

Programsko inženjerstvo

Ak. god. 2023./2024.

ConnectiNET

Dokumentacija, Rev. 2

Grupa: *Kraljevi*

Voditelj: *Luka Miličević*

Datum predaje: 19. 1. 2023.

Nastavnik: *Doc. Dr. Sc. Nikolina Frid*

Sadržaj

1 Dnevnik promjena dokumentacije	2
2 Opis projektnog zadatka	5
3 Specifikacija programske potpore	9
3.1 Funkcionalni zahtjevi	9
3.1.1 Obrasci uporabe	12
3.1.2 Sekvencijski dijagrami	26
3.2 Ostali zahtjevi	30
4 Arhitektura i dizajn sustava	31
4.1 Baza podataka	33
4.1.1 Opis tablica	33
4.1.2 Dijagram baze podataka	42
4.2 Dijagram razreda	43
4.3 Dijagram stanja	46
4.4 Dijagram aktivnosti	47
4.5 Dijagram komponenti	49
5 Implementacija i korisničko sučelje	51
5.1 Korištene tehnologije i alati	51
5.2 Ispitivanje programskog rješenja	54
5.2.1 Ispitivanje komponenti	54
5.2.2 Ispitivanje sustava	58
5.3 Dijagram razmještaja	62
5.4 Upute za puštanje u pogon	63
5.4.1 Priprema Lokalnog Razvojnog Okruženja	63
5.4.2 Postavljanje Backend-a s Docker Compose	63
5.4.3 Konfiguracija PostgreSQL Baze Podataka	63
5.4.4 Izgradnja i Pokretanje Frontend-a	64
5.4.5 Puštanje u pogon na Render Platformu	65

6 Zaključak i budući rad	67
Popis literature	69
Indeks slika i dijagrama	70
Dodatak: Prikaz aktivnosti grupe	71

1. Dnevnik promjena dokumentacije

Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Uređena prva stranica, napravljen opis zadatka	Luka Miličević	31.10.2023.
0.2	Ostali zahtjevi, dnevnik promjena, dnevnik sastanaka, tablica aktivnosti - inicijalno postavljanje	Luka Miličević	31.10.2023.
0.3	Funkcionalni zahtjevi - dionici i aktori	Duje Huljev	2.11.2023.
0.4	Obrasci uporabe 1-16	Filip Aleksić, Erik Pužar	2.11.2023.
0.5	Implementacija - korištene tehnologije u proteklom periodu razvoja	Domagoj Sviličić	3.11.2023.
0.6	Entiteti baze podataka	Stjepan Đelekovčan	3.11.2023.
0.7	Obrasci uporabe 17-25	Mate Papak	3.11.2023.
0.8	Arhitektura i dizajn sustava - opis i dijagrami razreda	Luka Miličević	6.11.2023.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	Opis promjene/dodataka	Autori	Datum
0.9	UC dijagrami	Filip Aleksić, Mate Papak, Duje Huljev, Erik Pužar	6.11.2023.
0.10	Sekvencijski dijagram UC1 prva verzija	Domagoj Sviličić	9.11.2023.
0.11	Sekvencijski dijagram UC1 i UC12	Domagoj Sviličić	11.11.2023.
0.12	Dorada relacijskog dijagrama	Stjepan Đelekovčan	12.11.2023.
0.13	Ispravak UC dijagrama	Duje Huljev	12.11.2023.
0.14	Dorada dijagrama klasa - DTO i Controllers	Luka Miličević	14.11.2023.
1.0	Verzija samo s bitnim dijelovima za 1. ciklus	*	17.11.2023.
1.1	Implementacija	*	10.1.2024.
1.2	Dijagram aktivnosti	Domagoj Sviličić	11.1.2024.
1.3	Ažurirane tablice, opisi tablica, dijagram razreda i relacijski dijagram	Stjepan Đelekovčan	15.1.2024.
1.4	Korekcija informacija i dijagrama prema implementaciji	Luka Miličević	16.1.2024.

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Rev.	Opis promjene/dodataka	Autori	Datum
1.5	Dijagram razmještaja, dijagram komponenti, korekcija dijagrama aktivnosti	Domagoj Sviličić	17.1.2024.
1.6	Dopuna korištenih tehnologija	Domagoj Sviličić	17.1.2024.
1.7	Dijagram stanja	Duje Huljev	17.1.2024.
1.8	Popravak detalja i dijagrama	Luka Miličević	17.1.2024.
1.9	Dopuna korištenih tehnologija, ispravak svih use-case dijagrama, dijagrama razmještaja i UC5 sekvencijskog dijagrama, napravljen zaključak, dodana skica kod arhitekture	Domagoj Sviličić	18.1.2024.
1.10	Upute za puštanje u pogon	Luka Miličević	18.1.2024.
1.11	Ispitivanje programske potpore, upis sati i konačna dorada	Luka Miličević, Mate Papak, Filip Aleksić	19.1.2024.

2. Opis projektnog zadatka

Cilj ovog projekta je razviti programsku podršku za stvaranje web aplikacije za promociju i pronalazak zabavnih događaja u gradu, "ConnectiNET". Ova platforma omogućuje korisnicima prijavu u sustav kao posjetitelj ili organizator događaja, čime im se omogućuje pregledavanje i pretraživanje događaja, kao i stvaranje i uređivanje vlastitih događaja. Aplikacija je zamišljena kao više od mjesta za informiranje o aktivnostima, već i kao mjesto za povezivanje zajednice, te je besplatna za korištenje posjetiteljima i organizatorima koji ne organiziraju plaćene događaje.

Prijava zahtjeva unos korisničkog imena i lozinke, a registracija u aplikaciju zahtjeva sljedeće podatke:

- adresa e-pošte
- korisničko ime
- lozinka
- država
- vrsta računa - izbor između posjetitelja i organizatora

Korisnici registracijom i prijavom u sustav imaju pristup početnoj stranici, na kojoj se nalaze događaji u korisnikovom gradu. Početna stranica nudi mogućnosti filtriranja i sortiranja po kategorijama, datumu, lokaciji, cijeni i slično.

Posjetiteljima se putem značajki na početnoj stranici omogućuje pronalazak događaja koji ih zanimaju. Također, moguće je pregledati detalje o događaju, kao što su opis, lokacija, vrijeme, cijena i slično, čime korisnici mogu odlučiti razinu zainteresiranosti za događaj, i istu naznačiti na stranici što čini taj podatak vidljivim drugim korisnicima u obliku broja zainteresiranih posjetitelja. Posjetitelji su u mogućnosti ostavljati recenzije na događaje na kojima su bili u roku od 48 sati nakon završetka događaja. Time se ostvaruje interakcija između organizatora i posjetitelja.

Organizatorima je omogućeno stvaranje i uređivanje vlastitih događaja te unos detalja kao što su opis, slika, kategorija, lokacija i vrijeme. U slučaju da organizator želi kreirati događaj za koji se plaća ulaz, mora se pretplatiti kao premium

organizator. Preplata uključuje mjesecnu članarinu koja se plaća putem PayPal-a ili kreditne kartice. Organizatori mogu i obrisati vlastiti događaj. Na javnim profilima organizatora prikazane su informacije o organizatoru, kao što su opis, slika, događaji koje je organizator kreirao u protekle dvije godine, te poveznice na vlastite web stranice ili društvene mreže.

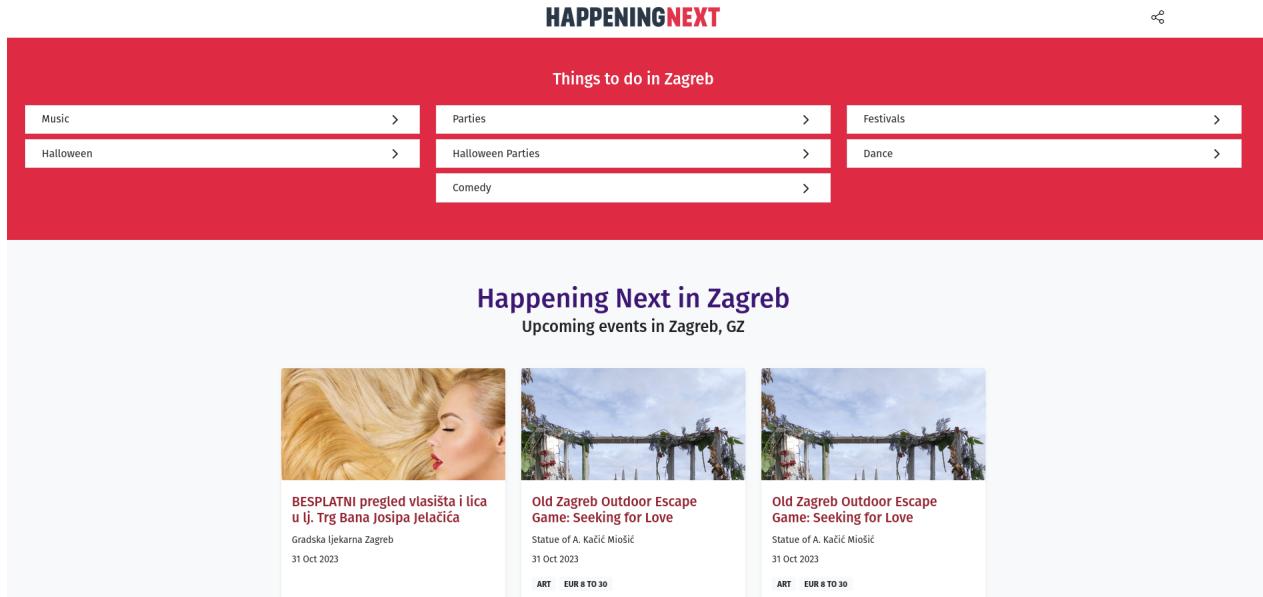
Osim tih mogućnosti, svim korisnicima (organizatorima i posjetiteljima) je omogućen pregled svojih korisničkih profila, te profila drugih korisnika. Korisnici su u mogućnosti i uređivati neke osobne podatke, poput imena, prezimena, adrese e-pošte, lozinke, korisničke slike i opisa. Moguće je i brisanje korisničkog računa.

Administratorima je omogućen pregled svih korisnika i događaja, brisanje korisničkih računa, događaja i recenzija, promjena statusa pretplate organizatorima, promjena cijene pretplate, tj. generalna administracija sustava.

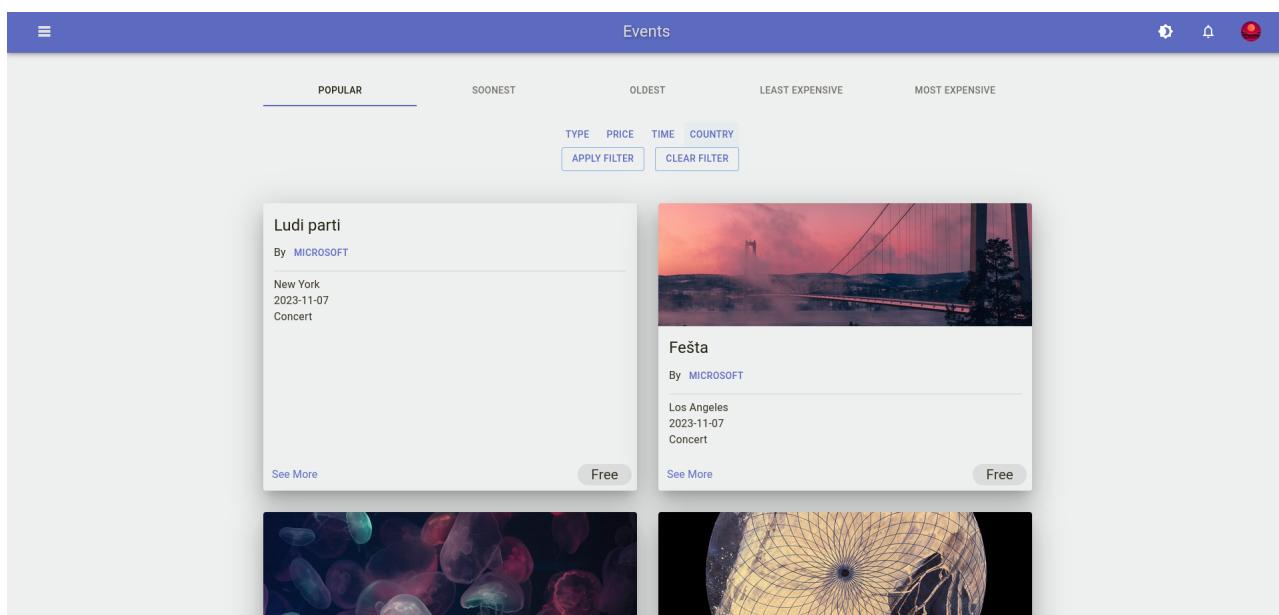
Iako već postoje slične aplikacije, naš sustav predstavlja unikatno rješenje za korisnike koji se žele uistinu bliže povezati sa svojom gradskom zajednicom i aktivno sudjelovati u njoj. To je ostvareno činjenicom da korisnici mogu ostavljati recenzije, označavati zainteresiranost i kreirati vlastite događaje, što su značajke nedostupne u mnogim aplikacijama u sličnom području, neke od kojih su:

- *HappeningNext: <https://happeningnext.com/>*
- *allevents.in: <https://allevents.in/>*
- *Turistička zajednica grada Zagreba: <https://www.infozagreb.hr/>*

Sljedeće je usporedba početne stranice happeningnext i našeg sustava, putem koje se u našem sustavu vidi izražena pažnja na interakciju korisnika:



Slika 2.1: HappeningNext



Slika 2.2: ConnectiNET

ConnectiNET je trenutno dostupan samo na engleskom jeziku, ali je u budućnosti lako moguće dodati podršku za hrvatski jezik. Trenutno je dostupan samo u obliku web aplikacije, ali razvitak mobilne aplikacije bi doveo nove korisnike i omogućio bolju pristupačnost velikom dijelu postojećih korisnika. Također, u planu je dodati mogućnost prijave putem Facebooka i Google računa te integracija s tim platfor-

mama, poput mogućnosti dodavanja događaja u svoj Google kalendar ili direktno dijeljenje događaja na Facebook. Postoji još značajki koje bi u budućnosti unaprijedile sustav za koristnike, poput stranice s kartom na kojoj su prikazani događaji u blizini korisnika, mogućnost slanja poruka između korisnika koji se međusobno "prate", livestream-anje događaja i još mnogo toga. No, trenutno opseg aplikacije je već širok, s obzirom da svim korisnicima u određenim gradovima omogućuje međusobno povezivanje i pronađak raznovrsnih zabavnih događaja u njihovoј blizini.

Podržan je rad više korisnika u stvarnom vremenu te pristup iz javne mreže protokolom HTTPS. Sustav je jednostavan i brz za korištenje, te je u potpunosti siguran i pouzdan.

3. Specifikacija programske potpore

3.1 Funkcionalni zahtjevi

Dionici:

1. Vlasnik
2. Klijenti
 - (a) Organizatori
 - (b) Posjetitelji
3. Administratori
4. Razvojni tim

Aktori i njihovi funkcionalni zahtjevi:

1. Neregistrirani/neprijavljeni korisnik (inicijator) može:
 - (a) prijaviti se u sustav, unijeti korisničko ime i lozinku
 - (b) registrirati se u sustav, stvoriti korisnički račun za koji su mu potrebni korisničko ime, lozinka i e-mail adresa
2. Prijavljeni korisnik (inicijator) može:
 - (a) pregledati popis događaja
 - (b) upravljati načinom prikaza popisa događaja
 - i. filtrirati događaje po kategoriji, lokaciji, vremenu održavanja
 - ii. sortirati događaje po kategoriji, udaljenosti, vremenu održavanja, cijeni, ocjeni
 - (c) pregledati detalje događaja
 - (d) pregledati recenzije drugih posjetitelja
 - (e) odjaviti se iz sustava
 - (f) pregledati osobne podatke
 - (g) uređivati osobne podatke
 - (h) izbrisati svoj korisnički račun

(i) pregledati javni profil pojedinog organizatora

3. Organizator (inicijator) može:

- (a) sve što prijavljeni korisnik može
- (b) stvoriti novi događaj i dodati mu potrebne informacije (opis, lokacija, vremenski raspored, cijena)
- (c) uređivati vlastiti postojeći događaj
 - i. izmijeniti nužne informacije o događaju (opis, lokacija, vremenski raspored, cijena)
 - ii. dodati, obrisati ili izmijeniti izborne informacije o događaju (slika, kategorija i sl.)
- (d) izbrisati vlastiti postojeći događaj
- (e) pregledati statistiku za vlastite događaje
- (f) dobiti uvid u recenzije događaja koje je organizirao
- (g) uređivati ili skruti vlastiti javni profil
- (h) upravljati pretplatom za premium organizatore
 - i. preplatiti se kao premium organizator putem PayPal-a ili kreditne kartice
 - ii. ukinuti preplatu

4. Posjetitelj (inicijator) može:

- (a) sve što prijavljeni korisnik može
- (b) odabrati i/ili promijeniti stupanj interesa za događaj („sigurno dolazim“, „možda dolazim“, „ne dolazim“)
- (c) ostaviti i urediti recenziju za posjećeni događaj koji je završio unutar zadnjih 48h (ocjena, komentar)

5. Administrator (inicijator) može:

- (a) sve što prijavljeni korisnik može
- (b) vidjeti popis svih registriranih korisnika i njihovih osobnih podataka
- (c) dodavati nove administratore
- (d) ukloniti recenzije koje krše pravila korištenja aplikacije
- (e) ukloniti događaje koji krše pravila korištenja aplikacije
- (f) ukloniti korisničke račune koji krše pravila korištenja aplikacije
- (g) ukinuti preplatu premium organizatorima koji krše pravila korištenja aplikacije

(h) postaviti cijenu članstva za premium organizatore

6. Baza podataka (sudionik):

- (a) pohranjuje sve podatke o korisnicima, njihovim detaljima i ovlastima
- (b) pohranjuje sve podatke o događajima, njihovim detaljima i recenzijama

3.1.1 Obrasci uporabe

Opis obrazaca uporabe

UC1 -Registracija

- **Glavni sudionik:** Korisnik
- **Cilj:** Stvoriti korisnicki račun za pristup sustavu
- **Sudionici:** Baza podataka
- **Preduvjet:**
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju za registraciju
 2. Sustav vraća korisniku obrazac za registraciju
 3. Korisnik unosi potrebne korisničke podatke
 4. Sustav provjerava točnost unesenih podataka i da li je korisničko ime već zauzeto
 5. Korisnik prima obavijest o uspješnoj registraciji
- **Opis mogućih odstupanja:**
 - 2.a Odabir vec zauzetog korisničkog imena i/ili e-maila, unos korisničkog podatka u nedozvoljenom formatu ili pruzanje neispravnoga e-maila
 1. Sustav obavjestava korisnika o neuspjelom upisu i vraća ga na stranicu za registraciju
 2. Korisnik mijenja potrebne podatke te zavrsava unos ili odustaje od registracije

UC2 - Prijava

- **Glavni sudionik:** Korisnik
- **Cilj:** Dobiti pristup korisničkom sučelju
- **Sudionici:** Baza podataka
- **Preduvjet:** Registracija
- **Opis osnovnog tijeka:**
 1. Unos korisničkog imena i lozinke
 2. Potvrda o ispravnosti unesenih podataka
 3. Pristup korisničkim funkcijama
- **Opis mogućih odstupanja:**
 - 2.a Neispravno korisničko ime/lozinka
 1. Sustav obavjestava korisnika o neuspjelom upisu i vraća ga na stranicu za prijavu

UC3 - Pregled osobnih podataka

- **Glavni sudionik:** Administrator, organizator, posjetitelj
- **Cilj:** Pregledati osobne podatke
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisnik odabire opciju "Osobni podatci"
 2. Aplikacija prikazuje osobne podatke korisnika

UC4 - Promjena osobnih podataka

- **Glavni sudionik:** Organizator, posjetitelj
- **Cilj:** Promijeniti osobne podatke
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen (vidi UC2)
- **Opis osnovnog tijeka:**
 1. Korisnik odabere opciju za promjenu podataka
 2. Korisnik mijenja svoje osobne podatke
 3. Korisnik sprema promjene
 4. Baza podataka se ažurira
- **Opis mogućih odstupanja:**
 - 2.a Korisnik promijeni svoje osobne podatke, ali ne odabere opciju "Spremi promjenu"
 1. Sustav obavjestava korisnika da nije spremio podatke prije izlaska iz prozora

UC5 - Brisanje računa

- **Glavni sudionik:** Administrator, korisnik
- **Cilj:** Izbrisati svoj korisnički račun
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisnik pregledava osobne podatke (vidi UC3)
 2. Otvara se stranica s osobnim podacima korisnika
 3. Korisnik briše račun
 4. Korisnicki račun se izbriše iz baze podataka

5. Otvara se stranica za prijavu
- **Opis mogućih odstupanja:**
 - 5.a Račun nije pronađen u bazi podataka
 1. Sustav prikazuje poruku "Greška - Račun ne postoji!"

UC6 - Pregled popisa događaja

- **Glavni sudionik:** Administrator, korisnik
- **Cilj:** Pregledati nadolazeće događaje
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisniku se prikazuje popis aktualnih događaja

UC7 - Kontrola pregleda popisa događaja

- **Glavni sudionik:** Administrator, korisnik
- **Cilj:** Prikaz događaja po korisnikovoj želji
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisnik ima mogućost filtriranja događaja po kategorijama, lokacijama, vremenu izvođenja i cijeni događaja
 2. Korisnik ima mogućnost sortiranja događaja po popularnosti (interesu), vremenu izvođenja te cijeni događaja

UC8 - Pregled podataka o događaju

- **Glavni sudionik:** Administrator, korisnik
- **Cilj:** Prikaz detalja o odabranom događaju
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Korisnik odabire događaj (vidi UC6)
 2. Prikazuje mu se detalji o događaju poput naziva, vrste, lokacije, vremena početka, trajanja, foto galerije te popis recenzija

UC9 - Odabir stupnja interesa za događaj

- **Glavni sudionik:** Posjetitelj

- **Cilj:** Odabir interesa
- **Sudionici:** Baza podataka
- **Preduvjet:** Posjetitelj je prijavljen
- **Opis osnovnog tijeka:**
 1. Posjetitelj otvara stranicu s popisom događaja (vidi UC6)
 2. Odabire događaj
 3. Odabire stupanj interesa između „sigurno dolazim“, „možda dolazim“ i „ne dolazim“

UC10 - Ostavljanje recenzije na dogadaj

- **Glavni sudionik:** Posjetitelj
- **Cilj:** Ostaviti recenziju
- **Sudionici:** Baza podataka
- **Preduvjet:** Posjetitelj je prijavljen i nije prošlo 48h od završetka događaja
- **Opis osnovnog tijeka:**
 1. Posjetitelj otvara stranicu s popisom događaja (vidi UC6)
 2. Odabire događaj
 3. Posjetitelj odabire opciju "Ostavi recenziju"
 4. Posjetitelj piše recenziju
 5. Posjetitelj odabire opciju "Potvrди recenziju"
- **Opis mogućih odstupanja:**
 - 5.a Korisnik nije ispravno popunio obrazac za recenziju
 1. Sustav prikazuje poruku "Neispravno popunjeno obrazac"

UC11 - Uvid u recenzije drugih posjetitelja

- **Glavni sudionik:** Organizator, posjetitelj
- **Cilj:** Pregledati recenzije
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator/posjetitelj je prijavljen
- **Opis osnovnog tijeka:**
 1. Posjetitelj otvara stranicu s popisom događaja (vidi UC6)
 2. Odabire događaj
 3. Posjetitelj odabire opciju "Prikaži sve recenzije"
- **Opis mogućih odstupanja:**
 - 3.a Događaj nema ni jednu recenziju
 1. Sustav prikazuje poruku "Nema recenzija za ovaj događaj"

UC12 - Organizacija vlastitog događaja

- **Glavni sudionik:** Organizator
- **Cilj:** Organizacija događaja
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen i ima premium verziju
- **Opis osnovnog tijeka:**
 1. Organizator odabire opciju "Kreiraj novi događaj"
 2. Organizator upisuje detalje o događaju
 3. Organizator odabire opciju "Kreiraj događaj"
 4. Ukoliko je obrazac ispravno popunjeno ispisuje se poruka "Događaj uspješno kreiran"
- **Opis mogućih odstupanja:**
 - 4.a nisu popunjena nužna polja za stvaranje događaja
 1. Sustav prikazuje poruku "Neispravno popunjeno obrazac"

UC13 - Uređivanje vlastitog događaja

- **Glavni sudionik:** Organizator
- **Cilj:** Uređivanje događaja
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen i ima kreirani događaj
- **Opis osnovnog tijeka:**
 1. Organizator odabire opciju "Moji događaji"
 2. Organizator odabire događaj
 3. Organizator odabire opciju "Uredi događaj"
 4. Organizator uređuje detalje o događaju
 5. Organizator odabire opciju "Pohrani promjene"
- 4.a nisu popunjena nužna polja za stvaranje događaja
 1. Sustav prikazuje poruku "Neispravno popunjeno obrazac"

UC14 - Pregled podataka o vlastitom događaju

- **Glavni sudionik:** Organizator
- **Cilj:** Pregled podataka o kreiranom događaju
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen i ima kreirani događaj
- **Opis osnovnog tijeka:**

1. Organizator otvara stranicu s popisom događaja
 2. Organizator odabire opciju "Moji događaji"
 3. Organizator odabire događaj
- **Opis mogućih odstupanja:**
 - 2.a Ne postoji niti jedan kreirani događaj
 1. Sustav prikazuje poruku "Nema događaja za prikazati"

UC15 - Pregled profila organizatora

- **Glavni sudionik:** Administrator, korisnik
- **Cilj:** Pregled podataka o organizatoru događaja
- **Sudionici:** Baza podataka
- **Preduvjet:** Korisnik je prijavljen
- **Opis osnovnog tijeka:**
 1. Posjetitelj otvara stranicu s popisom događaja
 2. Posjetitelj odabire događaj
 3. Posjetitelj odabire opciju "Posjeti profil organizatora događaja"

UC16 - Pretplata

- **Glavni sudionik:** Organizator
- **Cilj:** Pretplacivanje
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen i nije pretplaćen
- **Opis osnovnog tijeka:**
 1. Organizator odabire opciju "Osobni podatci"
 2. Organizator odabire opciju "Pretplata"
 3. Organizator odabire opciju "Pretplati se na ConnectiNET Premium"
 4. Organizator odabire vrstu plaćanja
 5. Organizator upisuje podatke potrebne za plaćanje
 6. Organizator odabire opciju "Potvrди pretplatu"
- **Opis mogućih odstupanja:**
 - 5.a Uneseni su neispravni bankovni podatci
 1. Sustav prikazuje poruku "Neispravni podaci o plaćanju"
 - 6.a Stanje na bankovnom računu je manje od iznosa pretplate

UC17 - Otkazivanje pretplate

- **Glavni sudionik:** Organizator

- **Cilj:** Otkazivanje pretplate
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator je prijavljen i ima pretplatu
- **Opis osnovnog tijeka:**
 1. Organizator odabire opciju "Osobni podatci"
 2. Organizator odabire opciju "Pretplate"
 3. Organizator odabire opciju "Otkaži Premium pretplatu"
 4. Organizator odabire opciju "Da, otkaži pretplatu"
 5. Pretplata je otkazana
- **Opis mogućih odstupanja:**
 - 4.a Organizator ipak ne želi otkazati pretplatu
 1. Organizator odabire opciju "Poništi"

UC18 - Primanje obavijesti o događajima

- **Glavni sudionik:** Posjetitelj
- **Cilj:** Primanje obavijesti o događajima
- **Sudionici:** Baza podataka
- **Preduvjet:** Posjetitelj je prijavljen
- **Opis osnovnog tijeka:**
 1. Posjetitelj odabire opciju "Osobni podatci"
 2. Posjetitelj odabire opciju "Obavijesti"
 3. Posjetitelj odabire željene opcije za primanje obavijesti
 - Primanje obavijesti putem e-pošte
 - Primanje obavijesti putem obavijesti na pregledniku

UC19 - Postavljanje cijene pretplate

- **Glavni sudionik:** Administrator
- **Cilj:** Postavljanje cijene pretplate
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju "Administratorske postavke"
 2. Administrator odabire opciju "Premium pretplate"
 3. Administrator upisuje željenu cijenu (u eurima) na mjesto "Cijena:"

UC20 - Uklanjanje korisnika

- **Glavni sudionik:** Administrator
- **Cilj:** Uklanjanje korisnika
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen i korisnik postoji
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju "Administratorske postavke"
 2. Administrator odabire opciju "Korisnici"
 3. Administrator upisuje ime korisnika u tražilicu
 4. Svi korisnici s navedenim pretraživanjem u imenu se prikazuju
 5. Administrator odabire željenog korisnika
 6. Administrator odabire opciju "Ukloni korisnika"
 7. Administrator odabire opciju "Da, ukloni korisnika"
 8. Korisnik je uklonjen
- **Opis mogućih odstupanja:**
 - 3.a Administrator odabire filter "Organizator" ili "Posjetitelj" umjesto "Svi"
 - 3.b Administrator odabire vrstu sortiranja rezultata
 - 4.a Korisnik ne postoji
 - 7.a Administrator ipak ne želi ukloniti korisnika
 1. Administrator odabire opciju "Poništi"

UC21 - Uklanjanje recenzija

- **Glavni sudionik:** Administrator
- **Cilj:** Uklanjanje recenzija
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen i recenzija postoji
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju "Administratorske postavke"
 2. Administrator odabire opciju "Korisnici"
 3. Administrator upisuje ime korisnika u tražilicu
 4. Svi korisnici s navedenim pretraživanjem u imenu se prikazuju
 5. Administrator odabire željenog korisnika
 6. Administrator odabire opciju "Recenzije"
 7. Korisnikove recenzije za sve događaje se prikazuju
 8. Administrator odabire opciju "Ukloni recenziju" na nekoj od recenzija
 9. Administrator odabire opciju "Da, ukloni recenziju"
 10. Recenzija je uklonjena

- **Opis mogućih odstupanja:**
 - 3.a Administrator odabire filter "Organizator" ili "Posjetitelj" umjesto "Svi"
 - 3.b Administrator odabire vrstu sortiranja rezultata
 - 4.a Korisnik ne postoji

11.a Administrator ipak ne želi ukloniti korisnika

 1. Administrator odabire opciju "Poništi"

UC22 - Uklanjanje pretplate

- **Glavni sudionik:** Administrator
- **Cilj:** Uklanjanje pretplate
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen i organizacija postoji
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju "Administratorske postavke"
 2. Administrator odabire opciju "Korisnici"
 3. Administrator odabire filter "Organizator" (preporučeno)
 4. Administrator upisuje ime organizatora u tražilicu
 5. Svi organizatori s navedenim pretraživanjem u imenu se prikazuju
 6. Administrator odabire željenog organizatora
 7. Administrator odabire opciju "Preplata"
 8. Administrator odabire opciju "Ukloni Premium pretplatu"
 9. Administrator odabire opciju "Da, ukloni pretplatu"
 10. Pretplata je uklonjena
- **Opis mogućih odstupanja:**
 - 3.b Administrator odabire vrstu sortiranja rezultata
 - 5.a Organizator ne postoji
 - 8.a Organizator već nema Premium pretplatu
 - 9.a Administrator ipak ne želi ukloniti pretplatu
 1. Administrator odabire opciju "Poništi"

UC23 - Uklanjanje događaja

- **Glavni sudionik:** Administrator
- **Cilj:** Uklanjanje događaja
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen i događaj postoji
- **Opis osnovnog tijeka:**

1. Administrator odabire opciju "Administratorske postavke"
 2. Administrator odabire opciju "Korisnici"
 3. Administrator odabire filter "Organizator" (preporučeno)
 4. Administrator upisuje ime organizatora u tražilicu
 5. Svi organizatori s navedenim pretraživanjem u imenu se prikazuju
 6. Administrator odabire željenog organizatora
 7. Administrator odabire opciju "Događaji"
 8. Administrator upisuje ime događaja u tražilicu
 9. Korisnikova recenzija odabranog događaja se prikazuje
 10. Administrator odabire opciju "Ukloni događaj"
 11. Administrator odabire opciju "Da, ukloni događaj"
 12. Događaj je uklonjen
- **Opis mogućih odstupanja:**
 - 3.b Administrator odabire vrstu sortiranja rezultata
 - 5.a Organizator ne postoji
 - 9.a Administrator ipak ne želi ukloniti pretplatu
 1. Administrator odabire opciju "Poništi"

UC24 - Dodavanje administratora

- **Glavni sudionik:** Administrator
- **Cilj:** Dodavanje administratora
- **Sudionici:** Baza podataka
- **Preduvjet:** Administrator je prijavljen
- **Opis osnovnog tijeka:**
 1. Administrator odabire opciju "Administratorske postavke"
 2. Administrator odabire opciju "Administratori"
 3. Administrator odabire opciju "Dodaj novog administratora"
 4. Administrator upisuje email novog administratora
 5. Administrator odabire opciju "Dodaj administratora"
 6. Administrator odabire opciju "Da, dodaj administratora"
 7. Administrator je dodan
- **Opis mogućih odstupanja:**
 - 6.a Administrator ipak ne želi dodati administratora
 1. Administrator odabire opciju "Poništi"

UC25 - Uređivanje javnog profila

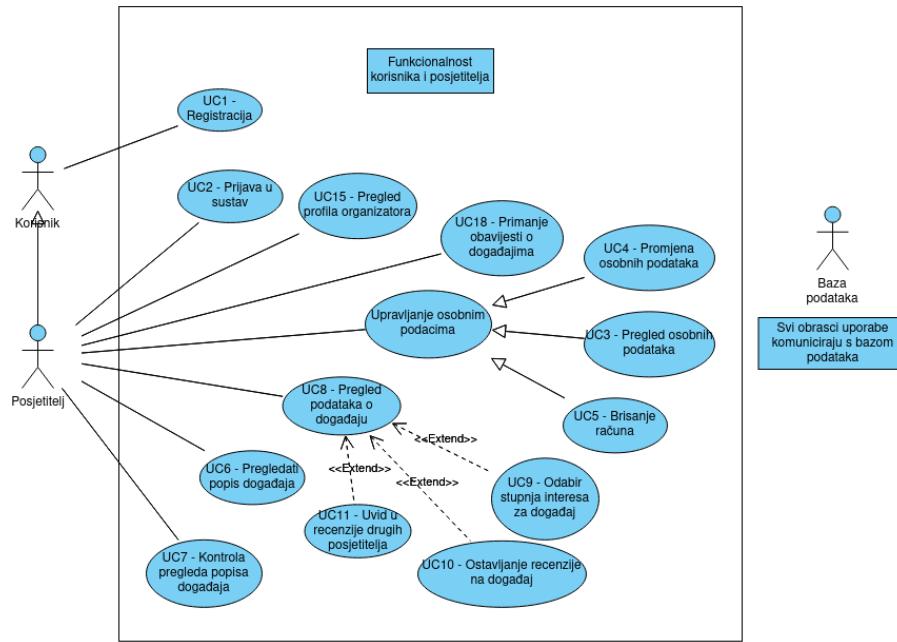
- **Glavni sudionik:** Organizator
- **Cilj:** Uređivanje javnog profila
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator ima kreiran javni profil
- **Opis osnovnog tijeka:**
 1. Organizator odabire opciju "Osobni podatci"
 2. Organizator unosi podatke o sebi
 3. Organizator odabire opciju "Potvrди promjene"
- **Opis mogućih odstupanja:**
 - 5.a Organizator je napravio neispravne promjene
 1. Sustav prikazuje poruku "Neispravno popunjeno obrazac"

UC26 - Skrivanje javnog profila

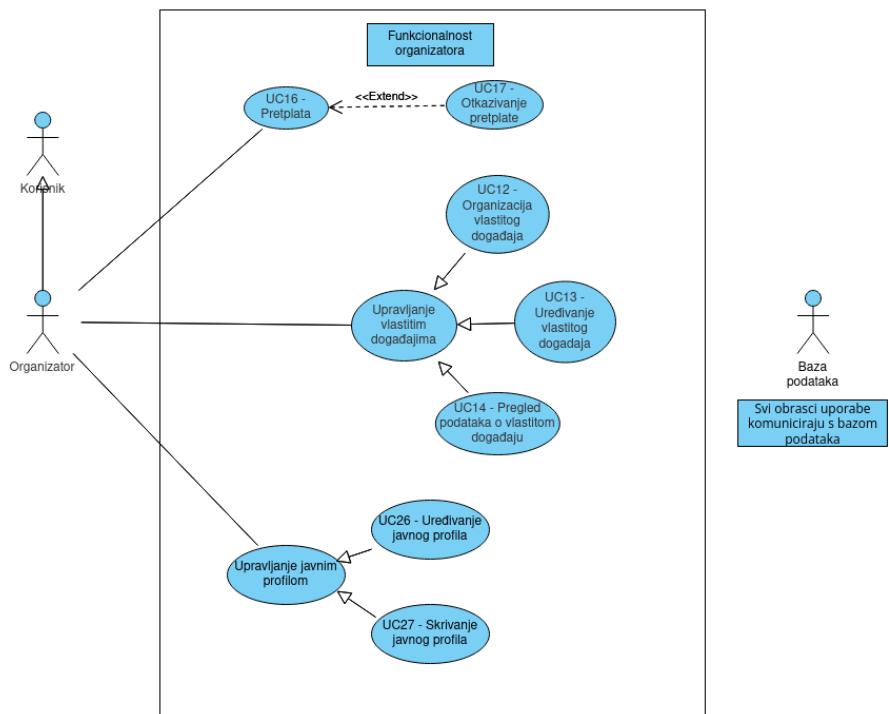
- **Glavni sudionik:** Organizator
- **Cilj:** Skrivanje javnog profila
- **Sudionici:** Baza podataka
- **Preduvjet:** Organizator ima kreiran javni profil
- **Opis osnovnog tijeka:**
 1. Organizator odabire opciju "Osobni podatci"
 2. Organizator odabire opciju "Sakrij javni profil"
 3. Organizator odabire opciju "Potvrdi"
- **Opis mogućih odstupanja:**
 - 4.a Organizator ipak ne želi izbrisati skriti profil
 1. Organizator odabire opciju "Poništi"

Dijagrami obrazaca uporabe

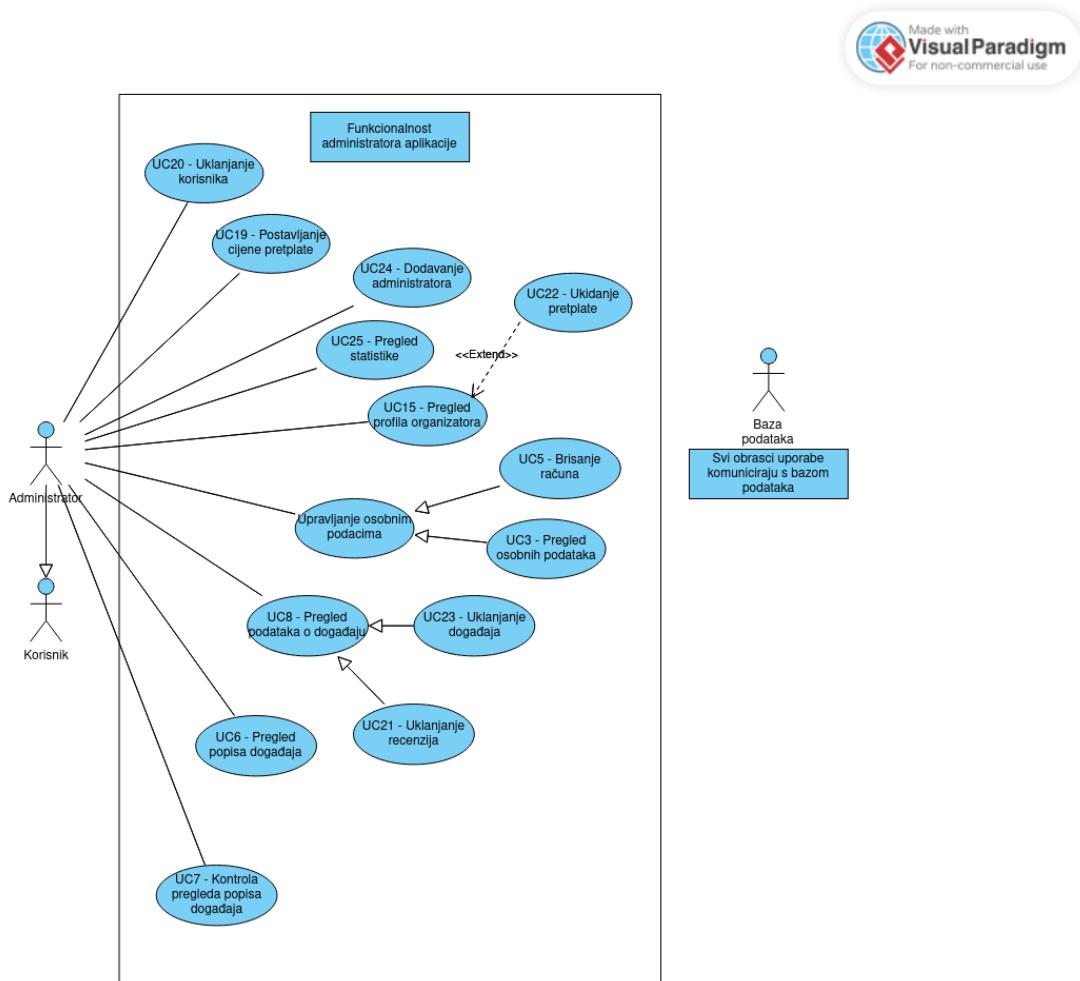
Made with
Visual Paradigm
 For non-commercial use



Slika 3.1: Dijagram obrasca uporabe, funkcionalnost posjetitelja



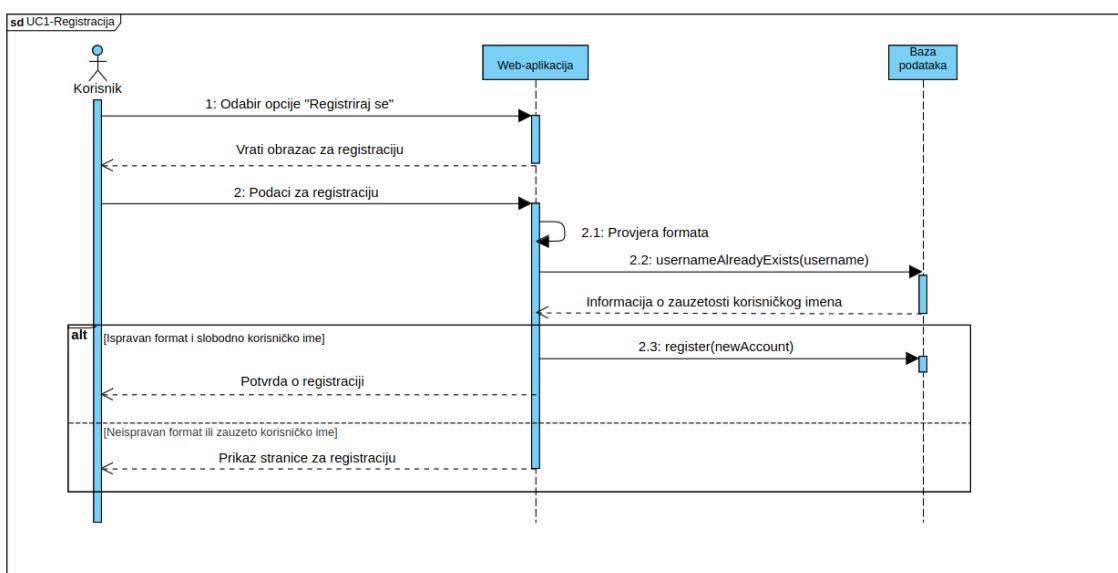
Slika 3.2: Dijagram obrasca uporabe, funkcionalnost organizatora



3.1.2 Sekvencijski dijagrami

UC1-Registracija

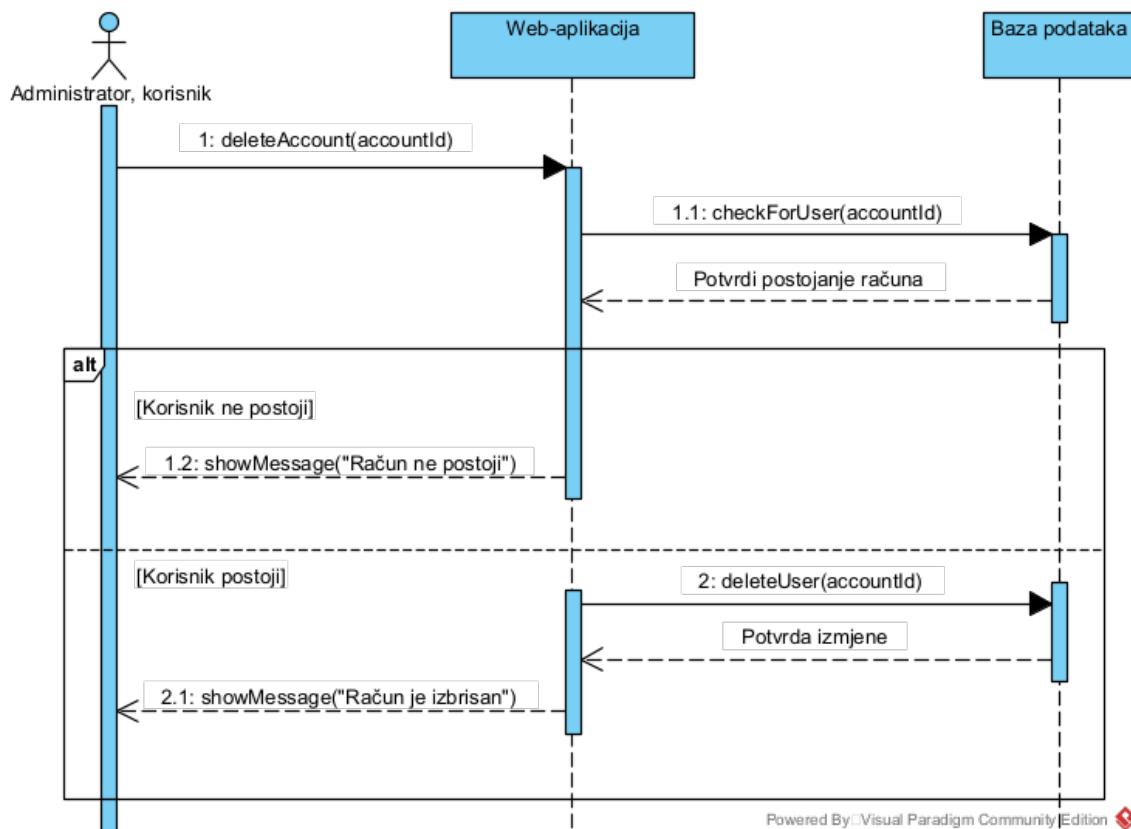
Korisnik odabire opciju za registraciju. Poslužitelj mu vraća obrazac za ispunjavanje osobnih podataka. Nakon unosa poslužitelj provjerava je li korisnik unio neki podatak u neispravnom formatu ili je pružio neispravnu adresu e-pošte. Nakon toga komunicira s bazom podataka kako bi provjerio da li slučajno postoji isto korisničko ime kao ono koje korisnik namjerava unijeti. Ukoliko korisničko ime nije zauzeto i ostali podaci su ispravni dodaje se novi korisnički račun u bazu podataka i korisniku se vraća potvrda o uspješnoj registraciji. U suprotnom prikazuje se početna stranica za registraciju.



Slika 3.4: Sekvencijski dijagram za UC1

UC5 - Brisanje računa

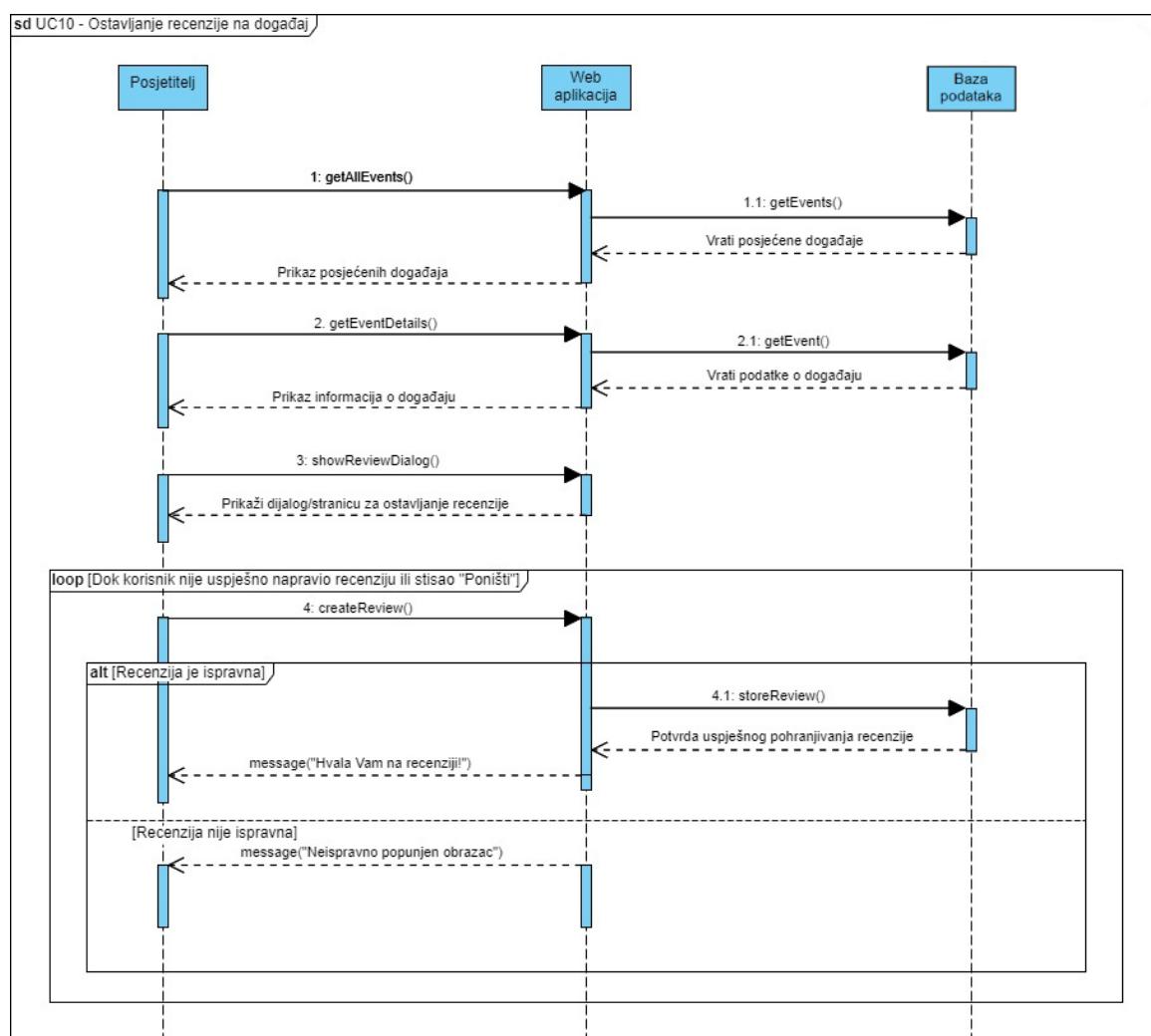
Administrator šalje zahtjev za brisanje računa po volji (korisnik šalje za svoj račun). Poslužitelj provjerava postoji li taj račun u bazi podataka. Baza vraća signal kojim negira ili potvrđuje postojanje računa. U slučaju nepostojanja računa, administratoru/korisniku se prikazuje poruka da navedeni račun ne postoji. Inače poslužitelj miče navedeni račun iz baze podataka i šalje administratoru/korisniku poruku da je račun uspješno izbrisano.



Slika 3.5: Sekvencijski dijagram za UC5

UC10 - Ostavljanje recenzije na događaj

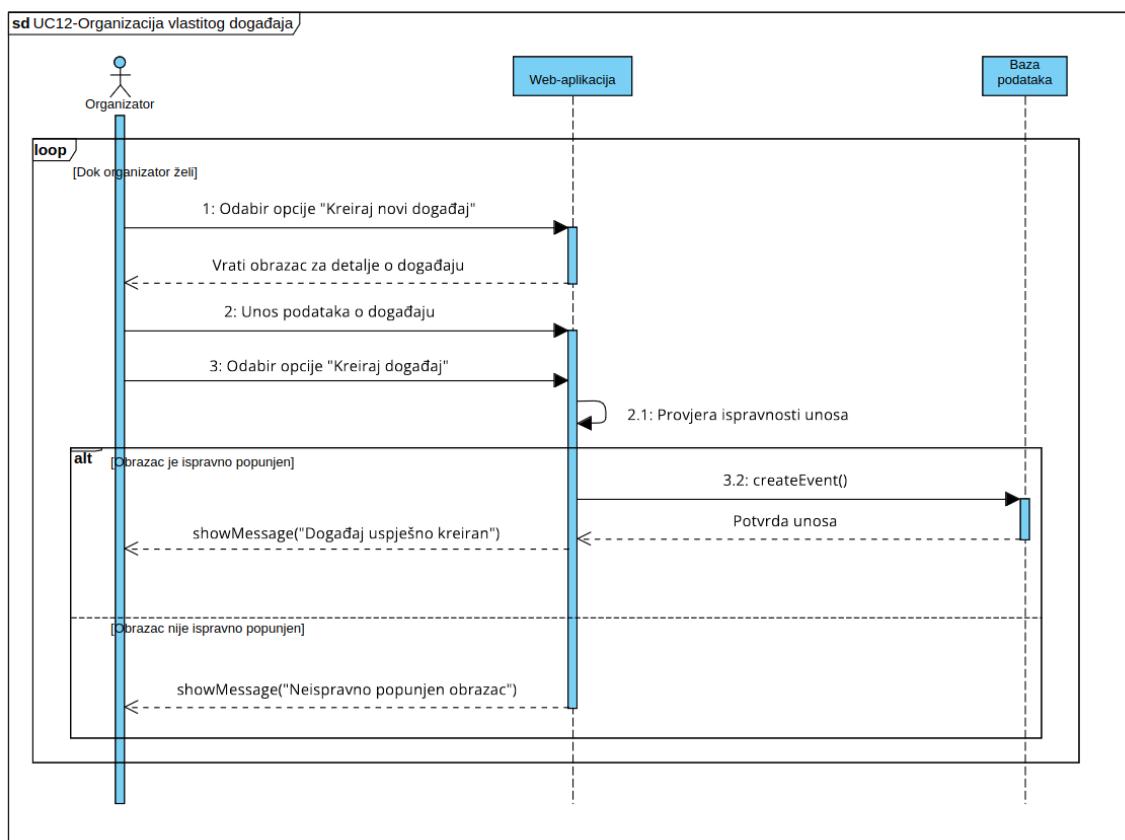
Posjetitelj šalje zahtjev za prikaz događaja na kojima je bio (kartica "Moji događaji") kako bi mogao odabrati događaj na kojem želi ostaviti recenziju. Poslužitelj dohvata takve događaje iz baze podataka i prikazuje ih. Odabriom događaja, poslužitelj dohvata podatke o događaju i prikazuje ih posjetitelju. Posjetitelj odabire opciju "Ostavi recenziju" te mu se prikazuje obrazac za ostavljanje recenzije. Posjetitelj unosi ocjenu i komentar te odabire opciju "Potvrди recenziju". Poslužitelj sprema recenziju u bazu podataka i prikazuje poruku o uspješnom ostavljanju recenzije. Ako posjetitelj nije odabrao ocjenu ili nije unio komentar, prikazuje se poruka o neispravno popunjenoj obrazcu. Ako se posjetitelj predomisli, može odabrati opciju "Poništi" te se vraća na stranicu s podacima o događaju.



Slika 3.6: Sekvencijski dijagram za UC10

UC12 - Uređivanje vlastitog događaja

Organizator objavljuje vlastiti događaj odabirom opcije "Kreiraj novi događaj". Potom u dobiveni obrazac upisuje detalje o događaju koji želi organizirati. Nakon provjere ispravnosti unesenih podataka poslužitelj dobivene podatke prosljeđuje bazi gdje ostaju zapamćeni. Organizator prima poruku "Događaj uspješno kreiran" te ponovo može pristupiti objavljivanju novog događaja. Ako obrazac nije ispravno popunjjen, poslužitelj upozorava organizatora porukom "Neispravno popunjjen obrazac".



Slika 3.7: Sekvencijski dijagram za UC12

3.2 Ostali zahtjevi

- Sustav treba omogućiti rad više korisnika u stvarnom vremenu
- Korisničko sučelje i sustav moraju podržavati hrvatsku abecedu (dijakritičkeznakove) pri unosu i prikazu tekstualnog sadržaja
- 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
- Neispravno korištenje korisničkog sučelja ne smije narušiti funkcionalnost i rad sustava
- Sustav treba biti jednostavan za korištenje, korisnici trebaju znati koristiti korisničko sučelje bez opširnih uputa
- Nadogradnja sustava ne smije narušavati postojeće funkcionalnosti sustava
- Sustav kao valutu koristi Euro
- Veza s bazom podataka mora biti kvalitetno zaštićena, brza i otporna na vanjske greške
- Pristup sustavu mora biti omogućen iz javne mreže pomoću HTTPS

4. Arhitektura i dizajn sustava

Arhitektura se može podijeliti na tri podsustava:

- Web poslužitelj
- Web aplikacija
- Baza podataka

Web Klijent je program koji omogućuje pregled web-stranica i multimedijskih sadržaja na istima s korisničkog računala. Izvorni kod internetskih stranica se interpretira na web pregledniku i prikazuje korisniku na pristupačan način i služi kao glavna pristupna točka aplikaciji s korisničke strane. Web preglednik je također zadužen za komunikaciju s web poslužiteljem putem zahtjeva.

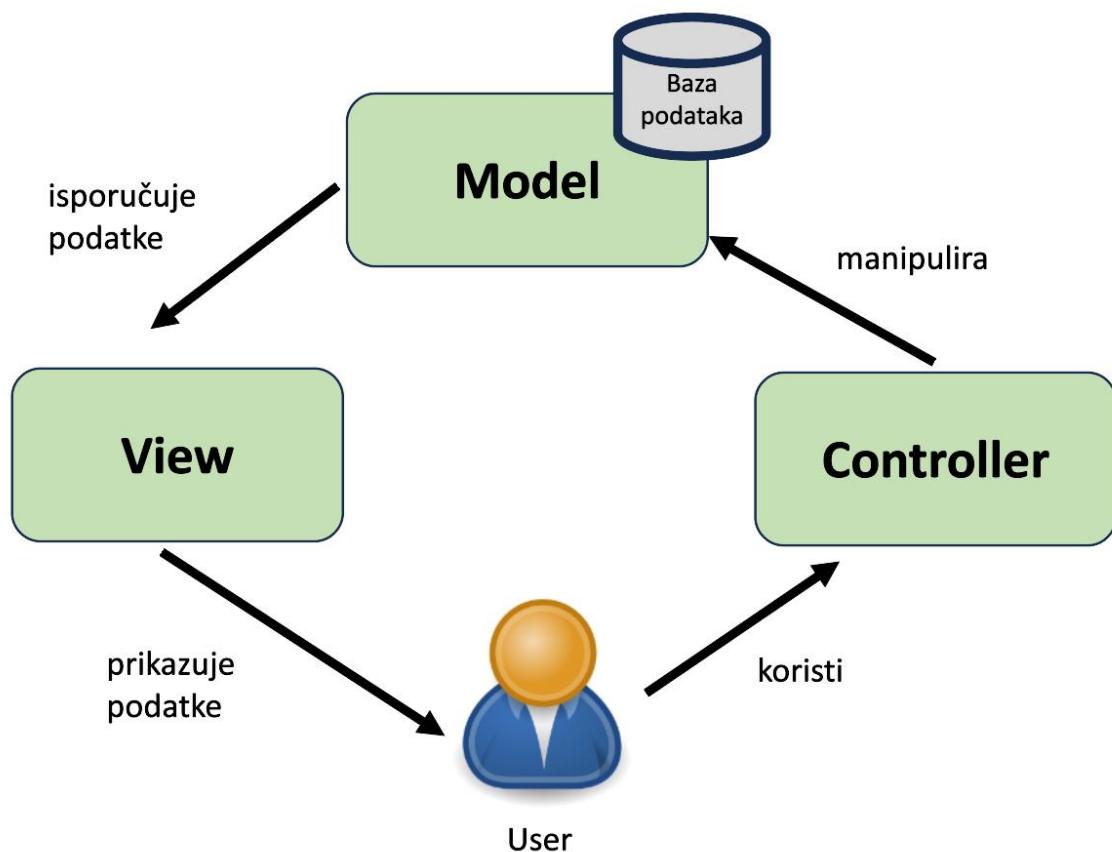
Web poslužitelj je program koji obrađuje zahtjeve klijenata i šalje im odgovor, time omogućavajući klijentu komunikaciju s aplikacijom. Web poslužitelj je zadužen za obradu zahtjeva i slanje odgovora putem HTTP-a (engl.*Hyper Text Transfer Protocol*), standardnog protokola za prijenos informacija na webu.

Web aplikacija je program kojemu je glavna svrha pružanje funkcionalnosti i usluga korisniku putem web preglednika u obliku HTML dokumenata. Web aplikacija je izgrađena na web poslužitelju i po potrebi komunicira s bazom podataka.

Baza podataka je sustav za pohranu podataka.

Programski jezici u kojem je web aplikacija izrađena su Python zajedno s Flask web okvirom, te JavaScript zajedno s React web okvirom. Python je interpretirani, objektno orijentirani programski jezik visoke razine. Flask je mikro web okvir za Python koji omogućuje brzo i jednostavno kreiranje web aplikacija. JavaScript je interpretirani, dinamički programski jezik visoke razine. React je JavaScript biblioteka za izgradnju korisničkih sučelja. Odabранo razvojno okruženje je Microsoft Visual Studio Code. Arhitektura sustava temeljiti će se na MVC arhitekturi (engl. *Model-View-Controller*). MVC dozvoljava nezavisan rad na različitim dijelovima aplikacije, što omogućava brži razvoj i održavanje cijelog sustava. Njezini dijelovi su:

- **Model** - Komponenta sustava odgovorna za pohranu i dohvatanje podataka. Predstavljena je dinamičkim strukturama podataka. Ima interakciju s Controller-om, od kojeg prima ulazne podatke te kojem pruža podatke za prikaz.
- **View** - Komponenta sustava odgovorna za prikaz podataka korisniku. Može biti formatirana u obliku HTML dokumenata, kao JSON ili u drugom formatu. Ima interakciju s Controller-om, od kojeg prima podatke za prikaz.
- **Controller** - Komponenta sustava koja prima ulazne podatke i proslijeđuje ih Model-u ili View-u. Zadužena je za korisničke zahtjeve i odgovore te obavlja interakciju s ostalim komponentama sustava.



Slika 4.1: Skica osnovne arhitekture sustava

4.1 Baza podataka

Za naš sustav koristit ćemo relacijsku bazu podataka čija je struktura pogodna za modeliranje stvarnog svijeta. Gradivna jedinka baze je relacija, odnosno tablica koja je definirana svojim imenom i skupom atributa. Zadaća baze podataka je brza i jednostavna pohrana, izmjena i dohvata podataka za obradu. Baza podataka naše aplikacije sastoji se od sljedećih entiteta:

- Račun
- Posjetitelj
- Organizator
- Preplata
- Plaćanje
- Događanje
- Interes
- Recenzija
- Media-Događanje
- Država
- Vrsta Događanja
- Obavijest-Vrsta-Događanja
- Obavijest-Država
- Podatak

4.1.1 Opis tablica

Račun

Ovaj entitet sadržava sve osnovne informacije o registriranom korisniku. Sadrži atribute: identifikator računa, korisničko ime, lozinka, e-mail, profilna slika, šifra država(ISO3), pripadna uloga. Ovaj entitet u vezi je *One-to-One* s entitetima Posjetitelj i Organizator preko atributa accountId i u vezi *Many-to-One* s entitetom Država preko atributa countryCode.

Account		
accountId	INT	jedinstveni identifikator računa

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Account		
username	VARCHAR	jedinstveni identifikator korisnika
eMail	VARCHAR	E-Mail korisnika
passwordHash	VARCHAR	raspršeno kriptirana lozinka korisnika
profileImage	VARCHAR	lokalna adresa na profilnu sliku korisnika
countryCode	CHAR(3)	jedinstveni id države kojoj račun pripada u formatu 3 slova (Država.countryCode)
roleId	INT	šifra uloge pridružene korisniku

Posjetitelj

Ovaj entitet specijalizacija je entiteta Račun namijenjena za "obične" korisnike. Sadrži atributе: identifikator računa, ime i prezime. Ovaj entitet u vezi je *One-to-One* s entitetom Račun i u vezi *One-to-Many* s entitetima Interes, Recenzija, Opcija-Država i Opcija-Vrsta-Događanja preko atributa accountId.

Visitor		
accountId	INT	jedinstveni identifikator računa (Račun.accountId)
firstName	VARCHAR	ime posjetitelja
lastName	VARCHAR	prezime posjetitelja

Organizator

Ovaj entitet specijalizacija je entiteta Račun namijenjena za korisnike koji su organizatori. Sadrži atributе: identifikator računa, ime organizatora, vidljivost profila, poveznica na društvene mreže organizatora. Ovaj entitet u vezi je *One-to-One* s entitetom Račun i u vezi *One-to-Many* s entitetima s entitetima Plaćanje, Pretplata i Događanje preko atributa accountId.

Organizer		
accountId	INT	jedinstveni identifikator računa (Račun.accountId)
organizerName	VARCHAR	naziv organizatora ili organizacije
hidden	BOOLEAN	zastavica koja govori o javnoj vidljivosti profila organizatora
socials	VARCHAR	poveznica na društvene mreže organizatora.

Preplata

Ovaj entitet opisuje preplatu koju je organizator nekad imao te koja može biti trenutno aktivna. Sadrži atribute: identifikator preplate, datum početka preplate, datum isteka preplate, identifikator računa organizatora. Ovaj entitet u vezi je *Many-to-One* s entitetom Organizator preko atributa accountId.

Subscription		
subscriptionId	INT	jedinstveni identifikator instance preplate
accountId	INT	identifikator organizatora na kojeg se preplata odnosi (Organizator.accountId)
startDate	DATE	datum početka preplate
expireDate	DATE	datum isteka preplate

Plaćanje

Ovaj entitet opisuje plaćanje koje je organizator napravio prema našoj aplikaciji. Sadrži atribute: identifikator plaćanja, identifikator organizatora, datum plaćanja, iznos, metoda plaćanja. Ovaj entitet u vezi je *Many-to-One* s entitetom Organizator preko atributa accountId.

Payment		
paymentId	INT	jedinstveni identifikator plaćanja
accountId	INT	identifikator organizatora na kojeg se plaćanje odnosi (Organizator.accountId)
date	DATETIME	datum i vrijeme plaćanja
amount	FLOAT	plaćen iznos
payment-Method	VARCHAR	način uplate

Događanje

Ovaj entitet opisuje događanje organizirano od strane organizatora. Sadrži atribute: identifikator događanja, identifikator organizatora, naziv, opis, državu, grad, lokaciju, datum i vrijeme, cijenu, naslovnu sliku događanja, vrstu događanja, trajanje. Ovaj entitet u vezi je *Many-to-One* s entitetima Organizator i Vrsta događanja preko atributa accountId i s entitetom Država preko atributa countryCode, vezi *One-to-Many* s entitetom Recenzija preko atributa eventId, vezi *Many-to-Many* s entitetom Posjetitelj preko veze Interes i atributa eventId te u vezi *One-to-Many* s entitetom Media-Događanje također preko atributa eventId.

Event		
eventId	INT	jedinstveni identifikator događanja
accountId	INT	identifikator organizatora koji organizira događanje (Organizator.accountId)
title	VARCHAR	naziv događanja
description	VARCHAR	opis događanja
price	FLOAT	cijena događanja
display-ImageSource	VARCHAR	lokalna adresa naslovne slike događanja
dateTime	DATETIME	vrijeme i datum događanja
countryCode	CHAR(3)	šifra države u kojoj se događanje održava (Država.countryCode)
city	VARCHAR	grad u kojem se događanje održava
location	VARCHAR	lokacija na kojoj se događanje održava
price	FLOAT	cijena događanja
duration	INTERVAL	trajanje događanja
eventType	INT	identifikator vrste događanja (EventType.typeId)

Interes

Ovaj entitet predstavlja *Many-to-Many* vezu interesa od strane Posjetitelja prema Događanju. Sadrži atribute: identifikator događanja, identifikator računa posje-

titelja, stupanj zainteresiranosti. Ovaj entitet u vezi je *Many-to-One* s entitetima Posjetitelj i Događanje preko atributa accountId i eventId.

Interest		
eventId	INT	identifikator događanja na koje se interes odnosi (Događanje.eventId)
accountId	INT	identifikator posjetitelja na kojeg se interes odnosi (Posjetitelj.accountId)
degreeOfInterest	INT	stupanj interesa

Recenzija

Ovaj entitet modelira recenzije ostavljene od strane Posjetitelja za pojedina Događanja. Sadrži atrbute: jedinstveni identifikator recenzije, identifikator događanja, identifikator računa posjetitelja, komentar, datum, vrijeme. Ovaj entitet u vezi je *Many-to-One* s entitetima Događanje i Posjetitelj preko atributa eventId i accountId.

Review		
reviewId	INT	jedinstveni identifikator ostavljene recenzije
accountId	INT	identifikator posjetitelja koji je ostavio recenziju (Posjetitelj.accountId)
eventId	INT	identifikator događanja na koje se recenzija odnosi (Događanje.eventId)
comment	VARCHAR	ostavljen komentar
dateTime	DATETIME	vrijeme i datum ostavljene recenzije

Media-Događanje

Ovaj entitet sprema lokalne adrese foto i video sadržaja koje pripada događanju. Sadrži atrbute: jedinstveni identifikator recenzije, identifikator događanja, identifikator računa posjetitelja, komentar, datum, vrijeme. Ovaj entitet u vezi je *Many-to-One* s entitetima Događanje i Posjetitelj preko atributa eventId i accountId.

EventMedia		
mediaId	INT	jedinstveni identifikator materijala
eventId	INT	identifikator događanja na koje se materijal odnosi (Događanje.eventId)
mediaType	VARCHAR	vrsta sadržaja; slika ili video
mediaSource	VARCHAR	lokalna adresa sadržaja

Država

Ovaj entitet modelira državu sa pripadnim šiframa i nazivom države. Sadrži atribute: identifikator države, sekundarni identifikator države, ime države. Ovaj entitet u vezi je *One-to-Many* s entitetima Događanje i račun preko atributa countryCode.

Country		
countryCode	CHAR(3)	jedinstveni identifikator države sastavljen od 3 slova
code	CHAR(2)	sekundarni jedinstveni identifikator države sastavljen od 2 slova
name	VARCHAR	naziv države

Vrsta Događanja

Ovaj entitet modelira vrstu događanja koji je definiran za svako Događanje. Sadrži atribute: identifikator vrste, naziv vrste. Ovaj entitet u vezi je *One-to-Many* s entitetima Događanje i Obavijest-Vrsta-Događanja preko atributa typeId.

EventType		
typeId	INT	jedinstveni identifikator vrste događanja
typeName	VARCHAR	naziv vrste događanja

Obavijest-Država

Ovaj entitet spremi informaciju o tome koji korisnik ima preference za događanja

u kojim državama. Sadrži atribute: identifikator posjetitelja, identifikator države. Ovaj entitet u vezi je *Many-to-One* s entitetom Posjetitelj preko atributa accountId te s entitetom Država preko atributa countryCode.

NotificationCountry		
accountId	INT	jedinstveni identifikator posjetitelja (Posjetitelj.accountId)
countryCode	CHAR(3)	identifikator države (Country.countryCode). (Primarni ključ je uređeni par accountId, countryCode)

Obavijest-Vrsta-Događanja

Ovaj entitet sprema informaciju o tome koji korisnik ima preference za vrstu događanja. Sadrži atribute: identifikator posjetitelja, identifikator vrste događanja. Ovaj entitet u vezi je *Many-to-One* s entitetom Posjetitelj preko atributa accountId te s entitetom Vrsta Događanja preko atributa eventType.

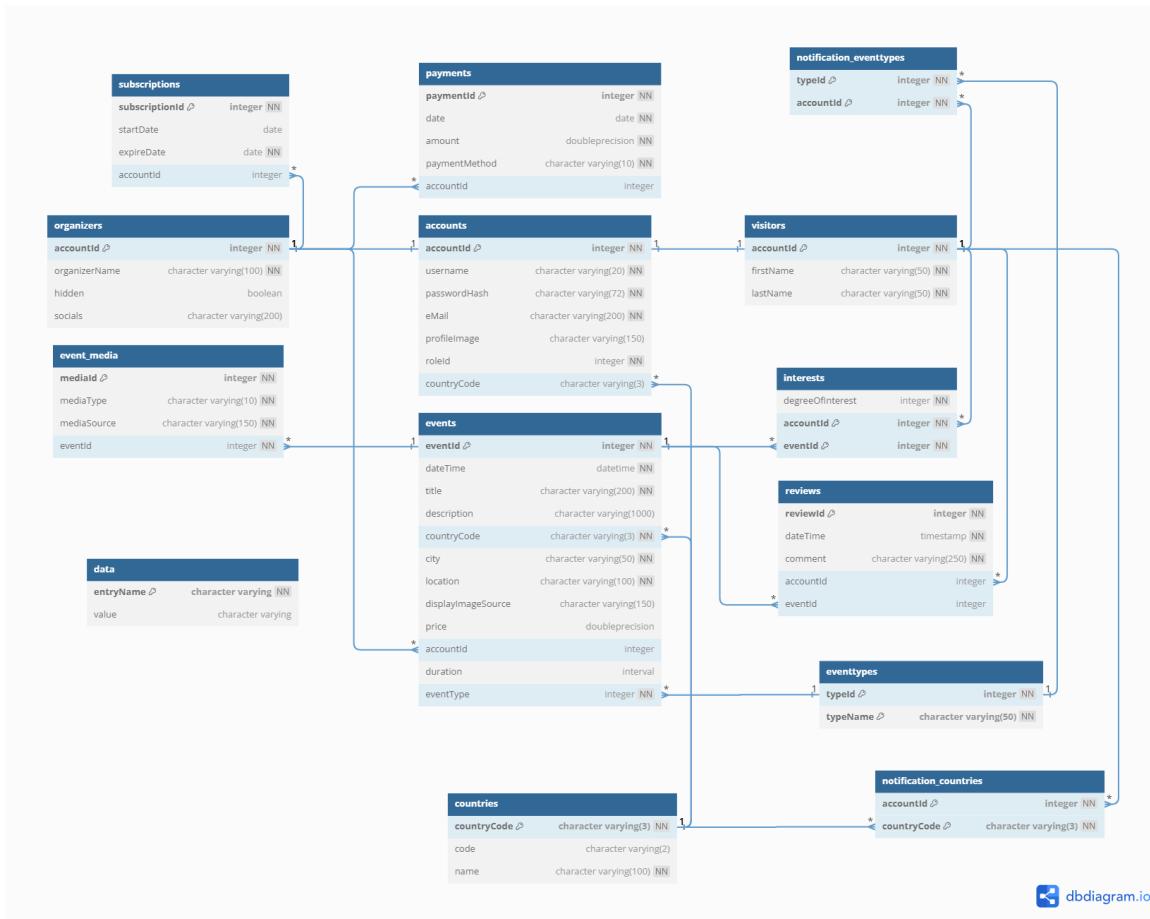
NotificationEventType		
accountId	INT	jedinstveni identifikator posjetitelja (Posjetitelj.accountId)
typeId	INT	identifikator vrste događanja (EventType.typeId). (Primarni ključ je uređeni par accountId, eventType)

Podaci

Ovaj entitet zadužen je spremati različite podatke potrebne za funkcioniranje i prezistenciju poslužiteljske strane. Sadrži atribute: naziv varijable, vrijednost varijable u string formatu. Ovaj entitet nije povezan s niti jednim drugim entitetom. Služi kao "look-up" tablica.

Data		
entryName	VARCHAR	naziv varijable
value	VARCHAR	vrijednost varijable u string formatu

4.1.2 Dijagram baze podataka

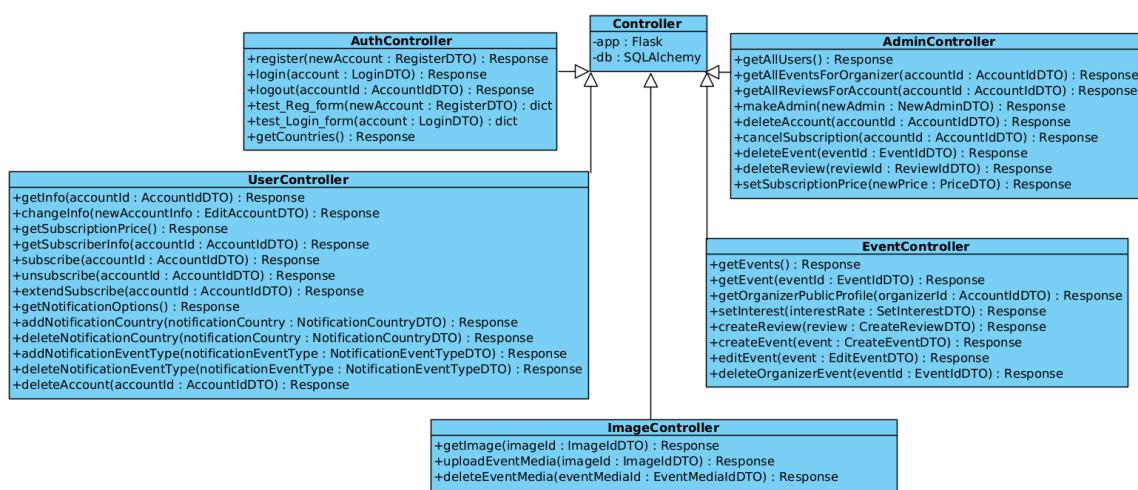


Slika 4.2: Relacijski dijagram baze podataka

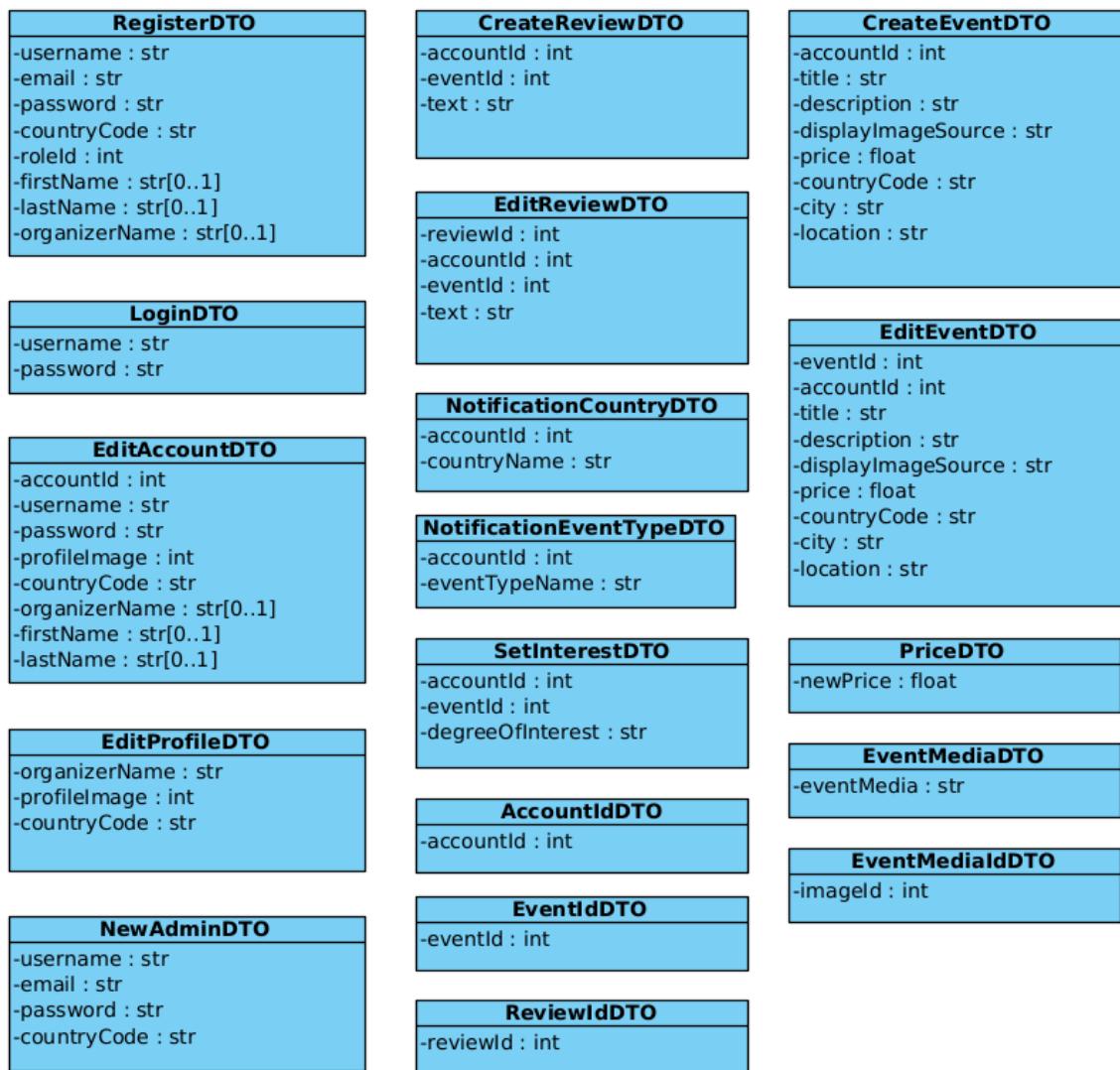
4.2 Dijagram razreda

Na slikama 4.2, 4.3 i 4.4 prikazani su dijagrami razreda za podsustave u backend dijelu arhitekture. Na slici 4.3. prikazani su razredi koji nasljeđuju Controller razred. Metode tih razreda koriste Model razrede za dohvati i spremanje podataka. Metode u razredima Controller vraćaju JSON datoteke u obliku HTML status koda i podatkovnog dijela. Na slici 4.4 prikazani su Data Transfer objekti. Na slici 4.5. prikazani su razredi koji nasljeđuju Model razred i predstavljaju (modeliraju) entitete baze podataka.

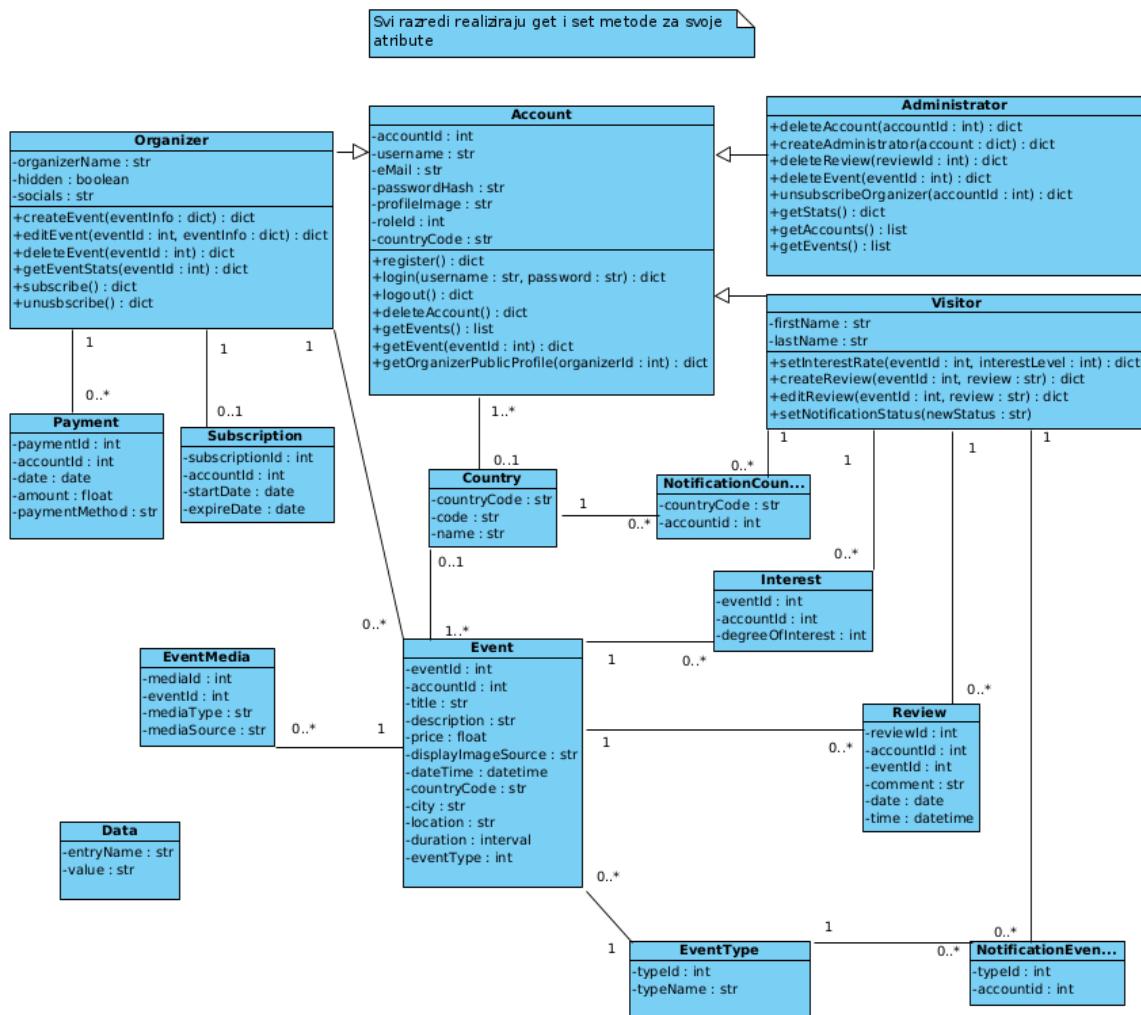
Zbog lakše organizacije i vidljivosti dijagrama razreda, razredi su podijeljeni u više dijagrama. Razredi su grupirani prema sličnim razinama apstrakcije i srodnim funkcionalnostima u arhitekturi MVC.



Slika 4.3: Dijagram klase, dio Controllers



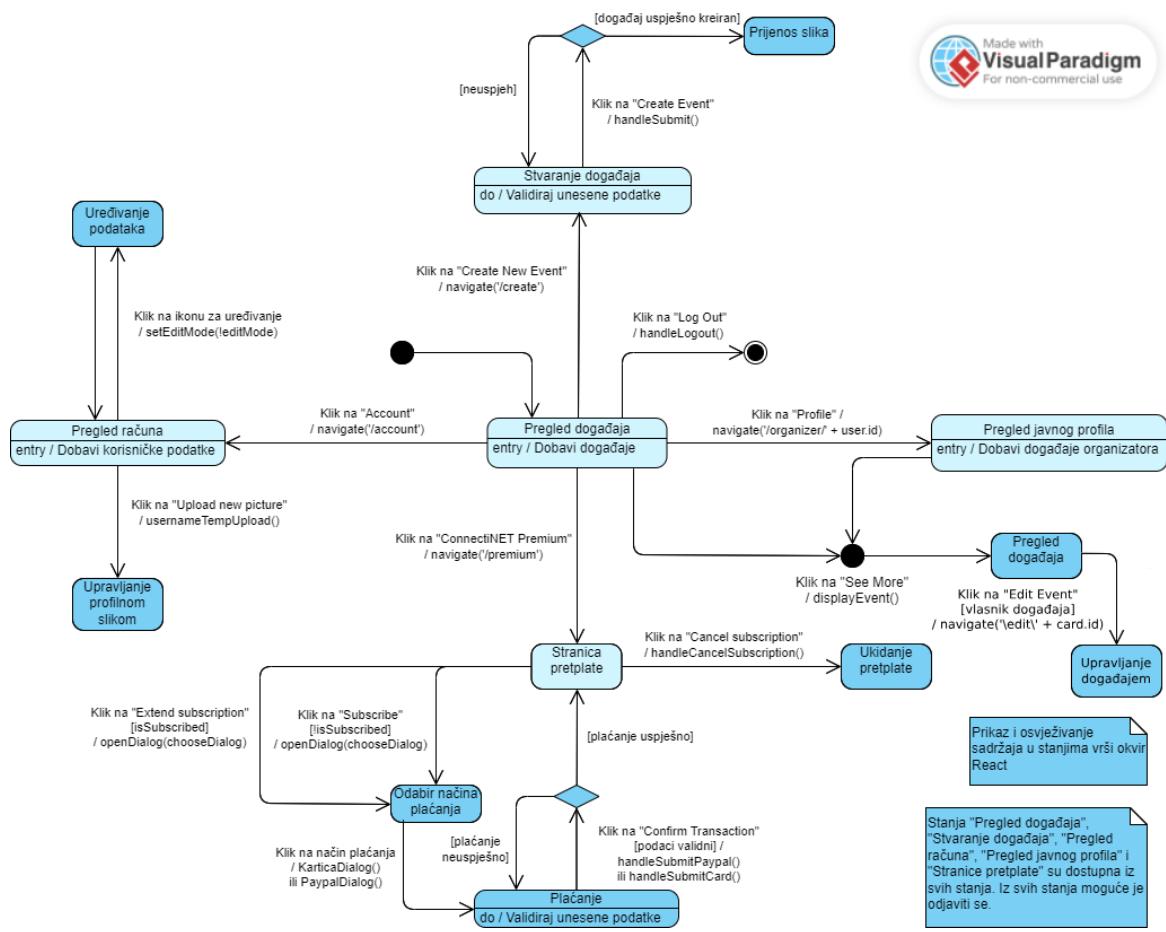
Slika 4.4: Dijagram klasa, dio DTO



Slika 4.5: Dijagram klasa, dio Models

4.3 Dijagram stanja

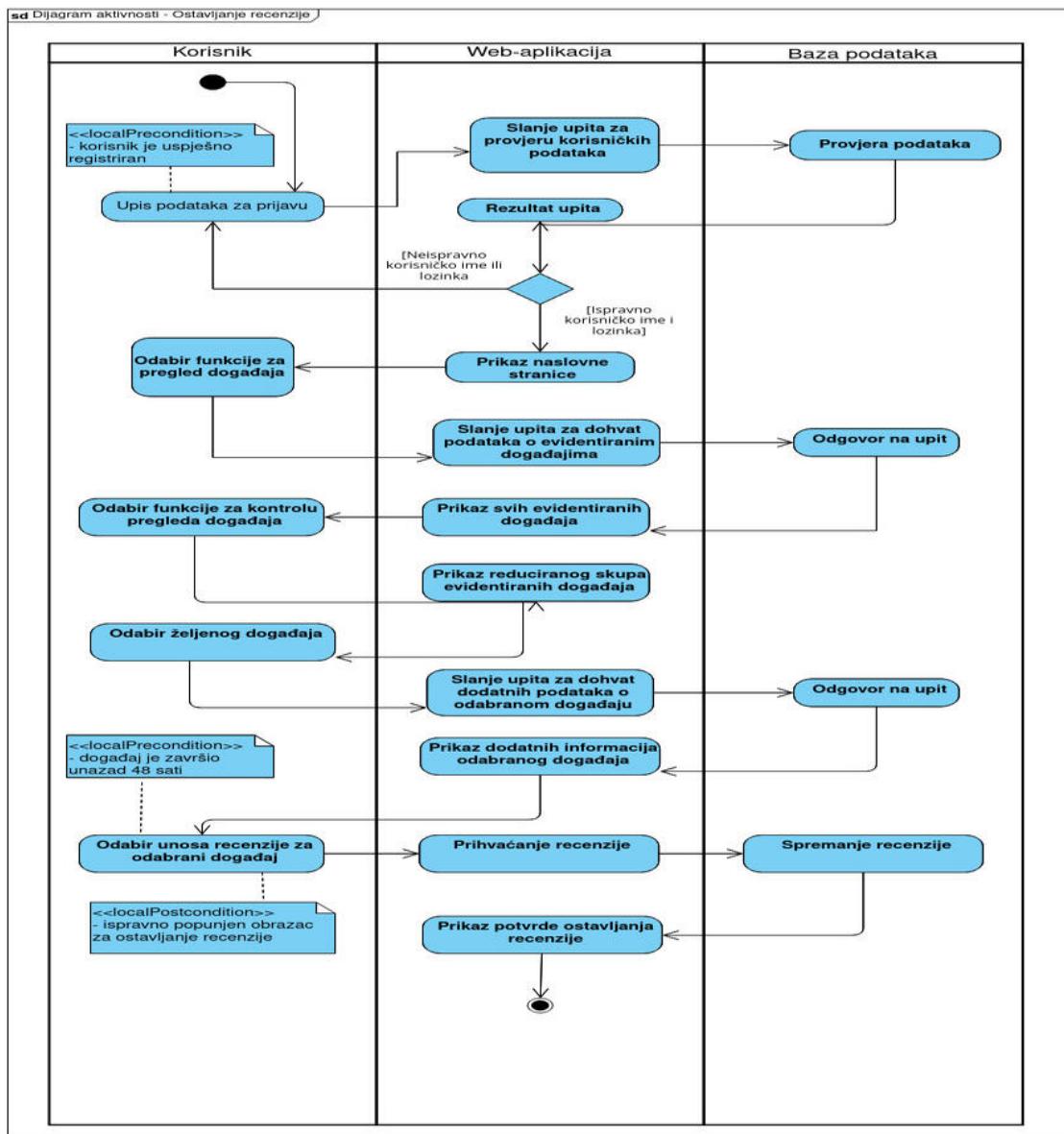
Dijagram stanja opisuje dinamičko ponašanje dijela sustava. Prikazuje stanja objekta te prijelaze temeljene na događajima. Na slici 4.5 prikazan je dijagram stanja za registriranog organizatora. Nakon prijave, organizatoru se prikazuje početna stranica na kojoj se nalaze svi događaji. Klikom na "See More" gumb na pojedinom događaju, organizatoru se prikazuje kartica događaja. Ako je organizator vlasnik događaja, ima mogućnost uređivanja događaja. Na stranici postoji bočna traka koja služi za navigaciju kroz aplikaciju. Klikom na "Profile" organizatoru se prikazuje njegov javni profil. Klikom na "Account" prikazuje se stranica za pregled i uređivanje osobnih podataka te profilne slike. Klikom na "ConnectiNET Premium" prikazuje se stranica za upravljanje pretplatom. Odabirom "Create Event" organizatoru se prikazuje stranica za stvaranje novog događaja.



Slika 4.6: Dijagram stanja

4.4 Dijagram aktivnosti

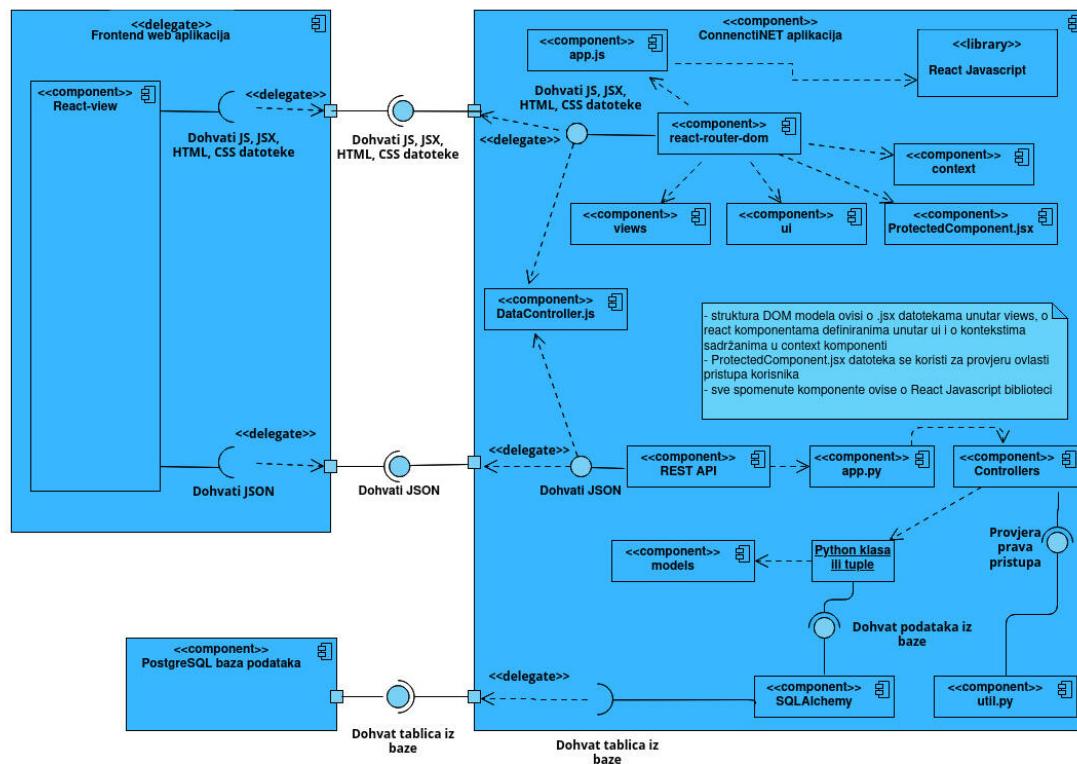
Na dijagramu aktivnosti prikazan je proces ostavljanja recenzije od strane korisnika. Korisnik se najprije prijavi u sustav ako je već jednom uspješno završio proces registracije. Nakon toga odabirom opcije pregleda događaja dolazi do popisa svih trenutno evidentiranih događaja u bazi podataka nad kojim odabirom neke od funkcija za kontrolu pregleda (sortiranje, filtriranje, traženje ključne riječi) reducira izbor prema vlastitim željama. Kada pronađe i odabere željeni događaj otvara mu se prikaz s dodatnim informacijama o dotičnom događaju. Ako je događaj završio unazad 48 sati, Korisnik odabirom funkcije za ostavljenje recenzija popunjava obrazac za ostavljanje recenzije. Ispravnim popunjavanjem obrasca, recenzija se pohranjuje u bazi podataka.



Slika 4.7: Dijagram aktivnosti

4.5 Dijagram komponenti

Sustavu se pristupa preko dva različita sučelja. Preko sučelja za dohvrat HTML, CSS, JS i JSX datoteka poslužuju se datoteke koje pripadaju frontend dijelu aplikacije. Router je komponenta koja na upit s url-a određuje koja datoteka će se poslužiti na sučelje. Frontend dio se sastoji od većeg broja JavaScript datoteka koje su raspoređene u logičke cjeline views, ui i context. Views datoteke definiraju osnovnu strukturu stranice koja će se prikazati korisniku. Ui datoteke predstavljaju komponente koje dopunjaju strukturu stranice. Context sadrži kontekste koji olakšavaju dijeljenje podataka između React komponenata. ProtectedComponent.jsx datoteka je React komponenta koja služi kao zaštitni omotač za druge komponente. Koristi se za provjeru je li korisnik prijavljen i ima li odgovarajuće ovlasti za pristup određenim dijelovima aplikacije. Ako korisnik nije prijavljen ili nema odgovarajuće ovlasti, komponenta preusmjerava korisnika na stranicu za prijavu ili na drugu stranicu. Sve JavaScript datoteke ovise o React biblioteci iz koje dohvaćaju gotove komponente kao što su gumbi, forme i slično. Preko sučelja za dohvrat JSON podataka pristupa se REST API komponenti. REST API poslužuje podatke koji pripadaju backend dijelu aplikacije. SQLAlchemy je zadužen za dohvaćanje tablica iz baze podataka pomoću SQL upita. Podaci koji su pristigli iz baze se šalju dalje MVC arhitekturi u obliku jednostavnih Python klasa ili tuple-ova koji se kasnijom obradom pretvaraju u JSON objekte. Util.py implementira sučelje koje se koristi za provjeru prava pristupa određenim podacima i rutama. React-view komponenta preko oba sučelja komunicira s ConnectiNET aplikacijom te ovisno o korisnikovim akcijama osvježava prikaz i dohvaća nove podatke ili datoteke. Oba sučelja ovise o komponenti DataController.js koja preuzetim zahtjevima s korisničkog sučelja dodaje potrebne parametre i šalje zahtjev na poslužitelj. Kada poslužitelj vrati odgovor, DataController.js obrađuje taj odgovor i prosljeđuje ga natrag korisničkom sučelju.



Slika 4.8: Dijagram komponenti

5. Implementacija i korisničko sučelje

5.1 Korištene tehnologije i alati

Komunikacija u timu realizirana je korištenjem aplikacije Discord¹, a kao sustav za upravljanje izvornim kodom Git². Za izradu UML dijagrama korišten je alat VisualParadigm³. Udaljeni repozitorij projekta je dostupan na web platformi GitHub⁴.

Kao uređivač izvornog koda korišten je Visual Studio Code⁵ napravljen od strane Microsofta s Electron Frameworkom, za Windows, Linux i macOS. Dolazi s ugrađenom podrškom za JavaScript, TypeScript i Node.js te ima bogat ekosustav proširenja za druge jezike i runtimeove. Neke od značajki uključuju podršku za debugiranje, isticanje sintakse, inteligentno dovršavanje koda, isječke, refaktoriranje koda i ugrađeni Git.

Za modeliranje baze podataka korišten je web alat ERDPlus⁶ koji između ostalog omogućuje jednostavno stvaranje ER dijagrama i relacijskih shema. Ovaj besplatan alat pruža pomoć u vizualizaciji i efikasnom dizajniranju baza podataka pomoću automatskog generiranja SQL DDL (Data Definition Language) izjava na temelju korisnički unesenih shema.

Od sustava za upravljanje bazama podataka korišten je PostgreSQL⁷. Sustav poštuje ACID principe pri izvođenju transakcija, proširljiv je i drži se većine SQL:2011 standarda.

Aplikacija je napisana koristeći web okvir Flask⁸. Razvijen je od strane Armina Ronachera, vođe Međunarodne grupe entuzijasta za Python. Temelji se na WSGI alatima i Jinja2 predlošku. Ovaj okvir pokriva širok spektar primjene, od osnovnih koncepta kao što su postavljanje i instalacija do naprednijih koncepta poput autentifikacije korisnika i integracije baze podataka.

¹<https://discord.com/>

²<https://git-scm.com/>

³<https://online.visual-paradigm.com/>

⁴<https://github.com/>

⁵<https://code.visualstudio.com/>

⁶<https://erdplus.com/>

⁷<https://www.postgresql.org/>

⁸<https://flask.palletsprojects.com/en/3.0.x/>

Za izradu frontenda korišten je React⁹ i jezik JavaScript¹⁰. React, također poznat kao React.js ili ReactJS, je biblioteka u JavaScriptu za izgradnju korisničkih sučelja. Održavana je od strane Facebooka. React se najčešće koristi kao osnova u razvoju web ili mobilnih aplikacija. Složene aplikacije u Reactu obično zahtijevaju korištenje dodatnih biblioteka za interakciju s API-jem.

Kao pomoć u razvoju korištena je usluga ElephantSQL¹¹ za hosting baze podataka. ElephantSQL nudi korisnički prijateljsku i skalabilnu platformu za upravljanje PostgreSQL bazama podataka u oblaku. Pojednostavljuje proces opskrbe, upravljanja i skaliranja PostgreSQL instanci, omogućujući razvojnom timu da se usredotoči na funkcionalnost svoje aplikacije umjesto na upravljanje infrastrukturom.

S ciljem pojednostavljenja procesa slanja e-mailova koji je integriran unutar aplikacije, korišten je Mailjet¹² - cloud-based pružatelj usluga e-pošte (ESP) koji omogućuje slanje transakcijskih i marketinških e-mail kampanja. Mailjetova platforma osigurava pouzdano slanje na sigurnoj i robustnoj infrastrukturi koja se nalazi na Google Cloud Platformi, te nadzire i optimizira ključne analitike poput stopa otvaranja, stopa odbijanja i spam pogodaka.

Za pohranu slika korišten je Firebase Storage¹³ - usluga koja dio platforme Firebase unutar Google Cloud Platform-a (GCP) i koja omogućuje pohranu i upravljanje medijima generiranim od strane korisnika web i mobilnih aplikacija. Kada koristimo Firebase Storage, datoteke se prenose izravno od i do klijenta. Kada korisnik prenese datoteku na Firebase Storage, generira se URL za tu datoteku. Taj URL se može zatim spremiti u bazu podataka na poslužitelju. Dakle, na poslužitelju ostaje samo poveznica na datoteku, a sama datoteka se pohranjuje u Firebase Storage. Ovo omogućuje efikasno upravljanje datotekama bez potrebe za velikom infrastrukturom na poslužitelju. Također, smanjuje se opterećenje na poslužitelju jer se prijenos datoteka obavlja izravno između klijenta i Firebase Storage-a.

Svoju primjenu u izradi ovog projekta pronašao je i Docker¹⁴ - otvorena platforma za razvoj, isporuku i pokretanje aplikacija. Docker omogućuje odvajanje aplikacija od infrastrukture kako bi se softver mogao brzo isporučiti. Docker pruža mogućnost pakiranja i pokretanja aplikacije u labavo izoliranom okruženju

⁹<https://reactjs.org/>

¹⁰<https://www.javascript.com/>

¹¹<https://www.elephantsql.com/>

¹²<https://www.mailjet.com/>

¹³<https://firebase.google.com/products/storage/>

¹⁴<https://www.docker.com/>

zvanom kontejner. Kontejneri su lagani i sadrže sve što je potrebno za pokretanje aplikacije. U kontekstu izrade web aplikacija, Docker pojednostavljuje razvoj omogućujući programerima stvaranje prijenosnih i konzistentnih okruženja aplikacija, smanjujući složenost postavljanja i održavanja razvojnih, testnih i producijskih okruženja. Kontejneri su odlični za kontinuiranu integraciju i kontinuiranu isporuku (CI/CD) radnih tokova.

Kao platforma za puštanje u pogon korišten je Render¹⁵ koji omogućuje brzo i jednostavno postavljanje web aplikacija. Podržava sve glavne jezike, besplatne certifikate, zaštitu od DDoS napada i automatsko postavljanje iz GitHuba. Također nudi i sučelje za brzo i jednostavno objavljivanje statičkog web sadržaja. Može se koristiti za postavljanje različitih vrsta aplikacija, uključujući Node.js, Express i Flask aplikacije.

Za izradu dokumentacije korišten je LaTeX¹⁶, markup jezik koji svoju osnovnu primjenu nalazi u izradi znanstvenih publikacija. Osnovna mu je značajka da pisac koristi konvencije označavanja koje predstavljaju ugrađene naredbe za definiranje opće strukture dokumenta, stiliziranje teksta, dodavanje citata, unakrsnih referenci i sl. Tako stvorenu LaTeX datoteku obraduje softver zvan TeX engine koji koristi ugrađene naredbe kako bi vodio i kontrolirao proces izgradnje profesionalno složenog PDF dokumenta.

¹⁵<https://render.com/>

¹⁶<https://www.latex-project.org/>

5.2 Ispitivanje programskog rješenja

5.2.1 Ispitivanje komponenti

Svi testovi komponenti su izvršeni koristeći python biblioteke unittest i requests. Ispitivanje se radilo po obrascima uporabe kako bi se provjerila osnovna funkcionalnost sustava i neki rubni slučajevi, pokut pokušaja dohvata bez tokena za autorizaciju. Prikazano je ispitivanje UC2, UC5, UC6, UC12 i UC13. Slijedi izvorni kod datoteke za testiranje, također vidljiv u /testing direktoriju repozitorija koda. Ispod izvornog koda je i slika izlaza terminala nakon pokretanja testiraanja.

```
import requests
import unittest

class TestApp(unittest.TestCase):
    def setUp(self):
        self.base_url = 'http://127.0.0.1:5000'

    def test_login(self, username, password):
        response = requests.post(
            self.base_url + '/login',
            json={'username': username, 'password': password}
        )
        self.assertEqual(response.status_code, 200, 'Status code is not 200')
        self.assertIn(
            'access_token',
            response.json()['data'],
            'Response does not contain access_token'
        )
        print('Test Login Passed')
        return response.json()

    def test_get_events(self, token):
        response = requests.get(
            self.base_url + '/getEvents',
            headers={'Authorization': 'Bearer ' + token}
```

```
)  
self.assertEqual(  
    response.status_code,  
    200,  
    'Status code is not 200'  
)  
self.assertIsInstance(  
    response.json()['data'],  
    list,  
    'Response data is not a list'  
)  
print('Test Get Events Passed')  
return response.json()  
  
def test_get_events_without_token(self):  
    response = requests.get(self.base_url + '/getEvents')  
    self.assertEqual(  
        response.status_code,  
        401,  
        'Status code is not 401'  
)  
    self.assertIn(  
        'msg',  
        response.json(),  
        'Response does not contain msg'  
)  
print('Test Get Events Without jwt failed, as expected')  
return response.json()  
  
def test_create_event(self, token):  
    response = requests.post(  
        self.base_url + '/api/createEvent',  
        headers={'Authorization': 'Bearer ' + token},  
        json={  
            'title': 'EventName',
```

```
'description':'Description',
'city':'City',
'location':'Place',
'countryCode':'CCK',
'eventType':'Concert',
'dateTime':'2024-01-26T12:00',
'duration':'2024-01-29T12:00',
'price':'1'
})
self.assertEqual(
response.status_code, 200,
'Status code is not 200'
)
self.assertIn(
'eventId',
response.json()['data'],
'Response does not contain eventId'
)
print('Test Edit Event Passed')
return response.json()

def test_edit_event(self, token, id):
response = requests.put(
self.base_url + '/api/editEvent/62',
headers={'Authorization': 'Bearer ' + token},
json={
'title':'NewEventName',
'description':'New Description',
'city':'New City',
'location':'New Place',
'countryCode':'HRV',
'eventType':'Community',
'dateTime':'2025-01-26T12:00',
'duration':'2025-02-29T12:00',
'price':'20'
```

```
    }
    self.assertEqual(
        response.status_code,
        200,
        'Status code is not 200'
    )
    self.assertIn(
        'success',
        response.json(),
        'Response does not contain success'
    )
    print('Test Edit Event Passed')
    return response.json()

def test_delete_account(self, token):
    response = requests.delete(
        self.base_url + '/api/deleteAccount',
        headers={'Authorization': 'Bearer ' + token}
    )
    self.assertEqual(
        response.status_code,
        200,
        'Status code is not 200'
    )
    self.assertIn(
        'success',
        response.json(),
        'Response does not contain success'
    )
    print('Test Delete Account Passed')
    return response.json()

# Instantiate the test class
```

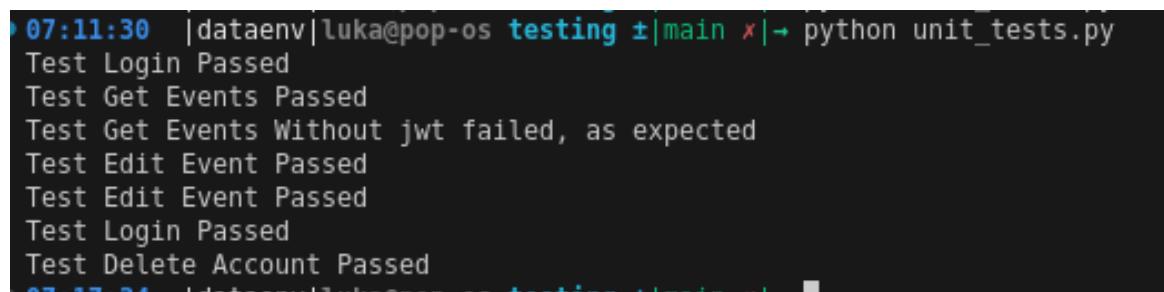
```
test_app = TestApp()
test_app.setUp()

# Login
jwt = test_app.test_login(
    'kraljevi',
    'pass123kraljevi')['data']['access_token']

# Attempt to get events
test_app.test_get_events(jwt)
test_app.test_get_events_without_token()

# Create and edit an event
new_event_id = test_app.test_create_event(
    jwt)['data']['eventId']
test_app.test_edit_event(jwt, new_event_id)

# Delete a user
jwt = test_app.test_login('testuser',
    'password123')['data']['access_token']
test_app.test_delete_account(jwt)
```



```
07:11:30 |dataenv|luka@pop-os testing ±|main ✘|+ python unit_tests.py
Test Login Passed
Test Get Events Passed
Test Get Events Without jwt failed, as expected
Test Edit Event Passed
Test Edit Event Passed
Test Login Passed
Test Delete Account Passed
```

Slika 5.1: Izlaz terminala nakon pokretanja testiranja

5.2.2 Ispitivanje sustava

Svi testovi izvršeni su koristeći Selenium WebDriver. Ispitivanje se radilo po obrascima uporabe kako bi se provjerila osnovna funkcionalnost sustava. Prikazano je

ispitivanje UC1 (i UC2), UC4, UC5, UC9 i UC12. Izvorni kod vidljiv je u direktoriju testing.

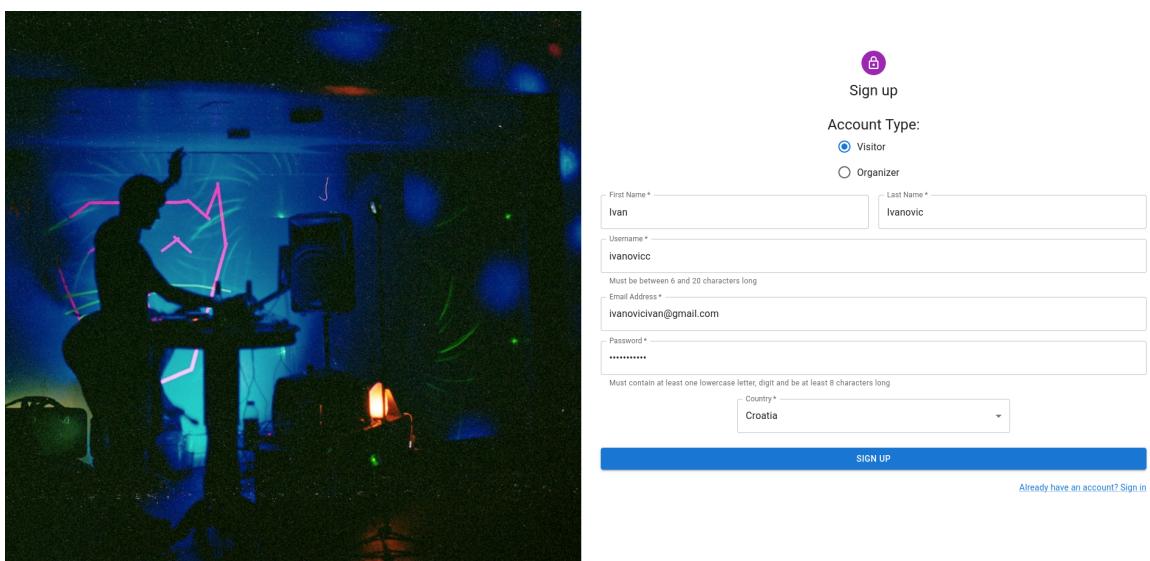
Ispitni slučaj 1: Stvaranje korisničkog računa

- **Ulaz:**

1. Otvaranje stranice za registraciju u web pregledniku
2. Unos uloge, imena, prezimena, korisničkog imena, email adrese, lozinke i države
3. Odabir opcija "Sign Up" i "Sign In"

- **Očekivani rezultat:**

- 1 Uneseni podaci su vidljivi na stranici za prijavu
 - 2.a Korisnik je uspješno registriran
 - 2.b Neki od unesenih podataka je u neispravnom formatu, obavijest upozrava korisnika
- **Rezultat:** Svi očekivani rezultati su ostvareni i provjereni uspješnom prijавom na kreirani korisnički račun. Aplikacija je prošla test.



Slika 5.2: Testiranje stvaranja korisničkog računa

Ispitni slučaj 2: Uređivanje korisničkog računa

- **Ulaz:**

1. Otvaranje Account stranice u web pregledniku
2. Odabir opcije "Edit Account" i "Save Changes"
3. Unos novih podataka u polja za uređivanje

- **Očekivani rezultat:**
 - 1 Uneseni podaci su vidljivi na stranici za uređivanje
 - 2.a Korisnik je uspješno uređen
 - 2.b Neki od unesenih podataka je u neispravnom formatu, obavijest upozrava korisnika
- **Rezultat:** Svi očekivani rezultati su ostvareni. Aplikacija je prošla test.

The screenshot shows the 'Account' section of the ConnectiNET application. On the left, there's a sidebar with 'Account information' containing fields for Username (ivanovic), Role (Visitor), Email (ivanovicivan@gmail.com), Country (Croatia), First Name (Pero), Last Name (Perić), and Password (hidden). A red 'DELETE ACCOUNT' button is at the top right of this section. Below it is a green 'SAVE CHANGES' button. In the center, there's a section for 'Your notification preferences' with tabs for 'Country preferences' and 'Event type preferences'. Under 'Country preferences', there are dropdowns for 'Choose new country *' and 'Choose new event type *', along with 'ADD COUNTRY' and 'ADD EVENT' buttons. On the right, there's a placeholder for 'Current profile image:' showing a blue placeholder icon. Below it is a section for 'Upload new profile image:' with a 'CHOOSE FILE' button and a note 'Image not selected.' At the bottom of the page, the 'ConnectiNET by Kraljevi' footer is visible.

Slika 5.3: Testiranje uređivanja korisničkog računa

Ispitni slučaj 3: Brisanje korisničkog računa

- **Ulaz:**
 1. Otvaranje Account stranice u web pregledniku
 2. Odabir opcije "Delete Account" i potvrde brisanja
- **Očekivani rezultat:**
 - 1.a Korisnički račun je obrisan nakon pritiska "Yes" za potvrdu brisanja
 - 1.b Korisnički račun nije obrisan nakon pritiska "No" za potvrdu brisanja
- **Rezultat:** Svi očekivani rezultati su ostvareni. Aplikacija je prošla test.

Ispitni slučaj 4: Stvaranje novog događaja

- **Ulaz:**
 1. Otvaranje stranice za registraciju u web pregledniku
 2. Unos i odabir potrebnih podataka za kreiranje novog događaja
 3. Odabir opcije za "Create event"

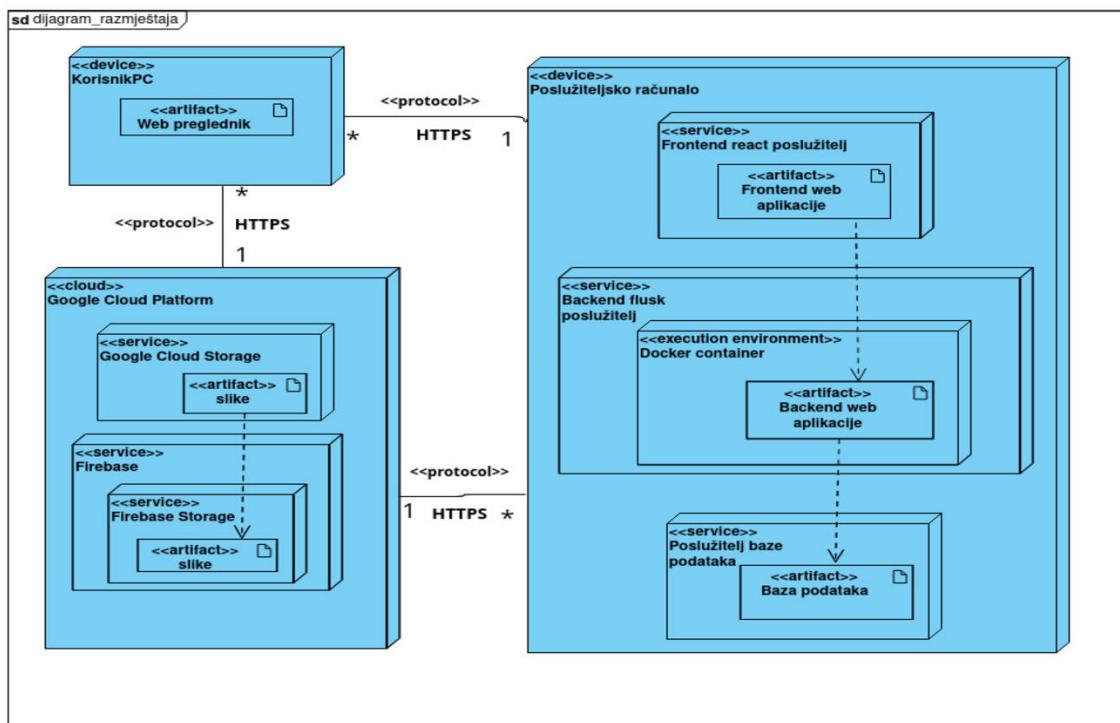
- **Očekivani rezultat:**
 - 1 Uneseni podaci su vidljivi na stranici za kreiranje novog eventa
 - 2.a Događaj je uspješno kreiran
 - 2.b Neki od unesenih podataka je u neispravnom formatu, obavijest upozrava korisnika
- **Rezultat:** Svi očekivani rezultati su ostvareni. Aplikacija je prošla test.

Ispitni slučaj 5: Ostavljanje razine zainteresiranosti

- **Ulaz:**
 1. Otvaranje stranice za registraciju u web pregledniku
 2. Ulazak u detaljni prikaz o događaju klikom na "See more"
 3. Odabir razine zainteresiranosti
- **Očekivani rezultat:**
 - 1 Uneseni podaci su vidljivi na stranici za prijavu
 - 2 Korisnik je oodabrao razinu zainteresiranosti
- **Rezultat:** Svi očekivani rezultati su ostvareni. Aplikacija je prošla test.

5.3 Dijagram razmještaja

Sustav je baziran na arhitekturi "klijent – poslužitelj", a klijenti koriste web preglednik kako bi pristupili web aplikaciji. Na poslužiteljskom računalu nalaze se poslužitelj frontend dijela web aplikacije, poslužitelj backend dijela web aplikacije te poslužitelj baze podataka. Backend dio aplikacije nalazi se unutar Docker kontejnera koji predstavlja okruženje koje sadrži sve što je potrebno za izvođenje dotičnog dijela aplikacije. Pohrana slika realizirana je uz pomoć usluge Firebase Storage koja je dio Firebase-a, platforme za razvoj mobilnih i web aplikacija. Firebase Storage se za pohranu i dohvrat korisnički generiranih sadržaja oslanja na uslugu Google Cloud Storage. Sve spomenute usluge dio su šire platforme pod nazivom Google Cloud Platform (GCP). Odgovornost backend dijela web aplikacije je da prihvati generiranu javnu poveznicu na tu sliku i pohrani je u bazu podataka nakon što se korisnička slika otpremi na Firebase Storage. Firebase Storage omogućuje izravan prijenos datoteke od i do klijenta čime se rasterećuje poslužitelj koji samo procesira poveznice. Komunikacija između osnovnih dijelova sustava odvija se preko HTTPS veze.



Slika 5.4: Dijagram razmještaja

5.4 Upute za puštanje u pogon

5.4.1 Priprema Lokalnog Razvojnog Okruženja

Za početak, potrebno je klonirati repozitorij s izvornim kodom na svoje računalo koristeći naredbu: `git clone https://github.com/jetpans/ConnectiNET-kraljevi`. Nakon toga je potrebno instalirati alate potrebne za pokretanje aplikacije lokalno:

- Instalirajte Docker i Docker Compose ili Docker Engine (<https://docs.docker.com/engine/install/>)
- Instalirajte Node.js i npm za frontend (<https://nodejs.org/en/download/>)
- Instalirajte Python za backend (u slučaju da želite pokrenuti backend lokalno bez korištenja Docker kontejnera) (<https://www.python.org/downloads/>)
- Instalirajte PostgreSQL bazu podataka (<https://www.postgresql.org/download/>)

Za daljnji razvoj aplikacije preporuča se koristiti radno okruženje poput Visual Studio Code-a (<https://code.visualstudio.com/>).

5.4.2 Postavljanje Backend-a s Docker Compose

U backend direktoriju se nalaze Dockerfile i docker-compose datoteke. Izvršite u naredbenom retku naredbu `docker-compose up --build` za izgradnju Docker kontejnera i pokretanje backend poslužitelja.

5.4.3 Konfiguracija PostgreSQL Baze Podataka

Stvaranje PostgreSQL baze podataka

- Prije nego što počnete, osigurajte da imate PostgreSQL instaliran na lokalnom sustavu.
- Koristite alat poput pgAdmin ili izravno putem terminala, stvorite novu bazu podataka za vašu aplikaciju.
- Definirajte korisnika i lozinku s odgovarajućim ovlastima na novoj bazi.

Punjjenje PostgreSQL Baze Podataka

Kako biste sagradili inicijalno stanje baze podataka, u razvojnoj fazi prvo smo napisali "SQLAlchemy" razrede koji modeliraju naše tablice u bazi podataka i spremili ih u datoteku models.py. Nakon toga, sagradili smo naše prazne entitete u bazi podataka sljedećim naredbama u python okruženju:

```
from models import app, db  
app.app_context().push()  
db.create_all()
```

Osnovne podatke za vrstu događanja ručno smo unijeli u bazu koristeći "INSERT INTO" naredbe. Podatke za države kopirali smo iz javno dostupnog GitHub repozitorija te ih također unijeli koristeći "INSERT INTO" naredbe. Nakon unosa svih potrebnih inicijalnih vrijednosti u bazu, ovo stanje baze podataka spremili smo u datoteku backup.sql koristeći naredbu naredbenog retka:

```
pg_dump -c "<URI baze podataka>" > backup.sql
```

Datoteka backup.sql sada ima sve potrebne PostgreSQL naredbe za čišćenje (-c) i unos starih podataka te se može koristiti za iniciranje baze podataka za vrijeme puštanja aplikacije u pogon.

Bazu sada možemo puniti iz prije napravljene sigurnosne kopije sa svim potrebnim, unaprijed unesenim, podacima. To činimo naredbom naredbenog retka:

```
psql -d "<URI baze podataka>" -f backup.sql
```

koja briše sve postojeće tablice i restaurira stanje baze podataka spremljeno u datoteci backup.sql.

5.4.4 Izgradnja i Pokretanje Frontend-a

Frontend direktorij sadrži izvorni kod React aplikacije. U njemu izvršite naredbu za instalaciju ovisnosti

```
npm install
```

Za pokretanje razvojnog servera namijenjenog za lokalni razvoj aplikacije koristi se naredbama:

```
npm run start
```

Za izgradnju optimiziranog produkcijskog koda izvršite naredbu:

```
npm run build
```

Na kraju, za pokretanje Express servera koji poslužuje frontend koristi se naredba:

```
node app.js
```

Nakon ovog koraka u potpunosti smo podigli lokalno razvojno okruženje i aplikacija je dostupna na localhost:3000. Za uspješan rad potrebno je i konfigurirati varijable okruženja, što je opisano u sljedećem koraku za puštanje u pogon na servis Render.

5.4.5 Puštanje u pogon na Render Platformu

Kreirajte račun na platformi Render (<https://render.com/>). Nakon što se ulogirate, redom dodajte:

- Novu PostgreSQL bazu podataka
- Novi web servis za backend dio aplikacije
 - Preporučuje se koristiti github repozitorij s izvornim kodom, s tim da za root direktorij postavite vrijednost /backend.
 - Za runtime odaberite Docker, a naredba za pokretanje je gunicorn -w 4 -b 0.0.0.0:5000 run:app.
 - U varijable okruženja dodajte sve potrebne varijable:

```
FLASK_ENV (production za produkciju, development za razvoj)  
SECRET_KEY (vaš tajni ključ)  
DB_CONNECT_URL_PROD (Render url za povezivanje  
s bazom podataka iz prethodnog koraka)  
MAIL_API_KEY (API ključ dobiven od Mailjet platforme)  
MAIL_SECRET (tajni ključ dobiven od Mailjet platforme)  
FIREBASE_URL (url dobiven od Firebase platforme)  
JWT_SECRET_KEY (tajni ključ za JWT token)
```

- Novi web servis za frontend dio aplikacije

- Preporučuje se koristiti github repozitorij s izvornim kodom, s tim da za root direktorij postavite vrijednost /frontend.
- Za runtime odaberite Node.js, za build naredbu odaberite `npm install` `npm run build`, a za start naredbu odaberite `node app.js`.
- U varijable okruženja dodajte sve potrebne varijable:

`REACT_APP_API_URL` (Render url za povezivanje
s backend poslužiteljem iz prethodnog koraka)

`HOST` (0.0.0.0)

`PORT` (3000)

Render će automatski otkriti promjene u repozitoriju i pustiti u pogon aplikaciju u slučaju da ste odabrali opcije za automatsko puštanje u pogon pri kreaciji servisa. U suprotnom, možete ručno pokrenuti puštanje u pogon. Aplikacija je sada dostupna na url-u dobivenom od Render platforme. Odlaskom na dobiveni url vidjet ćemo ekran za prijavu aplikacije ConnectiNET. Moguće je i direktno pristupiti backend poslužitelju na url-u dobivenom od Render platforme.

6. Zaključak i budući rad

Projektni zadatak našeg tima bio je razvoj web aplikacije za promociju događanja. Korisnici ovisno o ulozi posjetitelja, organizatora ili administratora mogu izvoditi različite radnje nad vlastitim i tuđim objavama. Provedba projekta odvijala se u dvije faze.

Prva faza projekta uključivala je okupljanje tima za razvoj aplikacije, upoznavanje s projektnim zadatka i intenzivan rad na dokumentiranju zahtjeva koje je naš krajnji projekt morao ispuniti. Kvalitetna provedba prve faze uvelike je olakšala daljnji rad pri realizaciji osmišljenog sustava jer je cijelokupnu problematiku uspješno podijelila na više manjih problema. Izrađeni obrasci i dijagrami (obrasci uporabe, sekvensijski dijagrami, model baze podataka, dijagram razreda) bili su od pomoći u kasnijoj fazi implementacije.

Druga faza projekta, iako nešto kraća od prve, bila je puno intenzivnija po pitanju samostalnog rada članova. Raznolikost među iskustvima članova tima u izradi sličnih implementacijskih rješenja primorao je iskusnije članove da svoja znanja adekvatno prenesu na kolege koje nisu imali doticaja s takvim izazovima. Poseban naglaskak bio je na samostalnom učenju odabranih alata i programskih jezika kako bi svi uspješno ispunili zadatke postavljene pred njih. Osim realizacije rješenja, u drugoj fazi je bilo potrebno dokumentirati ostale UML dijagrame i izraditi popratnu dokumentaciju kako bi budući korisnici mogli lakše koristiti ili vršiti preinake na sustavu. Dobro izrađen kostur projekta uštedio nam je mnogo vremena prilikom izrade aplikacije te smo izbjegli moguće pogreške u izradi koje bi bile vremenski skupe za ispravljanje u daljnjoj fazi projekta. Podijela zadataka bazirala se po funkcionalnostima koje je potrebno implementirati tako da je svaki član imao doticaja s cijelokupnom infrastrukturom.

Komunikacija među članovima odvijala se putem Discorda čime smo postigli efikasnu razmjenu znanja i savjeta za realizaciju zadataka postavljenih pred svakog člana. Moguće proširenje postojeće inačice sustava je izrada mobilne aplikacije čime bi cilj projektnog zadatka bio ostvaren u većoj mjeri nego s web aplikacijom. Od tehničkih izazova s kojima smo se susreli posebno bi izdvojili efikasnu pohranu i dohvatanje korisnički generiranog sadržaja poput slika. Adekvatnom komunika-

cijom i istraživačkim radom, rješenje smo pronašli u usluzi Firebase koja je podigla stupanj iskoristivosti naše aplikacije na višu razinu.

Sudjelovanje na ovakvom projektu bilo je vrijedno iskustvo svim članovima tima jer smo kroz cjelokupni period rada na projektu iskusili važnost zajedničke suradnje te stekli nove socijalne i profesionalne veze koje će nam biti od velike koristi u karijernom razvoju. Također, osjetili smo važnost dobre vremenske organiziranosti i koordiniranosti između članova tima te smo izuzetno zadovoljni postignutim rezultatima.

Popis literature

1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinzh>
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. Flask Dokumentacija, Pallets Projects, <https://flask.palletsprojects.com/>
8. Docker Dokumentacija, Docker, Inc., <https://docs.docker.com/>
9. React Dokumentacija, React team at Facebook, Inc., <https://reactjs.org/docs/getting-started.html>
10. Material UI Dokumentacija, Material-UI team, <https://mui.com/getting-started/installation/>
11. Selenium Dokumentacija , SeleniumHQ project, <https://www.selenium.dev/documentation/en/>

Indeks slika i dijagrama

2.1	HappeningNext	7
2.2	ConnectiNET	7
3.1	Dijagram obrasca uporabe, funkcionalnost posjetitelja	23
3.2	Dijagram obrasca uporabe, funkcionalnost organizatora	24
3.3	Dijagram obrasca uporabe, funkcionalnost administratora	25
3.4	Sekvencijski dijagram za UC1	26
3.5	Sekvencijski dijagram za UC5	27
3.6	Sekvencijski dijagram za UC10	28
3.7	Sekvencijski dijagram za UC12	29
4.1	Skica osnovne arhitekture sustava	32
4.2	Relacijski dijagram baze podataka	42
4.3	Dijagram klase, dio Controllers	43
4.4	Dijagram klase, dio DTO	44
4.5	Dijagram klase, dio Models	45
4.6	Dijagram stanja	46
4.7	Dijagram aktivnosti	48
4.8	Dijagram komponenti	50
5.1	Izlaz terminala nakon pokretanja testiranja	58
5.2	Testiranje stvaranja korisničkog računa	59
5.3	Testiranje uređivanja korisničkog računa	60
5.4	Dijagram razmještaja	62
6.1	Dijagram pregleda promjena	78

Dodatak: Prikaz aktivnosti grupe

Dnevnik sastajanja

1. sastanak

- Datum: 19. listopada 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - sastanak s asistenticom i demonstratorom
 - uvod u temu projekta

2. sastanak

- Datum: 22. listopada 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - analiza teme projekta
 - korištenje git-a i github-a
 - konačan odabir alata i tehnologija
 - uvod u naš github repozitorij

3. sastanak

- Datum: 30. listopada 2023.
- Prisustvovali: L. Miličević, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - definiranje funkcionalnih zahtjeva
 - podjela opisa obrazaca uporabe, opisa zadatka i arhitekture sustava

4. sastanak

- Datum: 2. studenoga 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:

- analiza do sada napravljenih funkcionalih zahtjeva
- diskusija o modeliranju baze podataka

5. sastanak

- Datum: 9. studenoga 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - sastanak i diskusija s asistentom

6. sastanak

- Datum: 12. studenoga 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - zajednička implementacija frontend dijela generičkih funkcionalnosti

7. sastanak

- Datum: 18. prosinca 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - plan nastavka rada na projektu
 - podjela poslova

8. sastanak

- Datum: 3. siječnja 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - detaljnija podjela dužnosti

9. sastanak

- Datum: 9. siječnja 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - praćenje napretka implementacije funkcionalnosti
 - diskusija o problemima i rješenjima

- plan nastavka rada

10. sastanak

- Datum: 16. siječnja 2023.
- Prisustvovali: L. Miličević, M. Papak, F. Aleksić, D. Huljev, S. Đelekovčan, D. Sviličić, E. Pužar
- Teme sastanka:
 - plan o završetku zadatka
 - riješavanje problema s implementacijom

Tablica aktivnosti

	Luka Miličević	Duje Huljev	Filip Aleksić	Mate Papak	Stjepan Deleković	Domagoj Svilicić	Erik Pužar
Upravljanje projektom	3						
Opis projektnog zadatka	3						
Funkcionalni zahtjevi		4	3	2		3	3
Opis pojedinih obrazaca		3	3	3		2	3
Dijagram obrazaca		3	2	2		1	2
Sekvencijski dijagrami		1				4	4
Opis ostalih zahtjeva	0.5						
Arhitektura i dizajn sustava	2						
Baza podataka					9		
Dijagram razreda	7						
Dijagram stanja		3					
Dijagram aktivnosti						6	
Dijagram komponenti						9	
Korištene tehnologije i alati						4	
Ispitivanje programskog rješenja	3		14	9			
Dijagram razmještaja						6	
Upute za puštanje u pogon							
Dnevnik sastajanja	1						
Zaključak i budući rad						3	

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

	Luka Miličević	Duje Huljev	Filip Aleksić	Mate Papak	Stjepan Deleković	Domagoj Svilicić	Erik Pužar
Popis literature	1						
Incijalno postavljanje frontenda aplikacije	7						
Inicijalno postavljanje backenda aplikacije	1				6		
Generičke funkcionalnosti - Login i Registracija	5	1	1	1	11	1	2
Puštanje aplikacije u pogon	5						
Infrastruktura i uređenje frontenda aplikacije	15	2					1
Infrastruktura i uređenje backenda aplikacije	2				6		1
Izrada infrastrukture za rad sa slikama		1			4		
Stranica za korisnički račun	1	1			15		2
Stranica za javni profil		1		11			
Stranice za pretplatu i administraciju	4				3		8
Stranice za izradu i uređenje događaja	4	12		12	6		2
Kartica za detalje o događaju			8				

Nastavljeno na idućoj stranici

Nastavljeno od prethodne stranice

Luka Miličević	Duje Huljev	Filip Aleksić	Mate Papak	Stjepan Delekovićan	Domagoj Svilicić	Erik Pužar
Kartica za pregled događaja - sortiranje i filtriranje	2	9				

Dijagrami pregleda promjena

Komentar na pregled izmjena: prikaz nije u potpunosti točan i prikazuje manje promjena za dio tima koji nije sam merge-ao svoje grane u develop ili devdoc grane.



Slika 6.1: Dijagram pregleda promjena