Programsko inženjerstvo

Ak. god. 2023./2024.

CookBooked

Dokumentacija, Rev. *2*

Grupa: *CodeCooks*

Voditelj: *Antonio Hohnjec*

Datum predaje: *19.1.2024.*

Nastavnik: *Nikolina Frid*

Sadržaj

1. Dnevnik promjena dokumentacije

Kontinuirano osvježavanje

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Učitan predložak dokumentacije.	Antonio Hohnjec	19.10.2023.
0.2	Dopunjeni nazivi i sudionici projekta. Nadopunjeno poglavlje ”2. Opis projektnog zadatka”	Antonio Hohnjec	25.10.2023.
0.3	Dodan *Use Case* dijagrami i sekvencijski dijagrami, nefunkcionalni zahtjevi i ostali zahtjevi te arhitektura i dizajn sustava sa bazom podataka i prikladnim tablicama te dijagramom baze podataka.	Benjamin Zrakić, Armis Kantarević, Luka Ivčević, Filip Vrbić, Marko Pavić	30.10.2023.
0.4	Upis poglavlja Arhitektura i dizajn sustava. Dodani dijagrami razreda za prvu predaju	Armis Kantarević, Antonio Zglavnik	9.11.2023.
1.1	Popravljene greške uočene prilikom prve predaje	Benjamin Zrakić	15.12.2023.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	Opis promjene/dodatka	Autori	Datum
1.2	Dodani dijagrami stanja i aktivnosti, ispunjena i poglavlja Korištene tehnologije i alati te upute za puštanje u pogon	Luka Ivčević, Antonio Hohnjec	4.1.2024.
1.3	Dodani dijagrami komponenti i razmještaja te ispunjeno poglavlje Zaključak i budući rad	Filip Vrbić, Antonio Hohnjec	15.1.2024.
1.4	Ispunjeno poglavlje Ispitivanje programskog rješenja i Dijagrami pregleda promjena	Antonio Hohnjec	18.1.2024.

2. Opis projektnog zadatka

CookBooked je projekt u kojem je potrebno razviti web aplikaciju koja omogućava korisnicima razmjenu recepata za kuhanje i pečenje kolača te povezivanje s autorima recepata.

Neregistrirani korisnici mogu samo pregledavati recepte temeljem kategorija, vrsta kuhinje ili specifičnih sastojaka. Za pristup svim ostalim mogućnostima platforme, korisnici se moraju registrirati s važećom adresom e-pošte.

Autori recepata imaju opciju komunikacije s ostalim korisnicima vezano za svoje recepte, kao npr. razmjenu poruka, čavrljanje ili video pozive. Ove značajke omogućuju korisnicima da se povežu s autorima recepata, ali su dostupne samo registriranim korisnicima. Autori recepata također mogu postaviti termine kada su dostupni za komunikaciju (npr. određene sate ili dane).

Registrirani korisnici mogu označavati, komentirati i spremiti recepte za buduću referencu. Korisnici mogu pratiti svoje omiljene autore recepata kako bi primili obavijest o novim receptima.

Registrirani korisnici imaju javne profile na kojima prikazuju svoje objavljene recepte, pratitelj i autore koje prate. Također imaju privatne profile gdje mogu upravljati osobnim informacijama, postavkama komunikacije i obavijestima za poruke i aktivnosti povezane s receptima.

Platformu održavaju sistemski administratori koji mogu upravljati korisnicima, mijenjati kategorije recepata ili brisati recepte.

2.1 Opis ideje aplikacije

CookBooked je aplikacija koja omogućava korisnicima razmjenu recepata za kuhanje i pečenje kolača te povezivanje s autorima recepata. Sam potencijal ove aplika-

cije leži u dobro razvijenom te interaktivnom sučelju koje omogućava registriranim korisnicima lako objavljivanje svojih recepata, dok u isto vrijeme omogućava nesmetano korištenje i pregled samih tih recepata bez mogućnosti objave.

Danas već postoje neke stranice za objavu recepata kao što su Coolinarka, ReciPeci, Zdrave navike i brojne druge. Iako su one već dobrim djelom razvijene te koriste brojne alate, aplikacija CookBooked omogućava veću personalizaciju, odabir jelovnika za pojedini dan te kategorizira jela i namirnice na načine koji su puno bliži i lakši za snalaženje svakom korisniku.

Slika 2.1: Coolinarka

Slika 2.2: ReciPeci

U prijašnjem odlomku spomenuli smo prednost odlične kategorizacije ove aplikacije pa tako gdje se ta prednost pokazuje jest upravo tako što imamo kategorije hrane za mlade, kategorije hrane za osobe sa bolestima te za starije osobe kojima

odgovara lagana prehrana.

Tako naša aplikacija zahvaća razne kategorije ljudi kao što su:

- *mlađe osobe koje tek uče kuhati*
- *osobe koje ne stignu raditi duge pripreme*
- *starije osobe koje ne mogu pripremati i konzumirati svu hranu*
- *odrasli koji žele naučiti kuhati*
- *osobe koje žele izraditi kompliciranije stvari sa nešto manje kulinarskog iskustva*

Sama aplikacija od razvoja pa do konačne aplikacije te nakon toga vrlo je lako prilagodljiva. Osim alata pisanja izvornog koda, dijelove aplikacije lako je prilagoditi u smislu izgleda, dodavanja novih kategorija, dodatnih opcija privatnog profila. Ako želimo znati zašto, to je jednostavno. Cijela aplikacija je u kodu strukturirana tako da se koriste strukture podataka koje je onda u slučaju bilo kakve modifikacije potrebno promijeniti samo na jednom mjestu.

Što se samih promjena tiče, aplikacija se razvija iterativno. Kod razvoja aplikacije u pogon se pušta prvo prototip koji sadrži osnovnu funkcionalnost zbog potrebe provjere ispravnosti aplikacije. U sljedećim verzijama aplikacija se nadograđuje profilima korisnika, poslovnim i privatnim profilima pa se nadalje dodaje i kategorije, izbornici, i slično te u konačnici mogućnost komunikacije između autora i recenzije njihovih recepata.

Nakon završetka postavljenih ciljeva pri izradi aplikacije, zbog same strukture i čitljivosti pri pisanju, aplikaciju je u naknadnim verzijama moguće nadograditi sa brojnim drugim značajkama.
Primjer takvih značajki:

- *komunikacija unutar aplikacije putem videopoziva*
- *AI pomoć pri odabiru recepta i kategorije*
- *AI stvaranje jelovnika prilagođenog za dan i posebne prilike*
- *Stvaranje odjela za događaje izrade i objave novih recepata*
- *Tečajevi za kuhanje putem videa i videopoziva*

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Administrator-voditelj tima
2. Razvojni tim CodeCooks
3. Profesori predmeta Programsko inženjerstvo
4. Korisnici

Aktori i njihovi funkcionalni zahtjevi:

1. Admin može:
   (a) Upravljanje korisnicima
   (b) Upravljanje receptima
      i. Brisanje recepata
      ii. Uređivanje recepata
   (c) Upravljanje komentarima
      i. Brisanje komentara
      ii. Cenzuriranje komentara
   (d) Mijenjanje kategorija
2. Registrirani korisnik može:
   (a) Objavljivanje recepata
   (b) Komuniciranje s ostalim registriranim korisnicima
   (c) Objavljivanje termina za komunikaciju
   (d) Označavanje recepata
   (e) Komentiranje recepata
   (f) Spremanje recepata
   (g) Praćenje drugih registriranih korisnika
   (h) Upravljanje dozvolama pratiteljima
   (i) Upravljanje osobnim informacijama

(j) Upravljanje postavkama komunikacije i obavijestima

3. Neregistrirani korisnik može:

   (a) Pregledavanje recepata temeljem kategorija
   (b) Pregledavanje recepata na početnoj stranici platforme

4. Baza podataka može:

   (a) Pohrana osobnih podataka registriranog korisnika
   (b) Pohrana podataka o pratiteljima i pratiocima korisnika
   (c) Pohrana recepata
   (d) Pohrana komentara

3.1.1 Obrasci uporabe

UC1 - Pregled recepata

- **Glavni sudionik: Neregistriran korisnik, Registriran korisnik**
- **Cilj: Pregled recepata**
- **Sudionici: Baza podataka**
- **Preduvjet: -**
- **Opis osnovnog tijeka:**
  1. Korisnik pristupi platformi
  2. S baze podataka se na platformu dohvaća popis recepata, kategorija, vrsta kuhinje i sastojaka
  3. Korisnik može odabrati na koji način želi pregledati recepte i ovisno o tome se oni učitavaju na platformi

UC2 - Registracija

- **Glavni sudionik: Neregistrirani korisnik**
- **Cilj: Stvaranje računa za pristup ostalim značajkama sustava**
- **Sudionici: Baza podataka**
- **Preduvjet:-**
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za registraciju
  2. Korisnik unosi potrebne korisničke podatke i odabire unos
  3. Podaci se šalju sustavu i on provjerava jesu li uneseni svi potrebni podaci i postoji li već korisnik s istim imenom
  4. Ako je sve u redu sustav sprema novog korisnika u bazu podataka i šalje povratnu obavijest o registraciji
  5. Korisnik je automatski nakon registracije prijavljen u sustav i može pregledati ostale značajke sustava
- **Opis mogućih odstupanja:**

  2.a Odabir već zauzetog korisničkog imena i/ili e-maila, unos korisničkog podatka u nedozvoljenom formatu ili pružanje neispravnoga e-maila
  1. Sustav obavještava korisnika o neuspjelom upisu i vraća ga na stranicu za registraciju
  2. Korisnik mijenja potrebne podatke te završava unos ili odustaje od registracije

UC3 - Prijava u sustav

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Omogućiti pristup ostali funkcijama sustava za korisnika**
- **Sudionici: Baza podataka**
- **Preduvjet: Registracija**
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za prijavu u sustav
  2. Na stranici za prijavu unosi korisničkog ime i lozinku i pritisne gumb za prijavu
  3. Podaci za prijavu šalju se sustavu i on provjera postoji li korisnik u bazi podataka koji ima isto korisničko ime i lozinku
  4. Ako je sve u redu korisniku se šalje potvrda o ispravnosti unesenih podataka i prijavljen je u sustav
  5. Nakon toga može pristupiti ostalim funkcijama sustava
- **Opis mogućih odstupanja:**

  2.a Neispravno korisničko ime/lozinka
  1. Sustav obavještava korisnika o neuspjelom upisu i vraća ga na stranicu za prijavu

UC4 - Objava recepata

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Objava vlastitog recepta na sustav**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen**
- **Opis osnovnog tijeka:**
  1. Korisnik odabire opciju za stvaranje novog recepta
  2. Na stranici za stvaranje recepta ispunjava formu gdje treba navesti naslov, sastojke, korake pripreme i vrijeme kuhanja
  3. Opcionalno može dodati oznake poput „vegetarijansko“, „bezglutensko“ i sl.
  4. Opcionalno može dodati slike i videozapise koji se odnose na pripremu jela
  5. Korisnik odabire objavu recepta
  6. Podaci se šalju sustav i on provjerava jesu li svi podatci u redu, ako jesu recept se sprema u bazu podataka
  7. Korisnik prima obavijest da je uspješno objavio recept i može ga ići pregledati

- **Opis mogućih odstupanja:**
  - 2.a Korisnik nije naveo naslov, sastojke, korake pripreme i vrijeme kuhanja
    1. Sustav obavještava korisnika o neuspjelom upisu u bazu i vraća ga na stranicu za objavu recepata (UC4.2)

UC5 - Uređivanje recepata od strane korisnika

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Uređivanje svog objavljenog recepta**
- **Sudionici: Baza podataka**
- **Preduvjet: Objavljen je recept i korisnik je prijavljen**
- **Opis osnovnog tijeka:**
  1. Korisnik na stranici recepta pregledava svoj recept
  2. Korisnik može odabrati gumb za uređivanje recepta
  3. Otvori se forma gdje korisnik može uređivati recepta kao pri objavi
  4. Korisnik može odabrati spremanje recepta ili odustajanje od promjena
  5. Ako je korisnik odlučio spremiti recept podaci o ažuriranom receptu šalju se sustavu koji provjerava jesu li u redu i ako jesu sprema se u bazu podataka, a ako je korisnik odustao od promjena korisnika se vraća na stranicu recepta

UC6 - Brisanje recepata od strane korisnika

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Brisanje svog objavljenog recepta**
- **Sudionici: Baza podataka**
- **Preduvjet: Objavljen je recept i korisnik je prijavljen**
- **Opis osnovnog tijeka:**
  1. Korisnik na stranici recepta pregledava svoj recept
  2. Korisnik može odabrati gumb za brisanje recepta
  3. U skočnom prozoru potvrđuje ili odustaje od brisanja recepta
  4. Ako je korisnik odlučio obrisati recept zahtjev se šalje u sustav koji obriše recept iz baze podataka, a ako je odustao vraća se na stranicu recepta

UC7 - Razmjena poruka

- **Glavni sudionik: Registrirani korisnici i oni koji su objavili recept**
- **Cilj: Komunikacija između korisnika u vezi recepata**
- **Sudionici: Baza podataka, Firebase**

- **Preduvjet: Korisnici su prijavljeni, autor je dostupan i ima objavljen recept**
- **Opis osnovnog tijeka:**
  1. Korisnik pregledava profil drugog korisnika gdje može odabrati gumb za početak razmjene poruka
  2. Korisniku se prikaže forma gdje može upisati predmet poruke i sadržaj poruke te ju poslati korisniku
  3. Sustav provjerava je li sve u redu s porukom i ako je ju sprema na Firebase poslužitelj preko kojega se onda poruka dostavlja drugome korisniku
  4. Drugi korisnik prima obavijest o primljenoj poruci i može ju pregledati te na nju odgovoriti

UC8 - Čavrljanje

- **Glavni sudionik: Registrirani korisnici i oni koji su objavili recept**
- **Cilj: Komunikacija između korisnika u vezi recepata**
- **Sudionici: Baza podataka, Firebase**
- **Preduvjet: Korisnici su prijavljeni, autor je dostupan i ima objavljen recept**
- **Opis osnovnog tijeka:**
  1. Korisnik pregledava profil drugog korisnika gdje može odabrati gumb za čavrljanje
  2. Ovaj zahtjev se pohranjuje na Firebase poslužitelj i šalje se zahtjev za čavrljanje drugom korisniku
  3. Drugi korisnik treba prihvatiti ili odbiti zahtjev za čavrljanje, ova informacija se onda pohranjuje na Firebase poslužitelj
  4. Ako je uspostavljenja komunikacija otvori se prozor za čavrljanje gdje korisnici mogu razmjenjivati poruke
- **Opis mogućih odstupanja:**

  2.a Autor nije dostupan u to doba za čavrljanje
    1. Korisnika obavještavamo da autor nije dostupan i vraćamo ga na pregled recepta

UC9 - Video poziv

- **Glavni sudionik: Registrirani korisnici i oni koji su objavili recept**
- **Cilj: Komunikacija između korisnika u vezi recepata**

- **Sudionici: Baza podataka, Firebase**
- **Preduvjet: Korisnici su prijavljeni, autor je dostupan i ima objavljen recept**
- **Opis osnovnog tijeka:**
    1. Korisnik pregledava profil drugog korisnika gdje može odabrati gumb za video poziv
    2. Zahtjev se sprema na Firebase poslužitelj i šalje drugom korisniku
    3. Drugi korisnik prima zahtjev za videopozivom i može prihvatiti ili odbiti poziv
    4. Ako je poziv prihvaćen traje sve dok ga jedan od korisnika ne prekine
- **Opis mogućih odstupanja:**
    2.a Autor nije dostupan u to doba za videopoziv
        1. Korisnika obavještavamo da autor nije dostupan i vraćamo ga na pregled recepta

UC10 - Označavanje recepata

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Označavanje da nam se recept svidio**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen**
- **Opis osnovnog tijeka:**
    1. Korisnik pregledava stranicu recept i može stisnuti gumb kako bi označio da mu se recept svidio
    2. Kada korisnik pritisne gumb sustav ovu informaciju šalje u bazu podataka i poveća se broj oznaka za pregledani recept

UC11 - Komentiranje recepata

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Postavljanje komentara na recept**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen i postoji recept**
- **Opis osnovnog tijeka:**
    1. Korisnik pregledava stranicu za recept gdje na dnu stranice postoji prostor gdje se prikazuju komentari, a uz njih i forma gdje korisnik može dodati svoj novi komentar

2. Kada korisnik ispuni formu podaci se šalju sustavu i ako su u redu pohranjuju se u bazu podataka te nakon toga prikazuju na stranici recepta

- **Opis mogućih odstupanja:**

2.a Prekršili smo ograničenje na broju znakova u komentaru

1. Obavještavamo korisnika o maksimalnom broju znakova i ne dopuštamo mu objaviti komentar

UC12 - Spremanje recepata za buduću referencu

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Spremanje recepta da ga korisnik može ponovno lako pronaći**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen i postoji recept**
- **Opis osnovnog tijeka:**
  1. Korisnik pregledava stranicu recepta i može odabrati gumb za spremanje recepta
  2. Ako korisnik ovo odabere ova informacija se šalje sustavu koji je onda pohranjuje u bazu podataka
  3. Ako je spremanje uspješno korisnik može pregledati svoje spremljene recepte na zasebnoj stranici

UC13 - Uklanjanje oznake recepata

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Uklanjanje recepta iz favorita**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen**
- **Opis osnovnog tijeka:**
  1. Korisnik pregledava stranicu recepta kojeg je već označio
  2. Korisnik oznaku može ukloniti ponovnim pritiskom na gumb za označavanje recepta
  3. Ako korisnik ovo pritisne ova informacija se šalje u bazu podataka i mijenja se broj oznaka za pregledavani recept

UC14 - Uklanjanje komentara

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Postavljanje komentara na recept**
- **Sudionici: Baza podataka**

- **Preduvjet: Korisnik je prijavljen i postoji recept**
- **Opis osnovnog tijeka:**
  1. Korisnik pregledava stranicu za recept gdje na dnu stranice postoji prostor gdje se prikazuju komentari, a uz njih može vidjeti i svoj komentar
  2. Na svome komentaru korisnik može vidjet gumb za uklanjanje komentar i ako ga pritisne šalju se informacije sustavu da treba obrisati taj komentar
  3. Kada sustav primi ovu informaciju briše traženi komentar iz baze podataka i on se više neće prikazivati na stranici

UC15 - Uklanjanje spremljenog recepta

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Spremanje recepta da ga korisnik može ponovno lako pronaći**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen i postoji recept**
- **Opis osnovnog tijeka:**
  1. Korisnik pregledava stranicu recepta kojeg je već spremio
  2. Korisnik može ponovno pritisnuti gumb za spremanje recepta kako bi ga uklonio iz svojih spremljenih recepata
  3. Kada korisnik ovo pritisne sustav obrađuje ovaj zahtjev i uklanja spremljeni recept iz tablice spremljenih recepata tog korisnika u bazi podataka

UC16 - Praćenje omiljenih autora recepata

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Praćenje omiljenih autora kako bi korisnik dobio obavijesti o novim receptima**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen**
- **Opis osnovnog tijeka:**
  1. Korisnik pregledava profil drugog korisnika na kojem se nalazi gumb za praćenje profila
  2. Ako korisnik pritisne gumb sustav obrađuje taj zahtjev i pohranjuje drugog korisnika kao profil koji prvi korisnik prati u tablicu pratioca, a uz to pohranjuje i prvog korisnika u tablicu pratitelja drugog korisnika u bazi podataka
  3. Korisnik sada prati drugog korisnika

UC17 - Uklanjanje praćenja omiljenih autora recepata

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Uklanjanje praćenja omiljenih autora recepata**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen**
- **Opis osnovnog tijeka:**
  1. Korisnik pregledava profil drugog korisnika kojeg već prati na kojem se nalazi gumb za praćenje profila
  2. Ako korisnik ponovno pritisne gumb sustav obrađuje taj zahtjev i uklanja informacije o praćenju tog profila iz baze podataka
  3. Korisnik više ne prati drugog korisnika

UC18 - Slanje obavijesti o novim receptima omiljenih autora

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Obavještavanje korisnika o novim objavama recepata**
- **Sudionici: Baza podataka**
- **Preduvjet: Korisnik je prijavljen i prati neke autore**
- **Opis osnovnog tijeka:**
  1. U aplikaciji postoji dio za obavijest, ima ikonicu zvonca i ako ima novih obavijest pored zvonca se pojavi trenutni broj obavijest, osim toga se korisnika obavijesti i notifikacijom
  2. Kada profil kojeg korisnik prati objavi nešto novo taj profil će dohvatiti popis svojih pratitelja iz baze podataka i poslati svima obavijest da je dodan novi recept

UC19 - Pregled javnog profila

- **Glavni sudionik: Neregistrirani korisnik, Registrirani korisnik**
- **Cilj: Pregled javnog profila nekog korisnika**
- **Sudionici: Baza podataka**
- **Preduvjet: -**
- **Opis osnovnog tijeka:**
  1. Korisnik odabire profil nekog korisnika za pregled preko recepta, može odabrati i vlastiti
  2. Sustav iz baze podataka dohvaća podatke o profilu korisnika kojeg želimo pregledati i učitava ih na stranici

3. Tamo korisnik može pregledati objavljene recepte, pratitelje i autore koje taj korisnik prati i koji prate njega
4. Ako profil pripada korisniku na profilu će se pojaviti gumb za pristup privatnom profilu gdje korisnik može mijenjati postavke, inače će se prikazati gumbi za komunikaciju i praćenje profila

UC20 - Pregled i promjena postavki privatnog profila

- **Glavni sudionik: Registrirani korisnik**
- **Cilj: Pregled vlastitog privatnog profila**
- **Sudionici: Baza podataka**
- **Preduvjet: Prijava korisnika**
- **Opis osnovnog tijeka:**
  1. Korisnik tijekom pregleda svog profila odabira pristup privatnom profilu
  2. Sustav iz baze podataka dohvaća podatke o profilu korisnika i učita ih u formu
  3. Tamo korisnik može pregledati osobne informacije, postavke komunikacije, obavijesti za poruke i aktivnosti povezane s receptima te izmjenjivati podatke prema želji
  4. Nakon izmjene ih može pohraniti ili odustati od pohrane
  5. Ako ih je odlučio pohraniti podaci se šalju sustavu koji ih obrađuje i provjerava te ako je sve uredu ažurira u bazi podataka

UC21- Upravljanje korisnicima od strane sistemskom administratora

- **Glavni sudionik: Sistemski administrator**
- **Cilj: Upravljanje korisnicima od strane sistemskom administratora**
- **Sudionici: Baza podataka**
- **Preduvjet: Postoje registrirani korisnici**
- **Opis osnovnog tijeka:**
  1. Administrator pregledava profil nekog korisnika na kojem se nalaze gumbi za uređivanje profila korisnika te za brisanje profila korisnika
  2. Ako administrator odabere uređivanje profila korisnika otvorila bi se forma u sličnom obliku kao i kod uređivanja privatnog profila korisnika gdje administrator može izmijeniti podatke korisnika te ih zatim pohraniti
  3. Nakon što ih pohrani sustav ih obrađuje i ažurira u bazi podataka

4. Ako administrator odabere brisanje korisničkog profila sustav šalje zahtjev za uklanjanje profila iz baze podataka

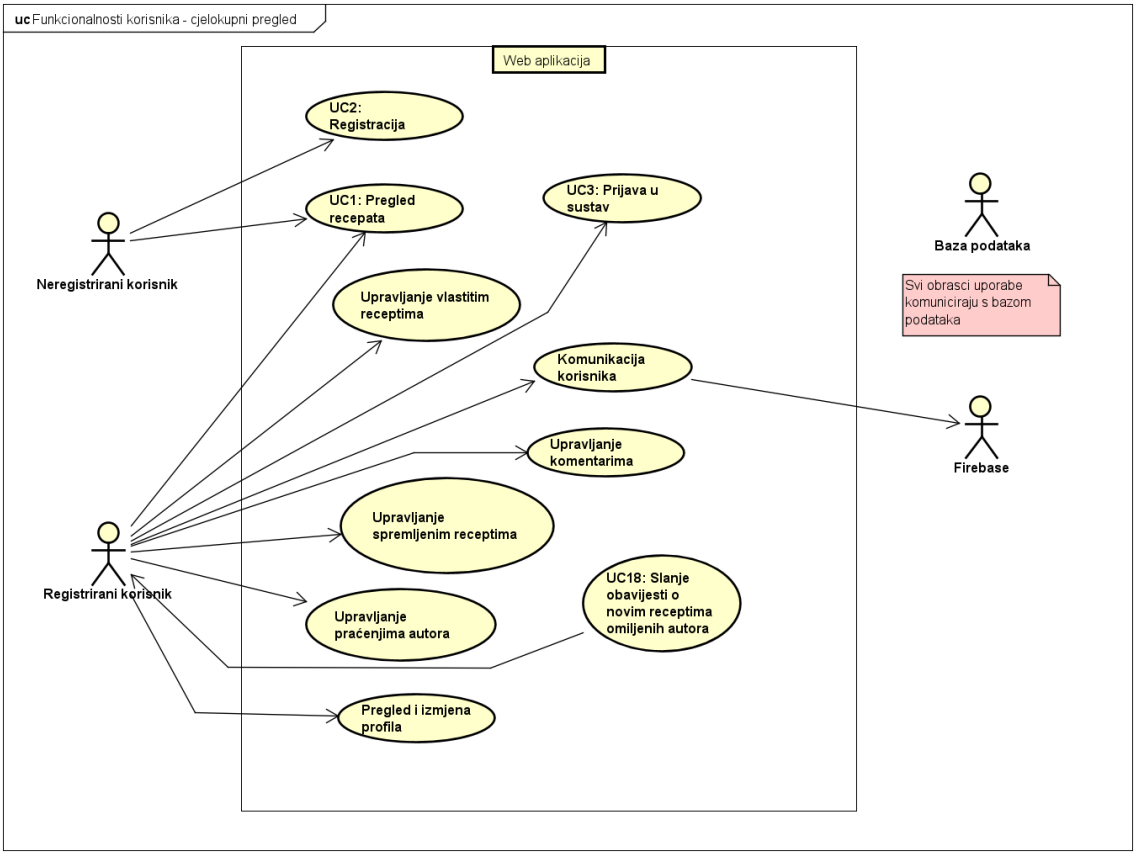
UC22 - Mijenjanje recepata od strane sistemskih administratora

- **Glavni sudionik: Sistemski administrator**
- **Cilj: Uređivanje (kategorija) recepta nekog korisnika**
- **Sudionici: Baza podataka**
- **Preduvjet: Postoji objavljeni recept**
- **Opis osnovnog tijeka:**
  1. Sistemski administrator pregledava recept na kojem postoji gumb za uređivanje recepta
  2. Ako administrator odabere navedeni gumb otvorit će mu se forma gdje može mijenjati podatke o receptu te ih zatim pohraniti
  3. Ako ih odluči pohraniti sustav ih obrađuje i ažurira u bazi podataka

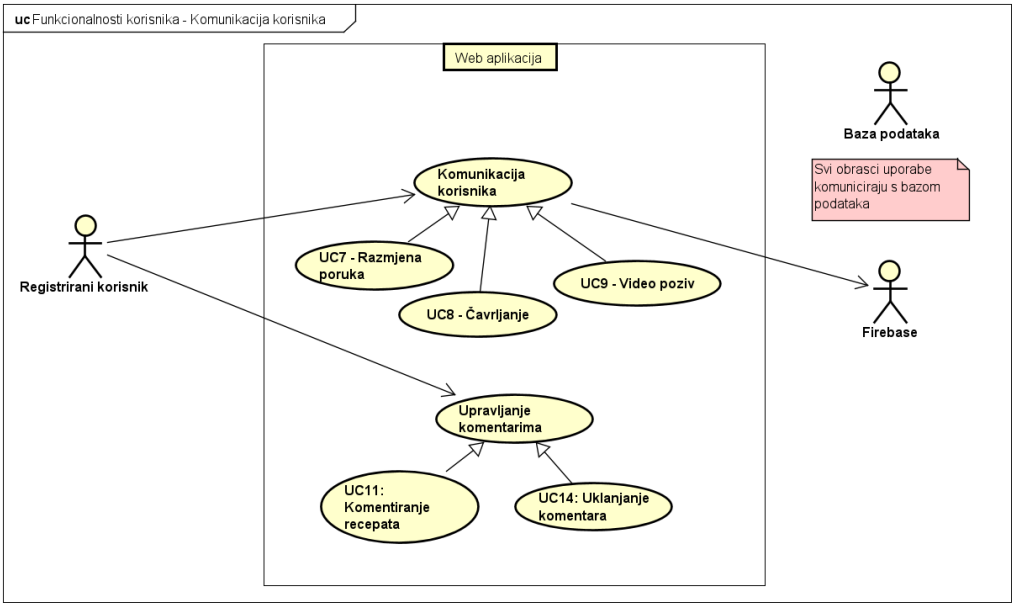
UC23 - Brisanje recepata od strane sistemskih administratora

- **Glavni sudionik: Sistemski administrator**
- **Cilj: Brisanje recepta nekog korisnika**
- **Sudionici: Baza podataka**
- **Preduvjet: Postoji objavljeni recept**
- **Opis osnovnog tijeka:**
  1. Sistemski administrator pregledava recept na kojem postoji gumb za brisanje recepta
  2. Ako administrator odabere navedeni gumb sustav će poslati zahtjev koji će ukloniti recept iz baze podataka

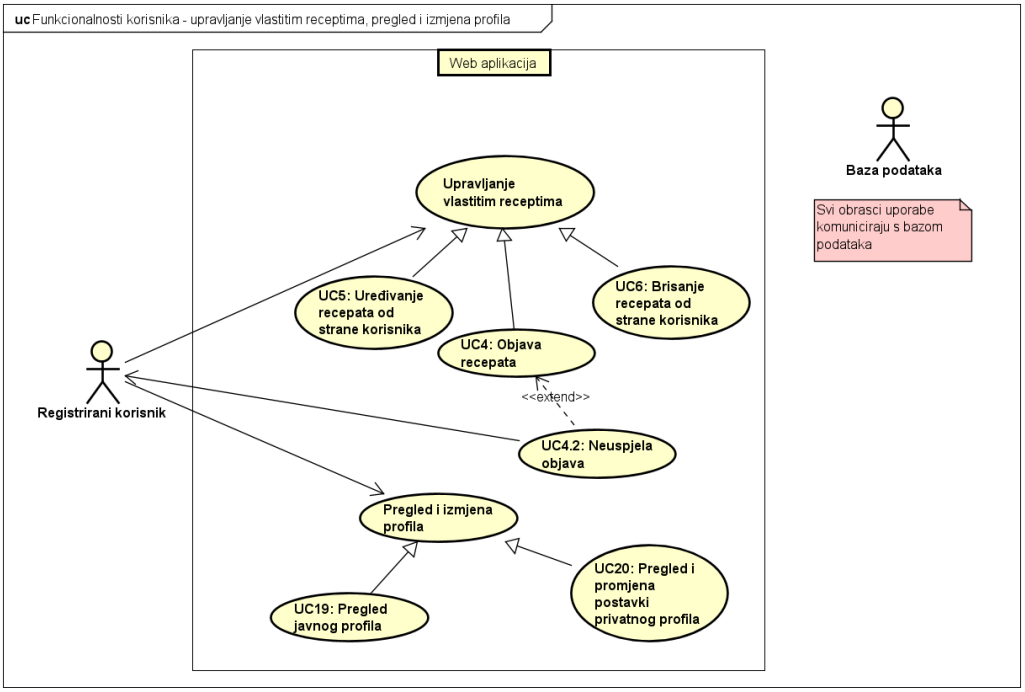
Dijagrami obrazaca uporabe



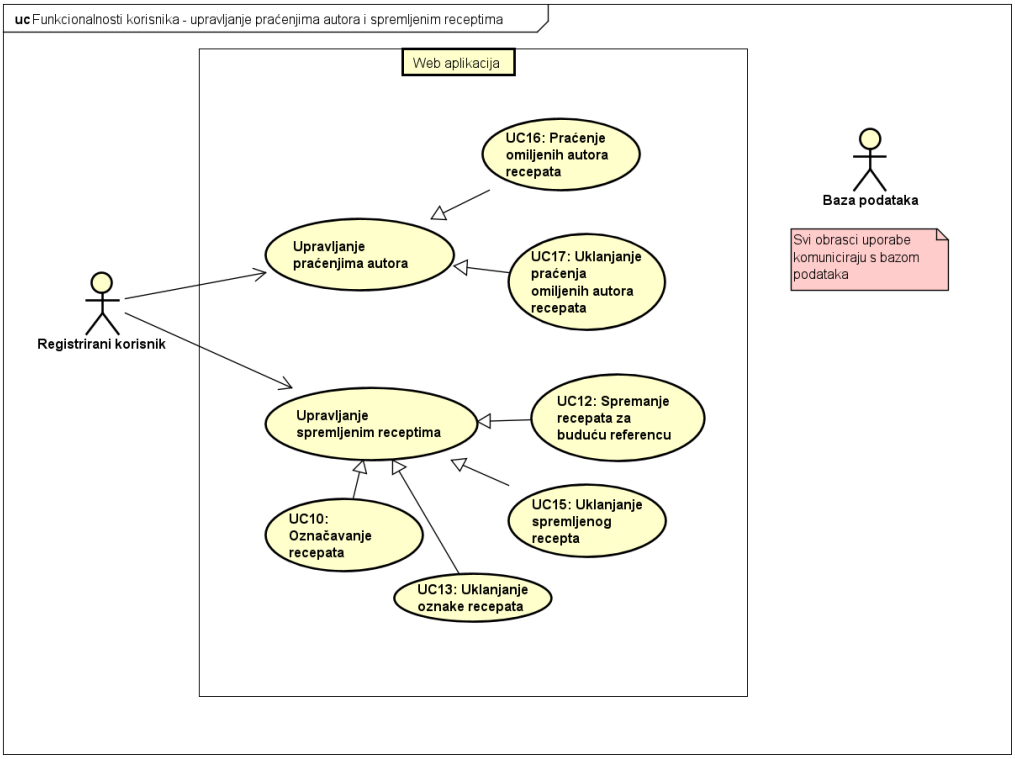
Slika 3.1: Dijagram obrazaca uporabe, funkcionalnosti korisnika



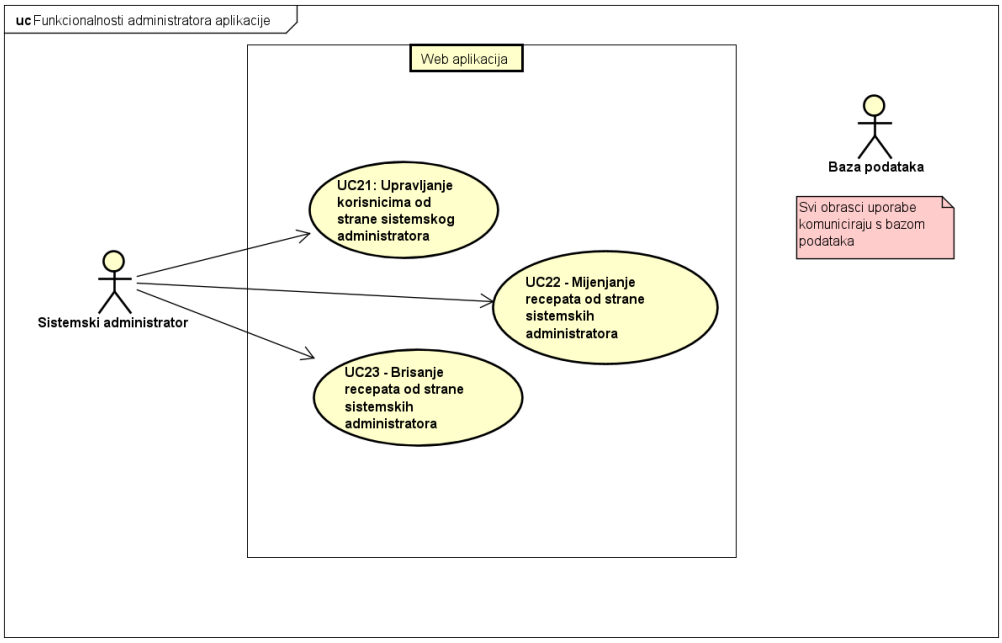
Slika 3.2: Dijagram obrazaca uporabe, komunikacija korisnika



Slika 3.3: Dijagram obrazaca uporabe, upravljanje receptima, pregled i izmjena profila



Slika 3.4: Dijagram obrazaca uporabe, praćenje autora i spremanje recepata

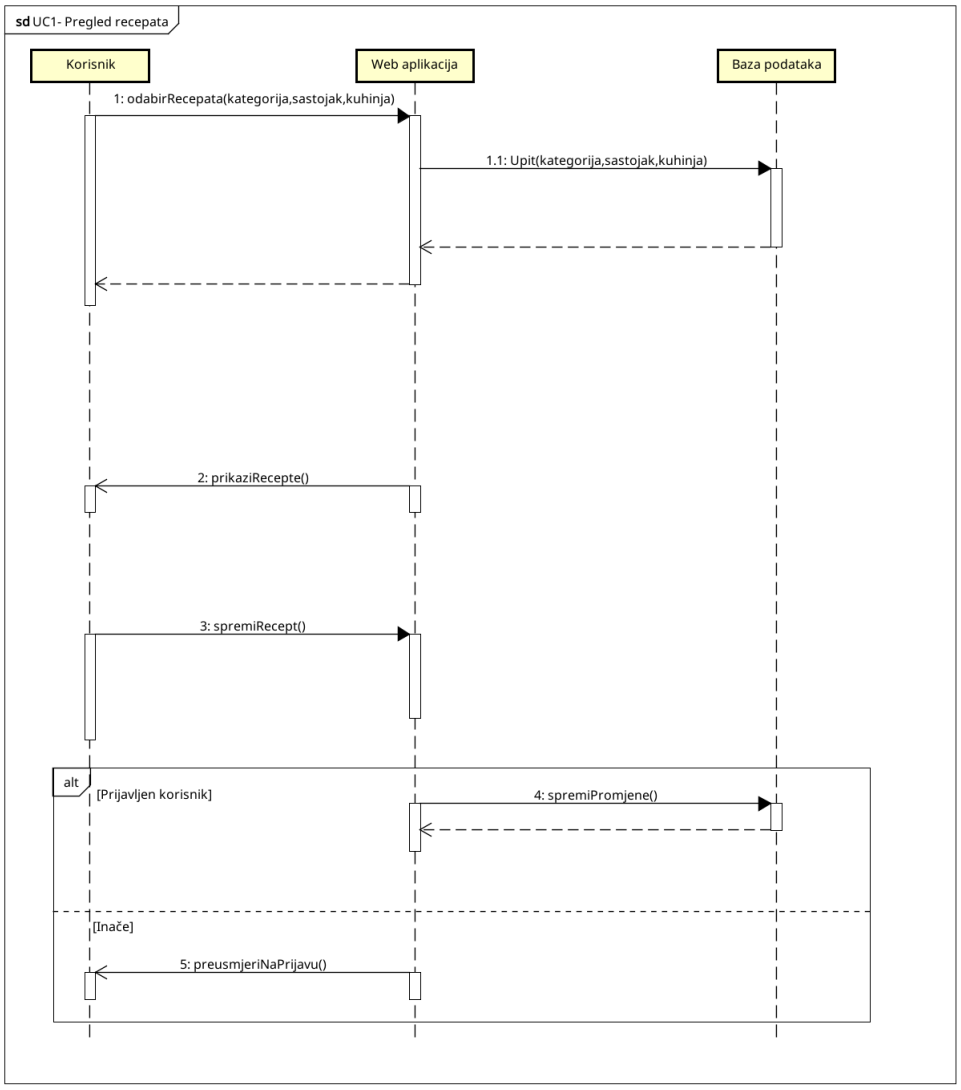


Slika 3.5: Dijagram obrazaca uporabe, funkcionalnosti administratora

3.1.2 Sekvencijski dijagrami

Obrazac uporabe UC1-Pregled recepata

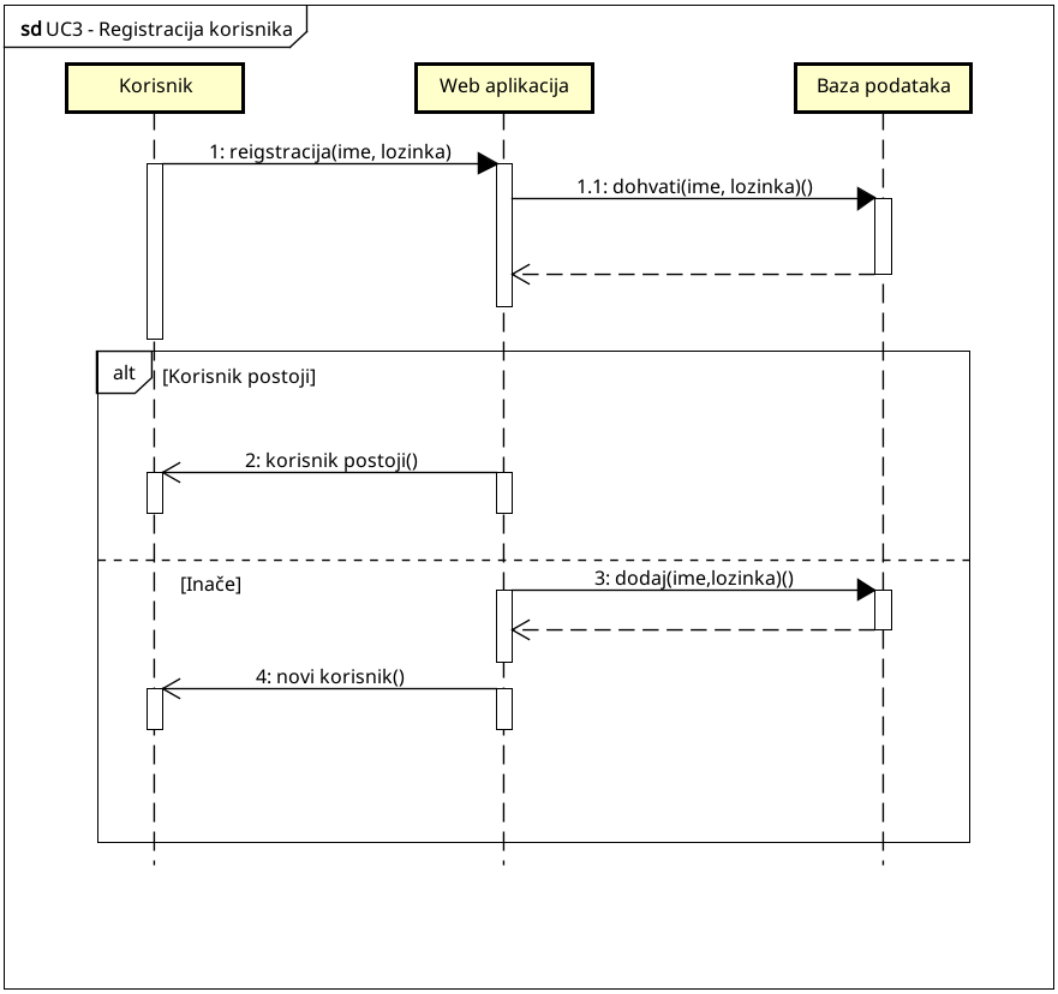
Korisnik šalje zahtjev za prikaz recepata po kategorijama, sastojcima i/ili kuhinjama kojim pripadaju. Poslužitelj dohvaća recepete koji zadovoljavaju uvjete i prikazuje ih korisniku. Korisnik sada može spremiti recepte, ako je prijavljen to se provodi, ako nije preusmjeri ga se na stranicu za prijavu.



Slika 3.6: Sekvencijski dijagram za UC1

Obrazac uporabe UC3-Registracija

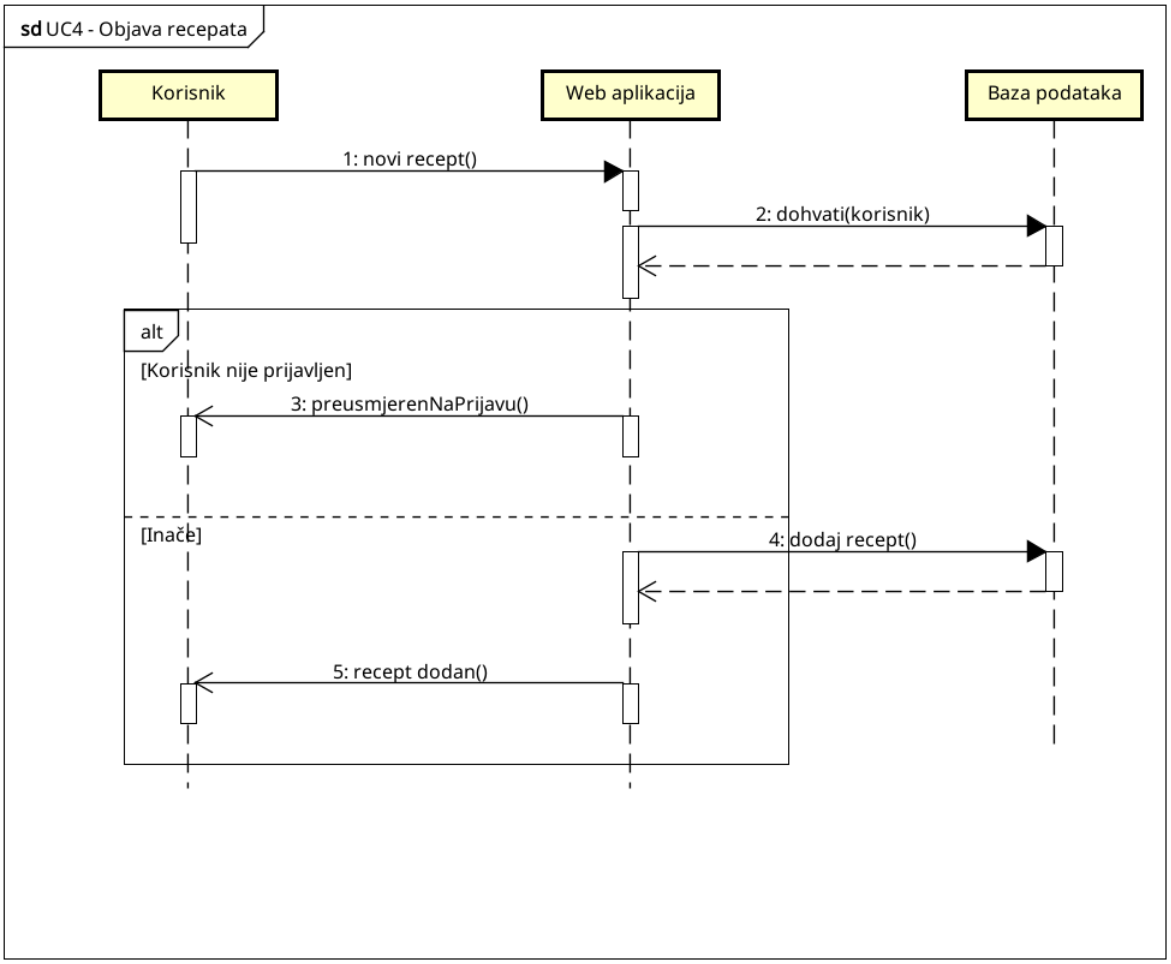
Korisnik se registrira s korisničkim imenom i lozinkom. Ako takav korisnik već postoji, korisniku se ispisuje greška. Inače, korisnik se uspješno registrirao i to se bilježi u bazi podataka.



Slika 3.7: Sekvencijski dijagram za UC3

Obrazac uporabe UC4-Prijava Korisnika

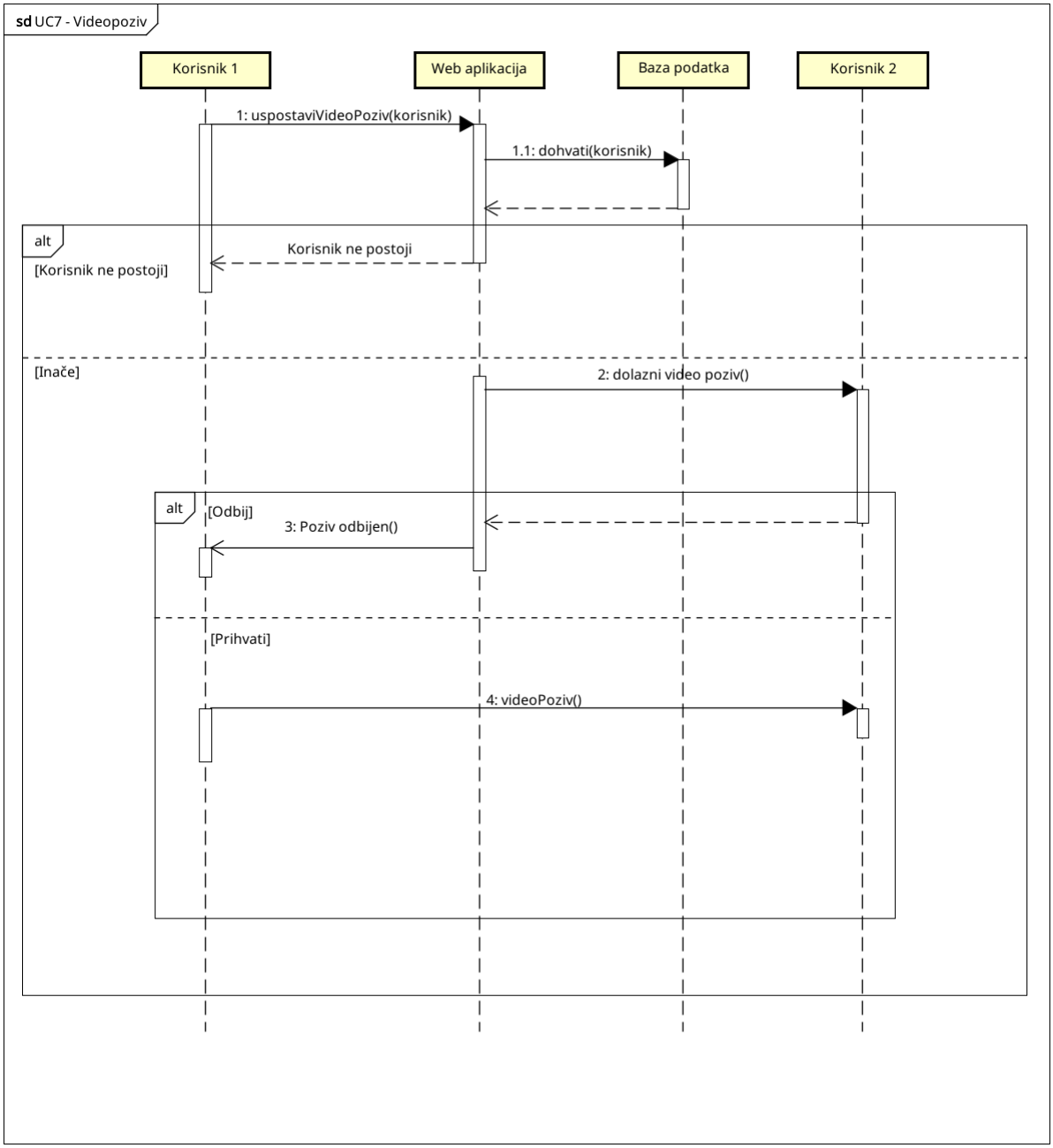
Korisnik pokuša objaviti novi recept. Ako nije prijavljen, preusmjeri ga se na prijavu. Inače, recept se dodaje u bazu podataka i o tome se obavijesti korisnik.



Slika 3.8: Sekvencijski dijagram za UC4

Obrazac uporabe UC8-Videopoziv

Korisnik zatraži video poziv s drugim korisnikom. Ako drugi korisnik ne postoji, poziv se ne uspostavlja i o tome se obavijesti prvog korisnika. Inače se šalje zahtjev za video pozivom drugom korisniku, koji ga može odbiti ili prihvatiti. Ako odbije poziv se ne uspostavlja i o tome se obavijesti prvi korsnik. Ako prihvati uspostavi se video poziv između dva korisnika.



Slika 3.9: Sekvencijski dijagram za UC7

3.2 Ostali zahtjevi

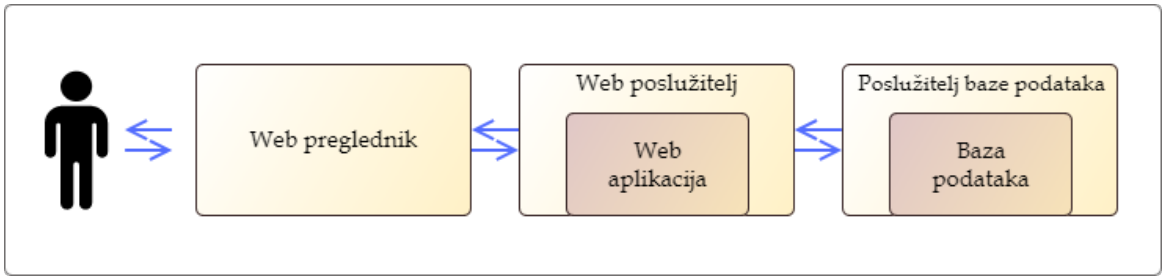
Nefunkcionalne zahtjeve koji će se navesti u nastavku teksta pojašnjavaju dodatne zahtjeve koje web aplikacija treba koristiti ili već koristi.

- Sustav treba biti implementiran kao web aplikacija koristeći objektno-orijentirane paradigme i norme kako bi se omogućilo ponovno korištenje dijelova koda/modula.
- Nepravilno korištenje web aplikacije ne smije rezultirati padom sustava, odnosno potrebno je usmjeriti korisnika na pravilno korištenje informacionim obavještenjima.
- U sustavu treba biti omogućen rad i korištenje web aplikacije od strane više korisnika (cca. 50).
- Sustav treba omogućiti komuniciranje korisnika s ostalim korisnicima putem tekstualnog chat-a i video poziva.
- Sustav mora podržati dijakritičke znakove hrvatskoj jezika, dakle treba podržati hrvatsku abecedu. Omogućit će se i promjena jezika s hrvatskog na engleski jezik.
- Hrvatski jezik je zadani jezik unutar web aplikacije.
- Sustav mora imati intuitivno sučelje koje neće stvarati nedoumice kod korisnika odnosno web-aplikacija treba biti jednostavna za korištenje.
- Konekcija s bazom podataka mora imati brz odziv. Bilo kakav pokušaj neovlaštenog pristupa informacijama u bazi podataka potrebno je spriječiti i istu adekvatno zašititi.
- Web aplikacija koristi proces kripitiranja lozinki za prijavu korisnika koji se tako u hash-ovima spremaju u bazu podataka.
- Sustav mora omogućiti korištenje određenih funkcionalnosti samo prijavljenim/registriranim korisnicima.
- Učitavanje početne stranice web aplikacije ne smije trajati duže od 5 sekundi.
- Pristup sustavu i razmjena podataka se vrši HTTPS protokolom.
- Pri razvoju web aplikacije koristi se React Native i Spring framework u Java programskom jeziku.

4. Arhitektura i dizajn sustava

Arhitektura sustava je bazirana na tri komponente koje komuniciraju jedna s drugom. Odnosno, sustav je podijeljen u tri dolje navedena sloja, a koji se mogu prikazati slikom ispod.

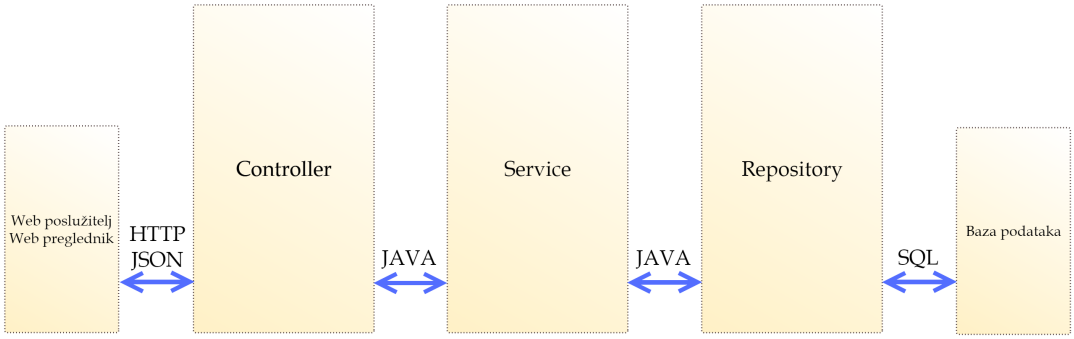
- *Web poslužitelj*
- *Web aplikacija*
- *Baza podataka*



Slika 4.1: Arhitektura sustava

Web preglednik omogućava korisniku prikaz web stranice koja pruža određene funkcionalnosti. Omogućava prikaz web stranice (prikaz videa, slika i ostalog multimedijalnog sadržaja) onako kako je definirana u datotekama, dakle omogućuje interpretiranje koda u koristan oblik ”običnom” korisniku. Putem web preglednika korisnik šalje zahtjev za željenu radnju koja se onda proslijedi idućim koponentama na obradu, te nakon obrade opet se prikazuju u vidljivom obliku kao vid povratne informacije.

Web poslužitelj je ključna komponenta u obradi korisničkih zahtjeva. Dakle, to je **središnji dio** aplikacije koji omogućava komunikaciju korisnika s aplikacijom. U središnjem sloju se odvijaju procesi koji su zaslužni za komunikaciju s bazom podataka ukoliko je to potrebno, a to se odvija putem kontrolera, servisa i pristupu bazi podataka. U ovom sloju za komunikaciju koristi se Java programski jezik. Odnos komponenti predstavljen je slikom ispod.

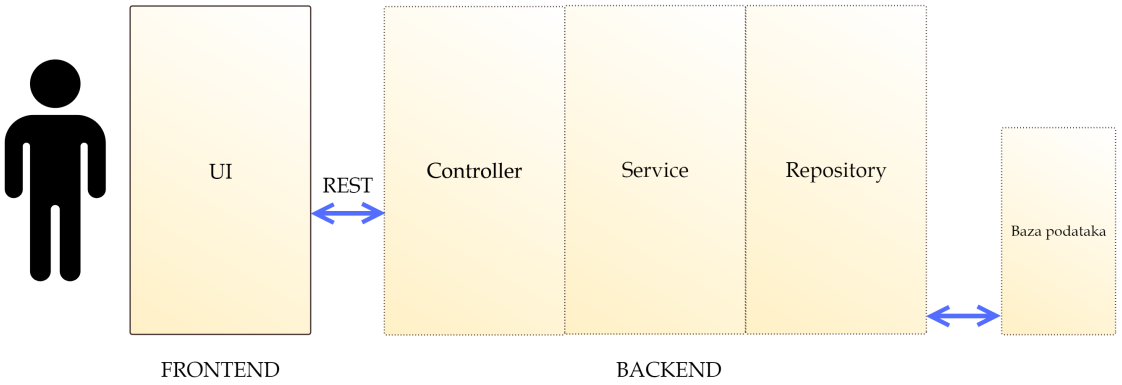


Slika 4.2: Arhitektura sustava backend

Za izradu ovakve web aplikacije kako bi se pokrile navedene specifične funkcionalnosti koristit će se Spring framework za Javu, te React za prikaz na web pregledniku u radnom okruženju IntelliJIDEA. Specifično za Spring framework, koristit će se navedeni tip arhitekture kao što je naveden slikom iznad, jer će se putem metoda JPARepository moći upravljati zahtjevima korisnika koji se vežu za upite u bazi podataka. Ukratko, obavljaju se operacije nad **bazom podataka H2**.

Kontroler upravlja korisničkim zahtjevima i prosljeđuje ih dalje prema **Service** gdje se obavlja logika nad upitima i zahtjevima. Service komunicira s **bazom podataka** preko JPARepository.

UI će biti povezan sa ostalim dijelovima web aplikacije putem **REST servisa**, odnosno REST servis komunicira s navedena tri sloja: Controller, Service i Repository. Takva vrsta komunikacije je predstavljena idućom slikom. UI je izveden uz pomoć **React** framework-a koji se koristi komponentama za organizaciju prikaza.



Slika 4.3: Arhitektura sustava backend i frontend

4.1 Baza podataka

Kao rješenje za web aplikaciju koju izrađujemo, odlučili smo se na relacijski model baze podataka jer nam ona omogućava precizno oblikovanje i modeliranje elemenata iz stvarnog svijeta. Međusobni odnos tablica se zasniva na relaciji dok se svaka tablica sastoji od naziva entiteta te njegovih pripadajućih atributa koji opisuju dani entitet. Ovakav tip baze podataka u ovom slučaju nam omogućava brzo spremanje, dohvat i izmjenu (uređivanje ili brisanje) podataka koji cirkulišu u web aplikaciji. Baza podataka koja je spremna za izradu ove web aplikacije se sastoji od sljedećih elemenata/tablica:

- Korisnik
- Recept
- Video
- Objava
- Kategorija
- ReceptKategorije
- ReceptSastojci
- Sastojak
- Komentar
- OmiljeniAutor
- OznačenRecept
- SpremljenRecept
- VrstaKuhinja
- Obavijesti

4.1.1 Opis tablica

Korisnik Entitet Korisnik služi za evidenciju podataka o korisnicima koji koriste web aplikaciju. Atributi koji ga opisuju su korisnickoIme, lozinkaKorisnik, imeKorisnik, prezimeKorisnik, brojTelefona, emailKorisnik, razinaOvlasti i Dostupan što označava kada je dostupan za komunikaciju s drugim korisnicima unutar web aplikacije. Entitet Korisnik je u *One-To-Many* vezi s entitetom Recept. Entitet Korisnik je i u *Many-To-Many* vezi s entitetom OmiljeniAutor koji bilježi korisnike koje ovaj korisnik prati. Također, ovaj entitet je u *Many-To-Many* vezi s entitetima SpremljenRecept, OznačenRecept, a u *One-To-Many* vezi s entitetima Objava i Komentar.

Korisnik		
IDKorisnik	INT	jedinstveni identifikator korisnika
KorisnickoIme	VARCHAR	korisničko ime
LozinkaKorisnik	VARCHAR	spremljena hash lozinka za pristup
ImeKorisnik	VARCHAR	ime korisnika
PrezimeKorisnik	VARCHAR	prezime korisnika
BrojTelefona	VARCHAR	broj telefona korisnika
EmailKorisnik	VARCHAR	e-mail adresa korisnika
RazinaOvlasti	VARCHAR	razina ovlasti u aplikaciji
Dostupan	TIME	vrijeme dostupnosti

Komentar Entitet Komentar služi za evidenciju komentara koji korisnici međusobno dijele u web aplikaciji. Opisuju ga atributi IDKomentar, IDKorisnik, IDObjava, te NaslovKomentar, SadrzajKomentar i DatumKomentar. Ovaj entitet je u *Many-To-One* vezi s entitetom Korisnik, te u istoj vezi s entitetom Objava.

Komentar		
IDKomentar	INT	jedinstveni identifikator komentara
IDKorisnik	INT	jedinstveni identifikator korisnika
IDObjava	INT	jedinstveni identifikator objave
NaslovKomentar	VARCHAR	naziv komentara
OpisKomentar	VARCHAR	sadržaj komentara
DatumKomentar	DATE	datum komentara

OmiljeniAutor Entitet OmiljeniAutor služi za evidenciju korisnika koje jedan korisnik zaprati. Opisan je sljedećim atributima: IDKorisnik i IDAutor. U *Many-To-Many* je vezi s entitetom Korisnik unutar web aplikacije.

OmiljeniAutor		
IDKorisnik	INT	jedinstveni identifikator pratitelja
IDAutor	INT	ID korisnika koji je zapraćen

Recept Entitet Recept služi za evidenciju podataka o receptima koji cirkulišu i nastaju u web aplikaciji. Opisan je atributima: IDRecept, NazivRecept, IDKategorija, IDSastojak, IDVrstaKuhinja, PripremaRecept, VrijemeKuhanja, IDOznaka, SlikaRecept i IDVideoRecept. Entitet Recept je u *Many-To-One* vezi s entitetom Korisnik, a u *Many-To-Many* vezi s entitetom Sastojak, te u *Many-To-Many* vezi s Kategorija, a *One-To-One* s VrstaKuhinja. Također je u *One-To-One* vezi s entitetom Objava.

Recept		
IDRecept	INT	jedinstveni identifikator recepta
NazivRecept	VARCHAR	naziv recepta
IDKategorija	INT	ID kategorije kojoj recept pripada
IDSastojak	INT	ID sastojka u receptu
IDVrstaKuhinja	INT	ID vrste kuhinje recepta
PripremaRecept	VARCHAR	Opis pripreme recepta
VrijemeKuhanja	TIME	Vrijeme potrebno za pripremu recepta
IDOznaka	INT	ID oznake recepta
SlikaRecept	VARCHAR	Slika recepta
IDVideoRecept	INT	ID Video recepta

Video Entitet Video služi za evidenciju videa koji se objavljuju uz recepte. Opisan je sljedećim atributima: IDVideo, NazivVideo i TrajanjeVideo što omogućuje pretragu recepata po dužini pripremanja. U *One-To-One* je vezi s entitetom Recept unutar web aplikacije.

Video		
IDVideo	INT	jedinstveni identifikator videa
NazivVideo	VARCHAR	naziv videa
TrajanjeVideo	TIME	trajanje videa

Objava Entitet Objava služi za evidenciju objava recepata koje korisnici objavljuju unutar web aplikacije. Opisan je atributima: IDObjava, IDKorisnik, IDRecept i DatumObjava. U *Many-To-One* vezi je s entitetom Korisnik, te u *Many-To-Many* vezi s entitetom Komentar i u *One-To-One* s entitetom Recept.

Objava		
IDObjava	INT	jedinstveni identifikator objave

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Objava		
IDKorisnik	INT	jedinstveni identifikator korisnika
IDRecept	INT	jedinstveni identifikator recepta
DatumObjava	DATE	datum objave

OznačenRecept Entitet OznačenRecept služi za evidenciju označenih recepata od strane korisnika. Opisan je atributima: IDRecept i IDKorisnik. U *Many-To-Many* vezi je s entitetom Korisnik.

OznačenRecept		
IDRecept	INT	jedinstveni identifikator recepta
IDKorisnik	INT	ID korisnika koji je označio recept

SpremljenRecept Entitet SpremljenRecept za evidenciju spremljenih recepata od strane korisnika. Opisan je atributima: IDRecept i IDKorisnik. U *Many-To-Many* vezi je s entitetom Korisnik.

SpremljenRecept		
IDRecept	INT	jedinstveni identifikator recepta
IDKorisnik	INT	ID korisnika koji je spremio recept

Kategorija Entitet Kategorija služi za evidenciju kategorija. Opisan je atributima: IDKategorija, NazivKategorija i OpisKategorija.

Kategorija		
IDKategorija	INT	jedinstveni identifikator kateogrije
NazivKategorija	VARCHAR	naziv kategorije
OpisKategorija	VARCHAR	opis kategorije

ReceptKategorije Entitet ReceptKategorije služi za evidenciju kategorija i recepata pod tim kategorijama. Opisan je atributima: IDKategorija, NazivKategorija i OpisKategorija. U *Many-To-Many* vezi je s entitetom Recept.

ReceptKategorije		
IDRecept	INT	jedinstveni identifikator kateogrije
IDKategorija	INT	jedinstveni identifikator recepta

VrstaKuhinja Entitet VrstaKuhinja služi za evidenciju različitih vrsta kuhinja. Opisan je atributima: IDVrstaKuhinja, NazivVrstaKuhinja i OpisKategorija. U *One-To-One* vezi je s entitetom Recept.

VrstaKuhinja		
IDVrstaKuhinje	INT	jedinstveni identifikator vrste kuhinje
NazivVrstaKuhinja	VARCHAR	naziv vrste kuhinje

Sastojak Entitet Sastojak služi za evidenciju različitih sastojaka. Opisan je atributima: IDSastojak, NazivSastojak.

Sastojak		
IDSastojak	INT	jedinstveni identifikator sastojka
NazivSastojak	VARCHAR	naziv sastojka

ReceptSastojci Entitet ReceptSastojci služi za evidenciju različitih sastojaka unutar recepta. Opisan je atributima: IDRecept, IDSastojak i KolicinaSastojak. U *Many-To-Many* vezi je s entitetom Recept.

ReceptSastojci		
IDRecept	INT	jedinstveni identifikator recepta

Nastavljeno na idućoj stranici

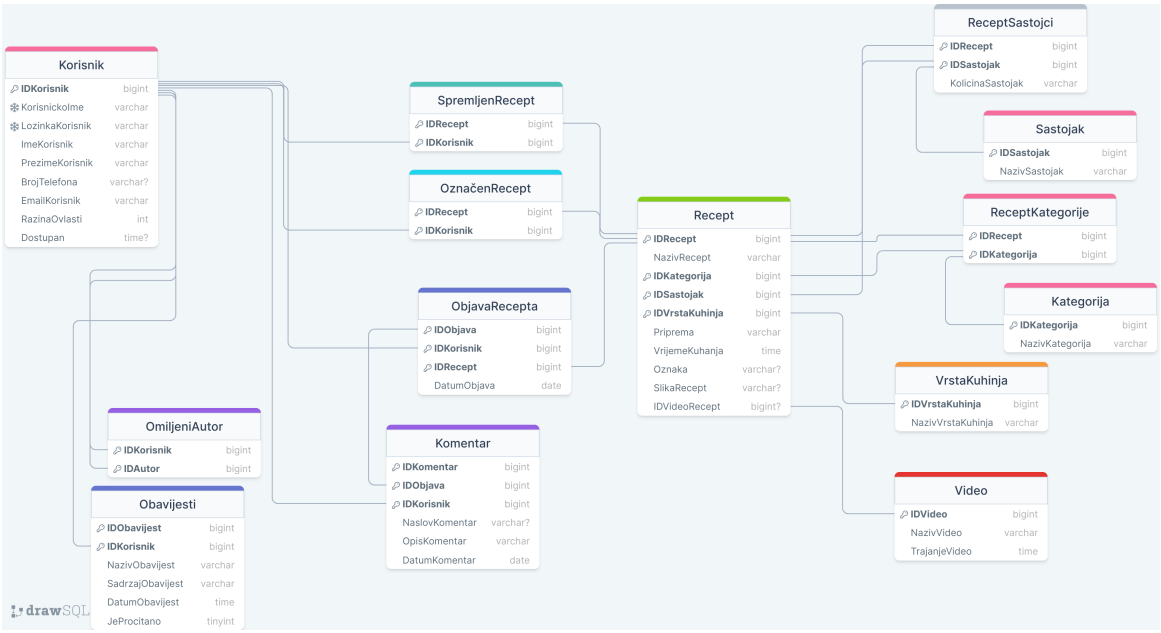
Nastavljeno od prethodne stranice

ReceptSastojci		
IDSastojak	INT	jedinstveni identifikator sastojka
KolicinaSastojak	VARCHAR	količina sastojka

Obavijesti Entitet Obavijesti služi za slanje obavijesti korisnicima koji prate određene autore recepata. Opisan je atributima: IDObavijest, IDAutor, NazivObavijest, SadrzajObavijest, DatumObavijest i JeProcitano. U *Many-To-One* vezi je s entitetom Korisnik.

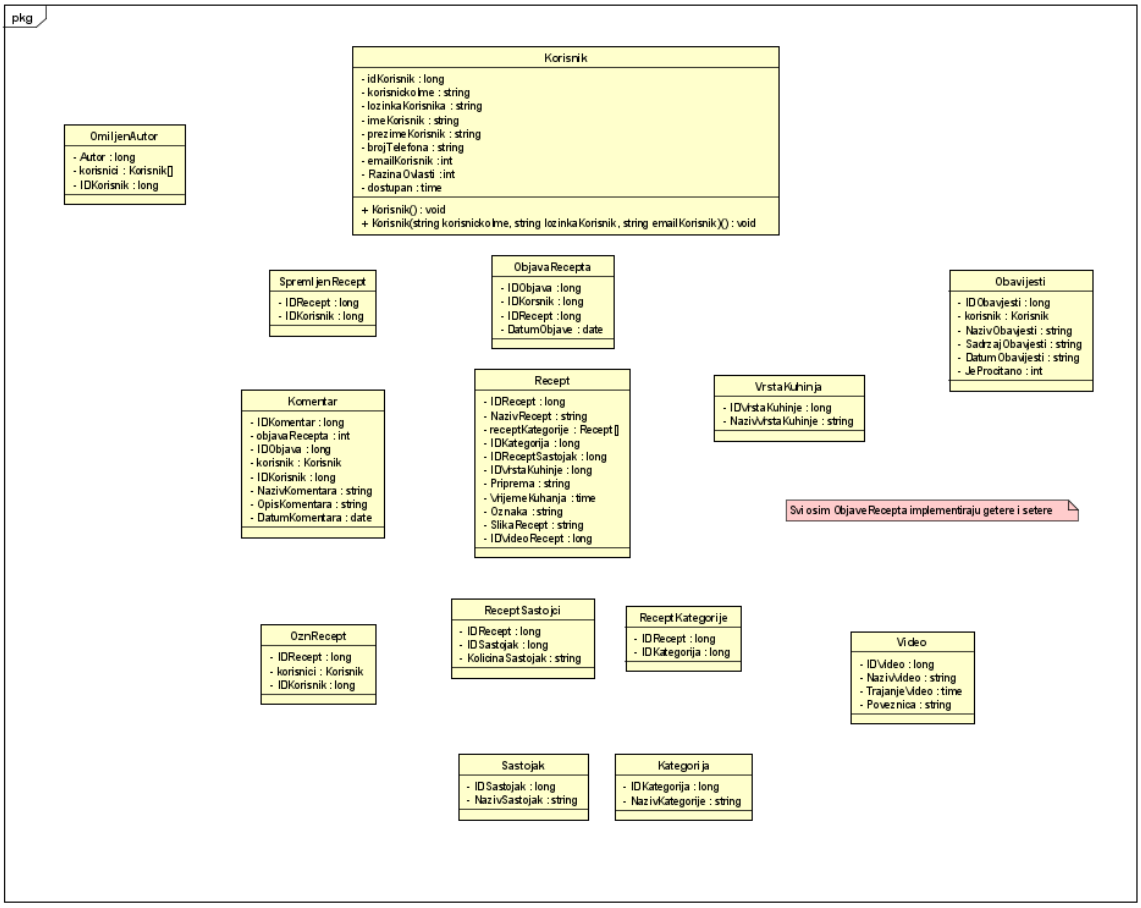
Obavijesti		
IDObavijest	INT	jed. identifikator obavijesti
IDKorisnik	INT	jedinstveni identifikator korisnika
NazivObavijest	VARCHAR	naziv obavijesti
SadrzajObavijest	VARCHAR	sadržaj obavijesti
DatumObavijest	DATE	datum obavijesti
JeProcitano	TINYINT	oznaka pročitane obavijesti

4.1.2 Dijagram baze podataka

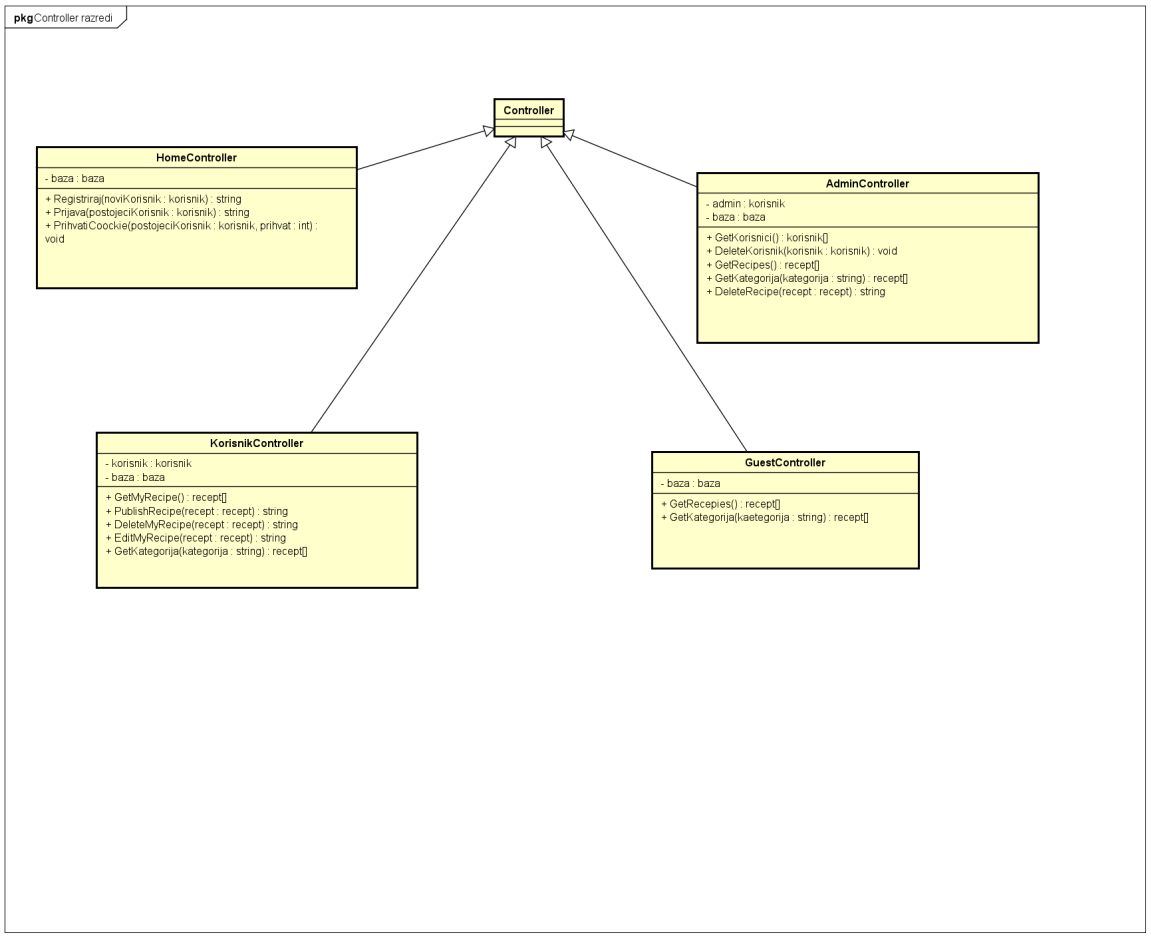


Slika 4.4: Dijagram relacijske baze podataka za web aplikaciju

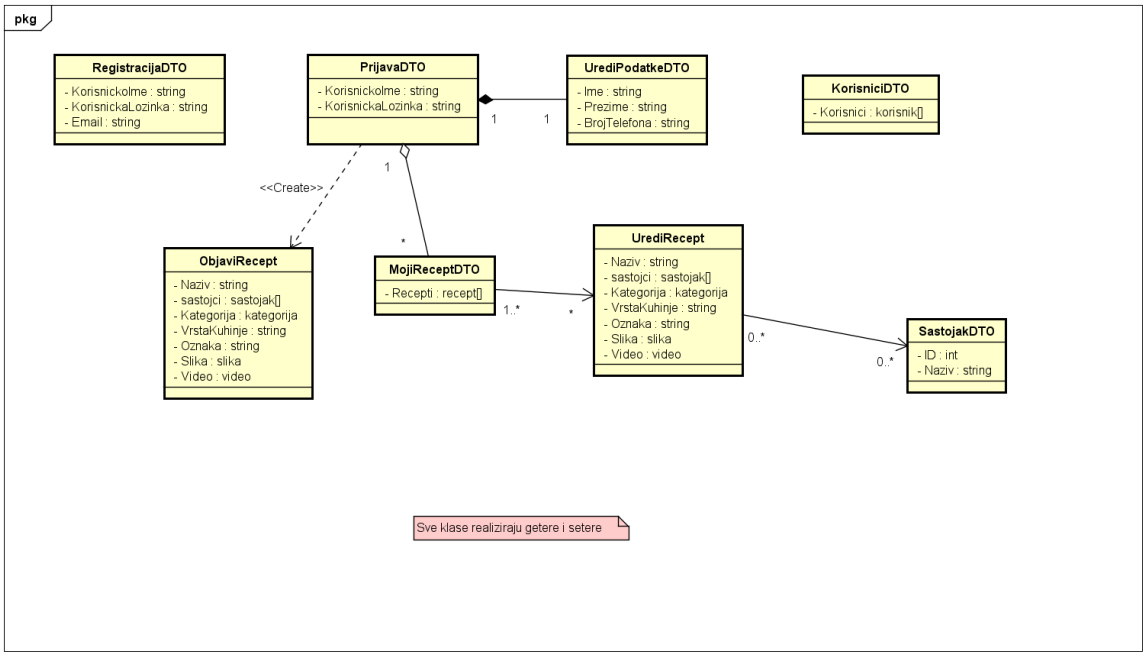
4.2 Dijagram razreda



Slika 4.5: Dijagram razreda



Slika 4.6: Dijagram razreda kontrolera



Slika 4.7: DTO

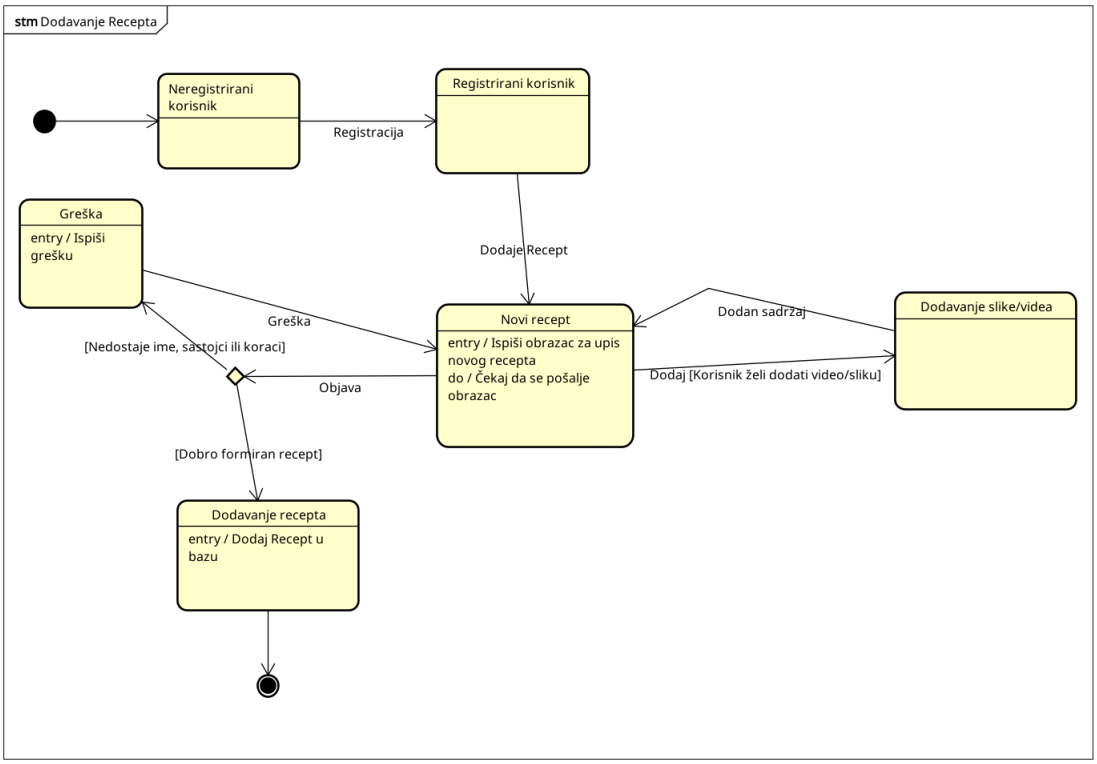
dio 2. revizije

Prilikom druge predaje projekta dijagram razreda i opisi moraju odgovarati stvarnom stanju implementacije

4.3 Dijagram stanja

Dijagram stanja - Dodavanje recepta

Registrirani korisnik šalje zahtjev za dodavanje recepta. Recept mora sadržavati ime, sastojke i korake pripreme. Dodatno, moguće je dodati video ili slike pripreme. Recept se sprema u bazu podataka.

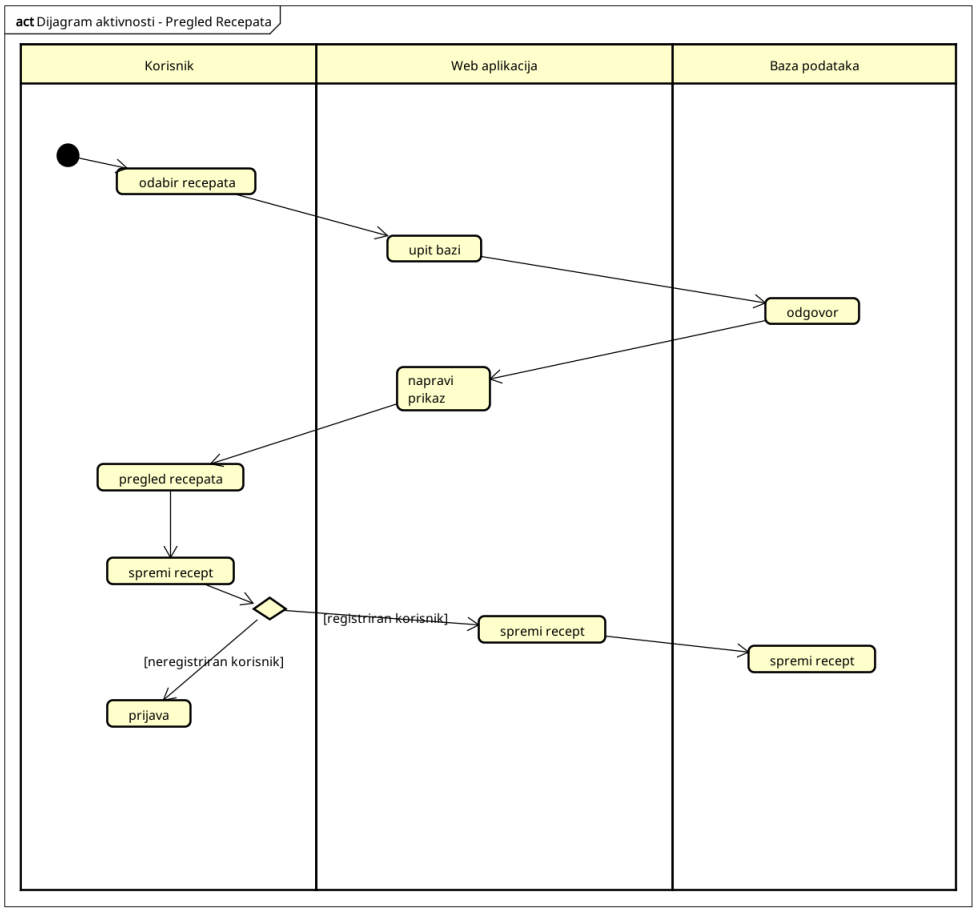


Slika 4.8: Dijagram stanja

4.4 Dijagram aktivnosti

Dijagram aktivnosti - Pregled recepta

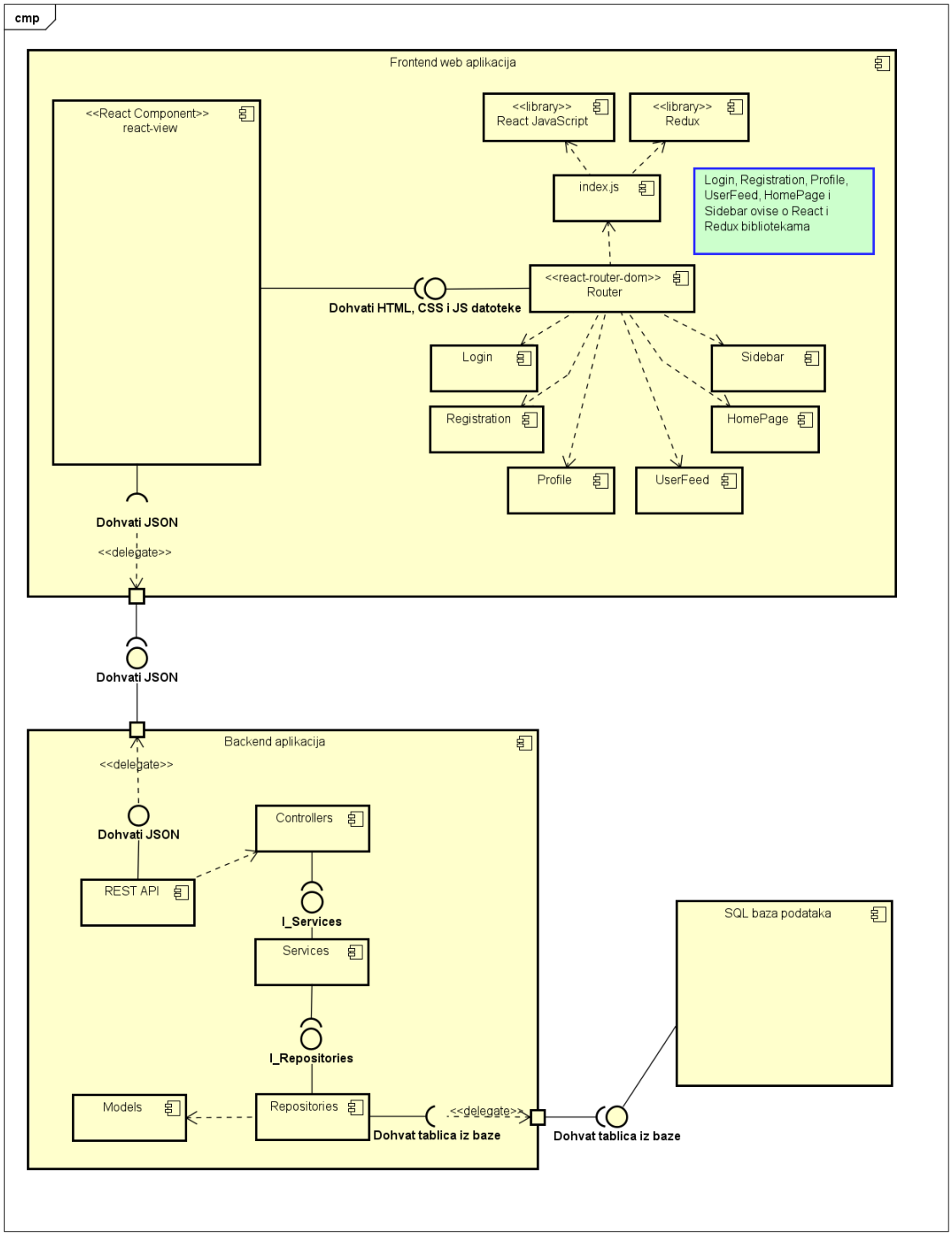
Korisnik šalje zahtjev za receptima po nekom kriteriju. Aplikacija šalje upit bazi i na temelju odgovora prikazuje recepte korisniku. Tada, korinsik može spremiti recept ako je registriran. Ako nije, preusmjeri ga se na registraciju.



Slika 4.9: Dijagram aktivnosti

4.5 Dijagram komponenti

Sustav se sastoji od *frontend* i *backend* dijelova, te baze podataka. *Frontend* dio sastoji se od niza JSX datoteka koje određuju prikaz stranice (HomePage, Login) ili dijela stranice (Sidebar). React-view komponenta komunicira sa komponentom Router i *backendom* te ovisno o korisnikovim akcijama osvježava prikaz i dohvaća nove podatke ili datoteke. Komponenta Router na upit sa URL-om odlučuje koja datoteka će se poslužiti na sučelje. JSX datoteke ovise o React i Redux bibliotekama. *Frontend* uspostavlja komunikaciju sa backendom koristeći REST API. Zahtjev se proslijeđuje Controllers komponenti koja koristi metode izložene u sučelju Services komponente. Komponenta Services ostvaruje komunikaciju sa komponentom Repositories koja je zadužena za dohvat tablica iz baze podataka.



Slika 4.10: Dijagram komponenti

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

Kako je prijašnje navedeno, za izradu aplikacije koristimo REST servis gdje za backend koristimo Java Spring Boot proširenje, za frontend koristimo React uz dodatak redux-a i jsx datoteka, a za bazu podataka koristimo PostgreSQL bazu podataka.

Za pregled podataka na PostgreSQL bazi podataka koristimo PgAdmin

REST servis je vrsta web servisa koja prati REST arhitekturu. REST je stil arhitekture gdje su frontend i backend odjeljeni kao dvije zasebne aplikacije te se odvija komunikacija između frontend-a i backenda uz pomoć REST servisa pozivanjem zahtjeva pristupa backendu.

Više o REST u Spring Boot-u: REST

Nadalje, na strani backend-a koristimo se MVC frameworkom što znači da naš backend je sastavljen od modela, pogleda i kontrolera te je kod odjeljen na modele, repozitorije, servise i kontrolere. Pomoću modela stvaramo tablice baze podataka i njihove entitete. Repozitoriji su grana koja je na backendu najbliža bazi podataka te se putem repozitorija zbiva komunikacija između backend-a i baze. Kontroleri su mjesto u kojemu se nalazi sam REST-api. Repozitorij je mjesto gdje zadajemo upite koje dalje koristimo u kontrolerima.

Više o Java Spring Boot na: Spring Boot, MVC u Spring Boot-u

Za razvoj frontend aplikacije, korišten je React kao biblioteka javascripta te kao pomoć koristi se Redux kojem je glavna svrha pojednostavljivanje upravljanja stanjem aplikacije, posebno kada je aplikacija složena s mnogo komponenti koje dijele podatke. Koriste se i jsx datoteke koje nam omogućavaju strukturiraniji kod te dinamičnost korištenja Reacta i javascripta.

Više o React-u na: React, Redux, JSX

Sva dokumentacija pisana je u latexu koji nam omogućava izradu pdf dokumenata dodavanjem oznaka i figura koje nam omogućuju uređivanje teksta i dodavanje vanjskih resursa kao što su slike, grafovi, tablice i slično.

Više o Latex-u: Latex

5.2 Ispitivanje programskog rješenja

dio 2. revizije

U ovom poglavlju je potrebno opisati provedbu ispitivanja implementiranih funkcionalnosti na razini komponenti i na razini cijelog sustava s prikazom odabranih ispitnih slučajeva. Studenti trebaju ispitati temeljnu funkcionalnost i rubne uvjete.

5.2.1 Ispitivanje komponenti

Potrebno je provesti ispitivanje jedinica (engl. unit testing) nad razredima koji implementiraju temeljne funkcionalnosti. Razraditi ***minimalno 6 ispitnih slučajeva*** *u kojima će se ispitati redovni slučajevi, rubni uvjeti te izazivanje pogreške (engl. exception throwing). Poželjno je stvoriti i ispitni slučaj koji koristi funkcionalnosti koje nisu implementirane. Potrebno je priložiti izvorni kôd svih ispitnih slučajeva te prikaz rezultata izvođenja ispita u razvojnom okruženju (prolaz/pad ispita).*

5.2.2 Ispitivanje sustava

Potrebno je provesti i opisati ispitivanje sustava koristeći radni okvir Selenium[1]*. Razraditi* ***minimalno 4 ispitna slučaja*** *u kojima će se ispitati redovni slučajevi, rubni uvjeti te poziv funkcionalnosti koja nije implementirana/izaziva pogrešku kako bi se vidjelo na koji način sustav reagira kada nešto nije u potpunosti ostvareno. Ispitni slučaj se treba sastojati od ulaza (npr. korisničko ime i lozinka), očekivanog izlaza ili rezultata, koraka ispitivanja i dobivenog izlaza ili rezultata.*

Izradu ispitnih slučajeva pomoću radnog okvira Selenium moguće je provesti pomoću jednog od sljedeća dva alata:

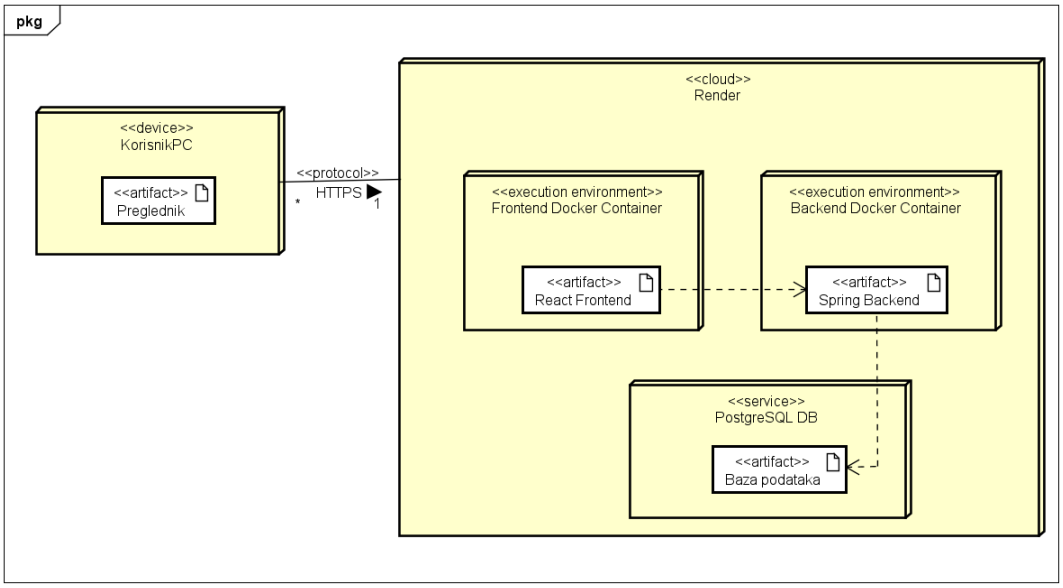
- *dodatak za preglednik* ***Selenium IDE*** *- snimanje korisnikovih akcija radi automatskog ponavljanja ispita*
- ***Selenium WebDriver*** *- podrška za pisanje ispita u jezicima Java, C#, PHP koristeći posebno programsko sučelje.*

Detalji o korištenju alata Selenium bit će prikazani na posebnom predavanju tijekom semestra.

[1] `https://www.seleniumhq.org/`

5.3 Dijagram razmještaja

Klijenti koriste web preglednik kako bi pristupili web aplikaciji. Aplikacija je puštena u pogon na Render cloud servisu. Sustav se sastoji od *frontend* dijela realiziranog uz pomoć React biblioteke, *backend* dijela implementiranog koristeći radni okvir Java Spring i PostgreSQL baze podataka. *Frontend* i *backend* dijelovi nalaze se u zasebnim Docker spremnicima. Ostvarena je komunikacija između *frontenda* i *backenda*, te *backenda* i baze podataka.



Slika 5.1: Dijagram razmještaja

5.4 Upute za puštanje u pogon

Za puštanje u pogon koristimo usluge web aplikacije Render koja nam omogućuje kreiranje PostgreSql baze podataka te upload Spring Boot backenda uporabom DockerFile-a te upload React aplikacije uporabom Node-a.

Za početak važno je izraditi korisnički račun na Renderu. Kada uđemo na početnu stranicu Rendera, pritisnemo tipku Get Started gdje nas tada preusmjerava na formu za registraciju ili prijavu. Nudi nam i prijave pomoću drugih usluga kao što su GitHub, GitLab i Google. Preporuka je da ako je aplikacija već na GitHub-u, da se prijavi putem github računa.

Nakon izrade korisničkog računa otvara se zaslon Render Dashboard. Sada za kreiranje PostgreSQL baze podataka slijedimo jednostavne upute:

New =>PostgreSQL =>Name: naziv vidljiv u Render Dashboard-u, Database: naziv naše baze što uzimamo nešto kratko i prepoznatljivo, User: najbolje staviti nešto generičko kao user, Region: podesiti jednu od ponuđenih regija koja je najbliža. =>Create Database

Sada kada kliknemo na generiranu bazu podataka, možemo vidjeti neke od informacija. Za spajanje na bazu podataka koristiti ćemo External Database URL.

Te informacije će nam biti potrebne kako bismo se spojili sa bazom podataka te mogli vidjeti i uređivati podatke u PgAdmin-u te za spajanje Spring Boot backenda na bazu.

Za spajanje na PgAdmin-u: Desni klik na Servers =>Register =>Server =>Name: dodamo ime =>Pritisak na karticu Connection =>Host name/address upišemo iz External Database URL podatka sve nakon znaka @ nakon čega slijedi dpg pa sve do zadnje kose crte.
Izgled prije: postgres://user:.....@dpg...../nazivbaze
Upisujemo: dpg.....
Port je po default-u 5432.
Maintnance database: Kopiramo polje Database.
Username: Kopiramo polje Username.
Password: Kopiramo polje Password.
Pritisnemo tipku save i trebalo bi prikazati bazu podataka na listi servera.

Za spajanje Spring Boot-a na bazu potrebno je otići u application.properties u folderu resources i upisivati podatke na sljedeći način:
spring.datasource.url = jdbc:postgresql://dpg.../nazivbaze
spring.datasource.username = Kopiramo polje Username
spring.datasource.password = Kopiramo polje Password
spring.jpa.hibernate.ddl-auto = odaberemo neku od opcija koje se mogu proučiti na ovoj stranici.
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect

Na dalje, opisati ćemo dodavanje Spring Boot aplikacije na Render. Kako bi to uopće bilo moguće, moramo dodati u Spring Boot DockerFile. Prilikom izrade Spring Boot-a, odabrali smo Gradle ili Maven. Ovdje smo odabrali maven te sada čemu unutar naše backend aplikacije napraviti folder docker, unutar njega folder maven te unutar njega smjestiti naš DockerFile. U DockerFile potrebno je voditi računa samo o verziji jave koju koristimo te na koji port želimo izložiti aplikaciju.

Kod:

```
# Container za izgradnju (build) aplikacije
FROM openjdk:17-alpine AS builder

# Kopiranje izvornog koda u container
COPY ../../.mvn .mvn
COPY ../../mvnw .
COPY ../../pom.xml .
COPY ../../src src
RUN chmod +x mvnw

# Pokretanje builda
RUN ./mvnw clean package

# Stvaranje containera u kojem ce se vrtiti aplikacija
FROM openjdk:17-alpine

## Ovdje je moguce instalirati alate potrebne za rad aplikacije. Vjerojatno vam nec
## Linux distro koji se koristi je Alpine, stoga se kao package manager koristi apk
```

```
#RUN apk install <nesto>

# Kopiranje izvrsnog JAR-a iz build containera u izvrsni container
COPY --from=builder target/*.jar /app.jar

# Izlaganje porta
EXPOSE 8080

# Naredba kojom se pokrece aplikacija
ENTRYPOINT ["java","-jar","/app.jar"]
```

Kada to spremimo, možemo otići u Render Dashboard i uploadati našu aplikaciju. New =>Build and deploy from a Git repository (sada je potrebno spojiti Github ili GitLab račun što je puno lakše ako smo račun izradili uz pomoć GitHub-a) =>Next =>Odaberemo naš račun i pritisnemo Connect =>Name: odaberemo ime, Region: odaberemo najbližu regiju, Branch: unutar našeg Git projekta odaberemo na kojem branchu se nalazi ta naša aplikacija. =>Sada pod Root Directory je važno staviti nešto ako imamo unutar našeg git projekta više različitih aplikacija kao što u našem slučaju na jednom git projektu se nalazi i frontend i backend aplikacija pa za Root Directory odabiremo IzvorniKod/NazivBackendAplikacije =>Runtime: Docker =>Advanced =>Dockerfile Path: ./docker/maven/DockerFile =>Create Web Service

Sada da bismo isto tako uploadali frontend na Render potrebno je u Reactu na app.js datoteci promijeniti neke stvari. Primjer koda us korištenje express-a je sljedeći:

```
const express = require("express");
const { createProxyMiddleware } = require("http-proxy-middleware");
require("dotenv").config();
const path = require("path")

const app = express();

// Configuration
```

```
const { API_BASE_URL } = process.env;

// Proxy
app.use(
"/api",
createProxyMiddleware({
target: API_BASE_URL,
changeOrigin: true,
})
);

app.listen(3000);

app.use(express.static(path.join(__dirname, 'build')))

app.get("/*", async (req, res) => {
res.sendFile(path.join(__dirname, 'build', 'index.html'))
}
);
```

Kada ovo spremimo možemo otići na Render Dashboard.

New Web Service =>Build and deploy from a Git repository =>Connect =>Connect na naš projekt =>Sada na što je potrebno obratiti pažnju jest mi sada Root Directory postavljamo na IzvorniKod/NazivDirektorijaFrontenda i Runtime postavljamo na Node =>Build Command: yarn build =>Start Command: yarn start-prod =>Sada još jedna važna stvar je da u enviroment varijable želimo postaviti dodatnu varijablu API_BASE _URL koju koristimo za spajanje frontenda na backend. Vrijednost te varijable postavimo na link tj. web adresu naše backend aplikacije. Još jedna Enviroment Varijabla koja može biti uzrokovati probleme ako se ne stavi jest NODE_VERSION gdje vrijednost postavimo na neku od zadnjih stablinih verzija. Aplikacija CookBooked je postavlje na verziju 18.8.0 =>Create Web Service

Ako je aplikacije uspješno postavljena, prikaže se oznaka Live što nam označi da je aplikacija online.

6. Zaključak i budući rad

Kroz trajanje cijelog ovog projekta, nailazili smo na brojne prepreke. Prva od njih je naravno bila usklađivanje korištenja gita sa svim članovima tima. Tu se događaju gubitci i korištenje reverta i slično, dok se nismo bolje upoznali sa opcijom merge. Merge kod izrade tehničke dokumentacije bih rekao da je bio izazovan zbog brojnih konflikata kod slika i pdf dokumenata i neiskustva kod bavljenja s istim.

Prelaskom na izradu aplikacije prvi problem na koji smo naišli bilo je uspostava backenda i spajanje Spring Boota na bazu podataka. Kada smo to riješili došlo je vrijeme za deploy aplikacije na web pomoću web aplikacije Render. Tu su krenuli brojni problemi najviše greška CORS koja nije dozvoljavala na frontendu fetchanje backenda koji nije u istoj domeni.

Taj problem rješili smo u dva dijela. Prvo, kada je proradilo, no i dalje je postojao određeni time lag te tada uz drugi puta uz pomoć korištenja axiosa i reduxa dobili smo efektivno i funkcionalno rješenje. No ta promjena uzrokovala je i promjenu sustava dohvaćanja podataka, što je iziskivalo upoznavanje sa našim novim sistemom dohvaćanja podataka sa backenda.

Kroz proces izrade cijele aplikacije, upoznali smo se s brojnim problemima sa kojima ćemo se susretati na kasnijim radovima, od git-a, do zahtjevnosti pisanja dokumentacije i neke od osnovnih stvari koje se često zanemaruju, a to je koliko je značajno i olakšava posao rad u timu. Kvalitetan rad u timu može uvelike ubrzati izradu aplikacije te olakšati teret što uvelike daje svakom članu time veću motivaciju za rad.

Naravno, stekli smo i znanja tehnologija React i Spring Boot, te kako izgleda upload aplikacije na web. Osim toga po prvi puta smo se susreli sa alatom za izradu pdf dokumenta, latex, što je sam po sebi zanimljiv i vrlo precizan alat.

Sve u svemu, smatramo kako smo stekli veliko iskustvo, koliko programsko riješenje

za web aplikaciju iziskuje vremena da se napravi u cjelosti te koliko različitih ljudi može stajati iza jedne uspješne aplikacije sa velikim brojem korisnika.

U aplikaciji nije implementiran video kao ni videopoziv između korisnika.

Popis literature

1. Programsko inženjerstvo, FER ZEMRIS, http://www.fer.hr/predmet/proinz
2. I. Sommerville, "Software engineering", 8th ed, Addison Wesley, 2007.
3. T.C.Lethbridge, R.Langaniere, "Object-Oriented Software Engineering", 2nd ed. McGraw-Hill, 2005.
4. I. Marsic, Software engineering book", Department of Electrical and Computer Engineering, Rutgers University, http://www.ece.rutgers.edu/~marsic/books/SE
5. The Unified Modeling Language, https://www.uml-diagrams.org/
6. Astah Community, http://astah.net/editions/uml-new

Indeks slika i dijagrama

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

Kontinuirano osvježavanje

U ovom dijelu potrebno je redovito osvježavati dnevnik sastajanja prema predlošku.

1. Uvodni sastanak
   - Datum: 19. listopada 2023.
   - Prisustvovali: A.Hohnjec, M.Pavić, A.Kantarević, L.Ivčević, B.Zrakić, F.Vrbić, A.Zglavnik
   - Teme sastanka:
     - Upoznavanje sudionika projektnog tima
2. Sastanak o raspodjeli poslova
   - Datum: 25. listopada 2023.
   - Prisustvovali: A.Hohnjec, M.Pavić, A.Kantarević, L.Ivčević, B.Zrakić, F.Vrbić, A.Zglavnik
   - Teme sastanka:
     - Dogovor o raspodjeli poslova pri izradi aplikacije
     - Dogovor o raspodjeli poslova pri izradi dokumentacije
     - Dogovor o načinu izvršavanja projekta
3. Merganje pojedino izrađene dokumentacije
   - Datum: 30. listopada 2023.
   - Prisustvovali: A.Hohnjec, M.Pavić, A.Kantarević, L.Ivčević, B.Zrakić, F.Vrbić, A.Zglavnik
   - Teme sastanka:
     - Merganje pojedino izrađene dokumentacije
4. Dogovor za daljnji napredak aplikacije
   - Datum: 15. prosinca 2023.
   - Prisustvovali: A.Hohnjec, M.Pavić, A.Kantarević, L.Ivčević, B.Zrakić, F.Vrbić, A.Zglavnik

- Teme sastanka:
  - Raspodjela daljnjih poslova
  - Pregled dosad napravljenoga
  - Dogovor za rad kroz praznike

5. Pregled napravljenog kroz praznike
   - Datum: 4. siječnja 2024.
   - Prisustvovali: A.Hohnjec, M.Pavić, A.Kantarević, L.Ivčević, B.Zrakić, F.Vrbić, A.Zglavnik
   - Teme sastanka:
     - Pregled napravljenog programskog dijela
     - Davanje dodatnih informacija za izradu dokumentacije
6. Dogovor oko završnih detalja
   - Datum: 15. siječnja 2024.
   - Prisustvovali: A.Hohnjec, M.Pavić, A.Kantarević, L.Ivčević, B.Zrakić, F.Vrbić, A.Zglavnik
   - Teme sastanka:
     - Završni detalji pri izradi aplikacije
     - Završavanje sa izradom dokumentacije

Tablica aktivnosti

Kontinuirano osvježavanje

Napomena: Doprinose u aktivnostima treba navesti u satima po članovima grupe po aktivnosti.

	Antonio Hohnjec	**Marko Pavić**	**Armis Kantarević**	**Luka Ivčević**	**Benjamin Zrakić**	**Filip Vrbić**	**Antonio Zglavnik**
Upravljanje projektom	10						
Opis projektnog zadatka	2						
Funkcionalni zahtjevi							
Opis pojedinih obrazaca				1	1	1	
Dijagram obrazaca					3	2	
Sekvencijski dijagrami				2			
Opis ostalih zahtjeva			1				
Arhitektura i dizajn sustava		1					
Baza podataka		4	3				
Dijagram razreda							2
Dijagram stanja							
Dijagram aktivnosti							
Dijagram komponenti							
Korištene tehnologije i alati							
Ispitivanje programskog rješenja							
Dijagram razmještaja							
Upute za puštanje u pogon							

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

	Antonio Hohnjec	**Marko Pavić**	**Armis Kantarević**	**Luka Ivčević**	**Benjamin Zrakić**	**Filip Vrbić**	**Antonio Zglavnik**
Dnevnik sastajanja	X						
Zaključak i budući rad							
Popis literature							
Dodatne stavke kako ste podijelili izradu aplikacije							
npr. izrada početne stranice		1			1		1
izrada baze podataka			1	1		1	
spajanje s bazom podataka	1						
deployment	1						

Dijagrami pregleda promjena

dio 2. revizije

Prenijeti dijagram pregleda promjena nad datotekama projekta. Potrebno je na kraju projekta generirane grafove s gitlaba prenijeti u ovo poglavlje dokumentacije. Dijagrami za vlastiti projekt se mogu preuzeti s gitlab.com stranice, u izborniku Repository, pritiskom na stavku Contributors.