

Programsko inženjerstvo

Ak. god. 2023./2024.

BytePit

Dokumentacija, Rev. 1

Grupa: *Rade*

Voditelj: *Marko Bolt*

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

Nastavnik: *Hrvoje Nuić*

Sadržaj

1	Dnevnik promjena dokumentacije	3
2	Opis projektnog zadatka	4
2.1	Primjeri u \LaTeX u	8
3	Specifikacija programske potpore	11
3.1	Funkcionalni zahtjevi	11
3.1.1	Obrasci uporabe	12
3.1.2	Sekvencijski dijagrami	13
3.2	Ostali zahtjevi	14
4	Arhitektura i dizajn sustava	15
4.1	Baza podataka	15
4.1.1	Opis tablica	15
4.1.2	Dijagram baze podataka	16
4.2	Dijagram razreda	17
4.3	Dijagram stanja	18
4.4	Dijagram aktivnosti	19
4.5	Dijagram komponenti	20
5	Implementacija i korisničko sučelje	21
5.1	Korištene tehnologije i alati	21
5.2	Ispitivanje programskog rješenja	22
5.2.1	Ispitivanje komponenti	22
5.2.2	Ispitivanje sustava	22
5.3	Dijagram razmještaja	23
5.4	Upute za puštanje u pogon	24
6	Zaključak i budući rad	25
	Popis literature	26

Indeks slika i dijagrama	27
Dodatak: Prikaz aktivnosti grupe	28

1. Dnevnik promjena dokumentacije

Kontinuirano osvježavanje

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak.	Marko Bolt	04.11.2023.

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

dio 1. revizije

Aplikacija ByteBit inovativno je rješenje namijenjeno provjeri programerskih vještina, omogućavajući korisnicima sudjelovanje u programskim natjecanjima i vježbanje rješavanja zadataka. Aplikacija se može implementirati kao web ili desktop platforma koristeći objektno-orijentirane programerske jezike, sa podrškom za kompilaciju i evaluaciju rješenja u najmanje jednom programskom jeziku.

Glavne funkcionalnosti aplikacije uključuju:

- **Pregledavanje zadataka i kalendar natjecanja:** Korisnici mogu pregledavati dostupne zadatke i kalendar natjecanja, što omogućava planiranje sudjelovanja i pripremu.
- **Profili korisnika:** Natjecatelji imaju profile s prikazom statistika njihovih rezultata, uključujući broj točno riješenih i isprobanih zadataka, kao i pehare za osvojena natjecanja. Profili voditelja sadrže informacije o učitanim zadacima i organiziranim natjecanjima.
- **Registracija korisnika:** Registracija zahtijeva osnovne podatke kao što su korisničko ime, fotografija, lozinka, ime, prezime i email adresa. Administrator potvrđuje registraciju, a u slučaju voditelja natjecanja potrebna je i dodatna verifikacija.
- **Natjecanja:** Početkom natjecanja, zadaci postaju vidljivi aktivnim natjecateljima. Nakon završetka, zadaci su dostupni svima. Postoji mogućnost slanja datoteka s programskim kodom i objavljivanje rang lista po bodovima, uzimajući u obzir vrijeme i točnost rješenja.
- **Prikaz rješenja:** Natjecatelji mogu vidjeti rješenja drugih natjecatelja nakon natjecanja, dok su tijekom natjecanja vidljiva samo ako su sami točno riješili zadatak.
- **Vježbanje:** Natjecatelji mogu vježbati zadatke i učitavati rješenja, s automatskom provjerom točnosti i vremenskih ograničenja.
- **Virtualna natjecanja:** Natjecatelji mogu stvoriti vlastita virtualna natjecanja

odabirom prošlih natjecanja ili nasumičnim odabirom zadataka, a rezultati se uspoređuju s originalnim.

- **Uloga voditelja:** Voditelji imaju mogućnost učitavanja novih zadataka, organiziranja natjecanja, i odabira zadataka koji će biti aktivni tijekom natjecanja, uz mogućnost uređivanja vlastitih zadataka.
- **Administrativne funkcije:** Administratori imaju ovlasti za upravljanje korisničkim računima i uređivanjem svih zadataka i natjecanja, bez mijenjanja prethodno ostvarenih rezultata.

Trenutno postoje brojne platforme poput Codeforces, LeetCode, HackerRank i drugih koje nude slične funkcionalnosti. Razlike u odnosu na BytePit bi se mogle ogledati u specifičnostima kao što su lokalizacija (jezik i regionalni fokus), specijalizacija za određene programerske jezike ili tehnologije, ili u razini personalizacije korisničkog iskustva. Slična rješenja i razlike:

- **Codeforces:** Fokusira se na konkurentno programiranje s redovitim natjecanjima. BytePit bi mogao uključivati više edukativnih resursa i prilagodljiva virtualna natjecanja.
- **LeetCode:** Specijaliziran za pripremu tehničkih intervjua. BytePit može nuditi širi raspon zadataka, uključujući one van konteksta intervjua.
- **HackerRank:** Osim izazova u programiranju, nudi i podršku za tvrtke u regrutiranju. BytePit bi mogao dodati vrednost kroz specifične profile natjecanja i detaljnu statistiku. Sličan koncept, ali s fokusom na poslovne korisnike. BytePit bi mogao biti više usmjeren na edukaciju i natjecanja.

BytePit bi mogao privući širok spektar korisnika, od početnika u programiranju koji traže mjesto za učenje i prakticiranje, do iskusnih programera koji žele oštriti svoje vještine ili sudjelovati u natjecanjima, te edukativnih institucija koje traže platformu za organiziranje natjecanja ili kao alat u nastavi. Ova raznolikost korisnika stvara dinamično okruženje u kojem se znanje i iskustvo neprestano razmjenjuju, pružajući beskrajne mogućnosti za učenje i napredak. S jedne strane, početnici imaju priliku učiti kroz praksu, rješavajući stvarne probleme i dobivajući povratne informacije od iskusnijih kolega. S druge strane, iskusni programeri mogu se suočavati s izazovnijim problemima i sudjelovati u zahtjevnijim natjecanjima, što ih stavlja u poziciju mentora, potičući ih na daljnji razvoj i dijeljenje znanja. Edukativne institucije mogu koristiti BytePit za stvaranje prilagođenih

tečajeva i kurikuluma, pružajući studentima praktično iskustvo koje je ključno za njihov razvoj. Poslodavci koji traže talentirane programere mogli bi koristiti platformu za identifikaciju potencijalnih kandidata, provodeći natjecanja ili izazove koji služe kao dio procesa zapošljavanja ili kroz analizu rješenja i pristupa problemima koje programeri koriste na platformi.

BytePit bi trebao biti projektiran s mogućnostima prilagodbe, što bi uključivalo konfigurabilne module za natjecanja, personalizirane profile korisnika, i fleksibilan sustav ocjenjivanja. Prilagodba omogućava korisnicima da kreiraju jedinstveno iskustvo koje odgovara njihovim specifičnim potrebama i stilu učenja ili rada. Personalizirani profili omogućavaju korisnicima da predstave svoje vještine i postignuća, kao i da prate svoj napredak kroz različite metrike i statistike. Fleksibilan sustav ocjenjivanja omogućava različite forme evaluacije, od standardnog bodovanja do složenih algoritama koji mogu vrednovati efikasnost koda ili kreativno rješavanje problema. Ovo omogućava platformi da raste i evoluiraju zajedno sa svojim korisničkim bazama i tehnološkim trendovima, ostajući relevantna i privlačna za široki spektar korisnika, od individualaca i obrazovnih institucija do korporacija i tehničkih zajednica.

Opseg projekta uključuje dizajniranje i implementaciju korisničkih sučelja, koje moraju biti intuitivna i pristupačna, osiguravajući ugodno iskustvo za sve korisnike, neovisno o njihovom tehničkom znanju ili iskustvu u programiranju. Backend servisi za upravljanje bazom podataka moraju biti robustni i optimizirani za brzu obradu upita, što je ključno za održavanje visoke performanse platforme čak i pod velikim opterećenjem. Sustav za evaluaciju koda mora biti precizan i pouzdan, sposoban za brzo i efikasno procjenjivanje raznovrsnih programskih rješenja koja korisnici predaju. Sustav za autentifikaciju i autorizaciju korisnika treba osigurati da su svi podaci sigurni i da pristup platformi

Nadogradnje projekta mogu uključivati:

- Integraciju s vanjskim servisima poput GitHub-a za uvoz postojećeg koda.
- Implementaciju naprednijih algoritama za detekciju plagijata.
- Proširenje podrške za više programskih jezika i tehnoloških okvira.
- Razvoj mobilne aplikacije za pristup platformi.
- Uvođenje umjetne inteligencije za personalizirane preporuke zadataka na te-

melju korisnikovih prethodnih aktivnosti i uspjeha.

- "Gamification" sustav za nagrađivanje korisnika za postignuća i napredak.

2.1 Primjeri u L^AT_EXu

Ovo potpoglavlje izbrisati.

U nastavku se nalaze različiti primjeri kako koristiti osnovne funkcionalnosti L^AT_EXa 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 L^AT_EXu - https://www.fer.unizg.hr/_download/repository/LaTeX-upute.pdf
- L^AT_EX 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	

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

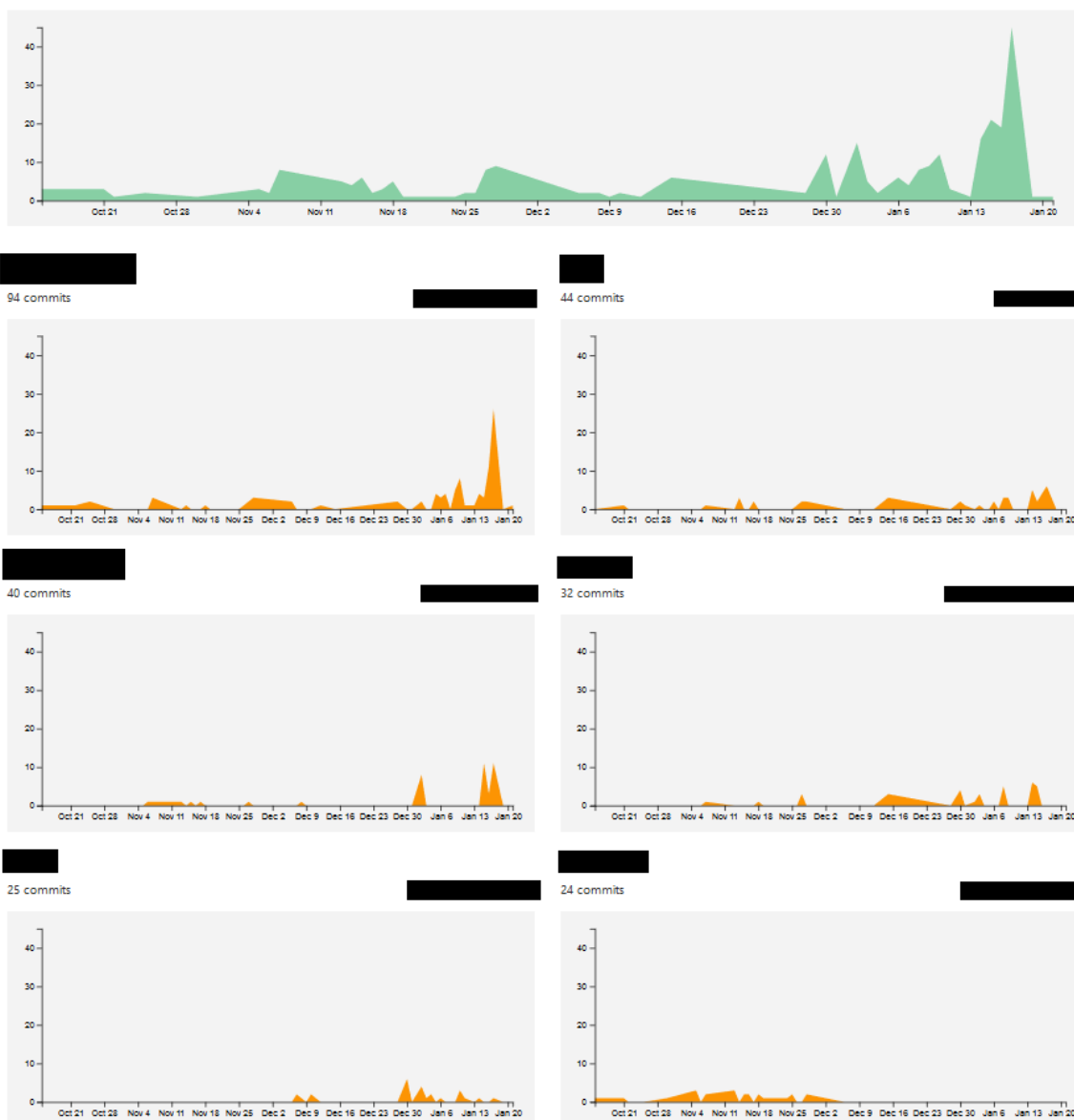
naslov unutar tablice		
ime	VARCHAR	
primjer	VARCHAR	

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 2.1: Primjer slike s potpisom



Slika 2.2: Primjer slike s potpisom 2

Referenciranje slike 2.2 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 tijekom 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-programске 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 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. 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

2.1	Primjer slike s potpisom	9
2.2	Primjer slike s potpisom 2	10

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: 4. studenoga 2023.
- Prisustvovali: I.Prezime, I.Prezime
- Teme sastanka:
 - opis prve teme
 - opis druge teme

2. sastanak

- Datum: u ovom formatu: 4. studenoga 2023.
- Prisustvovali: I.Prezime, I.Prezime
- Teme sastanka:
 - opis prve teme
 - opis druge teme

Tablica aktivnosti

Kontinuirano osvježavanje

Napomena: Doprinosi 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							
<i>Dodatne stavke kako ste podijelili izradu aplikacije</i>							
<i>npr. izrada početne stranice</i>							
<i>izrada baze podataka</i>							
<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.