SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

„OObjekt projekt“

Robert Tomičić  
Dan Novak  
Otto Singer

Zagreb, siječanj 2017.

Sadržaj

[1. Uvod 3](#_Toc473785428)

[2. Requirements 5](#_Toc473785429)

[1) Business requirements 5](#_Toc473785430)

[2) User requirements 5](#_Toc473785431)

[3) System requirements 5](#_Toc473785432)

[3. Opis (objektnog) modela 8](#_Toc473785433)

[Desktop aplikacija 8](#_Toc473785434)

[Web aplikacija 9](#_Toc473785435)

[4. Opis implementacije perzistencije 11](#_Toc473785436)

[5. Opis izgrađene desktop aplikacije 13](#_Toc473785437)

[6. Opis web aplikacije 16](#_Toc473785438)

[7. Baza Podataka 18](#_Toc473785439)

# Uvod

Aplikacija „OObjekt projekt“ služi za formiranje i praćenje grupa za projekt na predmetu Objektno oblikovanje na FER-u. Zadaća aplikacije jest da olakša profesoru interakciju s timovima pri odabiru teme za projekt. Studenti su oni koji rade većinu posla dok profesor nadgleda rad te samo sudjeluje u završnom djelu formiranja timova za projekt.

Postoje dva skupa funkcionalnosti aplikacije, po jedna za svaku vrstu korisnika. Vrsta korisnika se određuje odmah pri prijavi u sustav pomoću imena i lozinke. Prva vrsta korisnika je osoba koja polaže predmet, to jest običan korisnik, dok je druga vrsta korisnika profesor predmeta, to jest administrator.

Svaki korisnik može kreirati svoj projekt te pozivati druge članove u svoj tim koji radi na tom projektu. Ukoliko korisnik napravi svoj projekt, on je vođa tog projekta i tog tima, čak i ako je jedini član tima. Vođa projekta također može pozivati druge članove u svoj tim. Ukoliko korisnik ne želi napraviti novi projekt, nego se pridružiti tuđem timu, može pregledati ostale timove i poslati im zahtjev za pridruživanje. Pri slanju zahtjeva može opcionalno poslati poruku vezanu za taj zahtjev za ulazak u taj projektni tim. U timu može biti maksimalno 5 članova. Ukoliko je vođa tima odlučio da je tim finaliziran, može zaključati tim kako bi profesor vidio njegovu odluku.

Profesor ima sljedeće opcije: pregled projekata i pripadajućeg tima, dodavanje korisnika u sustav te odobravanje zaključanog tima. S obzirom na to da profesor neće sudjelovati u izradi projekta, njemu nisu potrebne opcije kreiranja timova ili pridruživanje drugim timovima, nego isključivo odobravanje formiranog tima i dodavanje korisnika u sustav kako bi ga korisnik mogao koristit.

U sklopu ovog projekta izrađena je desktop i web aplikacija. Desktop aplikacija sadrži sve opisane mogućnosti te prijavu administratora i običnog korisnika, dok će web aplikaciju koristiti samo studenti za prijavu u sustav.

Desktop verzija aplikacije ima sve navedene funkcionalnosti dok online verzija sustava ima funkcionalnosti koje može obaviti korisnik koji nije administrator. Razlog tomu je da administrator pristupa isključivo preko instalirane aplikacije na svojem računalu gdje je veća sigurnost.

# Requirements

## Business requirements

Sustav treba omogućiti preciznije i lakše upravljanje grupama za projekt. Cilj je da grupe budu jednoznačno formirane preko jednog izvora podataka umjesto pregledavanja e-maila za raznim izvorima "formiranja" timova, te je svaki član odgovoran za svoje mjesto u timu. Dodatno, cilj je da timovi mogu međusobno gledati teme od drugim timova radi usporedbe i to također, bez potrebe da se profesora uključi u to. Timovi se formiraju na način da ljudi koji su kreirali tim mogu slati i primati pozive ulaska u tim, te također izbaciti ljude iz svoga tima.

## User requirements

Korisnik koji je student(polagatelj predmeta) treba moći:

1. pomoću korisničkog imena i lozinke ući u sustav
2. kreirati projekt, pozvati druge u svoj projekt i zaključati svoj projekt
3. poslati zahtjev i pridružiti se tuđem projektu
4. vidjeti sve trenutno definirane projekte
5. Administrator(profesor) treba moći:
6. pomoću korisničkog imena i lozinke ući u sustav
7. vidjeti listu svih projekata i odobriti ih
8. definirati novog korisnika kako bi se navedeni mogao ulogirati

## System requirements

Desktop aplikacija je ostvarena programom „Windows Forms“, dok je web aplikacija napravljena u programu ASP.NET MVC 4. Kod pokretanja desktop aplikacije javlja se Login ekran u kojem se korisnika traži korisničko ime i lozinka, a daje mu se mogućnost da mu aplikacija zapamti zadnje korisničko ime s kojim se ulogirao. Nakon prijavljivanja, ovisno o tome je li račun od administratora, ili običnog korisnika, pojavljuje se novi ekran. Taj ekran ostaje upaljen do završetka rada, to jest dok se korisnik ne odjavi tipkom "Logout", dok se dodatni ekrani mogu upaliti i zatvarati po izboru. Ako se ulogirao običan korisnik, izbor dodatnih ekrana mu je sljedeći:

1. pregled projekta u kojem se nalazi, te ako je vođa može pozivati i izbacivati ljude iz tima, prihvaćati dolazeće zahtjeve i zaključati tim
2. pregled drugih projekata gdje se može prijaviti za ulazak u tim ako već nije u timu
3. kreiranje svojeg vlastitog tima, gdje postaje vođa
4. pregled i opcionalno prihvaćanje zahtjeva za ulazak u drugi tim

Administrator ima samo 3 funkcionalnosti koje su raspoređene u 2 prozora. Prva je definiranje novog korisnika upisom korisničkog imena, šifre, imena i prezimena i JMBAG-a u prazna polja te potvrdom upisa gumbom na ekranu. Druga funkcionalnost je pregled svih projekata te ljudi koji su članovi pripadnog tima, dok se odabirom u listi određuje koji tim je trenutno prikazan. Treća mogućnost je da može odobriti ili otključati tim koji je zaključan korištenjem gumbi.

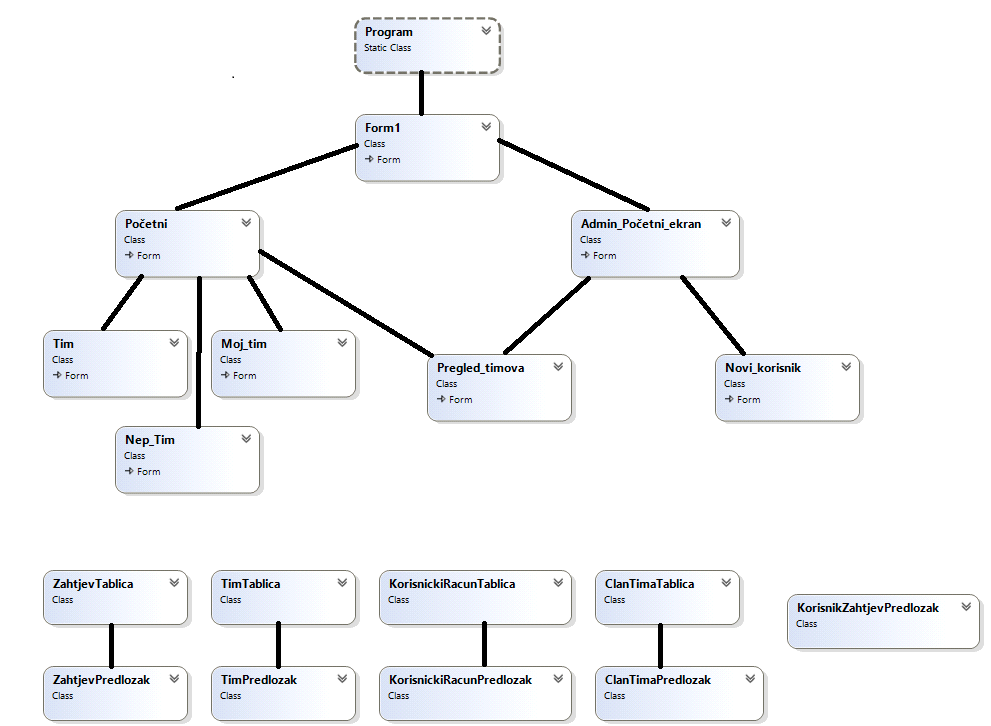
Kod pokretanja web aplikacije pojavljuje se navigacijska traka (kojoj se još ne može pristupati dok nije obavljena prijava u sustav) i forma za prijavu u web aplikaciju. Ukoliko se korisnik prvi put prijavljuje u sustav i još nema svoj tim, sustav će ga preusmjeriti na pregledavanje već postojećih timova, ali će uz pomoć alatne trake moći odabrati izradu novog tima. Mogućnosti ponuđene na navigacijskoj traci nešto su drugačije od onih u desktop verziji aplikacije, a korisnicima su dostupne ovisno nalaze li se već u timu ili ne. U web aplikaciji mogućnosti korisnka su:

1. pregled projekta u kojem se nalazi, te ako je vođa može izbacivati ljude iz tima, prihvaćati dolazeće zahtjeve i zaključati tim
2. pregled popisa drugih projekata gdje se može prijaviti za ulazak u tim ako već nije u timu
3. pregled informacija drugih projekata na posebnoj stranici za svaki projekt
4. kreiranje svojeg vlastitog tima, gdje postaje vođa

U razine aplikacijskog sloja, razlikujemo 4 vrste korisnika u sustavu: Administrator, osoba bez tima, osoba koja je u timu a nije vođa tima, te osoba koja je vođa tima. Svaki korisnik u svakom trenutku ima vidljive isključivo gumbe na koji ima pravo reagirati. Primjer: pri pregledu timova, jedino osoba bez tima može poslati zahtjev za pridružiti se timu, te jedino admin može odobriti i zaključati tim. Svi gumbi nalaze se na istoj formi, no ovisno o vrsti korisnika (koja se nalazi u bazi) prikazuju se neki od gumba.

# Opis (objektnog) modela

## Desktop aplikacija



Slika 1. Klasni dijagram

**Na slici 1. prikazan je raspored klasa. Iz klase Program se poziva klasa Form1 koja je ujedno i login ekran. Nakon login ekrana se pojavljuje se ili korisnikov početni ekran(Početni) ili administratorov početni ekran(Admin\_Početni\_ekran) ovisno o podacima koje je korisnik unio. Jedinu zajedničku klasu(formu) koju imaju i korisnik i administrator jest Pregled\_timova gdje se mogu vidjeti svi postojeći projekti te njihovi članovi. Klase ZahtjevPredlozak, TImPredlozak, KorisnickiRacunPredlozak te ClanTimaPredlozak imaju ulogu information holdera. Klase ZahtjevTablica, TImTablica,KorisnickiRacunTablica,ClanTimaTablica služe za interakciju s bazom te spramaju svoje podatke u reprezentativne predloške. Navedene klase XTablica te KorisnikZahtjevPredlozak pružaju servise formama koje traže intrakciju s bazom. Tko zove te servise? Bilo koji pritisak na gumb u bilo kojoj formi traži se ili upit ili izmjena nad bazom. Ako se traži upit, funkcija koja se poziva klikom na određeni gumb pozove jednu od servisa XTablica koje onda dohvate tražene podatke te ih spreme u XPredložak objekt. Taj objekt se zatim koristi za prikaz ili daljnju obradu. Upit se traži kod svakog pozivanja nove forme kako bi se u formi odmah na početku prikazale bitne informacije dovučene iz baze, te se koristi kod svake promjena odabira u listi korisnika ili timova. Ako se traži izmjena ili dodavanje u bazu, jedna o forma također pritiskom na gumb poziva servis od klasa XTablica koje spreme upisane podatke u XPredlozak te zatim predloške spreme u bazu ili izvrše izmjenu nad njom. S obzirom da klasa Xtablica osim pružanja servisa funkciji koju pozove forma također služi i za stvaranje objekata koji predstavljaju bazu, također je i factory. Nadalje, s obzirom da klase Xpredložak služe za držanje podataka iz base u njima, su repozitoriji.**

## **Web aplikacija**

**Web aplikacija izgrađena je na MVC modelu. Sastoji se od 3 controller-a (Project, Request i User), 5 model-a (KorisnickiRacun, Tim, Zahtjev, Project i User) te 6 view-ova (All, My, New, Profile, Login, \_Layout). Modeli služe za prikaz podataka iz baze podataka i prikaz podataka koji se šalje u view. Controlleri koriste modele i odlučuju koji će se view prikazati ovisno o dolaznom HTTP zahtjevu, dok je view samo sadržaj ekrana koji se prikazuje korisniku. Ovaj način izgradnje web aplikacije omogućava razdvajanje programske logike od same prezentacije sadržaja korisniku. Controller Login sastoji se od 3 metode:**

1. **SetSession() – postavlja sjednicu korisnika koja pamti njegov username, ime i prezime te identifikacijski broj**
2. **Login() – omogućava prijavu korisnika u sustav**
3. **Logout() – omogućava odjavu korisnika iz sustava**

**Project controller sastoji se od 9 metoda koje upravljaju projektima:**

1. **My() – definira kako će se prikazati ekran za člana tima, a kako za vođu**
2. **All() - šalje listu svih projekata iz baze podataka kako bi se mogao prikazati popis**
3. **New() – odlučuje treba li se korisniku omogućiti stvaranje novog tima ovisno o tome je li već član nekog tima.**
4. **Profile() – prikazuje posebnu stranicu za profila za svaki od projekata na popisu**
5. **Delete() – omogućuje akciju brisanja tima**
6. **RemoveUser() – omogućuje izbacivanja člana ili dobrovoljno napuštanje tima**
7. **Lock() – zaključavanje tima**
8. **Unlock() - otključavanje tima**
9. **Save() – spremanje novo napravljenog tima u bazu podataka**

**Request controlleri sadrži metode koje se bave zahtjevima pristupanja studenata u tim i koriste model zahtjeva iz baze podataka Zahtjev():**

1. **Send() – slanje zahtjeva pristupanja timu**
2. **Accept() – odobravanje zahtjeva pristupanja timu**
3. **Decline() – odbijanje zahtjeva pristupanja timu**

**Modeli KorisnickiRacun, Tim i Zahtjev reprezentacija su podataka zapisanih u bazi podataka. KorisnickiRacun i Tim sadrže konstruktore koji pri njihovom stvaranju model napune podacima iz baze podataka, ovisno o danom parametru. Model KorisnickiRacun sadržava metode provjere valjanosti korsničkog imena i lozinke, metodu za napuštanje tima i metode za provjeravanje ima li korisnik tim te jeli vođa nekog tima. Model Tim također sadrži metode za spremanje i brisanje iz baze podataka, metodu za prikaz svih zahtjeva poslanih timu, metodu za zaključavanje tima te metodu za dobivanje listu objekata svih članova i metodu za dobivanje objekta vođe tima. Model Zahtjev sadrži metode za slanje, prihvaćanje i odbijanje zahtjeva. Model User koristi se za spremanje podataka u sjednici korisnika, a model Project sadrži sve podatke koje controller šalje view-u na prikazivanje.**

**Uz navedene klase također su korištene dvije klase za pomoć pri radu programa. Dbhelper za komuniciranje sa bazom podataka te StringExt koji implementira metodu za rezanje string-a na željeni broj znakova pri spremanju u bazu podataka.**

**,**

# Opis implementacije perzistencije

Svi podaci kod zatvaranja aplikacije ostaju u bazi. Kod pokretanja bilo koje funkcionalnosti, ako je zadatak uspješno obavljen, prozor nas obavijesti o tome te se podaci u bazi planirano izmjene. Ako zadatak iz bilo kojeg razloga nije izvršen, podaci u bazi ostaju isti te nas prozor obavijesti o tome također. Tijekom rada aplikacije stvaraju se objekti koji sadrže informacije, no te informacije su kopije iz baze, te kod naglog prestanka rada zbog bilo kojeg slučaja podaci bi ostali konzistentni. Prilikom ponovnog pokretanja aplikacije informacije koje su prikazane bi bile iste kao i kod zatvaranja. To je većinom tako zbog organizacije logike baze i samih akcija korisnika. Korisnik u jednom trenutku vrši jednu radnju, te se kod završetka te radnje istovremeno ažurira baza podatka te prikaz na ekranu. Neće se dogoditi slučaj gdje korisnik istovremeno drži više tablica otvorenih te aplikacije čeka i blokira podatke. Ukratko, jedna akcija=jedna promjena u bazi, i to u najviše dvije relacije, odmah se događaju jedna iza druge. Jedini način da korisnik vidi drugačije podatke između dvije prijave u sustav jest ako gleda listu stvari koja uključuje druge timove ili osobe, a te osobe su kroz aplikaciju promijenile svoje stanje između ponovnog otvaranja forme koja prikazuje njihovo stanje.

Što se tiče mapiranja, djelomično je objašnjeno u prethodnom poglavlju bez korištenja ključne riječi mapiranje. Dakle Xpredložak, gdje X predstavlja jednu od riječi iz skupa [Zahtjev, Tim, KorisnickiRacun, ClanTima] služi za punjenje podacima iz baze tako što ga druga funkcija puni. Funkcija koja dohvaća podatke iz baze, te ujedno puni i predložak jest XTablica, gdje je X također jedna od riječi gore. Kod bilo koje promjene, pozivaju se funkcije klase Xtablica te puni potreban predložak podacima koji će se učitati ili dohvatiti iz baze. Nakon navedene promjene, ti podaci nam više nisu bitni osim u slučaju gdje se u određenim formama nalazi predložak korisnika koji je trenutno prijavljen, te predložak tima u koji je prikazan na ekranu. Na taj način prenošenja izbjegnuto je češći upit u bazu stvari koje su nam poznate od prije.

Baza podataka se sastoji od 4 relacije u kojoj su 2 entity-a i 2 vale object-a. Entity tablice su KorisnickiRacun u kojoj se nalaze podaci vezani za svakog korisnika posebno i Tim u kojoj se nalaze podaci za svaki tim posebno. Value object su veze između tima i osoba. Prva vrsta veze je zahtjev koji se šalje između tima i osobe, a druga vrsta je pripadnost osobe timu. Tablica Zahtjev se sastoji od ID-a osobe koja ga je poslala/primila, ID-a tima koji ga je poslao/primio te varijable „IniciraoKorisnik“ koja je „true“ ako je zahtjev upućen timu, a „false“ ako je zahtjev upućen osobi. Tablica ClanTima označava pripadnost osobi timu. Preciznije, sastoji se od ID-a osobe, ID-a tima u kojem se osoba nalazi, te varijablu Glavni koja označava je li osoba vođa tima. Naravno, u relaciji ClanTima s jednim ID-om tima, može biti vezana samo jedna osoba koja ima stupac Glavni postavljena na true. Primarni ključevi u istim tablicama su ID-evi ili kombinacija ID-eva.

S obzirom na ovisnosti primarnih ključeva, postojali su izazovi koje je trebalo riješiti. Kad korisnik napravi svoj tim, prvo je trebalo dodati podatke o timu u Tim tablicu, a tek onda u tablicu ClanTima kako bi se povezali tim i kreator tima. Dodatno, ako bi korisnik htio obrisati svoj tim, prvo je trebalo izbaciti iz tablice ClanTima sve osobe koje su u timu s danim ID-om, te nakon toga pobrisati i tim iz tablice Tim. Kod brisanja i zaključavanja tima, prvo je bilo potrebno pobrisati sve zahtjeve iz tablice Zahtjev, bilo pristigle ili poslane od korisnika, te nakon toga i tim. Dodatno, kada korisnik uđe u tim, u tablicu ClanTima se ubaci novi redak, ali iz tablice Zahtjev se izbrišu svi zahtjevi vezani za korisnika koji je upravo ušao u tim. Kada se tim zaključa, potrebno je promijeniti ID statusa tima u tablici Tim, no također i pobrisati sve zahtjeve u tablici Zahtjev za taj tim.

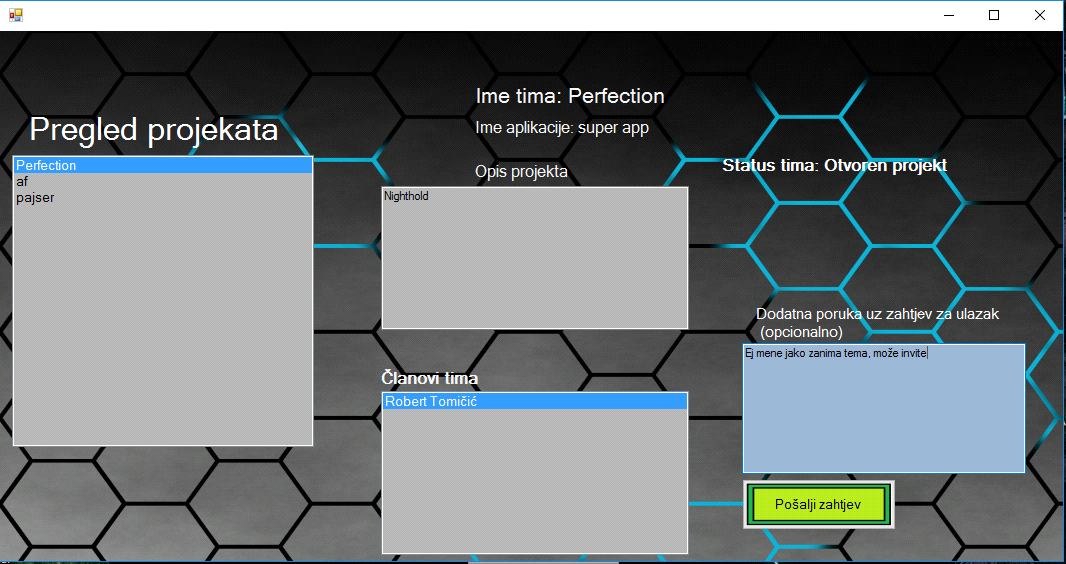
# Opis izgrađene desktop aplikacije

Kao što je već prije objašnjeno, sva interakcija korisnika s aplikacijom obavlja se preko prozora koji imaju svoje funkcionalnosti. Na slici 2 vidimo login ekran koji se pojavljuje pri paljenju aplikacije gdje korisnik(administrator ili običan korisnik) unosi svoje podatke te se prijavljuje gumbom "login". Opcionalno, ponuđeno mu je da korisničko ime bude automatski upisano u sljedećem loginu. Pritiskom gumba "login" u bazi se vrši upit kako bi se dobio identitet korisnika te taj identitet se predaje sljedećem prozoru radi jednostavnijeg rada aplikacije.



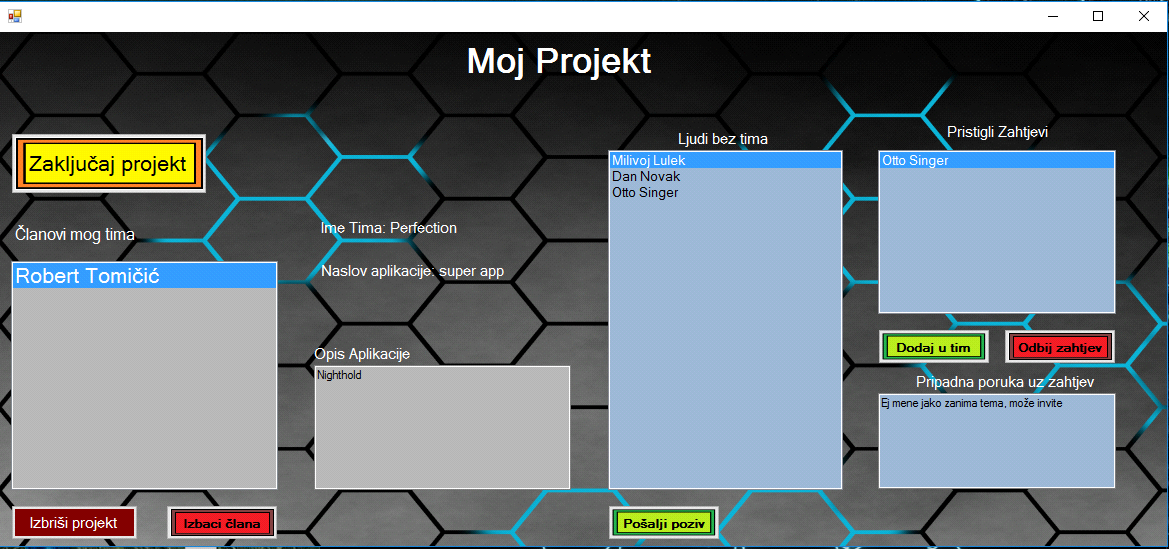
Slika 2. Login ekran

Na slici 3 prikazan je pregled svih timova. Biranje tima vide jednako svi korisnici, no administrator nema opciju za ulazak dok ostali korisnici nemaju opciju zaključaj/odobri. Trenutni prikaz je osoba koja traži ulazak u tim te šalje poruku za ulazak. Nakon pritiska gumba "pošalji zahtjev" u bazu u tablicu **Zahtjev** će se spremiti zahtjev od osobe koja ga šalje, tim koji se gleda, smjer zahtjeva, te dodatna poruka koju je korisnik unio opcionalno.



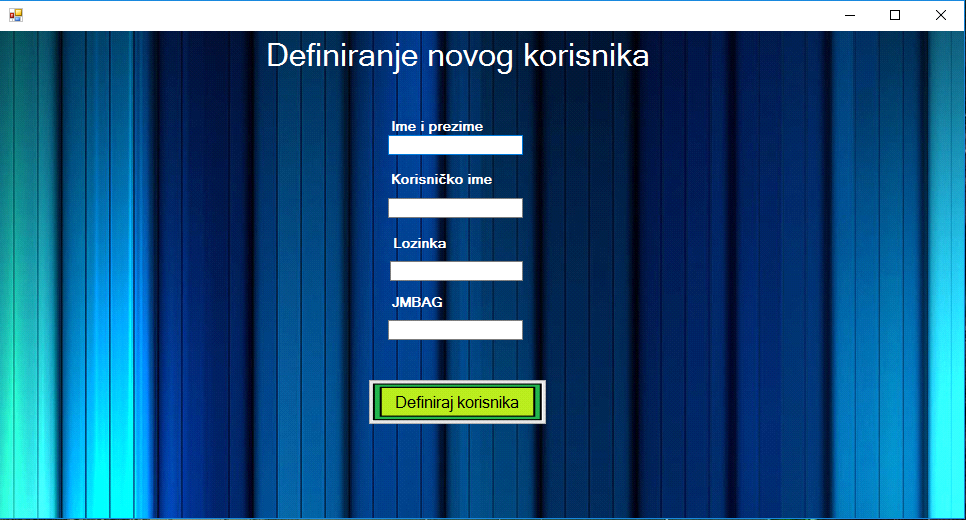
Slika 3. pregled timova

Na slici 4 prikazan je tim iz perspektive vođe tima. S lijeve strane ispisani su članovi koji se nalaze u timu, opis cijelog projekta te opcija da se projekt zaključa gdje tim više ne može primati i izbacivati članove. S desne strane prikazani su ljudi bez tima te ljudi koji žele ući u tim s pripadnom porukom. Vođa tima ima opciju izbrisati projekt, izbaciti člana, poslati poziv osobama bez tima, dodati ili odbiti osobu koja mu je poslala zahtjev. Sve te opcije nestaju pritiskom na gumb "zaključaj projekt". Korisnik koji je u ovom timu, a nije vođa tima će vidjeti sve što i vođa tima, no jedinu opciju koju može izabrati je "izlaz iz tima" čime korisnik prelazi u korisnika bez tima. Klikom na svaku od opcija vrši se interakcija s bazom podataka, a prozor ostaje upaljen dok se ne stisne X u desnom gornjem kutu prozora.



Slika 4. pregled projekta iz perspektive vođe tima

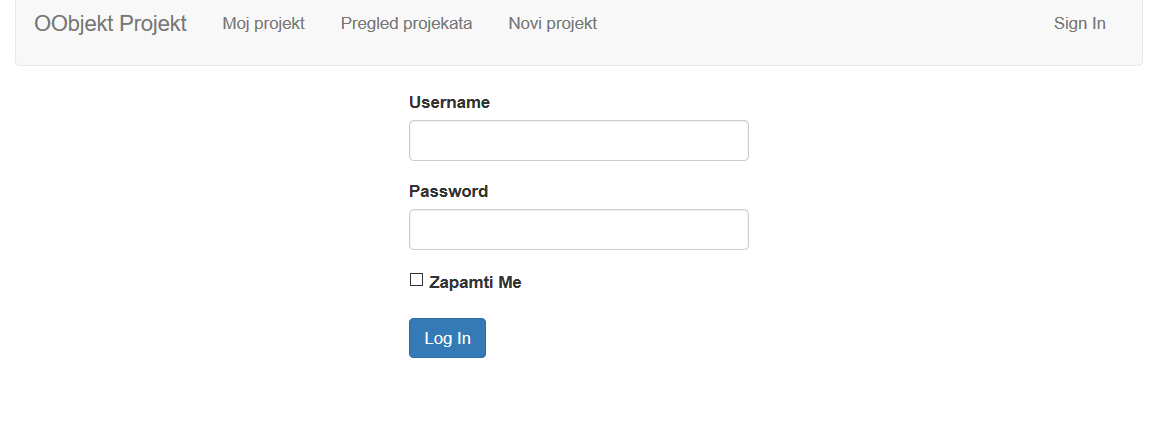
Na slici 4 vidimo kako administrator dodaje novog korisnika u sustav. Da bi došao do ovog ekrana, administrator se mora prijaviti te izabrati "definiranje novog korisnika" opciju. Nakon unosa podataka za novog korisnika te pritiskom gumba "definiraj korisnika" korisnik se dodaje u bazu u tablicu **KorisnickiRacun** te ta radnja omogućava korisniku da se prijavi u sustav.



Slika 5. definiranje novog korisnika

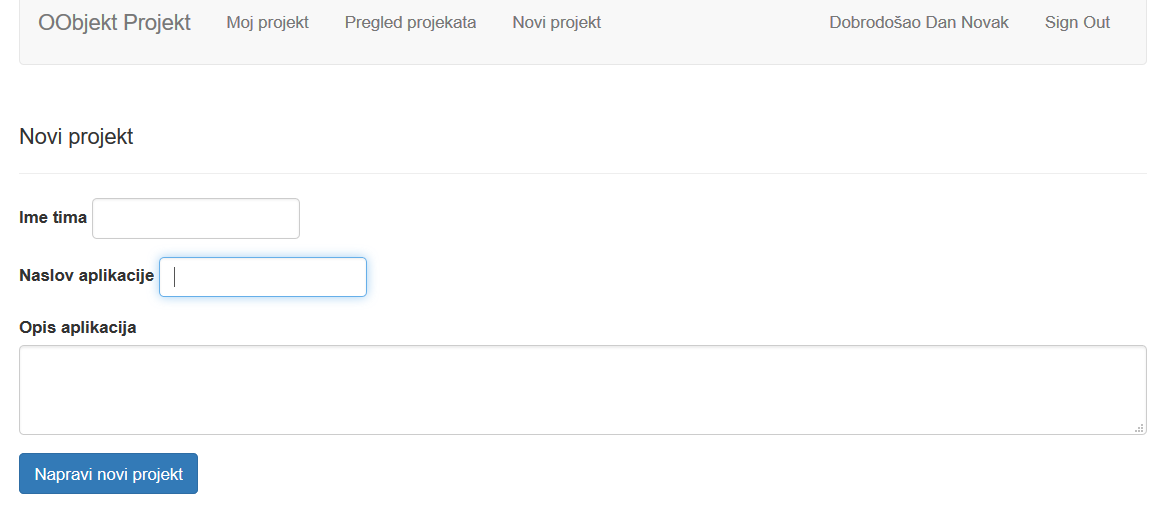
# Opis web aplikacije

Pri pokretanju web aplikacije korisniku se prikazuje sučelje za prijavu prikazano na slici 6. Sve ostale opcije na alatnoj traci nisu dostupne dok se korisnik ne prijavi u sustav.



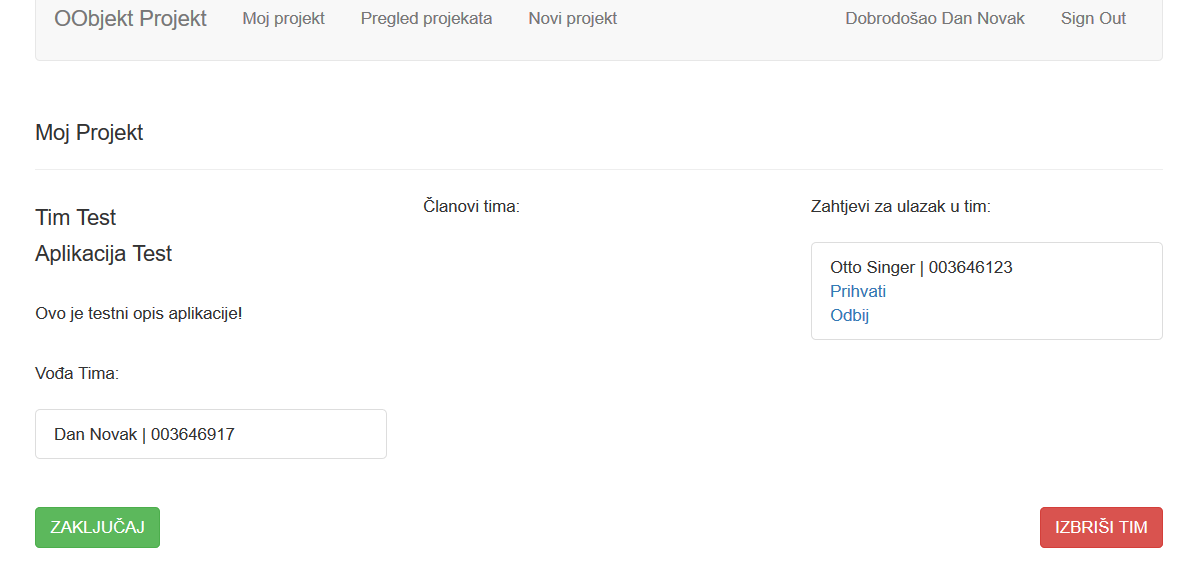
Slika 6. ekran za prijavu

Nakon što se korisnik prijavio u sustav, ukoliko se prvi put prijavljuje i još nije pristupio ni jednom timu, sustav će ga usmjeriti na *Pregled projekata*. Ako mu se ne sviđa ni jedan od trenutno dostupnih projekata, korisnik se može odlučiti za opciju *Novi projekt* na alatnoj traci koja će ga dovesti na sučelje za izradu novog projekta prikazano na slici 7.



Slika 7. ekran za izradu novog projekta

Kada je korisnik jednom izradio svoj projekt, postaje mu dostupna stranica *Moj Projekt* te se njegov projekt drugim korisnicima prikazuje na popisu projekata, a nedostupna postaje opcija za izradu novog projekta sve dok ne izbriše stari. Na profilnoj stranici svog projekta korisnik može vidjeti zahtjeve drugih korisnika za pristupanje projektnom timu te nakon što je projektni tim formiran, od vođe tima se očekuje da pritisne tipku za zaključavanje tima i čeka odobrenje admina. Korisnik može otključati tim i raditi promjene kao što su dodavanje i micanje članova sve dok nije još jednom zaključan od strane admina. Na slici 8 vidi se ekran *Moj projekt.*



Slika 8. ekran Moj projekt

# Baza Podataka

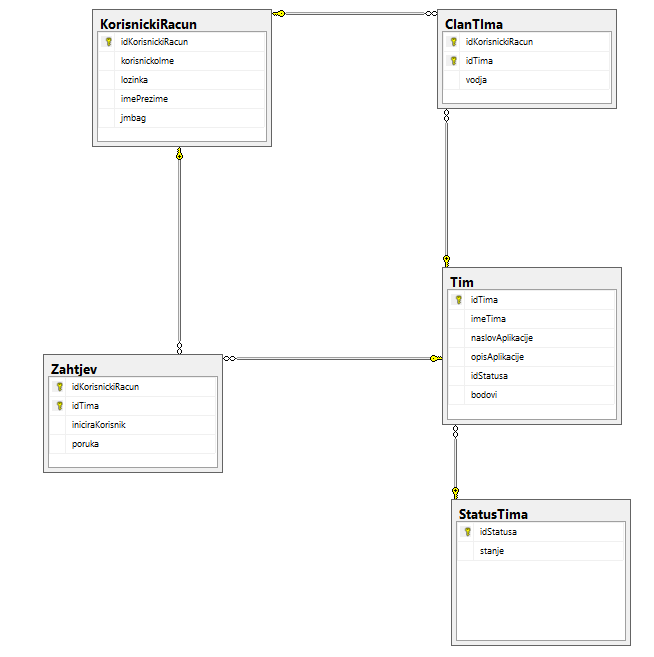
Za opis baze podataka korišten je jednostavan ER model baze podataka koji grafički prikazuje njen izgled. Iz njega se jednostavno dobiva relacijski model koji je korišten prilikom izrade baze u razvojnom okruženju. ER model se sastoji od entiteta koji su međusobno povezani vezama. Svaki sadrži atribute koji ga opisuju, između ostalih primarni ključ koji jedinstveno određuje n-torke unutar entiteta. Veze se sastoje od atributa koji su nastali od primarnih ključeva entiteta koje ta veza spaja. Popis nastalih relacija (primarni ključ je podcrtan):

* **KorisnickiRacun:** idKorisnickiRacun, korisnickoIme, lozinka, imePrezime, jmbag
* **Tim:** idTima, imeTima, naslovAplikacije, opisAplikacije, idStatusa, bodovi
* **ClanTIma:** idKorisnickiRacun, idTima, vodja
* **StatusTima:** idStatusa ,stanje
* **Zahtjev:** idKorisnickiRacun , idTima , iniciraKorisnik, poruka

KorisnickiRacun reprezentira račun svakog korisnika preko kojeg se prijavljuje u sustav te sadrži neke njihove osobne podatke koji se pojavljuju pri prikazu projekta. Tim reprezentira relaciju tima, sadrži neke osnovne informacije o timu te idStatusa koji označava zaključavanje tima. KorisnickiRacun i Tim povezani su relacijom ClanTIma uz pomoć koje se može saznati u kojem timu se nalazi svaki korisnik te tko je vođa tima uz pomoć atributa vođa. StatusTima je tablica koja sadrži dopuštene vrijednosti statusa tima (nezaključan, zaključan od vođe tima i zaključan od admina), a relacija Zahtjev prikazuje zahtjeve koje je korisnik naveden uz pomoć primarnog ključa korisničkog računa – idKorisnickiRacun, poslao timu prikazanog preko primarnog ključa tima - idTima.

Za implementaciju je korištena Microsoft SQL server baza podataka, a kao alat za manipuliranje nad bazom korišten je SQL Server DBMS (database managment system). Relacije su u bazi kreirane ručno kroz vizualni alat, nije pisan SQL kod za kreiranje. Atributi ID su u svakoj relaciji tipa "identity" što olakšava kod ubacivanja novih redaka u tablicu. Namješteni su strani ključevi i ostala ograničenja vezana uz atribute.

Na slici 9 je prikazan relacijski model baze podataka.



Slika 9. Relacijski model baze podataka