

Programsko inženjerstvo

Ak. god. 2023./2024.

Digitalni poster

Dokumentacija, Rev. 2.0

Grupa: *Posterized*

Voditelj: *Dominik Barukčić*

Datum predaje: 17. 11. 2023.

Nastavnik: *Miljenko Krhen*

Sadržaj

1 Dnevnik promjena dokumentacije	3
2 Opis projektnog zadatka	5
3 Specifikacija programske potpore	8
3.1 Funkcionalni zahtjevi	8
3.1.1 Obrasci uporabe	10
3.1.2 Sekvencijski dijagrami	17
3.2 Ostali zahtjevi	20
4 Arhitektura i dizajn sustava	21
4.1 Baza podataka	22
4.1.1 Opis tablica	23
4.1.2 Dijagram baze podataka	27
4.2 Dijagram razreda	28
4.3 Dijagram stanja	32
4.4 Dijagram aktivnosti	33
4.5 Dijagram komponenti	34
5 Implementacija i korisničko sučelje	36
5.1 Korištene tehnologije i alati	36
5.2 Ispitivanje programskog rješenja	37
5.2.1 Ispitivanje komponenti	37
5.2.2 Ispitivanje sustava	37
5.3 Dijagram razmještaja	38
5.4 Upute za puštanje u pogon	39
6 Zaključak i budući rad	48
Popis literature	49
Indeks slika i dijagrama	51

Dodatak: Prikaz aktivnosti grupe

52

1. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodatka	Autori	Datum
0.1	Napravljen predložak i dodani opisi obrazaca uporabe.	Barukčić	31.10.2023.
0.2	Napisani funkcionalni zahtjevi. Dodani dijagrami obrazaca uporabe.	Jukić, Samaržija	2.11.2023.
0.3	Dodan opis tablica i dijagram baze podataka.	Topolovec	05.11.2023.
0.4	Dodan jedan sekvencijski dijagram i opis.	Barić, Đunđek	08.11.2023.
0.5	Dodana arhitektura i dizajn sustava. Dodani ostali zahtjevi.	Topolovec, Barić	11.11.2023.
0.6	Dodan opis projektnog zadatka.	Božić, Barukčić	13.11.2023.
0.7	Dodan opis obrazaca uporabe 5 i ažuriran dijagram obrazaca uporabe.	Đunđek	13.11.2023.
0.8	Dodan drugi sekvencijski dijagram i opis	Barić	13.11.2023.
0.9	Dodan dijagram razreda	Topolevec, Đunđek	13.11.2023.
0.9.1	Dodan opis dijagrama razreda	Barić, Samaržija	14.11.2023.
1.0	Verzija samo s bitnim dijelovima za 1. ciklus	Barukčić	17.11.2023.
1.1	Korištene tehnologije i alati	Barukčić	30.12.2023.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	Opis promjene/dodatka	Autori	Datum
1.2	Upute za puštanje u pogon, Dijagram razmještaja	Barukčić	03.01.2024.
1.3	Dijagram komponenata	Barukčić	04.01.2024.
1.4	Dijagram aktivnosti	Barić	08.01.2024.
1.5	Dijagram stanja	Barukčić	09.01.2024.
1.6	Zaključak	Barukčić	09.01.2024.
1.7	Dijagram razreda rev. 2	Barić	16.01.2024.
1.8	Ispitivanje komponenti i sustava	Božić	*.01.2024.
1.9	Dijagrami pregleda promjena	Barukčić	18.01.2024.
2.0	Konačni tekst predložka dokumentacije	Barukčić	18.01.2024.

2. Opis projektnog zadatka

Razvijamo aplikaciju namijenjenu sudionicima stručne konferencije, s ciljem pojednostavljenja pregleda i ocjenjivanja radova. Naša misija je kreirati integrirano digitalno okruženje koje potiče interakciju između sudionika i autora te promiče znanstveni i stručni dijalog. U skladu s tim, implementiramo ključne korisničke zahtjeve kako bismo osigurali optimalno korisničko iskustvo i funkcionalnost sustava.

Potrebno je izgraditi sustav za naručitelje kojim bi se održane konferencije mogle pratiti putem aplikacije. Aplikacija je sustav preko kojeg se mogu pregledavati sadržaji poput postera i radova autora. Svaki autor može prijaviti svoj rad i administratoru konferencije proslijediti materijale potrebne za stavljanje svog rada u aplikaciju. Za svaki pojedini rad, korisnik ima priliku glasati u cilju biranja 3 najbolja rada. Tijekom održavanja konferencije, na aplikaciji postoji video prijenos u stvarnom vremenu te konferencije. Na kraju održavanja konferencije svaki korisnik može preuzeti fotografije s konferencije.

Ključni zahtjev je omogućiti istovremeni rad više korisnika u stvarnom vremenu. Naša aplikacija mora podržavati istovremene aktivnosti korisnika bez gubitka performansi. Na primjer, dok jedan sudionik pregledava rad, drugi može istovremeno davati svoje ocjene ili komentare. Ova funkcionalnost je ključna za dinamičnu interakciju i suradnju među sudionicima konferencije.

Još jedan važan zahtjev je brz pristup bazi podataka. Kada korisnici pristupe aplikaciji za pretraživanje radova ili unos svojih ocjena, odziv sustava mora biti brz i učinkovit. Ovaj zahtjev osigurava da korisnici ne doživljavaju frustracije zbog dugotrajnog čekanja na učitavanje informacija, što je posebno važno u okruženju konferencije gdje je vrijeme sudionika dragocjeno.

Naša aplikacija ima sigurnu, brzu i pouzdanu komunikaciju s bazom podataka i stabilnost veze sa serverom. Pružamo zaštitu povjerljivih podataka konferencije i korisnika te osiguravanje neprekidnog rada aplikacije.

Sustav je prilagođen za rad na mobilnim uređajima. Aplikacija je prilagodljiva i lako koristiva na različitim veličinama ekrana, od pametnih telefona do tableta. Sudionicima se omogućava da pristupe i koriste aplikaciju bilo kada i bilo gdje, što je posebno važno u današnjem mobilnom i povezanom svijetu.

Svi ovi zahtjevi su usmjereni na stvaranje korisničkog iskustva koje je učinkovito, intuitivno i prilagođeno potrebama sudionika stručne konferencije. Cilj nam je osigurati da aplikacija ispuní očekivanja korisnika pružajući inovativnu platformu za razmjenu znanstvenih i stručnih informacija.

Brojne su koristi ovog projekta, a one kojima naš sustav pridonosi su:

- Povećana Interaktivnost: Omogućuje aktivnije sudjelovanje posjetitelja kroz glasovanje i povratne informacije.
- Bolja Organizacija: Automatizira prijavu radova i administrativne procese, štedeći vrijeme i resurse.
- Pristupačnost: Digitalizacija sadržaja konferencije osigurava lakši pristup informacijama za sve sudionike.
- Ekološka Održivost: Smanjenje upotrebe papira kroz digitalne postere i materijale.
- Analitički Podaci: Skupljanje podataka o preferencijama sudionika, korisnih za buduće događaje.

Postoje razna slična rješenja ovakvih platformu poput Whova, EventMobi, i Attendify. Međutim, naša aplikacija se razlikuje specifičnim funkcijama poput prilagođenog glasovanja, notifikacija i integracije s lokalnim vremenskim uvjetima, što je prilagođeno specifičnim potrebama naše ciljane konferencije.

Korisnici ove aplikacije i zahtjevi svakog korisnika:

- Posjetitelji Konferencije: Željni pristupa sadržaju i interakcije.
- Autori Radova: Traže platformu za predstavljanje rada i dobivanje povratnih informacija.
- Organizatori Konferencija: Teže efikasnom upravljanju sadržajem i interakcijom sudionika.
- Sponzori: Žele promovirati svoje brendove putem aplikacije.

Aplikacija će biti dizajnirana modularno, omogućujući prilagodbu za različite tipove konferencija, broj sudionika, i specifične zahtjeve organizatora. Projekt uključuje razvoj i implementaciju aplikacije, testiranje funkcionalnosti, te suradnju s krajnjim korisnicima za povratne informacije i unaprjeđenja.

Postoje razne nadogradnje našeg projektnog zadatka, a neka od njih su sinteza sustava s ovim tehnologijama:

- Umjetna Inteligencija: Personalizirani prijedlozi sesija bazirani na interesima korisnika.
- Virtualna Stvarnost: Mogućnost virtualnog obilaska konferencije za korisnike koji ne mogu prisustvovati fizički.
- Integracija s Društvenim Mrežama: Olakšavanje dijeljenja sadržaja i povećanje vidljivosti konferencije.

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Naručitelj
2. Administrator
3. Neregistrirani korisnik
4. Registrirani korisnik
5. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Administrator (inicijator) može:
 - (a) dodati autore, radove, pokrovitelje konferencije i lokacije
 - (b) uređivati podatke konferencije
 - (c) započeti i završiti konferenciju
 - (d) postaviti fotografije koje su slikane tijekom konferencije u galeriju
 - (e) objaviti rezultate konferencije
2. Neregistrirani korisnik (inicijator) može:
 - (a) pristupiti sustavu
 - (b) unijeti lozinku za pristup konferenciji
 - (c) registrirati se u sustav
3. Registrirani korisnik (inicijator) može:
 - (a) prijaviti se u sustav
 - (b) pregledavati promotivne materijale pokrovitelja konferencije
 - (c) pregledavati radove sudionika
 - (d) glasati za jedan rad
 - (e) uz pomoć direktnog video prijenosa pratiti trenutna događanja u glavnoj konferencijskoj dvorani

- (f) pregledavati i spremati fotografije iz galeriji
- (g) vidjeti mjesto održavanje konferencije i podatke o trenutnim vremenskim uvjetima
- (h) vidjeti konačne rezultate konferencije

4. Baza podataka (sudionik) obavlja:

- (a) pohranjuje sve podatke o korisnicima
- (b) pohranjuje sve podatke o autorima i njihovim radovima
- (c) pohranjuje sve podatke o konferencijama
- (d) pohranjuje sve podatke o mjestu održavanja
- (e) pohranjuje sve podatke o fotografijama slikanim tijekom konferencije i pokroviteljima

3.1.1 Obrasci uporabe

UC1 - Stvaranje konferencije i dodjela administratora

- **Glavni sudionik:** Super Administrator
- **Cilj:** Stvoriti konferenciju i dodijeliti je administratoru.
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Super Administrator stvara konferenciju.
 2. Pridjeljuje ovlasti nad konferencijom odabranom administratoru.
- **Opis mogućih odstupanja:**
 - 2.a Administrator ne postoji
 1. Super Administrator stvara Administratora i ponavlja postupak.

UC2 - Unos podataka o konferenciji

- **Glavni sudionik:** Administrator
- **Cilj:** Dodati podatke o konferenciji u bazu podataka.
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Administrator unosi podatke o konferenciji i mjestu održavanja.
 2. Podaci se pohranjuju u bazu podataka.

UC3 - Unos podataka o autoru i radu

- **Glavni sudionik:** Administrator
- **Cilj:** Dodati podatke o autoru i radu u bazu podataka.
- **Sudionici:** Baza podataka
- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Administrator unosi podatke o autoru i radu preko grafičkog sučelja.
 2. Aplikacija dodaje podatke u bazu podataka.

UC4 - Početak konferencije

- **Glavni sudionik:** Administrator
- **Cilj:** Omogućiti korisnicima pristup konferenciji, pregled radova i glasovanje.

- **Sudionici:** Baza podataka
- **Preduvjet:** Uneseni su svi potrebni podaci o konferenciji.
- **Opis osnovnog tijeka:**
 1. Administrator započinje konferenciju.

UC5 - Dodavanje fotografija u galeriju

- **Glavni sudionik:** Administrator
- **Cilj:** Dodati fotografije konferencije u galeriju.
- **Sudionici:** Baza podataka
- **Preduvjet:** Konferencija postoji.
- **Opis osnovnog tijeka:**
 1. Administrator tijekom i nakon konferencije dodaje fotografije pomoću grafičkog sučelja.
 2. Aplikacija pohranjuje osnovne podatke o fotografijama u bazu podataka.

UC6 - Registracija

- **Glavni sudionik:** Neregistrirani korisnik
- **Cilj:** Registrirati se u sustavu kako bi se omogućio pristup svim funkcionalnostima aplikacije.
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik ima dobiven pin.
- **Opis osnovnog tijeka:**
 1. Korisnik pristupa registracijskoj stranici aplikacije.
 2. Unosi svoje osobne podatke za registraciju.
 3. Sustav provjerava podatke i stvara račun.
 4. Korisnik prima potvrdu o uspješnoj registraciji i pristupa stranici konferencije.
- **Opis mogućih odstupanja:**
 - 3.a Uneseni podaci nepotpuni ili neispravni.
 1. Sustav prikazuje upozorenje i traži ispravak podataka.

UC7 - Prijava u sustav

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Prijaviti se u sustav kako bi se pristupilo konferenciji.
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik ima korisnički račun.

- **Opis osnovnog tijeka:**

1. Korisnik pristupa stranici za prijavu u aplikaciju.
2. Unosi svoje korisničko ime i lozinku.
3. Sustav provjerava unesene podatke i omogućuje pristup konferenciji.

- **Opis mogućih odstupanja:**

- 3.a Uneseni podaci su netočni.
 1. Sustav prikazuje upozorenje o neuspjeloj prijavi i traži ponovni upis podataka.

UC8 - Pregled radova sudionika

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Pregledati radove sudionika konferencije.
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav.
- **Opis osnovnog tijeka:**
 1. Korisnik pristupa dijelu aplikacije koji prikazuje sve dostupne radove sudionika.
 2. Korisnik pregledava pojedinačne radove.

UC9 - Glasovanje za rad

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Dati svoj glas određenom posteru na konferenciji.
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustavu i konferencija je u tijeku.
- **Opis osnovnog tijeka:**
 1. Korisnik odabire rad za koji će glasati.
 2. Sustav bilježi glas korisnika za odabrani poster.
- **Opis mogućih odstupanja:**
 - 2.a Korisnik je već glasovao.
 1. Sustav ne bilježi glas i pokazuje upozorenje da je moguće samo jednom glasovati.

UC10 - Pregled fotografija u galeriji

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Pregledati fotografije snimljene tijekom konferencije.
- **Sudionici:** Baza podataka

- **Preduvjet:** Korisnik je prijavljen u sustav.
- **Opis osnovnog tijeka:**
 1. Korisnik pristupa dijelu aplikacije koji prikazuje dostupne fotografije s konferencije.
 2. Korisnik pregledava dostupne fotografije.

UC11 - Spremanje fotografija na svoj uređaj

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Spremiti odabrane fotografije na svoj uređaj.
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav i pregledava fotografije.
- **Opis osnovnog tijeka:**
 1. Korisnik pregledava fotografije.
 2. Korisnik odabire fotografiju koju želi spremiti i preuzima ju na svoj uređaj.

UC12 - Direktno video praćenje konferencije

- **Glavni sudionik:** Registrirani korisnik
- **Cilj:** Pratiti video prijenos događanja konferencije u stvarnom vremenu.
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen u sustav i ima stabilnu internetsku vezu.
- **Opis osnovnog tijeka:**
 1. Registrirani korisnik pristupa opciji "Direktno video praćenje" u aplikaciji.
 2. Sustav prikazuje video prijenos za praćenje.
- **Opis mogućih odstupanja:**
 - 2.a Korisnik nema stabilnu internetsku vezu.
 1. Sustav obavještava korisnika da je potrebna stabilna internetska veza za praćenje video prijensa.

UC13 - Završetak konferencije

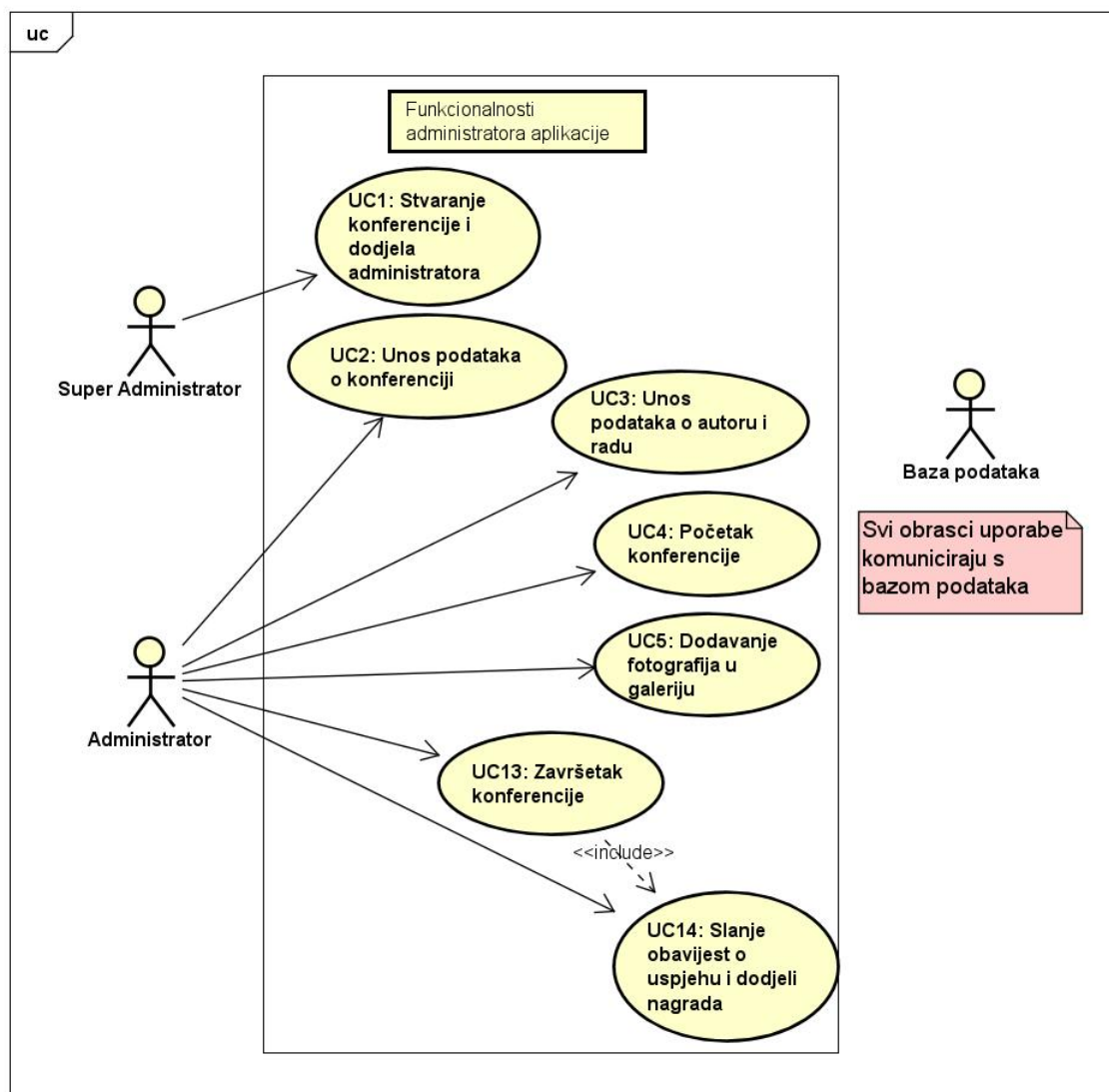
- **Glavni sudionik:** Administrator
- **Cilj:** Završiti konferenciju i prekinuti mogućnost glasovanja.
- **Sudionici:** Baza podataka

- **Preduvjet:** -
- **Opis osnovnog tijeka:**
 1. Administrator završava konferenciju.
 2. Zbrajaju se glasovi.

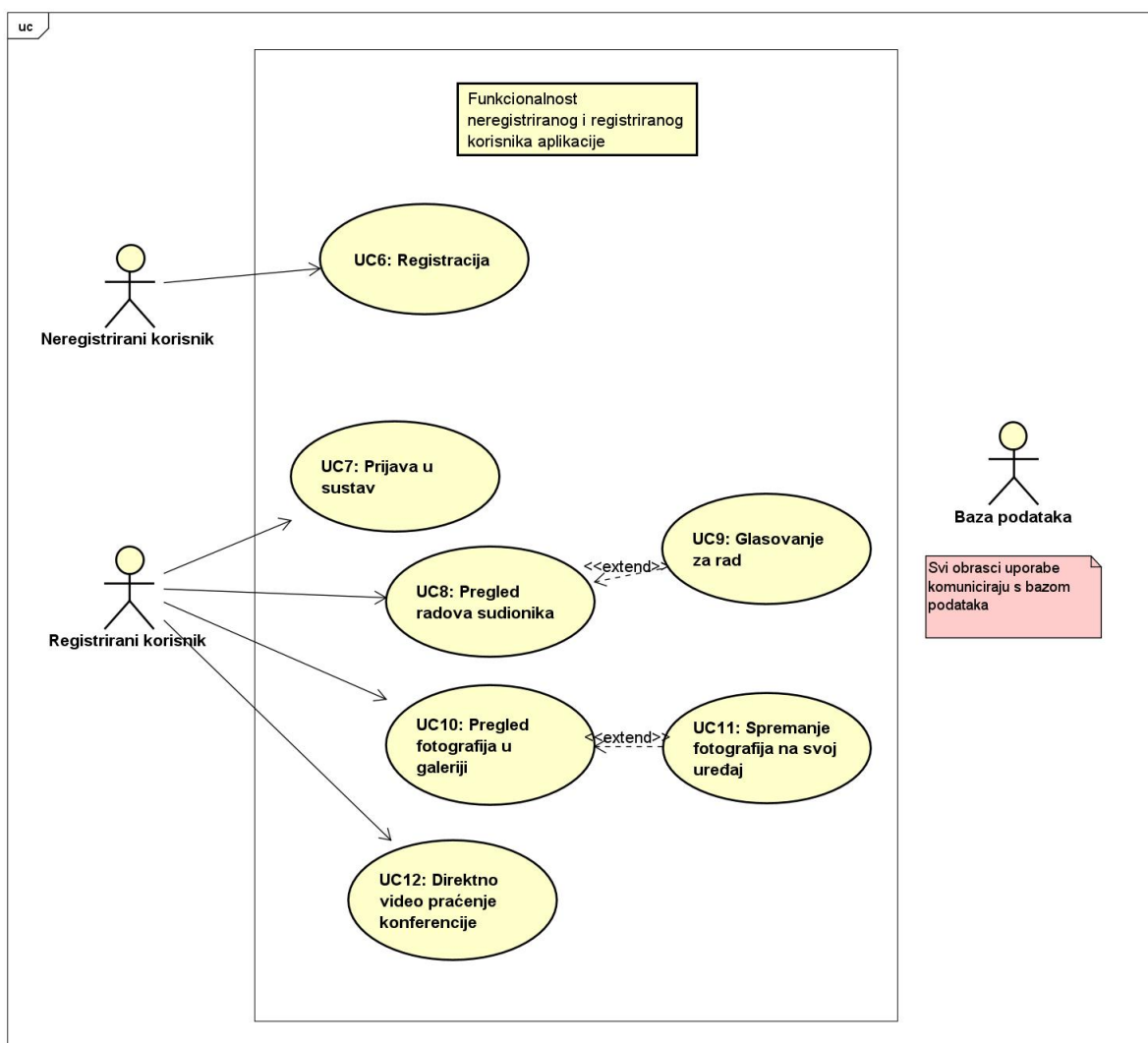
UC14 - Slanje obavijest o uspjehu i dodjeli nagrada

- **Glavni sudionik:** Administrator
- **Cilj:** Poslati e-mail autorima o njihovom uspjehu i pozivnicu na dodjelu nagrada za prva tri rada svim korisnicima i autorima.
- **Sudionici:** Baza podataka
- **Preduvjet:** Konferencija je završila.
- **Opis osnovnog tijeka:**
 1. Administrator šalje e-mail svim autorima u kojem ih obavještava o njihovom rangu prema zabilježenim glasovima i o mjestu i vremenu dodjele nagrada za prva tri rada.
 2. Administrator obavještava sve korisnike o mjestu i vremenu dodjele nagrada.

Dijagrami obrazaca uporabe



Slika 3.1: Dijagram obrazaca uporabe - administrator



Slika 3.2: Dijagram obrazaca uporabe - korisnici

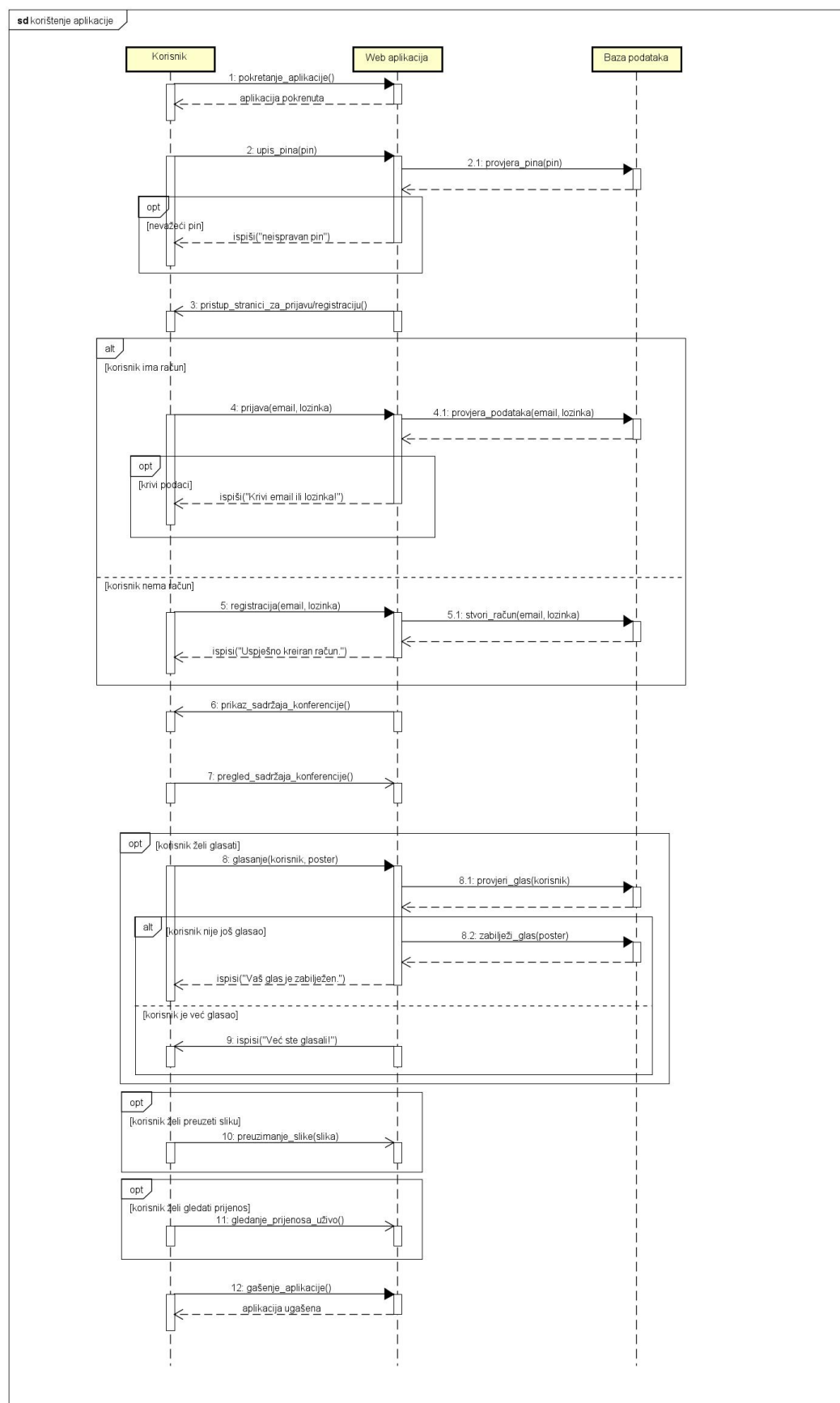
3.1.2 Sekvencijski dijagrami

Korisnik pokreće aplikaciju. Ima ograničen pristup sadržajima aplikacije sve dok ne upiše pin koji je dodijeljen sudionicima konferencije. Upisani pin se zatim provjerava i korisnik nastavlja na stranicu za prijavu/registraciju.

Kad korisnik ima stvoreni račun, prijavljuje se u aplikaciju pomoću adrese elektroničke pošte i lozinke. Prijavljeni korisnik onda može pristupiti sadržaju konferencije.

Za vrijeme sudjelovanja na konferenciji korisnik može pregledavati sadržaj konferencije i radove. Korisnik može glasovati samo za jedan rad i pratiti video prijenos trenutnih događanja u glavnoj konferencijskoj dvorani u realnom vremenu. Također ima dostupno pregledavanje i preuzimanje slika s konferencije - tijekom i nakon što konferencija završi. Korisnik može ugasiti aplikaciju kad god ima potrebu za tim.

Dijagram je prikazan na sljedećoj stranici.

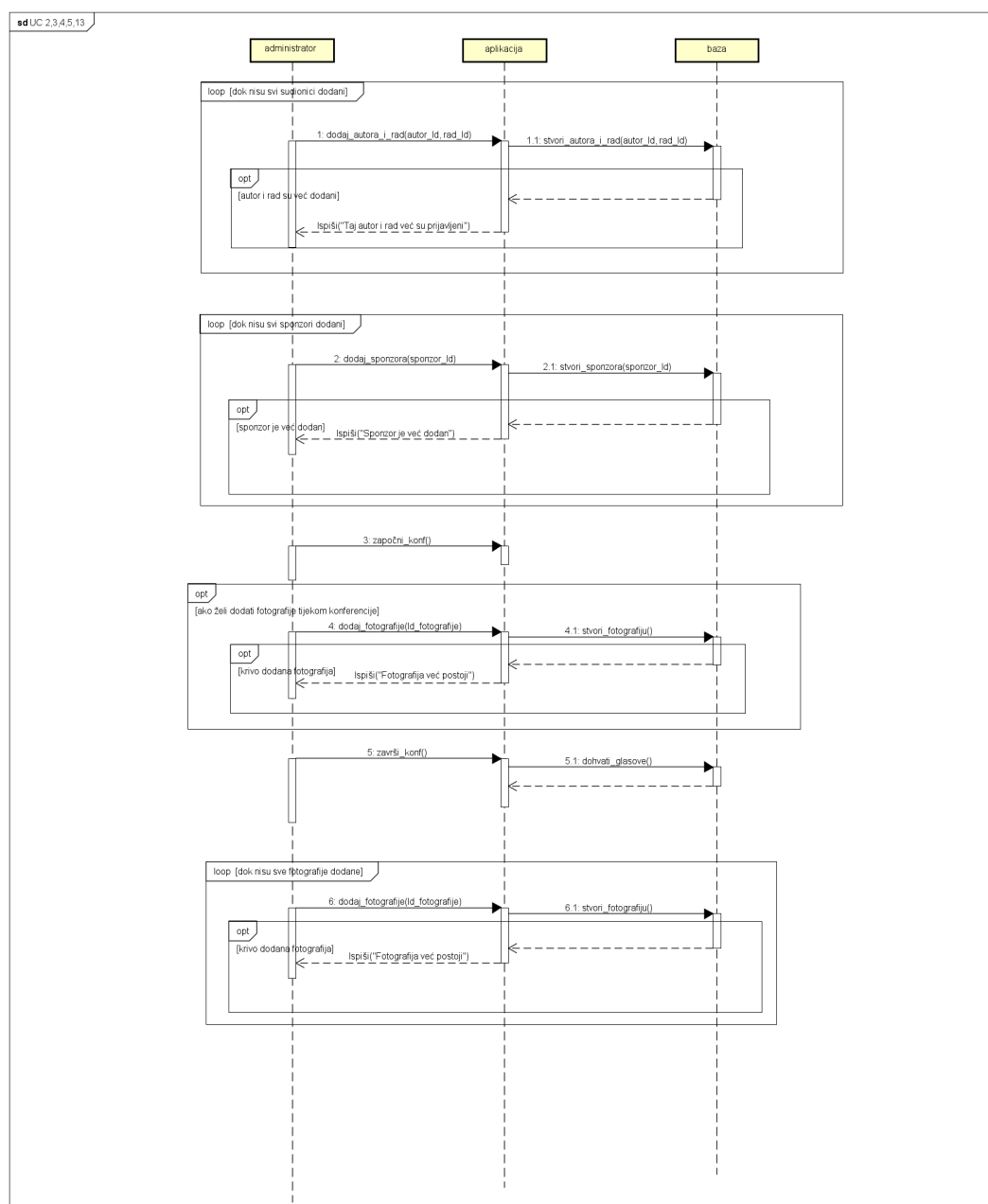


Slika 3.3: Sekvencijski dijagram korištenja aplikacije

Administrator dodaje autore i njihove radove prije početka konferencije. Nado-
daje ih dok ne doda sve sudionike (autore) i njihove radove. Zatim dodaje sponzore
konferencije dok ih sve ne upiše.

Administrator započinje konferenciju. Tijekom konferencije administrator ima
mogućnost dodavanja fotografija te konferencije. Administrator završava konfe-
renciju.

Administrator dodaje fotografije i nakon konferencije.



Slika 3.4: Sekvencijski dijagram korištenja aplikacije od strane administratora

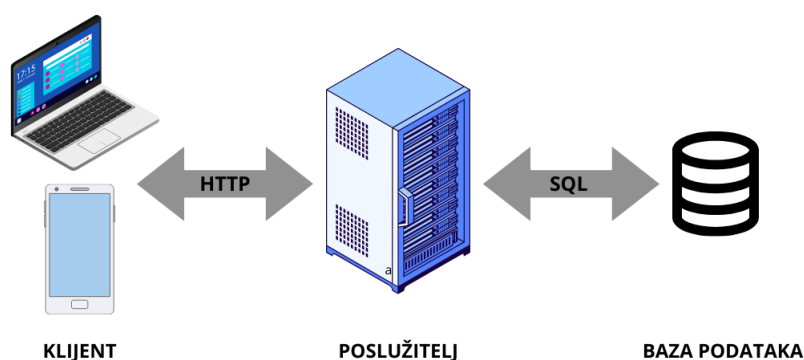
3.2 Ostali zahtjevi

- Sustav mora omogućiti istovremeni rad više korisnika u stvarnom vremenu
- Sustav i korisničko sučelje moraju podržavati znakovlje hrvatske abecede (dijakritičke znakove) prilikom prikazivanja tekstualnog sadržaja te unosa
- Pristupanje bazi podataka, tj. izvršavanje dijela programa u kojem se pristupa bazi podataka ne smije trajati duže od nekoliko sekundi
- Sustav treba biti implementiran kao web aplikacija koristeći objektno - orijentirane jezike
- Neispravnim korištenjem korisničkog sučelja, ne smije se narušiti funkcionalnost i rad sustava
- Sustav treba biti jednostavan i intuitivan za korištenje, odnosno korisnik ga mora moći koristiti bez korištenja (opširnih) uputa
- Prilikom nadogradnje sustava, ne smiju biti narušene njegove postojeće funkcionalnosti
- Veza s bazom podataka mora biti dobro zaštićena, brza i otporna na vanjske greške
- Sustav je responzivan na mobilnim uređajima
- Pristup sustavu treba biti omogućen iz javne mreže preko HTTPS protokola

4. Arhitektura i dizajn sustava

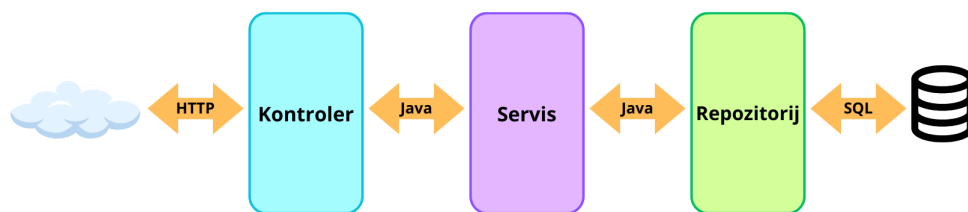
Na arhitekturu sustava najveći utjecaj imali su principi oblikovanja: Podijeli pa vladaj, Zadrži razinu apstrakcije te Oblikuj za prenosivost. Princip Podijeli pa vladaj očituje se u podijeli sustava na manje komponente radi povećane razumljivosti te lakše zamjene dijelova i ponovnog korištenja. Princip Zadrži razinu apstrakcije omogućava razumijevanje poante podsustava bez poznavanja nepotrebnih detalja. Korištenje Java kao objektno orijentiranog programskog jezika omogućuje nam upotrebu razreda, podatkovnih apstrakcija koje sadrže proceduralne apstrakcije (metode). Osim upotrebe razreda Java omogućuje rad na više platformi, čime je osigurana prenosivost.

Organizacija sustava s najviše razine apstrakcije je klijent-poslužitelj-baza podataka. Klijenta predstavlja preglednik weba koji omogućuje korisniku slanje zahtjeva poslužitelju protokolom HTTP (*engl. Hyper Text Transfer Protocol*). Poslužitelj je server koji poslužuje te zahtjeve, prosljeđuje ih web aplikaciji koja se pokreće preko poslužitelja te vraća odgovore koji se prikazuju preko klijenta (preglednika). Podaci su spremljeni u bazi podataka te joj po potrebi pristupa web aplikacija koristeći SQL upite.



Slika 4.1: Organizacija sustava s najviše razine apstrakcije

Arhitektura aplikacije je troslojna. Prvi sloj je **kontroler** koji prima zahtjeve, poziva odgovarajuće metode drugog sloja **servisa**, te na kraju vraća odgovore. Servis sadrži poslovnu logiku aplikacije, a za pristup podacima koristi treći sloj **repozitorij** koji komunicira s bazom podataka.



Slika 4.2: Organizacija aplikacije

Za izradu naše aplikacije korišten je Java Spring Boot okvir koji koristi MVC (*engl. Model-View-Controller*) oblikovni obrazac u kojem je poslužitelj organiziran u tri dijela u cilju razdvajanja nadležnosti. **Kontroler** prima zahtjeve koje prosljeđuje modelu te upravlja modelom i pogledom. **Model** je zadužen za obradu i dohvat podataka te komunicira s bazom podataka. **Pogled** prezentira dostavljene podatke.

4.1 Baza podataka

U aplikaciji će baza podataka bit prikazana relacijskim modelom podataka. Objekti relacijskog modela su relacije, a svaka ima jedinstveno ime unutar sheme baze podataka. Relacija je tablica čiji se imenovani stupci nazivaju atributi, a redci n-torke. Ključ entiteta je skup atributa koji jednoznačno određuje entitet. U našem sustavu entiteti baze podataka su:

- Rad
- Osoba
- Konferencija
- Prisutan_na
- Mjesto
- Fotografija
- Pokrovitelj
- Pokrovitelj_na
- Password_token

4.1.1 Opis tablica

Entitet **Rad** sadrži sve važne informacije o radu. Sadrži attribute: ID rada, naziv postera, naziv prezentacije, naslov rada, ID autora, ID konferencije na koju je prijavljen, osvojeni plasman, url postera i prezentacije i ukupan broj osvojenih glasova na toj konferenciji. Atributi naziv prezentacije, url prezentacije i plasman su opcionalni te stoga mogu poprimiti vrijednost null. Entitet Rad u binarnoj je vezi (*Many-to-One*) s entitetom Konferencija i u vezi (*Many-to-One*) s entitetom Osoba.

Rad		
id	SERIAL	jedinstveni identifikator rada
nazivPoster	VARCHAR	naziv postera koji prikazuje rad
nazivPptx	VARCHAR	naziv prezentacije koja prikazuje rad, može biti null
naslov	VARCHAR	naslov rada
ukupnoGlasova	INT	ukupan broj osvojenih glasova na konferenciji
urlPoster	VARCHAR	url postera koji prikazuje rad
urlPptx	VARCHAR	url prezentacije koja prikazuje rad, može biti null
plasman	INT	plasman na konferenciji, može biti null
idKonf	SERIAL	jedinstveni identifikator konferencije
idAutor	SERIAL	jedinstveni identifikator autora

Entitet **Osoba** sadrži informacije o autorima, korisnicima te adminima. Sadrži attribute: ID osobe, email, ime i prezime, lozinka (u slučaju da se radi o autoru koji ujedno nije i korisnik bit će null) i uloga koji može poprimiti vrijednosti "admin", "korisnik" ili "autor". Entitet Osoba u binarnoj je vezi s entitetom Rad (*One-to-Many*), s entitetom Konferencija (*One-to-Many*), s entitetom Konferencija (*Many-to-Many*) i s entitetom Password_token (*One-to-One*).

Osoba		
id	SERIAL	jedinstveni identifikator osobe
email	VARCHAR	email osobe
ime	VARCHAR	ime osobe
prezime	VARCHAR	prezime osobe
lozinka	VARCHAR	hash lozinke korisnika ili admina
uloga	VARCHAR	uloga osobe

Entitet **Konferencija** sadrži informacije o stručnoj konferenciji koja će se održati. Sadrži attribute: ID konferencije, poveznica na video prijenos konferencije, pin za ulazak na konferenciju, vrijeme početka i vrijeme kraja konferencije, ID admina zaduženog za konferenciju i adresu i poštanski broj mjesta u kojem se održava konferencija. Entitet Konferencija u binarnoj je vezi s entitetom Rad (*One-to-Many*), u binarnoj vezi (*Many-to-One*) i (*Many-to-Many*) s entitetom Osoba, u vezi (*Many-to-One*) s entitetom Mjesto, (*One-to-Many*) s entitetom Fotografija i (*Many-to-Many*) s entitetom Pokrovitelj.

Konferencija		
id	SERIAL	jedinstveni identifikator konferencije
urlVideo	VARCHAR	poveznica na direktno video praćenje trenutnih događanja u glavnoj konferencijskoj dvorani
pin	INT	jedinstveni pin konferencije
vrijemePocetak	TIMESTAMP	početak konferencije
vrijemeKraj	TIMESTAMP	kraj konferencije
adresa	VARCHAR	adresa konferencije
idAdmin	SERIAL	jedinstveni identifikator admina zaduženog za konferenciju
pbr	INT	poštanski broj mjesta u kojem se održava konferencija

Entitet **Prisutan na** sadrži informacije o prisutnosti pojedinog korisnika na određenoj

konferenciji te je li glasao na njoj ili ne. Sadrži attribute: ID konferencije, ID korisnika i glasao. Entitet **Prisutan_na** rezultat je binarne veze (*Many-to-Many*) entiteta Osoba s entitetom Konferencija.

Prisutan_na		
idKonf	SERIAL	jedinstveni identifikator konferencije
idKorisnik	SERIAL	jedinstveni identifikator korisnika
glasao	BOOLEAN	informacija je li korisnik već glasao na konferenciji

Entitet **Mjesto** sadrži informacije o pojedinom mjestu. Sadrži attribute: poštanski broj i naziv mjesta. Entitet Mjesto u binarnoj je vezi s entitetom Konferencija (*One-to-Many*).

Mjesto		
pbr	INT	poštanski broj mjesta
naziv	VARCHAR	naziv mjesta

Entitet **Fotografija** sadrži informacije o uslikanoj fotografiji te na kojoj konferenciji je uslikana. Sadrži attribute: ID fotografije, url fotografije i ID konferencije. Entitet Fotografija u binarnoj je vezi s entitetom Konferencija (*Many-to-One*).

Fotografija		
id	SERIAL	jedinstveni identifikator fotografije
urlSlike	VARCHAR	url fotografije
idKonf	SERIAL	jedinstveni identifikator konferencije

Entitet **Pokrovitelj** sadrži informacije o pokrovitelju. Sadrži attribute: ID pokrovitelja, url stranice pokrovitelja, naziv pokrovitelja i url slike. Entitet Pokrovitelj u binarnoj je vezi s entitetom Konferencija (*Many-to-Many*).

Pokrovitelj		
id	SERIAL	jedinstveni identifikator pokrovitelja
url	VARCHAR	poveznica na stranicu pokrovitelja
naziv	VARCHAR	naziv pokrovitelja
urlSlike	VARCHAR	logo pokrovitelja

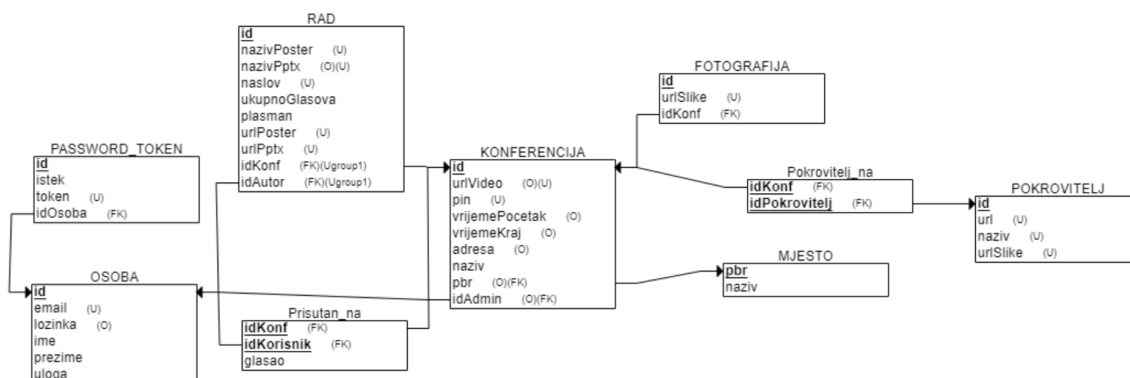
Entitet **Pokrovitelj_na** sadrži informacije o uključenosti pokrovitelja na pojedinoj konferenciji. Sadrži attribute: ID konferencije i ID pokrovitelja. Entitet Pokrovitelj_na rezultat je binarne veze (*Many-to-Many*) entiteta Pokrovitelj i Konferencija.

Pokrovitelj_na		
idKonf	SERIAL	jedinstveni identifikator konferencije
idPokrovitelj	SERIAL	jedinstveni identifikator pokrovitelja

Entitet **Password_token** sadrži informacije o tokenu koji je pojedina osoba dobila kada je zatražila promjenu lozinke. Sadrži attribute: ID tokena, istek tokena, token i ID osobe čiji je token. Entitet Password_token u binarnoj je vezi s entitetom Osoba (*One-to-One*). .

Pokrovitelj_na		
id	SERIAL	jedinstveni identifikator tokena
istek	DATE	datum isteka tokena
token	VARCHAR	token
idOsoba	SERIAL	jedinstveni identifikator osobe

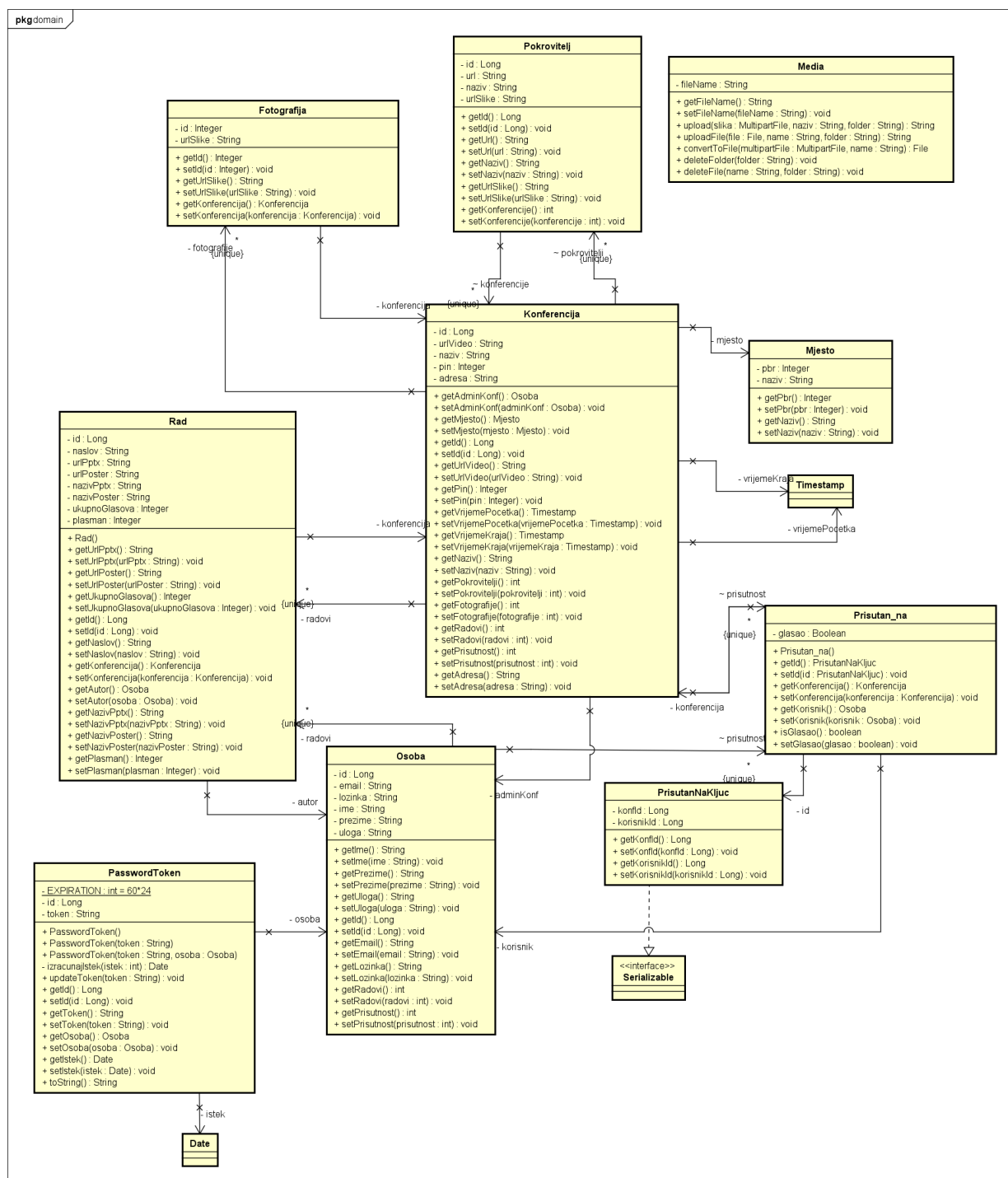
4.1.2 Dijagram baze podataka



Slika 4.3: Dijagram baze podataka

4.2 Dijagram razreda

Radi preglednosti, dijagram je razlomljen na nekoliko dijelova. Na njima su prikazani razredi koji pripadaju backend dijelu MVC arhitekture. Na slici 4.4 je prikazan Model dio. Model klase služe za prijenos podataka između baze podataka i serverske strane aplikacije. Ti su objekti zapravo preslika baze podataka, ali umjesto relacijske koristimo objektno-orijentiranu paradigmu. Razred Konferencija predstavlja konferenciju koja se prikazuje u aplikaciji. Razred Mjesto predstavlja lokaciju na kojoj se konferencija održava. Razred Osoba predstavlja čovjeka koji na neki način sudjeluje na konferenciji. Taj razred ima atribut uloga kojim se određuje je li ta osoba autor, administrator ili posjetitelj konferencije. Razred Rad predstavlja rad (poster i/ili pptx) kojim se autor predstavlja na konferenciji. Razred PrisutanNa omogućuje da pratimo tko je na konferenciji te da li je ta osoba glasala za neki rad. Razred Fotografija predstavlja fotografije konferencije koje administrator stavlja u aplikaciju tijekom ili nakon konferencije. Razred Pokrovitelj predstavlja sponzore konferencije. Razred PasswordToken omogućuje postavljanje i promjenu lozinku. Razred Media omogućuje lakši prikaz i pohranu medijskih sadržaja.



Slika 4.4: Dijagram razreda - dio Models

Na slikama 4.5, 4.6 i 4.7 je prikazan glavni dijagram u čijem središtu se nalazi JPARepository o kojem ovise ostala sučelja koja su specifična za svaki objekt. Ova sučelja nam omogućuju da izbjegnemo pisanje složenih SQL upita i umjesto toga koristimo generičke metode za izvođenje uobičajenih operacija s bazom podataka.

```

classDiagram
    class JpaPhotoRepository {
        <<interface>>
        findById(Long) List<Photo>
    }
    class PhotoRepository {
        <<interface>>
        findById(Long) List<Photo>
    }
    class JpaConferenceRepository {
        <<interface>>
        findById(Long) List<Conference>
    }
    class ConferenceRepository {
        <<interface>>
        findById(Long) List<Conference>
    }
    class PhotoService {
        <<interface>>
        findById(Long) List<Photo>
    }
    class PhotoController {
        <<interface>>
        findById(Long) List<Photo>
    }
    class JpaPhotoRepositoryImpl {
        <<implementation>>
        findById(Long) List<Photo>
    }
    class PhotoRepositoryImpl {
        <<implementation>>
        findById(Long) List<Photo>
    }
    class JpaConferenceRepositoryImpl {
        <<implementation>>
        findById(Long) List<Conference>
    }
    class ConferenceRepositoryImpl {
        <<implementation>>
        findById(Long) List<Conference>
    }
    class PhotoServiceImpl {
        <<implementation>>
        findById(Long) List<Photo>
    }
    class PhotoControllerImpl {
        <<implementation>>
        findById(Long) List<Photo>
    }
    JpaPhotoRepository <|-- PhotoRepository
    JpaConferenceRepository <|-- ConferenceRepository
    PhotoService <|-- PhotoServiceImpl
    PhotoController <|-- PhotoControllerImpl
    JpaPhotoRepositoryImpl <|-- PhotoRepositoryImpl
    JpaConferenceRepositoryImpl <|-- ConferenceRepositoryImpl
    PhotoServiceImpl <|-- PhotoServiceImpl
    PhotoControllerImpl <|-- PhotoControllerImpl
    JpaPhotoRepository <|-- JpaPhotoRepositoryImpl
    PhotoRepository <|-- PhotoRepositoryImpl
    JpaConferenceRepository <|-- JpaConferenceRepositoryImpl
    ConferenceRepository <|-- ConferenceRepositoryImpl
    PhotoService <|-- PhotoServiceImpl
    PhotoController <|-- PhotoControllerImpl
    JpaPhotoRepositoryImpl <|-- PhotoRepositoryImpl
    JpaConferenceRepositoryImpl <|-- ConferenceRepositoryImpl
    PhotoServiceImpl <|-- PhotoServiceImpl
    PhotoControllerImpl <|-- PhotoControllerImpl
    
```

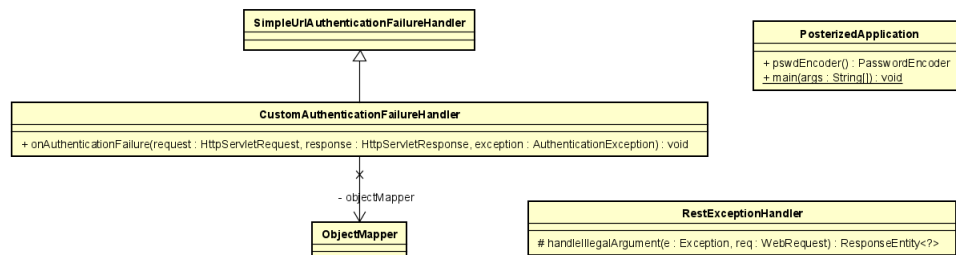
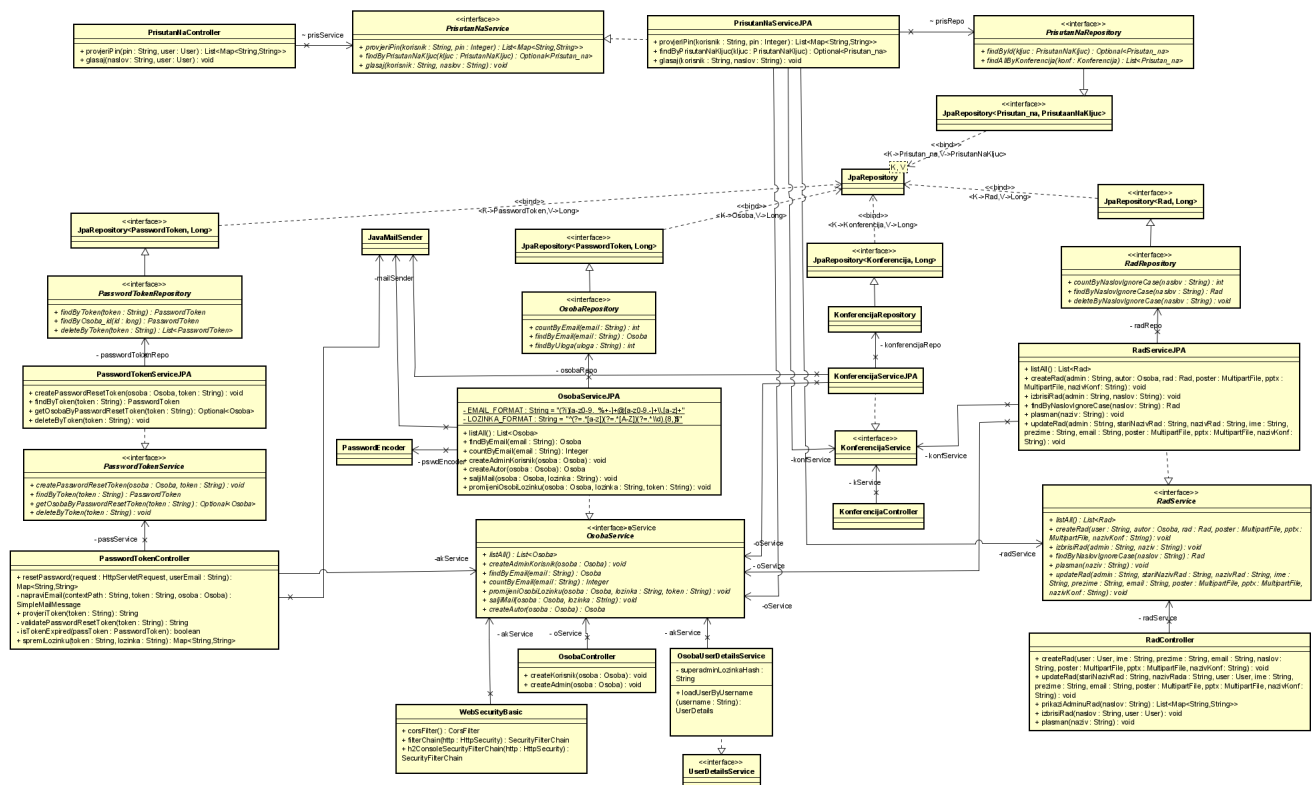
The diagram illustrates the architecture of a photo gallery application, organized into three main layers: Repository, Service, and Controller.

- Repository Layer:**
 - PhotoRepository:** An interface defining methods for photo management, including `findById(Long)` and `findAll()`.
 - ConferenceRepository:** An interface defining methods for conference management, including `findById(Long)` and `findAll()`.
 - JpaPhotoRepository:** An implementation of `PhotoRepository` using JPA.
 - JpaConferenceRepository:** An implementation of `ConferenceRepository` using JPA.
- Service Layer:**
 - PhotoService:** An interface defining business logic for photos, including `findById(Long)` and `findAll()`.
 - ConferenceService:** An interface defining business logic for conferences, including `findById(Long)` and `findAll()`.
 - PhotoServiceImpl:** An implementation of `PhotoService` that interacts with `PhotoRepository`.
 - ConferenceServiceImpl:** An implementation of `ConferenceService` that interacts with `ConferenceRepository`.
- Controller Layer:**
 - PhotoController:** An interface defining the API for photo management, including `findById(Long)` and `findAll()`.
 - ConferenceController:** An interface defining the API for conference management, including `findById(Long)` and `findAll()`.
 - PhotoControllerImpl:** An implementation of `PhotoController` that interacts with `PhotoService`.
 - ConferenceControllerImpl:** An implementation of `ConferenceController` that interacts with `ConferenceService`.

Key relationships and dependencies include:

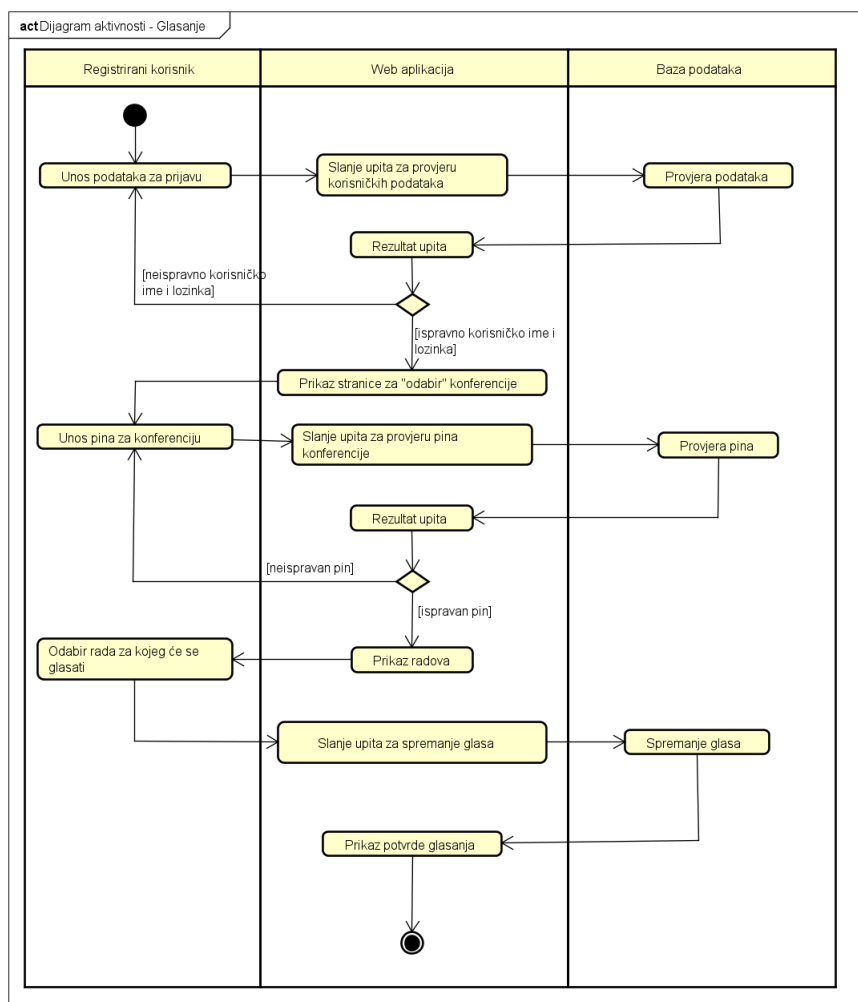
- `PhotoRepository` and `ConferenceRepository` are implemented by `JpaPhotoRepository` and `JpaConferenceRepository` respectively.
- `PhotoService` and `ConferenceService` are implemented by `PhotoServiceImpl` and `ConferenceServiceImpl` respectively.
- `PhotoController` and `ConferenceController` are implemented by `PhotoControllerImpl` and `ConferenceControllerImpl` respectively.
- The `PhotoServiceImpl` depends on `PhotoRepository` and `PhotoService`.
- The `ConferenceServiceImpl` depends on `ConferenceRepository` and `ConferenceService`.
- The `PhotoControllerImpl` depends on `PhotoService` and `PhotoController`.
- The `ConferenceControllerImpl` depends on `ConferenceService` and `ConferenceController`.

Slika 4.5: Dijagram razreda - glavni dijagram 1.dio



4.4 Dijagram aktivnosti

Dijagram aktivnosti primjenjuje se za opis modela toka upravljanja ili toka podataka. Ne upotrebljava se za modeliranje događajima poticanog ponašanja. U modeliranju toka upravljanja svaki novi korak poduzima se nakon završenog prethodnog, a naglasak je na jednostavnosti. Na dijagramu 4.8 prikazan je proces glasanja za najbolji rad. Registrirani korisnik se prijavljuje u sustav, a nakon uspješne prijave ima mogućnost prijaviti se na konferenciju. Korisnik se prijavljuje na konferenciju unosom pina te na web aplikaciji vidi mogućnosti povezane s tom konferencijom, uključujući i glasanje za najbolji rad. Odabire koji rad mu je najbolji te glasa za njega.



Slika 4.9: Dijagram aktivnosti

4.5 Dijagram komponenti

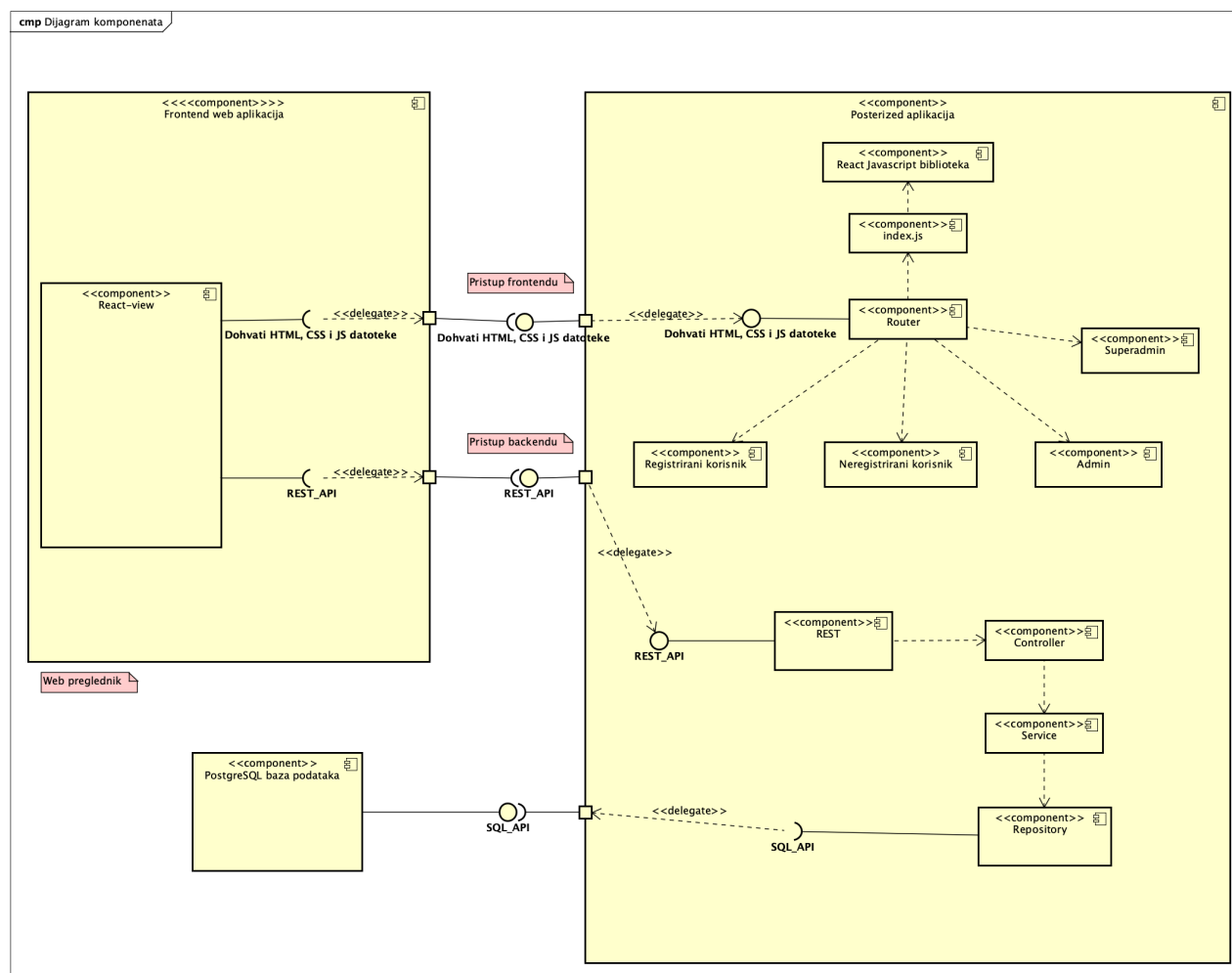
Dijagram komponenti prikazan na slici opisuje strukturu web aplikacije razdijeljenu na višestruke komponente koje omogućavaju njen rad i interakciju s korisnicima. Pristup sustavu ostvaruje se putem dva osnovna sučelja.

Prvo sučelje omogućava dohvat HTML, CSS i JS datoteka koje čine frontend dio aplikacije. Komponente frontenda koriste React biblioteku za izgradnju korisničkog sučelja i organizirane su u logičke cjeline prema tipovima korisnika koji im pristupaju, kao što su superadministratori, administratori, registrirani i neregistrirani korisnici.

Router je ključna komponenta koja određuje koja će se datoteka poslužiti temeljem URL-a na koji korisnik dođe. Kroz ovu komponentu, zahtjevi korisnika se usmjeravaju prema odgovarajućim dijelovima aplikacije.

Backend dio aplikacije pristupa se preko sučelja koje poslužuje JSON podatke. Ova komponenta se bavi obradom zahtjeva i komunikacijom s bazom podataka. Komunikacija s bazom podataka se odvija preko posebnog SQL API sučelja, a podaci se poslužuju kroz REST API.

Svi dijelovi sustava su međusobno povezani i ovise o funkcionalnostima koje pružaju jedni drugima. Na primjer, REST API poslužuje kao most između frontenda i baze podataka, omogućavajući dinamičan i responzivan korisnički doživljaj. Dijagram detaljno prikazuje kako korisnički zahtjevi prolaze kroz različite slojeve aplikacije kako bi se dobili potrebni podaci ili izvršile akcije.



Slika 4.10: Dijagram komponenta

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

Tim uspješno komunicira putem aplikacija Discord i WhatsApp, omogućavajući efikasnu i brzu razmjenu informacija.

UML dijagrami napisani su u okruženju AstahUML. Koristimo ih za vizualizaciju i analizu softverskog rješenja, što pomaže boljem razumijevanju arhitekture i funkcionalnosti sustava.

U procesu razvoja koristimo Git za upravljanje verzijama koda, s udaljenim repozitorijem na GitHubu. Ovo omogućava timsku suradnju, praćenje promjena i povratak na prethodne verzije koda po potrebi. Korištena razvojna okruženja su IntelliJ za programski jezik Java i WebStorm za p. jezik JavaScript, čime smo osigurali snažne alate za razvoj i debugiranje aplikacije. Što se tiče samog softverskog rješenja, frontend aplikacija je napisana u JavaScriptu, koristeći biblioteku ReactJS i Node.js za poslužitelja web aplikacije. Backend je napisan u Java Spring framework-u, time je pružena stabilnost serverskom dijelu.

Za registraciju koristimo Google reCAPTCHA API kako bi povećavali sigurnost aplikacije. Sve slike koje koristimo skladištimo na Firebase Cloud platformi, pružajući pouzdanu infrastrukturu za upravljanje multimedijским sadržajem. U fazi razvoja koristimo H2 bazu podataka, dok aplikacija u pogonu koristi udaljenu PostgreSQL instancu pruženu od strane Render platforme. Pogon aplikacije vršimo putem Render platforme, što omogućava jednostavan i efikasan proces.

Za pisanje dokumentacije koristimo TexStudio okruženje, koje podržava LaTeX jezik. Ova kombinacija omogućava strukturirano i profesionalno dokumentiranje rješenja.

Sve ove tehnologije i alati zajedno čine tim sposobnim za efikasan razvoj, održavanje i dokumentiranje softverskog rješenja.

5.2 Ispitivanje programskog rješenja

dio 2. revizije

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

5.2.1 Ispitivanje komponenti

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

5.2.2 Ispitivanje sustava

*Potrebno je provesti i opisati ispitivanje sustava koristeći radni okvir Selenium¹. Razraditi **minimalno 4 ispitna slučaja** u kojima će se ispitati redovni slučajevi, rubni uvjeti te poziv funkcionalnosti koja nije implementirana/izaziva pogrešku kako bi se vidjelo na koji način sustav reagira kada nešto nije u potpunosti ostvareno. Ispitni slučaj se treba sastojati od ulaza (npr. korisničko ime i lozinka), očekivanog izlaza ili rezultata, koraka ispitivanja i dobivenog izlaza ili rezultata.*

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

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

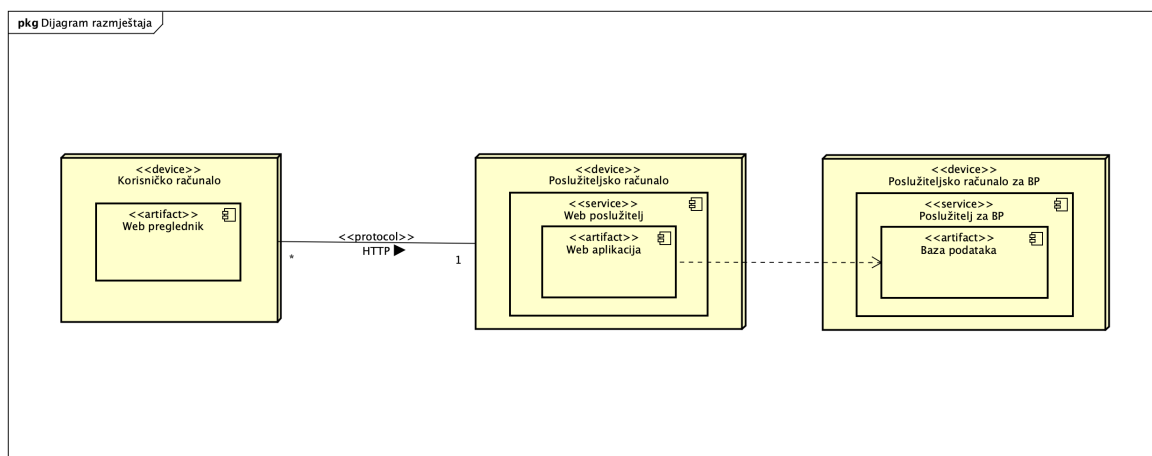
Detalji o korištenju alata Selenium bit će prikazani na posebnom predavanju tijekom semestra.

¹<https://www.seleniumhq.org/>

5.3 Dijagram razmještaja

Dijagrami razmještaja pružaju uvid u organizaciju hardverskih komponenti i programske podrške koja se koristi u implementaciji sustava unutar njegovog radnog okruženja. Na jednom od poslužiteljskih računala smješten je web poslužitelj, dok se na drugom nalazi poslužitelj baze podataka. Klijenti koriste web preglednik kako bi pristupili web aplikaciji. Sustav je temeljen na klijent-poslužitelj arhitekturi, pri čemu se komunikacija između računala korisnika i poslužitelja odvija putem HTTP veze.

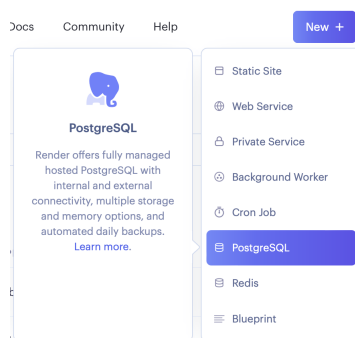
Web poslužitelj odgovoran je za pružanje web stranica i servisa korisnicima, dok poslužitelj baze podataka čuva, upravlja i omogućava pristup podacima. Korištenjem web preglednika, korisnici ostvaruju pristup aplikaciji koja se nalazi na sustavu, a komunikacija se odvija preko standardne HTTP veze, osiguravajući efikasnu interakciju između klijenta i poslužitelja.



Slika 5.1: Dijagram razmještaja

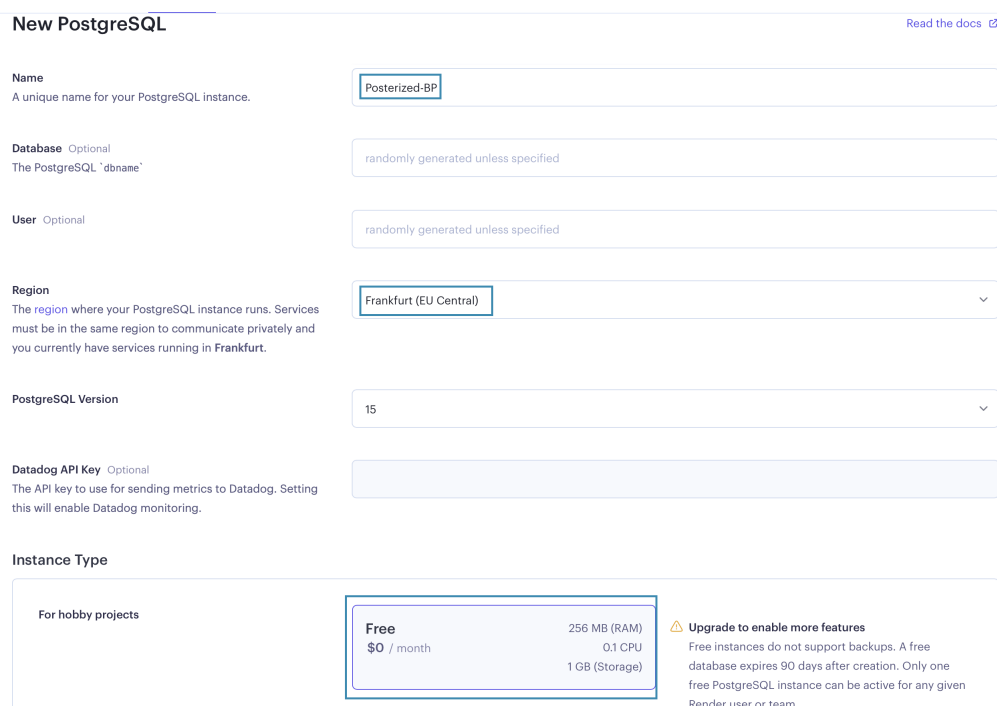
5.4 Upute za puštanje u pogon

Aplikacija je puštena u pogon koristeći alat Render. Koristeći korisničko sučelje koje Render pruža treba kreirati instancu baze podataka, instancu za web servis backenda te instancu za web servis frontenda aplikacije. Prvo treba kreirati bazu podataka PostgreSQL.



Slika 5.2: Izbornik za stvaranje nove baze podataka

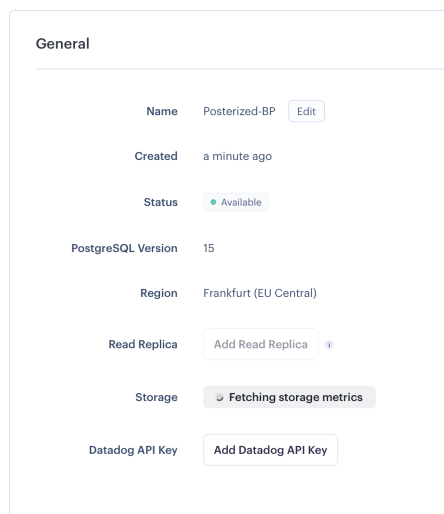
Potrebno je upisati ime baze, odabrati regiju poslužitelja instance i označiti tip instance koji će se koristiti.

The image shows the 'New PostgreSQL' form in the Render dashboard. The form has several fields: 'Name' (with a placeholder 'Posterized-BP'), 'Database' (optional, with a placeholder 'randomly generated unless specified'), 'User' (optional, with a placeholder 'randomly generated unless specified'), 'Region' (dropdown menu with 'Frankfurt (EU Central)' selected), 'PostgreSQL Version' (dropdown menu with '15' selected), 'Datadog API Key' (optional, with a placeholder), and 'Instance Type' (a section with a 'Free' plan highlighted, showing '\$0 / month', '256 MB (RAM)', '0.1 CPU', and '1 GB (Storage)'. There is also an 'Upgrade to enable more features' warning.

Slika 5.3: Stvaranje nove baze podataka

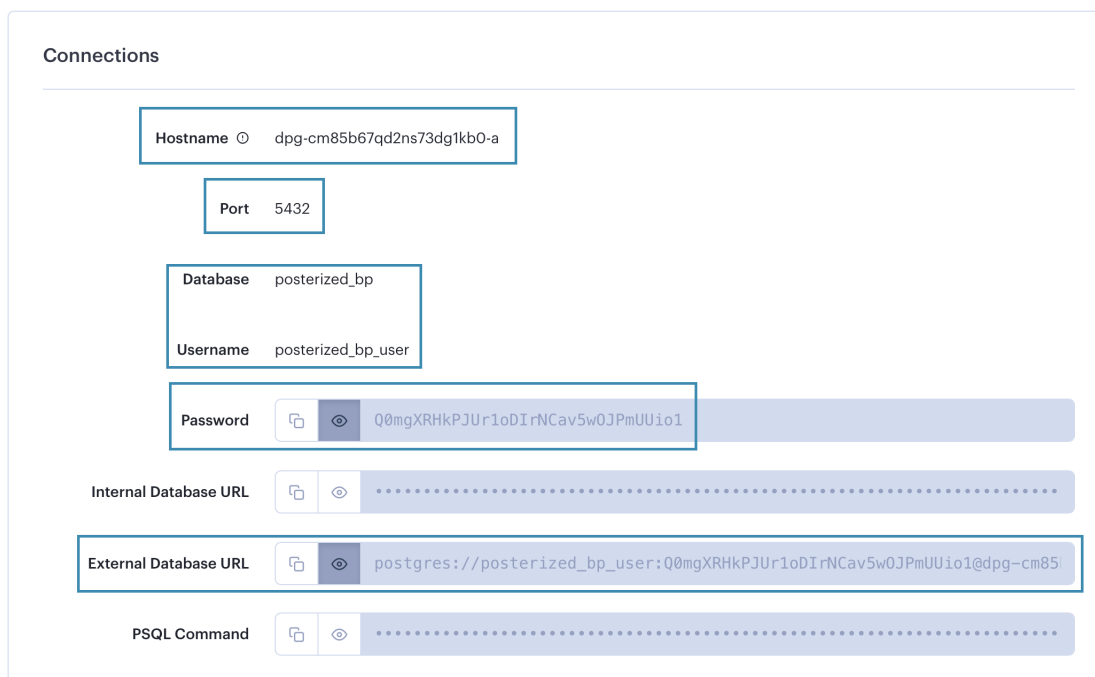
Nakon ovih koraka treba pokrenuti stvaranje pritiskom na tipku "Create".

Pokazuje se prozor u kojemu su navedeni osnovni podatci o bazi kao ime, verzija, regija, prostor za pohranu, API ključ, itd.



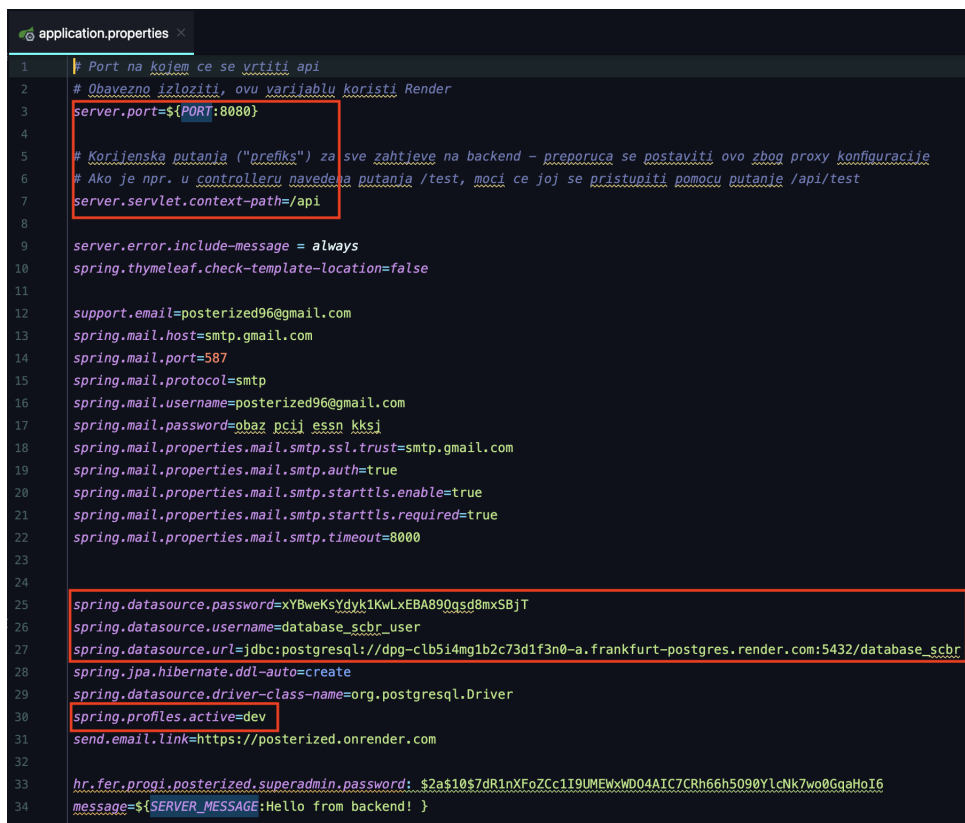
Slika 5.4: Osnovni podatci o statusu baze podataka

Kad je baza uspješno kreirana, potrebno je uzeti podatke koje treba dati backendu aplikacije. Ti podatci se nalaze u poljima *Hostname*, *Port*, *Database*, *Username*, *Password* i *External Database URL*.



Slika 5.5: Podatci za pristupanje bazi podataka

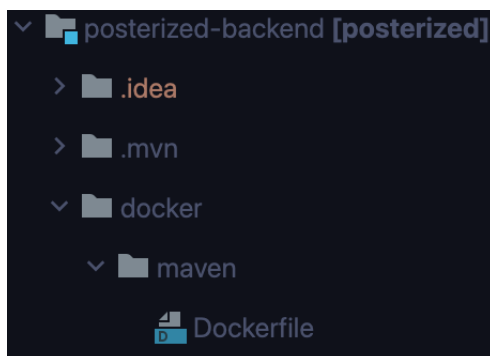
Za kreiranje backend web servisa potrebno je prvo napraviti pripremu pa automatsko kreiranje web servisa iz GitHub repozitorija. U projektu razvojnog okruženja treba dodati port servera, kontekstnu putanju, podatke za pristup bazi upisati u označeni prozor te izbrisati aktivni dev profil.



```
1 # Port na kojem ce se vrtiti api
2 # Obavezno izloziti, ovu varijablu koristi Render
3 server.port=${PORT:8080}
4
5 # Korijenska putanja ("prefix") za sve zahtjeve na backend - preporuca se postaviti ovo zbog proxy konfiguracije
6 # Ako je npr. u controlleru navedena putanja /test, moci ce joj se pristupiti pomocu putanje /api/test
7 server.servlet.context-path=/api
8
9 server.error.include-message = always
10 spring.thymeleaf.check-template-location=false
11
12 support.email=posterized96@gmail.com
13 spring.mail.host=smtp.gmail.com
14 spring.mail.port=587
15 spring.mail.protocol=smtp
16 spring.mail.username=posterized96@gmail.com
17 spring.mail.password=obaz pcij essn kksj
18 spring.mail.properties.mail.smtp.ssl.trust=smtp.gmail.com
19 spring.mail.properties.mail.smtp.auth=true
20 spring.mail.properties.mail.smtp.starttls.enable=true
21 spring.mail.properties.mail.smtp.starttls.required=true
22 spring.mail.properties.mail.smtp.timeout=8000
23
24
25 spring.datasource.password=xYBweKsYdyk1KwLxEBA890qsd8mxSBjT
26 spring.datasource.username=database_schr_user
27 spring.datasource.url=jdbc:postgresql://dpg-clb5i4mg1b2c73d1f3n0-a-frankfurt-postgres.render.com:5432/database_schr
28 spring.jpa.hibernate.ddl-auto=create
29 spring.datasource.driver-class-name=org.postgresql.Driver
30 spring.profiles.active=dev
31 send.email.link=https://posterized.onrender.com
32
33 hr.fer.progi.posterized.superadmin.password: $2a$10$dR1nXf0zCc1I9UMEWxwD04AIC7CRh66h5090Y1cNk7wo0GqaHoI6
34 message=${SERVER_MESSAGE:Hello from backend! }
```

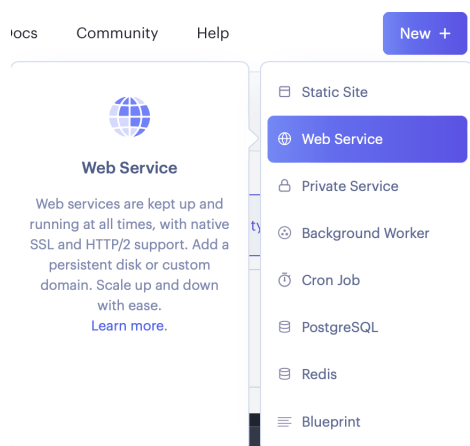
Slika 5.6: Datoteka *application.properties*

U projekt je potrebno ubaciti odgovarajući *Dockerfile* na putanju *./posterized-backend/docker/maven* koja je prikazana stablom.



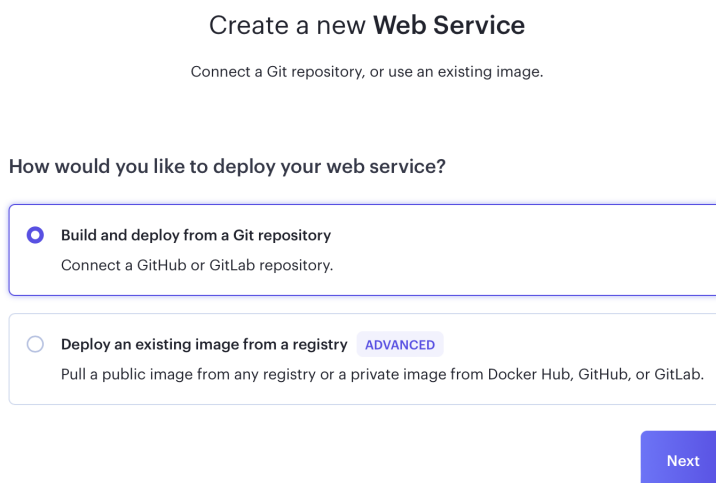
Slika 5.7: Stablo u kojem se nalazi *Dockerfile*

Kreiranje web servisa se pokreće iz Renderovog izbornika.



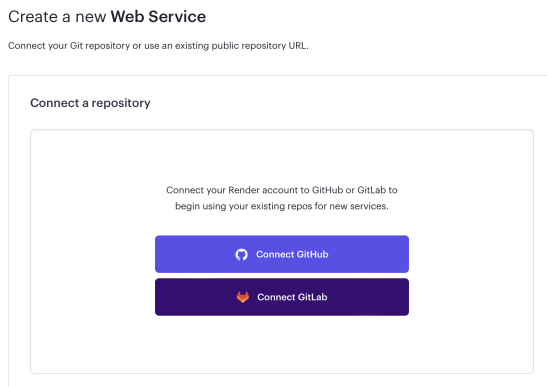
Slika 5.8: Izbornik za kreiranje web servisa

Potrebno je povezati GitHub repozitorij s Renderom. Označiti "Build and deploy from a Git repository" i kliknuti "Next".

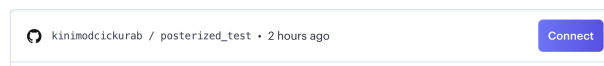


Slika 5.9: Kreiranje web servisa - povezivanje s udaljenim repozitorijem

Kliknuti "Connect GitHub", prijaviti se u račun, odobriti autorizaciju i kliknuti "Connect".



Slika 5.10: Povezivanje s repozitorijem



Slika 5.11: Pronađeni repozitorij

Upisati ime servisa i odabrati regiju koja je najbliža našoj lokaciji. Pod *Branch* odabrati master i upisati ime mape projekta. Odabrati tip instance koji će se koristiti.

You are deploying a web service for [kinimodcickurab/posterized_test](#).

Name
A unique name for your web service.

Region
The region where your web service runs. Services must be in the same region to communicate privately and you currently have services running in Frankfurt.

Branch
The repository branch used for your web service.

Root Directory Optional
Defaults to repository root. When you specify a root directory that is different from your repository root, Render runs all your commands in the specified directory and ignores changes outside the directory.

Runtime
The runtime for your web service.

Instance Type

For hobby projects

Free \$0 / month	512 MB (RAM) 0.1 CPU	Upgrade to enable more features Free instances spin down after periods of inactivity. They do not support SSH access, scaling, one-off jobs, or persistent disks. Select any paid instance type to enable these features.
----------------------------	-------------------------	---

Slika 5.12: Kreiranje web servisa - upisivanje osnovnih podataka

Otvoriti padajući izbornik "Advanced" i upisati putanju gdje se nalazi *Dockerfile*, zatim kliknuti "Create Web Service".

Advanced ^

Store plaintext files containing secret data (such as a `.env` file or a private key).

+ Add Secret File

Read these files during builds and at runtime from your app's specified root directory, or from the absolute path `/etc/secret<filename>`.

Health Check Path
If you're running a server, enter the path where your server will always return a `200 OK` response. We use it to monitor your app and for [zero downtime deploys](#).

/healthz

Docker Build Context Directory
[Docker build context directory](#). This is relative to your repository root. Defaults to the root.

posterized-backend/ .

Dockerfile Path
Path to your Dockerfile relative to the repository root. This is *not* relative to your Docker build context. For example, `./subdir/Dockerfile`.

posterized-backend/ ./docker/maven/Dockerfile

Slika 5.13: Navođenje *Dockerfile* putanje

Web servis backenda je uspješno kreiran i pušten u pogon.

December 30, 2023 at 6:57 PM Live

17c31b1 odkomentirana captcha

```
All logs Search Live tail GMT+1 ↑
```

```
tingFilter@324b6a56, org.springframework.security.web.authentication.www.BasicAuthenticationFilter@70997a94, org.springframework.se
curity.web.savedrequest.RequestCacheAwareFilter@52b959df, org.springframework.security.web.servletapi.SecurityContextHolderAwareReq
uestFilter@553d2579, org.springframework.security.web.authentication.AnonymousAuthenticationFilter@421d54b3, org.springframework.se
curity.web.access.ExceptionTranslationFilter@35eee641, org.springframework.security.web.access.intercept.AuthorizationFilter@59096b
66]
```

```
Dec 30 07:02:26 PM ⓘ Your service is live 🎉
```

```
Dec 30 07:02:19 PM ⓘ 2023-12-30T18:02:19.987Z INFO 1 --- [main] o.s.b.a.e.web.EndpointLinksResolver : Exposing
1 endpoint(s) beneath base path '/actuator'
```

```
Dec 30 07:02:21 PM ⓘ 2023-12-30T18:02:21.585Z INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat st
arted on port(s): 10000 (http) with context path '/api'
```

```
Dec 30 07:02:21 PM ⓘ 2023-12-30T18:02:21.786Z INFO 1 --- [main] h.f.p.posterized.PosterizedApplication : Started P
osterizedApplication in 129.09 seconds (process running for 140.28)
```

```
Dec 30 07:02:22 PM ⓘ 2023-12-30T18:02:22.889Z INFO 1 --- [main] o.s.b.a.b.JobLauncherApplicationRunner : Running d
efault command line with: []
```

Slika 5.14: Logovi stvaranja i status web servisa

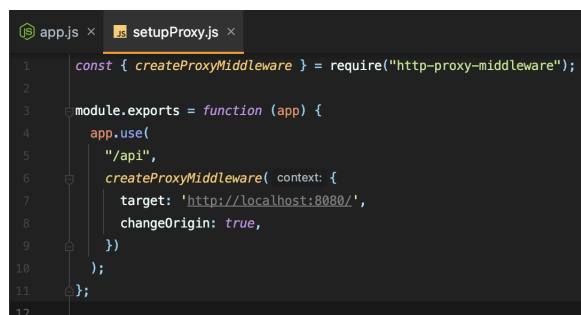
Prije kreiranja web servisa za frontend, potrebno je prvo pripremiti projekt frontenda za puštanje u pogon. Treba napisati *app.js* datoteku kao na slici. Ova datoteka je dio Node.js-a, odnosno Express frameworka koji je odgovoran u izgradnji RESTful API-ja za jednostrane, višestranne ili hibridne aplikacije.

The image shows a code editor window with a file named 'app.js'. The code is written in JavaScript and uses the Express.js framework. It includes imports for 'express', 'http-proxy-middleware', 'dotenv', and 'path'. The code sets up an Express application, configures environment variables, and uses 'http-proxy-middleware' to proxy requests to the API base URL. It also serves static files from the 'build' directory and sets up a default route for the index.html file.

```
1 const express = require("express");
2 const { createProxyMiddleware } = require("http-proxy-middleware");
3 require("dotenv").config();
4 const path = require("path")
5
6 const app = express();
7
8 var cors = require('cors')
9 app.use(cors())
10
11 // Configuration
12 const { PORT } = process.env;
13 const { HOST } = process.env;
14 const { API_BASE_URL } = process.env;
15
16 // Proxy
17 app.use(
18   "/api",
19   createProxyMiddleware( context: {
20     target: API_BASE_URL,
21     changeOrigin: true,
22   })
23 );
24 app.use(express.static(path.join(__dirname, 'build')))
25
26 dominik
27 app.listen(PORT, HOST, backlog: () => {
28   console.log(`Starting Proxy at ${HOST}:${PORT}`);
29 });
30
31 dominik
32 app.get("/*", async (req :... , res :Response<ResBody, LocalsObj> ) => {
33   res.sendFile(path.join(__dirname, 'build', 'index.html'))
34 });
```

Slika 5.15: Datoteka *app.js*

Napisati *setupProxy.js* za uspostavljanje proxyja između razvojnog okružja i API-ja koji služi izbjegavanju CORS grešaka što pridonosi lakšem razvoju aplikacije.

The image shows a code editor window with a file named 'setupProxy.js'. The code is written in JavaScript and uses the 'http-proxy-middleware' package. It defines a function 'createProxyMiddleware' that takes an 'app' object as an argument and sets up a proxy for the '/api' path, targeting 'http://localhost:8080/' and allowing CORS.

```
1 const { createProxyMiddleware } = require("http-proxy-middleware");
2
3 module.exports = function (app) {
4   app.use(
5     "/api",
6     createProxyMiddleware( context: {
7       target: 'http://localhost:8080/',
8       changeOrigin: true,
9     })
10   );
11 };
12
```

Slika 5.16: Datoteka *setupProxy.js*

Potrebno je izmijeniti datoteku *package.lock* kao na slici u označenom prozoru.

```
"scripts": {  
  "start": "react-scripts start",  
  "build": "yarn install && react-scripts build",  
  "start-prod": "node app.js",  
  "test": "react-scripts test",  
  "eject": "react-scripts eject"  
},
```

Slika 5.17: Datoteka *package.lock*

Kreiranje web servisa za frontend slično je kao za backend uz manje razlike. Započinje se u izborniku kao što je navedeno za backend.

U postupku kreiranja web servisa u označene prozore treba upisati ime, kao *Branch* odabrati master te ispod upisati ime mape projekta. Za *Runtime* odabrati Node i označiti tip instance koji će se koristiti.

You are deploying a web service for [kinimodcickurab/posterized_test](#).

Name
A unique name for your web service.

Posterized-FE

Region
The region where your web service runs. Services must be in the same region to communicate privately and you currently have services running in Frankfurt.

Frankfurt (EU Central)

Branch
The repository branch used for your web service.

master

posterized-frontend

Root Directory Optional
Defaults to repository root. When you specify a root directory that is different from your repository root, Render runs all your commands in the specified directory and ignores changes outside the directory.

Runtime
The runtime for your web service.

Docker

Instance Type

For hobby projects

Free \$0 / month	512 MB (RAM) 0.1 CPU
----------------------------	-------------------------

Upgrade to enable more features
Free instances spin down after periods of inactivity. They do not support SSH access, scaling, one-off jobs, or persistent disks. Select any paid instance type to enable these features.

Slika 5.18: Kreiranje web servisa za frontend

Ispod *Runtime* pojavili su se linije za naredbe build i start te ih je potrebno izmijeniti kao na slici.

Runtime
The runtime for your web service.

Build Command
This command runs in the root directory of your repository when a new version of your code is pushed, or when you deploy manually. It is typically a script that installs libraries, runs migrations, or compiles resources needed by your app.

Start Command
This command runs in the root directory of your app and is responsible for starting its processes. It is typically used to start a webserver for your app. It can access environment variables defined by you in Render.

Slika 5.19: Naredbe za build i start

Odabrati "Environment Variables" i kreirati novu varijablu okruženja za adresu backenda koja je dostupna u "Dashboard" na Renderu. Jedinstvena adresa se dodjeljuje svakom kreiranom web servisu. Zatim kliknuti "Create Web Service" i pričekati "build" i "deploy" procese.

Environment Variables Optional
Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more.](#)

Variable Name	Value
API_BASE_URL	https://posterized-be-rf1z.onrender.com/

+ Add Environment Variable

Slika 5.20: Varijable okruženja

Kada status servisa postane "Live", kliknuti na "Dashboard". Ako je sve uspješno postavljeno, instance će biti prikazane kao na slici.

SERVICE NAME	STATUS	TYPE	RUNTIME	REGION	LAST DEPLOYED ↓	
Posterized-FE	Deployed	Web Service	Node	Frankfurt	8 minutes ago	...
Posterized-BE	Deployed	Web Service	Docker	Frankfurt	21 minutes ago	...
Posterized-BP	Available	PostgreSQL	PostgreSQL 15	Frankfurt	an hour ago	...

Slika 5.21: Uspostavljene instance

Aplikacija je puštena u pogon i spremna za uporabu. Pristupa se preko adrese koja je navedena za instancu frontenda.

<https://posterized.onrender.com>

6. Zaključak i budući rad

Zadatak našeg tima bio je razviti web aplikaciju za digitalne postere namijenjene prezentacijama na konferencijama. Cilj projekta uspješno je postignut kroz dvije faze.

U prvoj fazi, formirali smo tim i međusobno se upoznali, raspravljajući o zadatku, mogućim idejama i planu razvoja projekta. Tim je bio podijeljen na backend i frontend podtimove, dok je voditelj projekta bio zadužen za dokumentaciju i organizaciju. Postignute su osnovne funkcionalnosti, uključujući početnu stranicu, registraciju i prijavu korisnika. Također, izrađen je značajan dio dokumentacije koji uključuje uvod, obrasce uporabe, prateće UML dijagrame, funkcionalne i nefunkcionalne zahtjeve, arhitekturu sustava te ostale UML dijagrame. Članovi tima su znatno pridonijeli pisanju dokumentacije i organizaciji projekta.

U drugoj fazi, dovršeni su preostali dijelovi dokumentacije i funkcionalnosti aplikacije. Backend i frontend timovi su usko surađivali i uspješno dovršili razvoj aplikacije. Dokumentacija je proširena uputama za puštanje aplikacije u pogon, dijagramom razmještaja, dijagramom aktivnosti, dijagramom stanja, dijagramom komponentata, revizijom dijagrama razreda i zaključkom.

Komunikacija unutar tima odvijala se putem serverskih kanala na platformi Discord, osiguravajući da su svi članovi jednako informirani i uključeni u projekt. Zadaci su bili individualno raspoređeni među članovima tima, s mogućnošću da svaki član preuzme inicijativu i uključi se u bilo koji segment projekta. Korištene tehnologije uključuju Java Spring, JavaScript, React biblioteku i platformu Render za puštanje aplikacije u pogon.

Kao moguća proširenja aplikacije, razmatramo dodavanje UX/UI dizajnera za unaprjeđenje korisničkog sučelja, razvoj mobilne aplikacije te redizajn aplikacije kako bi se omogućilo svakoj dvorani, hotelu ili bilo kojoj nekretnini s društvenom namjenom da ima vlastitu verziju aplikacije, slično platformi Moodle koja se koristi u akademskim ustanovama.

Popis literature

1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
2. I. Sommerville, "Software engineering", 8th ed, Addison Wesley, 2007.
3. T.C.Lethbridge, R.Langaniere, "Object-Oriented Software Engineering", 2nd ed. McGraw-Hill, 2005.
4. I. Marsic, Software engineering book", Department of Electrical and Computer Engineering, Rutgers University, <http://www.ece.rutgers.edu/~marsic/books/SE>
5. The Unified Modeling Language, <https://www.uml-diagrams.org/>
6. Astah Community, <http://astah.net/editions/uml-new>
7. Spring Boot Reference Documentation, <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>
8. Vite, Vite Getting Started Guide, <https://vitejs.dev/guide/>
9. Render, Render Docs, <https://render.com/docs>
10. React, React Reference, <https://react.dev/reference/react>

Indeks slika i dijagrama

3.1	Dijagram obrazaca uporabe - administrator	15
3.2	Dijagram obrazaca uporabe - korisnici	16
3.3	Sekvencijski dijagram korištenja aplikacije	18
3.4	Sekvencijski dijagram korištenja aplikacije od strane administratora	19
4.1	Organizacija sustava s najviše razine apstrakcije	21
4.2	Organizacija aplikacije	22
4.3	Dijagram baze podataka	27
4.4	Dijagram razreda - dio Models	29
4.5	Dijagram razreda - glavni dijagram 1.dio	30
4.6	Dijagram razreda - glavni dijagram 2.dio	31
4.7	Dijagram razreda - glavni dijagram 3.dio	31
4.8	Dijagram stanja	32
4.9	Dijagram aktivnosti	33
4.10	Dijagram komponenata	35
5.1	Dijagram razmještaja	38
5.2	Izbornik za stvaranje nove baze podataka	39
5.3	Stvaranje nove baze podataka	39
5.4	Osnovni podatci o statusu baze podataka	40
5.5	Podatci za pristupanje bazi podataka	40
5.6	Datoteka <i>application.properties</i>	41
5.7	Stablo u kojem se nalazi <i>Dockerfile</i>	41
5.8	Izbornik za kreiranje web servisa	42
5.9	Kreiranje web servisa - povezivanje s udaljenim repozitorijem	42
5.10	Povezivanje s repozitorijem	43
5.11	Pronađeni repozitorij	43
5.12	Kreiranje web servisa - upisivanje osnovnih podataka	43
5.13	Navođenje <i>Dockerfile</i> putanje	44
5.14	Logovi stvaranja i status web servisa	44
5.15	Datoteka <i>app.js</i>	45

5.16	Datoteka <i>setupProxy.js</i>	45
5.17	Datoteka <i>package.lock</i>	46
5.18	Kreiranje web servisa za frontend	46
5.19	Naredbe za build i start	47
5.20	Varijable okruženja	47
5.21	Uspostavljene instance	47

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak

- Datum: 19. listopada 2023.
- Prisustvovali: D.Barukčić, L.Barić, N.Božić, K.Đunđek, L.Jukić, E.Samaržija, T.Topolovec
- Teme sastanka:
 - upoznavanje
 - diskusija na temu projekta

2. sastanak

- Datum: 20. listopada 2023.
- Prisustvovali: D.Barukčić, L.Barić, N.Božić, K.Đunđek, L.Jukić, E.Samaržija, T.Topolovec
- Teme sastanka:
 - podjela poslova i grupiranje unutarnjih timova
 - dogovaranje oko tehnologija koje će se koristiti u izradi projekta

3. sastanak

- Datum: 26. listopada 2023.
- Prisustvovali: D.Barukčić, L.Barić, N.Božić, K.Đunđek, L.Jukić, E.Samaržija, T.Topolovec
- Teme sastanka:
 - razrada backenda
 - ideje o izgledu stranice

4. sastanak

- Datum: 3. studenoga 2023.
- Prisustvovali: D.Barukčić, L.Barić, N.Božić, K.Đunđek, L.Jukić, E.Samaržija, T.Topolovec
- Teme sastanka:
 - izrada stranice, homepagea, logina i registracije

- testiranje backenda

5. sastanak

- Datum: 15. studenoga 2023.
- Prisustvovali: D.Barukčić, L.Barić, N.Božić, K.Đunđek, L.Jukić, E.Samaržija, T.Topolovec
- Teme sastanka:
 - rasprava o deployu
 - završavanje dokumentacije
 - problemi u backendu

Tablica aktivnosti

	Dominik Barukčić	Lana Barić	Nika Božić	Kristina Đunđek	Lovro Jukić	Ena Samaržija	Tea Topolovec
Upravljanje projektom	15	0	0	0	0	0	0
Opis projektnog zadatka	1	0	2	0	0	0	0
Funkcionalni zahtjevi	1	0	4	0	5	5	0
Opis pojedinih obrazaca	1	0	3	0	3	1	0
Dijagram obrazaca	0	0	0	2	0	3	0
Sekvencijski dijagrami	0	2	0	3	0	0	0
Opis ostalih zahtjeva	0	3	0	0	0	0	0
Arhitektura i dizajn sustava	0	0	0	0	0	0	3
Baza podataka	0	0	0	0	0	0	4
Dijagram razreda	0	2	0	2	0	0	0
Dijagram stanja	0	0	0	0	0	0	0
Dijagram aktivnosti	0	0	0	0	0	0	0
Dijagram komponenti	2	0	0	0	0	0	0
Korištene tehnologije i alati	1	0	0	0	0	0	0
Ispitivanje programskog rješenja	0	0	0	0	0	0	0
Dijagram razmještaja	1	0	0	0	0	0	0
Upute za puštanje u pogon	2	0	0	0	0	0	0
Dnevnik sastajanja	1	0	0	0	0	0	0
Zaključak i budući rad	2	0	0	0	0	0	0

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

	Dominik Barukčić	Lana Barić	Nika Božić	Kristina Đundek	Lovro Jukić	Ena Samaržija	Tea Topolovec
Popis literature	1	0	0	0	0	0	0
<i>izrada baze podataka</i>	0	0	0	0	0	0	4
<i>back end</i>	0	13	0	14	0	0	11
<i>front end</i>	0	13	0	14	0	0	11
<i>deploy</i>	20	0	0	0	0	0	0

Dijagrami pregleda promjena

dio 2. revizije

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