Programsko inženjerstvo

Ak. god. 2023./2024.

SpotPicker

Dokumentacija, Rev. *1*

Grupa: *Šargarepoljupci*

Voditelj: *Matko Krnić*

Datum predaje: <*dan*>. <*mjesec*>. <*godina*>.

Nastavnik: *Hrvoje Nuić*

Sadržaj

1. Dnevnik promjena dokumentacije

Kontinuirano osvježavanje

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak.	*	22.08.2013.
0.2	Dopisane upute za povijest dokumentacije. Dodane reference.	*	24.08.2013.
0.5	Dodan *Use Case* dijagram i jedan sekvencijski dijagram, funkcionalni i nefunkcionalni zahtjevi i dodatak A	*	25.08.2013.
0.6	Arhitektura i dizajn sustava, algoritmi i strukture podataka	*	26.08.2013.
0.8	Povijest rada i trenutni status implementacije, Zaključci i plan daljnjeg rada	*	28.08.2013.
0.9	Opisi obrazaca uporabe	*	07.09.2013.
0.10	Preveden uvod	*	08.09.2013.
0.11	Sekvencijski dijagrami	*	09.09.2013.
0.12.1	Započeo dijagrame razreda	*	10.09.2013.
0.12.2	Nastavak dijagrama razreda	*	11.09.2013.
1.0	Verzija samo s bitnim dijelovima za 1. ciklus	*	11.09.2013.
1.1	Uređivanje teksta – funkcionalni i nefunkcionalni zahtjevi	* *	14.09.2013.
1.2	Manje izmjene:Timer - Brojilo vremena	*	15.09.2013.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	**Opis promjene/dodatka**	**Autori**	**Datum**
1.3	Popravljeni dijagrami obrazaca uporabe	*	15.09.2013.
1.5	Generalna revizija strukture dokumenta	*	19.09.2013.
1.5.1	Manja revizija (dijagram razmještaja)	*	20.09.2013.
2.0	Konačni tekst predloška dokumentacije	*	28.09.2013.

Moraju postojati glavne revizije 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, 0.10, 0.11.. sve do konačne revizije prvog ciklusa 1.0. U drugom ciklusu se nastavlja s revizijama 1.1, 1.2, itd.

2. Opis projektnog zadatka

Ideja našeg projekta je ostvariti aplikaciju koja će omogućiti korisniku da pregledava, rezervira i naplaćuje parkirna mjesta za automobile i bicikle. Aplikacija to ostvaruje na način da prilikom njezinog otvaranja prikazuje korisniku kartu nad kojom korisnik može birati željeno odredište, tip vozila koje koristi te trajanje parkinga. Nakon svih odabira, na karti mu se iscrtava najbrža ruta do slobodnog parkirališnog mjesta te ako je mjesto slobodno, ono se rezervira. Parkirališta za bicikle nemaju ucrtana pojedinačna mjesta te se ne rezerviraju ni naplaćuju, već aplikacija jedino prati ukupan broj slobodnih mjesta. Za računanje najbližeg dostupnog parkirališnog mjesta aplikacija koristi OSRM, što je kratica za Open Source Routing Machine. Riječ je o modernom C++ algoritmu za usmjeravanje koji se koristi za izračunavanje najkraćeg puta u cestovnom prometu.
Aplikacija koristi API za više funkcija:

- implementacija novčanika u koji korisnik uplaćuje sredstva
- osvježavanje parkirališnog mjesta radi provjere zauzetosti
- dohvaćanje početnih informacija o stanju parkirališnih mjesta pri čemu se koristi overpass API

Aplikacija se može pokrenuti bez registracije, no za pristup svim alatima aplikacije, potrebna je registracija. Korisnici koji nisu registrirani mogu na karti vidjeti sva dostupna parkirališta i parkirališna mjesta, ali nemaju informacije u njihovom stanju u stvarnom vremenu. Neregistrirani korisnik koji se želi registrirati šalje zahtjev za registraciju pri čemu ima ima opciju birati između dvije uloge (voditelj parkinga i klijent). Za registraciju su potrebni:

- korisničko ime
- ime
- prezime
- slika osobne
- IBAN račun
- email adresa

Za registraciju kao klijent dovoljna je samo potvrda preko email adrese. Ako je riječ o voditelju parkinga tada je potrebna i potvrda administratora. Registrirani korisnici također imaju mogućnost pregledavanja te izmjene vlastitih podataka, uključujući i opciju da obrišu svoj račun.

Klijent otvaranjem aplikacije dobiva pregled nad kartom te odabire lokaciju odredišta, tip vlastitog vozila, i trajanje parkinga. Klijent može rezervirati parkirališno mjesto na dva načina. Može prvo označiti parkirališna mjesta koja mu odgovaraju te mu tada aplikacija nudi moguće termine u kojima će ta mjesta biti slobodna. Druga opcija je da prvo definira odgovarajući termin nakon čega će mu aplikacija prikazati sva parkirališna mjesta koja su slobodna u tom terminu. Kada je klijent zadovoljan s parkirališnim mjestom, rezervira mjesto na proizvoljno vrijeme ako je slobodno te pri tome ima opciju da napravi rezervaciju ponavljajućom. Klijent jedino može rezervirati mjesto u budućnosti, odnosno, mjesto ne može biti rezervirano u istom datumu u kojem je korisnik otvorio aplikaciju. Plaćanje parkinga korisnik obavlja tijekom rezervacije ili po dolasku na parkirališno mjesto. Plaćanje se obavlja preko novčanika koji je implementiran unutar aplikacije u koji korisnik može proizvoljno ubaciti novac.

Voditelj parkinga ima veće ovlasti od klijenta. On ima dozvolu unijeti podatke o vlastitom parkiralištu koje uključuju:

- naziv
- opis
- fotografija
- cjenik i sl.

Voditelj nad svojim parkiralištem može individualno ucrtati svako parkirališno mjesto prema vlastitim željama. Nakon što ucrta mjesta, on odlučuje je li mjesto dostupno za rezervaciju. Dodatno, voditelj postavlja i senzor koji služi za dohvaćanje informacije o zauzetosti mjesta. Za svoje parkiralište voditelj određuje cijenu rezervacije ovisnu o vremenu. Voditelj također ima i pristup statističkim podacima o svom parkiralištu. Moguće je promatrati zauzetost cijelog parkirališta te pojedinačnih parkirališnih mjesta kroz vrijeme pomoću grafa.

Administrator aplikacije ima najveće ovlasti i sposoban je vidjeti popis svih registriranih korisnika i njihovih osobnih podataka te im po potrebi može mijenjati osobne podatke.

Motivacija za razvoj ovog projekta dolazi od mnogih prednosti koje aplikacija za parkiranje može donijeti u odnosu na klasičan način rezerviranja i plaćanja

parkinga. Za početak, korištenjem ove aplikacije korisniku se značajno smanjuje vrijeme potrebno za pronalaženje parkirališnog mjesta. Dodatno, plaćanje parkirališnog mjesta može biti izvor frustracije zbog korištenja kovanica i gomilanja sitniša ili zbog potrebe za vađenjem kartice. Aplikacija za taj problem nudi praktično rješenje implementacijom novčanika što omogućuje korisniku da sva svoja plaćanja obavi unaprijed preko mobilnog telefona. Jednostavnost i elegantnost ove aplikacije pomaže i pri smanjenju stresa kod korisnika. Pitanja poput: "Gdje ćemo parkirati?", "Smijem li ja parkirati na ovom mjestu?" ili "Ima li uopće slobodnih mjesta?" više neće biti razlog za brigu jer aplikacija obavlja svo razmišljanje za vas. Problem traženja parkirališnog mjesta u nepoznatom gradu će se svesti na samo par klikova na aplikaciji te će uštedjeti korisnicima i novac budući da neće gubiti vrijeme vozajući se po gradu, tražeći slobodno mjesto. Konačno, aplikacija pridonosi i okolišu jer manje vremena potrošeno na potražnju mjesta za parkiranje znači manje vremena koje trošimo na sagorijevanje goriva.

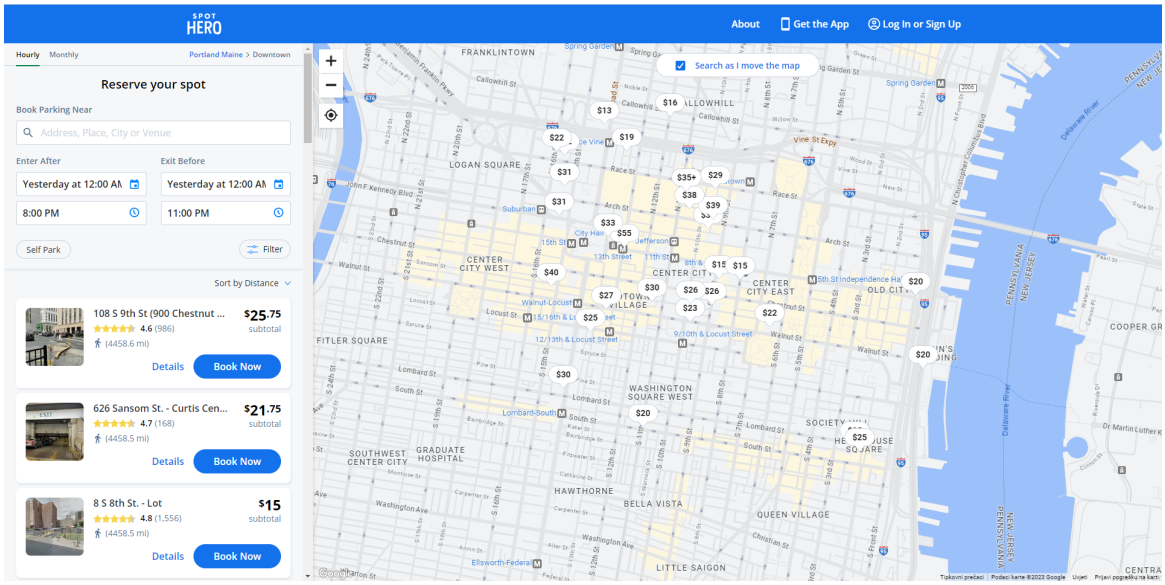
Neke od aplikacija koje pružaju slična rješenja su:

- SpotHero
- Best in Parking
- ParkWhiz
- Parkopedia
- Way

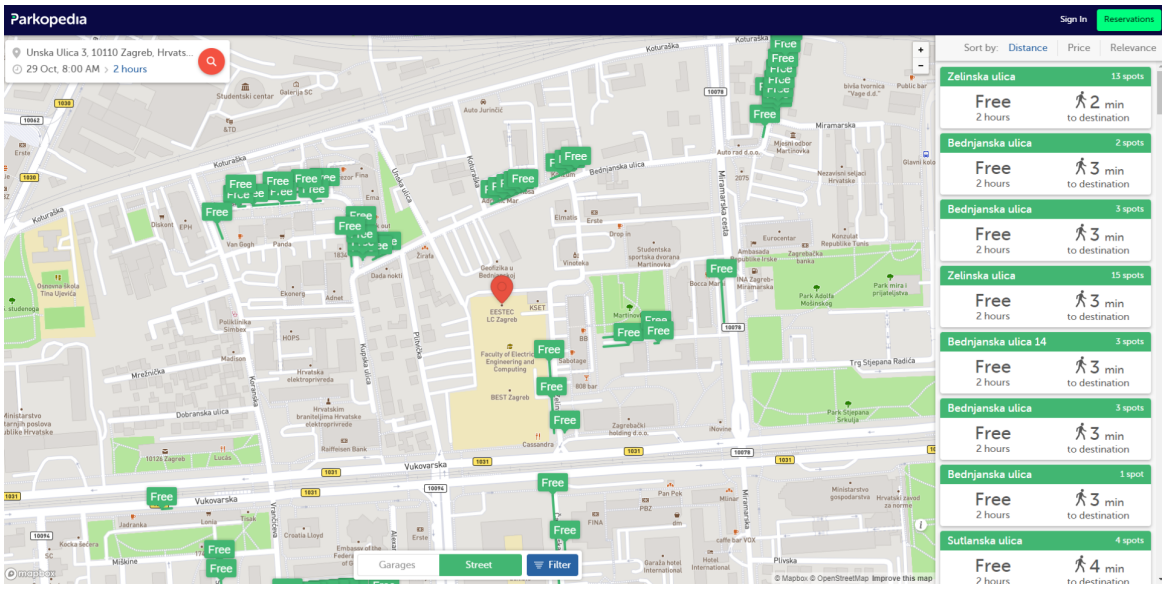
Kako bi bolje ilustrirali prednosti koje naša aplikacija ima u odnosu na neke od postojećih rješenja, uzet ćemo kao primjer *SpotHero*. *SpotHero* korisniku prikazuje dostupna parkirališna mjesta u blizini nakon što korisnik upiše adresu na kojoj se nalazi. Korisnik tada može kartu rezervirati i tada dobiva propusnicu koju skenira na parkirališnom mjestu. *SpotHero* omogućuje korisniku mjesečni parking te parking u zračnoj luci. Bitna razlika između naše aplikacije i *SpotHero* je to da kod korištenja *SpotHero-a* korisnik mora odmah definirati termin i vrijeme parkiranja dok naša aplikacija dozvoljava korisniku da prvo označi parkirališna mjesta, a nakon toga mu se nude dostupni termini. Aplikacije se razlikuju i u tome što *SpotHero* ne daje opciju za bicikle, već samo za osobna vozila. Dodatno, *SpotHero* je jedino dostupan u SAD-u i Kanadi.

Parkopedia je još jedna takva slična aplikacija. *Parkopedia* koristi Google Maps kako bi iscrtala najbližu rutu korisniku. Razlika između našeg projekta i ove aplikacije je što *Parkopedia* zahtijeva da se prvo unese adresa odredišta nakon čega nudi popis dostupnih parkirališta. *Parkopedia* na prikazu karte daje opciju između pri-

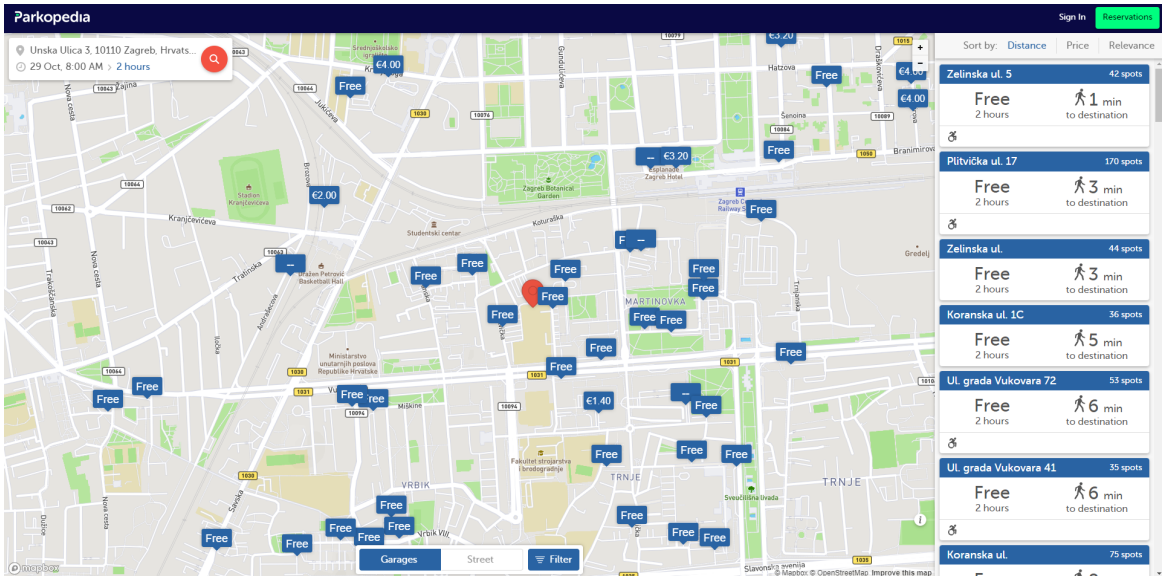
kazivanja parkirališnih mjesta na ulici i u garaži. Kao i kod *SpotHero-a*, *Parkopedia* ne nudi opciju za bicikle te pretraga prvo po terminu parkiranja nije moguća.



Slika 2.1: Prikaz aplikacije SpotHero



Slika 2.2: Prikaz aplikacije Parkopedia s označenim parkiralištima na ulici



Slika 2.3: Prikaz aplikacije Parkopedia s označenim parkirališima na garaži

Vjerujemo da našom aplikacijom možemo zainteresirati ljude koji su već upoznati s parking aplikacijama, ali smatraju da postoje područja u kojima trenutačno najpopularnije aplikacije nisu dovoljno prilagodljive korisničkim željama. Konačan plan i opseg ovog projekta je učiniti ovu aplikaciju dostupnu cijelom svijetu. Ovisno o potrebi i potražnji korisnika, aplikaciju je moguće u budućnosti i nadograditi funkcionalnostima koje druge aplikacije imaju poput mjesečnog rezerviranja parkinga, razlikovanja parkirališnih mjesta po razini i slično.

2.1 Primjeri u LaTeXu

Ovo potpoglavlje izbrisati.

U nastavku se nalaze različiti primjeri kako koristiti osnovne funkcionalnosti LaTeXa koje su potrebne za izradu dokumentacije. Za dodatnu pomoć obratiti se asistentu na projektu ili potražiti upute na sljedećim web sjedištima:

- Upute za izradu diplomskog rada u LaTeXu - `https://www.fer.unizg.hr/_download/repository/LaTeX-upute.pdf`
- LaTeX projekt - `https://www.latex-project.org/help/`
- StackExchange za Tex - `https://tex.stackexchange.com/`

podcrtani tekst, **podebljani tekst**, *nagnuti tekst*

primjer primjer primjer primjer primjer primjer

- primjer
- primjer
- primjer
  1. primjer
     1.a primjer
       b primjer
  2. primjer

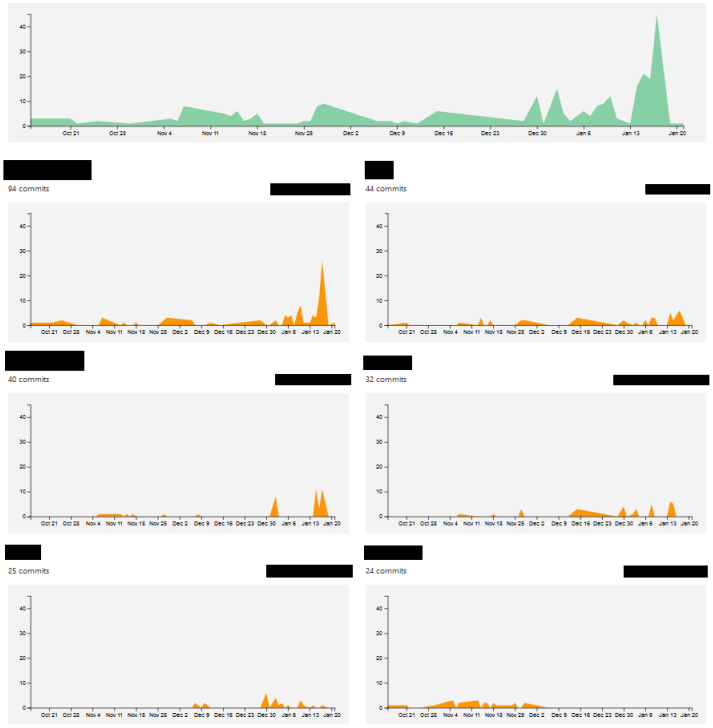
primjer url-a: `https://www.fer.unizg.hr/predmet/proinz/projekt`

posebni znakovi: # $ % & { } _ | < > ^ ~ \

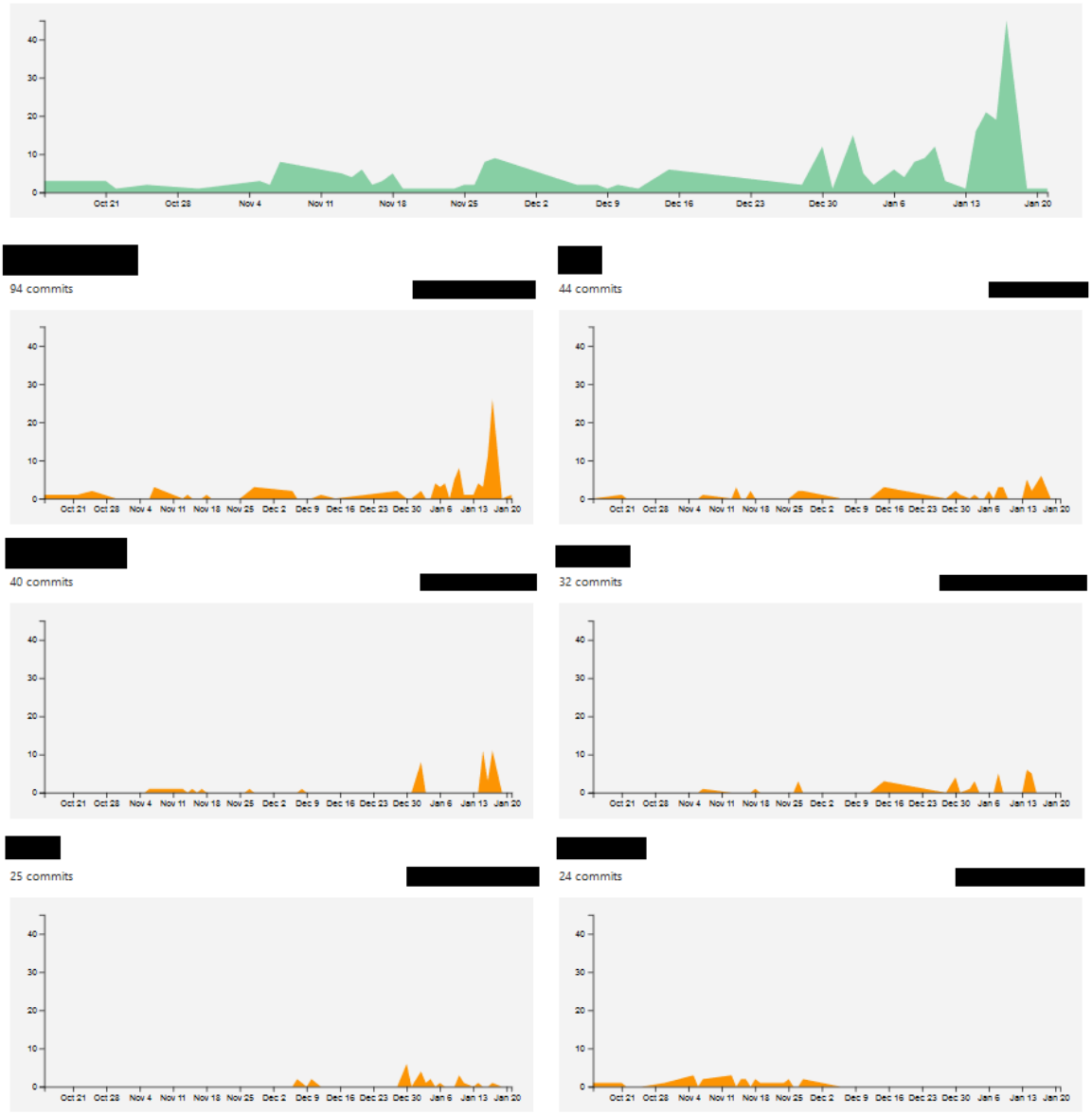
naslov unutar tablice		
IDKorisnik	INT	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
korisnickoIme	VARCHAR	
email	VARCHAR	
ime	VARCHAR	
primjer	VARCHAR	

Tablica 2.1: Naslov s referencom izvan tablice

IDKorisnik	INT	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
korisnickoIme	VARCHAR	
email	VARCHAR	
ime	VARCHAR	
primjer	VARCHAR	



Slika 2.4: Primjer slike s potpisom



Slika 2.5: Primjer slike s potpisom 2

Referenciranje slike 2.5 u tekstu.

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

dio 1. revizije

*Navesti **dionike** koji imaju **interes u ovom sustavu** ili **su nositelji odgovornosti**. To su prije svega korisnici, ali i administratori sustava, naručitelji, razvojni tim.*

*Navesti **aktore** koji izravno **koriste** ili **komuniciraju sa sustavom**. Oni mogu imati inicijatorsku ulogu, tj. započinju određene procese u sustavu ili samo sudioničku ulogu, tj. obavljaju određeni posao. Za svakog aktora navesti funkcionalne zahtjeve koji se na njega odnose.*

Dionici:

1. Dionik 1
2. Dionik 2
3. ...

Aktori i njihovi funkcionalni zahtjevi:

1. Aktor 1 (inicijator) može:
   (a) funkcionalnost 1
   (b) funkcionalnost 2
      i. podfunkcionalnost 1
      ii. podfunkcionalnost 2
   (c) funkcionalnost 3
2. Aktor 2 (sudionik) može:
   (a) funkcionalnost 1
   (b) funkcionalnost 2

3.1.1 Obrasci uporabe

dio 1. revizije

Opis obrazaca uporabe

Funkcionalne zahtjeve razraditi u obliku obrazaca uporabe. Svaki obrazac je potrebno razraditi prema donjem predlošku. Ukoliko u nekom koraku može doći do odstupanja, potrebno je to odstupanje opisati i po mogućnosti ponuditi rješenje kojim bi se tijek obrasca vratio na osnovni tijek.

UC<broj obrasca> -<ime obrasca>

- **Glavni sudionik:** <sudionik>
- **Cilj:** <cilj>
- **Sudionici:** <sudionici>
- **Preduvjet:** <preduvjet>
- **Opis osnovnog tijeka:**
  1. <opis korak jedan>
  2. <opis korak dva>
  3. <opis korak tri>
  4. <opis korak četiri>
  5. <opis korak pet>
- **Opis mogućih odstupanja:**

  2.a <opis mogućeg scenarija odstupanja u koraku 2>
    1. <opis rješenja mogućeg scenarija korak 1>
    2. <opis rješenja mogućeg scenarija korak 2>

  2.b <opis mogućeg scenarija odstupanja u koraku 2>

  3.a <opis mogućeg scenarija odstupanja u koraku 3>

Dijagrami obrazaca uporabe

Prikazati odnos aktora i obrazaca uporabe odgovarajućim UML dijagramom. Nije nužno nacrtati sve na jednom dijagramu. Modelirati po razinama apstrakcije i skupovima srodnih funkcionalnosti.

3.1.2 Sekvencijski dijagrami

dio 1. revizije

Nacrtati sekvencijske dijagrame koji modeliraju najvažnije dijelove sustava (max. 4 dijagrama). Ukoliko postoji nedoumica oko odabira, razjasniti s asistentom. Uz svaki dijagram napisati detaljni opis dijagrama.

3.2 Ostali zahtjevi

dio 1. revizije

Nefunkcionalni zahtjevi i zahtjevi domene primjene dopunjuju funkcionalne zahtjeve. Oni opisuju ***kako se sustav treba ponašati*** *i koja* ***ograničenja*** *treba poštivati (performanse, korisničko iskustvo, pouzdanost, standardi kvalitete, sigurnost...). Primjeri takvih zahtjeva u Vašem projektu mogu biti: podržani jezici korisničkog sučelja, vrijeme odziva, najveći mogući podržani broj korisnika, podržane web/mobilne platforme, razina zaštite (protokoli komunikacije, kriptiranje...)... Svaki takav zahtjev potrebno je navesti u jednoj ili dvije rečenice.*

4. Arhitektura i dizajn sustava

dio 1. revizije

Potrebno je opisati stil arhitekture te identificirati: podsustave, preslikavanje na radnu platformu, spremišta podataka, mrežne protokole, globalni upravljački tok i sklopovsko-programske zahtjeve. Po točkama razraditi i popratiti odgovarajućim skicama:

- *izbor arhitekture temeljem principa oblikovanja pokazanih na predavanjima (objasniti zašto ste baš odabrali takvu arhitekturu)*
- *organizaciju sustava s najviše razine apstrakcije (npr. klijent-poslužitelj, baza podataka, datotečni sustav, grafičko sučelje)*
- *organizaciju aplikacije (npr. slojevi frontend i backend, MVC arhitektura)*

4.1 Baza podataka

dio 1. revizije

Potrebno je opisati koju vrstu i implementaciju baze podataka ste odabrali, glavne komponente od kojih se sastoji i slično.

4.1.1 Opis tablica

Svaku tablicu je potrebno opisati po zadanom predlošku. Lijevo se nalazi točno ime varijable u bazi podataka, u sredini se nalazi tip podataka, a desno se nalazi opis varijable. Svjetlozelenom bojom označite primarni ključ. Svjetlo plavom označite strani ključ

korisnik - ime tablice		
IDKorisnik	INT	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

korisnik - ime tablice		
korisnickoIme	VARCHAR	
email	VARCHAR	
ime	VARCHAR	
primjer	VARCHAR	

4.1.2 Dijagram baze podataka

U ovom potpoglavlju potrebno je umetnuti dijagram baze podataka. Primarni i strani ključevi moraju biti označeni, a tablice povezane. Bazu podataka je potrebno normalizirati. Podsjetite se kolegija "Baze podataka".

4.2 Dijagram razreda

Potrebno je priložiti dijagram razreda s pripadajućim opisom. Zbog preglednosti je moguće dijagram razlomiti na više njih, ali moraju biti grupirani prema sličnim razinama apstrakcije i srodnim funkcionalnostima.

dio 1. revizije

Prilikom prve predaje projekta, potrebno je priložiti potpuno razrađen dijagram razreda vezan uz ***generičku funkcionalnost*** *sustava. Ostale funkcionalnosti trebaju biti idejno razrađene u dijagramu sa sljedećim komponentama: nazivi razreda, nazivi metoda i vrste pristupa metodama (npr. javni, zaštićeni), nazivi atributa razreda, veze i odnosi između razreda.*

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 tijek 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 Selenium[1]*. 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. Programsko inženjerstvo, FER ZEMRIS, http://www.fer.hr/predmet/proinz
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

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

Kontinuirano osvježavanje

U ovom dijelu potrebno je redovito osvježavati dnevnik sastajanja prema predlošku.

1. sastanak
    - Datum: u ovom formatu: 29. listopada 2023.
    - Prisustvovali: I.Prezime, I.Prezime
    - Teme sastanka:
        - opis prve teme
        - opis druge teme
2. sastanak
    - Datum: u ovom formatu: 29. listopada 2023.
    - Prisustvovali: I.Prezime, I.Prezime
    - Teme sastanka:
        - opis prve teme
        - opis druge teme

Tablica aktivnosti

Kontinuirano osvježavanje

Napomena: Doprinose u aktivnostima treba navesti u satima po članovima grupe po aktivnosti.

	Ime Prezime voditelja	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**
Upravljanje projektom							
Opis projektnog zadatka							
Funkcionalni zahtjevi							
Opis pojedinih obrazaca							
Dijagram obrazaca							
Sekvencijski dijagrami							
Opis ostalih zahtjeva							
Arhitektura i dizajn sustava							
Baza podataka							
Dijagram razreda							
Dijagram stanja							
Dijagram aktivnosti							
Dijagram komponenti							
Korištene tehnologije i alati							
Ispitivanje programskog rješenja							
Dijagram razmještaja							

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

	Ime Prezime voditelja	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**	**Ime Prezime**
Upute za puštanje u pogon							
Dnevnik sastajanja							
Zaključak i budući rad							
Popis literature							
Dodatne stavke kako ste podijelili izradu aplikacije							
npr. izrada početne stranice							
izrada baze podataka							
spajanje s bazom podataka							
back end							

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.