KONTROLNA TAČKA 1 (UPRAVLJANJE DIGITALNIM DOKUMENTIMA)

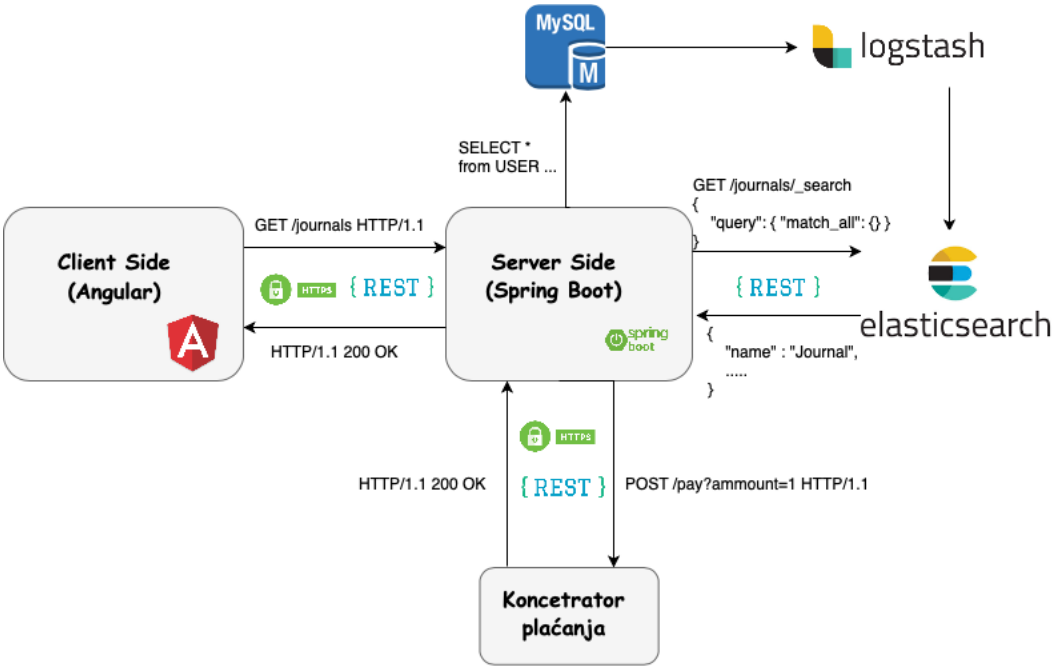
AUTOR RADA: DAVID VULETAŠ , R1 4-2018

1. ARHITEKTURA APLIKACIJE

Aplikacija za rad sa naučnim radovima će se sastojati od klijentske i serverske strane.

Za klijentsku stranu biće korišćen *Angular* framework verzije 7, dok će serverska aplikacija biti izrađena u *Java* framework-u *Spring Boot* 2.1.0.

Što se tiče skladištenja podataka, biće korišćena *MySql* baza podataka, dok će se radi brže i kompleksnije pretrage koristiti *Elasticsearch* baza podataka.



Slika 1 Arhitektura Sistema

2. KOMPONENTE SISTEMA

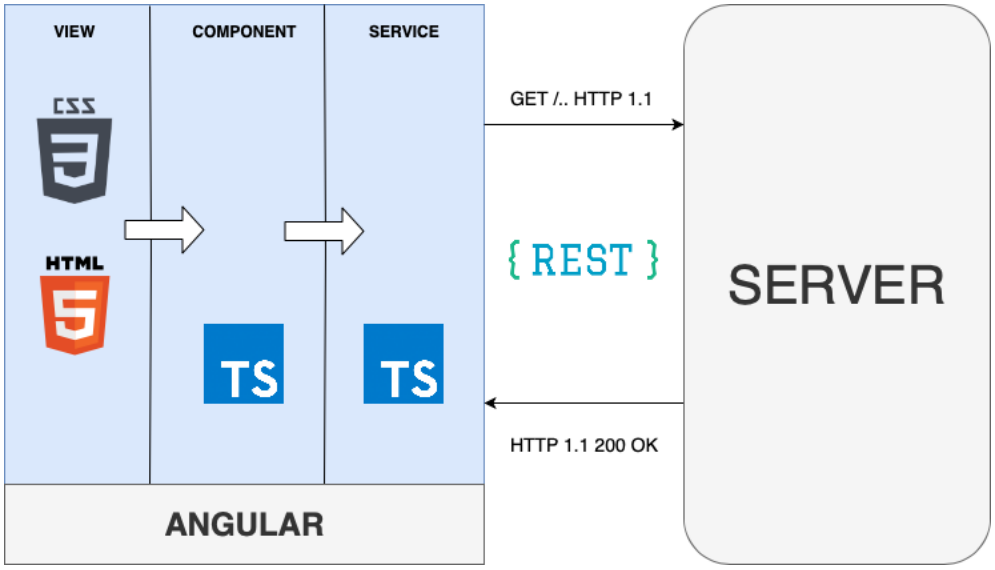
U nastavku biće opisana svaka komponenta sistema ponaosob, od čega se sastoji kao i na koji način komunicira sa povezanim komponentama.

A. Klijentska strana – Angular

Kako bi se korisniku omogućilo da koristi aplikaciju na što jednostavniji način potrebno je da postoji korisnički interfejs odnosno UI. Preko UI-a korisnik ne mora da zna na koji se način vrši komunikacija sa narednim komponentama, nego može da se fokusira na korišćenje jednostavnog korisničkog interfejsa.

Jedna od najrasprostranjenijih framework-a za izradu korisničkog interfejsa radi korišćenja web aplikacija predstavlja Angular. Za izradu klijentske strane aplikacije biće korišćen Angular verzije 7.0.

Komunikacija se odvija prema pattern-u koji koristi pomenuti framework (MVC), bitno je pomenuti dva pojma a to su komponente i servisi. Komponente služe za poslovnu logiku u okviru klijentske strane te preko DI (dependency injection) koristi servis, dok servisi vrše komunikaciju sa serverskom stranom preko RESTful HTTP poziva.



Slika 2 Komunikacija klijentske i serverske aplikacije

B. Serverska strana – Spring Boot

Serverska strana aplikacije predstavlja glavno čvorište sistema, odatle se odlučuje gde će se zadatak dalje nastaviti. Kao što je u prethodnom podpoglavlju pomenuto komunikacija klijentskog dela sa serverom obavlja se preko RESTful web servisa, da bi se razumeo dalji prolaz podataka ka ostalim servisima potrebno je razumeti samu arhitekturu serverske aplikacije. U Spring Boot framework-u najčešće se koristi arhitektura koja se sastoji od slojeva kontrolera, servisa i repozitorijuma. Ovi slojevi komuniciraju međusobno korišćenjem DI (Dependency injection).

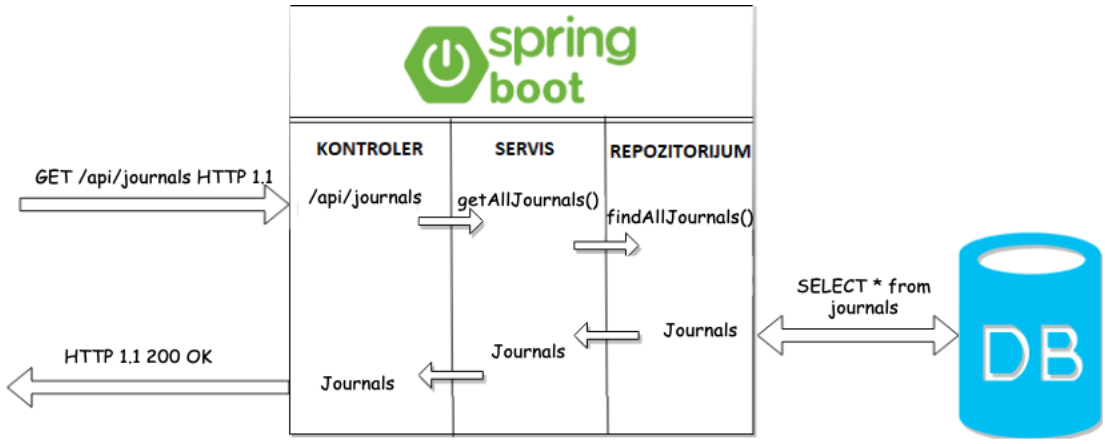
Sloj kontrolera predstavlja sloj koji je preko pristupnih tačaka vidljiv drugim aplikacijama, odnosno na osnovu definisanih endpoint-a neka aplikacija može da poziva server preko RESTful web servisa korišćenjem HTTP-a. Prethodno pomenuti sloj na osnovu podataka koje je dobio ili želi da dobavi preko DI injektuje sloj servisa te mu prosleđuje odgovarajuće parametre i nakon toga se posao nastavlja u sledećem sloju.

Naredni sloj predstavlja sloj servisa, on je zadužen za izvršavanje poslovne logike, odnosno na osnovu podataka dobijenih iz sloja kontrolera vrši pretvaranje u oblik pogodan za kontaktiranje sloja repozitorijuma koji se injektuje u sloj servisa korišćenjem DI.

Najniži sloj aplikacije predstavlja sloj repozitorijuma koji vrši direktno pozive ka bazi podataka. Zahvaljujući objektno relacionom maperu *Hibernate* odnosno njegovoj implementaciji *JPA,* veoma lako je izvršiti neke od osnovnijih upita bez pisanja samog SQL koda.

Nakon dobijenih podataka iz baze vrši se slanje unazad ka nivou servisa koji pretvara podatke u oblik pogodan za nivo kontrolera koji na kraju vraća klijentu podatke u obliku HTTP odgovora

Primer jedne takve komunikacije je prikazan na slici ispod (Slika 3)



Slika 3 Komunikacija slojeva serverske aplikacije

C. MySQL baza podataka

Ova komponenta predstavlja veoma bitan deo sistema zbog toga što omogućuje čuvanje podataka trajno.

Kao što je opisano u poglavlju vezanom za serversku stranu, sloj repozitorijuma je zadužen direktno za rad sa bazom. Server je povezan sa bazom tako što je u konfiguracionom fajlu definisano na koji URL treba da se server poveže.

Takođe podaci iz tabele Journal se na određeni vremenski period kopiraju u Elasticsearch preko Logstash-a što će biti opisano u sledećem poglavlju.

D. Logstash

Glavna namena ovog sistema je da prikuplja, procesira pa i da šalje prethodno obrađene podatke na jednu ili više destinacija. Najčešće je on zadužen za slanje logova sa nekog sistema na Elasticsearch, međutim u našem slučaju će se vršiti slanje podataka sa relacione baze podataka na Elasticsearch isključivo onih radova koji su objavljeni.

Da bi procesiranje podataka sa relacione baze na Elasticsearch funkcionisalo potrebno je da se kreira konfiguracioni fajl u kojem će se podesiti izvorište i odredište. Prema tome moglo bi se reći da je Logstash posrednik između relacione baze i Elasticsearch.

Na listingu (Listing 1) prikazan je izgled konfiguracionog fajla zaduženog za mapiranje.

```
input {
    jdbc {
        jdbc_connection_string =>
               "jdbc:mysql://localhost:3306/
                sc?autoReconnect=true&useSSL=false"
        jdbc_user => "root"
        jdbc_password => ""
        jdbc_driver_library =>
        "..\mysql-connector-java-6.0.6.jar"
        jdbc_driver_class =>
"com.mysql.cj.jdbc.Driver"
        schedule => "* * * * *"
        statement =>
         "SELECT * FROM Journals WHERE pubished=true"
    }
}
output {
    stdout { codec => json_lines }
    elasticsearch {
       hosts => ["localhost:9200"]
       index => "journals"
```

Listing 1 Konfiguracioni fajl zadužen za mapiranje

Prilikom pokretanja Logstash servera potrebno je naglasiti da se prethodno prikazan konfiguracioni fajl postavi za aktivnog.

Nešto više o ovoj ideji je prikazano na linku : http://softwaredevelopercentral.blogspot.com/2017/10/elasticsearch-logstash-kibana-tutorial.html

E. Elasticsearch

Komponenta sistema koja je veoma bitna za izvršavanje brzih i složenih pretraga nad naučnim časopisima sistema. Elasticsearch predstavlja search engine baziran na Lucene biblioteci koji obezbeđuje punu pretragu dokumenata, obezbeđujući korišćenje preko HTTP protokola, kao i skladištenje podataka u JSON formatu (NoSQL).

Može se koristiti za pretraživanje svih vrsta dokumenata, omogućava skalabilnost pretraživanja i koristi pretragu u realnom vremenu.

Najčešće kada se pomene Elasticsearch misli se na ELK stack koji predstavlja Elasticsearch, Logstash i Kibana, ova tri alata omogućuju jednostavnu pretragu uz vizuelizaciju.

Komunikacija Naučnog centra sa Elasticsearch-om ostvaruje se preko RESTful poziva. U trenutku kada korisnik započne pretragu u klijentskoj aplikaciji, vrši se slanje zahteva ka serverskoj strani koja vrši procesiranje pristiglog zahteva te šalje zahtev ka Elasticsearch serveru za izvršavanje upita u cilju pronalaženja određenih podataka.

Da bi komunikacija bila ostvarena potrebno je iskonfigurisati Elasticsearch server, nešto više o tome biće istraženo sa sledeću kontrolnu tačku.

Takođe, da bi se omogućila pretraga na Srpskom jeziku korišćen će biti plugin *SerbianAnalyzer* za Elasticsearch koji se može pronaći na linku: https://github.com/chenejac/udd06

F. Koncentrator plaćanja

Kao servis koji je zadužen za obradu transakcija plaćanja, koristi se koncentrator plaćanja. Veza sa ovom komponentom se ostvaruje kroz RESTful endpoint-e pozivima od Naučnog centra ka njemu. Bitno je naglasiti da se ova komunikacija odvija preko HTTPs zaštićenog protokola. Ova komponenta se smatra za eksterni servis sistema, gde korisnik ne bi trebalo da zna za postojanje ove komponente, već će za svaki oblik plaćanja biti zadužen pomenuti Koncentrator plaćanja. Bitno je naglasiti da je koncentrator kreiran tako da omogućuje plagabilnost.