Web aplikacija za prijavu i traženje zaposlenika

Prpić, Jakov

Master's thesis / Diplomski rad

2022

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:200:491607

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-05-04**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

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

Sveučilišni studij

WEB APLIKACIJA ZA PRIJAVU I TRAŽENJE ZAPOSLENIKA

Diplomski rad

Jakov Prpić

Osijek, 2022.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 02.12.2022.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime Pristupnika:	**Jakov Prpić**
Studij, smjer:	**Diplomski sveučilišni studij Računarstvo**
Mat. br. Pristupnika, godina upisa:	D-1153R, 13.10.2020.
OIB studenta:	15053564254
Mentor:	**Izv. prof. dr. sc. Ivica Lukić**
Sumentor:	,
Sumentor iz tvrtke:	Stjepan **Udovičić**
Predsjednik Povjerenstva:	Izv. prof. dr. sc. Mirko Köhler
Član Povjerenstva 1:	**Izv. prof. dr. sc. Ivica Lukić**
Član Povjerenstva 2:	**Izv.prof.dr.sc. Zdravko Krpić**
Naslov diplomskog rada:	**Web aplikacija za prijavu i traženje zaposlenika**
Znanstvena grana diplomskog rada:	**Informacijski sustavi (zn. polje računarstvo)**
Zadatak diplomskog rada:	**U diplomskom radu potrebno je omogućiti kreiranje životopisa na temelju već postojanih predložaka i njihovo uređivanje. Treba pružiti način spremanja i kasnijem pristupanju, te dijeljenju korisničkog životopisa. Nadalje, potrebno je pružiti način kako bi se kreirani životopisi mogli pretraživati prema ključnim riječima, te pronađene se korisnike treba moći kontaktirati. Ostvareno programsko rješenje potrebno je ispitati na prikladnim uvjetima korištenja te analizirati korisničko iskustvo. Rezervirano za: Josip Prpić**
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	**Primjena znanja stečenih na fakultetu: 3 bod/boda** **Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda** **Jasnoća pismenog izražavanja: 2 bod/boda** Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	02.12.2022.

Potvrda mentora o predaji konačne verzije rada:	***Mentor elektronički potpisao predaju konačne verzije.***
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 12.12.2022.

Ime i prezime studenta:	Jakov Prpić
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1153R, 13.10.2020.
Turnitin podudaranje [%]:	8

Ovom izjavom izjavljujem da je rad pod nazivom**: Web aplikacija za prijavu i traženje zaposlenika**

izrađen pod vodstvom mentora Izv. prof. dr. sc. Ivica Lukić

i sumentora ,

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD *1*

1.1. Zadatak završnog rada **1**

PREGLED PODRUČJA TEME *2*

2.1. Pregled sličnih rješenja **2**

2.1.1. LinkedIn 2

2.1.2. ZipRecruiter 3

2.1.3. MyOpportunity 4

2.1.4. Europass 4

2.2. Prijedlog rješenja **5**

2.2.1. Iskustvo kandidata 5

2.2.2. Iskustvo poslodavca 5

3. IDEJNO RJEŠENJE I ARHITEKTURA WEB APLIKACIJE *7*

3.1. Zahtjevi na web aplikaciju **7**

3.2. Prijedlog arhitekture rješenja **7**

3.2.1. Arhitektura rješenja iskustva kandidata 8

3.2.2. Arhitektura rješenja iskustva poslodavca 10

3.3. Predgled korištenih tehnologija **11**

3.3.1. HTML 11

3.3.2. Tailwind CSS 12

3.3.3. JavaScript 12

3.3.4. Vue.js 13

3.3.5. Vuex store 14

3.3.6. PHP 14

3.3.7. Laravel 15

4. PROGRAMSKO RJEŠENJE WEB APLIKACIJE *16*

4.1. Postavljanje razvojne okoline **16**

4.2. Dinamičko sučelje **17**

4.3. Kreiranje i uređivanje životopisa **19**

4.3.1. Unos adrese 19

4.3.2. Unos kontakta 20

4.3.3. Unos iskustva 21

4.3.4. Dodavanje vještina 21

4.4. Kreiranje opisa tvrtke i oglasa za posao **24**

4.4.1. Opis tvrtke 24

4.4.2. Oglasi za posao 24

4.5. Pretraga korisnika **26**

4.6. Preporučeni poslovi **27**

4.7. Prijava na oglas za posao **27**

4.8. Generiranje lažnih podataka **28**

5. ISPITIVANJE PROGRAMSKOG RJEŠENJA I KORIŠTENJE ***30***

5.1. Korisničko iskustvo kandidata **30**

5.2. Korisničko iskustvo poslodavca **38**

5.3. Administracijsko sučelje **43**

6. ZAKLJUČAK ***46***

LITERATURA ***47***

SAŽETAK ***48***

ABSTRACT ***49***

PRILOZI ***50***

1. UVOD

Digitalni životopisi postaju nužnost pri traženju poslovnih mjesta i pojavljuje se sve veća potražnja za njima. Različiti poslovi u različito specijaliziranim područjima znači da postoje i razlike u vrstama životopisa. Životopis za radno mjesto u banci, primjerice, izgleda vrlo drugačije od istog dokumenta za posao gdje kandidat mora biti kreativan. Digitalizacija životopisa nudi zapis svih kandidatovih specijalnosti, vještina, radnih iskustava te ih obrađuje online u jedinstvenu cjelinu.

Dugi niz godina papirnati životopisi bili su potencijalni način predstavljanja kandidata novom poslodavcu, odnosno svojih vještine i drugih karakteristike. Polako, kako se svijet komunikacije prebacuje na digitalno, postavlja se pitanje: Zašto bi bilo drugačije i sa životopisima? Digitalizacija životopisa poslodavcima uvelike olakšava posao time što se uvodi sposobnost pretraživanja tisuće i tisuće životopisa prema uvjetima koje poslodavac traži.

Jedna od prednosti digitalnog životopisa je dostupnost: umjesto da samo jedan poslodavac posjeduje životopis, omogući pregledavanje životopisa u bilo kojem trenutku od bilo koje potencijalno zainteresirane strane. Također, kandidat ne mora slati životopis poštom ukoliko se ne nalazi u blizini poslodavca i čekati nekoliko dana kako bi životopis stigao, već poslodavac životopis može pogledati s bilo kojeg mjesta, kada god želi.

Cilj ovog diplomskog rada je razvoj web aplikacije kojom će kandidat imati mogućnost kreiranja i uređivanja te dijeljenja životopisa. Nakon kreiranja životopisa, poslodavci će moći tražiti kandidate prema njihovim navedenim znanjima i vještinama. Takvim pristupom će poslodavac lakoćom doći do kandidata najsličnijima traženom. Nakon pronalaska traženog kandidata, poslodavcu je omogućen način kontaktiranja kandidata. Također, poslodavcima je pružena mogućnost kreiranja objave za posao. Poslodavac definira traženog zaposlenika s kojim se kandidatov životopis uspoređuje prilikom pretrage objava za posao. Tako će se korisnicima moći prikazati pozicije koje su njima najpristupačnije.

1.1. Zadatak završnog rada

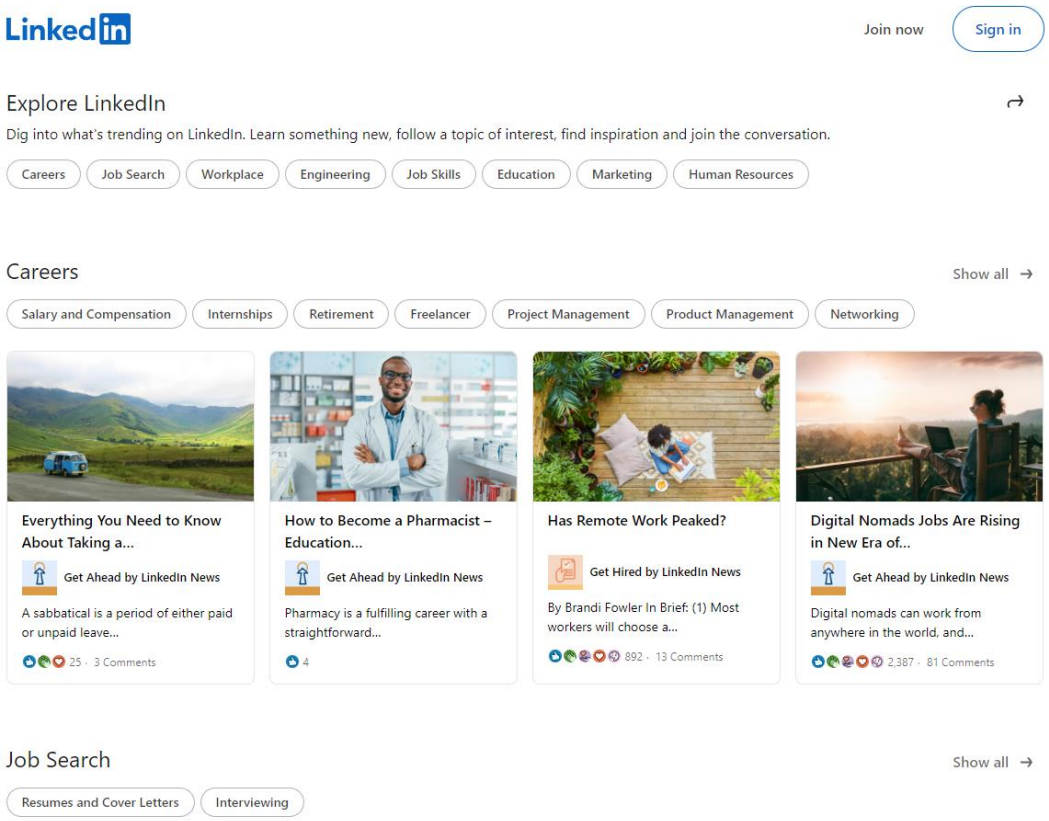
U diplomskom radu potrebno je omogućiti kreiranje životopisa na temelju već postojanih predložaka i njihovo uređivanje. Treba pružiti način spremanja i kasnijem pristupanju, te dijeljenju korisničkog životopisa. Nadalje, potrebno je pružiti način kako bi se kreirani životopisi mogli pretraživati prema ključnim riječima, te pronađene se korisnike treba moći kontaktirati. Ostvareno programsko rješenje potrebno je ispitati na prikladnim uvjetima korištenja te analizirati korisničko iskustvo.

PREGLED PODRUČJA TEME

2.1. Pregled sličnih rješenja

2.1.1. LinkedIn

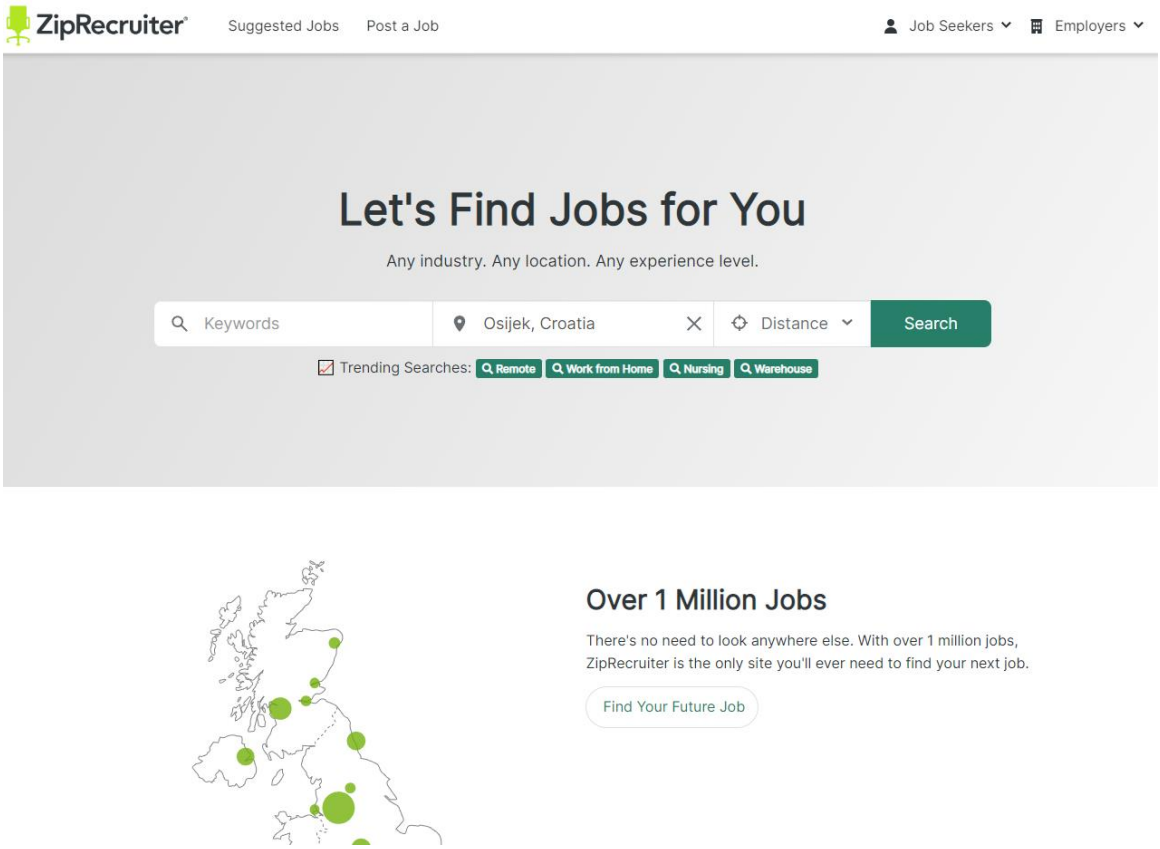
Jedna od najčešće spomenutih platformi kada se priča o digitalnim životopisima je LinkedIn. S obzirom na to da LinkedIn ima više od 875 milijuna korisnika u više od 200 zemalja i teritorija diljem svijeta, LinkedIn ostaje vodeća platforma za razmjenu digitalnih životopisa [1]. Jedne od prednosti ove platforme su istraživanje buduće tvrtke, odnosno lako je proučiti što tvrtka zahtjeva od kandidata u smislu vještina. Nadalje, platforma nudi komunikaciju s postojećim kontaktima, gdje se omogućava komunikacija poslodavca i kandidata, kao i komunikacija kolega i bivših menadžera. Zbog velikog broja korisnika, lako je proširiti svoju mrežu te stvoriti prave kontakte u tvrtkama za koje je korisnik zainteresiranih [2]. Iako, upravo veliki broj korisnika može dovesti do težeg zamjećenja. Jednostavno imati profil na ovoj platformi nije dovoljno te da bi poslodavac primijetio kandidata, kandidat mora privući značajnu količinu pažnje, odnosno redovno ažuriranje profila i redovita aktivnost što može biti stresno. Jedna od naslovnih stranica LinkedIn-a prikazana je sljedećom slikom 2.1.



Slika 2.1. *Naslovna stranica 'Istraži' LinkedIn-a*

2.1.2. ZipRecruiter

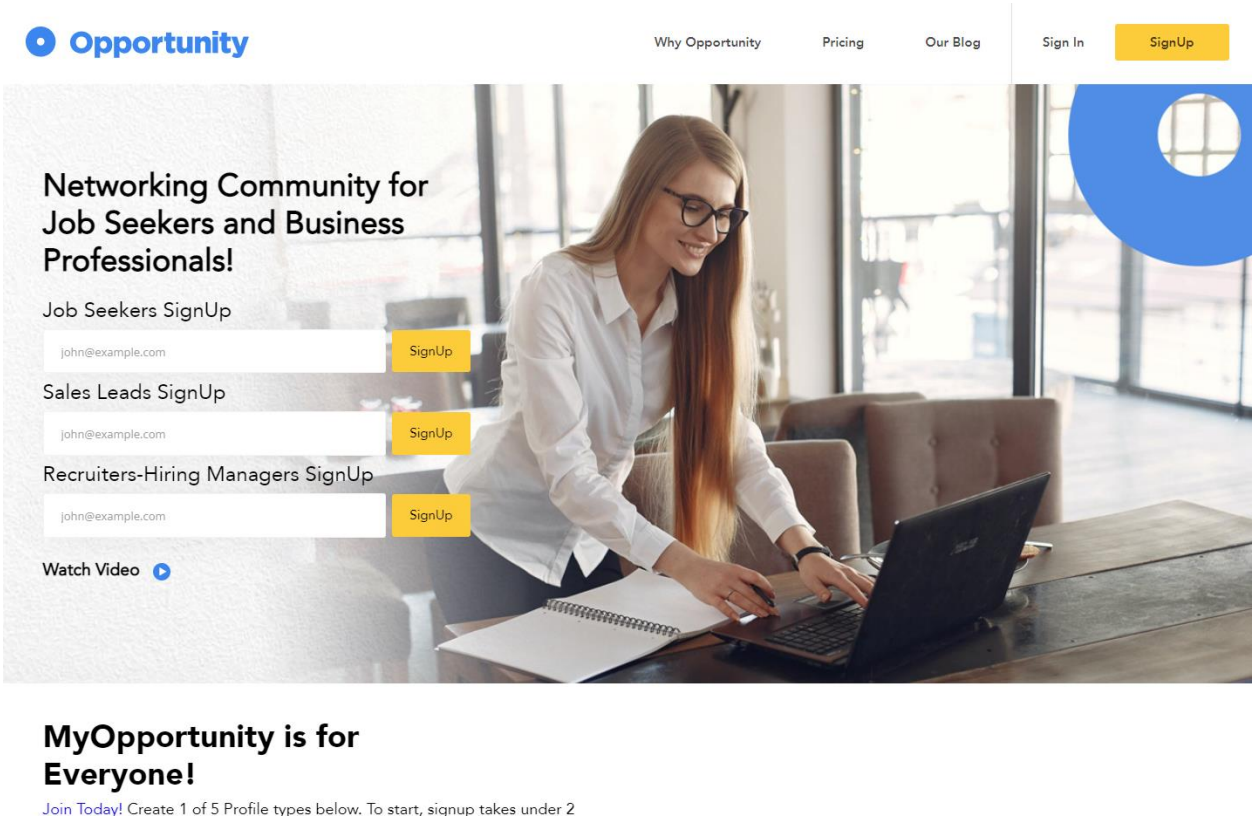
ZipRecruiter je iduće slično rješenje. To je platforma temeljena na oblaku koja omogućava široki raspon oglasnih ploča za različite poslove. Mjesečno preko 7 milijuna kandidata traži posao preko ZipRecruiter platforme [3]. Oglasi za posao se lako poredaju po vremenu, lokaciji i ključnoj riječi te je platforma jednostavna za korištenje općenito. Preporuka poslova korisniku odvija se uz pomoć umjetne inteligencije. Uz to, postoji mogućnost pretraživanja životopisa za kvalificirane kandidate što uvelike štedi vrijeme. Također, moguće je pretraživati više kandidata, dodavati bilješke na njihov profil i ocijeniti ih. No, loše strane se ističu. Izbori industrije su ograničeni te nisu idealni. Platforma se naplaćuje što je vrlo odbojno većini korisnika. I uz umjetnu inteligenciju, većinu puta pruža nekvalificirane kandidate za posao ili kandidatima nudi poslove koji im nimalo ne odgovaraju. Ukoliko se kandidat prijavi na više radnih mjesta u istoj tvrtki, poslodavcu se kandidat prikazuje odvojeno za svako mjesto te se kontaktira za svako mjesto odvojeno – iako je proces razgovora identičan za svaku poziciju. Također, moguće je poslati prazan životopis na oglas za posao, što ne bi trebalo biti dozvoljeno [4]. Naslovna stranica ZipRecruiter platforme je prikazana sljedećom slikom 2.2.



Slika 2.2. *Naslovna stranica ZipRecruiter platforme*

2.1.3. MyOpportunity

Zadnje razmatrano rješenje je MyOpportunity. To je društvena poslovna mreža koja koristi algoritam za pronalaženje stručnjaka koji pružaju prilike za zapošljavanje, umrežavanje i odnose. Nakon prijave kandidat navodi uslugu koju pruža i uslugu koju treba, te se na temelju navedenih usluga spaja s drugim potencijalnim partnerima. Problem s tim modelom spajanja je taj što stranica tjera korisnike da traže neku uslugu od drugih, iako nisu zaista zainteresirani. To dovodi do konstantnog obavještavanja korisnika o uslugama koje ih zapravo ne zanimaju [5]. Naslovna stranica prikazana je sljedećom slikom 2.3.



Slika 2.3. *Naslovna stranica MyOpportunity*

2.1.4. Europass

Za lako stvaranje i ažuriranje životopisa može se koristiti i Europass. Stvaranjem profila na Europass-u je moguće iz unesenih podataka automatski napraviti životopis. Stvaranje profila je dosta opsežno, a određeni unosi podataka se mogu izostaviti. Tijekom kreiranja životopisa odabire se njegov dizajn nakon čega se pretvara u PDF oblik. Uz to, razvijen je program za planiranje učenja ili karijera u Europi [6] . Naslovna stranica prikazana je na sljedećoj slici 2.4.

Slika 2.4. *Naslovna stranica europass*

2.2. Prijedlog rješenja

Kako je proces zapošljavanja kompleksan, smjer razvoja aplikacije treba biti pojednostavljenje tog procesa za kandidate i poslodavce. Temelj zapošljavanja osoba su njihovi životopisi i oglasi za posao, stoga aplikacija treba osigurati njihovo brzo i precizno stvaranje.

2.2.1. Iskustvo kandidata

Pisanje životopisa je mukotrpan proces, pa je potrebno staviti poseban naglasak na što veće pomaganje korisniku tijekom stvaranja životopisa uzimajući u obzir ne ograničavanje korisnika ni u kojem smislu, te da krajnji rezultat bude smislena cjelina koja ima značenje poslodavcu i da aplikacija može procesirati stvoreni životopis. Uz to, ako kandidat već ima postojani životopis, potrebno je osigurati brz prijepis informacija u obrazac aplikacije, što znači da obrazac životopisa treba pratiti konvencije standardnog životopisa. Nakon stvaranja životopisa, kandidatu je potrebno na temelju unesenih informacija pružiti oglase za posao koji mu zaista odgovaraju. Razmatranjem postojanih rješenja vidljivo je da to nije lako ostvarivo. Kako osobe tijekom traženja posla gledaju zahtjeve oglasa, potrebno je osigurati da preporučeni oglasi za posao traže onaj opseg vještina koje osoba ima. Uz vještine, treba paziti i na ostale aspekte posla, kao što su lokacija i potrebne godine iskustva. Također, kandidatu treba pružiti mogućnost detaljnijeg pregleda oglasa za posao u slučaju da aplikacija nije najbolje pretpostavila kompetenciju kandidata.

2.2.2. Iskustvo poslodavca

Proces zapošljavanja novih zaposlenika počinje definiranjem oglasa za posao koji sadrži odgovornosti i vještine budućeg zaposlenika. Nakon toga, definirani oglas za posao treba podijeliti

na mjestima gdje će privući potencijalne kandidate. Tek nakon ostvarenog kontakta s kandidatom, poslodavac dobiva pregled kandidatovog životopisa. Razvojem aplikacije moguće je otkloniti određene prepreke i tako ubrzati cijeli proces. Tijekom stvaranja oglasa za posao, potrebno je pružiti slobodu pisanja poslodavcima kako bi prenijeli kandidatima sve potrebne informacije. Uz to, treba osigurati spremanje podataka oglasa na način naknadnog dohvaćanja odgovarajućih korisnika. Nadalje, stvoreni oglas za posao potrebno je moći podijeliti osobama koje nisu prijavljene u aplikaciju. Kao što se kandidatu preporučuju oglasi za posao, tako se poslodavcima istim algoritmom trebaju preporučiti odgovarajući kandidati. Također, dohvaćanjem odgovarajućih kandidata poslodavac može pregledati njihove životopise, te na temelju predloženog odlučiti zadovoljava li kandidat opisane uvjete.

Kako se životopisi i oglasi za posao trebaju držati standardnih formi i prikazivati informacije kao i svaki drugi, mogućnost dijeljenja životopisa i oglasa za posao stvorenih unutar aplikacije uveliko bi povećalo korisničko iskustvo smanjenjem potrebe za ponovnim radom.

3. IDEJNO RJEŠENJE I ARHITEKTURA WEB APLIKACIJE

U ovome poglavlju navedeni su zahtjevi korištenja aplikacije te je prikazana arhitektura rješenja potrebna za razvoj aplikacije kako bi se navedeni zahtjevi uspješno ostvarili te se potrebne funkcionalnosti uspješno implementirale.

3.1. Zahtjevi na web aplikaciju

Web aplikacija treba pružati različita sučelja ovisno o ulozi prijavljenog korisnika. Kandidatu je potrebno omogućiti kreiranje, uređivanje i dijeljenje svog životopisa. Nakon kreiranja životopisa potrebno mu je prikazati postojane oglase za posao koji njemu najviše odgovaraju. Kako bi se odazvao na oglas za posao, potrebno je omogućiti način prijave za posao te prikaz povratne informacije poslodavca.

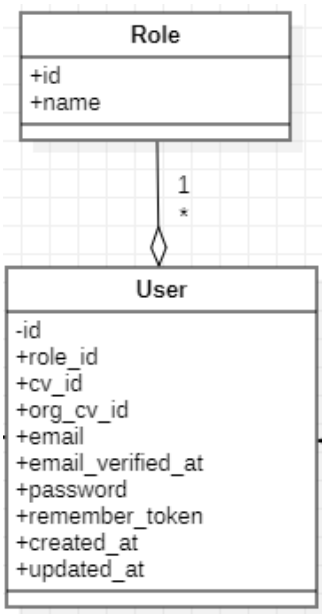
Sučelje za poslodavca treba omogućiti kreiranje opisa tvrtke koji bi kandidati mogli pročitati kako bi dobili općenitu ideju o tvrtki. Nakon toga, potrebno je omogućiti kreiranje oglasa za posao koji obuhvaća potrebne vještine koje kandidat treba imati kako bi uspješno obavljao navedeni posao. Korisniku tvrtke je također potrebno prikazati kandidate koji su se prijavili na određeni oglas, pružiti mogućnost detaljnijeg pregleda njihovog životopisa, te mogućnost odgovora prihvaćanja ili odbijanja kandidata.

Kako bi korisnik koji je zadužen za administraciju aplikacije imao svoje potrebne privilegije, potrebno je i osigurati sučelje administratora. Na takvom sučelju potrebno je pružiti sve spremljene podatke i pružiti mogućnost uređivanja i brisanja podataka.

3.2. Prijedlog arhitekture rješenja

Aplikacija sadrži tri načina korištenja ovisno o ulozi korisnika. Za svaku ulogu korisnika, napravljen je dijagram tijeka korištenja aplikacije, te je opisan proces pripreme baze podataka kako bi se unešeni podaci efektivno spremili za kasnije korištenje. Kako je proces registracije i prijave korisnika isti za svaku web aplikaciju, nije sadržan u dijagramima tijeka.

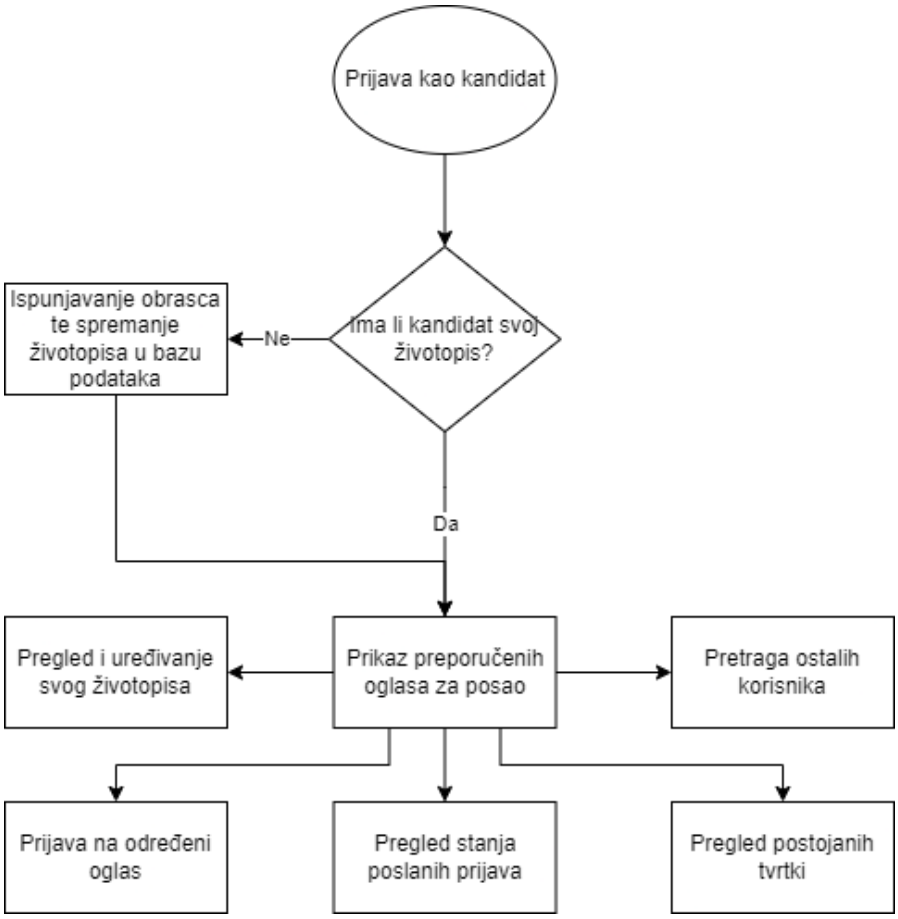
Kako bi se razdvojili korisnici uloga poslodavaca i kandidata, napravljena je tablica korisnik s opcionalnim stranim ključevima na životopis i opis tvrtke te je prikazana UML dijagramom na sljedećoj slici 3.3.



Slika 3.3. *UML dijagram tablice Korisnika*

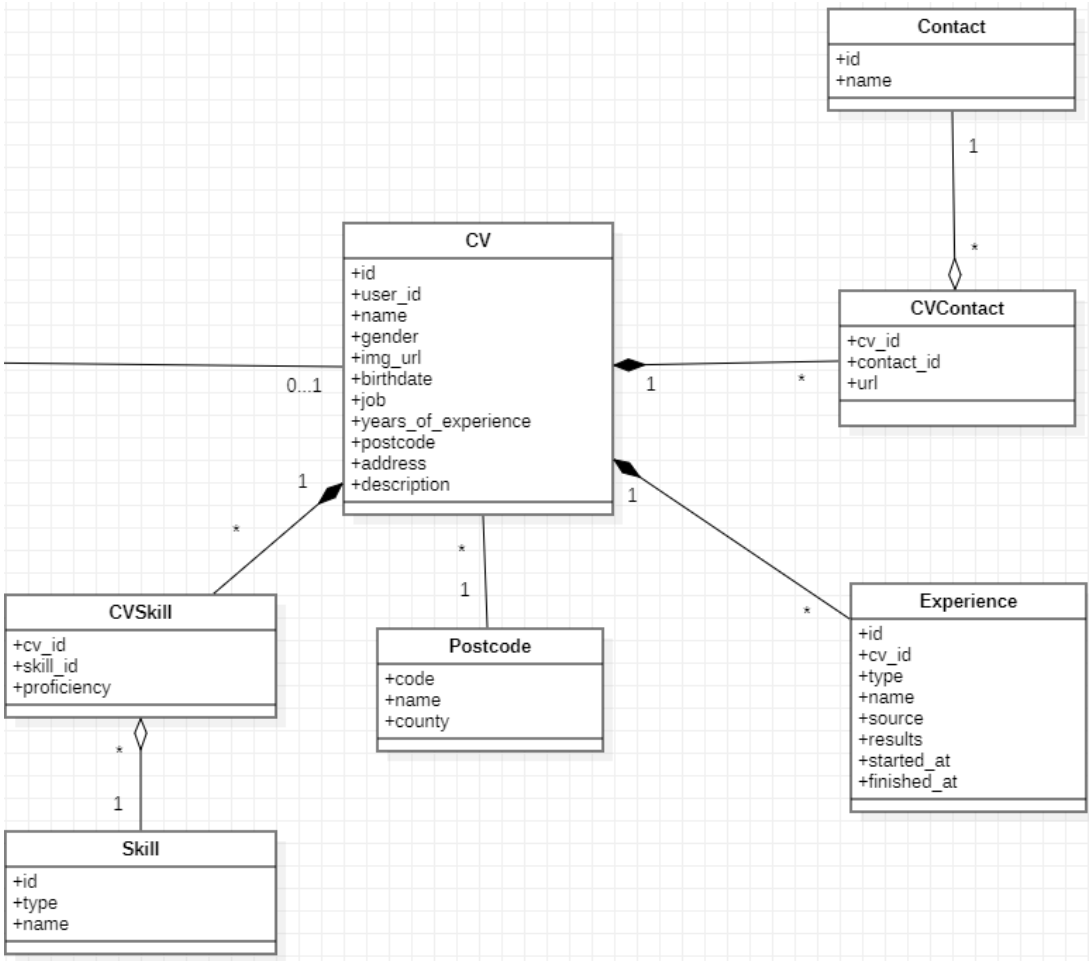
3.2.1. Arhitektura rješenja iskustva kandidata

Dijagram tijeka iskustva kandidata od prijave korisnika kao kandidat do mogućnosti prijavljivanja na različite oglase za posao prikazan je na sljedećoj slici 3.4.



Slika 3.4. *Dijagram tijeka iskustva kandidata*

Nakon prijave korisnika s ulogom kandidata, potrebno je osigurati da korisnik napravi svoj životopis. Ispunjavanje obrasca kreiranja životopisa treba biti intuitivno te pružiti korisniku što više slobode kako se ne bi osjećao ograničeno aplikacijom. Treba osigurati način ispunjavanja tako da iako svi korisnici imaju isti obrazac i da postoji očita razlika među životopisima korisnika aplikacije. Iako korisniku treba pružiti što više slobode, potrebno je paziti na način spremanja tih podataka kako bi se korisnikovi podaci mogli kasnije pretraživati s obzirom na zahtjeve poslodavaca. Na temelju opisanog, ostvarena struktura životopisa prikazana je UML dijagramom na sljedećoj slici 3.5.



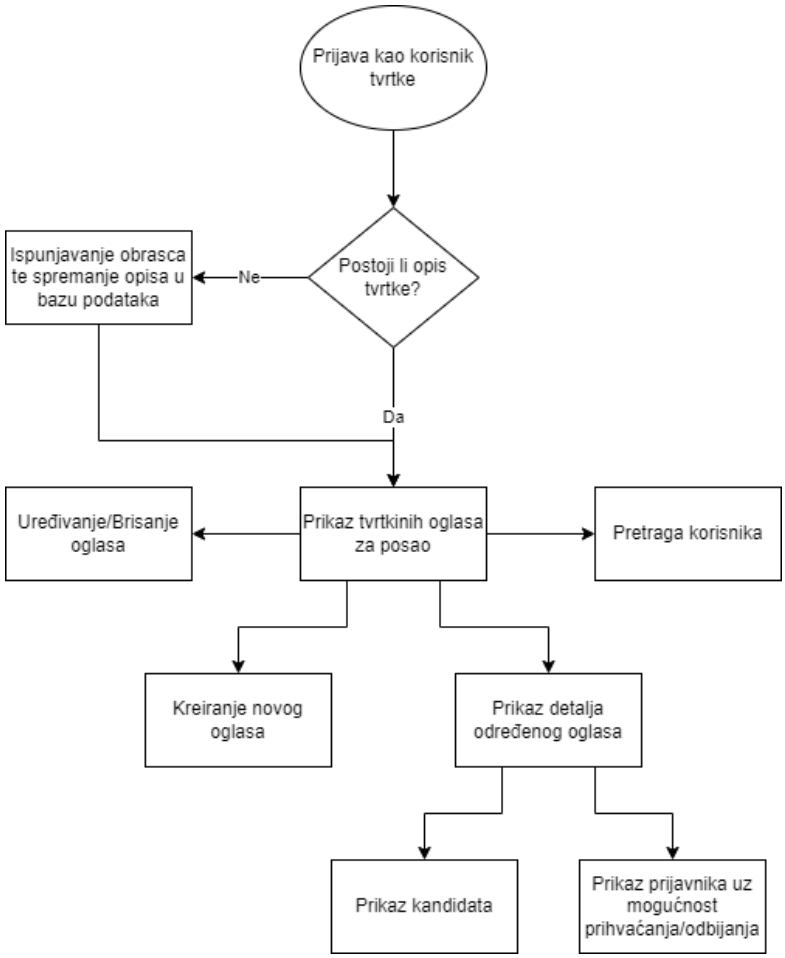
Slika 3.5. *UML dijagram životopisa*

Kreiranjem životopisa korisnik unosi opće informacije o sebi kao što su ime i prezime, dob, spol i ostale. Kako korisnik može imati više načina kontaktiranja, potrebno je spremiti proizvoljan broj oblika predodređenih vrsta kontakta što zahtjeva N:M vezu. Nadalje, korisnik može imati više iskustava gdje su bitne informacije gdje se iskustvo dogodilo, je li to iskustvo bilo radno ili edukacijsko, kada je započelo i kada je završilo te koji su rezultati tog iskustva. Iskustva se povezuju s korisnikom uz pomoć 1:N veze. Lokacija korisnika se sprema u obliku poštanskog broja koji je strani ključ na tablicu sa svim mogućim lokacijama te specifičnom adresom za

korisnika. Ova veza je također 1:N te je korisna pri kasnijem pretraživanju korisnika u slučaju da je lokacija bitna u oglasu za posao. Najbitniji dio strukture je način spremanja korisnikovih vještina. U bazi podataka prije nego što korisnik krene raditi svoj životopis potrebno je imati već unesene moguće vještine koje će korisnik moći odabrati i povezati sa svojim životopisom. Korisniku je prepušteno subjetkivno ocjenjivanje koliko je dobar u pojedinoj vještini. Takvo ponašanje je također ostvareno N:M vezom.

3.2.2. Arhitektura rješenja iskustva poslodavca

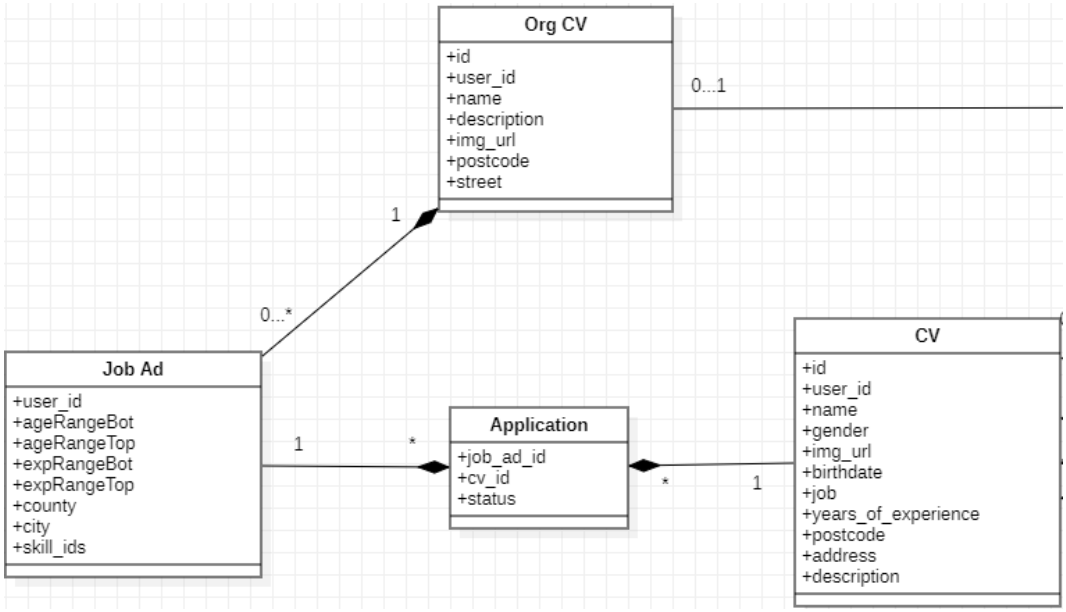
Dijagram tijeka iskustva poslodavca od prijave korisnika kao poslodavac do kreiranja opisa tvrtke i oglasa za posao te prihvaćanja/odbijanja prijavnika prikazan je na sljedećoj slici 3.6.



Slika 3.6. *Dijagram tijeka iskustva poslodavca*

Nakon prijave korisnika kao poslodavac, prvi korak je kreiranje opisa tvrtke. Ispunjavanjem ponuđenog obrasca spremaju se opće informacije o tvrtki. Nakon kreiranja opisa, poslodavcu je pružena mogućnost kreiranja oglasa za posao. Kreiranjem oglasa za posao poslodavac određuje parametre po kojima želi filtrirati kandidate. Nakon stvaranja oglasa za posao poslodavcima se preporučuju korisnici koji odgovaraju traženim parametrima. Također, pregledom kreiranih oglasa

za posao poslodavcima se prikazuju i kandidati koji su se prijavili na taj oglas, te je pružena mogućnost prihvaćanja ili odbijanja prijavnika. U svrhu ostvarivanja navedenog, napravljena arhitektura prikazana je UML dijagramom na sljedećoj slici 3.7.



Slika 3.7. *UML dijagram implementacije prijavljivanja na oglas*

Kreiranjem opisa tvrtke poslodavci kreiraju unos s kojim se mogu njihovi oglasi za posao povezivati. Opis tvrtke daje općenite informacije koje će kandidat vidjeti kada posjeti njihov opis s bilo kojeg oglasa. Kako tvrtka može imati više oglasa za posao veza tih tablica je 1:N. Oglasi za posao na temelju parametara vještina, područja, dobi i godina iskustva dohvaćaju odgovarajuće kandidate, te ih prikazuju poslodavcima. Također, nakon što se kreira oglas za posao, odgovarajućim kandidatima se prikazuju kako bi se mogli prijaviti na njih. Poveznica između kandidatovog životopisa i oglasa za posao je kandidatova prijava, što predstavlja međutablicu kojom se ostvaruje N:M veza oglasa i životopisa.

3.3. Predgled korištenih tehnologija

3.3.1. HTML

Za objavljivanje informacija za globalnu distribuciju, potreban je univerzalno razumljiv jezik. Izdavački jezik koji koristi World Wide Web (WWW) je HTML (engl. *HyperText Markup Language*). HTML daje autorima sredstva da: objave online dokumente s naslovima, tekstom, tablicama, popisima, fotografijama itd.; dohvaćanje online informacija putem hiper-tekstualnih veza jednim klikom na gumb; dizajniranje obrazaca za obavljanje transakcija s udaljenim uslugama i mnogo drugih.

HTML koristi *markup* za označavanje teksta, slika i drugog sadržaja za prikaz u web pregledniku. HTML element odvojen je od drugog teksta u dokumentu pomoću „tagova“ koji se sastoje od naziva elementa okruženog znakovima za manje-veće. Svaki od elemenata može se mijenjati određenim brojem atributa koji konfiguriraju elemente ili prilagođavaju njihovo ponašanje na određene načine. HTML dokumenti međusobno su povezani poveznicama [7]. HTML može ugraditi programe napisane u skriptnom jeziku, kao što je JavaScript, što utječe na ponašanje i sadržaj web stranica.

3.3.2. Tailwind CSS

Tailwind CSS je CSS framework koji na prvo mjesto stavlja korisnost. Umjesto da se usredotočuje na funkcionalnost stavke koja se stilizira, Tailwind je usredotočen na to kako bi ona trebala biti prikazana. Time se olakšava razvojnom programeru testiranje novih stilova i promjenu izgleda.

Drugim riječima, Tailwind je vrlo prilagodljiv CSS framework niske razine koji daje sve građevne blokove potrebne za izradu dizajna po mjeri bez velikog napora. Pozitivna stvar ovog framework-a je što ne nadmeće specifikaciju dizajna ili zadanu temu koja se mora koristiti – jednostavno se spajaju manje komponente kako bi se konstruiralo jedinstveno korisničko sučelje. Ono što Tailwind jednostavno radi jest uzima „sirovu“ CSS datoteku, obrađuje ju preko konfiguracijske datoteke i proizvodi jedinstveni izlaz. Također, Tailwind nudi tisuće ugrađenih klasa koje ne zahtijevaju kreiranje dizajna od nule. Stoga, programer ne mora sam pisati CSS pravila. Te CSS klase glavni su razlog zašto je izrada i stilizacija Tailwindom vrlo brza [8].

Prednosti ovog CSS frameworka su: nije potrebno pisati imena za CSS klase i ID-ove, minimalni broj redaka koda u CSS datoteci, prilagođavanje dizajna za izradu komponenti, web stranice su responzivne i razne druge [9].

3.3.3. JavaScript

JavaScript je objektno orijentirani skriptni jezik stvoren kako bi omogućio ne-programerima proširenje web stranica izvršnim kodom na strani klijenta. Za razliku od tradicionalnijih jezika, kao što su Java ili C#, JavaScript nema klase te ne potiče enkapsulaciju niti čak strukturiranje programiranja. Umjesto toga, JavaScript nastoji maksimizirati fleksibilnost [10].

JavaScript je jezik visoke razine, ima dinamičko tipkanje (engl. *dynamic typing*), objektnu orijentaciju temeljenu na prototipu te prvoklasne funkcije. Ima aplikacijska programska sučelja (API) za rad s tekstom, datumima, regularnim izrazima, standardnim strukturama podataka i

modelom objekta dokumenta (DOM). Izvorno, JavaScript se koristio samo u web preglednicima, no danas je osnovna komponenta nekih poslužitelja i raznih aplikacija.

Osnovni JavaScript jezik na strani klijenta sastoji se od nekoliko uobičajenih programskih značajki koje omogućuju slijedeće: pohranu korisnih vrijednosti unutar varijabli, operacije na dijelovima teksta (engl. *strings*), izvođenje koda kao odgovor na određene događaje na web stranici [11].

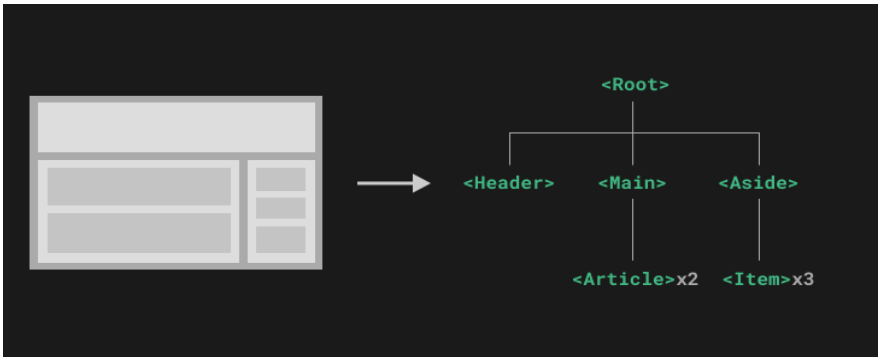
3.3.4. Vue.js

Vue.js je *front end* progresivni framework otvorenog koda (engl. *open-source*) zasnovan na model-view-viewmodel (MVVM) arhitekturi. Koristeći jezik JavaScript, Vue.js se koristi za kreiranje korisničkih sučelja te *Single-Page* aplikacija. Dizajniran je od temelja tako da se postepeno može usvojiti i lako skalirati između biblioteke i framework-a, ovisno o različitim slučajevima korištenja [12]. Framework ima postupno prilagodljivu arhitekturu koja se usredotočuje na deklarativno renderiranje i sastav komponenti.

Osnovna biblioteka fokusirana je samo na sloj pogleda (*engl.* view). S vremenom kako zahtjevi korisnika rastu, mogu se prilagoditi dodatne biblioteke za različite funkcionalnosti [13]. Napredne značajke, kao što su usmjeravanje (engl. *routing*), upravljanje stanjem i alati za izgradnju, nude se putem službeno održavanih pomoćnih biblioteka i paketa. Vue.js omogućuje proširenje HTML-a s HTML atributima koji se nazivaju direktivama. Direktive nude funkcionalnost HTML aplikacijama te dolaze kao ugrađene ili korisnički definirane direktive.

Dvije ključne značajke Vue.js-a su deklarativno renderiranje i reaktivnost. Deklarativno renderiranje označava mogućnost framework-a da proširuje standardni HTML sa sintaksom predloška koja omogućuje deklarativno opisivanje HTML izlaza na temelju stanja JavaScript-a. Reaktivnost pak označava sposobnost Vue.js-a da automatski prati promjene stanja JavaScript-a i učinkovito ažurira DOM (engl. *Document Object Model*) kada se dogode promjene.

Vue.js koristi komponente (engl. *components*) koje omogućuju podjelu korisničkog sučelja na nezavisne dijelove koji se mogu ponovno koristiti te se o svakome dijelu može razmišljati zasebno. Uobičajeno je da se aplikacija organizira u stablo ugniježđenih komponenti, prikazano na slici 3.8.

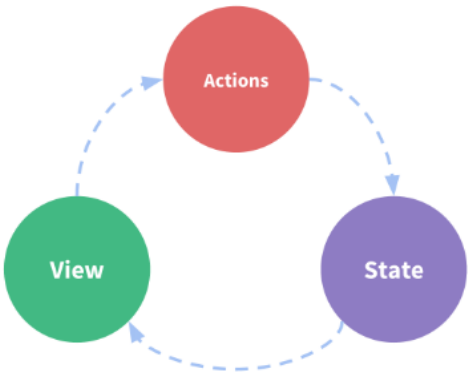


Slika 3.8. *Stablo ugniježđenih komponenti u Vue.js aplikaciji*

Taj način podjele je vrlo sličan načinu na koji se ugnježđuju izvorni HTML elementi, no Vue.js implementira vlastiti model komponenti koji omogućuje enkapsulaciju prilagođenog sadržaja i logike u svakoj komponenti.

3.3.5. Vuex store

Vuex je obrazac upravljanja stanja aplikacije i biblioteka za Vue.js. Služi kao središnje spremište za sve komponente u aplikaciji, s pravilima koja osiguravaju da se stanje može mijenjati samo na predvidljiv način. Sastoji se od tri dijela: stanje što predstavlja izvor istine aplikacije, pogled na stanje i akcija kojom se stanje može promijeniti kao reakcija na korisničke unose [14]. Zajednički rad navedenih dijelova prikazan je sljedećom slikom 3.9.



Slika 3.9. *Prikaz zajedničkog rada dijelova Vuex store-a*

3.3.6. PHP

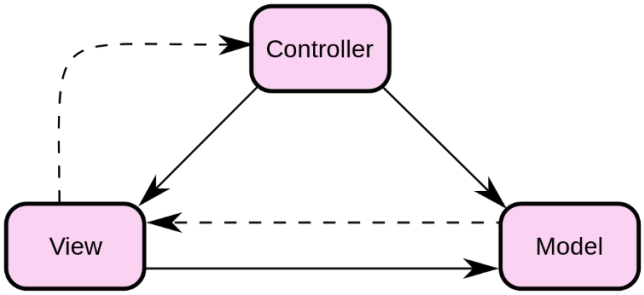
PHP je skriptni jezik opće namijene namijenjen web razvoju. PHP kod se obično obrađuje na web poslužiteljima pomoću PHP interpretera implementiranog kao modul, *daemon* ili kao izvršna datoteka Common Gateway Interface (CGI). Na web poslužitelju, rezultat interpretiranog i izvršenog PHP koda, koji može biti bilo koja vrsta podataka, npr. generirani HTML ili binarni slikovni podaci, činio bi cijeli ili dio HTTP odgovora.

Glavna značajka PHP jezika su objekti za rukovanje dijelovima teksta (engl. *string*) i nizovima (engl. *array*), kao i potpora objektnom orijentiranom programiranju. Uz korištenje standarnih i opcijskih modula proširenja, PHP aplikacija može komunicirati s bazama podataka poput *MySQL* ili *Oracle*, crtati grafove, kreirati PDF datoteke te čitati XML datoteke. Također, moguće je pisanje vlastitih modula proširenja u programskom jeziku C. PHP se može vrtiti na operacijskom sustavu Windows, što programeru dopušta kontrolu nad Windows aplikacijama poput *Word* ili *Excel*. PHP se može koristiti na tri primarna načina: programiranje na strani poslužitelja, skriptiranje pomoću komandne linije (engl. *command line*) i GUI (engl. *Graphical User Interface*) aplikacije na strani klijenta [15].

Dodatno, PHP se može koristiti za mnoge programske zadatke izvan web konteksta, kao što su samostalne grafičke operacije i robotsko upravljanje dronom.

3.3.7. Laravel

Laravel je *open-source* (softver otvorenog koda) web framework. Baziran je na PHP-u te je jedan od vodećih PHP framework-a. Koristi se *model-view-controller* (MVC) arhitektura pri razvoju web aplikacija. Način komunikacije komponenti arhitekture prikazan je sljedećom slikom 3.10.



Slika 3.10. *MVC koncept*

MVC model se sastoji od podataka, poslovnih pravila, logike funkcija ugrađenih u programsku logiku. Pogled (engl. *view*) predstavlja prikaz prethodno modeliranih podataka te zahtjeva od modela informacije potrebne za stvaranje prikaza modela korisniku. Model predstavlja podatke i poslovnu logiku određene aplikacije te dojavljuje sebi pridruženim pogledima i upraviteljima kada dođe do promjene u njegovom stanju. Upravitelj (engl. *controller*) upravlja korisničkim zahtjevima te šalje naloge modelu kojima ažurira njegovo stanje ili šalje naredbe u poglede kojima mijenja prikaz modela.

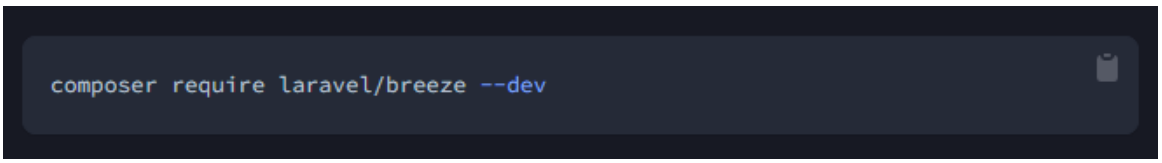
Laravel pruža mogućnosti postavljanja baze podataka, davanja sučelja preko kojeg se grade naredbe za pretragu, uspostava ovjere autentičnosti i sigurnosti i dr. Vrlo je pristupačan i moćan framework koji nudi razne alate za kreiranje kompleksnih aplikacija [16].

4. PROGRAMSKO RJEŠENJE WEB APLIKACIJE

Nakon stvaranja idejnog rješenja, tj. analiziranja zahtjeva i stvaranja arhitekture rješenja, slijedi programsko rješenje kojom se ostvaraju potrebne funkcionalnosti. Ovo poglavlje započinje postavljanjem razvojne okoline, te nastavlja prikazom razvijenih modela, kontrolera i pogleda u svrhu ostvarivanja potrebnih funkcionalnosti.

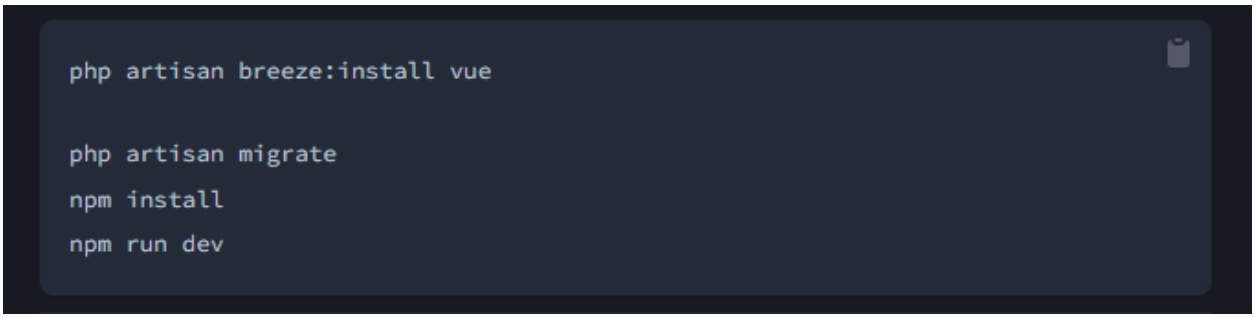
4.1. Postavljanje razvojne okoline

Laravel nudi više načina kako započeti projekt. Kako bi prepustili Laravel-u da se pobrine za registraciju, prijavu i autentikaciju korisnika, koristi se Laravel Breeze. Kako Javascript koristi NPM, tako Laravel koristi composer za instalaciju modula, te se instalacija pokreće naredbom prikazanom na idućoj slici 4.1.



Slika 4.1. *Dohvaćanje Laravel Breeze implementacije*

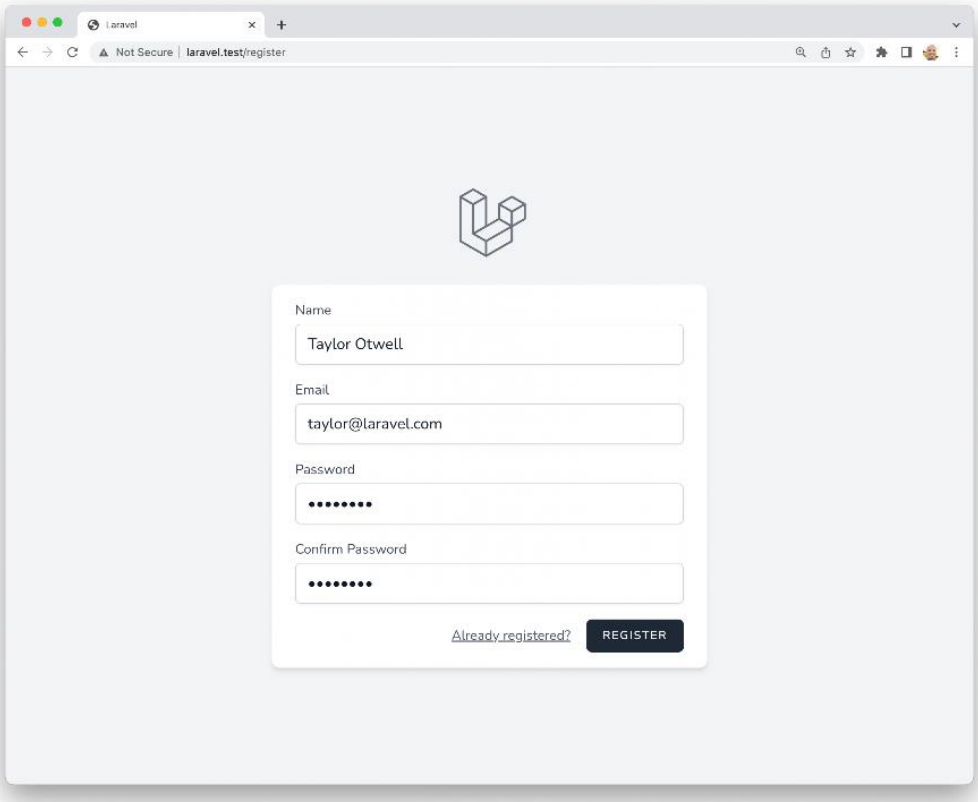
Nakon dohvaćanja potrebnih podataka, potrebno je odabrati koja se tehnologija želi koristiti za frontend. Radi ostvarivanja veće reaktivnosti web aplikacije koristi se Inertia.js, te su potrebne naredbe prikazane na idućoj slici 4.2.



Slika 4.2. *Instalacija Inertia.js opcije frontenda*

Nakon navedenog koraka svako pokretanje aplikacije izvršava se s php artisan serve naredbom. Tijekom razvojne faze aplikacije, poželjno je da se poslužena stranica kontinuirano ažurira s napravljenim promjenama te se takvo ponašanje omogućuje pokretanjem novog terminala i naredbom npm run watch.

Nakon instalacije Laravel Breeze-a, omogućena je registracija i prijava korisnika te je pružena osnovna tema aplikacije vidljiva na sljedećoj slici 4.3.



Slika 4.3. *Osnovna tema i prijava Laravel Breeze-a*

4.2. Dinamičko sučelje

Prijavom korisnika podaci se šalju poslužitelju gdje se provjerava točnost podataka. Nakon autentikacije korisnika, šalje ga se na naslovnu stranicu. Preusmjeravanjem na naslovnu stranicu kreira se zahtjev koji prolazi kroz DashboardController metodu koja odlučuje koja se stranica treba učitati. Uz učitavanje odgovarajuće stranice, klijentskoj strani se šalju i potrebni početni podaci o korisniku. Logika preusmjeravanja i slanje potrebnih podataka prikazano je sljedećim programskim kodom 4.4.

```
public function index(Request $request){
        $role = $request->user()->role->id;

        if($role == Role::ROLE_ADMIN){
            return Inertia::render('Dashboard/Admin');
        }
        else if($role == Role::ROLE_ORGANIZATION){
            $orgCv = $request->user()->orgCv;
            return Inertia::render('Dashboard/Organization',[
                'jobAds' => $orgCv?->jobAds
            ]);
        }
        else if($role == Role::ROLE_EMPLOYEE){
            $cv =  $request->user()->cv;
            $suggestedJobAds = null;
            if(isset($cv)){...}
```

```
            return Inertia::render('Dashboard/User',[
                'suggestedJobAds' => $suggestedJobAds
            ]);
        }
        return Inertia::render('Dashboard');
    }
```

Programski kod 4.4. *DashboardController preusmjeravanje*

Nakon preusmjeravanja korisnika na ispravnu stranicu, potrebno je učitati korisnikove podatke na klijentskoj strani i omogućiti njihovo korištenje u bilo kojem trenutku u aplikaciji. U svrhu toga koriste se zajedno HandleInertiaRequests middleware Laravela uz pomoć kojeg se u bilo kojem trenutku može pristupiti podacima sa poslužitelja i Vuex store koji služi kao spremište globalnih podataka na klijentu. Pruženi podaci middleware-a prikazani su sljedećim programskim kodom 4.5.

```
public function share(Request $request){
        return array_merge(parent::share($request), [
                'auth' => [
                        'user' => $request->user(),
                        'cv' => fn() => $request->user() ?
CV::getCurrentUserCV() : null,
                        'orgCv' => fn() => $request->user() ?
OrgCV::getCurrentUserCV() : null,
                ],
                'availableSkills' => CV\Skill::all(),
                'flash' => [
                        'status' => fn () => $request->session()->get('status')
                ],
                'ziggy' => function () {
                        return (new Ziggy)->toArray();
                },
        ]);
}
```

Programski kod 4.5. *HandleInertiaRequests dijeljeni podatci*

Nakon što se korisnika preusmjeri na ispravnu naslovnu stranicu, njegovi se podaci spremaju u Vuex spremište prije samog kreiranje naslovne stranice što je prikazano sljedećim programskim kodom 4.6.

```
<script>
import { usePage } from "@inertiajs/inertia-vue3";

export default {
    name: "AuthAdmin.vue",
    computed: {
        user() {
            return this.$store.getters.user;
        },
    },
    beforeCreate() {
        if (this.$store.getters.user === null) {
            this.$store.dispatch("setUser", usePage().props.value.auth.user);
        }
    },
};
```

```
</script>
```

Programski kod 4.6. *Spremanje podataka korisnika u Vuex store*

4.3. Kreiranje i uređivanje životopisa

Životopis kandidata je osnovni dio aplikacije. Na temelju kreiranog životopisa se kandidatu nude najpogodniji oglasi za posao, a poslodavcima se nude najbolji kandidati za posao. U ovom potpoglavlju prikazani su modeli, kontroleri i komponente potrebne za kreiranje, pregled i uređivanje životopisa.

Za pripremu kreiranja životopisa potrebno je pružiti već postojane podatke koje korisnik može birati. Iz razloga što se odabir vještina koristi na više dijelova web aplikacije, pružene su u HandleInertiaRequests middleware-u, prikazano programskim kodom 4.5. Postojane lokacije i vrste kontakta klijentu se prosljeđuju iz CVControllera uz stranicu koju treba učitati sljedećim programskim kodom 4.7.

```
function create(){
        $availableContacts = Contact::all();
        $availablePostcodes = Postcode::all();
        return Inertia::render('CV/Create', [
            'postcodes' => $availablePostcodes,
            'availableContacts' => $availableContacts,
        ]);
    }
```

Programski kod 4.7. *Slanje lokacija i vrsta kontakta web aplikaciji*

Poslane podatke također treba definirati na frontendu kako bi se mogli koristiti u ostatku aplikacije, što je prikazano sljedećim programskim kodom 4.8.

```
const props = defineProps({
    postcodes: {
        type: Array,
        required: true,
    },
    availableContacts: {
        type: Array,
        required: true,
    },
});
```

Programski kod 4.8. *Definiranje podataka koji će se primiti na frontendu*

Nakon dohvaćanja potrebnih podataka, korisniku se pruža obrazac koji ispunjuje. Tekstualni unosi podataka poput imena i prezimena, traženog posla, URL-a na profilnu sliku i ostalih omogućeno je uz text unosno polje. Uz njih, postoje polja koja imaju specifična ponašanja.

4.3.1. Unos adrese

Radi povećanja korisničkog iskustva i dodatne kontrole unosa, napravljena je komponenta za unos adrese. Na temelju upisanog poštanskog broja, web aplikacija treba automatski popuniti grad i

županiju. Navedena funkcionalnost ostvarena je na način da se korisniku prilikom unosa znamenki poštanskog broja nudi izbor poštanskih brojeva dobivenih s backenda, te klikom ili unosom cijelog poštanskog broja se polja grada i županije sama popunjuju. Navedena funckionalnost ostvarena je sljedećim programskim kodom 4.9.

```
updatePostcode(code) {
        this.$store.dispatch("updatePostcode", code);
},
fillAddress() {
        const postcode = this.postcodes.find(
                (postcode) => postcode.code.toString() === this.code
        );
        if (postcode) {
                this.updatePostcode(postcode.code);
                document.getElementById("postal_name").value = postcode.name;
                document.getElementById("county").value = postcode.county;
        }
}
```

Programski kod 4.9. *Automatsko popunjavanje polja grada i županije*

Osim automatskog popunjavanja, postojani poštanski brojevi se trebaju filtrirati na temelju upisanih znamenki, dok se tijekom praznog unosa ili unesenog ispravnog poštanskog broja ne treba ništa nuditi. Takvo ponašanje ostvareno je sljedećim programskim kodom 4.10.

```
searchedCodes() {
            if (
                this.code === "" ||
                this.postcodes.find(
                    (postcode) => postcode.code.toString() === this.code
                )
            ) {
                return null;
            }
            return this.postcodes.filter((postcode) =>
                postcode.code.toString().startsWith(this.code)
            );
        },
```

Programski kod 4.10. *Filtriranje postojanih poštanskih brojeva*

4.3.2. Unos kontakta

Korisniku treba omogućiti dodavanje proizvoljnog broja kontakta. Nakon dohvaćanja mogućih vrsta kontakta, klikom na gumb za novi kontakt, treba kreirati prazan objekt kako bi web aplikacija ponudila izbor vrste i polje za unos vrijednosti. Napravljena je komponenta ContactList koja u sebi sadrži listu ContactItem-a i gumb za dodavanje novog kontakta. Odabirom vrste kontakta ažurira se labela kojom korisnik zna kakvu vrijednost treba unijeti i mijenja se vrsta unosnog polja u HTML-u. Navedena funckionalnost napravljena je sljedećim programskim kodom 4.11.

```
contactInfo() {
        const contact = this.availableContacts.find(
                (contact) => contact.id === parseInt(this.contact.contact_id)
```

```
        );
        if (contact) {
                if (contact.name === "Email") {
                        return { label: "E-mail adresa", type: "email" };
                } else if (contact.name === "Phone") {
                        return { label: "Telefonski broj", type: "text" };
                } else {
                        return { label: "URL", type: "url" };
                }
        }
        return { label: "Vrijednost", type: "text" };
}
```

Programski kod 4.11. *Dohvaćanje labele i vrste unosnog polja kontakta*

4.3.3. Unos iskustva

Iskustvo je kompleksan objekt koji sadrži veći broj polja unosa. Kao što može biti proizvoljan broj kontakta, tako može biti i proizvoljan broj iskustava, unutar kojih može biti proizvoljan broj rezultata tog iskustva. Način dodavanja iskustava i rezultata je isti kao i kod kontakta. Klikom na gumb kreira se novi prazni objekt koji se nadodaje na već postojanu listu objekata. Korisniku preostaje samo ispuna novih polja. Klikom na gumb za kreiranje novog iskustva poziva se funkcija prikazana sljedećim programskim kodom 4.12.

```
addExperience(state) {
        state.cv.experiences.push({
                name: "",
                source: "",
                type: "",
                results: [""],
                started_at: null,
                finished_at: null,
        });
},
```

Programski kod 4.12. *Kreiranje novog praznog iskustva*

4.3.4. Dodavanje vještina

Dodavanje vještine radi na isti način kao dodavanje novog iskustva ili kontakta. Klikom na gumb kreira se novi objekt koji se nadodaje na već postojanu listu vještina. Kao što je već navedeno, lista postojanih vještina je dostupna na razini cijele web aplikacije putem HandleInertiaRequests middleware-a. Korisnik treba upisati vještinu koju ima, dok se tijekom unosa lista postojanih vještina filtrira na način sličan prikazan programskim kodom 4.10. Nakon odabira vještine, korisnik unosi razinu znanja te vještine.

Nakon što je korisnik unio sve podatke, podatci se šalju POST metodom CVControlleru na store metodu. Nakon dohvaćanja poslanog objekta na backendu, potrebno je odraditi validaciju poslanih podataka. Laravel nudi mogućnost kreiranja vlastitih validacijskih pravila i vlastitih validacijskih poruka. U slučaju da je došlo do greške tijekom validacije, korisnika se vraća na stranicu gdje je

unosio podatke uz poruku što nije ispravno. Proces validacije prikazan je sljedećim programskim kodom 4.13.

```
function store(Request $request) {
        $reqCV = $request->json()->all();
        $validator = Validator::make($reqCV, CV::VALID_RULES, CV::VALID_MSGS);
        if ($validator->stopOnFirstFailure()->fails()) {
                return Redirect::route('cv.create')->withErrors($validator);
        }
        CV::store($reqCV);
        return Redirect::route('cv.create')->with('status', 'Životopis
uspješno napravljen!');
}
```

Programski kod 4.13. *Proces validacije poslanog životopisa*

Tijekom kreiranja životopisa jako je bitno održavanje upisanih podataka. Kako web aplikacija koristi Vuex store, odlaskom na drugu stranicu i vraćanjem zadržavaju se uneseni podaci. Tako i nakon što korisnik pokuša predati nevaljani životopis, kada ga se vrati na stranicu kreiranja životopisa, sve unesene informacije ostaju. Kada korisnik pošalje ispravan životopis, poslani objekt se razlama na dijelove koji se spremaju u odgovarajuće tablice. Nakon spremanja, kao što je prikazano prijašnjim programskim kodom, korisnika se vraća natrag uz poruku da je životopis uspješno spremljen. Proces spremanja životopisa prikazan je sljedećim programskim kodom 4.14.

```
public static function store($CVJson, $userId = null){
        $user_id = Auth()->id();
        if($userId) {
                $user_id = $userId;
        }
        $cv = new self;
        // Create new CV
        $cv->user_id = $user_id;
        $cv->name = $CVJson['name'];
        $cv->img_url = $CVJson['img_url'];
        $cv->description = $CVJson['description'];
        $cv->street = $CVJson['street'];
        $cv->postcode = $CVJson['postcode'];
        $cv->sex = $CVJson['sex'];
        $cv->birthdate = new DateTime("{$CVJson['birthdate']}");
        $cv->years_of_exp = $CVJson['years_of_exp'];
        $cv->job = $CVJson['job'];
        $cv->references = $CVJson['references'];
        $cv->save();

        // Link new CV to authorized user
        $user = User::find($user_id);
        $user->cv_id = $cv->id;
        $user->save();

        // Fill in one-to-many relationships
        foreach($CVJson['contacts'] as $contact){
                $cvContact = new CV_Contact;
                $cvContact->cv_id = $cv->id;
                $cvContact->contact_id = $contact['contact_id'];
                $cvContact->value = $contact['value'];
```

```
                $cvContact->save();
        }

        foreach($CVJson['experiences'] as $experience){
                $cvExperience = new Experience;

                $cvExperience->cv_id = $cv->id;
                $cvExperience->type = $experience['type'];
                $cvExperience->name = $experience['name'];
                $cvExperience->source = $experience['source'];
                $cvExperience->results = $experience['results'];

                $started_at= new DateTime("{$experience['started_at']}");
                $cvExperience->started_at = $started_at;

                if($experience['finished_at']){
                        $finished_at = new
DateTime("{$experience['finished_at']}");
                        $cvExperience->finished_at = $finished_at;
                }

                $cvExperience->save();
        }

        foreach($CVJson['skills'] as $skill){
                $cvSkill = new CV_Skill;
                $cvSkill->cv_id = $cv->id;
                $cvSkill->skill_id = $skill['skill_id'];
                $cvSkill->proficiency = $skill['proficiency'];
                $cvSkill->save();
        }

        return $cv;
}
```

Programski kod 4.14. *Proces spremanja životopisa*

Kako se na temelju poslanog životopisa popunjava više tablica s 1:N vezom, uređivanje životopisa je teži postupak. Kako bi se postupak uređivanja tako kompleksnijeg objekta pojednostavio, uređivanje je ostvareno tako što se postojani životopis obriše, te se napravi novi na temelju poslanog objekta. Baza podataka je napravljena na način da ako se životopis obriše da se svi unosi 1:N veza automatski obrišu s njim. Takvo uređivanje je prikazano sljedećim programskim kodom 4.15.

```
public static function edit($CVJson, $user_id = null){
        // Editing CV is deleting CV and all associated entries
        // And then recreating CV and associated entries
        $cv = CV::find($CVJson['id']);
        if($cv == null){
                return;
        }

        // Unlink existing CV from current user
        $user = User::find(Auth()->id());
        $user->cv_id = null;
        $user->save();

        // Delete CV and all relationship entries
```

```
        // And create new CV
        $cv->delete();
        return CV::store($CVJson, $user_id);
}
```

Programski kod 4.15. *Proces uređivanja životopisa*

4.4. Kreiranje opisa tvrtke i oglasa za posao

Uloga poslodavca na web aplikaciji je stvaranje oglasa za posao na koje se kandidati mogu prijaviti. Kako poslodavac može napraviti više oglasa za posao, potrebno je prvobitno napraviti opis tvrtke na koji će budući oglasi za posao pokazivati stranim ključem u bazi podataka.

4.4.1. Opis tvrtke

Nakon registracije korisnika kao poslodavca, korisnik treba napraviti opis tvrtke. Takav obrazac sadrži osnovne informacije poput imena tvrtke i URL-a na profilnu sliku te lokaciju, kontakte i odjeljak 'o nama'. Unosi osim lokacije i kontakta su tekstualni unosi tako da nemaju specifično ponašanje. Način ponašanja obrasca i unos lokacije opisan je u potpoglavlju kreiranja životopisa te je prikazan programskim kodovima 4.9. i 4.10. Unos kontakta je također napravljen kao i kod kreiranja životopisa te je prikazan programskim kodom 4.11.

Za razliku od životopisa, opis tvrtke je drugi neobavezni strani ključ u tablici korisnika. Tako će kandidati imati strani ključ na životopis, dok će poslodavci imati strani ključ na opis tvrtke. Opisana tablica korisnika vidljiva je na slici 3.3. Tijekom kreiranja i spremanja opisa tvrtke koriste se model OrgCV i kontroler OrgCVController te prate sličan proces kreiranja i uređivanja opisa tvrtke kao što je prikazano programskim kodovima 4.13., 4.14. i 4.15. uz nedostatak iskustava i vještina.

4.4.2. Oglasi za posao

Nakon što poslodavac napravi opis tvrtke, omogućeno je stvaranje oglasa za posao. Oglasi za posao sadržavaju informacije o poslu, odgovornosti koje će budući zaposlenik imati i, što je najvažnije sa strane web aplikacije, parametre po kojima će se filtrirati kandidati. Detaljnost parametara po kojima se kandidati mogu filtrirati određena je na temelju koliko se uspješno mogu životopisi pretraživati. Kako su korisnici upisivali svoje datume rođenja, godine iskustva, lokaciju i vještine, to su parametri koje poslodavac može zadati tijekom kreiranja oglasa za posao. Dob i godine iskustva su numerički unosi pa nemaju specifično ponašanje, dok je odabir vještina jako sličan onome kada korisnik bira vještine za svoj životopis. Novo specifično ponašanje napravljeno je za lokaciju posla.

Unos lokacije

Za unos lokacije napravljena je nova komponenta LocationInput. Do sada se koristio način dohvaćanja grada i županije preko poštanskog broja, no sada je potrebno omogućiti neobavezna polja županije i grada. Nakon što se dohvate moguće lokacije s backenda, potrebno je korisniku pružiti pretraživanje na temelju tekstualnog unosa. Tijekom unosa županije, kako je više poštanskih brojeva povezano s jednom županijom, potrebno je vratiti sve jedinstvene županije kao ponuđeni izbor. Opisano ponašanje prikazano je sljedećim programskim kodom 4.16.

```
counties() {
        // get unique counties
        let counties = this.postcodes.map(function (value) {
                return value["county"];
        });
        let unique_counties = [...new Set(counties)];
        return unique_counties.map((v, i) => ({ name: v, index: i }));
}
```

Programski kod 4.16. *Dohvaćanje jedinstvenih županija*

U slučaju da je poslodavac odabrao županiju, odabrani grad mora pripadati toj županiji. U suprotnome, ako poslodavac nije odabrao županiju, potrebno je pružiti odabir svih jedinstvenih gradova. Opisana logika prikazana je sljedećim programskim kodom 4.17.

```
cities() {
        if (!this.jobCounty) {
                const cityArray = this.postcodes.map(function (value) {
                        return value["name"];
                });
                let unique_cities = [...new Set(cityArray)];
                return unique_cities.map((v, i) => ({ name: v, index: i }));
        } else {
                let countyCities = this.postcodes.filter(
                        (el) => el.county === this.jobCounty
                );
                countyCities = countyCities.map(function (value) {
                        return value["name"];
                });
                let unique_cities = [...new Set(countyCities)];
                return [
                        ...new Set(
                                unique_cities.map((v, i) => ({ name: v, index:
i }))
                        ),
                ];
        }
}
```

Programski kod 4.17. *Dohvaćanje odgovarajućih gradova*

Spremanje oglasa za posao prvo treba proći kroz validaciju objekta nakon čega se sprema u bazu, slično prikazanome programskim kodom 4.13.

4.5. Pretraga korisnika

Nakon stvaranja oglasa za posao, potrebno je filtrirati životopise prema navedenim parametrima. Da se kandidat uzme u obzir, mora prvobitno odgovarati lokaciji, dobi i godinama iskustva traženim na oglasu za posao. Pretraga korisnika prema navedenim parametrima prikazan je sljedećim programskim kodom 4.18. Iz razloga što je potreban detaljniji upit na bazu od onih koji su ponuđeni Laravelovim Eloquent ORM-om, napisan je direktan upit na bazu podataka.

```
$targets = [];
$locationClauses = [];
if(isset($options['county'])){
	array_push($locationClauses, ['postcodes.county', '=',
$options['county']]);
}
if(isset($options['city'])){
	array_push($locationClauses, ['postcodes.name', '=',
$options['city']]);
}
$cvs = DB::table('cvs')
	->join('postcodes', 'cvs.postcode', '=', 'postcodes.code')
	->select('cvs.*', 'postcodes.name as postcodes.name',
'postcodes.county as postcodes.county')
	->where([
		['cvs.birthdate', '<', Carbon::now()-
>subYears($options['minAge'])],
		['cvs.birthdate', '>', Carbon::now()-
>subYears($options['maxAge'])],
		['cvs.years_of_exp', '>=', $options['minExp']],
		['cvs.years_of_exp', '<=', $options['maxExp']],
		...$locationClauses
	])->get()->toArray();
```

Programski kod 4.18. *Filtriranje životopisa prema lokaciji, dobi i godinama iskustva*

Nakon dohvaćenih odgovarajućih kandidata, poslodavcu treba prikazati koje vještine kandidat ima, koje nema i postotak preklapanja traženih vještina. U slučaju da veći broj korisnika ima isti postotak odgovarajućih vještina, dodatno ih se sortira prema prosječnim ocjenama tih vještina. Opisana funkcionalnost ostvarena je sljedećim programskim kodom 4.19.

```
foreach($cvs as $cv){
	// Calculate ratio of found user skills and all user'\s skills.
	// Calculate average proficiency level among found skills.
	$userSkills = $cv->skillProficiencies->toArray();
	$foundSkills = $cv->skillProficiencies->whereIn('skill_id',
$searchedSkills)->all();
	$targetSkills = [
		'skillRatio' => count($foundSkills)/count($userSkills),
		'avgProficiency' => array_sum(array_column($foundSkills,
'proficiency')) / (count($foundSkills) ?: 1),
		'skills' => array_column($foundSkills, 'skill_id')
	];
	if(count($targetSkills['skills'])){
		array_push($targets, $cv->toArray() + $targetSkills);
	}
}
```

```
// Sort by amount of found skill / amount of searched skills
usort($targets, function ($a, $b) {
        if($a['skillRatio'] != $b['skillRatio']){
                return $a['skillRatio'] < $b['skillRatio'];
        }
        else{
                return $a['avgProficiency'] < $b['avgProficiency'];
        }
});
```

Programski kod 4.19. *Sortiranje životopisa prema postotku odgovarajućih vještina ili prosječnih ocjena*

Nakon opisanog filtriranja, dohvaćeni su poredani najbolji kandidati za napravljeni oglas za posao.

4.6. Preporučeni poslovi

Tijekom dohvaćanja kandidata za oglas za posao, pretražuju se životopisi kandidata prema određenim parametrima. U slučaju dohvaćanja preporučenih poslova, potrebno je pretražiti oglase za posao koji odgovaraju korisnikovom životopisu. Iako je ista logika, izračun postotka poklapajućih vještina i sortiranje oglasa za posao prema tom postotku napisana je sljedećim programskim kodom 4.20.

```
$intersectSkillIds = array_fill(0, count($jobAdSkillIds), $skillIds);
$matchingSkillIds = array_map(function ($jobAdSkillIds, $skillIds) {
        return array_intersect($jobAdSkillIds, $skillIds);
}, $jobAdSkillIds, $intersectSkillIds);

$differentSkillsIds = array_map(function ($jobAdSkillIds, $skillIds) {
        return array_diff($jobAdSkillIds, $skillIds);
}, $jobAdSkillIds, $intersectSkillIds);

usort($suggestedJobAds, function($a, $b) {
        $matchPercentageFirst = count($a["skills"]["matchingSkills"]) / (
count($a["skills"]["matchingSkills"]) +
count($a["skills"]["differentSkills"]));
        $matchPercentageSecond = count($b["skills"]["matchingSkills"]) / (
count($b["skills"]["matchingSkills"]) +
count($b["skills"]["differentSkills"]));

        return $matchPercentageFirst < $matchPercentageSecond;
});
```

Programski kod 4.20. *Sortiranje oglasa za posao prema postotku odgovarajućih vještina*

Navedenim kodom se za kandidata sortiraju oglasi za posao koji najviše odgovaraju njegovim vještinama. Zabilježavaju se koje vještine zadovoljava, koje nedostaju te postotak preklapanja traženih vještina.

4.7. Prijava na oglas za posao

Nakon sortiranja oglasa za posao, kandidatu je potrebno omogućiti prijavljivanje na njih. Tako se uz dohvaćanje preporučenih oglasa za posao vraća i unos u tablicu prijava za tog kandidata i taj oglas za posao, u slučaju da postoji. Navedena veza tablica prikazana je na slici 3.7. Konačno

dohvaćanje preporučenih oglasa za posao uz stanje prijave prikazano je sljedećim programskim kodom 4.21.

```
$suggestedJobAds = [];
foreach(array_keys($matchingJobAds) as $index){
        $jobAdInfo = JobAd::with(['orgCv' => function ($query) {
                $query->select('id', 'name', 'img_url');
        }])->select(['id','name','org_c_v_s_id'])-
>find($jobAds[$index]['id'])->toArray();
        $jobAd = array(
                "jobAd" => $jobAdInfo,
                "skills" => $matchingJobAds[$index],
                "isApplied" => Application::where('job_id', $jobAdInfo['id'])-
>where('user_id', $request->user()->id)->first()
        );
        $suggestedJobAds[] = $jobAd;
}
```

Programski kod 4.21. *Dohvaćanje preporučenih oglasa za posao uz prijavu na njih*

Nakon što se kandidat prijavi na oglas za posao, stvara se unos u tablicu prijava. Tako poslodavac kada ode na kreirani oglas za posao može vidjeti sve prijavljene kandidate. Dohvaćanje prijavnika i njihovih životopisa prikazano je sljedećim programskim kodom 4.22.

```
$applications = Application::where('job_id', $job_id)->get()->toArray();
$users = CV::with(['location','skillProficiencies'])-
>whereIn('user_id',array_column($applications, 'user_id'))->get()->toArray();
$result = [];
for($i = 0; $i < count($applications); $i++) {
        $result[] = array(
                "application" => $applications[$i],
                "user" => $users[$i]
        );
}
return response()->json($result);
```

Programski kod 4.22. *Dohvaćanje prijavnika na oglas za posao*

4.8. Generiranje lažnih podataka

Iz razloga što bi testiranje web aplikacije zahtjevalo da se napravi više kandidata sa smislenim podatcima i više poslodavaca s oglasima za posao, korištena je biblioteka FakerPHP za generiranje lažnih podataka. Uz kreiranje baze podataka moguće je popuniti bazu proizvoljnim brojem životopisa i oglasa za posao na temelju definiranih predložaka. Takvi predlošci opisuju se u klasama tvornica za određeni model.

Tako se kreiranje životopisa odvija tako da se prvo napravi korisnik kandidat, za svakog korisnika se napravi životopis s kojim se povezuju kontakti, iskustva i vještine. Potreban kod za ostvarivanje navedenog prikazano je kroz sljedeće programske kodove 4.23, 4.24.

```
User::factory($employeeCount)->create();
// $creations of users, but only last 10 are companies
CV::factory($employeeCount)
```

```
        ->has(CV_Contact::factory()
                ->count(1)
                ->email(), 'contacts'
        )
        ->has(CV_Contact::factory()
                ->count(1)
                ->phone(), 'contacts'
        )
        ->has(CV_Contact::factory()
                ->count(1)
                ->url(), 'contacts'
        )
        ->has(CV\Experience::factory()
                ->count(3)
                ->linked(), 'experiences'
        )
        ->create();
```

Programski kod 4.23. *Kreiranje korisnika, životopisa, kontakta i iskustva*

```
return $this->afterCreating(function ($CV){
                // Link the newly created CV to an existing user
                $user = User::find($CV['user_id']);
                $user->cv_id = $CV->id;
                $user->save();
                // Fill the many-to-many relationship with cv_skill table
                $randArray = range(1,CV\Skill::all()->count());
                shuffle($randArray);
                for($i = 0; $i < 20; $i++){
                        $cv_skill = new CV_Skill;
                        $cv_skill->cv_id = $CV->id;
                        $cv_skill->skill_id = $randArray[$i];
                        $cv_skill->proficiency = rand(1,5);
                        $cv_skill->save();
                }
        });
```

Programski kod 4.24. *Kreiranje vještina kandidata*

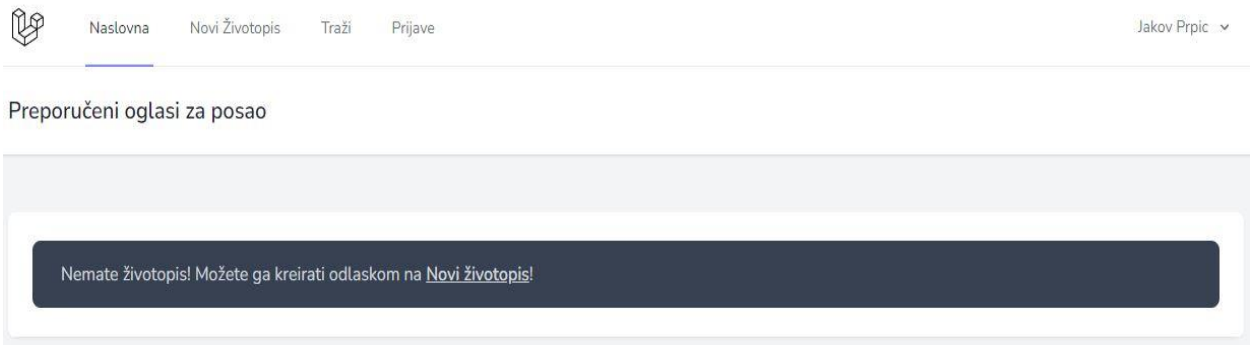
Generiranje poslodavaca, opisa tvrtki i oglasa za posao odvija se na sličan način. Nakon postavljanja navedenih predložaka, popunjavanje baze podataka tijekom kreiranja obavlja se naredbom php artisan migrate:fresh –seed.

5. ISPITIVANJE PROGRAMSKOG RJEŠENJA I KORIŠTENJE

U ovom poglavlju prikazano je korištenje web aplikacije. Kako postoje tri role korisnika, poglavlje je podijeljeno na korisničko iskustvo svake role.

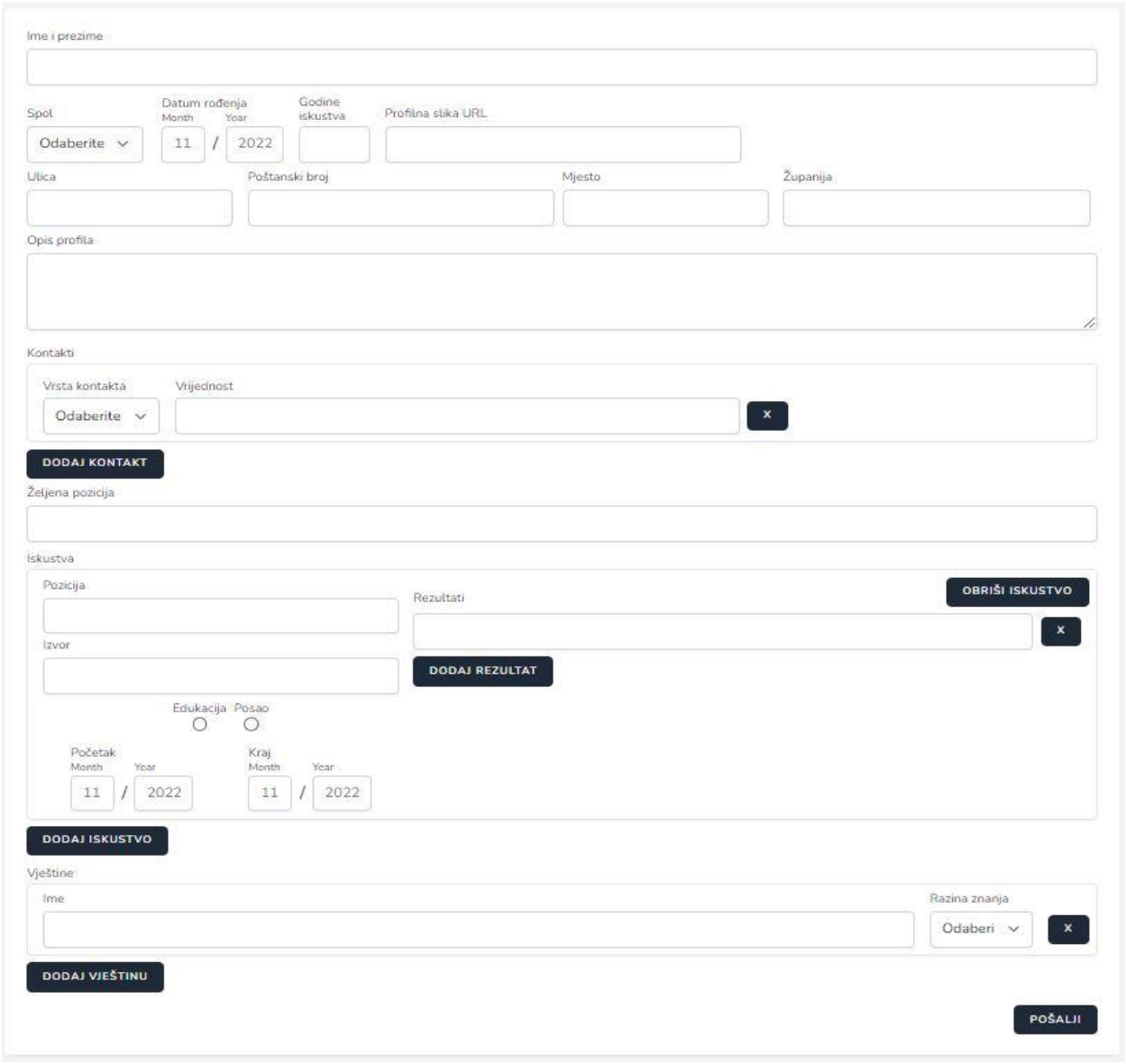
5.1. Korisničko iskustvo kandidata

Korisničko iskustvo kandidata podijeljeno je korake od kreiranja životopisa do prijave na ponuđene oglase. Nakon što se kandidat prijavi potrebno je osigurati da napravi životopis. Odmah nakon prijave, učitana je početna stranica na kojoj se nalazi poruka s poveznicom na stranicu kreiranja životopisa, prikazano na sljedećoj slici 5.1.



Slika 5.1. *Početna stranica kandidata*

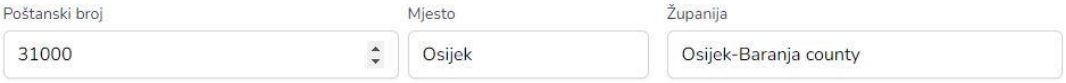
Klikom na navedenu poveznicu ili klikom na 'Novi životopis' u navigaciji prikazan je obrazac za stvaranje životopisa prikazan na sljedećoj slici 5.2.



Slika 5.2. *Obrazac za novi životopis*

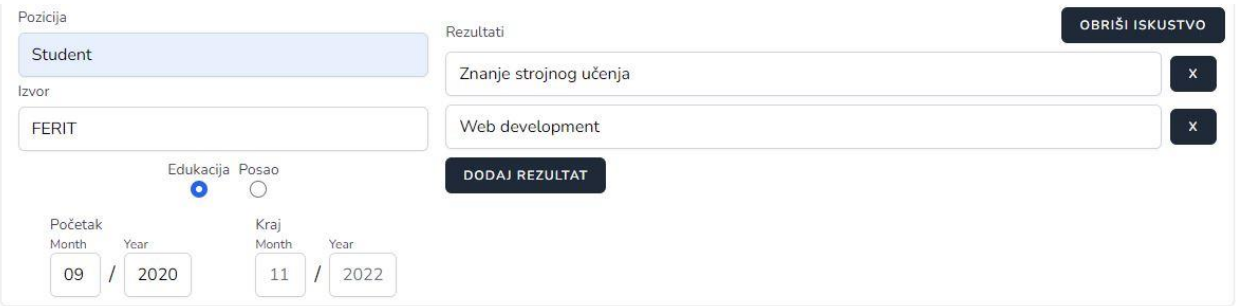
Kao što je navedeno u prijašnjem poglavlju, postoje polja unosa koja imaju specifična ponašanja. Prva takva polja su poštanski broj koji određuje grad i županiju korisnika. Tijekom unosa znamenki poštanskog broja korisniku se nudi automatsko popunjavanje podataka. Opisano ponašanje prikazano je sljedećim slikama 5.3 a) i 5.3 b).

Slika 5.3. a) *Pretraživanje poštanskih brojeva prema korisničkom unosu*



Slika 5.3. b) *Automatsko popunjavanje polja grada i županije*

Kako korisnik može unositi iskustva, potrebno je omogućiti unos iskustva koji se i dalje događa. Sljedećom slikom 5.4. prikazano je kako se izostavljanjem datuma završetka unosi takvo iskustvo.

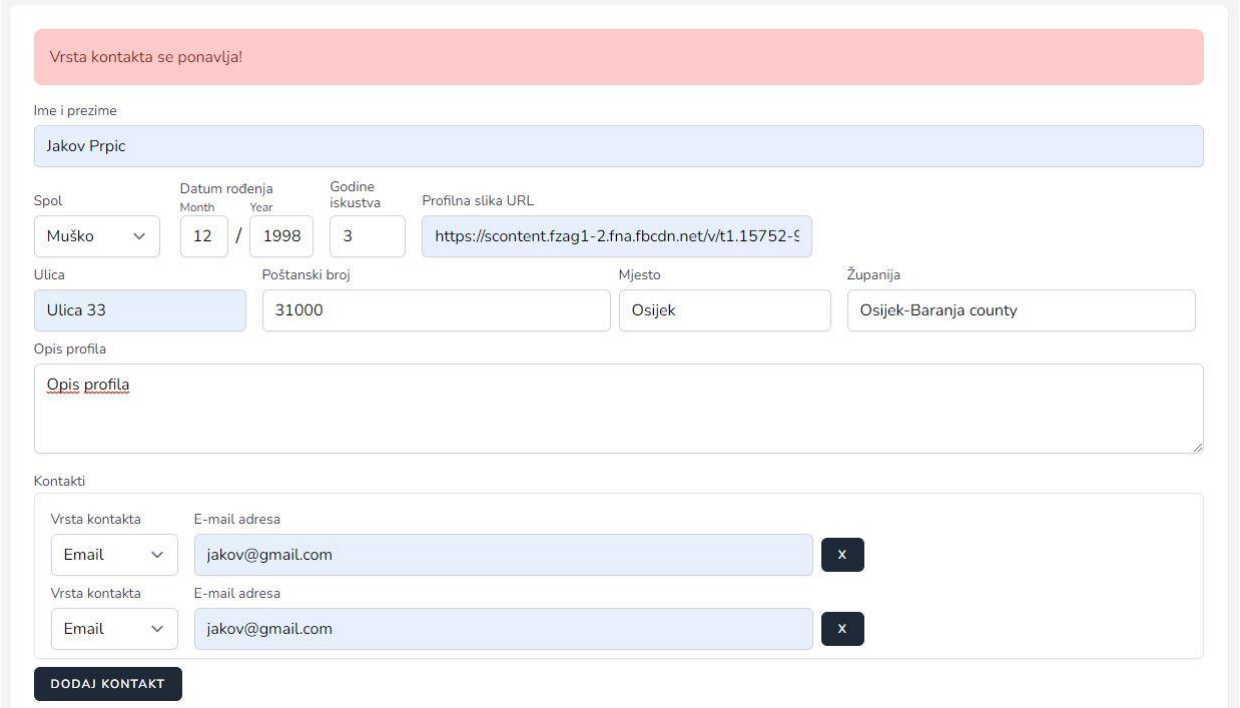


Slika 5.4. *Unos iskustva koje se i dalje odvija*

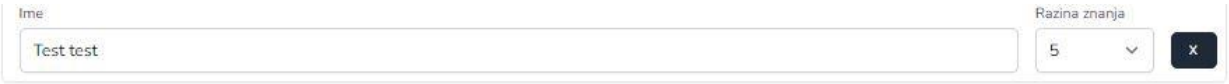
Sljedeće polje s automatskim popunjavanjem je polje vještina. Kako korisnik može odabrati samo postojane vještine, upisivanjem vještine nudi mu se izbor postojanih vještina na temelju upisanih slova. Pretraživanje vještina prikazano je sljedećom slikom 5.5.

Slika 5.5. *Izbor postojanih vještina*

Nakon popunjavanja obrasca, klikom na gumb Pošalji se pokušava životopis spremiti. Prvi korak tog procesa je validacija unešenog životopisa. Kao primjeri grešaka koje se mogu dogoditi su duplicirani kontakti i upis nepostojane vještine. Na sljedećim slikama 5.6. a), 5.6 b), 5.6 c) prikazane su greške i unos kojim dolazi do greški.



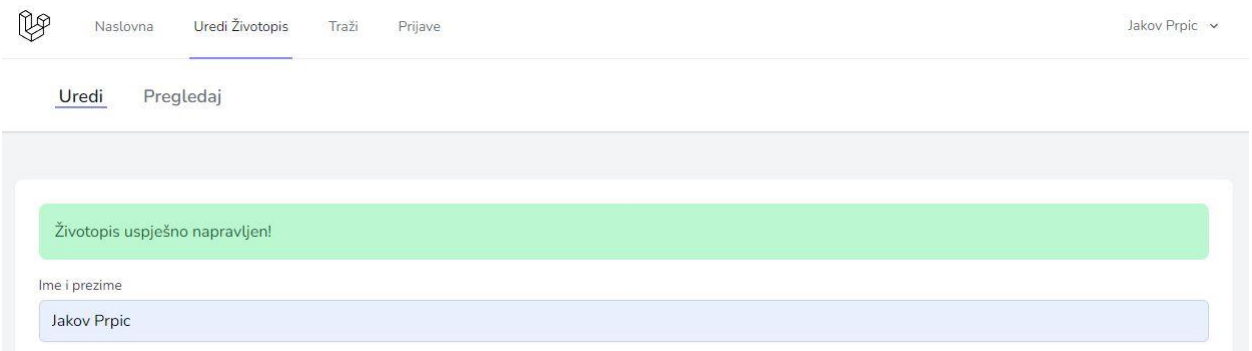
Slika 5.6. a) *Pogrešan unos kontakta i javljanje greške*



Slika 5.6. b) *Unos nepostojane vještine*

Slika 5.6. c) *Greška za unos nepostojane vještine*

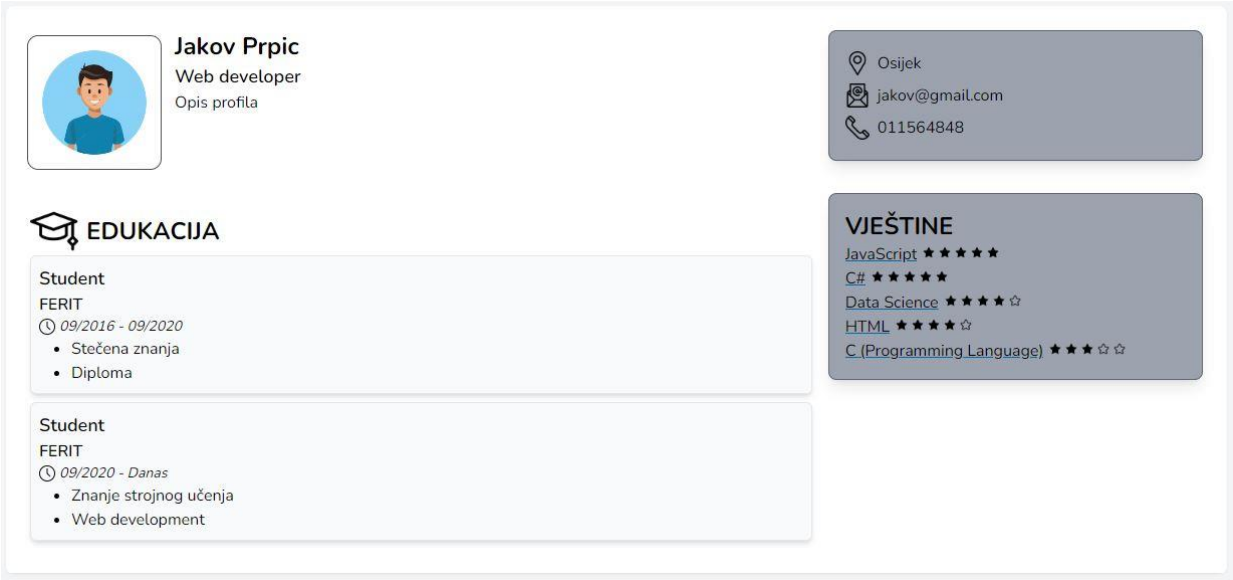
U slučaju da korisnik preda ispravne podatke za stvaranje novog životopisa ili uređivanje postojanog, korisniku se javlja pozitivna povratna informacija prikazana na sljedećim slikama 5.7. a) i 5.7. b).



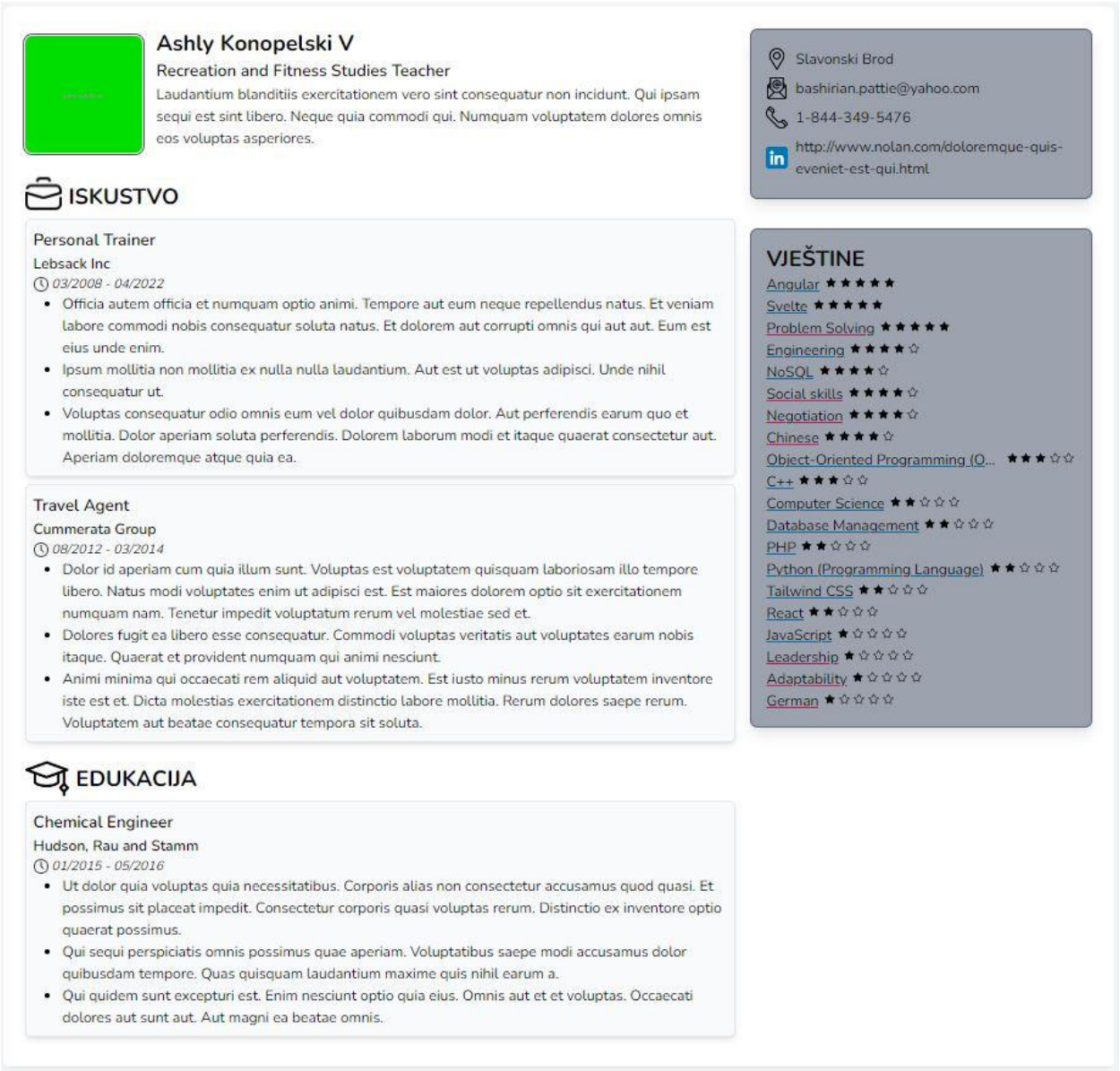
Slika 5.7. a) *Uspješno kreiranje životopisa*

Slika 5.7. b) *Uspješno uređivanje životopisa*

Nakon što je kandidat kreirao svoj životopis, životopis na temelju unesenih podataka na prijašnjim slikama prikazano je sljedećom slikom 5.8. a), dok je detaljniji životopis koji je generiran lažnim podacima prikazan slikom 5.8. b).

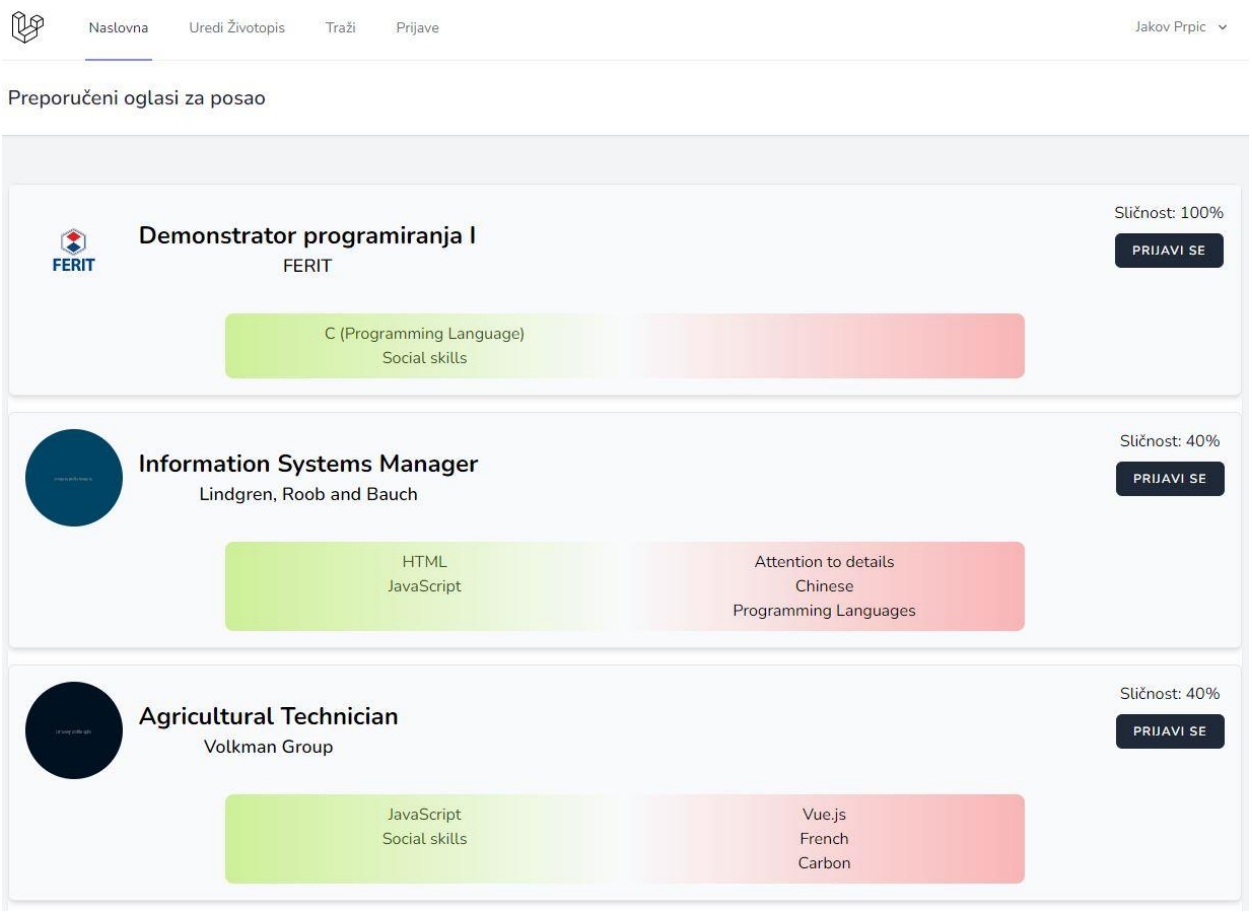


Slika 5.8. a) *Stvoreni životopis na temelju prijašnje unesenih podataka*



Slika 5.8. b) *Životopis generiran lažnim podacima*

Nakon kreiranja životopisa kandidat ima vještine na temelju kojih mu se mogu prikazati preporučeni oglasi za posao. Preporučeni oglasi za posao na naslovnoj stranci prikazani su sljedećom slikom 5.9.



Slika 5.9. *Preporučeni oglasi za posao*

Kao što je prikazano na prijašnjoj slici 5.9. kandidat se može prijaviti na oglase za posao. Moguća stanja prijave kandidata na oglas za posao prikazani su sljedećim slikama 5.10.a), 5.10.b), 5.10.c) i 5.10.d).

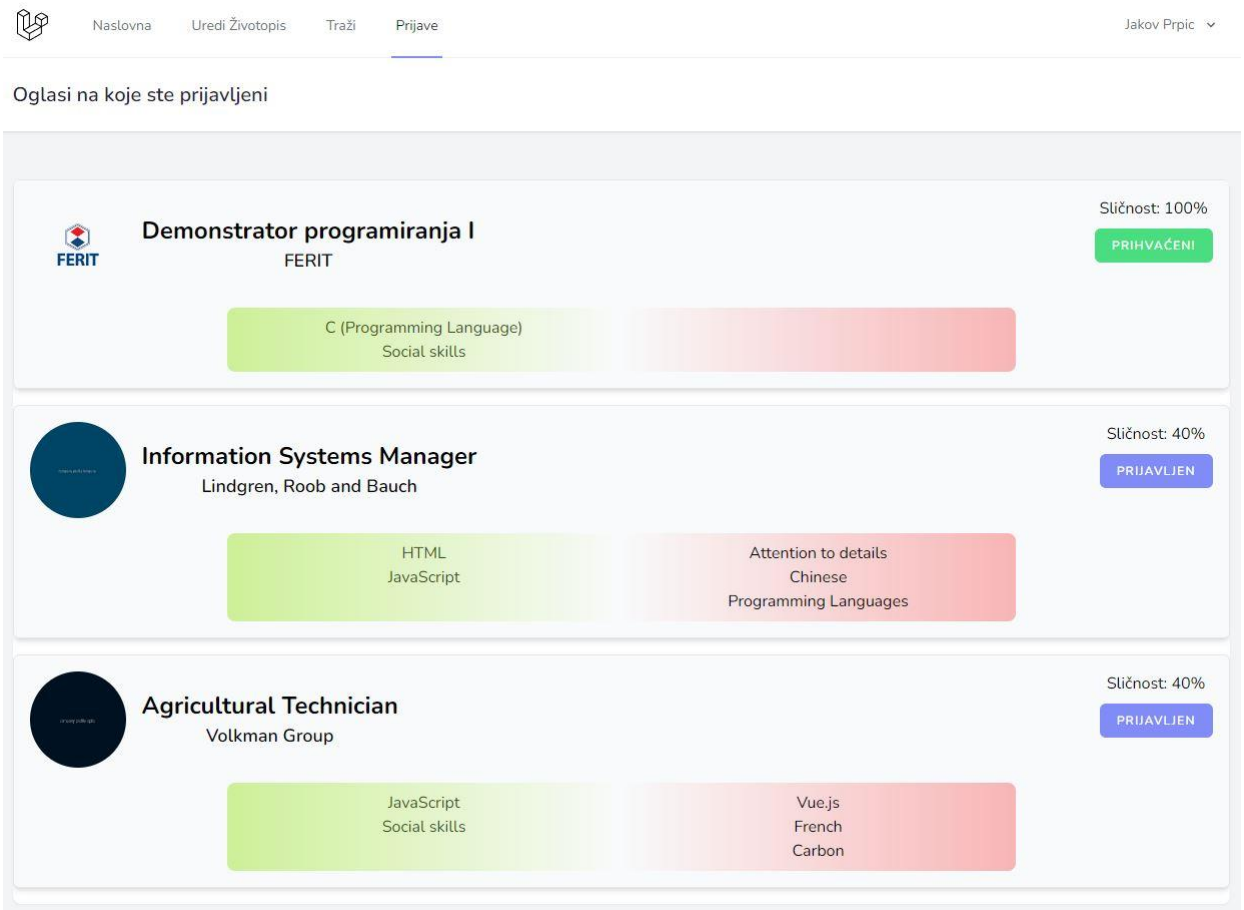
Slika 5.10.a) *Oglas za posao na koji se kandidat nije prijavio*

Slika 5.10.b) *Oglas za posao na koji se kandidat prijavio*

Slika 5.10.c) *Oglas za posao na kojem je kandidat prihvaćen*

Slika 5.10.d) *Oglas za posao na kojem je kandidat odbijen*

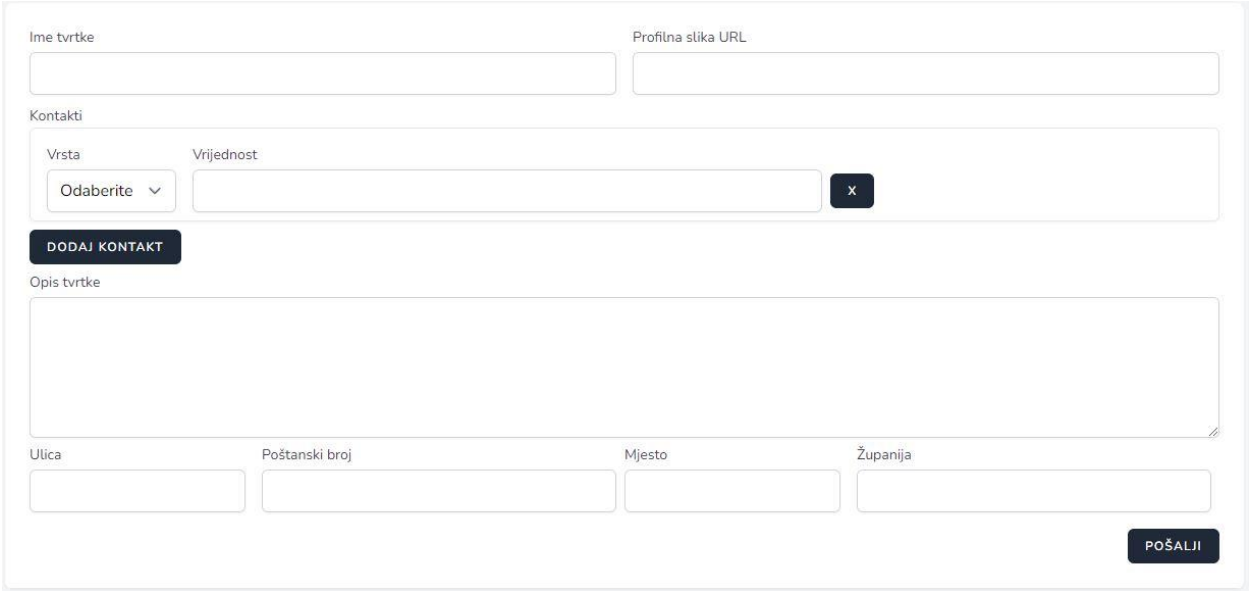
Uz prikaz preporučenih poslova na koje se kandidat može prijaviti, također je napravljena stranica na kojoj može pregledati sve svoje prijave, prikazano na sljedećoj slici 5.11.



Slika 5.11. *Prikaz oglasa za posao na koje je kandidat prijavljen*

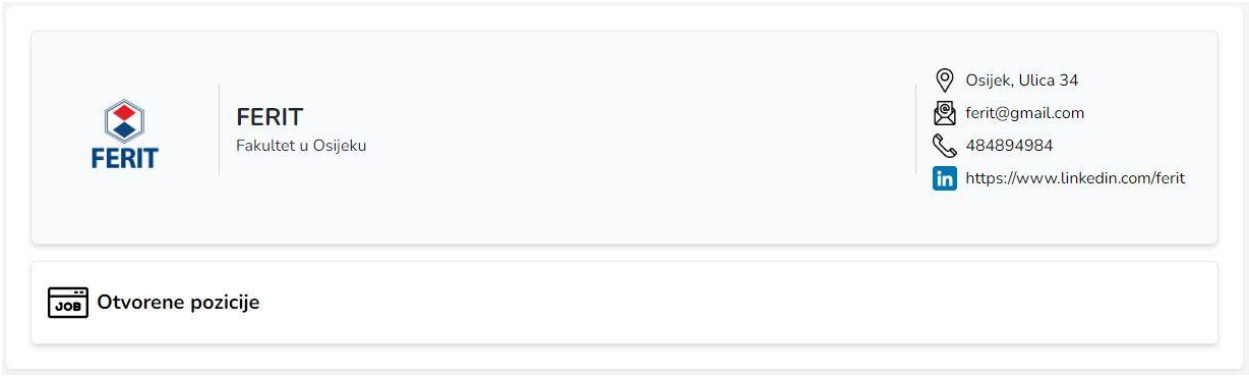
5.2. Korisničko iskustvo poslodavca

Korisničko iskustvo poslodavca podijeljeno je na korake od kreiranja opisa tvrtke do kreiranja oglasa za posao i prihvaćanja prijavnika. Nakon što se poslodavac prijavi potrebno ga je preusmjeriti na stranicu na kojoj kreira opis tvrtke. Način preusmjeravana obavljen je porukom na naslovnoj stranici kao na slici 5.1. Nakon dolaska poslodavca na stranicu kreiranja opisa tvrtke pružen je obrazac prikazan sljedećom slikom 5.12.



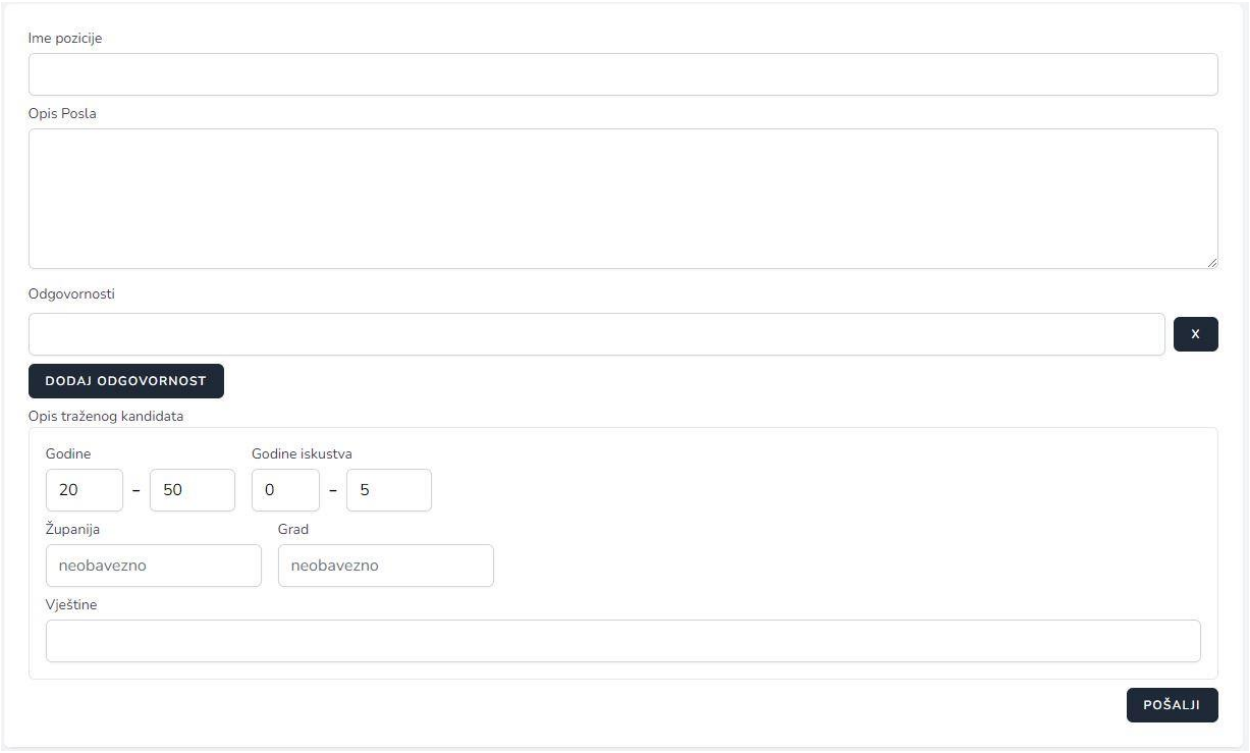
Slika 5.12. *Obrazac za kreiranje opisa tvrtke*

Unos kontakta i adrese napravljen je na isti način kao što tijekom kreiranja životopisa. Također način javljanja povratne informacije o uspješnosti validacije opisa tvrtke je napravljen isto. Pregled kreiranog opisa tvrtke prikazan je sljedećom slikom 5.13.



Slika 5.13. *Prikaz kreiranog opisa tvrtke*

Sljedeći korak je kreiranje oglasa za posao. Povratkom na naslovnu stranicu poslodavcu je pružena nova poruka za preusmjeravanje na obrazac za kreiranje oglasa za posao prikazan sljedećom slikom 5.14.

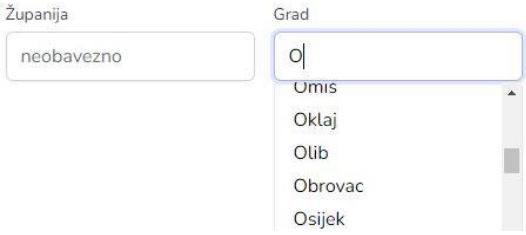


Slika 5.14. *Obrazac za kreiranje oglasa za posao*

Obrazac za oglas za posao ima novu vrstu unosa lokacije na temelju županije i grada. Izbor županije i grada obavljen je kao prijašnje prikazan način filtriranja postojanih podataka prilikom unosa slova. U slučaju da korisnik unese županiju, gradovi su filtrirani na temelju te županije, dok su bez županije ponuđeni svi gradovi. Opisano ponašanje prikazano je slikama 5.15.a) 5.15.b) i 5.15.c)

Slika 5.15.a) *Filtriranje županija*

Slika 5.15.b) *Filtriranje gradova uz odabranu županiju*

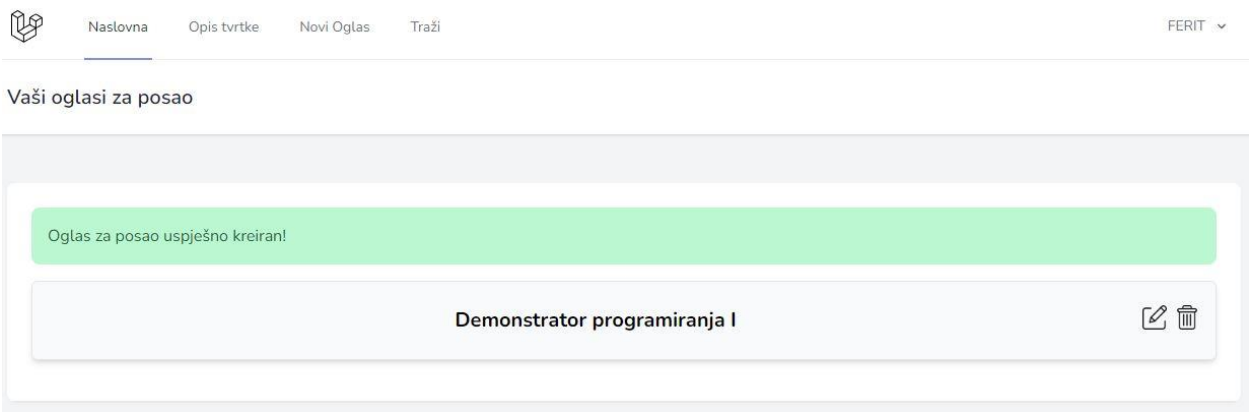


Slika 5.15.c) *Filtriranje gradova bez upisane županije*

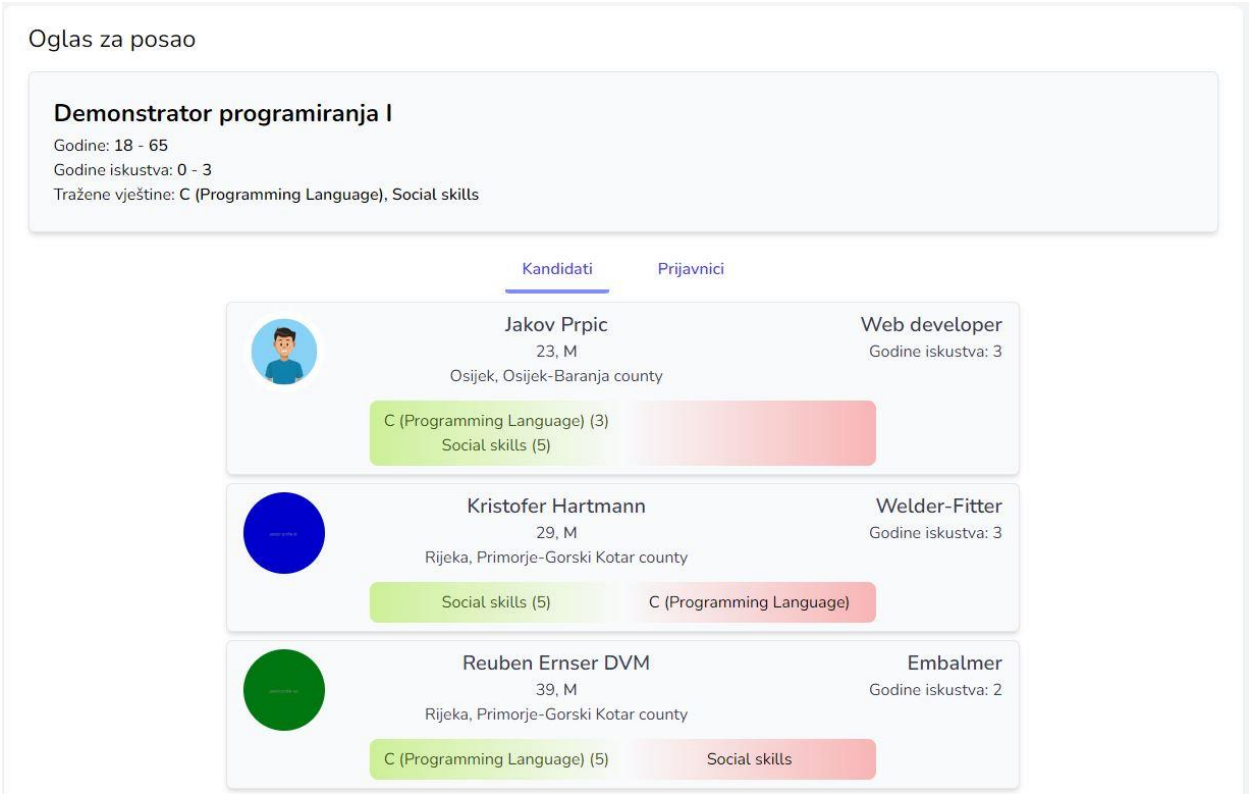
Odabir traženih vještina obavljen je kao i tijekom kreiranja životopisa dok je način prikazivanja odabranih vještina napravljen na način prikazan sljedećom slikom 5.16.

Slika 5.16. *Odabir traženih vještina*

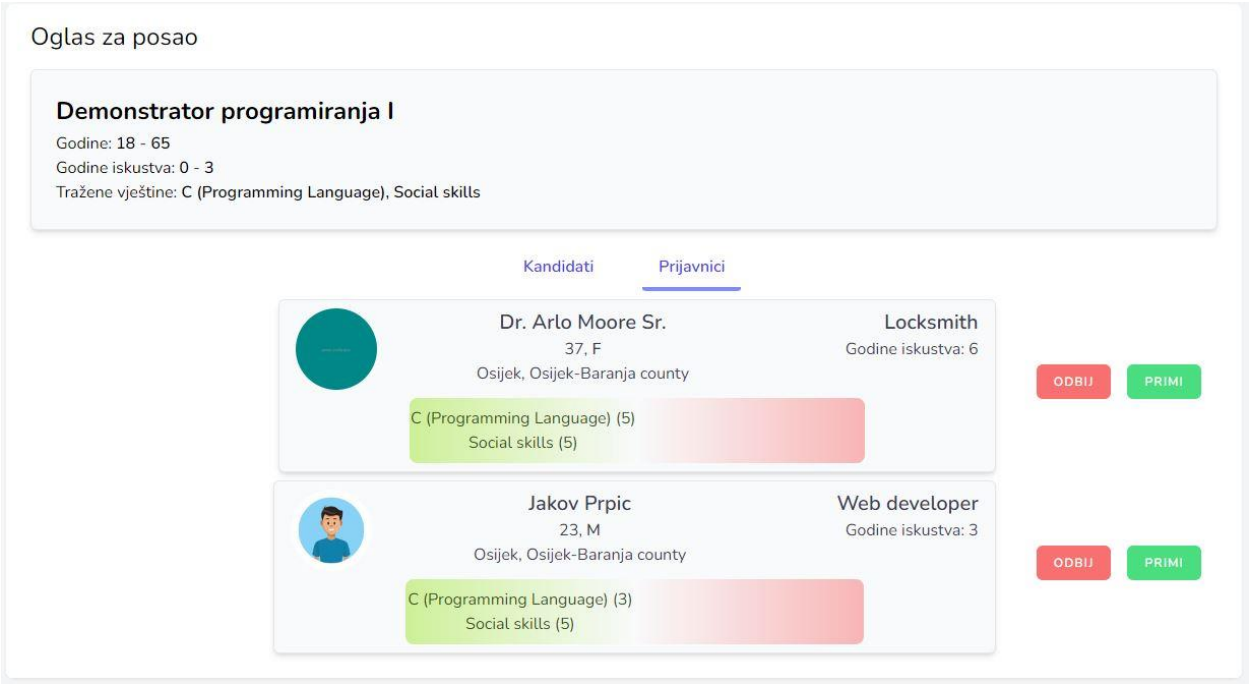
Nakon uspješnog kreiranja oglasa za posao, odlaskom na naslovnu stranicu poslodavcu su izlistani napravljeni oglasi za posao. Klikom na jedan od izlistanih oglas pruža se mogućnost pregleda kandidata i prijavnika za taj oglas, prikazano na sljedećim slikama 5.17.a), 5.17.b) i 5.17.c).



Slika 5.17.a) *Naslovna stranica s izlistanjem kreiranih oglasa za posao*

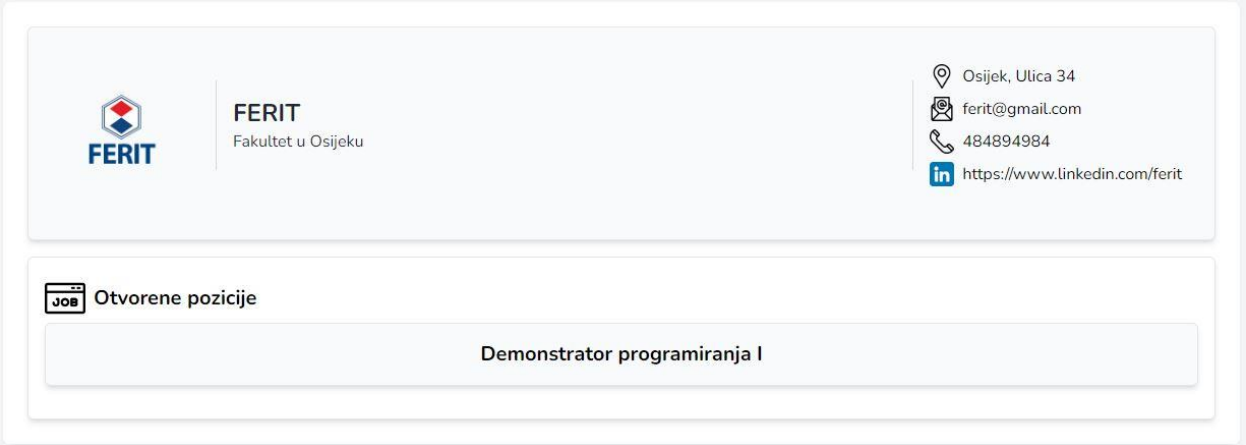


Slika 5.17.b) *Prikaz kandidata za oglas za posao*



Slika 5.17.c) *Prikaz prijavnika na oglas za posao*

Također, posjećivanjem opisa tvrtke mogu se posjetiti njihovi oglasi za posao. Uz to, omogućen je detaljniji pregled oglasa za posao koji mogu vidjeti i korisnici koji nisu prijavljeni. Navedeno je prikazano sljedećim slikama 5.18.a), 5.18.b)



Slika 5.18.a) *Prikaz oglasa za posao na pregledu opisa tvrtke*

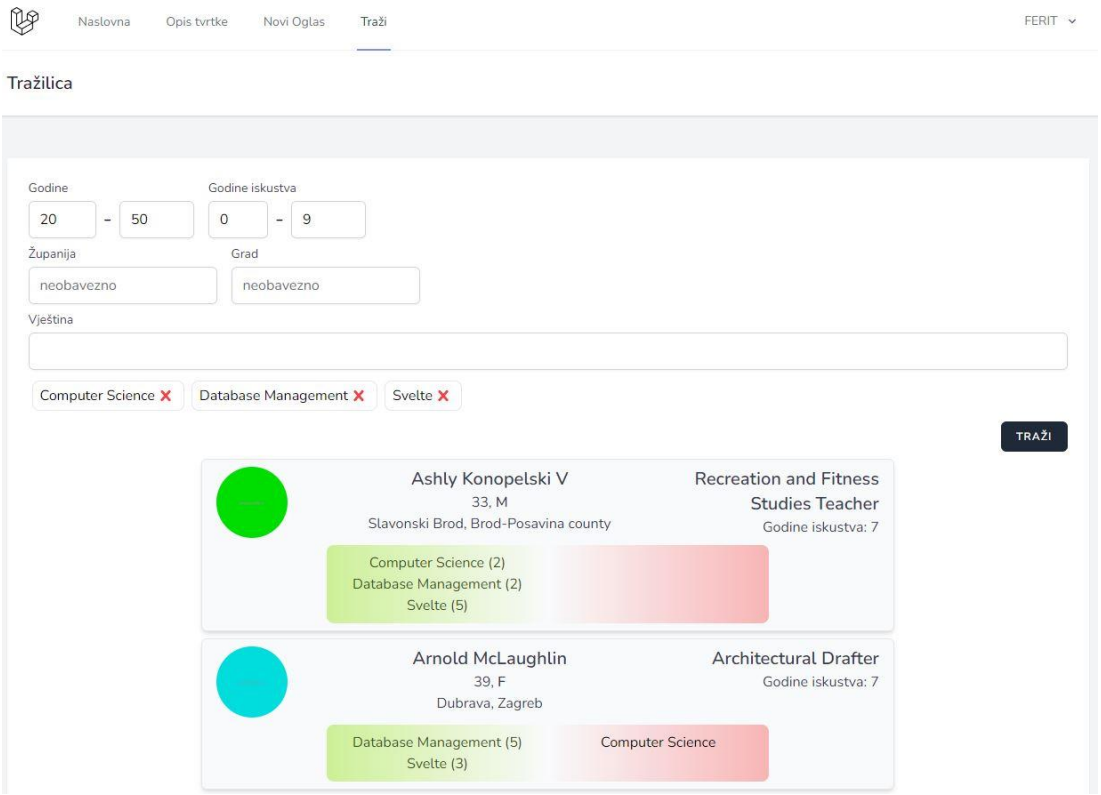
Slika 5.18.b) *Detaljniji prikaz oglasa za posao*

Za kraj, prijavnike je moguće odbiti ili prihvatiti za određeni oglas za posao. Klikom na jedan od gumbova se mijenja boja gumba, te se mijenja status prijave u bazi podataka. Odbijanje i prihvaćanje prijavnika prikazano je sljedećim slikama 5.19.a) i 5.19.b).

Slika 5.19.a) *Odbijanje prijavnika*

Slika 5.19.b) *Prihvaćanje prijavnika*

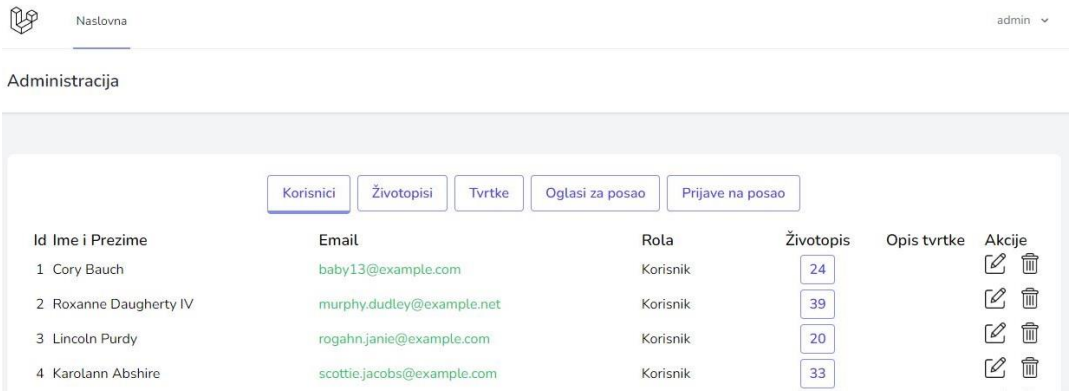
Osim pretrage korisnika preko stvaranja oglasa za posao, postoji tražilica za ručno pretraživanje korisnika prema istim parametrima kao u oglasu za posao. Primjer ručnog traženja kandidata prikazan je sljedećom slikom 5.20.



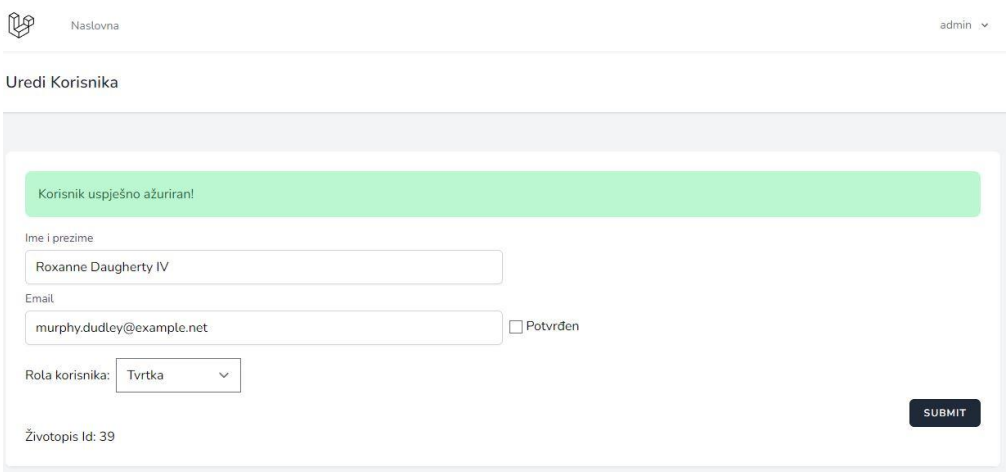
Slika 5.20. *Tražilica i odgovarajući kandidati*

5.3. Administracijsko sučelje

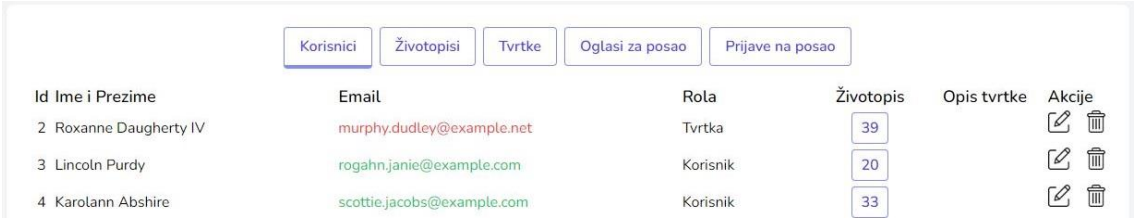
Za korisnika s administracijskim privilegijama napravljeno je sučelje s prikazom svih podataka uz pravo na uređivanje i brisanje svih podataka. Tablica korisnika, uređivanje jednog i povratna informacija prikazano sljedećim slikama 5.21.a), 5.21.b). i 5.21.c).



Slika 5.21.a) *Prikaz tablice korisnika*



Slika 5.21.b) *Prikaz uređivanja korisnika*



Slika 5.21.c) *Prikaz korisnika nakon uređivanja*

Kako su tablice povezane s različitim vezama, potrebno je paziti na to da se brisanjem kandidata obriše i njegov životopis, a brisanjem poslodavca da se obriše opis tvrtke i odgovarajući oglasi za posao. Ostale tablice životopisa, opisa tvrtki, oglasa za posao i prijava na posao prikazane su sljedećim slikama 5.22.a), 5.22.b), 5.22.c) i 5.22.d).

Slika 5.22.a) *Prikaz životopisa*

Slika 5.22.b) *Prikaz opisa tvrtki*

Slika 5.22.c) *Prikaz oglasa za posao*

Slika 5.22.d) *Prikaz prijava na posao*

6. ZAKLJUČAK

Kako tijekom kandidiranja za posao osoba treba predati životopis koji treba biti kontinuirano ažuriran i čitljiv, a poslodavac stvaranjem oglasa za posao treba privući pažnju traženim ljudima, razvijena je web aplikacija koja bi uvelike ubrzala i olakšala taj proces. Definirani su zahtjevi na aplikaciju te su uzeti u obzir koraci koji bi standardizirali životopise, poboljšali korisničko iskustvo i ubrzali proces stvaranja životopisa. Uz to, predložena je arhitektura na temelju koje bi se podaci sa životopisa spremali na takav način da poslodavci mogu lakoćom pretražiti proizvoljan broj životopisa i pronaći kandidate koji najviše odgovaraju predloženom oglasu za posao. Osim samog stvaranja životopisa kandidatima su predstavljeni oglasi za posao najbliži stvorenom životopisu. Na isti način su poslodavcima predstavljeni najbolji kandidati za određeni oglas za posao. Također, omogućena je prijava kandidata na oglas, te poslodavcu prihvaćanje i odbijanje prijavnika. Rješenje je ostvareno uz tehnologije Laravel i Vue.js čija kombinacija omogućuje glatko i reaktivno korisničko iskustvo bez dugog čekanja tijekom korištenja aplikacije te razvoj aplikacije na način da se određene razvijene komponente mogu koristiti na više mjesta.

LITERATURA

[1] About LinkedIn, https://about.linkedin.com/ (stranica posjećena 2.12.2022.)

[2] LinkedIn.com https://www.linkedin.com/ (stranica posjećena 2.12.2022)

[3] About ZipRecruiter https://www.ziprecruiter.co.uk/about (stranica posjećena 2.12.2022.)

[4] ZipRecruiter.com https://www.ziprecruiter.co.uk/ (stranica posjećena 2.12.2022.)

[5] MyOpportunity.com https://www.myopportunity.com/ (stranica posjećena 2.12.2022.)

[6] Novi Europass https://www.europass.hr/europass-dokumenti/novi-europass (stranica posjećena 12.12.2022)

[7] HTML: HyperText Markup Language, https://developer.mozilla.org/en-US/docs/Web/HTML (stranica posjećena 30.06.2022.)

[8] N. Young, **The Pros and Cons of Tailwind CSS**, https://www.webdesignerdepot.com/2021/09/the-pros-and-cons-of-tailwind-css/, September 2021 (stranica posjećena 30.06.2022.)

[9] Get started with Tailwind CSS, https://tailwindcss.com/docs/installation (stranica posjećena 30.06.2022.)

[10] G. Richards, S. Lebresne, B. Burg, J. Vitek, **An Analysis of the Dynamic Behaviour of JavaScript Programs**, S3 Lab, Department of Computer Science, Purdue University, West Lafayette, IN, June 2010

[11] What is JavaScript?, https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript (stranica posjećena 30.06.2022.)

[12] Vuejs.org, https://vuejs.org/guide/introduction.html (stranica posjećena 30.06.2022.)

[13] B. Nelson, **Getting to know Vue.js**, Apress Media, August 2018

[14] What is Vuex? https://vuex.vuejs.org/ (stranica posjećena 29.11.2022)

[15] R. Lerdorf, K. Tatroe, B. Kaehms, R. McGredy, **Programming PHP**, O'Reilly Media, March 2002

[16] Introduction to Laravel, https://laravel.com/docs/9.x/installation (stranica posjećena 30.06.2022.)

SAŽETAK

Web aplikacija razvijena u ovom diplomskom radu za cilj ima olakšati i ubrzati proces zapošljavanja. Korištenjem aplikacije kandidatu je pružena pomoć tijekom stvaranja životopisa na temelju kojeg mu nude preporučeni poslovi. Sukladno tome, poslodavcima je omogućeno stvaranje opisa tvrtke i oglasa za posao kojim se dohvaćaju kandidati koji najviše odgovaraju napisanom oglasu. Web aplikacija razvijena je tehnologijama Laravel i Vue.js. Parametri na temelju kojih se odvija preporuka kandidata ili oglasa za posao su dob, godine iskustva, lokacija i vještine koje kandidat treba posjedovati. Stvoreni životopisi, opisi tvrtke i oglasi za posao mogu se dijeliti i osobama koje nisu prijavljene na aplikaciju te se na taj način mogu slati životopisi i oglasi van web aplikacije. Uz sustav preporuke, kandidatima je omogućena prijava na željene oglase za posao, dok je poslodavcima omogućeno prihvaćanje i odbijanje kandidata.

Ključne riječi: Laravel, Oglas za posao, Sustav preporuke, Vue, Životopis

ABSTRACT

Web application for the recruitment process

The web application developed in this thesis aims to assist and speed up the recruitment process. By using the application, the candidate is helped during the creation of a CV, on the basis of which he is offered recommended jobs. Accordingly, employers are able to create company descriptions and job advertisements that retrieve candidates that best match the written advertisement. The web application was developed using Laravel and Vue.js technologies. The parameters on the basis of which the recommendation of a candidate or a job advertisement takes place are age, years of experience, location and the skills that the candidate should possess. Created resumes, company descriptions, and job ads can be shared with people who are not logged in to the application, and in this way, resumes and ads can be sent outside the web application. In addition to recommendation system, candidates are able to apply for desired job ads, while employers are able to accept and reject candidates.

Key words: CV, Job advertisement, Laravel, Recommendation system, Vue

PRILOZI

Prilog 1. Diplomski rad u docx formatu

Prilog 2. Diplomski rad u pdf formatu

Prilog 3. Programski kod web aplikacije