

Oblikovanje programske potpore

Ak. god. 2019./2020.

Manje smeće, više sreće

Dokumentacija, Rev. 1.0

Grupa: *Kombinacija*

Voditelj: *Sven Skender*

Datum predaje: *12. studeni, 2019.*

Nastavnik: *Tomislav Jukić*

Sadržaj

1 Dnevnik promjena dokumentacije	3
2 Opis projektnog zadatka	4
3 Specifikacija programske potpore	7
3.1 Funkcionalni zahtjevi	7
3.1.1 Obrasci uporabe	9
3.1.2 Sekvencijski dijagrami	19
3.2 Ostali zahtjevi	22
4 Arhitektura i dizajn sustava	23
4.1 Baza podataka	23
4.1.1 Opis tablica	23
4.1.2 Definicije tablica	24
4.1.3 Dijagram baze podataka	27
4.2 Dijagram razreda	28
4.3 Dijagram stanja	29
4.4 Dijagram aktivnosti	30
4.5 Dijagram komponenti	31
5 Implementacija i korisničko sučelje	32
5.1 Korištene tehnologije i alati	32
5.2 Ispitivanje programskog rješenja	33
5.2.1 Ispitivanje komponenti	33
5.2.2 Ispitivanje sustava	33
5.3 Dijagram razmještaja	34
5.4 Upute za puštanje u pogon	35
6 Zaključak i budući rad	36
Popis literature	37

Indeks slika i dijagrama	38
Dodatak: Prikaz aktivnosti grupe	39

1. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1.0	Napravljen predložak, ispunjene osnovne informacije i početak pisanja opisa projekta	Bićanić	24.10.2019.
0.1.1	Nastavak pisanja opisa projekta	Bićanić	27.10.2019.
0.1.2	Završena prva verzija opisa projekta	Bićanić	28.10.2019.
0.2.0	Započeto pisanje UC dijagrama	Vasilj	28.10.2019.
0.2.1	Završeno pisanje UC dijagrama	Vasilj	29.10.2019.
0.3.0	Započeto pisanje Opisa Baze Podataka	Bićanić	1.11.2019.
0.3.1	Završeno pisanje Opisa Baze Podataka	Bićanić	3.11.2019.
0.3.2	Dodan dijagram i ER model baze podataka	Bićanić	3.11.2019.
0.4.0	Započeto rasписivanje Use Caseova	Vasilj	29.10.2019.
0.4.1	Završeno rasписivanje Use Caseova	Vasilj	4.11.2019.
0.5.1	Rasписani ostali zahtjevi projekta	Vasilj	4.11.2019.
0.6.0	Dodani sekvencijski dijagrami i opisi istih	Vasilj	5.11.2019.
1.0.0	Napisano poglavlje o arhitekturi sustava	Bićanić	10.11.2019.

Tablica 1.1: Popis promjena dokumentacije

2. Opis projektnog zadatka

Svakodnevno se susrećemo sa neispravno odloženim otpadom, nerijetko upravo zato što su kontejneri predviđeni za njega puni ili čak pretrpani. Jedna od posljedica svega toga je i da recikliranje postaje znatno otežano.

Ovim projektom želimo riješiti taj problem tako što bismo razvili web aplikaciju za cjelokupan sustav odvoza otpada. Koristeći se njom, građani bi na karti mogli odabrati neki kontejner i označiti ga praznim, punim ili pretrpanim, a aplikacija bi bila integrirana sa postojećim komunalnim službama koje bi tim putem dobivale obavijesti o punim kontejnerima, što bi olakšalo planiranje rute i fokusiralo pražnjenje kontejnera na mjesta na kojima je stvarno potrebno.

Aplikacija prilikom pokretanja nudi izbornik kao i polje za unos datuma, te prikazuje kartu područja u okolini klijentovog uređaja na kojoj su označeni kontejneri. Putem izbornika se neregistrirani korisnici mogu registrirati, a neprijavljeni prijaviti. Aplikacija razlikuje tri vrste prijavljenih korisnika (poredano po razini ovlasti, od najniže prema najvišoj):

- *Građanin*
- *Komunalni radnik*
- *Administrator*

Koristeći se oznakama na karti, svi korisnici, uključujući i neregistrirane, mogu za svaki kontejner vidjeti informacije o prijavljivanju i pražnjenju kontejnera, i to za onaj dan koji je upisan u polju za unos datuma.

Građaninu je za registraciju dovoljna ispravna e-mail adresa i lozinka, pri čemu ne mogu postojati dva korisnika (bilo koje razine ovlasti) sa istom e-mail adresom. Nakon uspješne prijave i u slučaju da dan upisan u polju datuma odgovara današnjem, građaninu se pritiskom na neki kontejner na karti otvara izbornik nad tim kontejnerom putem kojega korisnik može:

- kontejner prijaviti kao pun
- kontejner prijaviti kao pretrpan

- kontejner prijaviti kao prazan, samo ako je taj kontejner već (lažno) označen kao pretrpan

Prilikom prijave kontejnera građanin uz prijavu može priložiti i fotografiju toga kontejnera kao dokaz stanja.

Ako upisani datum ne odgovara današnjem, onda je omogućen samo pregled svih prijava i pražnjenja kontejnera za taj dan. Bez obzira na datum u polju za unos datuma, korisnik može bilo koji kontejner spremi u osobni popis kontejnera kako bi imao brži pristup kontejnerima u koje češće odlaže otpad (kao što su, primjerice, oni u blizini stana).

Korisnik u ulozi komunalnog radnika ima sve ovlasti koje ima i građanin, izuzev činjenice da se on ne može registrirati sam, već tu akciju obavlja isključivo korisnik u ulozi administratora. Za komunalnog redara sustav dodatno pamti ime, prezime i Osobni Identifikacijski Broj (OIB). Svaki komunalni radnik je dodjeljen jednom području grada (ili općenito mjesta u kojem se aplikacija koristi), te za sve kontejnere toga područja ima dodatne ovlasti:

- može isplanirati rutu odvoza otpada (putem izbornika)
- može **svaki** kontejner označiti kao prazan
- može označiti kontejner kao lažno prijavljen

Ovlasti administratora sustava uz sve dosad navedene omogućuju i dodatne funkcionalnosti, a njegov korisnički račun se stvara ručno. Uloga administratora je:

- raspodjela radnika po područjima mjesta na kojem se aplikacija koristi
- premještanje radnika, ovisno o trenutnim uvjetima
- dodavanje radnika (stvaranje njegovog korisničkog računa)
- brisanje radnika i pripadajućeg korisničkog računa
- dodavanje i brisanje kontejnera

S obzirom da se u aplikaciju može prijaviti svatko, postoji izvjesna mogućnost zlouporabe u obliku lažnog označavanja kontejnera kao punog/pretrpanog. Zbog toga aplikacija interno za svakog građanina pamti reputaciju koja se gradi na temelju svih prijava nekog korisnika. Ukoliko komunalni radnik u svojem obilasku kontejnera naiđe na kontejner koji je prijavljen kao pun, onog trenutka kada ga

komunalni radnik isprazni i označi ispražnjenim, automatski se svim korisnicima koji su ga prijavili reputacija diže. Shodno tome, ukoliko naiđe na prazan kontejner koji je prijavljen kao pun, komunalni radnik prijavljuje lažnu prijavu te se svim korisnicima koji su taj kontejner prijavili reputacija smanjuje.

Prijava kontejnera od strane građana ima manju vrijednost nego prijava od strane komunalnog radnika ili administratora, uz obrazloženje da ljudi vjerojatno neće sabotirati sustav u koji su sami uključeni i u kojem rade. Vrijednost prijave građana proporcionalna je njegovoj reputaciji, ali nikad nije veća od vrijednosti prijave radnika ili administratora.

Da bi komunalne službe dobile konkretnu obavijest o punom ili pretrpanom kontejneru, potrebno je prikupiti više građanskih prijava. Kada se službi pošalje obavijest o tome da je kontejner u nekom stanju, korisnici i dalje mogu prijavljivati taj kontejner, ali se obavijesti više neće generirati. Iznimka ovom pravilu je prijavljivanje suprotnog stanja: ukoliko je kontejner lažno prijavljen kao pun ili pretrpan, i već je poslana obavijest službama, moguće je da više korisnika označi taj kontejner kao prazan te se tim putem generira protu-obavijest koja službe obavještava da je kontejner prazan.

Nakon što je neki korisnik prijavio kontejner kao pun, ne može ga više označiti kao takvog sve dok ga komunalne službe ili drugi građani ne označe praznim. Jednako tako, ako ga korisnik označi praznim, ne može to ponoviti sve dok ga dovoljno ljudi ne označi punim i generira se obavijest prema službama.

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Građanin
2. Komunalni radnik
3. Administrator
4. Vlasnik (naručitelj)
5. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani/ neprijavljeni korisnik može:
 - (a) za odabrani datum na karti pregledati kontejnere
 - (b) odabrati pojedini kontejner
 - i. pregledati popis prijava i pražnjena u tom danu
 - ii. pregledati grafički prikaz prijava i pražnjenja u tom danu
 - (c) registrirati se u sustav
2. Administrator može:
 - (a) prijaviti se u sustav
 - (b) dodati novo kvartovsko poduzeće
 - (c) izbrisati postojeće kvartovsko poduzeće
 - (d) dodati novi kontejner
 - (e) izbrisati postojeći kontejner
 - (f) upravljati komunalnim radnicima
 - i. dodati novog komunalnog radnika u sustav
 - ii. pridodati novog komunalnog radnika kvartu
 - iii. premjestiti komunalnog radnika u drugi kvart
 - iv. izbrisati komunalnog radnika iz sustava
 - (g) prijaviti stanje kontejnera te priložiti sliku istoga

- i. prijaviti da je kontejner prazan
- ii. prijaviti da je kontejner pun
- iii. prijaviti da je kontejner pretrpan

3. Građanin može:

- (a) prijaviti se u sustav
- (b) prijaviti stanje kontejnera te priložiti sliku istoga
 - i. prijaviti da je kontejner prazan
 - ii. prijaviti da je kontejner pun
 - iii. prijaviti da je kontejner pretrpan

4. Komunalni radnik može:

- (a) prijaviti se u sustav
- (b) preuzeti rutu za pražnjenje kontejnera
- (c) za svaki kontejner u ruti
 - i. prijaviti da je ispražnjen
 - ii. prijaviti lažnu dojavu ukoliko kontejner naveden u ruti nije pun
- (d) prijaviti stanje kontejnera te priložiti sliku istoga
 - i. prijaviti da je kontejner prazan
 - ii. prijaviti da je kontejner pun
 - iii. prijaviti da je kontejner pretrpan

5. Baza podataka (sudionik):

- (a) pohranjuje sve podatke o korisnicima i njihovim ovlastima
- (b) pohranjuje sve podatke o kvartovima, kontejnerima i prijavama

3.1.1 Obrasci uporabe

UC01-Registracija

- **Glavni sudionik:** Neregistrirani korisnik
- **Cilj:** Stvaranje korisničkog računa za pristup sustavu
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Korisnik odabire registraciju u sustav
 2. Korisnik unosi ime, prezime, email i lozinku
 3. Korisnik dobiva obavijest o uspješnoj registraciji u sustav
- **Opis mogućih odstupanja:**
 - 2.a Email je prethodno već zauzet
 1. Sustav obavještava korisnika da unese mail koji ne postoji u sustavu
 2. Korisnik ponovno unosi podatke ili odustaje od registracije
 - 2.b Unos neispravnog emaila
 1. Sustav obavještava korisnika da je unio neispravan podatak
 2. Korisnik ponovno unosi podatke ili odustaje od registracije
 - 2.c Korisnik je unio manje od osam znakova za lozinku
 1. Sustav obavještava korisnika da je unio neispravan podatak
 2. Korisnik ponovno unosi podatke ili odustaje od registracije

UC02-Prijava u sustav

- **Glavni sudionik:** Građanin, komunalni radnik, administrator
- **Cilj:** Mogućnost pristupa korisničkom sučelju
- **Sudionici:** Baza podataka
- **Preduvjet:** Registracija
- **Opis osnovnog tijeka:**
 1. Korisnik unosi korisničko email i lozinku
 2. Sustav vraća poruku o ispravnosti unesenih podataka
 3. Sustav korisniku omogućava pristup korisničkim funkcijama
- **Opis mogućih odstupanja:**
 - 2.a Unos neispravnog emaila ili lozinke
 1. Sustav obavještava korisnika da je unio neispravan podatak i vraća korisnika u korak 1.

UC03-Dodaj najkorištenije kontejnere

- **Glavni sudionik:** Građanin, komunalni radnik, administrator
- **Cilj:** Brži pristup kontejner koje su predmet česte uporabe
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire traženi kontejner na karti
 2. Sustav prikazuje opcije vezane za odabrani kontejner
 3. Korisnik odabire i označava opciju "omiljeni kontejner"
 4. Sustav sprema unesenu promjenu

UC04-Ukloni iz najkorištenijih

- **Glavni sudionik:** Građanin, komunalni radnik, administrator
- **Cilj:** Povećanje preglednosti karte i lakša uporaba iste
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire traženu kantu na karti
 2. Sustav prikazuje opcije vezane za odabrani kontejner
 3. Korisnik odabire i označava opciju "ukloni iz omiljenih kontejnera"
 4. Sustav sprema unesenu promjenu

UC05-Preuzimanje rute

- **Glavni sudionik:** Komunalni radnik
- **Cilj:** Preuzimanje optimalne putanje za pražnjenje kanti
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire preuzimanje rute
 2. Sustav prikazuje popis kontejnera za pražnjenje
 3. Korisnik pokreće rutu
- **Opis mogućih odstupanja:**
 - 1.a Nedostatan broj kontejnera spremnih za pražnjenje
 1. Sustav javlja da nema dovoljan broj kontejnera za stvoriti rutu i vraća korisnika u korak 1

UC06-Potvrda pražnjenja

- **Glavni sudionik:** Komunalni radnik
- **Cilj:** Potvrda pražnjenja kanti s rute
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen i pokrenuo je rutu (UC05)
- **Opis osnovnog tijeka:**
 1. Korisnik odabire kontejner iz rute
 2. Sustav prikazuje opcije vezane za odabrani kontejner
 3. Korisnik označava da je kontejner ispražnjen
 4. Sustav sprema unesenu promjenu
- **Opis mogućih odstupanja:**
 - 3.a kontejner nije pun
 1. Korisnik označava da je dojava lažna
 2. Sustav prikazuje sljedeći kontejner iz rute

UC07-Dodavanje komunalnog radnika

- **Glavni sudionik:** Administrator
- **Cilj:** Stvaranje korisničkog računa za novog korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire kvart
 2. Sustav prikazuje popis radnika u odabranom kvartu
 3. Korisnik odabire stvaranje novog radnika u odabranom kvartu
 4. Sustav otvara formu za registraciju novog radnika
 5. Korisnik unosi podatke o novom radniku
 6. Sustav sprema unesenu promjenu
- **Opis mogućih odstupanja:**
 - 3.a U odabranom kvartu ima previše radnika
 1. Sustav obavještava korisnika da nije moguće unijeti novog radnika i vraća ga u korak 2
 5. a Korisnik je unio neispravne podatke o novom radniku
 1. Sustav upozorava na pogrešku pri unosu i vraća korisnika u korak 4
 5. a Korisnik je pokušao stvoriti novog radnika koji već postoji u sustavu
 1. Sustav javlja kako radnik već postoji i vraća korisnika u korak 2

UC08-Premještanje komunalnog radnika

- **Glavni sudionik:** Administrator
- **Cilj:** Premještanje komunalnog radnika iz jedne kvartovske podružnice u drugu
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire kvart u kojem se trenutno nalazi traženi radnik
 2. Sustav prikazuje popis radnika u odabranom kvartu
 3. Korisnik odabire traženog radnika
 4. Sustav prikazuje podatke o odabranom radniku
 5. Korisnik mijenja kvart kojem je radnik pridjeljen
 6. Sustav sprema unesenu promjenu
- **Opis mogućih odstupanja:**
 - 5.a Prevelik broj radnika u ciljanom poduzecu
 1. Sustav obavještava korisnika o nemogućnosti izvršavanja tražene akcije i vraća korisnika u korak 4
 - 5.b Premalo radnika u početnom kvartu
 1. Sustav obavještava korisnika o nemogućnosti izvršavanja tražene akcije i vraća korisnika u korak 4

UC09-Prijava praznog kontejnera

- **Glavni sudionik:** Administrator, komunalni radnik, građanin
- **Cilj:** Dojaviti sustavu da kontejneru nije potrebno pražnjenje
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire kontejner na karti
 2. Korisnik označava da je kontejner prazan
 3. Sustav sprema unesenu promjenu
- **Opis mogućih odstupanja:**
 - 2.a Korisnik ima prevelik broj lažnih dojava
 1. Sustav ne sprema unsenu promjenu

UC10-Prijava punog kontejnera

- **Glavni sudionik:** Administrator, komunalni radnik, građanin
- **Cilj:** Dojaviti sustavu da je kontejneru potrebno pražnjenje

- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire kontejner na karti
 2. Korisnik označava da je kontejner pun
 3. Sustav sprema unesenu promjenu
- **Opis mogućih odstupanja:**
 - 2.a Korisnik ima prevelik broj lažnih dojava
 1. Sustav ne sprema unsenu promjenu

UC11-Prijava prepunog kontejnera

- **Glavni sudionik:** Administrator, komunalni radnik, građanin
- **Cilj:** Dojaviti sustavu da je kontejneru hitno potrebno pražnjenje
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire kontejner na karti
 2. Korisnik označava da je kontejner prepun
 3. Sustav sprema unesenu promjenu
- **Opis mogućih odstupanja:**
 - 2.a Korisnik ima prevelik broj lažnih dojava
 1. Sustav ne sprema unsenu promjenu

UC12-Fotografiranje kontejnera

- **Glavni sudionik:** Administrator, komunalni radnik, građanin
- **Cilj:** Priložiti sliku prijavi stanja kontejnera
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav i prijavio je stanje kontejnera (UC09 - UC11)
- **Opis osnovnog tijeka:**
 1. Korisnik slika kontejner i šalje sliku
 2. Sustav sprema unesenu promjenu

UC13-Priegled arhive

- **Glavni sudionik:** Anonimni korisnik
- **Cilj:** Pregled dosadašnjih prijava i pražnjenja kontejnera

- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Korisnik odabire datum
 2. Sustav prikazuje korisniku kartu sa kontejnerima za navedeni datum
 3. Korisnik odabire pojedini kontejner
 4. Sustav prikazuje popis prijava i pražnjenje tog kontejnera narednog dana
- **Opis mogućih odstupanja:**
 - 1.a Unos datuma za kojeg stanje nije poznato
 1. Sustav obavještava korisnika da nije moguće dohvatiti traženo stanje
 2. Sustav vraća korisnika u korak 1

UC14-Dodavanje kvarta

- **Glavni sudionik:** Administrator
- **Cilj:** Stvaranje nove kvartovske podružnice
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire popis kvartova
 2. Korisnik odabire stvaranje novog kvarta
 3. Sustav otvara formu za stvaranje novog kvarta
 4. Korisnik ispunjava formu s podacima o kvartu
 5. Sustav sprema unesenu promjenu
- **Opis mogućih odstupanja:**
 - 2.a Kvart već postoji
 1. Sustav obavještava korisnika da je unio neispravan podatak
 2. Korisnik ponovno unosi podatke ili odustaje od stvaranja novog kvarta

UC15-Brisanje kvarta

- **Glavni sudionik:** Administrator
- **Cilj:** Brisanje kvartovske podružnice
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire popis kvartova
 2. Korisnik odabire kvart za brisanje

3. Korisnik uklanjanja odabrani kvart
4. Sustav sprema unesenu promjenu
- **Opis mogućih odstupanja:**
 - 3.a Postoje djelatnici pridjeljeni odabranom kvartu
 1. Sustav dojavljuje kako nije moguće obrisati kvartovsko poduzeće dok su njemu pridjeljeni radnici i vraća korisnika u korak 2

UC16-Dodavanje kontejnera

- **Glavni sudionik:** Administrator
- **Cilj:** Dodavanje novog kontejnera na karti
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire mjesto na karti
 2. Korisnik dodaje novi kontejner na odabrano mjesto
 3. Sustav sprema unesenu promjenu

UC17-Brisanje kontejnera

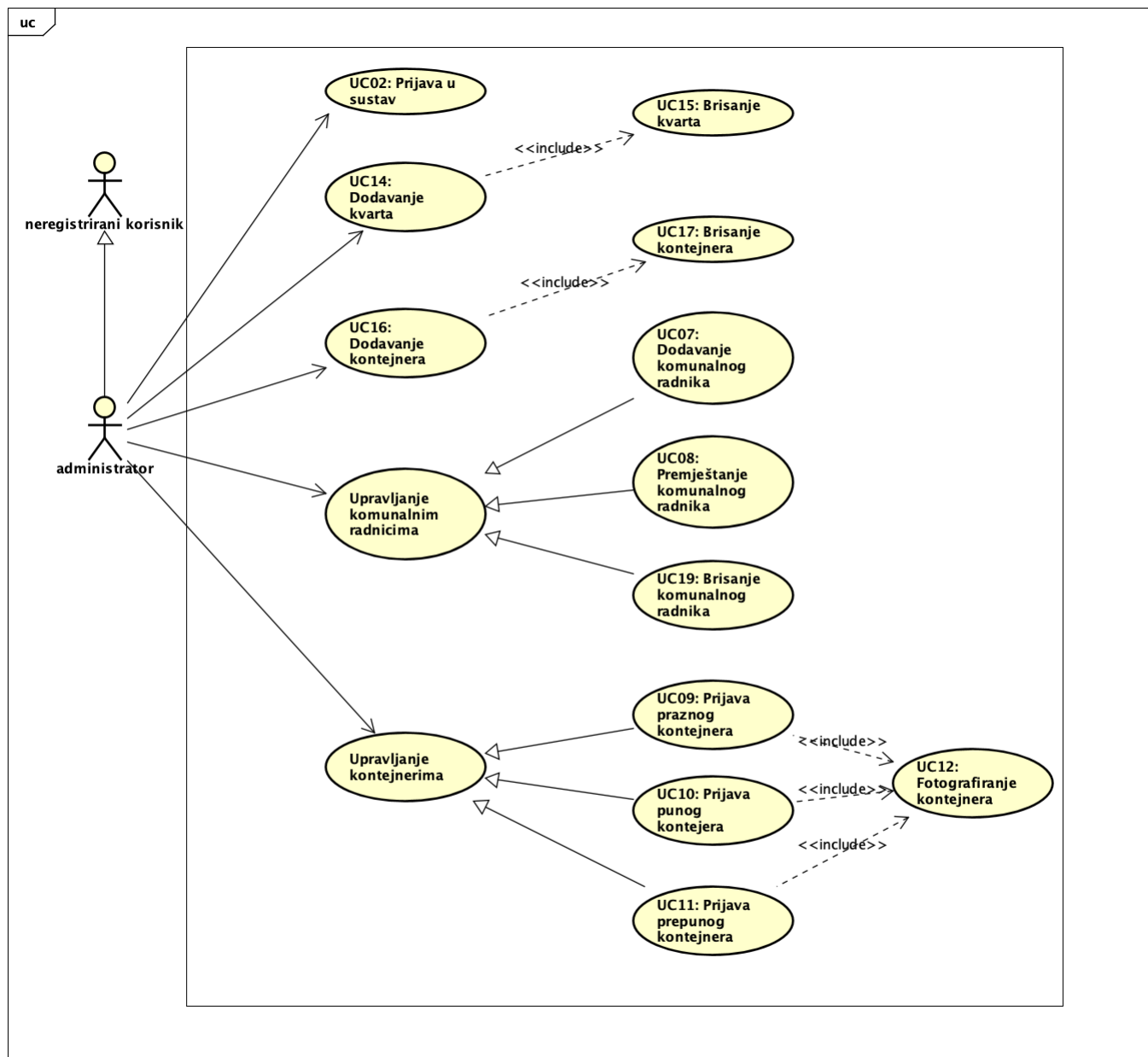
- **Glavni sudionik:** Administrator
- **Cilj:** Brisanje kontejnera s karte
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire kontejner na karti
 2. Korisnik uklanja odabrani kontejner
 3. Sustav sprema unesenu promjenu

UC19-Brisanje komunalnog radnika

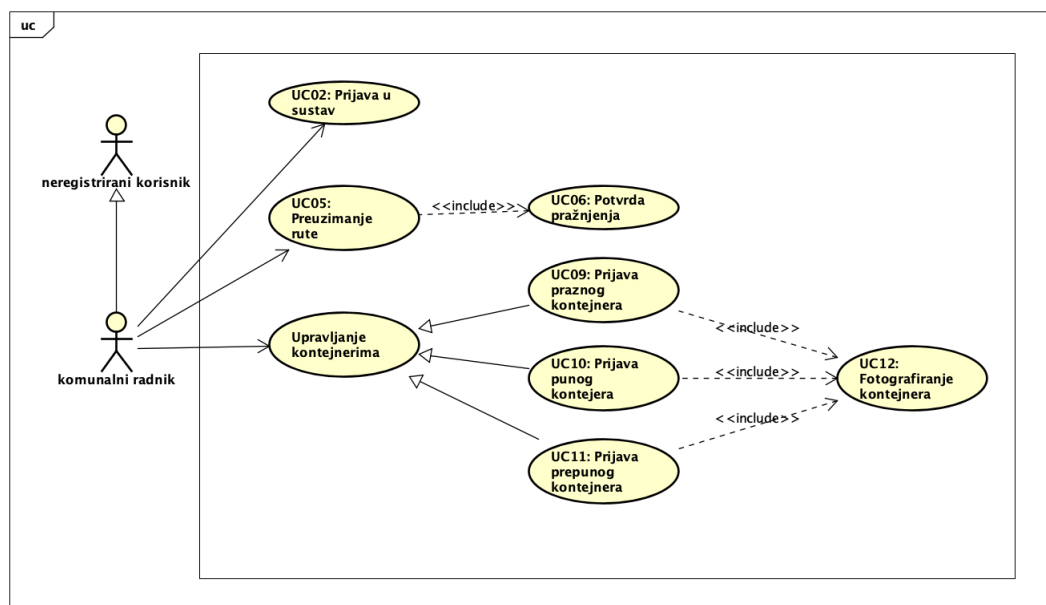
- **Glavni sudionik:** Administrator
- **Cilj:** Brisanje korisničkog računa komunalnom radniku
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav
- **Opis osnovnog tijeka:**
 1. Korisnik odabire kvart
 2. Sustava prikazuje popis radnika u odabranom kvartu
 3. Korisnik odabire komunalnog radnika

4. Sustav prikazuje podatke o odabranom radniku
5. Korisnik odabire brisanje odabranog radnika
6. Sustav sprema unesenu promjenu

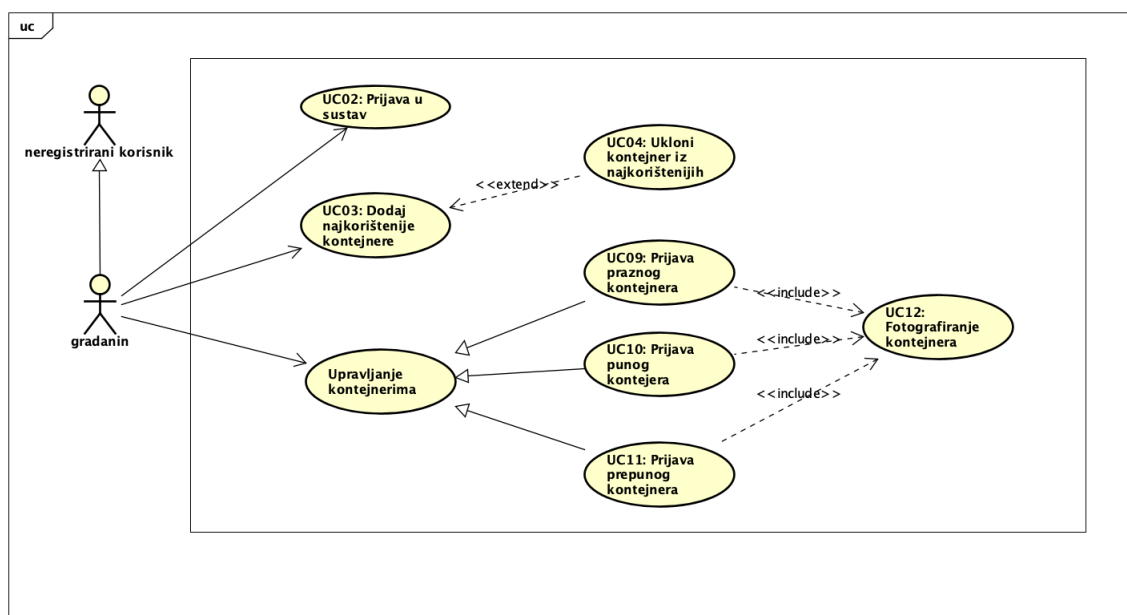
Dijagrami obrazaca uporabe



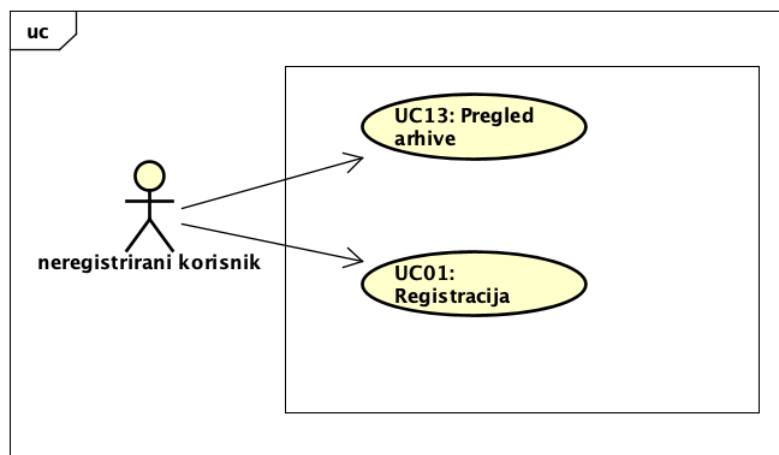
Slika 3.1: Dijagram obrasca uporabe, funkcionalnost administratora



Slika 3.2: Dijagram obrasca uporabe, funkcionalnost komunalnog radnika



Slika 3.3: Dijagram obrasca uporabe, funkcionalnost građanina

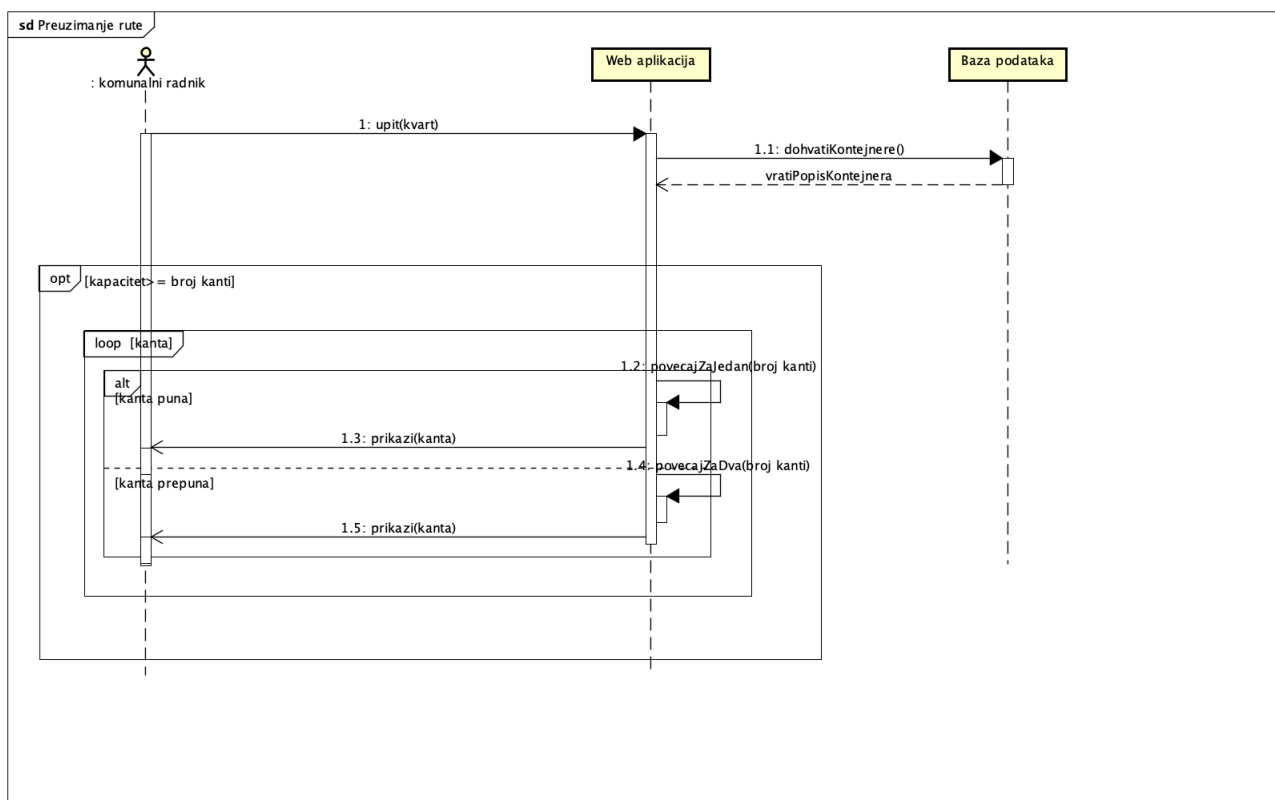


Slika 3.4: Dijagram obrasca uporabe, funkcionalnost neregistriranog korisnika

3.1.2 Sekvencijski dijagrami

Obrazac uporabe 05 - Preuzimanje rute

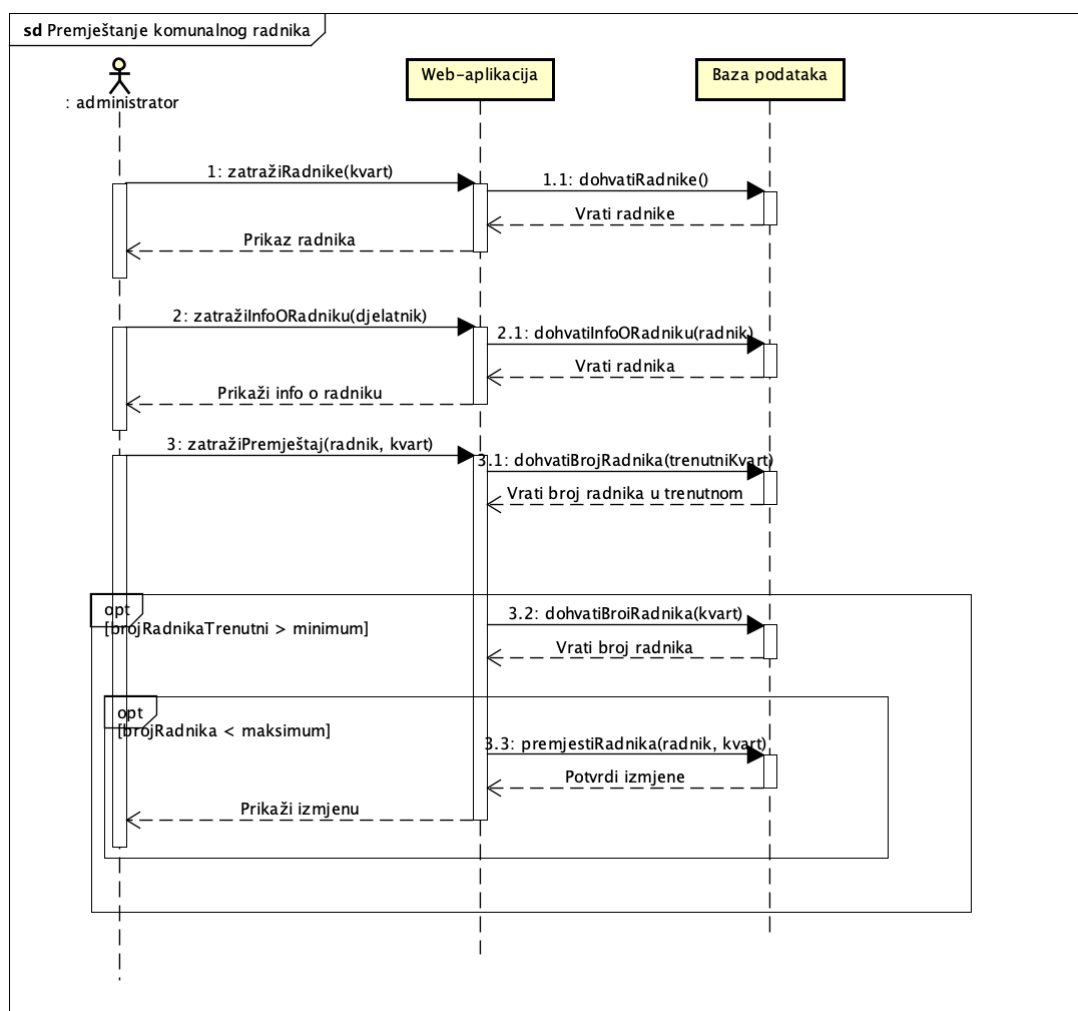
Komunalni radnik šalje zahtjev za dodjeljivanje rute kojom će prolaziti prilikom pražnjenja kontejnera u obliku popisa lokacija kontejnera koje mora isprazniti. Poslužitelj dohvaća kante u kvartu u kojem dotični komunalni radnik djeluje te mu ispišuje onoliko kontejnera koliko stane u njegov kamion. Uzmimo za primjer kamion kapaciteta 20, dodavanjem jednog punog kontejnera na popis kapacitet se smanjuje na 19, a dodavanjem slijedećeg prepunog kontejnera smanjit će se za 2, odnosno na 20 te će ispisati onoliko kontejnera koliko stane u taj kamion.



Slika 3.5: Sekvencijski dijagram za UC05

Obrazac uporabe 08 - Premještanje komunalnog radnika

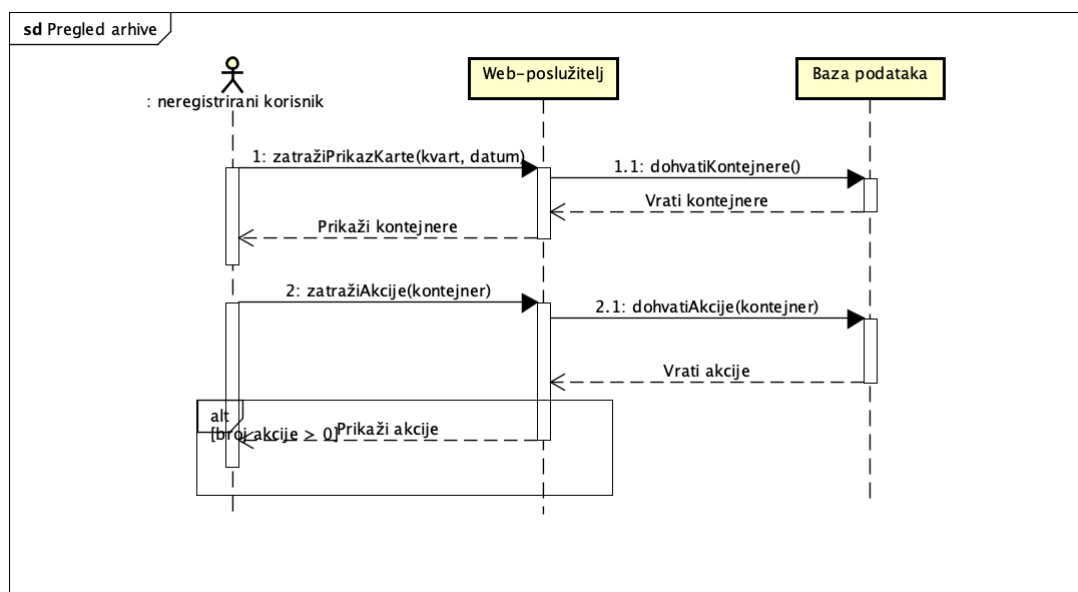
Kako bi izvršio premještanje radnika iz jednog kvartovskog poduzeća u drugo administrator mora poduzeti slijedeće; administrator bira kvart kojem je dotični radnik trenutno pridjeljen te traži u njemu istog radnika. Poslužitelj mu izlistava popis svih djelatnika u tom poduzeću te administrator bira traženog djelatnika. Poslužitelj mu izlistava informacije o tom djelatniku te mogućnosti s istim. Administrator bira premještanje te unosi željeni odredišni kvart. Ukoliko u trenutnom kvartovskom poduzeću tom akcijom neće biti premalo radnika niti u odredišnom previše poslužitelj obavlja zahtjev te dojavljuje korisniku informaciju o uspješnom izvođenju zahtjeva. U suprotnom premještanje se neće izvršiti.



Slika 3.6: Sekvencijski dijagram za UC08

Obrazac uporabe 13 - Pregled arhive

Kako bi neregistrirani korisnik mogao pregledavati arhivu mora prvotno unijeti kvart za kojeg želi pregledati arhivu i datum. Poslužitelj mu vraća prikaz karte sa kontejnerima koji su se u tom trenutku nalazili na tom kvartu. Korisnik može odabrati pojedinu kantu te će mu poslužitelj prikazati popis svih prijava i pražnjenja tog kontejnera toga dana.



Slika 3.7: Sekvencijski dijagram za UC13

3.2 Ostali zahtjevi

- Sustav treba omogućiti rad više korisnika u stvarnom vremenu
- Sustav i korisničko sučelje trebaju podržavati dijakritičke znakove
- Sustav treba biti implementiran kao web aplikacija koristeći objektno-orijentirane jezike
- Pristup sustavu mora biti omogućen iz javne mreže pomoću HTTPS-a
- Veza uspostavljena s bazom podataka treba biti sigurna, brza i otporna na vanjske greške
- Transakcije nad bazom podataka moraju imati ograničeno trajanje
- Nadogradnja sustava ne smije narušavati postojeće funkcionalnosti sustava
- Sustav treba biti jednostavan i praktičan za korištenje
- Pogreške pri radu korisnika u korisničkom sučelju ne smiju utjecati na ispravan rad sustava.

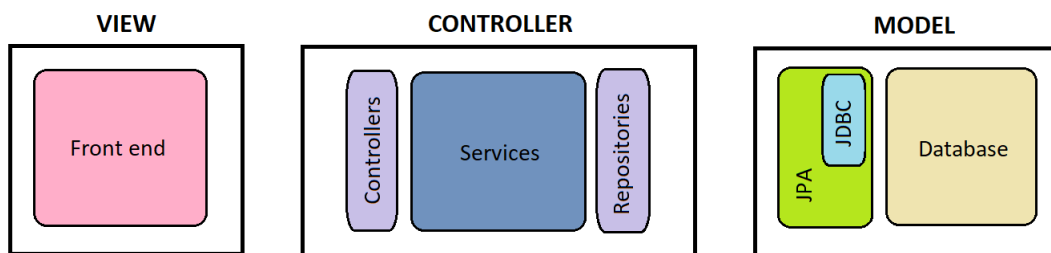
4. Arhitektura i dizajn sustava

dio 1. revizije

Za izradu aplikacije na strani *back end-a* odabrali smo programski jezik Java, koristeći se pritom Spring Boot radnim okvirom koji uvelike olakšava razvoj web aplikacija. Radi lakšeg uključivanja vanjskih biblioteka te radi postizanja neovisnosti o razvojnom okruženju koristimo Maven.

Na strani *front end-a* koristimo radni okvir Bootstrap, pomoću kojeg koristimo standardne jezike za dizajniranje i ponašanje web stranica na strani klijenta - HTML, CSS i JavaScript. Radi jednostavnijeg pisanja, uz *vanilla JavaScript* koristimo i *jQuery* biblioteku, kojom je uvelike olakšano slanje asinkronih (AJAX) zahtjeva na web poslužitelj.

Arhitektura sustava se temelji na konceptu MVC (*Model-View-Controller*), kako bi se što jasnije i smislenije razdvojili pojedini dijelovi aplikacije te omogućio njihov istodoban i neovisan razvoj.



Slika 4.1: MVC prikaz aplikacije

View označava podsustav zadužen za *user-friendly* prikaz podataka korisniku. U slučaju naše aplikacije, razvija se potpuno neovisno od ostalih dijelova, te je pokrenut na zasebnom poslužitelju. Osim što prikazuje *model*, *view* je zadužen i za slanje zahtjeva *controller-u*.

Controller označava podsustav koji upravlja zahtjevima pristiglim iz *view-a*, do-

hvaća stvarne podatke iz *model-a* te vrši logiku svih mogućih akcija i događaja. Središnji dio *controller-a* predstavljaju servisi, koji se u samoj aplikaciji nalaze u posebnom paketu, a koji obavljaju cjelokupnu poslovnu logiku aplikacije. Kako bi to obavljali uspješno, moraju moći komunicirati sa ostalim podsustavima.

Komunikacija s *view-om* se izvodi putem sučelja *Controllers*, koje na slici predstavlja cijeli paket u kojem je definiran API (*Application Programming Interface*) putem kojeg se aplikaciji šalju zahtjevi ili se dohvaćaju podaci. API koji smo odabrali je REST (*REpresentational State Transfer*) API, kako bi se omogućila lakša komunikacija sa drugim sustavima, te kako bi aplikacija bila lako proširiva (primjerice, na isti API se može spojiti i mobilna aplikacija).

Controller sa modelom komunicira putem sučelja *Repositories* koje također predstavlja cijeli paket kojim se iz baze podataka dohvaćaju konkretne objekte i podatke.

Model predstavlja sve strukture podataka, objekte i zapise vezane uz aplikaciju, koji su spremljeni u bazi podataka. Komunikacija s bazom je ostvarena putem JPA/JDBC specifikacije.

4.1 Baza podataka

Kao sustav upravljanja bazom podataka koristimo PostgreSQL *Sustav za Upravljanje Bazom Podataka* (SUBP). Na bazu se spajamo putem *Java DataBase Connectivity* (JDBC) specifikacije, koja omogućuje Java programima spajanje na SUBP i njegovo korištenje. Na JDBC specifikaciju nadograđuje se *Java Persistence API* (JPA) koji pruža usluge *Object-Relational Mapping* (ORM), čime se odabrani Java objekti automatski spremaju u bazu podataka.

4.1.1 Opis tablica

U bazi podataka nalazi se 9 relacija:

- **Persons** - relacija koja opisuje osobu u najširem smislu; osoba može biti korisnik aplikacije, komunalni radnik ili administrator sustava, te se u ovoj relaciji spremaju zajednički atributi tih aktera
- **Admins** - relacija u kojoj se nalaze zapisi svih administratora
- **Employees** - relacija u kojoj se nalaze zapisi svih komunalnih radnika

- **Citizens** - relacija u kojoj se nalaze zapisi svih građana - regularnih korisnika aplikacije
- **Container** - relacija u kojoj su spremljeni zapisi o kontejnerima
- **Neighborhoods** - relacija u kojoj su spremljeni zapisi o susjedstvima
- **Favorites** - relacija u kojoj su spremljeni zapisi o parovima Person-Container, sa značenjem da je osoba Person spremila kontejner Container radi brzog pristupa.
- **Pings** - relacija u kojoj su spremljeni zapisi o parovima Person-Container, sa značenjem da je osoba Person prijavila kontejner Container kao pun, pretrpan ili prazan.
- **Emptyings** - relacija u kojoj su spremljeni zapisi o pražnjenjima kontejnera oblika Employee-Container, sa značenjem da je radnik Employee ispraznio kontejner Container

Tablica Person je generička tablica za osobu, a nju nasljeđuju tablice Admin, Employee i Citizen, svaki sa svojim dodatnim atributima te stranim ključem koji pokazuje na tablicu Person gdje se nalaze zajednički atributi svojstveni svima trima entitetima. Tablice Favorites, Pings i Emptyings su tablice koje postoje kako bi se opisala N-N veza između relacija koje povezuju.

4.1.2 Definicije tablica

Person		
id	BIGINT	Identifikator osobe; svaka osoba ima svoj jedinstveni ID
name	VARCHAR	Ime osobe, ne smije biti NULL
last_name	VARCHAR	Prezime osobe, ne smije biti NULL
email	VARCHAR	E-mail osobe, koristi se pri prijavi korisnika u sustav, ne smije biti NULL i jedinstvena je vrijednost
pwd_hash	VARCHAR	Sažetak lozinke koju je osoba odabrala

Tablica 4.1: Tablica *Person*

Admin		
id	BIGINT	Jedini atribut tablice - strani ključ koji pokazuje na tablicu Person.

Tablica 4.2: Tablica *Admin*

Citizen		
id	BIGINT	Strani ključ koji pokazuje na tablicu Person.
reputation	INT	Atribut koji označava reputaciju korisnika; služi za procjenu vjerodostojnosti njegove prijave.

Tablica 4.3: Tablica *Citizen*

Employee		
id	BIGINT	Strani ključ koji pokazuje na tablicu Person.
oib	VARCHAR(11)	OIB komunalnog radnika.
neighborhood_id	BIGINT	Strani ključ koji pokazuje na tablicu Neighborhood. Određuje kojem kvartu pripada radnik.

Tablica 4.4: Tablica *Employee*

Container		
id	BIGINT	Identifikator kontejnera; svaki kontejner ima svoj jedinstveni ID
latitude	DOUBLE	Geografska širina lokacije kontejnera
longitude	DOUBLE	Geografska dužina lokacije kontejnera
pings_since_emptied	INT	Broj prijava kontejnera od njegovog zadnjeg pražnjenja
neighborhood_id	BIGINT	Strani ključ koji pokazuje na tablicu Neighborhood. Određuje u kojem se kvartu nalazi kontejner.

Tablica 4.5: Tablica *Container*

Neighborhood		
id	BIGINT	Identifikator susjedstva; svako susjedstvo (kvart) ima svoj jedinstveni ID
latitude	DOUBLE	Geografska širina lokacije komunalnog središta u susjedstvu
longitude	DOUBLE	Geografska dužina lokacije komunalnog središta u susjedstvu
name	VARCHAR	Ime susjedstva

Tablica 4.6: Tablica *Neighborhood*

Favorites		
id	BIGINT	Identifikator oznake favorita; svako označavanje ima svoj jedinstveni ID.
container_id	BIGINT	Strani ključ koji pokazuje na relaciju <i>Container</i>
owner_id	BIGINT	Strani ključ koji pokazuje na relaciju <i>Person</i>

Tablica 4.7: Tablica *Favorites*

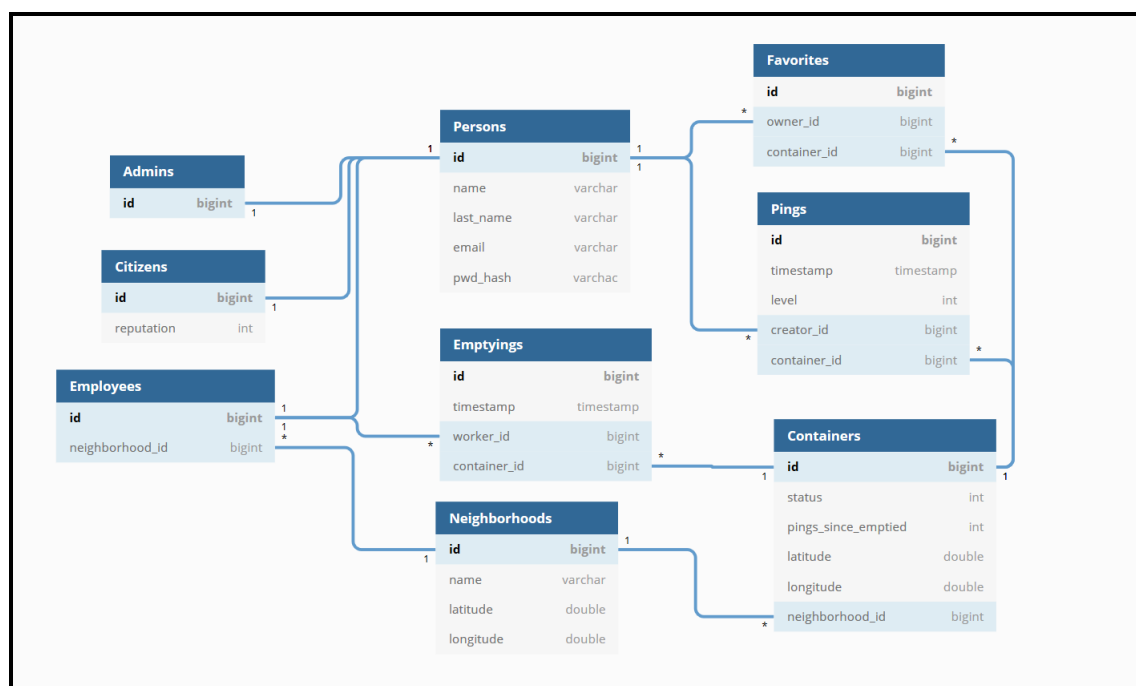
Pings		
id	BIGINT	Identifikator prijave; svako označavanje ima svoj jedinstveni ID.
level	INT	Level koji označava vrstu prijave: 0 je prijava praznog, 1 prijava punog i 2 prijava pretrpanog kontejnera
timestamp	BIGINT	Vremenska oznaka trenutka u kojem se dogodila prijava kontejnera
photo_path	VARCHAR	Mjesto na disku na kojem se nalazi fotografija koju je priložio korisnik
container_id	BIGINT	Strani ključ koji pokazuje na relaciju <i>Container</i>
owner_id	BIGINT	Strani ključ koji pokazuje na relaciju <i>Person</i>

Tablica 4.8: Tablica *Ping*

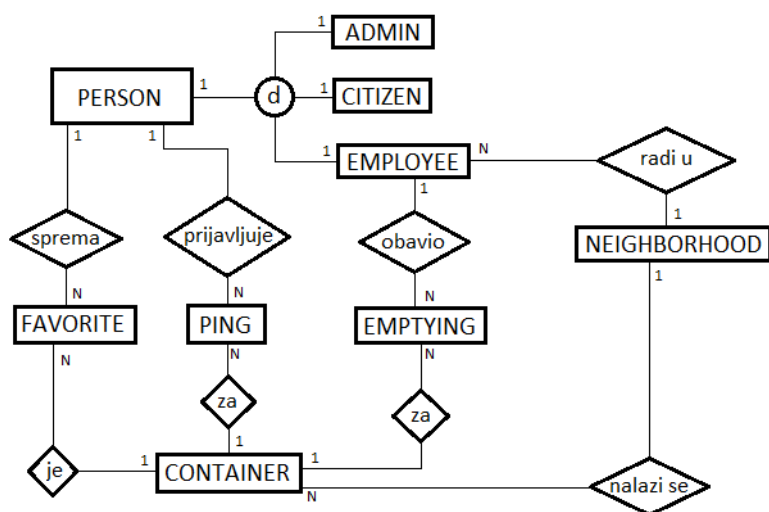
Emptyings		
id	BIGINT	Identifikator prijave; svako označavanje ima svoj jedinstveni ID.
timestamp	BIGINT	Vremenska oznaka trenutka u kojem se dogodila prijava kontejnera
container_id	BIGINT	Strani ključ koji pokazuje na relaciju Container
worker_id	BIGINT	Strani ključ koji pokazuje na relaciju Employee

Tablica 4.9: Tablica *Emptying*

4.1.3 Dijagram baze podataka



Slika 4.2: Dijagram baze podataka



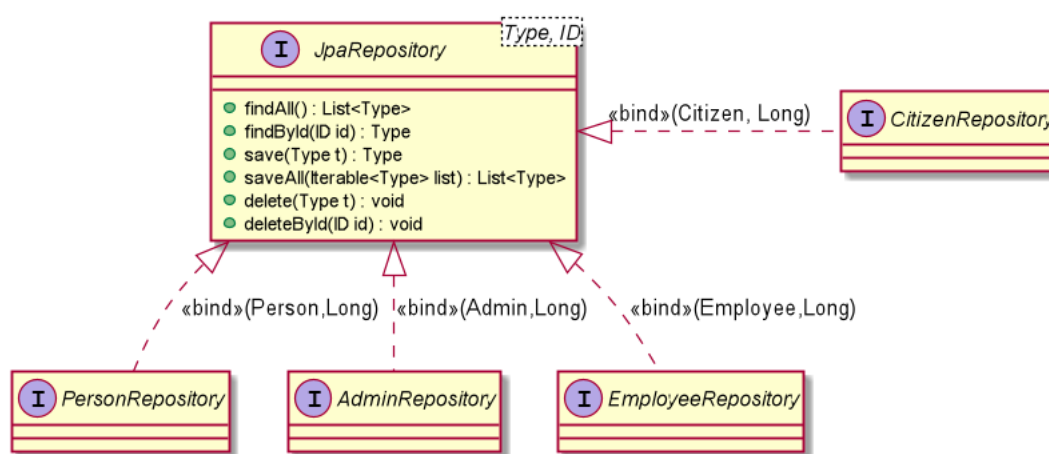
Slika 4.3: Entitetsko-Relacijski model baze podataka

4.2 Dijagram razreda

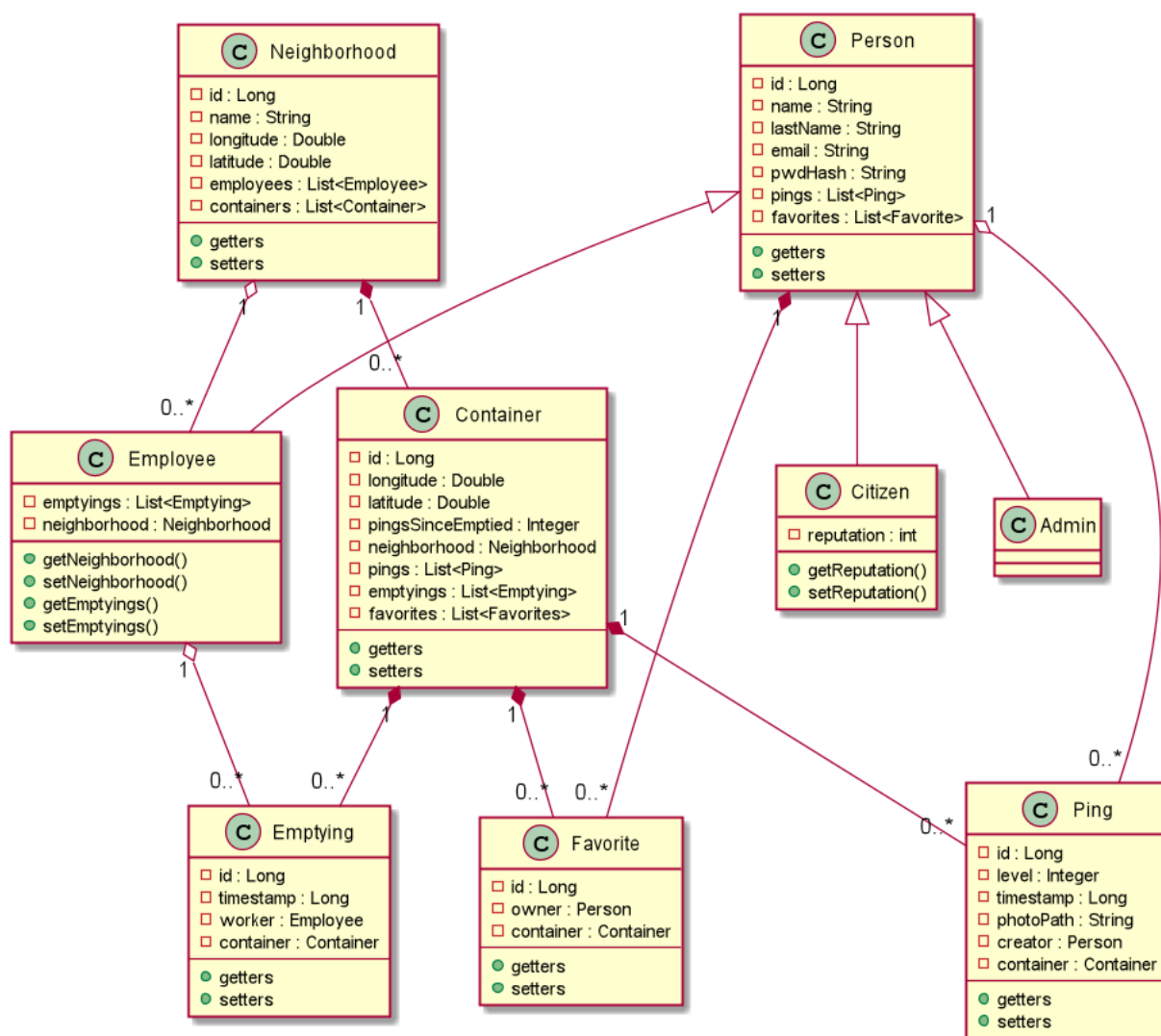
Na slici 4.4. prikazan je dijagram razreda paketa DAO. DAO (*Data Access Object*) je paket koji predstavlja sloj između logike aplikacije i baze podataka. Pomoću DAO objekata aplikacija dohvaća objekte iz baze podataka te ih uključuje u razne akcije, mijenja, stvara nove objekte vezane uz njih i sprema ih natrag u bazu. Sve metode su već implementirane u Spring-ovom sučelju *JpaRepository*, koje sva naša sučelja nasljeđuju.

Slika 4.5 prikazuje dijagram razreda koji predstavljaju model podataka - reprezentaciju stvarnog svijeta. On je vrlo sličan ER modelu i dijagramu baze podataka, ali ne i identičan: uspoređujući ih, vidljivo je kako se Java objekti i sve veze (One-to-one, One-to-many, Many-to-many) mapiraju u bazu podataka.

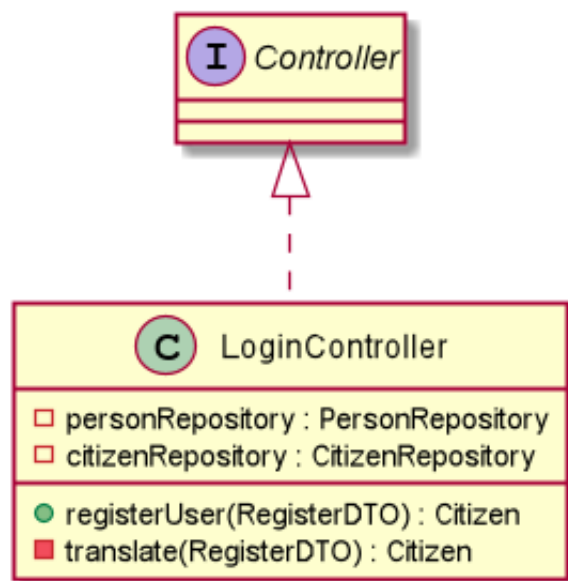
Konačno, slika 4.6 prikazuje dijagram razreda *controller-a*, koji su zaduženi za primanje HTTP zahtjeva na način da se svaka metoda u pojedinom *controller* razredu mapira na određeni URL i određenu HTTP metodu (GET, POST, PUT...). Tako je primjerice metoda *registerUser()* u razredu *LoginController* pozvana isključivo ako korisnik pošalje POST zahtjev na URL `"/register"`.



Slika 4.4: Dijagram razreda paketa DAO



Slika 4.5: Dijagram razreda koji predstavljaju model podataka



Slika 4.6: Dijagram razreda - controllera

dio 2. revizije

Prilikom druge predaje projekta dijagram razreda i opisi moraju odgovarati stvarnom stanju implementacije

4.3 Dijagram stanja

dio 2. revizije

*Potrebno je priložiti dijagram stanja i opisati ga. Dovoljan je jedan dijagram stanja koji prikazuje **značajan dio funkcionalnosti** sustava. Na primjer, stanja korisničkog sučelja i tijekom korištenja neke ključne funkcionalnosti jesu značajan dio sustava, a registracija i prijava nisu.*

4.4 Dijagram aktivnosti

dio 2. revizije

Potrebno je priložiti dijagram aktivnosti s pripadajućim opisom. Dijagram aktivnosti treba prikazivati značajan dio sustava.

4.5 Dijagram komponenti

dio 2. revizije

Potrebno je priložiti dijagram komponenti s pripadajućim opisom. Dijagram komponenti treba prikazivati strukturu cijele aplikacije.

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

dio 2. revizije

*Detaljno navesti sve tehnologije i alate koji su primijenjeni pri izradi dokumentacije i aplikacije. Ukratko ih opisati, te navesti njihovo značenje i mjesto primjene. Za svaki navedeni alat i tehnologiju je potrebno **navesti internet poveznicu** gdje se mogu preuzeti ili više saznati o njima.*

5.2 Ispitivanje programskog rješenja

dio 2. revizije

U ovom poglavlju je potrebno opisati provedbu ispitivanja implementiranih funkcionalnosti na razini komponenti i na razini cijelog sustava s prikazom odabranih ispitnih slučajeva. Studenti trebaju ispitati temeljnu funkcionalnost i rubne uvjete.

5.2.1 Ispitivanje komponenti

*Potrebno je provesti ispitivanje jedinica (engl. unit testing) nad razredima koji implementiraju temeljne funkcionalnosti. Razraditi **minimalno 6 ispitnih slučajeva** u kojima će se ispitati redovni slučajevi, rubni uvjeti te izazivanje pogreške (engl. exception throwing). Poželjno je stvoriti i ispitni slučaj koji koristi funkcionalnosti koje nisu implementirane. Potrebno je priložiti izvorni kôd svih ispitnih slučajeva te prikaz rezultata izvođenja ispita u razvojnom okruženju (prolaz/pad ispita).*

5.2.2 Ispitivanje sustava

*Potrebno je provesti i opisati ispitivanje sustava koristeći radni okvir Selenium¹. Razraditi **minimalno 4 ispitna slučaja** u kojima će se ispitati redovni slučajevi, rubni uvjeti te poziv funkcionalnosti koja nije implementirana/izaziva pogrešku kako bi se vidjelo na koji način sustav reagira kada nešto nije u potpunosti ostvareno. Ispitni slučaj se treba sastojati od ulaza (npr. korisničko ime i lozinka), očekivanog izlaza ili rezultata, koraka ispitivanja i dobivenog izlaza ili rezultata.*

Izradu ispitnih slučajeva pomoću radnog okvira Selenium moguće je provesti pomoću jednog od sljedeća dva alata:

- *dodatak za preglednik **Selenium IDE** - snimanje korisnikovih akcija radi automatskog ponavljanja ispita*
- ***Selenium WebDriver** - podrška za pisanje ispita u jezicima Java, C#, PHP koristeći posebno programsko sučelje.*

Detalji o korištenju alata Selenium bit će prikazani na posebnom predavanju tijekom semestra.

¹<https://www.seleniumhq.org/>

5.3 Dijagram razmještaja

dio 2. revizije

*Potrebno je umetnuti **specifikacijski** dijagram razmještaja i opisati ga. Moguće je umjesto specifikacijskog dijagrama razmještaja umetnuti dijagram razmještaja instanci, pod uvjetom da taj dijagram bolje opisuje neki važniji dio sustava.*

5.4 Upute za puštanje u pogon

dio 2. revizije

*U ovom poglavlju potrebno je dati upute za puštanje u pogon (engl. deployment) ostvarene aplikacije. Na primjer, za web aplikacije, opisati postupak kojim se od izvornog kôda dolazi do potpuno postavljene baze podataka i poslužitelja koji odgovara na upite korisnika. Za mobilnu aplikaciju, postupak kojim se aplikacija izgradi, te postavi na neku od trgovina. Za stolnu (engl. desktop) aplikaciju, postupak kojim se aplikacija instalira na računalo. Ukoliko mobilne i stolne aplikacije komuniciraju s poslužiteljem i/ili bazom podataka, opisati i postupak njihovog postavljanja. Pri izradi uputa preporučuje se **naglasiti korake instalacije uporabom natuknica** te koristiti što je više moguće **slike ekrana** (engl. screenshots) kako bi upute bile jasne i jednostavne za slijediti.*

Dovršenu aplikaciju potrebno je pokrenuti na javno dostupnom poslužitelju. Studentima se preporuča korištenje neke od sljedećih besplatnih usluga: Amazon AWS, Microsoft Azure ili Heroku. Mobilne aplikacije trebaju biti objavljene na F-Droid, Google Play ili Amazon App trgovini.

6. Zaključak i budući rad

dio 2. revizije

U ovom poglavlju potrebno je napisati osvrt na vrijeme izrade projektnog zadatka, koji su tehnički izazovi prepoznati, jesu li riješeni ili kako bi mogli biti riješeni, koja su znanja stečena pri izradi projekta, koja bi znanja bila posebno potrebna za brže i kvalitetnije ostvarenje projekta i koje bi bile perspektive za nastavak rada u projektnoj grupi.

Potrebno je točno popisati funkcionalnosti koje nisu implementirane u ostvarenoj aplikaciji.

Popis literature

Kontinuirano osvježavanje

Popisati sve reference i literaturu koja je pomogla pri ostvarivanju projekta.

1. Oblikovanje programske potpore, FER ZEMRIS, <http://www.fer.hr/predmet/opp>
2. I. Sommerville, "Software engineering", 8th ed, Addison Wesley, 2007.
3. T.C.Lethbridge, R.Langaniere, "Object-Oriented Software Engineering", 2nd ed. McGraw-Hill, 2005.
4. I. Marsic, Software engineering book", Department of Electrical and Computer Engineering, Rutgers University, <http://www.ece.rutgers.edu/~marsic/books/SE>
5. The Unified Modeling Language, <https://www.uml-diagrams.org/>
6. Astah Community, <http://astah.net/editions/uml-new>

Indeks slika i dijagrama

3.1	Dijagram obrasca uporabe, funkcionalnost administratora	16
3.2	Dijagram obrasca uporabe, funkcionalnost komunalnog radnika . .	17
3.3	Dijagram obrasca uporabe, funkcionalnost građanina	17
3.4	Dijagram obrasca uporabe, funkcionalnost neregistriranog korisnika	18
3.5	Sekvencijski dijagram za UC05	19
3.6	Sekvencijski dijagram za UC08	20
3.7	Sekvencijski dijagram za UC13	21
4.1	Dijagram baze podataka	27
4.2	Entitetsko-Relacijski model baze podataka	27

Indeks tablica

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

Kontinuirano osvježavanje

1. sastanak

- Datum: 14. listopada 2019.
- Prisustvovali: M. Bićanić, H. Hrvoj, S. Skender, M. Tropčić, M. Turina, M. Vasilj
- Teme sastanka:
 - Diskusija o projektnom zadatku
 - Definiranje aktera u projektu
 - Definiranje osnovnog modela baze podataka
 - Rasprava o tehnologijama Spring, JPA
 - Općenita raspodjela studenata u front-end, back-end, dev-ops i dokumentacijske pod-timove

2. sastanak

- Datum: 21. listopada 2019.
- Prisustvovali: M. Bićanić, H. Hrvoj, S. Skender, M. Turina, M. Vasilj
- Teme sastanka:
 - Rasprava o dosad napravljenom dijelu projekta
 - Dodjela konkretnih zadataka pojedinim članovima
 - Dogovor o organizaciji same aplikacije
 - Rasprava o dokumentaciji projekta

3. sastanak

- Datum: 30. listopada 2019.
- Prisustvovali: M. Bićanić, S. Skender, M. Tropčić, M. Vasilj
- Teme sastanka:
 - Redefiniranje baze podataka

- Rasprava o previđenim funkcionalnostima te dogovor oko implementacije
- Određen i specificiran REST API aplikacije

Tablica aktivnosti

Kontinuirano osvježavanje

	Sven Skender	Miroslav Bićanić	Hrvoje Hrvoj	Mario Tropčić	Marko Turina	Matea Vasilj				
Upravljanje projektom			10		5					5
Opis projektnog zadatka					3					
Funkcionalni zahtjevi										3
Opis pojedinih obrazaca										3
Dijagram obrazaca										
Sekvencijski dijagrami										
Opis ostalih zahtjeva										
Arhitektura i dizajn sustava										
Baza podataka					2					
Dijagram razreda										
Dijagram stanja										
Dijagram aktivnosti										
Dijagram komponenti										
Korištene tehnologije i alati										
Ispitivanje programskog rješenja										
Dijagram razmještaja										
Upute za puštanje u pogon										
Dnevnik sastajanja					1					
Zaključak i budući rad										
Popis literature										
<i>Dodatne stavke kako ste podijelili izradu aplikacije</i>										
<i>npr. izrada početne stranice</i>										
<i>izrada baze podataka</i>										

	Sven Skender	Miroslav Bićanić	Hrvoje Hrvoj	Mario Tropčić	Marko Turina	Matea Vasilj						
<i>spajanje s bazom podataka</i>												
<i>back end</i>												

Dijagrami pregleda promjena

dio 2. revizije

Prenijeti dijagram pregleda promjena nad datotekama projekta. Potrebno je na kraju projekta generirane grafove s gitlaba prenijeti u ovo poglavlje dokumentacije. Dijagrami za vlastiti projekt se mogu preuzeti s gitlab.com stranice, u izborniku Repository, pritiskom na stavku Contributors.