

# Programsko inženjerstvo

Ak. god. 2021./2022.

## PassDirect

Dokumentacija, Rev. 2

Grupa: *GabrijelovaDruzina*

Voditelj: *Gabrijel Jambrošić*

Datum predaje: *14. 1. 2022.*

Nastavnik: *Daria Primorac*

# Sadržaj

|                                                |           |
|------------------------------------------------|-----------|
| <b>1 Dnevnik promjena dokumentacije</b>        | <b>3</b>  |
| <b>2 Opis projektnog zadatka</b>               | <b>5</b>  |
| <b>3 Specifikacija programske potpore</b>      | <b>8</b>  |
| 3.1 Funkcionalni zahtjevi . . . . .            | 8         |
| 3.1.1 Obrasci uporabe . . . . .                | 10        |
| 3.1.2 Sekvencijski dijagrami . . . . .         | 17        |
| 3.2 Ostali zahtjevi . . . . .                  | 20        |
| <b>4 Arhitektura i dizajn sustava</b>          | <b>21</b> |
| 4.1 Baza podataka . . . . .                    | 22        |
| 4.1.1 Opis tablica . . . . .                   | 23        |
| 4.1.2 Dijagram baze podataka . . . . .         | 25        |
| 4.2 Dijagram razreda . . . . .                 | 27        |
| 4.3 Dijagram stanja . . . . .                  | 30        |
| 4.4 Dijagram aktivnosti . . . . .              | 32        |
| 4.5 Dijagram komponenti . . . . .              | 34        |
| <b>5 Implementacija i korisničko sučelje</b>   | <b>35</b> |
| 5.1 Korištene tehnologije i alati . . . . .    | 35        |
| 5.2 Ispitivanje programskog rješenja . . . . . | 36        |
| 5.2.1 Ispitivanje komponenti . . . . .         | 36        |
| 5.2.2 Ispitivanje sustava . . . . .            | 39        |
| 5.3 Dijagram razmještaja . . . . .             | 48        |
| 5.4 Upute za puštanje u pogon . . . . .        | 49        |
| 5.4.1 Postavljanje Herokua . . . . .           | 49        |
| <b>6 Zaključak i budući rad</b>                | <b>53</b> |
| <b>Popis literature</b>                        | <b>54</b> |

**Indeks slika i dijagrama** 56

**Dodatak: Prikaz aktivnosti grupe** 57

# 1. Dnevnik promjena dokumentacije

## *Kontinuirano osvježavanje*

| Rev. | Opis promjene/dodatka                                                                                 | Autori                                                    | Datum       |
|------|-------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|-------------|
| 0.1  | Napravljen predložak.                                                                                 | G. Jambrošić                                              | 19.10.2021. |
| 0.2  | Napisan opis projektnog zadatka.<br>Dodani funkcionalni zahtjevi.<br>Dodani obrasci uporabe za admina | P. Hrvoić,<br>I. Ivanković,<br>G. Jambrošić,<br>F. Marčec | 3.11.2021.  |
| 0.3  | Revizija strukture dokumenta,<br>uređivanje rasporeda slika i uklanjanje<br><i>placeholdera</i>       | L. Grgurić<br>Mileusnić                                   | 14.11.2021. |
| 0.4  | Završen opis tablica baze podatak.<br>Dodan dijagram baze podataka                                    | K. Jurič                                                  | 15.11.2021. |
| 0.5  | Dodani ostali zahtjevi sustava.<br>Manja revizija strukture dokumenta                                 | I. Ivanković                                              | 18.11.2021. |
| 0.6  | Dodan dijagram razreda                                                                                | K. Jurič                                                  | 18.11.2021. |
| 0.7  | Dodani dijagrami obrazaca uporabe.<br>Dodani sekvencijski dijagrami                                   | F. Mička                                                  | 19.11.2021. |

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

| Rev. | Opis promjene/dodatka                                                                  | Autori                                                                                                          | Datum       |
|------|----------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|-------------|
| 1.0  | Verzija samo s bitnim dijelovima za 1. ciklus                                          | L. Grgurić<br>Mileusnić, P.<br>Hrvoić, I.<br>Ivanković, G.<br>Jambrošić, K.<br>Jurič, F.<br>Marčec, F.<br>Mička | 19.11.2021. |
| 1.1  | Arhitektura sustava - update                                                           | I. Ivanković                                                                                                    | 29.12.2021. |
| 1.2  | Popravljeni dijagrami obrazaca uporabe, sekvencijski dijagram te funkcionalni zahtjevi | F. Mička, K.<br>Jurič, I.<br>Ivanković                                                                          | 10.1.2022.  |
| 1.3  | Dodan dijagram razmještaja                                                             | I. Ivanković                                                                                                    | 13.1.2022.  |
| 1.5  | Dodani dijagrami aktivnosti i stanja, te opis testiranja sustava                       | G. Jambrošić,<br>K. Jurič                                                                                       | 14.1.2022.  |
| 2.0  | Konačni tekst predloška dokumentacije                                                  | L. Grgurić<br>Mileusnić, P.<br>Hrvoić, I.<br>Ivanković, G.<br>Jambrošić, K.<br>Jurič, F.<br>Marčec, F.<br>Mička | 14.1.2022.  |

## 2. Opis projektnog zadatka

Cilj ovog projekta je razviti programsku podršku za stvaranje web aplikacije „PassDirect“ koja će omogućiti korisniku kupnju karte za vlak te upućivati putnike u odgovarajuće vagone vlakova u svrhu optimalne distribucije njihove težine.

Operateri vlakova, kao i njihovi održavatelji, svjesni su važnosti interakcije između vozila i pruge. Jedan od problema s kojim se suočavaju jest neravnomjerna raspodjela mase unutar vagona vlaka koja vodi do povećane istrošenosti kotača i slabljenja strukture osovine i time povećava vjerojatnost iskliznuća vlaka s tračnica. Nadalje, neoptimalna kvaliteta kotača i osovine također predstavlja veliki problem u kontekstu troškova održavanja, performansi i udobnosti putnika, što se na koncu odražava i na sliku koju javnost ima o željeznici.

Zajedničko rješenje navedenih problema ostvareno je korištenjem sustava senzora postavljenih na odabranim točkama duž tračnica kolosijeka. Nizozemske željeznice slove za jedne od najnaprednijih u Europi, dijelom zbog korištenja sustava *Gotcha* koji funkcionira na sljedeći način: pri prolasku vlaka, senzori mjere opterećenje kotača u stvarnom vremenu te se iz vremenskog niza očitanih podataka može procijeniti akumulativna šteta nastala na svakom kotaču i odrediti optimalan trenutak održavanja svakog vagona.



Slika 2.1: *Gotcha* sustav senzora

Opisani sustav predstavlja svojevrsnu revoluciju u održavanju vagona i tračnica, no ono što je zanimljivo u kontekstu projektnog zadatka jest mogućnost određivanja

najmanje okupiranog vagona u svrhu optimalne distribucije težine. Komunikacija između aplikacije i senzora koristi skup podataka koji pristiže od senzora, a koji u svakoj svojoj liniji sadrži identifikacijsku oznaku i brzinu vlaka koji je prošao, vrijeme mjerenja te niz prirodnih brojeva, po dva za svaki vagon. Ta mjerenja predstavljaju višak detektirane težine izražen u kilogramima.

Prilikom pokretanja sustava prikazuje se početni zaslon na kojem se nalaze pozdravna poruka i neke osnovne informacije o željeznici, sustavu senzora, sigurnosnim preporukama i sl. Pri vrhu stranice nalazi se gumb čiji pritisak vodi korisnika na prikaz voznog reda te gumbi za registraciju odnosno prijavu korisnika u sustav.

Aplikacija treba podržavati rad više korisnika u stvarnom vremenu. Neregistriranom korisniku omogućen je pregled voznog reda odnosno popisa ponuđenih stajališta vlaka.

Kod prikaza voznog reda, odabir pojedinog stajališta vodi korisnika na listu vlakova koji dolaze na to stajalište. Uz identifikacijsku oznaku vlaka prikazuje se i njegovo krajnje odredište, planirano vrijeme dolaska na tu stanicu i odlaska s nje te gumb koji omogućuje kupnju karte.

Svakom je korisniku u bilo kojem trenutku omogućeno klasično pretraživanje s filtriranjem koje uključuje odabir mjesta polaska i dolaska te datuma. Na zaslonu se potom izlistaju odgovarajuće linije vlakova sa svojim osnovnim informacijama i cijenom karte za svaku od njih.

Neregistriranom korisniku u bilo kojem trenutku omogućena je prijava u sustav s postojećim računom ili registracija novog računa. U tom slučaju potrebni su sljedeći podaci kako bi registracija bila valjana:

- ime
- prezime
- e-mail adresa
- lozinka

Registracijom u sustav korisniku se dodjeljuju prava putnika. Registriranom korisniku omogućeno je sve što i neregistriranom, uz dodatne mogućnosti pregleda i uređivanja osobnih podataka, brisanja korisničkog računa i kupnje vozne karte za vlak, bez mogućnosti rezervacije sjedala. Njemu je pri klasičnom pretraživanju uz cijenu karte prikazan gumb za kupnju karte pritiskom kojeg se na zaslonu otvara prozor za provođenje transakcije. U tom prozoru korisnik unosi svoje podatke potrebne za kupnju karte, a podaci ostaju zapamćeni kako ih pri sljedećoj kupnji karte ne bi morao ručno unositi.

Nakon uspješno provedene transakcije korisniku stižu dva e-maila. U prvom e-mailu, koji pristiže odmah nakon kupovine, nalazi se kupljena karta. Drugi e-mail pristiže nešto kasnije, odnosno nakon što vlak napusti prethodno stajalište. U njemu su sadržani informativni tekst, identifikacijska oznaka vlaka, vrijeme dolaska, kolosijek i uputa putniku u koji vagon i dio vagona bi se trebao ukrcati te na kojem peronu to može učiniti. Cilj aplikacije jest da se putnike upućuje na prazniji dio najmanje okupiranog vagona.

Posebna vrsta registriranog korisnika koji ima najveće ovlasti u sustavu jest administrator. Njemu je omogućen pristup bazi podataka s popisom svih registriranih korisnika, njihovim osobnim podacima i provedenim transakcijama. Također, administrator može pregledavati vozni red svih vlakova u sustavu.

Projektni zadatak moguće je proširiti na način da se omogući njegova primjena u realnom sektoru. U tom slučaju bilo bi potrebno (ili poželjno) ostvariti dodatne funkcionalnosti kao što su:

- mogućnost plaćanja putem sustava za online autorizaciju kreditnih kartica u realnom vremenu;
- dodavanje osnovnih informacija o pojedinom kolodvoru poput kontakt telefona ili e-mail adrese;
- odabir dodatnih mogućnosti pri kupnji karte kao što je razred;
- filtriranje pretraživanja - mogućnost odabira samo izravnih vlakova, brzih ili putničkih vlakova, povratnog putovanja itd.
- stranica s informacijama o zastoјima ili novim regulacijama u prometu;
- rezervacija sjedala u vagonu;
- karta s pregledom željezničkih stajališta;
- korisnička podrška poput *LiveChat*.

Iako je svrha izrade aplikacije prvenstveno optimalna distribucija težine po vagonima, navedena implementacija uvelike bi olakšala kupovinu karata, koja se više ne bi morala odvijati isključivo na samom kolodvoru. S druge strane, sustavi za online kupnju karata nisu novost i postoje praktički svugdje, no rijetko koji sugerira putniku u koji vagon se ukrcati. Upravo to čini aplikaciju „*PassDirect*“ inovativnom i prilagodljivom zahtjevima tržišta.



## 3. Specifikacija programske potpore

### 3.1 Funkcionalni zahtjevi

#### Dionici:

1. Administrator
2. Korisnici (putnici)
  - a) neregistrirani/neprijavljeni korisnik
  - b) registrirani korisnik
3. Željeznica
4. sustav senzora *Gotcha*
5. Razvojni tim

#### Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani/neprijavljeni korisnik (inicijator) može:
  - (a) pregledati red vožnje vlakova
  - (b) pregledati cijene vožnji
  - (c) filtrirati pretraživanje reda vožnje
  - (d) registrirati se u sustav, stvoriti vlastiti korisnički račun za koju su mu potrebni i korisničko ime, lozinka, ime, prezime, broj mobitela, e-mail adresa
  - (e) prijaviti se u sustav uz uvjet da je prethodno registriran
2. Prijavljen korisnik (inicijator) može:
  - (a) pregledati red vožnje vlakova
  - (b) pregledati cijene vožnji
  - (c) filtrirati pretraživanje reda vožnje
  - (d) pregledati i mijenjati svoje korisničke podatke
  - (e) izbrisati svoj korisnički račun
  - (f) kupiti kartu za odabranu vožnju (u slučaju prve kupovine unosi podatke o kartici)

(g) pregledati povijest svojih vožnji

3. Administrator (inicijator) može:

(a) vidjeti popis svih registriranih korisnika i njihovih osobnih podataka

(b) brisati korisnike

(c) pristupiti statistici (pregled kupovina)

4. Baza podataka (sudionik):

(a) pohranjuje sve podatke o korisnicima i njihovim ovlastima

(b) pohranjuje sve podatke o stanicama, kartama, vlakovima i njihovim kapacitetima

### 3.1.1 Obrasci uporabe

#### UC1 - Klasični pregled reda vožnje

- **Glavni sudionik:** Korisnik
- **Cilj:** pregledati red vožnje vlakova
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
  1. Korisnik nakon učitavanja stranice odabire početno i završno stajalište te datum
  2. Prikazuju se vožnje koje odgovaraju korisnikovim zahtjevima
  3. Prikazuju se i ostale informacije; identifikacijska oznaka vlaka, kolosijek dolaska, kašnjenje vlaka i cijena karte
- **Opis mogućih odstupanja:**
  - 1.a Korisnik je odabrao početno stajalište koje se nalazi iza konačnog, korisnik je unio datum i stajališta za koje ne postoje vožnje
    1. Sustav obavještava korisnika o neuspjelom odabiru stajališta i datuma, te ga vraća na stranicu za odabir
    2. Korisnik ispravlja prethodni unos ili odustaje od pregleda vožnji

#### UC2 - Prilagođeni pregled reda vožnje

- **Glavni sudionik:** Korisnik
- **Cilj:** prilagoditi red vožnje korisnikovim zahtjevima
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik mora biti registriran
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju naprednog pretraživanja
  2. Korisnik odabire jedno od ponuđenih stajališta
  3. Prikazuje se popis vlakova s vremenima dolaska na to stajalište unutar 24h.

#### UC3 - Registracija korisnika

- **Glavni sudionik:** Korisnik
- **Cilj:** Stvoriti korisnički račun za potpuni pristup sustavu
- **Sudionici:** Baza podataka
- **Preduvjet:** -

- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za registraciju
  2. Korisnik unosi potrebne korisničke podatke
  3. Korisnik prima obavijest o uspješnoj registraciji
- **Opis mogućih odstupanja:**
  - 2.a Korisnik je odabrao zauzeto korisničko i/ili e-mail, neki od korisničkih podataka unesen je u nedozvoljenom formatu, unos neispravne e-mail adrese, lozinka se sastoji od praznih znakova
    1. Sustav obavještava korisnika o neuspjeloj registraciji te ga vraća na stranicu za registraciju
    2. Korisnik ispravlja pogrešan unos ili odustaje od registracije

#### UC4 - Prijava u sustav

- **Glavni sudionik:** Korisnik
- **Cilj:** prijava u sustav i potpuni pristup sustavu
- **Sudionici:** Baza podataka
- **Preduvjet:** Registracija
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za prijavu
  2. Korisnik unosi korisničko ime i lozinku
  3. Potvrda ispravnosti korisničkih vjerodajnica
  4. Pristup korisničkim funkcijama
- **Opis mogućih odstupanja:**
  - 3.a Korisnik je unio neispravno korisničko ime/lozinku
    1. Sustav obavještava korisnika o neuspjeloj prijavi te ga vraća na stranicu za prijavu
    2. Korisnik ispravlja pogrešan unos ili odustaje od prijave

#### UC5 - Izmjena korisničkih podataka

- **Glavni sudionik:** Korisnik
- **Cilj:** Izmjena korisničkih podataka u sustavu
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijava
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za izmjenu korisničkih podataka
  2. Korisnik unosi novo korisničko ime i lozinku

3. Potvrda ispravnosti korisničkih vjerodajnica
4. Pohrana promjena
- **Opis mogućih odstupanja:**
  - 3.a Korisnik je odabrao zauzeto korisničko ime, neki od korisničkih podataka unesen je u nedozvoljenom formatu
    1. Sustav obavještava korisnika o neuspjeloj promjeni korisničkih podataka te ga vraća na stranicu za promjenu korisničkih podataka
    2. Korisnik ispravlja pogrešan unos ili odustaje od promjene korisničkih podataka

#### UC6 - Brisanje korisničkog računa

- **Glavni sudionik:** Korisnik
- **Cilj:** Trajno brisanje korisničkog računa iz sustava
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijava
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za brisanje korisničkog računa
  2. Korisnik dobiva upozorenje o trajnom brisanju računa te je od njega tražena potvrda za tu akciju
  3. Korisnik potvrđuje brisanje računa
  4. Korisnički račun je izbrisan te je neprijavljeni korisnik vraćen na početnu stranicu
- **Opis mogućih odstupanja:**
  - 2.a Korisnik je odustao od brisanja svojeg računa
    1. Sustav vraća korisnika na početnu stranicu

#### UC7 - Kupovina karte

- **Glavni sudionik:** Korisnik
- **Cilj:** Kupovina karte za željenu vožnju
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijava
- **Opis osnovnog tijeka:**
  1. Korisnik odabire vožnju pomoću pregleda reda vožnje
  2. Korisnik preko jednog od prikaza odabire željenu vožnju
  3. Korisnik se prikazuje cijena te on odabire opciju za kupovinu karte

4. Korisnik unosi potrebne podatke za plaćanje (samo prilikom orve kupovine, u ostalim kupovinama podaci se automatski ispunjavaju)
  5. Potvrda ispravnosti podataka za plaćanje
  6. Sustav obavještava korisnika o uspješnoj kupovini karte preko e-maila.
- **Opis mogućih odstupanja:**
    - 5.a Korisnik je unio neispravne podatke za plaćanje
      1. Sustav obavještava korisnika o neuspjeloj kupovini te ga vraća na stranicu za plaćanje
      2. Korisnik ponovno unosi podatke za plaćanje ili odustaje od kupovine

#### UC8 - Pregled korisnikovih prošlih kupovina

- **Glavni sudionik:** Korisnik
- **Cilj:** Pregled povijesti kupovina karata
- **Sudionici:** Baza podataka
- **Preduvjet:** Prijava
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju pregleda prošlih kupovina
  2. Korisniku se prikažu informacije o svim prošlim kupovinama

#### UC9 - Pregled korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Pregledati registrirane korisnike
- **Sudionici:** Baza podatka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
  1. Administrator odabire opciju pregledavanja korisnika
  2. Prikaže se lista svih ispravno registriranih korisnika s osobnim podacima

#### UC10 - Brisanje korisnika

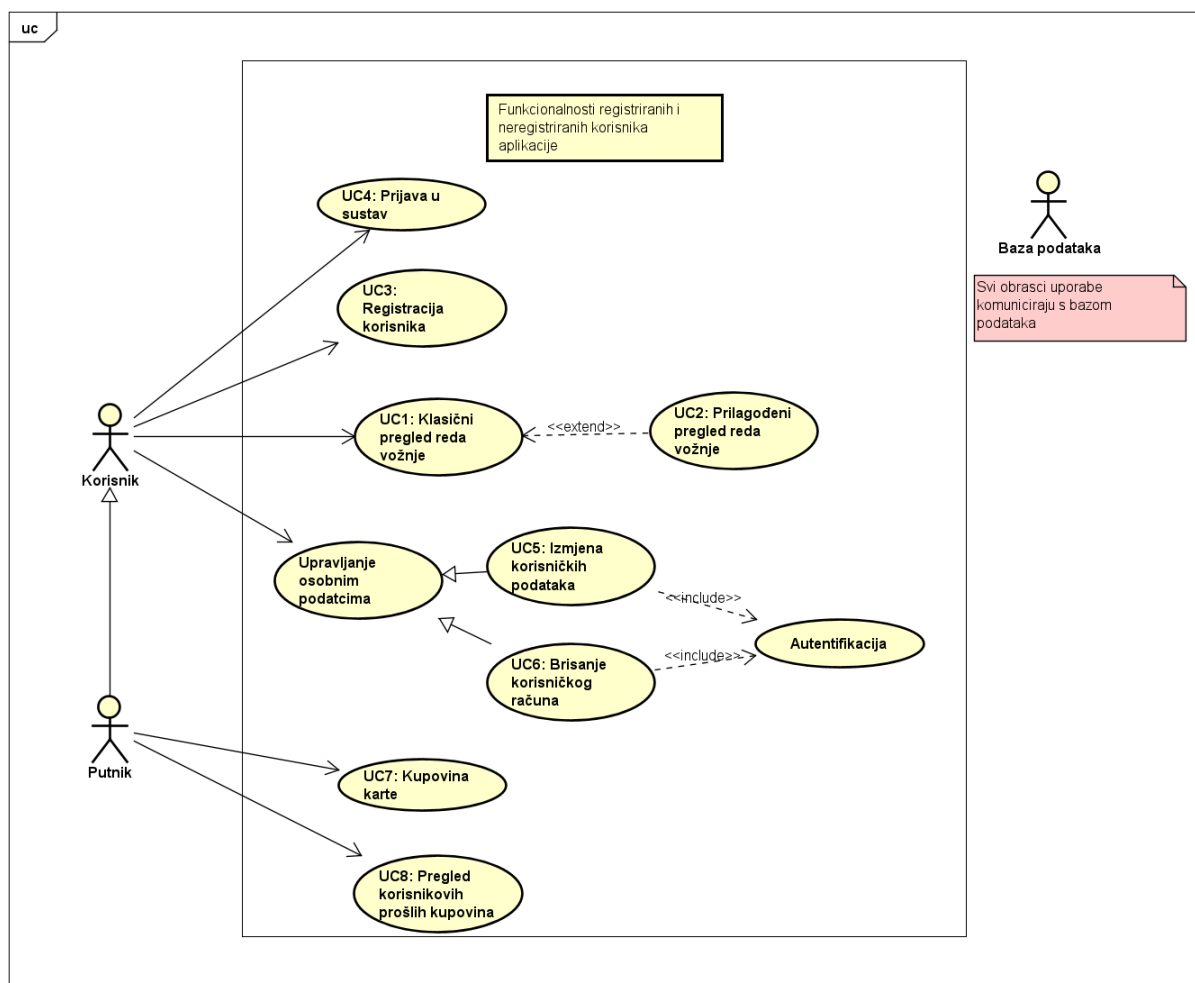
- **Glavni sudionik:** Administrator
- **Cilj:** Obrisati korisnika
- **Sudionici:** Baza podatka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**

1. Administrator pronalazi željenog korisnika
2. Administrator odabire opciju uklanjanja korisnika
3. Administrator uklanja željenog korisnika i njegove podatke iz baze podataka

#### **UC11 - Pregled svih prošlih kupovina**

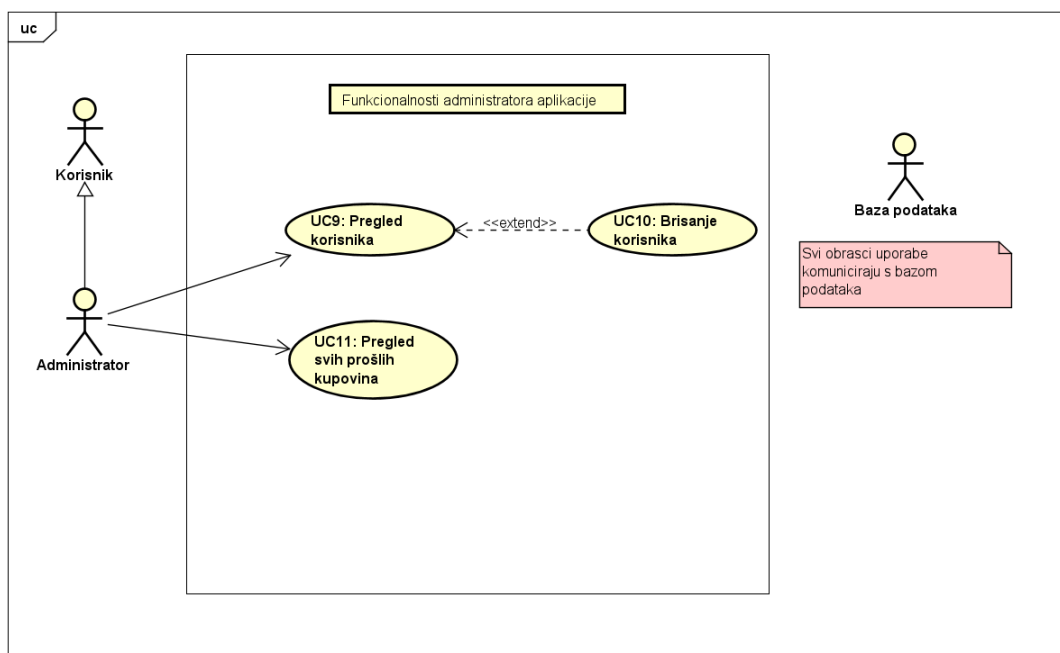
- **Glavni sudionik:** Administrator
- **Cilj:** Pregledati sve korisničke kupovine
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
  1. Administrator odabire opciju pregleda prošlih kupovina
  2. Administrator vidi informacije o prošlim kupovinama (tko je bio kupac, kojim vlakom i kada je putovao)

## Dijagrami obrazaca uporabe



Slika 3.1: Dijagram obrasca uporabe, funkcionalnost korisnika i klijenta



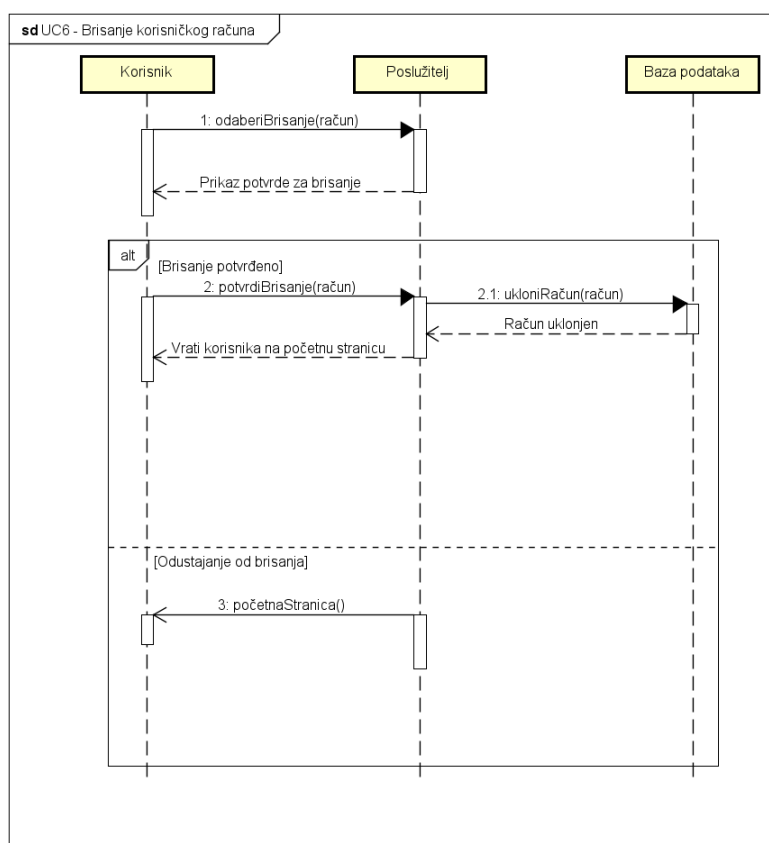


Slika 3.2: Dijagram obrasca uporabe, funkcionalnost administratora

### 3.1.2 Sekvencijski dijagrami

#### Obrazac uporabe UC6 – Brisanje korisničkog računa

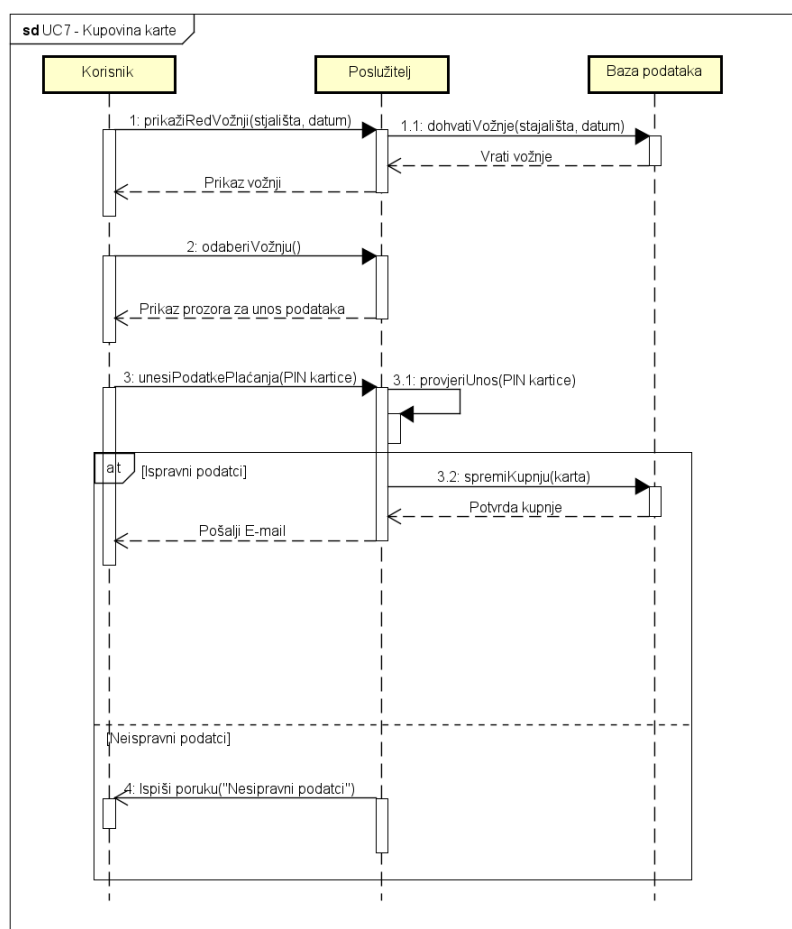
Korisnik odabire opciju brisanja korisničkog računa. Poslužitelj vraća upozorenje o trajnom brisanju računa i traži potvrdu od korisnika za akciju brisanja računa. Ukoliko korisnik potvrdi brisanje računa, poslužitelj javlja bazi podataka da obriše korisnikov račun i vraća korisnika na početnu stranicu. U slučaju odustajanja od brisanja računa, sustav vraća korisnika na početnu stranicu.



Slika 3.3: Sekvencijski dijagram za UC6

### Obrazac uporabe UC7 – Kupovina karte

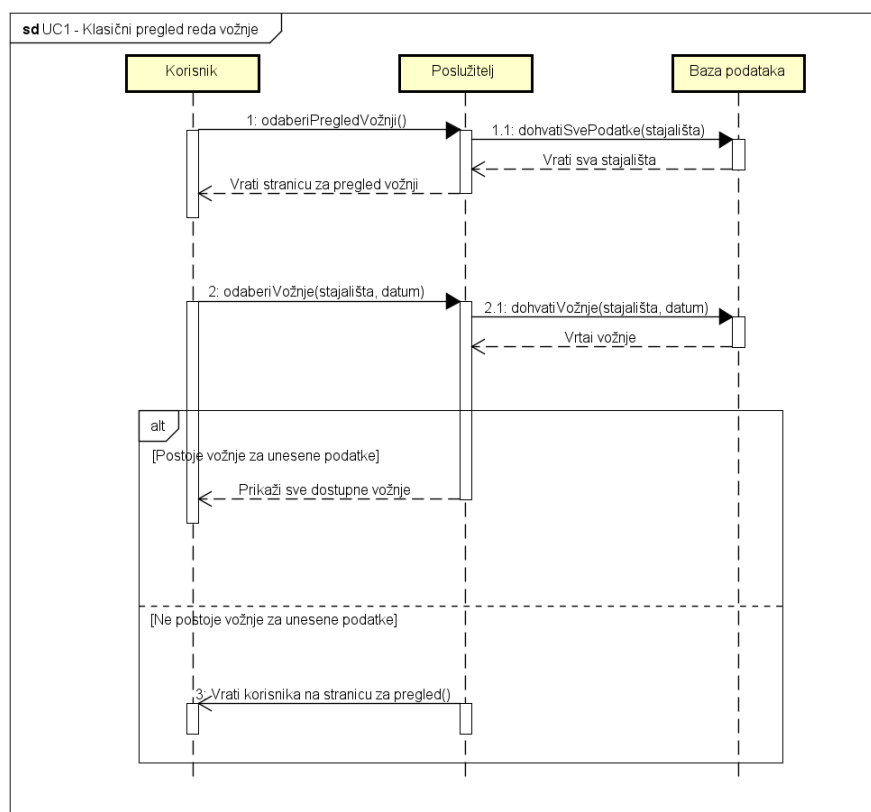
Korisnik šalje zahtjev za prikaz reda vožnji kako bi mogao odabrati vožnju. Poslužitelj dohvaća sve dostupne vožnje i prikazuje ih. Korisnik odabire željenu vožnju, te je cijena karte već prikazana korisniku. Korisnik odabire opciju za kupnju karte, te poslužitelj vraća prikaz za unos podataka za plaćanje. Zatim korisnik unosi potrebne podatke za plaćanje te poslužitelj provjerava ispravnost tih podataka. Ako su podaci za plaćanje ispravni, poslužitelj obavještava korisnika o kupnji karte putem e-maila. U slučaju unosa neispravnih podataka za plaćanje, poslužitelj vraća korisnika na stranicu za plaćanje.



Slika 3.4: Sekvencijski dijagram za UC7

**Obrazac uporabe UC1 – Klasični pregled reda vožnje**

Korisnik odabire početno i završno stajalište, te datum za koji želi pregledati vožnje. ako su uneseni podatci ispravni poslužitelj vraća sve vožnje koji odgovaraju zahtjevima. U slučaju unosa neispravnih podataka stanica i datuma, korisnik je vraćen na stranicu za unos podataka.



Slika 3.5: Sekvencijski dijagram za UC1

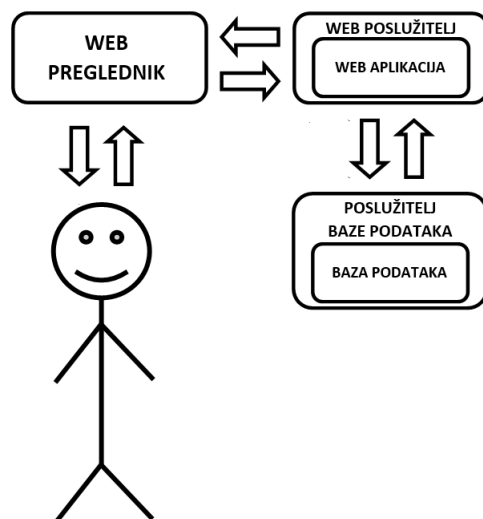
## 3.2 Ostali zahtjevi

- Sustav treba omogućiti rad više korisnika u stvarnom vremenu
- Korisničko sučelje i sustav moraju podržavati hrvatsku abecedu i dijakritičke znakove pri unosu i prikazu svog tekstualnog sadržaja
- Sustav treba biti implementiran kao web aplikacija uz korištenje objektno orijentiranih programskih jezika
- Pristup sustavu treba biti omogućen pomoću HTTPS protokola iz javne mreže
- Veza s bazom podataka mora biti kvalitetno ostvarena, brza i otporna na vanjske utjecaje
- Izvršavanje dijela programa u sklopu kojega se odvija spajanje na bazu podataka ne smije trajati dulje od nekoliko sekundi
- Sustav kao valutu u svim transakcijama i prikazima cijena koristi hrvatsku kunu (HRK)
- Sustav u tablicama voznog reda vrijeme treba računati po vremenskoj zoni GMT+1
- Korisničko sučelje treba biti nenametljivo, a sustav jednostavan za korištenje čak i bez opširnih uputa
- Sustav treba postaviti sigurnosno ograničenje na lozinku pri registraciji; lozinka ne smije biti kraća od šest znakova, ne smije biti slična unesenom e-mailu niti se sastojati samo od znamenki
- Sustav treba biti implementiran kao web aplikacija kompatibilna s nekoliko različitih najčešće korištenih web preglednika (*Mozilla Firefox, Google Chrome, Safari, Opera, Microsoft Edge*)

## 4. Arhitektura i dizajn sustava

Arhitekturno, sustav se može podijeliti na tri podsustava:

- Web poslužitelj
- Web aplikacija
- Baza podataka



Slika 4.1: Skica arhitekture sustava

*Web preglednik* je program koji korisniku omogućuje pregled web-stranica i multimedijalnih sadržaja vezanih uz njih. Svaki internetski preglednik ujedno je i prevoditelj, što znači da je stranica pisana u kodu koji preglednik nakon toga interpretira kao nešto svakome razumljivo. Korisnik putem web preglednika šalje zahtjev web poslužitelju.

*Web poslužitelj* osnova je rada web aplikacije. Njegova primarna zadaća je komunikacija klijenta s aplikacijom. Komunikacija se odvija preko HTTP (engl. *Hyper Text Transfer Protocol*) protokola, što je protokol u prijenosu informacija na webu. Poslužitelj je onaj koji pokreće web aplikaciju te joj prosljeđuje zahtjev.

Korisnik koristi *web aplikaciju* za obrađivanje željenih zahtjeva. Web aplikacija obrađuje zahtjev te ovisno o tom zahtjevu pristupa bazi podataka nakon čega preko

poslužitelja vraća korisniku odgovor u obliku HTML dokumenta vidljivog u web pregledniku.

Arhitektura sustava temeljena je na MVC (Model-View-Controller) konceptu. Karakteristika MVC koncepta je nezavisan razvoj pojedinih dijelova aplikacije, što za posljedicu ima jednostavnije ispitivanje, kao i jednostavno razvijanje i dodavanje novih svojstava u sustav.

Sastavnice MVC koncepta su:

- **Model** - Središnja komponenta sustava. Predstavlja dinamičke strukture podataka, neovisne o korisničkom sučelju. Izravno upravlja podacima, logikom i pravilima aplikacije. Također prima ulazne podatke od Controllera;
- **View** - Bilo kakav prikaz podataka, poput grafa. Mogući su različiti načini prikaza iste informacije, poput grafičkog ili tabličnog prikaza podataka;
- **Controller** - Prima ulaze i prilagođava ih za prosljeđivanje Modelu ili Viewu. Upravlja korisničkim zahtjevima i temeljem njih izvodi daljnju interakciju s ostalim elementima sustava.

U izradi aplikacije korišten je programski jezik JavaScript u frontend dijelu te Python u kombinaciji s Django radnim okvirom za backend.

## 4.1 Baza podataka

Za potrebe sustava korišten je relacijski model podataka. Objekti u sklopu tog modela nazivaju se relacijama, a to su zapravo tablice definirane svojim imenom i skupom atributa. Takva struktura baze omogućava visoki stupanj nezavisnosti podataka, što je jedna od glavnih prednosti relacijskog modela.

Baza podataka aplikacije „*PassDirect*“ sadrži sljedeće entitete:

- Korisnik
- Vlak
- Karta
- Stanica

i relaciju/vezu:

- VlakStanica.

### 4.1.1 Opis tablica

**Korisnik** Entitet sadrži sve informacije o korisniku aplikacije: identifikacijsku oznaku korisnika, e-mail, ime, prezime, lozinku, broj kartice, ulogu, oznaku aktivnosti, vrijeme zadnje prijave kao i datum kada se korisnik registrirao u sustav. Ovaj entitet u vezi je *One-To-Many* s entitetom Karta preko atributa identifikacijska oznaka korisnika.

| Korisnik          |           |                                            |
|-------------------|-----------|--------------------------------------------|
| IDKorisnik        | INT       | jedinstveni identifikator korisnika        |
| email             | VARCHAR   | e-mail adresa korisnika                    |
| ime               | VARCHAR   | ime korisnika                              |
| prezime           | VARCHAR   | prezime korisnika                          |
| password          | VARCHAR   | lozinka za korisnički račun                |
| brKartice         | VARCHAR   | broj kartice za kupovinu                   |
| isSuperuser       | BOOLEAN   | uloga korisnika u sustavu                  |
| lastLogin         | TIMESTAMP | vrijeme zadnje prijave                     |
| isActive          | BOOLEAN   | prikazuje trenutnu (ne)aktivnost korisnika |
| datumRegistracije | DATE      | datum u kojem se korisnik registrirao      |

**Karta** Ovaj entitet sadrži sve informacije o karti koju korisnik kupuje. Sadrži attribute: identifikacijsku oznaku karte, identifikacijsku oznaku korisnika, identifikacijsku oznaku vlaka, cijenu, vremena polaska i dolaska te polazište i odredište. Ovaj entitet je u dvije veze *One-To-One* s entitetom Korisnik preko atributa identifikacijska oznaka korisnika te s entitetom Vlak preko identifikacijske oznake vlaka.

| Karta      |     |                                     |
|------------|-----|-------------------------------------|
| IDKarta    | INT | jedinstveni identifikator karte     |
| IDKorisnik | INT | jedinstveni identifikator korisnika |
| IDVlak     | INT | jedinstveni identifikator vlaka     |

Nastavljeno na idućoj stranici



Nastavljeno od prethodne stranice

| Karta          |           |                                        |
|----------------|-----------|----------------------------------------|
| cijena         | DOUBLE    | cijena karte u kunama s dvije decimale |
| vrijemePolaska | TIMESTAMP | vrijeme polaska vlaka                  |
| vrijemeDolaska | TIMESTAMP | vrijeme dolaska vlaka                  |
| polaziste      | VARCHAR   | grad iz kojeg vlak polazi              |
| odrediste      | VARCHAR   | grad u koji vlak dolazi                |

**Vlak** Ovaj entitet sadrži sve bitne informacije o vlaku koji putuje željeznicom. Sadrži attribute: identifikacijsku oznaku vlaka, broj vagona i broj putnika koji mogu putovati u vagonu. Ovaj entitet je u vezi *One-To-Many* s entitetom Karta preko atributa identifikacijska oznaka vlaka te u relaciji *VlakStanica* s entitetom *stanica*, a povezani su preko identifikacijske oznake vlaka i identifikacijske oznake stanice.

| Vlak           |     |                                 |
|----------------|-----|---------------------------------|
| IDVlak         | INT | jedinstveni identifikator vlaka |
| brVagona       | INT | broj vagona određenog vlaka     |
| brPutnikaVagon | INT | broj putnika po vagonu          |

**Stanica** Entitet stanica sadrži informacije o stanicama kroz koje vlak na svojoj ruti prolazi. Sadrži dva atributa: IDStanica i naziv stanice. Ovaj entitet je u relaciji *VlakStanica* s entitetom *vlak*, a entiteti su povezani preko identifikacijske oznake vlaka i identifikacijske oznake stanice.

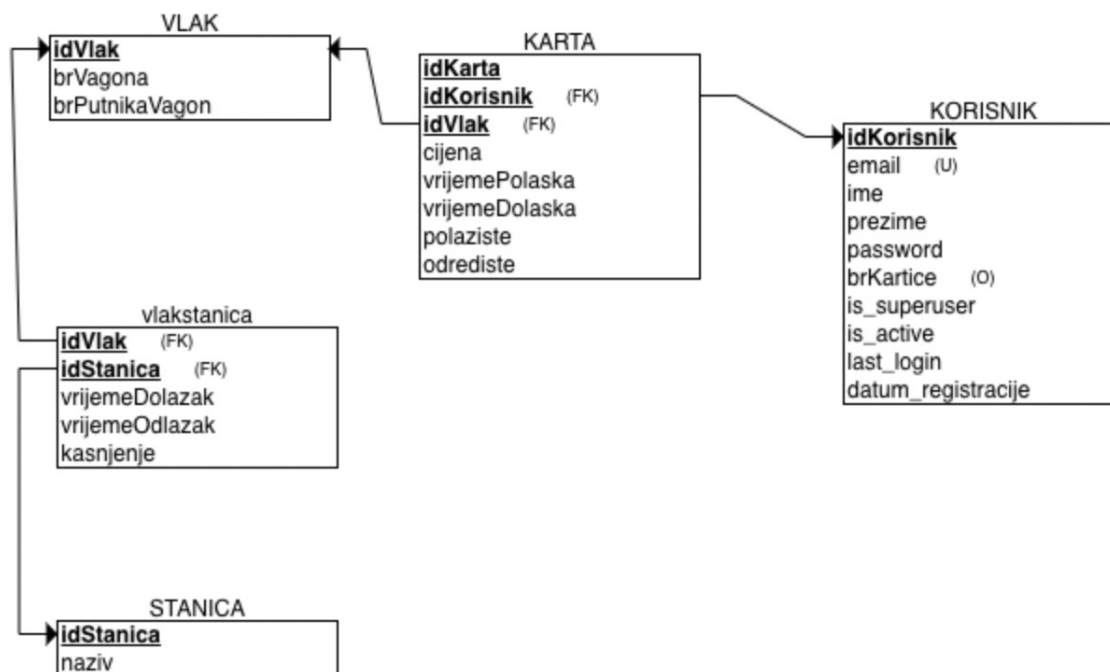
| Stanica      |         |                                   |
|--------------|---------|-----------------------------------|
| IDStanica    | INT     | jedinstveni identifikator stanice |
| nazivStanica | VARCHAR | naziv stanice na ruti vožnje      |

**VlakStanica** Relacija *VlakStanica* sadrži informacije koje su bitne za putnike koji se na određenoj stanici ukrcavaju na vlak. Sadrži attribute: identifikacijsku oznaku vlaka, identifikacijsku oznaku stanice, vrijeme dolaska i odlaska vlaka kao

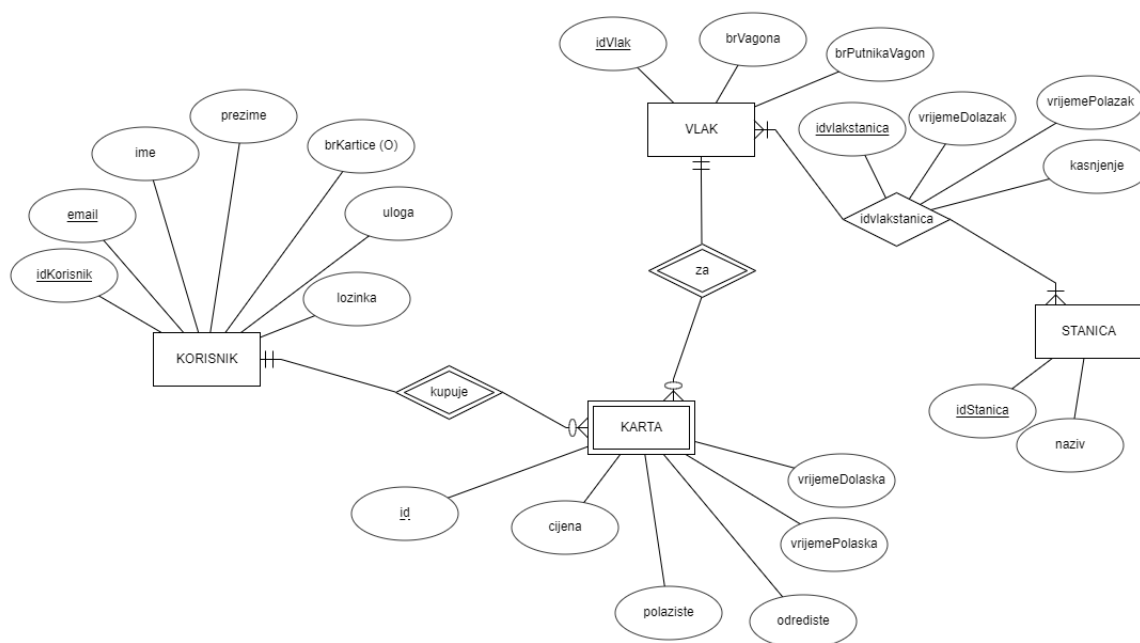
i kašnjenje vlaka.

| VlakStanica    |           |                                   |
|----------------|-----------|-----------------------------------|
| IDVlak         | INT       | jedinstveni identifikator vlaka   |
| IDStanica      | INT       | jedinstveni identifikator stanice |
| vrijemeDolazak | TIMESTAMP | vrijeme dolaska vlaka na stanicu  |
| vrijemeOdlazak | TIMESTAMP | vrijeme odlaska vlaka sa stanice  |
| kasnjenje      | TIME      | vrijeme kašnjenja                 |

#### 4.1.2 Dijagram baze podataka

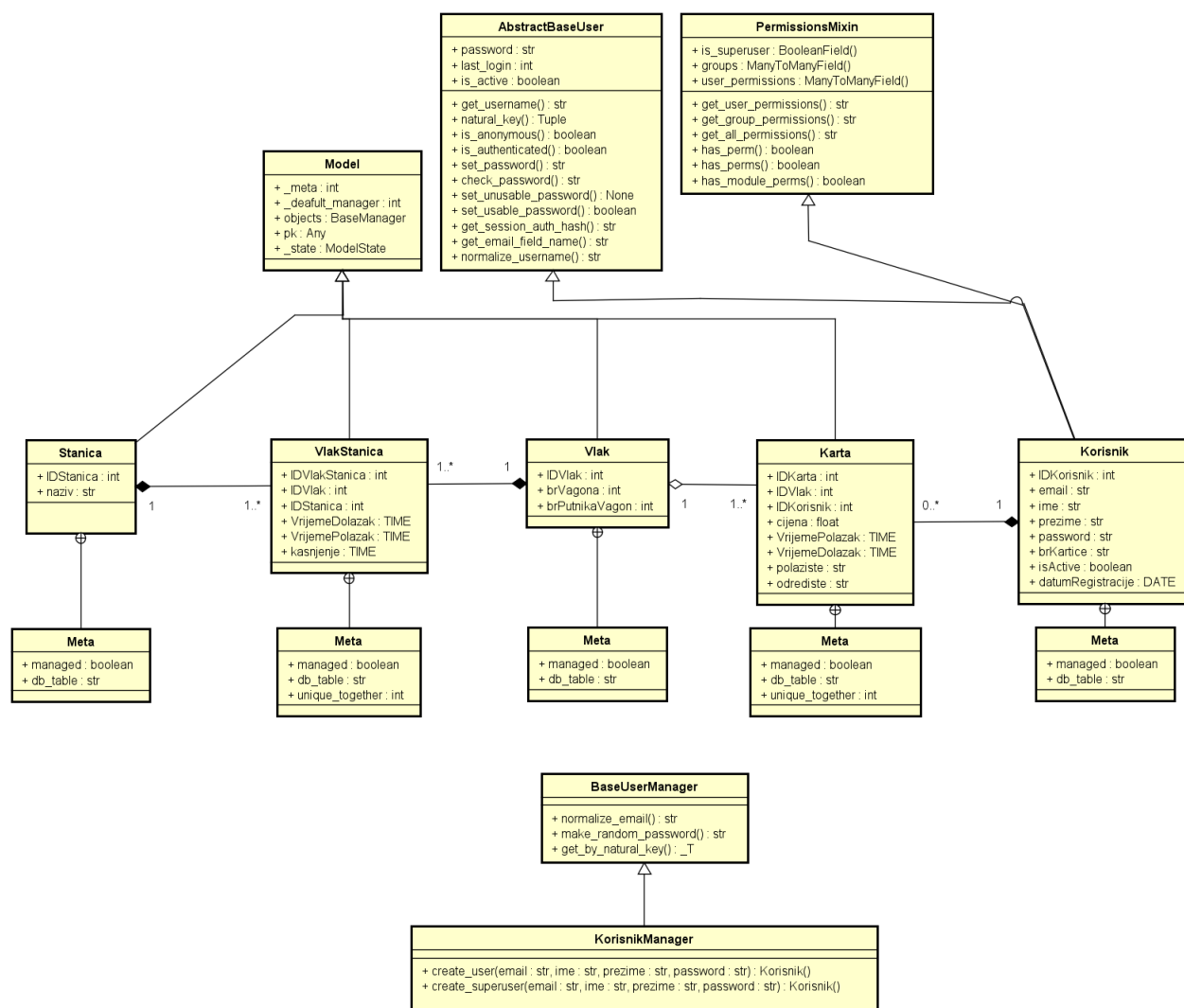


Slika 4.2: Dijagram baze podataka

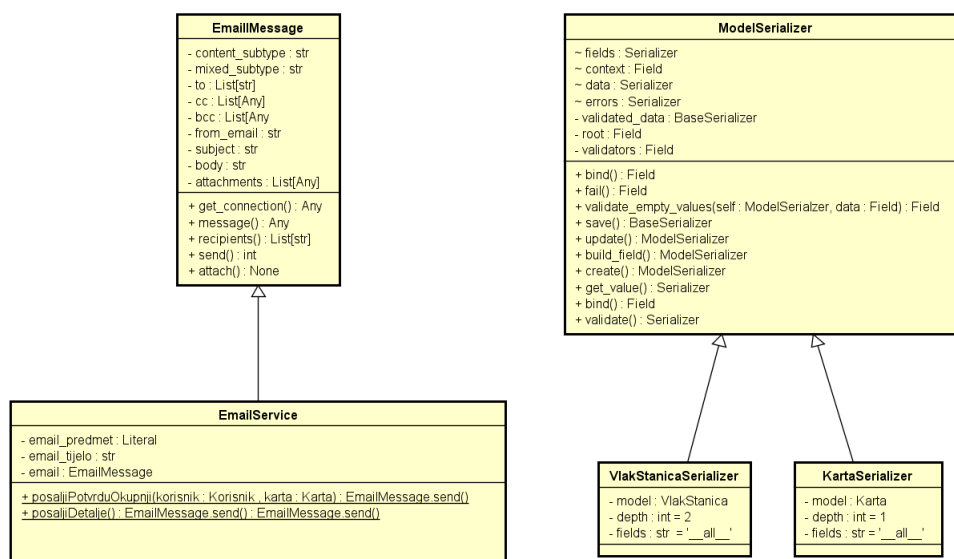


Slika 4.3: ER Dijagram baze podataka

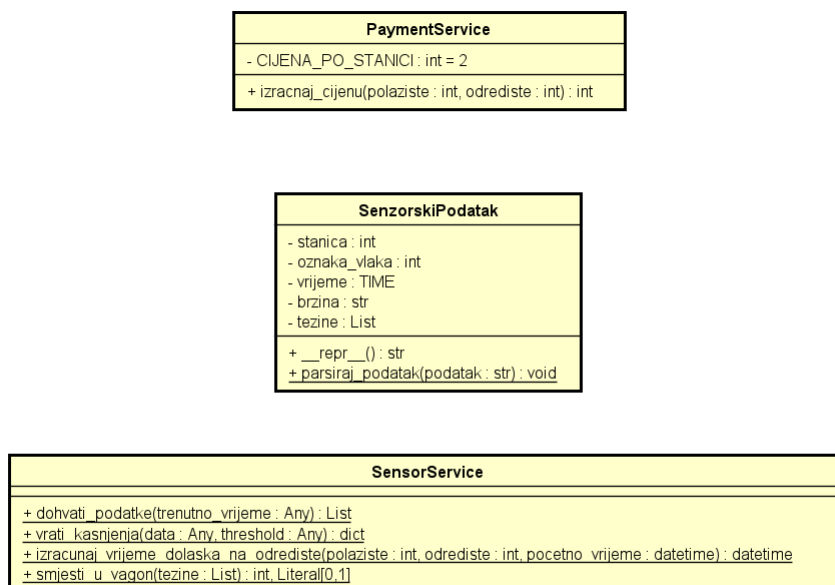
## 4.2 Dijagram razreda



Slika 4.4: Dijagram razreda 1



Slika 4.5: Dijagram razreda 2



Slika 4.6: Dijagram razreda 3

**Dijagram razreda** sa slike 4.3. prikazuje šest razreda koji su definirani samim zadatkom i četiri razreda koji su preuzeti iz *Django* framework-a. *Django* klase su *BaseUserManager*, *Model*, *AbstractUser* te *PermissionsMixin* i one omogućuju lakše upravljanje drugim klasama koje su definirane problematikom zadatka, a to

su Stanica, VlakStanica, Vlak, Karta, Korisnik i KorisnikManager.

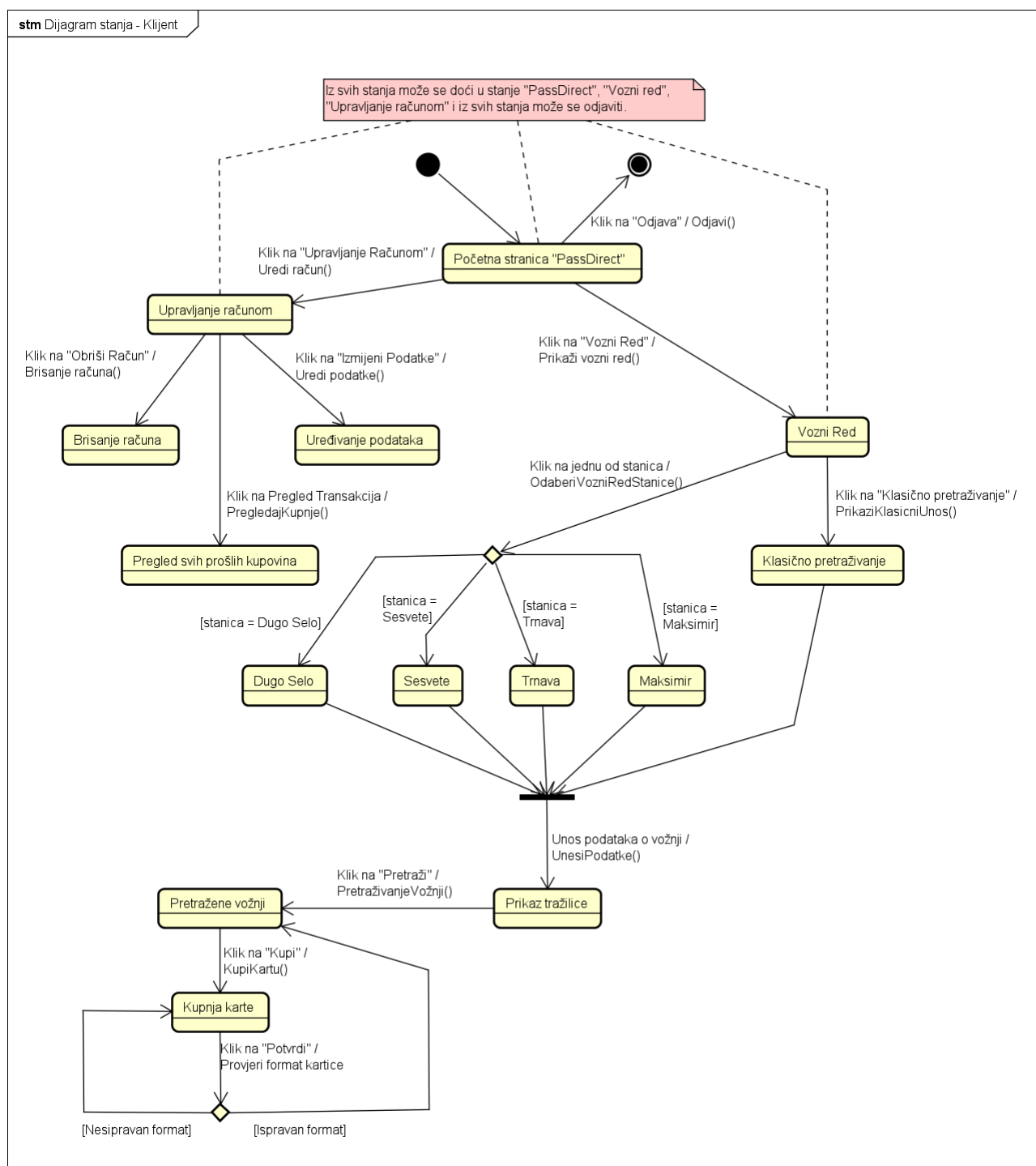
*Klase koje nisu preuzete iz Django-a sadrže skoro pa potpuno iste atribute kao i pripadni entiteti u Bazi podataka pa ih se ovdje neće ponovno navoditi.*

Razredi Stanica, VlakStanica, Vlak i Karta nasljeđuju klasu Model i svaki od njih ima svoju pripadajuću ugniježdenu klasu Meta.

Razred Korisnik nasljeđuje dvije klase: AbstractBaseUser i PermissionsMixin te ima svoju ugniježdenu klasu Meta. Klase koje je razred naslijedio sadrže metode poput *setUsablePassword* i *checkPassword* koje se brinu o korisničkoj lozinci kao i metode koje određuju razinu pristupa koju će korisnik imati; *getAllPermissions*, *getUserPermissions* itd.

Razred KorisnikManager ima dvije funkcije koje određuju hoće li korisnik biti *superuser* ili *user*, a nasljeđuje klasu BaseUserManager. Dijagram razreda sa **Slike 4.4.** prikazuje klase EmailMessage i EmailService potrebne za slanje mailova korisniku; pri registraciji, zaboravljenoj lozinci te pri slanju mailova za potvrdu kupovine. Razred ModelSerializer služi za serijalizaciju modela u json oblik podatak, a naslijedili su ga razredi VlakStanicaSerializer i KartaSerializer. **Slika 4.5.** opisuje treći dio Dijagrama razreda i sadrži tri razreda. Prvi je *PaymentService* koji sadrži metodu koja izračunava cijenu karte. Drugi razred, *SenzorskiPodatak*, modelira jedno očitavanje sa senzora sustava *Gotcha!*. I posljednji razred, *SensorService* nudi metode: *dohvati\_podatke*, *vrati\_kasnjenja*, *izracunaj\_vrijeme\_dolaska\_na\_odrediste* i *smjesti\_u\_vagon*. Metoda *dohvati\_podatke* vraća očitane podatke do predanog vremena, a *vrati\_kasnjenja* popunjava predani raspored s kašnjenjima i umanjuje ih za threshold. *Smjesti\_u\_vagon* vraća peron za predanu listu uređenih parova (tuple-a) očitanih težina vagona.

## 4.3 Dijagram stanja

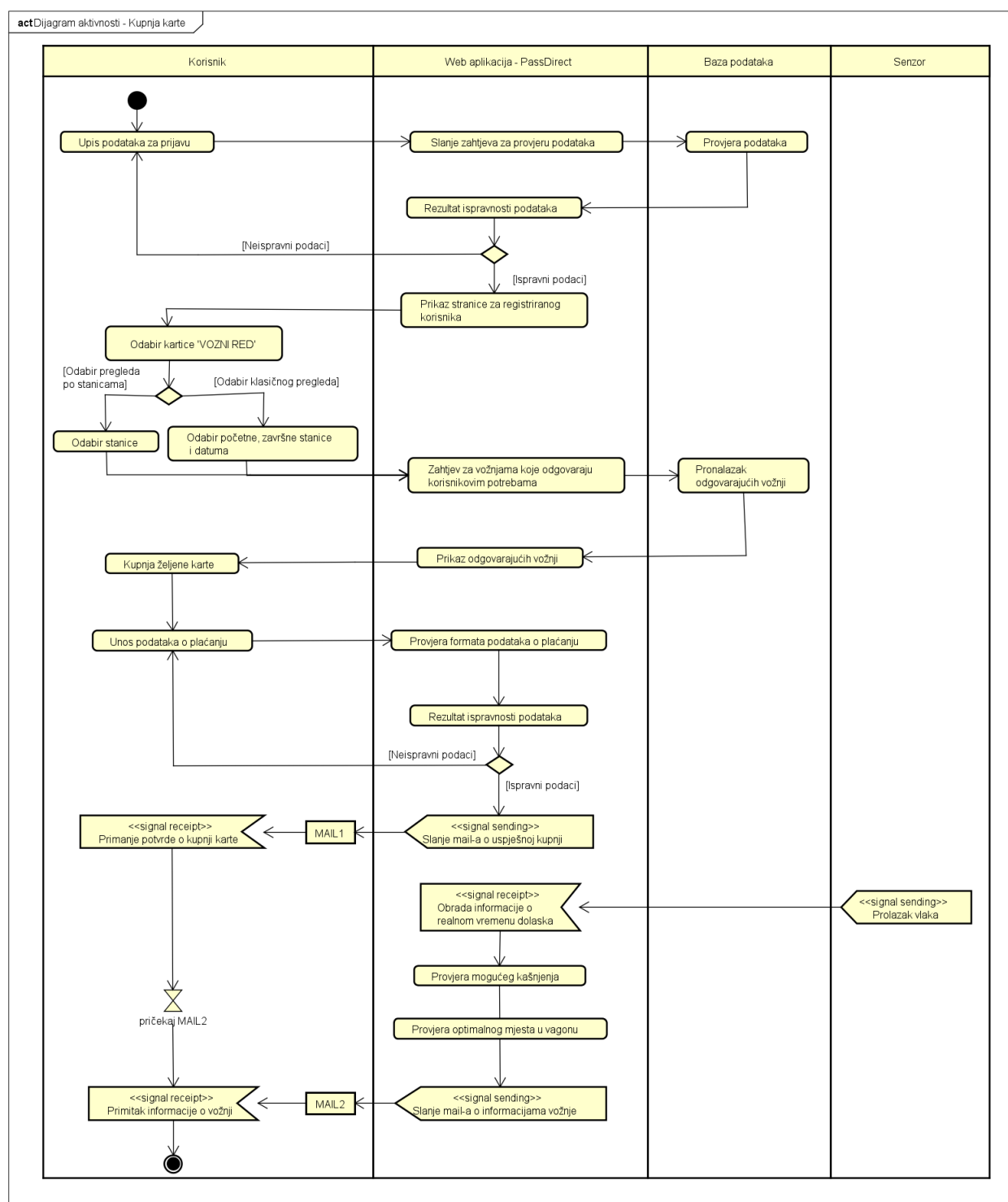


Slika 4.7: Dijagram stanja

**Dijagram stanja** prikazuje stanja objekta te prijelaze iz jednog stanja u drugo temeljene na događajima. Slika 4.6. prikazuje jedan takav dijagram stranice *PassDirect* za registriranog korisnika kojemu je cilj kupiti kartu. Nakon prijave učitava se početna stranica gdje korisnik ima izbor upravljati svojim računom ili vidjeti vozni red. Ukoliko se odluči za upravljanje računom imati će mogućnost urediti svoje podatke, izbrisati svoj račun ili pregledati sve svoje prošle kupovine. Ako se pak korisnik odluči za prikaz Voznog reda, tada će imati dvije mogućnosti: klasično pretražiti vožnje ili ih pretražiti preko stanica (Sesvete, Trnava, Dugo Selo, Maksimir). Bez obzira na vrstu prikaza - korisnik može kupiti kartu nakon što pronađe odgovarajuću vožnju. Ako prilikom kupovine unese neodgovarajući format kartičnog plaćanja korisnik će dobiti prikladno upozorenje, a ukoliko kupovina prođe uspješno korisnika će dočekati odgovarajuća obavijest na stranici i mail, te će biti vraćen na Prikaz vožnji. Tijekom cijelog postupka korisnik ima mogućnost Odjave.



## 4.4 Dijagram aktivnosti



Slika 4.8: Dijagram aktivnosti

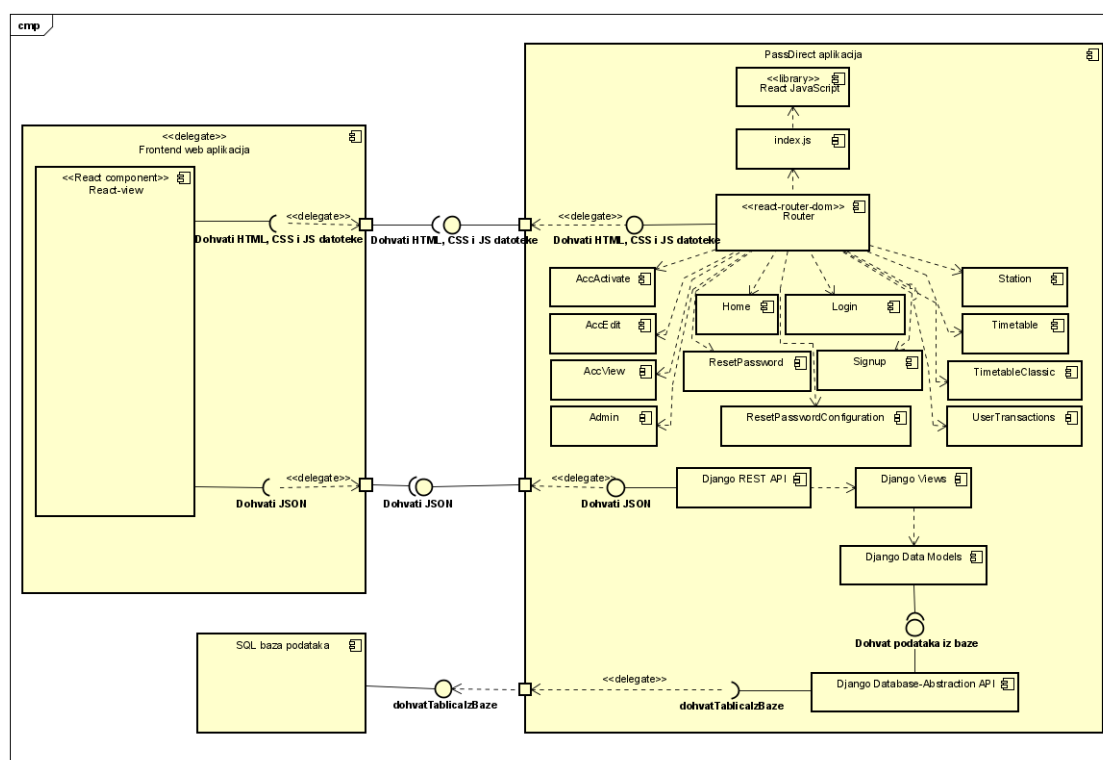
**Dijagram aktivnosti** koristi se za modeliranje ponašanja nizom akcija. U modeliranju svaki sljedeći korak poduzima se nakon završenog prethodnog. Na dijagramu aktivnosti sa **Slike 4.7.** prikazan je proces Kupovine karte za registriranog korisnika. Nakon unosa točnih podataka za prijavu (točnost se provjerava u komunikaciji s bazom podataka) korisnik odabire jednu od opcija prikaza Voznog reda. Nakon što odabere kartu koja odgovara njegovim zahtjevima (takve se pronalaze u bazi podataka) ima ju mogućnost kupiti, a za kupovinu je potrebna provjera ispravnosti formata kartičnog plaćanja koju obavlja web-aplikacija. Ukoliko su podaci bili točni korisnik će dobiti prvi mail sa potvrdom uspješne kupovine. Nakon što vlak prođe kroz senzor sustava *Gotcha!* on će poslati podatke o vremenu prolaska vlaka i opterećenosti vagona web-aplikaciji koja će tada obavijestiti putnika/korisnika o mogućem kašnjenju i najoptimalnijem putničkom mjestu u vagonu.

## 4.5 Dijagram komponenti

Dijagram komponenti opisuje organizaciju i međuovisnost komponenti, interne strukture te odnose prema okolini. Sustavu se pristupa preko dva različita sučelja. Preko sučelja za dohvat HTML, CSS i JS datoteka poslužuju se datoteke koje pripadaju *frontend* dijelu aplikacije. Router je komponenta koja na upit s url određuje koja datoteka će se poslužiti na sučelje.

*Frontend* dio sastoji se od niza JavaScript datoteka koje ovise o React biblioteci iz koje dohvaćaju gotove komponente kao što su gumbi, forme i slično. Preko sučelja za dohvat JSON podataka pristupa se REST API komponenti. REST API poslužuje podatke koji pripadaju *backend* dijelu aplikacije.

*Django Database Abstraction API* zadužen je za dohvaćanje tablica iz baze podataka pomoću SQL upita. Podaci koji su pristigli iz baze šalju se dalje MVC arhitekturi u obliku DTO (*Data transfer object*). React-view komponenta preko dostupnih sučelja komunicira s PassDirect aplikacijom te ovisno o korisnikovim akcijama osvježava prikaz i dohvaća nove podatke ili datoteke.



Slika 4.9: Dijagram komponenti

## 5. Implementacija i korisničko sučelje

### 5.1 Korištene tehnologije i alati

Komunikacija u timu pretežito je ostvarena preko aplikacije WhatsApp<sup>1</sup>, ali i Discord<sup>2</sup> servera. Za izradu UML dijagrama korišten je alat Astah Professional<sup>3</sup>, a kao sustav za upravljanje izvornim kodom Git<sup>4</sup>. Udaljeni repozitorij projekta dostupan je na platformi GitLab<sup>5</sup>.

U izradi aplikacije korišten je programski jezik JavaScript<sup>6</sup> u frontend dijelu te Python<sup>7</sup> u kombinaciji s Django<sup>8</sup> radnim okvirom za backend.

Frontend dio aplikacije oslanja se na korištenje *open-source* biblioteke React<sup>9</sup> kreirane za programski jezik JavaScript, koja omogućava sastavljanje korisničkog sučelja pomoću elemenata koji se još nazivaju komponentama. Posebnost Reacta je što je po prvi put omogućio takvu radnju koristeći samo osnovne funkcionalnosti JavaScripta (tzv. *vanilla JavaScript*). Odabrano razvojno okruženje za frontend je Microsoft Visual Studio.

Backend dio aplikacije ostvaren je pomoću *open-source* web radnog okvira Django napisanog u programskom jeziku Python. Prednost Djanga je arhitektura temeljena na MTV (engl. *Model-Template-View*) predlošku koja odvaja model, logičku strukturu i izgled web aplikacije te na taj način povećava sigurnost. Odabrano razvojno okruženje za backend je PyCharm<sup>10</sup>.

Komunikacija frontend i backend dijela aplikacije pretežno se oslanja na HTTP(S) protokol uz pomoć JavaScript biblioteke Axios<sup>11</sup>.

Za testiranje sustava i pokretanje ispitnih slučajeva korišten je Selenium<sup>12</sup>.

---

<sup>1</sup><https://www.whatsapp.com/>

<sup>2</sup><https://discord.com/>

<sup>3</sup><https://astah.net/products/astah-professional/>

<sup>4</sup><https://git-scm.com/>

<sup>5</sup><https://gitlab.com/>

<sup>6</sup><https://www.javascript.com/>

<sup>7</sup><https://www.python.org/>

<sup>8</sup><https://www.djangoproject.com/>

<sup>9</sup><https://reactjs.org/>

<sup>10</sup><https://www.jetbrains.com/pycharm/>

<sup>11</sup><https://www.axios.com/>

<sup>12</sup><https://www.selenium.dev/>

## 5.2 Ispitivanje programskog rješenja

### 5.2.1 Ispitivanje komponenti

#### Ispitni slučaj 1: Testiranje funkcije `get_all`

**Ulaz:** `HttpRequest{method="GET"}`

**Očekivani rezultat:** Funkcija vraća listu svih korisnika jer prijavljen korisnik ima administratorska prava (`user.is_superuser = True`).

**Rezultat:** Očekivani rezultat je zadovoljen. *Metoda je prošla test.*

#### Ispitni slučaj 2: Testiranje funkcije `get_all`

**Ulaz:** `HttpRequest{method="GET"}`

**Očekivani rezultat:** Funkcija vraća HTTP statusni kod 401 ("User is not superuser") jer prijavljen korisnik nema administratorska prava (`user.is_superuser = False`).

**Rezultat:** Očekivani rezultat je zadovoljen. *Metoda je prošla test.*

#### Ispitni slučaj 3: Testiranje funkcije `delete`

**Ulaz:** `HttpRequest{method="DELETE"}, 68`

**Očekivani rezultat:** Funkcija briše korisnika koji ima ulazni id i vraća HTTP statusni kod 204 ("User successfully deleted") jer prijavljen korisnik ima administratorska prava (`user.is_superuser = True`) i korisnik s `{idkorisnik = 68}` postoji.

**Rezultat:** Očekivani rezultat je zadovoljen. *Metoda je prošla test.*

#### Ispitni slučaj 4: Testiranje funkcije `delete`

**Ulaz:** `HttpRequest{method="DELETE"}, 68`

**Očekivani rezultat:** Funkcija vraća HTTP statusni kod 401 ("User doesn't exist") jer korisnik s `{idkorisnik = 68}` više ne postoji.

**Rezultat:** Očekivani rezultat je zadovoljen. *Metoda je prošla test.*

#### Ispitni slučaj 5: Testiranje funkcije `update_user_data`

**Ulaz:** `HttpRequest{method="POST", body={'ime': 'Fran', 'prezime': 'Test', 'brkar-`

tice': '0101010101010101'}}, 68

**Očekivani rezultat:** Funkcija pohranjuje dobivene podatke i vraća poruku: "User successfully updated" jer su sve vrijednosti elemenata tijela zahtjeva ispravno napisane.

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 6: Testiranje funkcije `update_user_data`

**Ulaz:** `HttpRequest{method="POST", body={'ime': 'Fr!an', 'prezime': 'Test', 'brkartice': '0101!!0101010101'}}}, 68`

**Očekivani rezultat:** Funkcija vraća HTTP statusni kod 401 ("User doesn't exist or error during decoding") jer vrijednosti elemenata tijela zahtjeva 'ime' i 'brkartice' sadrže nedozvoljene simbole (!).

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 7: Testiranje funkcije `karte_korisnik_id`

**Ulaz:** `HttpRequest{method="GET"}, u.idkorisnik`

**Očekivani rezultat:** Funkcija vraća listu svih kupljenih karata korisnika "u" jer prijavljen korisnik ima administratorska prava (`user.is_superuser = True`).

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 8: Testiranje funkcije `karte_korisnik_id`

**Ulaz:** `HttpRequest{method="GET"}, u.idkorisnik`

**Očekivani rezultat:** Funkcija vraća HTTP statusni kod 401 ("User is not administrator.") jer prijavljen korisnik nema administratorska prava (`user.is_superuser = False`).

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 9: Testiranje funkcije `karte_me`

**Ulaz:** `HttpRequest{method="GET"}`

**Očekivani rezultat:** Funkcija vraća listu svih kupljenih karata prijavljenog korisnika jer ima važeći access token.

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 10: Testiranje funkcije raspored\_filter

**Ulaz:** HttpRequest{method="GET"}, 1, 2, date.today().strftime("%d-%m-%Y")

**Očekivani rezultat:** Funkcija vraća filtriran raspored vožnji jer su vrijednosti parametra polaziste\_id i odrediste\_id između 1 i 5 te je odrediste\_id veće od polaziste\_id.

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 11: Testiranje funkcije raspored\_filter

**Ulaz:** HttpRequest{method="GET"}, 2, 1, date.today().strftime("%d-%m-%Y")

**Očekivani rezultat:** Funkcija vraća HTTP statusni kod 406 ("Polazište ne može biti poslije odredišta.") jer je odrediste\_id manje od polaziste\_id.

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 12: Testiranje funkcije raspored\_filter

**Ulaz:** HttpRequest{method="GET"}, 2, 6, date.today().strftime("%d-%m-%Y")

**Očekivani rezultat:** Funkcija vraća HTTP statusni kod 406 ("Odredište ne postoji.") jer je odrediste\_id veće od 5.

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 13: Testiranje funkcije raspored\_filter

**Ulaz:** HttpRequest{method="GET"}, 0, 2, date.today().strftime("%d-%m-%Y")

**Očekivani rezultat:** Funkcija vraća HTTP statusni kod 406 ("Polazište ne postoji.") jer je polaziste\_id manje od 1.

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

#### Ispitni slučaj 14: Testiranje funkcije dohvat\_vlakove

**Ulaz:** HttpRequest{method="GET"}

**Očekivani rezultat:** Funkcija vraća sve vlakove iz baze podataka.

**Rezultat:** Očekivani rezultat je zadovoljen. **Metoda je prošla test.**

### 5.2.2 Ispitivanje sustava

#### Ispitni slučaj 1: Prijava administratora u sustav te pregled korisnika i njihovih transakcija

##### Uvjeti:

1. Prilikom pokretanja testa korisnik ne smije već biti prijavljen
2. U bazi postoji korisnik administratorskih ovlasti s pruženim podacima za prijavu
3. Poslužitelj je budan (radi izbjegavanja stavljanja velikih čekanja u ispitni slučaj)

##### Ulaz:

1. Odabir stranice za prijavu
2. Unos administratorskog maila i lozinke
3. Odabir gumba za prijavu
4. Odabir administratorske stranice za pregled korisnika
5. Odabir gumba za pregled svih korisničkih transakcija
6. Odabir gumba za odjavu

##### Očekivani rezultat:

1. Prikazuje se stranica za prijavu
2. Vidi se unesen mail te skrivena lozinka
3. Prikazuje se poruka uspješne prijave te se korisnika odvodi na početnu stranicu
4. Otvara se stranica za pregled korisnika te mogućnost upravljanja njima
5. Na istoj stranici se prikazuju sve korisničke transakcije
6. Korisnik je odjavljen

**Rezultat:** Svi očekivani rezultati su zadovoljeni. **Aplikacija je prošla test.**

#### Ispitni slučaj 2: Filtriranje voznog reda

##### Uvjeti:



1. U bazi postoji korisnik s pruženim podacima za prijavu
2. U bazi postoji dovoljan broj stanica da automatski upis u padajući izbornik ne prijeđe njihov broj
3. Unesen je dozvoljen datum za pretraživanje voznog reda
4. Poslužitelj je budan (radi izbjegavanja stavljanja velikih čekanja u ispitni slučaj)

**Ulaz:**

1. Odabir stranice za prijavu
2. Unos korisničkog maila i lozinke
3. Odabir gumba za prijavu
4. Odabir stranice za klasično pretraživanje voznog reda
5. Unos podataka za pretraživanje voznog reda
6. Odabir gumba za pretraživanje
7. Odabir gumba za odjavu

**Očekivani rezultat:**

1. Prikazuje se stranica za prijavu
2. Vidi se unesen mail te skrivena lozinka
3. Prikazuje se poruka uspješne prijave te se korisnika odvodi na početnu stranicu
4. Otvara se stranica za pretraživanje voznog reda
5. Na stranici su popunjeni elementi za pretraživanje voznog reda
6. Na stranici se prikazuju sve dostupne vožnje za dane parametre s mogućom mogućnosti kupnje
7. Korisnik je odjavljen

**Rezultat:** Svi očekivani rezultati su zadovoljeni. [Aplikacija je prošla test.](#)

**Ispitni slučaj 3: Filtriranje voznog reda s neispravnim datumom****Uvjeti:**

1. U bazi postoji korisnik s pruženim podacima za prijavu
2. U bazi postoji dovoljan broj stanica da automatski upis u padajući izbornik ne prijeđe njihov broj
3. Unesen je nedozvoljen datum za pretraživanje voznog reda
4. Poslužitelj je budan (radi izbjegavanja stavljanja velikih čekanja u ispitni slučaj)

**Ulaz:**

1. Odabir stranice za prijavu
2. Unos korisničkog maila i lozinke
3. Odabir gumba za prijavu
4. Odabir stranice za klasično pretraživanje voznog reda
5. Unos podataka za pretraživanje voznog reda
6. Odabir gumba za pretraživanje
7. Odabir gumba za odjavu

**Očekivani rezultat:**

1. Prikazuje se stranica za prijavu
2. Vidi se unesen mail te skrivena lozinka
3. Prikazuje se poruka uspješne prijave te se korisnika odvodi na početnu stranicu
4. Otvara se stranica za pretraživanje voznog reda
5. Na stranici su popunjeni elementi za pretraživanje voznog reda
6. Na stranici se prikazuje poruka o neispravnom unešenom datumu
7. Korisnik je odjavljen

**Rezultat:** Svi očekivani rezultati su zadovoljeni. [Aplikacija je prošla test.](#)

**Ispitni slučaj 4: Prebacivanje sa prikaza stanica na klasični prikaz prilikom kupnje****Uvjeti:**

1. U bazi postoji korisnik s pruženim podacima za prijavu
2. Poslužitelj je budan (radi izbjegavanja stavljanja velikih čekanja u ispitni slučaj)

**Ulaz:**

1. Odabir stranice za prijavu
2. Unos korisničkog maila i lozinke
3. Odabir gumba za prijavu
4. Odabir početne stranice voznog reda
5. Odabir druge stanice u prikazu svih stanica
6. Odabir gumba za kupnju na prvoj prikazanoj vožnji

## 7. Odabir gumba za odjavu

### Očekivani rezultat:

1. Prikazuje se stranica za prijavu
2. Vidi se unesen mail te skrivena lozinka
3. Prikazuje se poruka uspješne prijave te se korisnika odvodi na početnu stranicu
4. Otvara se početna stranica voznog reda
5. Otvara se stranica s vožnjama za odabranu stanicu
6. Otvara se stranica za klasično pretraživanje vožnji s unesenim prvim parametrom pretraživanja
7. Korisnik je odjavljen

**Rezultat:** Svi očekivani rezultati su zadovoljeni. [Aplikacija je prošla test.](#)

## Ispitni slučaj 5: Izmjena podataka korisnika

### Uvjeti:

1. U bazi postoji korisnik s pruženim podacima za prijavu
2. Uneseni podaci za promjenu su ispravni
3. Poslužitelj je budan (radi izbjegavanja stavljanja velikih čekanja u ispitni slučaj)

### Ulaz:

1. Odabir stranice za prijavu
2. Unos korisničkog maila i lozinke
3. Odabir gumba za prijavu
4. Odabir stranice za izmjenu podataka
5. Unos novih podataka
6. Odabir gumba za potvrdu
7. Odabir gumba za odjavu

### Očekivani rezultat:

1. Prikazuje se stranica za prijavu
2. Vidi se unesen mail te skrivena lozinka
3. Prikazuje se poruka uspješne prijave te se korisnika odvodi na početnu stranicu

4. Otvara se stranica za izmjenu podataka
5. U formi se prikazuju novi podaci
6. Prikazuje se stranica za upravljanje računom s novim podacima
7. Korisnik je odjavljen

**Rezultat:** Svi očekivani rezultati su zadovoljeni. **Aplikacija je prošla test.**

---

```
const {Builder, By, Key, until} =
  require('selenium-webdriver');

(async function example() {
  let driver = await new
    Builder().forBrowser('chrome').build();
  try {
    await
      driver.get('http://passdirect-app.herokuapp.com/login');
    //await new Promise(resolve => setTimeout(resolve,
      10000));
    await driver.findElement(By.id('email-input'))
      .sendKeys('gabrijelovadruzina@gmail.com');
    await driver.findElement(By.name('password'))
      .sendKeys('adminadmin');
    await driver.findElement(
      By.className('anew btn btn-2 navlinkother btn-noborder'))
      .click();
    await new Promise(resolve => setTimeout(resolve, 2000));
    await
      driver.get('http://passdirect-app.herokuapp.com/admin');
    let gumbi = await driver.findElements(
      By.className('anew navlinkother btn btn-2 btn-noborder'))
    await gumbi[1].click();
    await new Promise(resolve => setTimeout(resolve, 3000));
    let navbar = await driver.findElements(
      By.className('anew navlinkother btn btn-2 '));
    await navbar[3].click();
    await new Promise(resolve => setTimeout(resolve, 1000));
  } finally {
    await driver.quit();
  }
})();
```

---

Slika 5.1: Ispitni slučaj 1: Prijava administratora u sustav te pregled korisnika i njihovih transakcija

---

```
const {Builder, By, Key, until} =
  require('selenium-webdriver');

(async function example() {
  let driver = await new
    Builder().forBrowser('chrome').build();
  try {
    await
      driver.get('http://passdirect-app.herokuapp.com/login');
    //await new Promise(resolve => setTimeout(resolve,
    //10000));
    await driver.findElement(By.id('email-input'))
      .sendKeys('pass.direct@hotmail.com');
    await driver.findElement(By.name('password'))
      .sendKeys('sifrasifra1');
    await driver.findElement(
      By.className('anew btn btn-2 navlinkother
        btn-noborder')).click();
    await new Promise(resolve => setTimeout(resolve, 2000));
    await driver
      .get('http://passdirect-app.herokuapp.com/timetable-classic');
    await new Promise(resolve => setTimeout(resolve, 1000));
    await
      driver.findElement(By.id('react-select-2-input')).click();
    await
      driver.findElement(By.id('react-select-2-option-0')).click();
    await
      driver.findElement(By.id('react-select-3-input')).click();
    await
      driver.findElement(By.id('react-select-3-option-2')).click();
    await
      driver.findElement(By.name('planDatPoc')).sendKeys('01012023');
    await driver.findElement(
      By.className('anew navlinkother btn btn-2
        btn-noborder')).click();
    await new Promise(resolve => setTimeout(resolve, 3000));
    let navbar = await driver.findElements(
      By.className('anew navlinkother btn btn-2 '));
    await navbar[2].click();
    await new Promise(resolve => setTimeout(resolve, 1000));
  } finally {
    await driver.quit();
  }
})();
```

---

Slika 5.2: Ispitni slučaj 2: Filtriranje voznog reda

---

```
const {Builder, By, Key, until} =
  require('selenium-webdriver');

(async function example() {
  let driver = await new
    Builder().forBrowser('chrome').build();
  try {
    await
      driver.get('http://passdirect-app.herokuapp.com/login');
    //await new Promise(resolve => setTimeout(resolve,
      10000));
    await driver.findElement(By.id('email-input'))
      .sendKeys('pass.direct@hotmail.com');
    await driver.findElement(By.name('password'))
      .sendKeys('sifrasifra1');
    await driver.findElement(
      By.className('anew btn btn-2 navlinkother
        btn-noborder')).click();
    await new Promise(resolve => setTimeout(resolve, 2000));
    await driver
      .get('http://passdirect-app.herokuapp.com/timetable-classic');
    await new Promise(resolve => setTimeout(resolve, 1000));
    await
      driver.findElement(By.id('react-select-2-input')).click();
    await
      driver.findElement(By.id('react-select-2-option-0')).click();
    await
      driver.findElement(By.id('react-select-3-input')).click();
    await
      driver.findElement(By.id('react-select-3-option-2')).click();
    await
      driver.findElement(By.name('planDatPoc')).sendKeys('01012024');
    await driver.findElement(
      By.className('anew navlinkother btn btn-2 btn-noborder
        ')).click();
    await new Promise(resolve => setTimeout(resolve, 3000));
    let navbar = await driver.findElements(
      By.className('anew navlinkother btn btn-2 '));
    await new Promise(resolve => setTimeout(resolve, 6000));
    await navbar[2].click();
    await new Promise(resolve => setTimeout(resolve, 1000));
  } finally {
    await driver.quit();
  }
})();
```

---

Slika 5.3: Ispitni slučaj 3: Filtriranje voznog reda s neispravnim datumom

---

```
const {Builder, By, Key, until} =
  require('selenium-webdriver');

(async function example() {
  let driver = await new
    Builder().forBrowser('chrome').build();
  try {
    await
      driver.get('http://passdirect-app.herokuapp.com/login');
    //await new Promise(resolve => setTimeout(resolve,
    //  10000));
    await driver.findElement(By.id('email-input'))
      .sendKeys('pass.direct@hotmail.com');
    await driver.findElement(By.name('password'))
      .sendKeys('sifrasifra1');
    await driver.findElement(
      By.className('anew btn btn-2 navlinkother
        btn-noborder')).click();
    await new Promise(resolve => setTimeout(resolve, 1000));
    await driver
      .get('http://passdirect-app.herokuapp.com/timetable/2');
    await new Promise(resolve => setTimeout(resolve, 2000));
    var kol = await driver.findElements(
      By.className('station-grid-item'));
    await kol[11].click();
    await new Promise(resolve => setTimeout(resolve, 3000));
    let navbar = await driver.findElements
      (By.className('anew navlinkother btn btn-2 '));
    await navbar[2].click();
    await new Promise(resolve => setTimeout(resolve, 1000));
  } finally {
    await driver.quit();
  }
})();
```

---

Slika 5.4: Ispitni slučaj 4: Prebacivanje sa prikaza stanica na klasični prikaz prilikom kupnje

---

```
const {Builder, By, Key, until} =
  require('selenium-webdriver');

(async function example() {
  let driver = await new
    Builder().forBrowser('chrome').build();
  try {
    await
      driver.get('http://passdirect-app.herokuapp.com/login');
    //await new Promise(resolve => setTimeout(resolve,
      10000));
    await driver.findElement(By.id('email-input'))
      .sendKeys('pass.direct@hotmail.com');
    await driver.findElement(By.name('password'))
      .sendKeys('sifrasifra1');
    await driver.findElement(
      By.className('anew btn btn-2 navlinkother
        btn-noborder')).click();
    await new Promise(resolve => setTimeout(resolve, 1000));
    await
      driver.get('http://passdirect-app.herokuapp.com/account');
    await new Promise(resolve => setTimeout(resolve, 1000));
    await
      driver.findElement(By.className('button-link')).click();
    await new Promise(resolve => setTimeout(resolve, 1000));
    await driver.findElement(By.id('ime')).sendKeys('Ime');
    await
      driver.findElement(By.id('prezime')).sendKeys('Prezime');
    await driver.findElement(By.id('broj_kartice'))
      .sendKeys('1234567890123456');
    await new Promise(resolve => setTimeout(resolve, 1000));
    await driver.findElement(
      By.className('anew btn btn-2 navlinkother
        btn-noborder')).click();
    await new Promise(resolve => setTimeout(resolve, 3000));
    let navbar = await driver.findElements(
      By.className('anew navlinkother btn btn-2 '));
    await navbar[2].click();
    await new Promise(resolve => setTimeout(resolve, 1000));
  } finally {
    await driver.quit();
  }
})();
```

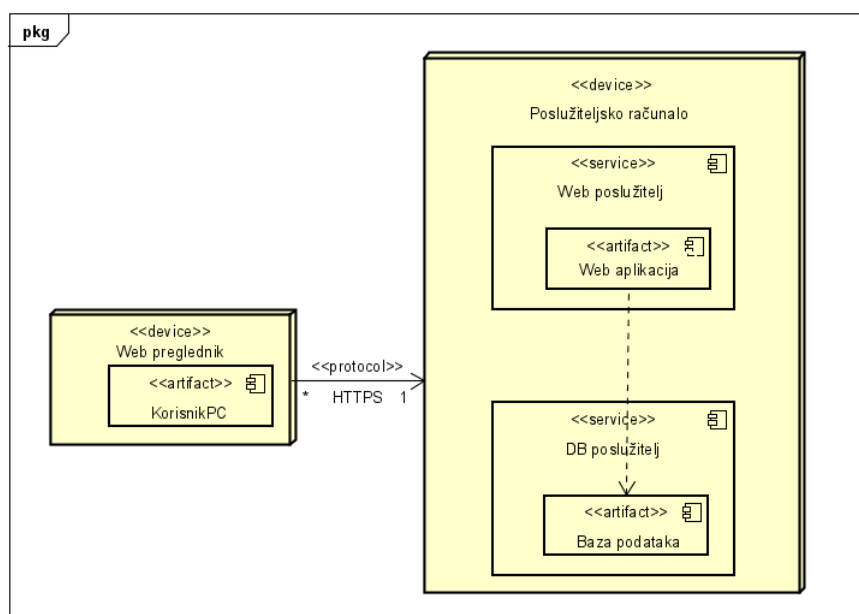
---

Slika 5.5: Ispitni slučaj 5: Izmjena podataka korisnika



## 5.3 Dijagram razmještaja

Dijagram razmještaja opisuje topologiju sklopovlja i programsku potporu korištenu u implementaciji sustava u njegovom radnom okruženju. Na poslužiteljskom računalu nalaze se web poslužitelj i poslužitelj baze podataka. Klijenti koriste web preglednik kako bi pristupili web aplikaciji. Sustav je baziran na arhitekturi "klijent – poslužitelj", a komunikacija između računala korisnika (klijent, zaposlenik, vlasnik, administrator) i poslužitelja odvija se preko HTTPS veze.

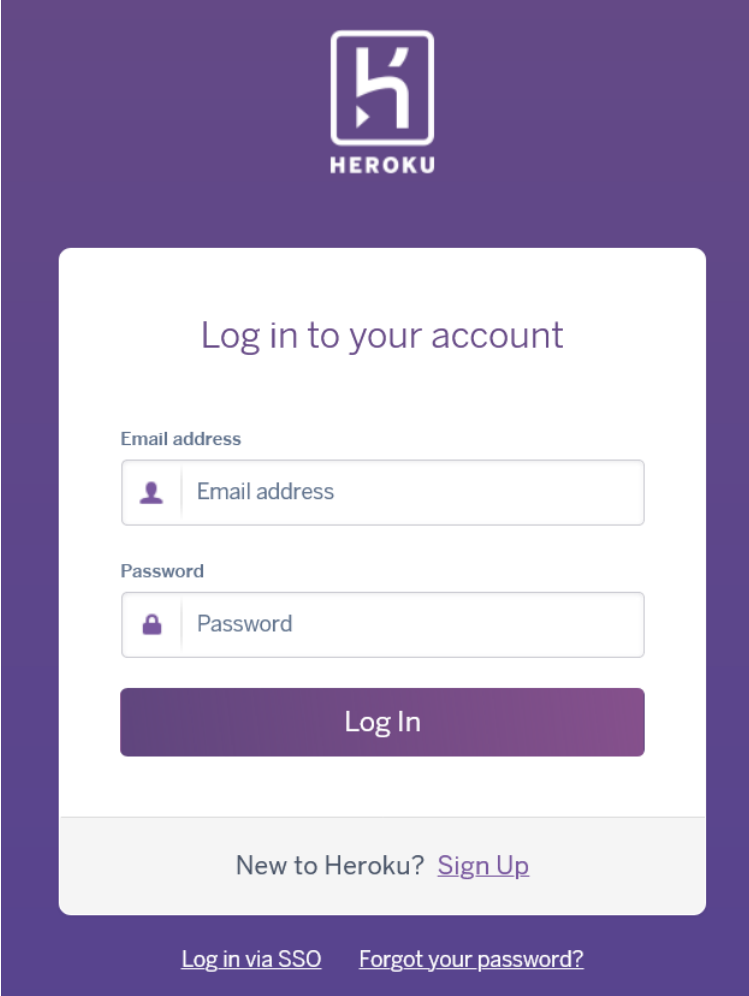


Slika 5.6: Dijagram razmještaja

## 5.4 Upute za puštanje u pogon

### 5.4.1 Postavljanje Herokua

Potrebno je kreirati račun (registrirati se) te se zatim u njega prijaviti preko Heroku web aplikacije.



Slika 5.7: Prijava u Heroku

Nakon prijave potrebno je podesiti ime aplikacije i lokaciju unutar Herokua. Potrebno je napraviti dvije aplikacije - po jednu za *frontend* i *backend* dio.

Nakon preuzimanja aplikacije s GitLaba, za *frontend* dio potrebno je pozicionirati se u mapu izvorniKod/frontend/passdirect/. Nakon toga, u komandnoj liniji u mapi projekta potrebno je redom pokrenuti sljedeće komande za kreiranje novog Git repozitorija i puštanje aplikacije u pogon:

```
$ cd <ime direktorija>
```

```
$ git init
```

```
$ heroku git:remote -a <ime aplikacije>
```

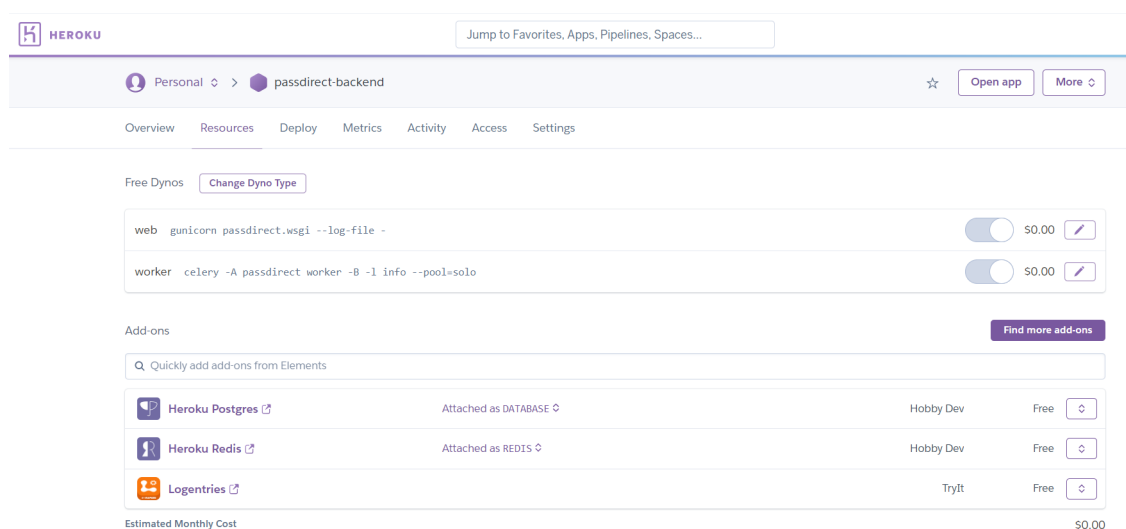
te komande za *deployanje* aplikacije na Heroku:

```
$ git add .
```

```
$ git commit -m "poruka"
```

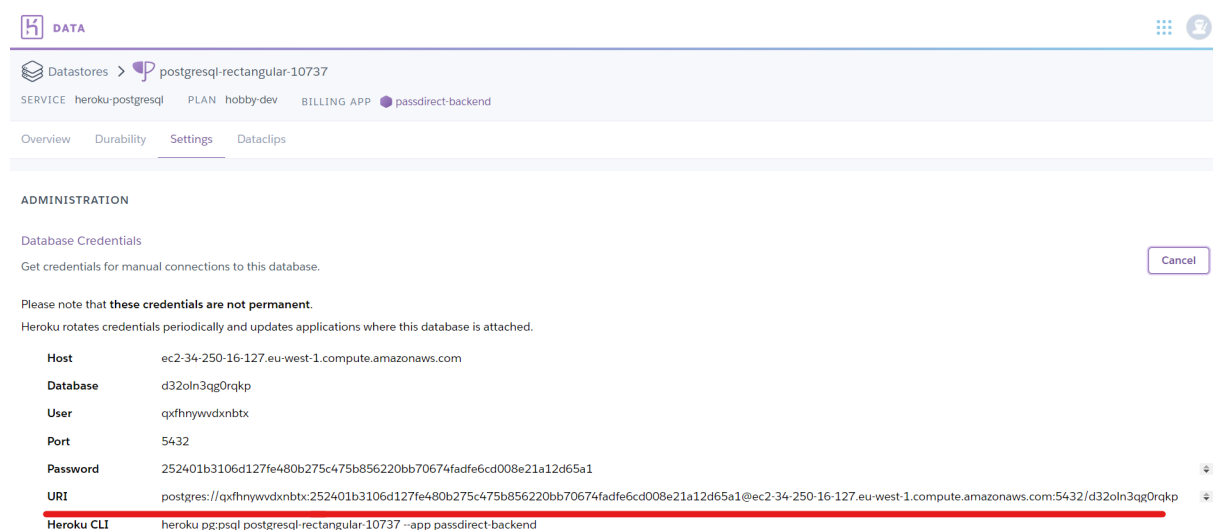
```
$ git push heroku master
```

Za *backend* dio potrebno je pozicionirati se u mapu izvorniKod/backend, a ostatak postupka je isti. Zatim je potrebno otići na *dashboard.heroku.com* i za backend aplikaciju instalirati dodatke Heroku Postgres i Heroku Redis.



Slika 5.8: Heroku dodaci

Potom je potrebno otići pod postavke Heroku Postgres dodatka i kopirati podatke za pristup bazi te zatim te podatke staviti u izvorniKod/backend/passdirect/settings.py .



**ADMINISTRATION**

Database Credentials

Get credentials for manual connections to this database.

Please note that **these credentials are not permanent**.  
Heroku rotates credentials periodically and updates applications where this database is attached.

|            |                                                                                                                                                                 |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Host       | ec2-34-250-16-127.eu-west-1.compute.amazonaws.com                                                                                                               |
| Database   | d32oln3qg0rqkp                                                                                                                                                  |
| User       | qxhnywvdxnbtx                                                                                                                                                   |
| Port       | 5432                                                                                                                                                            |
| Password   | 252401b3106d127fe480b275c475b856220bb70674fadfe6cd008e21a12d65a1                                                                                                |
| URI        | postgres://qxhnywvdxnbtx:252401b3106d127fe480b275c475b856220bb70674fadfe6cd008e21a12d65a1@ec2-34-250-16-127.eu-west-1.compute.amazonaws.com:5432/d32oln3qg0rqkp |
| Heroku CLI | heroku pg:psql postgresql-rectangular-10737 --app passdirect-backend                                                                                            |

Slika 5.9: Heroku Postgres Dodatak

```

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'd32oln3qg0rqkp',
        'USER': 'qxhnywvdxnbtx',
        'PASSWORD': '252401b3106d127fe480b275c475b856220bb70674fadfe6cd008e21a12d65a1',
        'HOST': 'ec2-34-250-16-127.eu-west-1.compute.amazonaws.com',
        'PORT': '5432',
        'TEST': {
            'MIRROR': 'default',
        },
        'OPTIONS': {
            'sslmode': 'require',
        },
    },
}

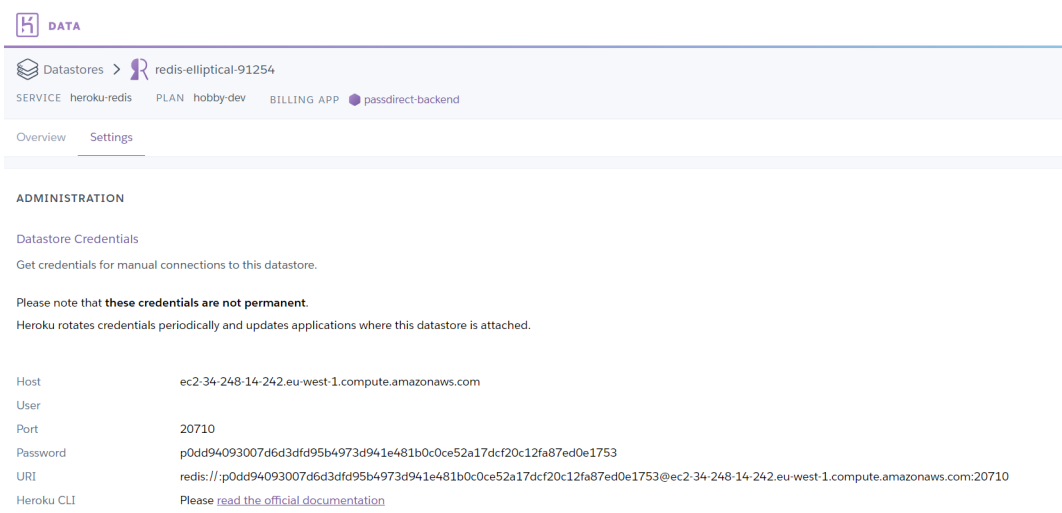
```

Slika 5.10: izvorniKod/backend/passdirect/settings.py

Da bi napravili tablice potrebno je pozicionirati se u direktorij izvorniKod/backend/ i pokrenuti skriptu manage.py s argumentom makemigrations dakle; *manage.py makemigrations* i nakon toga *manage.py migrate*.

Zatim korisnik treba popuniti tablice vlak, stanica i vlakstanica(raspored) vlakovima, stanicama i rasporedom. Pretpostavljeno je vrijeme čekanja na stanici od 5 minuta i vrijeme putovanja 10 minuta između svake stanice.

Da bi uspostavili brokera poruka za servis distribuiranih zadataka Celery, potrebno je otići u postavke Heroku Redis dodatka te kopirati podatke za pristup (port i URI).



Slika 5.11: Heroku Redis Settings

Navedene podatke potrebno je upisati u *izvorniKod/backend/passdirect/celery.py* na mjesto prikazno Slikom 5.12.

```
# - namespace='CELERY' means all celery-related configuration keys
# should have a "CELERY_" prefix.
app.config_from_object('django.conf:settings', namespace='CELERY')
os.environ['REDIS_URL'] = 'redis://:p0dd94093007d6d3dfd95b4973d941e481b0c0ce52a17dcf20c12fa87ed0e1753@ec2-34-248-14-242.eu-west-1.compu
app.config.update(BROKER_URL=os.environ['REDIS_URL'],
                  CELERY_RESULT_BACKEND=os.environ['REDIS_URL'])

# Load task modules from all registered Django apps.
app.autodiscover_tasks()
```

Slika 5.12: izvorniKod/backend/passdirect/celery.py

Zadnji korak sastoji se od unosa podataka za prijavu na mail poslužitelj.

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_PORT = 587
EMAIL_HOST_USER = 'gabrijelovadruzina@gmail.com'
EMAIL_HOST_PASSWORD = 'idnlnxlemksmvagj'
EMAIL_USE_TLS = True
DEFAULT_FROM_EMAIL = 'passdirect@noreply.com'
```

Slika 5.13: Mail Poslužitelj

## 6. Zaključak i budući rad

Zadatak projektnog tima bila je izrada web aplikacije čija je primarna svrha prikaz informacija o voznom redu i kupnju karata za vlakove jedne željezničke tvrtke, ali koja ima jednu dodatnu funkcionalnost koja ju čini inovativnom, a to je mogućnost predlaganja vagona i dijela vagona za ukrcaj putnika u svrhu optimalnog rasporeda mase u vlaku.

Nakon nepunih 17 tjedana timskega rada, zadani cilj je ostvaren. Sama izvedba projekta protezala se kroz dvije faze.

U prvoj fazi, koja je trajala od sredine listopada do početka prosinca, započela je formiranjem projektnog tima. Od početka se intenzivno radilo na implementaciji generičkih funkcionalnosti, u čemu su pomogli i izrađeni vizuali (obrasci i dijagrami). Sastanci projektnog tima u prvom ciklusu bili su česti jer se dio poslova preklapao, a članovi su samostalno istraživali tehnologije koje su im bile potrebne, a s kojima se dotada nisu susreli.

Na početku druge faze, koja je trajala od sredine prosinca do sredine siječnja, radilo se na prepravljanju pogrešaka iz prve verzije dokumentacije i izmjeni aplikacije prema zahtjevima asistentice. Nakon toga, počelo se intenzivno raditi na što skorijem dovršavanju aplikacije kako bi ostalo dovoljno vremena za eventualne izmjene i dopune.

Sudjelovanje u izradi aplikacije u sklopu ovog projekta predstavlja vrijedno iskustvo za sve članove tima jer smo kroz intenzivan rad tijekom više od četiri mjeseca uspjeli postići zadani cilj i izraditi jedan jednostavan, ali kompletan programski proizvod. Osim toga, uvidjeli smo važnost dobre vremenske organizacije te suradnje između svih članova, jer bi za eventualan neuspjeh projekta svi članovi projektnog tima snosili odgovornost. Iako smo svjesni da u aplikaciji postoji puno mjesta za usavršavanje i dotjerivanje, ponosni smo na finalan proizvod jer je to za većinu projektnog tima bio prvi takav projekt.

# Popis literature

1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
2. The Unified Modeling Language, <https://www.uml-diagrams.org/>
3. Astah Community, <http://astah.net/editions/uml-new>
4. Django 3.2 Documentation, <https://docs.djangoproject.com/en/3.2/>
5. MDN Web Docs - Django Web Framework, <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django>
6. Djoser Documentation, <https://djoser.readthedocs.io/en/2.1.0/index.html>
7. Simple JWT Documentation, <https://django-rest-framework-simplejwt.readthedocs.io/en/latest/>
8. React Documentation, <https://reactjs.org/>
9. Redux Usage Guide, <https://redux.js.org/usage/index>
10. The Overview of the Gotcha System, [https://www.researchgate.net/figure/The-overview-of-the-Gotcha-system\\_fig1\\_301428444](https://www.researchgate.net/figure/The-overview-of-the-Gotcha-system_fig1_301428444)
11. Celery - Distributed Task Queue Documentation, <https://docs.celeryproject.org/en/stable/>
12. Selenium Documentation, <https://www.selenium.dev/documentation/>

# Indeks slika i dijagrama

|      |                                                                                                            |    |
|------|------------------------------------------------------------------------------------------------------------|----|
| 2.1  | <i>Gotcha</i> sustav senzora . . . . .                                                                     | 5  |
| 3.1  | Dijagram obrasca uporabe, funkcionalnost korisnika i klijenta . . .                                        | 15 |
| 3.2  | Dijagram obrasca uporabe, funkcionalnost administratora . . . . .                                          | 16 |
| 3.3  | Sekvencijski dijagram za UC6 . . . . .                                                                     | 17 |
| 3.4  | Sekvencijski dijagram za UC7 . . . . .                                                                     | 18 |
| 3.5  | Sekvencijski dijagram za UC1 . . . . .                                                                     | 19 |
| 4.1  | Skica arhitekture sustava . . . . .                                                                        | 21 |
| 4.2  | Dijagram baze podataka . . . . .                                                                           | 25 |
| 4.3  | ER Dijagram baze podataka . . . . .                                                                        | 26 |
| 4.4  | Dijagram razreda 1 . . . . .                                                                               | 27 |
| 4.5  | Dijagram razreda 2 . . . . .                                                                               | 28 |
| 4.6  | Dijagram razreda 3 . . . . .                                                                               | 28 |
| 4.7  | Dijagram stanja . . . . .                                                                                  | 30 |
| 4.8  | Dijagram aktivnosti . . . . .                                                                              | 32 |
| 4.9  | Dijagram komponenti . . . . .                                                                              | 34 |
| 5.1  | Ispitni slučaj 1: Prijava administratora u sustav te pregled korisnika<br>i njihovih transakcija . . . . . | 43 |
| 5.2  | Ispitni slučaj 2: Filtriranje voznog reda . . . . .                                                        | 44 |
| 5.3  | Ispitni slučaj 3: Filtriranje voznog reda s neispravnim datumom . .                                        | 45 |
| 5.4  | Ispitni slučaj 4: Prebacivanje sa prikaza stanica na klasični prikaz<br>prilikom kupnje . . . . .          | 46 |
| 5.5  | Ispitni slučaj 5: Izmjena podataka korisnika . . . . .                                                     | 47 |
| 5.6  | Dijagram razmještaja . . . . .                                                                             | 48 |
| 5.7  | Prijava u Heroku . . . . .                                                                                 | 49 |
| 5.8  | Heroku dodaci . . . . .                                                                                    | 50 |
| 5.9  | Heroku Postgres Dodatak . . . . .                                                                          | 51 |
| 5.10 | izvorniKod/backend/passdirect/settings.py . . . . .                                                        | 51 |
| 5.11 | Heroku Redis Settings . . . . .                                                                            | 52 |



|                                                             |    |
|-------------------------------------------------------------|----|
| 5.12 izvorniKod/backend/passdirect/celery.py . . . . .      | 52 |
| 5.13 Mail Poslužitelj . . . . .                             | 52 |
| 6.1 Prikaz ukupne aktivnosti na repozitoriju . . . . .      | 61 |
| 6.2 Prikaz pojedinačne aktivnosti na repozitoriju . . . . . | 61 |

# Dodatak: Prikaz aktivnosti grupe

## Dnevnik sastajanja

### 1. sastanak

- Datum: 11. listopada 2021.
- Prisustvovali: L. Grgurić Mileusnić, P. Hrvoić, I. Ivanković, G. Jambrošić, K. Jurić, F. Marčec, F. Mička
- Teme sastanka:
  - instalacija git-a i definiranje strukture projekta
  - diskusija oko zadatka i njegovih dijelova

### 2. sastanak

- Datum: 14. listopada 2021.
- Prisustvovali: L. Grgurić Mileusnić, P. Hrvoić, I. Ivanković, G. Jambrošić, K. Jurić, F. Marčec, F. Mička
- Teme sastanka:
  - prvi sastanak s asistenticom i demonstratoricom
  - analiza i pojašnjenje zadatka

### 3. sastanak

- Datum: 28. listopada 2021.
- Prisustvovali: L. Grgurić Mileusnić, P. Hrvoić, I. Ivanković, G. Jambrošić, K. Jurić, F. Marčec, F. Mička
- Teme sastanka:
  - razrada dionika i njihovih zahtjeva i funkcionalnosti
  - dizajniranje baze podataka

### 4. sastanak

- Datum: 11. studenoga 2021.
- Prisustvovali: L. Grgurić Mileusnić, I. Ivanković, G. Jambrošić, K. Jurić, F. Marčec
- Teme sastanka:
  - online sastanak s asistenticom i demonstratoricom

- savjetovanje za daljnji napredak aplikacije, upućivanje na dijelove kojima je potrebna dorada

#### 5. sastanak

- Datum: 17. studenog 2021.
- Prisustvovali: L. Grgurić Mileusnić, I. Ivanković, G. Jambrošić, K. Jurič, F. Marčec, F. Mička
- Teme sastanka:
  - demonstracija prve verzije projekta

#### 6. sastanak

- Datum: 17. prosinca 2021.
- Prisustvovali: L. Grgurić Mileusnić, I. Ivanković, G. Jambrošić, K. Jurič, F. Marčec, F. Mička
- Teme sastanka:
  - plan daljnjeg rada na ispravljanju pogrešaka i dovršavanju aplikacije

#### 7. sastanak

- Datum: 5. siječnja 2021.
- Prisustvovali: L. Grgurić Mileusnić, I. Ivanković, G. Jambrošić, K. Jurič, F. Marčec, F. Mička
- Teme sastanka:
  - pregled dosad napravljenog posla
  - podjela posla pri izradi dokumentacije

#### 8. sastanak

- Datum: 13. prosinca 2021.
- Prisustvovali: L. Grgurić Mileusnić, I. Ivanković, G. Jambrošić, K. Jurič, F. Marčec, F. Mička
- Teme sastanka:
  - pregled napravljenog posla
  - finalne izmjene

## Tablica aktivnosti

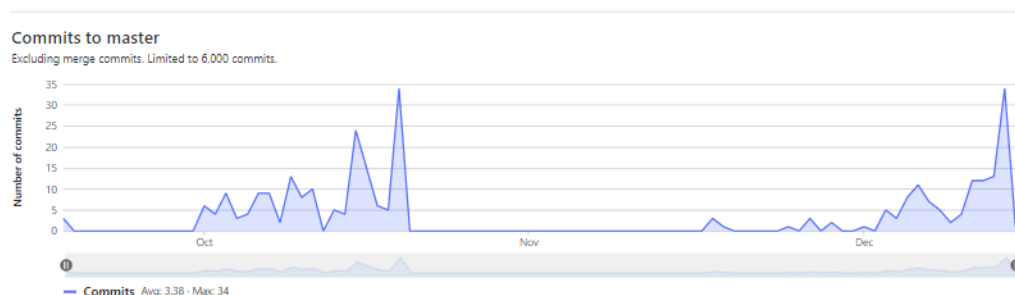
|                                  | Gabrijel Jambrošić | Lovro Grgurić Mileusnić | Petar Hrvoić | Iva Maria Ivanković | Katarina Jurič | Filip Marčec | Fran Mička |
|----------------------------------|--------------------|-------------------------|--------------|---------------------|----------------|--------------|------------|
| Upravljanje projektom            | 4                  | 3                       |              |                     |                | 3            |            |
| Opis projektnog zadatka          |                    | 1                       |              | 3                   |                |              |            |
| Funkcionalni zahtjevi            |                    | 1                       | 0.5          | 1                   | 0.5            | 2            |            |
| Opis pojedinih obrazaca          | 0.5                |                         | 1            |                     |                |              | 1          |
| Dijagram obrazaca                |                    |                         |              |                     |                |              | 2          |
| Sekvencijski dijagrami           |                    |                         |              |                     |                |              | 2          |
| Opis ostalih zahtjeva            |                    |                         |              | 1                   |                |              |            |
| Arhitektura i dizajn sustava     |                    | 1                       |              | 1                   | 2              |              |            |
| Baza podataka                    |                    | 4                       |              | 2                   | 1              | 1            |            |
| Dijagram razreda                 |                    | 1                       |              |                     | 3              |              |            |
| Dijagram stanja                  |                    | 0.5                     |              |                     | 2              |              |            |
| Dijagram aktivnosti              |                    |                         |              |                     | 2              |              |            |
| Dijagram komponenti              |                    | 0.5                     |              | 2                   |                |              |            |
| Korištene tehnologije i alati    |                    | 0.5                     |              | 1                   |                |              |            |
| Ispitivanje programskog rješenja | 5                  |                         | 3            |                     |                |              | 2          |
| Dijagram razmještaja             |                    |                         |              |                     |                |              |            |
| Upute za puštanje u pogon        |                    | 1.5                     |              |                     | 1              |              |            |
| Dnevnik sastajanja               | 0.5                |                         |              |                     | 1              |              |            |

Nastavljeno na idućoj stranici

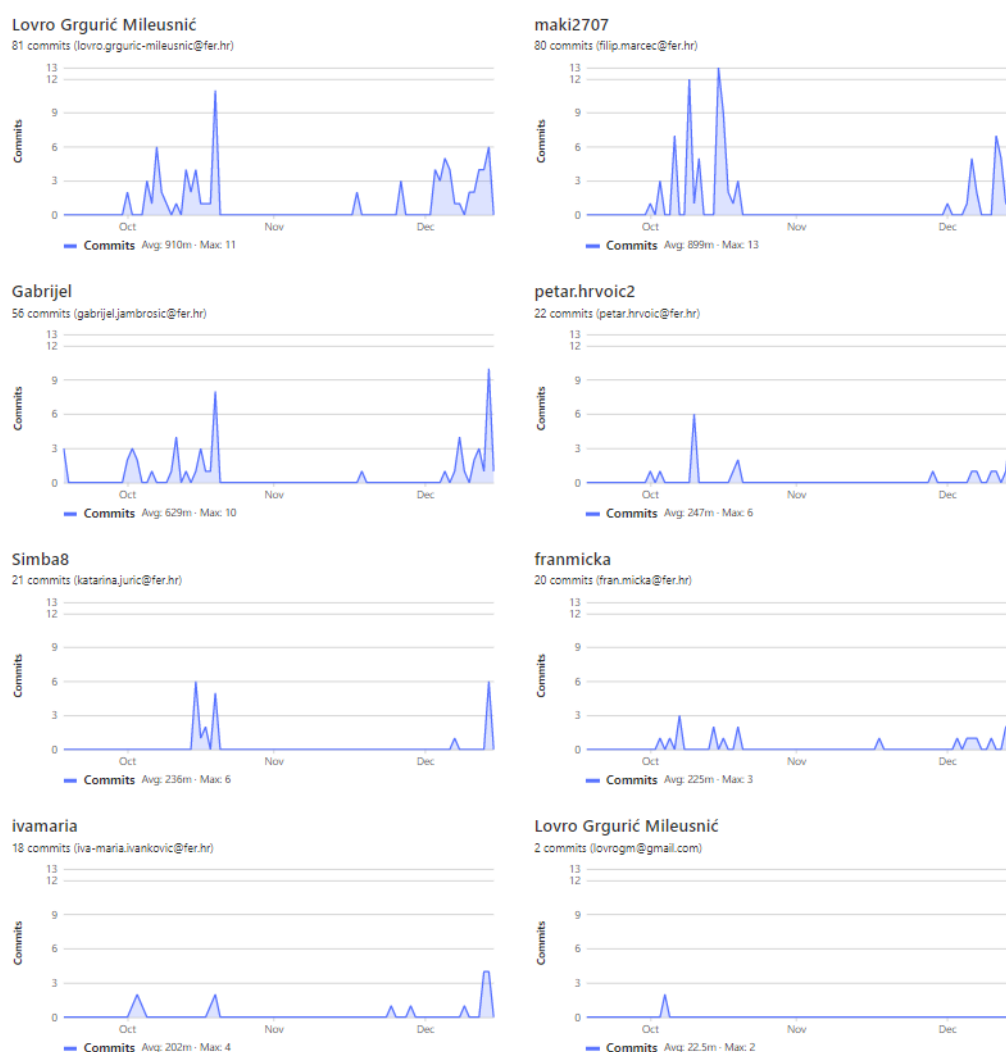
Nastavljeno od prethodne stranice

|                                        | Gabrijel Jambrošić | Lovro Grgurić Mileusnić | Petar Hrvoić | Iva Maria Ivanković | Katarina Jurić | Filip Marčec | Fran Mička |
|----------------------------------------|--------------------|-------------------------|--------------|---------------------|----------------|--------------|------------|
| Zaključak i budući rad                 |                    |                         |              | 1                   |                |              |            |
| Popis literature                       | 0.1                | 0.1                     |              |                     |                |              |            |
|                                        |                    |                         |              |                     |                |              |            |
| <b>Izrada aplikacije</b>               |                    |                         |              |                     |                |              |            |
| izrada početne stranice                |                    |                         | 6            |                     |                |              |            |
| izrada baze podataka                   |                    | 4.5                     |              |                     |                |              |            |
| spajanje s bazom podataka              |                    | 3                       |              |                     |                |              | 2          |
| back end                               |                    | 35                      |              |                     |                |              | 15         |
| izrada servisa za zakazivanje zadataka |                    | 12                      |              |                     |                |              |            |
| izrada autentifikacijskog sistema      |                    | 5                       |              |                     |                | 16           |            |
| izrada korisničkog sistema             | 13                 |                         |              |                     |                |              |            |
| izrada navigacijske trake              | 1                  |                         |              |                     |                | 0,5          |            |
| izrada voznog reda                     | 7                  |                         |              |                     |                | 7            |            |
| izrada kupnje karata                   |                    |                         | 11           |                     |                |              |            |
| izrada generatora podataka senzora     |                    |                         | 2            |                     |                |              |            |
| izrada svih pregleda transakcija       |                    |                         |              |                     |                | 7            |            |

## Dijagrami pregleda promjena



Slika 6.1: Prikaz ukupne aktivnosti na repozitoriju



Slika 6.2: Prikaz pojedinačne aktivnosti na repozitoriju