Projektni zadatak iz predmetā

Servisno orijentisane arhitekture, NoSQL baze podataka i Informaciona bezbednost

Školska 2025/2026. godina

Kroz predmetni projekat treba da implementirate aplikaciju za streaming muzike i upravljanje plej-listama, nalik na popularnu Spotify platformu.

Uloge u sistemu

- **Neautentifikovani korisnik (NK)** - Može da se registruje na sistem. Ako poseduje nalog, može da se prijavi na sistem.
- **Administrator (A)** - Autentifikovani korisnik u ulozi administratora. Ima mogućnost dodavanja i uređivanja muzičkog sadržaja i umetnika.
- **Redovan korisnik (RK)** - Redovan korisnik koji se uspešno prijavio na sistem. Ima mogućnost pretrage i pregledanja postojećeg sadržaja. Može da oceni sadržaj i da se pretplati na određeni sadržaj, kao i da rukovodi postojećim pretplatama. Dobija notifikacije za sadržaj na koji se pretplatio. Ima personalizovane preporuke na osnovu prethodnih interakcija sa sistemom.

Komponente sistema

- **Klijentska aplikacija** - Pruža grafički interfejs preko kog korisnici pristupaju funkcionalnostima sistema.
- **Serverska aplikacija** - Mikroservisna aplikacija koja sadrži čitavu poslovnu logiku sistema. Sastoji se iz sledećih servisa:
  - **Users** - Čuva kredencijale korisnika i njihove uloge u sistemu. Zadužen za proces registracije i prijave korisnika. Dodatno čuva osnovne informacije o korisniku poput imena i prezimena. Podatke čuva u dokument-orijentisanoj bazi podataka.
  - **Content** - Čuva žanrove, umetnike (ime, žanrovi, opis), albume (naziv, datum, žanr, umetnici) i pesme (naziv, dužina, žanr, album, umetnici…). Podatke čuva u dokument-orijantisanoj bazi podataka.
  - **Ratings** - Čuva informacije o ocenama koje su korisnici dali pesmama. Podatke čuva u NoSQL bazi po izboru.
  - **Subscriptions** - Čuva informacije o pretplatama korisnika. Podatke čuva u NoSQL bazi po izboru.
  - **Notifications** - Upravlja skladištenjem i slanjem notifikacija korisnicima. Podatke čuva u wide-column bazi podataka.

- **Recommendation** - Zadužen je za generisanje personalizovanog feed-a na osnovu istorije, ocena, žanrova, popularnosti i grafa sličnosti. Koristi graf bazu podataka.
- **Analytics -** Čuva informacije o istoriji aktivnosti korisnika i služi za računanje analiza i analitika vezanih za muzički sadržaj.

1. Funkcionalni zahtevi

1.1 Registracija naloga (uloge: NK)

Svaki nalog mora imati jedinstven username. Minimalan skup ličnih podataka koje korisnik mora uneti pri registraciji je: ime, prezime, imejl adresa, lozinka i ponovljena lozinka. Nametnuti korisniku upotrebu jake lozinke, kao i periodičnu promenu lozinke. Implementirati potvrdu registracije.

1.2 Prijava na sistem i odjava sa sistema (uloge: NK)

Implementirati kombinovanu autentifikaciju upotrebom korisničke lozinke i jednokratne lozinke koja se šalje na imejl adresu (OTPS). Omogućiti korisniku da promeni lozinku. Da bi se lozinka promenila, mora biti bar 1 dan stara. U svrhu promene lozinke implementirati imejl-bazirani reset lozinke sa poznatom imejl adresom i kratkotrajnim linkom.

Potrebno je demonstrirati auditabilnost korisničkog naloga onemogućavanjem prijave na sistem na određeno vreme nakon isteka roka važenja lozinke. Maksimalni period važenja aktivne lozinke je 60 dana (na odbrani simulirati periode za izmenu na kraće).

1.3 Povraćaj naloga

Implementirati autentifikaciju upotrebom magičnog linka.

1.4 Kreiranje i izmena umetnika (uloge: A)

Potrebno je omogućiti unos osnovnih informacija o umetniku, minimalno uključujući ime, biografiju i žanrove. Za postojeće umetnike moguće je menjati sve ove informacije.

1.5 Kreiranje albuma i pesama (uloge: A)

Administrator može da doda muzički sadržaj u vidu albuma i pesama. Nije moguće dodati pesmu ako album kojem ona pripada već nije dodat u sistem.

1.6 Pregled umetnika, albuma i pesama (uloge: A, RK)

Na početnoj stranici, korisniku se prikazuje lista svih umetnika u sistemu. Odabirom umetnika, korisnik se prebacuje na stranicu tog umetnika, koja sadrži osnovne

informacije i listu svih albuma tog umetnika. Odabirom albuma, prikazuje se stranica koja sadrži listu svih pesama tog albuma.

1.7 Reprodukcija pesme (uloge: RK)

Korisnik iz pretraživača može da preslušava pesme.

1.8 Filtriranje i pretraga umetnika i muzičkog sadržaja (uloge: A, RK)

Korisnik može da filtrira umetnike po žanru. Pored toga, može da pretražuje umetnike, albume i pesme po nazivu.

1.9 Ocenjivanje pesama (uloge: RK)

Korisnik može da oceni pesmu sa ocenom od 1 do 5. Postojeću ocenu može izmeniti ili ukloniti.

1.10 Pretplata na sadržaj (uloge: RK)

Korisnik može da se pretplati na sadržaj određenog umetnika ili žanra. Na svom profilu može da vidi na sadržaj kojh umetnika i žanrova se pretplatio, kao i da prekine pretplatu ukoliko više nije zainteresovan za taj sadržaj.

1.11 Generisanje notifikacija (uloge: RK)

Član treba da dobije notifikaciju:

- kada se doda novi album umetnika na čiji sadržaj je pretplaćen;
- kada se doda nova pesma umetnika na čiji sadržaj je pretplaćen;
- kada se doda novi umetnik žanra na koji je pretplaćen.

1.12 Pregled notifikacija (uloge: RK)

Svaki korisnik može da vidi sve notifikacije koje je dobio na svom profilu. Za ocenu 6 dovoljno je “ručno” popuniti bazu podacima i omogućiti endpoint koji ih dobavlja. Prikazati dobavljene notifikacije u klijentskoj aplikaciji.

1.13 Preporuke muzičkog sadržaja (uloge: RK)

Na početnoj stranici, umesto liste svih umetnika, prikazuju se preporuke muzičkog sadržaja na osnovu interakcije korisnika sa sistemom. U graf bazi se čuvaju informacije o korisnicima, umetnicima, pesmama, žanrovima, a veze između čvorova su: ocena koju je korisnik dodelio pesmi, pesma pripada žanru, korisnik se pretplatio na žanr... Potrebno je prikazati na početnoj stranici pesme koje pripadaju žanrovima na koje je pretplaćen korisnik, a korisnik im nije dodelio ocenu manju od 4 (dodelio im je ocenu bar 4 ili ih nije ocenio uopšte). Takođe, prikazati pesmu koja pripada žanru na koji korisnik

nije pretplaćen, a ima najviše ocena 5 dodeljenih od strane drugih korisnika. Ukoliko želite proširiti algoritam preporuke koji je ovde opisan, imate slobodu da to uradite.

1.14 Brisanje pesama (uloge: A)

Administrator može da obriše pesmu. Kada se to desi, potrebno je obrisati sve ocene i preporuke vezane za obrisani sadržaj.

1.15 Istorija aktivnosti korisnika (uloge: RK)

Redovan korisnik može da vidi istoriju svojih aktivnosti. U aktivnosti se ubraja slušanje pesme, kreiranje i uklanjanje pretplate na žanr ili umetnika, kao i davanje ocene pesmi.

1.16 Prikaz i računanje analitika (uloge: RK)

Svaki korisnik može da vidi sledeće analitike vezane za muzički sadržaj:

- broj odslušanih pesama,
- prosek svih ocena koje je dodelio pesmama,
- broj odslušanih pesama po svakom žanru,
- 5 umetnika koje je korisnik najviše slušao,
- broj umetnika na čiji sadržaj je pretplaćen.

2. Nefunkcionalni zahtevi

2.1 Dizajn sistema

Za svaki servis treba definisati model podataka, odnosno entitete sa kojima servis radi. Za svaki entitet potrebno je navesti atribute koji ga opisuju. Pored toga, potrebno je definisati stilove komunikacije između servisa.

2.2 API gateway

Predstavlja ulaznu tačku u sistem i sva komunikacija između serverske i klijentske aplikacije obavlja se putem nje. API gateway klijentima nudi REST API za komunikaciju.

2.3 Kontejnerizacija

Sve mikroservise, API Gateway i baze podataka potrebno je pokrenuti kao Docker kontejnere i koristiti Docker Compose alat.

2.4 Eksterna konfiguracija

Sva konfiguracija (port na kome se server pokreće, adrese do drugih servisa i baze podataka…) ne treba da bude zakucana u kod. Umesto toga, potrebno je ove informacije izdvojiti u konfiguracione fajlove ili environment promenljive.

2.5 Sinhrona komunikacija između servisa

Prilikom ocenjivanja pesme i kreiranja pretplate na umetnika, potrebno je proveriti da li taj umetnik/pesma postoje. To radite sinhronom komunikacijom između odgovarajućih servisa. Pored ovoga, primenite sinhronu komunikaciju između servisa na svim mestima gde je to adekvatno.

2.6 Asinhrona komunikacija između servisa

Graf u servisu za preporuke ažurira se tako što reguje na događaje koje emituju ostali servisi (primeri događaja: umetnik/korisnik/pesma/pretplata/ocena kreiran/obrisan). Takođe, na događaj o novom umetniku/pesmi reaguje i servis za pretplate koji treba da odluči koje korisnike treba obavestiti o tome. Pored ovoga, primenite asinhronu komunikaciju između servisa na svim mestima gde je to adekvatno.

2.7 Otpornost na parcijalne otkaze sistema

U slučaju da neki servis trenutno nije dostupan, svi ostali servisi treba da nastave da rade normalno i podrže funkcionalnosti koje nisu zavisne od navedenog servisa. Potrebno je implementirati sledećih sedam mehanizama koji se rade na vežbama.

2.7.1. Konfiguracija http klijenta

2.7.2 Postavljanje timeout-a na nivou zahteva kojim pozivamo drugi servis

2.7.3 Fallback logika kada servis koji pozivamo ne vrati odgovor

2.7.4 Circuit breaker

2.7.5 Retry mehanizam

2.7.6. Eksplicitno postavljen timeout za vracanje odgovora korisniku

2.7.7. Upstream servis odustaje od obrade zahteva ako je istekao timeout

2.8 API composition

Prilikom prikaza pesme, uz svaku pesmu je potrebno prikazati i broj ocena, kao i prosečnu ocenu. Ovu funkcionalnost treba uraditi upotrebom API composition šablona

2.9 CQRS

Kada korisnik pregleda svoja pretplate na umetnike, potrebno je uz pretplatu prikazati i ime umetnika. Ime umetnika ne treba svaki put dobavljati iz Content servisa, već upotrebiti CQRS šablon.

2.10 Tracing

Pomoću Jeager alata implementirati tracing u celoj mikroservisnoj aplikaciji, kako za sinhrone, tako i za asinhrone operacije.

2.11 HDFS

Audio zapise pesama potrebno je čuvati na HDFS-u.

2.12 Keširanje

Najslušanije audio zapise potrebno je keširati. Keš treba implementirati upotrebom Redis baze podataka.

2.13 Saga

Funkcionalnost brisanja pesama treba da bude implementirana preko saga šablona. Na odbrani je potrebno demonstrirati uspešan tok, ali i sve verzije neuspešnih tokova.

2.14 Event sourcing

Iskoristiti šablon event sourcing za implementaciju aktivnosti korisnika. Sve aktivnosti su predstavljene kao događaji. U aktivnosti se ubraja slušanje pesme, kreiranje ili uklanjanje pretplate na žanr i umetnika, kao i davanje ocene pesmi.

2.15 Event sourcing + CQRS

U kombinaciji sa zahtevom 2.11, primeniti CQRS šablon za implementaciju funkcionalnosti 1.17 (analitike).

2.16 Kubernetes

Sve komponente sistema treba pokrenuti unutar Kubernetes klastera.

2.17 Kontrola pristupa funkcionalnostima

Implementirati autorizaciju, koja se proverava za svaki zahtev (zahteve serverskih skripta i zahteve klijentske tehnologije). Ukoliko se *state* podaci

skladište na klijentskoj strani, potrebno je obezbediti njihovo šifrovanje i proveru integriteta.

Potrebno je demonstrirati zaštitu od uskraćivanja usluge (*DoS*) ograničavanjem broja transakcija koje jedan korisnik sme da izvrši.

2.18 Validacija podataka

Potrebno je implementirati tehnike validacije ulaznih podataka (*input validation*), kao što su različite provere stringova, *whitelisting, boundary checking, character escaping, numeric validation,* i provere specijalnih karaktera.

Implementirati bezbedno upravljanje datotekama koje korisnik unosi u sistem. Pod tim se podrazumevaju provera privilegija korisnika i provera tipa datoteke (*whitelisting*). Omogućiti proveru integriteta datoteka.

Validacija ulaznih podataka treba da postoji i na klijentskoj i na serverskoj strani.

Implementirati enkodovanje izlaza, kao tehniku koja uz validaciju ulaznih podataka predstavlja zaštitu od napada XSS i SQL Injection.

2.19 Zaštita podataka

Implementirati HTTPS protokol u komunikaciji između servisa i između API gejtveja i klijentske aplikacije. Voditi računa o HTTP metodu pri prenosu senzitivinih parametara.

Lozinke skladištiti u heširanom formatu i pri tome voditi računa o odabiru heš funkcije (hash&salt mehanizma).

2.20 Logovanje

Omogućiti logovanje sledećih događaja: neuspehe validacije ulaznih podataka, pokušaje prijave na sistem (uspešne i neuspešne), neuspehe kontrole pristupa, neočekivane promene *state* podataka, pokušaje pristupa sa nevalidnim ili isteklim tokenima sesije, administratorske aktivnosti i neuspešne bekend TLS konekcije.

Voditi računa o memorijskom zauzeću log-datoteka i implementirati mehanizam za rotaciju logova.

Log-datoteke treba zaštititi od neovlašćenog pristupa i obezbediti njihov integritet.

Voditi računa da logovi ne uključe osetljive podatke ili *stack trace* i *debugging* informacije.

2.21 Analiza ranjivosti

Napisati izveštaj o nivou bezbednosti aplikacije: koji alati su korišćeni za identifikaciju ranjivosti, koje ranjivosti su identifikovane i kako se one mogu potencijalno eksploatisati, kako prevazići identifikovane ranjivosti i kako se zaštititi od eksploatacije istih.

2.22 Demonstracija pokušaja napada

Pripremiti mehanizme za realizaciju napada XSS, SQL Injection, Brute-force attack i DoS i demonstrirati ih na odbrani projektnog rešenja.

Ocenjivanje na predmetima Servisno orijentisane arhitekture i NoSQL baze podataka

Za ocenu **6** treba implementirati zahteve

- 1.1 Registracija naloga
- 1.2 Prijava na sistem
- 1.3 Kreiranje i izmena umetnika
- 1.4 Kreiranje albuma i pesama
- 1.5 Pregled umetnika, albuma i pesama
- 1.11 Pregled notifikacija
- 2.1 Dizajn sistema
- 2.2 API gateway
- 2.3 Kontejnerizacija
- 2.4 Eksterna konfiguracija

Za ocenu **7** treba implementirati

- Sve navedeno za ocenu 6
- 1.6 Reprodukcija pesme
- 1.7 Filtriranje i pretraga umetnika i muzičkog sadržaja
- 1.8 Ocenjivanje pesama
- 1.9 Kreiranje pretplate na umetnika i žanrove
- 2.5 Sinhrona komunikacija između servisa
- 2.7 Otpornost na parcijalne otkaze sistema (2.7.1, 2.7.2, 2.7.3, 2.7.4)

Za ocenu **8** treba implementirati

- Sve navedeno za ocenu 7
- 1.10 Generisanje notifikacija
- 1.12 Preporuke muzičkog sadržaja
- 2.6 Asinhrona komunikacija između servisa
- 2.7 Otpornost na parcijalne otkaze sistema (2.7.5, 2.7.6, 2.7.7)

Za ocenu **9** treba implementirati

- Sve navedeno za ocenu 8
- 2.8 API composition
- 2.9 CQRS
- 2.10 Tracing
- 2.11 HDFS

Za ocenu **10** treba implementirati

- Sve navedeno za ocenu 9
- 1.13 Brisanje pesama
- 1.14 Istorija aktivnosti korisnika
- 2.12 Keširanje
- 2.13 Saga
- 2.14 Event sourcing

Za ocenu **10+** treba implementirati

- Sve navedeno za ocenu 10
- 1.15 Prikaz i računanje analitika
- 2.15 Event sourcing + CQRS
- 2.16 Kubernetes

Napomena: Kako biste za određeni zahtev ostvarili bodove, članovi tima moraju pokazati razumevanje koncepata vezanih za zahtev i jasno znati da objasne svoju implementaciju.

Ocenjivanje na predmetu Informaciona bezbednost

Ocena 6
– 1.1
– 1.2
– 1.3
– 2.17
– 2.18
– 2.19 bez HTTPS-a između servisa

Ocena 7
– sve za 6
– 2.19 sa HTTPS-om između servisa

Ocena 8
– sve za 7
– 2.20

Ocena 9
– sve za 8
– 2.21

Ocena 10
– sve za 9
– 2.22

Pravila polaganja

- Projekat se radi u timovima od po 4 člana. Članovi tima ne moraju slušati vežbe u istom terminu, sve dok mogu da se organizuju da na vežbe iz MRS predmeta dolaze u istom terminu.
- Za implementaciju serverske i klijentske aplikacije možete koristiti programske jezike i radne okvire po želji. Ukoliko odaberete tehnologije koje se razlikuju od

onih koje su pokrivene na vežbama, pomoć koju asistenti mogu pružiti pri rešavanju problema je ograničena.

- Klijentska aplikacija služi da demonstrirate rad sistema i ne ocenjuje se.
- Ako implementirate zahteve navedene za ocenu 10+, oslobođeni ste polaganja teorijskog dela ispita na SOA i NoSQL predmetima.
- U toku semestra održaće se kontrolna tačka za koju je neophodno implementirati funkcionalnosti navedene za ocenu 6. Odrađene funkcionalnosti neophodno je demonstrirati kroz klijentsku aplikaciju (nije dovoljna upotreba alata poput Postman-a ili cURL-a).
- Ko izađe na kontrolnu tačku i bude zadovoljan ocenom, ne mora da dolazi na finalnu odbranu, koja će se održati krajem februara.
- Ako ne izađete na kontrolnu tačku, **ocena vam se smanjuje za jednu na finalnoj odbrani** (na primer morate odraditi funkcionalnosti za ocenu 10 kako biste dobili ocenu 9).
- U septembru će biti održan još jedan termin finalne odbrane, na kom vam se **ocena smanjuje za jednu**.