Razvoj web aplikacije za vrednovanje izvođača građevinskih radova

Tomljanović, Marko

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:137:630250

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-16**

Repository / Repozitorij:

Digital Repository Juraj Dobrila University of Pula

Sveučilište Jurja Dobrile u Puli

Fakultet Informatike u Puli

MARKO TOMLJANOVIĆ

RAZVOJ WEB APLIKACIJE ZA VREDNOVANJE IZVOĐAČA GRAĐEVINSKIH RADOVA

Diplomski rad

Pula, rujan, 2021. Godine

Sveučilište Jurja Dobrile u Puli

Fakultet Informatike u Puli

MARKO TOMLJANOVIĆ

RAZVOJ WEB APLIKACIJE ZA VREDNOVANJE IZVOĐAČA GRAĐEVINSKIH RADOVA

Diplomski rad

JMBAG: 0165060109, izvanredni student

Studijski smjer: Diplomski sveučilišni studij 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 Marko Tomljanović, 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 nedozvoljeni 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, 2021 godine

Student

Marko Tomljanović

IZJAVA O KORIŠTENJU AUTORSKOG DJELA

Ja, Marko Tomljanović dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj diplomski rad pod nazivom „Razvoj web aplikacije za vrednovanje izvođača građevinskih radova“ 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, 2021 godine

Potpis

Marko Tomljanović

SAŽETAK

Tema diplomskog rada je oblikovanje web aplikacije „ocijeni Majstora“ koja će omogućavati vrednovanje izvođača građevinskih radova. Aplikaciju mogu koristiti svi koji planiraju graditi ili adaptirati stambene ili gospodarske objekte. Onima koji tek planiraju angažirati usluge izvođača aplikacija pomaže tako što imaju uvid u komentare i ocjene korisnika kojima je izvršena usluga. Nakon što usluga izvođača bude gotova, korisnici mogu komentirati i ostaviti željenu ocjenu. Bitno je da svaki put kada registrirani korisnik ocjeni firmu, ta firma dobiva obavijest o zaprimljenom komentaru. To je važno jer su tako obje strane informirane o svim događanjima u aplikaciji. Ocjenjivanje je uveliko olakšano zbog mogućnosti prijave preko vlastitog Google ili Facebook računa. U nastavku rada biti će objašnjene sve tehnologije koje su korištene, to su HTML, CSS, JavaScript, Vue.js, Node.js, Firebase baza, Bootstrap i BootstrapVue okviri. Govoriti će se o popratnim funkcionalnostima koje poboljšavaju iskustvo samog korisnika, kao što je prosječna cijena izvođača, portfolio u obliku galerije slika, poveznice na društvene mreže, email sustav obavještavanja i ostale funkcionalnosti. Objasniti će se što je i kako se koristi Netlify, kao i program za izradu prototipa sučelja Figma. Nakon implementacije i dijagrama u radu će detaljno biti objašnjeni koraci kako se koristi aplikacija pomoću slika sučelja web aplikacije.

KEY WORDS: HTML, CSS, JavaScript, Vue.js, Node.js, Firebase, Bootstrap, Netlify, Figma

ABSTRACT

The thesis is structured around forming web app “ocijeni Majstora” for rating construction workers and firms. The app gives benefit for those who are planning to build or renovate residential and work space. Users can see comments and ratings of preceding recipient of the app. Once the contractor’s work is complete, users can comment and leave the rating for them. It is important that every time a registered user evaluates a company, that company receives a notification about the received comment. This is important because this way both sides are informed about all the reviews in the application. Rating of the construction workers is simplified through login with Google or Facebook account. The thesis will also explain used technologies which are HTML, CSS, JavaScript, Vue.js, Node.js, Firebase base, Bootstrap and BootstrapVue frames. Afterwards, we will discuss accompanying functionalities which improve the user experience, such as, arithmetic mean of price, portfolio, gallery, email notify and links to construction firms social networks. The thesis will also discuss how to use Netlify and Figma, program for interface production. Afterwards, the thesis will precisely explain the steps of the app usage via pictures of interface of web app.

KEY WORDS: HTML, CSS, JavaScript, Vue.js, Node.js, Firebase, Bootstrap, Netlify, Figma

SADRŽAJ

1. UVOD 1
2. TEHNOLOGIJE KORIŠTENE U APLIKACIJI 3
2.1 Arhitektura web aplikacije 3
2.2. HTML 4
2.3. CSS 5
2.4. Node.js 5
2.5. JavaScript 6
2.6. Vue.js 7
2.7. Google Firebase 9
2.8. Bootstrap i BootstrapVue 10
3. RAZRADA FUNKCIONALNOSTI 11
3.1. Dijagram slučaja upotrebe 11
3.2. Dijagram klasa 13
3.3. Prototip sučelja – Figma 15
3.4. Netlify platforma 17
4. IMPLEMENTACIJA 18
4.1. Prijava i registracija 18
4.2. Baza podataka – Firebase 20
4.2.1. Authentication 21
4.2.2. Firestore 22
4.2.3. Storage 26
4.3. Registracija firme 27
4.4. Profil i njegove funkcionalnosti 30
4.4.1. Ocjenjivanje 32
4.4.2. Prosječna cijena usluge 33

4.4.3. Galerija slika 34
4.4.4. Brisanje komentara 35
4.4.5. Opcija – slažem se 35
4.5. Dinamičke rute 36
4.5.1 404 error 37
4.6. Filtriranje izvođača 38
4.7. Iskustva korisnika za stranicu 40
4.8. Email sustav obavijesti 42
4.9. Informacije 43
4.10. Podnožje stranice 45
4.11. Mobilna verzija 46
4.12. NPM paketi 49
5. DRUŠTVENE MREŽE 52
6. ZAKLJUČAK 53
LITERATURA 54
POPIS SLIKA 56

1. UVOD

Ocijeni majstora je web aplikacija koja ima za cilj pojednostaviti praćenje kvalitete usluga izvođača građevinskih radova. Web aplikacije su danas široko primijenjene te većina aplikacija ima razvijenu web verziju iste aplikacije zbog brzog i jednostavnog pristupa. Također uz jednostavnost pristupa tu je bitna činjenica da nema nikakvih zahtjeva i pripreme za korištenje web aplikacija. Nema potrebnih ažuriranja i ne mora ih se pohranjivati lokalno na računalo nego sve nalazi na poslužitelju. Potrebna je samo brza internetska veza koja je danas neizostavni faktor svakog kućanstva. Postoje i nedostaci ovakve vrste aplikacija. Vrlo su rijetke, ali kada pričamo o web aplikacijama uvijek se spominje sigurnost kao najveći neprijatelj. Sigurnost je bitna, no nažalost virusi su prisutni bilo gdje na internetu i to je nešto na što možemo utjecati antivirusnim programima. Drugi nedostatak je brzina interneta kao i dostupnost poslužitelja, te se taj nedostatak u donašanje vrijeme jako malo pojavljuje. Stoga ako promatramo prednosti i nedostatke dolazimo do zaključka kako je korisniku puno bolje i brže u današnje vrijeme koristiti web aplikacije.

Na tržištu nema dostupnih sličnih rješenja koja djeluju na našem području. Aplikacije koje nude takva rješenja su orijentirane na druge funkcionalnosti, kao što su izračun cijena i slično. Ocjeni majstora je zamišljena da prati isključivo vrednovanje firmi u svrhu boljeg odabira istih. Svaka firma imati će mogućnost predstaviti se i objaviti svoj portfolio u obliku galerije slika koja je smještena na profilu. Sama aplikacija mogla bi privući veliki broj ljudi koji imaju u planu graditi, adaptirati i slično. Na početnoj stranici imamo mogućnost brzo i jednostavno odabrati željenu kategoriju ili firmu. Također se nalazi ulaz (eng. input) za pretraživanje koji je interaktivan i odmah predlaže tražene rezultate. Od sada će se u radu koristi izraz input. U radu Postoji i opcija top 5, koja nudi pretraživanje firmi na osnovu županija tako što prikazuje pet najbolje ocjenjenih izvođača u odabranoj županiji. U samom profilu firme su svi osnovni podaci firme. Također i djelatnosti koje obavljaju i linkovi na sve društvene mreže koje su priložene uz registraciju firme. Na sučelju profila sa lijeve strane je prikazan podatak kolika je prosječna cijena usluga u obliku trake napretka (eng. progress bar). Od sada će se u radu koristi izraz progress bar. Ostavljanje osvrta je vrlo jednostavno kako bi se svi mogli koristiti aplikacijom. Isto tako sama prijava je olakšana jer uz običnu prijavu email-om i lozinkom, postoji mogućnost prijave preko

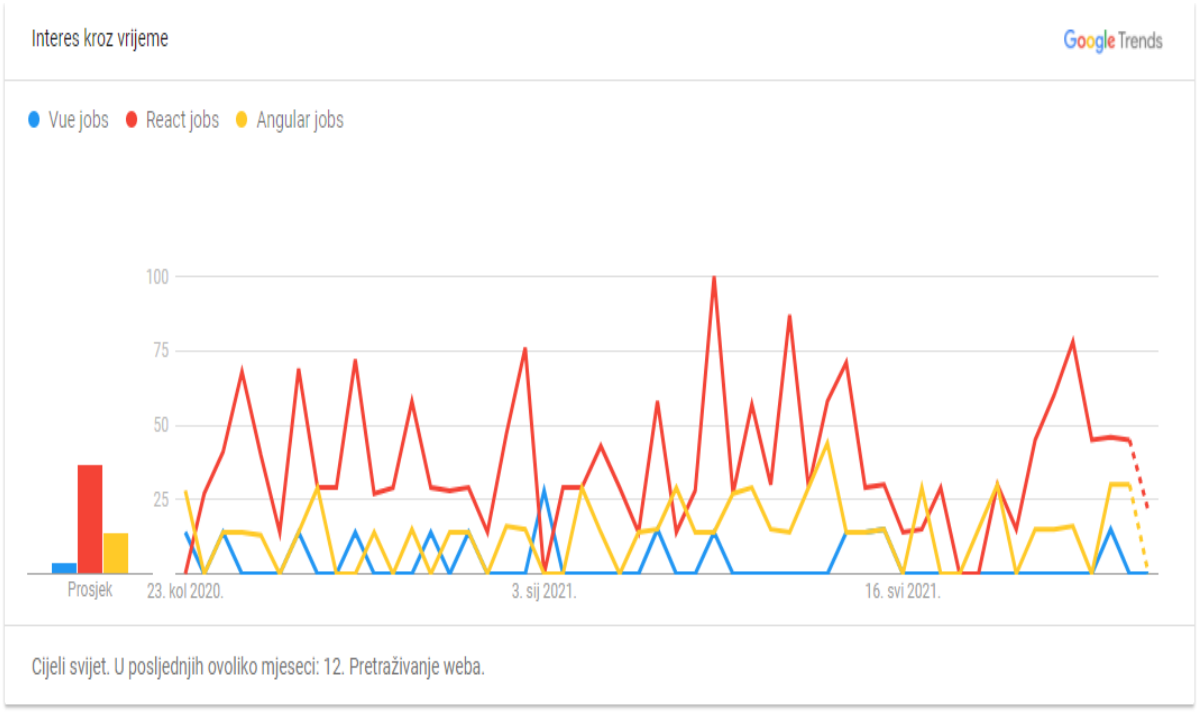
Google i Facebook računa. Tijekom izrade, aplikacija je konstantno ažurirana na GitHub profil gdje se mogu postaviti dodatna pitanja vezna uz samu web aplikaciju.

GitHub profil je na poveznici https://github.com/Marko-Tomljanovic/diplomski_rad

Cilj ovog rada je objasniti i prikazati aplikaciju uz sve njene funkcionalisti kako bi se prezentirala na što prihvatljiviji način. U slijedećem tekstu bit će iscrpno obuhvaćeni svi aspekti web aplikacije.

2. TEHNOLOGIJE KORIŠTENE U APLIKACIJI

Tehnologije koje su korištene u izradi aplikacije su HTML, CSS, Node.js, JavaScript, Vue.js i za bazu podataka Google Firebase. Između svih popularnih web tehnologija odlučeno je koristiti JavaScript i Vue.js. Trenutno je Vue.js u usponu i vrlo je popularan i jednostavan za korištenje. Po statistici Google Trends, okvir (eng. framework) Vue.js je najmanje zastupljen. Dok u GitHub statistici Angular pada u polarnosti, za razliku od njega Facebook-ov React i Vue.js su vrlo slični po popularnosti i korištenju. Slika *1* prikazuje graf u posljednjih 12 mjeseci za poslove koji se obavljaju u određenim tehnologijama [1]. Zaključno, Vue.js mlad na tržištu i nastaviti će se širiti njegova popularnost. U nastavku će biti detaljno razrađena povijest svih tehnologija koje su primijenjene u ovom radu.

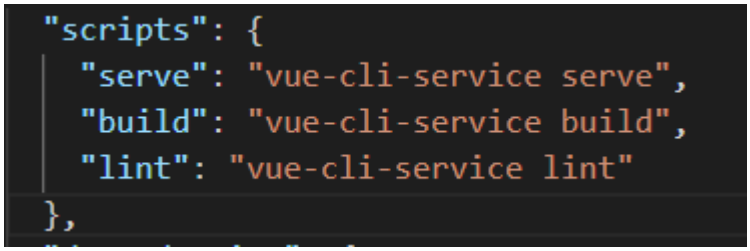


Slika 1. Interes kroz vrijeme [1]

2.1 Arhitektura web aplikacije

Aplikacija je pisana u Visual Studio Code programu. To je program koji podržava razne programske jezike. Program ima raznih dodataka (eng. plug-in) koji uveliko olakšavaju pisanje koda. Za svaki dio napisanog koda je ažuriran kod (eng. commit) na GitHub profilu. Od sada će se u radu koristi izraz commit. Stoga svakom elementu i funkcionalnosti koja je dodana u aplikaciju možemo direktno pristupiti preko

branch-a na GitHub profilu. Za upotrebu GitHub-a korištena je njihova desktop aplikacija. Aplikacija je efikasna i jednostavna te puno brža od klasičnog načina ažuriranja koda na GitHub. Za potrebu smještaja aplikacije (eng. hosting) korištena je Netlify platforma. Od sada će se u radu koristi izraz hosting. To je web platforma koja nudi besplatne usluge hostinga te surađuje sa GitHub-om. Nakon svakog commita na GitHub, automatski se vrši ažuriranje na Netlify platformi. Prije nego se izvrši commit, aplikacija se testira i pokreće lokalno na računalu. Nakon instalacije Node.js i Vue.js u terminal od Visual Code Studio upisujemo naredbu koja nam pokreće definiranu naredbu u „package,json“. Naredba je „npm run serve“ ili „npm run dev“. Na *slici 2* vidimo dio koje se pokreće prilikom upisa naredbe. Zatim se server pokreće te dobivamo adresu preko koje možemo pristupiti aplikaciji. Početak adrese počinje http://localhost:8080/, no ako postoji više pokrenutih onda će biti http://localhost:8081/, i tako nadalje.



Slika 2. package.json datoteka

2.2. HTML

HTML je prezentacijski jezik pomoću kojeg izrađujemo web stranice. Što je ujedno i naziv od kuda dolazi kratica (eng. HyperText Markup Language). Prvi put se spominje 1991. godine. Tada je bio vrlo jednostavan, pa sve do prve verzije 1993. godine. Tim Berners-Lee sa svojom organizacijom World Wide Web Consitorium poznatiju kao W3C su izdali spomenutu prvu verziju HTML-a. No i tada je bio poprilično ograničen. Tim Berners-Lee je bio autor HTML-a, te je i izumio World-Wide-Web 1989. godine u CERN-u [2]. Danas imamo inačicu 5.0 koja nam nosi brojne novitete. Za usporedbu sa 4.0 inačicom donosi mogućnost pregledavanja video sadržaja bez Adobe flash i ostalih, povuci i pusti (eng. drag and drop) i sličnih funkcionalnosti. Važno

je napomenuti da to nije programski jezik koji će nam pomoći izračunati nekakvu operaciju, nego služi samo za opis hipertekstualnih dokumenata. Osnovno obilježje HTML-a je da ima znakove koji nam govore kako će nešto izgledati i na koji način bi se trebalo prikazivati u našem pregledniku. Znakovi obično dolaze u paru, te drugi dio para ima zatvarajuću ulogu u obliku kose crte unutar zagrada. Ekstenzije HTML dokumenata su *.html i *.htm. U radu korišten je Vue.js koji je vrlo jednostavan i odlično ukomponiran sa HTML-om.

2.3. CSS

Kao i HTML, CSS je dugo u upotrebi. CSS je skraćenica od Cascading Style Sheets. Na početku razvoja HTML-a su korišteni tagovi koji definiraju jedan dio izgleda. Ubrzo je došla potreba za stilskim jezikom koji će obuhvatiti sve u jednu cjelinu. Pa dolazimo do zaključka da CSS definira sam izgled i uređuje raspored stranica. Ima vrlo jednostavnu sintaksu koja se sastoji od selektora i opisanog bloka koji je u vitičastoj zagradi. Selektor označava dio na koji će se primijeniti napisani CSS kod. Možemo CSS kod spremiti u posebni dokument pod ekstenzijom .css i povezati na željeni HTML dokument. To radimo zbog preglednosti. Također se može upisivati direktno u HTML dokument unutar elemenata <style>. Treći način bi bio da se direktno upiše unutar željenog elementa, kada želimo promijeniti samo boju i slično. Tada bi unutar elementa dodali atribut style=“color: blue“. Danas se često koristi i SCSS koji je podržan od strane CSS-a i puno je napredniji. SCSS sadrži sva obilježja CSS-a i sadrži puno novih obilježja zbog čega je programerima prvi izbor kada je riječ o stilskim jezicima. U radu je CSS korišten za uređenje većine elemenata i dok je isto tako služio za uređivanje i nadogradnju već uređenih elemenata iz Bootstrap okvira.

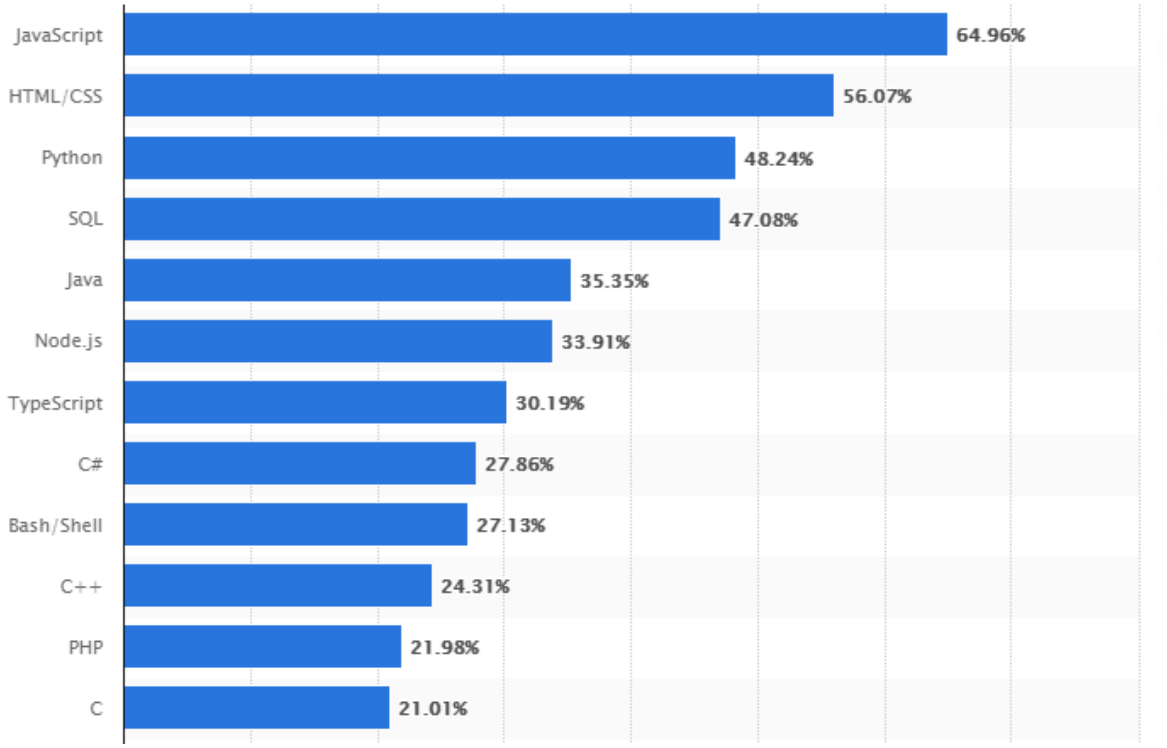
2.4. Node.js

Node.js je među novijim korištenim tehnologijama koje su korištene u radu. Razvio ga je Ryan Dahl 2009. godine. Naknadno je Node.js postao projekt otvorenog koda (eng. open-source) kojeg su mogli svi nadograđivati i poboljšavati. Od sada će se u radu koristi izraz open-source. Zbog toga mu je poprilično ubrzan razvoj. Node.js

je napravljen na V8 JavaScript engine-u [5]. Važno je kako Node.js omogućava rad sa JavaScriptom i izvan Internet preglednika. Najveća karakteristika je što je brz u odnosu Apache server. Također ima potpuni pristup datotekama kao i svaka druga aplikacija. Što omogućava direktno pisanje ili čitanje iz datoteka. Node.js ima jednu nit što znači da je baziran na događajima, ne na nitima. Isto tako je podržan na svim platformama, Windows, Mac, Linux, što je velika prednost uz već spomenuti open-source. Brzo se širi i počinje se sve više koristiti u svijetu. Primarni zadatak Node.js je omogućiti jednostavan i siguran način za realizaciju aplikacije značajnih performansi u JavaScriptu. U ovom projektu je instaliran i služi za pokretanje naredbi „npm run serve“ te instaliranje npm paketa.

2.5. JavaScript

JavaScript je trenutno među najpopularnijim skriptnim jezicima na svijetu. Kreator JavaScript-a 1995.godine je Brendan Eich iz Netscape-a. Na početku se skriptni jezik zvao Moha. Kasnije je preimenovan u LiveScript, te je na kraju u JavaScript. Samo ime skriptnog jezika podsjeća na objektno orijentirani jezik Java, no tu nema nikakve poveznice. Za usporedbu sa PHP-om ili drugim skriptnim jezicima, JavaScript se može ubaciti u bilo koju web stranicu. JavaScript je interpreter, što bi ukratko značilo da se skripta odrađuje istog trenutka funkciju po funkciju, bez prethodnog prevođenja programa i kreiranja izvršne datoteke. JavaScript se može izvršavati u bilo kojem poslužitelju pomoću web konzole. Prema Wikipediji JavaScript koristi preko 97% web stranica na internetu za barem jedan dio web stranice. JavaScript ima i svoje mane. Jedna od njih je klijentska sigurnost gdje se programske pogreške i pregledi mogu iskoristiti u loše svrhe. U radu je korišten u kombinaciji sa Vue.js i Node.js. U ovom radu je vrlo jednostavno sve povezano sa JavaScriptom, korišten je za povezivanje sa bazom podataka. Sa Google Firebase-om ima odličnu povezanost i kompatibilnost. Također u Firebase dokumentaciji je vrlo opširno i jednostavno pojašnjeno kako tumačiti i koristiti JavaScript za povezivanje sa Firesotre-om ili nekim drugim Firebase-ovim uslugama. Jednostavno se koristi i ima prilično široki spektar svojeg djelovanja. Slika 3 prikazuje najkorištenije programske jezike među programerima širom svijeta 2021. godine [3].



Slika 3. Najčešće korišteni programski jezici među programerima širom svijeta [3]

2.6. Vue.js

Vue.js je open-source JavaScript okvir za stvaranje dinamičkih stranica. Dopušta uključivanje raznih biblioteka. Vue.js je razvijen 2014.godine a osmislio ga je Evan You. On je u početku radio sa Angular okvirom u Google-u. Pogodan je za izradu modernih single-page aplikacija. Vue.js je spoj Angular i React okvira te je zbog toga vrlo popularan među programerima. Vue.js ima svoju sintaksu koja je jasno definirana. Sintaksa je sastavljena od tri oznake te svaka ima svoju ulogu. Prva oznaka je <template> gdje se piše HTML kod. Zatim <script> u kojem se nalazi JavaScript kod. I na kraju <style> gdje se nalazi stilski jezik. Slika 4 jasno prikazuje sintaksu okvira. Vidi se kako je raspored elemenata jasno raspoređen unutra tri već spomenute oznake.

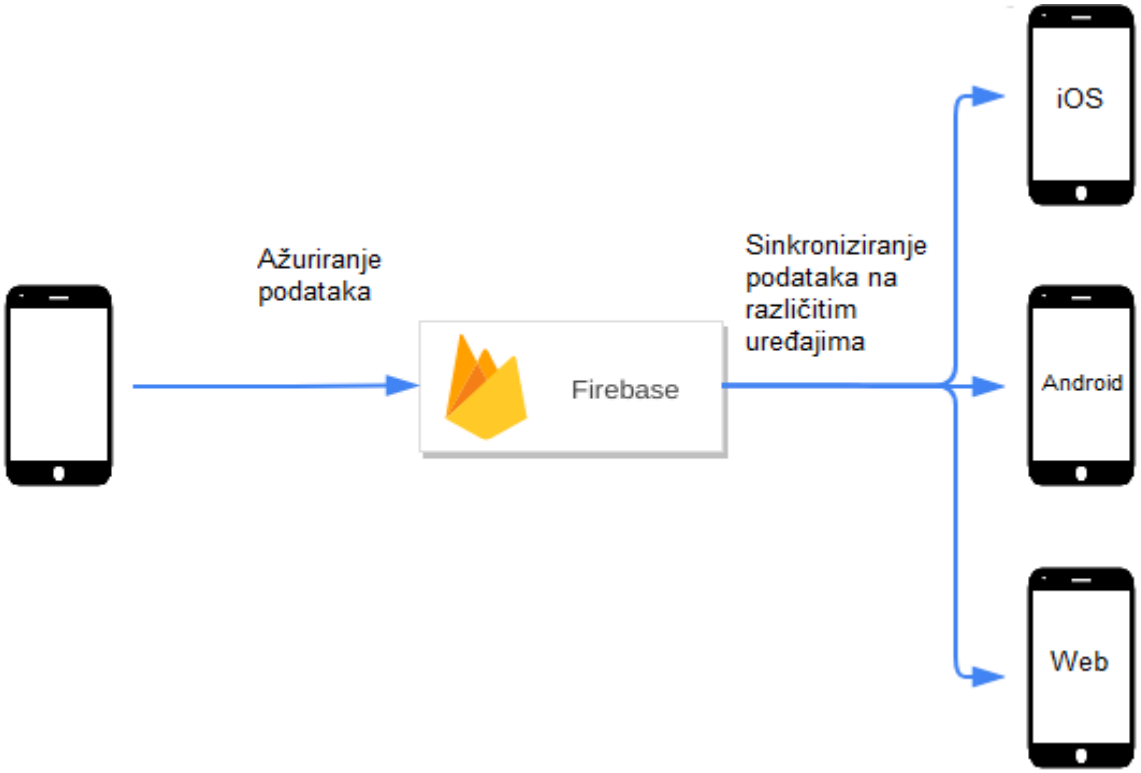
Slika 4. Sintaksa Vue.js

Na slici se vidi kako je u <template> djelu napisan HTML kod I jedna Vue.js direktiva, v-model koji je važan u Vue.js okruženju. V-model poprima input sa korisničke strane i sprema se u varijablu uz v-model. Da bi to radilo, potrebno je kreirati praznu varijablu u data() instanci koji se nalazi u <script> oznaci. Uz v-model postoje i druge direktive koje su od iznimnog značaja za programere. Jedne od najčešće korištenih direktiva su: v-bind, v-for, v-if, v-on. Spomenuta data() instanca nije jedina, tu su: components(), methods(), computed(), itd. Svaka instanca ima svoju ulogu u izradi stranice. Ono što Vue.js čini stvarno snažnim okvirom jesu komponente. Komponente su prilagođeni elementi koji se pozivaju u Vue.js dokumentu kao zapakirani element. U tu svrhu se najčešće koristi v-for direktiva kako bi taj isti element mogli ponavljati onoliko puta koliko želimo. Najčešća primjena je u karticama, slikama i ostalim nabrajanjima.

2.7. Google Firebase

Platforma koju su osnovali James Tamplin i Andrew Lee 2011 godine. Iako se misli da je od početka bila Google-ov proizvod, to nije točno. Razvijen je od firme zvane Envolve. Envolve je pružio programerima API koji omogućava integraciju funkcionalnosti internetskog čavrljanja (eng. chata) u njihove web starince. Kasnije, 2014 godine je Google otkupio Firebase [4]. U početku je bio cilj da Firebase postane platforma za pružanje usluga mobilnim programerima. Tako se i nastavio razvijati, no postao je vrlo pogodan i za web stranice koje su sve više koristile Firebase usluge. Danas Firebase podržava više programskih jezika, a to su: C++, Java, JavaScript/Node.js, Objective-C i Swift. Danas Firebase koristi preko tri miliona aktivnih aplikacija svaki mjesec [6]. Firebase je integrirao više različitih sustava, najpoznatiji njihov proizvod je Firestore. Za razliku od Realtime baze, Firestore prikuplja podatke u takozvane kolekcije koje zamjenjuju tablice. Firebase također ima mogućnost autorizacije, koja također nosi brojne mogućosti kao što su autorizacija preko mobilnog uređaja, potvrda email-om, Google ili Facebook račun itd. Slike se pohranjuju u Storage, gdje se slici može jednostavno pristupiti jer su organizirane u datotekama. Firebase također ima opciju za hosting, svoje vlastite funkcije i ostale funkcionalnosti. Postoje isto tako i ekstenzije koje olakšavaju rad sa bazom. Jedna od popularnih ekstenzija je Trigger Email koja dopušta slanje Email-a ovisno o podacima koji se pohrane u kolekciju. Platforma je besplatna za korištenje, no postoji i plaćena verzija. Besplatna verzija dopušta određen broj čitanja dokumenta iz pohranjenih kolekcija. Postoji dnevni limit od 50.000 čitanja. Ako je korisniku potreban veći broj čitanja potrebno je prijeći na plaćenu verziju. Naknade su izuzetno male te uz prelazak na plaćeni plan Firebase, dobiva se mogućnost korištenja svih Firebase funkcija.

U ovom radu korištena je funkcionalnost autorizacije. Moguće je registrirati i prijaviti se pomoću vlastitog Google ili Facebook računa. Isto tako korišten je Firestore Database kao baza podataka gdje su spremljeni svi podaci firme te komentari i ocjene korisnika. Osvrti korisnika su spremljeni na način da je načinjena pod-kolekcija koja se nalazi u glavnoj kolekciji. Tako se smanjuje dnevni broj čitanja iz baze, jer se jednim čitanjem smatra dohvaćanje jednog dokumenta. Storage je korišten za spremanja slika. Tamo su smještene slike profila firmi, te portfolio galerija slika uredno raspoređena po datotekama. Slika 5 prikazuje sinkroniziranje podataka direktno iz Firebase-a dok u tradicionalnom načinu zahtjev prvo ide na server.



Slika 5. Firebase ažuriranje podataka

2.8. Bootstrap i BootstrapVue

Bootstrap je jedan od najpopularnijih CSS okvira za izradu responzivnih i mobilnih web stranica. Najnovija inačica Bootstrap-a je 4.0, dok za BootstrapVue je 2.21.2. Kako je danas Vue.js jedan od najpopularnijih JavaScript okvira tako postoji i CSS okvir koji je prilagođen tim potrebama. Bootstrap ima veliku biblioteku koja nam nudi pregršt različitih komponenti koje se konstanto ažuriraju sa novim izgledom. Instalacija je vrlo jednostavna i postoji ih više vrsta. Ovisno o programeru moguće je izabrati instalaciju preko npm paketa, kopirati poveznicu na paket ili ručno skinuti datoteku koju je potrebno ubaciti u projekt. Kada jednom povežemo Bootstrap sa projektom, potrebno je pronaći željeni element u Bootstrap dokumentaciji i ubaciti u projekt određeni tag željenog elementa sa potrebnim atributima. U ovom projektu je instaliran npm paket BootstrapVue koji je prisutan u cijelom projektu. Uz njega dolaze dodatne funkcionalnosti koje se odlično slažu sa Vue.js i tako uveliko olakšavaju rad sa Bootstrap elementima. Najveći nedostatak u odnosu na ostale okvire za Vue.js kao što su Quasar i Vuetify, je taj što je teže uređivati elemente. Bootstrap ima svoju politiku vezano za paletu boja, veličine i slične stilske funkcije.

3. RAZRADA FUNKCIONALNOSTI

Aplikacija ocjeni Majstora prepoznaje dvije vrste korisnika, svaki ima određenu mogućnost pregleda i operacija u aplikaciji

- Autorizirani korisnik
- Nepoznati korisnik (eng. Anonymous)

Autorizirani korisnik je registriran sa svojim podacima na aplikaciji ili preko društvene mreže. Temeljem prijave ima pravo na svakom profilu firme komentirati i ostavljati ocjene za napravljene usluge. Također ima mogućnost dodati svoju firmu gdje mora upisati sve relevantne podatke firme kako bi se ista verificirala od strane admina web aplikacije. U ostatak aplikacije ima isti uvid kao i nepoznati korisnik.

Nepoznati korisnik je neregistrirani korisnik koji ima uvid u sve profile firmi. Može pregledati komentare i vidjeti prosječne ocjene, no nema pravo pristupa komentiranju i ostavljanju osvrta izvođačima.

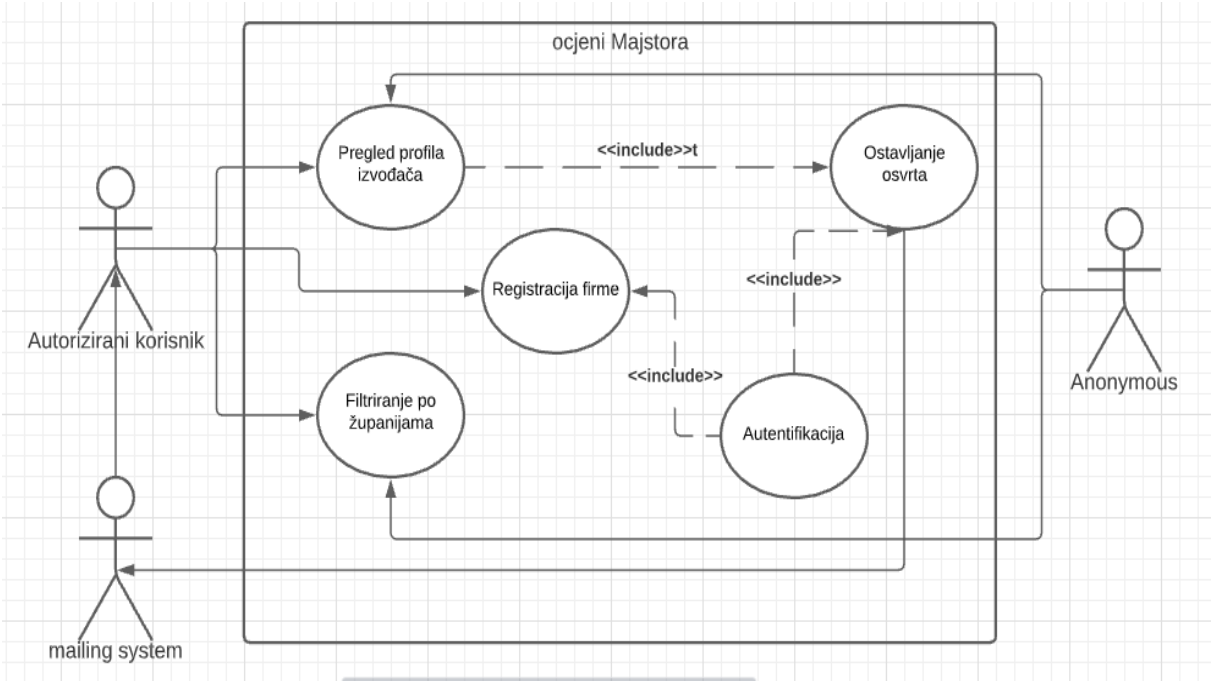
3.1. Dijagram slučaja upotrebe

Na temelju svega navedenog u poglavlju razrada funkcionalnosti, definiran je dijagram slučaja upotrebe (eng. use case). Od sada će se u radu koristi izraz use case diagram. Dijagram je napravljen u programu koji se zove LucidChart. Use Case model služi za jednostavan prikaz najvažnijih stvari koje aplikacija obavlja. Dijagram predstavlja funkcije visoke razine i način na koji će se korisnik kretati sustavom. Use case dijagram je ključni koncept jezika jednostavnog komuniciranja (eng. Unifited Modeling language) [7]. To je prvi korak izrade aplikacije, stoga treba u grubo postaviti temelje aplikacije. Nazivi trebaju biti specifični i jednoznačno dati do znanja o čemu se radi kao i točnu namjenu sustava aplikacije. Use Case dijagram sadrži aktere koji su bitan dio naše aplikacije. Akter nije jedna osoba, već se generalizira kao jedan dio korisnika koji koristi aplikaciju. Akter može biti organizacija, ili neki drugi vanjski sustav koji nije dio aplikacije. U dijagramu postoje veze koje komuniciraju sa elementima unutar dijagrama. A to su:

- asocijacija
- include
- extend
- nasljeđivanje

Jedna od tih veza je asocijacija, to je veza između aktera i samog Use Case. Asocijacija ima smjer koji obilježava tko je inicirao tu komunikaciju. Sljedeća bitna veza komunikacije u dijagramu je include koja nam definira da jedan obrazac obavezno uključuje dugi obrazac. Include također ima svoj smjer i ide po principu koji use case uključuje drugi use case. Razlikuje se od veze asocijacija tako što je prikazana isprekidanom linijom. Extend veza nam povezuje use case koji nije obavezan. Dakle korisnik je može inicirati, ali nije obavezan. Također je prikazana isprekidanom linijom i ima svoj smjer. Veza nasljeđivanja je prikazana sa neispunjenom strelicom, to jest bijelim trokutom. Koristimo je kada određeni obrazac možemo proširiti na dva ili više podobrazaca od kojih se jedan mora izvršiti.

Za Use Case dijagram aplikacije prvi akter biti će autorizirani korisnik, drugi nepoznati korisnik a treći sustav elektroničke pošte (eng. mailing system). Autorizirani korisnik komunicira sa vezom asocijacije obrascem pregleda profila izvođača. Pregled profila je include vezom spojen sa obrascem osvrta. Svaki autorizirani korisnik koristi aplikaciju da bi ostavio osvrt ili da registrira svoju firmu. Stoga vezom asocijacije autorizirani korisnik komunicira sa obrascem registracije firme. Sustav elektroničke pošte se aktivira kada dođe do ostavljanja osvrta, tada autorizirani korisnik prima email. Autorizirani korisnik također komunicira sa vezom asocijacije sa obrascem filtriranje županija. Postoji obrazac autentifikacije kako bi se napravila razlika između nepoznatog korisnika i autoriziranog korisnika. Nepoznati korisnik je povezan asocijacijom sa obrascima pregled profila izvođača i filtriranje po županijama. Slika 6 prikazuje UML Use Case dijagram aplikacije koji je napravljen u LucidChart programu.



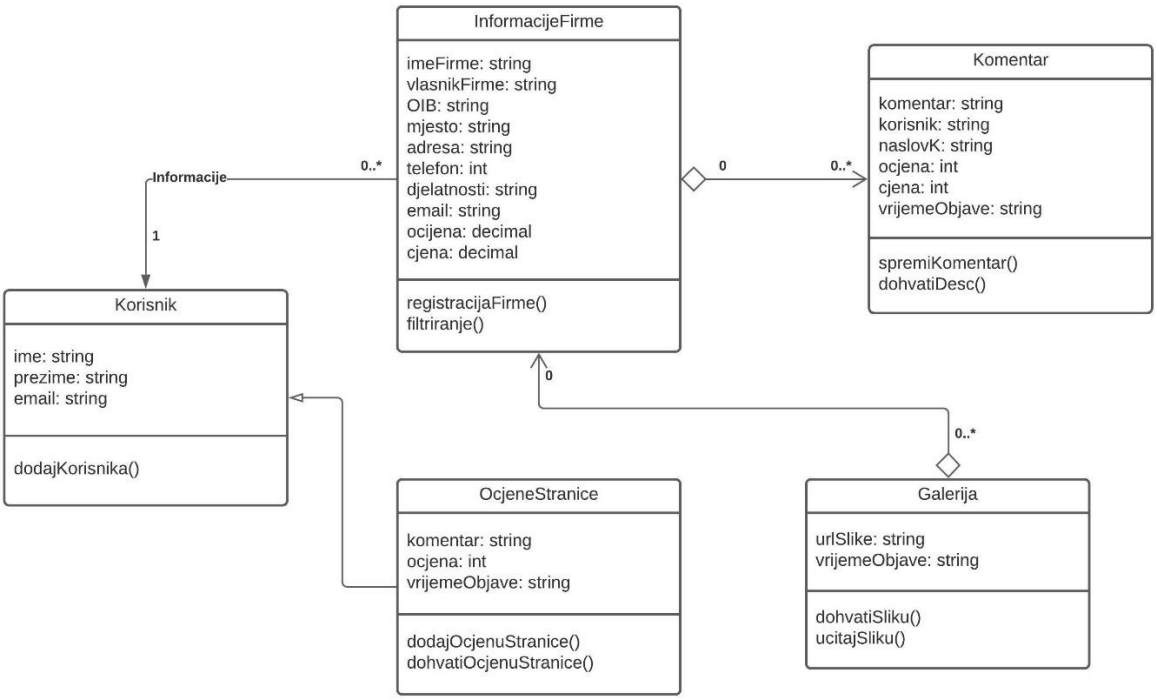
Slika 6. UML Use Case dijagram

3.2. Dijagram klasa

Dijagram klasa (eng. class diagram) je drugi dijagram koji je potrebno napraviti prije izrade aplikacije. Od sada će se u radu koristi izraz class diagram. On se primjenjuje za prikaze različitih namjena. Dijagramom možemo prikazati i strukturu objektno orijentiranog projekta [8]. Na ovoj aplikaciji će se primjenjivati kao domenski model web aplikacije. Dakle razmotrit će se koncepti i informacije bitne za ovu aplikaciju, a to su informacije koje kroz nju prolaze. Ovakav dijagram je jako dobra osnova za kasnije uređenje podataka u aplikaciji. Korišteni podaci u dijagramu će se sigurno spremati u bazu, bit će poslani u back-end i koristiti u front-end dijelovima aplikacije. Tako da je ovaj dijagram polazna točka koja nam prikazuje kako će se ti podaci povezati i strukturirati.

Prilikom izrade class dijagrama potrebno je obuhvatiti podatke koje je potrebno spremiti u aplikaciju. Naziv klase se upisuje u prvi pravokutnik, u drugi se upisuju atributi klase a u treći metode koje su podržane nad tom klasom. Slično kao i UML use case dijagramu imamo vezu koja se zove asocijacija. Asocijacija je poveznica između dvije klase koja ima kardinalnosti i označava se strelicom. Na kraju i na početku veze se nalaze kardinalnosti. Ako gledamo klasu sa lijeve strane, onda gledamo koliko ona

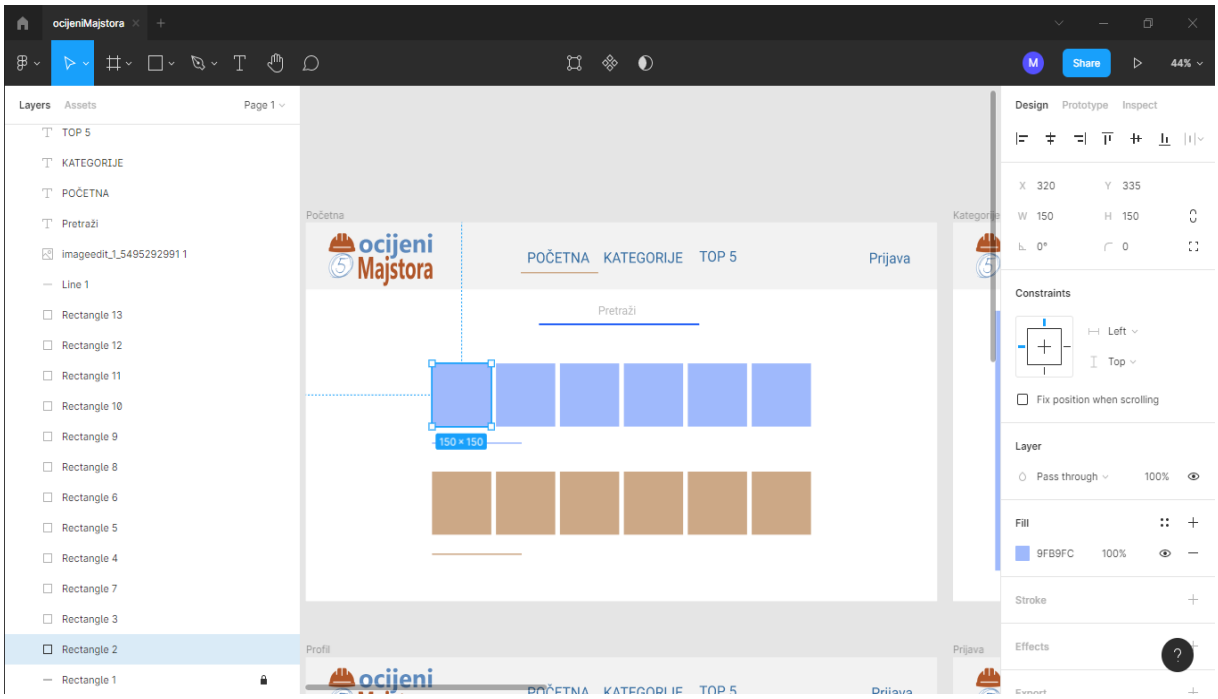
može imati instanci sa desne strane i obratno. Postoji i veza agregacija koja označava sadržaj između dvije instance klasa. Označava se sa bijelim rombom te ju se ne moramo imenovati kao vezu asocijacija. Sa svake strane sadrži kardinalnosti isto kao i veza asocijacija. Treća veza se zove kompozicija koja se označava sa crnim rombom. Sadrži jedno specijalno svojstvo a to je da obriše instanca klasa s lijeve strane automatski se briše instanca sa desne strane. U izradi dijagrama koristimo još jednu vezu koja se naziva nasljeđivanje. Specifično u vezi nasljeđivanje je da kada nema kardinalnosti između klasa. Nasljeđivanje koristimo kada postoje određene sličnosti između naših klasa. Class dijagram ima mnoga proširenja, jedno od proširenja nazivamo stereotipi. Najčešće korišten stereotip je enumeracija. Ona se navodi slično kao klasa, nema dijela za upis metoda i u gornjem djelu se navodi <<enumeration>> kao vrsta stereotipa. Enumeracija nam omogućava da definiramo jedan vlastiti tip, kao na primjer tip koji ima dozvoljene samo određene vrijednosti. Definiramo je sa strane napravljenog dijagrama i ne povezuje se jer se radi o proširenju. Slika 7 prikazuje UML class dijagram aplikacije ocjeni Majstora.



Slika 7. UML Class dijagram

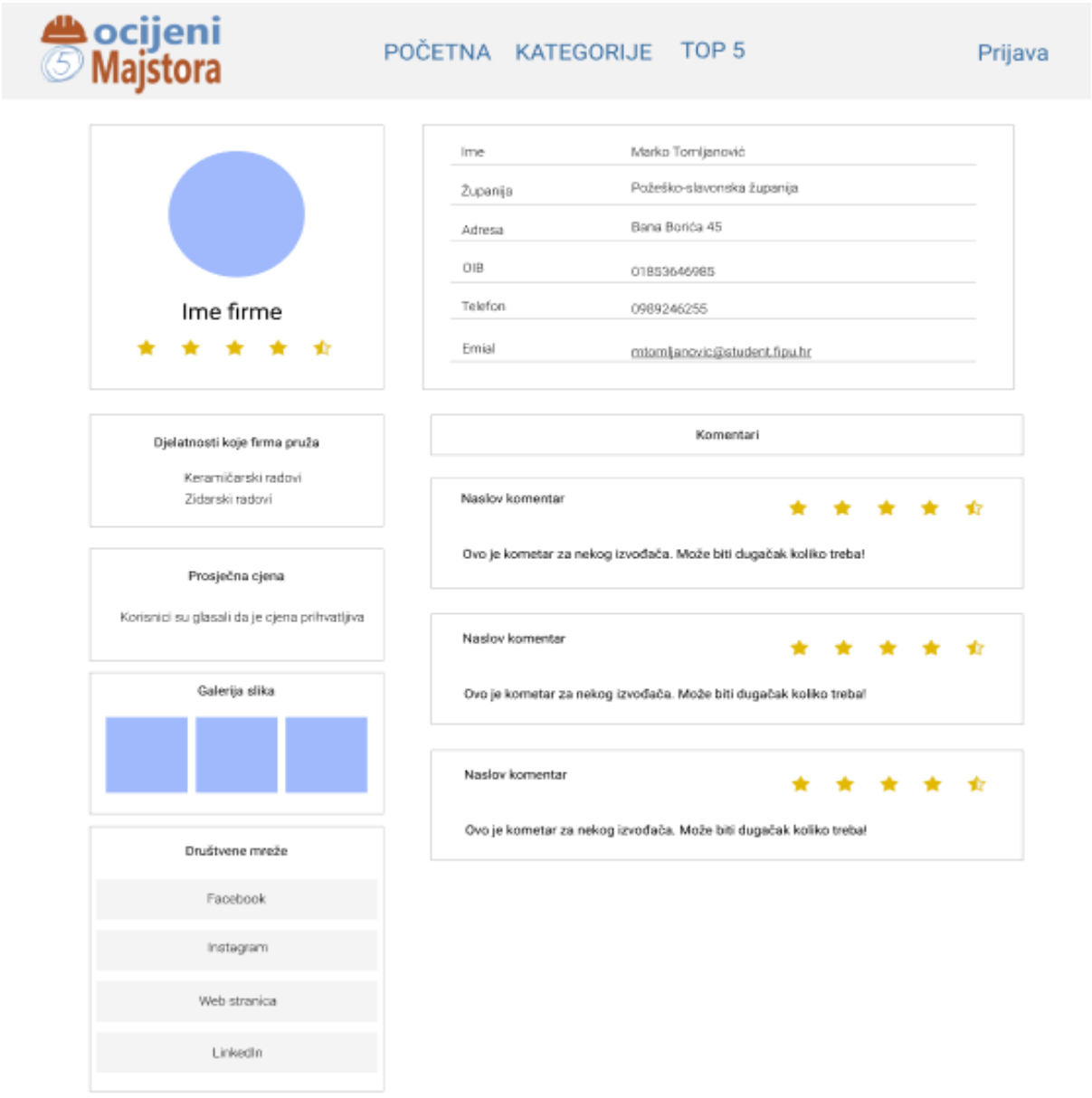
3.3. Prototip sučelja – Figma

Figma je program koji služi za izradu dizajna i prototipa za bilo koju vrstu digitalnog projekta. Temeljen je na cloud tehnologiji tako da korisnici Figme mogu međusobno surađivati na projektima. Postoji više sličnih rješenja na tržištu no ključan faktor je taj što Figma nudi mogućnost rada s timovima. Postoji opcija neograničenog broja sudionika koji imaju ulogu promatrača na projektu [9]. Figma je dovoljno dugo na tržištu te je stekla dobar ugled među programerima i dizajnerima. Napravljena je galerija stilskih elemenata kojoj se može pristupiti direktno iz programa. Trenutno je još u testnoj fazi ali ima široki spektar elemenata [9]. Program se koristi za izradu mobilnih aplikacija, web stranica te aplikacija za druge uređaje kao što su tableti, pametni televizori i ostali. Prije početka izrade web aplikacije u Figmi napravljen je prototip kako bi trebala izgledati aplikacija nakon završetka. Prilikom izrade aplikacije praćen je prototip i korištene su sve mogućnosti stilskog jezika uz manje korekcije izgleda same aplikacije. Slika 8 prikazuje sučelje programa Figma i jedan dio prototipa aplikacije ocjeni Majstora.



Slika 8. Sučelje programa Figma

Prije početka izrade aplikacije, prototip sučelja profila je izgledao kao na slijedećoj slici. Možemo primijetiti da je dizajn aplikacije vrlo sličan izgledu prototipa. Naknadno je uvedeno ocjenjivanje izvođača po osnovi cijene napravljene usluge te se prosječna cijena usluge svih ocjena prikazuje u kartici sa lijeve strane profila. Način ocjenjivanja cijene usluge sličan je ocjenjivanju firmi.



Slika 9. Prototip stranice profila

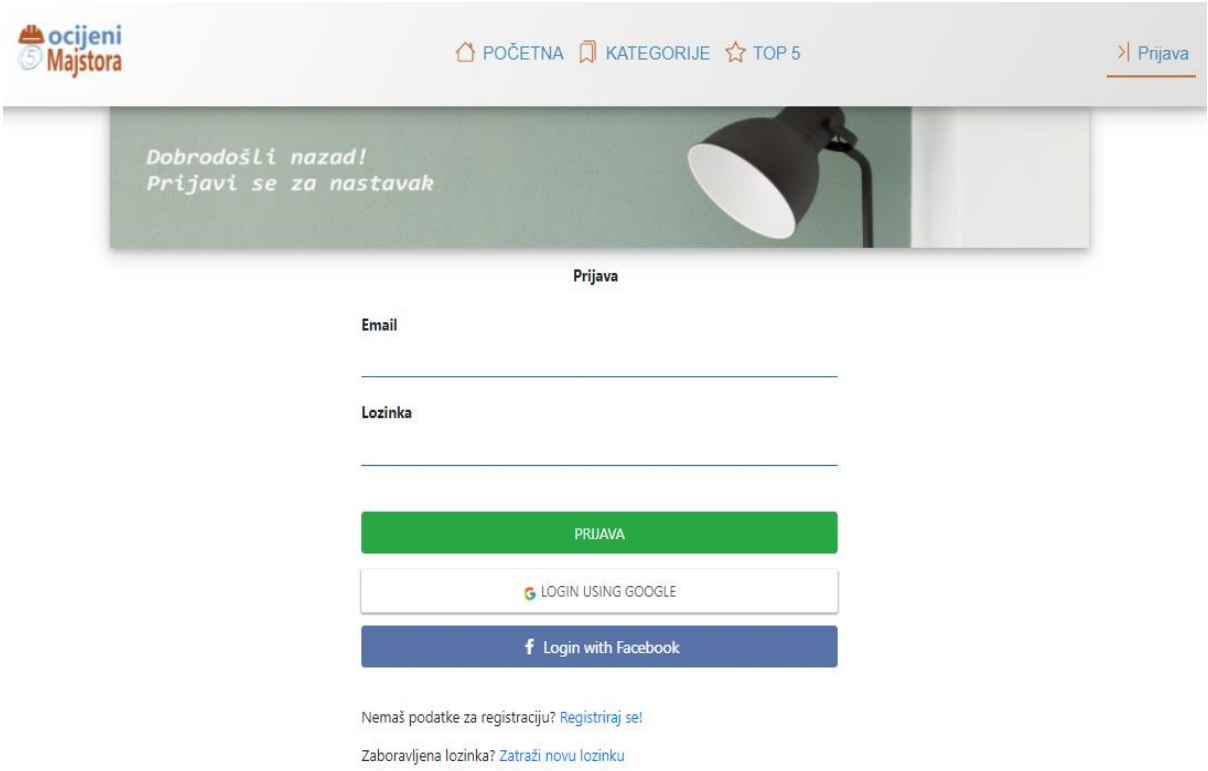
3.4. Netlify platforma

Netlify je platforma za web programere koja višestruko povećava produktivnost. Platforma pomaže web programerima da kreiraju, testiraju i objave (eng. deploy) stranicu. Od sada će se u radu koristi izraz deploy. Jedna je od najboljih platformi za hosting. Za razliku od ostalih platformi Netlify je pojednostavio proces razvoja web stranica. Postoji mogućnost povezivanja sa GitHub profilom te automatski vrši deploy stranice odabranog repozitorija [10]. Potrebno je samo napraviti commit koda na GitHub repozitorij te Netlify automatski vrši deploy stranice. Besplatna verzija ima ograničenja kao što je potrošnja prometa, no ograničenja su prihvatljiva za privatne korisnike. Kroz cijeli razvoj aplikacije ocjeni Majstora korištena je Netlify platforma za deploy stranice.

4. IMPLEMENTACIJA

4.1. Prijava i registracija

Prijava na aplikaciju je povezana direktno sa Firebase bazom podataka. U Firebase nadzornoj ploči odabiremo opciju „Authentification“ gdje se nalazi konfiguracija za autoriziranje te uvid u sve korisnike aplikacije. U postavkama se nalaze i načini prijave preko ostalih poslužitelja. Sve metode su već poznate i od svjetski poznatih poslužitelja. U aplikaciju se možemo prijaviti preko Google i Facebook računa. Slike 10 i 11 prikazuju sučelja stranica za prijavu i registraciju.



Slika 10. Sučelje stranice prijava

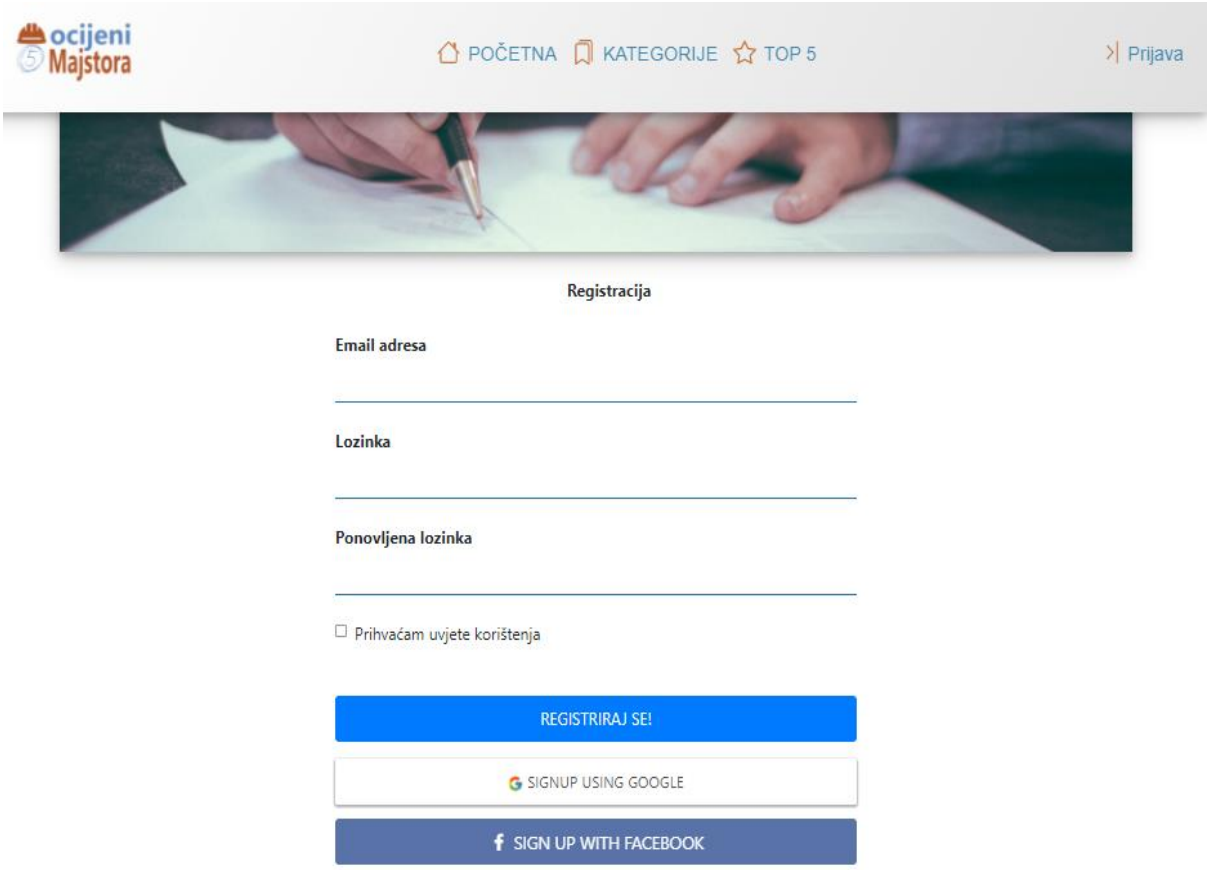
Na slici je vidljivo kako postoji mogućnost obnove lozinke, standardna funkcija koja je prisutna u cijelome svijetu. Koristi se ako je netko zaboravio svoju lozinku te ima pravo tražiti novu tako što upiše email kojim je registriran. Opcija je direktno povezana sa Firebase bazom tako što se zahtjev šalje preko Firestore funkcije „sendPasswordRestEmail“. Slika 11 prikazuje kod funkcije prijava(). Nakon uspješne

prijave aktivira se funkcija replace() koja otvara početnu stranicu. Prilikom neuspjele prijave, greška se upisuje u konzolu. Sličan kod je za registraciju korisnika gdje se koristi funkcija createUserWithEmailAndPassword().

```
methods: {
  prijava() {
    firebase
      .auth()
      .signInWithEmailAndPassword(this.korisnickoIme, this.lozinka)
      .then((result) => {
        console.log("Uspješna prijava", result);
        this.$router.replace({ name: "Home" });
      })
      .catch(function(e) {
        console.error("Greška", e);
        alert(e);
      });
  },
```

Slika 11. Funkcija prijava()

Slika 12 prikazuje sučelje registracije korisnika kojem se pristupa se preko stranice za prijavu. Sučelje je dizajnirano od slike na vrhu stranice, te sa tri inputa za upis korisničkih podataka. Ispod inputa su tri gumba koji služe registraciju i prijavu preko društvenih mreža. Gumbovi za prijavu preko Google i Facebook-a imaju istu ulogu kao u stranici za prijavu korisnika. Postavljeni su u obje stranice zbog boljeg korisničkog iskustva. Nakon uspješne registracije funkcija otvara početnu stranicu te korisnik dobiva pravo komentiranja i ostavljanja ocjena.



Slika 12. Sučelje stranice registracija

4.2. Baza podataka – Firebase

Firebase je izabran kao baza podataka ove aplikacije. Ima pregršt mogućnosti i opcija. Za povezivanje sa projektom potrebno je instalirati npm paket Firebase. Zatim inicijalizirati ga te kopirati konfiguraciju u kreirani firebase.js file. Firebase će zahtijevati poveznicu na domenu aplikacije zbog sigurnosti podataka. Na početku je potrebno postaviti „onAuthStateChanged()“ funkciju zbog definiranja trenutno prijavljenog korisnika. To je potrebno kako bi znali kada je korisnik prijavljen, a kada nije. Stanje prijavljenog korisnika definiramo tako što spremamo korisnik email u varijablu „trenutniKorisnik“ koja je dostupna kroz cijeli projekt u posebnom store.js dokumentu. Kada korisnik nije prijavljen u varijablu spremamo vrijednost null. Slika 13 prikazuje kod „onAuthStateChanged()“ funkcije koja se nalazi u App.vue.

```
firebase.auth().onAuthStateChanged(function(user) {
  if (user) {
    // User is signed in.
    console.log("**********", user.email);
    store.trenutniKorisnik = user.email;
  } else {
    console.log("No user");
    store.trenutniKorisnik = null;
  }
});
```

Slika 13. onAuthStateChanged() funkcija

Nakon uspješnog postavljanja, odabiru se potrebne usluge koja se moraju posebno konfigurirati i povezati sa aplikacijom. U nastavku će biti pojašnjeno na koji način su korištene usluge. Usluge korištene u aplikaciji su:

- Authentication
- Firestore
- Storage

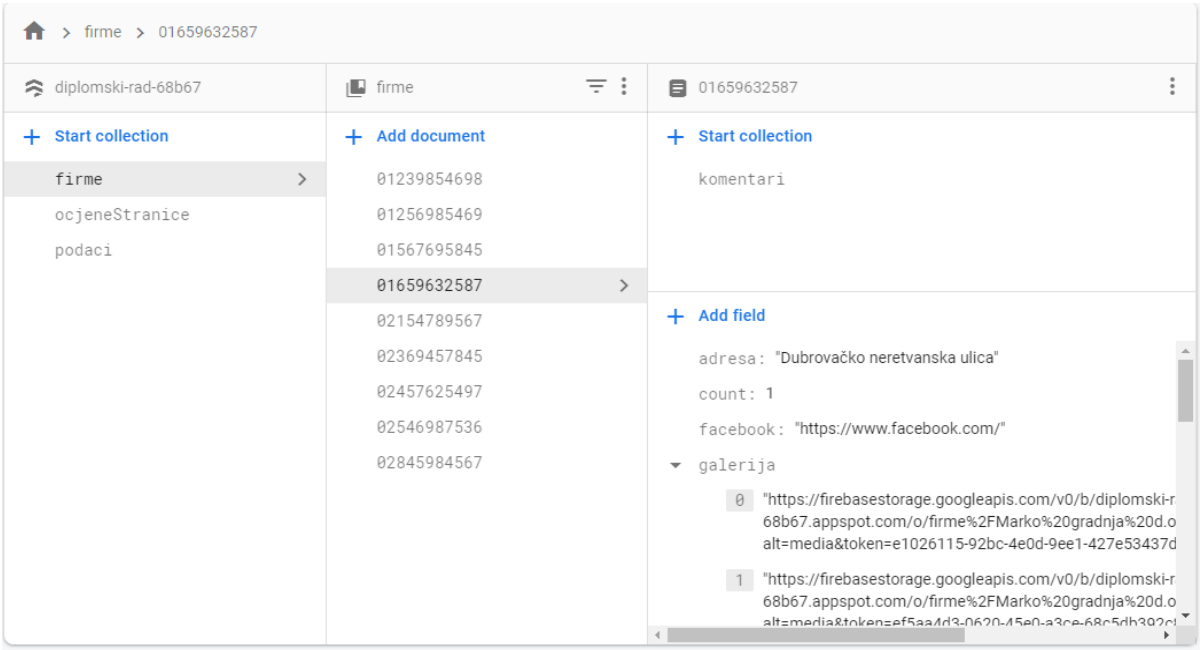
4.2.1. Authentication

Authentication usluga korištena je za prijavu i registraciju korisnika u aplikaciju. Prilikom uspostavljanja prijave i registracije u kodu, potrebno je koristiti Firebase dokumentaciju. Dokumentacija je prilično razgranata i opširna, no dobro je objašnjena. Ova usluga korištena je osim za prijavu i registraciju i za zaboravljenu lozinku korisnika. Email sa poveznicom za resetiranje lozinke je kreiran u nadzornoj ploči kao predložak. Za hosting korištena je već spomenuta Netlify platforma stoga je autorizirana domena ocijeniMajstora.netlify.app. Funkcije koje su korištene u aplikaciji:

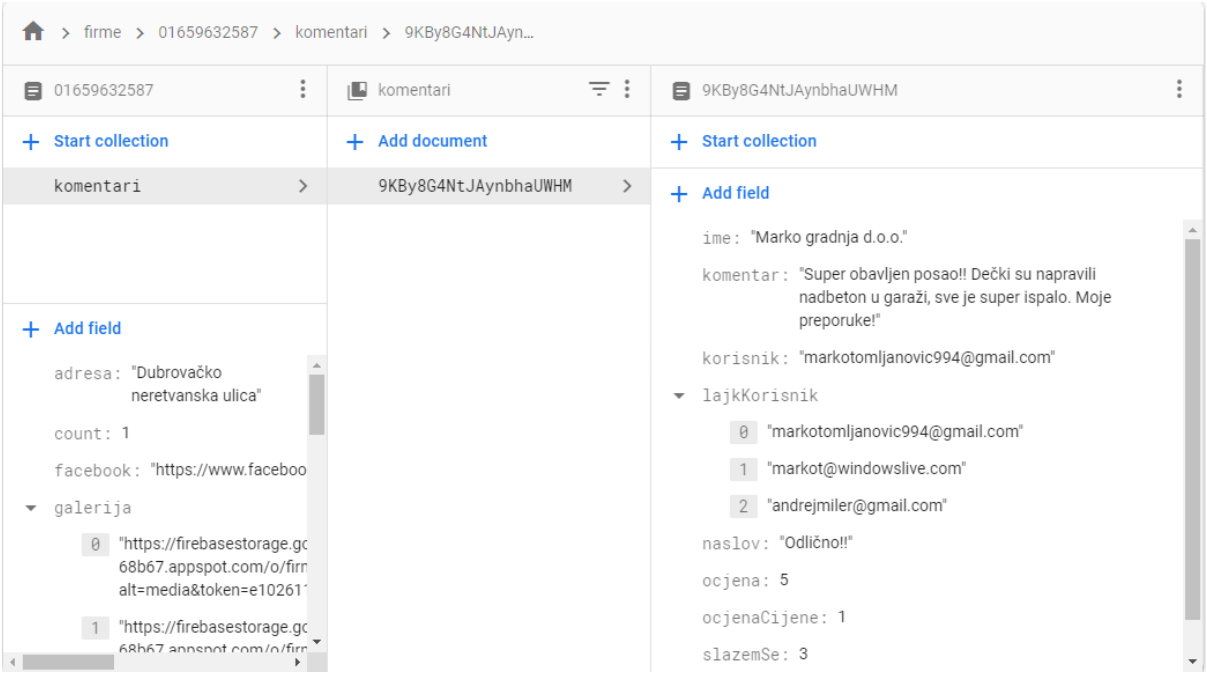
- auth.signinWithEmailAndPassword()
- auth.createUserWithEmailAndPassword()
- auth.signinwithemailandpassword
- auth.GoogleAuthProvider()
- auth.FacebookAuthProvider()
- auth.sendPasswprdResetEmail()

4.2.2. Firestore

Svi podaci spremaju se u Firestore bazu. Njegova baza ima kolekcije umjesto tablica tako da svaka kolekcija ima određen broj dokumenata u kojima se nalaze podaci. Za sigurnost pristupa i modifikacije nad podacima postoje pravila koja se pišu u Firestore nadzornoj ploči. Jedna od mogućnosti je da dopustimo ili zabranimo korisnicima čitanje, ažuriranje, brisanje i pisanje određene kolekcije, ovisno da li je korisnik prijavljen. Aplikacija ima dvije kolekcije u Firesotre bazi. Prva se zove „firme“ gdje su pohranjeni podaci koje je korisnik upisao prilikom registracije firme kao sto je ime firme, ime vlasnika, email, oib, i ostali podaci. Također, u kolekciji „firme“ je napravljena pod-kolekcija pod nazivom „komentari“. U pod-kolekciji su pohranjeni dokumenti u kojima se nalaze informacije svakog komentara. Dokumenti su automatski dobili Firebase ID koji nema veze sa pohranjenim podacima. Pod-kolekcijama se može pristupiti i preko indexa, odnosno atributa u kolekciji koja je povezana sa pod-kolekcijom. Tako su povezana dva podatka preko indexa ime i oib. U kolekciji je korištena funkcija incrise() koja povećava broj za određeni iznos. Korištena je kao brojač koji dodaje jedan za svaki dodani komentar u svrhu aritmetičke sredine upisanih ocjena. Druga kolekcija se zove „ocjeneStranice“ gdje su spremljeni podaci o ocjenama kojima su korisnici ocjenjivali aplikaciju i vrijeme objave. Slike 14 i 15 prikazuju strukturu podataka aplikacije.



Slika 14. Firestore kolekcija



Slika 15. Firestore pod-kolekcija

Slike 14 i 15 prikazuju kolekcije u kojima se nalaze pod-kolekcije. Dokumenti u kolekciji su organizirani po OIB podacima, tako da je svaki dokument unikatan i ne može doći do podudaranja sa drugim dokumentom. Podaci se mogu spremati i direktno u Firestore nadzornoj ploči, gdje se mogu testirati i izmjenjivati podaci. Postoji i Realtime baza koja je organizirana po tablicama no Firestore je novija te je zato korištena u aplikaciji. Struktura u Firestore bazi je dobar primjer za grananje komentara. Problemu grananja bi odlično odgovarale pod-kolekcije jer omogućavaju spremanje do sto pod-kolekcija u glavnu kolekciju. Grananje na forumima ili društvenim mrežama je moguće odlično definirati sa Firesotre bazom.

Slika 16 prikazuje kod za učitavanje podatka iz profil izvođača. Profil je dinamička stranica koja ovisi o izabranom izvođaču. Za dohvaćanje podataka su korištene funkcije koje su detaljno opisane u Firebase dokumentaciji. U varijablu db je pohranjen pristup Firebase-u, te zatim u „collection()“ upisujemo ime kolekcije. Kako bi našli traženu firmu koristimo funkciju „where()“ u koju upisujemo podatke kako bi pronašli traženi dokument. Nakon što su podaci spremljeni u varijablu, funkcija se referencira na pod-kolekciju za te iste podatke. Na taj način dohvaćamo komentare poredane po vremenu objave. Sortiranje je napravljeno pomoću funkcije „orederBy()“ koja može poredati ulazno i silazno bilo koje podatke iz baze. U bazu je spremljen

podatak „vrijemeObjave“ u obliku timestamp-a, što je zapravo broj koji predstavlja sekunde. Potrebno ga je pretvoriti u format datuma. To je učinjeno pomoću npm paketa koji nam daje mogućnost odabira između različitih formata. Pomoću istog paketa moguće je prikazati koliko je prošlo od učitavanja u bazu. Ta mogućnost se neće koristiti u projektu jer je pretpostavka da se jedna firma neće ocjenjivati svaki dan.

Podaci izvođača su spremljeni na Firestore bazu, u kolekciju „firme“. To su podaci koji će se najčešće koristiti u cijeloj aplikaciji te će se više puta dohvaćati iz baze. Svaki put će se koristiti drugačija funkcija za dohvaćanje podataka ovisno da li je potrebna pod-kolekcija sa podacima komentara.

Podaci koji su pohranjeni u za svaku firmu su:

- Ime firme
- Ime vlasnika
- OIB
- Županija
- Mjesto
- Adresa
- Kategorije djelatnosti
- Email
- poveznica profilne slike
- Putanja profila
- Profil kao jedna riječ (ako naziv ima više riječi)
- poveznice slika iz galerije
- Brojač komentara
- Ukupna ocjena
- Vrijeme objave
- Facebook, Web stranica, Instagram, LinkedIn, Twitter profili
- Pod-kolekcija za komentare

```
dohvatiFirme() {
  db.collection("firme")
    .where("ime", "==", this.id)
    .get()
    .then((query) => {
      query.forEach((doc) => {
        const data = doc.data();

        this.podaciProfila.push({
          ime: data.ime,
          vlasnikFirme: data.vlasnikFirme,
          zupanija: data.zupanija,
          mjesto: data.mjesto,
          adresa: data.adresa,
          sluzbeniEmail: data.sluzbeniEmail,
          telefon: data.telefon,
          oib: data.oib,
          pic: data.pic,
          profil: data.profil,
          galerija: data.galerija,
          facebook: data.facebook,
          webStranica: data.webStranica,
          youTube: data.youTube,
          instagram: data.instagram,
          twitter: data.twitter,
          avgOcjena: (data.ukOcjena / data.count || 0).toFixed(1),
          avgOcjenaCijene: (data.ukOcjenaCijene / data.count || 0).toFixed(
            2
          ),
          kategorije: data.kategorije,
        });
        doc.ref
          .collection("komentari")
          .orderBy("vrijemeObjave", "desc")
          .get()
          .then((query) => {
            query.forEach((doc) => {
              const dataK = doc.data();

              this.komentari.push({
                naslov: dataK.naslov,
                ocjena: dataK.ocjena,
                ocjenaCijene: dataK.ocjenaCijene,
                komentar: dataK.komentar,
                vrijemeObjave: moment(dataK.vrijemeObjave).format(
                  "DD-MM-YYYY"
                ),
              });
            });
            this.djelatnostiSlova();
          });
      });
    });
},
```

Slika 16. Kod dohvaćanja podataka iz Firesotre baze

4.2.3. Storage

Posljednja Firebase usluga koja je korištena u aplikaciji naziva se Storage. Ona služi za spremanje slika organiziranih po datotekama. U Storage-u su također primijenjena pravila sigurnosti kao i za Firestore bazu. Pravila su jednostavnija jer nema pod-kolekcija kao u Firestore-u te vrijede za sve putanje datoteka. U nadzornoj ploči Storage postaji i takozvani „Rules Playground“ gdje se može simulirati dohvaćanje, kreiranje, izmjena i brisanje trenutno definiranih putanja. U slijedećem kodu vidi se kako pravila sigurnosti vrijedi za sve putanje, te im omogućava čitanje i pisanje neovisno da li je korisnik prijavljen.

```
rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

Aplikacija Storage služi za spremanje slika profila izvođača, a isto tako i naslovnih slika aplikacije. Galerija slika izvođača je napravljana od datoteka u kojima se nalaze izvođači koji su postaviti slike. Svaka firma ima svoju putanju ovisno o nazivu firme. Neposredno nakon učitavanja, slike se dohvaćaju putem te iste putanje te poveznice slike koja se sprema u potrebnu kolekciju Firestore baze. Kada je potrebno dohvatiti poveznicu slike, ona se uvijek nalazi među podacima firme u istoj kolekciji. Stoga zaključujemo da Storage služi za spremanje slika, no kada je potreba za učitavanjem, to se događa putem poveznice koja je spremljena u potrebnu Firebase kolekciju.

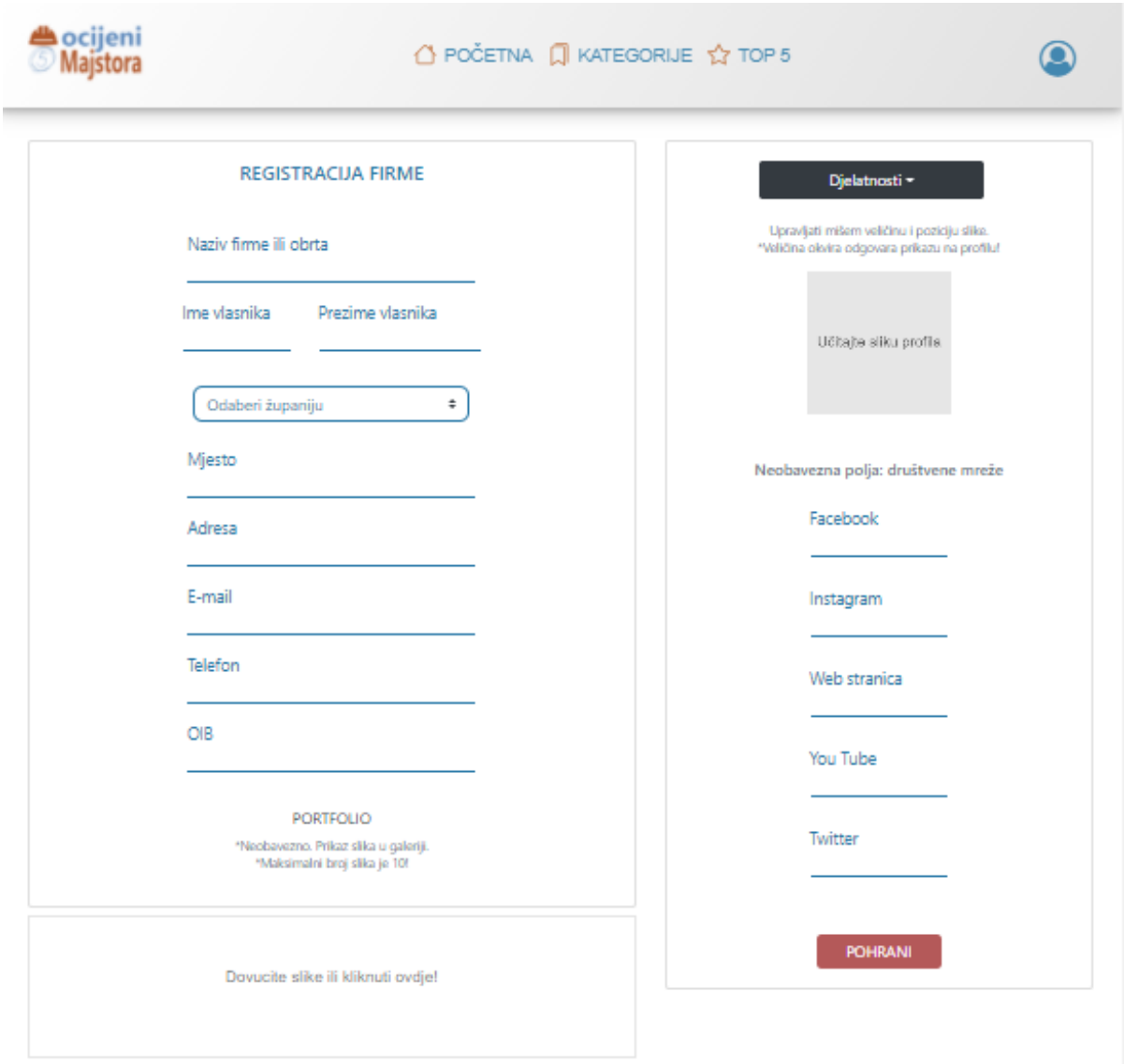
Slika 17. Spremanje slika u Storage

Slika 17 prikazuje funkciju koja sprema sliku profila u Storage. U slijedećem koraku će se poveznica spremiti u Firestore kolekciju gdje se spremaju svi podaci te iste firme. Putanja na spremljenu sliku glasi firme/imeFirme, dok ime slike ima naziv profilnaSlika_timestamp. Timestamp je broj koji ovisi o trenutnom datumu i vremenu postavljanja slike. Način na koji se sprema slika onemogućava da novo učitana slika ima jednaki naziv kao jedna od učitanih u bazi.

4.3. Registracija firme

Registracija firme je nužna za kreiranje profila iste. Registracija je moguća samo za prijavljene korisnike, a stranici se pristupa tako da u padajućem izborniku profila odabiremo „Nova firma“. Ako neprijavljeni korisnik pokuša pristupiti stranici tako da u preglednik upiše putanju za registraciju, ruter (eng. ruther) automatski preusmjerava korisnika na stranicu prijave. To je omogućeno pomoću funkcije ruther.beforeEach(to, from, next) koja ukazuje na putanju sa koje dolazi, kao trenutnu

i slijedeću. U <script> dijelu stranice korištena je već spomenuta funkcija za učitavanje podataka u Firestore i Storage. Korištene su i funkcije za provjeru upisa u polja za podatke firme. Jedna od funkcija je razdvojiTelefon() koja dodaje znak "/" nakon prve tri znamenke telefonskog broja. Funkcija oibProvjera() sprječava da u input za upis OIB-a ne bude više od 11 brojeva. Prilikom izrade dizajna stranice praćena je struktura stranice profila izvođača, stranica je podijeljena na lijevu i desnu stranu. Za unos podataka obavezne se sve bitne kontakt informacije kao što su ime i prezime, naziv firme, OIB firme, županija, grad, adresa, email i telefon. Ostali podaci nisu obavezni, služe samo kao pomoć pri izlaganju firme korisnicima koji čitaju ili komentiraju.



Slika 18. Sučelje registracije firme

```
ucitaj(event) {
  event.preventDefault();
  if (this.podaci.selected == null) {
    this.$swal.fire({
      icon: "warning",
      title: "Odaberi županiju",
    });
  } else if (this.podaci.selectedKategorija.length == 0) {
    this.$swal.fire({
      icon: "warning",
      text: "Obavezno odabrati minimalno jednu kategoriju!",
    });
  } else {
    this.imgRef.generateBlob((blobData) => {
      let imgName =
        "firme" +
        "/" +
        this.imeFirme +
        "/" +
        "profilnaSlika_" +
        Date.now() +
        ".png";

      storage
        .ref(imgName)
        .put(blobData)
        .then((result) => {
          //uspjesno spremanje
          result.ref
            .getDownloadURL()
            .then((url) => {
              db.collection("firme")
                .doc(this.oib)
                .set({
                  ime: this.imeFirme,
                  vlasnikFirme:
                    this.vlasnikFirmeIme + " " + this.vlasnikFirmePrezime,
                  zupanija: this.podaci.selected,
                  mjesto: this.mjesto,
                  adresa: this.adresa,
                  sluzbeniEmail: this.sluzbeniEmail,
                  telefon: this.telefon,
                  kategorije: this.podaci.selectedKategorija,
                  oib: this.oib,
                  userEmail: store.trenutniKorisnik,
                  profil: "/Profil/" + this.imeFirme, // .replace(" ", "", "g"),
                  pic: url,
                  galerija: this.images,
                  facebook: this.facebook,
                  instagram: this.instagram,
                  youTube: this.youTube,
                  twitter: this.twitter,
                  webStranica: this.webStranica,
                  count: 0,
                  ukOcjena: 0,
                  vrijemeObjave: Date.now(),
                })
                .then((doc) => {
                  console.log("Spremljeno", doc);
```

Slika 19. Funkcija ucitaj() u registracija.vue

Slika 18 prikazuje sučelje stranice za registraciju korisnika. Korisnik će biti obavješten ukoliko ne ispuni potrebne informacije u obavezna polja. Profilna slika nije obavezna, no ako korisnik nema potrebu dodati sliku, sustav će profilu dodijeliti univerzalnu sliku. Za dodavanje profilne slike korišten je paket „croppa“ koji omogućuje uređivanje veličine i rezanje slike na određenu dimenziju. Isto tako, za dodavanje slika u galeriju korišten je paket „Vue2-Dropzone“ koji omogućava drag&drop i dodavanje više slika, za razliku od paketa „croppa“. Više o paketima biti će riječ u poglavlju pod nazivom npm paketi. Slika 19 prikazuje funkciju „ucitaj()“ koja dohvaća podatke iz Firestore baze.

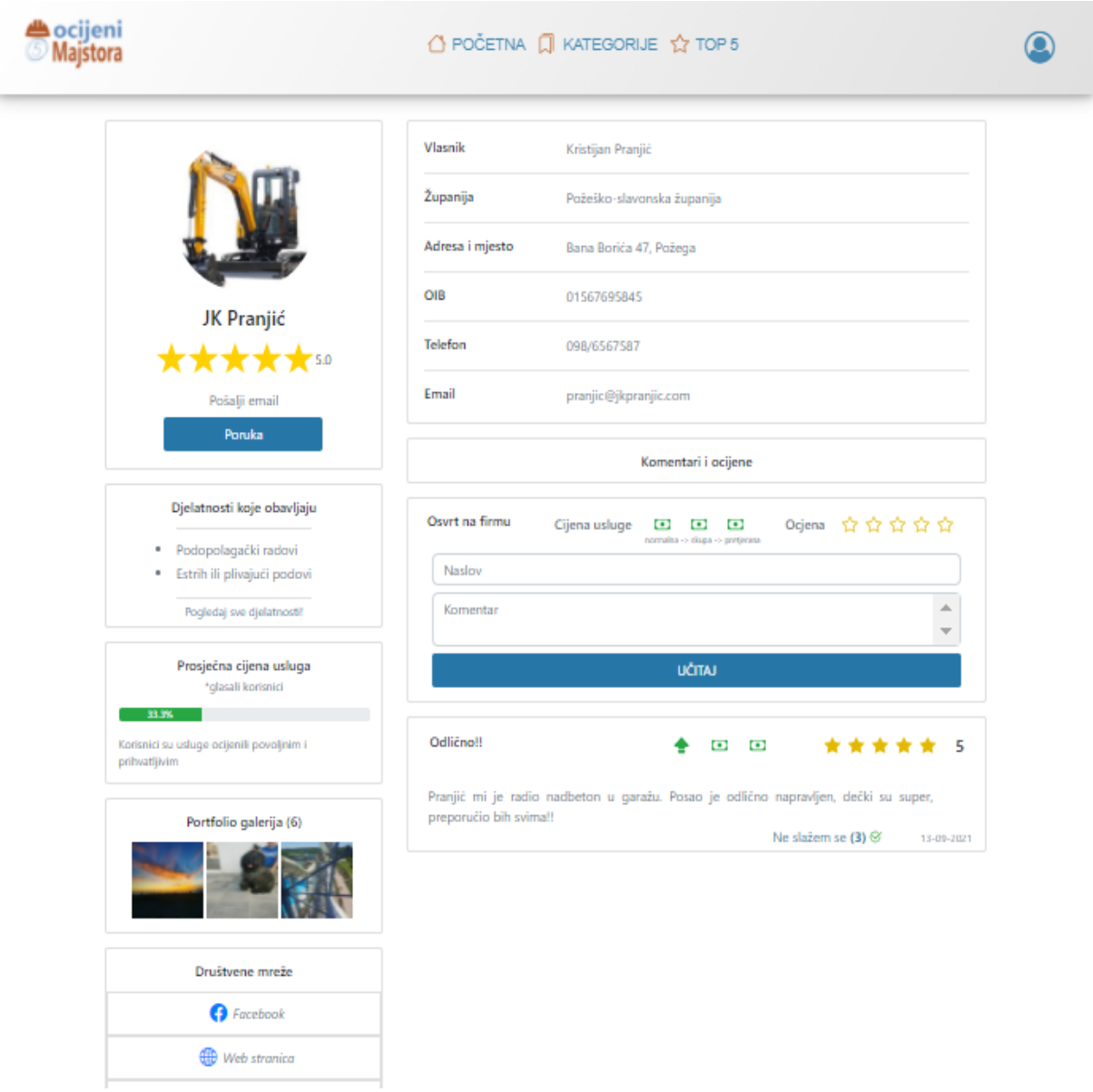
4.4. Profil i njegove funkcionalnosti

Za potrebe prikazivanja osobnih podataka izvođača radova te komentiranje istih napravljena je stranica profil. Stranica objedinjuje sve podatke koje je vlasnik firme uneo prilikom registracije firme te komentare i ocjene. Svaki neautorizirani korisnik ima pravo uvida u sve profile izvođača. Stranica je dinamička, što znači da broj stranica odgovara broju izvođača. Kada korisnik unese putanju koja nije dodijeljena niti jednoj stranici ruter (eng. ruther) prosljeđuje na komponentu 404 error, o kojem će biti više riječi u slijedećim poglavljima.

Forma za ostavljanje osvrta je jednostavno dizajnirana te se sastoji od dva polja za unos teksta, dvije forme za ocjenjivanje i gumba (eng. button) za učitavanje. U prvi dio polja za ocjenjivanje se upisuje naslov komentara, a u drugi komentar. Komentari su poredani silazno prema datumu objave. Korisnici koji nisu registrirani nemaju pravo komentirati niti ostavljati ocjene. To je omogućeno v-if direktivom, koja je slična if grananju. Pomoću direktive v-if skrivamo komponente kada korisnik nije registriran. U Vue.js direktive upisujemo HTML kod koje omogućavaju određene funkcionalnosti. Profil sadrži još tri bitne funkcionalnosti, a to su:

- Ocjenjivanje
- Prosječna cijena
- Galerija slika
- Oznaka „Slažem se“

Svaka funkcionalnost će biti obrađena u narednim podglavljima. Postoji mogućnost da se naknadno uvede nova funkcionalnost, ovisno o zahtjevima stvarnih korisnika aplikacije. Stranica profila nikada neće biti kompletno završena, uvijek će se nadograđivati i ažurirati sa svježim izgledom sučelja te novim funkcionalnostima koje zahtijevaju korisnici.



Slika 20. Sučelje profila

4.4.1. Ocjenjivanje

Glavni dio aplikacije je ocjenjivanje i prikaz prosječne ocjene svakog izvođača. Stoga je prosječna ocjena prikazana na svim mjestima gdje su prikazani izvođači. Ocjenjivanje kvalitete izvođača je potrebno radi informiranja budućih potencijalnih klijenata izvođača. Način na koji korisnik ocjenjuje je takav da u profilu izvođača u formi za osvrt upiše cijelu recenziju. Nakon toga podaci se unose u bazu te se automatski izračunava prosječna ocjena koja se prikazuje na profilu. Ocjenu koju korisnik može dati su cijeli brojevi od jedan do pet, dok je prosječna ocjena koja se prikazuje zaokružena na jednu decimalu. Ocjena komentara kao i prosječna ocjena prikazuje se u obliku žutih zvjezdica. Za takav prikaz korišten je BootstrapVue element koji ima dinamički način prikaza zvjezdica.

```
<b-form-rating
  style="margin-top:-4px"
  class="w-25 h-25 mr-2 float-right"
  id="rating-5"
  v-model="ocjenaKorisnika"
  stars="5"
  color="#e2b900"
  no-border
  size="lg"
></b-form-rating>
<p class="float-right mt-1 text-secondary">
  Ocijena
</p>
<b-form-rating
  icon-empty="cash"
  icon-full="capslock-fill"
  style="margin-top:-4px;margin-right:10px"
  class="w-25 h-25 mr-4 float-right"
  id="rating-5"
  v-model="ocjenaCijene"
  stars="3"
  variant="success"
  no-border
  size="lg"
></b-form-rating>
```

Slika 21. Kod za ocjenjivanje

Slika 21 prikazuje HTML kod za definiranje zvjezdica pomoću komponente <b-form-rating> i njegovih atributa koji služe sa definiranje ocjenjivanja. U kodu se može iščitati atribut „stars“ koji ograničava ocjenjivanje na 5 zvjezdica. Korištena je i v-model direktiva koja nam upisani input pohranjuje u varijablu te se ona sprema u bazu aplikacije. U ovom slučaju učitava se ocjena korisnika koju unosi preko broja zvjezdica od jedan do pet.

4.4.2. Prosječna cijena usluge

Svaki osvrt sadrži ocjenu cijene izvršene usluge izvođača. Aritmetička sredina cijena usluga pokazuje nam da li je izvođač povoljan ili nije. Pohranjivanje u bazu vrši se kao kod ocjenjivanja izvođača. Aritmetička sredina prikazuje se sa lijeve strane profila u obliku trake progress bar-a i kratkog opisa. Definirana su tri kratka opisa koji će biti prikazani uz postotak prosječne cijene. To je omogućeno pomoću već spomenute v-if direktive u kojoj su definirana ograničenja prosječne cijene. Prilikom davanja ocjene korisnik može birati između tri stupnja ocjene, to su:

- Niske cijene usluge
- Povoljne cijene usluge
- Visoke cijene usluge

Stupnjevi ocjenjivanja prikazani su u novčanicama koje svojim izgledom asociraju na cijenu. Niska cijena se prikazuje u intervalu između 0 i 1.6%. Zatim povoljna cijena usluge u intervalu od 1,6% do 2.6%, te će ta oznaka predstavljati srednju cijenu koja je prihvatljiva. Zadnji stupanj cijene je visoka cijena koja se nalazi u intervalu od 2.6% do 3% i predstavlja skupog izvođača koji se ne preporuča potencijalnim klijentima. Slika 19 prikazuje HTML kod koji prikazuje slično kao kod zvjezdica, način definiranja ocjenjivanja u obliku novčanice. Na lijevoj strani profila je prikazan progress bar te pokazuje postotak ocjene zaokružen na jednu decimalu. Kada na profilu nema niti jednog osvrta, tada ne postoji prosječna cijena usluge i u elementu za prikaz je ispisna poruka da nema cijena.

4.4.3. Galerija slika

Slike su uvijek najvjerniji prikaz stvarne situacije. Stoga je uvedena opcija za dodavanje slika koje bi se prikazivale na profilu izvođača. Slike služe kako bi se izvođači mogli predstaviti u najboljem izdanju. Galerija je portfolio završenih usluga te je moguće dodati maksimalno deset slika. Slike se dodaju prilikom registracije firme te je samo pohranjivanje jednostavno. Sliku je moguće mišem dovući u prozor za učitavanje ili jednostavno kliknuti na prozor kako bi se otvorilo pretraživanje slika lokalno spremljenih na računalu. Slike su pohranjene u Storage bazu koja je jedan dio Firebase usluga. Nakon učitavanja u bazu, dohvaća se poveznica na spremljenu sliku i sprema se u kolekciju zajedno sa ostalim podacima registrirane firme. Na stranici profila, galerija je smještana sa lijeve strane ispod prosječne cijene usluga. Prikazane su prve tri slike koje je moguće uvećati klikom na jednu od slika. Slika 22 prikazuje HTML kod za prikaz galerije na stranici.

```
<div v-if="podaciProfila[0].galerija" class="card mt-3">
  <h6 class="mb-3 mt-3 mx-auto">
    Portfolio galerija ({{ podaciProfila[0].galerija.length }})
  </h6>
  <div class="mx-auto mb-3" style="display: flex;">
    <b-link>
      <img
        class="mr-1"
        style="width:90px; height: 90px"
        v-for="(image, i) in podaciProfila[0].galerija.slice(0, 3)"
        :src="image"
        :key="i"
        @click="index = i"
      /></b-link>
  </div>
  <vue-gallery-slideshow
    :images="podaciProfila[0].galerija"
    :index="index"
    @close="index = null"
  ></vue-gallery-slideshow>
</div>
```

Slika 22. HTML kod za galeriju slika

4.4.4. Brisanje komentara

Nakon učitavanja komentara, postoji mogućnost da je korisniku ne sviđa ili je napravio pogrešku te ga želi obrisati. Zbog toga je kreirana mogućnost brisanja. Komentar se uklanja tako što se u profilu firme, ispod komentara nalazi ikona u obliku koša za smeće. Ikonu mogu vidjeti samo korisnici koji su učitali taj komentar. Nakon brisanja komentara iz kolekcije se smanjuje broj ukupnih komentara kao i ocjena. Da bi se umanjile ocjene koristila se firestore funkcija „.increment()“ koja može imati pozitivnu i negativnu vrijednost. Nakon brisanja te kada se stranica osvježi, ukupna ocjena profila i prosječne cijene poprimaju novu vrijednost umanjenu za obrisani komentar.

4.4.5. Opcija – slažem se

Korisnik koji se slaže sa jednim od učitanih komentara može označiti komentar sa oznakom „Slažem se“. Takva opcija je danas sveprisutna na internetu jer je odlično rješenje za dobivanje povratne informacije od ostalih korisnika.

Opcija se nalazi na dnu komentara te se uz nju nalazi broj koji prikazuje koliko korisnika se slažu sa komentarom. Nakon što se opcija odabere oznaka se promjeri u „Ne slažem se“ jer bi prilikom drugog odabira opcije poništili prvi odabir. Korisnici koji nisu prijavljeni u aplikaciju mogu vidjeti broj oznaka „Slažem se“ ali prilikom odabira opcije korisnik se obavještava kako se potrebno prijaviti za tu mogućnost. Slika 444 prikazuje dio funkcije koja broji oznake „Slažem se“.

```
} else if (!this.lajkKorisnik.includes(this.store.trenutniKorisnik)) {
  db.collection("firme")
    .doc(oib)
    .collection("komentari")
    .doc(idd)
    .update({
      slazemSe: zbrojii.increment(1),
      lajkKorisnik: zbrojii.arrayUnion(this.store.trenutniKorisnik),
    })
    .then(() => {
      console.log("Lajkano!");
      this.$router.go();
    })
    .catch(function(error) {
      console.error(error);
    });
```

Slika 23. Dio funkcije za opciju "Slažem se"

4.5. Dinamičke rute

Kako bi kreirali stanice profila posebno za svakog izvođača, bilo je potrebno napraviti dinamičke rute. U Vue.js okviru postoji objekt „route“ kojemu se može pristupiti iz bilo kojeg dijela koda. Route sadrži sve informacije vezane za trenutnu rutu te mu se može pristupiti preko određenih svojstva, a to su:

- $route.path
- $route.params
- $route.query
- $route.fullPath
- $route.matched
- $route.name
- $route.redirectedForm

Najčešće korišteno svojstva su $route.path i &route.params. $route.path koristi se kada je potrebno pristupiti trenutnoj ruti, dok je &route.params objekt koji sadrži parametre, ako ih nema bit će prazan objekt. $route.query je objekt koji se koristi za vraćanje podataka. $route.fullPath je cijeli URL koji uključuje querry i hash. $route.matched je niz koji sadržava podatke o putanji trenutne rute. $route.name prikazuje ime trenutne rute ako je definirano i ako postoji redirekcija i $route.redirectedForm prikazuje ime rute sa koje je redirekcija izvršena.

Za podešavanje dinamičke rute potrebno je u indeks.js definirati rutu te na kraju putanje u nastavku upisati „/:id“. To je dovoljno kako bi ruther osigurao stranicu za bilo koji naziv putanje. Slika 23. prikazuje definiranje putanje stranice profila u indeks.js. Kako bi se osiguralo da se prikažu izvođačevi podaci na nazivu putanje od istog izvođača, potrebno je iskoristiti već spomenuto svojstvo &route.params. U ovom slučaju koristiti će se &route.params.id da se pristupi trenutnoj putanji. Da bi se pristupilo trenutnoj putanji izvođača, u karticu izvođača potrebno je upisano njegovo ime. Stoga kada se dohvaćaju podaci iz baze, prvo se uspoređuje ime rute preko &route.params.id sa imenima izvođača u bazi. Kada dođe do podudaranja dohvaćaju se podaci te se popuni profil. Na taj način je napravljana dinamička ruta za stranicu

korisnikovog profila koja je različita za svakog izvođača. Zbog toga su potrebne dinamičke rute koje daju neograničeni broj mogućnosti.

```
{
  path: '/profil/:id',
  name: 'Profil',
  component: () => import('../views/Profil.vue')
},
```

Slika 24. Dinamička ruta profila

Do problema dolazi kada ne postoji podatak sa kojim bi se usporedila upisana putanja. To se može dogoditi kada korisnik upiše u nastavku dinamičke putanje bilo koji string, tada se otvara prazna stranica sa elementima iz App.vue dokumenta. Kako se to ne bi dogodilo potrebno je definirati i dizajnirati 404 error stranicu.

4.5.1 404 error

404 error stranica služi za obavještavanje korisnika da je došlo do pogreške, to jest da stranica koju pokušavamo učitati ne postoji. U aplikaciji je stranica definirana sa komponentom koja se učitava prilikom pokretanja nepostojeće putanje. Postoje određena pravila koja služe za pokretanje komponente ako dođe do putanje koja ne postoji. Slika 24 prikazuje kod u datoteci indeks.js gdje je sa „/pathMatch(.*)*“ definirano da se otvara komponenta „stranicNijePronadena.vue“.

```
{
  path: '/:pathMatch(.*)*',
  name: 'stranicaNijePronadena',
  component: () => import('../components/stranicaNijePronadena.vue')
}
```

Slika 25. Dinamička ruta 404 error-a

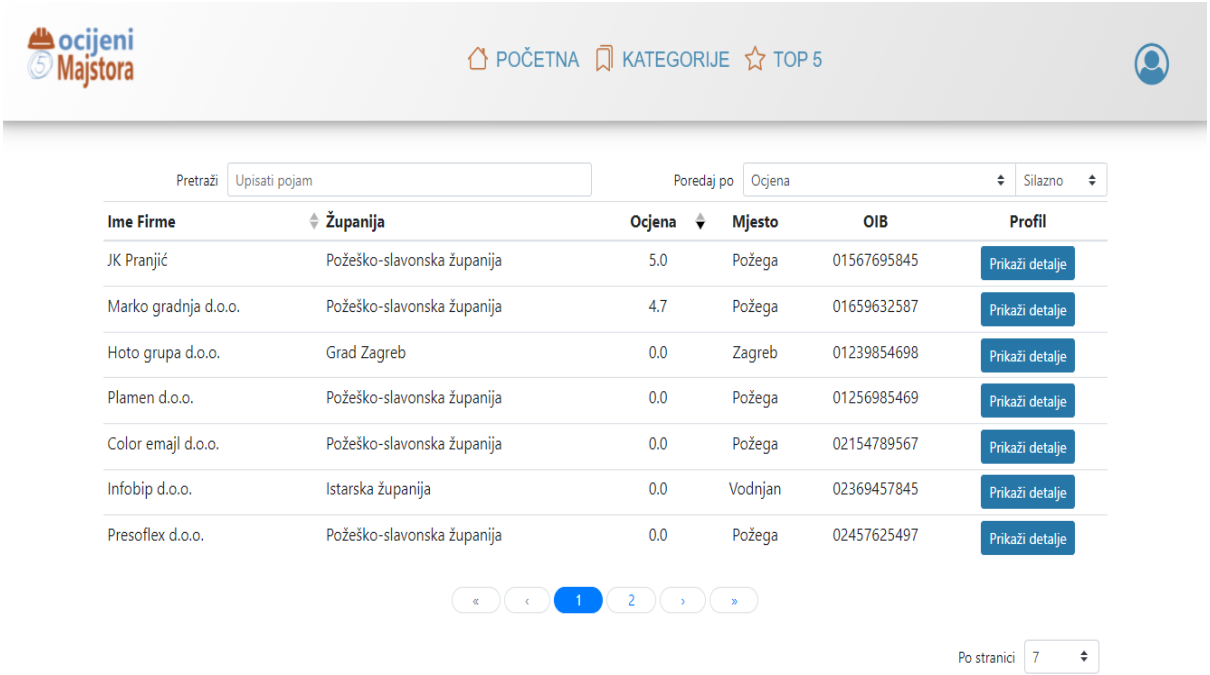
4.6. Filtriranje izvođača

Filtriranje izvođača je bitno jer je osnova za ovu aplikaciju ocjenjivanje izvođača. Stoga je od iznimne važnosti za korisnika da ima mogućnost pronaći traženog izvođača po bilo kojoj osnovi. Filtriranje je omogućeno na više stranica u aplikaciji. Početna stranica je najfrekventnija i zato su na njoj smješteni prozori u kojima se nalaze vrste građevinskih radova i svi izvođači koji se mogu pretraživati. Pretraživanje po imenu je interaktivno te prilikom svakog unosa dolazi do promjena u rezultatima pretrage. Slika 25 prikazuje funkciju za pretraživanje. U kodu je korištena ugrađena JavaScript funkcija includes() koja vraća true ako postoji dio upisane riječi u riječi koju tražimo.

```
filterRadovi() {
  let termin = this.store.traziPocetna;
  return this.Radovi.cards.filter((card) =>
    card.text.toLowerCase().includes(termin.toLowerCase())
  );
},
filterIzvodaci() {
  return this.karticaI.filter((izv) =>
    izv.ime.toLowerCase().includes(this.store.traziPocetna.toLowerCase())
  );
},
```

Slika 26. Funkcija za pretraživanje

U stranici za prikaz izvođača nalazi se tablica u kojoj su smješteni svi izvođači. To je stanica na kojoj je moguće filtrirati izvođače. Stranici se pristupa iz izbornika „kategorije“. Tablica omogućava filtriranje po ocjenama i abecedi. Na stranici se nalazi polje za upis traženog podataka kao i opcija za sortiranje. Tablica također ima direktan uvid u osnovne podatke tražene firme te je preko nje moguće direktno pristupiti profilu firme. Sama tablica je element iz BootstrapVue okvira koja ima mnogo opcija za filtriranje. Slika 26 prikazuje sučelje stranice izvođači, gdje se može vidjeti spomenuta tablica.



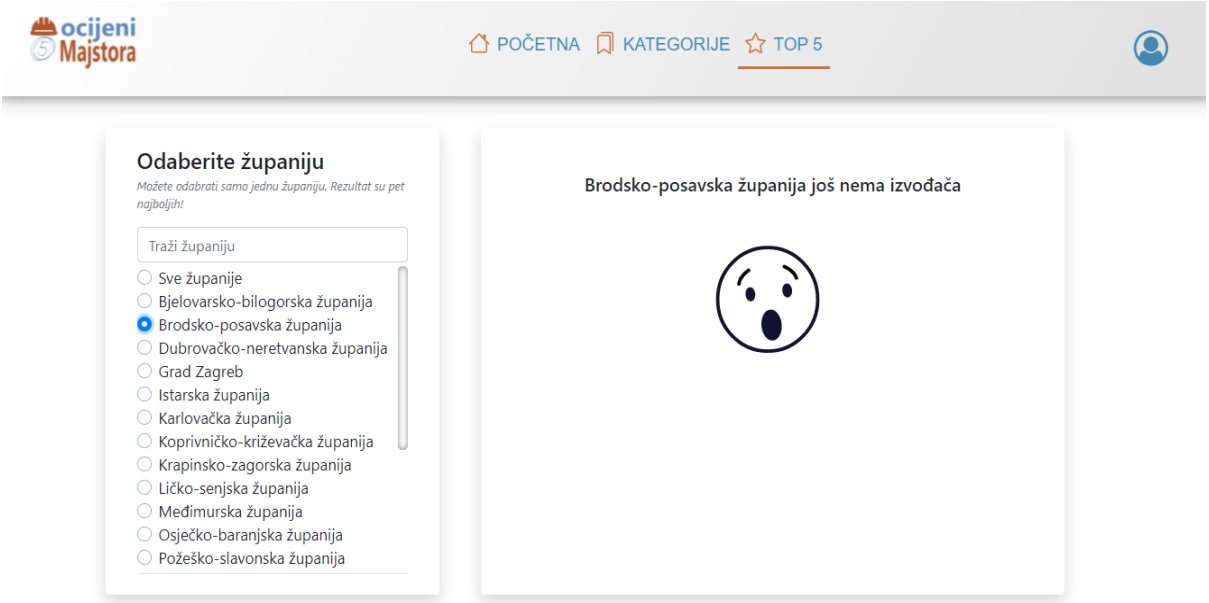
Slika 27. Sučelje stranice Izvođači

Sljedeća stranica na kojoj je moguće filtrirati izvođače je „top5“ stranica. Pristupa joj se direktno iz navigacijske trake. Ona ima mogućnost filtriranja izvođača po županijama. Sa lijeve strane su radijski gumbi (eng.radio buttons) koji označavaju sve županije u Hrvatskoj te kategorije gdje su svi izvođači. Odabirom jedne od županije prikazuje se pet najboljih izvođača u toj županiji. Uz pet najboljih se koristi još jedno filtriranje po ocjenama, zatim sortiranje od najveće ocjene prema najmanjoj. Za prikaz samo pet izvođača koristi se ugrađena JavaScript funkcija „slice()“ koja po potrebama programera može izrezati polje podataka. Slika 27 prikazuje funkciju za sortiranje izvođača prema prosječnim ocjenama. U funkciji je korištena ugrađena JavaScript funkcija „sort()“ koja sortira na principu prvog slova u riječi ili brojevima.

```
sortiraj() {
  this.filterZupanijeIzvodaci.sort((a, b) => {
    return b.avgOcjena - a.avgOcjena;
  });
},
```

Slika 28. Funkcija sortiranja prema ocjenama

Na kartici izvođača prikazano je njegovo ime te prosječna ocjena u obliku zvjezdica. Odabirom jednog izvođača pristupa se njegovom profilu. Ako odabrana županija nema registriranih izvođača, prikazuje se poruka da nema izvođača. Stranici „top5“ mogu pristupiti svi neprijavljeni korisnici aplikacije.



Slika 29. Sučelje stranice Top 5

4.7. Iskustva korisnika za stranicu

Aplikaciju ocjeni Majstora moguće je ocjenjivati i komentirati. Iskustva korisnika su vrlo bitna te svaki negativni komentar služi za razmatranje o potencijalnom ažuriranju aplikacije. Na stanici „ocjeneKorisnika“ možemo pregledati sve ocjene korisnika. Definirana je i paginacija koja sadržava četiri komponenta komentara na jednoj stranici. Za komentiranje i ocjenjivanje stranice je potrebno biti prijavljen. U podnožju stranice je moguće pristupiti pregledu svih ocjena te stranici na kojoj se ostavlja komentar. Isto tako na početnoj stanici su prikazana četiri najbolje ocjenjena komentara koja se nalaze u donjem dijelu stranice. Za dodavanje iskustva korištena je ista funkcija kao za dodavanje osvrta u profilu izvođača, koja je vidljiva na slici 29. Nakon spremanja komentara prikazuje se alert koji prikazuje poruku zahvale.

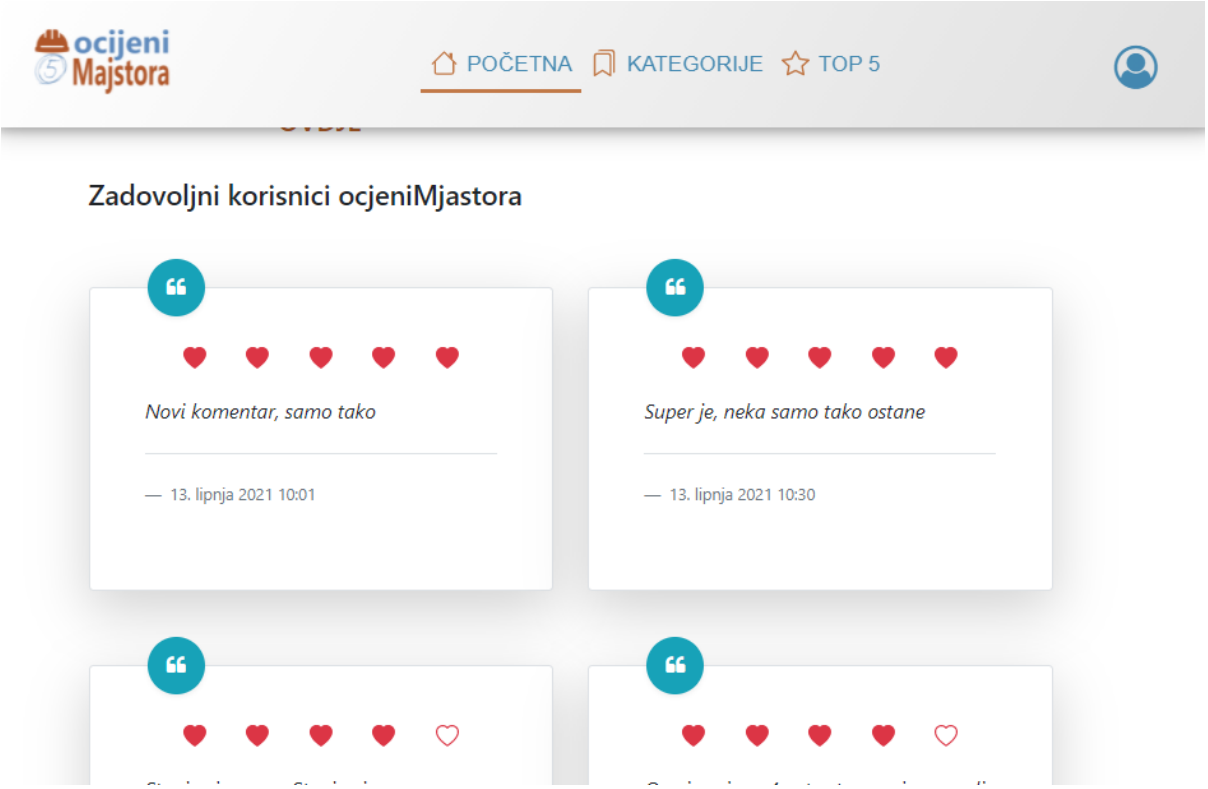
```
spremiOcjenu(){
      db.collection("ocjeneStranice")
                .add({
                  ocjena: this.ocjenaStranice,
                  komentar: this.text,
                  vrijemeObjave: Date.now(),
                })
                .then((doc) => {
                  console.log("Spremljeno", doc);
                        this.$swal.fire({
                        position: "top-end",
                        icon: "success",
                        title: "Hvala Vam što ste izdvojili vremena!",
                        showConfirmButton: false,
                        timer: 1850,
                         });
                         this.$router.replace("/");
                         this.ocjenaStranice = '';
                         this.text = ''
                                  })
                                  .catch((e) => {
                                    console.error(e);
                                  });
                                  },
```

Slika 30. Funkcija za učitavanje ocjena korisnika za stanicu

Prikaz komentara na početnoj stranici je dizajniran po karticama koje se nalaze ispod prikaza izvođača. Poredani su po dva komentara u jednom redu. Umjesto zvjezdica implementirana su srca koja su ispunjena kada su označena. Slika 30 prikazuje sučelje početne stranica na kojoj su prikazane ocjene. Za prikaz ocjena korisnika napravljana je komponenta „zadovoljstvoKorisnika“ koja se pomoću v-for direktive učitava onoliko puta koliko postoji komentara. Komentari zadovoljnih korisnika imaju svoju posebnu kolekciju u Firesotre bazi, a podaci su:

- Komentar
- Ocjena
- Vrijeme objave

Svi komentari mogu se vidjeti odabirom na dugme „pogledaj sve ocjene“ ili u podnožju stranice „ocjene korisnika“. Za aplikaciju se može ostaviti recenzija i na društvenim mrežama kao što su Facebook, Instagram i Twitter.



Slika 31. Prikaz komentara zadovoljnih korisnika starince

4.8. Email sustav obavijesti

Sustav obavještavanja je bitan za informiranje korisnika i radnji na samoj aplikaciji. Vlasnik firme obavješten je svaki puta kada netko komentira njegovu firmu. Prilikom registracije firme potrebno je upisati službeni email firme, na email bit će poslana obavijest da je profil komentiran. U poruci se nalazi poveznica koja direktno vodi na komentirani profil.

Na stranici profila nalazi se gumb pod nazivom „poruka“ s kojom možemo poslati email vlasniku firme. Prilikom odabira gumba otvara se lokalni email program na kojemu možemo pisati željenu poruku. Također, može se poslati poruka na službeni email aplikacije. Za to je potrebno odabrati ikonu u obliku koverte na dnu

aplikacije. Slika 31 prikazuje kod funkcije za slanje poruke preko email adrese. Sustav mora sadržavati službeni email i lozinku aplikacije kako bi pristupio poslužitelju te poslao mail na adresu izvođača.

```
posljiEmail() {
  window.Email.send({
    Host: "smtp.gmail.com",
    Username: "ocjenimajstoraobavjest@gmail.com",
    Password: "jedandva3",
    To: this.podaciProfila[0].sluzbeniEmail,
    From: "ocjenimajstoraobavjest@gmail.com",
    Subject: "Netko je komentirao Vašu firmu!",
    Body:
      "Poštovani, <br><br> Vaša firma je dobila novi komentar. Kako bi vidjeli komentar otiđite na <a href=
      "https://ocjenimajstora.netlify.app" +
      this.$route.path,
  }).then(() => console.log("Email uspješno poslan!"));
},
```

Slika 32. Funkcija za slanje email-a

U svrhu obavijesti i komunikacije sa automatiziranim programom za čavrljanje (eng. chatbot), uspostavljeno je slanje email poruke. Od sada će se u radu koristi izraz chatbot. Prilikom unosa teksta u chatbot, automatski se šalje poruka na službeni email aplikacije. Kasnije se ta poruka, ako je to potrebno dohvaća na odjeljak za najčešća pitanja na stranici za najčešća pitanja.

4.9. Informacije

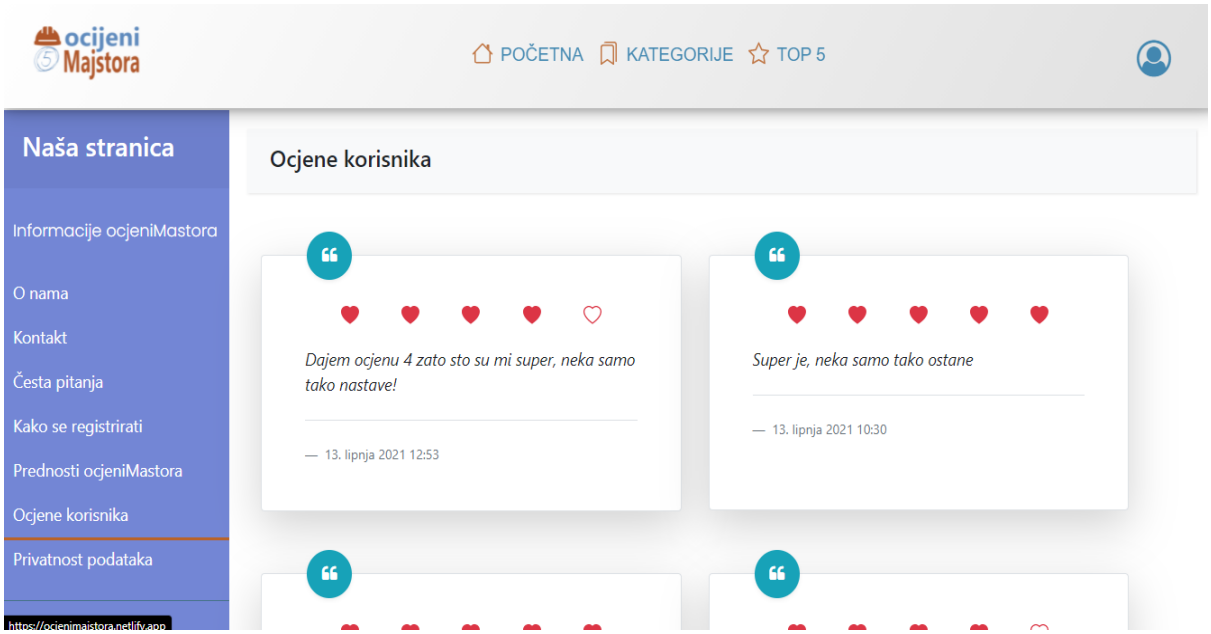
Svaka web stranica ima svoj dio za kontakt podatke, komunikaciju sa korisnicima i slične stavke. U aplikaciji ocijeni Majstor postoji druga navigacijska traka u kojoj su smještene sve bitne informacije vezane za aplikaciju, a to su :

- O nama
- Kontakt
- Česta pitanja
- Kako se registrirati
- Ocjene korisnika
- Privatnost podataka

Stranica je dizajnirana na način da je navigacijska traga sa opcijama smještena na lijevoj stani. Na dnu navigacijske trake se nalazi gumb „ocijeni nas“ koji vodi na stranicu ocjenjivanje zadovoljnih korisnika.

Stranica „o nama “ sadržava sve potrebne informacije o aplikaciji. Na stranici je prikazan i broj prijavljenih izvođača zajedno sa mjestom djelovanja. U „kontakt“ stranici su svi načini na koje je moguće kontaktirati admina stranice. Česta pitanja služe kako bi korisnici mogli vidjeti odgovore na najčešća pitanja. Ako netko želi postaviti pitanje koje će se pojaviti u na toj stranici, može to učiniti slanjem poruke u chatbot ili na službeni email aplikacije. U ocjenama korisnika imamo uvid u sve ocjene zadovoljnih i nezadovoljnih korisnika. U stranici za privatnost podatka prikazani su zakoni o čuvanju korisnikovih podataka, te korištenju istih.

Slika 32 prikazuje sučelje navigacijske trake za spomenute informacije aplikacije.



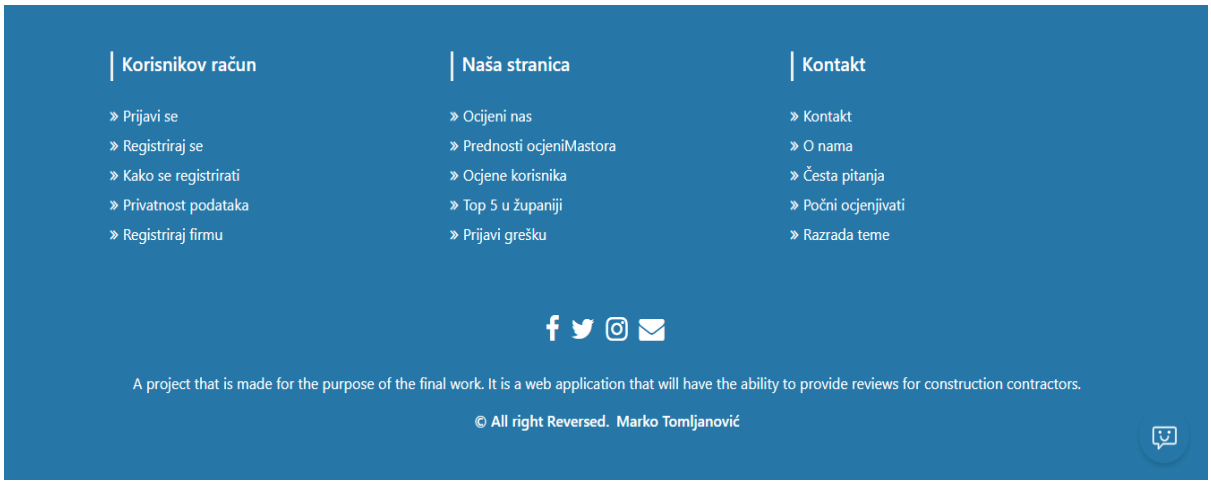
Slika 33. Sučelje stranice sa informacijama

4.10. Podnožje stranice

Podnožje stranice (eng. bottom) je definirano u komponenti te implementirano u App.vue datoteci. Slika 33 prikazuje sučelje podnožja stranice aplikacije. Dizajnirano je vrlo jednostavno sa poveznicama na stavke u aplikaciji. Napravljeno je u obliku komponente te se ta komponenta implementira u App.vue datoteku. Podnožje također sadrži poveznice u obliku ikona društvenih mreža. Cijelo podnožje stranice se dijeli na skupine, svaka skupina sadrži pet poveznica, a to su:

- Korisnikov račun
- Naša stranica
- Kontakt

Korisnikov račun su poveznice za prijavu, registraciju, kako se registrirati, privatnost podataka i registriraj firmu. Skupina naša stranica sadrži poveznice na ocijeni stranicu, prednosti ocjeniMajstora, ocjene korisnika, top 5 u županiji, prijavi grešku. Posljednjoj skupini kontakt pripadaju poveznice o nama, česta pitanja, počni ocjenjivanje, razrada teme.



Slika 34. Prikaz podnožja stranice

4.11. Mobilna verzija

Danas su sve web stranice na internetu prilagođene mobilnim uređajima te svaki korisnik provede više vremena na mobitelu nego za računalom. Stoga web stranice ne bi trebale biti kreirane isključivo za računala. Danas se više pažnje posvećuje izradi stranica i aplikacija za mobilne uređaje pa tek onda dizajnu za računala. Isto tako svjetski poznata tražilica Google 2015. godine je nadogradio svoje algoritme da se prilikom pretraživanja prvo prikazuju stranice koje su optimizirane za mobilne uređaje, pa tek onda ostale. Ako se sagledaju sve činjenice, dolazi se do zaključka kako je mobilna stranica više potreba nego što je dodatna stavka.

Najvažnija stvar kod optimizacije za mobilne uređaje je responzivnost. To je svojstvo koje nam omogućava da se elementi prilagođavaju stranici ovisno o veličini ekrana. Najčešće se primjećuju elementi kao što su umanjeni izbornik, smanjene kartice, oblik gumbova i dr. U aplikaciji ocijeni Majstora korišten je okvir BootstrapVue čiji su elementi responzivni bez dodatne optimizacije sa stilskim jezikom. No postoje elementi koje je bilo potrebno dopuniti, te se morala dopuniti dodatna responzivnost nad tim elementima. Da bi se to napravio potrebno je u stilskom jeziku CSS napisati kod koji se aktivira ovisno veličini ekrana. Kako bi aplikacija znala koja je veličina ekrana, u kod upisujemo minimalnu ili maksimalnu rezoluciju koja bi se trebala aktivirati. To se postiže sa pravilom (eng. event) „@media“, koje se koristi za provjeru više različitih stvari vezano za „media“ uređaje, a to su:

- Dužina i širina trenutnog prikaza
- Dužina i širina uređaja
- Orijentacija, da li je uređaj u portret položaju ili nije
- Rezolucija

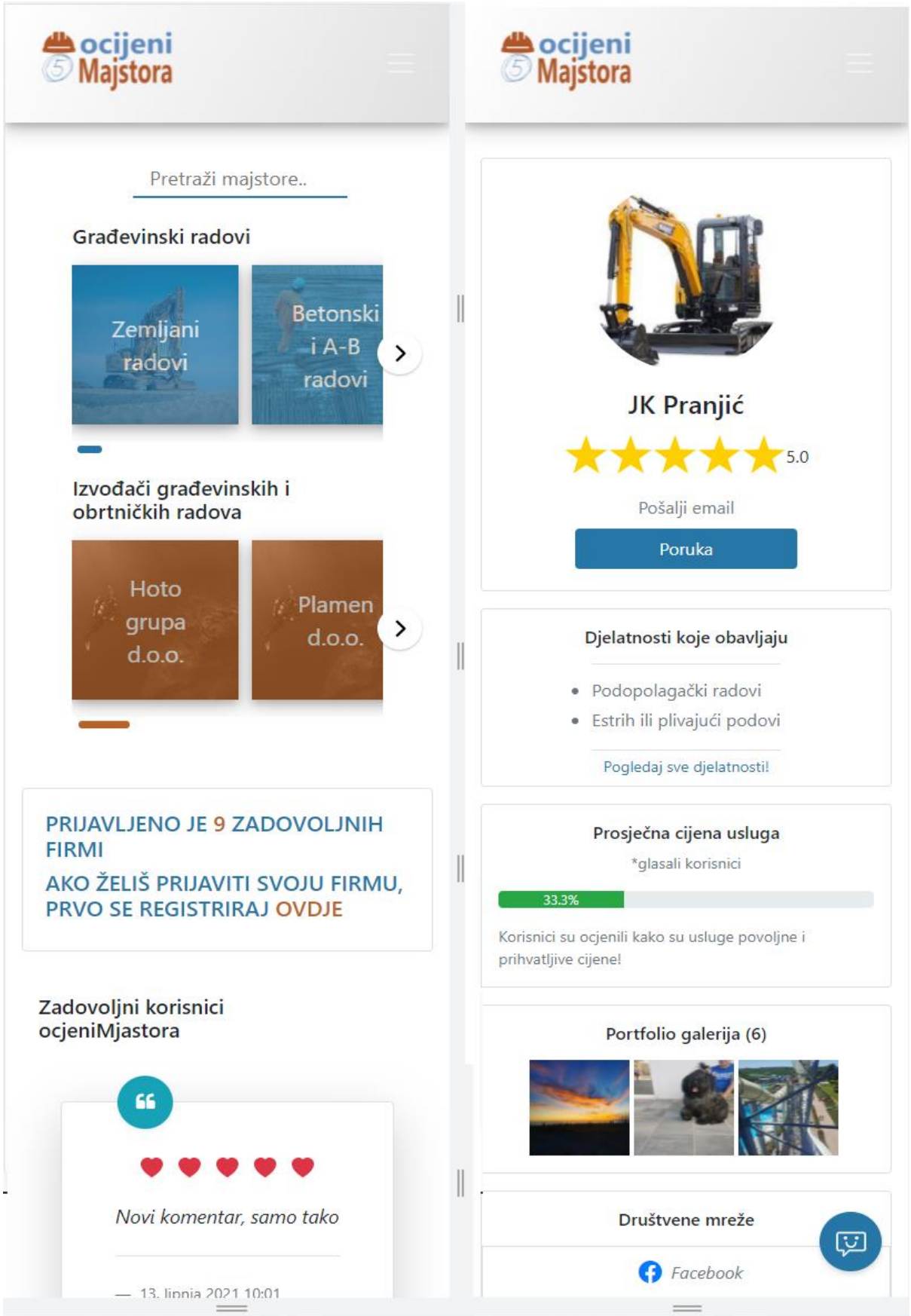
Slika 34 prikazuje CSS kod korišten u aplikaciji koji omogućava sakrivanje i prikaz odabranog elementa. Isti kod može se koristili ako je potrebno stilizirati elemente sa CSS kodom. Na primjer, veličinu slike koja ostane velika prilikom prikaza na mobilnom uređuju, te ju je tada potrebno optimizirati. U kodu „display“ property definira ponašanje prikaza elementa. Korištena je vrijednost „none“ koja briše kompletno cijeli element. Postoji mnogo vrijednosti koje se koriste za „display“, uz „none“ najčešće korištene su: inline, block, flex, initial, inherit.

Danas većina Internet preglednika ima ugrađenu konzolu u kojoj možemo aktivirati prikaz web stranice na manju rezoluciju za mobilne uređaje. Za izradu aplikacije korišten je Google Chrome preglednik u kojemu je moguće prikazati rezoluciju za jednu skupinu određenih mobilnih uređaja. Uz tu mogućnost može se još odabrati u kojem položaju je uređaj te da li je uređaj “mid-tier“, „low-end“ i offline.

```
@media only screen and (min-width: 650px) {
  .pokazi {
    display: none;
  }
}
@media only screen and (max-width: 650px) {
  .sakrij {
    display: none;
  }
}
```

Slika 35. CSS kod za responzivnost

Slika 35 prikazuje sučelje početne stranice i profila optimiziranih za mobilne uređaje. Na početnoj stranice vidi se horizontalni prikaz kartice smanjen na manju veličinu te se smanjuje veličina fonta na cijeloj stranici. Kartice komentara zadovoljnih korisnika postale su responzivne te se prilagodile rezoluciji na kojoj su prikazane. U stranici profila vidi se kako se cijela lijeva strana pomakla na vrh dok je ispod smještena desna strana. Elementi su lijepo prilagođeni veličini rezolucije te su pravilno po važnosti rasporeda sortirani. Također vidi se chatbot koji je umanjen te je ostao na istoj poziciji. Na kraju stranice za elemente osvrta i komentara bilo je potrebno koristiti već spomenuto pravilo „@media“ zbog prilagodbe određenih komponenti.



Slika 36. Sučelje početne stranice i profil za mobilne uređaje

4.12. NPM paketi

Npm je skraćenica za „Node Package Manager“ te pomoću njega možemo instalirati potrebne pakete za aplikaciju. Npm je najveća svjetska programska biblioteka, te se naziva i „Package Manager“. Sadrži više od osamsto tisuća programskih paketa. Programeri otvorenog koda koriste npm za dijeljene vlastitog koda i programa. On dolazi zajedno sa instalacijom Node.js kojeg je potrebno instalirati kako bi se mogao konfigurirati okvir Vue.js. Za prikazivanje trenutne verzije upisujemo u konzolu naredbu „npm -v“. Paketi koji se instaliraju pomoću npm-a uveliko pomažu koje web stranice. Programeri ih koriste kako bi skratili vrijeme izrade web stranice koristeći gotove funkcionalnost stranice. Nakon instalacije paketa, paket postaje zavisni paket (eng. dependacy). Za konfiguraciju takvih paketa koristimo već zadanu datoteku u programu koja se zove package.json. Prije verzije 5.0 npm bilo je potrebno upisati naredbu „–save“ prilikom instalacije paketa da bi se spremio paket u package.json file. U novijoj verziji npm-a, zadano je da se uvijek sprema paket bez obzira na naredbu „--save“. Prilikom izrade velikih projekata doprinos npm paketa je značajan.

Okvir Vue.js ima veliku podršku paketa prilagođenih za njega. Paketi korišteni u aplikaciji, osim Vue.js i Bootstrap-vue su:

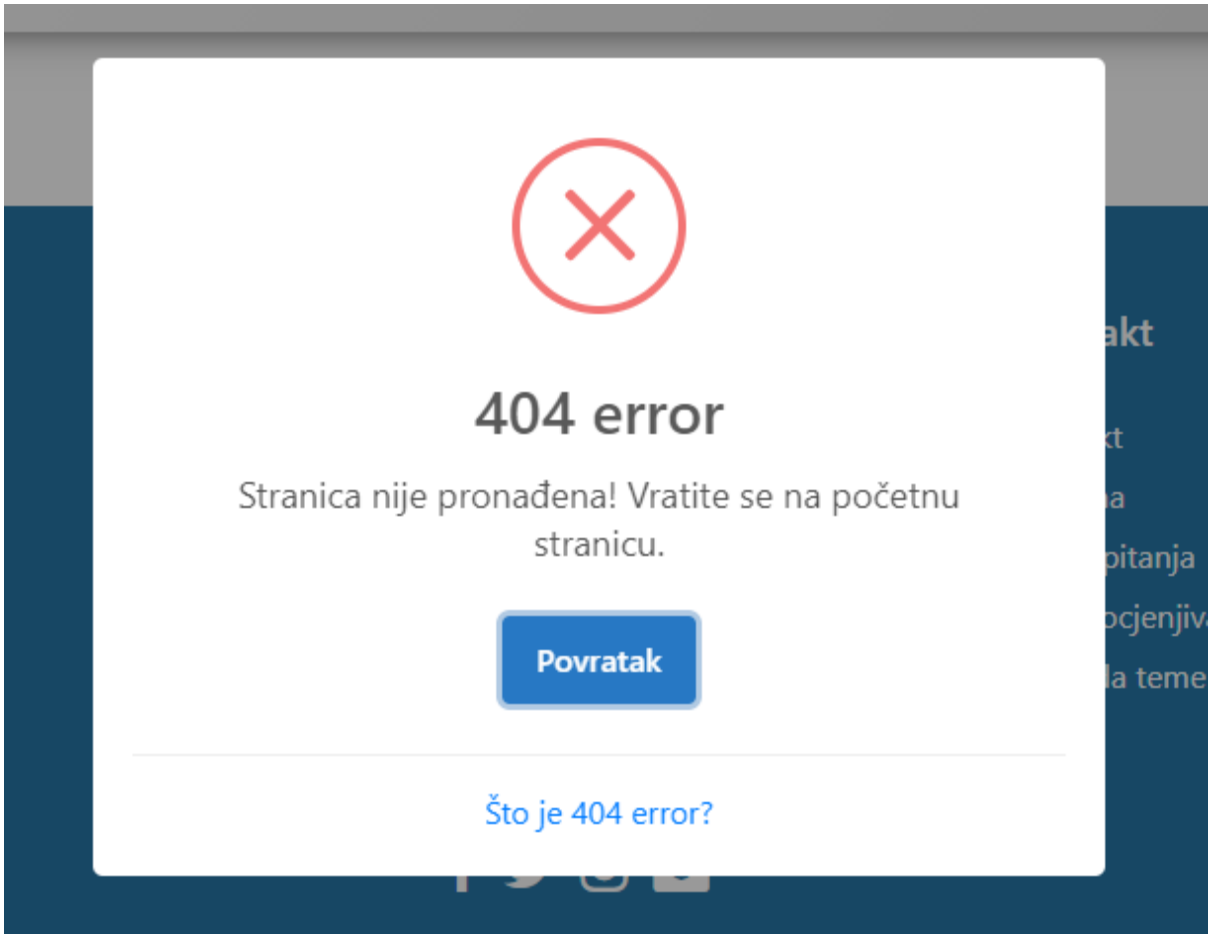
- moment
- Firebase
- sweetalert2
- vue-horizontal
- uuid
- croppa
- vue2-dropzone
- chatbot
- vue-gallery-slideshow

Paket „moment“ služi za provjeru, manipulaciju i oblikovanje formata datuma. Nakon što se dohvati „timestamp“, potrebno ga je pretvoriti u željeni format. Umjesto

formata može koristi i relativno vrijeme koje je prošlo od određene radnje, što je korisno za definiranje objava. Paket se instalira se naredbom „npm install moment“.

Firebase paket je baza podataka ove aplikacije. Nakon instalacije paketa u aplikaciju moguće je koristi Firestore, Storage, Realtime bazu i ostale Firebase usluge. Svaku uslugu je potrebno posebno inicijalizirati, stoga je nabolje kreirati novi JavaScript file za inicijalizaciju Firebase usluga. Paket se instalira se naredbom „npm install firebase“.

Paket sweetalert2 omogućuje korištenje dizajniranih obavijesti upozorenja. Postoji pregršt različitih obavijesti, od kojih se najčešće koriste obavijesti pohale i zabrane. Slika 36 prikazuje obavijest zabrane koja je među najučestalijim te već spomenuti 404 error. Paket se instalira se naredbom „npm install sweetalert2“.



Slika 37. Prikaz obavijesti i 404 error

Paket vue-horizontal je korišten za prikaz kartica izvođača i radova na početnoj stanici. Postoje razne funkcionalnosti koje se mogu dodati kao što su navigacijske strelice, scrollbar, navigacijski brojevi preko kojih se može pristupiti određenim karticama i ostale mogućnosti. Paket se instalira naredbom „npm install vue-horizontal“.

Paket croppa služi za učitavanje slika. Prilikom učitavanja, paket croppa daje mogućnost promjene veličine te rezanje slika. Postoji mnogo opcija kao što su ograničavanje veličine slike, formati slika koji su dozvoljeni, boja prozora za učitavanje i ostale opcije. U aplikaciji je korišten u obrascu za registriranje firme za učitavanje profilne slike. Paket se instalira se naredbom „npm install vue-croppa“.

Paket vue2-dropzone služi kako bi se dodalo više slika odjednom. U aplikaciji je korišten na stranci za registraciju nove firme kako bi se dodale slike u portfolio galeriju. Nakon spremanja u Storage bazu, dohvaćaju se poveznice na slike i spremaju u podatak „galerija“ u Firestore bazi. Paket se instalira se naredbom „npm install vue2-dropzone“.

Paket chatbot omogućava postavljanje chatbot-a u aplikaciju, no definiran je da ga mogu koristiti i vidjeti samo neprijavljeni korisnici. Komunikacija preko chatbot-a se izričito koristi za poboljšanje korisničkog iskustva. Paket se instalira se naredbom „npm i vue-gallery-slideshow“.

Paket vue-gallery-slideshow se koristi za galeriju slika na profilu izvođača. Prilikom odabira slike otvara se prozor preko cijelog ekrana te prikazuje uvećanu sliku. Paket se instalira se naredbom „npm i vue-gallery-slideshow“.

5. DRUŠTVENE MREŽE

Društvene mreže su elektronički mediji prezentirani u obliku web stranica ili aplikacija koje služe kao socijalni instrument za interakciju između korisnika na način da se stvaraju, razmjenjuju i konzumiraju informacije. Sadržaj koji se dijeli najčešće sadrži osobne informacije, tekstualne sadržaje, fotografije i videa. Neke od popularnijih platformi su Facebook, Instagram, Twitter, YouTube i Tik Tok. Korištene su od strane milijarde ljudi i koriste ih sve generacije, iako mlađa populacija u tome predvodi. Područje društvenih medija sklono je stalnim i brzim promjenama posebice kroz posljednje desetljeće. Prvobitna ideja je bila usmjerena komunikaciji između prijatelja, obitelji, interesnih skupina, a svojim razvojem su postale važna alatka u poslovnom svijetu marketinga. Sada je neizbježna metoda komunikacije između ciljane skupine potrošača i poduzeća koji plasira određeni proizvod. Preferencija ove prakse u svrhu oglašavanja donosi brojne prednosti u odnosu na tradicionalne oglašivačke metode. Primjerice lakše i preciznije se dolazi do ciljanog tržišta, a zbog instant komunikacije sa korisnicima brže i efikasnije se dopunjavaju zahtjevi i/ili nedostatci proizvoda ili usluge sukladno zahtjevima potrošača. Kako je široko upotrebljavan efektivno se prožima kroz različite kulture, socioekonomske statuse, ne nalazeći prepreku geografskih granica lakoćom djeluje kako lokalno tako i globalno.

U svrhu promoviranja aplikacije kreirani su profili na najutjecajnijim društvenim mrežama, a to su Facebook, Instagram, Twitter i Google My Business. Očekivani pregledi bi bili od mlađe populacije koja ima potrebu graditi ili adaptirati svoju prvu nekretninu. No isto tako i starija populacija ima mogućnost informirati se i stupiti u kontakt sa aplikacijom preko profila društvenih mreža. Poveznice na profile se nalaze u nastavku.

- Facebook – https://www.facebook.com/Ocijeni-Majstora-100583585711831
- Instagram – https://www.instagram.com/ocijenimajstora/
- Twitter – @ocijenimajstora

6. ZAKLJUČAK

Cilj diplomskog rada je oblikovati web aplikaciju za vrednovanje izvoditelja građevinskih radova. U teoretskom dijelu rada opisan je primijenjeni kod te tehnologije i funkcionalnost aplikacije. U radu je opisan korišteni Vue.js okvir koji svakim danom postiže veću popularnost zbog jednostavnosti korištenja. Na početku izrade aplikacije „Ocijeni majstora“ napravljen je prototip sučelja u programu koji je kasnije implementiran u aplikaciju. Također su prikazane korištene funkcije u JavaScript jeziku koji je danas jedan od najkorištenijih jezika. Prema istraživanju od strane web stranice „Stack Overflow“, JavaScript je na popisu najpopularnijih tehnologija koju koristi gotovo 70% profesionalnih programera u svijetu [11]. Baza podataka je Firebase baza koja se jednostavno implementira u paru sa JavaScript jezikom.

Aplikacija ima za cilj prikazati što veći broj izvoditelja građevinskih radova na jednom mjestu mogućim korisnicima usluga. Za takvo nešto trenutno ne postoji mnogo aplikacija na našem tržištu, stoga je plan istu ponuditi na korištenje sa mogućnošću dogradnje novih informacija po potrebi tržišta. Jedna od potencijalnih funkcionalnosti koja će se implementirati je uvođenje male naknade za rang poduzeća na listi, što bi uveliko pridonijelo marketingu samog poduzeća. Isto tako bi bila naknada za mogućnost uklanjanja jednom godišnje jednog lošeg komentara sa vlastitog profila. To bi spriječilo smanjenje ukupne ocjene firme zbog jednog komentara sa niskom ocjenom, te još jedan kriterij koji daje mogućnost ocijenjenim poduzećima odgovor na korisnikov komentar kako bi se zahvalili za dobar ili opravdali loš komentar i nisku ocjenu. Ovo je prva verzija aplikacije i zasigurno će se još optimizirati postojeći kod, te ažurirati sučelje aplikacije. Kod i opis aplikacije može se pronaći na GitHub profilu koji se nalazi na poveznici u prvom poglavlju.

Prikaz loga poduzeća je marketinški izraz ponuditelja, a visina ponude daje ozbiljnost i u konačnici sigurnost da je ponuditelj u mogućnosti izvršiti ponuđene radove. Stranica dobiva popularnost povećanjem ponuditelja tj. izvoditelja građevinskih radova i potražnjom investitora kao konačni rezultat ozbiljnosti u poslovnom svijetu.

LITERATURA

[1] Daityari, Shaumik. (2021.) Angular vs React vs Vue: Which Framework to Choose in 2021. Dostupno na: https://www.codeinwp.com/blog/angular-vs-vue-vs-react/ [Pristupljeno: 15.6.2021]

[2] Wikipedia.HTML. Dostupno na: https://en.wikipedia.org/wiki/HTML [Pristupljeno: 15.6.2021]

[3] Liu, Shanhong. (2021.) Most used programming languages among developers worldwide, as of 2021. Dostupno na: https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/ [Pristupljeno: 15.6.2021]

[4] Wikipedia. Firebase. Dostupno na: https://en.wikipedia.org/wiki/Firebase [Pristupljeno: 24.6.2021]

[5] Sufiyan, Taha. (2021.) What is Node.js: A Comprehensive Guide. Dostupno na: https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs [Pristupljeno: 1.7.2021]

[6] Richards, Kristen. (2021.) What's new from Firebase at Google I/O 2021. Dostupno na: https://firebase.googleblog.com/2021/05/whats-new-from-firebase-at-google-io.html [Pristupljeno: 17.7.2021]

[7] Walker, Alyssa. (2021.) UML Use Case Diagram: Tutorial with EXAMPLE. Dostupno na: https://www.guru99.com/use-case-diagrams-example.html [Pristupljeno: 28.7.2021]

[8] Nishadha. (2021.) UML Class Diagram Relationships Explained with Examples. Dostupno na: https://creately.com/blog/diagrams/class-diagram-relationships/ [Pristupljeno: 28.7.2021]

[9] Brody, Emily. (2021.) What's new in Figma: August 2021. Dostupno na: https://www.figma.com/blog/whats-new-in-figma-august-2021/ [Pristupljeno: 10.8.2021]

[10] Benjamin, Ogurinka. (2021.) 13 reasons why you should be using Netlify. Dostupno na: https://dev.to/ogurinkaben/13-reasons-why-you-should-be-using-netlify-kgl [Pristupljeno: 03.9.2021]

[11] Quincy, Larson. (2020.) The 2020 Stack Overflow Developer Survey – 65,000 Devs Share Their Salaries, Top Programming Languages, and More. Dostupno na: https://www.freecodecamp.org/news/stack-overflow-developer-survey-2020-programming-language-framework-salary-data/ [Pristupljeno: 11.9.2021]

POPIS SLIKA

Slika 1. Interes kroz vrijeme [1] 3
Slika 2. package.json datoteka 4
Slika 3. Najčešće korišteni programski jezici među programerima širom svijeta [3] ... 7
Slika 4. Sintaksa Vue.js 8
Slika 5. Firebase ažuriranje podataka 10
Slika 6. UML Use Case dijagram 13
Slika 7. UML Class dijagram 14
Slika 8. Sučelje programa Figma 15
Slika 9. Prototip stranice profila 16
Slika 10. Sučelje stranice prijava 18
Slika 11. Funkcija prijava() 19
Slika 12. Sučelje stranice registracija 20
Slika 13. onAuthStateChanged() funkcija 21
Slika 14. Firestore kolekcija 22
Slika 15. Firestore pod-kolekcija 23
Slika 16. Kod dohvaćanja podataka iz Firesotre baze 25
Slika 17. Spremanje slika u Storage 27
Slika 18. Sučelje registracije firme 28
Slika 19. Funkcija ucitaj() u registracija.vue 29
Slika 20. Sučelje profila 31
Slika 21. Kod za ocjenjivanje 32
Slika 22. HTML kod za galeriju slika 34
Slika 23. Dio funkcije za opciju "Slažem se" 35
Slika 24. Dinamička ruta profila 37
Slika 25. Dinamička ruta 404 error-a 37
Slika 26. Funkcija za pretraživanje 38
Slika 27. Sučelje stranice Izvođači 39
Slika 28. Funkcija sortiranja prema ocjenama 39
Slika 29. Sučelje stranice Top 5 40
Slika 30. Funkcija za učitavanje ocjena korisnika za stanicu 41
Slika 31. Prikaz komentara zadovoljnih korisnika starince 42
Slika 32. Funkcija za slanje email-a 43

Slika 33. Sučelje stranice sa informacijama 44
Slika 34. Prikaz podnožja stranice 45
Slika 35. CSS kod za responzivnost 47
Slika 36. Sučelje početne stranice i profil za mobilne uređaje 48
Slika 37. Prikaz obavijesti i 404 error 50