



## Sadržaj

Uvod .....	1
1. Korištene tehnologije.....	2
1.1. Korisnička strana .....	2
1.1.1. React .....	2
1.1.2. TypeScript .....	3
1.1.3. Vite .....	3
1.1.4. Google Maps API .....	3
1.2. Poslužiteljska strana .....	4
1.2.1. Spring Boot.....	4
1.2.2. SocketIO .....	4
1.3. Baza podataka.....	5
1.3.1. PostgreSQL.....	5
1.3.2. Liquibase .....	5
2. Zahtjevi sustava .....	6
2.1. Generički zahtjevi aplikacije .....	6
2.2. Specifični zahtjevi aplikacije.....	6
3. Arhitektura sustava.....	7
3.1. Model arhitekture sustava.....	7
3.2. Model baze podataka .....	8
3.2.1. Opisi tablica.....	8
4. Implementacija sustava.....	11
4.1. Struktura aplikacije.....	11
4.2. Implementacija korisničke strane .....	11
4.3. Bitniji korišteni paketi korisničke strane .....	12
4.3.1. country-flag-icons.....	12

4.3.2.	country-select-js.....	12
4.3.3.	react-toastify .....	12
4.3.4.	@vis.gl/react-google-maps.....	13
4.4.	Implementacija poslužiteljske strane.....	14
4.5.	Integracija s bazom podataka .....	15
4.6.	Implementacija sigurnosnih značajki .....	15
5.	Korištenje sustava.....	17
5.1.	Autentifikacija .....	17
5.2.	Početna stranica .....	18
5.3.	Korisnikovi planovi putovanja .....	19
5.4.	Korisnikov profil .....	20
5.5.	Uređivač plana putovanja .....	21
5.5.1.	Osnovni detalji plana putovanja .....	21
5.5.2.	Sekcije plana, mjesta u putovanju .....	22
5.5.3.	Interaktivna karta, prijedlozi atrakcija.....	24
5.5.4.	Ostali sudionici, objava plana, prijedlozi hotela.....	25
5.5.5.	Pregled objavljenog plana putovanja.....	25
5.6.	Administratorska upravljačka ploča .....	26
	Zaključak .....	28
	Literatura .....	29
	Popis slika.....	30
	Sažetak.....	31
	Summary.....	32

# Uvod

U današnjem digitalnom dobu, planiranje putovanja postalo je značajno olakšano razvojem raznih tehnoloških rješenja. Unatoč dostupnosti brojnih aplikacija koje pružaju različite aspekte putovanja, postoji potreba za sveobuhvatnom platformom koja integrira sve potrebne funkcionalnosti u jednom korisničkom sučelju. Ovaj završni rad ima za cilj razvoj takve aplikacije, koja će korisnicima omogućiti stvaranje novih planova putovanja, spremanje postojećih planova pod svoj profil te njihovo dijeljenje sa zajednicom.

Korisnici će imati priliku koristiti interaktivnu kartu za odabir ključnih mjesta koja žele posjetiti tijekom svog putovanja. Aplikacija će nuditi prijedloge atrakcija koje vrijedi posjetiti u određenom mjestu, te pružati informacije o obližnjem smještaju te vremenskim uvjetima.

Jedna od ključnih značajki ove aplikacije je mogućnost suradnje među korisnicima. Više korisnika moći će istovremeno raditi na jednom planu putovanja, čime se omogućava zajedničko planiranje i koordinacija među prijateljima, obitelji ili poslovnim partnerima. Ova funkcionalnost će se osigurati korištenjem real-time tehnologija koje omogućavaju trenutne promjene i ažuriranja.

Aplikacija će također korisnicima pružati mogućnost definiranja budžeta za sekcije putovanja. Ovo će korisnicima pomoći u boljoj financijskoj pripremi i upravljanju budžetom za putovanje. Dodatno, korisnici će moći dodavati bilješke u svoj itinerar, omogućavajući im da zabilježe važne informacije ili osobne napomene vezane za putovanje.

Sve navedene funkcionalnosti bit će dostupne putem intuitivnog web sučelja koje će omogućiti jednostavno korištenje aplikacije. Razvoj ove aplikacije obuhvatit će korištenje modernih web tehnologija, osiguravajući visoku razinu performansi, sigurnosti i korisničkog iskustva.

# 1. Korištene tehnologije

Pri izradi rada koristio sam nekoliko suvremenih i široko podržanih tehnologija koje omogućavaju visoku razinu funkcionalnosti i olakšavaju razvoj web-aplikacija. Ove tehnologije pružaju stabilnu osnovu za izradu skalabilnih i pouzdanih sustava te su ključne za postizanje kvalitetnog korisničkog iskustva. Korištene tehnologije općenito se mogu podijeliti u tri skupine: korisnička strana, poslužiteljska strana te baza podataka.

## 1.1. Korisnička strana

### 1.1.1. React

Za korisničku stranu aplikacije kao glavnu tehnologiju odabrao sam React, jednu od najpopularnijih JavaScript knjižnica za izradu korisničkih sučelja. Razvijen od strane Facebooka, React omogućava izradu dinamičkih i responzivnih web stranica koristeći komponente koje se mogu ponovno koristiti [1].

Komponente su neovisne jedna o drugoj i komuniciraju prosljeđivanjem podataka kroz hijerarhiju. Glavni način međudjelovanja komponenti je korištenjem takozvanih React „hooks“ (udica). Najkorisnije udice su upravo useState i useEffect udice.

useState se tipično koristi za definiranje varijable na razini komponente te funkcije koja postavlja vrijednost te varijable. Vrijednost varijable može se proslijediti, ali može se proslijediti i funkcija za postavljanje vrijednosti, što omogućava komponentama dublje u hijerarhiji da mijenjaju vrijednost varijable. Nadalje, korištenje useState udica omogućava da se dio web-stranice gdje se koristi „stateful“ varijabla iznova generira pri promjeni vrijednosti te varijable [1].

useEffect udica se koristi za definiranje „callback“ procedura koje se obavljaju pri promjeni neke (ili više) varijabli sadržanih u „dependency array“ (polju ovisnosti). Polje može biti i prazno, pri čemu se procedura izvrši jednom pri učitavanju komponente. Kao varijable se uglavnom koriste upravo „stateful“ varijable definirane u useState udicama [1].

### **1.1.2. TypeScript**

TypeScript je korišten kao nadogradnja na JavaScript zbog svojih značajki poput statičkog tipiziranja, što doprinosi većoj pouzdanosti i održivosti koda. TypeScript omogućava otkrivanje potencijalnih grešaka već u fazi razvoja, što značajno smanjuje broj grešaka pri testiranju aplikacije i štedi vrijeme na debuggiranju [2].

Statičko tipiziranje pokazalo se korisnim pri definiranju sučelja „Props“ u svakoj React komponenti, gdje se specificiraju parametri koje komponenta zahtijeva od roditelja te njihov tip. TypeScript također omogućava definiranje vlastitih tipova, što sam iskoristio za definiranje oblika objekata koje korisnička strana zaprimiti pri pozivu poslužitelja, čime se olakšava integracija s poslužiteljem.

### **1.1.3. Vite**

Kao potporu pri izgradnji korisničke strane aplikacije odabrao sam Vite, moderni alat koji se fokusira na brzinu i učinkovitost razvoja. Razvijen od strane tvorca Vue.js, Vite koristi inovativne tehnike za poboljšanje iskustva programera i performansi aplikacije [3].

Jedna od ključnih značajki Vitea je brzo pokretanje i razvoj, omogućeno korištenjem nativnih ES modula i pametnim određivanjem zavisnosti, što rezultira trenutnim pokretanjem razvojnog servera. Vite također podržava Hot Module Replacement (HMR), omogućujući brzo osvježavanje modula bez potrebe za ponovnim učitavanjem cijele stranice. Ova značajka značajno skraćuje vrijeme razvoja i testiranja, povećavajući produktivnost [3].

Sve ove značajke čine Vite izuzetno moćnim alatom za moderne projekte, poboljšavajući ukupnu učinkovitost i performanse aplikacije, te ga čine idealnim izborom za moj završni rad.

### **1.1.4. Google Maps API**

U sklopu rada također sam koristio Google Maps API, posebice Places API, Geocoding API, te Routes API, za integraciju lokacijskih podataka u razvijenu web-aplikaciju. Integracija ovog API-ja omogućila je prikazivanje odabranih mjesta u planu na interaktivnoj karti, te dohvaćanje podataka o obližnjim atrakcijama i smještaju. Kao glavnu

vizualnu komponentu koristio sam Google Maps kartu, te Google Maps markere koje sad iscrtavao na karti.

Proces integracije započeo je stvaranjem projekta unutar Google Developers konzole i generiranjem API ključa za pristup Google Maps API-ju. Integracija je ostvarena putem HTTP zahtjeva prema Google Maps API-u, koristeći odgovarajuće endpointe za pretraživanje i dohvaćanje detalja o lokacijama. Obradom dobivenih odgovora, aplikacija može prikazati relevantne informacije korisniku.

## **1.2. Poslužiteljska strana**

### **1.2.1. Spring Boot**

Za poslužiteljsku stranu aplikacije korišten je Spring Boot, koji je dio ekosustava Spring Frameworka. Spring Boot pojednostavljuje razvoj, testiranje i implementaciju robusnih i skalabilnih web-aplikacija u Javi. Ova tehnologija pruža niz značajki koje ubrzavaju razvoj aplikacije, kao što su automatska konfiguracija i ugrađeni poslužitelji poput Tomcata i Jettyja.

Spring je bio prirodan odabir zbog svoje široke podržanosti, robusne dokumentacije, lakoće korištenja, performansi te mnogih drugih poželjnih značajki [4]. Osim samog Spring Boota, jedan od glavnih dijelova koje sam koristio je Spring Security framework, koji omogućava relativno jednostavno postavljanje sigurnosne infrastrukture za web-aplikaciju, poput filtera i uloga [4]. Kao komplementarnu tehnologiju Spring Securityju uveo sam JSON Web Token (JWT), koji se koristi u kolačićima za autentifikaciju i autorizaciju korisnika.

### **1.2.2. SocketIO**

Osim samog Spring frameworka, koristio sam i implementaciju WebSocketa u Javi, odnosno SocketIO. SocketIO se pokreće kao zasebna komponenta poslužiteljske strane na različitom portu od glavnog servera te služi za facilitaciju komunikacije među korisnicima koji uređuju isto putovanje. U tu svrhu koristi se podjela na takozvane sobe (rooms) gdje korisnici koji uređuju isto putovanje ulaze u istu sobu te šalju poruke na SocketIO server, koji prosljeđuje te poruke svim ostalim korisnicima, omogućujući komunikaciju i kolaboraciju u stvarnom vremenu.

## 1.3. Baza podataka

### 1.3.1. PostgreSQL

PostgreSQL je odabran kao sustav za upravljanje bazom podataka zbog svoje stabilnosti, performansi i podrške za napredne SQL značajke. PostgreSQL je objektno-relacijska baza podataka otvorenog koda koja omogućava efikasno upravljanje velikim količinama podataka te pruža robusne sigurnosne značajke i podršku za transakcije, što je čini superiornijom od jednostavnih SQL baza podataka [5].

Jedna od ključnih prednosti PostgreSQL-a je njegova sposobnost rada s kompleksnim upitima i podrška za napredne funkcionalnosti poput rekurzivnih upita, common table expressions (CTE) i JSONB tipa podataka. Također, PostgreSQL pruža podršku za replikaciju i visoku dostupnost, što je ključno za aplikacije koje zahtijevaju konstantan pristup podacima i minimalne prekide u radu [5].

Tijekom razvoja, baza podataka bila je pohranjena lokalno, a za pregled i upravljanje bazom koristio sam alat pgAdmin 4. pgAdmin 4 je pružao intuitivno sučelje za administraciju, omogućavajući jednostavno izvršavanje upita, pregled strukture baze podataka, te upravljanje korisnicima i dozvolama.

### 1.3.2. Liquibase

Liquibase je alat za upravljanjem verzijom modela baze podataka. Korištenjem intuitivnog XML markup jezika korisniku se omogućuje lagano i transparentno upravljanje modelom baze podataka [6]. Promjene u modelu provode se kroz <changeSet> oznake, gdje se definira što treba promijeniti u modelu. U tu svrhu, u strukturu aplikacije poslužiteljske strane dodana je mapa resources/db.changelog, gdje se nalazi datoteka changelog.xml u kojoj se nalazi definicija modela. Neki od češće korištenih oznaka su <createTable> za kreiranje novih tablica u bazi, <column> za definiranje stupaca u tablici te <constraint> za definiranje ograničenja na neki stupac [6].

Model baze zatim se lako ažurira uporabom Liquibase „update“ naredbe iz komandnog retka ili pak integriranog razvojnog okruženja. Liquibase u model baze dodaje dvije dodatne tablice, databasechangelog u koju se pohranjuju već izvršene promjene baze te databasechangeloglock koji osigurava da je u svakom trenutku aktivna samo jedna instanca Liquibasea [6].



## **2. Zahtjevi sustava**

### **2.1. Generički zahtjevi aplikacije**

- Mehanizam autentifikacije i autorizacije, temeljen na korisničkom imenu i lozinki
- Spremanje i izmjena korisničkih podataka
- Intuitivno i easy-to-use korisničko sučelje
- Mogućnost naknadnog ažuriranja podataka
- Odzivnost i responzivnost
- Sigurnost i privatnost podataka

### **2.2. Specifični zahtjevi aplikacije**

- Stvaranje i uređivanje planova putovanja
- Pregled korisnikovih planova putovanja
- Interaktivna karta za vizualiziranje planova
- Preporučivanje atrakcija i smještaja u okolini mjesta na putu
- Dijeljenje planova putovanja sa zajednicom
- Real-time suradnja više korisnika na jednom planu putovanja
- Vremenska prognoza za planove unutar nekog raspona datuma
- Budžetiranje po sekcijama plana
- Bilješke za svaki dio puta

### **3. Arhitektura sustava**

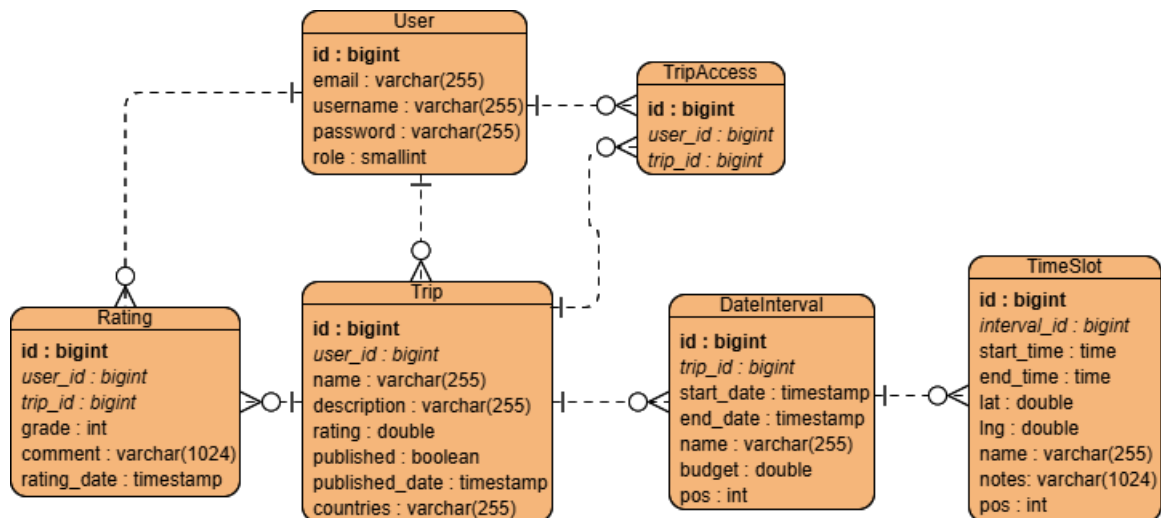
U ovom poglavlju opisat ću korišteni model arhitekture sustava te način na koji sam postavio bazu podataka.

#### **3.1. Model arhitekture sustava**

Web-aplikacija se oslanja na arhitekturu sustava klijent-poslužitelj, u kojoj korisnikov web-preglednik šalje zahtjeve na server te prikazuje odgovarajuću web-stranicu. Ovaj provjereni model omogućuje skalabilnost, modularnost i fleksibilnost sustava, što je od ključne važnosti za moj završni rad. Naime, klijentska strana je zadužena za prezentaciju i interakciju s korisnikom, dok poslužiteljska strana obavlja logičke operacije, pohranu podataka i integraciju s vanjskim servisima i bazom podataka [7]. Ovaj pristup omogućuje da se aplikacija lako proširuje i prilagođava novim zahtjevima i funkcionalnostima, što je bitno za postizanje uspjeha u današnjem dinamičnom online okruženju. Upravo ova arhitektura omogućuje stabilan temelj za razvoj aplikacije koja će biti robusna, skalabilna i korisnički orijentirana, a to je upravo ono što želim postići u okviru mog završnog rada [7].

## 3.2. Model baze podataka

U mom radu aplikacija se služi bazom podataka koja se nalazi na serverskoj strani. Baza podataka sastoji se od 6 međusobno povezanih tablica. ER dijagram baze dan je u nastavku poglavlja, kao i opisi svih tablica/entiteta prisutnih u bazi podataka.



Slika 3.1 ER dijagram baze podataka

### 3.2.1. Opisi tablica

Tablica users

- id – Jedinstveni identifikator korisnika
- email – Korisnikova e-mail adresa
- username – Korisnikov username
- password – Korisnikova lozinka
- role – Korisnikova uloga

Tablica trip

- id – Jedinstveni identifikator plana putovanja
- user\_id - Jedinstveni identifikator korisnika vlasnika
- name – Ime plana putovanja
- description – Opis plana putovanja
- rating – Ocjena plana putovanja (ako je objavljen)

- published – Zastavica koja određuje je li plan putovanja objavljen
- published\_date – Datum i vrijeme objave
- countries – Zemlje u planu putovanja (u formatu country\_code:country\_code:...)

#### Tablica trip\_access

- id - Jedinstveni identifikator pristupa
- user\_id - Jedinstveni identifikator korisnika kojem se daje pristup
- trip\_id - Jedinstveni identifikator plana putovanja kojem korisnik ima pristup

#### Tablica date\_interval (sekcije plana)

- id - Jedinstveni identifikator datumskog intervala
- trip\_id - Jedinstveni identifikator pripadajućeg plana putovanja
- start\_date - Početni datum datumskog intervala
- end\_date – Završni datum datumskog intervala
- name – Ime datumskog intervala
- budget – Budžet datumskog intervala
- pos – Redni broj datumskog intervala

#### Tablica timeslot

- id – Jedinstveni identifikator mjesta
- interval\_id – Jedinstveni identifikator pripadajućeg datumskog intervala
- start\_time – Početno vrijeme posjeta mjestu
- end\_time – Završno vrijeme posjeta mjestu
- lat – Geografska širina mjesta
- lng – Geografska duljina mjesta
- name – Ime mjesta
- notes – Bilješke o posjetu mjestu
- pos – Redni broj mjesta u sekciji plana (datumskom intervalu)

### Tablica rating

- id – Jedinstveni identifikator ocjene
- user\_id – Jedinstveni identifikator ocjenitelja
- trip\_id – Jedinstveni identifikator ocijenjenog plana putovanja
- grade – Ocjena dana planu putovanja
- comment – Komentar uz ocjenu
- rating\_date – Datum ocjenjivanja

## 4. Implementacija sustava

U ovom poglavlju opisat ću konkretnu implementaciju sustava te načine na koji su funkcionalni i nefunkcionalni zahtjevi sustava ispunjeni.

### 4.1. Struktura aplikacije

Aplikacija je strukturno podijeljena na korisničku i poslužiteljsku stranu, pri čemu je svaki dio sadržan u svojoj zasebnoj mapi radi bolje organizacije.

Osnovna struktura korisničke strane prati smjernice React knjižnice, dok je za inicijalno postavljanje projekta odabrana Vite platforma. Iako su osnovne React datoteke automatski generirane, sve ostale komponente ostavljene su korisniku na odabir, što je potaknulo moje nastojanje da što bolje organiziram React komponente radi preglednosti i lakšeg održavanja aplikacije.

Na poslužiteljskoj strani, aplikacija se sastoji od kontrolera, servisa, repozitorija te sigurnosne konfiguracije, sve implementirano uz pomoć Spring Boot anotacija. Korištenje kompozitnih Spring anotacija je uvelike olakšalo označavanje dijelova programa i dodijelilo im odgovarajuću ulogu, što je doprinijelo čistoći i jasnoći koda.

### 4.2. Implementacija korisničke strane

Struktura korisničke strane aplikacije je ključna za organizaciju i upravljanje kodom. Ovdje je pregled osnovnih mapa i konfiguracijskih datoteka:

- `package.json`: Ova datoteka upravlja ovisnostima projekta te olakšava ažuriranje i preuzimanje ovisnosti. Definira sve pakete potrebne za funkcionalnost projekta.
- `vite.config.ts`: Konfiguracijska datoteka za Vite knjižnicu. Sadrži opcije za upravljanje projektom, kao što je postavljanje proxy URL-a za slanje HTTP zahtjeva na server.
- `src` mapa: Svi TypeScript kodovi su smješteni u `src` mapi. Ključne datoteke uključuju `main.tsx` i `App.tsx`, koje sadrže vršne React komponente za renderiranje i

definiranje URL ruta [1]. U mapi components nalaze se dodatno podijeljene datoteke s kodom svih komponenti aplikacije.

- pages mapa: Ova mapa sadrži konkretne implementacije web-stranica koje korisnik vidi prilikom korištenja aplikacije, poput početne stranice ili stranice za prijavu.
- assets mapa: Sadrži sve statičke resurse poput slika, ikonica i polja boja koji se koriste u aplikaciji.

React komponente su ključni dio implementacije korisničke strane, sadrže TypeScript kod koji se pretvara u HTML za prikaz korisniku. Organiziranjem kodne strukture na ovaj način, olakšava se održavanje i nadogradnja aplikacije.

## **4.3. Bitniji korišteni paketi korisničke strane**

### **4.3.1. country-flag-icons**

Ovaj paket sadrži rječnik koji preslikava dvoslovni kod neke zemlje u jednostavnu ikonu njene zastave [8]. Također uključuje i React komponente za svaku zastavu, no ja sam preferirao upotrebu samih ikona u vlastitoj komponenti kako bih olakšao dinamičko dodavanje i uklanjanje zastava.

### **4.3.2. country-select-js**

Ovaj paket se koristi za pretvaranje tipičnog HTML elementa `<input>` u padajući izbornik zemalja. U projektu sam ga koristio za odabir zemalja prilikom kreiranja novog plana putovanja, za ažuriranje planova te za pretraživanje planova po zemljama koje sadrže. Implementacija ovog paketa uključuje učitavanje skripte nakon što se stranica prikaže, koja zatim poziva metodu `countrySelect()` nad `HTMLInputElement` objektom (u ovom slučaju, poljem za unos), pretvarajući ga u padajući izbornik zemalja [9]. Osim vidljivog izbornika, paket nudi i mogućnost korištenja nevidljivog polja tipa "text" uz izbornik, gdje se umjesto punog imena zemlje upisuje njezin dvoslovni kod [9]. Ovo sam iskoristio za prikaz zastava zemalja korištenjem paketa `country-flag-icons` navedenog u prethodnoj točki.

### **4.3.3. react-toastify**

U sklopu rada, toast poruke su široko primijenjene, posebno na stranici za uređivanje plana putovanja, ali i na ostalim dijelovima aplikacije. Umjesto klasične JavaScript `alert()`

metode, za obavještanje korisnika o raznim akcijama koristili smo `toast()` metodu za obične obavijesti te `toast.error()` metodu za prikazivanje obavijesti o greškama [10]. Toast poruke su efikasnije od klasičnih alert poruka jer ne ometaju rad korisnika, automatski nestaju nakon nekog vremena te imaju estetski privlačniji izgled.

React-toastify uvodi React komponentu `ToastContainer` koja služe kao kontejneri za toast poruke [10]. Dobra praksa je uključiti jedan takav kontejner kao vršni element u svaku stranicu aplikacije. Ovaj paket nudi mnoge mogućnosti prilagodbe izgleda i ostalih parametara toast poruka, čineći ih snažnim alatom za neinvazivnu komunikaciju s korisnicima.

#### **4.3.4. @vis.gl/react-google-maps**

Zadnji, ali ne manje važan korišteni paket je naslovljen `@vis.gl/react-google-maps`. Ovaj paket razvijao se samostalno dok Google nije službeno podržao njihov razvoj. Glavna značajka paketa je veoma olakšan pristup Google Maps API-ju koristeći jednu React komponentu, `APIProvider` [11].

Najlakše je i najbolja je praksa ovu komponentu staviti kao vršnu HTML komponentu u stranici koja zahtjeva pristup API-ju. Komponenti se predaju API ključ kao jedan od glavnih parametara, te komponenta dalje upravlja učitavanjem API-ja i relevantnih skripti. Unutar `APIProvidera`, sve ostale React komponente u paketu poput `Map`, `AdvancedMarker`, `Pin` i slično imat će pristup Maps API-ju i funkcionirati kako treba. Nudi se pristup cijelom Google Maps API-ju, uključujući `Geocoding API`, `Places API` te `Routes API`, koji se koriste u aplikaciji [11].

`Map` komponenta glavna je vizualna komponenta dostupna korisniku pri uređivanju plana putovanja, služeći kao prikaz njegova plana. Unutar `Map` komponente dodaju se mnogi `AdvancedMarker` elementi, koji su markeri koji se iscrtavaju na karti. Koriste se uz `Pin` elemente koji definiraju izgled svakog markera. Nadalje, za dohvaćanje atrakcija i smještaja koristi se `Places API`, te se za računanje i iscrtavanje ruta između mjesta na karti koristi upravo `Routes API` [11].



## 4.4. Implementacija poslužiteljske strane

Struktura poslužiteljske strane podijeljena je na nekoliko mapa koje sadrže ključne komponente Spring frameworka, omogućavajući izvršavanje poslovne logike i obradu HTTP zahtjeva.

Najvažniji dio implementacije poslužiteljske strane su REST kontroleri, koji se bave upravljanjem HTTP zahtjevima. U izvornom kodu ovih kontrolera definirane su pristupne točke programa na kojima se zaprimaju zahtjevi različitih HTTP metoda (GET, POST, PUT, DELETE). Aplikacija uključuje nekoliko kontrolera koji pokrivaju različite aspekte poslovne logike:

- **UserController:** Koristi se za dohvaćanje informacija o korisnicima i upravljanje korisničkim podacima.
- **AdminController:** Služi za dohvaćanje podataka potrebnih za administrativni panel.
- **AuthenticationController:** Odgovoran je za registraciju, prijavu korisnika u sustav te validaciju JSON web tokena.
- **TripController:** Najopširniji kontroler koji upravlja planovima putovanja, ažurira opće podatke o planu, pridjeljuje pristup planu te omogućava dohvaćanje i pretraživanje planova putovanja.
- **DateIntervalController:** Upravlja datumskim intervalima (sekcijama plana putovanja).
- **TimeSlotController:** Služi za upravljanje vremenima.
- **RatingController:** Koristi se za upravljanje ocjenama.

Osim kontrolera, poslužiteljska strana sadrži servise i repozitorije. Servisi sadrže nižu razinu implementacije poslovne logike, dok repozitoriji komuniciraju s bazom podataka s pomoću Spring frameworka.

Jedna od važnih komponenti poslužiteljske strane su i DTO (data transfer object) implementacije. Ovi objekti omogućuju jednostavnu komunikaciju i razmjenu podataka između korisničke i poslužiteljske strane koristeći unaprijed definirane formate objekata.

Za implementaciju SocketIO servera korišten je paket `com.corundumstudio.socketio`, koji sadrži osnovnu implementaciju `SocketIOServera` i različite promatrače (`ConnectListener`,

DataListener, DisconnectListener) za definiranje ponašanja servera pri zaprimanju zahtjeva na svom portu. SocketIO server pokreće se na zasebnom portu od Spring aplikacije, ali postoji način da se konfigurira port forwarding kako bi cijela aplikacija radila na istom portu.

## 4.5. Integracija s bazom podataka

Integracija s lokalnom bazom podataka provedena je korištenjem integriranih Spring framework rješenja. Za to su korištene `@Repository` anotacije u kombinaciji s `JpaRepository` sučeljima te `@Entity` anotacije uz klase koje predstavljaju konkretne tablice u bazi podataka. Svaki entitet ima definiran primarni ključ te veze s ostalim entitetima putem anotacija `@ManyToOne` i `@OneToMany` [12].

Spring, unutar svojeg starter-web paketa, koristi Hibernate kao sučelje između aplikacije i baze podataka. `JpaRepository` sučelje omogućuje pisanje SQL upita na jednostavan način, gdje se ime metode `findBy...` nadopunjuje uvjetima te može ovisiti o parametrima predanim funkciji, eliminirajući tako potrebu za ručnim pisanjem SQL upita. Osim tih korisničkih metoda, postoje i osnovne metode poput `save`, `delete` i `findById`, koje su manje-više samorazumljive [12].

Za konfiguriranje spajanja na bazu podataka definiraju se `spring.datasource.url`, `spring.datasource.username` i `spring.datasource.password` u `resources/application.properties` konfiguracijskoj datoteci. Također se može definirati i SQL dijalekt, koji je standardno postavljen na `postgresql` zbog Hibernate implementacije.

## 4.6. Implementacija sigurnosnih značajki

Sigurnost sustava temelji se na JSON web tokenima i datotekama `SecurityConfig` te `JWTAuthenticationFilter`. `SecurityConfig` datoteka definira lanac filtera kroz koje svaki primljeni zahtjev mora proći prije nego što ga obrađuje neki od kontrolera. U aplikaciji su definirane tri glavne rute: `/api/auth/`, `/api/admin` i `/api/core`. Pristup administratorskoj ruti zahtjeva ulogu administratora, dok su ostale dvije rute otvorene za bilo kojeg korisnika.

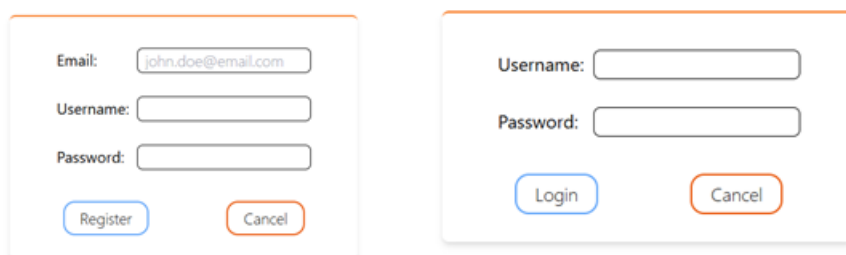
`JWTAuthenticationFilter` je ključni dio ovog lanca filtera. U njemu se provjerava polje "Authorization" u HTTP zaglavlju. Ako je zaprimljeni JSON web token validiran, korisnik se propušta kroz filter; u suprotnom, korisniku se vraća statusni kod 403 Forbidden. Ovaj

filter igra ključnu ulogu u autentifikaciji korisnika i osiguravanju pravilnog pristupa resursima u aplikaciji.

## 5. Korištenje sustava

U ovom poglavlju provest ću vas kroz korištenje aplikacije.

### 5.1. Autentifikacija



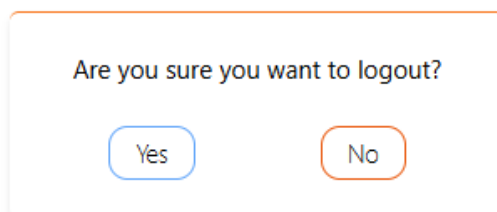
The image shows two side-by-side forms for user authentication. The left form is for registration, with fields for Email (pre-filled with 'john.doe@email.com'), Username, and Password. It has 'Register' and 'Cancel' buttons. The right form is for login, with fields for Username and Password. It has 'Login' and 'Cancel' buttons.

Slika 5.1 Forma za registraciju i prijavu u sustav

Kako bi se koristili stranicom, korisnici se najprije moraju registrirati, ili ako su veću registrirani, prijaviti u sustav. Registracija i prijava odvijaju se jednostavnom formom gdje korisnici upisuju svoje podatke.

Pri registraciji postoji nekoliko ograničenja; naime, korisnikova e-mail adresa i korisničko ime moraju biti jedinstveni, odnosno različiti od svih već registriranih korisnika. Nadalje, korisnikova lozinka mora biti duga barem 8 znakova, kako bi se osigurala minimalna jačina lozinke.

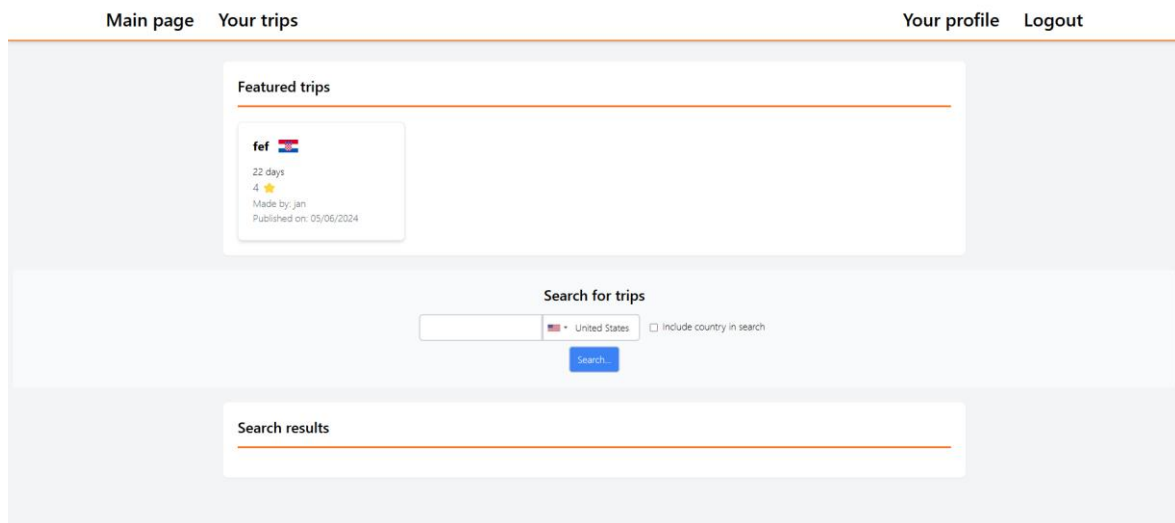
Ako se korisnik želi odjaviti iz sustava, može klikom na Logout u navigacijskoj traci otići na stranicu za odjavu. Na stranici za odjavu korisnik mora doista potvrditi da se želi odjaviti iz sustava, nakon čega ga se odjavljuje iz sustava i preusmjeruje na početnu stranicu.



The image shows a confirmation dialog box with the text 'Are you sure you want to logout?'. It has two buttons: 'Yes' and 'No'.

Slika 5.2 Upit korisniku o odjavi

## 5.2. Početna stranica



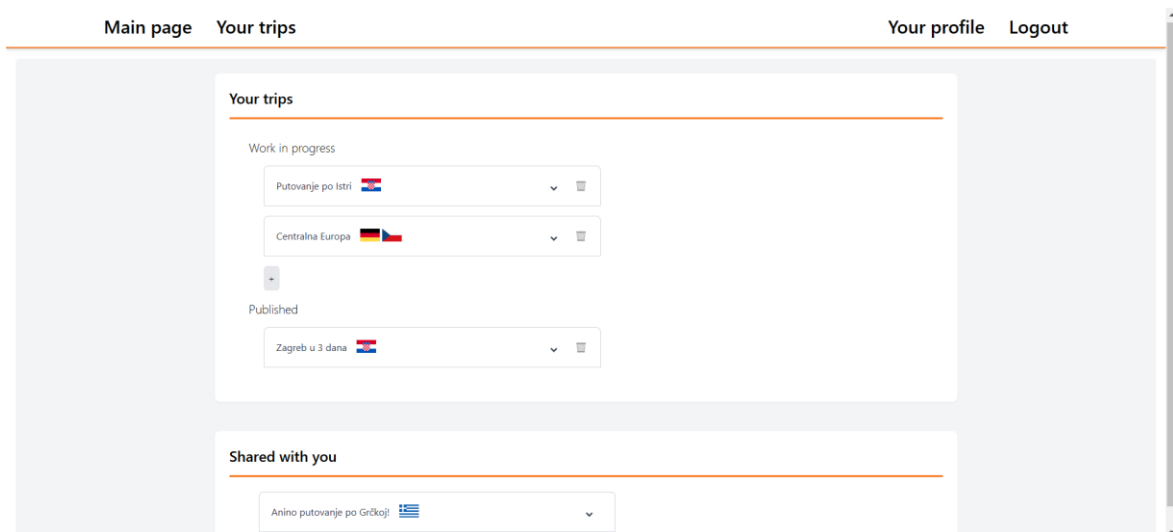
Slika 5.3 Početna stranica

Na slici 5.2 prikazana je početna stranica nakon što se korisnici prijave u sustav. Na svakoj stranici, pa tako i na početnoj, na vrhu se nalazi navigacijska traka. Kada je korisnik prijavljen u sustav, na traci se nalaze veze na njegove planove putovanja, njegov profil, te stranicu za izlazak iz sustava.

Glavni dio početne stranice sastoji se od sekcije s predloženim planovima putovanja i sekcije za pretraživanje planova putovanja. Predloženi planovi putovanja biraju se prema njihovoj ocjeni, te se 5 najbolje ocijenjenih planova prikaže u svojim kućicama.

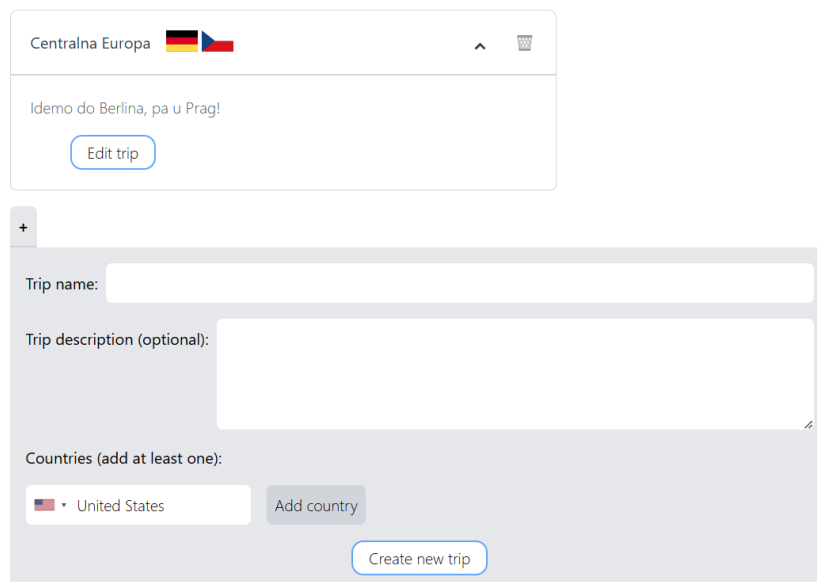
Traka za pretraživanje ima dva polja, polje za upis dijela imena plana putovanja, te polje za odabir jedne od zemalja plana putovanja, s dodatnom kućicom koja se označi ako se odabrana zemlja želi uzeti u obzir pri pretraživanju. Rezultati pretraživanja prikazu se u istom formatu kao i predloženi planovi putovanja, s iznimkom da se prikaže poruka ako pretraživanje ne pronade nijedan rezultat.

## 5.3. Korisnikovi planovi putovanja



Slika 5.4 Stranica s korisnikovim planovima putovanja

Na ovoj stranici korisnici mogu vidjeti svoje planove putovanja. U to su uključeni njihovi vlastiti planovi, oni u tijeku uređivanja te oni koje su već podijelili sa zajednicom. Svaki od korisnikovih vlastitih planova ima u svojem retku ikonicu koša za smeće kojim se plan briše, uključivo i iz baze objavljenih planova. Korisnik ne može obrisati plan koji je s njim samo podijeljen.



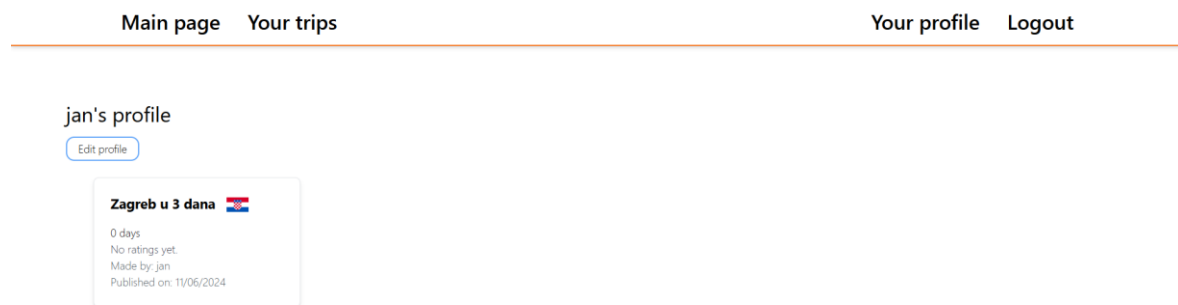
Slika 5.5 Pregled plana putovanja, forma za stvaranje novog plana putovanja

Kada korisnik pritisne na neki od planova putovanja, proširi se njegova kućica te se prikaže opis tog plana putovanja kao i gumb koji vodi na uređivanje tog plana. Ako je plan

već objavljen, tada gumb vodi na pregled tog plana, te samim time na gumbu piše „View Trip“ umjesto „Edit Trip“.

Klikom na plusić, otvara se forma za stvaranje novog plana putovanja. Forma od korisnika zahtjeva da obavezno unese ime plana putovanja, te barem jednu zemlju na koju se plan putovanja odnosi, dok je opis putovanja opcionalan i ne mora odmah biti specificiran. Klikom na gumb novi plan putovanja je stvoren te ga korisnik može započeti ispunjavati i uređivati.

## 5.4. Korisnikov profil



Slika 5.6 Korisnikov profil

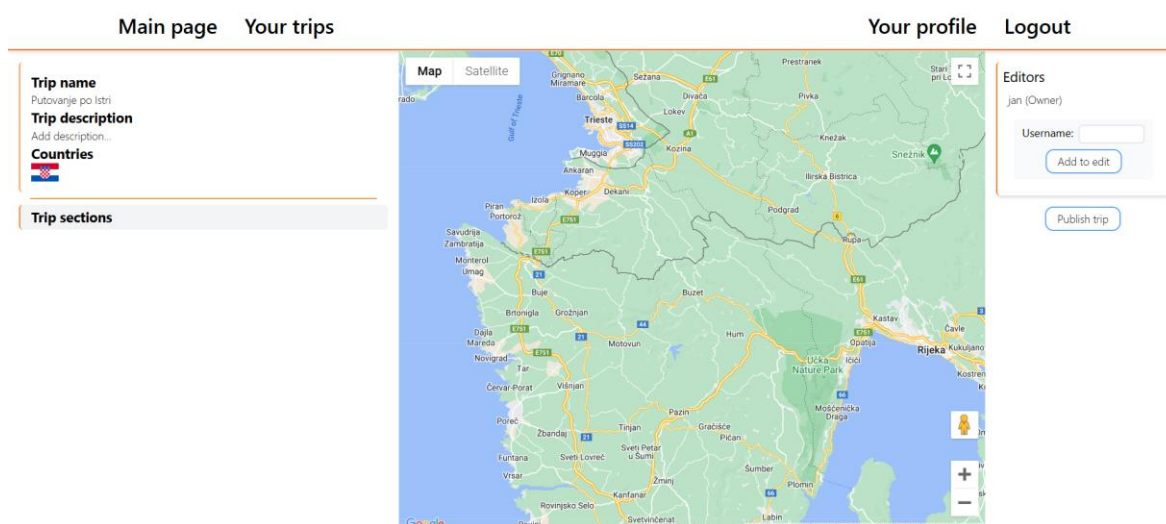
Klikom na gumb „Your profile“ u navigacijskoj traci korisnici mogu vidjeti vlastiti profil. Profil je relativno minimalističan, odnosno navedeni su samo korisnikovi objavljeni planovi putovanja. Korisnici mogu pregledavati profile ostalih korisnika, no gumb za uređivanje profila prikazuje se samo na korisnikovom vlastitom profilu. Klikom na gumb korisnika se odvede na stranicu za uređivanje profila.

Slika 5.7 Forma za uređivanje korisničkog profila

S pomoću ove forme, korisnici mogu mijenjati detalje svojeg korisničkog profila, odnosno svoje korisničko ime, e-mail, te lozinku. Za svaku promjenu potrebno je unijeti trenutnu korisničku lozinku. Korisnici su upozoreni ako novi e-mail ili novo korisničko ime već postoji, te se stoga ne može koristiti. Pritiskom na gumb „Update“ uz odsustvo greški, korisnički podaci se ažuriraju te se od korisnika ponovo zahtjeva da se prijavi u sustav.

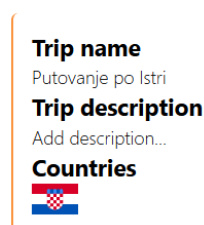
## 5.5. Uređivač plana putovanja

Glavni dio aplikacije upravo je stranica za uređivanje planova putovanja. Stranica je kao takva podijeljena na 3 glavna dijela, od kojih će svaki biti opisan zasebno.



Slika 5.8 Cijela stranica za uređivanje plana putovanja

### 5.5.1. Osnovni detalji plana putovanja



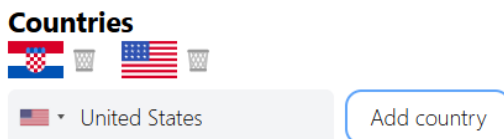
Slika 5.9 Prikaz osnovnih detalja plana putovanja

U gornjem lijevom kutu stranice nalazi se odjeljak gdje su navedeni osnovni parametri plana putovanja, njegovo ime, opis, te zemlje koje su sadržane u planu. Kada korisnik kursorom lebdi nad vrijednostima parametara, njihova pozadinska boja se promjeni,



ukazujući na mogućnost uređivanja tih parametara. Pritiskom na neku od vrijednosti otvara se polje gdje se može ažurirati taj parametar.

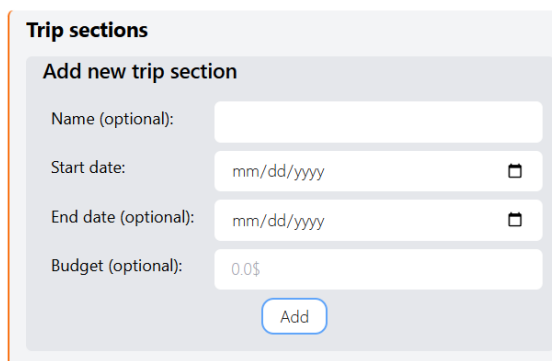
Ovaj postupak je standardan na razini cijele stranice te se manje-više svaka vrijednost može naknadno ažurirati na ovaj način te postoji indikacija da se klikom na polje ono može mijenjati.



Slika 5.10 Uređivanje zemalja putovanja

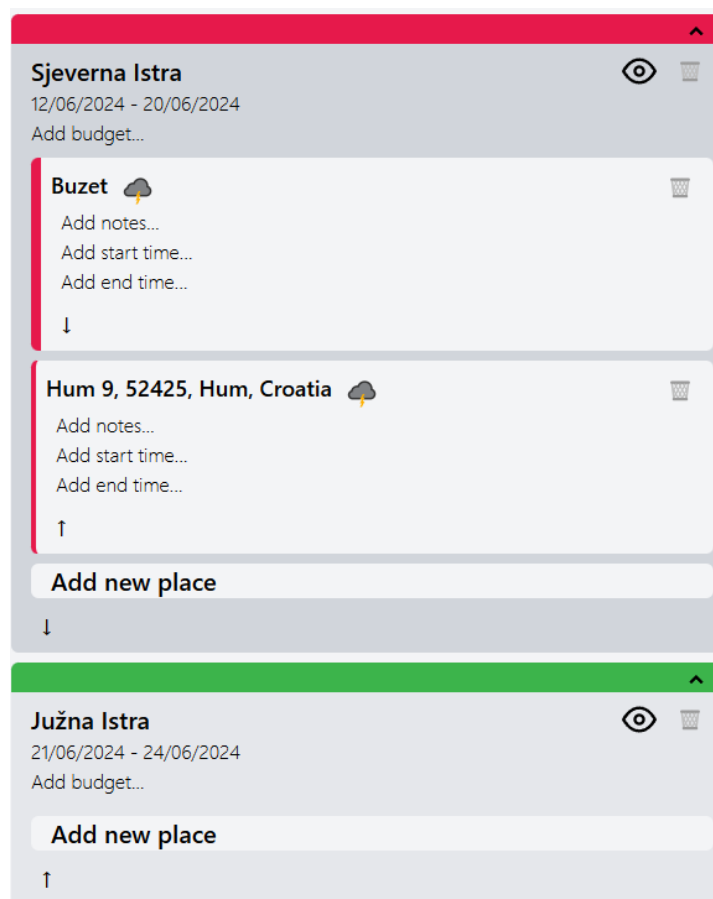
Klikom na polje sa zastavama zemalja, prikaže se sljedeća jednostavna forma. Kako je uvijek potrebno da je definirana barem jedna zemlja u putovanju, ikonice koša za smeće kojima se zemlje uklanjaju su prisutne jedino ako je definirano više od jedne zemlje u planu. Klikom na gumb „Add country“, odabrana zemlja dodaje se u plan putovanja slično kao i kod stvaranja novog plana, naravno uzimajući u obzir da se ista zemlja ne doda više puta.

### 5.5.2. Sekcije plana, mjesta u putovanju



Slika 5.11 Forma za kreiranje nove sekcije plana putovanja

Klikom na gumb naslov „Trip sections“ otvara se prikaz svih dijelova plana, te se dodatnim klikom na naslov „Add new trip section“ otvara forma za kreiranje nove sekcije plana. Sekcija se definira svojim početnim datumom, a sve ostalo je opcionalno. Ako sekciji nije pridodan krajnji datum, podrazumijeva se da ona traje samo jedan dan. Kućica za dodanu sekciju nosit će ime sekcije ako je ono uneseno, u suprotnom će se zvati „unnamed“. Budžet sekcije je također opcionalan te će biti 0\$ ako nije drugačije definirano. Svi parametri mogu se naknadno mijenjati.



Slika 5.12 Prikaz sekcija plana

Svakoj sekciji pridijeljena je distinktna boja kojom je obojeno njeno zaglavlje. U istoj boji dekorirana su i mjesta koja pripadaju toj sekciji.

Nakon dodavanja sekcije, možemo još jednom proširiti formu klikom na naslov „Add new place“, gdje možemo upisati detalje novog mjesta kojeg želimo posjetiti u toj sekciji. Ime mjesta te destinacija su obavezni parametri, dok su bilješke o mjestu te početno i završno vrijeme posjeta neobavezni parametri. Polje za destinaciju je zapravo Google Autocomplete za mjesta, te klikom na neku od ponuđenih opcija tu opciju potvrđujemo te se njeno ime zapisuje kao ime mjesta. Alternativno, moguće je klikom na zastavicu desno od polja aktivirati selektiranje na karti, čime je korisniku omogućeno da klikom na kartu odabere točno mjesto koje želi posjetiti. Pozadina ikone zastavice postane zatamnjena kada je takva selekcija aktivna.

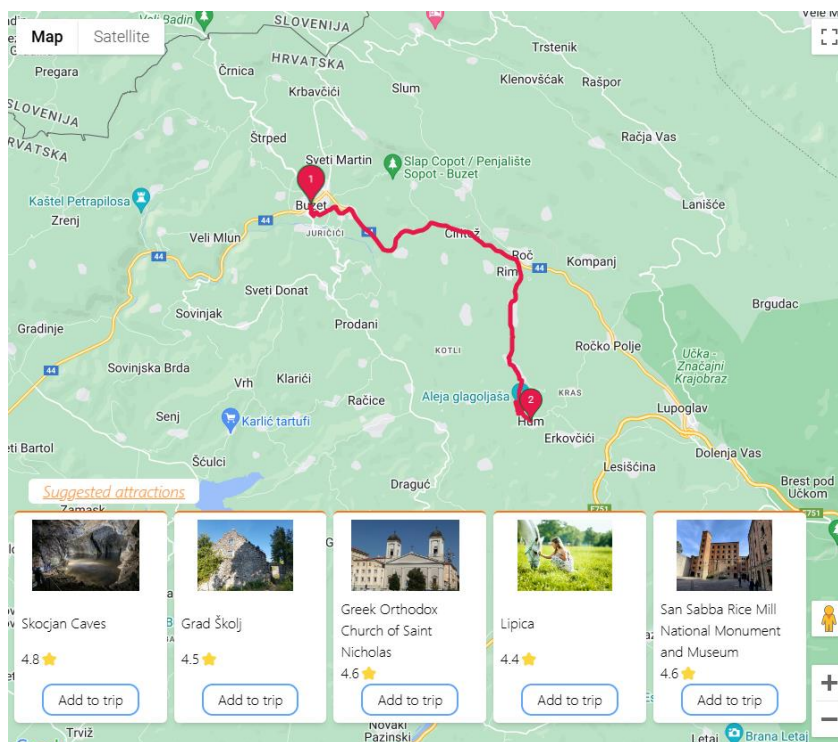
Kada je novo mjesto dodano, ono se pojavi iznad forme za stvaranje novog mjesta. Svi parametri su naknadno izmjenjivi. Ako je datum početka sekcije plana u narednih 14 dana od trenutnog datuma, aplikacija će automatski zatražiti vremensku prognozu te prikazati ikonicu predviđenih vremenskih prilika na taj datum.

Klikom na kućicu mjesta, to mjesto postaje selektirano. Selektirano mjesto ima deblji obrub s lijeve strane nego ostala mjesta, te je relevantno za ostale dvije glavne komponente ove stranice.

Također relevantno za komponentu karte, klikom na ikonu oka na vrhu sekcije, onemogućuje se iscrtavanje ruta za tu sekciju na karti, a ponovnim klikom se rute ponovno iscrtavaju.

Sve komponente mogu se sklopiti, čime se štedi na vertikalnom otisku komponente. Strelice gore-dolje na komponentama mogu se koristiti kako bi se uredio poredak dijelova plana.

### 5.5.3. Interaktivna karta, prijedlozi atrakcija

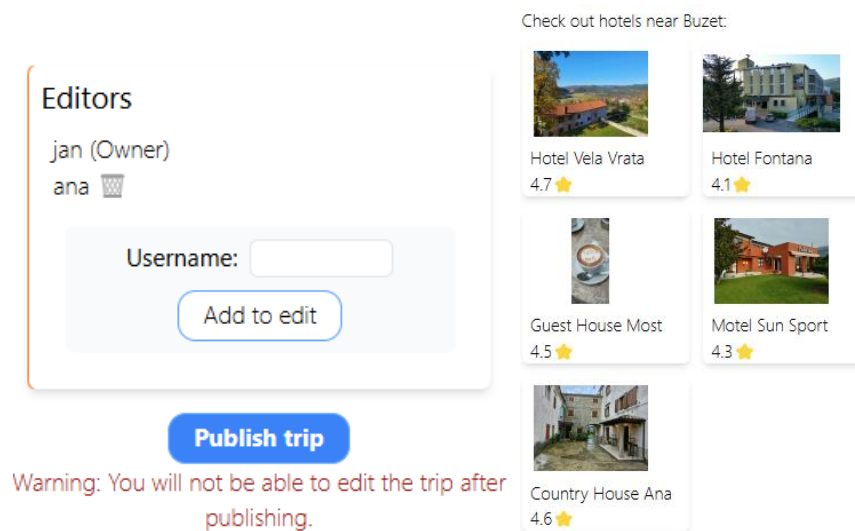


Slika 5.13 Interaktivna karta u sredini stranice

Interaktivna karta još je jedna od bitnih komponenti na stranici. Na karti se bojom određene sekcije plana iscrtavaju markeri koji su pozicionirani na mjesta sadržana u toj sekciji. Marker su numerirani redom kojim su mjesta navedena u sekciji plana. Nadalje, markeri se mogu vući po karti, čime se nanovo izračuna i iscrtava ruta te se pozicija mjesta u planu ažurira. Pri dodavanju novog mjesta u sekciju, karta se centrira na lokaciju novog mjesta. Kao što je navedeno u prijašnjem poglavlju, klikom na neko od već dodanih mjesta ono postaje selektirano. U tom slučaju na dnu karte se pojave kartice s atrakcijama koje su

blizu tog mjesta, te opcija da se te atrakcije ubaci u plan putovanja klikom na gumb. Prikazana je mala slika atrakcije te njena ocjena na Googleu.

#### 5.5.4. Ostali sudionici, objava plana, prijedlozi hotela



Slika 5.14 Desni dio stranice

Na desnoj strani stranice nalazi se dio za uređivanje sudionika. Ovdje se s pomoću korisničkog imena može pozvati ostale korisnike da pomognu pri planiranju putovanja. Naravno, jedino vlasnik plana ima pristup toj funkcionalnosti. Također, vlasnik može objaviti plan putovanja s pomoću klika na gumb „Publish trip“, čime se plan više ne može uređivati. Držanjem kursora nad gumbom prikaže se poruka vlasniku plana da se plan neće moći uređivati nakon što ga se objavi. Ako je neko mjesto selektirano, ispod gumba za objavu može se nekoliko kartica s hotelima u blizini tog mjesta. Klik na neki od hotela vodi na Google pretragu za ime hotela i datume sekcije plana u kojem je selektirano mjesto.

#### 5.5.5. Pregled objavljenog plana putovanja

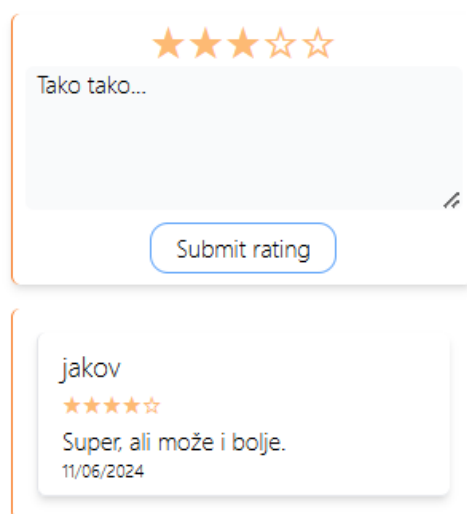
Objavljeni planovi putovanja gledaju se na istoj stranici gdje se uređuju s nekolicinom promjena. Na ovoj varijanti stranice nije moguće ništa mijenjati. Pristup ovoj varijanti stranice imaju i neregistrirani korisnici, no smiju samo gledati.

Iznad osnovnih detalja plana dodan je paragraf „Published by korisnicko\_ime“ te datum objave putovanja. Nadalje, u sekcijama i mjestima plana svi placeholderi koji su služili za

naknadno dodavanje i ažuriranje podataka su maknuti, a nepostavljene vrijednosti se jednostavno ne prikazuju. Na karti se markeri ne mogu vući, već su skroz nepomični.

Korisnici koji pregledavaju plan putovanja mogu odlučiti napraviti kopiju plana te nešto promijeniti ili ažurirati ako žele, za što se služe gumbom s desne strane s natpisom „Copy published trip and edit“. Neregistrirani korisnici umjesto toga vide gumb s natpisom „Sign in to copy trip“, koji ih vodi na stranicu za prijavu u sustav.

Vanjski korisnici koji nisu vlasnik ili jedan od sudionika plana, te nisu već ocijenili taj plan imaju mogućnost ocjenjivanja plana kroz formu koja se pojavi na desnoj strani stranice.



Slika 5.15 Ocjenjivanje plana putovanja

Korisnik klikom na jednu od zvjezdica određuje ocjenu plana od 1 do 5. Korisnik mora dodati i nekakav komentar kako bi se ocjena mogla poslati. Svoju ocjenu korisnik može naknadno obrisati. Ispod forme nalaze se sve trenutne ocjene, s imenom korisnika koji je ocjeni, ocjenom, komentarom i datumom ocjenjivanja.

## 5.6. Administratorska upravljačka ploča

Uz obične korisnike, aplikacija podržava i administratorski način rada. Kako bi se aktivirao administratorski način rada, administrator se mora prijaviti u sustav s posebnim računom, koji nosi rezervirano korisničko ime „admin“. Lozinka administratorskog računa unaprijed je zadana te je isti unesen u bazu podataka pri njenoj inicijalizaciji.

Users and their trips

Username  Trip name

ana	Anino putovanje po Grčkoj!	View trip <input type="button" value="🗑"/>
jan	Zagreb u 3 dana	View trip <input type="button" value="🗑"/>
	Centralna Europa	View trip <input type="button" value="🗑"/>
	Putovanje po Istri	View trip <input type="button" value="🗑"/>

Slika 5.16 Administratorska upravljačka ploča

Administrator na upravljačkoj ploči vidi pregled svih korisnika koji imaju bar 1 aktivan plan putovanja, što uključuje planove u fazi planiranje te objavljene planove. Koristeći alat za pretraživanje administrator može suziti područje pretrage ovisno o korisniku te o imenu plana putovanja. Klikom na košaricu uz plan putovanja administrator može prisilno obrisati taj plan putovanja. Pritiskom na gumb „View Trip“ administrator pristupa stranici za planiranje putovanja prikazanoj u poglavlju [5.5](#). Administrator ima pristup svoj funkcionalnosti stranice za planiranje putovanja te može manipulirati plan putovanja prema vlastitoj volji.

## Zaključak

Tijekom razvoja aplikacije uspješno su implementirane gotovo sve predviđene funkcionalnosti. Međutim, postoji nekoliko područja koja bi se mogla dodatno unaprijediti. Mogla bi se poboljšati korisnička sučelja radi još intuitivnijeg korištenja. Aplikacija bi mogla uključivati više podataka i prijedloga baziranih na korisničkim recenzijama i povratnim informacijama, čime bi se povećala relevantnost i kvaliteta prijedloga atrakcija. Dodatne funkcionalnosti poput integracije s društvenim mrežama i naprednije analitike korisničkih preferencija mogle bi dodatno obogatiti korisničko iskustvo. Nadalje, aplikaciji bi koristio dnevnik promjena plana, gdje vlasnik može vidjeti tko je što mijenjao i kada, te poništiti promjene ako mu ne odgovaraju.

Razvoj ove aplikacije pokazao je kako inovativna tehnološka rješenja mogu značajno olakšati planiranje putovanja te omogućiti korisnicima jednostavno dijeljenje svojih iskustava i suradnju s drugim putnicima. Nastavak rada na ovom projektu može donijeti još kvalitetnije i personaliziranije usluge korisnicima, čime bi se dodatno unaprijedilo iskustvo planiranja putovanja.

# Literatura

- [1] *React reference* (2024, lipanj). Poveznica: <https://react.dev/reference/react>; pristupljeno 4. lipnja 2024.
- [2] *TypeScript from scratch* (2024, lipanj). Poveznica: <https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html>; pristupljeno 4. lipnja 2024.
- [3] *Why Vite* (2024, lipanj). Poveznica: <https://vitejs.dev/guide/why.html>; pristupljeno 4. lipnja 2024.
- [4] *Why Spring* (2024, lipanj). Poveznica: <https://spring.io/why-spring>; pristupljeno 4. lipnja 2024.
- [5] *About PostgreSQL* (2024, lipanj). Poveznica: <https://www.postgresql.org/about/>; pristupljeno 4. lipnja 2024.
- [6] *Liquibase Concepts* (2024, lipanj). Poveznica: <https://docs.liquibase.com/concepts/home.html>; pristupljeno 10. lipnja 2024.
- [7] *Client-server model* (2024, travanj). Poveznica: <https://www.geeksforgeeks.org/client-server-model/>; pristupljeno 4. lipnja 2024.
- [8] *country-flag-icons* (2024, lipanj). Poveznica: <https://www.npmjs.com/package/country-flag-icons>; pristupljeno 6. lipnja 2024.
- [9] *country-select-js* (2021). Poveznica: <https://www.npmjs.com/package/country-select-js>; pristupljeno 6. lipnja 2024.
- [10] *react-toastify* (2024, ožujak). Poveznica: <https://www.npmjs.com/package/react-toastify>; pristupljeno 6. lipnja 2024.
- [11] *@vis.gl/react-google-maps* (2024, svibanj). Poveznica: <https://www.npmjs.com/package/@vis.gl/react-google-maps>; pristupljeno 6. lipnja 2024.
- [12] *Spring Data JPA* (2024, lipanj). Poveznica: <https://spring.io/projects/spring-data-jpa>; pristupljeno 6. lipnja 2024.



## Popis slika

Slika 3.1 ER dijagram baze podataka .....	8
Slika 5.1 Forma za registraciju i prijavu u sustav .....	17
Slika 5.2 Upit korisniku o odjavi.....	17
Slika 5.3 Početna stranica.....	18
Slika 5.4 Stranica s korisnikovim planovima putovanja .....	19
Slika 5.5 Pregled plana putovanja, forma za stvaranje novog plana putovanja .....	19
Slika 5.6 Korisnikov profil .....	20
Slika 5.7 Forma za uređivanje korisničkog profila.....	20
Slika 5.8 Cijela stranica za uređivanje plana putovanja .....	21
Slika 5.9 Prikaz osnovnih detalja plana putovanja .....	21
Slika 5.10 Uređivanje zemalja putovanja .....	22
Slika 5.11 Forma za kreiranje nove sekcije plana putovanja .....	22
Slika 5.12 Prikaz sekcija plana .....	23
Slika 5.13 Interaktivna karta u sredini stranice .....	24
Slika 5.14 Desni dio stranice .....	25
Slika 5.15 Ocjenjivanje plana putovanja .....	26
Slika 5.16 Administratorska upravljačka ploča .....	27

# Sažetak

## Web-aplikacija za planiranje putovanja

U ovom završnom radu razvijena je web-aplikacija koja omogućuje korisnicima stvaranje i spremanje planova putovanja, dijeljenje s zajednicom te suradnju u realnom vremenu. Korištenjem interaktivne karte, korisnici mogu odabrati ključna mjesta, dobiti prijedloge atrakcija te informacije o smještaju i vremenskoj prognozi. Aplikacija, razvijena koristeći tehnologije kao što su React, Spring Boot i TypeScript, pruža mogućnost budžetiranja i omogućuje dodavanje bilješki u itinerar. Implementirane funkcionalnosti unaprijedile su korisničko iskustvo te olakšale planiranje putovanja, čineći ga intuitivnijim i efikasnijim.

**Ključne riječi:** planiranje putovanja; web-aplikacija; interaktivna karta; suradnja korisnika; itinerar; vremenska prognoza; React; Spring Boot; TypeScript; Vite; SocketIO; Google Maps

# Summary

## **Web-application for travel planning**

In this thesis, a web application was developed that allows users to create and save travel plans, share them with the community, and collaborate in real time. Using an interactive map, users can select key locations, receive suggestions for attractions, and obtain information on accommodation and weather forecasts. The application, developed using technologies such as React, Spring Boot, and TypeScript, provides budgeting capabilities and allows users to add notes to their itinerary. The implemented functionalities have enhanced the user experience and made travel planning more intuitive and efficient.

**Keywords:** travel planning; web application; interactive map; user collaboration; itinerary; weather forecast; React; Spring Boot; TypeScript; Vite; SocketIO; Google Maps