VIŠEJEZIČNA WEB APLIKACIJA I PLAN TESTIRANJA ZA PRAVNO SAVJETOVANJE

Phiri, Dorothy

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Algebra University College / Visoko učilište Algebra**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:225:065760

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-19**

Repository / Repozitorij:

Algebra Univerity College - Repository of Algebra Univerity College

VISOKO UČILIŠTE ALGEBRA

ZAVRŠNI RAD

VIŠEJEZIČNA WEB APLIKACIJA I PLAN TESTIRANJA ZA PRAVNO SAVJETOVANJE

Dorothy Phiri

Zagreb, siječanj 2019.

Student vlastoručno potpisuje Završni rad na prvoj stranici ispred Predgovora s datumom i oznakom mjesta završetka rada te naznakom:

„Pod punom odgovornošću pismeno potvrđujem da je ovo moj autorski rad čiji niti jedan dio nije nastao kopiranjem ili plagiranjem tuđeg sadržaja. Prilikom izrade rada koristio sam tuđe materijale navedene u popisu literature, ali nisam kopirao niti jedan njihov dio, osim citata za koje sam naveo autora i izvor, te ih jasno označio znakovima navodnika. U slučaju da se u bilo kojem trenutku dokaže suprotno, spreman sam snositi sve posljedice uključivo i poništenje javne isprave stečene dijelom i na temelju ovoga rada“.

U Zagrebu, datum.

Dorothy Phiri

Predgovor

Želim se zahvaliti Visokom učilištu Algebra, svim profesorima i studentima koji su mi pružili podršku u zadnje tri godine da svladam hrvatski jezik i da naučim programirati. Hvala. Dorothy Phiri.

Prilikom uvezivanja rada, Umjesto ove stranice ne zaboravite umetnuti original potvrde o prihvaćanju teme završnog rada kojeg ste preuzeli u studentskoj referadi

Sažetak

Predmet završnog rada je web aplikacija koja spaja ponudu i potražnju na tržištu pravnih usluga. Aplikacija omogućuje pružateljima pravnih usluga, odvjetnicima, da se registriraju i otvore svoj profil s kratkim sažetkom područja djelatnosti, iskustva i kontakt podataka. S druge strane, osobe koje imaju potrebu za pravnih savjetima imaju mogućnost postaviti upit uz odabir određenog pravnog područja. Nakon što određeni odvjetnik odgovori na upit, korisnik ima mogućnost neposredno kontaktirati tog odvjetnika i dogovoriti profesionalno pružanje pravnih usluga. Moguće je da na jedan upit više odvjetnika da svoje odgovore što korisniku povećava vjerojatnost pronalaska odgovarajućeg odvjetnika. Rezultat Završnog rada je ubrzanje pronalaska odvjetnika specijaliziranih za određena pravna područja te povećanje transparentnosti na tržištu pravnih usluga.

Ključne riječi: online pravne usluge, web aplikacija, pravna područja, odvjetništvo.

Summary

The subject of the final year thesis is a web application that combines the offer and demand on the legal services market. The application enables legal service providers, attorneys, to register and open their profile with a brief summary of the area of law, experience and contact information. On the other hand, people in need of legal advice have the option of asking for a selection of a particular legal field. Once a lawyer answers a query, the user has the ability to contact that attorney directly and arrange for a professional legal service. It is possible that one request will receive many responses, which increases the likelihood of finding a suitable attorney to the user. The outcome of this final thesis is to accelerate the finding of lawyers specializing in certain legal areas and increasing transparency in the legal services market.

Key words: online legal services, web applications, legal areas, law

Sadržaj

1. Uvod 1

2. Korištene tehnologije 3

2.1. Programski okvir .NET 3

2.1.1. CLR - Common Language Runtime 3

2.1.2. Class Library 3

2.1.3. Jezici i programski okviri 4

2.2. ASP.NET 4

2.2.1. Kratka povijest i modeli 4

2.3. ASP.NET Identity Framework 5

2.3.1. Elementi ASP.NET Identity 5

2.4. Troslojna arhitektura 7

2.5. MVC 8

2.5.1. Model 8

2.5.2. Pogled 9

2.5.3. Upravljač 9

3. Projektni zadatak 10

3.1. Opis projektnog zadatka 10

3.2. Cilj i svrha aplikacije 10

3.3. Istraživanje tržišta 11

3.3.1. Istraživanje tržišta odvjetničkih usluga u Hrvatskoj 11

3.3.2. Istraživanje cijena odvjetničkih usluga 12

3.3.3. Istraživanje konkurentskih djelatnosti 13

4. Aplikacija Advocate 14

4.1. Komponenta Advocate DAL 14

4.2. Komponenta Advocate core 17
4.3. Komponenta Advocate common 18
5. MVC arhitektura aplikacije 19
5.1. Upravljači 19
6. ASP.NET internacionalizacija 21
6.1. Globalizacija aplikacije 21
6.1.1. Globalizacija validacijske poruke 22
6.1.2. Globalizacija sadržaja 22
6.1.3. Određivanje jezika 23
7. Slučajevi korištenja 25
7.1. Slučajevi korištenja za klijenta 25
7.1.1. Registracija klijenta 25
7.1.2. Prijava korisnika 27
7.1.3. Postavljanje upita 30
7.1.4. *Dashboard* klijenta 31
7.1.5. Pregled odgovora 31
7.1.6. Prihvat odgovora 32
7.1.7. Upravljanje profilom 33
7.2. Slučajevi korištenja za odvjetnike 34
7.2.1. Registracija odvjetnika 34
7.2.2. Prijava odvjetnika 36
7.2.3. Odgovaranje na upit 36
7.2.4. Upravljanje profilom odvjetnika 38
8. Plan testiranja 39
8.1. Testiranje sa strane klijenta 39
8.2. Testiranje sa strane odvjetnika 42

8.3. Promjena jezika 45
9. Analiza ispunjenosti postavljenih ciljeva prema istraživanju tržišta 46
Zaključak 47
Popis Kratica 48
Popis Slika 49
Popis tablica 50
Popis kôdova 51
Literatura 52

1. Uvod

Predmet ovog rada je web aplikacija za online pravne usluge. Prilikom provedenog istraživanja utvrdili smo da u Republici Hrvatskoj, kao i u regiji, na području tržišta pravnih usluga vlada velika netransparentnost prilikom pronalaska odgovarajućih pravnih stručnjaka tj. u pravilu odvjetnika koji su jedini zakonom ovlašteni pružatelji pravnih usluga. Ujedno je utvrđeno da je se cjenovna politika odvjetničkih ureda značajno razlikuje od slučaja do slučaja, usprkos propisanoj odvjetničkoj tarifi. Primjerice, domaći odvjetnici inozemnim pravnim osobama naplaćuju i do 300% veće cijene nego domaćim pravnim osobama. Osim toga, tražiteljima pravnih usluga je teško provjeriti reference odvjetnika s obzirom da nacionalne odvjetničke komore odvjetnicima zabranjuju bilo koji oblik reklamiranja i marketinga. Stoga je cilj ovog rada pronaći tehničko rješenje koje s jedne strane tražiteljima pravnih usluga olakšava pronalazak odgovarajućih odvjetnika, koje stvara stvarno tržište pravnih usluga te koje naposljetku i samim odvjetnicima olakšava pronalazak klijenata i lakšu specijalizaciju u određeno pravnom području.

Izvori podataka su: Zakon o odvjetništvu, Status Hrvatske odvjetničke komore, Kodeks odvjetničke etike, Odvjetnička Tarifa, izvori iz raznih medija i podataka objavljenih od strane Državnog zavoda za statistiku. Prikupljanje podataka izvršeno je putem javno dostupnih online izvora.

Rad je podijeljen na tri dijela. U prvom dijelu se opisuje korištena razvojna okolina, aplikacijski okvir ASP.NET MVC uz njegove tri glavne komponente (modeli, pregledi i upravljači), Entity Framework kao objektno-relacijski maper (engl. ORM – *object relational mapper*) i ASP.NET Identity za učinkovito upravljanje sigurnošću. U drugom dijelu je opisan projektni zadatak, detaljno objašnjeno istraživanje tržišta u Republici Hrvatskoj i programsko rješenje web aplikacije - koje se sastoji od prikaza programskih komponenti, primjera kôda za pojedine dijelove aplikacije, slučajevi korištenja aplikacije i internacionalizacije aplikacije, radi istovremene dostupnosti različitih jezika. U trećem dijelu obrađuje se plan testiranje aplikacije, i to način upotrebe aplikacije od strane odvjetnika i od strane klijenta, testiranje funkcionalnosti za prikaz popisa pitanja, unos novog pitanja, registraciju korisnika, prijavu, odgovaranje na pitanja, zatvaranje pitanja i slično. Na kraju se opisuje analiza ispunjenosti postavljenih ciljeva prema istraživanju tržišta

i način na koji funkcionalnost aplikacije ispunjava postavljene ciljeve utvrđene istraživanjem tržišta iz prvog dijela.

2. Korištene tehnologije

Prilikom razvoja web aplikacije Advocate, korištena je razvojna okolina Microsoft Visual Studio 2017 i aplikacijski okvir ASP.NET MVC za razvoj web aplikacije s programskim jezikom C#, ASP.NET Identity za učinkovito upravljanje sigurnošću i Entity Framework kao objektno-relacijskog maper (engl. ORM – *object relational mapper*) vidljivo iz [1] . U daljnjem tekstu bit će objašnjene korištene tehnologije i metode.

2.1. Programski okvir .NET

Programski okvir .NET je Microsoftova platforma za razvoj softvera. Prva verzija .NET Frameworka izdana je u ožujku 2010. godine i bila je namijenjena razvoju aplikacija za Windows sustav. Ova platforma pruža kontrolirano okruženje za programiranje u kojemu se softver može razvijati, instalirati i izvršavati na Windows operativnim sustavima. Ujedno, programerima je omogućeno da razviju web forme i web aplikacije, s time da se i web servisi mogu razvijati koristeći .NET Framework. Platforma podržava programske jezike poput Visual Basic i C#, te među jezičnu interoperabilnost što znači da svaki jezik može koristiti kôd pisan u drugim jezicima.NET Framework sastoji se od sljedećih ključnih komponenti:

2.1.1. CLR - Common Language Runtime

CLR je softversko okruženje u kojem se programi pisani za .NET Framework izvršavaju. To je virtualni stroj koji pruža usluge poput sigurnosti, upravljanje memorijom i upravljanje iznimkama.

2.1.2. Class Library

Class Library se sastoji od skupine metoda i funkcija koje se mogu koristiti za ključne svrhe poput čitanja i pisanja datoteka, implementacija struktura podataka, mrežne komunikacije, podrške za implementaciju bogatog klijentskog grafičkog sučelja za windows i web aplikacije. .NET Framework Class Library ili skraćeno FCL logično su grupirane prema njihovoj funkcionalnosti i svrsi u imenske prostore (engl. *namespaces*).

2.1.3. Jezici i programski okviri

Postoji velik broj jezika u kojima se mogu pisati aplikacije koje se mogu izvršavati u .NET Frameworku. Ti jezici se mogu koristiti za izradu aplikacija u različitim vrstama programskih okvira, primjerice:

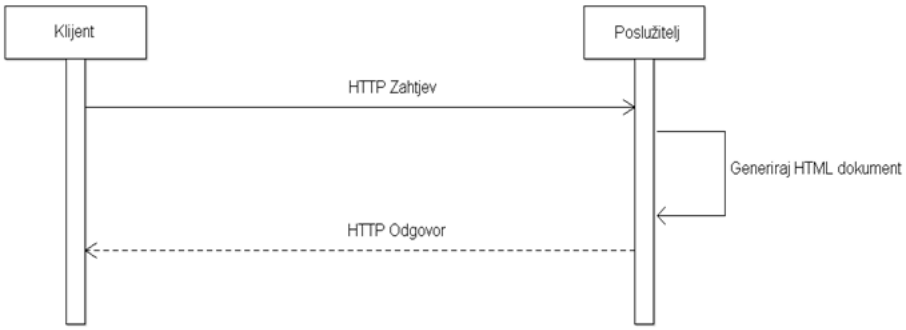
- WinForms za razvoj desktop aplikacija koja se izvode na stroju korisnika. Primjeri ovakvih aplikacija su Notepad, Word, Excel, Chrome i slično.
- ASP.NET se koristi za razvoj web aplikacija čiji se klijentski dio izvršava u bilo kojem pregledniku kao što su Edge, Chrome i Firefox.
- ADO.NET je tehnologija koja omogućuje komunikaciju aplikacije s bazama podataka kao što su Oracle ili Microsoft SQL.

2.2. ASP.NET

ASP.NET je web aplikacijski okvir koji programerima omogućuje izradu dinamičkih web stranica korištenjem bilo kojeg .NET jezika kao što su C#, VB.NET i JavaScript.

2.2.1. Kratka povijest i modeli

Prvi je put objavljen u siječnju 2002. kao verzija 1.0 .NET okvira kao nasljednik tehnologije *Active Server Pages*. ASP.NET okvir pruža sveobuhvatnu softversku infrastrukturu i različite usluge potrebne za izgradnju web aplikacija za računala i mobilne uređaje. ASP.NET podržava različite pristupe stvaranju web stranica. Ovi pristupi i modeli su *Classic* ASP, ASP.NET web format, ASP.NET MVC (engl. *Model View Controller*), ASP.NET web stranice, ASP.NET API (engl. *Application Programming Interface*) i ASP.NET Core. ASP.NET Framework radi na HTTP (Slika 2.1) protokolu koji je standardni protokol koji se koristi u svim web aplikacijama te koristi HTTP naredbe za postavljanje dvosmjerne komunikacije između preglednika i poslužitelja.



Slika 2.1 Dohvaćanje dinamičke web stranice

2.3. ASP.NET Identity Framework

ASP.NET Identity omogućuje učinkovito upravljanje sigurnošću ASP.NET aplikacija. ASP.NET Identity Framework definira skup API-ja koji omogućuju upravljanje korisnicima i njihovima dozvolama za pristup ASP.NET aplikacijama, upravljanje pravima i pravom pristupa različitim komponentama unutar ASP.NET aplikacije. Također omogućuje korisnicima da se registriraju u aplikaciji pomoću davatelja usluga za prijavu kao što su Facebook, Google, Twitter itd. vidljivo iz [2].

2.3.1. Elementi ASP.NET Identity

Glavni elementi su sljedeći:

- **One ASP.NET**

One ASP.NET uveden je u Visual Studio 2013. Može se koristiti u svim ASP.NET okvirima, kao što su ASP.NET MVC, web obrasci, Web API i SignalR. One ASP.NET omogučava izgradnju aplikacije i korištenje svih ASP.NET tehnologija u jednom projektu.

- **Upravljanje korisnicima**

ASP.NET identitet dolazi s API-em koji sadrži operacije kao što su stvaranje, uređivanje i brisanje korisnika.

- ***Control over schema***

Okvir omogućuje potpunu kontrolu nad informacijama o korisniku. Informacije o korisničkom profilu mogu se jednostavno proširiti tako da uključuju više informacija o korisniku, poput datuma rođenja ili bilo koje druge informacije. Identitet koristi Entity Framework kod da prvo pohrani podatke povezane s korisnikom u bazu podataka. Shema

baze podataka i vrsta podataka primarnog ključa mogu se promijeniti. Identitet također pruža kontrolu nad postojanošću podataka u bazi podataka. Budući da koristi Entity Framework, može se koristiti za čuvanje podataka u bilo kojem RDBMS-u kojeg podržava Entity Framework.

- **Dvo-faktorna autentifikacija koristeći e-mail/SMS**

Dvo-faktorna autentifikacija je proces u kojem se korisnik autentificira kombinacijom dviju različitih metoda. Na primjer slanjem lozinke i sigurnosnog koda korisnicima registriranom e-poštom ili mobilnim uređajima. Dvo-faktorna autentifikacija dodaje drugu razinu provjere autentičnosti u postupak prijave na korisničke račune.

- **Poruka potvrde računa**

E-mail s potvrdom korisničkog računa može se poslati korisniku radi verifikacije e-mail identiteta.

- **Zaključavanje računa**

Funkcija blokiranja računa onemogućuje korisničke račune ako korisnik unese pogrešnu lozinku određeni broj puta tijekom kratkog perioda.

- **Unit test**

Unit test omogućuje testiranje dijelova aplikacije u malim jedinicama kako bi se utvrdilo jesu li pogodni za korištenje.

- **Upravljanje ulogama**

Uloga (engl. *role*) je slična ulozi u starom sustavu ASP.NET Membership. One se pohranjuju u zasebnu tablicu. Operacije kao što su stvaranje, uređivanje i brisanje mogu se izvoditi na korisničkim ulogama.

- **Upravljanje tvrdnjama**

Tvrdnja (*engl.* claim) je određeni podatak o korisniku. To može biti njegovo ime, njegov identitet, e-mail ili bilo što važno za prijavu. Pohranjeni su kao par vrijednosti imena i vrijednosti. Ovo omogućuje razvojnim programerima da dodaju puno više informacija o korisniku od samo lozinke i korisničkog imena koje je pružao stari sustav članstva.

- **Vanjski pružatelji identiteta**

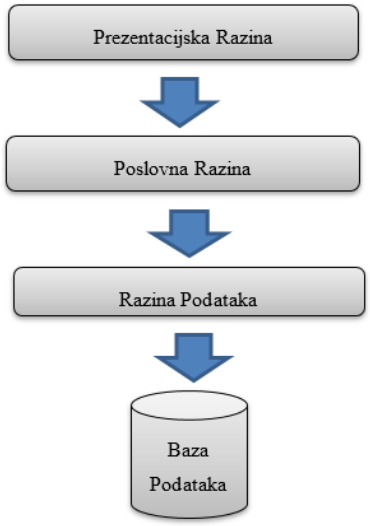
Dopušta korisnicima da se prijave pomoću bilo kojeg davatelja usluga za prijavu na društvenim mrežama, kao što su Facebook, Google +, Linkedin itd. Davatelji usluga vanjskog identiteta za Active Directory / Azure Active Directory su također dostupni.

- **OWIN Integracija**

ASP.NET Identitet koristi OWIN za generiranje kolačića i provjeru autentičnosti. Nije potrebno koristiti OWIN za identitet. U istu svrhu se može upotrijebiti i modul za provjeru autentičnosti obrazaca.

2.4. Troslojna arhitektura

Troslojnu arhitekturu razvio je John J. Donovan dok je radio u tvrtki u Open Environment Corporation (OEC), poduzeću za alate koje je osnovao u Cambridgeu, Massachusetts. Troslojna arhitektura (Slika 2.2) je model softverske arhitekture u kojem se prezentacija, poslovanje, pohrana podataka i pristup podacima razvijaju u zasebnim programskim modulima koji programerima omogućuju podjelu složenog projekta u manju funkcionalnosti aplikacija u tim neovisnim sustavima koji se mogu ponovno koristiti i čine aplikaciju fleksibilnom, upravljivom, održivom i skalabilnom.



Slika 2.2 arhitektura troslojne aplikacije

- **Prezentacijska razina**

Presentacijka razina je sloj kojem korisnici mogu izravno pristupiti (kao što je web stranica ili operativni sustav, grafičko korisničko sučelje) te dopušta manipulaciju podataka i unos

podataka. Također je odgovorna za komunikaciju između korisnika i web stranica koje koristi objekte iz poslovnog sloja za obradu i prikaz klijentskog sadržaja.

- **Poslovna razina**

Također se zove sloj poslovne logike, te sadrži svu aplikacijsku logiku, algoritme, izračune i pravila podataka kako bi se postigla veća produktivnost. Komponente koje čine ovaj sloj uglavnom postoje na poslužitelju kako bi omogućile dijeljenje resursa. Zbog toga komponente srednje razine nisu vezane za određenog klijenta, što znači da ovaj sloj nema znanja o bilo kojoj kontroli GUI i kako pristupiti bazama podataka. Klase razine pristupa podacima dostavit će potrebne podatke iz baza podataka u poslovni logički sloj.

- **Razina podataka**

Ovo je pristupni sloj s bazom podataka kojemu se može pristupiti kroz sloj poslovnih usluga. Razina podataka odgovorna je za dohvaćanje, pohranjivanje, ažuriranje informacija i slanje podataka natrag poslovnoj logici, pa je ova razina idealno predstavljena komercijalnom bazom podataka. Pohranjeni postupci smatraju se dijelom razine podataka jer povećavaju učinkovitost i transparentnost kôda u aplikaciji.

2.5. MVC

MVC je softverski arhitektonski uzorak koji se može koristiti za razvoj web aplikacija. MVC uzorak odvaja aplikaciju u tri logičke komponente koje su *Model* (hrv. model), *View* (hrv. pogled) i *Controller* (hrv. upravljač). Svaka komponenta je izgrađena za rukovanje specifičnom aplikacijskom logikom i odvojeno se razvija kako bi se omogućilo da se aplikacija može lako testirati, a podržava i *test driven* razvoj koji je idealan za razvoj složenih, ali *light-weight* aplikacija.

2.5.1. Model

Model je dio aplikacije koja implementira logiku domene, logike i pravila podataka aplikacijskih podataka. Ona odgovara podatkovnoj logici s kojom korisnik radi. Model može predstavljati poslovni sloj aplikacije koji se prenosi podatke između pogleda i upravljača. Ona također dohvaća i pohranjuje stanje modela u bazi podataka.

2.5.2. Pogled

Ova komponenta prikazuje podatke na korisničkom sučelju i odgovara za logiku korisničkog sučelja. Ovo obuhvaća komponente korisničkog sučelja kao što su gumbi, okviri za tekst, padajući izbornici i slično s kojima krajnji korisnik može komunicirati.

2.5.3. Upravljač

Obrađuju i reagiraju na interakciju s korisnicima, djeluje kao sučelje između modela i prikaza. Obrađuje svu poslovnu logiku i dolazne zahtjeve, manipulira podacima pomoću komponente modela i određuje pogled za prikaz korisnika.

3. Projektni zadatak

U ovom radu projektni zadatak je izrada web aplikacije u programskom jeziku C# koja će služiti za pružanje online odvjetničkim i pravnim uslugama. Završni rad će se baviti stvarnim rješenjem predviđenog zadatka.

3.1. Opis projektnog zadatka

Projektni zadatak ovog rada je razvoj programskog rješenja za spajanje odvjetnika i klijenata. Navedeno se obično sastoji od pisanja kôda u nekom od razvojnih okruženja, kreiranja potrebne baze podataka, potrebnih modela podataka te testiranja same web aplikacije. U ovom radu, postupak razvoja aplikacije se sastojao iz sljedećeg: kreiranje projekta u Microsoft Visual Studio-u kao odabranom razvojnom okruženju, stvaranje potrebnog modela podataka u istom korištenjem C# koda, uporaba Entity Frameworka te kreiranje baze podataka na osnovu stvorenog modela podataka korištenjem tzv. *Code-first* pristupa, kreiranje upravljača koje sadrže i opisuju potrebne akcije za upravljanje podacima (upravljanje profilom, dodavanje novog korisnika, zatvaranje pitanja, prikaz popisa pitanja, prikaz odgovora i sl.). Nakon kreiranja svih potrebnih komponenti, aplikacija je realizirana te je posljednji korak u postupku razvoja upravo testiranje same aplikacije. Uz unesene vrijednosti u bazu podataka, testiraju se sve moguće funkcionalnosti koje bi aplikacija trebala obavljati, traže se moguće greške i izuzeci koje je potrebno korigirati i ispraviti. Nakon obavljenog testiranja, aplikacija je uspješno stvorena te spremna za korištenje.

3.2. Cilj i svrha aplikacije

Cilj i svrha aplikacije je izrada informacijskog sustava koji omogućuje stvaranje online automatiziranog tržišta odvjetničkih usluga koje će s jedne strane olakšati malim i samostalnim odvjetnicima da se specijaliziraju i pronađu klijente, a građanima i trgovačkim društvima će olakšati da pronađu adekvatnog odvjetnika za rješenje svojih pravnih problema. Aplikacija omogućuje otvoren i besplatan pristup svim odvjetnicima Republike Hrvatske, kao i svim građanima i trgovačkim društvima. Smatram da bi stvarno korištenje aplikacije imalo velike pozitivne učinke na razvoj odvjetništva, transparentnosti, dostupnosti pravne pomoći i jačanje vladavine prava u Republici Hrvatskoj. U trenutnoj verziji aplikacija

je stvorena za pružanje tzv. besplatnih pravnih informacija, općih informacija o pojedinoj pravnoj stvari te je u tom smislu usklađena s propisima o besplatnoj pravnoj pomoći.

3.3. Istraživanje tržišta

U ovom poglavlju obrađuje se pitanje ukupnog broja odvjetnika i odvjetničkih vježbenika u Republici Hrvatskoj, njihovo poslovanje, propise koji reguliraju odvjetništvo, statističke podatke o potencijalnim klijentima, cijenama odvjetnika i potencijalno konkurentskim djelatnostima.

3.3.1. Istraživanje tržišta odvjetničkih usluga u Hrvatskoj

Pregledom dostupnih online izvora i službenih statističkih podataka Državnog zavoda za statistiku utvrdili smo da u Republici Hrvatskoj u 2018. godini trenutno djeluje oko 4.500 odvjetnika i oko 2.000 odvjetničkih vježbenika, s time da je većina odvjetnika djeluje na području Zagreba, oko 2.000 odvjetnika, kako je vidljivo iz [11]. Nadalje iz izjava visokih dužnosnika Hrvatske odvjetničke komore, ponajprije javnih intervjua predsjednika komore, utvrdili smo da je u periodu od 2010. do 2016. godine bilo preko 800 odvjetnika koji su bili u trajnom stanju blokade zbog nemogućnosti podmirenja dospjelih poreznih obveza [11]. Nadalje, iz istih izvora utvrdili smo da je odvjetnička djelatnost suficitarna, te da zbog krize iz 2008. godine, te zbog nekontroliranog priljeva diplomiranih pravnika na tržište rada dolazi do asimetrije u ponudi i potražnji odvjetničkih usluga, odnosno ponuda je značajno prevladala potražnju. Iz aplikacije MojPosao.hr proizlazi da je prosječna plaća ili neto primanje odvjetnika u Republici Hrvatskoj oko 6.800,00 do 7.500,00 kn, što znači da je nekada prestižno i cijenjeno zanimanje postalo srednje plaćeno zanimanje, vidljivo iz [8]. Nadalje, istraživanjem javno objavljenih top lista odvjetničkih društava koje imaju najviše godišnje prihode, kao i istraživanjem javno objavljenih podataka na sudskom registru, utvrdili smo da gotova sva tzv. uspješna odvjetnička društva imaju svojevrsnu patrijarhalnu vlasničku strukturu, tj. vlasnici odvjetničkih društava su uglavnom jedna do dvije osobe koje uzimaju cjelokupnu dobit, dok se ostalim zaposlenicima društva tj. zaposlenim odvjetnicima, isplaćuje fiksna mjesečna plaća, kako je vidljivo iz [11].

Uspoređujući podatke o broju odvjetnika sa službenim podacima Državnog zavoda za statistiku, dobivaju se vrlo negativni rezultati po trenutno stanje odvjetnika. Naime, prema zadnje objavljenim podacima u Republici Hrvatskoj ima oko 1,3 milijuna zaposlenih, od

čega samo 2/3 u privatnom sektoru, dok broj aktivnih trgovačkih društava iznosi oko 153 tisuće. Iz tih podataka proizlazi da na jednog odvjetnika dolazi oko 188 zaposlenih osoba u privatnom sektoru, odnosno oko 288 ukupno zaposlenih osoba, kako je vidljivo iz [9]. S druge strane, na jednog odvjetnika dolazi oko 34 aktivna trgovačka društva, vidljivo iz [10]. S obzirom na očiglednu okolnost da ne koriste sve osobe i sva trgovačka društva odvjetnike, smatramo da ovi statistički podaci pokazuju negativnu sliku.

Iz propisa koji reguliraju odvjetništvo proizlazi da je odvjetnicima sve do kolovoza 2018. godine bilo u cijelosti zabranjeno reklamirati se ili oglašavati svoje usluge na bilo koji način. Za web stranice odvjetnika postojali su posebnim propisi Hrvatske odvjetničke komore, tzv. Pravilnik o web stranici i Kodeks odvjetničke etike, koju su nalagali svakom odvjetniku da prije objave web stranice istu podnese na suglasnost tijelima odvjetničke komore, kako je vidljivo iz [6]. Pravilnik o web stranici je izrazito strogo regulirao sadržaj web stranice, tako da su odvjetnici smjeli objaviti samo osnovne i bazične podatke poput npr. područja prava kojima se bave i kontakt podatke. Pregledom strukture ovlaštenih dužnosnika Hrvatske odvjetničke komore zaključili smo da se većinom radi o osobama koje su ujedno i vlasnici najvećih odvjetničkih društava u državi, odnosno neznatan je udio tzv. malih ili samostalnih odvjetnika u upravljačkim tijelima, vidljivo iz [7].

3.3.2. Istraživanje cijena odvjetničkih usluga

Cijene odvjetničkih usluga ne određuju se prema načelu slobodnog tržišta tj. prema načelima ponude i potražnje, već su iste regulirane kao minimalne cijene sukladno tzv. Odvjetničkoj tarifi (dalje: Tarifa), internom aktu Hrvatske odvjetničke komore koji odobrava Ministarstvo pravosuđa, vidljivo iz [7]. Pojašnjavamo da odvjetnici ne mogu ponuditi cijenu nižu od one propisane Tarifom, već mogu naplatiti samo najnižu propisanu cijenu ili veću. Tarifom je propisano da se cijene u iznimnim slučajevima mogu smanjiti do 50% za pojedinu stavku, a s druge strane da se u slučaju odvjetničke specijalizacije mogu povećati do 100% za pojedinu stavku. Provjerom dostupnih online izvora i postavljanjem direktnih upita odvjetnicima i odvjetničkim društvima utvrdili smo da mnogi odvjetnici i odvjetnička društva u građanskim i trgovačkim stvarima primjenjuju obračun cijene po satu rada, te ista iznosi: (a) za područje manjih gradova i mjesta od 250,00 do 750,00 kn po satu rada, (b) za područje većih gradova od 500,00 do 1.000,00 kn po satu radu. Najveća odvjetnička društva u državi u kompleksnijim slučajevima znaju naplaćivati i satnice od 1.500,00 do 3.000,00 kn po satu, no to su iznimnim slučajevi.

3.3.3. Istraživanje konkurentskih djelatnosti

Iako je po Zakonu o odvjetništvu samo odvjetnicima priznato pravo da se profesionalno bave pružanjem pravne pomoći, u pojedinačnim intervjuima s odvjetnicima na području Zagreba utvrdili smo da su u zadnjih desetak godina javni bilježnici preuzeli veliki dio posla koji su prije obavljali odvjetnici, i to npr. veliki dio poslova vezano za osnivanje trgovačkih društava i unutarnje korporativne izmjene (imenovanja i razrješenja uprave i sl.), da javni bilježnici strankama sastavljaju ugovore o darovanju i pravno ih savjetuju u području nasljednog i ovršnog prava. Uz navedeno, uočeno je i da druge djelatnosti npr. knjigovođe često ulaze u domenu pravnog savjetovanja i strankama nude jednostavnije predloške ugovora (npr. ugovori o radu, najmu prostora i sl.), a da agencije za promet nekretninama često bez sudjelovanja odvjetnika zaključuju ugovore o kupoprodaji nekretnina i provode prijenos nekretnina kroz zemljišne knjige.

4. Aplikacija Advocate

Aplikacija Advocate se sastoji se od komponenti Advocate.Core, Advocate.DAL, Advocate.Common i Advocate.Web koje definiraju troslojnu arhitekturu aplikacije.

4.1. Komponenta Advocate DAL

U ovom sloju domene su definirane i stavljene u bazu podataka pomoću Entity Framework Code-First migracije za stvaranje baze podataka i umetanje tablica u bazu podataka kako je vidljivo iz [3]. Code-First pristup omogućava programerima da definiraju klase poslovnih modela. (Tablica 4.1) Iz definiranih poslovnih modela se baza podataka stvara pomoću Entity Framework-a.

Model	Opis Modela
User	Svojstva vezana uz korisnika
Question	Svojstva vezana za postavljena pitanja
Answer	Svojstva vezana za odgovor na upit
Specialization	Svojstva vezana za specijalizaciju odvjetnika
Area of Law	Svojstva područja prava

Tablica 4.1 Opis modela

Entity Framework pruža konverzije koje su skup zadanih pravila koja automatski konfiguriraju klase domena u shemu baze podataka. Primjerice, naziv klase bit će tablica u bazi podataka konvencijom, bez dodanih konfiguracija. U ovoj aplikaciji Entity Framework konverzije nisu korištene i svi modeli konfigurirani su pomoću Fluent API-ja jer pruža više funkcionalnosti za konfiguraciju nego alternativni pristupi. (**Pogreška! Izvor reference nije pronađen.**)

```
public class QuestionConfig : EntityTypeConfiguration<Question>
{
    public QuestionConfig()
    {
        HasKey(p => p.QuestionId);
        Property(p => p.QuestionId)
           .HasDatabaseGeneratedOption(DatabaseGeneratedOption.None);

        HasRequired(x => x.AreaOfLaw)
             .WithMany(x => x.Questions)
             .HasForeignKey(x => x.AreaOfLawId);

        HasRequired(x => x.User)
            .WithMany(x => x.Questions)
```

```
                .HasForeignKey(x => x.UserId);

        }
    }
```

Kôd 3.1 Konfiguracija klase Question

Da bi kod lakše upravljiv, svaki model domene u aplikaciji ima svoju vlastitu konfiguracijsku klasu. U ovom slučaju model domene „Upit“ ima konfiguracijsku klasu QuestionConfiguration koja nasljeđuje od EntityTypeConfiguration<TEntityType>, gdje je TEntityType klasa domene na kojoj će biti izvršene konfiguracije. EntityTypeConfiguration omogućuje konfiguracije na klasi domene pomoću različitih metoda za konfiguriranje određenih svojstava. Nakon konfiguriranja, klase se moraju primijeniti na bazu kroz DbContext. DbContext klasa često se naziva kontekst i odgovoran je za upravljanje Entity objektima tijekom vremena pokretanja, popunjavanje objekata s podacima iz baze, praćenje promjena na modelu ili entitetu i postojanja podataka u bazi podataka. U Code-First pristup se preporučuje definiranje vlastitih klasa (Kôd 3.2 Klasa CustomContext

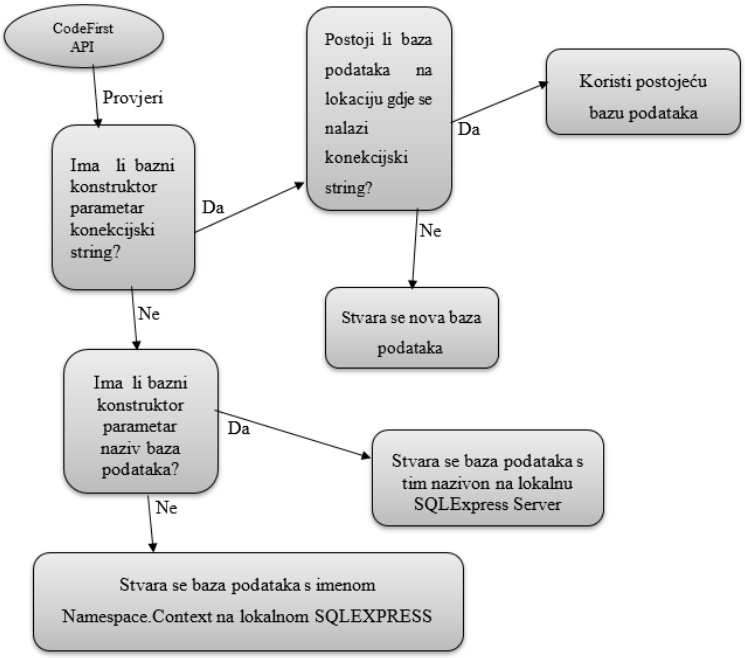
) koje nasljeđuju DbContext.

```
    public class CustomContext : IdentityDbContext<User, Role, string,
        IdentityUserLogin, UserRole, IdentityUserClaim>
    {
        public CustomContext() : base("DefaultConnection")
        {
        }
        public DbSet<UserRole> UserRoles { get; set; }
        public DbSet<Answer> Answers { get; set; }
        public DbSet<Question> Questions { get; set; }
        public DbSet<UserSpecialization> Specializations { get; set; }
        public DbSet<AreaOfLaw> AreasOfLaw { get; set; }

        protected override void OnModelCreating(DbModelBuilder
modelBuilder)
        {
          base.OnModelCreating(modelBuilder);

          modelBuilder.Configurations.Add(new AnswerConfig());
          modelBuilder.Configurations.Add(new QuestionConfig());
          modelBuilder.Configurations.Add(new
UserSpecializationConfig());
          modelBuilder.Configurations.Add(new AreaOfLawConfig());
          modelBuilder.Entity<User>().ToTable("UserProfile");
          modelBuilder.Entity<IdentityUserLogin>().ToTable("UserLogins");
          modelBuilder.Entity<IdentityUserClaim>().ToTable("UserClaims");
          modelBuilder.Entity<Role>().ToTable("Roles");
          modelBuilder.Configurations.Add(new UserRoleConfig());
        }
    }
```

Kôd 3.2 Klasa CustomContext

U ovoj aplikaciji kontekstna klasa nasljeđuje iz IdentityDbContext koji također zapravo nasljeđuje iz DbContext klase. Razlog nasljeđivanja od identiteta Dbcontext je zato što to omogućuje kontekstu da bude svjestan bilo kakvih odnosa između drugih klasa i IdentityUser klasa te uloge korisnika od Identity Framework.

Na slici (Slika 3.2) prikazan je proces inicijalizacije baze podataka, baziran na parametru u baznom konstruktoru (engl. base constructor) klase CustomContext (Kôd 3.2). Ako nije naveden parametar u baznom konstruktoru kontekstnog klasa, stvara se baza podataka s nazivom Namespace.CustomContext na lokalnom SQL Express server. U slučaju da je naziv baze podataka definiran u konstruktoru kao sto je prikazan (Kôd 3.2) tada se tijekom inicijalizacije baza podataka stvara bazu podataka s tim nazivom. U baznu konstruktoru se također može definirati konekcijski string kao parametar "name=ConnectionStringName" tijekom inicijalizacije baza podataka Entity Framework će koristiti postojeću bazu podataka ili stvoriti novu bazu podataka koristeći naziv baze podataka definirano u konekcijskom string-u.



Slika 4.1 Inicijalizacija baza podataka

Vlastiti kontekst mora sadržavati svojstva tipa DbSet za entitete koji mapiraju tablice i vlasničke preglede u bazi podataka te također omogućuje spremanje podataka u bazu podataka pomoću CRUD (engl. *Create*, *Read*, *Update*, *Delete*) operacija.

Kao što je prikazano u kodu (Kôd 3.2), metoda OnModelCreating ima svojstvo modelBuilder koje se koristi za dodavanje konfiguracije u klase domene i nadjačavanje zadanih konverzija.

4.2. Komponenta Advocate core

Komponenta Advocate core je most između prezentacijskog sloja, u ovom slučaju MVC, i podatkovnog sloja u kojem je poslovna logika. Ovaj sloj definira pogled-modele i servise koje kontrolori mogu pozivati u prezentacijskom sloju. Pogled-modeli mogu sadržavati samo entitete koji su potrebni za prikaz i mogu biti korisni u scenarijima kao što su prijenos podataka, prikazivanje podataka za više modela domene zapisivanje podataka i validacija za logiku prikaza.

```
    public class CustomerViewModel
    {
        public CustomerViewModel(User user)
        {
            Id = user.Id;
            Email = user.Email;
        }
        public string Id { get; set; }
        public string Code { get; set; }
        [Required]
        [EmailAddress]
        public string Email { get; set; }
        [Required]
        [StringLength(100, ErrorMessage = "*The {0} must be at least {2}
characters long.", MinimumLength = 6)]
        [DataType(DataType.Password)]
        [Display(Name = "Password")]
        public string Password { get; set; }
        [Required]
        [DataType(DataType.Password)]
        [Display(Name = "Confirm password")]
        [Compare("Password", ErrorMessage = "*The password and
confirmation password do not match.")]
        public string ConfirmPassword { get; set; }
    }
```

Kôd 4.5 Klasa CustomerViewModel sadržava svojstva za pogled

Kôd 3.5 prikazuje pogled-model za registraciju korisnika. Ovo je omotač oko User modela koji sadrži mnoga druga svojstva, ali za potrebe korisnika prilikom registracije u ovom slučaju potrebni su samo e-mail lozinke i lozinke za vlasnički pregled klijenta.

Pogled-model sadrži atribute podataka koji određuju pravila za provjeru valjanosti koja također omogućuju provjeru na klijentskoj strani, čime se postiže bolji korisnički doživljaj

bržim reagiranjem na razini preglednika, čineći ga prikladnijim za korisnike. Validacija na klijentskoj strani može se izvršiti pomoću skriptnog jezika kao što je JavaScript.

4.3. Komponenta Advocate common

Ova komponenta sadrži zajedničke funkcionalnosti koje se koriste u ostalim slojevima aplikacije. Primjerice, enumeracija RoleEnum sadrži uloge koje korisnik može imati na web aplikaciji, a to su SuperUser, Odvjetnik i Customer. SuperUser može vidjeti sve korisničke profile, pitanja, odgovore te može brisati neprikladna pitanja. Odvjetnik može vidjeti sva pitanja i može odgovarati na pitanja korisnika. Customer može postaviti pravna pitanja i vidjeti sve odgovore koji se odnose na određeno pojedinačno pitanje.

```
public enum RoleEnum
{
    SuperUser, Odvjetnik, Customer
}
```

5. MVC arhitektura aplikacije

Sloj prezentacije aplikacije sadržava upravljače koji prikupljaju podatke iz baze i prikazuju se u pogledu kroz Unit of Work servis.

5.1. Upravljači

Upravljač sadrži prezentacijsku logiku, obrađuje sve unose korisnika, obavlja izmjenu podataka s modelom i određuje pogled. Upravljač sadrži više akcija koje se izvode ovisno o unosu korisnika. Aplikacija sadrži sljedeće upravljače:

- Account: sadrži sve akcije vezane uz korisnike
- Home: sadrži opće stranice portala (Kategorija Prava, About)
- Category: sadrži pojedinosti o odabranoj kategoriji prava i obrazac zahtjeva
- Base: za zajedničke akcije / funkcije koje mogu koristiti svi upravljači koji ga nasljeđuju. Ove akcije mogu biti upravljanje iznimkama, vlastita autorizacija i slično.

Unos korisnika određuju tri elementa:

- Naziv upravljača
- Naziv akcije
- HTTP metodu

Naziv upravljača određuje koji upravljač obrađuje korisnikov unos, naziv akcije određuje akciju upravljača koja će se izvesti, dok HTTP metoda određuje zahtijeva li klijent podatke (GET metoda) ili šalje podatke (POST metoda). Tablica 4.2 prikazuje moguće korisničke unose kao kombinacije upravljača, akcije i HTTP metode. U tablici je vidljivo da akcije vezane uz isti upravljač mogu imati jednako ime, ali se u tome slučaju moraju razlikovati po HTTP metodi. To je uobičajena praksa jer je često potrebno dohvatiti stranicu (GET metoda) koja sadrži neku formu (npr. stranica za prijavu korisnika), a nakon ispunjavanja forme potrebno je poslati podatke (POST metoda).

Neke akcije su definirane samo GET metodom, takve akcije služe za dohvat stranica koje ne sadrže nikakvu formu za ispunjavanje podataka (npr. stranica za prikaz kategorija prava), svaka akcija definirana GET metodom povezana s istoimenim pogledom kojeg će

prezentirati kao odgovor na zahtjev (Tablica 5.1). Akcije definirane samo POST metodom predstavljaju akcije koje obavljaju određenu izmjenu podataka, ali im nisu potrebni podaci iz ranije dohvaćene forme (npr. brisanje korisnika, brisanje upita). Svaka akcija definirana POST metodom, nakon obrade podataka, također rezultira vraćanjem pogleda korisniku (Tablica 5.2)

Upravljač	Akcija	Pogled
Account	Login	Index
	LogOut	Login
	Register	Index
	Questions	Questions&Anwers
	Manage	Manage
	Delete/User	Users
Dashboard	Index	Index
	GetAllKategories	Index
Category	GetCategory	Category
	Question	Category

Tablica 5.1 Akcije (GET metoda) s vezanim pogledom

Upravljač	Akcija	Pogled
Account	Login	Index
	LogOut	Login
	Register	Index
	Questions	Questions&Anwers
	Manage	Manage
	Delete/user	Users
Category	GetCategory	Category
	Question	Category

Tablica 5.2 Akcije (POST metoda) s vezanim pogledom

6. ASP.NET internacionalizacija

Internacionalizacija (engl. *internationalization*) je proces koji uključuje globalizaciju i lokalizaciju softvera ili aplikacije kako bi se istovremeno učinilo dostupnom korisnicima različitih jezika i kultura vidljivo iz [4]. Globalizacija je proces oblikovanja i kodiranja aplikacija u svrhu podržavanja višejezičnosti, a lokalizacija se može definirati kao prevođenje i prilagodba aplikacije ili softvera određenom jezičnom području. Internacionalizacija se često skraćuje na "I18N". Ova kratica sadrži prva i posljednja slova i broj slova između njih, pa 18 označava broj slova između prvog "I" i zadnjeg "N". Isto vrijedi i za globalizaciju (G11N) i lokalizaciju (L10N). Postoje dva načina uključivanja različitih jezika i kultura u ASP.NET aplikaciju, a to su:

- **Pomoću datoteka resursa**

  ASP.NET podržava posebnu datoteku koji se zove resursna datoteka (engl. *resource file*) u kojoj se svi oni statički tekstovi u aplikaciji, koji trebaju biti prevedeni, pohranjuju s jedinstvenim ključem kojim se identificiraju.

  Za određivanje podrazumijevanog jezika, ekstenzija datoteke trebala bi sadržavati samo .resx, a za ostale jezike ekstenzija datoteke bi trebala uključivati kôd kulture Primjerice, kôdovi kulture mogu biti „en-US“ za engleski američki, „hr“ za hrvatski jezik itd.

- **Koristeći različite poglede za svaki jezik**

  Neke aplikacije mogu zahtijevati implementaciju različitih prikaza za svaki jezik, jer na taj način pogledi u aplikaciji mogu biti lakši za upravljanje i neopterećeni ključevima za prepoznavanje statičkih tekstova u datoteci resursa, ili zbog toga što raspored treba biti drukčiji za jezike koji se pišu s desna na lijevo (npr. arapski i slični jezici).

6.1. Globalizacija aplikacije

Zadani jezik aplikacije Advocate je hrvatski, a kako bi se globalizirala ova aplikacija, koristi se ASP.NET resursna datoteka. Svaki pogled ima vlastitu datoteku resursa s ključnim i vrijednosnim parovima statičnog teksta koji su prikazani na pogledu. Aplikacija također sadrži resursnu datoteku za poruke o validacijama na klijentskoj strani.

6.1.1. Globalizacija validacijske poruke

Za prikaz poruka o pogreškama na različitim jezicima na pogledima kao što su prijava, registracija itd, koriste se anotacije podataka (engl. *Data Annotations*). Anotacije omogućuju logiku validacije klijentske strane kao što su obavezan unos, duljina lozinke, e-mail regularni izraz za provjeru formata e-mail adresa, duljinu input teksta itd. Za globalizaciju validacijske poruke potrebno je dodati sljedeće parametre: parametar "ErrorMessageResourceType" koji označava vrsta datoteku resursa i parametar "ErrorMessageResourceName" koji označava naziv resursa za traženje pogrešaka poruka kao što je prikazano u kod (Kôd 6.2)

```
        public class CustomerViewModel
        {
            [Display(Name = "Email", ResourceType =
typeof(Resources.ValidationMessages))]
            [Required(ErrorMessageResourceType =
typeof(Resources.ValidationMessages), ErrorMessageResourceName =
"EmailRequired")]
            [RegularExpression(".+@.+\\..+", ErrorMessageResourceType =
typeof(Resources.ValidationMessages), ErrorMessageResourceName =
"EmailInvalid")]
            public string Email { get; set; }

            [Display(Name = "Password", ResourceType =
typeof(Resources.ValidationMessages))]
            [Required(ErrorMessageResourceType =
typeof(Resources.ValidationMessages), ErrorMessageResourceName =
"PasswordRequired")]
            [StringLength(ErrorMessageResourceType =
typeof(Resources.ValidationMessages), ErrorMessageResourceName =
"PasswordLength")]
            public string Password { get; set; }

            [Display(Name = "Confirm Password", ResourceType =
typeof(Resources.ValidationMessages))]
            [Required(ErrorMessageResourceType =
typeof(Resources.ValidationMessages), ErrorMessageResourceName =
"PasswordConfirmationRequired")]
            [Compare("Password", ErrorMessageResourceType =
typeof(Resources.ValidationMessages), ErrorMessageResourceName =
"ComparePasswordError")]
            public string ConfirmPassword { get; set; }

        }
```

Kôd 6.1 CustomerViewModel s globaliziranim validacijskim porukama

6.1.2. Globalizacija sadržaja

Globalizacija sadržaja uključuje globalizaciju statičkih tekstova, imena, opisa i drugog sadržaja na stranici koji nije dinamički izrađen ili pročitan iz baze podataka. Za globalizaciju

sadržaja u ovom radu napravljeno je da je svaki pogled ima vlastitu datoteku resursa s parovima ključa / vrijednosti koji se odnose na tekst, oznake i opise na svakom pogledu.

6.1.3. Određivanje jezika

Određivanje jezika ili kulture koju korisnik koristi može se odrediti čitanjem polja zaglavlja pod nazivom „Accept-Language“ koji preglednik šalje na svaki zahtjev. Ovo zaglavlje sadrži popis imena kulture zemlje ili jezika koju je korisnik konfigurirao u pregledniku, ali problem s upotrebom tog polja zaglavlja za prikaz sadržaja korisniku jest da jezik ili kultura u ovom polju možda ne odražavaju željeni jezik stvarnog korisnika. Stoga u ovoj aplikaciji korisnik može eksplicitno odabrati željeni jezik klikom na gumb „Jezik“ u glavnoj traci koji poziva akciju „SetLanguage“ u upravljaču, što korisniku omogućuje promjenu trenutnog jezika ili kulture i pohranjuje odabrani jezik u kolačić (Kôd 5.3).

```
        public ActionResult SetLanguage(string abr,string returnUrl)
        {
            if (abr != null)
            {
                Thread.CurrentThread.CurrentCulture =
CultureInfo.CreateSpecificCulture(abr);
                Thread.CurrentThread.CurrentUICulture = new
CultureInfo(abr);
            }

            HttpCookie cookie = new HttpCookie("Language");
            cookie.Value = abr;
            Response.Cookies.Add(cookie);

            return Redirect(returnUrl);

        }
```

Kôd 6.2 Akcijska metoda za promjenu i spremanje odabranog jezika u kolačić

Da bi se prikazao sadržaj aplikacije na temelju jezične postavke korisnika, bazni upravljač ima metodu BeginExecuteCore() kojom aplikacija provjerava postoji li kolačić i postavlja trenutni jezik na tu vrijednost kolačića. Kako bi se osiguralo da postojeći kolačić ima vrijednost, postoji pomoćna klasa (Kôd 5.4) koja sadrži kulture ili jezike koje ova aplikacija upotrebljava za provjeru jezika. U slučaju da vrijednost kolačića ne odgovara bilo kojem od jezika definiranih u pomoćnoj klasi, resursni menadžer tada koristi zadani jezik za ovu aplikaciju tj. hrvatski jezik.

```
    public static class CultureHelper
    {
```

```
        private static readonly List<string> _cultures = new List<string>
        {
            "hr", // Croatian
            "en",  // English
        };

        public static string GetImplementedCulture(string name)
        {

            if (string.IsNullOrEmpty(name))
                return GetDefaultCulture();

            if (_validCultures.Count(c => c.Equals(name,
StringComparison.InvariantCultureIgnoreCase)) == 0)
                return GetDefaultCulture();

            if (_cultures.Count(c => c.Equals(name,
StringComparison.InvariantCultureIgnoreCase)) > 0)
                return name;
        }

        public static string GetDefaultCulture()
        {
            return _cultures[0]; // return Default culture
        }
        public static string GetCurrentCulture()
        {
            return Thread.CurrentThread.CurrentCulture.Name;
        }
    }
```

Kôd 6.3 Pomoćna klasa koju ova aplikacija upotrebljava za provjeru jezika

7. Slučajevi korištenja

Web aplikacija za spajanje ponude i potražnje na tržištu pravnih usluga u trenutno predviđa dvije osnovne grupe korisnika:

- Klijent – Ovo je korisnik koji postavlja upit na stranici, prihvaća odgovor i ocjenjuje odvjetnike koji su odgovorili na postavljena pitanja.
- Odvjetnik – Ovo je korisnik koji odgovara na pitanje koje su postavili klijenti.

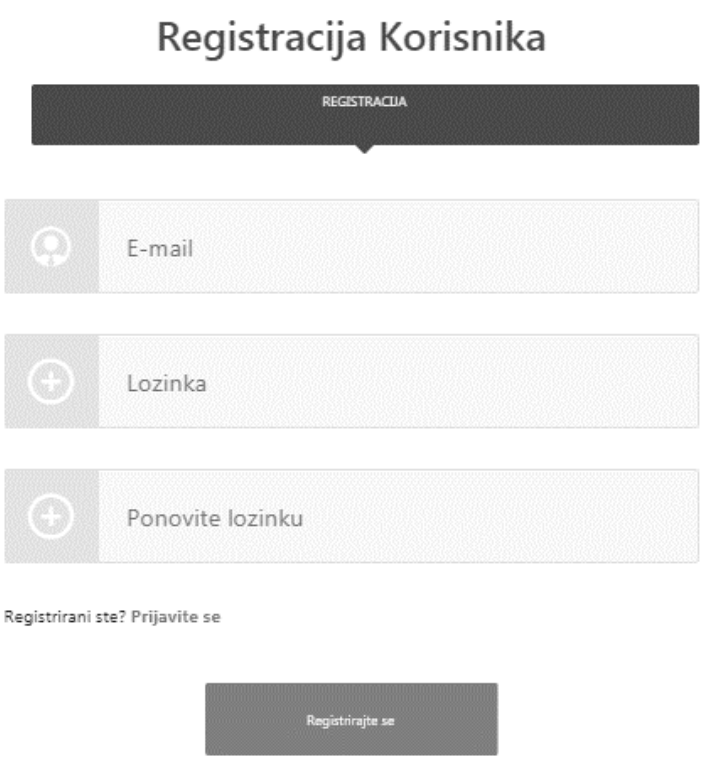
7.1. Slučajevi korištenja za klijenta

Klijent predstavlja osobu koja koristi web aplikaciju Advocate za postavljanje pitanja, pregledavanja odvjetničkih profila i ocjena, procjenjivanja odvjetničkih odgovora. Klijent na raspolaganju ima nekoliko osnovnih slučajeva korištenja koji su:

- Registracija
- Prijava
- Postavljanje pitanja
- Pregled pitanja i odgovori
- Prihvaćanje odgovora
- Ocjenjivanje najboljih odgovora i odvjetnika
- Zatvaranje pitanja

7.1.1. Registracija klijenta

Za registraciju klijenta potrebni su samo e-mail i lozinka. Stranica za registraciju klijenta dohvaća se GET zahtjevom /Account/RegisterCustomerUser. Odgovarajuća akcija je predznačena atributom [AllowAnonymous], koji omogućava pristup stranici svim korisnicima, nije potrebna bilo kakva autorizacija, rezultat akcije je vraćanje pogleda kao što je prikazano na slici (Slika 6.1).



Slika 7.1 obrazac za registraciju klijenta

Nakon popunjavanja obrasca i klikom na gumb “Registrirajte se” obrazac je validiran sa klijentske strane, pregledniku POST se šalje zahtjev iz preglednika, a akcija /Account/RegistracijaCustomerUser je pozvana. (Kôd 6.1)

```
        [HttpPost]
        [AllowAnonymous]
        [ValidateAntiForgeryToken]
        public async Task<ActionResult>
RegisterCustomerUser(CustomerViewModel model)
        {
            if (ModelState.IsValid)
            {
                try
                {
                    IdentityResult result = await
service.CreateCustomerUser(model);
                    if (result.Succeeded)
                    {
                        return RedirectToAction("Confirmation",
"Account", new { msg = "registracija-poslana" });
                    }
                    else
                    {
                        AddErrors(result);
                        ViewBag.errorMessage =
result.Errors.FirstOrDefault();
                        return View("Error");    }
                }
                catch (ValidationException e)
```

```
                {
                    ModelState.AddModelError("", e);
                    ViewBag.errorMessage = e.Message;
                    return RedirectToAction("Confirmation", "Account",
new { msg = e.Message });

                }
            }
            return View("Error");

        }
```

Kôd 7.1 Akcijska metoda za registriranje klijenta

RegisterCustomerUser model koji je proslijeđen kao parametar je validiran ako ima bilo kakve pogreške, a zatim servis.CreateCustomer(model) metoda je pozvana sa parametrom CustomerViewModel. Ako je korisnik uspješno kreiran, korisnik se preusmjerava na stranicu za prijavu. Ako model ima pogreške korisnik se preusmjerava natrag na prijavni obrazac (Slika 4.1) s odgovarajućom porukom zbog čega se račun nije mogao stvoriti.

7.1.2. Prijava korisnika

Stranica za prijavu registriranog korisnika dohvaća se GET zahtjevom /Account/LoginCustomer. Odgovarajuća akcija je predznačena atributom [AllowAnonymous], koji omogućava pristup stranici svim korisnicima, nije potrebna bilo kakva autorizacija, rezultat akcije je vraćanje pogleda prijava za korisnike.

```
        [AllowAnonymous]
        public ActionResult LoginCustomer(string returnUrl)
        {
            LoginViewModel loginVMOd = new LoginViewModel();
            ViewBag.ReturnUrl = returnUrl;
            return View(loginVMOd);

        }
```

Prije povrata pogleda u ViewBag.ReturnUrl sprema se znakovni niz returnUrl koji predstavlja URL stranice na koju će biti preusmjeren korisnik nakon uspješne prijave. Za dohvat taj URL nije od posebne važnosti, ali će biti pri POST zahtjevu. Stranica za prijavu sadrži formu za unos podataka (**Pogreška! Izvor reference nije pronađen.**) potrebnih za prijavu korisnika, a potrebni podaci su:

- Email (koji je ujedno i korisničko ime)
- Lozinka

Slika 7.2 Obrazac za prijavu korisnika

Nakon ispunjavanja podataka, pritiskom na gumb „Prijavite se“ šalje se POST zahtjev /Account/LoginCustomer (Kôd 7.2 Akcijska metoda LoginCustomer u Account controller-u

). Odgovarajuća akcija je predznačena [HttpPost] atributom koji označava da se akcija poziva POST zahtjevom. Predznačena je [ValidateAntiForgeryToken] atributom koji služi za zaštitu od krivotvorenja zahtjeva (ostali atributi objašnjeni su raniju u radu).

```
        [HttpPost]
        [AllowAnonymous]
        [ValidateAntiForgeryToken]
        public async Task<ActionResult> LoginCustomer(LoginViewModel
model, string returnUrl)
        {
            if (ModelState.IsValid)
            {
                try
                {
                    var user = await service.Authenticate(model.Email,
model.Password);

                    await SignInAsync(user, false);
                    if (service.GetCustomerRoleForUser(user.Id) == "N/A")
                    {
                        TempData["UnauthorizedMsg"] = "Please Login as a
User";
                        return RedirectToAction("Unauthorized", "Error");
                    }
```

```
                    else
                    {
                        Session["UserRole"] =
service.GetCustomerRoleForUser(user.Id);
                    }
                    return RedirectToAction("CustomerDashboard",
"Questions");
                }
                catch (ValidationException e)
                {
                    ModelState.AddModelError("", e.Message);
                    ViewBag.errorMessage = e.Message;
                    return View("Error");
                }
                catch (Exception e)
                {
                    ModelState.AddModelError("", "Internal server error.
Please contact support." + e);
                    ViewBag.errorMessage = e.Message;
                    return View("Error");
                }
            }

            return View(model);

        }
```

Kôd 7.2 Akcijska metoda LoginCustomer u Account controller-u

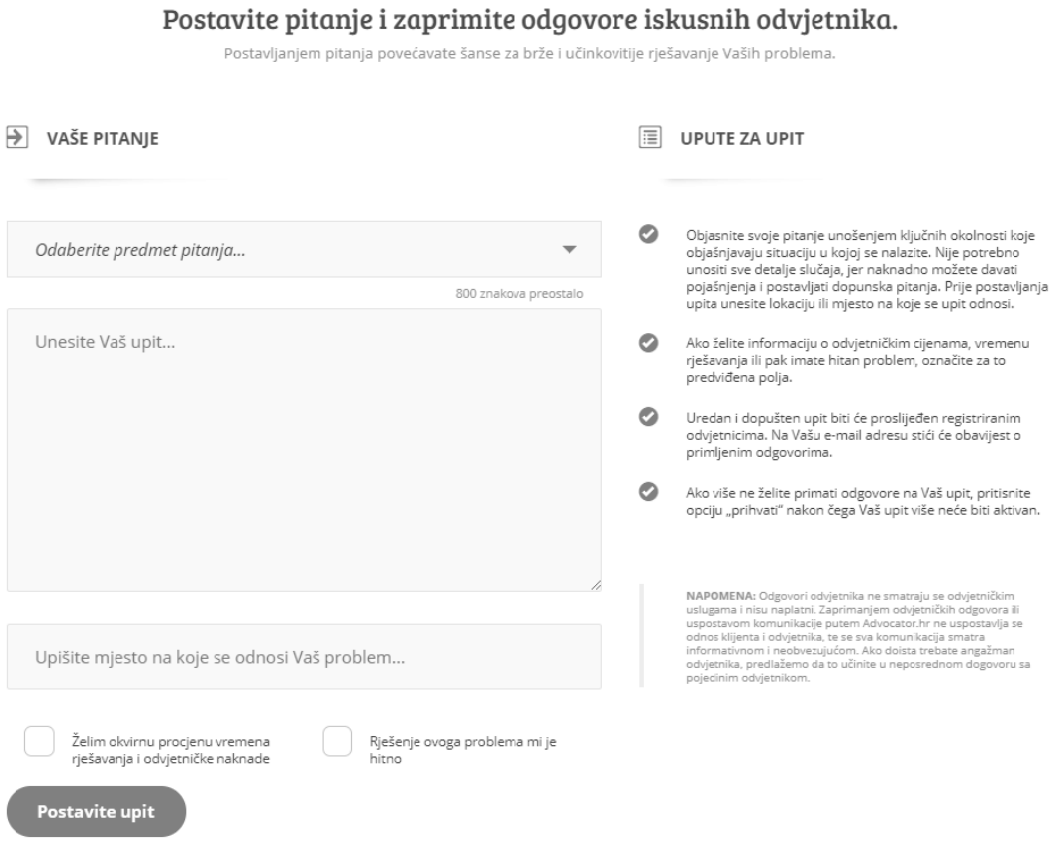
Ova akcija prvo provjerava valjanost modela kako se radi o LoginViewModel-u, vrši se provjera korisničkog email-a i lozinke. U definiciji modela (Kôd 3.5) su navedena polja predznačena s atributom [Required].

Tim atributom se osigurava da navedena polja u formi budu ispunjena, ukoliko korisnik ostavi prazna polja bit će obaviješten odgovarajućim porukama. Osim valjanosti modela akcija provjerava se i uspješnost prijave korisnika. Ukoliko model nije valjan ili ako prijava korisnika nije bila uspješna, korisnika se vraća na stranicu za prijavu korisnika uz odgovarajuće poruke zašto prijava nije bila uspješna. Ako je model valjan i prijava uspješna, korisnika se preusmjerava na returnUrl ukoliko je isti lokalni URL aplikacije. Ako nije lokalni URL aplikacije korisnika se preusmjerava na /Dashboard/Index (o tome brine funkcija RedirectToLocal).

Znakovni niz returnUrl je obično jednak URL-u zahtijevane stranice za koju korisnik nema potrebna prava pristupa. Nakon takvog zahtjeva se korisnika preusmjerava na stranicu za prijavu korisnika gdje se može prijaviti s korisničkim računom dovoljnih prava. Uobičajena vrijednost returnUrl-a je */Dashboard/Index* jer je to početna stranica aplikacije, a za koju je potrebno da je korisnik prijavljen. Stoga ukoliko korisnik nije prijavljen preusmjerava se na stranicu za prijavu korisnika gdje se može prijaviti važećim korisničkim računom.

7.1.3. Postavljanje upita

Da bi se postavio upit, potrebno je kliknuti na link „Pitaj Odvjetnika“ u glavnoj traci. Tada se korisniku prikazuje obrazac (Slika 6.3). Korisnik treba odabrati kategoriju prava na koje se pitanje odnosi, a zatim napisati pitanje koje se traži i ime grada u kojem je korisnik. Zatim korisnik može kliknuti na potvrdni okvir kako bi naznačio koliko hitno treba odgovoriti na pitanje te korisnik može kliknuti i na drugu potvrdnu kućicu kako bi naznačio želi li procjenu približnog vremena i cijenu. Da bi se pitanje poslalo, korisnik mora kliknuti na gumb “Postavite upit”. Ako je korisnik prijavljen, pop-up-om se pokazuje da je pitanje uspješno poslano, a korisnik je preusmjeren na *dashboard* (Slika 6.4) gdje se nalaze sva pitanja koja je korisnik poslao, prikazano kronološkim redoslijedom. Ako korisnik nije prijavljen ili registriran, korisnik se preusmjerava na obrazac za prijavu (Slika 6.2)



Slika 7.3 Obrazac za postavljanje upita

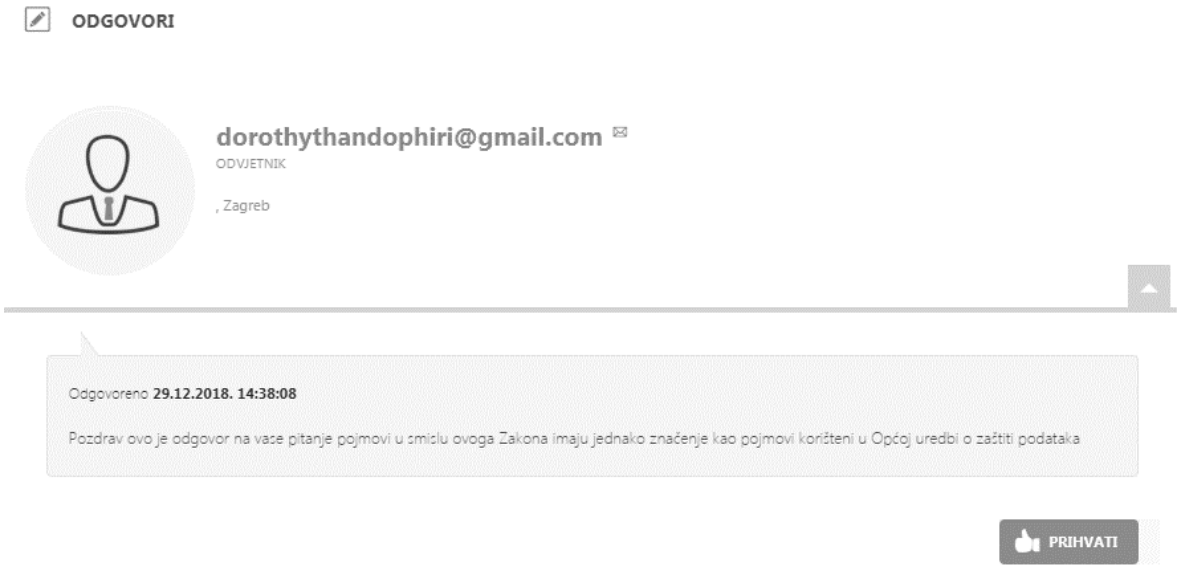
7.1.4. *Dashboard* klijenta

Na *dashboardu* klijenta (Slika 6.4) nalaze se sva pitanja aktivna i zatvorena koja je korisnik poslao prikazano kronološkim redoslijedom. Klikom na aktivna pitanja prikazuju se pitanja koja su u tijeku ili još nisu odgovorena. Isto tako korisnik može kliknuti „Zatvoreni upiti“ koji prikazuje pitanja koja su odgovorena i neaktivna.

Slika 7.4 *Dashboard* za klijenta

7.1.5. Pregled odgovora

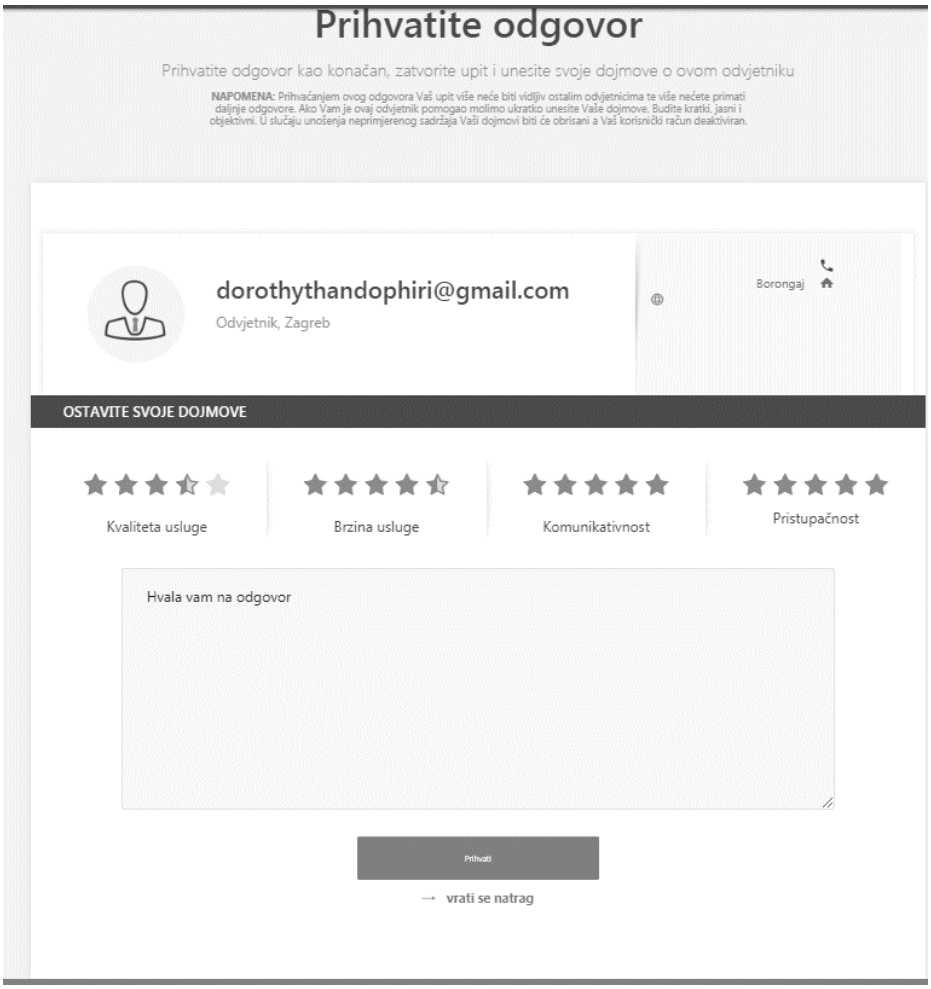
Pregled odgovora za aktivno ili zatvoreno pitanje koje je klijent postavio pitanje može se učiniti klikom na svako od pitanja. Ako postoje odgovori, prikazuju se i profili odvjetnika koji su odgovorili na to pitanje. Za pregled odgovora svakog odvjetnika korisnik može kliknuti na gumb sa strelicom za prikaz odgovora tog odvjetnika. (Slika 6.5)



Slika 7.5 Pregled odgovora od jednog odvjetnika

7.1.6. Prihvat odgovora

Za prihvaćanje odgovora nakon pregleda svih odgovora odvjetnika, klijent može prihvatiti samo jedan konačni odgovor s kojim je najviše zadovoljan. Klikom na gumb "Prihvati" (Slika 6.5) korisnika preusmjerava na stranicu "prihvati odgovor".



Slika 7.6 Prihvaćanje odgovora i ocjenjivanje odvjetnika

Na stranici prihvata odgovora korisnik može procijeniti odvjetnika s ocjenom od 1-5 (1 je najniža ocjena, a 5 najviša) prema sljedećim kategorijama: kvaliteta usluge, brzina usluge, komunikativnost, pristupačnost (Slika 6.6). Korisnik može ostaviti kratki komentar, a klikom na gumb zatvori korisnik konačno zatvora pitanje i preusmjerava se na *dashboard*.

7.1.7. Upravljanje profilom

Stranica za uređivanje korisničkog računa, odnosno promjena lozinke dohvaća se GET zahtjevom /Account/Manage koji zahtjeva da je korisnik prijavljen na web stranici. Ukoliko korisnik nije prijavljen preusmjerit će se na stranicu za prijavu (Slika 6.2) gdje se može prijaviti s valjanim korisničkim računom.

```
[Authorize]
public ActionResult ManageCustomer(ManageMessageId? message)
{
    ViewBag.StatusMessage =  message == ManageMessageId
        .ChangePasswordSuccess
    ? "Your password has been changed." : "";
```

```
        return View();
    }
```

Akcija postavlja statusnu poruku (ViewBag.StatusMessage) koja će u slučaju da je korisnik već uspješno promijenio lozinku sadržavati poruku o uspješnosti promjene, poruka će biti prikazana u pogledu. Ukoliko nije bilo promjene lozinke poruka će biti jednaka praznom nizu. Nakon postavljanja statusne poruke vraća pogled (Slika 4.5). Stranica sadrži formu za uređivanje korisničkog računa odnosno promjenu lozinke.

7.2. Slučajevi korištenja za odvjetnike

Osnovni slučaj korištenja za odvjetnike jest odgovaranje na aktivna pitanja od klijenata, čime odvjetnici eventualno mogu proširiti svoje baze postojećih klijenata.

7.2.1. Registracija odvjetnika

U svrhu provjere, odvjetnik mora upisati više informacija od klijenta. Za registraciju odvjetnika su potrebne sljedeće informacije: ime, prezime, e-mail, lozinka, poslovna adresa, OIB.

U upravljaču akcija Get request Account / RegisterOdvjetnikUser vraća pogled kao što je prikazano na slici (Slika 6.8) zajedno s modelom RegisterOdvjetnikViewModel. Pogled registracije odvjetnika također prikazuje uvjete i pravila koja odvjetnik mora pročitati i prihvatiti prije prijave na web stranicu.

Slika 7.7 Forma za registracija odvjetnika

Nakon popunjavanja obrasca i klikom na gumb „Registriraj se“, obrazac je potvrđen sa strane preglednika, a zatim se POST zahtjev šalje iz preglednika u kontrolor-u akcija /Account /RegisterOdvjetnikUser.

```
        [HttpPost]
        [AllowAnonymous]
        [ValidateAntiForgeryToken]
        public async Task<ActionResult>
RegisterOdvjetnikUser(RegisterOdvjetnikViewModel model)
        {
            if (ModelState.IsValid)
            {
                try
                {
                    IdentityResult result = await
service.CreateOdvjetnikUser(model);
                    if (result.Succeeded)
                    {
                        return RedirectToAction("Confirmation",
"Account", new { msg = "registracija-poslana" });
                    }
                    else
                    {
                        AddErrors(result);
                        ViewBag.errorMessage =
result.Errors.FirstOrDefault();
                        return View("Error");
                    }
                }
                catch (ValidationException e)
                {
                    ModelState.AddModelError("", e);
                    return RedirectToAction("Confirmation", "Account",
new { msg = e.Message });
                }
            }

            return View("Error");
        }
```

Kôd 7.3 Akcijska metoda za registriranje odvjetnika

Unutar akcijske metode RegisterOdvjetnikUser proslijeđen pogled-model je provjeren ako ima pogrešaka, a zatim servis.CreateOdvjetnikUser(model), metoda za stvaranje korisničkog računa, je pozvana i proslijeđen parametar validiran pogled-model (RegisterOdvjetnikViewModel). Ako je korisnički račun odvjetnika uspješno kreiran, korisnik se preusmjerava na stranicu Za Prijavu (**Pogreška! Izvor reference nije pronađen.**). Ako model ima pogreške, privatna metoda AddError se poziva s parametrom modela i tada se korisnik preusmjerava natrag na registracijski obrazac s odgovarajućom porukom zašto se račun nije mogao stvoriti.

```
        private void AddErrors(IdentityResult result)
```

```
{
    foreach (var error in result.Errors)
    {
        ModelState.AddModelError("", error);
    }
}
```

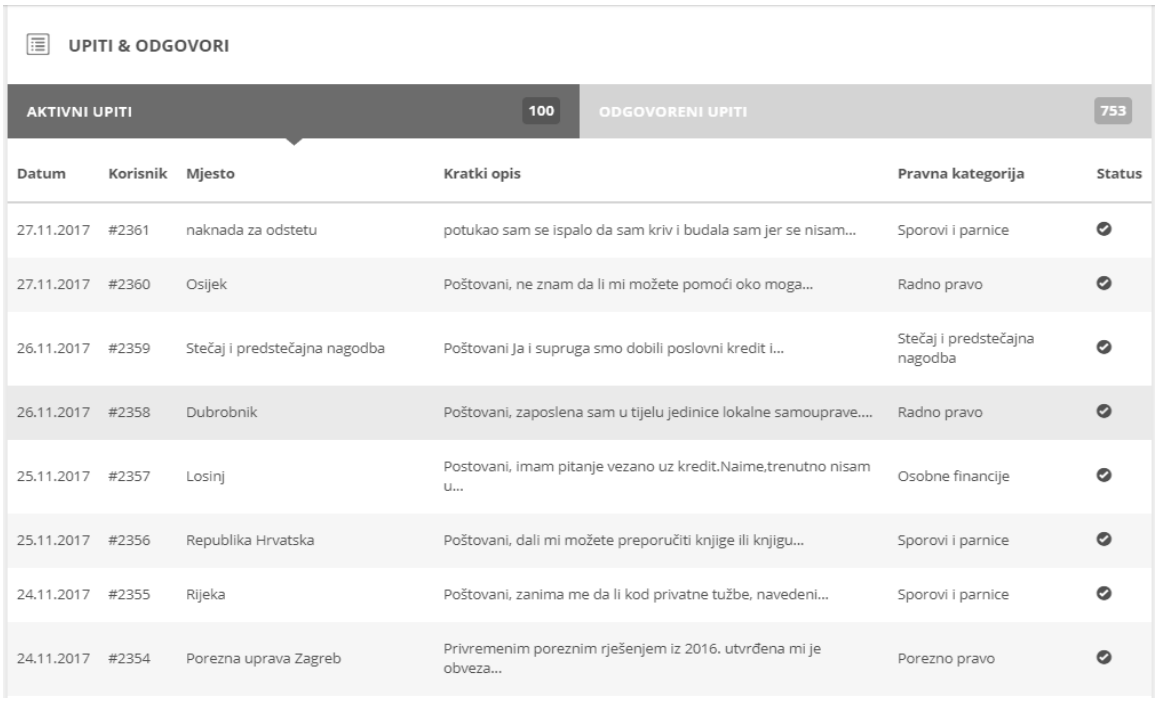
7.2.2. Prijava odvjetnika

Stranica za prijavu registriranog odvjetnika dohvaća se GET zahtjevom /Account/LoginOdvjetnikUser. Odgovarajuća akcija je predznačena atributom [AllowAnonymous], koji omogućava pristup stranici svim korisnicima, nije potrebna bilo kakva autorizacija, rezultat akcije je vraćanje pogleda prijave za odvjetnike (Slika 6.9)

Slika 7.8 Pogled prijavu za odvjetnike

7.2.3. Odgovaranje na upit

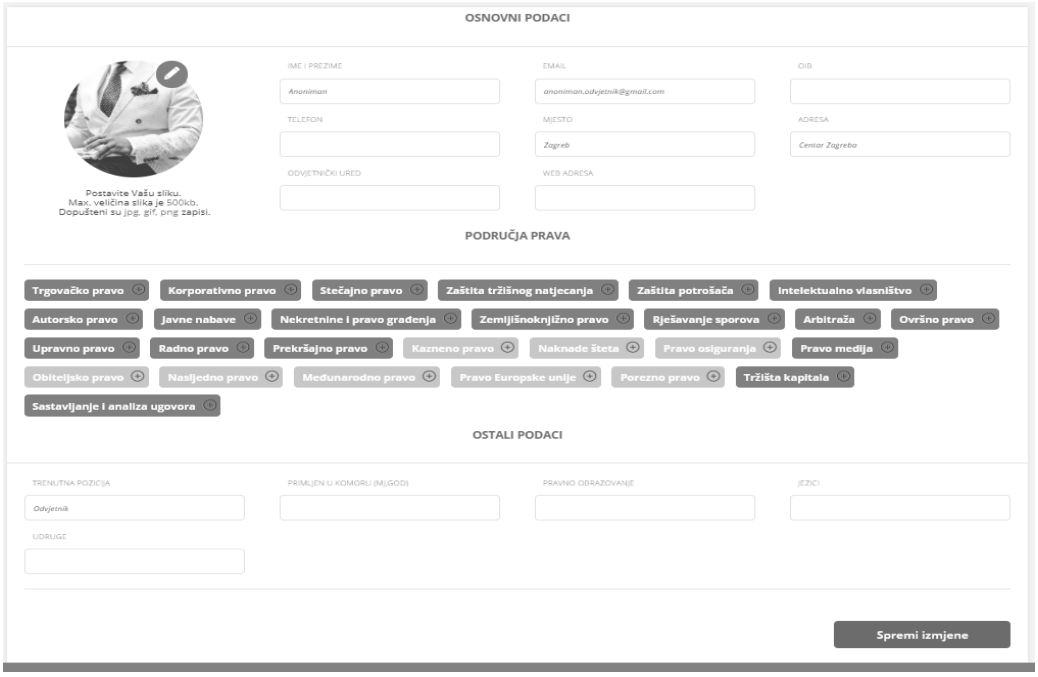
Kako bi odgovarao na pitanja, odvjetnik mora biti registriran na web stranici. Kada se odvjetnika prijavljuje, preusmjerava se na *dashboard* koji prikazuje popis aktivnih pitanja korisnika. (Slika 7.11)



Slika 7.9 Dashboard koji prikazuje popis pitanja

Kad odvjetnik klikne na aktivno pitanje, može ući u svako pravno pitanje, vidjeti tekst pitanja i ponuditi suradnju/odgovor. Odgovor odvjetnika je vidljiv samo od strane korisnika na čije je pitanje odvjetnik dao odgovor. Nakon što odvjetnik odgovori na pitanje, korisnik može zatvoriti pitanje ako je zadovoljan odgovorom.

7.2.4. Upravljanje profilom odvjetnika



Slika 7.10 Uređivanje profila odvjetnika

Logirani odvjetnik može kliknuti na padajući izbornik „Moj profil“. Ovaj izbornik omogućuje odvjetniku da u svoj profil unese detaljnije podatke o područjima prava kojima se bavi, i to odabirom unaprijed određenih područja prava kao posebnih kategorija. Odabirom jedne ili više kategorija i spremanjem izmjena, na profilu odvjetnika, koji mogu vidjeti korisnici, prikazati će se odabrane kategorije prava. Uz kategorije prava, odvjetnik također može unijeti ostale podatke poput datuma primitka u odvjetničku komoru, jezici koje govori, obrazovanje, kontakt podatke, te može *uploadati* fotografiju.

8. Plan testiranja

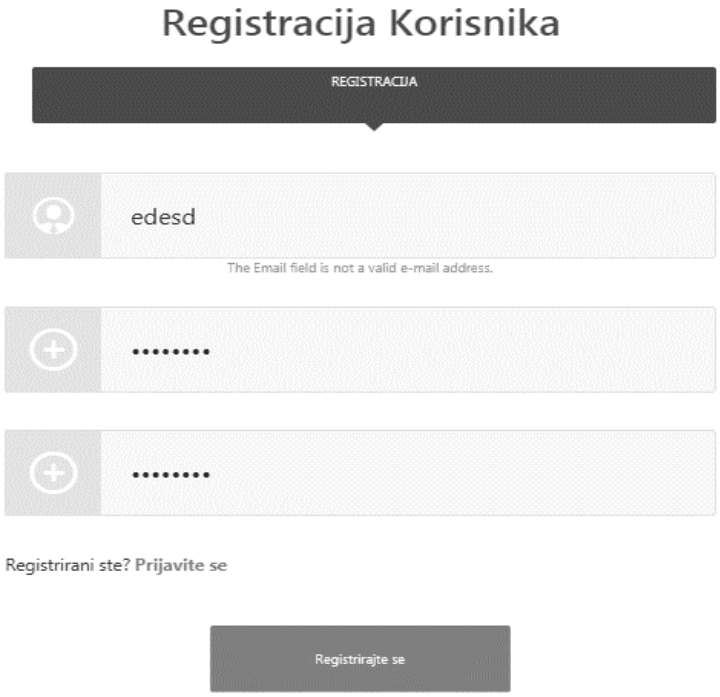
Testiranje aplikacije podrazumijeva testno korištenje aplikacije u svrhu ispitivanja osnovnih funkcionalnosti koje ona mora zadovoljavati vidljivo iz [5]. U ovom slučaju sa strane odvjetnika testira se registracija, prijava, odgovor na upit, uređivanje profila, promjena jezika i pregled aktivnih pitanja.

8.1. Testiranje sa strane klijenta

Sa strane klijenta testira se registracija, prijava, postavljanje pitanja, pregled odgovora, zatvaranje pitanja, ocjenjivanje odvjetnika, promjena jezika i promjena lozinke.

- **Registracija klijenta**

Klikom opcija „Registrirajte se“ korisnik pristupa na obrazac za registraciju korisnika (Slika 6.1). Korisnik upisuje email i lozinku nakon popunjavanja obrasca i odabire gumb “Registrirajte se” Ako su ispravno uneseni svi potrebni podaci, onda se uspješno šalje e-mail korisniku koji prikazuje poruku o uspješnoj registraciji. U slučaju da uneseni podaci nisu važeći ili su neispravnog formata, ispisuje se pogreška kao što je prikazano na slici (Slika 7.1).



Slika 8.1 Neispravan format email tijekom registracije korisnika

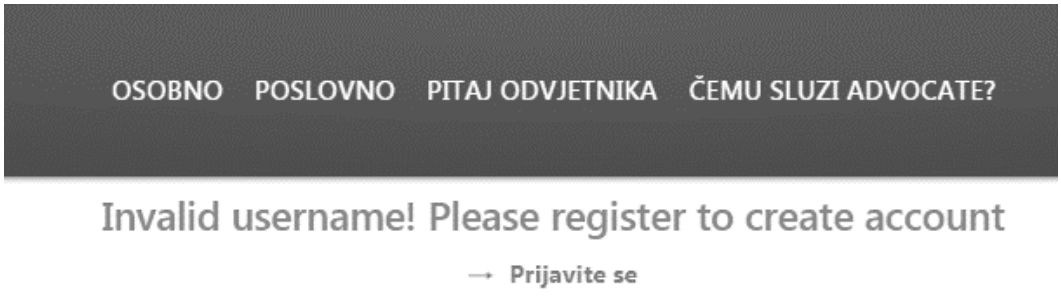
- **Prijava klijenta**

Klikom opcija „Prijavite se“ korisnik pristupa na obrazac za prijavu (Slika 6.2).Korisnik upisuje e-mail i lozinku klikom na gumb „Prijavite se“. Ako su ispravno uneseni e-mail i lozinka, te ako je korisnik registriran kao klijent, onda se uspješno pristupa *dashboardu* s prikazom svih aktivnih i zatvorenih pitanja tog klijenta. U slučaju da uneseni podaci nisu važeći ili su neispravnog formata, onda se prikazuje (Slika 7.2)

.

Slika 8.2 Prijava korisnika sa neispravnim podacima

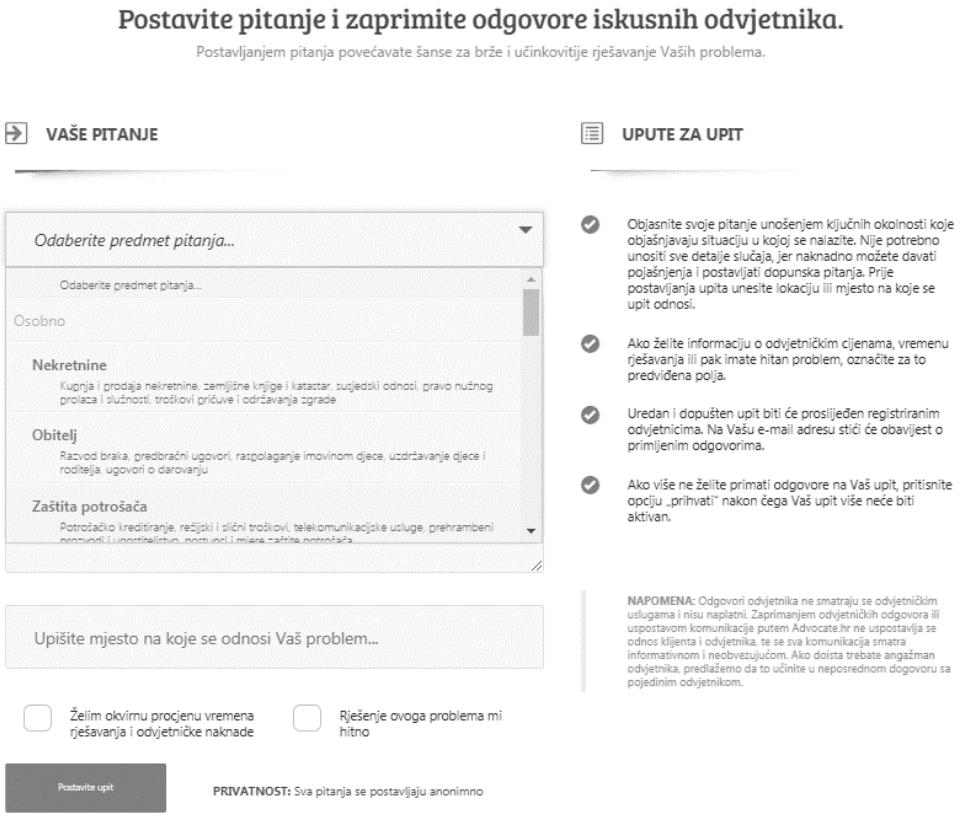
Ako se korisnik pokušava prijaviti s neregistriranom e-mailom, tada se preusmjerava na stranicu pogreške s odgovarajućom porukom(Slika 7.3)



Slika 8.3 Pokušaj prijave sa neregistriranim e-mailom

- **Postavljanje pitanja**

Korisnik može postaviti pitanja na dva osnovna načina. Prvi način je da se klikne na opciju „Postavi upit“ u glavnoj traci te se preusmjerava na obrazac za postavljanje pitanja u kojem može odabrati područje prava putem padajućeg izbornika. (Slika 7.4)



Slika 8.4 Obrazac za postavljanje upita s padajućim izbornikom s područjima prava

Drugi način je da korisnik odabere neko od ponuđenih područja prava u zaglavlju (Slika 7.5), kao osobno ili poslovno, te će tada obrazac za pitanje već automatski biti popunjen s odabranim područjem prava. Da bi postavio pitanje, korisnik mora biti logiran, a u slučaju da nije logiran, onda se preusmjerava na login stranicu za korisnike (Slika 6.1).

Slika 8.5 Područja prava u zaglavlju

- **Pregled odgovora**

Korisnik može vidjeti samo odgovore na svoja pitanja. Klikom na svako pitanje koje ima odgovore prikazuje sve profile odvjetnika koji su odgovorili na to pitanje. Za pregled odgovora svakog odvjetnika korisnik može kliknuti gumb sa strelicom za prikazuje odgovor tog odvjetnika. (Slika 6.5)

- **Zatvaranje pitanja**

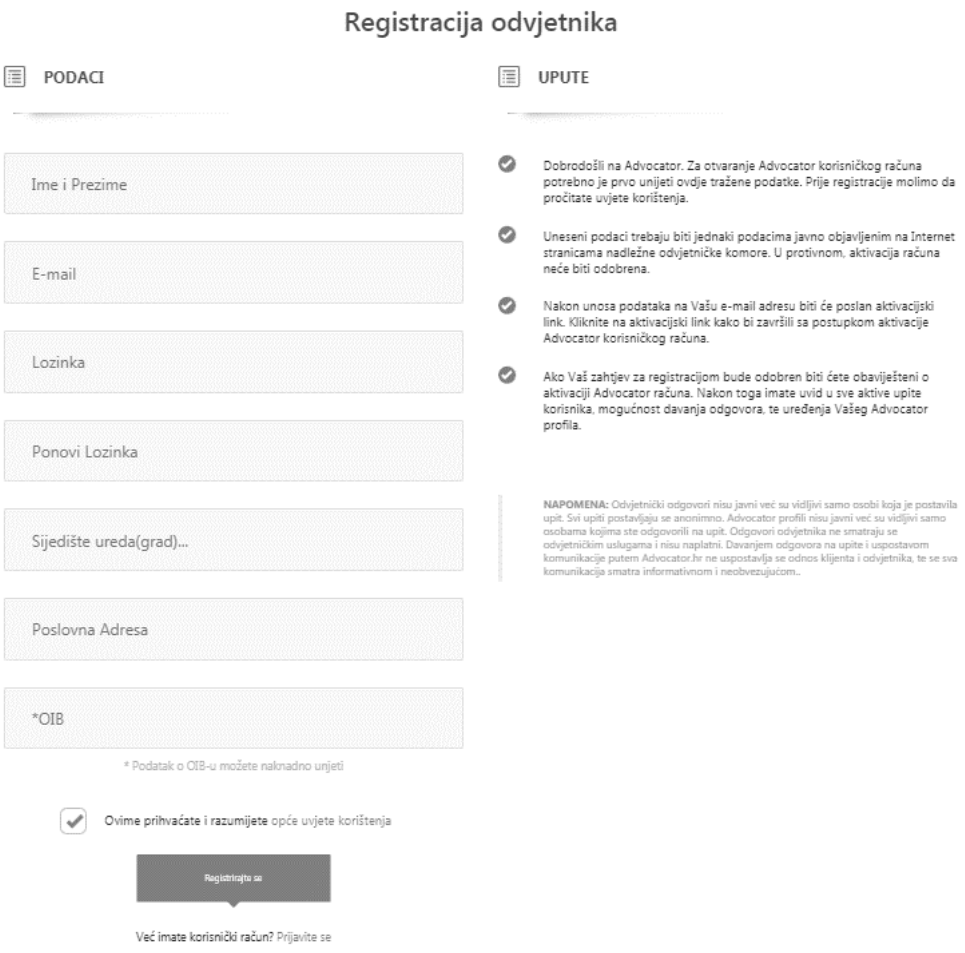
Da bi zatvorio pitanje, korisnik mora kliknuti gumb „Prihvati“ na jedan od ponuđenih odgovora te se preusmjerava na stranicu „Prihvati odgovor“ (Slika 6.6). Na stranici „Prihvati odgovor“ korisnik ocjenjuje odvjetnika ocjenama od 1-5 (1 je najniža, 5 je najviša) prema sljedećim kategorijama: kvaliteta usluge, brzina usluge, komunikativnost, pristupačnost (Slika 6.6). Korisnik može ostaviti kratki komentar i klikom na gumb „Zatvori“ korisnik zatvara pitanje.

8.2. Testiranje sa strane odvjetnika

U ovom slučaju sa strane odvjetnika testira se registracija, prijava, odgovor na upit i uređivanje profila.

- **Registracija odvjetnika**

Klikom na opciju „Registrirajte se“ za odvjetnike u glavnom izborniku, pristupa se obrascu za registraciju odvjetnika. Korisnik tj. odvjetnik upisuje ime, prezime, e-mail, lozinku, poslovnu adresu i OIB.



Slika 8.6 Obrazac za registraciju odvjetnika

Nakon popunjavanja obrasca i klikom na gumb „Registrirajte se“, ako su ispravno uneseni svi potrebni podaci, uspješno se šalje e-mail odvjetniku i prikazuje se poruka o uspješnoj registraciji (Slika 7.7). U slučaju da uneseni podaci nisu važeći ili su neispravnog formata, ispisuje se pogreška.

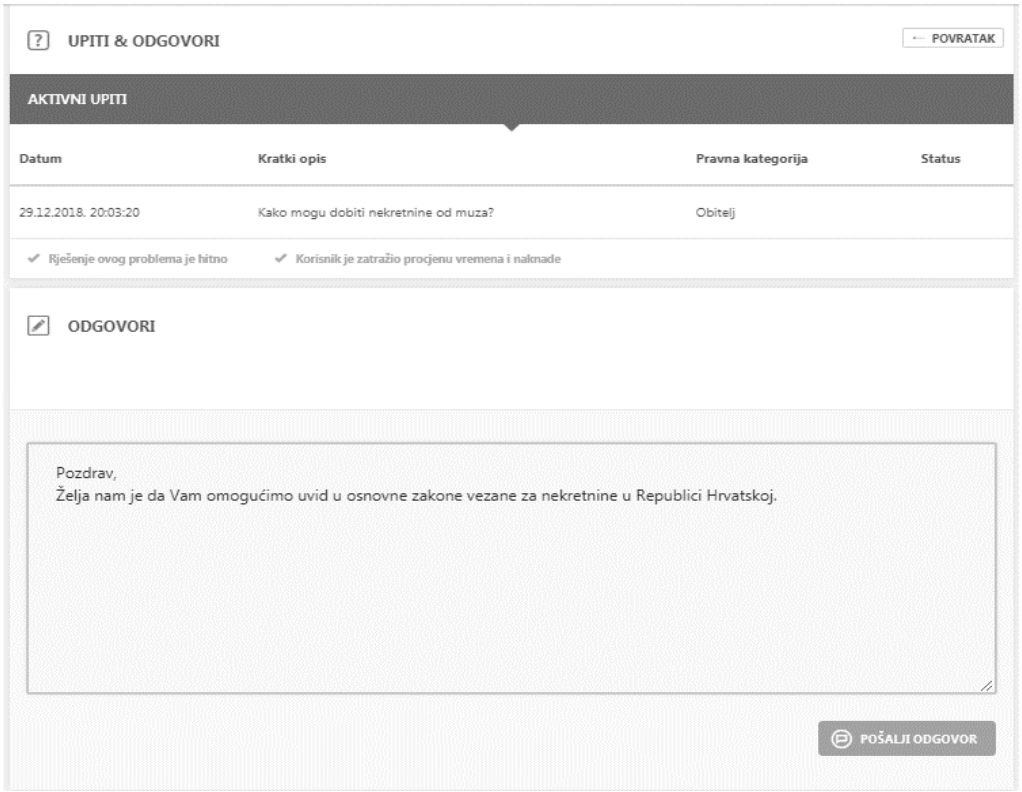
Slika 8.7 Poruka o uspješnoj registraciji korisnika

- **Prijava odvjetnika**

Za prijavu odvjetnika klikom na padajući izbornik „Za odvjetnike“ u glavnoj traci se preusmjerava na obrazac za prijavu za odvjetnike (Slika 6.2). Odvjetnik upisuje e-mail i lozinku ako je kao korisnik registriran. U tom slučaju odvjetnik uspješno pristupa *dashboardu* sa svim aktivnim pitanjima od svih klijenta i popisom pitanja na koje je već odgovorio (Slika 6.11). Ako se odvjetnik pokušava prijaviti s neregistriranim e-mailom ili ako se pokušava prijaviti s e-mailom klijenta, tada se preusmjerava na stranicu pogreške s odgovarajućom porukom (Slika 7.3)

- **Odgovor na upit**

Odvjetnik mora biti prijavljen kako bi odgovarao na pitanja. U slučaju da nije registriran preusmjerava se na stranicu za prijavu odvjetnika. Za odgovor na bilo koje aktivno pitanje, odvjetnik može kliknuti i ući u svako takvo pitanje, vidjeti tekst pitanja i ponuditi suradnju ili dati odgovor (Slika 7.8). Odgovor odvjetnika je vidljiv samo od strane korisnika čije je to pitanje te odvjetnika koji je dao odgovor.



Slika 8.8 Odgovor na upit

- **Uređivanje profila**

Logirani odvjetnik može kliknuti na padajući izbornik „Moj profil“. Ovaj izbornik omogućuje odvjetniku da u svoj profil unese detaljnije podatke o područjima prava kojima se bavi, i to odabirom unaprijed određenih područja prava kao posebnih kategorija (Slika 6.12). Uz kategorije prava, odvjetnik također može unijeti ostale podatke poput datuma primitka u odvjetničku komoru, jezike koje govori, obrazovanje, kontakt podatke, te može *uploadati* fotografiju. Dopušteni formati fotografije su jpg i png. Klikom na „Spremi izmjene“, ako su svi podaci ispravno uneseni, se spremaju izmjene i ažurira profil odvjetnika.

8.3. Promjena jezika

U ovoj aplikaciji zadani jezik je hrvatski. Promjena jezika na kojem korisnik želi koristiti aplikaciju se može učiniti klikom na padajući izbornik „Jezik“ u glavnoj traci (Slika 7.9). Ponuđeni jezik su engleski i hrvatski. Odabirom jednog od ovih jezika se mijenja cjelokupan statički tekst i forme na web stranici u odabrani jezik.

Slika 8.9 Padajući izbornik za promjenu jezika

9. Analiza ispunjenosti postavljenih ciljeva prema istraživanju tržišta

Funkcionalnosti aplikacije ispunjavaju sve postavljene ciljeve utvrđene istraživanje tržišta, i to: rješavanju problem velikog broja odvjetnika koji nemaju alat za pristup većem broju potencijalnih klijenata, rješavaju problem često netransparentnog određivanja cijena odvjetničkih usluga te rješavaju problem zakonski upitne konkurencije odvjetnicima od strane javnih bilježnika, knjigovođa i agenata za promet nekretninama.

S obzirom da aplikacija dopušta da se svaki odvjetnik u Hrvatskoj besplatno registrira i koristi aplikaciju, mali i samostalni odvjetnici bi trebali imati puno manje problema s konkurencijom od strane velikih odvjetničkih društava koja odrađuju i nabavljaju većinu unosnih odvjetničkih poslova. Korištenjem aplikacije jača se sloboda tržišta, a time i konkurentnost, što može imati velike pozitivne učinke na potrošače i korisnike odvjetničkih usluga, poglavito glede kvalitete pruženih usluga, a potencijalno i cijene. Povijesno je dokazano da jačanje konkurentnosti određene djelatnosti u pravilu ima pozitivne učinke i na samu tu djelatnost, na način da se ista racionalizira i specijalizira, što trenutno nije slučaj s hrvatskim odvjetništvom.

Što se tiče cijena, aplikacija je postavljena na način da svaka osoba s pravnim problemom može unaprijed zatražiti procjenu vremena rješavanja i visine odvjetničke naknade. Kako korisnik postavljanjem jednog upita na isti može dobiti više odvjetničkih odgovora, a time i više procjena vremena rješavanja i naknade, stvara se značajno bolji položaj potrošača da utvrdi realnu cijenu i vrijeme rješavanja njegovog problema. S druge strane, aplikacija oslobađa odvjetnika da više vremena provode rješavajući stvarne pravne probleme, umjesto da se moraju kontinuirano brinuti za održavanje postojećih i pronalazak novih klijenata. Aplikacija zapravo dovodi klijente direktno k odvjetnicima, tako da su pozitivni učinci aplikacije uzajamni.

Što se tiče nelojalne konkurencije javnih bilježnika, knjigovođa i agenata za promet nekretnina, ova aplikacija kao brza i početna kontaktna točka omogućuje građanima da brzo i efikasno zaobiđu takve neovlaštene pružatelje odvjetničkih usluga, te da izravno obrate odvjetnicima koji bi im, po samoj naravi njihove djelatnosti, trebali moći isporučiti značajno kvalitetnije i sigurnije dokumente, odnosno dati pravne savjete.

Zaključak

U ovom radu razrađene su korištene tehnologije .NET Framework, ASP.NET, ASP.NET Identity Framework, troslojna arhitektura aplikacije i MVC. Potom su razrađeni svi ključni elementi aplikacije Advocate, i to podatkovni sloj aplikacije, repozitorij aplikacije koji se povezuje na prezentacijski sloj aplikacije, te core aplikacije u kojoj se nalaze svi servisi aplikacije koji se pozivaju iz prezentacijskog sloja, odnosno upravljača. Sama aplikacija omogućuje spajanje ponude i potražnje za pravnim uslugama na način da se korisnici aplikacije dijele na administratore, klijente i odvjetnike. U bitnome, klijentima je omogućeno da postavljaju upite po određenim pravnim kategorijama, dok odvjetnici na iste upite mogu odgovarati, te ujedno imaju svoje profile s kontakt i drugim podacima. Korištenjem ove aplikacije moguće je znatno ubrzati i učiniti transparentnim proces pronalaska i pružanja pravnih usluga. Funkcionalnosti aplikacije ispunjavaju sve postavljene ciljeve utvrđene istraživanjem tržišta. S obzirom da aplikacija dopušta da se svaki odvjetnik u Hrvatskoj besplatno registrira i koristi aplikaciju, korištenjem aplikacije jača se sloboda tržišta, a time i konkurentnost, što može imati pozitivne učinke na potrošače i korisnike odvjetničkih usluga. Što se tiče cijena, stvara se značajno bolji položaj potrošača da utvrdi realnu cijenu i vrijeme rješavanja njegovog problema, s time istodobno aplikacija oslobađa odvjetnika da više vremena provode rješavajući stvarne pravne probleme. Vezano za konkurenciju ostalih djelatnosti, ova aplikacija omogućava isporuku značajno kvalitetnijih i sigurnijih dokumenata, kao i pravnih savjeta.

Popis Kratica

CLR *Common Language Runtime* softversko okruženje za izvršavanje kod

HTTP *Hyper Text Transfer Protocol* za postavljanje dvosmjerno komunikacije između server i aplikacija.

GUI *Grafic User Interface* grafičko korisničko sučelje

MVC *Model View Controller* softverska arhitektura

DAL *Data Access Layer* podatkovni sloj

CRUD *Create Read Update Delete* operacija nad entiteta

Popis Slika

Slika 2.2 Dohvaćanje dinamičke web stranice 5

Slika 2.4 arhitektura troslojne aplikacije 7

Slika 3.2 Inicijalizacija baza podataka 16

Slika 6.1 obrazac za registraciju klijenta 26

Slika 6.2 Obrazac za prijavu korisnika 28

Slika 6.3 Obrazac za postavljanje upita 30

Slika 6.4 *Dashboard* za klijenta 31

Slika 6.5 Pregled odgovora od jednog odvjetnika 32

Slika 6.6 Prihvaćanje odgovora i ocjenjivanje odvjetnika 33

Slika 6.8 Forma za registracija odvjetnika 34

Slika 6.9 Pogled prijavu za odvjetnike 36

Slika 6.11 Dashboard koji prikazuje popis pitanja 37

Slika 6.12 Uređivanje profila odvjetnika 38

Slika 7.1 Neispravan format email tijekom registracije korisnika 39

Slika 7.2 Prijava korisnika sa neispravnim podacima 40

Slika 7.3 Pokušaj prijave sa neregistriranim e-mailom 40

Slika 7.4 Obrazac za postavljanje upita s padajućim izbornikom s područjima prava 41

Slika 7.5 Područja prava u zaglavlju 42

Slika 7.6 Obrazac za registraciju odvjetnika 43

Slika 7.7 Poruka o uspješnoj registraciji korisnika 43

Slika 7.8 Odgovor na upit 44

Slika 7.9 Padajući izbornik za promjenu jezika 45

Popis tablica

Tablica 3.1 Opis modela 14

Tablica 4.1 Akcije (GET metoda) s vezanim pogledom 20

Tablica 4.2 Akcije (POST metoda) s vezanim pogledom 20

Popis kôdova

Kôd 5.2 CustomerViewModel s globaliziranim validacijskim porukama 22

Kôd 5.3 Akcijska metoda za promjenu i spremanje odabranog jezika u kolačić 23

Kôd 5.4 Pomoćna klasa koju ova aplikacija upotrebljava za provjeru jezika 24

Kôd 6.1 Akcijska metoda za registriranje klijenta 27

Kôd 6.2 Akcijska metoda LoginCustomer u Account controller-u 29

Kôd 6.4 Akcijska metoda za registriranje odvjetnika 35

Literatura

[1] Microsoft, Introduction to Entity Framework, 2016., https://msdn.microsoft.com/enus/library/aa937723(v=vs.113).aspx

[2] ASP.NET Identity tutorial, https://www.tektutorialshub.com/asp-net/asp-net-identity-tutorial-basics/

[3] Entity Framework Tutorial, Context Class in Entity Framework, http://www.entityframeworktutorial.net/basics/context-class-in-entityframework.aspx

[4] Globalizing and localizing .Net Applications, https://docs.microsoft.com/en-us/dotnet/standard/globalization-localization/

[5] Test basis, Guru 99, https://www.guru99.com/test-analysis-basis.html

[6] Kodeks odvjetničke etike, Zakon o odvjetništvu, Pravilnik o web stranici i Pravilnik o reklamiranju odvjetnika, Odvjetnička tarifa, izvor na http://www.hok-cba.hr/hr/dokumenti

[7] MojPosao.hr, prosječna plaća u kategoriji pravo https://www.moj-posao.net/Posao/412107/Odvjetnicki-vjezbenik-mz/

[8] Državni zavod za statistiku, zaposleni prema djelatnostima u studenom 2018. godine, priopćenje br. 9.2.1./11 od 24. prosinca 2018. godine, izvor: https://www.dzs.hr/

[9] Državni zavod za statistiku, broj i struktura poslovnih subjekata u ožujku 2018. godine, priopćenje br. 11.1.1/11, od 9. svibnja 2018. godine, izvor: https://www.dzs.hr/Hrv_Eng/publication/2018/11-01-01_01_2018.htm

[10] Večernji.hr, 12. studenog 2014. godine, intervju predsjednika HOK-a o broju blokiranih odvjetnika, izvor: https://www.vecernji.hr/vijesti/svake-godine-u-registar-ude-300-novih-odvjetnika-972884; ujedno i Express.hr, članak: https://www.express.hr/top-news/hrvatska-je-mala-bara-za-toliko-krokodila-14515

[11] Index.hr, članak o top listi odvjetnika https://www.index.hr/vijesti/clanak/top-10-odvjetnika-hanzekovic-i-dalje-prvi-prema-prihodima/977638.aspx, uvid u službene stranice Ministarstva pravosuđa, sudski registar na https://sudreg.pravosudje.hr/registar/f?p=150:1,