Oblikovanje programske potpore

Ak. god. 2015/2016

Sustav za arhivu i reprodukciju tonskih zapisa

Dokumentacija, revizija 1.0

Grupa: *BananaBlade*
Voditelj: *Zvonimir Jurelinac*

Datum predaje: *20. studeni 2015.*

Asistent: *Miljenko Krhen*
Nastavnik: *Vlado Sruk*

Sadržaj

1. Dnevnik promjene dokumentacije

Rev.	Opis promjena / dodataka	Autor(i)	Datum
0.1	Stvoren predložak za dokumentaciju	Jurelinac, Škalec	25. 10. 2015
0.1.1	Napisan dio opisa Dodan rječnik pojmova	Jurelinac	25. 10. 2015
0.2	Proširen opis zadatka	Škalec	02. 11. 2015
0.3	Proširen pojmovnik Započeti funkcionalni zahtjevi	Škalec	05. 11. 2015
0.4	Dovršeni funkcionalni zahtjevi Manje izmjene	Škalec	10. 11. 2015
0.5	Izmjene dijela opisa, izmijenjeni neki obrasci upotrebe, dodani neki novi, izmijenjen rječnik pojmova	Jurelinac	14. 11. 2015
0.7	Dodani dijagrami obrazaca uporabe, započeto opisivanje arhitekture sustava, stavljen ER model i opis relacija u bazi podataka	Jurelinac	16. 11. 2015.
0.8	Dodan opis klijentskog dijela i upravitelja	Ivošević, Jurelinac	17. 11. 2015.
0.9	Dodani dijagrami razreda	Jurelinac	18. 11. 2015.

2. Opis projektnog zadatka

Cilj ovog projekta jest razviti informacijski sustav u obliku web aplikaciju čija je namjena upravljanje tonskim zapisima internetske radio postaje. Aplikacija bi korisnicima – vlasniku, administratorima, glazbenim urednicima i registriranim korisnicima – trebala omogućiti brzo, jednostavno i lako dostupno ispunjavanje svojih zaduženja i sudjelovanje u radu radio postaje. Također, posjetiteljima web stranice na kojoj se nalazi aplikacija trebalo bi biti omogućeno slušanje trenutno sviranog zvučnog zapisa na toj radio postaji.

Kako bi aplikacija bila prikladna što širem spektru korisnika, poželjno je da ona bude pristupačna, pregledna i dovoljno jednostavna za korištenje kako bi se njome mogu služiti i korisnici bez velikog informatičkog znanja. Također, bilo bi poželjno da dizajn aplikacije bude privlačan i moderan.

Detaljniji rad ove aplikacije je sljedeći: Za svaki dan unaprijed će se stvarati nova glazbena lista radio postaje, i to na način da će svaki glazbeni urednik stvarati liste za njemu dodijeljena termine unutar toga dana (jedan dodjeljeni termin traje sat vremena). Svi zapisi za reprodukciju moraju biti poznati najmanje 24 sata prije vremena njihove reprodukcije. Registrirani će korisnici stvaranjem lista želja, u koje će urednici imati uvid, moći i sami sudjelovati u odlučivanju o programu radio postaje. Administratori radio postaje moći će upravljati zvučnim zapisima kojima postaja raspolaže, kao i drugim korisnicima – moći će postavljati glazbene urednike, odlučivati o njima dodijeljenim terminima te uređivati podatke svih ostalih korisnika (izuzev vlasnika i drugih administratora). Administratore postavlja vlasnik radio postaje, koji je određen prilikom izrade informacijskog sustava.

Korisnike sustava možemo podijeliti u pet grupa: vlasnik sustava, administrator, glazbeni urednik, registrirani korisnik i neregistrirani korisnik (posjetitelj).

Vlasnik je sustava odgovoran za definiranje administratora, te upisivanje kontakt podataka i podataka o radio postaji.

Administrator sustava, kao što je već rečeno, određuje glazbene urednike, upravlja zvučnim zapisima, te uređuje podatke o urednicima i registriranim korisnicima postaje.

Registrirani korisnici mogu sastavljati liste glazbenih želja.

Neregistriranim su korisnicima dostupne mogućnosti slušanja glazbe, besplatne registracije i kratak pregled informacija o postaji.

Sustav može imati jednog vlasnika, najviše deset administratora, te neograničen broj registriranih korisnika. Svi korisnici mogu istovremeno koristiti sustav.

Na početnoj se stranici web aplikacije nalaze osnovni podaci o radio postaji, područje za prijavu korisnika u sustav kao i za registraciju novih korisnika, te na najistaknutijem mjestu, glazbeni player uz koji se nalaze i podaci o trenutno sviranom glazbenom zapisu.

Prijavom u sustav, korisniku će biti dostupna i upravljačka stranica sa svim njemu dostupnim mogućnostima, ovisno o vrsti korisničkog računa. Svi korisnici kao ponuđenu mogućnost imaju upravljanje vlastitim računom: pregled i izmjena osobnih podataka, promjena lozinke te brisanje korisničkog računa. Obični korisnici imaju mogućnost stvaranja i pregleda svoje liste želja. Glazbeni urednik, ima prikazane mogućnosti slanja zahtjeva za dodjelom termina za uređivanje, te stvaranje i uređivanje lista za reprodukciju u dodijeljenim mu terminima. Administratoru su na raspolaganju mogućnosti pregledavanja i uređivanja podataka o drugim korisnicima, upravljanja zvučnim zapisima, upravljanje glazbenim urednicima i njihovim terminima, te pregledavanje statistika korisnika i zapisa.

Liste korisničkih glazbenih želja sastoje se od maksimalno deset zapisa. Korisnik može po volji često uređivati svoju listu želja, no ona je tako vidljiva samo njemu. Da bi se želje s nje učinile globalno dostupnima (urednicima i administratorima), korisnik mora svoju listu potvrditi, što može učiniti jednom svaka 24 sata (nakon jedne potvrde mora proći najmanje toliko vremena do iduće). Glazbeni urednici uvidom u globalnu listu želja dobivaju povratnu informaciju od korisnika o traženosti pojedinih zapisa, što im omogućava da se bolje prilagode interesima slušatelja.

Za uspjeh ovog projekta ključno je da glazbeni urednici redovito koriste sustav i kreiraju nove glazbene liste za reprodukciju. Ako neki glazbeni urednik ne kreira svoju listu na vrijeme, ponovit će se reprodukcija liste od njegovog prethodnog termina. Problem nastaje ako se to događa prečesto, ili više dana za redom; korisnici ne žele slušati iste pjesme iz dana u dan, te bi u tom slučaju sustav trebao reagirati na odgovarajući način. Također je

moguće da, propustom administratora, neki termin ostane nedodjeljen, te će i tada sustav reagirati na odgovarajući način kako bi se spriječio privremeni prestanak emitiranja sadržaja.

Zbog mogućnosti da korisnik koristi slabiju internetsku vezu, a aplikacija uključuje prijenos i reprodukciju zvučnih zapisa preko iste, potrebno je da aplikacija bude što manja u pogledu količine podataka, kako bi se poboljšala brzina i kvaliteta usluge korisniku.

Osobni podaci svakog korisnika koji su pohranjeni u sustavu uključuju:

- ime
- prezime
- e-mail adresu
- lozinku
- zanimanje

Svi se podaci naknadno mogu promijeniti. Ispravnost email adrese je bitna jer će se putem nje korisnici obavještavati o svim bitnim događajima i promjenama.

Postaja ima arhivu tonskih zapisa koji su dostupni za reprodukciju. Za svaki su zvučni zapis poznati sljedeći podaci:

- ime glazbenog zapisa
- ime izvođača
- putanja do datoteke zvučnog zapisa
- album
- nakladnik
- glazbeni žanr
- godina izdanja
- tip nosača
- trajanje zapisa

- frekvencija uzorkovanja
- format zapisa
- broj bitova kvantizacije

Svi ovi podaci bit će pohranjeni u bazi podataka na poslužitelju.

Web aplikacija bit će napisana u nekoliko trenutno popularnih web tehnologija, redom *Python Flask* za poslužiteljski dio aplikacije i komunikaciju s bazom podataka, *AngularJS* za klijentski dio aplikacije koji se izvršava u web pregledniku, te *JADE* i *SASS*, za dizajn i strukturu web stranice.

Detalji sustava i njegove implementacije navedeni su u nastavku ovog dokumenta.

3. Rječnik pojmova

Flask – Framework za izradu web aplikacija u programskom jeziku Python, popularan zbog svoje jednostavnosti i lakoće korištenja, kao i male veličine.

Peewee ORM – Python biblioteka koja olakšava dizajn i korištenje baze podataka

AngularJS – Javascript framework za izradu web aplikacija, omogućuje njihov brz i intuitivan razvoj

REST – *Representational State Transfer* – stil arhitekture mrežnih aplikacija koja komunikaciju između klijenta i servera ostvaruje putem HTTP zahtjeva

SASS – *Syntactically Awesome Style Sheets* – proširenje CSS jezika koje dodaje brojne mogućnosti i bitno olakšava pisanje stilskih datoteka, kao i snalaženje u njima

JADE – strukturirani predlošci koji olakšavaju pisanje i održavanje HTML koda

MVC – *Model/View/Controller* – obrazac arhitekture programske podrške, razdvaja sustav na **modele** koji opisuju podatke i njihove operacije, **poglede** koji vrše interakciju s krajnjim korisnicima (prikaz sučelja i podataka), te **upravitelje** koje povezuju modele s pogledima.

Typescript – Nadgradnja programskog jezika Javascript koja podržava statičke tipove podataka i potpuni OO model programiranja te olakšava samo pisanje koda

AJAX – *Asynchronous Javascript And XML* – vrsta komunikacije između web preglednika i poslužitelja koja omogućava asinkrone web aplikacije (promjena sadržaja bez potrebe za ponovnim učitavanjem stranice)

HTTP – *Hypertext Transfer Protocol* – internetski protokol koji se koristi na World Wide Webu za komunikaciju između web poslužitelja i web preglednika

JSON – *Javascript Object Notation* – format za prijenos podataka prikladan za korištenje pri prijenosu preko HTTP protokola

4. Funkcionalni zahtjevi

Dionici našeg sustava, odnosno osobe koje u njemu imaju interes, su:

- vlasnik sustava
- administrator
- glazbeni urednik
- registrirani korisnik
- posjetitelj (neregistrirani korisnik)
- korisnik (zajednički naziv za sve registrirane korisnike, urednike, administratore i vlasnika)

Aktorima se nazivaju oni koje vrše direktnu komunikaciju sa sustavom. To mogu biti inicijatori, koji pokreću procese u sustavu, ili sudionici, koji obavljaju zadane poslove.

Aktori i njihovi funkcionalni zahtjevi su:

- **vlasnik sustava**, inicijator:
  - može uređivati podatke o postaji
  - može postavljati administratore
  - može uređivati osobne podatke
  - može slušati program radio stanice
- **administrator**, inicijator:
  - može upravljati glazbenim urednicima
  - može upravljati zvučnim zapisima
  - može uređivati osobne podatke
  - može upravljati podacima drugih korisnika
  - može slušati program radio stanice

- **glazbeni urednik**, inicijator:
  - može slagati liste za izvođenje
  - može tražiti termine za reprodukciju
  - može uređivati osobne podatke
  - može slušati program radio stanice
- **registrirani korisnik**, inicijator:
  - može slagati listu želja
  - može uređivati osobne podatke
  - može slušati program radio stanice
- **korisnik**, inicijator
  - *apstraktni aktor, stvoren radi nasljeđivanja*
  - može uređivati osobne podatke
  - može slušati program radio stanice
- **posjetitelj**, inicijator:
  - može se registrirati
  - može slušati program radio stanice
- **baza podataka**, sudionik:
  - pohranjuje podatke o zvučnim zapisima
  - pohranjuje podatke o korisnicima
  - pohranjuje podatke o terminima izvođenja i zahtjevima za iste

4.1 Opis obrazaca uporabe

Napomena: Kako se radi o web aplikaciji, za sve obrasce uporabe nužan je preduvjet pristup Internetu

UC1 – RegistrirajNovogKorisnika

- **Glavni sudionik:** posjetitelj (neregistrirani korisnik)
- **Cilj:** stvoriti novi korisnički račun
- **Sudionici:** baza podataka
- **Preduvjeti:** nema ih
- **Rezultat:** stvoren je novi korisnički račun
- **Željeni scenarij:**
    1. Korisnik u odgovarajuća polja unosi svoje osobne podatke i email adresu te izabire lozinku
    2. Sustav provjerava točnost unesenih podataka, te koristi li se već odabrana email adresa
    3. Ako ne postoji, stvara se novi korisnički račun i na uneseni email se šalje pozdravna poruka s aktivacijskim linkom
    4. Klikom na taj link korisnik aktivira svoj račun te se sada može prijaviti u sustav
- **Mogući drugi scenariji:**
    1. Unesena email adresa se već koristi
        - Korisniku se dojavljuje greška i od njega se zahtjeva da odabere drugu email adresu

UC2 – PrijaviKorisnikaUSustav

- **Glavni sudionik:** korisnik

- **Cilj:** prijava u sustav
- **Sudionici:** baza podataka
- **Preduvjeti:** korisnik je registriran
- **Rezultat:** korisnik je prijavljen u sustav i sada su mu dostupne sve njegove mogućnosti
- **Željeni scenarij:**
    1. Korisnik unosi svoju email adresu i lozinku
    2. Sustav provjerava ispravnost unesenih podataka
    3. Ako uneseni podaci odgovaraju podacima korisničkog računa, korisnik se prijavljuje u sustav
- **Mogući drugi scenariji:**
    1. Uneseni su neispravni podaci
        - Korisniku se prikazuje odgovarajuća poruka o grešci i vraća ga se na prijavni obrazac

UC3 – UrediOsobnePodatke

- **Glavni sudionik:** korisnik
- **Cilj:** urediti osobne podatke
- **Sudionici:** baza podataka
- **Preduvjeti:** korisnik je prijavljen u sustav
- **Rezultat:** korisnik je izmijenio svoje osobne podatke
- **Željeni scenarij:**
    1. Korisniku se prikažu njegovi osobni podaci s mogućnošću promjene
    2. Korisnik mijenja neke od podataka te inicira pohranjivanje promjena
    3. Sustav vrši provjeru ispravnosti unesenih podataka
    4. Ako su uneseni podaci ispravni, pohranjuju se u sustav i korisniku se prikazuje poruka o uspjehu
- **Mogući drugi scenariji:**

1. Neki od unesenih podataka su neispravni
    - Korisniku se prikazuje poruka o grešci i od njega se traži da unese ispravne podatke

UC4 – PromijeniLozinku

- **Glavni sudionik:** korisnik
- **Cilj:** promijeniti lozinku
- **Sudionici:** baza podataka
- **Preduvjeti:** korisnik je prijavljen u sustav
- **Rezultat:** korisnik je promijenio svoju lozinku
- **Željeni scenarij:**
    1. Korisnik unosi redom svoju staru lozinku, novo-odabranu lozinku, te još jednom novo-odabranu lozinku kako bi potvrdio promjenu
    2. Sustav provjerava ispravnost stare lozinke, kao i jednakost dviju unesenih novih lozinki
    3. Ako su svi uneseni podaci ispravni, sustav pohranjuje promjenjenu lozinku
- **Mogući drugi scenariji:**
    1. Nisu uneseni ispravni podaci
        - Korisniku se prikazuje odgovarajuća poruka o grešci i od njega se traži da unese valjane podatke

UC5 – IzbrišiKorisničkiRačun

- **Glavni sudionik:** korisnik
- **Cilj:** izbrisati korisnički račun
- **Sudionici:** baza podataka
- **Preduvjeti:** korisnik je prijavljen u sustav, korisnik nije vlasnik sustava
- **Rezultat:** korisnički račun više ne postoji
- **Željeni scenarij:**

1. Korisnika unosi svoju lozinku (sigurnosna mjera)
2. Korisnik inicira brisanje korisničkog računa
3. Ako je lozinka ispravna, korisnički se račun zauvijek briše

- **Mogući drugi scenariji:**
  1. Korisnik nije unio ispravnu lozinku
     - U tom slučaju prikazuje mu se odgovarajuća poruka o grešci i od njega se traži da unese ispravnu lozinku

UC6 – SastaviListuŽelja

- **Glavni sudionik:** registrirani korisnik
- **Cilj:** izrada liste glazbenih želja
- **Sudionici:** baza podataka
- **Preduvjeti:** korisnik je prijavljen u sustav
- **Rezultat:** korisnik je sastavio i pohranio listu želja
- **Željeni scenarij:**
  1. Korisnik pregledava svoju listu želja (ako već postoji, ako ne, onda je prazna) te u nju unosi izmjene (dodaje ili briše pjesme, pri čemu se može služiti pretraživanjem pjesama)
  2. Kada je napravio sve planirane izmjene, inicira pohranjivanje
  3. Sustav pohranjuje korisnikovu listu želja

UC7 – PotvrdiListuŽelja

- **Glavni sudionik:** registrirani korisnik
- **Cilj:** potvrditi listu želja i time ju učiniti globalno dostupnom
- **Sudionici:** baza podataka
- **Preduvjeti:** korisnik je prijavljen u sustav, korisnik u protekla 24 sata već nije potvrđivao listu želja

- **Rezultat:** lista želja je potvrđena, želje za pjesmama na listi su sada vidljive glazbenim urednicima i administratorima
- **Željeni scenarij:**
    1. Korisnik pregledava svoju listu želja
    2. Ako je zadovoljan s njome, potvrđuje ju
    3. Sustav pohranjuje korisnikove želje u globalnu listu

UC8 – ZatražiTerminZaReprodukciju

- **Glavni sudionik:** glazbeni urednik
- **Cilj:** zatražiti dodjelu termina (jednog ili više) za reprodukciju od administratora
- **Sudionici:** baza podataka
- **Preduvjeti:** glazbeni urednik je prijavljen u sustav
- **Rezultat:** zahtjev za dodjelom termina je uspješno pohranjen u sustav
- **Željeni scenarij:**
    1. Glazbeni urednik pregledava kalendar sa označenim slobodnim terminima
    2. Urednik odabire termin(e) koji mu odgovara(ju) i šalje zahtjev za njima
    3. Sustav pohranjuje urednikov zahtjev

UC9 – SastaviListuZaReprodukciju

- **Glavni sudionik:** glazbeni urednik
- **Cilj:** sastaviti listu pjesama za reprodukciju za dani termin (trajanje 1 sat)
- **Sudionici:** baza podataka
- **Preduvjeti:** glazbeni urednik je prijavljen u sustav, dodijeljen mu je termin, do trenutka emitiranja ima više od 24 sata vremena
- **Rezultat:** lista zapisa za dani termin je sastavljena
- **Željeni scenarij:**
    1. Urednik pregledava i pretražuje pjesme, uzimajući u obzir korisničke želje
    2. Urednik odabire pjesme koje će se reproducirati u danom terminu

3. Urednik inicira pohranu liste
4. Sustav ispituje ispravnost sastavljene liste (trajanje najmanje 1 sat, zadnja pjesma ne počinje unutar 15 sekundi od kraja termina)
5. Ako su uvjeti zadovoljeni, lista se pohranjuje u sustav

- **Mogući drugi scenariji:**
    1. Nisu zadovoljeni uvjeti za listu
        - Korisniku se prikazuje odgovarajuća poruka o pogrešci i od njega se traži da sastavi ispravnu listu

UC10 – DodajZvučniZapis

- **Glavni sudionik:** administrator
- **Cilj:** u sustav dodati novi zvučni zapis
- **Sudionici:** baza podataka
- **Preduvjeti:** administrator je prijavljen u sustav
- **Rezultat:** u sustav je dodan novi glazbeni zapis sa svim bitnim podacima
- **Željeni scenarij:**
    1. Administrator unosi sve bitne podatke o zvučnom zapisu
    2. Administrator prilaže datoteku zvučnog zapisa
    3. Sustav provjerava ispravnost unesenih podataka
    4. Ako su podaci ispravni, pohranjuju se u sustav zajedno sa samom datotekom zapisa
- **Mogući drugi scenariji:**
    1. Uneseni su neispravni podaci
        - Prikazuje se odgovarajuća poruka o grešci i od administratora se traži da unese ispravne podatke

    1. Slanje datoteke zvučnog zapisa na sustav nije uspjelo

    - Prikazuje se odgovarajuća poruka o pogrešci i traži se ponovno obavljanje akcije

UC11 – UrediZvučniZapis

- **Glavni sudionik:** administrator
- **Cilj:** urediti podatke o zvučnom zapisu
- **Sudionici:** baza podataka
- **Preduvjeti:** administrator je prijavljen u sustav
- **Rezultat:** podaci o zvučnom zapisu su izmijenjeni
- **Željeni scenarij:**
    1. Administrator odabire zvučni zapis kojeg želi izmijeniti
    2. Administrator pregledava podatke o zvučnom zapisu i po želji ih mijenja
    3. Administrator inicira pohranu podataka
    4. Sustav ispituje ispravnost unesenih podataka
    5. Ako su podaci ispravni, pohranjuju se u sustav
- **Mogući drugi scenariji:**
    1. Uneseni su neispravni podaci
        - Prikazuje se odgovarajuća poruka o grešci, te se od administratora traži unos ispravnih podataka

UC12 – ObrišiZvučniZapis

- **Glavni sudionik:** administrator
- **Cilj:** obrisati zvučni zapis
- **Sudionici:** baza podataka
- **Preduvjeti:** administrator je prijavljen u sustav
- **Rezultat:** zvučni zapis je zauvijek izbrisan
- **Željeni scenarij:**
    1. Administrator odabire zvučni zapis kojeg želi izbrisati

2. Administrator inicira brisanje zvučnog zapisa
3. Sustav briše zvučni zapis

UC13 – UrediPodatkeKorisnika

- **Glavni sudionik:** administrator
- **Cilj:** urediti podatke korisnika
- **Sudionici:** baza podataka
- **Preduvjeti:** administrator je prijavljen u sustav, korisnik čiji se podaci uređuju nije vlasnik niti administrator
- **Rezultat:** korisnikovi podaci su uređeni
- **Željeni scenarij:**
    1. Administrator odabire korisnika kojem želi izmijeniti podatke
    2. Administrator pregledava podatke korisnika i po želji unosi promjene
    3. Administrator inicira spremanje promjena
    4. Sustav pohranjuje promjene

UC14 – PostaviGlazbenogUrednika

- **Glavni sudionik:** administrator
- **Cilj:** postaviti novog urednika
- **Sudionici:** baza podataka
- **Preduvjeti:** administrator mora biti prijavljen u sustav
- **Rezultat:** postavljen je novi urednik
- **Željeni scenarij:**
    1. Administrator pregledava popis korisnika
    2. Odabire jednog od korisnika i postavlja ga za glazbenog urednika
    3. Sustav bilježi promjene i mijenja tip korisničkog računa odabranog korisnika

UC15 – UkloniGlazbenogUrednika

- **Glavni sudionik:** administrator
- **Cilj:** ukloniti glazbenog urednika
- **Sudionici:** baza podataka
- **Preduvjeti:** administrator mora biti prijavljen u sustav
- **Rezultat:** glazbenom uredniku oduvima se uredički status
- **Željeni scenarij:**
    1. Administrator određuje urednika kojem želi oduzeti uredničke ovlasti
    2. Sustav bilježi promjene i mijenja tip korisničkog računa odabranog urednika

UC16 – OdlučiOZahtjevuZaTerminom

- **Glavni sudionik:** administrator
- **Cilj:** odlučiti o uredničkom zahtjevu za terminom
- **Sudionici:** baza podataka
- **Preduvjeti:** administrator mora biti prijavljen u sustav
- **Rezultat:** uredički zahtjev je ili prihvaćen i time je taj termin dodjeljen tom uredniku, ili je odbijen
- **Željeni scenarij:**
    1. Administrator odlučuje o prihvaćanju ili odbijanju zahtjeva
    2. Ako je zahtjev prihvaćen, taj se termin dodjeljuje uredniku, što se bilježi u sustavu

UC17 – PrikažiStatistiku

- **Glavni sudionik:** administrator
- **Cilj:** pregledati statistike o radu postaje
- **Sudionici:** baza podataka
- **Preduvjeti:** administrator je prijavljen u sustav
- **Rezultat:** administrator je dobio uvid u statistike radio postaje
- **Željeni scenarij:**

1. Administrator odabire jednu od ponuđenih statistika
2. Administratoru se prikazuje odabrana statistika

UC18 –PostaviAdministratora

- **Glavni sudionik:** vlasnik postaje
- **Cilj:** postaviti novog administratora
- **Sudionici:** baza podataka
- **Preduvjeti:** ne smije biti postavljeno više od deset administratora, vlasnik je prijavljen u sustav
- **Rezultat:** postavljen je novi administrator
- **Željeni scenarij:**
    1. Vlasnik odabire jednog od korisnika i dodjeljuje mu administratorske ovlasti
    2. Sustav bilježi promjene i mijenja tip korisničkog računa odabranog korisnika
- **Mogući drugi scenariji:**
    1. U sustavu već postoji 10 administratora
        - Akcija se ne dozvoljava, ispisuje se odgovarajuća poruka o grešci

UC19 – UkloniAdministratora

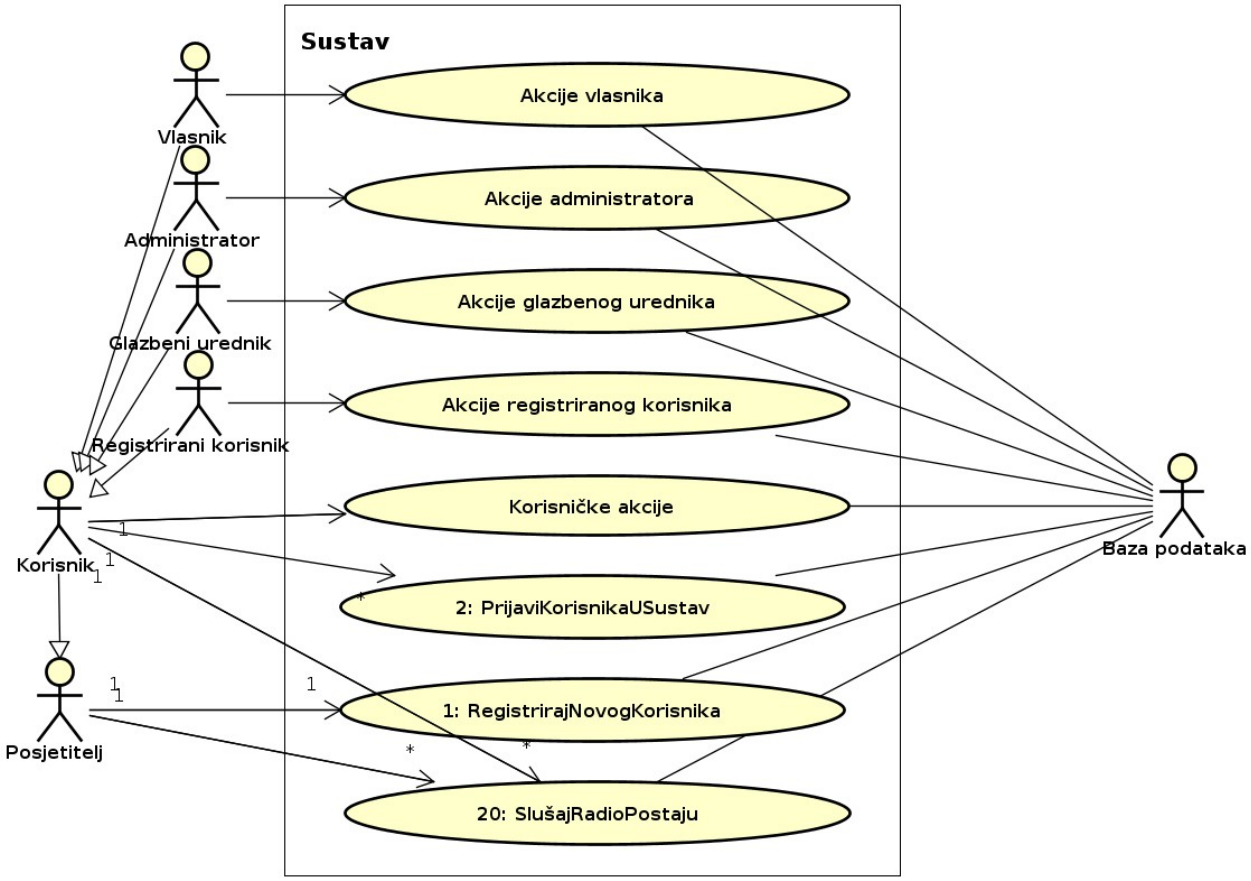
- **Glavni sudionik:** vlasnik postaje
- **Cilj:** ukloniti administratora
- **Sudionici:** baza podataka
- **Preduvjeti:** vlasnik postaje je prijavljen u sustav
- **Rezultat:** odabranom korisniku ukinute su administratorske ovlasti
- **Željeni scenarij:**
    1. Vlasnik postaje odabire jednog od administratora i uklanja mu administratorske ovlasti
    2. Sustav bilježi promjene i mijenja tip korisničkog računa odabranog korisnka

UC20 – UrediPodatkeOPostaji

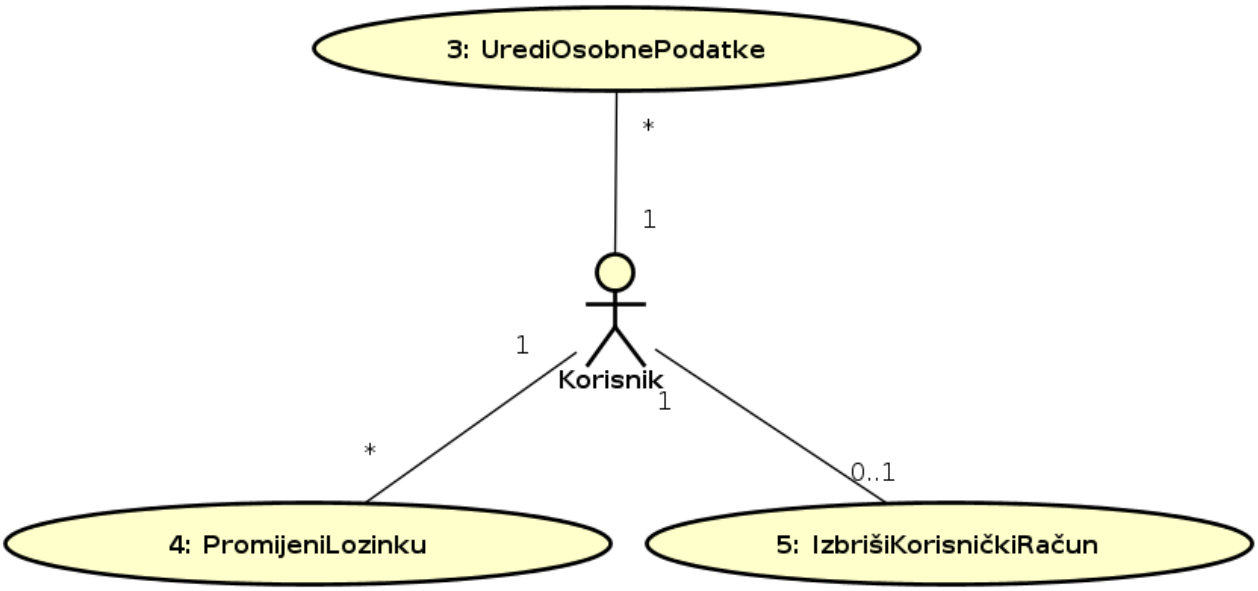
- **Glavni sudionik:** vlasnik postaje
- **Cilj:** unijeti ili urediti podatke
- **Sudionici:** baza podataka
- **Preduvjeti:** vlasnik je prijavljen u sustav
- **Rezultat:** uneseni su novi podaci o postaji
- **Željeni scenarij:**
    1. vlasnik unosi nove podatke, ili uređuje stare, te potvrđuje promjene
    2. baza podataka pohranjuje i čuva nove podatke
- **Mogući drugi scenariji:**
    1. vlasnik odustaje od izmjena

UC21 – SlušajRadioPostaju

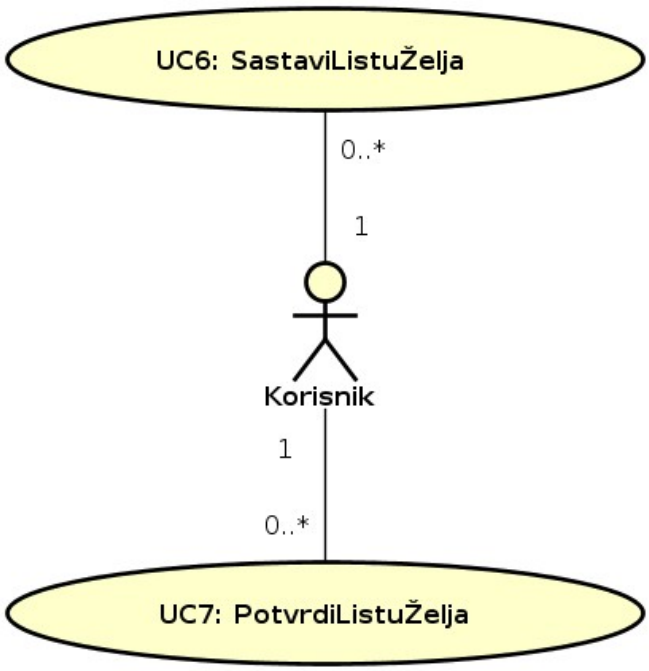
- **Glavni sudionik:** korisnik
- **Cilj:** slušati program radio postaje
- **Sudionici:** baza podataka
- **Preduvjeti:** nema
- **Rezultat:** korisnik uživa u programu radio postaje
- **Željeni scenarij:**
    1. Korisnik otvara web stranicu radio postaje
    2. Pokreće glazbeni player na početnoj stranici i započinje slušati program radio postaje



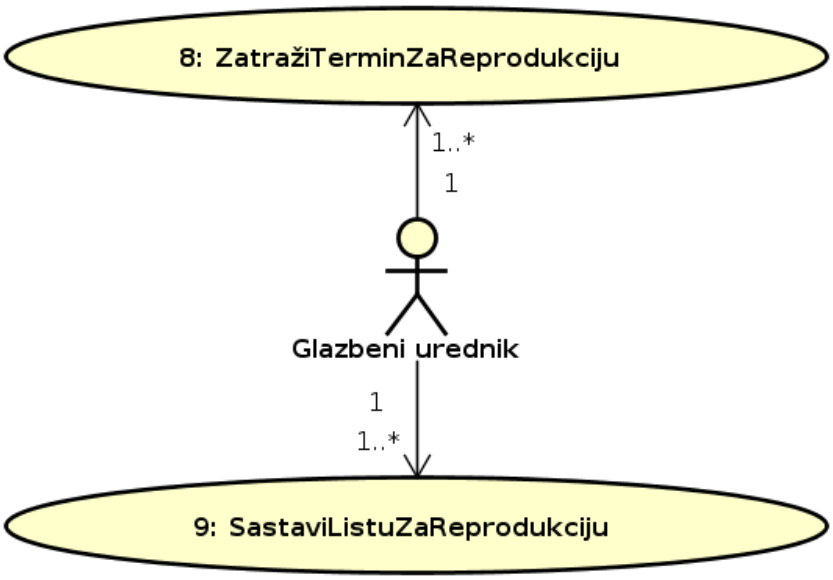
Slika 1: Dijagram obrazaca uporabe



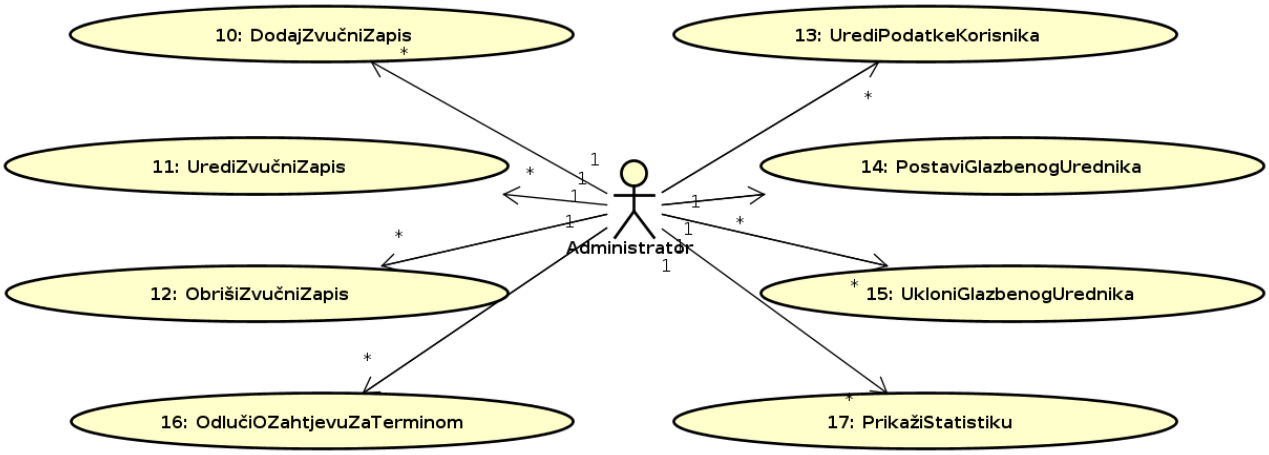
Slika 2: Dijagram obrazaca korisničkih akcija



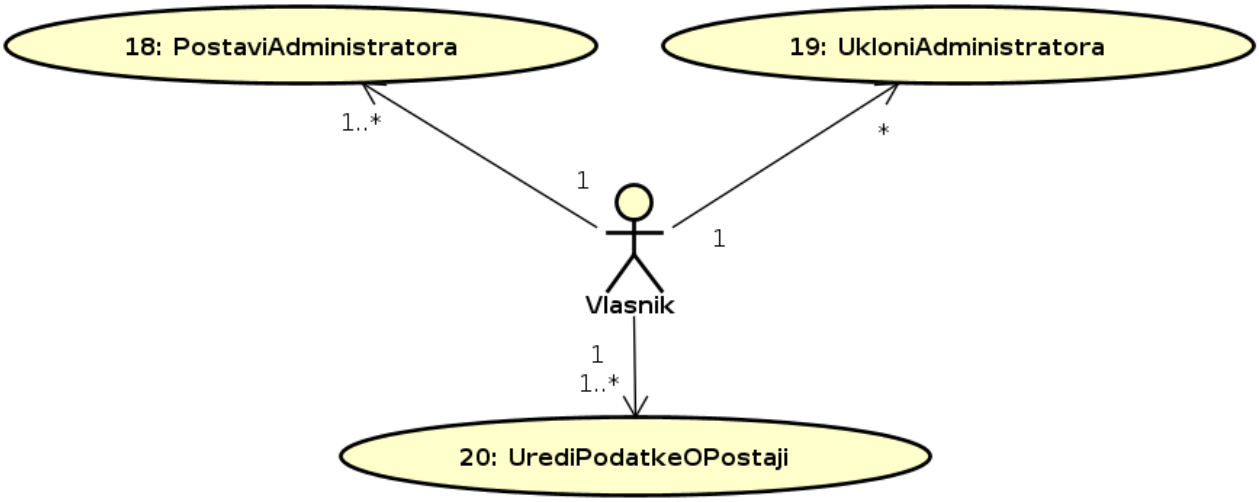
Slika 3: Dijagram obrazaca akcija registriranog korisnika



Slika 4: Dijagram obrazaca akcija glazbenog urednika



Slika 5: Dijagram obrazaca akcija administratora



Slika 6: Dijagram obrazaca akcija vlasnika

4.2 Sekvencijski dijagrami

5. Ostali zahtjevi

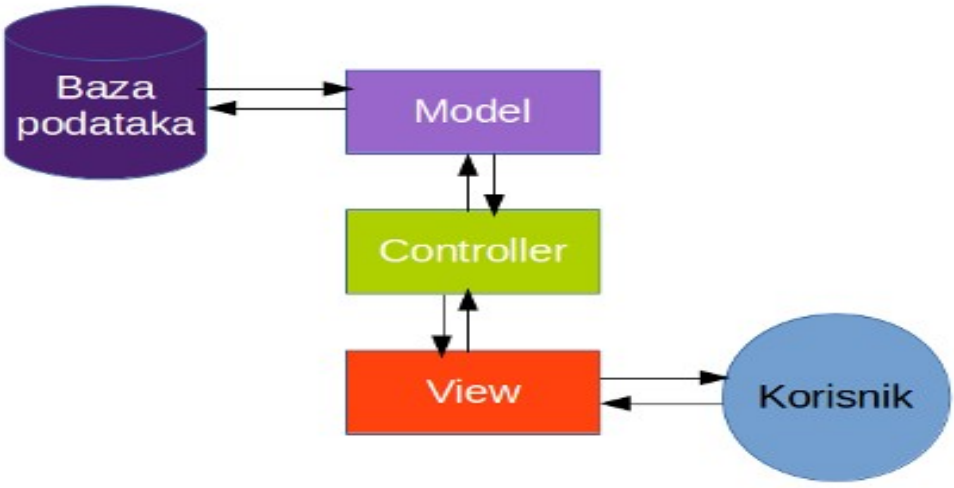
1. Sustav mora podržavati neograničen broj registriranih korisnika.
2. Postaja mora poštovati autorska prava i emitirati samo zapise koji su legalno dostupni.
3. Veoma je poželjna zastupljenost što više glazbenih žanrova, kako bi postaja privukla što više slušatelja.
4. Podaci o glazbenim zapisima moraju biti točni i pravilno uneseni.
5. Sustav mora biti jasan i pregledan, kako bi se korisnici lako snašli i mogli iskoristiti sve funkcionalnosti sustava bez obzira na razinu informatičke pismenosti.
6. Sustav mora svim aktorima omogućavati istovremeno korištenje svih funkcionalnosti; korisničko iskustvo ne smije biti ometeno zbog održavanja baze ili unošenja promjena u nju.

6. Arhitektura i dizajn sustava

6.1. Svrha, opći prioriteti i skica sustava

Kako je cilj ovog projekta napraviti informacijski sustav za internetsku radio postaju, prirodno se nameće ideja da se isti izradi u obliku **web aplikacije**. Prednosti takve arhitekture sustava nad primjerice arhitekturom **desktop klijent–poslužitelj** su prenosivost (svaku računalo danas ima web preglednik), jednostavnost korištenja (korisnici su naviknuti na rad u web pregledniku), kao i jednostavnost izrade te održavanja (laka izrada sučelja u HTML-u i CSS-u, mnogobrojni resursi za pomoć i podršku). Također, ako dođe do daljnjeg razvoja aplikacije, zbog centraliziranosti neće doći do problema s fragmentacijom – svi će korisnici koristiti istu, najnoviju inačicu aplikacije.

Aplikacija će biti podijeljena u dva dijela, jedan koji će se pokretati unutar web preglednika te drugi koji će se pokretati na poslužitelju i komunicirati s bazom podataka. Komunikacija između ta dva dijela aplikacije vršit će se putem AJAX zahtjeva, prema modelu REST sučelja.



Slika 1: Dijagram MVC obrasca

Oba dijela aplikacije bit će oblikovana prema *Model-View-Controller* oblikovnom obrascu, koji odvaja pojedine dijelove aplikacije ovisno o namjerni na modele koji opisuju podatke i operacije nad njima, poglede (*views*) koji su zaduženi za prikaz podataka korisnicima, te upravitelje (*controllers*) koji upravljaju korisničkim zahtjevima.

Klijentski dio

Klijentski dio aplikacije će se bazirati na AngularJS frameworku kao središnjem dijelu koji će obavljati dvosmjernu komunikaciju između modela i pogleda te će se brinuti za integritet i funkcionalnost same aplikacije. On će s poslužiteljskim dijelom komunicirati preko REST sučelja kojem će slati zahtjeve za raznim entitetima iz modela.

Angular2 karakterizira razdjeljivanje aplikacije u module koji se mogu smatrati zasebnim cjelinama te se uklapaju jedni u druge. Svaki modul ima svoj djelomični pogled koji se umeće u vanjski predložak na odgovarajuće mjesto i može imati svoj zaseban dizajn i logiku. Za interoperabilnost modula se koristi *dependency injection* mehanizam.

Za dizajniranje korisničkog sučelja koristit ćemo SASS koji je nadgradnja CSS-a i nudi brojne napredne funkcionalnosti, poput varijabli, hijerarhija elemenata, petlji i funkcija. Za strukturiranje korisničkog sučelja koristit će se JADE, a to je jezik koji se kompilira u HTML. On također predstavlja dodatan sloj funkcionalnosti te omogućava petlje, funkcije jednostavniju sintaksu, i još mnogo toga.

Za dizajniranje korisničkog sučelja ćemo koristiti SASS koji je „nadjezik“ CSS-a i nudi razne napredne funkcionalnosti kao što su varijable, hijerarhija elemenata, petlje i funkcije. Za stukturiranje korisničkog sučelja ćemo koristiti Jade, a to je jezik koji se kompilira u HTML. On također predstavlja dodatan sloj funkcionalnosti i također omogućava, petlje, jednostavniju sintaksu, funkcije i još mnogo toga.

Za upravljanjem korisničkim sučeljem ćemo koristiti Typescript koji je nadjezik JavaScripta i pruža dodatne funkcionalnosti koje su u nacrtu ECMAScript 6, a još nisu implementirane nativno u browserima. Također je mnogo veća podrška dokumentacijom za Angular2 i TypeScript nego li je za JavaScript ili Dart.

Komponente

Komponente koje ćemo implementirati se poklapaju s funkcijskim zahtjevima od same aplikacije budući da se na taj način modularizira klijentska strana. Komponente će biti ugniježđene u nadkomponente koje odgovaraju ulogama koje postoje u našoj aplikaciji. (Za detaljan popis svih komponenti pogledajte u popis funkcijskih zahtjeva) Nadkomponente će biti *Korisnik*, *Administrator*, *Vlasnik* … , a komponente će biti *Listen*, *Settings*¸ *Wishlist* od *Korisnik*, i tako dalje će se poklapati s funkcijskim zahtjevima i za

ostale nadkomponente.

Poslužiteljski dio

Poslužiteljski dio aplikacije bit će oblikovan kao *REST* sučelje koje će primati zahtjeve od klijentskog dijela, obaviti odgovarajuće akcije i potom vratiti rezultate klijentskom dijelu koji će ih prikazati korisniku. Ta komunikacija obavljat će se putem *HTTP* zahtjeva i odgovora, unutar kojih će podaci biti zapisani u *JSON* formatu.

Ovaj dio aplikacije bit će također razdijeljen prema MVC obrascu, no kako u općenitom slučaju on neće biti zadužen za prikaz podataka korisniku, uglavnom će sadržavati samo modele i upravitelje. Ulogu korisnika u interakciji s njime imat će klijentski dio aplikacije.

Poslužiteljski dio aplikacije vršit će i komunikaciju s bazom podataka, u koju će se pohranjivati svi podaci o korisnicima, zvučnim zapisima, listama želja i listama za reprodukciju, i svemu ostalom što je potrebno za rad aplikacije. No ta komunikacija neće biti izravna, već će se za nju pobrinuti *Peewee ORM*, biblioteka koja će iz entiteta definiranih kao razredi u programskom jeziku *Python* generirati odgovarajuće *SQL* tablice, kao i metode kojima će se implicitno vršiti spremanje, dohvaćanje i izmjena podataka u tablici.

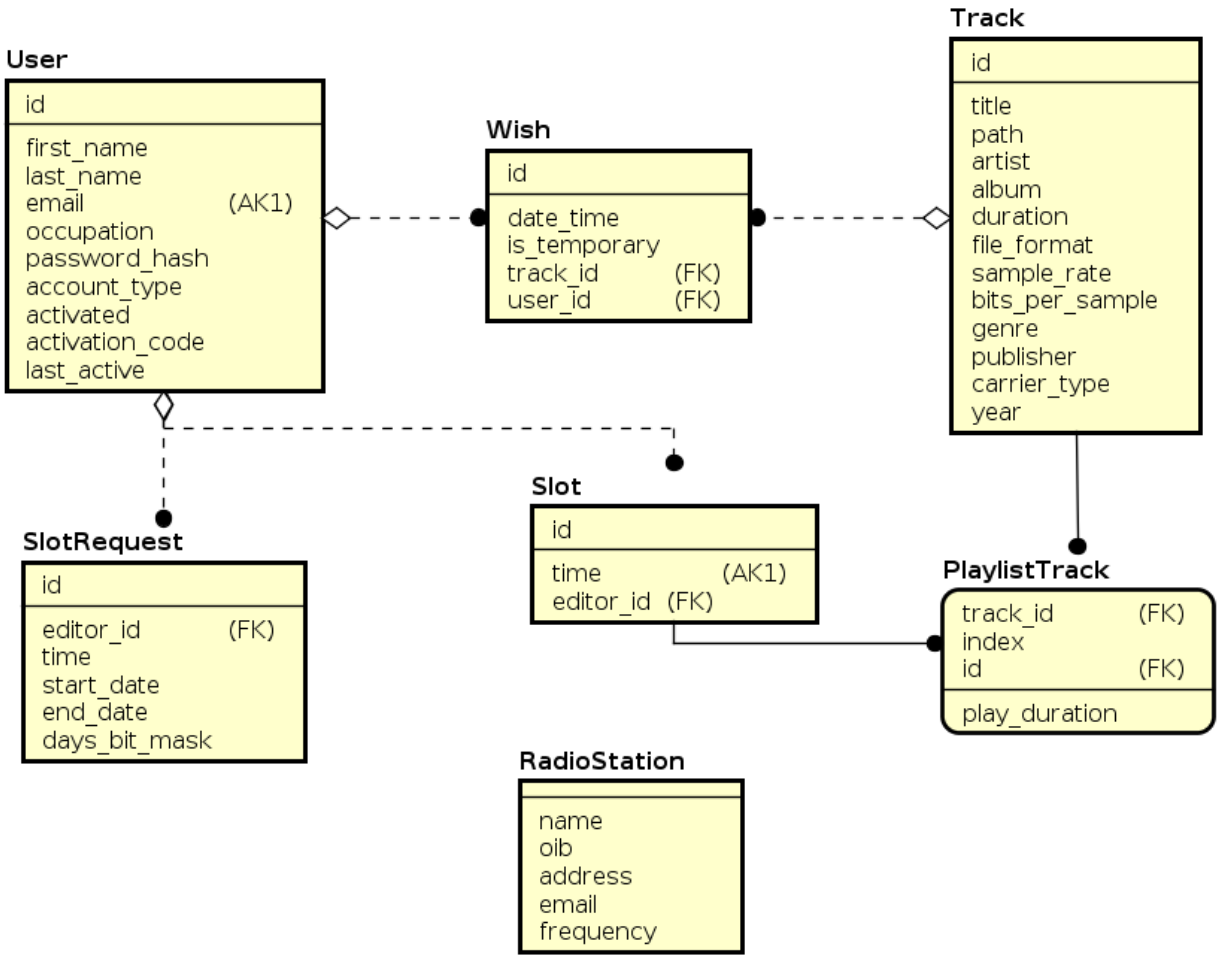
Za izradu poslužiteljskog dijela odabran je programski jezik *Python* te njegov framework *Flask*, koji se odlikuje velikom jednostavnošću, brzinom i lakoćom učenja, a za komunikaciju s bazom podataka odabran je već spomenuti *Peewee*.

Modeli

U MVC obrascu, modeli predstavljaju podatke kojima aplikacija upravlja te akcije koje nad njima može izvršiti. Oni se definiraju kao razredi sa svojim atributima i metodama. Ova aplikacija sadržavat će sljedeće modele: *Track*, *User*, *Slot*, *SlotRequest*, *PlaylistTrack*, *Wish* te *RadioStation*. Značenje njihovih atributa opisano je prilikom definiranja strukture baze podataka, dok su sve moguće akcije odgovarajućeg modela opisane u odjeljku 6.2, prilikom opisa dijagrama razreda.

Baza podataka

Iz prethodno navedenih modela automatski će se generirati SQL tablice u bazi podataka, kao i izvršavati sve potrebne SQL operacije. Izgled nastale sheme baze podataka dan je na slici:



Slika 2: ER model baza podataka

Detaljniji opis pojedinih relacija i značenja njihovih atributa dan je u sljedećim tablicama:

User	Korisnik aplikacije
id	*ID korisnika,* ***primarni ključ***
first_name	*Ime korisnika*
last_name	*Prezime korisnika*
email	*Adresa e-pošte,* ***alternativni ključ***
occupation	*Zanimanje korisnika*

password_hash	*Hash vrijednost lozinke*
account_type	*Tip korisničkog računa,* (**1** – reg. korisnik, **2** – urednik, **3** - administrator, **4** – vlasnik)
activated	*Je li korisnički račun aktiviran*
activation_code	*Jedinstveni kod kojim se vrši aktivacija korisničkog računa*
last_active	*Vrijeme zadnje aktivnosti, potrebno za statistiku*

Track	**Zvučni zapis**
id	*ID zapisa,* ***primarni ključ***
title	*Naziv zvučnog zapisa*
path	*Putanja do datoteke na poslužitelju*
artist	*Umjetnik – autor zapisa*
album	*Album na kojemu je zvučni zapis*
duration	*Trajanje zvučnog zapisa u sekundama*
file_format	*Format u kojemu je pohranjen zapis (MP3, WAV, OGG...)*
sample_rate	*Učestalost uzorkovanja*
bits_per_sample	*Broj bitova po uzorku*
genre	*Žanr kojem pripada glazbeni zapis*
publisher	*Izdavač zapisa*
carrier_type	*Vrsta nosača zvuka*
year	*Godina izdavanja*

Slot	**Urednički termin**
Id	*ID termina,* ***primarni ključ***
time	*Datum i vrijeme početka termina,* ***alternativni ključ***
editor_id	*ID urednika kojemu je dodijeljen taj termin,* ***strani ključ***

SlotRequest	**Zahtjev za terminom**
id	*ID zahtjeva,* ***primarni ključ***
time	*Vrijeme termina u danu, obavezno počinje na puni sat*
editor_id	*ID urednika koji je zatražio taj termin,* ***strani ključ***
start_date	*Početni datum od kojega bi započinjao dodijeljeni termin*

end_date	*Konačni datum do kojega bi termin bio dodijeljen*
days_bit_mask	*Za koje sve dane u tjednu je zatražen termin (enkodirano kao bit-maska, spremljena kao cijeli broj, npr.* 19 = 0010011_2 = pon., uto. i pet.*)*

Wish	**Korisnička želja**
id	*ID želje,* ***primarni ključ***
track_id	*ID zapisa koji je željen,* ***strani ključ***
user_id	*ID korisnika koji je izrazio želju,* ***strani ključ***
date_time	*Datum i vrijeme kada je želja izražena*
is_temporary	*Je li ta želja privremena (vidljiva samo korisniku i podložna promjenama), ili je već potvrđena (konačna, vidljiva administratorima i urednicima)*

PlaylistTrack	**Zapis na listi za reprodukciju**
track_id	*ID zapisa,* ***strani ključ***
time	*Vrijeme početka termina,* ***strani ključ*** *relacije* ***Slot***
index	*Redni broj pjesme u tom terminu*
play_duration	*Koliko će se dugo svirati ovaj zapis*
K = { track_id, time, index }	*Složeni primarni ključ*

RadioStation	**Podaci o radio postaji**
name	*Naziv radio postaje*
oib	*OIB radio postaje kao pravne osobe*
address	*Adresa*
email	*Adresa e-pošte*
frequency	*Frekvencija na kojoj se odašilje program radio postaje*

Napomene:

1. *U ovoj se tablici smije nalaziti samo jedan redak*
2. *Tablica stoga nema primarni ključ*

Za potrebe ove web aplikacije konkretno ćemo koristiti *SQLite* bazu podataka, koja je prikladna za manje aplikacije s ograničenom količinom prometa. No zbog korištenja *Peewee* biblioteke, ako se u budućnosti pojavi potreba za podržavanjem mnogo većeg prometa, moguć je vrlo jednostavan prijelaz na neki od moćnijih sustava kao što je *PostgreSQL*, bez ikakvih promjena koda same aplikacije.

Upravitelji

Za razliku od nekih drugih jezika i frameworka, u Pythonu i Flasku upravitelji su jednostavne funkcije, te su stoga opisani ovdje, a ne u odjeljku o dijagramima razreda.

Kako je poslužiteljski dio ove aplikacije modeliran kao *REST* sučelje, tako svaki pojedini *URL* koji predstavlja neku akciju ima svog upravitelja, koji prihvaća i obrađuje zahtjeve koje mu klijent pošalje, te na njih odgovara u dogovorenom obliku (JSON format).

Osim *REST* upravitelja, postoji i nekoliko njih drugačijeg tipa, koji ne odgovaraju na zahtjeve klijenta, već služe prikazivanju same web stranice, tj. njena dva dijela, početne stranice te stranice s postavkama i korisničkim mogućnostima.

Upravitelji koje sadrži poslužiteljski dio aplikacije, grupirani prema zajedničkim objektima djelovanja, su (uz naziv upravitelja nalazi se i njegov *URL*, u odnosu na temeljnu adresu aplikacije) :

Općenito, prikaz stranica i slušanje programa radio postaje

- **show_index()** *"/"*
  - Prikazuje početnu stranicu, na kojoj su glazbeni player za slušanje radio postaje, obrazac za prijavu i registraciju te osnovni podaci o radio postaji
- **show_settings()** *"/settings"*
  - *Za registrirane korisnike.* Prikazuje glavnu stranicu web aplikaciju, na kojoj su sve postavke i mogućnosti korisnika.
- **get_currently_playing_track()** *"/player/get"*
  - Služi za dohvaćanje zvučnog zapisa koji se trenutno pušta na radio postaji

Prijava, registracija i aktivacija korisničkih računa

- **process_login()** *"/user/auth/login"*
  - Obrađuje korisnikov pokušaj prijave u sustav, ako je pokušaj uspješan, prijavljuje korisnika i preusmjerava ga natrag na početnu stranicu
- **process_registration()** *"/user/auth/register"*
  - Obrađuje korisnikov zahtjev za registracijom, ako je uspješan, stvara novog korisnika i šalje email s aktivacijskim linkom
- **process_activation()** *"/user/auth/activate"*

    - Aktivira korisnički račun putem aktivacijskog linka, čime omogućuje korisniku da se ubuduće prijavljuje u sustav
- **process_signout()** *"/user/auth/signout"*
    - Odjavljuje korisnika iz sustava

Upravljanje vlastitim korisničkim računom

- **get_account_data()** *"/user/account/get"*
    - Vraća sve podatke o trenutnom korisniku
- **modify_account_data()** *"/user/account/modify"*
    - Mijenja korisničke podatke trenutnog korisnika
- **delete_account()** *"/user/account/delete"*
    - Zauvijek briše korisnički račun trenutnog korisnika iz sustava
- **change_account_password()** *"/user/account/change_password"*
    - Mijenja lozinku trenutnog korisnika, uz sigurnosnu provjeru (ispitivanje stare lozinke)

Korisničko upravljanje listom želja

- **get_wishlist()** *"/user/wishlist/get"*
    - Vraća trenutnu listu želja korisnika
- **set_wishlist()** *"/user/wishlist/set"*
    - Sastavlja novu listu želja korisnika
- **confirm_wishlist()** *"/user/wishlist/confirm"*
    - Potvrđuje korisnikovu listu želja

Administratorsko upravljanje zvučnim zapisima

- **add_track()** *"/admin/tracks/add"*
    - Dodaje zvučnog zapisa u sustav, postavlja datoteku na poslužitelj i unosi sve bitne podatke o zapisu u sustav
- **edit_track(id)** *"/admin/tracks/<id>/edit"*
    - Uređuje podatke o zvučnom zapisu s danim ID-om
- **delete_track(id)** *"/admin/tracks/<id>/delete"*
    - Briše zvučni zapis s danim ID-om iz sustava

Administratorsko upravljanje urednicima

- **list_editors()** *"/admin/editors/list"*
  - Vraća popisa svih glazbenih urednika u sustavu
- **add_editor(id)** *"/admin/editors/add/<id>"*
  - Postavlja jednog od registriranih korisnika (onog s danim ID-om) za novog glazbenog urednika
- **remove_editor(id)** *"/admin/editors/<id>/remove"*
  - Ukida urednički status korisniku s danim ID-om

Administratorsko upravljanje zahtjevima za terminima

- **list_requests()** *"/admin/requests/list"*
  - Vraća popis svih trenutno aktivnih zahtjeva za dodjelom termina za reprodukciju
- **allow_request(id)** "/admin/requests/<id>/allow"
  - Potvrđuje zahtjev urednika za dodjelom nekog termina, taj se termin u sustavu bilježi kao zauzet
- **deny_request(id)** "/admin/requests/<id>/deny"
  - Odbija zahtjev urednika za dodjelom termina

Administratorsko upravljanje korisnicima

- **get_user_data(id)** *"/admin/users/<id>/get"*
  - Vraća podataka korisničkog računa s danim ID-om
- **modify_user_data(id)** *"/admin/users/<id>/modify"*
  - Izmjenjuje podatake korisničkog računa s danim ID-om
- **delete_user(id)** "/admin/users/<id>/delete"
  - Briše korisnika s danim ID-om iz sustava

Uredničko upravljanje terminima za reprodukciju

- **list_editor_slots()** *"/editor/slots/list"*

    - Vraća popis svih termina dodjeljenih trenutnom korisniku (koji je urednik)
- **request_slot()** *"/editor/slots/request"*
    - Stvara zahtjev za dodjelom određenog termina uredniku
- **get_list(id)** *"/editos/slots/<id>/get_list"*
    - Vraća trenutno spremljenu listu za reprodukciju u danom terminu
- **set_list(id)** *"/editor/slots/<id>/set_list"*
    - Sastavlja novu listu za reprodukciju za dani termin

Vlasničko upravljanje administratorima

- **list_admins()** *"/owner/admins/list"*
    - Vraća popis svih administratora u sustavu
- **add_admin(id)** *"/owner/admins/add/<id>"*
    - Postavlja korisnika s danim ID-om za administratora sustava (ako već nema 10 administratora)
- **remove_admin(id)** *"/owner/admins/<id>/remove"*
    - Uklanja administratorski status korisniku s danim ID-om

Vlasničko upravljanje radio stanicom

- **modify_station_data()** *"/owner/station/modify"*
    - Mijenja podatke o radio stanici

Pregled i pretraživanje zvučnih zapisa

- **list_tracks()** *"/tracks/list"*
    - Vraća popis svih zvučnih zapisa u sustavu
- **get_track(id)** *"/tracks/<id>/get"*
    - Vraća podatke o zvučnom zapisu s danim ID-om
- **search_tracks()** *"/tracks/search"*
    - Vraća popis svih zvučnih zapisa koji zadovoljavaju kriterije pretrage

Pregled raznih statistika

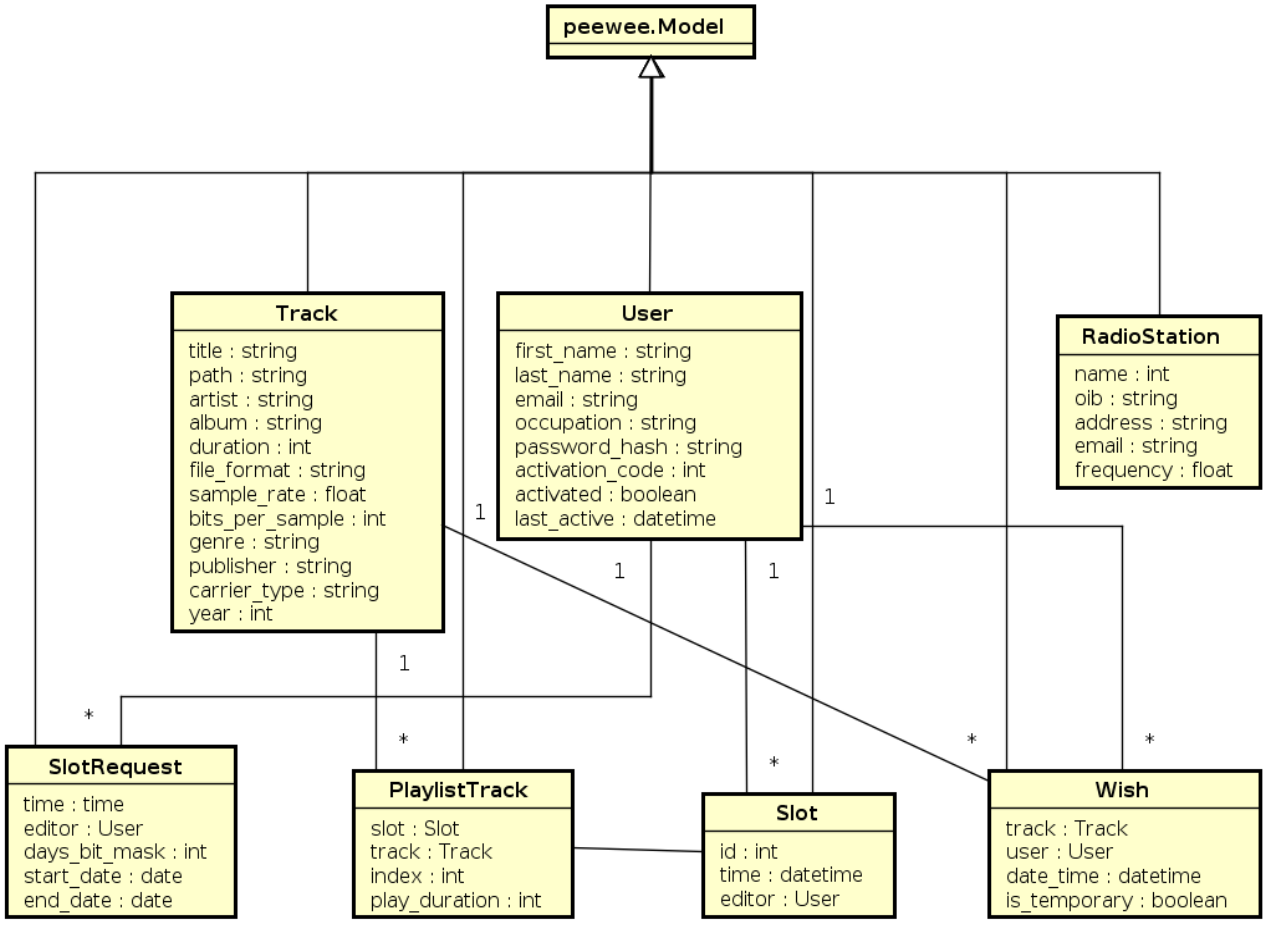
- **get_wishlist()** *"/stats/wishlist"*

    - Vraća popis svih zapisa koji su na listama želja korisnika, zajedno s brojačem pojavljivanja na listi. Moguće ograničavanje na neki vremenski interval.
- **get_active_users_count()** *"/stats/active_users/count"*
    - Vraća broj trenutno aktivnih korisnika u sustavu
- **get_active_admins_list()** *"/stats/active_admins/list"*
    - Vraća popis svih trenutno aktivnih administratora u sustavu
- **get_editor_preferred_track(id)** *"/stats/editor/<id>/preferred_tracks"*
    - Vraća popis zvučnih zapisa koje urednik s danim ID-om najčešće stavlja na liste za reprodukciju

6.2. Dijagram razreda s opisom

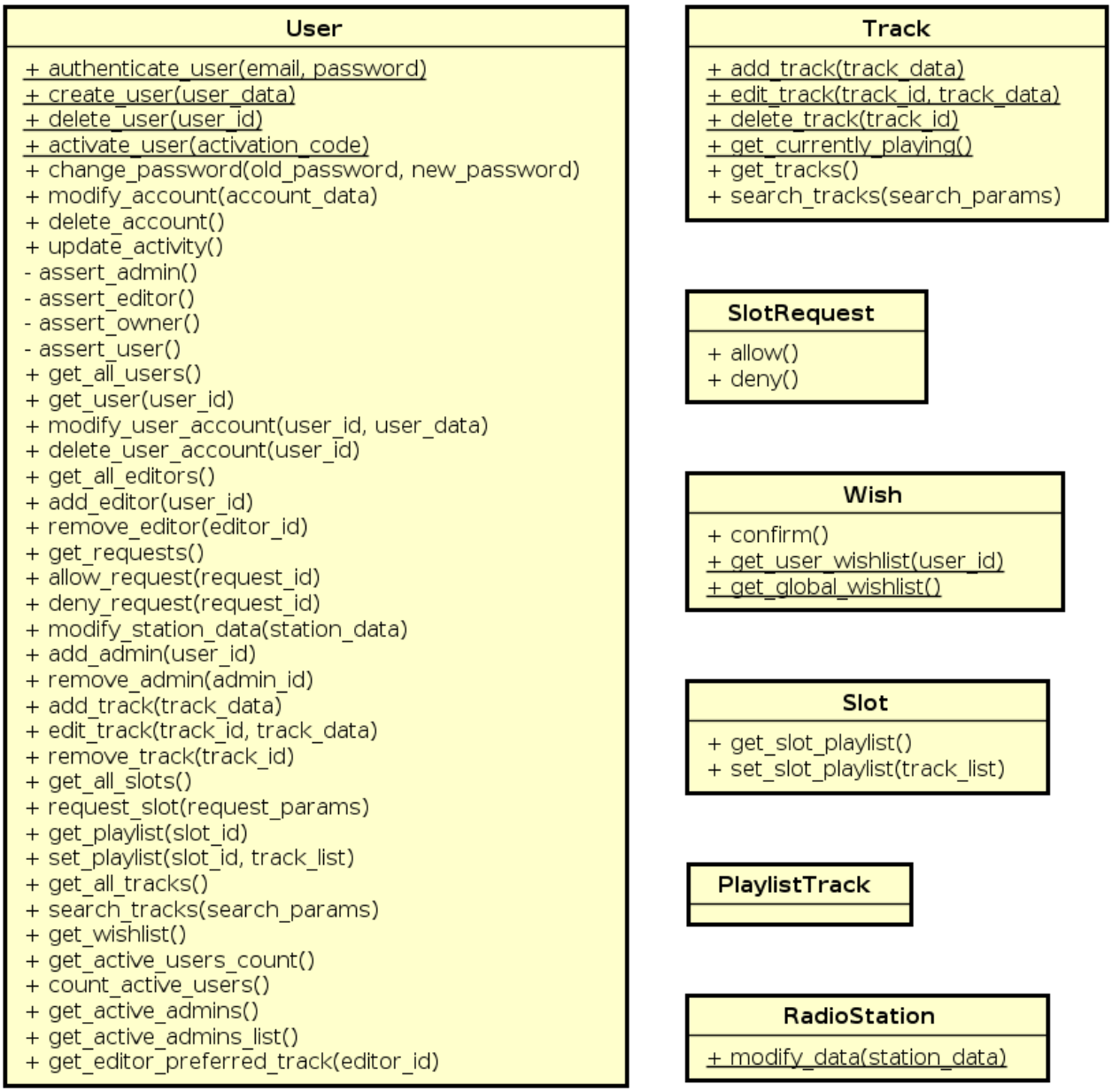
Napomena: U korištenom programskom jeziku, Pythonu, objektno orijentirana paradigma se upotrebljava na malo drugačiji način nego li primjerice u C#-u ili Javi, pa se tako ne koriste getter i setter metode, ne postoje eksplicitni modifikatori pristupa itd., što znači da će razredi možda biti drugačije strukturirani od "standardnog načina" na kojeg su mnogi naviknuti.

U oblikovanju ovog sustava koristi se objektno usmjerena paradigma, ponajprije pri izgradnji modela *MVC* obrasca. Svaki model oblikovan je kao poseban razred, s atributima čije je značenje navedeno pri opisivanju strukture baze podataka, te operacijama specifičnima za taj model. Modeli nasljeđuju razred *peewee.Model* koji pruža sve funkcionalnosti povezivanja s bazom podataka koja se izvršava implicitno, primjerice pozivima metoda *save()* ili *delete_instance()*. Prikaz nasljeđivanja i atributa svih modela dan je na slici 3.



Slika 3: Dijagram razreda – modeli – dio 1

Svaki model ima definirane osnovne CRUD (create, read, update, delete) operacije koje nasljeđuje od *peewee.Model* razreda, a dodatno i neke operacije specifične za taj model. Uz to, ključni dio logike cijele aplikacije sadržan je unutar modela *User*, koji sadrži operacije stvarnih korisnika, poput *create_wishlist*() ili *change_password()*. Popis svih operacija pojedinih modela dan je na slici 4.



Slika 4: Dijagram razreda – modeli – dio 2

6.3. Dijagram objekata

6.4. Ostali UML dijagrami

7. Implementacija i korisničko sučelje

7.1. Dijagram razmještaja

7.2. Korištene tehnologije i alati

7.3. Isječak programskog koda vezan za temeljnu funkcionalnost sustava

7.4. Ispitivanje programskog rješenja

7.5. Upute za instalaciju

7.6. Korisničke upute

8. Zaključak i budući rad

9. Popis literature

Dodatak A: Indeks (slika, dijagrama, tablica, ispisa koda)

Dodatak B: Dnevnik sastajanja

Dodatak C: Prikaz aktivnosti grupe