Servizo

Arhitekturni projekat

Verzija 1.0

Pregled izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 14.10.2025 | 1.0 | Inicijalna verzija | Lazar,Veljko,Aleksandar,Nemanja |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Cilj dokumenta

2. Opseg dokumenta

3. Reference

4. Predstavljanje arhitekture

5. Ciljevi i ograničenja arhitekture

6. Pogled na slučajeve korišćenja

6.1 Dijagrami slučajeva korišćenja

6.2 Kratak opis slučajeva korišćenja

6.2.1 Pregled menija

6.2.2 Pregled artikla (detalji)

6.2.3 Kreiranje narudžbine sa stola

6.2.4 QR narudžbine

6.2.5 Dodavanje stavke u narudžbinu

6.2.6 Izmena stavke narudžbine

6.2.7 Brisanje stavke iz narudžbine

6.2.8 Prenos stavki između stolova

6.2.9 Podela računa po stavci

6.2.10 Plaćanje celog računa

6.2.11 Plaćanje gotovinom

6.2.12 Plaćanje karticom (POS)

6.2.13 Online plaćanje

6.2.14 Upravljanje artiklima

6.2.15 Upravljanje kategorijama

6.2.16 Upravljanje cenama

6.2.17 Zalihe i dostupnost

6.2.18 Upravljanje stolovima

6.2.19 Dodela gosta stolu

6.2.20 Dodela konobara stolu

6.2.21 Upravljanje korisnicima i ulogama

6.2.22 Evidencija smena

6.2.23 Prijavljivanje

6.2.24 Odjavljivanje

6.2.25 Reset lozinke

6.2.26 Pregled istorije narudžbina

8. Pogled na procese

8.1 Procesi

8.1.1 WEB čitač (React SPA)

8.1.2 Web server (Reverse-proxy / Static)

8.1.3 ASGI aplikacija (Django JSON API)

8.1.4 MongoDB Server

9. Pogled na raspoređivanje sistema

9.1 Klijent

9.2 Web server

9.3 DBMS Server

10. Pogled na implementaciju sistema

10.1 Model domena

10.2 Šema baze podataka

10.3 Komponente sistema

10.3.1 Komponente korisničkog interfejsa

10.3.2 Komponente aplikacione logike na serverskoj strani

10.3.3 Komponente aplikacione logike na klijentskoj strani

10.3.4 Komponente za pristup podacima

11. Performanse

12. Kvalitet 16

**Arhitekturni projekat**

# Cilj dokumenta

Cilj ovog dokumenta je detaljni opis arhitekture Servizo aplikacije.

# Opseg dokumenta

Dokument se odnosi na Servizo web aplikaciju koja će biti razvijena od strane 404 Found tima. Servizo je centralizovani sistem za upravljanje restoranima i kafićima. Namena sistema je efikasno prezentovanje ponude, kreiranje i održavanje menija i cenovnika, obrada narudžbina, upravljanje stolovima i kontrola uloga zaposlenih (administrator, menadžer, konobar, kupac), naplata i fiskalizacija, praćenje zaliha, kao i generisanje izveštaja i analitika poslovanja.

# Reference

Spisak korišćene literature:

1. Servizo – Predlog projekta, Servizo, V1.0, 2025, 404 Found.
2. Servizo – Planirani raspored aktivnosti na projektu, V1.0, 2025, 404found.
3. Servizo – Plan realizacije projekta, V1.0, 2025, 404 Found..
4. Servizo – Vizija sistema, V1.0, 2025, 404 Found.
5. Servizo – Specifikacija zahteva, V1.0, 2025, 404 Found.

# 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 utiču na arhitekturu i projektovanje Servizo sistema:

1. Servizo je web aplikacija sa React.js SPA klijentom i Django-mongodb-backendom; baza podataka je MongoDB.
2. Klijentski deo je optimizovan za moderne pregledače: Chrome, Firefox, Edge i Safari (poslednje dve stabilne verzije), uz responsive prikaz na mobilnim uređajima.
3. Svi performansni zahtevi definisani u moraju biti uzeti u obzir pri izboru arhitekture i razvoju sistema.
4. Klasifikacija menija, artikala i alergena standardizuje se prema internoj tipologiji usaglašenoj sa važećim propisima.

# Pogled na slučajeve korišćenja

Servizo — obuhvata sledeće funkcionalnosti

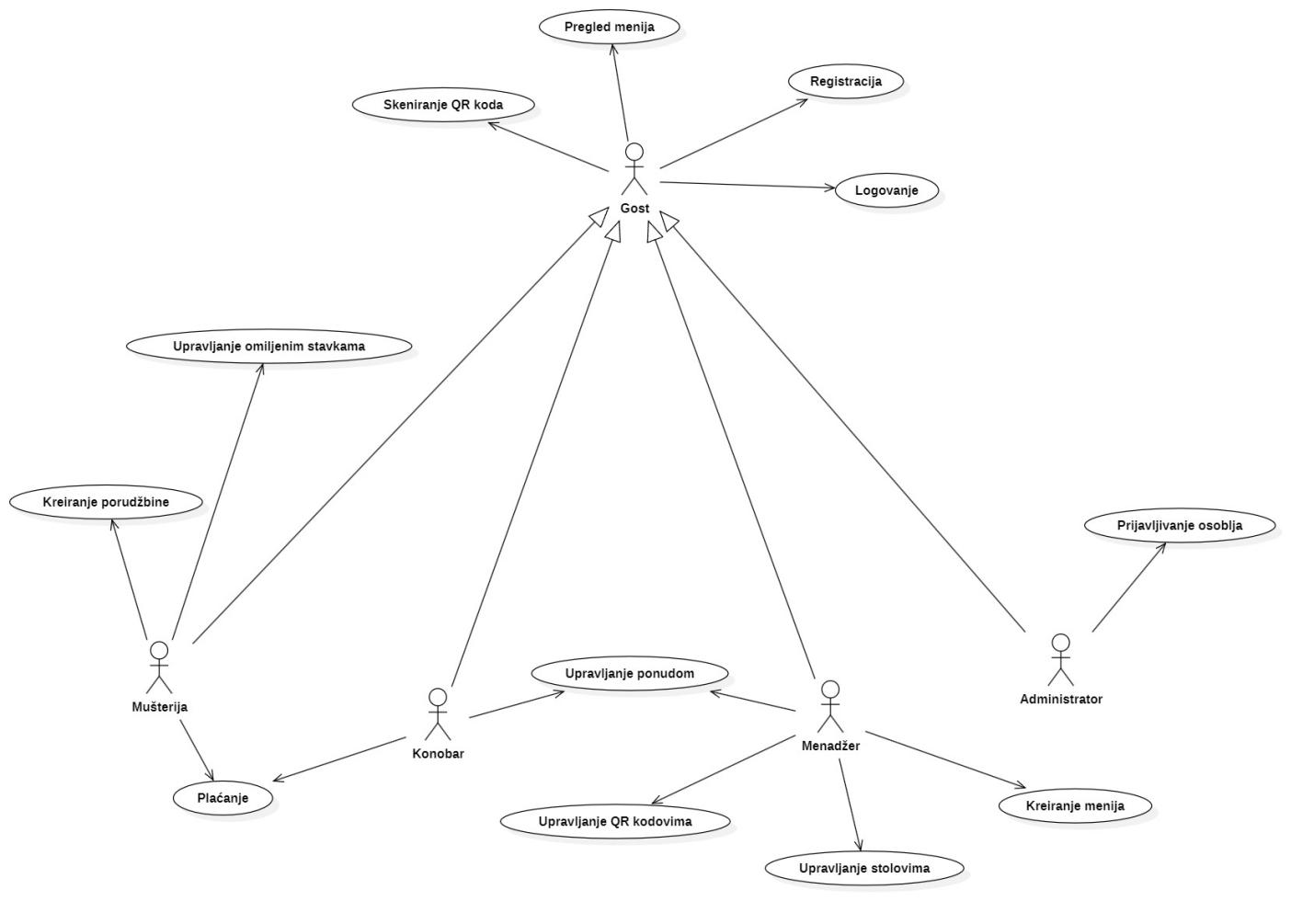
Pregled menija po kategorijama

* Pregled menija za različite tipove korisnika
* Pregled artikla (detalji, alergeni, varijante)
* Pregled podataka o ulogovanom korisniku
* Pregled stolova i narudžbina na njima
* Dodavanje stavki u korpu
* Potvrđivanje korpe (slanje narudžbine)
* Kreiranje narudžbine sa stola
* QR naručivanje
* Prenos stavki između stolova
* Podela računa po osobi ili po stavci
* Dodavanje stavki i kategorija na meni
* Izmena stavki i kategorija menija
* Brisanje stavki i kategorija menija
* Upravljanje cenama (akcije, vremenska važenja)
* Zalihe – označavanje dostupno/nedostupno
* Dodavanje stavke sa menija u favorites
* Uklanjanje stavke iz favorites
* Plaćanje karticom (POS)
* Plaćanje kešom – signalizacija za pozivanje konobara
* Online plaćanje (Stripe)
* Registracija
* Prijavljivanje
* Odjavljivanje (Logout)
* Reset lozinke
* Ažuriranje podataka o korisniku
* Kreiranje/brisanje korisnika i dodela uloga (RBAC)
* Dodavanje i brisanje stolova
* Dodela gosta stolu
* Dodela konobara stolu
* Obaveštavanje konobara o pristizanju nove narudžbine
* Pregled istorije narudžbina

Akteri koji mogu inicirati slučajeve korišćenja: gost, konobar, menadžer, administrator.

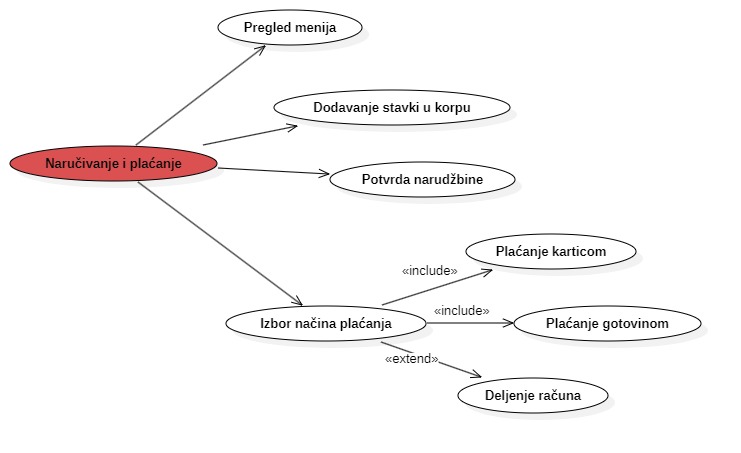
## Dijagrami slučajeva korišćenja

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

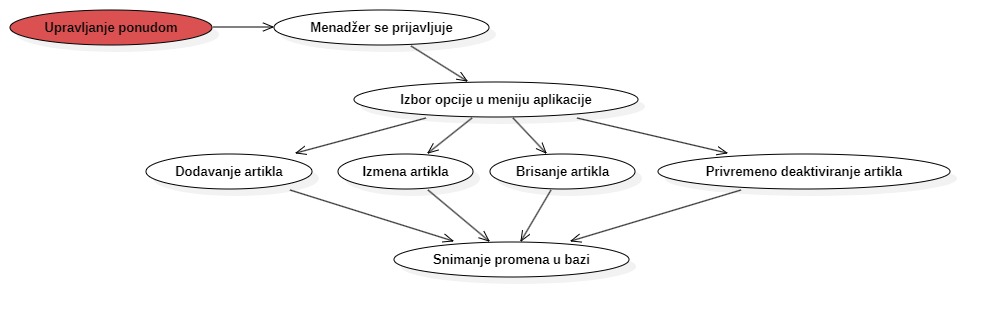


Slučajevi korišćenja *naručivanje i plaćanje, upravljanje ponudom* i *upravljanje omiljenim stavkama* obuhvataju složenije radnje koje se mogu dalje razložiti na pojedinačne slučajeve korišćenja.

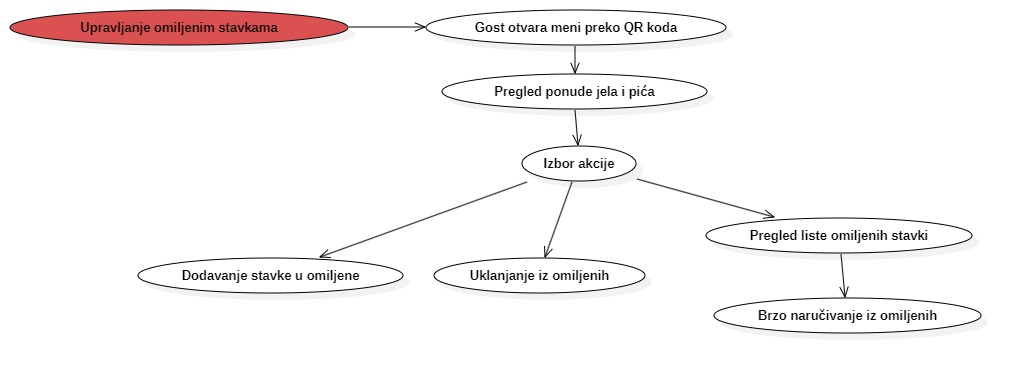
Detaljni UML dijagram za slučaj korišćenja *naručivanja i plaćanja* je prikazan na sledećoj slici:



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



Detaljni UML dijagram za slučaj korišćenja *upravljanje omiljenim stavkama* je prikazan na sledećoj slici:



## Kratak opis slučajeva korišćenja

Gost predstavlja lice koje nije ulogovano.

Musterija predstavlja lice koje se ulogovalo I ima benefite poput online narucivanja, online placanja, liste omiljenih stavki, pregled narudzbina, account page, itd..

**6.2.1 Pregled menija**Kratak opis: Prikaz kategorija i artikala sa opisom i cenom.  
Akteri: Gost, Mušterija, Konobar, Menadžer, Administrator.

**6.2.2 Pregled artikla (detalji)**  
Kratak opis: Detaljan prikaz artikla (opis, alergeni, varijante).  
Akteri: Gost, Mušterija, Konobar, Menadžer, Administrator.

**6.2.3 Kreiranje narudžbine sa stola**Kratak opis: Kreiranje narudžbine za konkretan sto.  
Akteri: Gost, Mušterija,Konobar, Menadžer.

**6.2.4 QR narudžbine**Kratak opis: Mušterija skenira QR kôd stola i pravi sopstvenu narudžbinu.  
Akteri: Mušterija.

**6.2.5 Dodavanje stavke u narudžbinu**Kratak opis: Ubacivanje artikla/varijante u otvorenu narudžbinu.  
Akteri:, Mušterija, Konobar.

**6.2.6 Izmena stavke narudžbine**  
Kratak opis: Promena količine, napomena i/ili varijante.  
Akteri: Mušterija, Konobar.

**6.2.7 Brisanje stavke iz narudžbine**Kratak opis: Uklanjanje artikla iz narudžbine pre zaključivanja.  
Akteri: Mušterija, Konobar.

**6.2.8 Prenos stavki između stolova**  
Kratak opis: Premeštanje jedne ili više stavki sa jednog stola na drugi.  
Akteri: Konobar, Menadžer.

**6.2.9 Podela računa po stavci**  
Kratak opis: Selektivna podela pojedinačnih stavki na odvojene račune.  
Akteri: Mušterija,Konobar, Menadžer.

**6.2.10 Plaćanje celog računa**  
Kratak opis: Plaćanje ukupnog iznosa otvorenog računa za sto.  
Akteri: Mušterija, Konobar, Menadžer, Administrator.

**6.2.11 Plaćanje gotovinom**  
Kratak opis: Evidentiranje naplate gotovinom i izdavanje fiskalnog računa.  
Akteri: Gost, Mušterija , Konobar, Menadžer, Administrator.

**6.2.12 Plaćanje karticom (POS)**Kratak opis: Evidentiranje kartičnog plaćanja preko POS uređaja i fiskalizacija.  
Akteri: Gost, Mušterija, Konobar, Menadžer, Administrator.

**6.2.13 Online plaćanje**  
Kratak opis: Plaćanje preko integrisanog procesa (npr. Stripe).  
Akteri: Mušterija, Menadžer, Administrator.

**6.2.14 Upravljanje artiklima**  
Kratak opis: Dodavanje/izmena/brisanje artikala; opis, varijante, slika.  
Akteri: Menadžer, Administrator.

**6.2.15 Upravljanje kategorijama**  
Kratak opis: Dodavanje/izmena/brisanje kategorija; redosled prikaza.  
Akteri: Menadžer, Administrator.

**6.2.16 Upravljanje cenama**  
Kratak opis: Izmena cena, akcijske cene, vremenska važenja.  
Akteri: Menadžer, Administrator.

**6.2.17 Zalihe i dostupnost**Kratak opis: Obeležavanje artikala kao dostupno/nedostupno; evidencija stanja.  
Akteri: Menadžer, Administrator.

**6.2.18 Upravljanje stolovima**Kratak opis: Kreiranje/izmena/brisanje stolova; QR kôdovi.  
Akteri: Menadžer, Administrator.

**6.2.19 Dodela gosta stolu**  
Kratak opis: Vezivanje narudžbine za konkretnog gosta/sto.  
Akteri: Konobar.

**6.2.20 Dodela konobara stolu**  
Kratak opis: Postavljanje odgovornog konobara za sto/salu/smenu.  
Akteri: Menadžer, Administrator.

**6.2.21 Upravljanje korisnicima i ulogama**  
Kratak opis: Kreiranje/brisanje/arhiviranje korisnika; dodela uloga.  
Akteri: Administrator.

**6.2.22 Evidencija smena**  
Kratak opis: Planiranje i vođenje evidencije smena konobara/menadžera.  
Akteri: Menadžer, Administrator.

**6.2.23 Prijavljivanje**  
Kratak opis: Autentikacija korisnika za pristup zaštićenim funkcijama.  
Akteri: Mušterija, Konobar, Menadžer, Administrator.

**6.2.24 Odjavljivanje**Kratak opis: Zatvaranje sesije korisnika.  
Akteri: Mušterija, Konobar, Menadžer, Administrator.

**6.2.25 Reset lozinke**Kratak opis: Slanje linka/koda i postavljanje nove lozinke.  
Akteri: Mušterija, Konobar, Menadžer, Administrator.

**6.2.26 Pregled istorije narudžbina**  
Kratak opis: Lista prošlih narudžbina sa statusima i iznosima.  
Akteri: Mušterija, Konobar, Menadžer.

# 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 organizaciju 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.

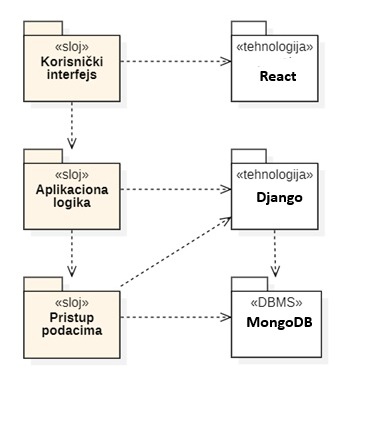
Logički pogled na **Servizo** sistem obuhvata 3 glavna paketa: **Korisnički interfejs**, **Aplikaciona logika** i **Pristup podacima**.

Paket **Korisnički interfejs** sadrži  **React aplikaciju** koja realizuje grafički dizajn i forme preko kojih korisnici sistema (gost, konobari, menadžeri i administratori) komuniciraju sa sistemom. Ovaj paket obuhvata komponente za pregled menija, naručivanje, plaćanje, prijavljivanje, kao i administrativne panele za upravljanje artiklima, kategorijama, stolovima i korisnicima.

Paket **Aplikaciona logika** predstavlja srednji sloj sistema koji sadrži **Django JSON API** servise zadužene za realizaciju poslovne logike i funkcionalnosti specifičnih za domen sistema Servizo, kao što su obrada narudžbina, plaćanja, evidencija zaliha, upravljanje korisnicima i dodela uloga. Ovaj sloj obezbeđuje JSON API interfejs između React aplikacije i baze podataka, uključujući validaciju, autentikaciju, autorizaciju i integraciju sa eksternim servisima za online plaćanje.

Paket **Pristup podacima** sadrži **MongoDB** kolekcije i klase za pristup, dodavanje i ažuriranje podataka koji se čuvaju u bazi. Ovaj sloj obezbeđuje čuvanje informacija o korisnicima, stolovima, narudžbinama, artiklima, kategorijama, plaćanjima i zalihama. ORM/ODM sloj povezuje objekte aplikacione logike sa strukturama u bazi podataka, čime se omogućava efikasno upravljanje transakcijama i konzistentnost podataka.

## Pregled arhitekture – organizacija paketa i podsistema u slojeve



### **Korisnički interfejs**

Ovaj sloj realizuje korisnički interfejs sistema **Servizo**. U njemu su sadržane sve React komponente i multimedijalni sadržaji koji generišu dinamičke stranice preko kojih korisnici komuniciraju sa sistemom. Sloj korisničkog interfejsa obuhvata prikaz menija, naručivanje, pregled stolova, prijavljivanje, upravljanje artiklima i ostale funkcionalnosti dostupne korisnicima različitih uloga (kupac, konobar, menadžer, administrator).  
Sloj korisničkog interfejsa zavisi od sloja aplikacione logike, kao i od React tehnologije i pomoćnih biblioteka za komunikaciju sa REST-stilom JSON API servisima.

### **Aplikaciona logika**

Sloj aplikacione logike predstavlja srednji sloj u troslojnoj arhitekturi sistema **Servizo**. Sadrži **Django Framework** module koji realizuju funkcionalnost karakterističnu za domen sistema, kao što su obrada narudžbina, upravljanje korisnicima, evidentiranje plaćanja, zaliha i stolova. Ovaj sloj uspostavlja vezu između korisničkog interfejsa i sloja za pristup podacima, obezbeđujući validaciju, autentikaciju, autorizaciju i integraciju sa eksternim servisima za plaćanje.  
Sloj aplikacione logike zavisi od sloja za pristup podacima i Django paketa.

### **Pristup podacima**

Sloj za pristup podacima nalazi se na dnu troslojne arhitekture i sadrži **ODM/model** komponente za MongoDB, zadužene za pribavljanje, dodavanje i ažuriranje podataka. Definišu se modeli/kolekcije za korisnike, narudžbine, artikle, kategorije, plaćanja i stolove, uz odgovarajuće indekse. Sloj je zavisan od Django model-sloja i MongoDB baze.

### **React**

Tehnologija **React** omogućava izgradnju dinamičkog korisničkog interfejsa kroz komponente koje reaguju na promene stanja aplikacije. React koristi JSX sintaksu i API komunikaciju sa Django backendom putem **REST-stila JSON endpoint**, čime se obezbeđuje interaktivnost i modularnost korisničkog interfejsa.

### **Django**

Tehnologija **Django** obezbeđuje okvir za razvoj serverske logike i izradu REST-stila JSON servisa pomoću samog **Django Framework-a**. U okviru projekta implementirani su prilagođeni slojevi za parsiranje/validaciju podataka i standardizovane JSON odgovore, uz oslanjanje na mehanizme za autentikaciju, autorizaciju i integraciju sa bazom podataka (MongoDB preko django-mongodb-backend). Django čini osnovu aplikacionog sloja sistema Servizo.

### **MongoDB DBMS**

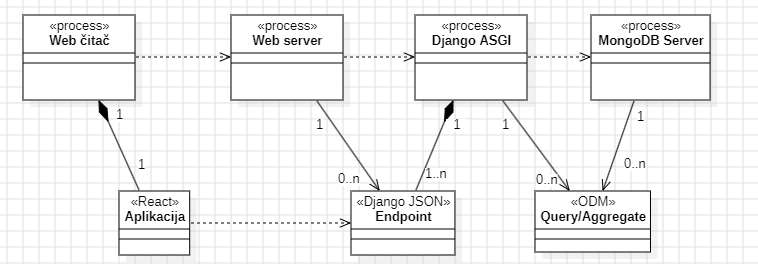
### **MongoDB** predstavlja sistem za upravljanje bazama podataka (DBMS) koji se koristi u okviru sistema **Servizo**. Kao dokumentno orijentisana NoSQL baza, omogućava fleksibilno čuvanje podataka o narudžbinama, artiklima, korisnicima i plaćanjima, što omogućava visok stepen skalabilnosti i performansi u radu aplikacije.

# Pogled na procese

U ovom odeljku je prikazan pogled na procesnu arhitekturu Servizo sistema. Opis sadrži zadatke (procese i niti), njihovu interakciju i konfiguraciju, kao i dodelu objekata/klasa tim zadacima. Servizo je zasnovan na React SPA (klijent), Django REST-stil JSON API na ASGI (backend) i MongoDB (baza podataka). **UML dijagram procesa** prikazuje sledeće procese.

## Procesi

Na sledećem UML dijagramu procesa prikazani su procesi koji učestvuju u izvršenju Servizo aplikacije zasnovane na React SPA + Django ASGI + MongoDB.



### **WEB čitač (React SPA)**

Web čitač je proces pregledača koji renderuje SPA bundle (index.html i prateće JS/CSS fajlove) i izvršava React kod. U jednom tabu postoji jedna instanca aplikacije. Sesija je **Django sesija u kolačiću**, a zahtevi ka serveru šalju se kao JSON preko rute **/api/**. CSRF token se preuzima i šalje u POST/PUT/DELETE zahtevima.  
Zavisnosti: statičke fajlove i index.html dobija od Web servera; API pozive šalje preko reverse-proxy rute **/api/**.

### **Web server (Reverse-proxy / Static)**

Web server opslužuje zahteve više klijenata: isporučuje statičke resurse (SPA bundle) i proksira API saobraćaj ka ASGI aplikaciji. Uobičajena podešavanja: TLS terminacija, gzip/brotli kompresija, cache-control, mapiranje ruta

### **ASGI aplikacija (Django REST)**

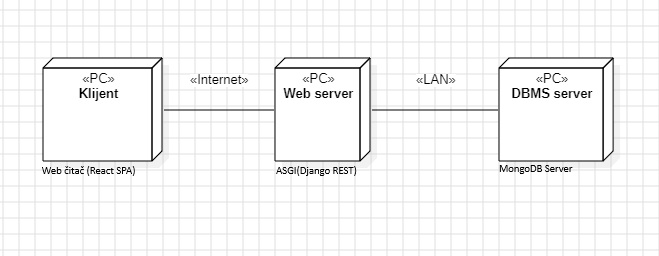
ASGI aplikacija obrađuje **HTTP JSON zahteve u REST-stilu**, sprovodi validaciju, autentikaciju i autorizaciju, upravlja domenima narudžbina, plaćanja i stolova i komunicira sa MongoDB-om (django-mongodb-backend). Prihvata **Stripe webhook** događaje radi ažuriranja statusa plaćanja. Radi u više worker procesa (Uvicorn/ASGI). Logika je organizovana po domenima: **accounts, menu, orders, payments, tables, waiters** uz pomoćne module (helpers).

### **MongoDB Server**

**MongoDB** je dokumentno orijentisani DBMS koji konkurentno prihvata upite i agregacije, izvršava ih nad kolekcijama i vraća rezultate ASGI aplikaciji. Koriste se kolekcije: **users, menuCategories, menuItems, orders, payments, tables, waiters**.

# 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 Servizo sistema.



## Klijent

Pristup Servizo sistemu se obavlja preko klijentskih računara i mobilnih uređaja 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. U pregledaču se izvršava **React SPA.**

## Web server

Računar na kome se izvršava Web server opslužuje više klijenata koji pristupaju preko Interneta. Web server isporučuje statičke resurse **(SPA bundle)** i radi kao reverse-proxy ka aplikaciji. U ovoj konfiguraciji na istom računaru se izvršava i **ASGI Django aplikacija** koja obrađuje **HTTP JSON zahteve u REST-stilu(plain Django).** U najopštijoj konfiguraciji DBMS se izvršava na posebnoj mašini koja je sa Web serverom u lokalnoj mreži (LAN).

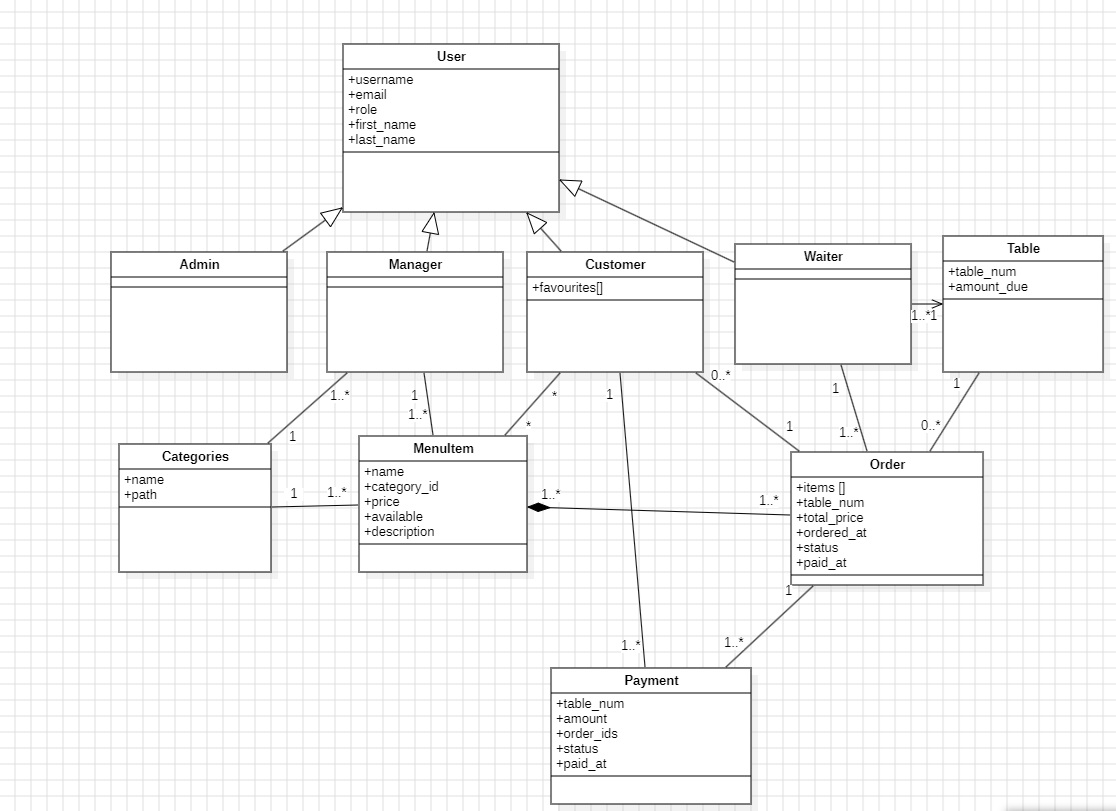
## DBMS Server

DBMS server je računar na kome se izvršava **MongoDB 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

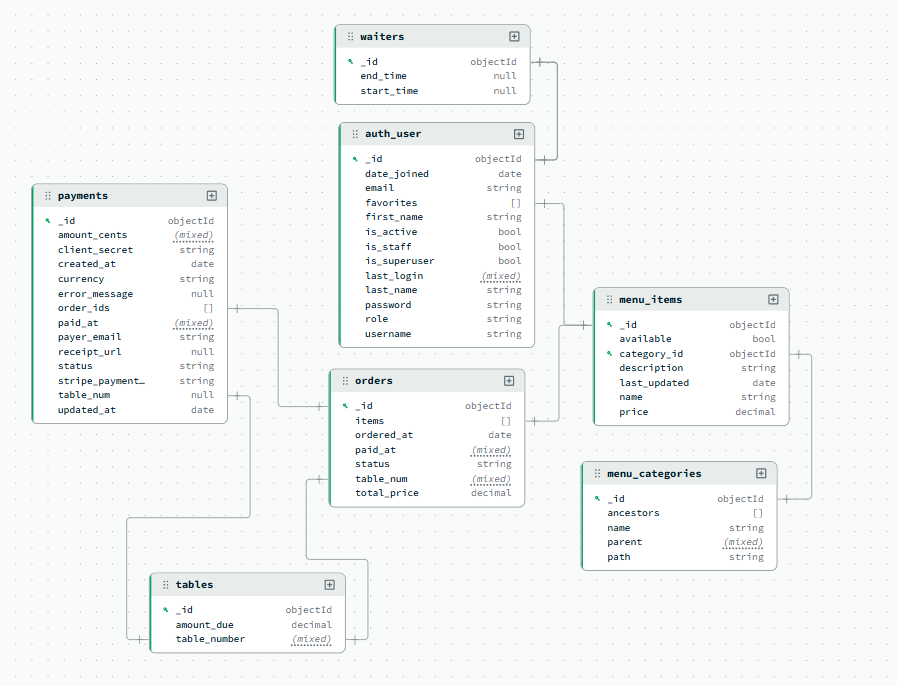
Pogled na implementaciju prikazuje različite aspekte bitne za implementaciju sistema. U slučaju Servizo sistema ovaj odeljak sadrži model domena, šemu podataka u MongoDB-u i prikaz komponenti sistema razvrstanih u ranije identifikovane pakete.

## Model domena

Model domena za koji se Servizo sistem 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 kolekcija i indeksa u bazi podataka, kao i identifikaciju komponenti **(REST-stila JSON endpointa i servisnih modula u Django-u)** koje će biti implementirane.

## Šema baze podataka

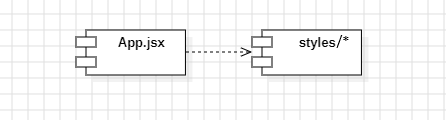
Detaljna šema baze podataka prikazana je na sledećem dijagramu. Baza je **MongoDB**, a dijagram je izvezen pomoću aplikacije **MongoDB Compass**.



## Komponente sistema

Komponente platforme Servizo su React komponente na klijentskoj strani i **REST-stila JSON endpointi** na serverskoj strani **(plain Django).** Pregled je dat po arhitekturnim slojevima, uz UML dijagrame komponenti i dijagrame klasa.

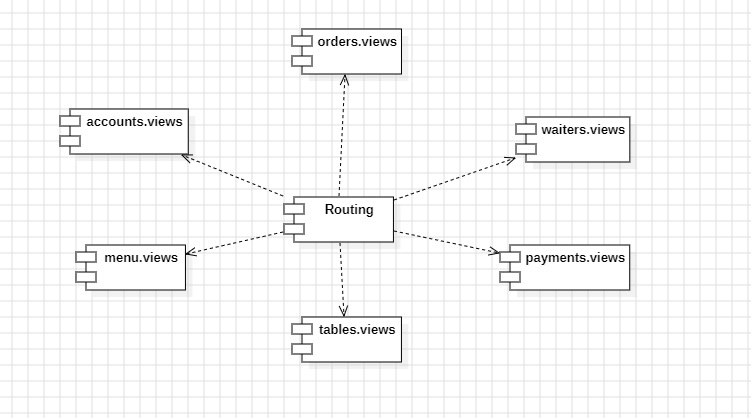
### Komponente korisničkog interfejsa



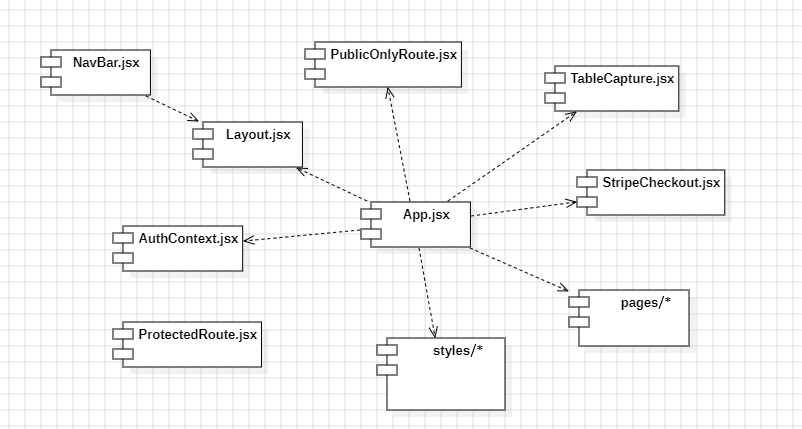
Dizajn korisničkog interfejsa obuhvata sledeće komponente:  
– **App.jsx** implementira ulaz u SPA, rutiranje i inicijalni CSRF poziv; učitava globalni stil **styles/index.css**.  
– **Layout.jsx** obezbeđuje zajednički okvir (navigacija, sadržaj) i uslovno prikazuje **NavBar.jsx**.  
– Stranice u **pages/**: Home.jsx, Login.jsx, Register.jsx, ManagerPage.jsx, Orders.jsx, Tables.jsx, Account\*.jsx, ResetPassword.jsx, ForgotPassword.jsx, Landing.jsx, NotFound.jsx.  
– Komponente u **components/**: NavBar.jsx, ProtectedRoute.jsx, PublicOnlyRoute.jsx, MenuGrid.jsx, MenuManager.jsx, CategoryManager.jsx, AddItemModal.jsx, StripeCheckout.jsx, TableCapture.jsx, LoadingIndicator.jsx, Form.jsx.  
– Konteksti u **context/**: AuthContext.jsx.  
– Stilovi u **styles/**: više CSS fajlova; **index.css** je globalan, ostali se učitavaju u odgovarajuće stranice/komponente.

### **Komponente aplikacione logike na serverskoj strani**

Komponente na serverskoj strani su **Django view-ovi** i pomoćne servisne funkcije. Rutiranje određuju **URLConf** fajlovi (*urls.py*). Poslovna logika obuhvata **validaciju ulaza**, rad sa **MongoDB** entitetima preko *django-mongodb-backend* (**ODM modeli**) i formiranje **standardizovanih JSON odgovora**. Autentikacija se oslanja na **Django sesiju**. **CSRF** je uključen za state-changing metode i front-end prosleđuje **CSRF token.** **Stripe webhook** je izuzet od CSRF.



### **Komponente aplikacione logike na klijentskoj strani**



Komponente domena uključuju se preko **App.jsx**. App.jsx centralizuje rutiranje, **AuthContext** i globalne stilove. **ProtectedRoute.jsx** i **PublicOnlyRoute.jsx** sprovode kontrolu pristupa. **TableCapture.jsx** i **StripeCheckout.jsx** pokrivaju rad sa stolovima i naplatom. Top-level stranice: Home.jsx, Landing.jsx, Login.jsx, Register.jsx, Orders.jsx, Tables.jsx, ManagerPage.jsx, Account.jsx, ForgotPassword.jsx, ResetPassword.jsx, NotFound.jsx.

### **Komponente za pristup podacima**

Pristup bazi podataka je zatvoren u servisne/repozitorijumske funkcije unutar Django aplikacija (accounts, menu, orders, payments, tables, waiters). Koristi se **MongoDB** preko *django-mongodb-backend* (**ODM**);

ne postoji direktan SQL pristup.

Operacije nad kolekcijama uključuju validaciju ulaza, **konverziju identifikatora (ObjectId)** i ažuriranje preko model-sloja.

A diagram of a computer program

AI-generated content may be incorrect.

### **Performanse**

U ovom odeljku su specificirani zahtevi koji definišu željene performanse sistema koji se razvija.

**Broj korisnika koji simultano pristupaju:**

Sistem će da podrži do 500 simultanih pristupa korisnika aplikaciji.

**Vreme odziva za pristup bazi podataka:**

Vreme potrebno za pristupanje bazi podataka u cilju izvršenje nekog upita ne sme da veće od 4,3 sekundi.

Zahtevane performanse su zadovoljene izborom tehnologija na kojima će sistem biti razvijen i definisane hardverske platforme .

# Kvalitet

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

* **Dostupnost:**
* Servizo aplikacija će biti dostupan 24 časa dnevno, 7 dana u nedelji. Vreme kada portal nije dostupan ne sme da pređe 5%.
* **Srednje vreme između otkaza:**
* Srednje vreme između dva sukcesivna otkaza ne sme da padne ispod 60 sati.