eRestoran

Sajt za onlajn rezervaciju stolova

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 26.04.2023. | 1.0 | Inicijalna verzija | Antonije, Aleksa, Marko |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 restoranu 8

6.2.2 Pregled recenzija restorana i jela 8

6.2.3 Pregled sale restorana 8

6.2.4 Pregled menija restorana 8

6.2.5 Prijavljivanje 8

6.2.6 Rezervacija stola 8

6.2.7 Ostavljanje komentara za restoran i jela 8

6.2.8 Ocenjivanje restorana i jela 8

6.2.9 Dodavanje nove sale 8

6.2.10 Dodavanje novog jela u meni 9

6.2.11 Ažuriranje osnovnih podataka o restoranu 9

6.2.12 Ažuriranje izgleda sale restorana 9

6.2.13 Ažuriranje postojećih jela iz menija 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 10

7.1.6 Angular 10

7.1.7 .NET Core 11

7.1.8 SQL Server 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 EntityFramework 11

8.1.4 SQL 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 web aplikacije eRestoran.

# Opseg dokumenta

Dokument se odnosi na eRestoran koji će biti razvijen od strane Trinity Force-a. Namena sistema je efikasno prezentovanje sadržaja restorana i onlajn rezervacija stolova.

# Reference

Spisak korišćene literature:

1. eRestoran – Predlog projekta, Trinity Force-01, V1.0, 2023, Trinity Force.
2. eRestoran – Vizija sistema, Trinity Force-02, V1.0, 2023, Trinity Force.
3. eRestoran – Plan realizacije projekta, Trinity Force-03, V1.0, 2023, Trinity Force.
4. eRestoran – Planirani raspored aktivnosti na projektu, Trinity Force-03, V1.0, 2023, Trinity Force.
5. eRestoran – Specifikacija zahteva, Trinity Force-03, V1.0, 2023, Trinity Force.

# 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. eRestoran će biti implementiran kao web aplikacija zasnovana na .NET Core i TypeScript jeziku i SQL Server bazi podataka [3].
2. Klijentski deo web aplikacije eRestoran će biti optimizovan za sledeće Web čitače: Internet Explorer 6.0 i noviji, Opera 8.0 i noviji, Google Chrome i Mozilla Firefox [3].
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].

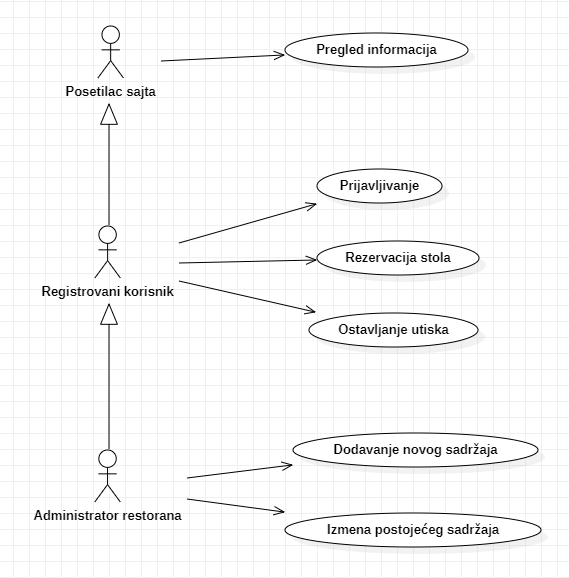
Slučajevi korišćenja web aplikacije eRestorana su:

* Pregled osnovnih podataka o restoranu
* Pregled recenzija restorana i jela
* Pregled sale restorana
* Pregled menija restorana
* Prijavljivanje
* Rezervacija stola
* Ostavljanje komentara za restoran ili jela
* Ocenjivanje restorana i jela
* Dodavanje nove sale
* Dodavanje novog jela u meni
* Ažuriranje osnovnih podataka o restoranu
* Ažuriranje izgleda sale restorana
* Ažuriranje postojećih jela iz menija

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

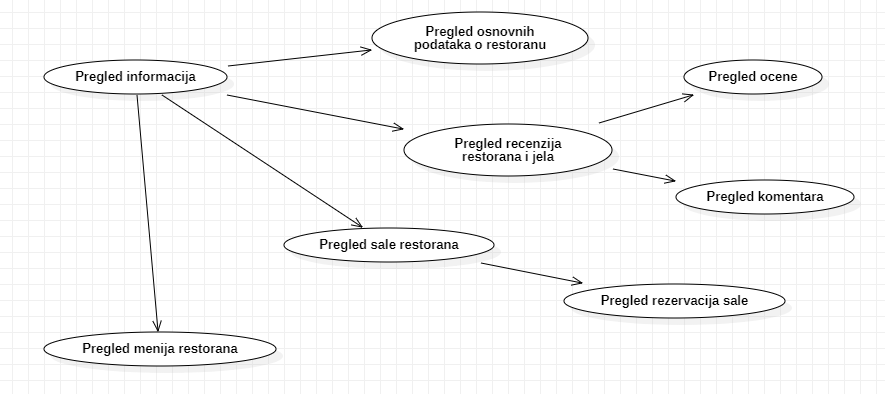
## Dijagrami slučajeva korišćenja

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

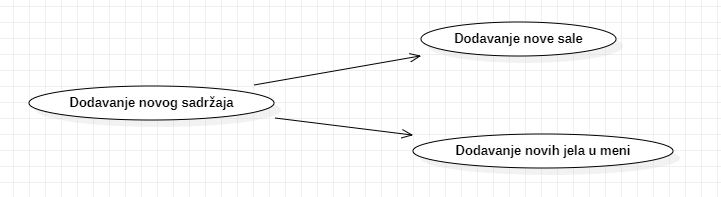


Slučajevi korišćenja *pregled informacija*, *ostavljanje utisaka*, *dodavanje novog sadržaja* i *izmena postojećeg sadržaja* obuhvataju složenije radnje koje se mogu razložiti dalje razložiti na pojedinačne slučajeve korišćenja.

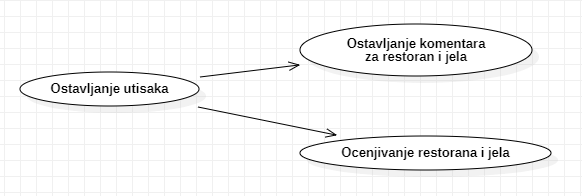
Detaljni UML dijagram za slučaj korišćenja *pregled informacija* je prikazan na sledećoj slici:



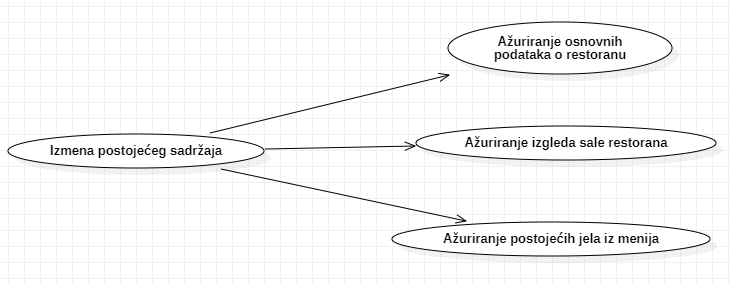
Detaljni UML dijagram za slučaj korišćenja *ostavljanje utisaka* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *dodavanje novog sadržaja* je prikazan na sledećoj slici:



Detaljni UML dijagram za slučaj korišćenja *izmena postojećeg sadržaja* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

### Pregled osnovnih podataka o restoranu

Kratak opis: Prikaz stranice sa osnovnim podacima o restoranu.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Registrovani korisnik, Administrator restorana.

### Pregled recenzija restorana i jela

Kratak opis: Prikaz stranice sa listom komentara i ocena za restoran i jela iz menija.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Registrovani korisnik, Administrator restorana.

### Pregled sale restorana

Kratak opis: Prikaz stranice sa iscrtanom salom restorana.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Registrovani korisnik, Administrator restorana.

### Pregled menija restorana

Kratak opis: Prikaz stranice sa menijem restorana.

Akteri koji iniciraju slučaj korišćenja: Posetilac sajta, Registrovani korisnik, Administrator restorana.

### Prijavljivanje

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

Akteri koji iniciraju slučaj korišćenja: Registrovani korisnik, Administrator restorana.

### Rezervacija stola

Kratak opis: Rezervacija stola u restoranu za određeni termin.

Akteri koji iniciraju slučaj korišćenja: Registrovani korisnik, Administrator restorana.

### Ostavljanje komentara za restoran ili jela

Kratak opis: Ostavljanje javnog komentara za neki restoran ili jelo iz menija.

Akteri koji iniciraju slučaj korišćenja: Registrovani korisnik, Administrator restorana.

### Ocenjivanje restorana i jela

Kratak opis: Ocenjivanje kvaliteta rada nekog restorana ili jela iz menija.

Akteri koji iniciraju slučaj korišćenja: Registrovani korisnik, Administrator restorana.

### Dodavanje nove sale

Kratak opis: Dodavanje nove sale restorana.

Akteri koji iniciraju slučaj korišćenja: Administrator restorana.

### Dodavanje novog jela u meni

Kratak opis: Dodavanje novog jela u meni restorana.

Akteri koji iniciraju slučaj korišćenja: Administrator restorana.

### Ažuriranje osnovnih podataka o restoranu

Kratak opis: Izmena osnovnih podataka o restoranu koji se prikazuju na stranici restorana.

Akteri koji iniciraju slučaj korišćenja: Administrator restorana.

### Ažuriranje izgleda sale restorana

Kratak opis: Promena sadržaja sale restorana.

Akteri koji iniciraju slučaj korišćenja: Administrator restorana.

### Ažuriranje postojećih jela iz menija

Kratak opis: Promena sadržaja nekog jela iz menija.

Akteri koji iniciraju slučaj korišćenja: Administrator restorana.

# 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.

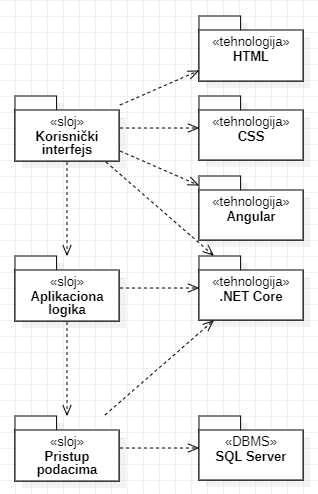
Logički pogled na web aplikaciju eRestoran obuhvata 3 glavna paketa: Korisnički interfejs, Aplikaciona logika, Pristup podacima.

Paket *Korisnički interfejs* sadrži Web stranice, front-end funkcionalnosti 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 back-end funkcije zadužene za realizaciju funkcionalnosti specifičnih za domen sistema koji se razvija.

Paket *Pristup podacima* sadrži back-end funkcionalnost 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, multimedijalni sadržaji i front-end funkcionalnosti koje generišu HTML stranice preko kojih korisnici komuniciraju sa sistemom.

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

### Aplikaciona logika

sloj

Sloj aplikacione logike je srednji sloj u troslojnoj arhitekturi web aplikacije eRestoran. Sadrži back-end funkcije koje realizuju funkcionalnost karakterističnu za domen primene web aplikacije 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 back-end funkcije zadužene za pribavljanje, dodavanje i ažuriranje podataka koji se čuvaju u SQL Server bazi podataka.

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

Tehnologija CSS-a definiše izgled stranica koje se prikazuju u Web čitaču.

### Angular

tehnologija

Angular tehnologija omogućava kreiranje funkcionalnosti na klijentskoj strani za obradu događaja i on je baziran na TypeScript jeziku.

### .NET Core

tehnologija

Tehnologija .NET Core-a obezbeđuje mehanizam za izvršavanje funkcionalnosti na strani servera. Uz pomoć EntityFramework ORM-a postoji mogućnost da se kreiraju baze podataka kojoj možemo pristupati u cilju pribavljanja, unosa i ažuriranja podataka.

### SQL Server

DBMS

SQL Server predstavlja sistem za upravljanje bazama podataka koji će se koristiti za realizaciju web aplikacije eRestoran.

# Pogled na procese

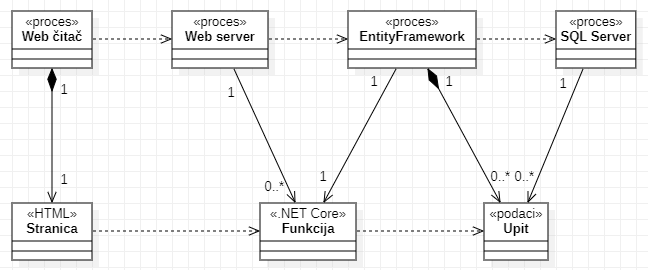
U ovom odeljku je sadržan pogled na procesnu arhitekturu sistema. Ovaj opis treba da sadrži specifikaciju različitih zadataka (procesa i niti) uključenih u rad sistema. Takođe je potrebno dati dijagrame koji pokazuju njihovu interakciju i konfiguraciju. Dodela objekata i klasa na određene zadatke takođe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

Web aplikacije 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 funkcija.

Ilustracije radi u nastavku je dat opis procesa uključenih u izvršenje web aplikacije.

## Procesi

Na sledećem UML dijagramu klasa prikazani su procesi koji učestvuju u izvršenju web aplikacije eRestoran. Dijagram je opšteg tipa i može se primeniti na bilo koju Web aplikaciju zasnovanu na .NET Core-u i SQL Server 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.

### Web server

Web server je proces koji izvršava funkcionalnost opsluživanja zahteva prispelih sa više Web čitača. Ukoliko je zahtevana stranica, Web server inicira slanje HTTP zahteva odgovarajućem procesu i generiše sadržaj koji se vraća čitaču. Web server može paralelno da inicira veći broj HTTP zahteva.

### EntityFramework

EntityFramework je object-relational mapper (ORM) framework koji omogućava da se generišu tabele u bazi podataka uz pomoć objekata klase u cilju olakšavanja rada sa bazom podataka i smanjuje potrbi kucanja koda u SQL upitnom jeziku. Obavlja posao zadate funkcije i generiše odgovarajući sadržaj koji Web server šalje Web čitaču. Za izvršenje funkcija ovaj proces može da zahteva usluge SQL Server-a. Komunikacija između EntityFramework procesa i SQL Servera se obavlja preko prosleđivanja upita i vraćanja rezultata.

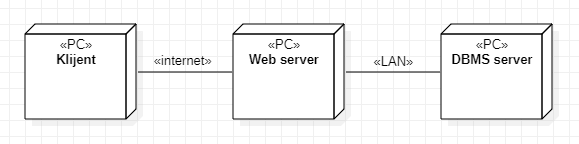
### SQL Server

SQL Server je proces koji izvršava funkcionalnost SQL 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 razmeštaja web aplikacije eRestoran.



## Klijent

Pristup web aplikaciji eRestoran 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-a koji vrše obradu zadatih funkcija. 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 SQL 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).

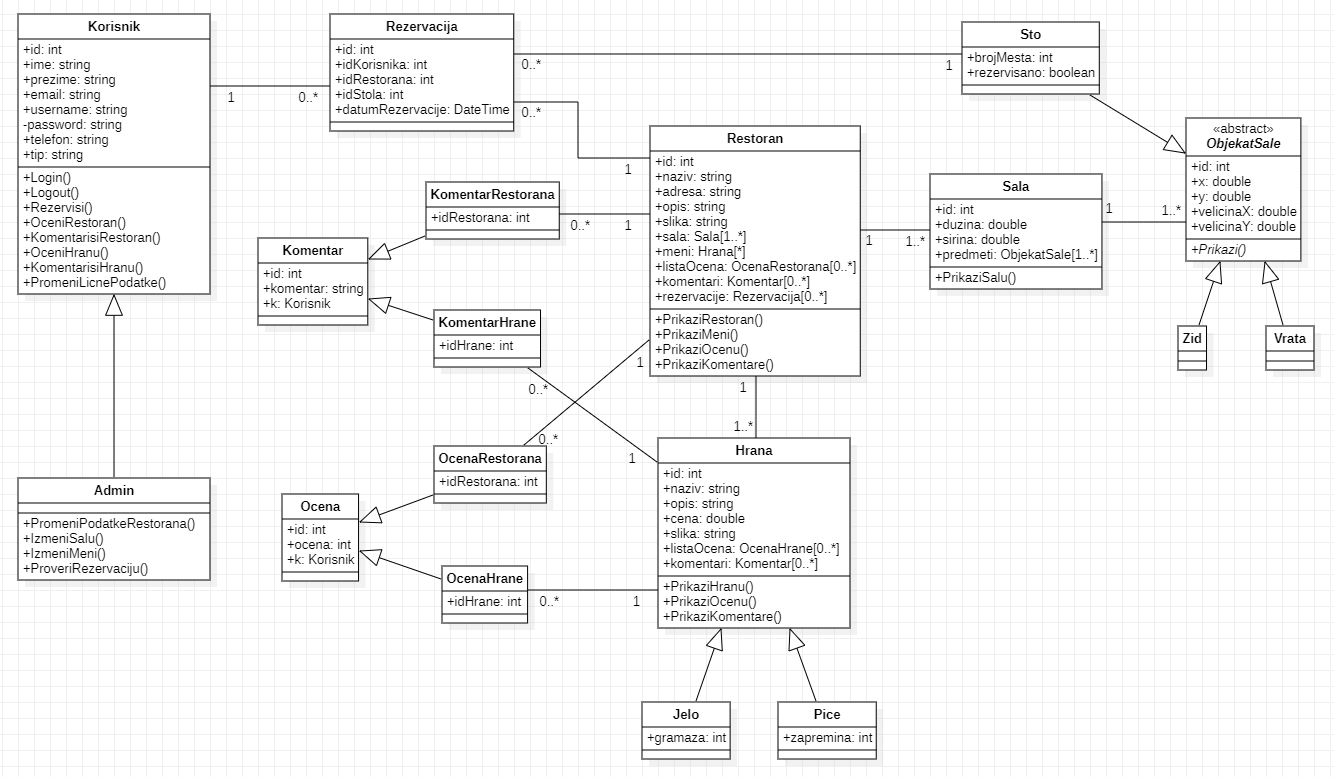
# Pogled na implementaciju sistema

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

## Model domena

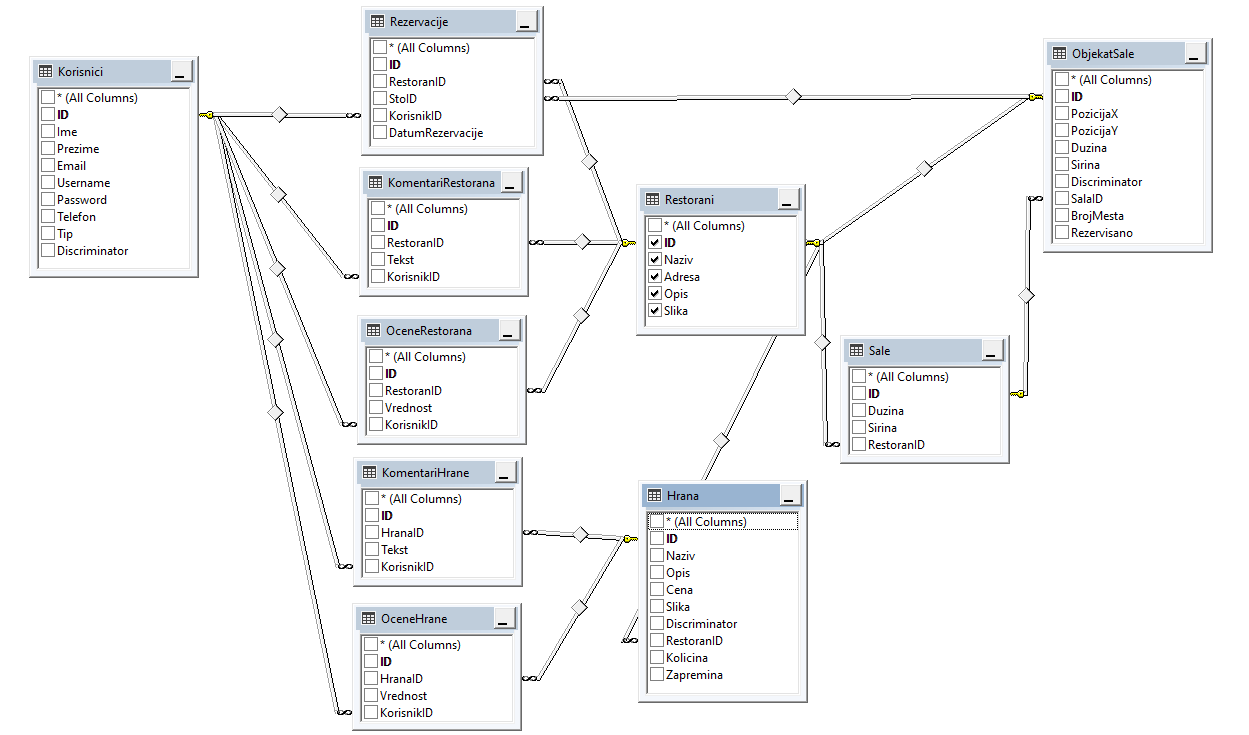
Model domena za koji se web aplikacija eRestoran projektuje je ilustrovan UML dijagramom klasa. U njemu su prikazane domenske klase, neki od njihovih atributa, kao i veze koje se mogu identifikovati između njih.

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 programa *SQL Server Management Studio Management Studio 19*, dok je SQL Server migracija obavljena pomoću *Entity Framework*-a.



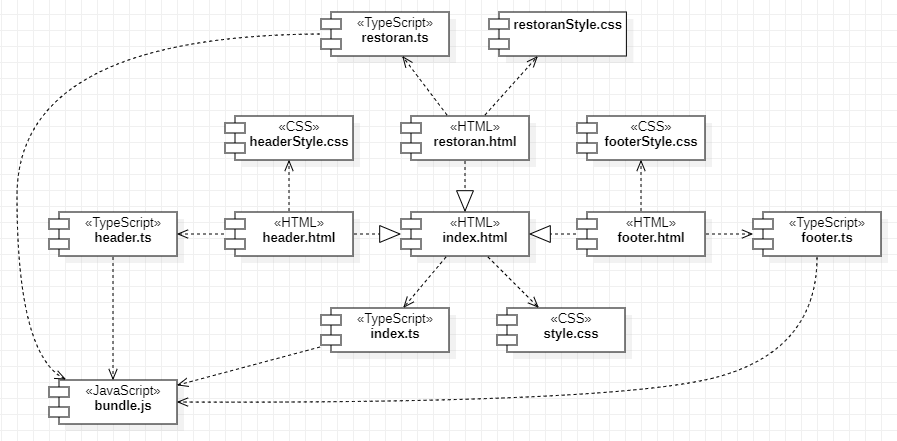
## Komponente sistema

*NAPOMENA: Web aplikacija eRestoran projekat nije kompletno implementirana, tj. implementirana je na nivou arhitekturnog prototipa. Zbog ovoga sledeću sekciju trebate shvatiti kao ilustraciju kako bi dokument trebao da izgleda. Opis komponenti sistema za vaše projekte mora biti kompletan! Procena složenosti vašeg rešenja će zavisiti od opisa sistema koji ovde date.*

Komponente sistema web aplikacije eRestoran su funkcije čiji će pregled biti dat po arhitekturnim slojevima. Za ilustraciju će biti korišćeni UML dijagrami komponenti, ali i dijagrami klasa. U slučajevima gde je metoda 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 sledećim komponentama:



Komponenta **index.html** je glavna stranica gde koja će prikazivati sadržaj ostalih komponenti. Sadržaj će se razlikovati u zavisnosti od toga na kojo stranici se korisnik nalazi. Ona je realizovana komponentama: header, footer i glavni sadržaj stranice.

Komponente **header.html** i **footer.html** će se prikazivati redom na vrhu i na dnu stranice. U header-u se nalazi navigacija i dugme za logovanje korisnika, a u footer-u se nalaze kontakt telefon i email restorana.

Komponenta **restoran.html** će prikazivvati glavni sadržaj stranice koju korisnik posećuje.

Svaka HTML komponenta ima svoju komponentu stila (**style.css**, **headerStyle.css**, **footerStyle.css**, **restoranStyle.css**) koja predstavlja opis stilova za pojedine HTML elemente koji se javljaju na različitim stranicama.

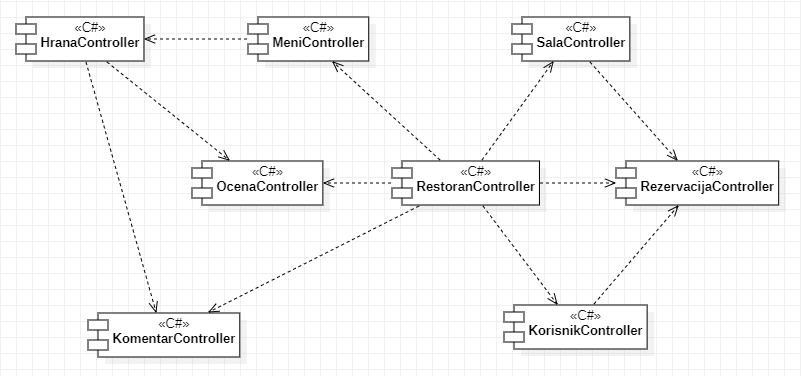
Takođe, svaka HTML komponenta ima svoju TypeScript komponentu koja služi da implementira funkcionalnost stranica na klijentskoj strani.

Komponente **index.ts**, **header.ts**, **footer.ts** i **restoran.ts** su komponente koje u sebi sadrže funkcionalnost na klijentskoj strani aplikacije i implementirane su u njihovim odgovarajućim HTML stranicama. Sve ove komponente se kompajliraju, skupljaju i izvršavaju u komponenti bundle.js.

Komponenta **bundle.js** je JavaScript komponenta koja sadrži funkcionalnost svih stranica i ona izvršava zadatke sa klijentske strane aplikacije.

### 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:

****

Svaka komponenta aplikacione logike je realizovana kao kontroler koji sadrži skup funkcija vezana za deo aplikacije.

Komponenta **RestoranController** sadrži funkcionalnosti i obrađuje zahteve vezane za restoran. Ovaj kontroler implementira funkcije za prikaz i izmenu restorana i osnovnih podataka o restoranu.

Komponenta **KorisnikController** sadrži funkcionalnosti i obrađuje zahteve vezane za registrovane korisnike. Ovaj kontroler implementira funkcije za logovanje korisnika, prikaz profila i izmenu osnovnih podataka o korisniku. Kontroler takođe implementira funkcije koje su vezane za administratora restorana.

Komponenta **MeniController** sadrži funkcionalnosti i obrađuje zahteve vezane za meni restorana. Ovaj kontroler implementira funkcije za prikaz menija i izmenu sadržaja menija kao što su dodavanje i brisanje jela iz menija.

Komponenta **HranaController** sadrži funkcionalnosti i obrađuje zahteve vezane za neko jelo ili piće. Ovaj kontroler implementira funkcije za prikaz i izmenu hrane i osnovnih podataka o hrani.

Komponente **OcenaController** i **KomentarController** sadrže funkcionalnosti i obrađuju zahteve vezane za ocenjivanje i komentarisanje restorana ili hrane. Ovi kontroleri implementiraju funkcije za prikaz ocenam i komentara za neki restoran ili jelo iz menija.

Komponenta **SalaController** sadrži funkcionalnosti i obrađuje zahteve vezane za salu restorana. Ovaj kontroler implementira funkcije za prikaz sale i njegovu izmenu, što podrazumeva dodavanje i brisanje objekata koji se nalaze u sali.

Komponenta **RezervacijaController** sadrži funkcionalnosti i obrađuje zahteve vezane za rezervaciju stola restorana. Ovaj kontroler implementira funkcije za rezervisanje stola u određenom vremenskom terminu.

### Komponente za pristup podacima

Pristup bazi podataka je realizovan uz pomoć kontrolera. Struktura baze je realizovana preko modela koji predstavljaju tabele baze. Model konteksta sadrži reference na sve ostale modele koji se mapiraju i generišu bazu podataka kojoj možemo pristupiti. Zahvaljujući tome, funkcije za pristup podacima su dostupne svim komponentama aplikacione logike.

# 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 neće imati tačno određen broj simultanih pristupa korisnika portalu.
2. Vreme potrebno za pristupanje bazi podataka u cilju izvršenje nekog upita ne sme da bude veće od 30 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. Web aplikacija eRestoran 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.