SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 1497

**WEB-APLIKACIJA ZA PLANIRANJE PUTOVANJA**

Jan Grbac

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 1497

**WEB-APLIKACIJA ZA PLANIRANJE PUTOVANJA**

Jan Grbac

Zagreb, lipanj 2024.

U okviru ovog završnog rada potrebno je razviti aplikaciju kojom korisnici mogu stvarati nove planove putovanja, spremiti postojeće planove pod svoj profil te ih podijeliti sa zajednicom. Pri planiranju puta, korisnici mogu koristiti interaktivnu kartu gdje odabiru ključna mjesta u svom putu. Web-aplikacija može korisnicima predložiti atrakcije za posjetiti u određenom mjestu, te se korisnici mogu informirati o prijevozu, hotelima, vremenskoj prognozi itd. Aplikacija bi trebala korisnicima omogućiti suradnju, odnosno da istovremeno više ljudi uređuje jedan plan. Aplikacija korisnicima daje i okvirnu cijenu određenih usluga koje su odabrali (hoteli, prijevoz). Također, korisnici mogu u svoj itinerar dodavati bilješke.

Sadržaj

[Uvod 1](#_Toc23263729)

[1. Naslov prvog poglavlja 2](#_Toc23263730)

[1.1. Prvo potpoglavlje 2](#_Toc23263731)

[1.2. Stilovi za tekst, naslove i podnaslove 2](#_Toc23263732)

[1.3. Stilovi za nabrajanje 2](#_Toc23263733)

[1.3.1. Stilovi za nabrajanje s točkama i crticama 3](#_Toc23263734)

[1.4. Slike 4](#_Toc23263735)

[1.5. Tablice 6](#_Toc23263736)

[1.6. Matematički izrazi 7](#_Toc23263737)

[1.7. Programski kôd 7](#_Toc23263738)

[Zaključak 9](#_Toc23263739)

[Literatura 10](#_Toc23263740)

[Sažetak 11](#_Toc23263741)

[Summary 12](#_Toc23263742)

[Skraćenice 13](#_Toc23263743)

[Privitak 14](#_Toc23263744)

# 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.

# 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: frontend (korisnička strana), backend (poslužitelj) te izvedba baze podataka (također na poslužitelju).

Na frontendu je korišten Vite s TypeScriptom i Reactom. Vite je moderni alat za izradu i razvoj aplikacija koji omogućava brže kompiliranje i brži razvojni proces. TypeScript, nadskup JavaScripta, omogućava statičko tipiziranje, što doprinosi ranom otkrivanju grešaka i poboljšava održavanje koda. React, popularni JavaScript okvir za izgradnju korisničkih sučelja, omogućava komponentno baziran pristup koji olakšava razvoj složenih i interaktivnih korisničkih sučelja. Korištenje ovih alata osigurava da aplikacija bude responzivna, interaktivna i jednostavna za proširivanje.

Na backendu je korišten Spring Boot, robustan okvir za izgradnju aplikacija temeljenih na Javi, koji olakšava kreiranje proizvodno spremnih sustava s minimalnim konfiguracijama. Spring Boot pruža bogat set alata za razvoj sigurnih, skalabilnih i održivih aplikacija. Za real-time kolaboraciju implementiran je Socket.IO server, koji omogućava dvosmjernu komunikaciju u stvarnom vremenu između klijenta i poslužitelja. Socket.IO je idealan za aplikacije koje zahtijevaju brzu i pouzdanu razmjenu podataka, kao što su chat aplikacije, aplikacije za suradnju u stvarnom vremenu i sl.

Baza podataka je PostgreSQL, moćna objekta-relacijska baza podataka poznata po svojoj stabilnosti, skalabilnosti i bogatom skupu značajki. PostgreSQL podržava napredne SQL funkcionalnosti i omogućava učinkovito upravljanje velikim količinama podataka. Ova baza podataka osigurava dosljednost i integritet podataka te omogućava provođenje složenih upita i transakcija.

Sve ove tehnologije zajedno omogućavaju izradu moderne, funkcionalne i pouzdane web-aplikacije koja ne samo da zadovoljava korisničke potrebe, već i omogućava jednostavno održavanje i buduće proširenje funkcionalnosti. Integrirani pristup korištenjem Vite-a, TypeScript-a, React-a, Spring Boot-a, Socket.IO-a i PostgreSQL-a omogućava razvoj aplikacija koje su skalabilne, sigurne i spremne za dinamične izazove suvremenog weba.

## Frontend

### 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.

Komponente su neovisne jedna o drugoj, koristeći prosljeđivanje podataka kroz hijerarhiju kako bi se ostvarila komunikacija. Glavni način međudjelovanja komponenti je korištenjem takozvanih React „hooks“, odnosno udica. Ispostavilo se da su najkorisnije udice upravo useState i useEffect udice.

useState se tipično koristi tako da se definira varijabla na razini komponente te funkcija koja postavlja vrijednost varijable, čija se vrijednost može proslijediti, no bitno, može se proslijediti i funkcija za postavljanje vrijednosti, čime komponente dublje u hijerarhiji mogu mijenjati vrijednost varijable. Nadalje, korištenje useState udica omogućava da se dio web-stranice gdje se „stateful“ varijabla koristi iznova generira pri promjeni vrijednosti te varijable.

useEffect udica se koristi za definiranje „callback“ procedura koje se obavljaju pri promjeni neke (ili više) varijabli sadržanih u „dependency array-u“ (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.

### 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 je značajno smanjilo broj grešaka pri testiranju aplikacije i uštedilo vremena na debuggiranje.

Statičko tipiziranje ispostavilo se korisno pri definiranju sučelja „Props“ u svakoj React komponenti, gdje se definira koje parametre ta komponenta zahtjeva od roditelja te kojeg su tipa. TypeScript također omogućava definiranje vlastitih tipova, što sam iskoristio za definiranje oblika objekata koje će korisnička strana zaprimiti pri pozivu poslužitelja, time omogućavajući lakšu integraciju s poslužiteljem.

## Backend

### Spring Boot

Za backend dio aplikacije korišten je Spring Boot, koji je dio ekosustava Spring Frameworka. Spring Boot pojednostavljuje razvoj, testiranje i implementaciju robusnih i skalabilnih backend aplikacija u Javi. Ova tehnologija pruža niz značajki koje ubrzavaju razvoj aplikacije, kao što su automatska konfiguracija i ugrađeni poslužitelji kao što su Tomcat i Jetty.

Spring je bio prirodan odabir zbog svoje široke podržanosti, robusne dokumentacija i izvedbe, lakoće korištenja, performansi te mnogih drugih poželjnih značajki.

Osim samog Spring Boota, jedan od glavnih dijelova koji sam koristio je upravo Spring Security framework, koji omogućava relativno lagano postavljanje sigurnosne infrastrukture za web-aplikaciju, kao što su filteri te uloge. Kao komplementarnu tehnologiju Spring Security-ju uveo sam JSON web token (JWT), koji se koristi u kolačićima za autentifikaciju i autorizaciju korisnika.

### SocketIO

Osim samog Spring frameworka, koristio sam i implementaciju WebSocketa u Javi, odnosno SocketIO. SocketIO se pokreće kao zasebna komponenta backenda na različitom portu od glavnog servera, te služu 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 uđu u istu sobu te šalju poruke na SocketIO server, koji prosljeđuje te poruke svim ostalim korisnicima, time ostvarujući komunikaciju i kolaboraciju u stvarnom vremenu.

## Baza podataka

### 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. Time se može smatrati nadogradnjom na jednostavni SQL. Baza je tijekom razvoja bila spremana lokalno te sam za pregled baze koristio alat pgAdmin 4.

# Zahtjevi sustava

## Generički zahtjevi aplikacije

* Mehanizam autentifikacije i autorizacije, temeljen na korisničkom imenu i lozinci
* Spremanje i izmjena korisničkih podataka
* Intuitivno i easy-to-use korisničko sučelje
* Odazivnost i responzivnost

## Specifični zahtjevi aplikacije

* Stvaranje i uređivanje planova putovanja
* Interaktivna karta za vizualiziranje planova
* Preporučivanje atrakcija i smještaja u okolini točaka puta
* Dijeljenje planova putovanja s zajednicom
* Real-time suradnja korisnika na jednom planu putovanja
* Vremenska prognoza za planove unutar nekog raspona datuma
* Troškovnik puta
* Bilješke za svaki dio puta

## Korisnički scenariji

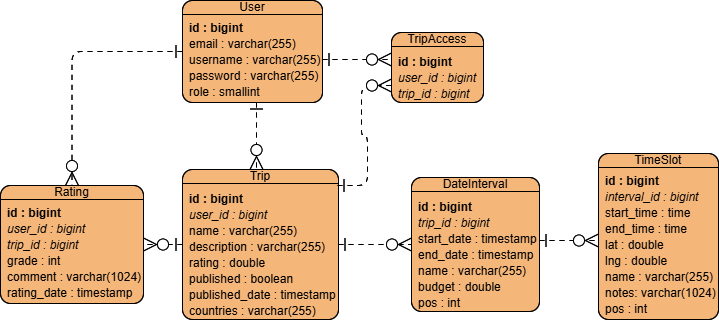
# Arhitektura sustava

## Model arhitekture sustava

Web-aplikacija se služi arhitekturom sustava klijent-poslužitelj, u kojoj korisnikov web-preglednik šalje zahtjeve na server te prikazuje odgovarajuću web-stranicu. Ovaj model omogućuje skalabilnost, modularnost i fleksibilnost sustava. Klijentska strana je odgovorna za prezentaciju i interakciju s korisnikom, dok poslužiteljska strana rukuje logikom, pohranom podataka i integracijom s vanjskim servisima. Na ovaj način, sustav može lako rasti i prilagoditi se novim zahtjevima i funkcionalnostima.

## Oblik baze podataka

U svojem radu aplikacija se služi bazom podataka koja se nalazi na serveskoj 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

### 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 ocjenjenog plana putovanja
* grade – Ocjena dana planu putovanja
* comment – Komentar uz ocjenu
* rating\_date – Datum ocjenjivanja

## Životni vijek zahtjeva

HTTP zahtjevi koji se generiraju kod klijenta šalju se na server, gdje ih zaprima neki od RESTful upravitelja (engl. controller) na jednoj od definiranih pristupnih točaka (engl. endpoint). API kojim se aplikacija koristi je interan i time poznat samo razvijatelju aplikacije. Zahtjevi uglavnom sadržavaju i JSON web token kojim korisnik dokazuje da ima autorizaciju da zahtjeva taj sadržaj, no postoje i pristupne točke koje su dostupne svima, neovisno o web tokenu. Nakon što upravitelj kroz servise i repozitorije izvrši svoju logiku, klijentu se vraća odgovor s HTTP status kodom i potencijalno s JSON objektom ako je klijent zatražio nekakve podatke.

# Implementacija sustava

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

## Struktura aplikacije

Aplikacija se strukturno dijeli na frontend i backend, gdje je svaki dio sadržan u svojoj mapi.

Struktura frontenda propisana je React knjižicom, te je za inicijalno postavljanje projekta korišten upravo Vite. Vite je lightweight frontend alat za razvoj korisničkih sučelja koji među ostalim omogućava hot reload frontenda, potporu za TypeScript i JavaScript te internalni routing zahtjeva na zadani backend URI.

Backend se sastoji od nekoliko upravljača, servise i repozitorija te sigurnosne konfiguracije, sve propisano Spring frameworkom. Kompozitne Spring anotacije ekstenzivno se koriste kako bi se olakšalo anotiranje dijelova programa, te im se time i pridodalo odgovarajuću ulogu.

## Implementacija korisničke strane

Frontend struktura sastoji se od nekoliko bitnih mapa i konfiguracijskih datoteka, među kojima su najbitnije upravo:

* package.json – Koristi se za upravljanje ovisnostima projekta te za lagano ažuriranje i skidanje ovisnosti pri razvoju na drugom računalu, definira sve pakete koje projekt treba kako bi mogao funkcionirati
* vite.config.ts – Konfiguracijska datoteka za Vite knjižicu, sadrži razne opcije za postavljanje i upravljanje projektom, primarno sam ju koristio kako bih postavio proxy URL za slanje HTTP zahtjeva na backend server.
* src mapa – Sav TypeScript kod sadržan je u src mapi. Od bitnih datoteka valja istaknuti main.tsx i App.tsx, dvije datoteke propisane React knjižicom koje sadrže vršne React komponente za renderiranje te URL rute pridijeljene stranicama na koje vode, respektivno. U components mapi nalaze se još dodatno podijeljene datoteke koje sadrže izvorni kod svih komponenti koje se koriste u aplikaciji.
* pages mapa – Sadrži sve konkretne izvedbe prikaza web-stranica s kojima se korisnik susreće pri korištenju web-aplikacije. Npr. home page, login page, itd.
* assets mapa – Sadrži sve statičke resurse koji nisu izvorni kod, kao što su slike, ikonice za vremesku prognozu, polje distinktnih boja, itd.

React komponente glavni su dio implementacije frontenda te sadrže sav TypeScript kod koji u biti iscrtava stranicu.

## Implementacija serverske strane

Struktura backenda podijeljena je na nekoliko mapa koje sadržavaju glavne komponente Spring frameworka koje omogućavaju izvršavanje business logike i zaprimanje HTTP zahtjeva.

Glavni dio backend implementacije koji se bavi HTTP zahtjevima su upravo REST controlleri. U izvornom kodu controllera definirane su pristupne točke programa na kojima se zaprimaju zahtjevi unaprijed definirane HTTP metode (GET, POST, PUT, DELETE). U aplikaciji postoji nekoliko kontrolera koji izvršavaju business logiku:

* UserController – koristi se za dohvaćanje informacija o korisnicima te upravljanje korisničkim podacima
* AdminController – koristi se za dohvaćanje podataka potrebnih za admin panel
* AuthenticationController – koristi se za izvedbu registracije i prijave u sustav, te validaciju JSON web tokena zaprimljenog od korisnika
* TripController – najopširniji kontroler, koristi se za upravljanje planom putovanja, ažuriranje općih podataka o planu, pridjeljivanje pristupa planu putovanja te dohvaćanje i pretraživanje planova putovanja po određenim kriterijima
* DateIntervalController – koristi se za upravljanje datumskim intervalima (sekcijama plana putovanja)
* TimeSlotController – koristi se za upravljanje terminima
* RatingController – koristi se za upravljanje ocjenama

Osim samih kontrolera, backend sadrži istoimene servise koji sadrže nižu razinu implementacije korisničke logike, te repozitorije kojima Spring framework komunicira s bazom podataka.

Od ostalih komponenti backenda, možemo naglasiti DTO (dana transfer object) implementacije. Ovi objekti služe kako bi se omogućila laka komunikacija i razmjena podataka između frontenda i backenda, koristeći unaprijed definirane formate objekata. Time i frontend i backend znaju kreirati i parsirati takve objekte.

SocketIO server implementiran je korištenjem paketa com.corundumstudio.socketio, koji sadrži baznu implementaciju SocketIOServer, te promatrače poput ConnectListener, DataListener i DisconnectListener kojima se definira ponašanje servera pri zaprimanju zahtjeva na svoj port. Ovaj server mora se pokrenuti na portu različitom od same Spring aplikacije jer funkcionira kao odvojeni proces, no postoje načini da se konfigurira neka vrsta port forwardinga kako bi cijela aplikacija radila na istom portu, no ja se time nisam pozabavio.

## Integracija s bazom podataka

Integracija s lokalnom bazom podataka provedena je integriranim Spring framework rješenjima, odnosno korištenjem @Repository anotacije s JpaRepository sučeljima te @Entity anotacije s klasama koje predstavljaju konkretne tablice u bazi podataka. Svaki entity ima definiran primarni ključ te veze s ostalim entityjima putem anotacija @ManyToOne i @OneToMany.

Spring u sklopu svoj starter-web paketa koristi Hibernate kao sučelje između aplikacije i baze podataka.

JpaRepository interface nudi mogućnost pisanja SQL upita na način da se ime metode findBy… nadopunjuje uvjetima te može ovisiti i parametrima predanima funkciji, što efektivno eliminira potrebu za ručnim pisanjem SQL upita. Osim tih korisničkih metoda, postoje i osnovne metode poput save, delete, findById, koje su manje-više samorazumljive.

Za spajanje na bazu podataka definira se spring.datasource.url, spring.datasource.username i spring.datasource.password u resources/application.properties konfiguracijskoj datoteci. Također se može definirati i SQL jezik, koji je defaultno postavljen na postgreSQL zbog Hibernate implementacije.

## Implementacija sigurnosnih značajki

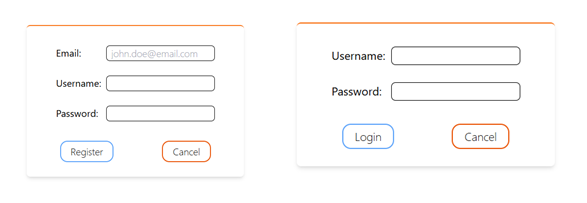
Sigurnost sustava počiva na JSON web tokenu te SecurityConfig i JWTAuthenticationFilter datotekama. SecurityConfig datoteka služi za definiranje lanca filtera kroz koje svaki zaprimljeni zahtjev mora proći prije no što ga zaprimi neki od kontrolera. Tri glavne definirane rute u aplikaciji su /api/auth/, /api/admin te /api/core. Pristup administratorskoj ruti zahtjeva upravo ulogu administratora, dok su ostale dvije rute otvorene bilo kojem prijavljenom korisniku.

JWTAuthenticationFilter je upravo taj filter kroz koji se šalju zahtjevi, te se u njemu provjerava polje "Authorization“ HTTP zaglavlja. Ukoliko je token validiran, korisnika se propušta; u suprotnom korisniku se vraća 403 Forbidden statusni kod.

# Korištenje sustava

U ovom poglavlju provest ću vas kroz korištenje aplikacije.

## Autentifikacija

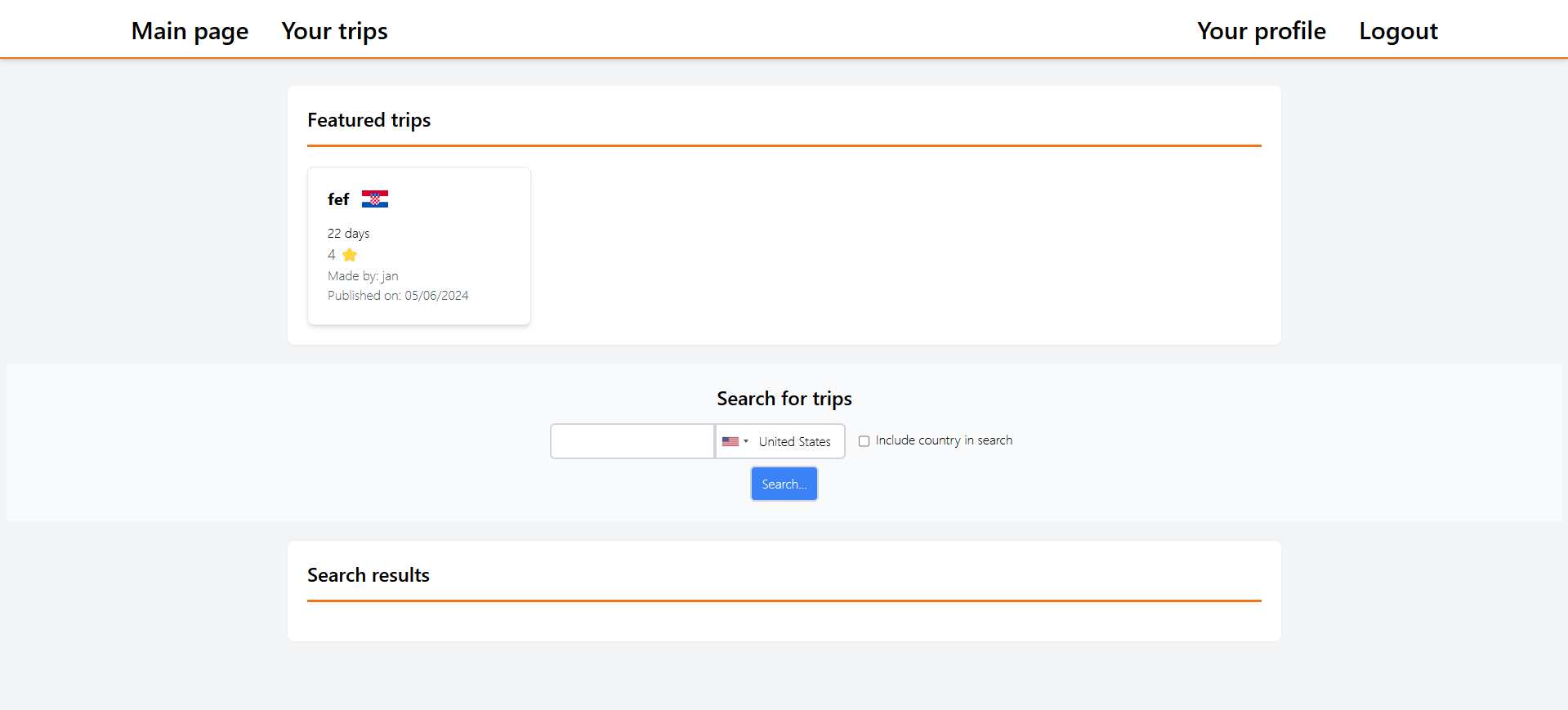


Slika 5.1 Forma za registraciju i prijavu u sustav

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.

## Početna stranica



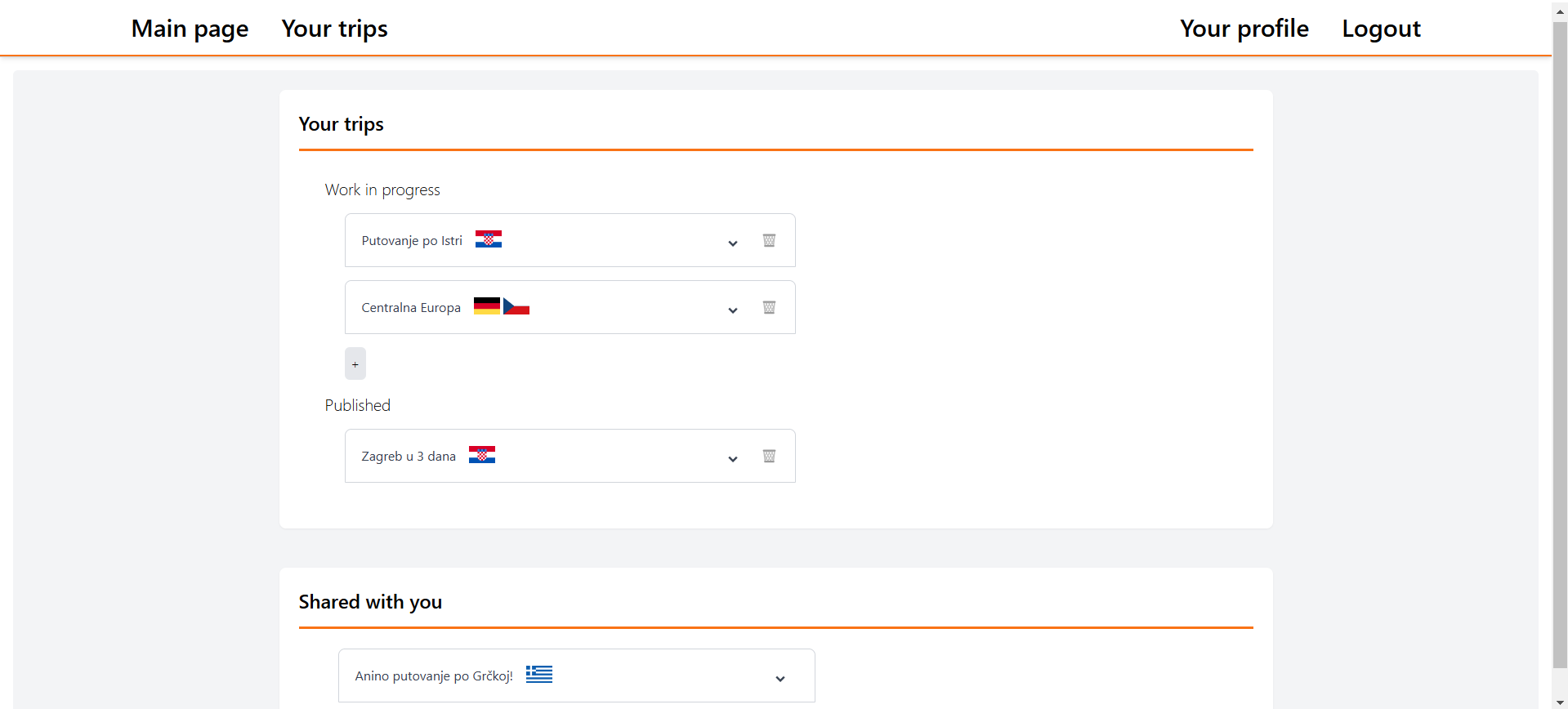
Slika 5.2 Početna stranica

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 putovanje, 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 ocjenjenih planova prikaže u svojim kućicama.

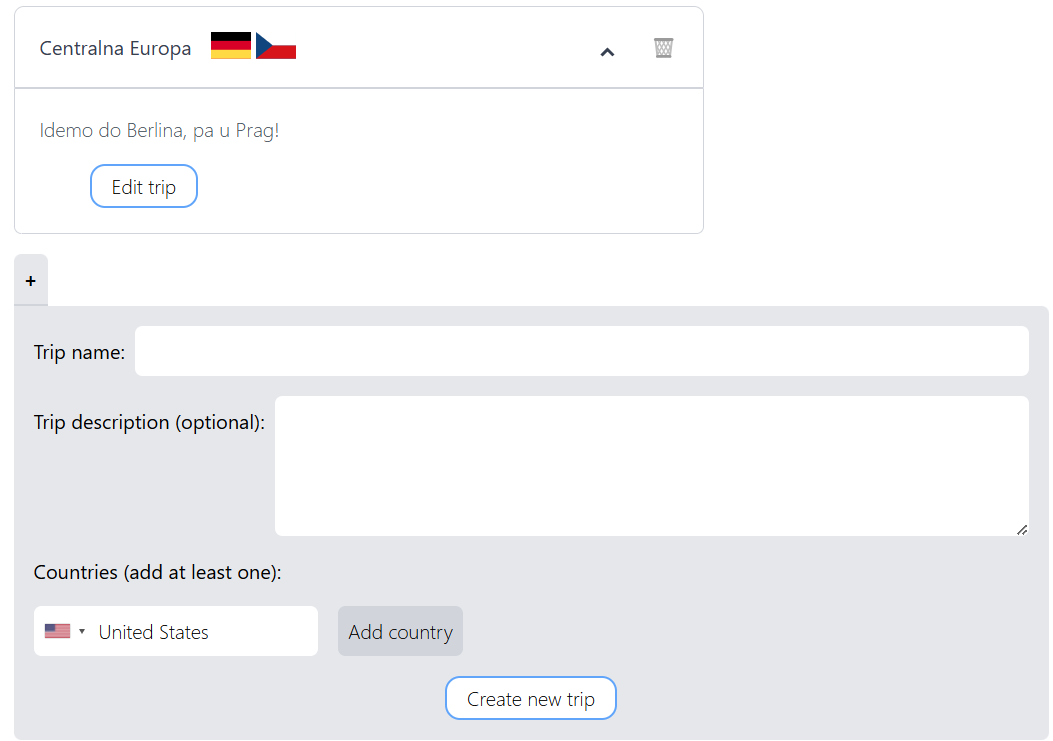
Traka za pretraživanje ima dva polja, polje za upis dijela imena plana putovanja, te polje za odabir jedne od zemalja plana putovanja, s dodatnom kućicom koja se označi ako se odabrana zemlja želi uzeti u obzir pri pretraživanju. Rezultati pretraživanja prikažu se u istom formatu kao i predloženi planovi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

## Korisnikovi planovi putovanja



Slika 5.3 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 5.4 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“.

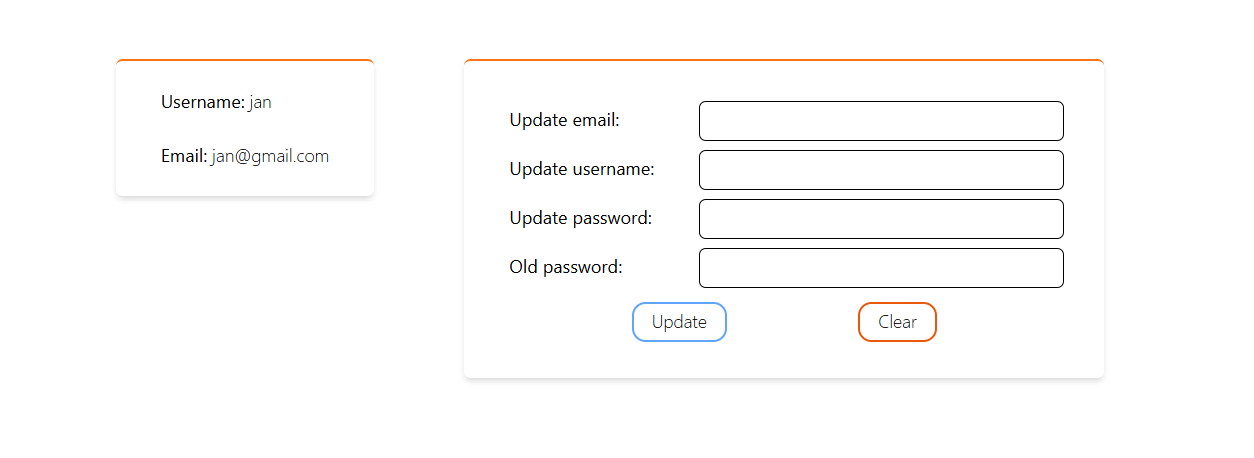
Klikom na plusić, otvara se forma za stvaranje novog plana putovanja. Forma od korisnika zahtjeva da obavezno unese ime plana putovanja, te barem jednu zemlju na koju se plan putovanja odnosi, dok je opis putovanja opcionalan i ne mora odmah biti specificiran. Klikom na gumb novi plan putovanja je stvoren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

## Korisnikov profil



Slika 5.5 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.

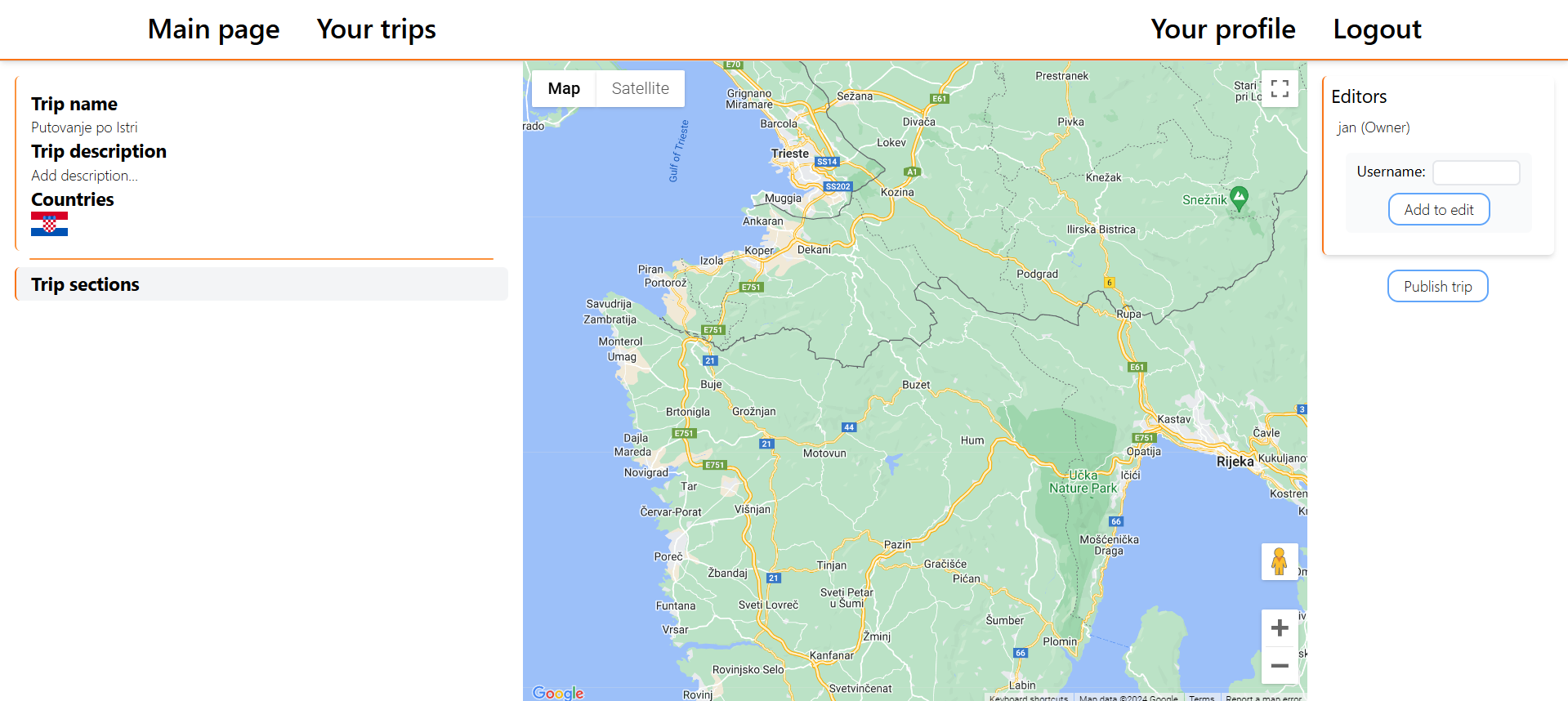


Slika 5.6 Forma za uređivanje korisničkog profila

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.

## 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 5.7 Cijela stranica za uređivanje plana putovanja

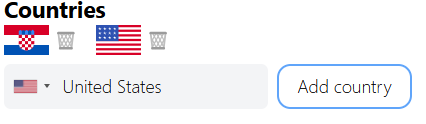
### Osnovni detalji plana putovanja



Slika 5.8 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.

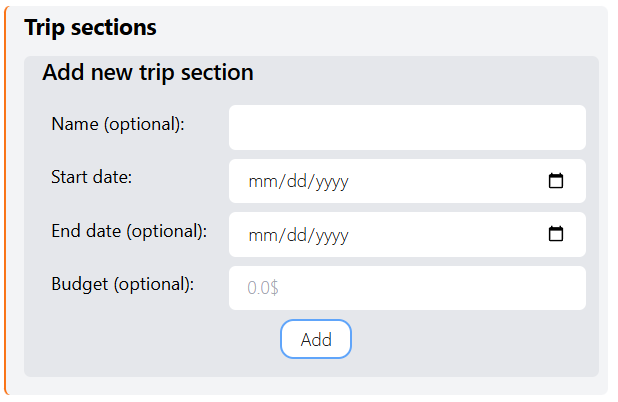
Ovaj postupak je standardan na razini cijele stranice te se manje-više svaka vrijednost može naknadno ažurirati na ovaj način te postoji indikacija da se klikom na polje ono može mijenjati.



Slika 5.9 Uređivanje zemalja putovanja

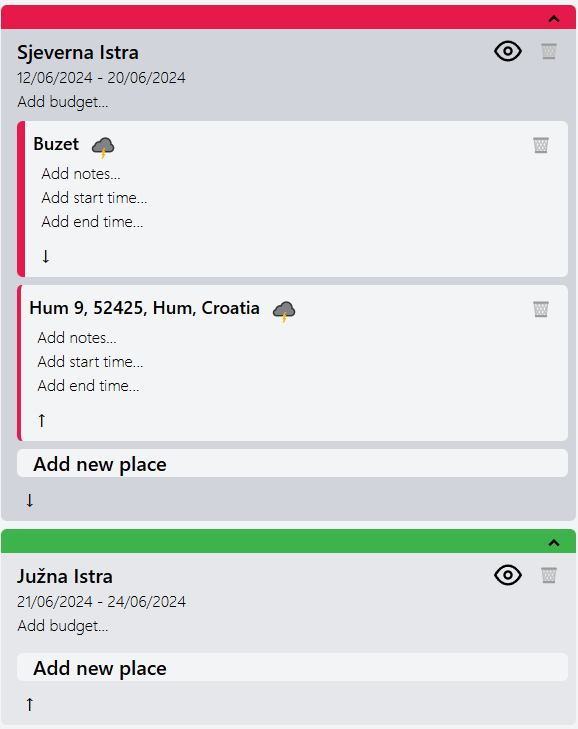
Klikom na polje sa zastavama zemalja, prikaže se sljedeća jednostavna forma. Kako je uvijek potrebno da je definirana barem jedna zemlja u putovanju, ikonice koša za smeće kojima se zemlje uklanjaju su prisutne jedino ako je definirano više od jedne zemlje u planu. Klikom na gumb „Add country“, odabrana 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.

### Sekcije plana, mjesta u putovanju



Slika 5.10 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. Ukoliko sekciji nije pridodan krajnji datum, podrazumijeva se da ona traje samo jedan dan. Kućica za dodanu sekciju nositi 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 5.11 Prikaz sekcija plana

Svakoj sekciji pridijeljena je distinktna boja kojom je obojano 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.

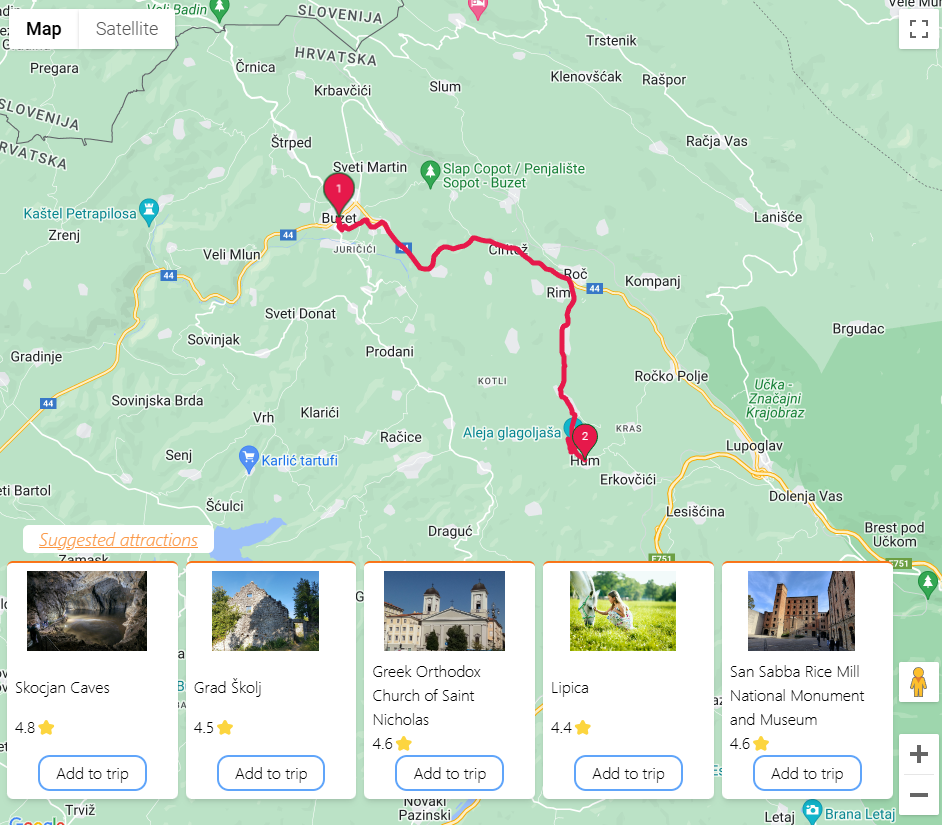
Kada je novo mjesto dodano, ono se pojavi iznad forme za stvaranje novog mjesta. Svi parametri su naknadno izmjenjivi. Ukoliko je datum početka sekcije plana u narednih 14 dana od trenutnog datuma, aplikacija će automatski zatražiti vremensku prognozu te prikazati ikonicu predviđenih vremenskih prilika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

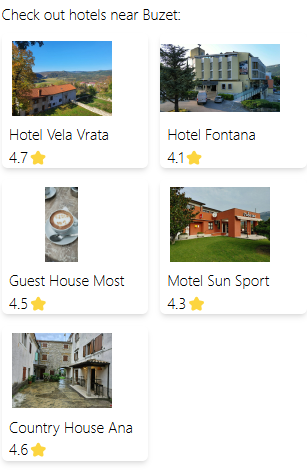
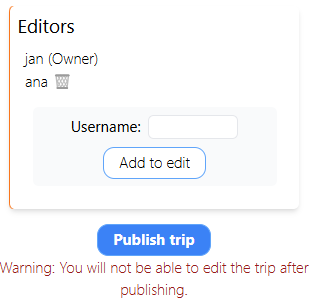
### Interaktivna karta, prijedlozi atrakcija



Slika 5.12 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. Markeri su numerirani redom kojim su mjesta navedena u sekciji plana. Nadalje, markeri se mogu vući po karti, čime se nanovo izračuna i iscrta 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 Google-u.

### Ostali sudionici, objava plana, prijedlozi hotela



Slika 5.13 Desni dio stranice

Na desnoj strani stranice nalazi se dio za uređivanje sudionika. Ovdje se 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 pomoću klika na gumb „Publish trip“, čime se plan više ne može uređivati. Ukoliko je neko mjesto selektirano, ispod gumba za objavu može se nekoliko kartica sa 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.

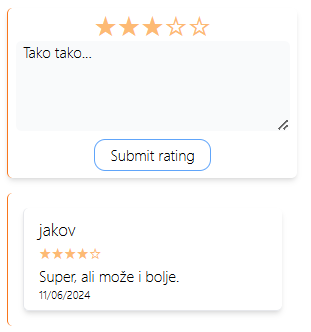
### Pregled objavljenog plana putovanja

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 ukoliko 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 5.14 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.

# 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

# Sažetak

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-aplikacije, interaktivna karta, suradnja korisnika, itinerar, vremenska prognoza, React, Spring Boot, TypeScript, Vite, SocketIO, Google Maps

# Summary

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 applications, interactive map, user collaboration, itinerary, weather forecast, React, Spring Boot, TypeScript, Vite, SocketIO, Google Maps