Univerzitet Crne Gore

ELEKTROTEHNIČKI FAKULTET

Studije primijenjenog računarstva

Baze podataka (napredni kurs)

-dokumentacija mini-projekta-

Tema: **Sound's bazaar**

Učesnici u projektu:

1. Amil Đešević 71/21 – lider grupe
2. Samed Mehmedović 107/21
3. Matija Gajović 97/21
4. Luka Zlaić 6/21
5. Damir Pilica 3/21

Podgorica 23.5.2024. godine

1. Uvod (o projektu)

Ideja tima za kreiranje ove *web* aplikacije proizilazi iz ljubavi prema muzici. **Sound bazaar** je inovativan naziv za *web* aplikaciju koja će svojim funkcionalnostima podsjećati na već postojeće *web* aplikacije kao što su Spotify, Sound Cloud, Deezer, Youtube i ostale.

Slika br. 1 – Logo platforme "Sound's Bazaar"

Ova *web* aplikacija će pružati isključivo muzički sadržaj. Osim snimljenih kompozicija u standardnim formatima (*.mp3, .mp4*) na platformi bi postojali i drugi formati muzičkih zapisa (*.midi, .wave, .adg*) koji bi bili namijenjeni za kompozitore, producente i sve ostale ljude koji žele stvarati svoja muzička djela.

Korisnici na ovoj platformi bi mogli preuzimati muziku ukoliko je admin koji je dodao tu melodiju dozvolio preuzimanje. Korisnici, *stakeholder-i* koji mogu biti dio aktivnosti na ovoj platformi su ljudi koji ažuriraju platformu, dodaju muziku i upravljaju njome (admini, kompozitori, producenti) i korisnici kojima će taj muzički sadržaj biti dostupan.

Korisnici će moći da kreiraju odgovarajuće *playlist*-e, da sačuvaju pojedinačno neku muzičku melodiju ili je preuzmu na svom uređaju. Sve melodije na ovoj *web* aplikaciji je moguće besplatno preslušati, dok je preuzimanje dozvoljeno korisnicima samo ako je admin sistema odobrio preuzimanje.

Osim ovih funkcionalnosti, sistem treba da ima opciju registracije korisnika i prijavljivanja na svoj nalog na aplikaciji gdje će svaki korisnik imati svoj nalog na kojem će se nalaziti sačuvana muzika ili *playlist*-a.

Grupa inspirisana ovim projektom ima za cilj da ponudi sličnu, ali jedinstvenu uslugu koja će kombinovati aspekte servisa vezanih za muzičke melodije i kompozicije, uz dodatne navedene funkcionalnosti.

2. Opis opštih funkcionalnosti

U ovom poglavlju će biti opisane opšte funkcionalnosti sistema do kojih je došlo tokom faze planiranja, analize i dizajna sistema. Korišćenjem tekstualnih i grafičkih modela olakšaće se implementacija tj. programiranje samog sistema.

2.1 Funkcionalni i nefunkcionalni zahtjevi

Funkcionalni i nefunkcionalni zahtjevi igraju ključnu ulogu u razvoju softverskih aplikacija jer definišu šta sistem treba da radi (funkcionalni zahtjevi) i kako bi sistem trebao da radi (nefunkcionalni zahtjevi).

Funkcionalni zahtjevi za ovaj projekat su:

- **Registracija korisnika** – Korisnici mogu kreirati (registrovati) svoj nalog putem forme za registraciju. Potrebno je validirati podatke prilikom registracije (e-mail adresa, lozinka i ostalo).
- **Prijava korisnika** – Ovaj funkcionalni zahtjev je vezan za registraciju korisnika. Nakon što se korisnik registruje na sistem, on se može prijaviti sa svojim podacima. Potrebno je implementirati autentifikaciju korisnika koristeći e-mail adresu i lozinku.
- **Stranica za muziku** – Treba da postoji stranica za muziku na kojoj će korisnici moći da pretražuju i slušaju određene numere. Pomoću modalnog prozora korisnicima se otvara odabrana pjesma i mogu je dodati na svoju *playlist*-u.
- **Kreiranje i upravljanje playlistama** – Korisnik može kreirati *playlist*-u u kojoj bi dodavao muziku. Prilikom kreiranja *playlist*-e korisnik unosi naziv liste (npr. Playlist 1) i opis (npr. žanr te liste). Svaka *playlist*-a treba da ima opciju (dugme) za dodavanje još numera unutar nje.

Nefunkcionalni zahtjevi za ovaj projekat su:

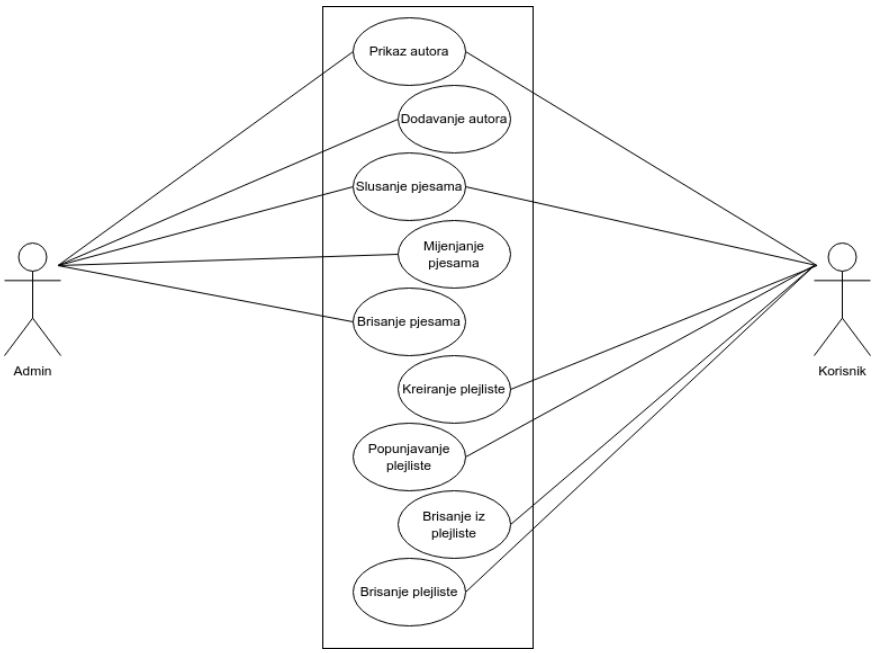
- **Performanse** – Aplikacija mora brzo reagovati na korisničke zahtjeve, sa minimalnim vremenom učitavanja. Sistem mora podržavati veliki broj ljudi istovremeno na platformi.
- **Sigurnost** – Ovaj nefunkcionalni zahtjev obuhvata implementaciju zaštite od sigurnosnih prijetnji (SQL inekcija, XSS napadi i CSRF). Prenos podataka na platformi mora biti zaštićen enkripcijom (posebno lozinke).
- **Pouzdanost** – Sistem treba da ima veliku dostupnost. Potrebno je implementirati proceduru za *backup* i *recovery* kako bi se osiguralo očuvanje podataka. Platforma treba da obuhvati SSL sertifikat kako bi se korisnici osjećali pouzdanije na platformi.
- **Prilagodljivost** – Sistem treba da bude prilagodljiv na svim uređajima (responzivnost), uključujući desktop uređaje, kao i mobilnu i tablet verziju uređaja. Aplikacija treba biti dizajnirana na način da može podržati sve veći broj korisnika i podataka u slučaju rasta platforme.

2.2 Analiziranje sistema

U ovom poglavlju će se kroz *UML* dijagrame koji se koriste tokom analize sistema objasniti šta ovaj sistem treba da radi.

2.2.1 Use case analiza

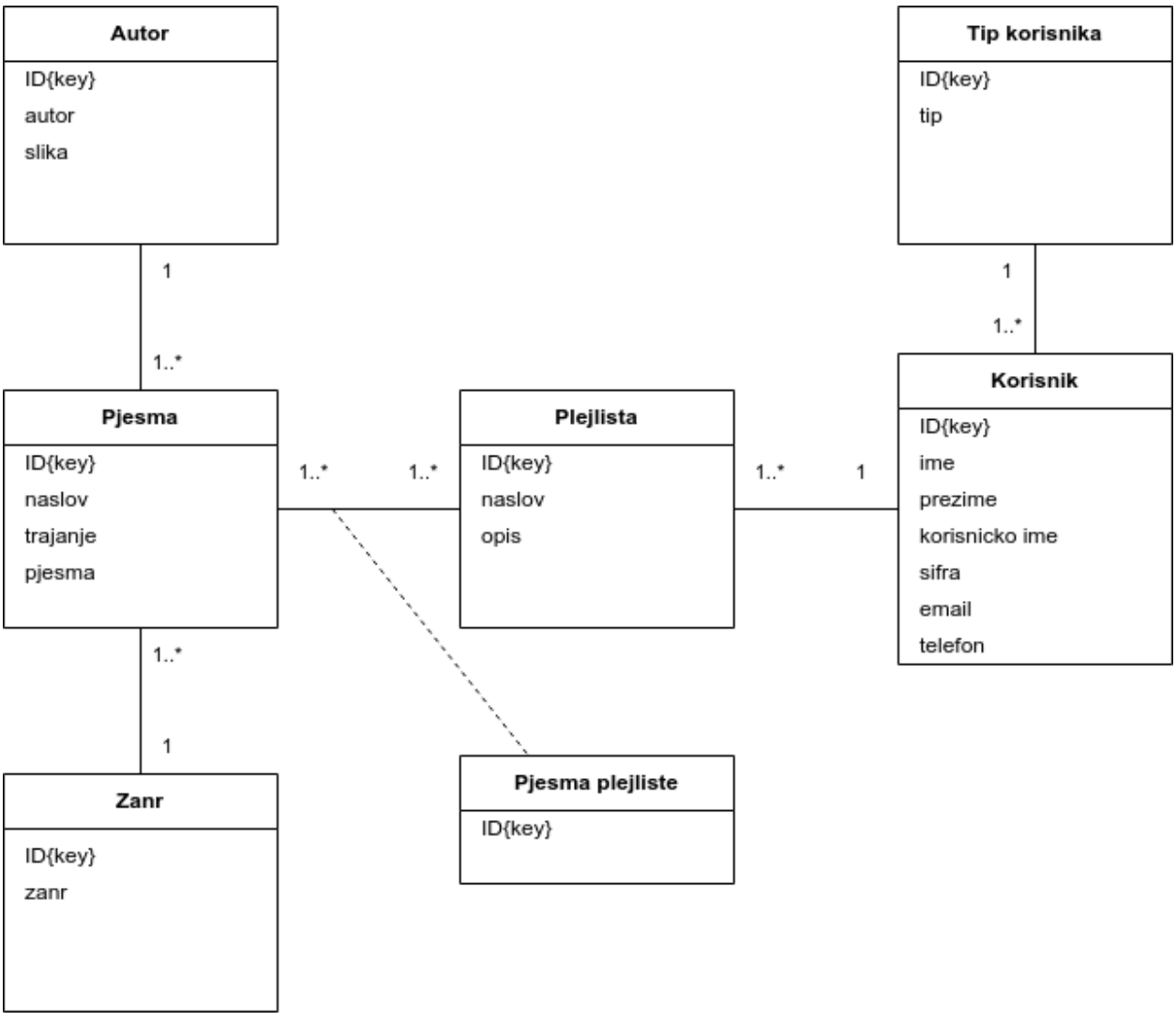
Slučaj korišćenja (*Use case*) je scenario u kojem će ovaj biti korišćen, i obično se sastoji od glagola i imenice (dodaj u korpu, naruči robu itd.). Slučajevi korišćenja u jednom sistemu se grafički prikazuju *Use Case* dijagramom.



Slika br. 2 – Use case dijagram

2.2.2 Dijagram klase u domenu problema

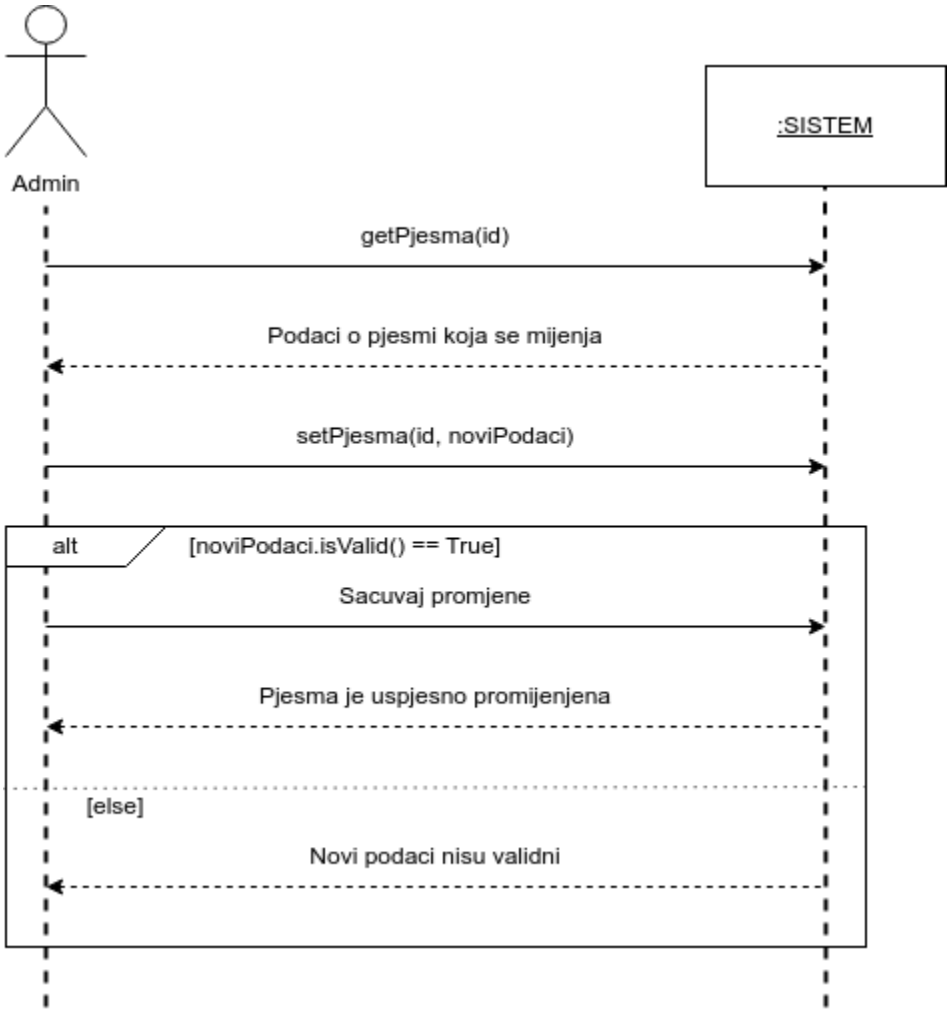
Dijagram klase u domenu problema grafički prikazuje sve klase koje će biti korišćene unutar ovog sistema. Osim klasa na ovom dijagramu su prikazani i atributi tih klasa kao i veze između klasa.



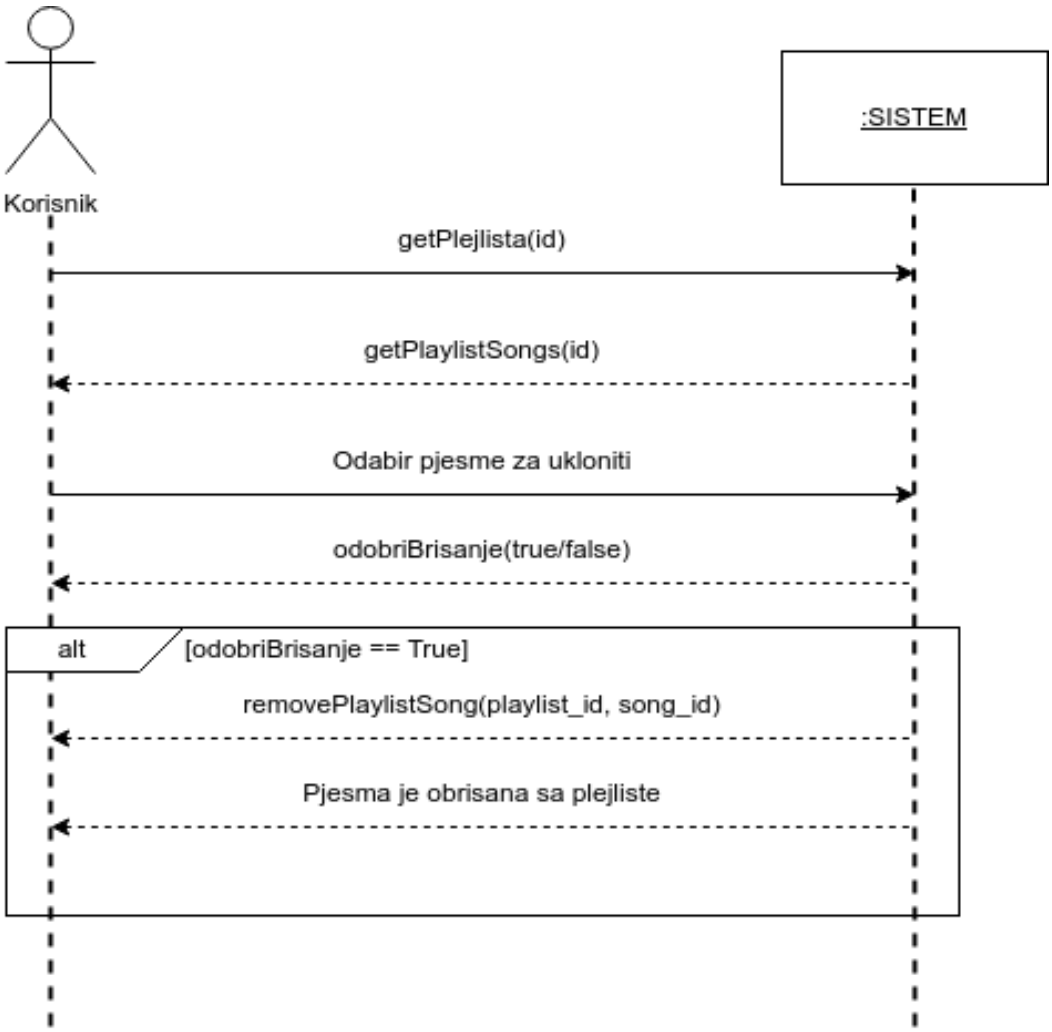
Slika br. 3 – Dijagram klase u domenu problema

2.2.3 *SSD (System sequence diagram)*

Dijagrami sistemske sekvence se koriste kako bi bolje objasnili komunikaciju između korisnika i sistema za pojedine slučajeve korišćenje. Najčešće se koriste za kompleksnije slučajeve korišćenja ili za one slučajeve koji nisu najjasniji. U slučaju ovog sistema napravljeni su *SSD* za dva slučaja korišćenja i to: Mijenjanje pjesama i brisanje iz plejliste.



Slika br. 4 – SSD (mijenjanje pjesama)



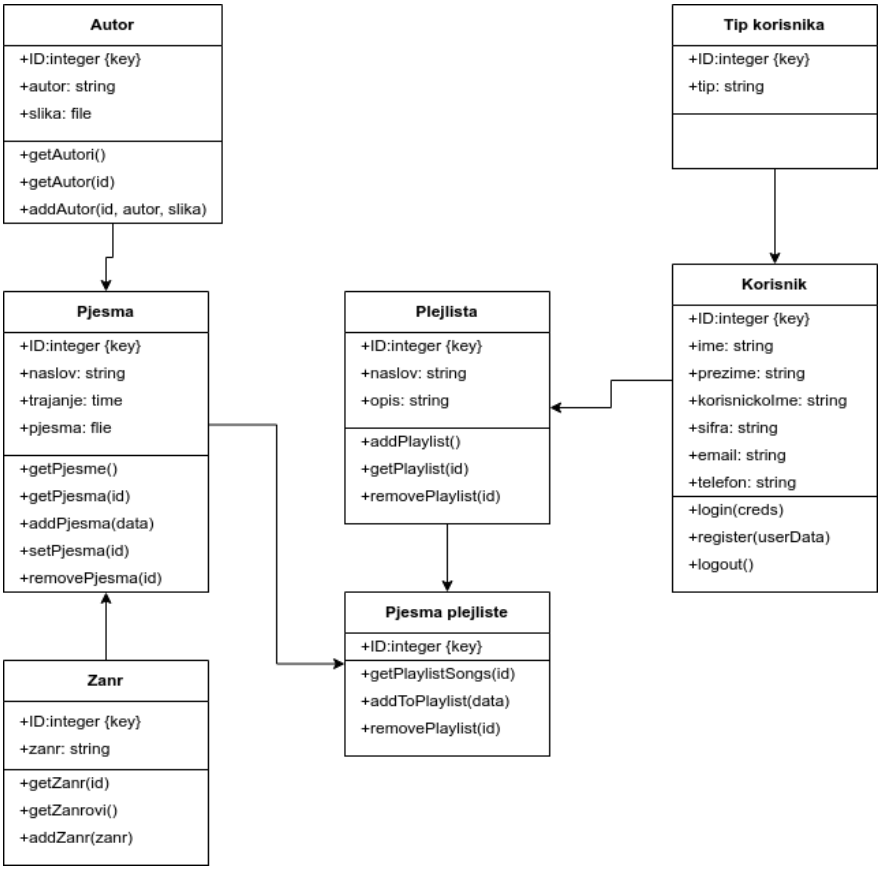
Slika br. 5 – SSD (brisanje iz playlist-e)

2.3 Dizajn sistema

Dizajn informacionih sistema je faza razvoja sistema u kojoj se tektualnim i/ili grafičkim modelima opisuje kako da se sistem koji je opisan u analizi napravi.

2.3.1 Dijagram dizajna klase

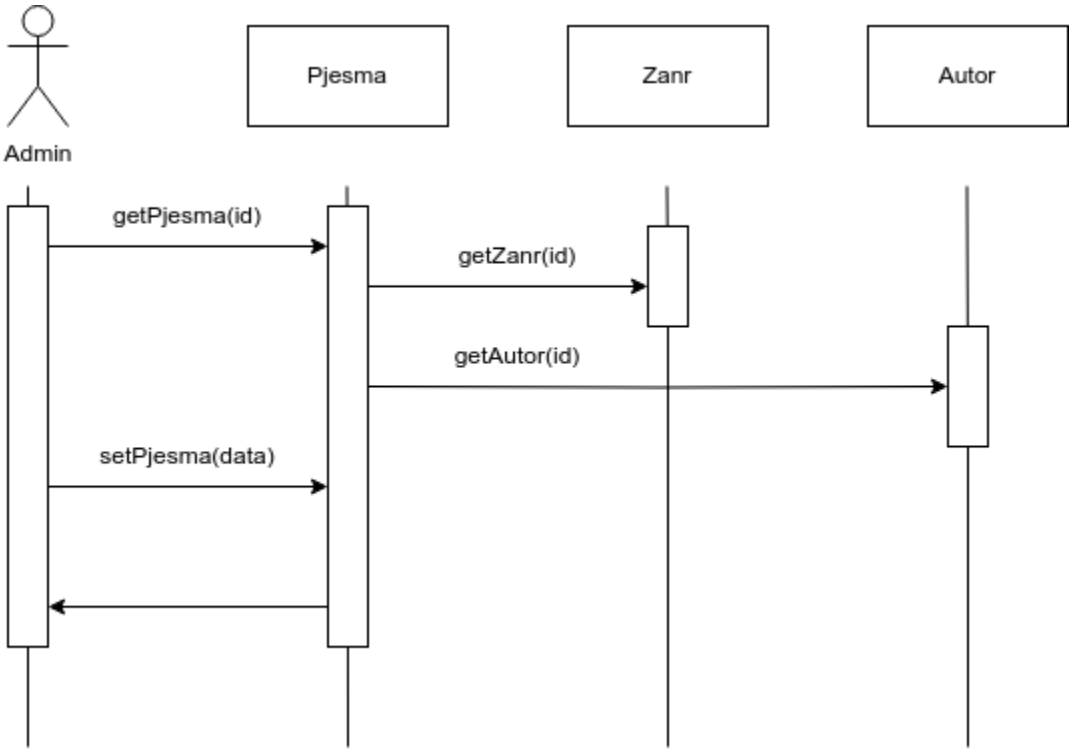
Dijagram dizajna klase je nadograđen dijagram klase u domenu problema iz analize, sa dodatkom metoda koje klase treba da implementiraju.



Slika br. 6 - Dijagram dizajna klase

2.3.2 Dijagram sekvence

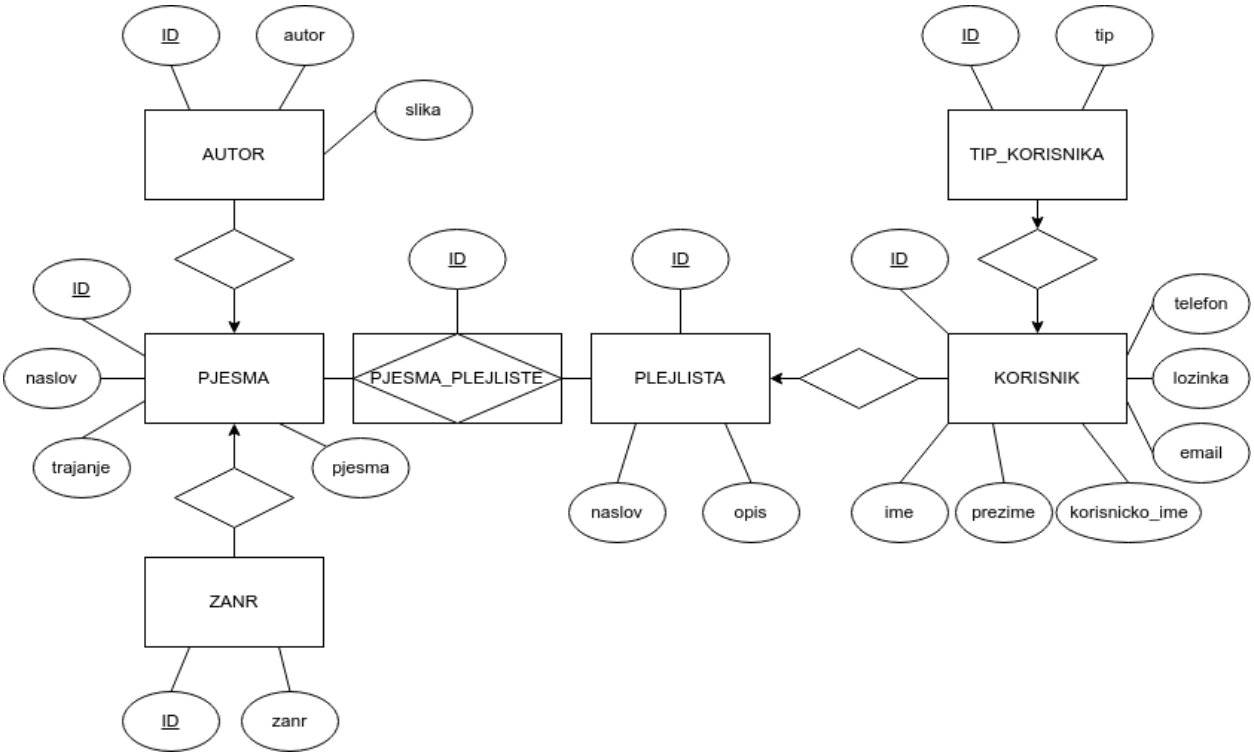
Dijagram sekvence je sličan *SSD*-u i na osnovu njega se i kreira za pojedinačni *use case*. Za razliku od *SSD*-a dijagram sekvence opisuje komunikaciju između korisnika i sistemskih klasa.



Slika br. 7 - Dijagram sekvence (mijenjanje pjesama)

2.3.3 Opis baze podataka

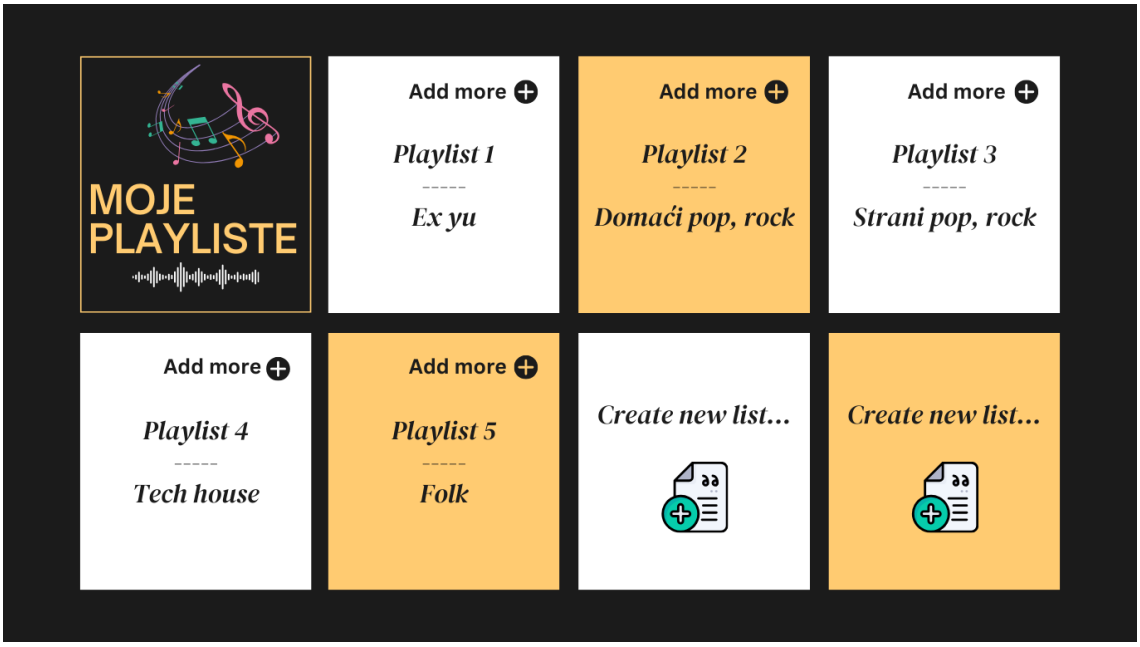
Dijagram kojim se objašnjava struktura baze podataka sa njenim tabelama, vezama i kolonama je *ERD* (*Entity relationship diagram*).



Slika br. 8 - ERD dijagram

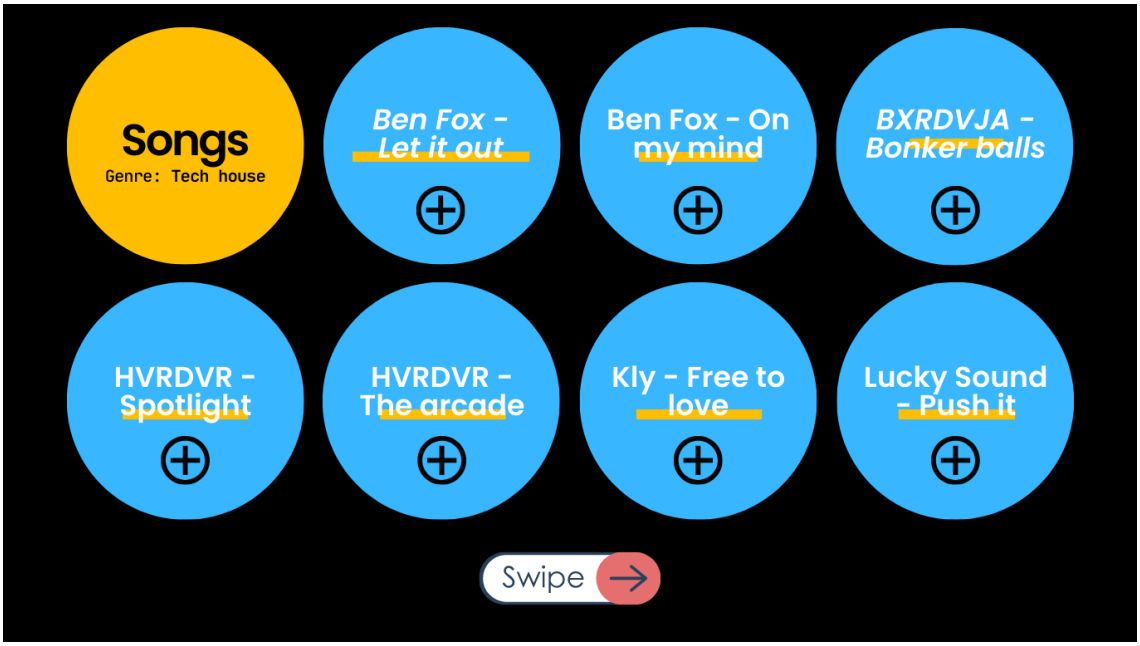
2.3.4 Skice interfejsa (*mokcup* ekrani)

Na skicama interfejsa (*mokcup* ekranima) su prikazane najbitnije funkcionalnosti ovog sistema. Na prvoj skici (*slika br. 8*) se nalaze *playlist*-e koje je korisnik sistema kreirao. Korisnik sistema može dodjeljivati ime ovim listama, dodati naziv žanra kojem pripadaju, dodati novu pjesmu unutar liste pomoću „Add more“ dugmeta i kreirati novu listu. Na ovoj skici je prikazano pet kreiranih *playlist*-i kao i prostor za kreiranje dvije liste koje za sada ne postoje. Ovi *mockup* ekrani služe kao primjer za pomoć pri kreiranju korisničkog interfejsa.



Slika br. 9 - Korisničke playlist-e

Na sljedećoj skici (*slika br. 9*) je prikazan *mockup* ekran stranice “Songs” koja sadrži pjesme koje su žanra *Tech house*. Na ovoj stranici, klikom na neku pjesmu, korisniku se otvara modalni prozor gdje može preslušati pjesmu ili je pomoću dugmeta ispod naziva dodati u svoju *playlist*-u. Dugme “Swipe” služi za prelazak na drugu stranicu gdje se nalazi još pjesama.



Slika br. 10 - Stranica sa pjesmama