Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.1:766
DOI: https://doi.org/10.24867/23BE15Blesic

SERVIS ZA DETEKCIJU PLAGIJARIZAMA U NAUČNIM RADOVIMA

SERVICE FOR DETECTION OF PLAGIARISM IN SCIENCE WORKS

Nikola Blesić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je predstavljena arhitektura i implementacija servisa za detekciju plagijata u naučnim radovima. Servis upoređuje sumnjiv dokument sa referentnom kolekcijom koja predstavlja skup dokumenata za koje se pretpostavlja da su originali. Takođe, u radu je predstavljeno prethodno istraživanje koje se odnosi na višeslojnu veb aplikaciju za prijavu i evidenciju radova, kao i potrebne izmene kako bi se izvršila integracija ova dva servisa. Na kraju rada predstavljeni su rezultati rada servisa.*

Ključne reči:*Elasticsearch, plagijarizam, naučni radovi*

Abstract – *This paper presents the architecture and implementation of the service for plagiarism detection in scientific papers. The service compares a suspect document with a reference collection which represents a set of documents that are assumed to be originals. Also, the paper presents previous research related to n-tier web application for thesis submission and assessment, as well as the necessary changes in order to integrate these two services. At the end of the work, the results of the service are presented.*

Keywords:*Elasticsearch, plagiarism, scientific papers*

1. UVOD

Plagijat u visokom obrazovanju bio je čest razlog za uzbunu u institucijama širom sveta u poslednjih nekoliko decenija [1]. Kada se provera plagijata radi ručno, to nije efikasno i veoma je zamorno za bilo kog pojedinca da bira datoteke jednu po jednu i upoređuje ih sa drugim datotekama kako bi utvrdio pojavu plagijata. Iz tog razloga, razvijaju se sistemi za detekciju potencijalnih plagijata koji sa velikom uspešnošću pronalaze podudaranja između dokumenata. Dakle, njihova glavna namena je ušteda vremena akademskom osoblju koje pokušava da otkrije plagijat u studentskom projektu, ali i da smanji prisutnost plagijata unutar institucije [2].

2. TEORIJSKE OSNOVE

Plagijat podrazumeva predstavljanje tuđeg rada ili ideja kao svojih, sa ili bez njihovog pristanka, tako što se ugrađuju u rad bez punog priznanja [3].

Sav objavljeni i neobjavljeni materijal, bilo u rukopisu, štampanom ili elektronskom obliku, obuhvaćen je ovom definicijom. Plagijat može biti nameran ili nepromišljen, ili nenameran [3]. Generalno, tehnike plagijata, po načinu nastanka, možemo klasifikovati na tri grupe: leksička, sintaksna i semantička [4].

Softver za detekciju potencijalnih plagijata, odnosno softver za otkrivanje sličnosti teksta, postao je širokodostupan proizvod, kako u obliku komercijalno dostupnih proizvoda, tako i u obliku otvorenog koda [5].

Softveri za otkrivanje sličnosti teksta implementiraju jedan od dva generička pristupa otkrivanju, jedan je spoljašnji, a drugi unutrašnji [5]. Predstavljeni servis za detekciju potencijalnih plagijata zasnovan je na spoljašnjem pristupu.

Spoljašnji pristup za otkrivanje upoređuje sumnjiv dokument sa referentnom kolekcijom koja predstavlja skup dokumenata za koje se pretpostavlja da su originali. Na osnovu izabranog modela dokumenta i unapred definisanih kriterijuma sličnosti, zadatak otkrivanja je da se pronađu svi dokumenti koji sadrže tekst čiji je stepen sličnosti iznad izabranog praga tekstu u sumnjivom dokumentu [5].

Softveri za detekciju plagijata oslanjaju se na oblast pronalaženja informacija. Pronalaženje informacija je oblast koja se bavi tehnikama za reprezentaciju, skladištenje, organizaciju, pristup i pronalaženje informacija. Sistemi za pronalaženje informacija najčešće imaju odvojene procese indeksiranja i pretraživanja [6].

Indeksiranje teksta je korak pretprocesiranja za pretraživanje teksta. Tokom procesa indeksiranja teksta, informacije se prikupljaju, parsiraju (raščlanjuju) i čuvaju, kako bi se omogućila brza i tačna pretraga [7].

Pretraživanje teksta definiše se kao podudaranje nekih navedenih korisničkih upita prema skupu tekstova. Kao rezultat pretrage teksta, tekstovi se rangiraju i prezentuju korisniku prema njihovoj relevantnosti u odnosu na upit (tj. informacionu potrebu) korisnika [7].

Pretprocesiranje igra važnu ulogu u pronalaženju informacija i predstavlja primenu određenih pravila na tekstualne sadržaje koji obezbeđuju fleksibilnost pretrage. Isto pretprocesiranje teksta se primenjuje na tekstove u dokumentima prilikom indeksiranja, kao i na reči u upitu prilikom pretraživanja. [6]

Postoje različite tehnike pretprocesiranja tekstualnih dokumenata. To su: pretvaranje velikih u mala slova, normalizacija, lematizacija, stemovanje, uklanjanje stop reči, uklanjanje akcenta, filtracija, itd. [6].

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivanović, red.prof.

3. TEORIJSKE OSNOVE

Java je objektno orijentisani programski jezik visokog nivoa, zasnovan na klasama, koji je dizajniran da ima što manje zavisnosti od implementacije. To je programski jezik opšte namene, namenjen da dozvoli programerima da program pišu jednom i pokreću ga bilo gde, što znači da kompajlirani Java kod može da radi na svim platformama koje podržavaju Javu, bez potrebe za ponovnim kompajliranjem. Java aplikacije se obično kompajliraju u bajt kod koji može da radi na bilo kojoj Java virtuelnoj mašini (JVM) bez obzira na arhitekturu računara [8].

Spring je najpopularniji okvir za razvoj aplikacija za Java platformu. Spring okvir je otvorenog koda. Spring je lagan (eng. *lightweight*), kada je u pitanju veličina i transparentnost. Osnovna verzija Spring okvira je oko 2MB. Osnovne karakteristike okvira može da koristi bilo koja Java aplikacija, ali postoje proširenja za pravljenje veb aplikacija na vrhu Java EE (Enterprise Edition) platforme [9].

Elasticsearch je distribuiran, besplatan i otvoren server za pretragu i analizu svih tipova podataka, uključujući tekstualne, numeričke, geoprostorne, strukturirane i nestrukturirane podatke. Napisan je u Java programskom jeziku što omogućava pokretanje na svim platformama. Baziran je na *Lucine* indeksima i omogućava korisnicima da pretraže veliku količinu podataka vrlo brzo.

Može se koristiti i za čuvanje podataka, ali je njegova glavna uloga indeksiranje i pretraga podataka u realnom vremenu. Poznat je po svom jednostavnom REST API-ju, distribuiranoj prirodi, brzini i skalabilnosti [10, 11].

4. SPECIFIKACIJA

Servis za detekciju plagijata je veb servis čija je glavna namena provera prisutnosti plagijata u naučnim radovima, a pored toga on omogućava proveru sličnosti teme, prilikom njenog unosa sa već postojećim temama. Korisnički zahtevi koje sistem za detekciju plagijata treba da ispuni su:

- unos nove teme,
- provera sličnosti teme u odnosu na već postojeće teme,
- unos novog rada i
- provera plagijarizama.

Ideja rada je da prikaže primenu ovog servisa na već postojećoj veb aplikaciji, koja je predstavljena u okviru diplomskog rada „Višeslojna veb aplikacija za prijavu i evidenciju diplomskih radova”. Shodno tome, u ovom radu biće navedene izmene veb aplikacije, odnosno slučajevi krorišćenja na koje se izmene odnose. To su sledeći slučajevi:

- slanje diplomskog rada na proveru kod profesora,
- evaluacija diplomskih radova,
- predlaganje teme,
- odobravanje/odbijanje predloženih tema i
- objavljivanje novih tema.

4.1. Opis arhitekture

Servis za detekciju plagijata je serverska aplikacija koja je napravljena prateći višeslojnu arhitekturu aplikacija i sastoji se iz 4 sloja. Slojevi su sledeći:

- API sloj,
- sloj poslovne logike,
- sloj za pristup bazi podataka i
- baza podataka.

API sloj se sastoji iz programskih interfejsa API-ja koji su napravljeni po principu REST arhitekture. Zahtevi koji se šalju na servis namapiraju se na odgovrajući REST kontroler [12].

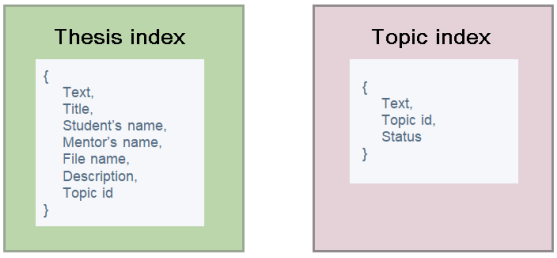
Sloj poslovne logike se sastoji pre svega iz modela. Model čini skup entiteta odnosno klasa koje se mapiraju u tabele baze podataka. Servisi su skup klasa sa zadatkom da izvršavaju neku konkretnu poslovnu logiku, odnosno funkcionalnost koja je srž aplikacije [12].

Sloj za pristup bazi podataka napravljen je po *repository* obrascu i sadrži klase koje izvršavaju upit ili komande za upis i brisanje nad bazom podataka.

Na kraju, tu i baza podataka koja služi za trajno čuvanje podataka koji su od važnosti za rad ovog servisa. Za bazu podataka korišćen je Elasticsearch, koji je NoSQL baza podataka.

4.2. Model podataka

Servis za detekciju plagijata sastoji se iz dva indeksa. Jedan indeks odnosi se na odbranjene radove, dok se drugi odnosi na teme radova. Na ovaj način obezbeđuje se brza i fleksibilna pretraga odbranjenih radova, kao i pretraga tema u bilo kom statusu (slobodna, zauzeta ili odbijena). Na slici 1. prikazani su indeksi sa pripadajućim poljima.



Slika 1. *Indeksi servisa za detekciju plagijata*

Jedna pokrenuta instanca Elasticsearch-a naziva se čvor (eng. *node*). *Shard* predstavlja mesto gde se podaci čuvaju i odakle se upiti izvršavaju. S obzirom da je trenutno količina podataka koju ova aplikacija obrađuje jako mala, reda veličina od nekoliko desetina radova, broj čvorova podešen je na 1. Takođe, broj *shard*-ova po jednom indeksu postavljen je na 1.

Vremenom broj podataka koje aplikacija obrađuje postaće sve veći. Imajući u vidu da broj dokumenata koje *shard* može da sačuva zavisi od kapaciteta čvora, zaključuje se da gore pomenuti jedan čvor i jedan *shard* neće moći da opsluže takve količine podataka. Rešenje ovog problema

je *sharding* i on podrazumeva dodavanje novih *shard*-ova i čvorova.

5. IMPLEMENTACIJA

Dve ključne funkcionalnosti na koje se servis za detekciju plagijata oslanja su: indeksiranje i pretraživanje.

Prilikom dodavanja novog rada vrši se otpremanje dokumenta koji je prosleđen. Potom, potrebno je izvršiti indeksiranje pomenutog dokumenta, tako što se vrši mapiranje parametara na dokument koji se indeksira (eng. *indexing unit*). Na listingu 1. prikazan je isečak koda koji prikazuje način na koji je implementiran *indexing unit*

```
@Document(indexName =
ThesisIndexUnit.INDEX_NAME,
replicas = 0)
public class ThesisIndexUnit {

	public static final String
INDEX_NAME = "thesislibrary";

	@Field(type =
	FieldType.Text, analyzer =
	"serbian", store = true)
	private String text;
	@Field(type =
FieldType.Text, store = true)
	private String title;
	@Field(type =
FieldType.Text, store = true)
	private String studentsName;
	@Field(type =
FieldType.Text, store = true)
	private String mentorsName;
	@Id
	@Field(type =
FieldType.Keyword, store = true)
	private String filename;
	@Field(type =
FieldType.Text, store = true)
	private ThesisStatus
thesisStatus;
	@Field(type =
FieldType.Text, index = false,
store = true)
	private String description;
	@Field(type =
FieldType.Keyword, store = true)
	private String topicId;
	...
}
```

Listing1. *Klasa ThesisIndexUnit*

koji se odnosi na odbranjene radove.

Prilikom kreiranja *indexing unit*-a, obavezno je definisati ime indeksa. Podrazumevani broj *shard*-ova i replika je 1. U ovom slučaju, zbog jednostavnosti projekta, broj replika postavljen je na 0.

Analizator koji je korišćen za indeksiranje i pretragu je *SerbianAnalyzer*. Koristi standardni tokenizator (eng. *standard tokenizer*) kako bi formirao tokene. Nakon tokenizacije, tokeni prolaze kroz sledećih pet filtera:

- *LowerCaseFilter* – transformiše znakove iz tokena u mala slova,
- *LatCyrFilter* – pretvara reči napisane malim slovima iz ćirilice u latinicu,
- *StopFilter* – uklanja stop reči/tokene,
- *SnowballFilter* – vrši stemovanje tokena, uzimajući koren reči i
- *RemoveAccentsFilter* – zamenjuje 'Dj' u 'D' [24].

Nakon što je *indexing unit* kreiran, potrebno je izvršiti njegov upis u *Elasticsearch*. *ElasticsearchRepository* proširuje *Spring* interfejs za repozitorijume tako da se može koristiti kao jedno od njih. Sve što je potrebno uraditi da bismo mogli da indeksiramo objekte u *Elasticsearch* je da na klasu koja se indeksira dodamo anotaciju *@Document* i da kreiramo interfejs repozitorijuma koji proširuje *ElasticsearchRepository*.

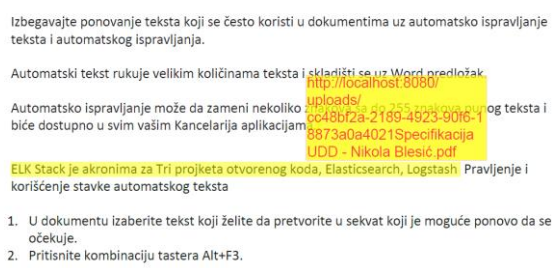
Algoritam provere plagijata, zasnovan je na pretraživanju prethodno indeksiranog sadržaja i podrazumeva sledeće korake:

- Na osnovu prosleđenog upita (napravljenog od dela teksta – maks. 500 termova), izvršiti pretragu u *Elasticsearch*-u i sačuvati dobijene rezultate.
- Prethodno dobijeni rezultati predstavljaju kolekciju potencijalnih izvora plagijata, odnosno delova teksta iz drugih dokumenata odakle je plagijat nastao. Potrebno je izvršiti filtriranje, te iz date kolekcije izdvojiti samo one elemente čija dužina je veća od unapred definisane dužine.
- U ovom trenutku imamo kolekciju delova dokumenata za koje se pretpostavlja da su izvor plagijata i čija dužina ne prelazi definisani minimum. Za svaki od pronađenih izvora potrebno je pronaći plagijat (sadržan u dokumentu koji se ispituje) na koji se dati izvor odnosi. Ovo se izvodi tako što se od svakog pronađenog izvora plagijata kreira upit i vrši se pretraga nad dokumentom koji se ispituje.
- Plagijati, zajedno sa izvorom odakle su nastali, čuvaju se u listu i u daljem procesiranju koriste za obeležavanje plagijata i prikaz rezultata provere plagijata.

Nakon završene provere rada na plagijate, ukoliko oni postoje, potrebno je napraviti izveštaj o njihovom prisustvu. Izveštaj predstavlja PDF dokument koji rad, za koji je vršena provera, sadrži u celosti, pri čemu delove teksta za koje je utvrđeno da su plagijati markira i u komentaru vezanom za markirani deo sadržaja navodi izvor plagijata.

6. PRIKAZ REZULTATA

Kada pristigne zahtev za proveru rada na plagijate, rad se preuzima i servis za detekciju plagijata vrši proveru. U slučaju da je plagijat detektovan, vrši se njegovo obeležavanje i pridružuje se komentar čiji sadržaj predstavlja listu referenci ka dokumentima u odnosu na koje je utvrđeno podudaranje. Na slici 2. prikazan je isečak PDF dokumenta, u kojem je pronađen i markiran potencijalni plagijat.



Slika 2. *Potencijalni plagijat*

6. ZAKLJUČAK

U ovom radu predstavljena je implementacija servisa za detekciju plagijata baziranog na tehnikama za pronalaženje informacija. Objašenjeni su pojmovi koji su neophodni za razumevanje funkcionisanja servisa. Fokus datih teorijskih osnova je razumevanje samog pojma plagijata, kao i mogućih načina na osnovu kojih može biti otkriven. Da bi se pokazala stvarna primena servisa za detekciju plagijata, izvršena je integracija pomenutog servisa sa veb aplikacijom koja je predstavljena u okviru diplomskog rada „Višeslojna veb aplikacija za prijavu i evidenciju diplomskih radova“. U radu su navedeni slučajevi korišćenja za koje potrebno izvršiti izmene, kako bi integacija bila uspešna.

Namena ovog servisa je da na brz i lak način uoči potencijalne plagijate u radovima studenata. Takođe, sistem pored pronađenih potencijalnih plagijata, pronalazi i delove teksta za koje je potrebno proveriti da li su dobro referencirani.

Ono što bi u velikoj meri olakšalo proces pregledanja rada od strane mentora, je automatska provera referenci. Zatim, u trenutnom rešenju, za svaki rad vrši se detekcija plagijata nezavisno u odnosu na prethodne verzije. U tom smislu, postoji prostor za kreiranje mehanizma koji bi prepoznao potencijalne plagijate iz prethodnih radova, a za koje je zaključeno da nisu plagijati i njih ne bi markirao. Takođe, trebalo bi uzeti u obzir da sistem može da greši i da tekst koji nije plagijat proglasi plagijatom i na taj način zaustavi dalji postupak odbrane rada. Dalji razvoj podrazumevao bi nadogradnju predstavljenog rešenja u cilju prevazilaženja navedenih nedostataka.

7. LITERATURA

[1] Kara Ronai, „Plagiarism Defined? A multiple case study analysis of institutional definitions.“, 2020.
[2] Ugo Chidera Chinedu, Dr. Charles Ikerionwu, Engr. Obi Nwokonkwo, „Plagiarism Detection Systems“ , 2021.
[3] Plagiarism, url: https://www.ox.ac.uk/students/academic/guidance/skills/plagiarism [Poslednji pristup: 30.1.2023.]
[4] El Mostafa Hambi, Faouzia Benabbou, „A systems for Ideas Plagiarism Detection: State of art and proposed approach“, 2022.
[5] Content similarity detection, url: https://en.wikipedia.org/wiki/Content_similarity_detection [Poslednji pristup: 30.1.2023.]
[6] Dragan Ivanović, Branko Milosavljević, „Upravljanje digitalnim dokumentima“, 2015.
[7] Haoda Huang, Benyu Zhang, „Text indexing and Retrieval. Encyclopedia of Database Systems“, 2009.
[8] Java (programming language), url: https://en.wikipedia.org/wiki/Java_(programming_language) [Poslednji pristup: 30.1.2023.]
[9] Spring Framework, url: https://en.wikipedia.org/wiki/Spring_Framework [Poslednji pristup: 30.1.2023.]
[10] What is Elasticsearch?, url: https://www.elastic.co/what-is/elasticsearch [Poslednji pristup: 30.1.2023.]
[11] Elasticsearch, url: https://blog.imi.pmf.kg.ac.rs/elasticsearch/ [Poslednji pristup: 30.1.2023.]
[12] Nikola Blesić, „Višeslojna veb aplikacija za prijavu i evidenciju diplomskih radova“, 2021.

Kratka biografija:

Nikola Blesić rođen je u Subotici 1998. godine. 2017. Godine završio je Srednju tehničku školu „Ivan Sarić“ u Subotici, smer Elektrotehničar informacionih tehnologija. Iste godine, upisuje Fakultet Tehničkih Nauka u Novom Sadu, smer Računarstvo i automatika. Osnovne studije završava 2021. godine, odbranom diplomskog rada na temu „Višeslojna veb aplikacija za prijavu i evidenciju diplomskih radova“. Nakon završenih osnovnih studija upisuje master akademske studije na istom fakultetu i završava ih 2023. godine.