

Programsko inženjerstvo ak.god 2025./2026.

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

Flip Memo

Tim: <TG08.2>

Ime tima: Prijatelji

Nastavnik: Vlado Sruk

Live demo

<https://fmimage.onrender.com>

Opis projektnog zadatka

Sažetak projekta

Cilj projekta je razvoj **web aplikacije za učenje stranih jezika** koja koristi **metodu ponavljanja s odmakom (spaced repetition)** za učinkovito pamćenje vokabulara.

Aplikacija omogućuje učenicima da kroz različite modove učenja (odabir prijevoda, unos riječi, snimanje izgovora) napreduju kroz „posude“ riječi ovisno o točnosti odgovora.

Administratori mogu upravljati rječnicima, riječima i korisnicima te koristiti vanjske API-je (npr. RapidAPI) za dohvat prijevoda i ocjenu izgovora.

Cilj i svrha

Cilj projekta je izgraditi intuitivan i učinkovit alat za učenje stranih jezika temeljen na znanstveno dokazanoj metodi ponavljanja s odmakom.

Svrha je olakšati dugoročno pamćenje vokabulara te pružiti platformu koja kombinira administrativnu kontrolu, fleksibilne modove učenja i praćenje napretka korisnika.

Ciljana skupina

Učenici i studenti

- korisnici koji žele efikasno proširiti vokabular
- osobe koje uče strani jezik za školu, fakultet ili certifikate

Profesori, lektori i edukatori

- mogu izrađivati specifične rječnike za svoje kolegije
- prate napredak učenika u strukturiranom okruženju

Odrasli samouki korisnici

- profesionalci koji žele proširiti poslovni vokabular
- korisnici koji preferiraju individualno učenje

Administratori jezika

- stručnjaci koji dodaju, uređuju i organiziraju sadržaj
- korisnici koji trebaju stabilnu administrativnu kontrolu

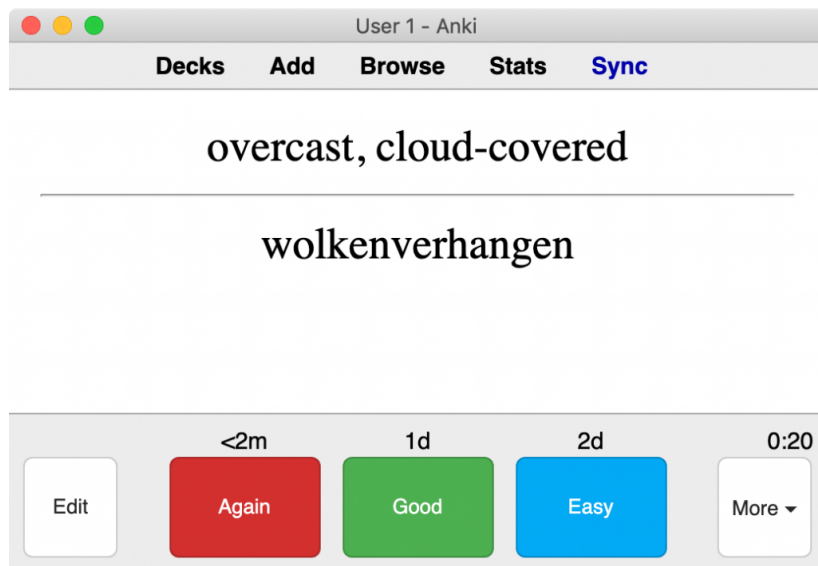
Sustav kombinira akademsku preciznost, jednostavnost korištenja i prilagodljivost potrebama različitih tipova korisnika.

Postojeća slična rješenja

Na tržištu postoji više poznatih rješenja za učenje vokabulara i jezika koja koriste elemente ponavljanja s odmakom. U nastavku su navedeni neki slični alati, zajedno sa slikom njihovog sučelja, prednostima, ograničenjima i usporedbom s našom aplikacijom.

Anki

Anki je najpopularniji open-source alat koji koristi **SRS (Spaced Repetition System)**. Temelji se na digitalnim karticama koje korisnik može sam kreirati ili preuzeti od zajednice.



Tijekom učenja ponuđeno više opcija za određivanje znanja (ne samo da ili ne) što pokazuje napredniji SRS algoritam. Naprotiv tome, sučelje je prepuno informacija i nije previše stilizirano što može odvući od fokusiranog učenja i otežati snalaženje novim korisnicima.

Prednosti:

- detaljan i učinkovit SRS algoritam
- alati za prilagodbu i napredne postavke
- podrška za dodatke i ekstenzije
- velika zajednica i baza rječnika

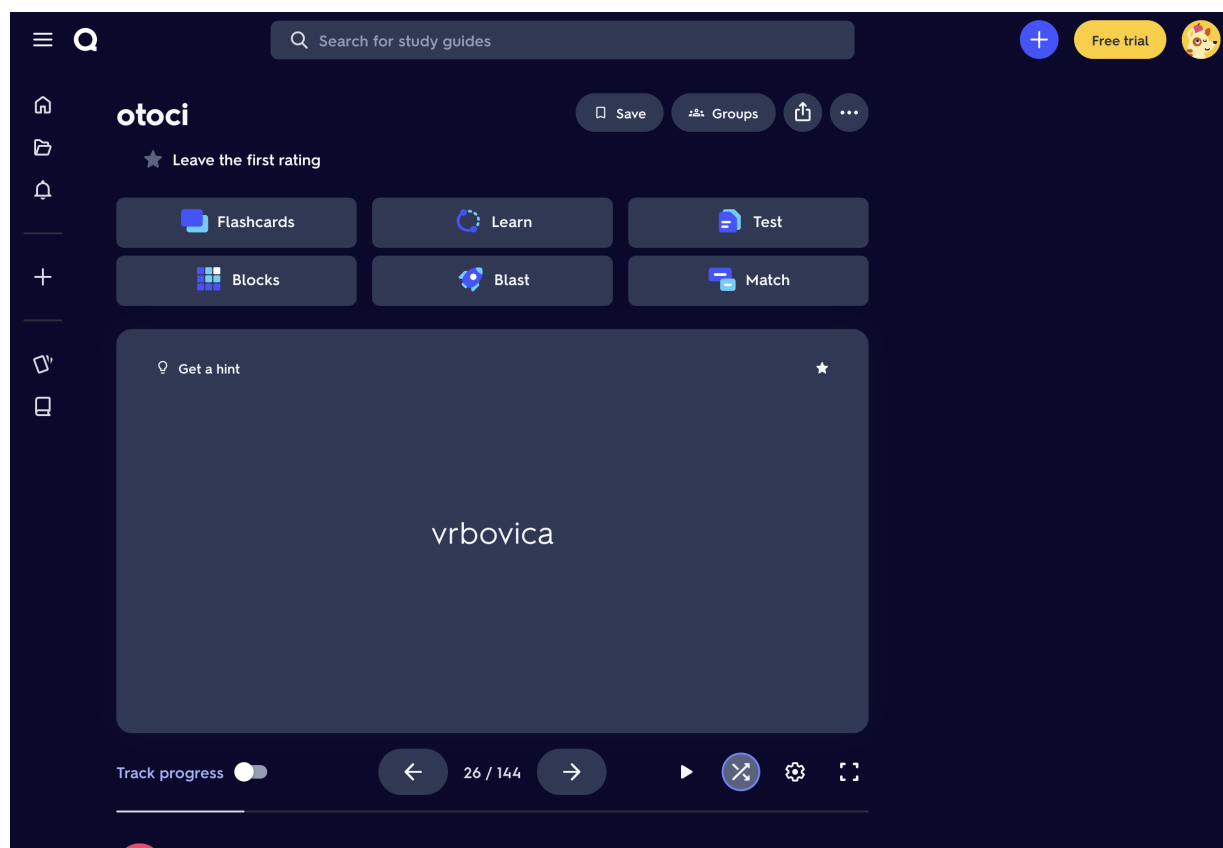
Nedostaci:

- sučelje je kompleksno i zastarjelo
- nije specijaliziran za jezike
- nema integrirano upravljanje rječnicima
- nema sustav snimanja izgovora ili provjere točnosti izgovora
- administratorske funkcionalnosti su vrlo ograničene

Naše rješenje zadržava učinkovitost SRS-a, ali ga pojednostavljuje i obogaćuje dodatnim modulima (izgovor, administracija, više modova učenja).

Quizzlet

Quizlet je edukacijski alat fokusiran na kartice i kvizove, ali ne koristi klasičan SRS.



Jako jednostavno i lijepo sučelje s manjkom nevažnih informacija. Aplikacija sadrži više načina učenja, uključujući opcije **Blocks** i **Blast** koje interaktivne igre koje uključuju rješavanje zadataka za učenje. Međutim, nema dobar sustav za pamćenje napretka za učenje.

Prednosti:

- jednostavno i privlačno sučelje
- brza izrada kartica
- veliki broj javno dostupnih skupova riječi
- različiti načini učenja

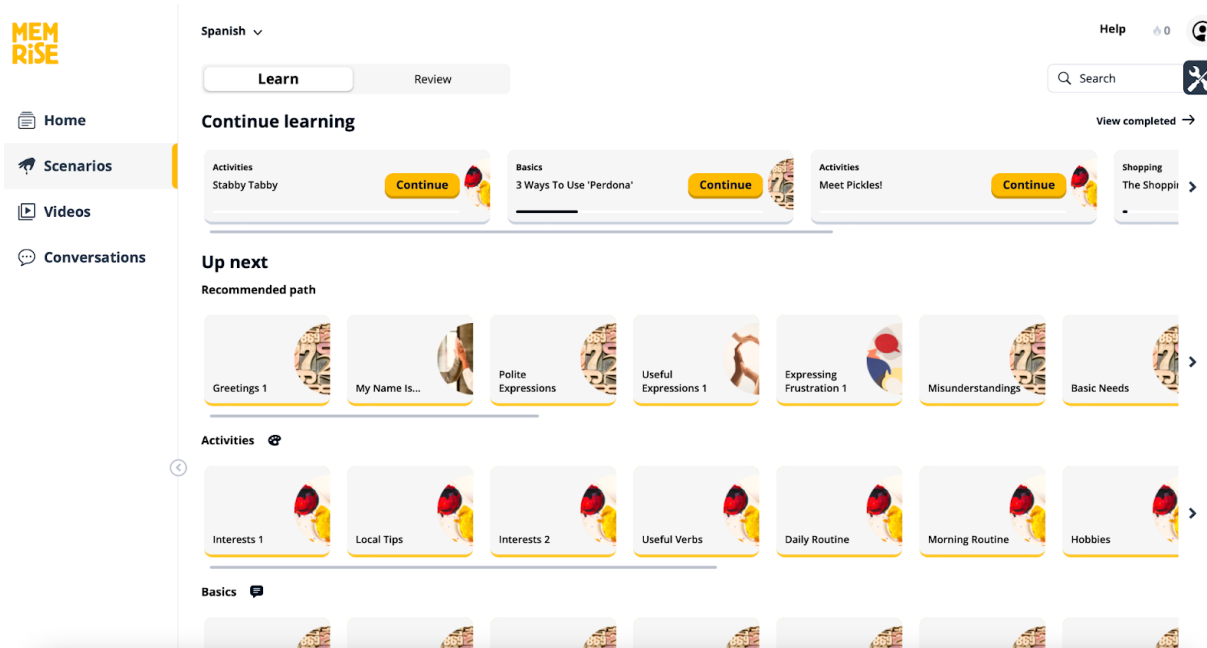
Nedostaci:

- ne koristi pravi SRS algoritam
- nije fokusiran isključivo na učenje jezika
- nema napredan sustav ponavljanja niti praćenja napretka
- administracija rječnika je ograničena

Naša aplikacija želi ponuditi **Quizletovu** pristupačnost, ali uz ozbiljniju logiku učenja i sustav posuda.

Memrise

Memrise je platforma usmjerena na učenje jezika kroz videozapise izvornih govornika, vokabular i gamifikaciju. Često se spominje kao natjecatelj s najpopularnijom aplikacijom za učenje jezika, **Duolingo**.



Memrise je jako sličan **Quizletu**. Oboje imaju moderna i privlačna sučelja što može poboljšati iskustvo korištenja aplikacije i učenja preko nje, no imaju manju transparentnost SRS algoritma što otežava korisniku da vidi logiku ponavljanja.

Prednosti:

- moderni vizuali i animacije
- korištenje izgovora iz stvarnog života
- motivacijski elementi (XP, levele, streakovi)

Nedostaci:

- algoritam ponavljanja nije transparentan
- ograničene administrativne mogućnosti
- nema precizno upravljanje rječnicima
- ne postoji sustav izvođenja vlastitih rječnika

Naše rješenje je više edukativno i akademski usmjereno, s jasnim SRS pravilima i kontrolom sadržaja.

Opći zaključak analize postojećih rješenja

Ova rješenja pružaju dobre primjere, ali svako ima ograničenja za akademski ili profesionalni rad s rječnicima.

Naš projekt objedinjuje njihove ključne prednosti:

- SRS učinkovitost (**Anki**)

- jednostavnost i vizualnu jasnoću (**Quizlet**)
- fokus na izgovor i audio materijale (**Memrise**)

i dodaje funkcionalnosti koje drugi alati nemaju:

- administratorski sustav
 - upravljanje rječnicima i jezicima
 - višestruke modove učenja
 - API integracije
 - simulaciju servisa za ocjenu izgovora
-

Moguće nadogradnje da je projekt trajao duže

Da je projekt imao više razvojnih faza i nekoliko dodatnih mjeseci, aplikacija bi se mogla značajno proširiti sljedećim funkcionalnostima:

1. Napredna analitika učenja

- vizualizacija napretka kroz grafove
- predikcija vremena potrebnog da se riječ nauči
- AI preporuke za ponavljanje

2. Realni servis za ocjenjivanje izgovora

- personalizirani savjeti za poboljšanje izgovora
- mobilna aplikacija
- offline mod
- push notifikacije za vrijeme ponavljanja

3. "Gamifikacija"

- XP sustav, značke, dnevni streak
- rang liste korisnika
- izazovi između učenika

4. Napredniji sustav krivih odgovora

- generiranje distraktora na temelju lingvističke sličnosti (AI)
- prilagodba težine odgovora korisnikovom napretku

5. Multi-user rječnici i timski rad

- profesori izrađuju rječnike dostupne cijeloj grupi
 - učenici mogu dijeliti napredak
-

Struktura dokumentacije

Projektna dokumentacija organizirana je u više cjelina koje detaljno opisuju pojedine aspekte sustava:

- [Analiza zahtjeva](#) — opis funkcionalnih i nefunkcionalnih zahtjeva sustava te korisničkih potreba.
 - [Specifikacija zahtjeva sustava](#) — formalni popis zahtjeva i definicija ponašanja sustava.
 - [Arhitektura i dizajn sustava](#) — opis arhitekture, modula i obrazaca dizajna.
 - [Arhitektura komponenata i razmještaja](#) — dijagrami komponenata, baza podataka i odnosa između servisa.
 - [Ispitivanje programskog rješenja](#) — metode i rezultati testiranja (jedinični, integracijski, E2E testovi).
 - [Tehnologije za implementaciju aplikacije](#) — opis korištenih tehnologija (React, Node.js, PostgreSQL, OAuth 2.0, itd.).
 - [Upute za puštanje u pogon](#) — koraci za pokretanje aplikacije lokalno i u produkciji.
 - [Zaključak i budući rad](#) — sažetak postignutog, izazovi i prijedlozi za nadogradnje.
 - [A. Dnevnik promjena dokumentacije](#) — evidencija svih izmjena i dopuna u projektnim dokumentima. Sadrži tablični prikaz promjena s datumima, opisima i potpisima članova tima koji su ažurirali dokumentaciju.
 - [A. Popis literature](#) — popis korištenih izvora, znanstvenih članaka, online rječnika, API dokumentacija i ostalih referenci. Ova stranica osigurava transparentnost i mogućnost provjere izvora korištenih u razvoju aplikacije.
 - [B. Prikaz aktivnosti grupe](#) — pregled rada tima kroz faze projekta. Sadrži organizirane sastanke, plan rada, tablice i dijagrame aktivnosti i evidenciju suradnje unutar tima.
-

Ključne funkcionalnosti

- Višestruki modovi učenja (odabir prijevoda, unos, izgovor).
 - Sustav ponavljanja s odmakom („posude“).
 - Administratorsko sučelje za upravljanje rječnicima i korisnicima.
 - Integracija s vanjskim API-jevima (riječnici, ocjena izgovora).
 - Sigurna autentifikacija putem **OAuth 2.0**.
 - Praćenje napretka i analiza učenja po korisniku.
-

Tehnički okvir

- **Frontend:** React
 - **Backend:** Node.js
 - **Baza podataka:** PostgreSQL
 - **Autentifikacija:** OAuth 2.0
 - **Vanjski servisi:** RapidAPI (riječnici), API za ocjenu izgovora
 - **Razvojno okruženje:** Docker, GitHub, CI/CD
-

Zaključak

Ova aplikacija ima za cilj spojiti moderne tehnologije i pedagoške metode učenja jezika u cjelovito rješenje koje je jednostavno za korištenje, skalabilno i prilagođeno različitim razinama znanja.

Prva stranica Wiki-ja služi kao pregled projekta i vodič kroz sve detaljnije sekcije dokumentacije.

Napomena:

Detaljniji opisi, dijagrami, testni primjeri i tehničke specifikacije nalaze se na pripadajućim Wiki stranicama, dostupnima putem poveznica iznad.

Funkcionalni zahtjevi

ID Zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvatanja
F001	Sustav omogućuje adminu dodavanje novog riječnika.	Srednji	Postojeći sustav	Na zahtjev administratora kreira se novi riječnik
F002	Sustav omogućuje adminu pretragu riječi koja bi se dodala u riječnik.	Srednji	Postojeći sustav	Aplikacija komunicira s vanjskim riječnikom kako bi adminu omogućila proširivanjem riječi dodavanjem kontekstnih rečenica.
F003	Sustav omogućuje adminu dodavanje novih riječi jednom ili više riječnika.	Srednji	Zahtjev dionika	Na zahtjev administratora se mogu dodavati riječi iz vanjskog riječnika u jedan ili više već postojećih aplikacijskih riječnika
F004	Sustav omogućuje adminu brisanje ili promjenu komponenata.	Srednji	Zahtjev dionika	Na zahtjev administratora već dodane riječi se mogu promijeniti ili izbrisati.
F005	Sustav omogućava korijenskom adminu da definira druge admina.	Srednji	Zahtjev dionika	Na zahtjev korijenskog administratora se mogu definirati novi administratori s istim ovlastima kao i korijenskog administratora.
F006	Sustav omogućuje korisniku da se registira adresom korisničke pošte.	Visok	Zahtjev dionika	Korisnik se može registrirati e-mailom, primiti e-mail za potvrdu i uspješno se prijaviti.
F007	Sustav omogućava korisniku promjenu lozinke.	Srednji	Zahtjev dionika	Korisnik može zatražiti promjenu lozinke i uspješno se prijaviti s novom lozinkom.
F008	Sustav omogućava korisniku brisanje korisničkog računa.	Srednji	Zahtjev dionika	Korisniku se pri zatraženju i adekvatnim upozorenjima dopušta brisanje korisničkog računa.
F009	Sustav omogućuje mod za učenje prijevodenja riječi sa stranog na materinji jezik.	Visok	Postojeći sustav	Pri dobitku stranog pojma, dani su ponuđeni prijevodi na materinji jezik od kojih je samo jedan točan.

ID Zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvatanja
F010	Sustav omogućuje mod za učenje prijevodenja riječi sa materinjeg na strani jezik.	Visok	Postojeći sustav	Pri dobitku pojma, dani su ponuđeni strani prijevodi od kojih je samo jedan točan.
F011	Sustav nasumično bira netočne odgovore iz istog riječnika.	Visok	Postojeći sustav	U pitanjima sa više ponuđenih odgovora, sustav osigurava nasumične krive prijevode iz istog riječnika.
F012	Sustav omogućava korisniku da odabere točan odgovor prijevoda	Visok	Postojeći sustav	Učenik odgovara s prijevodima riječi odabirući ispravan odabir od nekoliko ponuđenih alternativa koje su krive.
F013	Sustav omogućuje mod za provjeru točnosti pisanja riječi.	Visok	Postojeći sustav	Pri slušanju stranog pojma, korisnik treba napisati točno napisati poslušani sadržaj.
F014	Sustav provjerava točnost napisanih stranih riječi.	Visok	Postojeći sustav	Preko lokalne baze podataka se provjerava točnost napisanog sadržaja strane riječi.
F015	Sustav omogućuje mod za provjeru točnosti izgovora riječi.	Visok	Postojeći sustav	Pri čitanju tekstualne strane riječi, korisnik ju treba točno izgovoriti.
F016	Sustav provjerava točnost snimljenih izgovorenih stranih riječi uz vanjske servise.	Visok	Postojeći sustav	U sustavu je implementirano aplikacijsko sučelje koje ocjenjuje izgovor na skali od 1 do 10.
F017	Sustav omogućuje korisniku pokretanje odabranog riječnika u odabranom specifičnom modu.	Visok	Zahtjev dionika	Korisnik nastavlja učenje od zadnje upotrebe tog riječnika u tom specifičnom modu.
F018	Sustav omogućuje premještanje riječi u sljedeću posudu pri točnom odgovoru	Visok	Postojeći sustav	Ako učenik odgovori točnim odgovorom po planu učenja riječ ide u sljedeću posudu.

ID Zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvatanja
F019	Sustav omogućuje premještanje riječi u prvu posudu pri netočnom odgovoru.	Visok	Postojeći sustav	Ako učenik odgovori netočnim odgovorom po planu učenja riječ ide nazad na učenje iznova.
F020	Sustav omogućuje da posude imaju određeno vrijeme nakon čijeg isteka riječ se ponovno ispituje	Srednji	Postojeći sustav	Vrijeme isteka za posude raste, jer što je daljnja posuda u nizu trebaju se raditi veći razmaci između ponavljanja. Zadnja posuda ima vrijeme isteka nekoliko tjedana do mjesec dana, nakon vremena isteka i točno odgovorenog pitanja riječ se smatra naučenom.

Ostali zahtjevi

Zahtjevi za održavanje

ID Zahtjeva	Opis	Prioritet
NF-001	Aplikacija mora osigurati zaštitu osobnih podataka korisnika i sigurnost prijave u skladu s važećim propisima (OAuth 2.0 standard autentikacije)	Visok
NF-002	Pristup podacima i funkcionalnostima mora biti ograničen na temelju korisničkih uloga.	Visok
NF-003	Aplikacija mora biti raspoznavna i prilagođena za rad na različitim uređajima	Srednji
NF-004	Aplikacija mora biti laka za korištenje i intuitivna za nove korisnike.	Visok
NF-005	Sustav mora imati brzo vrijeme odziva za ključne korisničke akcije.	Visok
NF-006	Sustav mora biti u stanju nositi se s naglim porastom broja korisnika	Srednji
NF-007	Sustav mora osigurati da podaci o korisničkim računima i njihovom napretku ne budu izgubljeni ili oštećeni.	Visok
NF-008	Sustav treba biti oblikovan tako da omogućuje jednostavno održavanje.	Visok
NF-009	Sustav treba imati dovoljnu dokumentaciju.	Visok
NF-010	Sustav treba imati "Plan implementacije" za pravilno postavljanje sustava.	Visok

Dionici

Dionik je svaka osoba, grupa ili organizacija koja sudjeluje u projektu ili na koju projekt utječe, u tvrtki koja razvija projekt i izvan nje. U ovom projektu to su:

- **D-1 Korisnici** - osobe koje koriste sustav za učenje jezika
- **D-2 Administratori** - osobe zadužene za upravljanje sadržajem i korisnicima
- **D-3 Korijenski administrator** - administrator s potpunim ovlastima
- **D-4 Naručitelj** - korisnik ili organizacija koja je naručila razvoj sustava
- **D-5 Razvojni tim** - programeri i dizajneri koji razvijaju i održavaju sustav

Primarni dionici su dionici na koje projekt utječe izravno - oni aktivno sudjeluju u njegovu razvoju i implementaciji. Od gore navedenih to su administratori, korijenski administrator, naručitelj i razvojni tim. Sekundarni dionici su oni dionici na koje projekt utječe neizravno što su sami krajnji korisnici sustava.

Aktori i njihovi funkcionalni zahtjevi:

1. **A-1 Korisnik** (sudionik) može:

- registrirati se putem e-mail adrese (F005)
- promijeniti lozinku (F006)
- obrisati korisnički račun (F007)
- koristiti mod učenja prijevoda sa stranog na materinji jezik (F008)
- koristiti mod učenja prijevoda s materinjeg na strani jezik (F009)
- odabrati točan odgovor prijevoda (F011)
- koristiti mod za provjeru točnosti pisanja riječi (F012)
- koristiti mod za provjeru točnosti izgovora riječi (F014)
- pokrenuti odabrani rječnik u odabranom specifičnom modu (F016)

2. **A-2 Administrator** (inicijator) može:

- pretraživati riječi za dodavanje u rječnik (F001)
- dodavati nove riječi u jedan ili više rječnika (F002)
- brisati ili mijenjati komponente rječnika (F003)

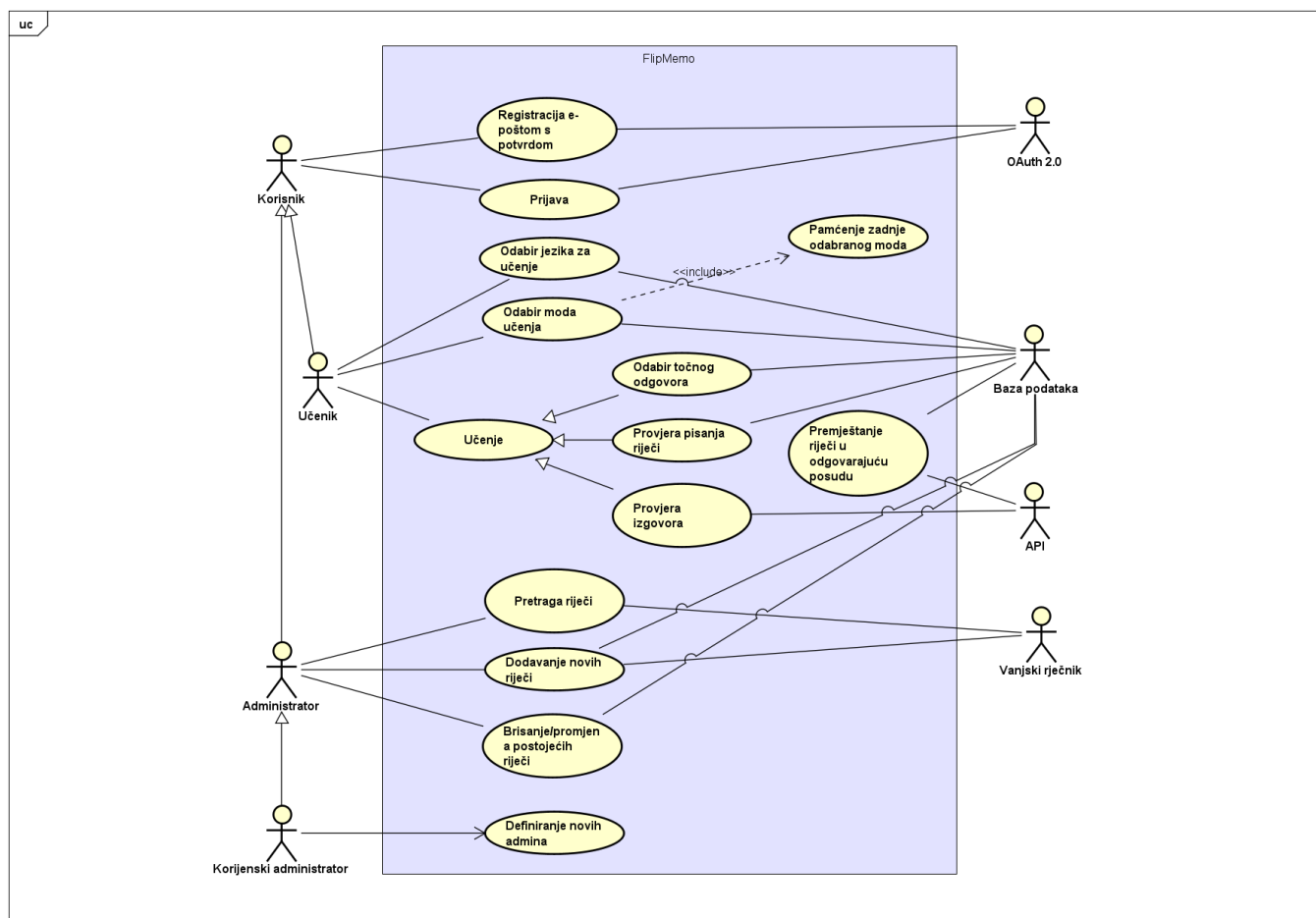
3. **A-3 Korijenski administrator** (inicijator) može:

- sve što i običan administrator (F001; F002; F003)
- definirati druge administratore (F004)

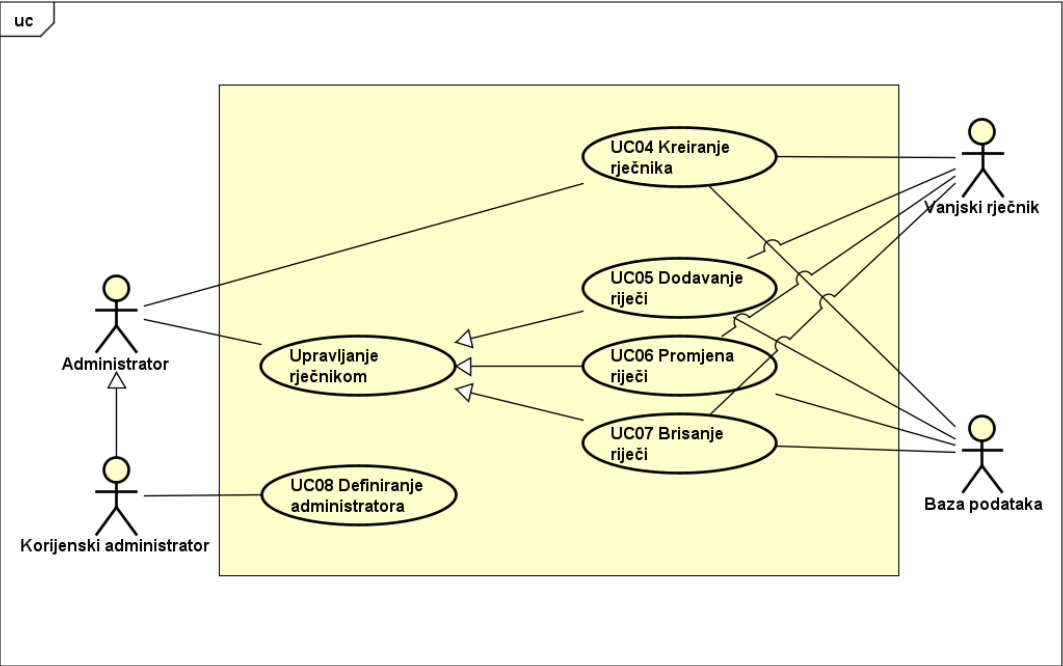
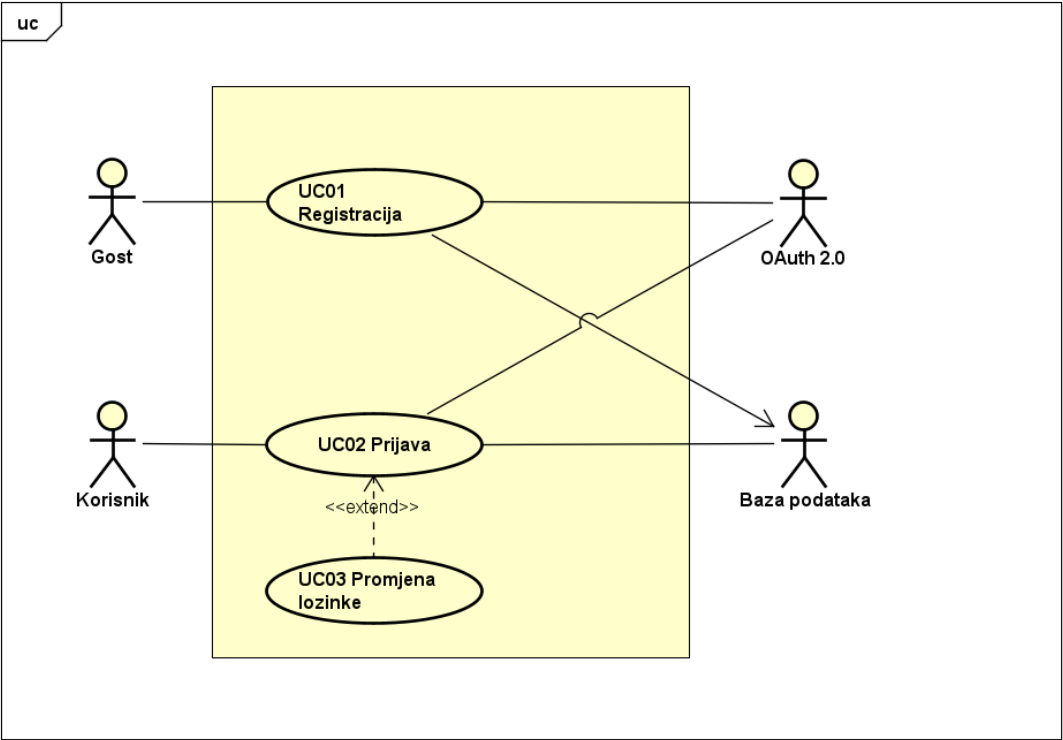
Obrasci uporabe

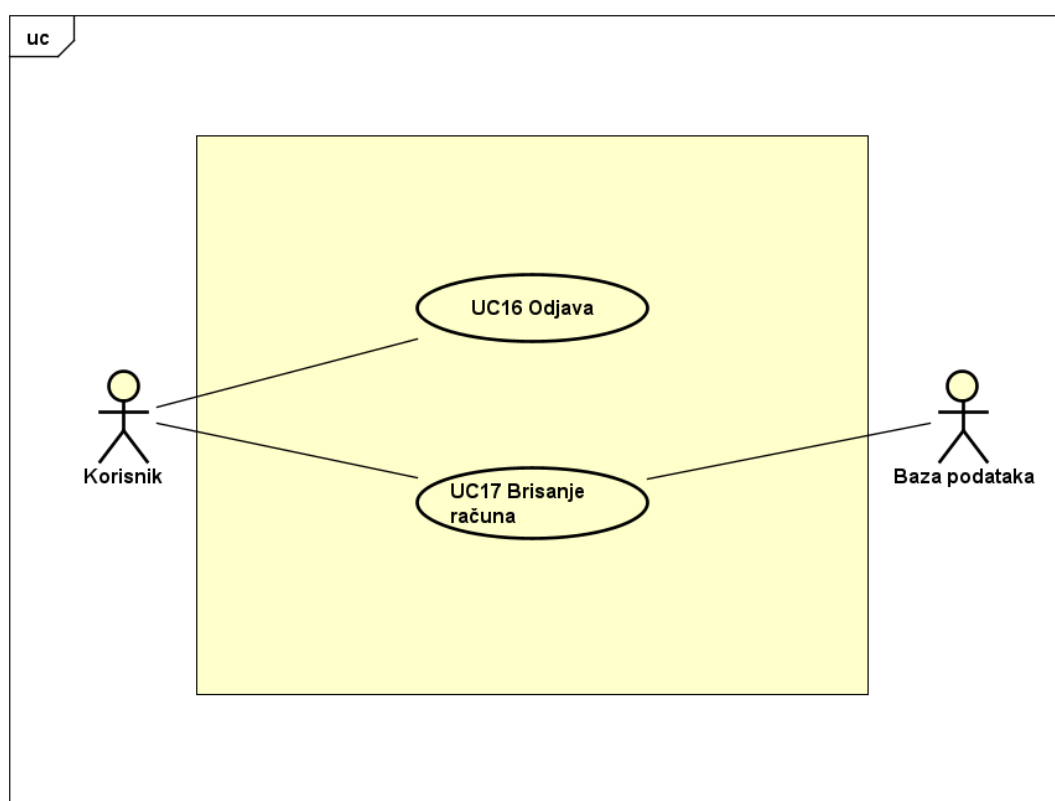
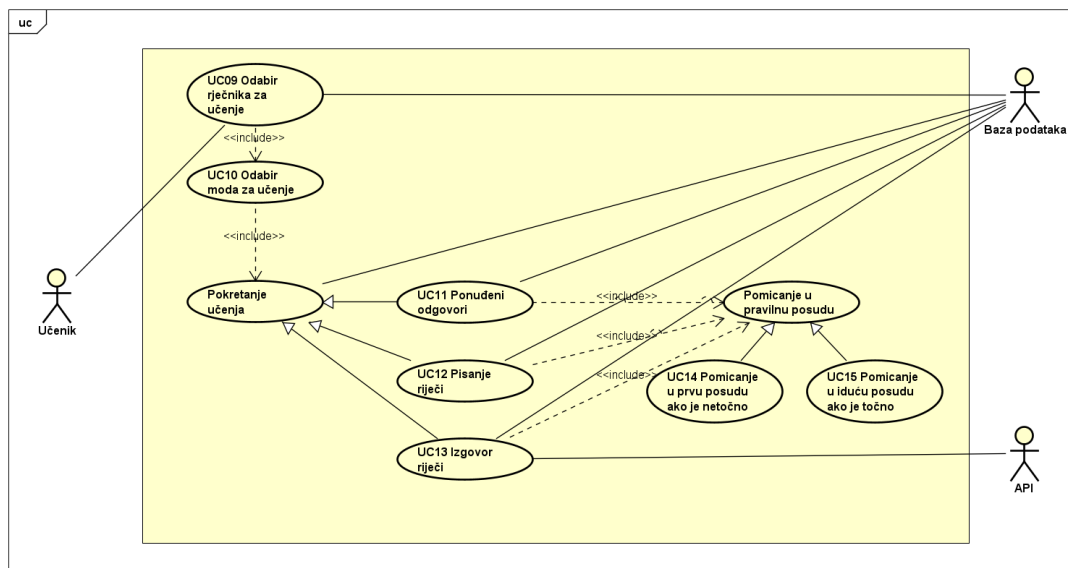
Dijagrami obrazaca uporabe

1. Visokorazinski dijagram obrazaca uporabe cijelog sustava



2. dijagram obrazaca uporabe za ključne značajke





Opis obrazaca uporabe

UC01 - Registracija

- Glavni sudionik: gost (neregistrirani korisnik)
- Cilj: registracija novog korisnika na stranicu
- Sudionici: gost, OAuth 2.0, baza podataka
- Preduvjet: gost nema već napravljen račun
- Opis osnovnog tijeka:

1. Gost otvara stranicu za registraciju na web aplikaciju
2. Upisuje svoje podatke koji se šalju na OAuth 2.0 server
3. OAuth 2.0 autentificira korisnika, provjerava je li već registriran i pravilnost unesenih podataka
4. Korisnički podaci zapisuju se u bazu podataka

5. Korisnik ima pristup na web aplikaciju

- Opis mogućih odstupanja:

3. U bazi podataka već postoji račun s tom e-mail adresom te se šalje obavijest gostu da može probat s drugim e-mailom ili otići na prijavu

UC02 - Prijava

- Glavni sudionik: korisnik
- Cilj: prijava korisnika na stranicu
- Sudionici: korisnik, OAuth 2.0, baza podataka
- Preduvjet: korisnik je već registriran na stranicu
- Opis osnovnog tijeka:

1. Korisnik otvara stranicu za prijavu na web aplikaciju
2. Korisnik upisuje svoje podatke koji se šalju na OAuth 2.0 server
3. OAuth 2.0 autentificira korisnika, provjerava je li već registriran i pravilnost unesenih podataka
4. Korisnik ima pristup na web aplikaciju

- Opis mogućih odstupanja:

3.a korisnički podaci su krivo uneseni, korisnik dobiva obavijest i može pokušati ponovno 3.b korisnički račun ne postoji, korisnik dobiva obavijest i preusmjerava se na stranicu za registraciju

UC03 - Promjena lozinke

- Glavni sudionik: korisnik
- Cilj: promjena lozinke korisničkog računa prijavljenog na stranicu
- Sudionici: korisnik, baza podataka
- Preduvjet: korisnik je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Korisnik izabire opciju promjene lozinke
2. Korisnik upisuje svoju trenutnu lozinku, novu lozinku, potvrdu nove lozinke
3. Sustav provjerava je li točno unesena trenutna lozinka, jesu li nova lozinka i potvrda nove lozinke jednake
4. Stara lozinka se briše iz baze podataka, a na njeno mjesto se upisuje nova
5. Korisnik dobiva obavijest da je promijenio svoju lozinku

- Opis mogućih odstupanja:

3.a Unesena trenutna lozinka i trenutna lozinka u bazi podataka se ne podudaraju, korisnik dobiva obavijest i može pokušati ponovno 3.b Unesena nova lozinka i potvrda nove lozinke nisu jednake, korisnik dobiva obavijest i može pokušati ponovno

UC04 - Kreiranje rječnika

- Glavni sudionik: administrator
- Cilj: kreirati rječnik za učenje novog jezika na stranici
- Sudionici: administrator, baza podataka

- Preduvjet: administrator je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Administrator odabire opciju "Kreiraj novi rječnik"
2. Administrator upisuje naziv rječnika, jezik te kratki opis
3. Novi rječnik je spremljen u bazu podataka
4. Administrator dobiva obavijest da je kreirao novi rječnik

- Opis mogućih odstupanja:

3. U bazi podataka već postoji rječnik s istim nazivom za taj jezik, administrator dobiva obavijest o grešci te se prekida stvaranje novog jezika

UC05 - Dodavanje riječi

- Glavni sudionik: administrator
- Cilj: dodavanje nove riječi u jedan ili više postojećih rječnika na stranici
- Sudionici: administrator, vanjski rječnik, baza podataka
- Preduvjet: administrator je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Administrator odabire opciju "Dodavanje riječi"
2. Administrator bira riječ, fraze, prijevod riječi i fraza, glasovnu datoteku riječi
3. Administrator bira u koje rječnike će dodati novu riječ te potvrđuje svoj odabir
4. Riječ se dodaje u rječnike te administrator dobiva obavijest o uspješnosti dodavanja riječi

- Opis mogućih odstupanja:

4. Ako riječ postoji u nekom jeziku, u njega se ne dodaje te je administrator obavješten da dodavanje riječi nije bilo uspješno za taj rječnik

UC06 - Promjena riječi

- Glavni sudionik: administrator
- Cilj: promjena neke od komponente izabrane riječi u nekom rječniku
- Sudionici: administrator, baza podataka
- Preduvjet: administrator je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Administrator bira željeni jezik za koji postoji barem jedan rječnik
2. Administrator bira željenu riječ iz tog jezika
3. Administrator odabire opciju "Promjena riječi"
4. Mijenja jednu ili više komponenata te riječi te sprema promjene

- Opis mogućih odstupanja:

4. Ako je polje za prijevod izbrisan i ostavljen prazan, promjena se događa te je administrator obavješten da to polje ne smije ostaviti prazno

UC07 - Brisanje riječi

- Glavni sudionik: administrator
- Cilj: izbrisati riječ iz nekog rječnika
- Sudionici: administrator, baza podataka
- Preduvjet: administrator je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Administrator bira željeni jezik za koji postoji barem jedan rječnik
2. Administrator bira željenu riječ iz tog jezika
3. Administrator odabire opciju "Brisanje riječi"
4. Sustav izbacuje poruku "Jeste li sigurni da želite izbrisati ovu riječ?"
5. Administrator bira hoće li odustati ili potvrđuje svoj izbor

- Opis mogućih odstupanja:

UC08 - Definiranje administratora

- Glavni sudionik: korijenski administrator
- Cilj: definiranje novog administratora
- Sudionici: korijenski administrator, baza podataka
- Preduvjet: korijenski administrator je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Korijenski administrator izabire opciju "Definiraj administratora"
2. Otvara se lista učenika na kojoj korijenski administrator jednog ili više korisnika
3. Korijenski administrator potvrđuje svoj odabir

- Opis mogućih odstupanja:

UC09 - Odabir jezika za učenje

- Glavni sudionik: učenik
- Cilj: odabir rječnika za učenje
- Sudionici: učenik, baza podataka
- Preduvjet: učenik je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Učenik izabire "Odabir rječnika"
2. Učenik s liste izabire rječnik

- Opis mogućih odstupanja:

UC10 - Odabir moda za učenje

- Glavni sudionik: učenik
- Cilj: odabir moda za učenje
- Sudionici: učenik
- Preduvjet: učenik je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Učenik izabire "Odabir moda"

2. Učenik s liste izabire mod

- Opis mogućih odstupanja:

UC11 - Odabir točnog odgovora

- Glavni sudionik: učenik
- Cilj: naučiti točan prijevod riječi
- Sudionici: učenik, baza podataka
- Preduvjet: učenik je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Dohvaćaju se riječi za izabrani rječnik i mod za koje je prošlo prikladno vrijeme od prošlog ponavljanja
2. Ispisuje se jedna od riječ na jeziku koji se uči te 4 prijevoda riječi od kojih je jedan točan
3. Učenik bira jedan od prijevoda
4. Sustav provjerava je li izabrani odgovor točan
5. Pokazuje se koji odgovor je točan
6. Učenik izabire "Dalje" za prikaz sljedeće riječi

- Opis mogućih odstupanja:

UC12 - Provjera pisanja riječi

- Glavni sudionik: učenik
- Cilj: naučiti pravilno napisati riječ
- Sudionici: učenik, baza podataka
- Preduvjet: učenik je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Dohvaćaju se riječi za izabrani rječnik i mod za koje je prošlo prikladno vrijeme od prošlog ponavljanja
2. Pojavljuje se glasovna datoteka od jedne riječi na jeziku koji uči i polje za upis
3. Učenik stišće "Play" te upisuje riječ koju čuje
4. Sustav provjerava je li upisani odgovor točan
5. Pokazuje se pravilan zapis riječi
6. Učenik izabire "Dalje" za prikaz sljedeće glasovne datoteke

- Opis mogućih odstupanja:

UC13 - Provjera izgovora riječi

- Glavni sudionik: učenik
- Cilj: naučiti pravilno izgovarati riječ
- Sudionici: učenik, baza podataka, API server za procjenu točnosti izgovora
- Preduvjet: učenik je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Dohvaćaju se riječi za izabrani rječnik i mod za koje je prošlo prikladno vrijeme od prošlog ponavljanja

2. Ispisuje se jedna od riječ na jeziku koji se uči te gumb "Listen"
3. Učenik stišće "Listen" te izgovara riječ
4. API server ocjenjuje izgovor od 1 do 10
5. Učenik izabire "Dalje" za prikaz sljedeće riječi

- Opis mogućih odstupanja:

UC14 - Pomicanje u prvu posudu ako je netočno

- Glavni sudionik: učenik
- Cilj: pomaknuti riječ u prvu posudu
- Sudionici: učenik, baza podataka
- Preduvjet: učenik je netočno odgovorio
- Opis osnovnog tijeka:

1. Pomicanje riječi u prvu posudu
2. Riječ će se opet ponoviti za učenje u odabranom modu i rječniku nakon jednog dana

- Opis mogućih odstupanja:

UC15 - Pomicanje u iduću posudu ako je točno

- Glavni sudionik: učenik
- Cilj: pomaknuti riječ u iduću posudu
- Sudionici: učenik, baza podataka
- Preduvjet: učenik je točno odgovorio
- Opis osnovnog tijeka:

1. Pomicanje riječi u iduću posudu
2. Ako je riječ već u 5. posudi, mičemo ju iz posuda i učenik ju više ne ponavlja

- Opis mogućih odstupanja:

UC16 - Odjava

- Glavni sudionik: korisnik
- Cilj: odjaviti se sa stranice
- Sudionici: korisnik
- Preduvjet: korisnik je prijavljen na stranicu
- Opis osnovnog tijeka:

1. Korisnik izabire opciju "Odjava"

- Opis mogućih odstupanja:

UC17 - Brisanje računa

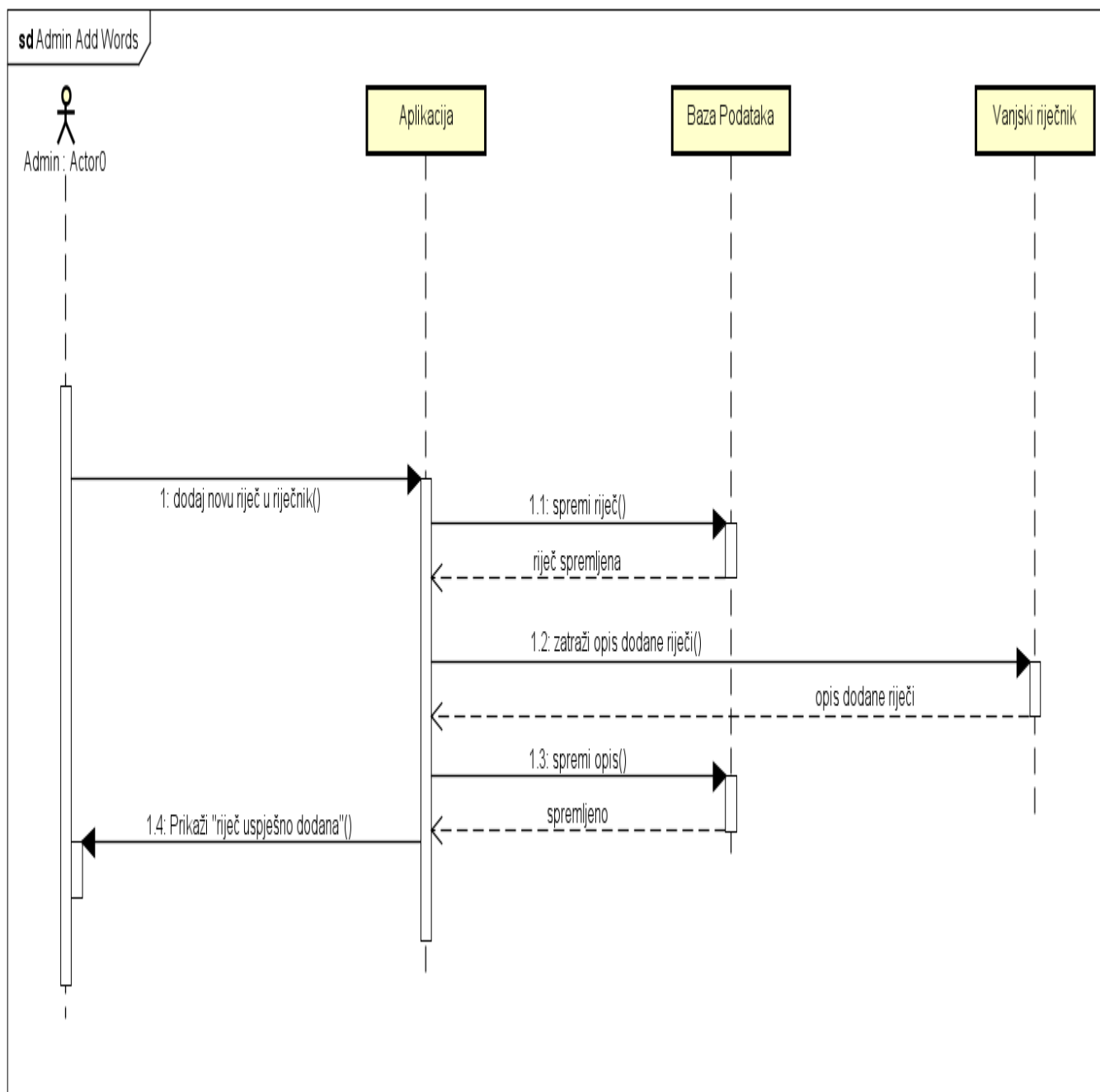
- Glavni sudionik: korisnik
- Cilj: izbrisati račun i sve podatke iz baze podataka
- Sudionici: korisnik, baza podataka
- Preduvjet: korisnik je prijavljen na stranicu

- Opis osnovnog tijeka:

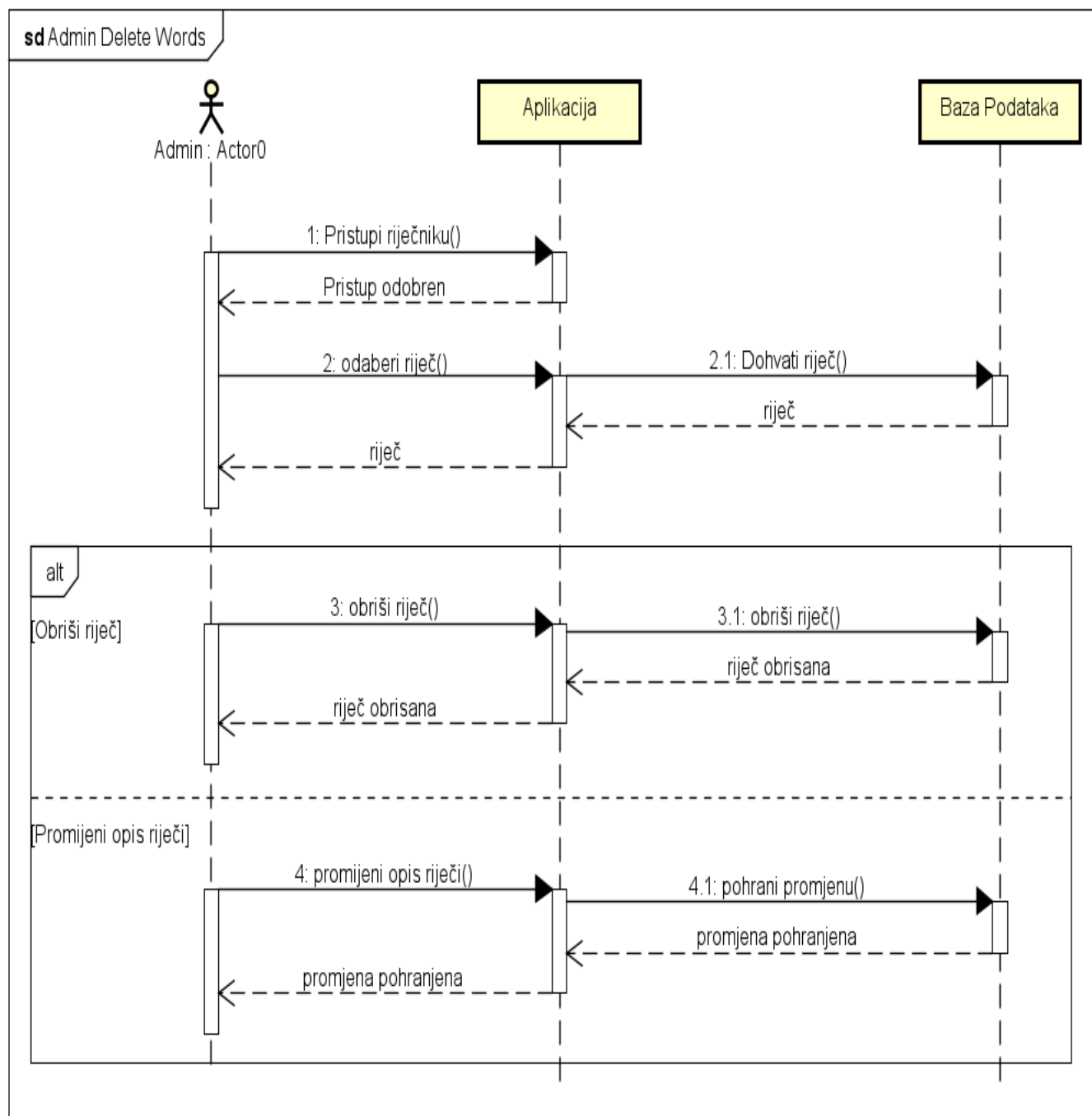
1. Korisnik izabire opciju "Brisanje računa"
2. Korisnika se pita je li siguran da želi izbrisati račun
3. Korisnik potvrđuje svoj izbor ili odustaje

- Opis mogućih odstupanja:

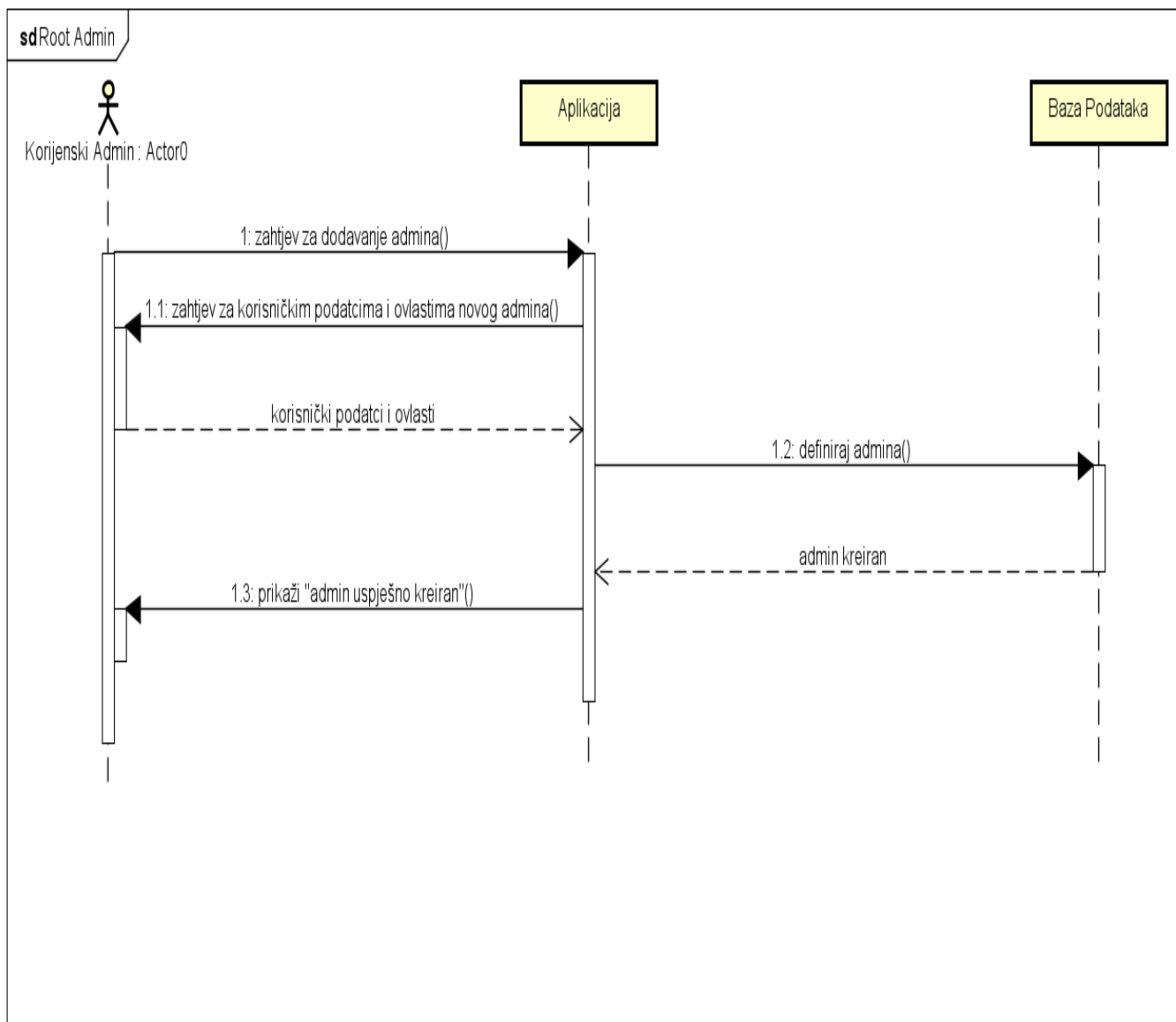
Sekvencijski dijagrami



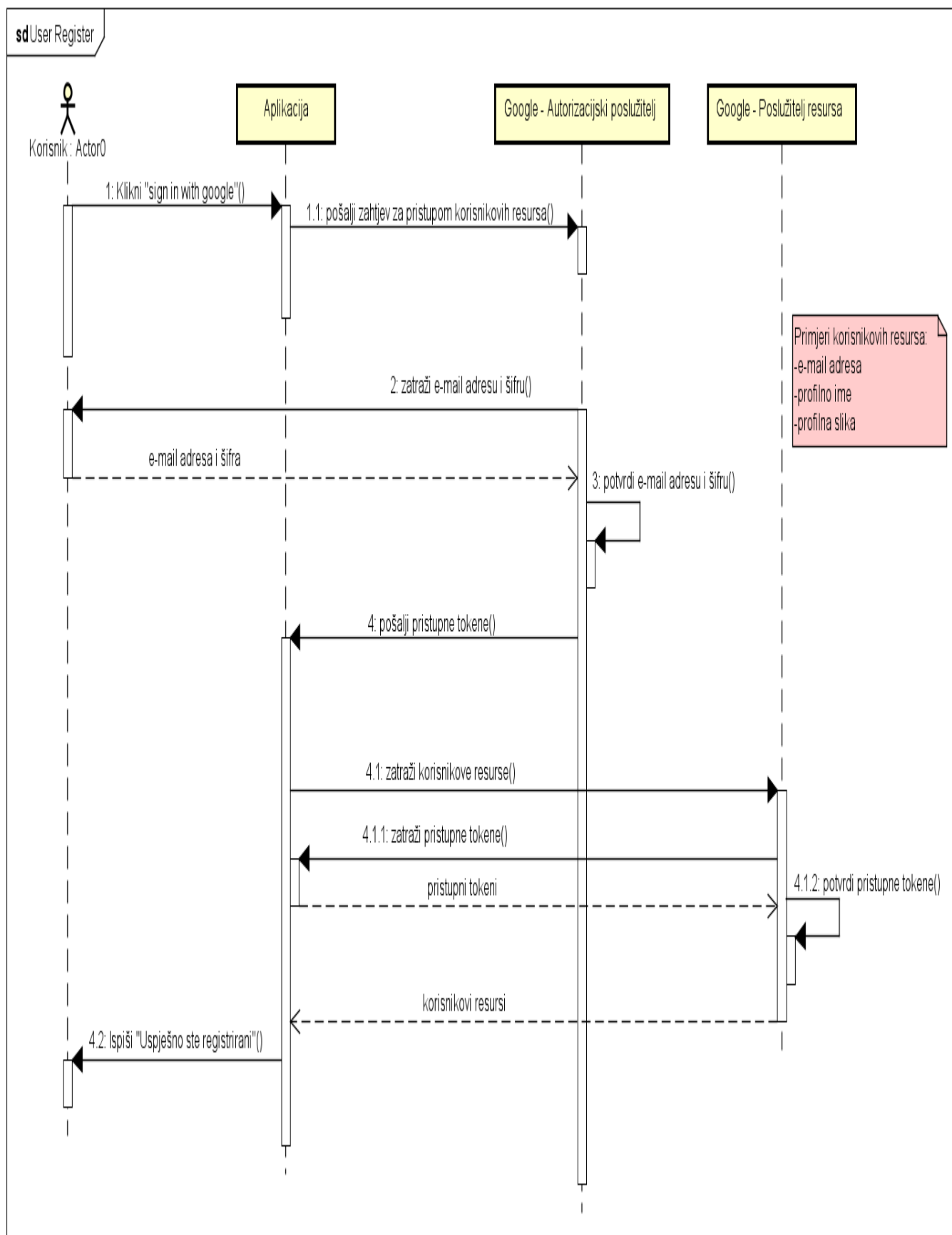
2.1 Sekvencijski dijagram prikazuje kako izgleda proces dodavanja nove riječi. Nakon što administrator doda novu riječ u riječnik, aplikacija u bazu podataka sprema riječ. Tada baza podataka aplikaciji javlja da je riječ uspješno spremljena, a aplikacija tada od vanjskog riječnika zatraži opis dodane riječi. Nakon što aplikacija dobije opis dodane riječi, ona ju sprema u bazu podataka. Ako je postupak spremanja opisa riječi uspješan, baza podataka će to dojaviti aplikaciji te će se administratoru prikazati poruka "Riječ uspješno dodana".



2.2 Sekvencijski dijagram prikazuje kako izgleda proces brisanja riječi ili mijenjanja njenog opisa. Nakon što administrator pristupi riječniku i odabere neku riječ, aplikacija će preko baze podataka tu riječ dohvatiti. Administrator tu riječ može obrisati, te će se naredba za brisanje riječi proslijediti bazi podataka. Nakon što baza podataka uspješno obriše riječ, poruka o uspješnosti će se proslijediti aplikaciji, a aplikacija će administratoru tu istu poruku i prikazati. Alternativno, administrator može i promijeniti opis odabrane riječi. Aplikacija će novi opis riječi proslijediti bazi podataka. Nakon što baza podataka uspješno promijeni opis riječi, poruka o uspješnosti će se proslijediti aplikaciji, a aplikacija će administratoru tu istu poruku i prikazati.

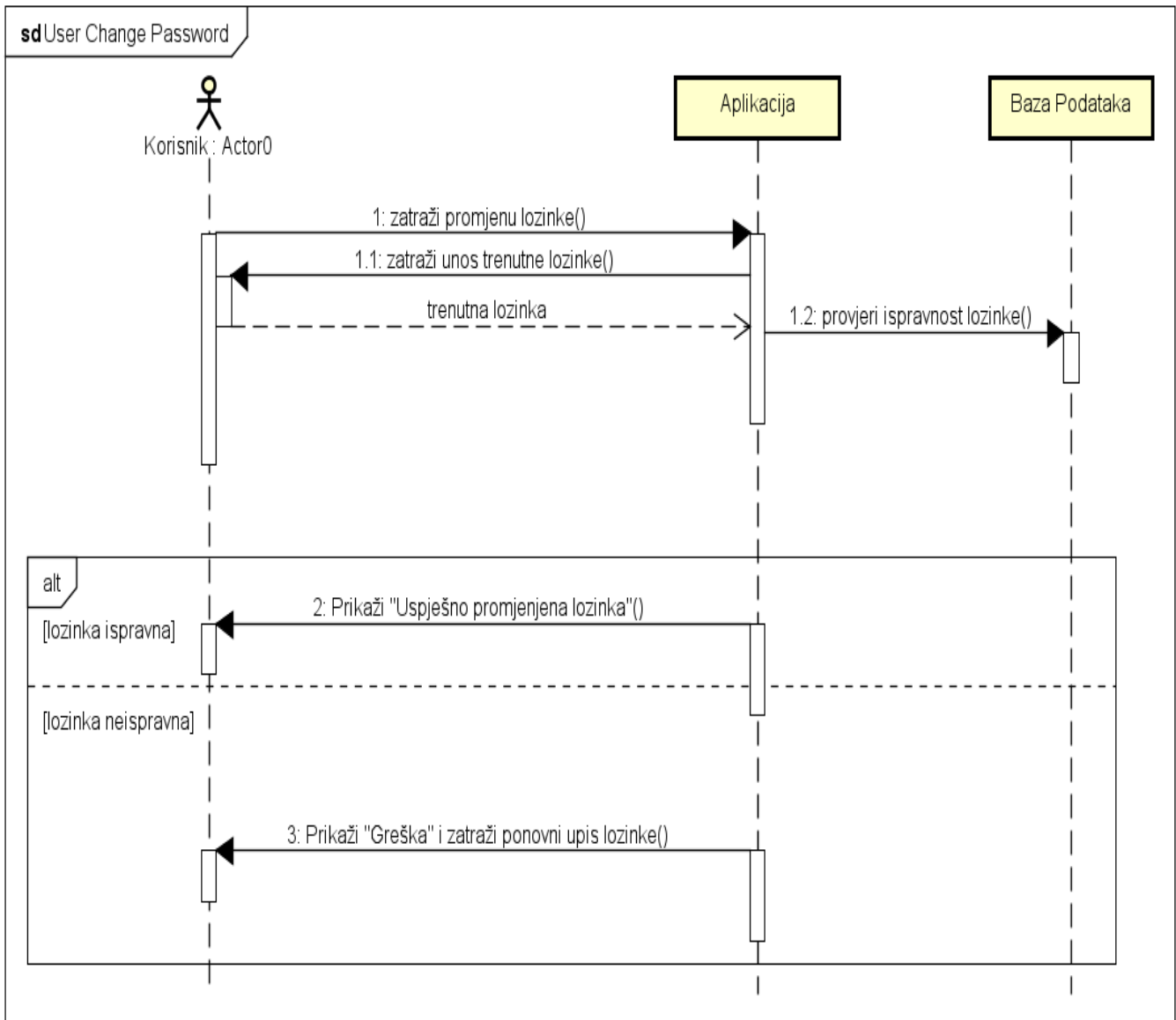


2.3 Sekvencijski dijagram prikazuje kako izgleda proces definiranja novog administratora od strane korijenskog administratora. Kada korijenski administrator odluči definirati novog administratora, aplikacija ga najprije zatraži korisničke podatke novog administratora te ovlasti koje bi on imao. Kada korijenski administrator priloži potrebne korisničke podatke novog administratora i definira njegove ovlasti, aplikacija bazi podataka prosljeđuje korisničke podatke i definirane ovlasti zajedno sa zahtjevom za kreacijom novog administratora. Ukoliko baza podataka uspješno definira novog administratora, ona će o tome obavijestiti aplikaciju, a aplikacija će zatim ispisati poruku "Admin uspješno kreiran".

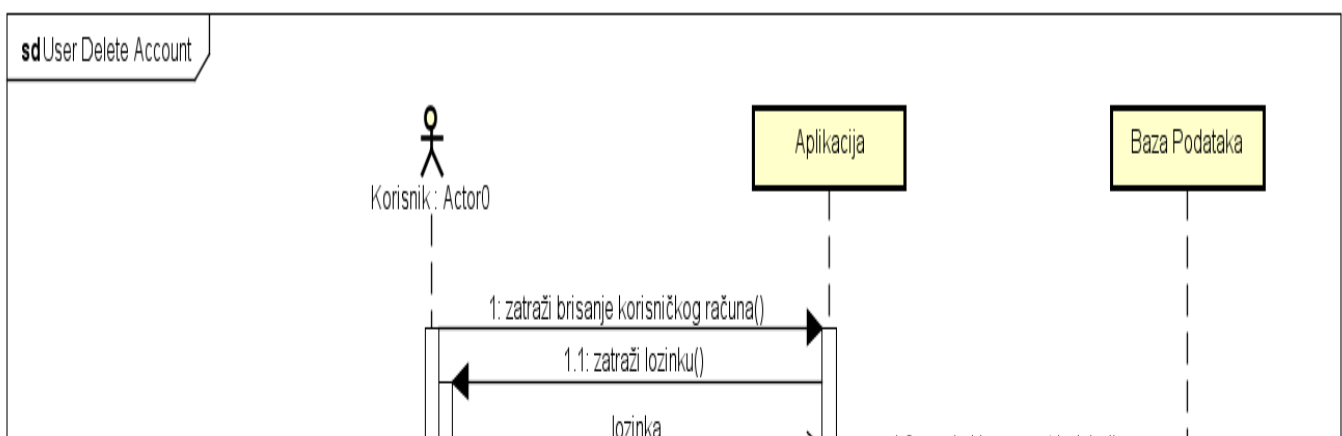


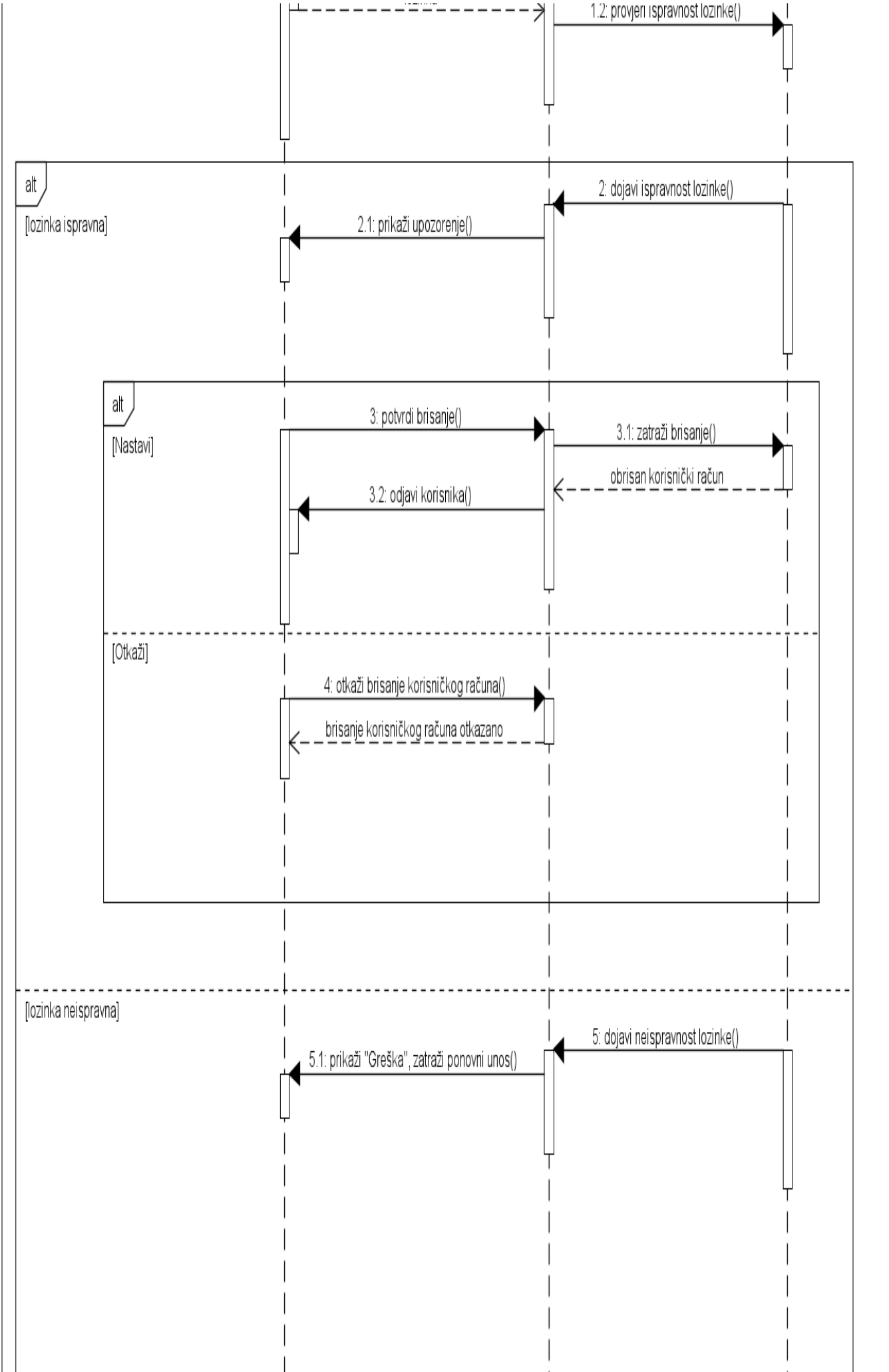
2.4 Sekvencijski dijagram prikazuje kako izgleda proces registracije novog korisnika. Proces registracije se obavlja isključivo putem Google korisničkog računa. Nakon što korisnik klikne na "Sign in with google", aplikacija će poslati zahtjev za pristupom korisnikovih resursa (primjeri korisnikovih resursa su navedeni na samom dijagramu) Googleovom autorizacijskom poslužitelju. Korisnik će tada biti preusmjeren na Google sign in stranicu gdje će ga se zatražiti e-mail adresa i šifra korisničkog računa. Nakon što korisnik uspješno unese svoju e-mail adresu i šifru i nakon što autorizacijski poslužitelj provjeri njihovu ispravnost,

autorizacijski poslužitelj aplikaciji šalje pristupne tokene. Aplikacija tada pošalje zahtjev za korisnikovim resursima na Googleov poslužitelj resursa. Poslužitelj resursa tada aplikaciju zatraži odgovarajuće pristupne tokene. Ako aplikacija uspješno pošalje ispravne pristupne tokene, poslužitelj resursa tada aplikaciji pošalje korisnikove resurse. Aplikacija tada korisnika informira o uspješnom registriranju.



2.5 Sekvencijski dijagram prikazuje kako izgleda proces mijenjanja korisnikove lozinke. Kada korisnik zatraži promjenu lozinke, aplikacija ga zatraži unos trenutne lozinke. Nakon što korisnik unese lozinku, u bazi podataka (gdje su smješteni korisnički podatci) se provjerava ispravnost trenutne lozinke. Ukoliko je lozinka ispravna, korisniku se prikazuje poruka "Uspješno promjenjena lozinka". Inače se ispisuje "Greška" i od korisnika se zatraži ponovni upis trenutne lozinke.







2.6 Sekvencijski dijagram prikazuje kako izgleda proces brisanja korisničkog računa. Kada korisnik od aplikacije zatraži brisanje svog korisničkog računa, aplikacija od njega zatraži unos trenutne lozinke. Nakon što korisnik unese svoju trenutnu lozinku, aplikacija ju zajedno s ostalim korisničkim podacima šalje bazi podataka tražeći od nje provjeru ispravnosti lozinke. Ukoliko je lozinka neispravna, baza podataka će aplikaciji dojaviti neispravnost lozinke a onda će aplikacija korisniku prikazati ispis "Greška" i zatražiti od njega ponovni upis lozinke. Suprotno, ukoliko je lozinka ispravna, baza podataka će aplikaciji dojaviti ispravnost lozinke, a aplikacija će korisniku prikazati obrazac upozorenja s opcijama nastavi i otkaži. Ukoliko korisnik odabere opciju nastavi, on će time aplikaciji poslati potvrdu odluke brisanja svog računa, a aplikacija će od baze podataka zatražiti brisanje korisničkog računa. Nakon uspješnog brisanja, baza podataka će o tome informirati aplikaciju, a aplikacija će odjaviti korisnika. Ako korisnik odabere opciju otkaži, postupak brisanja korisničkog računa će se prekinuti.

Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe

Arhitektura sustava

Opis arhitekture

Stil arhitekture

Sustav je zasnovan na **klijent-poslužitelj** arhitektonskom stilu koji koristi **višeslojnu** arhitekturu. Ovaj pristup omogućuje jasno odvajanje odgovornosti, modularnost te lakše održavanje i nadogradnju sustava.

- Klijent-poslužitelj stil definira fizičku organizaciju sustava - klijenti, što su učenici i administratori, su odvojeni od poslužitelja koji obrađuje zahtjeve i pristupaju bazi podataka. Takva strukturiranost omogućuje jednostavnu interakciju i podjelu između *frontenda* i *backenda*.
- Višeslojna arhitektura očituje se u organizaciji unutar samog sustava kroz tri osnovna sloja: prezentacijski (React), poslovni (Node.js) i podatkovni (PostgreSQL baza). Time se osigurava visoka kohezija i niska povezanost.

Podsustavi

Podsustav za autentifikaciju i korisničke račune: zadužen je za prijavu korisnika u aplikaciju putem korisničke adrese, promjenu lozinke i brisanje korisničkog računa.

Podsustav za upravljanje sadržajem: omogućuje administratorima pretragu vanjskih rječnika te dodavanje cijelog ili samo jednog njegovog dijela u sustav, kao i njihovo uređivanje i brisanje.

Podsustav za učenje i evaluaciju: omogućuje prevođenje riječi s materinjeg na strani jezik, implementira logiku premještanja riječi među posudama na temelju točnosti odgovora, različite modove koji provjeravaju ispravnost napisanih riječi i izgovora te ih ocjenjuju na ljestvici od 1 do 10.

Podsustav za sigurnost i autorizaciju: osigurava da je pristup podacima i funkcionalnostima aplikacije ograničen na temelju korisničkih uloga, implementira zaštitu osobnih podataka korisnika i sigurnosnu prijavu u skladu s OAuth 2.0. standardom.

Podsustav za infrastrukturu i održavanje: odgovoran je za praćenje performansi, skalabilnost sustava te osiguravanje trajnosti i oporavka podataka.

Preslikavanje na radnu platformu

Projekt je preslikan na radnu (deploy) platformu koristeći Docker i Render, čime je osigurano stabilno okruženje za pokretanje aplikacije.

U Dockeru smo napravili image za frontend i backend. Docker slika sadrži sve potrebne pakete i definira način pokretanja aplikacije. Render koristi tu sliku i pokreće je kao web servis bez dodatne ručne konfiguracije.

Deploy je izveden preko platforme Render, gdje su pokrenuta dva ključna servisa:

- Web servis - pokreće Docker sliku koja obuhvaća i frontend i backend
- PostgreSQL baza podataka - hostana unutar Render DB servisa

Konfiguracijske varijable (API ključevi, URL baze, OAuth postavke) definirane su unutar Render environment settings, čime se osigurava sigurna i jednostavna integracija svih dijelova sustava.

Ovime je aplikacija u potpunosti preslikana na radnu platformu kroz jasno definirane kontejnere, stabilan hosting i automatsko upravljanje infrastrukturom.

Spremište podataka

Za spremanje podataka sustava odabrana je relacijska baza podataka **PostgreSQL** čime se osigurava transakcijska pouzdanost i visok integritet podataka. U relacijskom modelu podaci su organizirani u tablice (entitete) koje su međusobno povezane vanjskim ključevima što omogućuje efikasno pretraživanje i provedbu kompleksne poslovne logike. Odnosi između entiteta (primjerice User, Word i Dictionarie) prikazani su putem dijagrama klasa izrađenog u alatu *AstahUML* koji je poslužio za vizualizaciju strukture podataka i njihovih veza.

Mrežni protokoli

U aplikaciji se koriste sljedeći mrežni protokoli:

- **HTTPS** - omogućuje sigurnu i pouzdanu razmjenu podataka između prezentacijskog sloja (*React*) i logičkog sloja (*Node.js*).
- **TCP/IP** - osigurava pouzdan prijenos paketa podataka između korisničkog računala i poslužitelja.
- **SSL/TLS** - pruža enkripciju, osigurava zaštitu osobnih podataka i tokena tijekom prijenosa kroz mrežu.
- **OAuth 2.0** - koristi se za autentifikaciju i prijavu korisnika putem vanjskih servisa, kao što su Google, što omogućuje sigurno upravljanje pristupom bez pohrane lozinki u aplikaciji.
- **PostgreSQL protokol (SQL over TCP/IP)** - koristi se za komunikaciju između logičkog sloja (*Node.js*) i podatkovnog sloja (PostgreSQL baze). Osigurava slanje SQL upita i primanje rezultata putem sigurne TCP veze.

Globalni upravljački tok

Globalni upravljački tok predstavlja kružni tok podataka i informacija koji se pokreće interakcijom korisnika te se drži pravila višeslojne arhitekture - komunikacija samo sa susjednim slojem. Tok je centraliziran na poslužitelju gdje se odvija sva poslovna i sigurnosna logika.

Općeniti tok kroz aplikaciju izgleda ovako:

- Korisnik putem *frontend* sučelja izrađenog u *Reactu* pokreće određenu radnju (primjerice prijavljivanje u sustav) nakon čega se prema poslužitelju šalje HTTPS zahtjev s pripadajućim podacima.
- Pri prijavi korisnika, poslužitelj provodi autentifikaciju i generira JWT koji sadrži korisnički identifikator. Nakon toga, svaki idući zahtjev se autorizira isključivo provjerom valjanosti i digitalnog potpisa tog JWT-a čime se ukida potreba za ručnom provjerom korisničkog ID-a nad bazom podataka.
- Ako je JWT valjan i korisnik ima potrebne ovlasti, zahtjev se obrađuje - dohvaćaju se podaci iz baze podataka pomoću SQL upita (preko TCP/IP), a rezultat se proslijeđuje nazad u *Node.js*.
- Nakon obrade, *backend* generira odgovor u JSON formatu s traženim podacima ili statusima radnje (primjerice "uspješna prijava" ili "pogrešan odgovor") i šalje ih *Reactu* gdje se zatim oni prikazuju

krajnjem korisniku.

Ako zahtjev dolazi od korisnika s administratorskim ovlastima tada su omogućene dodatne funkcionalnosti poput dodavanja novih riječi. Ti se podaci također spremaju u bazu pomoću istog toka obrade.

Sklopovsko-programski zahtjevi

Za pravilno funkcioniranje aplikacije potrebni su sljedeći sklopovsko-programski zahtjevi:

- minimalno 2GB RAM-a i dvojezgreni procesor
- stabilna internetska veza zbog komunikacije s poslužiteljem putem HTTPS zahtjeva
- moderni web-preglednik s isključenim naprednim sigurnosnim postavkama kolačića radi ispravnog rada autentifikacije
- za mod provjere izgovora riječi potreban je mikrofoni i zvučnici na klijentu
- *Node.js* i *React* okruženje (verzija v.18 ili novija) i *PostgreSQL* baza podataka za koju je potrebno osigurati dovoljno prostora na disku (minimalno 1 GB)

Obrazloženje odabira arhitekture

Izbor arhitekture temeljen na principima oblikovanja

Pri oblikovanju arhitekture sustava primijenjeni su sljedeći temeljni principi oblikovanja:

- **visoka kohezija** - svaki sloj ima jasno definiranu odgovornost pa se tako *frontend* bavi prikazom podataka, *backend* obradom poslovne logike, a baza pohranom.
- **slaba povezanost** - slojevi međusobno komuniciraju preko REST API sučelja čime se omogućuje paralelni razvoj i testiranje.
- **hijerarhija i modularnost** - omogućuje jednostavno dodavanje novih funkcionalnosti bez potrebe za izmjenom postojećih dijelova koda.

Razmatrane alternative

Moguće druge alternativne arhitekture:

1. **Monolitna arhitektura:** jednostavnija implementacija zbog toga što objedinjuje *frontend* i *backend* u jednu cjelinu no istovremeno otežava održavanje i fleksibilnost. Također, zbog načina funkcioniranja CDN-a (*Content Delivery Network*) odziv bi bio znatno sporiji jer se učitava više sadržaja. Zbog navedenik informacija, ova arhitektura je naposljetku odbačena.
2. **Mikroservisi:** podrazumijeva razdvajanje sustava na niz malih neovisnih servisa koji međusobno komuniciraju, što omogućuje visoku skalabilnost i neovisno razvijanje komponenti sustava. Takav pristup je optimalan za opširnije i brzorastuće sustave, što isključuje ovu arhitekturu upravo zbog svoje presloženosti za trenutni opseg projekta.

Organizacija sustava na visokoj razini

- **Klijent-poslužitelj:** sustav se sastoji od klijentskog i poslužiteljskog dijela, izrađenog u *React* i *Node.js*, koji međusobno komuniciraju putem REST API sučelja. *Frontend* omogućuje interakciju korisnika s aplikacijom (unos podataka, slanje zahtjeva) dok *backend* te zahtjeve prima, obrađuje ih i komunicira s bazom podataka.

- **Baza podataka:** relacijska baza podataka *PostgreSQL* služi za pohranu i dohvat podataka - čuva informacije o korisnicima i prijavama te riječima i rječnicima unutar tablica koje su međusobno povezane ključevima. Definirani entiteti su: User, Word, DictionaryWord, Phrase, Dictionary, Language, i UserLanguage. Korištenje ove baze podataka osigurava integritet i dosljednost podataka.
- **Datotečni podsustav:** zadužen je za pohranu svih ostalih podataka koji nisu prikladni za direktnu pohranu u relacijskoj bazi podataka. Poslužitelj upravlja ovim datotekama te generira i vraća putanje klijentu za dohvat.
- **Grafičko sučelje:** korisničko sučelje je izgrađeno korištenjem *Reacta*. Dizajnirano je tako da bude intuitivno i responzivno omogućujući korisniku jednostavnu navigaciju.

Organizacija aplikacije

- **Frontend slojevi:**

- sloj za prikaz sadržaja korisniku - odgovoran je za vizualni prikaz podataka preko CCS-a i HTML-a.
- sloj za slanje zahtjeva poslužitelju - omogućuje komunikaciju između korisničkog sučelja i *backend*a na način da šalje HTTPS zahtjeve i prima odgovore.

- **Backend slojevi:**

- sloj za komunikaciju s bazom podataka - rukuje pohranom, dohvatom i ažuriranjem podataka u bazi.
- sloj za autentifikaciju korisnika - provjerava identitet korisnika i osigurava da ima odgovarajuće ovlasti (primjerice administrator može dodavati nove riječi dok običan korisnik, učenik, to ne može). Provjera ovlasti obavlja se čitanjem iz priloženog JWT tokena te se koristi OAuth 2.0.
- sloj za implementaciju logike aplikacije - središnji sloj *backend*a gdje se implementira sama logika i korišteni algoritmi, obrada podataka i sama koordinacija između baze i *frontend* zahtjeva.

Baza podataka

Koristili smo relacijsku PostgreSQL bazu podataka.

U bazi podataka nalazi se entitet USERS koji sprema podatke registriranih korisnika u atributima userId, email, password i role. Entitet DICTIONARIES sadrži sve rječnike koje su admini napravili, a sadrže attribute dictId, dictName, description i langId koji je povezan s entitetom LANGUAGES i označava jezik rječnika. U LANGUAGES su zapisani svi jezici dostupni za učenje s atributima langId, langName i langImg. Entitet WORDS sadrži sve riječi, strane i prijevode, atributi su mu wordId, word, audioFile, langId te translationId kojim je entitet povezan na samog sebe tako da podatke strane riječi poveže s podacima prijevoda. DICTWORD je entitet koji povezuje riječi i rječnike jer svaka strana riječ može biti u jednom ili više rječnika pa su i atributi ovdje samo dictId i wordId. USERDICT je entitet koji ima attribute userId i dictId te se njime bilježi koje rječnike je učenik krenuo učiti, dok se entitetom USERWORD bilježi u kojoj posudi je koja riječ naučena po kojoj metodi učenja za svakog korisnika u atributima userId, wordId, method i container. Svaka riječ, strana i prijevod, može imati više fraza koje ju opisuju, a te fraze se zapisuju u entitet PHRASES s atributima phrase i wordId.

Opis tablica

USERS

Atribut	Tip podataka	Opis
userId	int	primarni ključ
email	charvar (100)	unique
password	charvar (50)	
role	charvar (10)	

USERS opisuje svakog prijavljenog korisnika u aplikaciju. UserId se automatski određuje, email i password opisuje korisnik (password se sprema hashiran), a role definira ulogu korisnika te se automatski definira kao 'student' pri registraciji.

LANGUAGES

Atribut	Tip podataka	Opis
langId	int	primarni ključ
langName	charvar (50)	
langImg	charvar (256)	

U LANGUAGES se spremaju svi dostupni jezici za učenje. LangId se automatski određuje pri stvaranju jezika, langName je naziv jezika, a langImg put do slikovne datoteke koja predstavlja taj jezik.

DICTIONARIES

Atribut	Tip podataka	Opis
dictId	int	primarni ključ
dictName	charvar (100)	
langId	int	strani ključ
description	charvar (256)	

U DICTIONARIES se spremaju svi rječnici. DictId se automatski određuje pri stvaranju rječnika, dictName je naziv pridružen rječniku, langId je strani ključ iz LANGUAGES koji označava na kojem jeziku je rječnik, a description je kratki opis sadržaja rječnika.

WORDS

Atribut	Tip podataka	Opis
wordId	int	primarni ključ
word	charvar (100)	

Atribut	Tip podataka	Opis
audioFile	charvar (256)	
langId	int	strani ključ
translationId	int	strani ključ

WORDS je tablica koja sadrži sve strane riječi i hrvatske prijevode. WordId je identifikacija riječi te se automatski određuje pri stvaranju, word je riječ, audioFile je put do dokumenta u kojem je spremljen izgovor riječi, langId je strani ključ iz tablice LANGUAGES i označava jezik na kojem je riječ, a hrvatski prijevod strane riječi povezan je s translationId koji je zapravo wordId tog prijevoda (za hrvatske riječi, tj. kada je langId onaj od hrvatskog jezika, translationId je NULL).

DICTWORD

Atribut	Tip podataka	Opis
dictId	int	primarni ključ, strani ključ
wordId	int	primarni ključ, strani ključ

DICTWORD povezuje tablice DICTIONARIES i WORDS, povezuje riječ sa svakim rječnikom u kojem se nalazi. Sastoji se od dictId koji je strani ključ iz DICTIONARIES i wordId koji je strani ključ iz WORDS.

USERDICT

Atribut	Tip podataka	Opis
userId	int	primarni ključ, strani ključ
dictId	int	primarni ključ, strani ključ

USERDICT povezuje tablice DICTIONARIES i USERS, povezuje korisnika sa svakim rječnikom kojeg je započeo učiti. Sastoji se od dictId koji je strani ključ iz DICTIONARIES i userId koji je strani ključ iz USERS.

USERWORD

Atribut	Tip podataka	Opis
userId	int	primarni ključ, strani ključ
wordId	int	primarni ključ, strani ključ
container	int	

USERWORD povezuje tablice DICTIONARIES i WORDS, povezuje korisnika (userId) sa svakom riječi (wordId) koju je učio te u posudi (container) u kojoj se ta riječ za tog korisnika nalazi.

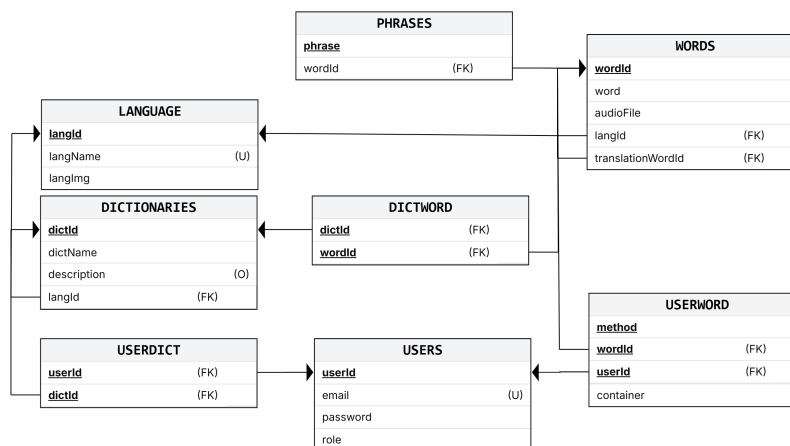
PHRASES

Atribut	Tip podataka	Opis
---------	--------------	------

Atribut	Tip podataka	Opis
phrase	charvar (256)	primarni ključ
wordId	int	strani ključ

Fraze koje opisuju riječ spremljene su u tablicu PHRASES, phrase sadrži frazu, a wordId je strani ključ iz tablice WORDS za onu riječ koju fraza opisuje.

Dijagram baze podataka



- ▶ ER diagram

Dijagram razreda

Ovaj dijagram razreda prikazuje glavne entitete sustava i njihove međusobne odnose.

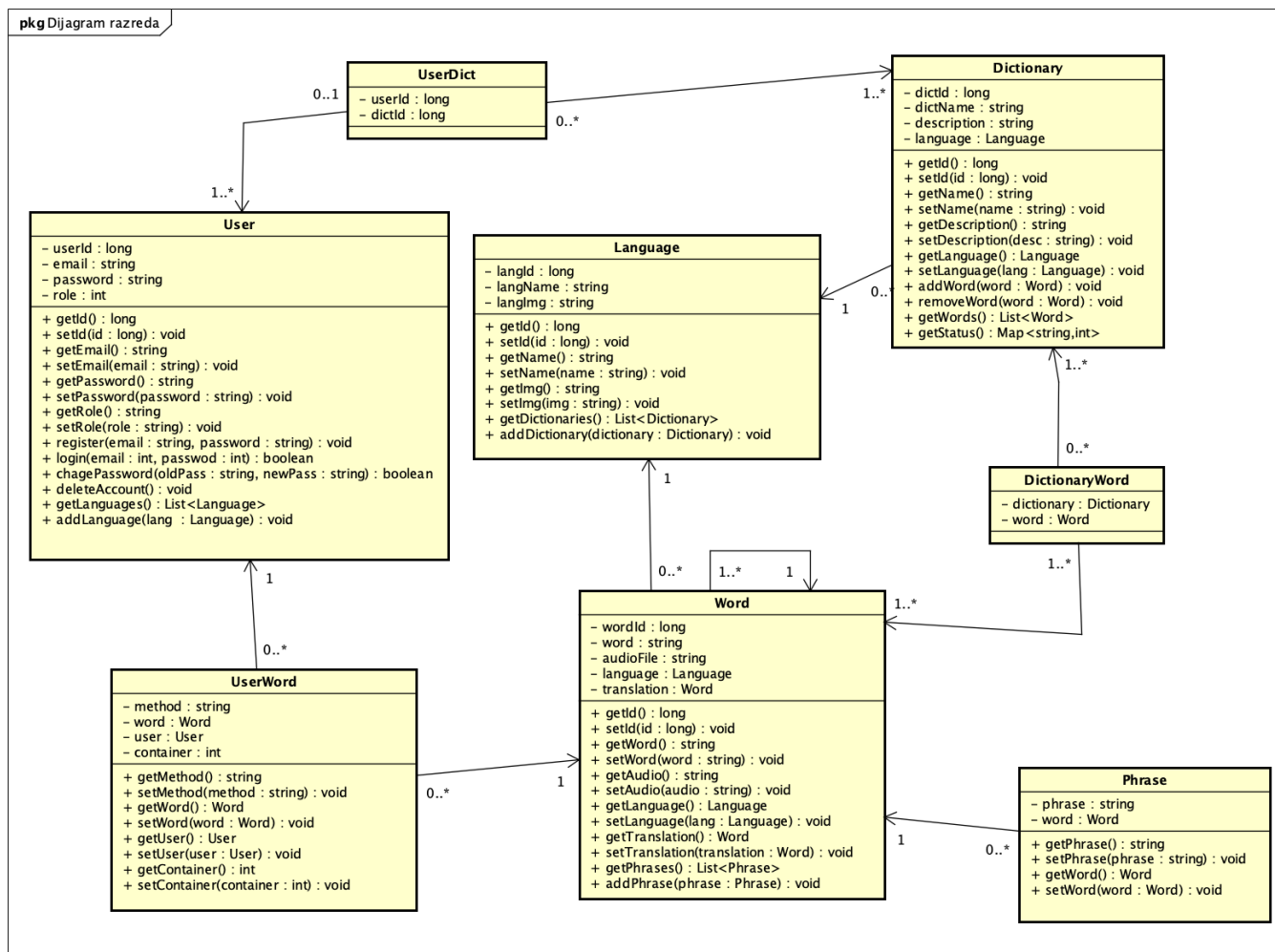
Atribiti klasa su privatni, zbog čega će svaka klasa (osim relacijskih) imati gettere i settere za te atribute.

Klasa **User** predstavlja korisnika koji se može registrirati, prijaviti, mijenjati lozinku i upravljati jezicima koje uči.

Klasa **Dictionary** sadrži riječi nekog jezika i posjeduje funkcije **addWord** i **removeWord** koje samo admin može pozvati. Također sadrži funkciju **getStatus** koja služi kao pomoćna funkcija koja provjera stanje riječnika (npr. može nam ispisati ukupni broj riječi, broj riječi bez prijevoda, broj riječi bez audioFile-a, itd.)

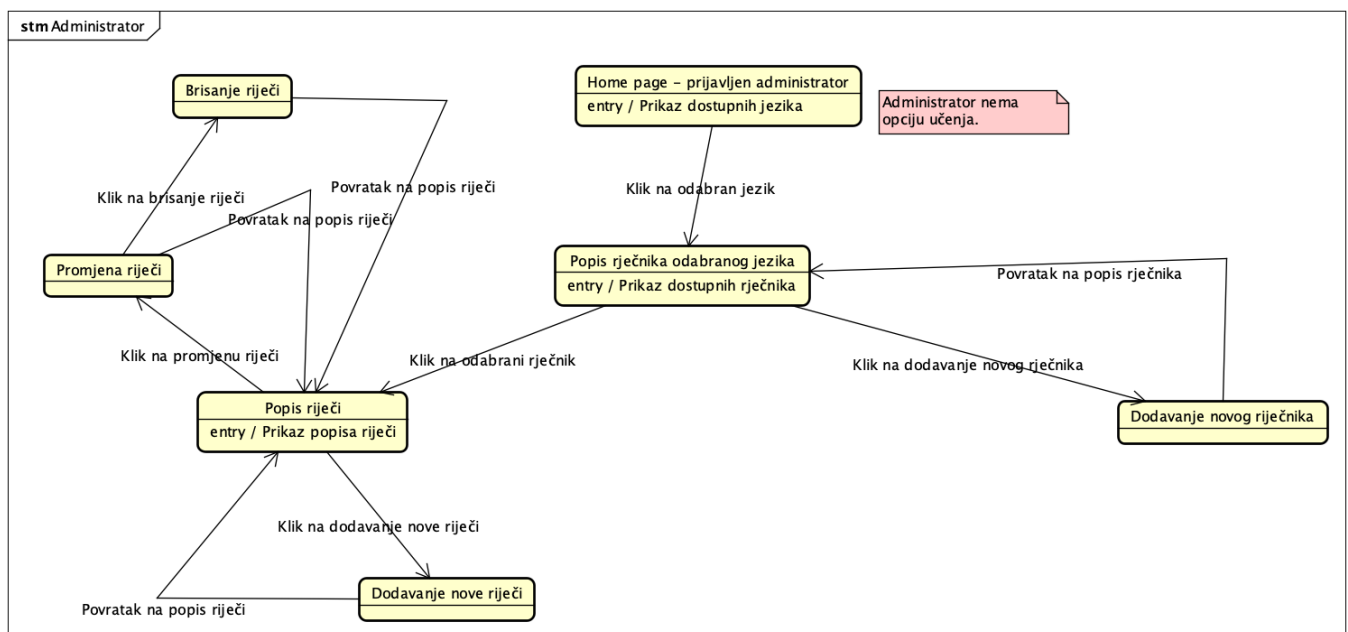
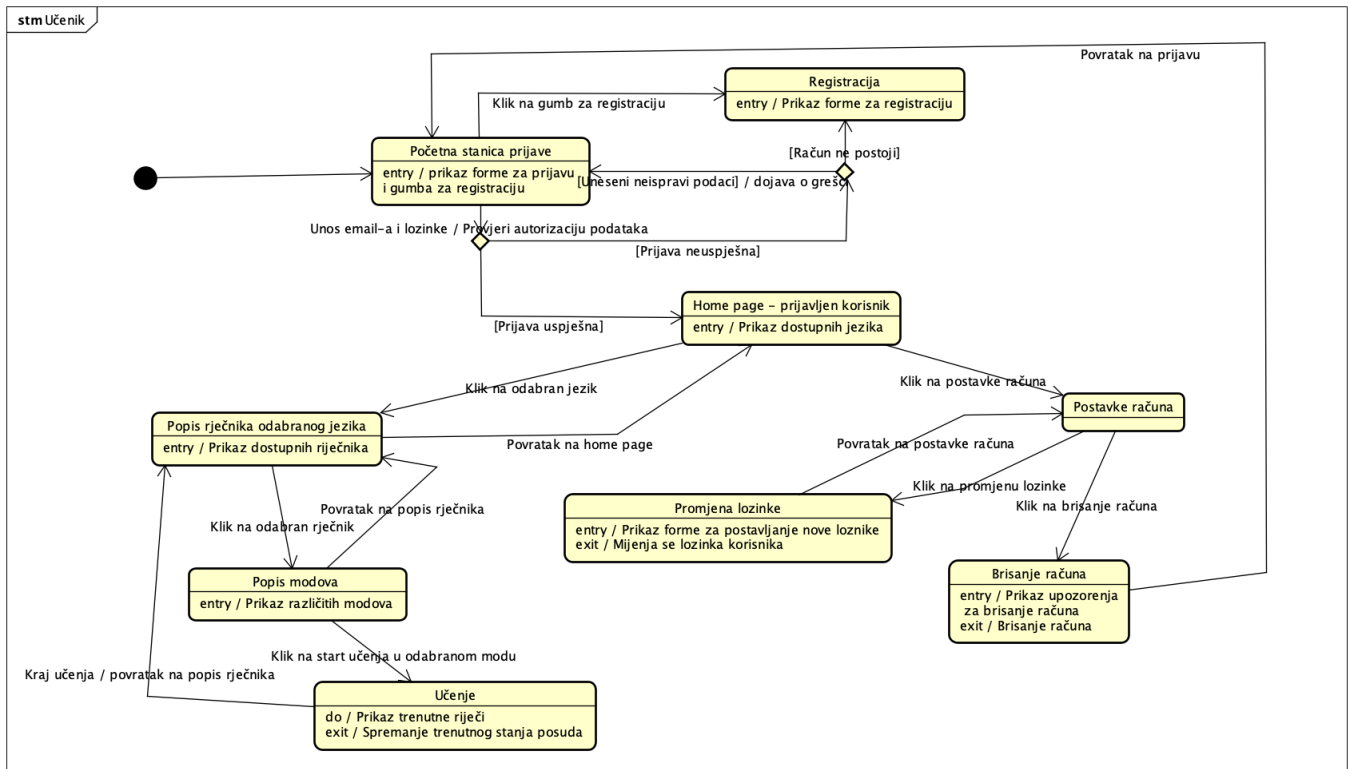
Klasa **Word** predstavlja riječ i njen prijevod, s povezanim frazama i izgovorom. Isto kao i za klasu **Disctionary**, sadrži funkcije koje samo admin može pokretati. Ima samoreferencijalnu vezu (translation: **Word**).

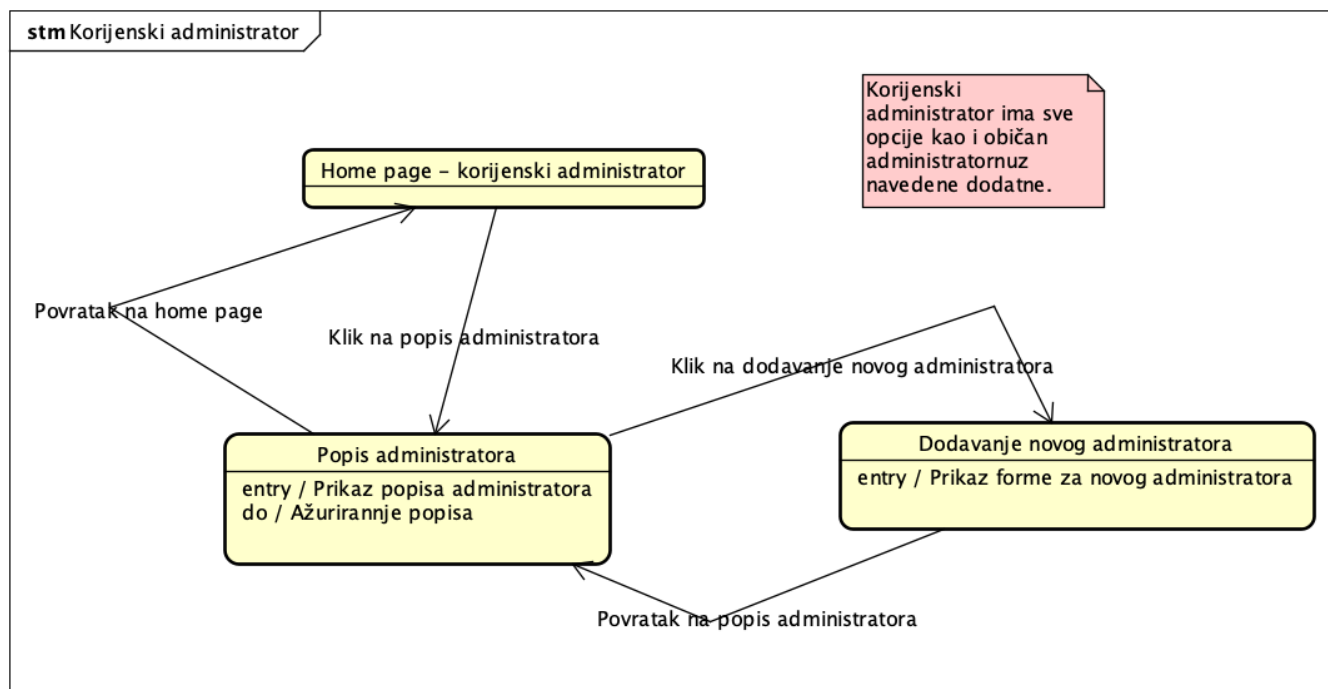
Klasa **UserWord** sadrži napredak učenja neke riječi za pojedinog User-a. Klasa, uz korisnika i riječi, sadrži metodu učenja i posudu u kojoj se riječ nalazi. Nakon što korisnik točno odgovori na pitanje, riječ se prebacuje na sljedeću posudu, tj. **container** se povećava za 1. Nakon što **container** pređe 5, korisnik je naučio riječ i bilježenje napretka za tu riječ se briše.



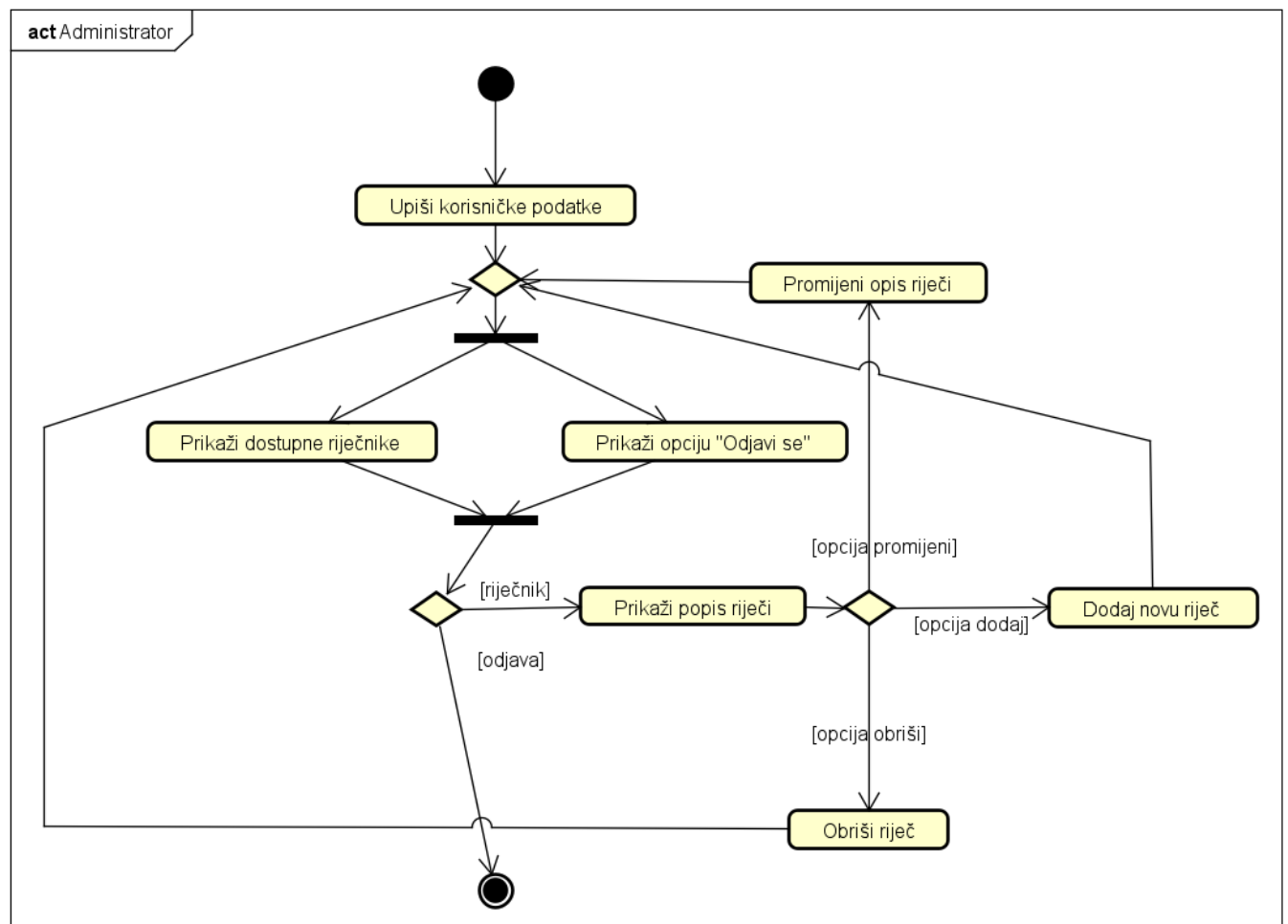
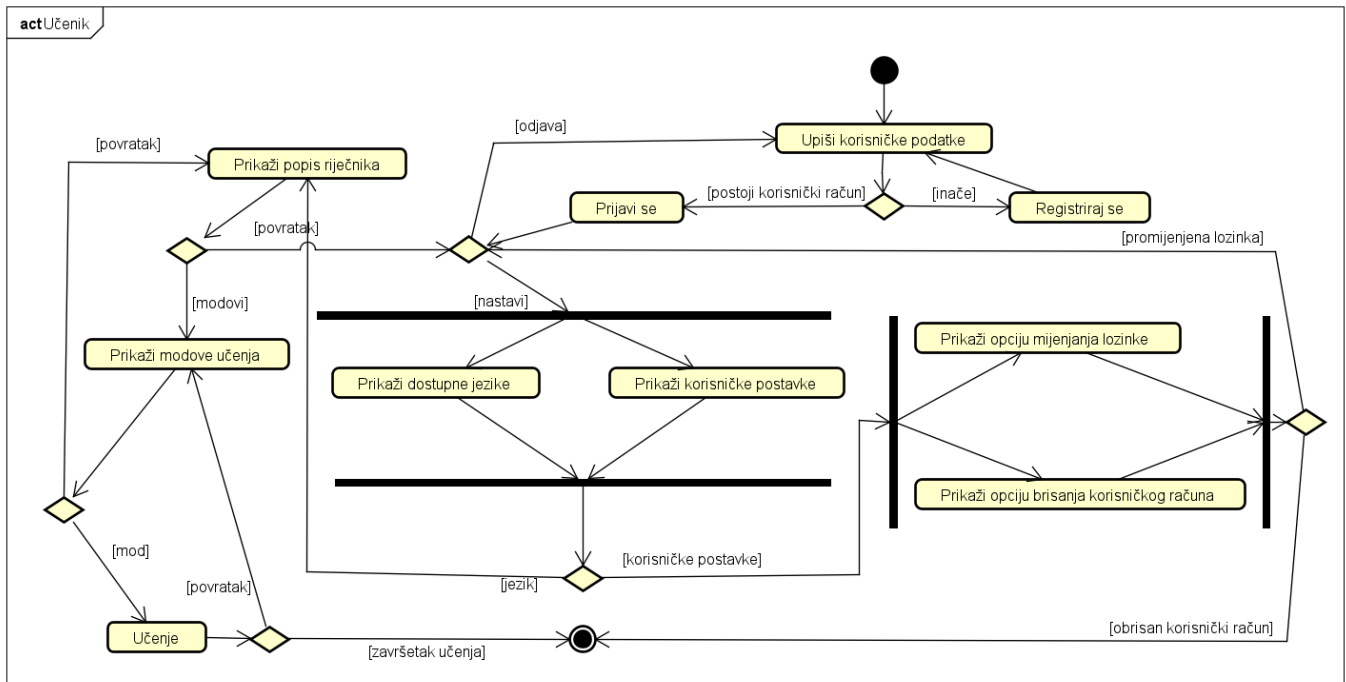
Dinamičko ponašanje aplikacije

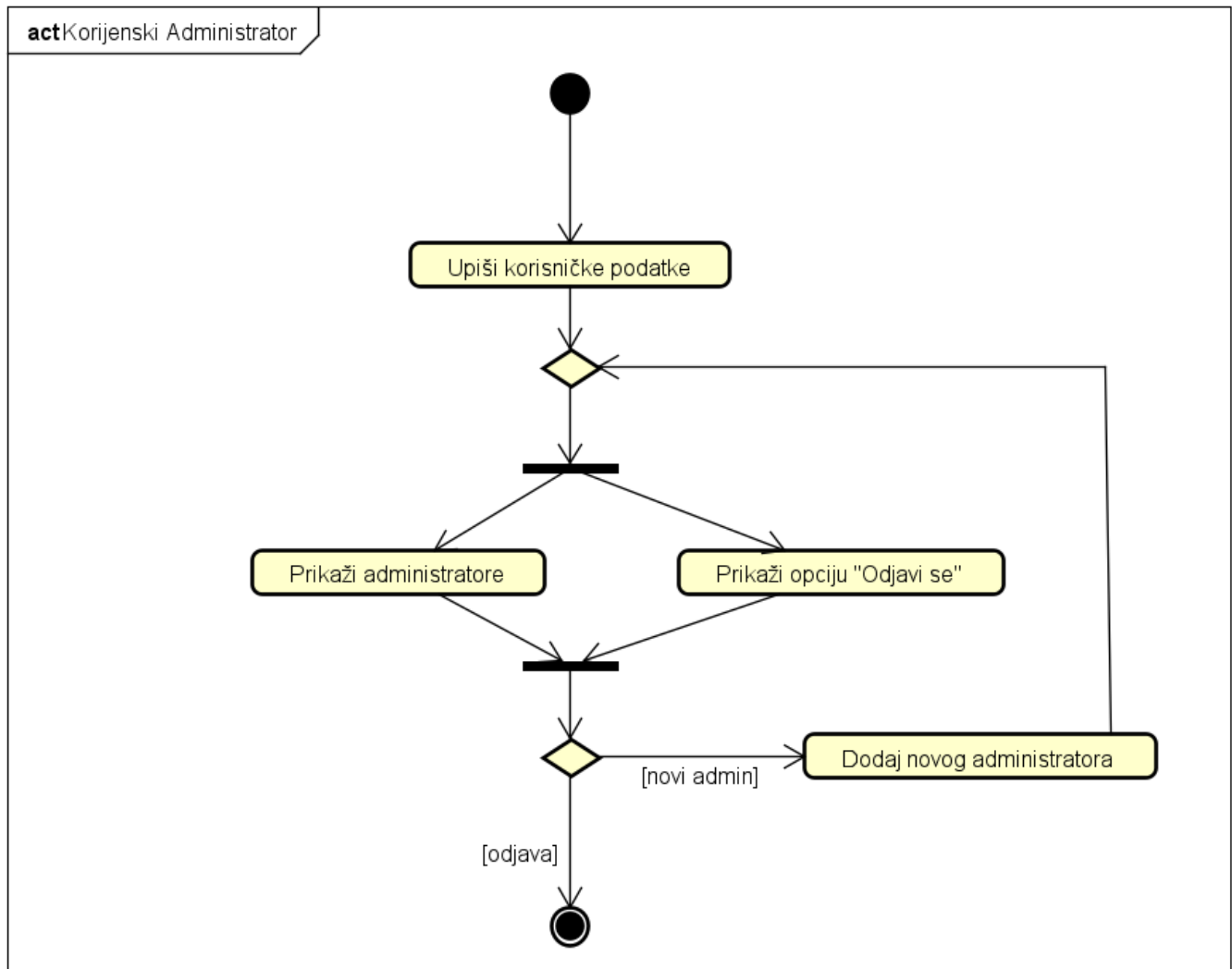
UML dijagrami stanja





UML dijagrami aktivnosti





Dijagram komponenata

Dijagram razmjestaja

Ispitivanje komponenti

****Ispitivanje sustava ****

Korištene tehnologije i alati

Opis prisutpa aplikaciji na javnom poslužitelju

...

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak	N. Mujo, E. Kitonić, D. Zaninović, E. Zebić	13.10.2025.
0.2	Analiza zahtjeva	N. Mujo, E. Kitonić, D. Zaninović, E. Zebić, F. Jengiće, F. Juričić	16.10.2025.
0.3	Dodatna analiza funkcionalnih zahtjeva	E. Kitonić, D. Zaninović	23.10.2025.
0.4	Dopunjen README	N. Mujo	24.10.2025.
0.5	Dopunjen opis projektnog zadatka	N. Mujo	25.10.2025.
0.6	UseCase dijagram	E. Zebić	27.10.2025.
0.7	Napravljen dio o dionicima sustava	F. Jengiće	28.10.2025.
0.8	Napravljeni potrebni sekvencijski dijagrami	F. Juričić	29.10.2025
0.9	Dodatna analiza nefunkcionalnih zahtjeva	E. Kitonić, D. Zaninović	30.10.2025.
1.0	Opis obrasca uporabe	E. Zebić	1.11.2025.
1.1	Dijagram stanja	E. Kitonić, D.Zaninović	5.11.2025
1.2	Napravljeni UML dijagrami aktivnosti	F. Juričić	9.11.2025
1.3	Dijagram baze podataka i tablice	E. Zebić	9.11.2025
1.4	Opis tablica baze podataka	E. Zebić	10.11.2025
1.5	Opis i obrazloženje odabira arhitekture	F. Jengiće	10.11.2025
1.6	Organizacija sustava na visokoj razini	F. Jengiće	11.11.2025
1.7	Dorađeni sklopovsko-programski zahtjevi	F. Jengiće	12.11.2025
1.8	Nadopunjavanje Opisa projektnog zadatka	N. Mujo	14.11.2025
1.9	Dodavanje Preslikavanje na radnu platformu	N. Mujo	14.11.2025

...

TO DO

Dnevnik sastajanja

Sastanak 15.10.2025.

- Vrijeme: 15:00 - 15:30.
- Prisustvovali: N. Mujo, F. Juričić, F. Jengić, E. Kitonić, E. Zebić, D. Zaninović
- Tema sastanka: Grupa se uživo sastala, napravila Discord server i uspješno raspodjelila uloge.

Uloge

- Frontend: Dora i Nikša
- Backend: Eva i Filip
- Database: Ema i Fran

Jedna osoba iz svake grupe piše dokumentaciju za dio koji pokriva.

Sastanak 16.10.2025.

- Vrijeme: 11:15 - 12:00
- Prisustvovali: N. Mujo, F. Juričić, F. Jengić, E. Kitonić, E. Zebić, D. Zaninović
- Tema sastanka: Odredili funkcijske i nefunkcijske zahtjeve, napravili plan rada za idući tjedan.

Odabrane tehnologije

- Dokumentacija i version control preko Github-a
- Najvjerojatniji kandidat je web aplikacija s backendom u node.js (možda i next.js ovisi o kompleksnosti projekta) + frontend React.js + spremanje podataka pomoću neke baze podataka (relacijska ili noSQL)

Najveći prioritet za početak nam je napraviti login page. Zato smo se svi zadužili napraviti login page do četvrtka 23.10., pri čemu je u planirano vrijeme uključeno i učenje tehnologija koje ćemo koristiti za njezinu implementaciju.

Sastanak 23.10.2025.

- Vrijeme: 11:30 - 12:15
- Prisustvovali: N. Mujo, F. Juričić, F. Jengić, E. Kitonić, E. Zebić, D. Zaninović
- Tema sastanka: Nadopunjavanje to do liste, wiki-a i Read Me-a. Rasprava o daljnjem radu na projektu.

Nakon razgovora sa asistentima smo odlučili prekinuti sa zadacima koji se tiču kodiranja aplikacije i odlučili smo se fokusirati na popunjavanje našeg GitHub repozitorija. To uključuje nadodavanje informacija u README, uspisivanje još funkcionalnih i nefunkcionalnih zahtjeva, dodavanje više poslova u TO DO listu te detaljniju dokumentaciju (koju smo do sada uradili te koju ćemo raditi nadalje). Na sastanku smo dodali zadatke za popunjavanje repozitorija te nam je cilj poboljšati ga do sljedećeg četvrtka. U međuvremenu ćemo se dogovoriti tko će koji dio popunjavati. Također nam je cilj za sljedeći tjedan uspostaviti alat koji ćemo koristiti

za korištenje sekvencijskih dijagrama te početi kreirati sekvencijske dijagrame za najvažnije funkcionalne zahtjeve našeg projekta.

Plan rada

Tablica aktivnosti

Dijagram pregleda promjena

Ključni izazovi i rješenja
