Izrada društvene mreže

Vratarić, Marina

SVEUČILIŠTE JOSIPA JURJA STORSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni diplomski studij

IZRADA I FUNKCIONALNO TESTIRANJE WEB APLIKACIJE ZA VOĐENJE DRUŠTVENE MREŽE

Diplomski rad

Marina Vratarić

Osijek, 2020.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 11.09.2020.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime studenta:	Marina Vratarić
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1024R, 22.09.2019.
OIB studenta:	97169650888
Mentor:	Izv. prof. dr. sc. Ivica Lukić
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Doc.dr.sc. Mirko Köhler
Član Povjerenstva 1:	Izv. prof. dr. sc. Ivica Lukić
Član Povjerenstva 2:	Doc.dr.sc. Zdravko Krpić
Naslov diplomskog rada:	Izrada društvene mreže
Znanstvena grana rada:	**Informacijski sustavi (zn. polje računarstvo)**
Zadatak diplomskog rada:	Potrebno je napraviti web aplikaciju koja će omogućiti vođenje društvene mreže. Potrebno je provesti različite vrste testova koje će potvrditi ispravan rad aplikacije. Tema je rezervirana za studenta: Marina Vratarić
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	11.09.2020.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 30.09.2020.

Ime i prezime studenta:	Marina Vratarić
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1024R, 22.09.2019.
Turnitin podudaranje [%]:	5

Ovom izjavom izjavljujem da je rad pod nazivom**: Izrada društvene mreže**

izrađen pod vodstvom mentora Izv. prof. dr. sc. Ivica Lukić

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA

o odobrenju za pohranu i objavu ocjenskog rada

kojom ja __Marina Vratarić_____, OIB: __97169650888_____, student/ica Fakulteta elektrotehnike, računarstva i informacijskih tehnologija Osijek na studiju _Diplomski sveučilišni studij Računarstvo _, kao autor/ica ocjenskog rada pod naslovom: _Izrada društvene mreže**,** dajem odobrenje da se, bez naknade, trajno pohrani moj ocjenski rad u javno dostupnom digitalnom repozitoriju ustanove Fakulteta elektrotehnike, računarstva i informacijskih tehnologija Osijek i Sveučilišta te u javnoj internetskoj bazi radova Nacionalne i sveučilišne knjižnice u Zagrebu, sukladno obvezi iz odredbe članka 83. stavka 11. *Zakona o znanstvenoj djelatnosti i visokom obrazovanju* (NN 123/03, 198/03, 105/04, 174/04, 02/07, 46/07, 45/09, 63/11, 94/13, 139/13, 101/14, 60/15).

Potvrđujem da je za pohranu dostavljena završna verzija obranjenog i dovršenog ocjenskog rada. Ovom izjavom, kao autor/ica ocjenskog rada dajem odobrenje i da se moj ocjenski rad, bez naknade, trajno javno objavi i besplatno učini dostupnim:

a) široj javnosti

b) studentima/icama i djelatnicima/ama ustanove

c) široj javnosti, ali nakon proteka 6 / 12 / 24 mjeseci (zaokružite odgovarajući broj mjeseci).

**U slučaju potrebe dodatnog ograničavanja pristupa Vašem ocjenskom radu, podnosi se obrazloženi zahtjev nadležnom tijelu Ustanove.*

__Osijek, 30.09.2020._____ _____

(mjesto i datum) (vlastoručni potpis studenta/ice)

Sadržaj

1. UVOD

Društvene mreže se danas koriste u svakodnevnoj uporabi, bilo na mobilnim uređajima ili računalima. Pojavljuju se po prvi puta devedesetih godina prošlog stoljeća. Neizostavan su dio današnjice i kao takve imaju veliku ulogu u zbližavanju ljudi, komuniciranju, dijeljenju znanja, educiranju, širenju vijesti čak i u istraživanju. Vrlo se često koriste u digitalnom marketingu. Najbrža je rastuća industrija u svijetu. Za primjer se može uzeti društvena mreža *Facebook* koja ima 2.2 milijarde aktivnih korisnika, zatim *Instagram* koji ima oko 1.1 milijardu korisnika i *Twitter* kao treća najpopularnija društvena mreža.

Cilj ovog diplomskog rada je opisati mogućnosti primjene društvenih mreža, a uz to cilj je i izraditi web aplikaciju koja će korisniku omogućiti korištenje društvene mreže u kojoj će moći pisati svoje objave, komentare, dijeliti mišljenja, ažurirati profil, pratiti druge korisnike, pretraživati druge korisnike i brojne druge stvari. Također, cilj je i napraviti testni dio kojim će se moći vidjeti ispravan rad aplikacije.

U drugom poglavlju detaljno je opisan teorijski dio ovog rada zajedno sa svim programskim jezicima i okruženjima koja se koriste za izradu web aplikacije. Treće poglavlje ovog rada odnosi se na programsko rješenje, odnosno praktičnu primjenu web aplikacije prilikom korištenja. U četvrtom poglavlju opisano je testiranje aplikacije kao i njen izgled. Posljednje poglavlje sadržava zaključak ovog diplomskog rada.

1.1. Zadatak diplomskog rada

Potrebno je napraviti web aplikaciju koja će omogućiti vođenje društvene mreže. Potrebno je provesti različite vrste testova koje će potvrditi ispravan rad aplikacije.

2. POSTOJEĆA RJEŠENJA

Danas se za izradu i vođenje društvenih mreža koristi niz različitih programskih jezika u zavisnosti od toga koje funkcionalnosti posjeduju, bilo to učitavanje videa, slika, funkcija chat-a i slično. Ono što je karakteristično za gotovo sve društvene mreže jest to da se za *front end* koristi JavaScript jezik, dok je broj jezika korištenih za *back end* puno veći i raznolikiji. React je trenutno najpopularnija JavaScript biblioteka. Neke od popularnih društvenih mreža koje koriste React su: Facebook, Instagram, Reddit, Imgur. Na grafu 2.1. vidljive su najpopularnije društvene mreže u svijetu. Na grafu je prikazan broj aktivnih korisnika izražen u milijunima, iz čega se da zaključiti da su društvene mreže skoro pa neizostavni dio svakidašnjice. Društvene mreže stječu sve veću popularnost jer korisnicima omogućavaju komunikaciju, širenje vijesti i znanja, zbližavanje, edukaciju.

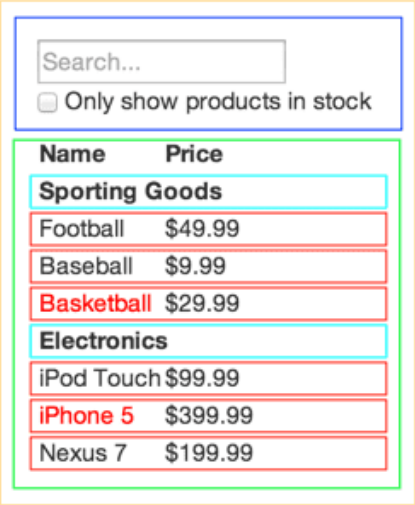
Slika 2.1. Graf najpopularnijih društvenih mreža

3. PREGLED KORIŠTENIH TEHNOLOGIJA I ALATA

Tehnologije i alati koji su primijenjeni u ovom diplomskom radu prilikom izrade web aplikacije opisani su u nastavku. Za *front end* korištena je Javascript biblioteka React, te HTML i CSS vezan za sadržaj i izgled istog, dok je za *back end* primijenjena Node.js okolina i Express okvir koji služe za postavljanje servera i za povezivanje s bazom podataka, a za bazu podataka rabljena je MongoDB baza podataka. Za potrebe testiranja *back end* dijela korišten je Postman, a sav kod pisan je u Visual studio code-u.

3.1. React

React je Javascript biblioteka koja služi za izgradnju korisničkog sučelja. Kreirana je od strane Facebooka. Prema [1], *Document Object Model*, u nastavku DOM, programski je API za HTML i XML dokumente koji definira logičku strukturu dokumenta i način na koji je taj dokument upotrebljen, korišten i pristupljen. Ta logička struktura je vrlo slična prikazu stabla. U React-u, kod se sastoji od manjih cjelina koje se zovu komponente. Obzirom na to da je manipulacija DOM elementima dosta spora, manipulacija React komponenta vrlo je brza te se koristi metoda *render* koja omogućava pretvaranje tih komponenti u DOM kako bi se rezultat mogao vidjeti i prikazati u internetskom izborniku. React služi za kreiranje različitih vrsta komponenata poput gumba, potvrdnog okvira (engl. *checkbox*), padajućeg izbornika, zatim za kreiranje navigacijskih komponenata ili komponenata koje daju nekakvu informaciju kao što je grafički indikator (engl. *progress bar*) ili kao što su obavijesti. Cijelo korisničko sučelje se može razbiti na manje dijelove, odnosno komponente koje se mogu koristiti višestruko puta u kodu, ali bez ponovnog pisanja iste logike. To je i sama srž React-a. Na slici 2.1. prikazan je primjer kako se čitavo sučelje može podijeliti na manje komponente.



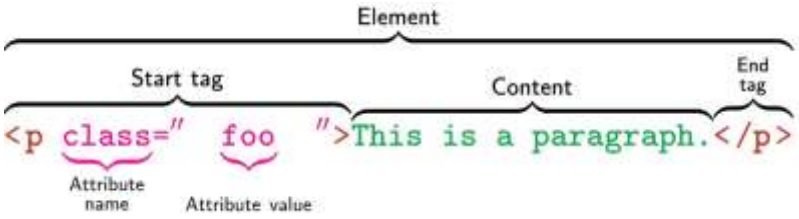
Slika 3.1. Prikaz sučelja i komponenti, [2]

Na slici 3.1. vidi se 5 komponenti koje su grupirane u drugačijim bojama jer predstavljaju jednu određenu grupu. Na primjer, narančasta predstavlja ukupni primjer, plava svaki korisnički unos, zelena prikazuje i filtrira podatke ovisno o korisnikovom unosu, tirkizna boja predstavlja naziv za svaku kategoriju, a crvena označava red svakog proizvoda.

3.2. HTML

HTML dolazi od skraćenice za *Hypertext Markup Language*, a predstavlja označni jezik za dokumente. *Hypertext* označava poveznice (engl. *links*) koje povezuju web stranice s drugim web stranicama, dok riječ *markup* služi kako bi se protumačio sadržaj web stranice. HTML dokument se prikazuje u internetskim preglednicima u kojima se može vidjeti tekst, slike i brojni drugi elementi stranice, odnosno njezin sadržaj, a sastoji se od HTML elemenata koji su zapravo osnovni gradivni elementi HTML stranice.

Ti gradivni elementi se prikazuju pomoću oznaka (engl. *tag*) koji sadržava naziv i par izlomljenih zagradi <, >. Pa tako za paragraf postoji početna oznaka koja mora imati i svoju zatvarajuću oznaku , a između se nalazi sadržaj. Unutar početne oznake mogu postoji i razni atributi poput klase koji idu u paru kao naziv atributa i vrijednost atributa pod navodnicima. Još neki od elemenata su: <head>, <ul>, <li>, <titile>, <img>, <audio>, <nav>, <header>, <footer>, <h3>. Postoje i prazni elementi poput
 kojim se označava prijelom retka i koji ne sadrži nikakav sadržaj. Na slici 3.2. prikazan je primjer jednog HTML elementa.



Slika 3.2. Primjer sintakse HTML elementa, [3]

U nastavku je dan primjer vrlo jednostavnog HTML koda s naslovom i kratkim paragrafom, te slikom 3.3. kako to bi taj kod zaista izgledao u internetskom pregledniku.

```
<!DOCTYPE html>
<html>
<body>
<h2 class=“title“>This is the title</h2>
<p class="content">This is some text with smaller font then title.</p>
</body>
</html>
```

This is the title

This is some text with smaller font then title.

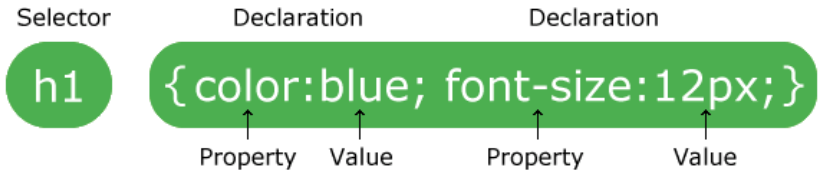
Slika 3.3. Prikaz HTML koda u internetskom pregledniku

Prema [4], prvu verziju HTML-a napisao je Tim Berners-Lee 1991. godine, a nakon toga su se razvijale broje druge inačice HTML-a. Najkorištenija verzija bila je 4.01, na čemu je i danas izgrađena većina web stranica uz kombinaciju XHTML 1.0. Zadnja verzija je HTML5 razvijena 2008. godine i podržava ju većina modernih internetskih preglednika.

3.3. CSS

Cascading Style Sheets, u nastavku CSS, je stilski jezik za opisivanje dokumenta kao što je HTML. Kao što i sam naziv govori, služi za dodavanje stila svim HTML elementima poput boje, vrste i veličine slova (eng. *font*), pozicije elemenata na stranici kao i brojne druge mogućnosti dizajna. Zapisuje se pomoću izbornika (engl. *selector*) kojim se bira HTML element koji se želi dizajnirati, te

od para vitičastih zagrada unutar kojih se zapisuje jedan ili više parova svojstvo:vrijednost (engl. *property:value*) odvojenih točkom sa zarezom. Kao primjer može se navesti sljedeće: za naslov veličine 3 (h3 izbornik) svojstvo može biti boja, a vrijednost može biti crvena, pa se zapisuje na sljedeći način: h3 { color: red; }. Na slici 3.4. prikazan je primjer sintakse CSS-a s višestrukim parovima. Prednost korištenja CSS-a je ta što se smanjuje potreba za dupliciranjem koda.



Slika 3.4. CSS sintaksa, [5]

3.4. Reactstrap

Prema [6], Reactstrap je biblioteka komponenata za React. Biblioteka sadržava React Bootstrap 4 komponente koje su kompatibilne s React-om, odnosno sadržava ugrađene Bootstrap komponente zbog kojih je olakšan način kreiranja korisničkog sučelja. Pruža veliku fleksibilnost i omogućava da se na brži i jednostavniji način kreiraju forme i elementi. Prednost korištenja ovog alata je ta da već postoje gotovi predlošci s definiranim stilom zbog kojeg se brže i lakše dizajniraju elementi. Na slici 3.5. prikazani su različiti gumbovi koji su već postojeći u predlošku, a jednostavno se mogu koristiti i mijenjati boje, veličinu slova i slično.

Slika 3.5. Reactstrap gumbovi, [7]

Linijom koda u nastavku može se dobiti crveni gumb ako specificiramo vrijednost boje kao 'danger': <Button color="danger">danger</Button>{' '}

3.5. Node.js

Prema [8], NodeJS nije novi jezik niti samo programski okvir, već se može smatrati izvršnom okolinom (engl. *runtime environment*) za Javascript koji je građen na Chrome V8 Javascript mašini (engl. *engine*). Ono što radi jest izvršavanje koda izvan internetskog preglednika. Razvio ga je Ryan Dahl 2009. godine. Jedan od najmoćnijih posebnosti NodeJS-a je njegovo neblokirajuće, asinkrono

izvršavanje, a za razliku od drugih jezika kao što je PHP, ASP.NET, Ruby, ima samo jednu nit na kojoj obavlja asinkrone procese kako bi omogućio još veću skalabilnost i performanse za aplikacije koje su opterećene većim web prometom. Napisan je u Javascript programskom jeziku što znači manje utrošenog vremena za učenje nove sintakse ili drugog programskog jezika.

3.6. Express okvir

Prema [9], Express je popularni web okvir (engl. *framework*) pisan u Javascript programskom jeziku i koji je udomaćen (engl. *hosted*) u Node.js izvršnu okolinu (engl. *runtime environment*). Express je objavljen 2010. godine, a trenutna inačica je 4.17.1. Pruža metode kako bi specificirao funkcije koje su pozvane za određeni HTTP glagol (engl. *verb*) kao što je PUT, GET, POST, DELETE, SET, zatim metode koje specificiraju uporabu određene mašine (engl. *engine*) i lokaciju datoteka te predložak za prikazivanje (engl. *render*) odgovora (engl. *response*). Pruža snaži set značajki i karakteristika za web i mobilne aplikacije.

3.7. MongoDB

MongoDB je baza podataka izgrađena za moderne aplikacije koja sprema podatke u fleksibilne, JSON oblik dokumente. Prema [10], model dokumenta mapira objekte u aplikacijski kod kako bi se s podacima lakše manipuliralo i kako bi se lakše s tim podacima radilo. Bogati je jezik upita (engl. *query language*) koji omogućava filtriranje, sortiranje po bilo kojem polju, bez obzira na ugniježđenost (engl. *nest*). Besplatna je baza podataka i otvorenog je koda (engl. *open-source*). Objavljena je po prvi puta 2009. godine, a pisana je u C++, Go, Javascript i Python programskom jeziku. Zadnja stabilna verzija je 4.2.6.

3.8. Visual studio code

Prema [11], Visual studio code besplatni je uređivač za pisanje koda (engl. *code editor*) kreiran od strane Microsoft-a za Windows, Linux i macOS operacijske sustave. Sadržava razne i brojne karakteristike poput prikaza teksta u različitim bojama i stilom slova (engl. *font*) koje označavaju različitu kategoriju kako bi istaknuo riječi radi bržeg uočavanja (engl. *syntax highlighting*), zatim podržava proces pronalaska i rješavanja problema, nedostataka ili grešaka unutar programskog koda koji sprečavaju ispravan rad (engl. *debugging*), ugrađeni Git i mnoge druge značajke. To je jedan od najpopularnijih alata za razvoj (engl. *developer tool*). Objavljen je 2015. godine. Daje podršku brojim programskim jezicima kao što su Javascript, Java, Node.js, C++, Go.

3.9. Postman

Postman je razvojni alat koji omogućava testiranje *Application Programming Interface*, u nastavku API, poziva (engl. *API calls*). Radi po principu unosa podataka koji će biti poslani određenoj web adresi, a na kraju će se ta informacija vratiti i prikazati korisniku kao uspješan ili neuspješan odgovor. Objavljen je 2012. godine. Postman također omogućava korisniku kreiranje kolekcija za API pozive radi jednostavnosti u organizaciji. Odgovor na svaki zahtjev je poslan u JSON formatu. API pozivi će odgovoriti s prikladnim HTTP status kodom. Status 200 OK označava da je sve prošlo u redu, dok 4XX ili 5XX označavaju da je u pitanju nekakva greška (engl. *error*).

3.10. Typescript

Typescript je programski jezik koji se koristi za razvijanje Javascript aplikacija. Prema [12], to je zapravo Javascript jezik koji ima dodatne značajke, odnosno nadskup je Javascript jezika koji je fokusiran na kod koji je predvidljiv. Glavna značajka je statičko pisanje (engl. *static* typing). Kod pisan u Typescript-u pretvoren je (engl. *complied*) u običan Javascript jezik, jer Typescript nije razumljiv internetskim preglednicima. Pretvaranje Javascript jezika u Typescript može se izvesti promjenom ekstenzije iz *.js* u *.ts*. Razlike Typescript-a u odnosu na Javascript su te što je Typescript objektno-orijentiran programski jezik, dok je Javascript skriptni jezik, zatim Typescript ima sučelje (engl. *Interface*) te podržava opcionalni parametar, za razliku od Javascript-a. Prednosti korištenja Typescript-a su razne, poput prikazivanje grešaka (engl. *errors*) za vrijeme razvijanja aplikacije, pisanje koda koji je razumljiviji zbog deklariranja tipova podataka i slično.). Razvijen je od strane Microsoft-a 2012. godine. Na slici 3.6. prikazan je primjer deklariranja sučelja (engl. *Interface*) u Typescript-u. Sučelje se zove „IPerson“ te ima samo jednog člana - „fullname“ koji je tipa string. Funkcija „displayPerson“ prima jedan argument koji je tipa navedenog definiranog sučelja.

```
interface IPerson {
    fullname: string;
}

function displayPerson(person : IPerson) {
    return "Hello, " + person.fullname;
}
```

Slika 3.6. Definiranje sučelja u Typescript-u, [13]

4. PROGRAMSKO RJEŠENJE

Cilj ovog poglavlja je izrada web aplikacije, odnosno društvene mreže koja će omogućiti korisnicima kreiranje profila, prijavu u aplikaciju, pretraživanje drugih korisnika, praćenje korisnika, pregled vlastitog profila i profila drugih korisnika, ažuriranje vlastitog profila, pisanje objava, komentara kao i brisanje istih, označavanje objave sa sviđa mi se, pregled pratitelja te odjavu iz aplikacije. Također, u ovom poglavlju opisani su koraci izrade aplikacije te ključni i važni dijelovi iste. Dana je i specifikacija svih zahtjeva koje korisnik može upotrijebiti.

4.1. Specifikacija zahtjeva i korisničkih slučajeva

U nastavku su objašnjeni svi korisnički slučajevi aplikacije kao i specifikacija zahtjeva, odnosno sve mogućnosti korištenja ove društvene mreže. U tablici 4.1. nalaze se svi korisnički slučajevi i zahtjevi, a ispod tablice su isti detaljno razrađeni i opisani.

Tablica 4.1. Popis zahtjeva i korisničkih slučajeva

Specifikacija zahtjeva	Korisnički slučaj
Korisnik se može registrirati u sustav	1
Korisnik se može prijaviti u sustav	2
Korisnik može napisati objavu	3
Korisnik može obrisati objavu	4
Korisnik može označiti objavu sa 'sviđa mi se'	5
Korisnik može komentirati objave	6
Korisnik može vidjeti svoj profil	7
Korisnik može pronaći druge korisnike	8
Korisnik može učitati profilnu sliku	9
Korisnik može urediti svoj profil	10
Korisnik se može odjaviti iz sustava	11
Korisnik može pratiti druge korisnike	12
Korisnik može obrisati komentare	13
Korisnik može označiti objavu kao privatnu ili javnu	14
Korisnik može vidjeti vrijeme kada je objava kreirana	15
Korisnik može vidjeti korisnike kojeg drugi prate	16
Korisnik može vidjeti pratitelje	17
Korisnik može pretraživati druge korisnike	18
Korisnik može vidjeti profile drugih korisnika	19

U korisničkom slučaju 1, korisnik može kreirati svoj račun na društvenoj mreži, odnosno može se registrirati u sustav.

- Prvi scenariji: Korisnik nije upisao vrijednosti u tražena polja. Klikom na gumb 'Sign Up' registracija je onemogućena, odnosno onemogućen je klik na isti gumb.
- Drugi scenariji: Korisnik je upisao sve vrijednosti u polja, međutim koristio je već postojeći email račun. Registracija je onemogućena te se prikazuje upozorenje da je račun već kreiran.
- Treći scenariji: Korisnik je upisao sve vrijednosti u polja, s ispravnim email računom no nije upisao lozinku. Pojavljuje se upozorenje. Registracija je onemogućena.

- Četvrti scenariji: Korisnik je upisao sve vrijednosti u polja, bez ikakvih upozorenja. Registracija je omogućena i korisnik je kreiran u sustavu.

U korisničkom slučaju 2, korisnik se može prijaviti u sustav ako je račun napravljen.

- Prvi scenariji: Korisnik nije upisao email niti lozinku u tražena polja. Prijava je onemogućena.
- Drugi scenariji: Korisnik je upisao email račun koji nije registriran u sustavu. Prijava je onemogućena.
- Treći scenariji: Korisnik je upisao ispravan email račun, no lozinka ne odgovara istom računu. Prijava je onemogućena.
- Četvrti scenariji: Korisnik je upisao ispravan email račun s odgovarajućom lozinkom. Prijava je omogućena.

U korisničkom slučaju 3, ukoliko je korisnik prijavljen u sustav, korisnik može napisati objavu.

U korisničkom slučaju 4, ukoliko je korisnik prijavljen u sustav i ukoliko objava postoji, korisnik može obrisati objavu.

U korisničkom slučaju 5, ukoliko je korisnik prijavljen u sustav i ukoliko objava postoji, korisnik može označiti objavu sa 'sviđa mi se' samo jednom.

U korisničkom slučaju 6, ukoliko je korisnik prijavljen u sustav i ukoliko objava postoji, korisnik može komentirati objavu.

U korisničkom slučaju 7, ukoliko je korisnik prijavljen u sustav, korisnik može vidjeti svoj profil.

U korisničkom slučaju 8, ukoliko je korisnik prijavljen u sustav, korisnik može pronaći druge korisnike na društvenoj mreži.

U korisničkom slučaju 9, ukoliko je korisnik prijavljen u sustav, korisnik može učitati fotografiju na svom profilu.

U korisničkom slučaju 10, ukoliko je korisnik prijavljen u sustav, korisnik može urediti svoj profil izmjenom imena, prezimena, opisa ili slike.

U korisničkom slučaju 11, ukoliko je korisnik prijavljen u sustav, korisnik se može odjaviti iz sustava prilikom čega iskače upozorenje je li siguran u to kako se ne bi odjavio zbog slučajnog klika na gumb 'Log out'. Drugom potvrdom se uspješno odjavljuje iz sustava.

U korisničkom slučaju 12, ukoliko je korisnik prijavljen u sustav, korisnik može pratiti druge korisnike, a također može i otpratiti iste.

U korisničkom slučaju 13, ukoliko je korisnik prijavljen u sustav, korisnik može obrisati svoj komentar u objavi ako isti postoji.

U korisničkom slučaju 14, ukoliko je korisnik prijavljen u sustav, korisnik može odabrati vidljivost svoje objave kao privatnu objavu koju mogu vidjeti korisnikovi pratitelji ili kao javnu objavu koju mogu vidjeti svi korisnici.

U korisničkom slučaju 15, ukoliko je korisnik prijavljen u sustav, korisnik može vidjeti vrijeme kada je objava kreirana na svakoj objavi.

U korisničkom slučaju 16, ukoliko je korisnik prijavljen u sustav, korisnik na svom profilu ili profilu drugog korisnika može vidjeti ostale korisnike kojeg drugi prate.

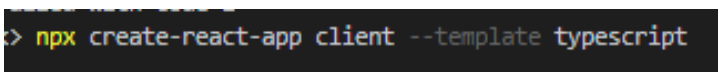
U korisničkom slučaju 17, ukoliko je korisnik prijavljen u sustav, korisnik na svom profilu ili profilu drugog korisnika može vidjeti sve pratitelje.

U korisničkom slučaju 18, ukoliko je korisnik prijavljen u sustav, korisnik može u tražilicu upisati ime ili prezime drugog korisnika kojeg želi pronaći.

U korisničkom slučaju 19, ukoliko je korisnik prijavljen u sustav, korisnik samo u slučaju praćenja drugog korisnika može otići na njegov profil.

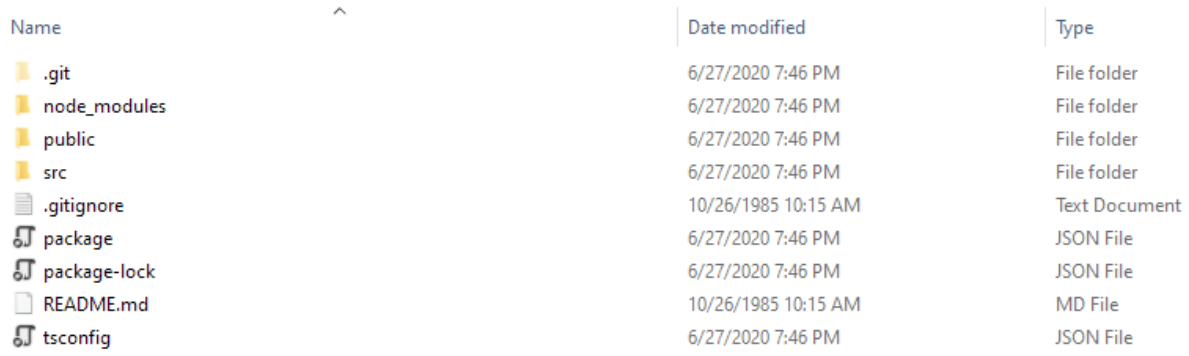
4.2. Izrada React aplikacije

React aplikacija može se kreirati koristeći vrlo jednostavnu naredbu, a to je 'npm create-react-app', nakon koje slijedi naziv. U ovom radu korišten je i Typescript koji proširuje Javascript dodavajući tipove podataka kako bi se brže uočile greške u kodu. Linija koda koja je korištena je prikazana slikom 4.1.



Slika 4.1. Naredba za kreiranje React aplikacije

Nakon toga kreiraju se sljedeće datoteke prikazane slikom 4.2.



Slika 4.2. Osnovne datoteke React aplikacije

Kako bi se aplikacija pokrenula u cijelosti, mora se pokrenuti i server skripta i klijentska skripta. Instalacijom paketa *nodemon*, server se automatski ponovo pokreće nakon spremanja datoteke i to je stavljeno u skriptu nazvanu 'server', a pokreće se naredbom *npm run server*. Klijentska skripta pokreće se naredbom *npm run client*, a kako bi se obje skripte odjednom pokrenule instaliran je paket *concurrently* koji to omogućava naredbom *npm run dev*, kao što je i prikazano slikom 4.3.

```
"scripts": {
  "start": "node server.js",
  "server": "nodemon server.js",
  "client": "npm start --prefix client",
  "dev": "concurrently \"npm run server\" \"npm run client\" "
},
```

Slika 4.3. Popis skripti za pokretanje aplikacije

4.3. Postavljanje servera i baze podataka

Kako bi se postavio server i baza podataka, potrebno je instalirati paket *express* koji je već naveden i objašnjen, *mongoose* koji služi za interakciju s bibliotekom, *nodemon* koji omogućava da se nakon spremanja datoteke server automatski ponovno pokrene bez da se eksplicitno mora to učiniti, te *concurrently* koji omogućava pokretanje više od jedne skripte istovremeno, na primjer kako bi se mogla pokrenuti i server i klijent skripta. Slika 4.4. prikazuje spajanje na server korištenjem porta 5000 i povezivanje na MongoDB bazu. U *config* datoteci stavljen je „MongoURI“, odnosno poveznica na kreiranu bazu podataka, koju dohvaćamo pomoću *get()* metode.

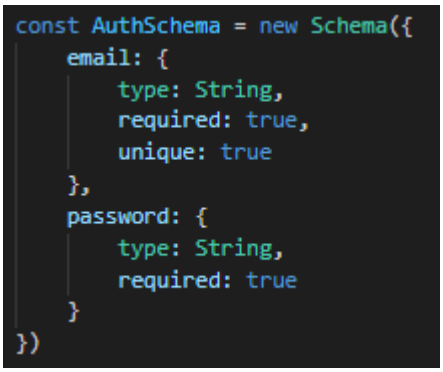
```
const express = require ('express')
const mongoose = require('mongoose')
const config = require('config')

const app = express()
app.use(express.json())

//MongoDB config
const db = config.get('mongoURI')
mongoose
    .connect(
        db,
        {
            useNewUrlParser: true,
            useUnifiedTopology: true
        })
    .then(() => console.log('Connected to MongoDB.'))
    .catch(err => console.log('Fail to connect.', err))

const PORT = 5000
app.listen(PORT, () => console.log(`Server started at port ${PORT}`))
```

Slika 4.4. Kreiranje servera i povezivanje na bazu

4.4 Registracija i prijava korisnika u sustav

Za registraciju i prijavu korisnika potrebno je napraviti odgovarajuće modele. Na sljedećim slikama prikazani su modeli koji su korišteni za registraciju - slika 4.5. i prijavljivanje u sustav - slika 4.6. Model za registraciju sadržava ime, prezime, email, lozinku, datum registracije koji se automatski kreira, profilnu fotografiju korisnika, biografiju korisnika, te pratitelje, dok model za prijavu ima samo email i lozinku, pri čemu email mora biti unikatan, a sva polja moraju biti popunjena.

Slika 4.5. Model za registraciju



Slika 4.6. Model za prijavu

Metoda rute (engl. *route method*) izvedena je iz jedne od nekoliko HTTP metoda kao što su GET, HEAD, POST, PUT, DELETE, PATCH. Na slici je prikazan 4.7. kod korišten za rutu koja je definirana za GET metodu. Korišten je međusloj (engl. *middleware*) 'auth' jer je ova ruta privatna, što znači da ukoliko je korisnik prijavljen u sustav mogu se dobiti korisnikovi podaci, u ovom slučaju bez njegove lozinke što je napravljeno pomoću metode *select('-password')*.

```
// @route   GET /api/auth/users
// @desc    Get all users
// @access  Private

router.get('/users', auth, (req, res) => {
    User
        .find()
        .select('-password')
        .sort({ registration_date: -1 })
        .then(user => res.json(user))
})
```

Slika 4.7. HTTP metoda GET

Nakon što su rute definirane, učitane su u server.js datoteku kao što je prikazano slikom 4.8. Njihova valjanost testirana je prvo u Postmanu, a zatim su korištene u *front end*-u kako bi se mogli slati zahtjevi (engl. *request*) i dobiti odgovori (engl. *response*).

```
app.use('/api/users', require('./api/users'))
app.use('/api/auth', require('./api/auth'))
```

Slika 4.8. Korištenje ruta

Na slici 4.8. prikazano je korištenje definirane API rute za kreiranje korisnika u sustav pomoću metode *post*. Ona prima tri parametra, prvi je URL, drugi parametar su podaci koji se šalju, a treći je konfiguracija. Nakon toga se dobije odgovor (engl. *response*), a ukoliko postoji greška ona se ispisuje. Ukoliko je odgovor ispravan, modal za registraciju se zatvara, te se otvara prozor s porukom da je korisnik uspješno registriran u sustav te da se može prijaviti.

```
const validateSignUp = (e: any) => {
    e.preventDefault();
    const user = {
        first_name: firstName,
        last_name: lastName,
        email,
        password,
    };
    Axios.post('/api/users', user, configWithoutToken)
        .then((res) => {
            window.alert('User successfully created! Proceed to login.');
            toggle(false)
        })
        .catch((err) => {
            setErrorMsg(err.response.data.msg);
            localStorage.removeItem('token');
        });
};
```

Slika 4.9. Kreiranje korisnika

Funkcija za prijavu korisnika vrlo je slična prethodnoj funkciji, osim što za podatke ne šalje ime i prezime, već samo email i lozinku. Email mora biti unikatan.

4.5. Učitavanje slike

Za potrebe učitavanja slike u *backend*-u korišten je multer. Multer je međusloj koji se koristi za rukovanje s višestrukim formatom podataka. Model slike prikazan je na slici 4.10.

```
const ImageSchema = new Schema({
    fileImage: {
        type: String,
        required: true
    }
})
```

Slika 4.10. Model slike

Učitavanje slike pomoću multer-a prikazan je na slici 4.11. DiskStorage metoda omogućava spremanje fotografije pri čemu ima dvije opcije, a to je 'destination' koji označava lokaciju mape u kojoj će se spremati sve učitane fotografije, te 'filename' koji označava naziv datoteke, a ukoliko nije definirana, dodijelit će joj se nasumično ime. Kako bi se fotografija učitala, u ulaznu formu (engl. *Input*) se definira tip koji je 'file' te metoda koja će rukovati na bilo kakvu promjenu, odnosno na odabir fotografije te istu sprema u varijablu pod nazivom 'file'. U post metodu šaljemo podatke 'formData', odnosno FormData sučelje koje je potrebno kako bi se podaci konstruirali parom ključ-vrijednost jer se koristi višestruki format podataka (engl. *multipart/form-data*). Podaci se spremaju, a

otvoreni prozor (engl. *modal*) se zatvara ukoliko je sve prošlo u redu. U slučaju greške ista će se ispisati. U 'put' zahtjevu osim podataka, šalje se i konfiguracija koja je postavljena na višestruki format podataka. Prikaz koda dan je na slici 4.12.

```
const multer = require('multer')

const storage = multer.diskStorage({
    destination: function (req, file, cb) {
        cb(null, './uploads')
    },
    filename: function (req, file, cb) {
        cb(null, file.originalname)
    }
})
```

Slika 4.11. Učitavanje slike pomoću multer-a

```
const updatePhoto = () => {
    const formData = new FormData();
    formData.append('fileImage', file);

    const config: any = { header: { 'Content-Type': 'multipart/form-data' } };
    Axios.put(`/api/users/${currentUser}/photo`, formData, config).then(() => {
        Axios.get('/api/auth/users', {
            headers: {
                'x-auth-token': window.localStorage.getItem('token'),
                'Content-Type': 'application/json',
            },
        })
            .then((res) => {
                setUsers(res.data);
                props.setIsPhotoModalOpen(false);
            })
            .catch((err) => console.log(err));
    });
};
```

Slika 4.12. Učitavanje fotografije

4.6. Ažuriranje profila

Korisnik ima mogućnost ažurirati svoj profil kojeg je napravio klikom na gumb kojim se otvara prozor. Tu se nalaze trenutni korisnikovi podaci, a to su ime, prezime i biografija koju korisnik može izmijeniti. Ti isti podaci koji su uneseni u polja se šalju serveru, te ukoliko je sve prošlo u redu dobije se odgovor (engl. *response*) s novim, ažuriranim podacima koji se spremaju u stanje, a pomoću tog stanja se mogu nove vrijednosti prikazati u aplikaciji. U slučaju bilo kakve greške, ista će se ispisati. Kako bi se podaci sačuvali i poslali, stanje se postavlja na *event.target.value* svojstvo koje

vraća element, odnosno njegovu vrijednost kojiža je okinuta (engl. *trigger*). Server prima te upisane vrijednosti iz *req.body*-a, pronalazi korisnika po *id*-u te ga ažurira s istim podacima koristeći metodu *findByIdAndUpdate()*. Na slici 4.13. prikazana je put zahtjev koji ažurira korisnika na serveru.

Slika 4.13. Ruta za ažuriranje korisnika

4.7. Kreiranje i brisanje objave

Model objave sadržava korisnikov id, sadržaj objave, datum kreiranja, polje sviđanja (engl. *likes*), zatim vidljivost posta koja može biti postavljena na *true* ili *false* što označava privatnu ili javnu objavu, uzimajući u obzir da je početno postavljena na *true*, odnosno javno, te polje komentara. Komentar sadržava tekst komentara kao i korisnikov id. Na slici 4.14. prikazan je model objave. Za kreiranje se koristi ruta 'post' koja će prvo spremiti objavu, a zatim poslati odgovor (engl. *response*) klijentu. Podatke id i sadržaj objave prima iz *req.body*-a, a prikaz takve rute vidljiv je na slici 4.15. Brisanje objave se ostvaruje pomoću rute 'delete' koja mora primiti u url-u id određene objave koja se briše, zatim se pronalazi po tom id-u metodom *findById()*. Funkcija za brisanje objave prikazana je slikom 4.16. Ona kao parametar prima id određene objave, te ju šalje serveru.

```
const PostSchema = new Schema({
    userID: {
        type: String,
    },
    content: {
        type: String,
        required: true,
    },
    postDate: {
        type: Date,
        default: Date.now,
    },
    likes: [
        {
            type: String,
        },
    ],
    isPublic: {
        type: Boolean,
        default: true,
    },
    comments: [
        {
            text: String,
            userID: String,
        },
    ],
});
```

Slika 4.14. Model objave

```
router.post('/', auth, (req, res) => {

    const newPost = new Post({
        userID: req.user._id,
        content: req.body.content
    })

    newPost
        .save()
        .then(post => {
            res.json(post)
        })
        .catch(err => console.log(err))
})
```

Slika 4.15. Ruta za kreiranje objave

```
const deletePost = (postID: string) => {
    Axios.delete(`/api/posts/${postID}`, config)
        .then(() =>
            Axios.get('/api/posts/', config).then((res) => {
                setPosts(res.data);
            }),
        )
        .catch((err) => console.log(err));
};
```

Slika 4.16. Funkcija za brisanje objave

4.8. Dodavanje funkcionalnosti sviđanja

Svaka objava ima gumb 'sviđa mi se'. Kada je on kliknut, poziva se metoda likePost() koja za parametar prima id objave, te isti šalje serveru rutom 'put'. Ta ruta će, za razliku od 'post' rute, samo ažurirati trenutnu objavu, dok 'post' ruta kreira novu. Kao još jedan parametar, ruta će poslati i konfiguraciju 'Content-type' koji je postavljen na 'application/json'. U frontendu, rukovanje s tom rutom prikazano je na slici 4.17., dok je rukovanje na backendu prikazano na slici 4.18.

```
const likePost = (postID: string) => {
    Axios.put('/api/posts/like', { postID }, config).then(() =>
        Axios.get('/api/posts/', config).then((res) => {
            setPosts(res.data);
        }),
    );
};
```

Slika 4.17. Metoda za označavanje objave sa 'sviđa mi se'

```
router.put('/like', auth, (req, res) => {
    Post.findByIdAndUpdate((req.body.postId), {
        $push: { likes: req.user._id }
    }, {
        new: true
    }).exec((err, result) => {
        if (err) res.status(422).json({ msg: err })
        res.json(result)
    })
})
```

Slika 4.18. Ruta za označavanje objave sa 'sviđa mi se'

U ruti se prvo pronalazi objava pomoću id objave, zatim se u polje pod nazivom 'likes' stavlja id korisnika koje je označio objavu sa 'sviđa mi se'. Za odznačavanje objave 'sviđa mi se' koristi se vrlo slična ruta, osim što se koristi 'pull', a ne 'push' metoda. 'pull' metoda vadi element iz polja,

odnosno vadi id korisnika iz polja sviđanja. Metoda na frontend-u je također vrlo slična, jedina razlika je ta što je url rute '/unlike', a ne 'like'.

4.9. Praćenje korisnika

Kako bi se korisnici mogli pratiti (engl. *follow*), napravljena je 'put' ruta u backend-u. Ta ruta pronalazi korisnika po njegovom id-u te ga ažurira. Ono što se ažurira jest polje pratitelja i sljedbenika pomoću 'push' metode koja će dodati u polje novog pratitelja, odnosno sljedbenika. Ruta je prikazana slikom 4.19.

```
router.put('/follow', auth, (req, res) => {
    User.findByIdAndUpdate(req.body.followId, {
            $push: { followers: req.user._id }
        },{
            new: true
        }, (err, result) => {
            if(err) res.status(422).json({msFg: err})
            User.findByIdAndUpdate(req.user._id, {
                $push: {following: req.body.followId}
            }, {
                new: true
            })
            .then(result => res.json(result))
            .catch(err => res.status(422).json({msg: err}))
        }
    )
})
```

Slika 4.19. Ruta za pratitelje i sljedbenike

Također, kako bi se omogućila suprotna radnja od praćenja (engl. *unfollow*), napravljena je vrlo slična ruta, koja umjesto 'push' metode poziva 'pull' metodu. Ta metoda vadi element iz polja. U frontend-u je napravljena funkcija za praćenje korisnika koja kao parametar prima id korisnika te ga pomoću rute 'put' šalje serveru, a u uspješnom odgovoru se dohvaćaju svi korisnici. Primjer je prikazan na slici 4.20.

```
const followUser = (userID: string) => {
    Axios.put(
        '/api/users/follow',
        {
            followId: userID,
        },
        config,
    ).then(() =>
        Axios.get('/api/auth/users', config)
            .then((res) => {
                setAllUsers(res.data);
            })
            .catch((err) => console.log(err)),
    );
};
```

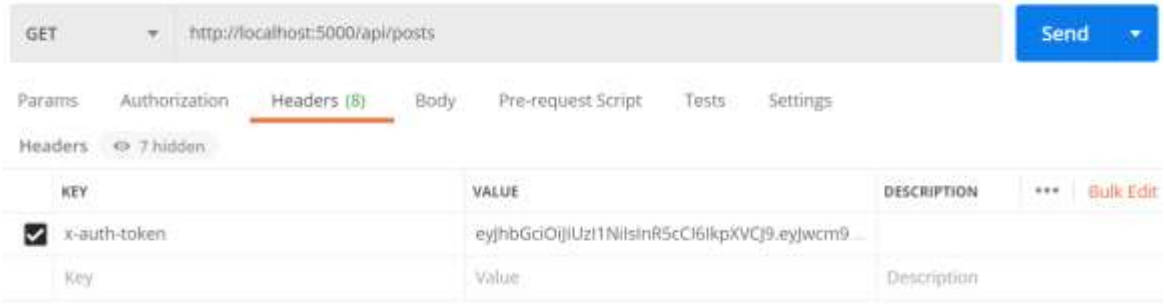
Slika 4.20. Funkcija za praćenje korisnika

5. TESTIRANJE I IZGLED WEB APLIKACIJE

U ovom poglavlju objašnjen je način testiranja *backend* dijela aplikacije te izgled svakog zaslona i prozora koji korisnik može otvoriti (engl. *modal*) zajedno s funkcionalnostima koje ova aplikacija nudi.

5.1. Testiranje aplikacije

Za potrebe testiranja *backend* dijela aplikacije korištena je platforma Postman. Pomoću nje se mogu testirati sve rute koje se koriste u aplikaciji. Rute se prvo testiraju u Postman-u, te ukoliko je odgovor (engl. *response*) dobar, možemo ju implementirati u *frontend* dijelu aplikacije. Postman, između ostaloga, sadržava mjesto gdje se unosi url rute koja se želi testirati, zatim 'headers' u kojem se definiraju tokeni, tip sadržaja i slično, te 'body' u kojem se zapisuju podaci koji će biti poslani. Te podatke će slati krajnji korisnik unosom ili odabirom u aplikaciji, no ovdje se mora upisati kako bi se moglo testirati. Također sadržava i prozor u kojem se dobije odgovor servera i status. Status '200 OK' označava da je sve prošlo u redu. Na slici 5.1. prikazan je dio platforme Postman u kojoj definiramo rutu i token za dohvaćanje svih postova iz aplikacije.



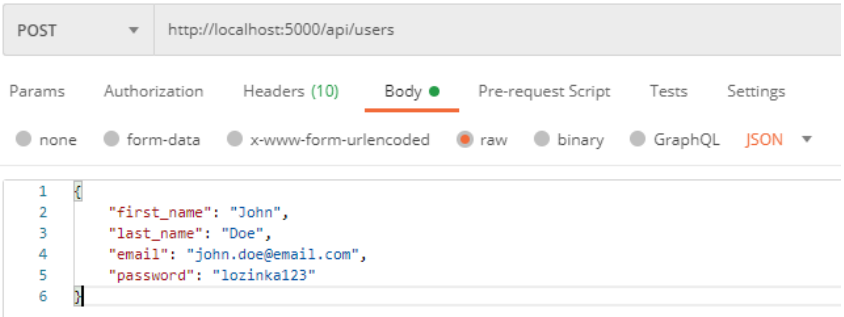
Slika 5.1. Definiranje rute i tokena u Postman-u

Na slici 5.2. prikazan je odgovor (engl. *response*) na definiranu 'get' rutu. Ono što je prikazano jest polje objekata koji izgledaju kao i model objave sadržavajući polje sviđanja, vidljivost, id objave, podatke korisnika, sadržaj objave, datum kreiranja te polje komentara. Ovim je testirano da je 'get' ruta za dohvaćanje svih objava uspješna.

```
[
    {
        "likes": [
            "5efb2296ca33ba3d981398ff"
        ],
        "visibility": "Public",
        "_id": "5f2869c1cdcd5c3104a3569b",
        "userID": {
            "profile_image": "uploads\\20191117_162232-01.jpeg",
            "_id": "5efa2f87647eea292c8b93ce",
            "first_name": "Pero",
            "last_name": "Perić"
        },
        "content": "my first post",
        "registration_date": "2020-08-03T19:47:13.260Z",
        "comments": [
            {
                "_id": "5f286a5fcdcd5c3104a3569d",
                "text": "comment",
                "userID": "5efb22a0ca33ba3d98139900"
            }
        ],
        "__v": 0
    },
    {
        "likes": [],
        "visibility": "Private",
        "_id": "5f2705d8ed804d42a89ce850",
```

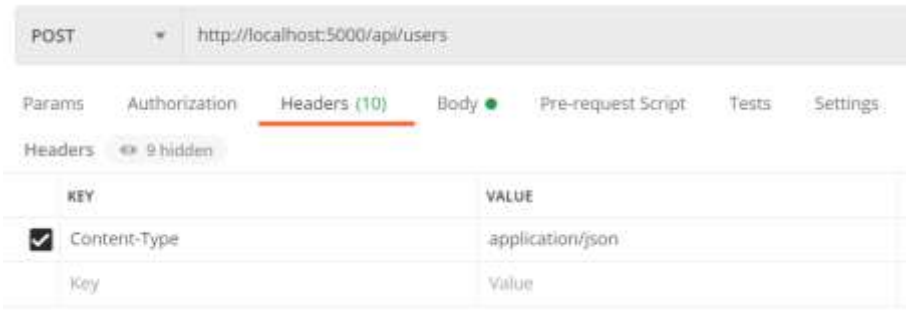

Slika 5.2. Odgovor na zahtjev u Postman-u

U idućem primjeru prikazana je 'post' ruta za kreiranje novog korisnika. Na slici 5.3. definirane su vrijednosti koje se moraju upisati prilikom registracije korisnika, odnosno ime, prezime, email adresa te lozinka. Bez toga se korisnik ne može registrirati.



Slika 5.3. Registracija korisnika u Postman-u

U stupcu pod nazivom 'headers' mora biti definiran tip sadržaja, a to je 'Content-type: application/ json ', vidljivo na slici 5.4.



Slika 5.4. Definiranje tipa sadržaja u Postman-u

Odgovor sadržava token i podatke korisnika poput id-a, imena, prezimena i email-a. Token je važan jer se koristi u ostatku aplikacije kako bi se znalo je li korisnik prijavljen u sustav. Bez toga korisnik ne može koristiti većinu aplikacije, tada je jedino dostupna početna stranica s mogućnosti registracije i prijavljivanja u sustav.

5.2. Izgled web aplikacije

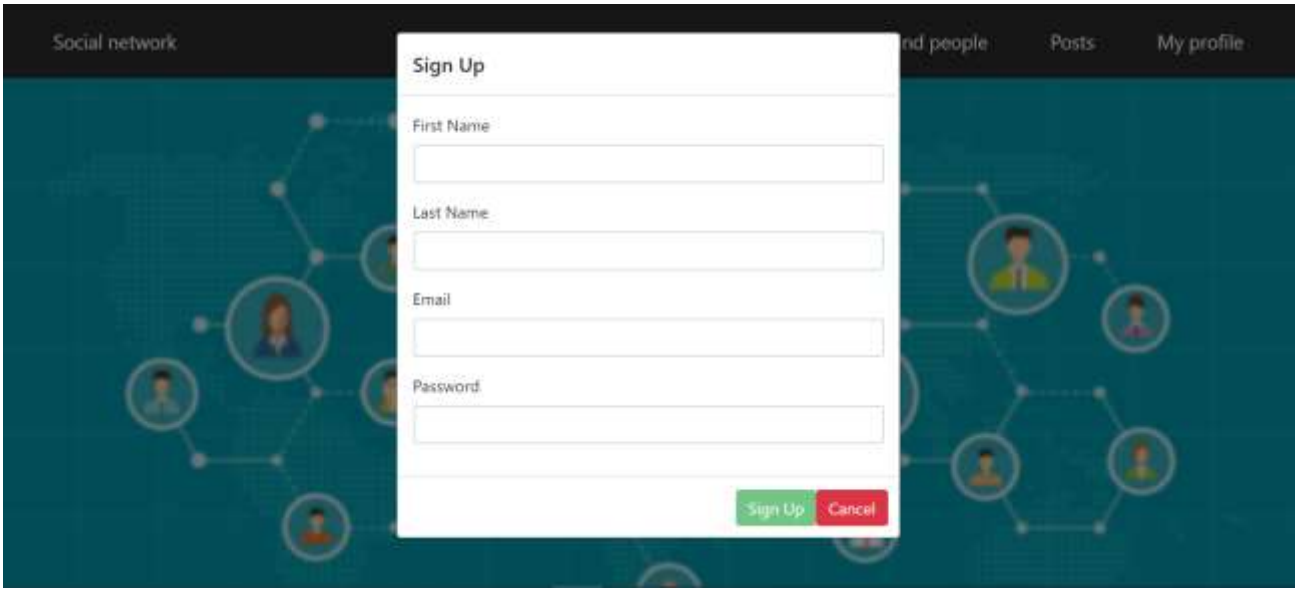
U ovom potpoglavlju detaljno je prikazan izgled aplikacije, odnosno svaki njezin zaslon. Na slici 5.5. prikazan je početni zaslon aplikacije. Korisnik bez prijave u sustav ne može koristiti ostatak aplikacije. Svi linkovi u navigaciji su nedostupni ukoliko korisnik nije prijavljen. Na početnom zaslonu nalaze se i dva gumba, 'Sign Up' te 'Sign In, koji otvaraju prozor za registraciju i za prijavu.

Slika 5.5. Početni zaslon aplikacije

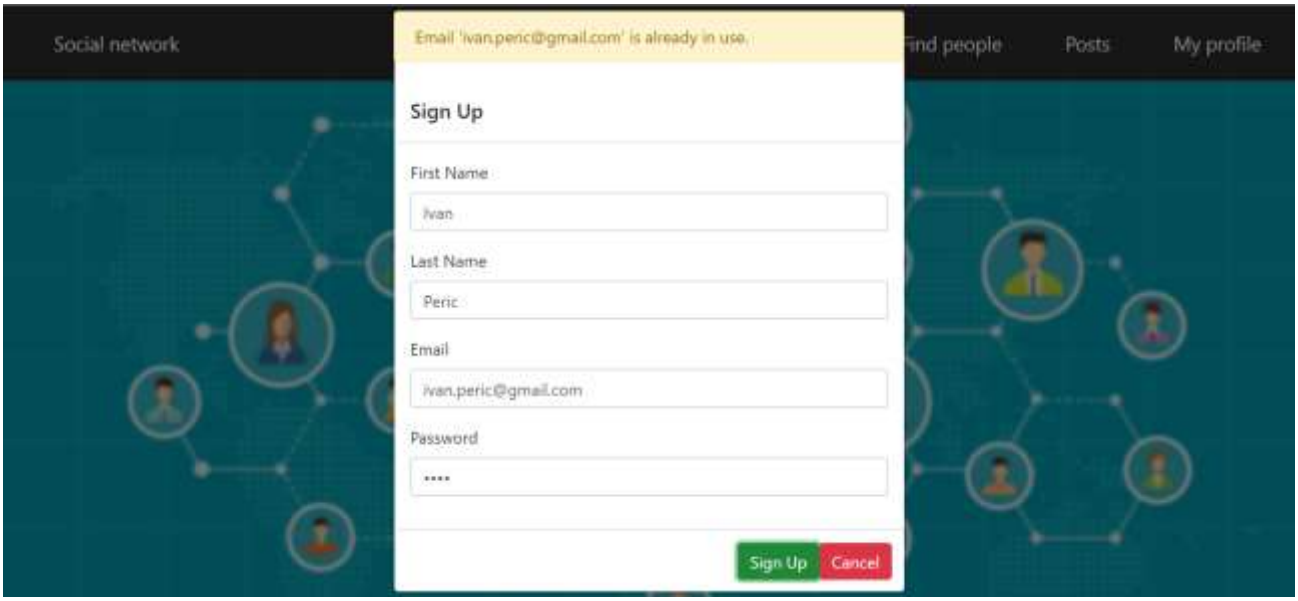
Ukoliko je korisnik registriran u sustavu, može se koristiti ostalim navigacijskim linkovima, a uz to vidljiv je i jedan dodatni link u navigaciji za odjavu iz sustava pod nazivom 'Log out' prikazano slikom 5.6.

Slika 5.6. Početna stranica aplikacije prijavljenog korisnika

Korisnik može otvoriti prozor za registraciju klikom na gumb 'Sign Up' na početnom zaslonu. Taj prozor sadržava polja za unos korisnikovog imena, prezimena, email adrese te lozinke. Također se u istom prozoru nalaze i dva gumba, 'Sign Up' i 'Cancel' koji omogućavaju registraciju, odnosno prekid registracije. Ukoliko je email adresa već korištena, a korisnik je kliknuo na gumb 'Sign Up', ispisat će se greška te se korisnik neće moći registrirati u sustav. Ukoliko nisu upisana sva tražena polja, gumb za registraciju nije omogućen. Na slikama 5.7. i 5.8. prikazan je prozor za registraciju bez upisanih polja te račun s već postojećom adresom.

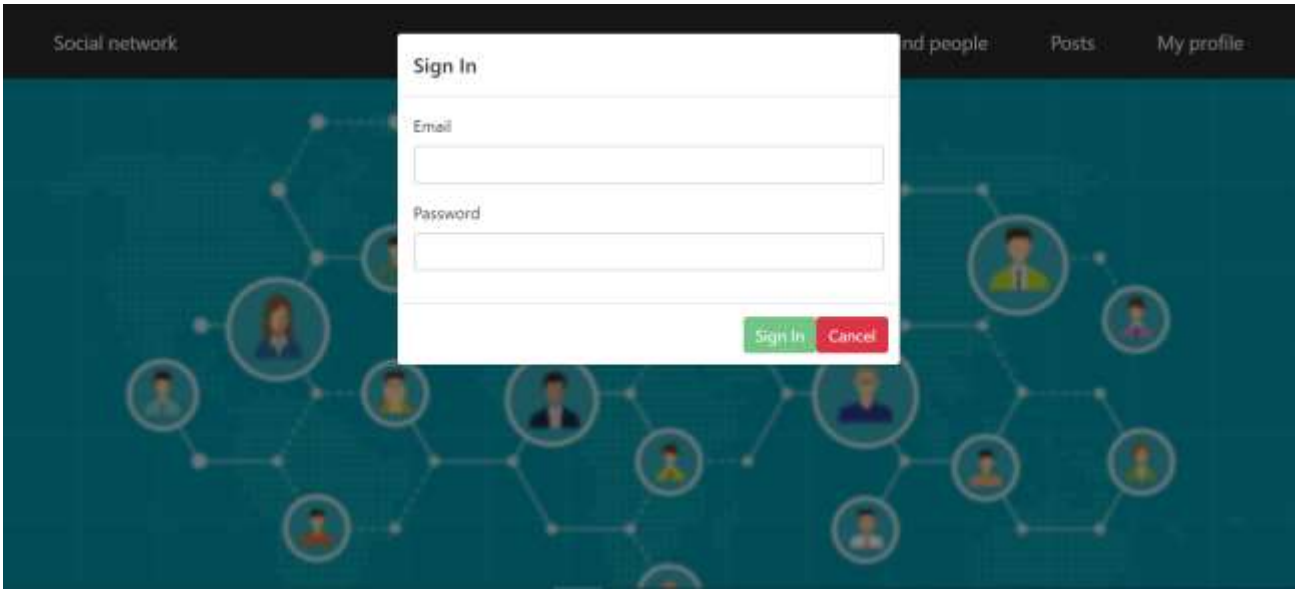


Slika 5.7. Početna stranica aplikacije s prozorom za registraciju korisnika

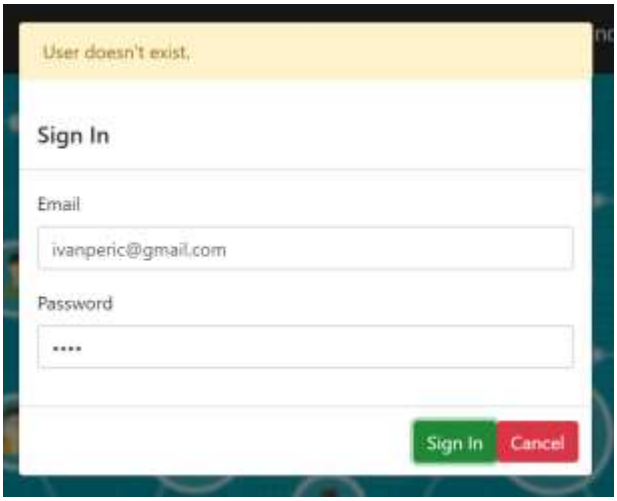


Slika 5.8. Prozor za registraciju korisnika s postojećim računom

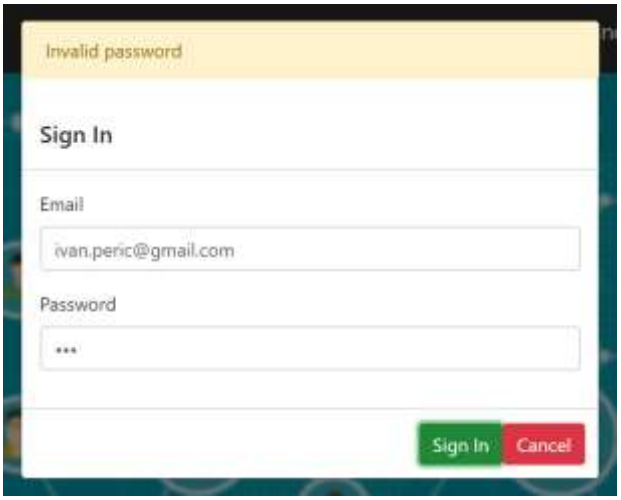
Na slici 5.9. prikazan je početni zaslon s prozorom za prijavu korisnika u sustav. Prozor sadržava polja za unos email adrese i lozinke, kao i dva gumba za prijavu i prekid radnje. Ukoliko su polja prazna, gumb za prijavu je onemogućen. Na slikama 5.10. i 5.11. prikazani su prozori ukoliko je prijava korisnika u sustav nevažeća. Na slici 5.10. korisnik je unesao krivu email adresu koja nije registrirana u sustavu, a na slici 5.11. korisnik je unesao krivu lozinku.



Slika 5.9. Početna stranica aplikacije s prozorom za prijavu korisnika

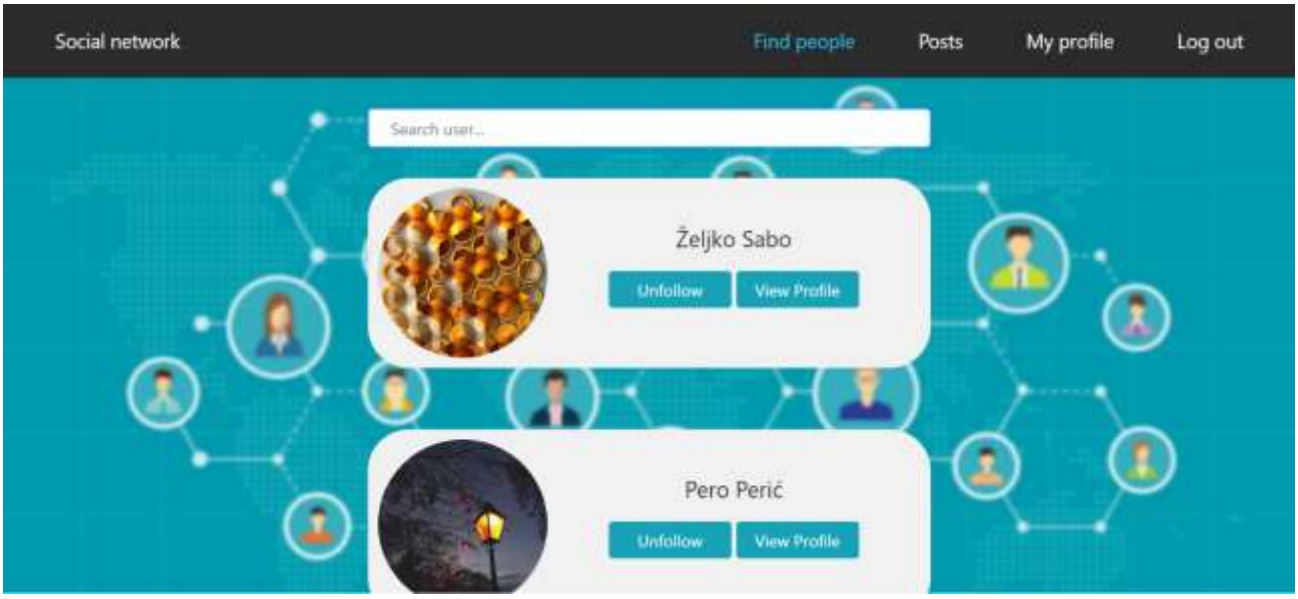


Slika 5.10. Prozor za prijavu s greškom

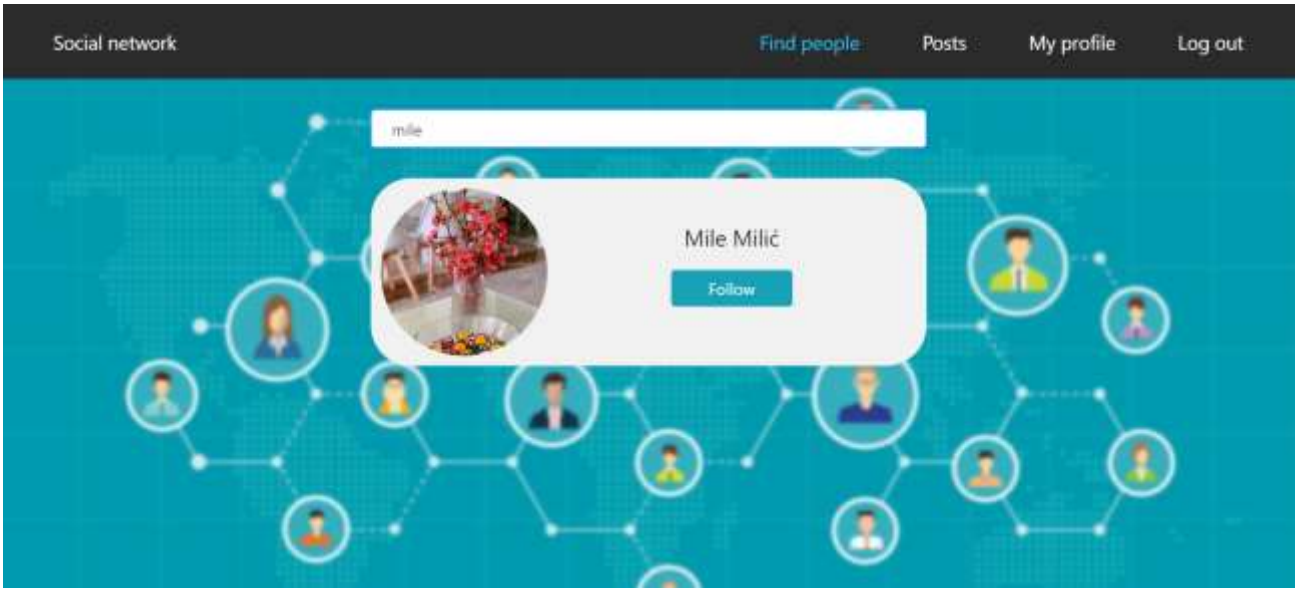


Slika 5.11. Prozor za prijavu s greškom

Na slici 5.12. prikazan je prozor pod nazivom 'Find people' sa svim korisnicima aplikacije osim trenutnog prijavljenog korisnika. Na ovom prozoru korisnici se mogu pretraživati u tražilici prikazano slikom 5.13. Svaki korisnik je prikazan slikom, imenom, prezimenom i gumbom za praćenje 'Follow', odnosno ukoliko se već korisnik prati dostupan je gumb za prestanak praćenja korisnika pod nazivom 'Unfollow'. Ukoliko se korisnik prati, moguće je vidjeti gumb 'View profile' kojim se može otići na korisnikov profil.



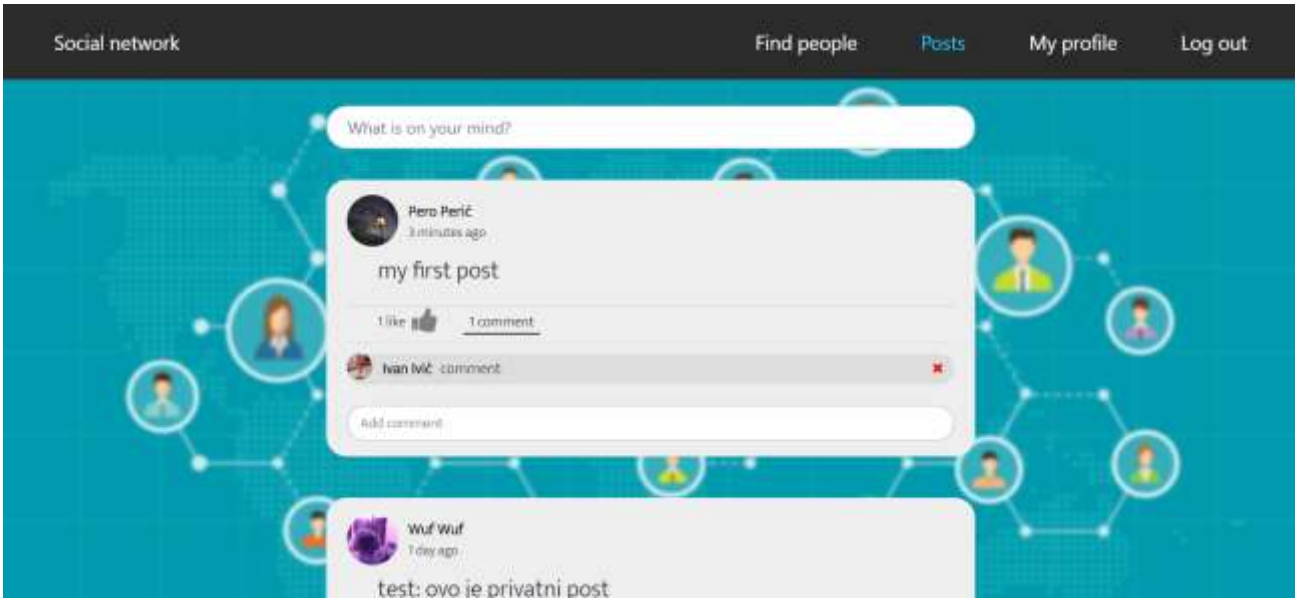
Slika 5.12. Prikaz u aplikaciji - 'pronađi korisnike'



Slika 5.13. Prikaz u aplikaciji - 'pronađi korisnike' s uključenom tražilicom

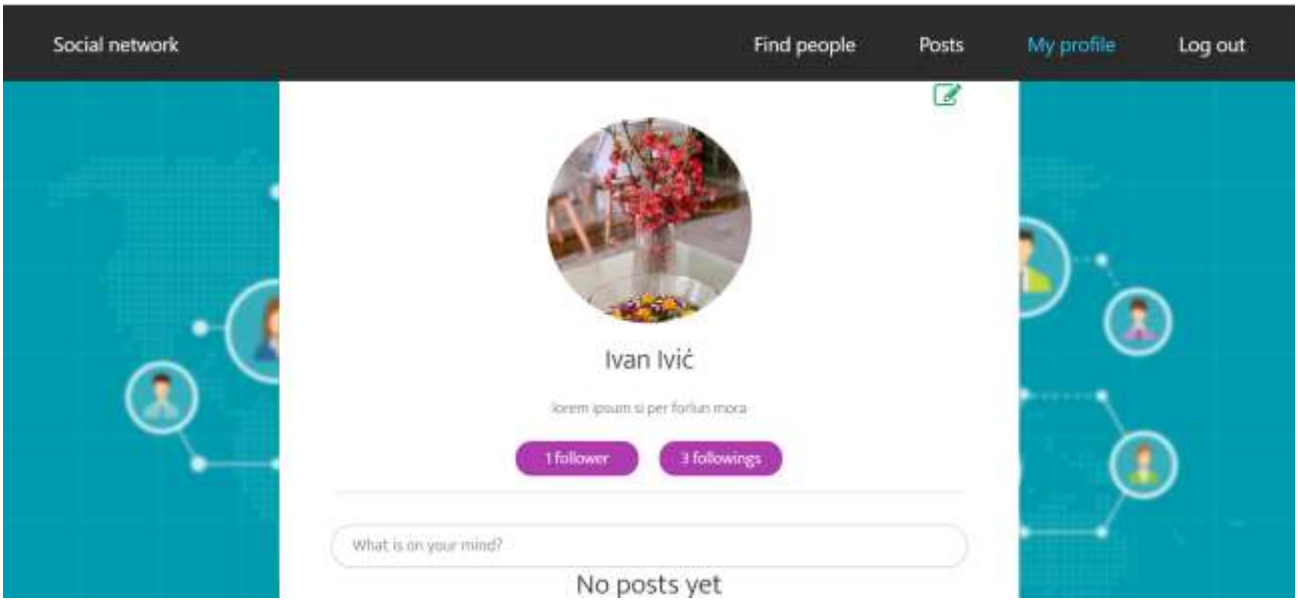
Na slici 5.14. prikazan je prozor aplikacije pod nazivom 'Posts' gdje su prikazane sve javne objave, kao i privatne i javne objave svih korisnika kojeg trenutni korisnik prati. U istom prozoru korisnik može napisati novu objavu. Svaka objava sadržava sliku korisnika, njegovo ime i prezime, vrijeme kada je objava kreirana, sadržaj objave, gumb za označavanje objave sa 'sviđa mi se', broj sviđanja kao i broj komentara, te sadržava sve komentare na objavu, a ako je trenutni korisnik napisao

komentar, može ga i obrisati. Ukoliko je korisnik napisao objavu, istu može obrisati. Komentari se mogu proširiti na sve komentare ili na prva tri komentara.



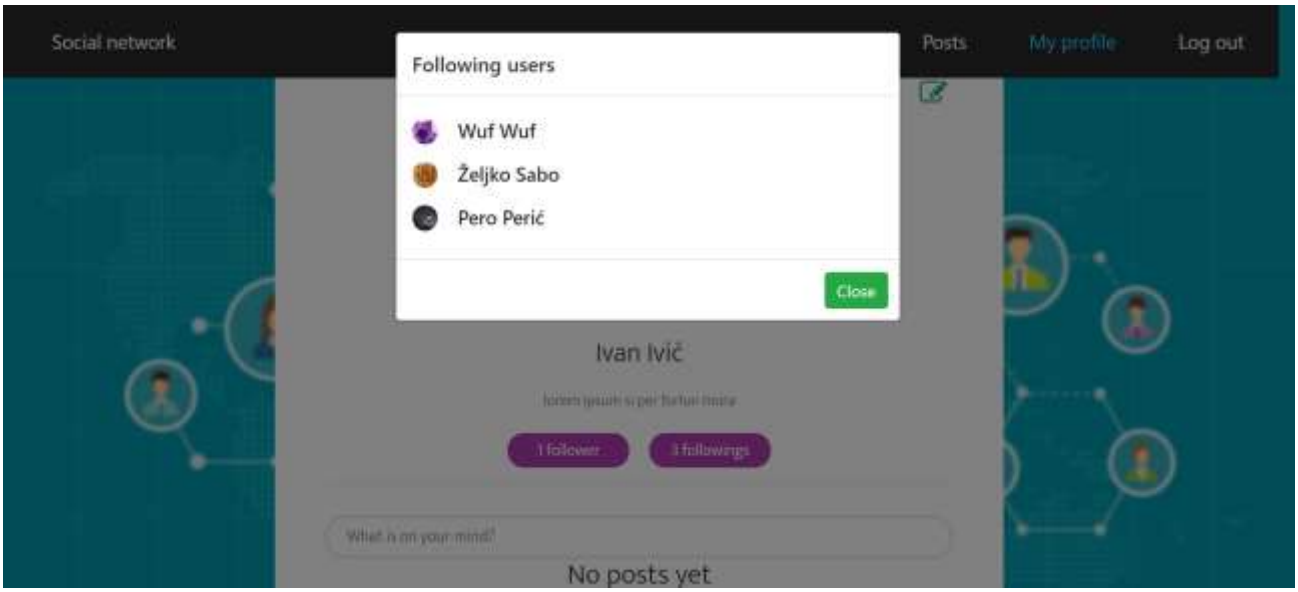
Slika 5.14. Prikaz u aplikaciji - 'Objave'

Na slici 5.15. prikazan je prozor 'My profile' koji prikazuje profil trenutnog korisnika. Profil sadržava opće informacije o korisniku, poput slike, imena i prezimena te njegove biografije. Sadržava gumb za uređivanje profila, prikaz svih njegovih objava i informacije o pratiteljima i sljedbenicima.



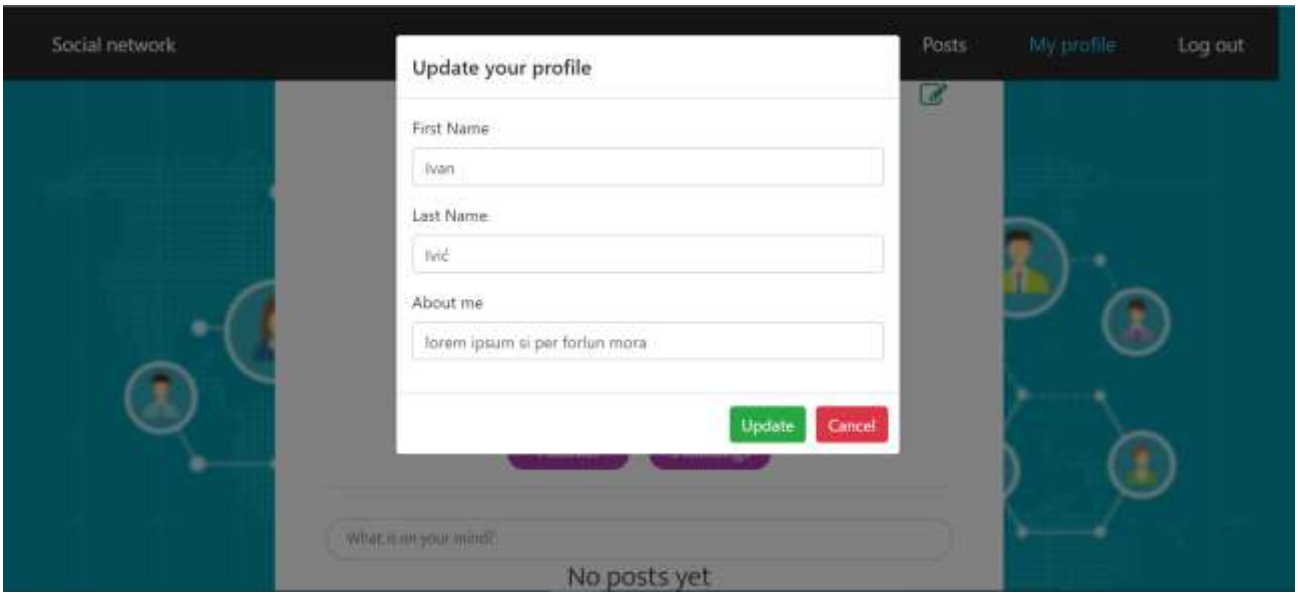
Slika 5.15. Prikaz u aplikaciji – 'Moj profil'

Na slici 5.16. prikazan je prozor sa svim pratiteljima. Svaki pratitelj je prikazan slikom, imenom i prezimenom.



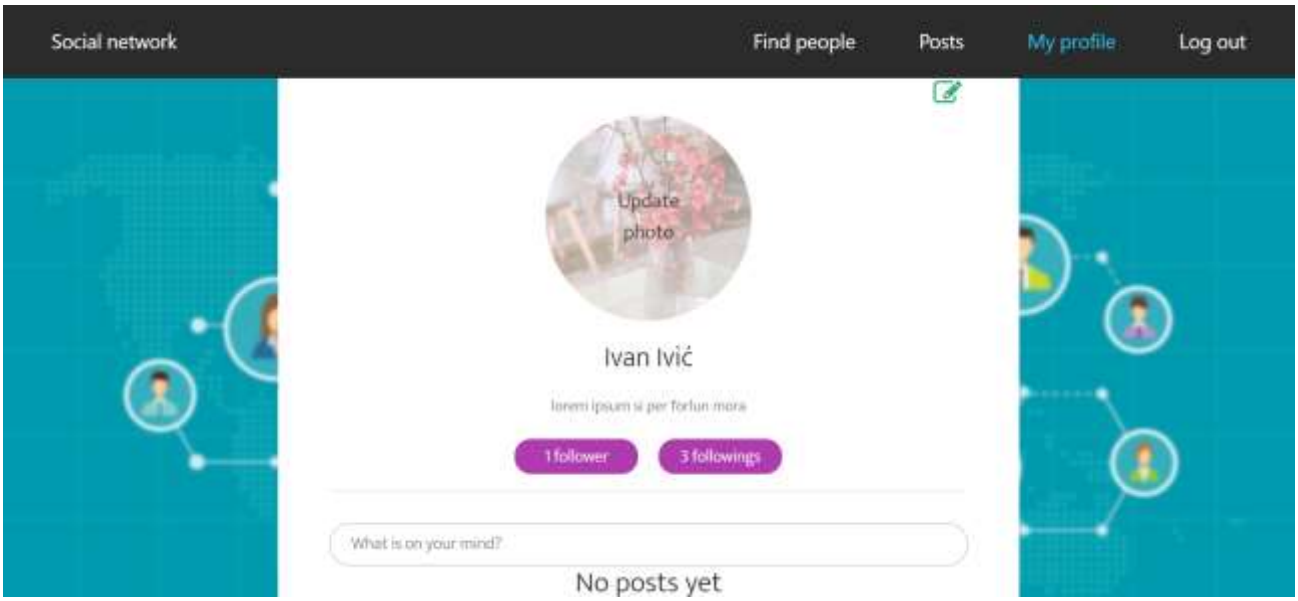
Slika 5.16. Prikaz u aplikaciji - 'Moj profil' s prozorom korisnika kojeg korisnik prati

Slika 5.17. prikazuje prozor za ažuriranje profila. Klikom na zelenu ikonu za uređivanje, otvara se prozor sa svim postojećim informacijama trenutnog korisnika. Korisnik ih može izmijeniti i kliknuti na gumb 'Update' za ažuriranje svih promjena.

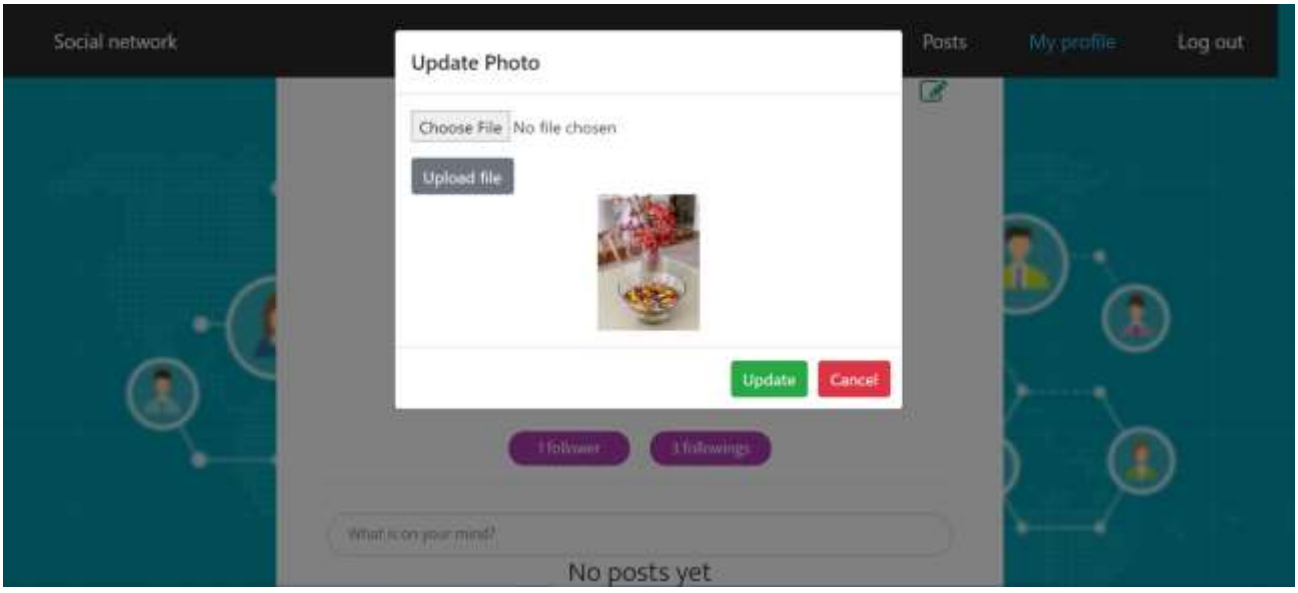


Slika 5.17. Prikaz u aplikaciji – 'Moj profil' s prozorom za ažuriranje profila

Korisnik može ažurirati i sliku profila prelaskom miša preko postojeće fotografije prilikom čega se pojavljuje mogućnost ažuriranja pod nazivom 'Update photo' prikazano na slici 5.18. Klikom na fotografiju, otvara se prozor prikazan slikom 5.19. u kojem se fotografija može učitati i ažurirati.

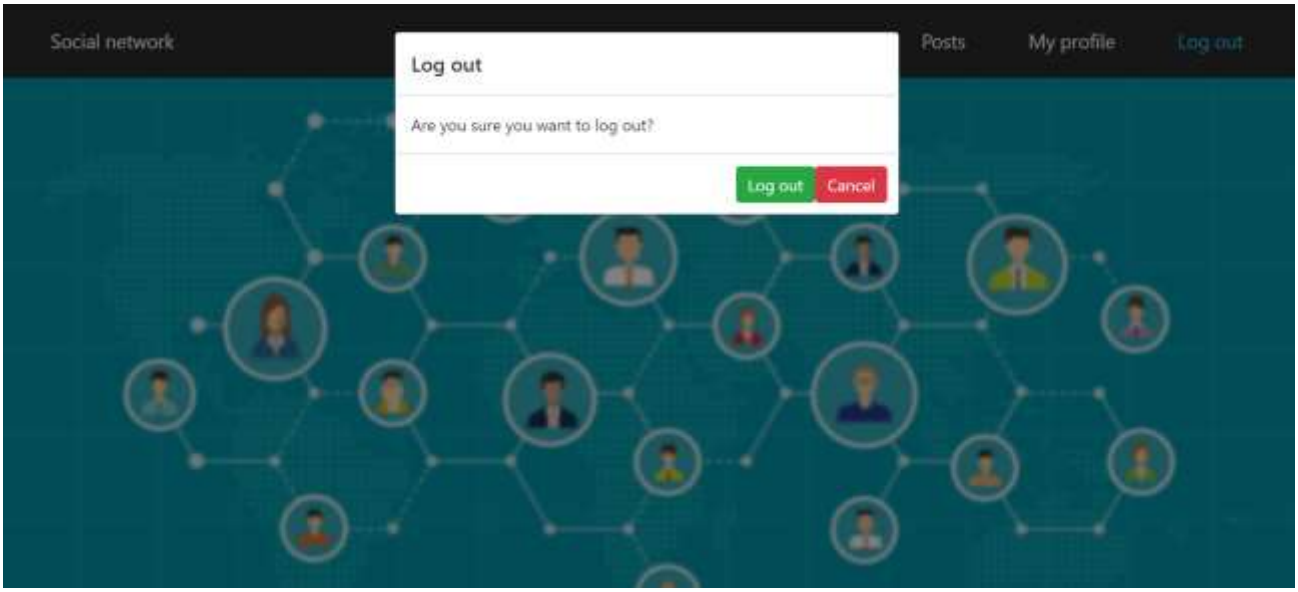


Slika 5.18. Prikaz u aplikaciji – 'Moj profil' za ažuriranje fotografije



Slika 5.19. Prikaz u aplikaciji – 'Moj profil' s prozorom za ažuriranje fotografije

Korisnik se može odjaviti iz aplikacije klikom na navigacijski link 'Log out' prilikom čega se otvara prozor s potvrdom za odjavljivanje ili odustajanjem od odjavljivanja prikazano na slici 5.20.



Slika 5.20. Prikaz u aplikaciji – 'Odjava'

6. ZAKLJUČAK

Cilj ovog diplomskog rada bio je napraviti web aplikaciju koja će služiti kao društvena mreža, a uz to cilj je bio i obaviti testiranje aplikacije. Razvijena je aplikacija koja korisnicima omogućava kreiranje profila, prijavu u sustav, pronalazak korisnika, pretraživanje korisnika, pregled profila korisnika, praćenje korisnika, pisanje objava, pregled vremena kreiranja objave, pisanje komentara, pregled svih komentara ili prva tri komentara, opciju sviđanja objave, postavljanje objave na privatno ili javno dostupnu, ažuriranje informacija trenutnog korisnika, ažuriranje slike profila, pregled svih sljedbenika i pratitelja te odjavu iz aplikacije. Aplikacija je u potpunosti funkcionalna što je pokazalo i testiranje iste. Ovu aplikaciju mogu koristiti brojni korisnici koji žele dijeliti svoja mišljenja, iskustva, događaje, koji se žele povezati s drugim ljudima i slično. Ova se aplikacija može dodatno unaprijediti i nadograditi mogućnostima poput objavljivanja fotografije u objavi, ugradnjom opcije sviđanja komentara, zatim slanja poruka, promjenom dizajna.

LITERATURA

[1] What is the Document Object Model?, Jonathan Robie, Texcel Research, https://www.w3.org/TR/WD-DOM/introduction.html [posjećeno 26.06..2020.]

[2] Thinking in React, https://reactjs.org/docs/thinking-in-react.html, [posjećeno 27.06.2020.]

[3] HTML elements, Learn to Code, https://learn-to-code.london.cloudapps.digital/lesson-1-html-and-css.html#html-elements, [posjećeno 27.06.2020.]

[4] HTML Tags: Past, Present, Proposed, Martin Rinehart, http://www.martinrinehart.com/frontend-engineering/engineers/html/html-tag-history.html, [posjećeno 27.06.2020.]

[5] CSS Syntax, w3schools, https://www.w3schools.com/css/css_syntax.ASP, [posjećeno 27.06.2020.]

[6] Add Reactstrap Components In ReactJS, Sanwar Ranwa, Web Dev Zone, https://dzone.com/articles/add-react-strap-components-in-reactjs, [posjećeno 27.06.2020.]

[7] Buttons, Reactstrap, https://reactstrap.github.io/components/buttons/, [posjećeno 27.06.2020.]

[8] M. Satheesh, B. Joseph D'mello, J. Krol: Web Development With MongoDB and NodeJS, str 2-3, Birmingham, UK, 2015.

[9] Express Web Framework (NodeJS/Javascript), MDN contributors, https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs, [posjećeno 27.06.2020.]

[10] A. Singh, S. Ahmad: MongoDB: Simply In Depth, str. 6, 2019.

[11] Visual Studio Code, Wikipedia, https://en.wikipedia.org/wiki/Visual_Studio_Code, [posjećeno 27.06.2020.]

[12] A. Freeman, Essential TypeScript: From Beginner to Pro, str. 35, London, UK, 2019.

[13] Simplest Thing Possible: Introduction to TypesScript, John V. Peterson, https://www.codemag.com/Article/1305051/Simplest-Thing-Possible-Introduction-to-TypeScript, [posjećeno 06.07.2020.]

SAŽETAK

Izrađena društvena mreža je web aplikacija koja omogućava korisnicima brojne funkcionalnosti poput pisanja objava, zbližavanja s drugim korisnicima, dijeljenje komentara, mišljenja, uređivanje profila, praćenje i slično. Cilj je pružiti korisnicima jednostavno sučelje koje će moći koristiti zajedno s mnogim funkcionalnostima koje ova aplikacija nudi. Za izradu aplikacije na *frontend*-u koristila se Javascript biblioteka React, za *backend* koristio se Nodejs, dok se za bazu podataka koristio MongoDB.

Ključne riječi: društvena mreža, MongoDB, Nodejs, React, web aplikacija

ABSTRACT

Social network web application

Social network web application allows users a number of functionalities such as posting and sharing posts, connecting with others, commenting, expressing thoughts, editing profiles, following other users etc. The main goal of this app is to provide users with simplistic and intuitive GUI which they will be able to use in conjunction with many functionalities which are presented in this app. In purpose of building this app the tools used in this project were Javascript library 'React' for frontend, Node.js for backend, while MongoDB was used as database.

Keywords: social network, MongoDB, Nodejs, React, web application

ŽIVOTOPIS

Marina Vratarić rođena je 29.12.1995. godine u Slavonskom Brodu. U Sibinju je završila osnovnu školu te nakon toga upisuje prirodoslovno-matematičku gimnaziju „Matija Mesić" u Slavonskom Brodu. Završetkom gimnazije 2014. maturira te se iste godine upisuje na Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, smjer Računarstvo. Tamo završava preddiplomski sveučilišni studij te dobiva zvanje prvostupnica inženjerka računarstva (univ. bacc. ing. comp.). 2018. godine upisuje na istom fakultetu diplomski sveučilišni studij, smjer Programsko inženjerstvo. Od stranih jezika služi se engleskim jezikom. Stručnu praksu odrađivala je u tvrtki Cobe Tech.

Marina Vratarić,

Potpis: _____

PRILOZI

- Elektronička verzija rada (dokument u .docx i .pdf formatu)
- Programski kod aplikacije