FoodHouse

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 26.04.2022. | 1.0 | Inicijalna verzija | Jovan Maletić, 17734  Filip Marković, 17752  Lazar Najdanović, 17834  Nikola Momčilović, 17832 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 7

6.2.1 Pregled recenzije 7

6.2.2 Pregled osnovnih podataka o proizvodima 7

6.2.3 Pregled proizvoda u okviru jednog domaćinstva 8

6.2.4 Sortiranje proizvoda 8

6.2.5 Pretraživanje proizvoda 8

6.2.6 Ažuriranje podataka o proizvodima o domaćinstvu 8

6.2.7 Kupovina proizvoda 8

6.2.8 Prijavljivanje 8

6.2.9 Ažuriranje podataka o sebi 8

6.2.10 Pregled svojih prethodnih kupovina 9

6.2.11 Odobravanje novog korisnika 9

6.2.12 Brisanje postojećih korisnika 9

6.2.13 Održavanje sistema 9

6.2.14 Ažuriranje javno dostupnih informacija o domaćinstvu 9

6.2.15 Detaljniji pregled podataka o jednom proizvodu 9

6.2.16 Kreiranje i postavljanje poslova 9

6.2.17 Brisanje postojećih poslova 9

6.2.18 Ažuriranje podataka o poslu 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 HTML 10

7.1.5 CSS 11

7.1.6 Bootstrap 11

7.1.7 JavaScript + React.js 11 7.1.8 MySQL 11

8. Pogled na procese 11

8.1 Procesi 12

8.1.1 Web čitač 12

8.1.2 Web server 12

8.1.3 React 12

8.1.4 MySQL 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 13

10.1 Model domena 13

10.2 Šema baze podataka 13

10.3 Komponente sistema 14

10.3.1 Komponente korisničkog interfejsa 14

10.3.2 Komponente aplikacione logike 16

10.3.3 Komponente za pristup podacima 17

11. Performanse 18

12. Kvalitet 18

Arhitekturni projekat

# Cilj dokumenta

Cilj ovog dokumenta je specifikacija zahteva u pogledu detaljnog opisa slučajeva korišćenja FoodHouse sistema.

# Opseg dokumenta

Dokument se odnosi na FoodHouse web aplikaciju koja će biti razvijena od strane 3M&N tima. FoodHouse predstavlja online platformu na kojoj proizvođači mogu da plasiraju svoje proizvode, ponude radna mesta za određene poslove. Pored toga namenjena je korisnicima koji mogu po kategorijama da izaberu proizvode za koje su zainteresovani kao i radna mesta ukoliko ih proizvođač nudi. Takođe pruža mogućnost kupcu da lično preuzme proizvode ili preko dostave. Ukoliko kupac ne preuzima lično proizvod, dostavu će izvršiti dostavljač koji je angažovan od strane proizvođača. Namena sistema je efikasno i jednostavno dodavanje, izmena i brisanje proizvoda od strane proizvođača, efikasno zakazivanje dostave i pronalaženje poslova.

# Reference

Spisak korišćene literature:

1. FoodHouse – Predlog projekta, V1.0, 2022, 3M&N tim.
2. FoodHouse – Vizija sistema, V1.0, 2022, 3M&N tim.
3. FoodHouse – Plan realizacije projekta, V1.0, 2022, 3M&N tim.
4. FoodHouse – Planirani raspored aktivnosti na projektu, V1.0, 2022, 3M&N tim.
5. FoodHouse – Specifikacija zahteva,V1.0, 2022, 3M&N tim.

# 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. FoodHouse web aplikacija će biti implementirana kao Web aplikacija zasnovana na JavaScript jeziku, koristeći React.js platformu i MySQL.
2. Klijentski deo FoodHouse aplikacije će biti optimizovan za sledeće Web čitače: Chrome, Mozilla Firefox i Microsoft Edge [2].
3. Svi zahtevi u pogledu performansi dati u [5] moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.

# Pogled na slučajeve korišćenja

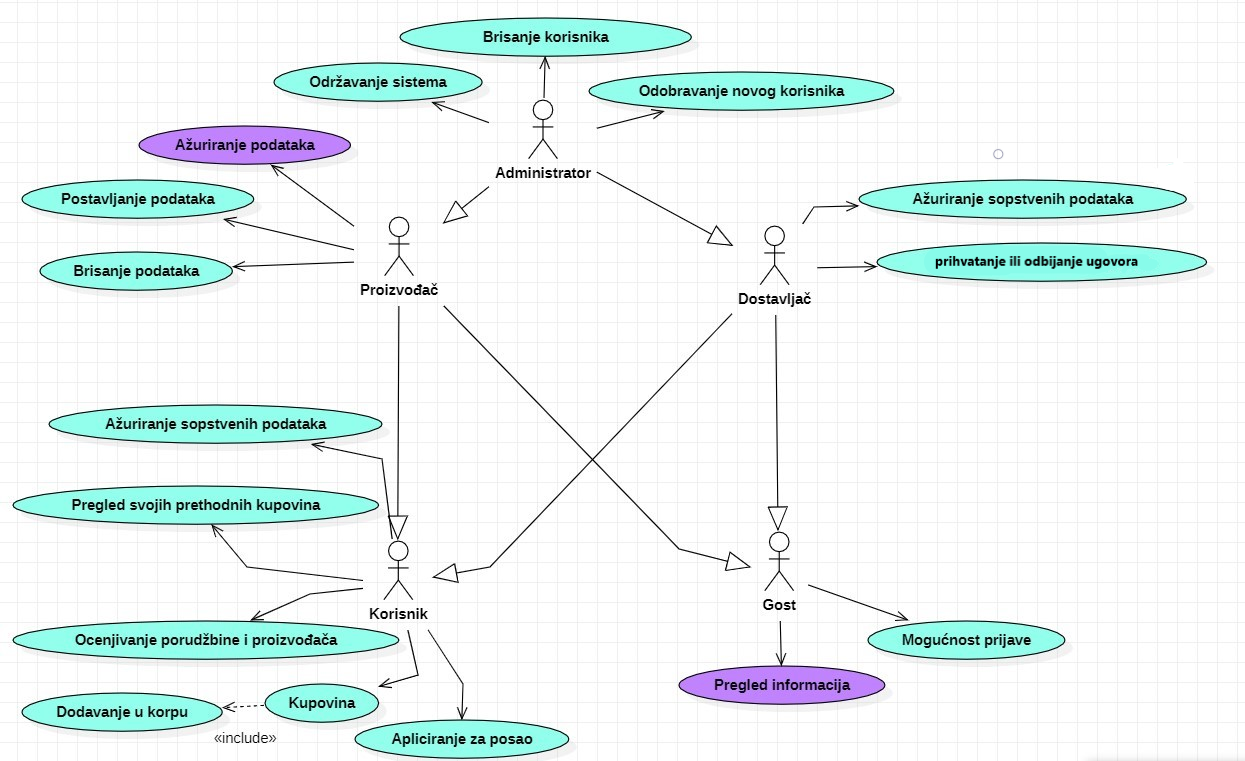
U ovom odeljku je dat pogled na slučajeve korišćenja definisane u specifikaciji zahteva [5].

Slučajevi korišćenja FoodHouse aplikacije su:

* Pregled recenzija
* Pregled osnovnih podataka o proizvodima
* Sortiranje proizvoda
  + Sortiranje po ceni
  + Sortiranje po abecedi
* Pretraživanje proizvoda
  + Pretraga po imenu
  + Pretraga po tipu proizvoda
* Ažuriranje podataka o proizvodima u domaćinstvu
  + Dodavanje novih proizvoda
  + Ažuriranje postojećih proizvoda
  + Brisanje postojećih proizvoda
* Kupovina proizvoda
* Prijavljivanje
* Ažuriranje podataka o sebi
  + Pregled svojih prethodnih kupovina
  + Odobravanje novog korisnika
  + Brisanje postojećih korisnika
  + Održavanje sistema
  + Ažuriranje javno-dostupnih informacija o domaćinstvu
  + Detaljniji pregled podataka o jednom proizvodu
  + Kreiranje i postavljanje poslova
  + Brisanje postojećih poslova
  + Ažuriranje podataka o poslu

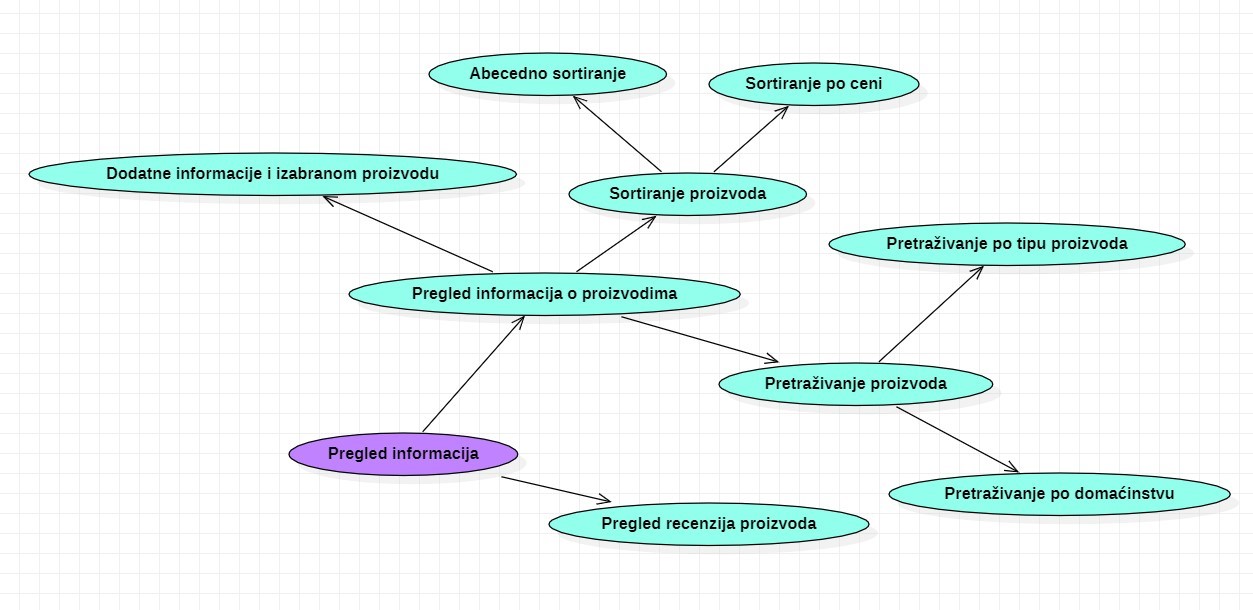
## Dijagrami slučajeva korišćenja

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

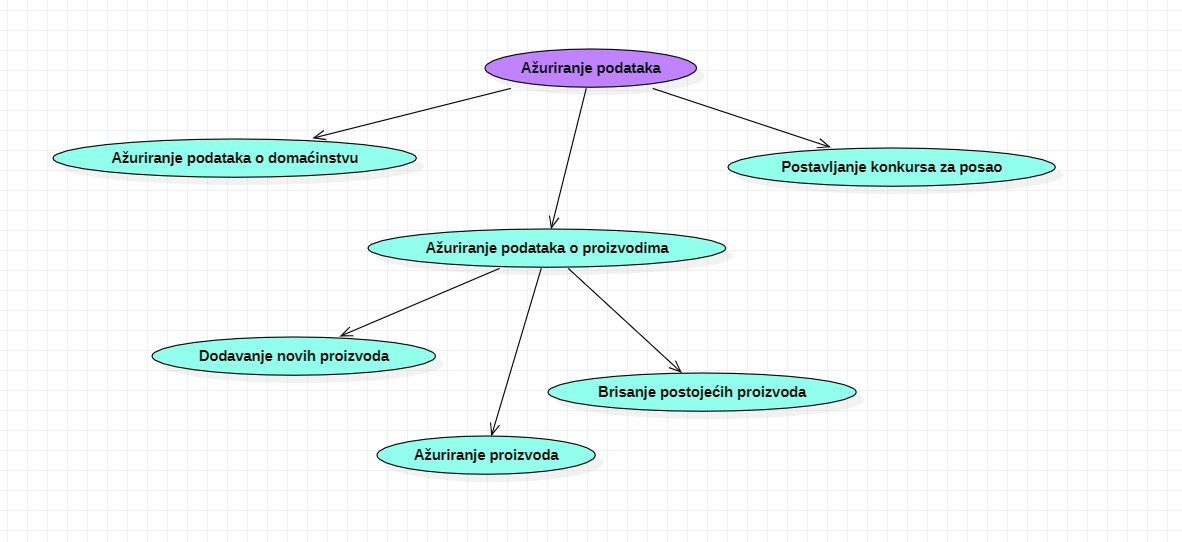


Slučajevi korišćenja *pregled informacija* i *ažuriranje podataka*  obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

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



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



## Kratak opis slučajeva korišćenja

### Pregled recenzija

Kratak opis**:**korisnik ima mogućnost pregleda postavljenih ocena.

Akteri: Gost, Proizvođač, Registrovani korisnik, Administrator.

### Pregled osnovnih podataka o proizvodima

Kratak opis: Prikaz stranice web aplikacije sa osnovnim podacima o domaćim proizvodima.

Akteri: Gost, Proizvođač, Registrovani korisnik, Administrator.

### Pregled proizvoda u okviru jednog domaćinstva

Kratak opis: Prikaz stranice domaćinstva sa spiskom proizvoda iste.

Akteri: Gost, Proizvođač, Registrovani korisnik, Administrator.

### Sortiranje proizvoda

## Sortiranje po ceni

Kratak opis: Prikaz proizvoda struktuiranih po ceni. Rastućim ili opadajućim redosledom u zavisnosti od želje samog korisnika.

Akteri: Gost, Proizvođač, Registrovani korisnik, Administrator.

* Sortiranje po abecedi

Kratak opis: Prikaz proizvoda struktuiranih po abecednom redosledu.

Akteri: Gost, Proizvođač, Registrovani korisnik, Administrator.

### Pretraživanje proizvoda

* Pretraga po imenu

Kratak opis: Prikaz stranice sa informacijama o određenom proizvodu.

Akteri: Gost, Proizvođač, Registrovani korisnik, Administrator.

* Pretraga po tipu proizvoda

Kratak opis: Prikaz stranice sa informacijama o proizvodima na osnovu tipa.

Akteri: Gost, Proizvođač, Registrovani korisnik, Administrator.

### Ažuriranje podataka o proizvodima u domaćinstvu

* Dodavanje novih proizvoda

Kratak opis: Proizvođač želi da doda nov proizvod u paletu svojih proizvoda**.**

Akteri: Proizvođač

* Ažuriranje postojećih proizvoda

Kratak opis: Korisnik želi da izmeni neku od javno dostupnih informacija o proizvodima u okviru jednog domaćinstva**.**

Akteri: Proizvođač

* Brisanje postojećih proizvoda

Kratak opis: Korisnik želi da izbriše jedan ili više proizvoda u okviru jednog domaćinstva**.**

Akteri: Proizvođač

### Kupovina proizvoda

Kratak opis: Korisniku se dodeljuje mogućnost kupovine proizvoda iz više domaćinstava.

Akteri: Registrovani korisnik.

### Prijavljivanje

Kratak opis: Prijavljivanje korisnika na FoodHouse sistem u cilju pristupa specifičnim funkcijama koje zahtevaju autorizaciju.

Akteri: Gost

### Ažuriranje podataka o sebi

### Kratak opis: Korisnik želi da izmeni neku od javno dostupnih informacija o sebi.

Akteri: Proizvođač, Dostavljač, Registrovani korisnik, Administrator.

### Pregled svojih prethodnih kupovina

Kratak opis: Korisnik ima uvid u istoriju svojih kupovina.

Akteri: Registrovani korisnik.

### Odobravanje novog korisnika

Kratak opis: Odobrenje korisničkog naloga za novog člana sistema FoodHouse.

Akteri: Administrator.

### Brisanje postojećih korisnika

Kratak opis: Brisanje korisničkog naloga i podataka za postojećeg člana FoodHouse sistema.

Akteri: Administrator.

### Održavanje sistema

Kratak opis: Administrator je dužan da obezbedi što lagodniji pristup FoodHouse sistema korisnicima.

Akteri: Administrator.

### Ažuriranje javno-dostupnih informacija o domaćinstvu

Kratak opis: Korisnik može menjati javno-dostupne informacije o svom domaćinstvu.

Akteri: Proizvođač.

### Detaljniji pregled podataka o jednom proizvodu

Kratak opis: Korisnik, osim osnovnih, ima uvid i u detaljnije podatke o pojedinačnom proizvodu.

Akteri: Gost, Registrovani korisnik, Proizvođač, Administrator.

### Kreiranje i postavljanje poslova

Kratak opis: Kreiranje novog posla i postavljanje na stranicu domaćinstva.

Akteri: Proizvođač

### Brisanje postojećih poslova

Kratak opis: Korisnik želi da izbriše posao u okviru jednog domaćinstva**.**

Akteri: Proizvođač

### Ažuriranje podataka o poslu

Kratak opis: Izmena podataka o poslu od strane proizvođača.

Akteri: Proizvođač

# 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.

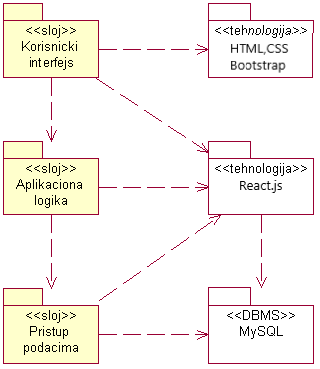
Logički pogled na FoodHouse portal obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice i React.js skripte koje realizuju grafički dizajn i forme preko kojih korisnici sistema komuniciraju sa sistemom.

Paket *Aplikaciona logika* predstavlja srednji sloj sistema koji sadrži React.js skripte zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži React.js 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 Web aplikacije. U njemu su sadržane sve HTML stranice, Bootstrap komponente, CSS stilove 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 React.js.

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi FoodHouse Web aplikacije. Sadrži React.js skripte koje realizuju funkcionalnost karakterističnu za domen primene aplikacije i uspostavljaju vezu između korisničkog interfejsa i sloja za pristup podacima.

Ovaj sloj zavisi od sloja za pristup podacima i React.js paketa.

### Pristup podacima

sloj

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

Ovaj sloj ne zavisi od drugih slojeva, ali je zavisan od paketa React.js i MySQL baze 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.

### Bootstrap

tehnologija

Bootstrap predstavlja web framework otvorenog koda za kreiranje web sajtova i web aplikacija.

Baziran je na HTML i CSS šablonima. Takođe, dizajniran je tako da podrži razvoj dinamičkih web aplikacija.

### JavaSctipt+React.js

tehnologije

JavaScript predstavlja programski jezik za programiranje klijentske strane. React.js predstavlja biblioteku napisanu u JavaScript-u koja pruža dodatne funkcionalnosti za programiranje i olakšava implementaciju korisničkih interfejsa baziranih na UI komponentama.

### MySQL

DBMS

MySQL predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju FoodHouse sistema.

# Pogled na procese

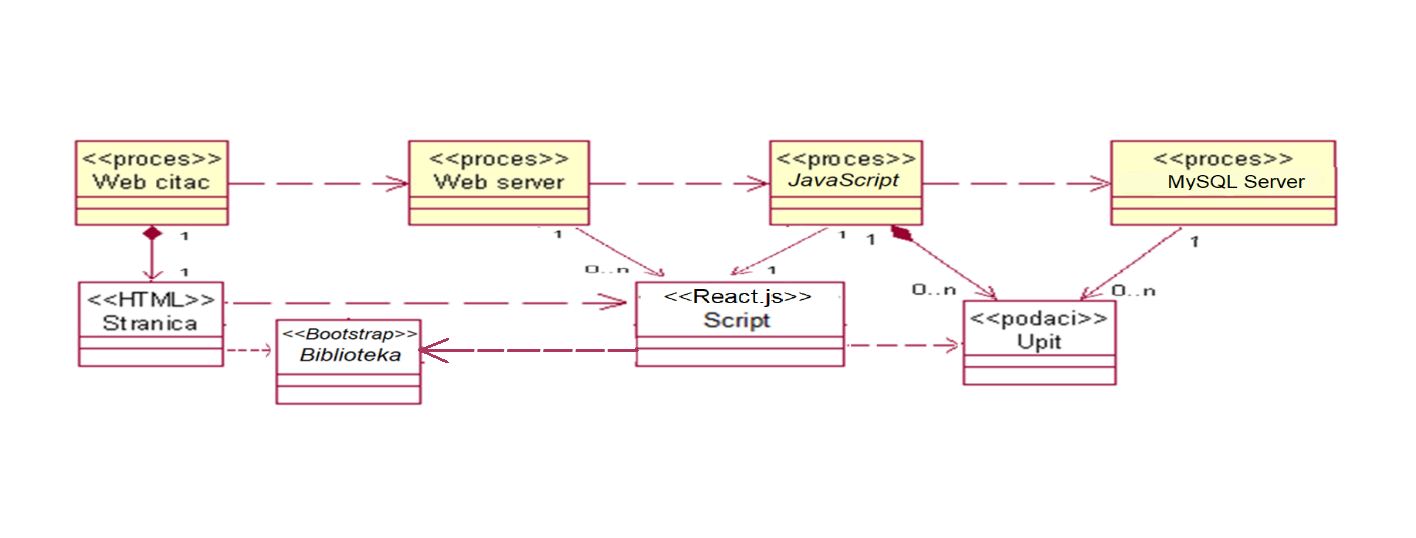
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

Web aplikacije zasnovane na MySQL, JavaScript i React 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 FoodHouse-a kao Web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju FoodHouse Web aplikacije. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na React-u i MySQL 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 koja koristi komponente Bootstrap-a.

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 zahtevana stranica Javascript, Web server inicira izvršenje React procesa koji obrađuje odgovarajući skript i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira veći broj React procesa.

### React

React proces obavlja posao obrade zadatog JavaScript-a i generiše odgovarajući tekstualni sadržaj koji Web server šalje Web čitaču. Za izvršenje JavaScripta ovaj proces može da zahteva usluge MySQL servera-a. Komunikacija između React procesa i MySQL servera se obavlja preko CRUD funkcija i vraćanja rezultata.

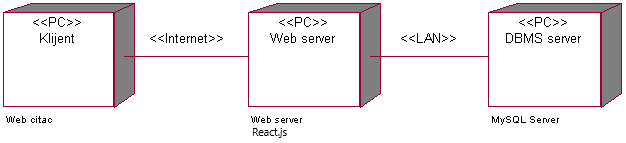
### MySQL Server

MySQL Server je proces koji izvršava funkcionalnost MySQL 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 FoodHouse sistema.



## Klijent

Pristup FoodHouse aplikaciji 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 React.js-a koji vrše obradu zadatih JavaScript skripti. U najopštioj konfiguraciji DBMS se izvršava na posebnoj mašini koja je sa Web serverom u lokalnoj mreži (LAN).

## DBMS server

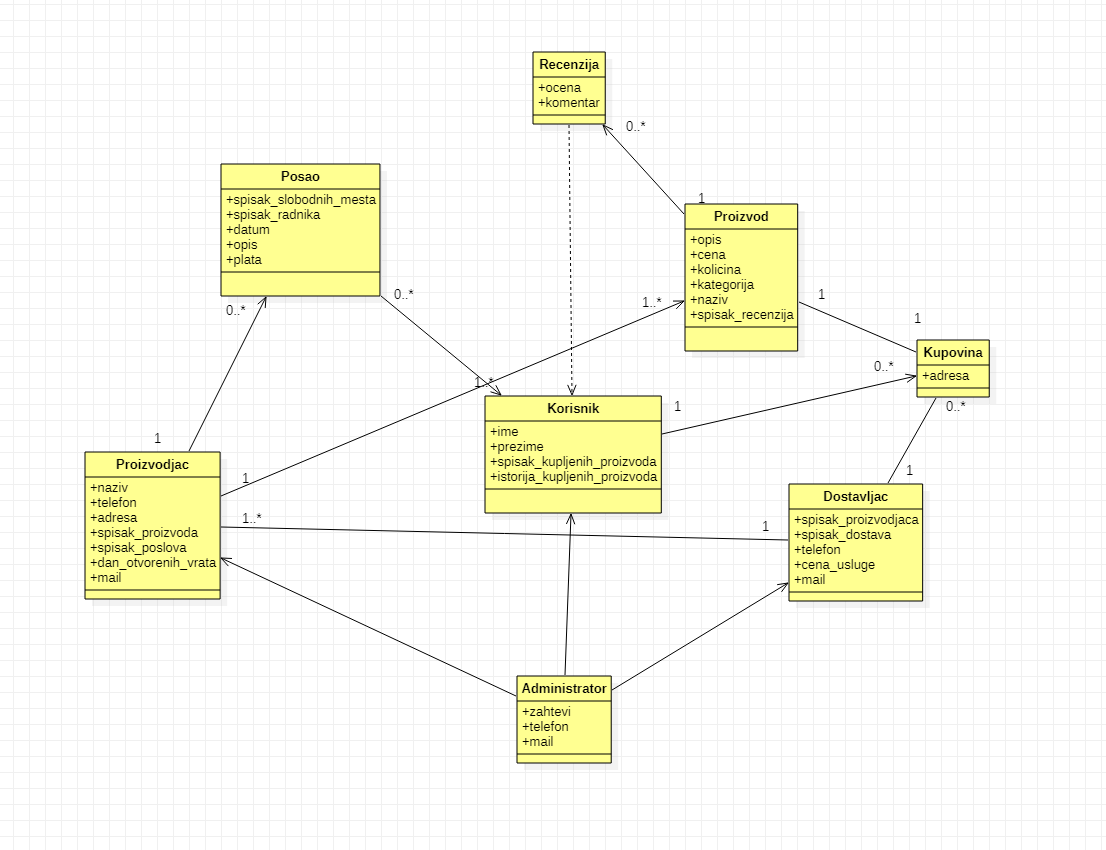
DBMS server je računar na kome se izvršava MySQL Server proces koji realizuje funkcionalnost sistema za upravljanje bazama podataka. Zbog sigurnosti podataka koji se na ovom računaru čuvaju pristup bazi je ograničen samo na računare iz lokalne mreže (LAN).

# Pogled na implementaciju sistema

Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju FoodHouse aplikacije ovaj odeljak sadrži model domena, šemu baze podataka i prikaz komponenti sistema razvrstanih u tri kategorije: komponente korisničkog interfejsa, komponente aplikacione logike i komponente za pristup podacima.

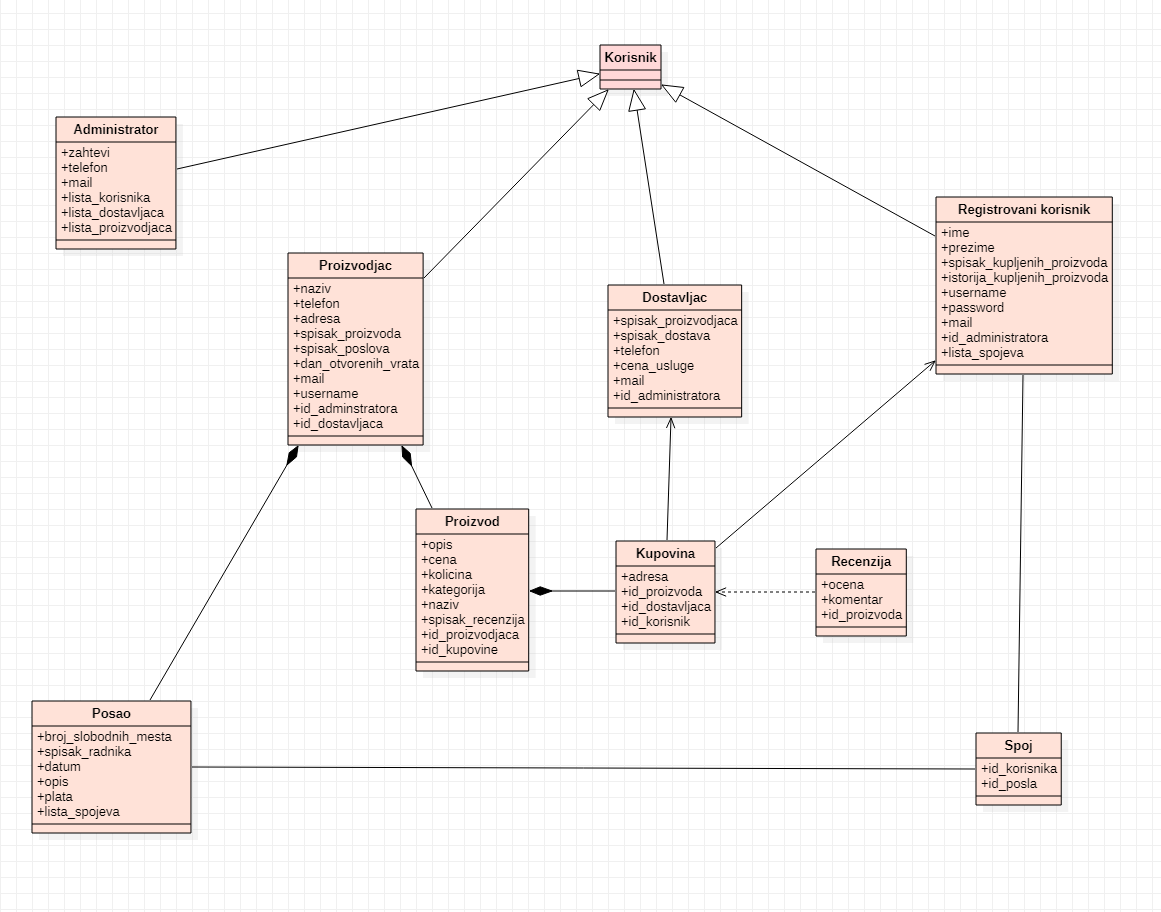
## Model domena

Model domena za koji se FoodHouse 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 je kreirana korišćenjem *MySQL-a*, a dijagram je kreiran korišćenjem StarUML-a.

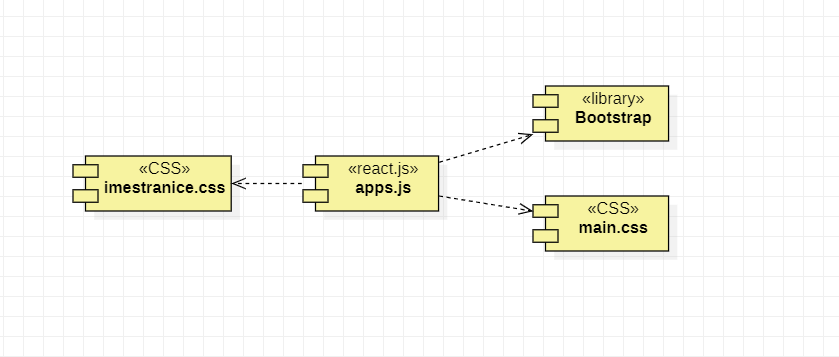


## Komponente sistema

Komponente sistema FoodHouse sajta su React.js skripte, CSS, HTML. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je React.js skript prikazan kao klasa, atributi predstavljaju ulazne podatke koji se uzimaju iz GET ili POST dela HTTP poruke, dok metodi predstavljaju funkcije definisane u okviru skripta i importovanih modula.

### Komponente korisničkog interfejsa

Dizajn korisničkog interfejsa je skup više različitih tipova komponenti. Sastoji se od app.js skripte, više css dokumenata i Bootstrap.



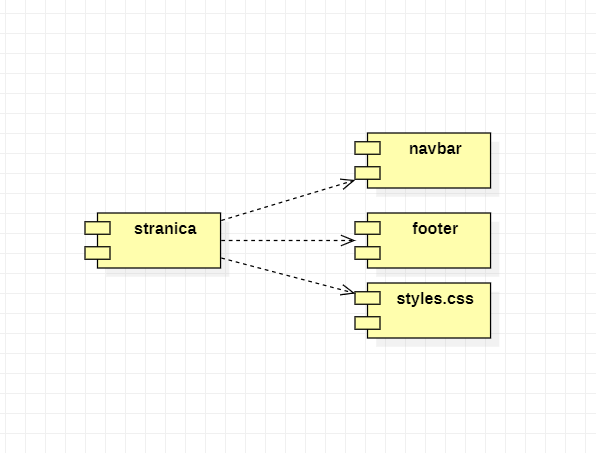
Komponenta **apps.js** predstavlja centar sistema i koristi se za generisanje dinamičkih HTML stranica čiji sadržaj može da varira u zavisnosti od parametara koji joj se proslede pri renderovanju.

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

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

Komponenta **Bootstrap** 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 UI komponente react-a, kao i parcijalni CSS stilovi. Ove parcijalne komponente su prikazane na sledećem dijagramu komponenti.



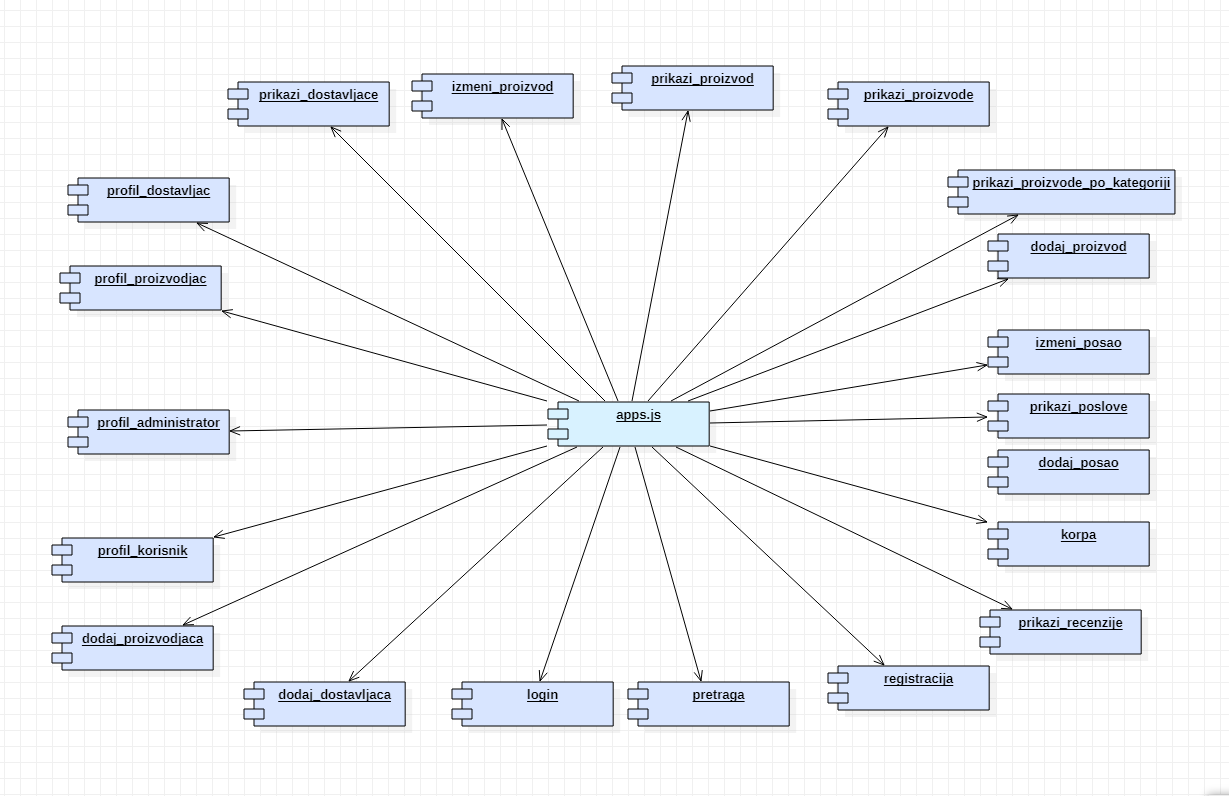
Komponenta **navbar** predstavlja navigacioni meni sajta i na svakoj strani se prikazuje isto.

Komponenta **footer** predstavlja zaglavlje stranice kao i navbar na svakoj strani se prikazuje na isti način.

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

### Komponente aplikacione logike

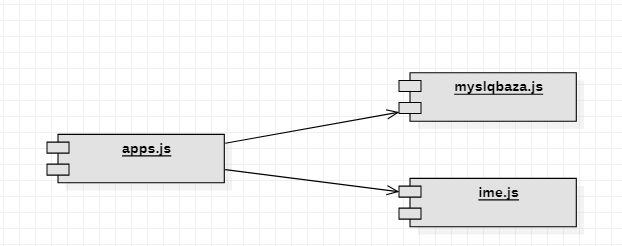
Komponente koje realizuju domen problema se uključuju isključivo preko **apps.js** komponente korisničkog interfejsa. Na sledećem dijagramu su prikazane komponente ovog sloja i njihove međusobne zavisnosti.



Kratak pregled funkcija svake komponente:

* login - stranica za logovanje korisnika
* registracija - stranica za registraciju novog korisnika
* profil\_dostavljac - stranica za prikaz podataka o ulogovanom dostavljaču
* profil\_administrator - stranica za prikaz podataka o ulogovanom administratoru
* profil\_proizvodjac - stranica za prikaz podataka o ulogovanom proizvođaču
* profil\_korisnik - stranica za prikaz podataka o ulogovanom korisniku
* pretraga - stranica za pretragu proizvoda i filtriranje
* prikazi\_proizvode - stranica za prikaz proizvoda
* prikazi\_proizvod – stranica za detaljni prikaz informacija o konkretnom proizvodu
* korpa – stranica za prikaz odabranih proizvoda
* prikazi\_recenzije – stranica za prikaz recenzija jednog proizvođača
* prikazi\_poslove – stranica za prikaz dostupnih poslova
* prikazi\_proizvode\_po\_kategoriji – stranica za prikaz proizvoda iz jedne kategorije
* prikazi\_dostavljace – stranica za prikaz svih dostavljača
* dodaj\_proizvodjaca – stranica za dodavanje novog proizvođača
* dodaj\_dostavljaca - stranica za dodavanje novog dostavljača
* dodaj\_posao - stranica za dodavanje novog posla
* dodaj\_proizvod - stranica za dodavanje novog proizvoda
* izmeni\_proizvod – stranica za ažuriranje podataka o proizvodu
* izmeni\_posao – stranica za ažuriranje podataka o poslu

### Komponente za pristup podacima

Pristup MySQL bazi podataka je omogućen pomoću funkcija koje su definisane u okviru eksternog React.js modula mysqlbaza.js. Povezivanje sa bazom se obavlja u okviru apps.js fajla. Kreirane su i skripte <ime>.js koje se koristi za čuvanje podataka u bazi.Uključuju se na početku apps.js,tako da su sve funkcije za pristup podacima i komunikaciju sa bazom dostupne svim kreiranim komponentama. 

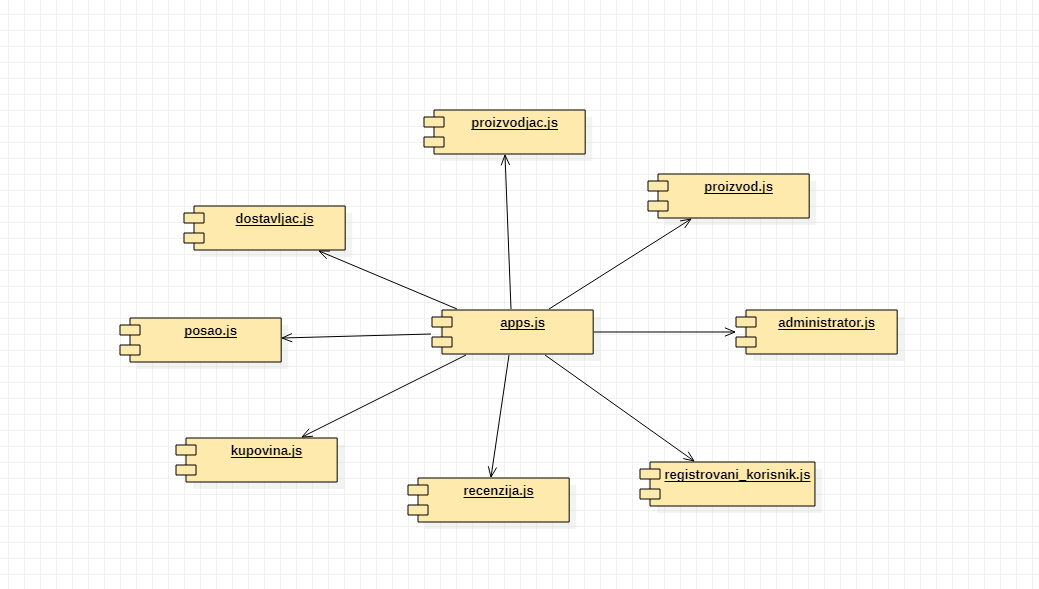
Na sledećem UML dijagramu klasa navedene su osnovne funkcije za pristup podacima iz baze:



Navedene funkcije obavljaju sledeće zadatke:

* connect - povezivanje sa bazom podataka, poziva se u db.js skriptu
* Schema – kreiranje nove šeme za pristup bazi
* model – kreiranje modela za bazu
* find– vraća objekte sa osobinama navedenim u objektu koji se prosledjuje kao parametar funkcije
* findOne– vraća jedan objekat sa osobinama navedenim u objektu koji se prosledjuje kao parametar funkcije
* updateOne – ažuriranje jednog objekta sa navedenim osobinama
* save – čuvanje novog objekta u bazi
* findById – vraća objekat na osnovu Id
* findByIdAndDelete – briše objekat iz baze na osnovu Id
* findByIdAndUpdate– ažuriranje podataka o objektu sa navedenim Id-om

Za čuvanje podataka u bazi koriste se modeli koji su kreirani u posebnim skriptama. Na sledećem dijagramu komponenti prikazani su modeli korišćeni za aplikaciju.



# 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 sajtu.
2. Vreme potrebno za pristupanje bazi podataka u cilju izvršenje nekog upita ne sme da veće od 5 sekunde.

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. FoodHouse će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada sajt nije dostupan ne sme da pređe 5%.
2. Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 168 sati.