SportsFever

Portal za organizaciju sportskih događaja

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 08.05.2021. | 1.0 | Inicijalna verzija | Nikola, Nikola, Luka, Ena |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Cilj dokumenta 5

2. Opseg dokumenta 5

3. Reference 5

4. Predstavljanje arhitekture 5

5. Ciljevi i ograničenja arhitekture 5

6. Pogled na slučajeve korišćenja 5

6.1 Dijagrami slučajeva korišćenja 6

6.2 Kratak opis slučajeva korišćenja 8

6.2.1 Pregled osnovnih podataka o korisniku 8

6.2.2 Pregled podataka o svim događajima 8

6.2.3 Pregled filtriranih događaja 8

6.2.4 Pregled publikacija po autoru 8

6.2.5 Pregled publikacija po tipu 8

6.2.6 Kreiranje događaja 8

6.2.7 Brisanje događaja 8

6.2.8 Izmena događaja 8

6.2.9 Ažuriranje podataka o članu 8

6.2.10 Upotreba kredita 9

6.2.11 Zahtev za dodavanje kredita 9

6.2.12 Komunikacija između korisnika 9

6.2.13 Kreiranje novog korisnika 9

6.2.14 Prijavljivanje 9

6.2.15 Prijavljivanje na događaj 9

6.2.16 Ocenjivanje korisnika 9

6.2.17 Dodela kredita 9

6.2.18 Suspendovanje korisnika 9

6.2.19 Brisanje prijavljenih događaja 9

7. Pogled na logičku arhitekturu sistema 9

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 10

7.1.1 Korisnički interfejs 10

7.1.2 Aplikaciona logika 10

7.1.3 Pristup podacima 10

7.1.4 ReactJS 10

7.1.5 ExpressJS 11

7.1.6 MongoDB 11

8. Pogled na procese 11

8.1 Procesi 11

8.1.1 Web čitač 11

8.1.2 Web server 11

8.1.3 ExpressJS 11

8.1.4 MondoDB Server 12

9. Pogled na raspoređivanje sistema 12

9.1 Klijent 12

9.2 Web server 12

9.3 DBMS server 12

10. Pogled na implementaciju sistema 12

10.1 Model domena 12

10.2 Šema baze podataka 13

10.3 Komponente sistema 14

10.3.1 Komponente korisničkog interfejsa 14

10.3.2 Komponente aplikacione logike 14

10.3.3 Komponente za pristup podacima 15

11. Performanse 16

12. Kvalitet 16

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture SportsFever portala.

# Opseg dokumenta

Dokument se odnosi na SportsFever portal koji će biti razvijen od strane 404Team-a. Namena portala je efikasno prezentovanje, kreiranje, učestvovanje i održavanje organizacije sportskih događaja.

# Reference

Spisak korišćene literature:

1. SportsFever – Predlog projekta, V1.0, 2021, 404! Team name does not exist.
2. SportsFever – Planirani raspored aktivnosti na projektu, V1.0, 2021, 404Team.
3. SportsFever – Plan realizacije projekta, V1.0, 2021, 404Team.
4. SportsFever – Vizija sistema, V1.0, 2021, 404Team.
5. SportsFever – Specifikacija zahteva, V1.0, 2021, 404Team.

# Predstavljanje arhitekture

Arhitektura sistema u dokumentu je prikazana kao serija pogleda na sistem: pogled na slučajeve korišćenja, pogled na logičku arhitekturu sistema, pogled na procese, pogled na razmeštaj komponenti sistema i pogled na implementaciju. Ovi pogledi su predstavljeni odgovarajućim UML dijagramima.

# Ciljevi i ograničenja arhitekture

Ključni zahtevi i sistemska ograničenja koja imaju značajan uticaj na izbor arhitekture i projektovanje sistema su:

1. SportsFever portal će biti implementiran kao Web aplikacija zasnovana na JavaScript skripting jeziku i MongoDB bazi podataka.
2. Klijentski deo SportsFever portala će biti optimizovan za sledeće Web čitače: Internet Explorer 6.0 i noviji, Opera 8.0 i noviji, kao i Firefox (Mozilla), Brave, Microsoft Edge.
3. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.

# Pogled na slučajeve korišćenja

U ovom odeljku je dat pogled na slučajeve korišćenja definisane u specifikaciji zahteva [5].

Slučajevi korišćenja SportsFever portala su:

* Prijavljivanje
* Ocenjivanje korisnika
* Ažuriranje podataka o događaju
* Kreiranje/brisanje događaja
* Pretraga događaja
* Prijavljivanje na događaj
* Komunikacija sa drugim korisnicima
* Ažuriranje podataka o sebi
* Upotreba kredita

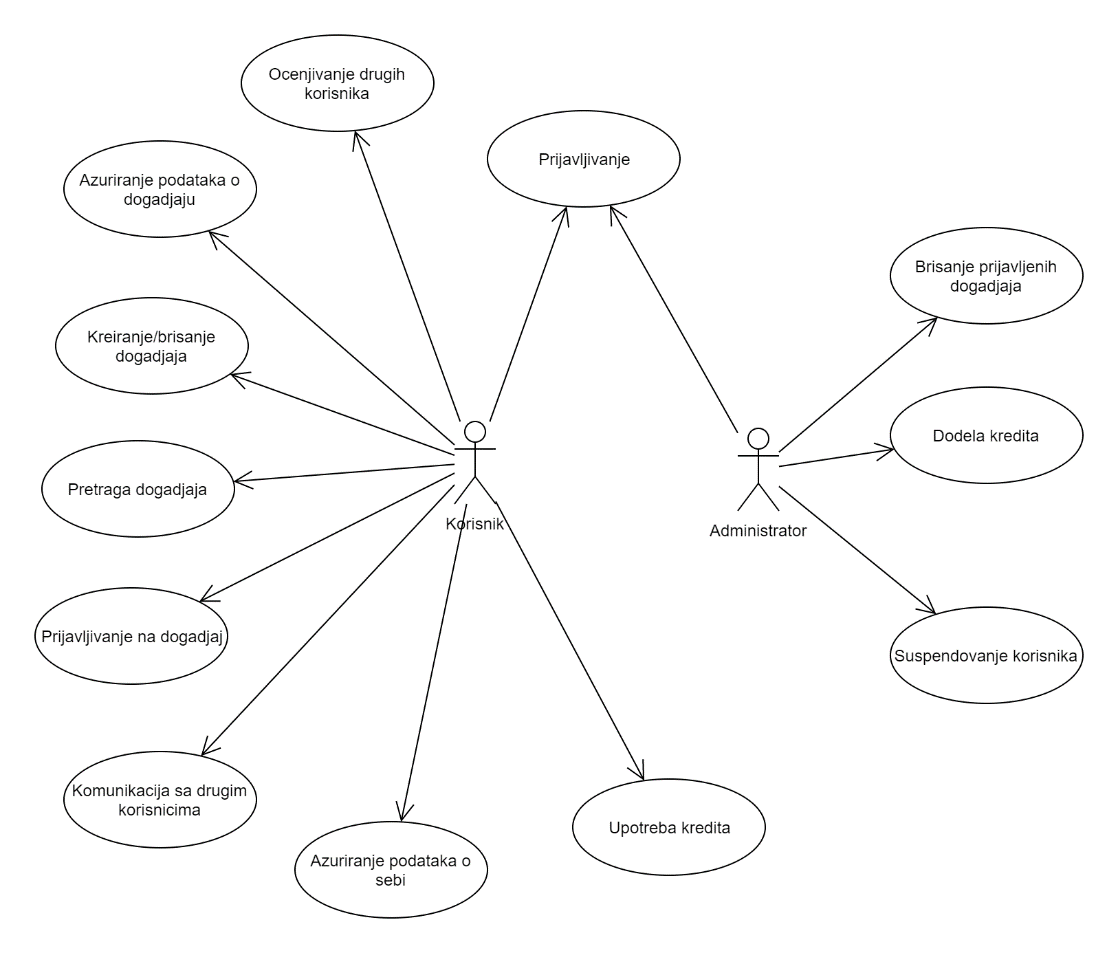
**Admin:**

* Suspendovanje korisnika
* Dodela kredita
* Brisanje događaja
* *Pregled informacija*
  + - *Pregled Podataka o događaju*
      * Pregled događaja po lokaciji
      * Pregled događaja po tipu
      * Pregled događaja po datumu
      * Pregled događaja po ceni
    - *Pregled podataka o korisniku*
      * Pregled podataka o kreiranim događajima
      * Pregled ličnih informacija
      * Pregled ocene i komentara kojima su drugi korisnici ocenili korisnika
* Upravljanje događajima
  + - Kreiranje događaja
    - Brisanje događaja
    - Izmena događaja
    - Arhiviranje događaja
* Upravljanje korisnicima
  + - Kreiranje korisnika
    - Arhiviranje korisnika

Ove slučajevi korišćenja mogu da iniciraju posetilac portala, ulogovani korisnik ili administrator.

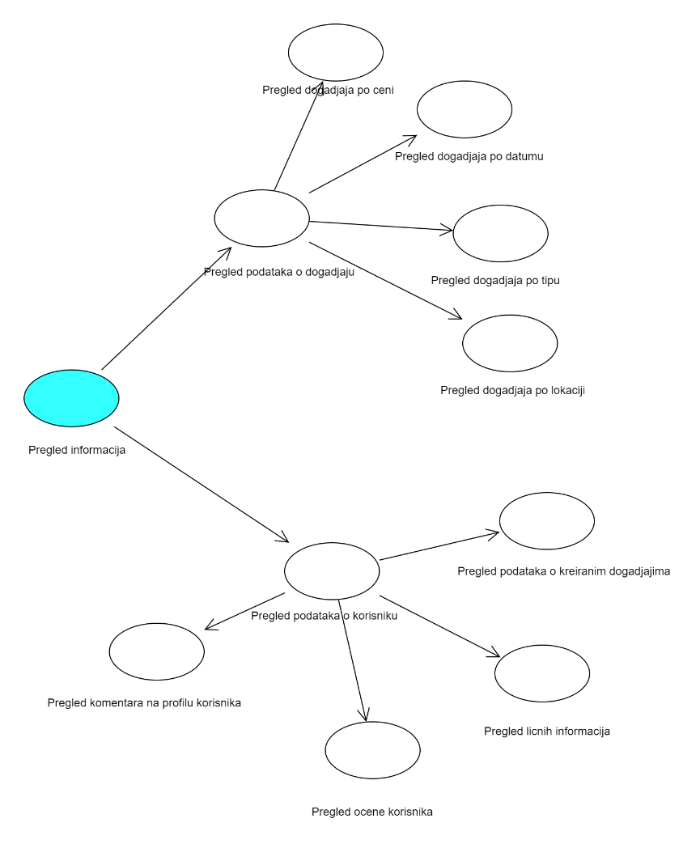
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja SportsFever portala prikazan je na sledećoj slici:

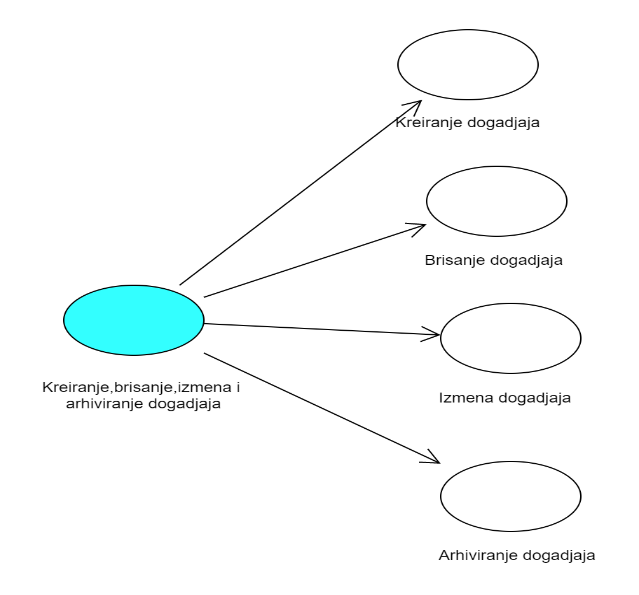


Slučajevi korišćenja *pregled informacija* i *kreiranje, brisanje i arhiviranje članova* i *ažuriranje podataka o publikacijama* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

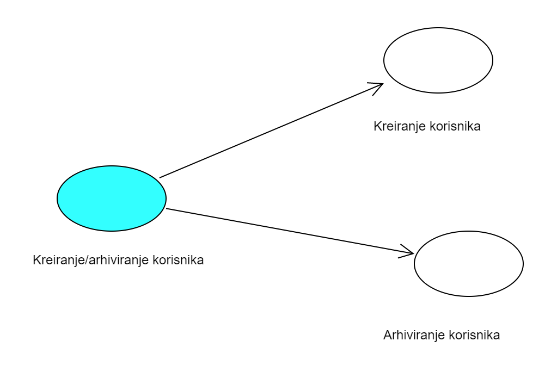
Detaljni UML dijagram za slučaj korišćenja *pregled informacija* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *kreiranje, brisanje i arhiviranje događaja* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *ažuriranje podataka o korisnicima* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Pregled osnovnih podataka o korisniku

**Kratak opis:**

Prikaz stranice portala sa osnovnim informacijama o prijavljenom korisniku.

**Akteri:**

Posetilac portala, Korisnik, Administrator.

### Pregled osnovnih podataka o događajima

**Kratak opis:**

Prikaz stranice portala sa mapom na kojoj se nalaze svi kreirani događaji.

**Akteri:**

Posetilac portala, Korisnik, Administrator.

### Pregled filtriranih događaja

**Kratak opis:**

Prikaz stranice sa informacijama o događajima koji su rezultati specifične pretrage.

**Akteri:**

Posetilac portala, Korisnik, Administrator.

### Pregled publikacija po autoru

Prikaz stranice sa spiskom publikacija čiji je jedan od autora određeni korisnik portala.

**Akteri:**

Posetilac portala, Korisnik, Administrator.

### Pregled publikacija po tipu

**Kratak opis:**

Prikaz stranice sa publikacijama razvrstanim po tipu.

**Akteri:**

Posetilac portala, Član laboratorije, Šef laboratorije, Vođa projekta, Administrator.

### Kreiranje događaja

**Kratak opis:**

Prikaz stranice sa odgovarajućom formom za unos podataka.

**Akteri:**

Korisnik, Administrator.

### Brisanje događaja

**Kratak opis:**

Prikaz stranice sa izabranim događajem kojeg korisnik želi da obriše.

**Akteri:**

Korisnik, Administrator.

### Izmena događaja

**Kratak opis:**

Prikaz stranice sa izabranim događajem kojeg korisnik želi da izmeni.

**Akteri:**

Korisnik, Administrator.

### Ažuriranje podataka o članu

**Kratak opis:**

Ažuriranje podataka o sebi od strane prijavljenog korisnika portala.

**Akteri:**

Korisnik, Administrator.

### Upotreba kredita

**Kratak opis:**

Upotreba virtuelne valute pri vršenju plaćanja za prijavljivanje na događaje.

**Akteri:**

Korisnik, Administrator.

### Dodavanje kredita

**Kratak opis:**

Korisnik šalje zahtev administratoru za dodavanje kredita na svoj nalog.

**Akteri:**

Korisnik, Administrator.

### Komunikacija između korisnika

**Kratak opis:**

Korisnici prijavljeni na isti događaj mogu međusobno komunicirati.

**Akteri:**

Korisnik, Administrator.

### Kreiranje novog korisnika

**Kratak opis:**

Kreiranje korisničkog naloga.

**Akteri:**

Posetilac portala, Administrator.

### Prijavljivanje

**Kratak opis:**

Prijavljivanje korisnika/administratora na portal.

**Akteri:**

Korisnik, Administrator.

### Prijavljivanje na događaj

**Kratak opis:**

Prijavljivanje korisnika na željeni sportski događaj.

**Akteri:**

Korisnik, Administrator.

### Ocenjivanje korisnika

**Kratak opis:**

Ocenjivanje drugih korisnika na osnovu iskustva sa održanim događajem.

**Akteri:**

Korisnik, Administrator.

### Dodela kredita

**Kratak opis:**

Administrator dodaje kredit korisniku koji je poslao zahtev i potvrdu o uplati.

**Akteri:**

Administrator, korisnik.

### Suspendovanje korisnika

**Kratak opis:**

Suspendovanje korisnika za neprikladno ponašanje na portalu.

**Akteri:**

Administrator, korisnik.

### Brisanje prijavljenih događaja

**Kratak opis:**

Brisanje događaja koje je korisnik prijavio kao neprikladan sadržaj.

**Akteri:**

Administrator.

# Pogled na logičku arhitekturu sistema

U ovom odeljku je dat pregled logičke arhitekture sistema. Ovaj pogled sadrži opis najznačajnijih klasa, njihove organizacije u pakete i podsisteme, i organizacija podsistema u slojeve. U cilju opisivanja dinamičkih aspekata arhitekture, ovaj odeljak može da uključi opise realizacije najznačajnijih slučajeva korišćenja. Da bi se ilustrovala veza između arhitekturno značajnih klasa, podsistema, paketa ili slojeva moguće je uključiti i odgovarajuće dijagrame klasa.

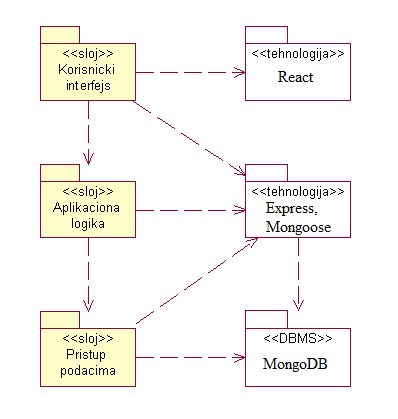
Logički pogled na SportsFever portal obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice, JavaScript(ReactJS) skripte i multimedijalni sadržaj koji realizuju grafički dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom.

Paket *Aplikaciona logika* predstavlja srednji sloj sistema koji sadrži JavaScript(ExpressJS) skripte zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži JavaScript(ExpressJS, mongoose) skripte koje predstavljaju interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### Korisnički interfejs

sloj

Ovaj sloj realizuje korisnički interfejs portala. U njemu su sadržane sve HTML, multimedijalni sadržaji i ReactJS skripte koje generišu HTML stranice preko kojih korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i paketa HTML i ReactJS.

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi SportsFever portala. Sadrži JavaScript i ExpressJS skripte koje realizuju funkcionalnost karakterističnu za domen primene portala i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima, JavaScript i ExpressJS paketa.

### Pristup podacima

sloj

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži ExpressJS i mongoose skripte zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u MongoDB bazi podataka.

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa ExpressJS, mongoose i MongoDB baza podataka.

### HTML

tehnologija

Tehnologija HTML-a definiše gradivne elemente stranica koje se prikazuju u Web čitaču i koje omogućavaju prikaz formatiranih informacija i realizaciju formi za unos i ažuriranje podataka.

### ReactJS

Tehnologija

Teahnologija ReactJS-a obezbeđuje mehanizam za pisanje skripti koje će se izvršavati na klijentskoj strani, koje će dinamički generisati HTML elemente i povezati događaje za HTML elemente.

### ExpressJS

tehnologija

Tehnologija ExpressJS obezbeđuje mehanizam za pisanje i izvršavanje skripti na strani servera. Ove skripte mogu da pristupaju bazi podataka u cilju pribavljanja, unosa i ažuriranja podataka.

### MongoDB

DBMS

MongoDB predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju SportsFever portala.

# Pogled na procese

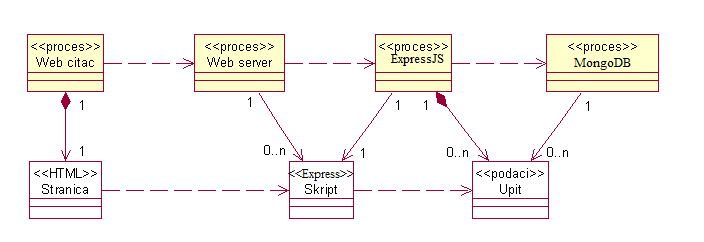
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe spada u opis procesne arhitekture.

Web aplikacije zasnovane na ExpressJS-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta ExpressJS Web aplikacije nije potrebno voditi računa o načinu rada Web servera i načinu izvršavanja skripti.

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje SportsFever portala kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju SportsFever portala. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na ExpressJS-u i MongoDB bazi podataka.



### Web čitač

Web čitač je proces koji izvršava funkcionalnost aplikacije za prikaz HTML stranica dobijenih od nekog Web servera. Web čitač zavisi od Web servera koji generiše i vraća odgovarajuću HTML stranicu na zahtev.

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više Web čitača. Ukoliko je zahtevan JSON objekat, Web server inicira izvršenje ExpressJS skripte koja obrađuje odgovarajući zahtev i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira veći broj JSON zahteva.

### ExpressJS

ExpressJS proces obavlja posao obrade zadatog JSON objekta i generiše odgovarajući tekstualni sadržaj koji Web server šalje Web čitaču. Za izvršenje ExpressJS skripta ovaj proces može da zahteva usluge MongoDB-a. Komunikacija između ExpressJS procesa i MongoDB servera se obavlja preko prosleđivanja upita i vraćanja rezultata.

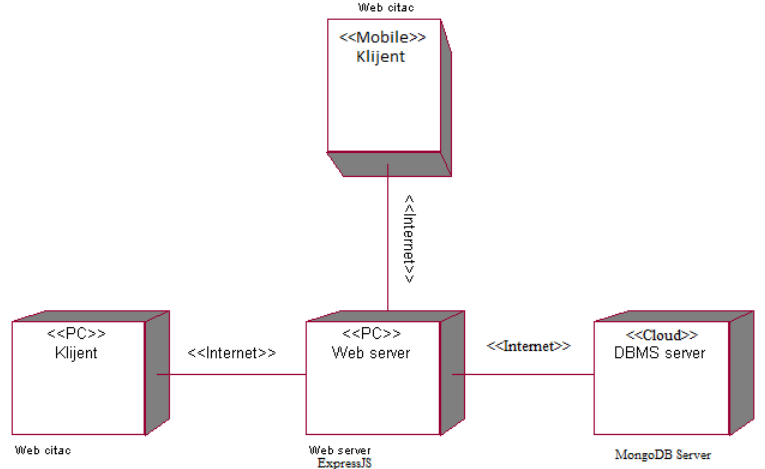
### MongoDB Server

MongoDB Server je proces koji izvršava funkcionalnost MongoDB Atlas sistema za upravljanje bazama podataka. Ovaj proces može konkurentno da prihvati određen broj upita, izvrši ih nad bazom podataka i vrati rezultate procesu koji je upite postavio.

# Pogled na raspoređivanje sistema

Pogled na raspoređivanje sistema prikazuje različite fizičke čvorove za najopštiju konfiguraciju sistema. Fizičkim čvorovima koji predstavljaju procesore vrši se dodeljivanje identifikovanih procesa.

Na sledećoj slici dat je UML dijagram raspoređivanja SportsFever portala.



## Klijent

Pristup SportsFever portalu se obavlja preko klijentskih računara i/ili mobilnih telefona na kojima se izvršava Web čitač. Za povezivanje između klijenta i Web servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta.

## Web server

Računar na kome se izvršava Web server opslužuje više klijenata koji pristupaju preko Interneta. Pored osnovnog procesa koji realizuje funkcionalnost Web servera, na ovom računaru mogu da se izvršavaju i procesi ExpressJS koji vrše obradu zadatih ReactJS skripti. U najopštioj konfiguraciji DBMS se izvršava na cloud-u.

## DBMS server

DBMS server je cloud na kome se izvršava MongoDB Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka. Zbog sigurnosti podataka koji se na ovom računaru čuvaju pristup bazi je ograničen samo na računare čija je ip adresa odobrena na MongoDB Atlas serveru.

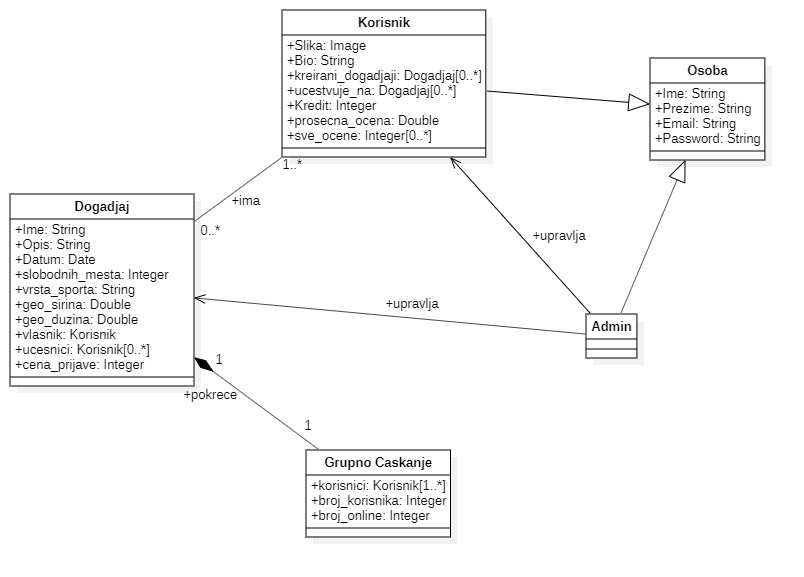
# Pogled na implementaciju sistema

Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju SportsFever portala ovaj odeljak sadrži model domena, šemu baze podataka i prikaz komponenti sistema razvrstanih u ranije identifikovane pakete.

## Model domena

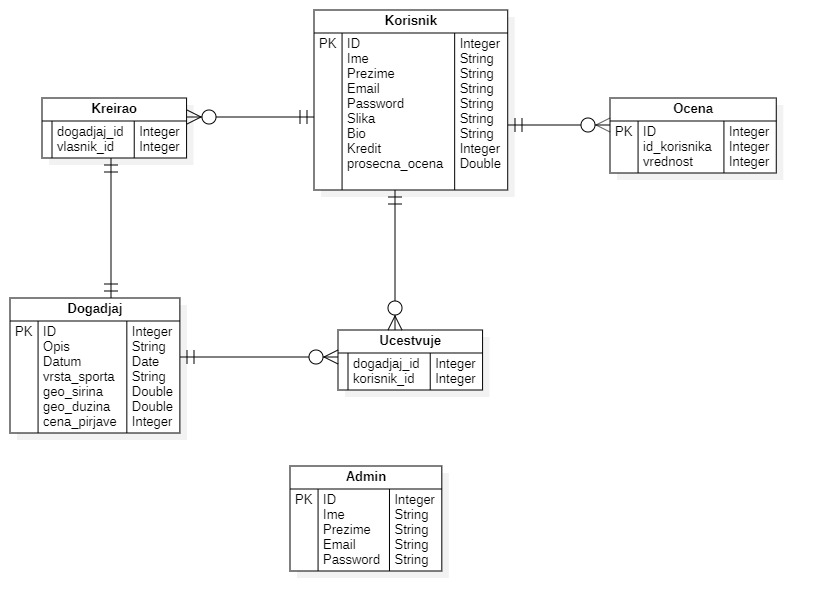
Model domena za koji se SportsFever portal projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

Model domena predstavlja osnovu za projektovanje baze podataka, ali i identifikaciju nekih od komponenti (JavaScript skripti) koje će biti implementirane.



## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem *STARUml*-a.



## Komponente sistema

Komponente sistema SportsFever portala su ReactJS i ExpressJS skripte čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je ReactJS/ExpressJS skripta prikazana kao klasa atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije definisane u okviru skripta.

### Komponente korisničkog interfejsa

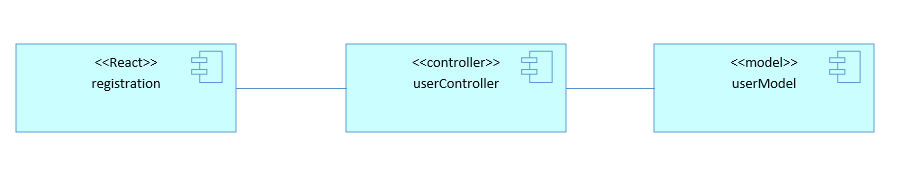
Dizajn korisničkog interfejsa je obuhvaćen dvema komponentama:



Komponenta **index.js** je implementira stranicu portala čiji sadržaj može da varira od parametra koji joj se proslede u zahtevu.

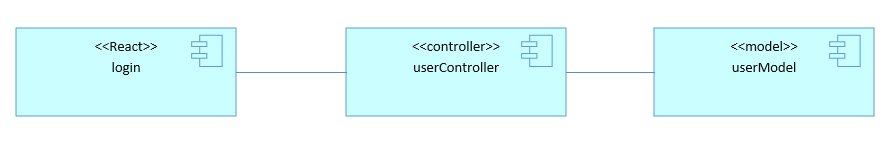
Komponenta **style.css** predstavlja opis stilova za pojedine HTML elemente koji se javljaju na različitim stranicama.

**Kreiranje naloga:**

****

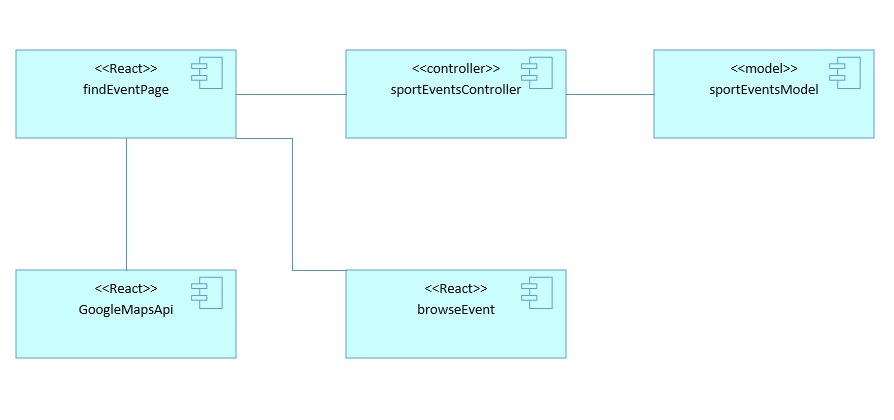
Potencijalni korisnik portala na stranici za kreiranje naloga unosi svoje podatke. ReactJS skripta prosleđuje te podatke User Controlleru za registraciju koji ih prosleđuje User Modelu koji vrši njihovu proveru i vraća korisnika na stranicu za prijavljivanje ili ga obaveštava o nevalidnosti određenih podataka. Te podatke pokuplja ReactJS i prosleđuje ih korisniku. Ukoliko je registracija uspešna korisnik se prosleđuje na portal kao ulogovani korisnik.

**Prijavljivanje:**



Korisnik na Login stranici popunjava formu sa svojim podacima za prijavu. ReactJS te podatke prihvata i prosleđuje User Controlleru koji ih dalje prosleđuje User Modelu koji vrši proveru heširanih podataka kao što su e-mail i password i ukoliko je prijava uspešna, tu informaciju pokuplja ReactJS koji prosleđuje korisnika na portal kao ulogovanog korisnika. Ukoliko je prijava neuspešna, ReactJS skripta prikazuje tu informaciju korisniku.

**Pretraživanje dogadjaja:**

****

Korisnik na početnoj stranici portala može pretraživati događaje filtriranjem bez obzira da li je ulogovan na sistem. Korisnik može izabrati opciju da dozvoli pretraživaču da koristi njegovu geolokaciju zbog precniznijeg pozicioniranja kako bi korisnik mogao da traži događaje u njegovoj blizini. Postoje i ostala rešenja kao što su pretraživanje specifične lokacije (usluga koju nam pruža google map api), a ukoliko korisnik ne dozvoli korišćenje geolokacije i još nije otkucao specifičnu lokaciju, početna tačka će biti Niš.

Kada se mapa učita na specifičnoj lokaciji u određenom radiusu se prikazuju svi događaji tako što nakon učitavanja mape ReactJS skripta šalje GET zahtev serverskom API-u koji u JSON obliku vraća odgovor react skripti koja događaje renderuje na mapi. Korisnik može filtrirati događaje kako bi se prikazivali samo oni događaji za koje je zainteresovan, nakon aplikacije filtera proces učitavanja događaja se ponavlja.

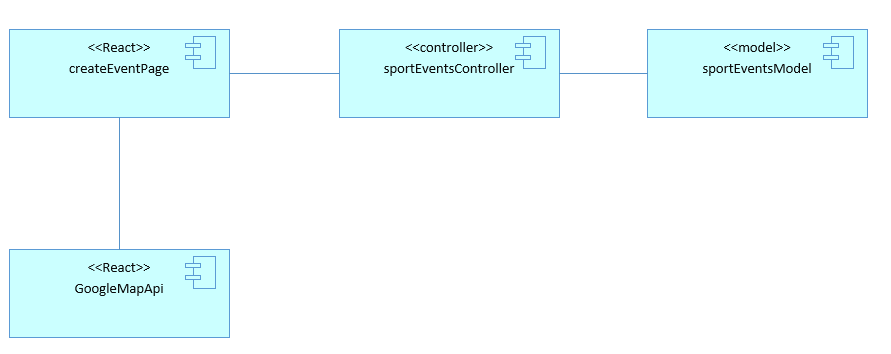
Da bi korisnik video detaljan opis željenog događaja, potrebno je samo da klikne na marker koji je iscrtan na Google mapi.

**Pregled događaja:**

****

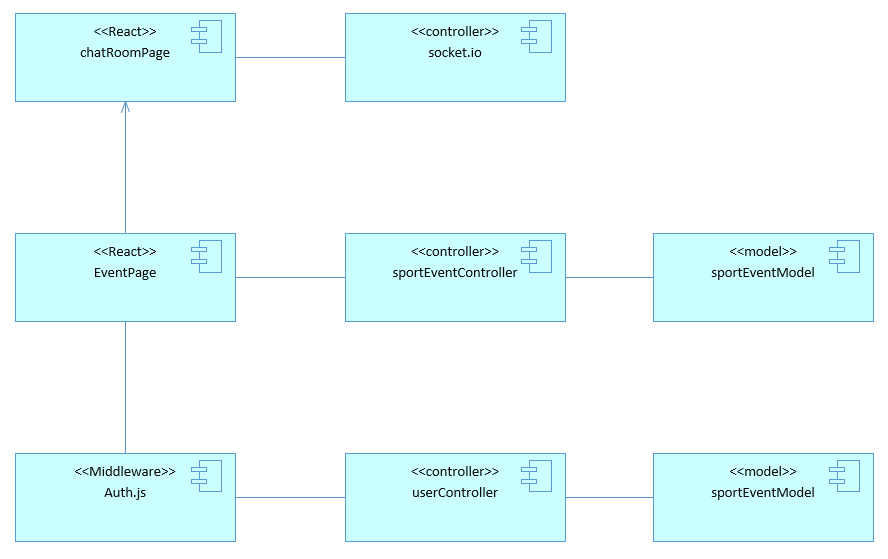
Kada se korisnik navigira na ovu stranicu ReactJS skripta šalje zahtev kontroleru za događaj koji uz pomoć Modela Događaja vraća ReactJS-u odgovarajući JSON objekat koji on kasnije renderuje. Na ovoj stranici se nalaze sve informacije o kreiranom sportskom događaju. Korisnik, ukoliko je kreator može ažurirati događaj, obrisati događaj ili ukloniti prijavljenog korisnika. Ako je u pitanju korisnik koji želi da prijavi dolazak na događaj, to može učiniti jednostavnim klikom na dugme za prijavljivanje, ukoliko je prijavljen na sistem. To se postiže tako što ReactJS skripta šalje zahtev koji Middleware ExpressJS skripta proverava i vraća nazad odgovarajući odgovor. Sam događaj, osim osnovnih informacija nudi i mogućnost chat-ovanja između prijavljenih korisnika koji se mogu dogovarati o ostalim detaljima.

**Kreiranje događaja:**

****

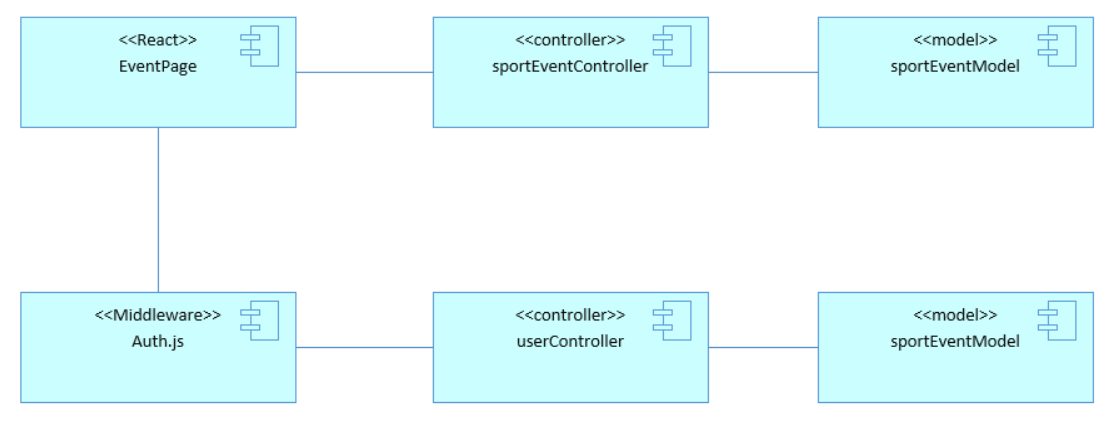
Kada se korisnik navigira na ovu stranicu, ReactJS skripta šalje zahtev Događaj kontroleru sa informacijama koje je prikupila iz forme koju popunjava korisnik. Google Map Api služi za markiranje tačnog mesta događaja i koordinate markera su takođe jedan od podataka koje React skripta šalje Događaj Kontroleru. Kontroler uz pomoć Događaj Modela vraća odgovarajući JSON odgovor, smešta podatke o novom događaju u bazu i novi događaj je kreiran i spreman za renderovanje na Google Mapi.

**Komunikacija između korisnika:**

****

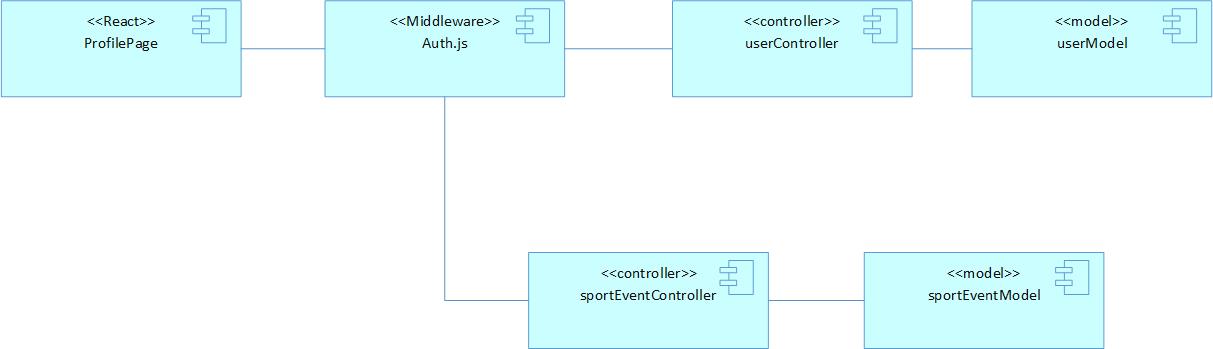
Kada je korisnik prijavljen na događaj, može komunicirati sa ostalim prijavljenim korisnicima korišćenjem posebne sobe za ćaskanje za događaj koja je realizovana pomoću socket.io JavaScript biblioteke.

**Prijavljivanje na događaj:**

****

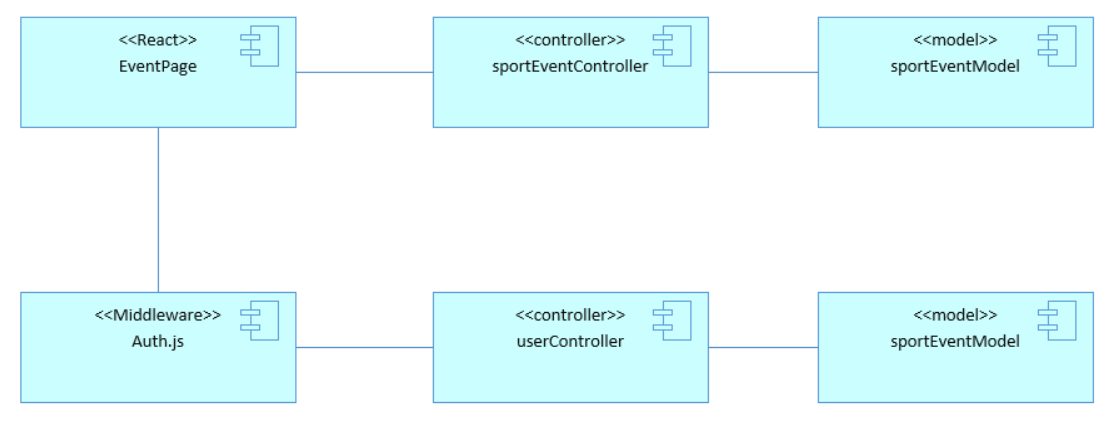
Kada se korisnik nalazi na stranici događaja, on može prijaviti učešće jednostavnim klikom na dugme „Prijavi Se“. Nakon te akcije, ReactJS skripta šalje zahtev middleware-u koji vrši validaciju da li je korisnik prijavljen.Nakon toga pomoću Korisnik Kontrolera prikuplja potrebne informacije o korisniku i ukoliko prijava na događaj zahteva upotrebu određenog broja kredita, vrši validaciju da li korisnik ima dovoljno kredita na računu i vraća odgovarajući odgovor. Kada odgovor stigne, prosleđuje se Događaj Kontroleru koji korisnika dodaje u spisak prijavljenih.

**Pregled korisničkog profila:**

****

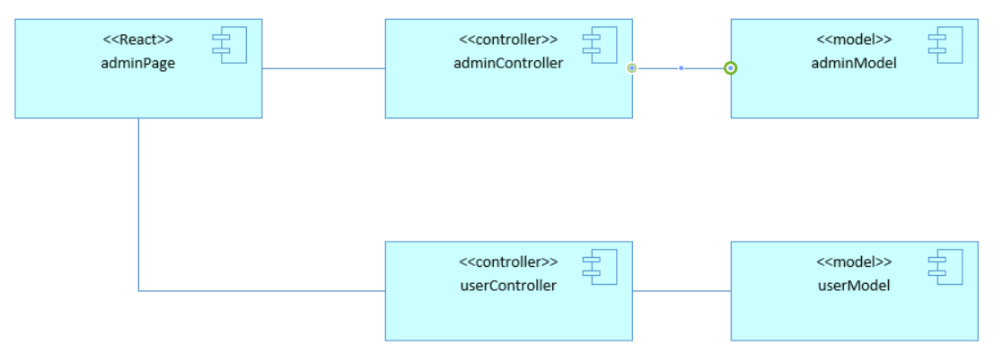
Korisnik ima svoju profilnu stranicu ukoliko je prijavljen na sistem, što se proverava preko middleware-a. Na profilnoj stranici se nalaze sve potrebne informacije o korisniku i o događajima koje je kreirao i na kojima je učestvovao. To se postiže tako što React skripta šalje zahtev serveru koji preko Korisničkog Kontrolera i Korisničkog Modela vraća skripti podatke o korisniku, a preko Događaj Kontrolera i Događaj Modela vraća podatke o događajima.

**Ocenjivanje korisnika:**

****

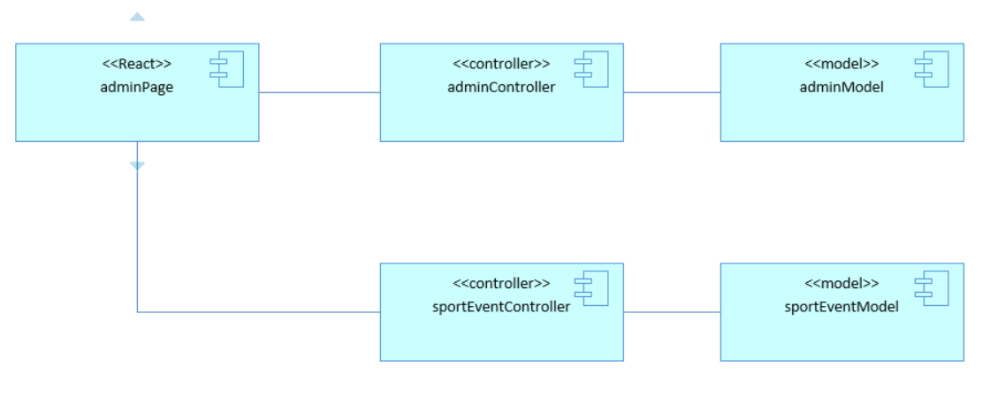
Nakon što se svaki događaj završi, biće omogućeno da korisnici ocenjuju jedni druge. Samo korisnici koji su učestvovali na događaj biće u mogućnosti da se ocenjuju. To se postiže tako što Događaj Kontroler i Događaj Model stranici događaja vraćaju potrebne informacije o događaju i o listi prijavljenih korisnika, a middleware proverava da li je korisnik prijavljen na sistem, ukoliko jeste, vraća podatke o svakom korisniku. Cilj ocenjivanja je da se postigne određeni nivo poverenja o korisniku i da poboljša korisničko iskustvo. Ocene korisnika biće dostupne i javne na stranici korisničkog profila.

**Suspendovanje ili brisanje korisničkih naloga:**

****

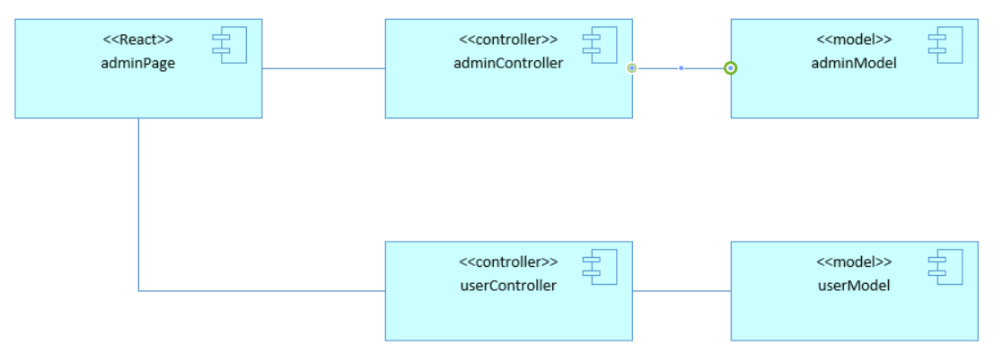
Na adminovoj stranici se nalazi kontrolna tabla sa korisnicima koji su prijavljeni od strane drugih korisnika zbog potencijalno nepoželjnog ponašanja. Admin obrađuje pristigle prijave i odlučuje da li će suspendovati, obrisati ili neće preuzeti nikakve akcije u vezi prijavljenog naloga.

**Brisanje događaja:**

****

Na adminovoj stranici se nalazi kontrolna tabla sa događajima koji su prijavljeni od strane korisnika zbog potencijalno nepoželjnog sadržaja. Admin obrađuje pristigle prijave i odlučuje da li će obrisati događaj ili neće preuzeti nikakve akcije.

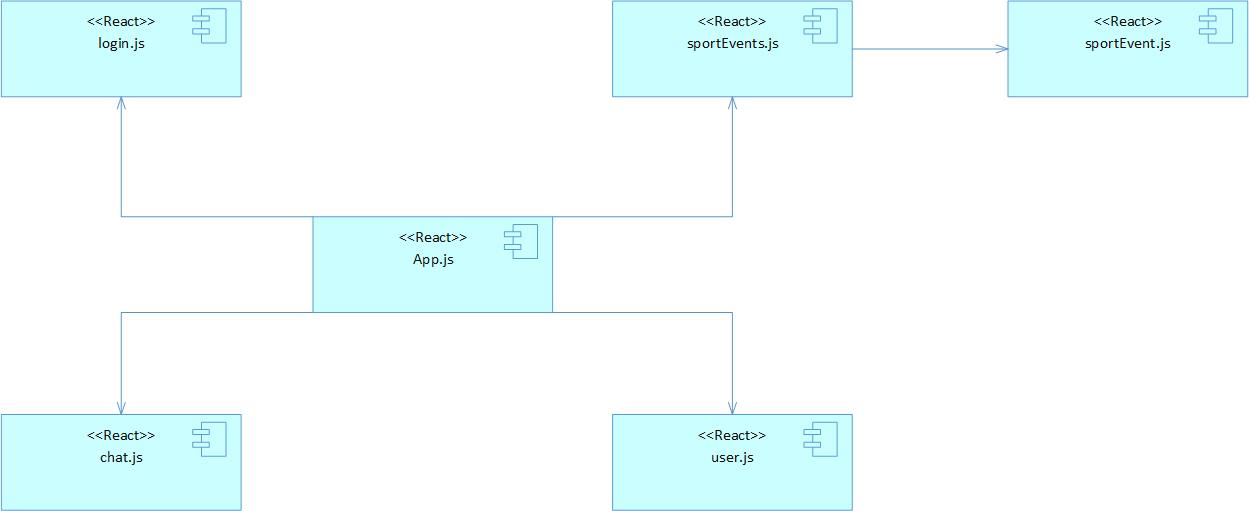
**Dodela kredita korisnicima:**

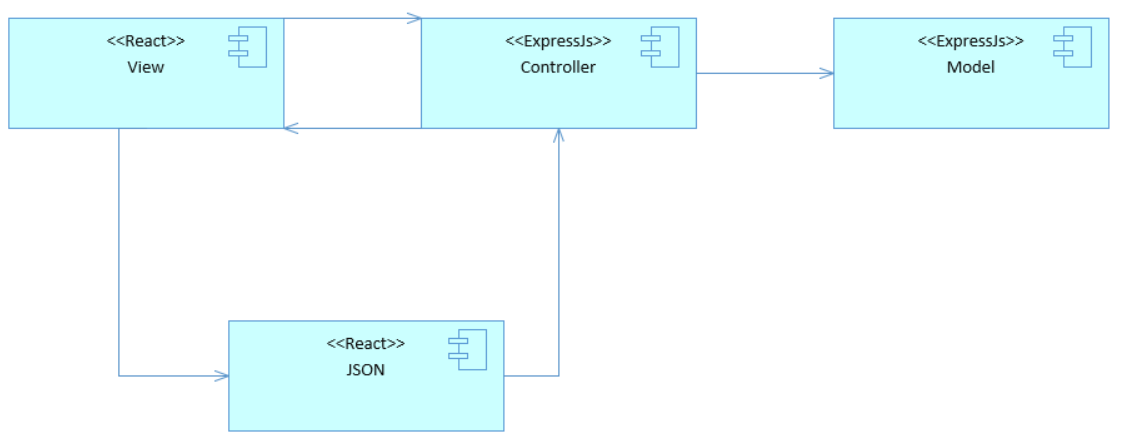
****

Na adminovoj stranici se nalazi kontrolna tabla sa pristiglim zahtevima za dodelu kredita. Korisnik može uplatiti realan novac u zamenu za virtuelne kredite koje može koristiti prilikom prijavljivanja na događaj. Admin obrađuje i potvrđuje validnost zahteva za dodelu i na osnovu toga odlučuje da li će korisniku dodeliti zahtevanu količinu kredita.

### Komponente aplikacione logike

Komponente koje realizuju domen problema se uključuju isključivo preko **App.js** komponente korisničkog interfejsa. Na taj način zadržavaju sva podešavanja stila definisana u ovom skriptu. Na sledećem dijagramu su prikazane komponente ovog sloja i njihove međusobne zavisnosti:





SportsFever portal je baziran na Model-View-Controller (MVC) projektnom obrascu. MVC je softverski pristup koji razdvaja aplikacionu logiku od prezentacione.

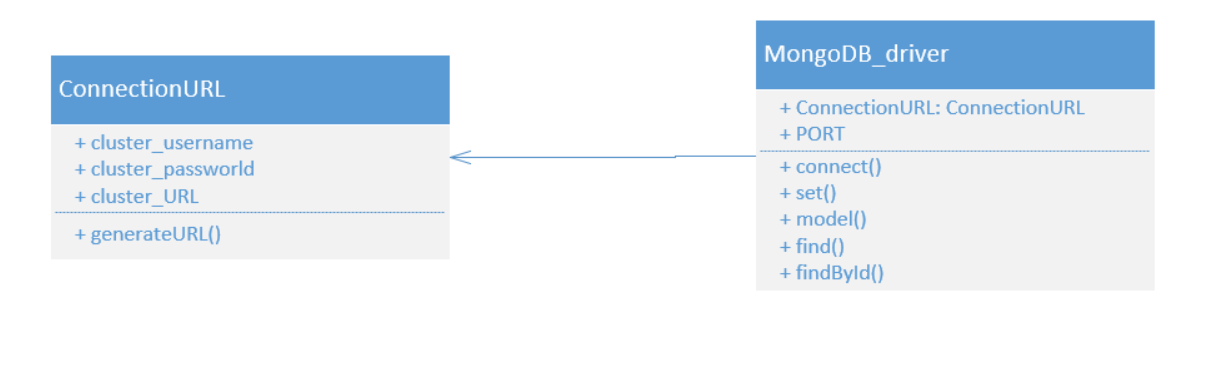
* **Model** predstavlja strukturu podataka. Mode klase sadrži funkcije pomoću kojih preuzimamo, dodajemo i ažuriramo podatke u bazi podataka.
* **View** je skup informacija koje se predstavljaju korisniku. View može biti stranica ali isto tako može biti deo kao što je header ili footer.
* **Controller** je međusloj između modela, view-a i bilo kog drugog resursa potrebnog za obradu HTTP zahteva i generisanja web stranice.

### Komponente za pristup podacima

Pristup bazi podataka je u potpunosti zatvoren u funkcije koje su definisane u okviru mongoose skripta **Mongo\_DB\_Driver**. Pomenuti skript se poziva unutar Modela i služi za prenos podataka između aplikacije i baze.



Na sledećem UML dijagramu klasa pobrojane su funkcije za pristup podacima iz baze:



Navedene funkcije obavljaju sledeće zadatke:

* Mongoose.connect() – povezivanje na bazu podataka, poziva se u samom skriptu
* Set() – Postavljanje mongoose opcija
* Model() – definiše model nalik JSON objektu
* Find() – vraća NoSql dokument
* findById() – vraća NoSql dokument sa specificiranim ID-em

# Performanse

Izabrana arhitektura softvera podržava zahteve u pogledu broja korisnika koji mogu simultano pristupati sistemu i vremena odziva za pristup bazi podataka specificirane u zahtevima u pogledu performansi [5]:

1. Sistem će da podrži do 1000 simultanih pristupa korisnika portalu.
2. Vreme potrebno za pristupanje bazi podataka u cilju izvršenje nekog upita ne sme da bude veće od 5 sekundi.

Zahtevane performanse su zadovoljene izborom tehnologija na kojima će sistem biti razvijen i definisane hardverske platforme [5].

# Kvalitet

Izabrana arhitektura softvera podržava zahteve u pogledu dostupnosti i srednjeg vremena između otkaza specificirane u zahtevima u pogledu pouzdanosti [5]:

1. SportsFever portal će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 10%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.