изменити податке и убацити нове тест случајеве.



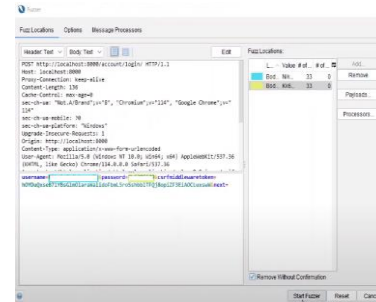
Слика 5.7. Селектовање улаза за генерисање нових тест случајева

Кликом на дугме *Add*, што је означено црвеном стрелицом на слици 5.7, отвара се нови прозор, приказан на слици 5.8, који служи за бирање начина за генерисање тест случајева.



Слика 5.8. Одабир начина генерисања улаза

Након одабира свих улаза и одређивање начина за генерисање тест случајева, као што је приказано на слици 5.9, fuzz тестирање се покреће кликом на дугме *Start Fuzzer*.



Слика 5.9. Одабир изабраних улаза за убацавање тест случајева за тестирање логовања

Ова процедура се понавља за тестирање било ког API-ја, а у наставку ће бити приказан само одабир изабраних улаза за убацавање тест случајева, као на слици 5.8 и анализа резултата тестирања.

5.5.1. Тестирање Login Api-ја

За тестирање *Login* API-ја предметне апликације, потребно је да се у улазне параметре *username* и *password* убаце тест случајеви као што је приказано у табlici 5.5. на слици 5.9.

Као улазни параметри променљиве *username* и *password* изабрани су тест случајеви: *SQL injection* и *Buffer Overflows*.

Након тестирања *Login* API-ја, приказани су резултати.

На графикону број одговора по времену, у интервалу од 15 секунди број одговора је од 2 до 18.

Дужина одговора по времену, дужина одговора не прелази 1 секунд, што имплицира да је одзив апликације задовољавајући.

Свих 2300 одговора, остао је непромењен током периода тестирања (200), што је жељени одговор.

Статус код по времену, остао је непромењен током периода тестирања (200), што је жељени одговор.

Дужина одговора по времену у одређеном интервалу, садржи више од 35000 карактера. Накнадном провером лог фајла, закључено је да је то из разлога, што је велики број карактера убачен у улазни параметар.

На основу приказаних резултата тестирања, закључено је да није дошло до грешке рада API-ја.

Исти поступак се понавља за тестирање логовања администраторске апликације, с тим да је *endpoint* за администраторску апликацију:

<http://localhost:8000/admin/login/>

Као и у примеру - тестирање *Login*-а над предметном апликацијом, разлог је велики број карактера убаčen у улазни параметар.

5.5.2. Тестирање API-ја за upload слике

Основни предуслов да би се извршило тестирање *upload*-а слике, је да корисник буде улогован у апликацију (да поседује валидан *CSRF token*). За тестирање API-ја за *upload* слике, потребно је да се у улазне параметре URL-а слике убаце тест случајеви. Као улазни параметри променљиве URL изабрани су тест случајеви: *URI SCHEMAS*, *XSS XML injection*, *SQL injection* и *Buffer Overflows*.

Тестирањем API-ја за *upload* слике, добијени су резултати.

На графикону, број одговора по времену у интервалу од 5 секунди је од 10 до 49.

Дужина одговора по времену је испод 0.012 секунди.

Број одговора, остао је непромењен током периода тестирања (200), што је жељени одговор.

Статус код по времену, остао је непромењен током периода тестирања (200), што је жељени одговор.

Дужина одговора по времену у одређеном интервалу је уједначена и садржи око 2400 карактера по захтеву.

На основу приказаних резултата тестирања, закључак је да није дошло до грешке рада овог API-ја.

5.5.3. Тестирање API-ја за брисање изабраног корисника из апликације

Претпоставка да би се извршило тестирање брисања корисника из апликације је да корисник који тестира буде улогован у апликацију (да поседује валидан *CSRF token*) као администратор.

Након тестирања API-ја за брисање изабраног корисника из апликације и посматрања података у бази података, закључено је да су подаци остали непромењени, односно да корисник који нема улогу администратора не може да обрише другог корисника.

6. ЗАКЉУЧАК

У овом раду посебан нагласак стављен је на примену fuzz тестирања користећи OWASP ZAP алат, као и Elastic stack (Filebeat, Elasticsearch и Kibana) за визуализацију и анализу резултата тестирања, како би се идентификовале потенцијалне рањивости у API-јима у Django апликацији која је тестирана.

Интеграција OWASP ZAP алата са Elastic stack-ом пружа могућност ефикасног прикупљања резултата тестирања и анализу истих, чиме се добија дубљи увид у потенцијалне рањивости API-ја, као и степен озбиљности рањивости. У раду је детаљно описано конфигурирање OWASP алата, Elastic stack-а и Django апликације за праћење логова.

Детаљно су тестирана четири API-ја користећи OWASP ZAP алат за генерисање и слање различитих облика неважећих и неочекиваних података.

Након пажљиве анализе резултата користећи Kibana као алат за визуелизацију података, може се закључити да није идентификована ниједна критична или озбиљна рањивост у анализираним API-јима.

Резултати изложени у раду показују да су имплементиране безбедносне мере у тестираним случајевима на одговарајућем нивоу и да су API-ји

отпорни на познате облике напада који су симулирани током fuzz тестирања.

7. ЛИТЕРАТУРА

- [1] "Automated Software Testing. 1999", Dustin, E.
- [2] "Detekcija sigurnosnih propusta faz testiranjem", Univerzitet u Novom Sadu, Fakultet tehničkih nauka, 2013, Nikolić Aleksandar, Sladić Goran, Milosavljević Branko, Konjović Zora.
- [3] "REST API Fuzzing by Coverage Level Guided Blackbox Testing", Chung-Hsuan Tsai Shi-Chun Tsai; Shih-Kun Huang, 2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)
- [4] "Fuzz testiranje REST API-ja", Univerzitet u Novom Sadu, Fakultet tehničkih nauka 2021, Tamara Veličković
- [5] "Metode I tehnike testiranja softvera" 2021, Sveučilište u Zagrebu Ekonomski fakultet Zagreb, Informatički menadžment, Andreja Mamić
- [6] "Neke metode za automatizovano testiranje softvera", Univerzitet u Beogradu Matematički fakultet, Kruna Matijević
- [7] "Primena Fuzz testiranja na DMS softver po SDL metodologiji" Univerzitet u Novom Sadu, Fakultet tehničkih nauka, 2010, Kristina Stojaković
- [8] What is Fuzzing, © 2017 Synopsys, Inc, <https://www.synopsys.com/content/dam/synopsys/sig-assets/whitepapers/what-is-fuzzing.pdf>
- [9] "Study of the techniques used by OWASP ZAP for analysis of vulnerabilities in web applications", 2022, Linköping University | Department of Computer and Information Science, Adam Jakobsson
- [10] "SCADA STATISTICS MONITORING USING THE Elastic Stack (Elasticsearch, Logstash, Kibana)", 2017, James Hamilton, Brad Schofield, Manuel Gonzalez Berges, Jean-Charles Tournier CERN, Geneva, Switzerland
- [11] "Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and analytics engine", 2015, Clinton Gormley, Zachary Tong
- [12] "Kibana 7 Quick Start Guide: Visualize your Elasticsearch data with ease", 2019, Anurag Srivastava
- [13] "Beginning Django Web Application Development and Deployment with Python", 2017, Daniel Rubio
- [14] "Using Sqlite", 2010, Jay A. Kreibich

Kratka biografija:



Никола Мајсторовић рођен је у Новом Саду 1993. године. Мастер рад на Факултету техничких наука из области Рачунарство и аутоматика – Електронско пословање одбранио је 2023. године контакт: majsta93@hotmail.com

ANALIZA GEOMETRIJE MANIFOLDA U AUTOENKODERIMA**MANIFOLD GEOMETRY ANALYSIS IN AUTOENCODERS**Vuko Jovičić, *Fakultet tehničkih nauka, Novi Sad***Oblast – RAČUNARSTVO I AUTOMATIKA**

Kratak sadržaj – Ovaj rad posvećen je pokušaju da se razumeju i rasvetle neki od principa u radu dubokih neuronskih mreža. Konkretno, kako širina središnjeg sloja u autoenkoderu, intrinzična dimenzionalnost ulaznog skupa podataka i dužina obučavanja mreže utiču na geometriju formiranog manifolda.

Ključne reči: Autoenkoderi, mašinsko učenje, manifoldi, interpolacija

Abstract – This paper attempts to help in understanding and clarifying some of the principles in the work of deep neural networks. Specifically, how the width of the bottleneck layer in the autoencoder, the intrinsic dimensionality of the input data and the training duration affect the geometry of the formed manifold.

Keywords: Autoencoders, machine learning, manifolds, interpolation

1. UVOD

Neuronske mreže i duboko učenje u današnje vreme predstavljaju standardnu tehniku za rešavanje problema mašinskog učenja. Ipak, čak i uz svakodnevnu pojavu novih modela i arhitektura, kao i sve bolje performanse koje oni pružaju, neuronske mreže u mnogome i dalje ostaju crna kutija a mnogobrojni detalji u njihovom funkcionisanju ostaju nepoznanica.

Neka od pitanja na koji istraživači pokušavaju da daju odgovor tiču se geometrije interne reprezentacije podataka u skrivenim slojevima mreže. Bolje razumevanje određenih topoloških i geometrijskih struktura unutar neuronske mreže kao i načina na koji se formiraju dovelo bi do mogućnosti modelovanja efikasnijih arhitektura, interpretabilnijih sistema i predvidivijih izlaza.

Podešavanje parametara mreže je jedan od osnovnih koraka u obučavanju svakog modela a svi oni na ovaj ili onaj način utiču na formiranje interne reprezentacije podataka. Iako se ova oblast sve više istražuje i dolazi se do novih saznanja, u praksi se i dalje odabir arhitekture za rešavanje određenog problema kao i podešavanje parametara u najvećoj meri zasniva na iskustvu i empirijskom pristupu.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Segedinac, vanr. prof.

Različita istraživanja i eksperimenti vršeni su u cilju boljeg razumevanja uticaja interne reprezentacije podataka na funkcionisanje neuronskih mreža.

Rad [1] posvećen je upravo razumevanju geometrije dubokih neuronskih mreža. Potvrđuje se hipoteza da glavni uspeh neuronskih mreža leži u činjenici da prirodni podaci visoke dimenzionalnosti zapravo leže u niže-dimenzionalnom prostoru (hipoteza o manifoldu). Rad [2] se bavio topološkim i geometrijskim osobinama manifolda i modelovanju istih u cilju što boljeg predstavljanja određenog skupa podataka sa određenom intrinzičnom topološkom strukturom. U [3], cilj istraživanja bio je ispratiti intrinzičnu dimenzionalnost interne reprezentacije slike kroz različite slojeve duboke neuronske mreže.

Jedan od glavnih motiva za izučavanje geometrije interne reprezentacije objekata jeste i interpolacija unutar prostora reprezentacija. Rad [4] se zasniva na ideji da dublji slojevi neuronske mreže predstavljaju apstraktnije karakteristike podataka od ranih slojeva i da interpolacija između tačaka u tom prostoru ima više semantičkog značaja nego u početnim slojevima mreže.

Cilj ovog rada bio je da kroz eksperimente pokuša dati svoj doprinos u otkrivanju određenih zakonitosti između različitih parametara mreže i rezultujuće geometrije manifolda.

2. TEORIJSKE OSNOVE

Za razumevanje metodologije i rezultata izvršenih eksperimenata potrebno je poznavati određene teorijske pojmove:

2.1. Manifold

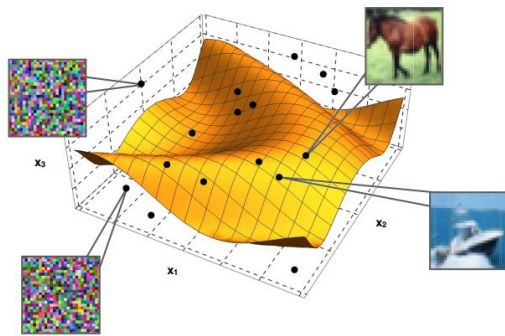
U matematičkoj teoriji, manifold predstavlja topološki prostor koji lokalno liči na euklidski. Preciznije, n -dimenzionalni manifold ili skraćeno n -manifold je topološki prostor sa osobinom da svaka tačka unutar njega ima okolinu koja je homeomorfna sa otvorenom jediničnom loptom u R^n .

2.2. Hipoteza o manifoldu

Hipoteza o manifoldu kaže da se mnogi visoko-dimenzionalni skupovi podataka koji se mogu naći u stvarnom svetu tj. prirodi zapravo koncentrišu u blizini niže-dimenzionalnih manifolda unutar tih visoko-dimenzionalnih prostora.

Opšte je prihvaćeno da hipoteza o manifoldu objašnjava efikasnost algoritama mašinskog učenja u opisivanju visoko-dimenzionalnih skupova podataka malim brojem

parametara tj. u rešavanju problema nelinearne redukcije dimenzionalnosti skupa podataka, kao što je u slučaju autoenkodera.



Slika 1. Pojednostavljeni prikaz hipoteze o manifoldu [5]

2.3. Autoenkoderi

Autoenkoderi predstavljaju posebnu vrstu veštačkih neuronskih mreža koji za cilj imaju da nauče reprezentativne karakteristike - kodove neoznačenih skupova podataka.

Kod standardnog autonekodera, središnji sloj u kome se čuva reprezentativni kod ulaznog skupa podataka manje je dimenzionalnosti od ulaznog sloja, pa je tako mreža prinuđena da odbaci neke od manje važnih karakteristika skupa (šum) a zadrži samo one najekspresivnije, stvarajući tako kompresovani oblik ulaznog skupa koji je u rešavanju mnogih problema korisniji od originalne reprezentacije. Prostor u kome se nalazi kompresovana reprezentacija podataka naziva se i latentni prostor.

2.4. Intrinzična dimenzionalnost

Intrinzična dimenzionalnost - ID predstavlja minimalni broj dimenzija neophodnih da se opišu elementi skupa bez značajnog gubitka informacija.

Problem pronalaženja ID određenog skupa je vrlo zahtevan i postoje mnogi algoritmi i metodi u njegovom rešavanju. Neki od estimatora ID su: DANCo [6], KNN, PCA, MADA [7], TwoNN [8] itd.

Za većinu realnih skupova podataka iz prirode, moguće je napraviti razliku između pojmova globalna intrinzična dimenzionalnost GID i lokalna intrinzična dimenzionalnost LID. Ovi koncepti usko su povezani sa konceptom manifolda. LID predstavlja dimenzionalnost manifolda u okolini tačke koja se posmatra, dok GID sagledava čitav skup tačaka i određuje dimenzionalnost konveksnog omotača manifolda.

2.5. Prokletstvo dimenzionalnosti

Ovaj pojam se odnosi na različite fenomene do kojih dolazi pri radu sa podacima u visoko-dimenzionalnim prostorima, za razliku od nisko-dimenzionalnih sa kojima se svakodnevno susrećemo.

Osnovna pojava do koje dolazi je da zapremina prostora raste eksponencijalno sa uvećavanjem broja dimenzija, pa tako i broj tačaka koji je potreban da bi se zadržala određena gustina mora rasti eksponencijalno.

Takođe, kako dimenzionalnost prostora raste, tačke tog prostora međusobno postaju sve udaljenije i razućenije,

sve do stepena da praktično postaju jednako udaljene jedne od drugih. Ova osobina, između ostalog, onemogućava rad mnogim algoritmima koji se oslanjaju na prepoznavanje određenih grupa tačaka ili celina koje dele određenu zajedničku osobinu.

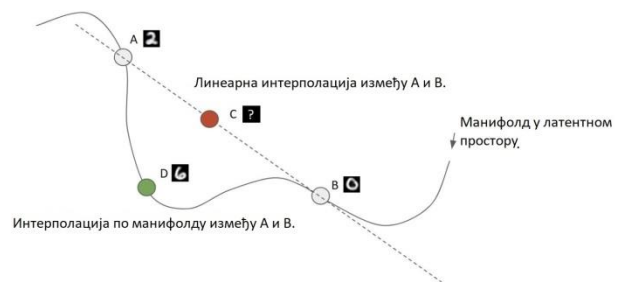
Eksperimentalni deo ovog rada baviće se i pokušajem da se donesu zaključci o promeni prirode skupa podataka pri drastičnom povećanju dimenzionalnosti.

2.6. Interpolacija i ekstrapolacija

Pojmovi interpolacije i ekstrapolacije su fundamentalni kako u mašinskom učenju tako i u drugim oblastima oblastima tehničkih nauka i matematike.

Do interpolacije dolazi, posmatrajući tačku x , kada god ona pripada konveksnom omotaču skupa tačaka $X = \{x_1, x_2, \dots, x_n\}$, u suprotnom dolazi do ekstrapolacije.

U kontekstu mašinskog učenja i dubokih neuronskih mreža, interpolacija se može povezati sa pojmom manifolda. Kao što se može videti na slici 2, linearna interpolacija kroz latentni prostor reprezentacija rezultovaoće nepredvidivim slikama, dok će interpolacija po manifoldu rezultirati poznatima primerima iz skupa podataka ili njihovim kombinacijama na čemu se u konačnici zasnivaju generativni modeli.

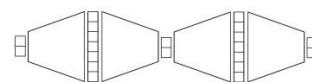


Slika 2. Prikaz linearne interpolacije i interpolacije po manifoldu u latentnom prostoru mreže [9]

3. EKSPERIMENTI

3.1. Eksperimenti sa niskodimenzionalnim podacima

Prvi eksperimenti vršeni su sa najprostijim skupovima podataka – tačkama, ravnomerno raspoređenim u 2D i 3D prostoru, koje čine mrežu. Kako je za ove skupove i početna i intrinzična dimenzionalnost dva odnosno tri, nemoguće je u stilu standardnog autoenkodera kompresovati podatke u prostor niže dimenzije. Iz ovog razloga autoenkoder je imao specifičnu strukturu, kao što se vidi na slici 3.

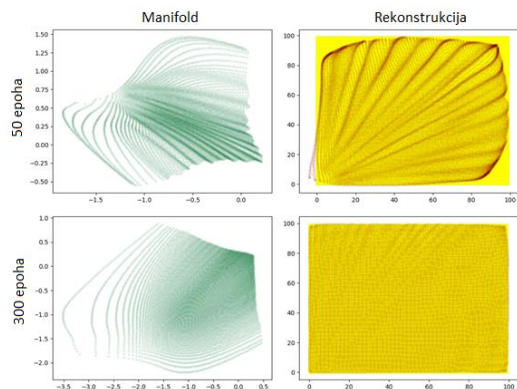


Slika 3. Arhitektura autoenkodera u inicijalnim eksperimentima

Dimenzionalnost skupa je veštački povećana da bi naknadno bila smanjena.

Rezultati eksperimenta pokazali su da duže obučavanje mreže „pegla” tj. ravna manifold, što dovodi do bolje rekonstrukcije, makar za visoko-strukturirane skupove podataka – slika 4. Takođe, do ovog efekta dovodi i pove-

ćavanje širine skrivenih slojeva u mreži tj. projektovanje podataka u više-dimenzionalne prostore pre kompresije.



Slika 4. Primer manifolda koji predstavlja mrežu tačaka u 2D i njegovu rekonstrukciju, u zavisnosti od dužine obučavanja autoenkodera.

3.2. Interpolacija i ekstrapolacija

Da bi se prevazišao problem veštačkog uvećavanja dimenzionalnosti, 2D tačke su u narednim eksperimentima prikazane kao bele tačke poluprečnika 4 piksela na crnoj slici veličine 28x28 piksela. Na ovaj način neosporno je održana ID = 2, ali je zato početna tj. stvarna dimenzionalnost skupa bila $28 \times 28 = 784D$, čime je autonekoderu data mogućnost da efikasno kompresuje podatke i iskaže određene karakteristike.

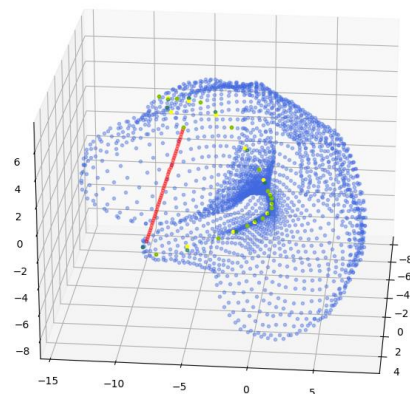
Slično prethodnom primeru, skup podataka predstavljao je skup 2D tačaka koje zauzimaju svaku od 784 moguće pozicije na slici. Mreža je trenirana 50 epoha, koristeći tanh aktivacionu funkciju, brzinu obučavanja od 0.001 i širinu središnjeg sloja (dimenzionalnost latentnog prostora LD) koji je varirao od 1 do 300.

S obzirom na ID od 2, obistinilo se očekivanje da će mreža sa LD 1, imati problem sa rekonstrukcijom primera iz skupa podataka. Za LD>1 rekonstrukcija je bila uspešna, a već kod LD=5, konvergirala je do apsolutno tačne. Uspešna rekonstrukcija bila je početni uslov za dalje eksperimentisanje.

Za svaki od 300 eksperimenata, vršeno je poređenje između linearne interpolacije i interpolacije po manifoldu. Kao početna i krajnja tačka interpolacije uzimane su prva i poslednja tačka skupa odnosno tačke u gornjem-levom i donjem-desnom uglu slike. Kod linearne, uzrokovane su tačke sa konstantnom međusobnom udaljenošću najkraćom mogućom putanjom između prve i početne tačke u euklidskom prostoru. Interpolacija po manifoldu sa druge strane rađena je na sledeći način: Odabrane su tačke dijagonale koje predstavljaju „ground truth” u interpolaciji između zadatih tačaka. Nakon toga izostavljena je svaka druga tačka i aproksimirana sa središnjom tačkom linearne interpolacije između njoj prethodne i naredne tačke. Potom je merena prosečna udaljenost aproksimiranih tačaka od njihovih pravih vrednosti.

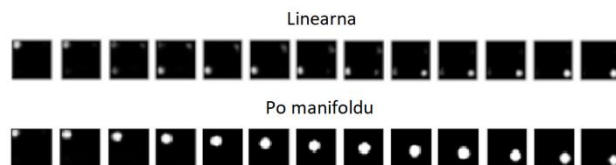
Na slici 5 može se videti primer linearne interpolacije (crvena trajektorija) i interpolacije po manifoldu (žuta trajektorija) u 3D prostoru, kao i odstupanje aproksimirane trajektorije od „ground truth” trajektorije – $mean\ dist = 0.280$.

MEAN DIST: 0.2802013



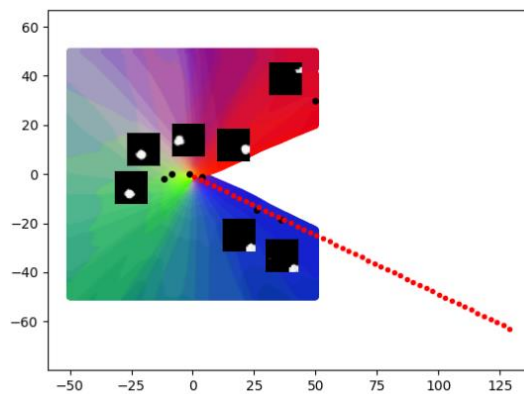
Slika 5. 2-manifold u 3D prostoru

Kao što se može videti na slici 6, linearne interpolacije je rezultirala tačkama koje sadrže deljene karakteristike svojih najbližih primera dok je interpolacija po manifoldu kreirala očekivanu translaciju tačke po dijagonali.



Slika 6. Rekonstrukcija nakon linearne i interpolacije po manifoldu

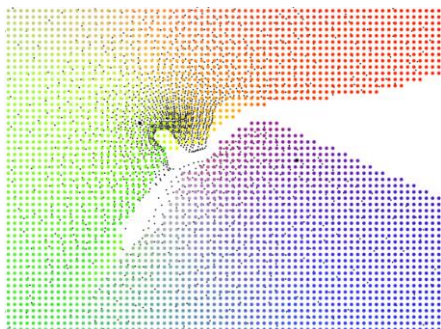
U cilju boljeg razumevanja dobijenih rezultata i izvođenja određenih zaključaka izvršen je dodatni eksperiment: uzrokovan je kompletan 2D latentni prostor u određenim granicama i svi uzorci dekodovani su u slike. Dobijene rekonstrukcije procesirane su tako da su slike čistih tačaka prikazane kao pikseli određenih boja koje koreliraju sa 2D prostorom, dok su slike sa artefaktima ili više od jedne tačke ostavljene u beloj boji. Rezultat se može videti na slici 7.



Slika 7. Uzrokovanje čitavog 2D prostora i predstavljanje rekonstruisanih slika bojama

Ovaj rezultat protumačen je na sledeći način: Informacije o određenom primeru iz skupa podataka najpreciznije su sadržane u tački koji je taj primer zauzeo na manifoldu. Ipak, čak i daleko od manifolda, te informacije ostaju zadržane i moguće je rekonstruisati primer iz udaljene tačke, dok god je ta tačka najbliža originalnoj tački. Ovo

viđenje razlikuje se od očekivanog ponašanja po kome bi uzrokovanje tačke udaljene od manifolda trebalo da rezultuje šumom. Takođe, sa slika 7 i 8 može se zaključiti da su bele, neodređene, regije prostora nastale tamo gde je manifold bio presavijen.



Slika 8. Crnom bojom prikazane su tačke manifolda

Ovaj zaključak potvrđen je i testiranjem ekstrapolacije iz određene tačke na manifoldu. Kao što se vidi na slici 7, crvenom trajektorijom prikazana je ekstrapolacija od tačke iz središta prostora u proizvoljnom pravcu. Čak i na velikoj udaljenosti, rekonstrukcija je ostala konstantna tj. zadržala je formu slike bele tačke u donjem desnom uglu.

Ovakva pojava nastavila se sve do $LD > 100$. Sa povećanjem broja dimenzija u latentnom prostoru, pojavile su se slike sa artefaktima u ekstrapolativnom režimu. Ovaj rezultat slaže se sa teorijom o visokodimenzionalnim prostorima i tezom da u takvim prostorima tačke postaju međusobno jednako udaljene pa tako poprimaju karakteristike tj. informacije svih ostalih tačaka.

Merenjem udaljenosti tačaka manifolda od koordinatnog početka kroz različite LD, ustanovljeno je da standardna devijacija raspodela tih udaljenosti postaje sve manja, tj. da se tačke koncentrišu oko određene hiper sfere ili makar u nekim njenim delovima, sve dok ne dostignu određeni limit tj. do konvergencije. Ta konvergencija se u ovom eksperimentu dešava oko dvadesete dimenzije, što se poklapa sa dimenzijom u kojoj konvergira i odstupanje aproksimirane od „ground truth” trajektorije.

5. ZAKLJUČAK

Upotrebom geometrijski visoko-strukturiranog skupa podataka sa poznatom ID i interpretabilnim rezultatima interpolacije, istraživana je unutrašnjost latentnog prostora neuronske mreže tj. autoenkodera i doneti su određeni zaključci:

- Dužina obučavanja mreže i dimenzionalnost središnjeg sloja utiču na stepen uvijenosti formiranog manifolda.
- Linearna interpolacija između tačaka u najvećem broju slučajeva, a pogotovo u prostorima visoke dimenzionalnosti, ne prolazi kroz tačke manifolda pa samim tim ne rezultira primerima iz skupa podataka.
- Neuronska mreža smešta informacije o enkodovanim primerima po celom latentnom prostoru, a ne samo u blizini manifolda. Ipak, daleko od manifolda a pogotovo kod velike LD, informacije se prepliću i nepredvidive su.

- Sa povećanjem LD, tačke manifolda koncentrišu se u blizini delova hiper-sfere, tj. razlika u razdaljinama tačaka manifolda od koordinatnog početka prostora postaje sve manja.

Ovaj rad samo je zagrebao u neistraženu prirodu strukture latentnih prostora i ostavlja mnoštvo pitanja za dalje istraživanje: Da li je moguće kretati se geodezično (sferično) po manifoldu pri dovoljno velikom LD? Kako bi uvođenje dodatnog skupa podataka različite strukture i ID uticalo na manifold? Kako se odnos GID i LID menja kroz različite LD i u kakvoj je vezi sa uvijenošću manifolda?

6. LITERATURA

- [1] Lei, Na, et al. "A geometric understanding of deep learning." *Engineering* 6.3 (2020): 361-374.
- [2] Rey, Luis A. Pérez, Vlado Menkovski, and Jacobus W. Portegies. "Diffusion variational autoencoders." *arXiv preprint arXiv:1901.08991* (2019).
- [3] Ansuini, Alessio, et al. "Intrinsic dimension of data representations in deep neural networks." *Advances in Neural Information Processing Systems* 32 (2019).
- [4] Ozair, Sherjil, and Yoshua Bengio. "Deep directed generative autoencoders." *arXiv preprint arXiv:1410.0630* (2014).
- [5] Goldt, Sebastian, et al. "Modeling the influence of data structure on learning in neural networks: The hidden manifold model." *Physical Review X* 10.4 (2020): 041044.
- [6] Ceruti, Claudio, et al. "DANCo: dimensionality from angle and norm concentration." *arXiv preprint arXiv:1206.3881* (2012).
- [7] Farahmand, Amir Massoud, Csaba Szepesvári, and Jean-Yves Audibert. "Manifold-adaptive dimension estimation." *Proceedings of the 24th international conference on Machine learning*. 2007.
- [8] Facco, Elena, et al. "Estimating the intrinsic dimension of datasets by a minimal neighborhood information." *Scientific reports* 7.1 (2017): 1-8.
- [9] Chollet, Francois. *Deep Learning with Python*. Manning Publications, 2017.

Kratka biografija:



Vuko Jovičić rođen je 1994. god. u Novom Sadu. Završio je gimnaziju „Svetožar Marković“ u Novom Sadu 2013. godine. Fakultet tehničkih nauka je upisao iste godine. Odbranio je diplomski rad 2019. godine na temu „Primena metoda mašinskog učenja u semantičkoj analizi stripa“, nakon čega nastavlja sa master studijama na istom fakultetu.
kontakt: jovicicvuko@gmail.com

ИМПЛЕМЕНТАЦИЈА ПОДРШКЕ ЗА ВИШЕСТРУКО ИНСТАНЦИРАЊЕ АКТИВНОСТИ У ПРОЦЕСНОМ ОКРУЖЕЊУ NEW WAVE**IMPLEMENTING SUPPORT FOR MULTIINSTANCE ACTIVITIES IN NEW WAVE PROCESS ENGINE**

Никола Ђорђевић, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – У раду је представљена имплементација модула који процесном окружењу NewWave Engine омогућава рад са вишеструко инстанцираним задацима (енгл. multiinstance task). Овај модул је написан у програмском језику Фаро.

Кључне речи: Пословни процес, Workflow engine, Фаро, NewWave

Abstract – This document presents the implementation of a module that will add support for multiinstance tasks to the NewWave Engine process execution environment. This module is written in the programming language Pharo.

Keywords: Business process, Workflow engine, Pharo, NewWave

1. УВОД

Пословни процеси су од кључног значаја за сваку компанију. Пословни процес се може дефинисати као активност или скуп активности којима се постиже пословни циљ компаније[1]. Тржиште које је све конкурентније и динамичније повлачи потребу организација да унапреде своје пословне процесе. Управљање пословним процесима (енгл. Business Process Management - BPM) је дисциплина која за примарни циљ има да унапреди пословне процесе, да се смањи комплексност, а повећа ефикасност. Workflow engine конвертује традиционални мануелни workflow од ИТ-вођених задатака до процеса којим управља човек или софтвер и врше рутирање и усмеравање информационих путања, одговорности и контролишу комуникационе канале при сарадњи различитих процеса како би се што ефикасније искористили ресурси. Већина користи БПМН процесни модел за дијаграм који усмерава workflow. Workflow engine је софтверски сервис који пружа окружење за праводобно извршавање задатака за процесне инстанце [2], [3], у складу са ограничењима и редоследом дефинисаним у процесном моделу.

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији ментор је био др Мирослав Зарић, ванр. проф.

Вишеструке активности (активности са више инстанци - енгл. Multi instance activities) - поред спецификације задатка који треба да се изврши, као улазни параметар имају и колекцију улазних објеката.

За сваки објекат дате колекције креира се једна инстанца задатка. Уколико се ради о корисничким задацима, сваку појединачну инстанцу типично извршава други корисник. Могу бити секвенцијални или паралелни. Секвенцијалне вишеструке активности су оне код којих се инстанце извршавају једна за другом – секвенцијално. Други задатак по реду неће почети са својим извршавањем све док се први не заврши, трећи ће чекати на други, итд. Обележавају се следећим маркером:



Слика 1. Графичка репрезентација маркера за секвенцијалне вишеструке активности

Код паралелних вишеструких активности сви инстанцирани таскови ће почети да се извршавају у истом тренутку – паралелно. Целокупна активност ће бити завршена тек онда када се оконча свака инстанца у оквиру посматране паралелне вишеструке активности. Обележавају се паралелним маркером:



Слика 2. Графичка репрезентација маркера за паралелне вишеструке активности

Фаро (енгл. Pharo) је једноставан, објектно оријентисан програмски језик отвореног кода (енгл. open-source). Open-source значи да је изворни код доступан свима, и за преглед, и за модификацију кода. Његова виртуална машина је у потпуности написана у Фару, што значајно олакшава симулацију, проналажење грешака, анализу и измене из самог Фаро.

PharoLauncher је апликација која омогућава преузимање и управљање фаро сликама (енгл. Pharo image). Кориснику даје приступ сликама који садрже специфичне библиотеке што значајно олакшава

њихову употребу и елиминише потребу да се ручно инсталирају и конфигуришу [4].

NewWave је окружење отвореног кода, за управљање пословним процесима, написано у програмском језику Фаро. У питању је лако проширив систем, са циљем да омогући једноставну спецификацију и имплементацију пословних процеса. Иако је у фази развоја, извршно окружење (*NewWave engine*) подржава основне обрасце контроле тока, као што су секвенцијални, паралелно гранање и спајање са синхронизацијом, ексклузивна гранања и спајања, различите типове догађаја, итд [5].

WaveEngine је главна компонента *NewWave* радног окружења, и она организује и контролише извршавање радних токова. Садржи један главни *WaveExecutor*, који је одговоран за само извршавање радног тока (енгл. *workflow*) и има задатак да пружи погодно окружење да би се извршио елемент који је дошао на ред. *Flowhandler* је компонента која на основу модела радног тока одређује који елемент је следећи по реду за извршавање.

2. ИМПЛЕМЕНТАЦИЈА СИСТЕМА

2.1. Бекенд апликације

Првобитно је потребно креирати класе за вишеструке задатке:

Узимајући у обзир да су задаци у суштини активности, одлучено је да ће класа под називом *NWMultiInstanceTask* наслеђивати *NWBaseActivity* класу. Ова класа садржи две варијабле на нивоу инстанци:

- *task*
- *users* (Колекција *Username*-ова)

Наредни корак је био да се специфицира шта и на који начин треба да се изврши када *FlowHandler* наиђе на *NWMultiInstanceUserTask*. Ово је урађено у класи – *NWMultiInstanceUserTaskBehavior*. Примарна метода у овој класи је *performExecution*. Она поседује два обавезна улазна параметра:

- *elemToExecute* – Ова променљива је инстанца објекта *NWMultiInstanceUserTask* и у себи садржи неопходне податке

(*task*, *users*) за креирање инстанци паралелних таскова.

- *waveExecutor* – Садржи инстанцу *WaveExecutor*-а који је задужен за извршавање ове активности.

На слици 3. је приказан код и начин имплементације ове методе.

Као што се може видети, на самом почетку је неопходно креирати чвор за паралелно спајање, инстанцирањем класе *NWParallelJoin*. Овај чвор (елемент) има задатак да спречи наставак извршавања пословног процеса, све док се не заврше сви таскови који имају *outGoingEdge* према њему.

```
performExecution: elemToExecute executor: waveExecutor
| pluginData userIds join endMulti |
pluginData := waveExecutor processHandler engine pluginData.

join := NWParallelJoin new.
join description: 'MultiInstance Join'.

userIds := elemToExecute users.
userIds do: [ :each |
| tempUserTask |
tempUserTask := (self generateDefaultUserTask:
each pluginData: pluginData).
tempUserTask addOutgoingEdge: join.
waveExecutor processHandler addSubExecutor: (NWSequence source:
nil target: tempUserTask) ].

endMulti := NWEndMultiInstanceEvent new.
endMulti description: 'End MultiInstance Task'.

join addOutgoingEdge: endMulti.

waveExecutor processHandler engineAnnouncer
when: EndMultiInstanceAnnouncer
do: [ :ann |
waveExecutor flowHandler node:
elemToExecute outgoingFlows first targetRef.
waveExecutor tryToExecuteNext: elemToExecute.
].
```

Слика 3. Метода *performExecution* класе *NWMultiInstanceUserTaskBehavior*

Затим се преузима колекција корисничких имена (енгл. *username*). За сваког корисника из дате колекције креирамо један предефинисани *UserTask* користећи методу *generateDefaultUserTask*, којој прослеђујемо *username* корисника као и променљиву *pluginData* која садржи податке о свим корисницима у систему. Ова метода је приказана на слици 4.

```
generateDefaultUserTask: userId pluginData: pluginData
| user task do adresa |
user := (pluginData at: #NWUserManagement) findUserByName:
userId asString.
Transcript show: user userId.

adresa := Adresa example3.
do := NWDataObject new. |
do valuedDataObject:
(NWClassDescriptionGenerator dataObject: adresa) createClassDescription.
do name: ('Address data object example ', userId asString).

task := NWUserTask new.
task id: userId.
task description: 'Entering the address'.
task name: 'Address task'.
task value: 'Task1 value'.
"tl group: (management findGroup: '1')."
task user: user.
task addDataOutputAssociation: do.
"wave waveAnnouncer announce: JoinEventAnnouncer new."
^ task
```

Слика 4. Метода *generateDefaultUserTask* класе *NWMultiInstanceUserTaskBehavior*

Када се метода покрене, користећи *pluginData* приступамо *NWUserManagement*. Прослеђивањем корисничког имена у *findUserByName* методу, довлачимо све податке о датом кориснику. У наредном кораку се креира *NWDataObject* који садржи информације о форми и подацима који се од корисника очекују да унесе у изгенерисану форму.

Даље је потребно инстанцирати *NWUserTask* и за њега унети тражене податке као што су нпр.:

- *id* - Идентификациони број таска
- *description* - Кратак опис таска
- *name* – Назив таска
- *value* – Вредност таска
- *user* – Корисник од којег се очекује да изврши задатак
- *dataObject* – Подаци о форми коју корисник попуњава када извршава задатак

Наредни корак у *performExecution* методи је да се сваки нови *UserTask* повеже са претходно креираним паралелним спајањем. У овом тренутку је завршено све што је неопходно за један *UserTask* и остаје само да се за њега дефинише подпроцес и додели објекат типа *NWSubExecutor* који ће бити задужен за његово извршавање.

Након што петља прође кроз све кориснике, инстанцира таскове и изгенерише подпроцесе потребно је креирати помоћни евент *NWEndMultiInstanceEvent*. Непосредно после, се чвор спајања повезује на почетак овог евента. Он има једноставан задатак, треба да позове *EndMultiInstanceAnnouncer* који ће обавестити вишеструки задатак да су сви корисници успешно завршили своје задатке и да процес може да настави са даљим извршавањем. Вратимо се на *NWMultiInstanceUserTask* и његову *performExecution* методу.

```
waveExecutor processHandler engineAnnouncer
when: EndMultiInstanceAnnouncer
do: [ :ann |
    waveExecutor flowHandler node:
    elemToExecute outgoingFlows first targetRef.
    waveExecutor tryToExecuteNext: elemToExecute.
].
```

Слика 5. Део класе *NWMultiInstanceUserTaskBehavior* где се чека на *announcement*

У овом исечку кода је приказан *listener* који чека на такозвани *announcement* од стране *EndMultiInstanceAnnouncer*-а. Када добије обавештење да је примио очекивани *announcement*, извршиће део кода који се налази испод у *do* блоку. С обзиром на то да је вишеструки задатак у принципу завршен, обавештавамо *FlowHandler* да су подпроцеси завршени и да је на ред извршавања дошао излазни ток од посматраног мулти-инстанц таска. Позива се затим *tryToExecuteNext*, и ту је целокупна активност званично завршена и прелази се на следећу.

2.2. Кориснички интерфејс - фронтенд апликација

Након што улоговани корисник изабере да жели да креира нови вишеструки задатак, изгенерисаће се једноставан *UI*(кориснички интерфејс) који ће му то омогућити. Са сервера је неопходно додати податке

о свим корисницима и ово је могуће учинити користећи методу *getAllUsers* класе *FUGetUserServer*. Податке прихватамо са серверу у форми *JSON* стринга. Наведени код са слике нам омогућава да *JSON* објекте ишчитамо у форми тока (енгл. *stream*). Добијени ток се потом мапира на низ објеката типа *NWUser*. Метода враћа податке у облику уређене колекције (енгл. *ordered collection*).

```
getAllUsers
| entity obj |
entity := (ZnEasy get:
'http://localhost:8081/allusers/') entity string.
entity := (NeoJSONReader fromString: entity).
obj:= (NeoJSONReader
on: entity readStream)
mapInstVarsFor: NWUser;
for: #ArrayOfNWUsers
customDo: [
: mapping | mapping listOfElementSchema: NWUser ];
nextAs: #ArrayOfNWUsers.

"obj := (NeoJSONReader fromString: entity)."
^ obj asOrderedCollection.
```

Слика 6. Метода *getAllUsers* класе *FUGetUserServer*

Након што се добаवे неопходни подаци, рендерује се једноставна форма са *MultiSelect* компонентом. У њој се налази листа имена корисника којима је могуће доделити вишеструки задатак.

Истовремено је могуће изабрати више корисника. Након што се корисници изаберу, кликне се на дугме *Submit* и на тај начин активира *callback*. Позива се метода *commitMultiInstanceTask* класе *FUCommitMultiInstanceTask*, где се као улазни параметар прослеђују корисничка имена селектованих корисника.

Захтев са стране корисничког интерфејса апликације стиже на сервер, где се активира метода *addNewMultiInstanceTask*. Улазни подаци се враћају назад у форми колекције. Првобитно се креира *NWStartEvent* који означава почетак рада процеса. Затим инстанцирамо *NWMultiInstanceUserTask* и додељујемо му колекцију корисника који су пристигли са корисничког интерфејса апликације.

3. ПРИКАЗ РАДА АПЛИКАЦИЈЕ

Након успешног логовања, корисник ће се усмерити ка главној страници са табеларним приказом свих прихваћених, а неизвршених таскова улогованог корисника. У заглављу странице може се уочити дугме 'Create Multi Task', левим кликом на њега прелазимо на страницу за креирање вишеструког задатка.

У *dropdown*-у су приказана корисничка имена пет корисника која су у сврхе демо-а генерисана приликом покретања апликације.

Слика 7. Страница на којој се може креирати нови вишеструког задатка за изабране кориснике.

Једноставном селекцијом једног или више понуђених корисника из листе и кликом на дугме 'Submit' послаћемо захтев серверу да генерише задатак за сваког од изабраних корисника. За овај пример биће селектовани корисници: *user2*, *user3* и *user5*. Потом, логовањем као *user2* може се установити да ли је за њега креиран одговарајући *UserTask* или није. Може се одмах уочити да задатак јесте креиран за овог корисника, понављањем поступка и за остала два, установљено је да су сви кориснички задаци, за сваког изабраног корисника, успешно инстанцирани.

Слика 8. Почетна страница корисника *user2* са новокреираним таском

Такође можемо из транскрипта унутар *PharoLauncher*-а уочити да се заиста јесу креирала три подпроцеса за *UserTask*-ове у оквиру вишеструког задатка, слика 9.

Слика 9. Транскрипт у *PharoLauncher*-и сапокретним подпроцесима

4. ЗАКЉУЧАК

Тема рада била је је имплементација модула за подршку креирања и извршавања вишеструких задатака у оквиру процесног окружења *NewWave*.

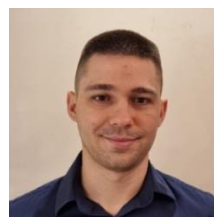
У раду је дат кратак преглед основних концепата софтверских система који пружају подршку управљању пословним процесима. Описани су основни ентитети који дефинишу процесе, и појашњен концепт вишеструких задатака. Код је написан у програмском језику *Фаро*, а за извршно окружење рада је коришћена *PharoLauncher* апликација.

Даљи правац развоја апликације може бити имплементација и модула који ће омогућити рад са секвенцијалним вишеструким тасковима. Такође би било интересантно и корисно када би се имплементирао графички приказ тока једног процеса. Самим тим би се кориснику омогућило да у реалном времену има јасну слику тока процеса и свих подпроцеса.

5. ЛИТЕРАТУРА

- [1] <https://www.techtarget.com/searchcio/definition/business-process> [Пристапљено 12.08.2022.]
- [2] *What Is a Workflow Engine* <https://www.ibm.com/cloud/blog/workflow-engine> [Пристапљено 15.09.2022.]
- [3] *Workflow Management Coalition Terminology & Glossary*, Workflow Management Coalition, 2 Crown Walk, Winchester, Hampshire SO23 8BB, United Kingdom, Фебруар 1999
- [4] Stéphane Ducasse and Gordana Rakic with Sebastijan Kaplar and Quentin Ducasse, *Pharo 9 by Example*, Октобар 2021
- [5] Sebastijan Kaplar, Miroslav Zarić, Gordana Milosavljević, *NewWave Workflow Engine*, Cologne, Germany, Август 2019

Кратка биографија:



Никола Ђорђевић је рођен 10.10.1996. у Београду, Република Србија. Основну школу „Петрефи Шандор“ завршио је 2011. Године. Након тога је уписао гимназију „Јован Јовановић Змај“ у Новом Саду, природно-математички смер. Гимназију је завршио 2015. године са одличним успехом, и исте године се уписује на Факултет техничких наука у Новом Саду, одсек Рачунарство и Аутоматика. Основне студије је завршио 2019. Године

OBRADA VELIKIH KOLIČINA METEOROLOŠKIH PODATAKA**PROCESSING LARGE AMOUNTS OF WEATHER DATA**Nikola Rončević, *Fakultet tehničkih nauka, Novi Sad***Oblast – RAČUNARSTVO I AUTOMATIKA**

Kratak sadržaj – U ovom radu prezentujemo moderan pristup za obradu velikih količina podataka koristeći tehnologije računarstva u oblaku. U ovom radu obradu radimo nad meteorološkim podacima, ali se principi mogu primeniti na različite domene.

Ključne reči: Velike količine podataka, Obrada podataka, Kloud

Abstract – In this article we present modern approach to processing large amounts of data utilizing cloud technologies. In this article the processing is done on weather data but the principles stand for various domains

Keywords: Big data, Data processing, Cloud

1. UVOD

U današnjem, digitalnom vremenu količina podataka koji su nam dostupni je sve veća. Sa porastom podataka raste i potreba za analizom istih radi izvođenja različitih zaključaka koji mogu doneti benefite. Činjenica je da je količina podataka u stalnom porastu i da će podataka koje treba obraditi biti sve više. Samim tim i rastu potrebe za unapređenjem sistema koji se bave obradom istih.

U ovom radu fokus je na jednom problemu obrade velikih količina podataka. Konkretno, u radu je prikazana analiza meteoroloških podataka koji se sakupljaju iz različitih izvora, transformišu se u odgovarajući format, vrši se analiza i na osnovu analize izvode određeni zaključci. Takođe, pripremićemo podatke na takav način da olakšamo buduće analize nad istima.

Tehnologije koje će biti korišćene u svrhu postizanja ovih rezultata su sledeće:

Azure functions [1] - koristićemo ih za prikupljanje podataka sa Interneta.

Azure data lake gen 2 [2] - koristićemo ga za čuvanje kako sirovih podataka tako i određenih podataka koji su pretvoreni u format prihvatljiviji za dalju obradu

Azure databricks [3] - će biti korišćen za pokretanje grupa kompjutera na kojima će se obrađivati podaci upotrebom spark-a.

SQL database [4] - u ovoj bazi podataka će se čuvati podaci u formatu adekvatnom za dalje analize u takozvanoj star/snowflake šemi podataka

PowerBI [5] - za prezentaciju podataka

2. PREGLED SLIČNIH SISTEMA

Sistemi za obradu velikih količina podataka su sve zastupljeniji. Podataka je sve više, obrada istih je veoma korisna i može doneti mnoge benefite. To su

- Veći profiti
- Unapređena sigurnost unutar sistema
- Identifikovanje šablona ponašanja kod korisnika
- Optimizacija troškova
- Unapređene performanse organizacije uviđanjem problematičnih tačaka
- Identifikovanje anomalija u realnom vremenu.

Sistemi koji obrađuju velike količine podataka su mnogobrojni i mnogi od njih nisu zasebni, već su deo već postojećih sistema koji se bave različitim problemima. Ovi sistemi se dele na:

- Sistemi koji obrađuju podatke u realnom vremenu (stream processing)
- Sistemi koji obrađuju skupove podataka (batch processing)

2.1. Sistemi koji obrađuju podatke u realnom vremenu

Sistemi koji obrađuju podatke u realnom vremenu dizajnirani su tako da prihvataju, obrađuju i isporučuju rezultate odmah nakon što podaci stignu. Ovi sistemi su ključni u scenarijima gde je potrebna trenutna obrada i reakcija na dolazne podatke. Karakteristika ovih sistema su:

Trenutna obrada - kako podaci stižu oni se odmah obrađuju, ove obrade su dosta prostije od obrada koje se rade u sistemima za obradu skupova podataka iz razloga što postoji vremensko ograničenje.

Minimalno kašnjenje - vremenski raspon između prijema i obrada podataka je obično od par milisekundi do najviše nekoliko sekundi.

Kontinualnost - sistem radi bez prekida.

2.2. Sistemi koji obrađuju podatke u skupovima

Kako su količine podataka kroz proteklih par decenija eksponencijalno porasle, potrebe za obradom istih podataka su sve veće. Moderni kompjuteri nisu u stanju da samostalno obrade ovakve količine podataka. Iz tog razloga su se javili novi sistemi koji obrađuju podatke u skupovima (batch processing systems). Ono što je karakteristično za ove sisteme je pre svega:

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Miroslav Zarić, red. prof.

- Skalabilno procesiranje podataka - ovi sistemi mogu da rastu horizontalno (horizontal scaling) što znači da umesto da uvećavamo procesorsku moć jednog kompjutera mi umesto toga dodajemo veći broj mašina koji zajedno radi procesiranje podataka.
- Integracija podataka (Data integration) - Ovakvi sistemi obično obrađuju podatke iz različitih sistema, u različitim formatima i pre nekih ozbiljnijih obrada je uvek potrebno prvo očistiti i transformisati ove podatke
- Tolerancija na greške (Fault tolerance) - veoma je bitno da ovi sistemi budu tolerantni na greške iz razloga što često procesi znaju da traju i danima, te bi fatalna greška mogla da bude kobna i veoma skupa.
- Optimizovani - zbog mogućnosti horizontalnog rasta ovi sistemi su veoma dobro optimizovani da isti iskoriste najefikasnije, ukratko sve aktivnosti koje je moguće raditi paralelno ovi sistemi izvršavaju na taj način, ovo će biti detaljnije objašnjeno u nastavku.

3. KORIŠĆENE TEHNOLOGIJE

U ovoj sekciji dat je detaljniji pregledati tehnologije koje su korišćene za implementaciju našeg sistema.

3.1. Azure funkcije

Serverless računarstvo je model računarstva koji omogućava da se izvršavanje aplikacija radi bez brige o arhitekturi u pozadini uključujući servere, održavanje istih kao i skaliranje. Osnovne karakteristike Azure funkcija su:

- Reaguju na događaje
- Skalabilne
- Integrisana sigurnost
- Mogućnost pamćenja stanja

3.2. Azure data lake storage gen2

Azure data lake storage gen2 je unapređena verzija Azure data lake storage gen1 sistema kao i Azure Blob Storage sistema. Kao rezultat ovoga dobijamo globalno skalabilan, siguran i visoko dostupan sistem za skladištenje podataka, koji nam omogućava da izvršavamo masivno paralelne analitike. Osnovne karakteristike Azure data lake gen2 su:

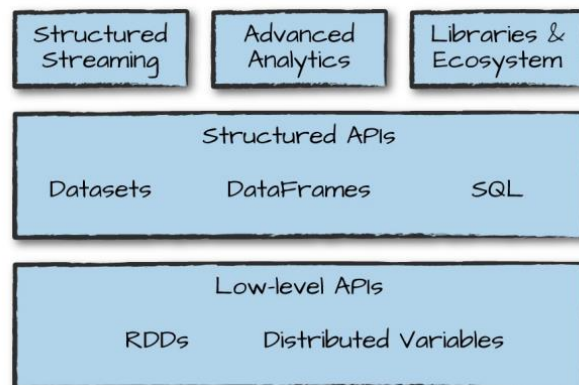
- Hijerarhijska struktura direktorijuma
- Skalabilno i performantno
- Bezbednost na visokom nivou
- Redundantnost podataka (Lokalna, Zonska, Regijska)
- Adekvatni nivoi skladišta srazmerno

3.3. Apache spark

Spark je okruženje i kolekcija biblioteka za paralelno procesiranje podataka na grupama kompjutera, slika 1. U ovom momentu Spark je najpopularniji javno dostupan i razvijen sistem za paralelno procesiranje velikih količina podataka.

Apache Spark započeo je sa razvojem 2009. godine kao istraživački projekat na univerzitetu UC Berkley. U to vreme dominantna tehnologija za procesiranje velikih količina podataka je bio Hadoop Map Reduce. Tim koji je razvija Apache Spark je napravio jedan od najvećih pomaka u domenu obrade velikih količina podataka

konceptom upotrebe RAM memorije zarad privremenog čuvanja podataka.



Slika 1. Pregled kompletnog sistema spark-a

3.3.1. Arhitektura Apache spark aplikacija

Apache Spark aplikacije sastoje se iz **driver** i **executor** procesa, slika 2.

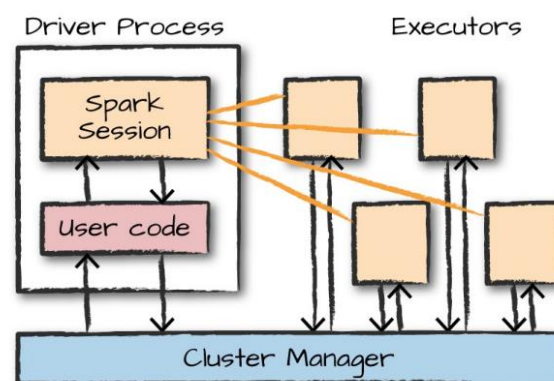
Driver proces je zadužen za pokretanje programa, nalazi se na jednom od kompjutera u grupi kompjutera, i zadužen je za tri stvari:

- Čuvanju informacija o Spark aplikaciji
- Komunikaciju sa korisnikom ukoliko u okviru procesa postoji potreba za time
- Analiziranju, raspoređivanju i zakazivanju procesa na ostalim kompjuterima u grupi kompjutera

Executor proces je zadužen za izvršavanje zadataka koje mu driver proces zada. Ovo znači da executor proces ima samo dva zadatka:

- da izvrši ono što mu se zada
- kada to uradi da javi rezultate driver procesu.

Grupom kompjutera koji će izvršavati Spark aplikaciju, upravlja cluster manager.



Slika 2. Pregled izvršavanja Apache Spark aplikacije

3.3.2. Apache spark Data frame

DataFrame je osnovna gradivna jedinica Spark API-ja. On predstavlja tabelu podataka sa redovima i kolonama. Ono što je bitno razumeti o DataFrame API-ju je to da je on distribuiran, odnosno DataFrame predstavlja logički naše

podatke, dok se oni u realnosti nalaze na grupi kompjutera razbacano. Podaci mogu biti podeljeni po kompjuterima po različitim parametrima i uslovima i ova podela se zove particionisanje podataka.

3.3.3. Apache spark Data frame

Da bi svaki executor mogao da izvršava svoj posao u paraleli, Spark razbija podatke u grupe koje se nazivaju particije. Particija je kolekcija redova podataka iz jednog DataFrame-a koji se nalaze na jednoj mašini.

Ukoliko imamo samo jednu particiju mogućnost paralelizma ne postoji. Ako imamo 5 particija podataka i 10 mašina, mogućnost paralelizma je 5, a ako pak imamo 100 particija i 10 kompjutera mogućnost paralelizma je 10. Ono što se iz ovoga može izvući je to da uvek treba imati barem onoliko particija koliko kompjutera se nalazi u našem skupu kompjutera, a često i više.

3.3.4. Transformacije nad podacima

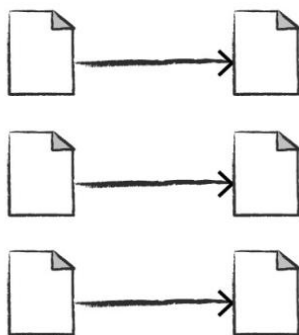
Particionisanje je ono što omogućava da podatke obrađujemo paralelno, ali šta tačno znači obrada podataka?

Obrada podataka u okviru Spark-a i generalno u sferi obrade velikih količina podataka znači isto što i inače, to mogu biti transformacije određenih kolona, dodavanje novih kolona kombinacijom postojećih, spajanje više tabela podataka, filtriranje podataka po određenim parametrima i slično.

Ono što je bitno da razumemo u vezi transformacija nad podacima je da se one mogu podeliti u dve grupe: Narrow transformacije i Wide transformacije podataka.

Narrow transformacije podataka možemo izvršiti na jednom kompjuteru dok Wide transformacije najčešće zahtevaju da kombinujemo podatke sa različitih kompjutera u skupu kompjutera, slika 3.

Narrow transformations
1 to 1



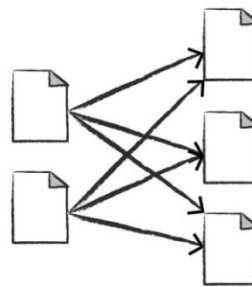
Slika 3. Narrow transformacije

Wide transformacije su komplikovanije, zato što one zahtevaju da se podaci sa različitih mašina sada nađu na jednoj, slika 4.

4. Sistem koji je implementiran

Zamisao ovog rada jeste da meteorološke informacije objedinimo na jednom mestu i pripremimo ih za dalju obradu uz poštovanje najboljih praksi izrade ovakvih sistema.

Wide transformations
(shuffles) 1 to N

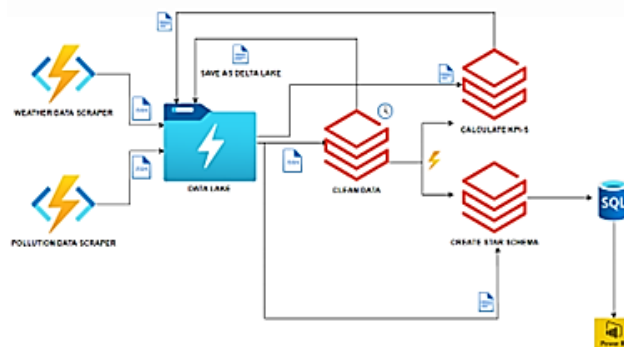


Slika 4. Wide transformacije

Takođe će biti demonstrirani neki od načina na koji ovi podaci mogu biti iskorišćeni ali to nije fokus ovog rada. Ove podatke treba da koriste domenski eksperti koji znaju tačno šta i kako žele da uvide, u okviru oblasti obrade velikih količina podataka, naš fokus je da im to omogućimo na što bolji i efikasniji način.

4.1 Weather data scrapper i pollution data scrapper

Uloga skupljača meteoroloških podataka (*weather data scrapper*) i skupljača podataka o zagađenju (*pollution data scrapper*) jeste ta da prikupljaju informacije o trenutnim vremenskim uslovima, kao i o zagađenjima na određenim lokacijama, slika 5. U našem slučaju to su različiti gradovi Srbije.



Slika 5. Pregled sistema

4.2 Data lake

Azure data lake gen 2 je rešenje za skladištenje velikih količina podataka, koje pruža Microsoft i mi ga koristimo za naš Data Lake. U njemu ćemo čuvati podatke u raznim formatima. Inicijalno se čuvaju podaci u formatu u kakvom se i dobavljaju. Kasnije, mi izvršavamo čišćenje ovih podataka kao i određene transformacije nad istima. Transformisane podatke, kao i određene kalkulacije koje želimo da budu javno dostupne, ćemo nakon obrade opet čuvati u našem Data Lake.

4.3 Clean dana

U ovom stadijumu našeg sistema vršimo čišćenje podataka kao i određene transformacije. Ovaj proces će se izvršavati na grupi kompjutera (cluster) koji će biti pokrenuti u okviru Azure databricks radnog prostora. Nakon ovih transformacija podaci će opet biti sačuvani nazad u Data Lake.

4.4 Calculate KPI's

U ovom segmentu sistema, koristeći očišćene podatke izvodimo određena standardna izračunavanja nad podacima. Ove podatke na kraju opet čuvamo u Data Lake gdje im se može pristupiti javno, a i mogu se koristiti za kasnije analize i obrade.

4.5 Create star scheme

U ovom koraku se fokusiramo na kreiranje star scheme, koja predstavlja očišćene podatke u normalizovanoj formi, ovi podaci se zatim čuvaju u SQL bazi podataka u tabelama. Pored njih u bazi podataka se nalaze i dodatne tabele o kojima će kasnije biti više rečeno.

4.6 Power BI graph

Nakon čuvanja podataka u bazi podataka uz pomoć Power BI ćemo pokazati na koji način možemo iskoristiti podatke iz baze podataka za dalje i detaljnije analize.

5. ZAKLJUČAK

Konstantno povećavanje količine podataka u digitalnom svetu nas tera da nalazimo nove načine da obrađujemo ove podatke. Postoji sve više moćnih rešenja za procesiranje, sakupljanje, čuvanje, analizu i vizualizaciju istih. Microsoft Azure nudi integrisana rešenja koja se mogu iskoristiti efikasno u ove svrhe, kao što je i demonstrirano u dosadašnjem radu.

Uspešna integracija Azure funkcija, Azure Data Lake-a, Azure Databricks-a, Azure SQL-a i PowerBI-a pokazuje da je ovaj ekosistem veoma kvalitetan.

6. LITERATURA

- [1] Azure functions - <https://learn.microsoft.com/en-us/azure/azure-functions/>
- [2] Azure data lake gen 2 - <https://learn.microsoft.com/en-us/azure/storage/blobs/data-lake-storage-introduction>
- [3] Azure databricks - <https://learn.microsoft.com/en-us/azure/databricks/>
- [4] SQL Database - <https://www.programiz.com/sql/database-introduction>
- [5] Power BI - <https://learn.microsoft.com/en-us/power-bi/>
- [6] Apache Flink - <https://nightlies.apache.org/flink/flink-docs-stable/>
- [7] Apache Spark (Spark streaming) - <https://spark.apache.org/docs/latest/streaming-programming-guide.html>
- [8] Apache Spark - <https://spark.apache.org/docs/3.4.1/>

Biografija

Nikola Rončević rođen je 2. III 1998. godine u Novom Sadu. Završio je osnovnu školu Jovan Popović u Novom Sadu kao i Gimnaziju Jovan Jovanović Zmaj. Osnovne akademske studije je završio na Fakultetu Tehničkih Nauka u Novom Sadu sa prosekom 9.17.

Radi u firmi Inviggo kao programer poslednje tri godine gde se bavi sistemima za obradu velikih količina podataka kao i razvojem sistema za banke iz sfere otvorenog bankarstva.

BEŽIČNO UPRAVLJANJE I KONTROLA PROGRAMABILNOG VOZILA PUTEM VEB I ANDROID APLIKACIJE**WIRELESS CONTROL OF PROGRAMMABLE VEHICLE VIA WEB AND ANDROID APPLICATION**Tamara Macura, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U ovom radu je opisano razvijanje celokupnog sistema za upravljanje vozilom. Iskorišćen je već postojeći hardverski uređaj za koji je pisana nova C aplikacija. Takođe su napisane dve korisničke aplikacije, uz pomoć kojih će korisnici upravljati krajnjim uređajem. Razvijena je jedna veb aplikacija, pisana JavaScript programskim jezikom i jedna Android aplikacija, pisana Java programskim jezikom. Ove dve aplikacije imaju potpuno iste funkcionalnosti.

Ključne reči: mikrokontroler, veb aplikacija, Android aplikacija, Bluetooth

Abstract – This project describes the development of the vehicle control system. The existing hardware device with new C application, was used. Two user applications have also been written, with the help of which users will control the vehicle. Web application was written in JavaScript programming language and Android application was written in Java programming language. The two applications have exactly the same functionality.

Keywords: microcontroller, web application, Android application, Bluetooth

1. UVOD

Pojam „Internet stvari” ili „Internet inteligentnih uređaja” (eng. *Internet of Things*, IoT) više nije stran, a podrazumeva međusobno umrežavanje fizičkih objekata - „stvari” u koje su ugrađeni senzori, aktuatori i softver u svrhu razmene podataka između njih, kao i njihovog kontroliisanja od strane ljudi, a sve to preko interneta [1].

Danas je nezamislivo da jedan savremen čovek ne poseduje i ne koristi „pametni” telefon, pretežno sa *Android* operativnim sistemom na njemu, koji stoga podržava *Android* aplikacije. *Android* aplikacije obezbeđuju raznovrsne funkcionalnosti u različitim sferama kako poslovanja, tako i zabave.

Zadatak ovog rada jeste da se realizuje sistem za upravljanje vozilom preko veb aplikacije i *Android* aplikacije. Celokupan sistem se sastoji iz hardverskog i softverskog dela. Hardverski deo čini pločica izrađena na projektu iz predmeta *Primenjena elektronika* na osnovnim studijama, dok softverski deo čine C aplikacija koja direktno komunicira sa pločicom i korisnička (web ili *Android*)

aplikacija pomoću koje se šalju naredbe za upravljanje vozilom. Iz C aplikacije naredbe se šalju mikrokontroleru, a putem *Bluetooth* komunikacije ostvarena je veza između korisničkih aplikacija i istog mikrokontrolera.

Rešenje ovog rada bazira se pre svega na mikrokontroleru *dsPIC30F4013* sa kojim komuniciraju kako C, tako i korisničke aplikacije. U programskom alatu *Altium Designer* realizovana je električna šema i PCB same pločice, nakon čega je ona izrađena. C aplikacija je realizovana u programskom paketu *MPLAB X* [2], a za programiranje mikrokontrolera korišćen je programator *PicKit3* [3]. Za realizaciju veb aplikacije korišćena su razvojna okruženja *Node.js* [4], za *backend* deo i *Vue.js* [5], za *frontend* deo, pisani *JavaScript* programskim jezikom. Za *Android* aplikaciju korišćeno je razvojno okruženje *Android Studio* i *Java* programski jezik. Ideja projekta jeste da se realizuje vozilo za lako i bezbedno upravljanje.

2. KONCEPT REŠENJA

Za hardverski uređaj izabrana je pločica koja podržava funkcionalnosti upravljanja kretanjem vozila. Podešavanjem PWM signala na odgovarajućim ulazima drajvera za motor *L298N* kontroliše se kretanje vozila. Komande se šalju od strane mikrokontrolera *dsPIC30F4013*.

Što se tiče kretanja vozila, ono može da ide napred i nazad i da se okreće u levu ili u desnu stranu. Vozilo ne može istovremeno, na primer, da se okreće u desnu stranu i ide napred, dakle potrebno je da se zaustavi, okrene za određeni ugao i nastavi u novom smeru da se kreće pravo. *Bluetooth* komunikacija se koristi za uspostavljanje veze između korisničkih aplikacija i vozila, tj. mikrokontrolera, koristeći *HC-06* modul. Iz korisničke aplikacije šalju se naredbe o načinu kretanja koje mikrokontroler prihvata i na osnovu njih šalje naredbe motoru vozila i upravlja njime na željeni način.

Veb aplikaciju sačinjavaju serverska i klijentska aplikacija. Na slici 1 prikazana je međusobna povezanost tri softverska dela sistema. Vidi se da C aplikacija i serverska aplikacija komuniciraju putem *Bluetooth* veze, a serverska i klijentska korišćenjem *HTTP* protokola.

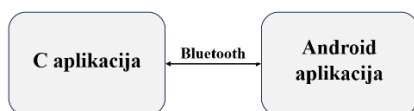


Slika 1. Komunikacija između C i veb aplikacije

Kod *Android* korisničke aplikacije ne postoje klijentski i serverski deo, nego je sve obezbeđeno iz jednog dela. Takođe, *Android* aplikacija sa C aplikacijom komunicira putem *Bluetooth* veze.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Predrag Teodorović, docent.



Slika 2. Komunikacija između C i Android aplikacije

2.1. Veb aplikacija

Kod veb aplikacije, **serverska** aplikacija ima tri uloge:

1. da uspostavi *Bluetooth* vezu sa C aplikacijom,
2. da uspostavi vezu sa korisničkom aplikacijom putem HTTP protokola,
3. da čuva podatke o putanjama u nerelacionoj bazi podataka.

Klijentska aplikacija je znatno složenija od serverske aplikacije. U njoj je ostvarena interakcija sa korisnikom. Ona sadrži jednu glavnu, početnu stranicu i dve sporedne, u koje je moguće otići iz glavne. U okviru početne stranice korisnik bira na koji način želi da upravlja vozilom, tj. na koji način želi da igra igricu i na osnovu tog odabira otvara mu se jedna od dve sporedne stranice. U svakoj od sporednih stranica postoji mogućnost povratka na početnu stranicu.

Načini na koje je moguće upravljati vozilom:

1. Upravljanje pritiskom na tastere koji se vide na ekranu

računara ili pritiskom na strelice tastere na tastaturi računara – postoje 4 tastera, dva za kretanje unapred i unazad i dva za okretanje ulevo i udesno.

Kada korisnik klikne na taster koji se vidi na ekranu, pokrene se odgovarajuća metoda koja putem HTTP protokola šalje informaciju o pritisnutom tasteru serveru, koji potom putem *Bluetooth* komunikacije obaveštava C aplikaciju koji taster je pritisnut i u kom smeru treba da se kreće ili okreće vozilo. Vozilo se kreće 1500ms u zadatom smeru ili se okreće, takođe 1500ms, što odgovara pravom uglu, tj. vozilo će promeniti pravac kretanja za 90 stepeni.

Kada se upravlja vozilom tasterima sa tastature, moguće je detektovati pritisak sve vreme dok je taster pritisnut. Jedan korak vozila traje 300ms i koraci se ponavljaju jedan za drugim dokle god korisnik drži pritisnut taster, pa se dobija prividno kontinualno kretanje vozila.

2. Zadavanjem putanje po kojoj će se vozilo kretati, tj. koraka koje će vozilo izvršiti prilikom svog kretanja – na ekranu se nalaze četiri polja gde se odabira broj koraka koliko korisnik želi da se vozilo kreće ili okreće u određenom smeru. Kada korisnik ukuca celu putanju, pritiskom na određeni taster potvrđuje ono što je uneo i putanja se u vidu stringa šalje putem HTTP protokola serverskoj strani. Na serverskoj strani se deli na korake i korak po korak šalje C aplikaciji, kao i u prethodnim slučajevima. Kretanje vozila takođe traje 1500ms, a zaokret ulevo ili udesno iznosi 90 stepeni. Pored tastera koji potvrđuje unetu putanju, nalazi se taster koji briše poslednji uneti korak i taster koji briše sve korake. Koraci koji su uneti ispisani su na ekranu, kako bi korisnik imao uvid u ono što je uneo i što će poslati kao naredbu. Pored toga, unetu putanju je moguće sačuvati i kao takvu iskoristiti ponovo ili modifikovati. Pre nego što korisnik pošalje putanju, biva pitan da li želi da je sačuva i ta informacija se takođe šalje serveru, i shodno tome se u

bazi podataka putanja čuva ili ne. Sačuvane putanje je moguće izbrisati iz baze, za šta se koristi poseban taster.

2.2. Android aplikacija

Android aplikacija funkcioniše nešto drugačije od veb aplikacije. Ona takođe ima tri osnovne uloge:

1. da ostvari interakciju sa korisnikom,
2. da uspostavi *Bluetooth* vezu sa C aplikacijom,
3. da čuva podatke u relacionoj bazi podataka.

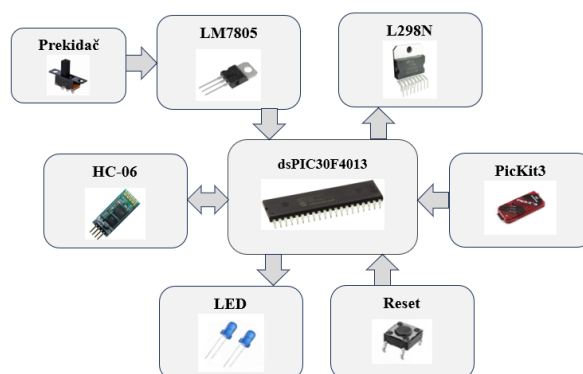
Kao i veb aplikacija, *Android* aplikacija sadrži jednu glavnu i dve sporedne i njihova međusobna povezanost je ista kao i kod veb aplikacije.

1. Ako korisnik odabere da upravlja vozilom pritiskom na tastere, otvara mu se prozor sa 4 tastera za upravljanje i jednim tasterom za povratak na početnu stranicu. Kada korisnik klikne na odgovarajući taster pokrene se odgovarajuća metoda koja putem *Bluetooth* komunikacije šalje informaciju o pritisnutom tasteru C aplikaciji. Ovde je za razliku od veb aplikacije izbegnut server i komunikacija putem HTTP protokola. U zavisnosti od pritisnutog tastera vozilo se kreće 1500ms u zadatom smeru ili se okreće, takođe 1500ms, što odgovara pravom uglu, tj. vozilo će promeniti pravac kretanja za 90 stepeni.

2. Ako korisnik izabere da upravlja vozilom unapred zadajući putanju po kojoj će se vozilo kretati, otvara mu se prozor sličan prozoru veb aplikacije. Dakle na ekranu se vide četiri polja gde se odabira broj koraka koliko korisnik želi da se vozilo kreće ili okreće u određenom smeru. Kada korisnik ukuca celu putanju, pritiskom na određeni taster potvrđuje ono što je uneo i putanja se najpre deli na korake i korak po korak šalje putem *Bluetooth* komunikacije C aplikaciji. Kretanje vozila takođe traje 1500ms, što predstavlja ni previše mali, ni previše velik korak, kako bi se sa lakoćom moglo upravljati vozilom. Takođe, zaokret ulevo ili udesno iznosi 90 stepeni i čini jedan korak. Kao i kod veb aplikacije prisutni su tasteri koji brišu poslednji uneti korak i sve korake istovremeno. Takođe, koraci koji su uneti ispisani su na ekranu, kako bi korisnik imao uvid u ono što je uneo i što će poslati kao naredbu. Unetu putanju je, takođe, moguće sačuvati, ovog puta u relacionoj bazi podataka, i kao takvu iskoristiti ponovo, modifikovati ili pak obrisati.

3. HARDVERSKI DEO SISTEMA

Na slici 3 prikazana je blok šema hardverskog uređaja sa svim delovima koji ga čine. Hardverski deo bazira se na korišćenju mikrokontrolera *dsPIC30F4013* koji predstavlja njegov glavni deo.



Slika 3. Blok šema hardverskog dela sistema

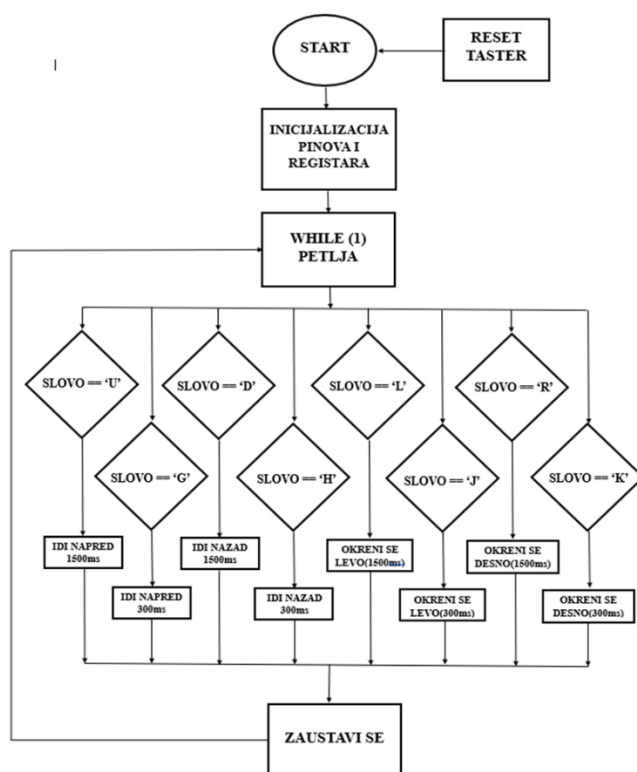
Pored toga, od ostalih neophodnih komponenata prisutni su regulator napona *LM7805*, drajver za motor *L298N*, *Bluetooth* modul *HC-06*.

Komponente koje su dodate radi lakšeg korišćenja sistema su prekidač pomoću kojeg se uključuje i isključuje ceo sistem, taster za reset sistema, kao i dve indikatorske diode koje signaliziraju da li je sistem uključen i da li je drajver za motor uključen. Programski kod se na mikrokontroler spušta pomoću programatora *PicKit3*.

4. SOFTVERSKI DEO SISTEMA

4.1. C aplikacija

Na slici 4 prikazan je algoritam rada „main” funkcije C aplikacije.



Slika 4. Algoritam rada „main” funkcije C aplikacije

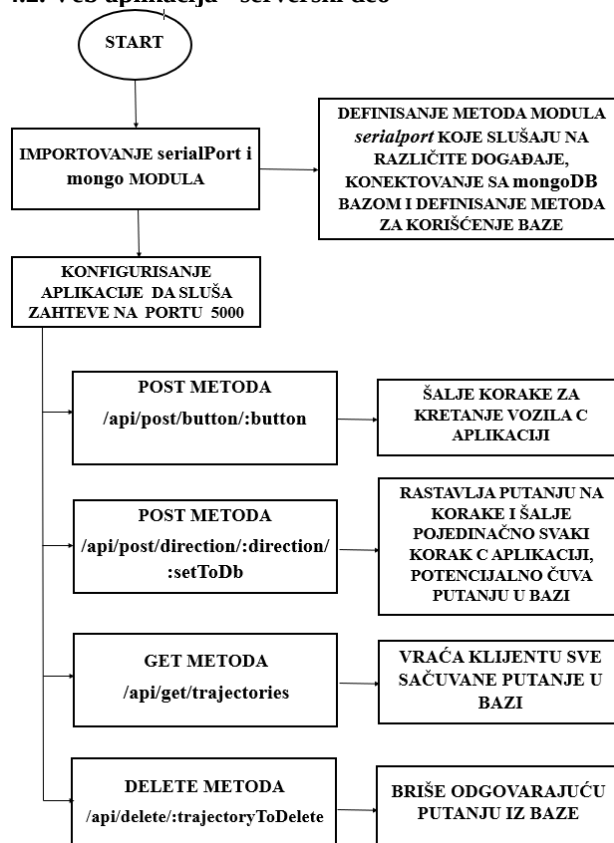
Funkcija počinje tako što se najpre podese i inicijalizuju svi neophodni registri i pinovi za pravilno korišćenje hardverskog uređaja. Nakon toga ulazi se u „while” petlju koja se stalno ponavlja, i proverava slova koja su pristigla od strane korisničke aplikacije, putem *Bluetooth* komunikacije.

Pošto korisnička aplikacija koja upravlja vozilom može da se koristi na dva načina bilo je potrebno napraviti dva različita pomeraja vozila, jedan od 1500ms, a drugi od 300ms, gde se praktično ne vidi pauza između koraka već se dobija utisak da se vozilo kreće kontinualno.

Na osnovu dužine pomeraja koje vozilo treba da uradi korisnička aplikacija šalje različita slova i na osnovu toga vozilo čini odgovarajući pomeraj.

Na kraju svakog pomeraja vozilo se zaustavi, što predstavlja i kraj tekuće iteracije beskonačne petlje, koja ponovo kreće ispočetka.

4.2. Veb aplikacija - serverski deo



Slika 5. Algoritam rada serverske aplikacije

Programski kod glavnog modula serverske aplikacije učitava se sekvencijalno, najpre se učitavaju moduli *serialPort* koji omogućava komunikaciju putem *Bluetooth* veze i *mongo*, koji omogućava pristup i kontrolisanje baze podataka.

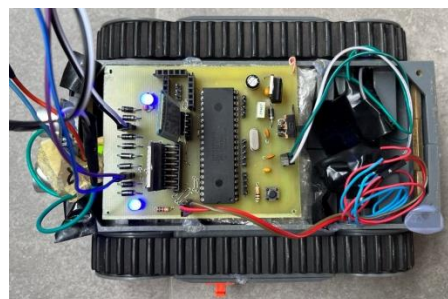
Zatim se server konfiguriše da sluša zahteve na portu 5000. Na prvi POST zahtev server će C aplikaciji poslati odgovarajući korak za koji vozilo treba da se pokrene. Na drugi POST zahtev putanja koju primi server se rastavlja na korake koji se potom pojedinačno šalju C aplikaciji.

Ako parametar *setToDb* ima vrednost *true* u *MongoDB* bazi sačuvace se podatak o primljenoj putanji.

Metoda GET klijentu vraća listu sa svim sačuvanim putanjama, a DELETE metoda briše iz baze odgovarajuću putanju. Korisničke aplikacije biće opisane kroz rezultate.

5. REZULTATI

Na slici 6 prikazana je hardverska pločica i samo vozilo koje u ovom slučaju ona pogoni.



Slika 6. Izgled vozila sa pločicom

5.1. Veb aplikacija

Na slici 7 prikazana je početna stranica, iz koje je moguće preći u jednu od dve sporedne.



Slika 7. Početna stranica korisničke veb aplikacije

Na slici 8 prikazana je stranica kada korisnik želi manu- elno da upravlja vozilom. Na ovoj stranici uočavamo tas- tere pritiskom na koje se šalje određena komanda server- skoj, a potom i C aplikaciji i vozilo se kreće u željenom smeru. Ispod tastera uočava se link pritiskom na koji se korisnik vraća na početnu stranicu, gde ponovo može da bira način igranja igrice.



Slika 8. Sporedna stranica sa koje se manualno upravlja vozilom iz veb aplikacije

Na slici 9 prikazana je stranica kada korisnik zeli da up- ravlja vozilom unapred zadajući putanju kretanja. Na vrhu stranice nalaze se polja u kojima se selektuje željeni broj koraka. Svaki novoodabrani korak biće dopisan u polje za ispis koraka. U bilo kojem trenutku moguće je izbrisati korak koji se nalazi poslednji u listi, pritiskom na taster DELETE STEP ili izbrisati sve korake istovremeno, pri- tiskom na taster DELETE ALL. Ako je korisnik zadovo- ljan odabranim koracima, potrebno je da klikne na taster SUBMIT, kada se otvara prozor gde korisnik može da izabere da li želi da sačuva putanju ili ne pritiskom na od- govarajući taster, u zavisnosti od kojeg će pored toga što će biti poslata serverskoj strani, putanja biti sačuvana u bazi ili ne. Pritiskom na taster DELETE TRAJECTORY može se odabrati, modifikovati ili izbrisati putanju iz baze.



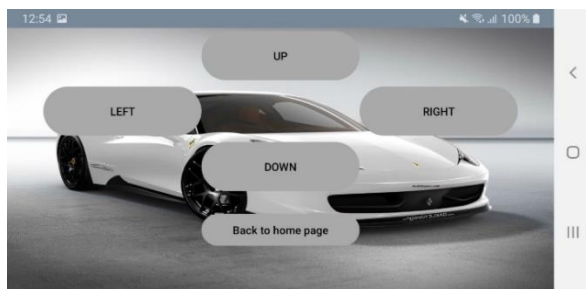
Slika 9. Sporedna stranica iz koje se vozilom upravlja zadajući putanju iz veb aplikacije

5.2. Android aplikacija

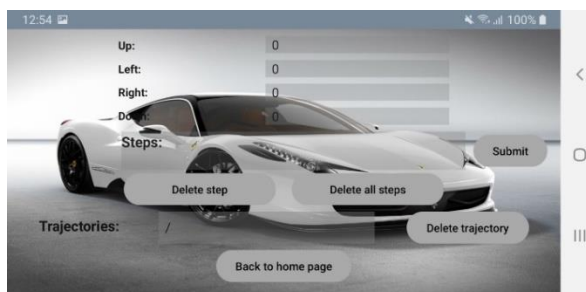
Android aplikacija je veoma slična veb aplikaciji i biće prikazane njena glavna stranica i dve sporedne, a opisi će biti izostavljeni pošto aplikacije funkcionišu na isti način.



Slika 10. Početna stranica korisničke Android aplikacije



Slika 11. Sporedna stranica sa koje se manualno upravlja vozilom iz Android aplikacije



Slika 12. Sporedna stranica iz koje se vozilom upravlja zadajući putanju iz Android aplikacije

6. LITERATURA

- [1] Antić, 2019, Šta je Internet of Things, <https://samoobrazovanje.rs/sta-je-internet-of-things-internet-intelligentnih-uredjaja/>, poslednji pristup oktobar 2023.
- [2] MPLAB® X Integrated Development Environment (IDE), <https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide>, poslednji pristup oktobar 2023.
- [3] PICKIT™ 3 IN-CIRCUIT DEBUGGER <https://www.microchip.com/en-us/development-tool/pg164130>, poslednji pristup oktobar 2023.
- [4] Šta je to Node?, <https://www.skolaprogramiranja.rs/sta-je-to-node/>, poslednji pristup oktobar 2023.
- [5] Introduction, Vuejs.org, <https://vuejs.org/guide/introduction.html#what-is-vue>, poslednji pristup oktobar 2023.

Kratka biografija:



Tamara Macura rođena je 1998. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Primenjena elektronika odbranila je u aprilu 2022. godine. Ima dve godine radnog iskustva, najpre u kompaniji RT-RK, a potom u kompaniji Continental. kontakt: tasak998@gmail.com

PODRŠKA ZA PYFLIES JEZIK U VS CODE EDITORU**SUPPORT FOR PYFLIES LANGUAGE IN THE VS CODE EDITOR**Dejan Šorgić, *Fakultet tehničkih nauka, Novi Sad***Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – U ovom radu implementirana je nadogradnja na postojeću ekstenziju za Visual Studio Code okruženje koja pomaže pri pisanju eksperimenata u PyFlies jeziku. U okviru rada je primarno implementiran jezički server, koji pomoću Language Server Protocol-a komunicira sa PyFlies ekstenzijom i tako proširuje njene funkcionalnosti.

Ključne reči: ekstenzija, pyFlies, DSL, Language Server Protocol, Visual Studio Code, jezički server

Abstract – This paper details the implementation of an enhancement to the pre-existing Visual Studio Code extension designed to facilitate experiment scripting in the PyFlies language. The central focus of this study is a language server, which uses Language Server Protocol to communicate with PyFlies extension and with this, it extends its functionalities.

Keywords: Extension, pyFlies, DSL, Language Server Protocol, Visual Studio Code, language server

1. UVOD

Ovaj rad obuhvata dizajn i implementaciju nadogradnje na podršku za Visual Studio Code radno okruženje za PyFlies jezik [1]. Inicijalna verzija PyFlies ekstenzije obuhvata bazične funkcionalnosti, a u okviru ovog rada je implementiran niz kompleksnih funkcionalnosti, kao što su analiza koda i prijava grešaka, skok na definiciju, prikaz referenci i još neke. Za implementaciju pomenutih funkcionalnosti, potrebno je osloniti se na *Language Server Protocol*, i napraviti jezički server. Jezički server i inicijalna verzija ekstenzije, koja u ovom kontekstu predstavlja klijenta, čine dve komponente ekstenzije, koje komuniciraju preko LSP-a.

PyFlies je jezik specifičan za domen, i njegova namena je pisanje kognitivnih testova. Zamišljen je tako da bude lako čitljiv i jednostavan za učenje. Jezik je baziran na *Python* programskom jeziku i sadrži modularnu arhitekturu, kako bi generatori mogli biti razvijeni kao odvojeni projekti. Trenutno dostupni generatori su *csv*, *log* i *psychopy* generator, od kojih su *csv* i *log* generatori ugrađeni u sam *PyFlies* projekat, dok je *psychopy* generator u odvojenom projektu.

PsychoPy [2] je platforma za pisanje i izvršavanje kognitivnih eksperimenata, a *psychopy* generator omogućava generisanje *Python* skripti koje se ovde mogu koristiti. Generisanje *log* i *csv* datoteka na osnovu *PyFlies* eksperimenata se izvršava u cilju analize, pregleda toka eksperimenata i pronalaska grešaka u eksperimentu, u slučaju *log* datoteka, odnosno u cilju ekstrakcije tabele uslova definisane u eksperimentu, u slučaju *csv* datoteka.

Dizajn *PyFlies* jezika je vođen sledećim principima: čitljivost, apstrakcija, koncepti domena, modularna arhitektura i proširivost, najbolje prakse i fokus na strukturu eksperimenata.

Praćenju ovih principa idu u korist činjenice da je *PyFlies* jednosavan za razumeti i lak za naučiti, i to da se za opis eksperimenata koriste apstraktni pojmovi, i usko povezani za domen, bez brige o detaljima implementacije, što umanjuje mentalni napor potreban za pisanje i razumevanje eksperimenata [1].

2. JSD I VS CODE PODRŠKE

Jezici specifični za domen, JSD (en. *Domain Specific Languages, DSL*) predstavljaju programske jezike ograničene ekspresivnosti, dizajnirane za rešavanje specifičnih problema u određenoj oblasti [1]. Ovi jezici konstruišu se tako da odgovaraju specifičnim potrebama domena, pružajući prilagođene koncepte, sintaksu i semantiku za izražavanje ideja i operacija u tom domenu. JSD-ovi menjaju generalizaciju za ekspresivnost u ograničenom domenu [4].

Prednosti korišćenja JSD-ova uključuju povećanu produktivnost u određenom domenu, smanjen napor za izražavanje ideja i implementaciju rešenja, kao i smanjeni broj linija programskog koda potreban za rešenje. Takođe, znatno je poboljšana čitljivost i umanjena mogućnost grešaka.

Visual Studio Code, često skraćeno sa **VS Code**, je besplatan tekstualni editor otvorenog koda (en. *open source*). Razvijen je od strane *Microsoft*-a, i namenjen je za operativne sisteme *Windows*, *Linux* i *macOS*. Ima ugrađenu podršku za programske jezike *JavaScript* i *TypeScript* i dalje omogućava podršku za druge jezike kroz bogat ekosistem ekstenzija. Kako *VS Code* koristi sistem foldera i radnog prostora (en. *workspace*) da interaguje sa projektom i sa sistemom dokumenata za manipulisanje tekstualnim fajlovima koji sadrže kod, ovo je omogućilo da se funkcionalnosti specifične za jezik dodele na osnovu ekstenzije fajla.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Igor Dejanović, red. prof.

2.1. Jezičke ekstenzije u VS Code

VS Code ne obezbeđuje ugrađenu podršku za jezike, ali nudi set API-ja koji omogućavaju bogate funkcionalnosti za jezike. Tržište ekstenzija za VS Code (en. *VS Code marketplace*), predstavlja centralizovanu platformu koja omogućava programerima da istraže, preuzmi i integrišu različite ekstenzije u svoje razvojno okruženje.

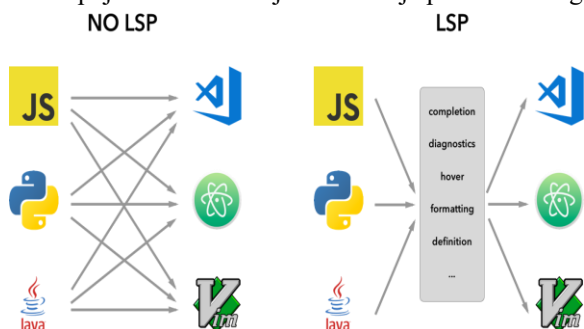
Postoje okvirno dve kategorije jezičkih funkcionalnosti;

Deklarativne jezičke funkcionalnosti – pretežno se bave definisanjem i određivanjem ponašanja ekstenzije putem struktuiranih konfiguracija. Dopušta programerima da izraze šta bi ekstenzija trebala raditi, bez potrebe da se detaljno upuštaju u implementaciju. Neki od primera ovih funkcionalnosti su: bojenje sintakse (en. *syntax highlighting*); automatsko dovršavanje koda (en. *snippet completion*); uparenost zagrada (en. *bracket matching*); automatsko zatvaranje zagrada (en. *bracket autoclosing*) itd.

Programske jezičke funkcionalnosti – omogućavaju programerima da pišu kod koji direktno interaguje sa VS Code editorom i njegovim osnovnim API-ima. Ovakve funkcionalnosti često su podpomognute Jezičkim serverom, programom koji analizira kod napisan u projektu i dinamički izvršava korisne akcije. Neki od primera programskih jezičkih funkcionalnosti su: prikaz informacija pri prelasku mišem (en. *hover information*); automatsko završavanje koda (en. *auto completion*); skok na definiciju (en. *jump to definition*); provera grešaka (en. *error checking*) itd.

3. ARHITEKTURA PYFLIES PODRŠKE

Jezičke podrške koje se oslanjaju na Protokol jezičkog servera (en. *Language server protocol, LSP*) predstavlja ključan napredak u oblasti integrisanih razvojnih okruženja i uređivača koda. To je otvoren i standardizovan protokol koji olakšava interakciju između uređivača koda i jezičkih servera, odvojenih procesa koji se bave analizom programskih jezika. LSP je, pre svega, uveden kako bi se rešio izazov pružanja bogate podrške za programske jezike u različitim uređivačima koda i IDE, budući da je održavanje jezičkih funkcionalnosti za svako pojedinačno razvojno okruženje postalo nemoguće.



Slika 1. Razlika u nekorišćenju i korišćenju LSP-a [5]

Na slici 1 se može videti ilustracija ovoga, da uvođenjem međusloja, u vidu Protokola jezičkog servera, postaje dovoljno da se podrška za određeni jezik implementira samo jednom, umesto za svaki IDE posebno.

Komunikacija u okviru LSP-a se odvija pomoću JSON-RPC i ovaj protokol definiše skup poruka koje omogućavaju efikasnu komunikaciju između jezičkog

servera i klijena. Ove poruke se svrstavaju u tri glavna tipa: poruke zahteva (en. *request messages*), obaveštenja (en. *notification messages*) i poruke odgovora (en. *response messages*).

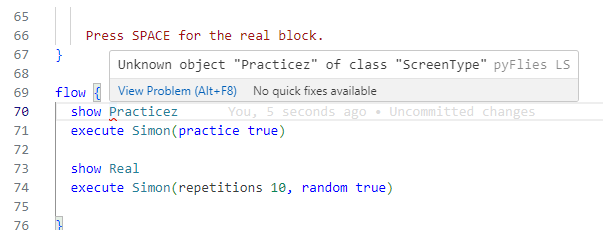
4. OPIS FUNKCIONALNOSTI EKSTENZIJE

Prvi korak ka korišćenju PyFlies podrške u VS Code-u je instalacija odgovarajuće ekstenzije. Ekstenzija je dostupna na VS Code tržištu, a korisnik može pretraživati i instalirati direktno iz radnog okruženja. Kada korisnik otvori dokument napisan u PyFlies jeziku, što će ekstenzija prepoznati kao bilo koji fajl sa nastavkom ".pf", ekstenzija će se automatski aktivirati i odmah će izvršiti određene akcije, uključujući bojenje sintakse i analizu koda.

Bojenje sintakse je jedna od primarnih funkcionalnosti PyFlies podrške, i to je prva stvar koju korisnik primeti pri otvaranju fajla napisanog u PyFlies jeziku. Ova funkcionalnost ima za cilj da postavi boju i stil na izvorni kod kako bi se olakšalo čitanje i razumevanje.

Analiza koda je takođe nešto što se izvršava pri otvaranju PyFlies fajla, i ukoliko ima grešaka, to će biti prijavljeno korisniku. Problematični delovi koda će biti podvučeni crvenom linijom, a pri pomeranju kursora miša preko označenog dela koda, korisniku će biti prikazan razlog za grešku.

Na slici 2 se može videti primer prijave greške u kodu, kao i funkcionalnost bojenja sintakse.



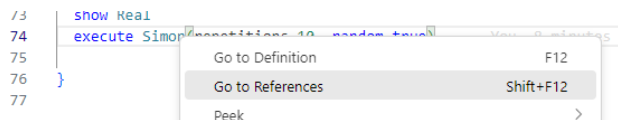
Slika 2. Primer prijave greške u izvornom kodu

Jedna od funkcionalnosti koja značajno olakšava proces pisanja koda je automatsko kompletiranje koda, što dovodi do smanjenja broja unetih karaktera, kao i grešaka pri kucanju. PyFlies podrška će prepoznati da li postoje delovi koda (en. *snippets*), koji mogu biti ubačeni na mestu gde se nalazi kursor, u zavisnosti od konteksta i unetok karaktera. Ako postoje predlozi, korisniku će biti prikazana lista opcija ispod koda koji se unosi. Korisnik može navigirati kroz listu predloga i izabrati neki od predloga. Izborom jedne od opcija, ce isečak koda će biti ubačen u izvorni kod, s tim što će se od korisnika očekivati da unese smislene vrednosti promenljivih, ukoliko izabrani isečak ima jednu ili više promenljivih u sebi.

Kada je reč o definisanim promenljivima, potrebno je pomenuti funkcionalnosti „skok na definiciju“ i „prikaz referenci“. Skok na definiciju je funkcionalnost koja omogućava korisniku da se sa bilo koje lokacije u dokumentu prebaci na lokaciju gde počinje definicija određene promenljive ili rutine. Sa druge strane, funkcionalnost prikaza referenci omogućava korisniku da vidi sve lokacije u izvornom kodu na kojima se koristi konkretna promenljiva ili rutina. Rezultat ove akcije će

biti specifičan prikaz, koji se u VS Code-u naziva prozor virenja (en. *peek window*), čiji je cilj da omogući korisniku da ostane u istom dokumentu, i na istoj lokaciji u dokumentu, pružajući uvid u sve druge lokacije gde se koristi tražena promenljiva ili rutina.

Na slici 3 se mogu videti opcije koje omogućavaju izvršavanje pomenutih funkcionalnosti.



Slika 3. Prikaz načina da se izvrše akcije skok na definiciju i prikaz referenci

Poslednja funkcionalnost koju je važno pomenuti je brza ispravka (en. *quick fix*). Ukoliko postoji greška u izvornom kodu i na početku linije koda na kojoj je prijavljena greška se pojavi ikonica sijalice, to je znak da je dostupna brza ispravka.

Klikom na ikoncu sijalice, dobija se prikaz opcija, kao što se može videti na slici 4. Trenutna verzija podrške za PyFlies nudi jednu vrstu brze ispravke, a to je ispravka greške u kucanju kada korisnik unosi referencu na neku promenljivu ili rutinu.



Slika 4. Prikaz ponude brze ispravke za grešku u kucanju

5. IMPLEMENTACIJA

U okviru ovog rada, serverska strana ekstenzije je kompletno implementirana, dok se na klijentsku stranu nadogradilo. Za implementaciju jezičkog servera korišćena je *pygls* biblioteka, kao i *lsprotocol*, paket koji predstavlja Microsoft-ovu implementaciju tipova objekata koji se koriste u okviru LSP-a.

Najbitniji dodatak klijentskoj strani ekstenzije je vezan za konfiguraciju jezičkog servera, instaliranje servera pri prvom pokretanju ekstenzije i pokretanje servera svaki sledeći put kada se aktivira ekstenzija.

Jezički server za *PyFlies* podršku je kreiran kao odvojen projekat u koji je smeštena cela logika. Taj projekat je postavljen na *PyPi* repozitorijum, i sa tog repozitorijuma je moguće instalirati *PyFlies* jezički server pokretanjem komande:

```
pip install pyflies-ls
```

Trenutna verzija *pyflies-ls* paketa, postavljena na *PyPi*, je 0.0.3, i postavljena je da zahteva *pygls* paket, verzije 1.0.2, i pakete *textx* i *pyflies*, bez konkretno određene verzije paketa.

Analiza napisanog programskog koda i prijava grešaka, kako sintakasnih, tako i semantičkih, predstavlja jednu od najvažnijih funkcionalnosti ove ekstenzije.

Takođe, predstavlja jednu od kompleksnijih sa strane implementacije.

```
@pyflies_server.feature(TEXT_DOCUMENT_DID_CHANGE)
def did_change(ls, params: DidChangeTextDocumentParams):
    """
    Text document did change notification.
    The method calls validation on the document in which the change occurred.
    """
    _validate(ls, params)

@pyflies_server.feature(TEXT_DOCUMENT_DID_OPEN)
async def did_open(ls, params: DidOpenTextDocumentParams):
    """
    Text document did open notification.
    The method calls validation on the opened document.
    """
    _validate(ls, params)
```

Listing 1. Registracija događaja otvaranja i izmene dokumenta

Kao što se može videti na listingu 1, implementira se registracija događaja otvaranja i izmene dokumenta, i kao odgovor na te događaje, poziva se *_validate* metoda.

Metoda *_validate* prvo instancira izvorni kod dokumenta, koristeći *load_document_source* metodu, koja se oslanja na mogućnosti koje nudi *pygls* biblioteka, da učita izvorni kod. Zarim se poziva metoda *validate*, gde se nalazi čitava logika vezana za analizu. Analiza se vrši pozivom metoda importovanih iz *textX-a*. Pozivom metode *metamodel_for_language* i proleđivanjem parametra *PyFlies* instancira se metamodel za jezik *PyFlies*. Za taj metamodel se poziva metoda *model_from_str*, kojoj se proleđuje izvorni kod. Cilj pozivanja ove metode jeste taj da izazovemo *PyFliesException* ili *TextXError* u slučaju da izvorni kod ima neke nepravilnosti. Ukoliko nepravilnosti postoje, kreiraju se objekti *Diagnostic* klase, koji u sebi sadrži *Range* objekat, koji će pokazati tačnu liniju u izvornom kodu koja uzrokuje nepravilnost, poruku sa opisom toga šta je uzrok nepravilnosti, i *source* vrednost, koja je uvek postavljena na *PyFlies LS*, kako bi se znalo koji je izvor dijagnostike. U suprotnom, kada je kod ispravan, metoda *validate* kao rezultat vraća praznu listu grešaka.

```
def validate(model: str) -> List[PyFliesException]:
    errors = []
    try:
        mm = metamodel_for_language("pyflies")
        mm.model_from_str(model)
    except PyFliesException as err:
        # TODO: How to determine col and line for PyFliesException
        errors.append(construct_diagnostic(err.args[0], 1, 1))
    except TextXError as err:
        msg = err.message
        col = err.col
        line = err.line

        errors.append(construct_diagnostic(msg, col, line))
    except Exception as e:
        print(e)
    return errors
```

Listing 2. Prikaz dela koda koji vrši validaciju izvornog PyFlies koda

Na listingu 2 može se videti opisani deo koda. Dakle, *PyFlies LS* će na događaj otvaranja ili promene u fajlu odgovoriti sa listom, koja će sadržati *Diagnostic* objekat ili će biti prazna. *LSP* će znati kako da interpretira

Diagnostic objekat na klijentskoj strani ekstenzije, pošto je to objekat *lsp* paketa.

Automatsko kompletiranje je implementirano dodavanjem metode *completions*, koja kao parametre prima instancu jezičkog servera i *CompletionParams* objekat. Proces izbora isečaka koda koji će biti ponuđeni se izvršava učitavanjem dokumenta, uzimanjem karaktera koji je korisnik uneo i filtriranjem mogućih isečaka tako da ostanu oni koji odgovaraju unetom karakteru. Sledeći korak je prosleđivanje učitano g dokumenta i filtriranih isečaka koda metodi koja će da instancira *PyFlies* model. Iteracijom kroz kandidatske iseče koda, radi se prover a za svaki, tako što se na odgovarajuću poziciju u dokumentu ubaci isečak i proverava se da li će instancirani model izazvati grešku, koja će biti tipa *TextXError*, ili ne. Ako ne dođe do izuzetka, znači da je isečak u tom delu koda, u tom kontekstu, validan i može biti ubačen. Klijentu će se vratiti lista *CompletionItem* objekata, ili prazna lista, ukoliko nema validnih isečaka za automatsko kompletiranje.

Funkcionalnosti **skok na definiciju i pronalazak referenci** implementirane su na sličan način. Prvi deo logike je vezan za uzimanje potpunog naziva simbola ili rutine za koju se izvršava ova operacija. To se radi tako što od pozicije kursora, koji se dobije kao podatak u zahtevu, ide ulevo i udesno, kako bi se prikupili svi karakteri koji čine naziv simbola. Kada je taj deo odrađen, u slučaju skoka na definiciju, traži se pozicija u dokumentu gde je definisana promenljiva ili rutina, pomoću instanciranja modela, što omogućava *textX*, sa prosleđenim izvornim kodom. Pristupanjem atributima instanciranog modela, može se doći do lokacije u dokumentu na kojoj je definisana promenljiva ili rutina. Taj podatak se klijentu vraća kao *Location* objekat.

Za prikaz referenci je razlika u tome što se koristi metoda *get_children*, takođe importovana iz *textX*-a, pomoću koje se formira lista *Location* objekata za traženu promenljivu ili rutinu.

Brza ispravka podrazumeva da postoji greška u kodu, tako da je prvi korak provera da li je to slučaj, i ukoliko nije, metoda završava, ne vraćajući odgovor. Ukoliko postoji greška u kodu, poziva se metoda koja će, po trenutnoj implementaciji, proveriti da li postoji greška koja ukazuje na nepostojeći, to jest nedefinisani objekat. Ovo se tretira kao greška u kucanju, u smislu da je namera bila da se unese referenca na neku od definisanih promenljivih ili rutina. Sa time na umu, iterira se kroz definisane nazive za odgovarajući tip objekta, i ukoliko postoji naziv definisanog objekta koji je dovoljno sličan onome što je korisnik uneo, a što je izazvalo grešku, onda se kreira *CompletionItem* objekat koji će zameniti uneti naziv promenljive sa ispravnim nazivom. Ukoliko ne postoji dovoljno dobar predlog za brzu ispravku, server neće vratiti ništa kao opciju.

6. ZAKLJUČAK

Tema ovog rada bila je opis podrške za *PyFlies* jezik u *VS Code* editoru, i u okviru istog pruženo je duboko

razumevanje ključnih aspekata implementacije ove podrške. Opisan je *PyFlies* jezik, *VS Code* radno okruženje, i arhitektura podrške koja koristi Protokol jezičkog servera. Sa osloncem na opisane pojmove, predstavljen je niz funkcionalnosti koje *PyFlies* podrška nudi, i opisana implementacija tih funkcionalnosti.

Održavanje *VS Code* ekstenzije i jezičkog servera je u direktnoj zavisnosti od izmena na *PyFlies* jeziku. Ukoliko u budućnosti dođe do promena u gramatici jezika, u smislu da se dodaju nova pravila ili da se izmene postojeća, potrebno je prilagoditi *snippets\PyFlies-snippets.json* fajl u jezičkom serveru, kako bi automatsko kompletiranje bilo u skladu sa najnovijom verzijom *PyFlies* jezika.

Takođe, kod održavanja treba imati na umu verzije paketa koji su potrebni kako bi jezički server mogao da funkcioniše. Tu konkretno spadaju *PyFlies* paket, podrazumevano, kao i *textX* i *pygls*.

Nije retka pojava da se u novijim verzijama paketa neki modul ili metoda preimenuje, promeni ili potpuno obriše.

Problem sa verzijama paketa od kojih jezički server zavisi se može rešiti navođenjem tačnih verzija koje će biti iskorišćenje pri instalaciji. Međutim, novije verzije pomenutih paketa mogu doneti benefite u vidu novih funkcionalnosti ili boljih performansi, pa to treba razmotriti, kada je u pitanju dalji razvoj jezičkog servera.

7. LITERATURA

- [1] Dejanović, I.; Dejanović, M.; Vidaković, J.; Nikolić, S. *PyFlies: A Domain-Specific Language for Designing Experiments in Psychology*. Appl. Sci. 2021, 11, 7823. <https://doi.org/10.3390/app11177823>
- [2] Peirce, J., Gray, J.R., Simpson, S. et al. *PsychoPy2: Experiments in behavior made easy*. Behav Res 51, 195–203 (2019). <https://doi.org/10.3758/s13428-018-01193-y>
- [3] <http://textx.github.io/textX/stable/>, pregledano 16. oktobar 2019.
- [4] Mernik, M., Heering, J., & Sloane, A. M. (2005). When and how to develop domain-specific languages. ACM Computing Surveys, 37(4), 316–344. doi:10.1145/1118890.1118892
- [5] Microsoft. 2018. Example - Language Server. (2018). <https://code.visualstudio.com/api/language-extensions/language-server-extension-guide>

Kratka biografija:

Dejan Šorgić rođen je u Beogradu 1996. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Softversko inženjerstvo i informacione tehnologije odbranio je 2023.god. kontakt: dejans1224@gmail.com

UTICAJ ERP SISTEMA I INTEGRACIONIH ALATA U INDUSTRIJI 4.0**THE IMPACT OF ERP SYSTEMS AND INTEGRATION TOOLS IN INDUSTRY 4.0**Bojana Nešković, *Fakultet tehničkih nauka, Novi Sad***Oblast – INŽENJERSTVO INFORMACIONIH SISTEMA**

Kratak sadržaj – Rad predstavlja analizu veza između integracije ERP sistema, Industrije 4.0 i praktične primene u IT industriji. Kroz sistematski pregled literature, analizirane su metodologije i tehnike implementacije, pružajući dublji uvid u teorijski okvir. Zatim je sproveden upitnik među IT stručnjacima, pružajući stvarne uvide u izazove i prednosti koje su iskusili u radu sa integracionim alatima. Na kraju, razvijena je aplikacija za praćenje zaliha i radnih naloga, integrišući principe Industrije 4.0, kako bi se poboljšali operativni procesi.

Ključne reči: ERP sistemi, integracioni alati, Industrija 4.0

Abstract: The paper presents an analysis of the links between ERP system integration, Industry 4.0, and practical application in the IT industry. Through a systematic literature review, implementation methodologies and techniques were analyzed, providing a deeper insight into the theoretical framework. A questionnaire was then conducted among IT professionals, providing real-world insights into the challenges and benefits they experienced working with integration tools. Finally, an application was developed to track inventory and work orders, integrating Industry 4.0 principles to improve operational processes.

Keywords: ERP systems, integration tools, Industry 4.0

1. UVOD

U današnjem dinamičnom i tehnološki naprednom poslovnom okruženju, integracija informacionih tehnologija ima ključnu ulogu u unapređenju efikasnosti, produktivnosti i konkurentne prednosti organizacija. Integracija sistema za planiranje resursa preduzeća (engl. *Enterprise Resource Planning* – ERP) sa konceptom Industrije 4.0 predstavlja važan korak ka postizanju digitalne transformacije u industrijskim procesima.

Industrija 4.0 je pametan sistem, koji se koristi kao fleksibilna proizvodna linija za realizaciju proizvodnih procesa u realnom vremenu, podržanih veštačkom inteligencijom, pri donošenju pametnih odluka, povezanih internetom stvari i sajber fizičkim sistemima (mašinama) [1]. ERP sistem je alat za integraciju svih podataka i menadžerskih veština, koje su u području delatnosti preduzeća, u jedinstvenu bazu podataka, od finansija, ljudskih resur-

sa, kroz elemente lanca snabdevanja i povezivanja proizvodnje sa dobavljačima, sa jedne strane i potrošačima sa druge strane. ERP sistem se sastoji od aplikativnih modula, koji međusobno komuniciraju standardnim protokolima, zahvaljujući jedinstvenoj bazi i obradi podataka [2]. Ovaj rad je podeljen u tri celine. Prva celina obuhvata sistematski pregled literature na zadatu temu, istražujući relevantne izvore koji se odnose na integraciju ERP sistema i Industrije 4.0. Biće postavljena istraživačka pitanja, kao i prikazane baze podataka i termini pretrage korišćeni za pronalaženje primarnih studija. Drugi deo je sprovođenje upitnika među zaposlenima u IT industriji. Fokus je na njihovim uvidima i iskustvima u vezi sa primenom integracionih alata u okviru Industrije 4.0. Treći deo se odnosi na prikaz integrisanog softverskog rešenja za neki vid planiranje proizvodnje, u ovom slučaju aplikacije za praćenje stanja zaliha i radnih naloga.

2. TEORIJSKE OSNOVE

U okviru ovog poglavlja su opisani ključni koncepti i pojmovi koji se koriste u radu, a koji su povezani sa integracijom u kontekstu Industrije 4.0.

ERP sistemi su se prvi put pojavili kasnih 1980-ih i početkom 1990-ih sa snagom interfunkcionalne koordinacije i integracije u celom preduzeću. Oni integrišu poslovne procese uključujući proizvodnju, distribuciju, računovodstvo, finansije, upravljanje ljudskim resursima, upravljanje projektima, upravljanje zalihama, servis i održavanje i transport, obezbeđujući pristupačnost, vidljivost i doslednost širom sveta. Tokom 1990-ih ERP dobavljači su dodali više modula i funkcija kao “dodatke” osnovnim modulima što je dovelo do “proširenih ERP-ova”. Ova ERP proširenja uključuju napredno planiranje i zakazivanje, e-poslovna rešenja, kao što su upravljanje odnosima sa klijentima i upravljanje lancem snabdevanja [3]. Preduzeće uvođenjem ERP sistema, može da zameni dve ili više nezavisnih aplikacija i da eliminiše potrebu za spoljnim interfesima, koji su ranije bili potrebni za spajanje ovih sistema. Ovo pruža dodatne pogodnosti, koje se kreću od standardizacije i smanjenih troškova održavanja (jedan sistem umesto dva), do lakših i/ili većih mogućnosti izveštavanja (pošto se svi podaci obično nalaze u jednoj bazi podataka) [4]. Integracioni alati predstavljaju ključnu komponentu u savremenom poslovanju, omogućavajući organizacijama da povežu različite sisteme, aplikacije i tehnologije. Ovi alati funkcionišu kao posrednici, olakšavajući komunikaciju između različitih delova sistema i omogućavajući im da rade sinhronizovano i efikasno. U poslovnom svetu gde se podaci često nalaze u različitim formatima i na različitim platformama, integracioni alati igraju ključnu ulogu u osiguravanju da se ovi podaci mogu deliti i koristiti na koordiniran način. Kroz njihovu

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bila dr Teodora Vučković, docent.

upotrebu, organizacije mogu automatizovati procese, poboljšati operativnu efikasnost i brže reagovati na promene u poslovnom okruženju. Integracioni alati nude različite funkcionalnosti, uključujući transformaciju podataka, usklađivanje različitih sistema, bezbedno deljenje podataka, praćenje performansi sistema i još mnogo toga. Oni omogućavaju organizacijama da se prilagode brzim tehnološkim promenama i ostanu konkurentske tako što omogućavaju fleksibilnost, skalabilnost i optimizaciju resursa.

U današnjem digitalnom svetu, nekoliko poznatih integracionih rešenja se ističe. MuleSoft je jedan od lidera na tržištu i omogućava lako povezivanje raznolikih sistema putem podrške za različite formate podataka.

Industrija 4.0 je revidirani pristup proizvodnji koji koristi najnovije tehnološke pronalaskе i inovacije, posebno u spajanju operativne i informaciono-komunikacione tehnologije. Industrija 4.0 primenjuje alate koji obezbeđuju napredak u operativnoj, komunikacijskoj i informacionoj tehnologiji za povećanje nivoa automatizacije i digitalizacije proizvodnje, u proizvodnim i industrijskim procesima. Cilj je upravljanje čitavim procesom lanca vrednosti, poboljšanjem efikasnosti u proizvodnom procesu i stvaranjem proizvoda i usluga vrhunskog kvaliteta. Ova vizija sledi maksimu višeg kvaliteta, a ne na račun niže cene. Ova filozofija je proizvela pametnu fabriku, gde su efikasnost i troškovi poboljšani, a profit povećan [5].

Devet je tehnoloških napretka koji su stvorili četvrtu revoluciju: autonomni roboti, simulacija, horizontalna i vertikalna integracija sistema, industrijski internet stvari, sajber bezbednost, oblak, aditivna proizvodnja, proširena stvarnost i veliki podaci i analitika [6]. Industrija 4.0 takođe menja način na koji radimo. Radnici postaju suštinski povezani sa tehnologijom i informacijama, omogućavajući im da rade sa kompleksnim sistemima i reaguju na vreme na promene u proizvodnji. Ova nova era proizvodnje ne samo da povećava efikasnost i produktivnost, već i omogućava kreativniju i inovativniju radnu snagu.

Četvrta industrijska revolucija zasnovana je na konceptu pametne fabrike. Pametne fabrike imaju potpuno novi pristup proizvodnji. Pametni proizvodi se uvek mogu identifikovati i locirati. Danas su se mnoge kompanije udaljile od masovne proizvodnje do masovne proizvodnje po meri. Glavni cilj je imati proizvodni sistem koji može da se odupre svim dinamičnim poslovnim procesima. Takav sistem mora da se odlikuje fleksibilnošću, tako da može da odgovori na poremećaje različitog porekla [7].

3. METODOLOGIJA RADA

Rad je započet sistematskim pregledom literature, što predstavlja detaljan proces analize relevantnih naučnih članaka, knjiga i izveštaja u vezi sa Industrijom 4.0 i ERP sistemima. Proučavanje literature omogućilo nam je da, ne samo razumemo osnovne koncepte i definicije vezane za Industriju 4.0, ERP sisteme i integracione alate, već i da analiziramo različite pristupe koje istraživači i stručnjaci koriste širom sveta. Videli smo primenu različitih ideja, što je omogućilo da obogatimo naše istraživanje različitim perspektivama i metodologijama koje su primenjene u različitim kontekstima. Sistematski pregled literature nam je takođe pomogao da identifikujemo praznine u postojećim istraživanjima.

Nakon što smo pažljivo analizirali literaturu vezanu za

Industriju 4.0, prešli smo na sledeći korak: razvoj odabranog mernog instrumenta, odnosno strukturiranog upitnika kojim je sprovedeno istraživanje u praksi, a koje se odnosi na integracione alate. Upitnik je bio glavni alat koji se koristio u istraživanju. Prvo smo identifikovali ključne teme i pitanja koja smo želeli da istražimo.

Upitnik je zatim distribuiran među ciljanom populacijom IT stručnjaka. Podaci prikupljeni putem ovog istraživanja bili su osnova za dalju analizu. Ovaj proces nam je omogućio da prikupimo kvantitativne i kvalitativne podatke, pružajući celovitu sliku stanja u industriji. Kao finalni korak, analizirani rezultati su pažljivo sintetisani kako bi se izvukli relevantni zaključci o stanju integracionih alata u Industrij 4.0.

Nakon završetka istraživačke faze i prikupljanja relevantnih podataka, sledeći korak je praktična implementacija. Pristupljeno je razvoju integrisanog informacionog sistema koji je fokusiran na praćenje stanja zaliha i radnih naloga.

4. SISTEMATSKI PREGLED LITERATURE

U nastavku ovog poglavlja je predstavljen sistematski pregled literature, uključujući i faze, na temu integracije ERP sistema i Industrije 4.0.

4.1. Planiranje pregleda literature

Prateći preporuke iz uputstva za sprovođenje sistematskog pregleda literature Barbare Kitchenham [8], potrebno je da definišemo cilj istraživanja. Osnovni cilj jeste da se kroz sistematski pregled literature otkriju trendovi i implikacije integracije ERP sistema i Industrije 4.0, odnosno na koji način se ERP sistemi mogu integrisati, koji si izazovi i opasnosti sa kojima se susrećemo i slično. Stoga su postavljena sledeća istraživačka pitanja (IP):

IP1. Kako se integracija ERP sistema i koncept Industrije 4.0 odražava na transformaciju tradicionalnih poslovnih modela?

IP2. Kako integracija tehnologija unutar okvira Industrije 4.0 može unaprediti efikasnost proizvodnje i optimizaciju procesa podržanih ERP sistemima?

Kako bismo pronašli relevantne radove u izvorima, bilo je potrebno definisati ključne pojmove koji se koriste u pretrazi. Generisan je upit koji predstavlja kombinaciju ključnih reči i dodatnih kriterijuma pretrage:

["Industry 4.0" and "ERP integration"] or ["digital transformation" and "ERP systems"] and pubyear > 2021

Naredni korak u planiranju pregleda literature podrazumeva definisanje kriterijuma inkluzije i ekskluzije. Identifikovani kriterijumi inkluzije (IK) su sledeći:

IK 1: Rad se mora odnositi na vezu ERP sistema i Industrije 4.0.

IK 2: Rad mora biti napisan na engleskom jeziku.

Identifikovani kriterijumi za isključivanje (EK) rada su sledeći:

EK 1: Potrebno je ukloniti duplikate.

EK 2: Potrebno je zanemariti radove objavljene pre 2022. godine.

4.2. Izvođenje pregleda literature

Inicijalna pretraga u sve tri odabrane indeksne baze (Scopus, ACM Digital Library i Science Direct) rezultirala je sa ukupno 197 radova koji odgovaraju prethodno

definisanom upitu. Najmanji broj radova pronađen je u Scopus bazi (16 radova ili 8.1% ukupnog broja), zatim u ACM Digital Library bazi (22 rada ili 11.2%), dok je najveći broj rezultata dobijen u bazi Science Direct (159 radova ili 80.7%). Nakon uklanjanja radova koji su prisutni u više od jedne indeksne baze, broj radova je smanjen.

Ukupno je izabrano 18 radova na kojima će se bazirati deskriptivna statistika i koji će nam omogućiti da odgovorimo na postavljena istraživačka pitanja.

4.3. Prikaz rezultata sistematskog pregleda literature

U ovom delu je odgovoreno na postavljena istraživačka pitanja definisana u prethodnoj sekciji.

Integracija ERP sistema i koncept Industrije 4.0 predstavlja ključnu tačku u transformaciji tradicionalnih poslovnih modela. Ova transformacija ima značajan uticaj na to kako organizacije posluju, upravljaju resursima i donose odluke. Jedan od ključnih nalaza iz proučenih radova je da implementacija ERP sistema i koncepta Industrije 4.0 može doneti značajne koristi, kako finansijske, tako i nefinansijske. ERP sistemi pružaju bolji uvid u poslovne procese, omogućavaju optimizaciju resursa i poboljšavaju donošenje odluka. Uvođenje tehnologija Industrije 4.0, kao što su IoT, pametna proizvodnja i sajber-fizički sistemi, dodatno podržava automatizaciju, digitalizaciju i povezanost u celokupnom lancu vrednosti. Ovo dovodi do smanjenja troškova, povećanja efikasnosti, boljeg upravljanja rizicima i stvaranja novih poslovnih modela.

Kroz sveobuhvatnu analizu prvih osam radova može se zaključiti da integracija ERP sistema i koncepta Industrije 4.0 može značajno poboljšati performanse organizacija. Ključni faktori uspeha u ovom procesu uključuju kvalitetan ERP sistem, adekvatnu IT infrastrukturu, pravilno upravljanje podacima i promišljeno uvođenje tehnologija. Ovi koraci omogućavaju transformaciju poslovnog modela ka agilnijem, fleksibilnijem i efikasnijem okruženju, sposobnom da odgovori na zahteve brzih promena u današnjem poslovnom svetu.

Naredna grupa radova odgovara na drugo istraživačko pitanje i istražuje različite aspekte tehnološkog napretka u oblasti industrije i upravljanja poslovnim procesima, sa fokusom na primenu tehnologija poput Industrije 4.0, pametnih fabrika, digitalnih replika, upravljanja lancem snabdevanja i sistema za planiranje i kontrolu proizvodnje. Svaki rad se bavi specifičnim temama i analizira kako tehnologije unutar Industrije 4.0 mogu unaprediti različite aspekte poslovanja. Svaki od radova doprinosi razumevanju kako tehnološki napredak može transformisati različite aspekte industrije i poslovnog upravljanja, donoseći nove mogućnosti za inovacije, efikasnost i održivost.

5. UTICAJ INTEGRACIONIH ALATA – ISTRAŽIVANJE

5.1. Definisanje ciljeva istraživanja

Ovo istraživanje ima za cilj pružiti dublji uvid u svet integracija u kontekstu Industrije 4.0, što će doprineti unapređenju poslovnih praksi u modernoj industriji. Fokusira se na efikasnost integracionih alata, identifikuje

izazove IT stručnjaka, procenjuje DevOps pristup, istražuje nivo obuke i podrške, i proučava integraciju ERP sistema. Ciljevi istraživanja su razumevanje brzine, olakšanja i doprinosa integracija, identifikacija ključnih izazova, analiza uticaja DevOps pristupa, evaluacija obuke i podrške, kao i proučavanje integrisanih ERP sistema.

5.2. Definisanje upitnika i njegova distribucija

Upitnik je odabran kao glavni instrument istraživanja iz nekoliko ključnih razloga. Upitnici omogućavaju sistematizovano prikupljanje podataka od velikog broja ispitanika u relativno kratkom vremenskom periodu. Ova efikasnost je ključna, posebno u istraživanjima koja pokrivaju širok spektar tema, kao što je slučaj sa kompleksnom temom Industrije 4.0 i integracija.

Upitnik kreiran za ovo istraživanje sadrži 20 pitanja, koja su raspoređena po oblastima.

Prvi deo pitanja se odnosi na osnovne informacije o ispitanicima, uključujući pol, radno iskustvo, starosnu grupu i trenutnu poziciju. Nakon toga, sledi grupa pitanja vezanih za korišćenje integracionih alata, odnosno na njihovu konkretnu upotrebu. Sledeće grupa pitanja se odnosi na primenu DevOps pristupa u integracionim projektima i evaluaciju nivoa obuke i podrške koje kompanije pružaju. I poslednja grupa pitanja se odnosi na korišćenje ERP sistema, ako su prisutni. Ova pitanja omogućavaju razumevanje veze između ERP sistema i integracija.

Upitnik je prosleđen jednoj IT kompaniji specijalizovanoj za integracije. Ova kompanija je odabrana zbog njihove stručnosti u domenu integracija. Ispitanici u ovoj kompaniji imaju praktično iskustvo sa primenom integracionih alata i DevOps pristupa u realnim projektima, čineći ih relevantnim izvorom podataka za istraživanje.

5.3. Analiza i interpretacija rezultata

Kao što je već pomenuto, glavni cilj ovog istraživanja jeste bolje razumevanje percepcije IT stručnjaka o integracijama u okviru Industrije 4.0. Glavni i jedini ispitanici su zaposleni u IT kompaniji.

Konačan set podataka nakon izvršenog procesa anketiranja sadrži odgovore od ukupno 30 ispitanika.

Na osnovu analize rezultata upitnika o integracionim alatima u Industriji 4.0, ključni zaključci su sledeći: firma ne koristi ERP sisteme, što ukazuje na potrebu za edukacijom i promocijom njihove upotrebe. MuleSoft i Tibco dominiraju kao najčešće korišćeni alati, naglašavajući važnost razvoja veština upravljanja ovim alatima. Ispitanici su visoko ocenili efikasnost integracionih alata, ukazujući na pozitivno iskustvo u njihovoj primeni. Glavni izazovi uključuju složenost implementacije i visoke troškove, što ukazuje na potrebu za efikasnijim rešenjima.

Ovi zaključci ukazuju na potrebu za intenzivnijim usmeravanjem obrazovanja i resursa ka razumevanju i implementaciji ERP sistema, kao i na važnost podrške u prevazilaženju izazova koji proizlaze iz složenosti i visokih troškova integracionih rešenja.

Takođe, ističe se važnost kontinuirane edukacije i osnaživanja IT stručnjaka u upravljanju i optimizaciji integracija u kontekstu Industrije 4.0.

6. RAZVOJ INTEGRISANOG INFORMACIONOG SISTEMA ZA PRAĆENJE ZALIH I RADNIH NALOGA

Cilj ovog projekta jeste razvoj integrisanog informacionog sistema koji će da obezbedi praćenje i upravljanje zalihama i radnim nalogima u jednoj kompaniji. Sistem za praćenje i upravljanje zalihama i radnim nalogima je ključni deo organizacije koja se bavi proizvodnjom, prodajom i distribucijom proizvoda. Dakle, kompanija kada primi nove zalihe materijala ili proizvoda u skladište, potrebno je da ih evidentira, odnosno da ažurira njihovo stanje, ukoliko se već nalaze u evidenciji. Nakon toga, ukoliko proces proizvodnje zahteva izvesnu količinu materijala ili proizvoda, neophodno je isporučiti ih, uz prethodno evidentiranje promena i kretanje radnog naloga.

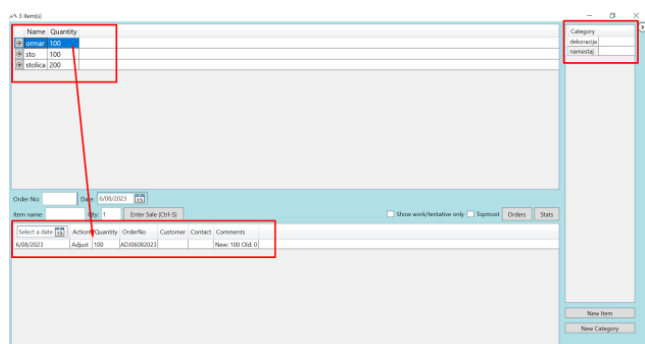
Takođe, prilikom prodaje nekih proizvoda iz skladišta, kreira se nalog i broj proizvoda na zalihama se smanjuje. U svakom trenutku stanje zaliha treba da bude ažurno i da prezentuje stvarno stanje na skladištu, uz knjiženje svih naloga koji su izazvali promene stanja.

Mnogo organizacija koristi manuelne procese i nepovezane sisteme za praćenje proizvoda, naručivanje i upravljanje radnim nalogima. To dovodi do neefikasnih procesa, grešaka u zalihama, kasnijih isporuka i problema sa praćenjem proizvoda u realnom vremenu.

Uvođenjem ove aplikacije, organizacija bi ostvarila značajne benefite u smislu vremenske i resursne efikasnosti, poboljšane tačnosti i pouzdanosti podataka, bolje upravljanje proizvodima i povećane produktivnosti. Sistem za praćenje i upravljanje zalihama i radnim nalogima omogućava organizaciji da se fokusira na svoje ključne poslovne aktivnosti, osigurava zadovoljstvo klijenata i stvara stabilnu osnovu za dalji razvoj i rast poslovanja.

Tehnologije koje su korišćene za izradu aplikacije za praćenje i upravljanje zalihama i radnim nalogima su: C#, .NET tehnologija za desktop aplikacije (engl. *Windows Presentation Foundation – WPF*) i MS SQL Server Management Studio.

Na slici 1.1 dat je prikaz aplikacije nakon pokretanja, odnosno nakon učitavanja glavnog prozora.



Slika 1.1. Glavni prozor aplikacije

7. ZAKLJUČAK

Kroz ovaj rad, ušli smo u svet digitalne industrije, istražujući kako se tehnologija može uskladiti sa poslovnim procesima. Analiza literature, zajedno sa prikazanim softverskim rešenjem, naglašava važnost ove integracije za unapređenje poslovnih performansi i stvaranje održivih konkurentskih prednosti.

Ova studija potvrđuje da integracija ERP sistema i Industrije 4.0 nije samo tehnološki trend, već nužnost za organizacije koje žele ostati konkurentne. Kroz pažljivo planiranje, implementaciju i stalno unapređenje ovih sistema, organizacije mogu stvoriti agilna, efikasna i inovativna poslovna okruženja. Ove karakteristike postaju ključne u savremenom poslovanju, omogućavajući organizacijama da ne samo prežive, već i prosperiraju u vremenu digitalne revolucije.

Dalji pravci razvoja u integraciji ERP sistema i Industrije 4.0 usmereni su ka unapređenju analitičkih alata kao što su veštačka inteligencija i mašinsko učenje za brže i preciznije donošenje odluka. Implementacija uređaja vezanih za internet stvari treba da se fokusira na smanjenje troškova i povećanje efikasnosti praćenja proizvodnih procesa. Edukacija zaposlenih i fokus na poboljšanju korisničkog iskustva igraju bitnu ulogu u budućnosti ove integracije.

8. LITERATURA

- [1] Majstorović, N. V., Majstorović, V. D., Mitrović, R., & Mišković, Ž. (2021). Industrija 4.0 i njena primena u medicini i stomatologiji, kao i borbi protiv pandemije COVID-19. *Tehnika*, 76(4), 509-520.
- [2] Tešić, Z., Milić, B., & Mitrović, V. ERP SISTEMI U INTELIGENTNOM PRIVREDIVANJU ERP SYSTEMS IN INTELLIGENT ENTERPRISING.
- [3] Rashid, M. A., Hossain, L., & Patrick, J. D. (2002). The evolution of ERP systems: A historical perspective. In *Enterprise resource planning: Solutions and management* (pp. 35-50). IGI global.
- [4] Jovićić, B., & Vlajić, S. (2007). Evolucija ERP sistema. *ИнфоМ-Часопис за информационе технологије и мултимедијалне системе*, 2007(22).
- [5] Gilchrist, A., & Gilchrist, A. (2016). Introducing Industry 4.0. *Industry 4.0: The industrial internet of things*, 195-215.
- [6] Tsvetkova, R. (2017). What does Industry 4.0 mean for sustainable development?. *Industry 4.0*, 2(6), 294-297.
- [7] Crnjac, M., Veža, I., & Banduka, N. (2017). From concept to the introduction of industry 4.0. *International Journal of Industrial Engineering and Management*, 8(1), 21.
- [8] Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, 33(2004), 1-26.

Kratka biografija:



Bojana Nešković rođena je u Ljuboviji 1998. godine. Diplomski rad pod nazivom „Web aplikacija za podršku upravljanju bibliotekom“ odbranila je 27. septembra 2021. godine na Fakultetu tehničkih nauka sa ocenom 10.

SISTEM ZA PODRŠKU RADA JAVNE UPRAVE U PROCESU IZDAVANJA PASOŠA I LIČNE KARTE**SUPPORT SYSTEM FOR THE OPERATION OF PUBLIC ADMINISTRATION IN THE PROCESS OF ISSUING PASSPORTS AND ID CARDS**

Petar Maksimović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – INŽENJERSTVO INFORMACIONIH SISTEMA

Kratak sadržaj – U ovoj studiji pružen je pregled procesa razvoja veb aplikacija namenjenih podršci elektronske uprave u procesu izdavanja dokumenata. Analizirane su prednosti i nedostaci tehnologija koje su odabrane za implementaciju, kao i samog izabranog rešenja. Pored toga, pružen je predlog mogućih poboljšanja u postojećem rešenju s ciljem optimizacije njegove efikasnosti i funkcionalnosti.

Ključne reči: *Informacione tehnologije, eUprava, veb aplikacija, baza podataka.*

Abstract – This study provides an overview of the development process of web applications designed to support electronic governance in the document issuance process. The advantages and disadvantages of the selected technologies for implementation, as well as the chosen solution itself have been analyzed. Additionally, suggestions for potential improvements to the existing solution are offered with the aim of optimizing its efficiency and functionality.

Keywords: *Information technologies, E-government, web application, database.*

1. UVOD

Napredak tehnologije je ključan faktor u širenju upotrebe interneta, koji je danas centralni aspekt komunikacije. Društvene mreže i onlajn zakazivanje su postali osnovni kanali komunikacije, ne samo u svakodnevnom životu već i u obrazovanju, poslovanju i raznim sektorima industrije. Ova promena dovodi do napretka onlajn prodaje, rastuće potrebe za zakazivanjem sastanaka i povećanje potražnje za veb developerima.

Internet sada više nego ikada, oblikuje način na koji komuniciramo, kupujemo i posluje, čineći veb developere (engl. *Web Developer*) ključnim igračem u podršci i unapređenju ove digitalne revolucije. Onlajn zakazivanje preko interneta danas postaju sve češća, evoluiraju iz dana u dan, bez obzira da li je upitanju zakazivanje u određenom servisu za automobile, zakazivanje pregleda kod doktora ili termina za izradu pasoša, lične karte, vozačkih dozvola ili registracije vozila.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Teodora Vučković, docent.

Ovakav način prijavljivanja, odnosno zakazivanja putem interneta nije bio ustaljen u nekim manjim zemljama uključujući i Srbiju. U današnje vreme okolnosti se menjaju, i broj onlajn zakazivanja klijenata se povećava. U drugim zemljama u svetu ovakav način funkcionisanja je prisutan duži vremenski period, gde predstavlja jednostavan i popularan način dolaska do termina. Korišćenje interneta kako među mlađim uzrastom, tako i među starijim osobama, u današnje vreme veoma brzo raste, a shodno tome raste i veća zainteresovanost za svetom onlajn kupovine ili zakazivanja.

2. TEORIJSKE OSNOVE

Internet je kompleksna mreža koja obuhvata servere, rutere, telekomunikacionu opremu, računare i druge ključne komponente. Da bi internet efikasno funkcionisao, neophodni su različiti tipovi softvera, a ova infrastruktura je presudna i za elektronsko poslovanje. U svetu informatičke revolucije, koja prati ove tehnološke procese, svako preduzeće se suočava s izazovom postizanja svetskog nivoa konkurencije, čak i kada svoje proizvode ili usluge prodaje samo na lokalnom ili regionalnom tržištu, kako bi opstalo na zahtevnom tržištu. Elektronsko poslovanje, predstavlja sve procese koje organizacija sprovodi putem računarske mreže, uključujući i internu i eksternu komunikaciju. Ova vrsta poslovanja zasniva se na primeni raznovrsnih informacionih i komunikacionih tehnologija [1].

Elektronski sistem uprave omogućava obavljanje administrativnih poslova iz kuće, kancelarije ili sa neke druge lokacije, čime se štedi na vremenu i ubrzava proces obavljanja zadataka. Tradicionalni način, poput čekanja u redu, postaju zastareli u poređenju s efikasnošću elektronskih rešenja. Glavni cilj elektronske uprave je transformacija odnosa između vlade i građana, stvarajući novu, dinamičnu paradigmu koja omogućava lakši pristup elektronskim uslugama za građane [1].

Pogled na e-vladu zavisi od interesa osobe koja je opisuje. Za poslovne ljude, to je brza elektronska registracija preduzeća, elektronsko vođenje katastra ili obavljanje javnih nabavki onlajn. Građane posebno zanima dostupnost različitih informacija koje pruža eUprava, kao što su upisi u škole i na fakultete, provera stanja na računima, plaćanje putem interneta i izborni sistem bez prevare. Novinarima pristup ovim informacijama je brz i jednostavan. Važno je istaći da eUprava nije samo internet stranica javne uprave. Sledeća definicija objašnjava vezu između e-poslovanja i javne uprave [2].

E-uprava predstavlja formu elektronskog poslovanja vlade i odnosi se na isporuku elektronskih servisa različitim grupama u javnosti, kao i na poslovnu saradnju i transakcije koje vlada obavlja sa različitim organizacionim entitetima, kao što su druge vladine službe, partneri ili agencije. Suština novog načina funkcionisanja javne uprave je primena elektronskog poslovanja [2].

3. OPIS REALNOG SISTEMA

3.1. Elektronska uprava – Izdavanje dokumenata

Prvi korak u stvaranju dokumenta je prepoznavanje potrebe za njegovim formiranjem, bez obzira da li se radi o prvom kreiranju ili produženju važenja već postojećeg dokumenta. Elektronska uprava pruža mogućnost da se ovaj proces započne i delimično obavi putem interneta, tačnije preko veb stranice elektronske uprave. Na ovaj način korisnik može podneti zahtev za izradu određenog dokumenta putem elektronskog interfejsa.

Koncept stvaranja informacionog sistema za automatizaciju procesa izdavanja pasoša i ličnih karti podrazumeva mogućnost registrovanja korisnika radi kreiranja novih dokumenata ili modifikacije već postojećih. Inicijalni korak u ovom procesu obuhvata kreiranje i unošenje korisničkih podataka u centralnu bazu.

Nakon uspešnog registrovanog korisnika, sistem bi omogućavao generisanje odgovarajućih dokumenata i njihovo povezivanje sa specifičnim korisničkim profilom.

3.2. Definisanje funkcionalnosti

Identifikacija funkcionalnosti je rezultat pažljive analize okoline u oblasti koju želimo automatizovati. Da bismo širok spektar potrebnih funkcionalnosti koje naš softver treba da obuhvati, neophodno je posmatrati tu oblast kroz iščitavanje potencijalnih korisnika softvera.

Neophodno je precizirati ko će biti korisnik softvera i na koji način će koristiti aplikaciju. U kontekstu ove aplikacije, prvenstveno je važno omogućiti kreiranje korisničkih naloga i njihovo unošenje u bazu podataka, s obzirom da dokument bez jasno definisanih vlasnika gube smisao.

Takođe, od suštinskog značaja funkcionalnost obuhvata mogućnost ažuriranja i brisanje podataka određenog korisnika ukoliko je to potrebno. Sledeća korisna opcija podrazumeva izlistavanje svih korisnika iz baze podataka i njihovo pregledavanje.

3.3. Dizajn aplikacije

Dizajn zapravo predstavlja kreativno rešenje određenog problema. Ključno je razumeti potrebe i želje korisnika kako bismo oblikovali efikasno rešenje za njih. Vizuelni dizajn fokusira se na elemente poput logo, rasporeda, fontova, boja i drugo. Pri razvijanju naše aplikacije za elektronsku upravu, usmeravali smo se ka postizanju jednostavnog izgleda s ciljem omogućavanja korisnicima da na najlakši i najintuitivniji način zadovolje definisane potrebe.

4. POSTUPAK IMPLEMENTACIJE SISTEMA

Kako bi se predstavio uvid u sam zadatak, u ovom delu rada predstavljen je prikaz korisničkog interfejsa.

4.1. Kreiranje baze podataka

Baza se kreira pomoću pristupa *Database First*, koji pruža alternativu pristupima *Code First* i *Model First* za entitetski model podataka. Iz baze podataka u projektu se generišu kodovi modela klase, svojstva (engl. *property*), *DbContext*, koji postaju veza između baze podataka i kontrolera (engl. *Controller*). Za potrebe našeg projekta, kreiranje baze podataka izvršeno je kroz *Microsoft SQL Server Management Studio* [3].

4.2. Web aplikacija

Veb aplikacije su softverski programi smešteni i izvršavani na izdvojenom serveru, pristupačni putem interneta pomoću veb pretraživača. Osim veb aplikacija, postoje i veb usluge i veb sajtovi, ali važno je napomenuti da mnogi od njih mogu obavljati specifične funkcije i, stoga, se takođe smatraju veb aplikacijama. Iako neke veb aplikacije mogu zahtevati određeni veb pretraživač, većina njih je univerzalno pristupačna preko različitih pretraživača.

Veb aplikacija koja simulira informacioni sistem za podršku rada javnoj upravi u procesu izdavanja pasoša i lične karte.

Veb aplikacija omogućava:

- logovanje korisnika i administratora (zaposlen),
- evidencija podataka o građanima,
- podaci o ličnim dokumentima,
- realizacija troškovi uplate,
- podnošenje zahteve za izdavanje dokumenata,
- izdavanje ličnih dokumenata
- evidencija produžetaka ličnih dokumenata,
- upravljenje izgubljenih dokumenata.

Veb aplikacija koja nam omogućava automatizaciju procesa poslovanja elektronske uprave zamišljena je da na početnoj strani, ponudi opcije za selekciju kategorije koja nas zanima, bilo to pregled pasoša/ličnih karti, građana, maloletnih lica, uplata, informacija o pasošima, produžetaka i izgubljenih dokumenata.

4.3. Postupak implementacije rešenja

U okviru ovog dela, biće prikazani neki delovi koda za backend i frontend deo aplikacije.

4.3.1. Backend

Na Slici 1, dat je prikaz metode *Index*, putem koje se vrši ispis liste podataka iz baze za tabelu pasoš.

```
// GET: Pasos
0 references
public ActionResult Index()
{
    var pasos = db.Pasos.Include(p => p.Građjanin).Include(p => p.InformacijeOPasosu).
        Include(p => p.IzgubljenPaso).Include(p => p.MaloletnoLouse).Include(p => p.Uplata);
    return View(pasos.ToList());
}
```

Slika 1. Metoda za ispitivanje liste pasoša

U kodu koji sam prikazao generiše se *GET* metoda za prikazivanje podataka o pasošima. Akcija '*Action Index*' metoda se odnosi na *HTTP GET* zahtev i naziva se akcijom koja predstavlja deo kontrolera koji odgovara na taj zahtev. Linija koda: '*return View(pasos.ToList());*' šalje podatke o pasošima na odgovarajući pogled (engl. *View*) za prikazivanje. *HTTP GET* je standardna akcija za prikazivanje podataka. Kada korisnik prvi put poseti stranicu ili izvrši zahtev za prikazivanje podataka. U ovoj

5. ZAKLJUČAK

Budući da veb aplikacije koje se fokusiraju na onlajn za izdavanje pasoša i lične karte u javnoj upravi nisu široko istražene i implementirane, glavni cilj ovog rada bio je pružiti optimalno i *user-freindly* rešenje za ovu oblast.

Upotrebom najpopularnijih tehnologija, kao što su *Visual Studio* i *EntityFramework*, implementirana je data veb aplikacija za izdavanje ličnih dokumenata (pasoša i lične karte). Ovaj rad obuhvata detaljan opis i razvoj projekta. Na početku, predstavljen je uvod u temu i teorijske osnove. Sledeći deo rada fokusira se na opis realnog sistema i okruženja u kojem je projekat implementiran. Da bi ste doveli aplikaciju do savršenstva, razmatranjem određenih unapređenja može značajno poboljšati korisničko iskustvo i bezbednost.

Dodavanje opcije za komentare i ocene korisnika može pružiti povratne informacije o kvalitetu usluga. Osveženje ili poboljšanje interfejsa može doprineti boljoj preglednosti i lakšem korišćenju. Implementacija dodatnih koraka autentifikacije, poput potvrde putem imejla ili broja telefona, pojačava sigurnost korisničkih naloga [4-5].

6. LITERATURA

- [1] Combe C. Introduction to E-business. Routledge; 2012.
- [2] Elektronsko poslovanje. Preuzeto sa: https://v2.link-onlineservice.com/media/files/EP_Skripta1.pdf. Poslednji put posećeno: 15.05.2023.
- [3] Dewson R. Sql server management studio. *Beginning SQL Server 2008 for Developers: From Novice to Professional*. 2008; 25-50.
- [4] Perez J.J., Guckenheimer S. Software engineering with Microsoft visual studio team system. Pearson Education 2006.
- [5] Dykstra T., Anderson R. Getting Started with Entity Framework 6 Code First using MVC 5. Microsoft Corporation 2014.

Kratka biografija:



Petar Maksimović rođen je u Novom Sadu 1991. god. Master rad na Fakultetu tehničkih nauka iz oblasti Inženjerstvo informacionih sistema odbranio je 2023.
kontakt:
maksimovic.petar991@gmail.com

**INTEGRISANI INFORMACIONI SISTEM ZA REZERVACIJU LEKARSKIH PREGLEDA
INTEGRATED INFORMATION SYSTEM FOR BOOKING MEDICAL APPOINTMENTS**Nađa Jovanović, *Fakultet tehničkih nauka, Novi Sad***Oblast – Informacione tehnologije**

Kratak sadržaj – Sa sve većim razvojem tehnologije, informacije se lakše i brže dobijaju. Upotreba tehnologije je doprinela da se informaciono-komunikacione tehnologije primene i u javnim upravama. U ovom radu da će se uvid u pojmove e-uprava i e-zdravstvo. Biće urađeno istraživanje o tome da li su sistemi za rezervaciju lekarskih pregleda potrebni kao sistemi i da li bi bili korišćeni i rezultati tog istraživanja.

Ključne reči: sistemi elektronske uprave, e-uprava, lekarski pregled, e-zdravstvo

Abstract – With the increasing development of technology, information is easier and faster to obtain. The use of technology has contributed to the application of information and communication technologies in public administrations. In this paper, we will look at the terms e-government and e-health. Research will be done on whether medical appointment booking systems are needed as systems and would be used and the results of that research.

Keywords: electronic administration systems, e-government, medical appointment, e-health

1. UVOD

Razoj tehnologije je omogućio da se neki procesi olakšaju i da su informacije dostupne neprestano. Tehnologija počinje da se koristi svakodnevno i sve više stvari postaje nezamislivo bez nje. To dovodi do toga da se informaciono-komunikacione tehnologije (u daljem delu teksta IKT) uvede i u javne uprave. Uveden je novi pojam **e-uprava**. Uvođenjem ovog pojma građanima će biti olakšani procesi javnih uprava i uštedeće im se vreme.

Ovaj rad će se baviti istraživanjem da li sistemi za rezervaciju lekarskih pregleda kao sistemi potrebni i da li bi bili korišćeni. To će se izvršiti anketom. Nakon toga će biti prikazani rezultati ankete i njena analiza.

Drugo poglavlje će se detaljnije opisati e-uprava. Treće poglavlje će detaljnije opisati e-zdravstvo. Četvrto je objašnjen sistemski pregled literature. Peto poglavlje će biti opisana metodologija istraživanja, a u šestom rezultati ankete. U sedmom poglavlju je objašnjen sistem. Poslednje poglavlje je zaključak o temi i literatura.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Teodora Vučković, docent.

2. E-UPRAVA

E-uprava je doprinela efikasnosti pružanja usluga državnih institucija, a njenim korisnicima je olakšalo pružanje tih usluga, kao i dostupnost informacija neprestano.

Ciljevi uvođenja e-uprave su:

- smanjenje troškova,
- brže i kvalitetnije pružanje usluga,
- unapređenje pružanja usluga,
- poboljšanje javne uprave,
- bolja pregledanost informacija, smanjenje vremena, i
- tačnost informacija.

Postoji više definicija e-uprave, ali sve te definicije pokazuju da e-uprava koristi IKT da bi svojim korisnicima olakšala procese i imali dostupnost informacija neprestano, a državnim institucijama poboljšala rad i smanjila vreme [1-3].

Korisnici e-uprave se dele u 5 kategorija [4]:

1. građani (C - Citizen)
2. poslovni sektor (B - Business)
3. javna uprava (G - Government)
4. zaposleni (E - Employee)
5. civilno društvo (CS – Civil Society)

Odnosi prethodno napomenutih kategorija [1-4]:

1. **G2C** (odnos javna uprava – građani) – interakcija javne uprave i građana, a interakcija se odvija pružanjem usluga.
2. **G2G** (odnos javna uprava – javna uprava) – razmenjivanje podataka i informacija između državnih organa.
3. **G2E** (odnos javna uprava – zaposleni) – razmenjivanje podataka i informacija između zaposlenih u javnoj upravi, koristeći IKT.
4. **G2B** (odnos javna uprava – poslovni sektor) – saradnja javne uprave i drugih poslovnih subjekata i pravnih lica.
5. **G2CS** – (odnos javna uprava – civilno društvo) – pruža usluge organizacijama civilnog društva koje koriste IKT.

Iako je uvođenje e-uprave doprinelo poboljšanju usluga javnih uprava, kao i bržem i konstatnom pristupu informacija, doprinelo je i nekim izazovima.

Neki od izazova uvođenja e-uprave su [1-4].:

- treba pružiti prekogranično pružanje usluga, i
- obezbediti odgovarajući nivo bezbednosti podataka i informacija bitnih za svakodnevno poslovanje javnih uprava, kao i zaštitu privatnosti.

Sistemi e-uprave je potrebno unapređivati i modernizovati kako se IKT razvija i unapređuje. Svaka država sveta je uvela e-upravu u rad svojih državnih institucija. U Sjedinjenim Američkim Državama, Danskoj, Švedskoj i Velikoj Britaniji je e-uprava najrazvijenija.

2.1. E-uprava u Republici Srbiji

Republika Srbija uvodi pojam e-uprava 2004. godine kad je uvela *Strategiju reforme državne uprave u Republici Srbiji*, kad i započinje njen razvoj. Ovaj dokument nastoji da će državne institucije da se modernizuju, reorganizuju i da će biti pripremljene za sve izazove koje će informaciono društvo da donese, kao i uvođenje IKT u rad državnih institucija [1, 2, 3, 5].

Usvaja se *Strategija javne uprave u Republici Srbiji*, pošto su strategijom uočene prepreke za funkcionisanje i uspostavljanje e-uprave. Ova strategija nastoji da upotreba IKT treba da bude usmerena na efikasnost i efektivnost obavljanja poslova državnih organa, ali i da pruži usluge građanima i javnim licima. Osim ove dve strategije, uvode se još dva, a to su *Strategija razvoja elektronske uprave u Republici Srbiji za period od 2015-2018* i *Akcionni plan za njeno sprovođenje 2015-2016 godine*. One nastoje da oblasti poslovanja e-uprave treba da budu objedinjene i sigurnost podataka treba da bude osigurana [5].

U Republici Srbiji, e-uprava je implementirana u vidu portala eUprava. Ovaj portal predstavlja sistem ka elektronskim uslugama državnih organa koje su namenjene fizičkim i pravnim licima. Portal poseduje 3 odeljka, pod nazivima „Građani“, „Privreda“ i „Država“. Svaki odeljak pruža posebne usluge, koje zavise od toga koje je pravo, obaveza ili pravni interes građana.

3. E-ZDRAVSTVO

Zdravstveni sistem je državna institucija za čiji sektor su svi članovi društva zainteresovani i njegovo poboljšanje je bitano za sve članove društva.

Journal of medical internet research definiše pojam e-zdravstvo kao ne samo tehnički razvoj, već i stanje uma, način razmišljanja, stav i posvećenost umreženom i globalnom razmišljanju, poboljšanju zdravstvene zaštite na lokalnom, regionalnom i svetskom nivou korišćenjem IKT. Takođe, da je e-zdravstvo novija oblast medicinske informatike, javnog zdravlja i poslovanja i odnosi se na zdravstvene usluge i informacije koje se isporučuju ili unapređuju putem interneta [7, 8]. Pojam e-zdravstvo može da se definiše i kao sveobuhvatni izraz za kombinovanje upotrebe IKT u zdravstvenom sektoru za klinička, obrazovna istraživanja i administrativne svrhe [9].

E-zdravstvom informacije će se pouzdano stvarati, obrađivati, i razmenjivati usled brzih promena i različitih sistema. Povezanost informacionih sistema zdravstvenih usluga može doprineti da se spreči nepotrebni medicinski postupci, što pacijentima smanjuje troškove [6].

Ciljevi primene e-zdravstva su:

- zdravstveni status se beleži u elektronskoj formi,
- neprestani pristup tim podacima,
- efikasnost sistema,

- smanjenje troškova,
- smanjenje broja poseta u ordinacijama,
- pružanje pravih informacija, pojednostavljenje medicinskih usluga,
- pojednostavljenje manipulacije receptima,
- poboljšanje kvaliteta nege,
- povećanje produktivnosti zdravstvenog sektora, i
- usluge su usmerene ka očuvanju i postizanju zdravlja.

Izazovi razvoja e-zdravstva [6, 7]:

- nedovoljna informisanost državnih institucija i zdravstvenih radnika o potencijalnim mogućnostima primene IKT u zdravstvenom sektoru,
- manjak standardizacije,
- nedostatak doslednosti prilikom provođenja donesenih planova i inicijativa,
- nedovoljno uključenih medicinskih stručnjaka, IKT stručnjaka i ostalih učesnika zdravstvenog sistema u razvoj e-zdravstva,
- nedostatak finansijskih sredstava za razvoj e-zdravstva,
- velike i komplikovane aplikacije,
- kvalitet informacija,
- zdravstvena pismenost pacijenata i njihova sposobnost da razumeju informacije,
- nedostatak zakonske regulative i fragmentacija e-zdravstva,
- mogućnost zloupotrebe podataka, i
- nejednak pristup uslugama e-zdravstva za sve.

Potrebno je ulagati u e-zdravstvo, da bi se on razvijao i unapređivao, i time bio efikasniji za njegove korisnike.

Neki od izvora finansiranja e-zdravstva su: subvencije države, regionalni fondovi, privatni investitori, banke, međunarodne institucije, kao što su Svetska zdravstvena organizacija i Svetska banka [9].

Evropska Unija podstiče da svaka njena članica ima razvijeno e-zdravstvo, tako što pomaže i sarađuje sa nacionalnim vlastima oko mobilnosti pacijenata, usklađivanja i povezivanja informacijskih i zdravstvenih sistema, infrastrukture, usvajanja inovativnih tehnologija kao i rešavanja zakonskih pitanja. Cilj Evropske Unije je da stvori evropsko područje e-zdravstva koje će biti okvir zajedničkog delovanja i interakcije u e-zdravstvu, i osiguraće povoljno okruženje za njegov dalji razvoj i širenje [6].

Države u kojima je e-zdravstvo najviše razvijeno su Danska, Holandija, Finska, Švedska i Velika Britanija.

4. SISTEMSKI PREGLED LITERATURE

Prvi korak ka planiranju pregleda literature je da se odabere tema i postave istraživačka pitanja. Tema za koju će se vršiti sistemski pregled literature su e-uprava i e-zdravstvo, sa većim fokusom na e-zdravstvo.

Istraživačka pitanja na koja treba dati odgovor su:

1. Koje su prednosti e-uprave?
2. Koje su prednosti e-zdravstva?
3. Kakav uticaj e-zdravstvo ima na način života?
4. Koje su pogodnosti e-zdravstva?

Za pretraživanje radova koristio se Google Scholar. Ključne reči koje su se koristile za pretragu su: e-health i e-government. Nakon unošenja ključnih reči, dobijeno je 1480 rezultata. Kriterijumima inkluzije i eskluizije će se utvrditi koje radove treba uključiti, a koje treba odbaciti. Na osnovu kriterijuma inkluzije i eskluizije, odabrano je 13 radova.

5. METODOLOGIJA ISTRAŽIVANJA

Kao što je već napomenuto, potrebno je istražiti da li su sistemi za rezervaciju lekarskih pregleda kao sistemi potrebni i da li bi bili korišćeni. To je bila primarna svrha sprovedenog istraživanja, razumeti stanje i perspektivu mogućeg razvoja ovakvog sistema. Razvijanje ovakvog sistema bi njenim korisnicima olakšala dolazak kod lekara, korisnici bi imali uvid u svoje zdravstveno stanje i u sve svoje odrađene preglede i i svoje rezervisane preglede.

Pre istraživanja, potrebno je uraditi teorijski pregled na temu koju istražujemo. Za potrebe sistema za rezervaciju lekarskih pregleda, urađen je teorijski pregled na temu e-uprava i e-zdravstvo. Nakon toga se pravi anketu, po informacijama koje smo dobili iz teorijskog pregleda i istraživanja, a zatim se započinje razvoj sistema. U poglavlju 5. biće objašnjeni rezultati anekete.

Kao osnovu za istraživanje, odabran je portal eUprava. Potrebno je pogledati da li već postoje ovakvi sistemi i na osnovu toga kreirati anketu da li da se takvi sistemi unaprede ili uvedu. Pošto se utvrdilo da takvi sistemi ne postoje, anketa se pravi na osnovu toga da li su takvi sistemi potrebni i da li bi bili korišćeni. Anketa se sastoji od 11 pitanja, i popunjena je od strane 111 ispitanika.

Pitanja iz ankete su:

1. Pol – muški ili ženski
2. Kojoj starosnoj grupi pripada ispitanik (18-24, 25-34, 35-44, 45-54, 5++)
3. Gde je ispitanik zaposlen – privatni preduzetnik, državna institucija, IT sektor, banka/osiguravajuće društvo, ostalo
4. Da li su čuli za portal eUprava – da ili ne
5. Da li su koristili portal eUprava – da ili ne
6. Koliko bi ispitaniku uvođenje sistema za rezervaciju lekarskih pregleda u javnim ustanovama olakšalo odlazak kod lekara – skala od 1 do 5, gde je 1 ne bi olakšalo, 5 olakšalo bi
7. Koliko bi ispitaniku uvođenje sistema za rezervaciju lekarskih pregleda olakšalo preglednost svih njegovih rezervacija - skala od 1 do 5, gde je 1 ne bi olakšalo, 5 olakšalo bi
8. Koliko bi ispitaniku uvođenje sistema za rezervaciju lekarskih pregleda olakšalo preglednost svih njegovih odrađenih pregleda - skala od 1 do 5, gde je 1 ne bi olakšalo, 5 olakšalo bi
9. Da li ispitanik misli da bi bilo efikasno uvesti sistem za rezervaciju lekarskih pregleda – bilo bi efikasno, ne bi bilo efikasno i nisam siguran/na
10. Da li bi ispitanik koristio sistem za rezervaciju lekarskih pregleda – da ili ne
11. Kad bi se uveo sistem za rezervaciju lekarskih pregleda, koje bi, po ispitanikovom mišljenju, bile njegove prednosti (Moguće je izabrati više odgovora)

– ušteda vremena, bolji pregled informacija, brzina dobijanja informacija, lakoća dobijanja informacija o slobodnim terminima i ostalo

Kad se utvrdi potrebnost sistema za rezervaciju lekarskih pregleda i da li će biti korišćeno, onda se započinje sa njenim razvojem, što će biti objašnjeno u poglavlju 6.

6. REZULTATI ISTRAŽIVANJA

Kao što je već rečeno, anketu je popunilo 111 ispitanika. Po rezultatima, anketu je popunilo više ispitanika ženskog pola 70.3%, nego muškog pola 29.7%.

Najviše ispitanika je činilo osobe starosne grupe 25-34, 36.9%, a najmanje ispitanika je činilo osobe između 18-24 godina. Starosnu grupu između 35-44 je činilo 25.2% ispitanika, a starosnu grupu između 45-54 je činilo 19.8% ispitanika, dok starosnu grupu 55+ nije niko popunio.

Najviše ispitanika je zaposleno kao privatni preduzetnik 31.5%, dok onih koji rade u državnim institucijama čini 27.9%. Oni koji su zaposleni u IT sektoru čini 18.9% ispitanika, a najmanje je zaposleno u bankama/osiguravajućim društvima 4.5%. 5.4% ispitanika čine studenti, a 3.6% ispitanika je nezaposleno, a 0.9% ispitanika je freelancer 0.9%, kao i onih koji su prevodioci, fudbalski trener, privatni sektor, radi u novinarskoj kući, poseduje privatnu kompaniju, radi u logistici i šrediciji, u hemijskoj industriji ili u zdravstvenoj ustanovi.

98.2% ispitanika je reklo da je čulo za portal eUprava, dok 1.8% ispitanika nije uopšte čulo za ovaj portal.

89.2% ispitanika je koristilo portal eUprava, dok 10.8% nije uopšte koristilo ovaj portal.

69.4% ispitanika misli da bi uvođenje sistema olakšalo odlazak kod lekara, 18% misli da bi u velikoj meri olakšalo, 10.8% ispitanika misli da bi u manjoj meri, dok 1.8% misle da ne bi uopšte olakšalo.

73% ispitanika misli da bi uvođenjem sistema olakšalo preglednost njihovih rezervacija, 12.6% misli da bi u velikoj meri olakšalo, 11.7% misli da bi u manjoj meri olakšalo, 2.7% misli da ne bi olakšalo, a niko nije odgovorio da ne bi olakšalo uopšte.

73.9% misli da bi uvođenjem sistema olakšalo preglednost svih odrađenih pregleda, 15.3% misli da bi olakšalo u velikoj meri, 9.9% misli da bi olakšalo u manjoj meri, 0.9% misli da ne bi olakšalo, dok niko nije odgovorio da ne bi uopšte olakšalo.

86.5% ispitanika misli da bi uvođenje sistema bilo efikasno, 11.7% ispitanika nije sigurno, a 1.8% misli da ne bi bilo efikasno.

93.6% ispitanika kaže da bi koristili sistem, dok 6.4% ispitanika ne bi koristilo ovaj sistem.

82% ispitanika je reklo da prednost uvođenja sistema ušteda na vremenu, 55.9% je reklo da je prednost bolji pregled informacija, 68.5% je reklo da je prednost brzina dobijanja informacija, a 82.9% lakoća dobijanja informacije o slobodnim terminima. Jedan ispitanik kaže da je prednost to što bi se izbegle gužve i bespotrebna čekanja, a dva ispitanika kažu da nemaju mišljenje i nisu sigurni.

7. KREIRANJE SISTEMA ZA REZERVACIJU LEKARSKOG PREGLEDA

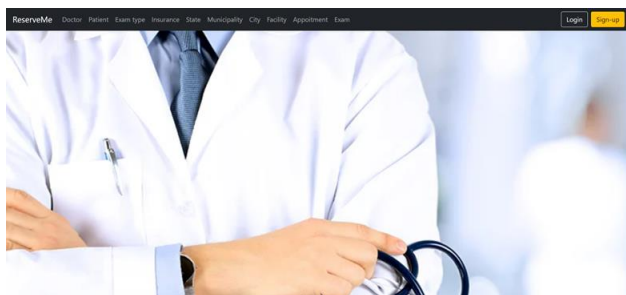
Nakon završene analize ankete i uvida u to da su sistemi za rezervaciju lekarskih pregleda potrebni i bili bi korišćeni, ide razvoj tog sistema.

Tehnologije koje su korišćene za razvoj ovog sistema su:

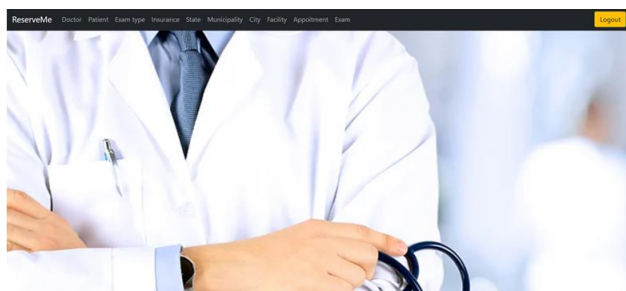
- **PostgreSQL** – za kreiranje baze podataka
- **IntelliJ IDEA** – za kreiranje backend dela
- **Angular** i **Visual Studio** - za kreiranje frontend dela
- **GenMyModel** i **LucidChart** – za kreiranje UML dijagrama

Baza podataka se kreirala na osnovu dijagrama klase. Dijagram klasa za lekarski pregled se sastoji od 10 klasa, koje su međusobno povezane. Klase su: Exam, ExamType, Insurance, Appointment, Facility, City, Municipality, State, User(Doctor i Patient). Tabele u bazi podataka su kreirane pomoću **DDL** (*Data Definition Language*), a te napravljene tabele se popunjavaju pomoću **DML** (*Data Manipulation Language*).

Na slici 1. prikazana je početna stranica aplikacije. Na slici 2. je prikazana početna stranica kad se korisnik uloguje.



Slika 1. Početna stranica aplikacije



Slika 1. Početna stranica aplikacije kad se korisnik uloguje

8. ZAKLJUČAK

E-uprava je olakšala mnoge procese državnim institucijama, informacije su dostupne u svakom trenutku. Ciljevi e-uprave su da smanji troškove, brže i kvalitetnije pružanje usluga, bolja pregledanost informacija, njihova tačnost, smanjuje se vreme... E-uprava koristi IKT da bi poboljšala rad državnih institucija i olakšala korisnicima proces. U Republici Srbiji se e-uprava implementirala u vidu portala eUprava.

E-zdravstvo je novija oblast medicinske informatike, javnog zdravlja i poslovanja, i odnosi se na zdravstvene usluge i informacije koje se isporučuju ili unapređuju putem interneta. Ciljevi e-zdravstva su: efikasnost sistema, smanjenje troškova, smanjenje broja poseta u ordinacijama, zdravstveni sistem se beleži u elektronskoj formi i neprestani pristup im informacijama, pružaju se prave informacije... Potrebno je ulagati i unapređivati u e-zdravstvo.

Dat je sistemski pregled literature na temu e-health i e-government. Na osnovu kriterijuma inkluzije i eskluizije, izabrano je 14 radova.

Na osnovu rezultata istraživanja utvrđeno je da, sistemi za rezervaciju lekarskih pregleda su potrebni i da bi bili korišćeni. Na osnovu ovih rezultata počinje razvoj sistema. Sistem za rezervaciju lekarskih pregleda sastoji se od 10 tabela, koje su kreirane na osnovu dijagrama klase: Exam, ExamType, Insurance, Appointment, Facility, City, Municipality, State, User(Doctor i Patient). Takođe, je prikazan izgled početne stranica aplikacije, kao i stranice kad se korisnik prijavi.

9. LITERATURA

- [1] Dr Prlja D., E-uprave u Evropskoj Uniji i u Srbiji, 2006., 55-81
- [2] Vidas-Bubanja M., Prednosti i ograničenja izgradnje e-uprave u Srbiji, 2011., 73-88
- [3] Bradić-Martinović A., E-uprava u Srbiji - podrška procesima javnih nabavki, 230-244
- [4] Stefanović D., Anderla A., Mitrović S., Nikolić J., Elektronska uprava u Srbiji, Vol. 13, 2014, 718-722
- [5] Vučinić D., Elektronska uprava – Koncept usluge sa osvrtom na elektronsku upravu u Republici Srbiji, br. 1, 2020, 46-56
- [6] Ostojić R., Bilas V., Franc S., E-zdravstvo – unapređenje zdravstvenog sistema primenom informacijske i komunikacijske tehnologije, 2012, 84-862
- [7] Wicks P., Stamford J., Grootenhius M. A., Haverman L., Ahmed S., Innovations in e-health, 2014, 195-203
- [8] Eysenbach G., What is e-health?, Journal of medical internet research
- [9] S. Yunkap Kwankam, What e-health can offer, 800-801

Kratka biografija:



Nada Jovanović rođena je 16.04.1997. Diplomirala na smeru Inženjerstvo informacionih sistema 2020 godine, kad upisuje master na istom smeru.

kontakt: j.nadja97@gmail.com

U realizaciji Zbornika radova Fakulteta tehničkih nauka u toku 2023. godine učestvovali su sledeći recenzenti:

Aco Antić	Dunja Vrbaški	Marinko Maslarić	Nemanja Tasić
Aleksandar	Đorđe Vukelić	Marko Marković	Nenad Simeunović
Anđelković	Goran Jeftenić	Marko Todorov	Nikola Vojnović
Aleksandar Kovačević	Goran Savić	Marko Vekić	Platon Sovilj
Aleksandar	Goran Sladić	Maša Bukurov	Predrag Teodorović
Kupusinac	Goran Stojanović	Mijodrag Milošević	Radivoje Dinulović
Aleksandar Selakov	Goran Tepić	Milan Delić	Radomir Kojić
Aleksandra Radulović	Gordan Stojić	Milan Gavrić	Romana Bošković-
Aleksandra Pešterac	Gordana Ostojić	Milan Marinković	Živanović
Ana Nešić Tomašević	Gordana	Milan Mirković	Sandra Dedijer
Andraš Anderla	Milosavljević	Milan Rapajić	Saša Medić
Andrija Rašeta	Igor Dejanović	Milan Rackov	Slavica Mitrović
Atila Zelić	Igor Maraš	Milan Segedinac	Senka Bajić
Bojan Batinić	Igor Peško	Milan Trivunić	Slađana Milićević
Bojan Matić	Ilija Bašićević	Milan Vidaković	Slobodan Morača
Bojan Tepavčević	Iva Šiđanin	Milena Krklješ	Slobodan Šupić
Bojan Jovanović	Ivan Mezei	Milica Vračarić	Slobodan Tabaković
Boris Agarski	Ivan Prokić	Milica Miličić	Srđan Popov
Boris Stojić	Ivana Mihajlović	Milica Kisić	Srđan Vukmirović
Branko Milosavljević	Ivana Vasiljević	Miloš Simić	Stevan Gostojić
Damir Đaković	Ivana Katić	Miloš Šešlija	Stevan Milisavljević
Danijela Ćirić	Ivana Maraš	Milovan Lazarević	Stevan Stankovski
Danijela Gračanin	Ivana Miškeljin	Milja Simeunović	Suzana Draganić
Danijela Lalić	Jelena Atanacković	Miljana Prica	Svetlana Bačkalić
Darko Čapko	Jeličić	Miodrag Milutinov	Svetlana Nikoličić
Darko Stefanović	Jelena Borocki	Miodrag Žigić	Tamara Škorić
Dejan Ecet	Jelena Ivetić	Miroslav Dramićanin	Teodora Vučković
Dejan Reljić	Jelena Radonić	Miroslav Zarić	Vesna Stojaković
Dejan Movrin	Jelena Slivka	Mirko Raković	Višnja Žugić
Dejan Ubavin	Jelena Spajić	Miro Govedarica	Vladimir Đaković
Dejana Nedučin	Kalman Babković	Miroslav Kljajić	Vladimir Mučenski
Dragan Adamović	Lazar Kovačević	Miroslav Zarić	Vojin Ilić
Dragan Dinu	Lidija Krstanović	Mladen Tomić	Vuk Bogdanović
Dragan Ivanović	Ljiljana Popović	Mladen Radišić	Vuk Vranjkovic
Dragan Ivetić	Ljubica Duđak	Nataša Milosavljević	Zoran Brujić
Dragan Jovanović	Ljubo Nedović	Nebojša Brkljač	Zoran Čepić
Dragan Pejić	Ljubomir Budinski	Nebojša Radović	Zoran Jeličić
Dragan Ružić	Magdolna Pal	Nebojša Ralević	Željen Trpovski
Dragana	Maja Turk Sekulić	Neda Milić Keresteš	Željko Kanović
Konstantinović	Maja Petrović	Nemanja Kašiković	
Dragoljub Šević	Marija Silađi	Nemanja Sremčev	
Drago Žarković			