

# Home

---

## Programsko inženjerstvo ak.god 2025./2026

---

Sveučilište u Zagrebu

---

Fakultet elektrotehnike i računarstva

---

### FlipMemo

---

Tim: <TG08.4>

Bitange i princeze

Članovi i pozicije:

Lana Vučen - voditelj, frontend, dizajn

Mihaela Novak - dokumentacija

Zvonimir Stracenski - frontend, dizajn

Jan Volenec - backend, baza podataka

Mislav Putarek - backend, baza podataka

Andrija Janušić - backend, baza podataka

Nikola Simunić - dokumentacija

Nastavnik: Vlado Sruk

# 1. Opis projektnog zadatka

## Cilj projekta

Ovaj projekt je rezultat timskog rada u sklopu projektnog zadatka kolegija [Programsko inženjerstvo](#) na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu.

Cilj ovog projekta je osmisliti i konstruirati web-aplikaciju (FlipMemo) koja omogućava učenje odabranog stranog jezika te razvoj vokabulara iz istog. Aplikacija bi istovremeno trebala pružiti jednostavan pristup učenju novog jezika, ali i ozbiljnu mogućnost usvajanja znanja zasnovan na tehnički učenja ponavljanja s odmakom (eng. spaced repetition).

## Analiza problema

Prema najnovijem izvješću EF English Proficiency Indexa (EF EPI) za 2024. godinu, globalna razina znanja engleskog jezika među odraslima kontinuirano opada. Ljudi često nisu sigurni odakle krenuti, a i opće poznata činjenica je da glad za znanjem nestaje s godinama. Naša aplikacija stoga nudi pristupačan način učenja, ponavljanja i usavršavanja engleskog jezika na interaktivan način.

Učenje stranih jezika u doba globalizacije predstavlja jedan od najaktualnijih problema za svakodnevnicu modernog čovjeka. Obrazovanje u stranom jeziku može se smatrati cjeloživotnim jer je na kraju krajeva i sam jezik evoluirajući te se konstantno razvija te problem nastaje u ograničenoj sposobnosti ljudskog pamćenja velikog broja riječi i izraza. Uz već brojne aplikacije koje se bave istim problemom njihova glavna mana je neoptimiziran proces ponavljanja prema individualnim rezultatima korisnika.

Glavni problemi koji se u praksi pojavljuju su:

- Nedostatak personalizacije: mnoge aplikacije koriste jednak tempo učenja za različite korisnike, bez prilagodbe koja ovisi o njihovom stvarnom znanju
- Pad motivacije korisnika: korisnicima nepersonaliziran pristup i monotono ponavljanje postaje zasitno te odustanu
- Ograničen fokus na vokabular: određene aplikacije stavljuju prevelik naglasak na gramatičku pismenost korisnika bez dovoljnog razvoja samog vokabulara korisnika
- Nedostatak analitičkih i statističkih povratnih informacija: korisnik nema uvid u to koje riječi najčešće zaboravlja, a koje je savladao te koliko im vremena treba za usvajanje određene lekcije.

## Korisnički zahtjevi

Aplikacija je usmjereni na studente i učenike stranih jezika, ali i sve ostale zainteresirane kao dodatna pomoć u usavršavanju jezičnih sposobnosti s fokusom na razvoj vokabulara. Korisnik u svrhu personalizacije ima mogućnost registriranju i stvaranja osobnog korisničkog računa.

Registrirani korisnici imaju pristup lekcijama stranog jezika te praćenju statistike ovisno o vlastitom uspjehu u pojedinoj lekciji. Korisnik na temelju toga može zaključiti kojoj lekciji se treba više posvetiti, a koju je savladao. Aplikacija također ima izražen fokus na razvoj vokabulara korisnika kako bi se što bolje i točnije mogao izraziti u bilo kojoj situaciji, bilo to u svakodnevnom životu ili u birokratske svrhe. Uz samu riječ, njen prijevod i značenje korisnicima je ponuđena glasovna snimka izgovora riječi koja im uz sve ostalo može bolje približiti kako je koristiti. Riječi su tematski povezane zbog lakše memorizacije korisnika. Provjerava se temeljito znanje riječi kroz više testova koji se mogu sažeti u znanje hrvatskog prijevoda riječi pomoću zadanoj obliku u stranom jeziku, znanje stranog oblika riječi pomoću zadanoj hrvatski prijevod, znanje izgovora riječi kroz zadani zapis riječi, znanje zapisa riječi pomoću zadanoj izgovora riječi. Učenje se događa primjenom različitih posuda ovisno o korisnikovom točnom ili netočnom odgovoru. Kroz određeno vrijeme riječi na koje je korisnik često točno odgovorio se prestanu pojavljivati jer se smatraju savladanim.

## Potencijalna korist projekta

---

Aplikacija nudi prihvatljiv oblik razvijanja vokabulara uz provjerene metode učenja.

Kroz određenu razinu nadgledanja i pomoć administratora aplikaciju je moguće koristiti u obrazovnim ustanovama i raznim tečajevima.

Korisnici imaju mogućnost individualnog pristupa besplatno.

Aplikacija sama, kroz rezultate korisnika zaključuje koja pitanja postavlja kako bi korisnik što bolje savladao vokabular jezika.

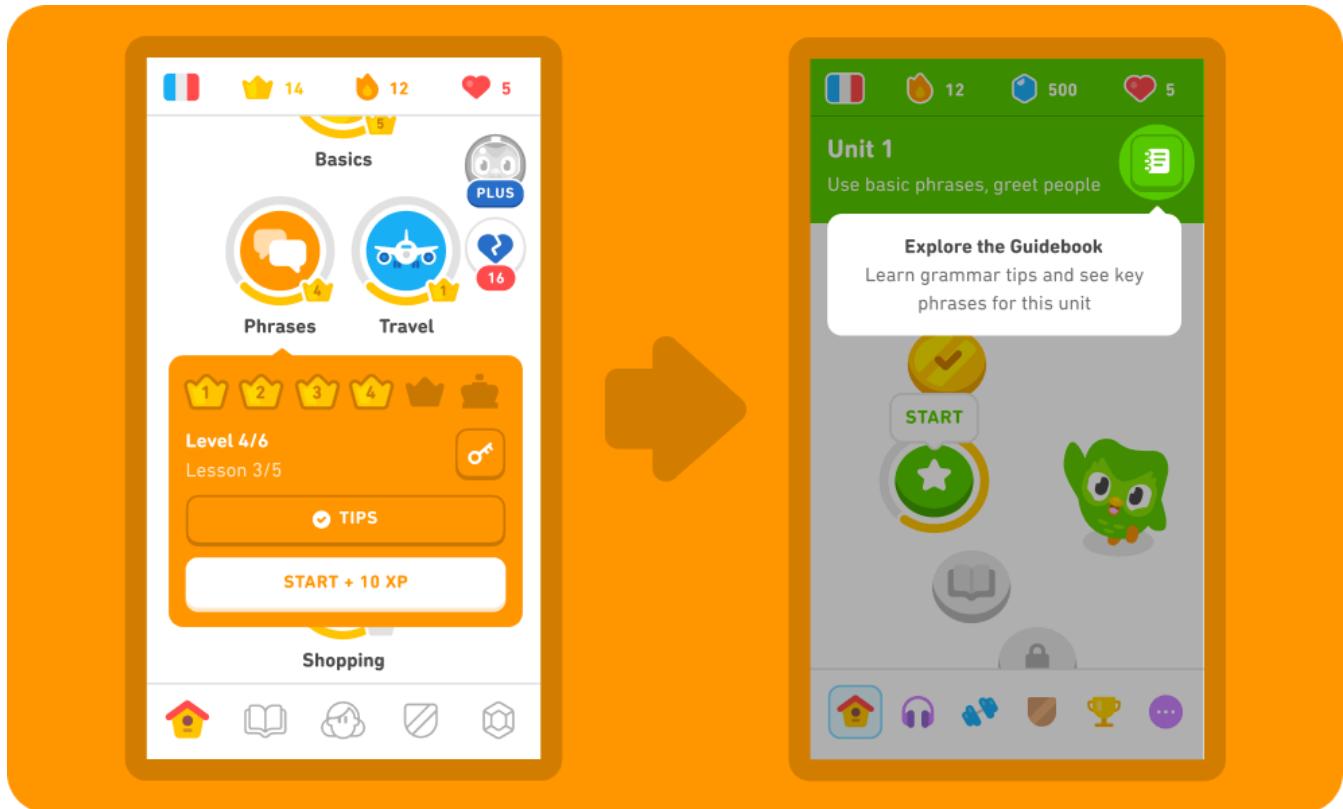
## Slična rješenja

---

Slična rješenja već dostupna na tržištu su aplikacije:

- Duolingo
- Memrise
- Quizlet

Duolingo predstavlja najpoznatiju aplikaciju za učenje stranih jezika, s preko 300 milijuna korisnika. Koristi pristup gamifikacije (korisnici prolaze lekcije i zarađuju bodove nakon rješavanja istih). Iako aplikacija uključuje ponavljanje riječi za razvoj vokabulara korisnika, algoritam nije u potpunosti prilagođen različitom tempu pamćenja svakog korisnika, već se temelji na unaprijed definiranim razinama znanja.



Usporedba s obzirom na FlipMemo: FlipMemo se razlikuje po otvorenosti i modularnosti. Dok Duolingo nudi gotove tečajeve, FlipMemo omogućuje Administratorima stvaranje novih rječnika i riječi. Time je sustav primjenjiv u edukacijskim institucijama i specijaliziranim kolegijima (npr. engleski za inženjere). Također FlipMemo detaljno pohranjuje stanje svakog korisnika.

Memrise je aplikacija koja također koristi metode ponavljanja i kratke videozapise iz stvarnog života kako bi prikazao upotrebu riječi u kontekstu. Koristi algoritme ponavljanja i motivacijske elemente poput levela da bi povećao korisnikov angažman.

**MEM  
RISE**

**Memrise**  
Gamification

**UI SOURCES**

Rank after completing a lesson

Weekly streak

Change daily goal

Usporedba s FlipMemo: Memrise je više orijentiran na unaprijed definirane tečajeve dok FlipMemo ima fluidnu kontrolu nad sadržajem kroz administrativnu promjenu i nadopunu rječnika. Memrise je namijenjen isključivo individualnim korisnicima dok FlipMemo može biti korišten kao obrazovni alat unutar škola i sveučilišta.

Quizlet omogućuje izradu kartica s pojmovima i definicijama te učenje pomoću različitih načina (testovi, igre, ponavljanja). Ne koristi eksplizitno algoritam ponavljanja s odmakom, već ponavljanje na temelju korisnikova izbora.

The screenshot shows the Quizlet search interface. At the top, there is a blue header bar with the Quizlet logo and a search bar containing the text "D'accord 1A, 1A". Below the header, it says "Search results for: D'accord 1A, 1A". There are three main categories displayed: "500 Study Sets", "500 Classes", and "182 Users". Below these categories are sorting options: "Most relevant" (which is selected), "Most recent", and two checkboxes for "Image sets only" and "Teacher-created sets only". The main content area lists seven study sets, each with a title, author, and term count:

- French 1A, D'accord, 1A by Alyson\_Faller (35 terms)
- French 1A, D'accord, 1A by vraman (TEACHER) (35 terms)
- D'accord 1-1A by greenrogers (TEACHER) (43 terms)
- D'accord 1-1A by mayatorres1 (TEACHER) (43 terms)
- D'accord 1-1A by mmegratte (TEACHER) (43 terms)
- D'accord 1-1A by Mme\_Stevens (TEACHER) (44 terms)
- D'accord 1-1A by msamluk (TEACHER) (43 terms)

Usporedba s FlipMemo: FlipMemo uvodi strukturirani sustav ponavljanja i mogućnost glasovnih vježbi. Također, provodi evidenciju vremena između ponavljanja prema modelu posuda, dok Quizlet nema takav mehanizam.

## Mogućnost prilagodbe rješenja

Korisnik trenutno ne može sam stvarati svoj rječnik te u njega stvarati riječi ako je zainteresiran za određen podtip riječi za koje nema rječnik. Moguća implementacija je da korisnici imaju sposobnosti stvarati svoje rječnike te dijeliti ih s drugim korisnicima. Drugi korisnici onda mogu ocjenjivati njihove rječnike na temelju njihove efikasnosti te se po tome mogu voditi budući korisnici.

Također s obzirom na trenutni fokus na engleski jezik može se razmatrati ideja dodavanja novih stranih jezika koja će povećati interes korisnika koji nužno ne zanima engleski jezik već neki drugi.

Mogla bi se i dodati opcija u kojoj korisnici biraju svoj izvorni jezik te im se prema tome nude prijevodi i definicije riječi iz stranog jezika.

## Opseg projektnog zadatka

---

Aplikacija u svojoj osnovi ima funkcije postavljanja pitanja korisniku o vokabularu stranog jezika te postavljanje riječi u određene posude ovisno o odgovoru korisnika. Jedna od najbitnijih stavki je uvid u statistiku individualnog korisnika. Korisnik ima pristup različitim modovima za učenje (npr. učenje riječi, zapisa riječi, izgovora riječi)

Definirane su različite uloge korisnika i administratora. Administratori su korisnici s najvećim ovlastima. Administrator ima pristup stvaranju novih rječnika te dodavanju riječi u rječnike, postoji jedan korijenski administrator koji je predefiniran te on može definirati nove administratore koji imaju jednake ovlasti. Korisnici imaju pristup izradi korisničkog računa koji je povezan s mailom, promjeni korisničkog imena te brisanju istog. Korisnici također mogu promjeniti lozinku korisničkog računa prilikom čega se traži unos prethodne lozinke.

Aplikacija zahtijeva fluidnu bazu podataka u koju se mogu dodavati i izbacivati riječi te konstantno praćenje od strane administratora u svrhu stvaranja novih rječnika te nadopunjavanja starih ovisno o potrebi korisnika.

Aplikacija je trenutno ne invazivna te zbog svog web oblika u pravilu ne može podsjetiti korisnika kada treba vježbati te se temelji na samoj kontinuiranoj zainteresiranosti korisnika.

## Moguće nadogradnje projektnog zadatka

---

Kako bi se FlipMemo unaprijedio i proširio svoju primjenjivost moguće je implementirati sljedeće nadopune:

- Gamifikacija procesa učenja: Uvođenje određenih nagrada za korisnika može povećati samu motivaciju korisnika
- Mobilna aplikacija: Razvoj mobilne aplikacije bi omogućio učenje u pokretu, obavijest o riječima za ponavljanje i veću pristupačnost
- Adaptivno učenje: Integracija umjetne inteligencije koja sama postavlja težinu pitanja i njihov interval ovisno o statistici uspješnosti pojedinog učenika umjesto fiksnog rasporeda posuda

## 2. Analiza zahtjeva

### Funkcijski zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-01.1	Sustav mora omogućiti korisnicima kreiranje računa pomoću e-mail adrese.	Visok	Zahtjev dionika	Korisnik se može registrirati e-mailom, primiti inicijalnu lozinku te se prijaviti.
F-01.2	Sustav mora obavijestiti korisnika o prikupljanju i načinu korištenja osobnih podataka.	Nizak	Postojeći sustav	Korisnik od sustava prima push obavijest s privolom o prikupljanju podataka prije kreiranja korisničkog računa.
F-01.3	Sustav mora omogućiti korisniku odbijanje prikupljanja podataka.	Nizak	Postojeći sustav	Korisnik ima opciju odbiti prikupljanje podataka unutar push obavijesti.
F-02	Sustav mora zahtijevati promjenu inicijalne lozinke.	Visok	Zahtjev dionika	Sustav mora tražiti od korisnika promjenu inicijalne lozinke prilikom prvog prijavljivanja.
F-03	Sustav omogućuje korisniku brisanje računa.	Nizak	Postojeći sustav	Korisnik može samostalno obrisati svoj korisnički račun klikom na gumb "Obriši moj račun".
F-04	Administrator može dodavati nove administratore.	Nizak	Postojeći sustav	Korisnik s najvećim ovlastima (administrator) može dodavati nove administratore koji će imati jednake ovlasti kao on.
F-05	Sustav ima korijenskog administratora.	Visok	Zahtjev dionika	Sustav predefinira jednog od administratora kao korijenskog administratora.
F-06	Riječ se sastoji od četiri komponente.	Visok	Zahtjev dionika	Riječ se sastoji od strane riječi, prijevoda na hrvatski, nekoliko fraza koje bolje opisuju riječ i glasovne datoteke izgovora na stranom jeziku.
F-07	Sustav omogućuje dodavanje novih riječi.	Visok	Zahtjev dionika	Administrator može dodavati nove riječi u bazu podataka (rječnik/e).
F-08.1	Sustav organizira riječi u rječnike.	Visok	Zahtjev dionika	Sustav organizira riječi u rječnike ovisno o temi.
F-08.2	Sustav zahtijeva da se rječniku dodijeli ime.	Visok	Zahtjev dionika	Svaki rječnik mora imati svoje jedinstveno ime.

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-08.3	Rječnici se sastoje od niza riječi.	Visok	Zahtjev dionika	Svaki se rječnik sastoji od niza riječi. Dopušteno je kreirati prazan rječnik.
F-09	Sustav omogućuje dodavanje iste riječi u više rječnika.	Visok	Zahtjev dionika	Sustav omogućuje dodavanje iste riječi u jedan ili više rječnika koji su prethodno definirani.
F-10	Sustav omogućuje brisanje riječi iz rječnika.	Visok	Zahtjev dionika	Administrator može brisati riječi iz baze podataka (rječnika).
F-11	Sustav omogućuje izmjenu riječi iz rječnika.	Visok	Zahtjev dionika	Administrator može mijenjati komponente riječi iz rječnika.
F-12.1	Sustav nudi pomoć prilikom definicije nove riječi.	Srednji	Postojeći sustav	Sustav nudi administratoru pomoć prilikom definicije korištenjem stranog rječnika putem API-ja, npr. <a href="#">RapidAPI</a> .
F-12.2	Sustav omogućuje pretraživanje stranog rječnika prilikom definicije nove riječi.	Srednji	Postojeći sustav	Sustav omogućuje administratoru pretragu stranog rječnika pomoću korisničkog sučelja u kojem administrator upiše dio riječi koju bi definirao te započinje procedura pretrage.
F-13	Aplikacija provodi učenje novih riječi postavljanjem serije pitanja.	Visok	Zahtjev dionika	Korisniku se postavlja niz pitanja sa stranim riječima koje su prethodno definirane u nekom rječniku.
F-14	Aplikacija nudi više odgovora kao moguće točnih.	Visok	Zahtjev dionika	Aplikacija korisniku nudi više opcija (prijevoda) neke strane riječi, od kojih je samo jedan točan.
F-15.1	Aplikacija stavlja riječi u posude, ovisno o točnosti odgovora.	Visok	Zahtjev dionika	Aplikacija na temelju korisnikovog odgovora stavlja riječ u određenu posudu.
F-15.2	Aplikacija za netočan odabir stavlja riječ u prvu posudu u nizu.	Visok	Zahtjev dionika	Korisnik je netočno odgovorio, zbog čega će sustav premjestiti riječ za koju je odabran pogrešan odgovor u prvu posudu u nizu.
F-15.3	Aplikacija za točan odabir stavlja riječ u sljedeću posudu u nizu.	Visok	Zahtjev dionika	Korisnik je točno odgovorio, zbog čega će sustav premjestiti riječ za koju je odabran ispravan odgovor u sljedeću posudu u nizu.
F-15.4	Po isteku vremena određene riječi, aplikacija ponovno prikazuje tu riječ u obliku pitanja.	Visok	Zahtjev dionika	Aplikacija postaje spremna za ponovno prezentiranje neke riječi po isteku njezinog trajanja, u ovisnosti o posudi u kojoj se nalazi.
F-15.5	Aplikacija ne prikazuje ponovno riječi iz posljednje posude po isteku njezinog vremena.	Visok	Zahtjev dionika	Riječi na čija pitanja je korisnik točno odgovorio smatraju se naučenima po isteku vremena u posljednjoj posudi i aplikacija ih više ne prikazuje.
F-16	Učenje u aplikaciji započinje odabirom rječnika.	Visok	Zahtjev dionika	Korisnik po ulasku u aplikaciju odabire jedan od ponuđenih rječnika te započinje proces učenja riječi iz tog rječnika.

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvaćanja
F-17.1	Aplikacija sadrži više modova učenja.	Visok	Zahtjev dionika	Korisnik može odabrat između više modova učenja prilikom odabira rječnika.
F-17.2	Prvi mod učenja temelji se na upitu strane riječi.	Visok	Zahtjev dionika	Aplikacija postavi korisniku pitanje u obliku strane riječi i očekuje odabir ispravnog hrvatskog prijevoda.
F-17.3	Drugi mod učenja temelji se na upitu hrvatske riječi.	Visok	Zahtjev dionika	Aplikacija postavi korisniku pitanje u obliku hrvatske riječi i očekuje odabir ispravnog stranog prijevoda.
F-17.4	Treći mod učenja temelji se na upitu izgovorom strane riječi.	Visok	Zahtjev dionika	Aplikacija postavi korisniku pitanje u obliku izgovorene strane riječi i očekuje ispravan upis riječi na stranom jeziku, čime se provjerava točno pisanje.
F-17.5	Četvrti mod učenja temelji se na upitu tekstualnim oblikom strane riječi.	Visok	Zahtjev dionika	Aplikacija postavi korisniku pitanje u tekstualnom obliku strane riječi i očekuje ispravan izgovor riječi na stranom jeziku, čime se provjerava točan izgovor strane riječi.
F-18	Aplikacija odabire netočne odgovore slučajnim odabirom iz skupa odgovora istog tipa.	Visok	Zahtjev dionika	Aplikacija odabire netočne odgovore slučajnim odabirom iz skupa odgovora istog tipa preostalih riječi (izostavljajući točan odgovor) u slučaju da je korisnik odabrao modove učenja koji podrazumijevaju ponuđeno više odgovora (prvi i drugi).
F-19.1	Aplikacija ima servis za prihvaćanje izgovora strane riječi.	Srednji	Postojeći sustav	Aplikacija ima aplikacijsko sučelje koje prihvata glasovne datoteke.
F-19.2	Aplikacijsko sučelje za izgovor strane riječi odgovara ocjenom izgovora.	Srednji	Postojeći sustav	Aplikacijsko sučelje šalje povratnu informaciju korisniku o njegovom izgovoru određene riječi u obliku ocjene od 1 do 10.
F-19.3	Sustav provjerava točnost izgovora lokalno.	Srednji	Postojeći sustav	Sustav uspoređuje primljenu glasovnu datoteku s postojećom glasovnom datotekom strane riječi u bazi podataka lokalno.
F-20.1	Aplikacija omogućuje korisniku prijavu u sustav.	Srednji	Postojeći sustav	Aplikacija korisniku omogućuje ponovnu prijavu u sustav s postojećim korisničkim imenom i promijenjenom inicijalnom lozinkom klikom na gumb "Prijava" na Home stranici aplikacije.
F-20.2	Aplikacija omogućuje korisniku odjavu iz sustava.	Srednji	Postojeći sustav	Aplikacija korisniku omogućuje odjavu iz sustava klikom na gumb "Odjava" na Home stranici aplikacije.
F-21	Sustav omogućuje korisniku promjenu korisničkog imena.	Nizak	Postojeći sustav	Sustav omogućuje korisniku promjenu korisničkog imena klikom na gumb "Promjeni moje korisničko ime".

# Nefunkcijski zahtjevi

ID zahtjeva	Opis	Prioritet
NF-01	Sustav koristi OAuth2.0 autorizaciju korisnika.	Visok
NF-02	Sustav se treba pridržavati Opće uredbe o zaštiti osobnih podataka.	Visok
NF-03.1	Sustav treba biti oblikovan tako da omogućuje jednostavno održavanje.	Visok
NF-03.1.1	Sustav treba imati dovoljnu dokumentaciju.	Visok
NF-03.1.2	Kôd sustava treba biti dokumentiran prema "Code Conventions for the Java Programming Language" dostupnim na <a href="#">Oracle</a> .	Visok
NF-03.1.3	Sustav treba biti opisan putem dokumenta oblikovanja /SRS/.	Visok
NF-03.1.4	Sustav treba biti popraćen "Priručnikom za rad" koji opisuje pravilnu upotrebu sustava.	Visok
NF-03.1.5	Sustav treba imati "Plan implementacije" za pravilno postavljanje sustava.	Visok
NF-04	Aplikacija treba imati responzivni dizajn za različite uređaje.	Nizak
NF-05.1	Aplikacija odbrojava određeno vrijeme za pojedine posude.	Visok
NF-05.2	Vrijeme trajanja prve posude je jedan dan, druge posude dva dana, treće posude četiri ili više dana i tako sve do posljednje posude (pete ili šeste), čije je vrijeme trajanja nekoliko tjedana (max. 30 dana).	Visok
NF-06.1	Aplikacija za svaki rječnik mora imati dodatnu oznaku koja označava jezik na koji se rječnik odnosi.	Nizak
NF-06.2	Aplikacija grupira rječnike prema oznaci jezika.	Nizak
NF-06.2	Aplikacija redovito prati korisnikov napredak u obliku dnevne i tjedne statistike.	Nizak

## Dionici

Navesti dionike koji imaju interes u ovom sustavu ili su nositelji odgovornosti. To su prije svega korisnici, ali i administratori sustava, naručitelji, razvojni tim. Navesti aktore koji izravno koriste ili komuniciraju sa sustavom. Oni mogu imati inicijatorsku ulogu, tj. započinju određene procese u sustavu ili samo sudioničku ulogu, tj. obavljaju određeni posao. Za svakog aktora navesti funkcionalne zahtjeve koji se na njega odnose.

- Razvojni tim
- Administratori
- Korijenski administrator
- Korisnici

### Aktori i njihovi funkcionalni zahtjevi:

A-1.1 korisnik (inicijator) može: F-01, F-02, F-03, F-16, F-17.1, F-20.1, F-20.2, F-21

A-1.2 korisnik (sudionik) može: F-01.2, F-08.1, F-08.2, F-08.3, F-13, F-14, F-15.1, F-15.2, F-15.3, F-15.4, F-15.5, F-17.2, F-17.3, F-17.4, F-17.5, F-18, F-19.1, F-19.2, F-19.3

A-2.1 administrator (inicijator) može: F-04, F-06, F-07, F-09, F-10, F-11

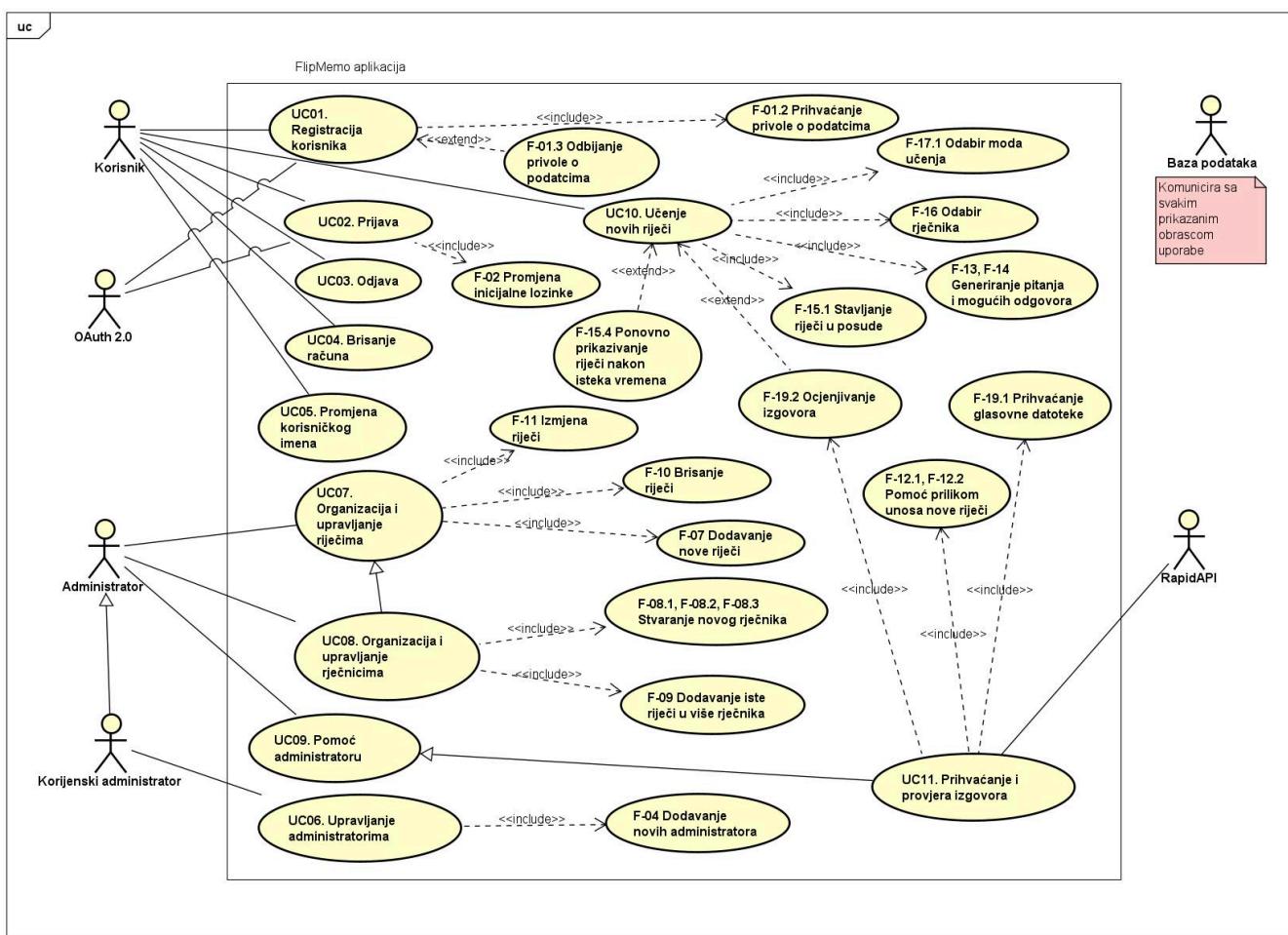
A-2.2 administrator (sudionik) može: F-12.1, F-12.2

A-3 korijenski administrator (sudionik) može: F-05

### 3. Specifikacija zahtjeva sustava

## Obrasci uporabe

### Visokorazinski dijagram obrazaca uporabe cijelog sustava

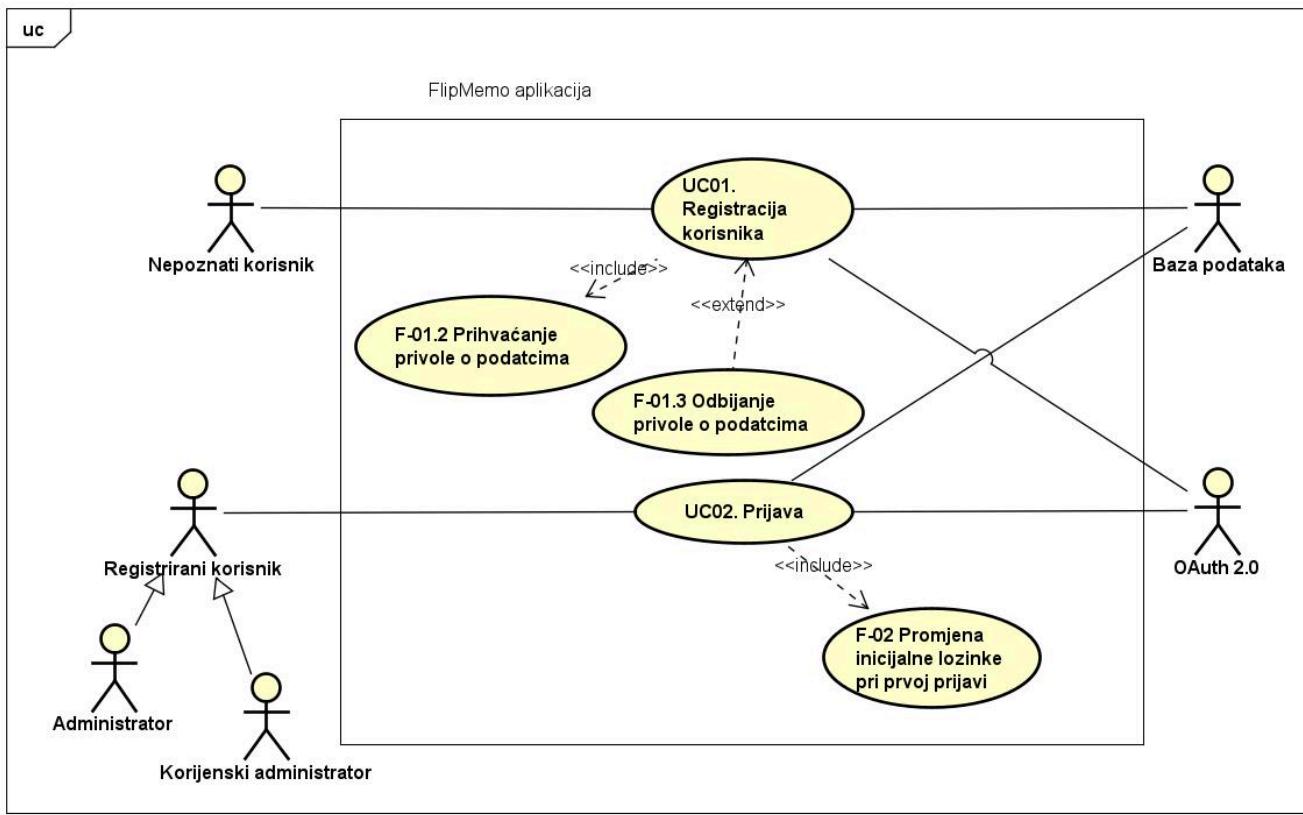


## Ključne funkcionalnosti

### Registracija i prijava

Uključeni obrasci uporabe: UC01 Registracija korisnika, UC02 Prijava.

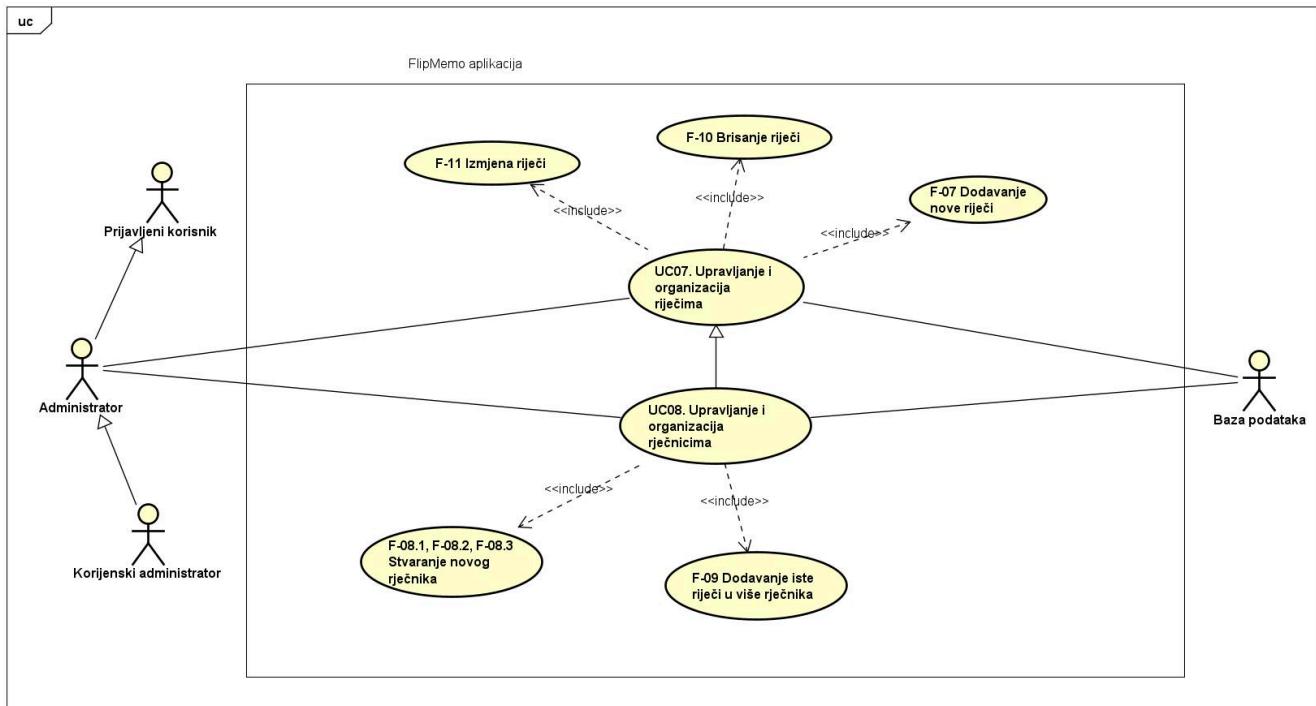
▼ Dijagram obrazaca uporabe



## Organizacija i upravljanje riječima i rječnicima

Uključeni obrasci uporabe: UC07 Organizacija i upravljanje riječima, UC08 Organizacija i upravljanje rječnicima.

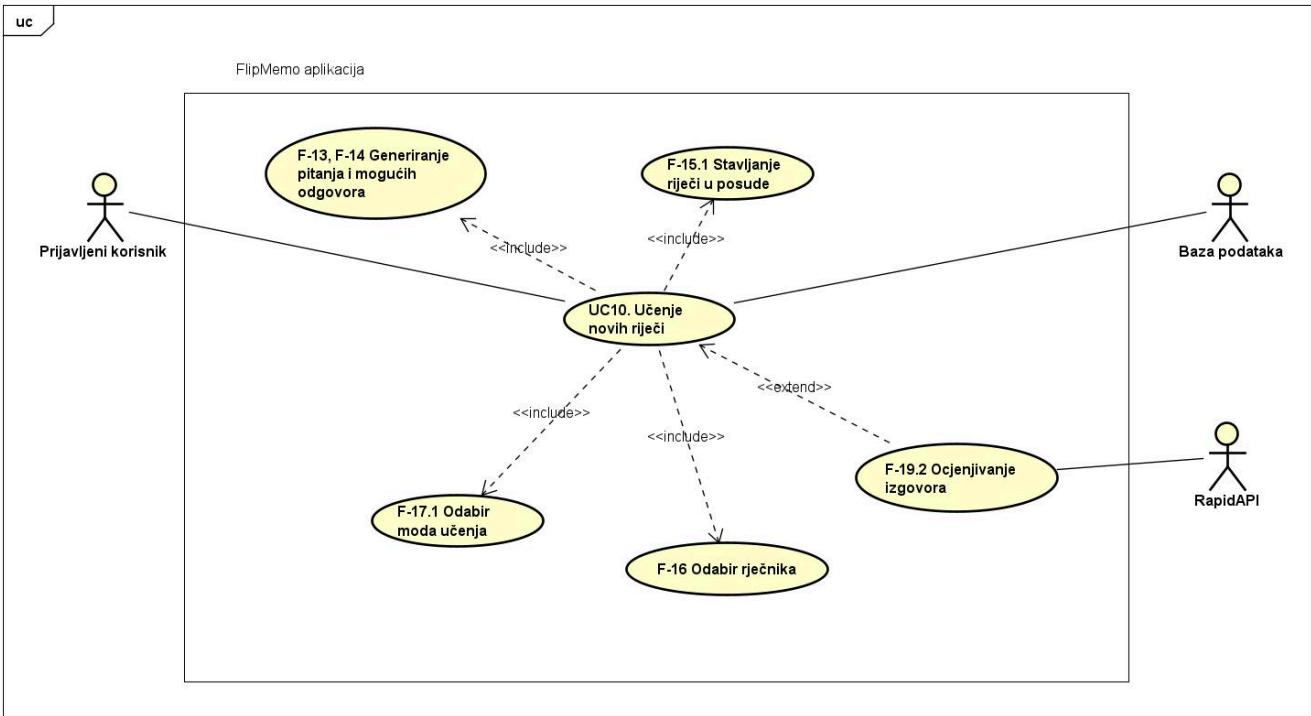
### ▼ Dijagram obrazaca uporabe



## Učenje novih riječi

Uključeni obrasci uporabe: UC10 Učenje novih riječi.

### ▼ Dijagram obrazaca uporabe

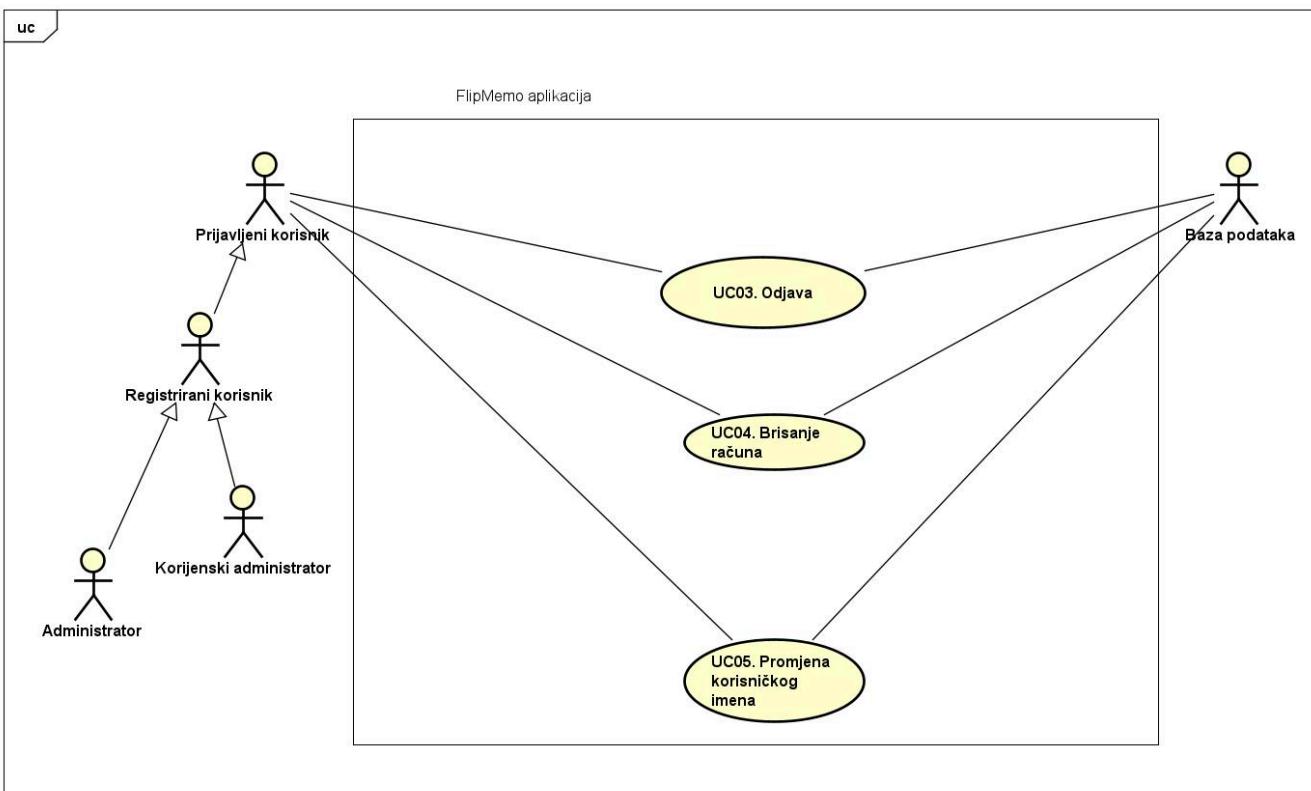


## Korisničke funkcionalnosti

### Odjava, brisanje računa i promjena korisničkog imena

Uključeni obrasci uporabe: UC03 Odjava, UC04 Brisanje računa, UC05 Promjena korisničkog imena.

#### ▼ Dijagram obrazaca uporabe

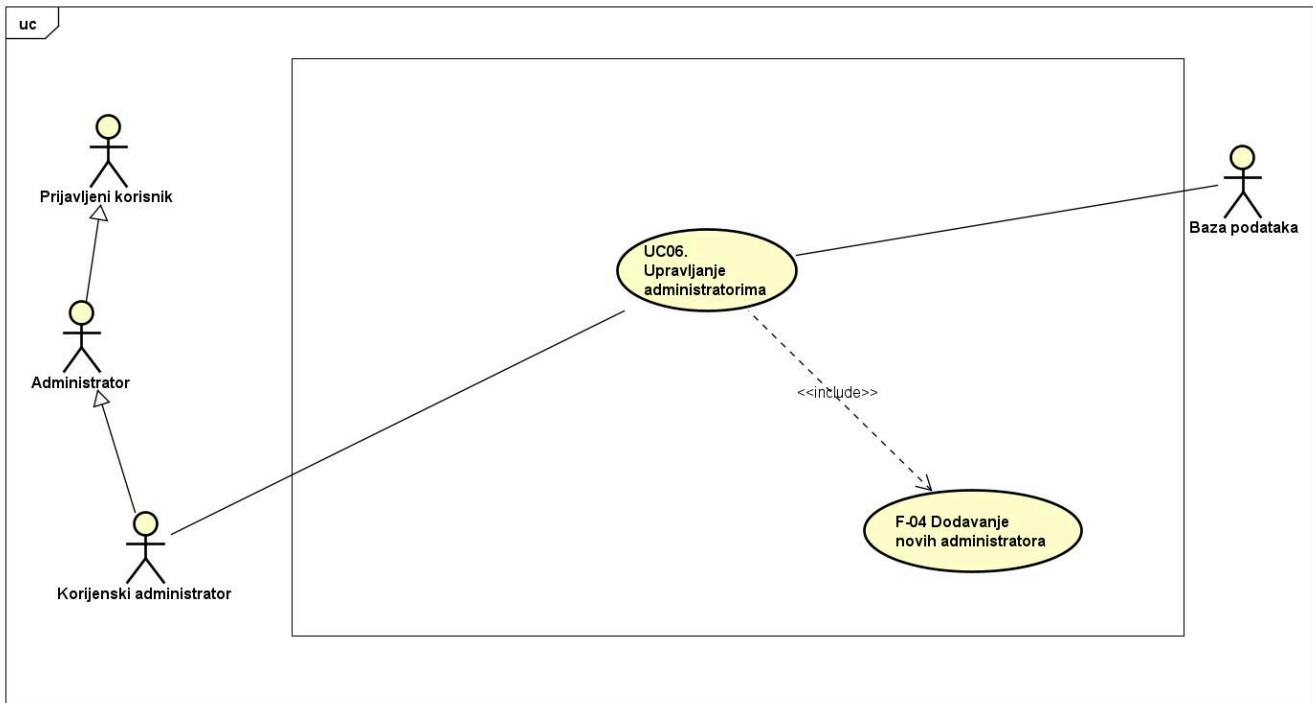


## Osnovni poslovni procesi

### Upravljanje administratorima

Uključeni obrasci uporabe: UC06 Upravljanje administratorima.

▼ Dijagram obrazaca uporabe

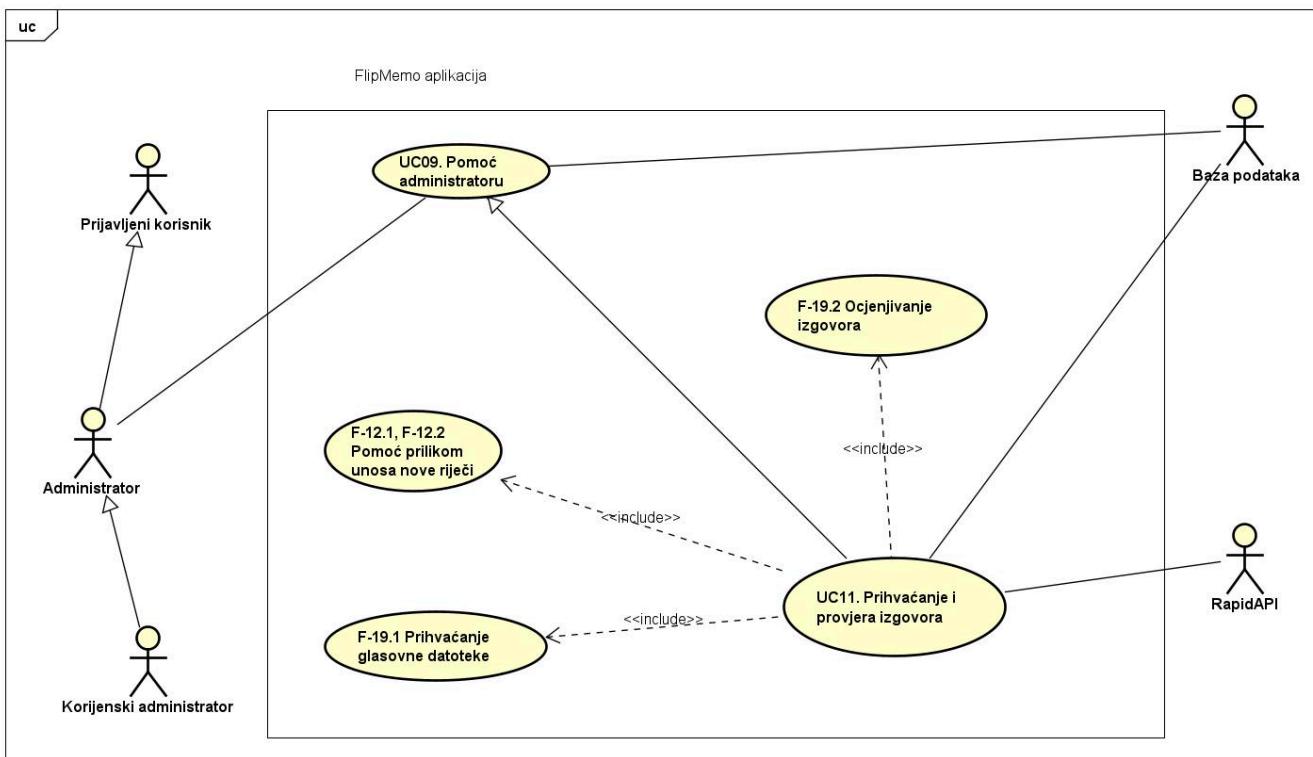


## Kritični sustavi i integracije

### Pomoć administratoru i prihvatanje i provjera izgovora

Uključeni obrasci: UC09 Pomoć administratoru, UC11 Prihvatanje i provjera izgovora.

▼ Dijagram obrazaca uporabe



# Opis obrazaca uporabe

---

## UC01 - Registracija

- Glavni sudionik: Nepoznati korisnik
- Cilj: Omogućiti nepoznatom korisniku registraciju u sustav s pomoću e-maila.
- Sudionici: Korisnik, Google OAuth 2.0 autentifikacijski poslužitelj
- Preduvjet: Korisnik koji se pokušava registrirati nema prethodno stvoren račun u FlipMemo aplikaciji.
- Opis osnovnog tijeka:
  1. Korisnik odabire opciju za registraciju putem Google računa.
  2. Sustav preusmjerava korisnika na Googleov OAuth 2.0 autentifikacijski poslužitelj.
  3. Korisnik se prijavljuje u svoj Google račun (ako već nije prijavljen).
  4. Sustav korisniku prikazuje zahtjev za davanje privole za pristup osnovnim osobnim podatcima (e-mail, ime, prezime), koju on prihvata.
  5. Nakon davanja privole, Googleov autentifikacijski poslužitelj OAuth 2.0 šalje autentifikacijski token aplikaciji.
  6. Aplikacija prima i dekodira token te iz njega dohvaća korisnikove podatke.
  7. Aplikacija provjerava postoji li već korisnik u sustavu.
    - Ako korisnik već postoji, sustav prikazuje poruku: "Već postoji račun s ovom e-mail adresom. Prijavite se."
    - Ako korisnik ne postoji, aplikacija sprema korisnikove podatke u bazu podataka i registrira korisnika.
  8. Registracija je uspješno izvršena, nakon čega se korisnika preusmjerava na Home stranicu aplikacije.
- Opis mogućih odstupanja:
  1. Korisnik može odbiti privolu, nakon čega ga sustav vraća na prethodnu stranicu uz poruku: "Registracija nije dovršena, molimo prihvativte privolu o podatcima."
  2. Tijekom komunikacije s Google OAuth 2.0 sustavom, može doći do greške, o čemu sustav obavještava korisnika te ga upućuje da ponovno pokuša kasnije.

## UC02 - Prijava

- Glavni sudionik: Registrirani korisnik
- Cilj: Omogućiti registriranom korisniku registraciju u sustav s pomoću e-maila.
- Sudionici: Korisnik, Google OAuth 2.0 autentifikacijski poslužitelj
- Preduvjet: Korisnik koji se pokušava prijaviti ima prethodno stvoren račun u FlipMemo aplikaciji.
- Opis osnovnog tijeka:
  1. Korisnik odabire opciju za prijavu putem Google računa.
  2. Sustav preusmjerava korisnika na Googleov OAuth 2.0 autentifikacijski poslužitelj.
  3. Korisnik se prijavljuje u svoj Google račun (ako već nije prijavljen).
  4. Aplikacija prima i dekodira token koji je korisnik primio od OAUTH 2.0 poslužitelja te iz njega dohvaća korisnikove podatke.
  5. Ako korisnik već postoji u bazi podataka, aplikacija ga preusmjerava na početnu stranicu.
  6. Korisnik može pristupiti svome profilu i ostalim funkcionalnostima aplikacije.
- Opis mogućih odstupanja:
  1. Ako korisnik ne postoji u bazi podataka, biva preusmjeren na stranicu za registraciju.

## UC03 Odjava

- Glavni sudionik: Prijavljeni (registrirani) korisnik
- Cilj: Omogućiti prijavljenom korisniku sigurnu odjavu iz sustava.
- Sudionici: Korisnik
- Preduvjet: Korisnik je prethodno prijavljen.
- Opis osnovnog tijeka:

1. Korisnik odabire opciju za odjavu iz FlipMemo aplikacije.
2. Sustav prekida korisničku sesiju.
3. Sustav uklanja sve lokalno spremljene podatke o sesiji (npr. cookies, cache token).
4. Sustav preusmjerava korisnika na početnu stranicu aplikacije.
5. Odjava je uspješno izvršena.

- Opis mogućih odstupanja:

1. Ako dođe do pogreške prilikom prekidanja sesije, sustav prikazuje poruku: "Došlo je do pogreške prilikom odjave. Pokušajte ponovno."

## UC04 Brisanje računa

- Glavni sudionik: Prijavljeni (registrirani) korisnik
- Cilj: Omogućiti prijavljenom korisniku uklanjanje/brisanje računa iz sustava.
- Sudionici: Korisnik, baza podataka sustava
- Preduvjet: Korisnik je prethodno prijavljen.
- Opis osnovnog tijeka:

1. Korisnik u postavkama odabire opciju "Obriši moj račun".
2. Sustav zahtijeva od korisnika potvrdu brisanja.
3. Korisnik potvrđuje da želi trajno obrisati račun.
4. Sustav briše korisničke podatke iz baze podataka (ime, prezime, e-mail, napredak i sl.).
5. Sustav prikazuje korisniku poruku: "Vaš račun je trajno obrisan."
6. Korisnik se automatski odjavljuje iz sustava.

- Opis mogućih odstupanja:

1. Ako korisnik odustane od brisanja, proces se prekida bez promjena.
2. Ako dođe po pogreške prilikom brisanja, sustav prikazuje poruku: "Račun trenutačno nije moguće obrisati. Pokušajte ponovno kasnije."

## UC05 Promjena korisničkog imena

- Glavni sudionik: Prijavljeni (registrirani) korisnik
- Cilj: Omogućiti prijavljenom korisniku promjenu korisničkog imena.
- Sudionici: Korisnik, baza podataka sustava
- Preduvjet: Korisnik je prethodno prijavljen.
- Opis osnovnog tijeka:

1. Korisnik u postavkama aplikacije odabire opciju "Promijeni moje korisničko ime".
2. Korisnik unosi novo korisničko ime.
3. Sustav provjerava dostupnost korisničkog imena.
4. Ako je korisničko ime dostupno, sustav ažurira podatke u bazi.
5. Sustav prikazuje poruku: "Korisničko ime uspješno promijenjeno."

- Opis mogućih odstupanja:

1. Ako korisničko ime nije dostupno, sustav prikazuje poruku: "Odabрано корисниčko име nije dostupno."
2. Ako dođe po pogreške prilikom komunikacije s bazom podataka, sustav prikazuje poruku: "Trenutačno nije moguće promijeniti korisničko imena. Pokušajte ponovno kasnije."

## UC06 Upravljanje administratorima

- Glavni sudionik: Korijenski administrator
- Cilj: Omogućiti korijenskom administratoru dodavanje i brisanje ostalih administratora.
- Sudionici: Korijenski administrator, baza podataka sustava
- Preduvjet: Korijenski administrator je prethodno prijavljen.
- Opis osnovnog tijeka:
  1. Korijenski administrator u svojim postavkama odabire opciju za upravljanje administratorima.
  2. Odabire neku od opcija: "Dodaj", "Uredi" ili "Ukloni" administratora.
  3. Sustav prikazuje odgovarajući obrazac za unos ili izmjenu podataka.
  4. Korijenski administrator unosi potrebne informacije i potvrđuje promjenu.
  5. Sustav ažurira bazu podataka i prikazuje poruku o uspjehu.
- Opis mogućih odstupanja:
  1. Ako su uneseni podaci neispravni, sustav o tome obaveštava korijenskog administratora.
  2. Ako dođe po pogreške prilikom komunikacije s bazom podataka, sustav prikazuje poruku: "Došlo je do pogreške. Pokušajte ponovno kasnije."

## UC07 Organizacija i upravljanje riječima

- Glavni sudionik: Administrator
- Cilj: Omogućiti administratoru dodavanje, uređivanje i brisanje riječi.
- Sudionici: Administrator, baza podataka sustava
- Preduvjet: Administrator je prethodno prijavljen.
- Opis osnovnog tijeka:
  1. Administrator odabire neku od opcija: "Dodaj", "Uredi", "Obriši" riječ.
  2. Sustav prikazuje odgovarajući obrazac za unos podataka.
  3. Korisnik unosi podatke o riječi - njezinim komponentama.
  4. Sustav sprema promjene u bazu podataka.
  5. Sustav prikazuje odgovarajuću poruku o uspješnosti obavljenog posla.
- Opis mogućih odstupanja:
  1. Ako su uneseni podaci nepotpuni, sustav o tome obaveštava administratora.
  2. Ako dođe po pogreške prilikom spremanja riječi u bazu podataka, sustav prikazuje poruku: "Riječ nije uspješno spremljena. Pokušajte ponovno kasnije."

## UC08 Organizacija i upravljanje rječnicima

- Glavni sudionik: Administrator
- Cilj: Omogućiti administratoru upravljanje rječnicima.
- Sudionici: Administrator, baza podataka sustava
- Preduvjet: Administrator je prethodno prijavljen.
- Opis osnovnog tijeka:
  1. Administrator odabire neku od opcija: "Kreiraj", "Uredi", "Obriši" riječ.

2. Sustav prikazuje odgovarajući obrazac za unos podataka.
3. Korisnik unosi podatke u rječnik ili o rječniku - dodaje nove riječi u njega, mijenja mu ime i sl..
4. Sustav sprema promjene u bazu podataka.
5. Sustav prikazuje odgovarajuću poruku o uspješnosti obavljenog posla.

- Opis mogućih odstupanja:

1. Ako su uneseni podaci nepotpuni, sustav o tome obaveštava administratora.
2. Ako dođe po pogreške prilikom spremanja promjena u rječniku, sustav prikazuje poruku: "Rječnik nije ažuriran. Pokušajte ponovno kasnije."

## UC09 Pomoć administratoru

- Glavni sudionik: Administrator
- Cilj: Omogućiti administratoru korištenje RapidAPI platforme prilikom kreiranja nove riječi.
- Sudionici: Administrator, RapidAPI platforma, baza podataka sustava
- Preduvjet: Administrator je prethodno prijavljen.
- Opis osnovnog tijeka:
  1. Administrator prilikom definiranja nove riječi odabire opciju "Pomoć".
  2. Sustav povezuje korisnika s RapidAPI platformom.
  3. U odgovarajućem obrascu administrator ima opciju pretraživanja stranog rječnika tako da upiše dio riječi koju želi definirati.
  4. RapidAPI platforma ponudi administratoru više riječi koje odgovaraju unesenim slovima.
  5. Administrator odabire neku od ponuđenih riječi i odabire opciju "Spremi".
  6. Sustav zabilježi promjene u bazi podataka.
  7. Sustav šalje odgovarajuću poruku administratoru po završetku spremanja promjene.

- Opis mogućih odstupanja:

1. Ako dođe po pogreške prilikom komunikacije s RapidAPI platformom, sustav prikazuje poruku: "Došlo je do greške. Rječnik nije ažuriran."

## UC10 Učenje novih riječi

- Glavni sudionik: Prijavljeni (registrirani) korisnik
- Cilj: Omogućiti korisniku učenje novih riječi i praćenje njegovog napretka.
- Sudionici: Korisnik, baza podataka sustava
- Preduvjet: Korisnik je prethodno prijavljen.
- Opis osnovnog tijeka:
  1. Korisnik odabire mod učenja.
  2. Ovisno o odabranom modu učenja, sustav započinje proces učenja.
  3. Ako je odabran prvi mod učenja, aplikacija prikazuje korisniku igru u kojoj korisnik dobiva stranu riječ i ponuđeno više prijevoda te treba odabrati onaj koji je ispravan.
  4. Ako je odabran drugi mod učenja, aplikacija prikazuje korisniku igru u kojoj korisnik dobiva hrvatsku riječ, a treba odabrati njezin prijevod na stranom jeziku, od više mogućih odgovora.
  5. Ako je odabran treći mod učenja, aplikacija daje korisniku glasovnu datoteku s izgovorenom stranom riječi, a od njega se očekuje da unese ispravno napisanu riječ na stranom jeziku.
  6. Ako je odabran četvrti mod učenja, aplikacija prikazuje korisniku riječ na stranom jeziku te od njega očekuje da kao odgovor preda glasovnu datoteku s ispravno izgovorenom riječi.
  7. Sustav unaša korisnikove odgovore u bazu podataka.

8. Ako je korisnik točno odgovorio, sustav stavlja odigranu riječ u sljedeću posudu u nizu, ovisno o tome gdje se tada nalazila, što je zapisano u bazi podataka.
9. Sa svakom novom odigranom riječi, sustav bilježi korisnikov napredak u bazi podataka.
10. Ako je korisnik netočno odgovorio, sustav stavlja odigranu riječ u prvu posudu u nizu.
11. Sustav prikazuje obrazac za igru dok se ne odigraju sve riječi ili dok korisnik ne odluči zaustaviti učenje.
12. Po završetku, sustav spremi promjene u bazu podataka.
13. Sustav prikazuje korisniku poruku za završetak igre, s ažuriranom statistikom i napretkom.

- Opis mogućih odstupanja:

1. Ako dođe do greške ili smetnji tijekom učenja riječi, sustav resetira igru i bilježi napredak do trenutka u kojem je došlo do prekida te o tome obavještava korisnika u obliku poruke: "Došlo je do greške tijekom učenja. Napredak je spremlijen djelomično."

## UC11 Prihvaćanje i provjera izgovora

- Glavni sudionik: Prijavljeni (registrirani) korisnik
- Cilj: Omogućiti prihvaćanje izgovora u sustavu te slanje korisniku povratne informacije.
- Sudionici: Korisnik, baza podataka sustava, RapidAPI platforma
- Preduvjet: Korisnik je prethodno prijavljen.
- Opis osnovnog tijeka:
  1. Korisnik odabire četvrti mod učenja.
  2. Sustav se povezuje s RapidAPI platformom.
  3. Sustav prikazuje korisniku obrazac sa stranom riječi i dijelom u koji može umetnuti glasovnu datoteku svojeg izgovora te riječi.
  4. Sustav prihvata korisnikov odgovor i šalje ga RapidAPI-ju te svojoj bazi podataka.
  5. Sustav uspoređuje primljenu glasovnu datoteku s postojećom glasovnom datotekom u svojoj bazi podataka.
  6. RapidAPI šalje korisniku povratnu informaciju o njegovom izgovoru u obliku ocjene od 1 do 10.
  7. Po završetku učenja, sustav prekida konekciju s RapidAPI platformom.
  8. Sustav spremi korisnikov napredak u bazu podataka.
  9. Sustav prikazuje korisniku poruku za završetak igre, s ažuriranom statistikom i napretkom.

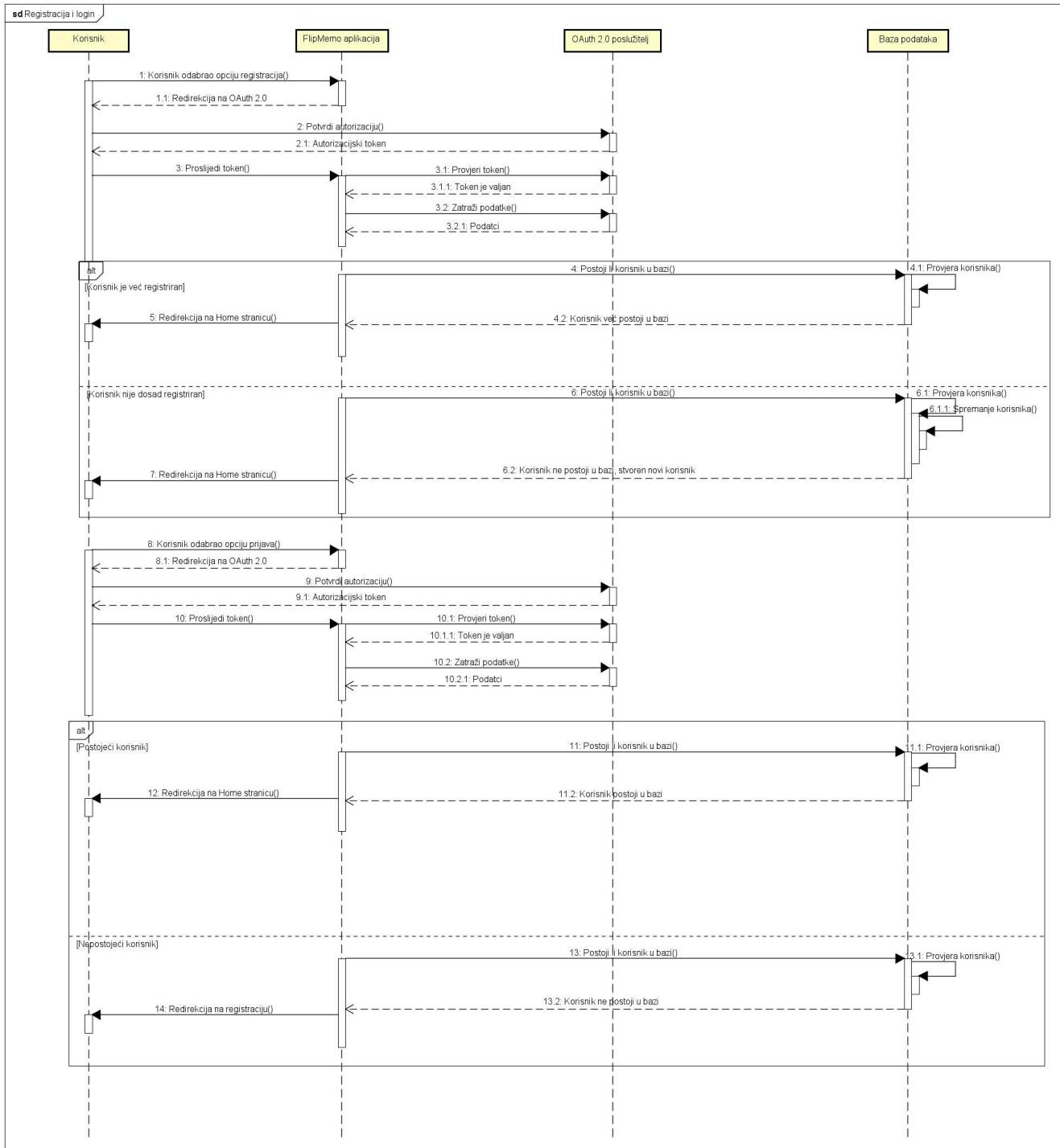
- Opis mogućih odstupanja:

1. Ako dođe do smetnji u konekciji s RapidAPI platformom, sustav nastavlja igru bez pružanja povratne informacije korisniku o njegovom izgovoru. Pritom mu prikazuje poruku: "Nemogućnost spajanja sa servisom RapidAPI, igra se nastavlja bez ocjene izgovora."
2. Ako dođe do neke druge greške, sustav prikazuje poruku: "Došlo je do greške. Molimo pokušajte kasnije."

## Sekvencijski dijagrami

### ▼ Registracija i prijava

Ovaj sekvenčni dijagram prikazuje kako funkcioniraju registracija i prijava između korisnika, FlipMemo aplikacije, OAuth 2.0 Google poslužitelja te baze podataka. Najprije se korisnik pokuša registrirati/prijaviti pomoću e-mail adrese. Tada ga se preusmjeri na stranicu Google OAuth 2.0 autentifikacijskog poslužitelja. Od njega korisnik prima autorizacijski token koji prosljeđuje aplikaciji koja za njega provjerava njegovu valjanost. Kada je korisnik prošao prvi korak, aplikacija provjerava postoji li korisnik s tom e-mail adresom već u njegovoj bazi podataka. Ako ne postoji u slučaju registracije, aplikacija FlipMemo vrši registraciju korisnika i preusmjerava ga na svoju Home stranicu. Ako ne postoji u slučaju prijave, aplikacija ga preusmjerava na stranicu za registraciju. Ako korisnik već postoji u bazi podataka, tada ga u slučaju registracije preusmjerava na stranicu za prijavu. U slučaju da korisnik već postoji i pokušava se prijaviti, aplikacija ga preusmjerava na Home stranicu.

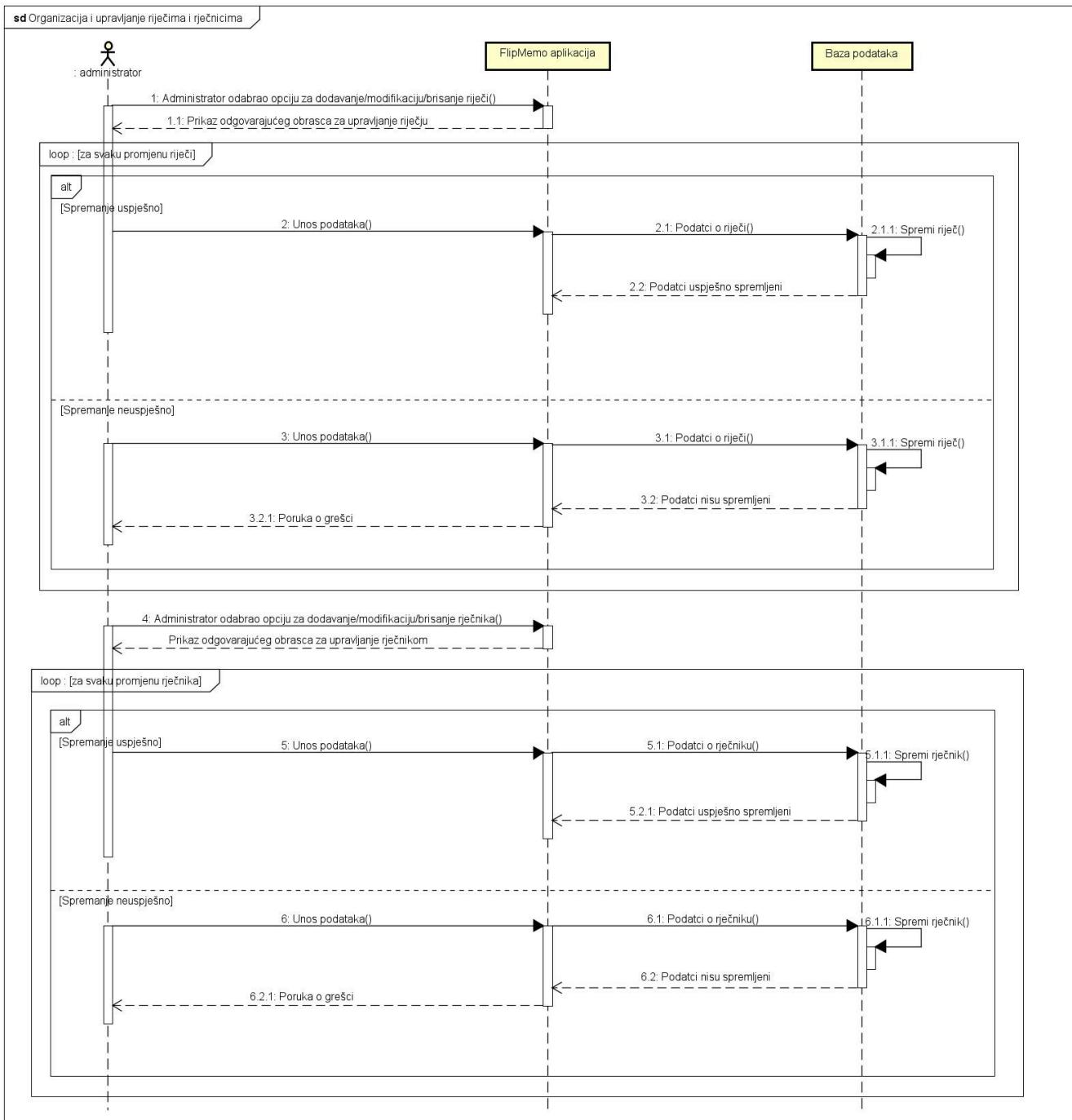


### ▼ Organizacija i upravljanje riječima i rječnicima

Ovaj sekvencijski dijagram prikazuje način organizacije i upravljanja riječima i rječnicima u sklopu FlipMemo aplikacije. Administrator odabire dodavanje/modifikaciju/brisanje riječi ili rječnika. Sustav mu prikazuje odgovarajući obrazac za unos podataka. Nakon što administrator unese podatke, aplikacija ih proslijeđuje bazi podataka. Podatci se unose u bazu.

Ako je spremanje podataka uspjelo, baza podataka šalje potvrdu o spremanju aplikaciji. Administrator ne prima nikakvu obavijest.

Ako spremanje podataka nije uspjelo, baza podataka šalje aplikaciji obavijest o neuspjehu. Administrator prima poruku o grešci: "Riječ nije uspješno spremljena. Pokušajte ponovno kasnije." ili "Rječnik nije ažuriran. Pokušajte ponovno kasnije."



### ▼ UC10 Učenje novih riječi

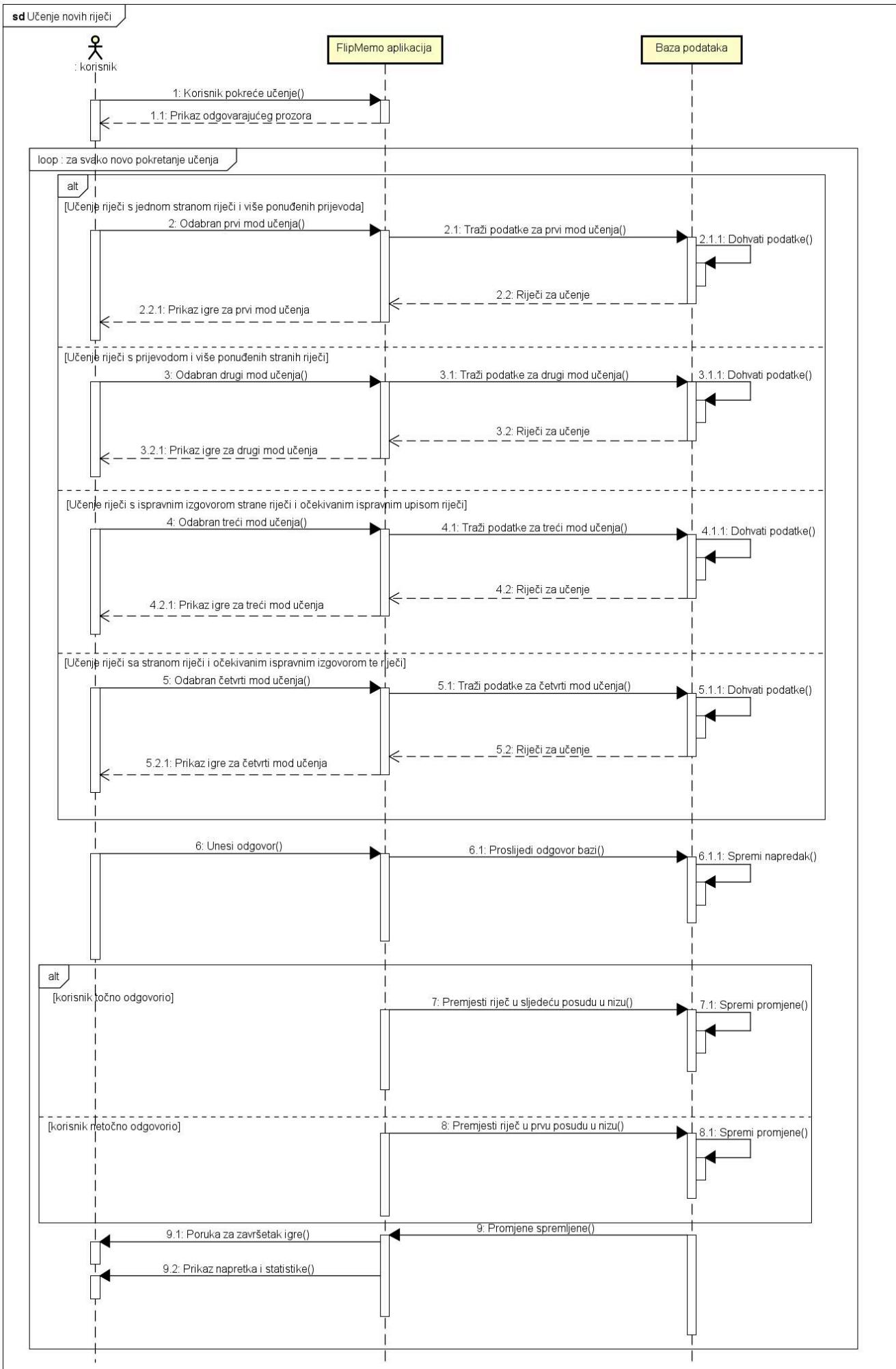
Ovaj sekvenčni dijagram prikazuje proces učenja novih riječi. On započinje kada korisnik odabere jedan od četiri moda učenja. Tada aplikacija iz baze podataka dohvaća riječi za učenje. Korisniku sustav postavlja niz pitanja s rijećima i više mogućih odgovora. Korisnik odgovara na pitanja. Aplikacija prosljeđuje bazi podataka korisnikov odgovor nakon svake odigrane riječi.

Ako je korisnik točno odgovorio, sustav prosljeđuje promjenu bazi podataka, koja stavlja riječ u posljednju posudu u nizu.

Ako je korisnik netočno odgovorio, sustav prosljeđuje promjenu bazi podataka, koja stavlja riječ u prvu posudu u nizu.

Na kraju igre, sustav šalje korisniku poruku o završetku te njegov napredak i statistiku.

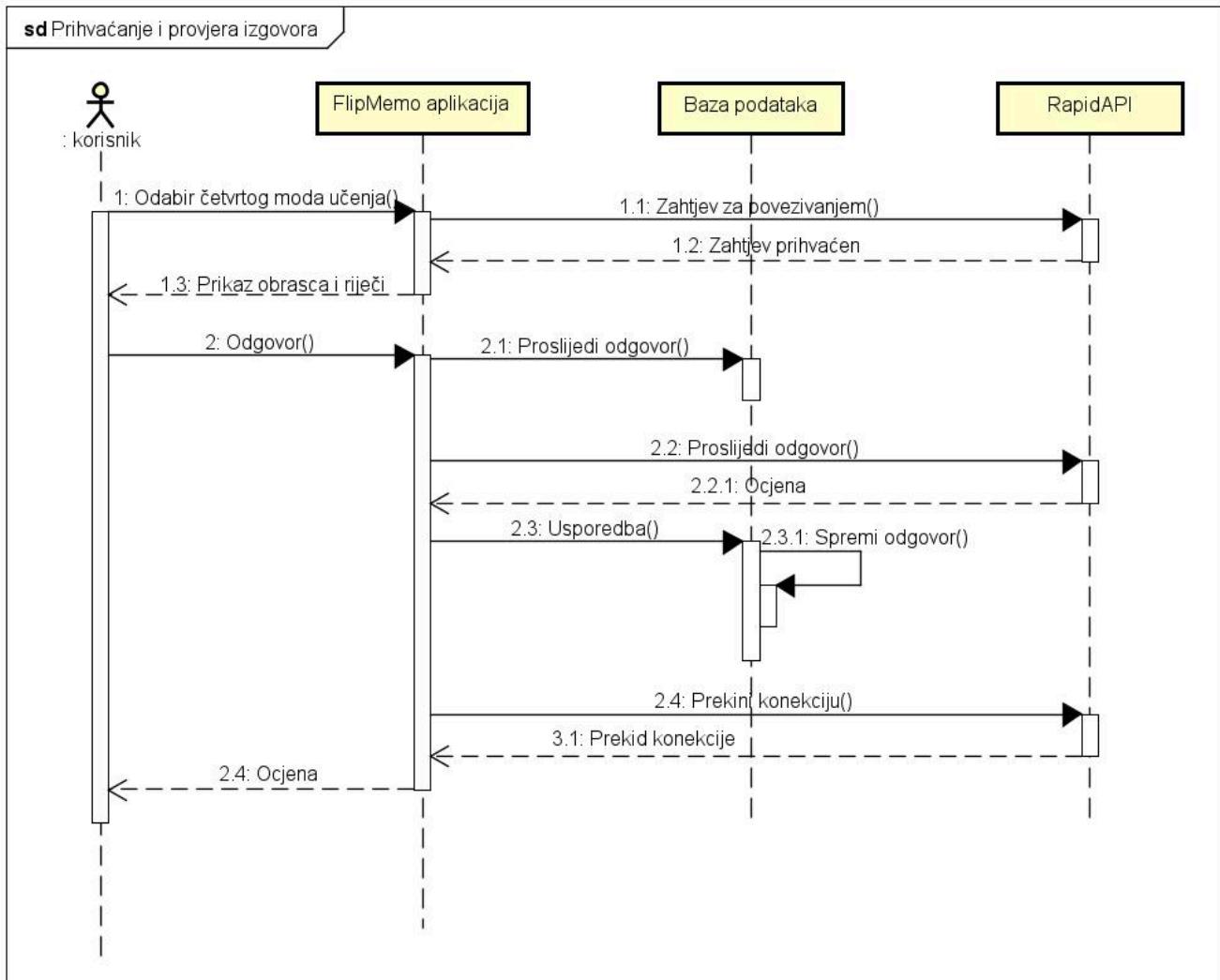
Ako je došlo do greške, sustav korisniku prikazuje poruku: "Došlo je do greške tijekom učenja. Napredak je spremlijen djelomično."



### ▼ UC11 Prihvaćanje i provjera izgovora

Ovaj sekvensijski dijagram prikazuje proces prihvaćanja i provjere korisnikovog izgovora riječi. Preduvjet je da korisnik odabere četvrti mod učenja. Najprije aplikacija traži potvrdu za konekciju od RapidAPI servisa. Nakon uspostave konekcije, aplikacija prikazuje korisniku odgovarajući obrazac s riječi i prozorčićem za prijenos glasovne datoteke. Sustav unesenu datoteku proslijeđuje RapidAPI platformi i svojoj bazi podataka. RapidAPI šalje ocjenu izgovora aplikaciji, koja to prosjeđuje primljenu snimku zvuka s postojećim snimkama u bazi podataka te sprema promjene u bazu.

Ako je došlo do greške, sustav korisniku prikazuje poruku: "Došlo je do greške. Molimo pokušajte kasnije."



## Provjera uključenosti ključnih funkcionalnosti u obrascu uporabe

ID obrasca	Naziva obrasca uporabe	Obuhvaćeni funkcijски zahtjevi
UC01	Registracija korisnika	F-01.1, F-01.2, F-01.3
UC02	Prijava	F-20.1
UC03	Odjava	F-20.2
UC04	Brisanje računa	F-03
UC05	Promjena korisničkog imena	F-21

ID obrasca	Naziva obrasca uporabe	Obuhvaćeni funkcijski zahtjevi
UC06	Upravljanje administratorima	F-04, F-05
UC07	Organizacija i upravljanje riječima	F-06, F-07
UC08	Organizacija i upravljanje rječnicima	F-08.1, F-08.2, F-08.3, F-09, F-10, F-11
UC09	Pomoć administratoru	F-12.1, F-12.2
UC10	Učenje novih riječi	F-13, F-14, F-15.1, F-15.2, F-15.4, F-15.5, F-16, F-17.1, F-17.2, F-17.3, F-17.4, F-17.5, F-18
UC11	Prihvatanje i provjera izgovora	F-19.1, F-19.2, F-19.3

## 4. Arhitektura i dizajn sustava

---

### Arhitektura sustava

Arhitektura koja se koristi u FlipMemo sustavu je arhitektura s Model-View-Control pristupom. Ova Arhitektura je odabrana zbog svojih prednosti u jasnom odvajanju odgovornosti (svaki sloj ima svoju ulogu), lakše testiranje i održavanje (budući da su logika, prikaz i upravljanje odvojeni, svaki dio može se zasebno testirati i razvijati), mogućnost paralelnog razvoja (različiti članovi tima mogu raditi na različitim komponentama istovremeno), povećana fleksibilnost i proširivost (promjene u korisničkom sučelju ne utječu na poslovnu logiku i obrnuto), jasna struktura projekta tijekom dizajna aplikacije (kod je organiziran i skalabilan).

### Opis arhitekture

#### Stil arhitekture

Odabran stil arhitekture je klijent-poslužitelj jer omogućava jednostavnu komunikaciju preko mreže i lako je proširiva odnosno, više klijenata može koristiti usluge s jednim serverom. U pravilu ovaj stil se smatra standardnim modelom za web aplikacije. Server centralno upravlja podacima i logikom. Nedostatak ovog stila je taj da u slučaju pada servera cijeli sustav prestaje raditi zbog čega je potrebna zaštita i dobro balansiranje opterećenja.

#### Podsistavi

U sustavu FlipMemo koriste se idući podsustavi:

- Korisnički podsustav koji omogućuje prijavu i registraciju korisnika. Za prijavu putem vanjskih identitetskih davatelja koristi se autorizacijski protokol OAuth 2.0
- Podsistav za učenje koji predstavlja ključni dio aplikacije, a čine ga mogućnost upravljanja karticama, stvaranje, uređivanje i brisanje riječi, implementacija spaced repetition algoritma te prikaz riječi i prijevoda
- Podsistav za praćenje napretka koji se brine o pohrani rezultata učenja, generiranju statistike i praćenja aktivnosti korisnika.
- Podsistav za obavijesti koji ovisno o aktivnosti korisnika šalje obavijesti putem push notifikacija ili e-maila.

#### Preslikavanje na radnu platformu

U implementacija frontenda koristi se React.js.

Za backend server koristi se Node.js.

Baza podataka je relacijska baza podataka dizajnirana u aplikaciji PostgreSQL, a s aplikacijom je spojena pomoću aplikacije Docker.

Koristi se Rest API za komunikaciju između frontenda i backenda.

#### Spremišta podataka

Relacijska baza podataka koja sprema podatke o korisnicima, riječima i jezicima.

## Mrežni protokoli

HTTP: osnovni komunikacijski protokol koji omogućava prijenos podataka između klijenta i servera. OAuth 2.0: standardni protokol za autorizaciju koji omogućava aplikacijama pristup resursima korisnika.

## Globalni upravljački tok

Korisnik se prijavljuje ili registrira. Aplikacijski sloj obrađuje sve zahtjeve korisnika bilo to pokretanje lekcije ili uvid u statistiku što se korisniku prikazuje putem klijentskog sloja. Podaci u bazi podataka su individualni za svakog korisnika te korisnik ima pristup samo svojim podacima što kontrolira backend. Korisnici ovisno o svojoj aktivnosti i trenutačnom uspjehu (količini posuda) dobivaju push notifikacije te obavijesti putem e-maila.

## Sklopopskoprogramski zahtjevi

- Poslužitelj treba imati spremno dovoljno resursa za održavanje servera i baze podataka
- Korisnik treba imati pristup uređaju koji podržava moderan web-preglednik

## Obrazloženje odabira arhitekture

---

Za realizaciju projekta odabrana je MVC arhitektura (Model–View–Controller) zbog svoje široke primjene i dokazane učinkovitosti u razvoju web-aplikacija. MVC se smatra standardom u suvremenom web dizajnu jer omogućuje jasno odvajanje odgovornosti između logike podataka, korisničkog sučelja i kontrolnog toka aplikacije. Takva struktura doprinosi boljoj organiziranosti koda, olakšava održavanje te omogućuje paralelni razvoj više dijelova sustava.

Zahvaljujući svojoj modularnosti i skalabilnosti, MVC arhitektura omogućuje jednostavno nadograđivanje i prilagodbu aplikacije budućim potrebama. Dodatna prednost je mogućnost zasebnog testiranja svake komponente (modela, prikaza i kontrolera), čime se povećava pouzdanost sustava i olakšava pronalaženje pogrešaka. Sve navedeno čini MVC arhitekturu logičnim i učinkovitim izborom za izradu ovog projekta, koji zahtijeva preglednu strukturu, lakoću održavanja i mogućnost daljnog razvoja.

## Organizacija sustava na visokoj razini

---

### Klijent-poslužitelj

Sustav je organiziran prema klijent-poslužitelj arhitekturi, gdje su funkcionalnosti podijeljene između:

Klijenta:

- Web preglednik ili mobilna aplikacija koja pruža korisničko sučelje učeniku ili administratoru
- Prikazuje rječnike, pitanja, modove učenja, registraciju i prijavu korisnika
- Komunicira sa serverom putem REST API-ja

Poslužitelja:

- Node.js aplikacija koja obrađuje zahtjeve klijenta i provodi poslovnu logiku

Funkcionalnosti: generiranje pitanja za učenje, upravljanje napretkom učenika kroz spaced repetition, autentikacija i autorizacija (OAuth2), pohrana i dohvata podataka te komunikacija s vanjskim servisima (rječnici, ocjena izgovora).

### Uloga baze

Sustav koristi relacijsku bazu podataka (PostgreSQL) za trajnu pohranu svih podataka.

Pohrana entiteta: User, Language, Words

- Upravljanje napretkom učenika kroz posude
- Pohrana audio zapisa riječi izgovorenih na engleskom jeziku u polju BYTEA, što omogućuje direktni binarni unos u bazu bez eksternog storage-a
- Omogućuje indeksiranje i upite za dohvati riječi spremnih za ponavljanje

## Grafičko sučelje

Funkcionalnosti:

- Registracija i prijava učenika, promjena lozinke
- Pregled i odabir rječnika po jeziku
- Učenje riječi kroz različite modove: multiple-choice, tipkanje, izgovor (evaluacija audio zapisa)
- Administracija rječnika i riječi za korisnike s administratorskim ovlastima

Povezanost s backendom:

- Sučelje šalje HTTP zahtjeve backendu za dohvati i pohranu podataka
- Audio datoteke se dohvaćaju direktno iz baze (BYTEA polja) i reproduciraju u pregledniku ili mobilnoj aplikaciji

## Organizacija aplikacije

---

Aplikacija je organizirana u dva osnovna sloja – frontend i backend, koji zajedno čine cjeloviti sustav za učenje stranih jezika. Ovakva dvoslojna arhitektura omogućuje jasno razdvajanje odgovornosti između korisničkog sučelja, algoritma za učenje jezika i podatkovnog sloja te olakšava održavanje i proširivost sustava.

Glavne komponente frontenda su:

- App.jsx – početna komponenta aplikacije koja upravlja rutama i autentifikacijom korisnika
- Igra.jsx – komponenta zadužena za provođenje igre i praćenje napretka korisnika
- Postavkelgre.jsx i PostavkeRjecnika.jsx – omogućuju konfiguraciju modova igre, jezika i rječnika
- Prijava.jsx i Registracija.jsx – služe za autentifikaciju i registraciju korisnika
- Profil.jsx – omogućuje pregled i uređivanje korisničkih podataka
- UpravljanjeUlogama.jsx – administratoru omogućuje dodjelu uloga i upravljanje korisnicima

Backend je implementiran pomoću Node.js okruženja, uz korištenje Express frameworka za izgradnju RESTful API-ja

Odgovornosti backend sloja uključuju:

- obradu zahtjeva za registraciju, prijavu i autentifikaciju korisnika
- pohranu i dohvati podataka o jezicima, riječima i napretku korisnika
- provjeru korisničkih uloga i prava pristupa
- slanje odgovora frontendu u JSON formatu

MVC arhitektura

Sustav slijedi principe MVC (Model–View–Controller) arhitekture:

- Model (M) – obuhvaća sve podatkovne strukture i logiku vezanu uz rad s bazom podataka (npr. modeli za korisnike, jezike i riječi). Model definira strukturu tablica i veze među njima te omogućuje dohvati i ažuriranje podataka.
- View (V) – implementiran u React komponentama na frontend strani. View prikazuje podatke korisniku i omogućuje interakciju putem forme, gumba i drugih elemenata sučelja.
- Controller (C) – implementiran u Node.js backendu i služi kao posrednik između Modela i View-a. Controller obrađuje HTTP zahtjeve, poziva modele prema potrebi, provodi poslovnu logiku te vraća rezultate frontendu u obliku JSON odgovora.

# Baza podataka

---

## Opis tablica

---

### Users tablica

Atribut	Tip podatka	Opis varijable
name	Text	korisničko ime
email	Text	primary key
password	Text	lozinka korisnika
role	Smallint	uloga (npr. korisnik, admin)

### Languages tablica

Atribut	Tip podatka	Opis varijable
language_id	Serial	primary key
languages_name	Text	Unique (ime jezika)

### Words tablica

Atribut	Tip podatka	Opis varijable
word_id	Serial	primary key
word_text	Text	pisani oblik riječi
language_id	Int	jezik kojem pripada
translation_to_croatian	Text	hrvatski prijevod riječi
phrases	Text	fraze u kojima je riječ korištena
pronounciation	BYTEA	izgovor riječi

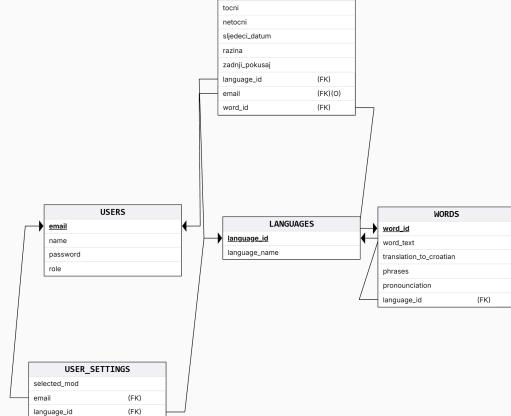
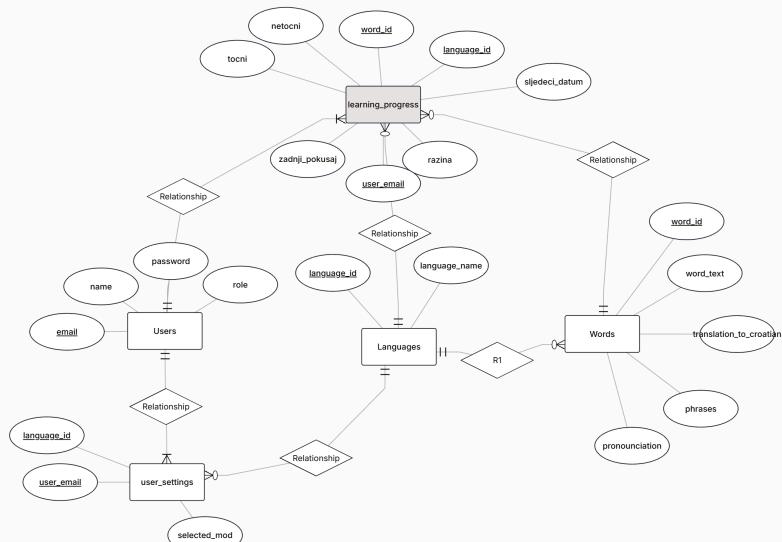
### learning\_progress tablica

Atribut	Tip podatka	Opis varijable
user_email	Text	referenca na Users(email), dio primarnog ključa
language_id	Int	referenca na Languages(language_id), dio primarnog ključa
word_id	Int	referenca na WordS(word_id), dio primarnog ključa
razina	Smallint	trenutna razina korisnika
sljedeci_datum	Timestamptz	idući datum kada korisnik treba igrati igru
zadnji_pokusaj	Timestamptz	kada je korisnik zadnji put igrao igru
tocni	Int	broj točnih odgovora
netocni	Int	broj netočnih odgovora

## user\_settings tablica

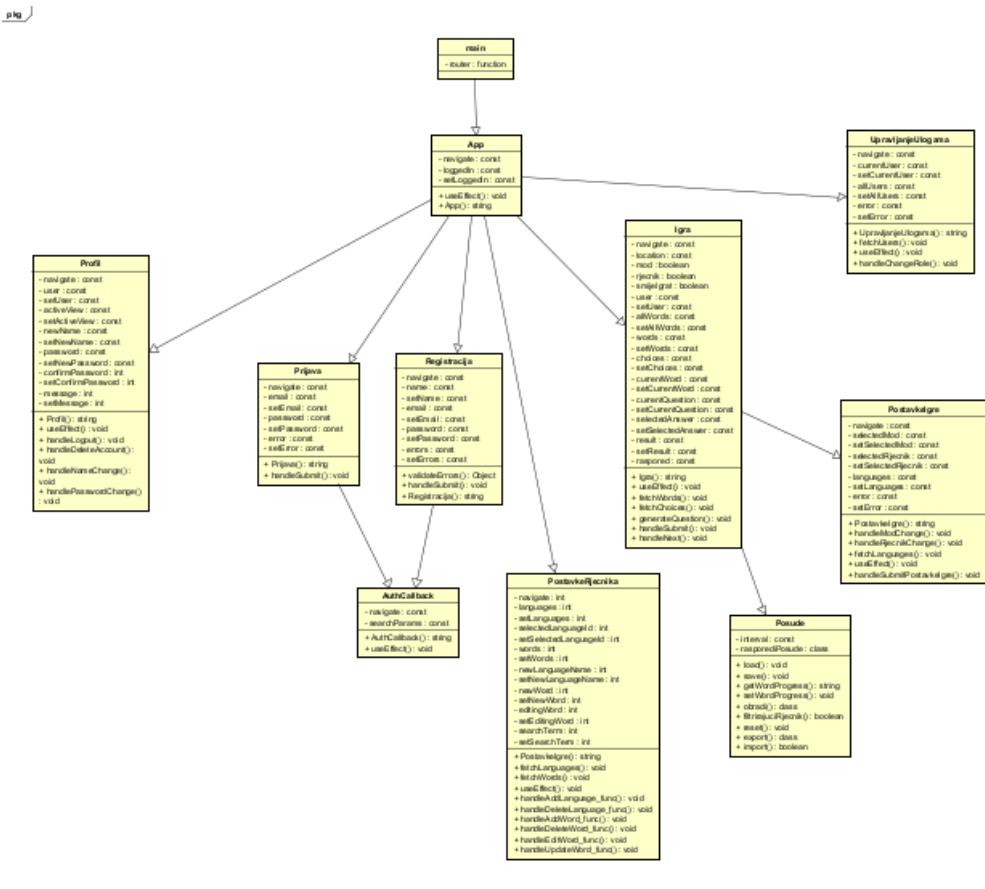
Atribut	Tip podatka	Opis varijable
user_email	Text	referenca na Users(email), primarni ključ
selected_language_id	Int	referenca na Languages(language_id), primarni ključ
selected_mod	Text	trenutno odabrani mod

## Dijagram baze podataka



Baza podataka se sastoji od pet tablica. Users tablica koja pohranjuje podatke korisnika, Languages tablica koja pohranjuje jezike dostupne za učenje, tablicu Words koja sadrži riječi pojedinog jezika, learning\_progress tablica koja služi za pohranjivanje napretka učenja pojedinog korisnika te user\_settings tablica koja pohranjuje odabrani mod korisnika. Jedan jezik može imati nula ili više riječi, svaka riječ nužno pripada točno jednom jeziku te je sukladno tome postavljen constraint koji brisanjem nekog jezika iz tablice Languages briše sve riječi koje pripadaju tom jeziku iz tablice Words. Kasnije dodatne izmjene baze podatke su moguće. Tablice learning\_progress i user\_settings su obe ovisne o korisniku te stoga s brisanjem korisnika (npr. odabir zatvaranja računa) i same se brišu.

# Dijagram razreda



## Opis dijagrama razreda

Aplikacija je sastavljena od više međusobno povezanih komponenti koje zajedno čine sustav za učenje jezika kroz igru. Glavna komponenta sustava je `App`, koja predstavlja početnu točku aplikacije. Ona upravlja prikazom početnog sučelja i ovisno o stanju prijave korisnika omogućuje pristup ostalim dijelovima sustava poput profila i same igre.

Proces prijave i autentifikacije korisnika odvija se kroz komponente `Prijava`, `Registracija` i `AuthCallback`. `Registracija` omogućuje stvaranje novog korisničkog računa, dok se prijava koristi za pristup postojećem profilu. Komponenta `AuthCallback` zadužena je za obradu povratnih podataka nakon autentifikacije, dekodiranje tokena i spremanje korisničkih podataka, čime se omogućuje nastavak rada u aplikaciji bez ponovne prijave.

Nakon što se korisnik prijavi, on ima pristup komponenti `Profil`, koja služi kao središnje mjesto za upravljanje korisničkim računom. U okviru profila korisnik može mijenjati osobne podatke, ažurirati lozinku, pregledavati vlastiti napredak ili se odjaviti iz sustava. Iz profila je moguće pristupiti i dodatnim administrativnim funkcijama ako korisnik ima takve ovlasti, putem komponente `UpravljanjeUlogama`, gdje se mogu mijenjati korisničke uloge i prava pristupa.

Središnji dio aplikacije čini komponenta `Igra`, u kojoj se odvija proces učenja jezika kroz interaktivna pitanja i odgovore. `Igra` koristi klasu `rasporediPosude`, koja pamti napredak korisnika i raspoređuje riječi prema razini poznavanja. Tako način sustav prilagođava težinu i redoslijed riječi, potičući učinkovitije učenje.

Prije nego što započne s igrom, korisnik može prilagoditi način rada u komponenti `PostavkeRjecnika`, gdje odabire jezik, rječnik i mod igre. Sustav potom na temelju tih postavki generira odgovarajuća pitanja.

Za korisnike (admine) koji žele uređivati sadržaj rječnika predviđena je komponenta `PostavkeRjecnika`, koja omogućuje dodavanje novih jezika, unos i uređivanje riječi te brisanje postojećih pojmovova. Tako aplikacija postaje prilagodljiva potrebama korisnika i omogućuje proširenje dostupnih materijala.

Sve komponente međusobno su povezane tako da čine logičan i funkcionalan tijek rada: od registracije i prijave, preko odabira postavki igre i učenja kroz igru, pa sve do praćenja napretka i administrativnog upravljanja sustavom.

### App.jsx

Atribut / Metoda	Tip	Pristup	Opis
loggedIn	boolean	private	Stanje koje označava je li korisnik prijavljen.
navigate	function	public	Funkcija za navigaciju između ruta.
useEffect()	void	private	Provjerava je li korisnik prijavljen pri učitavanju aplikacije

### AuthCallback.jsx

Atribut / Metoda	Tip	Pristup	Opis
navigate	function	public	Funkcija za navigaciju
searchParams	object	public	Parametri iz URL-a, uključujući token
useEffect()	void	private	Obrađuje token i spremi korisnika u localStorage

### Igra.jsx

Atribut / Metoda	Tip	Pristup	Opis
user	object	private	Trenutačni korisnik
allWords	array	private	Sve riječi dohvaćene iz API-ja
words	array	private	Filtrirane riječi prema rasporedu
choices	array	private	Odgovori za trenutačno pitanje
currentWord	object	private	Trenutačna riječ koja se prikazuje
currentQuestion	string	private	Tekst pitanja
selectedAnswer	string	private	Odabrani odgovor korisnika
result	string	private	Rezultat odgovora
raspored	rasporediPosude	private	Instanca klase rasporediPosude za filtriranje riječi
fetchWords(langId)	void	private	Dohvaća riječi s API-ja
fetchChoices(langId, wordId)	void	private	Dohvaća moguće odgovore s API-ja
generateQuestion()	void	private	Generira novo pitanje
handleSubmit()	void	public	Provjerava odabrani odgovor i ažurira napredak
handleNext()	void	public	Postavlja novo pitanje

### Postavkelgre.jsx

Atribut / Metoda	Tip	Pristup	Opis
selectedMod	string	private	Odabrani mod igre
selectedRjecnik	string	private	Odabrani rječnik za igru
languages	array	private	Lista dostupnih jezika

Atribut / Metoda	Tip	Pristup	Opis
error	string	private	Poruka o grešci
navigate	function	public	Funkcija za navigaciju
handleModChange(event)	void	public	Postavlja odabrani mod igre
handleRjecnikChange(event)	void	public	Postavlja odabrani rječnik
fetchLanguages()	void	public	Dohvaća jezike s API-ja
handleSubmitPostavkelgre(e)	void	public	Validira i pokreće igru s odabranim postavkama

#### PostavkeRjecnika.jsx

Atribut / Metoda	Tip	Pristup	Opis
languages	array	private	Lista dostupnih jezika
selectedLangId	string	private	Odabrani jezik za uređivanje
words	array	private	Riječi odabranog jezika
newLanguageName	string	private	Ime novog jezika za dodavanje
newWord	object	private	Novi riječ za dodavanje
editingWord	object	private	Riječ koja se trenutačno uređuje
searchTerm	string	private	Tekst za filtriranje riječi
fetchLanguages()	void	private	Dohvaća jezike
fetchWords(langId)	void	private	Dohvaća riječi za odabrani jezik
handleAddLanguage_func()	void	public	Dodaje novi jezik
handleDeleteLanguage_func(langId)	void	public	Briše jezik i povezane riječi
handleAddWord_func()	void	public	Dodaje novu riječ
handleDeleteWord_func(wordId)	void	public	Briše riječ
handleEditWord_func(word)	void	public	Pokreće uređivanje riječi
handleUpdateWord_func()	void	public	Spremanje promjena uređene riječi

#### Posude.jsx (klasa: rasporediPosude) Klasa: rasporediPosude

Atribut / Metoda	Tip	Pristup	Opis
langId	string	public	Identifikator jezika za koji se prati napredak
userEmail	string	public	Email korisnika kojem pripada napredak
storageKey	string	private	Ključ za pohranu podataka u localStorage
wordProgress	object	private	Objekt koji sadrži napredak pojedine riječi
constructor(langId, userEmail, options={})	-	public	Inicijalizira instancu klase
load()	void	private	Učitava podatke o napretku iz localStorage
save()	void	private	Sprema podatke o napretku u localStorage

Atribut / Metoda	Tip	Pristup	Opis
getWordProgress(wordId)	object	public	Vraća napredak za određenu riječ
setWordProgress(wordId, progress)	void	public	Postavlja napredak za određenu riječ
obradi(wordId, tocanOdgovor)	object	public	Ažurira razinu riječi ovisno o točnosti odgovora
filtrirajRijeci(words)	array	public	Vraća riječi koje su spremne za prikaz prema rasporedu
reset()	void	public	Resetira sav napredak u učenju
export()	object	public	Vraća podatke o napretku u formatu za izvoz
import(exportedData)	boolean	public	Uvozi prethodno izvezene podatke

Prijava.jsx

Atribut / Metoda	Tip	Pristup	Opis
email	string	private	Email korisnika
password	string	private	Lozinka korisnika
error	string	private	Poruka o grešci prijave
navigate	function	public	Funkcija za navigaciju
handleSubmit(e)	void	public	Provjerava prijavu i sprema korisnika u localStorage

Profil.jsx

Atribut / Metoda	Tip	Pristup	Opis
user	object	private	Trenutačni korisnik
activeView	string	private	Odabrani prikaz u profilu
newName	string	private	Novo ime korisnika
newPassword	string	private	Nova lozinka
confirmPassword	string	private	Potvrda nove lozinke
message	string	private	Poruka o statusu promjene podataka
navigate	function	public	Funkcija za navigaciju
handleLogout()	void	public	Odjavljuje korisnika
handleDeleteAccount()	void	public	Briše korisnički račun
handleNameChange()	void	public	Mjenja ime korisnika
handlePasswordChange()	void	public	Mjenja lozinku korisnika

Registracija.jsx

Atribut / Metoda	Tip	Pristup	Opis
name	string	private	Ime novog korisnika
email	string	private	Email novog korisnika

Atribut / Metoda	Tip	Pristup	Opis
password	string	private	Lozinka novog korisnika
errors	object	private	Objekt grešaka pri registraciji
navigate	function	public	Funkcija za navigaciju
validateForm()	boolean	public	Provjerava ispravnost podataka
handleSubmit(e)	void	public	Obrađuje registraciju i pohranjuje podatke

### UpravljanjeUlogama.jsx

Atribut / Metoda	Tip	Pristup	Opis
currentUser	object	private	Trenutačni korisnik (admin)
allUsers	array	private	Lista svih korisnika
error	string	private	Poruka o grešci
navigate	function	public	Funkcija za navigaciju
fetchUsers(admin)	void	private	Dohvaća sve korisnike za prikaz
handleChangeRole(targetEmail, newRole)	void	public	Mijenja ulogu korisnika

## Dinamičko ponašanje aplikacije

Dinamičko ponašanje aplikacije odnosi se na način na koji objekti u sustavu evoluiraju kroz vrijeme, uključujući prijelaze između različitih stanja. To uključuje aktivnosti, događaje, odluke i interakcije unutar aplikacije. UML dijagrami stanja omogućuju vizualizaciju tih promjena i olakšavaju razumijevanje dinamike sustava.

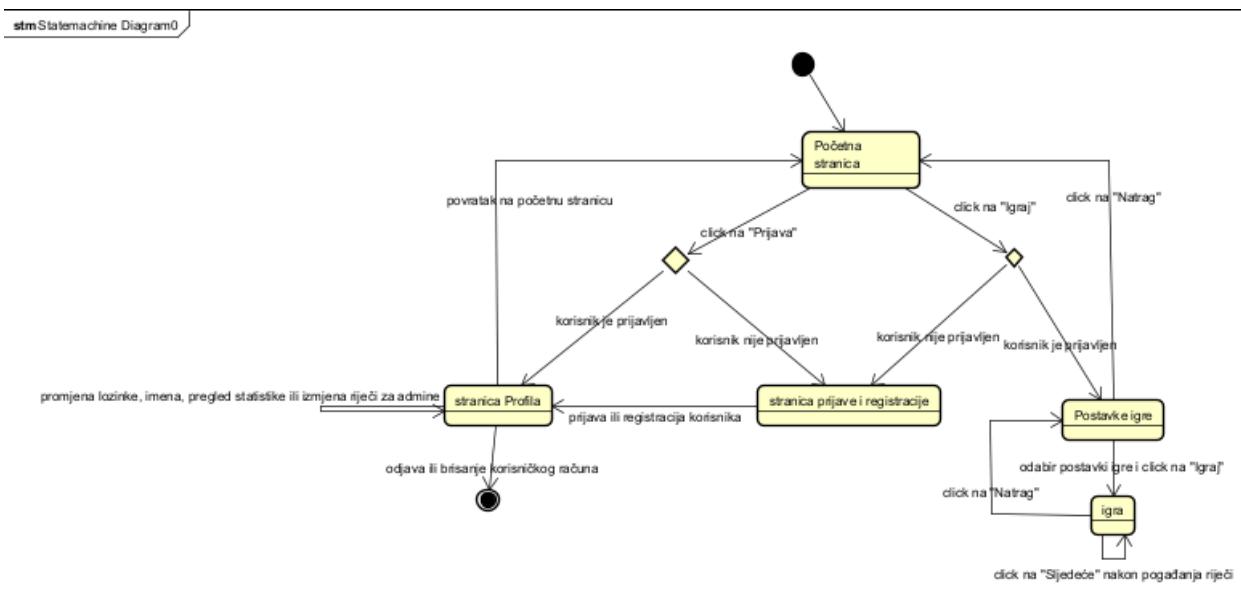
Razumijevanje promjena stanja neophodno je za pravilno funkcioniranje aplikacije jer pruža uvid u interakcije među objektima, komponentama i korisnicima tijekom rada sustava. Korištenjem UML dijagrama stanja i aktivnosti moguće je vizualizirati prijelaze i stanja objekata, identificirati potencijalne probleme, osigurati točnu implementaciju te poboljšati komunikaciju među članovima tima.

### Zadatak:

- Dokumentirajte dva dinamička dijagrama koji prikazuju značajne ili neke složene procese u aplikaciji. Registracija i prijava korisnika nisu ključni procesi u ovom kontekstu, stoga se usmjerite na specifične, značajne procese.

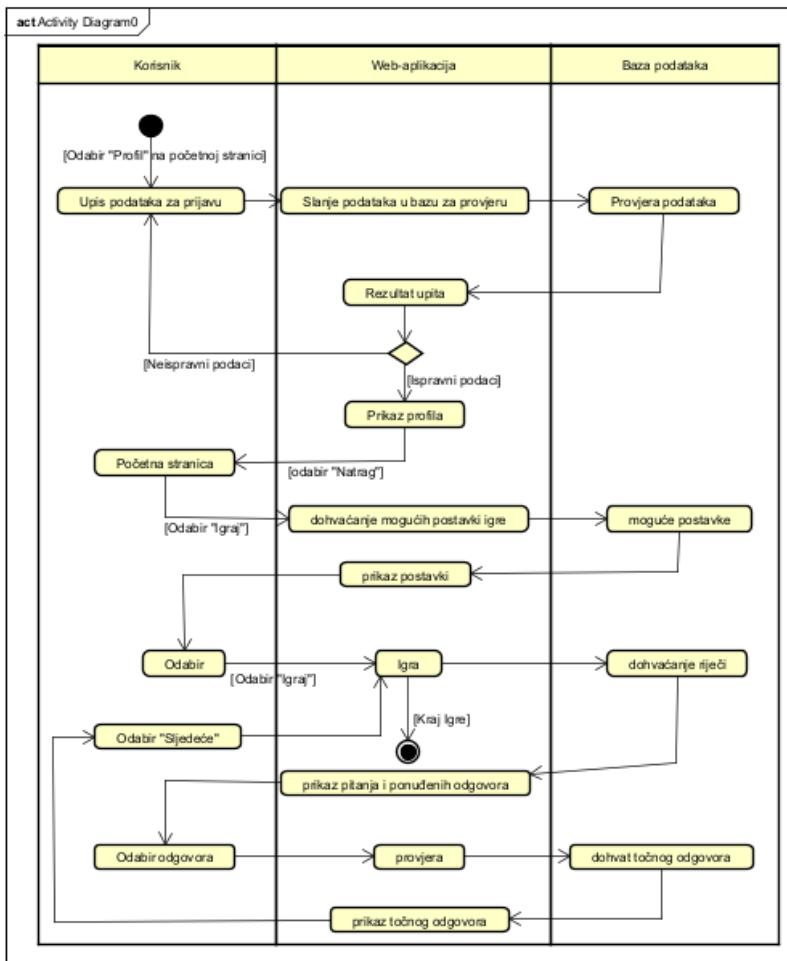
## UML dijagrami stanja

Dijagram stanja prikazuje općenito koordiniranje po web-stranici npr. gdje nas vodi koji navigacijski gumb te kako možemo navigirati do određene stranice cjelokupne aplikacije.



## UML dijagrami aktivnosti

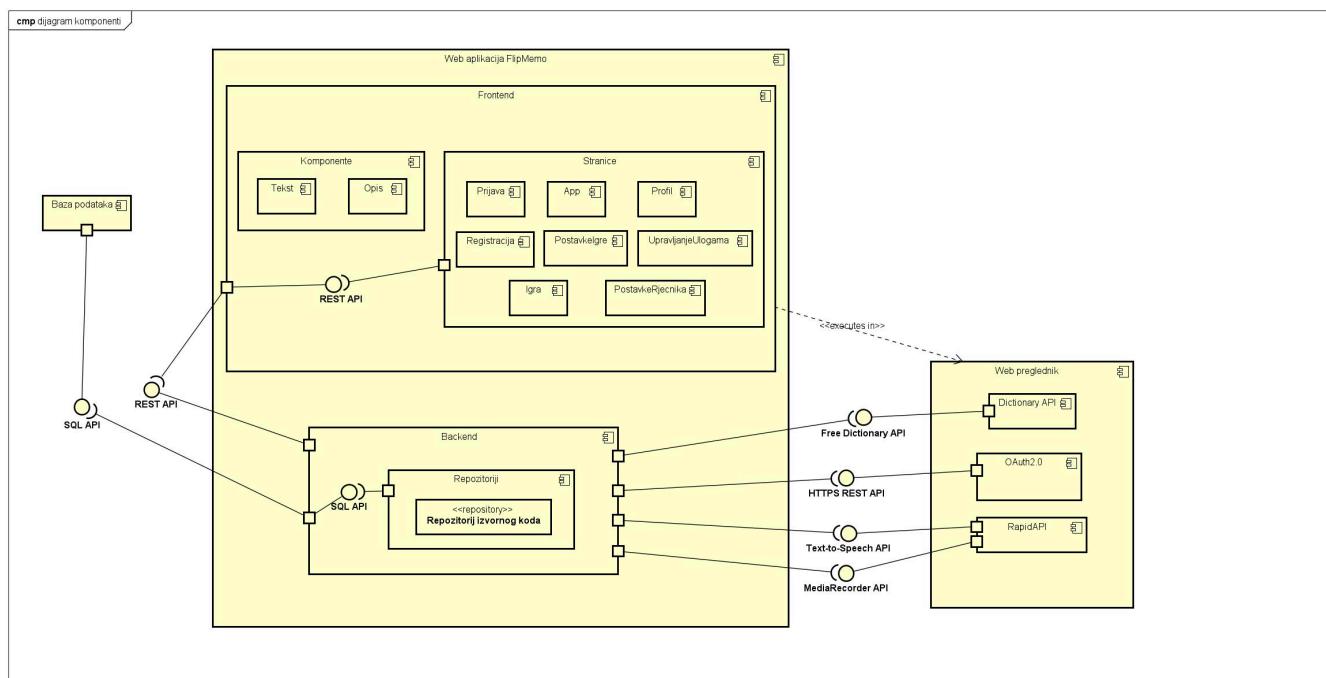
Dijagram aktivnosti prikazuje tijek izvršavanja određenog procesa. Osim za razumijevanje toka podataka unutar aplikacije, koristi se za analizu poslovnih procesa. Sljedeći dijagram prikazuje dijagram aktivnosti pri općenitom igranju igre na web-aplikaciji FlipMemo.



# 5. Arhitektura komponenata i razmještaja

## Dijagram komponenata

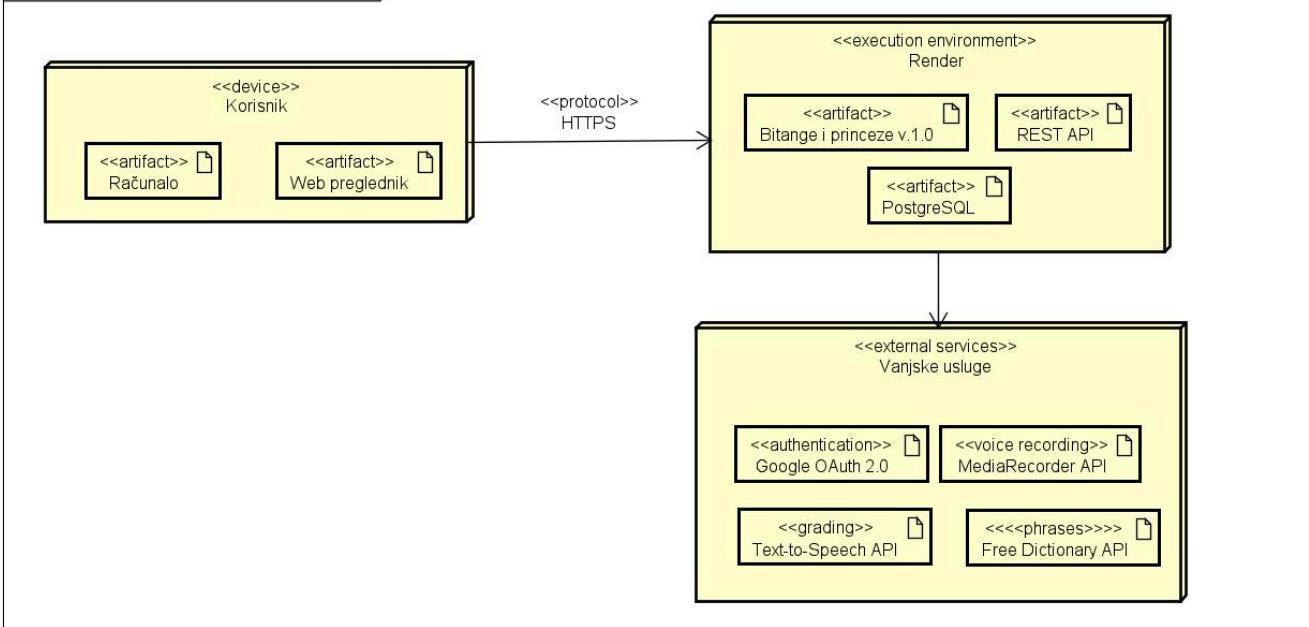
Web aplikacija FlipMemo koristi REST API kao pomoć u komunikaciji između frontenda i backenda. Backend komunicira s PostgreSQL bazom podataka s pomoću SQL API-ja. FlipMemo koristi OAuth2.0 za autentifikaciju korisnika prilikom registracije/prijave, a to se odvija s pomoću HTTPS sigurnosnog protokola. Kako bi se rječnik popunio frazama engleskog jezika, aplikacija koristi Free Dictionary API. Za pretvorbu teksta u govor, FlipMemo koristi Text-to-Speech API, a za snimanje izgovora MediaRecorder API.



## Dijagram razmještaja

U nastavku se nalazi prikaz dijagrama razmještaja na razini specifikacije aplikacije FlipMemo.

**pkg** dijagram razmještaja na razini specifikacije



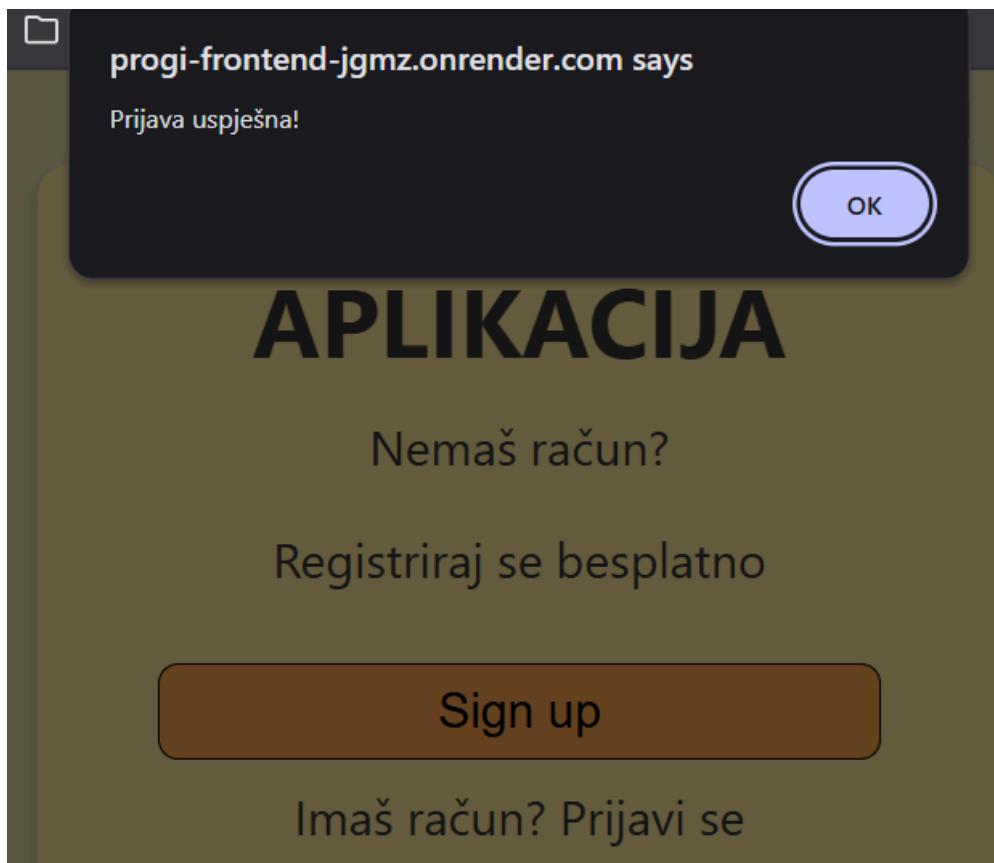
## 6. Ispitivanje programskog rješenja

### Ispitivanje komponenti

Cilj ispitivanja komponenti bio je izolirano provjeriti ispravnost rada pojedinih funkcionalnih cjelina aplikacije FlipMemo. Svaka komponenta testirana je neovisno o ostatku sustava kako bi se osiguralo da pravilno reagira na očekivane i neočekivane ulazne podatke.

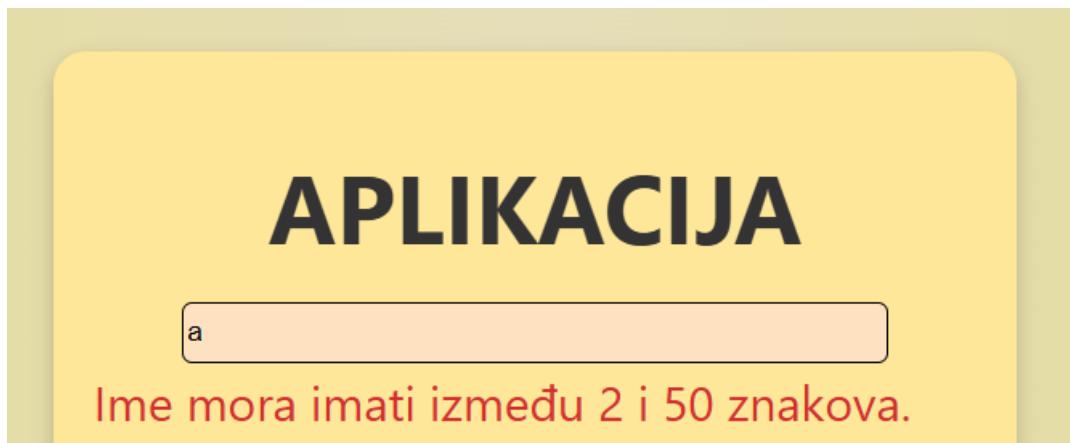
#### Ispitni slučaj 1 – Redovni slučaj: prijava korisnika

Opis funkcionalnosti: Prijava korisnika putem obrasca za prijavu. Ulagni podaci: Ispravna email adresa i pripadajuća lozinka. Očekivani rezultat: Korisnik se uspješno prijavljuje i preusmjerava na stranicu profila. Dobiveni rezultat: Funkcionalnost radi ispravno – test uspješan. Postupak ispitivanja: Komponenta Prijava testirana je simulacijom unosa valjanih korisničkih podataka i provjerom navigacije.



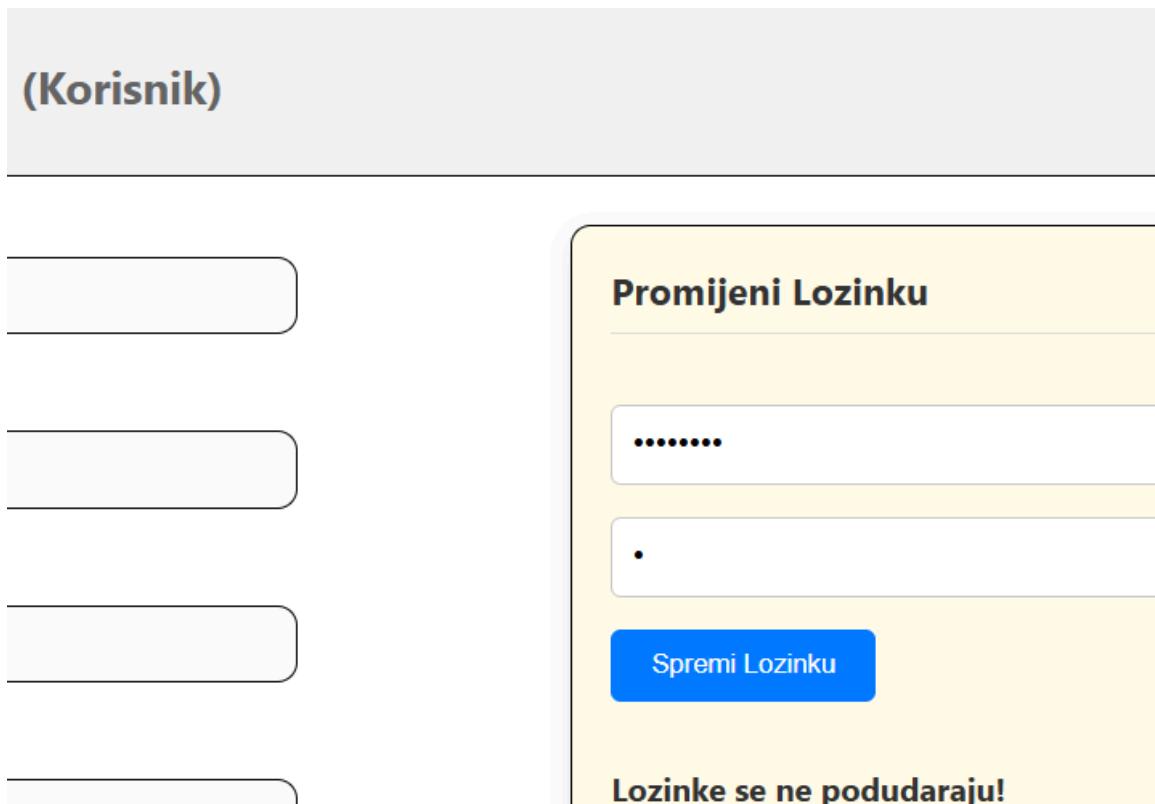
## Ispitni slučaj 2 – Rubni uvjet: registracija s prekratkim imenom

Opis funkcionalnosti: Validacija korisničkih podataka pri registraciji. Ulazni podaci: Ime kraće od minimalno dopuštene duljine. Očekivani rezultat: Prikaz poruke o pogrešci i onemogućavanje registracije. Dobiveni rezultat: Sustav ispravno odbija unos – test uspješan. Postupak ispitivanja: Komponenta Registracija testirana je s graničnim vrijednostima ulaza.



## Ispitni slučaj 3 – Izazivanje pogreške: promjena lozinke bez potvrde

Opis funkcionalnosti: Promjena lozinke unutar korisničkog profila. Ulazni podaci: Nova lozinka i različita potvrda lozinke. Očekivani rezultat: Prikaz poruke o neusklađenosti lozinki. Dobiveni rezultat: Sustav pravilno detektira pogrešku – test uspješan. Postupak ispitivanja: Testirana je komponenta Profil uz namjerno neispravne podatke.

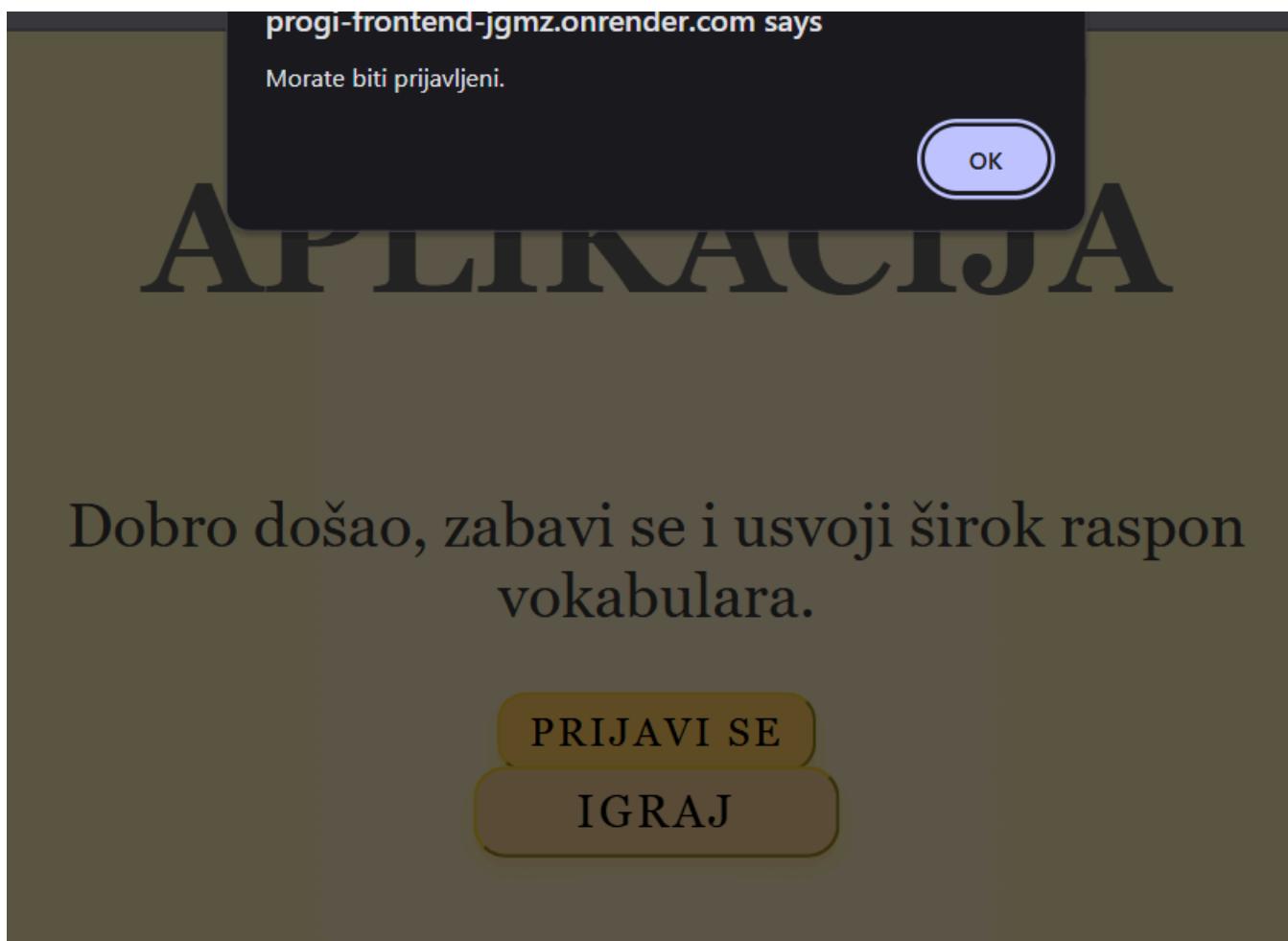


## Ispitni slučaj 4 – Redovni slučaj: filtriranje riječi u igri

Opis funkcionalnosti: Odabir riječi za prikaz u igri na temelju napretka korisnika. Ulazni podaci: Skup riječi i spremljeni napredak korisnika. Očekivani rezultat: Prikazuju se samo riječi koje su spremne za ponavljanje. Dobiveni rezultat: Filtriranje radi ispravno – test uspješan. Postupak ispitivanja: Testirana je klasa rasporediPosude izolirano od sučelja.

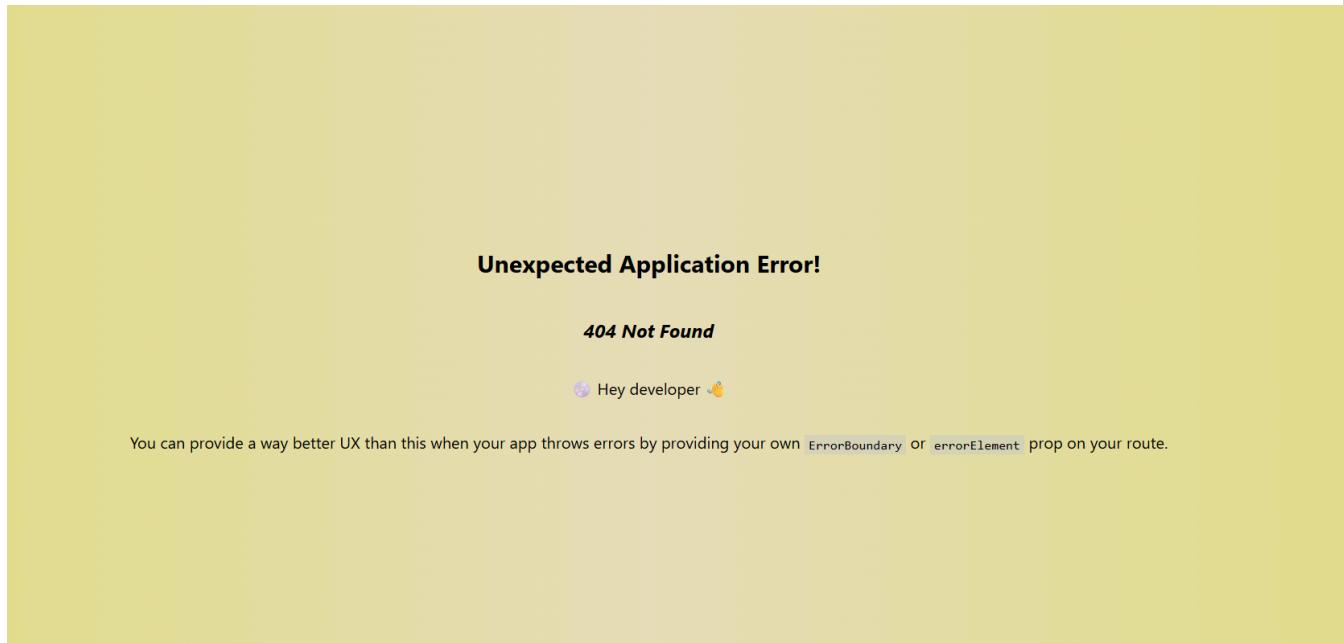
## Ispitni slučaj 5 – Izazivanje pogreške: pokušaj igranja bez prijave

Opis funkcionalnosti: Korisnik pokušava igrati igru bez prijave Ulazni podaci: Korisnik pokušava pristupiti igri putem navigacijskog gumba Očekivani rezultat: Prikaz poruke da je potrebna prijava te preusmjeravanje na stranicu za prijavu Dobiveni rezultat: Sustav se ponaša ispravno – test uspješan. Postupak ispitivanja: Testirana je funkcionalnost navigacijskog gumba Igraj u slučaju kada korisnik nije prijavljen



## Ispitni slučaj 6 – Nepostojeća funkcionalnost: pokušaj pristupa neimplementiranoj ruti

Opis funkcionalnosti: Reakcija sustava na nepostojeću rutu Ulazni podaci: Neispravna URL adresa. Očekivani rezultat: Obrada pogreške bez rušenja aplikacije. Dobiveni rezultat: Sustav pravilno hvata pogrešku – test uspješan. Postupak ispitivanja: Testiran je mehanizam obrade pogrešaka prilikom dohvatanja podataka.



## \*\*Ispitivanje sustava \*\*

Cilj ispitivanja sustava je testiranje ponašanja cijelog sustava u uvjetima stvarnog korištenja, uz posebnu pažnju na međusobnu povezanost svih komponenti. Ispitivanje treba obuhvatiti sve aspekte sustava i njegovu interakciju s korisnicima.

### Ispitni slučaj 1 – Redovni slučaj: prijava korisnika s početne stranice

Opis funkcionalnosti: Navigiranje korisnika do obrasca za prijavu i prijava korisnika putem obrasca za prijavu. Ulagani podaci: Koristi se user korisnički račun putem SeleniumIDE-a. Očekivani rezultat: SeleniumIDE ispravno se prijavljuje kao korisnik. Dobiveni rezultat: Funkcionalnost radi ispravno – test uspješan. Postupak ispitivanja: Navigiranje s početne stranice te prijava korisnika ispravno funkcioniраju.

The screenshot shows the Selenium IDE interface with a test log. The log consists of 9 numbered steps:

Step	Command	Target	Value
1	✓ open	/	
2	✓ set window size	1422x816	
3	✓ click	css=.profilbutton:nth-child(3)	
4	✓ click	css=.podatci:nth-child(6)	
5	✓ type	css=.podatci:nth-child(6)	user@gmail.com
6	✓ click	css=.podatci:nth-child(7)	
7	✓ type	css=.podatci:nth-child(7)	user
8	✓ click	css=.submitbutton	
9	✓ assert alert	Prijava uspješna!	

	Log	Reference	
Running 'login'			
1. open on / OK			17:52:56
2. setWindowSize on 1422x816 OK			17:52:56
3. click on css=.profilbutton:nth-child(3) OK			17:52:57
4. click on css=.podatci:nth-child(6) OK			17:52:58
5. type on css=.podatci:nth-child(6) with value user@gmail.com OK			17:53:00
6. click on css=.podatci:nth-child(7) OK			17:53:01
7. type on css=.podatci:nth-child(7) with value user OK			17:53:02
8. click on css=.submitbutton OK			17:53:03
9. assertAlert on Prijava uspješna! OK			17:53:04
'login' completed successfully			17:53:04

## Ispitni slučaj 2 - Redovni slučaj: ulaz prijavljenog korisnika u igru te povratak na početnu stranicu

Opis funkcionalnosti: Navigiranje korisnika do igre te navigiranje natrag do početne stranice. Ulazni podaci: SeleniumIDE sa zadanim clickovima. Očekivani rezultat: SeleniumIDE bi trebao ispravno doći do igre sa početne stranice te se vratiti natrag na istu. Dobiveni rezultat: Funkcionalnost radi ispravno - test uspješan. Postupak ispitivanja: SeleniumIDE ispravno navigira od početne stranice do igre i natrag.

	Log	Reference	
1. open on / OK			18:07:03
2. setWindowSize on 1423x816 OK			18:07:03
3. click on css=.profilbutton:nth-child(4) OK			18:07:03
4. click on css=.dropdown:nth-child(3) OK			18:07:04
5. select on css=.dropdown:nth-child(3) with value label=Engleski OK			18:07:04
6. click on css=.containerPostavkegre OK			18:07:04
7. click on css=.dropdown:nth-child(5) OK			18:07:04
8. select on css=.dropdown:nth-child(5) with value label=Engleska riječ → Hrvatski prijevod OK			18:07:05
9. click on css=.submitbutton:nth-child(1) OK			18:07:05
10. mouseOver on css=.third-color:nth-child(2) OK			18:07:05
11. click on css=.back OK			18:07:05
12. click on css=.submitbutton:nth-child(2) OK			18:07:06
'navigacija igra' completed successfully			18:07:06

## Ispitni slučaj 3 - Poziv nepostojećih funkcionalnosti: pristup nepostojećoj ruti

Opis funkcionalnosti: Korisnik pokušava pristupiti nepostojećoj ruti aplikacije putem URL-a. Ulazni podaci: SeleniumIDE koji upisuje krivu rutu u URL. Očekivani rezultat: Obrada pogreške bez rušenja aplikacije. Dobiveni rezultat: Sustav očekivano prihvata grešku. Postupak ispitivanja: SeleniumIDE u URL dodaje rutu koja je nepostojeća ili kojoj nema pristupa.

Running 'fakeroute'

1. open on /admin OK

'fakeroute' completed successfully

## Ispitni slučaj 4 - Rubni uvjet: Neispravan pokušaj promjene imena

Opis funkcionalnosti: Korisnik navigira do profila s početne stranice te tamo pokušava promijeniti ime. Ulazni podaci: SeleniumIDE sa novim zadanim imenom. Očekivani rezultat: Greška u promjeni zbog neispravnog imena. Dobiveni rezultat: Sustav očekivano reagira. Postupak ispitivanja: SeleniumIDE ispravno dolazi do stranice profila te u njoj neuspješno pokušava promijeniti ime korisničkog računa.

---

Running 'badNamechange'

1. open on / **OK**
2. setWindowSize on 1420x816 **OK**
3. click on css=.profilbutton:nth-child(3) **OK**
4. click on css=.kategorija:nth-child(3) **OK**
5. click on css=.dno **OK**
6. type on css=input with value a **OK**
7. click on css=button:nth-child(3) **OK**

**'badNamechange' completed successfully**

# 7. Tehnologije za implementaciju aplikacije

---

## Korištene tehnologije i alati

---

### 1. Programski jezici

- JavaScript ES2025/ES2026

### 2. Radni okviri i biblioteke

- React 19.1.1

### 3. Baza podataka

- PostgreSQL 18

### 4. Razvojni alati

- Eclipse IDE 2025-12 R

- VS Code 1.108

- Astah UML 11.0

- Postman 11.31.1

### 5. Alati za ispitivanje

### 6. Alati za razmještaj

- Docker

- Render

Za razvoj klijentskog dijela aplikacije korišten je JavaScript zbog dobre podrške u preglednicima i široke primjene u razvoju web stranica. Korisničko sučelje izrađeno je pomoću Reacta, JavaScript biblioteke koja omogućuje izradu ponovno iskoristivih komponenti i jednostavno održavanje aplikacije.

Podaci se pohranjuju u PostgreSQL bazi podataka, odabranoj zbog pouzdanosti, sigurnosti i podrške za relacijski model podataka.

Za razvoj i testiranje korišteni su alati VS Code, Eclipse, Astah UML i Postman, dok je za automatizirano testiranje korisničkog sučelja korišten Selenium.

Aplikacija je pripremljena za razmještaj pomoću Dockera, čime je osigurano konzistentno izvođenje u različitim okruženjima te je postavljena na cloud platformu Render radi jednostavnog deploymenta i održavanja.

# 8. Upute za puštanje u pogon

## Upute za puštanje u pogon

### 1. Instalacija, postavljanje i pokretanje

Preuzeti node.js (verzija 16) ako nemate: <https://nodejs.org/en/download/>

Postavljanje Docker Desktop Provjeriti imate li wsl instaliran (u terminalu upišite: wsl) Ako nemate napišite u terminal:  
wsl --install

Zatim je potrebno instalirati Docker Desktop (verzija po volji s preporukom na najnoviju):

<https://docs.docker.com/desktop/setup/install/windows-install/>

Bitno je da je opcija "Use WSL 2" označena. Bit će potrebno restartati računalo. Prije sljedećeg koraka otvoriti aplikaciju Docker Desktop.

U terminalu otvorite folder sa GitHub-a my-react-app (cd ...\\my-react-app) i upišite ove naredbe:

```
npm install  
docker-compose up -d  
docker exec -it progi-db psql -U postgres progi
```

```
CREATE TABLE users( name text not null, email text not null primary key, password text, role smallint default 0 not null,  
google_id text unique ); INSERT INTO users(name, email, password, role) VALUES('koradmin', 'koradmin@gmail.com',  
'koradmin', 2); INSERT INTO users(name, email, password, role) VALUES('admin', 'admin@gmail.com', 'admin', 1);  
INSERT INTO users(name, email, password, role) VALUES('user', 'user@gmail.com', 'user', 0);
```

```
CREATE TABLE languages ( language_id SERIAL PRIMARY KEY, language_name TEXT NOT NULL UNIQUE );
```

```
CREATE TABLE words ( word_id SERIAL PRIMARY KEY, word_text TEXT NOT NULL, language_id INT NOT NULL,  
translation_to_croatian TEXT, phrases TEXT[], pronunciation BYTEA,);
```

```
CREATE TABLE learning_progress ( user_email TEXT NOT NULL REFERENCES users(email) ON DELETE CASCADE,  
language_id INT NOT NULL REFERENCES languages(language_id) ON DELETE CASCADE, word_id INT NOT NULL  
REFERENCES words(word_id) ON DELETE CASCADE, razina SMALLINT NOT NULL DEFAULT 0, sljedeci_datum  
TIMESTAMPTZ NOT NULL DEFAULT NOW(), zadnji_pokusaj TIMESTAMPTZ, točni INT NOT NULL DEFAULT 0, netočni  
INT NOT NULL DEFAULT 0, PRIMARY KEY (user_email, language_id, word_id) );
```

```
CREATE TABLE user_settings ( user_email TEXT PRIMARY KEY REFERENCES users(email) ON DELETE CASCADE,  
selected_language_id INT REFERENCES languages(language_id), selected_mod TEXT NOT NULL DEFAULT 'mod1' );
```

```
INSERT INTO languages (language_name) VALUES ('Engleski'); INSERT INTO languages (language_name) VALUES  
( 'Njemački' ); INSERT INTO languages (language_name) VALUES ( 'Španjolski' );
```

```
npm run dev
```

Zatim otvorite link iz terminala

## 2. OAuth

OAuth Kreirajte u my-react-app textualnu datoteku ".env" i zalijepite u nju sljedeći tekst:

```
GOOGLE_CLIENT_ID= GOOGLE_CLIENT_SECRET= SESSION_SECRET=a_random_secret_string_for_sessions  
JWT_SECRET=another_random_secret_for_jwt
```

```
DB_USER=postgres DB_PASSWORD=bazepodataka DB_HOST=localhost DB_PORT=5433 DB_DATABASE=progi
```

Ako ste kreirali bazu iz trenutne verzije ovih uputa preskočite ove korake i idite na 6.1 Potrebno je malo prepraviti bazu. U terminalu upišite: docker exec -it progi-db psql -U postgres progi Zatim zalijepite ove naredbe:

```
ALTER TABLE users ALTER COLUMN password DROP NOT NULL;
```

```
ALTER TABLE users ADD COLUMN google_id TEXT UNIQUE;
```

6.1. Idite na Google Cloud Console (<https://www.google.com/url?sa=E&q=https%3A%2F%2Fconsole.cloud.google.com%2F>). Prijavite se sa svojim Google računom.

6.2. Na vrhu stranice, kliknite na padajući izbornik za projekte (pored "Google Cloud" loga) i odaberite "New Project". Nazovite ga npr. "React Aplikacija".

6.3. Nakon što je projekt stvoren, provjerite da je odabran.

6.4. U lijevom navigacijskom meniju (ili preko tražilice na vrhu), pronađite "APIs & Services" i kliknite na "Credentials".

6.5. Kliknite na veliki plavi gumb "+ CREATE CREDENTIALS" i odaberite "OAuth client ID".

6.6. Prvi put ćete morati konfigurirati "OAuth consent screen" (ekran za pristanak). - Upišite App name (npr. "Moja Aplikacija"). - Upišite User support email (vaš email). - Odaberite "External" i kliknite "Create". - Scrollajte dolje i upišite Developer contact information (opet vaš email). - Kliknite "Save and Continue" kroz sve korake (Scopes, Test Users). Ne trebate ništa dodavati za sada. Na kraju kliknite "Back to Dashboard".

6.7. Vratite se na "Credentials" tab, ponovno kliknite "+ CREATE CREDENTIALS" -> "OAuth client ID".

6.8. Sada popunite formu: - Application type: Odaberite "Web application". - Name: Može ostati "Web client 1". - Authorized redirect URLs (kopiraj točno ovo):

<http://localhost:3001/api/auth/google/callback>

- Authorized JavaScript origins (kopiraj točno ovo): <http://localhost:5173>

6.9. Kliknite "Create". Pojavit će se prozorčić s vašim ključevima! Kopirajte vrijednosti za "Your Client ID" i "Your Client Secret". To su vrijednosti koje trebate zalijepiti u svoju .env datoteku.

7.0. Stavite u .env datoteku ove ključeve, u slučaju da ne rade, probajte sami doći do tih ključeva na ovim linkovima: voicerss\_api: <https://rapidapi.com/voicerss/api/text-to-speech-1> pronunciation\_api\_key: <https://rapidapi.com/language-confidence-language-confidence-default/api/scripted-speech-assessment1>

```
TTS_RAPIDAPI_KEY=6448e15eefmsh981de6c7f7dbf44p1bc01bjsn36347a821c0f
```

```
VOICERSS_API_KEY=59359c880552423aa85703867d74a14f
```

```
PRONUNCIATION_API_KEY=6448e15eefmsh981de6c7f7dbf44p1bc01bjsn36347a821c0f
```

## 3. Upute za administratore

Administratori imaju pristup početnim podacima za prijavu:

Korijenski administrator

email: [koradmin@gmail.com](mailto:koradmin@gmail.com) password: koradmin

Administrator

email: [admin@gmail.com](mailto:admin@gmail.com) password: admin

Administrator ima dužnost redovitog arhiviranja baze podataka, mogućnost pregleda logova stranice te ažuriranje aplikacije

Smjernice za administratore aplikacije nakon puštanja u pogon:

## Opis prisutpa aplikaciji na javnom poslužitelju

---

### Pristup aplikaciji

Stranici se preko internet pregleda pristupa putem linka:

<https://progi-frontend-jgmz.onrender.com/>

Uz takav pristup moguće je koristiti se unaprijed definiranim profilima koji su se koristili u testiranju.

Korijenski administrator:

email: [koradmin@gmail.com](mailto:koradmin@gmail.com) password: koradmin

Administrator:

email: [admin@gmail.com](mailto:admin@gmail.com) password: admin

Korisnik:

email: [user@gmail.com](mailto:user@gmail.com) password: user

Ovakav pristup stranici uz sebe ne veže nikakva posebna ograničenja. Korisnik i dalje ima pristup izradi vlastitog profila, korištenju istoga za prijavu čak i nakon napuštanja stranice te igranju igre.

## 9. Zaključak i budući rad

---

### Zaključak

Tijekom razvoja aplikacije FlipMemo, članovi tima Bitange i princeze kontinuirano su radili na razvoju aplikacije i dokumentacije. Unatoč tome, tijekom rada pojavilo se nekoliko problema. Najveći izazov bio je dovršetak postavljenih zadataka do navedenog roka. Nadalje, s pojedinim dijelovima razvoja aplikacije neki od članova tima susreli su se po prvi puta. Navedene poteškoće uklonjene su jasnom komunikacijom među članovima tima i pojedinačnim usavršavanjem u područjima u kojima je to bilo nužno za uspješnu realizaciju finalne aplikacije.

Mogućnost rada u timu i razvoja prvog konkretnog proizvoda pružilo je članovima tima uvid u stvarnost rada u IT industriji te važnost pridržavanja rokova. Osim toga, upoznali su se s razvojem kvalitetne projektne dokumentacije - vrlo važnim segmentom svakog ozbiljnog i uspješnog projekta.

Kako bi članovi tima Bitange i princeze usavršili svoje vještine u području Programskog inženjerstva, predlaže se nastavak suradnje u budućnosti, kao i razvoj sličnih projekata.

Svi funkcionalni i nefunkcionalni zahtjevi su uspješno implementirani u aplikaciji FlipMemo.

## A. Popis literature

---

### Korišteni izvori

1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>

### Korišteni alati

1. React, <https://react.dev>
2. The Unified Modeling Language, <https://www.uml-diagrams.org/>
3. Astah Community, <http://astah.net/editions/uml-new>
4. RapidAPI, <https://rapidapi.com/>
5. Render, <https://render.com/>
6. Node, <https://react.dev>
7. Free Dictionary API, <https://dictionaryapi.dev>

## B. Dnevnik promjena dokumentacije

### Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak	M. N.	20.10.2025
0.2	Dodani funkcijски и nefunkcijски zahtjevi	M.N.	21.10.2025
0.3	Dodan opis projektnog zadatka	N.S.	22.10.2025
0.4	Dodana specifikacija zahtjeva	M.N.	3.11.2025
0.5	Dodan opis arhitekture sustava	N.S.	13.11.2025
0.6	Dodani dijagrami arhitekture sustava	N.S.	13.11.2025
0.7	Dodana arhitektura komponenata i razmještaja	M.N.	22.1.2026
0.8	Dodan zaključak i upute za budući rad	M.N.	22.1.2026
0.9	Osvježena arhitektura sustava	N.S.	23.1.2026
10	Dodano ispitivanje programskog rješenja	N.S.	23.1.2026
11	Dodane upute za puštanje u pogon	N.S.	23.1.2026
12	Posljednje uređivanje dokumentacije	M.N.	23.1.2026

## C. Prikaz aktivnosti grupe

### Dnevnik sastajanja

#### 1. sastanak

- Datum: 15. listopada 2025.
- Prisustvovali: L. Vučen, M. Novak, Z. Stracenski, J. Volenec, M. Putarek, A. Janušić, N. Simunić
- Teme sastanka:
  - raspodjela poslova na frontend, backend i dokumentaciju
  - kreirana grupa za komunikaciju među članovima na WhatsAppu/Discordu
  - kreiran GitHub rezitorij

#### 2. sastanak

- Datum: 22. listopada 2025.
- Prisustvovali: L. Vučen, M. Novak, Z. Stracenski, J. Volenec, M. Putarek
- Teme sastanka:
  - učinjen osvrt na dosadašnji napredak
  - dogovoreni koraci za daljnji razvoj aplikacije

#### 3. sastanak

- Datum: 13. listopada 2025.
- Prisustvovali: L. Vučen, M. Novak, Z. Stracenski, J. Volenec, A. Janušić
- Teme sastanka:
  - detaljan pregled ostvarenih funkcionalnosti aplikacije te dogovor oko poboljšanja
  - detaljan pregled dokumentacije

#### 4. sastanak

- Datum: 11. prosinca 2025.
- Prisustvovali: L. Vučen, Z. Stracenski, J. Volenec, M. Putarek, A. Janušić, N. Simunić
- Teme sastanka:
  - evaluacija dosadašnjeg rada (prva predaja)
  - plan za poboljšanje baze podataka

#### 5. sastanak

- Datum: 15. siječnja 2026.
- Prisustvovali: L. Vučen, M. Novak, Z. Stracenski, J. Volenec, M. Putarek, A. Janušić, N. Simunić
- Teme sastanka:

- komentiranje skoro završene aplikacije
- finalna podjela posla i postavljanje krajnjih rokova

## Plan rada

Tjedan	Rad na projektu	Odgovorni članovi
2.	uspostava komunikacije među članovima	Z.S., L.V.
3.	stvaranje GitHub repozitorija	L.V., M.N.
4.	raspisivanje funkcijskih i nefunkcijskih zahtjeva	M.N.
	dokumentacija	M.N., N.S.
4.	razvoj prvih verzija frontenda	Z.S., L.V.
5.	specifikacija zahtjeva	M.N.
5.	razvoj prvih verzija frontenda	Z.S., L.V.
5.	razvoj prvih verzija backenda	J.V.
6.	arhitektura i dizajn sustava, baza podataka, dijagram razreda, dio dijagrama stanja	N.S.
6.	razvoj funkcijskih i nefunkcijskih zahtjeva backenda	J.V., M.P., A.J.
7.	objava na javni poslužitelj	J.V.
10.	razvoj funkcijskih i nefunkcijskih zahtjeva backenda	J.V., M.P., A.J.
11.	razvoj funkcijskih i nefunkcijskih zahtjeva backenda	J.V., M.P., A.J.
11.	api, razvoj zadnjih funkcionalnosti	J.V., M.P., A.J.
12.	razvoj konačne verzije frontenda	L.V.
12.	izrada prezentacije	L.V.
12.	konačna inačica dokumentacije	M.N.
12.	dijagram arhitekture sustava, ispitivanje rada sustava, upute za pokretanje	N.S.
12.	dijagram komponenti i razmještaja, korištene platforme, zaključak	M.N.

## Tablica aktivnosti

ZADATAK	L.V.	M.N.	Z.S.	J.V.	M.P.	A.J.	N.S.
Dokumentacija	1	35					30
Dizajn i frontend	25		20				
Backend i baza podataka			10	50	20	15	
Upravljanje projektom	3						
Opis projektnog zadatka							6
Funkcionalni zahtjevi		5					

ZADATAK	L.V.	M.N.	Z.S.	J.V.	M.P.	A.J.	N.S.
Opis pojedinih obrazaca		2					
Dijagram obrazaca		7					
Sekvencijski dijagrami		8					
Opis ostalih zahtjeva		1					
Arhitektura i dizajn sustava						10	
Baza podataka			2			4	
Dijagram razreda						10	
Dijagram stanja						1	
Dijagram aktivnosti						3	
Dijagram komponenti		8					
Korištene tehnologije i alati	5			5			
Ispitivanje programskog rješenja	1			2		4	
Dijagram razmještaja		8					
Upute za puštanje u pogon			2	2			
Dnevnik sastajanja		2					
Zaključak i budući rad		1					
Popis literature		2					
Izrada početne stranice	15		27				
Izrada baze podataka				1			
Spajanje s bazom podataka			1	4			
Izrada prezentacije							
Razvoj backend funkcionalnosti			5	15	16	11	
Postavljanje na javni poslužitelj			5	8			
Razvoj frontenda i dizajna stranice	15						

## Dijagram pregleda promjena

---

### **zvonimir-stracenski's Commits**



### **JanVolenc's Commits**



### **mihaelanovak's Commits**



### **lanavucen's Commits**



## MislavPutarek's Commits



## Andrija-Janusic's Commits



## nsimu-bit's Commits



# Kjučni izazovi i rješenja

## Glavni izazovi

- kašnjenje s realizacijom zadataka
- nedostatak adekvatnih znanja u razvoju pojedinih dijelova aplikacije

## Rješenja

- maksimalno pridržavanje rokova i kontinuiran rad
- osobno usavršavanje u pojedinim područjima

Tijekom rada na razvoju aplikacije FlipMemo naučili smo da su brza i otvorena komunikacija uz savjestan i odgovoran pristup zadatku ključne stavke svakog uspješnog projekta.