WAZAP

Web aplikacija za akva park

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 17.04.2024 | 1.0 | Inicijalna verzija | Sava Mihajlović, Nemanja Miloičić, Lazar Trifunović |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 akva parku 8

6.2.2 Dodavanje korisnika 8

6.2.3 Brisanje korisnika 8

6.2.4 Ažuriranje korisnika 8

6.2.5 Pregled osnovnih informacija o korisniku 8

6.2.6 Obrada zahteva 8

6.2.7 Kreiranje zahteva za izdavanje karte 8

6.2.8 Prijavljivanje i registracija 8

6.2.9 Ažuriranje stanja ležaljki 8

6.2.10 Podnošenje zahteva za sezonski posao 8

7. Pogled na logičku arhitekturu sistema 8

7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve 9

7.1.1 Korisnički interfejs 9

7.1.2 Aplikaciona logika 9

7.1.3 Pristup podacima 9

7.1.4 HTML, CSS i ReactJS 9

7.1.5 C# .NET 10

7.1.6 Oracle 10

8. Pogled na procese 10

8.1 Procesi 10

8.1.1 Web browser 10

8.1.2 Web server 10

8.1.3 C# Controllers 10

8.1.4 Oracle Server 11

9. Pogled na raspoređivanje sistema 11

9.1 Klijent 11

9.2 Web server 11

9.3 DBMS server 11

10. Pogled na implementaciju sistema 11

10.1 Model domena 11

10.2 Šema baze podataka 12

10.3 Komponente sistema 13

10.3.1 Komponente korisničkog interfejsa 13

10.3.2 Komponente aplikacione logike 14

10.3.3 Komponente za pristup podacima 14

11. Performanse 15

12. Kvalitet 16

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture WAZAP web aplikacije.

# Opseg dokumenta

Dokument se odnosi na WAZAP web aplikaciju koja će biti razvijena od strane Solution4 tima. WAZAP predstavlja skraćenicu za Web aplikacija za akva park. Namena sistema je efikasno prezentovanje, pružanje usluga vezanih za rad samog sistema kao i njegovo održavanje.

# Reference

Spisak korišćene literature:

* 1. WAZAP – Predlog projekta, DO1\_Predlog\_Projekta, V1.0, 2024, Solution4.
  2. WAZAP – Vizija sistema, DO2\_Vizija\_Sistema V1.0, 2024, Solution4.
  3. WAZAP – Plan realizacije projekta, DO3\_Plan\_realizacije V1.0, 2024, Solution4.
  4. WAZAP – Specifikacija zahteva, D04\_Spec\_Zahteva V1.0, 2024, Solution4.

# 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. WAZAP će biti implementiran kao Web aplikacija zasnovana na .NET C# i MySQL bazi podataka..
2. Klijentski deo WAZAP aplikacije će biti optimizovan za sledeće Web čitače: Internet Explorer 6.0 i noviji, Opera 8.0 i noviji, Google Chrome kao i Firefox (Mozilla).
3. Svi zahtevi u pogledu performansi dati u [4] 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 [4].

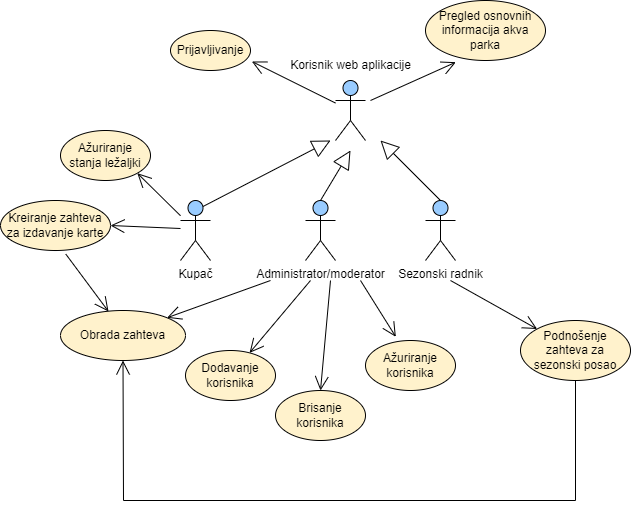
Slučajevi korišćenja WAZAP aplikacije su:

* *Pregled osnovnih informacija*
  + Pregled osnovnih informacija o akva parku
  + *Prijavljivanje i registracija*
  + Prijavljivanje na web aplikaciju
  + *Pregled podataka o članovima*
    - Dodavanje korisnika
    - Brisanje korisnika
    - Ažuriranje korisnika
    - Pregled osnovnih informacija o korisniku
  + *Pregled podataka o uslugama*
    - Obrada zahteva
    - Kreiranje zahteva za izdavanje karte
    - Ažuriranje stanja ležaljki
    - Podnošenje zahteva za sezonski posao

Ove slučajeve korišćenja mogu da iniciraju kupači, sezonski radnici i administratori/moderatori.

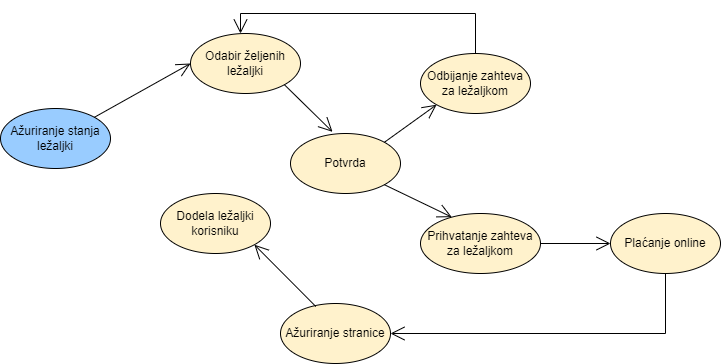
## Dijagrami slučajeva korišćenja

Osnovni UML dijagram koji prikazuje korisnike i slučajeve korišćenja WAZAP aplikacije prikazan je na sledećoj slici:

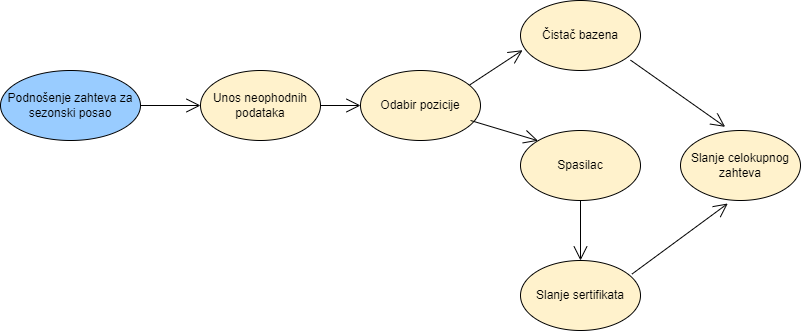


Slučajevi korišćenja *Ažuriranje stanja ležaljki, Podnošenje zahteva za sezonski posao* i *Obrada zahteva* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

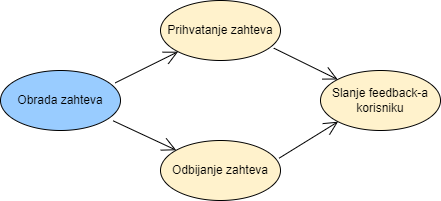
Detaljni UML dijagram za slučaj korišćenja *Ažuriranje stanja ležaljki* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *Podnošenje zahteva za sezonski posao* je prikazan na sledećoj slici:



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



## 6.2 Kratak opis slučajeva korišćenja

### 6.2.1 Pregled osnovnih podataka o akva parku

Kratak opis: Korisnicima je po ulasku na web aplikaciju omogućen pregled osnovnih informacija o akva parku.

Akteri su korisnici web aplikacije (kupači, administratori/moderatori i sezonski radnici).

### 6.2.2 Dodavanje korisnika

Kratak opis: Administrator/moderator ima mogućnost da doda korisnike i da tim korisnicima dodeli zahtevane privilegije. Akteri su administrator/moderator.

### 6.2.3 Brisanje korisnika

Kratak opis: Administrator/moderator ima mogućnost da ukloni korisnike sa web aplikacije. Akteri su administrator/moderator.

### 6.2.4 Ažuriranje korisnika

Kratak opis: Administrator/moderator ima mogućnost izmene podataka o korisnicima. Akteri su

administrator/moderator.

### 6.2.5 Pregled osnovnih informacija o korisniku

Kratak opis: Prikaz tabele sa korisnicima, informacije o njima i njihovim zahtevima. Akteri su administrator/moderator.

### 6.2.6 Obrada zahteva

Kratak opis: Dužnost administratora/moderatora jeste da obradi zahteve drugih profila korisnika, proveri njihovu validnost, eventualno ih odbije ili ih prihvati. Akteri su pošiljalac zahteva (kupač ili sezonski radnik) i administrator/moderator.

### 6.2.7 Kreiranje zahteva za izdavanje karte

Kratak opis: Korisnik ima mogućnost da zahteva od web aplikacije izdavanje mesečnih/polumesečnih karata. Akteri su kupači.

### 6.2.8 Prijavljivanje i registracija

Kratak opis: Prijavljivanje korisnika na web aplikaciju u cilju pristupa specifičnim funkcijama koje zahtevaju autorizaciju.

Akteri su kupač ili sezonski radnik i administrator/moderator.

### 6.2.9 Ažuriranje stanja ležaljki

Kratak opis: Kupači imaju mogućnost rezervisanja željenih ležaljki unapred. Akteri su kupači.

### 6.2.10 Podnošenje zahteva za sezonski posao

Kratak opis: Zainteresovani sezonski radnici imaju mogućnost podnošenja zahteva za posao. Akteri su sezonski radnik , administrator/moderator.

# 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.

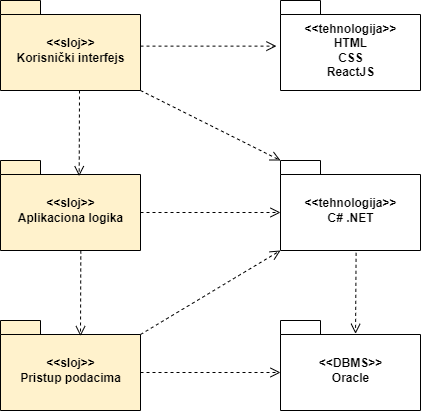
Logički pogled na WAZAP aplikaciju obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice, JS 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 API pisan u C# .NET framework-u zadužen za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži .NET Entity Framework koji predstavlja interfejs za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi podataka. Baza koja se koristi za perzistenciju podataka je Oracle.

## 7.1 Pregled arhitekture – organizacija paketa i podsistema u slojeve



### 7.1.1 Korisnički interfejs

Ovaj sloj realizuje korisnički interfejs WAZAP - a. U njemu su sadržane sve HTML stranice, CSS datoteke i ReactJS skripte koje generišu HTML stranice preko kojih korisnici komuniciraju sa sistemom.

Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i paketa C# .NET i paketa tehnologija HTML,CSS i ReactJS.

### 7.1.2 Aplikaciona logika

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi WAZAP - a . Sadrži API pisan u C# .NET framework-u koji realizuje funkcionalnosti karakteristične za domen primene web aplikacije i uspostavlja vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima i C# .NET paketa.

### 7.1.3 Pristup podacima

Sloj za pristup podacima se nalazi na dnu troslojne arhitekture i sadrži C# .NET zadužen za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u Oracle bazi podataka.

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od C# .NET paketa i Oracle baze podataka.

### 7.1.4 HTML, CSS i ReactJS

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.

Tehnologija CSS-a stilizuje gradivne elemente HTML – a doprinoseći boljem korisničkom iskustvu i korisničkom interfejsu.

Tehnologija ReactJS – a omogućava dinamičko dodeljivanje sadržaja stranici. Koristi se u svrhu reaktivnog programiranja.

### 7.1.5 C# .NET

Tehnologija C# .NET obezbeđuje mehanizam za pisanje i izvršavanje funkcija API - a na strani servera.

Njegov sastavni deo je Entity Framework koji predstavlja ORM alat odnosno apstrakciju za komunikaciju između servera i baze zasnovan na model-first pristupu.

### 7.1.6 Oracle

Oracle predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju WAZAP-a.

# Pogled na procese

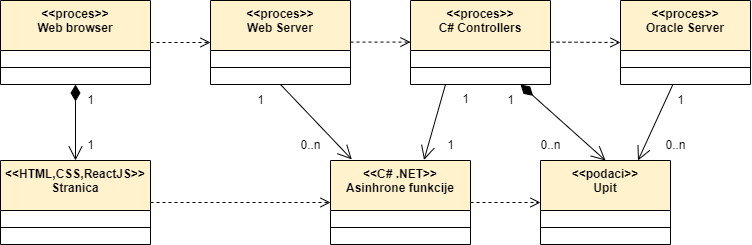
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

Web aplikacije zasnovane na C# .NET-u imaju relativno jednostavan procesni model koji je u potpunosti pod kontrolom Web servera. Sa stanovišta projektanta C# .NET web aplikacije nije potrebno voditi računa o načinu rada Web servera.

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje WAZAP - a kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju WAZAP aplikacije. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na C# .NET - u i Oracle bazi podataka.



### 8.1.1 Web browser

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

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

### Web server

Web server se komunicira sa kotrolerima čiji API čine asinhrone funkcije koje implementiraju ponašanje same web aplikacije u zavisnosti od asinhronih događaja na samoj stranici. Web server prihvata i opslužuje HTTP zahteve klijenata koje prosleđuju preko web browsera.

### C# Controllers

U kontekstu web razvoja, C# kontroleri su klase koje upravljaju logikom poslovne logike web aplikacije. Kontroleri prihvataju HTTP zahteve od klijenata (web browsera) i obrađuju ih kako bi generisali odgovarajući HTTP odgovor.

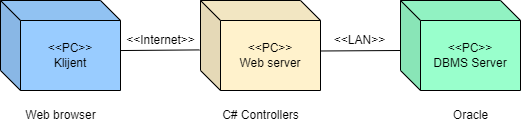
### 8.1.4 Oracle Server

Oracle server omogućava razvoj aplikacija koje koriste podatke iz baze putem standardnih jezika poput SQL-a ili kroz različite programerske interfejse. Takođe pruža alate za administraciju baze podataka, kao što su upiti za izradu izveštaja i analize podataka, kao i mehanizme za održavanje i sigurnost baze podataka.

# 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 WAZAP aplikacije.



## Klijent

Pristup WAZAP aplikaciji se obavlja preko klijentskih računara na kojima se izvršava Web browser. Za povezivanje između klijenta i Web servera koristi se Internet infrastruktura tako da nema ograničenja u pogledu lokacije klijenta. Takođe je za povezivanje sa serverom potrebna aktivna Internet konekcija.

## 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 se izvršava API u pogledu asinhronih funkcija C# .NET kontrolera. 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 Oracle 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).

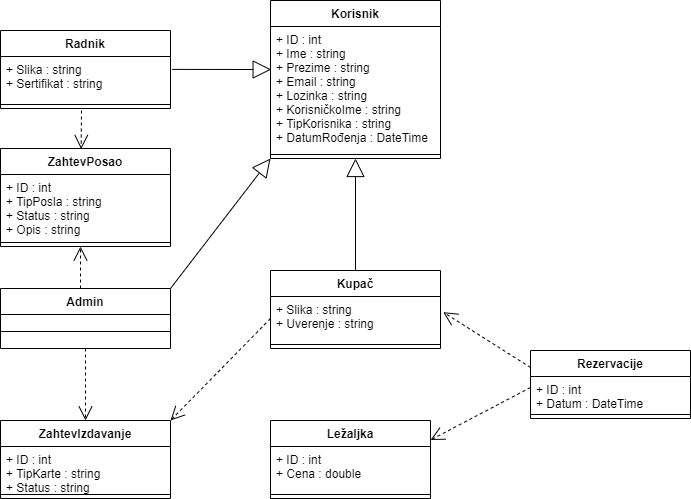
# Pogled na implementaciju sistema

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

## Model domena

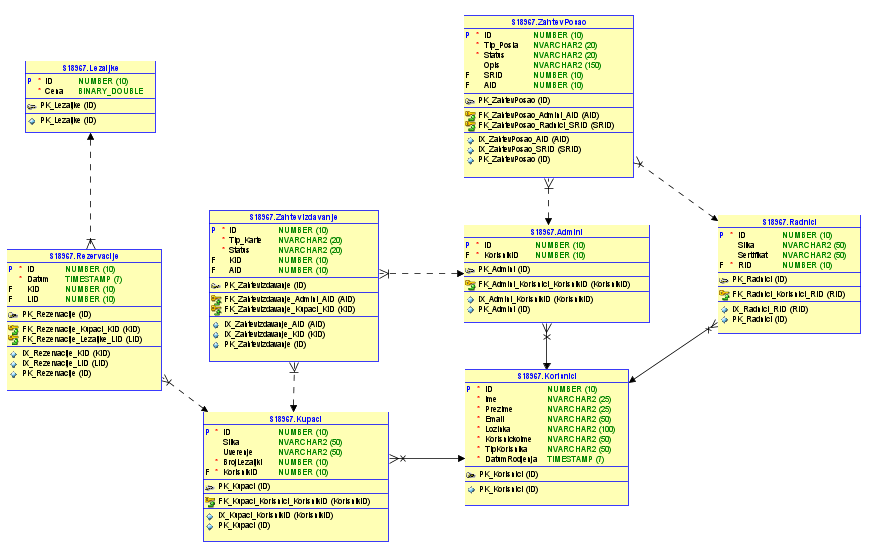
Model domena za koji se WAZAP aplikacija projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

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



## Šema baze podataka

Detaljna šema baze podataka je prikazana na sledećem dijagramu. Baza podataka i dijagram su kreirani korišćenjem *MS Access*-a, dok je migracija na Oracle obavljena pomoću *Entity Framework – a.*



## Komponente sistema

Komponente sistema WAZAP se sastoje iz 3 dela:

* Komponente korisničkog interfejsa
* Komponente aplikacione logike
* Komponente za pristup podacima

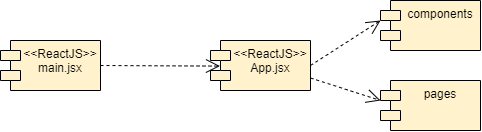
**Komponente korisničkog interfejsa** su implementirane u ReactJS – u zarad dinamičkog ažuriranja sadržaja naše web aplikacije.

**Komponente aplikacione logike** implementirane u .NET Core Framework – u omogućava model-first pristup dizajniranja baze s kojom će komponente korisničkog interfejsa da interaguju.

**Komponente za pristup podacima** su takođe implementirane u .NET - u kroz kontrolere i njihovih respektivnih funkcija.

### Komponente korisničkog interfejsa

Dizajn korisničkog interfejsa je obuhvaćen dvema komponentama:

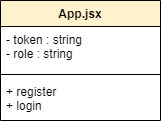


Korisnički interfejs WAZAP aplikacije je baziran na SPA u ReactJS – u. Inicijalno renderovanje će se odvijati u **main.jsx** skripti, a svaki re-render u **App.jsx** skripti.

**Main.jsx** instancira inicijalnu aplikaciju, odnosno poziva **App.jsx** skriptu na izvršenje.

**App.jsx** se sastoji od više ruta koje će biti linkabilne i predstavljaće vezu ka ostalim stranicama.

U direktorijumu **components** se nalaze konkretne komponente korisničkog interfejsa (UI – a) koje će predstavljati sastavni deo same SPA aplikacije. Njihova stilizacija se nalazi u zasebnim module.css fajlovima.

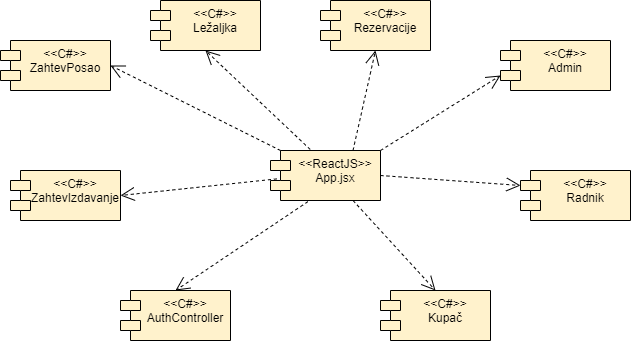


Značenje atributa je sledeće:

* token – predstavlja jedinstvenu sekvencu znakova neophodnu za čuvanje informacija korisnika u okviru browser-a.
* role – predstavlja vrstu korisnika odnosno njegovu autorizaciju u okviru same aplikacije.

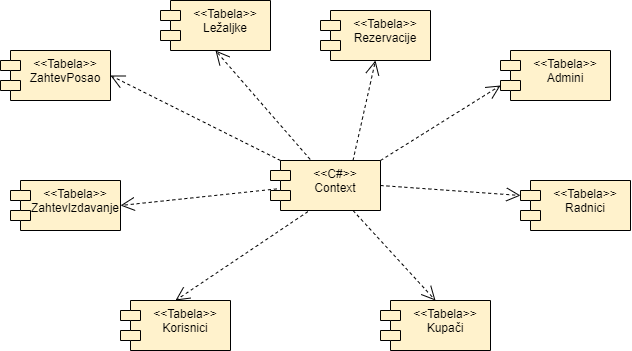
### Komponente aplikacione logike

Komponente koje realizuju domen problema se uključuju isključivo preko **App.jsx** skripte korisničkog interfejsa. Na taj način zadržavaju sva podešavanja stila definisana u ovom skriptu. Na sledećem dijagramu su prikazane komponente aplikacionog sloja i preko kojih App.jsx pribavlja i postavlja informacije:

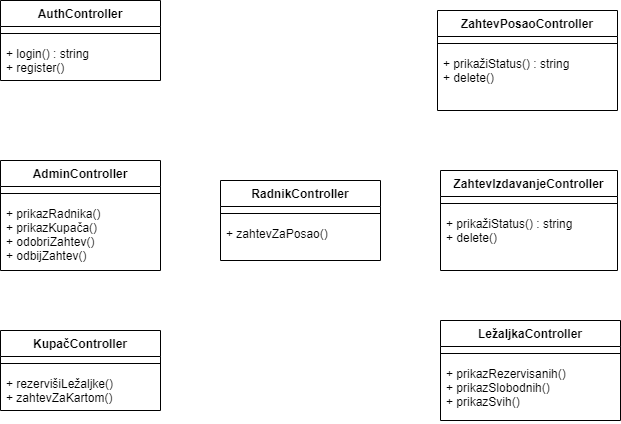


### Komponente za pristup podacima

Pristup bazi podataka obavlja se preko: **gislab-oracle.elfak.ni.ac.rs:1521/SBP\_PDB.** Šifra za pristup poznata je svima koji rade na projektu. Međutim, obraćanje bazi podataka se obavlja isključivo putem EntityFramework-a, konkretno u projektu preko **Context** klase.



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



# 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 [4]:

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 [4].

# Kvalitet

Izabrana arhitektura softvera podržava zahteve u pogledu dostupnosti i srednjeg vremena između otkaza specificirane u zahtevima u pogledu pouzdanosti [4]:

1. WAZAP će biti dostupan uvek, 24 sata dnevno, 7 dana u nedelji. Pojedine funkcionalnosti aplikacije će biti dostupne samo u okviru sezone kupanja (Jun, Jul, Avgust, Septembar).
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 120 sati.