

Oblikovanje programske potpore

Ak. god. 2017./2018.

Dokumentacija, Rev. 2

Grupa: *Kavon*

Voditelj: *Filip Sodić*

Datum predaje: *18. siječnja 2017.*

Nastavnik: *Miljenko Krhen*

## Sadržaj

1.	Dnevnik promjena dokumentacije .....	6
2.	Opis projektnog zadatka.....	8
2.1	Projektni prijedlog.....	8
2.2	Zahtjevi za informacijski sustav.....	8
2.2.1	Korisnici informacijskog sustava i njihove ovlasti/mogućnosti .....	8
2.2.2	Posebne funkcionalnosti sustava .....	9
2.3	Ostale funkcionalnosti sustava.....	10
2.4	Cilj i dugoročna korist rješenja .....	10
2.5	Potencijalni kupci i korisnici.....	11
3.	Pojmovnik.....	12
4.	Funkcijski zahtjevi.....	13
4.1	Dionici i aktori.....	13
4.2	Opisi obrazaca uporabe .....	13
4.2.1	UC1 – Registracija.....	13
4.2.2	UC2 – DodavanjeZaposlenika .....	14
4.2.3	UC3 – Prijava .....	14
4.2.4	UC4 – PregledIUređivanjePodataka.....	14
4.2.5	UC5 – UklanjanjeZaposlenika .....	15
4.2.6	UC6 – BrisanjeKorisnika.....	15
4.2.7	UC7 – DodavanjeLjubimca.....	16
4.2.8	UC8 – BrisanjeLjubimca .....	16
4.2.9	UC9 – StvaranjeRezervacije.....	16
4.2.10	UC10 – UređivanjeRezervacije.....	17
4.2.11	UC11 – PraćenjeRezervacije.....	17
4.2.12	UC12 - PrihvatanjeRezervacije .....	18
4.2.13	UC13 – PotvrdaPlaćeneRezervacije .....	18
4.2.14	UC14 – OtkazivanjeRezervacije .....	19
4.2.15	UC15 – SlanjePodsjetnika .....	19
4.2.16	UC16 – PregledRezervacija .....	19
4.2.17	UC17 – Odjava .....	20
4.3	Dijagrami obrazaca uporabe .....	21
4.4	Sekvencijski dijagrami.....	24

4.4.1 Obrazac uporabe UC1 (Registracija).....	24
4.4.2 Obrazac uporabe UC2 (DodavanjeZaposlenika) .....	25
4.4.3 Obrazac uporabe UC3 (Prijava) .....	26
4.4.4 Obrazac uporabe UC4 (PregledIUređivanjePodataka).....	27
4.4.5 Obrazac uporabe UC5 (UklanjanjeZaposlenika) .....	28
4.4.6 Obrazac uporabe UC6 (BrisanjeKorisnika).....	29
4.4.7 Obrazac uporabe UC7 (DodavanjeLjubimca) .....	30
4.4.8 Obrazac uporabe UC8 (BrisanjeLjubimca) .....	31
4.4.9 Obrazac uporabe UC9 (StvaranjeRezervacije).....	32
4.4.10 Obrazac uporabe UC10 (UređivanjeRezervacije).....	33
4.4.11 Obrazac uporabe UC11 (PraćenjeRezervacije).....	34
4.4.12 Obrazac uporabe UC12 (PrihvaćanjeRezervacije) .....	35
4.4.13 Obrazac uporabe UC13 (PotvrdaPlaćeneRezervacije) .....	36
4.4.14 Obrazac uporabe UC14 (OtkazivanjeRezervacije) .....	37
4.4.15 Obrazac uporabe UC15 (SlanjePodsjetnika) .....	38
4.4.16 Obrazac uporabe UC16 (PregledRezervacija) .....	39
4.4.17 Obrazac uporabe UC17 (Odjava) .....	40
5. Nefunkcionalni zahtjevi .....	41
6. Arhitektura i dizajn sustava.....	42
6.1 Svrha, opći prioriteti i skica sustava .....	42
6.1.1 Web poslužitelj .....	42
6.1.2 Web aplikacija .....	42
6.1.3 Baza podataka.....	43
6.2 Dijagrami razreda.....	46
6.2.1 Razredi modela .....	46
6.2.2 Razredi upravitelja .....	48
6.2.3 Repozitoriji.....	49
6.2.4 Razredi pretvarači .....	50
6.2.5 Razredi modela prezentacijskog sloja .....	51
6.2.6 Razredi modela formulara .....	52
6.2.7 Razredi usluga.....	52
6.2.8 Validacijski razredi .....	53
6.2.9 Konfiguracijski razredi .....	54
6.3 Dijagram objekata .....	55
6.4 Ostali UML dijagrami.....	56

6.4.1 Dijagrami stanja.....	56
6.4.2 Komunikacijski dijagram .....	59
6.4.3 Dijagram aktivnosti .....	59
6.4.4 Dijagram komponenti.....	60
7. Implementacija i korisničko sučelje.....	62
7.1 Dijagram razmještaja .....	62
7.2 Korištene tehnologije i alati .....	62
7.2.1 Eclipse.....	62
7.2.2 IntelliJ IDEA.....	62
7.2.3 Spring .....	63
7.2.4 MySQL Community Server.....	63
7.2.5 Hibernate ORM.....	63
7.2.6 JSON.....	63
7.2.7 Apache Tomcat .....	63
7.2.8 Semantic UI .....	63
7.2.9 jQuery.....	64
7.3 Isječak programskog koda vezan za temeljnu funkcionalnost sustava .....	65
7.3.1 Kod potreban za funkcionalnost Spring-Boot aplikacije.....	65
7.3.2 Primjer Upravitelja .....	67
7.4 Primjer repozitorija .....	68
7.4.1 Primjer servisa.....	69
7.5 Ispitivanje programskog rješenja.....	69
7.6 Upute za instalaciju .....	76
7.6.1 Instalacija programske okoline Java.....	76
7.6.2 Instalacija upravitelja projektom <i>Maven</i> .....	76
7.6.3 Instalacija i konfiguracija baze podataka .....	76
7.6.4 Pokretanje aplikacije.....	77
7.7 Korisničke upute.....	77
7.7.1 Pregled klijenta .....	77
7.7.2 Pregled zaposlenika .....	83
7.7.3 Pregled administratora .....	83
8. Zaključak i budući rad.....	87
9. Popis literature .....	88
10. Dodatak A: Indeks .....	89
11. Dodatak B: Dnevnik sastanaka.....	91

12.	Dodatak C: Prikaz aktivnosti grupe .....	96
12.1	Tablični prikaz sudjelovanja .....	96
12.2	Grafički prikaz aktivnosti grupe .....	99
13.	Dodatak D: Plan rada i ostvarenje .....	100
13.1	Ostvareno u rev. 1:.....	100
13.2	Ostvareno u rev. 2:.....	100
13.3	Plan za daljnji razvoj: .....	100

# 1. Dnevnik promjena dokumentacije

REV.	OPIS PROMJENA	AUTORI	DATUM
0.1	-napravljen inicijalni predložak -dodan projektni zadatak	Modrušan	25.10.2017.
0.2	-dodani početni obrasci upotrebe (Use Case)	Kravrščan	25.10.2017.
0.3	-dodan dnevnik sastajanja -naknadno ažuriranje i uređivanje dnevnika sastajanja	Paulinović, Galić	26.10.2017.
0.4	-dodan prijedlog modela baze podataka	Krmpotić-Đurđević	27.10.2017.
0.5	-završeni obrasci upotrebe (Use Case) -dodani sekvensijski dijagrami	Kravrščan, Paulinović	5.11.2017.
0.6	-prebacivanje, do sad napravljene, dokumentacije na Gitlab Wiki	Krmpotić-Đurđević, Modrušan, Galić	10.11.2017.
0.7	-dovšeni svi sekvensijski dijagrami i opisi	Paulinović, Galić	14.11.2017.
0.8	-arhitektura sustava (MVC) -dorađen ER model baze podataka	Sodić, Krmpotić-Đurđević	15.11.2017.
1.0	-kontrola dokumentacije -ispravljene pogreške i dodani detalji -konačna prva verzija dokumentacije	Galić, Paulinović, Sodić, Kravrščan, Krmpotić-Đurđević, Modrušan	17.11.2017.
1.1	-napravljeni dijagrami stanja	Galić	5.12.2017.
1.2	-napravljeni komunikacijski dijagrami	Modrušan	20.12.2017.

REV.	OPIS PROMJENA	AUTORI	DATUM
1.3	-napravljeni dijagrami aktivnosti	Paulinović, Sodić, Krmpotić-Đurđević	21.12.2017.
1.4	-završetak poglavlja "Ostali zahtjevi"	Krmpotić-Đurđević	5.1.2018.
1.5	-dijagram razmještaja -korištene tehnologije i alati	Paulinović, Galić	10.1.2018.
1.6	-isječak programskog koda vezan za temeljnu funkcionalnost sustava	Kravriščan	13.1.2018.
1.7	-ispitivanje programskog rješenja	Krmpotić-Đurđević	14.1.2018.
1.8	-upute za instalaciju -koricničke upute	Sodić, Modrušan	17.1.2018.
2.0	-finalni pregled i uređivanje komplentne dokumentacije, popravljanje grešaka	Galić, Paulinović, Sodić, Kravriščan, Krmpotić-Đurđević, Modrušan	17.1.2018.

## 2. Opis projektnog zadatka

### 2.1 Projektni prijedlog

Tvrtka za čuvanje kućnih ljubimaca nudi sljedeće usluge:

- Dolazak/odlazak po psa na čuvanje
- Odlazak veterinaru
- Šetnje
- Čuvanje pasa
- Hranjenje mačaka
- Čuvanje egzota, glodavaca, dvojezubaca
- Davanje terapije
- Edukacija
- Rezanje noktiju

Za tvrtku je potrebno razviti učinkovit informacijski sustav ostvaren u obliku web aplikacije. Sustav treba zaposlenicima omogućiti lakšu i kvalitetniju organizaciju posla, a klijentima pružiti opciju online rezervacija i naručivanja usluga. Aplikacija uključuje i intranet koji koriste zaposlenici.

Radno vrijeme tvrtke nije fiksno. Prijava ljubimaca za usluge aktivna je uvijek, a oni se zaprimaju/zbrinjavaju ovisno o dostupnosti. Preuzimanje ljubimaca također se vrši u dogovorenim terminima.

### 2.2 Zahtjevi za informacijski sustav

#### 2.2.1 Korisnici informacijskog sustava i njihove ovlasti/mogućnosti

Informacijski sustav razlikuje tri vrste korisnika: registrirani klijent, zaposlenik i administrator.

Registrirani klijent može rezervirati usluge slanjem zahtjeva. Sustav pamti njegove podatke i podatke o njegovim ljubimcima. Postupak i zahtjev za registraciju klijenata opisani su u odjeljku 2.2.1, a njegov pregled u odjeljku 2.2.2.

Zaposlenik može mijenjati određeni dio vlastitih informacija i svoju lozinku. Registracija zaposlenika opisana je u odjeljku 2.2.5, a pregled i ovlasti u odjeljku 2.2.6.

Administrator sustava također je zaposlenik, ali s najvećom razinom autorizacije. Njegov pregled i ovlasti opisani su u odjeljku 2.2.8.

## 2.2.2 Posebne funkcionalnosti sustava

### 2.2.2.1 Registracija klijenta

Klijent prilikom registracije unosi vlastite podatke. Obavezni podaci su njegovo ime, prezime, OIB, adresa stanovanja i elektroničke pošte te broj telefona. Također se nudi i opcija unošenja pojedinosti o jednom ili više kućnih ljubimaca s ciljem ubrzanja korištenja aplikacije u budućnosti. Ovisno o ljubimcu, sustav može tražiti neke dodatne obavezne informacije. Uz sve obavezne i neobavezne podatke, klijent mora izabrati i lozinku.

Kao korisničko ime koristiti će se unesena adresa elektroničke pošte koja je jedinstvena za svakog klijenta.

### 2.2.2.2 Pregled registriranog klijenta

Klijent vidi sve svoje zahteve i njihove trenutne razine. Postupak zaprimanja, određivanja razina i obrade zahtjeva detaljnije je objašnjen u odjeljcima 2.2.3 i 2.2.7.

Klijent može mijenjati dio svojih podataka i lozinku.

### 2.2.2.3 Prijava ljubimaca za usluge

Nakon spajanja na sustav, registrirani klijent ima mogućnost rezervacije željene usluge. Rezervacija usluge zahtjeva precizno određivanje njenog termina i vrste. Ovisno o odabranoj usluzi, nudi se i preferencijalni odabir zaposlenika kod kojeg klijent želi zbrinuti svog ljubimca. Registrirani klijenti ovdje imaju pristup svojim uobičajenim podacima. Jednom kad klijent uneše sve potrebne informacije, može poslati zahtjev čime se razina tog zahtjeva postavlja na 1. Dostupna je dodatna opcija podsjetnika koja je detaljno opisana u odjeljku 2.2.4.

Nakon slanja obrasca, klijent u svom pregledu vidi podneseni zahtjev. Uz zahtjev je navedena i njegova trenutna razina (od 1 do 3), što je pojašnjeno detaljnije u odjeljku 2.2.7. Klijent može mijenjati zahtjev dok god je on na razini 1.

### 2.2.2.4 Podsjetnik na zakazani termin usluge

Nakon uspješnog završetka prijave, zahtjev prolazi kroz proceduru opisanu u odjeljku 2.2.7. Čim zahtjev dosegne razinu broj 2, klijent dobija e-mail s ponudom. Ukoliko je klijent pri slanju zahtjeva odabrao da želi podsjetnik, sustav mu ga automatski šalje u određeno vrijeme prije termina usluge, također preko adresu elektroničke pošte.

### 2.2.2.5 Registracija zaposlenika

Svi zaposlenici u sustav se prijavljuju kao klijenti i inicialno imaju samo ovlasti klijenta, zaposliti ih može isključivo administrator. Za svakog se od njih uz osnovne podatke mogu navesti dodatni atributi poput životinja koje je pojedini zaposlenik spreman preuzeti te usluga koje je spreman obaviti.

Nakon što ga administrator zaposli, svaki zaposlenik ima opciju definiranja vremena u kojem nije u mogućnosti ili ne želi preuzeti posao (dalje u tekstu kao interval nedostupnosti).

Zaposleniku se na odabir nudi hoće li elektroničkom poštrom primati obavijesti za sve poslove, samo za poslove u kojima je naveden kao preferencijalni odabir ili nikada.

### 2.2.2.6 Pregled i ovlasti zaposlenika

Nakon uspješne prijave u sustav, zaposleniku se nude tri različita pregleda poslova:

- *Nepreuzeti poslovi* (zahtjevi razine 1)
- *Poslovi na čekanju* (zahtjevi razine 2)
- *Zaključeni poslovi* (zahtjevi razine 3)

Pregled *Nepreuzeti poslovi* zaposleniku prikazuje sve zadatke koje može preuzeti. Ovdje se mogu naći samo zadaci čije se vrijeme ne preklapa ni s intervalom nedostupnosti ni s vremenom bilo kojeg već prihvaćenog zadatka od strane tog zaposlenika. Prikazuju se svi podaci o zadatku koje je registrirani klijent upisao kao i podaci o dotičnom klijentu. Postupak preuzimanja poslova i obrade zahtjeva opisan je u odjeljku 2.2.7.

### 2.2.2.7 Postupak obrade zahtjeva

Kada klijent zatraži uslugu, taj posao se stavlja u pregled *Nepreuzeti poslovi*. Nakon što neki zaposlenik preuzme posao, on nestaje iz pregleda *Nepreuzeti poslovi* (svim zaposlenicima) te se prebacuje u pregled *Poslovi na čekanju* (samo zaposleniku koji je taj posao preuzeo). U ovom trenutku, klijent o čijem je zadatku riječ dobiva e-mail s ponudom te dotični zahtjev dobiva oznaku razine 2.

Nakon što klijent plati, šalje potvrdu o plaćanju administratoru koji mora odobriti zadatak. Nakon odobrenja, zaposleniku se zadatak premješta iz pregleda *Poslovi na čekanju* u pregled *Zaključeni poslovi*, te sam zahtjev dobiva oznaku razine 3.

### 2.2.2.8 Pregled i ovlasti Administratora

Administrator ima sve ovlasti nad sustavom:

- Dodavanje, brisanje i uređivanje zaposlenika
- Dodavanje, brisanje i uređivanje klijenata
- Dodavanje, brisanje i uređivanje zadataka
- Uvid u sve zadatke na svim razinama za sve zaposlenike

Administrator jedini može unaprijediti zahtjev na razinu broj 3 i to tek nakon što mu klijent pošalje potvrdu o plaćanju. Ima i opciju odbijanja/otkazivanja zahtjeva bilo koje razine. Za sve odbijene zahtjeve, sustav može automatski poslati odbijenicu putem elektroničke pošte. Administrator može u ime klijenata otvarati nove zahtjeve u slučajevima kada se netko naruči telefonom ili e-mailom, a ne putem web aplikacije.

## 2.3 Ostale funkcionalnosti sustava

Sustav mora omogućiti istovremeni rad administratora, zaposlenika i neograničenog broja registriranih klijenata.

## 2.4 Cilj i dugoročna korist rješenja

Cilj izrade web aplikacije je olakšati i ubrzati proces rezervacije usluga te pojednostavljenje komunikacije između zaposlenika tvrtke. Uz web aplikaciju povećanjem broja klijenata i zaposlenika neće se mijenjati kompleksnost organizacije

posla kao što bi to bio slučaj da se nastavi poslovati korištenjem e-maila i putem telefona.

## **2.5 Potencijalni kupci i korisnici**

Dosadašnje korisnike uputiti će se na novi način naručivanja usluga, odnosno korištenje web aplikacije, a modernizacija načina poslovanja mogla bi dovesti nove korisnike i tako povećati promet tvrtke.

### 3. Pojmovnik

**Java:** Objektno orijentiran jezik nezavisan o platformi koji se može koristiti za izradu web aplikacija.

**UML (Unified Modeling Language):** razvojni jezik modela opće svrhe čiji je cilj slikovno predstavljanje dizajna sustava.

**SQL:** Najpopularniji jezik za rad s bazom podataka. Objedinjuje funkcije jezika za definiciju podataka i jezika za rukovanje podacima.

**Git:** Distribuirani jezik za upravljanje izvornim kodom.

**Spring Framework:** Spring aplikacijski okvir može se koristiti za izradu web aplikacija u Java EE platformi.

**Eclipse:** Programska razvojna okolina (IDE) koja se najčešće koristi razvoj aplikacija u Javi. Sastoji se od osnovnog radnog prostora te ima mogućnost dodavanja raznih proširenja za uređivanje okoline.

**JavaScript:** Skriptni jezik koji se koristi u pretraživačima s ciljem stvaranja dinamičkog sadržaja.

**Model-view-controller:** Programska arhitektura za implementaciju korisničkih sučelja na računalima. Aplikacija se dijeli na tri međusobno povezana dijela kako bi se odvojila unutarnja reprezentacija informacije od načina na koji je ta informacija prikazana korisniku.

**JSON (JavaScript Object Notation):** Tekstualni format potpuno neovisan o programskom jeziku koji se često koristi za komunikaciju pretraživača sa poslužiteljem.

**AJAX (Asynchronous Javascript And XML):** Skup tehnika za razvoj web stranica koji omogućuje asinkrono slanje i primanje podataka sa poslužitelja tako da se sadržaj stranice može izmijeniti bez potrebe da se stranica ponovo učita.

## 4. Funkcijski zahtjevi

### 4.1 Dionici i aktori

Dionici:

- Vlasnik tvrtke
- Zaposlenici tvrtke
- Klijenti tvrtke
- Administratori web aplikacije
- Inženjeri koji održavaju sustav

Aktori i njihovi funkcionalni zahtjevi

- Administrator (inicijator)
- Klijent (inicijator)
- Zaposlenik (inicijator)
- Baza podataka (sudionik) - pohranjuje podatke o:
  - registriranim klijentima
  - registriranim zaposlenicima
  - postojećim uslugama
  - ljubimcima
- Neregistrirani korisnik (inicijator) - može se registrirati na web mjesto

### 4.2 Opisi obrazaca uporabe

#### 4.2.1 UC1 – Registracija

- **Glavni sudionik:** neregistrirani korisnik
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** -
- **Cilj:** dodati novog klijenta u sustav
- **Rezultat:** klijent dodan u sustav
- **Željeni scenarij:**
  1. Neregistrirani korisnik odabire opciju za registraciju
  2. Neregistrirani korisnik upisuje sve potrebne podatke
  3. Provjerava se ispravnost podataka
  4. Novonastali korisnik pohranjuje se u bazu podataka i inicijalno dobiva ulogu klijenta
- **Mogući drugi scenarij:**

1. Korisnik s unesenom adresom elektroničke pošte već postoji u bazi podataka
2. Sustav javlja poruku o grešci

#### 4.2.2 UC2 – DodavanjeZaposlenika

- **Glavni sudionik:** administrator
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** administratorske ovlasti, administrator se nalazi u pregledu podataka korisnika (UC4)
- **Cilj:** dodati novog zaposlenika u sustav
- **Rezultat:** korisnik zaposlen
- **Željeni scenarij:**
  1. Administrator odabire opciju za zapošljavanje korisnika
  2. Uloga korisnika promiče se u ulogu zaposlenika

#### 4.2.3 UC3 – Prijava

- **Glavni sudionik:** korisnik (administrator/zaposlenik/klijent)
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** korisnik postoji u sustavu
- **Cilj:** prijava korisnika u sustav
- **Rezultat:** korisnik prijavljen u sustav
- **Željeni scenarij:**
  1. Korisnik odabere opciju prijave u sustav
  2. Korisnik upisuje adresu elektroničke pošte i lozinku
  3. Poslužitelj provjerava podatke pomoću baze podataka
  4. Korisnik je prijavljen u sustav, započinje sjednica
- **Mogući drugi scenarij:**
  1. Podaci su pogrešni
  2. Poslužitelj obavijesti korisnika o grešci i traži ga ponovni unos podataka

#### 4.2.4 UC4 – PregledIUređivanjePodataka

- **Glavni sudionik:** korisnik
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** korisnik je prijavljen u sustav
- **Cilj:** pregled/promjena podataka korisnika

- **Rezultat:** podaci korisnika su promijenjeni/pregledani
- **Željeni scenarij:**
  1. Korisnik odabire opciju pregleda svojih podataka
  2. Korisniku se za neke podatke nudi i mogućnost izmjene
  3. Poslužitelj provjerava jesu li izmijenjeni podaci u odgovarajućem formatu
  4. Novi se podaci upisuju u bazu podataka
- **Mogući drugi scenarij:**
  1. Dio podataka je u pogrešnom formatu te poslužitelj obavještava korisnika o grešci

#### 4.2.5 UC5 – UklanjanjeZaposlenika

- **Glavni sudionik:** administrator
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** administratorske ovlasti, administrator se nalazi u pregledu podataka zaposlenika (UC4)
- **Cilj:** maknuti korisniku ulogu zaposlenika
- **Rezultat:** zaposlenik više nije u sustavu/bazi
- **Željeni scenarij:**
  1. Administrator odabere opciju uklanjanja uloge zaposlenika
  2. Korisnik gubi ulogu zaposlenika
- **Mogući drugi scenarij:**
  1. Zaposlenik trenutno ima dodijeljenu potvrđenu uslugu
  2. Sustav javlja poruku o grešci

#### 4.2.6 UC6 – BrisanjeKorisnika

- **Glavni sudionik:** korisnik
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** Klijent je prijavljen i nalazi se u pregledu vlastitih podataka (UC4)
- **Cilj:** obrisati korisnika iz sustava
- **Rezultat:** Korisnik više nije u sustavu/bazi
- **Željeni scenarij:**
  1. Korisnik odabere opciju brisanja sebe iz sustava
  2. Korisnik se obriše iz sustava

- **Mogući drugi scenarij**

1. Korisnik ima ulogu zaposlenika te sustav javlja poruku o nemogućnosti brisanja dok se ona ne oduzme

#### **4.2.7 UC7 – DodavanjeLjubimca**

- **Glavni sudionik:** klijent

- **Sporedni sudionik:** poslužitelj, baza podataka

- **Preduvjeti:** klijent je prijavljen u sustav

- **Cilj:** dodati novog ljubimca klijentu

- **Rezultat:** klijent ima novog ljubimca u sustavu

- **Željeni scenarij:**

1. Klijent odabire opciju dodavanja novog ljubimca
2. Klijent unosi sve potrebne podatke o ljubimcu
3. Podaci o ljubimcu upisuju se u bazu podataka

- **Mogući drugi scenarij:**

1. Dio podataka je u pogrešnom formatu te poslužitelj obavještava korisnika o grešci

#### **4.2.8 UC8 – BrisanjeLjubimca**

- **Glavni sudionik:** klijent

- **Sporedni sudionik:** poslužitelj, baza podataka

- **Preduvjeti:** klijent je prijavljen u sustav

- **Cilj:** obrisati jednog od ljubimaca klijenta

- **Rezultat:** ljubimac je obrisan

- **Željeni scenarij:**

1. Klijent odabire opciju brisanja ljubimca
2. Sustav dohvaća sve ljubimce koji pripadaju klijentu
3. Klijent odabire i briše ljubimca
4. Ljubimac je obrisan

#### **4.2.9 UC9 – StvaranjeRezervacije**

- **Glavni sudionik:** klijent

- **Sporedni sudionik:** poslužitelj, baza podataka

- **Preduvjeti:** Klijent je prijavljen u sustav

- **Cilj:** stvoriti rezervaciju
- **Rezultat:** rezervacija stvorena
- **Željeni scenarij:**
  1. Klijent odabire opciju rezerviranja usluge
  2. Klijent ispunjava detalje rezervacije
  3. Rezervacija je stavljena u listu gdje je zaposlenici mogu preuzeti te postaje dostupna klijentu na praćenje
- **Mogući drugi scenarij:**
  1. Dio podataka je u pogrešnom formatu te poslužitelj obavještava korisnika o grešci

#### 4.2.10 UC10 – UređivanjeRezervacije

- **Glavni sudionik:** klijent
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** rezervacija još nije preuzeta, klijent se nalazi u pregledu za praćenje rezervacija (UC11)
- **Cilj:** promijeniti podatke rezervacije
- **Rezultat:** promijenjeni podaci rezervacije
- **Željeni scenarij:**
  1. Klijent odabire rezervaciju koju želi urediti
  2. Klijent uređuje rezervaciju
  3. Klijent sprema promjene
  4. Podaci su ažurirani
- **Mogući drugi scenarij:**
  1. Netko je u međuvremenu preuzeo rezervaciju ili dio podataka je u pogrešnom formatu
  2. Poslužitelj javlja grešku

#### 4.2.11 UC11 – PraćenjeRezervacije

- **Glavni sudionik:** klijent
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** postoje rezervacije za praćenje te je prijavljeni klijent njihov vlasnik
- **Cilj:** saznati stanje rezervacije
- **Rezultat:** Klijent je dobio informaciju o stanju rezervacije

- **Željeni scenarij:**

1. Klijent ulazi u pregled za praćenje rezervacija
2. Poslužitelj iz baze dohvaća popis svih rezervacija dotičnog klijenta te mu ih prikazuje
3. Klijent dobiva informacije o stanjima svih svojih rezervacija

#### 4.2.12 UC12 - PrihvaćanjeRezervacije

- **Glavni sudionik:** zaposlenik
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** prijavljeni zaposlenik se nalazi u pregledu rezervacija (UC16), rezervacija se nalazi u listi nepreuzetih rezervacija
- **Cilj:** prihvatiti termin rezervacije (rezervacija se premješta iz liste nepreuzetih u zaposlenikovu listu preuzetih rezervacija)
- **Rezultat:** termin rezervacije prihvaćen/ rezervacija se premješta u listu preuzetih rezervacija zaposlenika
- **Željeni scenarij:**
  1. Zaposlenik pregledava listu nepreuzetih rezervacija
  2. Zaposlenik odabire rezervaciju koju želi prihvatiti
  3. Rezervacija se prebacuje u vlastitu listu preuzetih rezervacija dotičnog zaposlenika
  4. Korisniku se šalje poruka s ponudom
  5. Podaci se upisuju u bazu podataka
- **Mogući drugi scenarij:**
  1. Netko je drugi u međuvremenu prihvatio rezervaciju koju je zaposlenik htio preuzeti
  2. Poslužitelj javlja zaposleniku da je rezervacija već preuzeta

#### 4.2.13 UC13 – PotvrdaPlaćeneRezervacije

- **Glavni sudionik:** administrator
- **Sporedni sudionik:** poslužitelj, baza podataka
- **Preduvjeti:** administratorske ovlasti, administrator se nalazi u pregledu rezervacija (UC16), postoji zaposlenik koji je rezervaciju preuzeo
- **Cilj:** potvrditi rezervaciju kako bi mogao započeti stvaran posao
- **Rezultat:** rezervacija potvrđena, posao započinje
- **Željeni scenarij:**

- Administrator odabire opciju potvrđivanja rezervacije te se ona prebacuje u listu potvrđenih rezervacija

#### 4.2.14 UC14 – OtkazivanjeRezervacije

- Glavni sudionik:** administrator
- Sporedni sudionik:** poslužitelj, baza podataka
- Preduvjeti:** administratorske ovlasti, administrator se nalazi u pregledu rezervacija (UC16)
- Cilj:** otkazati rezervaciju neovisno o tome u kojoj se fazi / listi nalazi
- Rezultat:** rezervacija je otkazana
- Željeni scenarij:**
  - Administrator odabere opciju otkazivanja neke od rezervacija
  - Rezervacija je otkazana i miče se iz svih pregleda

#### 4.2.15 UC15 – SlanjePodsjetnika

- Glavni sudionik:** poslužitelj
- Sporedni sudionik:** klijent, baza podataka
- Preduvjeti:** -
- Cilj:** podsjetiti Klijenta da ima zakazani termin
- Rezultat:** Klijent je informiran
- Željeni scenarij:**
  - Poslužitelj periodički (jednom dnevno) provjerava postoje li potvrđene rezervacije unutar iduća 24 sata za koje je klijent odabrao opciju slanja podsjetnika
  - Za svaku rezervaciju koja ispunjava navedeni uvjet, poslužitelj šalje podsjetnik klijentu putem elektroničke pošte.

#### 4.2.16 UC16 – PregledRezervacija

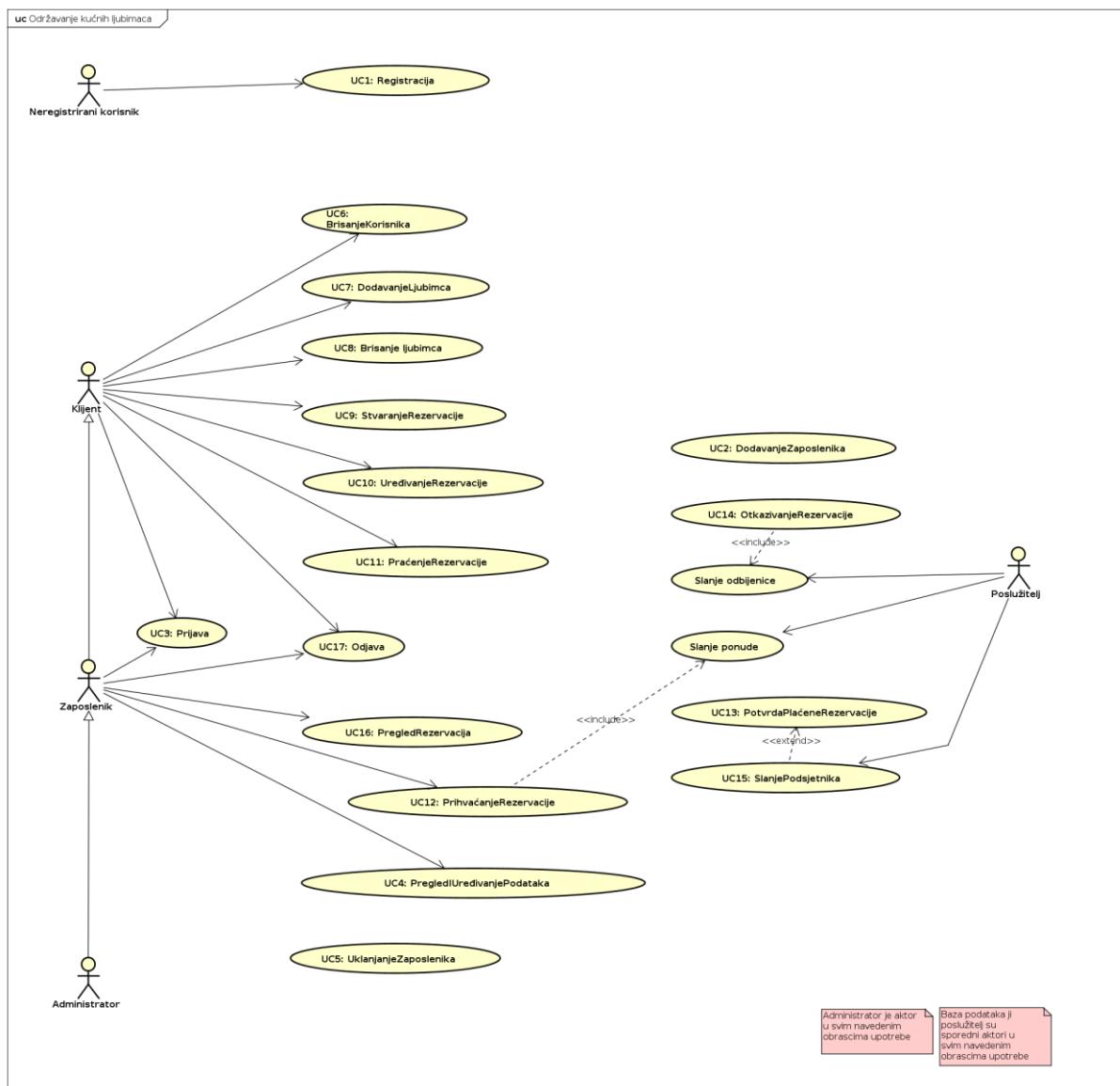
- Glavni sudionik:** zaposlenik
- Sporedni sudionik:** poslužitelj, baza podataka
- Preduvjeti:** zaposlenik je prijavljen u sustav
- Cilj:** pregledati postojeće rezervacije
- Rezultat:** -
- Željeni scenarij:**
  - Zaposlenik odabere opciju pregleda postojećih rezervacija
  - Zaposlenik vidi:

- Sve nepreuzete rezervacije u vremenu u kojem je dotični zaposlenik dostupan
- Sve rezervacije koje je on preuzeo, a još nisu plaćene
- Sve rezervacije koje je on preuzeo i plaćene su

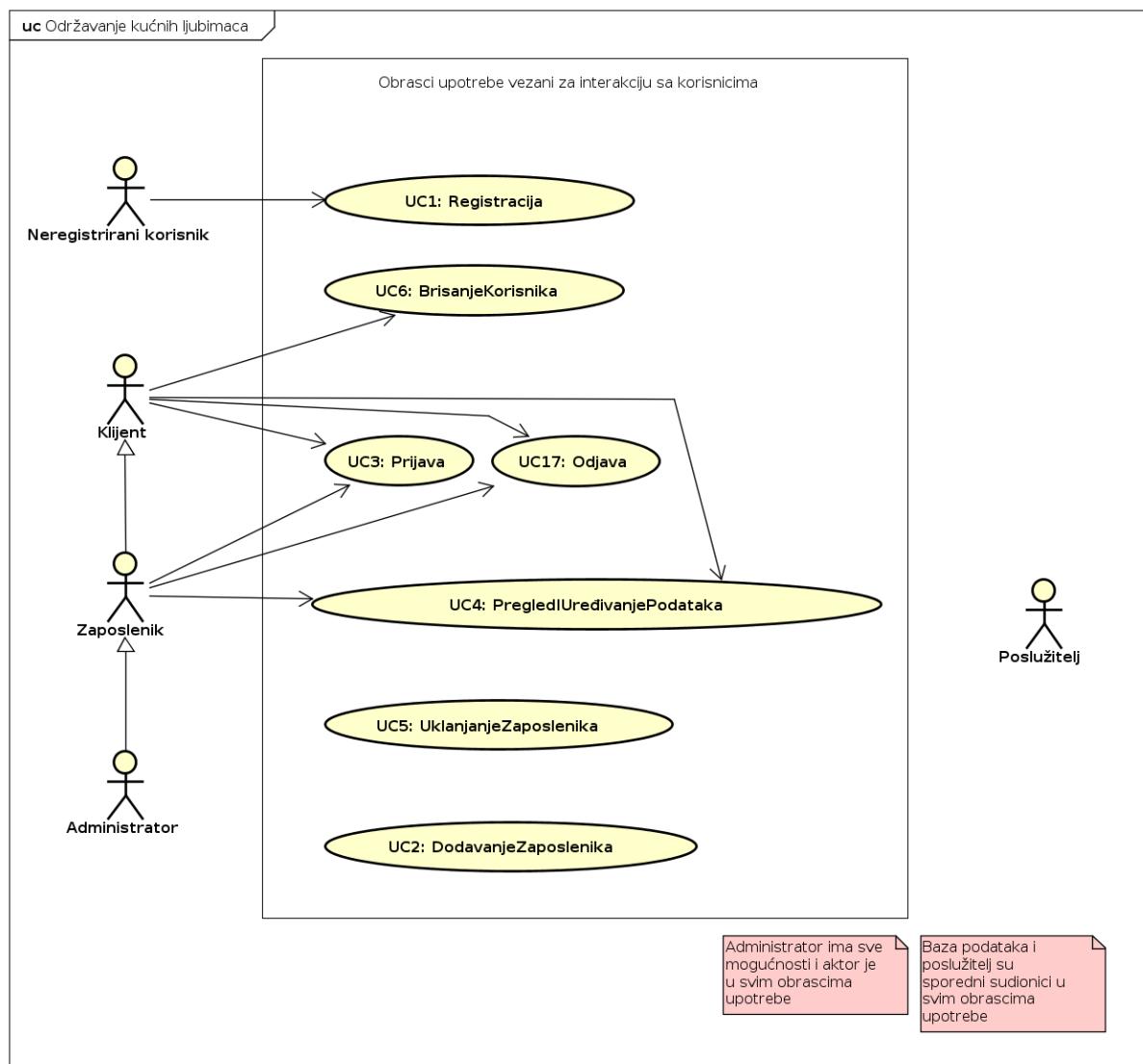
#### 4.2.17 UC17 – Odjava

- **Glavni sudionik:** korisnik
- **Sporedni sudionik:** poslužitelj
- **Preduvjeti:** korisnik je prijavljen u sustav
- **Cilj:** odjaviti se iz sustava
- **Rezultat:** korisnik više nije prijavljen, sjednica je zatvorena
- **Željeni scenarij:**
  1. Korisnik odabere opciju odjave iz sustava
  2. Poslužitelj odbacuje postojeću sjednicu i odjavljuje korisnika

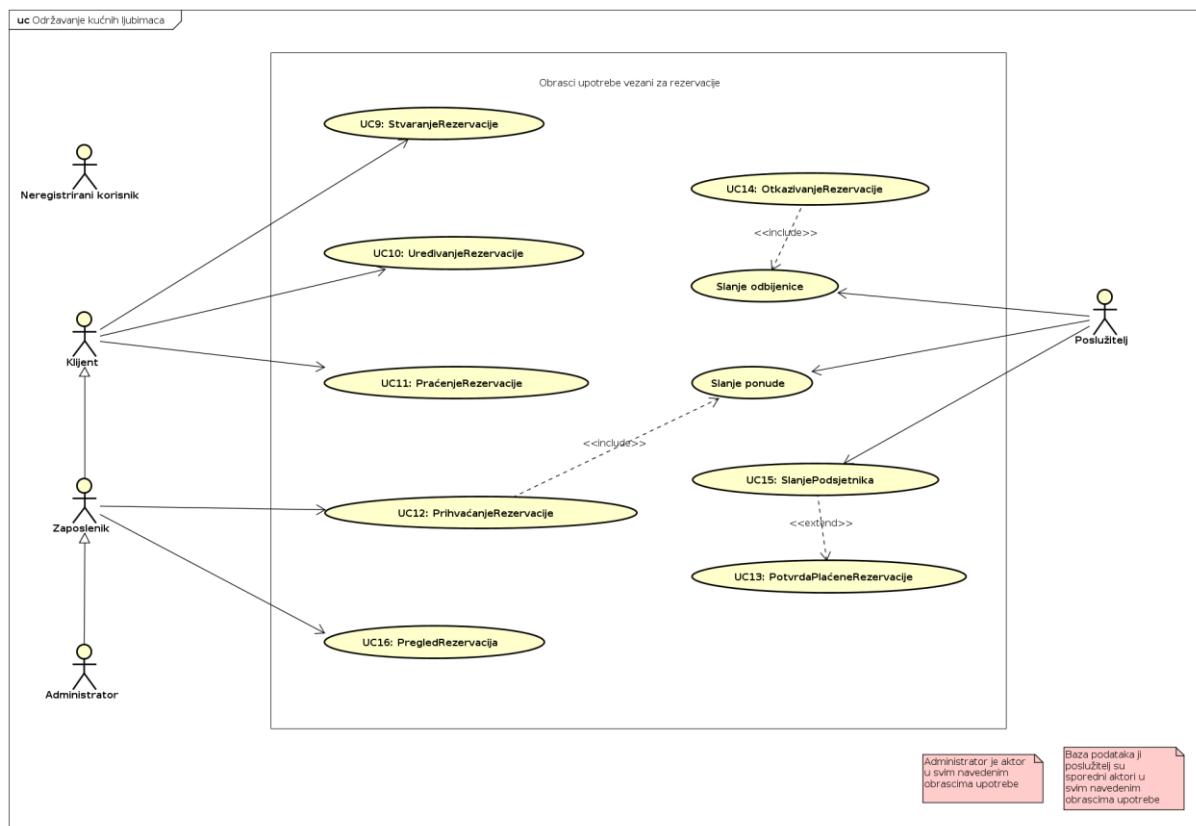
## 4.3 Dijagrami obrazaca uporabe



Slika 4.1 - Cjeloviti pregled svih obrazaca upotrebe



Slika 4.2 - Obrasci vezani za interakciju s korisnicima

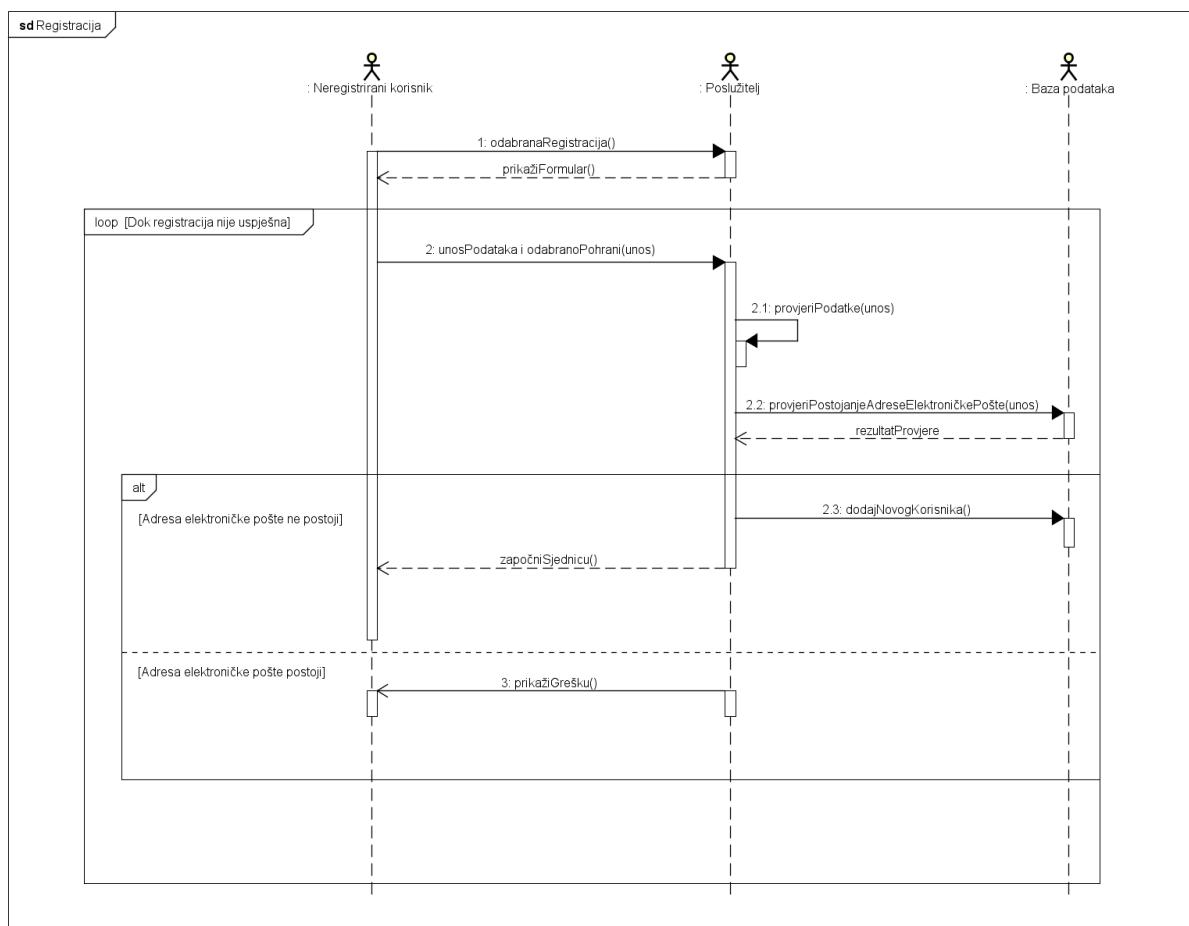


Slika 4.3 - Obrasci vezani za rezervacije

## 4.4 Sekvencijski dijagrami

### 4.4.1 Obrazac uporabe UC1 (Registracija)

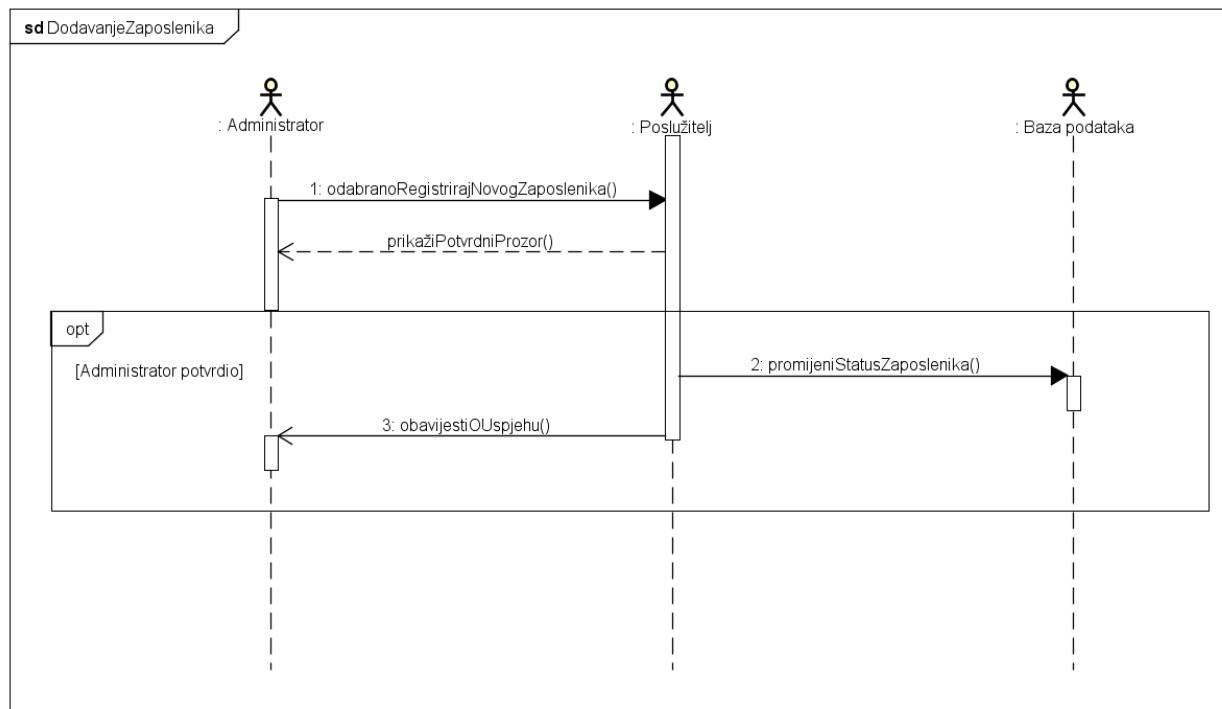
Neregistrirani korisnik odabire opciju registracije te mu poslužitelj prikazuje formular za registraciju. Sljedeći postupak se ponavlja dok neregistrirani korisnik uspješno ne dovrši registraciju. Neregistrirani korisnik unosi tražene podatke i odabire pohranu svojih podataka, nakon čega poslužitelj provjerava ispravnost podataka i provjerava postoji li klijent s danom adresom električne pošte u bazi podataka. U slučaju da su podaci ispravno uneseni te ne postoji korisnik u bazi podataka, on će biti uspješno registriran i započeti će svoju sjedincu. U suprotnom, neregistriranom će korisniku biti prikazana poruka o grešci i omogućen ponovni unos podataka.



Slika 4.4 - Sekvencijski dijagram za UC1 (Registracija)

#### 4.4.2 Obrazac uporabe UC2 (DodavanjeZaposlenika)

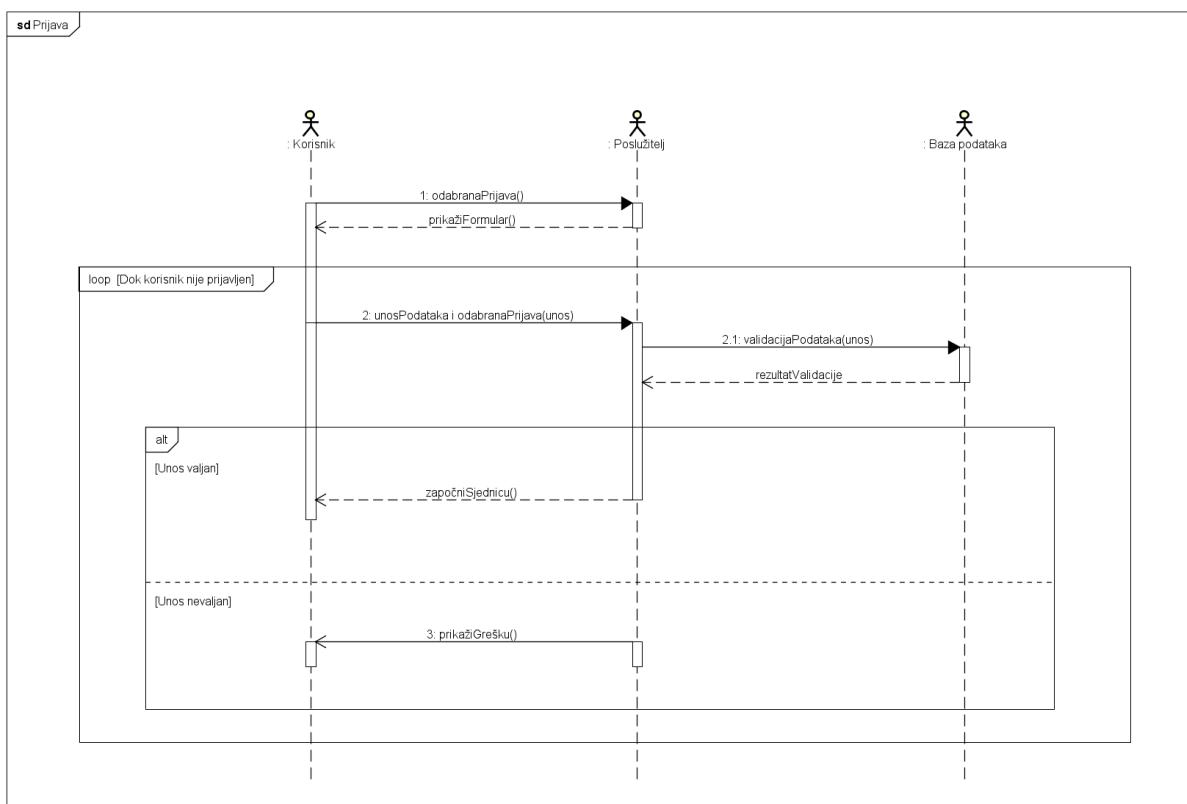
Administrator se nalazi na pregledu podataka korisnika. Ovdje može odabrati opciju dodavanja uloge zaposlenika korisniku, nakon čega mu poslužitelj prikazuje potvrđni prozor. Potvrdi li administrator navedenu opciju, korisniku se na ulogu klijenta dodaje i uloga zaposlenika.



Slika 4.5 - Sekvenčni dijagram za UC2 (DodavanjeZaposlenika)

#### 4.4.3 Obrazac uporabe UC3 (Prijava)

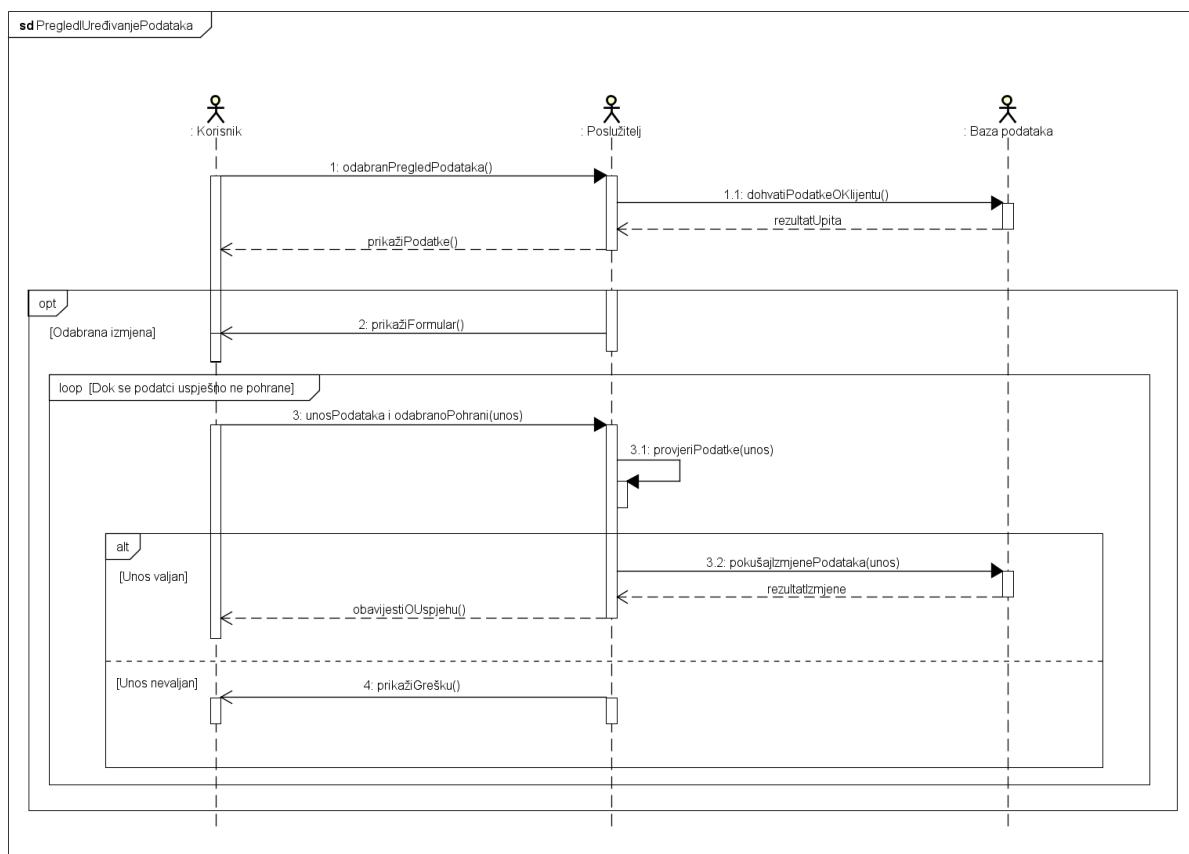
Korisnik odabire opciju prijave u sustav te mu poslužitelj prikazuje formular za prijavu. Sljedeći se postupak ponavlja dok korisnik uspješno ne dovrši prijavu. Korisnik unosi tražene podatke i odabire prijavu, nakon čega poslužitelj proslijeđuje podatke bazi podataka koja validira unos. Rezultat validacije vraća se poslužitelju koji u slučaju ispravnog unosa, započinje sjednicu korisnika. U suprotnome će korisniku biti prikazana poruka o grešci i omogućen ponovni upis podataka.



Slika 4.6 - Sekvencijski dijagram za UC3 (Prijava)

#### 4.4.4 Obrazac uporabe UC4 (PregledIUređivanjePodataka)

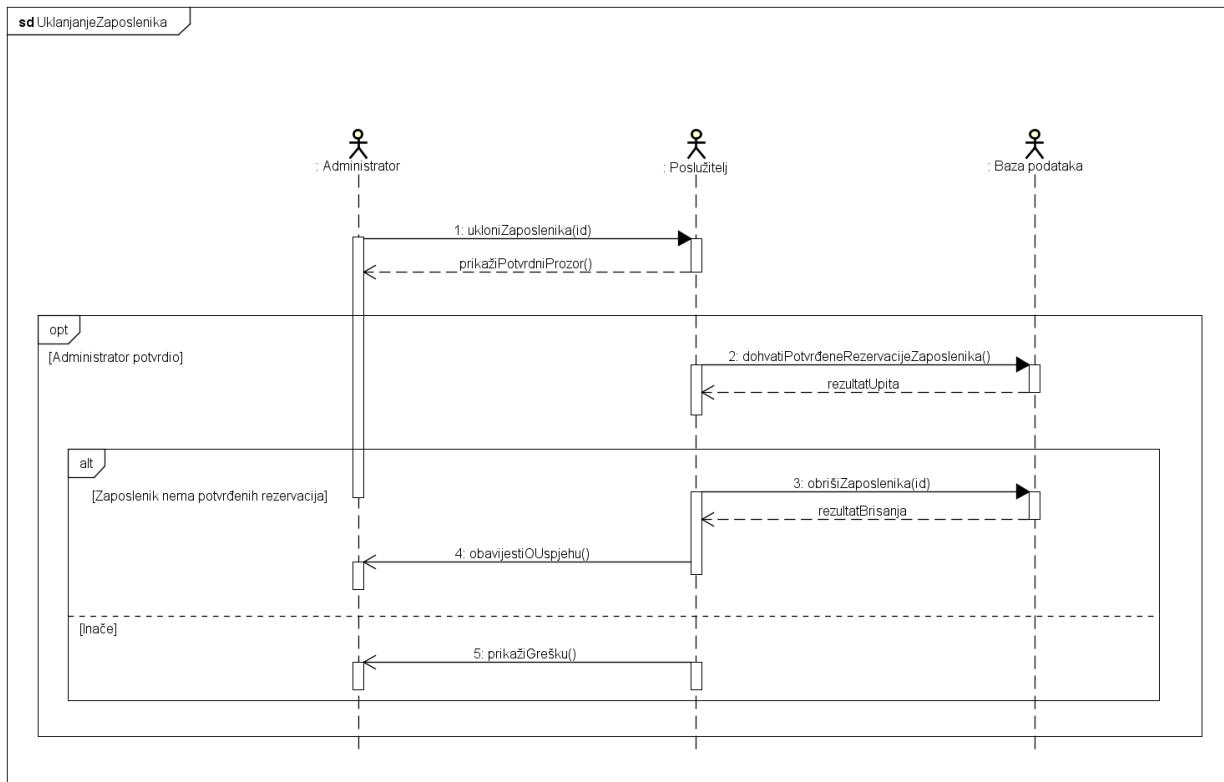
Korisnik odabire opciju pregleda svojih podataka nakon čega poslužitelj dohvaća relevantne podatke o korisniku iz baze podataka. Ako korisnik odabire uređivanje podataka prikazuje mu se formular. Sljedeći se postupak ponavlja dok se željene promjene uspješno ne pohrane. Korisnik potom unosi podatke i odabire pohranu, nakon čega poslužitelj provjerava ispravnost unesenih podataka. U slučaju da su podaci valjni, poslužitelj pokušava izmijeniti podatke spremljene u bazi podataka. Baza podataka vraća rezultat pokušaja izmjene poslužitelju, koji ga prikazuje korisniku. U slučaju da je korisnikov unos bio nevaljan, poslužitelj korisniku prikazuje poruku o grešci i omogućuje ponovni upis podataka.



Slika 4.7 - Sekvencijski dijagram za UC4 (PregledIUređivanjePodataka)

#### 4.4.5 Obrazac uporabe UC5 (UklanjanjeZaposlenika)

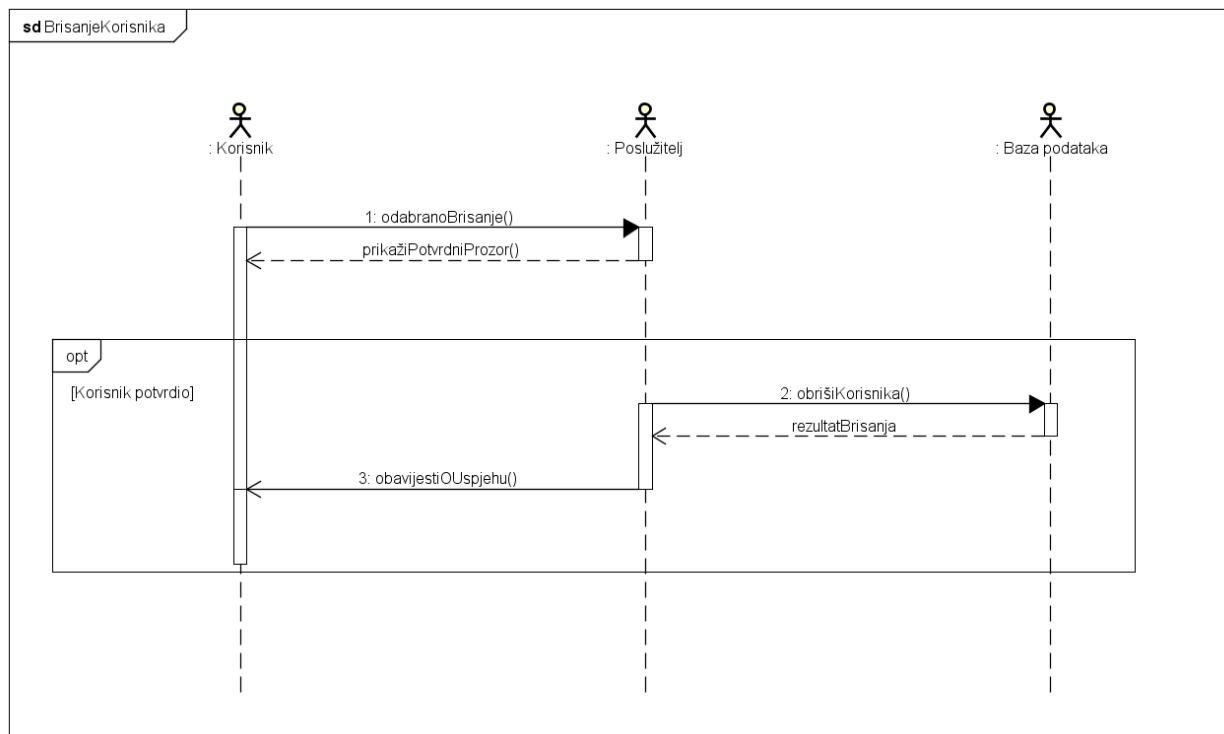
Administrator se nalazi na pregledu podataka zaposlenika i tamo odabire opciju uklanjanja zaposlenika. Poslužitelj mu nakon toga prikazuje potvrđni prozor. U slučaju da je administrator potvrdio i dotičnom zaposleniku trenutno nije dodijeljena ni jedna potvrđena rezervacija, miče mu se uloga zaposlenika. U suprotnom, sustav javlja grešku.



Slika 4.8 - Sekvencijski dijagram za UC5 (UklanjanjeZaposlenika)

#### 4.4.6 Obrazac uporabe UC6 (BrisanjeKorisnika)

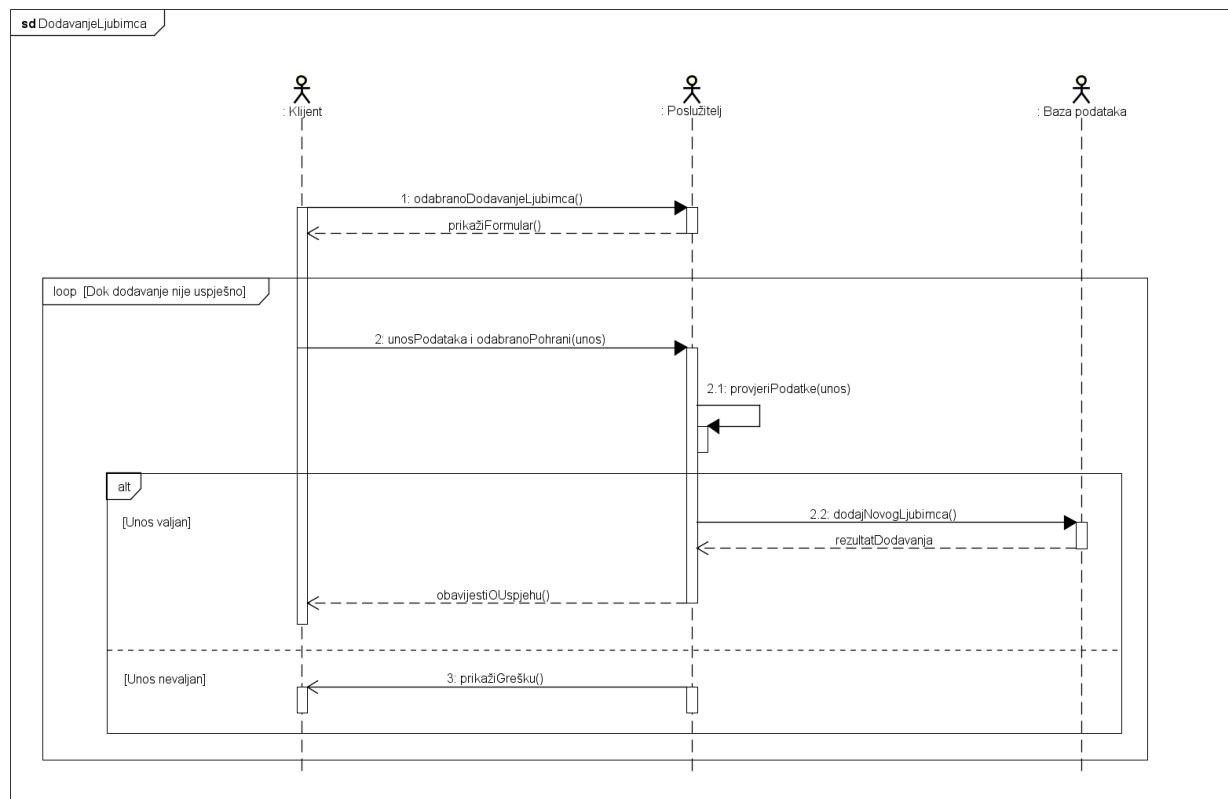
Korisnik odabire opciju brisanja svog korisničkog računa, nakon čega poslužitelj prikazuje potvrdni prozor. U slučaju da korisnik potvrdi brisanje, poslužitelj briše korisnika iz baze podataka i obavještava korisnika o uspješnosti brisanja iz sustava.



Slika 4.9 - Sekvencijski dijagram za UC6 (BrisanjeKorisnika)

#### 4.4.7 Obrazac uporabe UC7 (DodavanjeLjubimca)

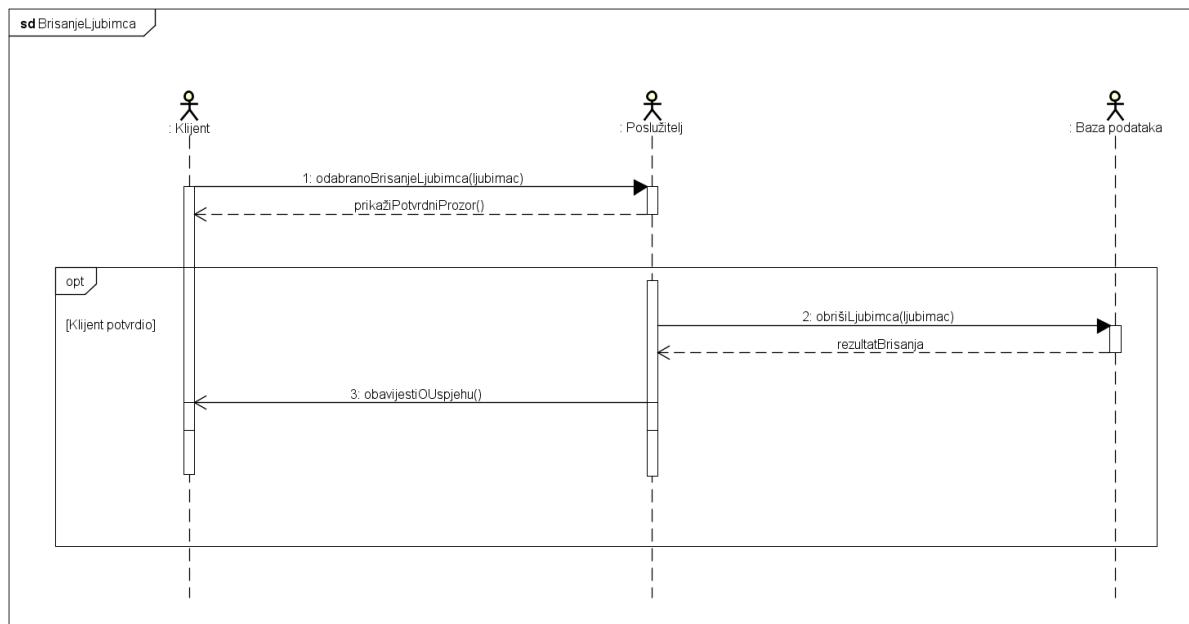
Klijent odabire opciju dodavanja novog ljubimca te mu poslužitelj prikazuje formular za dodavanje ljubimca. Sljedeći postupak se ponavlja dok dodavanje ljubimca nije uspješno. Klijent unosi podatke i odabire pohranu, nakon čega poslužitelj provjerava ispravnost podataka. U slučaju da su podaci ispravni, u bazu podataka se dodaje novi ljubimac i poslužitelj obaveštava klijenta o uspjehu. U suprotnom, klijentu će biti prikazana poruka o grešci i omogućen ponovni upis podataka.



Slika 4.10 - Sekvencijski dijagram za UC7 (DodavanjeLjubimca)

#### 4.4.8 Obrazac uporabe UC8 (BrisanjeLjubimca)

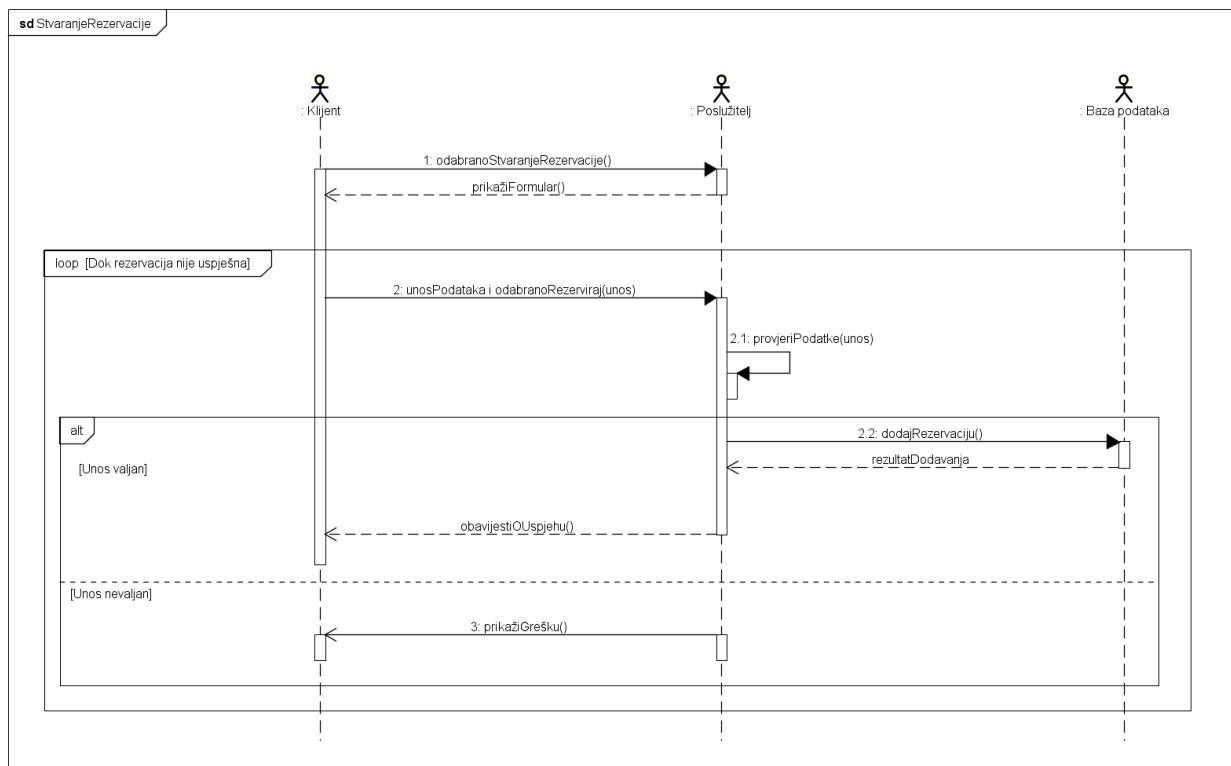
Klijent odabire opciju brisanja ljubimca nakon čega poslužitelj prikazuje potvrđni prozor. U slučaju da klijent potvrdi brisanje ljubimca, poslužitelj briše ljubimca iz baze podataka te prikazuje rezultat brisanja klijentu.



Slika 4.11 - Sekvencijski dijagram za UC8 (BrisanjeLjubimca)

#### 4.4.9 Obrazac uporabe UC9 (StvaranjeRezervacije)

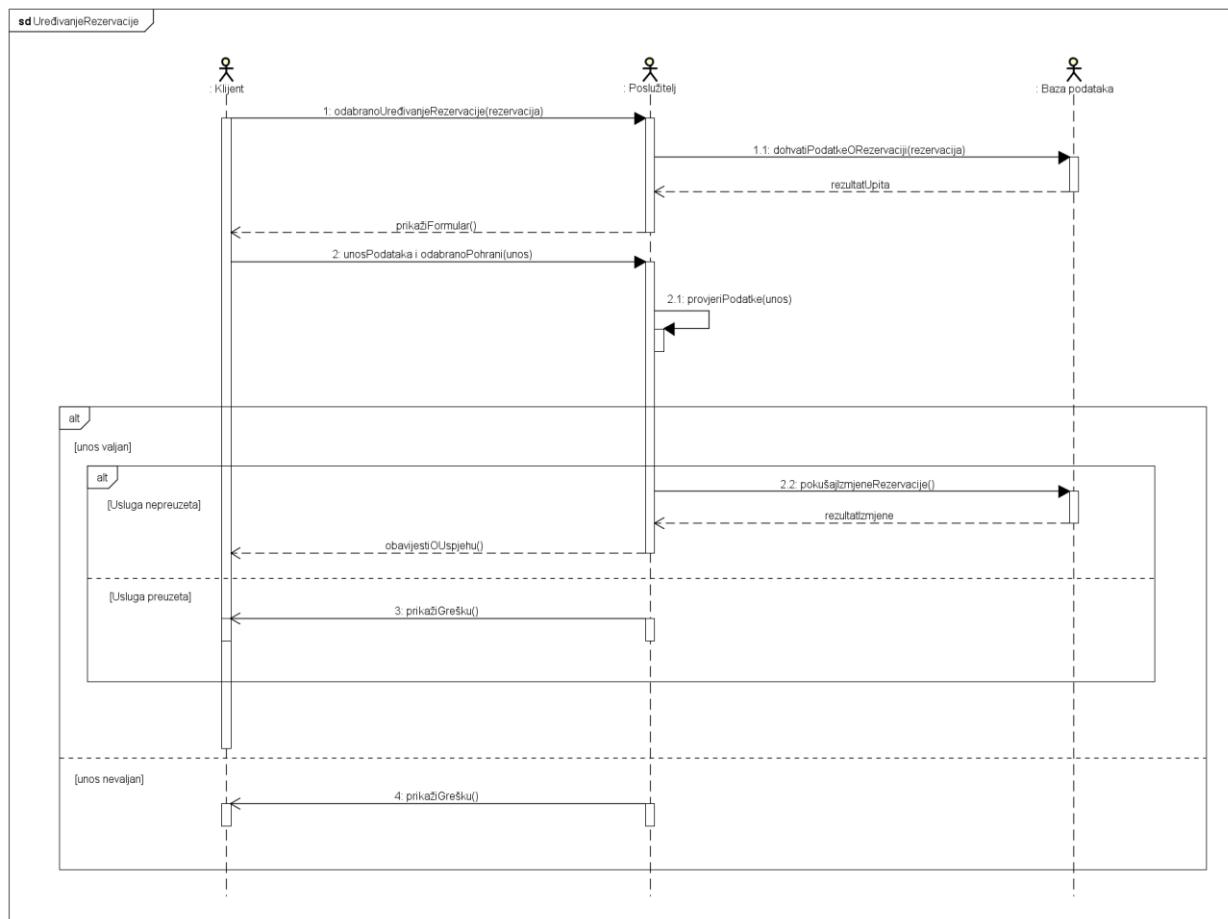
Klijent odabire opiju rezervacije nove usluge te mu poslužitelj prikazuje formular za rezervaciju usluge. Sljedeći postupak se odvija dok klijent ne rezervira novu uslugu. Klijent unosi tražene podatke i odabire pohranu podataka, nakon čega poslužitelj provjerava ispravnost podataka. U slučaju da je unos valjan, rezervacija se dodaje u bazu podataka i poslužitelj obavještava klijenta o uspješnosti rezervacije. U suprotnome, klijentu će biti prikazana poruka o grešci i omogućen ponovni upis podataka.



Slika 4.12 - Sekvencijski dijagram za UC9 (StvaranjeRezervacije)

#### 4.4.10 Obrazac uporabe UC10 (UređivanjeRezervacije)

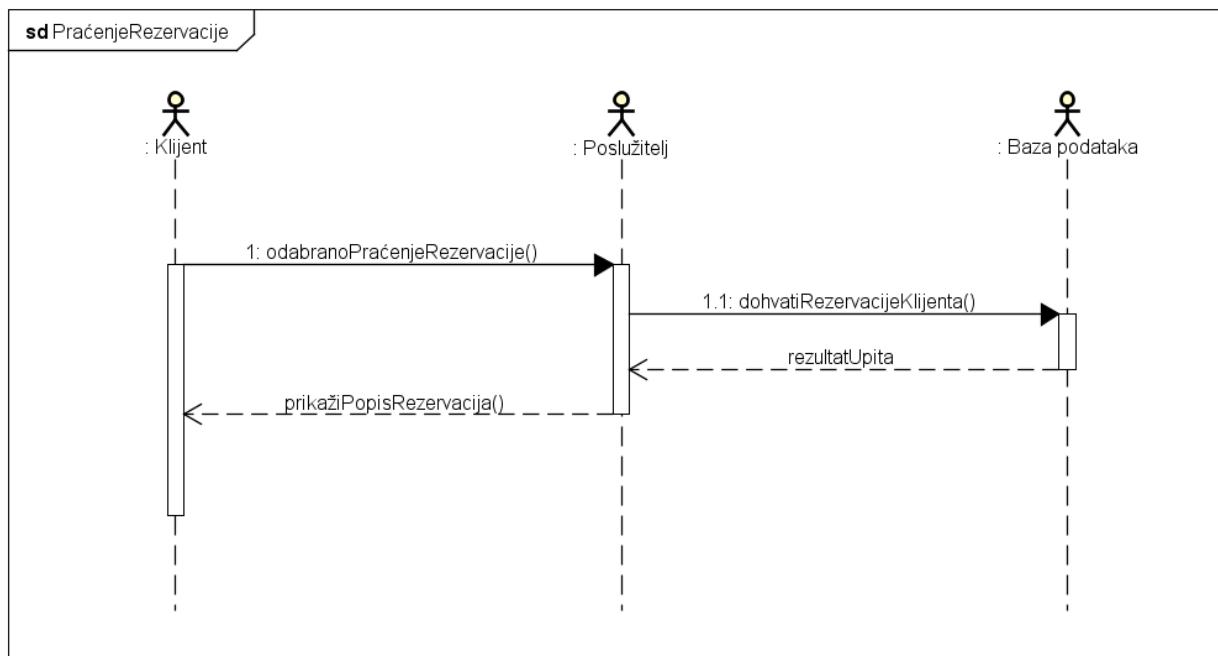
U pregledu rezervacija, korisniku se kraj rezervacija koje se još mogu uređivati nudi opcija "Uredi". Nakon odabira opcije za uređivanje, korisniku se kroz obrazac za uređivanje prikazuju detalji rezervacije koje može mijenjati. Korisnik vrši željene promjene te podnosi zahtjev za izmjenom. Sve dok podaci koje je korisnik unio nisu valjani, prikazuje se obavijest o grešci i traže se ponovni unos i podnošenje zahtjeva. Tek kada korisnik unese valjane podatke, još jednom se provjerava stanje rezervacije. Ako je neki zaposlenik u međuvremenu rezervaciju preuzeo, korisnik dobiva obavijest o nemogućnosti izmjene rezervacije.



Slika 4.13 - Sekvenčni dijagram za UC10 (UređivanjeRezervacije)

#### 4.4.11 Obrazac uporabe UC11 (PraćenjeRezervacija)

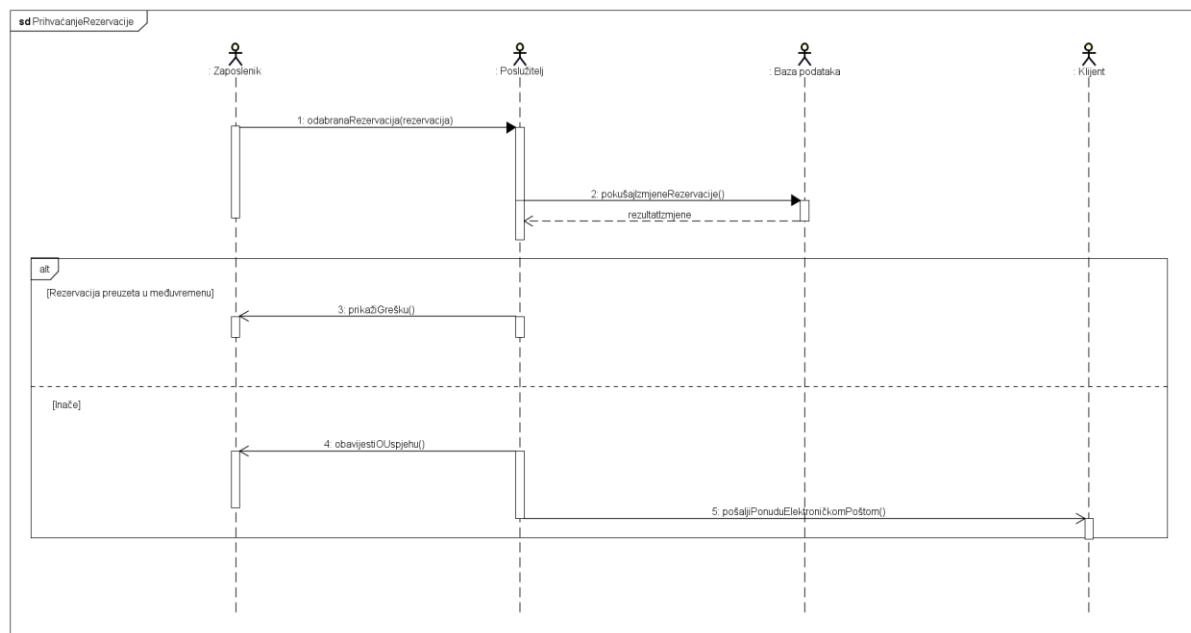
Klijent otvara pregled svih svojih rezervacija. Poslužitelj dohvaća usluge koje je rezervirao dotični klijent te prikazuje njihov popis kroz sučelje web aplikacije. Kraj svake su rezervacije vidljivi i njeni detalji, opcija otkazivanja rezervacije te (u nekim slučajevima) opcija za uređivanje rezervacija.



Slika 4.14 - Sekvencijski dijagram za UC11 (PraćenjeRezervacija)

#### 4.4.12 Obrazac uporabe UC12 (PrihvaćanjeRezervacije)

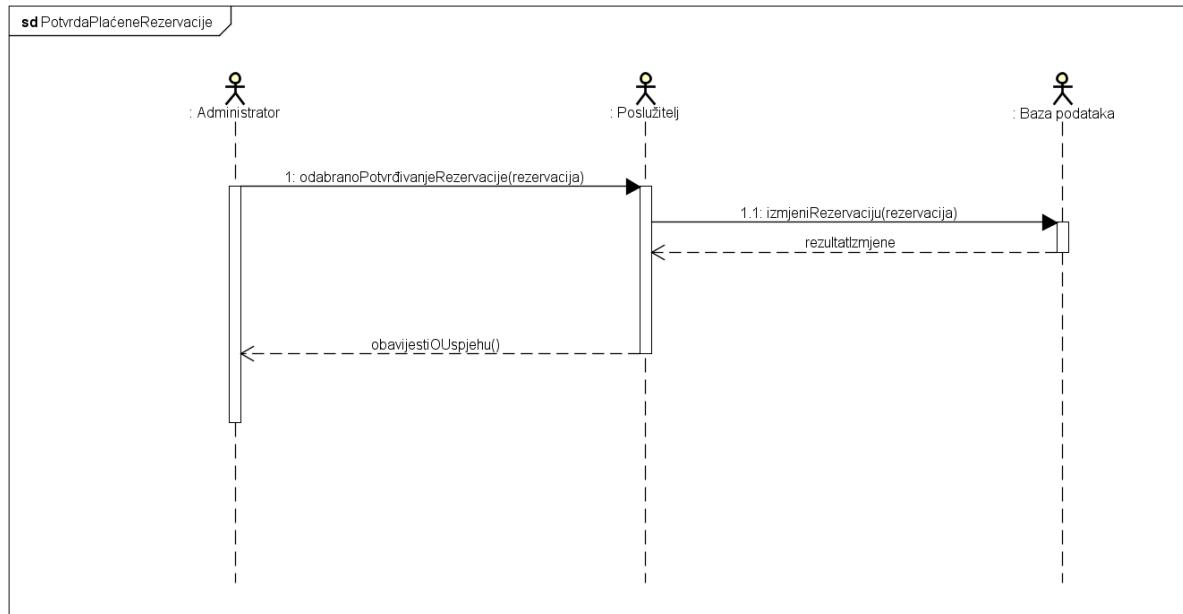
Zaposlenik se nalazi u pregledu svih rezervacija. Uz svaku se rezervaciju razine jedan (nepreuzetu rezervaciju) nudi opcija preuzimanja. Zaposlenik odabire preuzimanje neke od ponuđenih rezervacija. Ako u međuvremenu spomenutu uslugu nije preuzeo nitko drugi, ona se označava kao preuzeta. Poslužitelj joj ažurira stanje u bazi podataka te se ona time prebacuje iz liste nepreuzetih rezervacija u listu preuzetih rezervacija dotičnog zaposlenika te klijent dobiva poruku s ponudom.



Slika 4.15 - Sekvencijski dijagram za UC12 (PrihvaćanjeRezervacije)

#### 4.4.13 Obrazac uporabe UC13 (PotvrdaPlaćeneRezervacije)

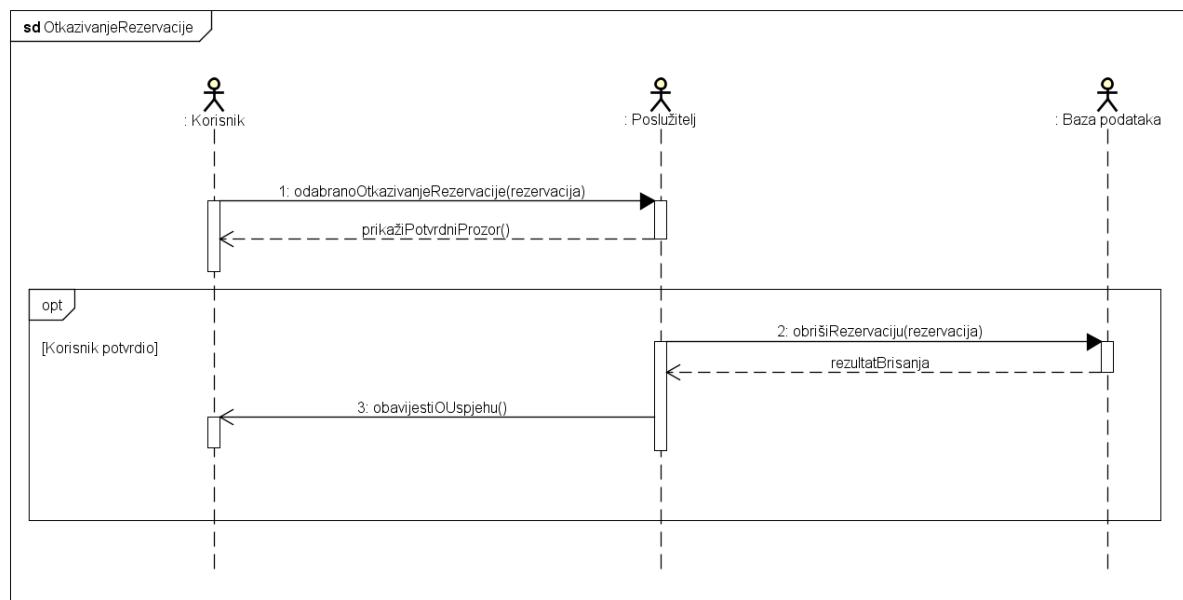
Administrator se nalazi u pregledu svih rezervacija. Svaka preuzeta usluga nudi opciju potvrde. Administrator može potvrditi bilo koju rezervaciju, tipično to radi nakon što je ona plaćena, ali sustav plaćanje ne regulira ni na koji način. Rezervacija se nakon potvrđivanja prebacuje u listu potvrđenih usluga te administrator dobiva obavijest o uspjehu akcije.



Slika 4.16 - Sekvencijski dijagram za UC13 (PotvrdaPlaćeneRegistracije)

#### 4.4.14 Obrazac uporabe UC14 (OtkazivanjeRezervacije)

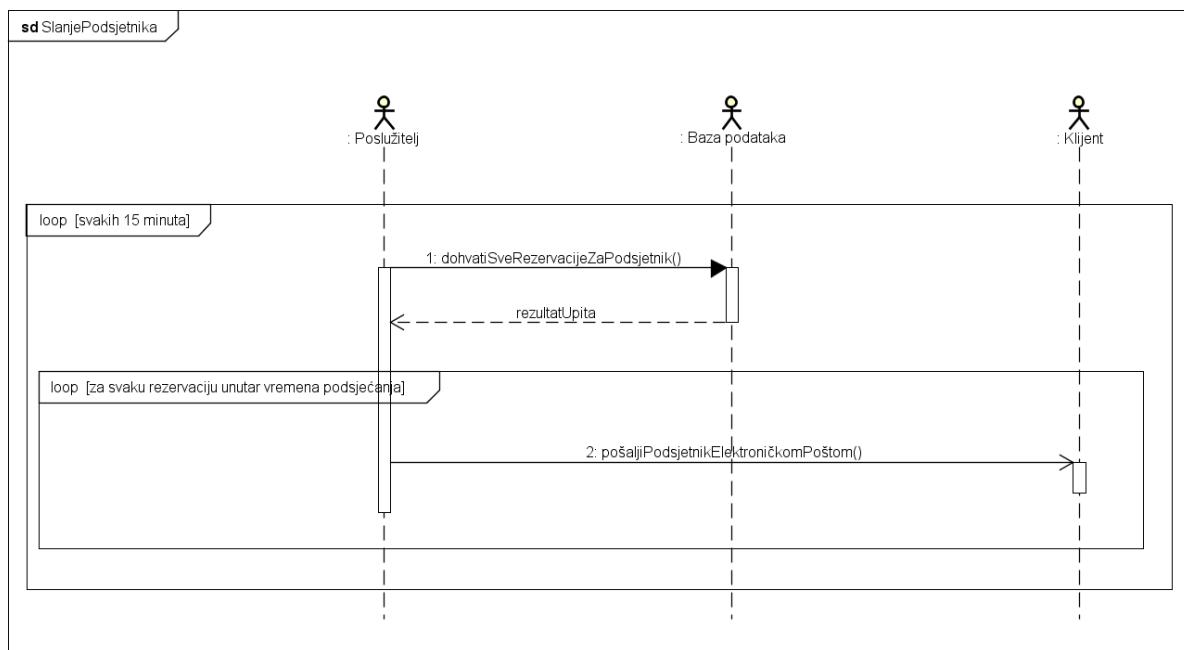
U pregledu rezervacija, korisniku se kraj svih rezervacija nudi opcija "Otkazi". Ova je opcija dostupna uvijek, neovisno o razini rezervacije. Nakon što korisnik odabere akciju otkazivanja, poslužitelj mu prikazuje potvrđni prozor. Potvrdi li korisnik otkazivanje, rezervacija se miče s popisa aktivnih rezervacija.



Slika 4.17 - Sekvencijski dijagram za UC14 (OtkazivanjeRezervacije)

#### 4.4.15 Obrazac uporabe UC15 (SlanjePodsjetnika)

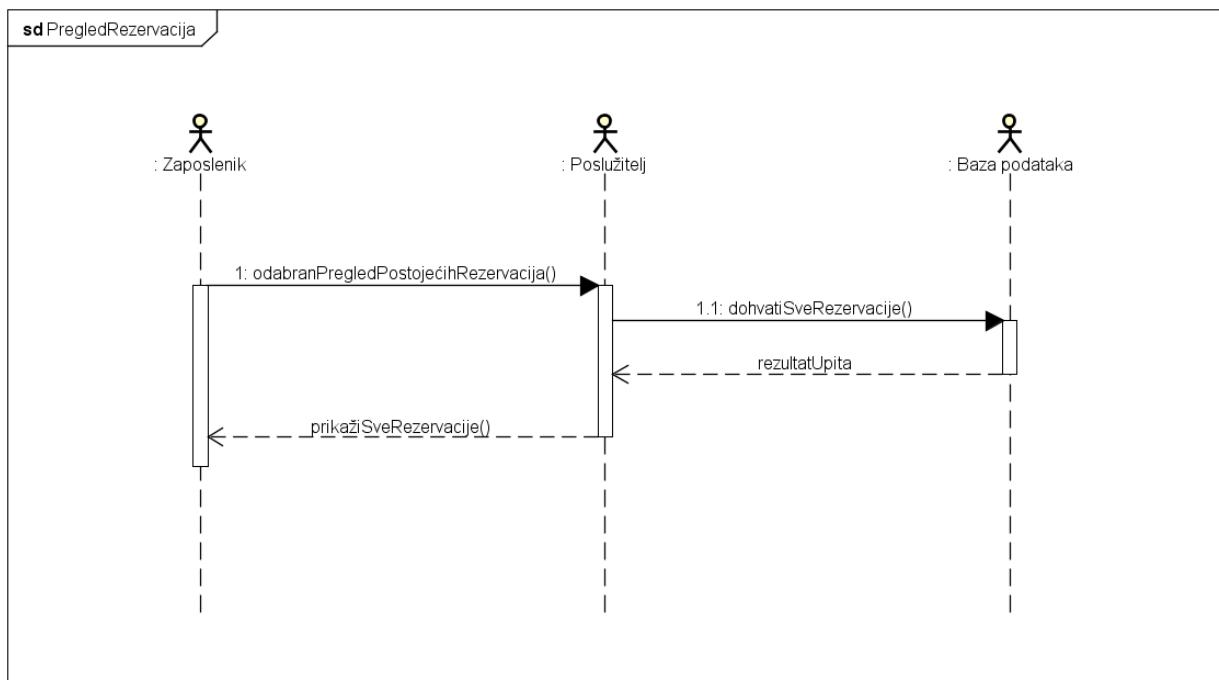
Poslužitelj svakoga dana u isto vrijeme prolazi kroz sve rezervacije u bazi podataka te traži one čiji su termini unutar vremena podsjećanja. Za svaku takvu rezervaciju, u slučaju da je klijent odabrao opciju podsjetnika, poslužitelj mu na adresu elektroničke pošte šalje podsjetnik.



Slika 4.18 - Sekvencijski dijagram za UC15 (SlanjePodsjetnika)

#### 4.4.16 Obrazac uporabe UC16 (PregledRezervacija)

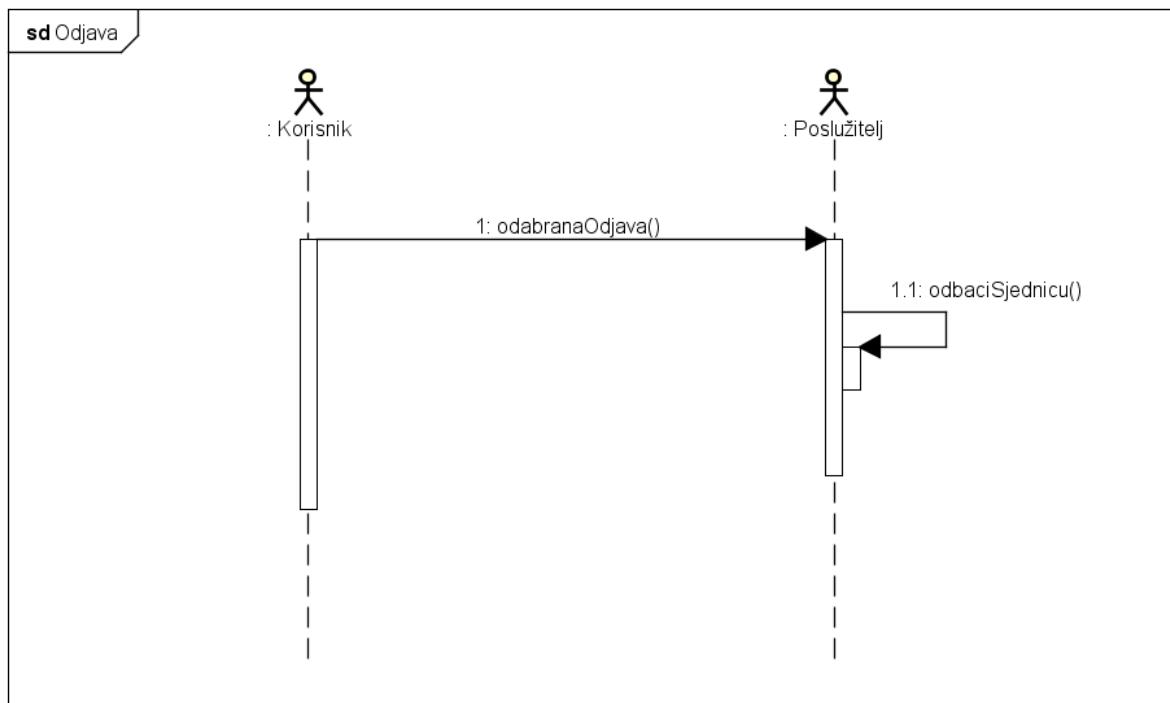
Zaposlenik otvara pregled rezervacija. Poslužitelj dohvaća sve aktivne rezervacije iz baze podataka i prikazuje ih zaposleniku. Rezervacije su podijeljene u tri kategorije (stupca). Prvu kategoriju čine sve rezervacije koje još nitko nije preuzeo. Druga su kategorija usluge koje je dotični zaposlenik preuzeo, ali nisu potvrđene od strane administratora. Treću kategoriju čine usluge koje je dotični zaposlenik preuzeo i potvrđene su od strane administratora.



Slika 4.19 - Sekvencijski dijagram za UC16 (PregledRezervacija)

#### 4.4.17 Obrazac uporabe UC17 (Odjava)

Korisnik odabire odjavu nakon čega poslužitelj briše trenutnu sjednicu i odjavljuje korisnika te ga obavještava o tome.



Slika 4.20 - Sekvencijski dijagram za UC17 (Odjava)

## 5. Nefunkcionalni zahtjevi

- Sustav treba omogućiti rad više korisnika u isto vrijeme
- Sustav treba podržavati znakove hrvatske abecede
- Sustav treba osiguravati vlastiti integritet i integritet baze podataka prilikom neispravnog korištenja

## 6. Arhitektura i dizajn sustava

### 6.1 Svrha, opći prioriteti i skica sustava

Uzveši u obzir način na koji bi poslovni subjekt trebao koristiti sustav za suradnju s krajnjim korisnicima, logičan je odabir arhitekture web aplikacije. Korisniku mora biti ponuđen jednostavan, brz i efikasan način dolaska do željene usluge. Potonji ne smije pred korisnika postaviti komplikiran postupak instalacije, a korištenje mora biti intuitivno. Osim korisničkog iskustva, važno je imati na umu da održavanje sustava mora biti jeftino, a nadogradnja jednostavna.

Cijeli sustav radi po principu arhitekture klijent-poslužitelj s većim naglaskom na poslu koji obavlja poslužitelj. Glavne podsustave arhitekture čine:

- **web aplikacija** - zasluzna za korisničko sučelje, logiku i pristup bazi podataka
- **baza podataka** - zadužena za pohranu informacija nužnih za ispravan rad sustava
- **web poslužitelj** - koristeći web aplikaciju obrađuje korisničke zahtjeve

#### 6.1.1 Web poslužitelj

Web poslužitelj naziv je za računalo na kojem se sustav nalazi. Na njemu se nalazi baza podataka te je na njemu sama web aplikacija pokrenuta. Poslužitelj s klijentom komunicira protokolom *HTTP*. Krajnji korisnik s klijentskog računala šalje zahtjeve, a poslužitelj ih obrađuje i šalje odgovore.

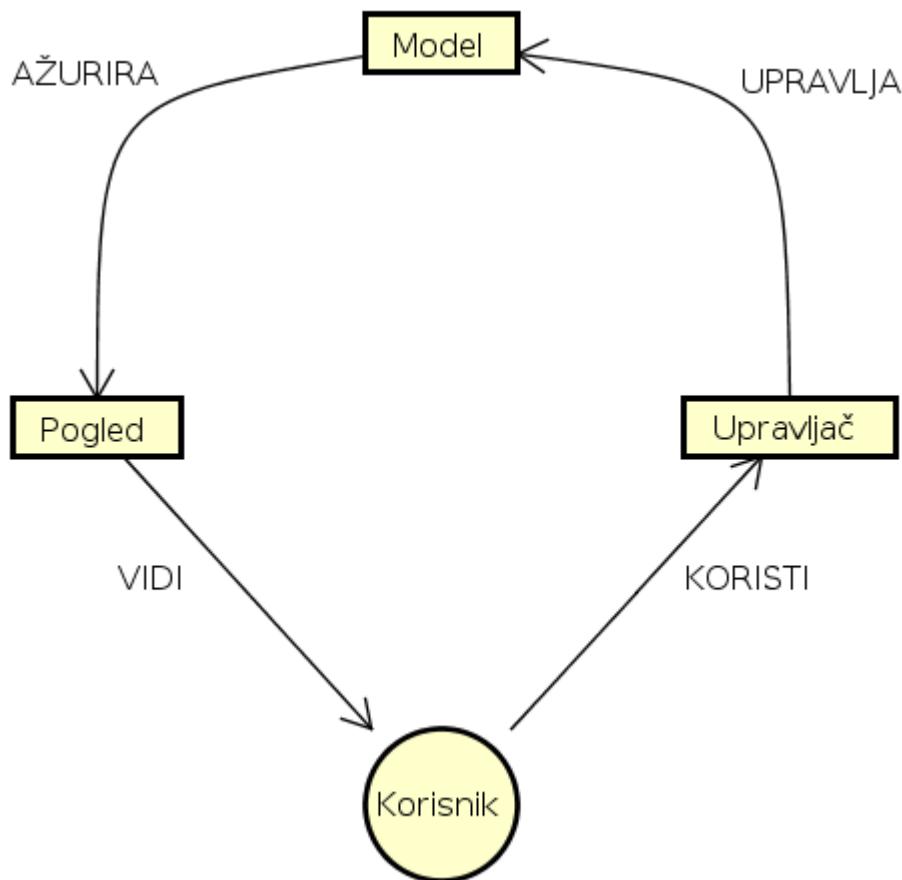
#### 6.1.2 Web aplikacija

Web aplikaciju čini program koji se vrti na poslužitelju, a zadužen je za obradu zahtjeva korisnika. Osim s korisnikom, web aplikacija komunicira i s bazom podataka. S obzirom na važnost kvalitete sloja aplikacije prezentiranog krajnjem korisniku aplikacije za arhitekturu aplikacije odabran je **obrazac MVC** (engl. *Model-View-Controller*) zahvaljujući kojem je izrada korisničkog sučelja jednostavna, testiranje i nadogradnja olakšane, a komunikacija između komponenata aplikacije brza. Obrazac MVC razdvaja dijelove aplikacije po dužnostima u tri kategorije:

- **Model** - dio aplikacije koji predstavlja podatkovne cjeline. Obuhvaća strukture koje razmjenjuju *upravitelj* i baza podataka, ali i strukture koje *upravitelj* prikazuje kroz *poglede*. Ova su dva modela razdvojena kako se *pogledima* ne bi slale redundantne informacije. Važno je napomenuti da *model* ni na koji način ne smije ovisiti o *pogledu* ili *upravitelju* već služi isključivo za reprezentaciju podataka.
- **Pogled** - dio aplikacije kojem korisnik neposredno pristupa, služi za prikaz modela pogleda i interakciju s korisnikom. Ima ulogu spojnica između korisnika i aplikacije, dijelovi *pogleda* s ostatkom aplikacije komuniciraju preko *upravitelja*

- **Upraviteљ** - dio aplikacije koji interpretira akcije korisnika, poziva odgovarajuće poglede te im prosljeđuje potrebne podatke koje moraju prikazati.

Glavna prednost MVC uzorka jest razdvajanje podatkovnog (*model*) i prezentacijskog (*pogled*) dijela umetanjem *upravitelja* između njih što povećava mogućnost ponovnog korištenja ranije napisanog koda te vrlo jasno razdvaja funkcionalnosti i dužnosti aplikacije (što je za razvoj u timu izuzetno bitno).



Slika 6.1 - način rada obrasca MVC

### 6.1.3 Baza podataka

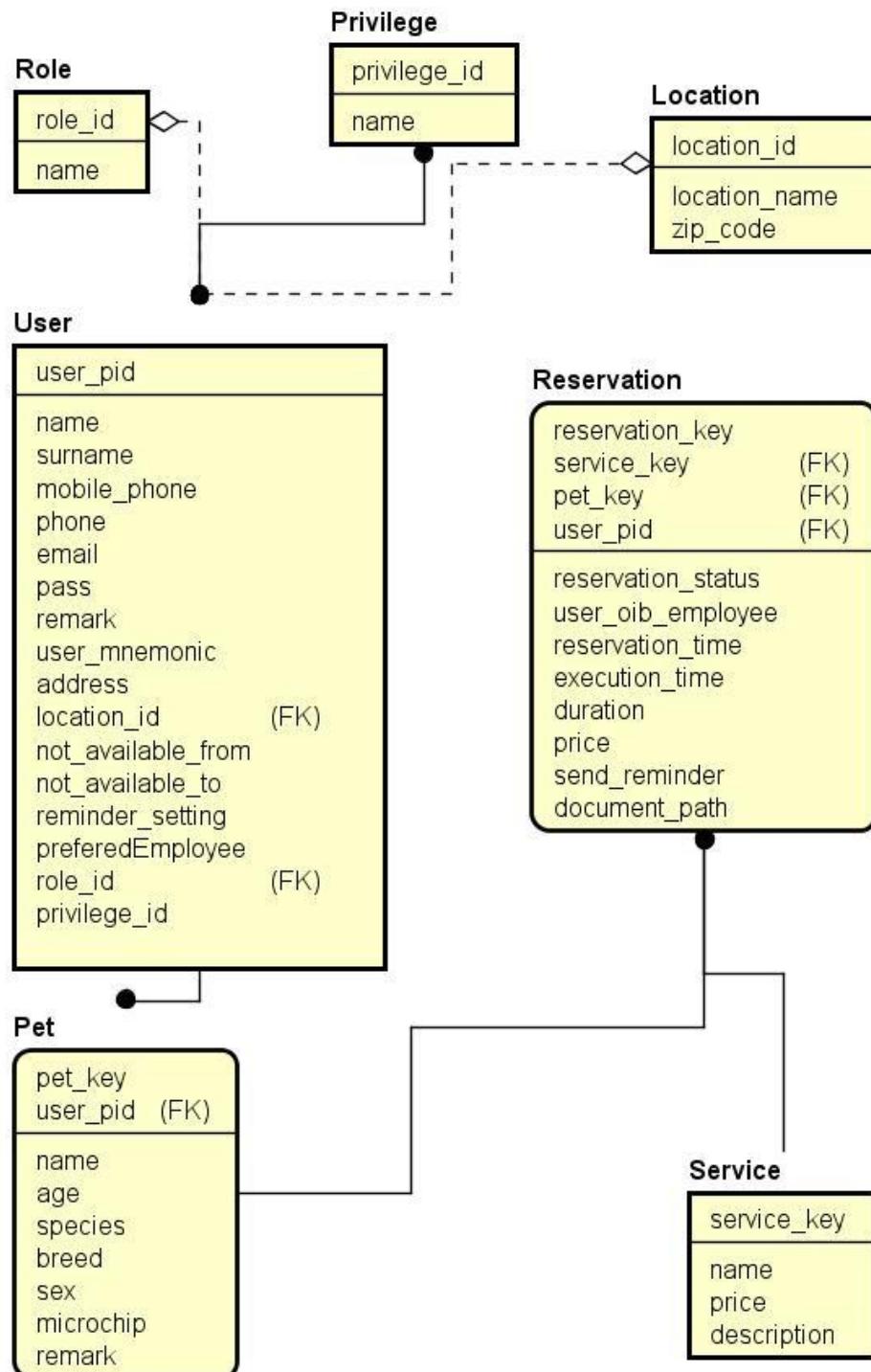
Kako bi sustav uspješno funkcionirao nužna je pohrana svih podataka koje koristi web aplikacija te u tu svrhu koristimo relacijsku bazu podataka. Tablice od kojih se sastoji baza podataka ovog sustava su sljedeće:

- **User**
  - Ovaj entitet sadrži informacije o korisnicima aplikacije. Atributi ovog entiteta možemo promatrati kao tri cjeline. Prva grupa atributa odnosi se na osnovne informacije o fizičkoj osobi nužne za jednoznačnu identifikaciju. Tu spadaju osobni identifikacijski broj osobe, ime, prezime, adresa elektroničke pošte te lozinka, a važan dio ove cjeline

jest i atribut uloga koji služi za upravljanje ulogama i dozvolama unutar sustava te određuje radi li se o klijentu, zaposleniku ili administratoru. Druga cjelina sastoji se od atributa nužnih za komunikaciju i vremensko-prostornu organizaciju podjele posla kao što su adresa stanovanja, poštanski broj, broj telefona, itd. U treću cjelinu spadaju atributi koji govore o preferencijama korisnika, to jest atributi napomena i postavke elektroničke pošte, atributi kojima su određeni početak i kraj radnog vremena zaposlenika te atribut kojim zaposlenik odabire hoće li elektroničkom poštrom primati obavijesti za sve poslove, samo za poslove u kojima je naveden kao preferirani odabir ili nikada.

- **Pet**
  - Ovaj entitet sadrži informacije o kućnim ljubimcima te je usko vezan uz entitet User preko atributa osobni identifikacijski broj vlasnika. Riječ je o *jedan-na-mnogo* vezi, odnosno korisnik može imati više ljubimaca, a ljubimac nužno ima jednog vlasnika. Osim toga ovaj entitet sadrži i identifikacijske attribute kao što su ime, vrsta, pasmina, dob i spol ljubimca te attribute mikročip i napomena koji daju dodatne informacije o ljubimcu.
- **Service**
  - Ovaj entitet predstavlja usluge koje poslovni subjekt nudi svojim klijentima. Osim atributa identifikacijski broj usluge sastoji se i od atributa tip usluge, cijena usluge i opis usluge koji daju sve informacije o ponuđenim uslugama. Ovaj entitet vezan je uz entitet *Reservation* *jedan-na-mnogo* vezom jer jedna rezervacija nužno ima jednu uslugu, a jedna usluga može biti vezana za više rezervacija.
- **Reservation**
  - Ovaj entitet predstavlja konkretnu uslugu za nekog klijenta, to jest njegovog ljubimca, a sastoji se od atributa poput statusa rezervacije, identifikacijskog broja ljubimca, osobnog identifikacijskog broja zaposlenika zaduženog za izvršenje ove usluge, vremena kada je rezervacija stvorena, vremena kada bi usluga trebala biti izvršena, vremena trajanja usluge te cijene. Iako je cijena usluge već određena u entitetu Service, nužno je postojanje ovog atributa kako bi bili pokriveni slučajevi individualnog popusta i sličnog.
- **Location**
  - Ovaj entitet sadrži attribute poštanski broj i naziv mjesta.
- **Role**
  - Ovaj entitet predstavlja ulogu dodijeljenu korisniku, odnosno predstavlja informaciju radi li se klijentu, zaposleniku ili administratoru. S entitetom User čini *jedan-na-mnogo* vezu, to jest jedna uloga može biti dodijeljena više korisnika, a jedan korisnik može imati samo jednu ulogu. Atributi ovog entiteta su identifikacijski broj te ime uloge.
- **Privilege**

- Ovaj je entitet u mnogo-na-mnogo vezi s entitetom User te daje informacije o privilegijama korisnika. Ključ ovog entiteta jest identifikacijski broj te ima jedan atribut koji predstavlja ime privilegije.



Slika 6.2 - ER model baze podataka

## 6.2 Dijagrami razreda

Podjela razreda u pakete napravljena je u skladu s MVC obrascem:

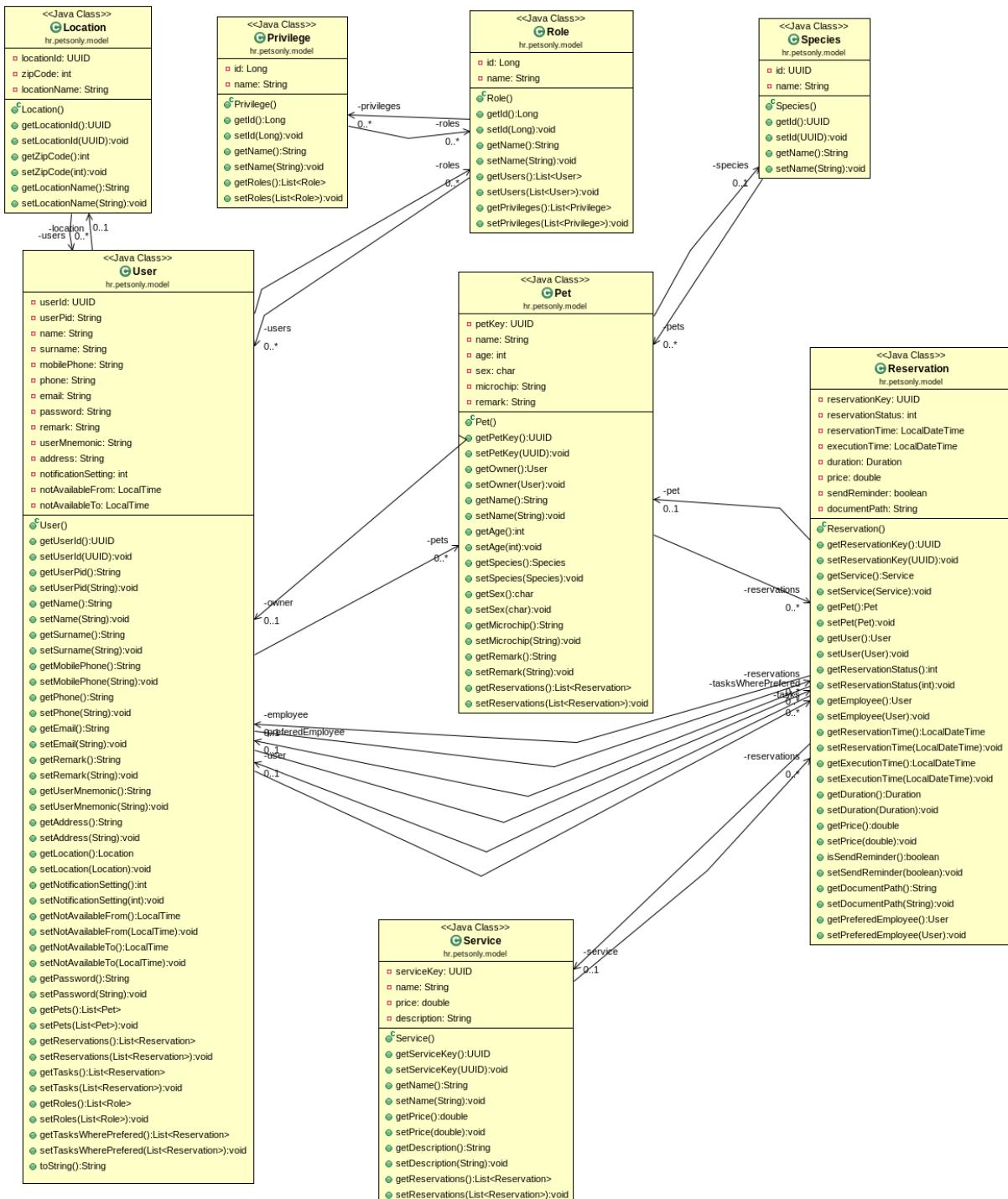
- **Razredi modela**
- **Razredi upravitelja**
- **Repozitoriji**
- **Razredi pretvarači**
- **Razredi modela prezentacijskog sloja**
- **Razredi modela formulara**
- **Razredi za validaciju**
- **Konfiguracijski razredi**

### 6.2.1 Razredi modela

Dužnost razreda modela opisivanje je entiteta iz baze podataka (engl. *OR mapping*). U ovaj paket spadaju sljedeći razredi:

- User - predstavlja entitet *User*
- Location - predstavlja entitet *Location*
- Pet - predstavlja entitet *Pet*
- Service - predstavlja entitet *Service*
- Reservation - predstavlja entitet *Reservation*
- Species - predstavlja entitet *Species*
- Role - predstavlja entitet *Role*
- Privilege - predstavlja entitet *Privilege*

Detaljan opis entiteta nalazi se u prethodnom poglavlju

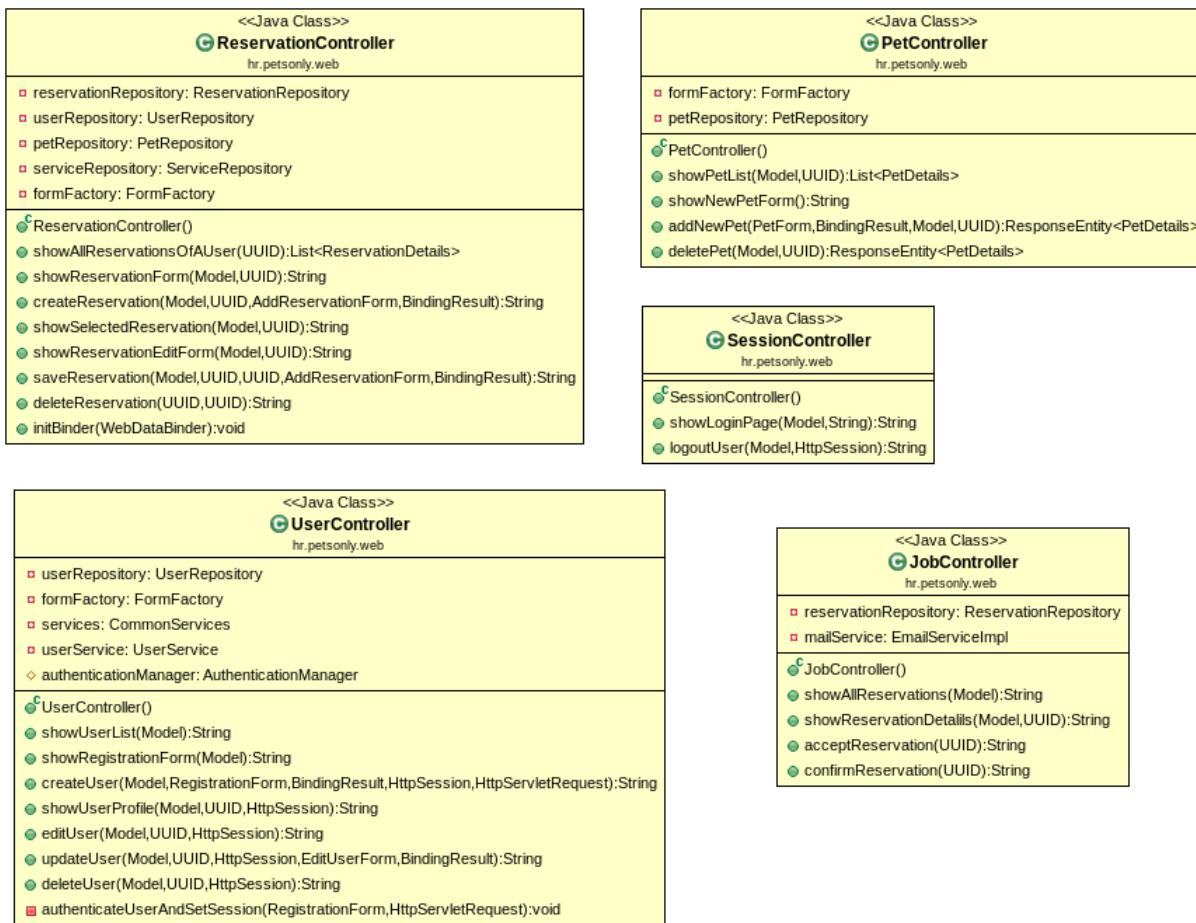


Slika 6.3 - Detaljan UML dijagram razreda modela

### 6.2.2 Razredi upravitelja

Razredi upravitelja obrađuju HTTP zahtjeve te šalju odgovore/rukaju pogledima. U ovaj paket spadaju sljedeći razredi:

- **UserController**
  - Rukuje svim funkcionalnostima koje su vezane s profilom korisnika: registracija, prijava, brisanje korisnika, zapošljavanje korisnika, uređivanje korisničkih podataka, odjava.
- **PetController**
  - Rukuje svim funkcionalnostima koje su vezane s kućnim ljubimcima: dodavanje ljubimaca, brisanje ljubimaca.
- **ReservationController**
  - Rukuje svim funkcionalnostima koje su vezane sa stvaranjem, uređivanjem i pregledom rezervacija iz perspektive klijenta.
- **JobController**
  - Rukuje svim funkcionalnostima vezanima uz obavljanje poslova. Drugim riječima, pruža pogled na rezervacija i mogućnosti rukavanja iz perspektive zaposlenika
- **SessionController**
  - Rukuje svim funkcionalnostima vezanima uz uspostavu i prekid sjednice korisnika.



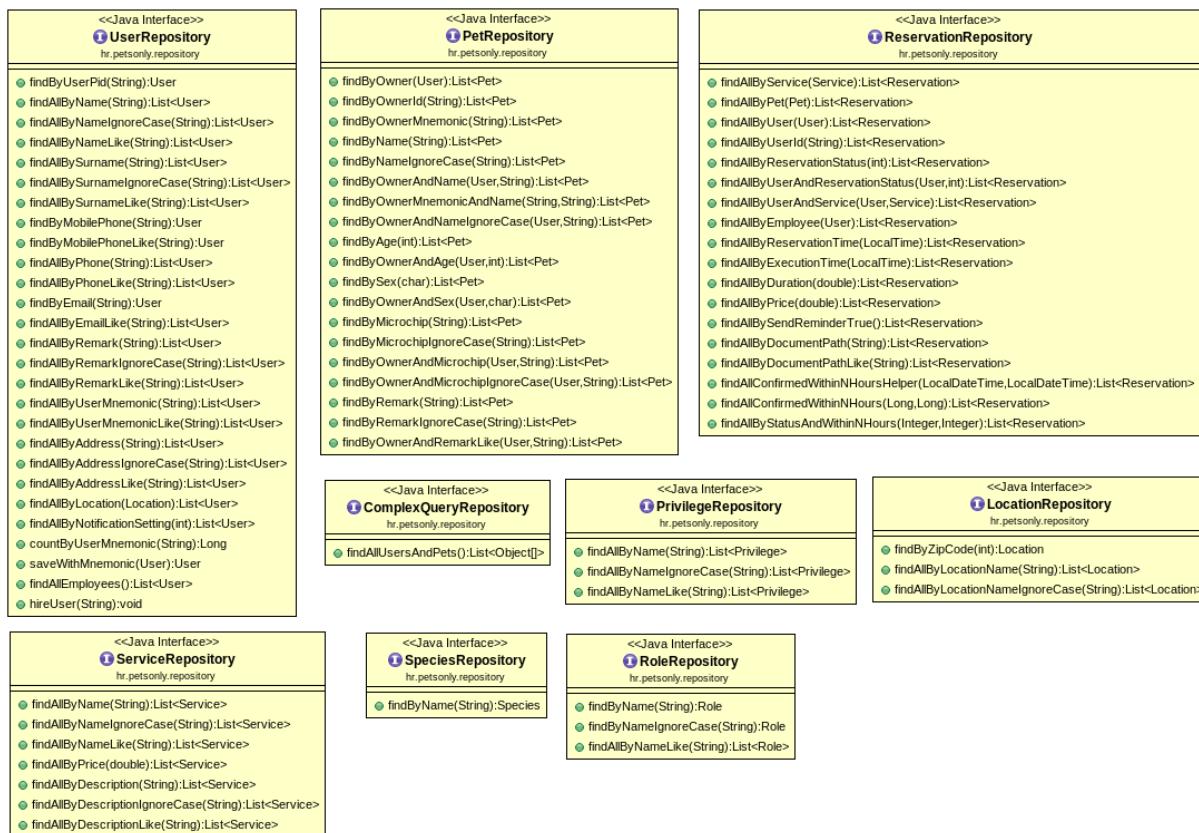
Slika 6.4 - Detaljan UML dijagram razreda upravitelja

### 6.2.3 Repozitoriji

Dužnost ovog paketa komunikacija je s bazom podataka

- **UserRepository** - rukuje entitetom *User*
- **LocationRepository** - rukuje entitetom *Location*
- **PetRepository** - rukuje entitetom *Pet*
- **ReservationRepository** - rukuje entitetom *Reservation*
- **ServiceRepository** - rukuje entitetom *Service*
- **PrivilegeRepository** - rukuje entitetom *Privilege*
- **RoleRepository** - rukuje entitetom *Role*
- **SpeciesRepository** - rukuje entitetom *Species*
- **ComplexQueryRepository** - rukuje objektima koji nisu entiteti u bazi podataka

Repozitoriji razredima upravitelja predstavljaju sučelje za komunikaciju s bazom podataka. Kao strukture podataka u toj komunikaciji koriste se razredi modela.

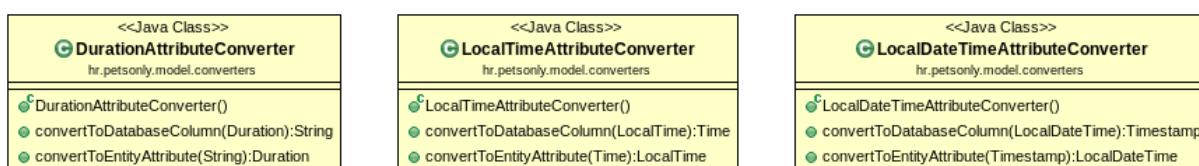


Slika 6.5 - Detaljan UML dijagram repozitorija

#### 6.2.4 Razredi pretvarači

Razredi pretvarači omogućavaju preslikavanje između Java objekata i entiteta za određene klase koje nisu podržane u trenutnim verzijama *OR mappera*.

- **DurationAttributeConverter** - pretvara klasu *Duration* u *String*
- **LocalTimeAttributeConverter** - pretvara klasu *LocalTime* u *Time*
- **LocalDateTimeAttributeConverter** – pretvara klasu *LocalDateTime* u *Timestamp*

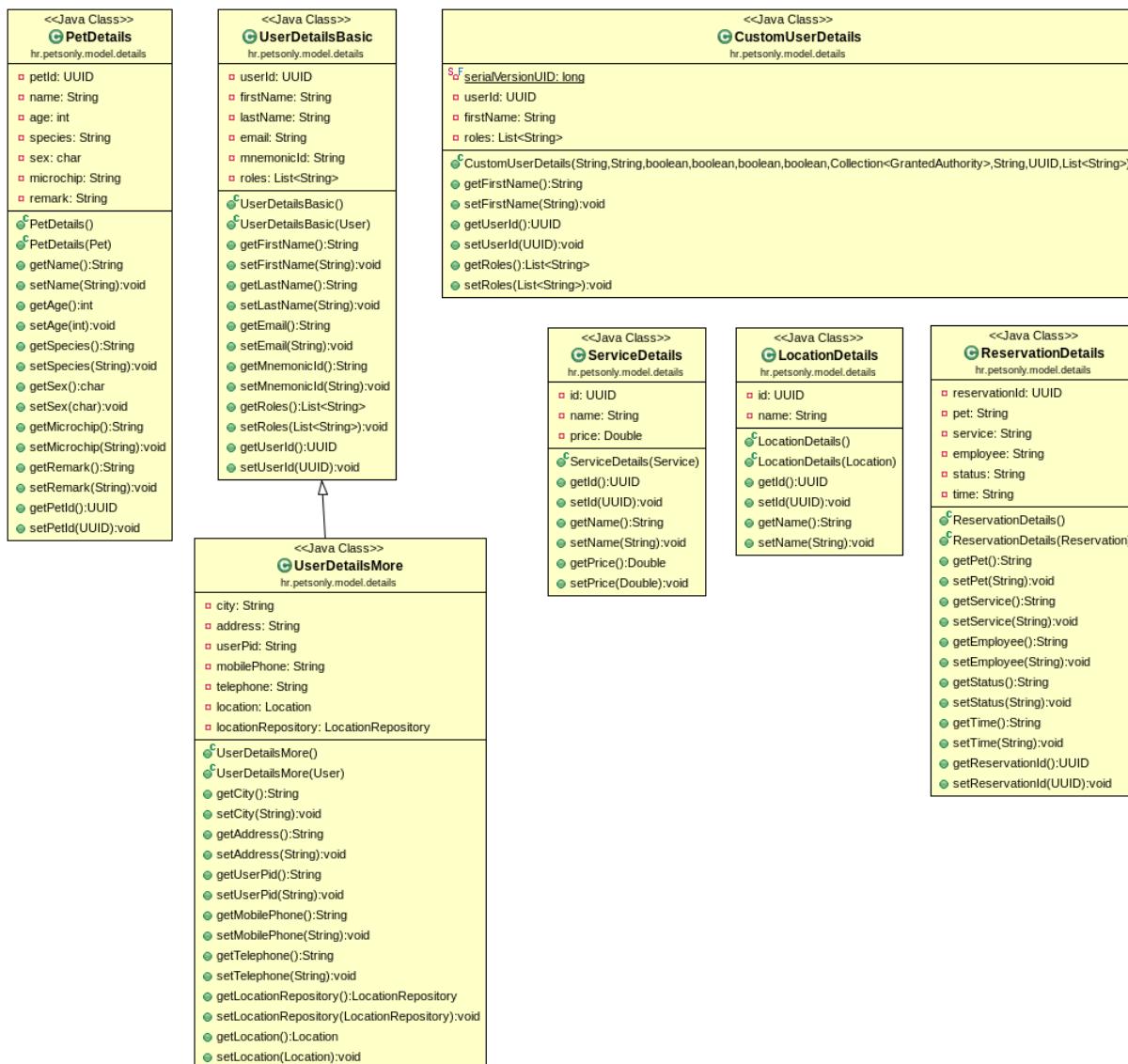


Slika 6.6 – Detaljan UML dijagram razreda pretvarača

## 6.2.5 Razredi modela prezentacijskog sloja

Razredi modela prezentacijskog sloja predstavljaju modele koji su dostupni prezentacijskom sloju unutar MVC obrasca.

- PetDetails - modelira Pet klasu
- UserDetailsBasic – modelira osnovne informacije User klase
- UserDetailsMore – modelira dodatne informacije User klase
- CustomUserDetails - modelira neke informacije User klase
- ServiceDetails – modelira Service klasu
- LocationDetails – modelira Location klasu
- ReservationDetails - modelira Reservation klasu

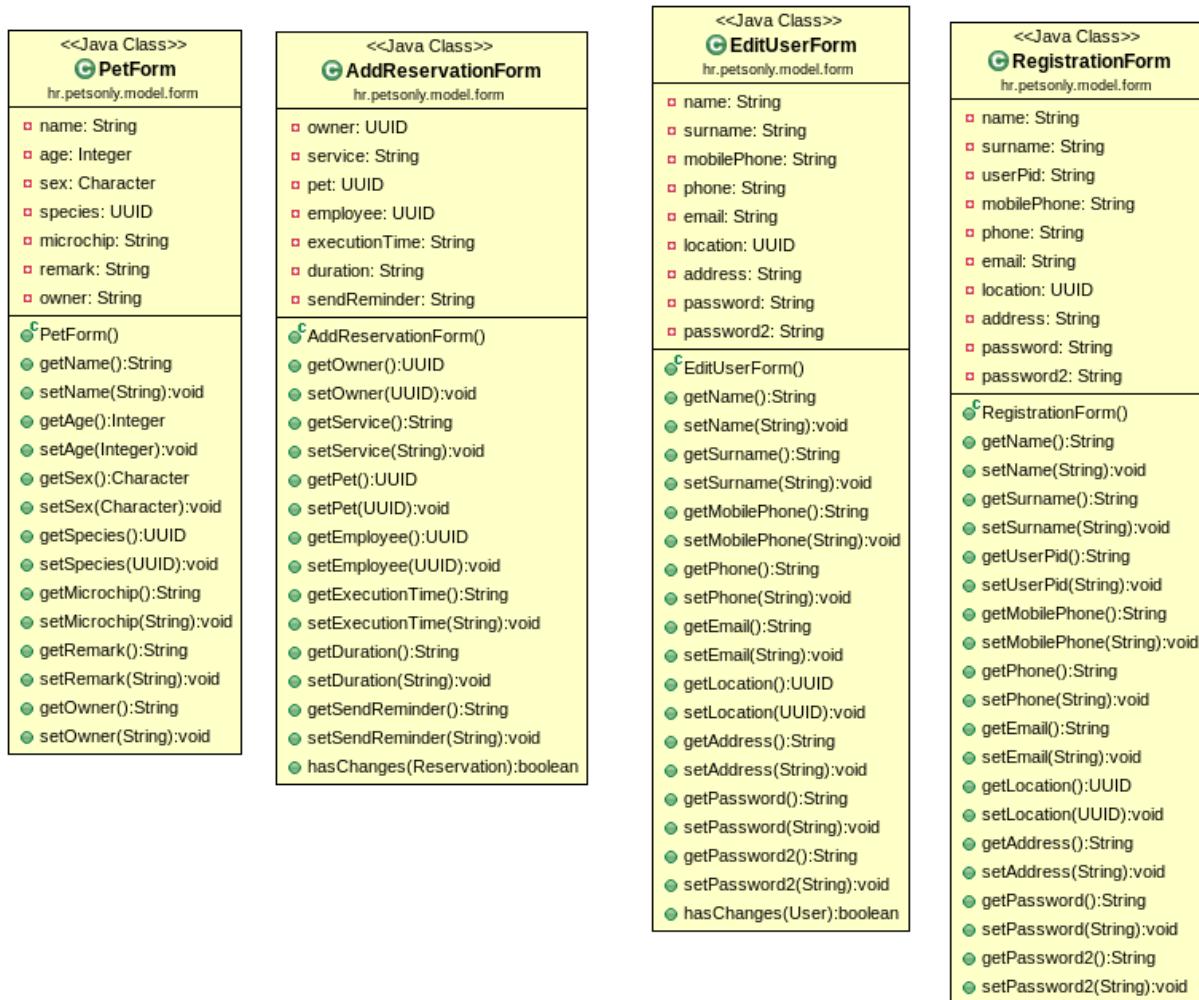


Slika 6.7 – Detaljan UML dijagram razreda modela prezentacijsko sloja

## 6.2.6 Razredi modela formulara

U ovom paketu nalaze se razredi koji modeliraju formulare koji se predaju aplikaciji nakon interakcije s korisnicima.

- PetForm - formular za klasu pet
- AddReservationForm – formular za klasu Reservation
- EditUserForm – formular za klasu User
- RegistrationForm - formular za klasu Registration



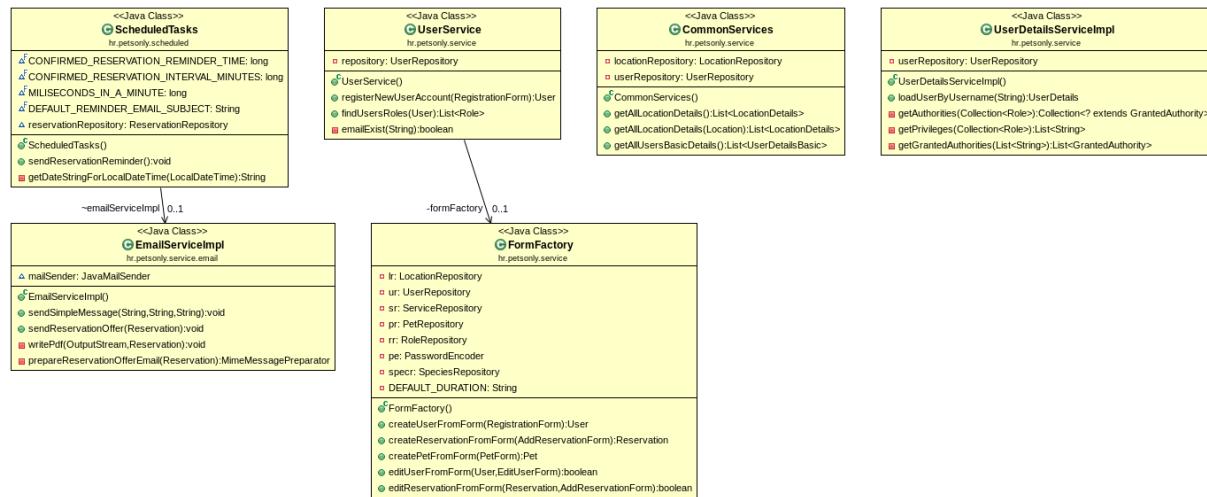
Slika 6.8 – Detaljan UML dijagram razreda modela formulara

## 6.2.7 Razredi usluga

U ovom paketu nalaze se razredi koji implementiraju sve dodatne funkcionalnosti potrebne razredima upravitelja. Paket čine:

- ScheduledTasks - periodički obavlja zadatke poslužitelja
- EmailService - pruža usluge elektroničke pošte

- **UserService** – pruža usluge vezane uz klasu User
- **FormFactory** - pruža usluge stvaranja objekata iz Form klase
- **CommonServices** – pruža općenite usluge
- **UserDetailsServiceImpl** – pruža usluge klasi UserDetails

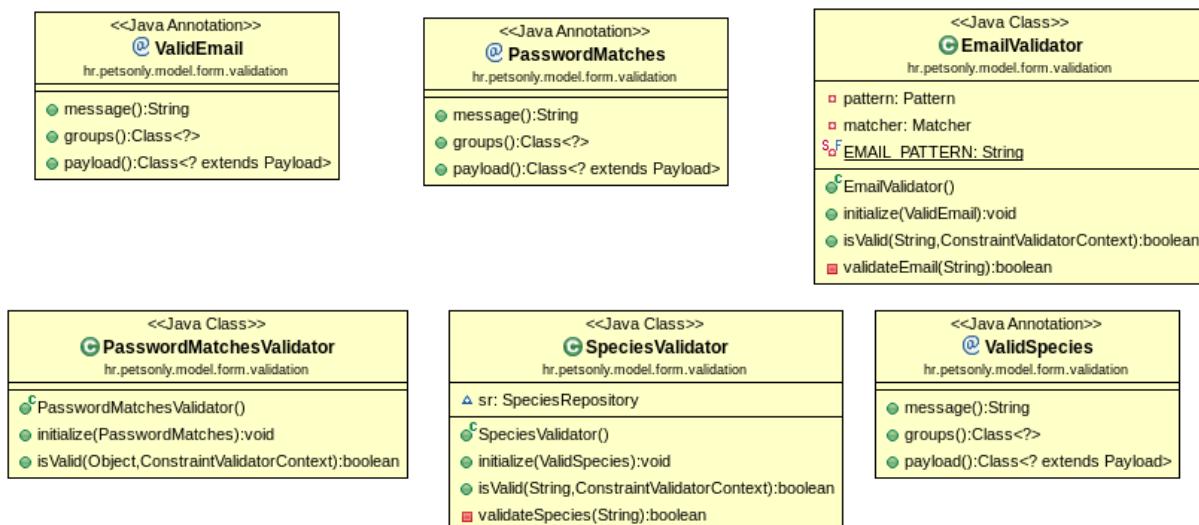


Slika 6.9 - Detaljan UML dijagram pomoćnih razreda

## 6.2.8 Validacijski razredi

Razredi ovog paketa omogućavaju validnost podataka prije no što su smješteni u druge klase, kako bi se osiguralo transparentno preslikavanje između objekata i entiteta.

- **ValidEmail** - anotacija za provjeru adrese elektroničke pošte
- **EmailValidator** – klasa koja provjerava adresu elektroničke pošte
- **PasswordMatches** – anotacija za provjeru istovjetnosti lozinke
- **PasswordMatchesValidator** - klasa koja provjerava istovjetnost lozinke
- **ValidSpecies** – anotacija za provjeru vrste životinje
- **SpeciesValidator** – klasa koja provjerava postojanje vrste životinje

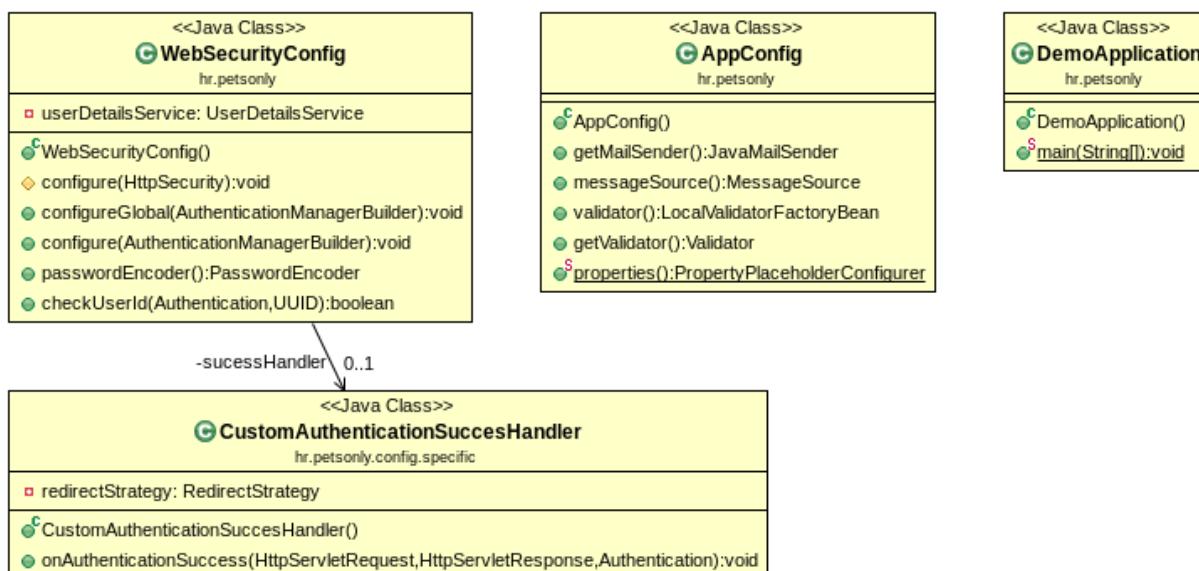


Slika 6.10 – Detaljan UML dijagram razreda za validaciju

### 6.2.9 Konfiguracijski razredi

Razredi ovog paketa zaduženi su za postavljanje parametara aplikacije za sigurnost, autentifikaciju te usluge koje aplikacija nudi.

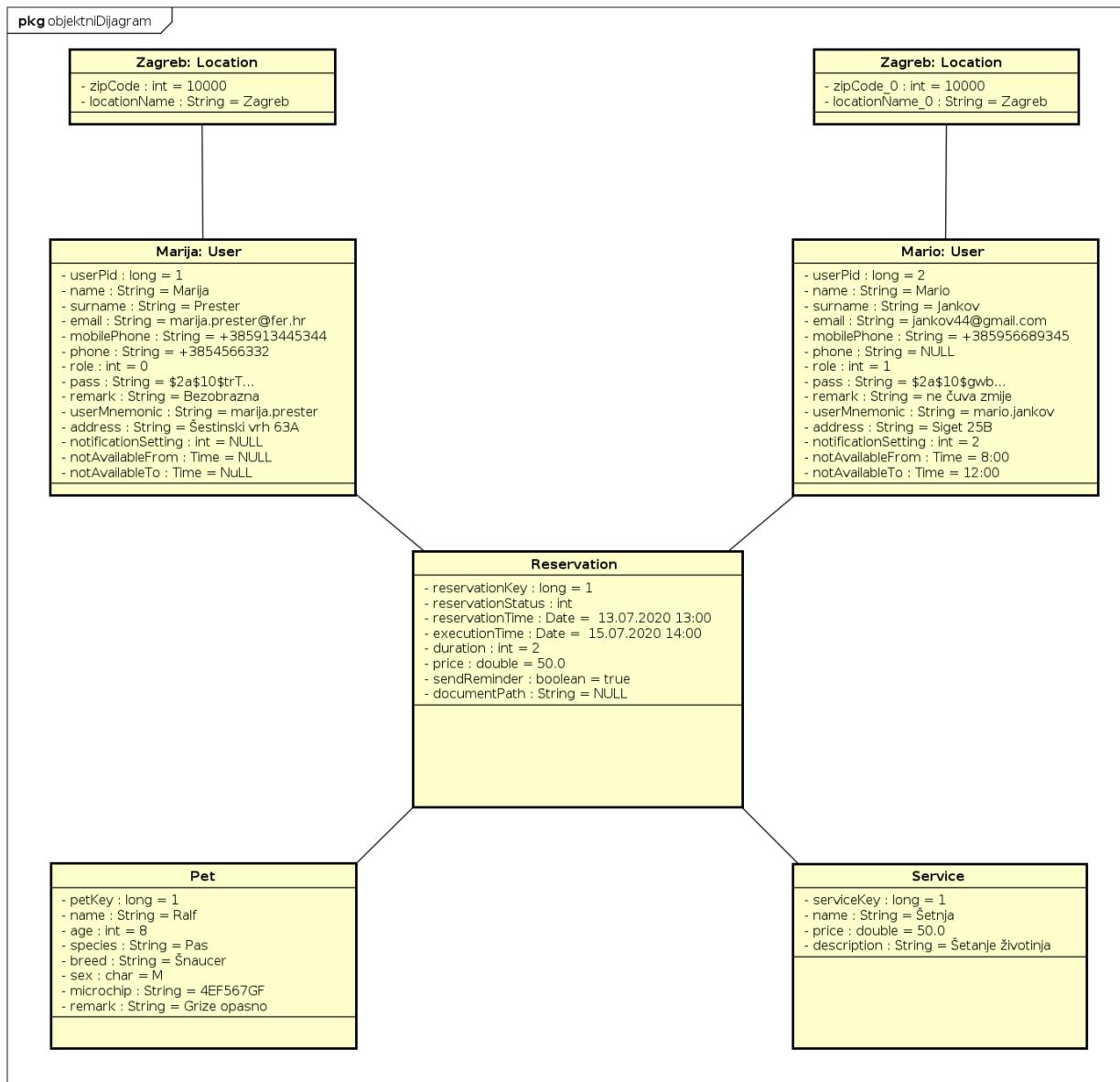
- AppConfig - klasa nadležna za osnovne parametre aplikacije i usluga
- CustomAuthenticationSuccessHandler – klasa za potvrđivanje uspješne autentifikacije
- WebSecurityConfig – klasa nadležna za postavljanje parametara web sigurnosti
- DemoApplication - klasa koja pokreće aplikaciju



Slika 6.11 – Detaljan UML dijagram razreda za konfiguracije

### 6.3 Dijagram objekata

Dijagram prikazuje odnos objekata u sustavu nakon što rezervaciju, koju je za svog psa Ralfa napravila klijentica Marija Prester, preuzme zaposlenik Mario Jankov.



Slika 6.9 - UML dijagram objekata

## 6.4 Ostali UML dijagrami

### 6.4.1 Dijagrami stanja

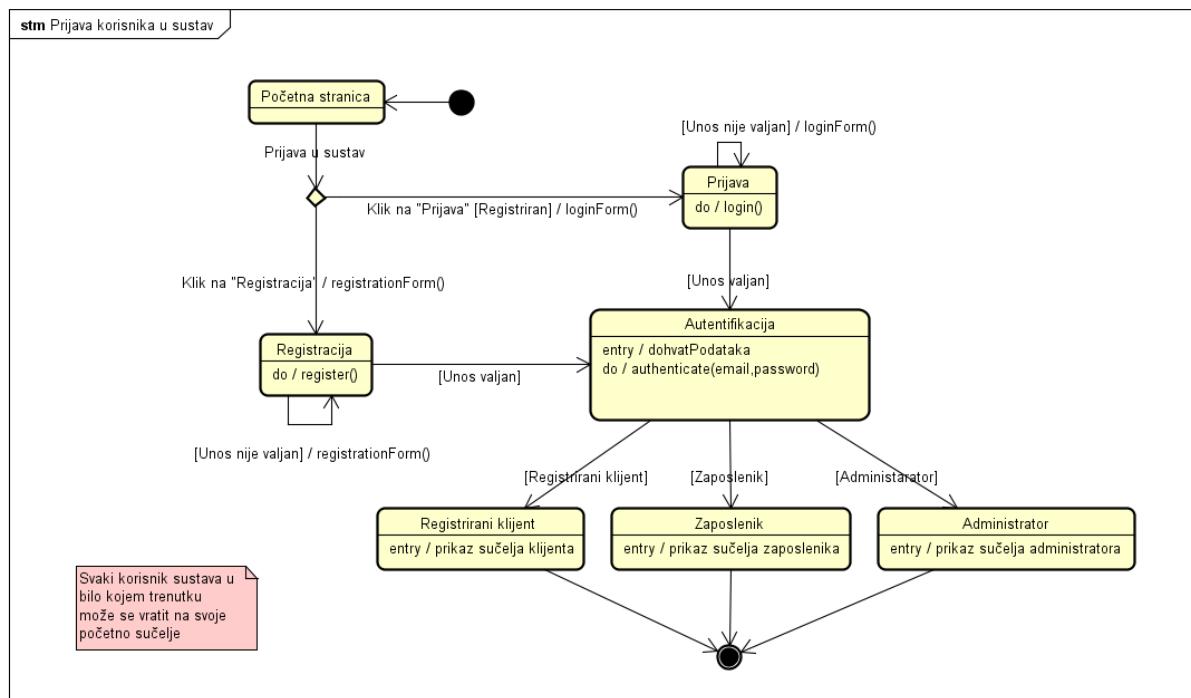
Dijagrami prikazuju konkretna stanja sustava i njegovo dinamičko ponašanje. Iz takvih dijagrama možemo očitati direktne prijelaze između pojedinih stanja sustava te što je uzrokovalo pojedini prijelaz. Možemo ih podijeliti u dvije skupine:

Generički:

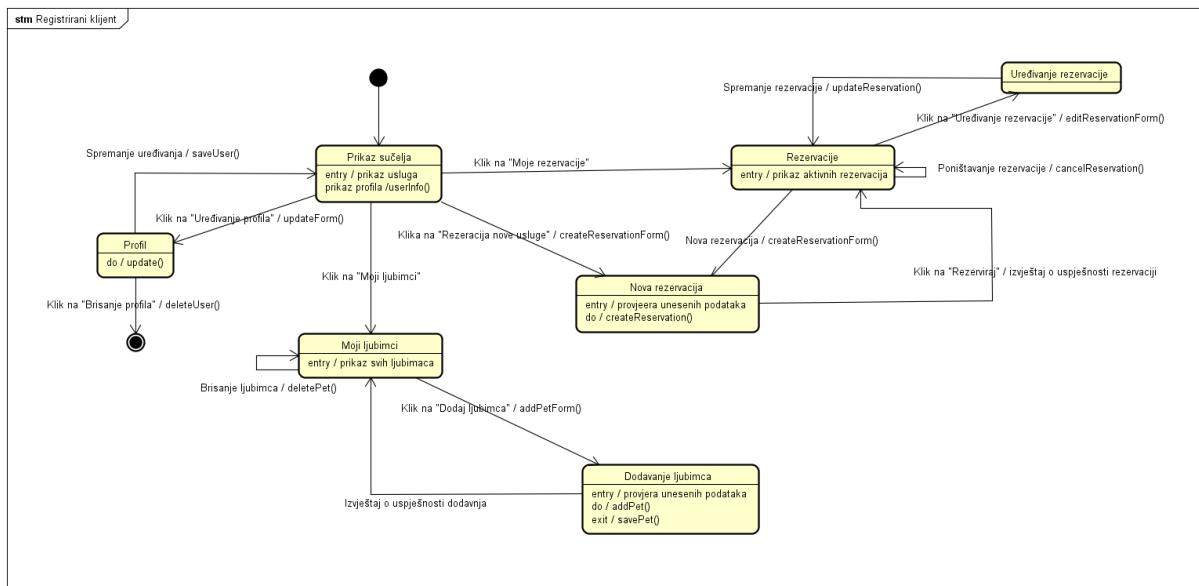
- Prijava korisnika u sustav - Slika 6.10
- Odjava iz sustava - Slika 6.14

Posebni:

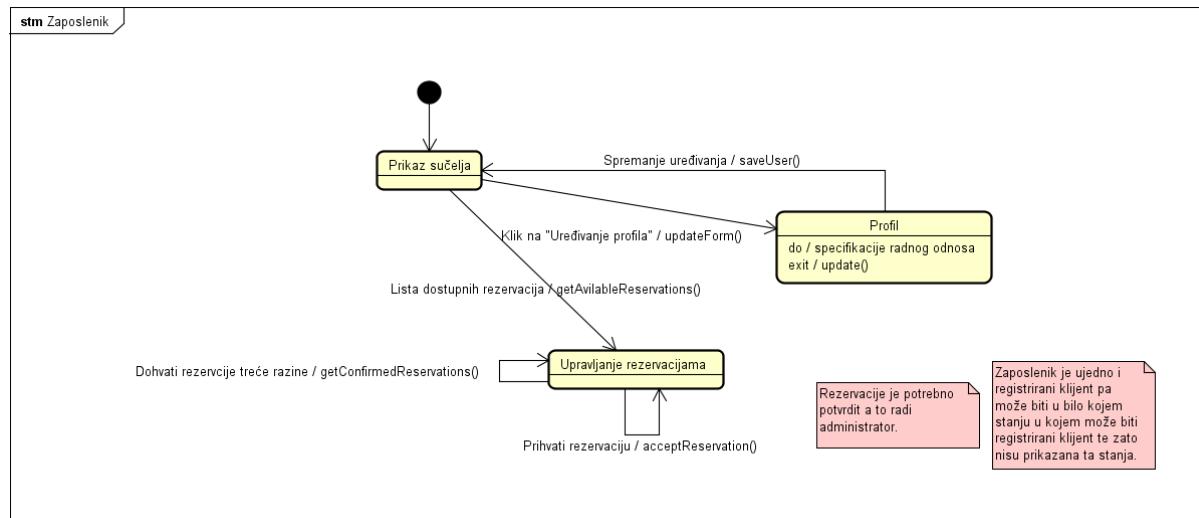
- Dijagram stanja registriranog klijenta - Slika 6.11
- Dijagram stanja zaposlenika - Slika 6.12
- Dijagram stanja administratora - Slika 6.13



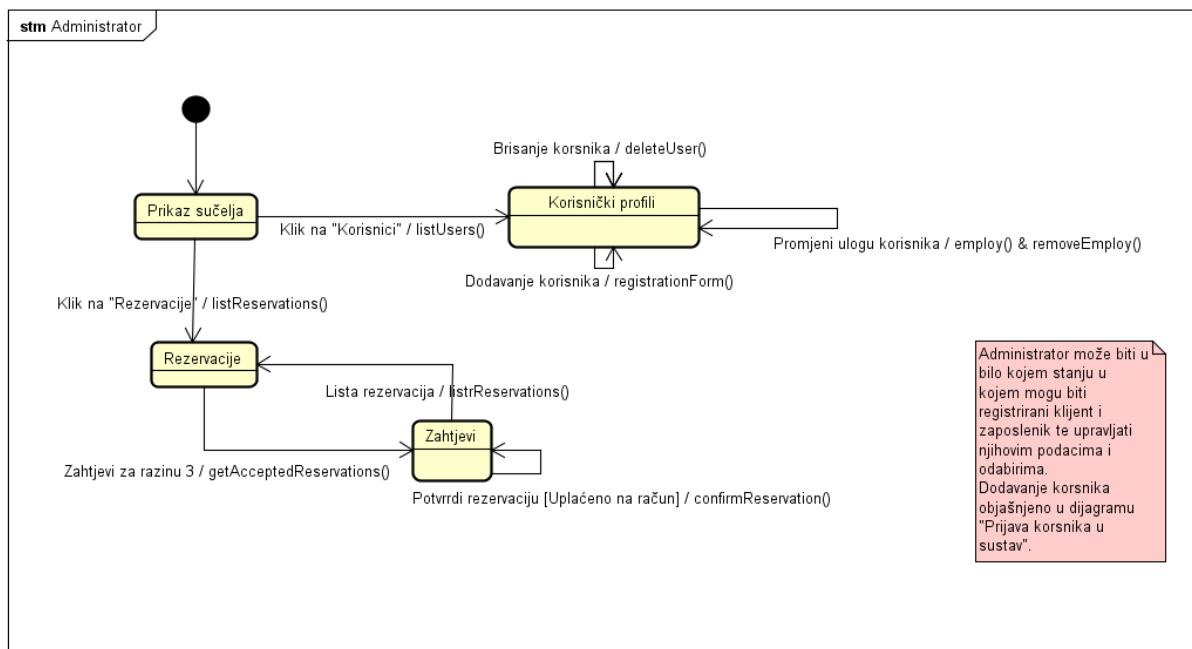
Slika 6.10 - Prijava korisnika u sustav



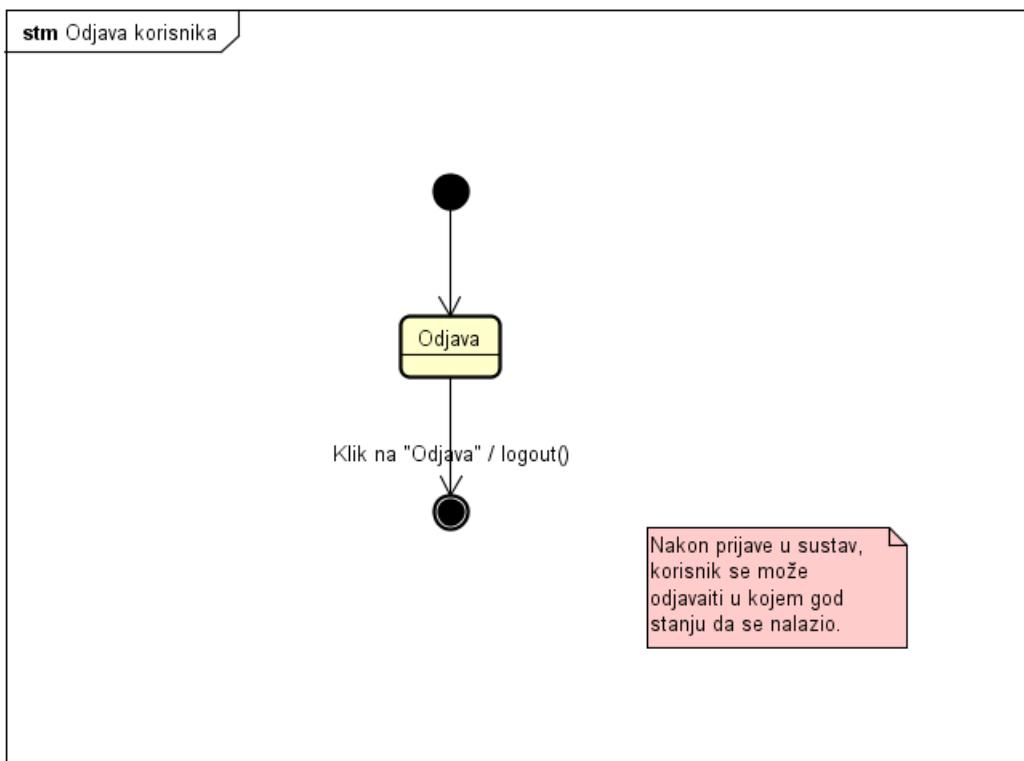
Slika 6.11 - Dijagram stanja registriranog klijenta



Slika 6.12 - Dijagram stanja zaposlenika



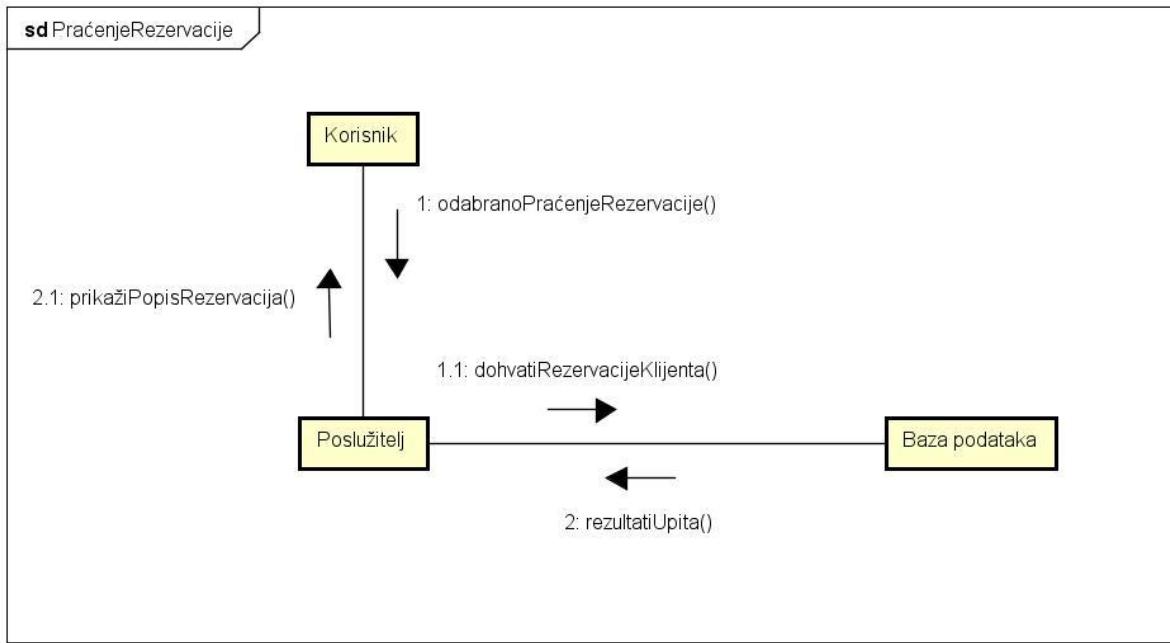
Slika 6.13 - Dijagram stanja administratora



Slika 6.14 - Odjava iz sustava

### 6.4.2 Komunikacijski dijagram

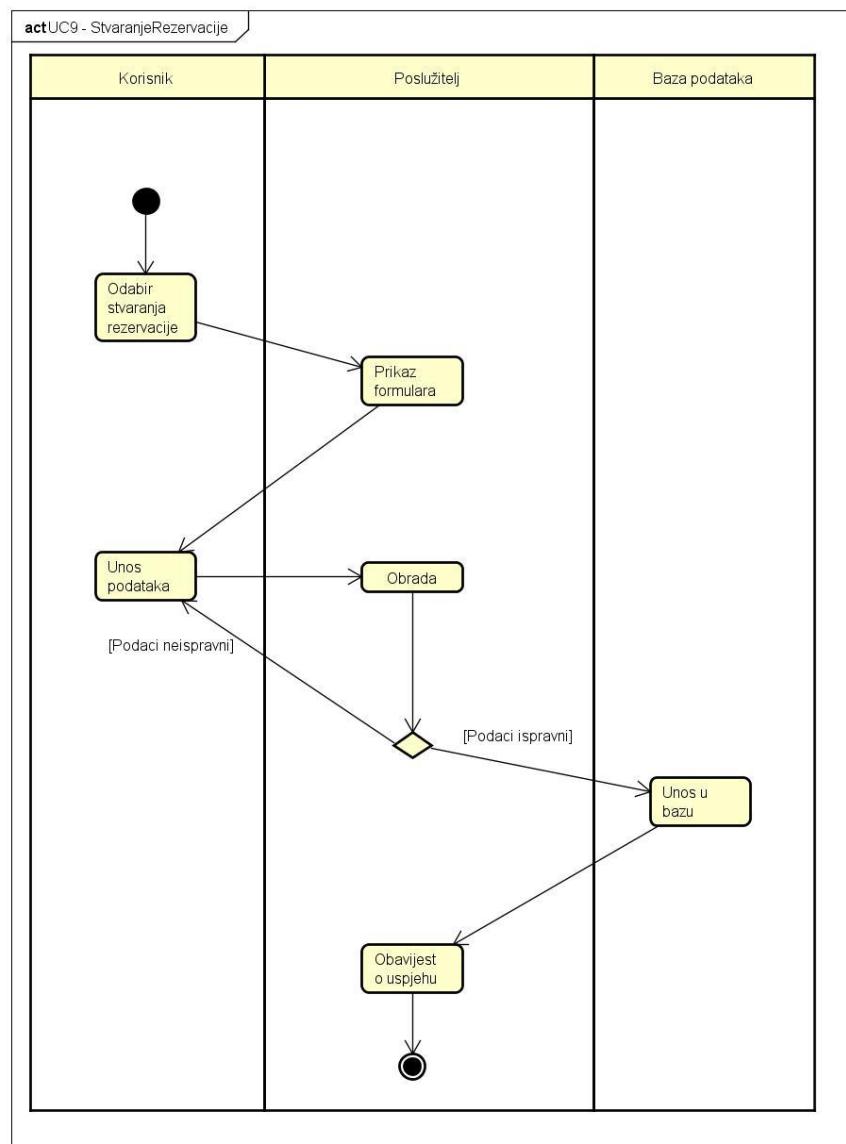
Komunikacijski dijagram prikazan slikom 6.15 prikazuje vremenski redoslijed međudjelovanja sudionika u sustavu kada korisnik odabere praćenje vlastitih rezervacija. Na korisnikov zahtjev poslužitelj dohvaća rezervacije koje pripadaju tom klijentu iz baze podataka. Baza podataka vraća rezultat upita poslužitelju koji potom tražene podatke dostavlja korisniku.



Slika 6. 15 - Praćenje rezervacije

### 6.4.3 Dijagram aktivnosti

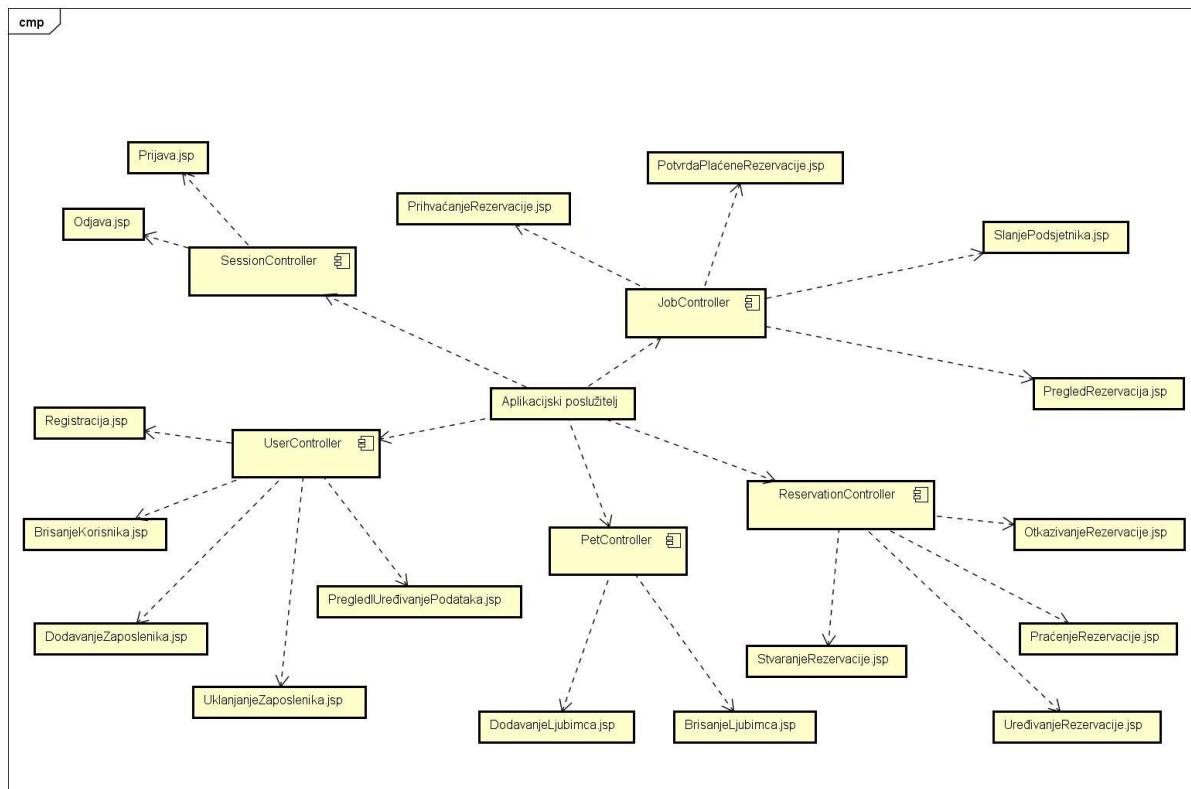
Dijagram aktivnosti prikazan slikom 6.16 prikazuje tok aktivnosti prilikom stvaranja rezervacije. Kada korisnik odabere opciju stvaranja nove rezervacije poslužitelj odgovara prikazom formulara u koji korisnik unosi potrebne podatke. Poslužitelj obrađuje podatke i traži ponavan upis podataka ukoliko su uneseni neispravni podaci ili neki od neophodnih podataka nedostaju. Kada su uneseni podaci ispravni spremaju se u bazu podataka koja odgovara obaviješću o uspjehu.



Slika 6.16 – Stvaranje rezervacije

#### 6.4.4 Dijagram komponenti

Slikom 6.17 prikazan je dijagram komponenti pomoću kojeg je predviđena struktura sustava. U središtu dijagrama komponenti nalazi se aplikacijski poslužitelj te upravljači koji određuju rad aplikacije. Svaki od upravljača zaslužan je za akcije koje zajedno čine smislenu cjelinu, primjerice, SessionController služi za prijavu i odjavu korisnika, JobController služi zaposlenicima kako bi mogli pregledavati nepreuzete poslove, svoje poslove, prihvataći rezervacije, potvrđivati plaćene rezervacije te kako bi klijentima i zaposlenicima bio poslan podsjetnik ukoliko to žele, UserController omogućava registraciju u sustav, manipulaciju korisnikovom ulogom u sustavu, odnosno dodavanje i brisanje zaposlenika, te uređivanje podataka i brisanje korisnika, PetController zadužen je za omogućavanje dodavanja i brisanja kućnog ljubimca, a ReservationController za stvaranje, uređivanje, praćenje i brisanje rezervacija.

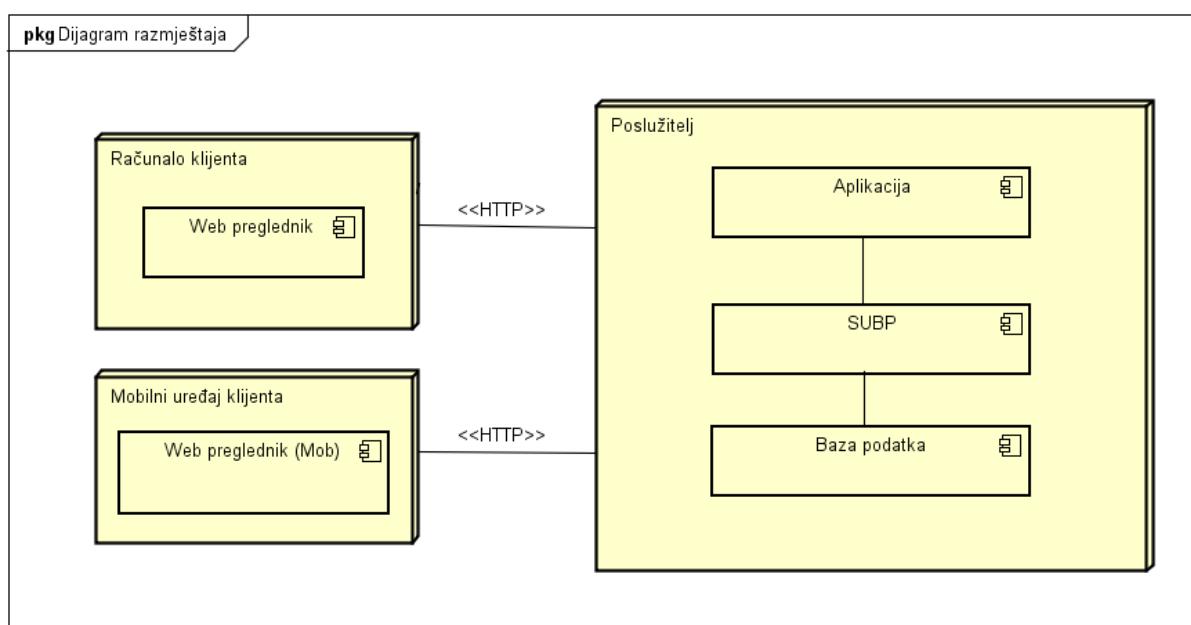


Slika 6. 17 – Dijagram komponenti

## 7. Implementacija i korisničko sučelje

### 7.1 Dijagram razmještaja

Arhitekturu sustava web aplikacije prikazujemo dijagramom razmještaja. Osnove ovakvog sustava čini komunikacija između klijentskih uređaja i poslužitelja. Na poslužitelju se nalazi sav programski kod i logika koja je potrebna za pravilan rad web aplikacije. Sva ta logika, za spremanje podataka, koristi bazu podataka s kojom se upravlja preko sustava za upravljanje bazom podataka (SUBP). Kao i sama aplikacija, SUBP i baza podataka se nalaze na poslužitelju. S druge strane, klijent preko web preglednika na svojim uređajima može koristiti aplikaciju i upravljati određenim podacima.



Slika 7.1. – Dijagram razmještaja

### 7.2 Korištene tehnologije i alati

#### 7.2.1 Eclipse

Eclipse je integrirano razvojno okruženje koje razvija Eclipse Foundation. U vrijeme pisanja, ono je najpopularnija razvojna okolina za programski jezik Javu. No, također omogućuje razvoj aplikacija za velik broj drugih jezika koji se danas koriste. Softver je u domeni otvorenog koda te slobodno dostupan svima. Više informacija može se naći na stranicama <https://www.eclipse.org/home/index.php>

#### 7.2.2 IntelliJ IDEA

IntelliJ IDEA je integrirano razvojno okruženje za razvoj Java aplikacija, koje razvija JetBrains. Dostupan je pod licencom otvorenog koda te pod plaćenom licencom koja nudi naprednije mogućnosti. Više informacija može se naći na stranicama <https://www.jetbrains.com/idea/>

### 7.2.3 Spring

Spring je radni okvir koji omogućava konfiguraciju i nudi brojne usluge za Java aplikacije. Zasniva se na nekoliko temeljnih mogućnosti, kao što su *Dependency Injection* (način upravljanja ovisnostima), podrška za JDBC (Java Database Connectivity), JPA (Java Persistence API), itd. Nadalje, Spring Security omogućava sofisticirano upravljanje autorizacijom i autentifikaciju te sprječavanje mogućih napada na stranicu. Spring MVC (*Model – View -Controller*) radni okvir za web omogućava logičku podjelu koda po MVC obrascu. Spring MVC izgrađen je Servlet sučelju. Spring Dana JPA nudi funkcionalnosti pristupnog sloja prema bazi podataka i olakšava korištenje implementacija JPA sučelja. Verzija korištena u projektu je 5.0.2. Više informacija može se naći na stranicama <https://projects.spring.io/spring-framework/>

### 7.2.4 MySql Community Server

MySql server je upravitelj relacijskim bazama podataka otvorenog koda. Trenutno je u vlasništvu tvrtke Oracle Corporation, koja osim licence otvorenog koda nudi i plaćene licence s dodatnim mogućnostima. S bazom se komunicira koristeći upitni jezik SQL (*Structured query language*). Verzija korištena u ovom projektu je 5.7.21. Više informacija može se naći na stranicama <https://www.mysql.com/>

### 7.2.5 Hibernate ORM

Hibernate ORM je radni okvir koji nudi objektno-relacijsko preslikavanje klasa u entitete te obrnuto. Također oslobađa programere od direktnog komuniciranja s bazom podataka i pisanja bilokakvog SQL koda. Uz to, nudi mogućnost validacije klasa preko svog sučelja, kako bi se održao integritet mapiranja između programske kode i baze podataka. Verzija korištena u ovom projektu je 5.4.1. Više informacija može se naći na stranicama <http://hibernate.org/>

### 7.2.6 JSON

Json (Javascript object notation) je format za razmjenu podataka čitljiv ljudima, no isto tako pogodan za parsiranje i korištenje pomoću računala. Prvotno razvijen za javascript, danas se koristi neovisno o jeziku. Više informacija može se naći na stranicama <https://www.json.org/>

### 7.2.7 Apache Tomcat

Apache Tomcat je web poslužitelj otvorenog koda koji implementira razne Java tehnologije poput Java Servlet, JavaServer Pages, Java WebSocket itd. Razvija ga Apache Software Foundation. Više informacija može se naći na stranicama <http://tomcat.apache.org/>

### 7.2.8 Semantic UI

Semantic UI je radni okvir za tematski razvoj web stranica. Omogućava brz i lagan razvoj elemenata u HTML-u i Javascriptu te olakšava traženje grešaka u kodu. Verzija korištena u ovom projektu je 2.2 Više informacija može se naći na stranicama <https://semantic-ui.com/>

### 7.2.9 jQuery

JQuery je biblioteka otvorenog koda koja pojednostavljuje skriptiranje HTML-a na korisničkom računalu. Biblioteka je otvorenog koda i višeplatformska. Razvija ju jQuery Team. Više informacija može se naći na stranicama <https://jquery.com/>.

## 7.3 Isječak programskog koda vezan za temeljnu funkcionalnost sustava

### 7.3.1 Kod potreban za funkcionalnost Spring-Boot aplikacije

#### 7.3.1.1 Osnovna konfiguracija

U ovom razredu se podešavaju postavke za slanje elektroničke pošte, postavke za referenciranje poruka prema korisniku koje se nalaze izvan koda, u posebnim datotekama.

```

19 @Configuration
20 public class AppConfig extends WebMvcConfigurerAdapter {
21
22    @Bean
23    public JavaMailSender getMailSender() {
24        JavaMailSenderImpl mailSender = new JavaMailSenderImpl();
25
26        mailSender.setHost("smtp.gmail.com");
27        mailSender.setPort(587);
28        mailSender.setUsername("fau53t7zss@gmail.com");
29        mailSender.setPassword("petsonlyzg");
30
31        Properties javaMailProperties = new Properties();
32        javaMailProperties.put("mail.smtp.starttls.enable", "true");
33        javaMailProperties.put("mail.smtp.auth", "true");
34        javaMailProperties.put("mail.transport.protocol", "smtp");
35        javaMailProperties.put("mail.debug", "true");// Prints out everything on screen
36
37        mailSender.setJavaMailProperties(javaMailProperties);
38        return mailSender;
39    }
40
41    @Bean
42    public MessageSource messageSource() {
43        ReloadableResourceBundleMessageSource bean = new ReloadableResourceBundleMessageSource();
44        bean.setBasename("classpath:messages");
45        bean.setDefaultEncoding("UTF-8");
46        return bean;
47    }
48
49    @Bean
50    public LocalValidatorFactoryBean validator() {
51        LocalValidatorFactoryBean bean = new LocalValidatorFactoryBean();
52        bean.setValidationMessageSource(messageSource());
53        return bean;
54    }
55
56    @Override
57    public org.springframework.validation.Validator getValidator() {
58        return validator();
59    }
60
61    @Bean
62    public static PropertyPlaceholderConfigurer properties() {
63        final PropertyPlaceholderConfigurer ppc = new PropertyPlaceholderConfigurer();
64        ppc.setIgnoreResourceNotFound(true);
65
66        final List<Resource> resourceLst = new ArrayList<Resource>();
67
68        resourceLst.add(new ClassPathResource("defaults.properties"));
69
70        ppc.setLocations(resourceLst.toArray(new Resource[] {}));
71
72        return ppc;
73    }
74}
75

```

### 7.3.1.2 Konfiguracija sigurnosnog paketa Spring-Security

Podešavanje sigurnosnih mjera. U 39. retku vidimo postavljanje putanje za upravljač koji će se baviti prijavom korisnika. Zatim u 41. i 42. retku vidimo postavljanje objekata koji će se pozivati pri uspjehu ili neuspjehu prijave.

U ovom se razredu još postavlja koder/dekoder korisničke lozinke, te servis koji se koristi tijekom sigurnosnog ciklusa.

```

22 @Configuration
23 @EnableWebSecurity
24 @EnableGlobalMethodSecurity(prePostEnabled = true)
25 public class WebSecurityConfig extends WebSecurityConfigurerAdapter {
26
27     @Autowired
28     private UserDetailsService userDetailsService;
29
30     @Autowired
31     private CustomAuthenticationSuccessHandler sucessHandler;
32
33     @Autowired
34     private CustomAuthenticationFailureHandler failureHandler;
35
36     @Override
37     protected void configure(HttpSecurity http) throws Exception {
38         http.formLogin()
39             .loginPage("/sessions/new")
40             .loginProcessingUrl("/sessions/")
41             .failureHandler(failureHandler)
42             .successHandler(sucessHandler);
43
44         http.csrf().disable();
45     }
46
47     @Override
48     public void configure(AuthenticationManagerBuilder builder) throws Exception {
49         builder.userDetailsService(userDetailsService).passwordEncoder(passwordEncoder());
50     }
51
52     @Bean
53     public PasswordEncoder passwordEncoder() {
54         return new BCryptPasswordEncoder(11);
55     }
56
57     public boolean checkUserId(Authentication auth, UUID id) {
58         Object o = auth.getPrincipal();
59         if (!(o instanceof CustomUserDetails))
60             return false;
61
62         CustomUserDetails user = (CustomUserDetails) o;
63
64         if (user.getRoles().contains("ROLE_ADMINISTRATOR"))
65             return true;
66         if (!user.getUserId().equals(id))
67             return false;
68
69         return true;
70     }
71 }
72 }
```

### 7.3.1.3 Pokretanje Spring-Boot aplikacije

```

7 @SpringBootApplication
8 @EnableScheduling
9 public class DemoApplication {
10
11     public static void main(String[] args) {
12         SpringApplication.run(DemoApplication.class, args);
13     }
14 }
```

### 7.3.2 Primjer Upravitelja

Prilikom izrade aplikacije koja koristi Spring, upravljači se povezuju sa putanjama u unutarnjem postupku Spring razvijačke okoline. U ovom slučaju smo definirali jednu vršnu putanju koja će se odnositi na sve metode ovog, a zatim svaka metoda ima ostatak putanje kako bi se mogli razlikovati svi pozivi određenih metoda.

```

31 @Controller
32 @RequestMapping("/users/{id}/pets")
33 public class PetController {
34
35     @Autowired
36     private FormFactory formFactory;
37
38     @Autowired
39     private PetRepository petRepository;
40
41     @ResponseBody
42     @GetMapping
43     public List<PetDetails> showPetList(Model model, @PathVariable UUID id) {
44
45         List<Pet> petList = petRepository.findByOwnerId(id.toString());
46         System.out.println(Arrays.toString(petList.toArray()));
47
48         List<PetDetails> petDetails = new ArrayList<>();
49
50         petList.forEach(pet -> {
51             petDetails.add(new PetDetails(pet));
52         });
53
54         return petDetails;
55     }
56
57     @GetMapping("/new")
58     public String showNewPetForm() {
59         return "addPet";
60     }
61
62     @ResponseBody
63     @PostMapping(consumes = MediaType.APPLICATION_JSON_VALUE)
64     public ResponseEntity<PetDetails> addNewPet(@RequestBody @Valid PetForm petForm,
65                                                 BindingResult result, Model model, @PathVariable UUID id) {
66         if (result.hasErrors()) {
67             System.out.println(result);
68             return new ResponseEntity<>(HttpStatus.BAD_REQUEST);
69         }
70
71         petForm.setOwner(id.toString());
72         Pet pet = formFactory.createPetFromForm(petForm);
73
74         pet = petRepository.save(pet);
75         PetDetails petDetails = new PetDetails(pet);
76
77         return new ResponseEntity<>(petDetails, HttpStatus.OK);
78     }
79
80     @ResponseBody
81     @DeleteMapping("/{petId}")
82     public ResponseEntity<PetDetails> deletePet(Model model, @PathVariable UUID petId) {
83
84         Pet pet = petRepository.findOne(petId);
85         if (pet == null) {
86             System.out.println("Ljubimac nije pronađen!");
87             return new ResponseEntity<>(HttpStatus.BAD_REQUEST);
88         }
89
90         PetDetails petDetails = new PetDetails(pet);
91         petRepository.delete(petId);
92
93         return new ResponseEntity<>(petDetails, HttpStatus.OK);
94     }
95 }
```

## 7.4 Primjer rezitorija

U ovom razredu definiramo metode koje će iz baze uzimati podatke. Pošto ovaj razred nasljeđuje JPA rezitorij, možemo u imenima metoda u određenom obliku samo upisati što želimo, a interno će se one razriješiti da rade točno što smo htjeli. Dakako, možemo implementirati i svoje metode uz pomoć anotacija, kao što smo napravili u recima 63, 72 i 77.

```

14 @Service
15 public interface UserRepository extends JpaRepository<User, UUID> {
16
17     User findByUserId(String userId);
18
19     List<User> findAllByName(String name);
20
21     List<User> findAllByNameIgnoreCase(String name);
22
23     List<User> findAllByNameLike(String name);
24
25     List<User> findAllBySurname(String surname);
26
27     List<User> findAllBySurnameIgnoreCase(String surname);
28
29     List<User> findAllBySurnameLike(String surname);
30
31     User findByMobilePhone(String mobilePhone);
32
33     User findByMobilePhoneLike(String mobilePhone);
34
35     List<User> findAllByPhone(String phone);
36
37     List<User> findAllByPhoneLike(String phone);
38
39     User findByEmail(String email);
40
41     List<User> findAllByEmailLike(String email);
42
43     List<User> findAllByRemark(String remark);
44
45     List<User> findAllByRemarkIgnoreCase(String remark);
46
47     List<User> findAllByRemarkLike(String remark);
48
49     List<User> findAllByUserMnemonic(String userMnemonic);
50
51     List<User> findAllByUserMnemonicLike(String userMnemonic);
52
53     List<User> findAllByAddress(String address);
54
55     List<User> findAllByAddressIgnoreCase(String address);
56
57     List<User> findAllByAddressLike(String address);
58
59     List<User> findAllByLocation(Location location);
60
61     List<User> findAllByNotificationSetting(int notificationSetting);
62
63     @Query(value = "SELECT COUNT(u.user_id) FROM users u WHERE u.user_mnemonic REGEXP :pattern", nativeQuery = true)
64     Long countByUserMnemonic(@Param("pattern") String pattern);
65     default User saveWithMnemonic(User entity) {
66         String pattern = entity.getName() + entity.getSurname();
67         Long num = countByUserMnemonic(pattern);
68         entity.setUserMnemonic(pattern + num);
69         return this.save(entity);
70     }
71
72     @Query(value = "SELECT * FROM users u INNER JOIN users_roles ur ON u.user_id = ur.user_id WHERE role_id = 2", nativeQuery = true)
73     List<User> findAllEmployees();
74
75     @Transactional
76     @Modifying(clearAutomatically = true)
77     @Query(value = "UPDATE users_roles ur SET ur.role_id = 2 WHERE ur.user_id = :user_id", nativeQuery = true)
78     void hireUser(@Param("user_id") String userId);
79 }
```

### 7.4.1 Primjer servisa

Ovdje vidimo servis koji radi neke specifične operacije nad korisnikom. U ovom primjeru vidimo metodu za registraciju novog korisnika, koja zatim koristi metodu za provjera da upisana elektronička pošta već postoji u sustavu.

```
14 @Service
15 public class UserService {
16     @Autowired
17     private UserRepository repository;
18
19     @Autowired
20     private FormFactory formFactory;
21
22     @Transactional
23     public User registerNewUserAccount(RegistrationForm rf) {
24
25         if (emailExist(rf.getEmail())) {
26             return null;
27         }
28
29         User user = formFactory.createUserFromForm(rf);
30
31         return repository.save(user);
32     }
33
34     @Transactional
35     public List<Role> findUsersRoles(User user) {
36         return user.getRoles();
37     }
38
39     private boolean emailExist(String email) {
40         User user = repository.findByEmail(email);
41         if (user != null) {
42             return true;
43         }
44         return false;
45     }
46 }
47 }
```

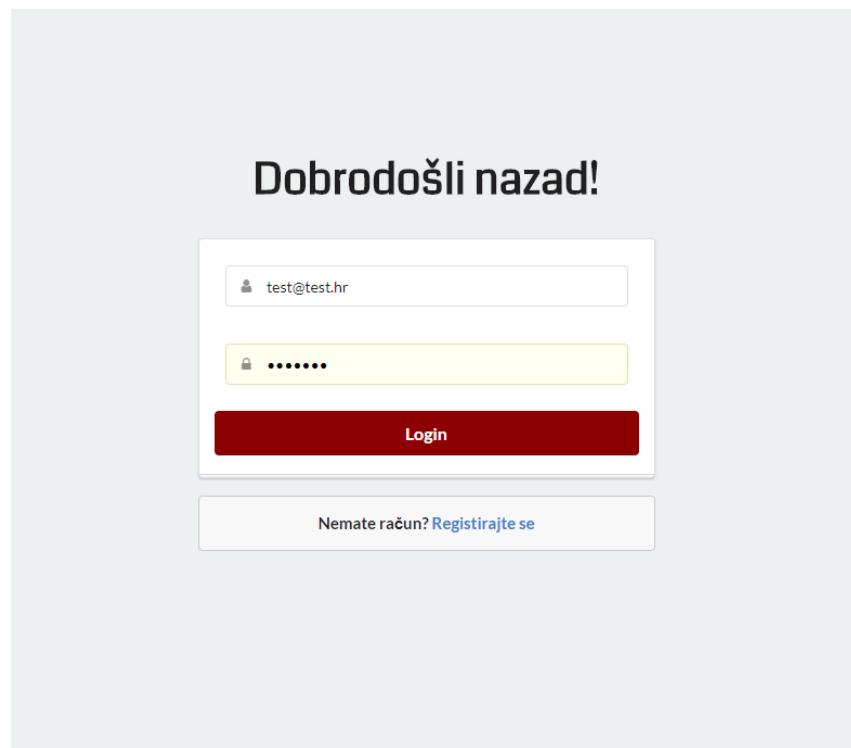
## 7.5 Ispitivanje programskog rješenja

### 1. slučaj: korisnik pristupa web aplikaciji

**Opis:** Korisnik pokušava naručiti uslugu.

**Očekivani rezultat:** Ukoliko korisnik nije prijavljen u sustav prikazuje se forma za prijavu te link na formu za registraciju. U suprotnom, korisniku se prikazuje forma za naručivanje usluge.

**Dobiveni rezultat:** Neprijavljenog korisnika preusmjerava se na obrazac za prijavu, prijavljenog korisnika preusmjerava se na obrazac za rezervaciju usluge.



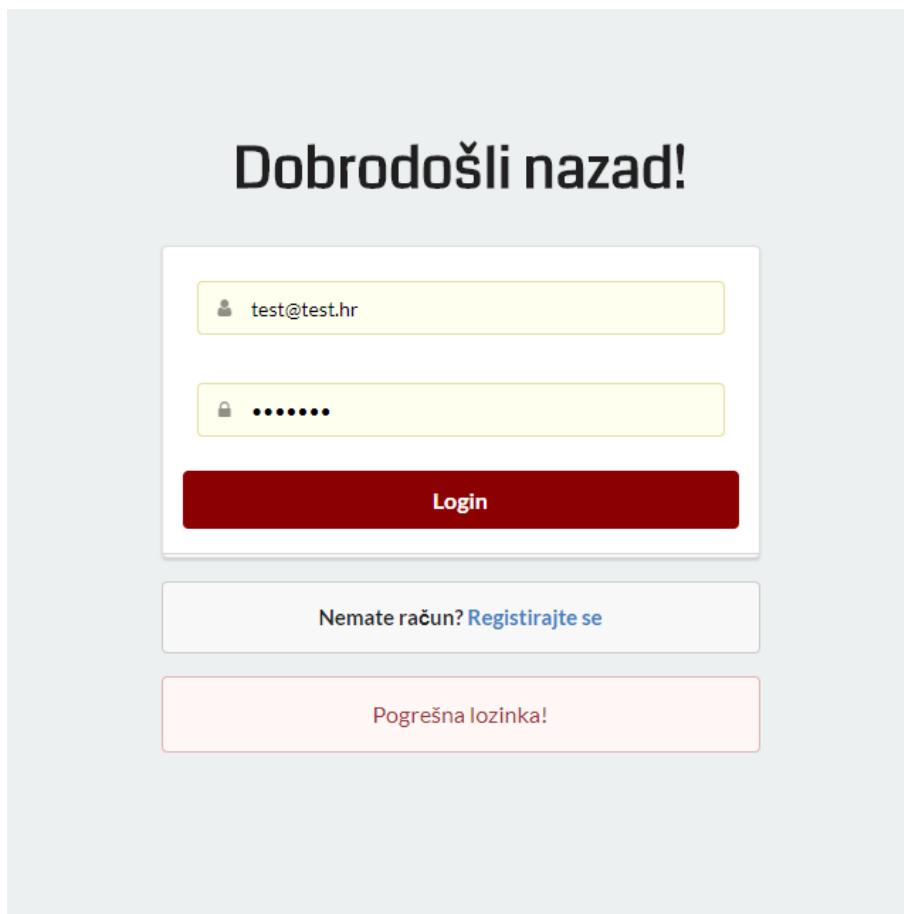
The image shows a reservation creation form titled "Nova rezervacija". It consists of several input fields: a dropdown menu for selecting a service ("Odaberite uslugu"), a dropdown menu for selecting a pet ("Odaberite svog ljubimca"), a date input field for entering the reservation date ("Unesite datum rezervacije" with placeholder "dd.mm.gggg. --::--"), a dropdown menu for selecting an employee ("Odaberite zaposlenika"), and a time input field for specifying the duration of the service ("Vrijeme trajanja usluge" with placeholder "--:--"). There is also a checked checkbox for receiving a reminder via email ("Želim dobiti podsjetnik na mail?"). A large red button at the bottom right is labeled "Napravi rezervaciju!".

## 2. slučaj: korisnik se prijavljuje u sustav

**Opis:** Korisnik unosi adresu elektroničke pošte i lozinku.

**Očekivani rezultat:** Ukoliko su uneseni podaci ispravni korisnik je uspješno prijavljen, u suprotnom se prikazuje poruka o pogrešci.

**Dobiveni rezultat:** Prilikom uspješnog unosa podataka korisnik je preusmjeren na stranicu koja prikazuje njegove osobne podatke te podatke o njegovim ljubimcima i rezervacijama. Pri unosu pogrešnih podataka prikazuje poruka o pogrešci.



### 3. slučaj: registracija korisnika

**Opis:** Korisnik unosi tražene podatke kako bi se registrirao u sustav.

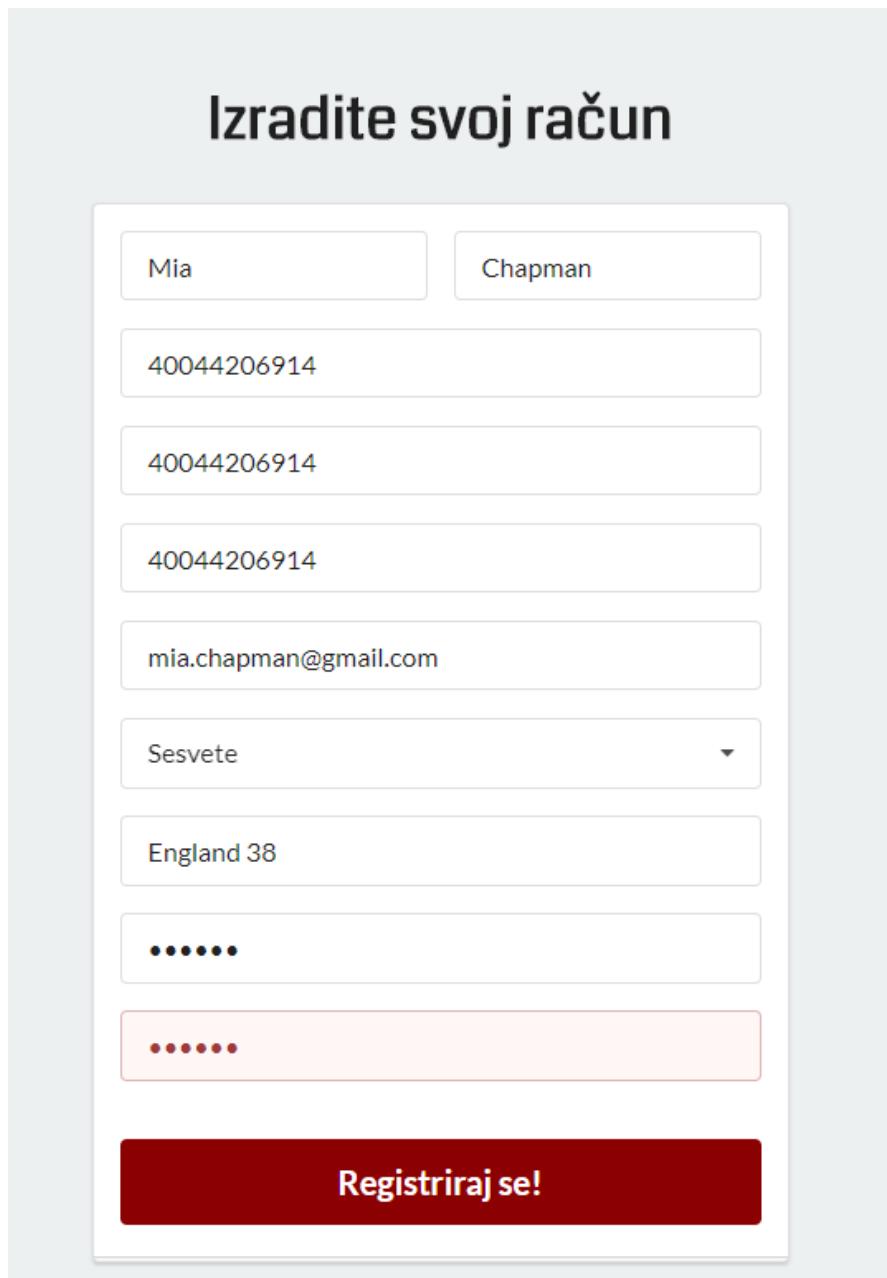
**Očekivani rezultat:** Ukoliko su uneseni podaci ispravni rezervacija je uspješno obavljena i korisnik je preusmjerjen na stranicu koja prikazuje osobne podatke. U slučaju unosa pogrešnih podataka prikazuje se poruka o pogrešci.

**Dobiveni rezultat:** U slučaju unosa pogrešnih podataka označeno je polje u kojem je došlo do pogreške. Ako su uneseni podaci ispravni korisnik je uspješno preusmjerjen na stranicu koja prikazuje njegove osobne podatke.

### Izradite svoj račun

Mia Chapman  
40044206914  
40044206914  
40044206914  
mia.chapman@gmail.com  
Sesvete  
England 38  
.....  
.....

**Registriraj se!**



#### 4. slučaj: dodavanje kućnog ljubimca

**Opis:** Korisnik dodaje vlastitog kućnog ljubimca.

**Očekivani rezultat:** Ako su uneseni svi obvezni podaci ljubimac je uspješno dodan, u suprotnom se prikazuje pogreška.

**Dobiveni rezultat:** Ukoliko nisu uneseni svi obvezni podaci označava se polje za koje uvjet nije zadovoljen, a u suprotnom je korisnik preusmjerjen na stranicu za prikaz svih njegovih kućnih ljubimaca.

The screenshot shows a modal dialog box titled "Dodaj ljubimca". It contains the following fields:

- Pet Name: Miki
- Age: 9
- Gender: Dečko
- Breed: Vrsta (highlighted in red)
- Microchip Number: Pšenični terijer
- Notes: Napomene...

A large red "Dodaj" (Add) button is located at the bottom of the form. The entire dialog box has a white background and is centered on a dark background.

## 5. slučaj: rezervacija usluge

**Opis:** Korisnik pokušava rezervirati uslugu za nekog od svojih ljubimaca.

**Očekivani izlaz:** Ukoliko su uneseni podaci ispravni rezervacija je uspješno stvorena te je korisnik preusmjeren na stranicu koja prikazuje njegove osobne podatke. U suprotnom, prikazuje se poruka o pogrešci.

**Dobiveni izlaz:** U slučaju uspješne rezervacije korisnik je preusmjeren na njegov profil, a ukoliko neki od podataka nisu uneseni označena su polja u kojima je došlo do pogreške.

### Nova rezervacija

Odaberite uslugu

Miki

Unesite datum rezervacije

22.02.2018. 13:05

Zaposlenik Zaposlenković

Vrijeme trajanja usluge

02:00

Želim dobiti podsjetnik na mail?

**Napravi rezervaciju!**

## 6. slučaj: dodavanje novog zaposlenika (analogno i u slučaju otpuštanja korisnika, uređivanja tuđeg profila te brisanja tuđeg profila)

**Opis:** Korisnik gleda popis korisnika i dodaje novog zaposlenika.

**Očekivani rezultat:** Ukoliko korisnik ima administratorske ovlasti radnja je uspješno obavljena, u suprotnom se prikazuje poruka o pogrešci.

**Dobiveni rezultat:** Pokuša li klijent pristupiti detaljima korisnika prikazuje se poruka o pogrešci uslijed nedostatka ovlasti te ni na koji način ne može zaposliti osobu. Ukoliko administrator obavi istu radnju korisnikova uloga uspješno će biti promijenjena u zaposlenik.

---

## Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Thu Jan 18 20:49:29 UTC 2018

There was an unexpected error (type=Forbidden, status=403).

Access is denied

## 7. slučaj: mijenjanje statusa rezervacije

**Opis:** Zaposlenik prihvata ili potvrđuje rezervaciju.

**Očekivani rezultat:** Ukoliko je korisnik koji obavlja radnju zaposlenik ili administrator radnja je uspješno obavljena, a običan korisnik ne može ni pristupiti popisu svih rezervacija.

**Dobiveni rezultat:** Pokuša li korisnik mijenjajući URL pristupiti popisu svih rezervacija biti će prikazana poruka o pogrešci. Ukoliko korisnik ima ovlasti zaposlenika ili administratora radnje će biti uspješno izvršene.

---

## Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Thu Jan 18 20:49:29 UTC 2018

There was an unexpected error (type=Forbidden, status=403).

Access is denied

## 7.6 Upute za instalaciju

### 7.6.1 Instalacija programske okoline Java

Kako je za izradu aplikacije korišten programski jezik *Java*, za pokretanje i uspješno korištenje aplikacije potrebno je instalirati **JDK**. Za razvoj je korištena *Java 1.8* pa je to ujedno i minimalna verzija koju treba instalirati za uspješan rad aplikacije. Za preuzimanje potrebnih datoteka i upute za instalaciju posjetite službene stranice na adresi <https://java.com/en/download/74>.

### 7.6.2 Instalacija upravitelja projektom *Maven*

Kako bismo izbjegli instalaciju i konfiguraciju poslužitelja kao i instalacije svih ovisnosti projekta, koristimo alat za pomoć pri upravljanju projektom, *Maven*.

Za više informacija i upute za instalaciju posjetite službene stranice na adresi <https://maven.apache.org/>.

### 7.6.3 Instalacija i konfiguracija baze podataka

Za bazu podataka koristi se baza podataka MySQL. Za više informacija i upute za instalacije posjetite službene stranice: <https://www.mysql.com/>.

Nakon instalacije, potrebno je kreirati odgovarajuću *schema* te konfigurirati datoteku *src/main/resources/application.properties*. U nastavku je objašnjenje na primjeru općenite konfiguracije.

```
server.port=<odabrana_vrata>

spring.mvc.view.prefix=/WEB-INF/jsp/
spring.mvc.view.suffix=.jsp

spring.datasource.initialize=true
spring.jpa.hibernate.ddl-auto=create
spring.datasource.url=jdbc:mysql://<adresa_baze_podataka>:<vrata_baze_podataka>/<ime_scheme>
spring.datasource.username=<odabran_korisnicko_ime>
spring.datasource.password=<odabrana_lozinka>
spring.jpa.show-sql=true
```

Navedena konfiguracija zahtjeva postojanje *schema* imena odabranog kao *<ime\_scheme>*, i korisnika s ovlastima pristupa, dodavanja i brisanja nad svim relacijama u *schemi* kojem su korisničko ime i lozinka jednaki onima odabranim kao *<odabran\_korisnicko\_ime>* i *<odabrana\_lozinka>*.

Aplikacija se pokreće na vratima odabranim kao *<odabrana\_vrata>*.

## 7.6.4 Pokretanje aplikacije

Za instalaciju svih ovisnosti brine se *Maven*. Prema tome, sve što je potrebno za uspješno pokretanje aplikacije je stvorena *schema* u funkcionalnoj bazi podataka.

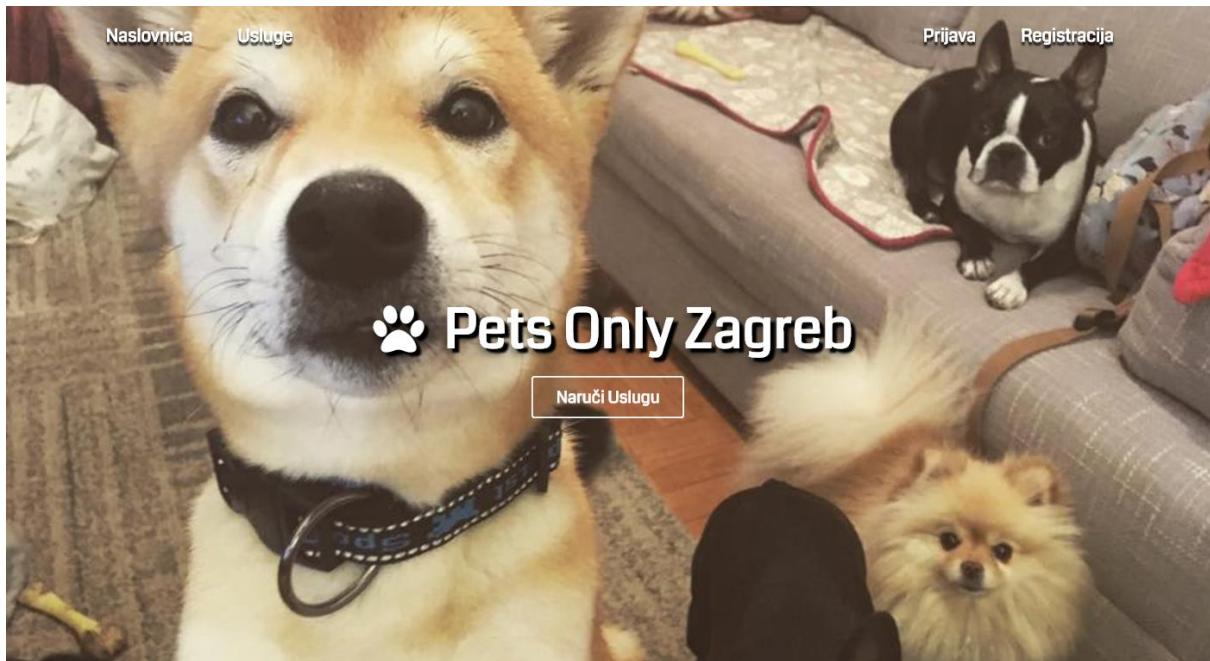
Sama aplikacija se pokreće pozicioniranjem u korijenski direktorij i pokretanjem skripte `./mvnw spring-boot:run`.

## 7.7 Korisničke upute

### 7.7.1 Pregled klijenta

Pri otvaranju aplikacije *PetsOnlyZg*, korisniku se otvara naslovna stranica te mu se nude sljedeće mogućnosti:

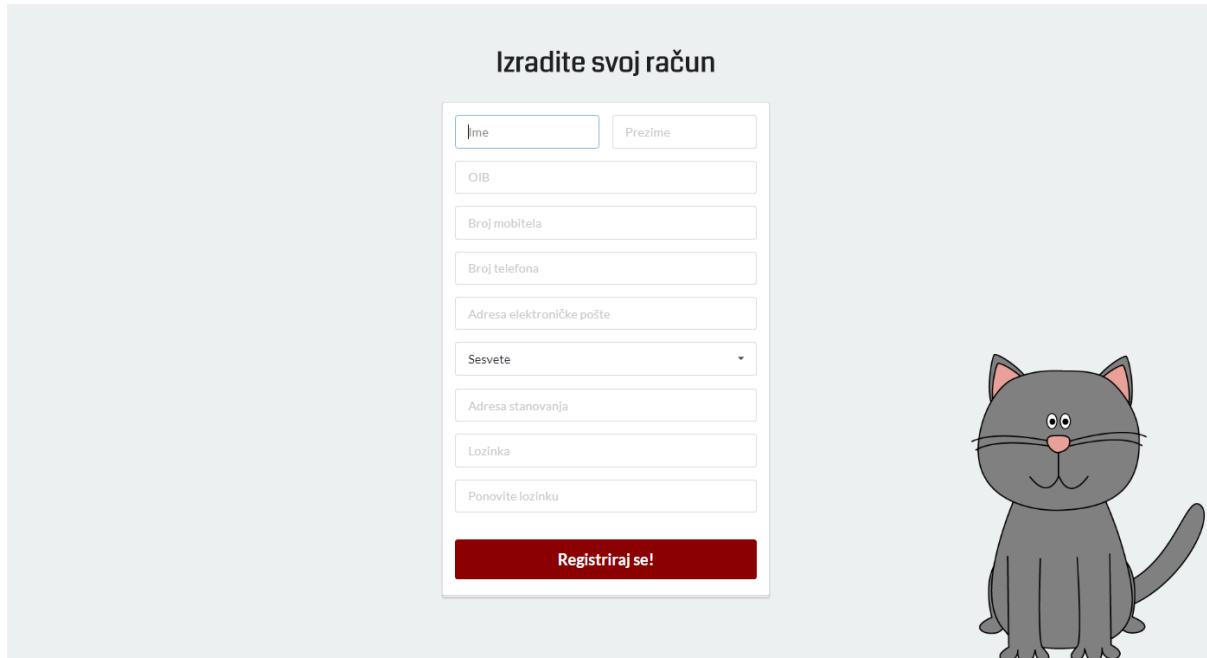
- pregled usluga koje su u ponudi
- registracija korisnika
- prijava korisnika te
- naručivanje usluge



Slika 7.2 - Naslovna stranica

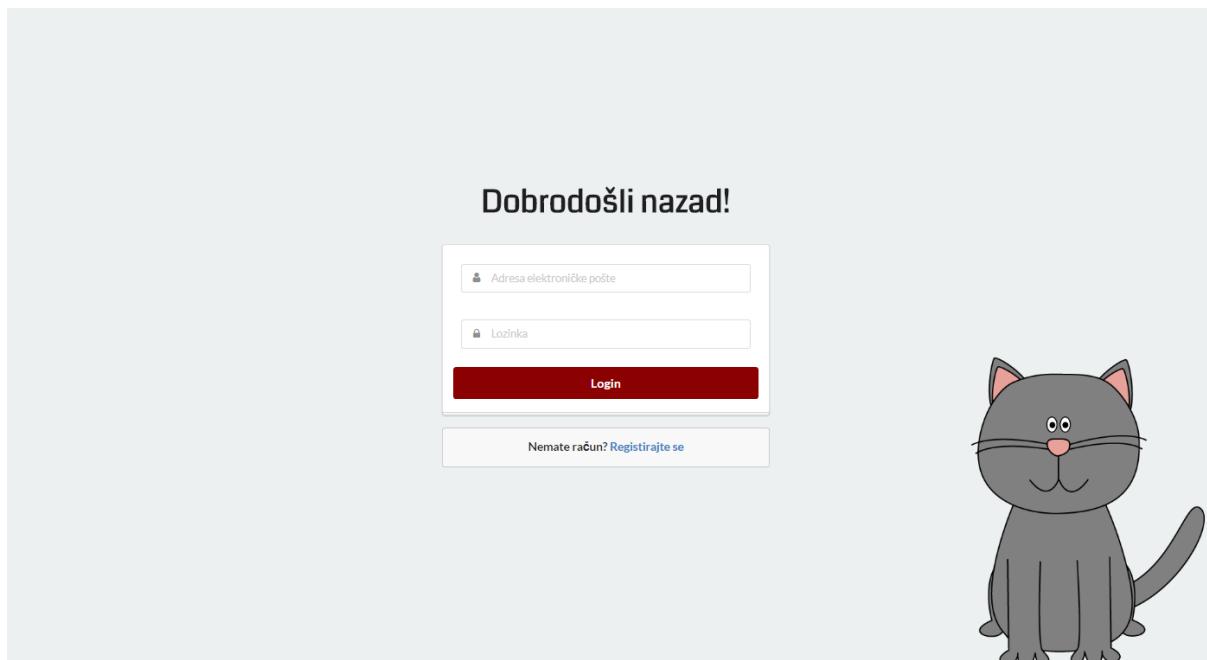
Pritiskom na „Naruči uslugu“ korisnika se vodi do obrasca za naručivanje usluge, ako je u sjednici. U suprotnom se korisniku prikazuje obrazac za prijavu u sustav.

Pritiskom na „Registracija“ korisniku se prikazuje obrazac za registraciju. Za uspješnu registraciju potrebno je ispuniti sva polja obrasca. Nakon uspješnog završetka registracije, korisnika se prijavljuje u sjednicu s novostvorenim računom.



Slika 7.3 - Registracija

Ukoliko je korisnik već registriran, odabirom „Prijava“ može stvoriti sjednicu.

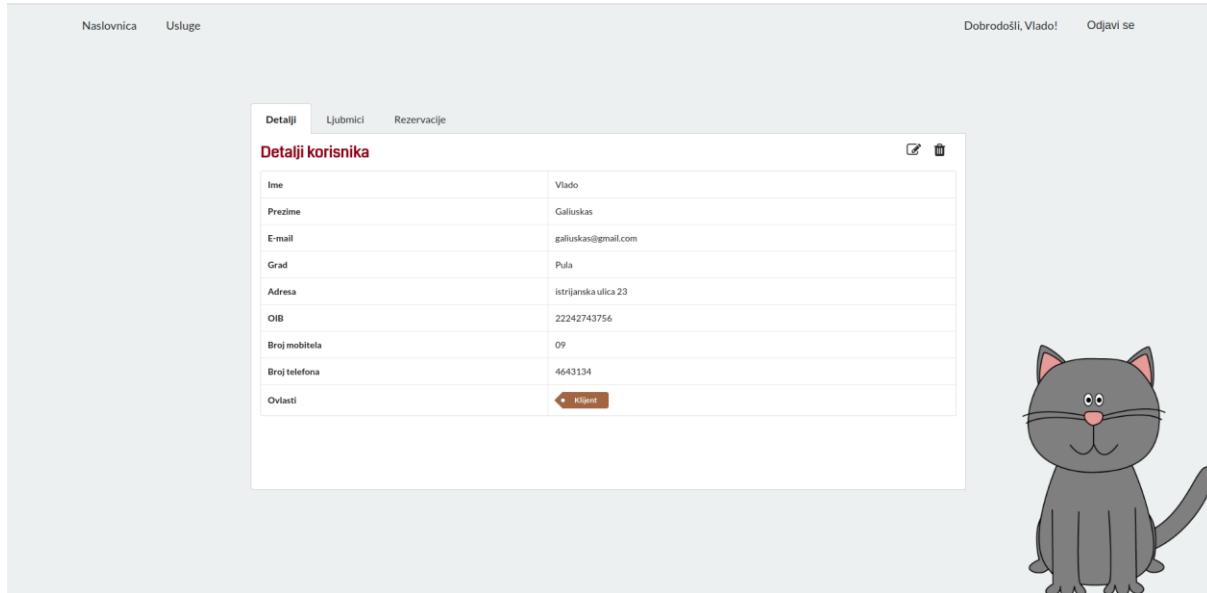


Slika 7.4 - Prijava

Nakon prijave, korisnik pomoću navigacijske trake na vrhu stranice ima mogućnost:

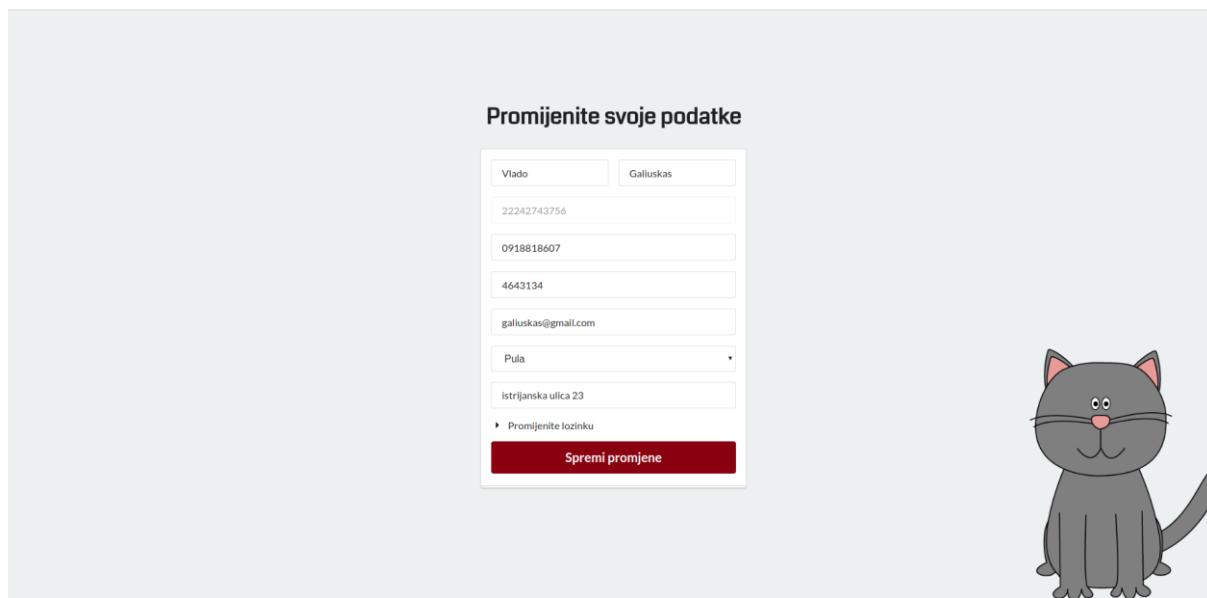
- povrata na naslovnicu
- prikaza profila te
- odjave

Korisniku se prikazuje stranica njegovog profila gdje su prikazani detalji njegovog računa te ima mogućnost izmjene podataka računa te brisanja računa.



Slika 7.5 - Profil korisnika

Odabirom oznake za izmjenu podataka profila prikazuje se obrazac ispunjen podacima korisnika i nudi se mogućnost izmjene telefonskog broja, grada u kojem se nalazi te lozinke. Za uspješnu izmjenu podataka profila, sva polja, osim onih za izmjenu lozinke, moraju biti popunjena i potrebno je unijeti ispravnu lozinku.



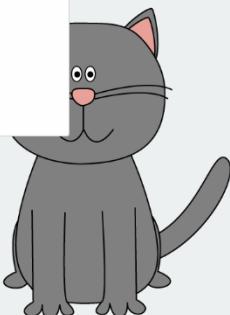
Slika 7.6 – Uređivanje profila

Odabirom oznake za brisanje profila korisniku se prikazuje potvrđni prozor pomoću kojeg može pobrisati svoj račun.

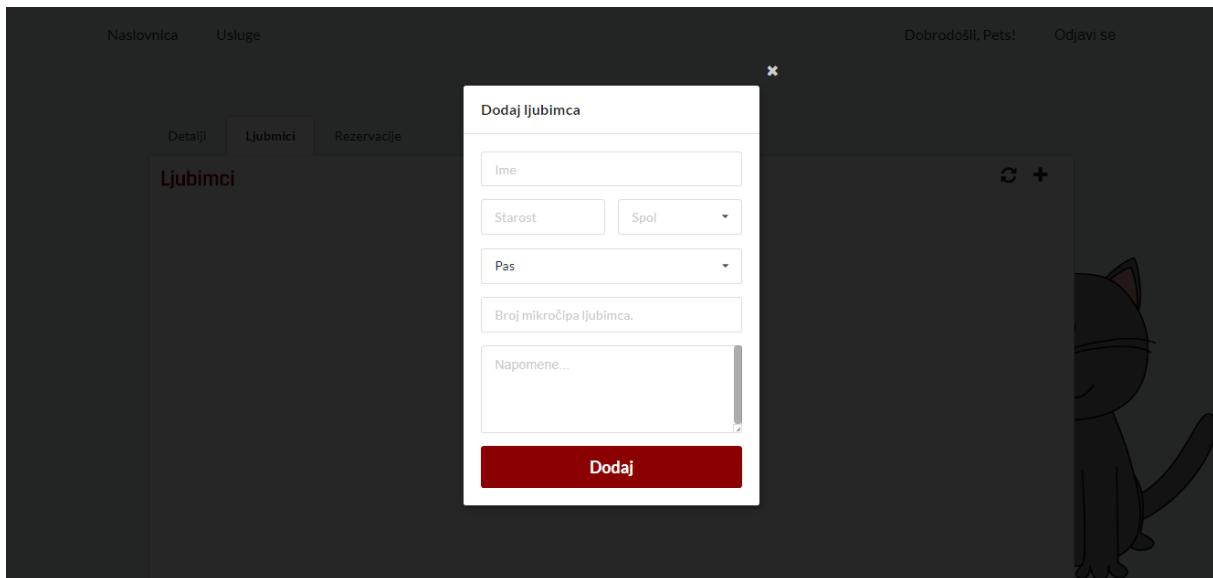
Detalji	Ljubimci	Rezervacije
<b>Detalji korisnika</b>		
Ime	Filip	
Prezime	Sodić	
E-mail	filip.sodic@gmail.com	
Grad	Bregana	<b>Brisanje korisničkog računa</b>
Adresa	sodiceva 32	Jeste li sigurni da želite obrisati korisnički račun?
OIB	69094866377	<input type="button" value="✗ Ne"/> <input checked="" type="button" value="✓ Da"/>
Broj mobitela	0994539394	
Broj telefona	4645542	
Ovlasti	[ACCEPT_RESERVATION, ADD_PET_OTHERS, ADD_RESERVATION_OTHERS, CONFIRM_RESERVATION, ELEVATE_USER_TO_EMPLOYEE, ROLE_ADMINISTRATOR, VIEW_ALL_RESERVATIONS, VIEW_FREE_RESERVATIONS]	

Slika 7.7 – Brisanje korisničkog računa

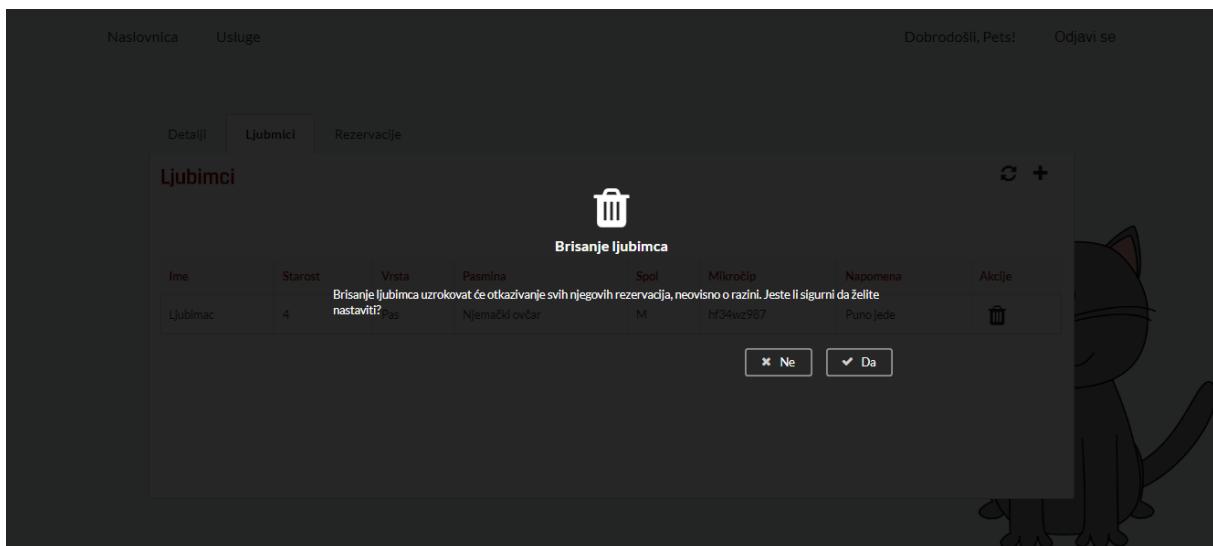
Pritisom na „Ljubimci“ korisniku se prikazuju njegovi ljubimci te ima mogućnost dodavanja novih ljubimaca ili uklanjanja postojećih zapisa o ljubimcima. Za dodavanje novog ljubimca potrebno je unijeti ime, starost, vrstu i spol ljubimca, a za brisanje ljubimca potrebno je potvrditi potvrđni prozor koji se prikaze nakon pritiska na oznaku za brisanje ljubimca.

Naslovnica	Usluge	Dobrodošli, Pets!	Odjavi se																						
<table border="1"> <thead> <tr> <th>Detalji</th> <th>Ljubimci</th> <th>Rezervacije</th> </tr> </thead> <tbody> <tr> <td colspan="3"><b>Ljubimci</b></td> </tr> <tr> <td>Ime</td> <td>Starost</td> <td>Vrsta</td> <td>Pasminka</td> </tr> <tr> <td>Ljubimac</td> <td>4</td> <td>Pas</td> <td>Njemački ovčar</td> </tr> <tr> <td>Spol</td> <td>Mikročip</td> <td>Napomena</td> <td>Akcije</td> </tr> <tr> <td>M</td> <td>hf34wz987</td> <td>Puno jede</td> <td></td> </tr> </tbody> </table> 				Detalji	Ljubimci	Rezervacije	<b>Ljubimci</b>			Ime	Starost	Vrsta	Pasminka	Ljubimac	4	Pas	Njemački ovčar	Spol	Mikročip	Napomena	Akcije	M	hf34wz987	Puno jede	
Detalji	Ljubimci	Rezervacije																							
<b>Ljubimci</b>																									
Ime	Starost	Vrsta	Pasminka																						
Ljubimac	4	Pas	Njemački ovčar																						
Spol	Mikročip	Napomena	Akcije																						
M	hf34wz987	Puno jede																							

Slika 7.8 – Prikaz ljubimaca

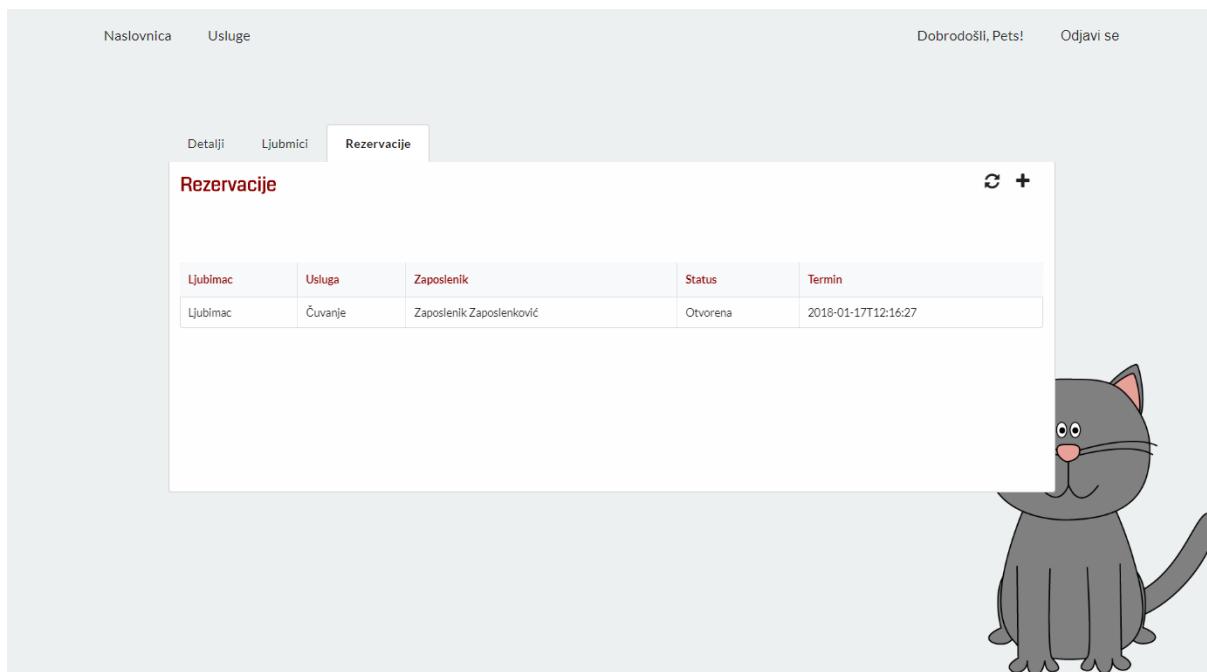


Slika 7.9 – Dodavanje ljubimca

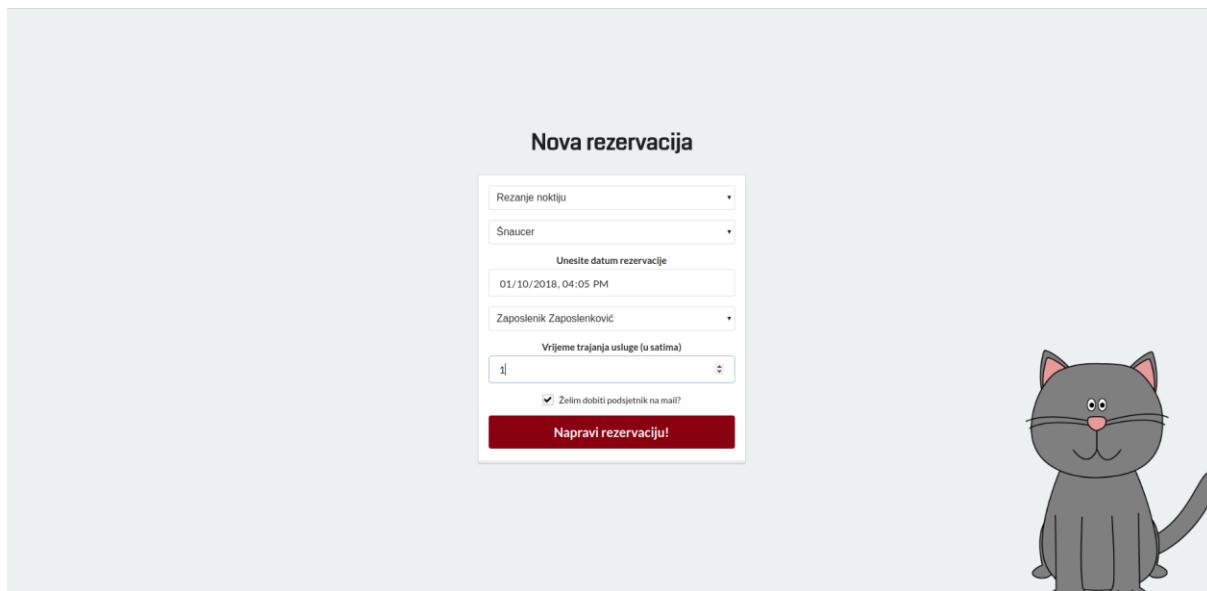


Slika 7.10 – Brisanje ljubimca

Pritiskom na „Rezervacije“ korisniku se prikazuju podaci o svim rezervacijama koje ima te mu se nude mogućnosti stvaranja novih rezervacija. Za uspješno stvaranje rezervacije potrebno je ispuniti sva polja obrasca koji se prikaze, osim polja za odabir zaposlenika. Korisniku se nudi i mogućnost primanje podsjetnika na zakazanu uslugu na e-mail kojim se prijavio. Podsjetnik se šalje najmanje 5 sati prije rezervacije.



Slika 7.11 – Prikaz rezervacija



Slika 7.12 – Stvaranje rezervacije

Nakon rezervacije usluge korisnik čeka da na e-mail dobije ponudu za tu rezervaciju. Tek nakon plaćanja usluge i potvrde administratora sustava, rezervacija je potvrđena.

Na dnu stranice također je moguće odskočiti na neki drugu stranicu aplikacije i prikazani su adresa, e-mail, telefonski broj i poveznica na Facebook stranicu PetsOnlyZg.



Slika 7.13 – Ostale informacije

Korisnik se u svakom trenutku može odjaviti iz aplikacije pritiskom na „Odjavi se“.

### 7.7.2 Pregled zaposlenika

Zaposlenici imaju iste ovlasti kao i korisnici, ali im je u navigacijsku traku dodana mogućnost pregleda rezervacija.

Pritiskom na „Poslovi“ prikazuju se nove rezervacije, prihvaćene rezervacije te potvrđene rezervacije. Zaposlenik može odabratkoje rezervacije može i želi obaviti, i pritiskom na „Prihvati“, poslati e-mail s ponudom o rezervaciji korisniku koji je tu rezervaciju naručio. Potvrdom rezervacije od stvane administratora, rezervacija se prebacuje u potvrđene rezervacije.

Nove rezervacije	Prihvaćene rezervacije	Potvrđene rezervacije
<b>Čuvanje, Ljubimac</b> Vrijeme: 2018-01-17T12:16:27 <a href="#">Detalji rezervacije</a> <b>Prihvati</b>	<b>Setnja, Snaucer</b> Vrijeme: 2017-01-06T23:15 <a href="#">Detalji rezervacije</a>	<b>Setnja, Lera</b> Vrijeme: 2018-01-12T22:52:13 <a href="#">Detalji rezervacije</a> <b>Završi</b>
<b>Rezanje noktiju, Spaner</b> Vrijeme: 2017-01-04T12:32:47 <a href="#">Detalji rezervacije</a> <b>Prihvati</b>		<b>Setnja, rex</b> Vrijeme: 2016-12-30T07:59:32 <a href="#">Detalji rezervacije</a> <b>Završi</b>
<b>Kupanje, setlo</b> Vrijeme: 2018-01-17T11:39:33 <a href="#">Detalji rezervacije</a> <b>Prihvati</b>		<b>Čuvanje, Lera</b> Vrijeme: 2018-01-12T23:32:20 <a href="#">Detalji rezervacije</a> <b>Završi</b>
<b>Odlazak veterinaru, Snaucer</b> Vrijeme: 2017-01-01T00:00 <a href="#">Detalji rezervacije</a> <b>Prihvati</b>		<b>Čuvanje, Spaner</b> Vrijeme: 2016-12-30T23:15 <a href="#">Detalji rezervacije</a> <b>Završi</b>
<b>Čuvanje, Ljubimac</b> Vrijeme: 2018-01-17T11:38:16 <a href="#">Detalji rezervacije</a> <b>Prihvati</b>		<b>Rezanje noktiju, Lera</b> Vrijeme: 2018-01-12T15:36:58 <a href="#">Detalji rezervacije</a> <b>Završi</b>

Slika 7.14 – Pregled rezervacija kao zaposlenik

### 7.7.3 Pregled administratora

Administrator posjeduje sve ovlasti kao i zaposlenik, uz što mu je omogućeno još nekoliko radnji.

Pritiskom na „Poslovi“ prikazuje mu se prikaz poslova kao i zaposleniku, no on ovdje ima i mogućnost potvrditi rezervacije koje su zaposlenici prihvatali.

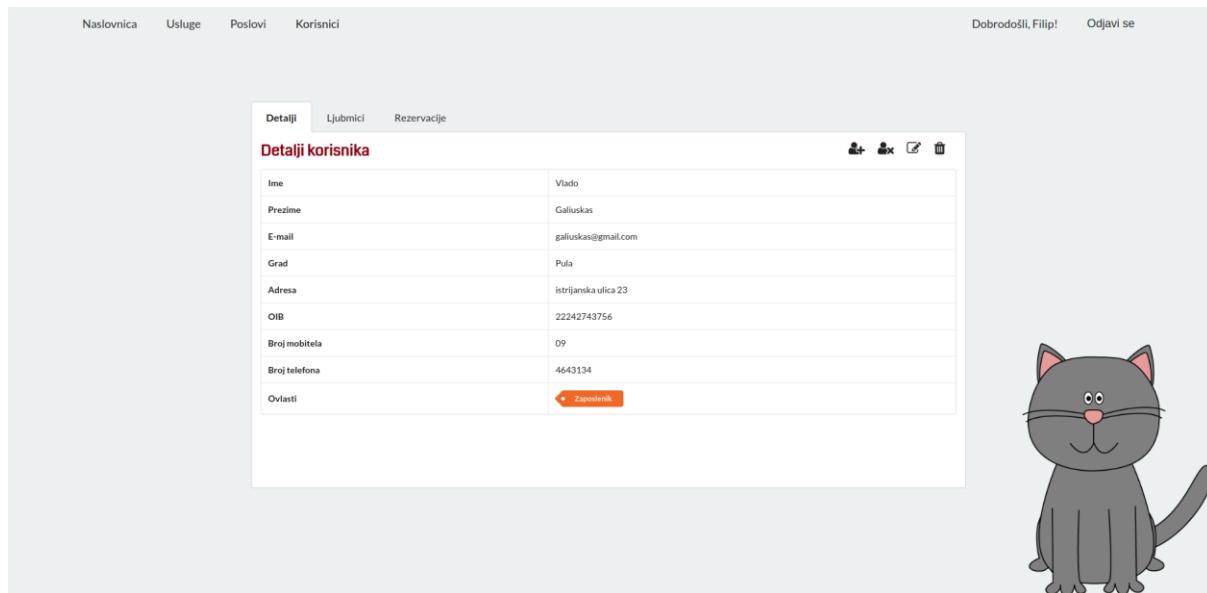
Nove rezervacije	Prihvaćene rezervacije	Potvrđene rezervacije
<b>Čuvanje, Ljubimac</b> Vrijeme: 2018-01-17T12:16:27 ► Detalji rezervacije  <b>Prihvati</b>	<b>Setnja, Šnaucer</b> Vrijeme: 2017-01-06T23:15 ► Detalji rezervacije  <b>Potvrdi</b>	<b>Setnja, Lera</b> Vrijeme: 2018-01-12T22:52:13 ► Detalji rezervacije  <b>Završi</b>
<b>Rezanje noktiju, Spaner</b> Vrijeme: 2017-01-04T12:32:47 ► Detalji rezervacije  <b>Prihvati</b>	<b>Setnja, rex</b> Vrijeme: 2016-12-30T07:59:32 ► Detalji rezervacije  <b>Završi</b>	<b>Čuvanje, Lera</b> Vrijeme: 2018-01-12T23:32:20 ► Detalji rezervacije  <b>Završi</b>
<b>Kupanje, setko</b> Vrijeme: 2018-01-17T11:39:33 ► Detalji rezervacije  <b>Prihvati</b>	<b>Čuvanje, Spaner</b> Vrijeme: 2016-12-30T23:15 ► Detalji rezervacije  <b>Završi</b>	<b>Čuvanje, Ljubimac</b> Vrijeme: 2018-01-17T11:38:16 ► Detalji rezervacije  <b>Prihvati</b>
<b>Odlazak veterinaru, Šnaucer</b> Vrijeme: 2017-01-01T00:00 ► Detalji rezervacije  <b>Prihvati</b>	<b>Rezanje noktiju, Lera</b> Vrijeme: 2018-01-12T15:36:58 ► Detalji rezervacije  <b>Završi</b>	
<b>Čuvanje, Ljubimac</b> Vrijeme: 2018-01-17T11:38:16 ► Detalji rezervacije  <b>Prihvati</b>		

Slika 7.15 – Pregled rezervacija kao administrator

U navigacijskoj traci administratoru je omogućen i pregled korisnika pritiskom na „Korisnici“. Na ovoj stranici prikazuju se osnovni podaci svih korisnika i njihova uloga u aplikaciji te se, pritiskom na „Detalji“, mogu vidjeti detalji profila odabranog korisnika.

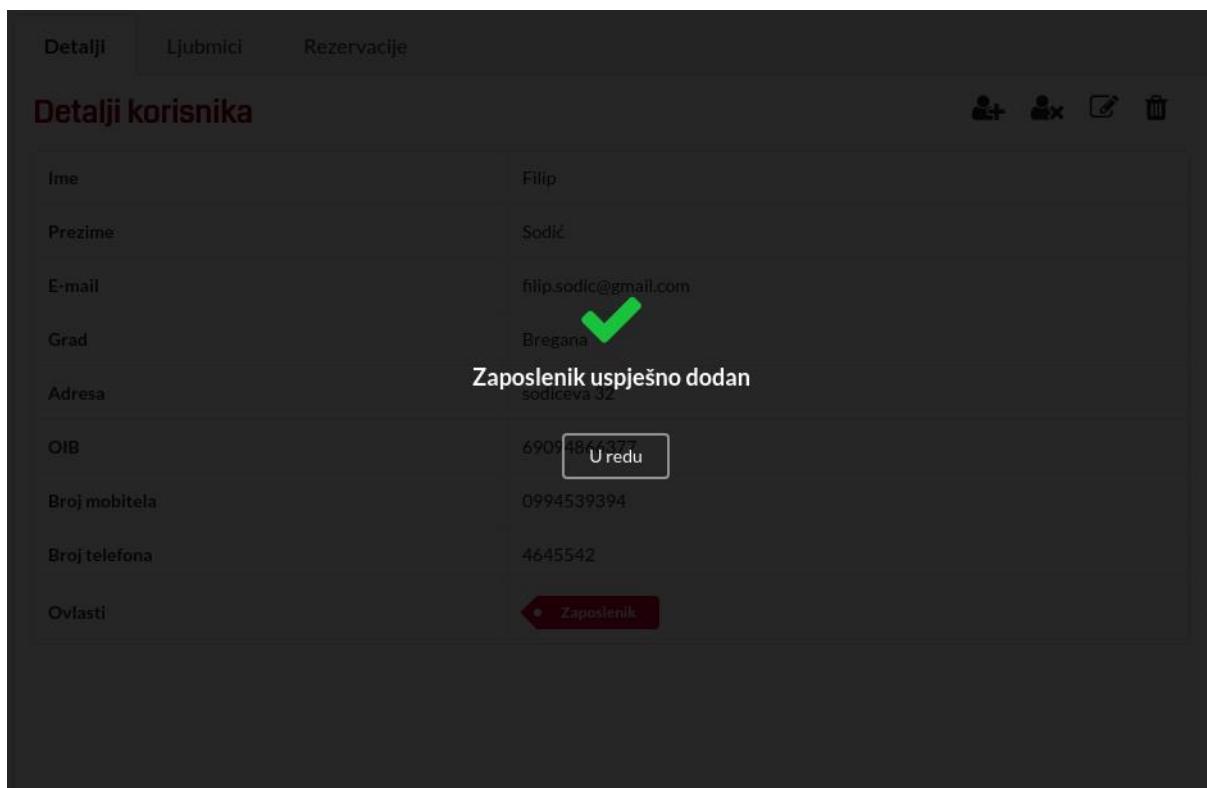
Ime	Prezime	E-mail adresa	Uloga
Vlado	Galiuskas	galiuskas@gmail.com	<b>Klijent</b> Detalji
Tomica	Kravričan	tomica.kravrsan@gmail.com	<b>Klijent</b> Detalji
Mate	Paulinović	mate.paulinovic@gmail.com	<b>Klijent</b> Detalji
Maja	Krmpotić-Durđević	maja.krmpi@gmail.com	<b>Klijent</b> Detalji
Zaposlenik	Zaposlenković	zaposlenik@gmail.com	<b>Zaposlenik</b> Detalji
Filip	Sodić	filip.sodic@gmail.com	<b>Administrator</b> Detalji

Slika 7.16 – Korisnici aplikacije

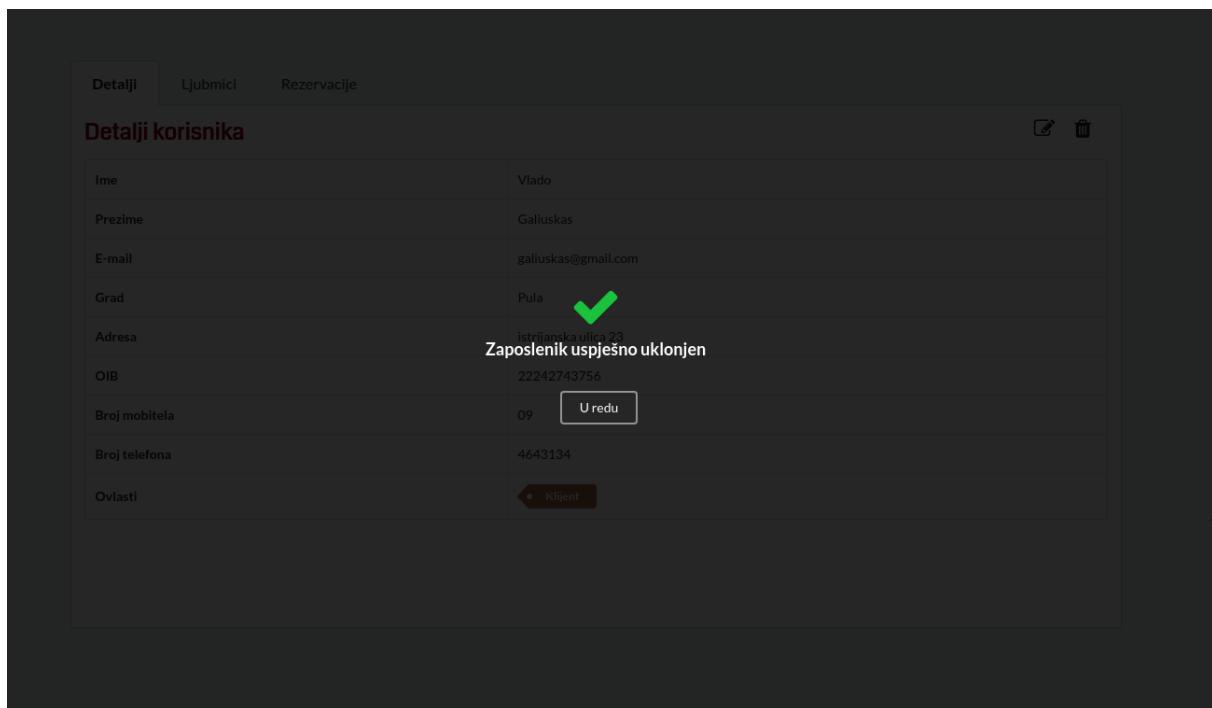


Slika 7.17 – Prikaz detalja o odabranom korisniku

Na ovoj stranici administrator može odabranog korisnika zaposliti ili zaposlenika otpustiti.



Slika 7.18 – Zapošljavanje novog korisnika



Slika 7.19 – Otpuštanje zaposlenika

## 8. Zaključak i budući rad

U sklopu izrađivanja prve verzije dokumentacije izradili smo opis projektnog zadatka, a sukladno njemu smo napravili i opise obrazaca upotrebe, dijagrame obrazaca upotrebe te sekvencijske dijagrame čime smo zaključili glavne funkcijeske zahtjeve. U sklopu dizajna sustava i arhitekture odredili smo način oblikovanja web-poslužitelja, Model-View-Controller (MVC) model te smo njegov rad predočili dijagramom. Također, napravili smo i dijagrame razreda, koje smo radi preglednosti odvojili u više dijagrama te model baze podataka.

U drugoj verziji smo implementirali dosada definirane modele, instancirali bazu podataka s pripadnom shemom, izradili korisničko sučelje te obavili testiranja nad sustavom kako bi bili sigurni da ne postoje nepredviđene greške. Dokumentacija je upotpunjena novim UML dijigramima kao što su dijagrami aktivnosti, stanja, komponenti, komunikacijski, raspodjele itd.

Jako dobro smo organizirani i rad je bio dobro raspodijeljen. Za praćenje verzija koristimo GitLab, a za komunikaciju Slack.

U budućnosti planiramo napraviti neka manja poboljšanja na korisničkom sučelju kako bi ga učinili ugodnijim za korištenje.

## 9. Popis literature

- [1] Oblikovanje programske potpore, FER ZEMRIS, <http://www.fer.hr/predmet/opp>
- [2] Interna skripta "Procesi programskog inžinjerstva",  
[http://www.fer.unizg.hr/predmet/opp/nastavni\\_plan/literatura](http://www.fer.unizg.hr/predmet/opp/nastavni_plan/literatura)
- [3] Spring MVC Tutorial for Beginners,  
<https://www.youtube.com/watch?v=BjNhGaZDr0Y>
- [4] Apache Tomcat 7, <http://tomcat.apache.org/tomcat-7.0-doc>
- [5] Spring Guides, <https://spring.io/guides>
- [6] jQuery AJAX Introduction ,  
[https://www.w3schools.com/jquery/jquery\\_ajax\\_intro.asp](https://www.w3schools.com/jquery/jquery_ajax_intro.asp)
- [7] Spring Security , <https://projects.spring.io/spring-security/>
- [8] [MySQL 5.7 Reference Manual/Tutorial](#),  
<https://dev.mysql.com/doc/refman/5.7/en/tutorial.html>

## 10. Dodatak A: Indeks

- Slika 4.1 - Cjeloviti pregled svih obrazaca upotrebe
- Slika 4.2 - Obrasci vezani za interakciju sa korisnicima
- Slika 4.3 - Obrasci vezani za rezervacije
- Slika 4.4 - Sekvencijski dijagram za UC1 (Registracija)
- Slika 4.5 - Sekvencijski dijagram za UC2 (DodavanjeZaposlenika)
- Slika 4.6 - Sekvencijski dijagram za UC3 (Prijava)
- Slika 4.7 - Sekvencijski dijagram za UC4 (PregledIUređivanjePodataka)
- Slika 4.8 - Sekvencijski dijagram za UC5 (UklanjanjeZaposlenika)
- Slika 4.9 - Sekvencijski dijagram za UC6 (BrisanjeKorisnika)
- Slika 4.10 - Sekvencijski dijagram za UC7 (DodavanjeLjubimca)
- Slika 4.11 - Sekvencijski dijagram za UC8 (BrisanjeLjubimca)
- Slika 4.12 - Sekvencijski dijagram za UC9 (StvaranjeRezervacije)
- Slika 4.13 - Sekvencijski dijagram za UC10 (UređivanjeRezervacije)
- Slika 4.14 - Sekvencijski dijagram za UC11 (PraćenjeRezervacije)
- Slika 4.15 - Sekvencijski dijagram za UC12 (PrihvatanjeRezervacije)
- Slika 4.16 - Sekvencijski dijagram za UC13 (PotvrdaPlaćeneRegistracije)
- Slika 4.17 - Sekvencijski dijagram za UC14 (OtkazivanjeRezervacije)
- Slika 4.18 - Sekvencijski dijagram za UC15 (SlanjePodsjetnika)
- Slika 4.19 - Sekvencijski dijagram za UC16 (PregledRezervacija)
- Slika 4.20 - Sekvencijski dijagram za UC17 (Odjava)
- Slika 6.1 - način rada obrasca MVC
- Slika 6.2 - ER model baze podataka
- Slika 6.3 - Detaljan UML dijagram razreda modela
- Slika 6.4 - Detaljan UML dijagram razreda upravitelja
- Slika 6.5 - Detaljan UML dijagram repozitorija
- Slika 6.6 - Detaljan UML dijagram pomoćnih razreda
- Slika 6.7 - Detaljan UML dijagram svih razreda
- Slika 6.8 - UML dijagram odnosa između svih razreda
- Slika 6.9 - UML dijagram objekata
- Slika 7.1. - Dijagram razmještaja
- Slika 7.2 - Naslovna stranica
- Slika 7.3 - Registracija
- Slika 7.4 - Prijava
- Slika 7.5 - Profil korisnika
- Slika 7.6 - Uređivanje profila
- Slika 7.7 - Brisanje korisničkog računa
- Slika 7.8 - Prikaz ljubimaca
- Slika 7.9 - Dodavanje ljubimca
- Slika 7.10 - Brisanje ljubimca
- Slika 7.11 - Prikaz rezervacija

- Slika 7.12 - Stvaranje rezervacije
- Slika 7.13 - Ostale informacije
- Slika 7.14 - Pregled rezervacija kao zaposlenik
- Slika 7.15 - Pregled rezervacija kao administrator
- Slika 7.16 - Korisnici aplikacije
- Slika 7.17 - Prikaz detalja o odabranom korisniku
- Slika 7.18 - Zapošljavanje novog korisnika
- Slika 7.19 - Otpuštanje zaposlenika

## 11. Dodatak B: Dnevnik sastanaka

Datum	Sudionici	Sadržaj sastanka
7.10.2017.	Sodić Paulinović Modrušan Krmpotić-Đurđević Galić	Definiranje projektnog zadatka
13.10.2017.	Sodić Paulinović Modrušan Krmpotić-Đurđević Kravršćan	Osnovne funkcionalnosti aplikacije i obrasci uporabe
22.10.2017.	Sodić Modrušan	Osnovne funkcionalnosti HTML-a, CSS-a i Javascripta
23.10.2017	Sodić Kravršćan Galić	Izrada funkcionalnih zahtjeva (obrasci uporabe definirani)
25.10.2017	Sodić Krmpotić-Đurđević Paulinović	Izrada sekvencijskih dijagrama i ER modela baze podataka
26.10.2017	Sodić	Konzultacije s profesorom

Datum	Sudionici	Sadržaj sastanka
29.10.2017	Modrušan Sodić Krmpotić-Đurđević Kravršćan Galić Paulinović	Dogovor oko završetka prve verzije dokumentacije
3.11.2017	Sodić	Konzultacije s profesorom
17.11.2017	Sodić Kravršćan Galić Paulinović	Konzultacije s profesorom i provjera finalne verzije dokumentacije
17.11.2017	Sodić Kravršćan Galić Paulinović	Završavanje i dorada prve verzije dokumentacije
20.11.2017	Sodić Kravršćan Galić Paulinović Krmpotić-Đurđević Modrušan	Dogovor oko tijeka projekta u drugom ciklusu
1.12. 2017	Sodić	Okvirni plan podjele posla

Datum	Sudionici	Sadržaj sastanka
	Kravršćan Galić Paulinović Krmpotić-Đurđević Modrušan	
9.12.2017	Sodić Galić	Dogovor u vezi <i>front-end</i> razvoja
9.12.2017	Modrušan Kravršćan	Dogovor u vezi logike razreda upravitelja
10.12.2017	Paulinović Krmpotić-Đurđević	Dogovor u vezi baze podataka
15.12.2017.	Sodić Galić Modrušan Kravršćan Paulinović Krmpotić-Đurđević	Međusobno informiranje u vezi postiglih dogovora
2.1.2018	Sodić Modrušan Paulinović Kravršćan	Pregled dosadašnjeg rada

Datum	Sudionici	Sadržaj sastanka
4.1.2018.	Sodić Galić Kravršćan Modrušan Krmpotić-Đurđević	Zajedničko razrješavanje problema
6.1.2018.	Sodić Galić Kravršćan Modrušan Paulinović Krmpotić-Đurđević	Zajednički rad
8.1.2018.	Sodić Galić Kravršćan Modrušan Paulinović Krmpotić-Đurđević	Konzultacije s profesorom i predaja alfa verzije
8.1.2018.	Sodić Galić Kravršćan Modrušan Paulinović Krmpotić-Đurđević	Dogovor za konačnu verziju projekta

Datum	Sudionici	Sadržaj sastanka
15.1.2018.	Sodić Galić Kravrščan Modrušan Paulinović Krmpotić-Đurđević	Zajednički rad u vezi dokumentacije
17.1.2018.	Sodić Galić Kravrščan Modrušan Paulinović Krmpotić-Đurđević	Konačni dogovori prije predaje projekta

## 12. Dodatak C: Prikaz aktivnosti grupe

### 12.1 Tablični prikaz sudjelovanja

Popis aktivnosti	Antun Modrušan	Filip Sodić	Maja Krmpotić-Đurđević	Mate Paulinović	Tomislav Kravarščan	Vlado Galić
Dnevnik promjena dokumentacije	0%	0%	0%	0%	0%	100%
Opis projektnog zadatka	15%	45%	10%	8%	7%	10%
Rječnik pojmova	80%	3%	0%	0%	0%	17%
Opis funkcionalnih zahtjeva	0%	15%	0%	40%	35%	10%
Opis ostalih zahtjeva	0%	0%	0%	100%	0%	0%
<b>Arhitektura i dizajn sustava</b>						
Svrha, opći prioriteti i skica sustava	0%	10%	90%	0%	0%	0%
Opis dijagrama razreda	0%	50%	0%	50%	0%	0%
<b>Implementacija i korisničko sučelje</b>						
Opis dijagrama razmještaja	0%	0%	0%	0%	0%	100%
Korištene tehnologije i alati	0%	0%	0%	100%	0%	0%
Isječak programskog koda	0%	0%	0%	0%	100%	0%
Ispitivanje	0%	0%	100%	0%	0%	0%

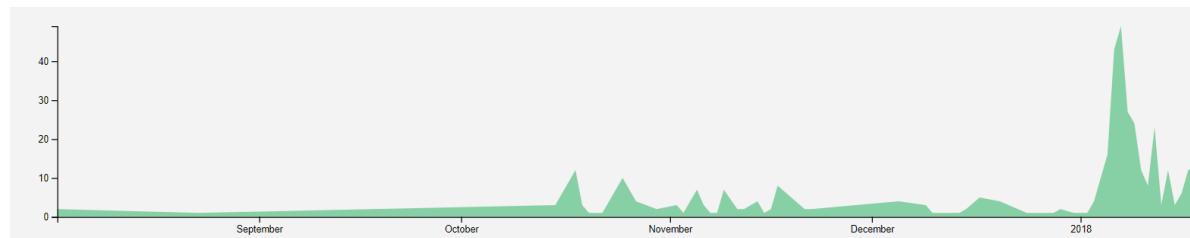
Popis aktivnosti	Antun Modrušan	Filip Sodić	Maja Krmpotić-Đurđević	Mate Paulinović	Tomislav Kravarščan	Vlado Galić
programskog rješenja						
Upute za instalaciju	0%	100%	0%	0%	0%	0%
Korisničke upute	90%	7%	0%	3%	0%	0%
Zaključak i budući rad	0%	0%	0%	0%	100%	0%
Popis literature	100%	0%	0%	0%	0%	0%
<b>Dodaci</b>						
Indeks	0%	0%	80%	0%	0%	20%
Dnevnik sastajanja	0%	0%	0%	100%	0%	0%
Prikaz aktivnosti grupe	0%	30%	0%	0%	70%	0%
Plan rada	0%	0%	0%	0%	0%	100%
Pregled rada i stanje ostvarenja	0%	0%	0%	0%	0%	100%
<b>Izrada dijagrama</b>						
Dijagrami obrazaca uporabe	5%	10%	5%	5%	70%	5%
Sekvencijski dijagrami	7%	5%	3%	70%	5%	10%
Nacrt baze podataka	4%	3%	80%	3%	5%	5%
Dijagrami razreda	5%	20%	5%	30%	30%	5%

Popis aktivnosti	Antun Modrušan	Filip Sodić	Maja Krmpotić-Đurđević	Mate Paulinović	Tomislav Kravarščan	Vlado Galić
Dijagram objekata	0%	100%	0%	0%	0%	0%
Komunikacijski dijagram	100%	0%	0%	0%	0%	0%
Dijagram stanja	0%	0%	0%	0%	0%	100%
Dijagram aktivnosti	5%	0%	90%	0%	5%	0%
Dijagram komponenti	100%	0%	0%	0%	0%	0%
Dijagram razmještaja	0%	0%	0%	0%	0%	100%
<b>Implementacija</b>						
Prezentacijski sloj i dizajn	5%	40%	5%	5%	5%	40%
Modeli prezentacijskog sloja	10%	5%	0%	10%	70%	5%
Poslovna logika	30%	0%	0%	10%	50%	10%
API i prihvaćanje HTTP zahtjeva	45%	0%	0%	5%	50%	0%
Komunikacija s bazom podataka	0%	0%	0%	90%	10%	0%
Modeli baze podataka	0%	0%	10%	70%	20%	0%
Izrada baze podataka	0%	0%	40%	60%	0%	0%
Razmještaj ( <i>deployment</i> )	0%	50%	0%	0%	50%	0%

## 12.2 Grafički prikaz aktivnosti grupe

August 2, 2017 – January 18, 2018

Commits to development, excluding merge commits. Limited to 6,000 commits.



## 13. Dodatak D: Plan rada i ostvarenje

### 13.1 Ostvareno u rev. 1:

*Dokumentacija:*

- Opis projektnog zadatka
- Funkcionalni zahtjevi
- Ostali zahtjevi
- Dnevnik sastanaka
- Dnevnik promjene dokumentacije
- Arhitektura i dizajn sustava

*Implementacija:*

- Inicijalno stvaranje *Maven* projekta
- Povezivanje okoline s *Gitlab*-om

### 13.2 Ostvareno u rev. 2:

*Dokumentacija:*

- Implementacija i korisničko sučelje
- Zaključak i budući rad
- Crtanje ostalih UML dijagrami
- Dodani bitni isječci koda
- Popis literature
- Pregled i konačna dorada cijele dokumentacije

*Implementacija:*

- Svih glavnih i sporednih dijelova sustava
- Izgled aplikacije
- Konačno ispitivanje funkcionalnosti i ispravnosti sustava

### 13.3 Plan za daljnji razvoj:

*Implementacija:*

Kako za ovaj projekt postoji stvarni klijent odnosno naručitelj web aplikacije tako je potrebno implementirati dodatne zahtjeve klijenta koji izlaze iz domene zahtjeva predmeta „Oblikovanje programske potpore“. Potrebno je još dodatno uljepšati izgled aplikacije i prilagoditi pojedine detalje stvarnoj uporabi.