

Sveučilište Juraja Dobrile u Puli
Fakultet informatike

REPOZITORIJ SOFTVERSKIH PROJEKATA

Projektna dokumentacija

Pula, 15. siječnja, 2019. godine

Sveučilište Juraja Dobrile u Puli
Fakultet informatike

REPOZITORIJ SOFTVERSKIH PROJEKATA

Projektna dokumentacija

Projektni tim:

Mladen Šverko, JMBAG: 0016076929

Dario Stastny, JMBAG: 0253005697

Marko Stupar, JMBAG: 0303046281

Kolegij: Programsko inženjerstvo

Mentor: doc. dr. sc. Tihomir Orehovački

Pula, , 15. siječnja, 2019. godine

Sadržaj

1.	Sažetak.....	4
2.	Uvod.....	4
3.	Motivacija.....	6
3.1	SWAT analiza.....	6
3.2	Daljnja poboljšanja	9
4.	Razrada funkcionalnosti.....	9
4.1	Use-case dijagram	9
4.2	Dijagrami slijeda (eng. <i>sequence diagram</i>)	11
4.3	Prototip	15
4.4	Klasni dijagram.....	18
5.	Implementacija.....	19
5.1	Softversko okruženje.....	19
5.2	Klase.....	19
1.1	Baza podataka.....	22
2.	Korisničke upute.....	22
3.	Popis slika	31
4.	Popis tablica.....	31
5.	Literatura	31

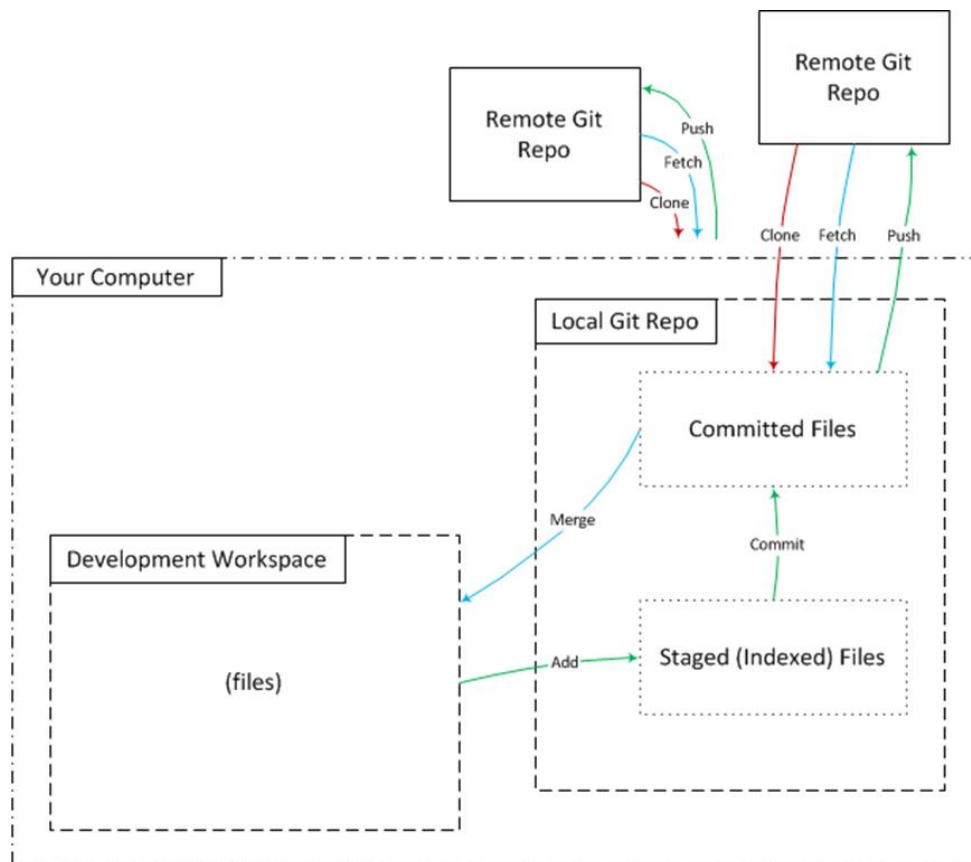
1. Sažetak

Aplikacija „Software projects repository“ predstavlja relativno pojednostavljenu verziju repozitorija kojom se nastoji predstaviti osnovne funkcionalnosti, karakteristične za repozitorij lociran na razini lokalne mreže unutar tvrtke, tj. lokalnog razvojnog okružja. Pripadna projektna dokumentacija ima za cilj pružiti sve relevantne informacije počevši od tehničkog djela kroz opis same aplikacije, razrade funkcionalnosti i implementacije, pa do ciljanog tržišta i SWOT analize sa navedenim tržišnim prednostima i nedostacima. U nastavku će, gore navedeno, biti strukturirano u zasebnim cjelinama, te ima za cilj pružiti cjelokupnu sliku razvoja projekta, zaključno sa korisničkim uputama.

2. Uvod

Premda danas postoje mnogobrojne aplikacije koje u okviru repozitorija nude pregršt dodatnih mogućnosti koje najčešće i znatno premašuju potrebe korisnika, u suštini, pojam repozitorij odnosi se na centralno mjesto pohrane i ažuriranja/aktualiziranje podataka (eng. *version control*) gdje se podatci mogu čuvati u višestrukim bazama podataka ili direktorijima. Takvo centralno mjesto pohrane može biti locirano na lokalnoj mreži, ili bilo gdje na internetu, ovisno o potrebama korisnika.

„Softver repozitorij, kolokvijalno poznat pod kraticom repo, jeste lokacija za pohranu odakle se softverski paketi mogu dohvatiti i instalirati na lokalno računalo“ (https://en.wikipedia.org/wiki/Software_repository, 2018-11-09).

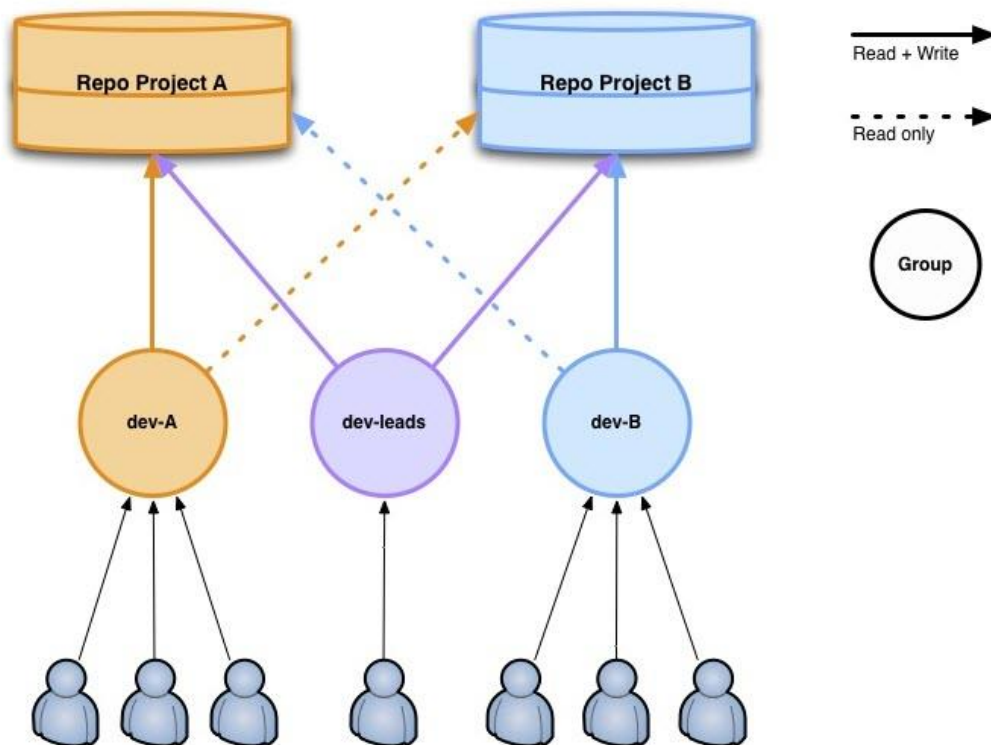


Slika 1: Osnovni koncept GitHub repozitorija

Izvor: <https://www.intertech.com/Blog/introduction-to-git-concepts>

Gornja slika prikazuje temeljni koncept i funkciju GitHub-a, trenutno jednog od najpoznatijih repozitorija. Neke druge aplikacije mogu se i znatnije razlikovati u raznim opcijama i funkcionalnostima, ali temeljni model se svodi na mogućnost dohvaćanja sadržaja sa centralne lokacije, što može biti tek sa ciljem uvida u dokument bez potrebe za izmjenom, ili pak sa namjerom da se sadržaj mijenja. Gledano sa korisničke strane, osnovni model rada razlikuje dva ključna tipa korisnika:

- voditelj tima (eng. *project manager, team-lead, lead engineer...*)
- programer, član razvojnog tima, razvojni inženjer (eng. *developer*)



Slika 2: Korisničke uloge

Izvor:

https://www.atlassian.com/blog/archives/fisheye_in_practice_setting_up_your_own_git_repositories

Prema gornjoj slici, voditelj razvojnog tima posjeduje razinu ovlaštenja koja mu dozvoljava intervencije u oba projekta, dok članovi razvojnih timova imaju prava pisanja isključivo nad sadržajem svojih projekata. Gore prikazan model i korisničke uloge dva su ključna elementa i temelj za razvoj predmetne aplikacije „Software projects repository“.

Ciljano tržište za takvu aplikaciju nalazi se u velikom broju malih softverskih tvrtki i razvojnih timova, te pojedinaca (*freelancer*) koji razvijaju relativno jednostavna softverska rješenja i nemaju potrebu za kompleksnijim aplikacijama koje se nude na tržištu. Krajnji korisnici su programeri (*developeri*), koji izravno manipuliraju projektnom dokumentacijom i vrše konstantno ažuriranje iste tijekom cjelokupnog razvojnog procesa. Uporabom aplikacije „Software projects repository“ omogućuje se višekorisnički rad, ušteda vremena, te eliminiranje problema dvostrukih dokumenata i višestrukih izmjena uz *version control*.

3. Motivacija

Već je u samom uvodu spomenuto kako danas postoji velik broj rješenja za repozitorij sa mnogobrojnim dodatnim mogućnostima povezanim sa arhiviranjem i praćenjem projekata, te raznim stupnjevima integracije sa razvojnim sučeljima (Integrated Development Interface) kao što su Visual studio, Visual studio code, IntelliJ IDEA, Eclipse, Atom.io, Glitch, Xcode i sl. Sva od navedenih razvojnih sučelja mogu se integrirati sa GitHub repozitorijem, a uz to postoji i Git-Gui ili Tortoise GIT koji se sa mogućnošću integriranja na razini OS-a.

Međutim, najveći dio dostupnih aplikacija nudi i previše dodatnih opcija koje često i nisu toliko opcionalne, u smislu da se ne mogu ukloniti sa ekrana ako ih korisnik doživljava kao suvišna ili čak kao smetnju u redovnom radu. Nadalje, većina aplikacija podrazumijeva da se repozitorij kao lokacija nalazi na negdje u oblaku, što ne mora nužno odgovarati svakom korisniku. Aplikacija „Software projects repository“, osim svoje osnovne funkcije repozitorija ima i dodatnu opciju u praćenju razvojnog ciklusa softverskih projekata koja vrlo rijetko ili uopće ne postoji u većini dostupnih aplikacija. Naime, jedna od ključnih faza u razvoju softvera(informacijskog sustava) je svakako puštanje u rad istog.

Ne treba posebno napominjati da se radi o fazi gdje u praksi dolazi do otkrivanja raznih nedostataka, te česte potrebe za potporom razvojnog tima. Osobito je to slučaj kada osobe(eng. *commissioning engineer*) koje puštaju sustav u rad nisu bile neposredno uključene u razvoj istog, ili barem nisu u onom segmentu gdje je nedostatak otkriven.

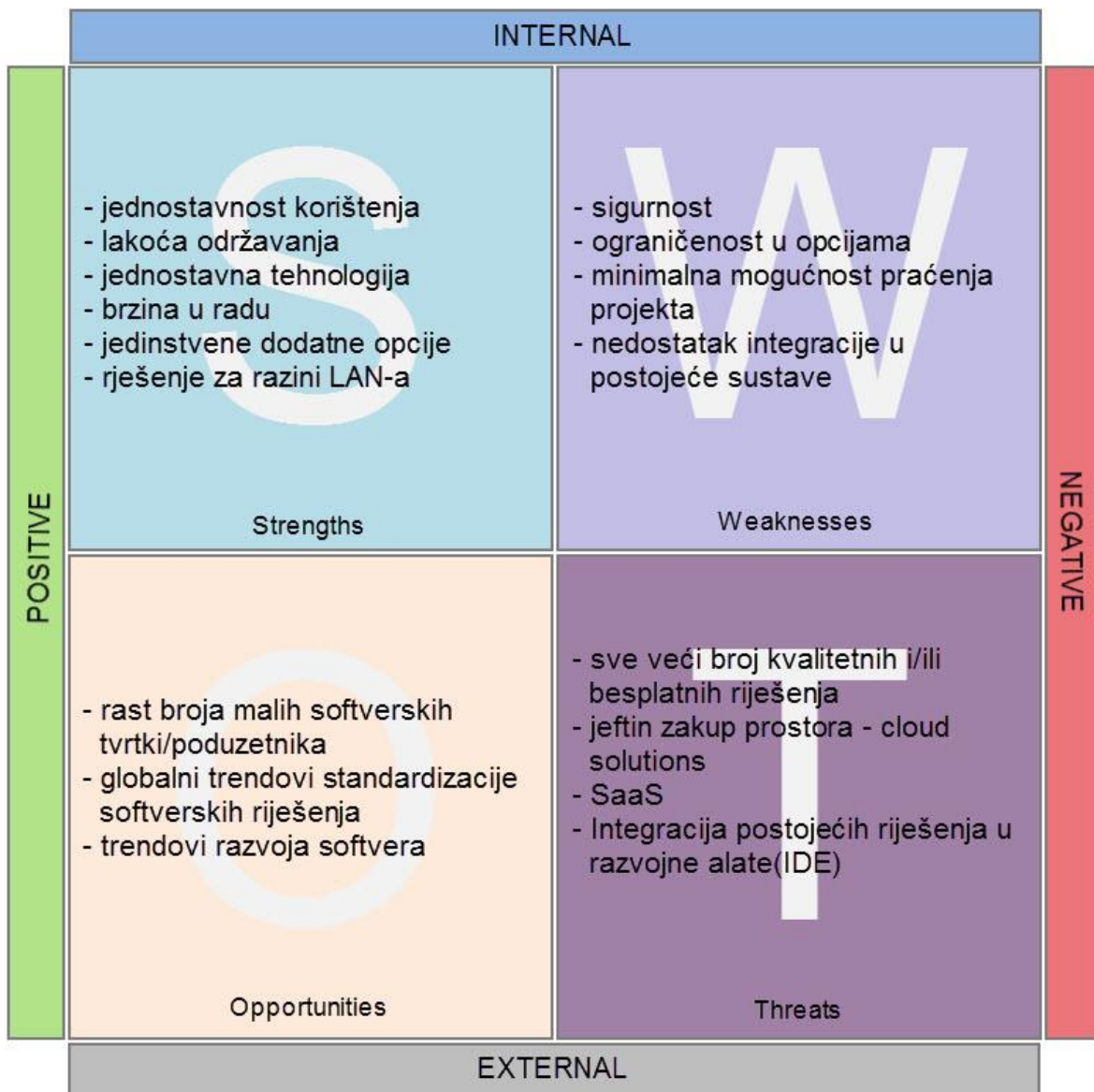
Kako je riječ o završnoj fazi razvoja sustava, često se takvi nedostaci otklanjaju u žurbi i pod pritiskom vremenskih rokova, pri čemu nerijetko izostaje dostatno dokumentiranje i time se znatno povećava rizik od ponavljanja sličnih grešaka u budućim projektima.

Ova aplikacija na jednostavan i intuitivan način omogućava da se takvi slučajevi otkrivanja nedostataka i potrebe za potporom tijekom puštanja u rad evidentiraju putem zahtijeva za potporom, te dodatno prema potrebi potkrijepe dokumentacijom i kao paket trajno arhiviraju u repozitorij kao dio projekta.

Imajući u vidu navedeno, ali i trend porasta broja malih softverskih tvrtki ili obrtnika kojima ne mora nužno biti potrebno neko kompleksno rješenje, dolazimo do zaključka kako na tržištu postoji prostora za jednostavno rješenje koje je u stanju obavljati funkciju repozitorija unutar lokalne mreže i to na jednostavan način bez suvišnih podešavanja sustava ili instalacije softvera, te istovremeno pružiti mogućnost praćenja projekata razvoja softvera u fazi puštanja u rad. Navedeno ne mora nužno, kao potencijalne korisnike, isključiti srednje i veće tvrtke. Imajući u vidu znatnu razliku u razvojnim procesima i procedurama kod razvoja softverskih rješenja za različite namjene i područja (transport, farmaceutika, industrijska automatizacija, bankarski sektor i sl.) proizlaze i znatne razlike među odjelima iste tvrtke, koji često djeluju sa velikim stupnjem autonomije i bitno se međusobno razlikuju. U tom slučaju, također postoji mogućnost da bi rješenje koje predstavlja predmetna aplikacija moglo biti prednost u odnosu na neki od kompleksnijih centraliziranih rješenja na razini tvrtke.

3.1 SWAT analiza

SWAT analizom na pregledan se način utvrđuju prednosti i nedostaci aplikacije kao proizvoda sa ciljem boljeg razumijevanja mogućnosti i opcija koje proizvod ima na tržištu, te isticanja razlika u odnosu na konkurentske proizvode.



Slika 3: SWAT analiza

Prikazana SWAT analiza rezimira većinu dosad iznesenih karakteristika razvijene aplikacije. Sve su poznate prednosti i nedostaci (eng. *strengths/weaknesses*) prikazane u gornjem djelu slike kao unutarnja svojstva proizvoda, te prilike i prijetnje (eng. *opportunities/threats*) kao vanjska svojstva koja imaju, ili se očekuje da imaju utjecaj na plasman proizvoda na tržištu.

Prednosti:

- jednostavnost korištenja – aplikacija nije opterećena brojnim dodatnim opcijama i sučelje je dovoljno jednostavno i intuitivno da korisniku omogući brzu i laganu prilagodbu bez dodatnog napora.
- lakoća održavanja – održavanje aplikacije se uglavnom odnosi na unos i brisanje podataka u bazi, što je lako izvedivo sa predviđenim korisničkim ovlastima na razini projekt menadžera.
- jednostavna tehnologija – riječ je od *stand-alone desk top* aplikaciji koja koristi lokalnu SQLite bazu podataka, čime je izbjegnuta ikakva potreba za instaliranjem softvera.

- brzina u radu – cjelokupno rješenje se sastoji od jednog jedinstvenog programskog modula razvijenog u C# jeziku, kojemu je jedina vanjska komunikacija prema bazi na LAN-u, i datotečnom sustavu OS-a.
- jedinstvene dodatne opcije – kako je već prethodno navedeno, dodatna opcija zahtijeva za potporom tijekom puštanja u rad je nešto što druge aplikacije na tržištu ne nude, a može biti od presudnog značaja za poboljšanje kvalitete budućih projekata koji se razvijaju unutar repozitorija.
- rješenje na razini LAN-a – može predstavljati prednost u odnosu na dostupna internetska rješenja ukoliko korisnik želi svoje podatke zadržati unutar vlastitog sustava, te u slučajevima kada postoji problem sa brzinom pristupa internetu.

Nedostaci:

- sigurnost – ne postoji nikakva enkripcija podataka, kao ni lozinki koje su nezaštićene i jednostavno pohranjene u bazu podataka. Također ne postoje nikakve restirkcije glede unosa sigurnosno rizičnih lozinki (eng. *weak passwords*).
- ograničenost u opcijama – osim opcije zahtijeva za potporom tijekom puštanja u rad, ne postoje dodatne opcije.
- minimalna mogućnost praćenja projekta - ne postoje dodatne statističke opcije praćenja projekta koje mogu biti od koristi projekt menadžeru.
- nedostatak integracije u postojeće sustave – kako je već prije spomenuto, aplikacija nema mogućnosti integracije u IDE, OS i slično, što može donekle ugroziti jednostavnost korištenja.

Uz gore navedene prednosti i nedostatke samog programskog rješenja, uočeni su slijedeći trendovi koji mogu ići u prilog ili na štetu plasmana na tržištu.

Prilike:

- rast broja malih softverskih tvrtki/poduzetnika – s obzirom da je velik dio rješenja na tržištu relativno kompleksan, a ponekad i zahtjevan za korištenje, za pretpostaviti je da će malim tvrtkama koje razvijaju relativno jednostavne projekte, jednostavnije rješenje koje zadovoljava njihove osnovne potrebe biti prihvatljivije.
- globalni trendovi standardizacije softverskih rješenja – posljednjih godina, na području informatičke industrije, postoji cijeli niz inicijativa koje idu u smjeru zakonskih regulativa, a sa ciljem postavljanja određenih okvira glede kvalitete, sigurnosti i standardizacije općenito za informacijske sustave i softverske proizvode općenito. S time u vidu, razvojnim timovima je u interesu da uvedu reda u razvojni proces, gdje pravilno korišteno programsko rješenje za repozitorij i praćenje puštanja u rad, može biti od koristi.
- trendovi razvoja softvera – današnji trendovi podrazumijevaju relativno brz razvoj koji rezultira čestim izmjenama dokumentacije „u hodu“. Takvim razvojnim tehnikama uz višestruke iteracije, lako dolazi do preklapanja sadržaja, te dvostrukim dokumentima. Takve se situacije lako izbjegavaju korištenjem repozitorija uz *version control*.

Prijetnje:

- sve veći broj kvalitetnih i/ili besplatnih rješenja – trenutno je u ponudi velik broj aplikacija od kojih su mnoge besplatne i dovoljno kvalitetne da pruže korisniku sve što mu je potrebno. Takvo stanje na tržištu znatno ugrožava poziciju predmetne aplikacije.
- jeftin zakup prostora (*cloud solutions*) – uz vrlo nisku cijenu zakupa prostora na

internetu, velika je vjerojatnost izbora nekog od internetskih rješenja za repozitorij.

- SaaS – rastom ponuda softvera kao usluge, čime je repozitorij jedna od komponenti, eliminira se potreba za zasebnim rješenjem.
- integracija postojećih rješenja u razvojne alate(IDE) – Za očekivati je da će timovi koji razvijaju softver u bilo kojem poznatijem razvojnom sučelju, prvenstveno biti zainteresirani za korištenje repozitorija čije se sučelje može integrirati u isti, i time ubrzati/olakšati rad.

3.2 Daljnja poboljšanja

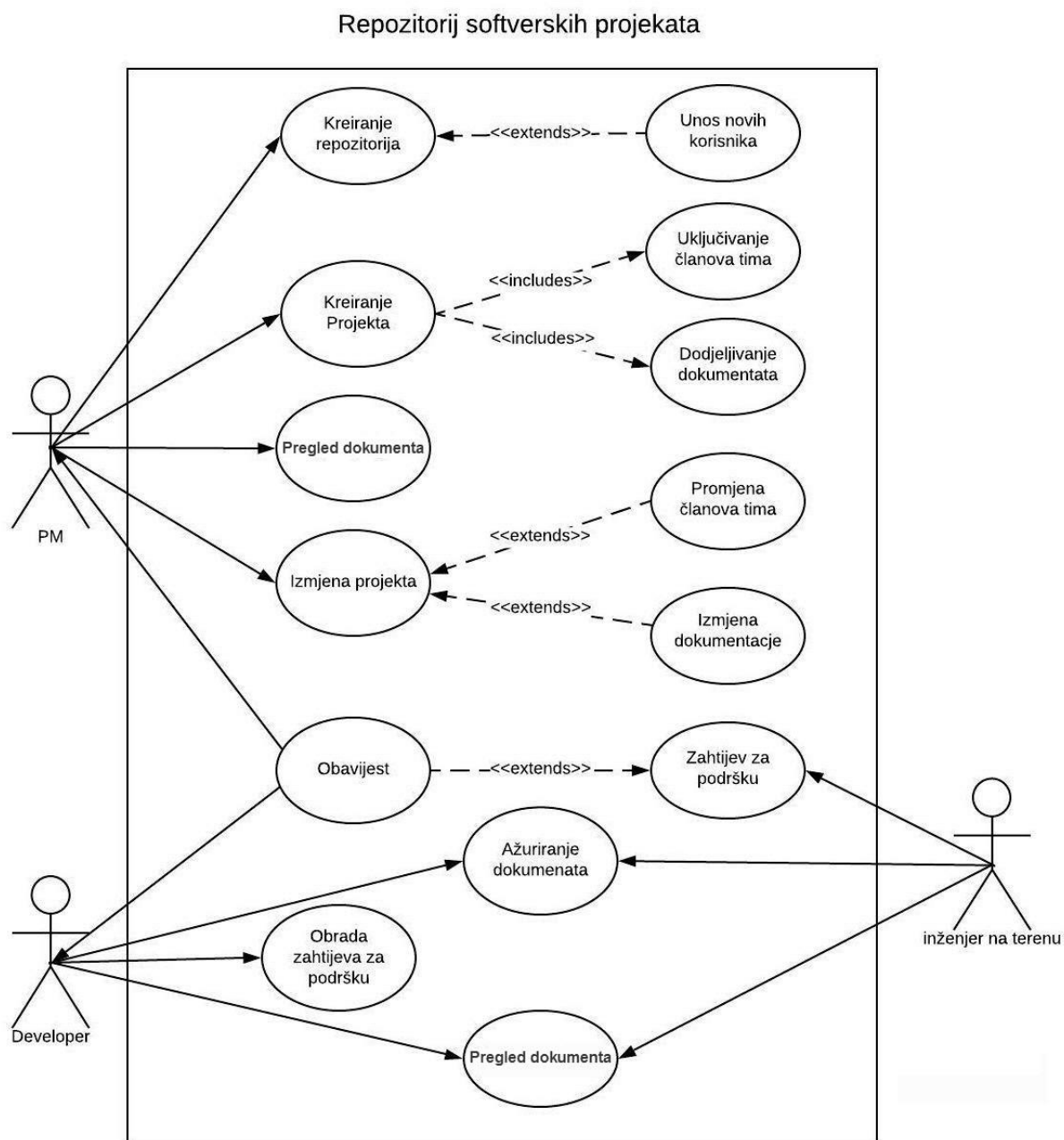
U daljnjem razvoju aplikacije potrebno je imati u vidu trendove razvoja standarda, osobito po pitanju sigurnosti podataka, te sukladno tome implementirati neki od algoritama enkripcije podataka, kao i poboljšati sigurnost trenutnih lozinki. Nadalje, imajući u vidu potrebe praćenja razvojnih faza projekata od strane projekt menadžera, evidentan je nedostatak dodatnih opcija koje bi pružile relevantne podatke vezane uz produktivnost i utrošak ljudskih resursa za pojedini projekt. Većina tih podataka je već sada dostupna u bazi, te ih je potrebno adekvatno grafički prikazati. Budući je repozitoriju moguće pristupiti i izravno putem direktorija, te na taj način zaobići aplikaciju u potpunosti, bilo bi korisno da postoji mogućnost oporavka (*eng. recovery*) aplikacije. Oporavak bi bio izveden u smislu usklađivanja sadržaja u bazi podataka sa realnim stanjem u direktorijima.

4. Razrada funkcionalnosti

Razrada funkcionalnosti projekta biti će u nastavku teksta pojašnjena putem korištenih UML dijagrama kojima će se pobliže objasniti interakcija korisnika i aplikacije(*use-case diagram*), funkcije aplikacije(*sequence diagram*), objekti i veze među njima(*class diagram*), te relacijski model baze podataka(*ER-diagram*). U konačnici, cjelokupna slika biti će upotpunjena prototipom sučelja aplikacije.

4.1 Use-case dijagram

Riječ je o ponašajnom dijagramu koji ima za cilj prikazati ponašanje sustava iz perspektive korisnika.



Slika 4: Use-case dijagram

U interakciji korisnika i aplikacije definirana su tri tipa korisnika od kojih je svaki u mogućnosti izvršiti određene aktivnosti u okviru aplikacije:

- PM – projekt menadžer
- Programer (*developer*)
- Inženjer na terenu (*commissioning engineer*)

Projekt menadžer(PM):

Može kreirati repozitorij, što ujedno predstavlja prvi korak u radu sa aplikacijom. Bez kreiranog mjesta za pohranu podataka logično je da projekti ne mogu biti kreirani. Nakon što je to učinjeno, moguće je unutar definiranog repozitorija kreirati projekt. Za kreirani projekt, mogu se dodati članovi tima birajući iz postojećih koji se nalaze u bazi (uključivanje članova tima). Ukoliko je to potrebno, PM može prethodno dodati u

bazu nove korisnike(unos novih korisnika). Kreiranjem projektu potrebno je dodijeliti sadržaj, odnosno dokumente (dodjeljivanje dokumenata).

Jednom kada je projekt kreiran i članovi su uključeni u tim, te su pridodani dokumenti, moguće je u isti intervenirati izmjenom navedenog (promjena članova tima, izmjena dokumentacije). Također je na raspolaganju pregled projekta, što razumljivo uključuje pregled dokumenata koji mogu biti izmijenjeni od strane drugih korisnika. Kao posljedica aktivnosti inženjera na terenu (commissioning engineer), PM može primiti poruku u svezi otvorenog zahtjeva za potporom.

Programer (*developer*):

Ima na raspolaganju aktivnosti vezane uz projekt koji mu je dodijeljen. Te aktivnosti mogu se odnositi na izmjenu(ažuriranje) dokumenata koji su pridodani projektu, ili samo uvid u dokumente bez potrebe za izmjenama (pregled projekta). Osnovna razlika navedenih aktivnosti je u zaključavanju dokumenta na razini repozitorija. Naime, ukoliko se dokument otvori samo za pregled, isti se prebacuje u privremeni direktorij odakle se otvara zadanom aplikacijom za danu ekstenziju dokumenta. Jednom otvoren u zadanoj aplikaciji, dokument se može mijenjati, ali to neće imati učinka na dokument koji se nalazi na repozitoriju. Nakon završenog pregleda, dokument u privremenom direktoriju će se izbrisati(čime će biti izgubljene i eventualne izmjene). U slučaju da se dokument sa repozitorija otvori za ažuriranje/izmjene, tada će se na razini repozitorija isti označiti kao zaključan, i takav će ostati dok korisnik ne potvrdi učinjene izmjene (*commit*). Tek nakon toga se izmijenjeni dokument iz privremenog direktorija prebacuje u matični direktorij ne repozitoriju i otključava se za daljnje korištenje.

Pod aktivnošću pregleda projekta podrazumijeva se i pregled pristiglih zahtjeva za potporom od strane inženjera na terenu, na koje programer može reagirati(obrađivanje zahtjeva za podršku). Pristigli zahtjevi za podršku prikazuju se u tabelarnom formatu u okviru odabranog projekta. Svaki od članova razvojnog tima ima uvid, te može poduzeti radnje potrebne za rješavanje pristiglog zahtjeva. Ukoliko neki od članova tima riješi zahtjev za potporu, isti se označava kao riješen.

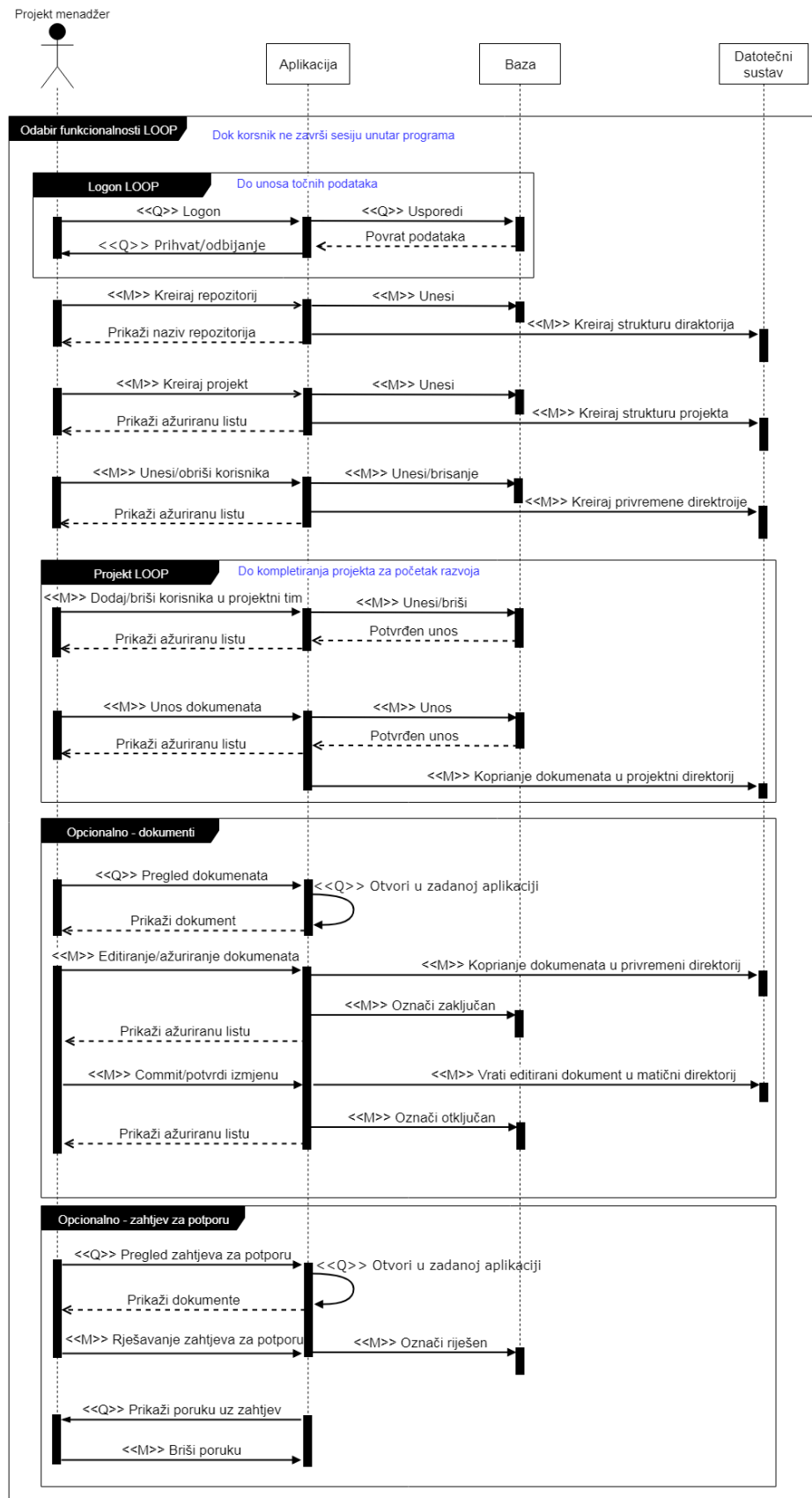
Inženjer na terenu(*commissioning engineer*):

ima mogućnosti intervencije u dodijeljeni projekt kao i programer, sa izuzetkom zahtjeva za potporom gdje su uloge suprotne. Inženjer na terenu inicira zahtjev za podrškom(Zahtjev za podršku) u koji može priložiti dodatne dokumente, i koji se nakon toga pohranjuje u repozitoriju unutar pripadnog projekta u zasebnom direktoriju. Uz generiranje zahtjeva za podršku, na raspolaganju je i dodatna mogućnost slanja poruke(obavijest) PM-u ili nekom od developera.

4.2 Dijagrami slijeda (eng. *sequence diagram*)

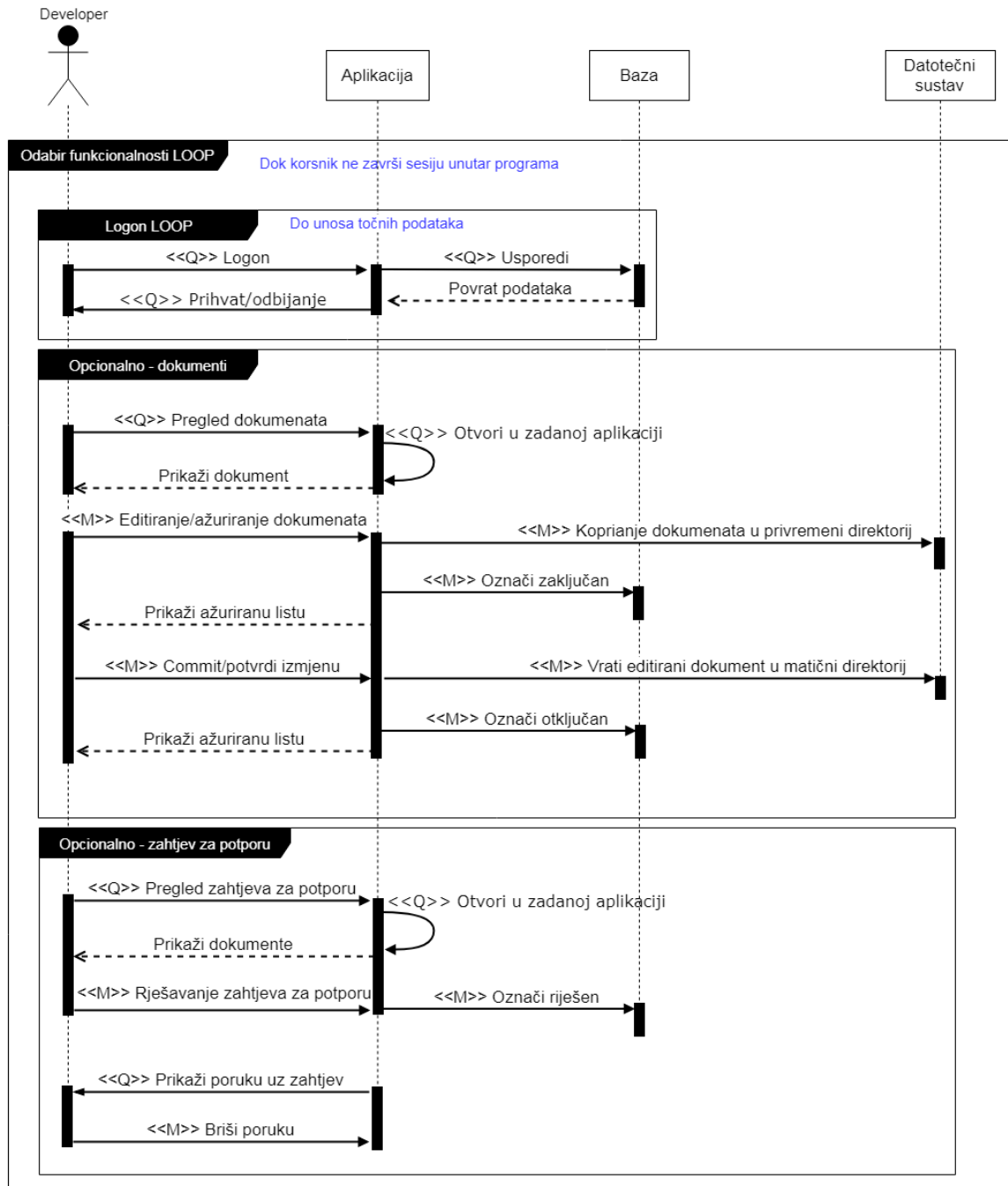
Dijagrami nastavku prikazati će raspoložive korisničke scenarije putem kojih će se pojasniti funkcionalnosti sustava.

PM sequence diagram:



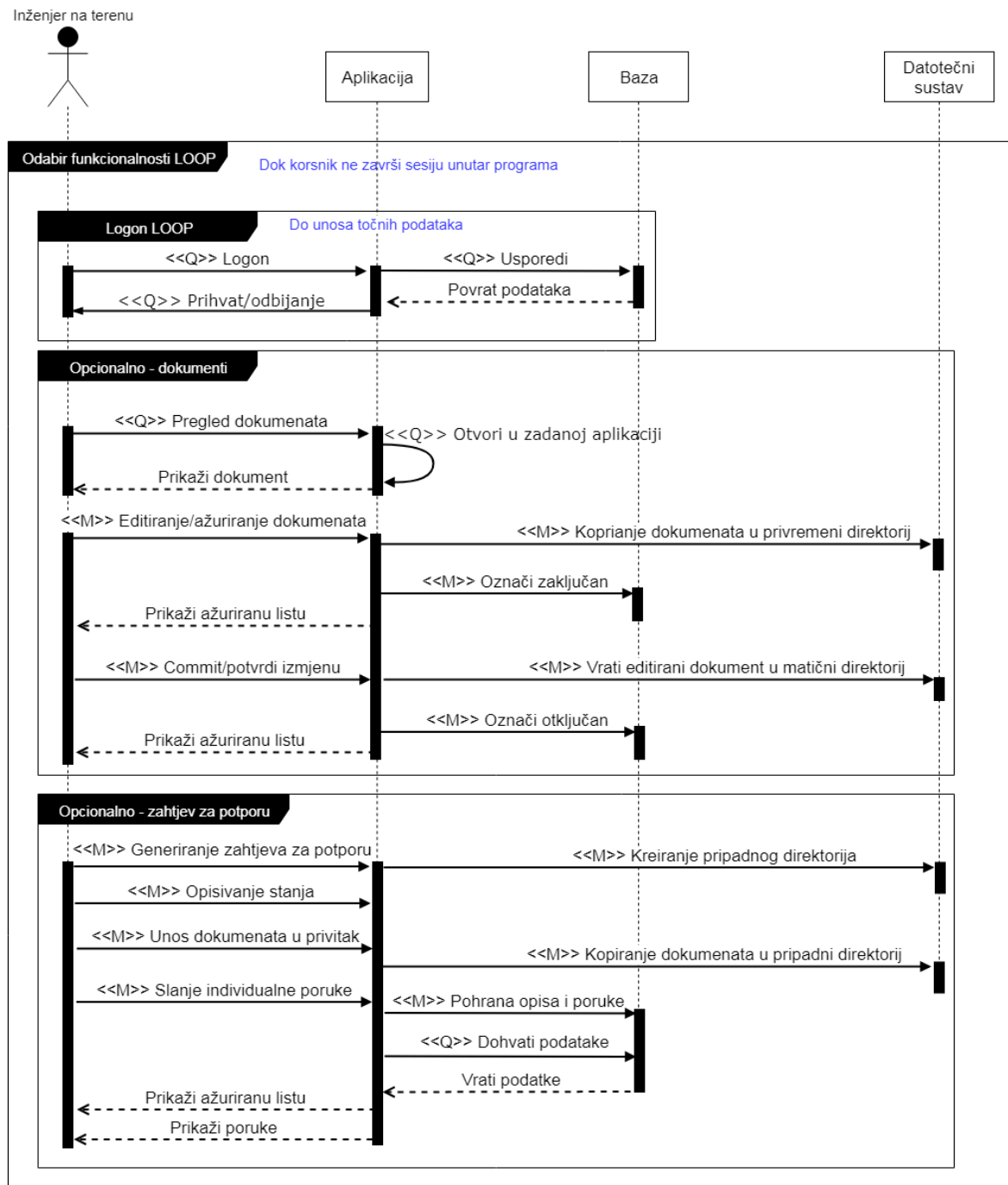
Slika 5: PM sequence diagram

Developer sequence diagram:



Slika 6: Developer sequence diagram

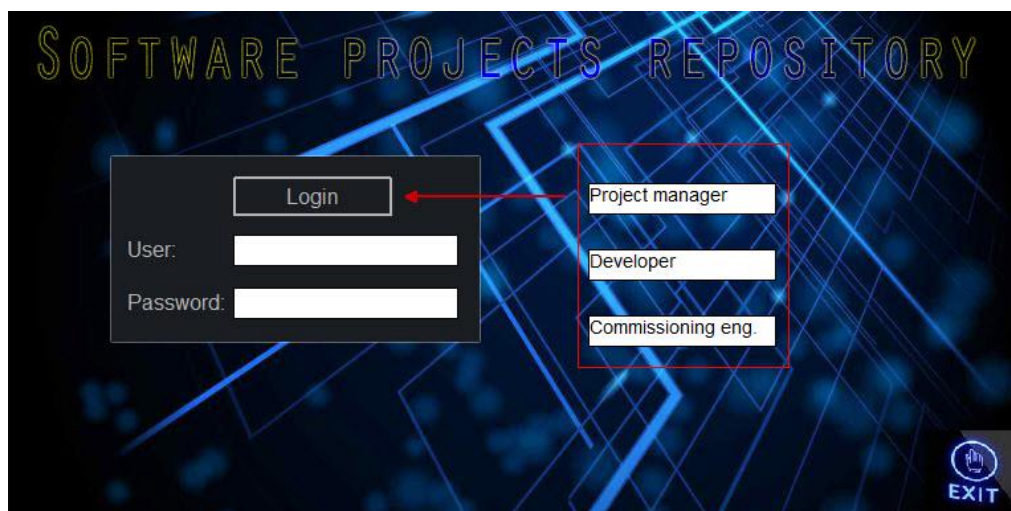
Inženjer na terenu sequence diagram:



Slika 7: Inženjer na terenu sequence diagram

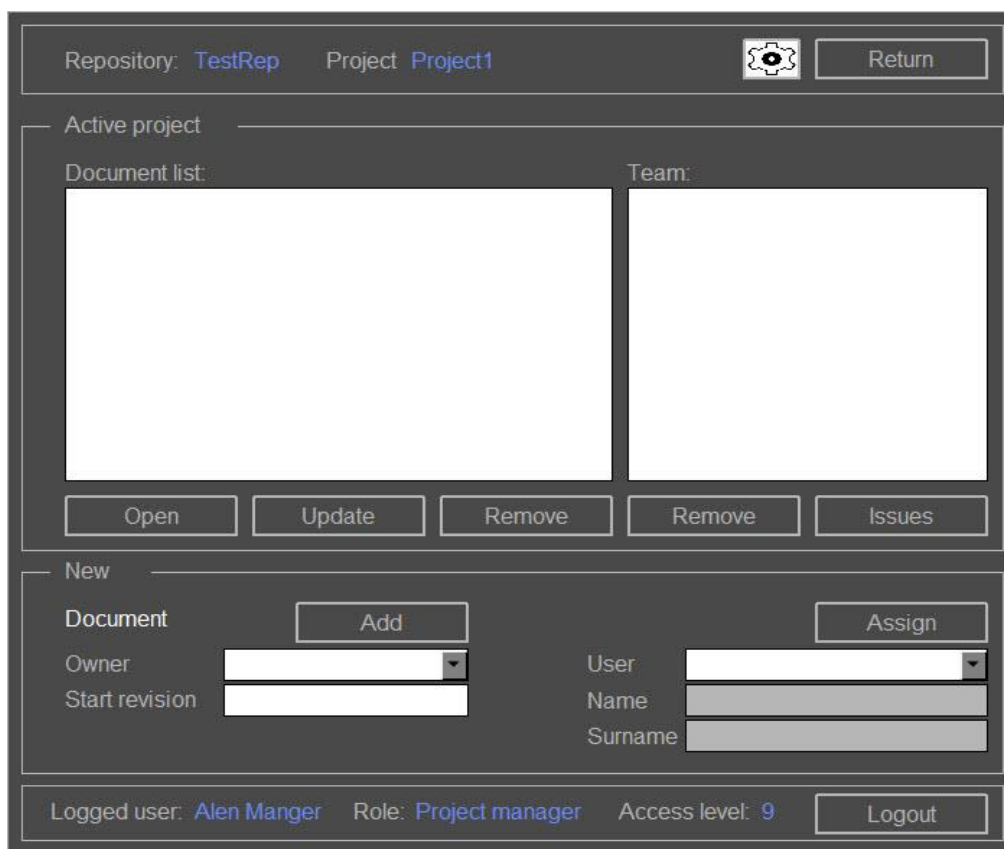
4.3 Prototip

Prototip predstavlja nacrt grafičkog rješenja sučelja u fazi modeliranja, te ima za cilj pojasniti korisničke opcije na slikovit i jednostavan način. Time se, uz olakšano razumijevanje iz korisničke uloge, dodatno otvara mogućnost pravovremenih intervencija prije faze implementacije.



Slika 8: Login forma

Ovisno o tome koja je razina autorizacije prijavljenog korisnika, u radu sa repozitorijem, odgovarajuće će funkcije biti onemogućene, te će se prikazati odgovarajuća početna forma.



Slika 9: Početna forma – Projekt menadžer

Početna forma odgovara prijavljenom korisniku sa razinom pristupa 9 (viši broj znači i višu razinu pristupa). Osnovni podatci poput naziv repozitorija, lista projekata i osnovni podatci o prijavljenom korisniku su zajednički za sve korisnike. Na alatnoj traci vidljivo je dugme za pristup postavkama što odgovara pravima pristupa prijavljenog korisnika. svi ostali elementi na formi služe za vođenje projekta (definiranje razvojnog tima i pripadne dokumentacije koja će biti dostupna na razini projekta).

Repository: [TestRep](#) [Return](#)

Team member

[Add](#)

Name

Surname

E-mail

Role

Access level

[Delete](#)

User

Name

Surname

Project

[Create](#)

Project name

Created: [No new projects created](#)

[Assign](#)

User

Name

Surname

Repository

[Create](#)

Name

Logged user: [Alen Manger](#) Role: [Project manager](#) Access level: [9](#) [Logout](#)

Slika 10: Forma postavke

Formi postavke se pristupa sa početne forme uz razinu pristupa > 8, i sadrži ključne funkcionalnosti za rad sa repozitorijem, uključujući i kreiranje istog što je nužno pri prvom pokretanju aplikacije.

Repository: [TestRep](#) Project [Project1](#) [Return](#)

Active project

Document list: Team:

[Open](#) [Update](#) [Issues](#)

Logged user: [Petar Petric](#) Role: [Junior developer](#) Access level: [2](#) [Logout](#)

Slika 11: Forma developer

Nakon što je korisnik sa rolom Projekt menadžer kreirao projekt i dodijelio mu razvojni tim, te pripadajuće dokumente, svaki korisnik sa rolom developera može pristupiti istima za pregled i/ili editiranje, te imati uvid u preostale članove tima.

Repository: [TestRep](#) Project [Project1](#) [Return](#)

Active project

Document list: Team:

[Open](#) [Update](#) [Issues](#)

Support request

Issue description: Send to: [\[dropdown\]](#)

[Send](#)

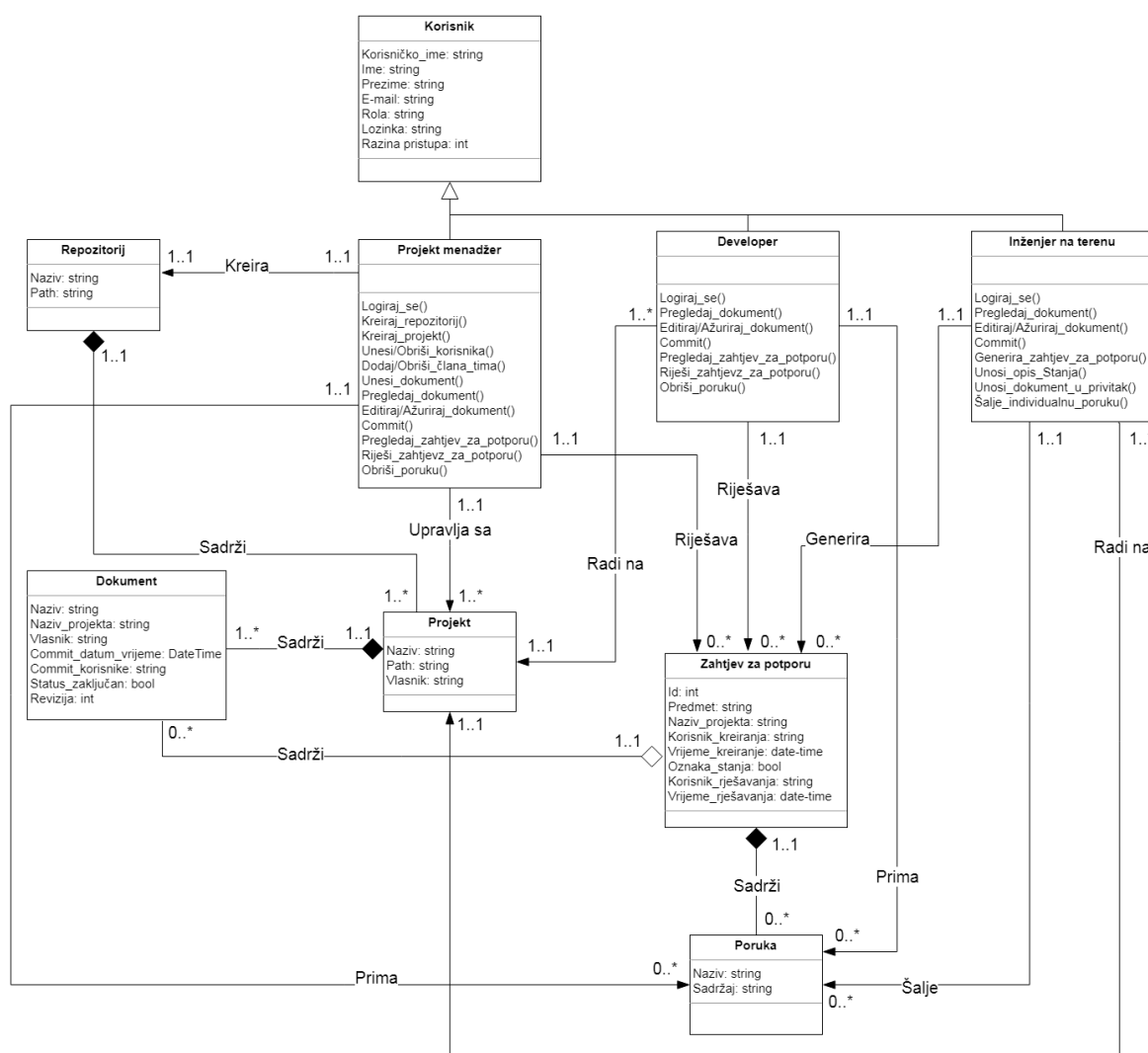
Logged user: [Grgur Grgic](#) Role: [Commissioning](#) Access level: [3](#) [Logout](#)

Slika 12: Forma inženjer na terenu

Početna forma za inženjera na terenu (*eng. commissioning engineer*) odgovara u sadržaju i opcijama formi za developera, sa ključnom razlikom u zahtjevu za potporom (*eng. support request*). Ta dodatna funkcionalnost na razini repozitorija omogućuje slanje i evidentiranje eventualnih odstupanja u projektu otkrivenih u fazi puštanja u rad.

4.4 Klasni dijagram

Klasni dijagram pomoći će potpunijem razumijevanju funkcionalnosti aplikacije prikazom objekata iz domene aplikacije i veza među njima.



Slika 13: Klasni dijagram

Iz gore prikazanog dijagrama, na pregledan se način daju uočiti slijedeće klase i veze među njima:

Klasa korisnik se sa svojim atributima(Ime, prezime, id) u aplikaciji može pojaviti kao Projekt menadžer(PM), Programer (*developer*), te Inženjer na terenu(*commissioning engineer*). svaki od njih može, ali i ne mora, pristupiti određenom projektu radi editiranja ili čitanja. Nadalje, svakom od njih dodijeljene su operacije u okviru ovlaštenja za rad na projektu. Sukladno tome, Inženjer na terenu jedini ima ovlasti kreirati obavijest (zahtjev za potporu) koju neki od developera može obraditi dok radi na

određenom Projektu. Veza između klase Projekt i Dokument dopušta da postoji projekt bez ijednog dokumenta, ali to je neophodno kao inicijalno stanje pri kreiranju novog projekta.

5. Implementacija

U fazi implementacije, teorijski opis sustava dobiven modeliranjem prenosi se u konkretno programsko rješenje izvedeno u odgovarajućoj tehnologiji za zadani problem.

5.1 Softversko okruženje

U ovom slučaju, projektni zadatak je podrazumijevao Windows softversku platformu, programski jezik C#, te preporučenu SQLite bazu podataka.

Tablica 1: Softversko okruženje za razvoj i distribuciju

Naziv	verzija
Windows OS	10x
.Net framework	4.6.01038
Visual Studio Community 2017	15.8.7
- NuGet Package manager	4.6.0
- GitHub.VisualStudio	5.2.5.2500
- SQL Server Data Tools	15.1.61808.07020
- System.Data.SQLite.Core	1.0.109.2
SQLite	3.25.2

Uz gore prikazano softversko okruženje i razvojne alate, programsko rješenje je razvijeno sa klasama prikazanim u tablicama koje slijede.

5.2 Klase

Općenito klase se dijele u grupe prema svojoj funkciji u programu. Sukladno tome, definirane grupe su:

- Osnovne klase – koje u principu služe za rad sa bazom podataka, te se uz pomoću njih dohvaća/mijenja/briše podatke putem odgovarajućih elemenata na formi koji su kodom povezani sa odgovarajućim metodama u osnovnim klasama.
- Pomoćne klase – služe za postizanje funkcionalnosti aplikacije, te definiranje poslovne logike.
- Klase forme – posebna grupa klasa generirana od strane VS-a za svaku formu. Ta skupina klasa služi izravno za grafičku prezentaciju prema korisniku, te u kodu instancira ostale klase i poziva metoda sukladno potrebama prezentacije podataka i postizanja tražene funkcionalnosti preko sučelja aplikacije.

Tablica 2: Osnovne klase

Baza.cs	<code>public string GetSingleLastValue(string sqlString, string col)</code> <code>public string GetRepozName()</code> <code>public string GetRepozPath()</code> <code>public void GetProjectsInCombo(ComboBox ComboName)</code> <code>public void LoadCombo(ComboBox ComboName, string col, string sqlString)</code> <code>public void LoadTextList(ListBox ListBoxName, string col, string sqlString)</code>
---------	--

	<pre> public void LoadVeiwListProjectDocuments(ListView listViewName) public void SendQueryToDb(string sqlString) public void LoadComboNameSurname(ComboBox ComboName, string sqlString) public void LoadVeiwListSupportRequests(ListView listViewName) public void LoadComboMsgTopicTimesamp(ComboBox ComboName, string sqlString) public void LoadVeiwListUsers(ListView listViewName, string sqlString) </pre>
--	---

Sukladno navedenom, gore prikazan klasa Baza.sc služi isključivo za komunikaciju da SQLite bazom podataka, te prosljeđivanje vrijednosti atributa u polja, tablice, padajuće izbornike, liste i sl. Metode definirane unutar klase su globalno dohvatljive, te gdje je to moguće, koriste se višenamjenski (ako ne vraćaju točno određenu strukturu, vezane su na točno određeni objekt, ili točno specifičan SQL upit). To je u principu slučaj sa metodama koje vraćaju vrijednosti, dok se metoda poput SendQueryToDb(string sqlString) koristi za sve upite tipa INSERT, UPDATE, DELETE.

Tablica 3: Pomoćne klase

SysLogic.cs	<pre> public void ClearFolder(string FolderName) public void DeleteFile(string fileName, string folderPath) public string GetUsersWorkingFolderPath(string username) public bool FindInColumn(string valToCheck, string column, string table) public string GetNameSurnameFromUsername(string username) </pre>
BizLogic.cs	<pre> public string GetAccLevelString(string role) </pre>

Pomoćne klase u gornjoj tablici služe za postizanje željene funkcionalnosti aplikacije koja nije u svezi sa bazom podataka. U tu kategoriju, sukladno postavljenom problemu za koji se razvija programsko rješenje, spada rad sa datotečnim sustavom, te definicija/manipulacija podacima vezanim uz poslovnu logiku (poput razina prava pristupa korisnika i sl.).

Tablica 4: Klase formi

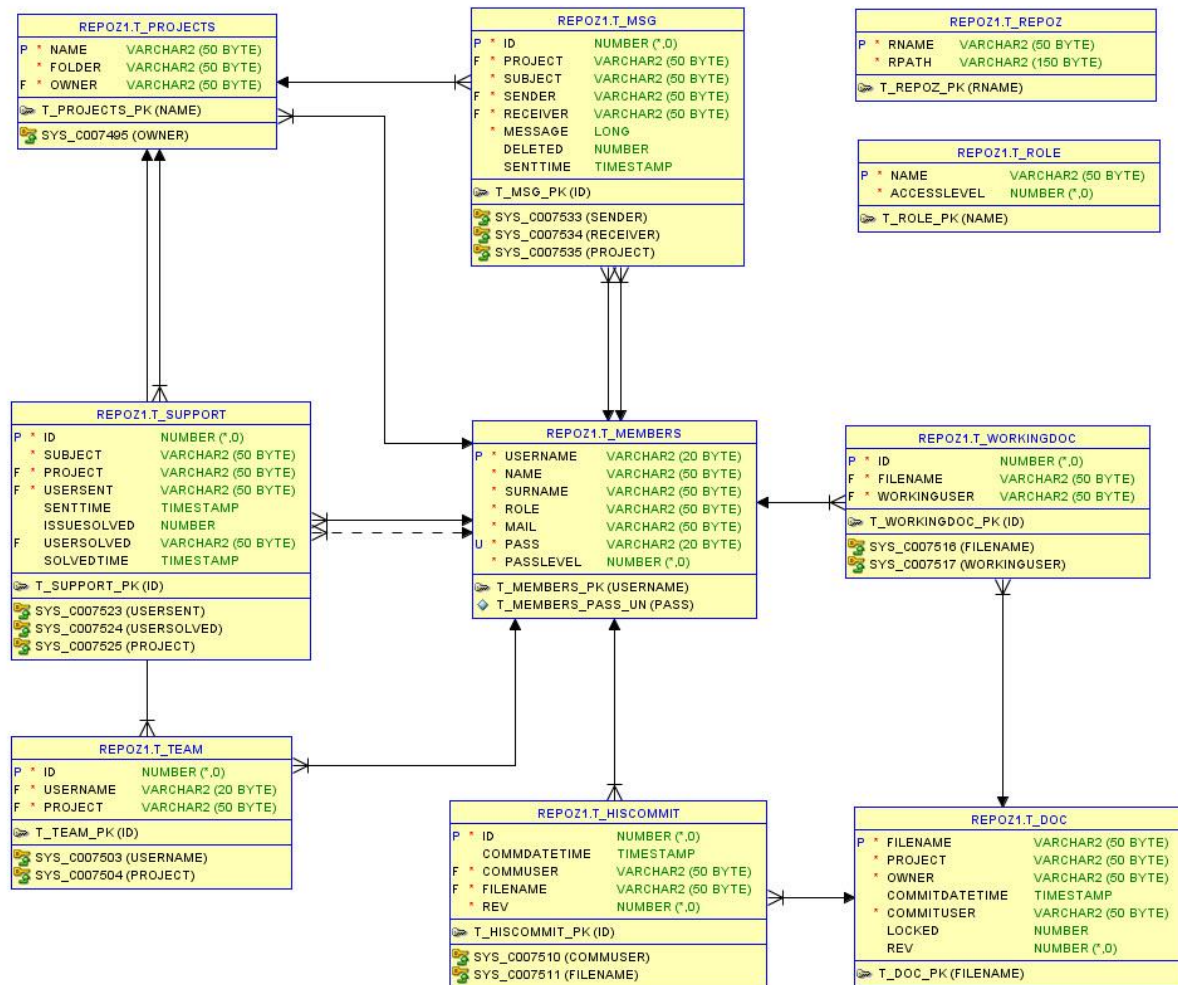
public partial class FormLogin	<pre> private void btnLoginEnter_Click(object sender, EventArgs e) private void btnLoginEnter_Click(object sender, EventArgs e) private void comboUsers_SelectedIndexChanged(object sender, EventArgs e) </pre>
public partial class FormStart	<pre> private void btnSettings_Click(object sender, EventArgs e) private void btnLogin_Click(object sender, EventArgs e) private void btnOpenProject_Click_1(object sender, EventArgs e) private void btnProjSet_Click(object sender, EventArgs e) private void btnEditDoc_Click(object sender, EventArgs e) private void btnOpenDoc_Click(object sender, EventArgs e) private void btnCommit_Click(object sender, EventArgs e) private void btnSupport_Click(object sender, EventArgs e) private void btnOpenSupport_Click(object sender, EventArgs e) private void btnSolved_Click(object sender, EventArgs e) private void comboMsgSubject_SelectedIndexChanged(object sender, EventArgs e) private void btnDeleteMsg_Click(object sender, EventArgs e) </pre>

	<code>private void btnTeamMembers_Click(object sender, EventArgs e)</code>
public partial class FormProjectSettings	<code>private void btnAssignUser_Click(object sender, EventArgs e)</code> <code>private void btnRemoveUser_Click(object sender, EventArgs e)</code> <code>private void btnAddDoc_Click(object sender, EventArgs e)</code> <code>private void FormProjectSettings_DragEnter(object sender, DragEventArgs e)</code> <code>private void FormProjectSettings_DragDrop(object sender, DragEventArgs e)</code> <code>private string getFileName(string path)</code> <code>private string getFolderPath(string path)</code> <code>private void btnRemoveDoc_Click(object sender, EventArgs e)</code>
public partial class FormSettings	<code>private void txtRepozName_Enter(object sender, EventArgs e)</code> <code>private void txtRepozName_Leave(object sender, EventArgs e)</code> <code>private void btnReturn_Click(object sender, EventArgs e)</code> <code>private void btnSelectFolder_Click(object sender, EventArgs e)</code> <code>private void txtNewProjectName_Enter(object sender, EventArgs e)</code> <code>private void txtNewProjectName_Leave(object sender, EventArgs e)</code> <code>private void btnCreateProject_Click_1(object sender, EventArgs e)</code> <code>private void btnAdduser_Click(object sender, EventArgs e)</code> <code>private void btnRemoveUser_Click(object sender, EventArgs e)</code>
public partial class FormSupport	<code>private void btnReturn_Click(object sender, EventArgs e)</code> <code>private void btnSendReq_Click(object sender, EventArgs e)</code> <code>private void btnAttach_Click(object sender, EventArgs e)</code> <code>private void btnSendMsg_Click(object sender, EventArgs e)</code> <code>private void grpRepoz_Enter(object sender, EventArgs e)</code>
public partial class FormTeamMembers	<code>private void FormTeamMembers_Load(object sender, EventArgs e)</code>

Klase formi generirane su iz IDE sučelja za svaku novu formu, te izravno omogućuju funkcionalnost korisničkog sučelja(formi). Svaka metoda gore prikazanih klasa povezana je sa nekim od grafičkih elementa na formi, odnosno na odgovarajući događaj (*event*) koji je raspoloživ za dani element. Prema zadanim postavkama, svaka metoda je generirana kao „private“, čime ista nije dostupna izvan klase, odnosno forme. Sukladno tome, sa određene forme nije omogućeno referenciranje na elemente druge forme.

1.1 Baza podataka

U projektu je korištena SQLite baza podataka. Ključno u izboru baze podataka jeste jednostavnost korištenja, te nepostojanje instalacijske procedure. Eventualne graničenja koja SQLite, ima u odnosu na neka robusnija rješenja poput Oracle, PostgreSQL, MS SQL i slično, nije od značaja za predmetni projekt, imajući u vidu mali broj konekcija, malu količinu podataka, te da baza služi isključivo za pohranu podataka, bez potrebe za izvršavanjem ikakve dodatne programske logika.



Slika 14: ER dijagram

2. Korisničke upute

Upute dane u nastavku teksta imaju za cilj pružiti cjelovit pregled sučelja, sa svim formama i objašnjenjem funkcionalnosti svake od njih, te pripadnih elemenata. Sadržaj je prvenstveno usmjeren na korisnika aplikacije, na način da ga pripremi za redovan rad korištenjem raspoloživih funkcionalnosti dostupnih na formama. Upute su orijentirane ka formama, odnosno, umjesto da upute prate varijante mogućih scenarija za određenog korisnika, svaka će forma redom biti objašnjena sa svim svojim funkcionalnostima.

Tablica 5: Korisničke role

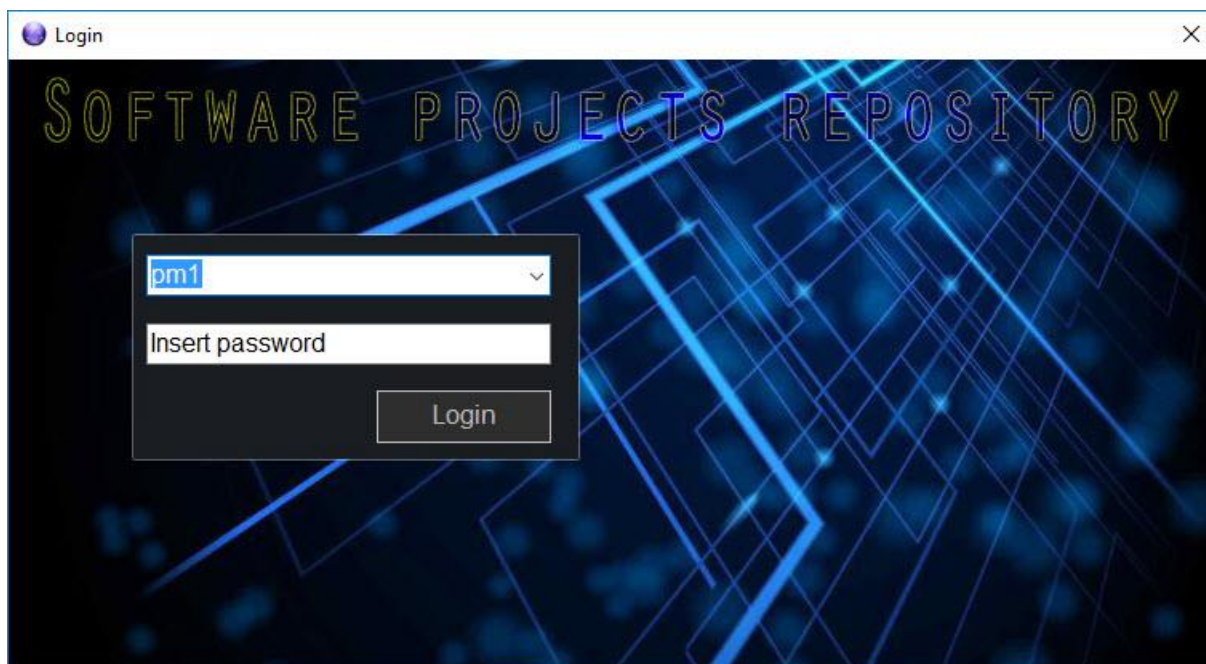
Korisnik \ Funkcionalnost	Projekt menadžer (PM)	Programer (<i>developer</i>)	Inženjer na terenu (<i>commissioning engineer</i>)
Kreiranje repozitorija	X		
Kreiranje projekta	X		
Unos novog korisnika	X		
Dodjela korisnika u projektni tim	X		
Unos dokumenta u projekt	X		
Pregled dokumenata	X	X	X
Editiranje dokumenata		X	X
Generiranje zahtjeva sa potporu			X
Slanje obavijesti korisnicima (uz zahtjev za potporu)			X
Rješavanje zahtjeva za potporu	X	X	X

Iz gornje tablice vidljivo je kako Projekt menadžer ima funkciju/ovlasti administratora repozitorija, te time može biti uključen u više projekata istovremeno, ali nema neograničena prava nas kreiranim projektima. Premda je unio određeni dokument u projekt, nema daljnja prava za editiranje unesenog. S druge strane, Programer ima ovlasti uglavnom nad dokumentima projekta u kojemu se nalazi kao član razvojnog tima. U odnosu na Programera, Inženjer na terenu razlikuje se jedino u tome što može generirati zahtjev za potporu. U svemu ostalome, radi se također o članu razvojnog tima nekog, ili više projekata.

Tablica 6: Razine pristupa korisničkih rola

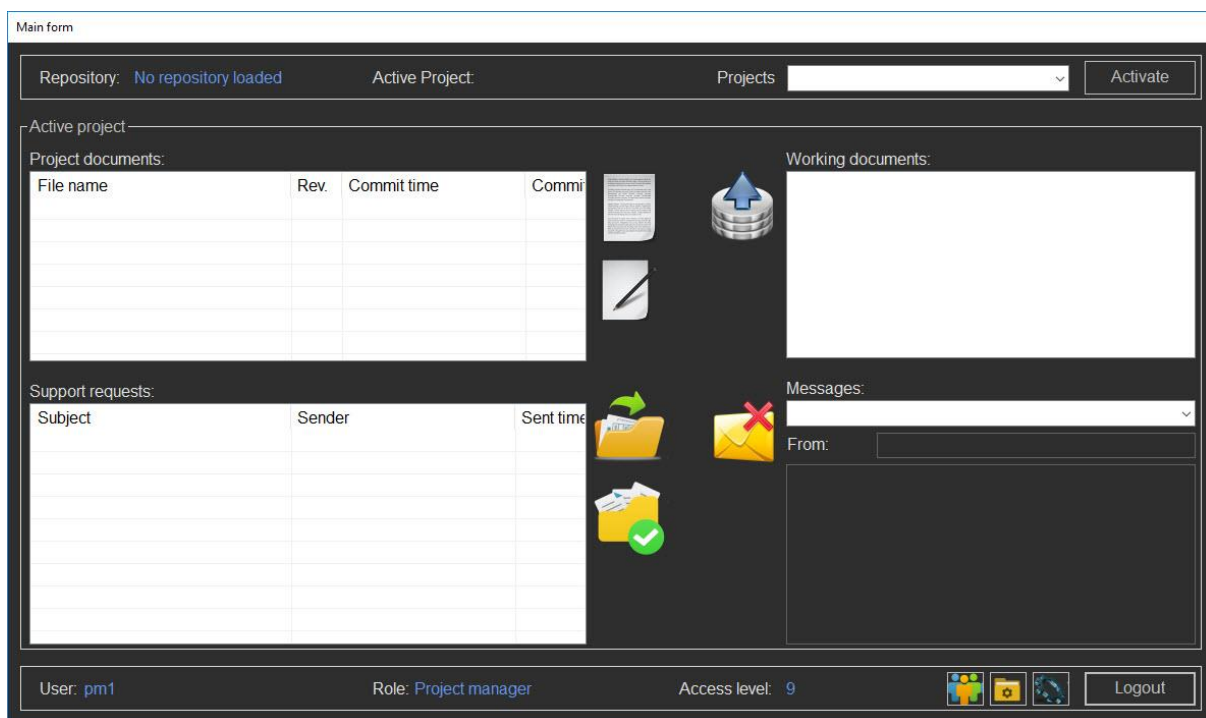
Razina pristupa	Role
2	Programer
3	Inženjer na terenu
9	Projekt menadžer

Dodatno na tablicu 1., pristup pojedinim funkcionalnostima aplikacije definiran je bročanom vrijednošću. Na taj način dopuštena je mogućnost da više rola bude ovlašteno za istu funkciju, te da autorizacija može uključiti više rola istovremeno (sa razinom pristupa višom od zadane).



Slika 15: Login forma

Prilikom pokretanja aplikacije, za odabir korisničkog imena nudi se izbor trenutno unesenih korisnika. Ukoliko se radi o prvom korištenju aplikacije, jedini korisnik na raspolaganju biti će Projekt menadžer(korisničko ime: PM) koji tek mora aktivirati repozitorij i obaviti unos ostalih korisnika. Ovisno odabranom korisniku, te nakon uspješne prijave, razina autorizacije ovisiti će o korisnikovoj roli, odnosno razini pristupa.



Slika 16: Glavna forma (*main form*) – inicijalno

Gore prikazana Glavna forma biti će u ovom obliku otvorena samo prilikom inicijalne(prve) prijave u sustav kada još ne postoji repozitorij. U ovom trenutku, sve su

opcije onemogućene osim otvaranja forme za konfiguraciju repozitorija (forma *Settings*).

Main form

Repository: Repo1 Active Project: Pro1 Projects: Pro1 Activate

Active project

Project documents:

File name	Rev.	Commit time	Commit
Dokument_1.docx	0	2018-11-12 15:58:29	pm1
SWOT_template.jpg	0	2018-11-12 15:58:37	pm1
PDF_Dokument_1.pdf	0	2018-11-12 16:01:20	pm1

Working documents:

Dokument_1.docx

Support requests:

Subject	Sender	Sent time

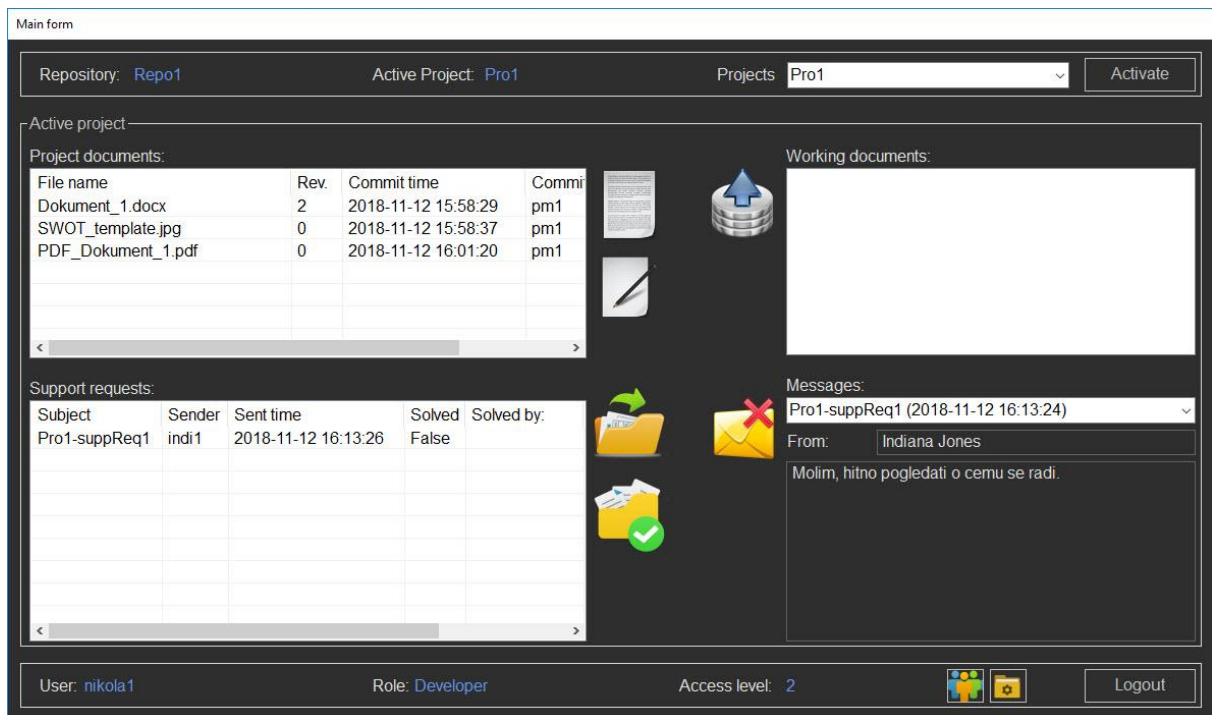
Messages:

From:

User: pm1 Role: Project manager Access level: 9 Logout

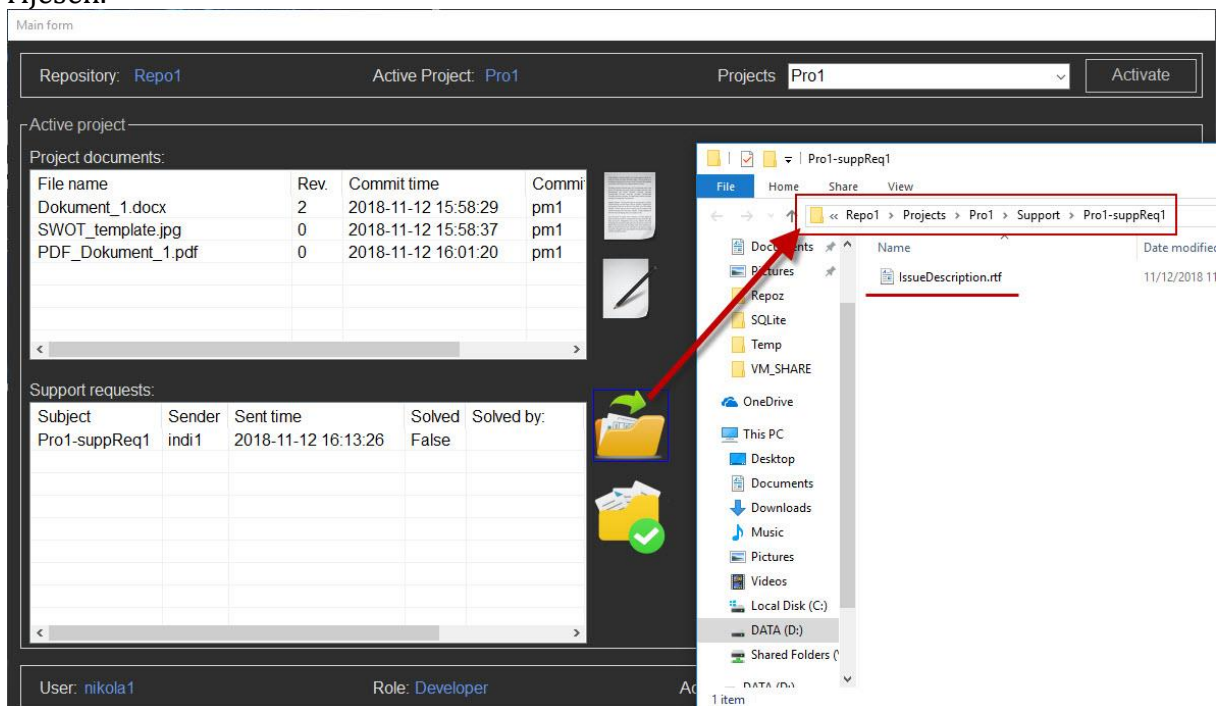
Slika 17: Glavna forma (*main form*) – PM

Ukoliko je repozitorij već prethodno kreiran, te zatim kreiran i jedan ili više projekata sa pripadnim dokumentima, tada će odabirom projekta iz ponuđenog padajućeg izbornika, te aktiviranjem istog klikom na dugme Activate, prikaz na formi odgovarati gornjoj slici. Ovako prikazana forma, predstavlja osnovni radni zaslon koji je zajednički za većinu korisnika, uz izuzetak role Inženjer na terenu, čiji korisnik ima mogućnost navigacije na formu za generiranje zahtjeva za potporu. Nakon aktivacije projekta, lista Project documents(gore lijevo) prikazuje sve dokumente koje je PM dodijelio projektu. Dokumenti se sa te liste mogu dohvatiti za čitanje i/ili ažuriranje/editiranje. U slučaju da je dokument otvoren za editiranje, isti se pojavljuje na listi Working documents(gore desno) i nakon što su izmjene nad njim sačuvane, korisnik ga vraća natrag na repozitorij klikom na dugme Commit(uz listu). Tek će se tada dokument koji se nalazi ne repozitoriju pregaziti i evidentirati pod novom revizijom(automatski se inkrementira) i korisničkim imenom pod kojim je *commit* izvršen. Dokument koji je jednom dohvaćen za editiranje i prebačen u na listu Working documents, biva zaključan za otvaranje do izvršenja *commit* naredbe kako bi se izbjegao problem preklapanja izmjena nad istim dokumentom.



Slika 18: Glavna forma (*main form*) – Developer

Gornji primjer sadržaja glavne forme odnosi se na slučaj kada postoji zahtjev za potporom (za aktivni projekt), generiran od strane Inženjera na terenu (*commissioning engineer*). U tom slučaju će se zahtjev za potporu prikazati na listi Support request (lijevo dolje), dok će eventualna dodatna poruka biti prikazana desno u produžetku. Svaki programer (član razvojnog tima aktivnog projekta) ima pravo pristupa u direktorij koji sadrži sve relevantne informacije određenog zahtjeva za potporu (slika dolje), te u slučaju da riješi problem, klikom na odgovarajuće dugme (uz listu), označi isti kao riješen.




Slika 19: Glavna forma (*main form*) – zahtjev za potporu


Jednom kada je određen slučaj zatvoren, pripadni zahtjev za potporu biva označen kao riješen zajedno sa korisničkim imenom aktivnog korisnika, i vremenom zaključenja.

Repository Settings

New



Repository:



User:

Name:

Surname:


Role: ▼

E-mail:

Username:

Password:

All users




Name	Surname	Username	Role	Level	E-mail
Bill	Gates	pm1	Project manager	9	b.gates

Slika 20: Forma postavke repozitorija (Repository settings) – inicijalno


Kako je prethodno objašnjeno, nakon inicijalnog pokretanja aplikacija, sa glavne forme moguće je pristupiti jedino formi Repository settings kako bi se kreirao repozitorij, i time omogućilo kreiranje projekata. Prema gornjoj slici, repozitorij se kreira unosom imena, te odabirom lokacije gdje će se isti nalaziti. Kada je repozitorij kreiran, u nastavku se može pristupiti kreiranju projekata, za što je dovoljno unijeti naziv istog. Klikom na dugme Create project, procedura će samostalno kreirati istoimeni vršni direktorij projekta, kao i pripadnu strukturu pod-direktorija. Kao što je već prethodno rečeno, inicijalno se u bazi nalazi samo jedan korisnik-PM (sa ovlastima za kreiranje repozitorija) koji je vidljiv na listi svih korisnika.

Repository Settings




Repository:

Select folder



Project:


Create project



User:

Name:
Surname:
Role:
E-mail:
Username:
Password:

Add user



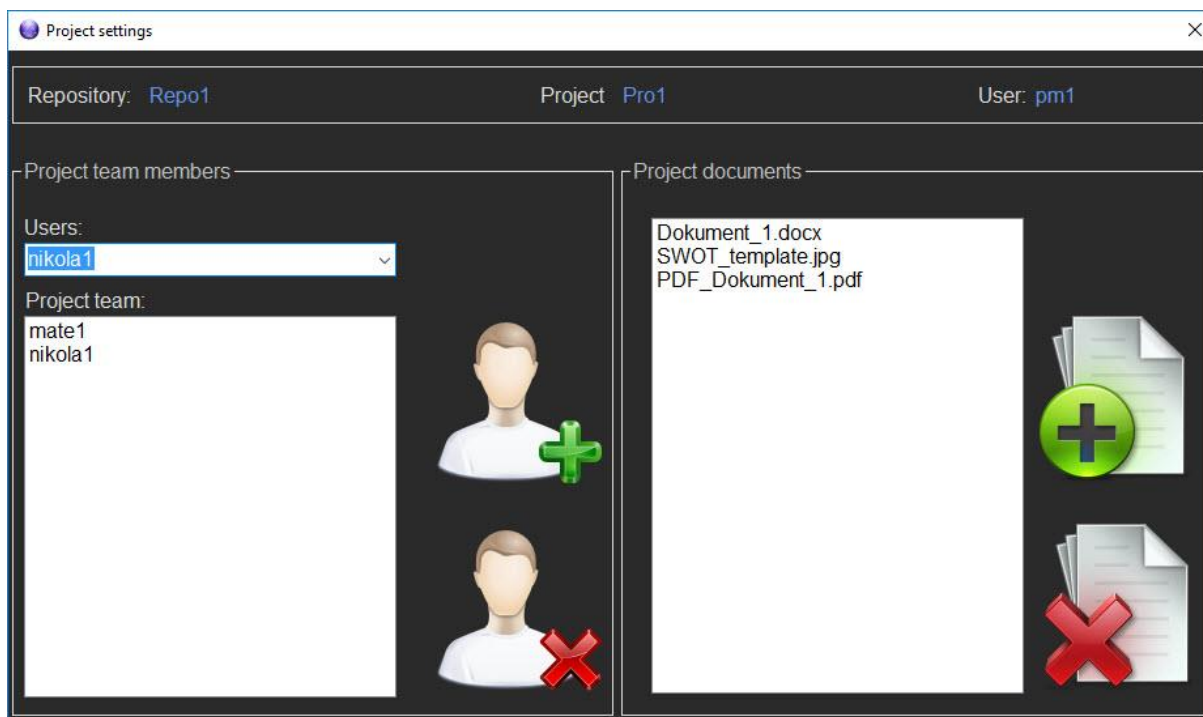
Name	Surname	Username	Role	Level	E-mail
Bill	Gates	pm1	Project manager	9	b.gates
Nikola	Tesla	nikola1	Developer	2	nikola.t
Mate	Rimac	mate1	Developer	2	mate.ri
Indiana	Jones	indi1	Commissioning eng.	3	indiana

Return

Slika 21: Forma postavke repozitorija (Repository settings) – novi korisnik

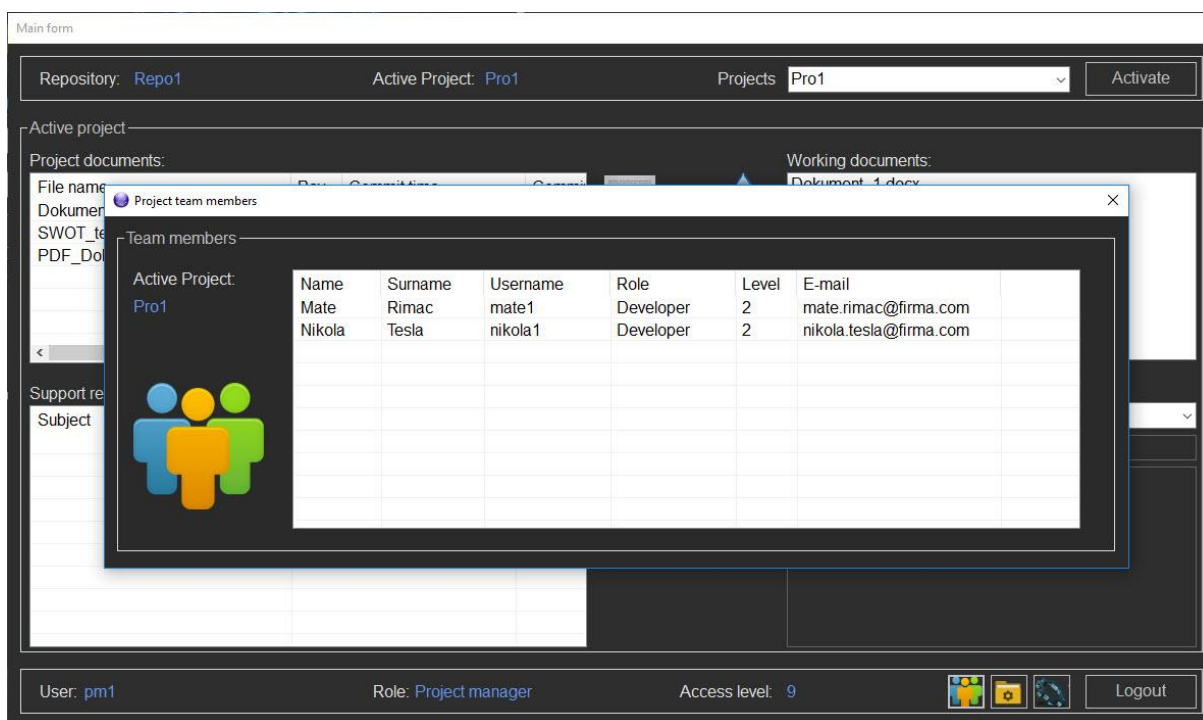
Pri unosu novog korisnika, pripadnu rolu moguće je birati od ponuđenih u skladu sa tablicom 1. Unos korisnika koji se izvrši na ovoj formi odnosi se na unos u bazu, te se ne smije poistovjetiti sa dodjeljivanjem korisnika na određeni projekt. Za to postoji forma Project settings. Jednom kada je korisnik unesen, njegovi su podatci vidljivi u tablici All users kako je prikazano na donjoj slici. Također, valja primijetiti da je dugme Select folder(gore lijevo) onemogućeno, što se događa nakon što je repozitorij jednom uspješno kreiran.

28



Slika 22: Forma postavke projekta (Project settings)

Jednom kada je projekt kreiran, potrebno mu je dodati nekakav sadržaj u obliku dokumenata, te dodijeliti neke od postojećih korisnika u razvojni tim. Navedeno je moguće učiniti putem gore prikazane forme kojoj se pristupa sa glavne forme preko alatne trake u dnu. Da bi to bilo moguće, projekt mora prethodno biti aktiviran, što se također čini sa glavne forme kako je prethodno već objašnjeno. Na prikazanoj formi se jednostavno i intuitivno dodaju/brišu dokumenti, te dodjeljuju/uklanjaju korisnici sa aktivnog projekta. Za pristup formi potrebna je razina pristupa 9 (rola PM).



Slika 23: Projektni tim (Project team members)

Formi Project team members pristupa se sa alatne trake na glavnoj formi. Da bi to bilo moguće, prethodno mora postojati aktivan projekt. Forma sadrži listu svih članova tima sa pripadnim podacima. Otvaranje forme je omogućeno za sve korisnike, a glavna joj je funkcija prikaz identiteta članova projektnog tima, budući se u prikazu podataka obično koristi korisničko ime, što ne mora uvijek ukazivati na identitet korisnika.

Support request

New Support request

Active project: **Pro1**

Subject: Pro1-suppReq1 Attach file

Issue description: ne radi commit dugme

Message to team member(optional): Nikola Tesla Send message Send request

Molim, hitno pogledati o cemu se radi.

Return

Slika 24: Forma zahtjeva za potporu (Support request)

Gornjoj formi moguće je pristupiti sa glavne forme uz uvjet da postoji aktivni projekt, i da je prijavljeni korisnik sa razinom pristupa 3 (Inženjer na terenu - *commissioning engineer*). Forma služi generiranju zahtjeva za potporu, tj. prijavi uočenih grešaka/poteškoća/nepравilnosti i sl. u fazi puštanja sustava u rad. Svaki prijavak sastoji se od polja Subject pod kojim će se prijavak voditi, kratak opis, te svih potrebnih dokumenata u privitku. Unesen opis biti će generiran u obliku tekstualnog dokumenta, te također pridodan u kreirani direktorij (naziv direktorija = tekst iz polja Subject). koji postaje dio strukture direktorija projekta. Opcionalno, moguće je poslati i dodatnu poruku određenom članu/članovima tima. Sve će se navedeno pojaviti na glavnoj formi aktivnog projekta i biti dostupno na uvid i daljnje postupanje članovima projektnog tima.

3. Popis slika

<i>Slika 1: Osnovni koncept GitHub repozitorija</i>	4
<i>Slika 2: Korisničke uloge</i>	5
<i>Slika 3: SWAT analiza</i>	7
<i>Slika 4: Use-case dijagram</i>	10
<i>Slika 5: PM sequence diagram</i>	12
<i>Slika 6: Developer sequence diagram</i>	13
<i>Slika 7: Inženjer na terenu sequence diagram</i>	14
<i>Slika 8: Login forma</i>	15
<i>Slika 9: Početna forma – Projekt menadžer</i>	15
<i>Slika 10: Forma postavke</i>	16
<i>Slika 11: Forma developer</i>	17
<i>Slika 12: Forma inženjer na terenu</i>	17
<i>Slika 13: Klasni dijagram</i>	18
<i>Slika 14: ER dijagram</i>	22
<i>Slika 15: Login forma</i>	24
<i>Slika 16: Glavna forma (main form) – inicijalno</i>	24
<i>Slika 17: Glavna forma (main form) – PM</i>	25
<i>Slika 18: Glavna forma (main form) – Developer</i>	26
<i>Slika 19: Glavna forma (main form) – zahtjev za potporu</i>	26
<i>Slika 20: Forma postavke repozitorija (Repository settings) – inicijalno</i>	27
<i>Slika 21: Forma postavke repozitorija (Repository settings) – novi korisnik</i>	28
<i>Slika 22: Forma postavke projekta (Project settings)</i>	29
<i>Slika 23: Projektni tim (Project team members)</i>	29
<i>Slika 24: Forma zahtjeva za potporu (Support request)</i>	30

4. Popis tablica

<i>Tablica 1: Softversko okruženje za razvoj i distribuciju</i>	19
<i>Tablica 2: Osnovne klase</i>	19
<i>Tablica 3: Pomoćne klase</i>	20
<i>Tablica 4: Klase formi</i>	20
<i>Tablica 5: Korisničke role</i>	23
<i>Tablica 6: Razine pristupa korisničkih rola</i>	23

5. Literatura

Internet

1. <https://www.intertech.com/Blog/introduction-to-git-concepts>
2. https://www.atlassian.com/blog/archives/fisheye_in_practice_setting_up_your_own_git_repositories