

Programsko inženjerstvo

Ak. god. 2020./2021.

Terminko za praonico rublja

Dokumentacija, Rev. 2

Grupa: Janezi

Voditelj: *Jan Grgić*

Datum predaje: *14. siječnja 2021.*

Nastavnik: *Izv. Prof. Dr. Sc. Vlado Sruk*

Sadržaj

1	Dnevnik promjena dokumentacije	3
2	Opis projektnog zadatka	6
2.1	Opis problema i motivacija	6
2.2	Postojeća slična rješenja	6
2.3	Potencijalno zainteresirani korisnici	9
2.4	Mogućnost prilagodbe rješenja	9
2.5	Opseg projektnog zadatka (zahtjevi sustava)	9
2.6	Nadogradnje projektnog zadatka i planovi za budućnost	10
3	Specifikacija programske potpore	12
3.1	Funkcionalni zahtjevi	12
3.1.1	Obrasci uporabe	14
3.1.2	Sekvencijski dijagrami	27
3.2	Ostali zahtjevi	32
4	Arhitektura i dizajn sustava	33
4.1	Baza podataka	35
4.1.1	Opis tablica	35
4.1.2	Dijagram baze podataka	39
4.2	Dijagram razreda	41
4.3	Dijagram stanja	45
4.4	Dijagram aktivnosti	47
4.5	Dijagram komponenti	49
5	Implementacija i korisničko sučelje	50
5.1	Korištene tehnologije i alati	50
5.2	Ispitivanje programskog rješenja	52
5.2.1	Ispitivanje komponenti	52
5.2.2	Ispitivanje sustava	56
5.3	Dijagram razmještaja	60

5.4 Upute za puštanje u pogon	61
6 Zaključak i budući rad	68
Popis literature	69
Indeks slika i dijagrama	71
Dodatak: Prikaz aktivnosti grupe	72

1. Dnevnik promjena dokumentacije

Kontinuirano osvježavanje

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak.	D. Grgić	20.10.2020.
0.2	Promijenjeni dijelovi predložka.	J. Grgić	21.10.2020.
0.3	Prepisan opis projektnog zadatka iz word dokumenta koji je sastavljen ranije radi prijave vlastite teme.	J. Grgić	22.10.2020.
0.4	Dodani dionici u poglavlju 3, navedeni aktori i funkcionalni zahtjevi za neregistrirane/neprijavljene korisnike. Napisani obrasci uporabe za neregistrirane/neprijavljene korisnike.	J. Grgić	23.10.2020.
0.5	Uređen dnevnik sastajanja i podjela poslova u grupi.	J. Grgić	23.10.2020.
0.6	Dodani funkcionalni zahtjevi za Administratora.	I. Joskić	23.10.2020.
0.7	Dodani funkcionalni zahtjevi za Korisnika.	B. Spiegl	23.10.2020.
0.8	Dodani obrasci uporabe za Administratora.	I. Joskić	23.10.2020.
0.9	Navedeni funkcionalni zahtjevi za Zaposlenika.	M. Dragošević	24.10.2020.
0.10	Dodani obrasci uporabe za Zaposlenika.	M. Dragošević	24.10.2020.
0.11	Dodani opisi tablica u bazi podataka.	D. Šmigovec	25.10.2020.
0.12	Promjena baze podataka.	D. Šmigovec	25.10.2020.
0.13	Dodan 3. sastanak u dnevnik sastajanja	J. Grgić	26.10.2020.
0.14	Dodani ostali zahtjevi.	D. Grgić	26.10.2020.
0.15	Dodano još obrazaca uporabe za Administratora.	I. Joskić	26.10.2020.

Rev.	Opis promjene/dodatka	Autori	Datum
0.16	Dodano još obrazaca uporabe za Registriranog korisnika i modificirani funkcionalni zahtjevi zajedno sa odgovarajućim obrascima upotrebe.	B. Spiegl	29.10.2020.
0.17	Dodan obrazac uporabe za plaćanje i prvi dijagram obrasca uporabe za registrirane i neregistrirane korisnike.	J. Grgić	31.10.2020.
0.18	Dodani sekvencijski dijagrami za UC1 i UC11.	I. Joskić	31.10.2020.
0.19	Dodani sekvencijski dijagrami za UC3 i UC4.	D. Šmigovec	31.10.2020.
0.20	Dodani opisi za sekvencijske dijagrame 3.4 i 3.7.	I. Joskić	1.11.2020.
0.21	Dodani dijagrami obrazaca uporabe za zaposlenika i administratora.	B. Spiegl	1.11.2020.
0.22	Dodan opis i slika arhitekture sustava.	D. Grgić	1.11.2020.
0.23	Popravljen dijagram obrazaca uporabe za registriranog i neregistriranog korisnika.	J. Grgić	1.11.2020.
0.24	Dodana shema Django baze.	D. Šmigovec	1.11.2020.
0.25	Popravljen UC2 i dijagram obrasca uporabe 1. Dodana literatura.	J. Grgić	1.11.2020.
0.26	Dodani razredni dijagrami.	D. Grgić	7.11.2020.
0.27	Nadopunjeni opisi tablica i promjena atributa na engleski.	D. Šmigovec	7.11.2020.
0.28	Nadopunjeni dijagrami razreda prema pretpostavljenoj završnoj implementaciji.	D. Grgić	9.11.2020.
1.0	Verzija za prvi ciklus predaje projekta.	J. Grgić	13.11.2020.
1.1	Dodan dijagram komponenti.	B. Spiegl	9.1.2021.
1.2	Dodani testovi.	J. Grgić	9.1.2021.
1.3	Dodani sastanci.	J. Grgić	9.1.2021.
1.4	Dodan dijagram aktivnosti za rezervaciju termina.	M. Dragošević	10.1.2021.
1.5	Dodan zaključak.	M. Dragošević	10.1.2021.
1.6	Dodane upute za puštanje u pogon.	J. Grgić	9.1.2021.

Rev.	Opis promjene/dodatka	Autori	Datum
1.7	Promjene na bazi podataka.	D. Šmigovec	11.1.2021.
1.8	Dodan opis korištenih tehnologija i alata.	D. Šmigovec	11.1.2021.
1.9	Promjene na dijagramima razreda i dodatak ispita sustava.	D. Grgić	12.1.2021.
1.10	Dodani dijagrami pregleda promijena.	J. Grgić	14.1.2021.
2.0	Verzija za drugi ciklus predaje projekta.	J. Grgić	14.1.2021.

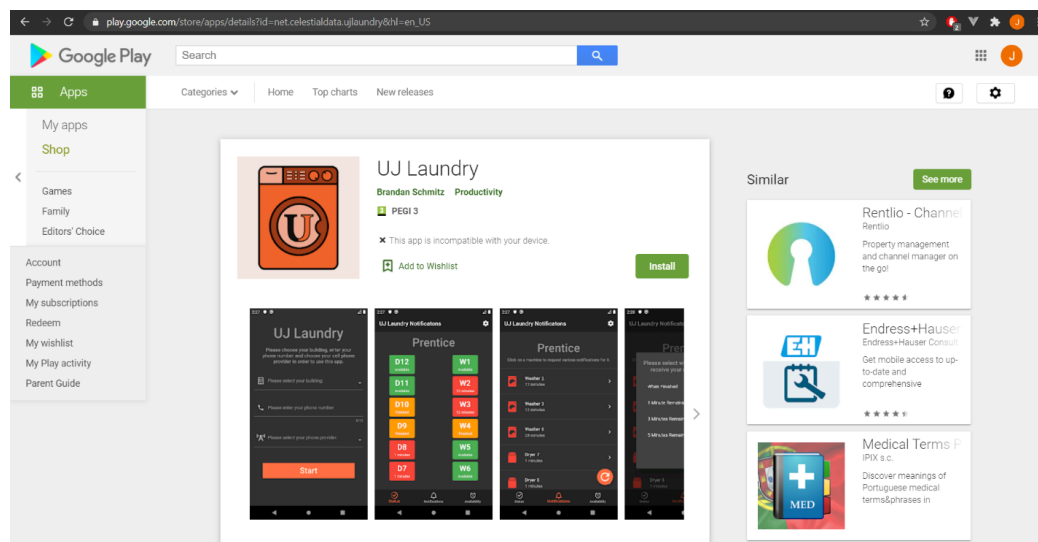
2. Opis projektnog zadatka

2.1 Opis problema i motivacija

Jedan je od glavnih problema života u studentskom domu pranje rublja. Studenti smješteni u domovima moraju paziti na raspored pranja više nego na rokove za prijavu ispita, jer što im znači prijavljen ispit ako na njega nemaju u čemu doći. Motivacija za izradu ove web aplikacije leži u nepraktičnom sustavu rezerviranja termina pranja odjeće. Često se zna dogoditi da se student dođe upisati, ali naiđe na zatvorena vrata zbog pauza koje su u različitim vremenima ili nađe kompletno popunjen kalendar te ne može oprati rublje. Želimo da se to iskustvo olakša studentima, ali i zaposlenicima praonice. Na ovaj bi se način mogli puno jednostavnije rezervirati, ali i otkazati termini za pranje koji bi se onda lako popunili te bi se studenti mogli puno bolje organizirati pa čak i upisati za termin koji je netom otkazan. Moderniziralo bi se i plaćanje te penaliziralo često otkazivanje dragocjenih termina.

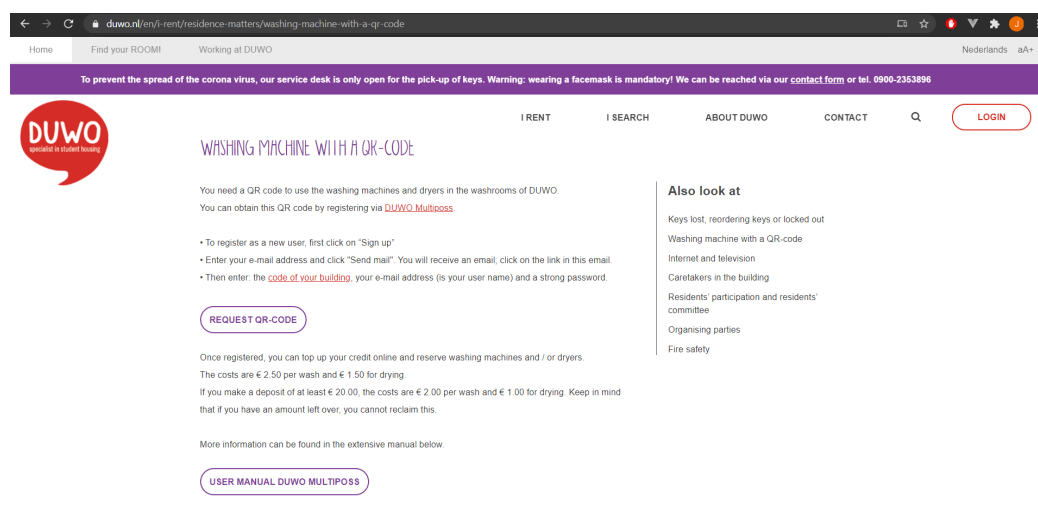
2.2 Postojeća slična rješenja

Istraživanjem tržišta, zaključili smo da postoji nekoliko aplikacija koje se bave navedenim problemima. Jedna od aplikacija koju smo uspjeli pronaći, a da je slična onome što želimo proizvesti, je sljedeća: https://play.google.com/store/apps/details?id=net.celestialdata.ujlaundry&hl=en_US (Slika 2.1). Ta je aplikacija razvijena za praonicu na University of Jamestown-u. Dakako, razlika je u tome što je ovo isključivo mobilna aplikacija (postoje verzije za Android i iOS), a mi bismo napravili responzivnu web aplikaciju lako dostupnu svima s bilo kojeg uređaja koji ima pristup internetskoj mreži. Također, imamo i neke dodatne funkcionalnosti koje su navedene u usporedbi sa sljedećom aplikacijom (posudba košare, recenzije radnika, objava slika, oglasi za posao...).



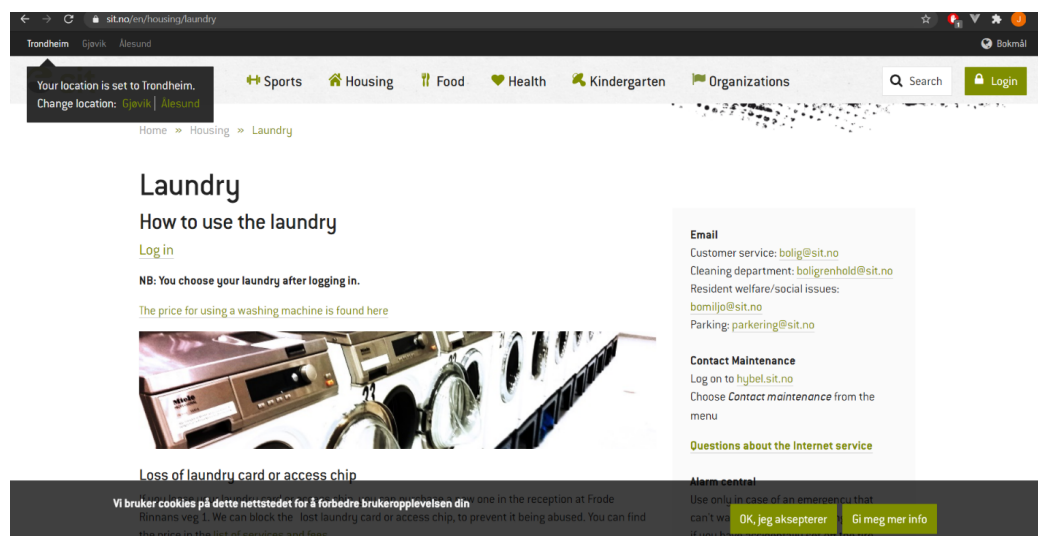
Slika 2.1: Slika aplikacije "UJLAUNDRY"

Također, pronašli smo i sljedeću aplikaciju nizozemskog porijekla: <https://www.duwo.nl/en/irent/residence-matters/washing-machine-with-a-qr-code> (Slika 2.2). Ta je aplikacija po glavnoj funkcionalnosti poprilično slična našoj (web aplikacija koja omogućuje online rezervaciju termina i plaćanje preko interneta. Razlike su u tome što bi naša aplikacija bila prilagođenija studentskim domovima. Nudila bi funkcionalnost označavanja osobe koja je posudila košaru, recenzije radnika u praoni (što druge aplikacije nemaju jer su većinom praonice samoposlužne, dok je u domu zaposlen radnik), mogućnost objavljivanja slika izgubljenih stvari, mogućnost administratora da objavi oglas za posao u praonici, te mogućnost promijene radnog vremena, odnosno vremena pauza na dnevnoj bazi. Također, ta aplikacija nema mogućnost slanja obavijesti na mail (npr. 1h prije početka termina).



Slika 2.2: Slika aplikacije "Washing machine with a QR code"

Treća pak stranica koju smo pronašli dolazi iz Norveške (Slika 2.3). Sličnih je funkcionalnosti kao i prethodno navedena Nizozemska aplikacija, osim što ima dodatnu funkcionalnost, a to je slanje sms poruka za obavijesti (razlika u odnosu na našu je što bi mi slali na mail). Što se razlika tiče, slične su kao i u gore navedenom primjeru, uz razliku u tome što ova aplikacija, kao i naša, implementira slanje obavijesti korisnicima.



Slika 2.3: Slika aplikacije za rezervaciju termina iz Norveške

Postoji i još nekoliko aplikacija sličnoga karaktera, no naveli smo 3 najpopularnije. Većina ostalih aplikacija dijele slične razlike u usporedbi s našom kao i gore navedene pa bi bilo redundantno navoditi ih.

2.3 Potencijalno zainteresirani korisnici

Za ovaj projekt očekujemo da će najveći interes pokazati studentski domovi s obzirom da je cilj projekta olakšavanje poslovanja već postojećih praonica rublja u sklopu domova. No unatoč prvobitnom usmjerenju projekta prema studentskim domovima, smatramo da je ovakvo rješenje također opće primjenjivo i na ostale javne praonice jer primjenom ovakvog rješenja sa mogućnošću online rezervacije termina korisnik ne bi morao riskirati dolazak u punu praonicu te bi mogao izbjeći nepotrebno čekanje. Projekt bi također bilo moguće uvesti u ostale ustanove koje nude mogućnosti pranja rublja kao što su npr. hoteli, kampusi itd.

2.4 Mogućnost prilagodbe rješenja

Kao što smo već naveli gore, projekt se vrlo lako može prilagoditi potrebama krajnjeg korisnika pa tako u vidu imamo i potrebne prilagodbe za javne praonice rublja kao i mogućnost izvoza proizvoda van hrvatske s obzirom da u planu imamo ponuditi i cijelu web aplikaciju na engleskom jeziku. Mogućnost izbora jezika također bi olakšala upotrebu aplikacije stranim studentima koji borave u hrvatskim studentskim domovima.

2.5 Opseg projektnog zadatka (zahtjevi sustava)

Korisnici ove aplikacije su: osoblje u SC-u zaduženo za rad praonice rublja u ulozi administratora, studenti, osobe zaposlene u praonici i neregistrirani korisnici. Neregistrirani korisnici u mogućnosti su samo napraviti svoj korisnički račun, vidjeti oglase za posao i radno vrijeme te nemaju mogućnosti ni za kakvu drugu radnju. Administratori (osoblje u SC-u zaduženo za rad praonice) jedini mogu postaviti oglas za posao u praonici na koji se mogu prijaviti ljudi koji su registrirani u aplikaciji. Oni također mogu mijenjati radno vrijeme praonice te zabilježiti eventualne promjene cijena pranja i sušenja rublja. Ako je potrebno, administrator može i blokirati pristup aplikaciji bilo kojem korisniku. Sljedeća su kategorija korisnika osobe zaposlene u praonici. Te osobe, poput administratora, mogu mijenjati radno vrijeme praonice te pauze. S obzirom da se termini za pranje i sušenje rublja mogu rezervirati svakih 1h vremena (npr. ako praonica radi od 8, onda su mogući termini 8:00, 9:00, 10:00, 11:00...), pauze mogu trajati maksimalno pola sata (između

2 termina) te se nikako ne smije dogoditi da pauza bude kad je vrijeme zamjene rublja. U aplikaciji osoba koja trenutno radi može označiti da je netko posudio košaru za rublje i ako ne vrati zna se kod koga je. Također, ako osoba ne dođe pokupiti svoje rublje, može mu poslati obavijest na mail da je rublje gotovo. Posljednja vrsta korisnika ove aplikacije su studenti. Oni će se moći prijaviti svojim AAI eduHr računom ako nam CARNET dozvoli korištenje AAI eduHr prijave, dok će u suprotnom moći napraviti korisnički račun te će morati čekati da im osobe zaposlene u praonici potvrde račun (nakon slanja zahtjeva za izradu računa morat će doći osobno u praonicu kako bi im ga osoba koja trenutno radi potvrdila). Glavna je funkcionalnost za ovu vrstu korisnika mogućnost rezervacije i otkazivanja termina za pranje i sušenje rublja. Njima je na naslovnoj stranici prikazan kalendar sa slobodnim i popunjenim terminima te lagano mogu odabrati vrijeme i datum za željenu rezervaciju. Prilikom rezervacije moguće je odabrati način plaćanja pranja – putem aplikacije ili na blagajni doma. Također, student može postaviti bilješku kod rezervacije npr. ne mogu doći po rublje zbog obaveza na fakultetu, kasnit ću 2 sata. Termin se smije otkazati do 24 sata prije termina, a ako netko otkaže neposredno prije ili nakon termina za otkaz dobiva negativne bodove (ako se učestalo ponavlja korisnik je dužan platiti kaznu). Studentu će sat vremena prije njegovog termina na mail stići obavijest kako ne bi zaboravio doći ili otkazati termin. Studenti mogu i ocijeniti radnika u praonici nakon pranja rublja i to samo onoga tko je bio u njihovom terminu jedanput po pranju. U ovoj će aplikaciji postojati i zid s objavama na kojem osoba zaposlena u praonici objavljuje slike izgubljenih stvari. Ako je stvar izgubljena dulje od mjesec dana objava se briše i stvar se donira u dobrotvorne svrhe.

2.6 Nadogradnje projektnog zadatka i planovi za budućnost

Moguća nadogradnja bi bila uključivanje više usluga koje se nude studentima u sklopu doma npr. rezervacija termina u teretani (koja ima ograničen broj mjesta i inače, a osobito sada u vrijeme pandemije), pregledi menija u menzi i radnog vremena, kupnja karti za Kino SC, Teatar &TD i slično. Sljedeći korak u razvoju bi mogao biti razvoj mobilne aplikacije jer bi većina studenata sustavu pristupala na taj način. Nakon razvitka mobilne aplikacije, može se pristupiti izgradnji interaktivnog vodiča za bruceše. Vodič bi naše bruceše na licu mjesta mogao provesti kroz način funkcioniranja perilica rublja, ponuditi im savjet koji program služi za

njihovu odjeću, te bi se time smanjila interakcija između zaposlenika i studenta. Preinakama originalne aplikacije mogli bi ju ponuditi i ostalim samoposlužnim praonicama po Hrvatskoj, ali i svijetu. Morali bi istražiti koje su im funkcionalnosti bitne, a koje bi trebalo ukloniti u odnosu na aplikaciju specifičnu za praonice u studentskim domovima.

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Studentski centar Sveučilišta u Zagrebu (naručitelj)
2. Korisnici praonice rublja
3. Zaposlenici praonice rublja
4. Administrator sustava
5. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani/neprijavljeni korisnik (inicijator) može:
 - (a) napraviti novi korisnički račun za koji su mu potrebni adresa e-pošte, lozinka, ime, prezime, JMBAG i broj mobitela
 - (b) vidjeti oglase za posao (osobe koje ne žive u domu i nisu studenti, odnosno nemaju pravo koristiti praonicu, mogu se zaposliti u praonici, prijave za oglas ne idu putem aplikacije, već se molba i životopis šalju na adresu e-pošte navedenu u oglasu)
 - (c) vidjeti podatke o praonici
 - i. radno vrijeme praonice
 - ii. cijene pranja i sušenja
2. Registrirani korisnik (inicijator) može:
 - (a) sve akcije navedene za neregistriranog korisnika
 - (b) prijaviti se u sustav
 - (c) pristupiti kalendaru u kojem se prikazuju slobodni i popunjeni termini
 - (d) rezervirati termin za pranje i sušenje rublja
 - (e) prilikom rezervacije odabrati način plaćanja (putem aplikacije ili na blagajni doma)
 - (f) postaviti bilješku na rezervaciju

- (g) urediti ili otkazati rezervirani termin (do 24 sata prije termina)
- (h) ocijeniti radnika u praonici koji je bio u njihovom terminu pranja
- (i) pristupiti zidu s obavijestima o izgubljenim stvarima
- (j) pregled vlastitog profila i uređivanje podataka

3. Zaposlenik (inicijator) može:

- (a) prijaviti se u sustav
- (b) pregledati vlastiti profil i uređivati podatke
- (c) pristupiti zidu s obavijestima o izgubljenim stvarima
- (d) promijeniti vrijeme pauze
- (e) označiti tko je posudio košaru za rublje
- (f) poslati obavijest na mail osobi kojoj je gotovo rublje
- (g) označiti da se netko nije pojavio u terminu i osloboditi ga
- (h) objaviti fotografiju izgubljenog (zaboravljenog) odjevnog predmeta u praonici
- (i) pregledati termine za radni dan
- (j) potvrditi registraciju korisnika

4. Administrator (inicijator) može:

- (a) sve akcije navedene za zaposlenika
- (b) objaviti, urediti i obrisati oglas za posao
- (c) promijeniti radno vrijeme praonice
- (d) obaviti promjenu cijene pranja
- (e) obaviti promjenu cijene sušenja
- (f) blokirati pristup aplikaciji bilo kojem korisniku
- (g) dodavanje, brisanje i uređivanje zaposlenika i korisnika
- (h) pregled recenzija svih zaposlenika

5. Baza podataka (sudionik) može:

- (a) pohranjuje sve podatke o korisnicima i njihovim ovlastima
- (b) pohranjuje sve podatke o praonici i terminima za rezervaciju
- (c) pohranjuje podatke o plaćanju
- (d) pohranjuje podatke o recenziji radnika
- (e) pohranjuje podatke o košarama za rublje

3.1.1 Obrasci uporabe

UC1 - Registracija

- **Glavni sudionik:** Neregistrirani korisnik
- **Cilj:** Stvoriti korisnički račun za pristup sustavu
- **Sudionici:** Baza podataka, Zaposlenik
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju za registraciju
 2. Korisnik unosi potrebne korisničke podatke
 3. Korisnik dobiva obavijest o slanju registracije na potvrdu
 4. Korisnik odlazi osobno u praonicu s potvrdom da stanuje u domu
 5. Radnik u praonici potvrđuje prijavu nakon uvjeravanja da osoba stanuje u domu
 6. Korisnik prima obavijest o uspješnoj registraciji
- **Opis mogućih odstupanja:**
 - 2.a Odabir već zauzete ili nepostojeće adrese e-pošte te neispravno popunjavanje obrasca
 1. Sustav obavještava korisnika o neuspjeloj registraciji i vraća ga na natrag stranicu za registraciju korisnika
 2. Korisnik mijenja potrebne podatke te se ponovno pokušava registrirati ili odustaje od registracije
 - 4.a Korisnik zaboravi odnijeti potvrdu o stanovanju u studentskom domu u praonicu
 1. Sustav obavještava korisnika da odnese potvrdu u praonicu
 2. Nakon određenog vremena pokušaj registracije briše se iz sustava
 - 5.a Radnik u praonici nije potvrdio registraciju korisnika
 1. Korisnik ponovno odlazi u praonicu s potvrdom
 - 5.a Radnik u praonici potvrdio je ili odbio zahtjev za registracijom krivog korisnika
 1. Radnik obavještava administratora o tome kojeg je korisnika zabunom potvrdio ili odbio, te administrator promijeni podatke u bazi podataka

UC2 - Prijava

- **Glavni sudionik:** Registrirani korisnik, Zaposlenik, Administrator

- **Cilj:** Prijava za pristup sustavu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran u sustavu
- **Opis osnovnog tijeka:**
 1. Korisnik ispunjava potrebne podatke za prijavu (e-mail i lozinku)
 2. Korisniku se omogućava pristup značajkama specifičnim za njegovu razinu pristupa
- **Opis mogućih odstupanja:**
 - 2.a Unos krive kombinacije e-mail-a i lozinke
 1. Sustav korisnika vraća na početnu stranicu i dojavljuje da prijava nije uspjela
 2. Korisnik ponovno upisuje podatke ili odustaje od prijave

UC3 - Pregled vlastitog profila i uređivanje podataka

- **Glavni sudionik:** Registrirani korisnik, Zaposlenik, Administrator
- **Cilj:** Omogućuje pregledavanje i modificiranje osobnih podataka
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju za prikaz vlastitog profila
 2. Korisniku se prikazuju njegovi osobni podaci
 3. Klikom na gumb za izmjenu korisniku se otvara obrazac za promjenu
 4. Korisnik ima opciju obrisati korisnički račun
- **Opis mogućih odstupanja:**
 - 2.a Korisnik neispravno popunjava obrazac za promjenu
 1. Korisniku se ispisuje upozorenje
 2. Promjena ostaje nezabilježena u sustavu

UC4 - Rezervacija termina za pranje veša

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Rezerviranje željenog termina u praonici
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju za rezervaciju termina
 2. Korisniku se prikazuje kalendar sa slobodnim terminima

3. Korisnik odabire slobodan termin
 4. Korisnik odabire način plaćanja
 5. Korisniku se nudi opcija stavljanja bilješke na rezervaciju
 6. Rezervacija se registrira u sustavu
- **Opis mogućih odstupanja:**
 - 2.a Nema slobodnih termina
 1. Korisniku se ispisuje poruka da nema slobodnih termina

UC4.1 - Plaćanje rezervacije

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Plaćanje pranja ili sušenja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju plaćanja
 2. Izvršava se plaćanje u sustavu ili na blagajni
- **Opis mogućih odstupanja:**
 - 2.a Korisnik nema sredstava na računu
 1. Korisniku se ispisuje upozorenje da nema sredstava na računu
 2. Korisniku se nudi plaćanje na blagajni

UC5 - Pristup kalendaru s terminima

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Pregledati slobodne termine
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju za prikaz kalendara
 2. Korisniku se prikazuje kalendar sa terminima

UC6 - Uređivanje rezerviranog termina

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Urediti postojeću rezervaciju termina
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav i ima rezervirani termin
- **Opis osnovnog tijeka:**

1. Korisnik u kalendaru odabire rezervirani termin
 2. Korisniku se nudi izbor pomicanja ili otkazivanja termina te opcija davanja bilješke
 3. Korisnik odabire željenu akciju
- **Opis mogućih odstupanja:**
 - 2.a Preostalo je manje od 24 sata do termina, a korisnik želi preseliti/otkazati termin
 1. Korisniku se pojavljuje poruka kako je termin nemoguće preseliti/otkazati termin
 2. Korisnika se vraća na kalendar

UC7 - Ocjenjivanje radnika u terminu

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Ocijeniti radnika u terminu korištenja praonice
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisniku se nakon termina pranja nudi opcija ocjenjivanja
 2. Korisnik unosi ocjenu
 3. Ocjena se bilježi u sustavu

UC8 - Pregled zida s obavijestima o izgubljenim stvarima

- **Glavni sudionik:** Registrirani korisnik, Zaposlenik, Administrator
- **Cilj:** Pristup zidu s obavijestima o izgubljenim stvarima
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju zida s obavijestima o izgubljenim stvarima
 2. Korisniku se prikazuje zid s obavijestima o izgubljenim stvarima

UC9 - Pregled podataka o praonici

- **Glavni sudionik:** Registrirani korisnik, Neregistrirani korisnik
- **Cilj:** Prikaz podataka o praonici
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**

1. Korisnik odabire opciju za prikaz podataka o praonici
2. Korisniku se prikazuju podaci o praonici

UC10 - Pregled oglasa za posao

- **Glavni sudionik:** Registrirani korisnik, Neregistrirani korisnik
- **Cilj:** Prikaz oglasa za posao
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju za prikaz oglasa za posao
 2. Korisniku se prikazuju oglasi

UC11 - Objava oglasa za posao

- **Glavni sudionik:** Administrator
- **Cilj:** Objaviti oglas za studentski posao
- **Sudionici:** Baza podataka
- **Preduvjet:** Postoji otvorena pozicija u praonici i korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju za stvaranje novog oglasa
 2. Administrator popunjava tekst i podatke oglasa
 3. Oglas se objavljuje odabirom opcije za objavljivanje oglasa.
- **Opis mogućih odstupanja:**
 - 2.a Pogrešno uneseni podaci o natječaju
 1. Odabire opciju za uređivanje oglasa nakon objave
 2. Administrator ispravlja pogrešku
 3. Odabire opciju za spremanje promjena

UC12 - Promjena radnog vremena

- **Glavni sudionik:** Administrator
- **Cilj:** Promijeniti radno vrijeme praonice
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju promjene radnog vremena
 2. Prikaže se prozor za odabir novog radnog vremena

3. Administrator unosi novo radno vrijeme
4. Administrator odabire opciju za potvrdu novog radnog vremena

UC13 - Promjena cijene pranja

- **Glavni sudionik:** Administrator
- **Cilj:** Promijeniti cijenu jednog pranja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju promjene cijene pranja
 2. Prikaže se modal za unos nove cijene pranja
 3. Administrator unosi novu cijenu pranja
 4. Administrator odabire opciju za potvrdu nove cijene pranja

UC14 - Promjena cijene sušenja

- **Glavni sudionik:** Administrator
- **Cilj:** Promijeniti cijenu jednog sušenja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju promjene cijene sušenja
 2. Prikaže se prozor za unos nove cijene sušenja
 3. Administrator unosi novu cijenu sušenja
 4. Administrator odabire opciju za potvrdu nove cijene sušenja

UC15 - Zabrana pristupa korisniku

- **Glavni sudionik:** Administrator
- **Cilj:** Zabraniti pristup korisniku uslugama praonice
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju zabrane pristupa korisnicima
 2. Prikaže se lista svih ispravno registriranih korisnika
 3. Administrator odabire korisnika ili više njih
 4. Administrator odabire opciju za potvrdu zabrane pristupa
- **Opis mogućih odstupanja:**

- 2.a Ne postoje registrirani korisnici
 - 1. Prikaže se odgovarajuća poruka

UC16 - Uređivanje oglasa za posao

- **Glavni sudionik:** Administrator
- **Cilj:** Promijeniti podatke unesene u oglasu za posao
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora, te postoje objavljeni oglasi
- **Opis osnovnog tijeka:**
 - 1. Administrator odabire opciju pregleda oglasa
 - 2. Prikaže se lista svih objavljenih oglasa
 - 3. Administrator odabire oglas koji želi urediti
 - 4. Administrator unosi izmjene
 - 5. Administrator odabire opciju za potvrdu izmjena
- **Opis mogućih odstupanja:**
 - 2.a Nema objavljenih oglasa
 - 1. Prikaže se odgovarajuća poruka

UC17 - Brisanje oglasa za posao

- **Glavni sudionik:** Administrator
- **Cilj:** Obrisati oglas za posao
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora, te postoje objavljeni oglasi
- **Opis osnovnog tijeka:**
 - 1. Administrator odabire opciju pregleda oglasa
 - 2. Prikaže se lista svih objavljenih oglasa
 - 3. Administrator odabire oglas koji želi obrisati
 - 4. Administrator odabire opciju za brisanje
 - 5. Administrator odabire opciju za potvrdu izmjena
- **Opis mogućih odstupanja:**
 - 2.a Nema objavljenih oglasa
 - 1. Prikaže se odgovarajuća poruka

UC18 - Dodavanje novih korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Dodati novog korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju pregleda korisnika
 2. Prikaže se lista svih registriranih korisnika i zaposlenika
 3. Administrator odabire opciju za stvaranje novog korisnika
 4. Administrator unosi podatke o novom korisniku
 5. Administrator odabire opciju za potvrdu izmjena

UC19 - Brisanje korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Obrisati korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora, te postoje registrirani korisnici
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju pregleda korisnika
 2. Prikaže se lista svih registriranih korisnika i zaposlenika
 3. Administrator odabire korisnika kojeg želi obrisati
 4. Administrator odabire opciju za brisanje korisnika
 5. Administrator odabire opciju za potvrdu izmjena
- **Opis mogućih odstupanja:**
 - 2.a Ne postoje registrirani korisnici
 1. Prikaže se odgovarajuća poruka

UC20 - Uređivanje korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Urediti podatke o korisniku
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora te postoje registrirani korisnici
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju pregleda korisnika
 2. Prikaže se lista svih registriranih korisnika i zaposlenika

3. Administrator odabire korisnika čije podatke želi urediti
 4. Administrator unosi izmjene
 5. Administrator odabire opciju za potvrdu izmjena
- **Opis mogućih odstupanja:**
 - 2.a Ne postoje registrirani korisnici
 1. Prikaže se odgovarajuća poruka

UC21 - Pregled recenzija svih zaposlenika

- **Glavni sudionik:** Administrator
- **Cilj:** Pregled recenzije zaposlenika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je registriran i dodijeljena su mu prava administratora te postoje recenzije zaposlenika
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju pregleda zaposlenika
 2. Prikaže se lista svih zaposlenika
 3. Administrator odabire zaposlenika čije recenzije želi pogledati
 4. Administrator odabire opciju za pregled recenzija
- **Opis mogućih odstupanja:**
 - 2.a Ne postoje registrirani korisnici
 1. Prikaže se odgovarajuća poruka

UC22 - Oznaka posudbe košare za rublje

- **Glavni sudionik:** Zaposlenik
- **Cilj:** Označiti osobu koja je posudila košaru za rublje
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Zaposlenik odabire korisnika koji je posudio košaru
 2. Zaposlenik odabire opciju za posudbu
- **Opis mogućih odstupanja:**
 - 2.a Korisnik već ima posuđenu košaru
 1. Onemogućuje mu se posudba nove košare dok ne vrati staru
 - 2.b Korisnik ne vrati posuđenu košaru
 1. Korisnik dobiva opomenu

UC23 - Slanje obavijesti na mail

- **Glavni sudionik:** Zaposlenik, Administrator
- **Cilj:** Obavijestiti korisnika kojem je rublje gotovo
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Zaposlenik odabire korisnika kojem je rublje gotovo
 2. Zaposlenik odabire njegovu mail adresu
 3. Zaposlenik šalje obavijest
 4. Korisnik prima obavijest

UC24 - Promjena radnog vremena

- **Glavni sudionik:** Zaposlenik, Administrator
- **Cilj:** Obavijestiti korisnike o promjeni radnog vremena praonice
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Zaposlenik odabire opciju "Radno vrijeme"
 2. Zaposlenik postavlja novo radno vrijeme
 3. Korisniku to vrijeme postaje vidljivo u aplikaciji

UC25 - Promjena vremena pauze

- **Glavni sudionik:** Zaposlenik, Administrator
- **Cilj:** Obavijestiti korisnike o promjeni vremena pauze
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Zaposlenik odabire opciju "Vrijeme pauze"
 2. Zaposlenik postavlja novo vrijeme pauze
 3. Korisniku to vrijeme postaje vidljivo u aplikaciji
- **Opis mogućih odstupanja:**
 - 2.a Pokušava se postaviti vrijeme pauze koje nije dozvoljeno (u vrijeme zamjene rublja)
 1. Sustav obavještava zaposlenika da to vrijeme pauze nije dozvoljeno i nudi mu opciju mijenjanja termina pauze

UC26 - Oznaka da se netko nije pojavio u terminu i njegovo oslobođenje

- **Glavni sudionik:** Zaposlenik, Administrator
- **Cilj:** Obavijestiti korisnike o oslobodenju termina i zabilježba osobe koja je propustila svoj termin
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Zaposlenik odabire korisnika koji se nije pojavio u svom terminu
 2. Zaposlenik dodjeljuje negativne bodove korisniku
 3. Zaposlenik njegov termin označava oslobodenim
 4. Termin postaje vidljiv u aplikaciji i spreman za ponovnu rezervaciju
- **Opis mogućih odstupanja:**
 - 1.a Korisnik svoj termin otkaže 24 sata prije samog termina
 1. Zaposlenik mu ne dodjeljuje negativne bodove
 - 1.b Korisnik ima previše negativnih bodova
 1. Sustav mu onemogućava daljnju rezervaciju termina i obavještava ga o naplati kazne

UC27 - Objava slike izgubljenog odjevnog predmeta u praonici

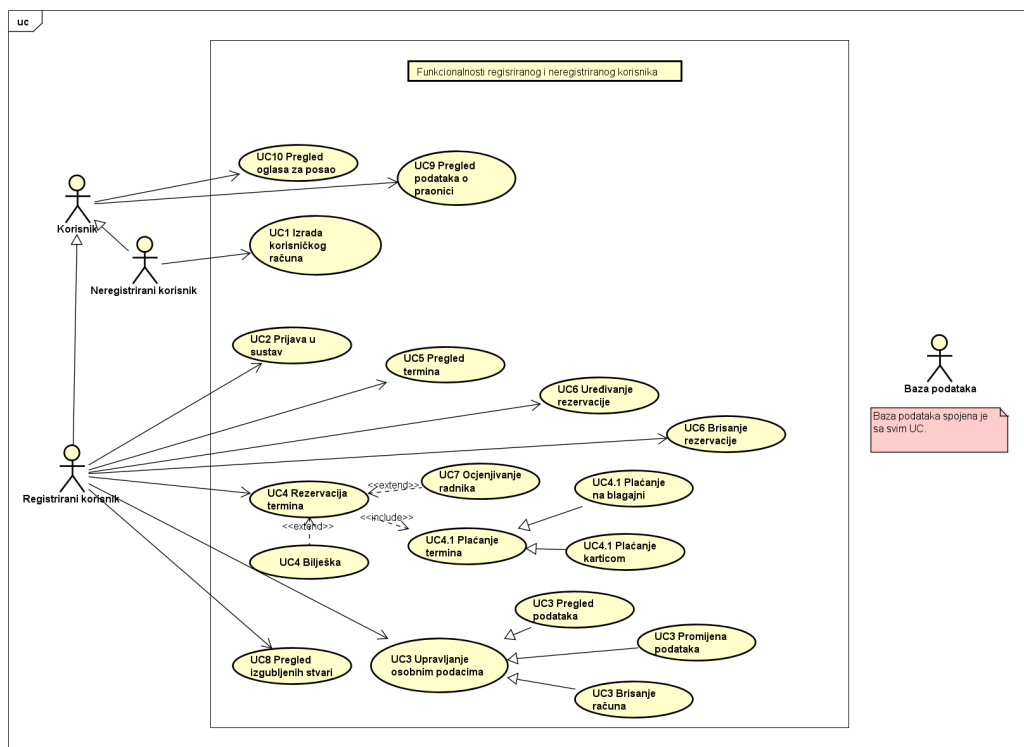
- **Glavni sudionik:** Zaposlenik, Administrator
- **Cilj:** Obavijestiti korisnike o izgubljenom predmetu u praonici
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Zaposlenik fotografira odjevni predmet
 2. Zaposlenik je učitava na zid s objavama
 3. Fotografija postaje vidljiva u aplikaciji

UC28 - Pregled termina za radni dan

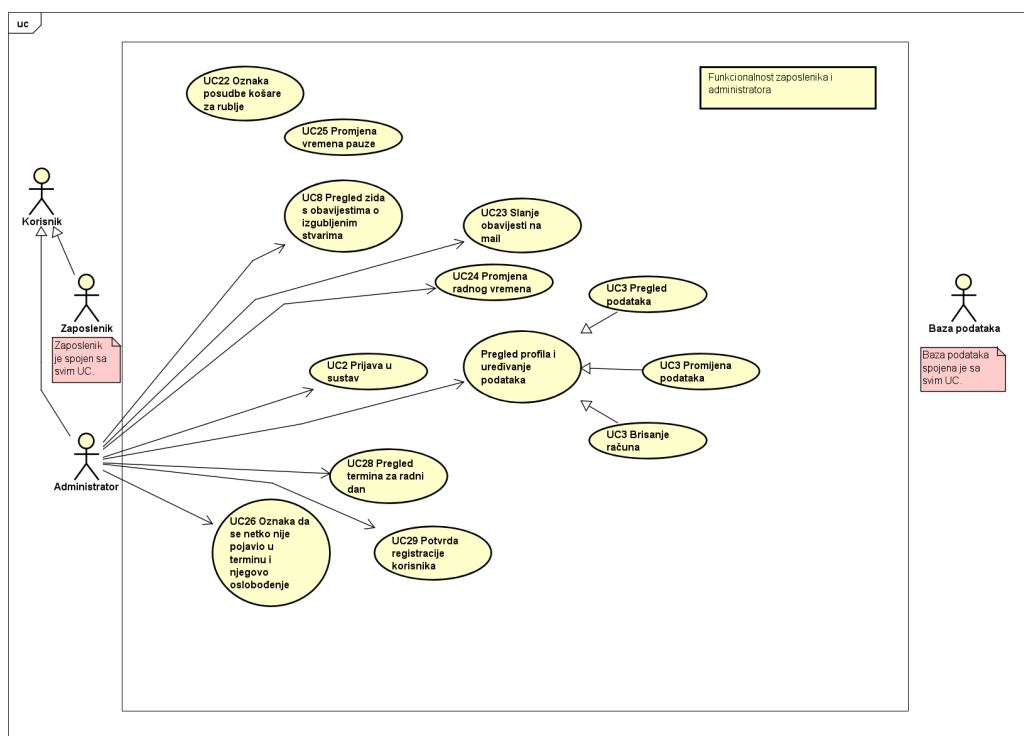
- **Glavni sudionik:** Zaposlenik, Administrator
- **Cilj:** Pregledati termine za taj radni dan
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Zaposlenik odabire opciju "Termini"
 2. Zaposleniku se prikazuju svi termini za taj dan
 3. Korisniku to vrijeme postaje vidljivo u aplikaciji

UC29 - Potvrda registracije korisnika

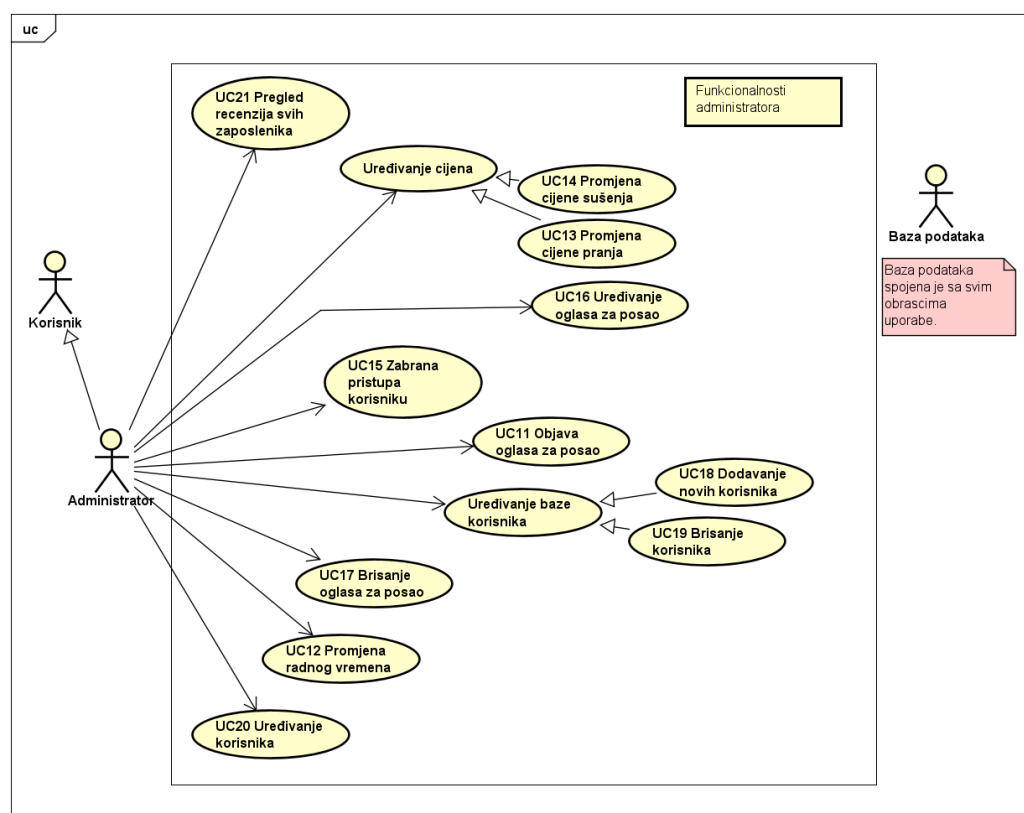
- **Glavni sudionik:** Zaposlenik, Administrator
- **Cilj:** Potvrditi registraciju korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Pregled potvrde da korisnik stanuje u domu
 2. Zaposlenik odabire opciju "Potvrdi registraciju"
 3. Korisnik prima obavijest o uspješnoj registraciji
- **Opis mogućih odstupanja:**
 - 1.a Korisnik ima nevažeću potvrdu o stanovanju u studentskom domu
 1. Zaposlenik mu ne potvrđuje registraciju

Dijagrami obrazaca uporabe

Slika 3.1: Slika dijagrama obrasca uporabe registriranih i neregistriranih korisnika



Slika 3.2: Slika dijagrama obrasca uporabe zaposlenika i administratora

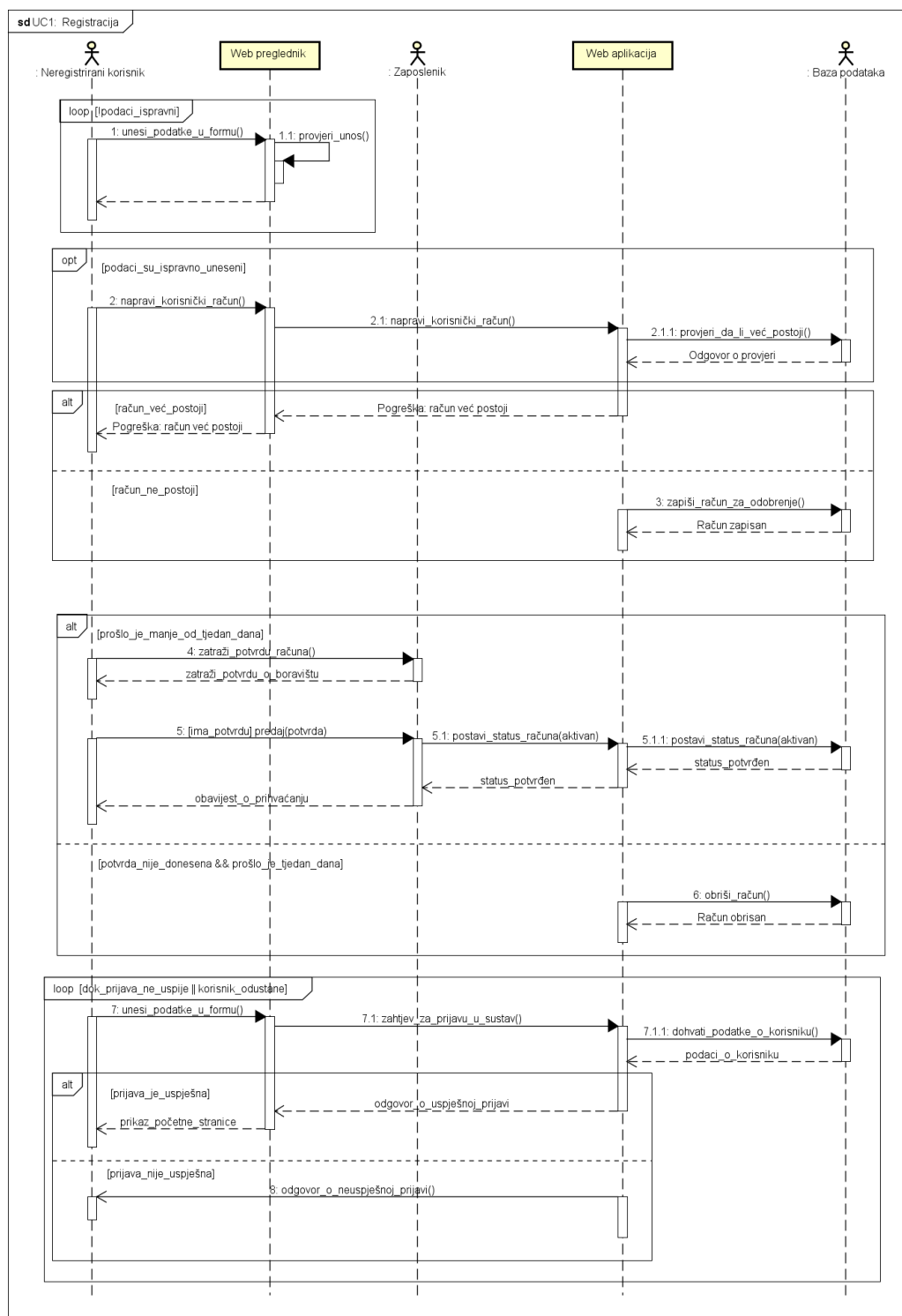


Slika 3.3: Slika dijagrama obrasca uporabe administratora

3.1.2 Sekvencijski dijagrami

Obrazac uporabe UC1 i UC2 - Registracija i Prijava korisnika

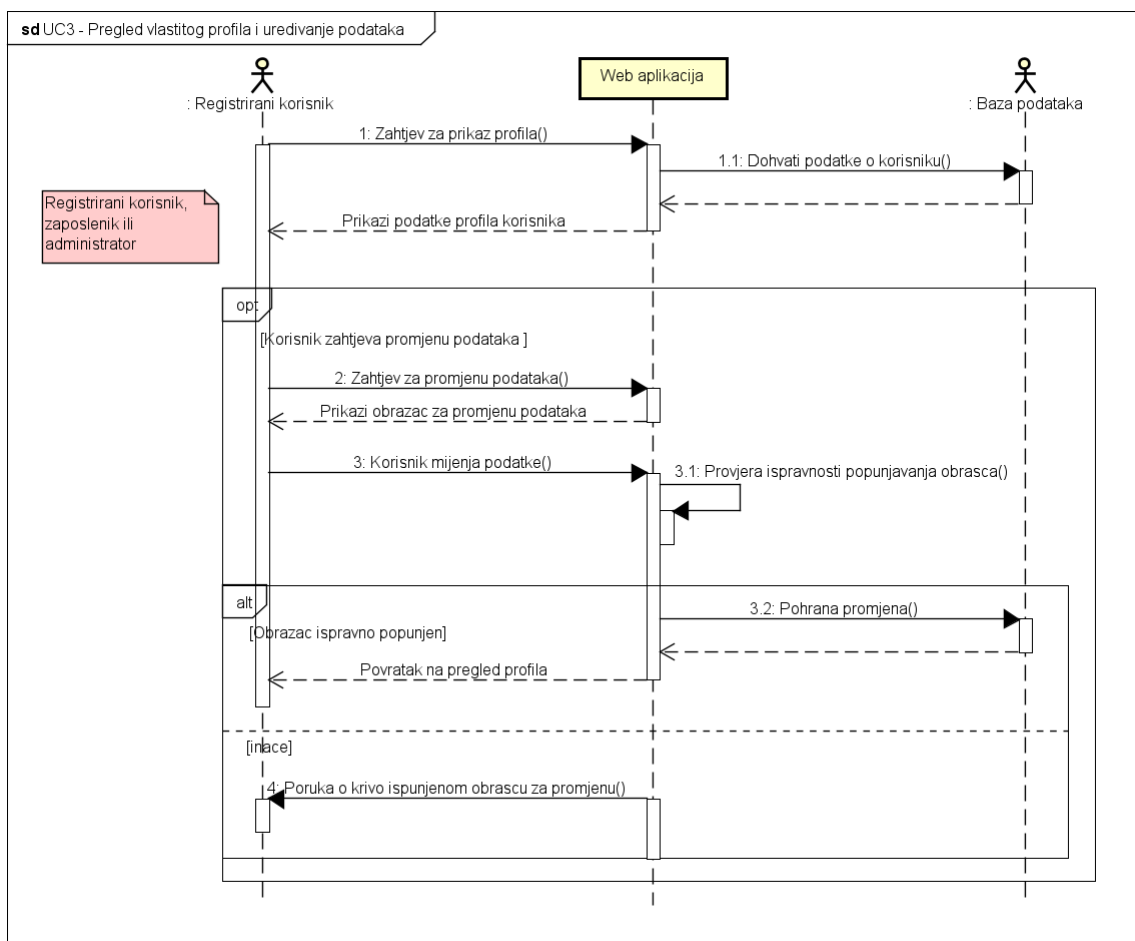
Neregistrirani korisnik unosi podatke u formu. Web preglednik vrši provjeru podataka i vraća povratnu informaciju korisniku sve dok ne unese potrebne podatke ili ne odustane. Kada se unesu točni podaci šalje se zahtjev za stvaranje računa prema web aplikaciji. Web aplikacija šalje upit prema bazi podataka kojim provjerava postoji li korisnik od prije. Ako račun već postoji korisnika se obavještava o pogrešci, inače račun se zapisuje u bazu podataka. Tada korisnik (student) treba otići u praonicu sa potvrdom o boravištu. Zaposlenik nakon provjere potvrde aktivira korisnikov račun. Nakon toga korisnik se može prijaviti u sustav. Prijavu započinje unosom podataka u formu za prijavu. Web aplikacija dohvaća podatke o korisniku iz baze podataka. Provjerom utvrđuje ispravnost unesenih podataka. Ako su ispravni korisniku se prikazuje početna stranica, inače se dojavljuje pogreška.



Slika 3.4: Sekvencijski dijagram za UC1 i UC2

Obrazac uporabe UC3 - Pregled vlastitog profila i uređivanje podataka

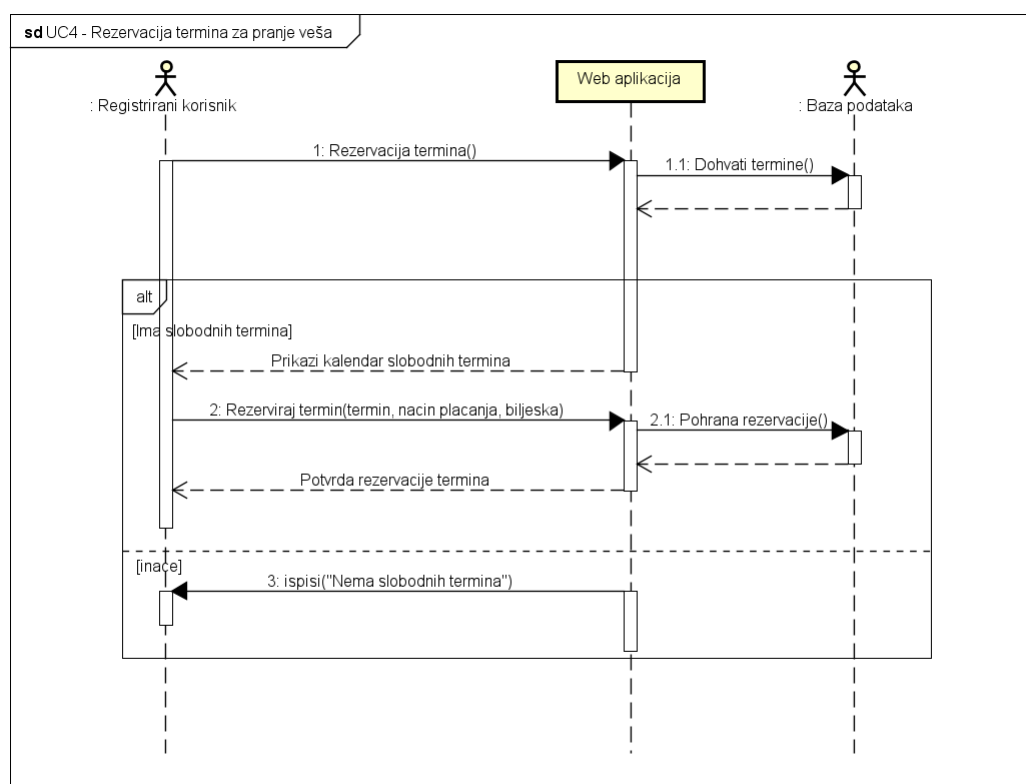
Registrirani korisnik, zaposlenik ili administrator odabire opciju za prikaz vlastitog profila. Web aplikacija dohvaća podatke o korisniku iz baze podataka te ih prikazuje na stranici profila. Korisnik može urediti svoje podatke. Klikom na gumb za izmjenu podataka otvara se obrazac. Korisnik popunjava obrazac, a web aplikacija provjerava ispravnost popunjenih podataka. Ukoliko su podaci krivo popunjeni, web aplikacija ispisuje upozorenje, a promjena ostaje nezabilježena. Ispravno popunjen obrazac se pohranjuje u bazu podataka.



Slika 3.5: Sekvencijski dijagram za UC3

Obrazac uporabe UC4 - Rezervacija termina za pranje rublja

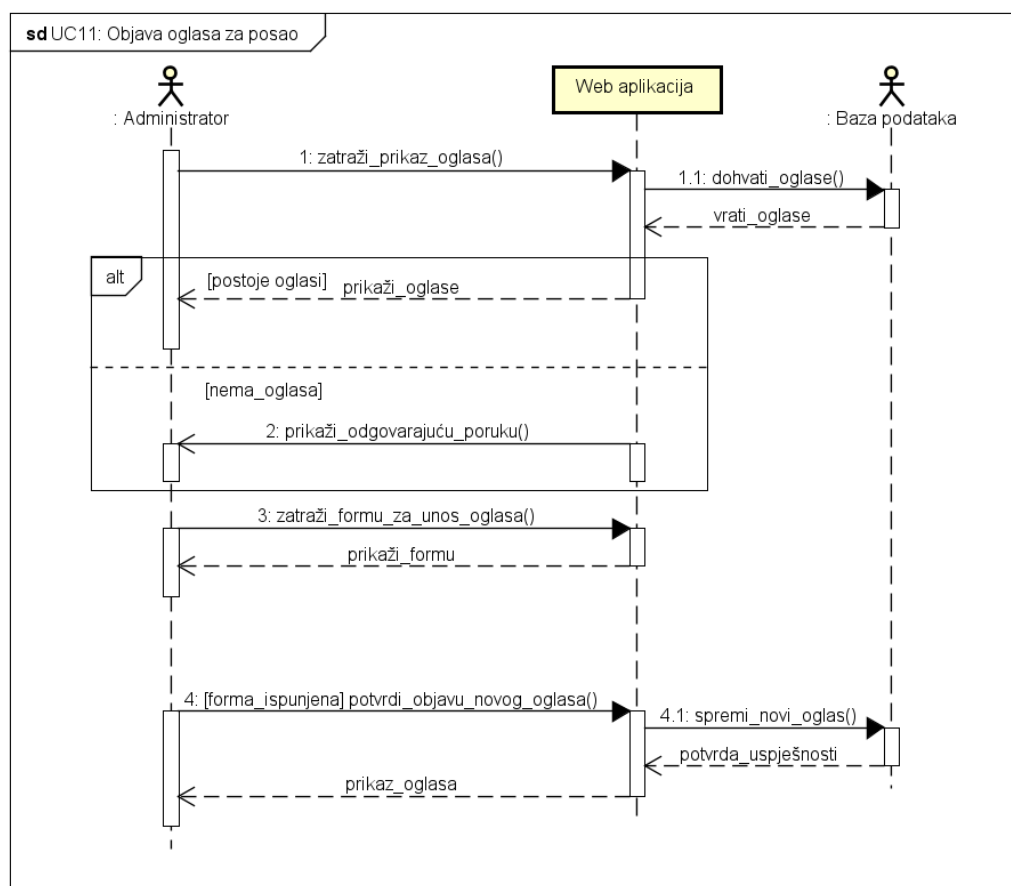
Registrirani korisnik odabire opciju za rezervaciju termina. Poslužitelj dohvaća podatke o terminima iz baze podataka i prikazuje kalendar sa slobodnim terminima registriranom korisniku. Korisnik odabire jedan od slobodnih termina, način plaćanja (online ili uživo) te može ostaviti bilješku. Podaci o terminu se pohranjuju u bazu podataka. Korisnik dobiva potvrdu o uspješnoj rezervaciji termina. Ukoliko nema slobodnih termina, korisniku se ispisiuje poruka "Nema slobodnih termina".



Slika 3.6: Sekvencijski dijagram za UC4

Obrazac uporabe UC11 - Objava oglasa za posao

Administrator šalje zahtjev za prikaz svih oglasa. Web aplikacija dohvaća oglase iz baze podataka. Oglasi se prikazuju administratoru. Administrator odabire opciju za prikaz forme za unos novog oglasa. Nakon što je ispunjena forma, oglas se sprema u bazu podataka. Administratoru se prikazuje stranica sa oglasima.



Slika 3.7: Sekvencijski dijagram za UC11

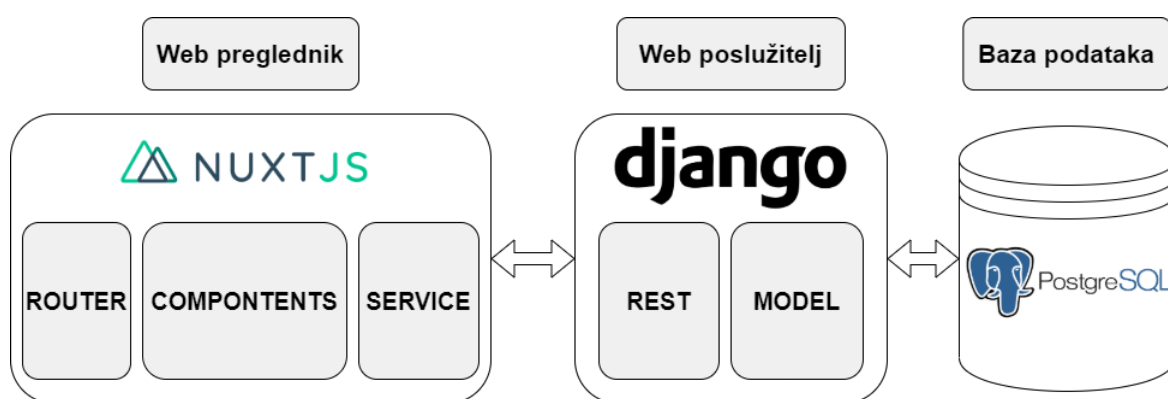
3.2 Ostali zahtjevi

- Treba biti omogućen rad više korisnika u stvarnom vremenu
- Sustav i korisničko sučelje moraju imati podršku za sve znakove hrvatske abecede
- Dizajn korisničkog sučelja treba biti responzivan, odnosno takav da se može koristiti na svakom uređaju bez obzira na veličinu ekrana
- Potrebno je omogućiti korisniku zamjenu ili uređenje odabranog termina 24 sata prije početka
- Sustav će periodički svakih tjedan dana brisati nepotvrđene rezervacije iz baze podataka
- Radno vrijeme moći će se mijenjati za sve dane koji su dva tjedna nakon tekućeg dana
- Morat će postojati ograničenje rezervacije dva termina dnevno za svakog registriranog korisnika
- Neispravno korištenje korisničkog sučelja mora dati povratnu informaciju korisniku u stvarnom vremenu
- Sustav treba omogućiti korištenje kolačića (engl. *cookies*) u svrhu pohrane privremeno korištenih podataka
- Za sve novčane transakcije koristi HRK kao valutu
- Sustav treba biti dovoljno jednostavan kako bi ga mogla koristiti osoba bez opširnih uputa
- Lozinke korisnika u bazi podataka trebaju biti zaštićene PBKDF2 (*Password-Based Key Derivation Function 2*) kriptografskim algoritmom kako bi se smanjila mogućnost narušavanja sigurnosti sustava
- Pristup sustavu omogućen je korištenjem protokola HTTP/2

4. Arhitektura i dizajn sustava

Arhitektura se može podijeliti na sljedeće sustave:

- Web preglednik
- Web poslužitelj
- Baza podataka



Slika 4.1: Arhitektura sustava

Web preglednik je program čije je svrha omogućiti pregled web-stranice te njegov multimedijalnog sadržaja. Osim samog pregleda, web preglednik mora moći prevesti kôd stoga jer je on ujedno i kompajler. U našem slučaju web preglednik će prevoditi kôd radnog okvira (engl. *framework*) Nuxt.js koje je proširenje Vue.js frameworka.

Web poslužitelj je sustav koji treba omogućiti odvijanje komunikacije web preglednika i baze podataka. Njegova zadaća je obrađivati zahtjeve koje zahtjeva web preglednik, po potrebi komunicirati s bazom podataka te vratiti rezultat obrade. Rezultat obrade biti se će SSR (engl. *Server Side Rendered*) - statički generirana stranica. Komunikacija s poslužiteljem odvija se putem protokola HTTP (engl. *Hyper Text Transfer Protocol*). U našem slučaju za izradu web poslužitelja koristimo Python framework Django koji je besplatan i *open source*.

Baza podataka je sustav za trajnu pohranu podataka. U našem slučaju koristimo relacijsku bazu podataka PostgreSQL.

Zadana arhitektura temelji se na prikazu statički generirane stranice dobivene s web poslužitelja (Django servera). Prema zahtjevu web klijenta Nuxt.js generira zahtjev na API web poslužitelja te dobiva odgovor u obliku generirane statičke stranice. Web poslužitelj prema potrebi šalje zahtjev za dohvat podataka iz baze podataka kako bi mogao stranicu popuniti s relevantnim podacima i multimedijom.

Django nije tradicionalni MVC (Model-View-Controller) framework, ali se ponaša vrlo slično MVC koncept modelu. MVC koncept model sastoji se od sljedećih komponenata:

- Model - Dinamičke strukture podataka koje se popunjavaju iz baze podataka za svrhu prikaza ili se popunjavaju iz korisničkog sučelja u svrhu pohrane.
- View - Korisničko sučelje preko kojega korisnik podatke prenosi u Model.
- Controller - Sučelje između Modela i View-a. Manipulira podacima kako bi ih Model mogao obraditi ili manipulira podatke da ih View može prikazati.

Django razvojni tim preferira koristi koncept MTV (*Model-Template-View*) modela. Razlika između MVC modela i MTV modela je što se komponenta View naziva Template, a komponenta Controller se naziva View.

4.1 Baza podataka

Za potrebe našeg sustava koristit ćemo relacijsku bazu podataka PostgreSQL. Baza se sastoji od relacija (tablica) i njihovih atributa. Bazu podataka koristimo za pohranu informacija o korisnicima, njihovim rezervacijama i podacima o praonici. Baza podataka ove aplikacije sastoji se od sljedećih entiteta:

- Machine
- Appointment
- User
- Post
- Review
- Laundry
- Card

Osim naših tablica, Django sam stvara nove tablice poput permissiona, sessiona i logova.

4.1.1 Opis tablica

Machine Ovaj entitet sadrži sve važne informacije vezane za uređaj u praonici. Sadrži attribute: id uređaja i type. Ovaj entitet u vezi je *One-to-Many* s entitetom Appointment preko id-a uređaja.

Machine		
id	INT	jedinstveni brojčani identifikator
type	BOOLEAN	vrsta uređaja (perilica ili sušilica)

Appointment Ovaj entitet sadrži sve važne informacije vezane za rezervirani termin u praonici. Sadrži attribute: id termina, start, machine_id, price, opcionalni note, paid, basket_taken, user_id, employee_id i missed. Ovaj entitet u vezi je *Many-to-One* s entitetom Machine preko id-a uređaja, vezi *Many-to-One* s entitetom User preko id-a korisnika i id-a zaposlenika i vezi *One-to-One* s entitetom Review preko id-a termina.

Appointment		
id	INT	jedinstveni brojčani identifikator

Appointment		
start	TIMESTAMP	datum rezerviranog termina u praonici i vrijeme početka
machine_id (FK)	INT	jedinstveni identifikator uređaja kojeg rezerviramo u terminu (machine.id)
price	FLOAT	cijena usluge koju korisnik plaća (pranje ili sušenje)
note	VARCHAR	opcionalna bilješka zaposleniku
paid	BOOLEAN	korisnik bira hoće li platiti termin online ili uživo
basket_taken	BOOLEAN	korisnik može posuditi jednu košaru po terminu iz praonice
user_id (FK)	INT	jedinstveni identifikator korisnika koji je rezervirao termin (user.id)
employee_id (FK)	INT	jedinstveni identifikator zaposlenika koji radi za vrijeme rezerviranog termina (user.id)
missed	BOOLEAN	zaposlenici vode evidenciju propuštenih termina

User Ovaj entitet sadrži sve važne informacije vezane za različite korisnike u praonici. Sadrži attribute: id korisnika, username, password, first_name, last_name, jedinstveni JMBAG, jedinstveni email, is_active, is_staff, card_id, negative_points, date_joined, last_login, is_superuser i baskets. Ovaj entitet u vezi je *One-to-Many* s entitetom Appointment preko id-a korisnika, u vezi *One-to-Many* s entitetom Post preko id-a korisnika, u vezi *One-to-Many* s entitetom Review preko id-a korisnika, u vezi *One-to-One* s entitetom Card preko id-a kreditne kartice i u vezi *Many-to-Many* s entitetom Laundry preko id-a korisnika

User		
id	INT	jedinstveni brojčani identifikator korisnika
username	VARCHAR	korisničko ime
password	VARCHAR	lozinka korisničkog računa
first_name	VARCHAR	ime korisnika
last_name	VARCHAR	prezime korisnika
JMBAG	VARCHAR	JMBAG korisnika
email	VARCHAR	jedinstveni email korisnika

User		
is_active	BOOLEAN	svi korisnici čije registracije su potvrđene od zaposlenika i koji nisu blokirani
is_staff	BOOLEAN	oznaka je li korisnik i zaposlenik
is_superuser	BOOLEAN	pokazuje je li korisnik ujedno i administrator
card_id (FK)	INT	jedinstveni identifikator kreditne kartice (card.id)
negative_points	INT	broj negativnih bodova korisnika
date_joined	DATE	datum registracije korisnika
last_login	TIMESTAMP	vrijeme zadnje prijave u sustav
baskets	INT	broj košara koje je su trenutno kod korisnika

Post Ovaj entitet sadrži sve važne informacije vezane za objavu. Sadrži attribute: id objave, photo, text, date, type i employee_id. Ovaj entitet u vezi je *Many-to-One* s entitetom User preko id-a korisnika.

Post		
id	INT	jedinstveni brojčani identifikator objave
photo	BYTEA	opcionalna slika priložena uz objavu
text	VARCHAR	tekst objave
date	DATE	datum objavljivanja objave
type	BOOLEAN	ako je vrijednost zastavice 1, objava se prikazuje na "LostAndFound" stranici
posted_by_id (FK)	INT	id korisnika koji je napravio objavu (user.id)

Review Ovaj entitet sadrži sve važne informacije vezane za recenzije. Sadrži attribute: id recenzije, user_id, employee_id, text, grade i appointment_id. Ovaj entitet u vezi je *Many-to-One* s entitetom User preko id-a zaposlenika i id-a korisnika i u vezi *One-to-One* s entitetom Appointment preko id-a termina.

Review		
id	INT	jedinstveni brojčani identifikator recenzije
user_id (FK)	INT	id korisnika koji je ostavio recenziju (user.id)

Review		
employee_id (FK)	INT	id recenziranog zaposlenika (user.id)
text	VARCHAR	tekst recenzije
grade	INT	ocjena zaposlenika od 1 do 5
appointment_id (FK)	INT	id termina koji se recenzira (appointment.id)

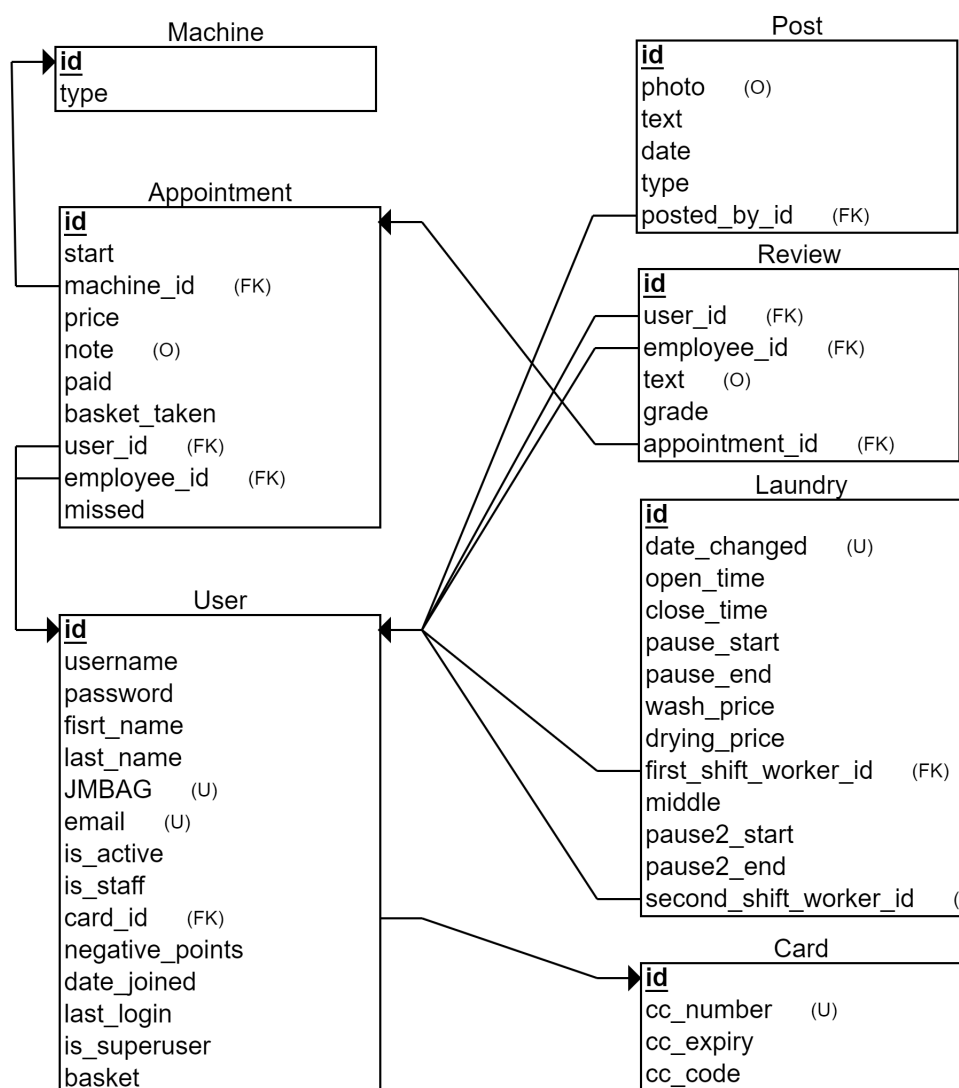
Laundry Ovaj entitet sadrži sve važne informacije vezane za radni dan praonice. Sadrži attribute: id radnog dana, date_changed, open_time, close_time, pause_start, pause_end, wash_price, drying_price, first_shift_worker_id, middle, pause2_start, pause2_end i second_shift_worker_id. Ovaj entitet u vezi je *Many-to-Many* s entitetom User preko id-a radnika prve i druge smjene.

Laundry		
id	INT	jedinstveni brožčani identifikator radnog dana praonice
date_changed	TIMESTAMP	vrijeme promjene radnog vremena
open_time	TIME	početak radnog vremena
close_time	TIME	kraj radnog vremena
pause_start	TIME	početak prve pauze
pause_end	TIME	kraj prve pauze
wash_price	FLOAT	cijena jednog pranja
drying_price	FLOAT	cijena jednog sušenja
first_shift_worker_id (FK)	INT	id zaposlenika prve smjene (user.id)
middle	TIME	sredina dana, početak druge smjene
pause2_start	TIME	početak druge pauze
pause2_end	TIME	kraj druge pauze
second_shift_worker_id (FK)	INT	id zaposlenika druge smjene (user.id)

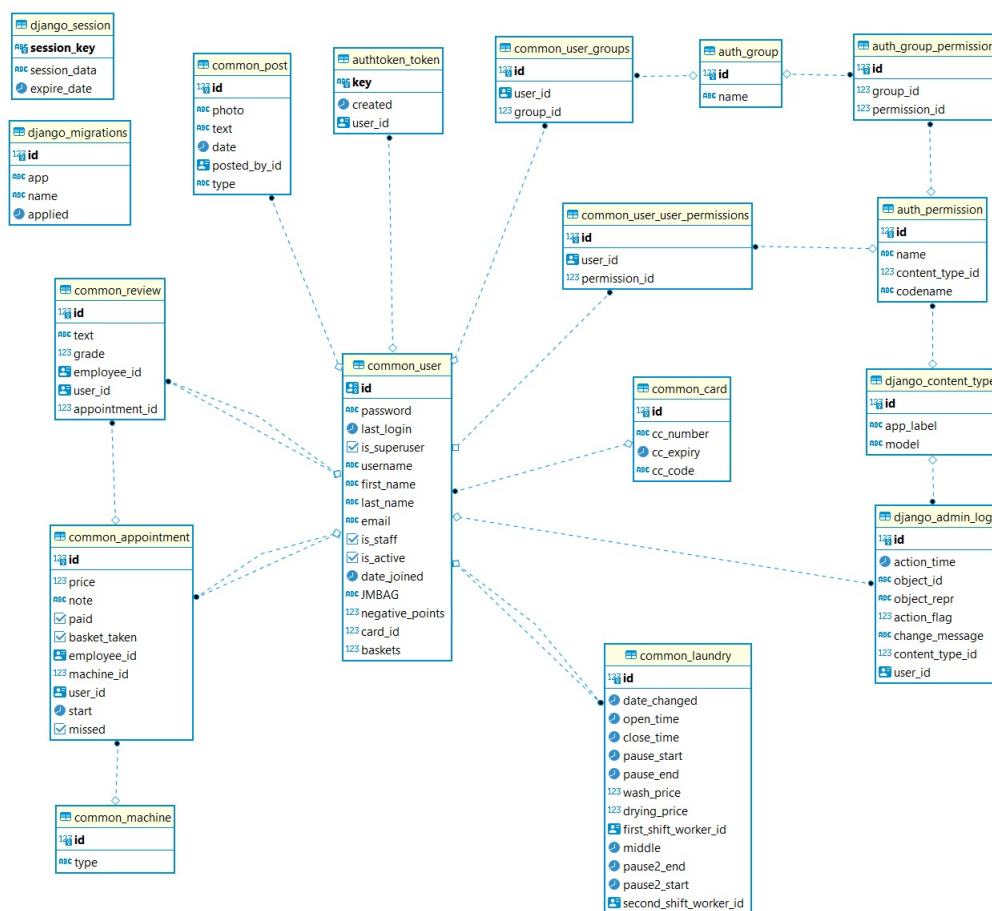
Card Ovaj entitet sadrži sve važne informacije vezane za kreditnu karticu korisnika. Sadrži attribute: id kartice, cc_number, cc_expiry i cc_code. Ovaj entitet u vezi je *One-to-One* s entitetom User preko id-a kreditne kartice.

Card		
id	INT	jedinstveni brojčani identifikator
cc_number	VARCHAR	broj kreditne kartice
cc_expiry	DATE	datum isteka kreditne kartice
cc_code	VARCHAR	CVV kod kreditne kartice

4.1.2 Dijagram baze podataka



Slika 4.2: Relacijska shema baze podataka

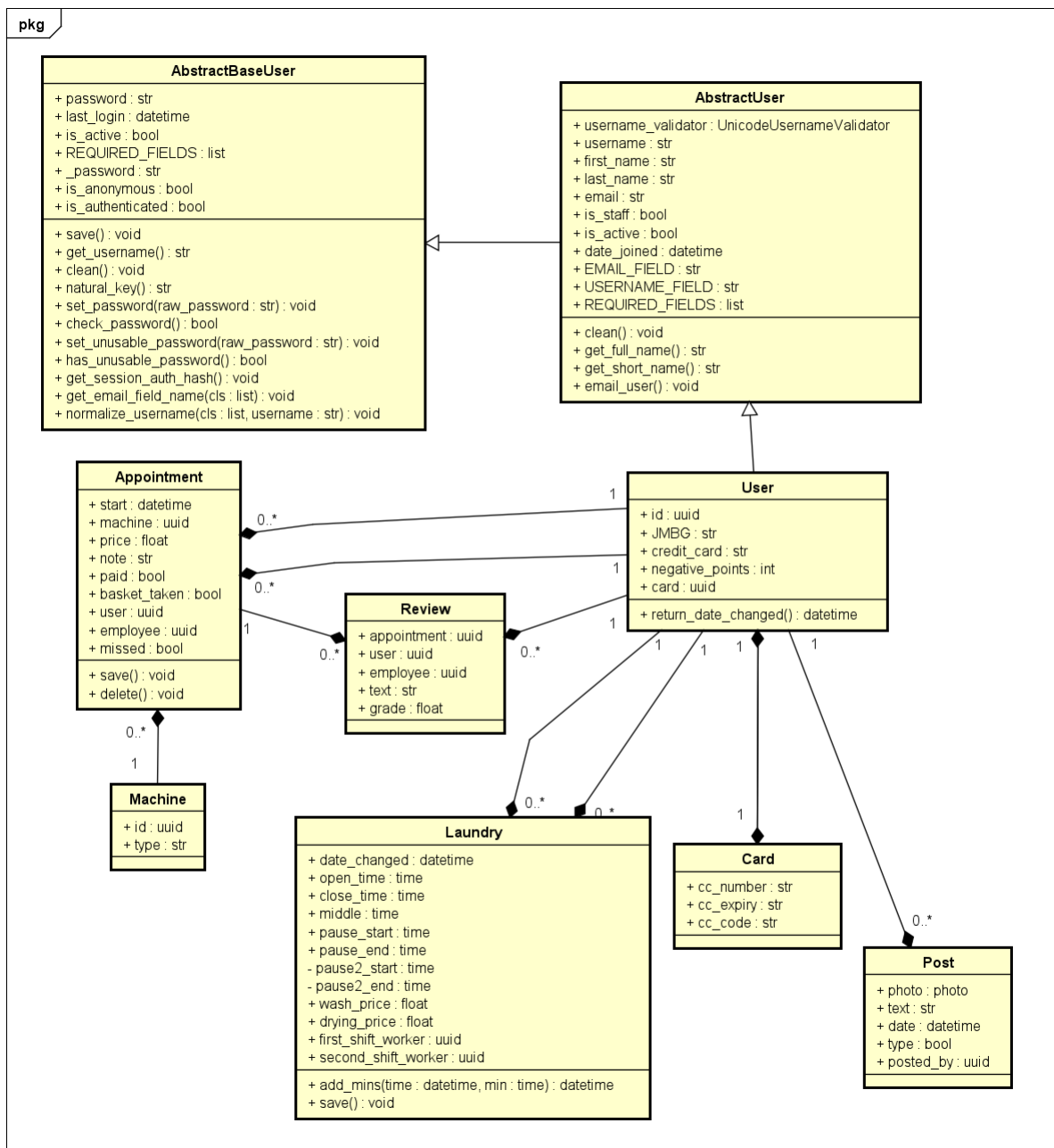


Slika 4.3: Relacijska shema Django baze podataka

4.2 Dijagram razreda

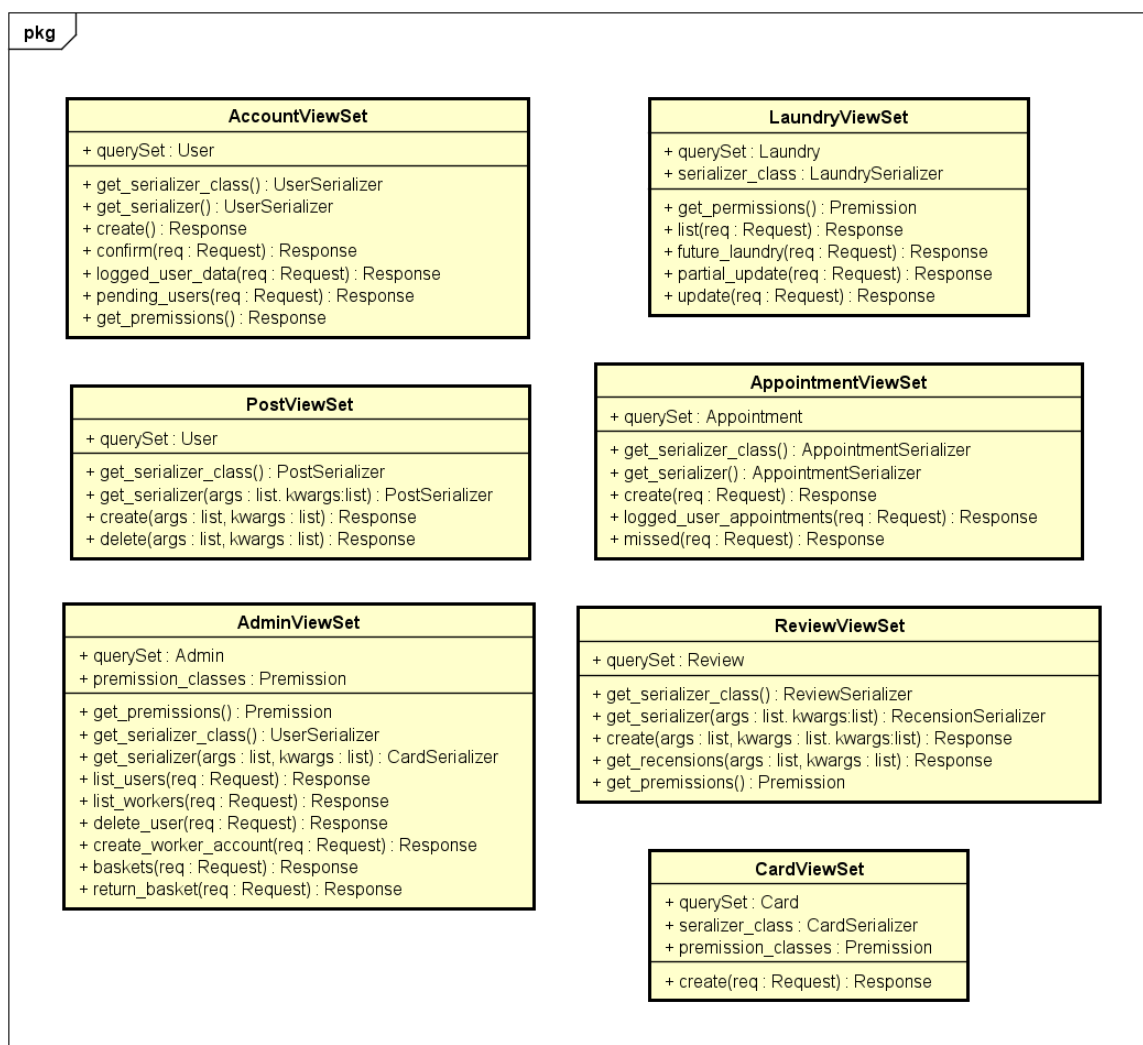
Zbog toga što je web poslužitelj napisan u Pythonu, ne možemo reći da u našem web modelu postoje razredi u pravom smislu riječi. Naime, moguće je pisati razred u Pythonu, ali nije moguće koristiti vrste pristupa. Sve metode, razredi i atributi su javne (i tako će biti označene na dijagramima). Moguće je naznačiti da želimo da se komponenta ponaša kao privatna korištenjem donje crtice u njenom imenu. Stoga zaključujemo da razredi u Pythonu postoje i moguće je izraditi dijagram razreda za komponente web aplikacije iako razredi nemaju potpune funkcionalnosti objektno orijentiranog jezika. Django koristi razrede prilikom svoje implementacije. Kao što je već navedeno, Django se temelji na modelu MTV (*Model-Template-View*).

Models u Django su Python razredi koji se automatski prenose u obliku relacija u bazu podataka korištenjem Djangovih karakterističnih *migrate* funkcionalnosti. Prilikom pisanja Modela potrebno je obratiti pozornost na njihovu povezanost i buduću brojnost u bazi podataka.



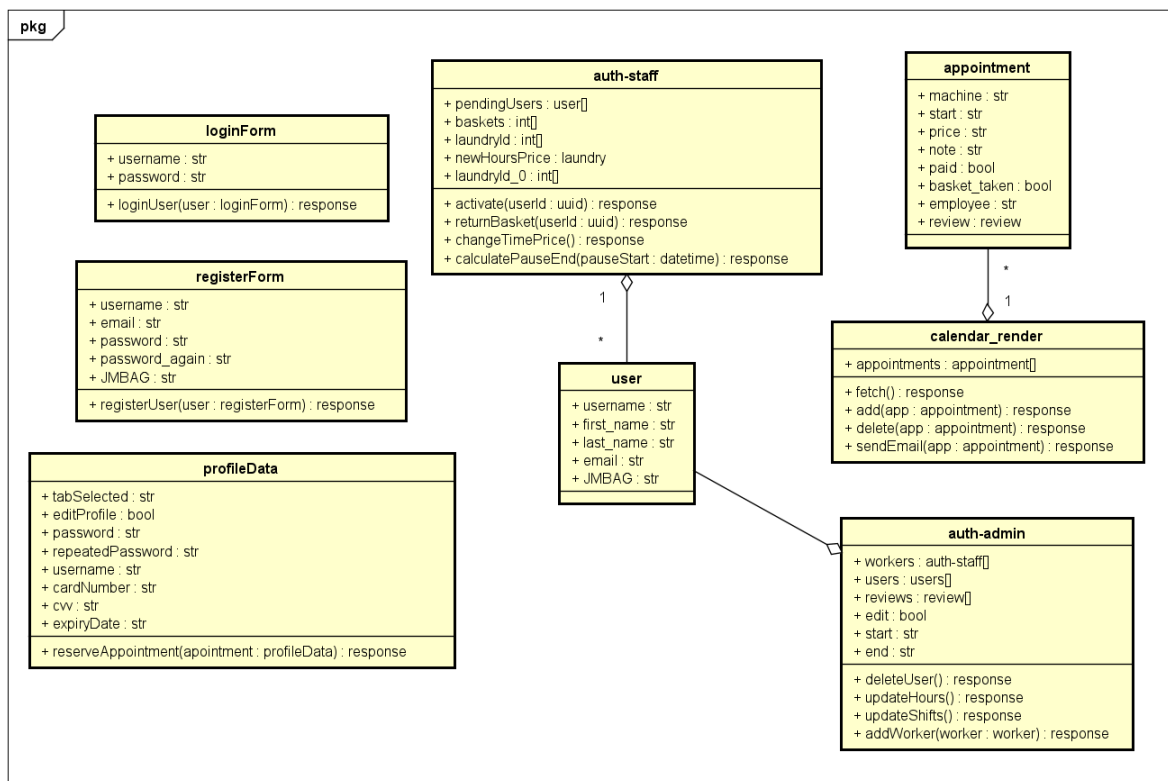
Slika 4.4: Dijagram razreda - dio Models

Views sadrže svu funkcionalnost i strukture podataka potrebne da se povežu funkcionalnosti Templatea i Modela. Osim što povezuju ta dva dijela sustava, *views* mogu sadržavati dodatnu funkcionalnost.



Slika 4.5: Dijagram razreda - dio Views

Templates sadrže sve strukture podataka potrebne da se podatak prenese s korisničkog sučelja u model. U našem slučaju korisničko sučelje dio je Nuxt.js radnog okvira. Nuxt.js uz pomoć *axios* sustava prenosi podatke na View.



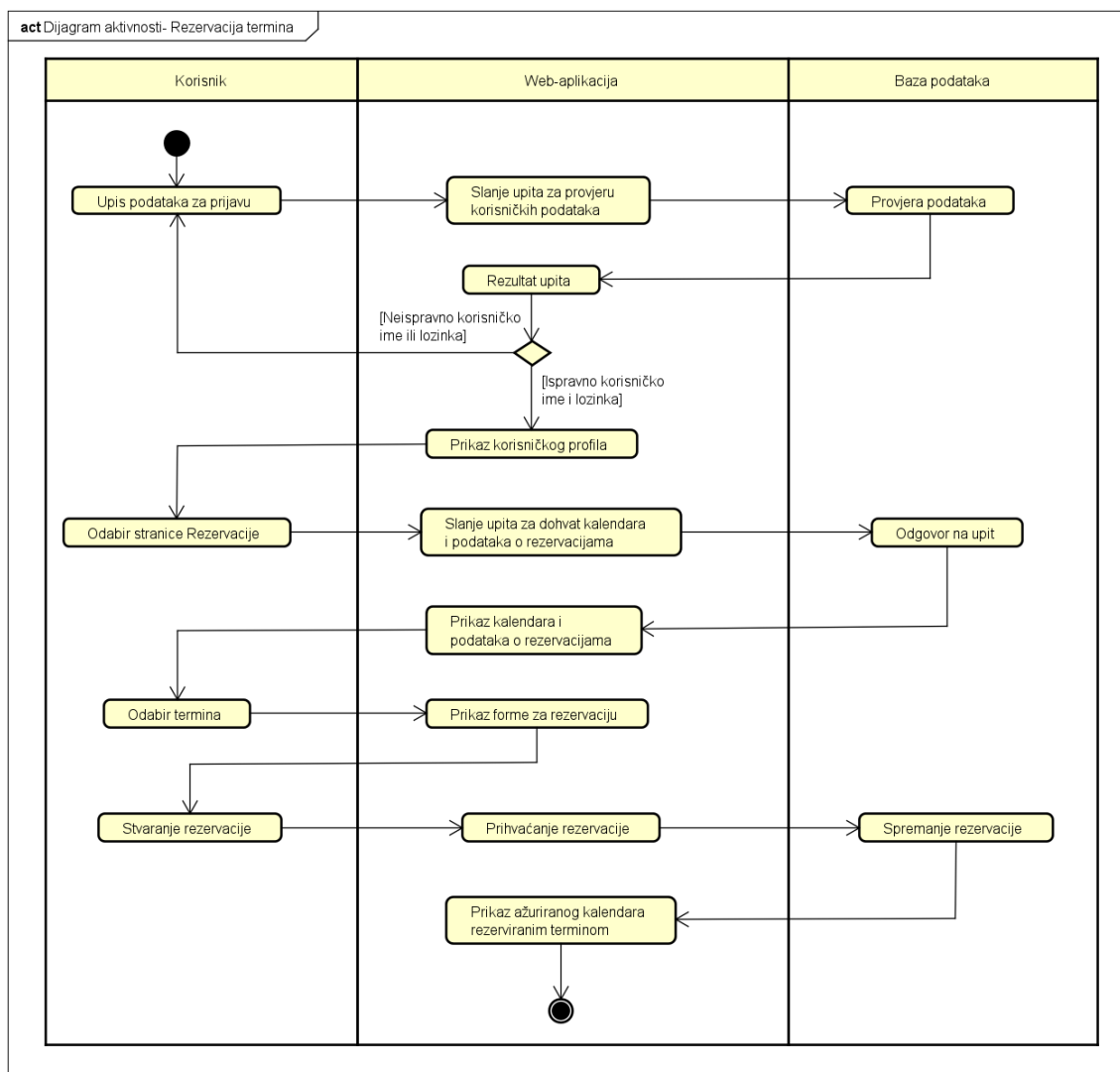
Slika 4.6: Dijagram razreda - dio Templates

4.3 Dijagram stanja

Dijagram stanja prikazuje stanja objekta te prijelaze iz jednog stanja u drugo temeljene na događajima. Na slici je prikazan dijagram stanja za zaposlenika. Nakon prijave, zaposleniku se prikazuje početna stranica. Učitaju se radno vrijeme i cijena pranja te sušenja kako bi se prikazali na početnoj stranici. Sa početne stranice, i bilo koje druge, može doći do stranica: Profil, Poslovi, Zaboravljeno, Zaposlenik i Rezervacije. Stranica Profil ima tri podstranice: Detalji, Lista rezervacija i Podaci za plaćanje. Detalje računa i podatke o plaćanju je moguće uređivati. Na stranicama Poslovi i Zaboravljeno moguće je napisati i objaviti objavu. Na stranici rezervacije je prikazan kalendar sa terminima. Odabirom zauzetog termina, on se može otkazati ili se može poslati mail korisniku. Za slobodni termin se može odabrati posudba košare te se može rezervirati. Na zaposleniku se mogu prihvatiti novi korisnici, vraćati košare i može se promijeniti vrijeme pauze.

4.4 Dijagram aktivnosti

Dijagram aktivnosti primjenjuje se za opis modela toka upravljanja ili toka podataka. Ne upotrebljava se za modeliranje događajima poticanog ponašanja. U modeliranju toka upravljanja svaki novi korak poduzima se nakon završenog prethodnog, a naglasak je na jednostavnosti. Na dijagramu aktivnosti 4.7 prikazan je proces rezervacije termina u praonici. Korisnik se prijavi u sustav, odabire stranicu Rezervacije koja prikaže kalendar s dostupnim terminima za rezervaciju i odabire slobodan termin. Zatim mu se prikazuje forma preko koje može odabrati posudbu košare te napisati bilješku. Nakon toga odabire Rezerviraj čime se termin rezervira te se korisniku prikaže osvježeni kalendar s njegovom upisanom rezervacijom termina.



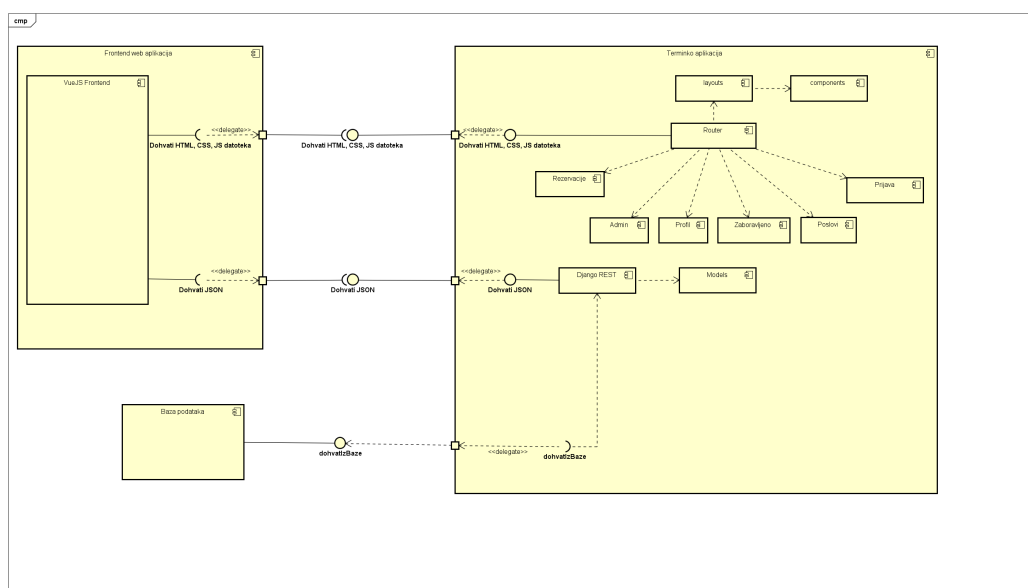
Slika 4.8: Dijagram aktivnosti

4.5 Dijagram komponenti

Dijagram komponenti prikazan na slici 4.9 prikazuje komponente i njihovu ovisnost unutar aplikacije.

Nuxt ima funkciju routera koji u ovisnosti o url-u na frontend web aplikacije šalje odgovarajuće HTML, CSS i Javascript dokumente. Sami frontend je podijeljen na logičke komponente koje su nazvane ovisno o imenu stranice tj. dijelu aplikacije kojemu se pristupa. Sve frontend komponente ovisne su o *layouts* i *components* komponentama iz kojih povlače podatke o zaglavlju stranice koje je vidljivo u svakom trenutku.

Django REST API komunicira s bazom podataka te poslužuje potrebne JSON podatke koji su mu dostupni unutar baze.



Slika 4.9: Dijagram komponenti

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

Komunikacija u timu realizirana je korištenjem aplikacije WhatsApp¹, a organizacija i podjela zadataka korištenjem aplikacije Trello². Za izradu UML dijagrama korišten je alat Astah UML³, a kao sustav za upravljanje izvornim kodom Git⁴. Udaljeni repozitorij projekta dostupan je na web platformi GitLab⁵.

Kao razvojno okruženje korišten je Microsoft Visual Studio Code⁶ - besplatni uređivač teksta tvrtke Microsoft. Sadrži podršku za debugging, izvođenje zadataka te sustav za upravljanje izvornim kodom. Koristi se za razvoj računalnih programa za operacijski sustav Windows, kao i za web-stranice, web-aplikacije, te ostale web-usluge. Jedan je od najpopularnijih razvojnih alata. Bazira se na radnom sučelju Electron, koje se koristi za razvoj Node.js Web aplikacija.

Aplikacija je napisana koristeći radni okvir Django⁷ i jezik Python⁸ za izradu backenda te Nuxt.js⁹ i jezik JavaScript¹⁰ za izradu frontenda. Nuxt.js je radni okvir koji se bazira na Vue.js¹¹, Node.js¹², Webpack¹³ i Babel.js¹⁴. Podržava generiranje statičkih web stranica i bazira se na modularnoj arhitekturi. Postoji preko 50 modula koji razvoj čine bržim i jednostavnijim. Django radni okvir omogućava brz razvoj sigurnih i održivih web stranica. Prati MTV (model-template-view) arhitekturni obrazac. Neke od poznatih stranica koje koriste Django su Instagram, Mozilla, The Washington Times, Bitbucket itd.

¹<https://www.whatsapp.com/>

²<https://trello.com/>

³<https://astah.net/products/astah-uml/>

⁴<https://git-scm.com/>

⁵<https://gitlab.com/>

⁶<https://code.visualstudio.com/>

⁷<https://www.djangoproject.com/>

⁸<https://www.python.org/>

⁹<https://nuxtjs.org/>

¹⁰<https://www.javascript.com/>

¹¹<https://vuejs.org/>

¹²<https://nodejs.org/en/>

¹³<https://webpack.js.org/>

¹⁴<https://babeljs.io/>

Za lokalno upravljanje i slanje upita na bazu podataka koristili smo alat pgAdmin¹⁵. Baza podataka se trenutno nalazi na poslužitelju Heroku¹⁶.

¹⁵<https://www.pgadmin.org/>

¹⁶<https://www.heroku.com/>

5.2 Ispitivanje programskog rješenja

5.2.1 Ispitivanje komponenti

U ovom ćemo poglavlju predstaviti 9 ispitnih slučajeva (unit testova). Svi se ispitni primjeri mogu pronaći u datoteci `janezi/izvorniKod/backend/common/tests.py` koja je sastavljena od 4 klase koje ukupno sadrže 9 testova te sadrži 137 linija programskog koda.

Ispitni slučajevi 1 i 2 odnose se na prijavu korisnika. Oba se ispitna slučaja nalaze u klasi `AccountTests`. U prvom se slučaju korisnik ne može prijaviti jer mu korisnički račun nije potvrđen, dok se u drugom slučaju, odmah nakon potvrde računa, korisnik može uspješno prijaviti. U prvom je slučaju očekivan odgovor 400 (pogreška), dok je u drugom slučaju očekivan odgovor 200. Oba ispitna primjera daju očekivani rezultat.

```
class AccountTests(TestCase):
    def setUp(self):
        User.objects.create_user(JMBAG="1234567890", username='test', password='1Zaboraviti2', is_active=False)

    def test_unactivated_user_cant_login(self):
        """Izaziva grešku jer korisnik nije aktiviran"""
        response = self.client.post('/api/token-auth/', {'username': 'test', 'password': '1Zaboraviti2'})
        self.assertEqual(response.status_code, status.HTTP_400_BAD_REQUEST)

    def test_activated_user_can_login(self):
        """Rubni uvjet: može li se ulogirati odmah nakon što je potvrđen račun"""
        user = User.objects.filter(username='test').first()
        user.is_active = True
        user.save()
        response = self.client.post('/api/token-auth/', {'username': 'test', 'password': '1Zaboraviti2'})
        self.assertEqual(response.status_code, status.HTTP_200_OK)
```

Slika 5.1: Ispitni slučajevi vezani za prijavu korisnika

Ispitni slučajevi 3 i 4 vezani su za administratora te provjeravaju ispravnost funkcionalnosti blokiranja korisnika. Oba se ispitna slučaja nalaze u klasi `AdminTests`. Prvi slučaj izaziva pogrešku (kod 401 UNAUTHORIZED) jer korisnik koji nije administrator pokušava blokirati račun drugog korisnika. U drugom slučaju, korisnik koji je administrator blokira korisnika i provjerava se točnost akcije. Oba ispitna primjera daju očekivani rezultat.

```
class AdminTests(TestCase):
    def setUp(self):
        User.objects.create_user(JMBAG="1234567890", username='test', password='1Zaboraviti2')
        User.objects.create_user(JMBAG="1234567899", username='admin', password='1Zaboraviti2', is_superuser=True, is_staff=True)

    def test_user_can_block_user(self):
        """Izaziva grešku jer korisnik nije admin"""
        user_id = User.objects.filter(username='test').first().id
        response = self.client.delete(f'/api/admin/{user_id}/delete_user/')
        self.assertEqual(response.status_code, status.HTTP_401_UNAUTHORIZED)

    def test_admin_can_block_user(self):
        """Trebalo bi se smjeti kad je admin ulogiran"""
        user = User.objects.get(username='admin')
        client = APIClient()
        # client.login(username='admin', password='1Zaboraviti2')
        token = self.client.post('/api/token-auth/', {'username': 'admin', 'password': '1Zaboraviti2'}).data.get('token')
        # print(self.client.get('/api/account/logged_user_data/', headers={'Authorization': 'token ' + token}).data)
        user_id = User.objects.filter(username='test').first().id
        client.force_authenticate(user=user, token=token)
        response = client.delete(f'/api/admin/{user_id}/delete_user/') # , headers={'Authorization': 'token ' + token})
        # force_authenticate(response, user=user, token=token)
        self.assertEqual(response.status_code, status.HTTP_200_OK)
```

Slika 5.2: Ispitni slučajevi vezani za prijavu korisnika

Ispitni slučajevi 5, 6 i 7 odnose se na funkcionalnost vraćanja košare. Oba se ispitna slučaja nalaze u klasi WorkerTests. Prvi slučaj predstavlja situaciju u kojoj korisnik koji ne radi u praonici pokušava vratiti košaru. Taj bi slučaj trebao vratiti poruku 401 UNAUTHORIZED kako netko ne bi mogao zloupotrebjavati povrat košare. Drugi test provjerava ponašanje aplikacije u slučaju kad radnik pokuša označiti da je netko vratio košaru te osim provjere uspješnosti (kod 200), provjerava se i je li se broj košara tog korisnika smanjio na 0 (linija 75 u kodu). Sva tri ispitna primjera daju rezultat u skladu s očekivanjima.

```
class WorkerTests(TestCase):
    def setUp(self):
        User.objects.create_user(JMBAG="1234567890", username='test', password='1Zaboraviti2', baskets=1)
        User.objects.create_user(JMBAG="1234567899", username='admin', password='1Zaboraviti2', is_superuser=True, is_staff=True)

    def test_return_basket_non_worker(self):
        """Izaziva grešku jer korisnik nije zaposlenik"""
        user_id = User.objects.filter(username='test').first().id
        response = self.client.post(f'/api/admin/{user_id}/return_basket/')
        self.assertEqual(response.status_code, status.HTTP_401_UNAUTHORIZED)

    def test_return_basket_worker(self):
        """Ne izaziva grešku jer je korisnik zaposlenik"""
        user = User.objects.get(username='admin')
        client = APIClient()
        # client.login(username='admin', password='1Zaboraviti2')
        token = self.client.post('/api/token-auth/', {'username': 'admin', 'password': '1Zaboraviti2'}).data.get('token')
        # print(self.client.get('/api/account/logged_user_data/', headers={'Authorization': 'token ' + token}).data)
        user_id = User.objects.filter(username='test').first().id
        client.force_authenticate(user=user, token=token)
        response = client.post(f'/api/admin/{user_id}/return_basket/') # , headers={'Authorization': 'token ' + token})
        self.assertEqual(response.status_code, status.HTTP_200_OK)
        self.assertEqual(User.objects.filter(username='test').first().baskets, 0)

    def test_return_basket_when_no_baskets_are_borrowed(self):
        """Provjera hoće li broj košara biti negativan"""
        user = User.objects.get(username='admin')
        client = APIClient()
        # client.login(username='admin', password='1Zaboraviti2')
        token = self.client.post('/api/token-auth/', {'username': 'admin', 'password': '1Zaboraviti2'}).data.get('token')
        # print(self.client.get('/api/account/logged_user_data/', headers={'Authorization': 'token ' + token}).data)
        user_id = User.objects.filter(username='test').first().id
        client.force_authenticate(user=user, token=token)
        response = client.post(f'/api/admin/{user_id}/return_basket/') # , headers={'Authorization': 'token ' + token})
        # force_authenticate(response, user=user, token=token)
        self.assertEqual(response.status_code, status.HTTP_200_OK)
        self.assertEqual(User.objects.filter(username='test').first().baskets, 0)
```

Slika 5.3: Ispitni slučajevi vezani za vraćanje košara

Posljednja klasa ispitnih primjera, nalazi se u klasi AppointmentTests i sadrži ispitne primjere 8 i 9. Osmi ispitni primjer provjerava može li neregistrirani korisnik napraviti rezervaciju te je očekivano ponašanje pogreška s kodom 401. Ispitni primjer br. 9 ispituje rezervaciju termina nakon prijave i provjerava je li radnja bila uspješna (kod 201 CREATED). Oba ispitna primjera daju očekivani rezultat.

```
class AppointmentTests(TestCase):
    def setUp(self):
        User.objects.create_user(JMBAG="1234567890", username='test', password='1Zaboraviti2')
        Machine.objects.create(type='washer')
        Laundry.objects.create(
            open_time=datetime.time(8, 0),
            close_time=datetime.time(20, 0),
            pause_start=datetime.time(10, 0),
            pause2_start=datetime.time(16, 0),
            drying_price=10,
            wash_price=10,
            date_changed=datetime.datetime.now(pytz.UTC) - datetime.timedelta(days=1)
        )

    def test_unregistered_user_can_make_an_appointment(self):
        """Izaziva grešku jer korisnik nije ulogiran"""
        user_id = User.objects.filter(username='test').first().id
        response = self.client.post(
            f'/api/appointment/',
            {
                'machine': Machine.objects.first().id,
                'start': '2021-10-10T8:00Z',
                'paid': False
            }
        )
        self.assertEqual(response.status_code, status.HTTP_401_UNAUTHORIZED)
```

Slika 5.4: Ispitni slučajevi vezani za rezervaciju termina, slika 1/2

```
def test_registered_user_can_make_an_appointment(self):
    """Trebalo bi se smjeti kad je korisnik ulogiran"""
    user = User.objects.get(username='test')
    client = APIClient()
    # client.login(username='admin', password='1Zaboraviti2')
    token = self.client.post('/api/token-auth/', {'username': 'admin', 'password': '1Zaboraviti2'}).data.get('token')
    # print(self.client.get('/api/account/Logged_user_data/', headers={'Authorization': 'token ' + token}).data)
    user_id = User.objects.filter(username='test').first().id
    client.force_authenticate(user=user, token=token)
    response = client.post(
        f'/api/appointment/',
        {
            'machine': Machine.objects.first().id,
            'start': '2021-10-10T8:00Z',
            'paid': False
        }
    )
    # , headers={'Authorization': 'token ' + token})
    # force_authenticate(response, user=user, token=token)
    self.assertEqual(response.status_code, status.HTTP_201_CREATED)
```

Slika 5.5: Ispitni slučajevi vezani za rezervaciju termina, slika 2/2

Svi se spomenuti testovi mogu pokrenuti iz terminala aktivacijom virtualnog okruženja, pozicioniranjem u mapu backend i pozivanjem naredbe `./manage.py test`. Kao što je već spomenuto, svi se testovi izvršavaju točno, a slika 5.6 pokretanja naredbe `./manage.py test` dana je u prilogu.


```
(janezi) jan@jans-zenbook:~/FER/Projects/janezi/izvorniKod/backend$ ./manage.py test
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
.....
-----
Ran 9 tests in 2.398s

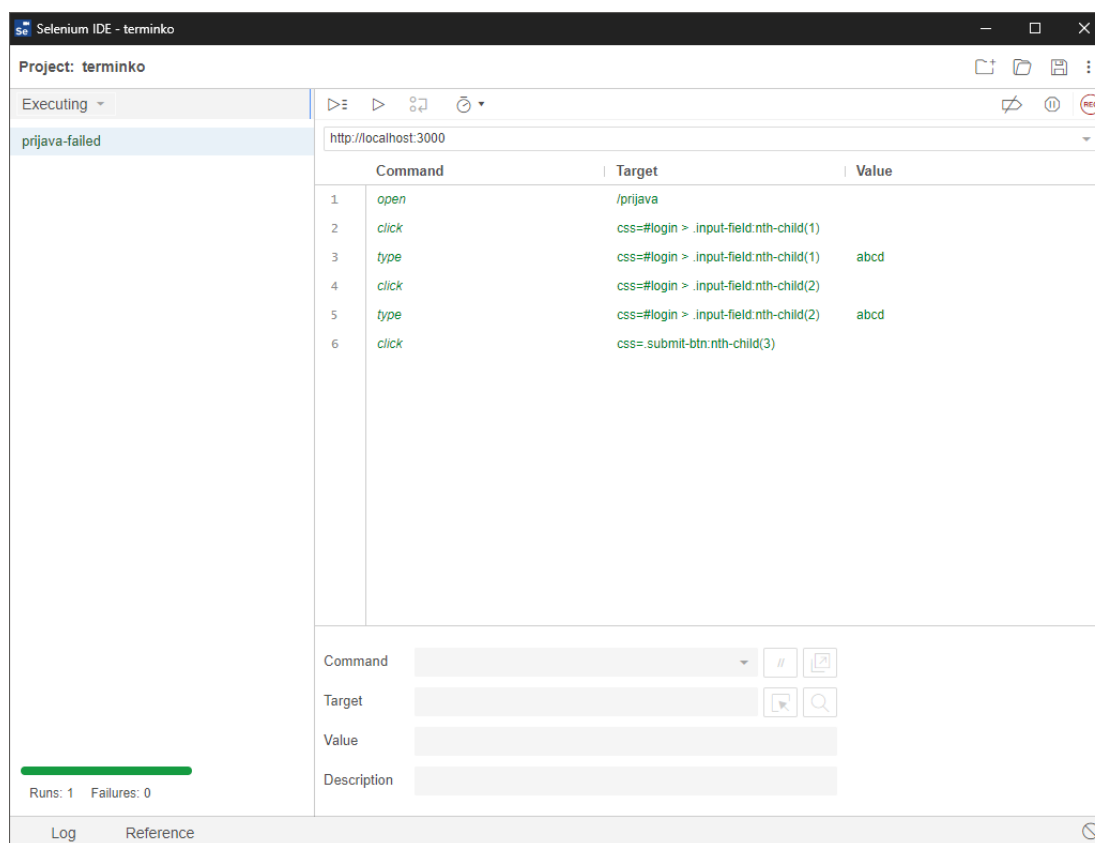
OK
Destroying test database for alias 'default'...
```

Slika 5.6: Pokretanje svih ispitnih slučajeva u terminalu

5.2.2 Ispitivanje sustava

1. Pokušaj prijave neodgovarajućom lozinkom.

- Ulaz: korsničko ime i kriva lozinka u formi za prijavu
- Izlaz: poruka sa ispisom "Pogrešna lozinka"

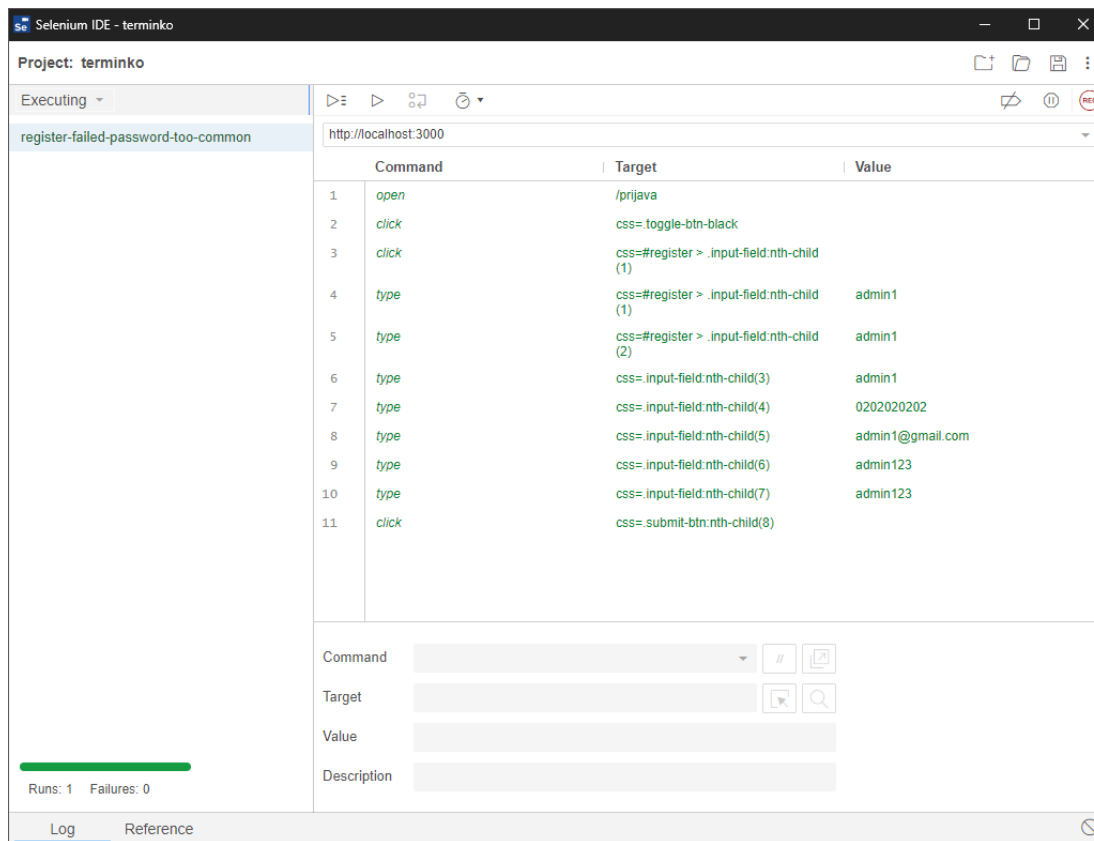


Slika 5.7: Selenium konfiguracija za test broj 1

Ovim ispitom provjeravamo pokušaj prijave registriranog korisnika u sustav sa neispravnom lozinkom (UC2 - Prijava). Korisniku se ne daje pristup sustavu te se ispisuje poruka "Pogrešna lozinka" kao povratna poruka njegove radnje.

2. Pokušaj registracije sa previše uobičajnom lozinkom.

- Ulaz: korsničko ime i preuobičajna lozinka u formi za registraciju
- Izlaz: poruka sa ispisom "Password too common"

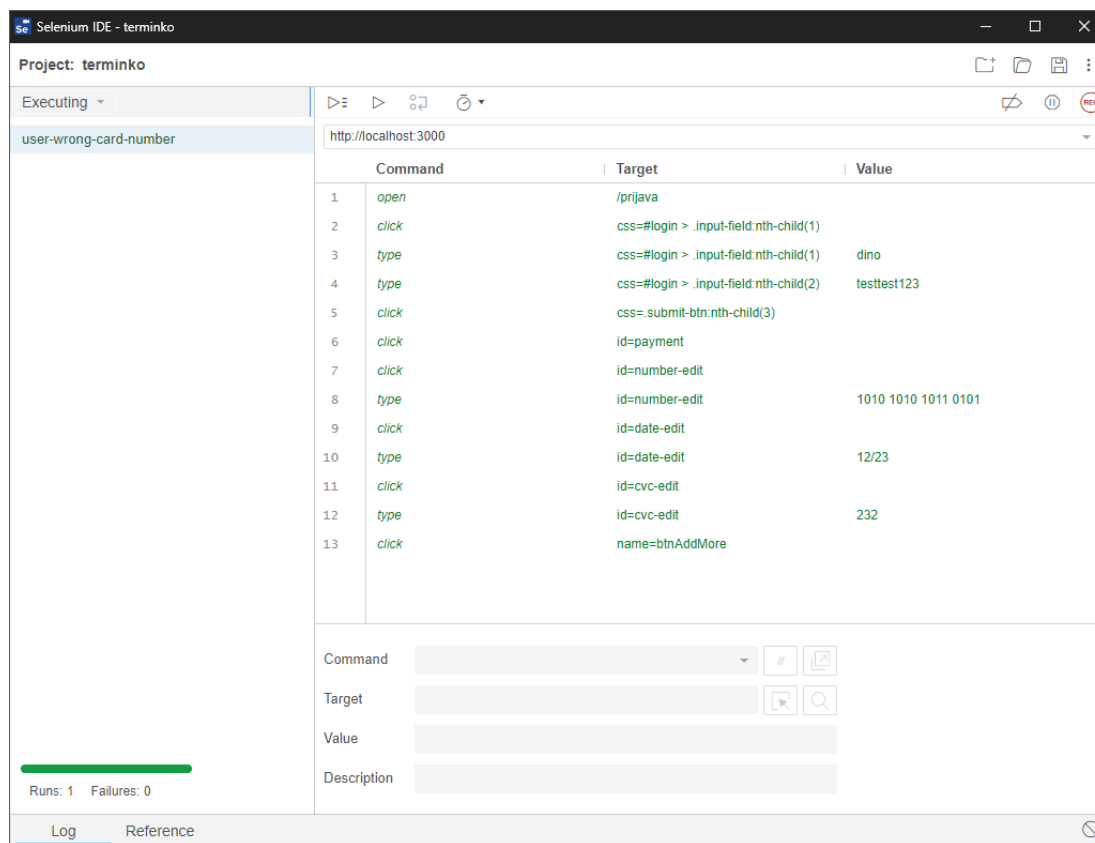


Slika 5.8: Selenium konfiguracija za test broj 2

Ispitujemo pokušaj registracije korisnika se previše uobičajnom lozinkom (UC1 - Registracija). Ako se korisnik pokušava registrirati u sustav sa svim ispravnim podacima ali uobičajnom lozinkom (kao što su 123456789, abcdefgh, ime123,...). Odziv sustava je neuspjela registracija s povratnom porukom "Password too common".

3. Pogrešni podaci kreditne kartice.

- Ulaz: broj kreditne kartice, CVV i datum isteka
- Izlaz: poruka sa ispisom "Nevaljajuća kreditna kartica"

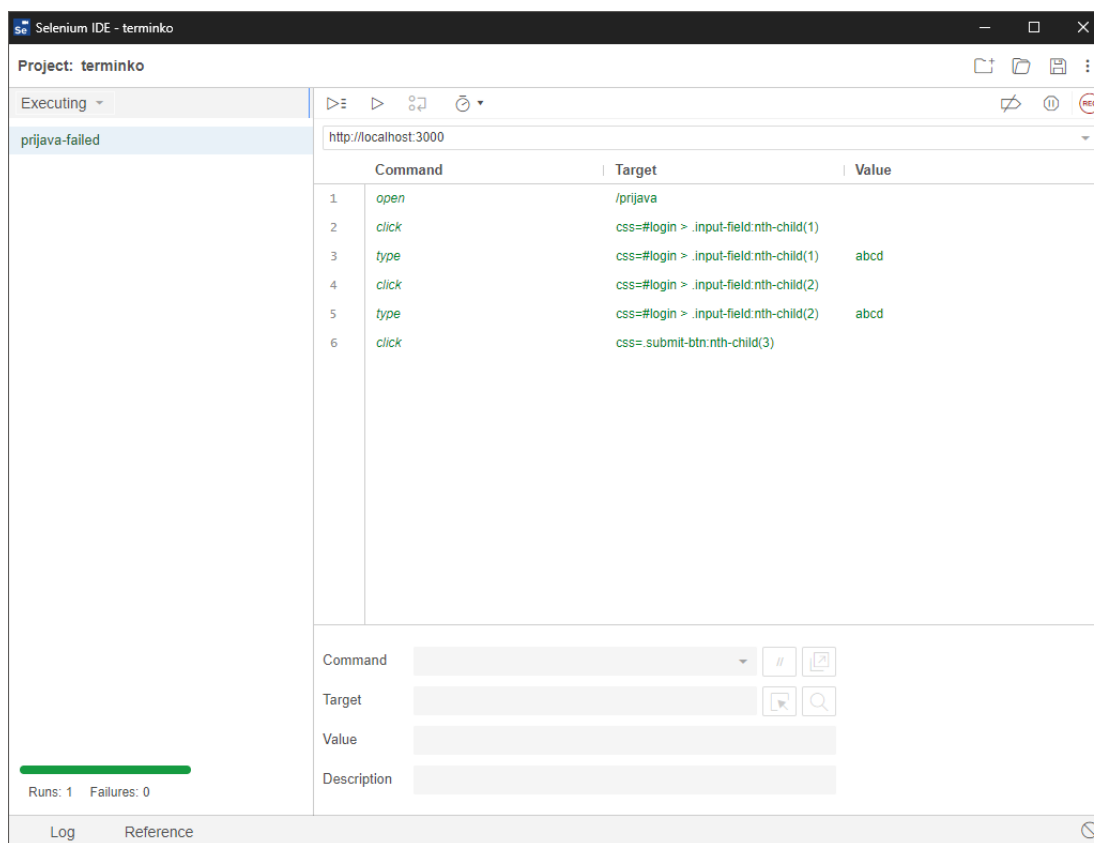


Slika 5.9: Selenium konfiguracija za test broj 3

Ovim ispitom provjeravamo pokušaj dodavanja nepostojeće kreditne kartice u sustav korisnika (UC3 - Pregled vlastitog profila i uređivanje podataka). Ako jedan od odgovarajućih potrebnih podataka za kartično plaćanje nije točan (broj kreditne kartice, CVV ili datum isteka valjanosti) sustav ne dozvoljava spremanje navedene kartice na profil korisnika i vraća poruku "Nevaljajuća kreditna kartica".

4. Registracija i brisanje postojećeg slobodnog termina.

- Ulaz: odabir termina registracija i brisanje
- Izlaz: uspješna dodjela termina i brisanje (uz negativne bodove po potrebi)

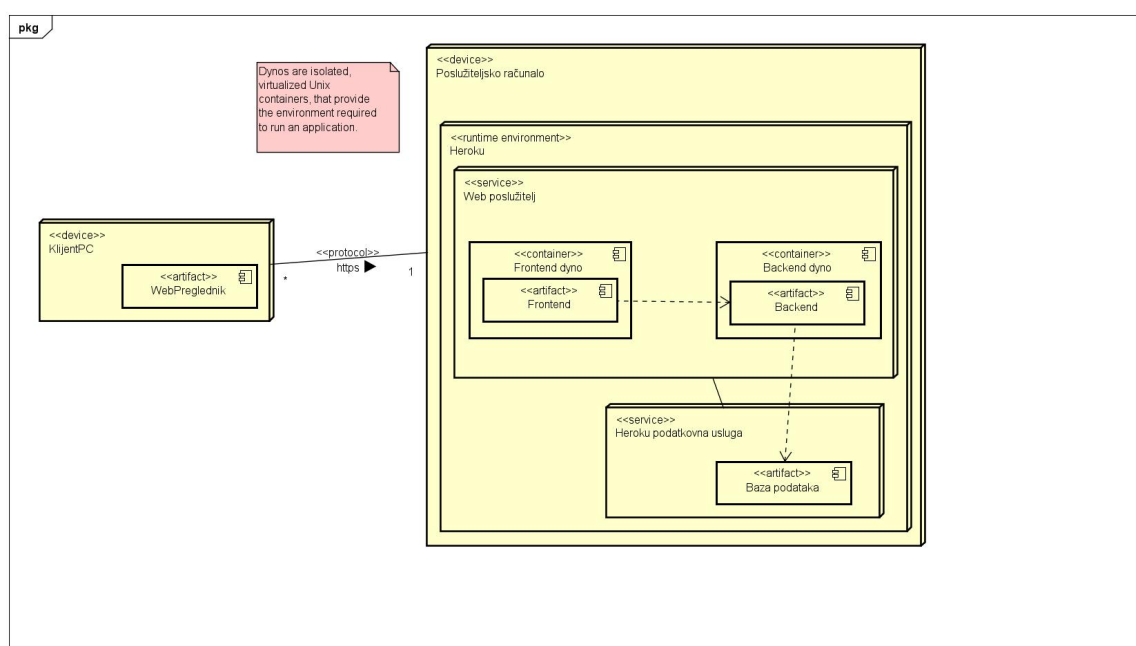


Slika 5.10: Selenium konfiguracija za test broj 4

Ispitujemo pokušaj rezervacije termina te brisanje vlastitog rezerviranog termina (UC4 - Rezervacija termina za pranje veša). Ako slobodni termin ne postoji u kalendaru korisnik ga ne može rezervirati. Korisnik isto tako može obrisati rezervirani termin, ali ako ga obriše tri sata prije početka termina dodjeljuju mu se negativni bodovi u koje uvid ima administrator sustava.

5.3 Dijagram razmještaja

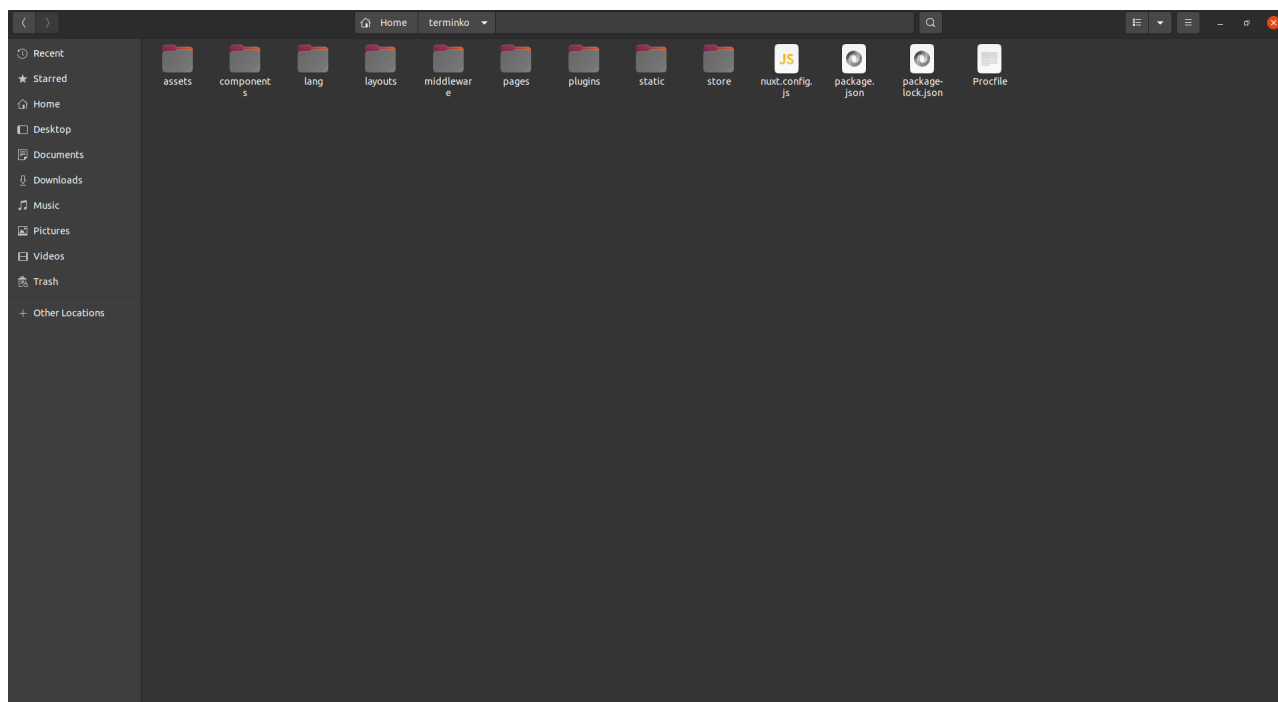
Na poslužiteljskom računalu se nalaze web poslužitelj i poslužitelj baze podataka. Oni se nalaze unutar Heroku izvršne okoline koja sadrži usluge koje organiziraju i upravljaju izvršavanjem i razmjerom aplikacije. Frontend i backend se pokreću u Dynosima. To su lagana, izolirana okruženja koja pružaju računalnu snagu, memoriju, OS i "Ephemeral" datotečni sustav. Klijenti koriste web preglednik kako bi pristupili web aplikaciji. Sustav je baziran na arhitekturi "klijent-poslužitelj", a komunikacija između računala korisnika (klijent, zaposlenik, administrator) i poslužitelja odvija se preko HTTPS veze.



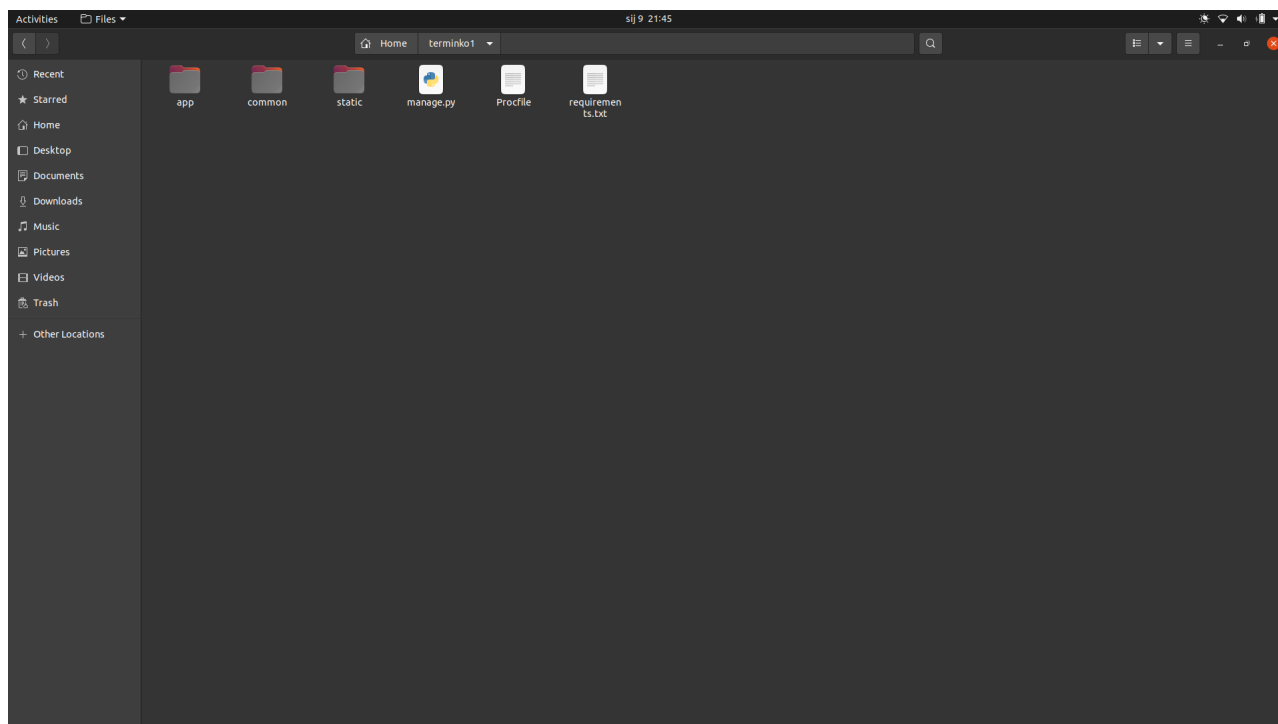
Slika 5.11: Dijagram stanja

5.4 Upute za puštanje u pogon

Za puštanje u pogon, koristili smo heroku server (<https://www.heroku.com/>). Prvi korak za puštanje u pogon bila je podjela aplikacije na poslužiteljsku i korisničku stranu. Svaki se od tih djelova aplikacije treba staviti u zasebnu mapu. Imena mapi moraju odgovarati imenima aplikacija na heroku poslužitelju. Mi smo se odlučili za naziv terminko za korisničku stranu te naziv terminko1 za poslužiteljsku stranu.

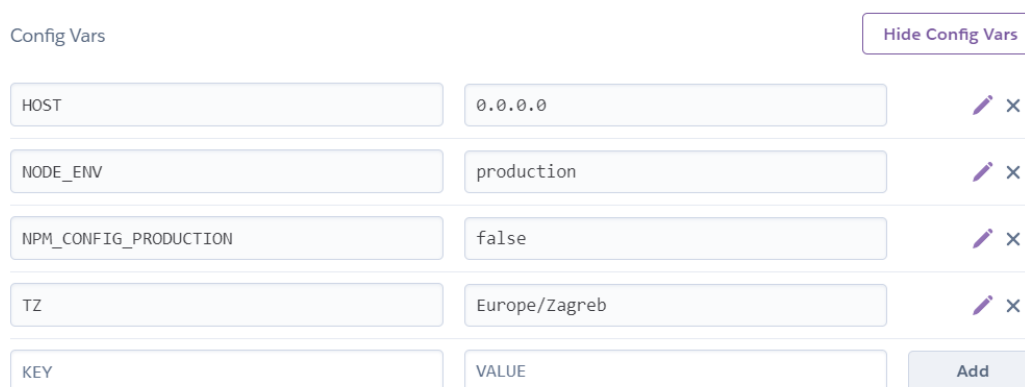


Slika 5.12: Mapa za frontend: terminko

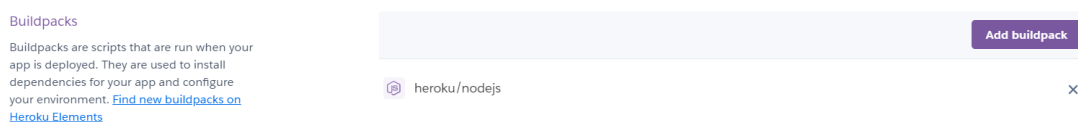


Slika 5.13: Mapa za backend: terminko1

Prvo smo na heroku postavili klijentsku stranu na način da smo prvo napravili aplikaciju terminko. Nakon izrade aplikacije, u odjeljku settings postavili smo konfiguracijske varijable kako je prikazano na slici te smo dodali "buildpack" za Node.js.

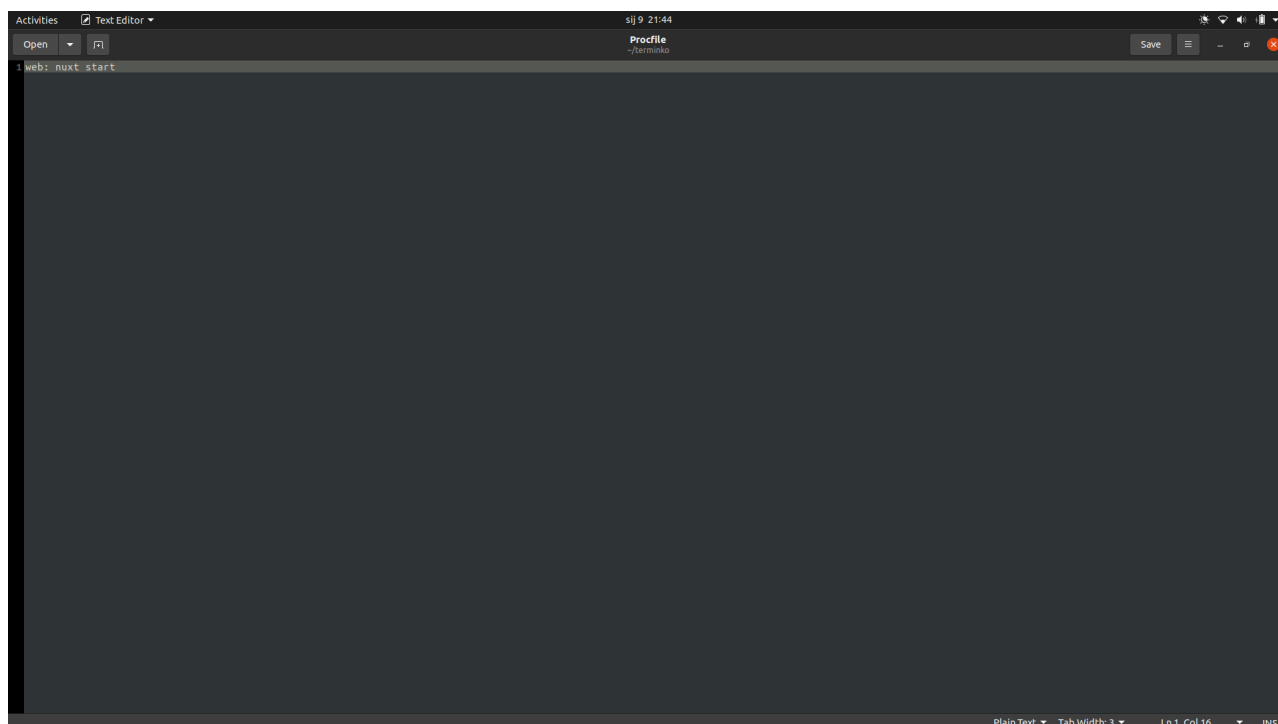


Slika 5.14: Konfiguracija za frontend

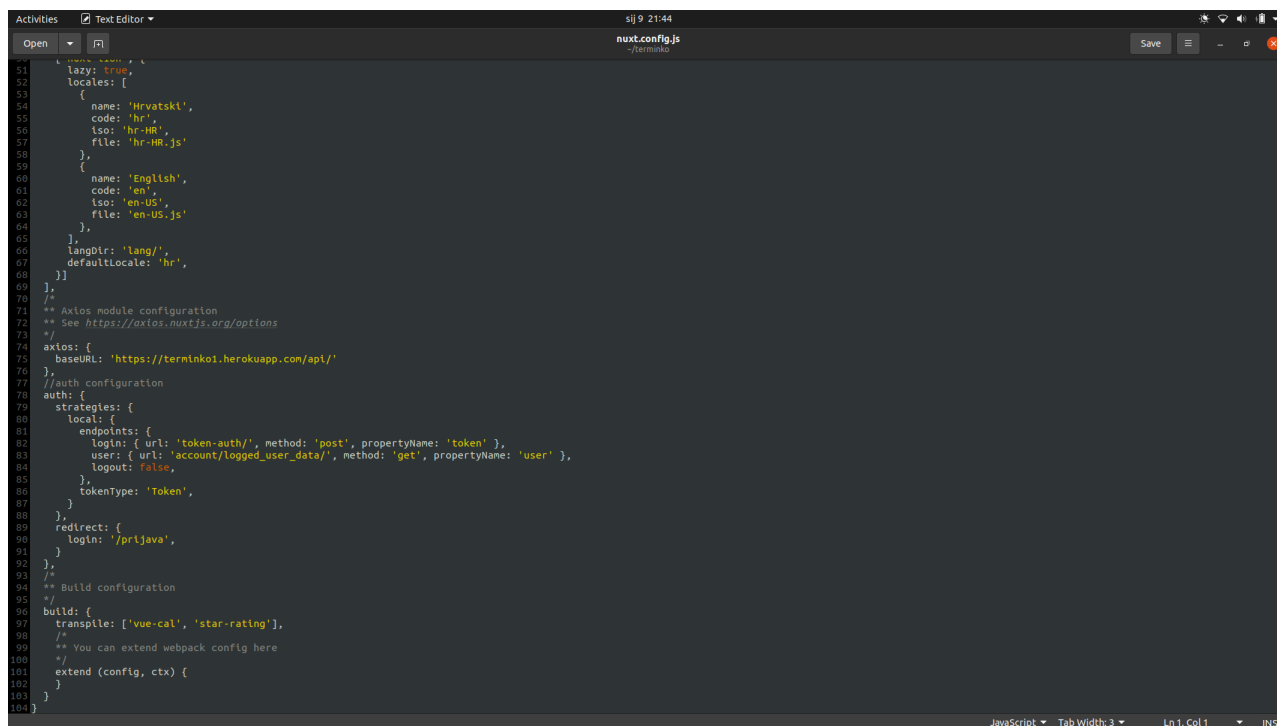


Slika 5.15: Buildpack za frontend

Na lokalnom smo stroju, u mapu terminko dodali Procfile i u nuxt-config.js dodali novi base-URL za upite na backend



Slika 5.16: Procfile za frontend

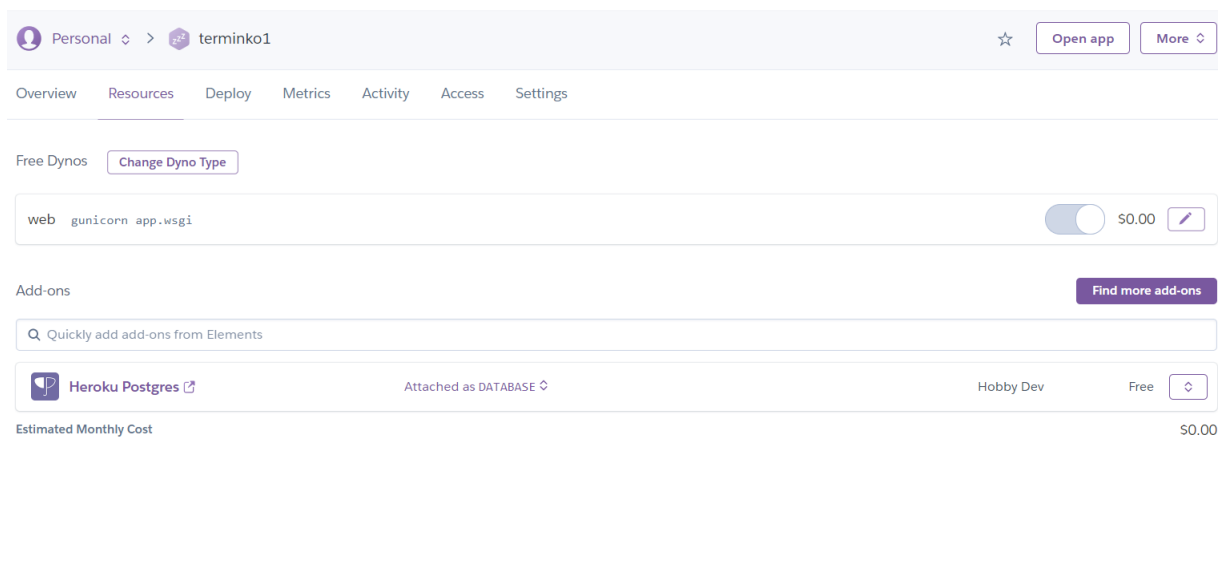


```
51  lazy: true,
52  locales: [
53    {
54      name: 'Hrvatski',
55      code: 'hr',
56      iso: 'hr-HR',
57      file: 'hr-HR.js'
58    },
59    {
60      name: 'English',
61      code: 'en',
62      iso: 'en-US',
63      file: 'en-US.js'
64    },
65  ],
66  langDir: 'lang/',
67  defaultLocale: 'hr',
68  })
69  },
70  /*
71  ** Axios module configuration
72  ** See https://axios.nuxtjs.org/options
73  */
74  axios: {
75    baseURL: 'https://terminko1.herokuapp.com/api/'
76  },
77  //auth configuration
78  auth: {
79    strategies: {
80      local: {
81        endpoints: {
82          login: { url: 'token-auth/', method: 'post', propertyName: 'token' },
83          user: { url: 'account/logged_user_data/', method: 'get', propertyName: 'user' },
84          logout: false,
85        },
86        tokenType: 'Token',
87      },
88    },
89    redirect: {
90      login: '/prijava',
91    },
92  },
93  /*
94  ** Build configuration
95  */
96  build: {
97    transpile: ['vue-cal', 'star-rating'],
98    /*
99    ** You can extend webpack config here
100    */
101    extend (config, ctx) {
102    }
103  }
104 }
```

Slika 5.17: Promjena axios base-URL

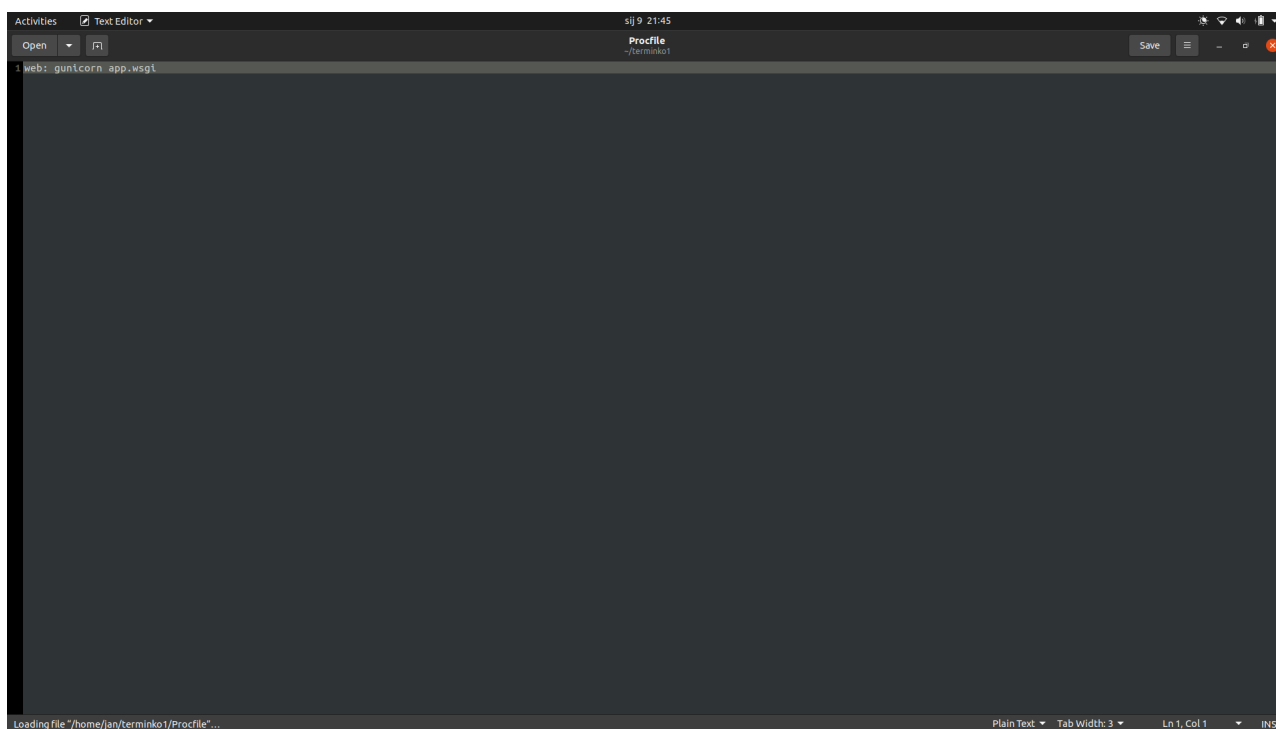
Nakon toga, još nam je preosalo pozicionirati se u terminko mapu u terminalu, napisati naredbe `git init`, `heroku git:remote -a terminko` i `git push heroku master`. Nakon toga je klijentska strana bila spremna te smo krenuli s poslužiteljskom stranom.

Za poslužiteljsku smo stranu napravili aplikaciju `terminko1` na heroku. Na poslužiteljskoj strani nije bilo konfiguracijskih varijabli, dok smo za "buildpack" stavili `heroku/python` u settings odjeljku. Za razliku od fontenda, na backendu smo na heroku, u tabu resources dodali Heroku Postgres.



Slika 5.18: Postavljanje Postgres baze podataka na heroku

Dakako i u mapu terminko1 morali smo dodati Procfile.



Slika 5.19: Procfile na backendu

Nakon toga, dobili smo podatke za prijavu na postgres server koje smo stavili u settings.py datoteku na backendu. U istoj smo datoteci promjenili i CORS origin whitelist, koji označava odalke nas backend može primiti zahtjeve, te allowed

hosts, koji označava na kojim se poslužiteljima naš backend smije pokretati.

```
90
91 DATABASES = {
92     'default': {
93         'ENGINE': 'django.db.backends.postgresql_psycopg2',
94         'NAME': 'd2ggeuqildk62p',
95         'USER': 'ztfxvoeypxntox',
96         'PASSWORD': 'a56fbc8b838d1e3a6c9472095cb14ab69c80860f5d85fae27eeec3d4c38267df',
97         'HOST': 'ec2-54-75-225-52.eu-west-1.compute.amazonaws.com',
98         'PORT': '5432',
99     }
100 }
101
```

Slika 5.20: Postavljanje Postgres konfiguracije u settings.py

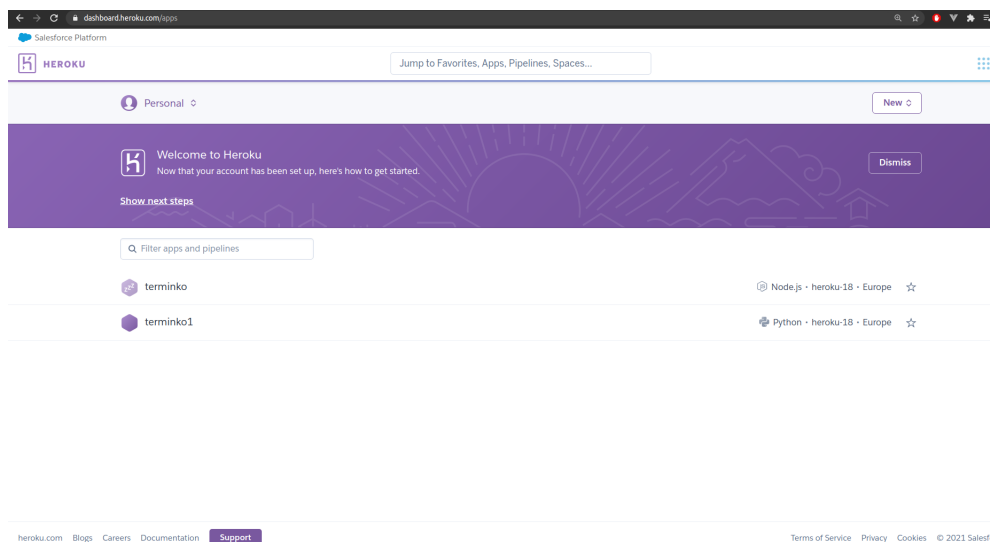
```
58
59 CORS_ORIGIN_WHITELIST = [
60     'https://localhost:3000',
61     'https://127.0.0.1:3000',
62     'https://terminko.herokuapp.com',
63     'https://terminko1.herokuapp.com',
64 ]
65
```

Slika 5.21: Postavljanje cors whitelist u settings.py

```
26
27 ALLOWED_HOSTS = ['0.0.0.0', 'terminko1.herokuapp.com', '127.0.0.1']
28
```

Slika 5.22: Postavljanje allowed hosts varijable u settings.py

Nakon toga, još nam je preosalo pozicionirati se u terminko1 mapu u terminalu, napisati naredbe git init, heroku git:remote -a terminko1 i git push heroku master. Nakon toga, cijela je aplikacija bila spremna.



Slika 5.23: Prikaz stanja na heroku po završetku deploya

6. Zaključak i budući rad

Projekt kojim smo se bavili tijekom semestra bio je izrada web aplikacije za rezerviranje termina u praonici rublja koja bi prvenstveno olakšala taj proces studentima, ali i zaposlenicima iste. Aplikacija ima potencijal primjene i na druge sustave. Nakon tromjesečnog timskog rada i razvoja aplikacije ostvarili smo zadani cilj. Izvedba projekta odvijala se kroz dvije faze.

U prvoj fazi projekta naglasak je bio na okupljanju tima, njegovom upoznavanju, analiziranju sposobnosti i okvirnoj raspodjeli budućih poslova. Isto tako, većina se vremena posvetila osmišljavanju funkcionalnosti, analiziranju zahtjeva te dokumentiranju istih. Detaljna dokumentacija u koju smo uložili mnogo truda i vremena bila je izvrstan temelj za drugu fazu izvedbe projekta. Funkcionalni zahtjevi, obrasci uporabe, sekvencijski dijagrami, model baze podataka i dijagram razreda bili su veoma koristan alat kojim smo rješavali implementacijske nedoumice.

Druga faza projekta sadržavala je u najvećem dijelu programiranje i implementaciju dokumentirane web aplikacije. Ta faza zahtjevala je poseban samostalni angažman članova tima koji se dotada nisu susreli s tehnologijama koje smo odlučili koristiti, ali i angažman članova tima koji su s tim tehnologijama bili upoznati kako bi pomogli kolegama da ih što uspješnije savladaju. Tijekom tog vremena vladao je visok intenzitet rada i odlična timska kohezija. Isto tako, bilo je potrebno napisati preostalu dokumentaciju i izraditi UML dijagrame.

Pri izradi projekta mnogo toga smo naučili, od korištenja novih tehnologija, alata, usavršavanja programskih jezika, izrade dokumentacije do važnosti i koristi koje nam pruža timski rad, kolegijalni i dobri timski odnosi, organiziran i marljiv voditelj tima koji koordinira svim aktivnostima, ali i vrijedni članovi koji prihvaćaju sugestiju te teže napretku u svakom smislu. Također smo naučili vrijednost kontinuiranog rada koji nam je olakšao da sve zadatke obavimo na vrijeme i bez panike. Izuzetno smo zadovoljni izrađenom web aplikacijom, ali smo isto tako svjesni da ima dosta mjesta za napredak i usavršavanje iste. Jedna od mogućnosti bi bila proširenje aplikacije za hotelske sustave ili samoposlužne praonice, ali i izrada mobilne aplikacije.

Popis literature

1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
2. Astah Community, <http://astah.net/editions/uml-new>
3. Nuxt documentation, <https://nuxtjs.org/>
4. Django documentation, <https://docs.djangoproject.com/en/3.1/>

Indeks slika i dijagrama

2.1	Slika aplikacije "UJLAUNDRY"	7
2.2	Slika aplikacije "Washing machine with a QR code"	8
2.3	Slika aplikacije za rezervaciju termina iz Norveške	8
3.1	Slika dijagrama obrasca uporabe registriranih i neregistriranih korisnika	25
3.2	Slika dijagrama obrasca uporabe zaposlenika i administratora	26
3.3	Slika dijagrama obrasca uporabe administratora	26
3.4	Sekvencijski dijagram za UC1 i UC2	28
3.5	Sekvencijski dijagram za UC3	29
3.6	Sekvencijski dijagram za UC4	30
3.7	Sekvencijski dijagram za UC11	31
4.1	Arhitektura sustava	33
4.2	Relacijska shema baze podataka	39
4.3	Relacijska shema Django baze podataka	40
4.4	Dijagram razreda - dio Models	42
4.5	Dijagram razreda - dio Views	43
4.6	Dijagram razreda - dio Templates	44
4.7	Dijagram stanja	46
4.8	Dijagram aktivnosti	48
4.9	Dijagram komponenti	49
5.1	Ispitni slučajevi vezani za prijavu korisnika	52
5.2	Ispitni slučajevi vezani za prijavu korisnika	53
5.3	Ispitni slučajevi vezani za vraćanje košara	54
5.4	Ispitni slučajevi vezani za rezervaciju termina, slika 1/2	55
5.5	Ispitni slučajevi vezani za rezervaciju termina, slika 2/2	55
5.6	Pokretanje svih ispitnih slučajeva u terminalu	56
5.7	Selenium konfiguracija za test broj 1	56
5.8	Selenium konfiguracija za test broj 2	57

5.9	Selenium konfiguracija za test broj 3	58
5.10	Selenium konfiguracija za test broj 4	59
5.11	Dijagram stanja	60
5.12	Mapa za frontend: terminko	61
5.13	Mapa za backend: terminko1	62
5.14	Konfiguracija za frontend	62
5.15	Buildpack za frontend	63
5.16	Procfile za frontend	63
5.17	Promjena axios base-URL	64
5.18	Postavljanje Postgres baze podataka na heroku	65
5.19	Procfile na backendu	65
5.20	Postavljanje Postgres konfiguracije u settings.py	66
5.21	Postavljanje cors whitelist u settings.py	66
5.22	Postavljanje allowed hosts varijable u settings.py	66
5.23	Prikaz stanja na heroku po završetku deploja	67
6.1	Contributors	76
6.2	Contributors	77
6.3	Contributors	77

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak

- Datum: 2. listopada 2020.
- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, L. Inkret, J. Grgić
- Teme sastanka:
 - smišljanje teme za projekt

2. sastanak

- Datum: 15. listopada 2020.
- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, L. Inkret, J. Grgić
- Teme sastanka:
 - rasprava o funkcionalnostima sustava
 - rasprava o bazi podataka

3. sastanak

- Datum: 26. listopada 2020.
- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, J. Grgić
- Teme sastanka:
 - rasprava o funkcionalnostima sustava
 - rasprava o bazi podataka
 - promijene i dodaci u do sad napisanom dokumentu vezane za bazu podataka, funkcionalne i nefunkcionalne zahtjeve i obrazaca uporabe

4. sastanak

- Datum: 29. listopada 2020.
- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, J. Grgić

- Teme sastanka:
 - rasprava bazi podataka
 - rasprava o dijagramima obrasca uporabe
 - prijedlozi za dodatke i promijene u dokumentu

5. sastanak

- Datum: 7. studenoga 2020.
- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, J. Grgić
- Teme sastanka:
 - dovršavanje generičkih funkcionalnosti
 - provjera rada klijentske strane aplikacije
 - provjera rada poslužiteljske strane aplikacije

6. sastanak

- Datum: 20. studenoga 2020.
- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, J. Grgić
- Teme sastanka:
 - raspodjela poslova oko izrade programske potpore
 - dogovor oko baze podataka

7. sastanak

- Datum: 4. prosinca 2020.
- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, J. Grgić
- Teme sastanka:
 - provjera i dogovor oko izrada funkcionalnosti za predaju alfa inačice
 - ispravak grešaka u kodu

8. sastanak

- Datum: 18. prosinca 2020.
- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, J. Grgić
- Teme sastanka:
 - priprema aplikacije za deploy
 - odabir načina deploja

9. sastanak

- Datum: 8. siječnja 2020.

- Prisustvovali: I. Joskić, D. Grgić, B. Spiegl, D. Šmigovec, M. Dragošević, J. Grgić
- Teme sastanka:
 - dogovor oko pisanja dokumentacije i pripreme za predaju

Tablica aktivnosti

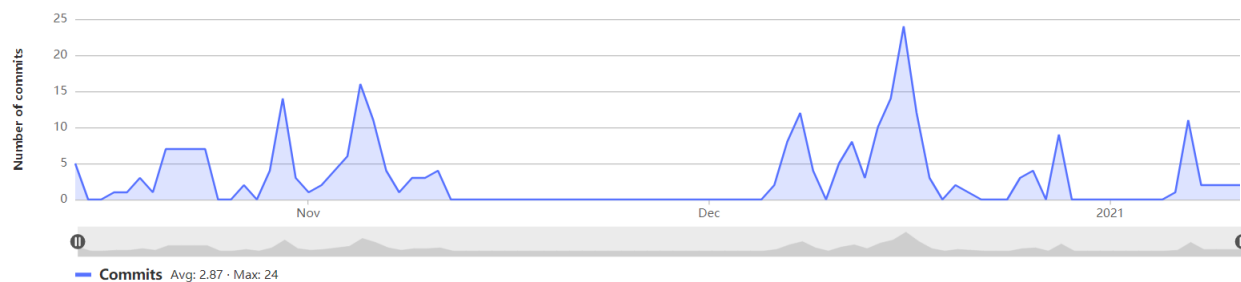
	Jan Grgić	Dunja Šmigovec	Marija Dragošević	Bernard Spiegl	Dino Grgić	Ivan Joskić	Leonard Inkret
Upravljanje projektom	10						
Opis projektnog zadatka	5	2	1:30	2	2	2	
Funkcionalni zahtjevi	1	0:30	1	1	0:30	1:30	
Opis pojedinih obrazaca	3:30	1	3	3	1	5	
Dijagram obrazaca	1:30	1	1	2:30	1	1	
Sekvencijski dijagrami		3				4	
Opis ostalih zahtjeva					1		
Arhitektura i dizajn sustava					4		
Baza podataka	3	7:30	0:30	0:30	0:30	0:30	
Dijagram razreda					8		
Dijagram stanja							
Dijagram aktivnosti			1:30				
Dijagram komponenti				2			
Korištene tehnologije i alati		1					
Ispitivanje programskog rješenja	3				2		
Dijagram razmještaja							
Upute za puštanje u pogon	10						
Dnevnik sastajanja	1						
Zaključak i budući rad			1:30				
Popis literature							
<i>izrada frontenda</i>	12	15	32	25	16	10	
<i>izrada backenda</i>	56				2		
<i>spajanje frontenda s back endom</i>	10		2		3		

Dijagrami pregleda promjena

U nastavku su slike dijagrama promjena preuzete s gitlaba. Na nekim se slikama pojavljuju isti studenti s različitim mailom jer su pushali s više različitih računa.

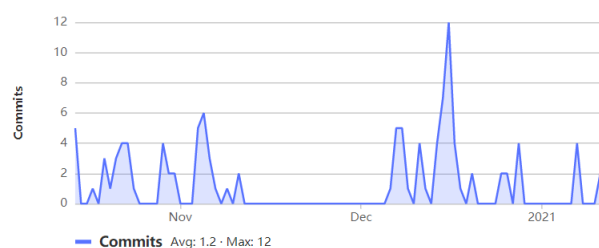
Commits to master

Excluding merge commits. Limited to 6,000 commits.



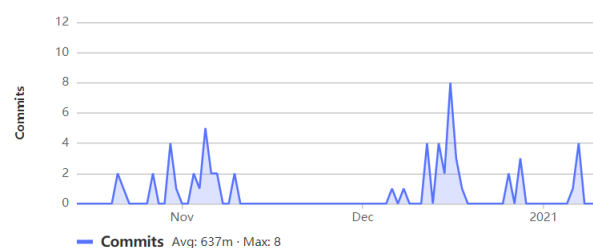
Jan Grgić

109 commits (jan.grgic@fer.hr)



bronemos

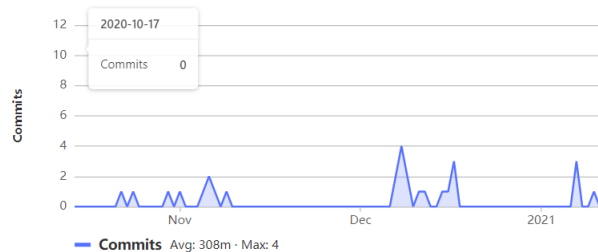
58 commits (spieglb@gmail.com)



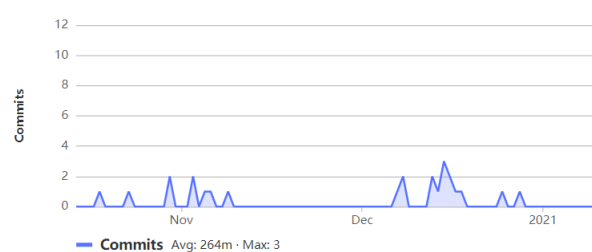
Slika 6.1: Contributors

MarijaDragosevic

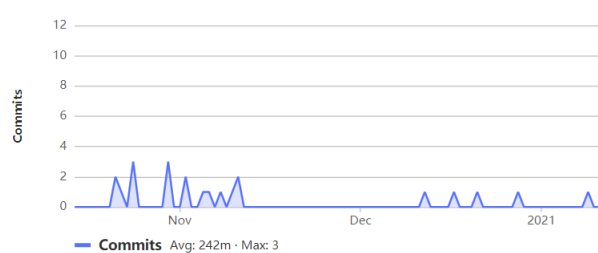
28 commits (marija.dragosevic@fer.hr)

**dinogrgic1**

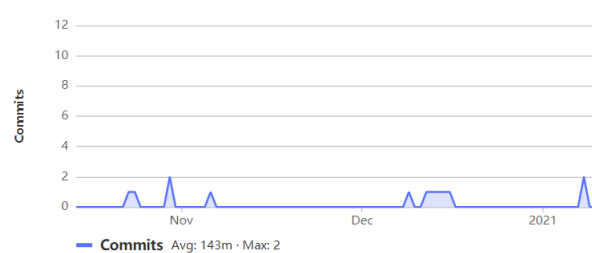
24 commits (dino.grgic1@gmail.com)

**IvanJoskic**

22 commits (ij51735@fer.hr)

**dsmigovec**

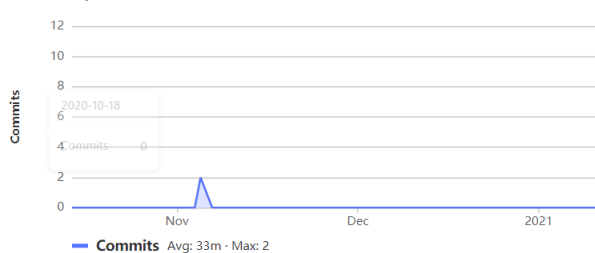
13 commits (dunja.smigovec@fer.hr)



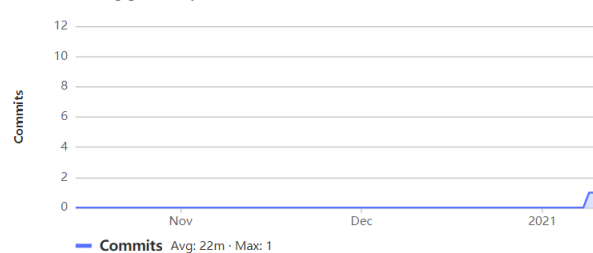
Slika 6.2: Contributors

IvanJoskic

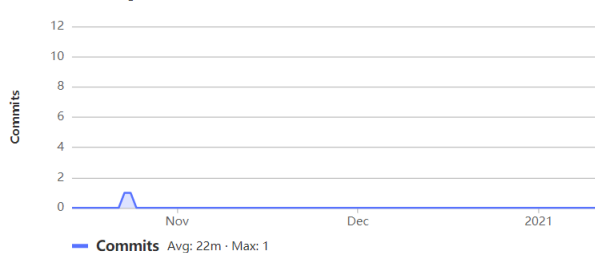
3 commits (ijoskic@outlook.com)

**dinogrgic**

2 commits (dino.grgic@unitfly.com)

**FER\dg51627**

2 commits (Dino.Grgic@fer.hr)



Slika 6.3: Contributors