Fakultet elektrotehnike i računarstva

Zavod za primijenjeno računarstvo

**Objektno oblikovanje**

Seminar

Matea Bardun, 0036484568

Petar Jalušić, 0036484339

Toni Sredanović, 0036484318

Lucija Žbulj, 0036484115

Tanja Židić, 0036488247

Zagreb, 26.01.2018.

# Uvod

## Design story

Godina izrade

* 2018

Status

* U izradi

Korištena tehnologija

* .NET, Azure

Dostupnost

* Pristup sustavu omogućen je kroz desktop, web ili mobilnu aplikaciju za pregledavanje podataka dok se izmjena te brisanje pojedinih podataka ostvaruje prethodnom prijavom kroz jednu od prethodno navedenih aplikacija.

Poveznica

* <https://github.com/dashydasher/OO_Seminar>

Sadržaj

* Sustav za natjecateljsko plivanje je sustav izrađen za potrebe vođenja podataka u sklopu Hrvatskog plivačkog saveza. Struktura i sadržaj sustava prilagođeni su pružanju dostupnih podataka te njihovoj izmjeni i brisanju.

Podaci

* Sustav pruža informacije o bazenima, dvoranama, natjecanjima, utrkama, klubovima, plivačima, trenerima, sucima i rekordima u sklopu plivačkog saveza.

Ciljani korisnici

* Ministarstvo znanosti, obrazovanja i sporta, ustanove upravljanja sportskim objektima, plivački savezi, stručna i šira javnost.

## Opis domene

Plivački savez je organizacija koja upravlja svim podacima o natjecateljskom plivanju u državi gdje je osnovan.

Sustav za natjecateljsko plivanje im to omogućava na brz, efikasan i interaktivan način. Četiri glavna sudionika u interakciji sa sustavom su administrator, trener, sudac i neregistrirani korisnik.

Administrator ima potpunu kontrolu nad svim podacima u sustavu te mogućnost davanja ovlasti nad pojedinim dijelovima sustava trenerima i sucima.

Sudac se, da bi dobio određene ovlasti, najprije mora registrirati u sustav pomoću licence koju mu je dodijeli administrator. Tek nakon toga, uz uspješnu prijavu u sustav, ima mogućnosti zapisivanja rezultata pojedinih utrka.

Trener se mora registrirati u sustav pomoću licence koju mu je dodijelio administrator kako bi dobio svoje ovlasti koje uključuju registraciju kluba, prijavu plivača na utrku, dodavanje te izbacivanje plivača iz kluba, dodavanje natjecanja i pojedinih utrka za sukladno natjecanje.

Neregistrirani korisnik ima pristup pregledu svih relevantnih informacija.

Osnovni podaci o natjecanju su naziv, vrijeme početka, vrijeme kraja te dvorana u kojoj se održava. Svako natjecanje sadrži popis utrka te plivača iz pojedinih klubova koji se natječu.

Osnovni podaci za svaku utrku su vrijeme početka, vrijeme kraja, kategorija i spol plivača za koji se održava, duljina utrke, stil, te bazen i dvoranu u kojoj se održava. Uz to svaka utrka sadrži popis plivača koji su u njoj sudjelovali i suca koji je na njoj sudio.

Klub sadrži podatke o svom imenu te adresi, treneru koji ga vodi i svim plivačima koji su u njega učlanjeni.

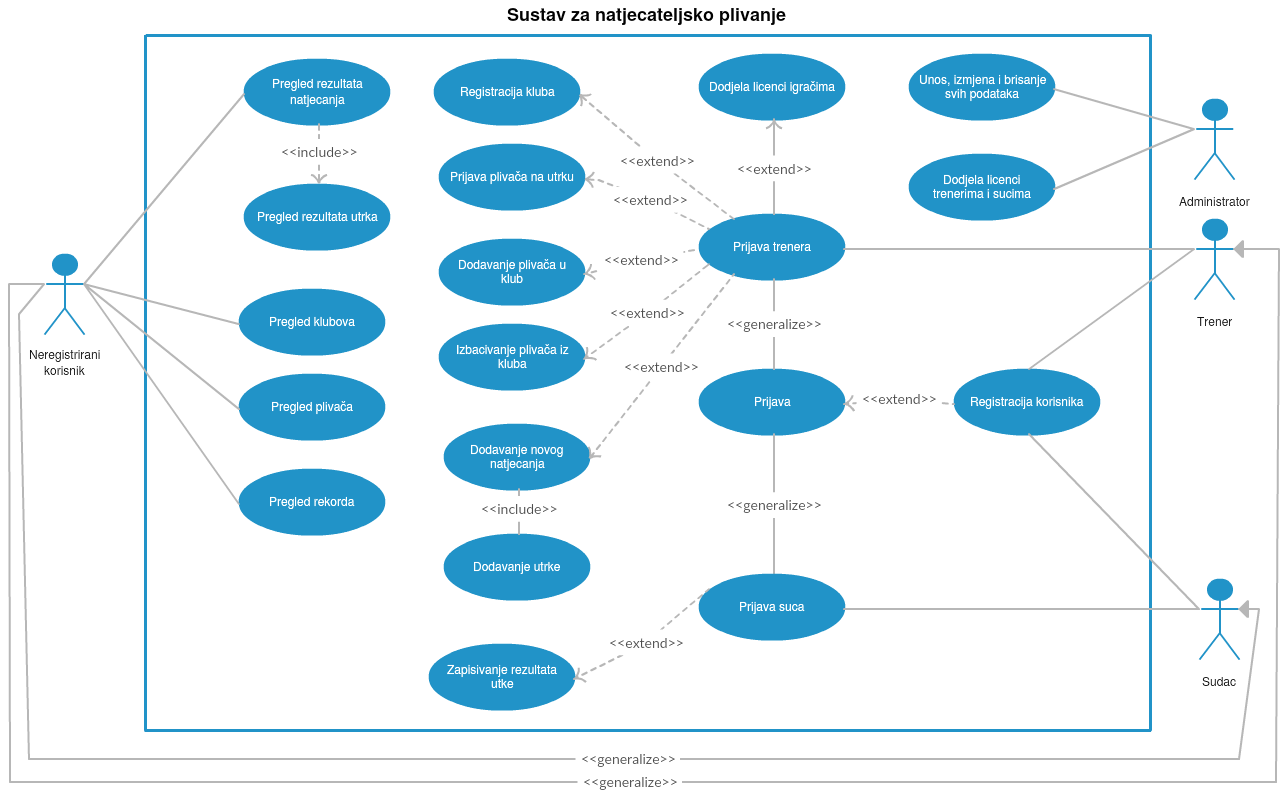
Svaki pojedini plivač je opisan imenom, prezimenom, datumom rođenja koji mu određuje kategoriju i spolom.

Rekordi posebno sadržavaju plivače kao nositelje pojedinih rekorda za sve kombinacije spola, kategorije, stila i duljine utrke.

# Zahtjevi

Zahtjeve korisnika opisat ćemo kroz use caseove. U nastavku slijedi slika koja prikazuje sve use caseove u interakciji korisnika sa sustavom. Nakon slike svi use caseovi detaljno su opisani.

## Use case dijagram



Slika 1 - Use case dijagram

## Use case razrada

**Pregled rezultata natjecanja**

Glavni sudionik

* neprijavljeni korisnik

Sudionici i interesanti

* sustav i ljudi koji prate plivačka natjecanja

Preduvjet

* u bazi podataka postoje natjecanja

Postconditions

* korisnik dobiva pregled svih natjecatelja i utrka odabranog natjecanja

Osnovni tijek

* korisnik odabire natjecanje
* sustav korisniku vraća popis svih natjecatelja i utrka

Alternativni tijek

* nema

**Pregled rezultata utrka**

Glavni sudionik

* neprijavljeni korisnik

Sudionici i interesanti

* sustav i ljudi koji prate plivačka natjecanja

Preduvjet

* u bazi podataka postoji natjecanje u kojem je odabrana utrka

Postconditions

* korisnik dobiva pregled rezultata svih utrka odabranog natjecanja

Osnovni tijek

* korisnik odabire natjecanje
* korisnik odabire popis svih utrka
* sustav vraća korisniku pregled rezultata svih utrka

Alternativni tijek

* nema

**Pregled rekorda po kategorijama**

Glavni sudionik

* neprijavljeni korisnik

Sudionici i interesanti

* sustav i ljudi koji prate plivačka natjecanja

Preduvjet

* nema

Postconditions

* korisnik dobiva pregled svih rekorda odabrane kategorije

Osnovni tijek

* korisnik odabire popis svih rekorda određene kategorije
* sustav korisniku vraća popis svih rekorda odabrane kategorije

Alternativni tijek

* nema

**Pretraživanje i pregled plivača**

Glavni sudionik

* neprijavljeni korisnik

Sudionici i interesanti

* sustav i ljudi koji prate plivačka natjecanja

Preduvjet

* nema

Postconditions

* korisnik dobiva pregled svih plivača

Osnovni tijek

* korisnik odabire popis svih plivača po kategorijama
* sustav korisniku vraća popis svih plivača po kategorijama

Alternativni tijek

* nema

**Prijava korisnika**

Glavni sudionik

* trener/sudac

Sudionici i interesanti

* treneri, suci i sustav

Preduvjet

* korisnik je registriran u sustavu

Postconditions

* korisnik je uspješno prijavljen u sustav

Osnovni tijek

* korisnik upisuje e-mail i lozinku
* korisnik odabire opciju Prijava
* sustav provjerava valjanost licence korisnika
* sustav dojavljuje korisniku uspješnu prijavu

Alternativni tijek

* korisnik upisuje pogrešan e-mail ili lozinku
* sustav korisniku dojavljuje grešku i vraća ga na stranicu za prijavu

Alternativni tijek (2)

* korisnik nema valjanu licencu za prijavu
* sustav korisniku dojavljuje da nema valjanu licencu

**Iščlanjenje plivača iz kluba**

Glavni sudionik

* trener

Sudionici i interesanti

* plivač, trener i sustav

Preduvjet

* plivač je član trenerovog kluba

Postconditions

* plivač je uspješno iščlanjen iz kluba

Osnovni tijek

* trener odabire popis svih plivača svog kluba
* sustav treneru vraća popis svih plivača njegovog kluba
* trener odabire određenog plivača
* sustav iščlanjuje plivača iz kluba
* sustav treneru dojavljuje uspješnost iščlanjenja plivača

Alternativni tijek

* nema

**Dodavanje plivača u klub**

Glavni sudionik

* trener

Sudionici i interesanti

* plivač, trener i sustav

Preduvjet

* plivač postoji u bazi („registriran je“)

Postconditions

* plivač je uspješno dodan u klub

Osnovni tijek

* trener odabire popis svih plivača
* sustav treneru vraća popis svih plivača
* trener odabire određenog plivača
* sustav dodaje plivača u trenerov klub
* sustav treneru dojavljuje uspješnost dodavanja plivača

Alternativni tijek

* plivač je već član nekog kluba
* sustav dojavljuje neuspješnost dodavanja plivača

**Prijavljivanje plivača za utrku**

Glavni sudionik

* trener

Sudionici i interesanti

* plivač, trener i sustav

Preduvjet

* plivač je član trenerovog kluba

Postconditions

* plivač je uspješno prijavljen za utrku

Osnovni tijek

* trener odabire natjecanje
* trener odabire utrku
* trener odabire popis svih plivača svog kluba
* sustav treneru vraća popis svih plivača
* trener odabire određenog plivača
* sustav prijavljuje plivača za utrku
* sustav treneru dojavljuje uspješnost prijave plivača

Alternativni tijek

* plivač je već prijavljen za maksimalan broj utrka na tom natjecanju
* sustav dojavljuje neuspješnost prijave plivača

Alternativni tijek (2)

* plivaču je istekla licenca
* sustav dojavljuje istek valjanosti licence plivača

**Zapisivanje rezultata utrke**

Glavni sudionik

* sudac

Sudionici i interesanti

* plivač, sudac i sustav

Preduvjet

* utrka je završena

Postconditions

* sudac je uspješno unio rezultate utrke

Osnovni tijek

* sudac odabire natjecanje
* sustav sucu vraća popis svih utrka tog natjecanja
* sudac odabire utrku
* sustav sucu vraća popis svih plivača te utrke
* sudac odabire određenog plivača i upisuje njegovo vrijeme
* sustav zapisuje rezultat utrke odabranog plivača
* sustav sucu dojavljuje uspješnost zapisa rezultata

Alternativni tijek

* sudac je upisao nevaljano vrijeme
* sustav dojavljuje pogrešan unos

**Dodavanje novog natjecanja**

Glavni sudionik

* trener

Sudionici i interesanti

* treneri i sustav

Preduvjet

* natjecanje se nalazi u mjestu u kojem je trenerov klub

Postconditions

* trener je dodao novo natjecanje
* poslana je obavijest ostalim trenerima

Osnovni tijek

* trener odabire stvaranje novog natjecanja
* trener upisuje podatke o natjecanju (naziv, vrijeme početka, vrijeme kraja i dvoranu)
* sustav treneru dojavljuje uspješnost dodavanja natjecanja
* sustav šalje obavijest ostalim trenerima o održavanju natjecanja

Alternativni tijek

* dvorana je zauzeta u vrijeme održavanja natjecanja
* sustav dojavljuje grešku i vraća trenera na stranicu dodavanja novog natjecanja

**Dodavanje utrke**

Glavni sudionik

* trener

Sudionici i interesanti

* plivač, sudac i sustav

Preduvjet

* dodano je novo natjecanje

Postconditions

* trener je dodao novu utrku
* sucu je poslana obavijest o održavanju utrke

Osnovni tijek

* trener odabire natjecanje kojem želi dodati utrku (natjecanje u matičnom gradu svog kluba)
* trener upisuje podatke o utrci (kategorija, stil, udaljenost, spol, bazen, vrijeme početka i završetka, sudac)
* sustav treneru dojavljuje uspješnost dodavanja utrke
* sustav sucu šalje obavijest o održavanju utrke

Alternativni tijek

* trener je odabrao suca koji ima nevažeću licencu
* sustav dojavljuje grešku i vraća trenera na stranicu dodavanja nove utrke

**Dodjela licence**

Glavni sudionik

* administrator

Sudionici i interesanti

* sudac/trener i sustav

Preduvjet

* osoba nema važeću licencu

Postconditions

* dodijeljena je nova licenca

Osnovni tijek

* administrator odabire osobu kojoj želi dodijeliti licencu
* sustav administratoru dojavljuje uspješnost dodjele licence
* sustav osobi šalje obavijest o dodjeli licence

Alternativni tijek

* administrator je odabrao osobu koja već ima važeću licencu
* sustav dojavljuje grešku i vraća administratora na stranicu dodjele licence

**Dodavanje novog kluba**

Glavni sudionik

* administrator

Sudionici i interesanti

* trener i sustav

Preduvjet

* administrator je prijavljen u sustav

Postconditions

* dodan je novi klub

Osnovni tijek

* administrator upisuje podatke o novom klubu (naziv, adresa, trener)
* sustav administratoru dojavljuje uspješnost stvaranja kluba
* sustav treneru šalje obavijest o dodjeli kluba

Alternativni tijek

* administrator je odabrao trenera koji ima nevažeću licencu
* sustav dojavljuje grešku i vraća administratora na stranicu dodavanja kluba

**Registracija korisnika**

Glavni sudionik

* trener/sudac

Sudionici i interesanti

* trener, sudac i sustav

Preduvjet

* korisnik ima licencu

Postconditions

* korisnik je registriran u sustav

Osnovni tijek

* korisnik upisuje podatke o registraciji (ime, prezime, e-mail, lozinka, id licence)
* sustav u popisu svih licenci trenera/sudaca pronalazi licencu korisnika
* sustav korisnika deklarira kao trenera/suca sudeći prema vrsti pronađene licence
* sustav korisniku dojavljuje uspješnost registracije

Alternativni tijek

* korisnik je upisao nevaljanu e-mail adresu
* sustav dojavljuje grešku i vraća korisnika na stranicu za registraciju

**Dodavanje plivača u sustav**

Glavni sudionik

* administrator

Sudionici i interesanti

* plivač i sustav

Preduvjet

* administrator je prijavljen u sustav

Postconditions

* dodan je novi plivač

Osnovni tijek

* administrator upisuje podatke o plivaču (ime, prezime, datum rođenja, spol)
* administrator plivaču dodjeljuje licencu
* sustav administratoru dojavljuje uspješnost dodavanja plivača

Alternativni tijek

* plivač već postoji u sustavu
* sustav dojavljuje grešku i vraća administratora na stranicu dodavanja plivača

**Produljenje plivačke licence**

Glavni sudionik

* administrator

Sudionici i interesanti

* plivač i sustav

Preduvjet

* plivač postoji u sustavu i licenca mu je istekla

Postconditions

* plivaču je produljena licenca

Osnovni tijek

* administrator odabire plivača kojem će produljiti licencu
* administrator plivaču dodjeljuje licencu
* sustav administratoru dojavljuje uspješnost produljenja licence

Alternativni tijek

* nema

**Dodavanje nove dvorane u sustav**

Glavni sudionik

* administrator

Sudionici i interesanti

* sustav

Preduvjet

* administrator je prijavljen u sustav

Postconditions

* dodana je nova dvorana

Osnovni tijek

* administrator upisuje podatke o novoj dvorani (naziv, adresa)
* sustav administratoru dojavljuje uspješnost dodavanja dvorane

Alternativni tijek

* naziv dvorane već postoji
* sustav administratoru dojavljuje grešku i vraća ga na stranicu dodavanja dvorane

**Dodavanje bazena**

Glavni sudionik

* administrator

Sudionici i interesanti

* sustav

Preduvjet

* administrator je prijavljen u sustav

Postconditions

* dodan je novi bazen u dvoranu

Osnovni tijek

* administrator odabire dvoranu u koju će dodati bazen
* administrator upisuje podatke o bazenu (duljina)
* sustav administratoru dojavljuje uspješnost dodavanja bazena

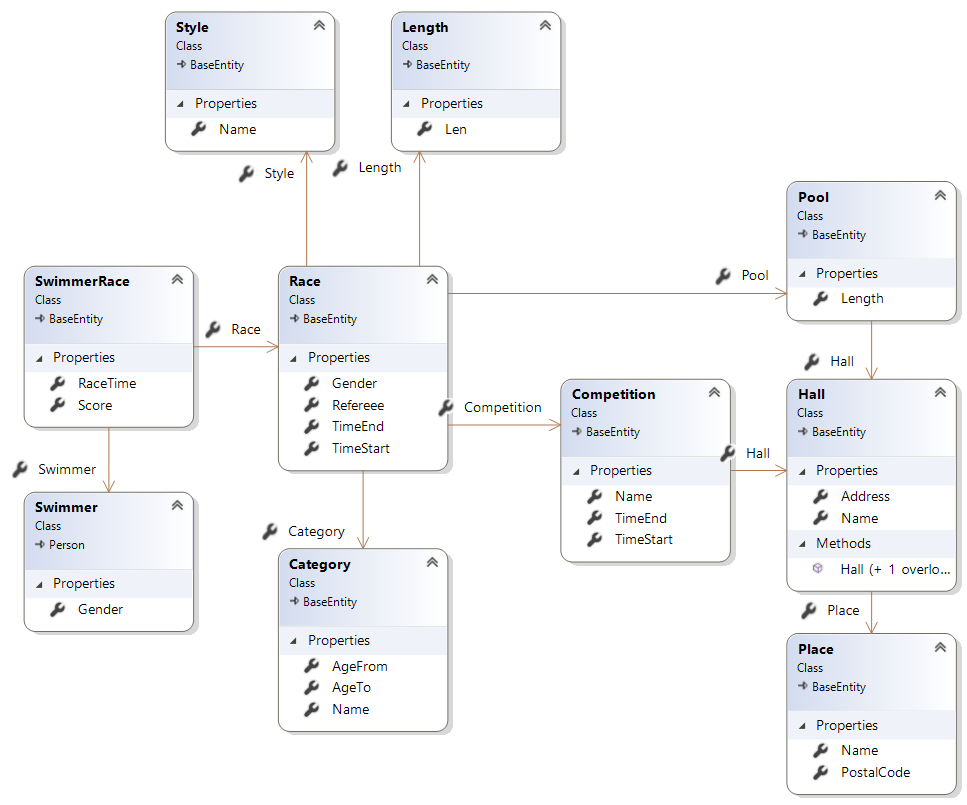
Alternativni tijek

* administrator je upisao nevaljanu duljinu bazena
* sustav administratoru dojavljuje grešku i vraća ga na stranicu dodavanja bazena

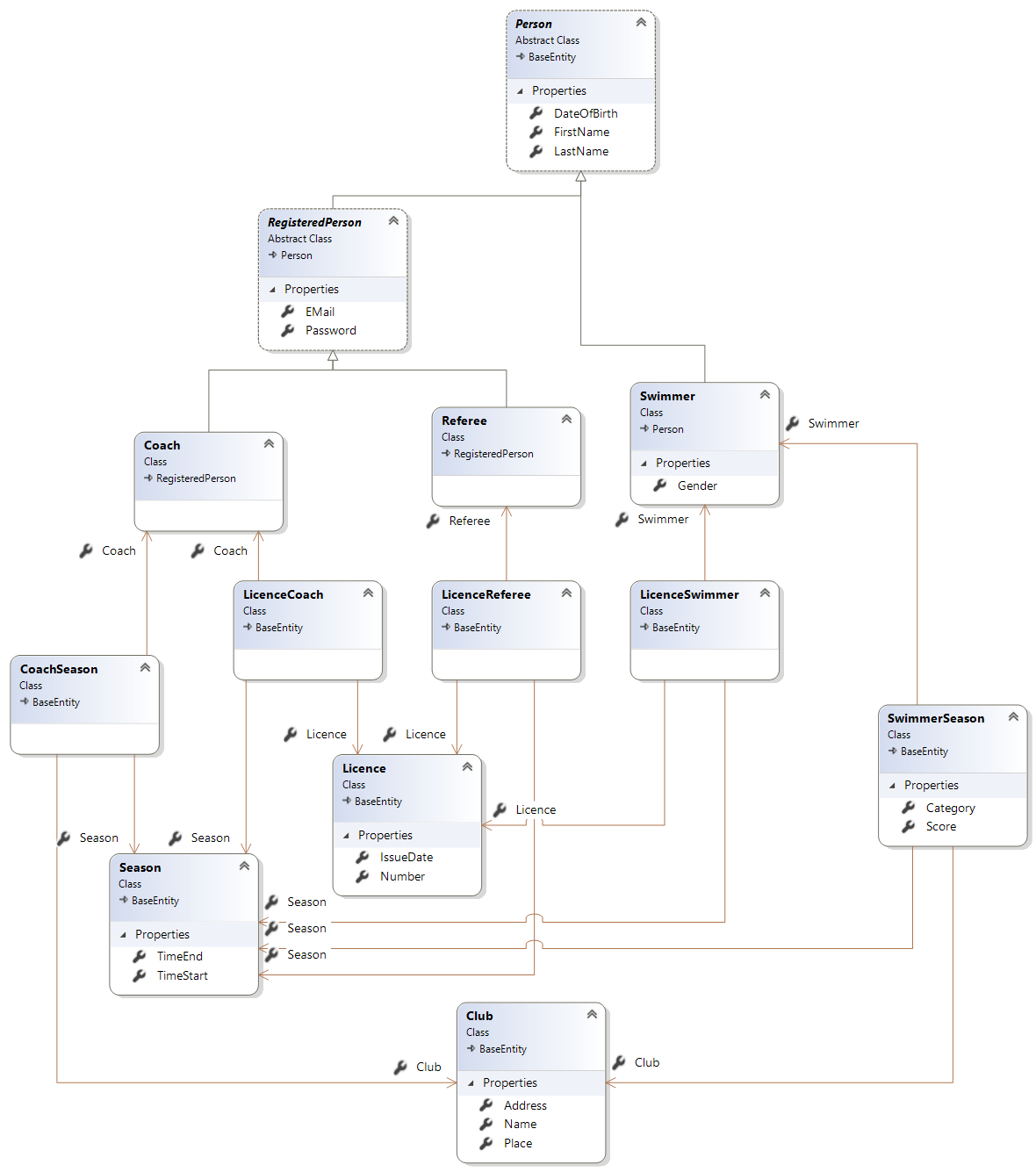
# Opis modela

## Class dijagrami

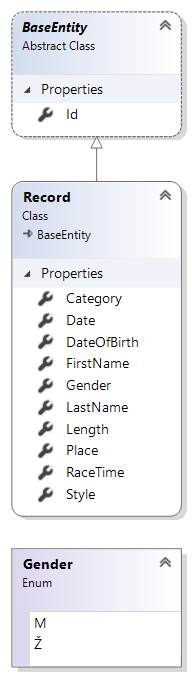
Iz svake od sljedećih slika koje prikazuju dijagram klasa isključeno je radi preglednosti nasljeđivanje klase BaseEntity koja sadrži samo atribut Id, no to se svojstvo može iščitati iz klase.



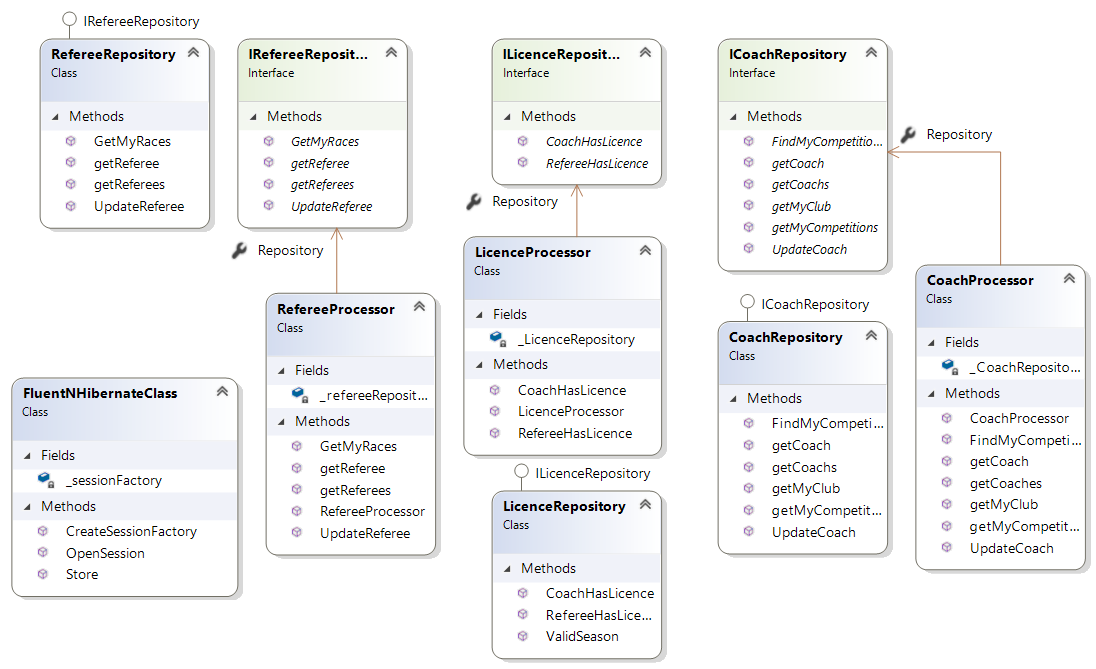
*Slika 3.1 Dijagram klasa dijela objektnog modela*



*Slika 3.2 Dijagram klasa dijela objektnog modela*



*Slika 3.3 Dijagram klasa dijela objektnog modela*

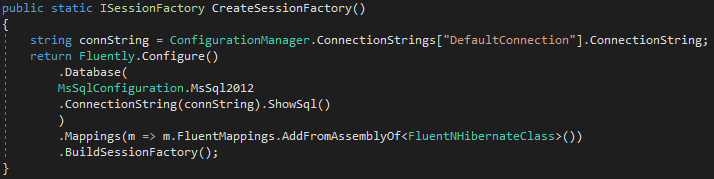


*Slika 3.4 Dijagram klasa dijela repozitorija*

## Opis pojedinih klasa

S obzirom da je model domene složen te se u objektnom modelu nalazi mnogo klasa, probali smo ih podijeliti na neke smislene cjeline. Klase sadrže samo atribute koji služe za mapiranje. Uzmimo na primjer klasu Hall iz slike 3.1: ona sadrži atribute Name, Address i Place što je ustvari referenca na klasu Place, a to se postiže upravo mapiranjem.

Na slici 3.4 prikazani su repozitoriji, sučelja repozitorija i procesori. Također je i prikazana FluentNHibernateClass klasa koju smo kreirali za spajanje s bazom te kreiranja i otvaranja sesije kada je to potrebno. Metoda CreateSessionFactory prikazana je na slici 3.5 u nastavku. Ona koristi Fluently kako bi uspostavila konekciju s bazom i pritom vrši sva definirana mapiranja naših klasa te vraća objekt tipa ISessionFactory koji se kasnije koristi za komunikaciju s bazom.



*Slika 3.5 Spajanje s bazom i mapiranja*

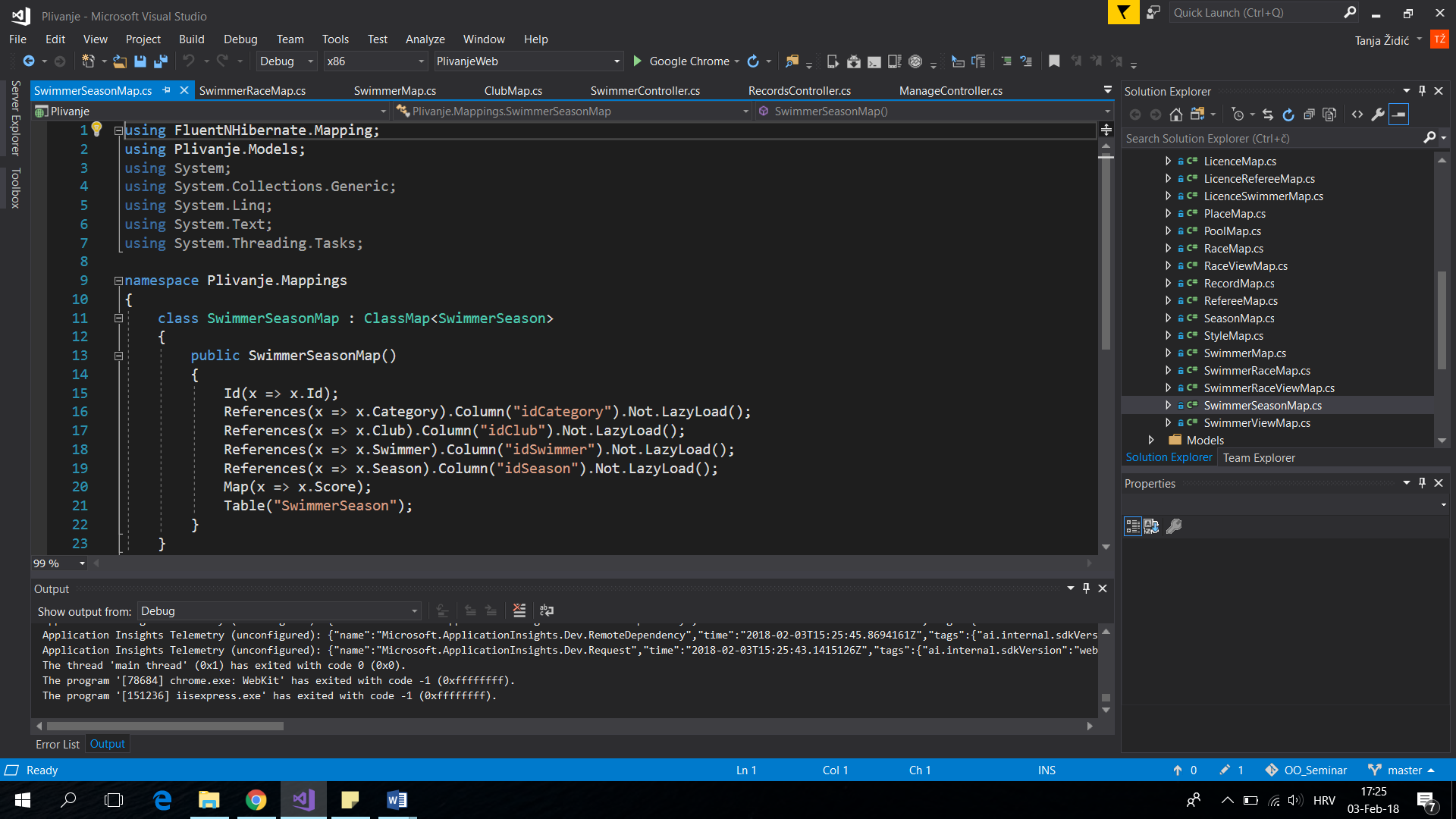
Repozitoriji i procesori služe za obavljanje operacija koje koriste bazu podataka (sve CRUD operacije).

# Opis implementacije perzistencije

Kao bazu podataka koristili smo SQL Server bazu podataka pohranjenu na Azure serveru, te smo koristeći FluentNHibernate obavili OR mapiranje između tablica u relacijskog bazi podataka i klasa u našem modelu domene.

4.1. Mapiranje i repozitoriji

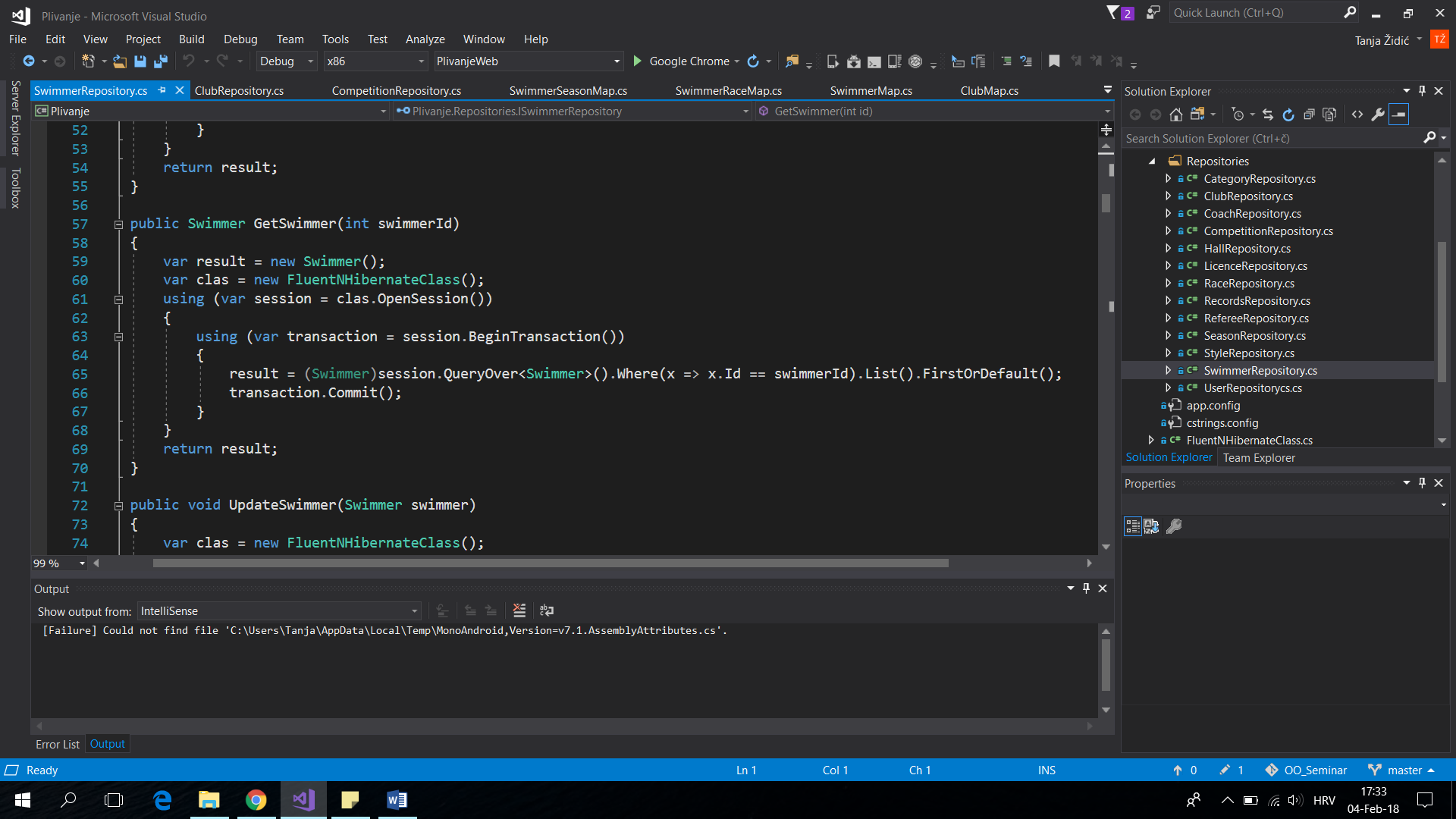
Sva mapiranja nalaze se unutar datoteke *Mappings,* te su ostvarena na način da za svaku klasu imamo mapiranje koje nasljeđuje *ClassMap* nad određenom baznom klasom. Za svaku klasu mapiraju se sva njezina svojstva koristeći neke od ključnih riječi FluentNhinernate, pa tako Id označava da se radi o identifikatoru relacije, References označava da se taj stupca referencira na neku drugu klasu u modelu, dok Map označava običan atribut pojedine klase.



*Slika 4.1. Mapiranje klase SwimmerSeason*

U repozitorijima se nalazi sva logika za interakciju s bazom podataka ( poput metoda dohvata, ažuriranja i brisanja). Osim jednostavnih upita definirani su i malo složeniji upiti nad bazom koji se vrše nad više relacija. Svaki repozitorij je definiran sučeljem koje govori koje sve metode mora repozitorij sadržavati kako bi zadovoljio to sučelje. Također sučelje je napravljeno zbog mock testiranja.

Svaka od metoda koristi svoju transakciju za spremanje i dohvat podataka. Nakon obavljanja upita transakcija se commita.

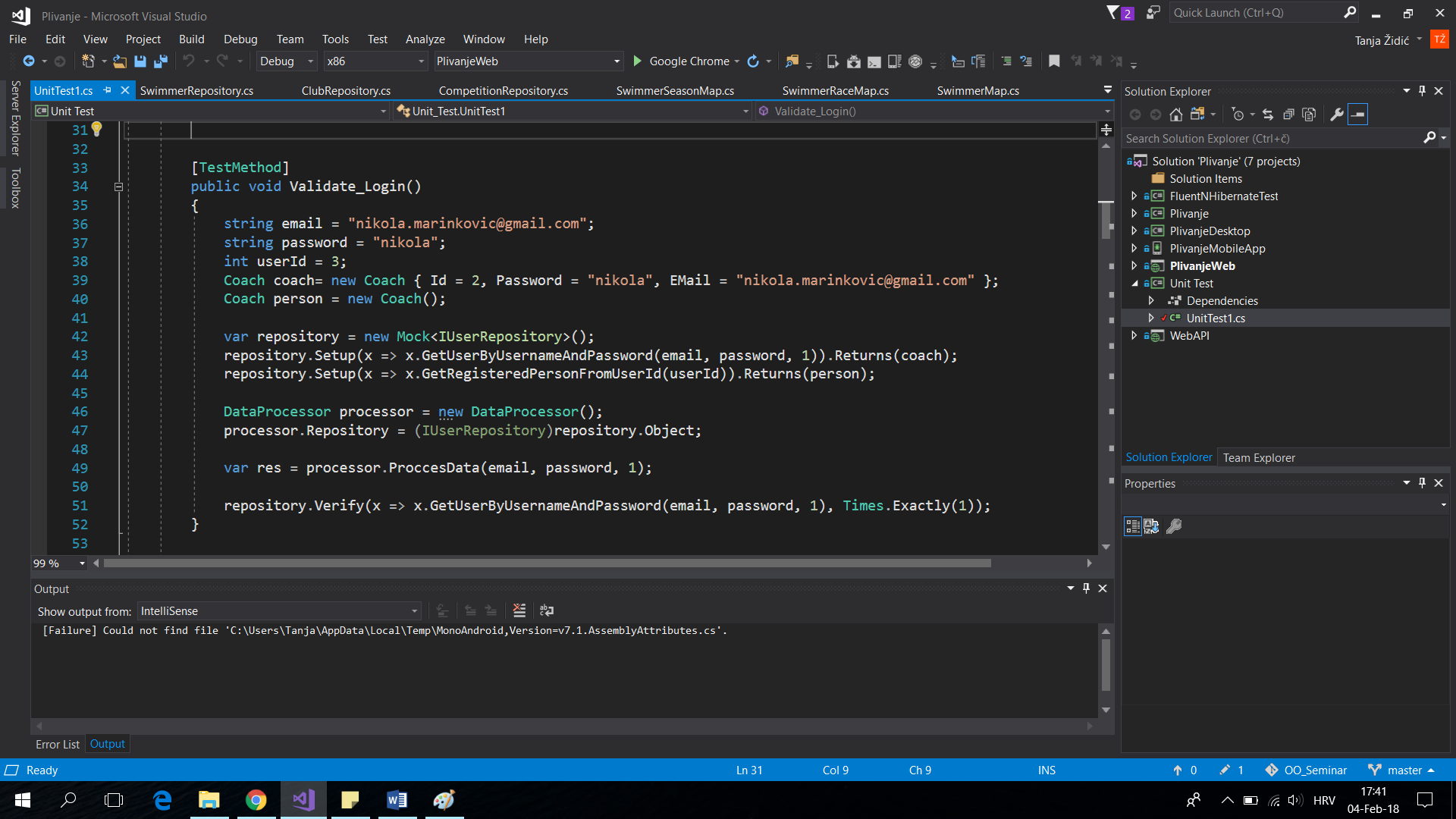


*Slika 4.2. Metoda unutar SwimmerRepositoryja*

4.2. Testiranje

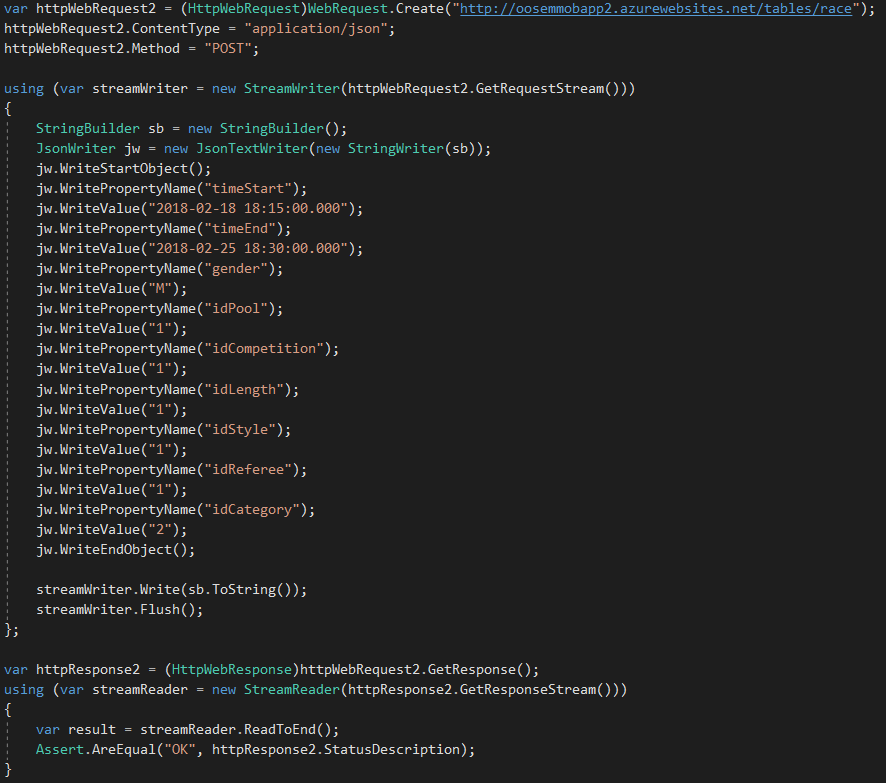
Prvi dio testiranja odnosi se na testiranje repozitorija jer nam se sva funkcionalnost i interakcija sa bazom podataka uglavnom odvijala preko njih.

Na sljedećoj slici ( slika 4.3.) prikazan je jedan primjer mock testiranja koristeći *Moq* framework. U ovoj metodi se testira prijavljivanje korisnika u sustav, odnosno testira se metoda ProccesData u DataProcessor-u koja poziva metodu iz repozitorija UserRepostitory. U prvom koraku testiranja se postavljaju mock objekti koje ćemo koristiti, te ostali objekti koji će nam biti potrebni za provedbu testiranja. U drugom koraku pozivamo metodu koju želimo testirati, dok u trećem provjeravamo da li je ista uzrokovala poziv metode GetUserByUsernameAndPassword i to točno jednom. Ukoliko je metoda pozvana uvjeti su zadovoljeni i test prolazi. Ukoliko bi provjeravali neku drugu metodu, koja se ne poziva unutar testirane, test bi završio padom.



*Slika 4.3. Primjer mock testa*

Drugi dio testiranja odnosi se na testiranje API-ja. API je korišten u mobilnoj aplikaciji na način da sva komunikacija aplikacije s bazom ide upravo preko API-ja. Na slici 4.4 prikazan je isječak koda jednog testa gdje se između ostalog dodaje nova utrka preko API-ja. Prvo kreiramo web request s adresom <http://oosemmobapp2.azurewebsites.net/tables/race> na koju šaljemo podatke u json formatu i POST zahtjev. Nakon toga otvaramo stream writer i zapisujemo sve podatke koje utrka mora imati. Tada dohvaćamo response i provjeravamo je li status 200 OK. Da ne bi svakim pokretanjem testova kreirali novi zapis u bazi, nakon ovog dijela koda izvršava se i brisanje tog upravo dodanog zapisa iz baze, naravno uz provjeru je li bilo uspješno.



*Slika 4.4. Primjer dodavanja novog kluba*

# Opis izgrađene desktop aplikacije

Desktop aplikacija namijenjena je trenerima, sucima i neregistriranim korisnicima (simpatizerima plivanja). Aplikacija omogućuje pregled i uređivanje natjecanja, utrka, klubova i plivača te pregled rekorada.

## 5.1. Opis WPF-a

Aplikacija je izgrađena korištenjem WPF-a i MVVM patterna. MVVM razdvaja strukturu koda na Model, View i ViewModel. Model sadrži podatke aplikacije, View predstavlja formu, prikaz podataka korisniku, a ViewModel služi kao poveznica Modela i Viewa. ViewModel sadrži mogućnost data binding-a koje služi za pružanje podataka Viewu, dopušta korisniku interakciju s podacima i enkapsulira interakcijsku logiku za View. Svaki View je povezan sa svojim ViewModelom. U nastavku slijedi opis pojedinih ViewModel-a.

*ClubViewModel* omogućuje dohvat svih klubova i dohvat trenerovog kluba iz baze podataka korištenjem *ClubProcessor-a* i *ClubRepository-a.*

*CompetitionViewModel* omogućuje dohvat svih natjecanja, dohvat trenerovih natjecanja i dodavanje natjecanja u bazu podataka uz pomoć naredbi iz *ClubProcessor-a* i *ClubRepository-a*.

*RaceViewModel* nudi učitavanje utrka određenog natjecanja i dodavanje nove utrke korištenjem naredbi *RaceProcessor-a* i *RaceRepository-a*.

*RecordViewModel* nudi dohvat muških, odnosno ženskih rekorada uz pomoć naredbi *RecordProcessor-a* i *RecordRepository-a*.

*RegisterViewModel* omogućuje registraciju trenera, odnosno suca njihovim dodavanjem u bazu podataka pomoću *CoachProcessor-a* i *CoachRepository-a*, odnosno *RefereeProcessor-a* i *RefereeRepository-a*.

*SwimmerRaceViewModel* nudi dohvat plivača određene utrke te prijavljivanje plivača na utrku korištenjem *SwimmerProcessor-a*, *ClubProcessor-a* i *RaceProcessor-a*.

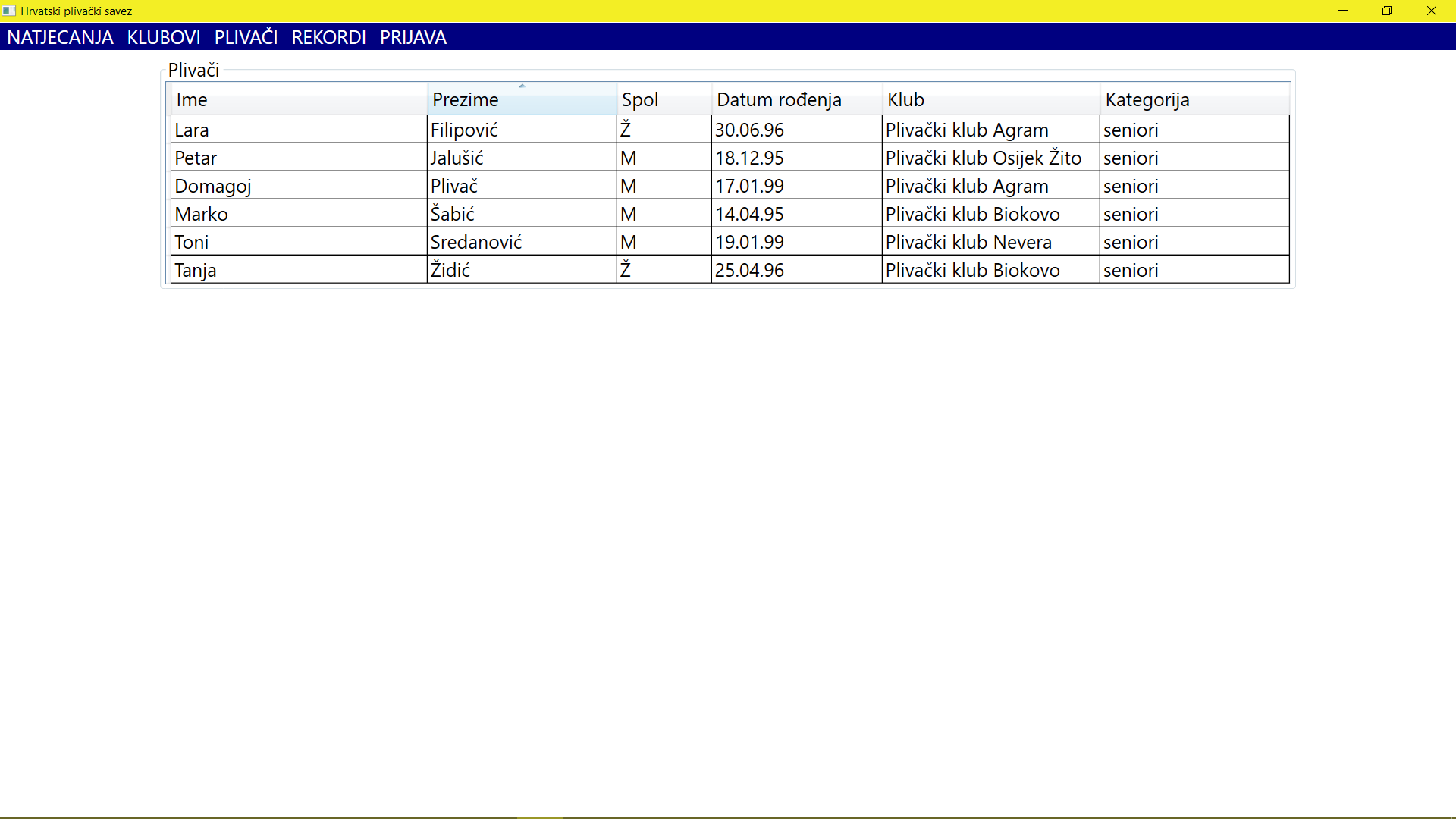
*SwimmerResultViewModel* omogućuje zapis rezultata određene utrke pomoću naredbi iz *RaceProcessor-a* i *SwimmerProcessor-a*.

*SwimmerViewModel* omogućuje dohvat plivača određene kategorije i dohvat plivača određenog kluba. Osim toga, nudi i mogućnost učlanjenja i iščlanjenja plivača iz kluba pomoću *SwimmerProcessor-a* i *ClubProcessor-a*.

*UserViewModel* omogućuje prijavu korisnika korištenjem *CoachProcessor-a*, *RefereeProcessor-a* i *LicenceProcessor-a*.

## 5.2. Korištenje aplikacije

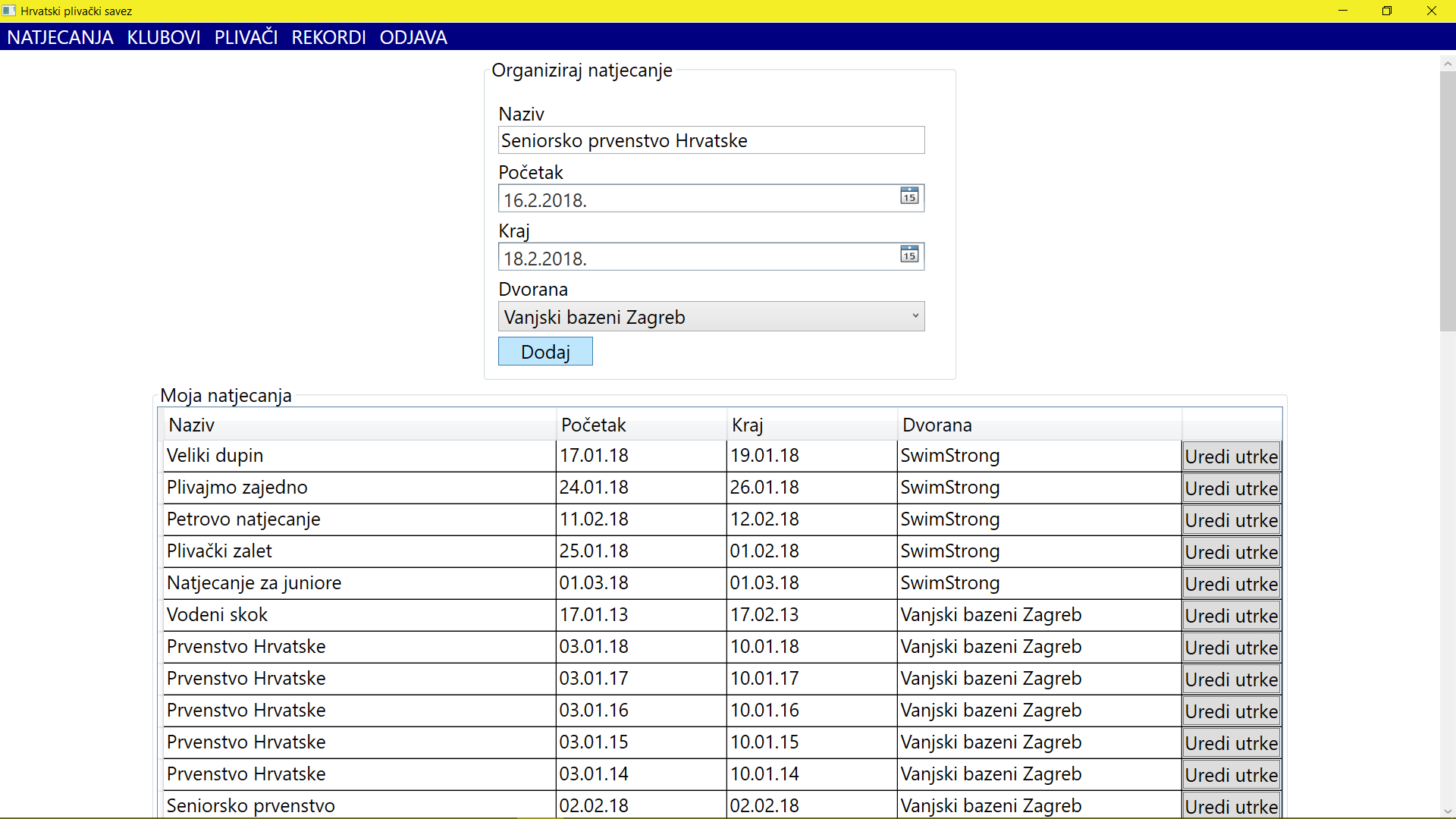
Neregistrirani korisnici imaju mogućnost pregleda rezultata natjecanja i njihovih utrka, pregled klubova, plivača i rekorda. Klikom na gumbe *Natjecanja*, *Klubovi*, *Plivači* i *Rekordi*, korisnicima se otvaraju željene stranice koje zatim mogu pregledati. *Slika 5.1* prikazuje plivače kategorije seniori.



*Slika 5.1. Prikaz plivača kategorije “seniori” neprijavljenom korisniku*

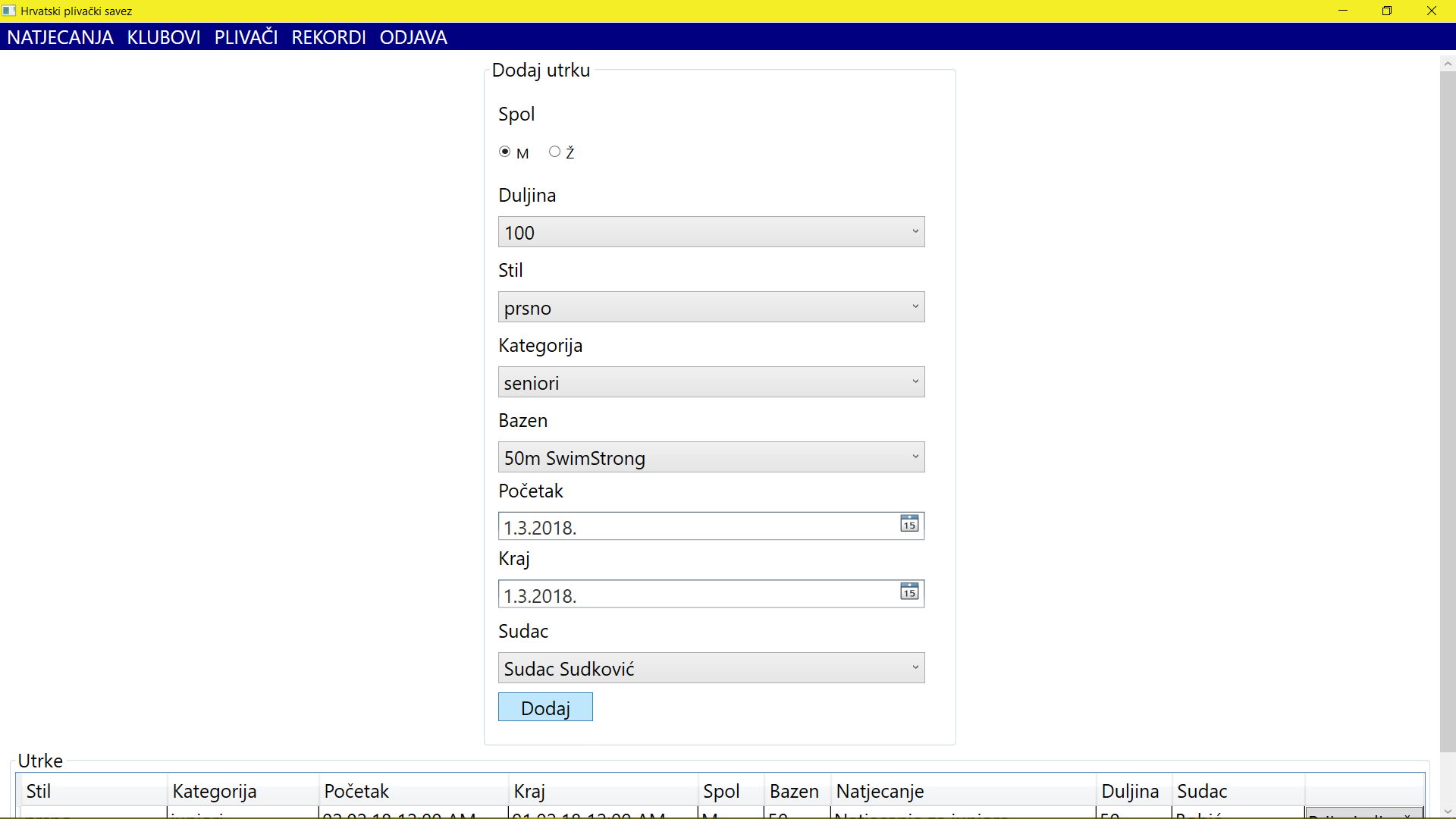
Klikom na gumb *Prijava* registrirani korisnici mogu se prijaviti u aplikaciju. Mogućnost prijave imaju treneri i suci kojima je dodijeljena važeća licenca. Ukoliko korisnik nije registriran, može to učiniti klikom na gumb *Registriraj se* te upisom traženih podataka potrebnih za registraciju.

Treneri nakon prijave imaju mogućnost pregleda informacija o svom klubu te učlanjenja i iščlanjenja plivača iz svog kluba. Osim toga, omogućena im je i prijava plivača na utrku, te organiziranje novih natjecanja i utrka (*Slika 5.2, Slika 5.3*). Za organiziranje natjecanja trener treba upisati tražene podatke i pritisnuti gumb *Dodaj* kojim se dodaje novo natjecanje.



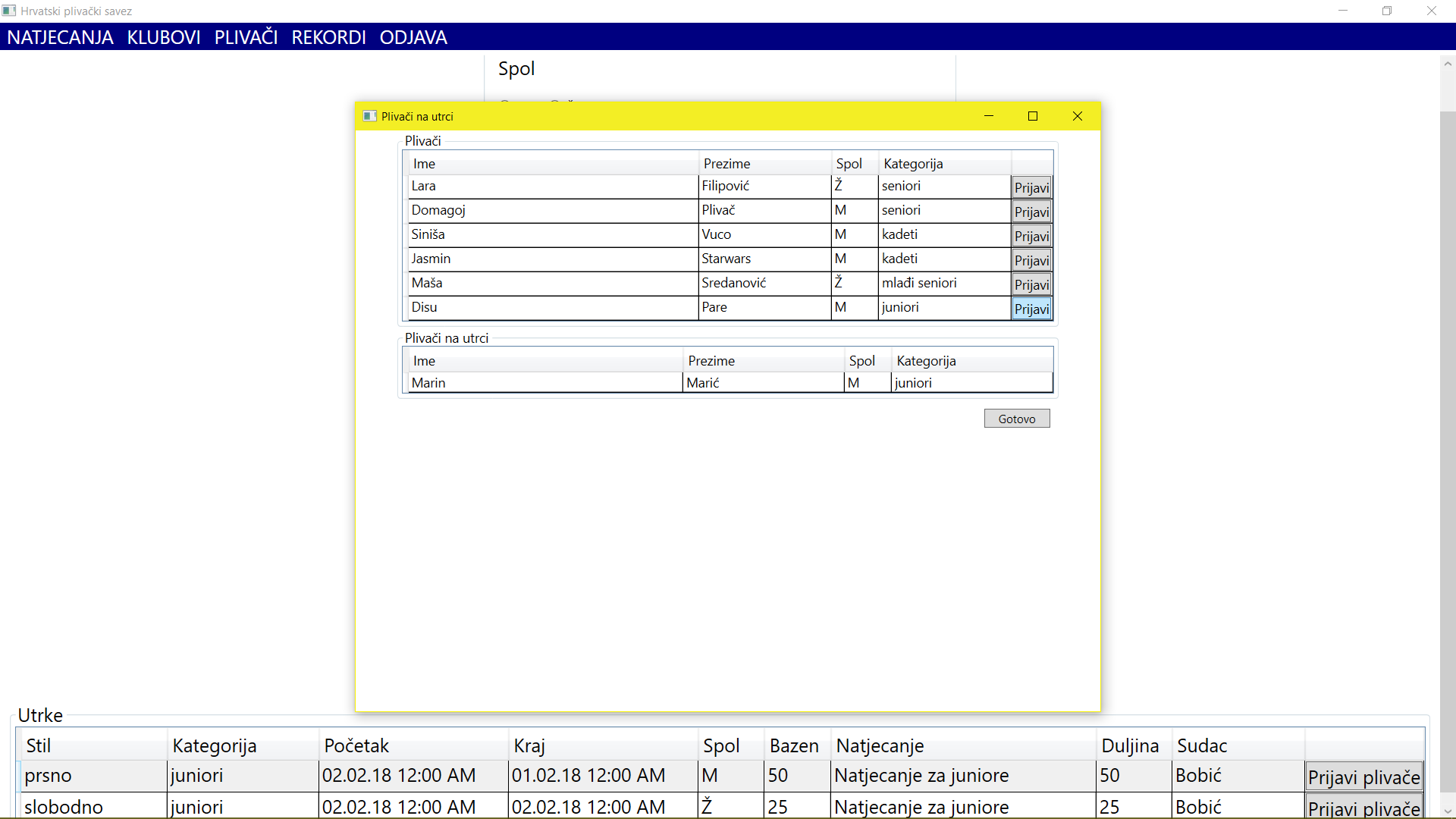
*Slika 5.2. Organizacija natjecanja u trenerovom matičnom gradu*

Ukoliko trener želi organizirati novu utrku za određeno natjecanje, to može učiniti na stranici *Natjecanja* pritiskom na gumb *Uredi utrke*. Time mu se otvara nova stranica na kojoj može pregledati utrke tog natjecanja te dodati novu utrku. Upisom podataka o utrci i pritiskom na gumb *Dodaj* dodaje se nova utrka.



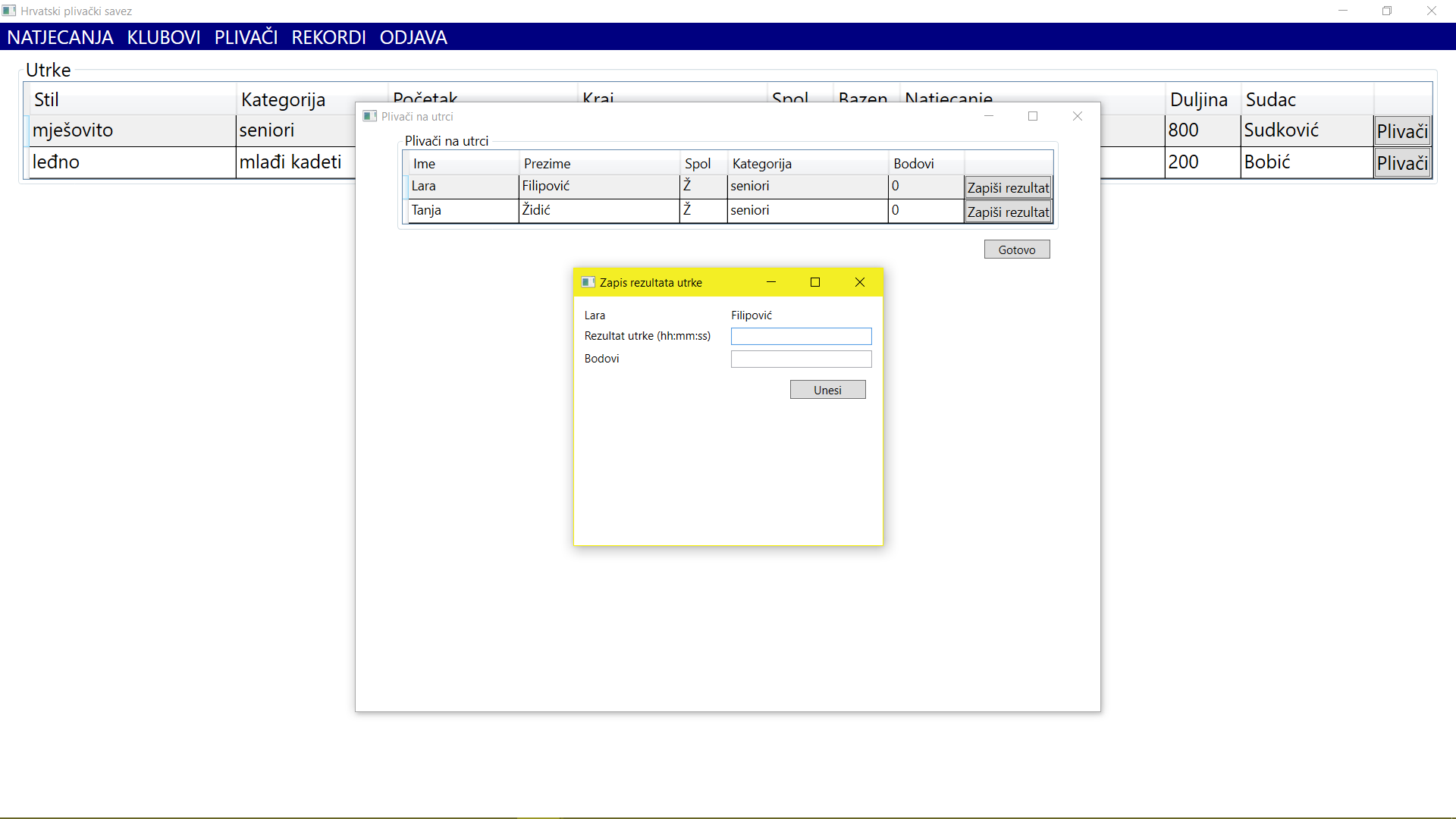
*Slika 5.3. Dodavanje nove utrke natjecanju*

Trener prijavljuje članove svog kluba na utrke klikom na *Prijavi plivače* u retku utrke, zatim klikom na *Prijavi* pored odabranog člana u novootvorenom prozoru (*Slika 5.4.*). Može prijaviti samo plivače koji se svojom kategorijom i spolom poklapaju s propozicijama natjecanja (npr. samo se juniori mogu prijaviti na utrku juniorskog natjecanja).



*Slika 5.4. Prijavljivanje plivača na utrku*

Sucu je omogućeno upisivanje rezultata utrka na kojima je sudio (*Slika 5.5*). Klikom na gumb *Plivači* otvara se prozor s popisom plivača koji su sudjelovali u odabranoj utrci. Sucu je klikom na *Zapiši rezultat* dostupno uređivanje rezultata te bodova odabranog natjecatelja.



*Slika 5.5. Upisivanje rezultata utrke*

# Opis izgrađene web aplikacije

6.1. Organizacija MVC-a

Web aplikacija napravljena je pomoću ASP.NET MVC frameworka. Unutar projekta ove web aplikacije komponente su podijeljene na Model, Controller te View.

Model predstavlja poslovnu logiku samog sustava, odnosno model domene. Controlleri služe tome da prenose modelu poruku kako da se promijeni s obzirom na input korisnika.

View prikazuje trenutno stanje modela, te se svaki View nalazi u podmapi Controllera koji vodi brigu o tom View-u. Za većinu View-a definirani su GET i POST zahtjevi unutar Controllera, odnosno prilikom pristupanja stranici i dohvatu podataka poziva se GET metoda, a prilikom pohrane izmjena koje je korisnik obavio na pogledu poziva se POST metoda unutar pridruženog Controllera.

*Controlleri* s bazom podataka komuniciraju putem *Processora*, koji zatim dalje vode komunikaciju sa *Repositoryem*.

Za izgradnju cjelokupne web aplikacija zaduženo je osam Controllera koji su pobliže objašnjeni u nastavku.

*AccountController.cs* zadužen je za provjeru korisničkog imena i lozinke, koje je korisnik unio preko odgovarajućeg pogleda, sa odgovarajućim podacima u bazi podataka. Dio provjere obavlja se na *backend-u*, kroz *UserProcessor*, te *Controller* kao povratnu vrijednost dobiva informaciju o ispravnosti (true/false).

*ClubController.cs* zadužen je za upravljanje podacima (dohvat iz baze podataka preko

*ClubProcessora* te *ClubRepository-a*) o klubovima općenito, te o klubu za kojeg je zadužen trenutno prijavljeni trener. Unutar njega se također obavlja pridruživanje nekog plivača nekom klubu kao i uklanjanje nekog plivača iz kluba.

*CompetitionController.cs* zadužen je za upravljanje podacima (dohvat iz baze podataka preko *CompetitionProcessora i CompetitionRepositorya*) i slanje tih podataka odgovarajućem pogledu za prikaz informacija o natjecanju. Osim toga ovaj Controller zadužen je i za prikaz utrka unutar pojedinog natjecanja, dodavanje novog natjecanja, uređivanje postojećeg te dodavanje nove utrke tom natjecanju.

*HallController.cs* zadužen je za upravljanje podacima (dohvat iz baze podataka preko *HallProcessora* i *HallRepositorya*) i slanje tih podataka odgovarajućem pogledu za prikaz informacija o dvoranama i bazenima koje ta dvorana posjeduje.

*HomeController.cs* zadužen je za upravljanje podacima (dohvat iz baze podataka preko *CompetitionProcessora* i *CompetitionRepositorya*) i slanje tih podataka početnom pogledu, za prikaz budućih natjecanja.

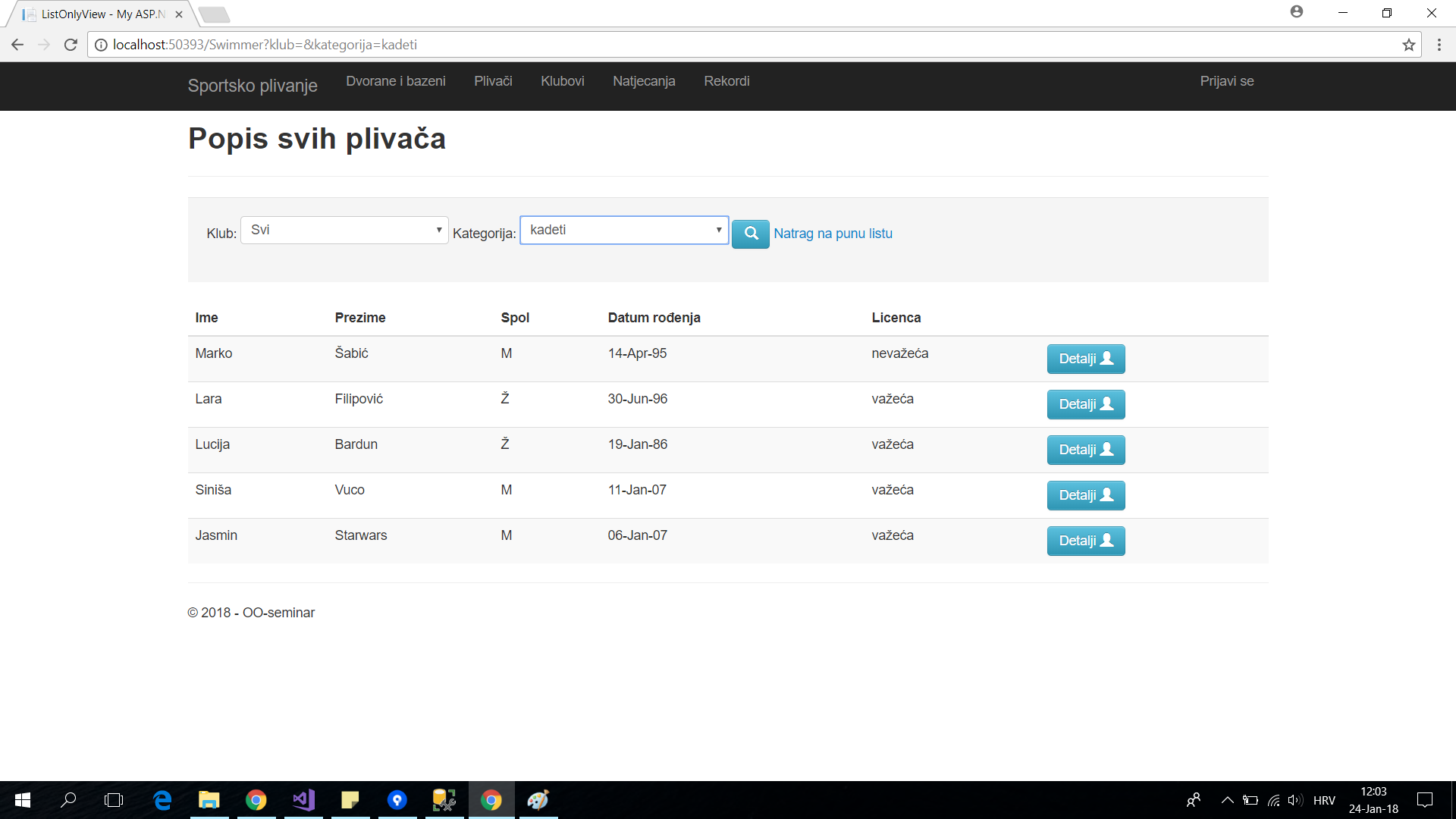
*RaceController.cs* zadužen je za upravljanje podacima (dohvat iz baze podataka preko *RaceProcessora* i *RaceRepositorya*) i slanje tih podataka odgovarajućem pogledu za pregled te uređivanje pojedine utrke.

*RecordsController.cs* zadužen je za upravljanje podacima (dohvat iz baze podataka preko *RecordsProcessora* i *RecordsRepositorya*) i slanje tih podataka odgovarajućem pogledu za pregled svih rekorada u sustavu.

*SwimmerController.cs* zadužen je za upravljanje podacima (dohvat i izmjena iz baze podataka preko *SwimmerProcessora i SwimmerRepositorya*) i slanje tih podataka odgovarajućem pogledu za prikaz svih plivača koji su trenutno u sustavu, kao i onih koji odgovaraju nekom od kriterija. Osim toga ovaj Controller sadrži POST metode za dodavanje novog plivača kao i za uređivanje informacija o plivaču.

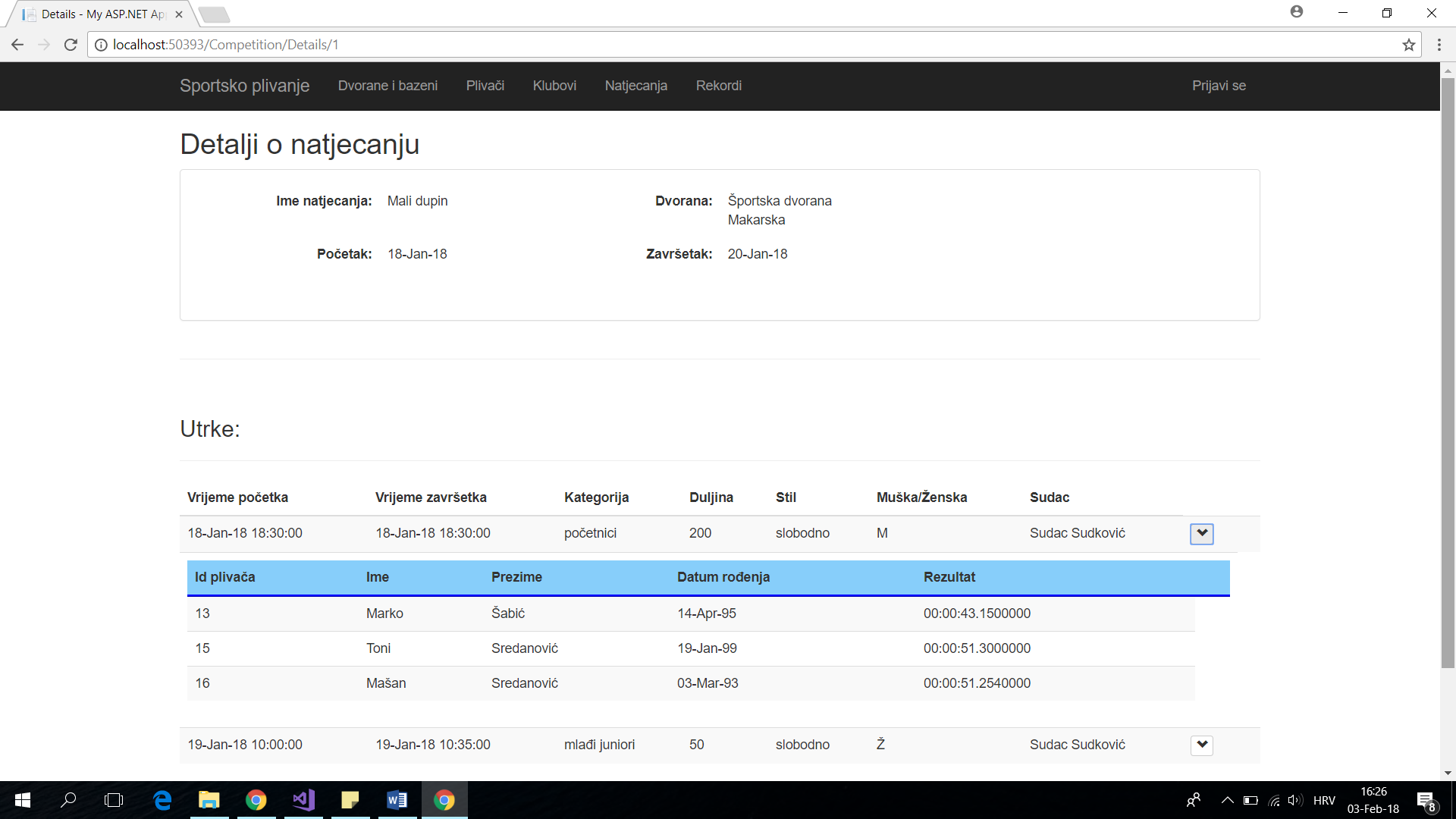
6.2. Opis korisničkog sučelja

Web aplikaciji mogu pristupiti neregistrirani i registrirani korisnici. Neregistriranim korisnicima omogućeno je pregledavanje i pretraživanje plivača po klubu i kategoriji kojoj pripada, kao i detaljni pregled plivača po sezonama i utrkama na kojima je sudjelovao.



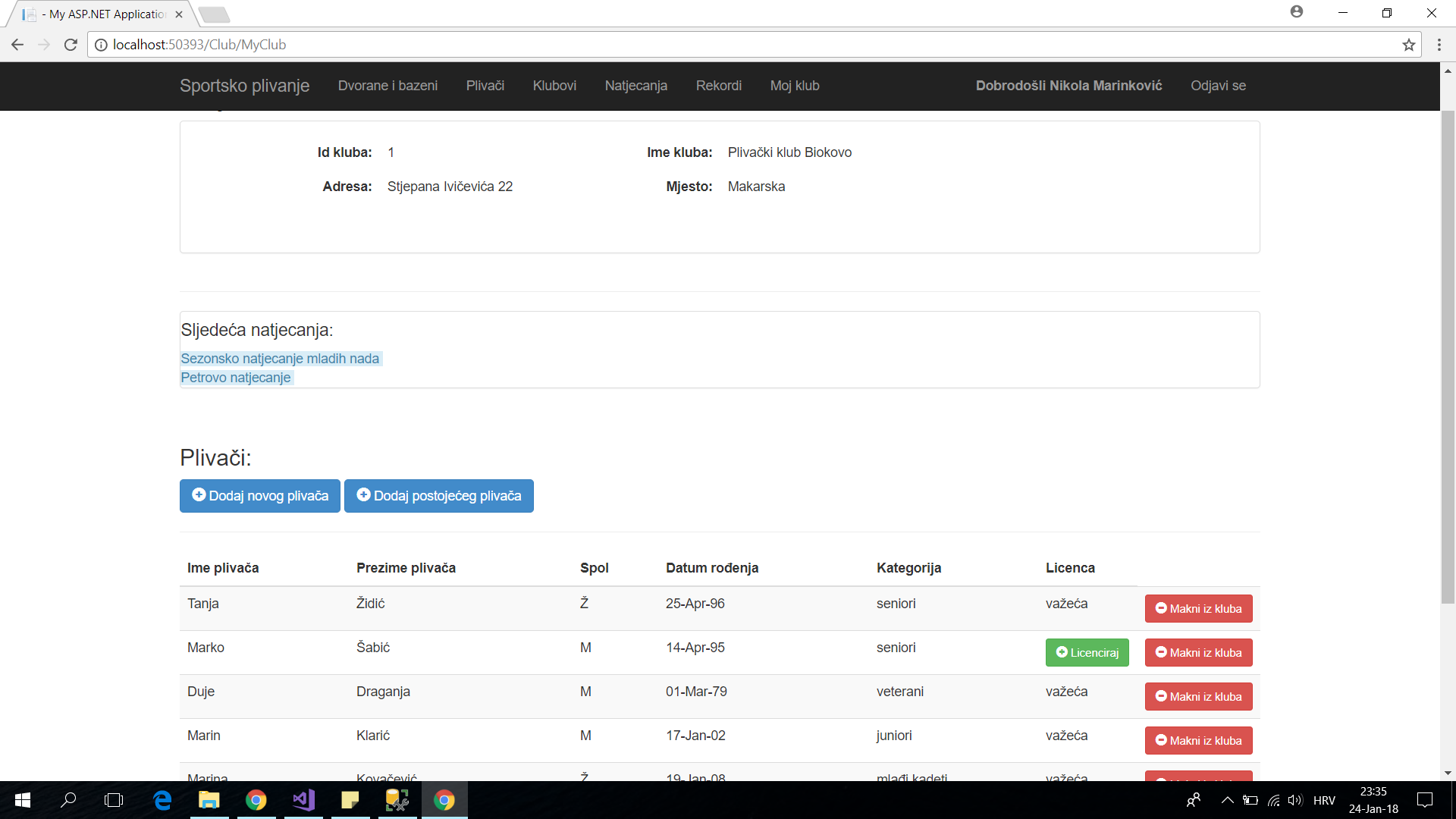
*Slika 6.1. Popis svih plivača u kategoriji kadeti*

Također je omogućen pregled klubova i natjecanja, te utrka na pojedinim natjecanjima. Korisnik je osim toga može i pregledati sve hrvatske rekorde po kategorijama.



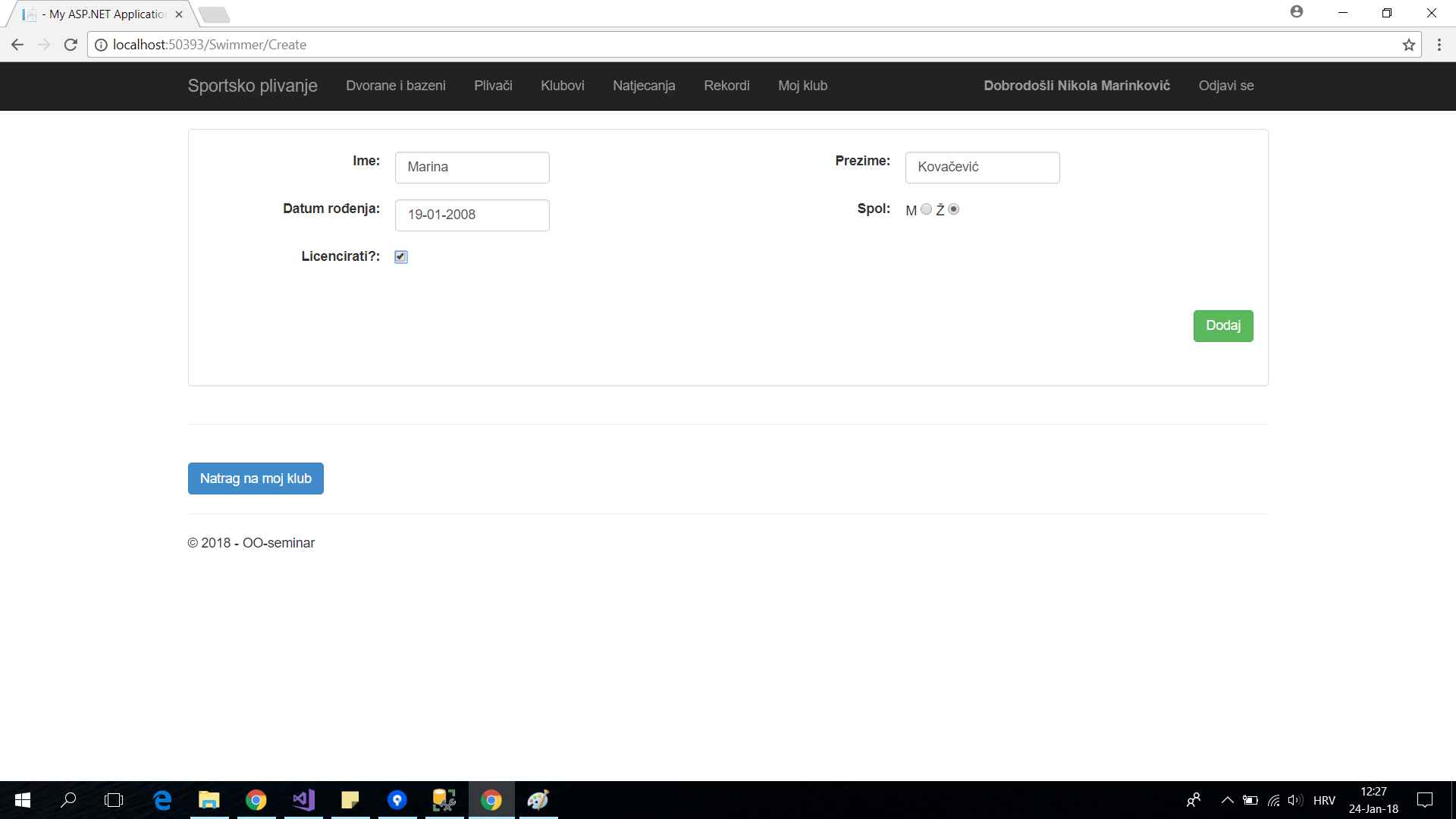
*Slika 6.2. Detalji o natjecanju*

Registrirani korisnici mogu biti treneri ili suci. Svakom treneru omogućen je pristup novoj stranici s podacima o njegovom klubu. Tu mu je omogućeno dodavanje novih ili postojećih plivača u njegov klub, uklanjanje plivača iz kluba, te dodijeljivanje licence plivaču. Također su mu vidljiva predstojeća natjecanja na kojima se natječu plivači iz njegovog kluba.



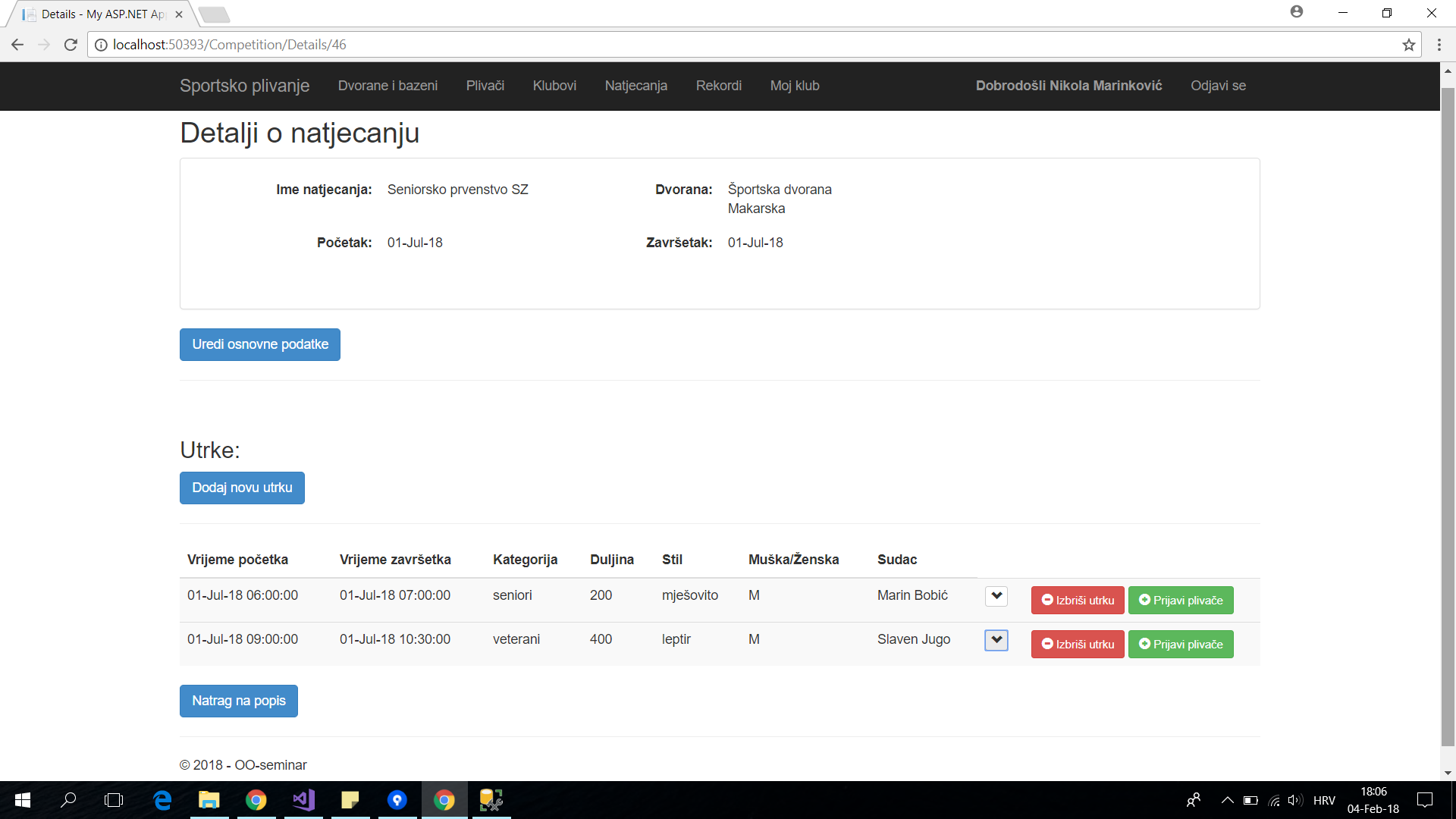
*Slika 6.3. Detalji o trenerovom klubu*

Kod dodavanja novog plivača trener mora unijeti sve potrebne podatke o plivaču, a ako želi dodati plivača koji je već registriran u sustavu može izabrati iz liste plivača koji nisu već u nekom drugom klubu. Ukoliko trener odabere opciju dodavanja novog plivača ima mogućnosti davanja licence plivaču, te na taj način automatski dotičnom plivaču u sustavu dodijeli licencu za ovu sezonu.



*Slika 6.4. Dodavanje novog plivača*

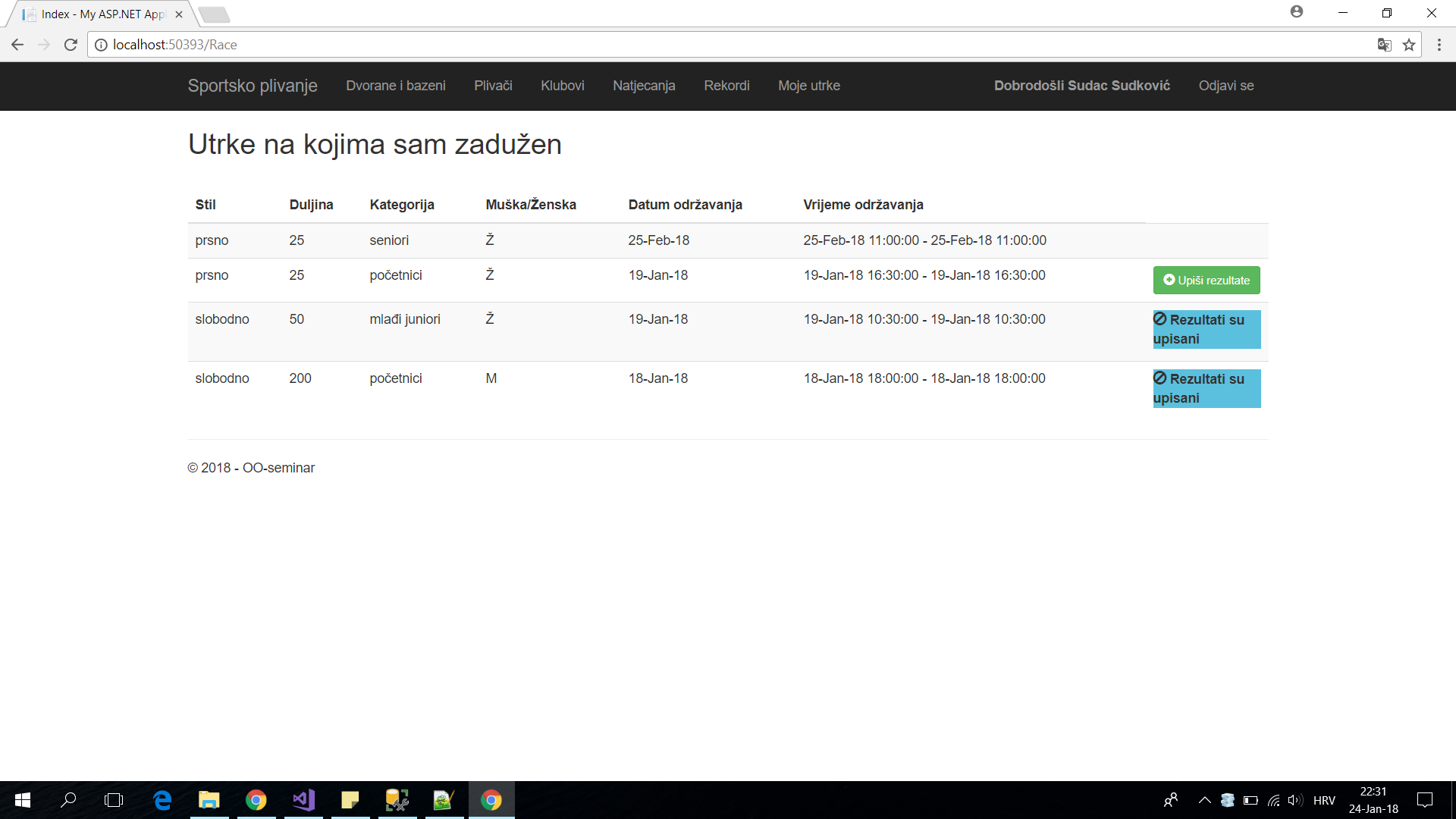
Osim toga trener može dodavati nova natjecanja, i za ta natjecanja može dodavati nove utrke. Za sva natjecanja trener može prijaviti svoje plivače za neku od utrka, ako plivač odgovara toj kategoriji i ako isti već nije prijavljen za tu trku.



*Slika 6.5. Uređivanje natjecanja*

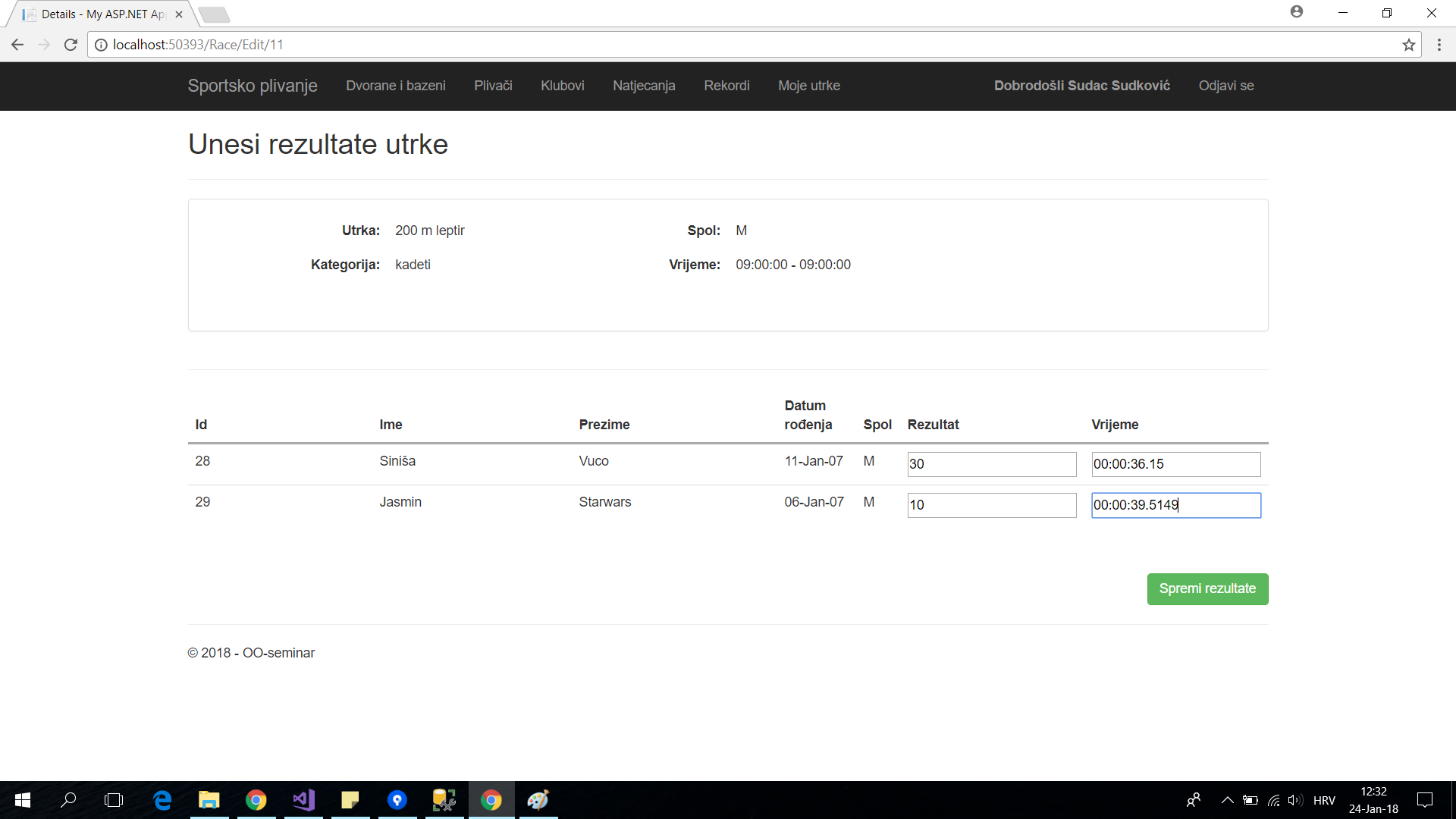
Kod dodavanja novog natjecanja trener kao dvoranu u kojoj se održava natjecanje može odabrati samo dvoranu koja je u njegovom gradu. Kada doda novo natjecanje za isto može dodati i sve utrke koje će se održavati, pri tome se pazi da vrijeme održavanja utrke bude unutar vremena kada se održava natjecanje.

Sudac s druge strane ima uvid u sve trke za koje je zadužen. Prilikom odabira opcije “Moje trke” u glavnom izborniku sucu se prikazuju sve trke za koje je bio zadužen. Za svaku trku je navedena opcija “Upiši rezultate” ukoliko je trka završena a rezultati još nisu upisani.



*Slika 6.6. Pregled utrka za koje je sudac zadužen*

Nakon odabira opcije “Unesi rezultate” treneru se otvara novi View u kojem može, za svakog plivača pojedinačno, unijeti njegovo vrijeme te rezultat koji je ostvario na toj utrci.



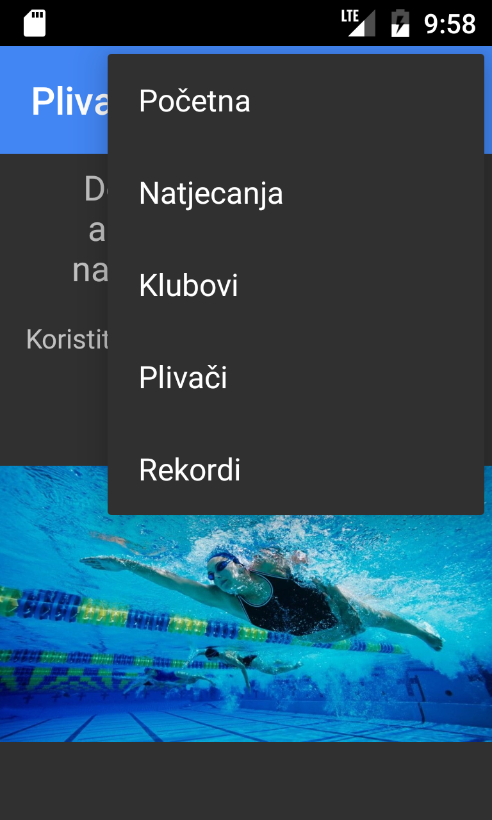
*Slika 6.7. Unos rezultata utrke*

# Opis izgrađene mobilne aplikacije

## Opis

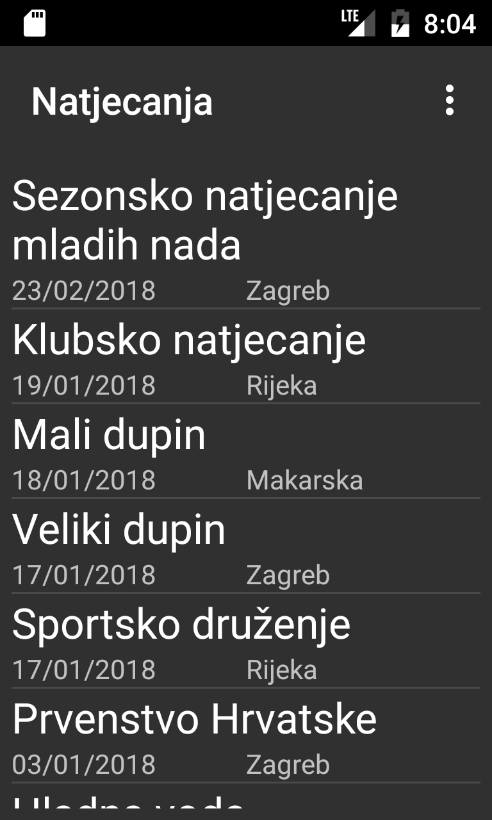
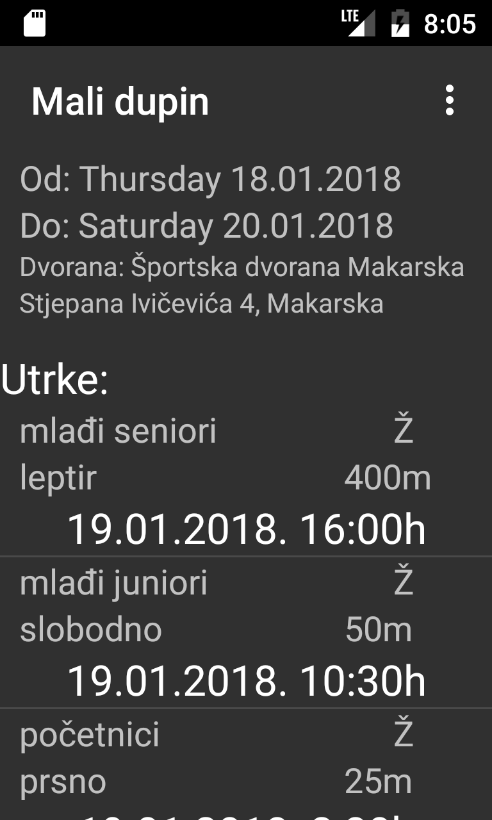
Kod mobilne aplikacije primarni fokus je na prezentaciji podataka svakodnevnom korisniku. Svim korisnicima omogućit će se pregled informacija o bazenima, dvoranama, natjecanjima, utrkama, klubovima, plivačima, rezultatima i rekordima u sklopu plivačkog saveza.

Padajući izbornik za odabir pregleda informacija nalazi se u gorenjm desnom kutu aplikacije te nudi na izbor pregledavanje informacija o natjecanjima, o klubovima, o plivačima i o rekordima (4 glavna skupa informacija).



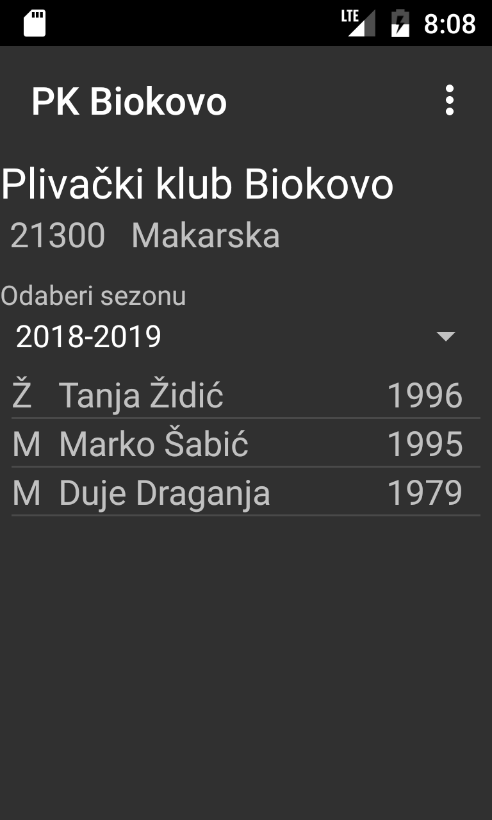
|  |  |
| --- | --- |
| Slika 7.1 - Početna stranica | Slika 7.2 - Glavni izbornik |

Kod odabira informacija o natjecanju dobivaju se izlistana natjecanja prema vremenu održavanja uz mogućnost pregleda detaljnijih informacija o svakom pojedinom natjecanju.

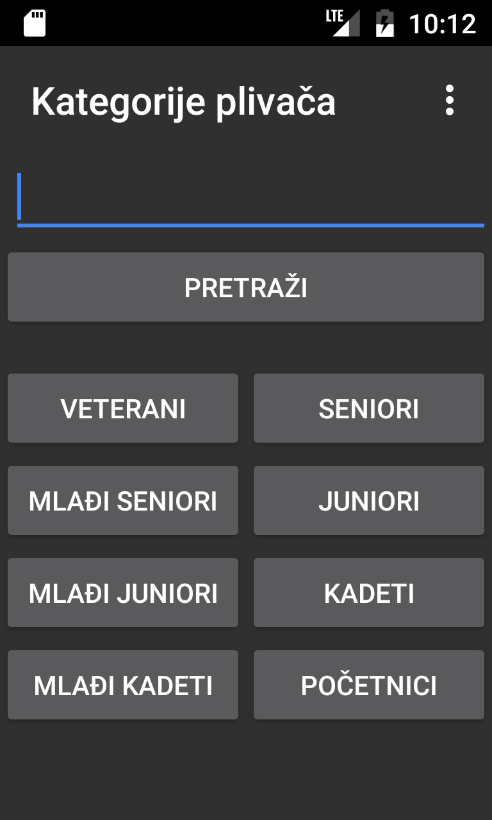
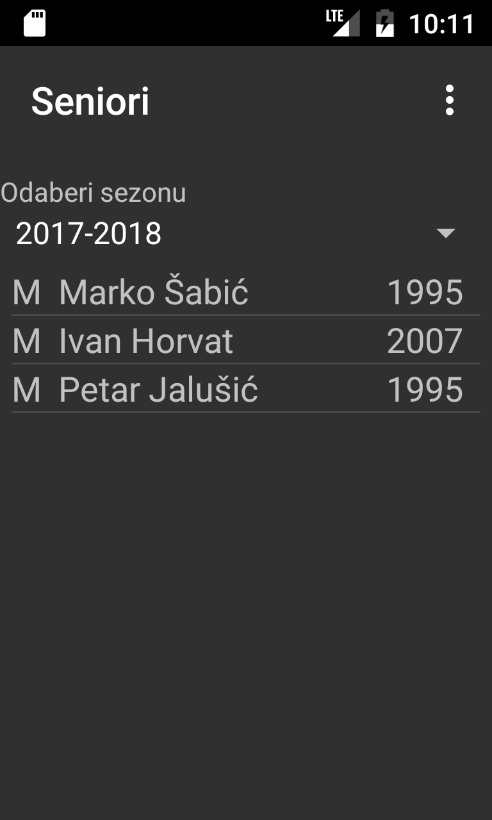
 

|  |  |
| --- | --- |
| Slika 7.3 - Pregled svih natjecanja | Slika 7.4 - Pregled jednog natjecanja |

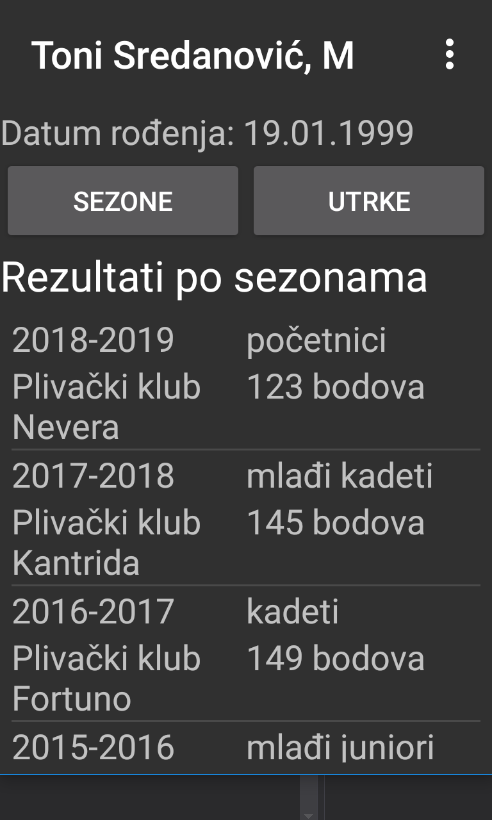
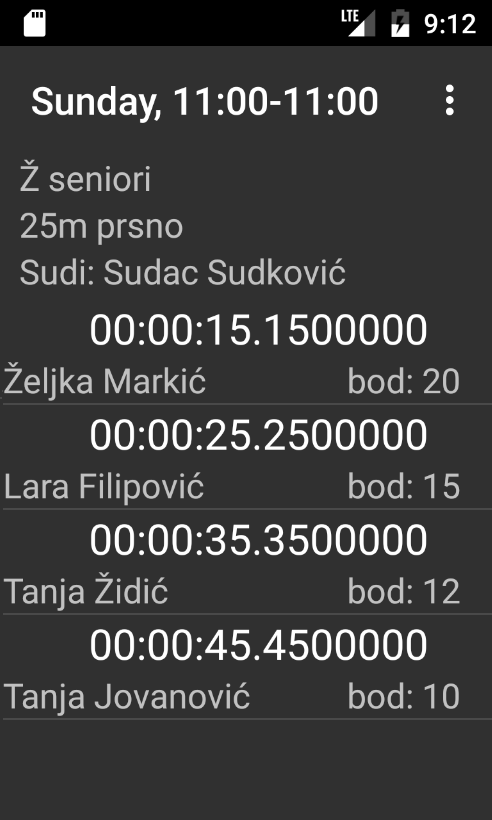
Na sličan način odvija se i prikaz informacija o klubovima, rekordima te igračima, uz nadodane specifične funkcionalnosti kao što je pretraga igrača po imenu.

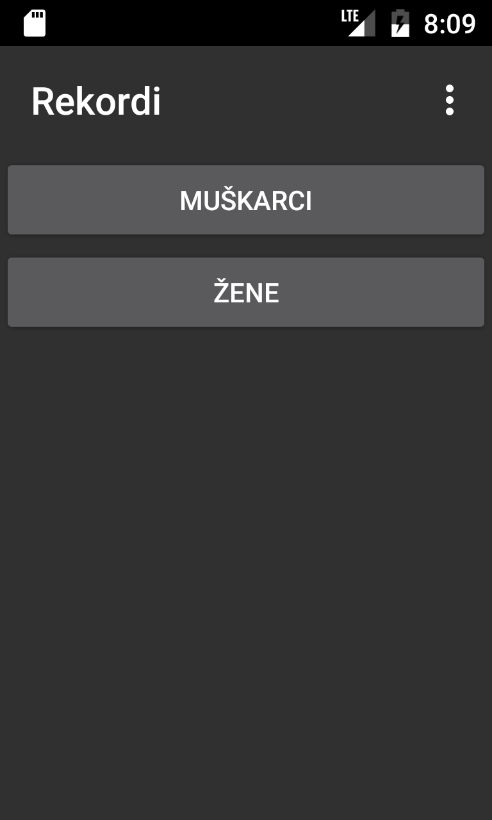
|  |  |
| --- | --- |
| Slika 7.5 - Pregled svih klubova | Slika 7.6 - Pregled jednog kluba |

|  |  |
| --- | --- |
| Slika 7.7 - Pretraga igrača | Slika 7.8 - Pregled određenih igrača |

|  |  |
| --- | --- |
| Slika 7.9 - Pregled jednog igrača | Slika 7.10 - Pregled rezultata utrke |

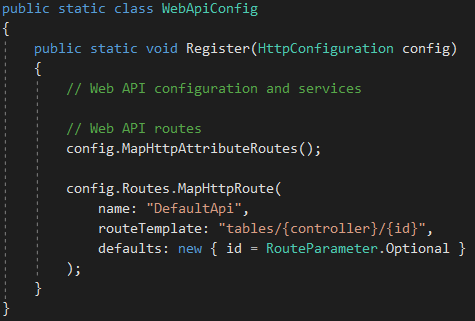
|  |  |
| --- | --- |
| Slika 7.11 - Odabir rekorda | Slika 7.12 - Pregled odabranih rekorda |

## 

## Web API

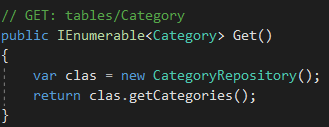
U sklopu mobilne aplikacije su definirani API-ji preko kojih mobilna aplikacija komunicira s bazom podataka (dohvaćanje, izmjena, brisanje). Izgradnja je napravljena pomoću ASP.NET Web API 2 frameworka prema REST arhitekturnim načelima. Za svaku funkciju koju korisnik može obaviti putem aplikacije definirano je jedno sučelje. Svi podaci koji se šalju i primaju su u obliku JSON objekta. Aplikacija je postavljena na server kojem se može pristupiti preko sljedeće poveznice: <https://oosemmobapp2.azurewebsites.net/>.

Na slici 7.13 prikazana je konfiguracija WEB API-ja. Bitno je za primijetiti da su adrese postavljene na /tables umjesto na /api zbog toga što nam to omogućuje lakše dohvaćanje podatke preko mobilne aplikacije.



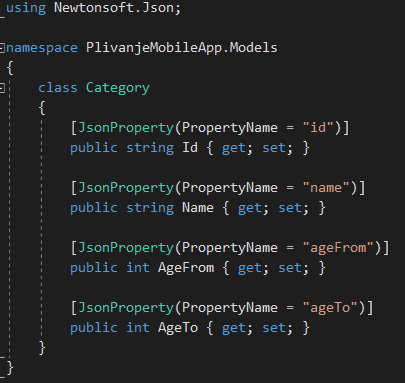
*Slika 7.13 Konfiguracija ruta*

Nakon toga kontroler za dohvat svih kategorija izgleda kao što je prikazano na slici 7.14, odnosno ruta za dohvat je /tables/Category.



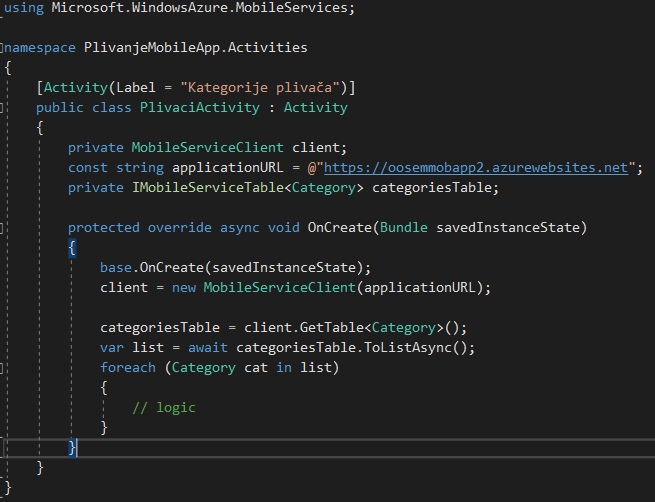
*Slika 7.14 Dohvat svih kategorija*

Tada u mobilnoj aplikaciji koristimo sljedeću klasu za model kategorije kao što je prikazano na slici 7.15.



*Slika 7.15 Model kategorije*

Primjer activityja koji dohvaća sve kategorije prikazan je na slici 7.16. Korišteni su mobile services koji se automatski spajaju na /tables rutu našeg WEB API-ja te smo zbog toga odabrali takvu rutu kao što je bilo prikazano na slici 7.13. S obzirom da koristimo klasu Category, linija koda: *client.GetTable<Category>* će se spojiti na /tables/Category [rutu](https://oosemmobapp2.azurewebsites.net/tables/Category), dohvatiti podatke u json formatu i automatski obaviti mapiranje na klasu Category. Nakon toga dohvaćamo sve kategorije metodom *.ToListAsync()*.



*Slika 7.16 Primjer dohvata kategorija*

U nastavku slijede definicija sučelja našeg WEB API-ja.

**Category** - operacije za tablicu Category

GET /tables/Category - Traži sve zapise u tablici Category

POST /tables/Category - Umeće zapis u tablicu Category

DELETE /tables/Category/{id} - Briše određeni zapis u tablici Category

GET /tables/Category/{id} - Traži određeni zapis u tablici Category

PATCH /tables/Category/{id} - Ažurira određeni zapis u tablici Category

POST /tables/Category/{id} - Vraća prethodno obrisani zapis u tablici Category

**Club** - operacije za tablicu Club

GET /tables/Club - Traži sve zapise u tablici Club

POST /tables/Club - Umeće zapis u tablicu Club

DELETE /tables/Club/{id} - Briše određeni zapis u tablici Club

GET /tables/Club/{id} - Traži određeni zapis u tablici Club

PATCH /tables/Club/{id} - Ažurira određeni zapis u tablici Club

POST /tables/Club/{id} - Vraća prethodno obrisani zapis u tablici Club

**Competition** - operacije za tablicu Competition

GET /tables/Competition - Traži sve zapise u tablici Competition

POST /tables/Competition - Umeće zapis u tablicu Competition

DELETE /tables/Competition/{id} - Briše određeni zapis u tablici Competition

GET /tables/Competition/{id} - Traži određeni zapis u tablici Competition

PATCH /tables/Competition/{id} - Ažurira određeni zapis u tablici Competition

POST /tables/Competition/{id} - Vraća prethodno obrisani zapis u tablici Competition

**Race** - operacije za tablicu Race

GET /tables/Race - Traži sve zapise u tablici Race

POST /tables/Race - Umeće zapis u tablicu Race

DELETE /tables/Race/{id} - Briše određeni zapis u tablici Race

GET /tables/Race/{id} - Traži određeni zapis u tablici Race

PATCH /tables/Race/{id} - Ažurira određeni zapis u tablici Race

POST /tables/Race/{id} - Vraća prethodno obrisani zapis u tablici Race

**Record** - operacije za tablicu Record

GET /tables/Record - Traži sve zapise u tablici Record

POST /tables/Record - Umeće zapis u tablicu Record

DELETE /tables/Record/{id} - Briše određeni zapis u tablici Record

GET /tables/Record/{id} - Traži određeni zapis u tablici Record

PATCH /tables/Record/{id} - Ažurira određeni zapis u tablici Record

POST /tables/Record/{id} - Vraća prethodno obrisani zapis u tablici Record

**Season** - operacije za tablicu Season

GET /tables/Season - Traži sve zapise u tablici Season

POST /tables/Season - Umeće zapis u tablicu Season

DELETE /tables/Season/{id} - Briše određeni zapis u tablici Season

GET /tables/Season/{id} - Traži određeni zapis u tablici Season

PATCH /tables/Season/{id} - Ažurira određeni zapis u tablici Season

POST /tables/Season/{id} - Vraća prethodno obrisani zapis u tablici Season

**Swimmer** - operacije za tablicu Swimmer

GET /tables/Swimmer - Traži sve zapise u tablici Swimmer

POST /tables/Swimmer - Umeće zapis u tablicu Swimmer

DELETE /tables/Swimmer/{id} - Briše određeni zapis u tablici Swimmer

GET /tables/Swimmer/{id} - Traži određeni zapis u tablici Swimmer

PATCH /tables/Swimmer/{id} - Ažurira određeni zapis u tablici Swimmer

POST /tables/Swimmer/{id} - Vraća prethodno obrisani zapis u tablici Swimmer

**SwimmerRace** - operacije za tablicu SwimmerRace

GET /tables/SwimmerRace - Traži sve zapise u tablici SwimmerRace

POST /tables/SwimmerRace - Umeće zapis u tablicu SwimmerRace

DELETE /tables/SwimmerRace/{id} - Briše određeni zapis u tablici SwimmerRace

GET /tables/SwimmerRace/{id} - Traži određeni zapis u tablici SwimmerRace

PATCH /tables/SwimmerRace/{id} - Ažurira određeni zapis u tablici SwimmerRace

POST /tables/SwimmerRace/{id} - Vraća prethodno obrisani zapis u tablici SwimmerRace

**ClubView** - operacije za pogled ClubView

GET /tables/ClubView - Traži sve zapise u pogled ClubView

**CompetitionView** - operacije za pogled CompetitionView

GET /tables/CompetitionView - Traži sve zapise u pogledu CompetitionView

**RaceView** - operacije za pogled RaceView

GET /tables/RaceView - Traži sve zapise u pogledu RaceView

**SwimmerRaceView** - operacije za pogled SwimmerRaceView

GET /tables/SwimmerRaceView - Traži sve zapise u pogledu SwimmerRaceView

**SwimmersView** - operacije za pogled SwimmersView

GET /tables/SwimmersView - Traži sve zapise u pogledu SwimmersView