Izrada internetske stranice i chatbota za udrugu Novogradiške njuške

Solić, Marko

Sveučilište Jurja Dobrile u Puli
Fakultet Informatike u Puli

MARKO SOLIĆ

IZRADA INTERNETSKE STRANICE I CHATBOTA ZA UDRUGU NOVOGRADIŠKE NJUŠKE

Završni rad

Pula, rujan 2022.godine

Sveučilište Jurja Dobrile u Puli
Fakultet Informatike u Puli

MARKO SOLIĆ

IZRADA INTERNETSKE STRANICE I CHATBOTA ZA UDRUGU NOVOGRADIŠKE NJUŠKE

Završni rad

JMBAG: 0303076291, redovni student

Studijski smjer: Sveučilišni preddiplomski studij informatika

Kolegij: Osnove IKT

Znanstveno područje: Društvene znanosti

Znanstveno polje: Informacijske i komunikacijske znanosti

Znanstvena grana: Informacijski sustavi i informatologija

Mentor: doc.dr.sc Snježana Babić

Pula, rujan 2022.godine

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Marko Solić, ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

U Puli, rujan 2022. godine

Student

IZJAVA

o korištenju autorskog djela

Ja, dolje potpisani Marko Solić dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom „Izrada internetske stranice i chatbota za udrugu Novogradiške njuške“ koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama. Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, rujan 2022. godine

Student

Sažetak

U današnje vrijeme chatbot je postigao veliku popularnost i njegova primjena je sve učestalija, često se opisuje kao jedan od najnaprednijih i najperspektivnijih izraza interakcije između ljudi i strojeva. Razlog njegovoj popularnosti je što daje zaposlenicima vremena da se usredotoče na važnije zadatke i sprječava klijente da čekaju na odgovore. Cilj ovog završnog rada je izraditi chatbot i internetsku stranicu za udrugu Novogradiške njuške koja je usmjerena na spašavanje, zbrinjavanje, i liječenje napuštenih kućnih ljubimaca. Web stranica i chatbot su izrađeni kao primjer kako sadašnja tehnologija može pozitivno utjecati i pridonijeti veliku pomoć raznim udrugama. U udruzi kao što su Novogradiške njuške, najbitnija je razmjena informacija, kako bi se omogućilo što uspješnije djelovanje izrađena je internetska stranica na kojoj se nalaze sve važne informacije o kućnim ljubimcima. U konačnici, primarna zadaća bila bi omogućiti bolju komunikaciju, to jest lakšu i bržu razmjenu informacija između dvije vrste korisnika, onih koji su spremni udomiti svog novog kućnog ljubimca i onih koji su željni dati ih na udomljavanje (Jadranka Grupa) (Vesti.rs).

Ključne riječi: web stranica, chat-bot, MongoDB, React.js, Express.js, Node.js

Izvorni kod klijent dijela aplikacije: https://github.com/MarkoSolic/novogradiske-njuske-frontend

Izvorni kod server dijela aplikacije : https://github.com/MarkoSolic/novogradiske-njuske-backend

Abstract

Nowadays chatbot has achieved great popularity and its application is all participants, it is often described as one of the most advanced and promising expressions of interaction between humans and machines. The reason for its popularity is that it gives employees time to focus on more important tasks and for clients to wait for answers. The goal of this final work is to create a chatbot and a website for the Novogradiške njuške association, which is focused on rescuing, caring for and treating abandoned pets. The website and chatbot were created as an example of how current technology can positively influence and contribute greatly to various associations. In an association such as Novogradiške njuške, the most important thing is the exchange of information, in order to enable the most successful operation, an internet page was created, which contains all important information about pets. Ultimately, the primary task would be to enable better communication, that is, an easier and faster exchange of information between two types of users, those who are looking for their new pets and those who are eager to give them up for adoption (Nacionalni repozitorij završnih i diplomskih radova ZIR) (Ritzer, 2005).

Key words: web page, chat-bot, MongoDB, React.js, Express.js, Node.js

Izvorni kod klijent dijela aplikacije: https://github.com/MarkoSolic/novogradiske-njuske-frontend

Izvorni kod server dijela aplikacije : https://github.com/MarkoSolic/novogradiske-njuske-backend

Sadržaj

1. Uvod

U današnje doba internet je jedna od osnovnih elemenata normalnog života. Internet je svjetski sustav međusobno povezanih računalnih mreža. Zahvaljujući razvoju informacijske i komunikacijske tehnologije, postao je osnova suvremene elektroničke komunikacije, a postupno dobiva i značenje vodećeg komunikacijskog medija današnjice. Dokaz tome bi bila tvrdnja da danas mnogi ljudi rade i žive od interneta. Internet se koristi u svakodnevici bilo to za gledanje današnje vremenske prognoze ili vijesti, kupovine ili naručivanje kod doktora, pored svega toga najčešće korištenje interneta ide u pretraživanje mnogih informacija koje su dostupne na njemu. Pojedinac je u mogućnosti pregledati samo mali komadić informacija koje se nalaze na internetu u svojem cijelom životu zato što količina tih informacija samo nastavlja rasti iz dana u dan. Svaki dan se stvaraju nove web stranice, nadopunjuju postojeće te brišu stare. Postoji i mnogo servisa koje ljudi izrađuju, a kojima je cilj olakšati ljudski svakodnevni život. S vremenom je postalo isplativo izraditi prijašnje navedene servise i web stranice zato što pružaju mogućnost velike zarade te danas najbogatije osobe imaju mnoge kompanije koje su bazirane na radu na internetu. Većina informacija koje se nalaze na internetu korisnici pronalaze u obliku web stranica. Web stranice su, moglo bi se reći lice interneta te ako netko ima želju prenijeti svoju informaciju ili uslugu preko interneta, nema puno drugih opcija osim kreiranja web stranice.

Cilj ovog rada je izraditi internetsku stranicu i chatbota za udrugu Novogradiške njuške. U sljedećem poglavlju objasniti će se proces razvoja internetske stranice, zatim navesti će se web tehnologije i alati korišteni za izradu internetske stranice. Objasniti će se implementacija koda koji se koristio unutar web stanice i koda koji je korišten za spajanje na bazu podataka. Također objasniti će se izrada samog chatbota i njegova povijest. Na kraju će biti nekoliko riječi za zaključak.

Izvorni kod klijent dijela aplikacije: https://github.com/MarkoSolic/novogradiske-njuske-frontend

Izvorni kod server dijela aplikacije : https://github.com/MarkoSolic/novogradiske-njuske-backend

2. Proces razvoja internetske stranice

Proces izrade web stranice ne počine kod početka pisanja koda. Prije nego što programeri počnu pisati kod web stranice oni moraju prikupiti relevantne polazne informacije o onom što žele izraditi. Proces prikupljanja informaciji varira ovisno o programeri ili tvrtki koja izrađuje web stranicu. Proces se odvija polazno od prikupljanja informacija, preko dizajnera i programera pa sve do nastavlja održavanja web stranice nakon inicijalnog završetka izgradnje.

2.1. Metodologije razvoja softvera

Postoji nekoliko metodologija za razvoj sustava, nekoliko najpoznatijih su (Saravanan, 2017):

- *Metodologija vodopada* (engl. Waterfall) je tradicionalna metoda te često se na nju gleda kao staromodnu. No bitno je razumjeti povijest i strukturu metodologije vodopada da bi se više cijenila fleksibilnosti novijih metodologija. Kreirana je u 1970- oj godini, metodologija vodopada zahtjeva veliku količinu strukture i dokumentacije prije početka projekta. Podijeljena je u korake, prvi korak je vitalan jer zahtjeva potpuno razumijevanje oboje programera i korisnika o ciljevima i zahtjevima projekta prije nego što projekt krene u izgradnju. Nakon tog koraka slijede koraci koji se većinom sastoje od određivanja zahtjeva za nadolazeći korak, analize tih zahtjeva, dizajna, implementacije, testiranja, izbacivanja promjena te održavanja. Svaki korak je početak za sljedeći te se koraci odrađuju sekvencijalno sve do kraja projekta.

Razvoj temeljen na značajkama(engl. Feature-Driven Development - FDD) je iterativan i inkrementalan pristup softverskom programiranju. Nastao je iz Agilne metodologije te je sličan metodologiji vodopada. Cilj ove metodologije je dostavljanje funkcionalnih softvera u čestim iteracijama te je fokusiran na čestu interakciju s klijentima što ga čini dobrim pristupom za manje timove programera. Značajke projekata su temeljni dijelovi FDD-a te se teži da svaka dva tjedna budu dostavljene nove značajke klijentima. FDD se sastoji od pet koraka, prvi je izgradnja osnovnog modela projekta, sljedeći je ispis svih značajki od kojih se projekt sastoji, zatim ide izrada planova za svaku od tih značajki te zadnja dva koraka su dizajniranje značajki i izgradnja značajki.

- *Agilna metodologija* (engl. Agile methodology) je razvijena kao odgovor ne fleksibilnoj vodopadnoj metodologiji. Ovaj pristup je dizajniran da se softver brze producira. Koristeći agilnu metodologiju, timovi stvaraju kratke sprintove ili iteracije čiji je cilj izbaciti funkcionalni softver što brže, ili barem radeći, testirajući oblik tog softvera. Metodologija se fokusira na suradnju tima, zajedno sa unutarnjim povratnim informacijama iz različitih odjela ili klijenata. Zadovoljstvo klijenata je najbitnije unutar agilne metodologije.

- *Ekstremno programiranje* (engl. Extreme programming - XP) je pristup koji se fokusira na proizvodnju što kvalitetnijeg softvera koristeći najbolje prakse unutar programiranja. Kao i agilnim pristupom, XP prati česta izbacivanja novih značajki softvera korisnicima.

2.2. Koraci u izgradnji web stranice

Proces izgradnje web stranice se sastoji od nekoliko međusobno povezanih koraka. Ovo su glavni koraci u procesu u kojima programeri surađuju s klijentima u cilju izrade što kvalitetnije web stranice. Koraci variraju ovisno o kompaniji ili programeru ali obično prate isti princip, Slika 1 prikazuje jedan oblik koraka koji se korite. Koraci koji će biti navedeni u ovom djelu su (Infolio, 2020):

1. Analiza
2. Dizajn
3. Programiranje
4. Optimizacija
5. Testiranje
6. Isporuka
7. Praćenje

Slika 1 Koraci u izgradnji web stranice

Izvor: autor rada

2.2.1. Analiza

Analiza je prvi korak izgradnje web stranica, u ovom koraku programeri intenzivno komuniciraju sa klijentima u cilju prikupljanja što više informacija. Osim toga što je to prvi korak, analiza je ujedno i najznačajniji korak. Programeri doznaju koje su specifikacije klijenata, koja je ciljana skupina ljudi koje klijent nastoji dohvatiti ili poslužiti. Doznaje se domena projekta i slično. Tijekom ove faze se doznaju tehničke specifikacije koje web stranica treba imati. Ispituju se i tehnologije koje moraju biti korištene, da li se mora koristiti CMS sustav, sustav za rezervaciju, web shop, online naplata i nadalje. Analizom dobivenih podataka stvara se tehnička specifikacija projekta (Aspekt.co, 2019).

2.2.2. Dizajn

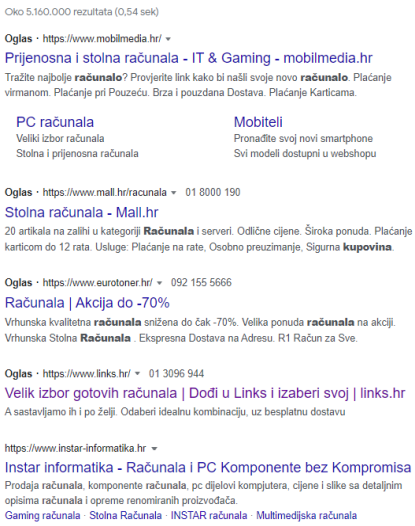
U ovoj fazi je naglasak na vizualni izgled web stranice. Prilikom dizajniranja web stranice obično se koristi tehnička specifikacija kao referenca za sve karakteristike, posebne korisničke zahtjeve, sve funkcionalnosti koje web stranca treba posjedovati i ostale bitne značajke. U nekim slučajevima se dizajn radi iterativno u smislu da se napravi prototip dizajna koji je predstavljen korisnicima te korisnici odluče što da se zadrži, a što da se odbaci. U smišljanje dizajna ulazi i funkcionalnost web stranice te korisnici kojoj je ta web stranica namijenjena. Osim web stranice često se izrađuju i logotipi koji će biti prikazani na danoj web stranici. Faza dizajniranja završava kada klijent bude zadovoljan sa dizajnom i kada dizajn zadovoljava sve tehničke specifikacije.

2.2.3. Programiranje

Nakon izrade dizajna, web stranica je spremna za izradu, drugim riječima za programiranje. Također se koristi tehnička specifikacija kao glavna referenca za izradu web stranice, ali uz tehničku specifikaciju se koristi i dizajn kao referenca za izradu dijela stranice koji se naziva frontend. Frontendu je osim izgleda bitno da li je web stranica statička ili dinamička, drugim riječima da li na stranici klijenti unose neke podatke, ako ne unose, radi se o statičkoj web stranici, u suprotnom se radi o dinamičkoj. Dinamičke web stranice većinom imaju i backend dio koji se sastoji od baze podatka i servisa. Servis se brine da komunikacija između frontenda i baze podataka radi kako je zamišljena. Faza programiranja u principu završava kada su implementirane sve funkcionalnosti i kada je izgrađen cijeli izgled web stranice. U ovoj fazi se najčešće koriste prijašnje spomenute metodologije kao što je vodopad, agilna i ostale.

2.2.4. Optimizacija

Korak optimizacije se u dosta često koraka preskače ili se smatra korakom 3 koji spada pod programiranje. Ovaj korak je vrlo važan u izradi kvalitetnih web stranica iz razloga što postoji mnogo aspekata koji utječu na brzinu učitavanja web stranice ili dohvaćanja podataka za web stanice. Većina korisnika nema namjeru čekati sporo učitavanje web stranica te iz tog razloga napuštaju i traže alternative, to nije idealan scenarij za kompanije ili individualce koji posjeduju navedene web stranice. Osim optimizacije brzine postoji i SEO optimizacija. SEO optimizacija je ukratko optimizacija web stranica da se prikazuju na višim rangovima unutar korisnikovih web preglednika. Glavni razlog tome je što većina ljudi posjećuje prvih 5 stranica koje vide što drugim riječima znači da tih top 5 stranica drže monopol nad određenim ključnim riječima. SEO optimizacija uključuje niz aktivnosti koje je potrebno poduzeti kako bi pretraživači što bolje rangirali web stranicu za relevantne ključne riječi. Mnogo je detalja koji utječu na rangiranje web stranice u pretrazi i svakako je važno da web stranica bude barem tehnički optimizirana. Osim SEO optimizacije, jako veliku pomoć nudi plaćanje Google pretraživaču da se stavi određena stranica na vrh liste, u obliku oglasa. Na drugu stranu, postoje i web stranice od firmi kojima SEO optimizacija nije toliko bitna iz razloga što je njima cilj da ih posjećuju samo korisnici koji unaprijed znaju za tu firmu i imaju potrebu koristiti neke od servisa koje im nudi web stranica te firme (Aspekt.co, 2019).



Slika 2 Google rezultat pretraživanja

Izvor: Google (2022)

2.2.5. Testiranje

Nakon završetka programiranja slijedi sljedeći korak koji se sastoji od testiranja napravljenog proizvoda prije nego što se preda korisniku na korištenje. Testiranje je još jedan jako bitan korak, ako ne i najbitniji iz razloga što ne poželjni problemi ne stvaraju samo poteškoće korisnicima nego i smanjuju ugled programera ili tima koji je napravio tu stranicu. Faza testiranja u dosta slučajeva traje čak i duže od faze programiranja. Razlog tome je što postoji jako mnogo pristupa i načina koji mogu prouzrokovati neželjeno ponašanje web stranice. U slučajevima u kojima su web stranice malo kompleksnije, unutar timova postoje programeri koji se specifično bave samo testiranjem. Nakon što otkriju problem, javljaju drugim programerima te ga oni ispravljaju. Postoje i neki početni standardi koji pomažu testirati web stranice te brzo otkriti one očigledne probleme. Neki od tih alata su W3C validator koji detaljno prolaze kroz HTML dio web stranice i javljaju eventualne greške. Osim testiranja funkcionalnosti, testira se i prijašnje izrađena SEO optimizacija i brzina web stranice u mnogim različitim uvjetima. Nakon što se poprave većina testova ide se na sljedeći korak.

2.2.6. Isporuka

Kada se ustanovi da nema daljnjih problema ide se na korak koji se zove isporuka. Značenje toga je da se web stranica radi dostupna novim korisnicima na internetu te započinje njezin namijenjen rad. Nakon uspješnog lansiranja završava faza lansiranja te kreće zadnja faza.

2.2.7. Održavanje

Nakon što stranica dođe na internet, programeri su zaduženi pratiti rad web stranice, to je neka vrsta garancije koja govori da će programeri popraviti web stranicu u slučaju da nešto krene po krivu. Osim praćenja za problemima programeri nastavljaju komunikaciju sa klijentom te ovisno o željama klijenta ponekad i implementiraju nove funkcionalnosti. Te nove funkcionalnosti obično prolaze svih sedam koraka ponovno te po završetku zamjenjuju postojeću web stranicu sa nadograđenom. Nakon toga se proces nastavlja ponavljati u krug (50languages).

3. Web tehnologije i alati

Sljedeće navedene tehnologije su korištene tijekom izrade ove web stranice. Za izradu dizajna web stranice se koristio programski jezik JavaScript, specifično React okvir koji pomaže izradi dizajna iz razloga što je jedan od najpopularnijih okvira danas. Za dio koji komunicira sa bazom podataka se koristio Express.js paket koji je isto pisan u JavaScript-u te naposljetku, za bazu podatka se koristio MongoDB. Programiranje se vršilo u Visual Studio Code-u.

3.1. JavaScript

JavaScript je jedan od najpopularnijih programskih jezika, većinom se piše kao skriptni jezik ali podržava i pisanje u objektno-orijentiranom obliku. Popularan je među početnicima zbog pristupačnosti i opće poznate popularnosti. Većina pretraživača koriste JavaScript za prikazivanje sadržaja na web stranicama, specifično upravljanje kako se dinamičke stranice ponašaju. Slika 3 prikazuje primjer JavaScript komada koda unutar ove web stranice (Nacionalni repozitorij završnih i diplomskih radova ZIR).

```
3  const auth = async (req, res, next) => {
4    try {
5      const token = req.headers.authorization.split(" ")[1];
6      const isCustomAuth = token.length < 500;
7
8      let decodedData;
9
10     if (token && isCustomAuth) {
11       decodedData = jwt.verify(token, "test");
12
13       req.userId = decodedData?.id;
14     } else {
15       decodedData = jwt.decode(token);
16
17       req.userId = decodedData?.sub;
18     }
19
20     next();
21   } catch (error) {
22     console.log(error);
23   }
24 };
```

Slika 3 JavaScript kod

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

3.2. React

React je najpoznatiji okvir za izgradnju korisničkog sučelja web stranice, osmišljen od strane poznate Facebook kompanije te kasnije mnogih drugih programera. Omogućuje oblikovanje kompleksnih sučelja spajajući manje individualne komponente u jednu veliku cjelinu. Umjesto klasičnog HTML oblika, React se piše pomoću JSX ekstenzije koja je osmišljena da nalikuje HTML-u ali se zapravo ispod nalazi normalan JavaScript. Glavne prednosti React-a su da je brz, fleksibilan, skalabilan i najbitnije relativno jednostavan za naučiti ako postoji neko osnovno predznanje JavaScript-a i HTML-a (Pandit, 2021).

```
26          <Grid item xs={12} sm={7}>
27            <Posts setCurrentId={setCurrentId}></Posts>
28          </Grid>
29          <Grid item xs={12} sm={4}>
30            <Form currentId={currentId} setCurrentId={setCurrentId}></Form>
31          </Grid>
```

Slika 4 React kod

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

3.3. MongoDB

MongoDB je objektno-orjentirana, jednostavna i skalabilna NoSQL baza podataka. Bazira se na dokumentima kao oblik spremanja podataka. Dokumenti se spremaju unutar kolekcija umjesto redova i stupaca kao u starijim relacijskim bazama podataka. Glavna motivacija takvog načina rada su bile visoke performanse i automatska skalabilnost. MongoDB je poprilično jednostavan za korištenje i instaliranje. Sintaksa u kojoj se mogu dokumenti spremati u kolekcije je JSON ili BSON, te je MongoDB dostupan na većini operativnih sustava (Botelho, 2020).

3.4. Node.js

Node.js je cross platforma na server strani izrađena u JavaScriptu, glavna zadaća je omogućiti pokretanje i izvođenje samostalnog programa na vlastitom računalu. Ryan Dahl je razvio Node.js 2009. godine. Izgrađena je na Chromeovom Javascript vremenu izvođenja radi lakše izrade skalabilnih i brzih mrežnih aplikacija. Node.js nudi moćnu biblioteku različitih JavaScript modula što nam u velikoj mjeri pojednostavljuje razvitak web-aplikacija koje upotrebljavaju Node.js (esprints.fri-lj.si) (Tvrdinić, 2020).

3.5. Express

Express je fleksibilni okvir web-aplikacije Node.js koji pruža skup značajki za razvoj web i mobilnih aplikacija. Omogućuje brzi razvoj web-aplikacija temeljenih na Node.js-u. Slijedi nekoliko osnovnih značajki Express okvira su Omogućuje postavljanje međusoftvera (engl. Middleware) kako bi odgovarao na HTTP zahtjeve. Definira tablicu ruta koje se koriste za izvođenje različitih radnji na temelju HTTP metoda i URL-a. Definira tablicu ruta koje se koriste za izvođenje različitih radnji na temelju HTTP metoda i URL-a. Express.js je okvir koji je na poslužitelju web-aplikacije Node.js. Posebno je dizajniran za izradu aplikacija s jednom stranicom, više stranica i hibridnih web-aplikacija. Postao je standardno razvojno okruženje na poslužiteljskoj strani za Node.js. Express je dio MEAN arhitekture koji se nalazi na strani poslužitelja (Tvrdinić, 2020).

3.6. Material UI

Material UI kreiran od strane američke multinacionalne korporacije „Google“ je React biblioteka koja omogućuje korištenje komponenti za kreiranje korisničkog sučelja. Sadrži mrežni raspored koji nudi veliku pomoć oko pozicioniranja različitih dijelova stranice i omogućuje skalabilnost stranice. Samim time štedi vrijeme programerima jer ne moraju sav kod pisati od početka. Neke od komponenta koji su upotrebljavanje u internetsko stranici „Novogradiške njuške“ su Container, Grid, Typography, Avatar (Vertal,2021).

4. Izrada internetske stranice za udrugu Novogradiške njuške

Za potrebe završnog rada izrađena je internetska stranica Novogradiške njuške, svrha stranice je omogućiti korisnicima da pronađu određenog kućnog ljubimca kojega bi htjeli udomiti ili objaviti post o kućnom ljubimcu kojega bi htjeli dati na udomljavanje. U nadolazeća dva pod poglavlja ćemo detaljnije objasniti kod tako da ćemo u par rečenica opisati pojedinu komponentu.

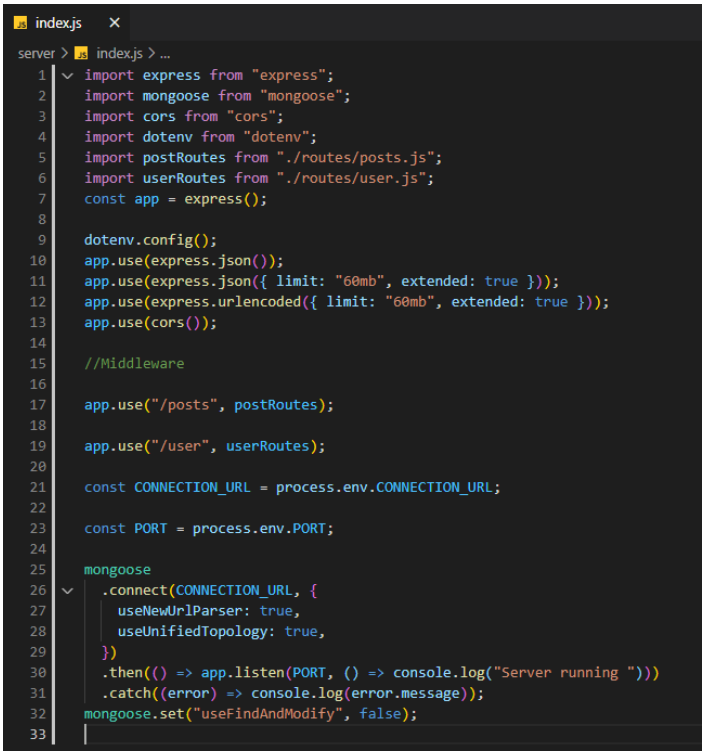
Back-End

Back-end ili server strana je dio aplikacije koje korisnik ne može vidjeti. Zaslužan je za pohranu i organizaciju podataka, to jest omogućuje komunikaciju između baze podataka i internetskog preglednika. Za izradu internetske stranice na server strani korišten je Node.js, baza podataka MongoDB, razvojni okvir Express.js te paket naziva dotenv pomoću kojeg se spremaju varijable koje su potrebne za pravilan rad aplikacije ali nisu dostupne vanjskim korisnicima. Također varijable se mijenjaju u slučaju da se aplikacija pokreće lokalno na računali ili na vanjskom serveru.

Ulazna datoteka

Serverska strana kreće od ulazne datoteke *index.*js , dodani su dodatni paketi: dotenv čija je svrha spremanja varijabla okruženja, neke od tih varijabla su port i url koji omogućuje spavanje na bazu, razlog zbog kojeg se koristi paket dotnev je da navedene varijable ne budu vidljive tokom preuzimanja ili korištenja internetske stranice od strane korisnika već su vidljive samo programeru. CORS (Cross-Origin Resource Sharing) HTTP header koji omogućuje preuzimanje resursa sa različitih izvora, dopušta da se nedostupni resursi, a koji se nalaze na internetskoj stranici, zahtijevaju od druge domene koja je van one iz koje je zahtjev za resursom potekao. Mongoose, paket za MongoDB koji omogućuje odnos između podataka i provjeru sheme, te u ovome slučaju koji je prikazan na slici 8. omogućuje spajanje na bazu i ispis pogreške ako je došlo do greške

tokom spajanja. Bitan dio koji se nalazi u ulaznoj datoteci index.js su rute, u ovom slučaju postoje dvije glave rute za objave i korisnika (sr.wikipedia.org).



Slika 5 Datoteka index.js

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Rute

Back-end kreće od ulazne datoteke index.js Slika 5. ona je prva na redu kada korisnik pošalje zahtjev na server, neovisno o tome koja je ruta pozvana. Zadaća datoteke indeks.js je da pročita rutu u pozivu i ovisno o tome pozove odgovarajuću funkciju koja je predodređena pozvanoj ruti. Ako poziv u sebi sadrži prefiks „/posts“ , poziva se jedna funkcija od prikazanih funkcija na Slici 10. od 14. do 18. linije. Koja će se funkcija pozvati ovisi o tome koji je drugi parametar unutar adrese. Rute smiju imati potpuno istu adresu samo ako sadrže različite pozive. Najčešći pozivi su GET, POST, PATCH, PUT i DELETE, primjer ovome su rute na liniji 14. i 15. Slika 7. adresa

im je ista ali su im pozivi različiti. Dok u slučaju da poziv u sebi sadrži prefiks „/users“, pozivaju se funkcije unutar datoteke users.js Slika 9. linija 6. i 7. i ovisno o adresi poziva se funkcija „signup“ ili „signin“. Tako da bi se moglo reći da nam datoteka služi kao usmjerivač ruta za navigaciju korisnika između stranica u aplikaciji (FPZG repozitorij).

```
app.use("/posts", postRoutes);
app.use("/user", userRoutes);
```

Slika 6 Glavne rute

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Post rute

Prilikom poziva rute, pozivaju se određene funkcije koje rješavaju zadatke ovisno o tome koju je rutu korisnik pozvao i koja metoda je korištena (GET,POST,PATCH,DELETE), važno je napomenuti da je za uspješno objavljivanje, uređivanje, brisanje i „lajkanje“ potrebna je autentifikacija korisnika Slika10. Datoteka posts.js sadrži navedene funkcije:

- getPosts – uzima postojeće postove iz baze podataka
- createPost – kreira novi post
- updatePost – omogućuje ažuriranje postojećih postove
- likePost – omogućuje ocjenjivanje posta
- deletePost- briše post

posts.js

server > routes > posts.js > ...

```
1   import express from "express";
2
3   import {
4     getPosts,
5     createPost,
6     updatePost,
7     likePost,
8     deletePost,
9   } from "../controllers/posts.js";
10
11  const router = express.Router();
12  import auth from "../middleware/auth.js";
13
14  router.get("/", getPosts);
15  router.post("/", auth, createPost);
16  router.patch("/:id", auth, updatePost);
17  router.delete("/:id", auth, deletePost);
18  router.patch("/:id/likePost", auth, likePost);
19
20  export default router;
21
```

Slika 7 Datoteka posts.js

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Funkcija likePost Slika 8. služi za ocjenjivanje postojećih postova, u konstantnoj varijabli se pohranjuju podatci o korisniku iz parametara, to jest pohranjuje se korisnikov identitet. Zatim u prvoj if selekciji se šalje poruka „Unauthenticated“ u slučaju da id ne postoji. Druga selekcija se izvršava ako ne postoji post sa korisničkim identitetom i ispisuje odgovarajuću poruku. U varijabli post su pohranjeni podatci o postu koji je korisnik kliknuo, te u varijablu indeks pohranjuje korisnikov identitet, u slučaju da taj identitet već postoji. Što znači da je korisnik prethodno lajkao stranicu, lajk se briše, a ako ne postoji dodaje se. Na kraju se šalje status 200 to jest da je post ažuriran.

```
71  export const likePost = async (req, res) => {
72    const { id } = req.params;
73
74    if (!req.userId) {
75      return res.json({ message: "Unauthenticated" });
76    }
77
78    if (!mongoose.Types.ObjectId.isValid(id))
79      return res.status(404).send(`No post with id: ${id}`);
80
81    const post = await PostMessage.findById(id);
82
83    const index = post.likes.findIndex((id) => id === String(req.userId));
84
85    if (index === -1) {
86      post.likes.push(req.userId);
87    } else {
88      post.likes = post.likes.filter((id) => id !== String(req.userId));
89    }
90    const updatedPost = await PostMessage.findByIdAndUpdate(id, post, {
91      new: true,
92    });
93    res.status(200).json(updatedPost);
94  };
95
96  export default router;
97
```

Slika 8 Funkcija likePost

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

User rute

User datoteka ima sličan zadatak kao i Posts datoteka, ovisno o tome koja je ruta pozvana, koristit će se prikladna funkcija. Kada poziv u sebi sadrži prefiks „/signin“ poziva se funkcija za prijavu korisnika Slika 11., ako sadrži prefiks „/signup“ poziva se funkcija za registraciju Slika 10. Način rada navedenih funkcija biti će objašnjen u naredna dva poglavlja.

Slika 9 Datoteka user.js

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Signup omogućuje registraciju korisnika, nakon izrade računa korisnik može objavljivati, uređivat i lajkati postove. U funkciji se uzimaju podatci: email, password, ime i prezime koje je korisnik unio na klijent strani aplikacije, zatim se provjerava uneseni email i ako postoji isti takav u bazi podataka, proces se obustavlja i šalje se status 400 sa porukom koja kaže da korisnik već postoji. Ako uneseni email ne postoji, lozinka se kriptira i zajedno sa ostalim podatcima izrađuje se dokument u MongoDB bazi. Nakon toga se izrađuje JWT token i korisnik je tada prijavljen na stranicu. JWT token (JSON Web Token) je kompaktan i samostalan način za sigurno prenošenje informacija između više stranka u obliku JSON objekta. Služi kako bi se potvrdila autentičnost korisnika tokom korištenja raznih aktivnosti na stranici (Sveučilištu u Beogradu, Fakultetu organizacijskih znanosti).

```
export const signup = async (req, res) => {
  const { email, password, firstName, lastName } = req.body;

  try {
    const oldUser = await UserModal.findOne({ email });

    if (oldUser)
      return res.status(400).json({ message: "User already exists" });

    const hashedPassword = await bcrypt.hash(password, 12);

    const result = await UserModal.create({
      email: email,
      password: hashedPassword,
      name: `${firstName} ${lastName}`,
    });

    const token = jwt.sign({ email: result.email, id: result._id }, secret, {
      expiresIn: "1h",
    });

    res.status(201).json({ result, token });
  } catch (error) {
    res.status(500).json({ message: "Something went wrong" });

    console.log(error);
  }
};
```

Slika 10 Funkcija signup

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Funkcija „signin“ omogućuje prijavu korisnika. U varijabli „oldUser“ pohrani se korisnički email koji se nalazi u bazi podataka, pod uvjetom da je korisnik prethodno izradio račun. U slučaju da ne postoji, to jest da se korisnik nije prethodno registrirao šalje se status 404. i poruka „User doesn't exist“. Protivno, ako postoji email u bazi podatak, kreće provjera lozinke tako da se usporedi kriptirana lozinka iz baze podatak (koja se prije toga dekriptira) i lozinka iz body-a odnosno iz inputa koja je unesena od strane korisnika. Zatim ako se lozinke ne podudaraju šalje se status 400 sa porukom „Invalid credentials“, ako se podudaraju izrađuje se JWT token koji ističe u roku od sat vremena. Token se šalje na preglednik i sprema u web pohranu.

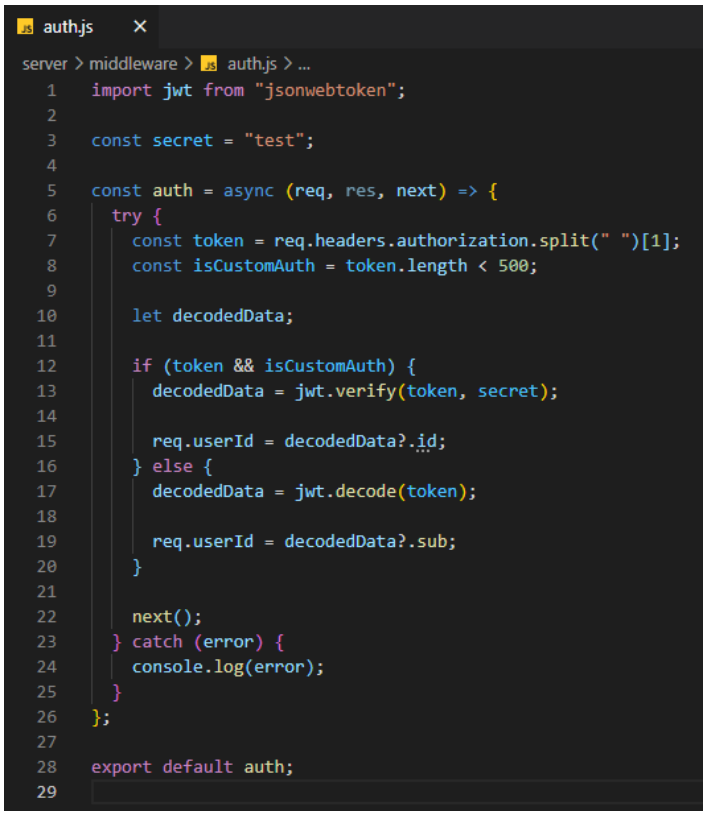
```
export const signin = async (req, res) => {
  const { email, password } = req.body;

  try {
    const oldUser = await UserModal.findOne({ email });

    if (!oldUser)
      return res.status(404).json({ message: "User doesn't exist" });

    const isPasswordCorrect = await bcrypt.compare(password, oldUser.password);

    if (!isPasswordCorrect)
      return res.status(400).json({ message: "Invalid credentials" });

    const token = jwt.sign({ email: oldUser.email, id: oldUser._id }, secret, {
      expiresIn: "1h",
    });

    res.status(200).json({ result: oldUser, token });
  } catch (err) {
    res.status(500).json({ message: "Something went wrong" });
  }
};
```

Slika 11 Funkcija signin

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Autentifikacija

Prije vršenja aktivnosti ocjenjivanja, ažuriranja, izrade i brisanja objave potrebno je provjeriti autentičnost korisnika. Autentičnost se provjerava dekriptiranjem tokena koji se nalazi na zaglavlju poruke, potom od de kriptiranih podataka se uzima korisnikov identitet koji se dobije nakon korisnikove prijave. Nakon što je korisnik uspješno autoriziran, nastavlja se odrađivati funkcija koja je bila prvobitno pozvana. Može se primijetiti da je programski kod ugniježđen nečim što se zove try – catch blok, njegova svrha je, u slučaju da se nešto pravilno ne izvrši unutar try sekcije, proces izvođenja se završava, te ulazi u catch sekciju, u kojoj se najčešće ispisuju pogreške.



Slika 12 Autentifikacija

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

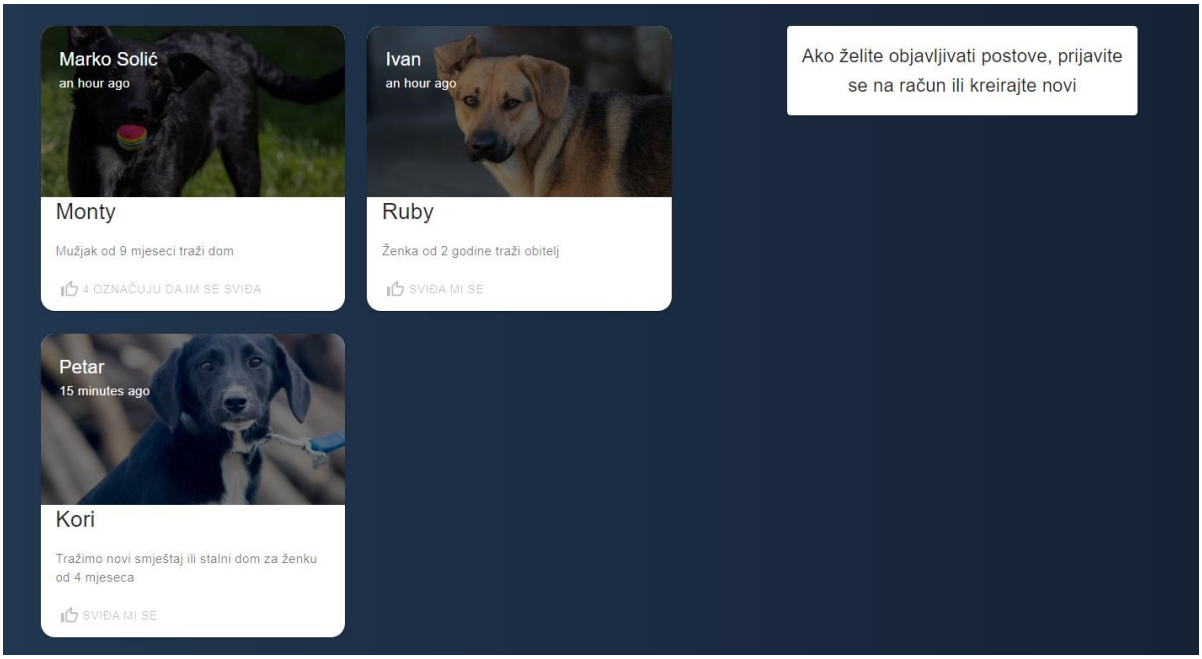
Front-End

Ovo poglavlje opisuje korisničku stranu internetske stranice, to jest izgled stranice i njezinu funkcionalnost. Stranica je izrađena pomoću React razvojnog okvira i biblioteke Material UI koji omogućuju vizualni prikaz stranice i njezinih komponenti, definira njihovu strukturu i sadržaj te omogućava pravilno prikazivanje. Glavni zadatak internetske stranice je olakšati korisnicima proces oko udomljavanja kućnih ljubimaca, to jest pružiti im potrebe informacije o pojedinom ljubimcu kako bi lakše odabrali sebi prikladnu životinju. Da bi navedeno bilo moguće izrađene su sljedeće komponente: traka za navigaciju, podnožje, Post i Forma, te Auth komponenta za registraciju i prijavu (FPZG repozitorij).

.

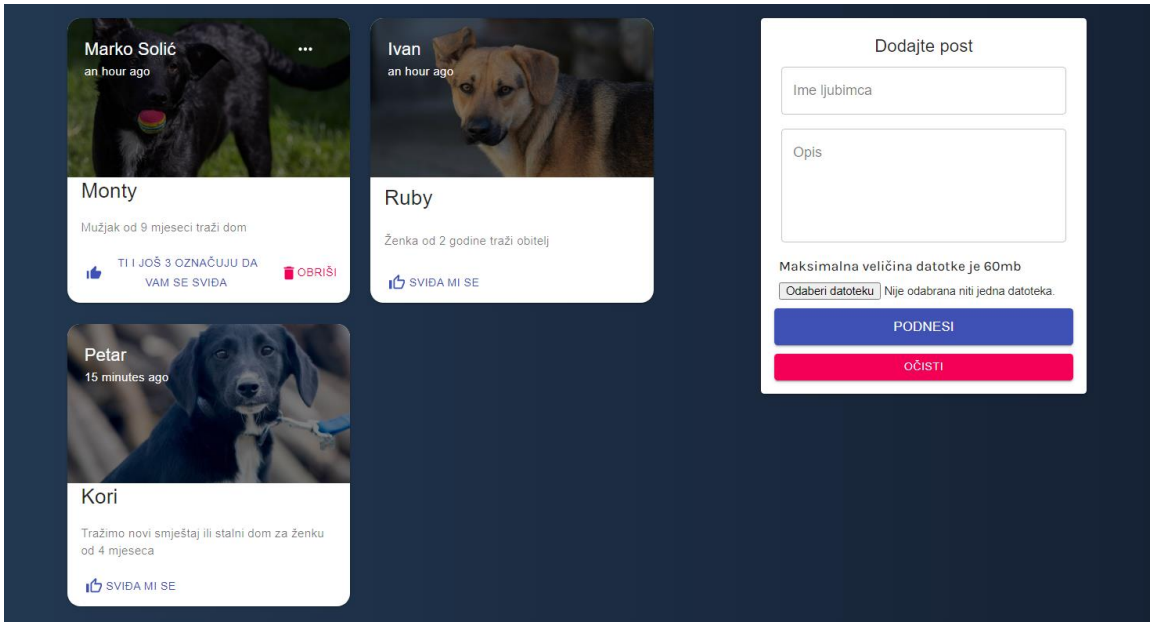
Početna stranica

Na početnoj stranici koja započinje u App komponenti koja u sebi poziva razne druge komponente. Izgled početne stranice se može vidjeti na Slici 16. Izgled stranice se razlikuje ovisno o tome ako je korisnik prijavljen ili nije. Slika 13. prikazuje izgled početne stranice kada korisnik nije prijavljen. U slušaju da se korisnik prijavi, omogućene su mu dvije glavne funkcionalnosti, a to su objava posta i „lajkanje“ postova. Slika 14. prikazuje izgled početne stranice kada je korisnik prijavljen (FPZG repozitorij).



Slika 13 Izgled početne stranice prije prijave ili registracije korisnika

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021



Slika 14 Izgled početne stranice nakon prijave ili registracije korisnika

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Nakon uspješne registracije ili prijave korisnika, omogućeno je ispunjavanje forme Slika 19. tj. objava nove objave. Za uspješno objavljenu objavu korisnik mora unijeti ime ljubimca, kratki opis o ljubimcu i priložiti sliku, nakon toga klikom na dugme „podnesi“ objava je objavljena. Također korisniku je omogućeno „lajkanje“ objave, brisanje i uređivanje objava koje je korisnik objavio to jest koji su u njegovom vlasništvu. Nakon prijave, dugme „Prijavi se“ zamjenjuje se sa dugmetom „Odjavi se“ i iznad toga je dodana profilna slika.

Slika 15 Izgled trake za navigaciju prije prijave ili registracije korisnika

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Slika 16 Izgled trake za navigaciju nakon prijave ili registracije korisnika

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Promjena izgleda navbar-a postiže se definiranjem if uvjeta Slika 18. od 80. do 110. linije, gdje se provjerava stanje konstante „user“ Slika 17. koja uzima podatke pod nazivom „profile“ iz web pohrane koje se dobiju nakon korisnikove prijave ili registracije. Potom raščlanjuje podatke iz JSON formata u JavaScript objekt, taj JS objekt se koristi, za prikaz korisnikovog imena i avatara. Zatim u if-u se provjerava stanje, ako u web pohrani postoje podatci, vrijednost varijable „user“ se popunjava tim podatcima, u suprotnom „user“ ima vrijednost „null“. U slučaju da je stanje „user“ varijable različito od „null“, ispisuje se prvi dio if uvjeta (od 80. do 99. linije) to jest ispisuje se dugme „Odjavi se“, korisnikovo ime i avatar. Suprotno, ispisuje se drugi dio (od 101. do 109.) to jest prikazuje se samo dugme „Prijavi se“.

```
const [user, setUser] = useState(JSON.parse(localStorage.getItem("profile")));
```

Slika 17 Varijabla user

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

```
<Toolbar className={classes.toolbar}>
  {user ? (
    <div classes={classes.profile}>
      <Avatar
        className={classes.purple}
        alt={user.result.name}
        src={user.result.imageUrl}
      >
        {user.result.name.charAt(0)}
      </Avatar>
      <Typography className={classes.userName} variant="h6">
        {user.result.name}
      </Typography>
      <Button
        variant="contained"
        className={(classes.buttonEdit, classes.logout)}
        color="secondary"
        onClick={logout}
      >
        Odjavi se
      </Button>
    </div>
  ) : (
    <Button
      className={classes.buttonEdit}
      component={Link}
      to="/auth"
      variant="contained"
      color="primary"
    >
      Prijavi se
    </Button>
  )}
</Toolbar>
```

Slika 18 if uvjet za navbar

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Slika 19 Izgled forme za izradu posta

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Ispunjavanjem forme Slika 19. korisniku se omogućava stvaranje objave, slično kao i u navbar komponenti u if uvjetu se provjerava dali korisnik postoji u bazi podataka te da li je trenutačno prijavljen ,u slučaju da postoji, forma je prikazana, suprotno, forma se ne prikazuje. Nakon što su sva polja unutar forme ispunjena, klikom da dugme „Podnesi“ koji je tipa „submit“ poziva se funkcija „handleSubmit“. Razlog zašto se poziva funkcija „handleSubmit“ je zato što joj je dodijeljen obrađivač događaja „onSubmit“ koji je standardni dio React-a. Funkcija provjerava da li postoji trenutni identitet od objave koji se generira kada prvi puta izradimo objavu, ako ne postoji stvaramo novu objavu, a ako postoji mijenjamo staru objavu sa novo unesenim podatcima. Prije nego što se kreira nova objava ili promjeni stara, odradi se funkcija „dispatch“. Dispatch ili useDispatch je funkcija unutar Redux-a, koju koristimo kada želimo napraviti promjenu na „state-u“, tako da „dispatchamo“ akciju koja u sebi sadrži tip, u ovom slučaju tip je „Create“ sa „payload-om“ koji u sebi sadrži podatke o objavi. Zatim se podatci provjeravaju, te ovisno o tipu, radi odgovarajuću promjenu i „data“ se izmjenjuje na komponenti gdje se i prikazuje (FPZG repozitorij).

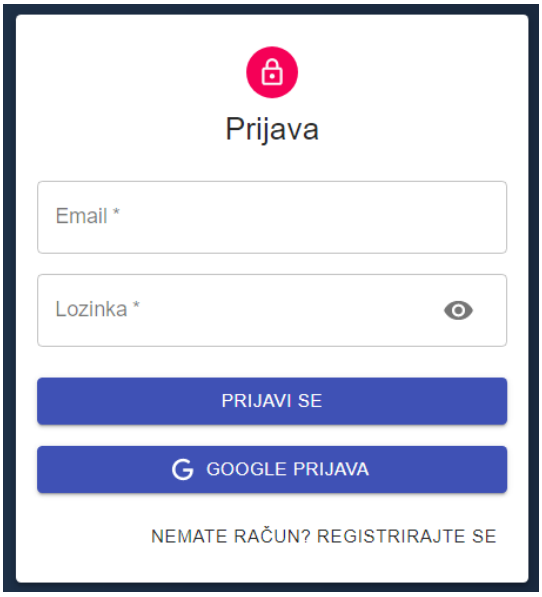
```
const handleSubmit = async (e) => {
  e.preventDefault();

  if (currentId === null) {
    dispatch(createPost({ ...postData, name: user?.result?.name }));
    clear();
  } else {
    dispatch(
      updatePost(currentId, { ...postData, name: user?.result?.name })
    );
    clear();
  }
};
```

Slika 20 Funkcija handleSubmit

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Prijava i registracija

Forme za prijavu i registraciju nalaze se u istoj datoteci i omeđene su „if“ uvjetima pomoću kojih se postiže promjena forme sa prijave na registraciju, ili obrnuto. Promjena forme se odrađuje pomoću funkcije „switchMode“ i „useState“ “hook-a“. Funkcija „switchMode“ se okida klikom na zdanje dugme u formi i mijenja trenutno stanje varijable „isSignup“ koja je definirana u „useState-u“. Navedena varijabla je tipa „bool“, te ako je stanje „false“ prikazuje se forma za prijavu, a ako je „true“ prikazuje se produžena forma, to jest registracija.

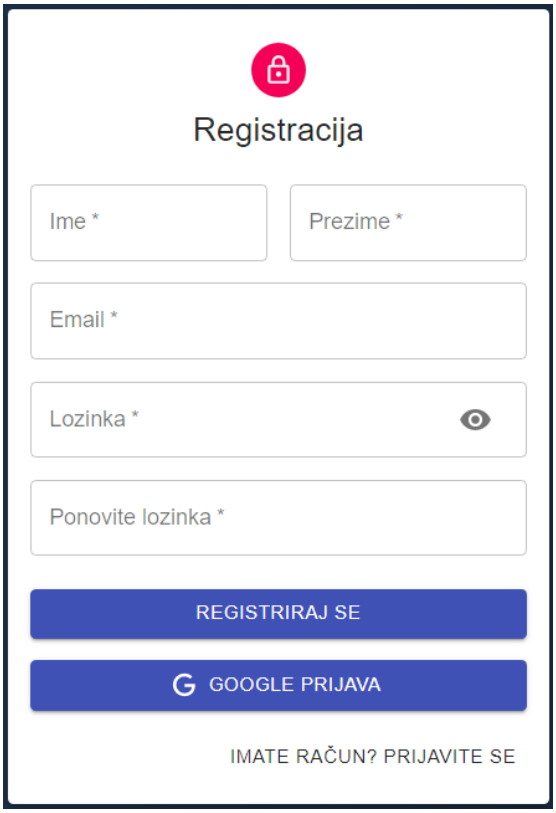


Slika 21 Forma za prijavu

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

Prijava korisnika kreće klikom na dugme „Prijavi se“ koje se nalazi u navbar-u, zatim se prikazuje forma za prijavu Slika 21, gdje je potrebno ispuniti dva polja a to su email i lozinka kako bi se korisnik uspješno prijavio, uzimajući u obzir da je prethodno uspješno obavio registraciju. Standardna prijava se izvršava pomoću funkcije „handleChange“ koja puni prvobitno prazan „state“ sa unesenim vrijednostima u „inputu“ pomoću React „hook-a“ naziva „useState“. Nakon toga

funkcija „handleSubmit“ provjerava što je potrebno poduzeti sa novo unesenim podatcima, ako je ispunjena forma za prijavu poziva se funkcija „singin“ u suprotnom ako je ispunjena forma za registraciju poziva se funkcija „signup“.



Slika 22 Forma za registracija

Izvor: Adrian Hajdin, JavaScript Mastery (Youtube kanal), 2021

5. Chatbot

Chatbot je aplikacija koja se većinom koristi za izvođenje internetskog ili online razgovora. Dizajnirani su da simuliranju ljudsko ponašanje u raznim situacijama tijekom razgovora. Chatbot sistemi tipično zahtijevaju konstantno poboljšavanje i testiranje te usprkos svemu tome većina chatbotova nisu u mogućnosti adekvatno razgovarati s ljudskim korisnicima ili proći poznati Turing test. O Turing testu će se govoriti kasnije, prije toga će se ukratko proći o povijesti chatbota (TehnoKlik, 2021).

5.1. Povijest

Prvi ikad chatbot je osmišljen čak prije nego što su se osobna računala lansirala na opće tržište. Osmislio ga je znanstvenik Joseph Weizenbaum iz MIT -a 1966. godine. Ime prvog chatbota je bilo Eliza, Slika 23 ispod prikazuje Elizu. Eliza je proučavala ključne riječi koje su joj dane te bazirano na tim riječima vraćala odgovore. Taj pristup generiranja odgovora se i danas koristi unutar mnogih chatbotova, uključujući i chatbot koji je prisutan u ovom radu. Sljedeći chatbot je bio imenovan Parry, njega je napisala psihologinja Kenneth Colby iz Stanford Sveučilišta. Ideja ovog chatbota je bila da simulira osobu koja ima šizofreniju. Nakon njega je došla A.L.I.C.E koju je osmislio Richard Wallace. A.L.I.C.E je osvojila Loebner-ovu nagradu tri puta, ali je i onda pala Turingov test (Sakina Vohra, 2020).

Slika 23 Eliza

Izvor: Siddhartha, 2021

5.2. Turingov test

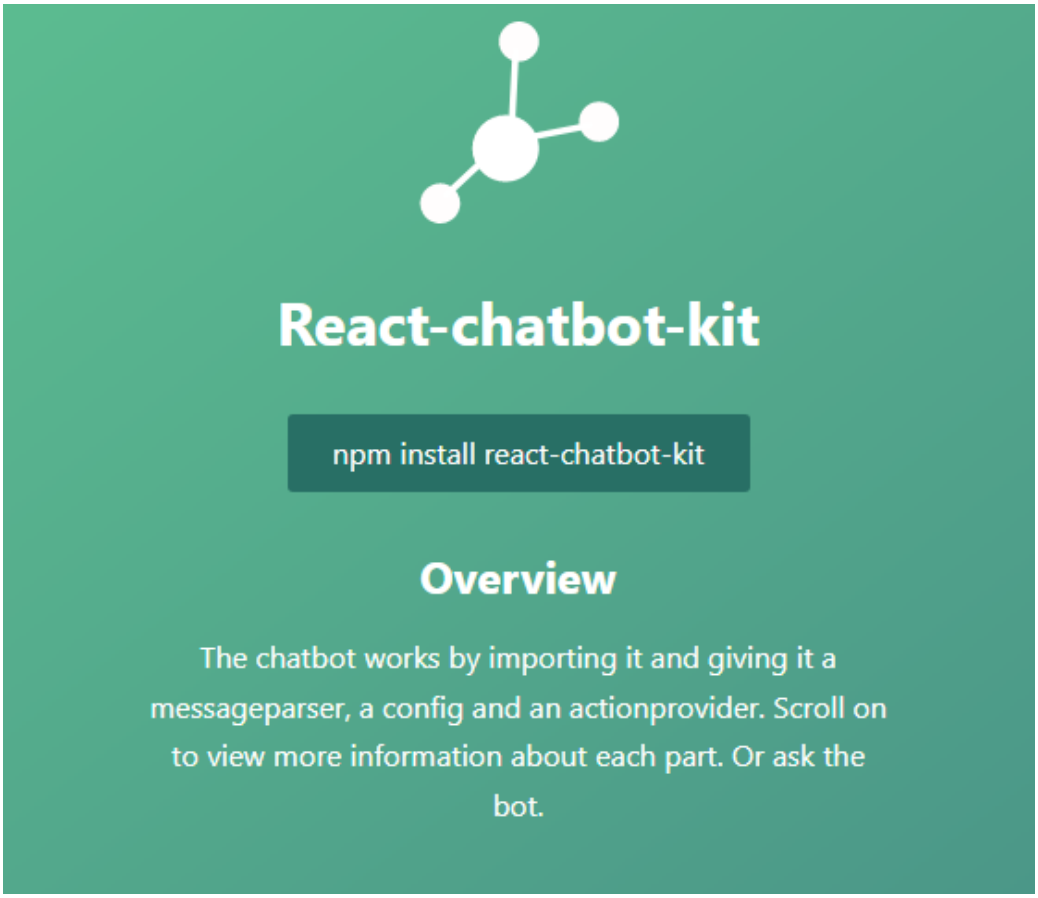
Turingov test je poznata metoda kojoj je cilj otkrivanje da li umjetna inteligencija ima mogućnost razmišljanja kao ljudsko biće. Test je nazvan po Alan Turingu, tvorcu Turing testa i engleskom računalnom znanstveniku, kripto analitičaru i matematičaru. Turing je predložio ideju da se za računalo može reći da posjeduje umjetnu inteligenciju samo u slučaju ako je u mogućnosti oponašati ljudske odgovore pod posebnim uvjetima. Originalni Turnig test zahtjeva tri terminala, svaki od njih fizički odvojeni od ostala dva. Jednim terminalom bi upravljalo računalo dok bi druga dva terminala upravljali ljudi. Tijekom testa jedna bi osoba imala poziciju ispitivača dok bi računalo i drugi čovjek imali uloge osoba koje odgovaraju na postavljana pitanja. Ispitivač ispituje ispitanike unutar specifične domene i pod specifičnim kontekstom i formatom. Nakon određenog vremenskog razdoblja ili određenog broja pitanja, ispitivač je pitan da odabere koji je od ispitanika bio čovjek, a koji računalo. Test bi se ponavljao mnogo puta. Ako ispitivač točno pogodi u više od pola slučajeva, računalo nije uspjelo proći Turingov test te se ne smatra umjetnom inteligencijom. Danas postoji mnogo varijacija Turing testa koje se koriste svakodnevno. Neki od tih varijacija su obrnuti Turing test ili više poznatiji kao CAPTCHA. U CAPTCHI ljudsko biće pokušava uvjeriti računalo da je ono uistinu ljudsko biće, a ne drugo računalo. Druga varijacija je totalni Turingov test. U ovom slučaju ispitanik ispituje perceptivne sposobnosti te sposobnosti manipuliranja objektima. Zadnji od varijacija bi bio test minimalnog inteligentnog signala u kojem se daju samo da/ne odgovori. Osim navedenih, postoji još mnogo drugih naprednijih varijacija Turing testa koje su većinom izmišljene zato što je stari Turingov test imao dosta grešaka (Physics World, 2021).

Slika 24 Alan Turing

Izvor: Vishnu, 2021

6. Izrada Chatbot-a za udrugu Novogradiške njuške

Za izradu Chatbota- korišten je paket „React-chatbot-kit“ [18] koji je u vlasništvu Fredrika Oseberga. Fredrick objavljuje Youtube videa na kanalu imena „Fredrik Oseberg Coding Lessons“ u kojem podučava gledatelje o osnovama web programiranja, kao što su HTTP protokoli, API – ovi, rad servera, chatbot i mnogi drugi. Chatbot se bazira na tri glavna dijela potrebni za ispravan rad, a to su config, MessageParser i ActionParser, u sljedećem poglavlju biti će navedene informacije o navedenim dijelovima.



Slika 25 React-chatbot-kit

Izvor: Oseberg, 2020

6.1. Postupak izrade

Prvi korak je instalacija paketa, instalacija se postiže upisom navedene komande koja je prikazana na slici Slika 26. (npm install react-chatbot-kit) . Pošto je chatbot potreban samo na klijent (engl. Client) dijelu web stranice, dovoljno ga je samo tu instalirati, nakon toga uključujemo chatbot u App.js gdje nam se nalaze sve komponente za prikaz na stranici, a to je App.js. Također u App.js-u izrađujemo dugme, pomoću kojega ćemo omogućiti prikaz i skrivanje chatbota. Navedeno se postiže uz pomoć „useState hooka“ i uvjetne naredbe , gdje nam je početna vrijednost „true“ što znači da je po zadamom chatbot prikazan na stranici, te sa klikom na dugme vrijednost se mijenja na protivnu vrijednost prije klika, početna vrijednost je „true“ i nakon klika vrijednost će se promijeniti u „false“ te chatbot postaje sakriven. Prema naputku iz react-chatbot-kit dokumentacije, izrađujemo tri datoteke imena MessageParser, ActionProvider i config koje uključujemo u App.js.

```
npm install react-chatbot-kit
```

Slika 26 Komanda za instalaciju Reac-chat-bot-kita-a

Izvor: Oseberg, 2020

```
<div className={classes.chatbotContainer}>
  {showDiv ? (
    <Chatbot
      config={config}
      actionProvider={ActionProvider}
      messageParser={MessageParser}
    ></Chatbot>
  ) : null}
</div>
<button
  className={classes.chatbotButton}
  onClick={() => setShowDiv(!showDiv)}
>
  <RedditIcon />
</button>
```

Slika 27 Chatbot komponenta

Izvor: Oseberg, 2020

6.1.1. Config

Config datoteka primarno služi za konfiguriranje chatbota to jest služi za uređivanje i dodavanje različitih dijelova grafičkog sučelja. Uređivanje grafičkog sučelja se postiže promjenom svojstava u definiranim objektima koji su opisani u „react-chatbot-kit“ dokumentaciji. Objekti za uređivanje su (Oseberg, 2020):

- botName – pruža mogućnost promjene naziva chatbot-a
- initialMessages – polje objekata koji ispisuje početnu poruku chatbota, koristi se svojstvo „createChatbotMessage“ pomoću kojeg u navodnicima navodimo tekst za ispis
- state – definira objekt koji će biti uključen u chatbot state, koji se potom može koristiti u widgetima
- customeComponents – definira objekt prilagođenih komponenti koji će zamijeniti dioničke komponente chatbota
- customeStyles – definira objekt prilagođenih stilova, npr. promjena boje dugmeta
- widgets – definira niz widgeta koje želite generirati s porukom chatbota

```
const config = {
  botName: botName,
  initialMessages: [
    createChatBotMessage(
      ` Pozdrav ja sam ${botName}. Dobrodošli na stranicu novogradiške njuške`,
      {
        widget: "options",
      }
    ),
  ],

  customComponents: {
    header: () => (
      <div
        style={{
          backgroundColor: "#233953",
          padding: "15px",
          borderRadius: "5px 5px 0px 0px",
          color: "white",
        }}
      >
        Razgovor sa Chatbot-om
      </div>
    ),
    botAvatar: (props) => <BotAvatar {...props} />,
  },

  customStyles: {
    botMessageBox: {
      backgroundColor: "#233953",
    },

    chatButton: {
      backgroundColor: "#233953",
    },

    innerContainer: {
      width: "500px",
    },
  },

  widgets: [
    {
      widgetName: "options",
      widgetFunc: (props) => <Options {...props} />,
      mapStateToProps: ["gist"],
    },
  ],
};
```

Slika 28 Datoteka config.js

Izvor: Oseberg, 2020

Iz slike Slika 28. može se vidjeti sva upotrijebljena svojstva, npr. „header“ nam omogućuje promjenu zadanog zaglavlja, te uz dodavanje odgovarajućih stilova omogućena nam je kompletna promjena izgleda zaglavlja. Pomoću „botAvatar“ možemo promijeniti izgled zadanog chatbot avatara, itd. U „customeStyles“ su promijenjene boje korisnički i chatbotov oblačić i izmijenjana je širina chatbot kontejnera.

6.1.2. Message Parser

Message Parser je komponenta koja unutar chatbota prima upisane riječi od strane korisnika, to jest prima korisnikove poruke te ovisno o tome što pojedina poruka sadržava, donosi se odluka koji će odgovor biti poslan nazad korisniku (Oseberg, 2020).

```
class MessageParser {
  constructor(actionProvider) {
    this.actionProvider = actionProvider;
  }

  parse(message) {
    const lowercase = message.toLowerCase();

    if (lowercase.includes("pozdrav")) {
      this.actionProvider.pozdrav();
    } else if (lowercase.includes("doviđenja")) {
      this.actionProvider.dovidjenja();
    } else if (lowercase.includes("o udruzi") || lowercase.includes("udruga")) {
      this.actionProvider.onama();
    } else if (
      lowercase.includes("pomoc") ||
      lowercase.includes("pomoć") ||
      lowercase.includes("pomoć udruzi")
    ) {
      this.actionProvider.pomoc();
    } else if (
      lowercase.includes("gdje se nalazite") ||
      lowercase.includes("mjesto")
    ) {
      this.actionProvider.mjesto();
    } else if (
      lowercase.includes("udomljavanje") ||
      lowercase.includes("udomljavanje životinja") ||
      lowercase.includes("udomljavanje zivotinja")
    ) {
      this.actionProvider.udomljavanje();
    } else {
      this.actionProvider.neispravno();
    }
  }
}

export default MessageParser;
```

Slika 29 Datoteka MessageParser.js

Izvor: Oseberg, 2020

Slika 29. prikazuje komponentu MessageParser. Konstanta imena „lowecase“, u sebi sadrži korisnikovu poruku (message) i funkciju (toLowerCase()) koja pretvara sav tekst u mala slova, što znači ako korisnik upiše tekst sa velikim slovima, on će biti pretvoren nazad u mala. Funkcija služi kako bi se olakšala komunikaciju između MessageParser-a i ActionProvider-a, jer program gleda riječi „pozdrav“ i „Pozdrav“ kao dvije različite riječi. Zatim prolazi kroz ugniježđenu selekciju gdje se provjerava da li

konstanta sadrži određeni string, ako sadržava i ako ne sadržava proces se dalje odvija do ActionProvider komponente. Daljnji postupak je naveden u sljedećem poglavlju.

6.1.3. ActionProvider

U ActionProvider-u definiramo odgovore, te uz pomoć MessageParser-a odlučujemo koji ćemo odgovor aktivirati. Omogućuje fleksibilnost u donošenju odluka, to jest programer može sam donijeti odluku na koji će način Chatbot odgovoriti, također može odrediti stupanj kompleksnosti u terminima unutar MessageParser -a (Oseberg, 2020).

```
} else if (
  lowercase.includes("gdje se nalazite") ||
  lowercase.includes("mjesto")
) {
  this.actionProvider.mjesto();
```

Slika 30 else if uvjet

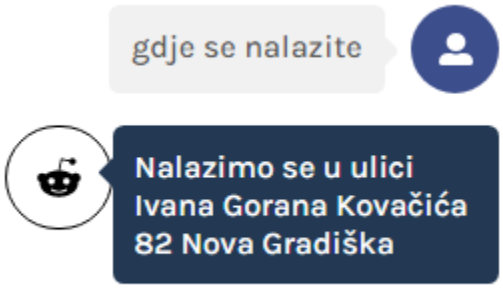
Izvor: Oseberg, 2020

Primjer, kada korisnik unese jednu od ključnih riječi definiranih na Slici 30. „gdje se nalazite“ ili „mjesto“) iz ActionProvider-a se ispisuje funkcija imena „mjesto“ Chatbot poruka Slika 31. sa tekstom koju programer svojevoljno napiše i poruka se dodaje na state, i ispisuje se na korisničkom sučelju Chatbot-a. Cijeli postupak u konačnici izgleda kao na Slici 32.

```
mjesto = () => {
  const message = this.createChatBotMessage(
    "Nalazimo se u ulici Ivana Gorana Kovačića 82 Nova Gradiška"
  );
  this.addMessageToState(message);
};
```

Slika 31 Funkcija mjesto

Izvor: Oseberg, 2020

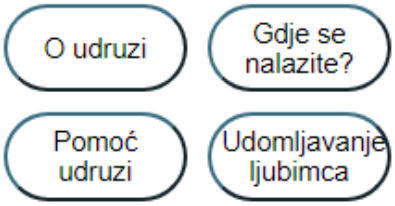


Slika 32 Chat-bot odgovor

Izvor: Oseberg, 2020

6.1.4. Widgets

React-chatbot-kit pruža mogućnost dodavanja prilagođenih značajki, to jest omogućuje programeru da svojevoljno doda željeno svojstvo u chat-bot. Slika 33. prikazuje „Options widget“ koji se sastoji od četiri dugmeta, u svakom dugmetu navedeno je jedno pitanje, korisnik klikom na željeno pitanje zauzvrat dobiva odgovor. Navedena pitanja su četiri najčešća upita za chatbot, te se ovako korisniku omogućuje brži pristup informaciji, bez unošenja ključnih riječi (Oseberg, 2020).



Slika 33 Options widget

Izvor: Oseberg, 2020

7. Zaključak

U svrhu završnog rada izrađena je internetska stranica i chat-bot, stranica je napravljena pomoću MERN stack-a, glavne biblioteke i servisi koje čine MERN stack su: program za bazu podataka MongoDB, okvir Express.js, JavaScript biblioteka React i platforma Node.js. Razvoj web aplikacije uz pomoću MERN razvojnog okvira pruža odličnu izradu fleksibilnih i skalabilnih web aplikacija. Ako je programer uredno napisao kod koji omogućuje višestruku uporabu samim tim omogućava izrada ponovno iskoristivih komponentni, olakšava se njihovo ponovno korištenje i implementacija te se također olakšava pronalazak grešaka.

Za izradu chat-bota, korišten je komplet imena React-chatbot-kit autora Fredrika Oseberga. Glavna zadaća internetske stranice i chatbot-a je omogućiti jednostavniju i bržu komunikaciju između dvije strane korisnika za udrugu koja promiče prava životinja osnovana 2019. godine imena „Novogradiške Njuške“. Temeljni dio aplikacije su objave, pomoću kojih korisnik opisuje kućnog ljubica i označuje ga kao spremnog za udomljavanje. Također objave služe kao kolekcije kućnih ljubimaca za osobe koje su u potrazi za novim četveronošcima. Izrađena web stranica prikazuje kakvu ulogu ima tehnologija u današnje doba to jest kako može pozitivno pridonijeti svakoj osobi koja je zna na ispravan način iskoristiti.

Literatura

[1] JavaScript, MDN Web Docs, dostupno na https://developer.mozilla.org/en-US/docs/Web/JavaScript [pristupljeno 4.7.2021.]

[2] React Dokumentacija, Introducing JSX, dostupno na: https://reactjs.org/docs/introducing-jsx.html, [pristupljeno 4.7.2021.]

[3] MongoDB dokumentacija, https://www.mongodb.com/docs/ [pristupljeno 8.7.2021.]

[4] Express.js dokumentacija, dostupno na https://expressjs.com/en/5x/api.html [pristupljeno 8.7.2021.]

[5] Microsoft, Visual studio Code, dostupno na https://code.visualstudio.com/docs [pristupljeno 5.7.2021.]

[6] Node.js dokumentacija, dostupno na https://nodejs.org/en/docs/ [pristupljeno 4.7.2021.]

[7] Material UI, MUI, dostupno na https://mui.com/ [pristupljeno 4.9.2021.]

[8] Reac Redux dokumentacija, dostupno na https://react-redux.js.org/introduction/getting-started [pristupljeno 7.7.2021.]

[9] React Router dokumentacija, dostupno na https://reactrouter.com/docs/en/v6 [pristupljeno 14.7.2021.]

[10] dotenv, dostupno na https://www.npmjs.com/package/dotenv [pristupljeno 4.9.2021.]

[11] CORS, MDN Web Docs, dostupno na https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS [pristupljeno 4.7.2021.]

[12] Postman, dostupno na https://www.postman.com/ [pristupljeno 4.9.2021.]

[13] Facebook, META, dostupno na https://www.facebook.com/ [pristupljeno 4.9.2021.]

[14] The MERN stack: A complete tutorial, Nur Islam ,dostupno na https://blog.logrocket.com/mern-stack-tutorial/ [pristupljeno 4.9.2021.]

[15] Redux for Beginners, DevEd (Youtube), dostupno na https://www.youtube.com/watch?v=CVpUuw9XSjY [pristupljeno 4.9.2021.]

[16] Full React Tutorial, The Net Ninja (Youtube), dostupno na https://www.youtube.com/watch?v=j942wKiXFu8&list=PL4cUxeGkcC9gZD-Tvwfod2gaISzfRiP9d [pristupljeno 4.9.2021.]

[17] Full Stack MERN Project ,Adrian Hajdin, JavaScript Mastery (Youtube), dostupno na https://www.youtube.com/watch?v=ngc9gnGgUdA https://www.youtube.com/watch?v=aibtHnbeuio&ab_channel=JavaScriptMastery [pristupljeno 5.9.2021.]

[18] React-chatbot-kit, Frederick Oseberg Coding Lessons (Youtube), dostupno na https://www.youtube.com/watch?v=AeojRYwfAMo&list=PL_kr51suci7UQAxHOF2GitkM5WrOBPcpf [pristupljeno 5.9.2021.]

[19] MongoDB in 30 minutes, Traversy Media (Youtube), dostupno na https://www.youtube.com/watch?v=pWbMrx5rVBE&t=12s [pristupljeno 5.9.2021.]

[20] Reddit, Advance Publications, dostupno na https://www.reddit.com/ [pristupljeno 5.11.2021.]

[21] Tomšić Damjan, Što je Internet, dostupno na http://www.oblakznanja.com/2011/07/sto-je-internet/ [pristupljeno 4.8.2021]

[22] TehnoKlik, Šta je chatbot i kako ga se koristi?, dostupno na https://tehnoklik.ba/sta-je-chatbot-i-kako-ga-koristiti/ [pristupljeno 4.8.2021]

[23] Sakina Vohra, Everything you need to know About Chatbots: A detailed history, dostupno na https://chatbotslife.com/everything-you-need-to-know-about-chatbots-a-detailed-history-837ce9db5aaf [pristupljeno 6.8.2021]

[24] Physics World, The Turing Test 2.0, dostupno na https://physicsworld.com/a/the-turing-test-2-0/ [pristupljeno 6.8.2021]

[25] Nitin Pandit, What And Why React.js, dosutpno na https://www.c-sharpcorner.com/article/what-and-why-reactjs/ [pristupljeno 4.7.2021]

[26] Bridget Botelho, MongoDB, dostupno na https://www.techtarget.com/searchdatamanagement/definition/MongoDB [pristupljeno 4.7.2021]

[27] Taha Sufiyan, What is Node.js: A Comprehensive Guide, dostupno na https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs [pristupljeno 4.7.2021]

[28] Kadeisha Kean, What is Express.js and Why Should You Use It?, dostupno na https://www.makeuseof.com/what-is-express/ [pristupljeno 20.12.2021]

[29] K. Saravanan, System Development Methodologies dostupno na https://www.ijsr.in/upload/105047310905.pdf [pristupljeno 21.12.2021]

[30] Infolio, 7 Steps of Web Desing Development Process, dostupno na https://www.infolio.co/use-cases/design-team-management/7-steps-of-web-design-development-process-template [pristupljeno 20.12.2021]

[31] Example of Eliza, Siddhartha, dostupno na https://www.researchgate.net/figure/Example-of-ELIZA-ELIZA-a-chatbot-was-designed-by-Joseph-Weizenbaum-to-imitate-a_fig1_348306833 [pristupljeno 20.12.2021]

[32] Alan Turing, Vishnu https://medium.com/lessons-from-history/alan-turing-a-prodigy-whose-life-was-curtailed-for-being-homosexual-5bd5e686c1c0 [pristupljeno 20.12.2021]

[33] FPZG repozitorij https://repozitorij.fpzg.unizg.hr/ [pristupljeno 25.12.2021]

[34] Aspekt.co, Što je CMS sustav? dostupno na https://aspekt.co/blog/sto-je-cms-sustav [pristupljeno 25.12.2021]

[35] Aspekt.co, Što je SEO optimizacija Web stranice za tražilice? dostupno na https://aspekt.co/blog/sto-je-seo-optimizacija-web-stranice-za-trazilice [pristupljeno 25.12.2021]

[36] Nacionalni repozitorij završnih i diplomskih radova ZIR dostupno na https://zir.nsk.hr/ [pristupljeno 25.12.2021]

[37] Razvoj Wev aplikacije u React I Node.js okruženjima, Karlo Tvrdinić, dostupno na file:///C:/Users/korisnik/Downloads/diplomski-rad_tvrdinic.pdf [pristupljeno 25.12.2021]

[38] Sveučilištu u Beogradu, Fakultetu organizacijskih znanosti, dostupno na http://www.bg.ac.rs/en/members/faculties/FOS.php [pristupljeno 27.12.2021]

[39] Jadranka grupa, dostupno na https://www.jadranka.hr/ [pristupljeno 25.12.2021]

[40] Vesti.rs, dostupno na https://www.vesti.rs/ [pristupljeno 25.12.2021]

[41] Encyclopedia of Social Theory, Ritzer, dostupno na http://philosociology.com/UPLOADS/_PHILOSOCIOLOGY.ir_Encyclopedia%20of%20Social%20Theory.%20Vols.%201%20v%202_COMPLETE_Ritzer%20G.(ed)_2005_1030%20pgs.pdf [pristupljeno 23.12.2021]

[42] Univerza v Ljubljani Fakultet za računalstvo in informatiko, dostupno na https://fri.uni-lj.si/sl [pristupljeno 25.12.2021]

[43] 50Languages, dostupno na https://www.50languages.com/ [pristupljeno 26.12.2021]

[44] sr.wikipedia.org, dostupno na https://sr.wikipedia.org/wiki/%D0%93%D0%BB%D0%B0%D0%B2%D0%BD%D0%B0_%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B0 [pristupljeno 23.12.2021]

Popis slika