**Igraj košarku**

**WEB aplikacija za organizovanje košarke**

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 01.05.2022. | 1.0 | Inicijalna verzija | Aleksa Milić |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 Izbor jezika 8

6.2.2 Pregled podataka o košarkaškim aktivnostima 8

6.2.3 Pregled podataka o specijalnim košarkaškim događajima 8

6.2.4 Pregled podataka o stanju na terenima 8

6.2.5 Pregled opisa stanja terena 8

6.2.6 Pregled komentara igrača 8

6.2.7 Pregled vremena održavanja 9

6.2.8 Pregled timova 9

6.2.9 Pregled broja igrača 9

6.2.10 Pregled pravila igre 9

6.2.11 Pregled lokacije terena 9

6.2.12 Pregled mape 9

6.2.13 Registracija 9

6.2.14 Kreiranje naloga 9

6.2.15 Logovanje 9

6.2.16 Unos stanja terena 9

6.2.17 Prikazivanje stanja terena 9

6.2.18 Ažuriranje stanja terena 9

6.2.19 Ostavljanje komentara/ocene 9

6.2.20 Unos košarkaške aktivnosti 9

6.2.21 Prikazivanje košarkaške aktivnosti 10

6.2.22 Ažuriranje košarkaške aktivnosti 10

6.2.23 Ostavljanje komentara/ocene aktivnosti 10

6.2.24 Ažuriranje i brisanje profila korisnika 10

6.2.25 Ažuriranje i brisanje članova 10

6.2.26 Kreiranje tima 10

6.2.27 Dogovor u vezi terena 10

6.2.28 Komunikacija korisnika 10

7. Pogled na logičku arhitekturu sistema 10

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 11

7.1.1 Korisnički interfejs 11

7.1.2 Aplikaciona logika 11

7.1.3 Pristup podacima 11

7.1.4 HTML 12

7.1.5 CSS 12

7.1.6 JavaScript 12

7.1.7 .NET Core 12

7.1.8 PostgreSQL 12

8. Pogled na procese 12

8.1 Procesi 12

8.1.1 Web čitač 13

8.1.2 Web server 13

8.1.3 .NET Core 13

8.1.4 PostgreSQL Server 13

9. Pogled na raspoređivanje sistema 13

9.1 Klijent 14

9.2 Web server 14

9.3 DBMS server 14

10. Pogled na implementaciju sistema 14

10.1 Model domena 14

10.2 Šema baze podataka 15

10.3 Komponente sistema 16

10.3.1 Komponente sistema Igraj košarku portala će biti date po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je prikazan kao klasa atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije. 16

10.3.2 Komponente korisničkog interfejsa 16

10.3.3 Komponente aplikacione logike 17

10.3.4 Komponente za pristup podacima 18

11. Performanse 20

12. Kvalitet 20

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture **Igraj košarku** Web aplikacije.

# Opseg dokumenta

Dokument se odnosi na Igraj košarku Web aplikaciju koja će biti razvijena od strane tima Dva stranca**. Igraj košarku** predstavlja sajt za organizovanje košarkaških aktivnosti. Namena sistema je jednostavno i efikasno organizovanje i učestvovanje u raznim košarkaškim aktivnostima.

# Reference

Spisak korišćene literature:

1. Igraj košarku – D01\_Predlog\_Projekta, SWE-Igraj kosarku-01, V1.0, 2022, Dva stranca.
2. Igraj košarku – D02\_Vizija\_Sistema, SWE-Igraj kosarku-02, V1.0, 2022, Dva stranca.
3. Igraj košarku – D03\_Plan\_Realizacije, SWE-Igraj kosarku-03, V1.0, 2022, Dva stranca.
4. Igraj košarku – D04\_Spec\_Zahteva, SWE-Igraj kosarku-04, V1.0, 2022, Dva stranca.

# 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. ***Igraj košarku*** Web aplikacija će biti implementirana kao Web aplikacija zasnovana na .NET framework-u jeziku i PostgreSQL bazi podataka [4].
2. Klijentski deo ***Igraj košarku*** aplikacije će biti optimizovan za sledeće Web čitače: Edge 89.0 i noviji, Opera 8.0 i noviji, kao i Firefox (Mozilla) [4].
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].

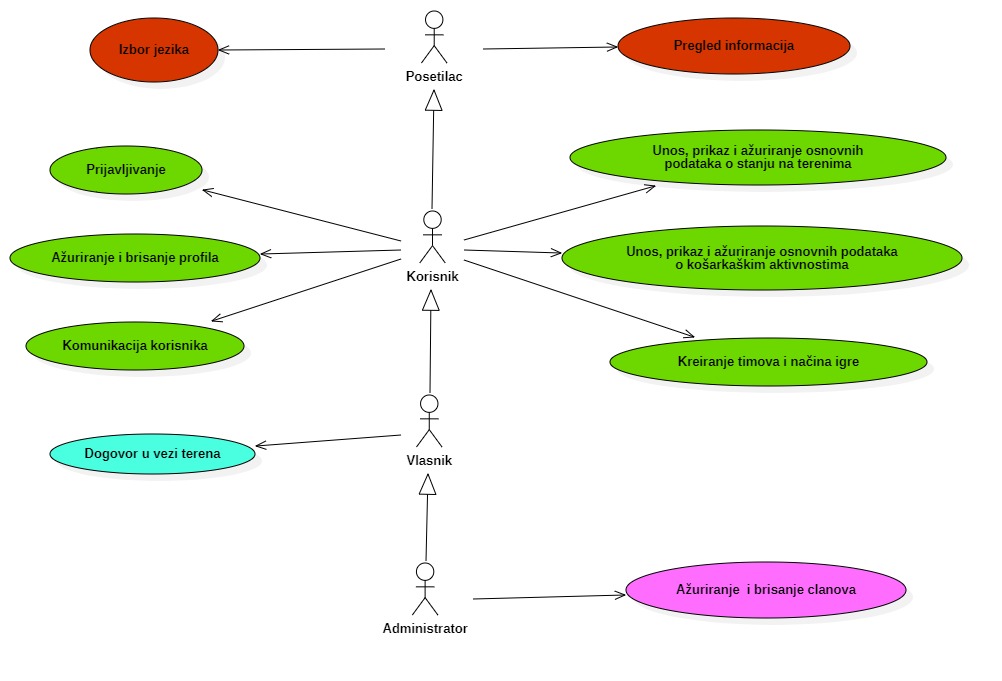
Slučajevi korišćenja ***Igraj košarku*** aplikacije su:

* Izbor jezika
* Pregled podataka o košarkaškim aktivnostima
* Pregled podataka o specijalnim košarkaškim događajima
* Pregled podataka o stanju na terenima
* Pregled opisa stanja terena
* Pregled komentara igrača
* Pregled vremena održavanja
* *Pregled učesnika*
  + Pregled timova
  + Pregled broja igrača
* Pregled pravila igre
* Pregled lokacije terena
* Pregled adrese
* Pregled mape
* *Prijavljivanje*
  + Registracija
  + Kreiranje naloga
  + Logovanje
* *Unos, prikaz i ažuriranje osnovnih podataka o stanju na terenima*
  + Unos stanja terena
  + Prikazivanje stanja terena
  + Ažuriranje stanja terena
  + Ostavljanje komentara/ocene
* *Unos, prikaz i ažuriranje osnovnih podataka o košarkaškim aktivnostima*
  + Unos košarkaške aktivnosti
  + Prikazivanje košarkaške aktivnosti
  + Ažuriranje košarkaške aktivnosti
  + Ostavljanje komentara/ocene
* Ažuriranje i brisanje profila korisnika
* Ažuriranje i brisanje članova
* Kreiranje tima
* Dogovor u vezi terena
* Komunikacija korisnika

Ove slučajevi korišćenja mogu da iniciraju **Posetilac portala**, **Prijavljeni korisnik**, **Vlasnik terena**, **Administrator**.

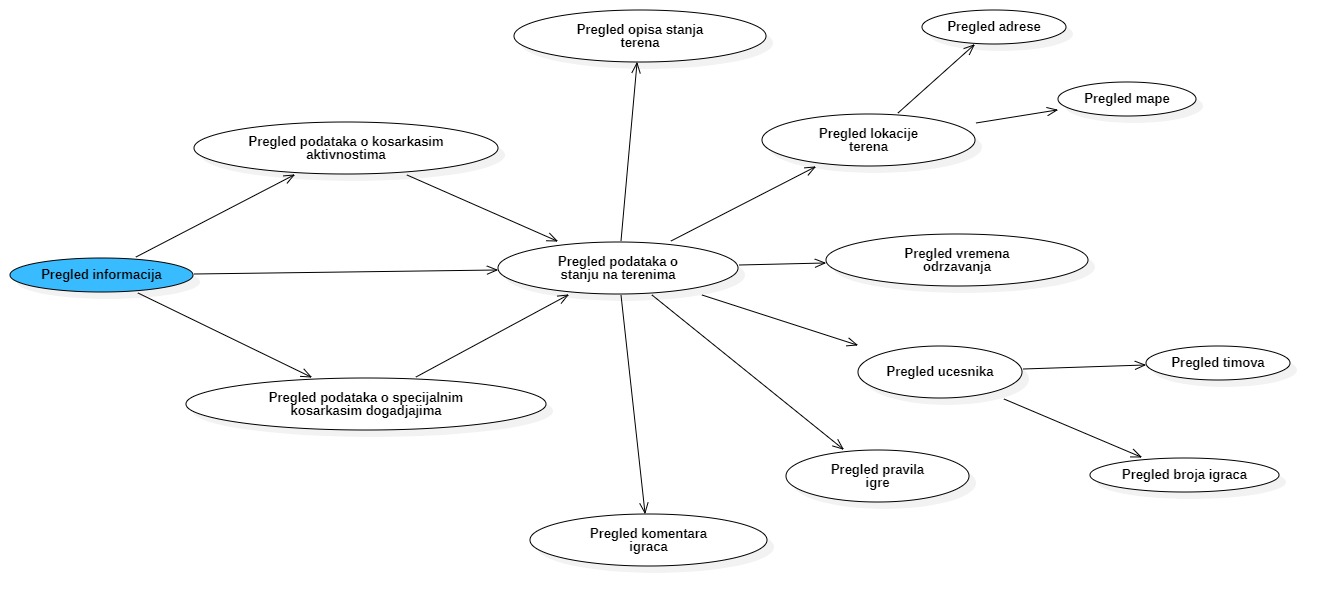
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja Igraj košarku Web aplikacije prikazana je na sledećoj slici:

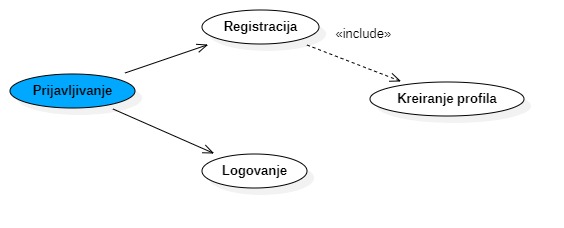


Slučajevi korišćenja „*Unos, prikaz i ažuriranje osnovnih podataka o stanju na terenima“, „Prijavljivanje“, „Pregled informacija“ i* „*Unos, prikaz, ažuriranje o košarkaškim aktivnostima“* obuhvataju složenije radnje koje se mogu dalje razložiti na pojedinačne slučajeve korišćenja.

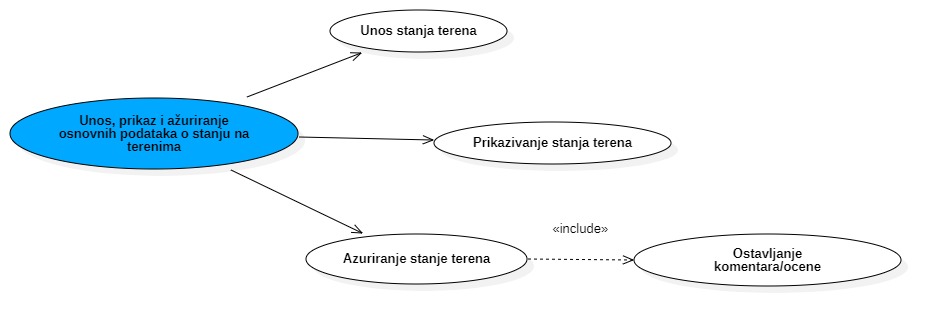
Detaljni UML dijagram za slučaj korišćenja *Pregled informacija* je prikazan na sledećoj slici:



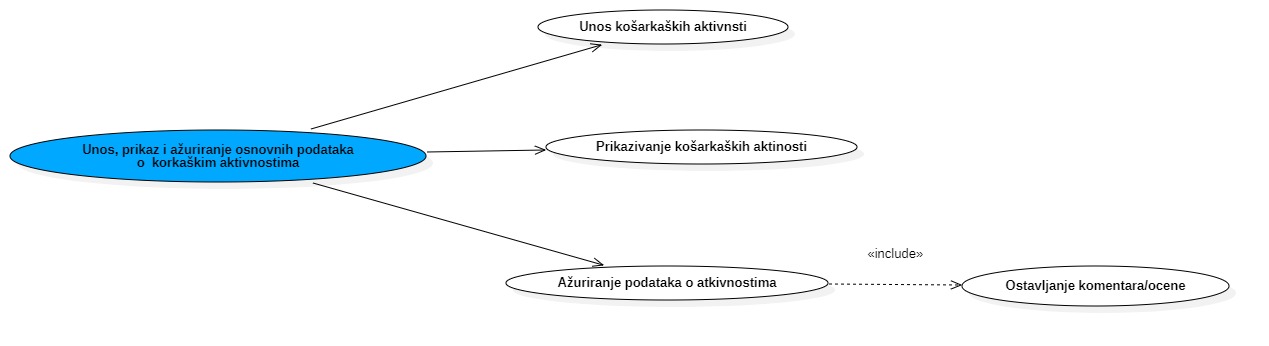
Detaljni UML dijagram za slučaj korišćenja *Prijavljivanja* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *Unos, prikaz i ažuriranje osnovnih podataka o stanju na terenima* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *Unos, prikaz i ažuriranje osnovnih podataka o košarkaškim aktivnostima* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Izbor jezika

Kratak opis: Izbor između srpskog i engleskog jezika za prikaz informacija na sajtu.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled podataka o košarkaškim aktivnostima

Kratak opis: Prikaz stranice sa informacijama o košarkaškim aktivnostima.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled podataka o specijalnim košarkaškim događajima

Kratak opis: Prikaz stranice sa informacijama o specijalnim košarkaškim događajima.

Akteri koji iniciraju slučaj korišćenja Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled podataka o stanju na terenima

Kratak opis: Prikaz stranice sa informacijama o košarkaškim terenima.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled opisa stanja terena

Kratak opis: Prikaz stranice sa informacijama o košarkaškim terenima sa njihovim opisom.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled komentara igrača

Kratak opis: Prikaz stranice sa informacijama o komentarima igrača za stanje na terenu.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled vremena održavanja

Kratak opis Prikaz stranice sa informacijama o timovima na terenu.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled timova

Kratak opis: Prikaz stranica sa spiskom projekata koji se realizuju u okviru laboratorije.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled broja igrača

Kratak opis: Prikaz stranice sa informacijama o broju igrača na terenu.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled pravila igre

Kratak opis: Prikaz stranice sa informacijama o pravilima igre za to košarkaško dešavanje.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled lokacije terena

Kratak opis: Prikaz stranice sa informacijama o adresi za to košarkaško dešavanje.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Pregled mape

Kratak opis Prikaz stranice sa informacijama o mapi za to košarkaško dešavanje.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Korisnik sajta, Vlasnik terena, Administrator.

### Registracija

Kratak opis Registracija na sajt.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta.

### Kreiranje naloga

Kratak opis: Kreira se korisnički nalog.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta.

### Logovanje

Kratak opis: Prijavljivanje na sajt sa postojećim nalogom.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta.

### Unos stanja terena

Kratak opis: Izmena podataka o stanju terena.

Akteri koji iniciraju slučaj korišćenja: Korisnik sajta, Vlasnik terena, Administrator.

### Prikazivanje stanja terena

Kratak opis: Prikazivanje podataka o stanju terena.

Akteri koji iniciraju slučaj korišćenja: Korisnik sajta, Vlasnik terena, Administrator.

### Ažuriranje stanja terena

Kratak opis: Ažuriranje podataka o stanju terena.

Akteri koji iniciraju slučaj korišćenja: Korisnik sajta, Vlasnik terena, Administrator.

### Ostavljanje komentara/ocene

Kratak opis: Ostavljanje podataka o stanju na terenu.

Akteri koji iniciraju slučaj korišćenja: Korisnik sajta, Vlasnik terena, Administrator.

### Unos košarkaške aktivnosti

Kratak opis: Izmena podataka o stanju aktivnosti.

Akteri koji iniciraju slučaj korišćenja: Korisnik sajta, Vlasnik terena, Administrator.

### Prikazivanje košarkaške aktivnosti

Kratak opis: Prikazivanje podataka o aktivnostima.

Akteri koji iniciraju slučaj korišćenja: Korisnik sajta, Vlasnik terena, Administrator.

### Ažuriranje košarkaške aktivnosti

Kratak opis: Ažuriranje podataka o aktivnosti.

Akteri koji iniciraju slučaj korišćenja: Korisnik sajta, Vlasnik terena, Administrator.

### Ostavljanje komentara/ocene aktivnosti

Kratak opis: Ostavljanje podataka o datoj aktivnosti.

Akteri koji iniciraju slučaj korišćenja: Korisnik sajta, Vlasnik terena, Administrator.

### Ažuriranje i brisanje profila korisnika

Kratak opis Korisnik ažurira ili briše svoj profil.

Akteri koji iniciraju slučaj korišćenja: Prijavljeni korisnik – Igrač.

### Ažuriranje i brisanje članova

Kratak opis: Administrator ažurira ili briše ostale profile.

Akteri koji iniciraju slučaj korišćenja: Administrator.

### Kreiranje tima

Kratak opis: Prikazuje se stranica za unos i kreiranje novog tima.

Akteri koji iniciraju slučaj korišćenja: Bilo koji Korisnik(član) sajta.

### Dogovor u vezi terena

Kratak opis Administrator i vlasnici se dogovaraju za iznajmljivanje terena.

Akteri koji iniciraju slučaj korišćenja: Vlasnik terena, Administrator.

### Komunikacija korisnika

Kratak opis: Korisnici putem komentara ostvaruju komunikaciju.

Akteri koji iniciraju slučaj korišćenja: Korisnici sajta, Vlasnik terena, 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.

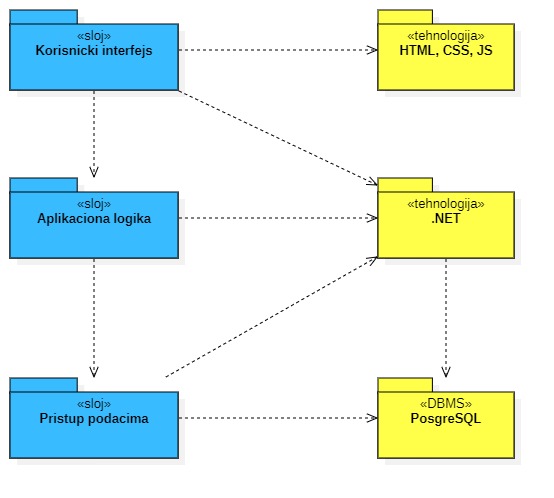
Logički pogled na ***Igraj košarku*** aplikacija obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice, .NET Core 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 .NET Core skripte zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži .NET Core 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 React.js skripte koje generišu HTML stranice preko kojih korisnici komuniciraju sa sistemom.

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

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi ***Igraj košarku*** Web portala. Sadrži .NET 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 i .NET Core paketa.

### Pristup podacima

sloj

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

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa .NET Core i PostgreSQL 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.

### CSS

tehnologija

Dok HTML definiše gradivne elemente stranica, CSS je jezik formatiranja pomoću kog se definiše izgled elemenata Web-stranice.

### JavaScript

tehnologija

JavaScript je skriptni, dinamički, netipizirani, prototipski jezik visokog nivoa, sa ECMAScript specifikacijama. JavaScript je univerzalni jezik klijentske strane. JavaScript se izvršava na klijentskoj mašini pošto se dokument učitava ili u nekom drugom trenutku (npr. kada se aktivira link).

### .NET Core

Tehnologija

.NET Core je framework koji je namenjen svim platformama za razliku od svog pretka (.NET Framework). Za realizaciju serverskog dela ćemo koristiti .NET Core u programskom jeziku C# koji je statički, tipiziran i objektno orijentisan jezik.

### PostgreSQL

DBMS

PostgreSQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju **Igraj košarku** portala.

# Pogled na procese

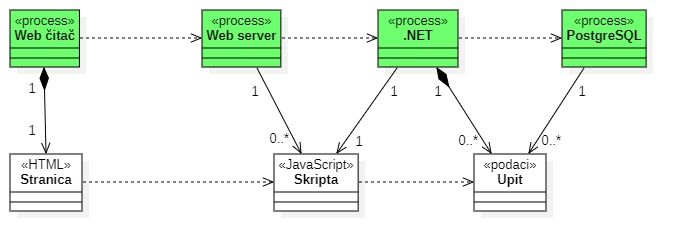
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

Web aplikacije zasnovane na .NET Core-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta 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 ***Igraj košarku*** portala kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju ***Igraj košarku*** portala. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na .NET Core-u i PostgreSQL bazi podataka.



### Web čitač

Web čitač je proces koji izvršava funkcionalnost aplikacije za prikaz HTML stranica dobijenih od nekog Web servera. U najopštijem slučaju Web čitač u jednom trenutku može da prikazuje samo jednu HTML stranicu.

Web čitač zavisi od Web servera koji generiše i vraća odgovarajuću HTML stranicu na zahtev.

Po uništavanju procesa Web čitač, uništava se i prikazana HTML stranica.

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više Web čitača. Ukoliko je zahtevana stranica JavaScript, Web server inicira izvršenje .NET Core procesa koji obrađuju pristigli upit i generiše sadržaj koji se vraća Web čitaču. Web server može paralelno da inicira veći broj .NET procesa.

### .NET Core

.NET Core je proces koji u ovom delu vrsi manipulaciju sa podacima. Pribavlja ili upisuje u bazu podataka podatke u zavisnosti od dobijenog zahteva. Pristupi bazi podataka mogu da se izvršavaju konkurentno, imajući u vidu da baza podataka može konkurentno da prihvata određeni broj upita.

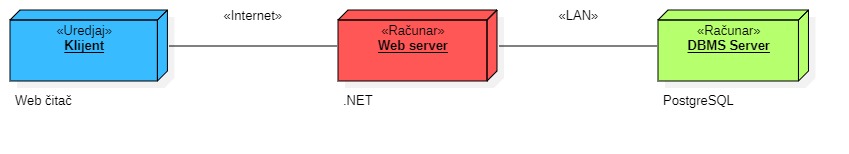
### PostgreSQL Server

PostgreSQL Server je proces koji izvršava funkcionalnost PostgreSQL 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 ***Igraj košarku*** portala.



## Klijent

Pristup ***Igraj košarku*** portal se obavlja preko klijentskih računara 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 .NET Core koji vrše obradu zadatih JavaScript skripti. U najopštijoj konfiguraciji DBMS se izvršava na posebnoj mašini koja je sa Web serverom u lokalnoj mreži (LAN).

## DBMS server

DBMS server je računar na kome se izvršava PostgreSQL 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 iz lokalne mreže (LAN).

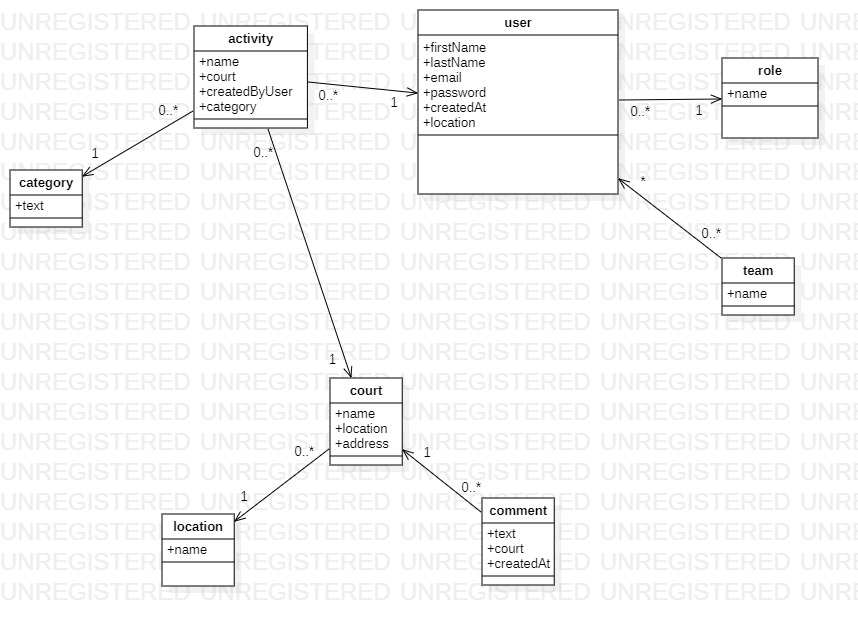
# Pogled na implementaciju sistema

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

## Model domena

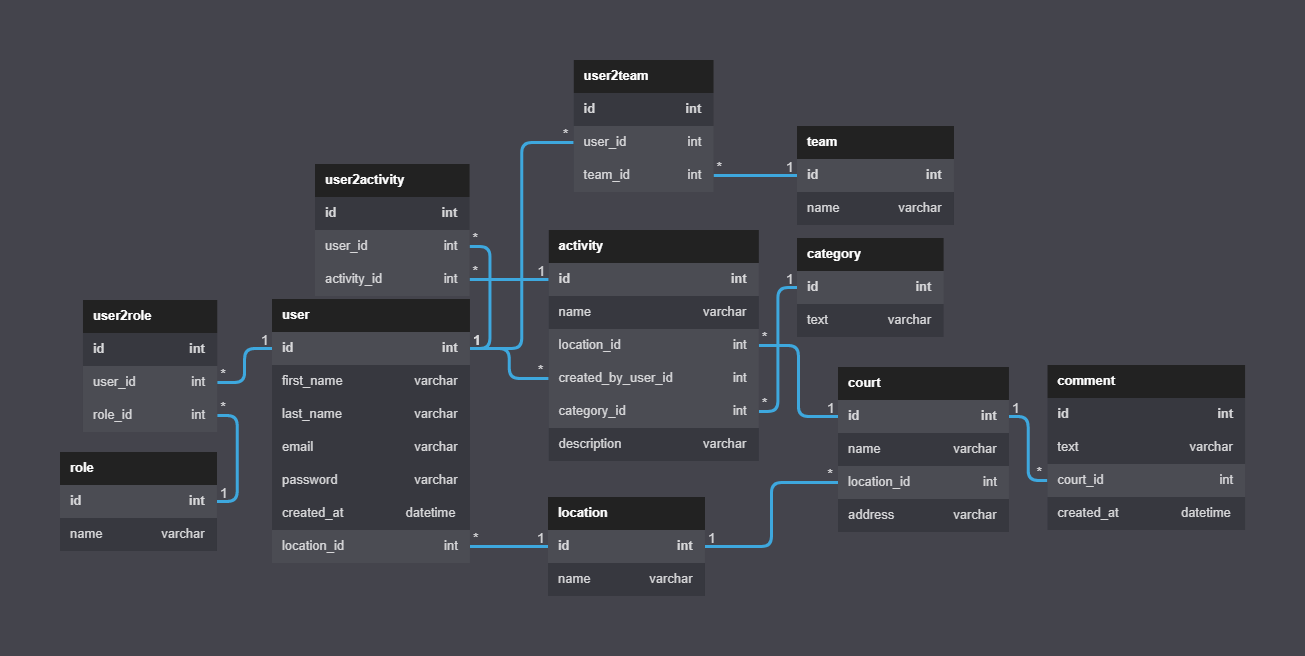
Model domena za koji se ***Igraj košarku*** Web portal projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

Model domena predstavlja osnovu za projektovanje baze podataka, ali i identifikaciju nekih od komponenti koje su u planu za implementaciju.



## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem *dbdiagram.io*, migracija na PostgreSQL obavljena pomoću *export opcije na istom sajtu*.

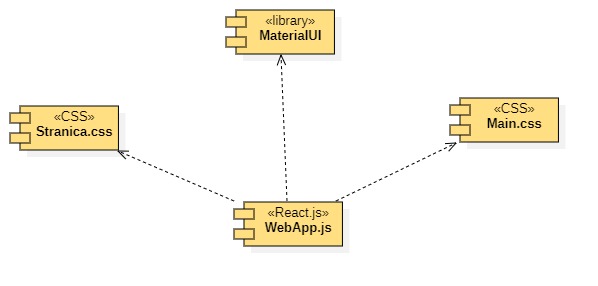


## Komponente sistema

### Komponente sistema Igraj košarku portala će biti date po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je prikazan kao klasa atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije.

### Komponente korisničkog interfejsa

Dizajn korisničkog interfejsa je obuhvaćen dvema komponentama:



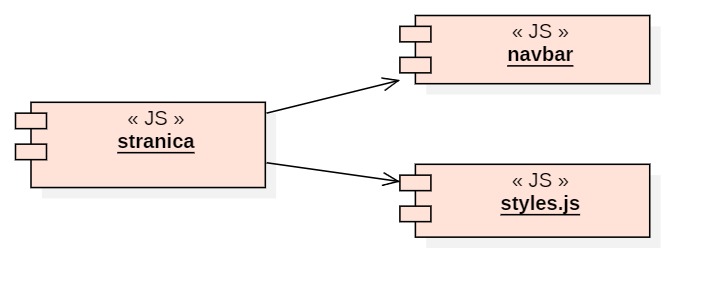
Komponenta **Stranica.css** predstavlja generalizaciju CSS fajlova koji se koriste za dodatno stilizovanje pojedinačnih dinamičkih HTML stranica.

Komponenta **WebApp.js** predstavlja centar sistema i koristi se za generisanje dinamičkih HTML stranica i prosleđivanje već kreiranih statičkih front end datoteka.

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

Komponenta **Material-UI** predstavlja eksternu biblioteku za stilizovanje HTML elemenata i koristi se za kreiranje modernog i responzivnog korisničkog interfejsa. Importuje se u svaki HTML fajl.

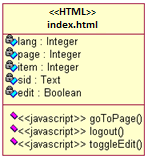
Za jedinstven dizajn u celokupnom sistemu koriste se i takozvane parcijalne dinamičke JS strane, kao i parcijalni CSS stilovi. Ove parcijalne komponente su prikazane na sledećem dijagramu komponenti.



Komponenta **navbar** predstavlja navigacioni meni sajta i na svakoj strani se prikazuje isto.

Komponenta **styles.js** predstavlja stilizovanje nekih elemenata koje bi trebalo prikazati na isti način na različitim stranicama.

Parametri koji utiču na izbor i jezik za prikaz stranice ilustrovani su sledećim dijagramom klasa:

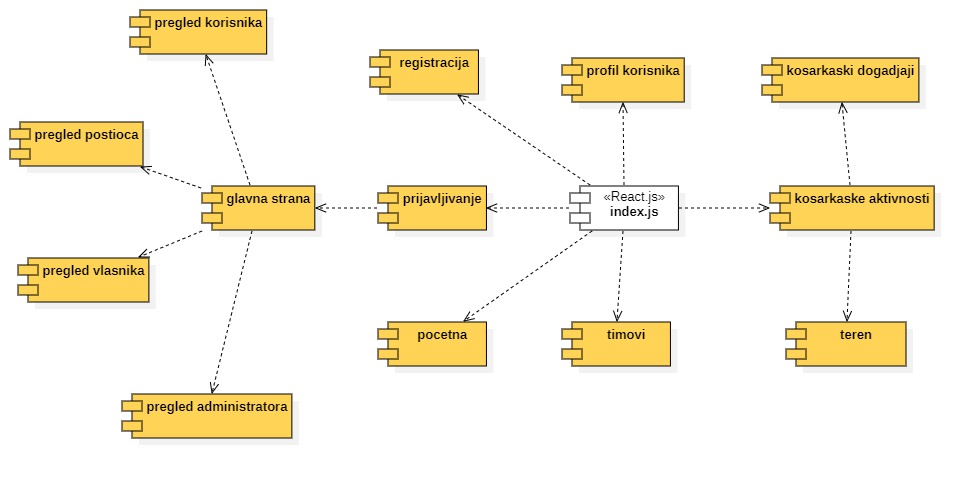


Značenje atributa je sledeće:

* lang – identifikator jezika (JID iz tabele Jezik)
* page – interni identifikator stranice (o laboratoriji, članovi, publikacije, projekti)
* item – identifikator stavke koja se detaljno prikazuje na stranici (određeni član ili projekat)
* sid – identifikator sesije kada je korisnik ulogovan
* edit – definiše da li je uključeno uređivanje

### Komponente aplikacione logike

Komponente koje realizuju domen problema se uključuju isključivo preko **index.html** 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:

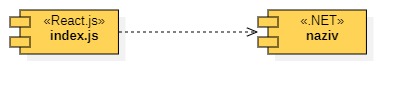


Kratak pregled funkcija svake komponente:

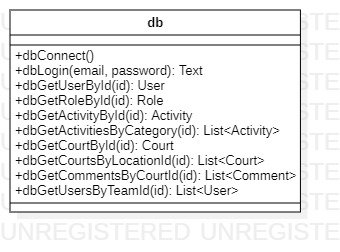
* Početna – početna stranica sajta; prikaz osnovnih informacija o novostima, grupama i oglasima
* registracija – stranica za kreiranje profila korisnika
* profil korisnika – stranica za prikaz podataka o ulogovanom korisniku
* prijavljivanje – prijavljivanje na sajt
* pregled vlasnika/korisnika/administratora/posetioca - različti pregled stranice za različite korisnike
* timovi – pregled timova
* košarkaški događaji/aktivnosti

### Komponente za pristup podacima

Pristup NoSql bazi podataka je omogućen pomoću funkcija koje su definisane u okviru eksternog .Net modula. Povezivanje sa bazom se obavlja u okviru index.js fajla. Kreirane su i skripte <naziv>.js koje se koristi za čuvanje podataka u bazi, a PostgreSQL i <name>.js skripte se uključuju na početku index.js, tako da su sve funkcije za pristup podacima i komunikaciju sa bazom dostupne svim kreiranim komponentama.



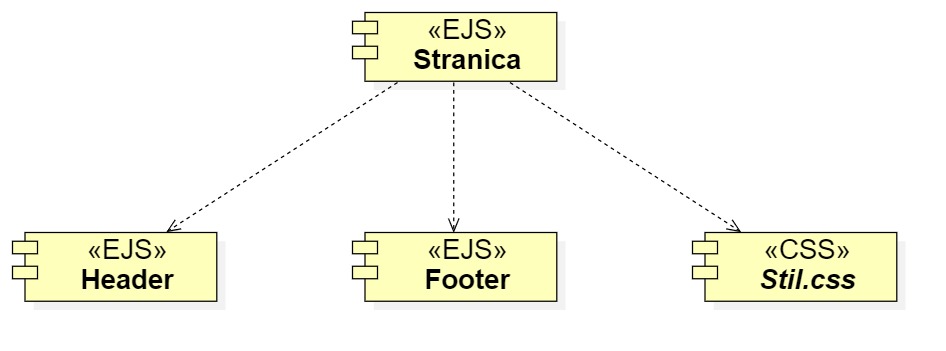
Na sledećem UML dijagramu klasa pobrojane su funkcije za pristup podacima iz baze:



Navedene funkcije obavljaju sledeće zadatke:

* dbGetUserById vraca korisnika koji ima prosledjeni id
* dbGetRoleById vraca rolu koja ima prosledjeni iddbGet – vraća vrednost polja iz rezultata upita
* dbGetActivityById vraca activity koji ima prosledjeni iddbLogin – loguje korisnika i vraća SID
* dbGetActivitiesByCategory vraca listu aktvnosti koje imaju kategoriju sa prosledjenim id-jem
* dbGetCourtById vraca teren/halu sa prosledjenim id-jem
* dbGetCourtsByLocationId vraca listu igralista sa prosledjenim id-jem
* dbLogin – loguje korisnika i vraća SID
* dbConnect – povezivanje na bazu podataka, poziva se u samom skriptu
* dbGetCommentsByCourtId vraca listu komentara vezanih za igraliste sa prosledjenim id-jem
* dbLogin – loguje korisnika i vraća SID
* dbGetUsersByTeamId vraca listu korisnika koji su clanovi tima ciji je id prosledjen

Za jedinstven dizajn u celokupnom sistemu koriste se i takozvane parcijalne dinamičke EJS strane, kao i parcijalni CSS stilovi. Ove parcijalne komponente su prikazane na sledećem dijagramu komponenti.



Komponenta **Header** sadrži navigacione elemente sajta i sadrži minimalne razlike u izgledu u zavisnosti od vrste profila kojim se pristupa sajtu.

Komponenta **Footer** predstavlja zaglavlje i prikazuje se isto na svakoj strani.

Komponenta **Stil.css** sadrži css kod za stilizovanje elemenata strana koje imaju isti izgled a prikazuju se na više različitih stranica.

# 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 2000 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. ***Igraj košarku*** portal će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 12%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 100 sati.