Izrada mrežne aplikacije u Laravel programskom okviru

Medvarić, Dominik

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
STRUČNI PRIJEDIPLOMSKI STUDIJ RAČUNARSTVO

Dominik Medvarić, 0313023900

IZRADA MREŽNE APLIKACIJE U LARAVEL PROGRAMSKOM OKVIRU

Završni rad

Čakovec, rujan 2024.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
STRUČNI PRIJEDIPLOMSKI STUDIJ RAČUNARSTVO

Dominik Medvarić, 0313023900

IZRADA MREŽNE APLIKACIJE U LARAVEL PROGRAMSKOM OKVIRU

DEVELOPING A WEB APPLICATION IN THE LARAVEL FRAMEWORK

Završni rad

Mentor:
dr. sc. Sanja Brekalo, prof. struč. stud.

Čakovec, rujan 2024.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

PRIJAVA TEME I OBRANE ZAVRŠNOG/DIPLOMSKOG RADA

Stručni prijediplomski studij:

☑ Računarstvo ☐ Održivi razvoj ☐ Menadžment turizma i sporta

Stručni diplomski studij Menadžment turizma i sporta: ☐

Pristupnik: Dominik Mezovski, JMBAG: 0313023000
(ime i prezime)

Kolegij: PHP programiranje
(na kojem se piše rad)

Mentor: doc. dr. sc. Sanja Brekalo, pred. stručni stud.
(ime i prezime, zvanje)

Naslov rada: IZRADA MREŽNE APLIKACIJE U LARAVEL PROGRAMSKOM OKVIRU

Naslov rada na engleskom jeziku: DEVELOPING A WEB APPLICATION IN THE LARAVEL FRAMEWORK

Članovi povjerenstva: 1. Bruno Tiskenjak, predsjednik
(ime i prezime, zvanje)
2. Jurica Tiskenjak, član
(ime i prezime, zvanje)
3. Sanja Brekalo, mentor
(ime i prezime, zvanje)
4. Marija Mezonaruk, zamjenski član
(ime i prezime, zvanje)

Broj zadatka: 2023 RAČ 10

Kratki opis zadatka: Aplikacija se izrađuje u PHP u korištenjem Laravel programskog okvira. Koristi se baza podataka za pohranjivanje podataka koji se skupljaju od strane korisnika i administratora. Aplikacija se izrađuje na primjeru mrežne trgovine za kupnju i iznajmljivanje nekretnina. Za aplikaciju se predviđa da korisnici koji su prijavljeni unose svu robu koja se prodaje u sustav. Pozadinski dio aplikacije omogućuje brisanje računa i vođenje dostupnih nekretnina za iznajmljivanje. Administratori brišu račune automatski ali je omogućeno mijenjanje podataka na bilo kojim računima. Klijenti u aplikaciji mogu pregledavati vlastite narudžbe i ispisivati račune u PDF formatu.

Datum: 17.09. Potpis mentora:

ZAHVALA

Želio bih zahvaliti svim profesorima Međimurskog veleučilišta u Čakovcu koji su mi pomogli u stjecanju znanja, a posebice svojoj mentorici na pomoći i razumijevanju tijekom izrade završnog rada. Veoma sam zahvalan svojim kolegama uz koje sam studirao, a osobito svojoj obitelji koji su također bila velika podrška.

Dominik Medvarić

SAŽETAK

U radu se koristi programski jezik PHP, poznat po svojoj popularnosti u razvoju web aplikacija. Njegova široka upotreba rezultirala je razvojem brojnih programskih okvira koji olakšavaju razvoj aplikacija programerima.

Cilj rada bio je izraditi web-aplikaciju za iznajmljivanje i kupnju nekretnina pomoću Laravel programskog okvira. Izrađena je aplikacija koja omogućuje korisnicima da kreiraju vlastite korisničke račune kako bi pregledavali dostupne nekretnine, pristupili detaljnim informacijama o nekretninama te postavljali vlastite oglase za prodaju ili iznajmljivanje. Aplikacija omogućuje izdavanje računa i omogućuje jednostavno pretraživanje nekretnina kako bi korisnici lakše pronašli traženu nekretninu. Izrađena aplikacija podijeljena je na korisnički i administratorski dio, pri čemu administrator ima ovlasti za dodavanje novih korisnika, uređivanje postojećih korisničkih računa i brisanje korisnika.

Ključne riječi: web-aplikacija, Laravel, PHP, nekretnine, programski okvir

ABSTRACT

The PHP programming language is used in this project because it is among the more popular languages for web application development, and its widespread use has led to the development of many frameworks that facilitate application development for programmers.

The aim of this project was to create a web application for renting and purchasing real estate using the Laravel framework. The developed application allows users to create their own accounts to browse available properties, access detailed information about each property, and even post their own ads for sale or rent. The application also supports invoice issuance and provides an easy property search feature, enabling users to find their desired property more easily. The application is divided into a user section and an administrative section, where the administrator has the authority to add new users, edit existing user accounts, and delete users.

Key words: web application, Laravel, PHP, real estate, framework

SADRŽAJ

1. UVOD

Web aplikacije u današnje vrijeme igraju ključnu ulogu u svakodnevnom životu i poslovanju. Web aplikacije za iznajmljivanje i kupnju nekretnina pružaju brojne prednosti kako korisnicima, tako i vlasnicima nekretnina. Glavna prednost ovih aplikacija je što korisnicima omogućuju pregled dostupnih nekretnina s bilo koje lokacije ili putem mobilnih uređaja, što smanjuje potrebu za fizičkim posjetom agencijama ili razgledavanjem nekretnina uživo.

PHP (*engl. Hypertext Pretprocessor*) je skriptni jezik osmišljen za izradu dinamičkog web-sadržaja, te se najviše i koristi za izradu mrežnih stranica. [1]Poznat je po svojoj jednostavnosti i fleksibilnosti, što ga čini pristupačnim za sve razine programera. PHP je stvorio Rasmus Lerdorf 1994. godine kao skup Perl skripti za održavanje svoje osobne web-stranice. S vremenom se PHP razvio u programski jezik s velikom zajednicom programera koji kontinuirano doprinose njegovom razvoju i poboljšanju. U ranijem web-razvoju, programeri su pisali kod za svaku komponentu web-aplikacije, uključujući autentifikaciju, pristup bazi podataka i mnoge druge uobičajene funkcije. Kako bi se proces olakšao, razvijeni su programski okviri koji pružaju gotove alate i strukturu za izgradnju aplikacija. Rad se fokusira na Laravel, PHP okvir otvorenog koda koji može olakšati razvoj web-aplikacija. [2] Laravel je razvijen na osnovi Model-View-Controller (MVC) arhitekture, što ga čini organiziranim i skalabilnim za izradu aplikacija. U radu je opisana povijest, značajke i funkcionalnosti Laravela, istražena je njegova arhitektura te prikazana struktura i komponente tipičnog Laravel projekta.

Izrađena je web-aplikacija za kupnju i iznajmljivanje nekretnina kao primjer praktične primjene Laravela. Aplikacija omogućuje korisnicima pregledavanje nekretnina te izdavanje računa za kupljene i iznajmljene nekretnine.

2. LARAVEL PROGRAMSKI OKVIR

Programski okvir *(engl. framework)* je temeljna struktura ili platforma koja olakšava razvoj web-aplikacija. To je skup alata, biblioteka i pravila koji pomažu programerima u bržoj i efikasnijoj izgradnji aplikacija.[3]

Laravel je besplatni programski okvir otvorenog koda koji pruža skup alata i resursa za razvoj PHP web-aplikacija te se ubraja među popularnije PHP programske okvire, uz Symfony, Zend i CodeIgniter. Laravel je jedan od robusnijih okvira za razvoj web-aplikacija, koji nudi razne značajke i funkcionalnosti za izradu aplikacija[4]. Implementirane značajke čine Laravel popularnim izborom među programerima. Okvir pojednostavljuje proces razvoja aplikacija, čime skraćuje vrijeme izrade i olakšava planiranje procesa izrade aplikacija. Laravel uključuje mnoge karakteristike i koncepte iz drugih tehnologija poput ASP.NET MVC-a, Ruby on Railsa i mnogih drugih, što ga čini poželjnim alatom za izradu web-aplikacija. Jedna od ključnih prednosti Laravela je bogat ekosustav i zajednica. Uz Laravel se nude gotova rješenja kao što su Eloquent ORM za rad s bazama podataka, Artisan CLI za automatizaciju zadataka, *Blade* za izradu predložaka, fleksibilan sustav za definiranje ruta i *middleware* za pretprocesiranje HTTP zahtjeva. Navedeni alati omogućuju brzu izradu složenih i sigurnih web aplikacija. Laravel se redovito ažurira, što ga čini modernim i relevantnim za razvoj PHP aplikacija.

2.1. Povijest Laravela

CodeIgniter je 2011. godine bio jedan od popularnijih programskih okvira korištenih u PHP-u.[5] Bio je najčešće korišten jer je bio jednostavan za naučiti i dobro dokumentiran. Web-programeri su stvorili mnoge projekte korištenjem okvira CodeIgniter, ali nedostajale su mu određene bitne značajke kao što su autorizacija i autentifikacija korisnika.

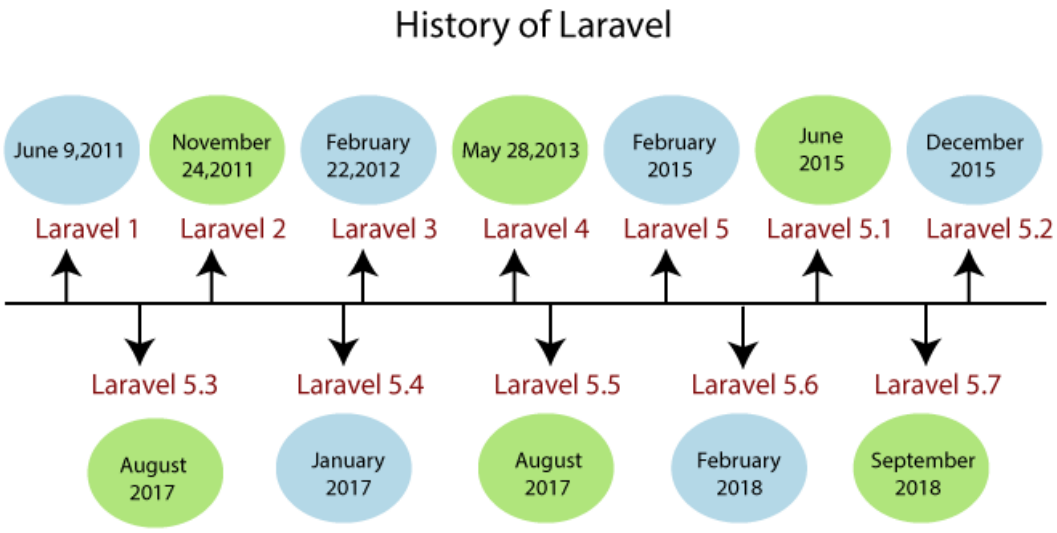
Taylor Otwell, .NET razvojni programer, započeo je vlastiti projekt koristeći ideje iz .NET infrastrukture u lipnju 2011. godine i istog mjeseca objavio Laravel. Prva je verzija pružala ugrađenu podršku za autentifikaciju, lokalizaciju, sesije, jednostavan mehanizam za kontrolu toka (engl. routing), forme te modele i poglede. Druga je verzija objavljena u rujnu iste godine dodavši podršku za kontrolere, te je tako upotpunila Laravel do pravog MVC razvojnog okruženja. U veljači 2012. objavljena je verzija Laravel 3 s brojnim novim svojstvima poput sučelja naredbenog retka *(engl. command-line interface, CLI)* nazvanog Artisan, te podrške za sustave upravljanja bazama podataka.

Nakon pet mjeseci kreatori su odlučili iznova napisati cijelo razvojno okruženje u obliku skupa paketa distribuiranih putem Composer-a. Taylor je razvio skup komponenata pod kodnim imenom Ilimunate i u svibnju 2013. godine objavio Laravel 4, s potpuno novom strukturom. Laravel 5 je objavljen u veljači 2015. godine. Uveden je paket Scheduler koji omogućava organiziranje periodičkih zadataka, Flysystem koji omogućuje korištenje udaljenog spremišta poput lokalnog datotečnog sustava i unaprijeđeno je rukovanje paketima uz pomoć Elixir-a. Laravel se nastavio razvijati velikom brzinom, te je svaka nova verzija donijela značajne inovacije i poboljšanja. Laravel 5.1, objavljen u lipnju 2015., bio je prva dugoročno podržana verzija (LTS) koja je osiguravala podršku za sigurnosne zakrpe i pogreške tijekom tri godine. Laravel 5.3, objavljen u kolovozu 2016., donio je poboljšanja u radu s bazama podataka, te novi sustav obavijesti. U kolovozu 2017. godine, Laravel 5.5 je predstavio novu dugoročno podržanu verziju, uz unaprijeđeno upravljanje pogreškama i poboljšane funkcionalnosti sučelja naredbenog retka Artisan. Laravel 5.7, objavljen u rujnu 2018., donio je unaprijeđene funkcionalnosti poput poboljšane validacije podataka i boljih opcija za korisničku autentifikaciju. Na slici 1 prikazana je povijest verzija Laravela od njegovog prvog izdanja do verzije 5.7. Slika prikazuje izdavanje glavnih verzija, počevši od Laravel 1 koji je izdan 9. lipnja 2011. pa sve do Laravel 5.7 koji je objavljen u rujnu 2018. godine. U veljači 2019. godine Laravel 5.8 donio je unaprjeđenja u brzini izvođenja zadataka i poboljšanja u upravljanju događajima. Laravel 6, objavljen u rujnu 2019., označio je

prelazak na semantičko verzioniranje i donio dugoročnu podršku, kao i novi paket Laravel Vapor, koji omogućava skaliranje aplikacija na AWS Lambda bez potrebe za upravljanjem serverima. Laravel 7, predstavljen u ožujku 2020., donio je nove mogućnosti kao što su poboljšane Blade komponente, Laravel Airlock (kasnije preimenovan u Laravel Sanctum) za autentifikaciju SPA-ova (Single Page Applications) i poboljšanja u performansama. Laravel 8, objavljen u rujnu 2020., uveo je Laravel Jetstream kao unaprijeđeno rješenje za autentifikaciju i korisničke račune, te nove alate za generiranje koda.

Posljednja verzija Laravel 10, objavljena 14. veljače 2023., trenutno je najnovija verzija. Laravel redovito organizira konferenciju pod nazivom Laracon koja se održava u različitim gradovima diljem svijeta. Ova konferencija privlači istaknute stručnjake iz Laravel zajednice te nudi raznovrstan program s predavanjima i radionicama. Laravel se kontinuirano razvija, donoseći nova poboljšanja i značajke koje olakšavaju razvoj web-aplikacija i čine ga jednim od vodećih PHP radnih okvira.

Slika 1. Povijest Laravela



Izvor: https://www.javatpoint.com/history-of-laravel (pristup: 1.9.2023.)

3. PREDNOSTI I NEDOSTACI LARAVELA

Laravel donosi niz prednosti, ali također nosi i određene nedostatke koje bi programeri trebali pažljivo razmotriti pri odabiru ovog okvira.

Laravel nudi sljedeće prednosti:

- Elegantna sintaksa: Laravel je poznat po svojoj čistoj i elegantnoj sintaksi koja olakšava razvoj i održavanje koda.
- MVC arhitektura: Laravel upotrebljava Model-View-Controller (MVC) arhitekturu koja olakšava organizaciju koda i razdvajanje logike aplikacije.
- Ugrađeni alati: Laravel ima ugrađene alate koji pomažu u izradi web-aplikacija kao što su autentifikacija, autorizacija, rute i još mnogo toga.
- Artisan Console: Artisan je komandna linija za upravljanje različitim zadacima, kao što su migracije baze podataka i drugi korisni zadaci.
- Velika zajednica: Laravel ima veliku i aktivnu zajednicu programera, što znači da postoje mnogi resursi i tutorijali za učenje radnog okvira.
- Sigurnost: Laravel uključuje ugrađene mehanizme zaštite poput CSRF zaštite i enkripcije lozinki.
- Besplatan: Laravel je besplatan radni okvir, što znači da nema troškova licence.

Unatoč brojnim prednostima, Laravel ima sljedeće nedostatke:

- Performanse: Iako Laravel pruža dobar balans između brzine razvoja i performansi, neki drugi radni okviri mogu biti brži u specifičnim scenarijima.
- Izazov za manje iskusne timove: Za manje iskusne timove ili početnike, Laravel može biti prevelik i složen za brzo usvajanje.
- Teže rješavanje koda: Duboki kod i mnoge slojevite funkcionalnosti mogu otežati rješavanje problema u usporedbi s jednostavnijim radnim okvirima.

4. MODEL-VIEW-CONTROLLER (MVC) ARHITEKTURA

Laravel je programski okvir koji podržava arhitekturu MVC i olakšava razvoj web aplikacije primjenom MVC arhitekture. MVC *(engl. Model View Controller)* je vrsta softverske arhitekture koja opisuje način na koji se strukturira aplikacija. [6] Trenutno je jedna od popularnijih arhitektura u razvoju web-aplikacija. Osim Laravela koriste je i mnoga radna okruženja: Ruby on Rails, CodeIgniter, Zend, Angular i mnogi drugi.

MVC se sastoji od triju komponente od kojih je svaka zadužena za obavljanje specifičnih funkcija, a to su:

- Model *(engl. Model)*
- Prikaz *(engl. View)*
- Upravitelj *(engl. Controller)*

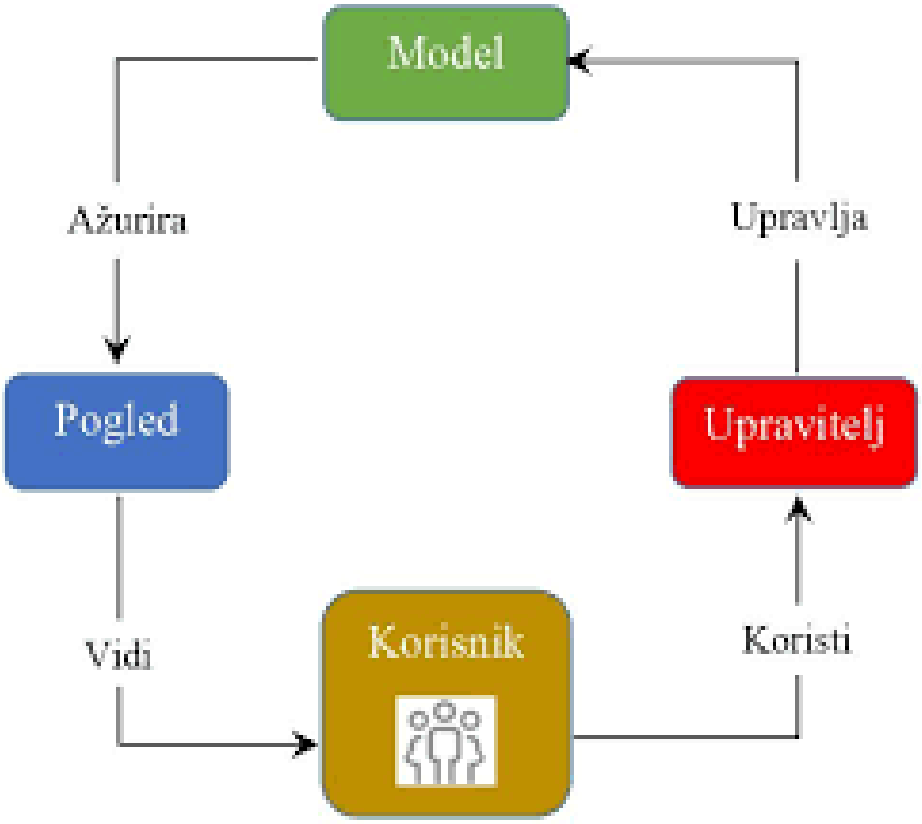
Model *(engl. Model)* se sastoji od glavnih programskih podataka i pripadnih funkcija. Predstavlja jednu ili više klasa, te ima ulogu poveznicu između upravljača i prikaza. Model se bavi upravljanjem i obradom podataka kao što su informacije o korisniku i informacije o objektima iz baze. U sebi najčešće sadrži metode koje mogu manipulirati podacima.

Prikaz *(engl. View)* je odgovoran za prikaz podataka i prikupljanje podataka. Zaslužan je za ono što korisnik vidi, kao primjerice: tablica s podacima i obrazac za unos podataka. Komunicira s upravljačem, odnosno prikazuje dobivene vrijednosti iz upravitelja. Tehnologije koje se koriste u prikazu su obično HTML, CSS i JavaScript.

Upravljač *(engl. Controller)* upravlja logikom aplikacije i komunikacijom između modela i prikaza. Kada korisnik napravi zahtjev (npr. klikom na određeni gumb), taj se zahtjev šalje na odgovarajući upravitelj. On tada koristi model da dohvati potrebne podatke iz baze, obrađuje ih i zatim prosljeđuje na prikaz kako bi bili prikazani korisniku. Upravitelj često sadrži metode koje odgovaraju različitim aplikacijama koje korisnik može izvršiti na aplikaciji(npr. prijava, registracija, slanje forme itd.).

Primjena MVC arhitekture u Laravelu donosi brojne prednosti, uključujući organizaciju koda, ponovnu upotrebu, testiranje, fleksibilnost, bolje upravljanje timskim radom i podršku dokumentacije. [7] Ova arhitektura često se smatra ključnom za razvoj modernih i pouzdanih web-aplikacija.

Slika 2. MVC dijagram



Izvor: https://repozitorij.oss.unist.hr/islandora/object/ossst%3A1946/datastream/PDF/view (pristup: 1.9.2023.)

Slika 2. prikazuje proces koji počinje zahtjevom korisnika (zahtjev može biti napravljen i samim otvaranjem prikaza) kojeg prvo analizira upravljač koji sadrži funkcije koje je potrebno izvršiti. Nakon toga upravljač podatke prosljeđuje modelu. Model radi s podacima i komunicira s bazom podataka, te dodatno obrađuje primljene podatke. Nakon toga model podatke prosljeđuje prikazu, gdje je korisniku vidljiv rezultat njegovog zahtjeva.

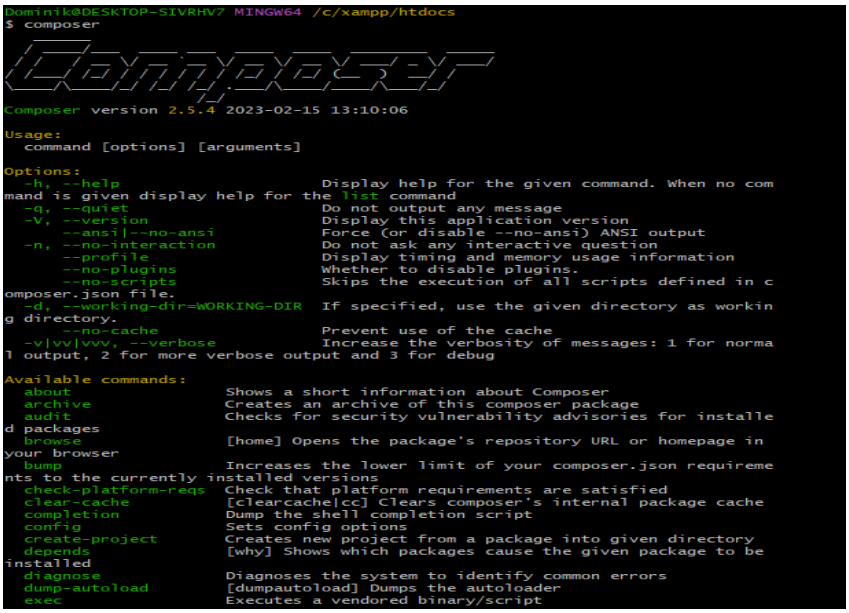
5. LARAVEL RADNO OKRUŽENJE

Laravel podržava dva ključna alata Composer i Artisan koji se koriste za pojednostavljivanje procesa razvoja aplikacija u Laravelu. Oba alata imaju svrhu ubrzavanja radnji kao što su stvaranje novih modela, upravitelja, tablica, prikaza i instalacija novih paketa.

5.1. Composer

Composer je alat za upravljanje ovisnim paketima u aplikacijama napisanim u PHP programskom jeziku i moguće je instalirati i ažurirati sve pakete koji su navedeni kao potrebni u aplikaciji koju programer razvija. [8] Za korištenje Composera potrebno ga je prvo preuzeti s njegove službene internetske stranice „getcomposer.org“, gdje se nalaze i upute za instalaciju. Kada je Composer instaliran na računalu, spreman je za korištenje. Nakon preuzimanja programa, potrebno ga je instalirati naredbom „composer install“ u komandnoj liniji. Prije pokretanja naredbe važno je postaviti se u početnu mapu aplikacije, u koju želite instalirati potrebne pakete. Na slici 3 prikazana je instalacija Composer alata unutar mape „htdocs“.

Slika 3. Ispis naredbe Composer u terminalu



Izvor: Autor

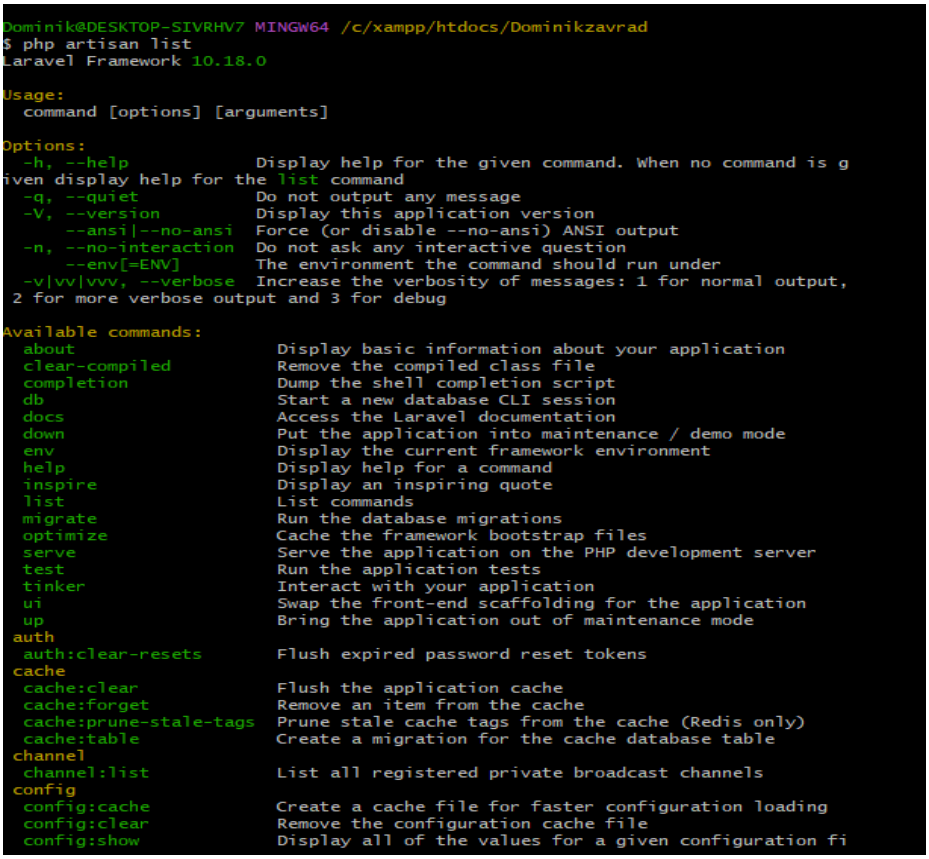
Nakon što se instalacija završi, stvaranje novog projekta provodi se korištenjem naredbe „composer create-project laravel/laravel noviprojekt“. Pomoću ove naredbe

automatski se instaliraju svi potrebni dijelovi i stvoren je novi prazan projekt, odnosno osnovni dijelovi aplikacije. Naredba se može upisati primjerice pomoću Git Bash aplikacije koja pruža terminal okruženje za Winsows u mapi „htdocs“.

5.2. Artisan

Artisan je sučelje naredbenog retka koje se često koristi u Laravelu i uključuje skup korisnih naredbi za razvoj aplikacije. [9] Artisan se koristi za pokretanje i kreiranje migracije, za kreiranje modela i upravitelja i prikazuje popis svih ruta i raznih drugi zadataka. Artisanu se pristupa iz mape u kojoj je kreiran Laravel projekt. Kako bi se provjerila lista naredbi koje sadrži Artisan, potrebno je u Git Bashu unijeti naredbu php artisan list, a rezultat je vidljiv na slici 4.

Slika 4. php artisan list



Izvor: Autor

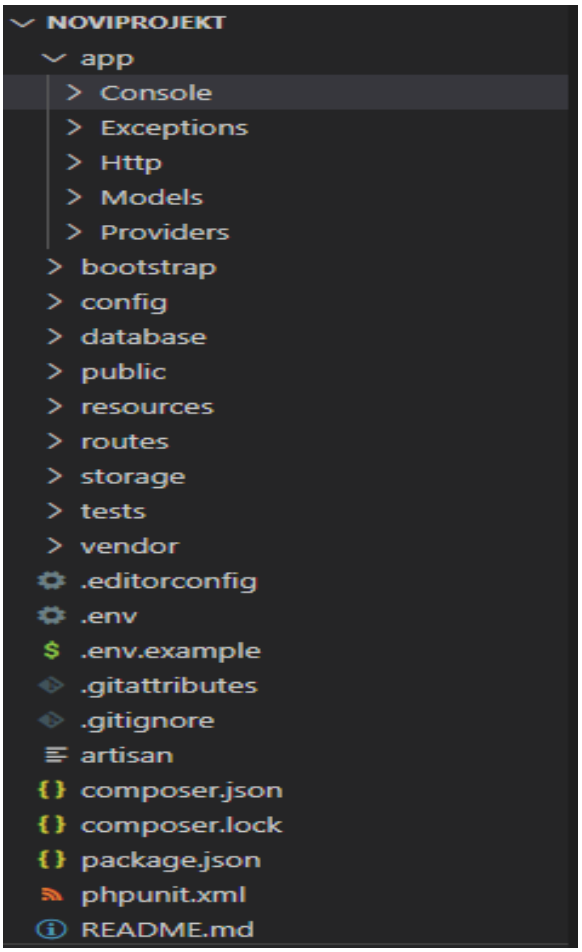
6. ANALIZA STRUKTURE LARAVEL PROJEKTA

Nakon kreiranja novog Laravel projekta kreirat će se mapa s određenim direktorijama i datotekama koji čine novu Laravel aplikaciju. Direktoriji i datoteka koji se nalaze u Laravel projektu su:

- App – ovdje se nalazi veći dio aplikacije kao što su modeli, upravljači i *middleware* koji služi za kontrolu zahtjeva klijenta prema serveru.
- Bootstrap – sadrži „app.php“ datoteku koja služi za podizanje radnog okvira i „cache“ direktorij koji u svojim datotekama ima spremljene podatke predmemorije.
- Config – direktorij u kojem se nalaze sve konfiguracijske datoteke.
- Database – nalaze se „migrations“, seeders“ za popunjavanje tablica s testnim podacima i „factories“ gdje se definira oblik testnih podataka.
- Public – sadrži „index.php“ datoteku koja je ulazna početna točka aplikacije i nalaze se javno dostupne datoteke kao što su slike, CSS datoteke i JavaScript datoteke.
- Resources – direktorij koji sadrži sve poglede web-aplikacije.
- Routes – direktorij koji sadrži sve potrebne rute, a dvije glavne datoteke unutar ovog direktorija su „web.php“ i „api.php“.
- Storage – ovaj direktorij sadrži sve datoteke koje Laravel aplikacija generira i pohranjuje, uključujući privremene datoteke, log datoteke i datoteke za pohranu sesija.
- Tests – sadrži testne skripte.
- Vendor – sadrži sve datoteke potrebne za rad Composera.
- .editorconfig – datoteka koja daje upute o Laravelovim standardima za kodiranje.
- .env i env.example – služe kao konfiguracijske datoteke za nove web-aplikacije.
- .gitignore i .gitattributes – to su konfiguracijske Git datoteke.
- Artisan – omogućava pokretanje Artisan naredbi.
- Composer i composer.lock – ovo su konfiguracijske datoteke za Composer.
- Package – datoteka koja sadrži upute za NPM (*engl. Node Packet Manager*).

- Phpunit – konfiguracijska datoteka za PHPUnit, koju Laravel koristi za uobičajeno testiranje.

Slika 5. Pregled direktorija i datoteka Laravel projekta



Izvor: Autor

6.1. .env Datoteka

„.env“ datoteka je konfiguracijska datoteka za novu web-aplikaciju i obično se podešava na početku rada na aplikaciji. Korištenje „.env“ datoteke omogućava jednostavnu prilagodbu konfiguracije aplikacije za različita okruženja bez potrebe za mijenjanjem izvornog koda aplikacije. Neke postavke koje se mogu izmijeniti u ovoj datoteci su:

- APP_NAME – ime vaše Laravel aplikacije.
- APP_ENV – okolina aplikacije npr. „local“.

- APP_KEY – ključ za kriptiranje aplikacije. Njega možemo stvoriti pomoću naredbe „php artisan key:generate“.
- APP_DEBUG – određuje hoće li korisnici vidjeti pogreške za ispravljanje.
- APP_URL – URL adresa aplikacije.
- DB_CONNECTION – vrsta baze podataka koju koristi aplikacija.
- DB_HOST – host servera baze podataka.
- DB_PORT – port baze podataka.
- DB_USERNAME, DB_PASSWORD, DB_DATABASE – korisničko ime , lozinka za pristup bazi i naziv baze podataka s kojom je povezana aplikacija

Slika 6. .env datoteka

```
.env
1   APP_NAME=Laravel
2   APP_ENV=local
3   APP_KEY=
4   APP_DEBUG=true
5   APP_URL=http://localhost
6
7   LOG_CHANNEL=stack
8   LOG_DEPRECATIONS_CHANNEL=null
9   LOG_LEVEL=debug
10
11  DB_CONNECTION=mysql
12  DB_HOST=127.0.0.1
13  DB_PORT=3306
14  DB_DATABASE=laravel
15  DB_USERNAME=root
16  DB_PASSWORD=
```

Izvor: Autor

Ovo su samo neke od varijabli koje se mogu konfigurirati u „.env“ datoteci i važno je napomenuti da datoteka sadrži osjetljive informacije poput lozinki za bazu podataka i ključeva za enkripciju. Stoga je važno da se ne dijeli s drugima kako bi se zaštitile osjetljive informacije.

6.2. app direktorij

U Laravel-u je „app“ direktorij jedan od ključnih direktorija u strukturi Laravel projekta. Ovaj direktorij sadrži kod programera i glavnu logiku Laravel aplikacije.

Poddirektoriji „app“ direktorija su:

- Console – unutar njega se nalazi datoteka „Kernel.php“ koja sadrži sve Artisan naredbe.
- Http – sadrži poddirektorije „Controllers“ i „Middleware“ koji služi za sigurnost aplikacije.
- Exceptions – sadrži sve metode potrebne za rukovanje iznimkama i sadrži datoteku „Handle.php“ koja obrađuje sve iznimke.
- Providers – sadrži sve davatelje usluga aplikacije koji omogućuju pripremu aplikacije za buduće zahtjeve.
- Mail: ovdje se mogu smjestiti Mailer klase koje olakšavaju slanje e-pošte iz aplikacije.

Svi poddirektoriji i komponente unutar „app“ direktorija bitni su u organizaciji i funkcionalnosti Laravel aplikacije, te omogućuju strukturiranje i razvoj aplikacija na način koji je organiziran, skalabilan i održiv. [10]

7. ANALIZA KOMPONENTI LARAVEL PROJEKTA

Laravel je PHP programski okvir dizajniran za razvoj web aplikacija, opremljen širokim spektrom komponenata koje značajno pojednostavljuju proces izgradnje aplikacija. U ovom je poglavlju detaljnije objašnjena najvažnija komponenta u Laravel projektu.

7.1 Usmjeravanje

Jedna od bitnih komponenta u Laravelu je usmjeravanje. Rute u osnovi služe za usmjeravanje svih zahtjeva aplikacije do odgovarajućeg upravitelja. [11] U direktoriju *Routes* nalaze se datoteke koje Laravel automatski učitava, a to su api.php, channels.php, console.php i web.php. Najvažnije su web.php i api.php. U web.php definiraju se rute za web - aplikaciju ,a api.php se koristi za definiranje API ruta. Za definiranje rute koriste se klase Route te se definira vrsta. Laravel podržava sljedeće zahtjeve: GET, POST, PUT, PATCH, DELETE i OPTIONS. Pri definiciji rute određuje se ime rute (npr. /welcome ili /home definira funkciju tj. upravitelja koji će dohvatiti navedenu rutu). Moguće je dodati i *middleware* koji se izvršava prije nego što se ruta dohvati. Na slici 7 su definirane rute za CRUD (*Create, Read, Update, Delete)* operacije na korisnicima u Laravel aplikaciji. Rute omogućavaju prikaz liste korisnika, kreiranje, uređivanje, ažuriranje i brisanje korisnika. GET metoda koristi se za dohvaćanje liste korisnika i prikaz forme za uređivanje, dok POST metoda omogućava kreiranje novih korisnika i ažuriranje postojećih. DELETE metoda koristi se za brisanje korisnika. Sve su rute zaštićene middlewareom „is_admin“, osiguravajući da samo administratori mogu pristupiti tim rutama.

Slika 7. Primjer rute

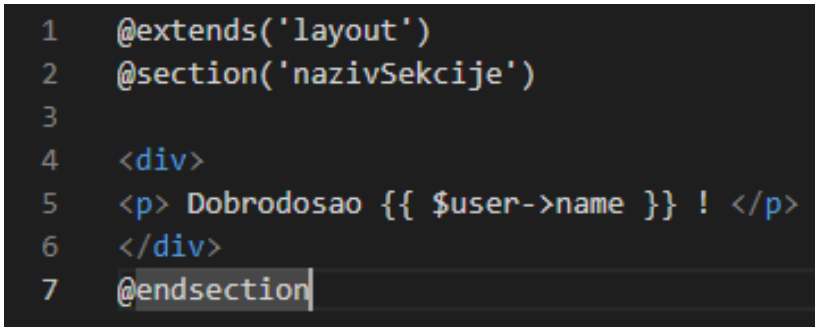
```
Route::get('/korisnici', [App\Http\Controllers\HomeController::class, 'ShowUserlist'])->middleware('is_admin')->name('korisnici');
Route::post('create',[App\Http\Controllers\HomeController::class, 'store'])->middleware('is_admin');
Route::get('/edit/{id}',[App\Http\Controllers\HomeController::class, 'edit'])->middleware('is_admin')->name('edit');
Route::post('/edit/{id}',[App\Http\Controllers\HomeController::class, 'update'])->middleware('is_admin')->name('update');
Route::delete('deleteuser/{id}',[App\Http\Controllers\HomeController::class, 'remove'])->name('delete')->middleware('is_admin');
```

Izvor: Autor

7.2. Blade

Blade je alat za kreiranje predložaka u Laravel okviru. Koristi se za definiranje pogleda *(engl. View)* unutar Laravel aplikacija. [12] Blade omogućava programerima da koriste sintaksu za ugradnju PHP koda izravno u HTML datoteke, čineći kod čitljivim i lakšim za održavanje. Blade predlošci nude vlastitu strukturu, uključujući uvjetne izjave i petlje. Za izradu Blade predloška izrađuje se datoteka i sprema s ekstenzijom .blade.php umjesto .php. Obično se pohranjuju u /resoursce/views direktorij. Glavna prednost upotrebe Bladea je mogućnost nasljeđivanja predložaka i kreiranje sekcija. Kako bi se definirao izgled pogleda, stvara se layout.blade.php u kojoj se deklarira mjesto za sekciju koristeći @yield(nazivSekcije) gdje će svaki pojedini pogled unijeti svoj sadržaj prema potrebi.

Slika 8. Primjer Blade predloška



Izvor: Autor

Podaci koji se predaju pogledu dohvaćaju se unutar dvostrukih zagrada s nazivom varijable. Petlje poput foreach ili if/else ugrađuju se u kod jednostavnim dodavanjem znaka @ bez potrebe za posebnim oznakama za PHP ili HTML kod. Svi pogledi koriste Bootstrap CSS razvojno okruženje, iako njegova upotreba nije obavezna.

7.3. Middleware

Middleware pruža praktičan mehanizam za pregled i filtriranje dolaznih HTTP zahtjeva u web aplikaciji. [13] Middleware se često koristi za zadatke kao što su autentifikacija korisnika i manipulacija zahtjeva i odgovora. Na primjer, u Laravelu postoji middleware koji provjerava autentifikaciju korisnika. Ako korisnik nije prijavljen,

middleware će preusmjeriti korisnika na stranicu za prijavu. S druge strane, ako je korisnik prijavljen middleware će dopustiti da zahtjev nastavi dalje u aplikaciji.

Slika 9. Primjer middlewarea

Izvor: Autor

U Laravelu postoje dvije glavne kategorije middlewarea: globalni middleware i middleware rute. Globalni middleware su middlewareovi koji se primjenjuju na svaki dolazni HTTP zahtjev koji ulazi u aplikaciju, neovisno o tome koja je ruta pozvana. Oni se definiraju u datoteci kernel.php u svojstvu $middleware. Middleware rute su middlewareovi koji se primjenjuju samo na određene rute i one se definiraju u datoteci kernel.php u svojstvu $routeMiddleware.

7.4. Laravel CSRF zaštita

CSRF *(engl. Cross-Site Request Forgery)* je zlonamjeran napad koji se događa kada napadač iskoristi povjerenje prijavljenog korisnika kako bi izveo zlonamjernu radnju u njegovo ime, bez njegovog znanja ili dopuštenja. Laravel automatski pruža CSRF zaštitu kako bi se aplikacija zaštitila od CSRF napada. Laravel generira jedinstveni CSRF token za svaki zahtjev koji dolazi s formom unutar aplikacije. Ovaj se token obično uključuje u sve forme kao skriveno polje (obično nazvano _token). Kada se forma pošalje na server, Laravel provjerava je li se token zahtjev podudara s tokenom koji je spremljen u korisnikovoj sesiji.

Ako se tokeni ne podudaraju, zahtjev se smatra nevažećim. Za uključivanje CSRF zaštite u Laravelu koristi se „@csrf“ Blade naredba, koja automatski generira i uključuje CSRF token u formu.

Slika 10. Primjer CRSF zaštite u formi

```
<form method="POST" actions="/add">
@csrf
<input type="hidden" name="imevlasnika" value="{{Auth::user()->name}}"/>
<label>Ime nekretnine</label>
  <input type="text" name="ime" id="ime" placeholder="ime" />
  <label>Mobile</label>
  <input type="text" name="mobile" id="mobile"placeholder="Mobile" />
  <label>Grad</label>
  <input type="text" name="mjesto" id="mjesto" placeholder="Grad" />
  <label>Adresa</label>
```

Izvor: Autor

8. IZRADA LARAVEL PROJEKTA - APLIKACIJA ZA KUPNJU I IZNAJMLJIVANJE NEKRETNINA

Cilj završnog rada bio je izraditi web-aplikaciju korištenjem Laravel programskog okvira. Pri izradi aplikacije korišteni su sljedeći alati i programi:

- Visual Studio Code
- phpMyAdmin
- XAMPP
- Composer
- GitBash

Izrađena web-aplikacija naziva se *RealEstate* i napravljena je kao primjer mrežne trgovine za kupovinu i iznajmljivanje nekretnina. Kroz ovu aplikaciju korisnicima je omogućena kupnja ili iznajmljivanje nekretnina, a također postoji opcija postavljanja vlastitih nekretnina za prodaju ili iznajmljivanje. Nakon kupnje nekretnina, omogućen je pristup vlastitim računima u PDF formatu. Administrator može mijenjati podatke PDF računa. U razvojnom okruženju web-aplikacija se pokreće korištenjem Artisan naredbe „php artisan serve“. Nakon što se ova naredba izvrši, aplikacija je dostupna putem preglednika tipično na URL-u http://localhost:8000.

Slika 11. Početna stranica web-aplikacije

Izvor: Autor

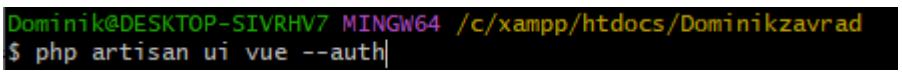
8.1. Autentifikacija korisnika

Laravel značajno pojednostavljuje proces implementacije autentifikacije za programere. Gotovo sve opcije su već prethodno konfigurirane i pohranjene u konfiguracijskoj datoteci config/auth.php.

Za implementaciju autentifikacije prvi je korak instaliranje Composer paketa laravel/ui, koji pruža komponente korisničkog sučelja. Nakon toga se izvršava naredba „php artisan ui vue –auth“. Evo kako ta naredba funkcionira:

- Php artisan ui – je naredba za generiranje korisničkog sučelja u Laravelu.
- Vue – označava da želite koristiti Vue.js kao JavScript okvir za korisničko sučelje.
- --auth – je opcija koja označava da želite generirati osnovne resurse za autentifikaciju.

Slika 12. Naredba za autentifikaciju



Izvor: Autor

Ova naredba automatski stvara potrebne rute i generira *HomeController* s unaprijed definiranim metodama koje se izvršavaju nakon prijave korisnika. Također se generiraju kontroleri za registraciju novog korisnika, autentifikaciju, slanje poveznica za resetiranje

lozinke te kontroler za resetiranje lozinke. Laravel/ui automatski generira sve potrebne prikaze i sprema ih u direktorij resources/views/auth. Ovaj postupak omogućuje brz početak s implementacijom autentifikacije u Laravel aplikaciji i štedi vrijeme pri postavljanju osnovnih funkcionalnosti autentifikacije. Nakon što su resursi generirani, mogu se dodatno prilagoditi i proširiti prema specifičnim potrebama projekta.

8.2. Registracija korisnika

Registracija je važan dio web aplikacije jer omogućuje korisnicima da stvore svoje račune i pristupe osobnim podacima i funkcionalnostima aplikacije. Prilikom registracije, korisnik je obavezan unijeti sljedeće informacije: ime, adresu e-pošte, lozinku i ponovljenu lozinku. Unutar *RegisterController* nalaze se dvije važne funkcije: *validator* i *create*. Funkcija *validator* na slici 13 koristi se za postavljanje pravila validacije koja određuju kako se provjeravaju i prihvaćaju podaci koje korisnik unosi tijekom registracije. Ova funkcija osigurava da podaci budu ispravni i sigurni, sprečavajući potencijalne probleme.

Ovo su neka od pravila koja su definirana u funkciji *validator*:

- *Required* – označava da je polje obavezno i da mora biti ispunjeno.
- *String* – očekivani unos u obliku teksta.
- *Max* i *min* – ograničava na broj znakova.
- *Unique:users* – provjerava jedinstvenost e-mail adrese u tablici *users.*
- *Email* – označava da se očekuje valjani e-mail format.
- *Confirmed* – označava da postoji i polje za potvrdu lozinke koja se mora podudarati s poljem lozinka.

Funkcija *create* na slici 13. ima svrhu stvaranja novog korisničkog računa u bazi podataka nakon što su svi korisnički podaci prošli kroz proces provjere i validacije. U okviru ove funkcije koristi se i *Hash* kako bi se osiguralo sigurno pohranjivanje korisničke lozinke u bazi podataka. Korištenjem funkcije *Hash*, lozinka se pretvara u kriptirani oblik prije nego što se sprema u bazu podataka. Ovaj korak povećava sigurnost aplikacije jer osigurava da lozinke korisnika nisu pohranjene u čitljivom obliku, čime se štite korisnički podaci od potencijalnih prijetnji i neovlaštenog pristupa.

Slika 13. RegisterController

```
    protected function validator(array $data)
    {
        return Validator::make($data, [
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
            'password' => ['required', 'string', 'min:6', 'confirmed'],
        ]);
    }

    /**

     *
     * @param  array  $data
     * @return \App\Models\User
     */
    protected function create(array $data)
    {
        return User::create([
            'name' => $data['name'],
            'email' => $data['email'],
            'password' => Hash::make($data['password']),
        ]);
    }
}
```

Izvor: Autor

8.2. Prijava korisnika

Kod procesa prijave, od korisnika se traži unos njegove registrirane e-mail adrese i lozinke. Unutar *LoginController* nalazi se funkcija login koja omogućuje korisnicima da se prijave u aplikaciju. Dodana je linija koda „if(auth()->user->is_admin==1“, koja provjerava vrijednost polja „is_admin“ za prijavljenog korisnika u bazi podataka. Ako je vrijednost ovog polja postavljana na 1, to znači da je korisnik administrator i u tom slučaju preusmjeren je na administratorsku stranicu. U suprotnom slučaju, ako korisnik nije administrator, preusmjeren je na početnu stranicu za običnog korisnika.

U slučaju da autentifikacija ne uspije, korisnik je preusmjeren natrag na stranicu za prijavu s odgovarajućom porukom o pogrešci. Ovaj postupak omogućuje različite razine pristupa aplikaciji, što je posebno važno u ovoj aplikaciji jer je potrebno razlikovanje između administratora i običnih korisnika.

Slika 14. LoginController

```php
public function login(Request $request)
{
    $input=$request->all();

    $this->validate($request,[
        'email'=>'required|email',
        'password'=>'required',
    ]);

    if(auth()->attempt(array('email'=>$input['email'],'password'=>$input['password'])))
    {
        if(auth()->user()->is_admin==1){
            return redirect('admin');
        }else{
            return redirect()->route('home');
        }
    }else{
        return redirect()->route('login')
        ->with('error','E-mail adresa ili lozinka su pogrešno upisana!');
    }
}
```

Izvor: Autor

9. VRSTE KORISNIKA U APLIKACIJI

U aplikaciji su korisnici kategorizirani kao administratori ili obični korisnici, što se postiže korištenjem polja „is_admin“ u bazi podataka. Polje „is_admin“ igra ključnu ulogu u utvrđivanju ima li određeni korisnik administratorske ovlasti i pristup određenim stranicama. Administratori su označeni vrijednošću 1, dok su obični korisnici označeni vrijednošću 0. Ova vrijednost automatski se postavlja za svakog registriranog korisnika tijekom procesa registracije. Za kontrolu pristupa određenim stranicama temeljenim na ulozi korisnika (administrator ili obični korisnik), koristi se *middleware* nazvan IsAdmin. Na slici 15 prikazan je kod *middleware*a *IsAdmin* koji igra ključnu ulogu u osiguranju da korisnici koji nisu administratori nemaju pristup određenim stranicama koje su rezervirane samo za administratorske ovlasti.

Slika 15. Middleware IsAdmin

```
public function handle(Request $request, Closure $next): Response
{
    if(auth()->user()->is_admin==1){
        return $next($request);

    }
    return redirect('home')->with('error',"Ne mozete pristupiti ovoj stranici niste admin!");
}
```

Izvor: Autor

Nakon odabira stranice najprije se provjerava „is_admin“ atribut prijavljenog korisnika. Ako je „is_admin“ postavljen na 1, što znači da je korisnik administrator, tada se korisniku dopušta pristup stranici. Ako prijavljeni korisnik nije administrator, korisnik će biti preusmjeren na početnu stranicu s porukom o pogrešci. Ovaj je *middleware* koristan kada se želi ograničiti pristup određenim dijelovima aplikacije.

9.1. Administrator

Administratori su posebni korisnici koji imaju dodatne ovlasti za upravljanje aplikacijom. Njihove ovlasti uključuju dodavanje, uređivanje ili brisanje nekretnina. Mogu vidjeti koje nekretnine korisnici žele kupiti, te su odgovorni za potvrdu kupnje kako bi ona bila uspješno potvrđena. Administratori također imaju pristup svim računima za kupljene i iznajmljene nekretnine i mogu ih mijenjati i brisati podatke. Nakon što se administratori prijave u aplikaciju, automatski su preusmjereni na administratorsku nadzornu ploču *(engl. dashboard)*.

Na administratorskoj nadzornoj ploči administrator ima pristup različitim informacijama i statistikama koje su ključne za upravljanje aplikacijom. To uključuje uvid u broj registriranih korisnika, broj dostupnih nekretnina za kupnju i iznajmljivanje, te broj potvrđenih računa kao što je prikazano na slici 16. Ove informacije omogućuju administratoru praćenje aktivnosti unutar aplikacije.

Slika 16. Administratorska početna stranica

Izvor: Autor

Administrator ima mnogo ovlasti i može obavljati različite operacije nad korisnicima, nekretninama i računima unutar aplikacije Ove su operacije temeljene na osnovnim CRUD principima, što predstavlja skraćenicu za stvaranje *(engl. Create)*, čitanje *(engl. Read*), ažuriranje *(engl. Update)* i brisanje *(engl. Delete)*.

Svaka od ovih operacija ima svoje specifično značenje i ulogu u manipulaciji podacima u bazi podataka:

- Stvaranje omogućuje unos novih podataka.
- Čitanje omogućuje izvlačenje ili dohvaćanje podataka iz baze podataka.
- Ažuriranje omogućuje izmjenu postojećih podataka.
- Brisanje omogućuje uklanjanje podataka.

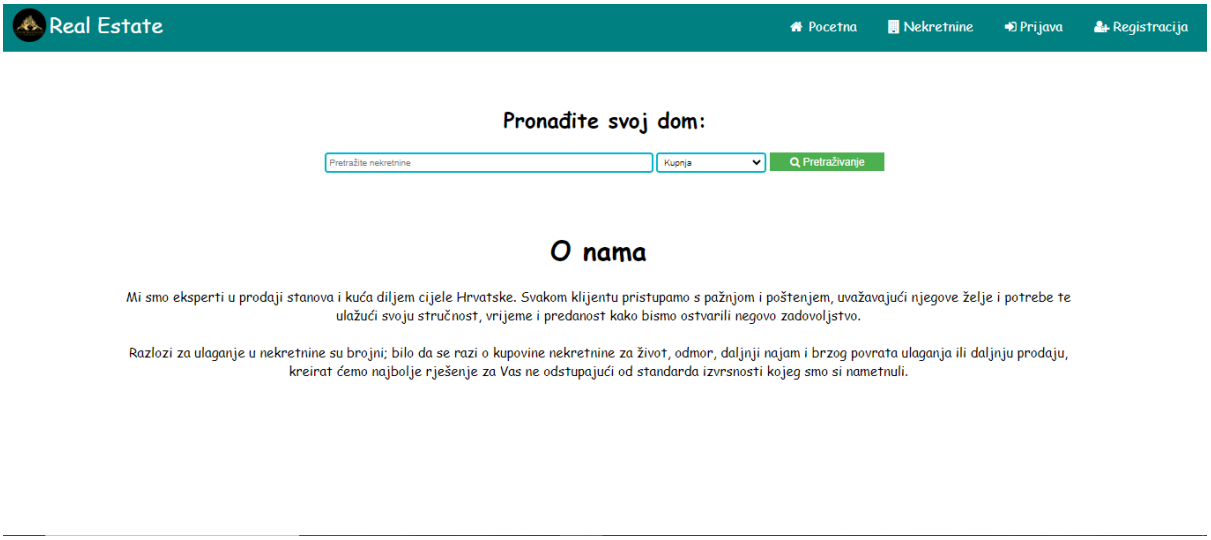
Slika 17. CRUD za nekretnine

Izvor: Autor

9.2. Korisnik

Neprijavljeni korisnici imaju ograničeni pristup aplikaciji te mogu samo pregledavati i pretraživati nekretnine, kao što je prikazano na slici 18. Za punu funkcionalnost i pristup ostalim dijelovima aplikacije, korisnici moraju izvršiti prijavu ili registraciju. Registrirani korisnici mogu pretraživati nekretnine, kupiti nekretninu ili iznajmiti, dodavati ih u košaricu i dodavati svoje nekretnine za prodaju ili iznajmljivanje, mogu pregledavati svoje nekretnine koje su dodali, mijenjati im podatke i brisati ih, te imaju pristup računima za nekretnine koje su prodali ili kupili. Ovaj pogled prikazan je na slici 19.

Slika 18. Pogled neprijavljenog korisnika



Izvor: Autor

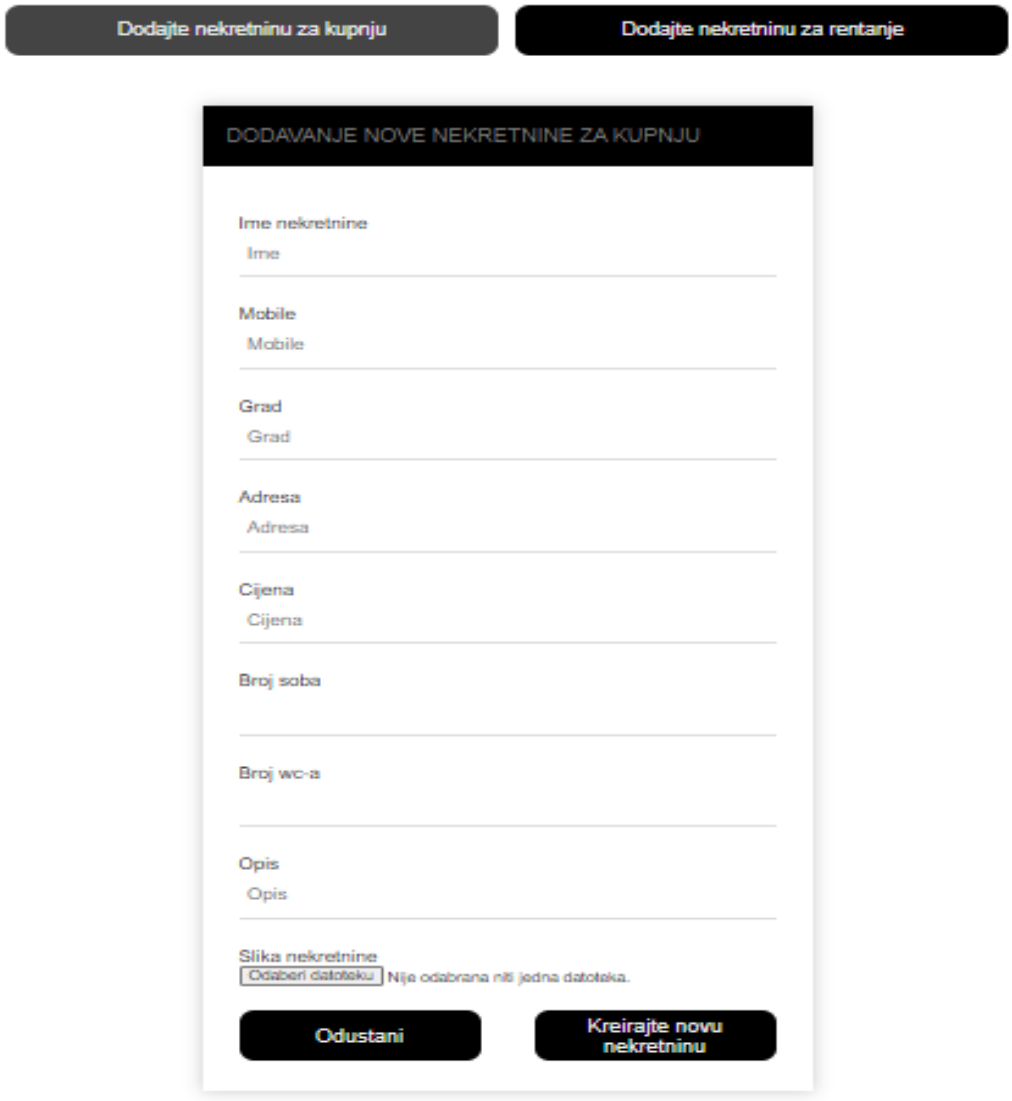
-

Slika 19. Pogled prijavljenog korisnika

10. KUPNJA I IZNAJMLJIVANJE NEKRETNINA

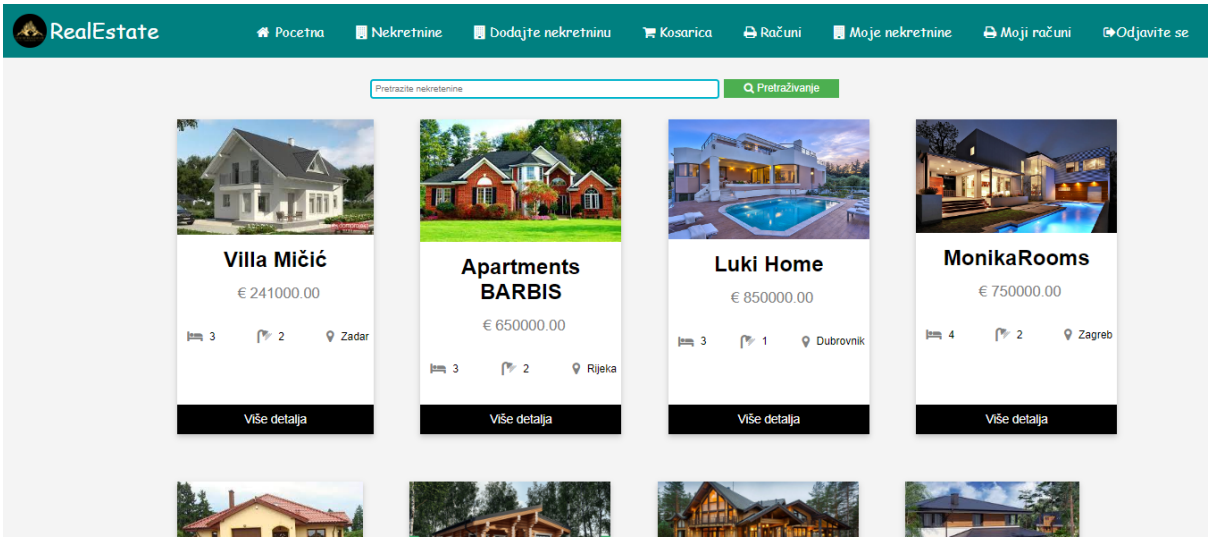
Unutar aplikacije za kupnju i iznajmljivanje nekretnina, registrirani korisnici imaju mogućnost dodavanje vlastitih nekretnina koje žele ponuditi na prodaju ili iznajmljivanje. Prilikom dodavanja nekretnina potrebno je ispuniti formu za dodavanje . U aplikaciji su različite forme, jedna je za dodavanje nekretnine za prodaju, a druga je za nekretnine za iznajmljivanje. Unutar forme korisnik treba navesti sljedeće informacije o nekretnini: ime nekretnine, broj mobitela za kontakt, grad lokacije nekretnine, adresu nekretnine, cijenu, broj soba, broj WC-a, opis i dodati sliku nekretnine.

Slika 20. Forma za dodavanje nekretnine



Izvor: Autor

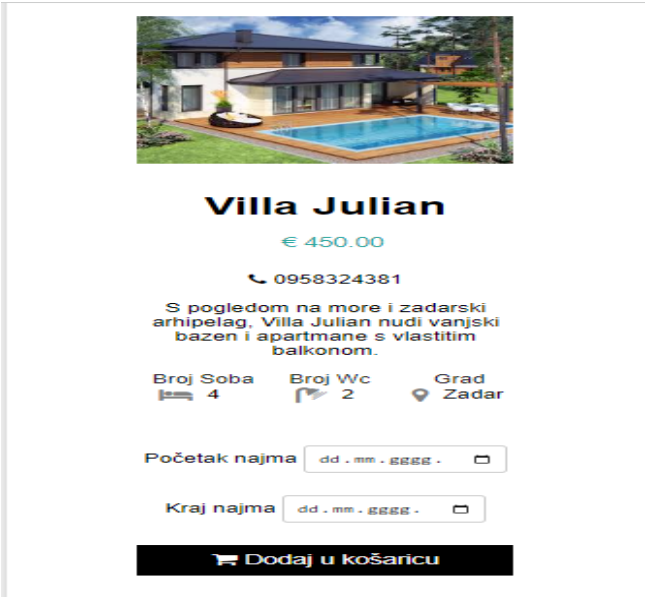
Slika 21. Nekretnina za kupnju



Izvor: Autor

Klikom na određenu nekretninu, korisnik može dobiti dodatne informacije o toj nekretnini i opciju dodavanja u košaricu što je prikazano na slici 22. Za nekretnine koje se iznajmljuju prije nego se dodaju u košaricu, potrebno je odabrati datume za iznajmljivanje te nekretnine, odnosno odrediti željeni datum početka i datum završetka najma. Nakon što se nekretnina doda u košaricu, potrebno je potvrditi kupnju ili najam za određeni period. Nakon potvrde korisnika, transakcija mora biti odobrena od strane administratora kako bi kupnja ili najam bili uspješni.

Slika 22. Kartica za dodavanje nekretnine u košaricu



Izvor:Autor

12. RAČUNI

Kada se korisnik odluči za kupnju, mora potvrditi da želi kupiti odabranu nekretninu. Nakon što korisnik potvrdi da želi kupiti nekretninu i administrator mora potvrditi kako bi nekretnina bila uspješno kupljena. Aplikacija omogućuje korisnicima pregledavanje računa u PDF formatu. Račun u PDF formatu sadrži sve bitne informacije o odabranoj nekretnini, datum iznajmljivanja i ukupnu cijenu. Za generiranje i preuzimanje ovog PDF računa koristi se funkcija „print_pdf“ koja je prikazana na slici 23. Ova funkcija omogućuje korisnicima da jednostavno preuzmu račun kao trajan zapis o svojoj kupnji ili iznajmljivanju nekretnina.

Slika 23. Funckija print_pdf

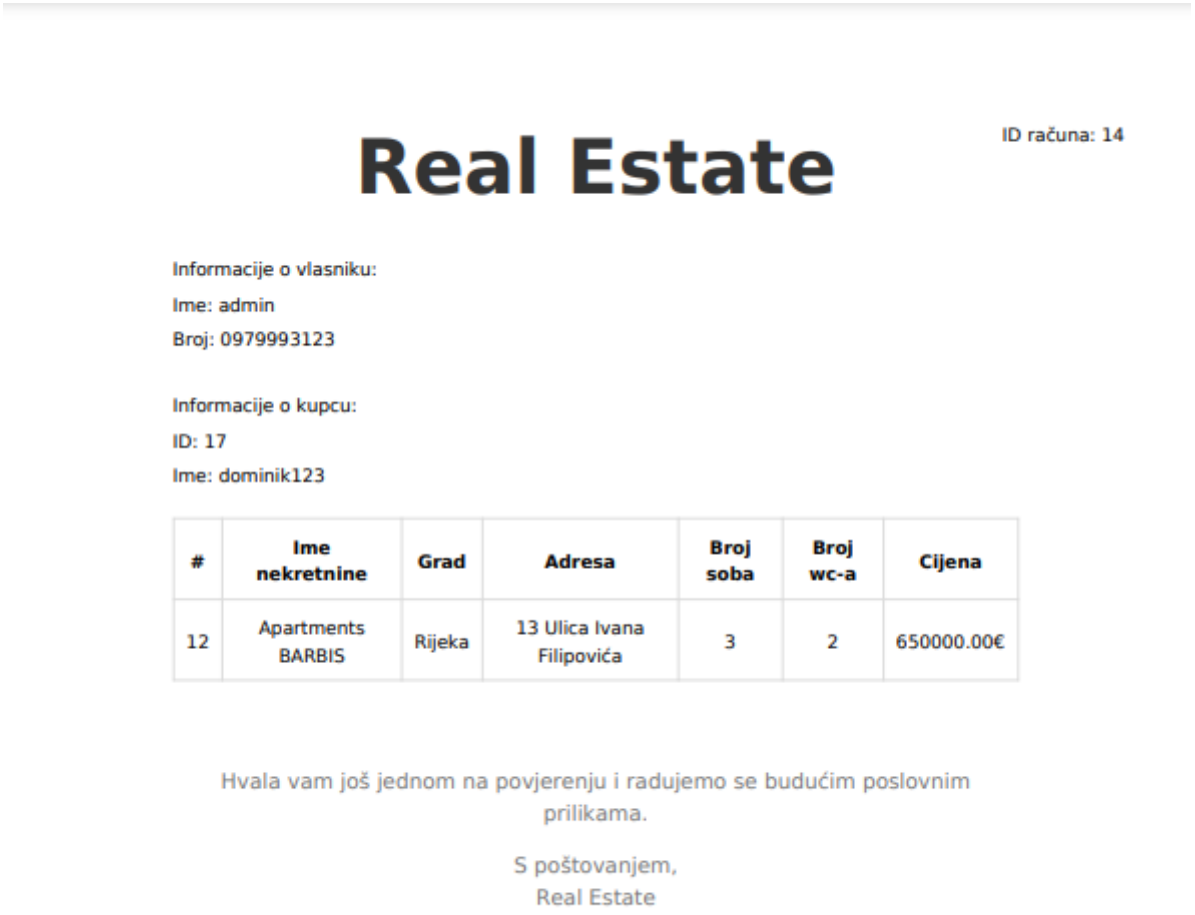
```
        public function print_pdf($id){
         $bill = bill::find($id);

$pdf=PDF::loadView('bill.pdf',compact('bill'));
return $pdf->download('bill.pdf');
        }
```

Izvor: Autor

U prvoj liniji koda traži se račun u bazi podataka koji ima identifikator (ID) jednak onome koji je proslijeđen kao argument funkciji „print_pdf“. Ovo se postiže korištenjem Eloquenta, što omogućuje interakciju s bazom podataka. Nakon što se pronađe račun, u drugom dijelu koda koristi se određena biblioteka za generiranje PDF-ova kako bi se stvorio sam PDF dokument. U tom PDF dokumentu su sve informacije o računu za nekretnine, a primjer računa je prikazan je na slici 24. Nakon što je PDF dokument uspješno generiran, koristi se metoda „download“ kako bi se omogućilo korisniku preuzimanje tog PDF dokumenta. Naziv PDF datoteke je „bill.pdf“, koji se koristi kao naziv prilikom preuzimanja.

Slika 24. Račun



Real Estate

ID računa: 14

Informacije o vlasniku:

Ime: admin

Broj: 0979993123

Informacije o kupcu:

ID: 17

Ime: dominik123

#	Ime nekretnine	Grad	Adresa	Broj soba	Broj wc-a	Cijena
12	Apartments BARBIS	Rijeka	13 Ulica Ivana Filipovića	3	2	650000.00€

Hvala vam još jednom na povjerenju i radujemo se budućim poslovnim prilikama.

S poštovanjem,
Real Estate

Izvor: Autor

13. BAZA PODATAKA

Baza podataka je ključna u Laravelu jer omogućuje pohranu i učinkovito upravljanje podacima kroz Eloquent ORM i migracije. Ona olakšava organizaciju podataka, manipulaciju njima te definiciju odnosa među tablicama, što čini rad s podacima bržim. U aplikaciji se koristi phpMyAdmin, besplatan alat pisan u PHP-u, koji služi za upravljanje i administraciju MySQL-a putem web sučelja. Omogućuje kreiranje, uređivanje i brisanje baza podataka; tablicu i zapis, kao i izvođenje SQL upita i migracija. Laravel migracije omogućuju jednostavno izvođenje promjena u bazi podataka bez potrebe za izravnim pristupom alatu za upravljanje bazom podataka, poput phpMyAdmina. Migracije omogućuju kreiranje, izmjenu i brisanje tablica i stupaca. Naredba „php artisan migrate“ koristi se za izvršavanje migracije u Laravel aplikaciji. Laravel koristi Eloquent ORM i Query Builder za interakciju s bazom podataka. Eloqent ORM (*engl. Object-Relational Mapping*) je sustav u Laravelu koji omogućuje razumljivu interakciju s bazom podataka putem modela. Svaki model u Eloquentu predstavlja jednu tablicu u bazi podataka, a redci u toj tablici predstavljaju pojedine zapise. Query Builder u Laravelu je alat koji omogućuje programerima da kreiraju i izvršavaju SQL upite na fleksibilan način, koristeći PHP sintaksu umjesto da pišu sirove SQL upite. On pruža čitljiv način za rad s bazama podataka, omogućujući složene upite bez potrebe za direktnim SQL-om. Baza podataka kreira se putem web-preglednika odabirom opcije „DataBases“ i unosom naziva baze podataka. Nakon toga se odabire utf84mb4_general_ci kao *collation* kako bi se osigurala podrška za različite jezike i posebne znakove. Klikom na *Create* kreira se baza podataka. Nakon što je baza podataka kreirana, veza s bazom podataka konfigurira se u .env datoteci, gdje se unosi naziv kreirane baze podataka.

14. ZAKLJUČAK

Laravel je programski okvir otvorenog koda koji omogućuje brz i učinkovit razvoj web-aplikacija. Aplikacija za kupnju i iznajmljivanje nekretnina izgrađena na Laravelu pruža korisnicima i administratorima mnoge prednosti i funkcionalnosti. Izrađena aplikacija olakšava korisnicima pretragu i pregled nekretnina, dok administratorima omogućuje učinkovito upravljanje korisničkim računima, nekretninama i računima.

Glavne značajke aplikacije uključuju mogućnost registracije korisnika, pretragu nekretnina i dodavanje nekretnina za prodaju ili iznajmljivanje te generiranje PDF računa.

Laravel kao programski okvir osigurava stabilnost i sigurnost aplikacije te olakšava razvoj i održavanje. Ova aplikacija za kupnju i iznajmljivanje nekretnina pruža korisnicima jednostavan način za interakciju s nekretninama, a administratorima alate za učinkovito vođenje web-aplikacije.

Literatura

1. April Bohnert, What is PHP?, https://www.hackerrank.com/blog/what-is-php-programming-language-introduction/ (Datum pristupa:10.10.2023.)
2. Eric L. Barnes, Laravel news, https://laravel-news.com/ (Datum pristupa: 25.8.2023.)
3. Javatpoint, Laravel Tutorial, https://www.javatpoint.com/laravel (Datum pristupa: 2.9.2023.)
4. Justin, 10 Most popular frameworks in PHP, https://www.atatus.com/blog/ten-most-popular-frameworks-in-php/ (Datum pristupa: 29.10.2023.)
5. Kinsta, The Laravel PHP Framework, https://kinsta.com/knowledgebase/what-is-laravel/ (Datum pristupa: 5.9.2023.)
6. Laravel, Laravel Documentation, https://laravel.com/ (Datum pristupa: 25.8.2023.)
7. Maks Surgury, History of Laravel PHP framework, https://maxoffsky.com/code-blog/history-of-laravel-php-framework-eloquence-emerging/ (Datum pristupa: 27.8.2023.)
8. Marko Ivančić, Uvod u upravljanje php paketima pomoću alata composer - 1.dio ,https://sistemac.srce.hr/index.php/node/101(Datum pristupa:15.9.2023.)
9. Pusher, How Laravel implements MVC and how to use it effectively: https://pusher.com/blog/laravel-mvc-use/ (Datum pristupa:1.9.2023.)
10. Saurabh Barot, Laravel vs ASP.NET: What are the differences? ,https://aglowiditsolutions.com/blog/laravel-vs-asp-dot-net/ (Datum pristupa:10.8.2023.)
11. Sunny Patel, Laravel routing, https://www.cmarix.com/blog/laravel-routing/#:~:text=What%20Is%20the%20Main%20Purpose,request%20triggers%20the%20correct%20response (Datum pristupa:5.11.2023.)
12. Tutorialspoint, Laravel – Blade Templates, https://www.tutorialspoint.com/laravel/laravel_blade_templates.htm (Datum pristupa: 1.9.2023.)
13. w3scholls.in, Laravel Tutorial – W3schools, https://www.w3schools.in/laravel (Datum pristupa: 2.9.2023.)

Popis slika/fotografija

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

Bana Josipa Jelačića 22/a, Čakovec

IZJAVA O AUTORSTVU

Završni/diplomski rad isključivo je autorsko djelo studenta te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, internetskih i drugih izvora) bez pravilnog citiranja. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom i nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, DOMINIK MEDVARIĆ (ime i prezime studenta) pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog rada pod naslovom IZRADA MREŽNE APLIKACIJE U LARAVEL PROGRAMSKOM OKVIRU

te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:

Medvarić

(vlastoručni potpis)