Aplikacija za prijavu i evidenciju stručnih praksi

Došlić, Antonio

Master's thesis / Diplomski rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:904117

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-28**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

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

Diplomski sveučilišni studij

APLIKACIJA ZA PRIJAVU I EVIDENCIJU STRUČNIH PRAKSI

Diplomski rad

Antonio Došlić

Osijek, 2019.

Sadržaj

1. UVOD 1
1.1. Zadatak diplomskog rada 1
2. TEHNOLOGIJE I PROGRAMSKI JEZICI 2
2.1. HTML i CSS 2
2.2. JavaScript 3
2.3. React 3
2.4. Bootstrap 4
2.5. PHP 5
2.6. Laravel 5
2.7. MySQL 6
2.8. XAMPP 7
3. IZRADA APLIKACIJE 8
3.1. Stvaranje lokalnog servera 8
3.2. Stvaranje baze podataka 8
3.3. Instalacija Composera 9
3.4. Stvaranje projekta 9
3.5. Dodavanje React-a u projekt 10
3.6. .env datoteka 10
3.7. Provjera identiteta 11
3.8. Migracije za bazu podataka 11
3.9. Modeli 12
3.10. Rute 13
3.11. Kontroleri 13
3.11.1. AuthController 14
3.11.2. StudentController 14
3.11.3. ProfileController 15
3.11.4. PractiseController 16
3.11.5. ReportController 17
3.12. Pogledi 18
4. IZGLED APLIKACIJE 19
4.1. Početna stranica 19
4.2. Registracija 20
4.3. Prijava 21
4.4. Profil 22

4.5. Dodavanje studenta 24
4.6. Dodavanje prakse 25
4.7. Odabir prakse 26
4.8. Slanje izvještaja 28
4.9. Ocjenjivanje izvještaja 29
4.10. Ocjena 30
4.11. Prikaz ocjena 31
5. ZAKLJUČAK 32
LITERATURA 33
SAŽETAK 34
ABSTRACT 35
ŽIVOTOPIS 36
PRILOZI 37

1. UVOD

Studenti se prilikom zapošljavanja suočavaju s brojnim problemima među kojima se nalazi i nedostatak radnog iskustva. Kako bi se taj problem izbjegao bilo je potrebno nešto poduzeti. Fakulteti, s jedne strane, te tržište rada, s druge strane, ujedinjuju snage i predlažu stručne prakse kao rješenje toga problema. Studenti prolaskom kroz te stručne prakse dobivaju uvid u konkretne projekte te prolaze cjelokupni proces rješavanja određenih problema koji su svakodnevna pojava u praksi. Ono što je studentu najveći dobitak prolaskom kroz stručnu praksu je stjecanje radnog iskustva koje mu je od velike važnosti prilikom daljnjeg zapošljavanja.

Taj proces ponude i potražnje stručne prakse nije nimalo jednostavan budući da je potrebno sve dokumentirati, evidentirati i učiniti dostupnim svima. Iz tog razloga, stvorila se ideja o izradi aplikacije koja bi olakšala prijavu i evidenciju stručnih praksi.

1.1. Zadatak diplomskog rada

Zadatak diplomskog rada je izrada aplikacije koja će služiti da olakša odrađivanje stručnih praksi te pojednostavi prijavu i evidenciju istih. Osim same izrade aplikacije, potrebno je generalizirati sustav evidencije stručne prakse te ga implementirati u aplikaciju.

2. TEHNOLOGIJE I PROGRAMSKI JEZICI

U ovom poglavlju opisane su tehnologije i programski jezici korišteni pri izradi aplikacije.

2.1. HTML i CSS

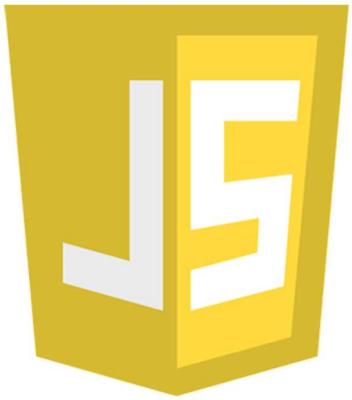
HTML je skraćenica od engleskih riječi „HyperText Markup Language“ [1]. HTML opisuje strukturu web stranica. Sastoji se od niza elemenata koji govore pregledniku kako prikazati sadržaj na web stranici. Elementi su prikazani oznakama, a neke od oznaka su: zaglavlje, paragraf, tablica i slično. Oznake najčešće dolaze u parovima, odnosno oznaka se sastoji od početne i krajnje oznaku. Preglednici ne prikazuju HTML oznake, nego ih upotrebljavaju za prikaz sadržaja stranice. Trenutno je aktualna verzija HTML5.

CSS je skraćenica od engleskih riječi „Cascading Style Sheets“ [2]. CSS opisuje kako HTML elementi trebaju biti prikazani na zaslonu, papiru ili nekom drugom mediju. Pomoću njega možemo uređivati fontove, boje, veličine te mnoge druge karakteristike HTML oznaka. CSS kod se može povezati s HTML dokumentom na dva načina: interno i eksterno. U internom načinu CSS pišemo direktno u HTML dokumentu, dok bi eksterni način bio da vanjsku CSS datoteku uključimo u zaglavlje HTML dokumenta. Trenutno je aktualna verzija CSS3.

Slika 2.1. HTML5 i CSS3 logo

2.2. JavaScript

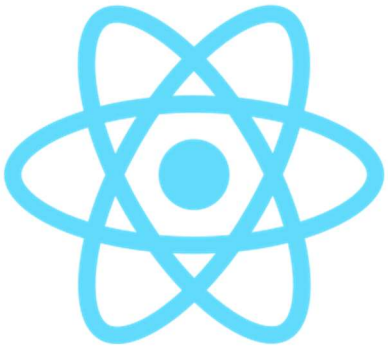
JavaScript, često viđen i pod skraćenicom JS, je programski jezik za HTML i web[3]. Uz HTML i CSS, jedan je od temeljnih tehnologija WWW-a (World Wide Weba). JS omogućuje interaktivne web stranice i bitan je dio web aplikacija. Podržava stilove programiranja temeljene na događajima, funkcionalne i imperativne. Ima brojne API-je za rad s tekstovima, nizovima, datumima te mnogim drugim[4]. U početku je implementiran samo na klijentskoj strani u web preglednicima, no danas ga možemo pronaći i na strani web poslužitelja, kao i izvan weba.



Slika 2.2. JavaScript logo

2.3. React

React, također poznat kako React.js ili ReactJS, je JavaScript biblioteka za izgradnju korisničkih sučelja[5]. Održava ga Facebook i zajednica pojedinih programera i tvrtki. Može se koristiti kao osnova u razvoju jednostrukih ili mobilnih aplikacija jer je optimalan za dobivanje velike količine podataka koje je potrebno zabilježiti. Koristi JSX (JavaScript XML). JSX je XML sintaksno proširenje JavaScripta. Baš kao i kod HTML-a, JSX sadrži oznake i atribute.



Slika 2.3. React logo

2.4. Bootstrap

Bootstrap je besplatno frontend web okruženje za dizajniranje web stranica i web aplikacija[6]. Sastoji se od HTML i CSS predložaka za korisničko sučelje kao i od opcionalnih JavaScript predložaka. Razvili su ga Mark Otto i Jacob Thornton. Razvijen je na Twitteru kao predložak unutarnjih alata, no nakon što je razvijen, objavljen je kao projekt otvorenog tipa 2011. godine. Prvobitni naziv Bootstrapa je Twitter Blueprint. Do danas su objavljene četiri verzije od kojih je verzija 4.3 najnovija.

Slika 2.4. Bootstrap logo

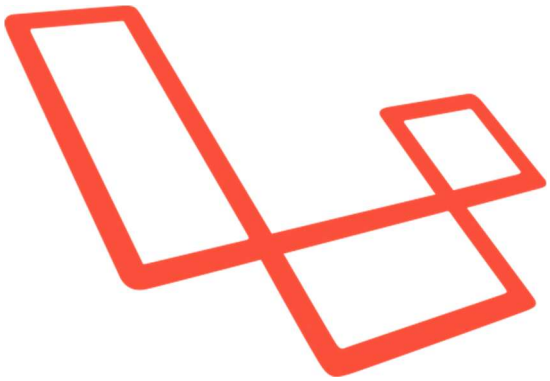
2.5. PHP

PHP je skraćenica od engleskih riječi „Hypertext Preprocessor“ [7]. To je skriptni programski jezik opće namjene koji je izvorno dizajniran za web razvoj. Prva verzija je izašla 1994. godine, a do danas ih je objavljeno sedam. PHP kod se može izvršiti pomoću komandne linije (CLI), ugrađenog u HTML dokument, ili se može koristiti u kombinaciji s drugim sustavima web-predložaka, sustavima za upravljanje web sadržajem i web okvirima. PHP kod najčešće obrađuje PHP interpreter ugrađen kao modul u web poslužitelj ili kao izvršni program.

Slika 2.5. PHP logo

2.6. Laravel

Laravel je besplatno razvojno okruženje otvorenog koda[8]. Baziran je na PHP-u. Namijenjen je razvoju web aplikacija po uzoru na MVC (Model View Controller) arhitekturu[9]. Model je dio u kojem se drži poslovna logika aplikacije, u pogledu se nalazi korisničko sučelje aplikacije, a kontroler je komponenta koja povezuje model i pogled. Neke od njegovih glavnih karakteristika su brzina i skalabilnost. Laravel ima izražajnu sintaksu koja olakšava razvoj web aplikacija.



Slika 2.6. Laravel logo

2.7. MySQL

MySQL je sustav za upravljanje relacijskom bazom podataka[10]. Besplatan je softver otvorenog koda. MySQL baze su relacijskog tipa što se pokazalo kao najbolji način spremanja i pretraživanja velikih količina podataka. Jedan je od najzastupljenijih sustava u svijetu ponajviše zato što je besplatan. Ima podršku brojnih programskih jezika među kojima su Java, PHP, Python i drugi. Trenutno najnovija verzija je 8.0.

Slika 2.7. MySQL logo

2.8. XAMPP

XAMPP je besplatni open-source više platformski internetski poslužitelj razvijen od strane Apache Friends zajednice[11]. XAMPP predstavlja skraćenicu. X označava da je namijenjen svim platformama, A označava Apache HTTP servere, M označava MariaDB baze podataka, prvi P označava PHP, a drugi P Perl. Trenutno najnovija verzija je 7.3 te ju je moguće preuzeti na službenim stranicama.

Slika 2.8. XAMPP logo

3. IZRADA APLIKACIJE

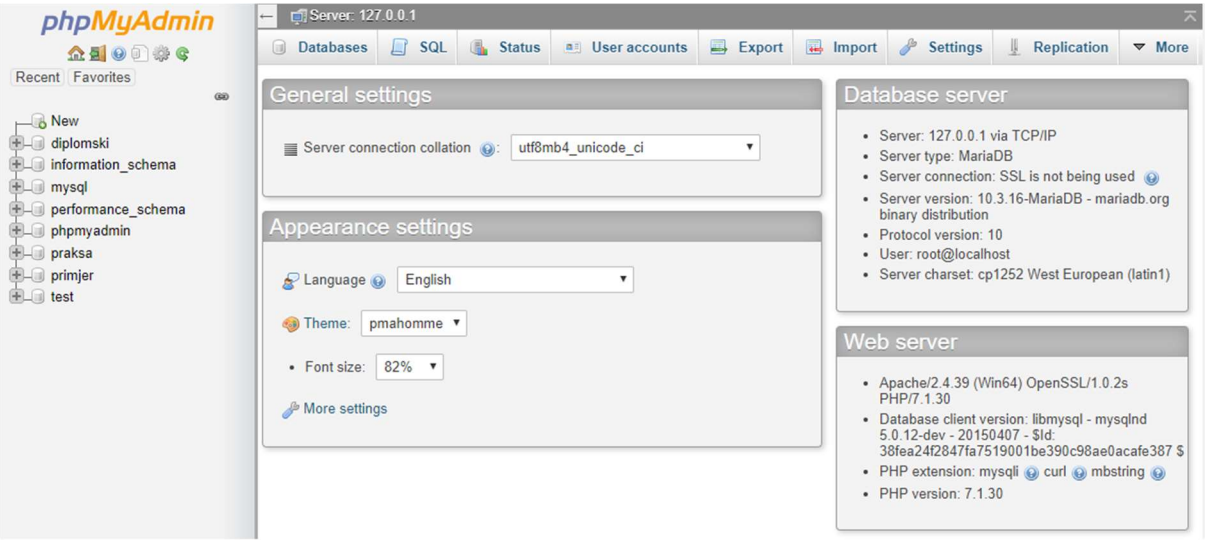
3.1. Stvaranje lokalnog servera

Prvi korak u stvaranju aplikacije bio je stvoriti lokalni server na kojem će se aplikacija pokretati. Pri stvaraju lokalnog servera, korišten je XAMPP. Dostupan je na službenim stranicama te se vrlo lako instalira. Nakon instalacije dovoljno je bilo otvoriti komandnu ploču XAMPP-a te pokrenuti Apache i MySQL kako bi pokrenuli lokalni server.

3.2. Stvaranje baze podataka

Nakon kreiranja lokalnog servera, sljedeći korak je bio stvoriti bazu podataka. Da bi se stvorila baza podataka bilo je potrebno u adresnu traku web preglednika upisati http://localhost/phpmyadmin/ . Otvaranjem te stranice prikazuje nam se phpMyAdmin sučelje.

Na lijevoj strani sučelja imamo ponuđeno naredbu "New". Klikom na tu naredbu otvara nam se polje za unos imena baze podataka. Nakon dodjele imena i klikom na „Create" završili smo sa stvaranjem baze podataka.



Slika 3.1. phpMyAdmin sučelje

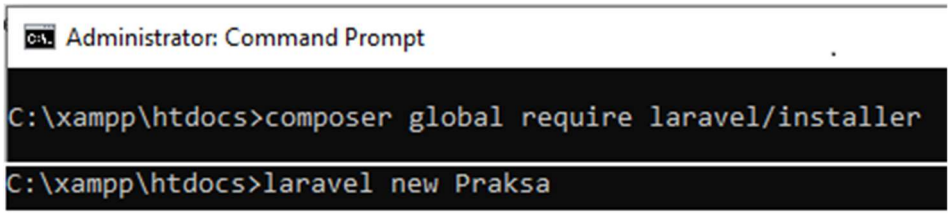
3.3. Instalacija Composera

Zadnji korak prije stvaranja projekta je bio instalacija Composera, dependency managera za PHP[12]. Composer je dostupan na službenim stranicama te nakon preuzimanja slijedi vrlo jednostavna instalacija.

Slika 3.2. Composer logo

3.4. Stvaranje projekta

Budući da je Laravel odabran kao razvojna okolina projekta bilo je potrebno kreirati novi projekt u Laravelu. Stvaranje novog projekta se omogućuje uz pomoć dvije vrlo jednostavne naredbe. Naredbom „*composer global require laravel/installer*" instaliramo Laravel na svoje računalo, a naredbom "*laravel new naziv_projekta*" kreiramo novi projekt sa željenim nazivom projekta. Naredbe je potrebno upisati unutar naredbenog retka (cmd-a) vodeći računa o ispravnom pozicioniranju unutar odgovarajućeg direktorija.



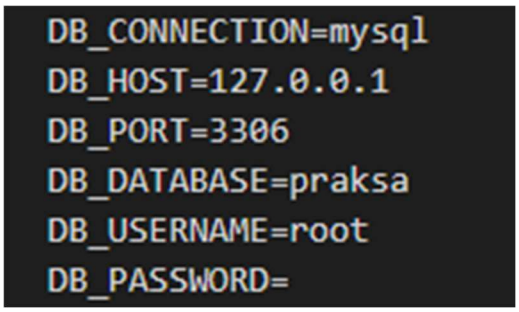
Slika 3.3. Naredbe za kreiranje projekta u Laravelu

3.5. Dodavanje React-a u projekt

Nakon stvaranja Laravel projekta bilo je potrebno dodati React razvojnu okolinu u projekt budući da osnovni Laravel projekt dolazi s Vue razvojnom okolinom. Prebacivanje s Vuea na React je učinjeno uz pomoć jedne naredbe „php artisan preset react".

3.6. .env datoteka

.env datoteka je posebna po tome što se u njoj nalaze osnovni parametri aplikacije. U toj datoteci smo unijeli naziv baze podataka, korisničko ime te lozinku kako bi povezali naš Laravel projekt s kreiranom bazom podataka.



Slika 3.4. Podaci za spajanje na bazu podataka

3.7. Provjera identiteta

Budući da aplikacija ima tri uloge korisnika (fakultet, tvrtka i student), bilo je nužno kreirati autentifikaciju korisnika. Budući da je korišten React razvojno okruženje nije se mogla koristiti naredba „php artisan make:auth“ jer zahtjeva dodatnu komunikaciju sa serverom. React omogućuje kreiranje jednostraničnih aplikacija (SPA - single page application) kod kojih se samo početna stranica učita sa servera i sve ostale se dinamički izmjenjuju preko te osnovne kako bi se izbjeglo višestruko učitavanje stranica sa servera[13].

Rješenje tog problema je bio „Laravel passport“ koji omogućuje autentifikaciju osnovanu na tokenima[14]. Registracija i prijava korisnika odvija se preko API poziva te ukoliko je uspješan poziv dobijemo token pomoću kojeg možemo ući u aplikaciju.

3.8. Migracije za bazu podataka

U projektu postoji 13 tablica, od kojih je pet dodano od strane „Laravel passporta“. Za svaku je tablicu napravljena migracija.

Users – sadrži informacije o korisnicima

Faculties – sadrži informacije o fakultetima

Companies – sadrži informacije o tvrtkama

Students – sadrži informacije o studentima

Practises – sadrži informacije o praksama

Reports – sadrži informacije o izvještajima s prakse

Tablice koje su dodane od strane „Laravel passporta“ sadrže podatke vezane za tokene pomoću kojih se korisnici prijavljuju, vrstu prijave i slične informacije.

3.9. Modeli

Projekt sadrži šest modela koji nam omogućuju komunikaciju s bazom podataka. Svaki model je zadužen za jednu tablicu. Modeli su:

- User
- Faculty
- Company
- Student
- Practise
- Report

Slika 3.5. prikazuje model „Company“. Model se sastoji od polja s nazivima stupaca u tablici, te od dvije funkcije u kojima smo uz pomoć Eloquenta napravili relacije između tablica u bazi podataka.

```php
class Company extends Model
{
    protected $fillable = [
        'name', 'email', 'address', 'phone', 'OIB', 'user_id'
    ];

    public function user(){
        return $this->belongsTo(User::class);
    }

    public function practise(){
        return $this->hasMany(Practise::class);
    }
}
```

Slika 3.5. Prikaz modela "Company"

3.10. Rute

Budući da je aplikacija zamišljena kao jednostranična (SPA) nisu bile dostupne klasične rute koje bi upisivali u web.php datoteku nego su korištene API rute koje se nalaze u api.php datoteci. Na slici 3.6. su prikazane sve rute u aplikaciji koje se pozivaju pri određenim API pozivima.

```
Route::middleware('auth:api')->get('/user', function (Request $request) {
    return $request->user();
});

Route::post('/login', 'AuthController@login');
Route::post('/register', 'AuthController@register');
Route::middleware('auth:api')->post('/logout', 'AuthController@logout');

Route::middleware('auth:api')->resource('profile', 'ProfileController');
Route::middleware('auth:api')->resource('students', 'StudentController');
Route::middleware('auth:api')->resource('practise', 'PractiseController');
Route::middleware('auth:api')->resource('reports', 'ReportController');
```

Slika 3.6. Prikaz ruta za upravljanje aplikacijom

3.11. Kontroleri

Aplikacija se sastoji od pet kontrolera:

- AuthController
- StudentController
- ProfileController
- PractiseController
- ReportController

3.11.1. AuthController

AuthController sadrži funkcije za registraciju, prijavu i odjavu korisnika iz aplikacije.

Slika 3.7. prikazuje isječak koda iz AuthController.php datoteke, točnije funkciju za registraciju korisnika. Na početku funkcija provjerava želi li se registrirati student ili netko drugi, te ovisno o tome uvjetu provodi određeni dio koda zadužen za provjeru unesenih vrijednosti. Zadnji dio programskog koda kreira korisnika u bazi podataka.

```
public function register(Request $request){
    if($request->role == 'Student'){
        $request->validate( [
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
            'password' => ['required', 'string', 'min:8'],
        ]);
    }
    else{
        $request->validate( [
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
            'password' => ['required', 'string', 'min:8', 'confirmed'],
        ]);
    }

    return User::create([
        'name' => $request->name,
        'email' => $request->email,
        'password' => Hash::make($request->password),
        'role' => $request->role,
    ]);
}
```

Slika 3.7. Prikaz funkcije za registraciju korisnika

3.11.2. StudentController

StudentController sadrži osnovne funkcije koje su kreirane dodatnim parametrom „resource“ pri pozivanju naredbe za kreiranje kontrolera.

Slika 3.8. prikazuje funkciju za kreiranje studenta u „students“ tablici gdje se nalaze dodatne informacije o studentima.

```php
public function store(Request $request)
{

    $faculty = auth()->user()->name;

    $student = new Student;

    $student->indexNumber = $request->input('indexNumber', '');
    $student->faculty = $faculty;
    $student->study = $request->input('study', '');
    $student->course = $request->input('course', '');
    $student->yearsOfStudy = $request->input('yearsOfStudy', '');
    $student->OIB = $request->input('OIB', '');
    $student->user_id = $request->input('user_id');

    if($student->save()){
        return $student;
    }
}
```

Slika 3.8. Prikaz funkcije za kreiranje studenta

3.11.3. ProfileController

Kontroler ProfileController je kreiran naredbom za kreiranje kontrolera „php artisan make:controller“ uz dodatan parametan „resource“.

Slika 3.9. prikazuje funkciju za stvaranje korisničkog profila nakon što korisnik prvi puta uđe u aplikaciju. Funkcija vraća popis polja koja se moraju ispuniti kako bi profil bio kreiran.

```php
public function create()
{
    $role = auth()->user()->role;
    if($role == 'Tvrtka'){
        $table = 'companies';
        $columns = Schema::getColumnListing($table);
    }
    else if($role == 'Fakultet'){
        $table = 'faculties';
        $columns = Schema::getColumnListing($table);
    }
    else if($role == 'Student'){
        $table = 'students';
        $columns = Schema::getColumnListing($table);
    }
    return $columns;
}
```

Slika 3.9. Prikaz funkcije kojom se dohvaćaju informacija o korisniku

3.11.4. PractiseController

Kontroler je također kreiran uz pomoć osnovne naredbe za kreiranje kontrolera uz dodatan parametar „resource“.

Slika 3.10. opisuje dohvaćanje informacija o pojedinoj praksi s relacijom prema tablici „company“ i tablici „user“.

```php
public function show($id)
{
    $practise = Practise::with('company.user')->where('id', $id)->get();
    return $practise;
}
```

Slika 3.10. Prikaz funkcije kojom prikazujemo praksu

3.11.5. ReportController

Slika 3.11. prikazuje funkciju koja se poziva nakon što student stisne gumb za spremanje izvještaja sa stručne prakse. Prvo se provjerava postoji li dokument u poslanom zahtjevu, nakon toga postoji li već dokument s tim imenom u bazi podataka, ako postoji onda se dokument zamjenjuje. Ukoliko dokument ne postoji, stvara se novi. Novi dokument nosi naziv studenta koji ga je kreirao popraćan brojem kako ne bi u slučaju istoimenih studenata došlo do gubitka izvještaja.

```
public function store(Request $request)
{
    if($request->hasFile('file')) {
            $user = auth()->user();
            if($request->fileName){
                $fileNameToStore = $request->fileName;
            }
            else{
                $userName = $user->name;
                $userName = preg_replace('/\s/', '', $userName);
                $extension = $request->file('file')->getClientOriginalExtension();
                $fileNameToStore = $userName.'_'.time().'.'.$extension;
            }
            $path = $request->file('file')->storeAs('public/report', $fileNameToStore);
            $report = Report::where('student_id', $user->id)->first();
            if($report == []){
                $report = new Report;
                $report->practise_id = $request->practise_id;
                $report->student_id = $user->id;
                $report->grade = '';
                $report->comment = '';
                $report->facultyGrade = '';
                $report->facultyComment = '';
            }
                $report->file = $fileNameToStore;
                $report->save();
                return $report;
    }
    return 'No File';
}
```

Slika 3.11. Prikaz funkcije kojom spremamo studentov izvještaj s prakse

3.12. Pogledi

Za prikaz su korištene komponente koje su glavna karakteristika Reacta. Komponente su dio koda namijenjen višekratnoj upotrebi. Mogu primati neki ulaz, te na osnovu njega vratiti React element koji opisuje kako korisničko sučelje treba izgledati. Komponenta je klasa ili funkcija pisana u Javascriptu. Ukoliko je riječ o klasi, onda mora sadržavati funkciju „render“ koja je zadužena za vraćanje povratne vrijednosti koja je u najvećem broju slučaja React element.

Slika 3.12. prikazuje „Home“ komponentu koja vraća div oznaku kao povratnu vrijednosti. Unutar te oznake vidimo pozivanje druge komponente koja se naziva „Navigation“.

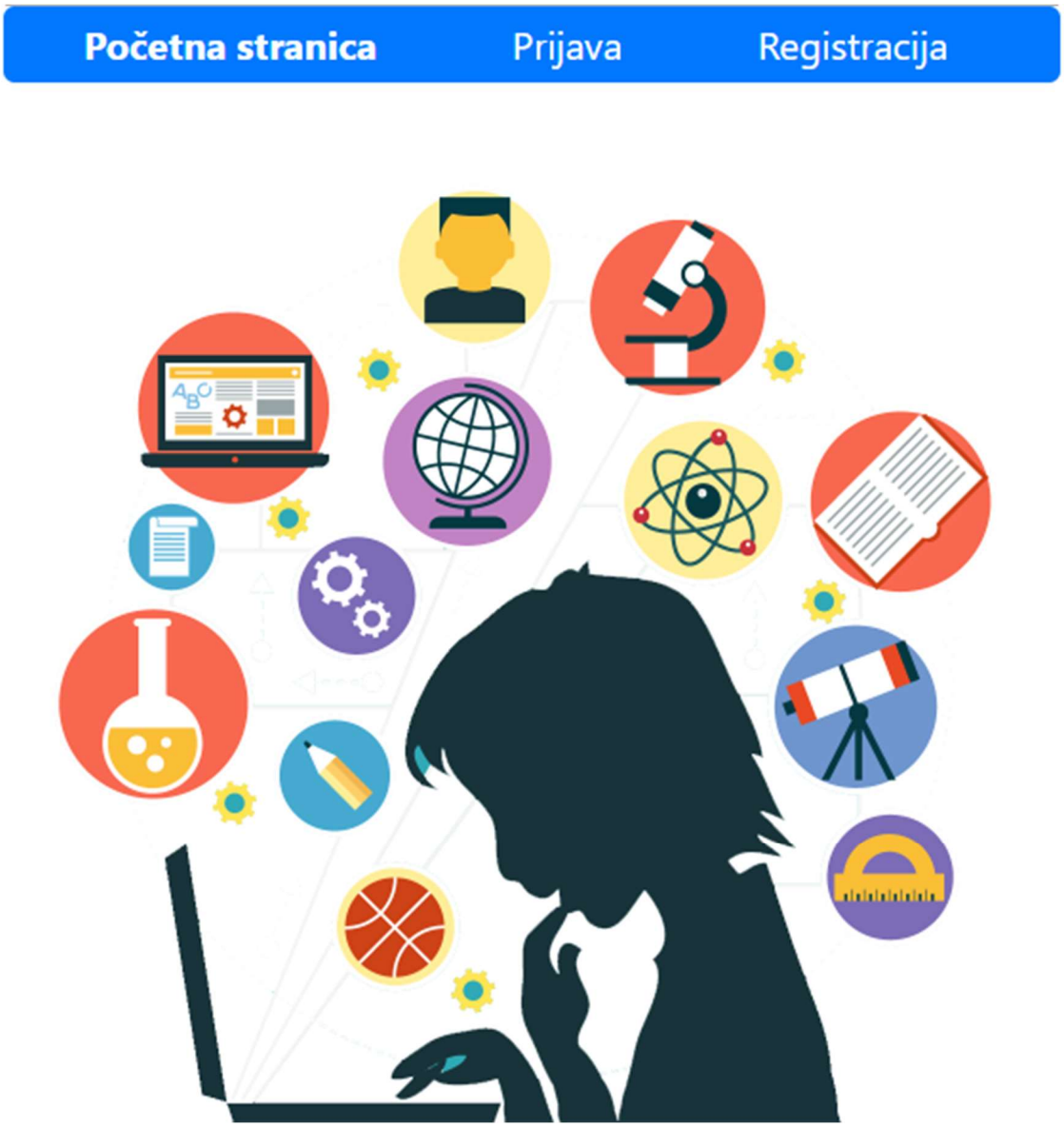
```
import React, { Component } from 'react';
import Navigation from './Navigation';

class Home extends Component {
    render() {
        return (
            <div className='hole-page'>
                <div className="container">
                    <Navigation/>
                    <div className="logo">
                        <img src='/images/logo.png'/>
                    </div>
                </div>
            </div>
        );
    }
}
export default Home;
```

Slika 3.12. Prikaz "Home" komponente

4. IZGLED APLIKACIJE

4.1. Početna stranica

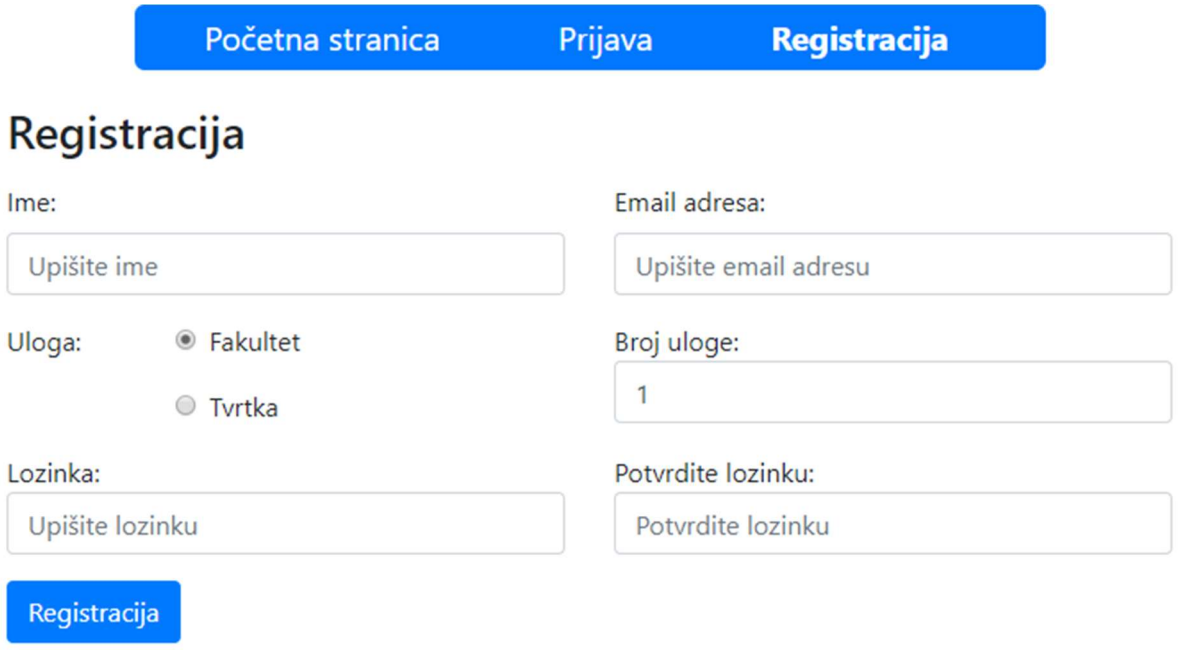
Početna stranica, koja je prikazana na slici 4.1., sadrži navigacijsku traku i sliku. Početna stranica je vidljiva samo korisnicima koji nisu prijavljeni.



Slika 4.1. Početna stranica aplikacije

4.2. Registracija

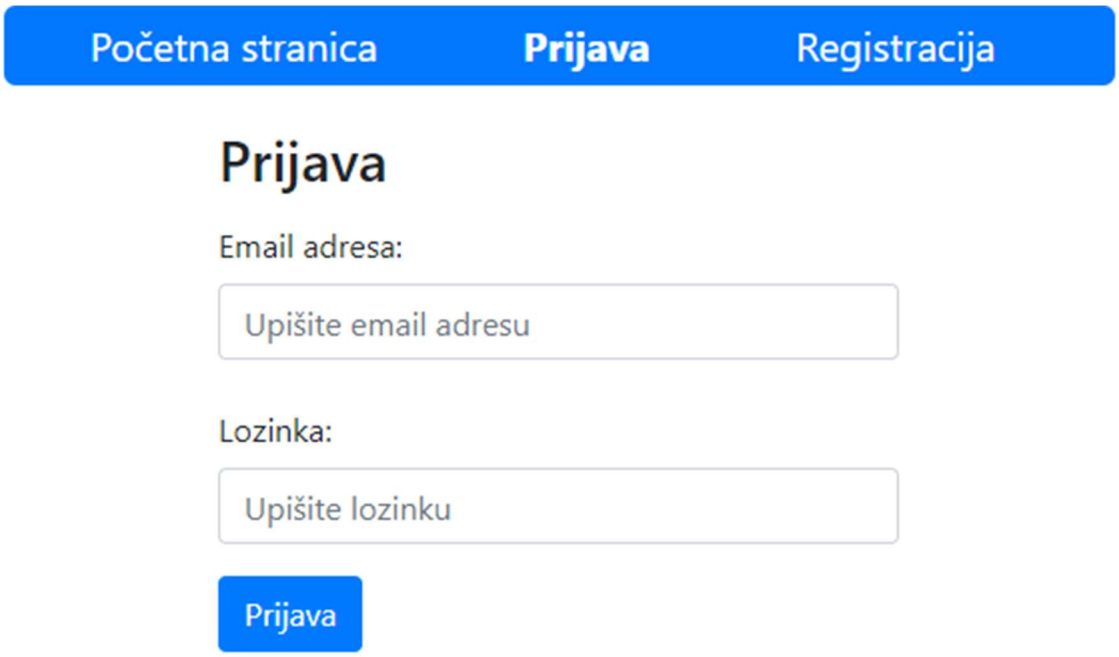
Registracija je omogućena samo tvrtkama i fakultetima, kako što je i na slici 4.2 prikazano. Odabir uloge korisnika je popraćen brojem uloge kako bi se spriječile potencijalne prijevare pri registraciji.



Slika 4.2. Prikaz obrasca za registraciju korisnika

4.3. Prijava

Nakon registracije, korisnik se preusmjerava na stranicu za prijavu. Stranica za prijavu sadrži obrazac, kao što prikazuje slika 4.3.

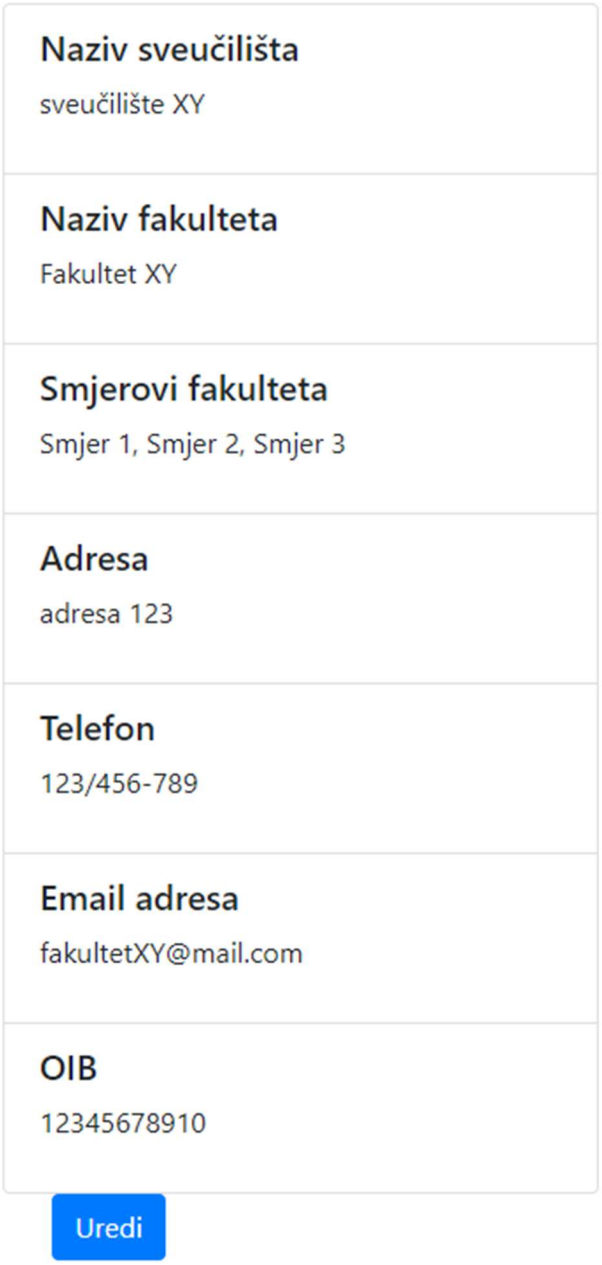


Slika 4.3. Prikaz obrasca za prijavu korisnika

4.4. Profil

Nakon uspješne prijave korisniku se prikazuje stranica s korisničkim profilom. Ukoliko se korisnik prvi put prijavljuje u aplikaciju prikazat će mu se obrazac koji zahtjeva od njega dodatne informacije kako bi profil bio potpun.

Slika 4.4. Obrazac za dodatne informacije o fakultetu

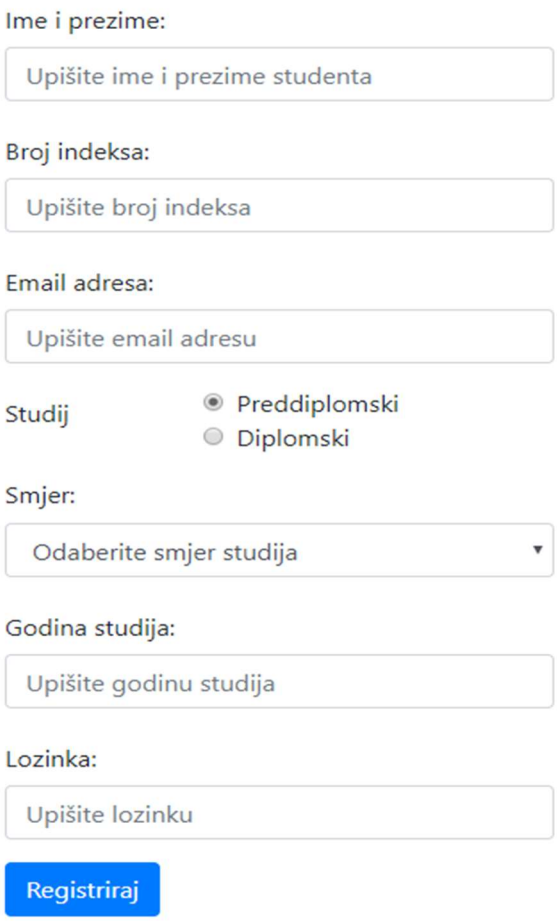


Slika 4.5. Prikaz korisničkog profila fakulteta

4.5. Dodavanje studenta

Klikom na „studenti“ u navigacijskoj traci odlazimo na stranicu sa studentima koja je vidljiva samo fakultetima. Stranica sa studentima prikazuje popis studenata tog fakulteta te im omogućuje dodavanje novih studenata. Informacije o studentima koji su na popisu, mogu se uređivati i detaljnije prikazati.

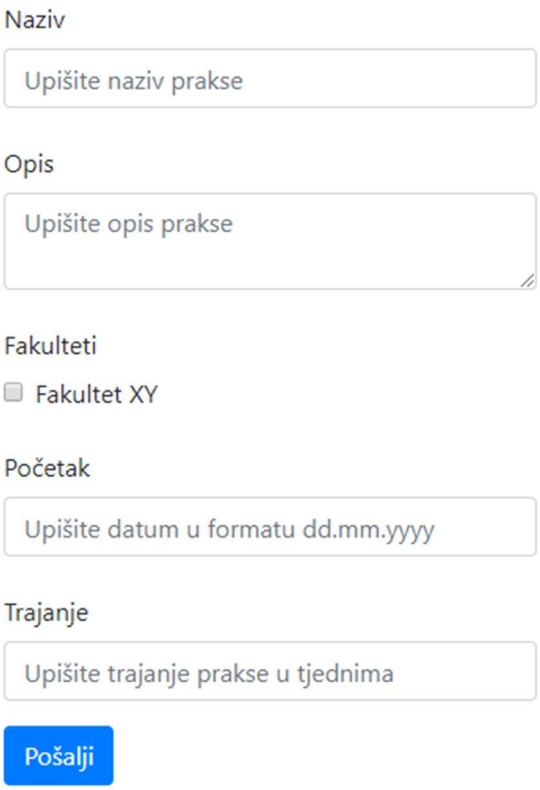
Slika 4.6. Prikaz popisa studenata



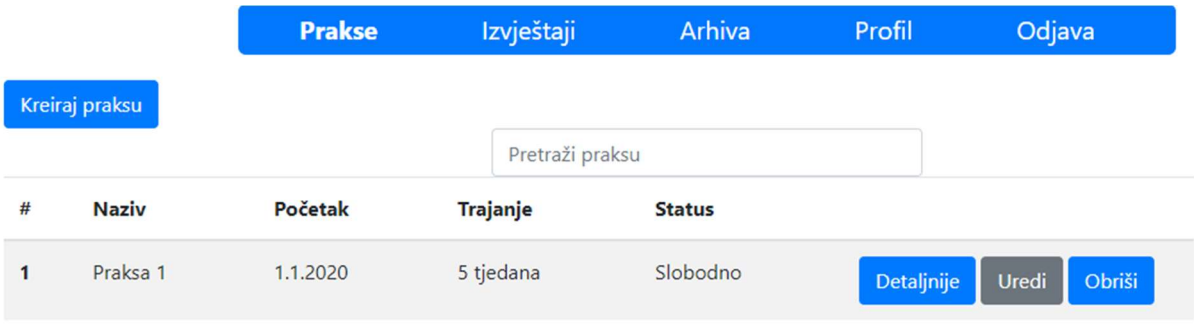
Slika 4.7. Obrazac za kreiranje novog studenta

4.6. Dodavanje prakse

Klikom na gumb prakse, prikazuje nam se popis praksi koje su dostupne. Ukoliko student ili fakultet otvori stranicu prakse, prikazat će se prakse dostupne za taj fakultet, odnosno fakultet koji student pohađa. Ukoliko tvrtka otvori stranicu s praksama prikazat će se njene prakse i mogućnost kreiranja novih. Kreiranje novih praksi prikazuje slika 4.8. Slika 4.9. prikazuje i tražilicu pomoću koje možemo filtrirati popis praksi.



Slika 4.8. Obrazac za kreiranje nove prakse



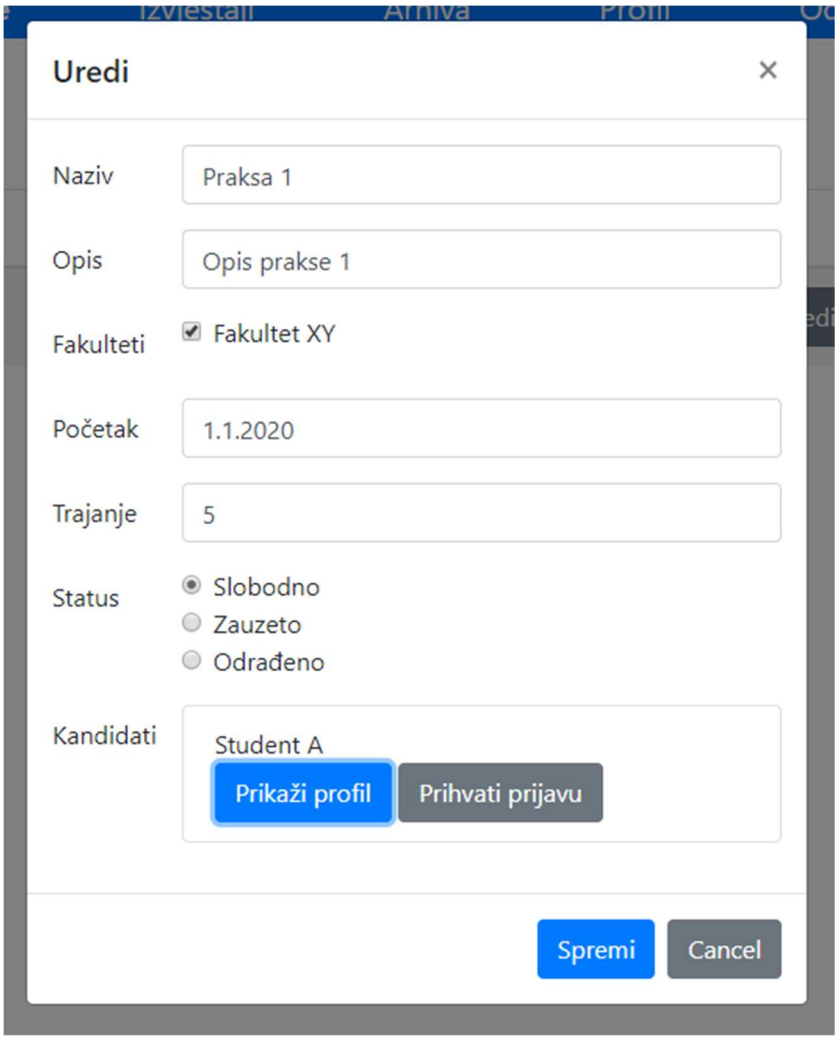
Slika 4.9. Prikaz popisa praksi

4.7. Odabir prakse

Student kada dobije svoje korisničke podatke od fakulteta, može se prijaviti u aplikaciju. Odlaskom na stranicu za prakse prikazuje mu se popis praksi na koje se može prijaviti klikom na gumb prijava. Pored gumba za prijavu, nalazi se gumb pomoću kojeg student može vidjeti detaljnije informaciji o praksi. Nakon prijave, gumb mijenja boju kako bi student uočio da se prijavio za tu praksu, te mu nudi mogućnost odjave.

Slika 4.10. Prikaz prijave studenta za praksu

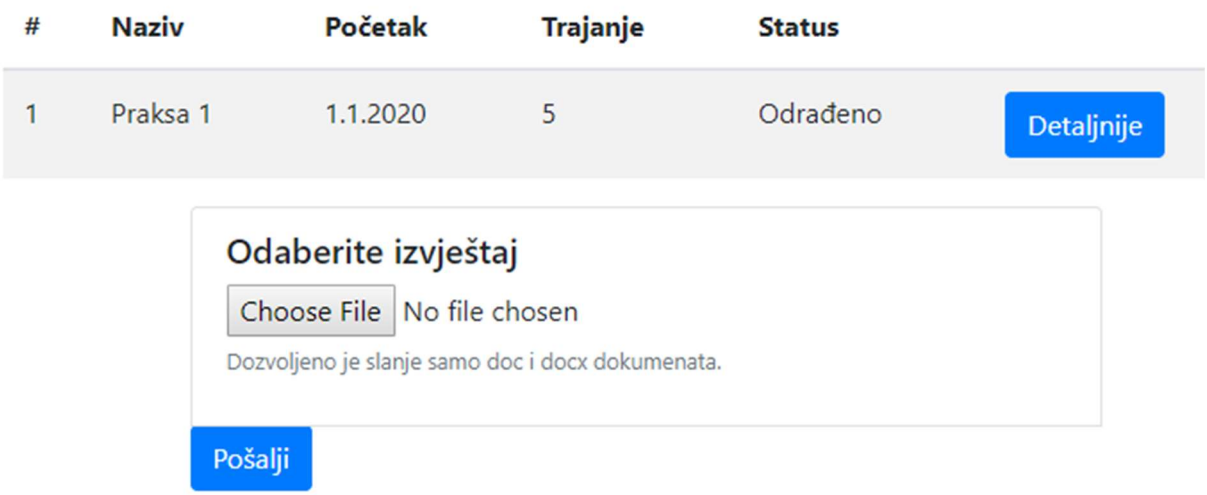
Nakon prijave studenta, tvrtka može vidjeti sve kandidate koji su se prijavili za pojedine prakse. Tvrtka može prikazati profil studenta ili mu odobriti prijavu, kao što je prikazano na slici 4.11.



Slika 4.11. Prikaz uređivanja prakse

4.8. Slanje izvještaja

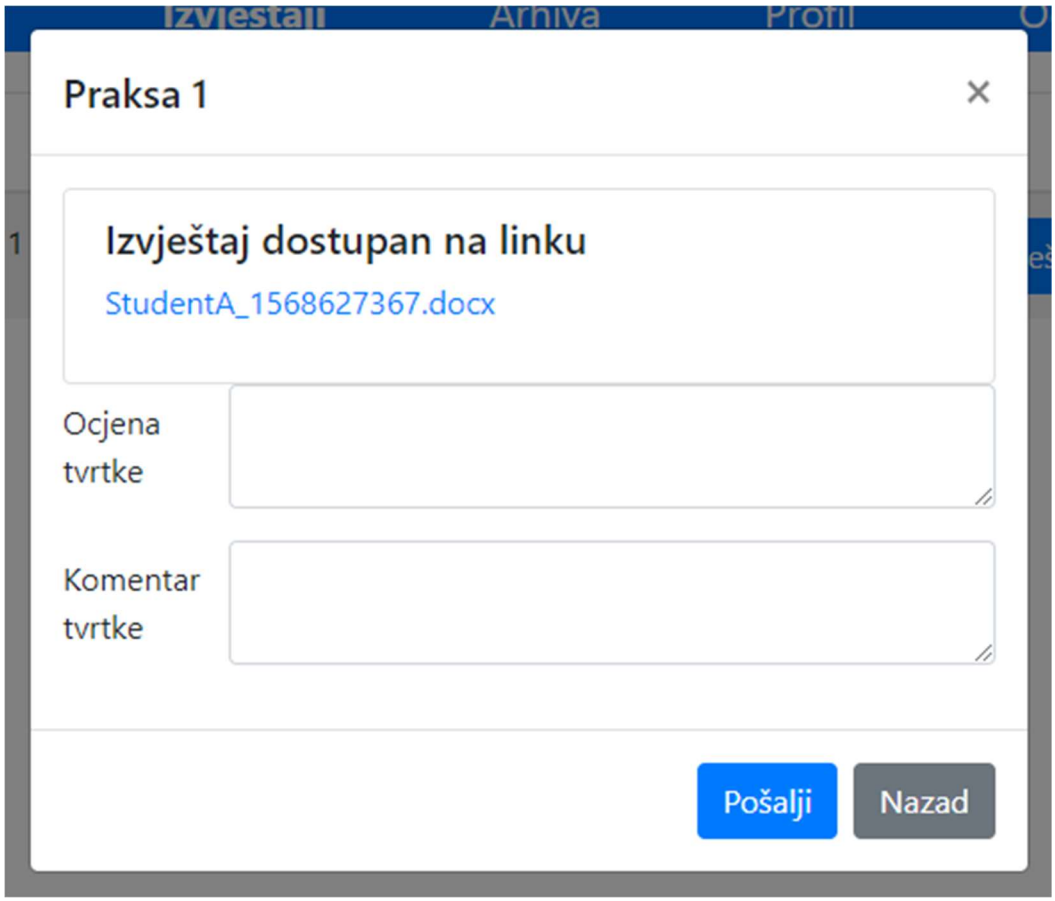
Kada student odradi praksu, na stranici „moja praksa“ prikazuje mu se praksa na koju je prijavljen te mu se nudi mogućnost slanja izvještaja sa stručne prakse.



Slika 4.12. Prikaz obrasca za dodavanja izvještaja

4.9. Ocjenjivanje izvještaja

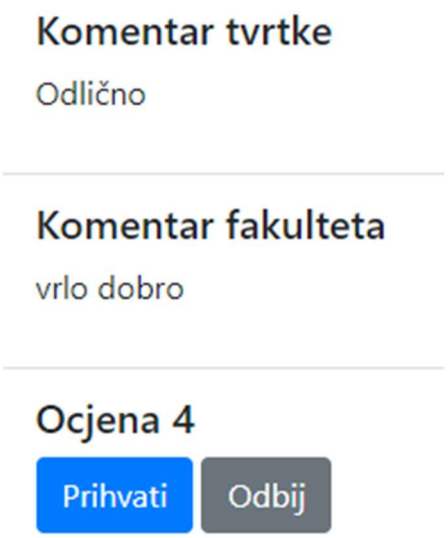
Nakon što student pošalje izvještaj sa stručne prakse, on prvo odlazi do tvrtke na ocjenjivanje. Odlaskom na stranicu „izvještaji“ prikazuje se popis izvještaja koji čekaju ocjenjivanje tvrtke. Prikazivanjem izvještaja otvara se prozor koji je prikazan na slici 4.13. Nakon ocjenjivanja izvještaj se prosljeđuje fakultetu koji ga mora također ocijeniti.



Slika 4.13. Prikaz obrasca za ocjenjivanje izvještaja

4.10. Ocjena

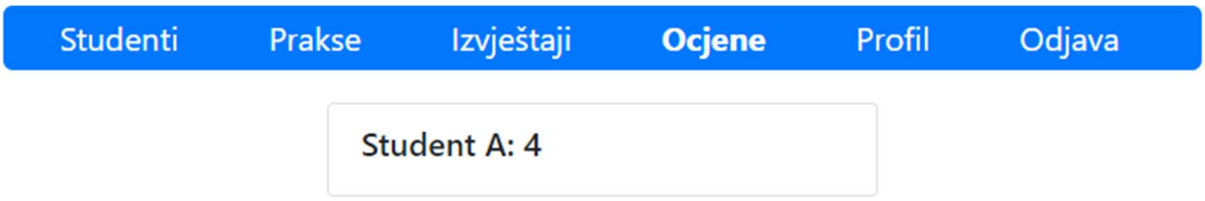
Nakon što fakultet ocjeni izvještaj, studentu se ponudi ocjena za njegovo obavljanje stručne prakse. Student može tu ocjenu prihvatiti ili odbiti. Ukoliko student odbije ocjenu, mora ponovo poslati izvještaj te čekati ocjenu tvrtke i fakulteta. No, ukoliko student prihvati ocjenu, praksa se zaključava i njegova ocjena se sprema u bazu podataka.



Slika 4.14. Prikaz ponude ocjene studentu

4.11. Prikaz ocjena

Zadnja stavka aplikacije je prikaz ocjena koja je omogućena samo fakultetima. Odlaskom na tu stranicu, fakultetima se ispisuju svi studenti koji imaju ocjenjenu stručnu praksu.



Slika 4.15. Prikaz popisa studenta s ocjenjenim praksama

5. ZAKLJUČAK

Cilj ovog rada bila je praktična izrada web aplikacije za stručne prakse koristeći neko razvojno okruženje. Za izradu web aplikacije korišten je Laravel za backend i React za frontend.

Aplikacija je izgrađena za tri različite uloge gdje korisnici, ovisno o svojoj ulozi, imaju određene mogućnosti i funkcionalnosti stavljene na raspolaganje. Izradom web aplikacije nastojala su se pokazati teorijska znanja o programiranju web aplikacije koja su opisana u prvim dijelovima rada, a praktično pokazana u drugom dijelu objašnjavajući primjere programskog koda.

Aplikacija je potpuno funkcionalna, ali to ne znači da nema mjesta za nadogradnju i unaprjeđenje. Moguća su brojna poboljšanja kao što su dodavanja slike u profilu korisnika, pretraživanje praksi po kandidatima, boljeg dizajna i slično.

LITERATURA

[1] HTML, https://en.wikipedia.org/wiki/HTML [posjećeno lipanj 2019.]

[2] CSS, https://developer.mozilla.org/en-US/docs/Web/CSS [posjećeno lipanj 2019.]

[3] JavaScript, https://developer.mozilla.org/en-US/docs/Web/JavaScript [posjećeno lipanj 2019.]

[4] API, https://en.wikipedia.org/wiki/Application_programming_interface [posjećeno lipanj 2019.]

[5] React, https://reactjs.org/ [posjećeno lipanj 2019.]

[6] Bootstrap, https://getbootstrap.com/ [posjećeno lipanj 2019.]

[7] PHP, https://www.php.net/ [posjećeno lipanj 2019.]

[8] Laravel, https://laravel.com/ [posjećeno lipanj 2019.]

[9] MVC, https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller [posjećeno kolovoz 2019.]

[10] MySQL, https://www.mysql.com/ [posjećeno lipanj 2019.]

[11] XAMPP, https://www.apachefriends.org/index.html [posjećeno lipanj 2019.]

[12] Composer, https://getcomposer.org/ [posjećeno srpanj 2019.]

[13] SPA, https://en.wikipedia.org/wiki/Single-page_application [posjećeno srpanj 2019.]

[14] Laravel Passport, https://laravel.com/docs/5.8/passport [posjećeno kolovoz 2019.]

SAŽETAK

U radu je prikazana izrada web aplikacije u Laravel razvojnom okruženju uz pomoć React razvojnog okruženja. Prikazani su programski kodovi korišteni pri izradi popraćeni detaljnim objašnjenjima. Aplikacija je povezana s MySQL bazom podataka te je omogućena registracija i prijava korisnika. Osim korisničkih podataka, u bazi podataka se nalaze podaci o stručnim praksama, njihovim izvještajima, te stečenim ocjenama. Aplikacija je namijenjena studentima, fakultetima i tvrtkama s ciljem da olakša proces potražnje i odrađivanja stručnih praksi. Vrlo je dinamična i jednostavna za korištenje.

Ključne riječi: Laravel, React, stručna praksa, web aplikacija

ABSTRACT

Web application for application and recording of professional practices

This paper describes the creation of a web application in Laravel development environment with the help of React development environment. The programming codes used in the development are shown and followed by detailed explanations. The application is linked to the MySQL database and is enabled for user registration and login. In addition to user data, the database contains information on professional practices, their reports, and the grades obtained. The application is intended for students, faculties and companies with the aim of facilitating the process of seeking and developing professional practices. It is very dynamic and easy to use.

Key words: Laravel, React, professional practice, web application

ŽIVOTOPIS

Antonio Došlić rođen je 31.7.1995. godine u Osijeku. Živi u Kuševcu na adresi Ljudevita Gaja 11. Osnovnoškolsko obrazovanje započinje 2002. godine u OŠ Josipa Antuna Ćolnića u Đakovu. Nakon osnovnoškolskog obrazovanja 2010. godine upisuje se u Gimnaziju Antuna Gustava Matoša, smjer Prirodoslovna matematička gimnazija. Nakon srednjoškolskog obrazovanja upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija gdje 2017. godine stječe višu stručnu spremu sa zvanjem sveučilišnog prvostupnika inženjera računarstva. Trenutno je redovan student na diplomskom studiju Fakulteta elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, smjer programsko inženjerstvo. Posjeduje znanje u govoru i pismu engleskog jezika te vozačku dozvolu B kategorije.

_____

Potpis: Antonio Došlić

PRILOZI

Na CD-u se nalazi diplomski rad u .docx i .pdf formatu te programski kodovi web aplikacije.