Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/25BE16Popovic

DEVOPS PRISTUP RAZVOJU APLIKACIJE SA INTEGRACIJOM GENERATIVNOG JEZIČKOG MODELA

DEVOPS APPROACH TO APPLICATION DEVELOPMENT WITH INTEGRATION OF A GENERATIVE LANGUAGE MODEL (GPT)

Miloš Popović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – PRIMENJENE RAČUNARSKE NAUKE I INFORMATIKA

Kratak sadržaj – *U ovom radu opisan je razvoj i praćenje (monitoring) aplikacije po DevOps principima. Aplikacija služi za generisanje plana objava i ideja na društvenim mrežama uz pomoć GPT generativnog jezičkog modela.*

Ključne reči: *DevOps, monitoring, generativno jezički modeli.*

Abstract – *This paper describes the development and monitoring of an application following DevOps principles. The application is used for generating social media posting schedules and ideas with the assistance of the GPT generative language model.*

Keywords: *DevOps, monitoring, generative language models*

1. UVOD

DevOps pristup se sve više prepoznaje kao ključni faktor za brz i pouzdan razvoj, implementaciju i održavanje aplikacija. Sa druge strane, generativni jezički modeli, kao što je "Chat GPT", pružaju mogućnost za automatizaciju procesa komunikacije i kreacije sadržaja. Integracija ovih generativnih jezičkih modela u aplikacije poboljšava korisničko iskustvo pošto na jednostavan način možemo omogućiti da aplikacija vrši kompleksnije provere korisničkog unosa, kao i da omogućimo generisanje sadržaja personalizovanog za svakog korisnika, bez potrebe za ručno programiranim kompleksnim pravilima i logikom.

Za potrebe rada razvijana je aplikacija koja korisnicima, koji u ovoj aplikaciju predstavljaju vlasnici biznis naloga na društvenim mrežama, omogućava da generišu ideje za objave na društvenim mrežama. Proces aplikacije se sastoji iz dva osnovna koraka. U prvom koraku korisnik unosi delatnost za koju želi da generiše objave na društvenim mrežama. Nakon toga GPT model generiše pitanja karakteristična za tu delatnost na koje korisnik treba da odgovori.

Drugi korak integriše te odgovore u novi zahtev za generisanje ideja i plana objava za društvene mreže.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Miroslav Zarić, vanr. prof.

1.2. *GPT - Jezički generativni model*

GPT (*Generative Pre-trained Transformer*) je vrsta naprednog računarskog modela zasnovanog na dubokom učenju koji se koristi za obradu prirodnog jezika. Ovaj model je obučen na ogromnom skupu tekstualnih podataka i sposoban je za razumevanje, generisanje i obradu teksta na prirodnom jeziku. GPT modeli su razvijeni da bi razumeli kontekstualne veze između reči i fraza u tekstu i mogu da izvršavaju različite zadatke, uključujući prevođenje jezika, generisanje teksta, odgovaranje na pitanja i mnoge druge.

Ono što GPT modele čini posebnim jeste njihova sposobnost prepoznavanja i generisanja ljudski razumljivog teksta na osnovu ulaznih informacija. Modeli kao GPT-3, na primer, mogu generisati vrlo složene tekstove koji zvuče kao da su napisani od strane stvarne osobe. Ovi modeli su postali ključni u mnogim aplikacijama koje zahtevaju obradu prirodnog jezika, kao što su automatsko odgovaranje na pitanja, generisanje sadržaja, personalizovane preporuke i drugo.

GPT pored tekstualnog upita prima i sledeće parametre:

- **Temperature**: Ovaj parametar kontroliše koliko je generisanje teksta predvidljivo. Više vrednosti povećavaju nasumičnost odgovora, dok niže vrednosti (bliže 0) deluju deterministčnije i odabiraju najverovatniji izlaz.
- **Maximum length**: Ovaj parametar kontroliše maksimalnu dužinu odgovora koji generiše model. Ako je postavljena visoka vrednost, model će biti sposoban da generiše duže odgovore.
- **Stop sequences**: Ovaj parametar omogućava da se postave određene sekvence teksta na kojima može se zaustaviti generisanje teksta, na primer, ako se uključi reč "kraj" kao stop sekvencu, model će prestati sa generisanjem teksta kada naiđe na ovu reč.
- **Top P**: Predstavlja širinu vokabulara iz kojeg model može uzeti reči pri generisanju teksta. Na primer, ako je Top P niže, model će koristiti samo najverovatnije reči. Ako je Top P više, raznovrsnije reči mogu biti uključene.
- **Frequency Penalty**: Ovaj parametar smanjuje verovatnoću često korišćenih fraza. Više vrednosti znače veću kaznu za česte fraze.
- **Presence Penalty**: Ovaj parametar smanjuje verovatnoću reči koje su već upotrebljene. Više vrednosti znače veću kaznu za reči koje su već prisutne u tekstu.

1.3. Jaeger Telemetry

Jaeger telemetry je sistem za prikupljanje i analizu telemetrijskih podataka u kontekstu praćenja i analize performansi i dostupnosti aplikacija i servisa. Ovo je jedan sistem otvorenog koda koji pomaže inženjerima da brže otkrivaju i rešavaju probleme u svojim aplikacijama, smanjuje vreme do otkrivanja grešaka i na taj način poboljšava performanse aplikacije.

Jaeger je posebno koristan za identifikaciju uskih grla u performansama, otkrivanju grešaka i rešavanja problema u sistemu. Osim toga, omogućava efikasno analiziranje performansi tokom vremena, što je ključno za optimizaciju aplikacija.

1.4. FluentBit

Fluentbit je softver otvorenog koda za dobavljanje i prosleđivanje podataka u realnom vremenu. On je dizajniran da omogući efikasno prikupljanje i obradu podataka iz raznih izvora u raznim okruženjima. Fluentbit je najčešće korišćen za prikupljanje, filtriranje i prosleđivanje logova.

Podržava razne izvore podataka kao što su tekstualni logovi, standardni ulaz/izlaz aplikacije (stdio), JSON poruke, sistemski logovi, i mnoge druge formate. Omogućava filtriranje podataka koristeći brojne filtere kako bi se izdvojili i obradili željeni podaci. Ovo pomaže u smanjenju obima podataka koji se prosleđuju, čim se smanjuje opterećenje mreže i sistema za prikupljanje podataka. Fluentbit ima brojne izlazne destinacije, uključujući Elasticsearch, InfluxDB, Amazon S3, i mnoge druge. Ovo omogućava integraciju sa različitim sistemima za analizu i vizualizaciju podataka.

1.5 Prometheus

Prometheus je takođe sistem otvorenog koda za nadgledanje i alarmiranje, razvijen od strane SoundCloud-a. On predstavlja moćan alat za prikupljanje, nadgledanje i analizu podataka o radu aplikacija i infrastrukturi u realnom vremenu. Ovo je veoma korisno u okruženjima kod kojih je neophodno pružiti visoku dostupnost aplikacije [1].

Prometheus koristi tzv „strugače“ - skrejpere (*eng.* "*scrapers*") za prikupljanje metrika iz različitih izvora, kao što su aplikacije, servisi i infrastruktura. Skrejperi na određeni period (*long pooling*), zahtevaju informacije sa ciljanih izvora i prikupljaju metrike. Omogućava postavljanje alarma i pravila koja prate metrike i izazivaju alarme kada se zadovolji određeni uzorak [1].

Prometheus je dizajniran da bude lak za instaliranje i konfigurisanje, a ima i dobre performanse i skalabilnost. Možete ga koristiti i na malim i na velikim infrastrukturnim okruženjima.

1.5. Grafana

Grafana je popularna platforma otvorenog koda za prikaz i analizu podataka koja omogućava korisnicima da vizualizuju podatke iz različitih izvora na jednom mestu, kao što su metrike, logovi, tragovi (traces). Glavni akcenat Grafane je na monitoringu i analizi sistemskih podataka, ali je takođe korisna i za prikaz istorijskih podataka, kreiranje panela za upravljanje.

Popularni alati koji mogu da se koriste sa Grafanom su:

- Jaeger ili Zipkin: Za praćenje performansi i analiza traga, Jaeger i Zipkin su popularni alati koji se koriste zajedno sa Grafanom.
- Grafana Loki: Loki je alat za skladištenje logova razvijen od strane istih ljudi koji su radili na Prometheus-u. Loki omogućava skalabilno i efikasno skladištenje logova i može se integrisati direktno sa Grafanom. U ovom projektu koristiće se FluentBit koji šalje logove na Loki.
- Prometheus: U kombinaciji sa Grafanom, omogućava vam vizualizaciju metrika i pravljenje panela za praćenje performansi aplikacija i infrastrukture.
- ELK Stack: Za praćenje logova, možete koristiti ELK stek koji uključuje Elasticsearch za skladištenje logova, Logstash za obradu i filtriranje logova, i Kibana (koji se takođe može integrisati sa Grafanom) za vizualizaciju i analizu logova.

Grafana ima bogat ekosistem dodataka (plagina) koji omogućavaju integraciju različitih izvora podataka i alata za logove, tragove (*tracing*) i metrike.

1.6. Pregled relevantne literature

DevOps integriše oblasti, razvoja i operacija, koristeći automatizaciju isporuke i nadzora infrastrukture. Ovaj pristup ima za cilj da omogući brži razvoj, bolju saradnju između timova i pouzdanu isporuku softvera.

Ključni elementi DevOps pristupa uključuju:

• Kontinualna integracija: Proces u kojem se promene u kodu redovno integrišu u glavnu granu repozitorijuma, praćene testovima koko bi se očuvala stabilnost aplikacije.

• Kontinualna isporuka: Automatski proces isporuke softvera u produkciji nakon uspešne integracije i testiranja.

• Automatizacija Infrastrukture: Upotreba skripti i alata za definisanje infrastrukture kao koda radi brže implementacije i skaliranja resursa.

• Praćenje Logova i Metrike: Skup alata i praksi za praćenje performansi aplikacije i identifikaciju problema u realnom vremenu.

”DevOps” nije jedan univerzalni proces ili metodologija kao što je na primer skram (eng. Scrum). ”DevOps” treba da se posmatra kao konceptualni okvir koji organizacije treba da prilagode svom jedinstvenom okruženju [3].

U radu ”Ebert, C., Gallardo, G., Hernantes, J. and Serrano, N., 2016. DevOps” navedeni su različiti alati koji se koriste u ovom pristupu razvoja softvera [2]:

1. Apache Ant: Alat za izgradnju aplikacija.
2. Maven: Slično kao Apache Ant samo unapređen
3. Jenkins: Ovo je alat za kontinuiranu integraciju.
4. Puppet: Ovo je alat za upravljanje konfiguracijama na serverima.
5. Chef: Ovaj alat takođe služi za automatizaciju upravljanja konfiguracijama.

6. Ansible: Ovo je alat za upravljanje konfiguracijom preko SSH (Secure Shell).
7. Loggly: Ovaj alat omogućava prikupljanje log podataka iz aplikacija, platformi i servera u realnom vremenu.
8. Nagios: Ovo je alat za nadgledanje i upravljanje zdravljem sistema.
9. New Relic: Omogućava praćenje performansi aplikacija.

Praktične primene GPT modela su opisane u radu ”GPT-3: What’s it good for?” Robert Dale [4]:

1. Kreativno pisanje: GPT-3 može generisati kreativne tekstove, uključujući pripovedanja, poeziju i dijaloge. Ovaj aspekt tehnologije može biti koristan u kreiranju filmova, video igara i drugih kreativnih projekata.
2. „Pojačano“ tj. unapređeno pisanje: GPT-3 može služiti kao alatka za pomoć u pisanju. Neke aplikacije već postoje koje omogućavaju korisnicima da unesu svoj tekst i dobiju alternativnu verziju na osnovu stila i sadržaja. Ovo može biti korisno za pisanje mejlova, blogova, reklamnih materijala itd.
3. Pitanja i odgovori: GPT-3 može odgovarati na pitanja na osnovu prethodno datog teksta. Ovaj aspekt tehnologije može biti koristan u obrazovanju, kada je potrebno dobiti odgovore na pitanja iz različitih oblasti znanja.
4. Prevod: GPT-3 može biti koristan za mašinski prevod teksta sa jednog jezika na drugi. Međutim, treba biti oprezan prilikom korišćenja ove funkcionalnosti, posebno kod prevoda pravnih dokumenata ili drugih važnih tekstova, jer GPT-3 može generisati netačne prevode.
5. Pomoć u fitnesu: Postoje aplikacije koje koriste GPT-3 za odgovaranje na pitanja iz oblasti fitnesa i zdravlja. Ovo može biti korisno za dobijanje saveta o vežbanju i ishrani.

Uprkos svojim moćnim sposobnostima, GPT-3 još uvek nema pravo razumevanje prirodnog jezika i suočava se sa izazovima u zadacima poput zaključivanja na osnovu prirodnog jezika i razumevanja pročitanog.

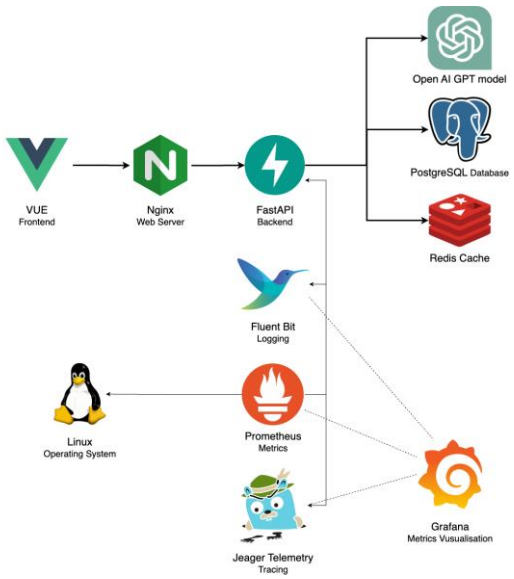
Ipak, GPT-3 je inspirisao nove ideje u oblasti dubokog učenja i ima potencijalnu primenu u inteligentnim pomoćnim zadacima [5].

2. SPECIFIKACIJA SISTEMA

Na slici 2.1 prikazana je arhitektura sistema pomenutog projekta za generisanje ideja za objave na društvenim mrežama pomoću GPT modela.

Aplikacija se sastoji iz Vue klijentskog dela aplikacije, koji predstavlja korisnički interfejs. Sa serverske strane se nalazi Nginx server koji je jedini otvoren prema internetu.

Nginx sve servere prosleđuje FastAPI aplikaciji (Glavnom servisu). Servisni deo aplikacije komunicira sa GPT modelom pomoću REST API-ja, PostgreSQL relacionom bazom podataka i Redis kešom.

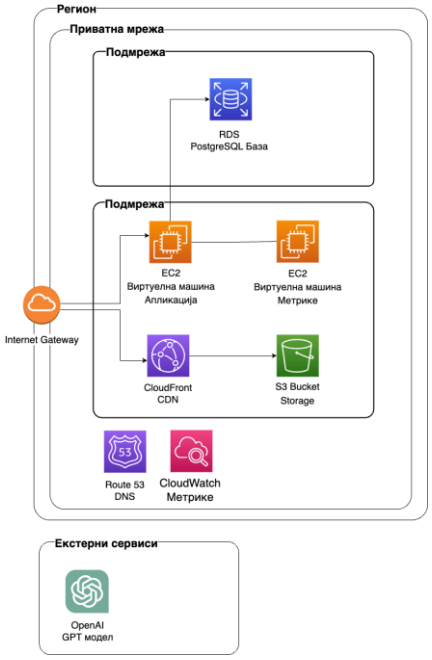


Slika 2.1 *Arhitektura sistema*

Od alata za praćenje (monitoring), koristi se Jaeger za praćenje tragova (tracing), FluentBit za agregaciju logova aplikacije, Prometheus za dobavljanje metrika sistema i aplikacije i Grafana za vizuelizaciju svih ovih podataka.

3. ISPORUKA NA CLOUD

Da bi aplikacija bila javno dostupna podignuta je na AWS cloud servis. na slici 3.1.1 je prikazana, infrastruktura servisa na AWS-u.



Slika 3.1.1 *Infrastruktura aplikacije na AWS-u*

Aplikacija je dodatno prilagođena da pomoću *opentelemetry* biblioteke generiše i šalje tragove sa informacijama o trajanju zahteva na Jaeger server, takođe pomoću iste biblioteke se ispisuju logovi koji se šalju na Fluentbit alat koji prikuplja logove i prosleđuje ih na Grafana Loki alat za agregaciju i dobavljanje logova.

Na mašini na kojoj je podignuta aplikacija instaliran je servis Node Exporter koji čita podatke o samoj mašini kao što su opterećenje procesora, zauzeće RAM-a, zauzeće masovne memorije i objavljuje ih na port 9100. Podatke sa tog porta čita alat Prometheus koji na određen vremenski period preuzme vrednosti i sačuva. Konačno Grafana alat se koristi za vizuelizaciju i pretragu svih podataka o radu aplikacije (tragovi, logovi i metrike). Svi dodatni servisi za monitoring aplikacije su podignuti na posebnoj EC2 instanci, osim Node Exportera koji je instaliran na instanci gde se nalazi aplikacija kako bi preuzimao informacije o toj mašini.

Za frontend aplikaciju je korišćen S3 bucket za smeštanje statičkih fajlova i CloudFront za distribuciju tih fajlova (frontend aplikacije). CloudFront je povoljno rešenje pošto je veoma skalabilno i naplaćuje se po količini zahteva, takođe pruža besplatan sertifikat za naš domen. Sertifikat za backend deo aplikacije je obezbeđen pomoću Certbot alata koji obezbeđuje "Let's Encrypt" sertifikat i automatski ga obnavlja.

GitHub je korišćen za kontrolu verzija projekta tokom razvijanja. Shodno tome GitHub akcije su upotrebljene za automatizaciju isporuke aplikacije. Prilikom postavljanja koda na glavnu granu repozitorijuma pokreće se akcija koja ima sledeće korake:

1. Preuzimanje sadržaja iz repozitorijuma
2. Pokretanje testova u aplikaciji
3. Kopiranje "bekend" programskog koda na udaljeni server (EC2 instancu)
4. Pokretanje skripte na serveru za ažuriranje
   - 4.1. Pravi kopiju trenutno aktivne aplikacije
   - 4.2. Pokreće kopiju privremeno na posebnom portu
   - 4.3. Menja port Nginx-a na privremeni (koji i dalje ima staru verziju
   - 4.4. aplikacije)
   - 4.5. Menja fajlove glavne aplikacije sa novom verzijom
   - 4.6. Pokreće se nova verzija
   - 4.7. Preusmerava se port u Nginx-u na početni na kom se sada nalazi nova
   - 4.8. aplikacija
   - 4.9. Gasi se i briše privremena verzija
5. Izgrađuje se Vue 3 "Frontend" aplikacija
6. Ažurira se "Frontend" aplikacija
   - 6.1. Briše se stari sadržaj iz S3 Bucket-a
   - 6.2. Otprema se novi statički fajlovi za "frontend"
   - 6.3. Invalidira se sadržaj CloudFront-a da bi se preuzela nova verzija na produkciji

5. ZAKLJUČAK

U ovom radu prikazan je jedan primer jezičkog generativnog modela u aplikaciju koja se isporučuje na platformi za računarstvo u oblaku korišćenjem DevOps principa. Prikazani su i načini za detaljno praćenje same aplikacije, kao što su praćenje logova, performansi servisa pomoću tragova (tracing) i opterećenja samog servera na kojoj se nalazi aplikacija. Prikazan je jedan od načina za isporuku aplikacije na AWS. Implementirana je kontinualna integracija, monitoring servisa na produkciji. Rešeni su sertifikati kako bi se omogućila bezbedna HTTPS konekcija.

Potencijalna unapređenja aplikacije bi išla u smeru zamene upotrebe virtuelnih mašina, koje može biti nezgodno za održavanje, kontejnerizovanim servisnim modulimas i isporuci na neki od servisa za rad sa kontejnerima. Mana tog pristupa je što može biti skuplji u odnosu na jednostavne virtuelne mašine. Pošto aplikacija trenutno ne rukuje sa velikim brojem korisnika i nije od kritičnog značaja da imamo savršenu dostupnost, možemo da prihvatimo nedostatke korišćenja virtuelnih mašina za servise.

6. LITERATURA

[1] Gibbs, J., 2018. Prometheus: Games User Research Tool Development Using Best Practices.

[2] Ebert, C., Gallardo, G., Hernantes, J. and Serrano, N., 2016. DevOps

[3] Erich, F., Amrit, C. and Daneva, M., 2014. Report: Devops literature review. University of Twente, Tech. Rep.

[4] Dale, R., 2021. GPT-3: What's it good for?. Natural Language Engineering, 27(1), pp.113-118.

[5] Zhang, M. and Li, J., 2021. A commentary of GPT-3 in MIT Technology Review 2021. Fundamental Research, 1(6), pp.831-833.

Kratka biografija:

Miloš Popović rođen je 25.01.2000. godine u Šapcu. 2018. godine upisuje osnovne studije na Fakultetu tehničkih nauka u Novom Sadu na smeru Softversko inženjerstvo i informacione tehnologije. 2022. godine je diplomirao i potom upisao master studije na istom fakultetu.