Oblikovanje programske potpore

Ak. god. 2014./2015.

*Parking*

Dokumentacija, Rev. *1*

Grupa: *Epsilon*

Voditelj: *Ivan Grgurina*

Datum predaje: *16.11.2014.*

Nastavnik: *Nikolina Frid*

Popis članova grupe i zaduženja

Ivan Grgurina – voditelj projekta, dizajn aplikacije, izrada aplikacije – backend, općenita podjela poslova, organizacija sastanaka, izrada baze podataka, oblikovanje programske potpore, arhitektura i dizajn sustava, kontrola dokumentacije, korištene tehnologije i alati, ispitivanje programskog rješenja

Zoran Krišto – opis projektnog zadatka, rječnik pojmova, funkcionalni i nefunkcionalni zahtjevi, opis obrazaca uporabe, sekvencijski dijagrami, opis sekvencijskih dijagrama, dizajn UML dijagrama, komunikacijski dijagrami, dijagrami komponenti, dijagram razmještaja, ispitivanje programskog rješenja, zaključak

Antun Maldini – ispravci dokumentacije, izrada aplikacije – backend, kontrola, spajanje i formatiranje dokumentacije, opis obrazaca uporabe, funkcionalni i nefunkcionalni zahtjevi, sekvencijski dijagrami, dizajn UML dijagrama, komunikacijski dijagrami, ispitivanje programskog rješenja

Katarina Matić – ispravci dokumentacije, opis obrazaca uporabe, funkcionalni i nefunkcionalni zahtjevi, dijagrami obrazaca uporabe, sekvencijski dijagrami, opis sekvencijskih dijagrama, dizajn UML dijagrama, komunikacijski dijagrami, dijagrami komponenti, dijagram razmještaja, ispitivanje programskog rješenja, skica sustava

Ante Tomić – dizajn aplikacije, izrada aplikacije – frontend, opis projektnog zadatka, rječnik pojmova, ostali zahtjevi, sekvencijski dijagrami, opis sekvencijskih dijagrama, lektoriranje i formatiranje dokumentacije, dnevnik sastajanja, upute, korištene tehnologije i alati, ispitivanje programskog rješenja, zaključak

Josip Vukušić – izrada aplikacije – backend, opis obrazaca uporabe, funkcionalni i nefunkcionalni zahtjevi, dizajn dijagrama obrazaca uporabe, sekvencijski dijagrami, dnevnik sastajanja, dizajn UML dijagrama, dijagrami stanja, dijagrami aktivnosti, ispitivanje programskog rješenja, opis dijagrama razreda

Ivan Žagar – izrada aplikacije – backend, podjela poslova kod izrade UML dijagrama i obrazaca uporabe, opis obrazaca uporabe, funkcionalni i nefunkcionalni zahtjevi, sekvencijski dijagrami, spajanje dokumentacije, dijagrami stanja, dijagrami aktivnosti, ispitivanje programskog rješenja, opis dijagrama razreda

Sadržaj

[1. Dnevnik promjena dokumentacije 5](#_Toc408928068)

[2. Opis projektnog zadatka 6](#_Toc408928069)

[3. Rječnik pojmova 9](#_Toc408928070)

[4. Funkcionalni zahtjevi 11](#_Toc408928071)

[4.1. Opis obrazaca uporabe 13](#_Toc408928072)

[4.2. Sekvencijski dijagrami 27](#_Toc408928073)

[5. Ostali zahtjevi 46](#_Toc408928074)

[6. Arhitektura i dizajn sustava 47](#_Toc408928075)

[6.1. Svrha, opći prioriteti i skica sustava 47](#_Toc408928076)

[6.1.1. Web-aplikacija 47](#_Toc408928077)

[6.1.2. Baza podataka 49](#_Toc408928078)

[6.2. Dijagram razreda s opisom 52](#_Toc408928079)

[6.2.1. Models 52](#_Toc408928080)

[6.2.2. Controllers 56](#_Toc408928081)

[6.2.3. Helpers 58](#_Toc408928082)

[6.3. Dijagram objekata 59](#_Toc408928083)

[6.4. Ostali UML dijagrami 61](#_Toc408928084)

[6.4.1. Komunikacijski dijagrami 61](#_Toc408928085)

[6.4.2. Dijagrami stanja 63](#_Toc408928086)

[6.4.3. Dijagrami aktivnosti 65](#_Toc408928087)

[6.4.4. Dijagram komponenata 67](#_Toc408928088)

[7. Implementacija i korisničko sučelje 68](#_Toc408928089)

[7.1. Dijagram razmještaja 68](#_Toc408928090)

[7.2. Korištene tehnologije i alati 69](#_Toc408928091)

[7.3. Isječak programskog koda vezan za temeljnu funkcionalnost sustava 70](#_Toc408928092)

[7.4. Ispitivanje programskog rješenja 71](#_Toc408928093)

[7.5. Upute za instalaciju 72](#_Toc408928094)

[7.6. Korisničke upute 78](#_Toc408928095)

[7.6.1. Neregistrirani korisnici (posjetitelji) 79](#_Toc408928096)

[7.6.2. Registrirani korisnici (klijenti) 80](#_Toc408928097)

[7.6.3. Administratori 81](#_Toc408928098)

[8. Zaključak 82](#_Toc408928099)

[9. Reference 83](#_Toc408928100)

[Dodatak A: Dnevnik sastajanja 84](#_Toc408928101)

1. Dnevnik promjena dokumentacije

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Opis promjene/dodatka | Autor(i) | Datum |
| 0.1 | Napravljen predložak. | Matić | xx.10.2014. |
| 0.2 | Dodani funkcionalni i nefunkcionalni zahtjevi. | Žagar | 20.10.2014. |
| 0.21 | Promjena dijela funkcionalnih zahtjeva | Grgurina | 27.10.2014. |
| 0.3 | Dodani *Use Case* dijagrami | Grgurina | 27.10.2014. |
| 0.4 | Dodani opis projektnog zadatka, rječnik pojmova i dnevnik sastajanja za prva tri sastanka | Tomić, Krišto | 29.10.2014. |
| 0.41 | Dodani dnevnik sastajanja za četvrti sastanak i ostali zahtjevi, ispravke unutar dokumentacije | Tomić, Vukušić | 09.11.2014. |
| 0.5 | Dodani dijagrami obrazaca uporabe | Krišto, Žagar | 12.11.2014. |
| 0.51 | Dodani sekvencijski dijagrami | Matić, Vukušić | 13.11.2014. |
| 0.55 | Manje izmjene dokumentacije | Matić, Žagar | 14.11.2014. |
| 0.6 | Formatiranje i lektoriranje dijela dokumentacije | Tomić | 14.11.2014. |
| 0.7 | Arhitektura i dizajn sustava | Grgurina | 15.11.2014. |
| 0.75 | Završno formatiranje i lektoriranje dokumentacije | Tomić | 15.11.2014. |
| 1.0 | Prva verzija dokumentacije | Grgurina | 16.11.2014. |
| 1.1 | Unesene sve ispravke dokumentacije | Tomić, Krišto, Matić | 30.12.2014. |
| 1.2 | Dodani dijagrami stanja, aktivnosti, razmještaja i komunikacijski dijagrami | Krišto, Maldini, Matić, Tomić, Vukušić, Žagar | 12.01.2015. |
| 1.3 | Dodani dio uputa i zaključak | Tomić | 14.01.2015. |
| 1.4 | Dodan opis dijagrama razreda | Vukušić, Žagar | 14.01.2015. |
|  |  |  |  |

1. Opis projektnog zadatka

Cilj je ovog projekta razviti informacijski sustav za tvrtku Gradski Parking d.o.o. koji će služiti za neprekidno praćenje slobodnih parkirališnih mjesta na lokacijama diljem grada Zagreba u vlasništvu ove tvrtke. Nakon što je tvrtka izrazila želju za unaprjeđenjem njihove usluge, odlučeno je da se izgradi web-stranica s javnim sučeljem prema postavljenim zahtjevima, koja će osim prikazivanja prikupljenih informacija o slobodnim parkirališnim mjestima omogućiti i niz drugih funkcionalnosti spomenutih u nastavku koje se mogu pružiti korisnicima.

Objekti u vlasništvu tvrtke, koji sadrže parkirališna mjesta, dijele se u dvije skupine: garaže i otvorena parkirališta. Svaki je objekt jednoznačno određen svojom pozicijom koja se sastoji od adrese te geografske dužine i širine. Oni su pomoću tih podataka pridruženi karti koja je dostupna na početnoj stranici. Također se svaki objekt sastoji od parkirališnih mjesta te svi objekti inicijalno moraju imati više od deset parkirališnih mjesta. Kontrola parkirališnih mjesta na bilo kojem objektu, koja mogu zauzeti samo automobili ili motocikli, vrši se putem senzora koji radi na principu očitanja promjene stanja te ovisno o tome je li odabrano parkirališno mjesto slobodno ili zauzeto šalje odgovarajući podatak u središnji sustav koji taj podatak sprema. Poslani se podatci prikazuju na izrađenoj web-stranici u obliku najprikladnijem za korisnika te služe kao povratna informacija korisniku o dostupnosti pojedinog parkirališta. Osim što bi ovako implementiran sustav trebao biti pregledan i jednostavan za korištenje svim korisnicima kojima će trebati dostupne usluge, on bi također trebao omogućiti registraciju klijenta čime on stječe dodatne pogodnosti poput raznih popusta i mogućnosti rezervacije parkirališnog mjesta.

Predviđene su sljedeće vrste korisnika koji su ujedno i dionici sustava prema čijim se potrebama i načinima korištenja oblikuje sustav:

1. neregistrirani korisnici (posjetitelji)
2. registrirani korisnici (klijenti)
3. administratori

Neregistrirani korisnici mogu pregledavati sadržaj web-stranice, kartu s lokacijama svih parkirališta i garaža te imaju mogućnost registracije ako su punoljetni i pritom daju tražene osobne podatke. Sam je sustav zamišljem kao web-aplikacija s javnim sučeljem te je karta glavni sadržaj web-stranice koji prikazuje sve garaže i parkirališta s pripadajućim lokacijama. Klikom na pojedino parkiralište korisniku postaju dostupne informacije o trenutnom stanju poput broja slobodnih parkirališnih mjesta. Sustav na dostupnoj karti (preko Google Maps servisa) nudi mogućnost unosa trenutne lokacije korisnika na temelju čega se prikazuju prijedlozi najbližih objekata (parkirališta i garaža) sa slobodnim mjestima. Nakon unosa trenutne lokacije u sustav, izračunat će se udaljenost svakog objekta od lokacija te će se prikazati parkiralište ili garaža najbliže korisniku uz uvjet da mora imati više od deset slobodnih mjesta u trenutku unosa korisnikove lokacije. Korisniku će biti ponuđen objekt koji odgovara odabranim uvjetima, dok on sam može odabrati neko drugo parkiralište ili garažu s karte. U slučaju kada na geografsko najbližoj lokaciji ima manje od deset slobodnih mjesta, prednost se daje sljedećoj lokaciji koja ima veći broj slobodnih mjesta radi sprječavanja nedostatka slobodnih parkirališnih mjesta u trenutku vozačeva dolaska na odabrano mjesto. Naime, u slučaju da korisnik ode na parkiralište ili garažu s manje od deset slobodnih mjesta, postoji mogućnost da u trenutku korisnikova dolaska na odabrano mjesto više ne bude slobodnih mjesta za parkiranje jer su drugi korisnici već prije njega došli na isti objekt i zauzeli slobodna parkirališna mjesta. Postavljanjem ovog ograničenja nastojimo izbjeći takvu situaciju te pokušavamo omogućiti da bilo koji posjetitelj ili klijent može naći bar jedno parkirališno mjesto. Dodatno su dostupni cjenik usluga i registracija korisnika. Da bi posjetitelj mogao rezervirati mjesto, mora ili napraviti registraciju ili se prijaviti s registriranim korisničkim imenom i lozinkom ako je već napravio registraciju. Ako nije napravljena registracija, mogući punoljetni klijent mora tijekom registracije osim odabranog željenog jedinstvenog korisničkog imena i lozinke preko *online* obrasca priložiti sljedeće podatke:

* OIB
* ime
* prezime
* adresa
* e-mail
* kontakt telefon
* broj kreditne kartice

Dva korisnika ne mogu imati isto korisničko ime te se tijekom registracije provjerava postoji li odabrano korisničko ime u bazi podataka. U slučaju da korisnik želi odabrati već zauzeto korisničko ime, sustav će javiti grešku te će korisnik morati odabrati drugo korisničko ime koje nije zauzeto da može završiti registraciju. Završenom registracijom posjetitelji postaju klijenti te dobivaju dodatne mogućnosti koje su dostupne registriranim korisnicima. Oni se prijavljuju u sustav pomoću odabranog korisničkog imena i lozinke. Uz sve dostupne sadržaje posjetiteljima, klijenti mogu rezervirati parkirališno mjesto jednokratno ili ponavljajuće uz mogućnost otkazivanja rezervacije te se mogu prijaviti i odjaviti iz sustava i izbrisati svoj korisnički račun. Rezervacija se vrši pomoću karte gdje klijent odabire željeno parkirališno mjesto na odabranom ili predloženom parkiralištu (vlastiti odabir ili preko predloženog najbližeg parkirališta na temelju unesene trenutne lokacije korisnika). Također je važno da klijent u trenutku kad želi napraviti rezervaciju odluči želi li jednokratnu ili ponavljajuću rezervaciju. Jednokratna rezervacija znači da klijent može u sadašnjem trenutku rezervirati parkirališno mjesto barem 6 sati unaprijed na period kraći od 24 sata, dok ponavljajuća rezervacija znači da klijent može rezervirati parkirališno mjesto ili trajno ili na najmanje 1 sat te se takva rezervacija ponavlja barem jednom tjedno kroz mjesec dana. Ponavljajuću je rezervaciju moguće otkazati ili produžiti te ta odluka ovisi o samom klijentu. U slučaju otkazivanja rezervacije klijentu se ne vraćaju novčana sredstva koja je potrošio na rezervaciju te se odabrana rezervacija poništava uz oslobađanje parkirališnog mjesta za drugog klijenta koji će možda htjeti napraviti rezervaciju na istom mjestu. Sve se vrste rezervacija naplaćuju u trenutku kad korisnik potvrdi svoj odabir izravnim terećenjem kreditne kartice ako to dopuštaju dostupna sredstva na kreditnoj kartici: kod jednokratnih se rezervacija sredstva s kreditne kartice skidaju u trenutku kad korisnik rezervira parkirališno mjesto, dok se kod trajnih sredstva skidaju svakih 30 dana. Prije svakog terećenja sustav provjerava klijentova sredstva na računu te u slučaju nedostatka sredstava sustav sam poništava rezervaciju. Cijena parkiranja je određena cjenikom koji je lako dostupan na web-stranici te registrirani korisnici imaju pravo popusta na sve usluge parkiranja. Popust za registrirane korisnike vrijedi od trenutka registracije te je također definiran i dostupan na istoj web-stranici kao i cjenik.

Administratori su još jedna važna skupina korisnika. Uz potpuni pristup sustavu i nadzoru svakog njegovog dijela, oni mogu proglasiti postojećeg korisnika administratorom te mijenjati podatke o postojećim resursima (podatci o svakom dostupnom otvorenom parkiralištu ili garaži za parkiranje, izmjena podataka o ukupnom broju parkirališnih mjesta na odabranom objektu, itd.) i klijentima. Također mogu ovisno o uspješnosti u poslovanju tvrtke stvarati nova parkirališta, garaže i parkirališna mjesta te brisati postojeće. Treba nadodati da, iako administrator može ažurirati broj parkirališnih mjesta na odabranom objektu, za njega postoji ograničenje koje ne dopušta da se broj parkirališnih mjesta nekoga parkirališta ili garaže smanji ispod deset.

Nadogradnje sustava su moguće i potrebne nakon završetka projekta radi zadržavanja kohezije i sprječavanja daljnjeg rasipanja funkcionalnosti. One mogu biti napravljene u obliku nadogradnje informacijskog sustava ili zamjenjivanja postojećih senzora kvalitetnijim i novijim senzorima koji bi onda mogli davati i druge informacije osim stanja slobodnih parkirališnih mjesta. Informacijski sustav može biti unaprijeđen tako da mu se dodaju nove funkcionalnosti ili da sustav bude još interaktivniji. Sve se nadogradnje mogu naknadno raspraviti i implementirati ako će se ovaj sustav nastaviti dodatno razvijati.

Pošto će naš informacijski sustav biti web-aplikacija, odlučili smo se za objektno-orijentirani jezik PHP5 zbog jednostavnosti i mogućnosti te njegove prilagođenosti Webu.

1. Rječnik pojmova

**ANSI:** Američki nacionalni institut za standarde je neprofitna organizacija u SAD-u koja se bavi razvojem i održavanjem volonterskih standarda za produkte, servise, sisteme i osoblje.

**Baza podataka:** Baza podataka je izmjenjivo spremište informacija iz svijeta aplikacija te programska podrška koja sakriva od korisnika fizičku organizaciju i reprezentaciju informacija. Informacijama pohranjenim u bazi podataka može se pristupati s logičke razine bez potrebe za uključivanjem fizičkih koncepata implementacije.

**Bootstrap:** Najpopularniji framework za HTML, CSS i JavaScript koji se sastoji od raznih predložaka koji olakšavaju izradu web-stranica.

**CSS:** Dodatak prezentacijskom jeziku HTML-u koji se zbog svoje sličnosti u strukturi najviše koristi za postavljanje osnovnih stilova pa da se kod ekvivalentan HTML-u ne mora ponovno pisati. Također se dosta koristi za pozicioniranje.

**Framework:** U programskom inženjerstvu programski okvir je apstrakcija u kojoj se softver koji pruža generičku funkcionalnost može mijenjati dodatnim funkcionalnostima i kodovima koje je napisao sam korisnik čime se osigurava specifičnost aplikacije.

**Google Maps:** Web-usluga za mapiranje i prikaz geografskog područja.

**HTML:** Prezentacijski jezik za izradu web-stranica.

**JavaScript:** Najpopularniji skriptni programski jezik koji se izvršava u web pregledniku na strani korisnika. Najviše se koristi kao dio web-browsera što omogućava klijentima (skriptama) komunikaciju s korisnikom, kontrolu preglednika i promjenu prikazanog sadržaja.

**jQuery:** Najpopularnija besplatna JavaScript biblioteka napravljena da pojednostavni pojedine funkcionalnosti poput manipulacije dokumentima, animacija i Ajaxa.

**MVC:** Oblikovni obrazac koji dijeli aplikaciju na tri sloja: Model (model), View (pogled) i Controller (kontroler).

**PHP**: Objektno-orijentirani jezik napravljen za razvoj web-stranice ili privatne intranetske mreže, ali se također koristi i kao programski jezik za širu uporabu.

**SQL:** Strukturirani upitni jezik za upravljanje bazom podataka. Omogućava pristup samoj bazi te manipulaciju njezinim podatcima. SQL je stardardiziran preko standarda ANSI i ISO.

**UML**: Unified Modeling Language ili jezik za unificirano modeliranje grafički je jezik za vizualiziranje, specificiranje, konstruiranje i dokumentiranje artefakata softverski intenzivnog sustava.

**Yii**: Open source, objektno orijentirani framework napravljen tako da prati MVC PHP oblikovni obrazac.

1. Funkcionalni zahtjevi

**Dionici su:**

* Neregistrirani korisnici
* Registrirani korisnici
* Administrator(i)

**Aktori i njihovi funkcionalni zahtjevi:**

* Neregistrirani korisnik (posjetitelj), inicijator
  + može vidjeti stanje svih parkirališnih mjesta i parkirališta *online*
  + može vidjeti cjenik *online*
  + može unijeti trenutnu lokaciju i dobiti prijedlog najbližih parkirališta
  + svaka se punoljetna osoba može registrirati te pritom daje svoje osobne podatke
* Registrirani korisnik (klijent), inicijator
  + može vidjeti status parkirališnih mjesta i parkirališta *online*
  + može vidjeti cjenik *online*
  + može unijeti trenutnu lokaciju i dobiti prijedlog najbližih parkirališta
  + može se prijaviti u sustav i odjaviti iz sustava
  + može vidjeti svoje osobne podatke i napravljene rezervacije
  + može deaktivirati svoj račun
  + može rezervirati parkirališno mjesto
    - jednokratno – barem 6h unaprijed, na period **T** < 24h, naplata odmah
    - ponavljajuće – na period 1h < **T** < 24h, ponavlja se frekvencijom **f** >= 7 dana kroz barem mjesec dana (granulacija od 30 dana)
  + može otkazati postojeću rezervaciju
* Administrator, inicijator
  + može se prijaviti u sustav i odjaviti iz sustava
  + može proglasiti postojećeg korisnika administratorom
  + može mijenjati svoje podatke
  + može dodavati, mijenjati i brisati parkirališta
* Baza podataka, sudionik
  + sadrži sve podatke o korisničkim računima, parkiralištima, rezervacijama i cijenama
  + pruža sučelje za pristup podatcima
* Senzor(i), inicijator
  + mijenjaju podatke o statusu pojedinih parkirališnih mjesta
* Kartična kuća, sudionik
  + provodi plaćanje
* Poslužitelj, sudionik
  + pristup bazi podataka na zahtjev korisnika
  + *hosting*
* Google Maps8 servis, sudionik
  + interaktivan prikaz lokacija i informacija o svim parkiralištima
  1. Opis obrazaca uporabe

U preduvjetima se podrazumijeva da glavni sudionik ima pristup mreži, a poslužitelj i baza podataka su dostupni. U tom polju se stoga navode samo dodatni preduvjeti.

**Korisnik** – dijeli se na posjetitelja (nema korisnički račun), klijenta (ima korisnički račun) i administratora (klijent s dodatnim ovlastima)

**UC1** – Pregled informacija o parkiralištima

* **Glavni sudionik:** Korisnik
* **Cilj:** Pregledati informacije o parkiralištu
* **Ostali sudionici:** Baza podataka, Google Maps servis, poslužitelj
* **Preduvjeti:**
* **Rezultat:** Informacije o svim parkiralištima i slobodnim parkirališnim mjestima.
* **Željeni scenarij:**

1. Na web-stranici korisnik odabire prikaz informacija o parkiralištima.
2. Poslužitelj dohvaća informacije o svim parkiralištima iz baze podataka za sva parkirališta i prikazuje ih na mapi koristeći Google Maps servis.
3. Korisnik odabire parkiralište koje ga zanima.
4. Korisnik dobiva uvid u detaljne informacije o parkiralištu (lokacija, veličina, broj slobodnih mjesta).

* **Mogući drugi scenariji:** Nema

**UC2** – Pregled cjenika usluga

* **Glavni sudionik:** Korisnik
* **Cilj:** Pregledati cijene usluga
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:**
* **Rezultat:** Cijene usluga parkiranja na otvorenom parkiralištu ili u zatvorenoj garaži
* **Željeni scenarij:**

1. Korisnik na početnoj stranici slijedi poveznicu "Cjenik".
2. Poveznica odvede korisnika na prikaz cjenika na početnoj stranici.
3. Korisnik vidi sve cijene usluga (parkirališta na otvorenom i garaže).

* **Mogući drugi scenariji:** Nema

**UC3** – Izmjena cjenika usluga

* **Glavni sudionik:** Administrator
* **Cilj:** Izmijeniti cijene usluga
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Administrator je prijavljen u sustav
* **Rezultat:** Cijene usluga su uspješno izmijenjene
* **Željeni scenarij:**

1. Administrator odabire opciju izmjene cjenika.
2. Administrator unosi izmjene.
3. Poslužitelj potvrđuje da su unesene izmjene ispravne i sprema ih u bazu podataka.

* **Mogući drugi scenariji:**

3. U slučaju kad uneseni podatci nisu ispravni poslužitelj odbija izvršiti izmjene i daje povratnu informaciju o problemu

**UC4** – Dobivanje prijedloga

* **Glavni sudionik:** Korisnik
* **Cilj:** Dobiti lokaciju najbližeg parkirališta sa slobodnim parkirališnim mjestima
* **Ostali sudionici:** Google Maps servis, poslužitelj, baza podataka
* **Preduvjeti:** Nema
* **Rezultat:** Korisnik je dobio lokaciju najbližeg parkirališta sa slobodnim parkirališnim mjestima
* **Željeni scenarij:**

1. Korisnik klikom miša ili upisivanjem koordinata unese svoju trenutnu lokaciju.
2. Poslužitelj na temelju lokacije korisnika i informacija iz baze podataka pronalazi najbliže parkiralište s više od deset slobodnih parkirališnih mjesta.
3. Poslužitelj daje prijedlog najbližeg parkirališta sa slobodnim parkirališnim mjestom.

* **Mogući drugi scenariji:**

1. Ne postoji parkiralište s deset ili više slobodnih mjesta te umjesto toga korisnik kao prijedlog dobiva lokaciju najbližeg parkirališta s barem jednim slobodnim mjestom.
2. Ne postoji parkiralište s barem jednim slobodnim mjestom te korisnik dobiva povratnu informaciju od poslužitelja da su sva parkirališta popunjena.

**UC5** – Registracija korisnika

* **Glavni sudionik:** Posjetitelj
* **Cilj:** Stvoriti korisnički račun
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Posjetitelj je stariji od 18 godina, nije klijent niti administrator
* **Rezultat:** Uspješno stvoren korisnički račun.
* **Željeni scenarij:**

1. Posjetitelj putem web-obrasca unosi potrebne informacije (*željeno jedinstveno korisničko ime, OIB, ime, prezime, adresa, e-mail, kontakt telefon, broj kreditne kartice*) i zahtijeva stvaranje korisničkog računa.
2. Poslužitelj potvrđuje ispravnost unesenih informacija, sprema ih u bazu podataka, stvara korisnički račun i daje povratnu informaciju o uspješno stvorenom računu.

* **Mogući drugi scenariji:**

1. Uneseni podatci nisu ispravni ili ne zadovoljavaju uvjete korištenja te se zbog toga korisnički račun ne stvara i poslužitelj daje posjetitelju povratnu informaciju o problemu.

**UC6** – Prijava korisnika u sustav

* **Glavni sudionik:** Administrator ili klijent
* **Cilj:** Uspješna prijava – mogućnost korištenja funkcionalnosti za koje je potrebno biti prijavljen u sustav
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Glavni sudionik je registriran i nije prijavljen u sustav te njegov korisnički račun nije ni blokiran ni neaktivan
* **Rezultat:** Glavni sudionik je prijavljen i može koristiti dodatne funkcionalnosti (UC7, UC8, UC9)
* **Željeni scenarij:**

1. Glavni sudionik unosi korisničko ime i lozinku.
2. Poslužitelj provjerava postoji li korisnički račun u bazi podataka s tom kombinacijom korisničkog imena i lozinke.
3. Poslužitelj utvrđuje da takav račun postoji te započinje sesiju s glavnim sudionikom.
4. Sesija završava odjavom glavnog sudionika ili njegovom neaktivnošću tijekom vremena duljeg od propisanog.

* **Mogući drugi scenariji:**

1. Ne postoji korisnički račun s tom kombinacijom korisničkog imena i lozinke te poslužitelj odbija uspostaviti sesiju.

**UC7** – Pregled vlastitog korisničkog računa

* **Glavni sudionik:** Administrator ili klijent
* **Cilj:** Pregled vlastitih korisničkih podataka i rezervacija
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:**
* **Rezultat:** Glavni sudionik dobiva uvid u svoje podatke i rezervacije.
* **Željeni scenarij:**

1. Glavni sudionik na početnoj stranici slijedi poveznicu "Moj profil".
2. Poslužitelj iz baze podataka dohvaća tražene korisničke podatke .
3. Traženi podatci se prikazuju u prozoru preglednika.

* **Mogući drugi scenariji:** Nema

**UC8** – Izmjena vlastitog korisničkog računa

* **Glavni sudionik:** Administrator ili klijent
* **Cilj:** Promjena osobnih podataka vlastitog korisničkog računa
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Glavni sudionik je prijavljen u sustav
* **Rezultat:** Promijenjeni su podatci uspješno pohranjeni u bazu podataka
* **Željeni scenarij:**

1. Glavni sudionik šalje zahtjev za dohvat osobnih podataka iz baze podataka koje želi izmijeniti.
2. Poslužitelj provjerava ispravnost poslanog zahtjeva te dohvaća i šalje zatražene podatke ako je upit dozvoljen.
3. Glavni sudionik mijenja podatke koristeći korisničko sučelje web-aplikacije.
4. Poslužitelj potvrđuje da su unesene izmjene ispravne i sprema ih u bazu podataka (*ime, prezime, adresa, e-mail, kontakt telefon, broj kreditne kartice*).

* **Mogući drugi scenariji:**

1. Administrator pokušava izmijeniti vlastiti status računa te poslužitelj odbija izmijeniti podatke i daje povratnu informaciju o problemu.
2. Unesene izmjene nisu ispravne te poslužitelj odbija izmijeniti podatke i daje povratnu informaciju o problemu.

**UC9** – Deaktivacija vlastitog korisničkog računa

* **Glavni sudionik:** Klijent
* **Cilj:** Deaktivirati korisnički račun
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Klijent je prijavljen u sustav, nije administrator
* **Rezultat:** Korisnički račun je u bazi podataka označen kao neaktivan
* **Željeni scenarij:**

1. Klijent u opcijama korisničkog računa odabire deaktivaciju korisničkog računa.
2. Nakon utvrđivanja da korisnički račun nema aktivne rezervacije, poslužitelj u bazi podataka označava korisnički račun kao neaktivan.

* **Mogući drugi scenariji:**

1. Korisnički račun ima aktivnih rezervacija te poslužitelj odbija deaktivirati račun i javlja klijentu grešku zbog postojanja aktivnih rezervacija.

**UC10** – Stvaranje rezervacije

* **Glavni sudionik:** Klijent
* **Cilj:** Primitak, obrada i spremanje korisničkog zahtjeva za rezervacijom parkirališnog mjesta.
* **Sudionici:** Baza podataka, poslužitelj, kartična kuća
* **Preduvjeti:** Klijent ima pristup internetu i prijavljen je u sustav. Poslužitelj je dostupan.
* **Rezultat:** Klijent je ostvario potrebnu rezervaciju. Podatci o toj rezervaciji su uspješno poslani i spremljeni u bazu podataka.
* **Željeni scenarij:**

1. Klijent nakon pretraživanja željenog parkirališta odabire parkiralište, vrijeme i način rezervacije (jednokratna, periodična). Ako postoji ponavljajuća rezervacija, sustav šalje zahtjev za provjeru sredstava na računu i produživanje rezervacije bez zahtjeva korisnika.
2. Sustav traži od kartične kuće provjeru sredstava na korisničkom računu.
3. Kartična kuća obavlja transakciju i šalje novčani iznos koji odgovara rezervaciji s računa korisnika na račun tvrtke koja je vlasnik parkirališta.
4. Podatci o rezervaciji se šalju i spremaju u bazu podataka.
5. Nakon uspješne transakcije klijent dobiva obavijest o uspješno obavljenoj rezervaciji.

* **Mogući drugi scenariji:**

1. Klijent nema dovoljno sredstava na računu za obnavljanje periodične rezervacije te se rezervacija otkazuje uz spremanje njezinog statusa u bazi podataka.

**UC11** – Otkazivanje rezervacije

* **Glavni sudionik:** Administrator ili klijent
* **Cilj:** Otkazivanje korisničke rezervacije koja glavnom sudioniku više nije potrebna
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:**
* **Rezultat:** Glavni sudionik je uspješno otkazao rezervaciju te je ona označena neaktivnom u bazi podataka
* **Željeni scenarij:**

1. Glavni sudionik pri pregledu svojih rezervacija odabire onu koju želi otkazati te potvrđuje željenu akciju.
2. Poslužitelj u bazi podataka označuje rezervaciju neaktivnom.
3. Glavni sudionik dobiva obavijest o uspješnom otkazivanju rezervacije.

* **Mogući drugi scenariji:**
  + Nema

**UC12** – Blokiranje korisničkog računa

* **Glavni sudionik:** Administrator
* **Cilj:** Blokiranje korisničkog računa
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Administrator je prijavljen u sustav
* **Rezultat:** Korisnički račun je uspješno blokiran
* **Željeni scenarij:**

1. Administrator odabire kojem korisniku želi blokirati račun.
2. Poslužitelj potvrđuje da odabrani korisnički račun ne pripada administratoru i u bazi podataka ga označava blokiranim.

* **Mogući drugi scenariji:**

1. korisnički račun pripada administratoru te poslužitelj odbija blokirati račun i vraća povratnu informaciju o grešci

**UC13** – Izmjena privilegija korisničkog računa

* **Glavni sudionik:** Administrator
* **Cilj:** Proglašenje klijenta administratorom ili obratno
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Administrator je prijavljen u sustav
* **Rezultat:** Korisničkom računu je uspješno izmijenjen status privilegija
* **Željeni scenarij:**

1. Administrator odabire korisnički račun kojemu želi promijeniti status.
2. Poslužitelj potvrđuje da odabrani račun ne pripada glavnom sudioniku i u bazi podataka mijenja status privilegija odabranog korisničkog računa.

* **Mogući drugi scenariji:**

1. Sudionik pokušava izmijeniti vlastiti status privilegija te poslužitelj odbija promijeniti status i daje povratnu informaciju o grešci.

**UC14** – Dodavanje novog parkirališta

* **Glavni sudionik:** Administrator
* **Cilj:** Dodavanje novog parkirališta
* **Ostali sudionici:** Baza podataka, Google Maps servis, poslužitelj
* **Preduvjeti:** Administrator je prijavljen u sustav, parkiralište ima barem 10 mjesta
* **Rezultat:** Novo je parkiralište uspješno dodano
* **Željeni scenarij:**

1. Administrator ispunjava web-obrazac kako bi dodao novo parkiralište.
2. Poslužitelj provjerava ispravnost unosa.
3. Poslužitelj dodaje novo parkiralište u bazu podataka.

* **Mogući drugi scenariji:**

1. Lokacija novog parkirališta je već zauzeta te sustav odbija obaviti operaciju i daje povratnu informaciju o grešci.

**UC15** – Izmjena podataka o parkiralištu

* **Glavni sudionik:** Administrator
* **Cilj:** Izmjena podataka o postojećem parkiralištu
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Administrator je prijavljen u sustav, parkiralište postoji
* **Rezultat:** Podatci o postojećem parkiralištu su uspješno izmijenjeni
* **Željeni scenarij:**

1. Administrator odabire kojem parkiralištu želi promijeniti podatke.
2. Administrator unosi nove podatke o odabranom parkiralištu.
3. Poslužitelj potvrđuje da su novi podatci ispravni i sprema ih u bazu podataka.

* **Mogući drugi scenariji:**

1. Administrator smanjuje broj parkirališnih mjesta ispod 10 te poslužitelj odbija izmjeniti podatke i daje povratnu informaciju o grešci.
2. Nova je lokacija parkirališta već zauzeta te poslužitelj odbija izmjeniti podatke i daje povratnu informaciju o grešci.

.

**UC16** – Brisanje parkirališta

* **Glavni sudionik:** Administrator
* **Cilj:** Brisanje postojećeg parkirališta
* **Ostali sudionici:** Baza podataka, Google Maps servis, poslužitelj
* **Preduvjeti:** Administrator je prijavljen u sustav, parkiralište postoji
* **Rezultat:** Postojeće je parkiralište uspješno označeno kao nepostojeće
* **Željeni scenarij:**

1. Administrator odabire koje parkiralište želi izbrisati.
2. Poslužitelj potvrđuje da se parkiralište smije izbrisati i u bazi ga označava kao nepostojeće.

* **Mogući drugi scenariji:**

2. Odabrano parkiralište ne smije izbrisati te u tom slučaju poslužitelj odbija obaviti operaciju i daje povratnu informaciju o grešci

**UC17** – Pregled informacija o svim rezervacijama

* **Glavni sudionik:** Administrator
* **Cilj:** Primiti informacije o rezervacijama iz baze podataka
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Administrator je prijavljen u sustav
* **Rezultat:** Uspješno primljene informacije o rezervacijama parkirališnog mjesta
* **Željeni scenarij:**

1. Administrator šalje zahtjev za dohvat informacija o rezervacijama iz baze podataka poslužitelju.
2. Poslužitelj dohvaća tražene informacije iz baze podataka te ih šalje administratoru.

* **Mogući drugi scenariji:**

1. Traženi upit ne postoji u bazi podataka te poslužitelj obavještava administratora da nema ni jednog rezultata pretrage.

**UC18** – Pregled informacija o svim korisnicima

* **Glavni sudionik:** Administrator
* **Cilj:** Primiti informacije o korisniku iz baze podataka
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:** Administrator je prijavljen u sustav
* **Rezultat:** Uspješno primljene informacije o korisniku
* **Željeni scenarij:**

1. Administratoršalje poslužitelju zahtjev za dohvatom informacija o svim korisnicima iz baze podataka.
2. Poslužitelj dohvaća tražene informacije iz baze podataka te ih šalje administratoru.

* **Mogući drugi scenariji:**

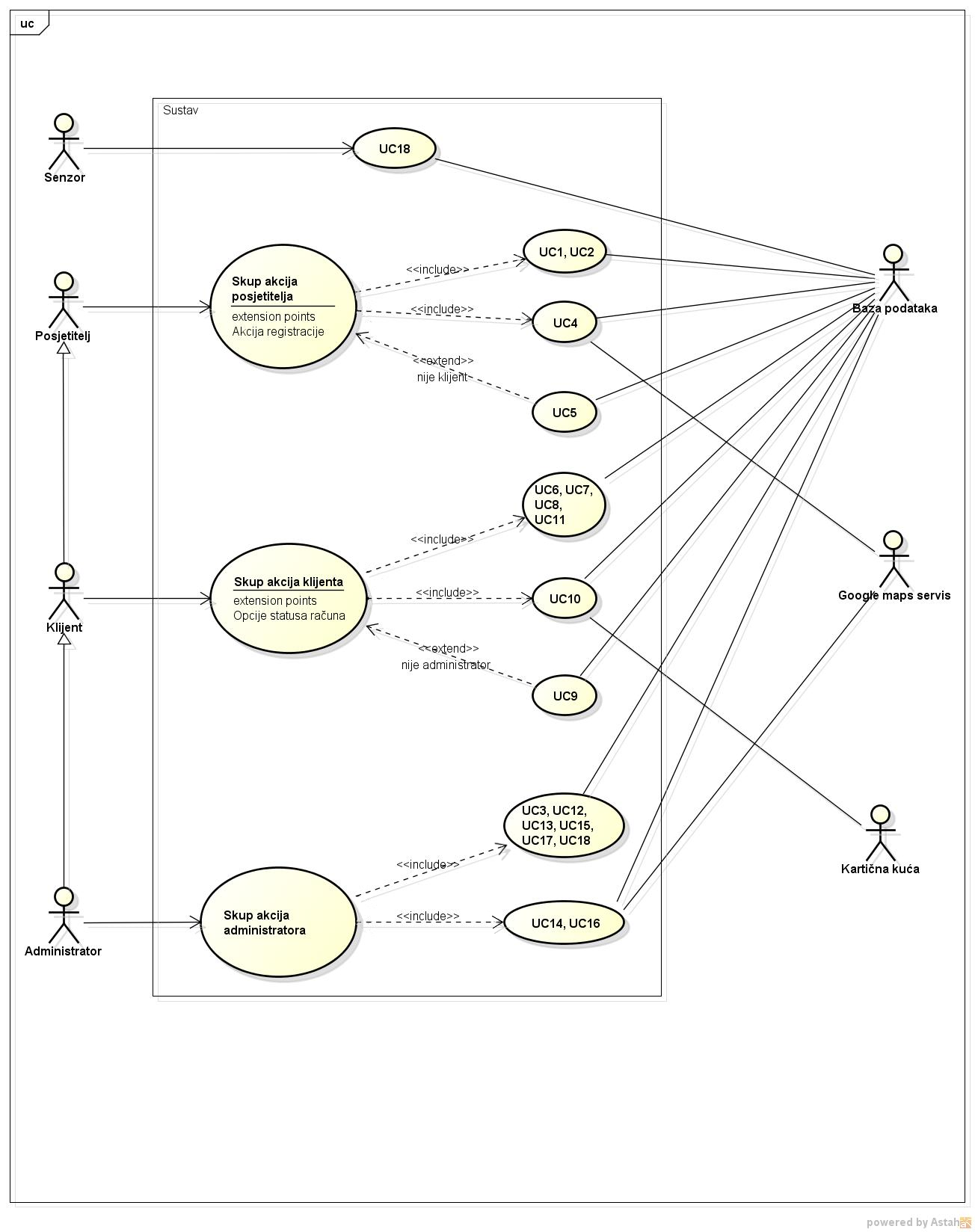
1. traženi upit ne postoji u bazi podataka te poslužitelj obavještava administratora da nema ni jednog rezultata pretrage

**UC19** – Promjena statusa parkirališnog mjesta

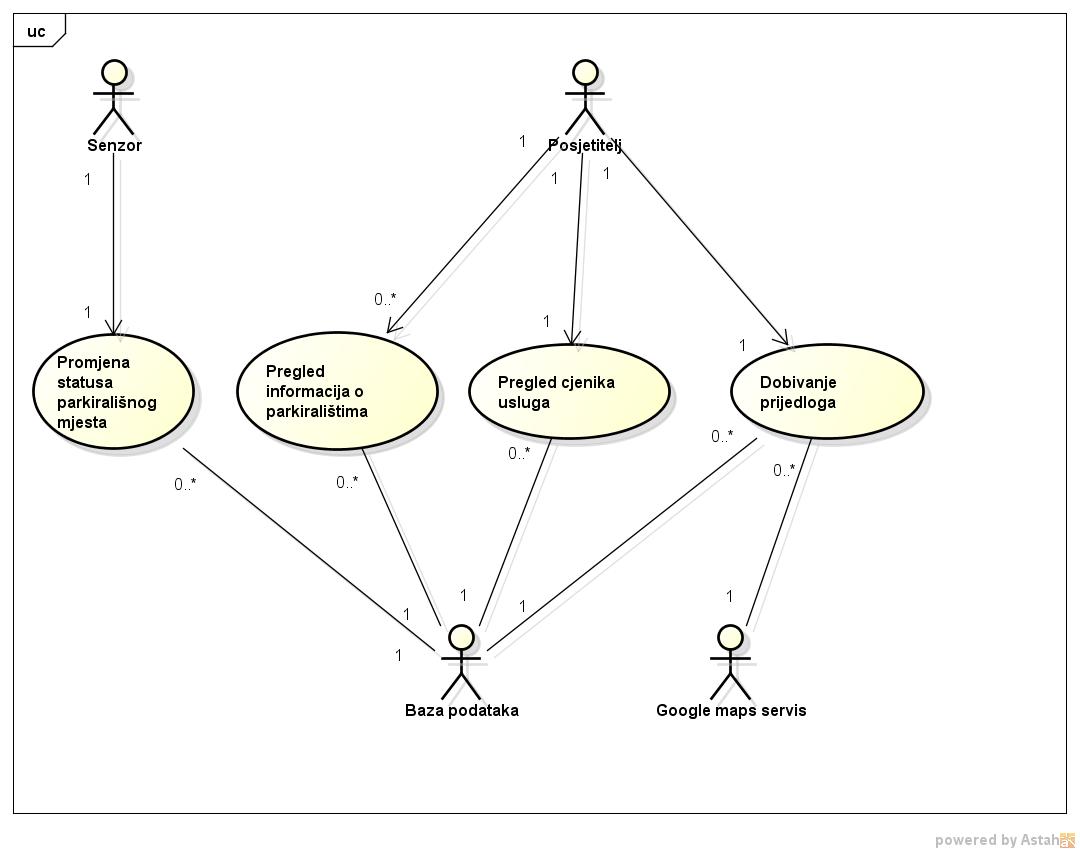
* **Glavni sudionik:** Senzor
* **Cilj:** Slanje podatka sa senzora na poslužitelj te upis promjene u bazu podataka
* **Ostali sudionici:** Baza podataka, poslužitelj
* **Preduvjeti:**
* **Rezultat:** Uspješno izmijenjeni odgovarajući atributi u bazi podataka
* **Željeni scenarij:**

1. Senzor očita promjenu stanja parkirališnog mjesta.
2. Senzor šalje poslužitelju zahtjev za promjenom statusa svog parkirališnog mjesta.
3. Poslužitelj na temelju *parkingSpotId*-a mijenja vrijednost atributa *sensor* u relaciji *parkingSpot* u bazi podataka u jednu od sljedećih vrijednosti:  
      
   TRUE (zauzeto mjesto) ako je prethodno bila vrijednost FALSE.  
   FALSE (slobodno mjesto) ako je prethodno bila vrijednost TRUE.

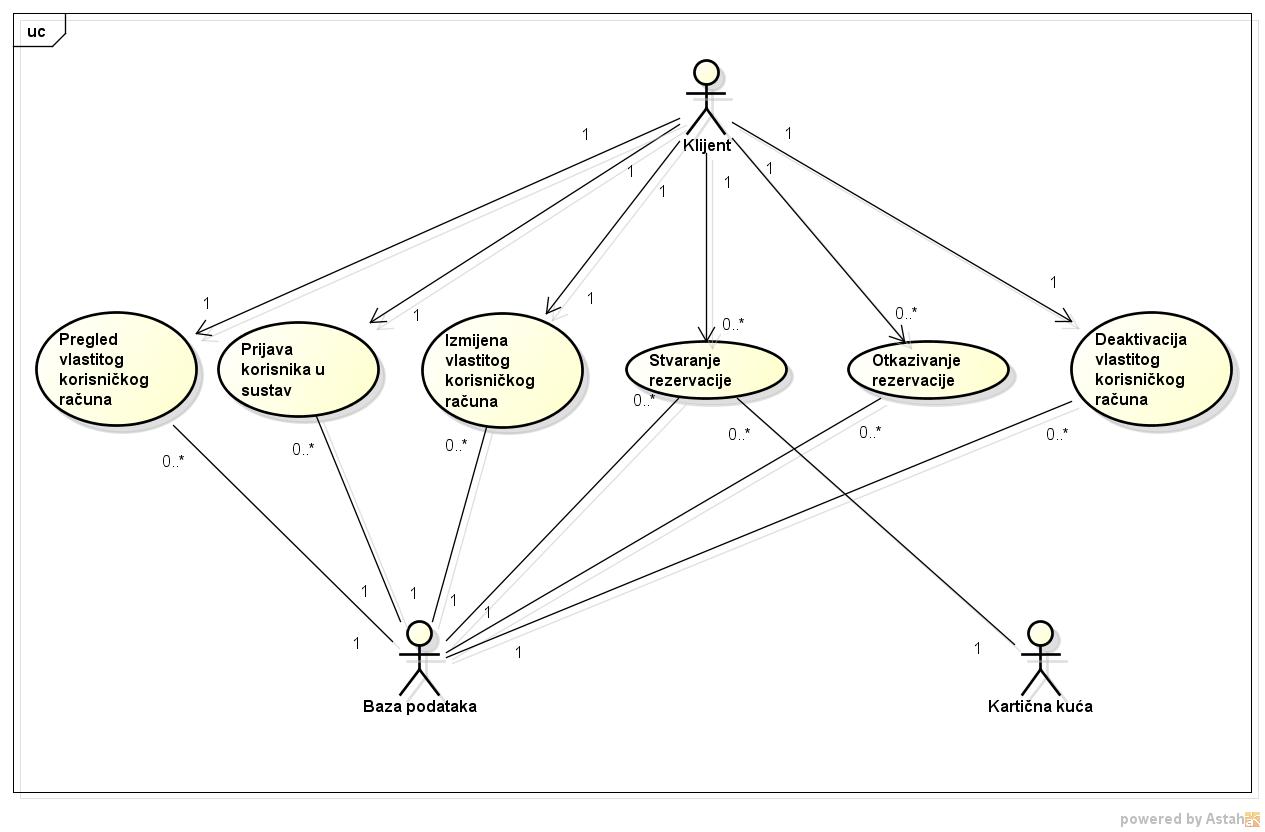
* **Mogući drugi scenariji:** Nema



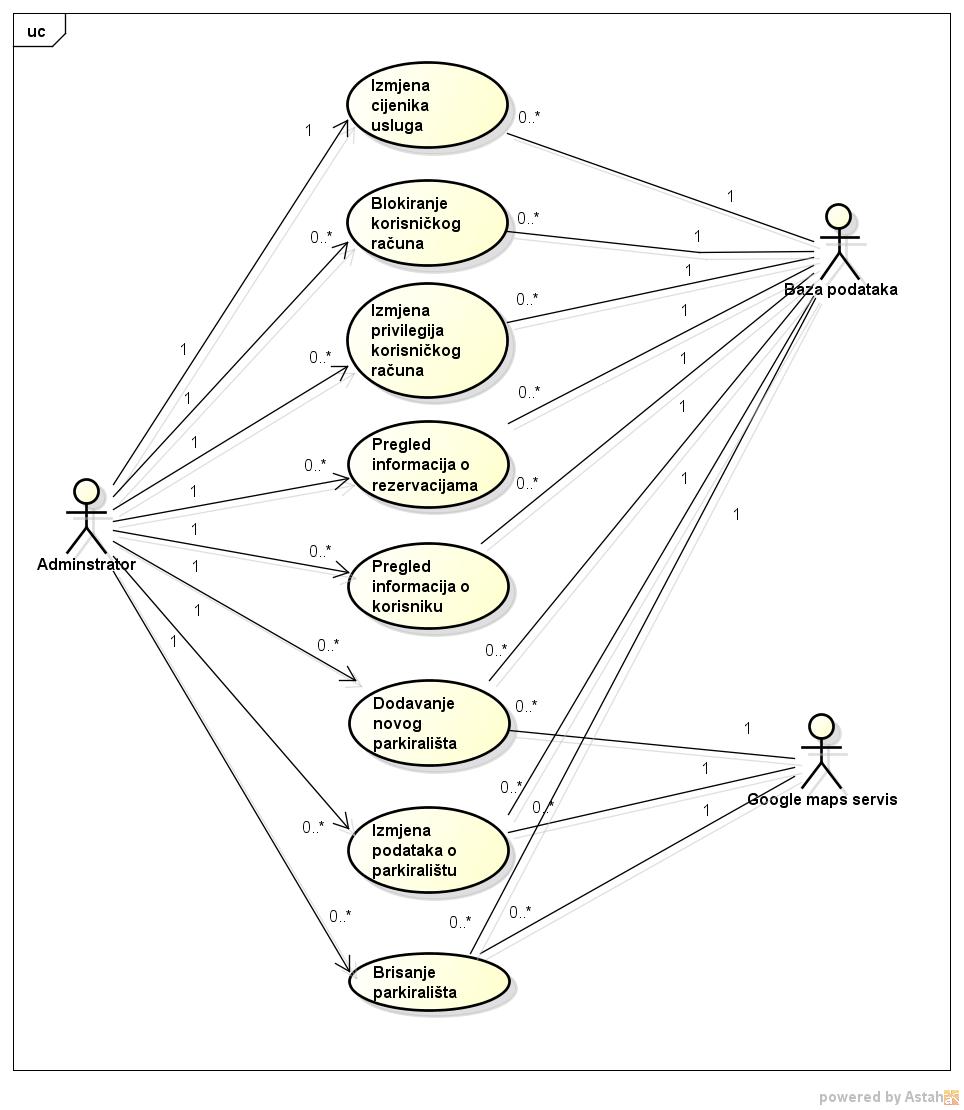
**Slika 4.1.1.**: Dijagram obrasca uporabe 1



**Slika 4.1.2.**: Dijagram obrasca uporabe 2 - Posjetitelj



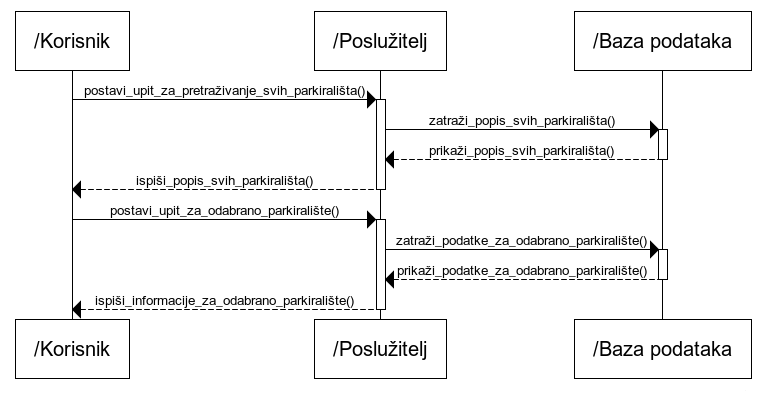
**Slika 4.1.3.**: Dijagram obrasca uporabe 3 - Klijent



**Slika 4.1.4.**: Dijagram obrasca uporabe 4 - Administrator

* 1. Sekvencijski dijagrami

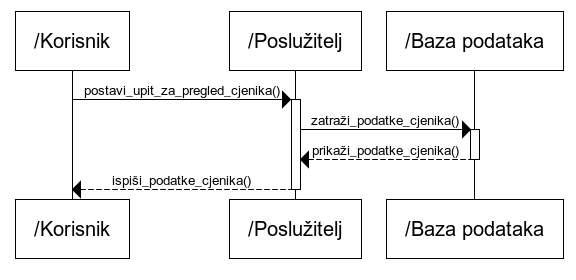
**UC1** – Pregled informacija o parkiralištima

Korisnik postavlja upit za pretraživanje parkirališta te isti upit koji je server dobio šalje do baze podataka koja mu vraća podatke koje server ispisuje korisniku u obliku popisa svih parkirališta. Korisnik odabire parkiralište koje ga zanima i ponovno šalje upit serveru koji traži informacije o odabranom parkiralištu od baze podataka. Baza podataka šalje tražene podatke o lokaciji, veličini parkirališta i broju slobodnih mjesta na tom parkiralištu serveru koji ih zatim prikazuje korisniku. Odgovarajući dijagram prikazan je na slici 4.2.1.

**Slika 4.2.1.** Sekvencijski dijagram za obrazac uporabe UC1 – *Pregled informacija o parkiralištima*

**UC2** – Pregled cjenika usluga

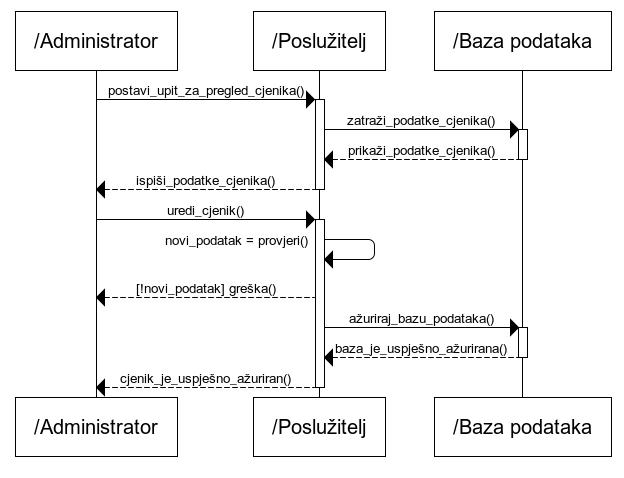
Korisnik postavlja upit za pregledom cjenika te upit šalje serveru koji od baze podataka traži podatke koji se odnose na cjenik. Baza podataka šalje serveru podatke koji se zatim prikazuju korisniku u obliku tablice. Odgovarajući dijagram prikazan je na slici 4.2.2.



**Slika 4.2.2.** Sekvencijski dijagram za obrazac uporabe UC2 *– Pregled cjenika usluga*

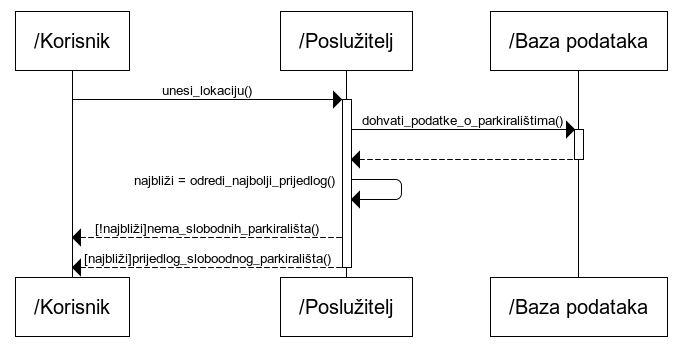
**UC3** – Izmjena cjenika usluga

Administrator šalje upit za pregledom cjenika te upit šalje serveru koji od baze podataka traži podatke koji se odnose na cjenik. Baza podataka šalje serveru tražene podatke koji se zatim prikazuju administratoru u obliku tablice. Zatim administrator od servera traži izmjenu podataka unutar servera koji provjerava je li novi podatak u skladu s ograničenjima te vraća administratoru poruku o neuspješnosti ažuriranja cjenika ako nije. Ako je novi podatak u skladu s pravilima, server šalje zahtjev bazi podataka za izmjenom podataka. Ako je podatak uspješno izmijenjen, šalje se "True" poruka serveru koji prikazuje administratoru da je podatak uspješno izmijenjen. Ako podatak nije uspješno izmijenjen, unutar baze se šalje poruka serveru "False" koji prikazuje administratoru da je podatak neupješno izmijenjen. Odgovarajući dijagram prikazan je na slici 4.2.3.



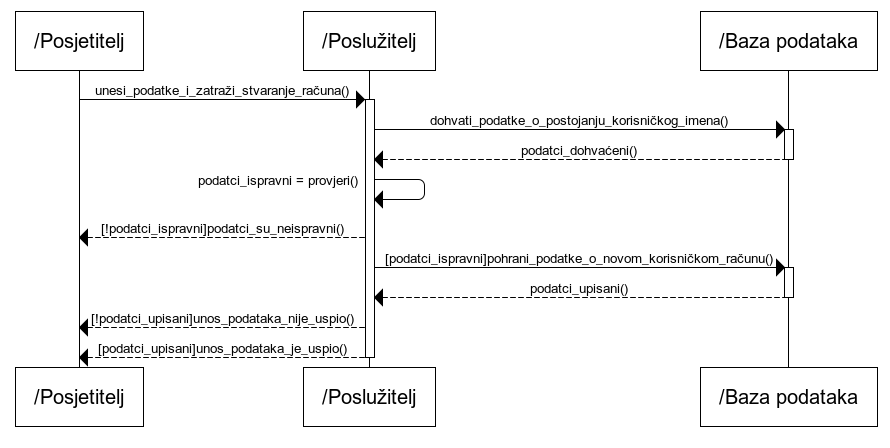
**Slika 4.2.3.** Sekvencijski dijagram za obrazac uporabe UC3 – *Izmjena cjenika usluga*

**UC4** – Dobivanje prijedloga

Korisnik šalje svoju lokaciju poslužitelju. Poslužitelj potom dohvaća podatke o parkiralištima iz baze podataka te na temelju dohvaćenih podataka i lokacije određuje najbolji prijedlog koji prosljeđuje natrag korisniku: ako nema parkirališta s barem jednim slobodnim mjestom, korisnik dobiva poruku da nema slobodnih parkirališta, dok u suprotnom slučaju dobiva prijedlog najbližeg parkirališta (neovisno o broju slobodnih parkirališnih mjesta). Odgovarajući dijagram prikazan je na slici 4.2.4.

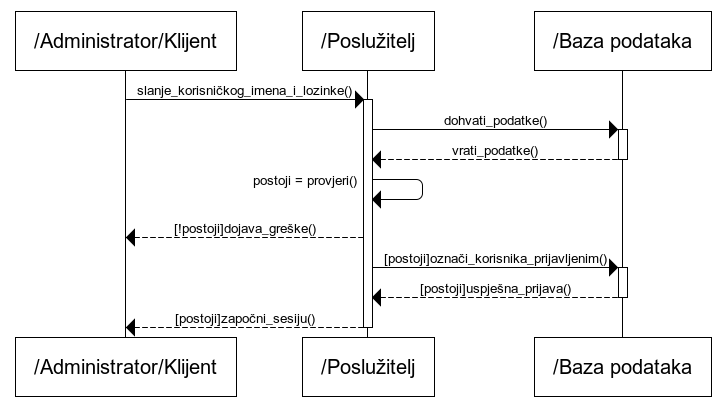
**Slika 4.2.4.** Sekvencijski dijagram za obrazac uporabe UC4 *– Dobivanje prijedloga*

**UC5** – Registracija korisnika

Posjetitelj unosi svoje podatke i traži stvaranje korisničkog računa. Poslužitelj potom dohvaća podatke o korisničkim imenima iz baze podataka, provjerava dostupnost zatraženog korisničkog imena i ispravnost ostalih unesenih podataka. Ako su uneseni podatci ispravni, poslužitelj stvara novi korisnički račun tako da pohranjuje podatke o njemu u bazu podataka. Ako postoji greška među unesenim podatcima, poslužitelj obavještava posjetitelja o neuspješnom stvaranju korisničkog računa. Odgovarajući dijagram prikazan je na slici 4.2.5.

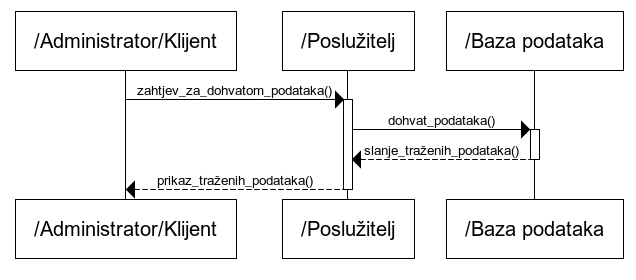
**Slika 4.2.5.** Sekvencijski dijagram za obrazac uporabe UC5 *– Registracija korisnika*

**UC6** – Prijava korisnika u sustav

Administrator ili klijent upisuju svoje korisničko ime i lozinku, a poslužitelj dohvaća podatke iz baze podataka na temelju unosa. Ako taj račun ne postoji, glavnom sudioniku se dojavljuje greška. Ako račun postoji, glavnog sudionika se označava prijavljenim u bazi podataka koja šalje potvrdu uspješne prijave. Poslužitelj zatim započinje sesiju. Odgovarajući dijagram prikazan je na slici 4.2.6.

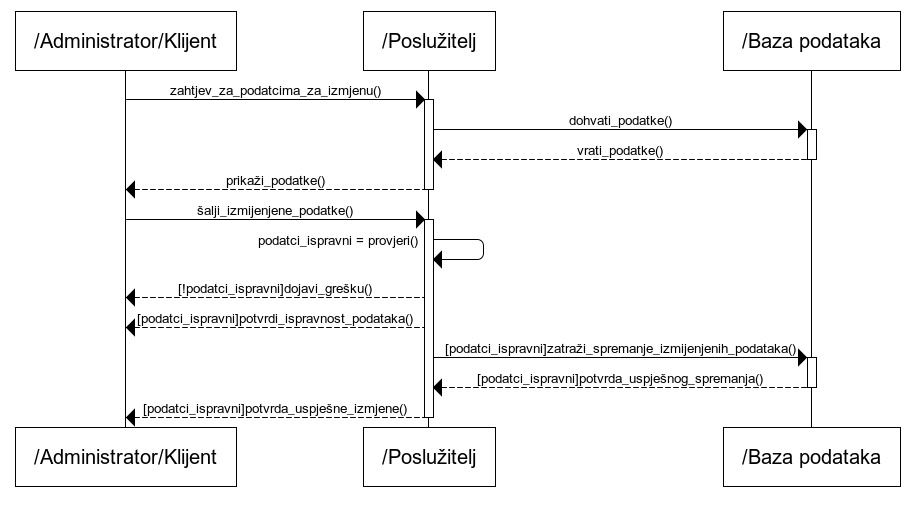
**Slika 4.2.6.** Sekvencijski dijagram za obrazac uporabe UC6 – *Prijava korisnika u sustav*

**UC7** – Pregled vlastitog korisničkog računa

Administrator ili klijent postavlja zahtjev za prikazom podataka odabirom poveznice, poslužitelj ih dohvaća iz baze podataka te se podatci prikazuju na web-stranici. Odgovarajući dijagram prikazan je na slici 4.2.7.

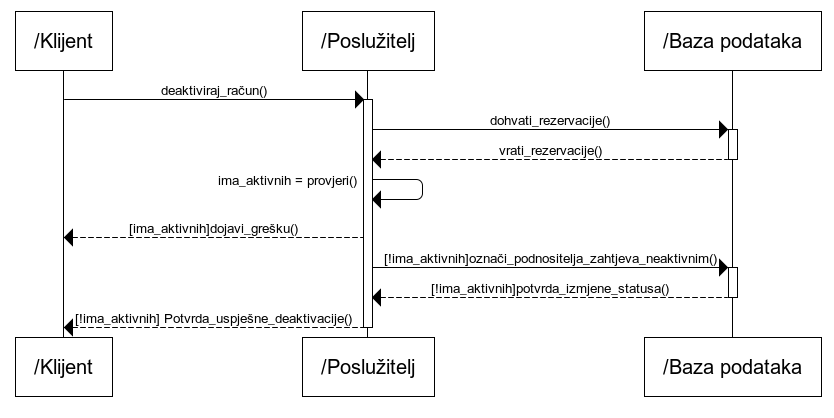
**Slika 4.2.7.** Sekvencijski dijagram za obrazac uporabe UC7 – *Pregled vlastitog korisničkog računa*

**UC8** – Izmjena vlastitog korisničkog računa

Administrator ili klijent odabire podatke koje želi izmijeniti, a poslužitelj ih dohvaća iz baze podataka te se oni prikazuju glavnom sudioniku. Glavni sudionik mijenja podatke, a poslužitelj provjerava jesu li ispravnog tipa i smiju li se mijenjati. Ako je tip podataka kriv ili se podatci ne smiju mijenjati, poslužitelj glavnom sudioniku javlja grešku. Ako to nije slučaj, glavnom sudioniku se potvrđuje da su podatci ispravni te ih sprema u bazu podataka. Uspješno spremanje se javlja poslužitelju, a zatim i glavnom sudioniku. Odgovarajući dijagram prikazan je na slici 4.2.8.

**Slika 4.2.8.** Sekvencijski dijagram za obrazac uporabe UC8 – *Izmjena vlastitog korisničkog računa*

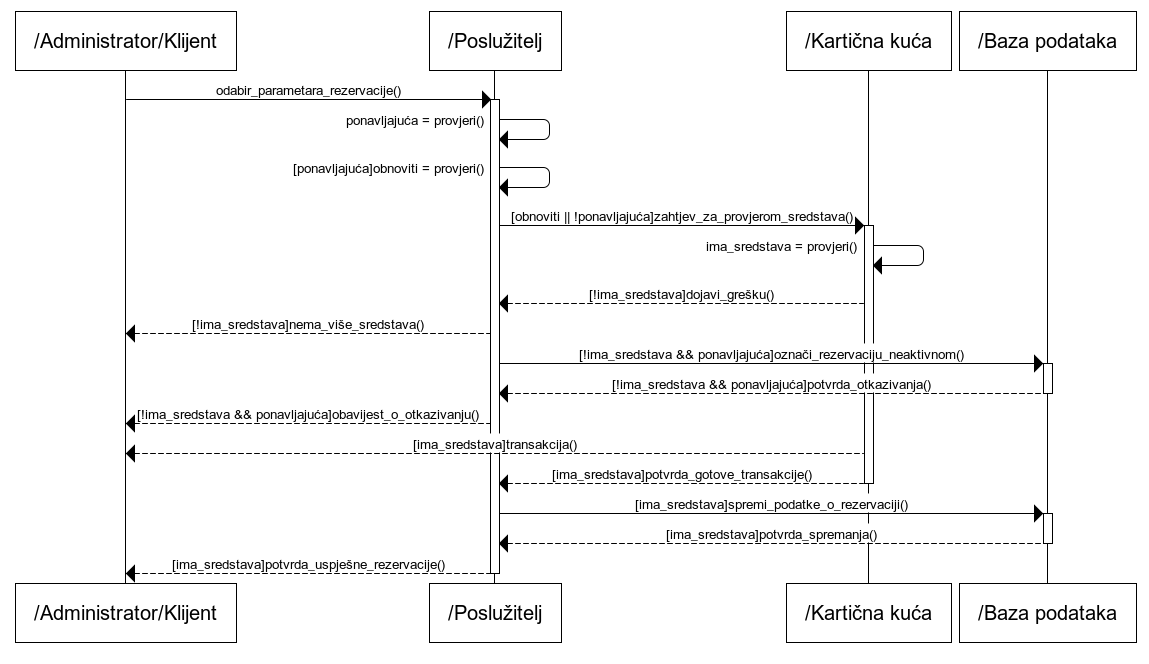
**UC9** – Deaktivacija vlastitog korisničkog računa

Klijent šalje zahtjev za deaktivacijom korisničkog računa poslužitelju koji provjerava ima li u bazi podataka aktivnih rezervacija za taj račun. Ako nema aktivnih rezervacija, moguće je deaktivirati račun pa se račun podnositelja zahtjeva za deaktivaciju označava neaktivnim u bazi podataka, inače se javlja greška klijentu. Baza javlja uspješnost operacije spremanja statusa poslužitelju koji zatim javlja uspješnost deaktivacije podnositelju zahtjeva. Odgovarajući dijagram prikazan je na slici 4.2.9.

**Slika 4.2.9.** Sekvencijski dijagram za obrazac uporabe UC9 – *Deaktivacija vlastitog korisničkog računa*

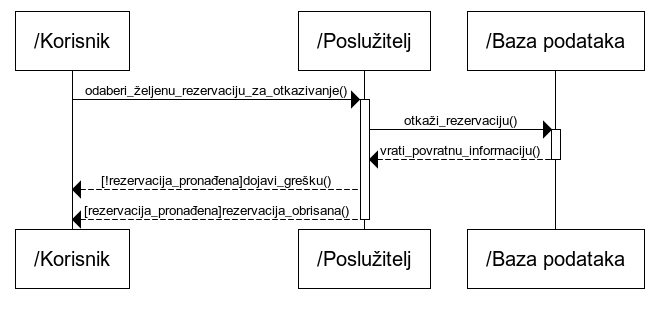
**UC10** – Stvaranje rezervacije

Administrator ili klijent odabire način i vrijeme rezervacije na određenom parkiralištu. Poslužitelj provjerava je li rezervacija ponavljajuća.  
Ako rezervacija koju traži glavni sudionik nije ponavljajuća, poslužitelj šalje zahtjev za provjerom sredstava sudionika kartičnoj kući. Ako glavni sudionik nema sredstava na računu, dojavljuje mu se greška. Ako ima dovoljno sredstava na računu, obavlja se transakcija između kartične kuće i glavnog sudionika te se ona potvrđuje poslužitelju. Poslužitelj zatim sprema podatke o rezervaciji u bazu, a baza potvrđuje uspješno spremanje. Glavnom sudioniku se potvrđuje uspješno obavljanje rezervacije.

Ako je neka rezervacija bila označena kao ponavljajuća, poslužitelj u ovisnosti o vremenu (mjesec dana) provjerava je li rezervaciju potrebno obnoviti (zbog novčane transakcije). Ako je to potrebno, poslužitelj šalje zahtjev za provjerom sredstava podnositelja zahtjeva za ponavljajućom rezervacijom. Ako ima sredstava, provodi se transakcija između kartične kuće i podnositelja zahtjeva. Ako nema sredstava, ta se rezervacija u bazi označava neaktivnom te se podnositelj zahtjeva za rezervacijom obavještava o otkazivanju rezervacije. Odgovarajući dijagram prikazan je na slici 4.2.10.

**Slika 4.2.10.** Sekvencijski dijagram za obrazac uporabe UC10 *– Stvaranje rezervacije*

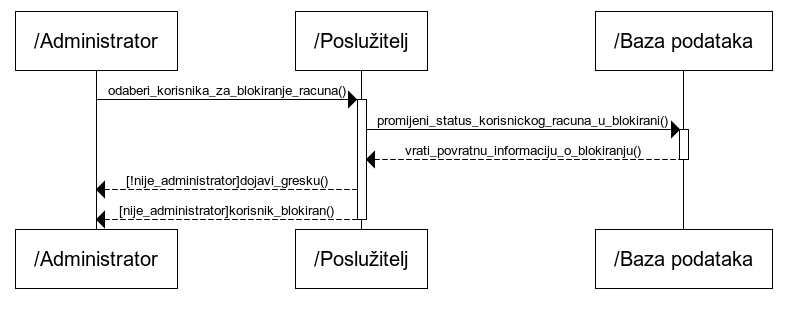
**UC11** – Otkazivanje rezervacije

Korisnik (administrator ili klijent) odabire iz svojih rezervacija onu koju želi otkazati te potvrđuje željenu akciju. Poslužitelj zatim u bazu podataka šalje željene promjene te baza podataka nakon obavljenih izmjena vraća povratnu poruku o uspješnosti izmjena. Korisniku se prikazuje poruka o uspješnom ili neuspješnom otkazivanju rezervacije. Odgovarajući dijagram prikazan je na slici 4.2.11.

**Slika 4.2.11.** Sekvencijski dijagram za obrazac uporabe UC11 – *Otkazivanje rezervacije*

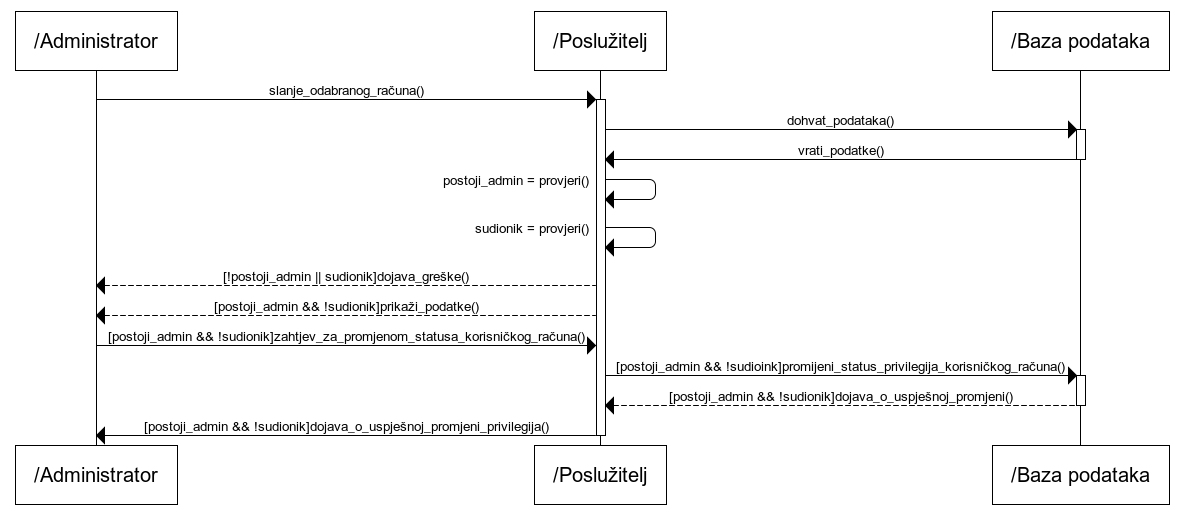
**UC12** – Blokiranje korisničkog računa

Administrator odabire kojem korisniku želi blokirati račun te zahtjev za otkazivanjem šalje poslužitelju. Poslužitelj šalje izmjene u bazu podataka koja vraća povratnu informaciju o blokiranju. Ako je odabrani korisnik jedan od administratora informacijskog sustava, administrator koji je poslao zahtjev za blokiranjem će dobiti grešku, dok će u drugim slučajevima vidjeti poruku o uspješnosti blokiranja računa. Odgovarajući dijagram prikazan je na slici 4.2.12.

****

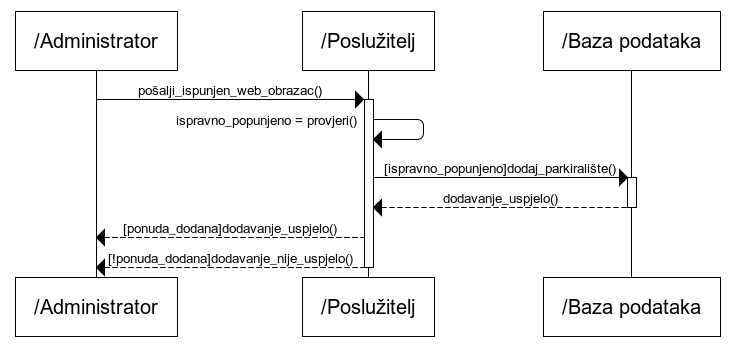
**Slika 4.2.12.** Sekvencijski dijagram za obrazac uporabe UC12 – *Blokiranje korisničkog računa*

**UC13** – Izmjena privilegija korisničkog računa

Administrator odabire račun kojemu treba promijeniti ovlasti te poslužitelj dohvaća podatke o računu iz baze podataka. Provjerava se postojanost računa i je li sudionik njegov vlasnik. Ako račun postoji, ali pripada glavnom sudioniku, javlja mu se greška. Ako račun postoji i ne pripada glavnom sudioniku, prikazuju mu se podatci o odabranom računu. Glavni sudionik odabire promjenu privilegija računa (spuštanje ili dizanje administratorskih ovlasti) te ih poslužitelj mijenja u bazi podataka u skladu sa zahtjevom i o tome obavijesti glavnog sudionika. Odgovarajući dijagram prikazan je na slici 4.2.13.

**Slika 4.2.13.** Sekvencijski dijagram za obrazac uporabe UC13 – *Izmjena privilegija korisničkog računa*

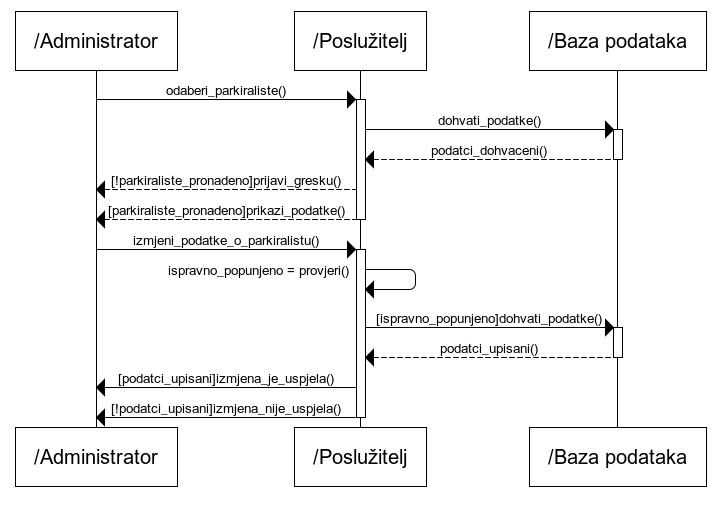
**UC14** – Dodavanje novog parkirališta

Administrator želi dodati novo parkiralište te šalje ispunjeni web-obrazac poslužitelju. Poslužitelj provjerava ispravnost podataka te obavještava administratora ako su podatci neispravni. Ako su podatci ispravni, unosi ih u bazu podataka te administratora obavještava o uspješnosti dodavanja parkirališta. Odgovarajući dijagram prikazan je na slici 4.2.14.

**Slika 4.2.14.** Sekvencijski dijagram za obrazac uporabe UC14 – *Dodavanje novog parkirališta*

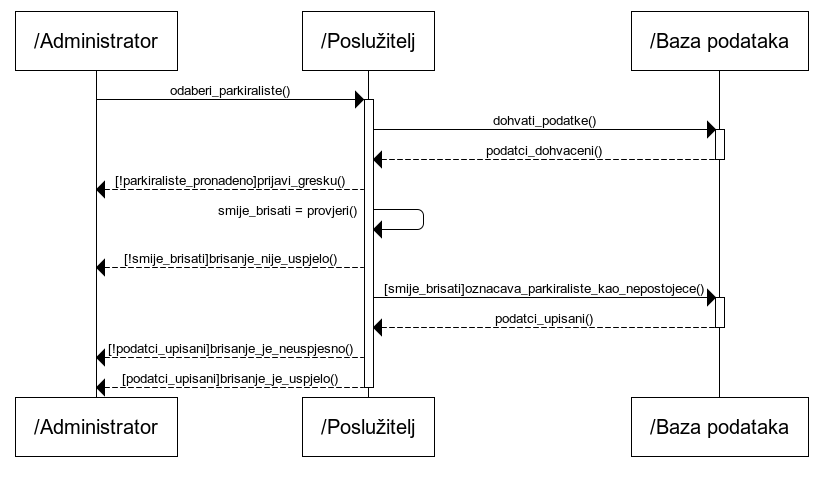
**UC15** – Izmjena podataka o parkiralištu

Administrator odabire parkiralište kojemu želi promijeniti podatke. Poslužitelj dohvaća podatke iz baze podataka te ih ili prikazuje administratoru ili šalje obavijest ako je došlo do greške. Administrator izmijenjene podatke šalje poslužitelju koji provjerava točnost podataka i unosi ih ili šalje administratoru obavijest o neispravnosti podataka. Poslužitelj obavještava administratora o usješnosti izmjene podataka. Odgovarajući dijagram prikazan je na slici 4.2.15.

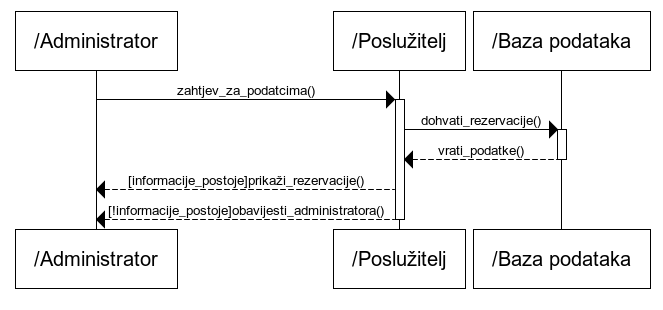
**Slika 4.2.15.** Sekvencijski dijagram za obrazac uporabe UC15 – *Izmjena podataka o parkiralištu*

**UC16** – Brisanje parkirališta

Administrator odabire parkiralište za brisanje. Poslužitelj dohvaća podatke iz baze te provjerava smije li se odabrano parkiralište obrisati. Ako nije dopušteno brisanje odabranog parkirališta, poslužitelj šalje obavijest administratoru, inače poslužitelj mijenja status parkirališta u nepostojeće te promijenjeni status sprema u bazu podataka. Poslužitelj obavještava administratora o uspješnosti izmjene statusa parkirališta. Odgovarajući dijagram prikazan je na slici 4.2.16.

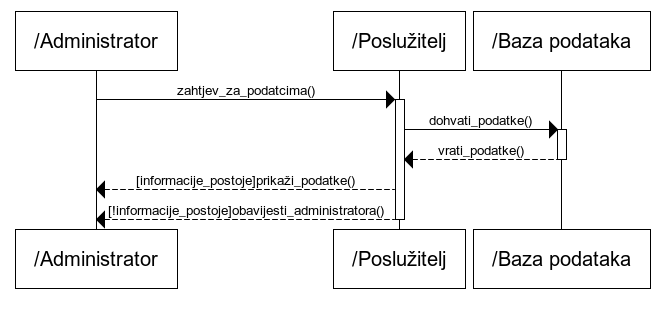
**Slika 4.2.16.** Sekvencijski dijagram za obrazac uporabe UC16 – *Brisanje parkirališta*

**UC17** – Pregled informacija o svim rezervacijama

Administrator šalje poslužitelju odgovarajući zahtjev za dohvat informacija o rezervacijama. Poslužitelj dohvaća informacije o rezervacijama iz baze podataka i prikazuje ih administratoru ako postoje rezervacije za odgovarajući upit. Ako rezervacije ne postoje, tada se administratora obavještava odgovarajućom porukom o neuspješnosti. Odgovarajući dijagram prikazan je na slici 4.2.17.

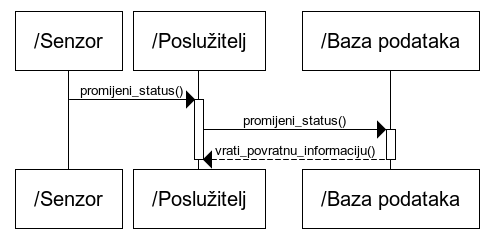
**Slika 4.2.17.** Sekvencijski dijagram za obrazac uporabe UC17 – *Pregled informacija o svim rezervacijama*

**UC18** – Pregled informacija o svim korisnicima

Administrator šalje poslužitelju odgovarajući zahtjev za dohvat informacija o korisniku. Poslužitelj dohvaća informacije o korisniku iz baze podataka i prikazuje ih administratoru ako postoje informacije za odgovarajući upit. Ako informacije ne postoje, tada se administratora obavještava odgovarajućom porukom o neuspješnosti. Odgovarajući dijagram prikazan je na slici 4.2.18.

**Slika 4.2.18.** Sekvencijski dijagram za obrazac uporabe UC18 – *Pregled informacija o svim korisnicima*

**UC19** – Promjena statusa parkirališnog mjesta

Nakon što se dogodila promjena statusa parkirališnog mjesta, senzor šalje poslužitelju zahtjev za promjenom statusa svog parkirališnog mjesta u bazi podataka. Poslužitelj na temelju vrijednosti parkingSpotId-a mijenja vrijednost atributa *sensor* u relaciji *parkingSpot* u bazi podataka. Baza podataka vraća poslužitelju povratnu informaciju o uspješnosti izmjena. Odgovarajući dijagram prikazan je na slici 4.2.19.

**Slika 4.2.19.** Sekvencijski dijagram za obrazac uporabe UC19 – *Promjena statusa parkirališnog mjesta*

1. Ostali zahtjevi

* Aplikacija treba omogućiti prikaz hrvatskih dijakritičkih znakova.
* Podatci aplikacije trebaju biti lako dohvatljivi (cjenik, itd.) te sama aplikacija mora biti jednostavna za korištenje.
* Sustav mora omogućiti istovremeni rad više korisnika.
* Parkirališta moraju inicijalno biti udaljena više od 50 metara.
* Vrijeme odziva aplikacije ne bi trebalo biti veće od 4 sekunde.
* Informacijski sustav potrebno je realizirati kao web-aplikaciju pa smo se odlučili za objektno-orijentirani programski jezik PHP.
* Sustav ne bi trebao omogućiti brisanje rezervacija iz baze podataka. Sve rezervacije se čuvaju te je za svaku naveden njezin status (aktivna ili neaktivna).
* Nadogradnja sustava nije moguća bez izvornog koda.
* Nepravilne i nepredviđene korisnikove akcije ni u jednom trenutku ne smiju narušiti bilo koju od funkcionalnosti sustava.
* Nadogradnja sustava i njegovih funkcionalnosti ne smije narušiti funkcionalnost sustava ili bilo koje postojeće funkcionalnosti.

1. Arhitektura i dizajn sustava
   1. Svrha, opći prioriteti i skica sustava
      1. Web-aplikacija

Izbor arhitekture sustava važan je korak u oblikovanju sustava. Bitno je razmatranje različitih stilova te njihovih prednosti i nedostataka u odnosu na zahtjeve određenog sustava kako bi se odabrala arhitektura koja će najbolje zadovoljiti tražene sljedeće zahtjeve: dovoljna fleksibilnost i moguća nadogradnja sustava.

Za arhitekturu je ovog sustava izabrana objektno-usmjerena arhitektura zbog stabilnosti, pouzdanosti, pojednostavljenog oblikovanja i mogućnosti jednostavnijeg održavanja sustava. Sustav je podijeljen na razrede što nam omogućava višu razinu apstrakcije koja je bliža ljudskom načinu razmišljanja o problemu.

Naše namjere da sustav bude što jednostavniji za korištenje svakodnevnom korisniku i da su njegove funkcionalnosti dostupne na internetu jasno nam nalažu da se odlučimo za arhitekturu web aplikacije. Alternativa koja bi se mogla razmotriti jest arhitektura klijent-poslužitelj, no takav bi pristup zahtijevao da korisnik koristi posebnu programsku podršku, dok primjenom arhitekture web-aplikacije može koristiti web-preglednike koji su danas dio osnovne korisničke programske opreme.

Ta arhitektura nam također omogućava centralizirane promjene sustava čime se ponovno izbjegava izravan utjecaj na korisnika. Ključni dio arhitekture je web-poslužitelj pa aplikacija mora zadovoljavati njihove današnje standarde.

Arhitektura našeg sustava sastoji se od sljedećih podsustava:

* Web-poslužitelj
* Web-aplikacija
* Baza podataka

**Web-poslužitelj**

Web-poslužitelj je neizbježan za izvođenje bilo kakve web-aplikacije. Danas već postoji veliki broj kvalitetnih i široko korištenih web-poslužitelja pa nemamo potrebe za razvijanjem zasebnog, već će aplikacija biti prilagođena za izvođenje na postojećim poslužiteljima poput Apacheovog6.

**Web-aplikacija**

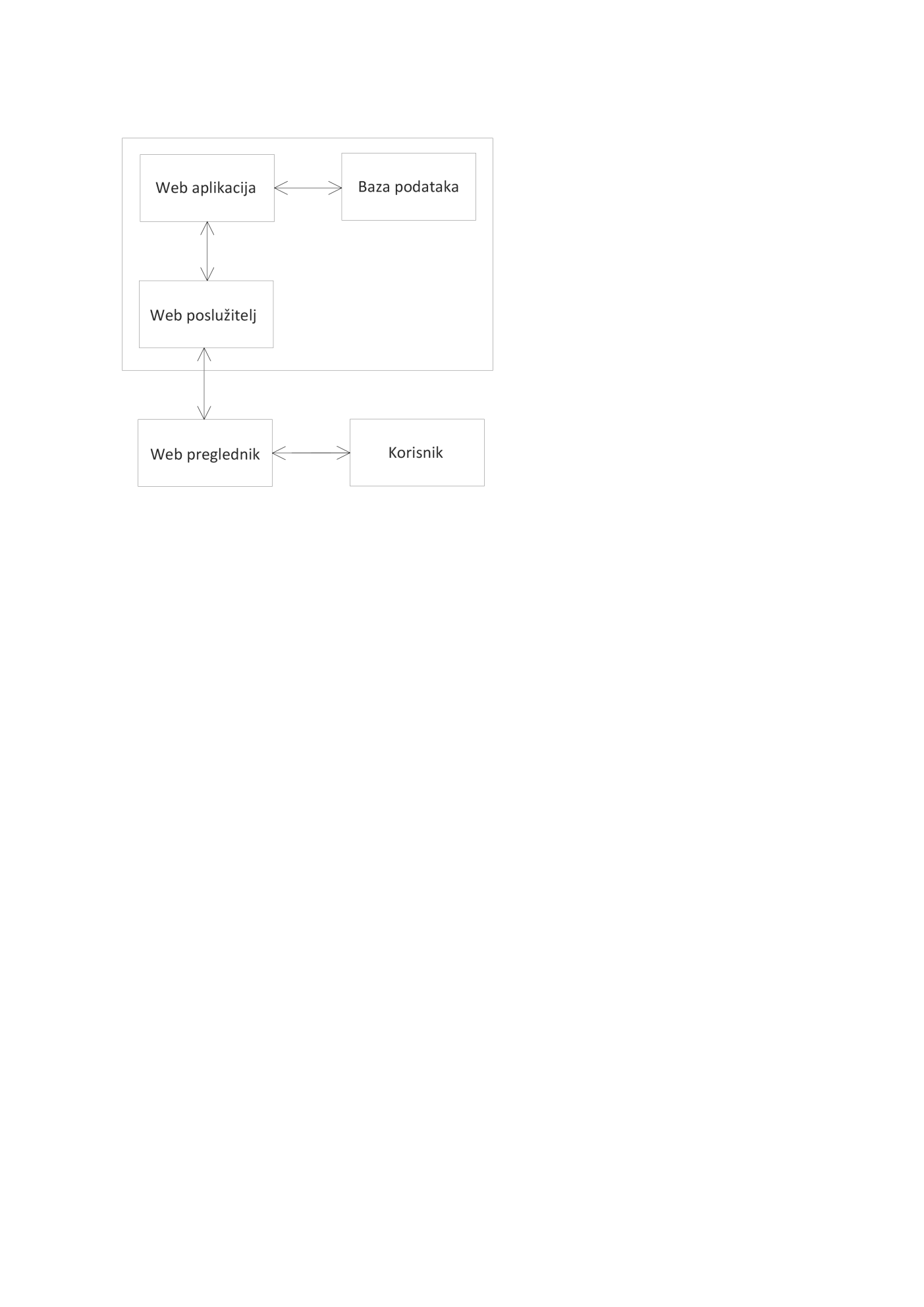
Unutar same web-aplikacije nalazi se logika koja korisničke zahtjeve pretvara u operacije sustava. Ona je prozor u bazu podataka koja sama za sebe ne zna apstraktne operacije koje korisnik može zahtijevati te mora vršiti provjere ispravnosti podataka i razlagati složene operacije u konkretne upite bazi podataka. Ako nije drugačije rečeno u konfiguracijskim datotekama, aplikacija se treba sama brinuti o bazi podataka koju koristi. Ona ju stvara, popunjava inicijalnim vrijednostima i radi na njoj izmjene u skladu sa svojim funkcionalnostima.

Web-aplikacija se također brine o prezentaciji podataka korisniku i za to se oslanja na funkcionalnosti web-preglednika kojega korisnik koristi. Dok se sama aplikacija brine o obradi podataka, njihovu reprezentaciju velikim dijelom prepušta dodatnim resursima u obliku kostura HTML-dokumenata koji će se ispuniti tim podatcima. Time je moguće raditi izmjene rada aplikacije na razini reprezentacije korisniku bez prekida rada.

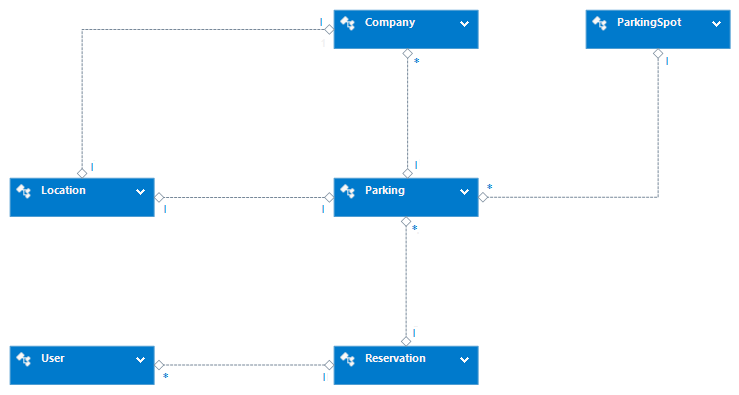
**Baza podataka**

Za potrebe naše web-aplikacije koristit ćemo relacijsku bazu podataka koja je u današnje vrijeme najkorišteniji model baze podataka jer je relacijska struktura u velikoj mjeri podudarna s predodžbom događaja stvarnog svijeta i podataka koje ti događaji generiraju. Relacijska baza podataka sastoji se od skupa relacija odnosno tablica pri čemu je svaka relacija opisana relacijskom shemom koja obuhvaća naziv relacije i skup pripadajućih atributa. Sve relacije su svedene na 3. normalnu formu kako u bazi podataka ne bi došlo do redundancije odnosno zalihosti.

Za izradu naše baze podataka koristit ćemo MySQL7, a kako će ona izgledati prikazano je na slici ER-modela u kojem možemo vidjeti u kakvim su vezama pojedini entiteti baze podataka. Sve relacije koje su generirane iz ER-modela prikazane su nakon ER-modela s odgovarajućim opisima atributa i primarnim ključevima.



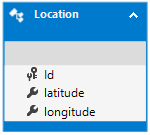
**Slika 6.1.1.** Skica sustava

* + 1. Baza podataka

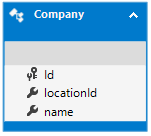
ER-model baze podataka

**Slika 6.1.2.** ER-model baze podataka

Opisi relacija generiranih iz ER-modela baze podataka

Location

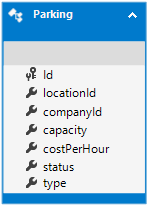
|  |  |  |
| --- | --- | --- |
| Ime atributa | Tip atributa | Opis |
| Id | integer | primarni ključ |
| latitude | double | geografska širina |
| longitude | double | geografska dužina |

Company

|  |  |  |
| --- | --- | --- |
| Ime atributa | Tip atributa | Opis |
| Id | integer | primarni ključ |
| locationId | Integer | lokacija kompanije |
| name | String | ime kompanije |

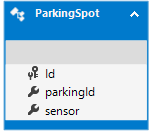
Parking

|  |  |  |
| --- | --- | --- |
| Ime atributa | Tip atributa | Opis |
| Id | integer | primarni ključ |
| locationId | integer | lokacija parkirališta |
| companyId | integer | parkiralište pripada kompaniji |
| capacity | integer | broj parkirališnih mjesta |
| costPerHour | integer | vrijednost koliko košta sat parkiranja |
| status | integer | trenutni status parkirališta - otvoreno, zatvara se, zatvoreno |
| type | integer | tip parkirališta - garaža ili otvoreno parkiralište |
| companyId | integer | parkiralište pripada kompaniji |
| capacity | integer | broj parkirališnih mjesta |
| costPerHour | integer | cijena po satu parkiranja |
| status | integer | trenutni status parkirališta - otvoreno, zatvara se, zatvoreno |

**

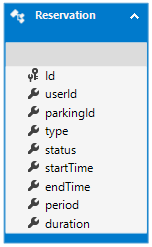
ParkingSpot

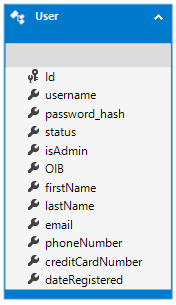
|  |  |  |
| --- | --- | --- |
| Ime atributa | Tip atributa | Opis |
| Id | integer | primarni ključ |
| parkingId | integer | parkirališno mjesto pripada jednom parkiralištu |
| sensor | boolean | istina = zauzeto, laž = slobodno |



Reservation

|  |  |  |
| --- | --- | --- |
| Ime atributa | Tip atributa | Opis |
| Id | integer | primarni ključ |
| userId | integer | korisnik koji je napravio rezervaciju |
| parkingId | integer | parking na kojem vrijedi rezervacija |
| type | integer | tip rezervacije - jednokratna, ponavljajuća, trajna |
| status | integer | prošla, otkazana, ... |
| startTime | timestamp | vrijeme kad registracija počinje |
| endTime | timestamp | vrijeme kad rezervacija završava |
| duration | integer | trajanje rezervacije u satima |
| period | timestamp | period ponavljanja periodičke rezervacije |



User

|  |  |  |
| --- | --- | --- |
| Ime atributa | Tip atributa | Opis |
| Id | integer | primarni ključ |
| username | string | korisničko ime |
| password | string | lozinka |
| status | integer | aktivan ili blokiran |
| isAdmin | boolean | povlašteni korisnik? |
| OIB | integer | - |
| firstName | string | ime korisnika |
| lastName | string | prezime korisnika |
| email | string | email adresa |
| contactPhoneNumber | integer | broj kontakt telefona |
| creditCardNumber | integer | broj kreditne kartice |
| dateCreated | datetime | datum registracije korisnika |

* 1. Dijagram razreda s opisom

Opis dijagrama razreda

Dijagram razreda prikazuju razrede, atribute i operacije razreda, njihova svojstva i ograničenja, sučelja, pridruživanja, vlastite tipove podataka, enumeracije, pakete i komentare.

Yii je danas jedna od najučinkovitijih i najčešće korištenih radnih okvira za izradu web aplikacija pa smo se i mi odlučili za taj radni okvir. Yii je objektno orjentiran te implementira model-view-controller (MVC) uzorak. MVC je obrazac softverske arhitekture za stvaranje web aplikacija. Koristi se u programskom inženjerstvu za odvajanje pojedinih dijelova aplikacije u komponente ovisno o njihovoj namjeni. MVC dijeli aplikaciju na tri glavne komponente: model, view i controller (model, pogled i kontroler).

* **Model** – komponenta koja predstavlja upravljanje podatcima
* **View** - komponenta koja upravlja prikazom podataka
* **Controller** – komponenta koja izvršava interakciju s korisnikom, radi s modelima i odabire koji view će se prikazati

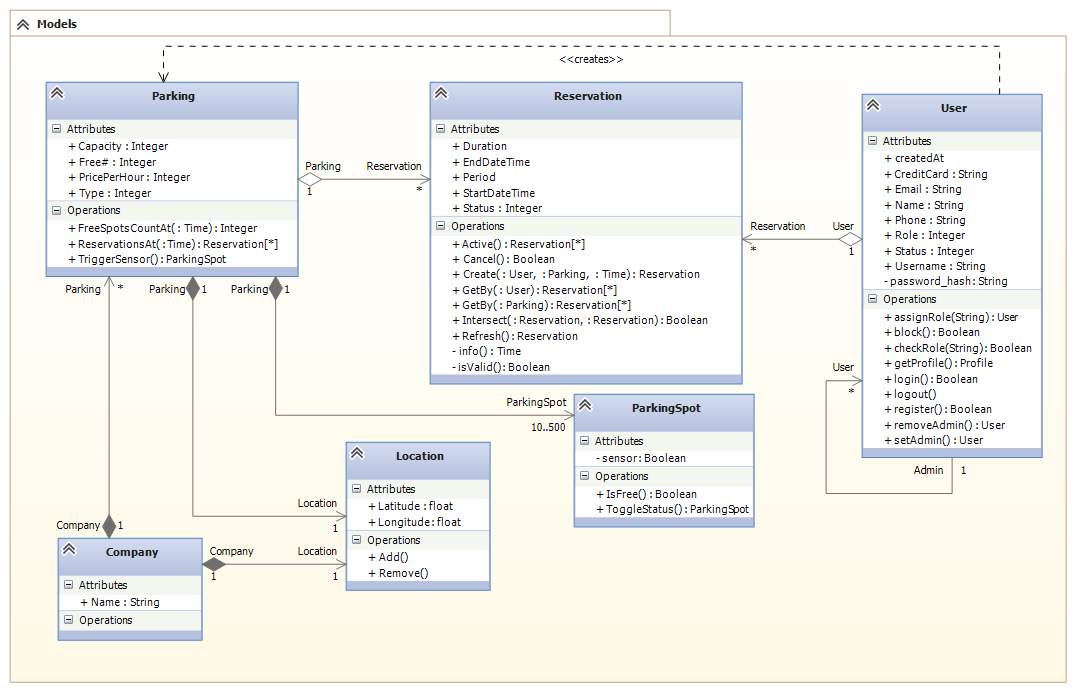
Osnovna prednost MVC arhitekture je razdvajanje projekta na smislene i odvojene cjeline, što predstavlja veliku prednost pri izradi velikih projekata na kojima radi više osoba. Iz toga proizlazi i druga prednost koja se odnosi na izmjenu, nadogradnju i budući razvoj.

**Slika 6.2.1.** – Model – View – Controller u Yii aplikaciji

U nastavku je svaki sloj MVC uzorka pobliže opisan te su navedeni i opisani  
razredi svakog sloja.

* + 1. Models

**Slika 6.2.2.** Models

Model je je dio MVC arhitekture u kojemu se nalazi cjelokupna podatkovna logika**.** On sadrži informacije o objektima (klasama, svojstvima, entitetima u bazi podataka i vezama). Modeli se prosljeđuju pogledima (end. View) ili bazi (kako bi mijenjali podatke u bazi).

Parking

**Atributi**: kapacitet, tip, broj trenutno slobodnih mjesta i cijena parkiranja po satu.

**Metode**:

* odredi broj slobodnih mjesta u zadanom vremenskom intervalu (u budućnosti)
* pronađi sve rezervacije aktivne kroz zadani vremenski interval
* pokreni slučajnim odabirom jedan senzor na parkiralištu

**Odnosi**:

Company: brisanjem jedne kompanije briše se više njenih parkirališta

Location: jedno parkiralište se nalazi na jednoj lokaciji

ParkingSpot: brisanjem jednog parkirališta briše se više parkirališnih mjesta

Reservation: brisanjem jednog parkinga ostaje sačuvano više rezervacija

Reservation

**Atributi**: status rezervacije - aktivna, poništena, …, vrijeme početka i kraja, period i trajanje

**Metode**:

* dohvati listu svih aktivnih rezervacija
* kreiraj novu rezervaciju korisnika na odabranom parkiralištu u zadanom vremenskom intervalu
* poništi rezervaciju
* pronađi sve rezervacije od danog korisnika
* pronađi sve rezervacije na danom parkiralištu
* vrati objekt vremenskog intervala rezervacije
* provjeri da li je rezervacija u redu
* osvježi periodičnu (uključujući trajnu) rezervaciju
* provjeri da li se dvije zadane rezervacije poklapaju vremenski, a da su na istom parkingu

**Odnosi**:

Parking: moguće je više rezervacija obaviti za pojedini parking. brisanjem parkinga ostaju rezervacije.

User: pojedini korisnik može imati više rezervacija. brisanjem korisnika ostaju rezervacije.

Company

**Atributi**: ime kompanije

**Metode**: -

**Odnosi**:

Parking: jedna kompanija ima više parkinga koji se svi brišu skupa sa kompanijom.

Location: jedna kompanija ima jednu lokaciju, koja se briše u slučaju brisanja kompanije.

Location

**Atributi**: geografska dužina i širina

**Metode**:

* dodaj lokaciju u bazu podataka
* ukloni lokaciju iz baze podataka

**Odnosi**:

Company: jedna kompanija ima jednu lokaciju koja se briše u slučaju brisanja kompanije.

Parking: jedan parking ima jednu lokaciju, koja se briše u slučaju brisanja parkinga.

ParkingSpot

**Atributi**: vrijednost senzora

**Metode**:

* provjeri da li je parkirališno mjesto slobodno
* promijeni status parkirališnog mjesta, ako je bilo slobodno, proglasi ga zauzetim, i obrnuto

**Odnosi**:

Parking: jedan parking ima 10 do 500 parkirališnih mjesta koji se brišu brisanjem parkirališta.

User

**Atributi**: osobni podatci koji korisnik treba unijeti pri registraciji (ime, prezime, e-mail, kontakt telefon, broj kreditne kartice), korisničko ime i lozinka; pamtimo i njegovu ulogu (ROLE\_CLIENT, ROLE\_ADMIN), njegov status (STATUS\_ACTIVE, STATUS\_INACTIVE, STATUS\_BLOCKED), kao i datum registracije

**Metode**:

* pridruži ulogu korisniku
* blokiraj korisnika
* provjeri da li korisnik ima traženu ulogu
* vrati profil korisnika, tj. njegove osobne podatke
* prijavi korisnika u sustav
* odjavi korisnika iz sustava
* kreiraj korisnika
* pomoćna metoda – proglasi korisnika klijentom
* pomoćna metoda – proglasi korisnika administratorom

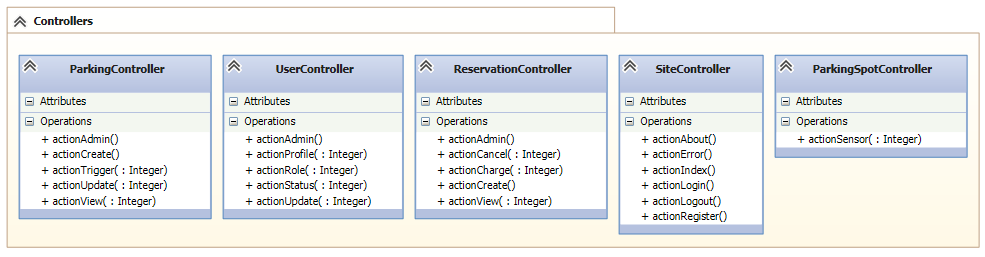
**Odnosi**:

Reservation: jedan korisnik može imati više rezervacija

User: refleksivni odnos, administrator kontrolira ostale korisnike

* + 1. Controllers

Ovaj sloj upravlja interakcijom s korisnikom, na temelju korisničkih akcija šalje ili dohvaća podatke iz modela, te prikazuje odgovarajuć i pogled (View). Sastoji se od sljedećih razreda.



**Slika 6.2.3.** Controllers

ParkingController

**Metode**:

* prikazuje sučelje za administriranje parkirališta, obavlja potrebne provjere
* prikazuje web-obrazac za kreiranje parkirališta zajedno s mapom za odabir lokacije, sprema rezultat POST zahtjeva, obavlja potrebne provjere
* ne prikazuje ništa, slučajnim odabirom parkirališnog mjesta mijenja vrijednost senzora
* prikazuje web-obrazac za izmjenu parkirališta zajedno s mapom za premještanje lokacije, sprema rezultat POST zahtjeva, obavlja potrebne provjere
* prikazuje prikaz jednog parkirališta

UserController

**Metode**:

* prikazuje sučelje za administriranje korisnika, obavlja potrebne provjere
* prikazuje korisnički profil (prikaz osobnih podataka korisnika)
* pomoćna akcija za promjenu korisničke uloge preko AJAX-a
* pomoćna akcija za promjenu statusa korisnika preko AJAX-a
* prikazuje web-obrazac za izmjenu korisničkih podataka

ReservationController

**Metode**:

* prikazuje sučelje za administriranje rezervacija, obavlja potrebne provjere
* otkazuje rezervaciju
* naplaćuje rezervaciju
* prikazuje web-obrazac za stvaranje nove rezervacije, potvrđuje podatke nakon POST zahtjeva i sprema ih
* renderira pregled rezervacije

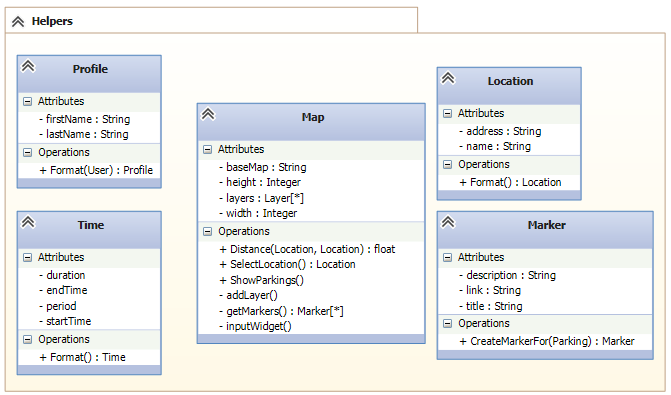
SiteController

**Metode**:

* prikazuje stranicu s kratkim opisom projekta i članova tima
* prikazuje *error* stranicu
* prikazuje početnu stranicu s velikom kartom svih parkirališta i cjenikom
* prikazuje obrazac za prijavu korisnika
* obavlja odjavu trenutno prijavljenog korisnika
* prikazuje registracijski obrazac

ParkingSpotController

**Metode**:

* mijenja vrijednost senzora
  + 1. Helpers

**Slika 6.2.4.** Helpers

Profile

Formatira osobne podatke korisnika u format za prikaz.

Time

Formatira vremensku komponentu rezervacije u format pogodan za prikaz.

Predstavlja temelj analize vremenskih preklapanja rezervacija.

Map

Wrapper oko Google Maps ekstenzije za Yii2 framework koji je u potpunosti usko prilagođen potrebama ove aplikacije.

Location

Formatira lokaciju, dodaje adresu i ime ako postoje.

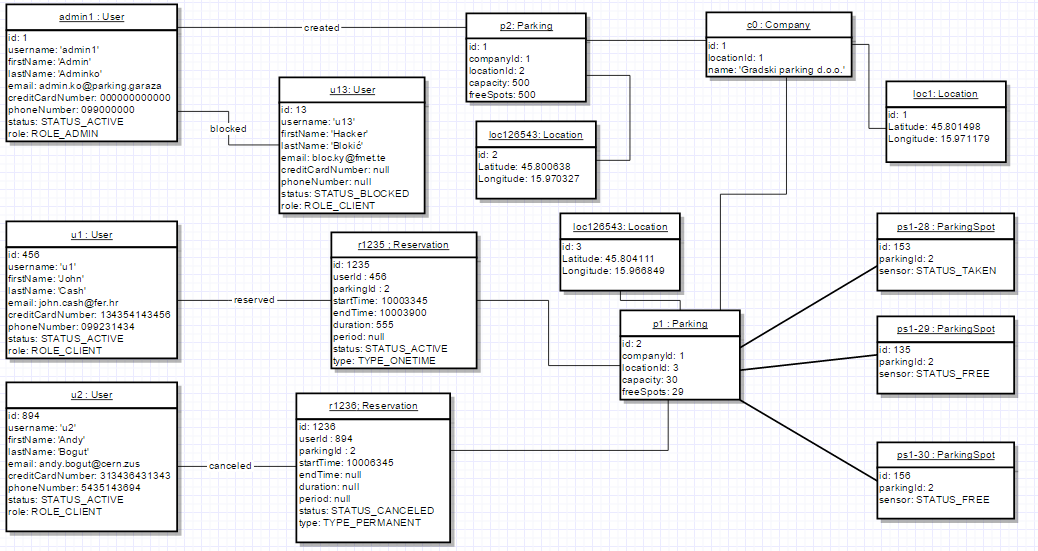
Marker

Sadrži podatke potrebne za efikasno generiranje prikaza parkirališta na mapi.

* 1. Dijagram objekata

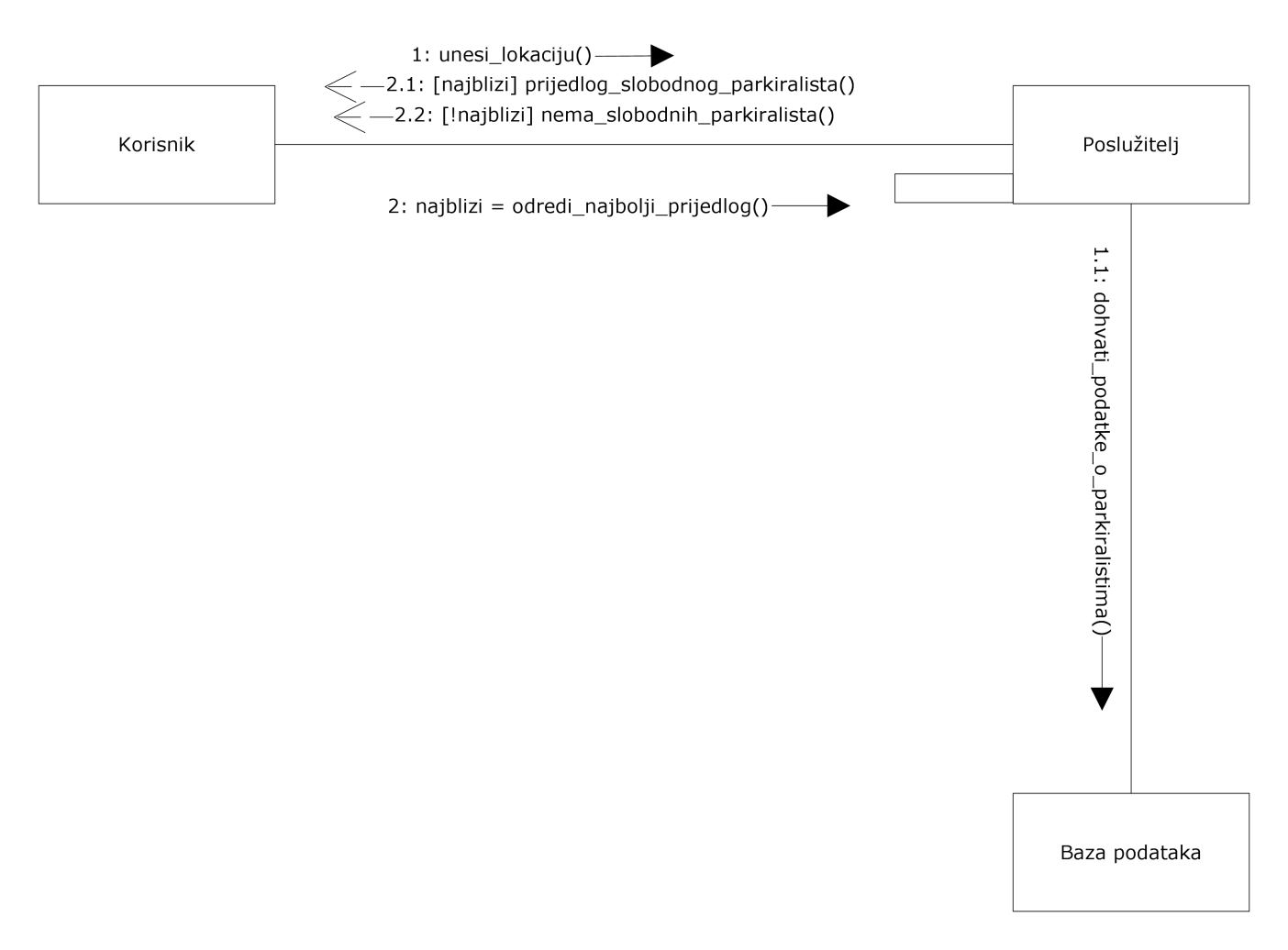
Opis dijagrama objekata

Na slici 6.3.1. prikazan je dijagram objekata u trenutku kad je administrator stvorio parkiralište s 500 mjesta koje je pridruženo tvrtci Gradski Parking d.o.o. i nakon blokiranja klijenta pod imenom Hacker Blokić. U istom je trenutku klijent John Cash na odabranom parkiralištu jednokratno rezervirao parkirališno mjesto, dok je klijent Andy Bogut na istom parkiralištu otkazao svoju trajnu rezervaciju. Njihovim je rezervacijama u tom trenutku promijenjen broj slobodnih mjesta. Poznata je i lokacija svakog parkirališta i tvrtke.

**Slika 6.3.1.** Dijagram objekata

* 1. Ostali UML dijagrami
     1. Komunikacijski dijagrami

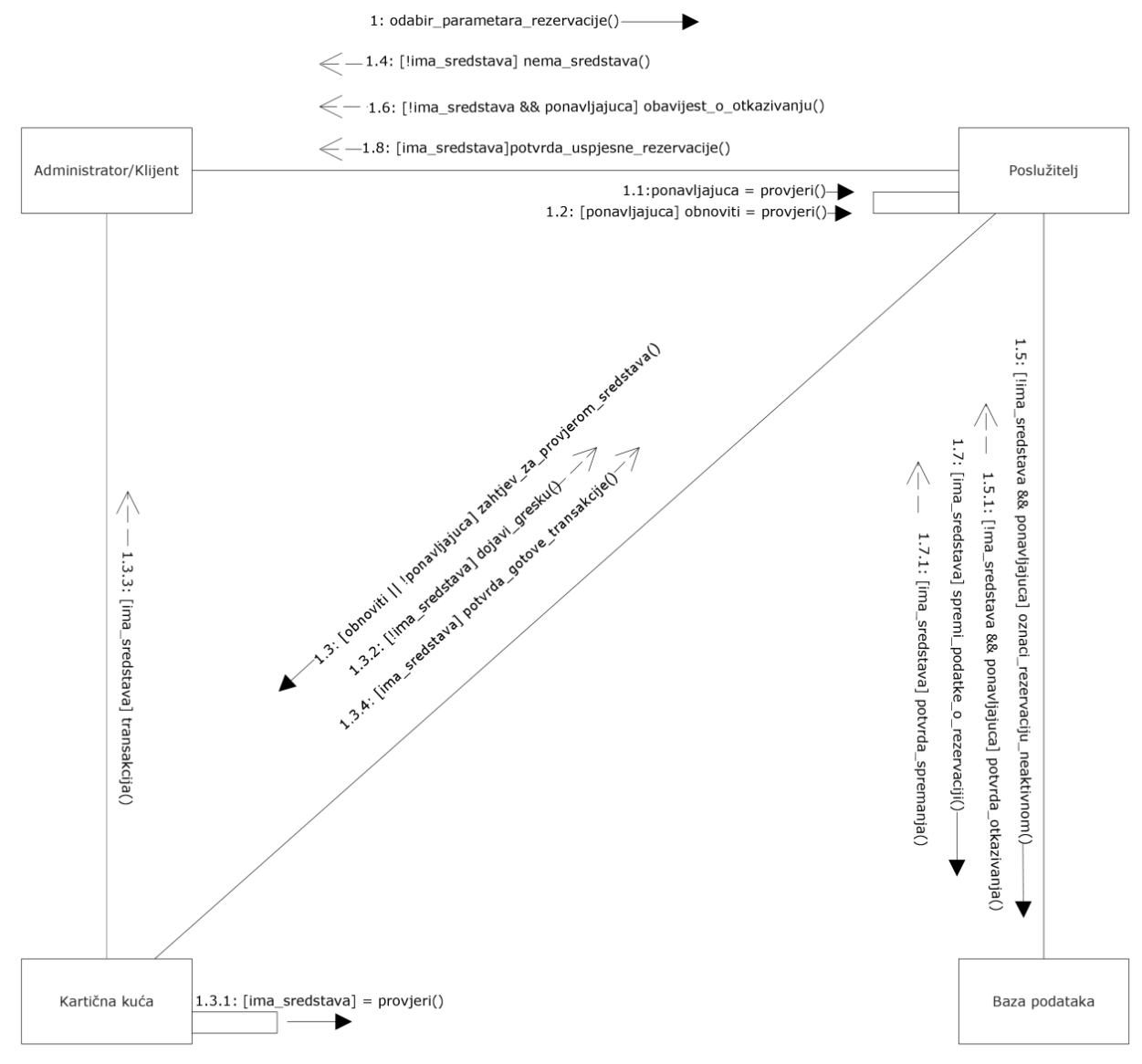
**UC4** – Dobivanje prijedloga

Korisnik unosi svoju lokaciju putem weba i šalje je poslužitelju. U trenutku primanja lokacije poslužitelj iz baze podataka dohvaća podatke o parkiralištima. Iz tih podataka određuje najbolji prijedlog te broj slobodnih mjesta. Ako ima, poslužitelj korisniku predlaže ranije određeni parking te ako nema, javlja mu da nema slobodnih parkirališta.

**Slika 6.4.1.** Komunikacijski dijagram za obrazac uporabe UC4 – *Dobivanje prijedloga*

**UC10** – Stvaranje rezervacije

Administrator ili klijent odabire parametre rezervacije na određenom parkiralištu. Poslužitelj potom provjerava je li rezervacija ponavljajuća i potrebu za obnavljanjem. Ako rezervacija nije ponavljajuća ili je treba obnoviti, poslužitelj šalje kartičnoj kući zahtjev za provjerom sredstava. Kartična kuća provjerava ima li sredstava na računu korisnika te provodi transakciju i šalje poslužitelju potvrdu o obavljenoj transakciji ovisno o sredstvima na računu korisnika. Potom poslužitelj, ako je riječ o ponavljajućoj rezervaciji te na računu nema sredstava, u bazi podataka označuje rezervaciju neaktivnom te javlja korisniku da je rezervacija otkazana. Alternativno, ako je od kartične kuće dobio potvrdu o obavljenoj transakciji, poslužitelj sprema u bazu podataka informacije o stvorenoj/produženoj rezervaciji te nakon potvrde o uspješnom spremanju javlja korisniku da je rezervacija uspješno stvorena ili produžena.

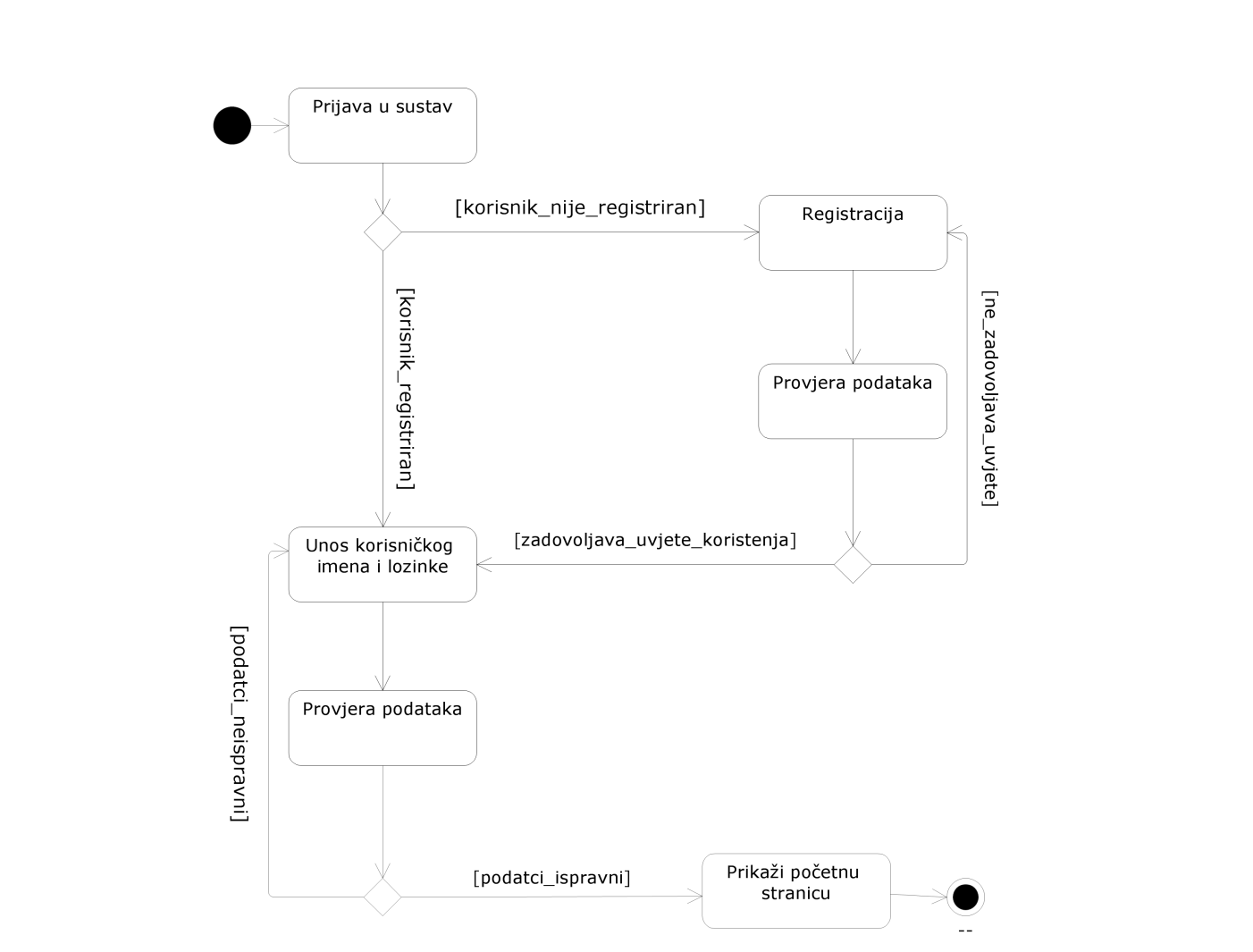
****

**Slika 6.4.2.** Komunikacijski dijagram za obrazac uporabe UC10 – *Stvaranje rezervacije*

* + 1. Dijagrami stanja

**UC6** – Prijava korisnika u sustav

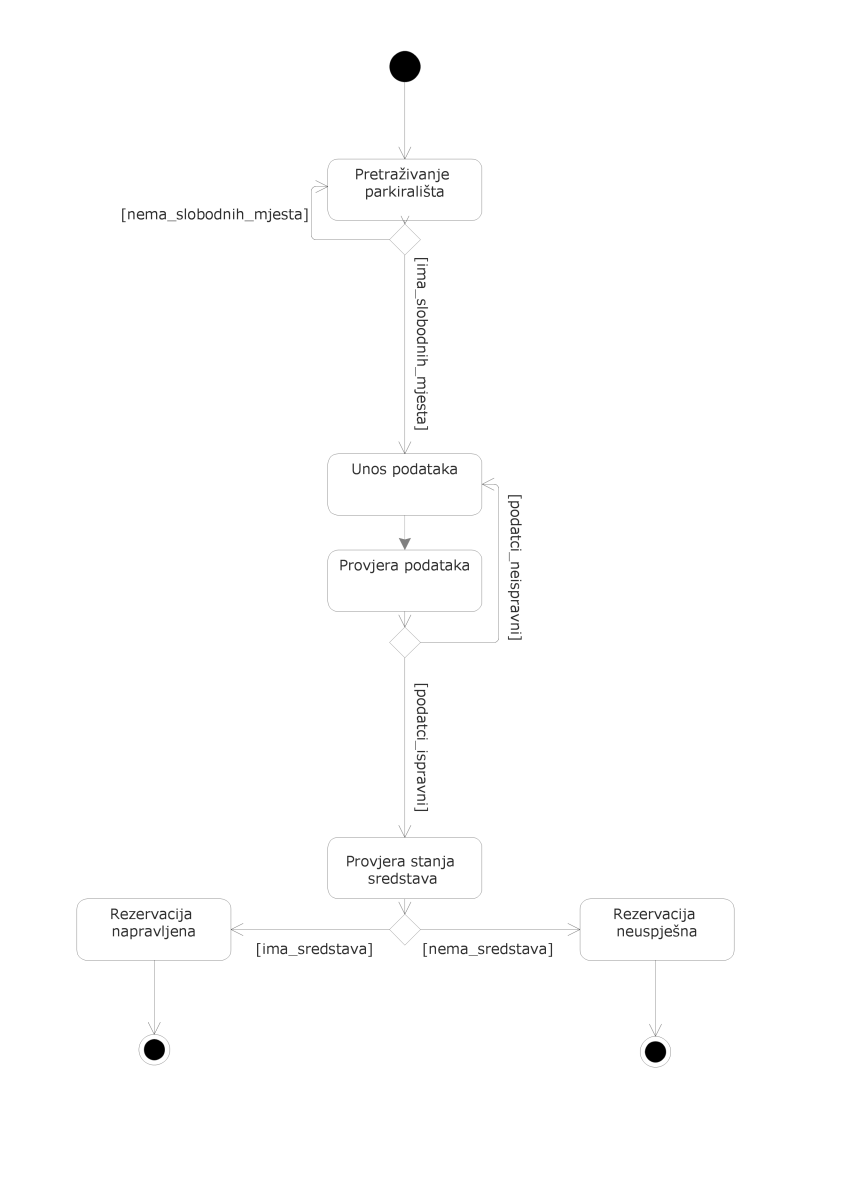
Dijagram sa slike ispod ovog ulomka prikazuje stanja prilikom prijave klijenta u sustav. Ukoliko korisnik nije registriran, treba obaviti registraciju. Nakon upisanih podataka, provjerava se jesu li zadovoljeni uvjeti registracije. Ukoliko uneseni podatci nisu ispravni, klijent se vraća u prethodno stanje te ponavlja registraciju. Ukoliko su u novom pokušaju uvjeti zadovoljeni, korisnik prelazi u stanje *Unosa korisničkog imena i lozinke* gdje unosi podatke. Uneseni podatci se šalju na poslužitelj te se njihova ispravnost provjerava u stanju *Provjera podataka*. U slučaju uspješne prijave korisnik prelazi u stanje *Prikaži početnu stranicu,* a ako prijava nije uspjela, aplikacija vraća korisnika u stanje ponovnog unosa korisničkog imena i lozinke.



**Slika 6.4.3.** Dijagram stanja za za obrazac uporabe UC6 – *Prijava korisnika u sustav*

**UC10** – Stvaranje rezervacije

Dijagram sa slike ispod ovog ulomka prikazuje stanja prilikom rezervacije parkirališnog mjesta. U trenutku završene prijave u sustav klijent odabire pretraživanje parkirališta. Ako sustav ne nađe ni jedno slobodno mjesto, vraća klijenta u prethodno stanje te nudi ponovno pretraživanje. Ako pak postoje slobodna parkirališna mjesta, tada klijent odabire željeno parkiralište te način i vrijeme rezervacije. Sama rezervacija može biti jednokratna ili ponavljajuća kroz određeni period. Ako podatci nisu ispravni, sustav se vraća u stanje unosa podataka, inače prelazimo u stanje *Provjere stanja sredstava* gdje kartična kuća provjerava jesu li sredstva na računu dovoljna za obavljanje periodične rezervacije. Ako klijent ima dovoljno sredstava na kreditnoj kartici, rezervacija je napravljena te sustav prelazi u konačno stanje, dok je u slučaju nedostatka sredstava rezervacija neuspjela te sustav također prelazi u konačno stanje.

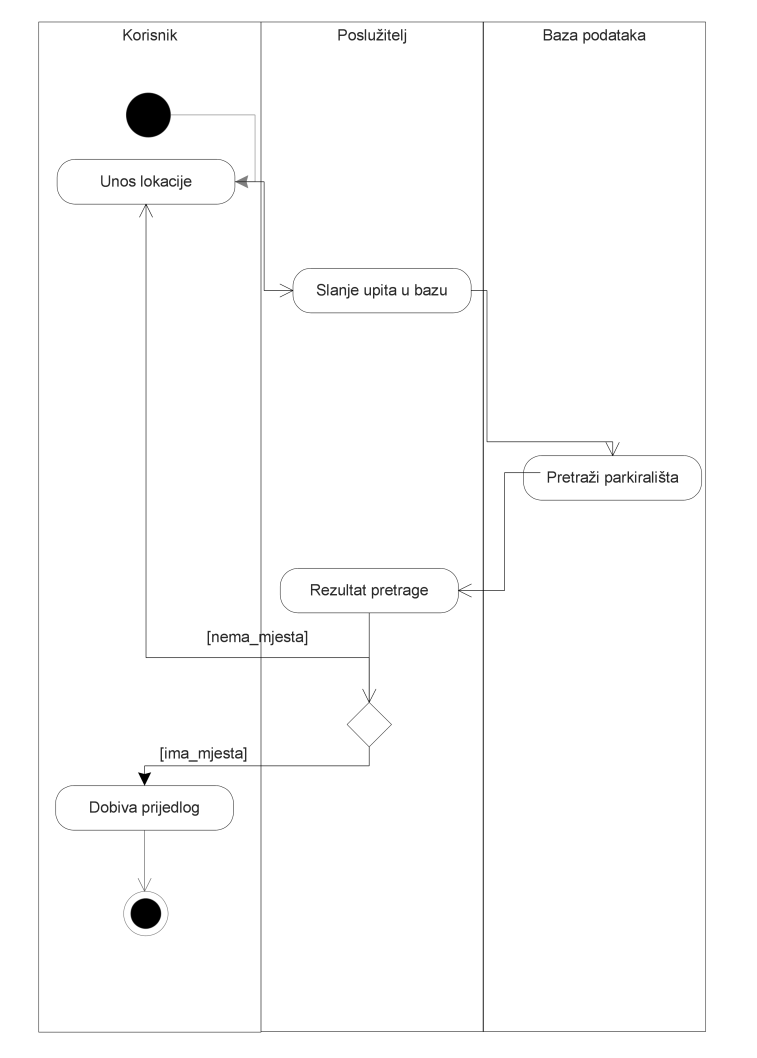


**Slika 6.4.4.** Dijagram stanja za za obrazac uporabe UC10 – *Stvaranje rezervacije*

* + 1. Dijagrami aktivnosti

**UC4** – Dobivanje prijedloga

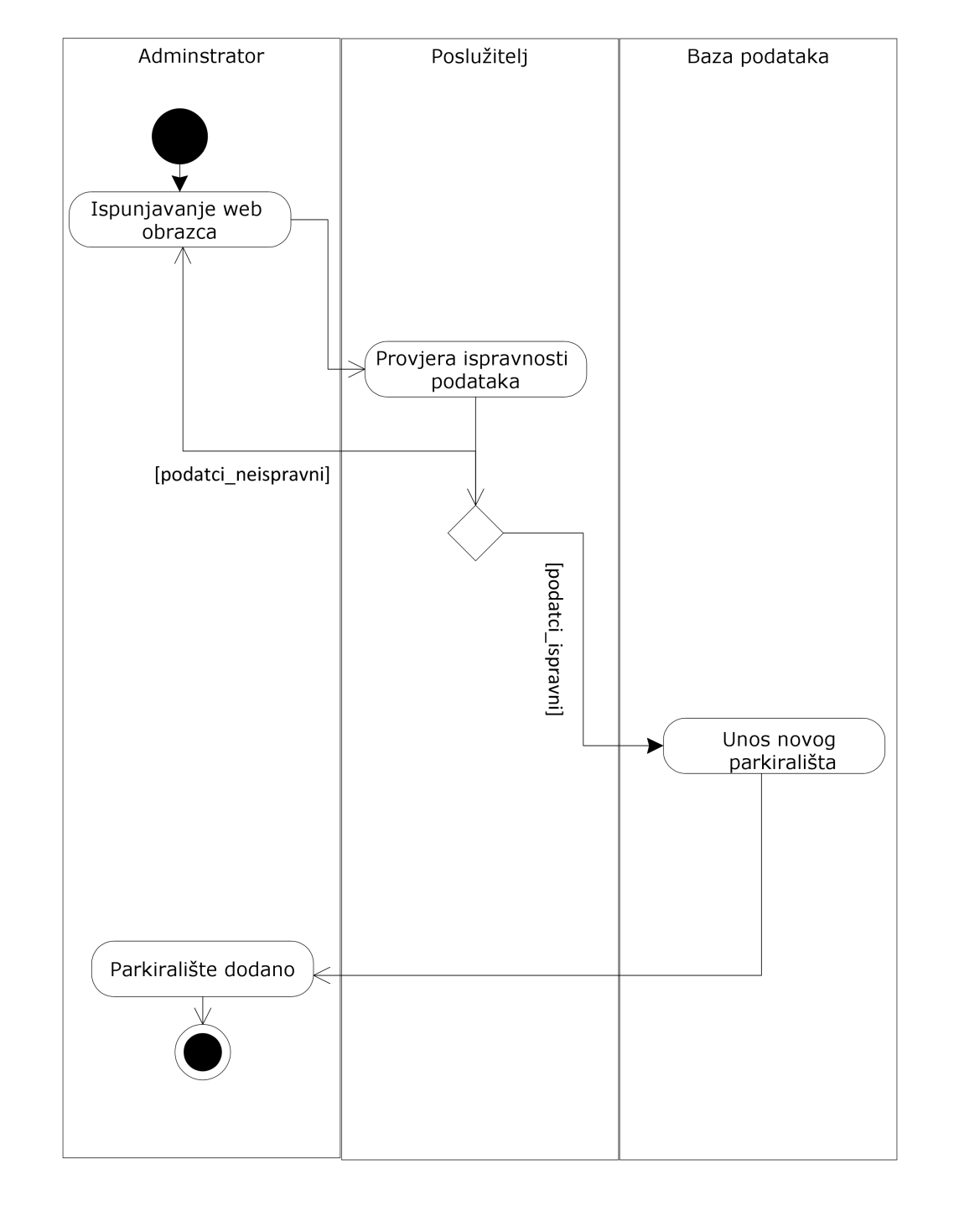
Dijagram aktivnosti na slici ispod ovog ulomka prikazuje dobivanje lokacije najbližeg parkirališta sa slobodnim parkirališnim mjestom. U početku korisnik unosi svoju lokaciju te potom poslužitelj šalje zahtjev za dohvatom podataka o parkiralištima u bazu podataka. Poslužitelj obrađuje podatke te na temelju unosa lokacije donosi odluku o rezultatu pretrage. Moguća su dva ishoda pretrage koja ovise o postojanju parkirališnih mjesta. U slučaju da nema ni jednog slobodnog parkiralištnog mjesta, korisnik može ponovno unijeti svoju lokaciju. Ako postoji slobodno parkirališno mjesto, tada korisnik dobiva prijedlog o najbližem parkiralištu.



**Slika 6.4.5.** Dijagram stanja za za obrazac uporabe UC4 – *Dobivanje prijedloga*

**UC14** – Dodavanje novog parkirališta

Dijagram aktivnosti na slici (6.4.6) prikazuje aktivnosti dodavanja novog parkirališta. Administrator popunjava web obrazac te se on prosljeđuje na poslužitelj gdje se provjerava ispravnost unosa. U slučaju da neka od polja nisu ispravno popunjena, poslužitelj šalje obrazac natrag do administratora te ga on ponovno popunjuje. Ako su sva polja ispravna, parkiralište se stavlja u bazu podataka te se obavještava administratora da je dodavanje novog parkirališta uspjelo.

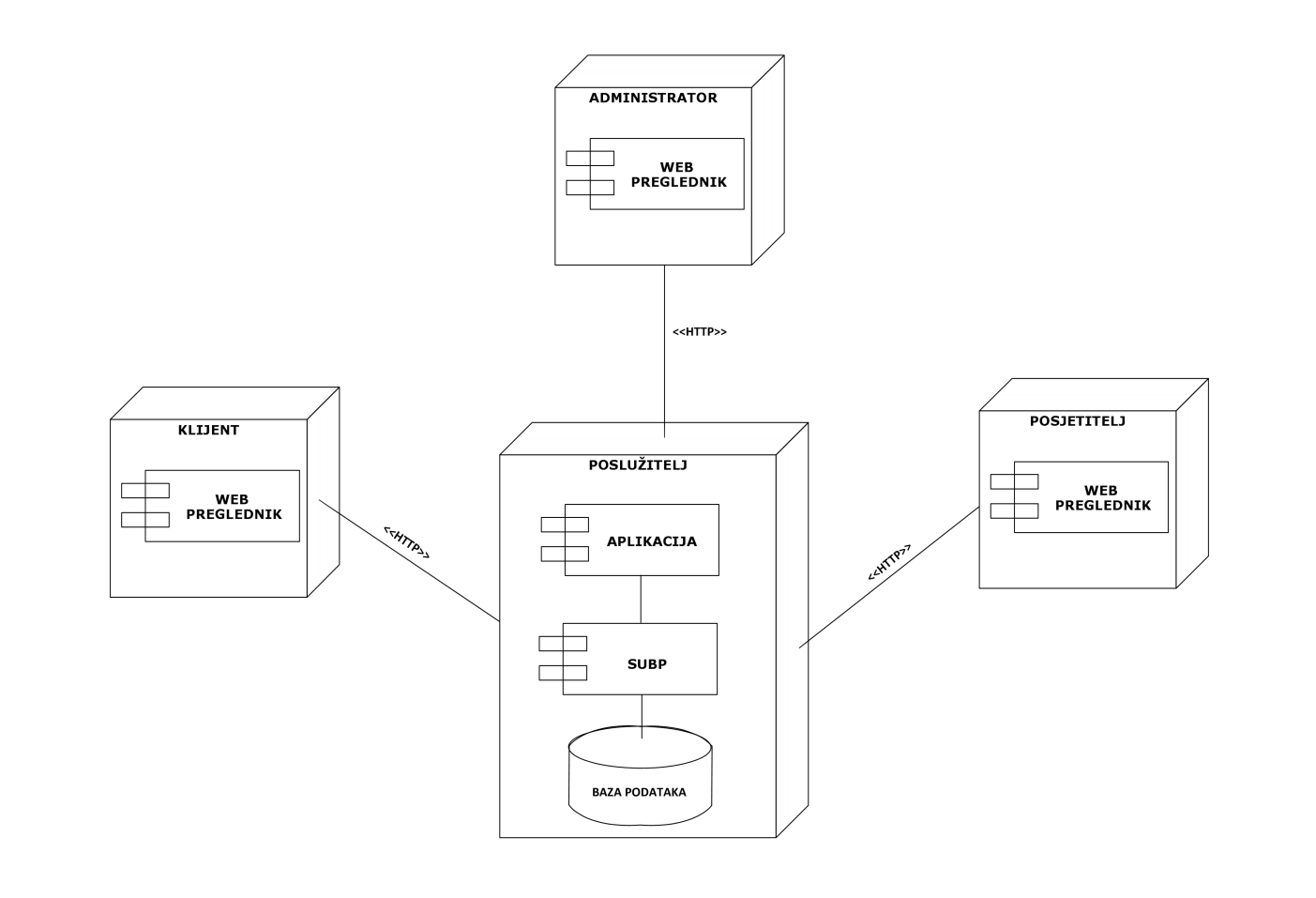


**Slika 6.4.6.** Dijagram aktivnosti za za obrazac uporabe UC14 – *Dodavanje novog parkirališta*

* + 1. Dijagram komponenata

1. Implementacija i korisničko sučelje
   1. Dijagram razmještaja

Sustav koristi klijent-server arhitekturu. Na računalu poslužitelja se izvršavaju poslužiteljska aplikacija i program sustava za upravljanje bazom podataka, a na njemu se nalazi i sama baza podataka. Korisnici pristupaju poslužitelju sa svojih računala putem *web*-preglednika (*http* klijenata), naravno uz uvjet da imaju pristup Internetu.



**Slika 7.1.1.** Dijagram razmještaja

* 1. Korištene tehnologije i alati

Pri izradi informacijskog sustava za tvrtku Gradski Parking d.o.o. korišteni su razvojna okolina aplikacija JetBrains (PHPstorm, Webstorm) i radni okvir Yii2 MVC.

Za rad s mySQL bazom podataka i testiranje aplikacije korištene su dvije slične aplikacije: WampServer i Xampp. Razlika među njima je u podršci (WampServer se može pokrenuti samo na Windowsima, dok Xampp radi na više operacijskih sustava), ali obje pružaju Apache-MySQL-PHP okruženje.

Prilikom razvoja aplikacije korišten je Github za razmjenu programskog koda te Basecamp prilikom izrade dokumentacije radi lakše komunikacije i organizacije. Prilikom razvoja sve navedene programske podrške korišten je Subversion sustav za kontrolu verzija.

Pri izradi UML dijagrama korišteni su ConceptDraw i Astah Community.

Za komunikaciju unutar projektnog tima korištena je Facebook grupa.

* 1. Isječak programskog koda vezan za temeljnu funkcionalnost sustava

*U ovom poglavlju potrebno je prikazati isječak programa koji prema mišljenju studenta ostvaruje temeljnu funkcionalnost u sustavu (ili nekom modulu).*

* 1. Ispitivanje programskog rješenja

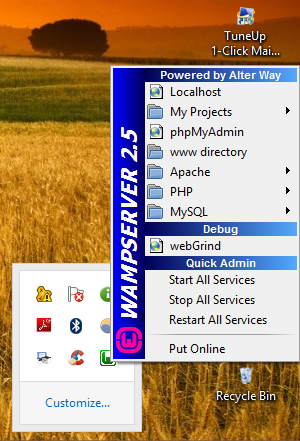
*U ovom poglavlju je potrebno opisati provedbu ispitivanja implementiranih funkcionalnosti s prikazom odabira ispitnih slučajeva.*

*Prilikom prezentacije svojih Ispitnih scenarija (minimalno 4) studenti trebaju ispitati temeljnu funkcionalnost i rubne uvjete. Poželjno je da se napravi i ispitni slučaj koji koristi funkcionalnosti koje nisu implementirane te da se vidi na koji način sustav reagira kada nešto nije u potpunosti ostvareno.*

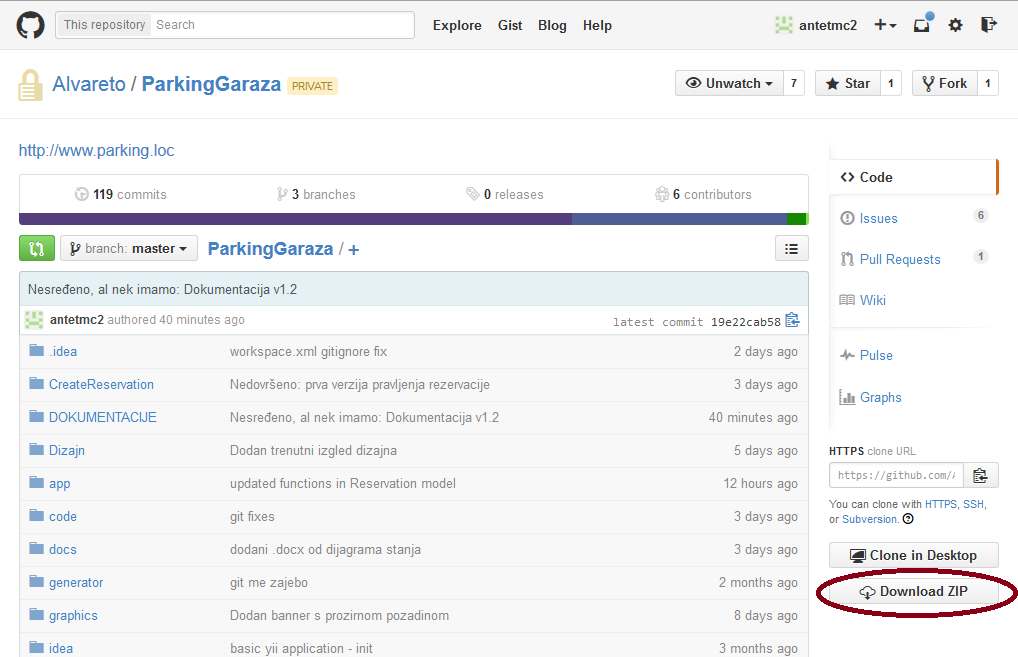
*Različiti ulazi za ispitne scenarije trebaju pokrivati temeljnu funkcionalnost nekog modula i nekoliko rubnih uvjeta.*

* 1. Upute za instalaciju

Instalacije aplikacije i svega potrebnog odvija se na sljedeći način:

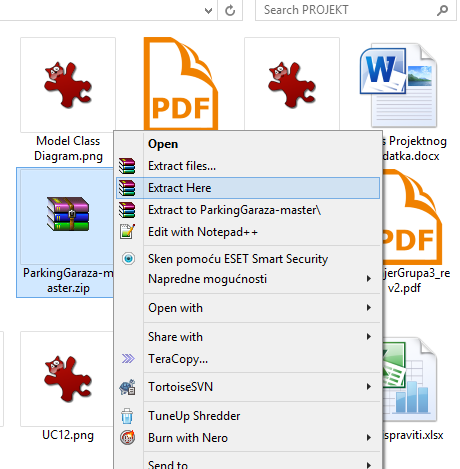
1. Instalacija alata:
   1. WampServer
   2. Xampp  
      Dovoljno je instalirati jedan od ova dva alata, a mi ćemo zbog jednostavnosti ovdje opisati korištenje aplikacije pomoću alata WampServer.
   3. Neki od alata za otpakiranje ZIP arhive (preporučujem 7-zip ili WinRAR), ali ovaj korak nije obavezan ukoliko ćemo koristiti alat za ZIP arhive koji dolazi uz Windowse.
2. Klikom na prečicu do aplikacije otvara se sama aplikacija. Potrebno je pričekati dok se sve ne podigne, a to ćemo znati po tome kad ikona pozeleni.  
     
     
   Kad se aplikacija podigne i dobijemo izgled sličan slici 7.5.1., to znači da se aplikacija uspješno podignula i možemo krenuti s njezinim korištenjem.

**Slika 7.5.1.** – Pokretanje aplikacije

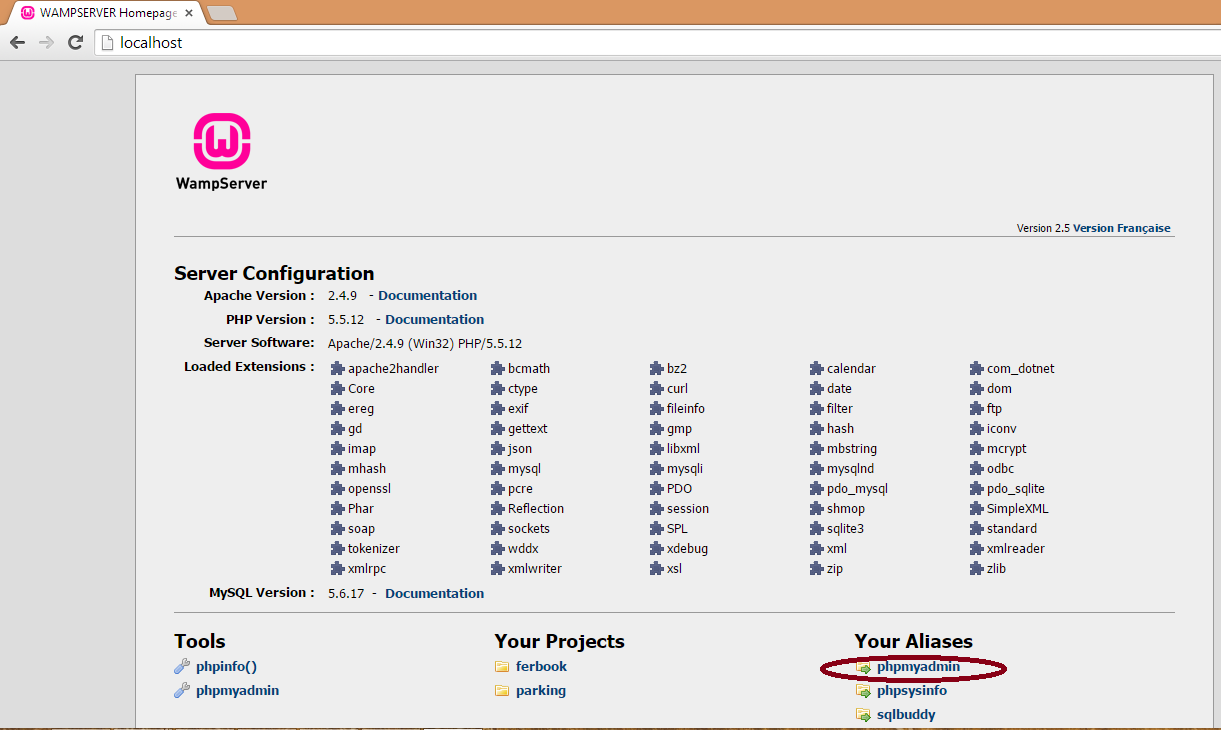
1. Za sljedeći je korak potrebno dohvatiti aplikaciju. To možemo napraviti na više načina, a mi ćemo navesti najlakši i najpraktičniji način da se to napravi. Potrebno je putem bilo kojeg web-preglednika slijediti sljedeću poveznicu: <https://github.com/Alvareto/ParkingGaraza>. Klikom na nju se otvara novi prozor sa svim datotekama koje se koriste u aplikaciji (pošto su upute rađene u trenutku kad aplikacija još nije bila završena, screenshot će se razlikovati).

**Slika 7.5.2.** – Github repozitorij web-aplikacije

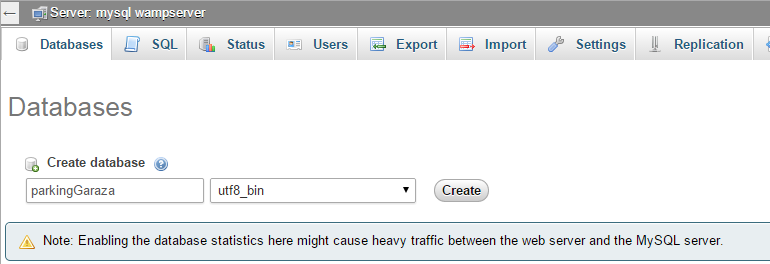
U lijevom se dijelu poveznice nalaze sve datoteke koje su stavljene u repozitorij te se iste mogu dohvatiti odvojeno od aplikacije. Također je moguće vidjeti mnoge druge podatke poput promjena, grana i članova projekta koji su sudjelovali u dodavanju datoteka. Nama je najvažniji dio zaokružen crvenim kružićem na screenshotu. Klikom na Download ZIP možemo preuzeti cijeli repozitorij koji onda možemo uređivati i sređivati na koji god način želimo. Ovaj korak je dosta važan jer nam je potreban za pregledavanje naše aplikacije.

1. Nakon što smo skinuli ZIP arhivu, potrebno ju je otpakirati i smjestiti u odgovarajući repozitorij. Ovdje je prikazano otpakiranje ZIP arhive pomoću alata WinRAR, ali se na sličan način radi i kod alata koji dolazi uz Windowse. Nakon što smo skinuli arhivu iz repozitorija, potrebno ju je locirati i kliknuti desnim klikom miša i na Extract Here kako bismo ju otpakirali cijelu u trenutnom folderu. Ovaj je postupak prikazan na slici ispod.  
     
   Nakon završenog otpakiranja stvorena je mapa ParkingGaraza-master te je potrebno novonastalu mapu smjestiti u repozitorij jednog od dva instalirana alata za testiranje aplikacije (to je u mom slučaju WampServer i poveznica C:\wamp\www, dok je u osnovnom slučaju ta lokacija jednaka root\wamp\www gdje root predstavlja lokaciju gdje je spremljena mapa wamp, tj. u mom je slučaju to C:\).

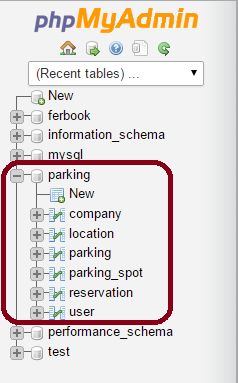
**Slika 7.5.3.** – Otpakiranje skinutog repozitorija



**Slika 7.5.4.** – WampServer: početna stranica

1. Sad smo spremni za pokretanje WampServera te je na početku potrebno otvoriti neki internetski preglednik (moja preporuka je Google Chrome zbog brzine i boljeg pregleda elemenata) i u adresnu traku upisati localhost.   
   Dobijemo izgled stranice koji je prikazan na slici 7.5.4. (dio Your Projects nije trenutno važan pa se razlike mogu zanemariti). Početna localhost stranica prikazuje izbornik WampServera, ali je nama najvažnija opcija phpmyadmin koja je zaokružena na slici crvenim kružićem. Klikom na tu poveznicu se otvara sama aplikacija phpmyadmin na kojoj ćemo podesiti našu bazu podataka.

**Slika 7.5.5.** – phpmyadmin: dodavanje nove baze podataka

1. Početna stranica phpmyadmin-a prikazuje na lijevoj strani sve dostupne baze podataka, dok su na desnoj strani prikazane razne opcije. Za dodavanja baze podataka moramo kliknuti na New nakon čega se otvara prozor prikazan na slici 7.5.5. na kojem ćemo početi stvarati našu bazu. U prostor za unos teksta unosimo željeno ime baze podataka, dok iz izbornika desno izaberemo *utf8\_bin* kojim su nam omogućeni hrvatski dijakritički znakovi. Nakon što to obavimo, kliknemo na Create te nam se u lijevom dijelu phpmyadmin-a pojavila novostvorena baza podataka. Na istoj je slici prikazan dio izbornika uključujući i opciju Import koja i nama treba za dodavanje baze podataka u novonastalu bazu.  
   Klikom na novostvorenu bazu pa na Import dobivamo novi prozor u kojem nam je jedini važan dio File to Import. Potrebno je kliknuti na gumb i locirati SQL datoteku u koju je spremljena baza podataka. Ako se radi u WampServeru, baza podataka *parking.sql* se nalazi na lokaciji root\wamp\www\parking\docs\database gdje root opet predstavlja lokaciju na kojoj je spremljena mapa wamp. Nakon što ju odaberemo, samo odemo na dno stranice, kliknemo na Go i sačekamo da završi postupak. Ako je postupak dodavanja baze bio uspješan, kod novonastale bi baze trebao biti prikazan sadržaj poput sadržaja na slici ispod.

**Slika 7.5.5.** – phpmyadmin: uspješno dodana baza podataka

1. Baza podataka je spremna i web-aplikaciju pokrećemo tako da unesemo localhost/parking/app/web/ u adresnu traku internetskog preglednika što nas vodi na početnu stranicu web-aplikacije. Moguća je registracija, a korisnik se može prijaviti sa sljedećim podacima:
   1. **Korisničko ime:**
   2. **Lozinka:**
   3. Korisničke upute

Korisničke su upute podijeljene prema mogućim vrstama korisnika web-aplikacije:

1. neregistrirani korisnici (posjetitelji)
2. registrirani korisnici (klijenti)
3. administratori

Za svaku će vrstu korisnika biti opisane sve mogućnosti uz navedene usluge, a ispod ovog odlomka možete vidjeti *screenshot* početne stranice aplikacije. Sama aplikacija je napravljena tako da bude jednostavna za korištenje prosječnim korisnicima.

(ubaciti screenshot kad bude gotova početna stranica – trenutak pretrage ili kad se pregledava neko parkiralište)

* + 1. Neregistrirani korisnici (posjetitelji)

Pregled lokacija parkirališta, informacija i cjenika

Prilikom otvaranja početne stranice posjetitelj može vidjeti veliku Google Maps kartu s pretragom na kojoj su markerima označena sva parkirališta i objekti s pripadajućim lokacijama te informacijski oblačići s ukupnim i slobodnim brojem parkirališnih mjesta. Posjetitelj može unijeti ili klikom na kartu označiti svoju lokaciju na temelju koje može dobiti informaciju o najbližem parkiralištu. Na početnoj stranici se još nalaze i cjenik na karti koji prikazuje najvažnije informacije o cijeni parkiranja te traka s poveznicama koja je podijeljena u dva dijela. Na lijevoj strani trake se nalaze poveznice vezani uz svaku vrstu korisnika te su to za posjetitelja *Prijava* i *Registracija*, dok se na desnoj strani nalaze poveznice koji se pojavljuju svakoj vrsti korisnika neovisno o statusu: *Upute* i *O nama*. Korisnika će klik na neku od poveznica odvesti na web-stranicu na koju vodi odabrana poveznica. Poveznica *Prijava* vodi na web-stranicu na kojoj se posjetitelj može prijaviti ukoliko već ima registrirano korisničko ime i lozinku, dok iste može napraviti pomoću poveznice *Registracija* ako još uvijek nema. Poveznica *Upute* vodi na web-stranicu na kojoj su prikazani detaljniji cjenik i upute o korištenju web-stranice, dok poveznica *O nama* prikazuje informacije o projektu i članovima projektnog tima.

(screenshot kad se završi početna)

Prijava i registracija posjetitelja

Posjetitelj se može prijaviti i registirati klikom na jedan od izbornika. Klikom na poveznicu *Prijava* posjetitelj se može prijaviti pomoću jedinstvenog korisničkog imena i lozinke. Još su dostupne opcije pamćenja prijave pomoću koje se korisnik ne mora prijavljivati stalno i ponovnog slanja lozinke e-mail pošto koju posjetitelj može iskoristiti ukoliko je zaboravio svoje korisničke podatke.

Klikom na poveznicu *Registracija* posjetitelj može

* + 1. Registrirani korisnici (klijenti)
    2. Administratori

1. Zaključak

Implementacija informacijskog sustava za parkiranje tvrtke Gradski Parking d.o.o pokazala se kao zahtjevan i zanimljiv zadatak. Razvijena je aplikacija koja je funkcionalna, stabilna te zadovoljava sve postavljene zahtjeve klijenta. Također je priložena detaljna dokumentacija sa svim bitnim informacijama. Iako je razvijena samo osnovna funkcionalnost aplikacije, prostora za napredak uvijek ima ako će se nastaviti s razvojem.

Najvećim izazovom se pokazala izrada same web-aplikacije zbog manjka iskustva u dizajniranju takvih aplikacija. Pošto smo imali dva člana tima s iskustvom u izradi web-aplikacija prije nego što smo počeli službeno raditi na aplikaciji za projekt, oni su napravili većinu potrebnog posla. U početku izrade aplikacije smo isto imali manjih problema s prilagodbom na Yii2 Framework, ali smo sve uspjeli riješiti. Također je ovo područje u kojem smo svi stekli najviše iskustva pošto smo se svi okušali u izradi web-aplikacija na backendu ili frontendu. Možda bi veće znanje o izradi web-aplikacija i iskustvo pomogli u još boljoj izradi informacijskog sustava, ali mislimo da smo napravili dobar posao.

Daljnji bi rad na projektu bio usmjeren prema proširivanju funkcionalnosti i nadogradnji. Jedna od mogućih ideja je izrada jedinstvenog nadograđenog informacijskog sustava za parkiranje koji bi uključivao neke novitete poput naprednijih senzora koji bi mogli prikazivati još više podataka, *pametniji* sustav rezervacija koji bi mogao uključivati još više tipova rezervacija s većim mogućnostima kombiniranja, više tipova objekata i parkirališnih mjesta za parkiranje te suradnje više sličnih tvrtki čime bi se stvorila još interaktivnija web-stranica jer bi sav potreban sadržaj bio dostupan na jedinstvenoj web-stranici. Već je danas teško naći parkirališno mjesto pa bi ovakav informacijski sustav mogao biti vrlo zanimljiv i dopadljiv ljudima ukoliko bi ga bilo moguće izvesti jer se radi o nečemu što bi moglo biti jako potrebno i korisno. U slučaju da bi ovakav sustav bio izvediv, za nadogradnju bi trenutni sustav trebalo proširiti kako bi se mogli pokriti svi novopostavljeni zahtjevi. Moguća popularnost (u slučaju uspjeha aplikacije) i opseg ovakvog sustava može zahtijevati barem nekoliko stručnjaka za izradu web-aplikacija i održavanje web-poslužitelja pa je moguće da bi bilo potrebno zaposliti još ljudi.

Sve u svemu, ovaj je projekt bio jedno zanimljivo i poučno iskustvo zahvaljujući kojem smo mogli iskušati kako je to raditi u timu na projektu i kako ostvariti sve ciljeve i zahtjeve koji su postavljeni pred nas. Također su neki od nas mislili da je ovaj projekt dobra ideja za stvarni sustav jer smo mogli uključiti vlastita vozačka iskustva pošto su neki članovi tima vozači koji žive u Zagrebu i okolici. Uglavnom smo zadovoljni svime što smo ostvarili te se nadamo da će nam ovo iskustvo puno koristiti u ostatku studija i nakon njega.

1. Reference
2. Oblikovanje programske potpore, FER ZEMRIS, <http://www.fer.hr/predmet/opp>
3. Oblikovanje programske potpore, FER ZEMRIS, <http://www.zemris.fer.hr/predmeti/opp>
4. UML-dijagrami; Alan Jović, Marko Horvat, Igor Grudenić; Zagreb 2014.
5. ArgoUML, <http://argouml.tigris.org/>
6. PHP, <http://php.net/>
7. Apache HTTP Server, <http://httpd.apache.org/>
8. MySQL, <http://www.mysql.com/>
9. Google Maps, <https://developers.google.com/maps/documentation/>

Dodatak A: Dnevnik sastajanja

**Sastanak br. 1: srijeda, 8. listopada 2014. od 17:00 do 19:00**

Prisutni: Ivan Grgurina, Zoran Krišto, Antun Maldini, Katarina Matić, Ante Tomić, Josip Vukušić, Ivan Žagar

Tema sastanka: prvi sastanak grupe

Ciljevi: upoznavanje, okvirno planiranje posla

Sadržaj sastanka: Nakon što smo se svi skupili i upoznali, raspravili smo moguće zahtjeve projektnog zadatka i sam voditelj projekta je naglasio što će nam sve trebati kod projekta. Odlučili smo da ćemo se naći još jednom kada dobijemo projektni zadatak. Dogovorena je ravnopravna rasporeda poslova među članovima projekta koja će biti odlučena na budućim sastancima.

**Sastanak br. 2: četvrtak, 16. listopada 2014. od 10:30 do 12:00**

Prisutni: Ivan Grgurina, Zoran Krišto, Antun Maldini, Katarina Matić, Ante Tomić, Josip Vukušić, Ivan Žagar

Tema sastanka: projektni zadatak, funkcionalni i nefunkcionalni zahtjevi

Ciljevi: dogovoriti buduće korake kod projekta

Sadržaj sastanka: Nakon što smo svi postavili svoje Basecamp i Github račune, odlučili smo se sastati da detaljnije razradimo projektni zadatak. Sam voditelj nam je spomenuo najvažnije stvari koje je dogovorio na konzultacijama te je pokazao neke osnovne alate koje ćemo koristiti. Također smo dogovorili da ćemo do sljedećeg termina konzultacija pripremiti funkcionalne i nefunkcionalne zahtjeve.

**Sastanak br. 3: subota, 25. listopada 2014. od 17:00 do 19:00**

Prisutni: Ivan Grgurina, Zoran Krišto, Antun Maldini, Katarina Matić, Josip Vukušić, Ivan Žagar

Tema sastanka: zahtjevi, obrasci uporabe, UML

Ciljevi: definirati i raspodijeliti izradu obrazaca uporabe

Sadržaj sastanka: Složeni su konačni funkcionalni zahtjevi, dodijeljeno je tko će od prisutnih izraditi koje obrasce uporabe te je postavljen rok završetka. Dogovoreno je da ćemo zapisati sve nejasnoće i pitanja koja će netko od članova projektnog tima (tko će imati vremena) pitati asistenticu na sljedećem terminu konzultacija.

**Sastanak br. 4: petak, 7. studenoga 2014. od 11:00 do 13:00**

Prisutni: Ivan Grgurina, Antun Maldini, Katarina Matić, Ante Tomić, Josip Vukušić, Ivan Žagar

Tema sastanka: obrasci uporabe i dijagrami obrazaca uporabe

Ciljevi: završiti dijagrame obrazaca uporabe i dogovoriti daljnje poslove

Sadržaj sastanka: Raspravili smo, ispravili, nadopunili i završili dijagrame obrazaca uporabe te smo dogovorili daljnje stvari koje moramo napraviti unutar sljedećih 9 dana.

**Sastanak br. 5: subota, 10. siječnja 2015. od 10:00 do 23:00**

Prisutni: Ivan Grgurina, Zoran Krišto, Antun Maldini, Katarina Matić, Ante Tomić, Josip Vukušić, Ivan Žagar

Tema sastanka:

Ciljevi:

Sadržaj sastanka:

**Sastanak br. 6: nedjelja, 11. siječnja 2015. od 15:00 do 23:00**

Prisutni: Ivan Grgurina, Zoran Krišto, Antun Maldini, Katarina Matić, Josip Vukušić, Ivan Žagar

Tema sastanka:

Ciljevi:

Sadržaj sastanka:

**Sastanak br. 7:**

Prisutni:

Tema sastanka:

Ciljevi:

Sadržaj sastanka: