Arhitekturni projekat

FoodPin

Članovi tima:

Tea Mitić 17274
Dimitrije Mitić 17269

Sadržaj

1 Kontekst i cilj softverskog projekta

FoodPin je web aplikacija u vidu društvene mreže namenjena ljubiteljima hrane. Aplikacija pruža svojim korisnicima širok spektar raznovrsnih recepata prilagođen na osnovu njihovih interesovanja (tagova koji prate) kao i prikaz recepata njegovih prijatelja na društvenoj mreži. Korsinici mogu da kreiraju sopstvene recepte koji će biti prikazani na njihovim profilima. Samim tim, korisnici mogu videti recepte drugih korisnika na svojim početnim stranicama i sačuvati one koji im se dopadaju. Radi lakšeg oragnizovanja sopstevnih i sačuvanih recepata, i kasniju pretragu istih, na profilima je moguće grupisanje recepata u Board-ove.

Cilj ovog projekta je kreiranje zajednice u kojoj se širi znanje raznih ljudi na planeti i razmena kuliranskih iskustava iz kuhinja širom sveta, kao i želja da se ljudima približi svet hrane kako bi svakodnevnicu učinili lepšom i otkrili gurmane u sebi.

2 Arhitekturno specifični zahtevi

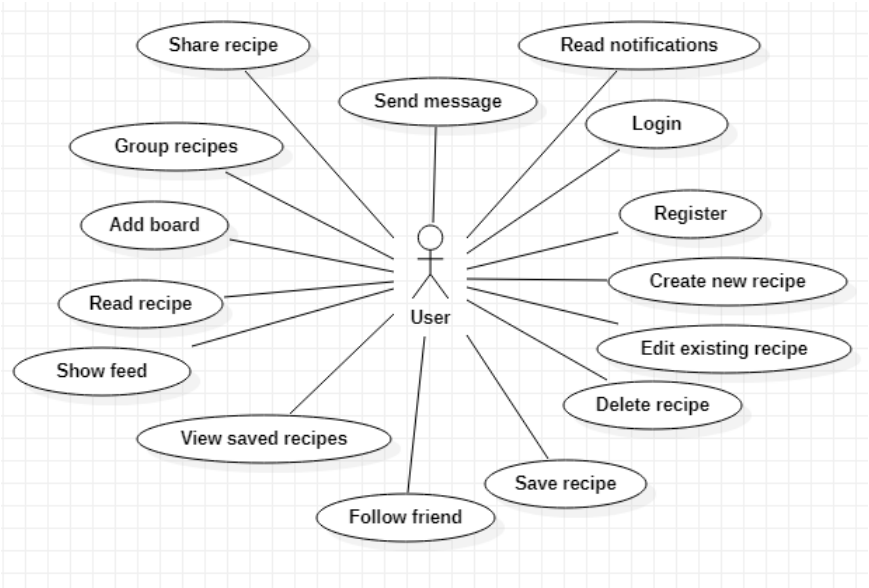
U ovom odeljku biće prikazani arhitektuni zahtevi vezani za realizaciju FoodPin aplikacije, uključujući funkcionalne zahteve, ne-funkcionalne zahteve, odnosno atribute kvaliteta i tehnička i poslovna ograničenja.

2.1 Funkcionalni zahtevi

Glavni funkcinalni zahtevi FoodPin aplikacije su:

- Kreniranje korisničkog naloga i logovanje na isti
- Dodavanje novog recepta
- Izmena dodatog recepta
- Brisanje recepta
- Čuvanje tuđeg recepta
- Pregled recepata na početnoj stranici
- Pregled izabranog recepta
- Pregled sačuvanih recepata
- Kreiranje board-a
- Grupisanje recepata u board-ove
- Deljenje recepata među korisnicima
- Komunikacija korisnika sa svojim prijateljima
- Skladištenje podataka

- Slanje notifikacija o pristiglim porukama prijatelja, kao i obaveštenja o zapraćivanju korisnikovog profila i obaveštenja da je korisnikov recept sačuvan od strane drugog korisnika.
- Zapraćivanje drugih korisnika



2.2 Ne-funkcionalni zahtevi

Pri projektovanju i realizaciji sistema teži se da se ostvare sledeći atributi kvaliteta:

- **Pouzdanost** – potrebno je da aplikacija bude perzistenta, odnosno da će određena akcija biti vidljiva svim ostalim korisnicima u nekom konačnom vremenu.
- **Sigurnost** – potrebno je da aplikacija ima implementiranu autentifikaciju i autorizaciju profila korisnika, kako bi se sprečila zloupotreba tuđih podataka
- **Perfomanse** – potrebno je da aplikacija bude interaktivna, što znači da ima što manji odziv sistema na poslate zahteve.
- **Skalabilnost** – potrebno je da aplikacija podrži povećanje broja korisnika.
- **Dostupnost** – potrebno je da aplikacija bude dostpna 24/7.
- **Modifikabilnost** – potrebno je da aplikacija bude sposobna za buduće promene sistema.
- **Upotrebljivost** – potrebno je da aplikacija bude intuitivna i laka za korišćenje.

2.3 Tehnička i poslovna ograničenja

Sistem mora biti kreiran poštujući sledeća tehnička i poslovna ograničenja:

- **Pristup preko web browser-a** – Zbog distribuirane prirode aplikacije, potrebno je implementirati sistem korišćenjem adekvatnih web tehnologija kako bi pristup korisnicima bio olakšan i time ukinuli potrebu za instalacijom na njihovim uređajima.
- **Sinhrona i asinhrona komunikacija** – Sistem treba da podrži sinhronu komunikaciju između klijentskog dela i serverskog dela pri normalnom korišćenju aplikacije, kao i asinhronu komunikaciju između klijenata prilikom dopisivanja i primanja notifikacija.
- **Skrivenost baze podataka –** Potrebno je sakriti način reprezentacije samih podataka u bazi i obezbediti da svaki klijent dobije odgovarajuće podatke.
- **Laka upotrebljivost aplikacije –** Sistem treba biti interaktivne prirode i intiutivan za korišćenje bez dodatnih korisničkih upustava.

3 Arhitekturni dizajn

U ovom delu biće opisan arhitekturni dizajn FoodPin aplikacije, koji uključuje arhitekturne obrasce, generalnu arhitekturu sistema, kao i osnovne dijagrame komponenti i njihovih veza korišćenjem strukturnih i bihevioralnih pogleda.

3.1 Arhitekturni obrasci

FoodPin aplikacija se sastoji od sledećih arhitekturnih obrazaca:

- *Layered obrazac*

  Glavna struktura aplikacije FoodPin biće odrađena po Layered obrascu koji će biti podeljen na tri glavna sloja, to su: klijentski sloj, serverski sloj, i sloj baze podataka.

  Klijentska komponenta predstavlja interfejs prema korisnicima preko web browser-a.

  Serverski sloj je srednji sloj sistema koji povezuje klijentsku stranu i bazu podataka. Ovaj sloj će biti dalje raščlanjen u tri podsloja.

  1. Prvi sloj je sloj prihvatanja api poziva sa klijentske komponente što omogućava interakciju servera i klijenta.
  2. Drugi sloj je sloj poslovne logike koji je moguće obraditi sinhrono, komunikacija sa serverom pomoću RESTful API poziva, i asinhrono, korišćenjem usluga message broker-a.
  3. Treći sloj je sloj perzistencije i upravljanja klijentskih podataka u bazi podataka.

Sloj baze podataka je namenjen trajnoj perzistenciji podataka. Ovaj sloj je sačinjen od dve baze podataka: graf baza (zadužena za konekcije između samih podataka i skladištenje većeg dela podataka) i relaciona baza (zadužena za skladištenje notifikacija i poruka između klijenata)

- *Model-View-ViewModel*
  Model-View-ViewModel obrazac će biti implicitno implementiran korišćenjem Vue.js tehnologije (koja koristi sam taj obrazac) na klijentskoj strani aplikacije.

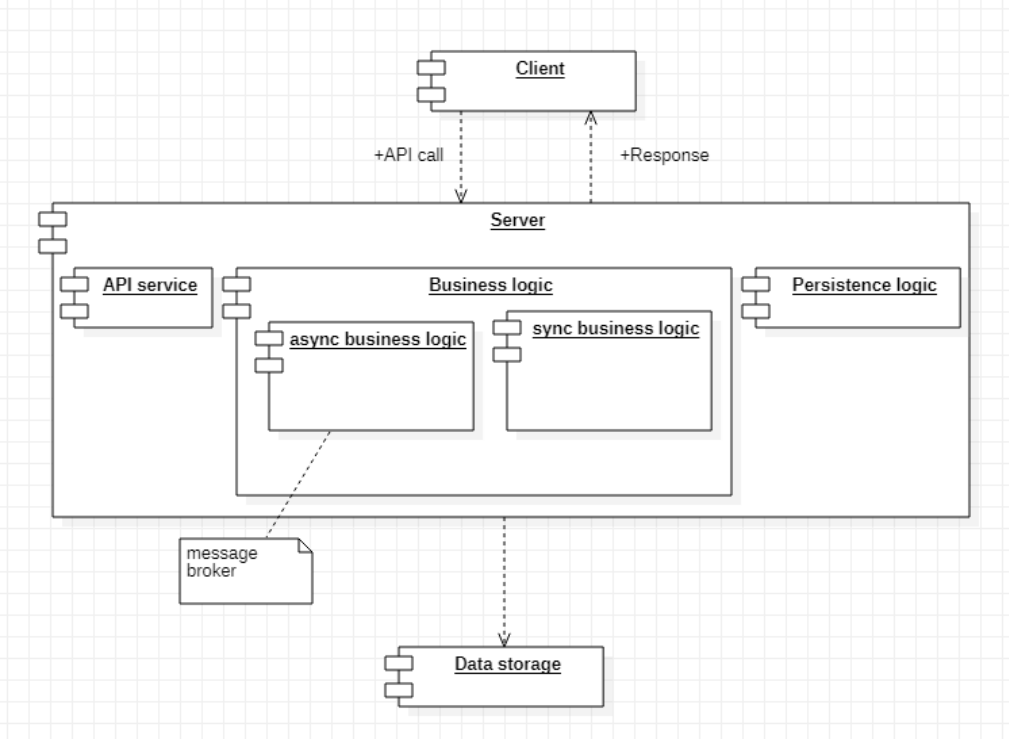
- *Publisher/Subscriber obrazac*

  Publisher/Subscriber obrazac će biti iskorišćen u vidu message broker-a za implementaciju asinhrone komunikacije. Komunikacija obuhvata obaveštenja korisnika o pristiglim porukama od drugih korisnika, kao i sistemske notifikacije prouzrokovane akcijama drugih korisnika (kao što su *Follow* notifikacija i *Save Recipe* notifikacija). Klijenti će biti automatski obavešteni o svim navedenim promenama u sistemu koji se njega tiču.

- *Repository obrazac*
  Repository obrazac će biti iskorišćen za apstrakciju database layer-a kako bi sav pristup kodu za interakciju sa bazom (izvršavane upita) bio lokalizovan, što redukuje ponavljanje koda database sloja. Time se olakšava izmena trenutnog i dodavanje novog koda, a samim tim se pruža interfejs višem sloju aplikacije (sloju biznis logike).Ovaj pattern će biti iskorišćen za dve baze, graf bazu i relacionu bazu.

3.2 Generalna arhitektura

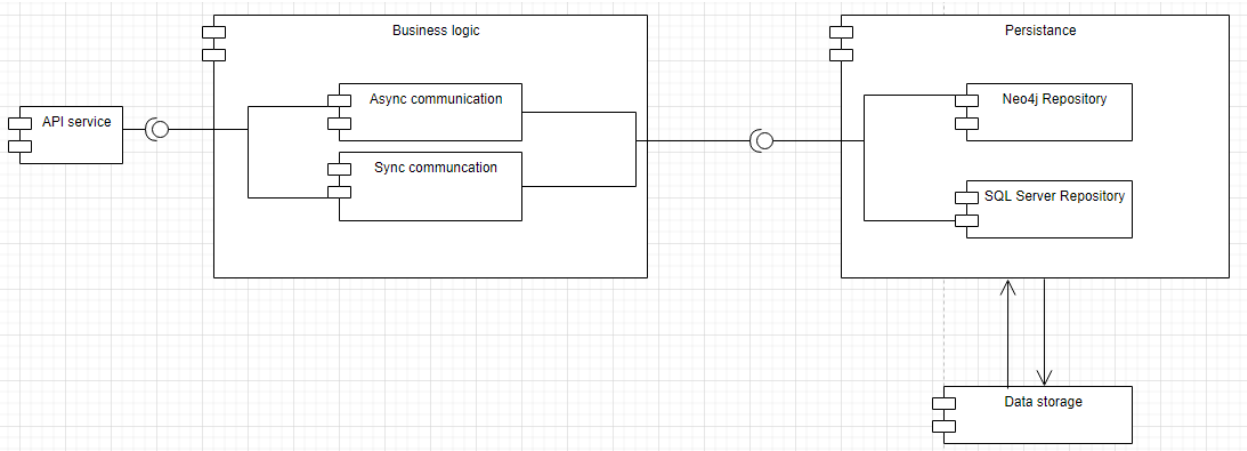
Generalna arhitektura je opisana sledećim dijagramom:



3.3 Strukturni pogled

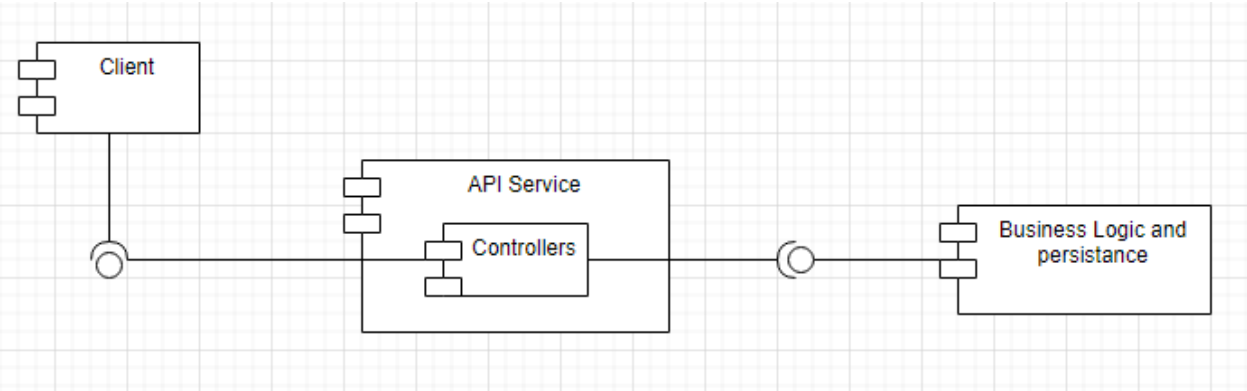
Komponetni dijagram serverske komponente i povezanost serverskih podkomponenti.

Serverska komponenta se deli na komponentu za poslovnu logiku i komponentu perzistencije. Komponenta za poslovnu logiku se dalje deli na dve podkomponente. Zaduženje prve podkomponente je da primeni poslovnu logiku nad akcijama same aplikacije, a zaduženje druge podkomponente je da obezbedi asinhronu komunikaciju sa serverom pomoću message broker-a. Komponenta za perzistenciju obezbeđuje dvosmernu komunikaciju sa skladištem podataka koja se sastoji od dve baze: SQL Server relaciona baza, koja skladišti notifikacije i poruke između korisnika, i Neo4J graf baza koja skladišti sve druge informacije ove aplikacije.



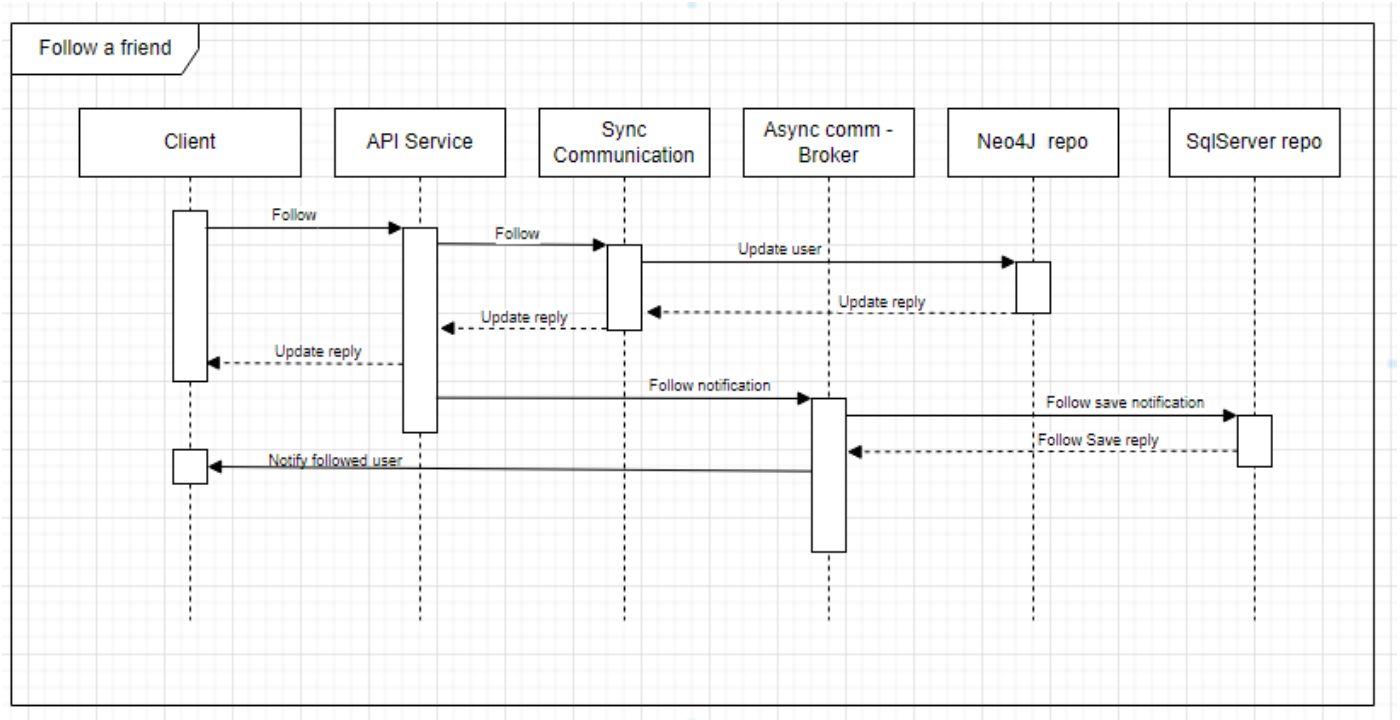
Komponentni dijagram API komponente

Krajnji korisnici koriste usluge sistema pozivajući API funkcije smeštene na serveru. API funkcije predstavljaju endpoint-ove tj. pristupne tačke sa serverom aplikacije. API funkcije delegiraju zaduženje sloju biznis logike i slojevima ispod, čekaju na odgovor od nižih slojeva i vraćaju odgovor klijentu koji je poslao API zahtev za nekom funkcionalnošću.



3.4 Bihevioralni pogledi

Navedeni dijagram prikazuje sled akcija kada jedan korisnik zaprati drugog korisnika društvene mreže FoodPin. Šalje se zahtev za praćenje api servisu, koji prosleđuje zahtev poslovnom sloju. Poslovni sloj obrađuje zahtev i komunicira sa graf bazom podataka u kojoj se stvara konekcija između dva čvora (klijent i novi prijatelj). Api servis istovremeno šalje zahtev message broker-u koji komunicira sa relacionom bazom podataka u kojoj će se sačuvati notifikacija o praćenju. Nakon toga, message broker šalje notifikaciju klijentu da ga je drugi klijent zapratio.



3.5 Implementaciona pitanja

U ovom delu navedene su biblioteke, komponente i okviri (frameworks) koji će biti korišćeni za implementaciju FoodPin aplikacije.

Vue.js – framework za izradu klijentskih SPA web aplikacija

Node.js – framework za izradu backend-a aplikacije

Express.js – dodatni framework za Node.js radi lakšeg kreiranja API Servisa

SQLServer – relaciona baza podataka

Neo4J – graf baza podataka

4 Analiza arhitekture

Potencijalni problemi koji se mogu pojaviti su:

Sistem planira da koristi jedan server za serverski deo aplikacije i jedan server za skladištenje podataka. Pri povećanju broja korisnika i njihovih interakcija, može doći do prevazilaženja granica ovih servera, pri čemu serveri postaju potencijalno usko grlo. Jedan od načina prevazilaženja ovog problema za serverski deo aplikacije je instalacija dodatnih čvorova i kreiranje distribuiranog sistema, pri čemu će se teret ravnomerno rasporediti. Što se tiče servera za skladištenje podataka, on takođe može dostići svoj maksimalni kapacitet pa je opcija za rešavanja ovog problema razdvajanje skladišta na dva čvora, jedan samo za SQL server, a drugi samo za Neo4J.

Povećanje broja korisnika može da dovede do problema geografske udaljenosti, što znači da će pojedinim korisnicima zahtevi biti sporije obrađivani. Jedan od rešenja za ovakav problem je replikacija svih servera na više geograskih lokacija.