

# Oblikovanje programske potpore

Ak. god. 2019./2020.

## Giger

Dokumentacija, Rev. 2

Grupa: *ChillCrew*

Voditelj: *Ivan Juren*

Datum predaje: *16. siječnja 2020.*

Nastavnik: *Tomislav Jukić*

# Sadržaj

<b>1</b>	<b>Dnevnik promjena dokumentacije</b>	<b>3</b>
<b>2</b>	<b>Opis projektnog zadatka</b>	<b>5</b>
2.1	Tijek i opseg aplikacije . . . . .	5
2.1.1	Opcionalna proširenja aplikacije . . . . .	6
2.2	Slična rješenja problema . . . . .	7
2.2.1	Amy . . . . .	7
2.2.2	BandFriend . . . . .	7
<b>3</b>	<b>Specifikacija programske potpore</b>	<b>9</b>
3.1	Funkcionalni zahtjevi . . . . .	9
3.1.1	Obrasci uporabe . . . . .	11
3.1.2	Sekvencijski dijagrami . . . . .	26
3.2	Ostali zahtjevi . . . . .	30
<b>4</b>	<b>Arhitektura i dizajn sustava</b>	<b>32</b>
4.1	Baza podataka . . . . .	34
4.1.1	Opis tablica . . . . .	35
4.1.2	Dijagram baze podataka . . . . .	44
4.2	Dijagram razreda . . . . .	44
4.3	Dijagram stanja . . . . .	54
4.4	Dijagram aktivnosti . . . . .	55
4.5	Dijagram komponenti . . . . .	57
<b>5</b>	<b>Implementacija i korisničko sučelje</b>	<b>58</b>
5.1	Korištene tehnologije i alati . . . . .	58
5.2	Ispitivanje programskog rješenja . . . . .	60
5.2.1	Ispitivanje komponenti . . . . .	60
5.2.2	Ispitivanje sustava . . . . .	62
5.3	Dijagram razmještaja . . . . .	72
5.4	Upute za puštanje u pogon . . . . .	73

<b>6 Zaključak i budući rad</b>	<b>74</b>
<b>Popis literature</b>	<b>76</b>
<b>Indeks slika i dijagrama</b>	<b>78</b>
<b>Dodatak: Prikaz aktivnosti grupe</b>	<b>79</b>

# 1. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak, modificirana glavna .tex datoteka	Lanča	26.10.2019.
0.2	Unesen dnevnik sastanaka	Lanča	27.10.2019.
0.3	Dodani <i>Use Case</i> dijagrami br. 2, 3, 4, 5, 6	Jurić	29.10.2019.
0.4	Dodan <i>Use Case</i> dijagram br. 9	Nosil	29.10.2019.
0.5	Dodani <i>Use Case</i> dijagrami br. 10, 11, 12, 13, 17, 18	Gaši	29.10.2019.
0.6	Dodani <i>Use Case</i> dijagrami br. 1, 7, 8, 14, 15, 16, 19, 20	Zec	30.10.2019.
0.7	Dodani funkcionalni zahtjevi	Zec	04.11.2019.
0.8	Nadopunjen zapisnik sastanka, dodani ostali zahtjevi	Lanča	04.11.2019.
0.9	Dodan opis projektnog zadatka	Gaši	05.11.2019.
0.10	Dodani <i>Use Case</i> dijagrami br. 21-35	Zec	07.11.2019.
0.11	Rastavljeni neki <i>Use Case</i> -ovi na više njih	Gaši	08.11.2019.
0.12	Dodane opisne tablice baze	Gaši	08.11.2019.
0.13	Uneseni sekvencijski dijagrami njihov opis	Jurić, Gaši	12.11.2019.
0.14	Unesen opis arhitekture	Lanča	12.11.2019.
0.15	Unesen opis tablica	Krmek	12.11.2019.
0.16	Dodani dijagrami obrazaca uporabe	Zec	12.11.2019.
0.17	Promijenjeni opisi aktora i use case 1 do 15	Nosil	12.11.2019.
0.18	Dodana slika u opis arhitekture te dodan zahtjev	Lanča	12.11.2019.
0.19	Promjenjeni Use case-ovi 16 do 36	Nosil	12.11.2019.
0.20	Izbrisan duplikat arhitekture, osvježen dnevnik sastajanja	Lanča	14.11.2019.
0.21	Popunjena tablica aktivnosti	Lanča	14.11.2019.
0.22	Dodan odjeljak o deploymentu	Juren	14.11.2019.
0.23	Uklonjeni suvišni dijelovi predloška	Juren	14.11.2019.

Rev.	Opis promjene/dodatka	Autori	Datum
0.24	Dodani DTO i Controllers dijagrami	Zec	14.11.2019.
0.25	Dodani ostali opisi tablica baze podataka	Krmek	15.11.2019.
0.26	Lektorirana dokumentacija	Lanča	15.11.2019.
0.27	Dodani dijagrami Service	Zec	15.11.2019.
0.28	Dodani dijagrami razreda entiteta	Zec	15.11.2019.
0.29	Dodani security dijagram	Zec	15.11.2019.
1.0	Verzija samo s bitnim dijelovima za 1. ciklus	Gaši	15.11.2019.
1.01	Dijagram kontrolera, servisa i repozitorija	Zec	04.01.2019.
1.02	Dijagram securitya, entiteta, dto i errors	Zec	05.01.2019.
1.03	Dijagram razmještaja, opisi dijagrama razmještaja, razreda errors i repository	Zec	12.01.2019.
1.04	Dodan dijagram aktivnosti i njegov opis	Gaši	13.01.2019.
1.05	Dodano ispitivanje komponenti	Juren	13.01.2019.
1.06	Dodan zaključak, prepravljen datum predaje	Lanča	16.01.2019.
1.07	Dodan dijagram komponenti	Zec	16.01.2019.
1.08	Dodana nova verzija slike baze	Gaši	16.01.2019.
1.09	Popravljene tablice baze	Gaši	16.01.2019.
1.10	Popravljeni opisi tablica baze	Gaši	16.01.2019.
1.11	Dodan dnevnik sastanaka	Lanča	16.01.2019.
1.12	Popunjena tablica aktivnosti	Zec	16.01.2019.
2.0	Dodan dnevnik pregleda promjena	Gaši	16.01.2019.

## 2. Opis projektnog zadatka

Cilj ovog projekta je razviti programsku podršku za stvaranje web aplikacije *Giger* koja je namijenjena glazbenicima, ljudima kojima su potrebne usluge glazbenika (organizatori različitih proslava, menadžeri) te ljudima koji su zainteresirani za obližnje događaje. Na taj način na jednom mjestu se omogućava:

- upoznavanje glazbenika i njihovo povezivanje u bendove
- olakšana komunikacija i organizacija unutar bendova
- promocija glazbenika
- dogovaranje nastupa s organizatorima koji jednostavno mogu pronaći prikladan bend za određeni događaj
- pregled zanimljivih nadolazećih događaja dostupan svima
- uvid u recenzije bendova/organizatora

Organizacija osobnog kalendara i dogovaranje termina koji zahtijevaju prisustvo više ljudi je težak zadatak, a s tim se problemom na gotovo dnevnoj razini susreću glazbenici kada dogovaraju nastupe. Isto tako, ljudi koji organiziraju razne proslave za koje trebaju glazbenike, kao i ljudi koji žele otići na neku živu svirku, ponekad ne znaju kakva je glazbena ponuda u njihovoj okolini, a ako su i čuli za neki događaj u blizini, ne postoji jedinstveno mjesto gdje mogu pročitati recenzije o glazbenicima, bendu ili organizatoru kako bi bili sigurni u kvalitetu događaja. *Giger* je platforma koja rješava navedene probleme integrirajući kalendar glazbenika sa servisom namijenjenim za sastajanje bendova i organizatora nastupa.

Česta je situacija da je jedan glazbenik član više bendova pa tako dostupnost svakog benda ovisi o dostupnosti svih njegovih članova. Kalendar svakog glazbenika ujedinjuje sve obaveze iz bendova u kojima je član pa dostupnost benda postaje trivijalna informacija.

### 2.1 Tijek i opseg aplikacije

Prilikom pokretanja aplikacije, neregistriranom korisniku prikazuju se opće informacije o javnim događajima te mu se nudi mogućnost prijavljivanja u sustav s

postojećim računom (potrebno upisati email i lozinku) te kreiranje novog računa. Za stvaranje novog računa potrebni su:

- korisničko ime
- email adresa
- lozinka

Registrirani korisnik može pregledati, mijenjati osobne podatke i izbrisati svoj korisnički račun. Takvom korisniku nudi se mogućnost pisanja recenzija te razmjenjivanja poruka s drugim korisnicima. On u svojim postavkama može postati glazbenik i/ili organizator.

Glazbenik može odabrati instrumente koje svira, osnovati bend, pridružiti se postojećem, dodati članove u bend te uređivati svoj profil i kalendar. On na svojoj stranici može dodavati različite medije te se tako promovirati.

Organizator ima mogućnost kreiranja nastupa. On može filtrirati bendove prema vrsti glazbe, tipu nastupa, lokaciji i drugo. Nakon što dobije listu raspoloživih bendova, može pregledavati njihove profile. Profil benda prikazuje osnovne informacije o bendu, razne novosti koje uređuju njegovi članovi, popis nadolazećih javnih nastupa te recenzije korisnika. Organizator ima mogućnost kontaktiranja benda putem poruka ugrađenih u aplikaciju. Imajući takav skup podataka, *Giger* svojim korisnicima, običnim ljudima željnim zabave, može preporučiti nadolazeće događaje u njihovoj blizini.

Uz glazbenika i organizatora postoji i uloga administratora koji ima mogućnost uređivanja popisa ponuđenih instrumenata te blokiranja korisnika.

Aplikacija će biti izvedena kao web aplikacija prilagođena (engl. responsive) mobilnom uređaju i podržavat će rad više paralelnih korisnika sa sučeljem koji je jednostavan za korištenje kako bi korisnici imali što bolje iskustvo.

### 2.1.1 Opcionalna proširenja aplikacije

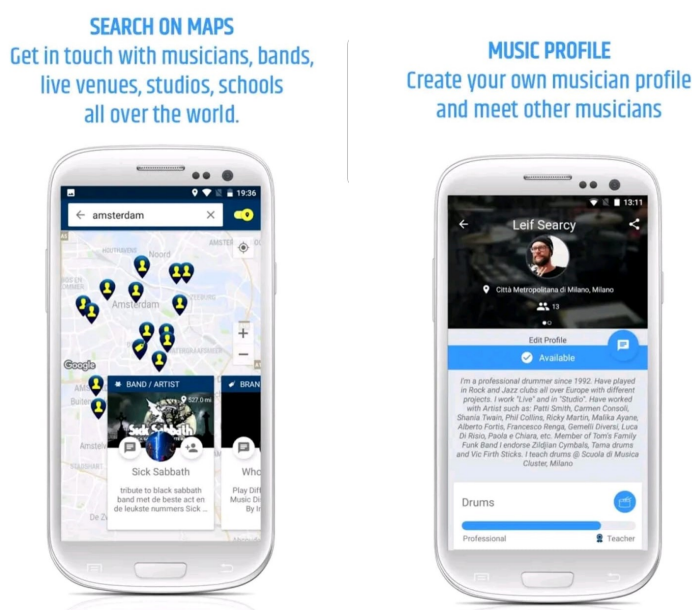
- Bendovi mogu postaviti oglas da traže nove članove, na koje se glazbenici mogu javljati
- Moguće organizirati „Nasumična druženja“ da se hrpa glazbenika nađe i sviraju zajedno bez da su dio istog benda.
- Dodavanje komentara na objave

- Dodavanje novih funkcionalnosti za organizatore (mogućnost unajmljivanja ton majstora, fotografa itd.)
- Dodati „Podijeli na Facebook“ mogućnost za javne nastupe

## 2.2 Slična rješenja problema

### 2.2.1 Amy

*Amy* je mobilna aplikacija koja nakon registracije nudi različite opcije vrste korisnika (glazbenik, DJ i slično). Nakon odabira vrste korisnika nudi i unos instrumenata koje korisnik svira te odabir razine profesionalnosti na pojedinom instrumentu. Također, nudi i odabir glazbenih žanrova, kalendar s obavezama te pregledavanje glazbenika u blizini. Aplikacija ima jako puno potencijala, ali se često ruši što utječe na iskustvo korisnika.



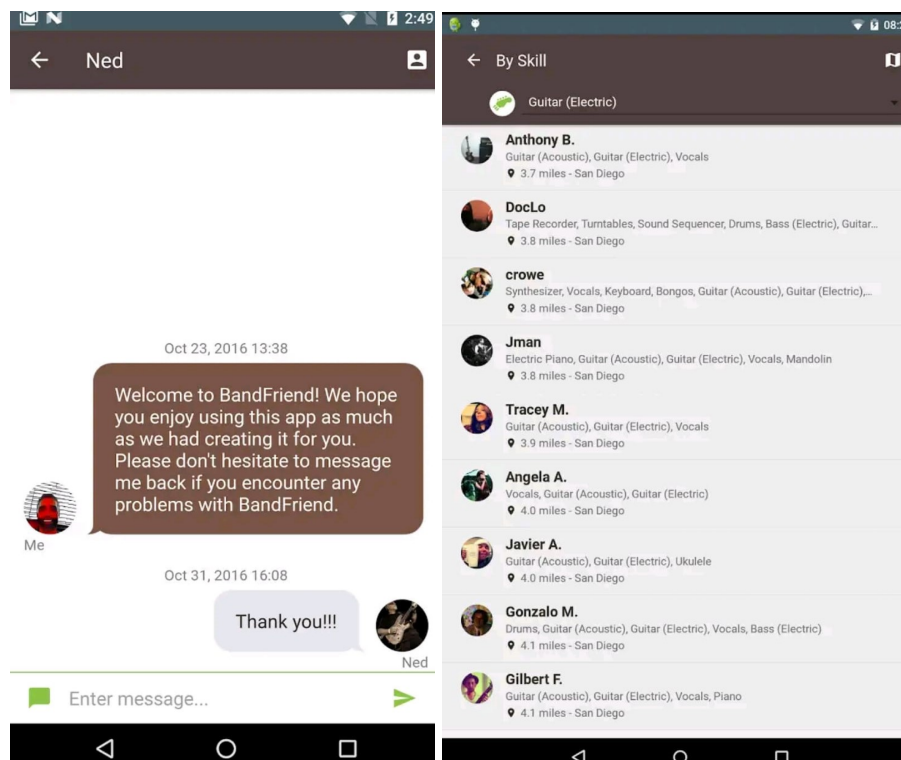
Slika 2.1: Prikaz *Amy* aplikacije

### 2.2.2 BandFriend

*BandFriend* je mobilna aplikacija koja prilikom registracije traži dosta informacija o korisniku. Nakon izrade profila, mogu se pretraživati novi glazbenici, glazbenici najbliži trenutnom korisniku, najbliži po lokaciji i slično. Aplikacija nudi



i razgovor porukama s drugim korisnicima. Neke korisnike tolika lista zahtjeva prilikom registracije može odbiti te će *Giger* tražiti samo korisničko ime, email i lozinku, a kasnije svaki korisnik, ukoliko to želi, može dodati više informacija o sebi. Također, *BandFriend* ne nudi kalendar s obavezama korisnika, stvaranje bendova ili događaja.



Slika 2.2: Prikaz *BandFriend* aplikacije

## 3. Specifikacija programske potpore

### 3.1 Funkcionalni zahtjevi

#### Dionici:

1. Naručitelj
2. Javnost
3. Korisnici
4. Organizator
5. Glazbenik
6. Voditelj benda
7. Administrator
8. Razvojni tim

#### Aktori i njihovi funkcionalni zahtjevi:

1. Neprijavljeni/neregistrirani korisnik (javnost) može:
  - (a) Pregledati javne događaje
  - (b) Pregledati profil benda
  - (c) Pročitati recenzije
  - (d) Registrirati se na aplikaciji
2. Korisnik (inicijator) može:
  - (a) Sve što i Javnost
  - (b) Prijaviti se na aplikaciju
  - (c) Pregledavati i uređivati korisnički profil
  - (d) Pregledati postojeće poruke s drugim korisnicima
  - (e) Pregledati svoj kalendar
  - (f) Slati nove poruke drugim korisnicima
  - (g) Recenzirati događaje
  - (h) Komentirati objave glazbenika
  - (i) Postati glazbenik i/ili organizator

- (j) Pisati objave na svom profilu
- (k) Odjaviti se

3. Glazbenik (inicijator) može:

- (a) Sve što i Korisnik
- (b) Prihvatiti ili odbiti poziv u bend
- (c) Postati članom jednog ili više bendova
- (d) Održati samostalan nastup
- (e) Stvoriti bend

4. Organizator (inicijator) može:

- (a) Sve što i Korisnik
- (b) Pretraživati bendove na aplikaciji
- (c) Stvarati događaje
- (d) Uređivati događaje
- (e) Recenzirati bend kao organizator događaja

5. Voditelj benda (inicijator) može:

- (a) Sve što i Glazbenik
- (b) Pozivati i dodavati nove članove u bend
- (c) Dodavati obaveze u kalendar benda
- (d) Vidjeti kalendare članova benda
- (e) Recenzirati organizatora u ime benda
- (f) Napisati objave na stranici benda
- (g) Uređivati profil benda

6. Administrator (inicijator) može:

- (a) Blokirati korisnike koji su prekršili uvjete korištenja aplikacije
- (b) Uređivati popis instrumenata
- (c) Dodati događaje koji su stvoreni na drugim platformama

7. Baza podataka (sudionik) može:

- (a) Pohraniti podatke o korisnicima, glazbenicima, organizatorima i bendovima
- (b) Pohraniti podatke o nastupima, recenzijama, objavama, komentarima
- (c) Pohraniti razmijenjene poruke između korisnika

### 3.1.1 Obrasci uporabe

#### Opis obrazaca uporabe

##### UC1 - Pristup listi javnih događaja

- **Glavni sudionik:** Javnost
- **Cilj:** Vidjeti listu javnih događaja na aplikaciji
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Javnost na početnoj stranici odabire prikaz događaja
  2. Aplikacija prikazuje javne događaje

##### UC2 - Pregled profila benda

- **Glavni sudionik:** Javnost
- **Cilj:** Vidjeti profil benda
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Javnost odabire bend čiji profil želi pregledati
  2. Aplikacija prikazuje profil odabranog benda

##### UC3 - Pregled recenzija

- **Glavni sudionik:** Javnost
- **Cilj:** Vidjeti pregled recenzija
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**
  1. Javnost odabire popis recenzija
  2. Aplikacija prikaže popis recenzija
  3. Javnost odabere recenziju koju želi vidjeti
  4. Aplikacija korisniku prikaže odabranu recenziju

##### UC4 - Registracija

- **Glavni sudionik:** Javnost
- **Cilj:** Registrirati se
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Javnost odabire opciju za registraciju
  2. Aplikacija prikazuje prozor za unos podataka
  3. Javnost unosi tražene korisničke podatke
  4. Aplikacija korisniku šalje email za verifikaciju email-a
  5. Korisnik prima obavijest o uspješnoj registraciji
- **Opis mogućih odstupanja:**
  - 2.a Odabir već zauzetog korisničkog imena i/ili e-maila, unos podataka u nedozvoljenom formatu
    1. Sustav obavještava korisnika o neuspjehom upisu i vraća ga na stranicu za registraciju.
    2. Korisnik mijenja potrebne podatke ili odustaje od registracije.

#### UC5 - Prijava u sustav

- **Glavni sudionik:** Korisnik
- **Cilj:** Korištenje aplikacije
- **Sudionici:** Baza podataka
- **Preduvjet:** Napravljena registracija
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju "Prijava"
  2. Aplikacija prikazuje prozor za prijavu
  3. Korisnik unosi podatke
  4. Baza autorizira unesene podatke te dozvoljava korisniku korištenje aplikacije
  5. Aplikacija korisniku prikazuje početnu stranicu
- **Opis mogućih odstupanja:**
  - 2.a Aplikacija ne dozvoljava Korisniku korištenje aplikacije ako su podaci neispravni

#### UC6 - Uvid u popis dodanih instrumenata

- **Glavni sudionik:** Administrator
- **Cilj:** Izmjena krivo unesenih instrumenata u bazi
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen u sustav
- **Prioritet:** Low
- **Opis osnovnog tijeka:**
  1. Administrator odabire pregled dodanih instrumenata
  2. Baza ispisuje dodane instrumente
  3. Administrator mijena krivo unesene zapise

#### UC7 - Blokiranje korisnika koji su prekršili uvjete korištenja

- **Glavni sudionik:** Administrator
- **Cilj:** Blokirati korisnike koji krše uvjete korištenja aplikacije
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen u sustav
- **Prioritet:** Low
- **Opis osnovnog tijeka:**
  1. Administrator uočava korisnika koji krši opća pravila korištenja sustava
  2. Administrator blokira korisnički račun određenoga korisnika na neodređeno vrijeme

#### UC8 - Dodavanje događaja

- **Glavni sudionik:** Administrator
- **Cilj:** Dodati događaje koji su kreirani van sustava u sustav
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen u sustav
- **Prioritet:** Low
- **Opis osnovnog tijeka:**
  1. Administrator odabire opciju za dodavanje događaja u sustav
  2. Administrator dodaje događaj u sustav

#### UC9 - Pregled poruka

- **Glavni sudionik:** Korisnik
- **Cilj:** Uvid u postojeće poruke
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen

- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Korisnik odabire "Poruke"
  2. Aplikacija prikazuje osobe s kojima se već vodio razgovor
  3. Korisnik odabire osobu te se prikazuju poruke
- **Opis mogućih odstupanja:**
  - 2.a Korisnik nema prošlih poruka
    1. Aplikacija prikazuje poruku „Nema poruka“

### UC10 - Pisanje poruke

- **Glavni sudionik:** Korisnik
- **Cilj:** Komunikacija među korisnicima
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Korisnik odabire poruke
  2. Aplikacija prikazuje korisnikove razgovore
  3. Korisnik odabire drugog korisnika s kojim želi komunicirati
  4. Aplikacija prikazuje postojeće poruke s odabranom osobom
  5. Korisnik unosi novu poruku
  6. Korisnik odabere „Pošalji“ za slanje poruke
- **Opis mogućih odstupanja:**
  - 3.a Korisnik želi poslati poruku osobi s kojom još nije komunicirao
    1. Korisnik odabire opciju „Nova poruka“
    2. Pronalazi korisnika kojemu želi napisati poruku

### UC11 - Pisanje recenzije o bendu

- **Glavni sudionik:** Korisnik
- **Cilj:** Napisati recenziju o bendu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti prijavljen
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**
  1. Korisnik pristupa profilu benda
  2. Korisnik u okvir za recenzije napiše svoje mišljenje te da ocjenu

## 3. Korisnik odabere "Završi recenziju"

**UC12 - Pisanje recenzije za događaj**

- **Glavni sudionik:** Korisnik
- **Cilj:** Napisati recenziju za događaj
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za recenziranje događaja
  2. Aplikacija otvara prozor za unos recenzije
  3. Korisnik u okvir za recenzije napiše svoje mišljenje te da ocjenu
  4. Korisnik odabere "Završi recenziju"
  5. Aplikacija sprema recenziju

**UC13 - Pregled profila korisnika**

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregled profilne stranice korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Korisnik odabire prikaz profila korisnika
  2. Aplikacija prikazuje profil korisnika
    - (a) Ako je glazbenik onda mu se prikazuju dodatne informacije (kalendar, instrumenti, popis događaja na kojima svira)
    - (b) Ako je organizator onda mu se prikazuju dodatne informacije (menager name i recenzije kao organizator)

**UC14 - Komentiranje objava glazbenika**

- **Glavni sudionik:** Korisnik
- **Cilj:** Komentiranje objave glazbenika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**



1. Korisnik odabire "Komentiraj" na objavi glazbenika
2. Aplikacija otvara prozor za unos komentara
3. Korisnik piše komentar
4. Korisnik odabere "Spremi"
5. Aplikacija sprema objavu

#### **UC15 - Uređivanje profila korisnika**

- **Glavni sudionik:** Korisnik
- **Cilj:** Promjeniti informacije na profilu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za uređivanje profila
  2. Korisnik mijenja podatke
  3. Korisnik odabirom opcije za spremanje potvrđuje i sprema načinjene promjene
- **Opis mogućih odstupanja:**
  - 3.a Korisnik nije unio obavezne podatke ili su podatci nevaljani
    1. Korisnik ponovno unosi obavezne podatke
    2. Korisnik odustaje od uređivanja profila

#### **UC16 - Stvaranje profila organizatora**

- **Glavni sudionik:** Korisnik
- **Cilj:** Dodavanje privilegija organizatora korisniku
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Korisnik odabire uređivanje profila
  2. Korisnik odabire opciju "Organizator"
  3. Aplikacija prikazuje dodatna polja za unos informacija
  4. Korisnik popunjava potrebne informacije
  5. Korisnik spremanjem promjena postaje Organizator
- **Opis mogućih odstupanja:**
  - 5.a Korisnik nije unio obavezne podatke ili su podatci nevaljani

1. Korisnik ponovno unosi obavezne podatke
2. Korisnik odustaje od izmjene profila

### UC17 - Stvaranje profila glazbenika

- **Glavni sudionik:** Korisnik
- **Cilj:** Dodavanje privilegija glazbenika korisniku
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Korisnik odabire uređivanje profila
  2. Korisnik odabire opciju "Glazbenik"
  3. Aplikacija prikazuje dodatna polja za unos informacija
  4. Korisnik popunjava potrebne informacije
  5. Korisnik spremanjem promjena postaje Glazbenik
- **Opis mogućih odstupanja:**
  - 4.a Korisnik nije unio obavezne podatke ili su podatci nevaljani
    1. Korisnik ponovno unosi obavezne podatke
    2. Korisnik odustaje od izmjene profila

### UC18 - Odjava

- **Glavni sudionik:** Korisnik
- **Cilj:** Odjavljivanje iz aplikacije
- **Sudionici:** -
- **Preduvjet:** Korisnik je prijavljen
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za odjavu

### UC19 - Uređivanje profila glazbenika

- **Glavni sudionik:** Glazbenik
- **Cilj:** Promijeniti informacije na profilu
- **Sudionici:** Baza podataka
- **Preduvjet:** Glazbenik je prijavljen u sustav
- **Prioritet:** High
- **Opis osnovnog tijeka:**

1. Glazbenik odabire opciju za uređivanje profila
  2. Glazbenik mijenja podatke
  3. Glazbenik odabirom opcije za spremanje potvrđuje i sprema načinjene promjene
- **Opis mogućih odstupanja:**
    - 3.a Glazbenik nije unio obavezne podatke ili su podatci nevaljani
      1. Glazbenik ponovno unosi obavezne podatke
      2. Glazbenik odustaje od uređivanja profila

#### UC20 - Promjena liste instrumenata

- **Glavni sudionik:** Administrator
- **Cilj:** Promijeniti listu instrumenata
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijavljen je administrator
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**
  1. Administrator dobiva popis prijedloga instrumenata
  2. Administrator označi koje instrumente želi dodati
  3. Administrator odabire "Dodaj"

#### UC21 - Pisanje objava

- **Glavni sudionik:** Korisnik
- **Cilj:** Objaviti objavu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**
  1. Korisnik u izborniku bira opciju "Post"
  2. Aplikacija prikaže prozor za unos sadržaja objave
  3. Korisnik piše objavu
  4. Korisnik odabire opciju "Objavi"
  5. Aplikacija sprema objavu i prikazuje ju na profilu korisnika

#### UC22 - Poziv u bend

- **Glavni sudionik:** Voditelj benda
- **Cilj:** Dodati glazbenika u bend

- **Sudionici:** Baza podataka, Glazbenik
- **Preduvjet:** Voditelj benda je prijavljen
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Voditelj benda nalazi glazbenika kojeg želi dodati u bend u listi glazbenika
  2. Voditelj benda odabire opciju "Dodaj glazbenika"
  3. Glazbenik dobiva poziv za učlanjenje u bend

#### UC23 - Učlanjenje u bend

- **Glavni sudionik:** Glazbenik
- **Cilj:** Ući u bend
- **Sudionici:** Baza podataka
- **Preduvjet:** Glazbenik je prijavljen u sustav
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Glazbenik odabire prikaz obavijesti o pozivu u bend
  2. Aplikacija prikazuje obavijest
  3. Glazbenik odabire opciju "Prihvati" ili "Odbij"

#### UC24 - Pregled kalendara

- **Glavni sudionik:** Glazbenik
- **Cilj:** Pregledati vlastiti kalendar
- **Sudionici:** Baza podataka
- **Preduvjet:** Glazbenik je prijavljen u sustav
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Glazbenik odabire opciju "Prikaži kalendar"
  2. Aplikacija prikazuje glazbenikov kalendar

#### UC25 - Dodavanje obaveza u kalendar

- **Glavni sudionik:** Voditelj benda
- **Cilj:** Dodati obaveze u kalendar benda
- **Sudionici:** Baza podataka
- **Preduvjet:** Voditelj benda je prijavljen u sustav
- **Prioritet:** High

- **Opis osnovnog tijeka:**

1. Voditelj benda odabire opciju "Dodaj obavezu" na profilu benda
2. Aplikacija prikazuje polje unos događaja u kalendar
3. Voditelj benda unosi događaje u kalendar

- **Opis mogućih odstupanja:**

- 3.a Voditelj benda nije unio obavezne podatke ili su podatci nevaljani
  1. Voditelj benda ponovno unosi podatke
  2. Voditelj benda odustaje od uređivanja profila

### UC26 - Stvaranje benda

- **Glavni sudionik:** Glazbenik

- **Cilj:** Stvoriti bend

- **Sudionici:** Baza podataka

- **Preduvjet:** Glazbenik je prijavljen u sustav

- **Prioritet:** High

- **Opis osnovnog tijeka:**

1. Glazbenik bira opciju "Stvori bend"
2. Aplikacija prikazuje polja za unos podataka
3. Glazbenik u polja za unos unosi naziv benda i žanr
4. Glazbenik bira opciju "Spremi"
5. Aplikacija pohranjuje bend

- **Opis mogućih odstupanja:**

- 4.a Glazbenik nije unio obavezne podatke ili su podatci nevaljani
  1. Glazbenik ponovno unosi podatke
  2. Glazbenik odustaje od uređivanja profila

### UC27 - Objavljivanje objava benda

- **Glavni sudionik:** Voditelj benda

- **Cilj:** Objaviti objavu na profilu benda

- **Sudionici:** Baza podataka

- **Preduvjet:** Voditelj benda je prijavljen u sustav

- **Prioritet:** Medium

- **Opis osnovnog tijeka:**

1. Voditelj benda bira opciju "Post"
2. Aplikacija prikazuje prozor za unos sadržaja objave
3. Voditelj benda piše objavu

4. Voditelj bira opciju "Objavi"
5. Aplikacija sprema objavu i prikazuje ju na profilu benda

#### **UC28 - Pregled kalendara članova benda**

- **Glavni sudionik:** Voditelj benda
- **Cilj:** Pregledati kalendar bilo kojeg člana benda
- **Sudionici:** Baza podataka
- **Preduvjet:** Voditelj benda prijavljen je u sustav
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**
  1. Voditelj benda odabire opciju za pregled kalendara članova benda
  2. Aplikacija prikazuje kalendare članova

#### **UC29 - Uređivanje profila benda**

- **Glavni sudionik:** Voditelj benda
- **Cilj:** Promijeniti informacije na profilu benda
- **Sudionici:** Baza podataka
- **Preduvjet:** Voditelj benda prijavljen je u sustav
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**
  1. Voditelj benda odabire opciju za uređivanje profila benda
  2. Voditelj benda mijenja podatke
  3. Voditelj benda odabirom opcije za spremanje potvrđuje i sprema načinjene promjene
- **Opis mogućih odstupanja:**
  - 3.a Voditelj benda nije unio obavezne podatke ili su podatci nevaljani
    1. Voditelj benda ponovno unosi obavezne podatke
    2. Voditelj benda odustaje od uređivanja profila

#### **UC30 - Pretraga bendova**

- **Glavni sudionik:** Organizator
- **Cilj:** Pregledati bendove prema određenom kriteriju
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen u sustav
- **Prioritet:** High
- **Opis osnovnog tijeka:**

1. Organizator bira opciju "Bendovi"
2. Aplikacija prikazuje popis bendova
3. Organizator može odabrati filtrirati bendove

### **UC31 - Pregled povijesti benda**

- **Glavni sudionik:** Javnost
- **Cilj:** Pregledati povijest benda
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Prioritet:** Low
- **Opis osnovnog tijeka:**
  1. Javnost bira opciju "Prikaži biografiju" na stranici benda
  2. Aplikacija prikazuje biografiju benda

### **UC32 - Kreirati događaj**

- **Glavni sudionik:** Organizator
- **Cilj:** Kreirati događaj
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen u sustav
- **Prioritet:** High
- **Opis osnovnog tijeka:**
  1. Organizator bira opciju "Kreiraj događaj"
  2. Aplikacija prikazuje prozor za unos događaja
  3. Organizator unosi potrebne informacije
  4. Organizator bira opciju "Spremi"
  5. Aplikacija sprema događaj u bazu
- **Opis mogućih odstupanja:**
  - 4.a Organizator nije unio obavezne podatke ili su podatci nevaljani
    1. Organizator ponovno unosi obavezne podatke
    2. Organizator odustaje od kreiranja događaja

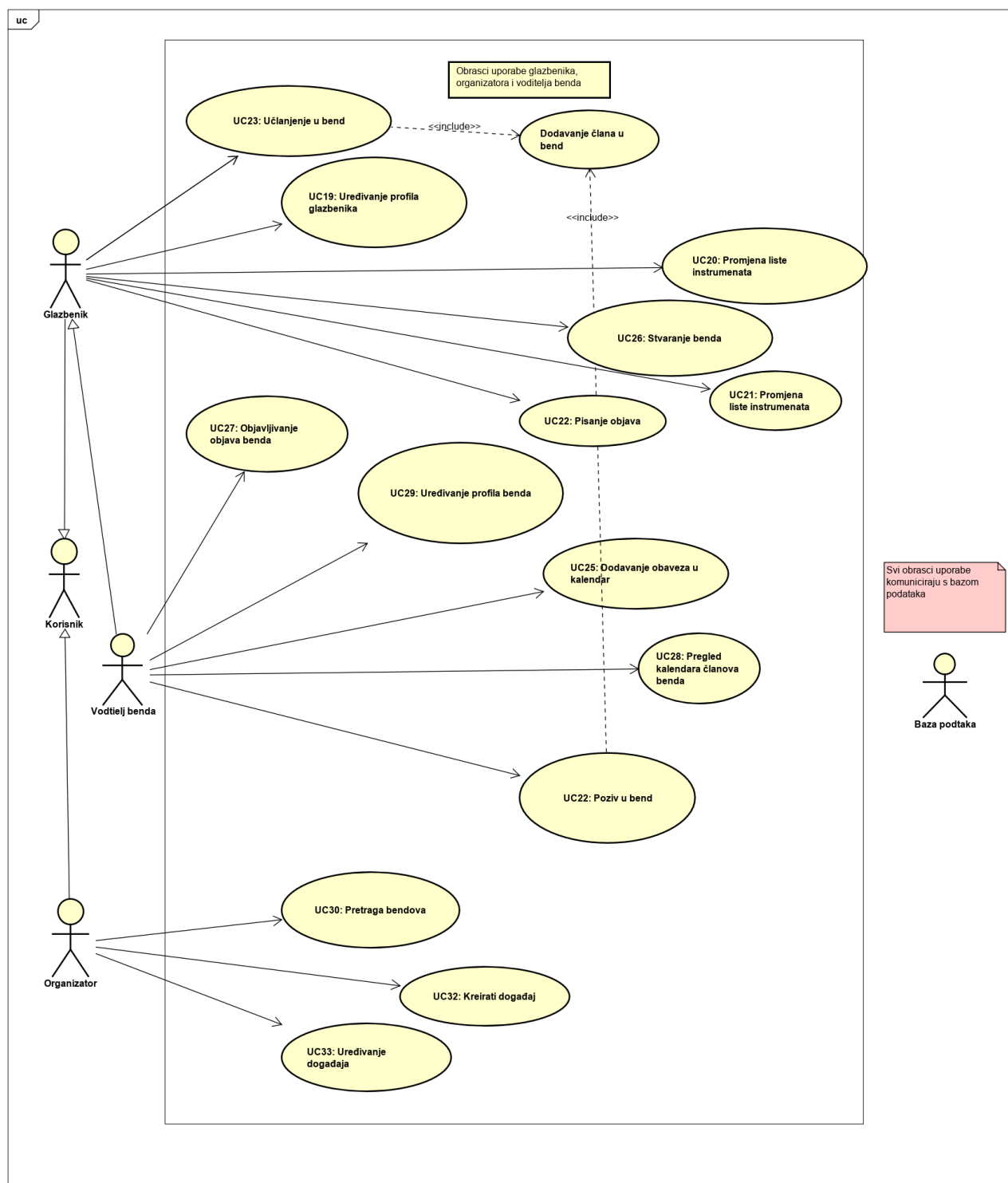
### **UC33 - Uređivanje događaja**

- **Glavni sudionik:** Organizator
- **Cilj:** Urediti informacije o događaju
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen u sustav

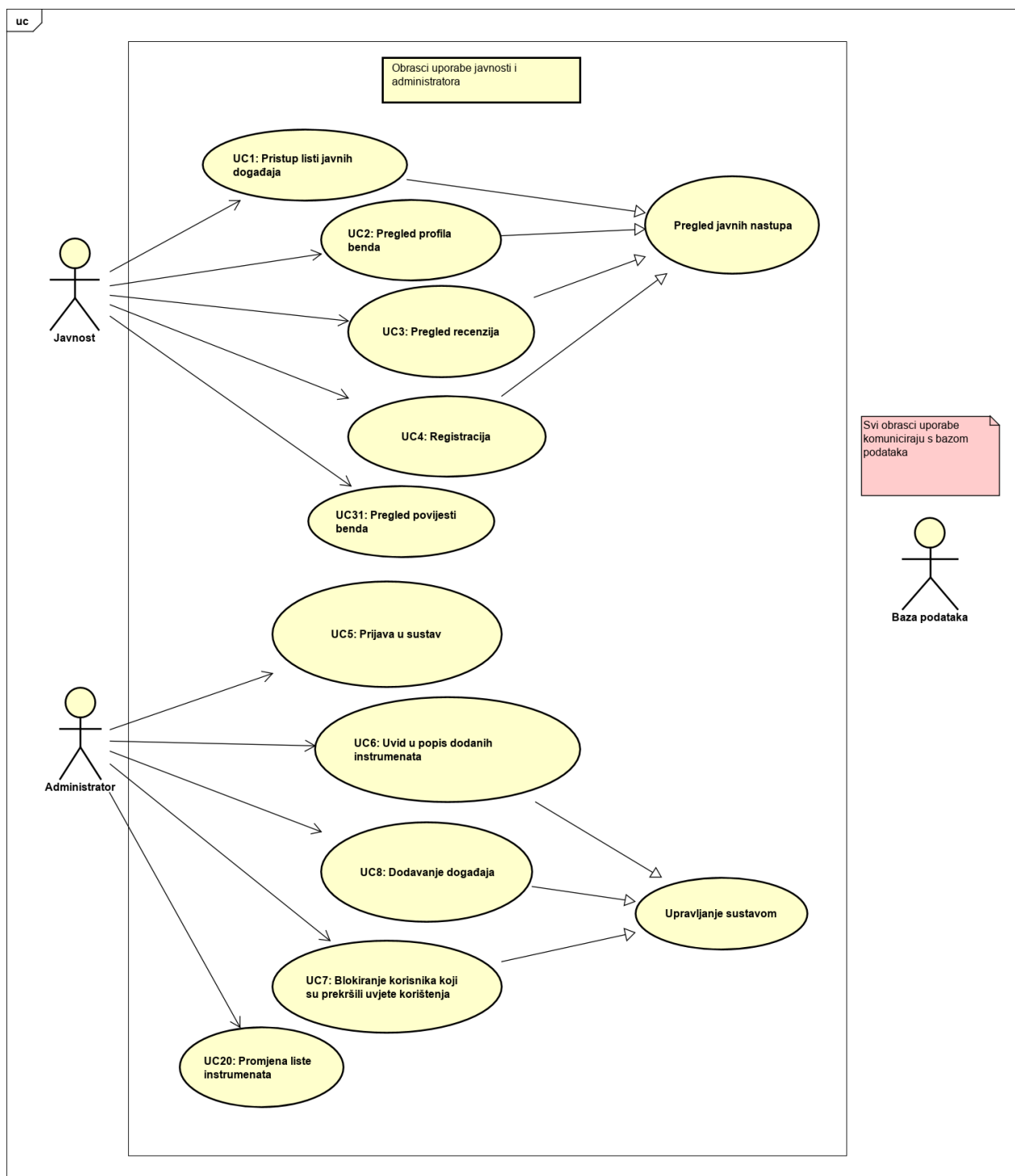
- **Prioritet:** Medium
- **Opis osnovnog tijeka:**
  1. Organizator odabire opciju "Uredi" na stranici događaja
  2. Aplikacija prikazuje prozor za izmjenu događaja
  3. Organizator bira opciju "Spremi"
  4. Aplikacija sprema promjene u bazu
- **Opis mogućih odstupanja:**
  - 3.a Organizator želi unijeti nevaljane podatke
    1. Organizator ponovno unosi podatke
    2. Organizator odustaje od izmjene događaja
  - 3.b Organizator želi promijeniti podatke za koje nije dozvoljena izmjena
    1. Organizator ponovno unosi podatke
    2. Organizator odustaje od izmjene događaja



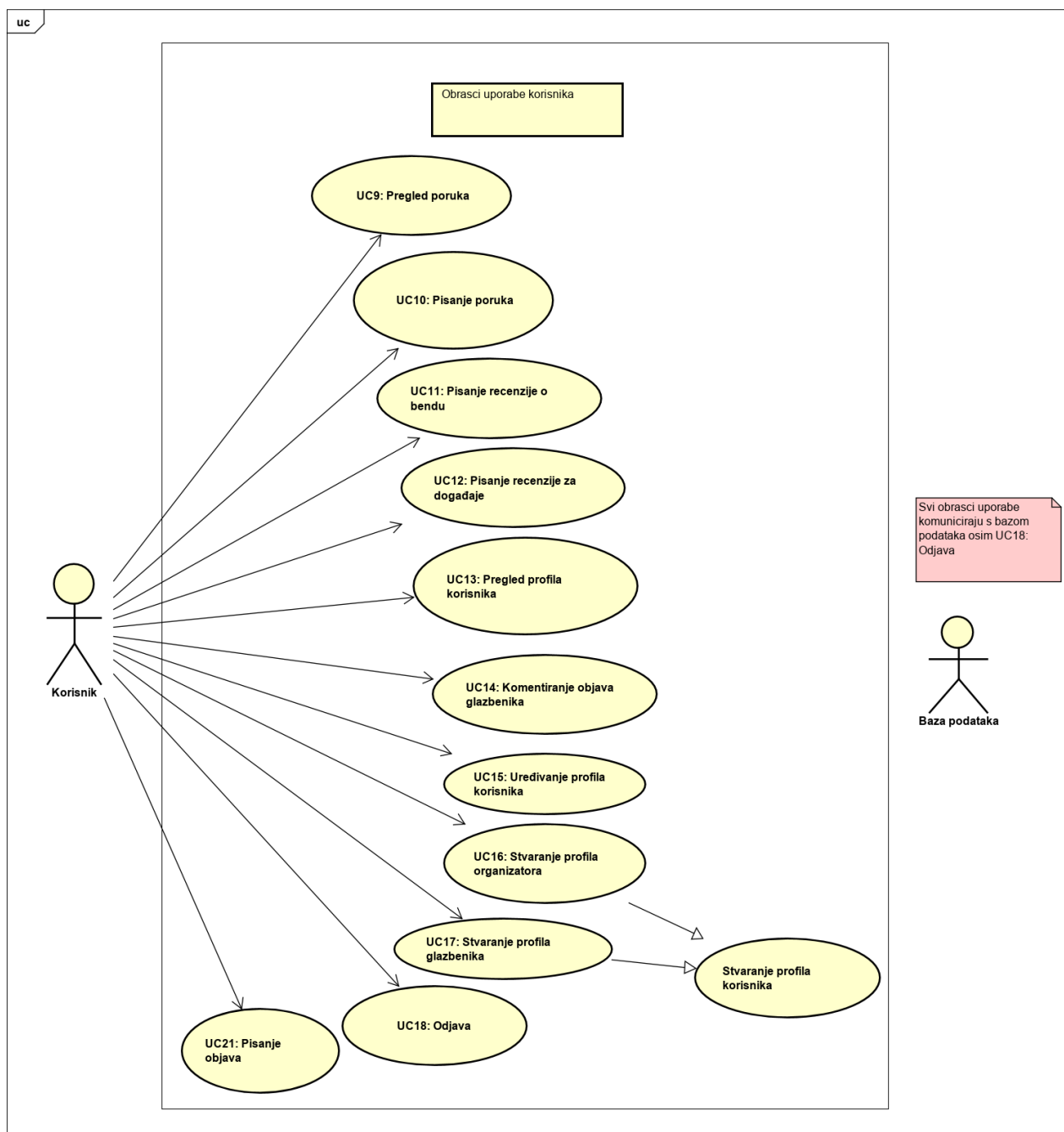
## Dijagrami obrazaca uporabe



Slika 3.1: Obrasci uporabe za korisnika i organizatora



Slika 3.2: Obrasci uporabe za javnost i administratora



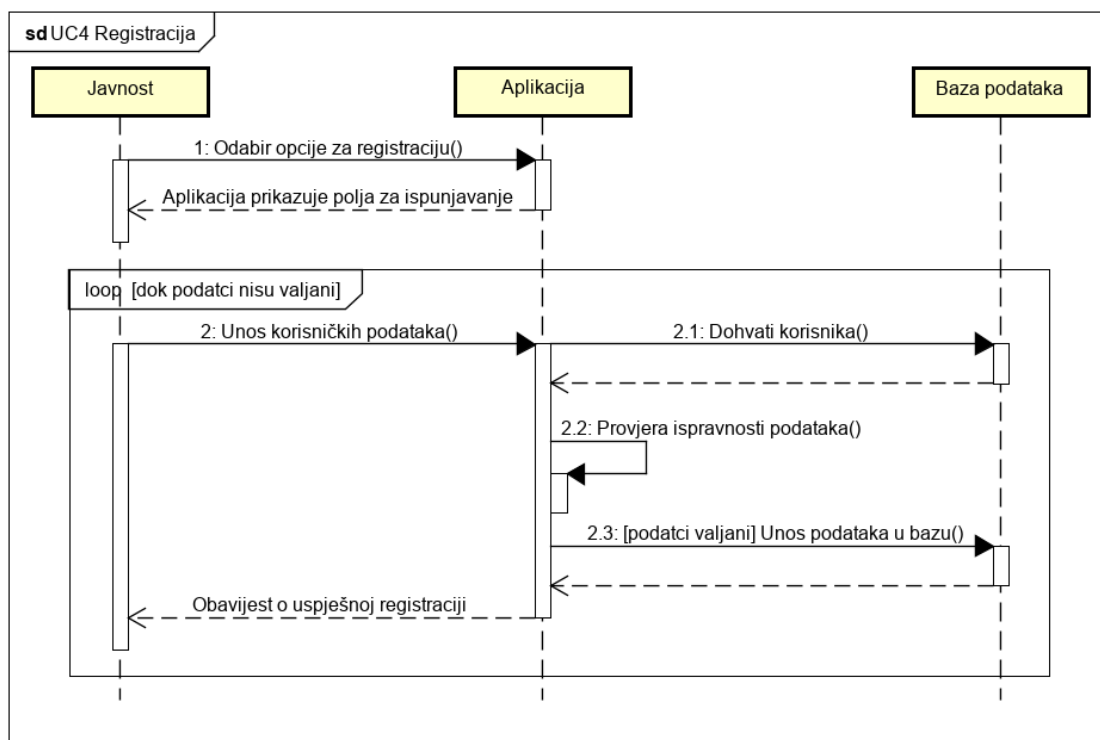
Slika 3.3: Obrasci uporabe za korisnika

### 3.1.2 Sekvencijski dijagrami

#### Obrazac uporabe UC4 - Registracija

Javnost (neregistrirani korisnik) odabire opciju za registraciju na što mu aplikacija odgovara prikazom polja za ispunjavanje (korisničko ime, email i lozinka). Dok unos podataka nije ispravan, korisnik unosi podatke te aplikacija dohvaća koris-

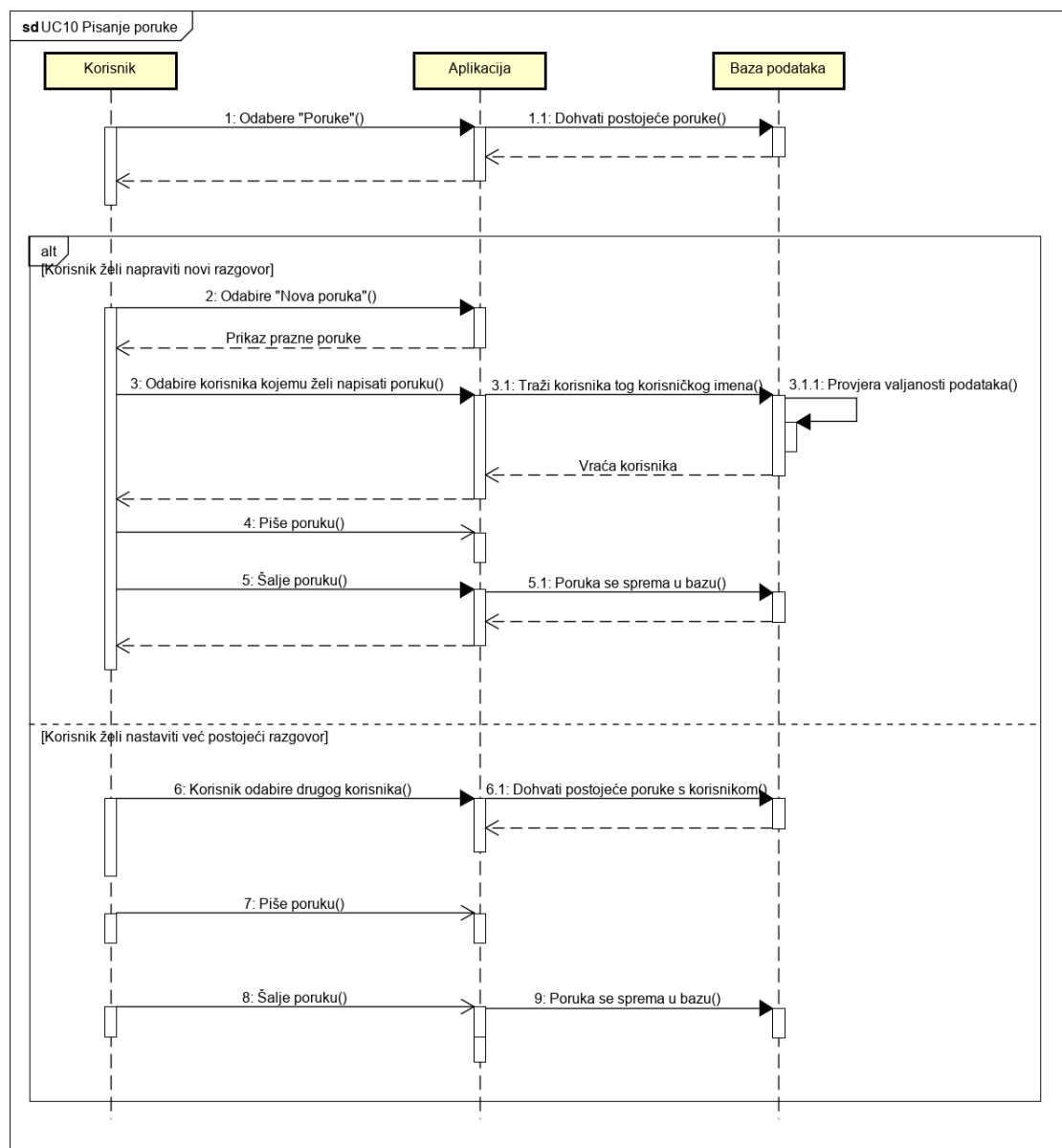
nike iz baze nakon čega provjerava postoji li korisnik istog korisničkog imena ili emaila. Vrš se provjera ispravnosti podataka te ukoliko su podatci valjani, unose se u bazu. Nakon unosa podataka u bazu, korisniku se šalje obavijest o uspješnoj registraciji.



Slika 3.4: Sekvencijski dijagram za UC4

### Obrazac uporabe UC10 - Pisanje poruke

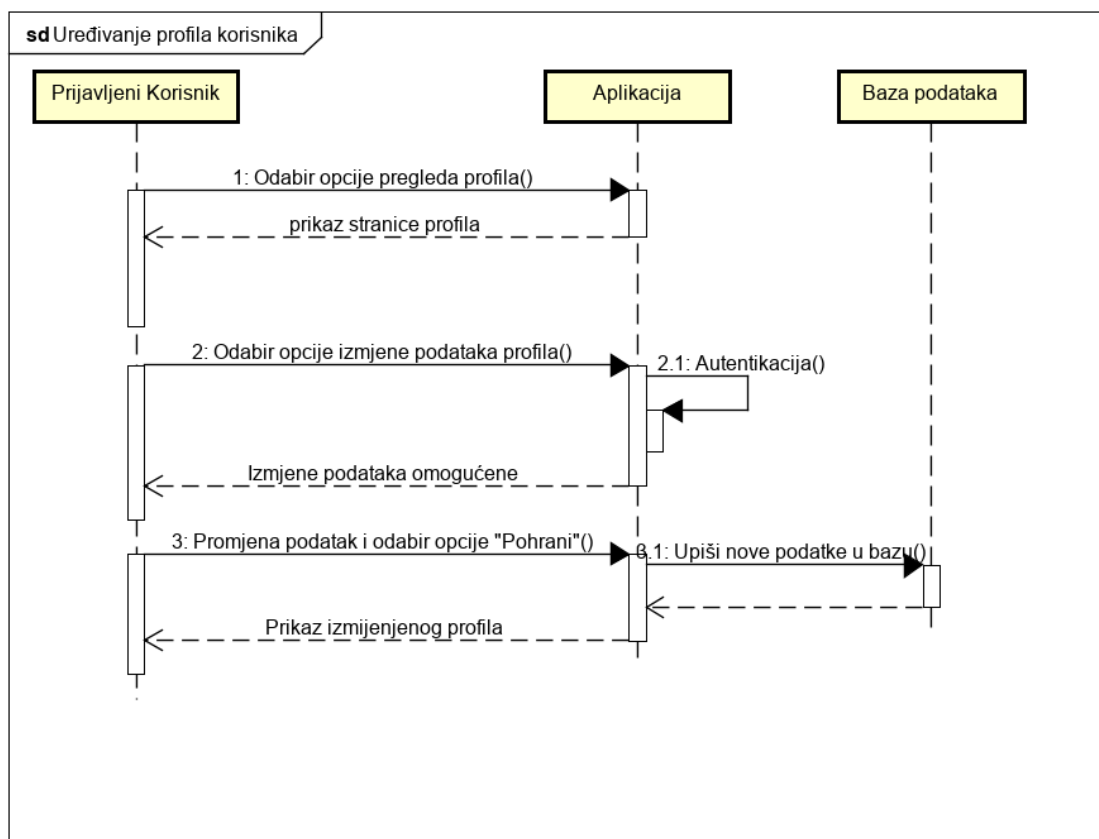
Korisnik odabire "Poruke" nakon čega se iz baze dohvaćaju postojeće poruke korisnika. Korisnik može ili napraviti novi razgovor, ili odabrati postojeći. Ukoliko želi napraviti novi razgovor, odabire opciju "Nova poruka" te mu se prikaže prazna poruka. Nakon toga, odabire korisnika kojemu želi napisati poruku, a taj korisnik se traži u bazi podataka. Ukoliko korisnik želi razgovarati s korisnikom s kojim je već komunicirao, odabire razgovor s tim korisnikom. Aplikacija dohvaća iz baze podataka prošle poruke s odabranim korisnikom. U oba opisana slučaja, korisnik piše željenu poruku te ju šalje nakon čega se poruka sprema u bazu.



Slika 3.5: Sekvencijski dijagram za UC10

**Obrazac uporabe UC15 - Uređivanje profila korisnika**

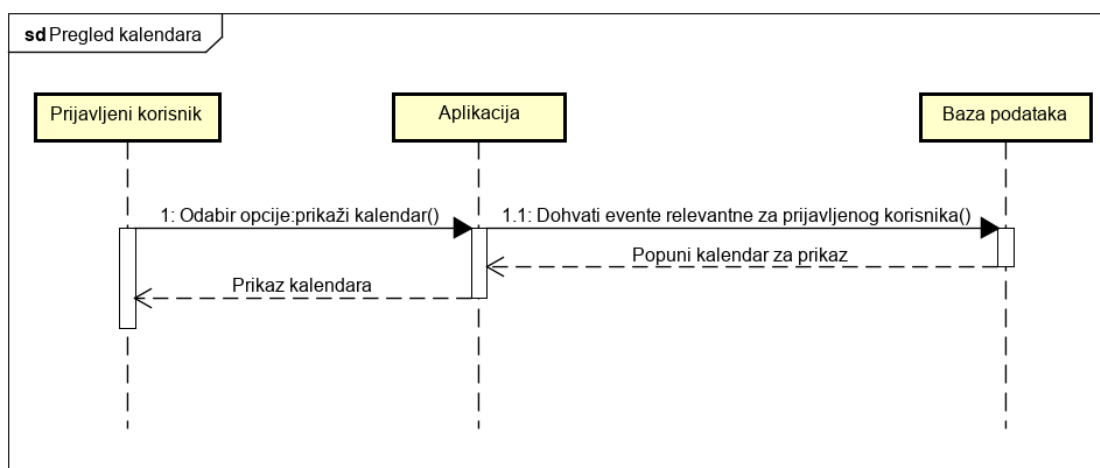
Prijavljeni korisnik ide na stranicu svog profila. Korisnik potom bira opciju izmjene podataka na svom profilu, nakon čega slijedi proces autentikacije korisnika. Ukoliko je autentikacija prošla, omogućene su izmjene podataka korisniku. Korisnik po želji mijenja podatke nakon čega odabire opciju "Izmijeni". Promijenjeni podatci se zapisuju u bazu, a korisniku se prikazuje izmijenjena stranica profila.



Slika 3.6: Sekvencijski dijagram za UC15

### Obrazac uporabe UC24 - Pregled kalendara

Prijavljeni korisnik bira opciju prikaži kalendar, nakon čega aplikacija uzima iz baze podataka sve događaje koji su relevantni za korisnika (one na kojima je bio/će biti ili je svirao/će svirati). Aplikacija puni kalendar tim događajima i prikazuje ih korisniku.



Slika 3.7: Sekvencijski dijagram za UC24

## 3.2 Ostali zahtjevi

- Sustav treba omogućiti rad više korisnika u isto vrijeme
- Neispravno korištenje sučelja ne smije na bilo koji način promijeniti ili obustaviti rad sustava
- Sustav mora biti jednostavan za uporabu, tj. mora biti intuitivan
- Aplikacija mora podržavati hrvatske dijakritičke znakove
- Sustav mora biti u stanju brzo obraditi dobivene podatke kako korisnik ne bi dugo čekao promjenu
- Sustav treba biti implementiran kao web aplikacija, s tim da bi joj većina korisnika pristupala preko mobilnih uređaja
- Web aplikacija treba biti implementirana koristeći objektno-orijentirane jezike

- Treba osigurati sigurnost aplikacije, tj. lozinke moraju biti enkriptirane te mora biti osigurana sigurnost veze između korisnika aplikacije i baze podataka



## 4. Arhitektura i dizajn sustava

Da bi dugoročno uštedjeli vrijeme, uložili smo dio vremena na konfiguriranje CI (engl. continuous integration) i CD (engl. continuous delivery) procesa. Za isporuku aplikacije odabrali smo Heroku.

Heroku je jedna od najpoznatijih platformi za isporučivanje aplikacija koja se ističe svojom jednostavnošću. Za razliku od AWS-ovih servisa, koje smo također razmatrali, Heroku se sam brine oko instanci i arhitekture sustava na kojem se izvodi naša aplikacija. Zbog ograničenih resursa odlučili smo isporučiti aplikaciju na besplatnu instancu Heroku-a.

Besplatna instanca Heroku-a ima određena ograničenja, a najupečatljivije od njih je način na koji se pokreće projekt. Heroku sam prepoznaje kojeg je tipa projekt pa da pokrenemo frontend i backend trebamo dvije instance. CD je integriran putem GitLab-ovih pipelinesa za koje je bilo potrebno napisati `.gitlab-ci.yml` datoteku u kojoj smo konfigurirali GitLab-ov pipeline. GitLab-ov pipeline konfiguriran je tako da se na svaki commit u dev grani izgradi aplikacija, pokrenu i uspješno završe testovi te krene isporuka aplikacije na Heroku.

Da bi osigurali maksimalno vrijeme dostupnosti naše platforme, `.yml` datoteka također je konfigurirana tako da prilikom commita u master isporuči aplikaciju na druge dvije instance. Ukupno imamo četiri pokrenute instance Heroku-a, od koje su dvije backend, a dvije frontend. Backend i frontend imaju svaki svoju razvojnu i produkcijsku instancu. Poveznice na instance:

- <http://giger-fer.herokuapp.com/>
- <https://giger-fer-dev.herokuapp.com/>
- <https://giger-backend-dev.herokuapp.com/>
- <http://giger-backend.herokuapp.com/>

U skladu s time, Spring Boot aplikacija ima dvije `.properties` datoteke. Jedna od njih je namijenjena lokalnom izvođenju aplikacije te sadrži postavke lokalne PostgreSQL baze, dok je druga konfigurirana tako da postavke čita iz varijabli

okruženja. Varijable okruženja postavljene su na Heroku tako da čak niti pristupom u git repozitorij vanjski korisnik ne može doći do akreditacije (credentials) kojima bi mogao pristupiti bazi.

Prednost ostvarena automatiziranjem procesa isporuke jest povećanje vjerojatnosti uspješnosti iste te povećanje udjela dostupnosti aplikacije zbog dva para instanci servera.

Uvidjevši prednosti korištenja CD-a, primijenili smo to znanje i na prevođenje dokumentacije. Unutar repozitorija, osim pipeline-a za isporuku aplikacije, postoji pipeline za automatsko prevođenje dokumentacije čiji je rezultat .pdf dokument. Potencijalni prostor za napredak bio bi korištenje Docker tehnologije tako da prilikom pokretanja aplikacije korisnik ne mora imati instaliranu PostgreSQL bazu već ju pokrene u Docker kontejneru.

Od mogućih arhitektura sustava, za svoj projekt smo odabrali objektno usmjerenu arhitekturu. Tu arhitekturu smo odabrali zato što se koristi u industriji te je de facto standard razvoja složenih programskih rješenja. Osim toga, ona je fleksibilna, omogućuje recikliranje koda te logički razdjeljuje sustav na više cjelina, što je bitno s obzirom da više ljudi radi na implementaciji aplikacije. Zahvaljujući modularnosti programskog rješenja, greške su lako ispravljive, a nove mogućnosti dodaju se bez poteškoća.

Odlučili smo se za web aplikaciju, koja je prilagođena mobilnim uređajima, obzirom da glazbenici, a time i bendovi nemaju uvijek pristup računalu, a ne želimo da je korisnik ograničen samo na mobilne uređaje.

Arhitekturu sustava možemo podijeliti na četiri podsustava:

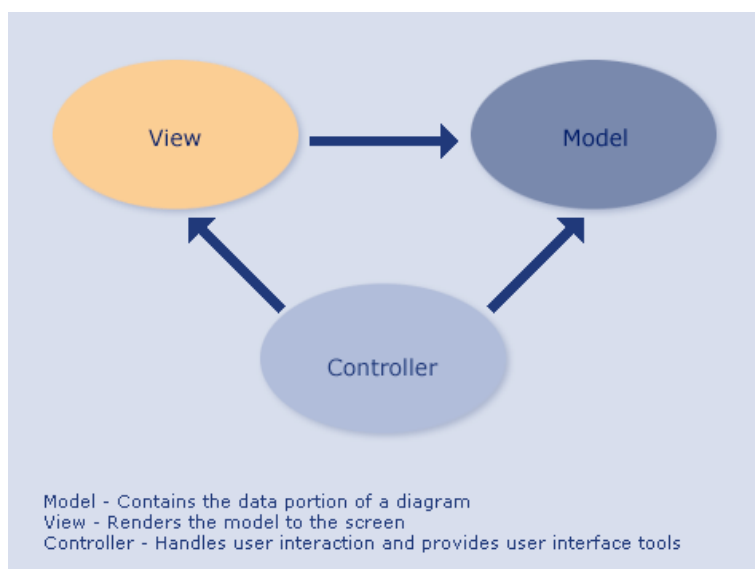
- Web preglednik
- Web poslužitelj
- Web aplikacija
- Baza podataka

Korisnik (javnost, glazbenik, bend, administrator) pristupa web aplikaciji uz pomoć svog web preglednika, s time da se u sredini nalazi web poslužitelj. Na njemu se nalazi aplikacija koju on pokreće, te uz pomoć protokola komunicira s korisnicima.

Klijentski (frontend) dio aplikacije omogućuje da korisnik korištenjem sučelja može pristupiti serveru (backend) aplikacije. Ovisno o tome što korisnik hoće,

taj server ima mogućnost spajanja na bazu podataka kako bi korisniku prikazao informacije.

Backend je napisan u Javi 11, a kao razvojni okvir koristimo Java Spring Boot 2.2.0. Dodani su projekti Spring Data JPA kako bi backend mogao lako komunicirati s bazom, Spring Web MVC za rukovanje zahtjevima te Spring Security kako bi zaštitili aplikaciju od vanjskih napada. Za pregledniji kod koristimo Lombok.



Slika 4.1: Pojednostavljeni prikaz MVC-a

Za frontend koristimo React. On je moderan i jednostavan framework koji koristi HTML, CSS, JSX i JavaScript uz pomoć kojeg smo napravili sučelje za našu aplikaciju. Uz pomoć React-a možemo lagano komunicirati s backendom koristeći REST.

## 4.1 Baza podataka

Za potrebe razvoja *Gigera* koristit će se objektno relacijsko mapiranje. To je metoda koja se koristi u objektno-orijentiranim jezicima te se na taj način stvara virtualna objektna baza podataka. Za implementaciju baze podataka odabrali smo PostgreSQL, zbog generalno pozitivnog iskustva u korištenju te implementacije baze podataka u dosadašnjem fakultetskom obrazovanju. Bitno je naglasiti da na osobnim računalima u svrhu razvijanja aplikacije koristimo istu implementaciju baze kao i na web poslužitelju kako bi minimizirali neočekivano ponašanje.

Baza podataka sastoji se od sljedećih tablica:

- Message
- Conversation\_messages
- Conversation
- Conversation\_participants
- System\_person
- Person
- System\_person\_roles
- Organizer
- Band
- Band\_occasions
- Occasion
- Musician\_occasions
- Band\_invited\_back\_up\_members
- Band\_invited
- Band\_back\_up\_members
- Band\_members
- Band\_acceptable\_gig\_types
- Band\_invitation\_gigs
- Band\_gigs
- Band\_posts
- Gig
- Gig\_reviews
- Review
- Post
- Post\_comments
- Comment
- Musician
- Musician\_instruments
- Instrument
- Musician\_past\_gigs

#### 4.1.1 Opis tablica

**Message** je entitet koji sadrži informacije o poruci. Ima attribute: id poruke, sadržaj poruke, vrijeme kada je poruka poslana, id pošiljatelja. Ovaj entitet je u *Many-to-*

One vezi s entitetima: Person i Band. Pošiljatelj može biti korisnik ili bend. Ako je bend poslao poruku, tada je `sender_id` null, a ako ju je poslao korisnik tada je `sender_band_id` null.

Message		
<b>id</b>	BIGINT	jedinstveni identifikator poruke
<b>content</b>	VARCHAR	sadržaj poruke
<b>sent_time</b>	TIMESTAMP	vrijeme kada je poruka poslana
<i>sender_id</i>	BIGINT	jedinstveni identifikator pošiljatelja
<i>sender_band_id</i>	BIGINT	jedinstveni identifikator benda pošiljatelja

**Conversation\_messages** je vezna tablica koja sadrži informacije koje poruke pripadaju određenom razgovoru. Sadrži attribute: id razgovora i id poruke. Ovaj entitet je u *Many-to-One* vezi s entitetima Conversation i Message.

Conversation_messages		
<i>conversation_id</i>	BIGINT	jedinstveni identifikator razgovora
<i>messages_id</i>	BIGINT	jedinstveni identifikator poruke razgovora

**Conversation** je entitet koji sadrži informacije o razgovoru. Sadrži attribute: id razgovora, url slike razgovora, ime razgovora i id benda. Ovaj entitet je u *Many-to-One* vezi s entitetom Band.

Conversation		
<b>id</b>	BIGINT	jedinstveni identifikator razgovora
<b>picture_url</b>	VARCHAR	url slike razgovora
<b>title</b>	VARCHAR	naziv razgovora
<i>band_id</i>	BIGINT	jedinstveni identifikator benda

**Conversation\_participants** je vezna tablica koja sadrži informacije o sudjelovanju korisniku u razgovoru. Sadrži attribute: id korisnika i id razgovora. Ovaj entitet je u *Many-to-One* vezi s entitetima Conversation i Person.

Conversation_participants		
<i>conversation_id</i>	BIGINT	jedinstveni identifikator razgovora
<i>participants_id</i>	BIGINT	jedinstveni identifikator korisnika koji sudjeluje u razgovoru

**System\_person** je entitet koji sadrži podatke korisnika potrebne sustavu za sistemsku logiku. Sadrži attribute: id korisnika, email, locked i verified zastavice te šifriranu lozinku. Ovaj entitet je u *One-to-One* vezi s entitetima: Person, Musician i Organizer.

System_person		
id	BIGINT	jedinstveni identifikator sustavskih podataka o korisniku
email	VARCHAR	email adresa osobe
locked	BOOLEAN	korisnik ima zabranu korištenja aplikacije ili ne
password_hash	VARCHAR	hash lozinke osobe
verified	BOOLEAN	email adresa potvrđena ili ne

**Person** je entitet koji sadrži podatke korisnika potrebne za poslovne svrhe. Sadrži attribute: id korisnika, telefonski broj, url slike korisnika te korisničko ime kojim se predstavlja javnosti. Ovaj entitet je u *One-to-One* vezi s entitetima: Musician, System\_person i Organizer.

Person		
id	BIGINT	jedinstveni identifikator korisnika
phone_number	VARCHAR	telefonski broj korisnika
picture_url	VARCHAR	url slike korisnika
username	VARCHAR	korisničko ime korisnika

**System\_person\_roles** je vezna tablica koja sadrži n-torke iz kojih možemo iščitati dodijeljene uloge pojedinim korisnicima. Sadrži attribute: system\_person\_id i cijeli broj uloge koji predstavlja enumeraciju.

System_person_roles		
<i>system_person_id</i>	BIGINT	jedinstveni identifikator sustavskih podataka o korisniku
roles	INT	uloga korisnika

**Organizer** je entitet koji sadrži informacije o organizatoru. Sadrži attribute: id organizatora te ime organizatora. Ovaj entitet je u *One-to-One* vezi s entitetima:

Musician, System\_person.

Organizer		
<b>id</b>	BIGINT	jedinstveni identifikator organizatora
manager_name	VARCHAR	ime organizatora

**Band** je entitet koji sadrži podatke o kreiranom bendu. Sadrži attribute: id, opis, datum formiranja, adresu sjedišta, dodatni opis adrese, par koordinata, maksimalnu udaljenost unutar koje bend želi svirati, naziv benda, url slike benda i id voditelja benda. Ovaj entitet je u *Many-to-One* vezi s Musician za potrebu evidencije voditelja benda.

Band		
<b>id</b>	BIGINT	jedinstveni identifikator benda
bio	VARCHAR	opis benda
formed_date	DATE	datum osnutka benda
address	VARCHAR	adresa benda
extra_description	VARCHAR	dodatak opis benda
x	DOUBLE	x koordinata lokacije
y	DOUBLE	y koordinata lokacije
max_distance	DOUBLE	najveća udaljenost koju bend želi prijeći zbog gaže
name	VARCHAR	ime benda
picture_url	VARCHAR	url slike benda
leader_id	BIGINT	jedinstveni identifikator voditelja benda

**Band\_occasions** je vezna tablica iz koje se može iščitati zauzetost benda. Sadrži attribute: identifikator benda i identifikator događaja. Ovaj entitet je u *Many-to-One* vezi s entitetima Band i Occasion.

Band_occasions		
<b>band_id</b>	BIGINT	jedinstveni identifikator benda koji sudjeluje na događaju
<b>occasion_id</b>	BIGINT	jedinstveni identifikator događaja

**Occasion** je entitet koji sadrži podatke o događaju. Sadrži attribute: id događaja, opis, datum te zastavicu privatnosti.

Occasion		
<b>id</b>	BIGINT	jedinstveni identifikator događaja
description	VARCHAR	opis događaja
local_date_time	DATE	datum i vrijeme održavanja događaja
personal_occasion	BOOLEAN	privatan događaj ili ne

**Musician\_occasions** je vezna tablica iz koje se može iščitati zauzetost glazbenika. Sadrži attribute: identifikator glazbenika i identifikator događaja. Ovaj entitet je u *Many-to-One* vezom s entitetima Musician i Occasion.

Musician_occasions		
<b>musician_id</b>	BIGINT	jedinstveni identifikator glazbenika
<b>occasions_id</b>	BIGINT	jedinstveni identifikator događaja

**Band\_invited\_back\_up\_members** je vezna tablica iz koje se mogu iščitati poslane i neodgovorene pozivnice za pričuvnog člana benda. Sadrži attribute: identifikator benda i identifikator glazbenika. On je u *Many-to-One* vezi s entitetima Band i Musician.

Band_invited_back_up_members		
<b>band_id</b>	BIGINT	jedinstveni identifikator benda
<b>invited_back_up_members_id</b>	BIGINT	jedinstveni identifikator glazbenika pozvanih u bend kao pričuveni član

**Band\_invited** je vezna tablica iz koje se mogu iščitati poslane i neodgovorene pozivnice za člana benda. Sadrži attribute: identifikator benda i identifikator glazbenika. On je u *Many-to-One* vezi s entitetima Band i Musician.

Band_invited		
<b>band_id</b>	BIGINT	jedinstveni identifikator benda
<b>invited_id</b>	BIGINT	jedinstveni identifikator glazbenika pozvanih u bend

**Band\_back\_up\_members** je vezna tablica iz koje se mogu iščitati pričuveni članovi bendova. Sadrži attribute: identifikator benda i identifikator glazbenika. On je u *Many-to-One* vezi s entitetima Band i Musician.



Band_back_up_members		
<i>band_id</i>	BIGINT	jedinstveni identifikator benda
<i>invited_back_up_members_id</i>	BIGINT	jedinstveni identifikator glazbenika koji su pričuveni članovi

**Band\_members** je vezna tablica iz koje se mogu iščitati članovi bendova. Sadrži attribute: identifikator benda i identifikator glazbenika. On je u *Many-to-One* vezi s entitetima Band i Musician.

Band_members		
<i>band_id</i>	BIGINT	jedinstveni identifikator benda
<i>members_id</i>	BIGINT	jedinstveni identifikator glazbenika koji je u bendu

**Band\_acceptable\_gig\_types** je vezna tablica iz koje se mogu iščitati sve vrste nastupa koje izvodi određeni bend. Sadrži attribute: id benda i naziv tipa nastupa. Ovaj entitet je u *Many-to-One* vezi s entitetom Band.

Band_acceptable_gig_types		
<i>band_id</i>	BIGINT	jedinstveni identifikator benda
<i>acceptable_gig_types</i>	VARCHAR	vrsta nastupa

**Band\_invitation\_gigs** je vezna tablica iz koje se mogu iščitati nastupi na koje je bend pozvan svirati. Sadrži attribute: id benda i id nastupa. Ovaj entitet je u *Many-to-One* vezom s entitetima Band i Gig.

Band_invitation_gigs		
<i>band_id</i>	BIGINT	jedinstveni identifikator benda
<i>invitation_gigs_id</i>	BIGINT	jedinstveni identifikator nastupa za koji se poziva

**Band\_gigs** je vezna tablica gdje se mogu vidjeti nastupi određenog benda. Sadrži attribute: id benda i id nastupa. Ovaj entitet je u *Many-to-One* vezom s entitetima Band i Gig.

Band_gigs		
<i>band_id</i>	BIGINT	jedinstveni identifikator benda
<i>gigs_id</i>	BIGINT	jedinstveni identifikator nastupa benda

**Band\_posts** je vezna tablica iz koje se mogu iščitati sve objave benda. Sadrži attribute: id benda i id objave. Ovaj entitet je u *Many-to-One* vezom s entitetima Band i Post.

Band_posts		
<i>band_id</i>	BIGINT	jedinstveni identifikator benda
<i>posts_id</i>	BIGINT	jedinstveni identifikator objave benda

**Gig** je entitet koji sadrži informacije o nastupima. Sadrži attribute: id nastupa, datum i vrijeme održavanja nastupa, opis nastupa, očekivano trajanje nastupa, oznaku za postignut dogovor, vrstu nastupa, adresa održavanja nastupa, dodatan opis nastupa, x koordinata lokacije, y koordinata lokacije, naziv nastupa, oznaku za privatan nastup, preporučenu cijenu ulaznice, id organizatora. Ovaj entitet je u *Many-to-One* vezi s entitetom Organizer.

Gig		
<b>id</b>	BIGINT	jedinstveni identifikator nastupa
date_time	TIMESTAMP	datum i vrijeme održavanja nastupa
description	VARCHAR	opis nastupa
expected_duration	VARCHAR	očekivano trajanje nastupa
final_deal_achieved	BOOLEAN	dogovor postignut ili ne
gig_type	INT	vrsta nastupa
address	VARCHAR	adresa održavanja nastupa
extra_description	VARCHAR	dodatan opis nastupa
x	DOUBLE	x koordinata lokacije
y	DOUBLE	y koordinata lokacije
name	VARCHAR	naziv nastupa
private_gig	BOOLEAN	nastupa privatan ili ne
proposed_price	INT	preporučena cijena ulaznice
<i>organizer_id</i>	BIGINT	jedinstveni identifikator organizatora

**Gig\_reviews** je vezna tablica iz koje pronalazimo pripadnost recenzije pojedinom nastupu. Sadrži attribute: id nastupa i id recenzije. Ovaj entitet je u *Many-to-One* vezom s entitetima Gig i Review.

Gig_reviews		
<i>gig_id</i>	BIGINT	jedinstveni identifikator nastupa
<i>reviews_id</i>	BIGINT	jedinstveni identifikator recenzije nastupa

**Review** je entitet koji sadrži informacije za recenziju. Sadrži attribute: id recenzije, sadržaj recenzije benda, sadržaj recenzije organizatora, vrijeme objave recenzije, ocjenu benda, ocjenu organizatora te id autora. Ovaj entitet je u *Many-to-One* vezi s entitetom Person.

Review		
<b>id</b>	BIGINT	jedinstveni identifikator recenzije
content_of_review_for_band	VARCHAR	sadržaj komentara benda
content_of_review_for_organizer	VARCHAR	sadržaj komentara organizatora
created	TIMESTAMP	vrijeme objave komentara
grade_band	INT	ocjena benda
grade_organizer	INT	ocjena organizatora
<i>author_id</i>	BIGINT	jedinstveni identifikator korisnika koji je autor recenzije

**Post** je entitet koji sadrži podatke o objavi. Sadrži attribute: id objave, sadržaj, datum i vrijeme objave.

Post		
<b>id</b>	BIGINT	jedinstveni identifikator objave
content	VARCHAR	sadržaj objave
published_on	TIMESTAMP	datum i vrijeme objave

**Post\_comments** je vezna tablica iz koje se može saznati koji komentari pripadaju određenoj objavi. Sadrži attribute: id objave i id komentara. Ovaj entitet je u *Many-to-One* vezom s entitetima Post i Comment.

Post_comments		
<i>post_id</i>	BIGINT	jedinstveni identifikator objave benda
<i>comments_id</i>	BIGINT	jedinstveni identifikator komentara objave

**Comment** je entitet koji sadrži jedan komentar. Sadrži attribute: id komentara, id autora, sadržaj te vrijeme objavljivanja. Ovaj entitet je u *Many-to-One* vezi s entitetom Person.

Comment		
id	BIGINT	jedinstveni identifikator komentara
content	VARCHAR	sadržaj komentara
posted_on	TIMESTAMP	datum i vrijeme objave komentara
author_id	BIGINT	jedinstveni identifikator autora komentara

**Musician** je entitet koji sadrži informacije o glazbeniku. Sadrži attribute: id glazbenika, oznaku za privatni kalendar te opis glazbenika. Ovaj entitet je u *One-to-One* vezi s entitetima: System\_person, Organizer i Person.

Musician		
id	BIGINT	jedinstveni identifikator glazbenika
bio	VARCHAR	opis glazbenika
public_calendar	BOOLEAN	kalendar glazbenika javan ili ne

**Musician\_instruments** je vezna tablica iz koje se može iščitati koje instrumente svira pojedini glazbenik. Sadrži attribute: id instrumenta te id glazbenika. Ovaj entitet je u *Many-to-One* vezom s entitetima Musician i Instrument.

Musician_instruments		
musician_id	BIGINT	jedinstveni identifikator glazbenika
instruments_id	BIGINT	jedinstveni identifikator instrumenta glazbenika

**Instrument** je entitet koji sadrži informacije o instrumentima. Sadrži attribute: id instrumenta, ime instrumenta te vrstu instrumenta.

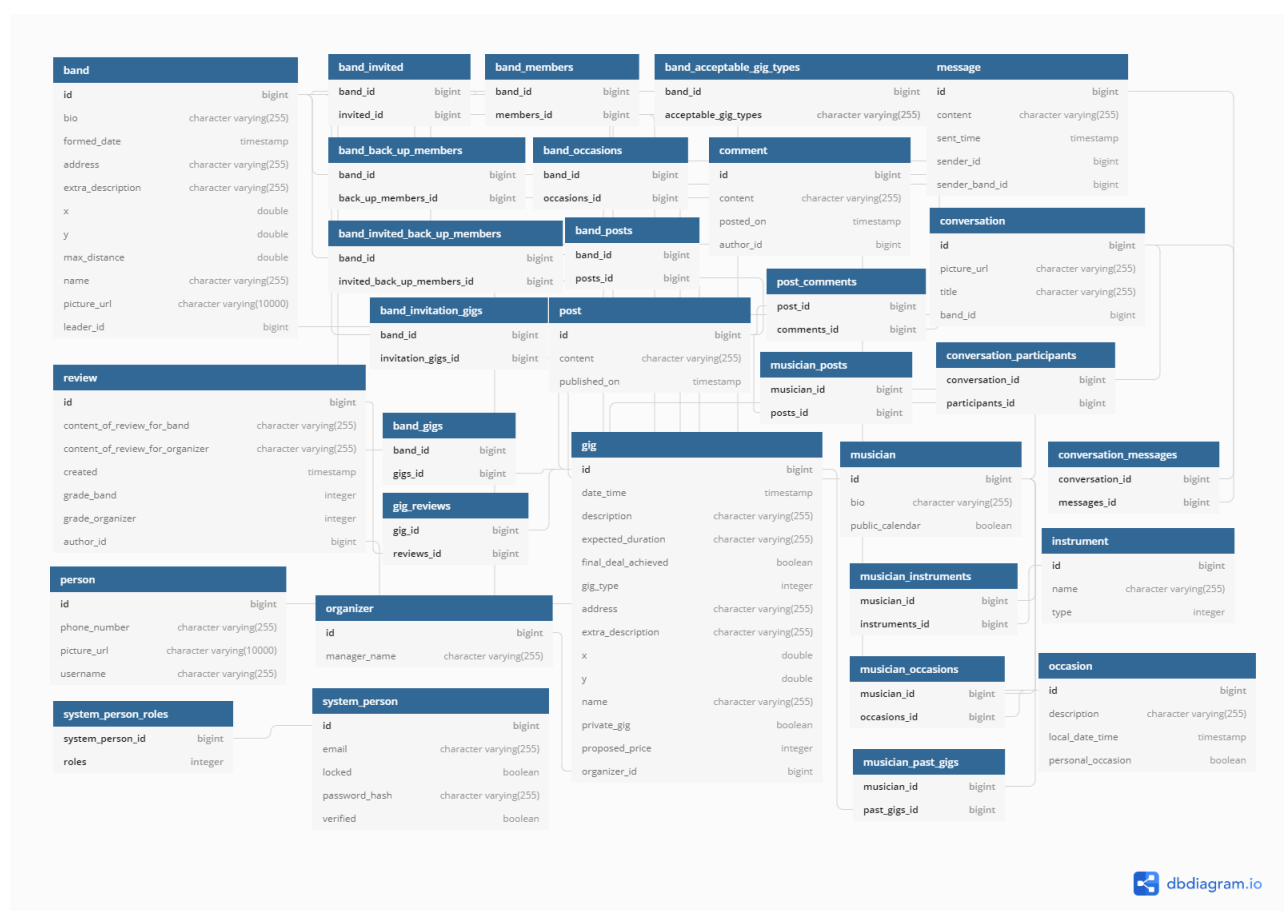
Instrument		
id	BIGINT	jedinstveni identifikator instrumenta
name	VARCHAR	ime instrumenta
type	INT	vrsta instrumenta

**Musician\_past\_gigs** je vezna tablica iz koje se mogu iščitati prošli nastupi glaz-

benika. Sadrži attribute: identifikator glazbenika i identifikator nastupa. Ovaj entitet je u *Many-to-One* vezom s entitetima Musician i Gig.

Musician_past_gigs		
<i>musician_id</i>	BIGINT	jedinstveni identifikator glazbenika
<i>past_gigs_id</i>	BIGINT	jedinstveni identifikator nastupa

### 4.1.2 Dijagram baze podataka

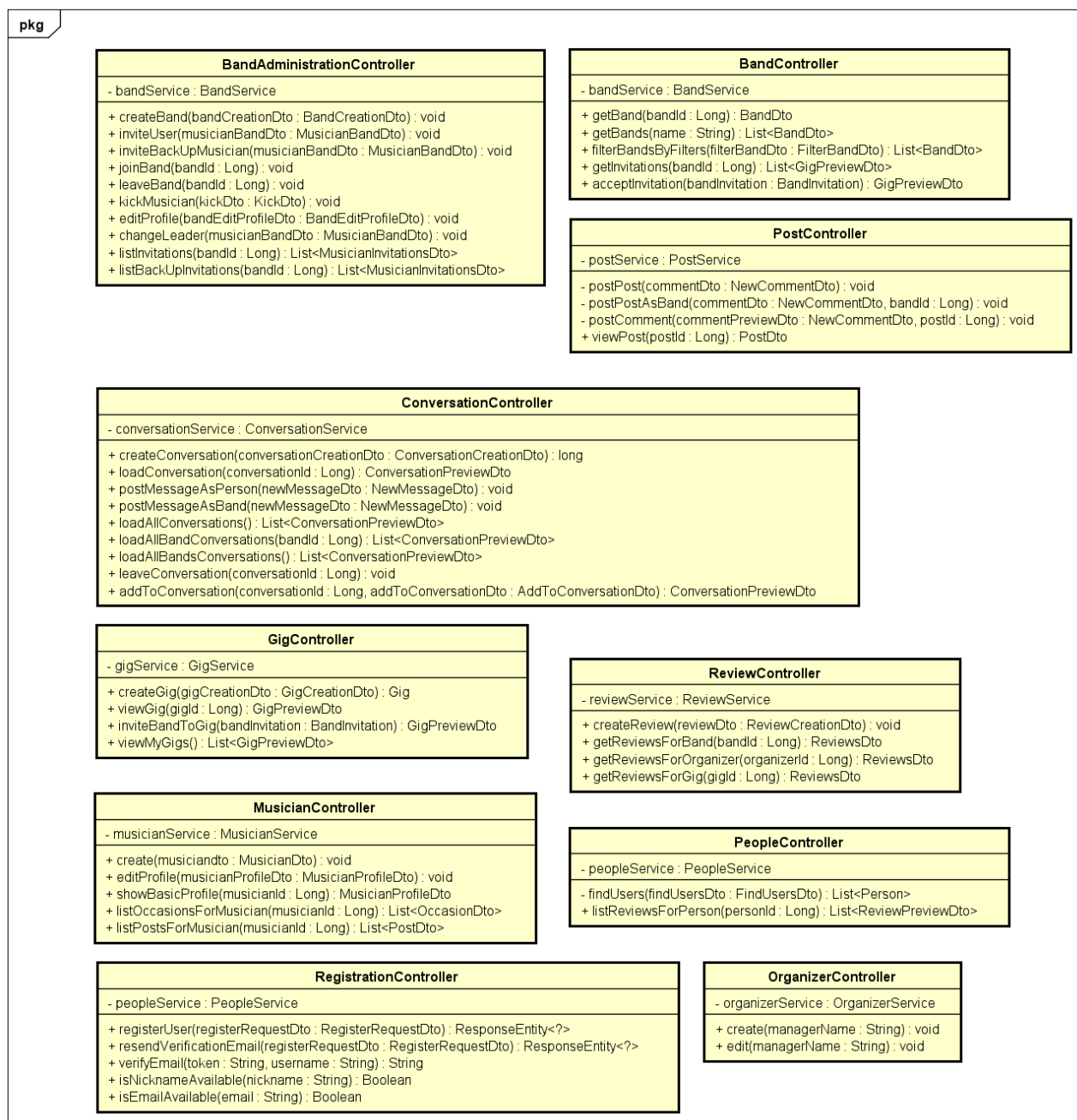


Slika 4.2: Dijagram baze podataka

## 4.2 Dijagram razreda

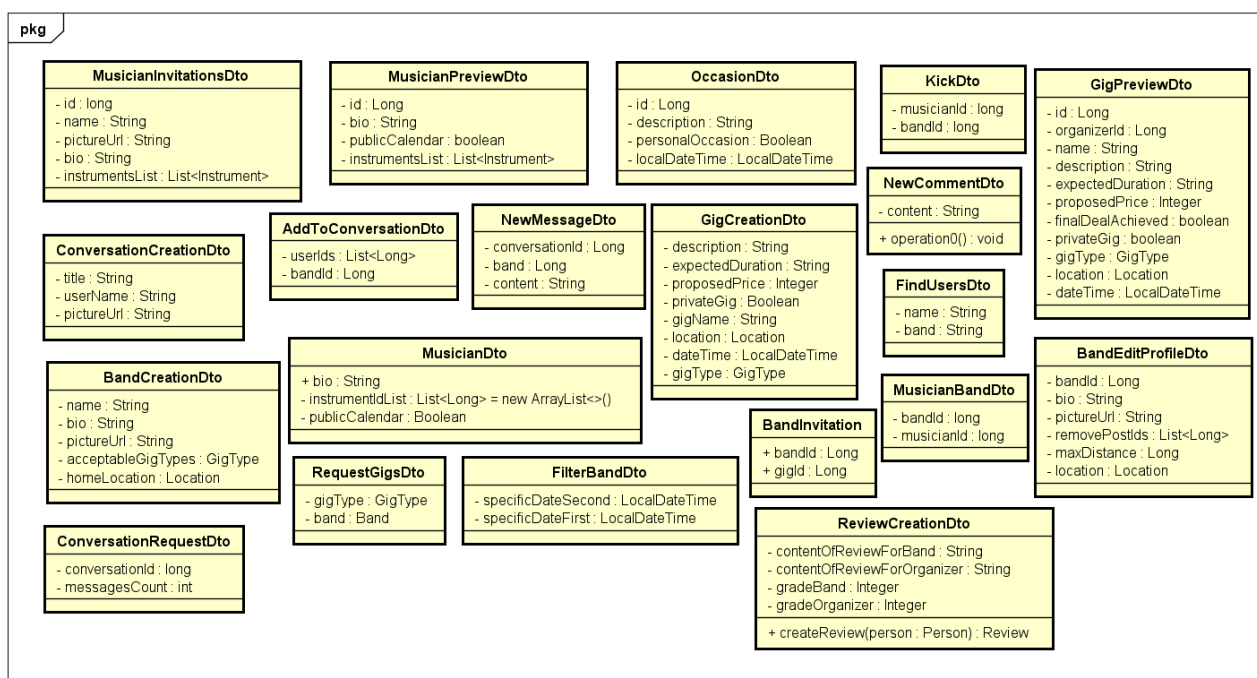
Na slici 4.3 prikazan je Controllers dio backend aplikacije. Controlleri su jedina izložena točka u aplikaciji te nad njima frontend izvršava upite. Svi Controller-i su zaštićeni Spring Security-jem te se prije svakog propuštanja zahtjeva na Controller

autorizira token koji se nalazi u zaglavlju zahtjeva. Jedina iznimka su Controller-i koji služe za registraciju i prijavu. Nakon što se zahtjev autorizira Controller-i pozivaju servisni sloj aplikacije te od njih zahtjevaju da izvrše dio poslovne logike za koju su napisani. Povratni tip Controller-a su DTO-ovi (Data Transfer Objects) prikazani na slici 4.4. Njima se na frontend vraća samo dio informacije prikupljene od servisa.



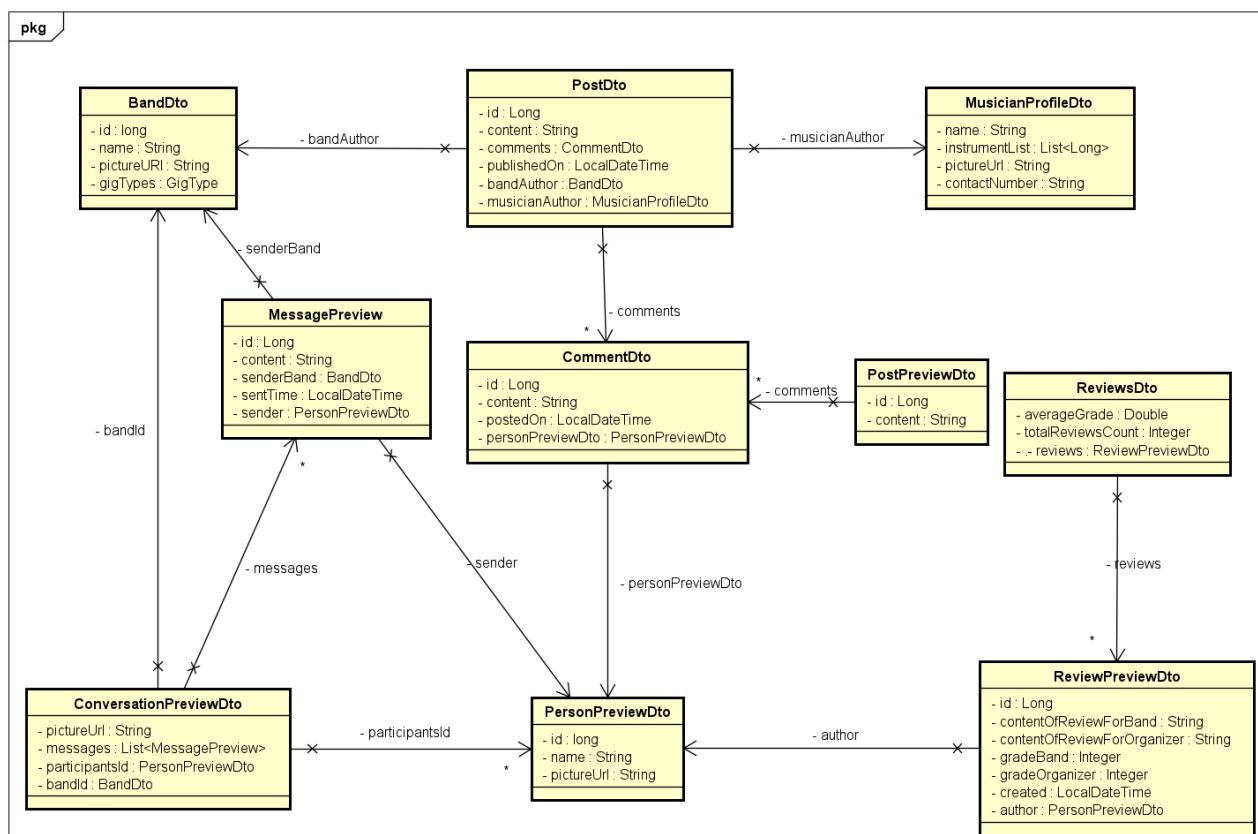
Slika 4.3: Dijagram razreda - dio Controllers

Slika 4.4 prikazuje DTO-ove kojima backend dio aplikacije komunicira s frontendom. DTO-ove smo modelirali tako da izbjegnemo kružne reference objekata koje dobijemo iz baze podataka. Kao posljedica, DTO-ovi sadrže uglavnom primitivne tipove ili neke druge DTO-ove (npr. ConversationPreviewDTO sadrži listu PersonPreviewDTO koji predstavljaju sudionike razgovora). DTO-ove koristimo u oba smjera komunikacije backenda i frontenda.



Slika 4.4: Dijagram razreda - dio Data transfer objects, prvi dio

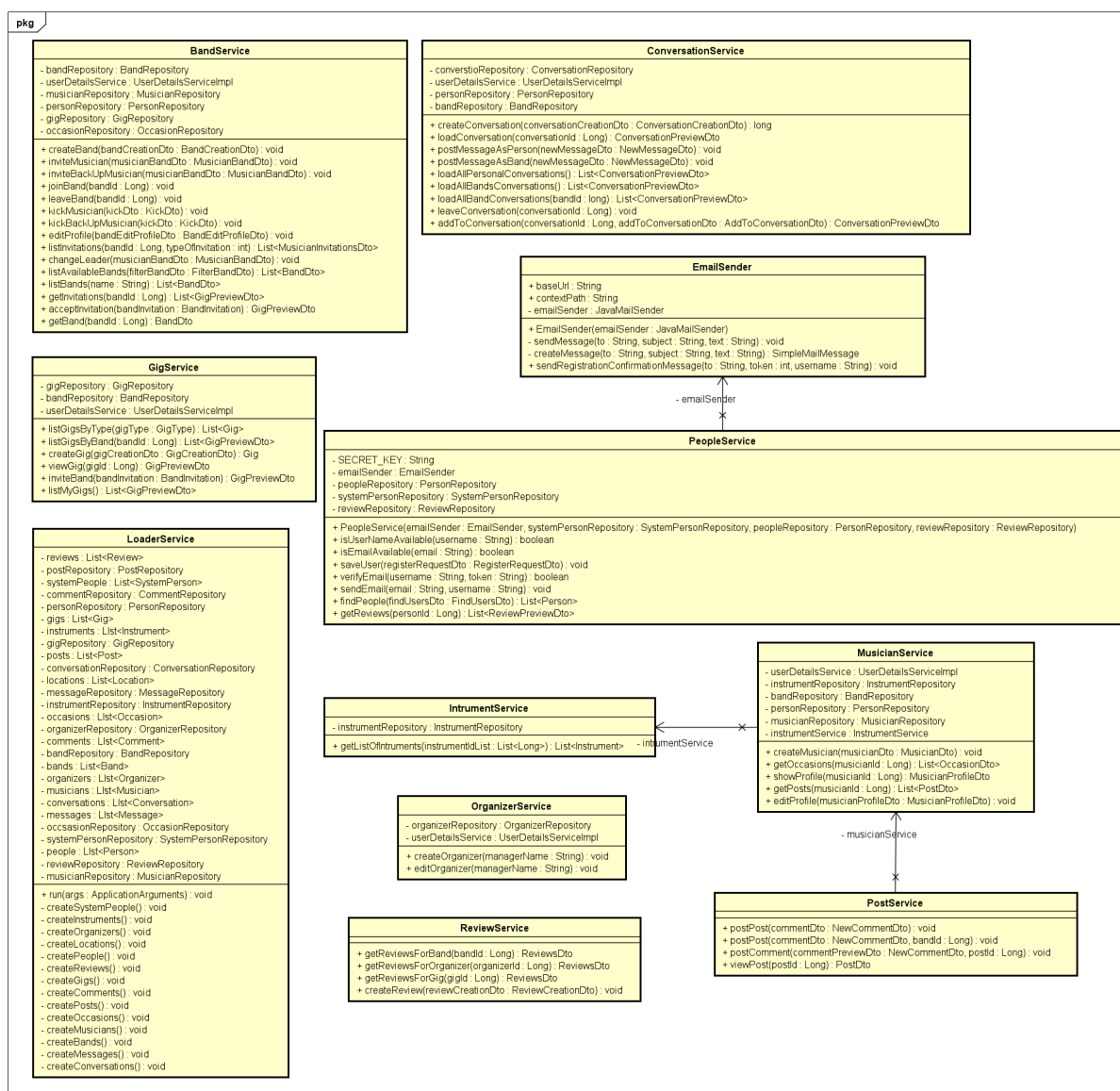
Na ovom dijagramu prikazani su razredi Dto među kojima postoje veze jer pojedini razredi sadrže reference na druge razrede.



Slika 4.5: Dijagram razreda - dio Data transfer objects, drugi dio

Slika 4.5 prikazuje dijagram razreda servisnog sloja. Servisi komuniciraju s repozitorijima koji pristupaju bazi i Controller-ima od kojih dobivaju i kojima vraćaju podatke. Servisi iz baze dobivaju instance objekata koji mogu biti povezani s drugim podacima, itd. i njihov je cilj poštivajući poslovnu logiku obraditi te podatke i kao rezultat svog izvođenja vraćaju DTO-ove. Servisi sadrže svu poslovnu logiku. Gotovo svaki Controller ima pripadajući servis, a svaki je servis logički objedinjen skup funkcija poslovne logike. Iznimke koje se bacaju u servisima omataju se GigerException-om koji nasljeđuje RuntimeException. Ako se pogreška propagira iz sustava, možemo provjeriti njezin tip te ako je zamotana u GigerException, znači da je to iznimka koju smo očekivali, u protivnom je došlo do neočekivane situacije u sustavu.

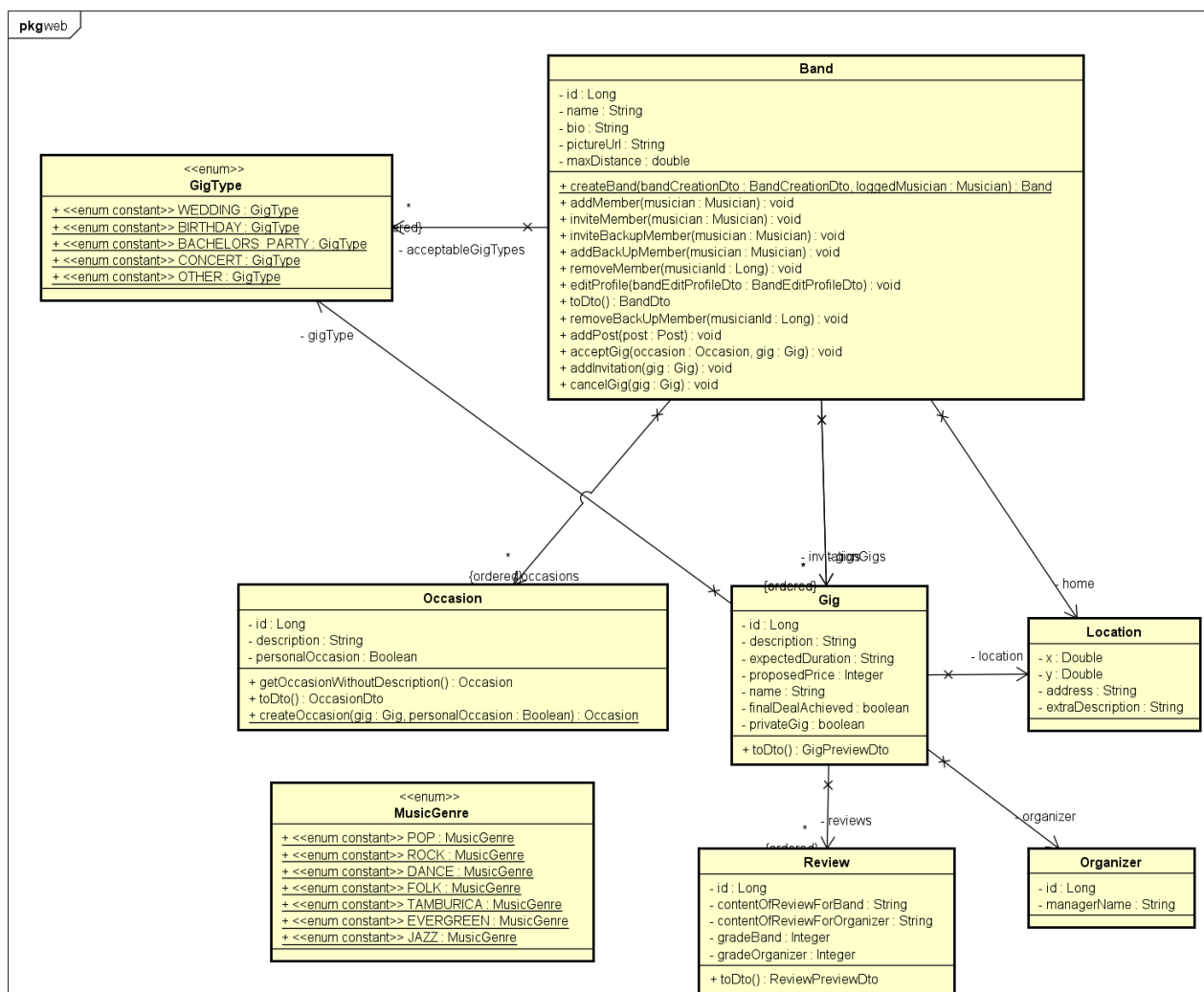




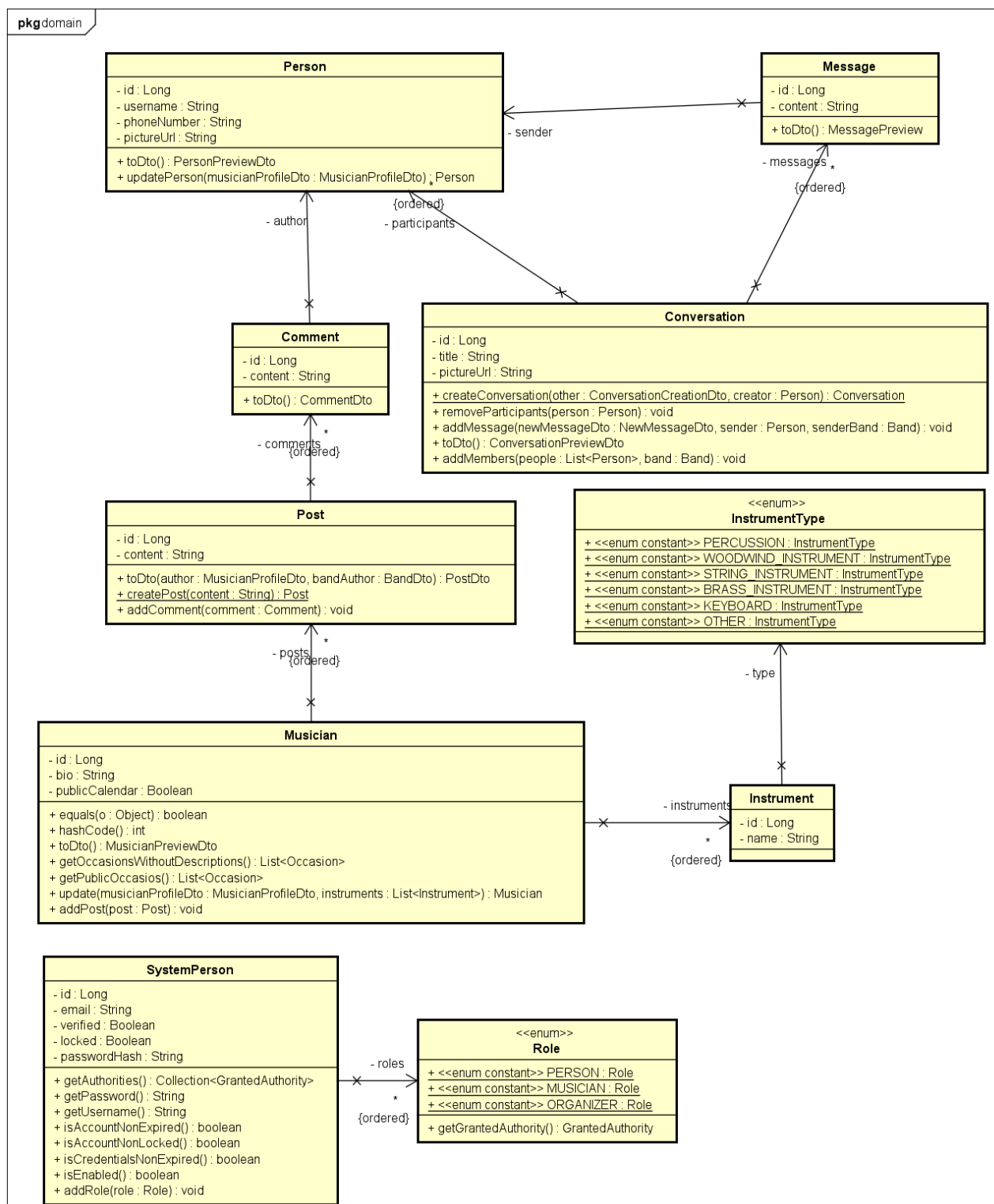
Slika 4.6: Dijagram razreda - dio Service

Slika 4.6 prikazuje razrede koji predstavljaju entitete entitetsko - relacijskog modela u bazi podataka. Razred Band predstavlja bend kojim upravlja glazbenik koji je postavljen za voditelja benda. Razred SystemPerson predstavlja razred u kojem su objedinjeni svi sustavski podaci vezani za osobu kao što su: email, hash lozinke i id. Gig predstavlja nastup nekog benda koji organizira određeni organizator i pri tom enkapsulira sve logističke informacije o tom nastupu. Razred Conversation omogućuje komunikaciju između aktera glazbenika i organizatora. Glazbenik je opisan razredom Musician, dok je organizator opisan razredom Organizer. Razredi Post, Comment, Location, Instrument, Message služe za enkapsulaciju informacija

kako bi dopunili razrede poput Band, Conversation i Musician. GigType i Role predstavljaju enumeracije.



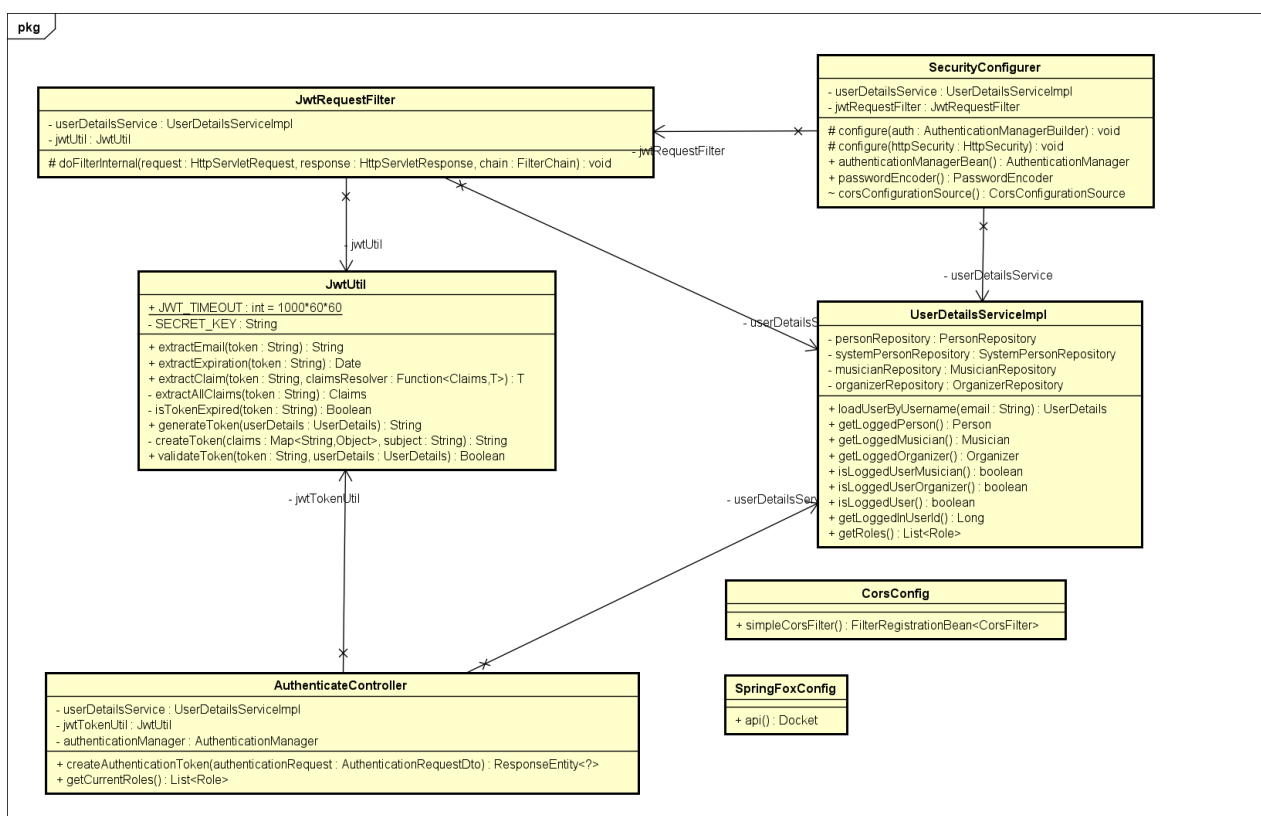
Slika 4.7: Dijagram razreda - razredi entiteta 1



Slika 4.8: Dijagram razreda - razredi entiteta 2

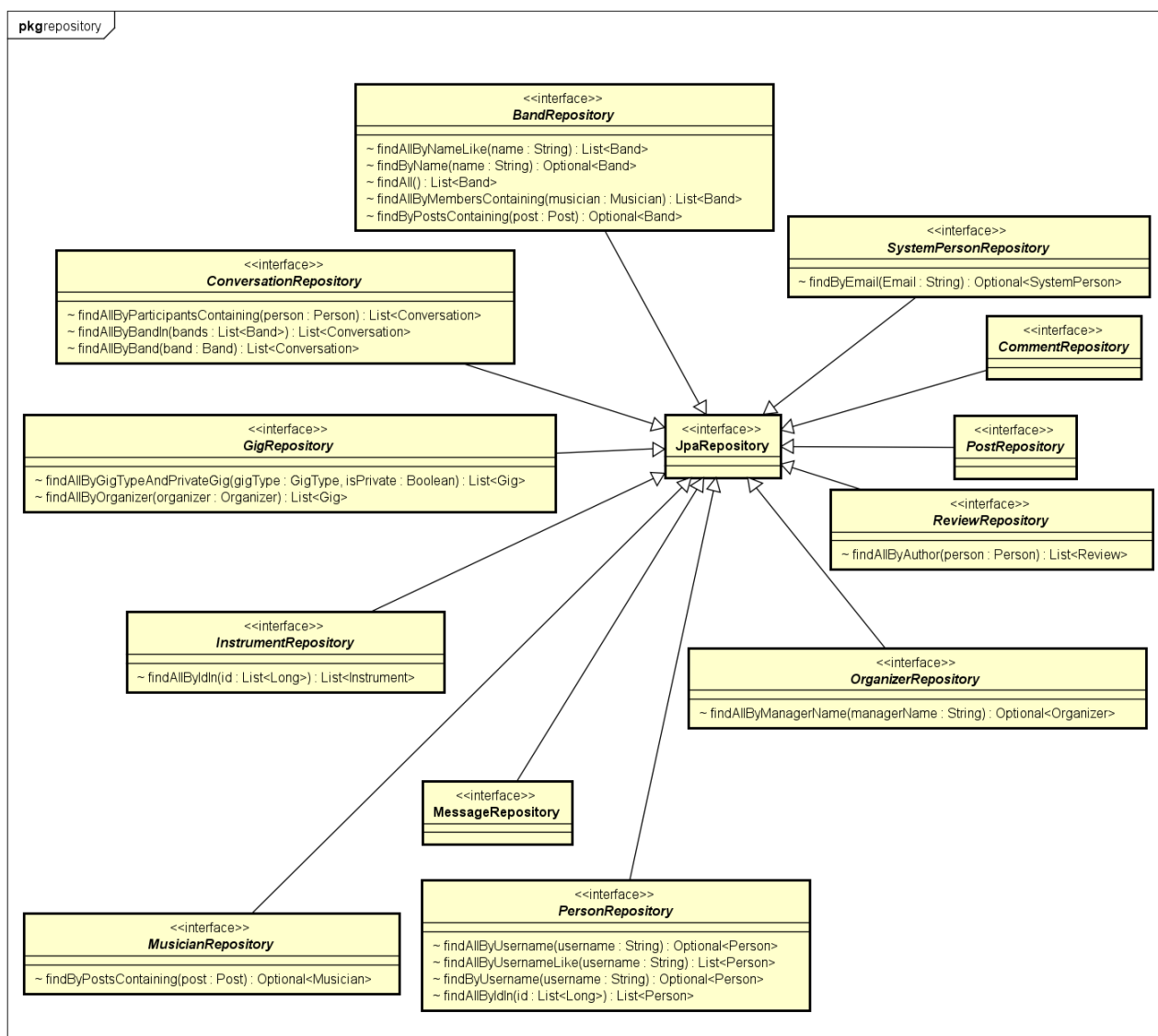
Na slici 4.7 prikazan je dijagram razreda zaduženih za Spring Security. Ulazna točka za autorizaciju zahtjeva je definirana u JwtRequestFilteru koji poziva User-

DetailsServiceImpl da provjeri postoji li u bazi podataka zapis s akreditacijama koje se dekodiraju iz jwt tokena. AuthenticateController je zadužen za pružanje jwt tokena ukoliko u bazi pronađe zapis s odgovarajućim vrijednostima. Aplikacija ne pamti sesiju niti stanje (STATELESS) tako da korisnik prilikom postavljanja zahtjeva mora dostaviti svoj jwt token putem kojeg se autentificira i autorizira. JwtUtil klasa je puna pomoćnih metoda za baratanje jwt tokenom, a UserDetailsServiceImpl je posrednik između baze korisnika i AuthenticateControllera. Vežano uz autorizaciju, modelirana su tri DTO objekta koji definiraju objekte koje backend prima i daje kao zahtjev ili odgovor.



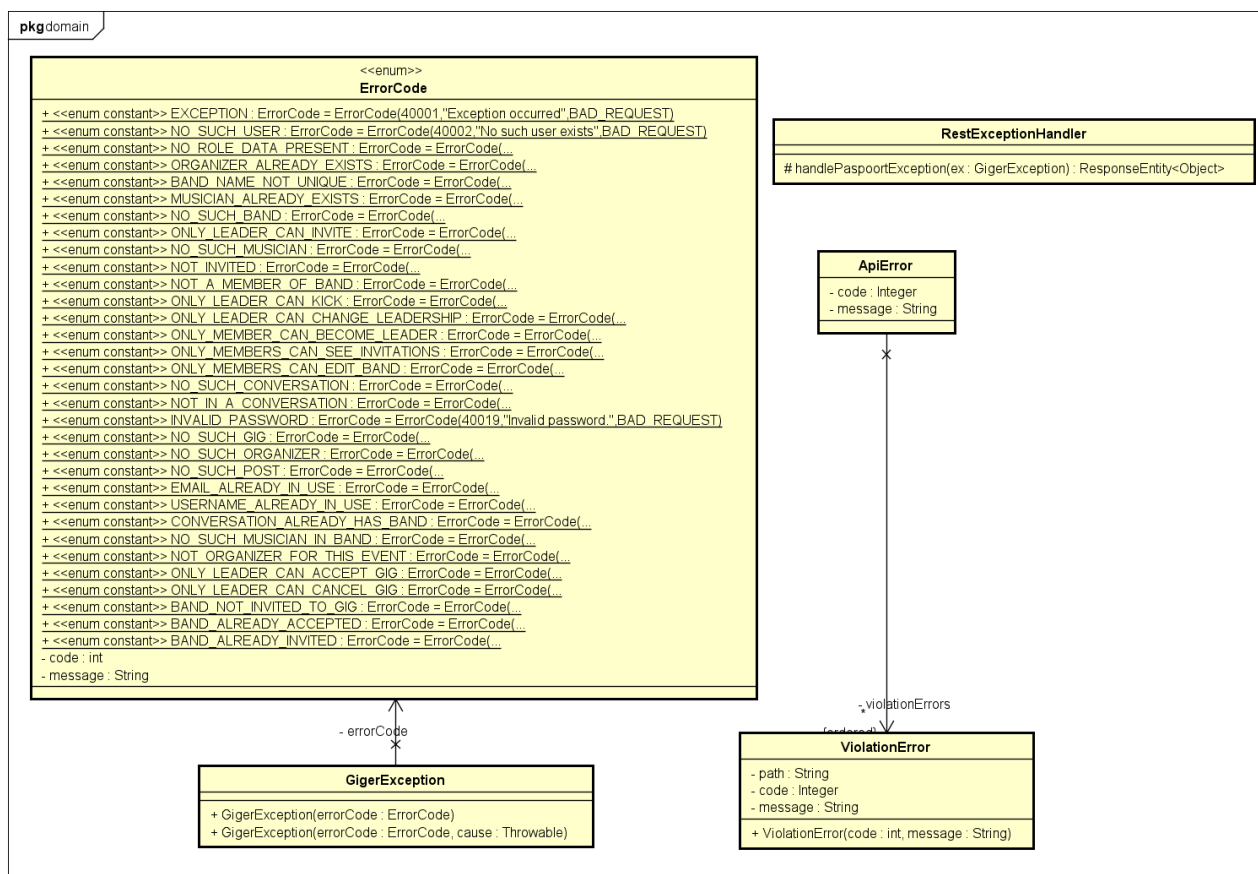
Slika 4.9: Dijagram razreda - security

Za dohvat podataka iz baze podataka koristimo Jakarta Persistence. Jakarta Persistence je specifikacija programskog sučelja Java aplikacije koja opisuje upravljanje relacijskim podacima u aplikaciji. U ovoj aplikaciji za svaki entitet definiran je zasebni repozitorij koji nasljeđuje JpaRepositoryj. Jpa Repository sadrži osnovne metode za dohvat podataka, a Spring Boot omogućuje programeru da specificiranjem samo imena metode dobije implementaciju iste na korištenje.



Slika 4.10: Dijagram repozitorija - repository

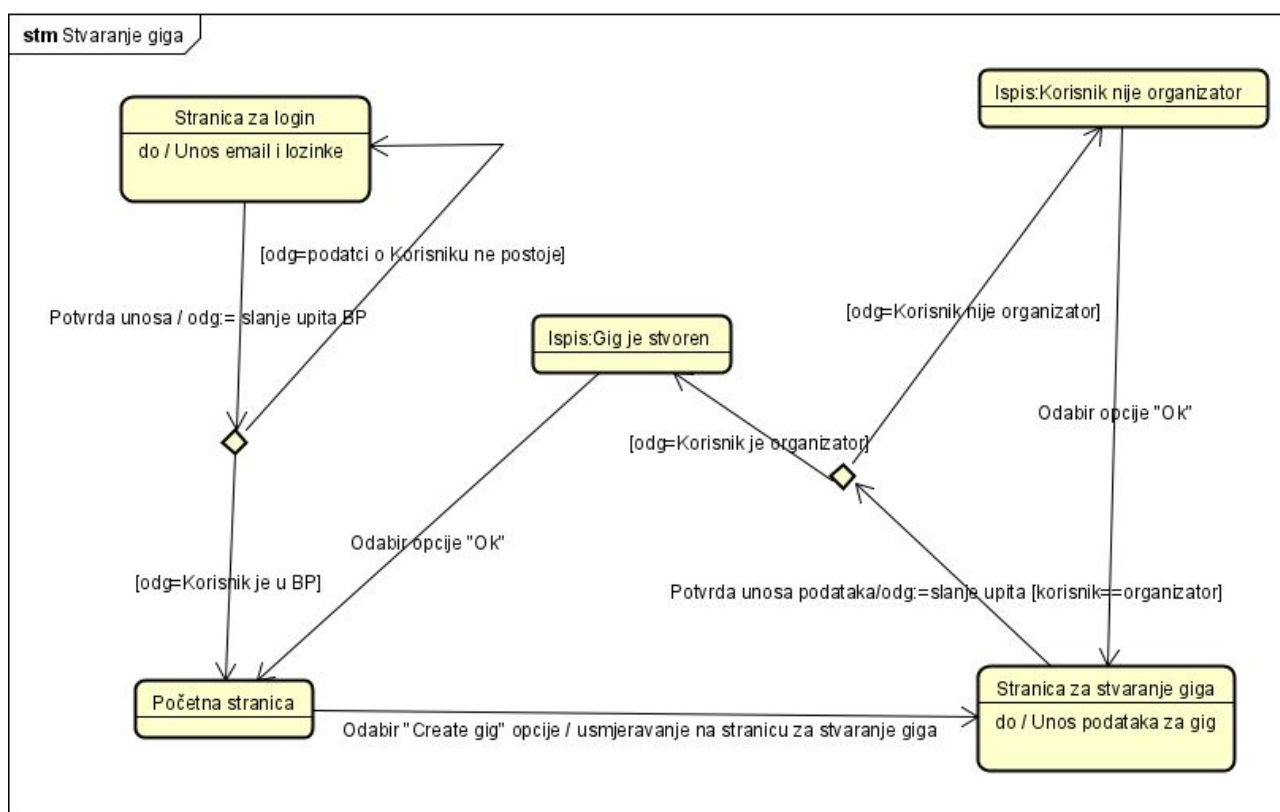
Ovim dijagramom prikazane su moguće pogreške kao i iznimke. Pogreške se u ovom slučaju nalaze u Enumu Errorcode i služe za zaustavljanje operacija čijim izvođenjem se krši pravo pristupa ili se pokušava izvesti nemoguća akcija. Sve ostale logičke iznimke omotavaju se u iznimu GigerException. Za reprezentacijsko stanje prijenosa uvedena je iznimka RestExceptionHandler.



Slika 4.11: Dijagram razreda - errors

### 4.3 Dijagram stanja

Sljedeći dijagram stanja prikazuje stvaranje giga. Nakon uspješnog stvaranja giga, isti i dalje nije viđen pod opcijom "View public gigs". Da bi ostali korisnici mogli vidjeti taj gig, on prvo mora biti finaliziran. To znači da organizator mora pozvati bend u svoj gig te nakon što bend prihvati nastup, gig postaje finaliziran (javan).



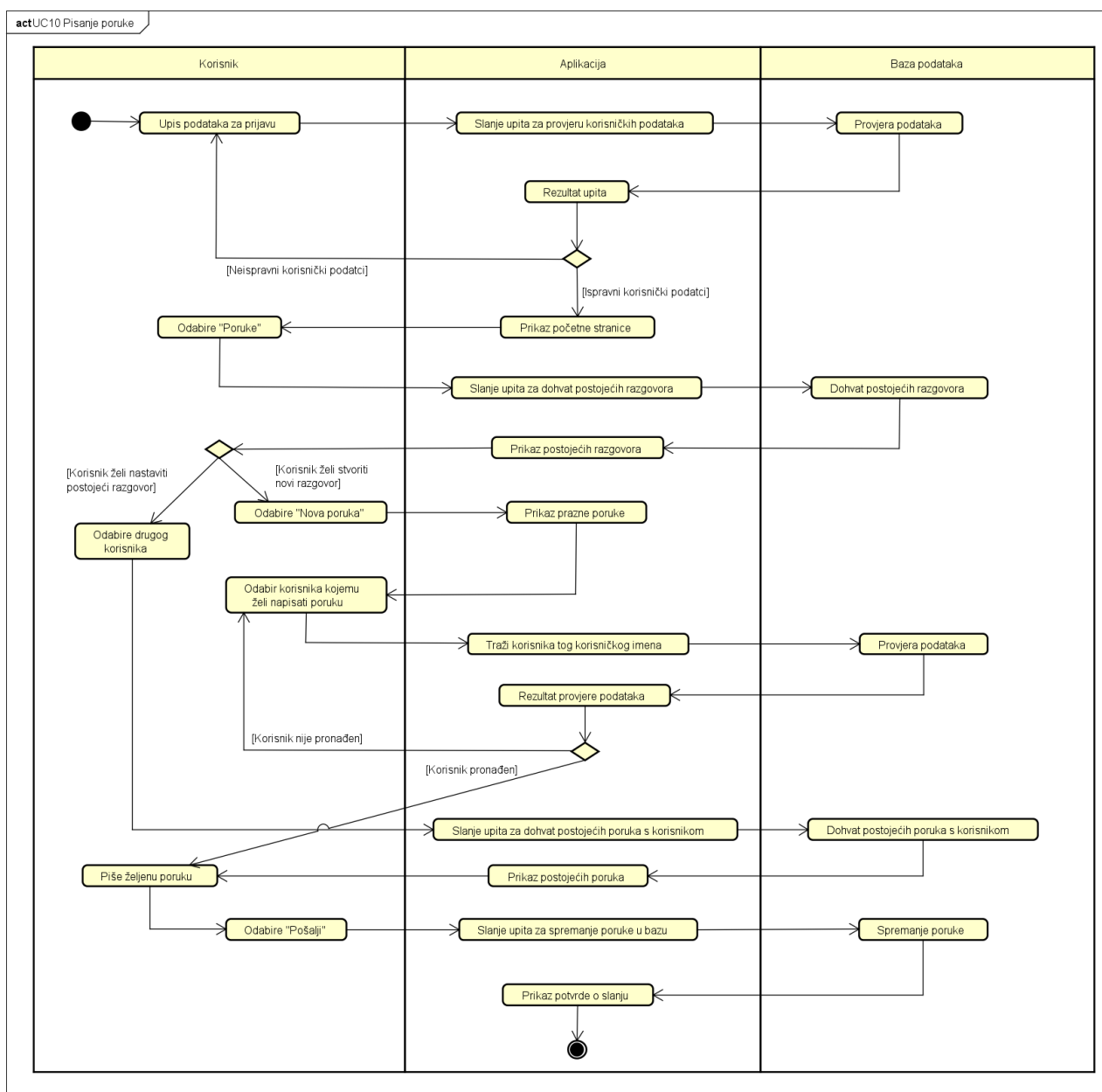
Slika 4.12: Dijagram stanja - stvaranje giga

## 4.4 Dijagram aktivnosti

UML dijagram aktivnosti prikazuje detaljan tijek obrazaca uporabe, njegovu proceduralnu logiku te sudionike pojedinoga obrasca uporabe.

Na slici 4.11 prikazan je dijagram aktivnosti pisanja poruke. Korisnik se prijavi u sustav nakon čega odabire "Poruke". Nakon toga se iz baze dohvaćaju već postojeće poruke korisnika te mu ih aplikacija prikaže. Korisnik tada može ili započeti novi razgovor, ili odabrati postojeći. Ukoliko želi stvoriti novi razgovor, odabire opciju "Nova poruka" te mu aplikacija prikazuje praznu poruku. Nakon toga, odabire korisnika kojemu želi napisati poruku sve dok se on ne pronađe uspješno u bazi podataka. Ukoliko korisnik želi razgovarati s korisnikom s kojim je već vodio razgovor, odabire razgovor s tim korisnikom. Aplikacija iz baze dohvaća prošle poruke s odabranim korisnikom. U oba opisana slučaja, korisnik piše željenu poruku, odabire "Pošalji" nakon čega se poruka sprema u bazu te aplikacija prikaže potvrdu o slanju.

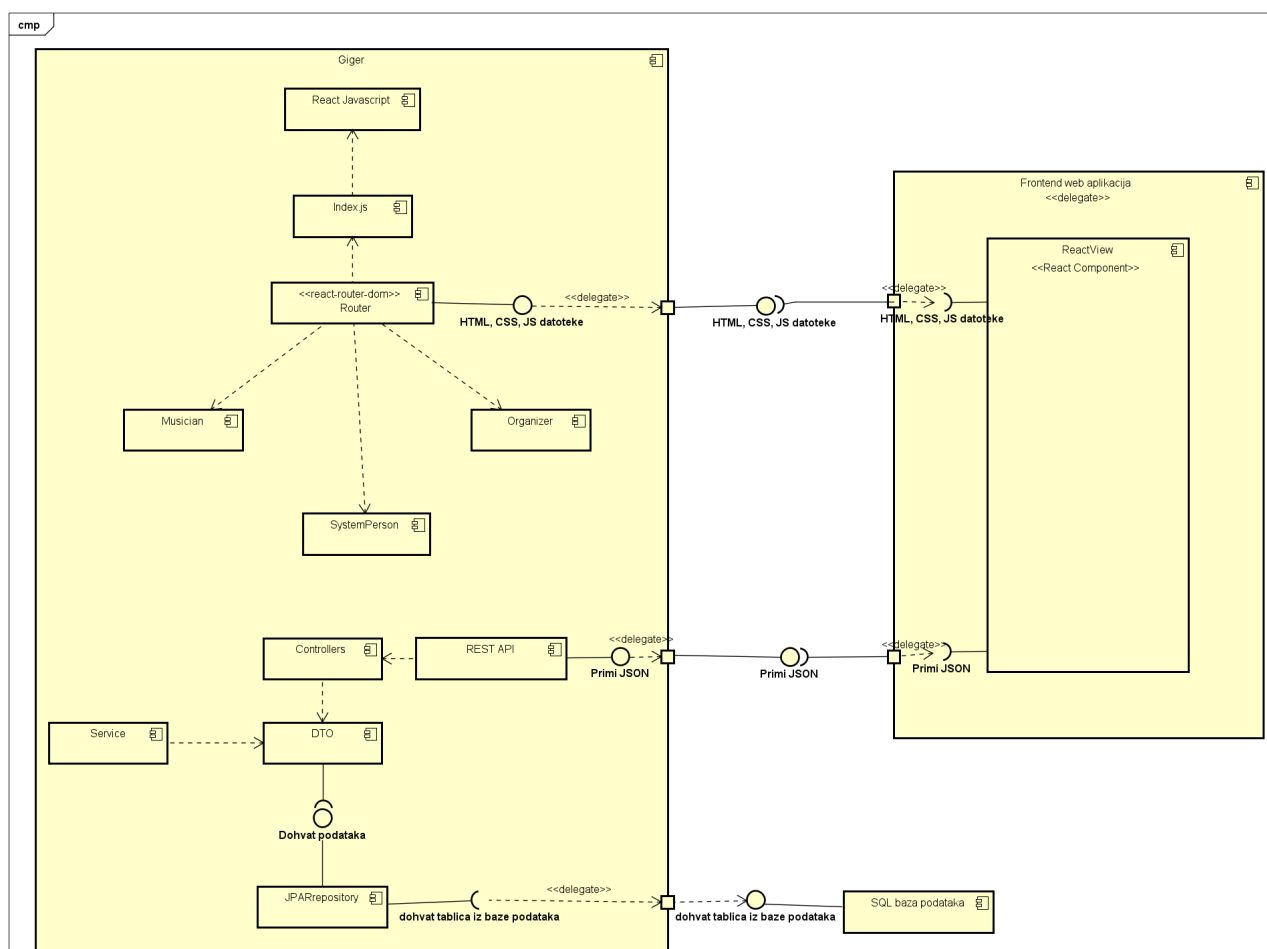




Slika 4.13: Dijagram aktivnosti

## 4.5 Dijagram komponenti

Dijagram komponenti prikazuje zavisnost programskih komponenata i fizičku strukturu koda u terminima kodnih komponenti. Graf komponenti povezan je vezama ovisnosti kako bi se omogućila lakša analiza reakcije ostalih komponenti na promjene u jednoj komponenti. Dohvat podataka iz SQL Postgres baze podataka vrši se sučeljima koja nasljeđuju JPA Repository pomoću SQL upita. Podaci koji pristižu iz baze podataka stavljaju se u DTO objekte koji se koriste u ostatku aplikacije. REST API šalje podatke iz backend aplikacije u obliku JSON-a frontend aplikaciji. Frontend aplikacija se sastoji od Javascript datoteka koje su grupirane u ovisnosti koji aktor im pristupa. Router je komponenta koja određuje koje datoteke se prikazuju klijentu. Sustavu se pristupa preko sučelja preko kojega mu se šalju HTML i JS datoteke iz frontend aplikacije.



Slika 4.14: Dijagram komponenti

## 5. Implementacija i korisničko sučelje

### 5.1 Korištene tehnologije i alati

Backend		
PostgreSQL	<a href="https://www.postgresql.org/">https://www.postgresql.org/</a>	Objektno-relacijska baza podataka
Java 11	<a href="https://www.oracle.com">https://www.oracle.com</a>	Programski jezik u kojem je napisan backend dio aplikacije
Java Spring Boot	<a href="https://spring.io/projects/spring-boot">https://spring.io/projects/spring-boot</a>	Razvojni okvir
Spring Web MVC	<a href="https://docs.spring.io/spring/">https://docs.spring.io/spring/</a>	Web framework za rukovanje zahtjevima
Spring Security	<a href="https://spring.io/projects/spring-security">https://spring.io/projects/spring-security</a>	Moćan i vrlo prilagodljiv okvir za provjeru autentičnosti i kontrolu pristupa
Lombok	<a href="https://projectlombok.org/">https://projectlombok.org/</a>	Java library za pregledniji kod

Frontend		
React	<a href="https://reactjs.org/">https://reactjs.org/</a>	JavaScript library za izgradnju sučelja
Ant Design	<a href="https://ant.design/">https://ant.design/</a>	Design library sa komponentama za lakšu izgradnju korisničkog sučelja
NPM	<a href="https://www.npmjs.com/">https://www.npmjs.com/</a>	Upravitelj paketa za programski jezik JavaScript
OpenCage Geocoder	<a href="https://opencagedata.com/">https://opencagedata.com/</a>	API za dohvaćanje koordinata iz adrese

Komunikacija		
Slack	<a href="https://slack.com/intl/en-hr/">https://slack.com/intl/en-hr/</a>	Platforma koju smo koristili za lakšu komunikaciju
Trello	<a href="https://trello.com/en">https://trello.com/en</a>	Alat koji nam je olakšao zajednički rad i raspoređivanje projektnih zadataka

## 5.2 Ispitivanje programskog rješenja

### 5.2.1 Ispitivanje komponenti

Da bismo testirali backend, napravili smo dvije vrste testova. JUnit testove za testiranje servisa te integracijske testove koje je najlakše opisati kao automatizirane Postman zahtjeve.

Servisi su testirani na način da ih gledamo kao crne kutije koje za određeni ulaz trebaju odraditi određene akcije ili vratiti određene objekte. Nomenklatura testovi servisa (`BandServiceTest`) slijedi pravilo `given.when.then`. Naprimjer, ako testiramo metodu naziva `createMusician`, pretpostavljajući da on još ne postoji i očekujući da se kreira glazbenik u bazi, naziv testa bio bi `noSuchMusician_createMusician_createAndPersistateMusician()`.

Pomoću Mockito frameworka u testovima mockamo (oponašamo) ulaze te na taj način kontroliramo ulaze i okolinu metode koju testiramo. Drugim riječima, kada radimo test za neku komponentu ili servis, pretpostavljamo da su svi ulazi dobri i ne zanima nas utjecaj naše komponente na drugu komponentu, već samo direktni izlaz. Na taj način dobivamo testove koji su međusobno nepovezani i koji definiraju željeno ponašanje aplikacije. Ukoliko u daljnjem razvoju neki od razvojnih programera promijeni neko ponašanje koje ima utjecaj na druge komponente, pravilno napisani testovi trebali bi pasti i upozoriti ga da će se njegova promjena propagirati dublje u aplikaciju. Testovi koji su pisani ciljano na pojedine komponente sustava u kontroliranim uvjetima prilikom izvođenja precizno ukazuju na vjerojatan izvor pogreške. Na primjer, ako promijenimo implementaciju kreiranja glazbenika te on u trenutku kreiranja ne dobije id, samo testovi koji provjeravaju parametre nakon inicijalizacije bi trebali pasti, a ne svi testovi koji se u nekom trenutku pozivaju na tu funkcionalnost.

Svaki napisan test odijeljen je u tri cjeline, a to su Arrange, Act, Assert. U prvom dijelu uređujemo i mockamo ulaze u testirajuću komponentu, na prije opisan način. U drugom dijelu testa poziva se akcija ili niz akcija čije djelovanje želimo provjeriti, a u trećem dijelu testa provjeravamo jesu li posljedice izvršavanja drugoga dijela testa u skladu s očekivanjima.

Na slici 5.1 nalazi se primjer unit testa. Linije 380-399 su Arrange, linija 401 je Act, dok su linije 404-408 Assert dio testa.

```
@Test
public void listInvitations_members() {
    Musician musician = mock(Musician.class);
    Musician musicianOther = mock(Musician.class);

    Band band = new Band();
    band.setMembers(newArrayList(musicianOther));
    band.setLeader(musician);
    band.setPosts(newArrayList());
    band.setInvited(newArrayList(musicianOther));

    Person person = new Person();
    person.setUsername("Mickey");
    person.setPictureUrl("picture url");

    when(userDetailsService.getLoggedMusician()).thenReturn(musician);
    when(bandRepository.findById(1L)).thenReturn(of(band));
    when(musicianOther.getId()).thenReturn(2L);
    when(musicianOther.getBio()).thenReturn("bio");
    when(musicianOther.getInstruments()).thenReturn(newArrayList());
    when(personRepository.getOne(id: 2L)).thenReturn(person);

    var results = bandService.listInvitations( bandId: 1L, typeOfInvitation: 0);

    assertEquals( expected: 1, results.size());
    assertEquals( expected: 2, results.get(0).getId());
    assertEquals( expected: "picture url", results.get(0).getPictureUrl());
    assertEquals( expected: "Mickey", results.get(0).getName());
    assertEquals( expected: 0, results.get(0).getInstrumentList().size());
}
```

Slika 5.1: Primjer JUnit testa

Pošto je pisanje testova često i iscrpnije od pisanja implementacije (BandService ima 210 linija koda, dok BandServiceTest koji ni ne testira baš sve metode u njemu ima 558), za ovaj projekt nismo radili TDD (test driven development) već smo naknadno radili testove za postojeću implementaciju kako bismo potvrdili implementaciju i programski dokumentirali očekivano ponašanje.

Uz dodatak BandService testovima postoji i nekoliko testova u EmailSenderTest koji pokazuju kako testirati komponentu čiju implementaciju ne znamo.

Uz tridesetak JUnit testova čija je glavna prednost brzo izvođenje, napisali smo desetak integracijskih testova. Kao što smo već spomenuli, integracijski testovi slični su ručnom pregledavanju u Postmanu. Svaki integracijski test pokreće aplikaciju ispočetka tako da su mu stanje baze i aplikacije (kontekst) jednaki kao kad se

pokrene aplikacija. U ovim testovima koristimo MockMvc kako bismo simulirali http request kojemu moramo odrediti metodu (GET, POST) te dodati pripadajuća zaglavlja. U ovom testu verificiramo je li ono što je vratila metoda jednako onome što se očekivalo (najčešće usporedbe DTO-ova).



```
@Test
@DirtiesContext
public void login_allOk_success() throws Exception {
    MockHttpServletResponse response = mockMvc.perform(
        post(AUTHENTICATE)
            .contentType(APPLICATION_JSON_VALUE)
            .with(csrf())
            .content(readJsonFile(JSON_GROUP, JSON_PATH_LOGIN)))
        .andExpect(status().isOk())
        .andReturn()
        .getResponse();

    AuthenticationResponseDto authenticationResponseDto = objectMapper.readValue(response.getContentAsString(), AuthenticationResponseDto.class);

    assertEquals("expected: \"john.doe@giger.com\", jwtUtil.extractEmail(authenticationResponseDto.getToken());
    assertTrue(jwtUtil.extractExpiration(authenticationResponseDto.getToken()).after(new Date()));
}
```

Slika 5.2: Primjer integracijskog testa

Tehnike testiranja programske potpore nadahnute su knjigom Test-Driven Development Kenta Becka.

## 5.2.2 Ispitivanje sustava

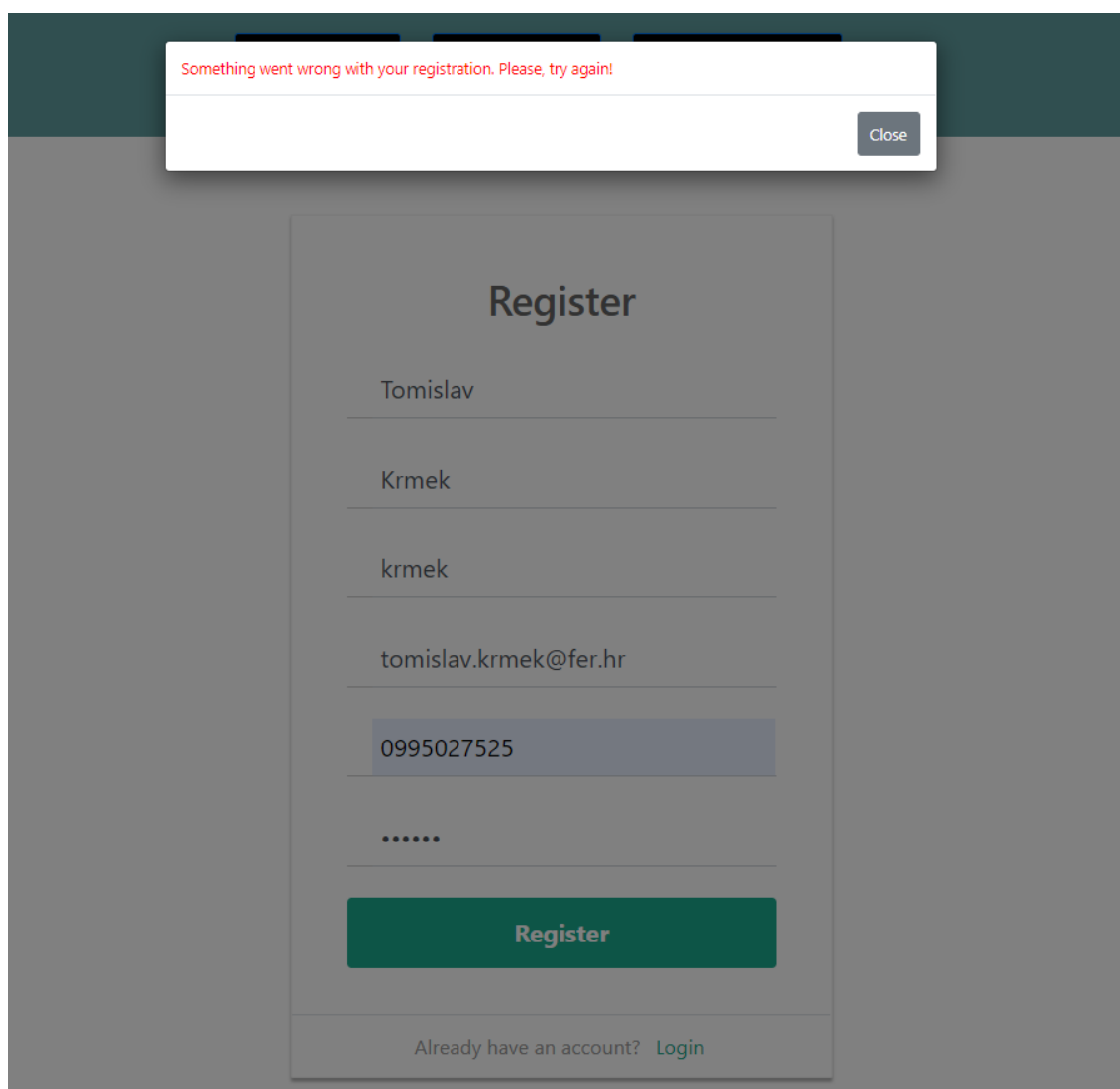
Za potrebe ispitivanja sustava, koristili smo Selenium IDE.

### Test 1: Registracija korisnika

**Očekivano:** Klikom na gumb *Register* otvara se forma s potrebnim poljima za registraciju: imena, prezimena, korisničkog imena, e-mail adrese, kontakt telefona te korisničke lozinke. Ukoliko su sva polja ispravna korisnik će se pokušati registrirati u bazu, u protivnom će sustav dojaviti grešku.

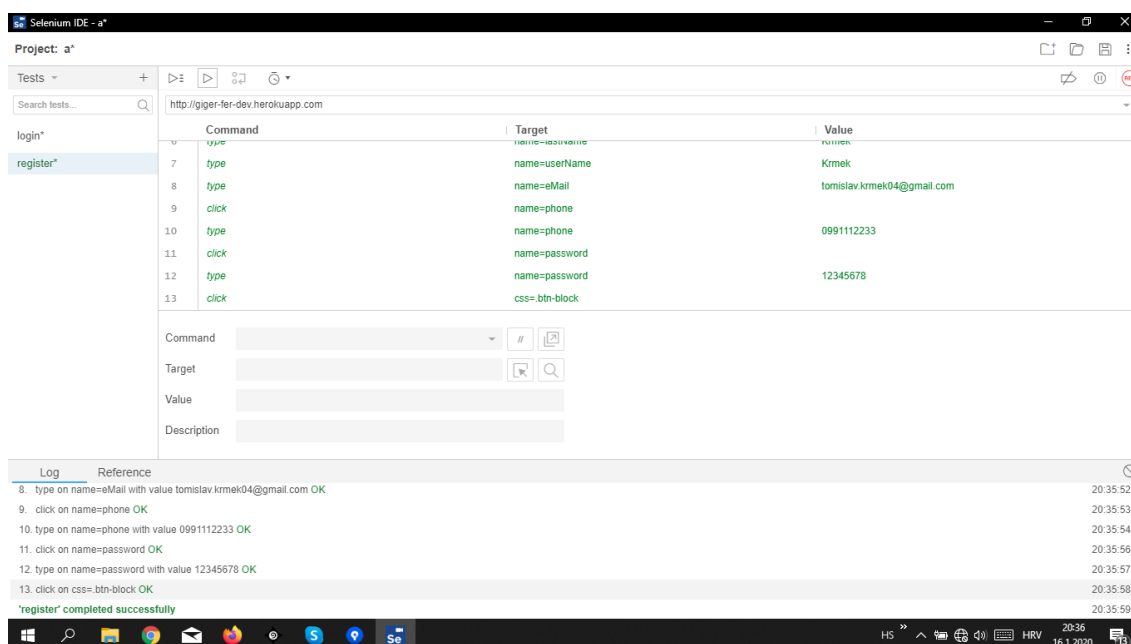
**Rezultat:** Korisnik je uspješno registriran u bazu podataka ili je sustav dojavio grešku.

Na slici 5.3 prikazana je poruka koja se prikaže prilikom neispravne registracije. Slika 5.4 prikazuje kako uspješna registracija izgleda u Seleniumu. U polju *Value* se vide uneseni podatci za koje se testira. U *Log* dijelu se vide dijelovi testa te poruka jesu li prošli ili nisu. Dok se na kraju ispiše poruka o uspješnosti testa.



Slika 5.3: Prikaz poruke za neispravnu registraciju





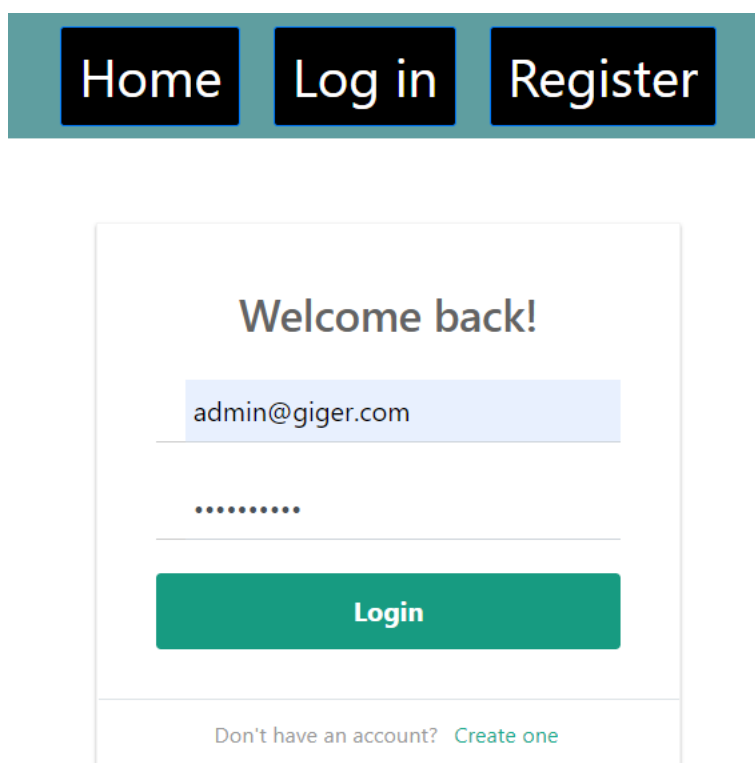
Slika 5.4: Selenium test koji prikazuje ispravnu registraciju

## Test 2: Prijava korisnika u aplikaciju

**Očekivano:** Klikom na gumb *Login* otvara se forma s poljima: e-mail korisnika i lozinka. U slučaju ispravno unesenih polja i podataka, korisnik se prijavljuje u aplikaciju. U protivnom, aplikacija dojavljuje grešku.

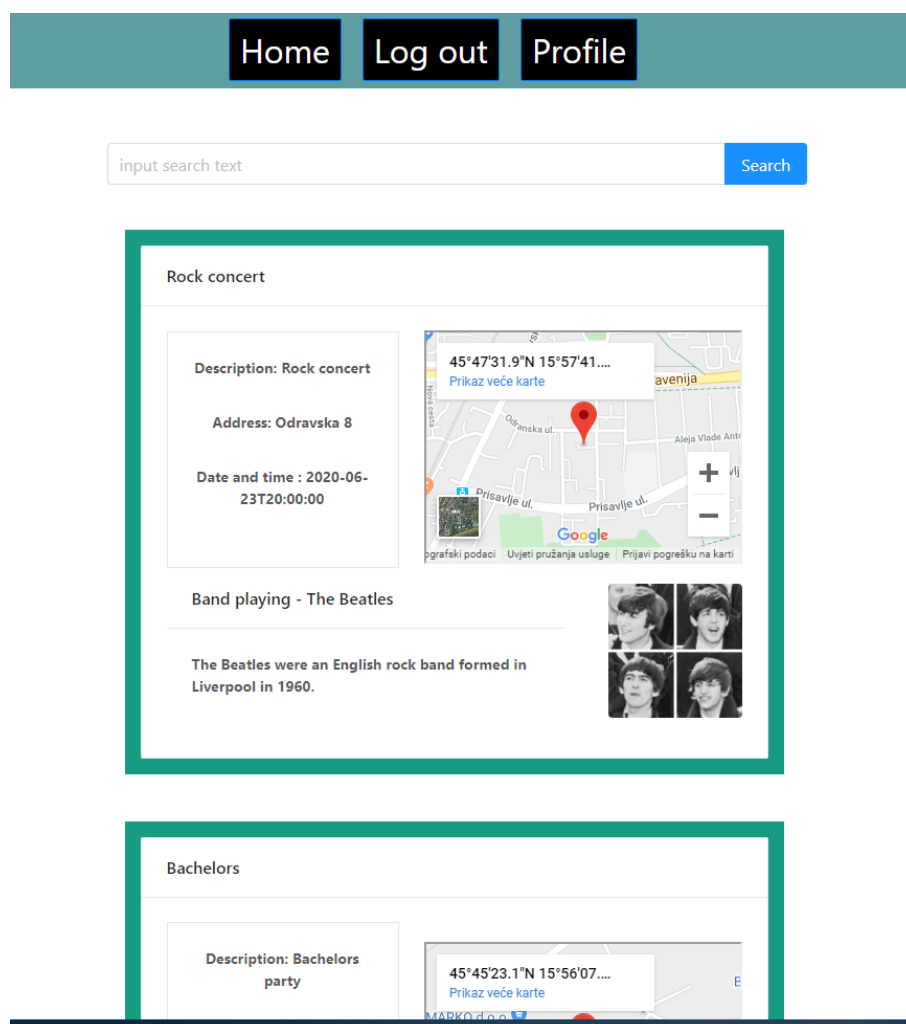
**Rezultat:** Korisnik je prijavljen u aplikaciju ili je dojavljena greška.

Na slici 5.5 prikazana je prijava u aplikaciju dok je na slici 5.6 prikaz javnih nastupa koji se prikaže nakon uspješne prijave. Slika 5.7 prikazuje kako to izgleda i u Seleniumu.

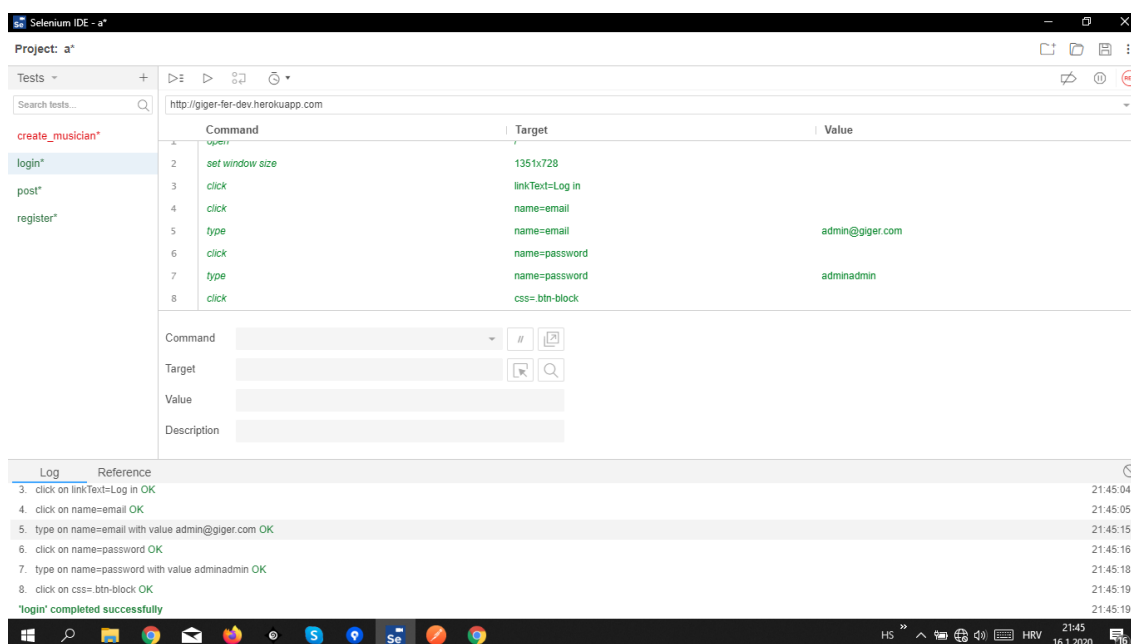


The image shows a login form interface. At the top, there is a teal header bar containing three black buttons with white text: 'Home', 'Log in', and 'Register'. Below this, the main content area is white. It starts with the text 'Welcome back!' in a bold, dark grey font. Underneath is a light blue input field containing the email address 'admin@giger.com'. Below the email field is a password field represented by a series of dots. A teal 'Login' button is positioned below the password field. At the bottom of the form, there is a link that says 'Don't have an account? Create one'.

Slika 5.5: Prikaz prijave u aplikaciju



Slika 5.6: Nakon uspješne prijave, aplikacija preusmjerava na prikaz javnih nastupa



Slika 5.7: Prikaz uspješnog logina u Seleniumu

### Test 3: Stvaranje profila *Glazbenika*

**Očekivano:** Nakon prijave u aplikaciju, klikovima na gumbove: *Profile*, *Settings*, *Change profile type*, nude se opcije za postati glazbenik ili organizator. Kod stvaranja organizatora je potrebno unijeti ime kao menadžera, dok kod stvaranja glazbenika je moguće unijeti biografiju, odabrati instrumente koje glazbenik svira te je li kalendar glazbenika javan ili ne. Korisnik ispunjavanjem forme i klikom na gumb *Confirm* stvara profil glazbenika.

**Rezultat:** U slučaju da je korisnik već glazbenik, aplikacija će ga u pokušaju stvaranja o tome obavijestiti. U suprotnom, aplikacija dojavljuje da je korisniku uspješno dodijeljena uloga glazbenika.

Na slici 5.7 prikazano je uspješno stvaranje glazbenika, dok je na slici 5.8 prikazano što se prikaže na istom mjestu u aplikaciji kada već jesmo glazbenik. Na slici 5.9 je prikazano i kako izgleda Selenium test za slučaj kada korisnik već je glazbenik. U tom slučaju, kao što vidimo, Selenium test pada.

le ...web-lokaciji giger-fer-dev.herokuapp.com navodi se sljedeće

Musician created!

U redu

Organizer name

Confirm

I am a musician

Test biography

Guitar x Oboe x

My calendar will be public: ☒

Confirm

Slika 5.8: Forma za stvaranje profila glazbenik

Home Log out Profile

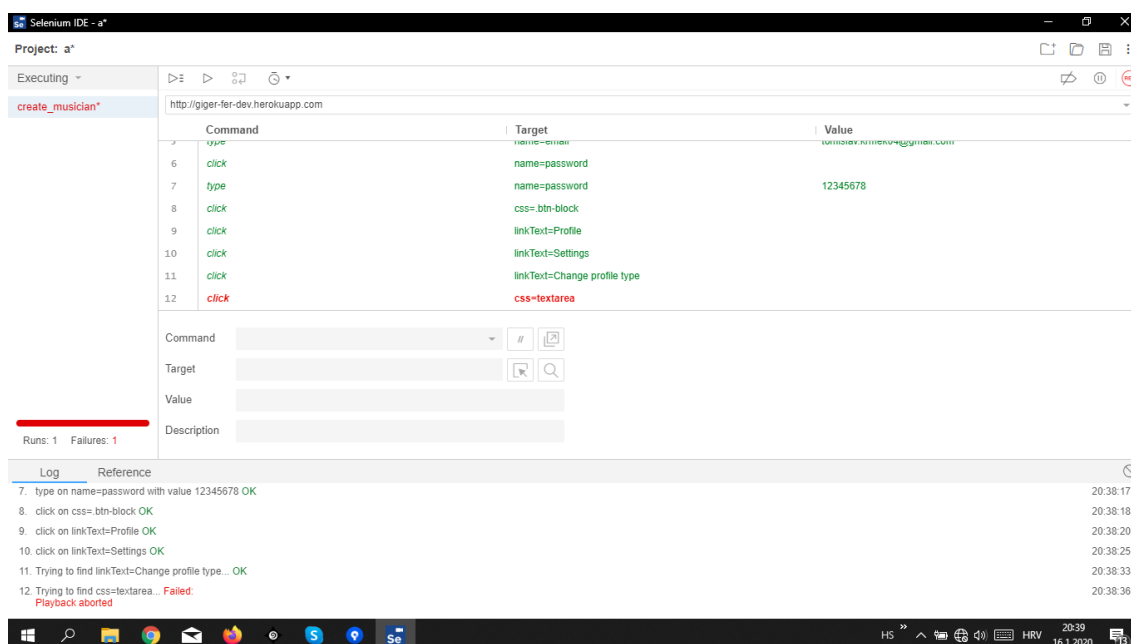
I want to make an organizer profile

You are already an organizer

I am a musician

You are already a musician

Slika 5.9: Forma za stvaranje profila glazbenik



Slika 5.10: Prikaz Selenium testa za stvaranje profila glazbenika koji je već glazbenik

#### Test 4: Objavljivanje *Post-ova*


**Očekivano:** Nakon što se prijavimo u aplikaciju i jesmo Glazbenik, odemo na *Profile*, *Musician\_profile* te *View* gdje se prikazuju sve objave prijavljenog glazbenika. Ondje imamo i polje u koje upišemo željeni tekst objave te pritisnemo *Post* kako bi se on objavio.

**Rezultat:** Post se objavi i prikazuje se među ostalim objavama glazbenika.


Na slici 5.11 je prikazano dodavanje teksta nove objave, a na 5.12 je prikazan Selenium test za dodavanje nove objave koji uspješno prođe.

[Home](#) [Log out](#) [Profile](#)

admin.doe




Instruments I play:

Marimba

Today I had a beautifull day,

Post

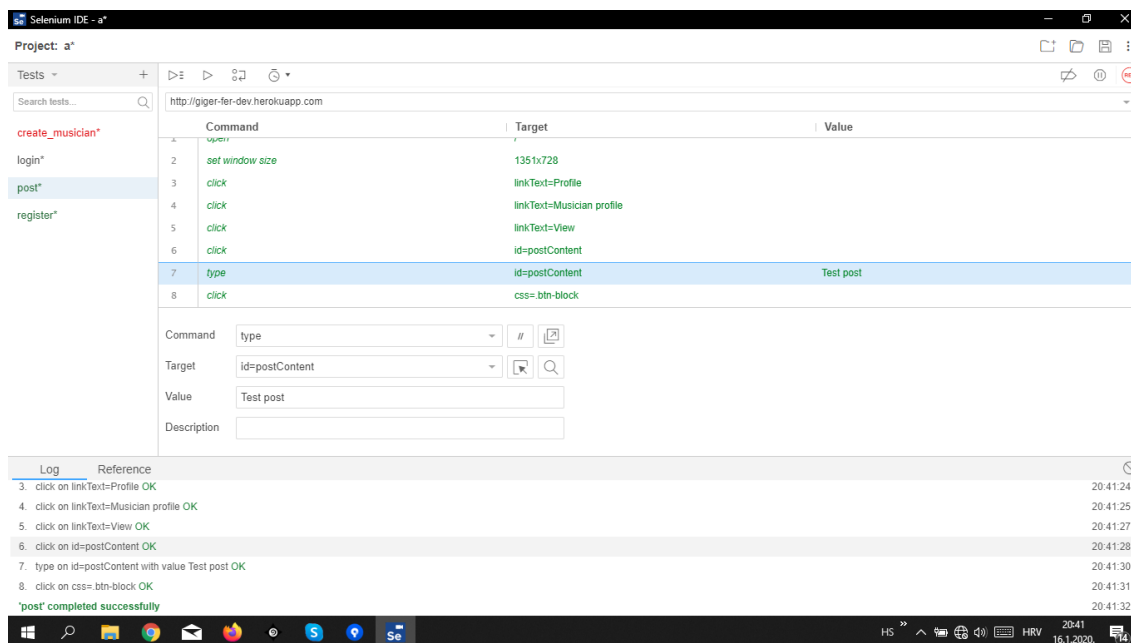
admin.doe2019-12-07T19:00:00

 First post

Comment (1)

admin.doe2019-12-08T19:00:00

Slika 5.11: Prikaz dodavanja teksta za novu objavu

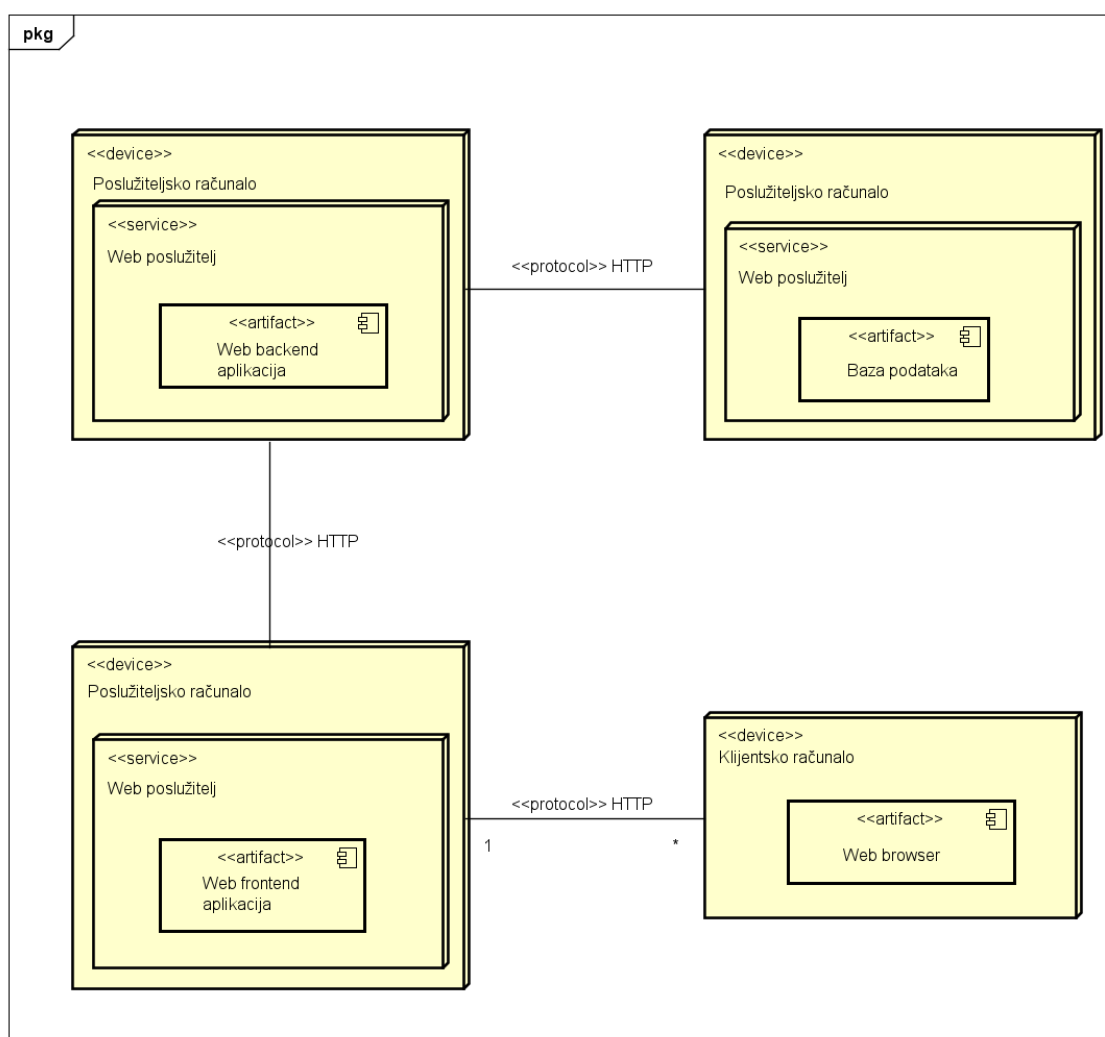


Slika 5.12: Prikaz Selenium testa za dodavanje objave



## 5.3 Dijagram razmještaja

Dijagrami razmještaja opisuju topologiju sklopovlja i programsku potporu koja se koristi u implementaciji sustava u njegovom radnom okruženju. Sve komponente programske potpore smještene su na oblak platformu Heroku. Heroku kao platforma omogućuje programerima izvođenje i operiranje nad aplikacijama u oblaku. U ovome slučaju pozadinska i prednja aplikacija razmještene su na zasebene poslužitelje kao i baza podataka. Sustav je baziran na arhitekturi klijent - poslužitelj i komunikacija između njih odvija se HTTP protokolom.



Slika 5.13: Dijagram razmještaja

## 5.4 Upute za puštanje u pogon

Ponajprije je potrebno preuzeti Postgres SQL bazu podataka na operacijski sustav, poželjno Windows. Nakon toga je potrebno provesti standardnu instalaciju. U servisu LoaderService nalaze se testni podaci za inicijalno punjenje baze. Za postavljanje lozinke baze potrebno je u password polje u system-local.properties postaviti prethodno definiranu lozinku za bazu podataka. Alternativno moguće je u root direktoriju standardnog terminala pokrenuti naredbu docker-compose up koja će u virtualnom kontejneru pokrenuti servise koji su predefinirani u dockeru docker-compose.yml datoteci. Za konfiguraciju aplikacije na Heroku poslužitelju koristi se system.properties iz kojeg se čita pokretanje Spring Boot aplikacije defaultnim profilom. U izradi aplikacije korišten je radni okvir Spring Boot. Za pokretanje Javine aplikacije potrebno je imati instaliran Java Runtime Environment v11. Za pokretanje frontend aplikacije potrebno je imati instaliranu platformu Node.js. Za pokretanje također je potrebno otpakirati arhivu server.var. Kako bi se pokrenuo build koristi se automatizirani sustav Gradle.

## 6. Zaključak i budući rad

Zadatak našeg projekta je bio razvoj web aplikacije koja bi olakšala praćenje i stvaranje muzičkih događaja, uglavnom koncerata. Namijenjena je glazbenicima, organizatorima te javnosti. Svaka osoba se može registrirati i prijaviti, te u svojem profilu naznačiti da je glazbenik i/ili organizator. Glazbenici mogu stvoriti profil benda te pozvati druge glazbenike u bend, dok organizatori mogu stvoriti događaje te pozvati bend da nastupa na događaju. Javnost ima pristup popisu događaja te njihovim detaljima, kao i prikaz lokacije događaja na Google Maps-u. Aplikacija korisnicima nudi i ostale funkcionalnosti, kao što su poruke, komentari, objave itd.

Prva faza projekta je uključivala međusobno upoznavanje ljudi u grupi i dogovor oko zadatka projekta koji se sastojao od generalnog opisa aplikacije. Nakon prvog sastanka sa asistentom te demonstratorom, te nakon što nam je odobren naš prijedlog zadatka, naša grupa je, uz učestale sastanke sa asistentom, počela ulaziti u detalje razrade samog zadatka. Nakon što je zadatak bio razrađen, i nakon naučenog gradiva sa predavanja kolegija, počeo je i rad na dokumentaciji. Taj rad se sastojao od opisa zadatka, navođenja specifikacija programske potpore(funkcionalni te ostali zahtjevi) te opisa arhitekture i raznih dijagrama. Na taj način smo stvorili "kostur" aplikacije, te nam je početak samog programskog rješavanja zadatka zbog toga bio puno lakši. Za kraj prve faze, naša aplikacija je imala funkcionalnu registraciju te prijavu korisnika.

Druga faza projekta je u početku imala glavni fokus na programskoj implementaciji svih zahtjeva. Obzirom da 5 od 7 članova grupe nije imala nekakvog prethodnog iskustva sa radom na aplikacijama, ova faza je bila zahtjevnija, kompliciranija te izazovnija. Članovi grupe su se morali učiti nove vještine kako bi uspjeli implementirati funkcionalnu aplikaciju. Iskusniji članovi su pomogli manje iskusnima te ih naučili nešto novo, ili u krajnjem slučaju ih uputili korisnim resursima na internetu. Što se dokumentacije tiče, za drugu fazu smo dokumentaciju morali nadopuniti sa više dijagrama koji detaljnije opisuju naš zadatak, te informacijama vezane za alate koje smo koristili, ispitivanje sustava te uputa za puštanje aplikacije u pogon.

Kroz obje faze projekta su članovi tima komunicirali preko platforme Slack,

preko koje su se i dogovarali sastanci uživo kako bi se lakše dogovorili oko određenih stvari te radi lakše i bolje komunikacije općenito. Isto tako, redovitim sastancima sa asistentom se uvelike poboljšao uvid u zadatak svakog člana.

Za završetak, naša grupa nije uspjela implementirati sve funkcionalne zahtjeve, ali svi zahtjevi visokog te skoro svi zahtjevi srednjeg prioriteta su implementirani. Grupa je naišla na brojne prepreke različitih oblika tijekom razvoja aplikacije, od malog predznaka i vremena obzirom na ostale obaveze na fakultetu, kompleksnih prepreka vezane za implementacije određenog dijela funkcionalnosti unutar aplikacije do problema koji su bili van naše moći, kao naprimjer greška u najnovijoj inačici nužnog paketa za frontend.

Bez obzira na sve probleme koji su se pojavili u vremenu projektnog zadatka, članovi grupe su generalno zadovoljni rješenjem zadatka. Uz to što su svi naučili nešto novog strane tehnologija, programiranja te pisanja dokumentacije, članovi su stekli i nova poznanstva te su naučili kako komunicirati i raditi kao tim.

# Popis literature

1. Oblikovanje programske potpore, FER ZEMRIS, <http://www.fer.hr/predmet/opp>
2. Astah Community, <http://astah.net/editions/uml-new>
3. GitLab Pipelines, <https://docs.gitlab.com/ee/ci/pipelines.html>
4. Sprint Boot, <https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/>
5. Kent Beck, *Test Driven Development*, 2000.

# Indeks slika i dijagrama

2.1	Prikaz <i>Amy</i> aplikacije . . . . .	7
2.2	Prikaz <i>BandFriend</i> aplikacije . . . . .	8
3.1	Obrasci uporabe za korisnika i organizatora . . . . .	24
3.2	Obrasci uporabe za javnost i administratora . . . . .	25
3.3	Obrasci uporabe za korisnika . . . . .	26
3.4	Sekvencijski dijagram za UC4 . . . . .	27
3.5	Sekvencijski dijagram za UC10 . . . . .	28
3.6	Sekvencijski dijagram za UC15 . . . . .	29
3.7	Sekvencijski dijagram za UC24 . . . . .	30
4.1	Pojednostavljeni prikaz MVC-a . . . . .	34
4.2	Dijagram baze podataka . . . . .	44
4.3	Dijagram razreda - dio Controllers . . . . .	45
4.4	Dijagram razreda - dio Data transfer objects, prvi dio . . . . .	46
4.5	Dijagram razreda - dio Data transfer objects, drugi dio . . . . .	47
4.6	Dijagram razreda - dio Service . . . . .	48
4.7	Dijagram razreda - razredi entiteta 1 . . . . .	49
4.8	Dijagram razreda - razredi entiteta 2 . . . . .	50
4.9	Dijagram razreda - security . . . . .	51
4.10	Dijagram repozitorija - repository . . . . .	52
4.11	Dijagram razreda - errors . . . . .	53
4.12	Dijagram stanja - stvaranje giga . . . . .	54
4.13	Dijagram aktivnosti . . . . .	56
4.14	Dijagram komponenti . . . . .	57
5.1	Primjer JUnit testa . . . . .	61
5.2	Primjer integracijskog testa . . . . .	62
5.3	Prikaz poruke za neispravnu registraciju . . . . .	63
5.4	Selenium test koji prikazuje ispravnu registraciju . . . . .	64
5.5	Prikaz prijave u aplikaciju . . . . .	65

5.6	Nakon uspješne prijave, aplikacija preusmjerava na prikaz javnih nastupa . . . . .	66
5.7	Prikaz uspješnog logina u Seleniumu . . . . .	67
5.8	Forma za stvaranje profila glazbenik . . . . .	68
5.9	Forma za stvaranje profila glazbenik . . . . .	68
5.10	Prikaz Selenium testa za stvaranje profila glazbenika koji je već glazbenik . . . . .	69
5.11	Prikaz dodavanja teksta za novu objavu . . . . .	70
5.12	Prikaz Selenium testa za dodavanje objave . . . . .	71
5.13	Dijagram razmještaja . . . . .	72
6.1	Dijagram pregleda promjena na grani Master . . . . .	83

# Dodatak: Prikaz aktivnosti grupe

## Dnevnik sastajanja

### 1. sastanak

- Datum: 3. listopada 2019.
- Prisutni: I. Juren, T. Krmek, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča
- Teme sastanka:
  - predlaganje ideja za projektni zadatak
  - odabir između web ili mobilne aplikacije
  - svaki član je iznio koja predznanja ili iskustva ima vezano za stvaranje aplikacije

### 2. sastanak

- Datum: 9. listopada 2019.
- Prisutni: I. Juren, T. Krmek, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča
- Teme sastanka:
  - upoznavanje s mentorima i demonstratorom
  - razgovor o tehnologijama koje ćemo koristiti
  - dogovoren način komunikacije s asistentom i demonstratorom
  - upoznavanje s ponuđenim projektom te razgovor o tome kako poboljšati naš prijedlog projekta

### 3. sastanak

- Datum: 14. listopada 2019.
- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, P. Lanča
- Teme sastanka:
  - sastanak s asistentom
  - nacrtana gruba shema različitih korisnika s pripadajućim potrebnim pristupom
  - predlaganje funkcionalnosti, dogovoreno što se obavezno mora implementirati

### 4. sastanak



- Datum: 22. listopada 2019.
- Prisutni: I. Juren, T. Krmek, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča
- Teme sastanka:
  - razrada must i could have funkcionalnosti
  - izjašnjavanje svojih nedoumica te njihovo razrješavanje, eventualno stavljene na popis za pitanja na sastanku s asistentom
  - razriješena problematika benda (glavni i rezervni članovi)
  - nakon internog, sastanak s asistentom: dogovorena detaljnija implementacija, napravljen popis zadataka koje treba odraditi do idućeg sastanka

#### 5. sastanak

- Datum: 28. listopada 2019.
- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča
- Teme sastanka:
  - nabranje usecase-ova
  - podjela rada
  - određena pitanja za idući sastanak s asistentom

#### 6. sastanak

- Datum: 29. listopada 2019.
- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča
- Teme sastanka:
  - sastanak s asistentom: pokazano što je sve napravljeno
  - napravljen popis zadataka koji moraju biti gotovi do idućeg sastanka s asistentom i sve što još treba za prvu verziju
  - riješena dilema oko recenzija
  - razriješen problem solista, bit će jednočlani bend
  - rasprava oko baze podataka, što treba promijeniti i poboljšati

#### 7. sastanak

- Datum: 11. studenog 2019.
- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča, T. Krmek
- Teme sastanka:
  - određivanje preostalih poslova te njihov raspored po članovima
  - na sastanku dovršen frontend te je aplikacija isporučena

#### 8. sastanak

- Datum: 12. listopada 2019.

- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča, T. Krmek
- Teme sastanka:
  - sastanak s asistentom: pokazana aplikacija te dokumentacija
  - razriješene neke nedoumice oko baze podataka

#### 9. sastanak

- Datum: 10. prosinca 2019.
- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča, T. Krmek
- Teme sastanka:
  - sastanak s asistentom: razrada plana, pitanja vezana za aplikaciju
  - raspored poslova

#### 10. sastanak

- Datum: 7. siječnja 2019.
- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča, T. Krmek
- Teme sastanka:
  - sastanak s asistentom: prikaz alfa verzije, razrada daljnjeg plana, što popraviti
  - raspored poslova

#### 11. sastanak

- Datum: 14. siječnja 2019.
- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča, T. Krmek
- Teme sastanka:
  - sastanak s asistentom: pokazivanje popravaka dizajna i dodanih funkcionalnosti
  - raspored poslova

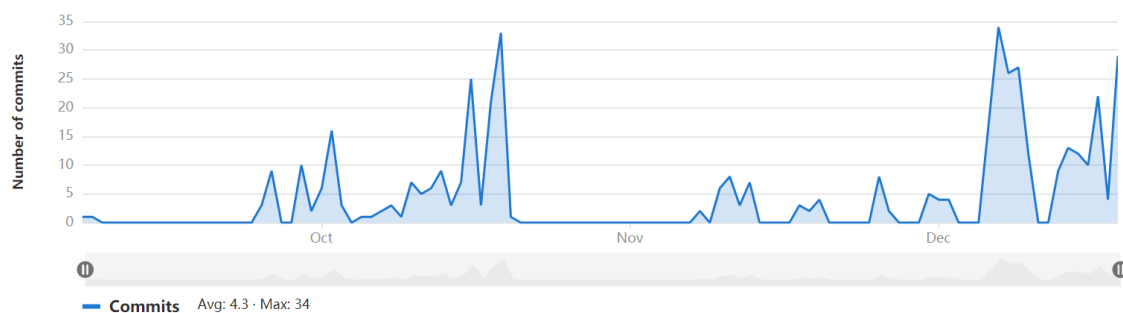
#### 12. sastanak

- Datum: 16. siječnja 2019.
- Prisutni: I. Juren, M. Jurić, M. Zec, S. Gaši, M. Nosil, P. Lanča, T. Krmek
- Teme sastanka:
  - interni sastanak i završetak radova na projektu, popravljjanje i ispravljanje

## Tablica aktivnosti

	Ivan Juren	Stela Gaši	Marin Jurić	Tomislav Krnek	Paolo Lanča	Mihael Nosil	Mario Zec
Upravljanje projektom	4	3		1	3		3
Opis projektnog zadatka	3	2					1
Funkcionalni zahtjevi	2	2				1	2
Opis pojedinih obrazaca		2	1		2	10	4
Dijagram obrazaca					1		3
Sekvencijski dijagrami		2	2		1		
Opis ostalih zahtjeva			1		1		
Arhitektura i dizajn sustava	6	2			2		
Baza podataka	10	8		3			
Dijagram razreda	2	1					15
Dijagram stanja			1				
Dijagram razmještaja							2
Dijagram komponenti							4
Dijagram aktivnosti		3					
Korištene tehnologije i alati			1				
Ispitivanje programskog rješenja	10	10		3			
Upute za puštanje u pogon	8					2	
Zaključak i budući rad						2	
Dnevnik sastajanja	1	1			3		
Popis literature	1	1					
<i>Frontend</i>	10		55	70	60	50	
<i>Backend</i>	100	40					40
<i>Vrijeme provedeno na sastancima</i>	15	15	15	7	10	9	10

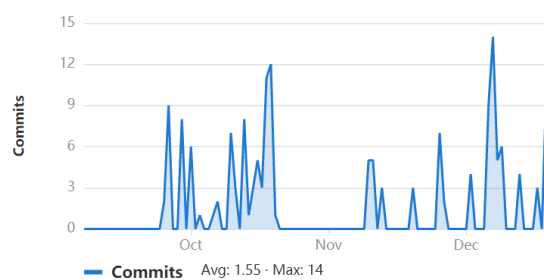
## Dijagrami pregleda promjena



Slika 6.1: Dijagram pregleda promjena na grani Master

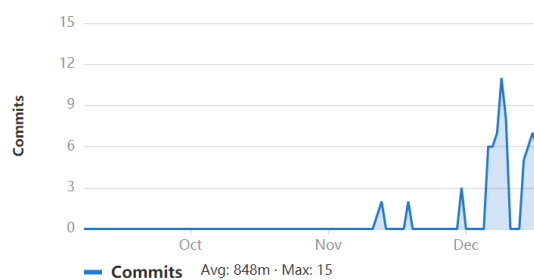
## jurenivan

163 commits (juren.ivan@gmail.com)



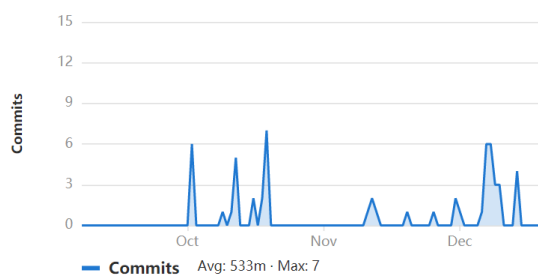
## paolo

89 commits (paolo.lanca@fer.hr)



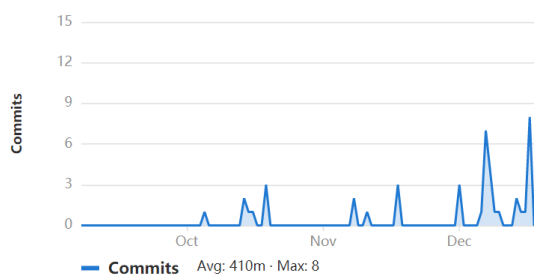
## Stela Gaši

56 commits (sg50985@fer.hr)



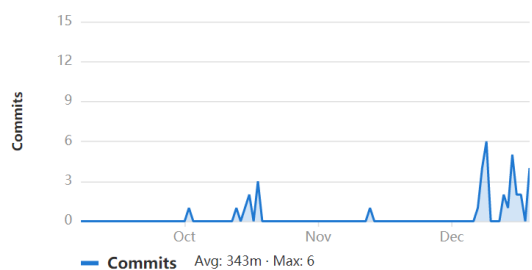
## Tomislav Krmek

43 commits (tomislav.krmek@triple-innovations.com)



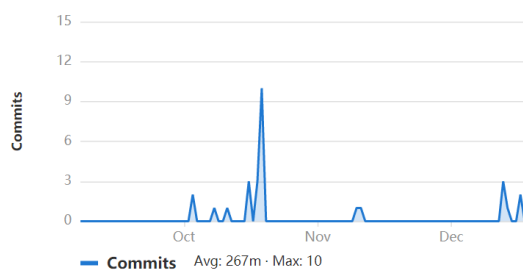
## Frootek

36 commits (marin.juric@fer.hr)



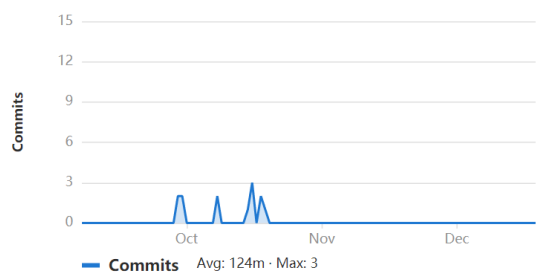
## Mario Zec

28 commits (mario.zec@fer.hr)



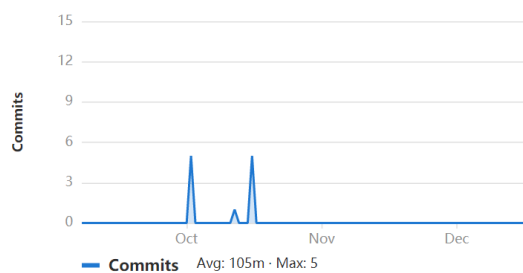
## Benena

13 commits (paolo.lanca@fer.hr)



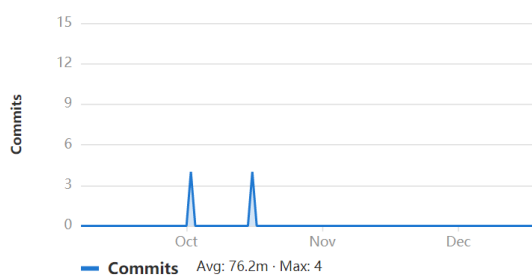
## mihael nosil

11 commits (nosil.mihael@gmail.com)



## Mihael

8 commits (nosil.mihael@gmail.com)



## JurenIvan

5 commits (juren.ivan@gmail.com)

