Oblikovanje programske potpore

Ak. god. 2018./2019.

*Moj Kvart*

Dokumentacija, Rev. *<1 ili 2>*

Grupa: *Javoljupci*

Voditelj: *Mate Gašparini*

Datum predaje: *<dan>. <mjeseca> <godine>.*

Nastavnik: *Nikolina Frid*

Sadržaj

[1.](#_2grqrue) Dnevnik promjena dokumentacije 3

[2.](#_1fob9te) Opis projektnog zadatka 5

[3.](#_2et92p0) Pojmovnik 6

[4.](#_3dy6vkm) Funkcionalni zahtjevi 7

[5.](#_4d34og8) Ostali zahtjevi 8

[6.](#_17dp8vu) Arhitektura i dizajn sustava 9

[6.1.](#_26in1rg) Svrha, opći prioriteti i skica sustava 10

[6.2.](#_35nkun2) Dijagram razreda s opisom 11

[6.3.](#_44sinio) Dijagram objekata 12

[6.4.](#_z337ya) Ostali UML dijagrami 13

[7.](#_1y810tw) Implementacija i korisničko sučelje 14

[7.1.](#_2xcytpi) Dijagram razmještaja 14

[7.2.](#_3whwml4) Korištene tehnologije i alati 15

[7.3.](#_vx1227) Isječak programskog koda vezan za temeljnu funkcionalnost sustava 16

[7.4.](#_3fwokq0) Ispitivanje programskog rješenja 17

[7.5.](#_1v1yuxt) Upute za instalaciju 18

[7.6.](#_4f1mdlm) Korisničke upute 19

[8.](#_2p2csry) Zaključak i budući rad 20

[9.](#_3o7alnk) Popis literature 21

[Dodatak A: Indeks (slika, dijagrama, tablica, ispisa kôda) 22](#_ihv636)

[Dodatak B: Dnevnik sastajanja 23](#_32hioqz)

[Dodatak C: Prikaz aktivnosti grupe 24](#_1hmsyys)

[Dodatak D: Plan rada / Pregled rada i stanje ostvarenja 26](#_41mghml)

*Sadržaj bi se trebao automatski osvježavati prema tekstu (desni klik, „Update Field“) ako se bude držalo zadanih formata poglavlja.*

# Dnevnik promjena dokumentacije

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev.** | **Opis promjene/dodatka** | **Autor(i)** | **Datum** |
| 0.1 | Popisani dionici, razrađeni aktori i njihovi funkcionalni zahtjevi te napravljeno prvih sedam use-caseova. | Assouline, Dukić | 22.10.2018. |
| 0.2 | Napisan opis projektnog zadatka. | Dodigović | 22.10.2018. |
| 0.21 | Dodani use-caseovi od osmog do dvadesetog. | Dukić, Assouline | 23.10.2018. |
| 0.22 | Poboljšani obrasci uporabe. | Dukić, Assouline | 23.10.2018. |
| 0.23 | Sistematizirani obrasci uporabe. | Dukić | 26.10.2018. |
| 0.3 | Dodani dijagrami obrazaca uporabe. | Kršić, Kurtović | 27.10.2018. |
| 0.31 | Dodani sekvencijski dijagrami. | Kršić, Kurtović | 27.10.2018. |
| 0.32 | Obrisane funkcionalnosti vezane uz sustav poruka kod obrazaca uporabe. | Dukić | 28.10.2018. |
| 0.33 | Izmijenjeni dijagrami obrazaca uporabe i dodan sekvencijski dijagram. | Kršić | 3.11.2018. |
| 0.34 | Dodani dionici, poboljšane definicije funkcionalnih zahtjeva. | Assouline, Dukić | 3.11.2018. |
| 0.35 | Kombiniranje sekvencijskih dijagramima. | Kurtović | 5.11.2018. |
| 0.4 | Dodani ostali zahtjevi. | Gašparini | 5.11.2018. |
| 0.5 | Dodana skica arhitekture sustava. | Dodigović | 5.11.2018. |
| 0.6 | Dodana skica baze podataka. | Gašparini | 15.11.2018. |
| 0.61 | Dodan opis baze podataka. | Kurtović | 22.11.2018. |
| 0.7 | Dodan dijagram razreda. | Dodigović | 23.11.2018. |
| 0.71 | Dodan dijagram objekata. | Kršić | 24.11.2018. |
| 0.8 | Sitne izmjene baze podataka i oblikovanja. | Gašparini | 24.11.2018. |
| 0.81 | Izmjena dijagrama razreda. | Gašparini | 26.11.2018. |
|  |  |  |  |

*Moraju postojati glavne revizije (verzije, inačice) dokumenata 1.0 i 2.0 na kraju prvog i drugog ciklusa. Između tih revizija mogu postojati manje revizije već prema tome kako se dokument bude nadopunjavao. Očekuje se da nakon svake značajnije promjene (dodatka, izmjene, uklanjanja dijelova teksta i popratnih grafičkih sadržaja) dokumenta se to zabilježi kao revizija. Npr., revizije unutar prvog ciklusa će imati oznake 0.1, 0.2, …, 0.9, (ukoliko bude više od toga nastavlja se sa 0.10, 0.11…) sve do konačne revizije prvog ciklusa 1.0. U drugom ciklusu se nastavlja s revizijama 1.1, 1.2, itd. Ukoliko se želi naglasiti veličina promjene/dodatka unutar revizije može se napraviti različita „granulacija“ u označavanju revizija. Npr. nakon 0.1 može biti 0.11, 0.12, 0.15 pa tek onda 0.2. Ili se za veće promjene/zahvate može nakon 0.1 odmah skočiti na 0.5. itd. Bitno je da na krajevima ciklusa revizije koje se predaju budu s oznakama* ***1.0*** *i* ***2.0*** *(Preporuka je da te revizije budu podebljane unutar liste).*

# Opis projektnog zadatka

Cilj ovog projekta je izrada web aplikacije koja će stanovnicima unutar odabrane gradske četvrti omogućiti bolju komunikaciju, lakšu razmjenu informacija te učinkovitiju organizaciju događaja.

Iako ljudi na dnevnoj razini prolaze pokraj svojih susjeda često ih niti ne upoznaju te time propuštaju priliku za moguću suradnju, druženje ili razmjenu korisnih informacija. Osim društvenog kontakta, stanovnike četvrti vežu stvari koje se tiču svakog pojedinačno. Samo neki od primjera su izvješće sa sjednice Vijeća četvrti, uređenje kvartovske okoline, mogući problemi oko zbrinjavanje otpada ili slično. Također, web aplikacija je prikladno mjesto za početak ostvarivanja odnosa na prethodno opisanim razinama zato što se ljudi zainteresirani za ostvarivanje komunikacije ujedinjuju na jednom mjestu i time si daju mogućnost za daljnji napredak u odnosima u stvarnom životu.

Stanovnik četvrti pri otvaranju računa upisuje osnovne informacije na temelju kojih se stvara njegov korisnički račun. Informacije potrebne za registraciju su:

* Ime
* Prezime
* Lozinka
* Adresa stanovanja
* Adresa e-pošte

Na temelju tih informacija aplikacija prema adresi stanovanja automatski određuje kojoj četvrti korisnik pripada i dodjeljuje mu se uloga stanovnika četvrti. Ostali korisnici kojima je dodijeljena viša uloga se pri otvaranju računa ne registriraju nego za njih račune stvaraju postojeći administratori sustava. Registracija je za sve korisnike, neovisno o ulozi, besplatna.

Da bi aplikacija zadovoljila sve funkcionalne zahtjeve koje četvrt zahtjeva, ona je sastavljena od tri osnovne cjeline. Prva cjelina je forum čija je svrha održavanje rasprava koje su tematski vezane uz aktualna događanja i sam život stanovnika u četvrti, sljedeća cjelina su događanja koja najavljuju buduće događaje u četvrti te posljednja, vijeće četvrti gdje se u kratkim natuknicama objavljuje sadržaj sjednica Vijeća gradske četvrti uz najavu novih tema.

Aplikacija razlikuje četiri vrste korisnika:

* Stanovnik četvrti
* Vijećnik
* Moderator četvrti
* Administrator aplikacije

Stanovniku četvrti je omogućeno vođenje rasprava unutar foruma i predlaganje najave događaja u četvrti. Pri definiranju prijedloga potrebno je definirati naziv, mjesto, vrijeme trajanja te kratak opis događaja.

Vijećnik uz mogućnosti redovnog stanovnika ima ovlasti pisanja objava u cjelini "Vijeće četvrti" gdje piše kratko izvješće sa zadnje sjednice.

Moderatori četvrti također imaju mogućnosti redovnih stanovnika, ali uz to pregledavaju najavljene događaje nakon čega ih odbijaju ili potvrđuju. Događaji se odbijaju ako sadrže vrijeđanje ili diskriminaciju na bilo kojoj razini. Potvrđene događaje objavljuju u rubrici "Događaji". Na raspolaganju im je i uređivanje najava zbog mogućih jezičnih pogrešaka. Osim toga, unutar foruma im je omogućeno brisanje objava neprikladnog sadržaja.

Administratori se ne smatraju stanovnicama niti jedne četvrti. Njihov je zadatak tehničke naravi te su oni zaduženi za upravljanje korisničkim računima, definiranje četvrti i njihovih granica.

Skup korisnika koji bi mogli biti zainteresirani za razvijeni sustav su stanovnici četvrti koji bi željeli uvesti neke promjene u svoju četvrt jer im se na ovaj način pruža lakši i učinkovitiji način diskusije, oni koji bi jednostavno htjeli poboljšati društvene odnose na razini kvarta i na kraju, oni koje zanima brži i lakši protok informacija vezanih uz život u kvartu.

# Pojmovnik

*Lista pojmova (napisanih abecednim redom) potrebnih za razumijevanje teksta dokumentacije. Svaki pojam treba biti dosljedno opisan u jednoj ili dvije rečenice da se izbjegnu nejasnoće u specifikaciji sustava i opisu implementacije i rezultata.*

# Funkcionalni zahtjevi

**Dionici:**

* Stanovnik četvrti
* Vijećnik
* Razvojni programer
* Naručitelj proizvoda (FER)

**Aktori i njihovi funkcionalni zahtjevi:**

Aktori mogu biti inicijatori i sudionici. Inicijatori su stanovnici četvrti, vijećnici, moderatori četvrti i administratori aplikacije. Sudionik je baza podataka.

* **Stanovnik četvrti**
  + Može se registrirati
  + Može diskutirati o temama na forumu
  + Može predlagati najave budućih događanja u susjedstvu
* **Vijećnik**
  + Ima sve ovlasti kao i stanovnik četvrti (osim registracije)
  + Ima ovlast pisanja objava u cjelini „Vijeće četvrti“
* **Moderator četvrti**
  + Ima sve ovlasti kao i stanovnik četvrti (osim registracije)
  + Može pregledavati najave događanja koje su predložili stanovnici četvrti i objaviti ih u cjelini „Događanja“
  + Može uređivati prijedloge stanovnika četvrti u cjelini „Događanja“
  + Može odbaciti prijedloge neprikladnog sadržaja iz cjeline „Događanja“
  + Može ukloniti objave neprikladnog sadržaja s foruma
* **Administrator aplikacije**
  + Otvara račune za vijećnike, moderatore i druge administratore
  + Upravlja korisničkim računima
  + Definira četvrti i područja koja one obuhvaćaju
* **Baza podataka**
  + Pohranjuje podatke o svim korisnicima
  + Pohranjuje podatke o četvrtima i ulicama od kojih se one sastoje
  + Pohranjuje objave s foruma
  + Pohranjuje objave vijećnika gradske četvrti iz cjeline „Vijeće četvrti“
  + Pohranjuje najave iz cjeline „Događanja“

**UC1** Registracija

**Glavni sudionik** Stanovnik četvrti

**Rezultat** Stvaranje novog korisničkog računa za stanovnika četvrti

**Ostali sudionici** Baza podataka

**Željeni scenarij** 1. Korisnik upisuje svoje ime, prezime, željeno korisničko ime,

lozinku i adresu stanovanja (ulica u kojoj stanuje)

                    2. Koristeći unesene podatke, sustav stvara novi korisnički račun

3. Podaci o novostvorenom korisniku dodaju se u bazu podataka

**Mogući scenarij** 1. Za unesenu e-mail adresu već postoji korisnički račun

2. Sustav odbacuje pokušaj registracije i javlja grešku korisniku

**UC2**                Otvaranje povlaštenih korisničkih računa

**Glavni sudionik** Administrator

**Rezultat** Administrator stvara korisnički račun za vijećnika, moderatora ili

drugog administratora; ovaj postupak ima isti rezultat kao i registracija kod stanovnika četvrti

**Ostali sudionici** Vijećnik, moderator, baza podataka

**Preduvjeti** 1. Administrator je prijavljen u sustav

**Željeni scenarij** 1. Administrator popunjava obrazac za dodavanje novog

korisničkog računa

2. Pritiskom na gumb „Stvori korisnika” novi korisnik se stvara u

bazi podataka s ulogom koju mu je dodijelio administrator

**Mogući scenarij** 1. Za unesenu e-mail adresu već postoji korisnički račun

2. Sustav odbacuje pokušaj stvaranja novog računa i javlja

grešku administratoru

**UC3** Prijava

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator, administrator

**Rezultat** Prijava u sustav čime se omogućuje korisniku rad u aplikaciji

**Ostali sudionici** Baza podataka

**Željeni scenarij** 1. Korisnik upisuje svoje korisničko ime i lozinku

2. Sustav prepoznaje važeću prijavu i daje korisniku

odgovarajuće ovlasti

**Mogući scenarij**1. Korisnik je upisao nepostojeće korisničko ime ili neispravnu

lozinku

                    2. Sustav odbacuje pokušaj prijave

3. Sustav o tome obavještava korisnika

**UC4** Odjava

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator, administrator

**Rezultat** Korisnik se odjavljuje iz sustava i vraća na početnu stranicu

foruma

**Preduvjeti** 1. Korisnik je prijavljen u sustav

**Željeni scenarij** 1.Korisnik pritišće gumb „Odjavi se”

                    2. Sustav završava korisničku sjednicu

**UC5**                        Objavljivanje na forumu

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator

**Rezultat** Stvaranje nove objave

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Korisnik je prijavljen u sustav

**Željeni scenarij** 1. Korisnik upisuje tekst objave i potvrđuje unos

2. Sustav prihvaća objavu i prikazuje ju u odabranoj temi na

forumu

**UC6**                     Brisanje vlastite objave s foruma

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator

**Rezultat** Objava je obrisana iz baze podataka

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Korisnik je prijavljen u sustav

**Željeni scenarij** 1. Korisnik potvrđuje svoj odabir

                    2. Sustav briše objavu iz baze podataka

**UC7** Uređivanje vlastitih objava s foruma

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator

**Rezultat** Promjena sadržaja objave i pohranjivanje novog sadržaja u bazu podataka

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Korisnik je prijavljen u sustav

**Željeni scenarij** 1. Korisnik upisuje novi sadržaj i potvrđuje ga

2. U bazi podataka se izmjenjuje sadržaj originalne objave

3. Sustav prikazuje novu verziju ostalim korisnicima

**UC8** Pregled objava na forumu

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator

**Rezultat** Objave se prikazuju korisniku

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Korisnik je prijavljen u sustav

**Željeni scenarij** 1. Korisnik pritišće gumb „Forum“

2. Korisnik bira temu koju želi pregledati

3. Objave u određenoj temi se prikazuju korisniku klikom na ime

teme

**UC9** Stvaranje teme na forumu

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator

**Rezultat** Tema je otvorena na forumu i moguće je pisati objave u njoj

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Korisnik je prijavljen

**Željeni scenarij**   1. Korisnik pritišće gumb „Forum”

2. Korisnik stvara novu temu pritiskom na gumb „Stvori temu“ i

unosi ime nove teme

**UC10** Brisanje teme s foruma

**Glavni sudionik** Moderator

**Rezultat** Tema je obrisana iz baze podataka zajedno sa svim objavama

koje su bile u njoj

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Moderator je prijavljen

**Željeni scenarij**   1. Moderator pritišće gumb „Forum”

2. Moderator odabire temu i pritišće gumb „Obriši temu”

3. Tema se uklanja iz baze podataka i iz cjeline „Forum”

**UC11** Uklanjanje objava s foruma

**Glavni sudionik** Moderator

**Rezultat** Objava je obrisana iz baze podataka

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Moderator je prijavljen

**Željeni scenarij**   1. Moderator pritišće gumb „Obriši”

2. Objava se uklanja iz baze podataka

**UC12**                      Prijedlog najave događaja

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator

**Rezultat** Najava događaja poslana je na razmatranje moderatoru koji je

zadužen za dotičnu četvrt

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Korisnik je prijavljen u sustav

**Željeni scenarij** 1.Korisnik potvrđuje unos

                    2. Prijedlog se pohranjuje u bazu podataka

                    3. Uneseni prijedlog se prikazuje moderatoru

**UC13** Objavljivanje u cjelini „Događanja“

**Glavni sudionik** Moderator

**Rezultat** Moderator dobiva prijedlog objave od stanovnika četvrti i odlučuje hoće li ga odobriti ili odbaciti

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Moderator je prijavljen

**Mogući scenarij** a)

1. Moderator objavljuje najavu pritiskom na gumb „Objavi”

2. Odobrena najava se pohranjuje u bazu podataka

3. Odobrena najava postaje javna i vidljiva svim korisnicima

aplikacije

b)

1. Moderator pregledava prijedlog pritiskom na gumb „Uredi”

2. Moderator unosi potrebne izmjene

3. Moderator pohranjuje prijedlog u bazu podataka pritiskom na

gumb „Pohrani prijedlog” ili ga objavljuje odmah nakon uređivanja pritiskom na gumb „Objavi”

c)

1. Moderator pritišće gumb „Obriši”

2. Prijedlog se uklanja iz baze podataka

**UC14** Pregled odobrenih objava iz cjeline „Događanja“

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator

**Rezultat** Odobrene objave se prikazuju korisniku

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Korisnik je prijavljen u sustav

**Željeni scenarij** 1. Korisnik pritišće gumb „Događanja“

2. Objave se dohvaćaju iz baze podataka i prikazuju korisniku

**UC15** Objavljivanje u cjelini „Vijeće četvrti“

**Glavni sudionik** Vijećnik

**Rezultat** Objave (kratka izvješća i najave budućih tema) koje pišu vijećnici se stvaraju i moguće ih je pregledavati

**Ostali sudionici**Baza podataka

**Preduvjeti** 1. Vijećnik je prijavljen u sustav

**Željeni scenarij** 1. Vijećnik upisuje tekst objave i potvrđuje unos

2. Sustav prihvaća objavu i prikazuje ju u cjelini „Vijeće četvrti“

**UC16**                     Pregled objava vijećnika iz cjeline „Vijeće četvrti“

**Glavni sudionik** Stanovnik četvrti, vijećnik, moderator

**Rezultat** Kratka izvješća koja pišu vijećnici se prikazuju korisniku

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Korisnik je prijavljen u sustav

**Željeni scenarij** 1. Korisnik pritišće gumb „Vijeće četvrti“

2. Objave iz cjeline „Vijeće četvrti“ se dohvaćaju iz baze podataka i prikazuju korisniku

**UC17** Dodavanje četvrti

**Glavni sudionik** Administrator

**Rezultat** Nova četvrt je definirana ovom akcijom

**Ostali sudionici** Moderator, baza podataka

**Preduvjeti** 1. Administrator je prijavljen

**Željeni scenarij** 1. Administrator definira novu četvrt i unosi ulice koje se nalaze

u novodefiniranoj četvrti

2. Administrator odabire moderatora za četvrt

3. Podaci se pohranjuju u bazu podataka

**Mogući scenarij** 1. Četvrt koja se dodaje postoji u bazi podataka

2. Sustav odbacuje zahtjev za stvaranje nove četvrti

**UC18** Uređivanje četvrti

**Glavni sudionik** Administrator

**Rezultat** Administrator dodaje neku novu ulicu u sastavni dio određene četvrti

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Administrator je prijavljen

**Željeni scenarij** 1. Administrator nalazi četvrt koju treba urediti preko tražilice

koristeći ime četvrti

2. Administrator pritiskom na gumb „Dodaj ulicu” dobiva pristup

tekstualnom polju za upis ulice

3. Pri unosu imena ulice i pritiskom na gumb „Ok” ulica se dodaje

u sastav četvrti, a time i u bazu podataka

**UC19** Brisanje četvrti

**Glavni sudionik** Administrator

**Rezultat** Četvrt je obrisana iz baze podataka

**Ostali sudionici** Baza podataka

**Preduvjeti** 1. Administrator je prijavljen

2. Četvrt postoji u bazi podataka

**Željeni scenarij** 1. Administrator nalazi četvrt koju treba obrisati preko tražilice

koristeći ime četvrti

2. Administrator pritiskom na gumb „Obriši četvrt” nepovratno

briše četvrt iz baze podataka

# Dijagrami obrazaca uporabe



Slika 1: Vršni dijagram obrazaca uporabe



Slika 2: Dijagram obrazaca uporabe - Rad na forumu



Slika 3: Dijagram obrazaca uporabe - cjelina "Događanja"



Slika 4: Dijagram obrazaca uporabe - cjelina "Vijeće četvrti"



Slika 5: Dijagram obrazaca uporabe - Rad sa četvrtima

# Sekvencijski dijagrami



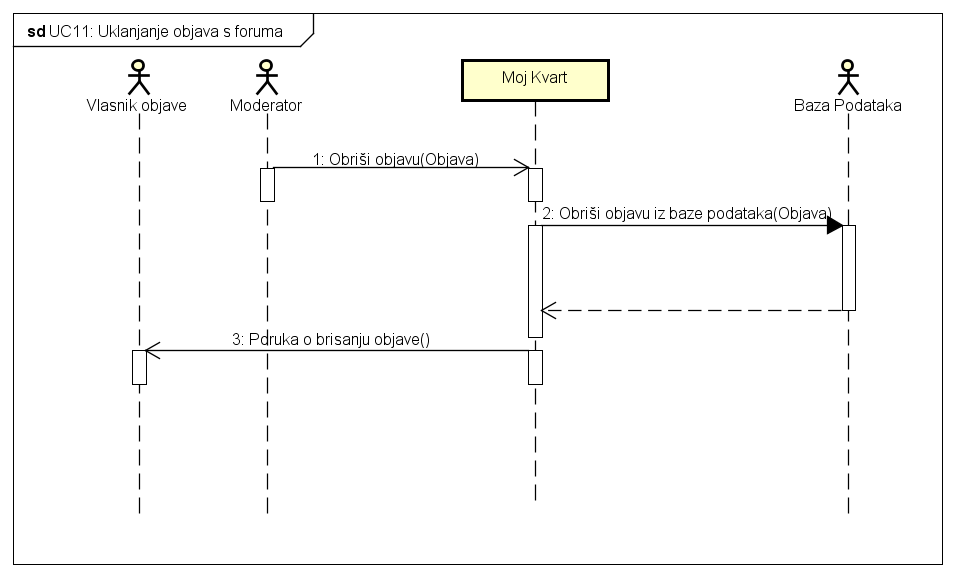
Slika 6: UC2: Otvaranje povlaštenih korisničkih računa



Slika 7: UC5 i UC8: Objavljivanje na forumu i Pregled objava na forumu



Slika 8: UC9: Stvaranje teme na forumu



Slika 9: UC11: Uklanjanje objava s foruma



Slika 10: UC12 i UC13: Prijedlog najave događaja i Objavljivanje u cjelini "Događanja"



Slika 11: UC18 i UC19: Uređivanje četvrti i Brisanje četvrti

# Ostali zahtjevi

* Korisničko sučelje mora biti oblikovano u skladu sa standardnim hrvatskim jezikom
* Sustav mora podržavati hrvatske dijakritičke znakove
* Sustav mora omogućiti paralelni rad više korisnika
* Vrijeme odziva u normalnim uvjetima ne smije biti veće od 10 sekundi
* Korisniku je ograničen broj znakova koje može unijeti u interakciji s pojedinim segmentima aplikacije
* Eventualne korisničke greške ne smiju utjecati na rad sustava i na stanje baze podataka
* Korisnik mora biti obaviješten o eventualnom neispravnom korištenju sustava
* Podaci se od poslužitelja do klijenta moraju prenositi protokolom HTTPS
* Korisničke lozinke ne smiju biti spremljene u bazu podataka kao jednostavan tekst, već se za njihovo šifriranje mora koristiti BCrypt funkcija za računanje sažetka
* Korisnik/gost ne smije imati uvid u dijelove sustava za koje nema ovlasti
* Web-aplikacija mora biti napisana koristeći se objektno orijentiranim jezikom
* Web-aplikacija i pripadna dokumentacija moraju biti u cijelosti dovršene do siječnja 2019. godine

# Arhitektura i dizajn sustava

*Ovo je je jedno od bitnijih poglavlja jer se opisuju temelji implementacije sustava. Moguće je razlikovati više aspekata arhitekture i oblikovanja sustava od kojih je većinu potrebno dokumentirati već prema zahtjevima projektnog zadatka. Neki od tih su: stil arhitekture i identifikacija podsustava, preslikavanje na radnu platformu, spremišta podataka, mrežni protokoli, globalni upravljački tok, sklopovsko-programski zahtjevi.*

*U svakoj stavci navedenoj u nastavku potrebno je dati tekstovni opis. te ilustrirati odgovarajućim UML dijagramima. Projektna grupa odlučuje gdje i koje sve dijagrame treba prikazati u pojedinim stavkama, ali cijela točka 6. Arhitektura i dizajn sustava, mora sadržavati barem po jedan:*

*Dijagram razreda*

*Dijagram objekata*

*Dijagram stanja*

*Dijagram aktivnosti*

*Komunikacijski dijagram*

*Dijagram komponenti*

*Ostali UML dijagrami (npr. Sekvencijski za objekte) mogu se koristiti po potrebi.*

*Za crtanje UML dijagrama preporuka je koristiti alat Astah Community.*

*Preporuča se da ovo poglavlje podijelite po sljedećim potpoglavljima:*

*6.1. Svrha, opći prioriteti i skica sustava*

*6.2. Dijagram razreda s opisom*

*6.3. Objektni dijagram.*

*6.4. Ostali UML dijagrami*

## Svrha, opći prioriteti i skica sustava

Razvoj programskog produkta se sastoji od četiri generičke aktivnosti: specifikacija, razvoj i oblikovanje, validacija te evaluacija. Fazi razvoja i oblikovanja kao prva, odnosno, početna aktivnost pripada izbor i oblikovanje arhitekture. Arhitektura sustava podrazumijeva osnovna pravila stvaranja i strukturiranja cijelog sistema te je ona, kada je jednom osmišljena i provedena, nešto što se ne može tek tako mijenjati pa je zbog toga prije same implementacije potrebno arhitekturi pridati mnogo pažnje i vremena.

Pravilan odabir arhitekture ima niz prednosti, od kojih su samo neke niža cijena razvoja i održavanja programskog produkta, ponovna uporaba, veća razumljivost te uočavanje i analiza ranih pogrešaka u oblikovanju. Iz navedenih razloga je vidljivo da je odabir arhitekture ključan za razvoj programskog produkta pa je u svrhu izrade web aplikacije čija će implementacija biti opisana u kasnijim poglavljima odabrana objektno orijentirana arhitektura.

Osnovni elementi koje svaka objektno orijentirana arhitektura uključuje su:

• **Objekt** (*engl. Object)* – osnovna jedinica, primjerak razreda. U objektno orijentiranoj paradigmi teži se uporabi članskih atributa. Isto tako metode pripadaju nekom konkretnom razredu.

• **Učahurivanje** (*engl. Encapsulation*) – ili enkapsulacija. Pošto ne postoje globalne varijable, pristup varijablama razreda je moguć jedino preko neke metode iz te klase. Time je onemogućeno nepredviđeno mijenjanja parametara.

• **Nasljeđivanje** (*engl. Inheritance*) – razredi mogu naslijediti svojstva svojih roditelja te ih nadopuniti svojstvima specifičnim samo za taj razred.

• **Višeobličje** (*engl. Polymorphism*) – mogućnost definiranje metoda istog imena, a različitih parametara.

Objektno orijentirana paradigma općenito je način modeliranja u kojem se svijet promatra kroz objekte. Takav način oblikovanja je već ustaljen u svijetu programiranja, modularan, razumljiv i kod je oblikovan u smislene cjeline koje nazivamo razredima.

Ovaj projekt će prilikom oblikovanja pratiti MVC (“Model-View-Controller”, odnosno model – pogled – nadglednik) obrazac, a sam sustav se bazira na arhitekturi klijent – poslužitelj.

Razvoj web aplikacije bit će podijeljen u tri mehanizma:

1. **Model** – objektni model podataka. Služi skladištenju struktura podataka korisničkog sučelja.
2. **View** – prikaz grafičkog korisničkog sučelja.
3. **Controller** – logika interakcije / promjene.



Slika 12: MVC obrazac

Arhitektura sustava ovog projekta sastoji se od:

* Klijentska aplikacija
* Administratorska aplikacija
* Središnji poslužitelj
* Baza podataka

**Klijentska aplikacija**

Klijentska aplikacija je namijenjena krajnjem korisniku (u vidu ovog projekta to je stanovnik četvrti). Ona nudi grafičko korisničko sučelje koje je razumljivo i intuitivno za korištenje.

**Administratorska aplikacija**

Administrator sustava upravlja računima korisnika, otvara povlaštene račune, definira četvrti i područja koja ona obuhvaćaju . Administrator je neovisan od stanovnika četvrti i ne preuzima njegove uloge te on navedenim ovlastima pristupa preko administratorske aplikacije koja je neovisna o klijentskoj aplikaciji.

**Središnji poslužitelj**

Središnji poslužitelj vrši obradu zahtjeva klijentske i administratorske aplikacije. Svaki zadatak koji uključuje slanje i primanje podataka komunicira sa središnjim poslužiteljem čiji je zadatak obrada i slanje povratnih informacija klijentu. Središnji poslužitelj pri obavljanju rada komunicira s bazom podataka.

**Baza podataka**

Baza podataka je važan dio sustava jer su u njoj pohranjeni svi podaci koje koristi web aplikacija. Korištena baza podataka je relacijskog tipa. Takav model organizira podatke u jednu ili više tablica (realcija) između kojih su definirane veze.



Slika 13: Skica baze podataka

Baza ovog sustava sadrži tablice s navedenim atributima:

* **Korisnik** (*user*)
* šifra korisnika
* korisničko ime
* sažetak lozinke
* datum kreiranja računa
* **Administrator** (*administrator*)
* **Stanovnik** (*resident*)
* ime stanovnika
* prezime stanovnika
* email adresa stanovnika
* šifra ulice u kojoj stanovnik živi
* **Moderator** (*moderator*)
* **Vijećnik** (*councilor*)
* **Objava** (*forum\_post*)
* šifra objave
* sadržaj objave
* vrijeme objave
* šifra autora objave
* šifra teme kojoj objava pripada
* **Tema** (*forum\_thread*)
* šifra teme
* naslov teme
* vrijeme otvaranja teme
* autor teme
* šifra četvrti za koju je vezana tema
* **Četvrt** (*neighborhood*)
* šifra četvrti
* ime četvrti
* **Događaj** (*event*)
* šifra događaja
* ime događaja
* lokacija događaja
* vrijeme početka događaja
* vrijeme kraja događaja
* **Predložen događaj** (*suggested\_event*)
* šifra autora prijedloga događaja
* vrijeme predlaganja događaja
* **Prihvaćen događaj** (*accepted\_event*)
* vrijeme prihvaćanja događaja
* **Ulica** (*street*)
* šifra ulice
* ime ulice
* šifra četvrti u kojoj je ulica

## Dijagram razreda s opisom



Slika 14: Dijagram razreda

Razred **User** predstavlja registriranog korisnika. Sadrži korisničko ime, sažetak lozinke i vrijeme otvaranja računa.

**Administrator** je registrirani korisnik s posebnim ovlastima.

**Resident** je uobičajeni registrirani korisnik koji stanuje u nekoj četvrti. Sadrži puno ime, e-mail adresu i ulicu u kojoj stanuje.

**Resident** može dodatno biti **Councilor** (vijećnik četvrti) ili **Moderator**.

Razred **Street** predstavlja imenovanu ulicu, a **Neighborhood** skup ulica, odnosno imenovanu četvrt.

**Neighborhood** sadrži kolekcije instanci razreda **ForumThread** (tema na forumu), **AcceptedEvent** (prihvaćeni događaj), **SuggestedEvent** (predloženi događaj) i **CouncilReport** (izvješće vijeća četvrti).

**ForumThread** sadrži naslov, trenutak otvaranja i stanovnika (**Resident**) koji ju je otvorio te se sastoji od više postova na forumu (**ForumPost**).

**ForumPost** sadrži tekst objave i vrijeme objave.

Razred **Event** predstavlja općenite informacije o nekom događaju, a sadrži naslov, lokaciju, trenutak početka i trenutak završetka (koji može biti i nepoznat).

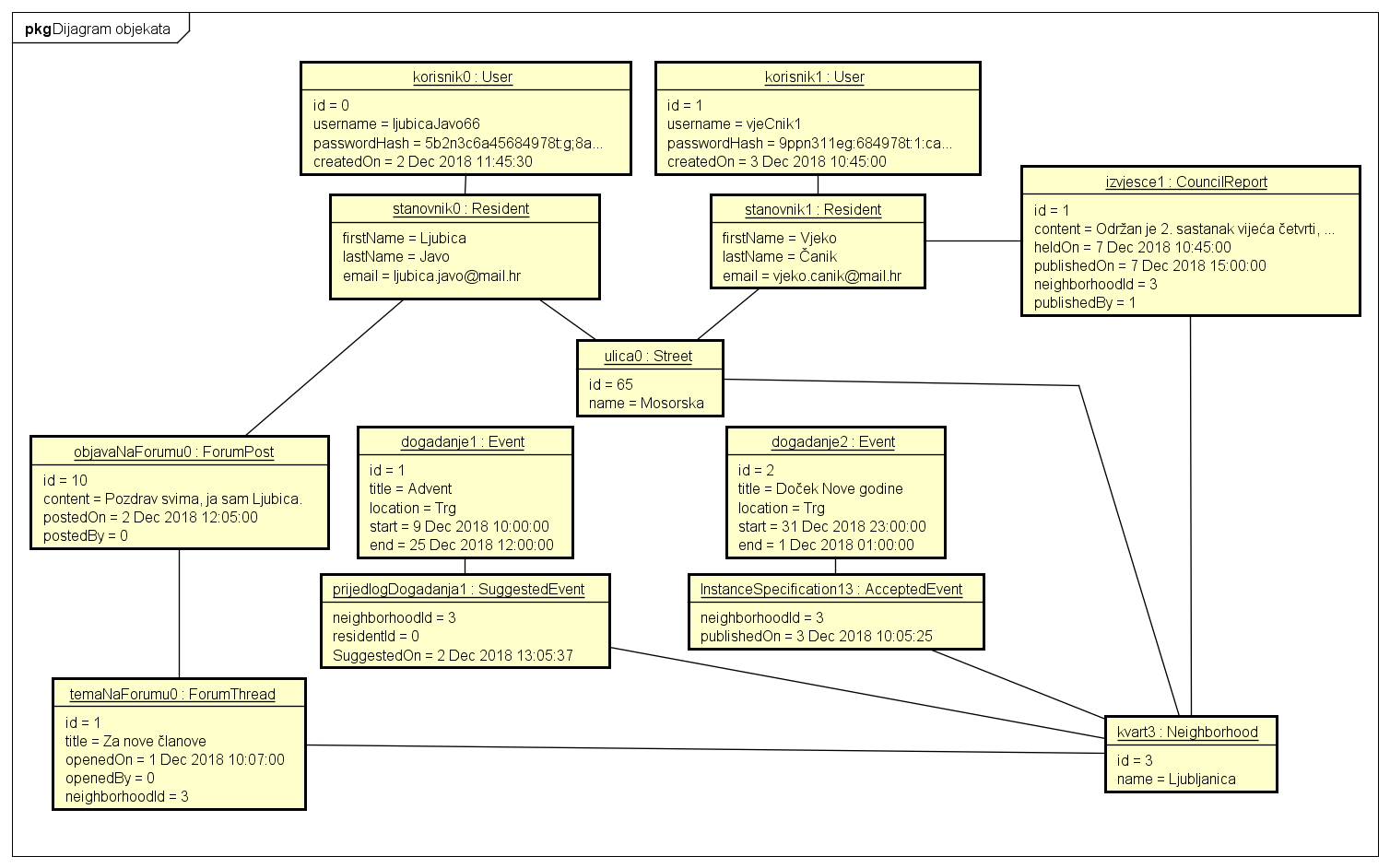
**SuggestedEvent** je **Event** kojeg je neki stanovnik predložio. Osim reference na tog stanovnika (**Resident**), sadrži i trenutak kad ga je predložio.

**AcceptedEvent** je **Event** koji je prihvaćen i objavljen. Sadrži trenutak objavljivanja.

**CouncilReport** predstavlja izvješće s vijeća četvrti. Sadrži tekst izvješća, vrijeme održavanja, vrijeme objavljivanja i referencu na vijećnika (**Councilor**) koji ga je objavio.

## Dijagram objekata

Sljedeći dijagram objekata prikazuje djelomično stanje sustava nastalo registracijom i radnjama dvaju korisnika - redovnog stanovnika koji je predložio dva događanja i ostavio objavu na forumu te vijećnika koji je objavio izvješće vijeća četvrti.



Slika 15: Dijagram objekata

## Ostali UML dijagrami

*Ovdje počinju sadržaji Revizije 2.*

*U ovom potpoglavlju potrebno je uvrstiti dodatna četiri dijagrama koji prikazuju arhitekturu sustava i to: komunikacijski dijagram, dijagram stanja, dijagam aktivnosti i dijagram komponenti. Potrebno je dati barem po jedan primjerak svakog od dijagrama. Potrebno je da svaki od dijagrama prikazuje (jedan) bitan dio funkcionalnosti sustava. Dijagram komponenti treba prikazivati sve komponente sustava. Prema potrebi, ovdje se mogu dodati i neki dodatni sekvencijski dijagrami za objekte.*

# Implementacija i korisničko sučelje

*U ovom poglavlju potrebno je:*

*dati dijagram razmještaja (deployment dijagram)*

*navesti koje su tehnologije i alati korišteni u razvoju sustava*

*dati isječak programskog koda koji implementira neku od temeljnih funkcionalnosti u sustavu*

*objasniti kako je ispitano implementirano rješenje i pokazati bar 4 ispitna scenarija*

*dati upute za instalaciju*

*dati upute za korištenje*

## Dijagram razmještaja

*Potrebno je umetnuti dijagram razmještaja i po potrebi ga opisati.*

## Korištene tehnologije i alati

*Navesti sve tehnologije i alate koji su primijenjeni pri izradi projekta te ih ukratko opisati; njihovo značenje i mjesto i način primjene.*

## 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).*

## Ispitivanje programskog rješenja

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

*Prilikom prezentacije svojih Ispitnih scenarija (minimalno četiri) 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.*

## Upute za instalaciju

*U ovom poglavlju potrebno je dati upute za instalaciju ostvarenog prototipa.*

## Korisničke upute

*Korisničke upute ovisit će o količini implementirane funkcionalnosti. Očekuje se da upute budu na oko pet A4 stranica koje će dati potpuni opis funkcionalnosti sustava sa stajališta krajnjeg korisnika.*

# Zaključak i budući rad

*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.*

# Popis literature

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

1. Oblikovanje programske potpore, FER ZEMRIS, <http://www.fer.hr/predmet/opp>
2. Oblikovanje programske potpore, FER ZEMRIS, <http://www.zemris.fer.hr/predmeti/opp>
3. I. Sommerville, „Software engineering“, 8th ed, Addison Wesley, 2007.
4. T.C.Lethbridge, R.Langaniere, „Object-Oriented Software Engineering“, 2nd ed. McGraw-Hill, 2005.
5. Software engineering ,Rutgers University, <http://www.ece.rutgers.edu/~marsic/Teaching/SE>
6. I. Marsic, „Software engineering book“, Department of Electrical and Computer Engineering, Rutgers University, <http://www.ece.rutgers.edu/~marsic/books/SE>
7. Concepts: Requirements, <http://www.upedu.org/upedu/process/gcncpt/co_req.htm>
8. UML 2 Class Diagram Guidelines, <http://www.agilemodeling.com/style/classDiagram.htm>
9. Domain Class Diagram Modeling Standards and Guidelines, <http://www.bced.gov.bc.ca/imb/downloads/classdiagramstandards.pdf>
10. Astah Community, <http://astah.net/editions/community/>

# Dodatak A: Indeks (slika, dijagrama, tablica, ispisa kôda)

*U ovom dijelu dodatka potrebno je ispisati (numerirane) liste slika, dijagrama, tablica, ispisa kôda i ostalih pomoćnih sadržaja iz teksta dokumentacije.*

# Dodatak B: Dnevnik sastajanja

*U ovom dijelu dodatka potrebno je prenijeti sadržaj dnevnika sastajanja, tj. datoteke 'dnevnik.txt' iz repozitorija grupe. Datoteku 'dnevnik.txt' potrebno je voditi kroz repozitorij, a ovdje će to biti preneseno kao dio konačne dokumentacije.*

# Dodatak C: Prikaz aktivnosti grupe

*U ovom dijelu dodatka potrebno je navesti*

* *popis članova grupe i njihovih zaduženja, tj. odrađenih poslova (u postocima ukupno odrađenog posla)*
* *prenijeti dijagram pregleda promjena nad datotekama projekta. Potrebno je u* BitBucket *sučelju instalirati dodatak* AwesomeGraph *koji daje prikaz grafički prikaz aktivnosti grupe, tj. '*commit' *akcija tijekom trajanja projekta, te na kraju projekta generirane grafove prenijeti u ovo poglavlje dokumentacije.*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Popis aktivnosti** | **Članovi grupe** (abecednim redom) | | | | | | |
| Ime Prezime voditelja | Ime Prezime 2 | Ime Prezime 3 | Ime Prezime 4 | Ime Prezime 5 | Ime Prezime 6 | Ime Prezime 7 |
| **Upravljanje projektom** |  |  |  |  |  |  |  |
| **Opis projektnog zadatka** |  |  |  |  |  |  |  |
| **Rječnik pojmova** |  |  |  |  |  |  |  |
| **Opis funkcionalnih zahtjeva** |  |  |  |  |  |  |  |
| **Opis ostalih zahtjeva** |  |  |  |  |  |  |  |
| **Arhitektura i dizajn sustava** |  | | | | | | |
| Svrha, opći prioriteti i skica sustava |  |  |  |  |  |  |  |
| Dijagram razreda s opisom |  | | | | | | |
| Dijagram objekata |  |  |  |  |  |  |  |
| Ostali UML dijagrami |  |  |  |  |  |  |  |
| **Implementacija i korisničko sučelje** |  | | | | | | |
| Dijagram razmještaja |  |  |  |  |  |  |  |
| Korištene tehnologije i alati |  |  |  |  |  |  |  |
| Isječak programskog kôda |  |  |  |  |  |  |  |
| Ispitivanje programskog rješenja |  |  |  |  |  |  |  |
| Upute za instalaciju |  |  |  |  |  |  |  |
| Korisničke upute |  |  |  |  |  |  |  |
| **Plan rada** |  |  |  |  |  |  |  |
| **Pregled rada i stanje ostvarenja** |  |  |  |  |  |  |  |
| **Zaključak i budući rad** |  |  |  |  |  |  |  |
| **Popis literature** |  |  |  |  |  |  |  |
| **Dodaci** |  | | | | | | |
| Indeks |  |  |  |  |  |  |  |
| Dnevnik sastajanja |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

*Napomena: Doprinose u aktivnostima treba navesti u postocima po članovima grupe. Zbroj postotaka u svakom retku treba biti 100%.*

Pregled pohrana kroz vrijeme trajanja projekta (primjer):



# Dodatak D: Plan rada / Pregled rada i stanje ostvarenja

*U ovom poglavlju potrebno je navesti:*

* *(u rev. 1) koji je plan rada za rev. 2,*
* *(u rev. 2) koji je status implementacije u odnosu na postavljene ciljeve, procjenu vremena dovršetka projekta (ako zadatak nije u potpunosti ispunjen), koje bi bile smjernice za daljnji rad kad bi se nastavilo s projektom te u čemu bi se sastojale buduće nadogradnje.*