Razvoj web aplikacije za iznajmljivanje dobara

Kuča, Alen

Master's thesis / Diplomski rad

2021

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Pula / Sveučilište Jurja Dobrile u Puli**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:137:356809

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-08-05**

Repository / Repozitorij:

Digital Repository Juraj Dobrila University of Pula

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

ALEN KUČA

RAZVOJ WEB APLIKACIJE ZA IZNAJMLJIVANJE DOBARA

Diplomski rad

Pula, rujan, 2021. godine

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

Alen Kuča

RAZVOJ WEB APLIKACIJE ZA IZNAJMLJIVANJE DOBARA

Diplomski rad

JMBAG: 0303054168, redoviti student
Studijski smjer: Informatika

Predmet: Napredni algoritmi i strukture podataka
Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija
Mentor: izv. prof. dr. sc. Tihomir Orehovački

Pula, rujan, 2021. godine

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani **Alen Kuča**, kandidat za magistra **informatike** ovime izjavljujem da je ovaj Diplomski 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 Diplomskog 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.

Student

U Puli, 07.09.2021. godine

IZJAVA O KORIŠTENJU AUTORSKOG DJELA

Ja, **Alen Kuča** dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom **razvoj web aplikacije za iznajmljivanje dobara** 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.

Student

Alen Kuča

U Puli, 07.09.2021. godine

RAZVOJ WEB APLIKACIJE ZA IZNAJMLJIVANJE DOBARA

Alen Kuča

Sažetak: Svrha ovog diplomskog rada bila je izraditi web aplikaciju za iznajmljivanje dobara. Aplikacija je podijeljena na klijentsku i poslužiteljsku stranu a napravljena je pomoću PERN tehnologija čija je uloga i svrha opisana kroz rad. Kroz dijagram obrasca uporabe vizualno su prikazane funkcionalnosti aplikacije. U nastavku su prikazani i objašnjeni pojedini dijelovi aplikacijskog sučelja, njihova uloga i način korištenja. Implementacija objašnjava tijek i način izrade aplikacije sa pojedinim tehnologijama i alatima. Prikazane su slike pojedinih blokova koda koji najbolje prikazuju funkcionalnost i rad aplikacije. Korisničke upute opisuju način korištenja aplikacije kroz sve moguće funkcionalnosti. Aplikacija je objavljena preko Heroku cloud hosting na https://rentmi.herokuapp.com besplatnoj domeni.

Ključne riječi: najam, iznajmljivanje, web aplikacija, JavaScript React.js, Node.js, Express.js, PostgreSQL

DEVELOPMENT OF A WEB APPLICATION FOR RENTING GOODS

Alen Kuča

Abstract: The purpose of this thesis was to create a web application for renting goods. The application is divided into client and server side and is made using PERN technologies whose role and purpose are described throughout the work. The application functionalities are visually displayed through the use case diagram. The individual parts of the application interface, their role and method of use are presented and explained below. Implementation explains the course and method of application development with individual technologies and tools. Images of individual blocks of code are shown, which best show the functionality and operations of the application. User instructions describe how to use the application through all possible functionalities. The application was published through Heroku cloud hosting on the https://rentmi.herokuapp.com free domain.

Keywords: rent, renting, web application, JavaScript React.js, Node.js, Express.js, PostgreSQL

Sadržaj

1. Uvod 1
   1.1. Cilj aplikacije 2
   1.2. Konkurencija 2
   1.3. Problem koji aplikacija rješava 2
2. SWOT analiza 4
3. Dijagram obrasca uporabe 5
4. Korištene tehnologije i alati 6
   4.1. React.js 6
   4.2. Node.js 6
   4.3. Express.js 7
   4.4. PostgreSQL 7
   4.5 pgAdmin 8
5. Razrada funkcionalnosti 9
   5.1. Stranica za izradu korisničkog računa 9
   5.2. Prijava 10
   5.3. Moje stvari 10
   5.4. Moje objave 12
   5.5. Pretraživanje 13
   5.6. Stranica proizvoda 14
   5.7. Postupak najma 15
   5.8. Šifre najma 17
   5.9. Unajmljeno 18
   5.10. Postupak vraćanja 19
   5.11. Iznajmljeno 20
   5.12. Stranica statistike 21
   5.13. Profil 21
   5.14. Poruke 22
   5.15. Obavijesti 23
6. Implementacija 25
   6.1. Klijentska strana 25
      6.1.1. Struktura direktorija 25
      6.1.2. Rute i navigacija 26
      6.1.3. Komponente 27

6.1.4. React stanje ... 28
6.1.5 Karta ... 30
6.1.6. Sass ... 31
6.2. Poslužiteljska strana ... 32
6.2.1 Express.js Rute ... 33
6.2.2. Baza podataka ... 33
6.2.3. Tablica korisnika ... 34
6.2.4. Cloudinary ... 34
6.2.5. Postman ... 35
6.2.6. Heroku i Github ... 36
7. Korisničke upute ... 37
7.1. Izrada računa i prijava ... 37
7.2. Dodavanje proizvoda ... 37
7.3. Iznajmljivanje proizvoda ... 37
7.4. Unajmljivanje proizvoda ... 38
7.5. Završetak najma i vraćanje proizvoda ... 38
7.6. Promjena adrese i profilne slike ... 38
7.7. Slanje i odgovaranje na poruke ... 39
7.8. Notifikacije ... 39
8. Zaključak ... 40
9. Literatura ... 42
Popis slika ... 43

1. Uvod

Web aplikacija je računalni program koji funkcionira prema arhitekturi klijent-poslužitelj (engl. Client-server) i koji koristi web preglednik za obavljanje određene funkcije. Dizajniranje i izrada web aplikacija vrši se na različite načine i dostupne su za korištenje od organizacija i pojedinca iz brojnih razloga.

Zadatak ovog diplomskog rada bio je razvoj web aplikacije koja omogućava iznajmljivanje raznih dobara. Aplikacija ima naziv „Rentmi“, objavljena je na heroku cloud hosting i izrađena je pomoću React.js, Node.js, Express.js i PostgreSQL tehnologija. Skup ovih tehnologija ima popularan naziv „PERN Stack“. Prednost ovog skupa tehnologija je korištenje SQL relacijske baze podataka koja je efikasna kod velikih i kompleksnih transakcija. Izgled komponenta aplikacije definiran je pomoću SASS pred procesora koji se prevodi u klasičan CSS prilikom izrade produkcijske verzije aplikacije.

Aplikacija pruža korisniku uvid u proizvode koji su trenutno dostupni za iznajmljivanje. Detaljnije pretraživanje moguće je pomoću filtera koji filtriraju objave prema županiji i vrsti proizvoda. Također, korisnik ima opciju dodavanja i objavljivanja svojih vlastitih proizvoda, te pregled u povijesne podatke o svojim najmovima. Unajmljivanje proizvoda vrši se u kratkim koracima gdje se odabire vrsta dostave i trajanje najma u danima. Početak najma započinje unosom jednokratne šifre od unajmitelja sa iznajmiteljeve strane. Najmovi u tijeku su dizajnirani na način da obavještavaju vlasnika i unajmitelja o trenutnom stanju najma. Moguće je poništiti najam od strane unajmitelja, te vratiti proizvod postupkom povratka prije isteka ugovorenog vremena. Završetak najma označuje vlasnik nakon primitka proizvoda. Za dogovaranje i rješavanje nesporazuma između korisnika, omogućena je razmjena poruka unutar aplikacije preko sustava poruka te koje je moguće brisati i označivati kao nepročitane. Naslovna traka prikazuje sve nedavne obavijesti kako bi korisnik bio informiran o svim bitnim događajima vezanim za njegove najmove i primljene poruke. Aplikacija je prilagođena za sve uređaje koji imaju pristup web pregledniku.

1.1. Cilj aplikacije

Cilj aplikacije je omogućit korisnicima iznajmljivanje i unajmljivanje raznih dobara, pružanje uvida o trenutnim i bivšim najmovima i mogućnost komunikacije unutar aplikacije sa ostalim korisnicima.

1.2. Konkurencija

Istraživanjem tržišta u Republici Hrvatskoj nije pronađena aktivna stranica koja omogućuje iznajmljivanje dobara između korisnika ili slično. Pronađene su samo Facebook grupe koje su većinom fokusirane na najam stanova. Istraživanjem vanjskog tržišta pronađene su aplikacije poput „RentItems“ koja nudi svoje usluge na cijelom svijetu. Stranica je neaktivna i nije pronađen niti jedan proizvod dostupan za iznajmljivanje. Zatim stranica „Ruckify“ koja se čini aktivna i uspješna ali uz proizvode korisnika, iznajmljuje proizvode iz svojih trgovina i također ima svoje zaposlene dostavljače. Najuspješnija aplikacije je „Rnters“ koja posluje prvenstveno u Portugalu ali isto iznajmljuje proizvode iz ostalih trgovina. U vrijeme pisanja ima preko pet tisuća objavljenih proizvoda.

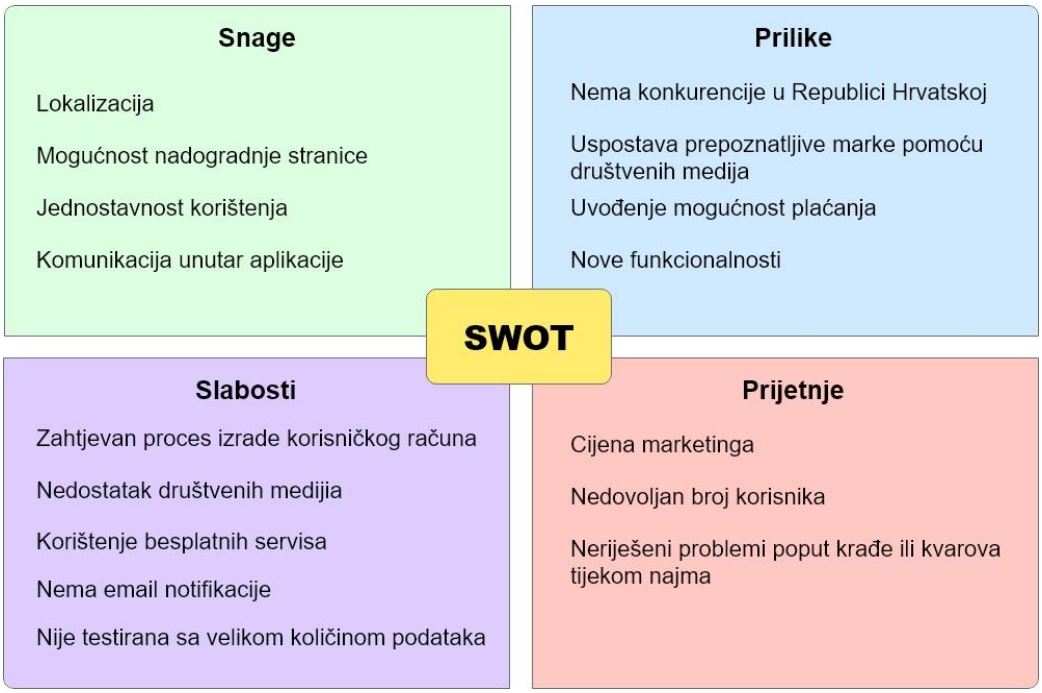
1.3. Problem koji aplikacija rješava

Problem je što za prosječnu osobu u Hrvatskoj ne postoji način laganog i brzog iznajmljivanja željenih dobara. Jedina slična opcija je kupiti proizvod te ga nakon korištenja, ako je moguće vratiti i tražiti povrat novaca. Kada bi osoba znala da postoji način brzog iznajmljivanja bilo to u blizini ili ostatku Hrvatske, možda bi odlučila iznajmiti proizvod na određeno vrijeme umjesto kupnje i s time uštedjeti novac. Naravno nakon najma, odluči li osoba kupiti takav proizvod, imat će više znanja i iskustva tijekom važne odluke koja je kupnja.

Savršen slučaj za iznajmljivanje dobara je kada osoba ima potrebu za nekim proizvodom ali samo na određeno vrijeme. To su slučajevi poput izleta, putovanja, okupljanja, istraživanje novih interesa ili općenita znatiželja o proizvodu.

2. SWOT analiza

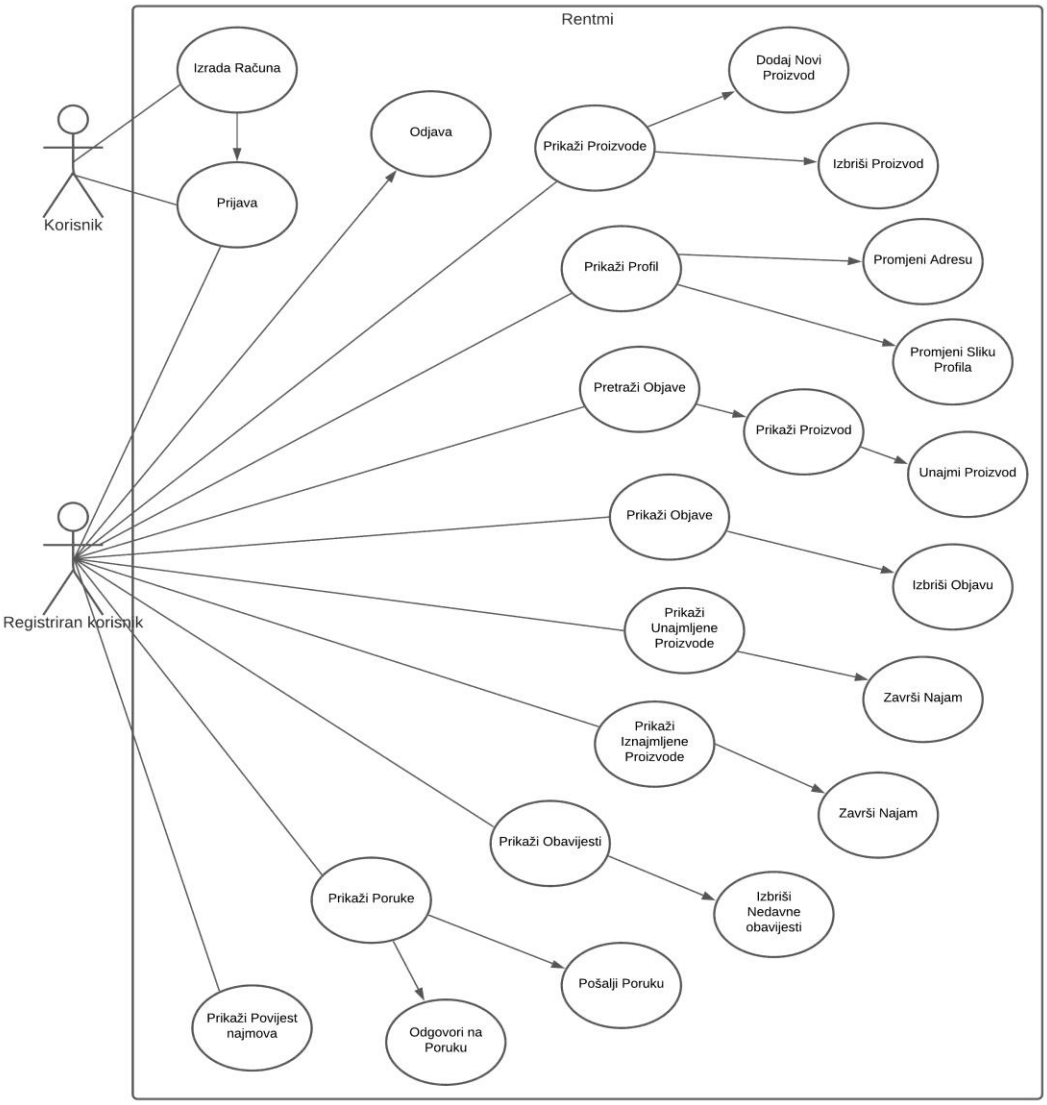
Iz Swot analize prikazane na slici 1, vidljive su prednosti i mane aplikacije. Glavna prednost aplikacije je to što je lokalizirana i namijenjena samo za korisnike Republike Hrvatske gdje nema sličnih aplikacija ili konkurencije. Glavni problemi kod aplikacije su korištenje besplatnih servisa kod izrade aplikacije, nedostatak email obavijesti i ne optimiziranost za veliku količinu podataka.



Slika 1. SWOT analiza

3. Dijagram obrasca uporabe

Dijagram obrasca uporabe (engl. Use case diagram) prikazuje ponašanje sustava u interakciji sa sudionicima sustava (akterima), dakle u slici 2, prikazana je samo funkcionalnost koju vide određeni korisnici te prikazuje kako se sustav može upotrebljavati. Unutar krugova definirane su funkcionalnosti aplikacije kojima samo korisnik povezan linijom ima pristup. Vidljivo je kako samo registrirani korisnik ima pristup glavnim funkcionalnostima aplikacije, dok korisnik bez korisničkog računa, pristupa samo funkcionalnostima za izradu korisničkog računa i prijave.



Slika 2. Dijagram obrasca uporabe

4. Korištene tehnologije i alati

4.1. React.js

React.js je JavaScript biblioteka otvorenog koda koja se koristi za izgradnju korisničkih sučelja, posebno za aplikacije na jednoj stranici [1]. Koristi se za upravljanje sloja za prikaz kod mobilnih i web aplikacija. React.js također omogućava izradu komponenata za višekratnu i ponovnu upotrebu.

React.js pruža priliku programerima za izradu velikih web aplikacija koje mogu mijenjat podatke bez ponovnog učitavanja stranice. Glavna svrha React-a je biti brz, skalabilan i jednostavan. Postoje mnogi dodatni paketi koji su namijenjeni specifično za React.js i omogućuju dodatne funkcionalnosti. Jedan od najkorištenijih je React-router koji služi za konfiguraciju navigacije unutar aplikacije. U React.js-u, umjesto uobičajenog JavaScript-a za izradu predloška, koristi se JSX. JSX je sintaksno proširenje JavaScripta i omogućuje izravno pisanje HTML koda u React.js-u. React.js verzija 16.8 uvodi React Hooks funkcionalnost koja omogućava korištenje stanja i ostalih React funkcionalnosti unutar funkcijskih komponenata. Za razliku od klasnih komponenata, funkcijske komponente korištene sa React Hooks su znatno više deklarativne i time prikladnije za React ekosustav [2].

4.2. Node.js

Node.js je okruženje koji izvršava JavaScript kod izvan preglednika i omogućuje programerima stvaranje web poslužitelja i mrežnih alata pomoću JavaScript-a i zbirke modula koji obrađuju različite osnove funkcije. Popularnost Node.js-a porasla je zajedno sa popularnošću modernih web aplikacija i izlaskom naprednijih web preglednika. Isto tako unapređenje jezika JavaScript, također je utjecalo na popularnost Node.js-a [3].

Najznačajnija prednost između Node.js-a i PHP-a je u tome što većina naredba u PHP-u budu izvršene tek nakon što prethodne naredbe budu završene, dok se Node.js naredbe izvršavaju istodobno ili paralelno i koristi povratne pozive za signalizaciju završetka ili greške. Ovaj način izvršavanja naredbi promijenilo je način izrade modernih web aplikacija.

4.3. Express.js

Express.js je okvir osmišljen za korištenje Node.js-a i specifično za izvođenje web poslužitelja. Znatno olakšava integraciju i pisanje kompleksnog Node.js koda. Express.js je dizajniran za izgradnju snažnih web aplikacija i API-ja. Poznat je po visokoj brzini i minimalističkoj strukturi. Iako je sam Express.js prilično minimalističan, programeri su stvorili kompatibilne pakete (engl. middleware) za rješavanje gotovo svih problema web razvoja. Postoje knjižnice za rad s kolačićima (engl. cookies), sesijama, prijavama korisnika, parametrima URL-a, POST podacima, sigurnosnim zaglavljima (engl. headers) i mnogim drugim [4].

4.4. PostgreSQL

PostgreSQL je sustav baze podataka otvorenog koda koji koristi i proširuje SQL jezik u kombinaciji s mnogim značajkama koje sigurno pohranjuju i skaliraju podatke. Podrijetlo PostgreSQL-a datira od 1986. godine u sklopu projekta POSTGRES na Kalifornijskom sveučilištu u Berkeley-u i ima više od 30 godina aktivnog razvoja [6]. PostgreSQL zaslužio je snažnu reputaciju svojom dokazanom arhitekturom, pouzdanošću, integritetom podataka, robusnim skupom značajki, proširivošću i predanošću zajednice otvorenog koda, koja stoji iza softvera kako bi dosljedno isporučivao učinkovita i inovativna rješenja.

4.5 pgAdmin

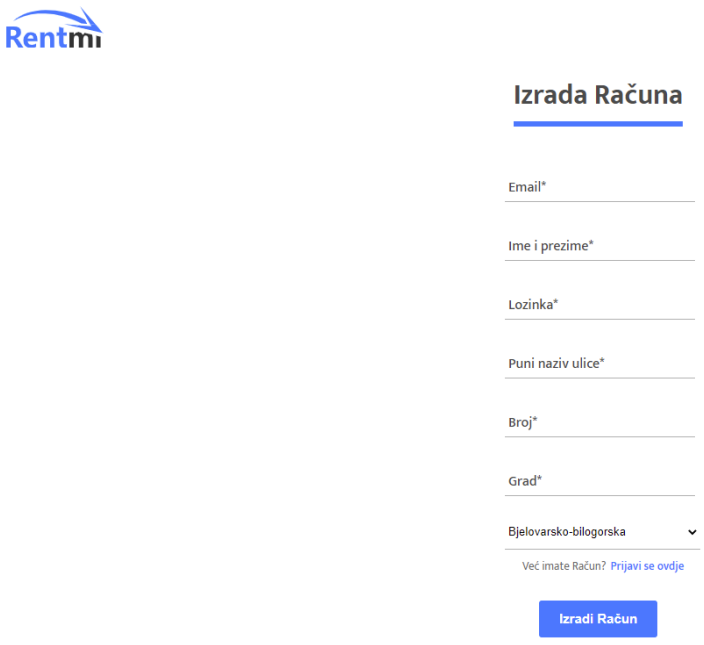
pgAdmin je najpopularniji, besplatan GUI (engl. graphical user interface) platforma otvorenog koda za PostgreSQL koji se može pokrenuti preko desktop-a ili web preglednika. Platforma služi za administraciju i upravljanje bazom podataka, te za olakšano korištenje ostalih funkcionalnosti PostgreSQL-a. Unutar pgAdmin dostupne su brojne opcije za upravljane bazom podataka. pgAdmin omogućuje spajanje na bazu podataka bilo ona lokalna ili na udaljenom poslužitelju, a potrebni podaci za upis su ime domaćina, port, korisničko ime i lozinka. Osim pgAdmin, postoje i ostali slični programi, a alternativa njima je korištenje naredbene linije sa psql komandama. Psql komade se najčešće koriste za pokretanje SQL upita ili za brzo pretraživanje baze [5].

5. Razrada funkcionalnosti

Ovo poglavlje opisuje razradu funkcionalnosti aplikacije prema korisniku aplikacije. Aplikacija ima jednu vrstu korisnika sa jednom vrstom korisničkog računa. Prikazano je sučelje i opisane su funkcionalnosti po pojedinim dijelovima aplikacije.

5.1. Stranica za izradu korisničkog računa

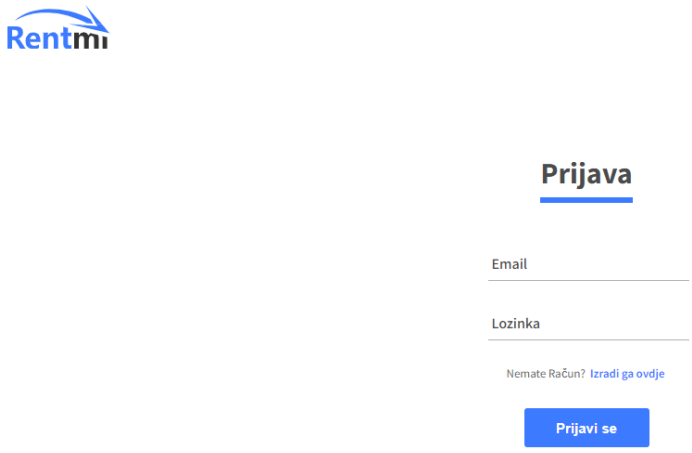
Na stranici za izradu korisničkog računa prikazana na slici 3, korisnik izrađuje svoj vlastiti korisnički račun. Traži se od korisnika da upiše svoje osobne podatke poput email, ime i prezime, adresu i županiju, te da odabere šifru. Klikom na *Izradi račun* i ako su svi podaci ispravno upisani, korisnika se automatski vodi na početnu stranicu sa nazivom „Moje stvari“



Slika 3. Stranica za izradu korisničkog računa

5.2. Prijava

Kako bi se korisnik prijavio u aplikaciju potrebno je koristit stranicu za prijavu vidljivu na slici 4. Potreban je samo email i šifra od već napravljenog korisničkog računa. Uspješnom prijavom, trenutni web preglednik postavlja jedinstven znak u lokalnu memoriju tražilice i kojeg koristi za automatsku prijavu prilikom sljedećeg korištenja aplikacije. Jednom prijavljeni korisnik ostaje prijavljen u aplikaciji deset dana ili sve dok ne klikne na gumb *odjavi se,* koji briše znak iz lokalne memorije. Klikom na *Prijava,* pojedinca se vodi na početnu stranicu koja je vidljiva na slici 6.

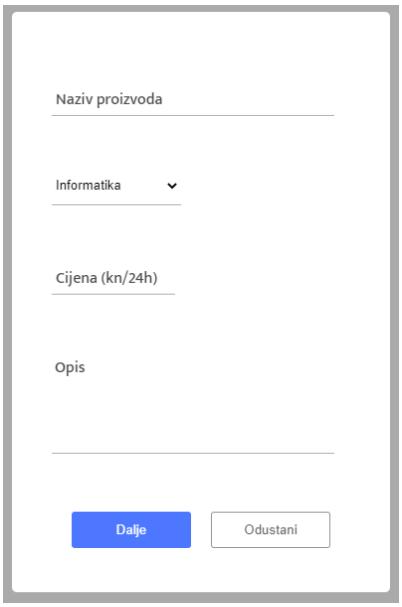


Slika 4. Stranica za prijavu

5.3. Moje stvari

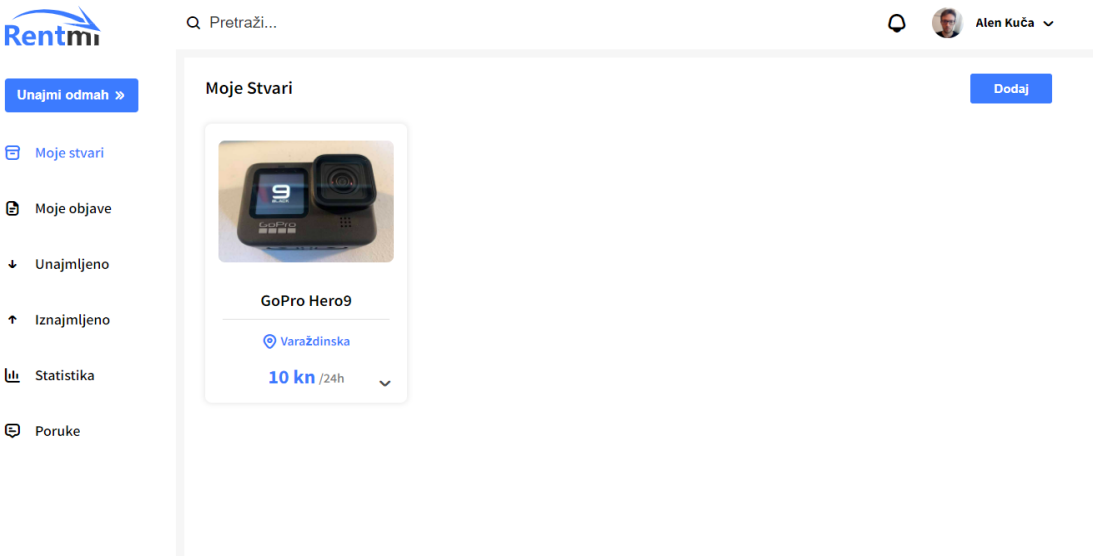
Stranica sa nazivom „Moje Stvari“ služi za prikaz proizvoda koje je korisnik izradio ali nisu trenutno objavljeni, ili iznajmljeni. Dakle proizvodi koji su trenutno neaktivni i time nevidljivi drugim korisnicima. Klikom na strijelcu na kartici proizvoda otvara se padajući izbornik sa dvoje opcije *Objavi* i *Ukloni.* Klikom na ukloni otvara se prozor sa upozorenjem, te se ponovnim klikom proizvod briše i nestaje iz aplikacije. Kako bi korisnik omogućio

iznajmljivanje nekog proizvoda potrebno je da ga prvo napravi te objavi. Objavljen proizvod znači da je vidljiv ostalim korisnicima i omogućen za unajmljivanje a čini se klikom na *Objavi* u padajućem izborniku proizvoda. Dodavanje novog proizvoda radi se na trenutnoj stranici pomoću opcije *Dodaj* koja otvara novi prozor i pokreće postupak dodavanja odnosno izrade novog proizvoda kao što je vidljivo na slici 5.



Slika 5. Izrada novog proizvoda

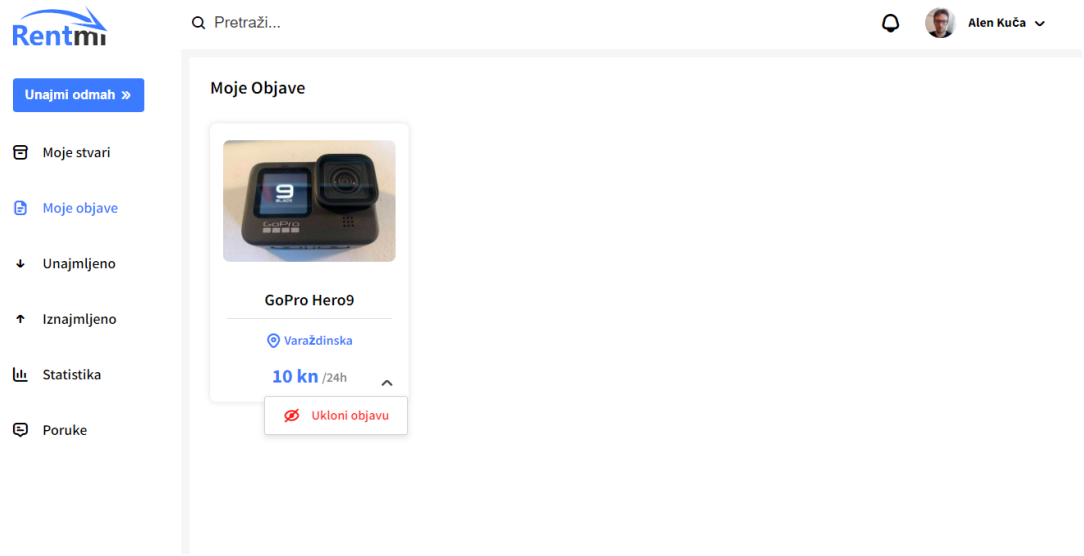
Potrebno je upisat ime, cijenu po danu, kratak opis i odabir kategorije proizvoda, te klikom na *Dalje,* otvara se sljedeći koraku u kojem korisnik odabire prikladnu sliku za proizvod sa svojeg uređaja. Ponovnim klikom na *Dodaj* proizvod se izrađuje i prikazuje na stranici „Moje Stvari“ prikazanom na slici 6. Svaki proizvod prikazan je karticom u kojoj je vidljiva slika proizvoda, cijena najma po danu i županija u kojoj se vlasnik nalazi.



Slika 6. Stranica „Moje Stvari“

5.4. Moje objave

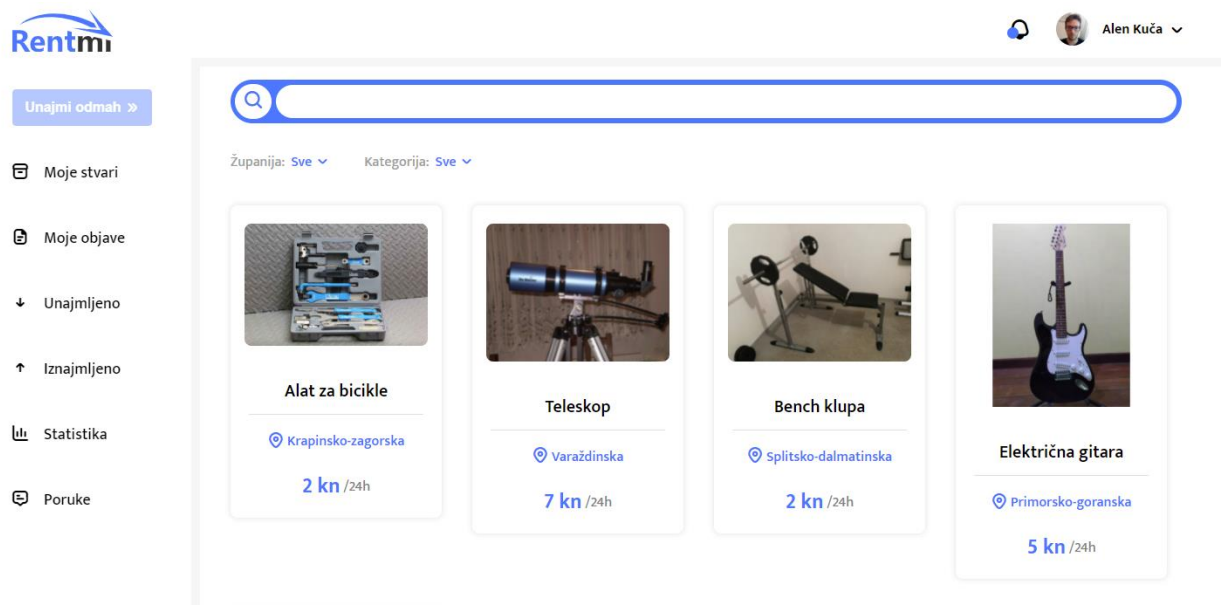
„Moje Objave“ stranica prikazana na slici 7, služi za prikaz svih korisničkih proizvoda koji su trenutno objavljeni odnosno vidljivi od ostalih korisnika i dostupni za iznajmljivanje. Jedina opcija na ovoj stranici je brisanje objave koja se čini na isti način kao i brisanje proizvoda, ali klikom na *Ukloni objavu*. Nakon brisanja, objava nestaje i proizvod se vraća na stranicu “Moje Stvari”. Objava također automatski nestaje kada proizvod bude unajmljen od drugog korisnika.



Slika 7. Stranica „Moje Objave“

5.5. Pretraživanje

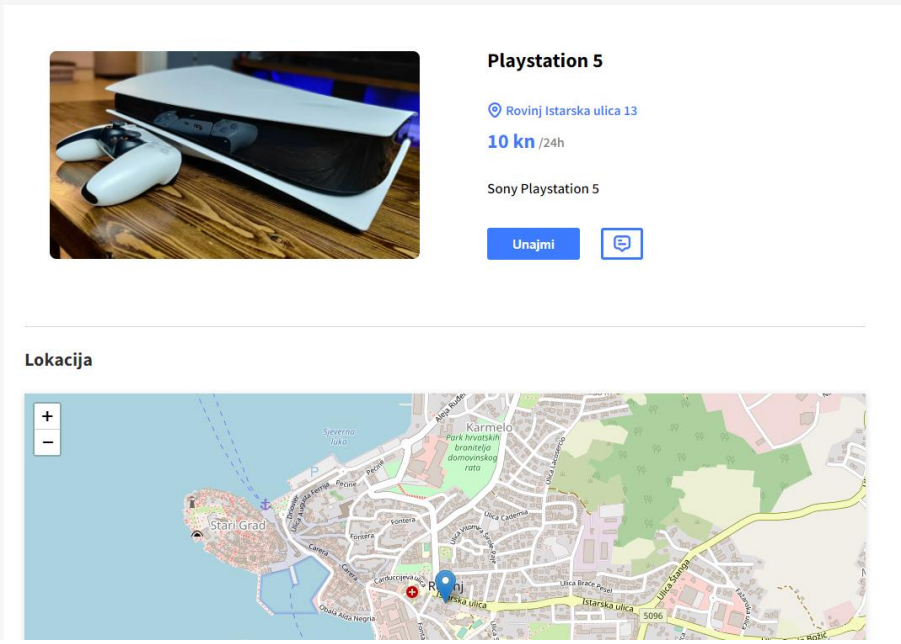
Preko stranice za pretraživanje, prikazane na slici 8, pojedinac pretražuje sve objave koje predstavljaju proizvode. Stranica se pristupa klikom na gumb *Unajmi odmah* ili upisom traženog pojma u traci za pretraživanje koja se nalazi na vrhu stranice. Dostupna su dva padajuća izbornika koji filtriraju objave po odabranoj županiji i kategoriji proizvoda. Upisom pojma u traku za pretraživanje, objave se dodatno filtriraju prema nazivu. Pošto svaka objava sadržava sliku i time znatno povećava količinu prenesenih podataka, uveden je sustav paginacije koji prikazuje samo deset objava po stranici. Navigacija kroz stranice vrši se klikom na broj stranice ili pomoću lijeve i desne strjelice. Klikom na sliku objave, otvara se stranica odabranog proizvoda koja je prikazana na slici 8.



Slika 8. Stranica za pretraživanje

5.6. Stranica proizvoda

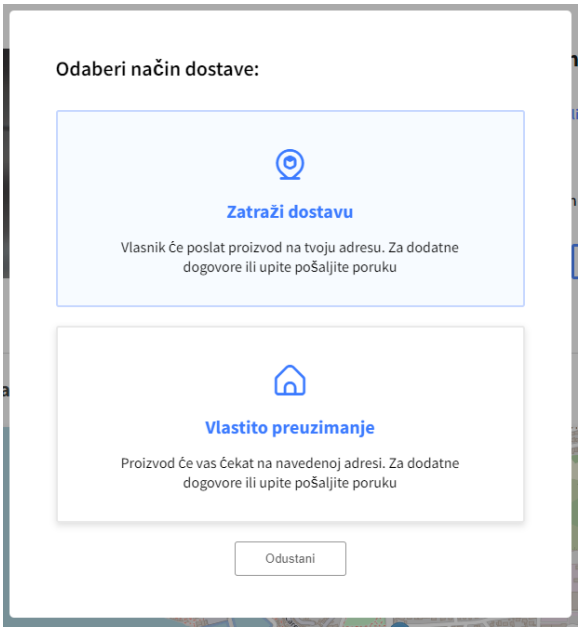
Stranica proizvoda pruža uvid u sve detalje proizvoda poput opisa, cijene, adrese i karte sa markerom na lokaciji vlasnika. Ispod karte se nalaze sve informacije o vlasniku zajedno sa profilnom slikom, a klikom na ime vlasnika, korisnika se vodi na profilnu stranicu vlasnika. Dostupne su i opcije za unajmljivanje i slanje poruka vlasniku proizvoda. U slučaju da vlasnik posjećuje stranicu svojeg proizvoda, prikazuje se opcija za brisanje proizvoda, a opcija za novu poruku nestaje. Stranica je prikazana na slici 9.



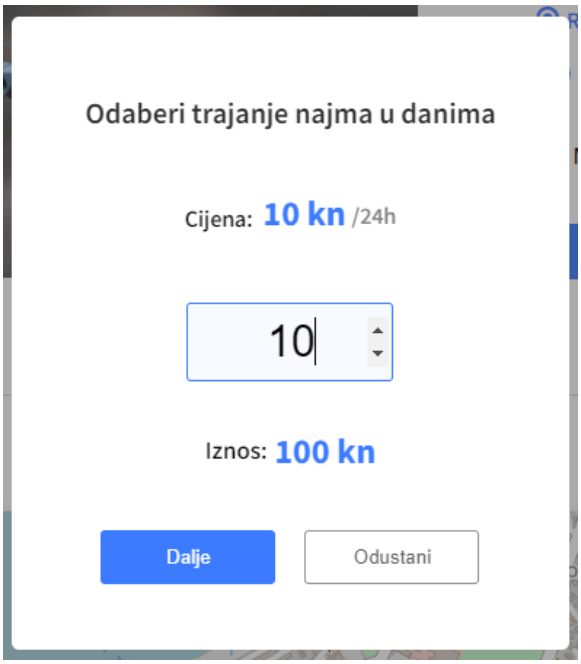
Slika 9 Stranica proizvoda

5.7. Postupak najma

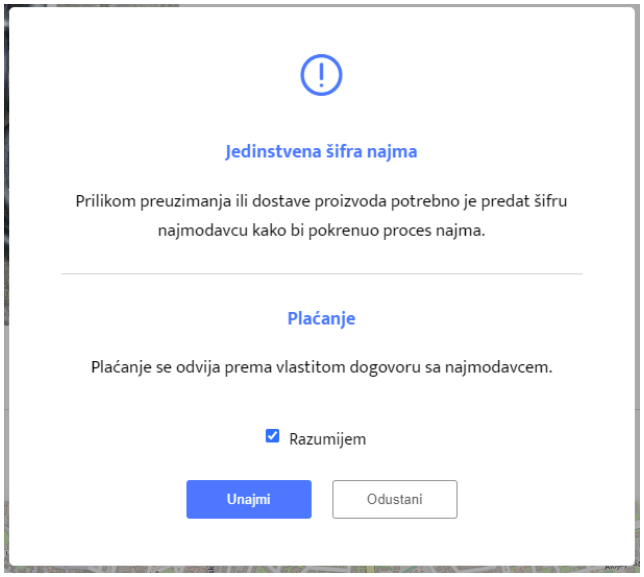
Klikom na *Unajmi* na stranici proizvoda, otvara se prozor od više koraka koji je prikazan na slikama 10,11 i 12. U prvom koraku korisnik odabire hoće li sam preuzet proizvod ili želi zatražit dostavu na svoju adresu. U sljedećem koraku odabire količinu dana najma gdje se cijena automatski izračuna i prikazuje zavisno o odabranoj količini dana. Zadnji korak prikazuje napomene u vezi šifre najma, koju je potrebno predati vlasniku proizvoda prilikom preuzimanja proizvoda i napomena u vezi plaćanja koja obavještava unajmitelja da se plaćanje odvija vlastitim dogovorom sa vlasnikom proizvoda. Ponovnim klikom na *Unajmi* korisnika se vodi na stranicu “Unajmljeno” gdje unajmljen proizvod postaje vidljiv, a vlasnika se putem obavijesti obavještava o učinjenom najmu i odabranoj vrsti dostave.



Slika 10 Način dostave



Slika 11. Trajanje najma



Slika 12. Napomene o najmu

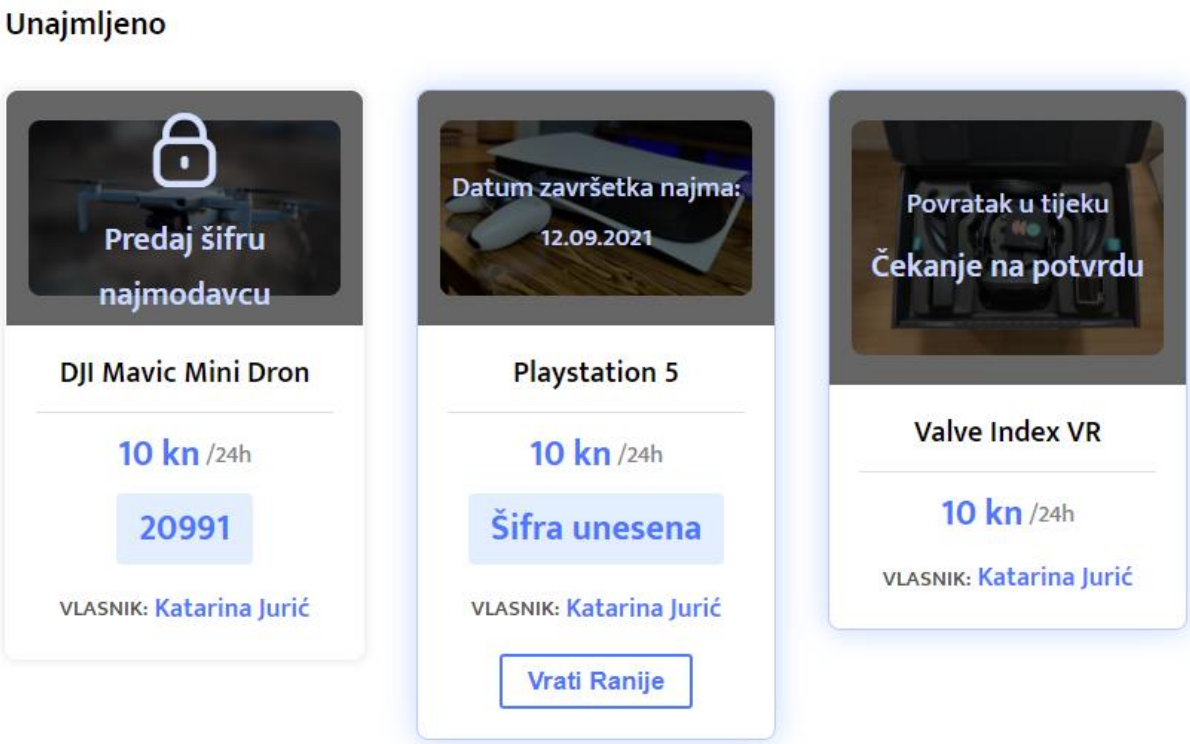
5.8. Šifre najma

Svaki unajmljen proizvod generira dvije peteroznamenkaste šifre. Prvu šifru dobiva unajmitelj prilikom unajmljivanja proizvoda i koju predaje vlasniku tijekom preuzimanja proizvoda ili u slučaju dostave, elektroničkim putem. Vlasnik unaša šifru sa svoje strane i započinje najam prema odabranom vremenu. Drugu šifru predaje vlasnik unajmitelju tijekom povratka proizvoda završenog ili ranije završenog najma. Unajmitelj unaša šifru sa svoje strane i time vlasniku omogućava opciju *Završi*, koja završava najam i vraća proizvod na stranicu "Moje Stvari". Prozor za unos šifre vidljiv je na slici 13.

Slika 13. Unos šifre najma

5.9. Unajmljeno

Stranica „Unajmljeno“ prikazuje sve proizvode koje korisnik trenutno unajmljuje. Prikazuju se sve potrebne informacije kako bi korisnik bio točno obavješten o stanju najma. Kada je najam u tijeku, odnosno još vrijeme nije isteklo, prikazan je datum završetka najma, a ako je vrijeme isteklo, prikazuje se tekst „Vrati proizvod“ sa opcijom *Vrati* koja pokreće postupak vraćanja proizvoda vlasniku. Proizvod ostaje na stranici „Unajmljeno“ sve dok unajmitelj ne označi kraj najma klikom na *Završi*. Primjer unajmljenih proizvoda prikazan je karticama na slici 14.



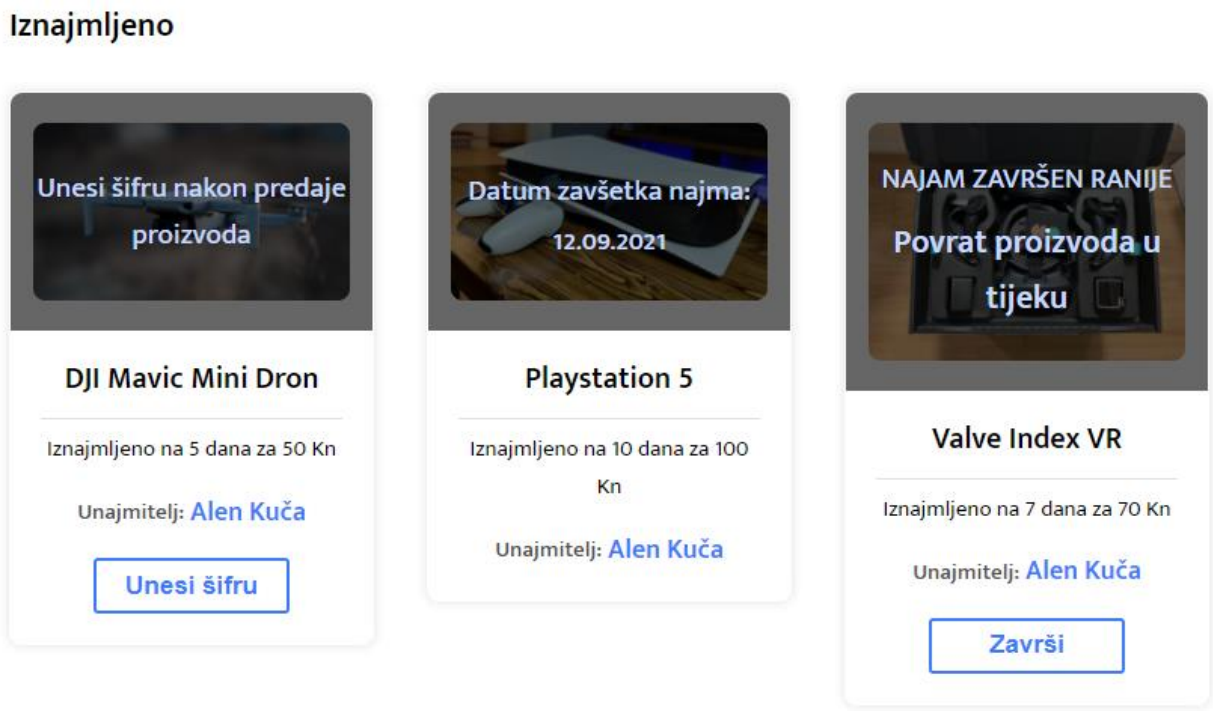
Slika 14. Unajmljeni proizvodi

5.10. Postupak vraćanja

Nakon završetka ili poništavanja najam, korisnik pokreće postupak vraćanja proizvoda pomoću klika na opciju *Vrati* ili *Vrati ranije*. U prvom koraku korisnik odabire na koji način vraća proizvod, a u drugom koraku se prikazuje karta sa informacijama o lokaciji vlasnika. Prilikom vraćanja proizvoda, potrebno je preuzeti šifru od vlasnika i unijeti u prozor za unos šifre i time potvrditi uspješno vraćanje proizvoda. Nakon završetka navedene radnje, vlasniku proizvoda se prikazuje opcija „Završi“ koja uspješno završava najam i vraća proizvod na stranicu „Moje Stvari“ i „Statistika“ pod kategoriju *iznajmljeno*, a na strani unajmitelja pod kategoriju *unajmljeno.*

5.11. Iznajmljeno

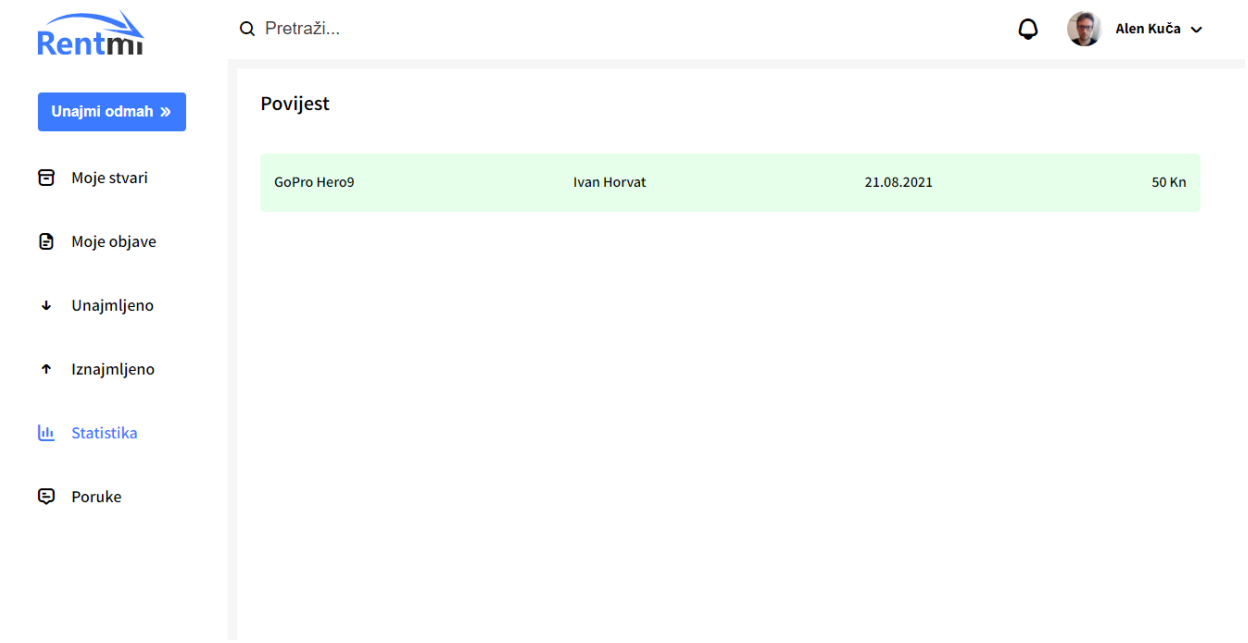
Stranica „Iznajmljeno“ služi za prikaz svih korisničkih proizvoda koji su trenutno iznajmljeni. Prikazane su informacije poput šifre koju je potrebno predat unajmitelju, stanje o unosu šifre i datum završetaka najma. Nakon završetka najma i uspješno vraćenog proizvoda, prikazuje se šifra koju je potrebno predati unajmitelju kako bi istu unio sa svoje strane i potvrdio vraćanje proizvoda. Nakon unosa šifre, prikazuje se opcija *Završi* koja završava najam i vraća proizvod na stranicu „Moje Stvari“ Primjer iznajmljenih proizvoda prikazan je slikom 15.



Slika 15. Stranica "Iznajmljeno"

5.12. Stranica statistike

Stranica sa nazivom “Statistika” služi za prikaz povijesnih podataka o iznajmljivanju proizvoda. Završeni najmovi prikazani su ovdje u kategoriji *iznajmljeno* i *unajmljeno*. Iznajmljeni proizvodi su označeni zelenom bojom, a unajmljeni sa plavom. Navedeni su detalji poput cijene najma, vlasnika proizvoda, i datuma. Pošto ove podatke nije moguće brisati, uvedena je opcija paginacije koja maksimalno prikazuje deset stavki po stranici. Stranica statistike prikazana je na slici 16.



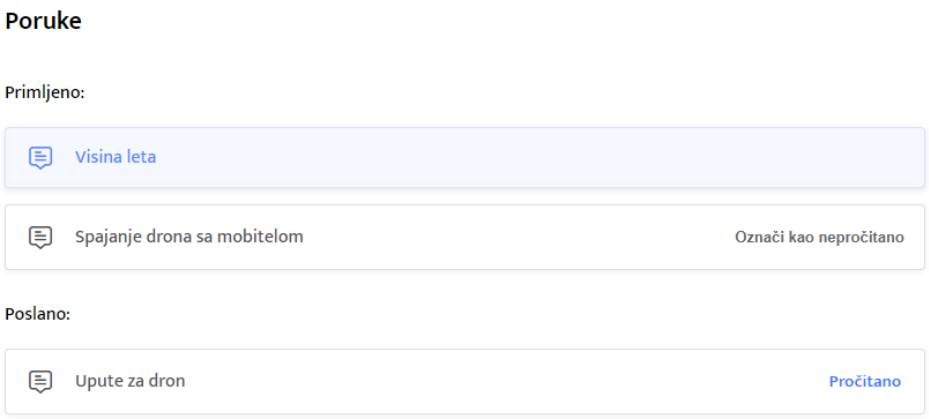
Slika 16. Stranica statistike

5.13. Profil

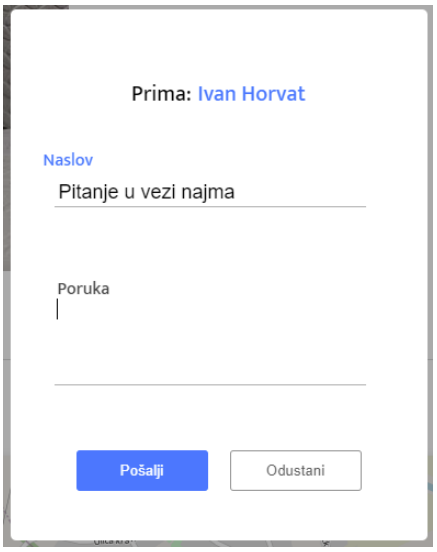
Profil stranica služi za prikaz osobnih informacija o korisniku. Prikazana je i karta sa markerom na adresi korisnika. Vidljiva je opcija za slanje poruka i ako je trenutni posjetitelj profila također i vlasnik profila prikazuje se opcija *Uredi* preko koje korisnik ima mogućnost promjene vlastite adrese. Klikom na profilnu sliku, otvara se opcija za izbor nove slike sa korisnikovog uređaja.

5.14. Poruke

Uvedena je mogućnost slanja poruka između korisnika kako bi korisnici direktno mogli komunicirat u vezi iznajmljivanja proizvoda. Nakon primitka poruke, poruka se pojavljuje na stranici “Poruke”, gdje primatelj ima mogućnost odgovaranja pošiljatelju sa opcijom *Odgovori*. Razlikuju se pročitane i nepročitane primljene poruke gdje nepročitane imaju posebnu oznaku sve dok primatelj poruke ne otvori navedenu. Pročitana poruka prikazuje opciju *Označi kao nepročitano* koja mijenja stanje u nepročitano. Poslane poruke prikazuju stanje „Pročitano“ ili „Nepročitano“. Stranica sa porukama je prikazan na slici 17, a izrada nove poruke na slici 18.



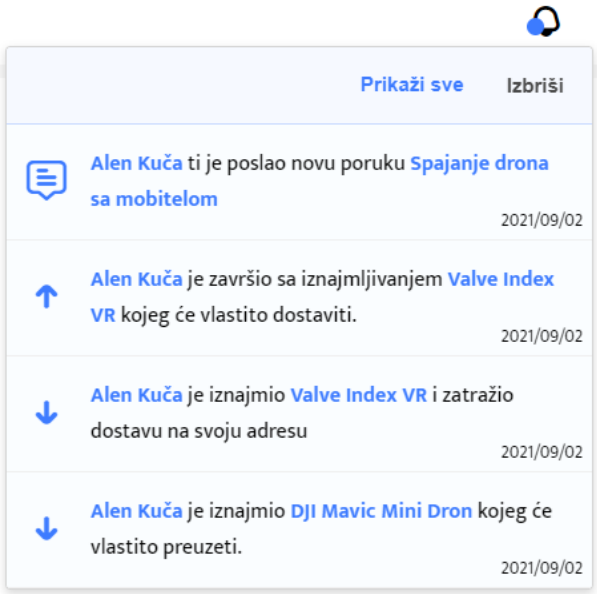
Slika 17. Stranica sa porukama



Slika 18. Nova poruka

5.15. Obavijesti

Obavijesti služe kako bi korisnik bio obaviješten o važnim događajima u vezi najma ili primljenih poruka. Nedavne obavijesti se prikazuju u padajućem prozoru na naslovnoj traci gdje još postoje dvije opcije, *Izbriši* koja briše nedavne obavijesti i opcija *Prikaži sve* koja vodi korisnika na stranicu sa svim obavijestima. Korisnik dobiva obavijest kada njegov proizvod bude unajmljen, primitak nove poruke ili završetak najma. U slučaju novog ili završenog najma, ispisuje se vrsta dostave kako bi korisnik znao na koji način predati ili preuzeti proizvod. Obavijesti su interaktivne, te klikom na plavi tekst aplikacija otvara određenu stranicu. Izgled prozora za obavijesti sa primjerima prikazan je na slici 19.



Slika 19. Obavijesti

6. Implementacija

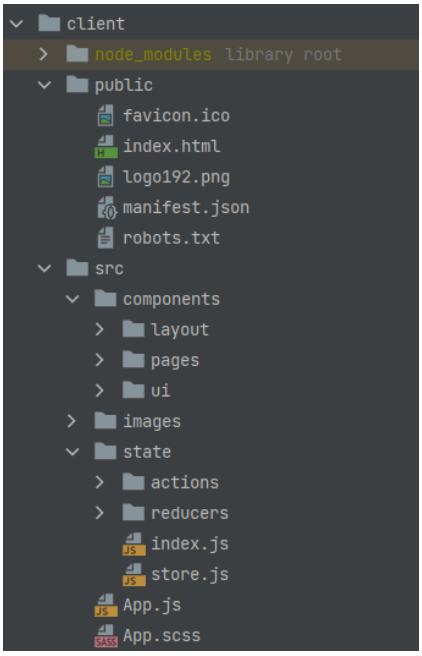
U ovom djelu je opisana implementacija aplikacije i način korištenja navedenih tehnologija tijekom izrade aplikacije. Implementacija je podijeljena na dva dijela, klijentska strana (engl. Front end) i poslužiteljska strana (engl. Back end). Klijentska strana je prednji dio aplikacije odnosno prezentacijski sloj, a poslužiteljska strana je sloj za pristup podacima. Za izradu web aplikacije postoje mnoga rješenja i skup alata, a za ovaj rad korištene su PERN Stack tehnologije.

6.1. Klijentska strana

Klijentska strana je izrađena pomoću *create-react-app* predloška koji služi za brzo kreiranje svih potrebnih postavka i modernih alata u vezi React.js. Prilikom navedene radnje postoji i opcija za odabir predložaka koji dolazi sa dodatnim unaprijed postavljenim postavkama. Pokretanjem komande „npm start“, pokreće se lokalni poslužitelj sa adresom „http://localhost:3000“ na kojem se nalazi nova React.js aplikacija i sa kojim je moguće komunicirati. Kreirana aplikacija dolazi sa opcijom testiranja koja koristi Jest kao pokretač testova. Komanda „npm run build“ kreira statične datoteke i produkcijsku verziju aplikacije koja je kasnije korištena za objavljivanje aplikacije.

6.1.1. Struktura direktorija

Sva logika vezana za klijentski dio aplikacije nalazi se u direktoriju “client”. Zatim unutar nje se nalazi direktorij “public” koji služi za spremanje statičnih podataka i direktorij „src“ unutar kojeg se nalazi direktorij „components“, koji sadrži sve komponente podijeljene na direktorije “pages”, “ui” i “layout”. Zatim direktorij „Images“ koji sadrži sve statične slike i direktorij „state“ unutar kojeg se nalazi sva logika vezana za globalno stanje aplikacije i Redux, a podijeljena je u dva direktorija „actions“ i „reducers“. Navedeno je vidljivo na slici 20.



Slika 20. Struktura direktorija poslužiteljske strane

6.1.2. Rute i navigacija

Svaka ruta ima definiranu komponentu koju prikazuje ako su zadovoljeni uvjeti. Rute koje se nalaze pod odredištem „/auth*“*, su rute preko kojih se korisnik autenticira. Za autentikaciju se koristi JWT znak (engl. token), koji se dobiva sa GET fetch zahtjevom i koji je detaljnije objašnjen u kasnijem dijelu. Nakon točne autentikacije, funkcija „chekcIfAuthenitcated()“ postavlja znak u lokalnu memoriju tražilice. Rute pod odredištem „/dashboard“ su zaštićene i dostupne samo ako je pravedan znak postavljen u memoriji tražilice. Ako znak nije postavljen, korisnika se automatski preusmjerava na stranicu za prijavu. Za navigaciju i preusmjeravanje kroz aplikaciju koriste se komponente iz React-router zbirke. Konfiguracija ruta nalazi se u datoteci „App.js“ koja je prikazana na slici 21.

```
return (
    <div>
        {!loading ?
        <div className="App">
            <Route path="/auth/register"
                render={props => <RegisterPage {...props} setAuth={setAuth} isAuthenticated={isAuthenticated} />}
            />
            <Route path="/auth/login"
                render={props => <LoginPage {...props} setAuth={setAuth} isAuthenticated={isAuthenticated} />}
            />
            <PrivateRoute isAuthenticated={isAuthenticated} path="/">
                <DashboardContent setAuth={setAuth} >
                    <Switch>
                        <Route exact path="/dashboard/stvari" component={ItemsPage}/>
                        <Route path="/dashboard/trazi" component={SearchPage}/>
                        <Route path="/dashboard/stvar/:item_id" component={ItemPage}/>
                        <Route path="/dashboard/unajmljeno" component={RentedPage}/>
                        <Route path="/dashboard/iznajmljeno" component={RentedOutPage}/>
                        <Route path="/dashboard/profil/:user_id" component={ProfilePage}/>
                        <Route path="/dashboard/poruke" component={MessagesPage}/>
                        <Route path="/dashboard/poruka/:message_id" component={MessagePage}/>
                        <Route path="/dashboard/objave" component={PostsPage}/>
                        <Route path="/dashboard/statistika" component={StatisticsPage}/>
                        <Route path="/dashboard/obavijesti" component={NotificationsPage}/>
                    </Switch>
                </DashboardContent>
            </PrivateRoute>
        </div>
            :
            <Loader/>
        }
    </div>
);
```

Slika 21. Rute

6.1.3. Komponente

Aplikacija se sastoji od raznih komponenata koje su strukturirane na način kako bi bile ponovno upotrebljive. Na slici 22 je prikazan primjer jednostavne ponovno upotrebljive funkcijske komponente sa nazivom „pageTitle“.

```
import React from 'react';

const PageTitle = props => {
    return (
        <div className="title-container">
            <h1>{props.title}</h1>
            {props.renderButton &&
            <button onClick={props.buttonAction}>{props.buttonText}</button>
            }
        </div>
    )
}
export default PageTitle;
```

Slika 22. Komponenta „pageTitle“

Prilikom pozivanja komponente potrebno je pridružiti unaprijed definirana svojstva kako bi dobili željeni prikaz. U ovom slučaju to su svojstva „renderButton“, „buttonText“, „buttonAction“ i „title“.

Komponente koje se nalaze u direktoriju pages, su komponente koje su povezane sa određenom rutom definirane u datoteci „App.js“. Sastoje se od komponenata za strukturu izgleda poput „Navbar“, „Sidebar“ i „DashboardContent“. „DashboardContent“ prikazuje ostale dinamične podatke zavisno o trenutnoj ruti i prikazan je slikom 23.

```
<div>
    <Navbar setAuth={props.setAuth}/>
    <Sidebar/>
    <div className="dashboard-container">
        <div className="dashboard-content">
            {props.children}
        </div>
    </div>
</div>
```

Slika 23. DashboardContent Komponenta

6.1.4. React stanje

Aplikacija koristi Redux kao sustav za spremanje podataka dobivenih iz baze podataka. Redux se sastoji od reduktora i akcija, gdje se pomoću akcija šalju podaci iz

aplikacije u reduktor skladište pomoću „store.dispatch“ funkcije. Za razliku od React stanja, Redux stanje je dostupno na globalnoj razini aplikacije. Primjer Redux stanja i akcije vidljivi su na slici 24 i 25, gdje se dohvaćaju podaci iz baze podataka i privremeno spremaju u reduktor. Aplikacija se ukupno sastoji od pet logično podijeljenih reduktora koji su spojeni pomoću „combineReducers“ funkcijom.

```
const initialState = {
    loading: true,
    notifications: {}
};

const reducer = (state : {loading: boolean, notifications: {}} = initialState, action) => {
    switch (action.type) {
        case "GET_USER_NOTIFICATIONS": {
            return {
                ...state,
                loading: false,
                notifications: action.payload,
            }
        }
```

Slika 24. Redux reduktor

Reduktor je funkcija koja prima akciju i prijašnje stanje aplikacije te vraća novu vrijednost stanja. Moguće je definirati koji dio stanja aplikacije treba mijenjati, a u slici 24 je vidljivo kako se mijenja stanje „loading“ koje je u početnom stanju true u suprotno stanje false. Notifikacije koje se nalaze u „action.payload“, spremaju se u objekt „notifications“ koji je u početnom stanju prazan.

```
export const getUserNotifications = () => async dispatch => {
    const token = localStorage.getItem( key: "token");
    try {
        const res = await fetch(
            input: "/notifications/getUserNotifications",
            init: {
                method: "GET",
                headers: {
                    "Content-type": "application/json",
                    "token": token
                }
            }
        );
        const parseRes = await res.json()
        if(res.ok){
            dispatch({
                type: "GET_USER_NOTIFICATIONS",
                payload: parseRes
            })
            return parseRes
        }
    } catch (err) {
        console.error(err.message);
    }
}
```

Slika 25. Redux akcija

Redux akcija vidljiva na slici 25 služi za hvatanje podataka o svim notifikacijama koje korisnik posjeduje pomoću fetch GET zahtjeva. Nakon uspješnog primanja podataka, poziva se „dispatch“ funkcija koja prosljeđuje podatke dalje u reduktor. Ovaj način primanja i spremanja podataka koristi se i u ostatku aplikacije.

6.1.5 Karta

Pomoću „react-leaflet“ paketa, prikazana je karta koja označuje lokaciju adrese korisnika sa markerom. Pošto karta prihvaća lokaciju u obliku koordinata, potrebno je adresu korisnika pretvorit u koordinate pomoću „OpenStreetMap“ GET zahtjeva. Ova radnja vidljiva na slici 26, odvija se tijekom registracije korisnika ili nakon mijenjanja adrese korisnika preko profilne stranice. Pretvaranje se vrši unutar funkcije „getCoordinates“ koja šalje informacije o adresi korisnika pomoću GET zahtjeva i nakon uspješnog povrata podataka, vraća podatke o koordinatama koji se dalje koriste u procesu izrade korisničkog računa.

```
// reverse geocoding (transform address to lat, long)
const getCoordinates = async () => {
    try {
        const res = await fetch(
            input: `https://nominatim.openstreetmap.org/search?q=
            ${formData.street}+${formData.street_number}+
            ${formData.city}&format=geojson`, init: {
            method: "GET",
        });
        const parseRes = await res.json();
        return parseRes.features[0].geometry.coordinates
    } catch (err) {
        console.error(err.message);
    }
};
```

Slika 26. Transformacija adrese

6.1.6. Sass

Sass je stilski jezik koji se prevodi u css. Omogućuje korištenje varijabli, ugniježdenih pravila, mješavina (engl. mixins), funkcija i još mnogo toga, a sve to s potpuno CSS kompatibilnom sintaksom. Sass pomaže u održavanju velikih stilova i olakšava dijeljenje dizajna unutar i između projekta [7]. Sav CSS korišten od aplikacije definiran je u App.scss datoteci. Korištenjem Sass pred procesora moguće je pisat običan CSS jezik na efikasniji način. Najkorišteniji mixin unutar aplikacije je „center“ koji centrira sve objekte unutar odabranog objekta pomoću flex-box svojstva. Spomenuti mixin vildjiv je na slici 27. Pomoću *CSS* medijskih upita (engl. media query) definirane su veličine pri kojima se mijenja izgled aplikacije ako veličina ekrana dostigne određenu vrijednost.

```
@mixin center {
  display: flex;
  justify-content: center;
  align-items: center;
  flex-direction: column;
}
```

Slika 27 mixin center

6.2. Poslužiteljska strana

Svrha poslužiteljske strane aplikacije je da dohvaća sve potrebne podatke iz baze podataka pomoću REST API arhitekture. Sve API rute su definirane u „routes“ direktoriju i podijeljene na više datoteka. „index.js“ datoteka se koristi za inicijalizaciju svih potrebnih okvira i paketa poput Express.js i inicijalizacija ruta. Slika 28 prikazuje navedeno.

```
const express = require("express")
const app = express();
const cors = require('cors')
const PORT = process.env.PORT || 5000;
const path = require("path");
const methodOverride = require('method-override')
app.use(cors());

app.use(express.static( root: 'public'));
app.use(express.json( options: { limit: '50mb' }));
app.use(express.urlencoded( options: { limit: '50mb', extended: true }));

app.use(methodOverride());

app.use(express.json()); // req.body

if (process.env.NODE_ENV === "production") {
    //serve static content
    //npm run build
    app.use(express.static(path.join(__dirname, "client/build")));
}

console.log(__dirname);
console.log(path.join(__dirname, "client/build"));

//ROUTES//

//register and login routes
app.use("/auth", require("./routes/jwtAuth"));
app.use("/dashboard", require("./routes/dashboard"));
app.use("/items", require("./routes/items"));
app.use("/rentedItems", require("./routes/rentedItems"));
app.use("/messages", require("./routes/messages"));
app.use("/profile", require("./routes/profile"));
app.use("/notifications", require("./routes/notifications"));

app.get("*", (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<ResBody, Locals> ) => {
    res.sendFile(path.join(__dirname, "client/build/index.html"));
})

app.listen(PORT, hostname: () => {
    console.log(`Server is starting on port ${PORT}`);
});
```

Slika 28 Datoteka index.js

6.2.1 Express.js Rute

Express.js rute se koriste za dodavanje, brisanje ili manipulaciju podataka spremljenih u bazi podataka. U aplikaciji postoji više datoteka sa rutama podijeljene prema ulogama. Na slici 29 vidljiva je ruta koja se pokreće sa klijentske strane pozivom definirane rute „/notifications/getUserNotifications“. Nakon poziva, ruta odrađuje svoju funkciju i vraća podatke zavisno o uspješnosti radnje.

```
router.get( path: "/getUserNotifications", authorization, async (req, res) => {
    const user_id = req.user;
    try {
        const notifications = await pool.query
        ("SELECT * FROM notifications WHERE notification_owner_id = $1",
            [user_id.id]);
        res.json(notifications.rows);
    } catch (err) {
        console.error(err.message);
        res.status( code: 500).send( body: "Server error");
    }
});
```

Slika 29. Express.js ruta

6.2.2. Baza podataka

Korištena je PostgreSQL relacijska baza podataka koja se sastoji od više tablica. Manipulacija, pretraživanje i filtriranje podataka u bazi vrši se pomoću SQL upita. Primjer upita vidljiv je na slici 30.

```
await pool.query("DELETE FROM items WHERE item_id=$1 AND item_owner=$2", [item_id, user_id]);
```

Slika 30. SQL DELETE upit

Ovaj SQL upit briše podatke iz tablice „items“ gdje je „item_id“ i „item_owner“ jednak predanim podacima. Izrada i upravljane bazom podataka učinjeno je lokalno pomoću „pgAdmin“ alata koji služi za interakciju sa bazom podataka.

Aplikacija koristi dvije nezavisne baze podataka, jedna na lokalnoj strani koja služi za testiranje prilikom izrade novih funkcionalnosti, a druga je objavljena zajedno sa aplikacijom na Heroku poslužitelju. Manipulacija i pregled podataka unutar tablice učinjena je pomoću navigatora baze podatka koji dolazi u sklopu korištenog IDE-a (engl. Integrated development environment).

6.2.3. Tablica korisnika

Tablica korisnika sa nazivom „users“, koristi se za spremanje svih podataka vezano za korisnika u bazu podataka. Svaki korisnik ima svoj id, email, ime i prezime, lozinku, adresu, županiju, koordinate i url od profilne slike. Za sve ostale radnje poput dohvaćanja ostalih podataka u vezi korisnika koristi se automatsko generirani uuid koji je također i primarni ključ navedene tablice. Isječak podataka iz tablice korisnika vidljiva je na slici 31.

user_street ▲ 1	user_city	user_state	lat	long	user_street_number
Andrije Štampara	Slavonski Brod	Brodsko-posavska	18.0232583	45.1521918	43

Slika 31. Isječak podataka korisnika

6.2.4. Cloudinary

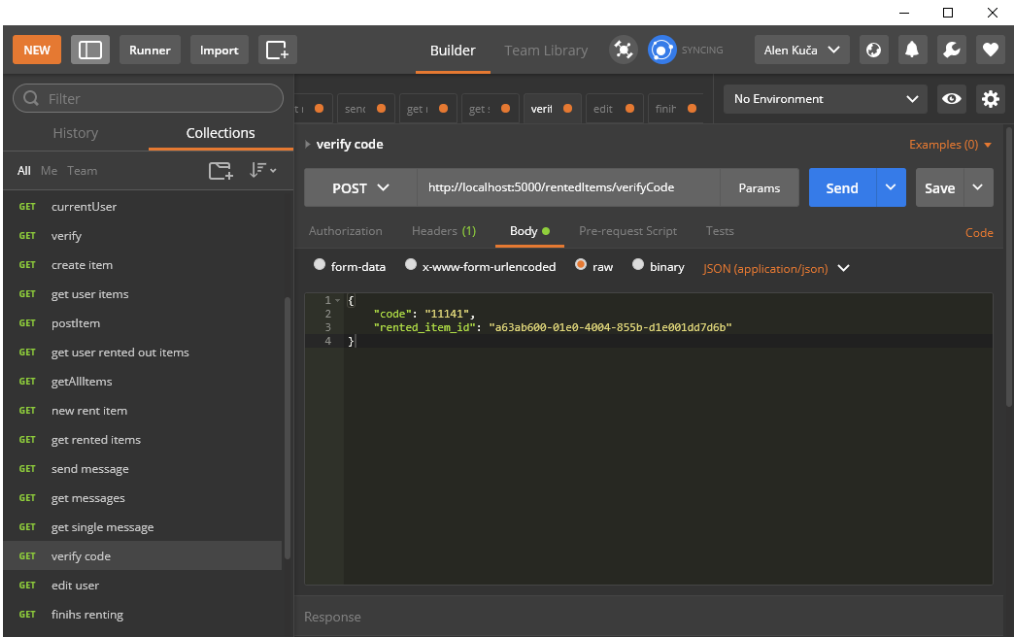
Pohranjivanje slika vrši se na Cloudinary poslužitelju koji ima dostupan besplatan paket za sve korisnike. Ograničenja su u količini pohranjenih i prenesenih podataka. Pohranjivanje se vrši preko definirane metode koja sprema sliku na Cloudinary poslužitelj te nakon toga pohranjuje url slike u određenu tablicu baze podataka aplikacije. Na slici 32 vidljiva je ruta korištena prilikom promjene profilne slike korisnika. Trenutni url slike se zamjenjuje url-om nove slike

```
router.post( path: "/changeAvatar", authorization, async (req, res) => {
    const user_id = req.user.id;
    const user_image = req.body.data;
    try {
        await pool.query("UPDATE users SET user_image=$2 WHERE user_id = $1", [
            user_id,user_image
        ]);
        res.json(user_image)
    } catch (err) {
        console.error(err.message);
        res.status( code: 500).send( body: "Server error");
    }
});
```

Slika 32. changeAvatar ruta

6.2.5. Postman

Prije korištenja, izrađeni HTTP zahtjevi testirani su pomoću Postman aplikacije kako bi se identificirale potencijale greške prije integracije sa poslužiteljskom stranom aplikacije. Ovaj način rada znatno olakšava i ubrzava identificiranje i ispravljanje grešaka. Primjer korištenja Postman-a vidljiv je na slici 33.



Slika 33. Postman

6.2.6. Heroku i Github

Sav kod aplikacije nalazi se na Github-u unutar glavne grane pod nazivom „main branch“. Pošto objavljivanje aplikacije zahtijeva dodatne postavke unutar aplikacije, napravljena je dodatna grana sa nazivom „deployment“, u kojoj se nalazi slična verzija aplikacija sa svim potrebnim promjenama i postavkama za objavljivanje.

Aplikacija zajedno sa bazom podataka je objavljena pomoću Heroku usluga i nalazi se na adresi „https://rentmi.herokuapp.com”.

7. Korisničke upute

7.1. Izrada računa i prijava

Kako bi korisnik mogao koristit aplikaciju, potrebno je da izradi račun na stranici za izradu računa. Potrebno je unijeti točnu adresu sa punim nazivom ulice kako bi se karta ispravno prikazala. Nakon uspješne izrade računa, korisnika se automatski preusmjerava na stranicu “Moje Stvari”. Prijavljen korisnik je validan 10 dana, te se nakon tog vremena ponovno mora prijavit preko stranice za prijavu gdje upisuje svoj email i šifru.

7.2. Dodavanje proizvoda

Korisnik ima opciju izradu vlastitih proizvoda kako bi iste mogao iznajmiti. Sve što je potrebo za izradu proizvoda je odlazak na stranicu “Moje Stvari” i klik na *Dodaj* koji se nalazi u gornjem desnom kutu. Nakon klika, otvara se prozor u kojem se traži ime proizvoda, cijena po danu, odabir kategorije i kratak opis proizvoda. U sljedećem i zadnjem koraku je ponuđena opcija za odabir slike sa uređaja korisnika. Klikom na *Objavi*, proizvod se uspješno dodaje te se korisnik preusmjerava na stranicu „Unajmljeno“.

7.3. Iznajmljivanje proizvoda

Sve što je potrebno kako bi korisnik iznajmio proizvod je otići na stranicu “Moje Stvari”, odabrat proizvod i kliknut na opciju *Objavi*. Proizvod postaje vidljiv ostalim korisnicima i omogućuje se za unajmljivanje. Kada proizvod bude unajmljen od drugog korisnika, vlasnik dobiva obavijest o uputama za dostavu.

7.4. Unajmljivanje proizvoda

Upisom u naslovnu traku ili klikom na gumb *Unajmi odmah*, korisnika se vodi na stranicu za pretraživanje gdje se nalaze sve opcije potrebne za filtriranje i detaljno pretraživanje proizvoda koji su trenutno objavljeni od ostalih korisnika. Vidljivi su i filteri koji omogućuju filtriranje proizvoda po županijama i kategoriji proizvoda. Klikom na željeni proizvod, korisnika se prosljeđuje na stranicu proizvoda gdje se nalazi glavni gumb sa nazivom "*Unajmi*". Klikom na *Unajmi*, korisnik odabire trajanje najma i vrstu dostave. Klikom na *Potvrdi*, naručuje se proizvod i obavještava se vlasnik o novom najmu. Prilikom dostave ili primitka proizvoda potrebno je predat šifru potvrde vlasniku proizvoda kako bi on započeo proces najama.

7.5. Završetak najma i vraćanje proizvoda

Tijekom najma, na stranici „Unajmljeno", i ako vrijeme najma nije isteklo, dostupna je opcija *Vrati ranije* koja pokreće proces povratka proizvoda. Korisnik u prvom koraku odabire vrstu dostavljana proizvoda, a u drugom koraku bude prikazana karta sa označenom adresom na koju je potrebno dostaviti proizvod bilo to osobno ili dostavnom službom. Ako je vrijeme najma isteklo pokreće se isti proces ali pomoću opcije *Vrati.* Nakon dostave proizvoda, vlasnik predaje šifru unajmitelju koja uspješnim unosom potvrđuje povratak proizvoda. Zadnji korak u povratku najma je kada vlasnik odabere opciju *Završi* koja je dostupna nakon uspješne potvrde povratka proizvoda.

7.6. Promjena adrese i profilne slike

Profil korisnika vidljiv je ostalim korisnicima ako trenutno ima barem jedan proizvod objavljen. Moguće je promijeniti trenutnu adresu sa novom, a to se čini na stranici vlastitog profila pomoću gumba *Uredi*. Mijenjanje korisničke adrese ne mijenja adresu proizvoda.

Ako Korisnik želi promijeniti adresu proizvoda potrebno je proizvod izbrisati i ponovno dodati.

7.7. Slanje i odgovaranje na poruke

Aplikacija omogućava razmjenu poruka kako bi korisnici mogli komunicirati o potencijalnim nejasnoćama. Za slanje poruke potrebno je posjetiti profil korisnika i kliknuti na opciju *Poruka* koja otvara novi prozor za unos naslova i teksta poruke, te nakon klika *Pošalji* poruka bude poslana odabranom korisniku. Poruke je moguće brisati pomoću gumba *Izbriši,* a opcija *Označi kao nepročitano* označuje poruku kao nepročitanu.

7.8. Notifikacije

Naslovna traka sadrži padajući prozor gdje su prikazane nedavne obavijesti. U prozoru se nalaze dvije opcije, *Izbriši* i *Prikaži sve*. Klikom na *Izbriši* obavijesti nestaju, a klikom na *Prikaži sve*, korisnika se vodi na stranicu sa svim obavijestima.

8. Zaključak

U svrhu diplomskog rada napravljena je web aplikacija pomoću PERN Stack tehnologija. Logika aplikacije podijeljena je na klijentski i poslužiteljski dio. React.js je korišten za izradu korisničkog sučelja i logike oko klijentskog dijela, a Node.js i Express.js za dohvaćanja podataka i ostale logike oko poslužiteljskog dijela. Baza podataka izrađena je sa PostgreSQL relacijskom bazom podataka i sastoji se od više tablica. Sav izgled aplikacije izrađen je pomoću SASS pred procesora i bez korištenja dodatnih CSS biblioteka. Struktura aplikacije izrađena je na način kako bi lagano bilo primijeniti nove funkcionalnosti.

Korisnici aplikacije mogu iznajmljivati vlastite ili unajmljivati tuđe proizvode i time uštedjeti ili zaraditi novac. Korisnici također mogu otkriti nepoznate proizvode i time razviti nove interese. Unutar aplikacije dostupna je komunikacija između korisnika pomoću sustava poruka. Obavijesti služe kako bi korisnik bio obavješten o stanju njegovih najmova i primljenih poruka. Kao što je spomenuto, u Hrvatskoj ne postoji aktivna stranica koja pruža ove usluge, ali u ostalim državama postoje slične i uspješne aplikacije koje imaju aktivne korisnike.

Pojedine funkcionalnosti aplikacije ograničene su zbog korištenja besplatnih servisa poput Cloudinary koji ima mjesečno ograničenje za prijenos medijskih datoteka i Heroku hosting koji stavlja objavljenu aplikaciju u stanje mirovanja nakon trideset minuta neaktivnosti. Funkcionalnost izračunavanja vremenske udaljenosti između korisnika nije bila moguća korištenjem OpenStreetMap besplatne karte i react-leaflet paketa.

Moguće su nadogradnje aplikacije sa dodanim funkcionalnostima poput sustava komentiranja i ocjenjivanje najmova, povratna poruka vlasniku sa ocjenom i stupnjem zadovoljstva nakon završetka najma, te opcija za ponudu kupnje proizvoda ili produženje najma. Izrada nove početne stranice za prikaz novih ili popularnih proizvoda u okolini korisnika i kalendar sa označenim danima događanja poput završetka najma ili povrat proizvoda.

Izrada web aplikacije pomoću PERN Stack tehnologija zahtijeva znanje iz JavaScript-a, SQL-a, HTML-a i CSS-a, a znatno je olakšano sa velikom količinom dostupnih materijala i resursa te pomoćnih alata poput Postman i pgAdmin.

9. Literatura

[1] React Documentation – a JavaScript library for building user interfaces, Facebook Inc., 2021, dostupno na: https://reactjs.org, [pristupljeno: 10.06.2021]

[2] Bugl Daniel (2019), Learn React Hooks – Build and refactor modern React.js applications using Hooks, dostupno na: https://books.google.hr/books?id=Tkm4DwAAQBAJ&printsec=frontcover&dq=react.js&hl=hr&sa=X&redir_esc=y#v=onepage&q=react.js&f=false, [pristupljeno: 05.09.2021]

[3] Wandschneider Marc (2017), Learning Node.js – A Hands-On Guide to Building Web Applications in JavaScript, dostupno na: https://books.google.hr/books?id=rBDGDQAAQBAJ&printsec=frontcover&dq=node.js&hl=hr&sa=X&redir_esc=y#v=onepage&q&f=false, [pristupljeno: 06.09.2021]

[4] Express web framework (Node.js/JavaScript), Mozilla and individual contributors, 2021, dostupno na: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs, [pristupljeno: 05.09.2021]

[5] Obe R. & Hsu L. (2018), PostgreSQL Up & Running – A practical guide to the advanced open source database, dostupno na: https://books.google.hr/books?id=Xj85DwAAQBAJ&printsec=frontcover&hl=hr&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false, [pristupljeno: 05.09.2021]

[6] PostgreSQL Documentation, The PostgreSQL Global Development Group, 2021, dostupno na: https://www.postgresql.org/docs, [pristupljeno: 15.07.2021]

[7] Sass Documentation, Sass team and numerous contributors, 2021, dostupno na: https://sass-lang.com/documentation, [pristupljeno: 12.06.2021]

Popis slika

Slika 1. SWOT analiza 4
Slika 2. Dijagram obrasca uporabe 5
Slika 3. Stranica za izradu korisničkog računa 9
Slika 4. Stranica za prijavu 10
Slika 5. Izrada novog proizvoda 11
Slika 6. Stranica „Moje Stvari“ 12
Slika 7. Stranica „Moje Objave“ 13
Slika 8. Stranica za pretraživanje 14
Slika 9 Stranica proizvoda 15
Slika 10 Način dostave 16
Slika 11. Trajanje najma 16
Slika 12. Napomene o najmu 17
Slika 13. Unos šifre najma 18
Slika 14. Unajmljeni proizvodi 19
Slika 15. Stranica "Iznajmljeno" 20
Slika 16. Stranica statistike 21
Slika 17. Stranica sa porukama 22
Slika 18. Nova poruka 23
Slika 19. Obavijesti 24
Slika 20. Struktura direktorija poslužiteljske strane 26
Slika 21. Rute 27
Slika 22. Komponenta „pageTitle“ 28
Slika 23. DashboardContent Komponenta 28
Slika 24. Redux reduktor 29
Slika 25. Redux akcija 30
Slika 26. Transformacija adrese 31
Slika 27 mixin center 31
Slika 28 Datoteka index.js 32
Slika 29. Express.js ruta 33
Slika 30. SQL DELETE upit 33
Slika 31. Isječak podataka korisnika 34
Slika 32. changeAvatar ruta 35
Slika 33. Postman 35