Web aplikacija za razmjenu usluga

Madžarac, Luka

Undergraduate thesis / Završni rad

2023

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:200:831190

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-26**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Stručni studij

WEB APLIKACIJA ZA RAZMJENU USLUGA

Završni rad

Luka Madžarac

Osijek, 2023. godina

Sadržaj

1. UVOD **1**

1.1 Zadatak završnog rada **2**

2. PREGLED PODRUČJA TEME **3**

3. PRIKAZ RADA APLIKACIJE **6**

3.1 Sučelje početne stranice **6**

3.2 Sučelje stranice detalja usluge **9**

3.3 Sučelje profilne stranice **10**

3.4 Sučelje stranice za prijavu **12**

3.5 Sučelje stranice za registraciju **12**

3.6 Korisničke uloge na web aplikaciji **14**

4. POSTUPAK IZRADE APLIKACIJE **18**

4.1 *Symfony* programski okvir **18**

4.2 Baza podataka i administratorska nadzorna ploča **19**

4.3 *Backend* razvoj naslovne stranice **23**

4.4 *Backend* razvoj profilne stranice **28**

4.5 *Backend* razvoj stranice za prijavu i registraciju **33**

5. ZAKLJUČAK 38

LITERATURA 40

SAŽETAK 41

ABSTRACT 42

PRILOZI 43

1. UVOD

Tema ovog završnog rada je web aplikacija za razmjenu usluga. Zadatak i ideja ove web aplikacije je pružanje mogućnosti ljudima određenih vještina objavljivanje i oglašavanje istih putem web portala jednostavnog sučelja kojeg bi mogli koristiti pripadnici raznih dobnih grupa. Osim objavljivanja oglasa za pružanje vlastitih usluga, web aplikacija također implementira mogućnost stvaranja oglasa za usluge koje vlasnik oglasa traži. Korisniku ove web aplikacije se pruža mogućnost izrade vlastitog korisničkog računa pomoću kojega bi mogao objavljivati oglase za usluge koje traži ili nudi.

Usluge koje se objavljuju na web portal mogu biti raznih vrsta te mogu biti jednokratne ili dugoročne po dogovoru davatelja usluge i primatelja usluge. Karakteristika ove web aplikacije koja se pokušava najviše istaknuti je upravo raznovrsnost usluga koje korisnici mogu oglašavati na stranici. Korisnik je u mogućnosti ispitati svoje vještine i sposobnosti kako bi potencijalno zamislio te ponudio uslugu drugim korisnicima na stranici. Kao što je opisano u nastavku, jednokratne specifične usluge vrlo često nemaju platformu na kojoj bi se oglašavale niti poticale. Zbog toga, ova web aplikacija pruža platformu ljudima vještim u izvođenju raznovrsnih usluga koje bi mogle biti od velike koristi drugim korisnicima. Tim ciljem je inspiriran i naziv ovog web portala: „Pametni Portal“. U ovom slučaju portal ne čini pametnim tehnologija kojom je implementirana web aplikacija nego upravo ljudi koji nude portalu svoja znanja i vještine.

U dobu velike razmjene informacija korisnici raznih web usluga često se ne mogu snaći u web aplikacijama kojima je cilj davanje velikog broja mogućnosti, informacija i ponuda. Korisnici imaju cilj dolaska na određene web stranice te je svrha tih web stranica upravo to da zadovolje uvjete i dovedu korisnika na taj cilj. Ovim se načinom implementiranja web aplikacije postižu pozitivna iskustva korisnika te osigurava korisnikovo vraćanje na aplikaciju ako za to bude imao potrebu.

Pomoću jednostavnog sučelja ova web aplikacija je implementirana na način da se vrlo brzo i jednostavno može doći do izmjene informacija i uspostavljanja kontakata među ljudima. Upravo to je i razlog dijeljenja oglasa na početnoj stranici web aplikacije na usluge koje se nude i koje se traže.

U realizaciji ove web aplikacije korišten je PHP programski jezik te *Symfony* okvir. *Symfony* je „skup višekratnih PHP komponenti i PHP okvir za izradu web aplikacija, API-ja, mikroservisa i web servisa”[1]. Upravo je taj bogati spektar komponenti i paketa korišten u izradi ove web aplikacije te u komunikaciji aplikacije s bazom podataka. Svojom jednostavnom strukturom, *Symfony* okvir nudi podjelu različitih dijelova web aplikacije te tako razdvaja *backend i frontend* dio razvojnog procesa.

Nakon pregleda područja teme u kojem će se razmotriti slična web rješenja, opis web aplikacije će se podijeliti u dva poglavlja. Prvo će se razmotriti *frontend* dio aplikacije koji podrazumijeva prikaz sučelja krajnjem korisniku te dostupne korisničke uloge i sve mogućnosti koje one daju korisnicima. *Backend* dio aplikacije je opisan u slijedećem poglavlju koje se odnosi na pregled programskog okvira, baze podataka i samog *kôda*. Prije zaključka, razmotriti će se moguće buduće nadogradnje web aplikacije koje bi bile korisne krajnjim korisnicima te ju dovele na novu razinu.

1.1 Zadatak završnog rada

Potrebno je napraviti web aplikaciju koja će omogućiti spajanje korisnika koji traže i nude usluge. Web aplikacija bi imala nekoliko korisničkih uloga (minimalno dvije). Potrebno je omogućiti detaljno pretraživanje i filtriranje usluga. Također, potrebno je implementirati nekakav način upravljanja novčanim transakcijama.

[1] *Symfony* - https://symfony.com/

2. PREGLED PODRUČJA TEME

U pregledu područja ove teme lako se definiraju web aplikacije sa sličnom idejom i namjenom. *Fiverr* je „internetska platforma koja kupcima i prodavačima omogućuje povezivanje kako bi mogli kupovati i prodavati digitalne usluge“[2]. Platforma je dobila naziv po početnoj cijeni usluge u iznosu od 5 dolara. Zasniva se na *freelance* načinu rada. *Freelancer* je „samozaposlena osoba, koja radi više poslova za različite poslodavce istovremeno“[3]. Takve osobe su obično plaćene po odrađenoj usluzi tj. završenom projektu. Vrlo često rade kratkoročno i uvijek su u fazi traženja slijedećega posla za klijenta koji treba njihove usluge. Neki od najčešćih poslova koje rade *freelanceri* su programiranje u raznim programskim jezicima, grafičko dizajniranje, fotografiranje i ostali poslovi slične prirode. *Fiverr* je platforma na kojoj davatelji usluga većinu svojih usluga pružaju digitalno. Neki od primjera usluga što se mogu naći na *Fiverr* web stranici su: snimanje glazbe i izrada originalne glazbe, snimanje zvukova i videa raznih tipova i glasovnu sinkronizaciju za različite svrhe. Platforma također sadrži oglase velikog broja instruktora raznih područja koji pružaju usluge učenja digitalnim putem. Najpopularnije kategorije na web stranici su digitalni marketing, grafički dizajn, pisanje i prijevod, video i animacija, glazba i zvuk te programiranje i tehnologija. Pri filtriranju usluga po različitim državama lako se mogu pronaći one usluge koje su na neki način povezani sa Hrvatskom. Oglasa na takve usluge nema u velikom broju te je većina ovako kategoriziranih usluga vezano za savjetovanje turista koji traže dobar plan putovanja po različitim dijelovima Hrvatske. Također može se naići na korisnike koji putem svojih objavljenih usluga nude snimanje kratkih videa na hrvatskom jeziku, učenje pričanja i prevođenje hrvatskog jezika. Sve u svemu *Fiverr* je odlična opcija za ljude koji traže usluge koje se mogu obaviti preko interneta. Ideja web aplikacije ovog završnog rada je pružanje mogućnosti obavljanja usluga ne samo preko interneta nego i uživo. Neki od primjera usluga koje bi se morale obaviti uživo su trener nekog sporta, krečenje, spremanje te čišćenje kuća.

[2] „*How does Fiverr work*“ (ddiy.co) - https://ddiy.co/how-does-fiverr-work
[3] „Što je *freelancer*“ (dir.hr) - https://dir.hr/sto-je-freelancer/

Toptal platforma je još jedan primjer web aplikacije sa sličnom primjenom. Ova platforma je „*freelance* web stranica koja povezuje visokokvalificirane *freelance* stručnjake s tvrtkama. *Toptal* ima širok raspon *freelancera,* od web programera i web dizajnera do financijskih konzultanata i voditelja proizvoda“[4]. Web aplikacija pruža platformu za objavljivanje oglasa za razne usluge u kategorijama kao što su razvojne usluge, dizajnerske usluge, usluge financijskih stručnjaka, usluge organizatora projekata te također posjeduje kategoriju koja skuplja razne stručnjake za poslovne potrebe klijenata. *Toptal* nudi brzo zapošljavanje u roku od čak 48 sati te pristup *freelancerima* na internetu. Neke od usluga što se mogu naći na *Toptal* web platformi su: izrada aplikacija za igre, dizajner proizvoda i programiranje u raznim jezicima. Osoba koja razmišlja o prijavljivanju na ovu platformu mora proći kroz proces testiranja jer ona prihvaća samo 3 posto najboljih. Razlog ovome je prikaz pouzdanosti poslodavcima kojim mogu biti uvjereni da primaju usluge vrhunskih kvaliteta. U isto vrijeme i *freelancerima* je zagarantirano da će usluge obavljati s najboljima u tom dijelu industrije. Najveća razlika prijašnjeg primjera web platforme *Fiverr* i *Toptal* platforme su upravo kvalifikacijski zahtjevi za prijavu na poziciju davatelja usluge gdje *Fiverr* ima puno manje zahtjeve, no puno više prijavljenih *freelancera* od kojih klijenti mogu izabirati. Kao i kod *Fiverr* web platforme, *Toptal* nudi *freelance* osobe čije se usluge obavljaju samo preko interneta.

Upwork web platforma je jedna od najpopularnijih u području ove teme. *Upwork* „povezuje tvrtke s neovisnim stručnjacima i agencijama diljem svijeta“[5]. Ova web aplikacija, za razliku od *Toptal* aplikacije, pruža jednostavno i besplatno pridruživanje, objavljivanje traženih usluga te automatsko pretraživanje sposobnih osoba koji tu uslugu mogu odraditi. Platforma pruža poslovanje s vrlo pouzdanim i iskusnim *freelancerima*. Najpopularnije kategorije su vrlo sličnog tipa prijašnje dvije platforme, a to su: razvojna rješenja i IT, AI usluge, kreativno dizajniranje, marketing i prodaje, pisanje i prijevod, administracijska i korisnička usluga, financije, inženjerstvo i arhitektura. Pouzdanost *Upwork* platforme osigurava priznanje Carol Taylora, direktora za iskustvo u sadržaju iz *Microsoft*-a: "Jedna od prednosti korištenja *freelancera* je brzo pronalaženje talenata s različitim vještinama kako se naše potrebe mijenjaju." Iako su prijašnji primjeri platforma vrlo pouzdani i korisni za obavljanje usluga preko interneta, dodavanje usluga koje bi se mogle provoditi i fizičkim putem bi bio jako dobar dodatak web aplikacijama.

[4] „*Best freelance websites*“ (*Hostinger*) - https://www.hostinger.com/tutorials/best-freelance-websites#2_Toptal
[5] *Upwork* - www.upwork.com

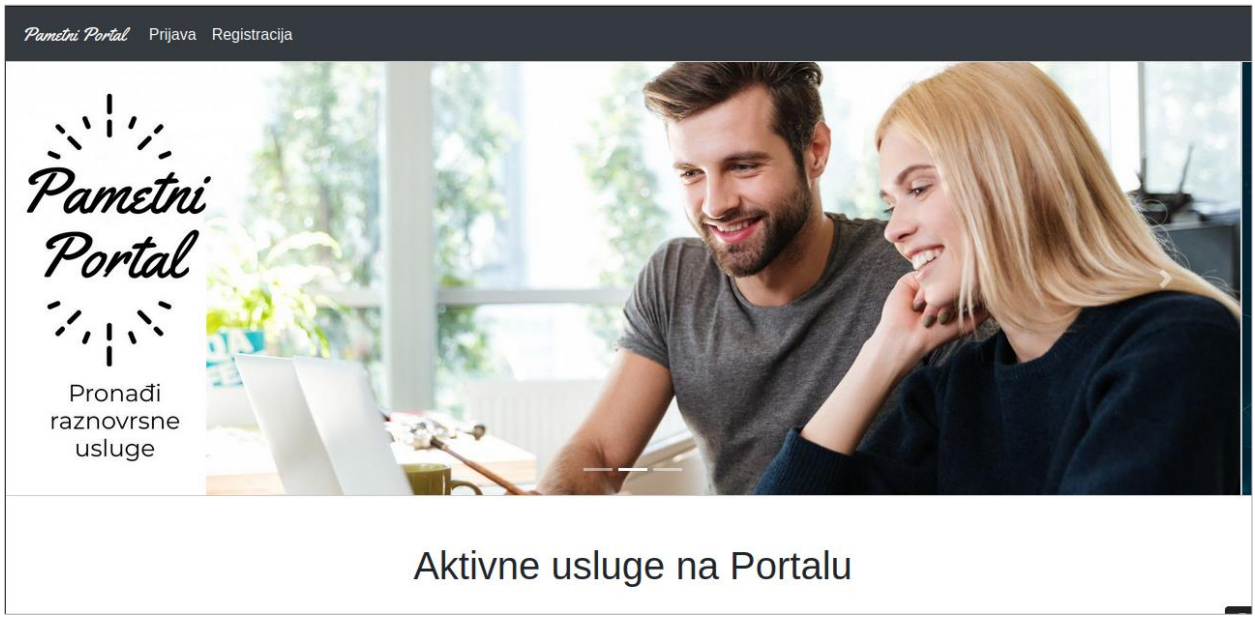
Važno je napomenuti da su stranice poput *Upwork* i *Toptal* platforme namijenjene stručnim korisnicima. Svaki korisnik koji naiđe na ove platforme bi u početku trebao biti upoznat s tehnikama pronalaska željenih informacija među velikom količinom podataka i opcija. „Pametni Portal“ implementira sličnu ideju spomenutih platformi uz dodatak obavljanja usluga uživo na području Republike Hrvatske. Ovo znatno povećava opseg mogućih usluga koje je se oglašavaju na web aplikaciji.

3. PRIKAZ RADA APLIKACIJE

U ovom je poglavlju prikazan izgled svakog dijela web aplikacije te opis korisničkih uloga. Prikazano je kako sučelje izgleda za krajnjeg korisnika te što pojedina korisnička uloga omogućuje korisniku.

3.1 Sučelje početne stranice

Kao što je prethodno spomenuto, cilj web aplikacije „Pametni portal“ je da bude jednostavna i efikasna za korištenje te se to najviše vidi na samom korisničkom sučelju aplikacije. Slika 3.1 prikazuje početnu stranicu portala te prvi uvid u portal kojeg dobiju posjetitelji.



Slika 3.1 Početna stranica web aplikacije

Iz slike 3.1 se može primijetiti logo samog portala, traka sa slikama te jednostavne ponuđene opcije na navigacijskoj traci. Na navigacijskoj traci s lijeve strane nalazi se lagano primjetljiv i jednostavan logo web aplikacije te opcije za prijavu i registraciju.

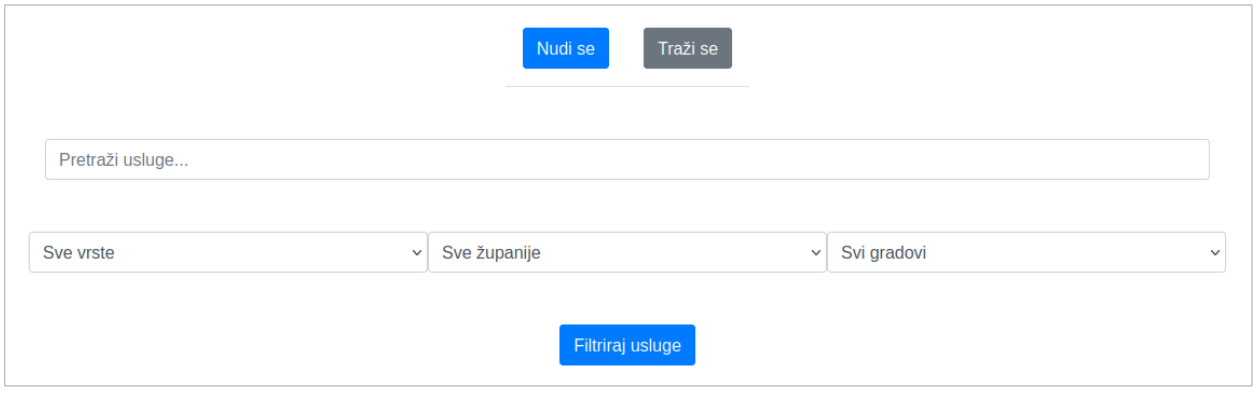
Ovaj pogled početne web stranice je iz perspektive neprijavljene osobe dok će prijavljena osoba na navigacijskoj traci imati opcije „Moj profil“ i „Odjava“.

Uzevši u obzir da bi portal posjećivali novi korisnici, glavni opis aplikacije je implementiran kako bi bio što kraći i jasniji. Na ovaj bi način posjetitelj dobio točnu informaciju o cilju i daljnjim iskustvima što može imati na web aplikaciji.

Ispod opisa portala nalazi se lista aktivnih usluga koje su korisnici portala objavili. Prije liste, kao što je prikazano na slici 3.2, nalazi se nekoliko opcija koje trebaju korisnicima olakšati pronalaženje oglasa. Prva opcija koju korisnik odabire je „Nudi se“ ili „Traži se“. Kao što ime predlaže, klikom na prvu opciju će se prikazati samo oglasi preko kojih korisnici nude svoje usluge, a klikom na drugu opciju će se prikazati samo oglasi preko kojih korisnici traže određene usluge.

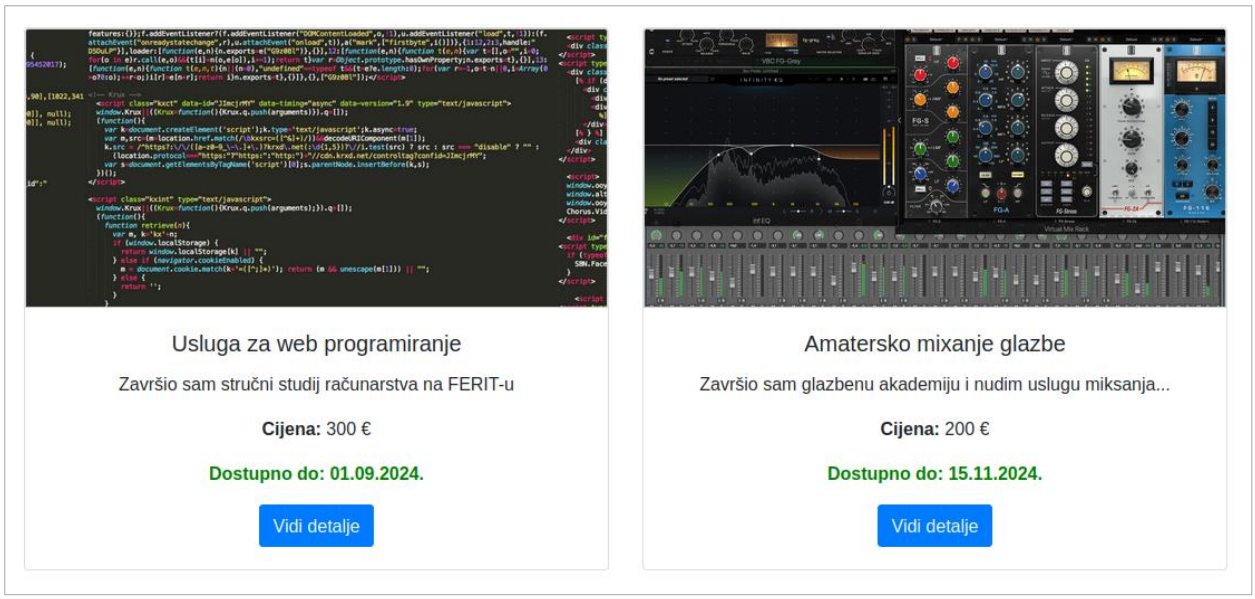
Ispod toga nalazi se jednostavna traka za pretraživanje po ključnoj riječi preko koje korisnik može vrlo lako skratiti listu oglasa na oglase koji sadrže određeni pojam ili frazu koju je unio u traku.

Nakon trake za pretraživanje nalaze se 3 padajuća izbornika koja dodatno filtriraju listu oglasa ukoliko je korisniku to potrebno. Dakle, ako korisnik želi pretražiti usluge po određenoj vrsti, otvoriti će prvi padajući izbornik, izabrati vrstu usluge te kliknuti na gumb „Filtriraj usluge“ nakon čega će se na stranici prikazati samo usluge koje pripadaju vrsti koju je korisnik odabrao. Korisnik također može nadalje filtrirati oglase po gradu ili županiji u kojoj je uslugu moguće odraditi.



Slika 3.2 Opcije pretraživanja i filtriranja na naslovnoj stranici

Slika 3.3 prikazuje aktivne usluge koje se nude i traže. Ista je funkcionalnost prikazana i prijavljenom i neprijavljenom korisniku te se pomoću jednostavnog klika na „Nudi se“ ili „Traži se“ na web stranici izmjenjuju oglasi odabranog cilja. Oglas svake pojedine usluge se sastoji od naslova usluge, kratkog opisa, cijene, statusa, roka te gumba koji vodi na stranicu s više detalja o samom oglasu.

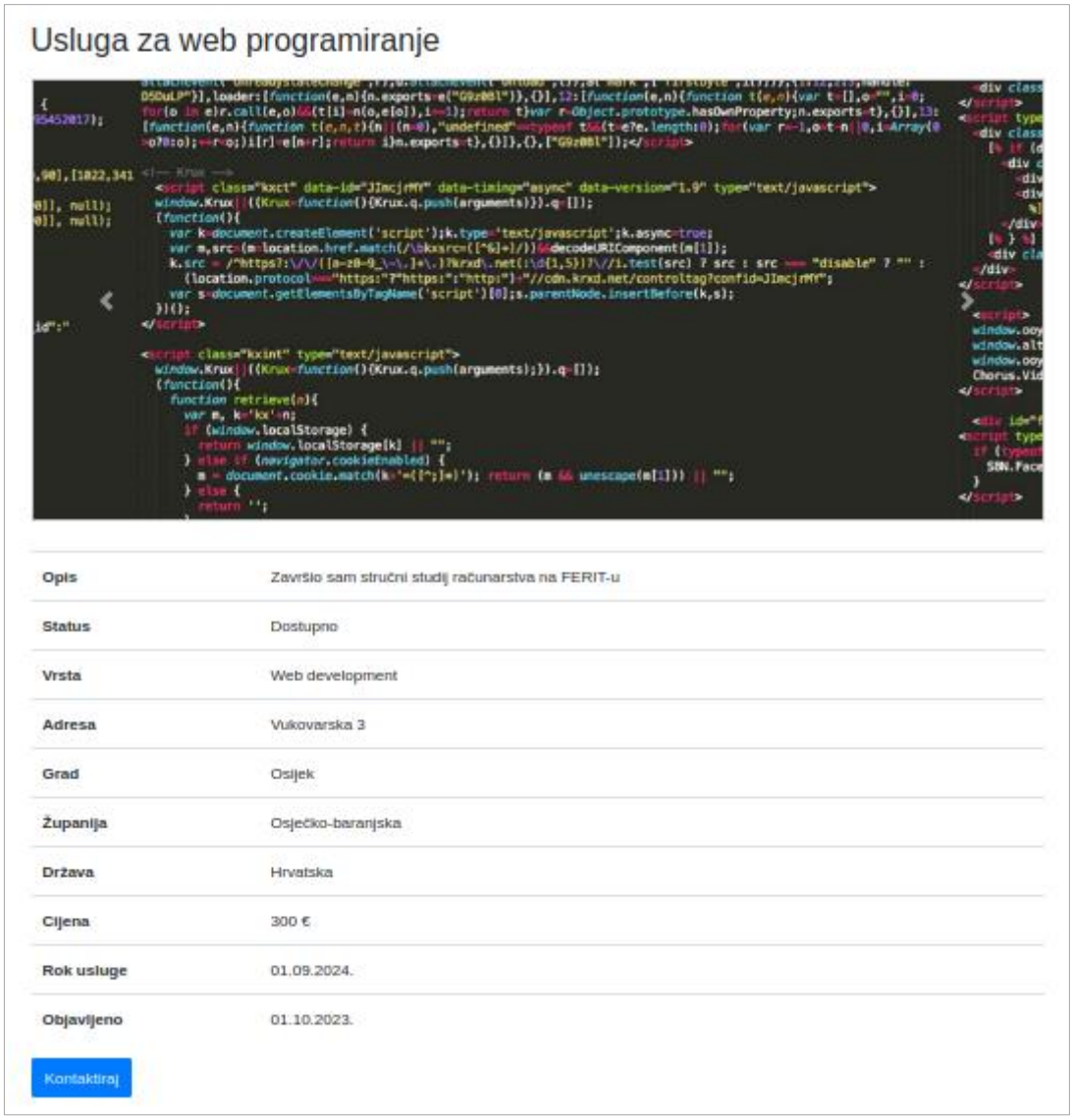


Slika 3.3 Pregled usluga na početnoj stranici web aplikacije

Naslov usluge ima ograničen broj slova i trebao bi biti što kraći te prenijeti točnu potrebu ili dobit usluge. U opis usluge oglašivač može napisati sve tehničke informacije o samoj usluzi kao i kvalifikacije osobe s kojom bi htio surađivati te informacije o cijeni i javljanju. Važno je napomenuti da će se na naslovnoj stranici prikazati samo prva rečenica opisa, a kada korisnik uđe u detalje oglasa moći će vidjeti puni opis. Nakon toga slijedi cijena usluge izražena u eurima i status tj. dostupnost usluge. Usluga može imati 3 statusa: dostupno, nedostupno i privremeno nedostupno. Na glavnoj stranici web portala će se prikazati samo usluge pod statusom „dostupno“ i „privremeno nedostupno“ dok će „nedostupne“ usluge moći vidjeti samo vlasnik usluge na vlastitom profilu.

3.2 Sučelje stranice detalja usluge

Na slici 3.4 moguće je vidjeti stranicu koja se prikaže kada korisnik klikne na gumb „Vidi detalje“. Prvi podatak je naziv usluge, a zatim slijedi traka sa svim objavljenim slikama ove usluge te opis koji je ovdje prikazan u potpunosti. Slijedeći podatak je status usluge koji informira korisnike o dostupnosti usluge. Nakon toga slijedi vrsta usluge koja opisuje u koju kategoriju spada oglašavana usluga. Na primjer, usluga instrukcije sviranja bubnjeva će biti u kategoriji za glazbene instrukcije. Slijede informacije za adresu, grad, županiju te državu usluge, a nakon toga dolazi cijena izražena u eurima.

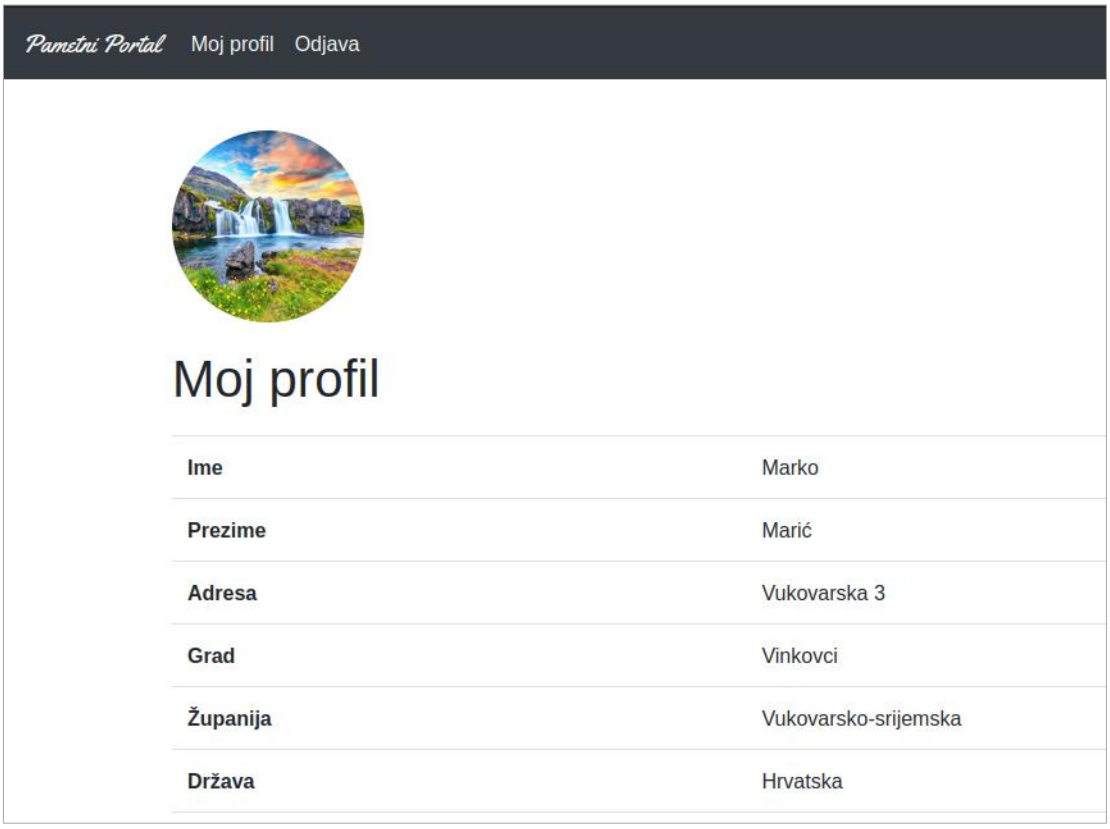


Slika 3.4 Stranica s detaljima usluge

Cijena na oglasu kod ponuđene usluge označava iznos koji davatelj usluge traži, a kod tražene usluge iznos koji bi tražitelj platio osobi za odrađivanje te usluge. Slijedeća dva podatka su datum objave i rok usluge. Na kraju se nalazi gumb „Kontaktiraj" koji će posjetitelja stranice preusmjeriti na profilnu stranicu osobe koja je objavila uslugu.

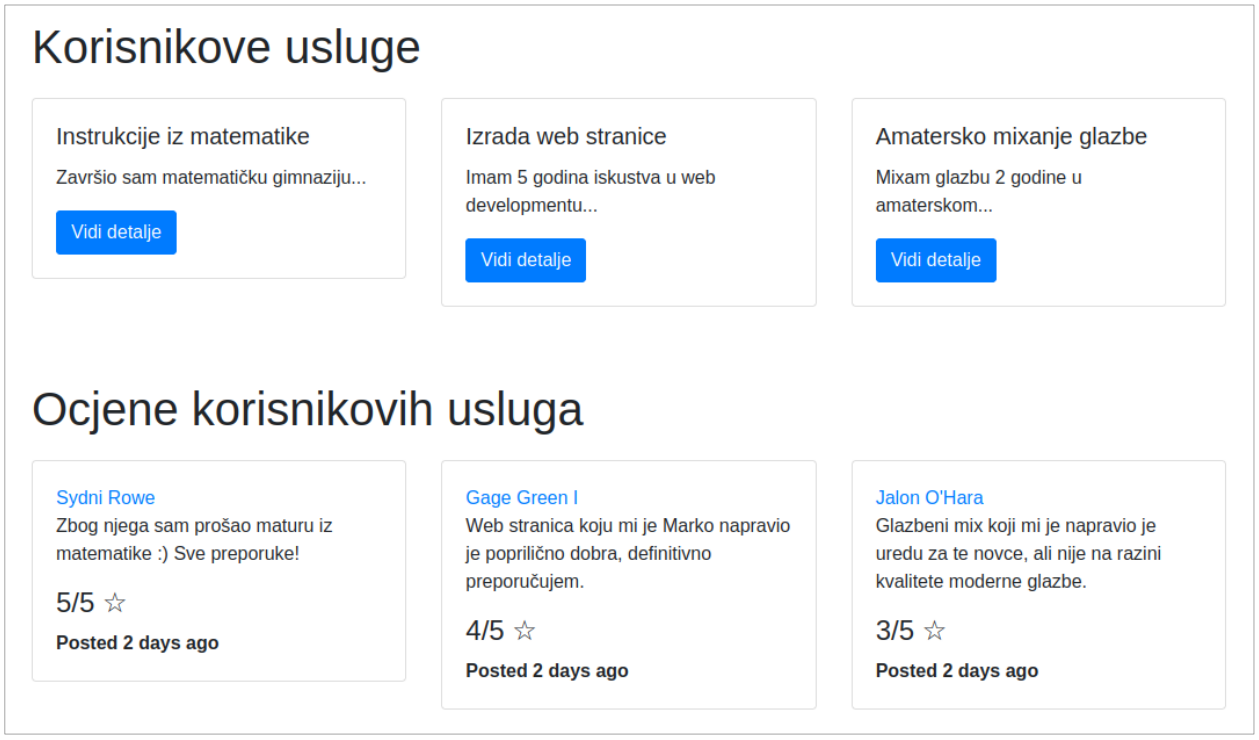
3.3 Sučelje profilne stranice

Na stranici profila korisnika, kao što je to prikazano na slici 3.5, se mogu vidjeti podaci o toj osobi. Ako osoba pregledava svoj profil, vidjeti će sve podatke koje je unijela prilikom registracije te će ih moći i urediti. Ako osoba pregledava tuđi profil, neće biti u mogućnosti vidjeti informacije kao što su adresa i ime osobe. Dakle, biti će prikazani samo podaci za kontakt i korisničko ime profila.



Slika 3.5 Prikaz dijela profila prijavljene osobe

Nadalje, profil osobe sadrži popis usluga koje je korisnik profila objavio te sve ocjene koje je korisnik zadobio od drugih korisnika kao što se vidi iz slike 3.6. Ocjene sadržavaju korisničko ime osobe koja je ostavila ocjenu pomoću kojeg se klikom može posjetiti njen profil. Nakon toga dolazi tekst ocjene u kojem ocjenjivač objašnjava razlog ocjene koja je objavljena, te ocjena koja može biti od 1 do 5 zvjezdica. Po tim ocjenama se računa prosječna ocjena profila koja se prikazuje pri samom vrhu profila.

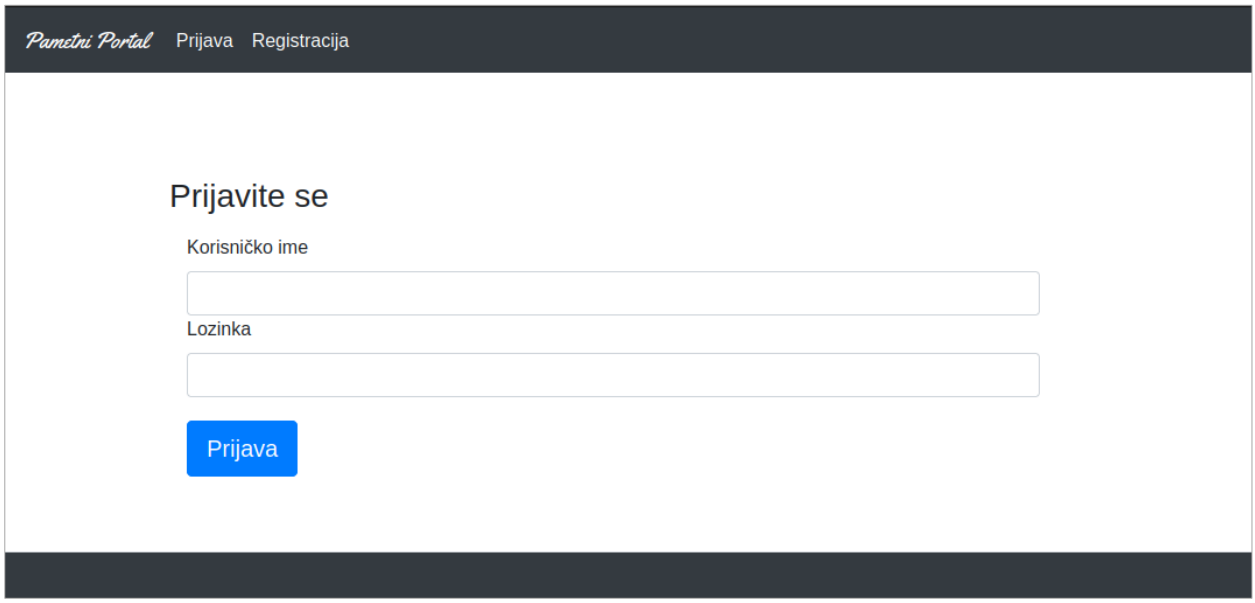


Slika 3.6 Lista usluga i ocjena korisnika profila

3.4 Sučelje stranice za prijavu

Slika 3.7 predstavlja stranicu za prijavu gdje se nalazi ista navigacijska traka kao na početnoj stranici. Klikom na logo portala korisnik se lako može vratiti na početnu stranicu web aplikacije.

Obrazac za prijavu se sastoji od tekstualnog polja za korisničko ime i tekstualnog polja za zaporku koje je korisnik odabrao pri registraciji na portal. Nakon uspješnog prijavljivanja korisnika će se uputiti na početnu stranicu web aplikacije te će joj se na navigacijskoj traci ponuditi opcije prelaska na profil i odjave.

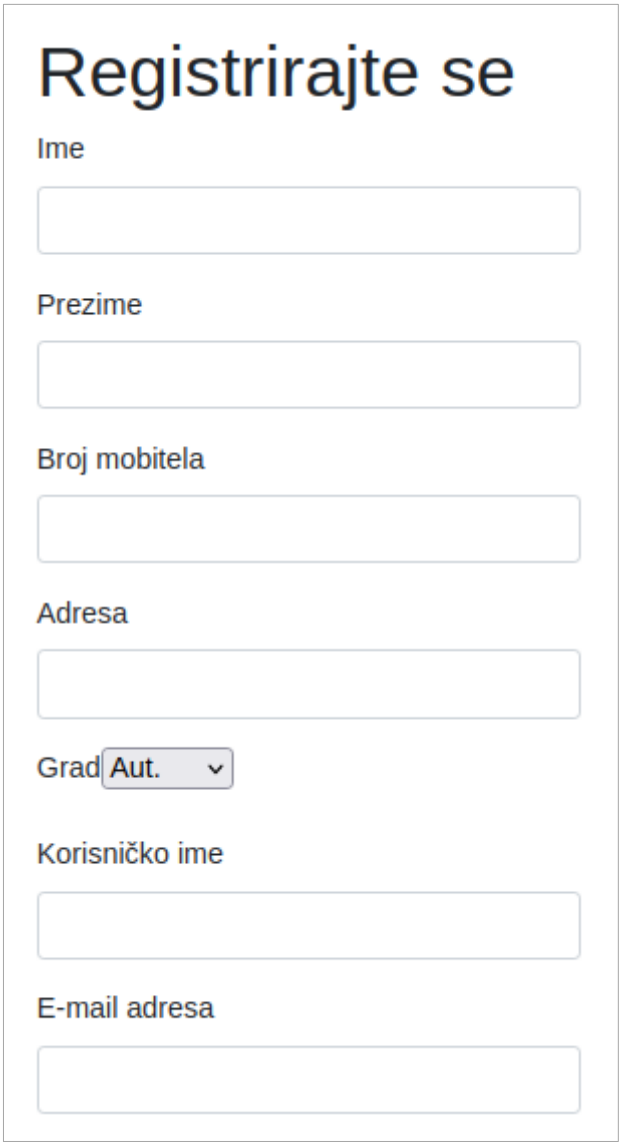


Slika 3.7 Prikaz stranice za prijavu

3.5 Sučelje stranice za registraciju

Kao što se može vidjeti na slici 3.8, registracijski obrazac pruža korisniku nekoliko polja koje treba ispuniti svojim podacima. Prvo i drugo polje predstavlja ime i prezime koje se neće prikazivati prilikom oglašavanja, dok treće polje predstavlja broj mobilnog telefona koji će biti prikazan svim posjetiteljima web aplikacije. Dalje slijedi adresa i grad prebivališta osobe, dok će se županija i država automatski spremiti. Nakon toga se prikazuje polje za proizvoljno korisničko ime koje će biti prikazano posjetiteljima stranice na oglasu korisnikovog računa.

Slijedeće tekstualno polje predstavlja proizvoljnu zaporku korisnika koja mora imati barem šest znakova te se nakon toga nalazi polje za odabir slike koja će služiti kao profilna slika. Na samom kraju stoji gumb kojim će korisnik potvrditi podatke koje je unio i registrirati se na web portal.



Slika 3.8 Prikaz dijela registracijskog obrasca stranice

3.6 Korisničke uloge na web aplikaciji

Korisničke uloge se mogu podijeliti na 5 vrsta ovisno o namjerama određenog korisnika. Važno je napomenuti da korisničke uloge mogu također biti i hibridne. Na primjer, prijavljeni korisnik može biti i davatelj i tražitelj usluga.

Način na koji su uloge napravljene u *backend* dijelu aplikacije prikazan je na slici 3.9. Kompletan programski *kôd* se može naći u prilogu P1. Svaki korisnik u bazi ima brojčanu vrijednost na stupcu „role". Pomoću te brojčane vrijednosti se saznaje koja sva dopuštenja korisnik ima, tj. koje sve uloge ima. Svako dopuštenje predstavljeno je jednim brojem koji je potencija broja 2. Budući da korisnik može imati više dopuštenja, brojevi svakog dopuštenja se zbrajaju i spremaju u bazu kao jedan broj. Kada je potrebno saznati koja dopuštenja određeni korisnik ima, pogleda se broj pod stupcem „*role*" i pomoću njega se saznaje točno koja dopuštenja treba dati korisniku. Način na koji to saznaje je pomoćnu binarne operacije kojom se dobiva binarni oblik broja te se gleda koje znamenke imaju vrijednost 1. Svaka znamenka predstavlja zasebno dopuštenje tako da aplikacija priznaje dopuštenje korisniku ako znamenka za to dopuštenje ima vrijednost 1, a odbija dati dopuštenje korisniku ako znamenka ima vrijednost 0.

Na primjer, ako korisnik ima ulogu pod brojem 5, nakon binarne pretvorbe aplikacija će dodijeliti korisniku uloge *ROLE_ADMIN* i *ROLE_USER* te uz to sve mogućnosti koje te uloge donose.

```php
public function getRoles()
{
    //1=> admin
    //2 => editor
    //4 => user
    //8 => tester


    $array = [1 => 'ROLE_ADMIN', 2 => 'ROLE_EDITOR', 4 => 'ROLE_USER', 8 =>
    'ROLE_ALLOWED_TO_SWITCH'];
    $number = $this->role;
    $allUserRoles = [];

    foreach ($array as $mask => $string) {
        if ($number & $mask) {
            array_push($allUserRoles, $string) . PHP_EOL;
        }
    }

    return $allUserRoles;
}
```

Slika 3.9 Način definiranja i dohvaćanja usluga

Posjetitelji koji su prvi put došli na web aplikaciju će podrazumijevano biti neprijavljeni korisnici. Neprijavljeni korisnici imaju mogućnost pristupa portalu te pregleda usluga i profila korisnika. Ova je korisnička uloga pogodna za osobe koje po prvi put isprobavaju web aplikaciju te potencijalno uzimanju svoju prvu uslugu kao neprijavljeni korisnici. Neprijavljeni korisnici lako mogu doći do svih informacija vezanih za pojedinu uslugu kao i kontakt informacija za javljanje osobi koja je objavila određeni oglas.

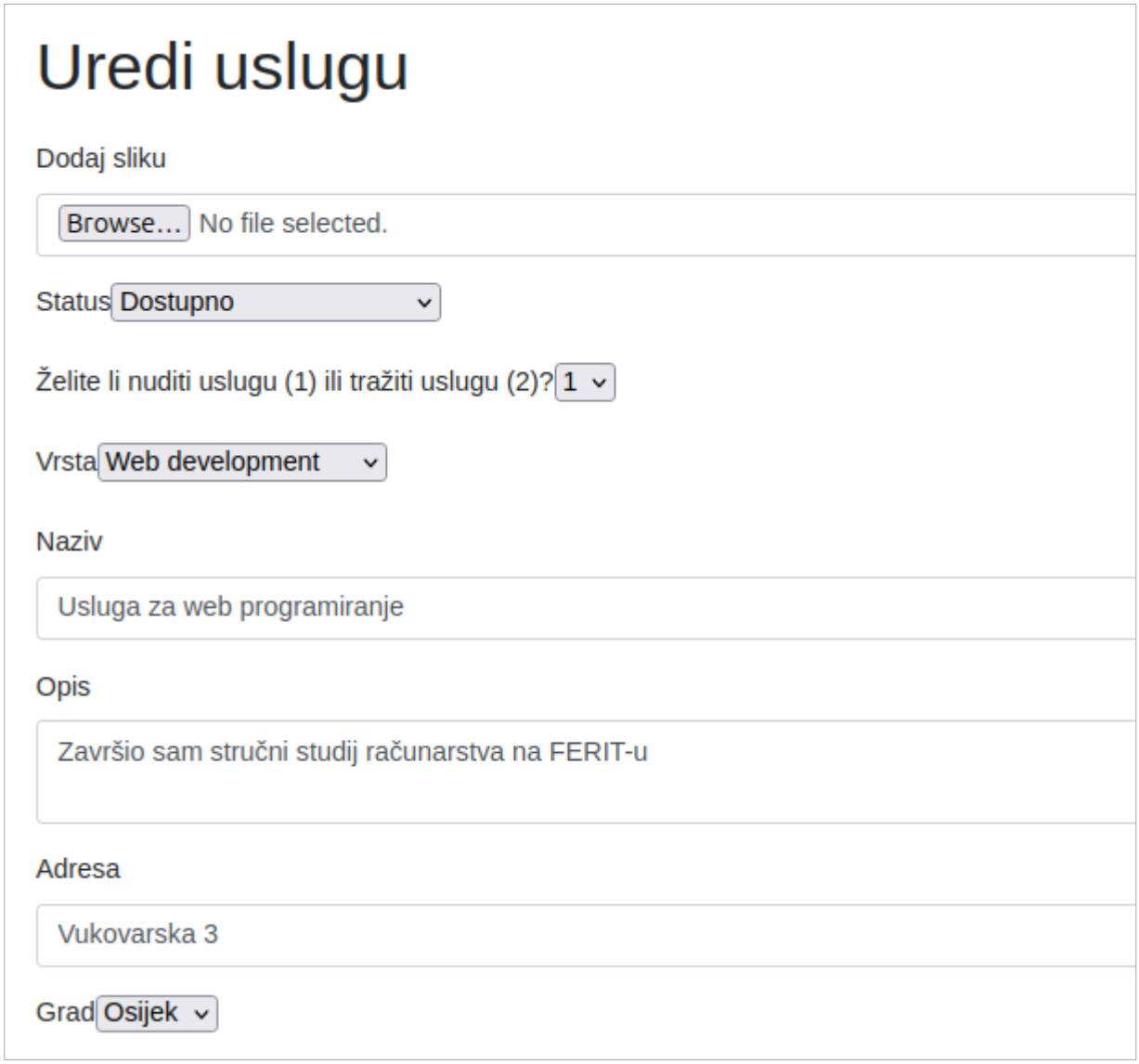
Međutim, neprijavljeni korisnici nemaju mogućnost objavljivanja usluga koje traže ili nude te spremanja vlastitih podataka na web aplikaciju. Dakle, ako neprijavljeni korisnik u bilo kojem trenutku korištenja web aplikacije poželi objaviti oglas zbog traženja ili nuđenja usluga, isto neće moći provesti jer mu za to nisu postavljena prava tj. uloge. U tom slučaju neprijavljeni korisnik u svakom trenutku može otići na registracijski dio web aplikacije i jednostavno izraditi svoj korisnički račun te će mu se automatski dodijeliti uloga prijavljenog korisnika. Navigacijska traka neprijavljenog korisnika izgleda kao na prethodno prikazanoj slici 3.1. Dakle neprijavljen korisnik nema niti jednu dodanu ulogu pa čak niti osnovnu ulogu *ROLE_USER* koja se daje korisniku kada napravi svoj korisnički račun.

Za razliku od neprijavljene osobe prijavljena osoba posjeduje mogućnost korištenja i uređivanja vlastitog korisničkog računa. U slučaju da je osoba izradila korisnički račun na web aplikaciji u svrhu oglašivanja vlastitih usluga osoba je slobodna objavljivati opise i uvjete svojih usluga neograničenim brojem oglasa koje može napraviti. Davatelj usluga ima opciju promjene unesenih podataka pomoću kojih će ih zainteresirani korisnici kontaktirati te opciju jednostavnog i brzog dodavanja oglasa za svoje usluge. Svoje usluge odmah može i pronaći na glavnoj stranici web aplikacije kada ih odluči objaviti. Profil korisnika izgleda kao na prethodno prikazanoj slici 3.5. Prijavljeni korisnik će za vrijednost u polju „*role“* u bazi podataka imati vrijednost 4 tj. biti će mu dodijeljene sve mogućnosti koje *ROLE_USER* podrazumijeva.

U slučaju da je korisnik izradio korisnički račun u svrhu objavljivanja oglasa, dodaje mu se uloga koja joj daje mogućnost objavljivanja neograničenog broja oglasa za uslugu koju traži. Dakle, kao i kod davatelja usluga, tražitelj usluga ima mogućnost postavljanja i uređivanja vlastitih podataka na svom profilu preko kojih će ga kontaktirati osoba koja nudi traženu uslugu. Dodavanje oglasa ima sličan obrazac i za traženje i za nuđenje usluga tako da prijavljeni korisnici koji traže usluge lako mogu opisati i objaviti usluge koje nude. Tražitelj usluga će imati ista dopuštenja kao i davatelj usluga. Vrijednost u polju „*role*” u bazi podataka biti će 4 tj. korisnik će imati sve mogućnosti koje *ROLE_USER* podrazumijeva.

Korisnik koji ima ulogu uređivača, kao što to prikazuje slika 3.10, posjeduje mogućnosti prijave i spremanje vlastitih podataka u bazu podataka te uređivanje oglasa koje su objavili korisnici portala. Dakle uređivač po potrebi može izmijeniti određene informacije na oglasima, no nema dopuštenje brisati oglase, izmjenjivati ili brisati korisnike sa portala te druge slične radnje.

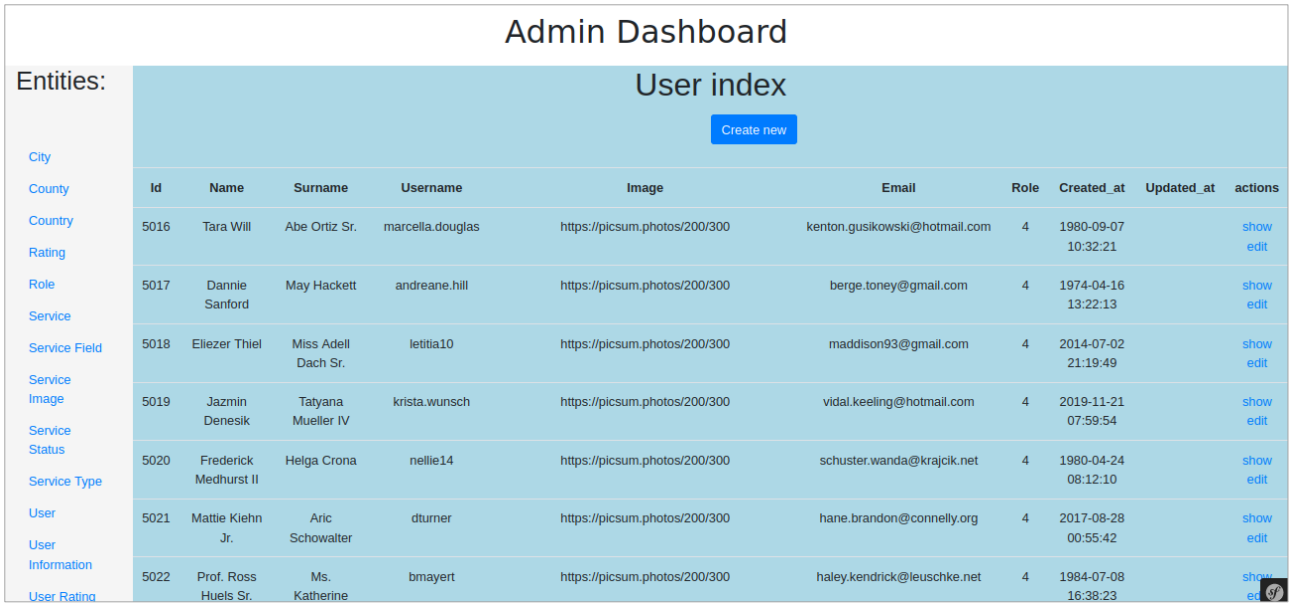
Vrijednost na polju „*role*" u bazi podataka za uređivača biti će 6 tj. korisnik će imati sve mogućnosti koje *ROLE_USER* te *ROLE_EDITOR* podrazumijevaju.



Slika 3.10 Sučelje za uređivanje usluge

Korisnik koji ima ulogu administratora posjeduje sve mogućnosti koje su navedene u prošlim ulogama. Dakle administrator ima svoj korisnički profil na kojem može mijenjati podatke i dodavati oglase te uređivati bilo koji oglas s portala. Dodatno, administrator ima pristup administratorskoj nadzornoj ploči, kao što se može vidjeti na slici 3.11, preko koje može izmjenjivati sve tablice iz baze podataka.

Uz uloge *ROLE_USER*, *ROLE_EDITOR* i *ROLE_ADMIN*, administrator također ima *ROLE_ALLOWED_TO_SWITCH* koji mu daje posebno dopuštenje da se prijavi kao bilo koji korisnik iz baze podataka i testira web aplikaciju.



Slika 3.11 Prikaz administratorske nadzorne ploče

4. POSTUPAK IZRADE APLIKACIJE

U ovom je poglavlju detaljnije opisan programski okvir web aplikacije, baza podataka te *backend* razvoj web aplikacije. Opis programskog okvira daje uvid u svaki funkcionalni dio aplikacije dok opis *backend* razvoja daje detaljniji prikaz istih funkcionalnosti.

4.1 *Symfony* programski okvir

Funkcionalna pozadina web aplikacije je napisana na PHP programskom jeziku pod okvirom *Symfony*-a. Svaki programski *kôd* prikazan na slikama u ovom poglavlju je moguće naći u prilogu P1 koji sadrži kompletan projekt u *Symfony* okviru.

Kao što je rečeno u uvodu, *Symfony* je PHP okvir koji sadrži veliku količinu korisnih paketa koji web aplikaciji pružaju jednostavnost i pouzdanost. Velika količina alata koji su dostupni u ovom okviru omogućuju korisniku da izbjegne pisanje *kôda* za svaku kompliciraniju radnju koju želi napraviti. Paketi koji su sadržani u *Symfony* okviru su raznih vrsta kao što su sigurnosni paketi, paketi za uređivanje podataka u HTML *kôdu*, paketi za automatsko generiranje velike količine podataka te paketi koji olakšavaju stvaranje entiteta i upravljača. Ovakvi paketi su korišteni za postizanje ciljeva web aplikacije te će naknadno biti dodatno objašnjeni. *Symfony* okvir razdvaja logiku *kôda* od prikaza *kôda* te tako znatno olakšava pisanje i snalaženje u *kôdu*. Važno je napomenuti 3 dijela preko kojih *Symfony* okvir implementira web aplikaciju.

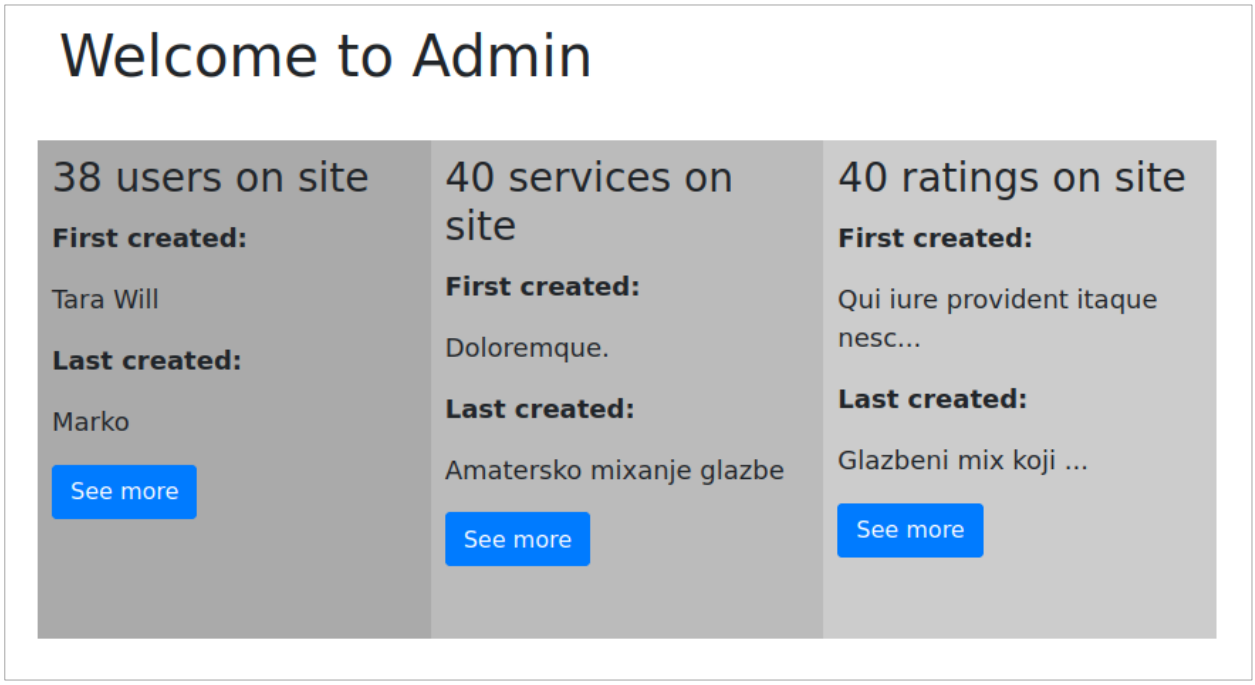
Upravljači (*controllers* na engleskom) su dijelovi *Symfony* okvira kojima je zadaća odgovaranje na korisničke zahtjeve. Upravljač na izlazu uvijek daje odgovor jer mu je glavna svrha da za dobro definirani zahtjev daje točan i potpun odgovor.

Šablone (*templates* na engleskom) su dijelovi *Symfony* okvira koji sadrže HTML *kôd* i kojim je zadaća vizualni prikaz podataka korisnicima u sučelju koje je pogodno i lako za koristiti. Dakle šablonu poziva upravljač nakon što dobije zahtjev za pristupom na određeni dio stranice.

Entiteti (*entities na engleskom*) su dijelovi *Symfony* okvira koji predstavljaju tablicu u bazi podataka i sve parametre koje tablica sadrži. Entiteti također sadržavaju funkcije kojima se obavlja određena logika potrebna za postavljanje i dohvaćanje parametara iz tablice. Na primjer, jedan entitet je „korisnik” koji kao parametre ima varijable „ime”, „prezime” i „adresa”.

4.2 Baza podataka i administratorska nadzorna ploča

Baza podataka koja se koristila za ovu web aplikaciju je MYSQL, a za izvršavanje operacija nad bazom se koristio naredbeni redak s MYSQL nadogradnjom. Vizualni prikaz baze podataka se najbolje vidi na administratorskoj nadzornoj ploči preko koje administrator ima uvid u sve tablice baze podataka.



Slika 4.1 Izgled naslovne stranice administratorske nadzorne ploče

Kada administrator pristupi početnoj stranici nadzorne ploče, aplikacija prikaže nekoliko informacija o podacima iz baze kao što je to vidljivo na slici 4.1. Administratoru se daje na uvid koliko korisnika je trenutno registrirano na stranici, koliko usluga su korisnici ukupno objavili na stranici te koliko su puta korisnici stranice ocijenjeni. Uz to su prikazane i informacije o tome koja je prva napravljena instanca određenog entiteta te koja je zadnja napravljena instanca. Nakon toga slijede gumbovi pomoću kojih administrator može lako doći na nadzornu ploču za odgovarajući entitet.

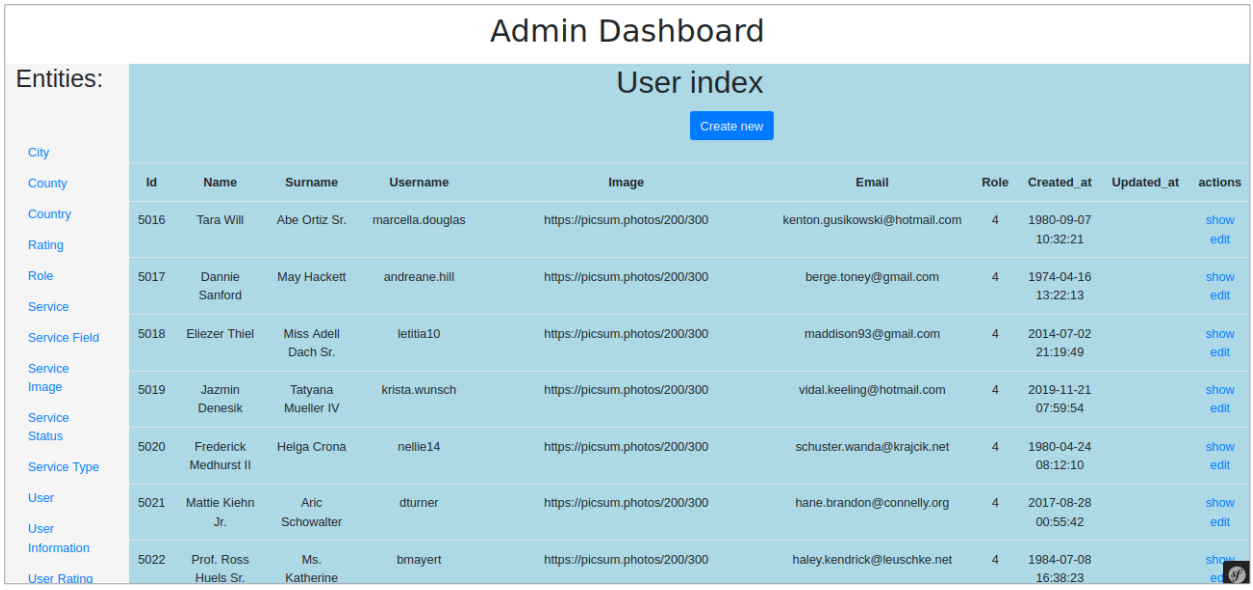
Osim toga, na ovoj administratorskoj početnoj stranici nalazi se i jednostavna navigacijska traka koja ima opciju „*Admin dashboard*“ koja, nakon što administrator klikne na nju, vodi na nadzornu ploču svih entiteta iz baze te se prikazuje slična tablica kao što je to bilo prikazano na slici 3.11.

Uz lijevi rub nadzorne ploče nalazi se lista entiteta koji postoje u bazi podataka te se administratoru omogućuje prikaz svih instanci tih entiteta nakon što on klikne na njih.

Za izradu baze podataka ove web aplikacije je bilo korišteno 13 entiteta od kojih svaki ispunjava određenu svrhu. Tri najvažnija entiteta na aplikaciji su entitet korisnika, usluga i ocjena.

Kao što se može vidjeti na slici 4.2, entitet „*User*“ ima parametre za identifikacijsku oznaku, ime, prezime, sliku, email, ulogu, datum stvaranja i datum ažuriranja. Najprije je važno napomenuti da ovo nisu svi podaci koje korisnik unese u registracijski obrazac. Nakon što korisnik ispuni registracijski obrazac navedeni podaci se spremaju u „*User*“ entitet, a ostatak podataka, koji predstavljaju detaljnije podatke o korisniku, se spremaju u drugi entitet zvan „*UserInformation*“ koji je relacijski povezan sa svojim pripadajućim korisnikom u „*User*“ entitetu. Spomenute detaljnije informacije o korisniku uključuju stvari kao što su adresa, grad, županija i broj telefona koji će biti kasnije spomenuti.

Važno je napomenuti da je pomoću parametra „*role*“, kao što je prije objašnjeno, moguće dobiti točna dopuštenja koje korisnik ima prilikom korištenja web portala.

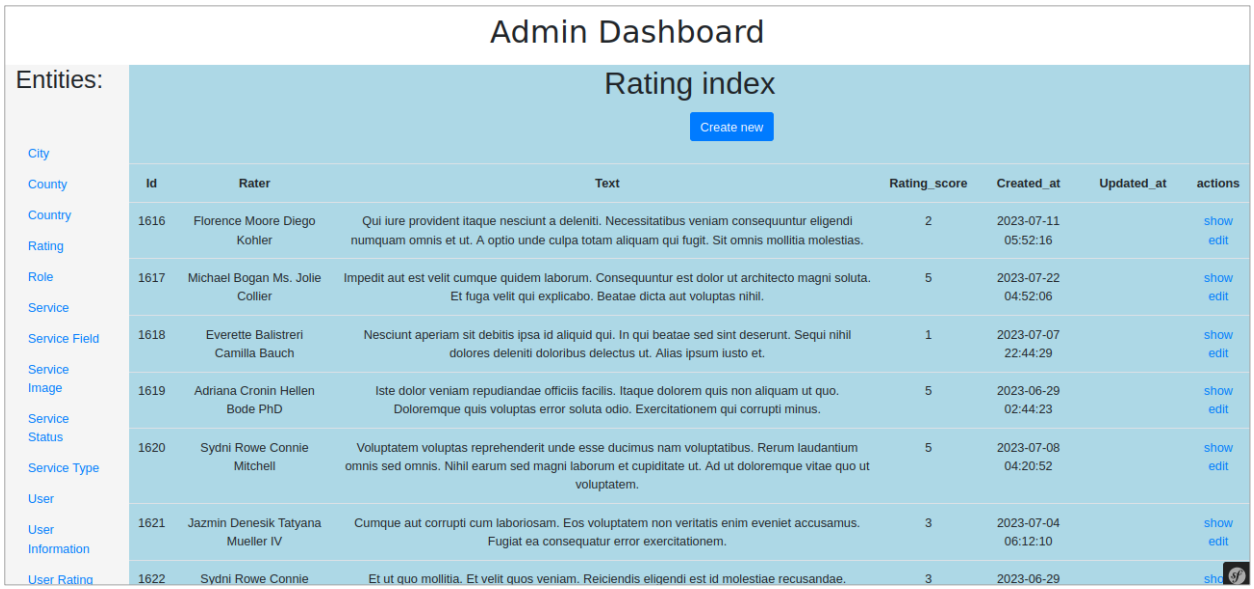


Slika 4.2 Parametri korisničkog entiteta na nadzornoj ploči

Na slici 4.3 se može vidjeti izgled nadzorne ploče kada je odabran tablični prikaz usluga. Ovaj entitet sadrži više parametara nego entitet „*User*“ te se on relacijski veže za korisnika koji je tu uslugu napravio. Dodatno, specifična usluga se relacijski veže za tri korisnika: svog vlasnika, kreatora i uređivača. Ovakvom se podjelom dobiju detaljnije informacije o tome tko je vlasnik oglasa, tj. koji korisnik nudi ili traži uslugu, tko je taj oglas objavio na web portal te koji je korisnik naknadno uredio taj oglas. U većini slučajeva za sva 3 parametra će biti ista osoba koja je vlasnik, kreator i uređivač vlastitog oglasa, no u slučajevima kada administrator ili uređivač mijenjanju određene parametre oglasa to će biti jasno dokumentirano. Parametar status, kao što je prethodno objašnjeno, opisuje kakva je dostupnost određene usluga. Slijedeći parametar predstavlja tip usluge tj. je li korisnik usluge traži ili nudi tu uslugu. Nakon toga slijedi parametar koji opisuje vrstu usluge koja može biti jedna od raznih koje se nude na web portalu. Slijedi opis, adresa, grad, županija, država te rok usluge. Korisnik postavlja rok usluge prilikom kreiranja te usluge te je on prikazan posjetiteljima portala.

Slika 4.3 Parametri entiteta usluge na nadzornoj ploči

Na slici 4.4 vidljiv je tablični prikaz ocjena korisničkih profila. Svaki entitet ocjene je relacijski povezan na osobu koja je dala tu ocjenu te osobu koja je ocjenjivana. Parametri koje ispunjava ocjenjivač su tekst i konačna ocjena za primljenu uslugu.



Slika 4.4 Parametri entiteta ocjene na nadzornoj ploči

Entitet „*City*“ sadrži parametre o svom imenu, poštanskom broju te županiji s kojom je relacijski vezan. Entitet „*County*“ predstavlja županiju te sadrži samo parametar za ime. U bazi podataka su uneseni svi gradovi i županije Republike Hrvatske, a pod entitetom „*Country*“ trenutno postoji samo Republika Hrvatska budući da je web aplikaciji cilj oglašavanje usluga na ovom području. Spremljene uloge u bazi podataka su iste uloge koje su prethodno navedene te se isto može reći i za vrstu, tip i status usluge. „*ServiceImage*“ je entitet koji sadrži ime slike koja je lokalno spremljena a pripada određenoj usluzi s kojom je relacijski povezana. Na ovaj način jedna usluga može sadržavati više slika. Zadnji entitet je „*UserRating*“ koji sadrži dodatne informacije o objavljenoj ocijeni i relacijski povezuje korisnike.

4.3 *Backend* razvoj naslovne stranice

Zahvaljujući programskom okviru, cjelokupan *kôd* korišten u uspostavljanju naslovne stranice podijeljen je na nekoliko dijelova.

Kada korisnik pošalje zahtjev za pristupom na naslovnu stranicu on dolazi i obrađuje se od strane upravljača koji u sebi sadrži funkciju za obradu tog zahtjeva. Sve funkcije vezane za stranicu koji krajnji korisnik koristi se nalaze u upravljaču „*BaseController*".

Kada upravljač dobije zahtjev on točno zna kojoj funkciji mora poslati taj zahtjev po ruti kojoj je korisnik pristupio. U ovom slučaju korisnik je pristupio ruti „/" što označava početnu stranicu web aplikacije. Budući da ispred svake funkcije piše ruta za koju funkcija obrađuje zahtjev upravljač pronalazi točnu funkciju te se zahtjev nadalje obrađuje preko nje.

Ova funkcija je zadužena za nekoliko stvari. Osnovna svrha joj je da vrati odgovor za zahtjev koji je poslan. Dakle odgovor koji će biti vraćen je prikaz šablone koja sadrži HTML *kôd* početne stranice te parametri koji će se proslijediti toj šabloni kao što je prikazano na slici 4.5. Među tim parametrima prosljeđuju se informacije o korisniku koji je pristupio stranici, usluge koje se trebaju prikazati na stranici, te parametri vezani za filtriranje tih usluga.

```
return $this->render('base/main.html.twig', [
    'services' => $services,
    'user' => $user,
    'serviceFields' => $serviceFields,
    'cities' => $cities,
    'counties' => $counties,
    'selectedField' => $selectedField,
    'selectedCity' => $selectedCity,
    'selectedCounty' => $selectedCounty
]);
```

Slika 4.5 Odgovor pomoću kojeg se prikazuje šablona i šalju parametri za prikaz

Prije nego što upravljač pošalje ove parametre mora ih prvo pronaći te spremiti u varijablu kao što je vidljivo iz slike 4.6. Prva stvar koja se pronalazi je korisnik koji je poslao zahtjev pomoću funkcije „*getUser()*" iz *security* paketa koju *Symfony* okvir nudi. Nakon toga, upravljač će pristupiti repozitorijima koji sadrže sve vrste usluga, županije i gradove te će ih spremiti u novodefinirane varijable.

Slika 4.6 Dohvaćanje korisnika i parametara za filtriranje

Nakon toga slijedi dio *kôda* koji je zaslužan za provođenje pretraživanja i filtriranja usluga koji je vidljiv na slici 4.7. Kada korisnik upiše ključnu riječ po kojoj želi pretražiti usluge ili postavi vrstu, grad ili županiju po kojoj usluge želi pretraživati, te će se vrijednosti privremeno spremiti u varijable i koristiti u svrhu reduciranja usluga po navedenim parametrima. Nadalje se ispituje je li korisnik unio ključnu riječ ili parametre za filtriranje te, ako nije, dohvaćaju se sve usluge sa stranice i kao takve se prosljeđuju šabloni. Ako je korisnik unio ključnu riječ ili filtrirao po parametrima, tada se ti parametri pronalaze u svojim repozitorijima i poziva se funkcija „*search()*" kojoj se prosljeđuju svi parametri te unesena ključna riječ.

```php
$searchTerm = $request->query->get('q');
$selectedField = $request->query->get('field');
$selectedCity = $request->query->get('city');
$selectedCounty = $request->query->get('county');

if ($searchTerm || $selectedField || $selectedCity || $selectedCounty) {
    $field = $serviceFieldRepository->findOneBy(['field' => $selectedField]);
    $county = $countyRepository->findOneBy(['county' => $selectedCounty]);
    $city = $cityRepository->findOneBy(['city' => $selectedCity]);
    $services = $serviceRepository->search($searchTerm, $field, $county, $city);
    shuffle($services);


} else {
    $services= $serviceRepository->findAll();
    shuffle($services);
}
```

Slika 4.7 Dohvaćanje korisnikovih zadanih filtera te filtriranje usluga iz baze

Funkcija „*search()*”, koja se nalazi u repozitoriju za usluge, će potom provesti niz upita na bazu preko DQL jezika. Baza će vratiti aplikacije točno one usluge koje u svojem nazivu ili opisu imaju ključnu riječ koju je korisnik naveo i koje imaju postavljene parametre koje je korisnik odabrao. Te će se usluge potom spremiti u varijablu i kao takve će se proslijediti šabloni za prikaz na stranici.

Tek nakon što su se ove operacije provele upravljač šalje odgovor sa svim prethodno navedenim varijablama i šablonom te se to prikazuje krajnjem korisniku.

Šablona početne stranice se nalazi u mapi „*templates*” u *Symfony* okviru te ona prima parametre poslane od upravljača pomoću kojih prikazuje komponente korisniku. Na početku ove šablone nalazi se naredbe za proširenje HTML *kôda* gdje mu se dodaje *kôd* iz „*index*” HTML datoteke koja sadrži navigacijsku traku i podnožje stranice. Ovo se radi da se izbjegne ponavljanje *kôda* za komponente koje se nalaze na više dijelova aplikacije. Slijedi manji dio *JavaScript kôda* koji služi za mijenjanje boja gumbova „Traži se” i „Nudi se” kao što se može vidjeti na slici 4.8.

```
function changeButtons(){
    var nudim = document.getElementById("nudim");
    var trazim = document.getElementById("trazim");

    if(nudim.classList.contains("btn-primary")){
        nudim.classList.remove("btn-primary");
        nudim.classList.add("btn-secondary");
        trazim.classList.remove("btn-secondary");
        trazim.classList.add("btn-primary");
    }else{
        trazim.classList.remove("btn-primary");
        trazim.classList.add("btn-secondary");
        nudim.classList.remove("btn-secondary");
        nudim.classList.add("btn-primary");
    }
}
```

Slika 4.8 Javascript kôd za promjenu stanja gumbova „Traži se” i „Nudi se”

Nakon toga nalazi se HTML *kôd* koji je zaslužan za prikazivanje trake sa slikama, gumbovi za tip oglasa, traka za pretraživanje i padajući izbornici za filtriranje. Provjerava se postoje li usluge u varijabli koju je šablona dobila od upravljača te je li trenutno aktiviran gumb „Traži se” ili „Nudi se”. Zatim se postavlja „for” petlja koja će ispisati svaku uslugu dobivenu od upravljača.

Pri ispisu pojedine usluge prvo se provjerava ima li usluga objavljene slike kao što je vidljivo na slici 4.9. Ako ima, prvo će se prikazati slika na kartici za uslugu, a ako nema onda će se prikazati slika s tekstom „Slika nije dostupna”.

```
{% for service in services %}
{% if service.serviceType == '1' %}

    <div class="col-sm-6">
        <div class="card mb-4">
            {% if service.serviceImages.first != null %}
                {% set break = false %}
                {% for image in service.serviceImages %}
                    {% if not break %}
                        {% if 'http' in image.image %}
                            <img class="card-img-top" src="{{ image.image }}"
                                 style="...">
                        {% else %}
                            {% set imgPath = 'uploads/' ~ image.image %}
                            <img class="card-img-top" src="{{ asset(imgPath) }}"
                                 style="...">
                        {% endif %}
                    {% set break = true %}
                    {% endif %}
                {% endfor %}

            {% else %
                <img class="card-img-top" src="https://excel..."
                     style="..."alt="Card image cap">
            {% endif %}
```

Slika 4.9 Prikaz slike na kartici usluge

Kao što dalje prikazuje slika 4.10, ispisati će se naslov, opis, cijena i rok usluge koje se lako dobiju pomoću varijable „*services*" primljene od upravljača. Kako bi dobili opis određene usluge potrebno je samo nakon „*services*" varijable napisati „*description*" i *Symfony* okvir će se pobrinuti za to da HTML *kôd* primi opis navedene usluge. Na kraju kartice usluge nalazi se gumb koji vodi na stranicu za dodatne detalje usluge gdje korisnik može vidjeti više o usluzi koja ga zanima.

```
<div class="card-body">
    <h5 class="card-title">{{ service.title }}</h5>
    <p class="card-text">{{ service.description|u.truncate(50, '...', false) }}</p>
    <p class="card-text"><b>Price: </b>{{ service.price }} €</p>
    <p class="card-text"><b>Ends {{ service.deadline ? service.deadline|ago }}</b></p>

    <a href="{{ path('app_service_details', {'id': service.id})}}" class="btn btn-primary">Vidi detalje</a>
</div>
```

Slika 4.10 Ispis tijela kartice usluge

4.4 *Backend* razvoj profilne stranice

Kao što je prethodno bilo opisano za naslovnu stranicu, zahtjev za pristupom na profilnu stranicu dolazi do upravljača koji je zaslužan za procesiranje tog zahtjeva. U ovom slučaju to je također „*BaseController*" koji u sebi sadrži funkciju s rutom „*/{id}/profile/*" gdje „*id*" predstavlja jedinstvenu identifikacijsku oznaku koju svaki korisnik dobije pri registraciji. Nadalje „*/profile*" naznačuje da se posjećuje profil čija je oznaka prethodno navedena.

Kao odgovor, ovaj upravljač vrača šablonu profilne stranice te nekoliko parametara koji su joj potrebni što je vidljivo na slici 4.11. Vrača identifikaciju korisnika koji pokušava pristupiti profilnoj stranici, identifikaciju vlasnika profilne stranice, dodatne informacije osobe koja pristupa stranici, usluge te ocjene koje posjeduje osoba s profilne stranice i prosječnu ocjenu profila.

```
return $this->render('base/profile.html.twig', [
    'user' => $user,
    'profileUser' => $profileUser,
    'userInformation' => $userInformation,
    'services' => $services,
    'ratings' => $userRatings,
    'averageRating' => $averageRating,
]);
```

Slika 4.11 Odgovor i parametri profilnog upravljača

Kao i u prethodnoj funkciji, korisnik se identificira pomoću *security* paketa i sprema u varijablu „*user*". Zatim se uzimaju dodatne informacije iz druge tablice u bazi koja je relacijski povezana s tablicom korisnika pomoću „*findOneBy*" funkcije. Ona će naći dodatne informacije za korisnika koji je u pitanju. Nakon toga se koristi vrlo slična funkcija, „*findBy*", koja će pronaći sve ocjene vlasnika profila i spremiti u varijablu „*userRatings*". Ovo dohvaćanje podataka se može vidjeti na slici 4.12.

```php
$user = $this->security->getUser();
$userInformation = $informationRepository->findOneBy(array('user' => $profileUser));
$userRatings = $userRatingRepository->findBy(array('user' => $profileUser));
```

Slika 4.12 Dohvaćanje identifikacije posjetitelja stranice i vlasnika profila

Nakon toga potrebno je izračunati prosječnu ocjenu profila kako bi se ona prikazala posjetitelju profila. To se odvija u *kôdu* na slici 4.13. Prvo se provjerava postoji li ijedna ocjena na profilu osobe te, ako ne postoji, šabloni će se poslati prazna varijabla što će uputiti šabloni da osoba s profila nije ni jednom ocijenjena. Ako postoji barem jedna ocjena na profilu, „*if*" grananje nastavlja računati koliko ukupno ima ocjena te će za svaku ocjenu dohvatiti uneseno bodovanje pomoću „*getRatingScore()*" funkcije te ih sve zbrojiti u varijabli „*sumOfRatings*". Nakon toga se računa prosječna ocjena tako da se suma podijeli s brojem ocjena te se to nadalje šalje šabloni u odgovoru upravljača. Jedino što je ostalo za dohvatiti su usluge vlasnika profila koje se jednostavno dohvaćaju sa „*getServices()*" funkcijom.

```php
$averageRating = null;
if($userRatings != null){
    $sumOfRatings = 0;
    $numberOfRatings = count($userRatings);
    foreach($userRatings as $rating)
    {
        $sumOfRatings = $sumOfRatings + $rating->getRating()->getRatingScore();
    }
    $averageRating = round($sumOfRatings / $numberOfRatings, 1);
}

$services = $profileUser->getServices();
```

Slika 4.13 Računanje prosječne ocjene vlasnika profila i dohvaćanje usluga

Kao i šablona za naslovnu stranicu, profilna šablona također započinje s naredbom proširivanja sa „index.html” datotekom koja sadrži navigacijsku traku i podnožje. Šablona započinje s provjerom posjeduje li vlasnik profila sliku profila kao što je vidljivo iz slike 4.14. Ako ne posjeduje, umjesto profilne slike će se postaviti uobičajena slika korisnika, a ako posjeduje provjeriti će se koja je vrsta slike u pitanju te će se prikazati u kružnom obliku na sučelju.

```
{% if profileUser.image != null %}
    {% if 'http' in profileUser.image %}
        <img src="{{ profileUser.image }}" style="" alt="Avatar" width="150px" height="150px">
    {% else %}
        {% set imgPath = 'uploads/' ~ profileUser.image %}
        <img src="{{ asset(imgPath) }}" style="" alt="Avatar" width="150px" height="150px">

    {% endif %}

{% else %}
    {% set imgPath = 'uploads/generic_profile.png' %}
    <img src="{{ asset(imgPath) }}" style="" alt="Avatar" width="150px" height="150px">
{% endif %}
```

Slika 4.14 Provjera i postavljanje slike profila

Nadalje se provjerava je li posjetitelj profila vlasnik profila. Ovo je vrlo važno budući da će stranica izgledati drugačije ako je posjetitelj ujedno i vlasnik. Biti će prikazane dodatne funkcije kao što su izmjena osobnih podataka i dodavanje usluga te dodatne informacije o profilu. Dakle ako se radi o vlasniku profila, ispisani naslov će glasiti „Moj profil”, a ako to nije slučaj, ispisani naslov će glasiti „Korisnički profil” kao što je vidljivo na slici 4.15. Također će se ispisati prosječna ocjena profila koja je prethodno izračunata u upravljaču.

```
{% if user == profileUser %}
    <h1 style="margin-bottom:20px;">Moj profil</h1>
{% else %}
    <h1 style="margin-bottom:20px;">Korisnički profil</h1>
{% endif %}
{% if ratings != null%}
    <h4>Prosječna ocjena: {{ averageRating }} / 5 &#9734; </h4>

{% endif %}
```

Slika 4.15 Provjera identiteta posjetitelja profila te ispis prosječne ocjene profila

Nadalje se ispisuju podaci o profilu osobe. Ako je posjetitelj profila ujedno i vlasnik profila ispisat će se sve informacije koje je korisnik unio prilikom registracije, a ako to nije slučaj, ispisati će se samo kontakt podaci i korisničko ime osobe. Zatim se ponovno ispituje identitet posjetitelja profila i po odgovoru se postavlja ili gumb za izmjenu informacija profila ili gumb za ocjenjivanje profila kao što je prikazano na slici 4.16.

```
{% if user == profileUser %}
    <a class="btn btn-warning" href="{{ path('app_profile_user_edit') }}"
       role="button">Uredi profil</a>
{% else %}
    <a class="btn btn-secondary" href="{{ path('app_rate_user', {'id': profileUser.id}) }}"
       role="button">Ocijeni korisnika</a>
{% endif %}
```

Slika 4.16 Ispisivanje gumba za izmjenu informacija ili za ocjenjivanje profila

Slijedeće će se ispisati sve usluge koje je vlasnik profila objavio u skraćenom obliku s poveznicom na detalje svake usluge. *Kôd* za ovo je vrlo sličan ispisu usluga na naslovnoj stranici pomoću „for" petlje. Ako vlasnik profila nije objavio niti jednu uslugu tj. upravljač je predao šabloni praznu varijablu, ispisati će se tekst „Korisnik nema objavljene usluge".

Na slici 4.17 se može vidjeti *kôd* kojim se ispisuju sve ocjene vlasnika profila na dnu profila. Započinje ispitivanjem postoji li barem jedna ocjena vlasnika profila te, ako postoji, „for" petlja prolazi kroz svaku ocjenu te ispisuje ime korisnika koji je dao ocjenu, tekst ocjene i samu ocjenu na kraju kartice. Ako ne postoji niti jedna ocjena ispisati će se tekst „Korisnik nema ocjena."

```
{% if ratings is not empty %}
{% for rating in ratings %}

    <div class="col-sm-4">
        <div class="card mb-4">
            <div class="card-body">
                <a href="{{ path('app_profile', {'id': rating.rating.rater.id})}}"
                   class="card-title">{{ rating.rating.rater }}</a>
                <p class="card-text">{{ rating.rating.text }}</p>
                <h4 class="card-text">{{ rating.rating.ratingScore }}/5 &#9734;</h4>
                <p class="card-text"><b>Posted {{ rating.rating.createdAt ? rating.rating.createdAt|ago }}</b></p>
            </div>
        </div>
    </div>
{% endfor %}
```

Slika 4.17 Ispis svake ocjene vlasnika profila

Stranica sa detaljima pojedine usluge je strukturirana na isti način. Dakle provjerava se je li posjetitelj stranice isto tako i vlasnik oglasa te, ako je to slučaj, prikazuju se dodatne mogućnosti kao što je izmjena oglasa. Nadalje se ispisuju sve informacije o usluzi zajedno sa slikama koje oglas usluge sadržava.

4.5 *Backend* razvoj stranice za prijavu i registraciju

Zahtjev za stranicom prijave obrađuje upravljač „*SecurityController*" koji je omogućen preko *security* paketa *Symfony* okvira. Sam po sebi, upravljač ne sadržava puno *kôda*, a obavlja posao validacije i odjave korisnika koji je poslao zahtjev. Prijavu i odjavu obrađuju dvije zasebne funkcije u upravljaču.

Funkcija za prijavu obrađuje zahtjev koji dolazi sa rute „*/login*" te ona samo prikazuje šablonu za prijavu zajedno s pogreškom koja je uslijedila prilikom pristupa stranici. Funkcija za odjavu obrađuje zahtjev s rute „*/logout*" te je ona prazna zbog sposobnosti *security* paketa koje će prepoznati da treba odjaviti korisnika ako funkcija sadrži ime „*logout*".

Šablona koja se prikazuje korisniku je također vrlo jednostavne strukture. Ona sadrži produženje pomoću „index.html" datoteke, naslov, dva tekstualna polja i gumb za prijavu.

Sam proces verificiranja korisnika se događa u autentifikatoru. Autentifikator je klasa koja može biti automatski generirana od strane *security* paketa, a svrha joj je validacija i autentifikacija korisnika uz točne podatke koje je korisnik unio. Autentifikator za prijavu se naziva „*LoginFormAuthenticator*". On počinje definiranjem tri varijable pomoću koje se dohvaćaju svi korisnici iz baze, definira pomoćna varijabla za lakše preusmjeravanje na upravljače, te se definira funkcija za *hashiranje* zaporke koja će naknadno biti jako važna.

Glavni mehanizam za prijavu se nalazi u funkciji „*authenticate*" koji se može vidjeti na slici 4.18. Funkcija započinje dohvaćanjem korisničkog unosa za polja korisničkog imena i zaporke te ih sprema u varijablu kako bi se oni verificirali. Zatim se stvara nova instanca klase „*UserBadge*" kojoj je zadaća validacija prethodno unesenih podataka. Prvo se dohvaća korisnik u bazi za kojeg je osoba upisala korisničko ime te se provjerava postoji li korisnik u bazi za to navedeno ime. Funkcija vrača validacijski objekt „*Passport*" ako je zaporka koju je korisnik unio jednaka zaporki korisnika iz baze. To se provjerava pomoću funkcije *„password_verify"* koja *hashira* zaporku koja je unesena od strane korisnika i uspoređuje takav oblik zaporke sa *hashiranim* oblikom zaporke u bazi. Ako su obje zaporke u *hashiranom* obliku jednake, korisnik je unio točnu zaporku i nadalje proces preuzima funkcija „*onAuthenticationSuccess*".

```
public function authenticate(Request $request): Passport
{
    $email = $request->request->get('username');
    $password = $request->request->get('password');
    $userBadge = new UserBadge($email, function($userIdentifier) {
        $user = $this->userRepository->findOneBy(['username' => $userIdentifier]);

        if (!$user) {
            throw new UserNotFoundException();
        };

        return $user;
    });

    return new Passport($userBadge,
        new CustomCredentials(function($credentials, User $user) {
            if(password_verify($credentials, $user->getPassword())) {
                return true;
            }
        }, $password)
    );
}
        }
    }

    return $allUserRoles;
}
```

Slika 4.18 Funkcija za validaciju korisnika pri prijavi

Funkcija „*onAuthenticationSuccess*" samo preusmjerava korisnika na upravljač zadužen za naslovnu stranicu koji će mu prikazati istu. Nakon ovoga korisnik će biti prijavljen i moći će pristupiti svojoj profilnoj stranici. Ako validacija nije bila uspješna, pozvati će se funkcija „*onAuthenticationFailure*" koja će korisnika preusmjeriti na prethodno objašnjeni upravljač za prijavnu stranicu.

Registracijski proces se u potpunosti odvija u upravljaču „*RegistrationController*". Kada osoba pošalje zahtjev za pristupom ruti „*/register*" pokreče se funkcija zadužena za registraciju korisnika. Na samom početku funkcije definira se nekoliko varijabli kao što je vidljivo iz slike 4.19.

```
$user = new User();
$userInformation = new UserInformation();
$formUser = $this->createForm(RegistrationFormType::class, $user);
$formUserInfo = $this->createForm(UserInformationFormType::class, $userInformation);
$formUser->handleRequest($request);
$formUserInfo->handleRequest($request)
```

Slika 4.19 Stvaranje novog korisnika i definiranje obrazaca

Stvara se nova instanca entiteta „*User*" i „*UserInformation*" koja će se spremiti u bazu ako dođe do uspješne registracije. Nakon toga stvaraju se obrasci koje će korisnik morati ispuniti putem registracije da bi uspješno izradio vlastiti račun.

Obrasce čine lista parametara entiteta koje će se korisniku prikazati kao neka vrsta komponente za unos kao što je tekstualno polje ili padajući izbornik. Te komponente su automatski generirane pomoću obrasca. Dakle u obrascu je važno navesti one parametre entiteta koje korisnik mora ispuniti te preko upravljača ju poslati šabloni, a zatim šablona točno zna koje vrste unosa mora generirati. Dio registracijskog obrasca može se vidjeti na slici 4.20.

```
$builder
    ->add('name')
    ->add('surname')
    ->add('username')
    ->add('email')
    ->add('image', FileType::class, [
        'mapped' => false,
        'required' => false,
    ])
```

Slika 4.20 Dio registracijskog obrasca

Nakon definiranja obrazaca provjerava se je li korisnik unio sve podatke u obrascima i jesu li oni validni pomoću „*if*" grananja kao što je vidljivo na slici 4.21. Ako korisnik još nije unio podatke u obrazac, upravljač će jednostavno vratiti prikaz registracijske šablone i poslati joj navedene obrasce. Ako je korisnik unio validne podatke, upravljač kreče sa definiranjem parametara korisnika. On postavlja korisničku ulogu na broj 4. Kao što je spomenuto u prethodnim poglavljima, ovaj broj označava dopuštenja koja će korisnik imati na web portalu. Broj 4 predstavlja ulogu *ROLE_USER* tj. korisnik će od sada imati svoj vlastiti profil i mogućnost objave usluga i ocjenjivanja drugih korisnika. Svaki entitet u bazi sadrži parametre „*createdAt*" i „*updatedAt*" u slučaju da ti podaci zatrebaju u budućnosti te se oni, prilikom kreiranja novih instanca, postavljaju na trenutno vrijeme. Slijedeći parametar koji se postavlja je zaporka, no prije toga će ona proći proces *hashiranja* kako se u bazu ne bi spremila u čitljivom obliku.

To se radi pomoću sučelja „*UserPasswordHasherInterface*" i funkcije „*hashPassword*" kojoj se navodi korisnik čija je zaporka u pitanju te zaporka u čitljivom obliku. Ona zatim vrača *hashiranu* zaporku koja se sprema u bazu. Potom se instanca dodatnih informacija korisnika veže za instancu korisnika.

```
if ($formUser->isSubmitted() && $formUser->isValid() && $formUserInfo->isSubmitted() && $formUserInfo->isValid()) {
    $user->setRole(4);
    $user->setCreatedAt(new \DateTime());
    $user->setPassword(
        $userPasswordHasher->hashPassword(
            $user,
            $formUser->get('plainPassword')->getData()
        )
    );

    $userInformation->setUser($user);
    $userInformation->setCreatedAt(new \DateTime());

return $allUserRoles;

}
```

Slika 4.21 Provjera valjanosti obrasca i postavljanje početnih parametara

Zadnje što se treba postaviti je slika koju je korisnik unio kao sliku profila. Prvo se provjerava je li korisnik unio sliku budući da to nije bilo obavezno da registracija obavi. Ako je korisnik unio sliku, ona će se spremiti u lokalni repozitorij „*public/uploads*". Prije samog spremanja joj se promjeni ime i dodaje jedinstveni identifikator kako bi se slika mogla lakše pretražiti i razlikovati od drugih spremljenih slika. Ovo se radi pomoću funkcije „*urlize*" te će se ime slike u takvom formatu spremiti u bazu. Kada profilna stranica bude trebala prikazati tu sliku krajnjem korisniku, upravljač će po imenu slike koje dobiva iz baze pretražiti tu sliku u lokalnom repozitoriju te ju prikazati korisniku. Proces spremanja slike u repozitorij može se vidjeti na slici 4.22.

```php
$image = $formUser["image"]->getData();

if($image){
    $destination = $this->getParameter('kernel.project_dir').'/public/uploads';
    $originalFilename = pathinfo($image->getClientOriginalName(), PATHINFO_FILENAME);
    $newFilename = Urlizer::urlize($originalFilename).'-'.uniqid().'.'.$image->guessExtension();
    $user->setImage($newFilename);
    $image->move(
        $destination,
        $newFilename
    );
}
```

Slika 4.22 Proces spremanja slike u repozitorij i imena slike u bazu

Zadnja stvar koja će se odraditi je spremanje prethodno definiranih podataka korisnika u bazu pomoću „*entityManager*" klase i funkcija „*persist*" i „*flush*. Upravljač će na kraju vratiti autentifikator kojim će automatski odraditi i prijavu korisnika na web portal.

Ovaj način definiranja i validacije obrasca te spremanja u bazu također se ponavlja na stranicama za objavu i izmjenu usluge, izmjenu vlastitih podataka te ocjenjivanje.

5. ZAKLJUČAK

„Pametni portal“, web aplikacija za razmjenu usluga, je razvijen s ciljem pružanja platforme za oglašavanje usluga koje često nemaju priliku biti oglašivane. Taj prioritet je oblikovao početnu bazu za ovu web aplikaciju te dizajn aplikacije. Kao što je objašnjeno, cilj dizajna korisničkog sustava je taj da bude jednostavan i razumljiv te da pruži brzo rukovanje aplikacijom. U današnje doba web aplikacije mogu biti razvijene s ciljem pružanja što više informacija, opcija i mogućnosti svojim korisnicima ne uzimajući u obzir da korisnici često dolaze na web aplikacije s ciljem rješavanja jednostavnog problema.

Zbog toga, navigacijska traka sadržava samo najvažnije opcije, traka sa slikama sadrži kratke i jasne informacije zbog kojih korisnici dolaze na stranicu te su, pri pregledu oglasa, usluge jasno podijeljene na one koje se nude i one koje se traže. Sam profil korisnika sadržava sve potrebne informacije i opcije koje su lako razumljive te je proces dodavanja novog oglasa napravljen slično kao i proces registracije korisnika. Na ovaj se način korisniku ne pruža previše novih informacija koje mora analizirati.

Korisničko iskustvo na web aplikaciji je također poprilično izravno i modelirano po potrebama i ugodnostima korisnika web aplikacije. Korisniku se nudi pet uloga koje može uzeti prilikom korištenja portala ovisno o njegovim potrebama i ciljevima. Korisnik koji je po prvi put došao na web aplikaciju ili mu je u cilju samo pregledavanje usluga će imati ulogu neprijavljenog korisnika, dok će s druge strane korisnici kojima je u cilju objavljivanje oglasa izraditi korisnički račun te postati prijavljeni korisnik s ciljem davanja ili primanja usluga.

Pozadinske funkcionalnosti „Pametnog portala“ su također namijenjene da budu točno ono što je na web aplikaciji potrebno. Nakon razvoja funkcionalnosti za registraciju korisnika te su funkcionalnosti iskorištene i kod drugih kao što je prijava korisnika.

S druge strane, nedostatak web portala je upravo manjak posredovanja između izmjene usluga. Novčane transakcije bi omogućile korisnicima rezervaciju pojedine usluge te način sigurnog procesa dogovora i plaćanja. U opisu rezervacije, zainteresirani korisnik bi se mogao predstaviti oglašivaču te objasniti razloge i uvjete suradnje. Oglašivač bi mogao prihvatiti zadovoljavajuću rezervaciju te nastaviti dogovore u vezi datuma i cijene obavljanja usluge. Na kraju, primatelj usluge bi preko stranice mogao uplatiti davatelju usluge ili podići spor (*dispute* na engleskom) ako nije zadovoljan primljenom uslugom. Video oglašavanje bi donijelo puno potencijala u predstavljanju davatelja usluga jer bi posjetitelji mogli vidjeti kako bi se usluga odradila i uvjeriti

se u kvalificiranost davatelja usluge. Nedostatak aplikacije za razgovore preko stranice također nije idealan za portal kojem je cilj povezivanje osoba koje traže i osoba koje nude usluge.

U svakom slučaju, iako nesavršena, web aplikacija „Pametni Portal“ daje jednostavno i izravno korisničko iskustvo te funkcionalnosti jednostavne prirode no značajno potrebne za pružanje platforme uslugama raznih vrsta. Web aplikacija nije dizajnirana da uzima prevelik utjecaj od sličnih i puno uspješnijih stranih web rješenja, nego joj je svrha da na pojednostavljen način pruža svojim korisnicima nešto što nisu u mogućnosti dobiti na mnogim oglasnim stranicama: platformu koja daje priliku i velikim i malim uslugama.

LITERATURA

1) Symfony - https://symfony.com/

1) Netokracija - https://www.netokracija.com/top-freelancing-platforme-usporedba-153498

2) ddiy.co - https://ddiy.co/how-does-fiverr-work

3) Dir.hr - https://dir.hr/sto-je-freelancer/

4) Hostinger - https://www.hostinger.com/tutorials/best-freelance-websites#2_Toptal

5) Upwork - www.upwork.com

SAŽETAK

Naslov: Web aplikacija za razmjenu usluga

Sažetak: Cilj web aplikacije za razmjenu usluga je pružiti platformu za oglašavanje raznovrsnih usluga kroz jednostavno web sučelje. Korisnicima stranice je ponuđena objava oglasa kojim nude ili traže usluge te preko kojih drugi korisnici mogu stupiti u kontakt s njima. Vrste usluga uključuju usluge programiranja, slikanja, fotografiranja, instrukcija te glazbene produkcije. Web aplikacija sadrži jednostavno sučelje te korisnicima nudi mogućnost izrade vlastitog korisničkog računa na kojem će biti vidljive sve usluge koje korisnik traži ili nudi. Korisnicima se također daje mogućnost ocjenjivanja drugih korisnika što će formirati prosječnu ocjenu korisničkog računa. Web aplikacija je realizirana pomoću PHP programskog jezika te *Symfony* programskog okvira koji nudi pomoćne pakete za izradu stranica kao što su paketi za sigurnosni aspekt aplikacije, uređivanje HTML *kôda* te jednostavnije kreiranje upita na bazu. Web aplikacija se na jednostavan način predstavlja korisniku te nastoji što jasnije objasniti korisnikove mogućnosti u svakom trenutku.

Ključne riječi: jednostavno sučelje, korisnički računi, objava oglasa, raznovrsne usluge

ABSTRACT

Title: Web application for exchange of services

Summary: The goal of the service exchange web application is to provide a platform for advertising a variety of services through a simple web interface. The users of the site are offered the publication of advertisements in which they offer or request services and through which other users can get in touch with them. Types of services include programming, painting, photography, instruction and music production services. The web application contains a simple interface and offers users the possibility of creating their own user account, which will display all the services that the user requests or offers. Users are also given the option to rate other users which will form an average user account rating. The web application was created using the PHP programming language and the Symfony programming framework, which offers auxiliary packages for creating pages, such as packages for the security aspect of the application, editing HTML code, and simpler creation of database queries. The web application is presented to the user in a simple way and tries to explain the user's options as clearly as possible at all times.

Keywords:, posting ads, simple interface, user accounts, various services

PRILOZI

Prilog P1 - svi programski *kôdovi* objašnjeni u 4. i 5. poglavlju nalaze se na *github* računu Luke Madžarca. *Github* profil se nalazi na linku: https://github.com/lukaMadzarac7, a repozitorij web aplikacije se nalazi na linku: https://github.com/lukaMadzarac7/ZavrsniRad-Symfony.