# Projektni zadatak iz Internet Tehnologija

# E-Kladionica

Git Hub Link:

<https://github.com/Banega00/ITEH-Projekat>

## Branislav Stojanović 79/18

## David Čuturilo 185/18

## Predrag Tasić 268/18

Beograd, Maj 2022

# Korisnički zahtev

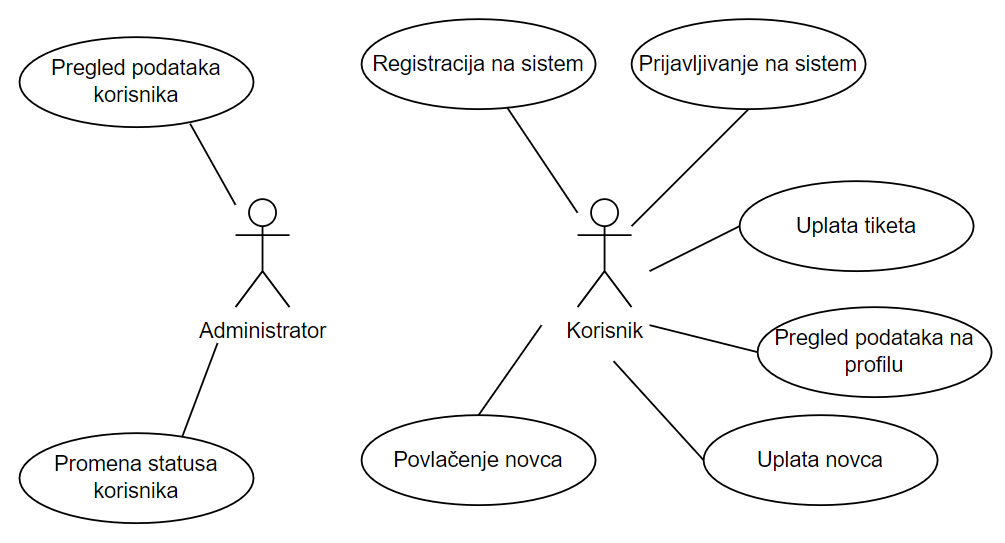
Better je web sajt koji predstavlja e-kladionicu koja u ponudi ima klađenje na veliki broj sportskih događaja.

Aplikacija Better ima za cilj da ljubiteljima sporta i sportske prognoze pruži zadovoljstvo da okušaju svoju sreći prognozirajući rezultate sportskih mečeva, a u slučaju da u tome budu uspešni bivaju finansijski nagrađeni u skladu sa rezultatima koje su pogodili i novcem koji su uložili u tu opkladu.

Korisnik ulaskom na web sajt dobija prikaz mečeva, grupisan prema državama i ligama u kojima se mečevi odvijaju. On u skladu sa svojom intuicijom bira rezultate za koje veruje da će biti ishod mečeva, pravi tiket, bira koliko će novca uložiti na tiket i zatim uplaćuje tiket.

Korisnik na svom profilu ima uvid u tikete koje je odigrao kao i rezultate koje je na njima prognozirao, novac koji je uplatio za taj tiket, kao i potencijalni dobitak u slučaju da je pogodio sve rezultate na tiketu.

# Slučajevi korišćenja



Slika 1 - Dijagram slučajeva korišćenja

## SK1 – Prijavljivanje na sistem

**Naziv SK: Prijavljivanje na sistem**

**Aktori SK:** Korisnik

**Učesnici SK:** Korisnik i sistem

**Preduslov:** Sistem je pokrenut, Korisnik je pristupio stranici za prijavaljivanje na sistem.

**Osnovni scenario SK:**

1. Korisnik **unosi** kredencijale (korisničko ime i šifru). (**APUSO**)

2. Korisnik **proverava** ispravnost unetih kredencijala. (**ANSO**)

3. Korisnik **poziva** sistem da ga prijavi. (**APSO**)

4. Sistem **prijavaljuje** korisnika. (**SO**)

5. Sistem **prebacuje** korisnika na početnu stranicu aplikacije. (**IA**)

**Alternativni scenariji:**

5.1. Ukoliko korisnik sa unetim korisničkim imenom ne postoji u bazi sistema biće prikazana poruka “Pogrešno korisničko ime”.(**IA**)

5.2 Ukoliko korisničko ime postoji ali se uneta šifra ne podudara sa šifrom iz sistema sistem prikazuje poruku “Pogrešna šifra”.(**IA**)

5.3 Ukoliko je nalog korisnika blokiran sistem prikazuje poruku “Nalog je blokiran”.(**IA**)

## SK2 – Uplata tiketa

**Naziv SK: Uplata tiketa**

**Aktori SK:** Korisnik

**Učesnici SK:** Korisnik i sistem

**Preduslov:** Sistem je pokrenut, Korisnik je prijavljen na sistem i nalazi se na početnoj stranici na kojoj su izlistani sportski mečevi.

**Osnovni scenario SK:**

1. Korisnik bira prognoze za sportske događaje. (**APUSO**)

2. Korisnik **proverava** ispravnost izabranih prognoza. (**ANSO**)

3. Korisnik **unosi** svotu novca koju želi da uplati na tiket. (**APUSO**)

4. Korisnik **poziva** sistem da uplati tiket. (**APSO**)

5. Sistem **pamti** tiketkorisnika. (**SO**)

6. Sistem **prikazuje** poruku “Tiket je uspešno uplaćen”. (**IA**)

**Alternativni scenariji:**

6.1. Ukoliko korisnik nema dovoljno sredstava na računu sistem prikazuju poruku “Nemate dovoljno sredstava”. (**IA**)

## SK3 – Promena satusa korisnika

**Naziv SK: Promena statusa**

**Aktori SK:** Administrator

**Učesnici SK:** Administrator i sistem

**Preduslov:** Sistem je pokrenut, Administrator je prijavljen na sistem.

**Osnovni scenario SK:**

1. Administrator **bira** zaposlenog čiji status želi da promeni. (**APUSO**)

2. Administrator **poziva** sistem da promeni status naloga korisnika, ukoliko zaposleni blokiran, biće odblokiran i ako odblokiran biće blokiran. (**APSO**)

3. Sistem **menja** status naloga korisnika. (**SO**)

4. Sistem **prikazuje** poruku “Zaposleni je blokiran/odblokiran”. (**IA**)

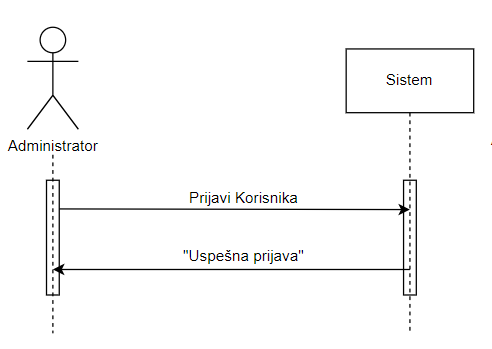
# Dijagrami sekvenci

## DS1 – Prijavljivanje na sistem

**Osnovni scenario SK:**

1. Korisnik **poziva** sistem da ga prijavi. (**APSO**)

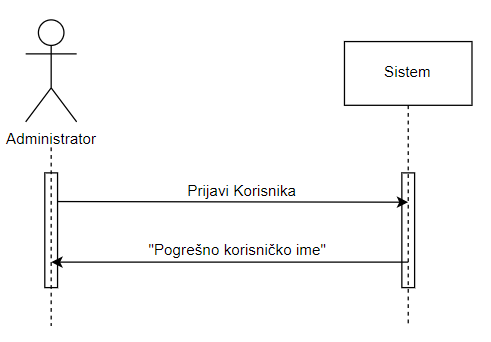
2. Sistem **prebacuje** korisnika na početnu stranicu aplikacije. (**IA**)



**Alternativni scenario SK:**

1. Korisnik **poziva** sistem da ga prijavi. (**APSO**)

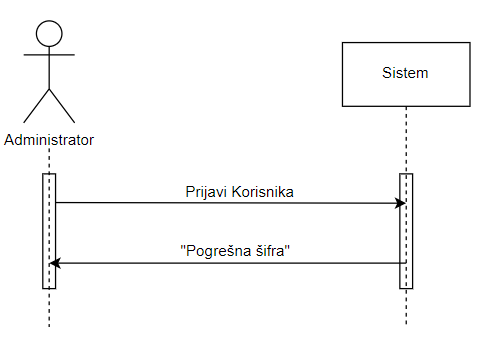
2. Ukoliko korisnik sa unetim korisničkim imenom ne postoji u bazi sistema biće prikazana poruka “Pogrešno korisničko ime”.(**IA**)



**Alternativni scenario SK:**

1. Korisnik **poziva** sistem da ga prijavi. (**APSO**)

2. Ukoliko korisničko ime postoji ali se uneta šifra ne podudara sa šifrom iz sistema sistem prikazuje poruku “Pogrešna šifra”.(**IA**)



**Alternativni scenario SK:**

1. Korisnik **poziva** sistem da ga prijavi. (**APSO**)

2. Ukoliko je nalog korisnika blokiran sistem prikazuje poruku “Nalog je blokiran”.(**IA**)

## 

## DS2 – Uplata tiketa

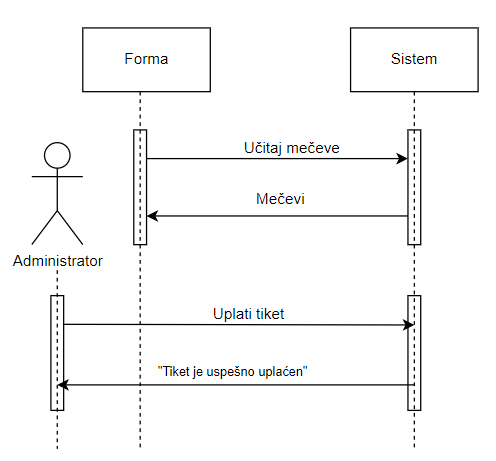
**Osnovni scenario SK:**

1. Forma **poziva** sistem da učita sportske mečeve. (**APSO**)

2. Sistem **vraća** sportske mečeve. (**IA**)

3. Korisnik **poziva** sistem da uplati tiket. (**APSO**)

4. Sistem **prikazuje** poruku “Tiket je uspešno uplaćen”. (**IA**)



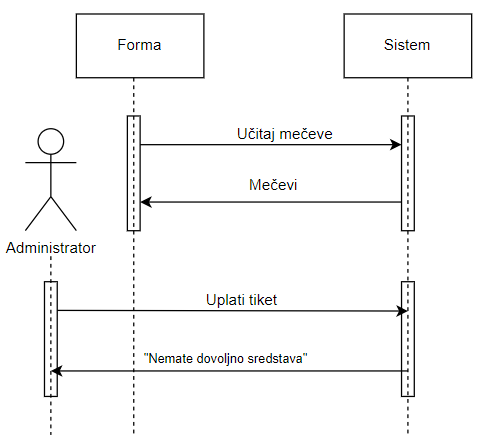
**Alternativni scenario SK:**

1. Forma **poziva** sistem da učita sportske mečeve. (**APSO**)

2. Sistem **vraća** sportske mečeve. (**IA**)

3. Korisnik **poziva** sistem da uplati tiket. (**APSO**)

4. Ukoliko korisnik nema dovoljno sredstava na računu sistem prikazuju poruku “Nemate dovoljno sredstava”. (**IA**)



## DS3 – Promena statusa korisnika

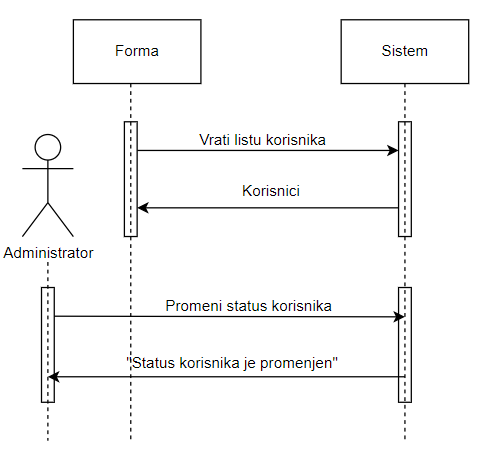
**Osnovni scenario SK:**

1. Forma **poziva** sistem da vrati listu svih korisnika. (**APSO**)

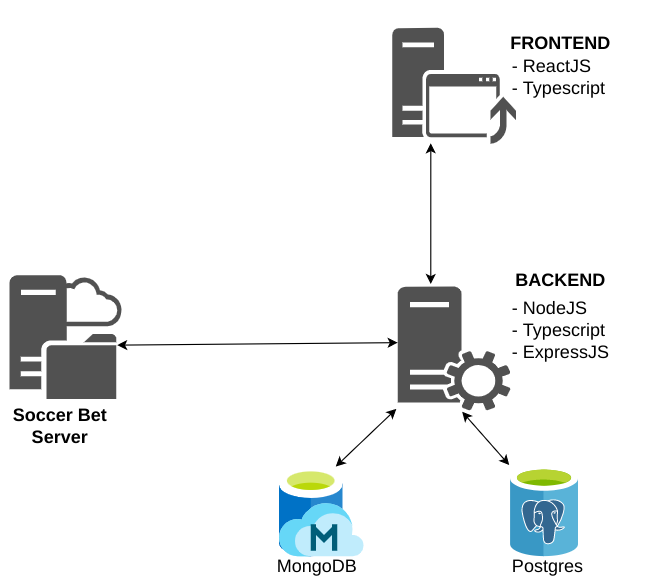
2. Sistem **vraća** listu svih zaposlenih. (**IA**)

3. Administrator **poziva** sistem da promeni status korisnika (Blokira ako je korisnik odblokiran i odblokira ako je korisnik odblokiran) . (**APSO**)

4. Sistem **prikazuje** poruku “Zaposleni je blokiran/odblokiran”. (**IA**)



# Arhitektura aplikacije



Slika 2 - Dijagram komponenti - Arhitektura aplikacije

Sistem se sastoji od dva glavna dela. Backend server je implementiran u NodeJS-u koji predstavlja runtime okruženje za izvršavanje Javasripta, s tim što je u ovom slučaju korišćen Typescript kao superset Javascript jezika kako bi se obezbedila veća sigurnost i robusnost prilikom razvoja sistema. Na Backend serveru korišćen je ExpressJS kao framework za implementaciju REST Web servisa.

Backend server koristi MongoDB za čuvanje korisničkih sesija korisnika koji pristupaju sistemu. Svi ostali važni entiteti se čuvaju u Postgres bazi.

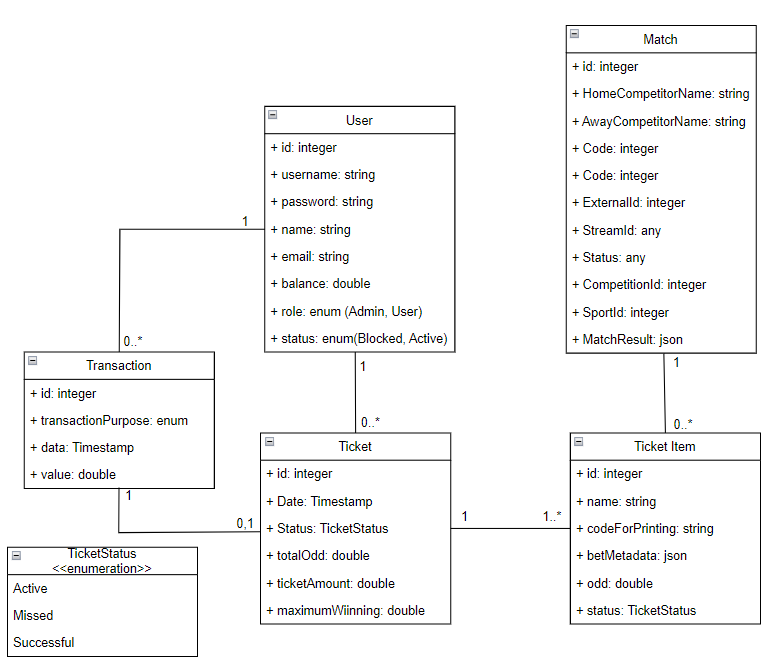
Druga važna komponenta sistema je Frontend aplikacija napisana korišćenjem ReactJS frameworka za Javascript, s tim što je i u ovom slučaju korišćen Typescript kao sredstvo za razvoj aplikacije. Frontend aplikacija koristi React Material biblioteku sa gotovim UI komponentama kako bi se obezbedilo standardizovano i prijatno korisničko iskustvo.

Za pribavljanje podataka o sportskim mečevima i kvotama sistem komunicira sa serverom Soccer bet kladionice koji se nalazi izvan našeg sistema.

Sve komponente komuniciraju slanjem zahteva i odgovora preko HTTP protokola.

# Model Podataka

Dijagram klasa



# Tehnologije

Sistem se sastoji od dva glavna dela: Backend Server i Frontend aplikacija

Backend server je implementiran u NodeJS-u za koji predstavlja runtime okruženje izvršavanje Javasripta, s tim što je u ovom slučaju korišćen Typescript kao superset Javascript jezika kako bi se obezbedila veća sigurnost i robusnost prilikom razvoja sistema. Na Backend serveru korišćen je ExpressJS kao framework za implementaciju REST Web servisa. Radi lakšeg rada sa bazom korišćen je TypeORM kao objektno relacioni maper zbog njegove dobre kompatibilnosti sa Typescript jezikom. Upravljanje korisničkim sesijama realizovano je uz pomoć express-session biblioteke koja je kompatibilna sa ExpressJS frameworkom. Backend server koristi MongoDB za čuvanje korisničkih sesija korisnika koji pristupaju sistemu. Svi ostali važni entiteti se čuvaju u Postgres bazi. Enkripcija i hashovanje realizovano je uz pomoć **bcrypt** biblioteke za NodeJS. Za slanje HTTP zahteva korišćena je **node-fetch** biblioteka.

Druga važna komponenta sistema je Frontend aplikacija napisana korišćenjem ReactJS frameworka za Javascript, s tim što je i u ovom slučaju korišćen Typescript kao sredstvo za razvoj aplikacije. Frontend aplikacija koristi React Material biblioteku sa gotovim UI komponentama kako bi se obezbedilo standardizovano i prijatno korisničko iskustvo. Pored React Material-a neke komponente posebno su stilizovane uz pomoć SCSS-a, koji se kompajlira u običan CSS koji browser razume.

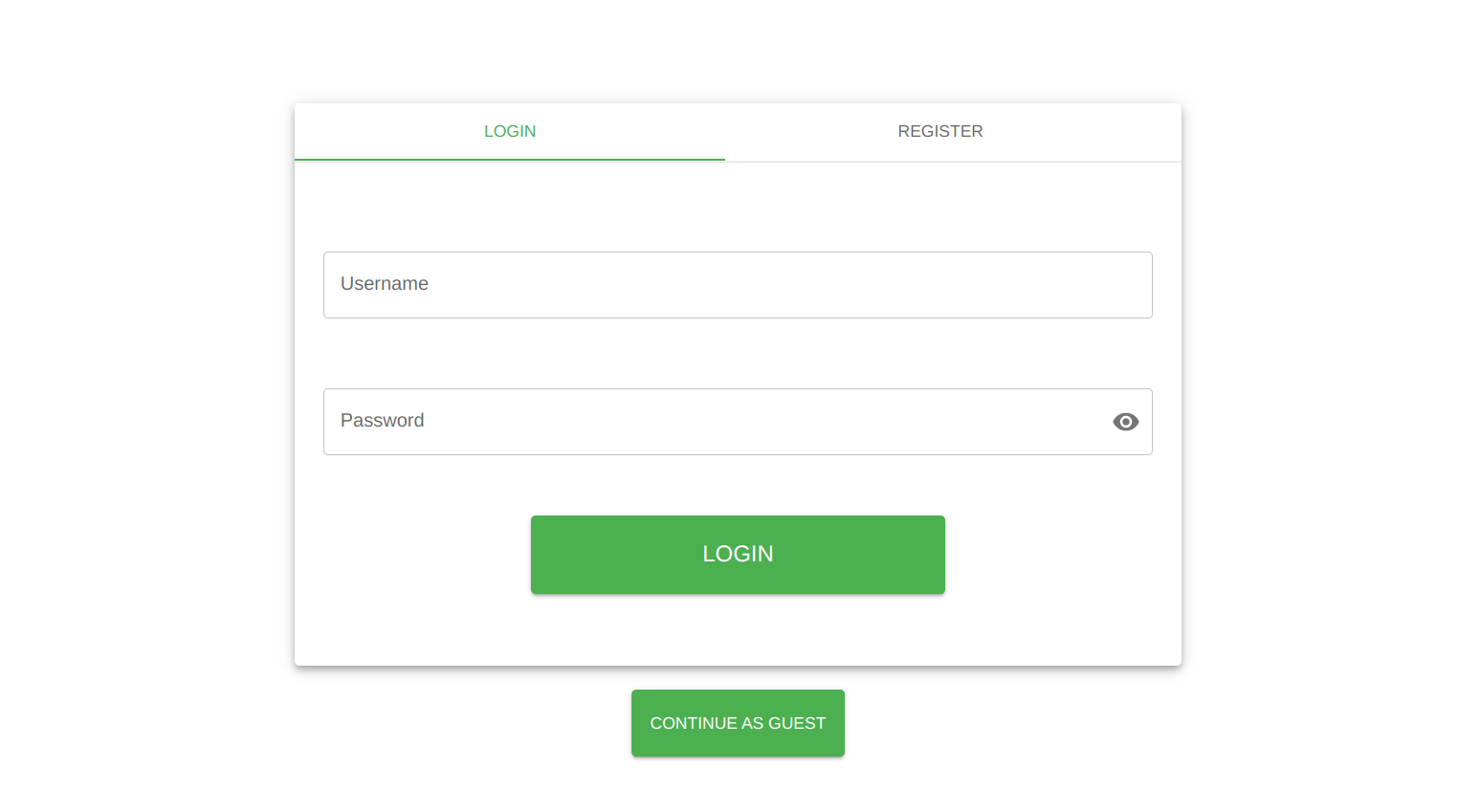
# Korisničko uputstvo

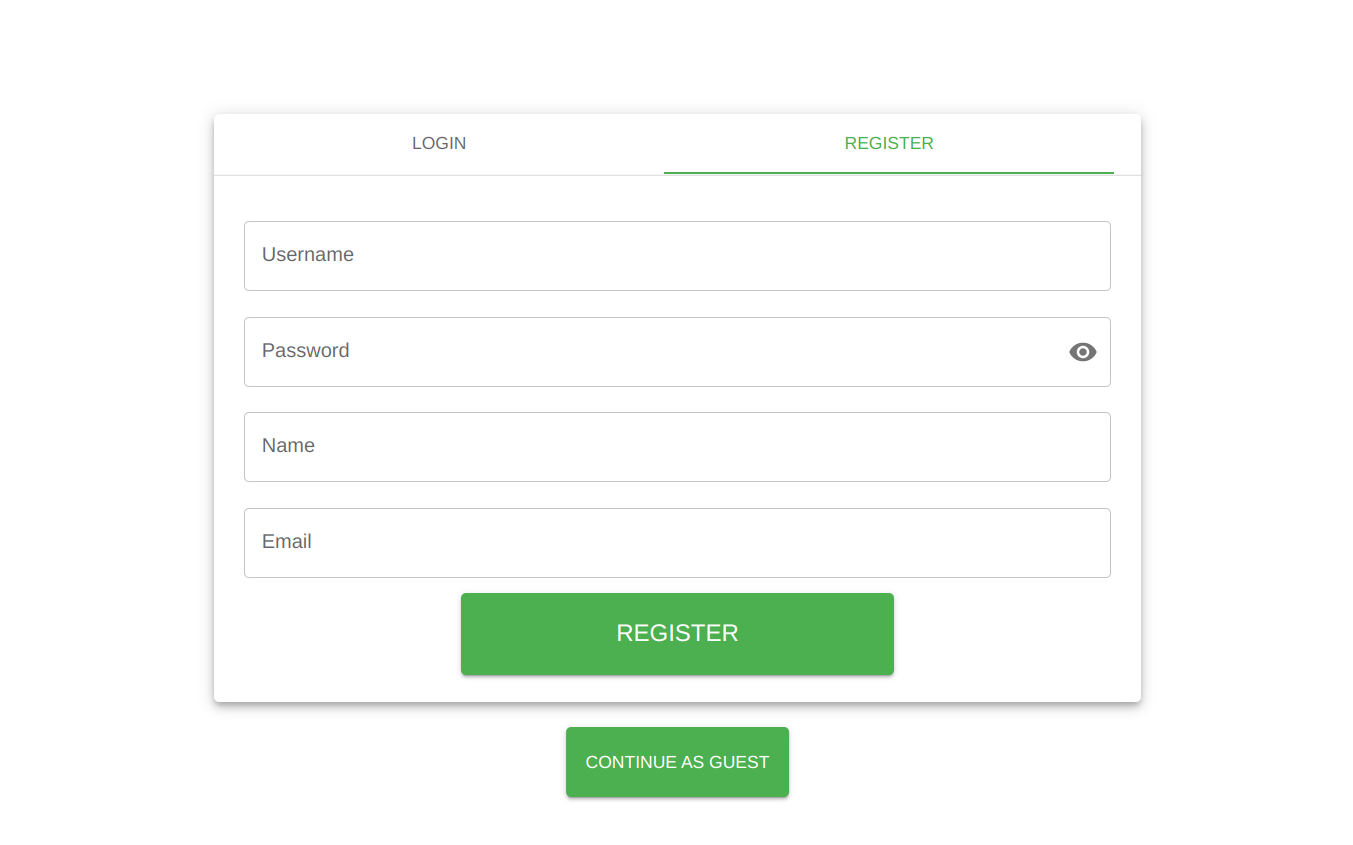
## Prijavljivanje na sistem

Ulaskom na stranicu /signup korisniku se prikazuje forma za prijavu na sistem. Korisniku se daje mogućnost da se prijavi ako već poseduje nalog za aplikaciju kao i mogućnost da se registruje ukoliko ne poseduje nalog. Korisnik takođe može klikom na dugme **Continue as Guest** da nastavi poseduje sajta kao gost, ali će mu u tom slučaju biti onemogućeno da uplati tiket kao i da pregleda svoj profil.

Da bi se korisnik sa postojećim nalogom prijavio na sistem potrebno je da unese korisničko ime (**Username**) i šifru (**Password**) i da pritisne dugme **Login**.

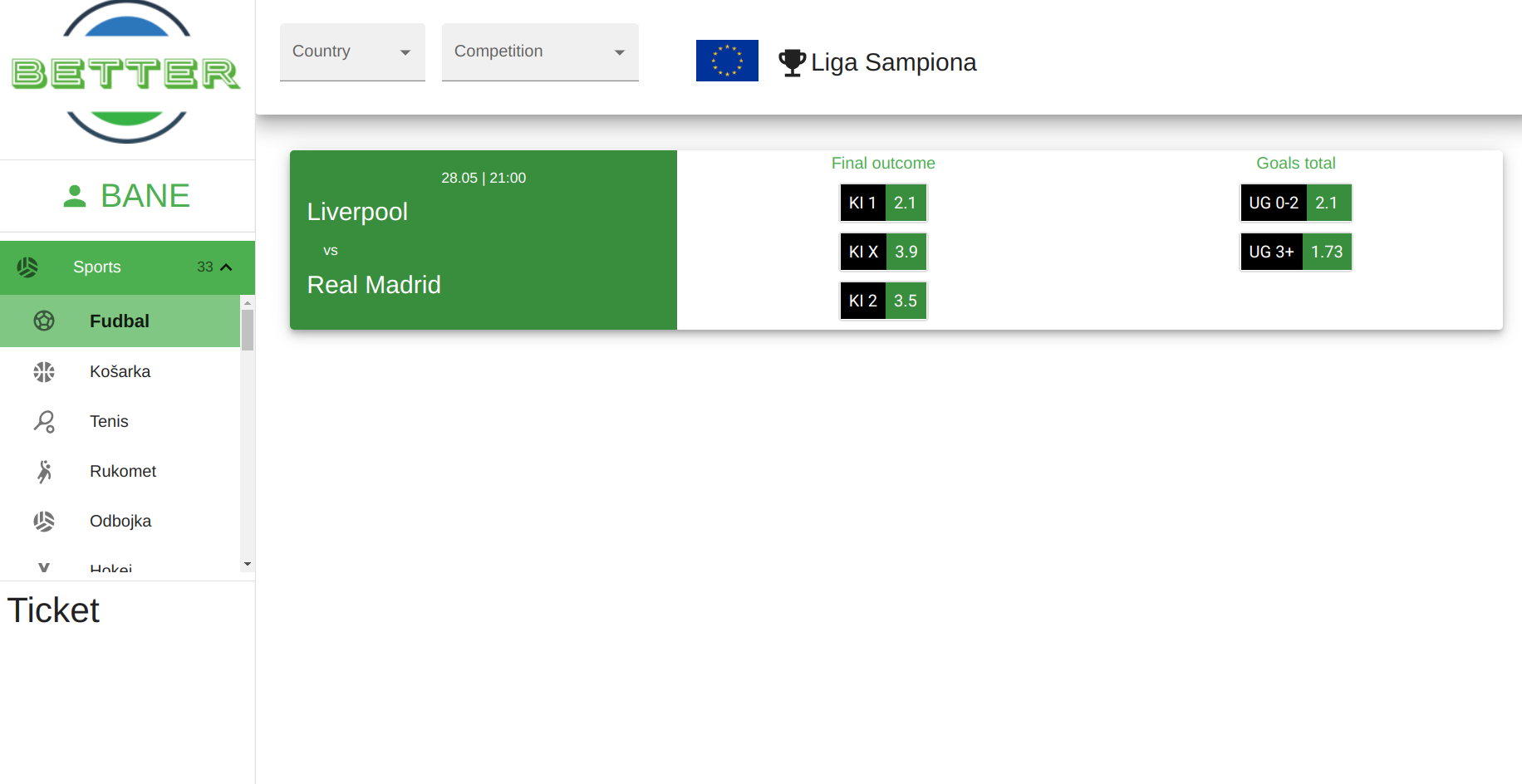
Ukoliko korisnik želi da kreira novi nalog, potrebno je da se prebaci u odeljak za kreiranje naloga, klikom na dugme **Register** i da popune sve potrebne podatke.



Slika 3 - Prijavaljivanje na sistem

Slika 4 - Registracija na sistem

## Početna strana

Početna stranica aplikacije nalazi se na putanji **/home**.

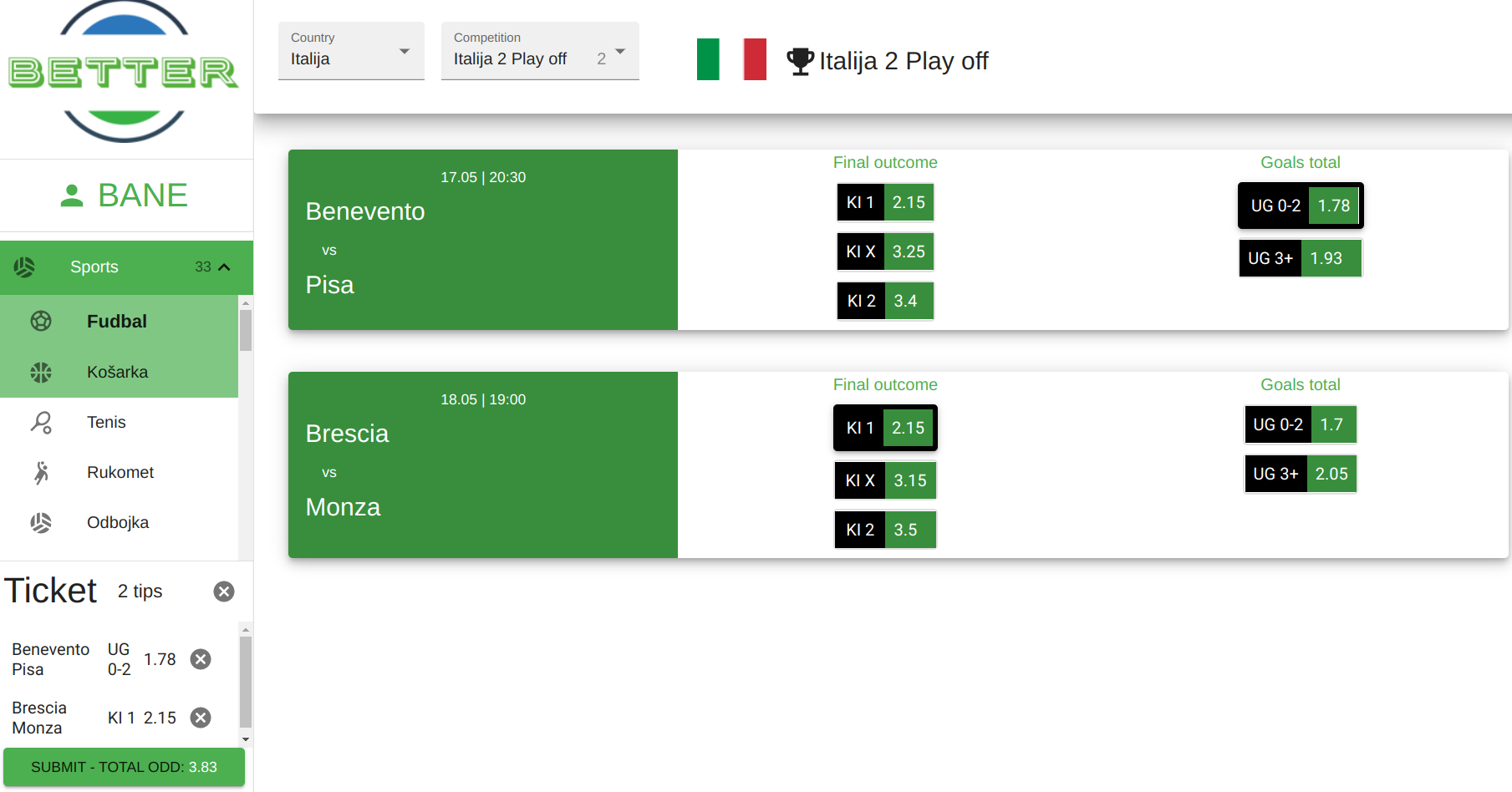
Slika 5 - Izlistavane mečeva

Sa leve strane početne stranice nalaze se izlistani sportovi za koje je moguće raditi prognozu rezultata. Ispod logoa nalazi se korisničko ime trenutno ulogovanog korisnika (u slučaju da korisnik nije ulogovan pisaće **GUEST**).

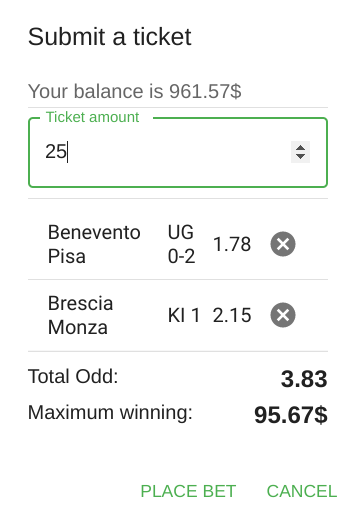
Potrebno je da korisnik prvo odabere sport koji ga zanima (u listi sa leve strane). Zatim da izabere državu kojoj taj meč pripada i takmičenje u okviru kojeg se meč održava.

Nakon toga biće izlistani svi mečevi koji zadovoljavaju kriterijume koje je korisnik uneo.

## Uplata tiketa

Kada korisnik pronađe meč koji želi da odigra potrebno je da klikne na rezultat koji misli da će biti ishod tog meča i taj meč će biti dodat u tiket koji će korisnik kasnije moći da uplati. 

Slika - Pravjenje tiketa

Sa leve strane biće izlistani svi rezultati koje je korisnik za sada izabrao i biće izračunata ukupna kvota na trenutnom tiketu. Kada je korisnik gotov sa izborom rezultata i spreman da uplati tiket potrebno je da pritisne dugme **Submit**.

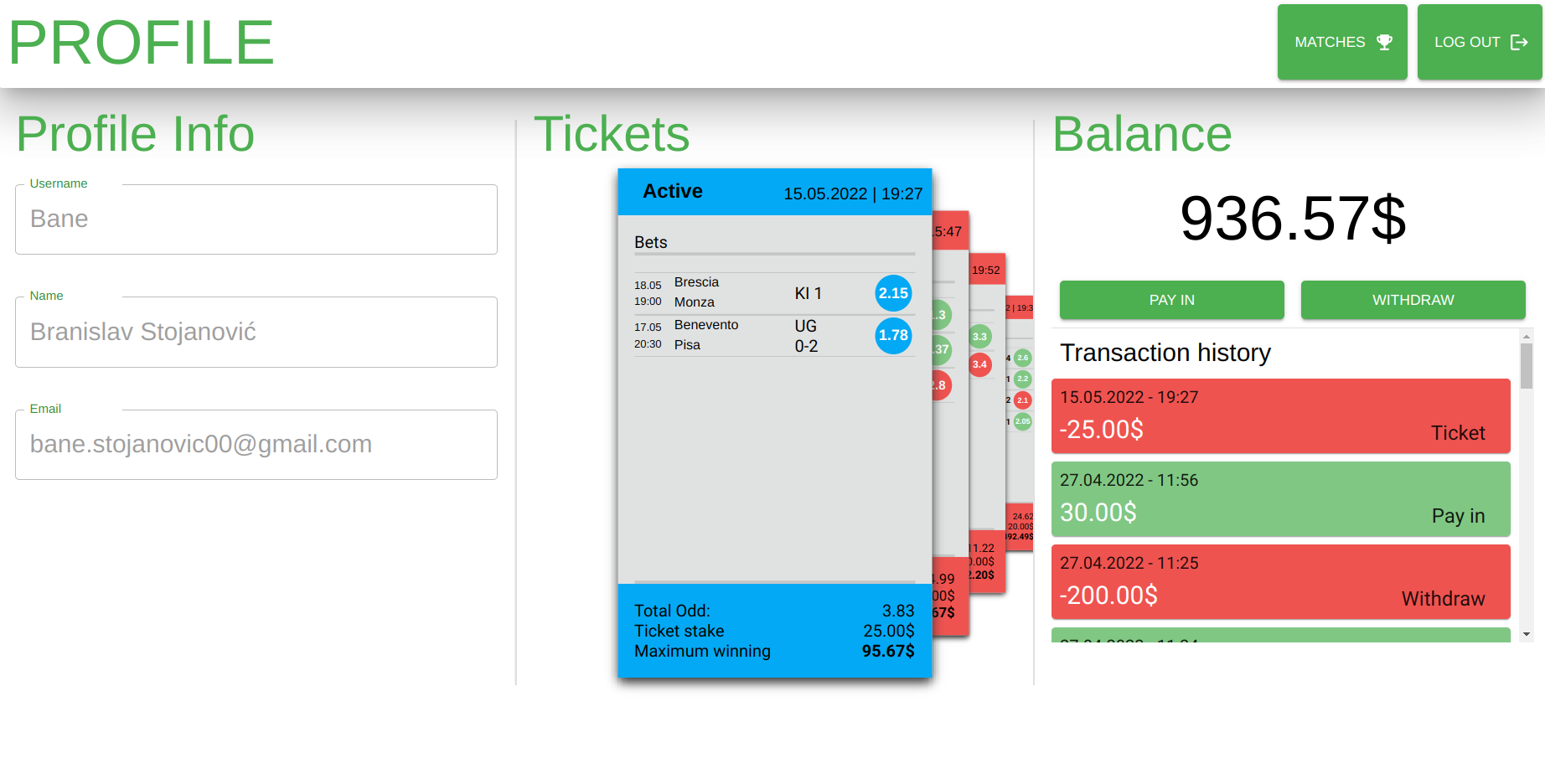
Pritiskom na dugme **Submit** korisniku se otvara modal za uplatu tiketa na kojem može ponovo da pregleda odigrane mečeve i da izbaci neki od izabranih rezultata. Takođe korisnik u polje **Ticket amount** treba da unese količinu novca (u dolarima) koju želi da uloži u ovaj tiket. Automatski će mu biti izračunat maksimalni potencijalni dobitak u slučaju da pogodi sve rezultate sa tiketa.

Klikom na dugme **Place Bet** tiket će biti uplaćen.

Ukoliko korisnik nema dovoljno sredstava da uplati tiket, biće mi prikazana poruka o tome.

## Pregled odigranih tiketa

Pregled odigranih tiket moguć je na profilu korisnika. Profil korisnika nalazi se na stranici **/profile**, do koje korisnik može doći klikom na njegovo korisničko ime na početnoj stranici.

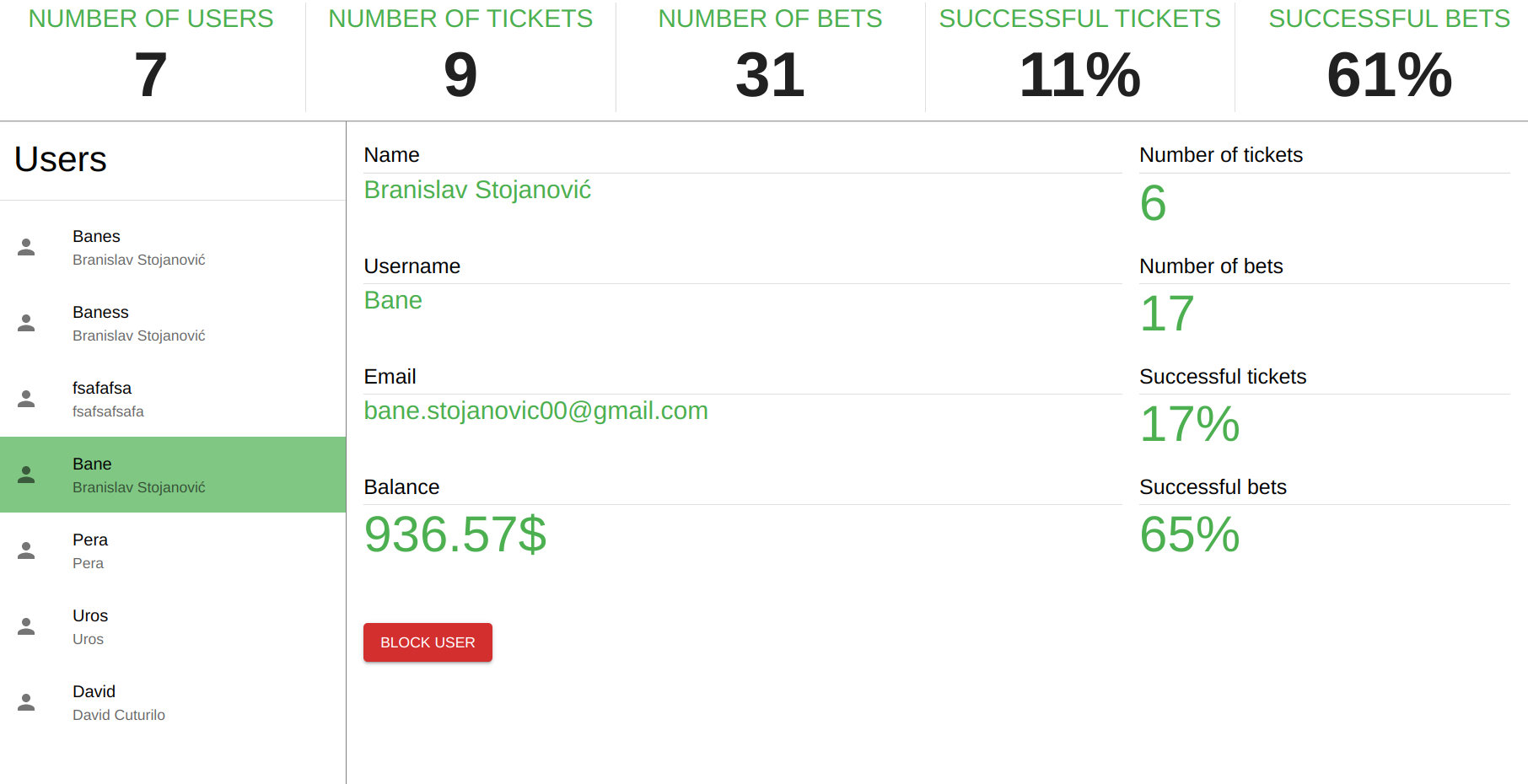


Stranica profil se sastoji od tri sekcije:

* Informacije o korisniku
* Tiketi – plavi tiketi su još uvek aktivni, crveni su promašeni tiketi a zelni su dobitni tiketi
* Balans (stanje računa) – tu je prikazano koliko novca korisnik ima na računu kao i istorija svih novčanih transakcija. U ovom odeljku korisnik može uplatiti novac (dugme **Pay In**) ili povući novac (dugme **Withdraw**)

U gornjem desnom uglu nalazi se dugme **Matches** koje korisnika vodi nazad na početnu stranicu kao i dugme **Log Out** koje će izlogovati korisnika i vratit ga na stranicu za prijavljivanje.

## Administratorski back-office

Administrator ima pristup administratorskoj tabli.

Administrator ima uvid u određene statiskičke podatke, kao što je broj ukupno odigranih tiket, broj prognoziranih rezultata, procenat uspešnih tiketa, broj korisnika...

Sa leve strane administratorske table izlistani su svi postojeći korisnici u sistemu. Klikom na korisnika Administrator ima uvid u profil tog korisnika, na kojem se mogu videti njegovi podaci i statistika o njegovom korišćenju e-kladionice.

# Reprezentativni delovi koda

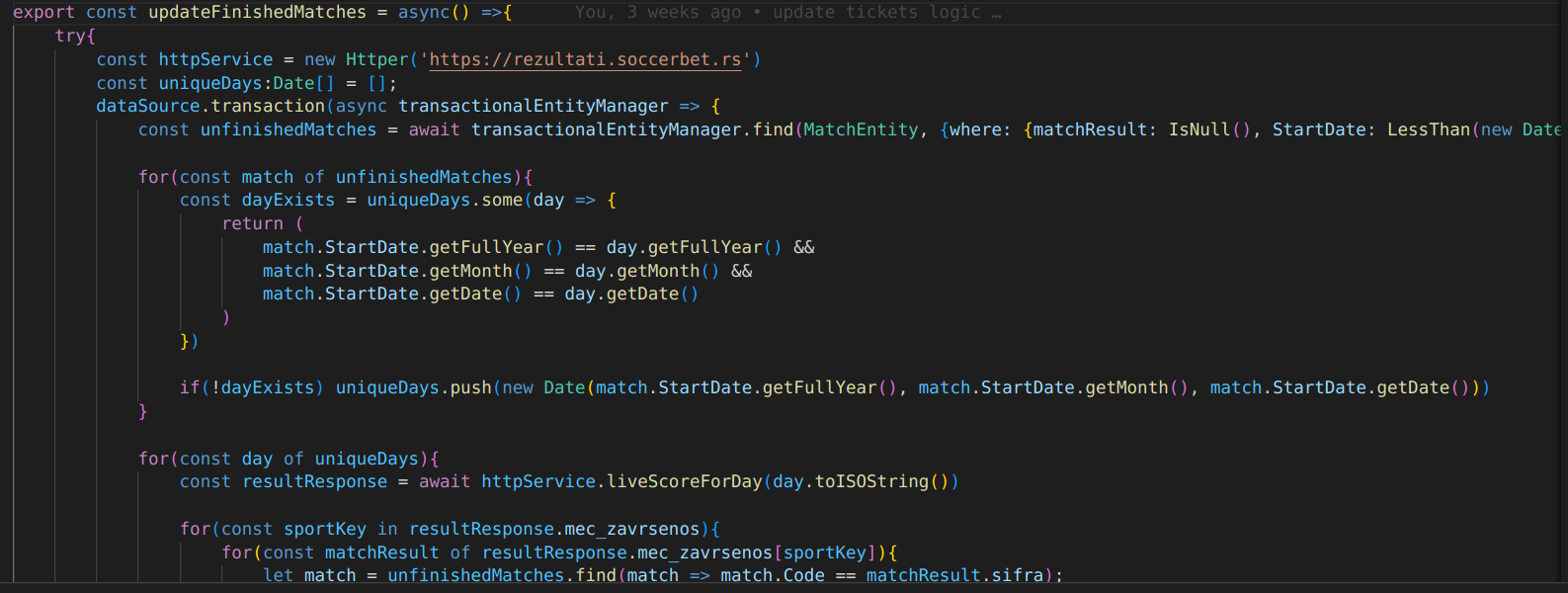
**User Entity – Entitet korisnika**

UserEntity je klasa koja reprezentuje entitet korisnika u modelu sistema. Dekorator @Entity služi TypeORM-u (Objektno Relacionom Maperu) da poveže entitet korisnika sa tabelom korisnika u Bazi. Dekoratori @Column služe za definisanje meta podataka kolona tabele Korisnika.

**AdminDashboard komponenta**

****

AdminDashboard je React komponenta koja se prikazuje administratoru kada se uloguje. U njoj se uz pomoć useEffect i useState React hookova dovlače podaci o korisnicima.

**updataUnfinishedMatches** – funkcija koja ažurira rezultate utakmica kako bi se ažurirali tiketi na kojima su igrane te utakmice. 

Funkcija šalje HTTP Poziv na do servera SoccerBet-a koji vraća sve rezultate sportskih mečeva za određeni dan.