**DIZAJN SISTEMA**

**4 RATE**

Aleksandar Ćirić 1132/20

Marko Zeljković 1131/20

Gordana Tubonjić 1188/20

Danica Delić 1190/20

Suzana Rašić 11110/21

**Sadržaj**

[1. Uvod 1](#_Toc167982895)

[1.1. Namjena sistema 1](#_Toc167982896)

[1.2. Projektni ciljevi 2](#_Toc167982897)

[1.3. Definicije, skraćenice 2](#_Toc167982898)

[1.4. Referentni dokumenti 2](#_Toc167982899)

[2. Arhitektura postojećeg sistema 3](#_Toc167982900)

[3. Predložena arhitektura 4](#_Toc167982901)

[3.1. Kratak pregled arhitekture i funkcionalnosti podsistema 4](#_Toc167982902)

[3.2. Dekompozicija sistema 6](#_Toc167982903)

[3.3. HW/SW mapiranje 8](#_Toc167982904)

[3.4. Perzistentni sloj 10](#_Toc167982905)

[3.5. Kontrola prava pristupa i sigurnosti 11](#_Toc167982906)

[3.6. Kontrola toka 12](#_Toc167982907)

[3.7. Granična stanja sistema 13](#_Toc167982908)

# 1. Uvod

## 1.1. Namjena sistema

Sistem „4RATE“ je platforma namijenjena prikazu informacija o restoranima, omogućavanju povratnih informacija gostiju, dijeljenju recenzija u vidu ocijena i tekstualnih kritika, interakciji korisnika, kao i rezervacije stolova u restoranima. Korisnici pristupaju sistemu putem web pretraživača, a njihove mogućnosti unutar sistema zavise od tipa korisničkog naloga. Postoje tri tipa korisnika: gost, menadžer i administrator.

**Gost** ima sljedeće funkcionalnosti:

* Pregled svih restorana
* Ocjenjivanje i recenziranje restorana
* Komentarisanje restorana
* Pregled rezervacije koje je kreirao

**Menadžer** ima sljedeće funkcionalnosti:

* Kreiranje zahtjeva za dodavanje novog restorana
* Uređivanje informacija o restoranu
* Upravljanje rezarvacijama za restoran
* Pregled poslovanja restorana kroz nedeljne izvještaje

**Administrator** ima sljedeće funkcionalnosti:

* Upravljanje svim korisničkim nalozima
* Uklanjanje restorana
* Prihvatanje ili odbijanje zahtjeva za dodavanje novih restorana

Cilj sistema je poboljšanje iskustva korisnika prilikom odabira restorana, pružanje platforme za transparentnu u korisnu razmjenu informacija, kao i podrška restoranima u efikasnom upravljanju rezervacijama i poslovanjem. Restorani mogu koristiti platformu za privlačenje novih gostiju, pri čemu recenzije i visoke ocjene doprinose u izgradnji reputacije. Sistem pomaže korisnicima da pronađu i rezervišu najbolje restorane prema svojim zahtjevima, dok menadžerima i administratorima omogućava lakše upravljanje restoranima i korisnicima.

## 

## 1.2. Projektni ciljevi

* *Niska cijena* – aplikacija će biti besplatna čime je osigruano da svi korisnici imaju mogućnost korišćenja bez ikakvih finansijskih prepreka
* *Dobra dokumentacija* – uz aplikaciju se prilaže detaljna specifikacija softverskih zahtjeva
* *Lakoća upotrebe* – korisnički interfejs je jednostavan i intuitivan za koriščenje
* *Prilagodljivost* – postoji mogućnost prilagođavanja sistema potrebama i ukusima korisnika
* *Fleksibilnost* – sistem će biti realizovan kao web aplikacija, tako da će pristup biti omogućen sa bilo kog uređaja koji ima instaliran web pretraživač
* *Lakoća testiranja* – kako će sistem biti realizovan u više slojeva, samim tim i proces testiranja komponenti pojedinačnih slojeva će biti olakšan
* *Sigurnost* – prava pristupa određenim opcijama i podacima će biti određenana osnovu vrste naloga koju korisnik posjeduje, pri čemu neće biti moguć bilo kakav vid neovlaštenog pristupa, a sigurnost će takođe biti osigurana i u aplikativnom sloju i sloju baze podataka
* *Slojevitost i transparentnost* – server će imati distribuiranu arhitekturu, ali ga klijenti vide kao jedinstven „logički“ servis

## 1.3. Definicije, skraćenice

|  |  |
| --- | --- |
| Pojam | Opis |
| DMBS | Database Management System – sistem za upravljanje bazom podataka |
| HTTP | Hypertext Transfer Protocol – najčešći protokol za prenos informacija na web-u |
| UML | Unified Modeling Language – opšti jezik za modelaovanje |

## 1.4. Referentni dokumenti

Specifikacija softverskih zahtjeva – 4RATE, aplikacija za rezervaciju mjesta u restoranima

# 2. Arhitektura postojećeg sistema

Trenutno ne postoji nijedan sistem koji ispunjava funkcionalnosti koje nudi ova aplikacija.   
U narednom poglavlju su navedeni detalji o predloženoj arhitekturi ovog sistema.

# 3. Predložena arhitektura

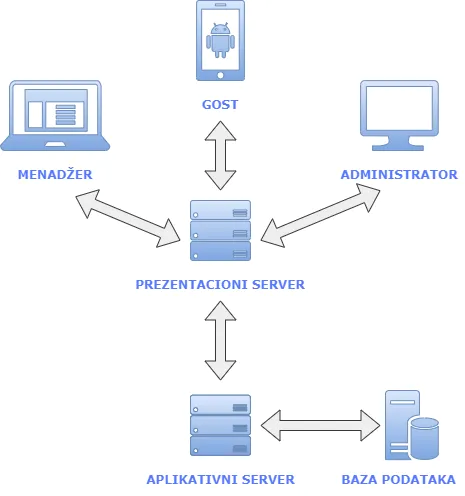
## 3.1. Kratak pregled arhitekture i funkcionalnosti podsistema

Sistem će biti realizovan kao troslojna web aplikacija, odnosno sistem sačinjavaju tri slabo spregnuta podsistema, pri čemu svaki ima odgovarajuću funkcionalnost koju obezbjeđuje. Prvi sloj sistema je prezentacioni sloj koji reprezentuje korisnički interfejs sa kojim korisnici (gosti, menadžeri i administratori) interaguju i šalju HTTP zahtjeve putem web pretraživača. Ovaj sloj sistema komunicira sa drugim slojem koji predstavlja sloj poslovne logike u cilju dohvatanja i rada sa podacima koji se prikazuju korisnicima u čitljivom formatu. Drugi sloj omogućava obradu podataka odnosno izvršava poslovnu logiku sistema, pri čemu se podacima pristupa pomoću sloja baze podataka. Treći sloj čini sloj baze podataka (DBMS) koji se koristi za pohranu podataka, kao i njihovo dohvatanje.

Sistem „4RATE“ možemo podijeliti na sledeće podsisteme:

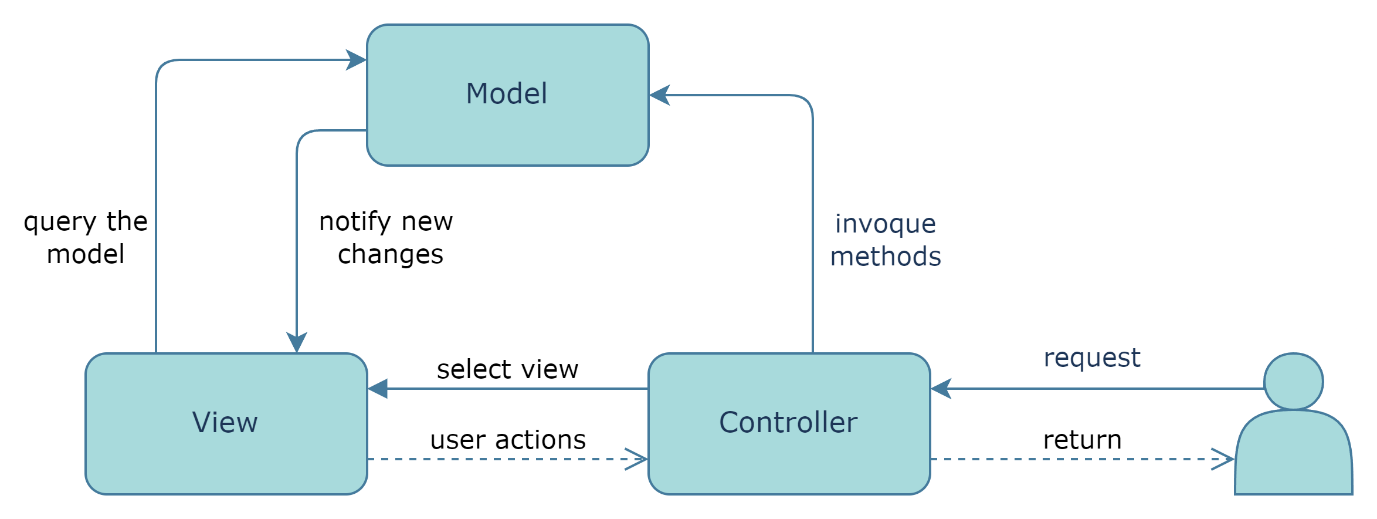
* Podsistem za goste
* Podsistem za menadžere
* Podsistem za administratore

Svaki od podsistema pristupa jednom serveru, tj. serveru sa korisničkim grafičkim interfejsom, a koji u cilju ostvarivanja poslovne logike i dohvatanja podataka, pristupa drugom serveru koji je krajnjim korisnicima transparentan, čime se postiže viši nivo sigurnosti pri radu sistema a omogućava i lakšu proširivost sistema u budućnosti bez narušavanja grafičkog korisničkog interfejsa. Udaljeni server koji vrši poslovnu logiku komunicira sa serverom baze podataka u cilju dohvatanja i dugotrajnog smještanja podataka.

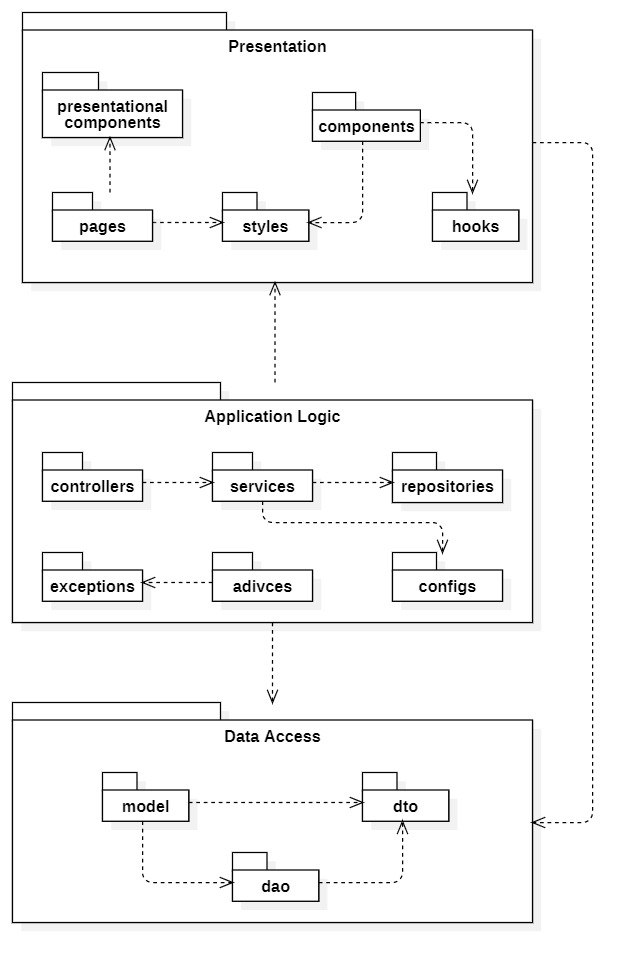


## 3.2. Dekompozicija sistema

Za realizaciju aplikacije korsitimo MVC arhitekturalni stil. Model se koristi za rad sa podacima (domenskim objektima). View (pogled) se koristi za prezentaciju podataka korisniku. Controller (kontroler) je odgovoran za interkaciju sistema i korisnika i prosleđivanja promjena modela prema pogledu.



Sistem možemo dekomponovati, tj. podijeliti na tri osnovna paketa: prezentacija, aplikativna logika i pristup podacima.

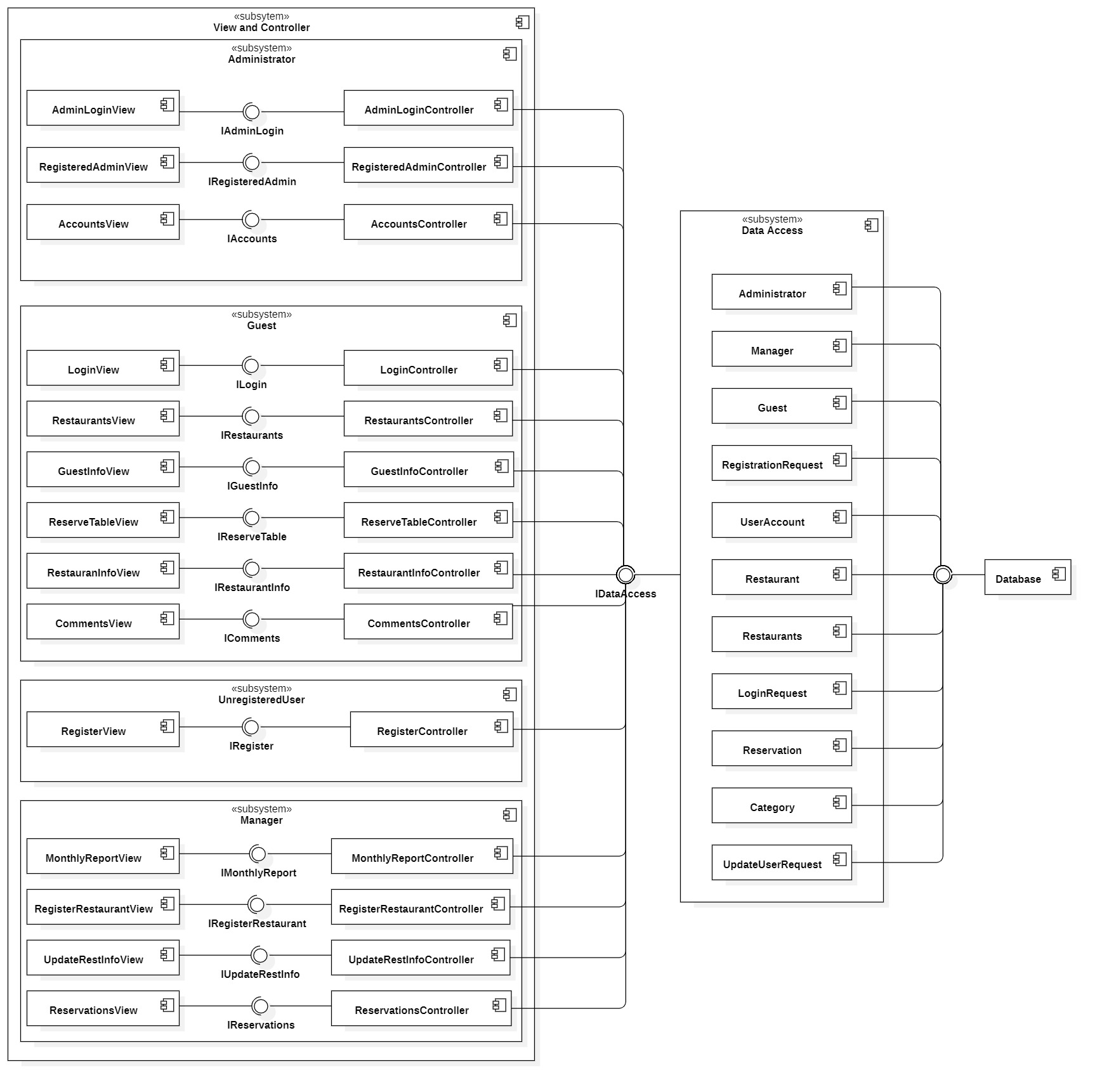


## 3.3. HW/SW mapiranje

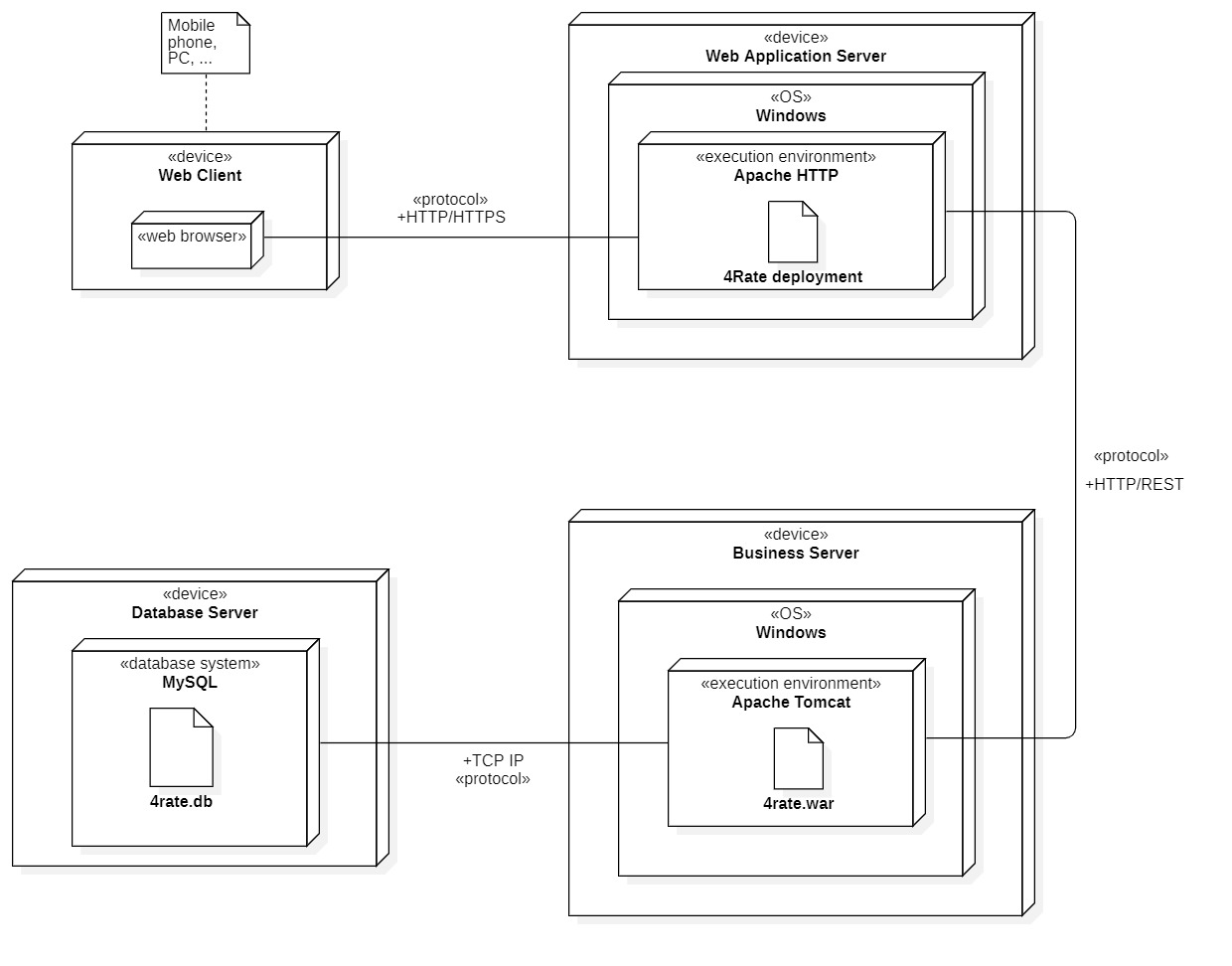
UML dijagrami koji se koriste za HW/SW mapiranje:

* + Dijagram komponenata
  + Dijagram razmještaja

Dijagram komponenata se koristi za modelovanje zavisnosti između komponenata (deisgn time, compile time i run time).

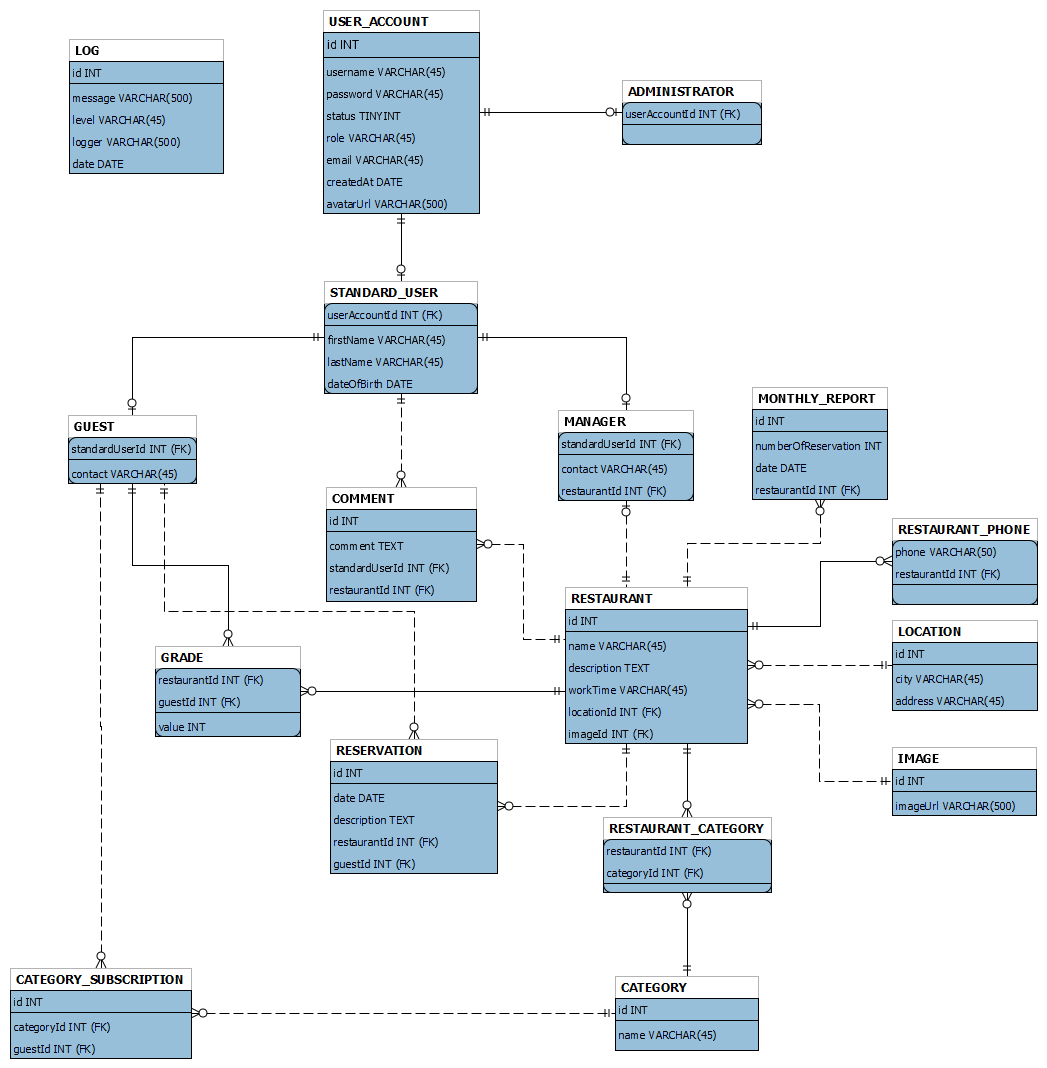


Dijagram razmještaja se koristi za modelovanje rasporeda komponenata u eksploataciji sistema (run time).



## 3.4. Perzistentni sloj

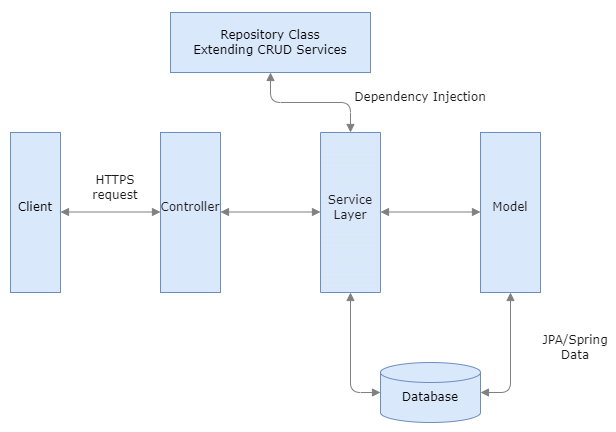
Perzistentni sloj sistema opisan je konceptualnim modelom baze podataka. Model koristi standardnu IE notaciju (Information Engineering) koja je prilagođena za alat MySQL WorkBench u kojem je ovaj model i kreiran.  
Podaci se čuvaju u MySQL relacionoj bazi podataka. Baza je centralizovana, što znači da će biti centralizovana i održavana na jednom mjestu.



## 3.5. Kontrola prava pristupa i sigurnosti

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Klasa | Operacija | Učesnik | | | |
| Administrator | Menadžer | Gost | Neregistrovani korisnik |
| Nalog | ažuriranjePodataka() | ✅ | ✅ | ✅ |  |
| dohvatiPodatke() | ✅ | ✅ | ✅ |  |
| izmjenaPodataka() | ✅ | ✅ | ✅ |  |
| kreiranjeNaloga() |  |  |  | ✅ |
| PrikazSvihNaloga() | ✅ |  |  |  |
| uklanjanjeNaloga() | ✅ |  |  |  |
| suspendovanjeNaloga() | ✅ |  |  |  |
| Zahtjev | potvrdaZahtjeva() | ✅ |  |  |  |
| odbijanjeZahtjeva() | ✅ |  |  |  |
| kreiranjeZahtjeva() |  | ✅ |  |  |
| pregledZahtjeva() | ✅ |  |  |  |
| obradaZahtjeva() | ✅ |  |  |  |
| Komentar | komentariši() |  | ✅ | ✅ |  |
| obrišiKomentar() | ✅ |  | ✅ |  |
| Recenzija | recenziranje() |  |  | ✅ |  |
| Izvještaj | generisanjeIzvještaja() |  | ✅ |  |  |
| prikazIzvještaja() |  | ✅ |  |  |
| Restoran | dohvatiInformacije() |  | ✅ |  |  |
| ažurirajInformacije() |  | ✅ |  |  |
| pretragaRestorana() | ✅ | ✅ | ✅ | ✅ |
| prikazRestorana() | ✅ | ✅ | ✅ | ✅ |
| uklanjanjeRestorana() | ✅ |  |  |  |
| pregledOmiljenihRestorana() |  |  | ✅ |  |
| DodavanjeRestoranaUOmiljene() |  |  | ✅ |  |
| Rezervacija | pregledRezervacija() |  | ✅ | ✅ |  |
| rezervišiSto() |  |  | ✅ |  |
| otkažiRezervaciju() |  | ✅ |  |  |
| čekirajRezervaciju() |  | ✅ |  |  |
| Obavještenje | prikažiObavještenje() |  |  | ✅ |  |

## 3.6. Kontrola toka



* Klijent kreira HTTP zahtjev (GET, PUT, POST, itd)
* HTTP zahtjev se prosljeđuje kontroleru. Kontroler mapira zahtjev. Obrađuje podatke i poziva serversku logiku.
* Poslovna logika se izvodi u sloju usluga. Spring Boot obavlja svu logiku nad podacima baze podataka koji se mapiraju u klasu modela Spring Boot preko JPA (Java Persistance Library).
* JSP (Java Server Page) stranica se vraća kao odgovor od kontrolora

## 3.7. Granična stanja sistema

* *Inicijalizacija / start sistema* – ovo stanje predstavlja period od pokretanja sistema do trenutka kada je sistem spreman za upotrebu. Prvo se pokreće web server koji hostuje aplikaciju, a nakon toga se učitavaju sve konfiguracione datoteke potrebne za rad sistema (baze podataka, postavke aplikacija, itd.). Zatim se uspostavlja veza sa bazom podataka gdje se čuvaju informacije o korisnicima, restoranima, rezervacijama, itd. Nakon toga se vrši provjera integriteta, odnosno provjera konzistentnosti podataka u bazi podataka i ispravnost svih potrebnih resursa, a potom slijedi pokretanje glavne aplikacije i omogućavanje korisničkog pristupa putem web pretraživača.
* *Terminiranje / završetak rada* – ovo stanje predstavlja proces sigurnog isključivanja sistema. Prvo se obavještavaju aktivni korisnici da će sistem biti isključen, a nakon toga slijedi sigurno zatvaranje svih aktivnih korisničkih sesija i očuvanje podataka, nakon čega se vrši odjava svih korisnika iz sistema, te sigurno zatvaranje veze s bazom podataka kako bi se spriječilo gubljenje podataka. Poslije toga se vrši isključivanje web servera i ostalih sistemskih komponenti.
* *Otkazi* – ovo stanje nastupa usljed nepredviđenih kvarova ili grešaka koje onemogućavaju normalan rad sistema. Otkaz sistema može se desiti kao posljedica različitih faktora kao što su hardverski kvarovi, greške u softveru ili napadi. Prvo se vrši detekcija, odnosno otkrivanje greške putem monitoringa sistema, a zatim se administratoru šalje obavijest o otkazu sistema. Nakon toga slijedi izolacija problematičnog dijela sistema kako bi se spriječilo dalje širenje greške i aktivira se backup sistema kako bi se osigurala konzistentnost podataka, a potom se vrši popravka sistema, odnosno pronalaženje i ispravljanje greške te ponovno pokretanje sistema.