IZRADA AUTO OGLASNIKA KORIŠTENJEM RAZVOJNOG OKVIRA DJANGO

Laušić, Josip

Undergraduate thesis / Završni rad

2021

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Split / Sveučilište u Splitu**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:228:515226

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Repository of University Department of Professional Studies

zir.nsk.hr

UNIVERSITY OF SPLIT

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

JOSIP LAUŠIĆ

Z A V R Š N I R A D

IZRADA AUTO OGLASNIKA KORIŠTENJEM RAZVOJNOG OKVIRA DJANGO

Split, rujan 2021.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

Predmet: Programiranje na internetu

Z A V R Š N I R A D

Kandidat: Josip Laušić

Naslov rada: Izrada auto oglasnika korištenjem razvojnog okvira Django

Mentor: Marina Rodić, predavač

Split, rujan 2021.

Sadržaj

Sažetak 1
Summary 1
1. Uvod 2
2. Korištene tehnologije 4
2.1. Django 4
2.2. MySQL 4
2.3. jQuery 5
2.4. Bootstrap 5
3. Instalacija programa i alata potrebnih za pokretanje aplikacije 7
4. Opis aplikacije 9
5. Implementacija i izgled aplikacije 11
5.1. Modeli 11
5.2. Prikazi i predlošci 12
6. Zaključak 24
Literatura 25

Sažetak

Cilj ovog završnog rada je izrada web aplikacije za oglašavanje prodaje automobila. Web aplikacija izrađena je u razvojnom okviru (engl. *framework*) Django, koji je pisan programskim jezikom Python. Naglasak je stavljen na napredno filtriranje (po marki automobila, po modelu automobila, po lokaciji automobila, po cijeni automobila, po godini proizvodnje automobila, po broju prijeđenih kilometara automobila, po vrsti motora automobila, po prikazu auto oglasa samo sa slikama, itd.). Više navedenih kriterija može se istovremeno primijeniti. Također je omogućeno pretraživanje po naslovu auto oglasa te sortiranje po raznim parametrima.

Aplikaciji mogu pristupiti neregistrirani i registrirani korisnici. Registrirani korisnici mogu imati administratorska prava i prava oglašivača auto oglasa.

Ključne riječi: auto oglasnik, Django, Python, web aplikacija

Summary

Car advertising application development using Django framework

The purpose of this final paper is to create a web application for advertising car sales. The web application was developed in the Django framework which is written in the Python programming language. Emphasis is placed on advanced filtering (by car brand, by car model, by car location, by car price, by car production year, by car mileage, by type of car engine, by the display of car ads with images only, etc.). Several of the above criteria can be applied simultaneously. It is also possible to search by the title of the car ad and sort by various parameters.

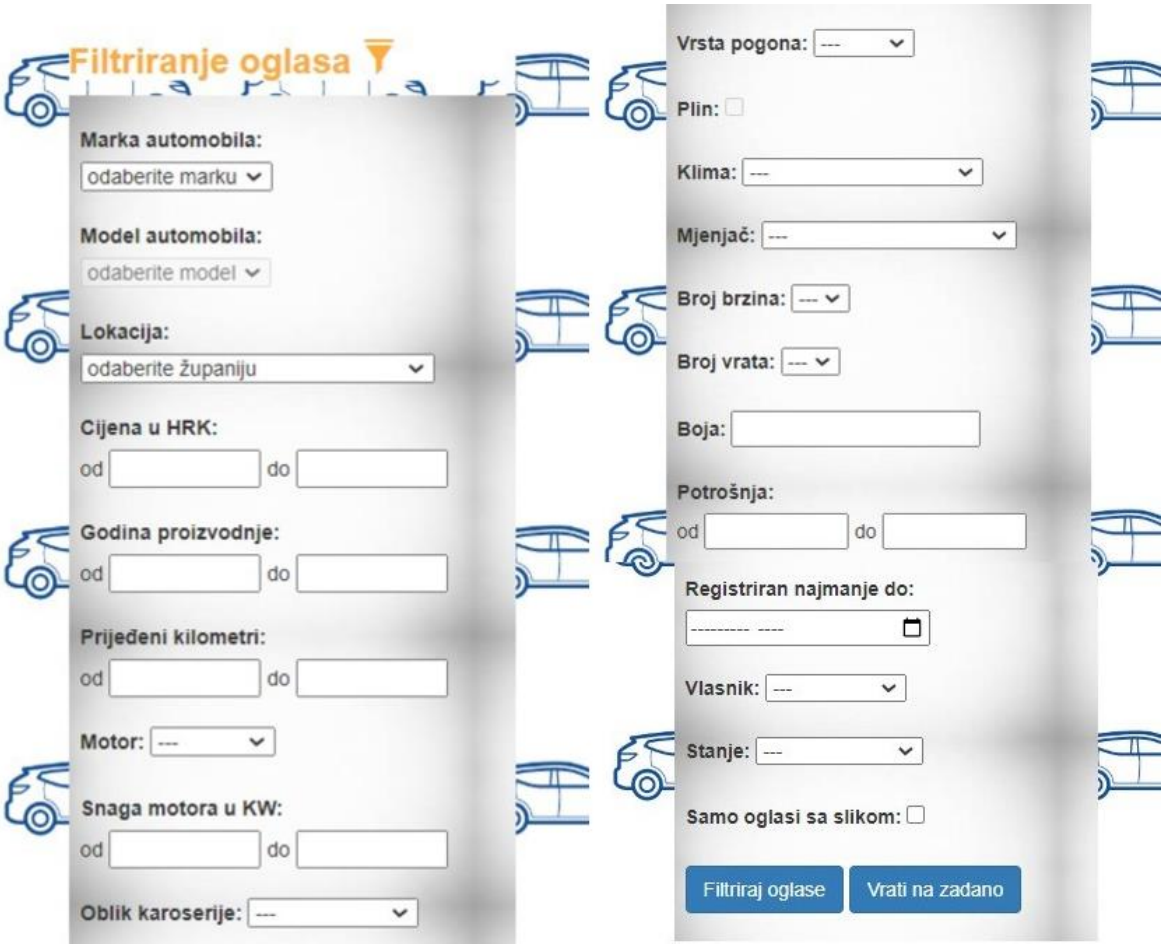
The application can be accessed by unregistered and registered users. Registered users can have administrator rights and car advertiser rights.

Keywords: car advertising, Django, Python, web application

1. Uvod

U ovom završnom radu izrađena je web aplikacija, auto oglasnik, koja omogućuje oglašivačima jednostavnu prodaju automobila te zainteresiranim kupcima dostupnost auto oglasa. Unutar ove web aplikacije postoji mogućnost pretraživanja postojećih i objavljivanja novih auto oglasa. Aplikaciji mogu pristupiti registrirani i neregistrirani korisnici tj. gosti. Korisnici se dijele na one s administracijskim pravima i na one s običnim tj. s pravima kupaca i prodavača automobila.

Objavljeni auto oglasi, uz pretraživanje, mogu se filtrirati i sortirati. Na slici 1 vidljivi su parametri po kojima je omogućeno filtriranje objavljenih auto oglasa.



Slika 1: Filtriranje auto oglasnika

Auto oglasi mogu biti aktivirani i deaktivirani. Prilikom objave novog auto oglasa, isti je automatski aktivan te u slučaju da ga oglašivač (ili administrator) u međuvremenu ne deaktivira, njegova aktivnost traje mjesec dana. Korisnici aplikacije vide samo aktivne auto oglase, a vlasnik auto oglasa i administrator mogu vidjeti i deaktivirane. Auto oglasi mogu biti obrisani od strane korisnika koji ga je objavio ili od strane administratora aplikacije. Prilikom brisanja auto oglasa, iz baze podataka kaskadno se brišu svi vezani podaci.

Ovaj rad sadrži pet poglavlja. Nakon uvodnog slijedi drugo poglavlje „Korištene tehnologije“ u kojem se opisuju tehnologije korištene pri izradi ovog rada. U trećem poglavlju govori se o instalaciji aplikacija i alata potrebnih za pokretanje ove web aplikacije. U četvrtom poglavlju „Opis aplikacije“, aplikacija se opisuje kroz uloge (engl. *roles*) korisnika i mogućnosti njihovih uloga. U petom poglavlju „Implementacija i izgled aplikacije“, opisuje se implementacija aplikacije kroz modele (engl. *models*), prikaze (engl. *views*) i predloške (engl. *templates*) te se kroz slike i opis prikazuje izgled aplikacije. Šesto poglavlje je zaključak.

2. Korištene tehnologije

2.1. Django

Django je razvojni okvir otvorenog kôda koji koristi MVT (engl. *Model View Template*) arhitekturu. Napravili su ga Adrian Holovaty i Simon Willison 2003. godine. MVT arhitektura je vrsta arhitekture u kojoj model predstavlja bazu podataka (pristup podacima i njihova obrada), prikaz predstavlja funkcije upravljanja aplikacijom te predložak predstavlja korisničko sučelje.

Django je pisan u Pythonu, programskom jeziku otvorenog kôda koji podržava nekoliko vrsta programiranja kao što su objektno-orijentirano, aspektno i strukturirano orijentirano programiranje.

Django prilikom izrade web aplikacija pruža programeru brzinu i jednostavnost, siguran razvoj te uz to što je prilagodljiv razvojima različitih vrsta web aplikacija, također je pristupačan za održavanje istih.

Kreiranje projekta se izvodi preko naredbenog retka (engl. *Command Prompt*). Prvi korak je instalacija Pythona, zatim kreiranje direktorija u koji će se spremati projekt. Iza toga se pokreće virtualno okruženje venv u kojem se preko Python upravitelja paketa pip instalira Django te ostali zavisni paketi. Posljednji korak kreiranja projekta završava naredbama `startproject` i `startapp`. Nakon konfiguracije unutar „settings.py“ datoteke moguće je pokrenuti web aplikaciju. Web aplikacija se pokreće naredbom `runserver` koja se nalazi u upraviteljskoj datoteci „manage.py“.

2.2. MySQL

MySQL je sustav otvorenog kôda za upravljanje relacijskom bazom podataka. Za upravljanje relacijskom bazom podataka koristi SQL (engl. *Structured Query Language*) jezik.

Kao podršku za spajanje na MySQL modul korišten je XAMPP (engl. *X* (bilo koji od četiri operativna sustava), *Apache, MySQL, PHP and Perl*) web server otvorenog koda razvijen od strane Apache Friends. Za upravljanje MySQL relacijskom bazom podataka

korišten je phpMyAdmin. Ovaj softver je napisan u programskom jeziku PHP, a namijenjen je za upravljanje MySQL i MariaDB bazama podatka preko interneta.

2.3. jQuery

jQuery je biblioteka otvorenog koda napisana programskim jezikom JavaScript. Služi za lakši razvoj web stranica te se koristi unutar HTML dokumenta tj. unutar HTML elementa <script>.

Koristi se kako bi se s klijentske strane omogućila veća interakcija i privlačniji izgled aplikacije. Unutar ovog završnog rada najčešće je korištena u interakciji s korisnikom aplikacije u vidu raznovrsnih ograničenja ili zabrana koja popratno slijede prije ili nakon neke radnje korisnika.

2.4. Bootstrap

Bootstrap je razvojni okvir otvorenog kôda koji kao spoj JavaScripta, HTMLa i CSSa služi za dizajniranje korisničkog sučelja aplikacija.

HTML (HyperText Markup Language) je jezik koji služi za izradu web stranica. Dokument izrađen pomoću HTML elemenata predstavlja web stranicu čiji prikaz omogućuje web preglednik. Svaki HTML dokument na početku mora imati element <!DOCTYPE>, zatim <html> element koji se dijeli na <head> i <body>.

CSS (Cascading Style Sheets) je stilski jezik, koji služi za uređivanje izgleda i prikaza HTML elemenata.

Bootstrap je fleksibilan i lagan za korištenje te je zbog toga učestao alat pri dizajniranju web aplikacija. Korištenjem Bootstrap rešetkastog sustava (engl. *grid system*), dizajn aplikacije postaje prilagodljiv veličini ekrana uređaja. Rešetkasti sustav se dijeli na četiri vrste, a to su xs za mobilne telefone, sm za tablete, md za ekran srednje veličine te lg za veće ekrane. Rešetkasti sustav radi tako da se ekran dijeli na neograničen broj redaka koji se sastoji od dvanaest stupaca. Navedeni redci i stupci nalaze se unutar jednog kontejner (engl. *container*). Svi stupci ne moraju biti iskorišteni te moguće je spojiti nekoliko stupaca

u jedan. Moguće je kombinirati više vrsta oznaka za veličinu te se tako dobije prilagodljivost različitim ekranima.

Na ispisu 1 može se vidjeti primjer rešetkastog sustava s dva retka. U prvom retku su zadane širine osam i četiri (spojeni stupci), sveukupno dvanaest. U drugom retku širina stupaca se sama prilagođava, u ovom slučaju na četiri jer je dvanaest podijeljeno na tri stupca jednako četiri.

```
<div class="container">
      <div class="row">
        <div class="col-sm-8">col-sm-8</div>
        <div class="col-sm-4">col-sm-4</div>
      </div>
      <div class="row">
        <div class="col-sm">col-sm</div>
        <div class="col-sm">col-sm</div>
        <div class="col-sm">col-sm</div>
      </div>
</div>
```

Ispis 1: Primjer rešetkastog sustava Bootstrapa

3. Instalacija programa i alata potrebnih za pokretanje aplikacije

Programi i alati potrebni za pokretanje aplikacije su: Python, Django, Visual Studio Code te XAMPP.

Verzija Pythona korištena za izradu ove aplikacije je Python 3.9.2. Ta inačica izdana je 19. veljače 2021. godine. Instalacijska datoteka može se preuzeti s poveznice https://www.python.org/downloads/release/python-392/.

Instalacija Pythona je vrlo jednostavna te se odvija preko čarobnjaka za instalaciju koji se pokreće preuzetom .exe datotekom. Odabiru se potrebne značajke za instalaciju, a za ovaj projekt bitno je odabrati instalaciju pip upravitelja paketa. U daljnjoj instalaciji odabire se opcija dodavanja Python varijable okruženja te ukoliko ima potrebe mijenja se predložena lokacija instalacije. Po završetku instalacije Python je spreman za korištenje.

Instalacija Djanga opisala se u poglavlju „Korištene tehnologije“. Nakon namještanja virtualnog okruženja venv naredbom `pip install Django` te nakon postavljanja potrebne konfiguracije, Django je spreman za korištenje. Verzija Djanga korištena u izradi ove aplikacije je Django 3.2.2.

Preko pip upravitelja paketa instaliran je paket mysqlclient verzije 2.0.3 koji služi za komunikaciju Djanga s bazom podataka. Popis svih instaliranih paketa može se vidjeti u datoteci „requirements.txt“ unutar direktorija projekta koja je napravljena preko naredbe `pip freeze`.

Visual Studio Code je uređivač kôda koji pruža mogućnost izrade aplikacija. Može se koristiti na više operacijskih sustava kao što su Windows, Linux, macOS. Koristi se kao uređivač kôda za razne programske jezike kao na primjer Python, Java, Node.js, PHP, Go i slično. Instalacijska datoteka može se preuzeti s poveznice https://code.visualstudio.com/.

Nakon preuzimanja pokreće se čarobnjak za instalaciju gdje se odabire mjesto instalacije te je tim korakom instalacija gotova. U programu s lijeve strane nalazi se izbornik s ekstenzijama te je potrebno instalirati ekstenziju Python koja omogućuje provjeru sintakse Python jezika.

Za pristup bazi podataka koristi se XAMPP program, kojeg je moguće preuzeti s poveznice https://www.apachefriends.org/index.html. Instalacija se odvija preko čarobnjaka u kojem se odabiru željene komponente, potrebne za ovaj projekt su Apache i mySQL. Sijedi odabir lokacije instalacije te je s tim korakom instalacija završena.

Kako bi se aplikacija povezala s bazom podataka, koja se nalazi na mySQL modulu XAMPP programa, potrebno je konfigurirati informacije o bazi podataka kao što su: ime baze podatka, korisničko ime, lozinka, ime poslužitelja i port. Baza podataka stvara se tako da se pokrene migracija nad modelima aplikacije. Migracija se izvodi preko naredbi `makemigrations` i `migrate`.

Nakon što su sve ove navedene radnje izvršile, korisnik može pokrenuti aplikaciju naredbom `python manage.py runserver`.

4. Opis aplikacije

Aplikacija služi kao alat kojim se služe kupci i prodavači automobila. Nudi mogućnost pretraživanja postojećih i objavljivanje novih auto oglasa. Aplikaciji mogu pristupiti registrirani i neregistrirani korisnici tj. gosti. Unutar aplikacije postoje dvije uloge koje dijele prava korisnika na administratorska i na obična tj. na prava kupaca i prodavača automobila.

Neregistrirani korisnik tj. gost aplikacije ima mogućnosti pretraživanja objavljenih auto oglasa po naslovu, sortiranja istih po datumu objavljivanja auto oglasa, po cijeni automobila, po najpopularnijim rezultatima koji ovise o broju pregleda auto oglasa, itd. Također, neregistrirani korisnik ima mogućnost filtriranja objavljenih auto oglasa po marki automobila, po modelu automobila, po lokaciji automobila, po cijeni automobila, po godini proizvodnje automobila, po prikazu auto oglasa samo sa slikama, itd. Uz to neregistrirani korisnik može i pregledavati profil oglašivača (prodavača) na kojem vidi osobne podatke o oglašivaču te listu svih aktivnih oglasa oglašivača.

Pri registraciji aplikacija od korisnika traži korisničko ime, adresu elektroničke pošte, lozinku te ponovljenu lozinku. Pri prijavi aplikacija od registriranog korisnika traži korisničko ime te lozinku.

Registrirani korisnik aplikacije, uz mogućnosti koje ima neregistrirani korisnik, ima mogućnosti uređivanja osobnih podataka (ime, prezime, broj mobitela, adresu stanovanja, slika profila te promjena lozinke), predaje novog auto oglasa, pregleda aktivnih i neaktivnih osobnih auto oglasa te njihovo uređivanje, aktiviranje, deaktiviranje i brisanje. Auto oglas se automatizmom nakon isteka roka od mjesec dana deaktivira.

Administrator aplikacije ima sve mogućnosti kao i registrirani korisnik aplikacije, ali nema pravo predaje auto oglasa. U skladu s tim nema mogućnost pregleda osobnih auto oglasa te ostalih radnji vezanih za iste, ali ima mogućnost pregleda, uređivanja, aktiviranja, deaktiviranja i brisanja svih auto oglasa aplikacije. Administratoru je omogućen prikaz svih registriranih korisnika aplikacije gdje ima uvid u njihove korisničke račune i njihove aktivne i neaktivne auto oglase. Administrator ima pristup dijelu aplikacije koji se unutar Djanga naziva administratorsko sučelje. Ono daje mogućnost stvaranja novih te uređivanja i brisanja postojećih zapisa o korisnicima i o skupu podataka koji su dio auto oglasa (marka

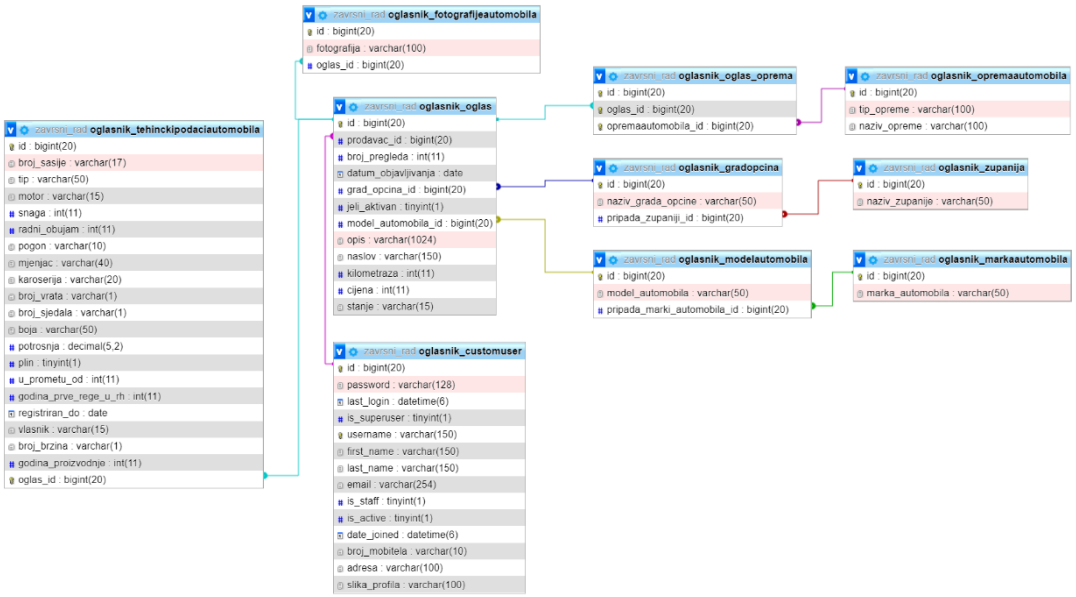
automobila, model automobila, oprema, županija u kojoj se nalazi automobil te grad/općina). Administrator aplikacije vodi brigu o aplikaciji na način da je odgovoran za korisnike (može aktivirati i deaktivirati njihove korisničke račune), dodavanje marki i vezanih modela automobila, praćenje trendova vezanih za pojedine dijelove auto oglasa koje korisnik prilikom kreiranja novog auto oglasa odabire, a ne unosi (kao što je oprema automobila) i slično.

5. Implementacija i izgled aplikacije

Web aplikacija auto oglasnika napravljena je u Visual Studio Code uređivaču kôda. Korišten je razvojni okvir Django koji je pisan u programskom jeziku Python te koristi MVT arhitekturu. Konfiguracija ove aplikacije nalazi se u datoteci „settings.py“. U toj datoteci postavljaju se informacije o bazi podataka, instaliranim aplikacijama, lokaciji gdje se spremaju mediji korišteni unutar aplikacije i slično. Putanje stranica aplikacije spremaju se u datoteci „urls.py“.

5.1. Modeli

Za izradu baze podataka korišten je 'code-first' način. Ovim načinom se prvotno stvaraju modeli tj. klase podataka unutar kôda, koji se migracijom preslikavaju u tablice baze podataka. Modeli se kreiraju u datoteci „models.py“. Na slici 2 može se vidjeti relacijski model podataka.



Slika 2: Relacijski model podataka

Model `CustomUser` je proširenje klase `AbstractUser`. Nadodani atributi su broj mobitela, adresa i slika profila. Postojeći atributi su korisničko ime, ime, prezime, lozinka,

e-mail, datum izrade korisničkog računa, datum zadnje prijave, je li aktivan te je li administrator. Jedinu vezu ostvaruje prema modelu `Oglas`, a tip veze je jedan na prema više.

Model `TehnickiPodaciAutomobila` je klasa koja sadrži tehničke podatke o automobilu kao na primjer vrsta motora, vrsta mjenjača i boja automobila. Jedinu vezu ostvaruje prema modelu `Oglas`, a tip veze je jedan na prema jedan.

Model `FotografijeAutomobila` je klasa koja ostvaruje vezu prema modelu `Oglas`, a tip veze je više na prema jedan.

Model `MarkaAutomobila` je klasa koja ostvaruje jedinu vezu prema modelu `ModelAutomobila`, a tip veze je jedan na prema više.

Model `ModelAutomobila` je klasa koja ostvaruje vezu prema modelu `Oglas`, a tip veze je više na prema jedan.

Model `Zupanija` je klasa koja ostvaruje jedinu vezu prema modelu `GradOpcina`, a tip veze je jedan na prema više.

Model `GradOpcina` je klasa koja ostvaruje vezu prema modelu `Oglas`, a tip veze je više na prema jedan.

Model `OpremaAutomobila` je klasa koja sadrži podatke o opremi automobila koja se dijeli na dodatnu opremu, sigurnosnu opremu, klimatizacija, zračni jastuci, itd. Jedinu vezu ostvaruje prema modelu `Oglas`, a tip veze je više na prema više.

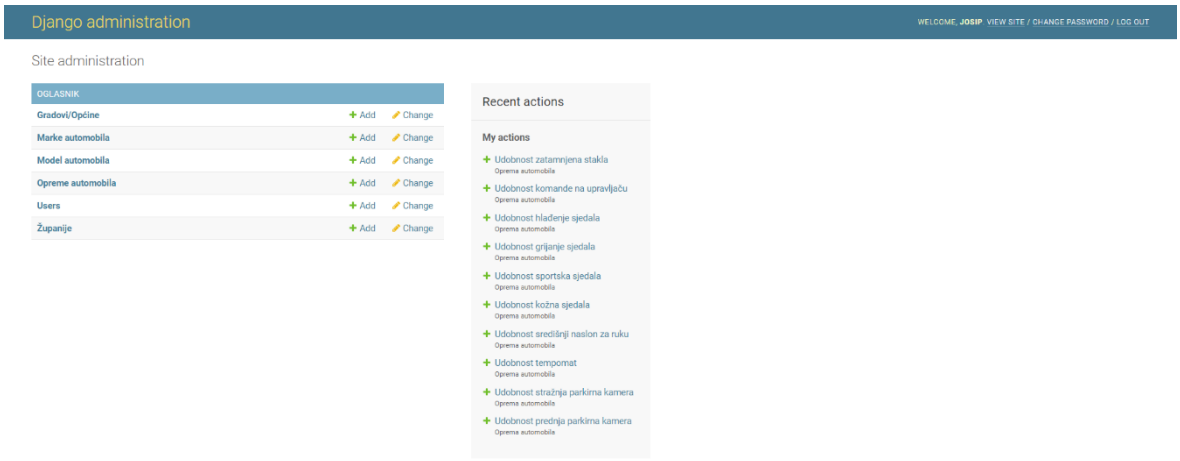
Model `Oglas` je klasa s kojom ostale klase ostvaruju najviše veza te ima još atributa kao što su naslov oglasa, opis oglasa, cijena automobila, kilometraža automobila, stanje automobila, datum objavljivanja oglasa, broj pregleda i je li oglas aktivan.

5.2. Prikazi i predlošci

U daljnjem tekstu opisati će se funkcionalnost aplikacije kroz komunikaciju prikaza i pripadajućeg predloška. Komunikacija informacijama između navedenih ostvarivala se preko gore opisanih modela ili preko Django formi koje služe za razmjenu informacija između HTML elemenata i modela. Spremanje podatka s klijentske strane u bazu podataka ide kroz idući proces: s HTML elementa preko Django forme prave se modeli koji se

preslikavaju u tablice unutar baze podataka. Suprotan proces je izvlačenje podataka iz baze na klijentsku stranu s ciljem uređivanja već postojećih. Proces u kojem Django forma ne sudjeluje je sami prikaz podataka iz baze na korisničku stranu.

Administratorsko sučelje, vidljivo na slici 3, omogućuje stvaranje, uređivanje i brisanje modela koji su registrirani na sučelje. Registracija modela se obavlja u „admin.py“ datoteci. Registrirani modeli za ovu aplikaciju su svi oni o kojima administrator treba voditi brigu, a to su primjerice korisnik, županija, grad/općina, marka automobila, model automobila, oprema automobila, itd.



Slika 3: Administratorsko sučelje

Neregistrirani korisnik ima mogućnost odlaska na stranicu za registraciju kojom upravlja funkcija `registerPage(request)` koja koristi Django formu `CreateCustomUserForm`. Postojeći registrirani korisnik ima mogućnost prijave, preko ispravnog korisničkog imena i lozinke, koju kontrolira funkcija `loginPage(request)`. Odjava trenutno prijavljenog korisnika kontrolirana je funkcijom `logoutUser(request)`. Pregled profila se obavlja preko stranice koju kontrolira funkcija `profil(request)`. Promjena lozinke se obavlja na zasebnoj stranici koju kontrolira funkcija `passwordChange(request)`.

Unutar ove aplikacije korišten je Djangov sustav za autentikaciju. Ovim načinom osigurano je da nigdje u bazi podataka nije zapisana lozinka u vidu običnog teksta (lozinka je šifrirana). Kako bi se ograničio pristup stranicama, korišten je dekorater `@loginrequired`

kojim se zahtjeva da je korisnik prijavljen. Osim ovog dekoratera, ograničavanje pristupa vršeno je provjerama da li je prijavljeni korisnik administrator ili je prijavljeni korisnik vlasnik auto oglasa.

Administrator ima mogućnost pregleda svih registriranih korisnika, a za to je zadužena funkcija `svi_korisnici(request)`. Na tom pregledu administrator klikom na korisničko ime odlazi na profil korisnika na administratorskom sučelju.

Na administratorskom sučelju administrator može uređivati i brisati korisničke račune. Važan dio uređivanja korisničkog računa je deaktivacija istog. Korisnik isto može uređivati svoj račun, ali brisanje mu nije dozvoljeno. Dozvoljena mu je deaktivacija računa iz razloga sigurnosti aplikacije, točnije zbog toga da se ne poremeti konzistentnost podataka unutar aplikacije. Administrator može i deaktivirati i izbrisati korisnički račun. Brisanjem računa iz baze podataka se brišu svi vezani podaci. Deaktivacijom računa svi auto oglasi vezani za taj račun postaju deaktivirani kako ne bi bili vidljivi ostalim korisnicima. Za pregled i uređivanje profila od strane korisnika zadužena je funkcija `profil(request)`, a za deaktivaciju korisničkog računa zadužena je funkcija `deaktiviraj_racun(request)`. Na ispisu 2 može se vidjeti kôd funkcije za deaktivaciju korisničkog računa.

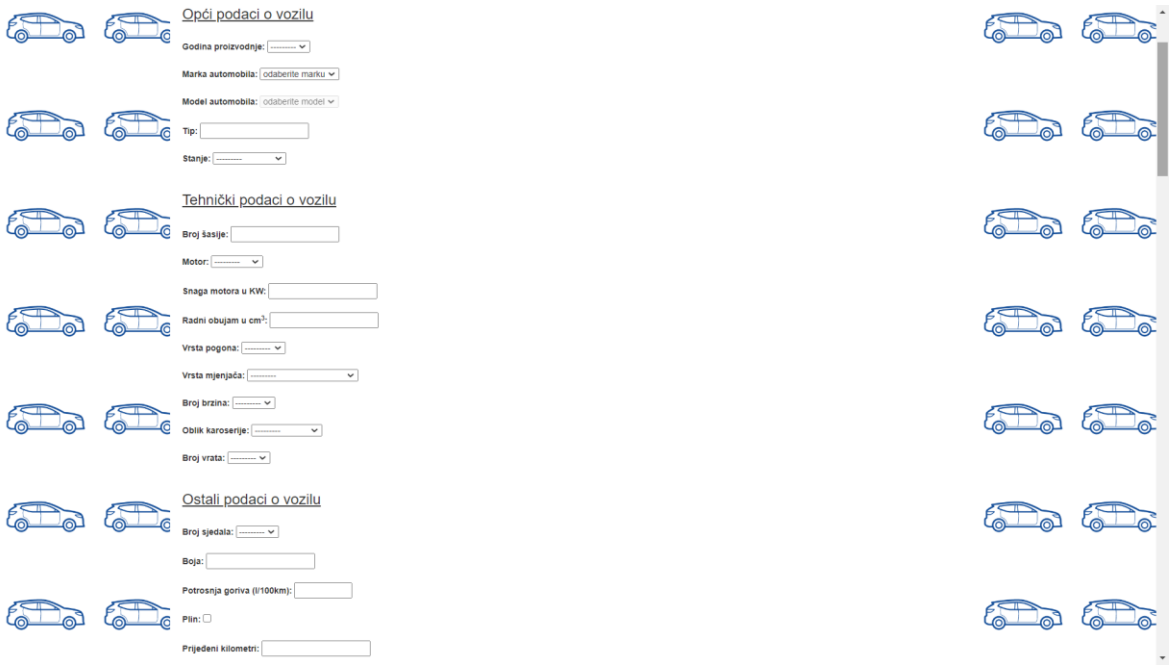
```
@login_required
def deaktiviraj_racun(request):
    user = CustomUser.objects.get(id = request.user.id)
    user.is_active = False
    user.save()
    Oglas.objects.filter(prodavac__id = request.u-
ser.id).update(jeli_aktivan = False)
    return logoutUser(request)
```

Ispis 2: Funkcija za deaktivaciju korisničkog računa

Kao što je već spomenuto, model korisnika je proširen s atributima od kojih je jedan atribut slika profila. Fotografija se sprema u direktorij `media/profilne`, a u bazu podataka se sprema putanja. Sve dok korisnik ne promijeni željenu sliku profila zadana slika profila je `media/profilne/default.png`. Brisanjem korisničkog računa, ukoliko je korisnik promijenio zadanu sliku profila, fotografija se briše iz baze podataka i iz direktorija

`media/profilne`. Zadana slika profila se ne briše iz direktorija kako bi bila dostupna za ostale korisnike.

Samo registrirani korisnik ima mogućnost objavljivanja auto oglasa preko stranice za stvaranje novog auto oglasa, a za to je zadužena funkcija `predajaOglasa(request)`. Dio stranice za stvaranje novog auto oglasa može se vidjeti na slici 4.



Slika 4: Dio stranice za stvaranje novog auto oglasa

Kao što je vidljivo na prethodnoj slici, korisnik ispunjava HTML formu koja predajom popunjava Django forme `OglasForm` i `TehnickiPodaciForm` za stvaranje novog auto oglasa. U funkciji nakon prijenosa podataka s klijentske strane te prije spremanja u bazu provjerava se ispravnost podatka kojima su te forme ispunjene. Kao što je već navedeno u dijelu ovog rada gdje se opisuju modeli, auto oglas se sastoji od više podataka koji se spremaju u više tablica. Zbog specifičnosti nekih podataka iz auto oglasa, isti su maknuti (engl. *exclude*) iz formi pa se popunjavaju unutar funkcije. Primjer takvih su: prodavač koji postaje trenutno prijavljeni korisnik, grad/općina (zbog veze sa županijom i nemogućnosti da korisnik odabere grad ili općinu prije nego odabere županiju), model automobila (zbog veze sa markom automobila i istom nemogućnosti kao prethodni primjer), datum objavljivanja koji postaje datum u trenutku spremanja auto oglasa, aktivnost koja se u

funkciji automatski stavlja na aktivna, oprema automobila te broj pregleda koji se povećava svakim dolaskom korisnika na stranicu za pregled oglasa.

Ograničenja pri objavljivanju novog auto oglasa su iduća: svi podaci osim fotografija i opreme su obavezni za unošenje, podaci „Vozilo u prometu od“ i „Godina prve registracije u RH“ ne mogu biti uneseni prije podatka „Godina proizvodnje“ te moraju biti isti ili veći od podatka „Godina proizvodnje“. Model automobila se ne može odabrati prije marke automobila te odabirom marke, modeli automobila će se filtrirati prema odgovarajućoj marki. Vrijednost ne smije ići u negativno za podatke „Snaga motora u KW“, „Radni obujam u cm3“, „Potrošnja goriva (l/100km)“, „Prijeđeni kilometri“ te „Cijena u HRK“, podatak „Broj šasije“ ne može biti duži od 17 znakova. Podatak o opciji plina će biti onemogućen za odabir ako vrsta motora nije „Benzin“, podatak „Grad/općina“ se ne može odabrati prije odabira „Županija“ te odabirom županije „Grad/općina“ će se filtrirati prema odgovarajućoj županiji. Dozvoljeno je postaviti maksimalno 7 fotografija te prilikom postavljanja korisnik ima mogućnost pretpregleda fotografija i uklanjanja istih.

Stranici za uređivanje postojećih auto oglasa mogu pristupiti administrator i registrirani korisnik, a kontrolira je funkcija `uredivanjeOglasa(request, id)` gdje je `id` primarni ključ auto oglasa. Pregled oglasa obavlja se preko stranice koju kontrolira funkcija `pregledOglasa(request, id)` gdje je `id` primarni ključ auto oglasa. Na pregledu auto oglasa ispisuju se podaci o pojedinom auto oglasu, a ukoliko postoje fotografije za taj auto oglas omogućen je galerijski prikaz implementiran preko biblioteke SimpleLightbox. Ukoliko auto oglas nema fotografija, dodana je zadana fotografija koja je spremljena u direktorij `media/automobili` kako bi se prikazala u ovom slučaju.

Svi korisnici aplikacije imaju pristup stranici za pregled oglasa, ali samo administrator i vlasnik auto oglasa imaju mogućnost uređivanja, brisanja, aktiviranja i deaktiviranja auto oglasa. Za brisanje auto oglasa koristi se funkcija `brisanje_oglasa(request, id)` koja kaskadno briše iz baze podataka sve podatke vezane za taj auto oglas te fotografije koje se spremaju u direktorij `media/automobili` (putanja fotografije se sprema u bazu podataka).

Funkcija `brisanje_oglasa(request, id)` preko ida pronalazi željeni auto oglas za brisanje, zatim pored toga što je već provjereno korisnik prijavljen, provjerava da li je trenutno prijavljeni korisnik vlasnik auto oglasa koji se želi obrisati te da li je administrator.

U slučaju da je prijavljeni korisnik vlasnik auto oglasa ili administrator, funkcija dozvoljava brisanje vezanih fotografija automobila i samog auto oglasa te korisnika preusmjerava na pregled svih aktivnih oglasa. U suprotnom preusmjerava ga na pregled oglasa, što bi značilo da nema pravo brisanja auto oglasa. Kôd funkcije je prikazan u ispisu 3.

```
@login_required
def brisanje_oglasa(request, id):
    oglas = Oglas.objects.get(id=id)
    if oglas.prodavac.id != request.user.id and not request.user.i
s_superuser:
        return redirect('pregled_oglasa', id = oglas.id)
    else:
        fotografije = FotografijeAutomobila.objects.filter(oglas_i
d = id)
        for foto in fotografije:
            foto.delete()
        Oglas.objects.get(id=id).delete()
        return redirect('moji_oglasi_aktivni', id=oglas.prodavac.i
d)
```

Ispis 3: Funkcija za brisanje auto oglasa

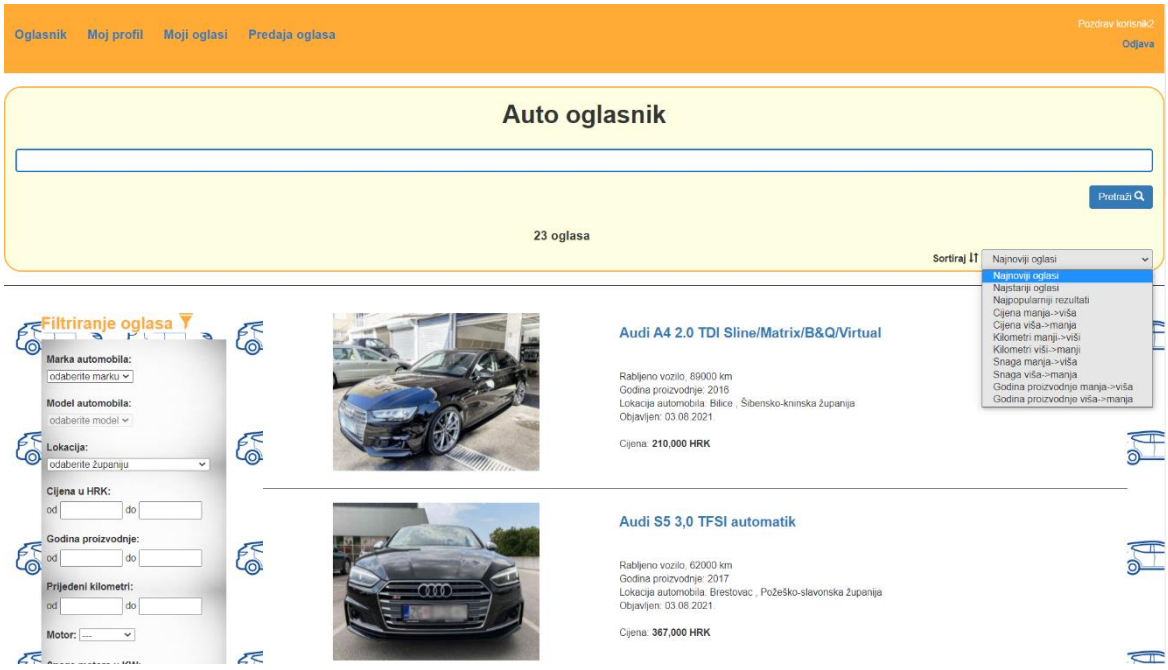
Brisanje fotografije je specifična radnja zbog toga što se osim zapisa u bazi podataka treba obrisati i iz direktorija `media/automobili`. Ovakvo brisanje postignuto je prekrivanjem (engl. *overriding*) metode `delete` unutar modela `FotografijeAutmobila`. Osim dozvoljavanja kaskadnog brisanja prilikom brisanja korisnika ili njegovog auto oglasa, bilo je potrebno nadograditi brisanje od strane administratora. Administratorsko sučelje ne koristi funkciju u kojoj je nadodano brisanje iz direktorija uz brisanje iz baze podataka. Kako bi se ova radnja izvršila ispravno, tj. kako bi se izbrisala fotografija iz direktorija, unutar „models.py“ datoteke nadodana je funkcija `foto_delete(sender, instance, **kwargs)` koja se aktivira zbog dekoratera `@receiver` koji prima signal prije brisanja. Kôd ove radnje prikazan je na ispisu 4.

```
@receiver(pre_delete, sender=FotografijeAutomobila)
def foto_delete(sender, instance, **kwargs):
    instance.fotografija.delete(False)
```

Ispis 4: Funkcija za brisanje fotografije iz direktorija

Aktiviranje kontrolira funkcija `aktiviraj_oglas(request, id)` koja status aktivnosti auto oglasa postavlja u istinit i postavlja trenutni datum za datum objavljivanja. Deaktiviranje kontrolira funkcija `deaktiviraj_oglas(request, id)` koja status aktivnosti auto oglasa postavlja u neistinit.

Prikaz početne stranice unutar funkcije za prikaz `home(request)` za zadatak ima iz baze podataka povući aktivne auto oglase te provjeriti jesu li im datumi objave stariji od mjesec dana. U slučaju da jesu, status oglasa se mijenja u deaktiviran te se takvi oglasi neće prikazati korisniku. Funkcija `home(request)` omogućava i pretragu, filtriranje i sortiranje. Na slici 5 može se vidjeti izgled početne stranice sa pripadajućim mogućnostima za pretragu, filtriranje i sortiranje.



Slika 5: Početna stranica

Pretraživanje radi tako da se povuče podatak iz input HTML elementa i filtriraju se auto oglasi tako da preostanu oni koji u naslovu sadrže traženi podatak. Filtriranje radi tako da korisnik iz ponuđenog skupa podataka odabire one podatke po kojima želi filtrirati auto oglase. Korisnik ima mogućnost filtriranja auto oglasa po marki automobila, po modelu automobila, po lokaciji automobila, po cijeni automobila, po godini proizvodnje automobila, po broju prijeđenih kilometara automobila, po vrsti motora automobila, po snazi motora automobila, po obliku karoserije automobila, po vrsti pogona automobila, po imanju opcije plina, po vrsti klime automobila, po vrsti mjenjača automobila, po broju brzina mjenjača automobila, po broju vrata automobila, po boji automobila, po potrošnji goriva automobila, po trajanju registracije automobila, po broju vlasnika automobila, po stanju automobila te po prikazu auto oglasa samo sa slikama. Navedene mogućnosti filtriranja vidljive su na slici 1.

Kod filtriranja postoje ograničenja poput nemogućnosti odabiranja modela automobila prije nego se odabere marka automobila. Preko jQuerya, na klijentskoj strani, polje koje pripada modelu automobila je onemogućeno za odabir sve dok se polje koje pripada marki automobila ne odabere. Također, polje za filtriranje auto oglasa prema plinu ima jQuery funkciju koja ga kontrolira te u slučaju da polje za vrstu motora za vrijednost nema opciju „Benzin“, opcija filtriranja prema plinu je onemogućena. Kôd funkcije prikazan je u ispisu 5.

```
<script>
	var $motor = $( '#motor' ),
	$plin = $('#plin');
	$motor.change(function() {
		if($motor.val() == "Benzin" || $motor.val() == "
		"){
			$plin.removeAttr('disabled');
		}
		else{
			$plin.attr('disabled', 'disabled');
			$plin.attr('checked', false);
		}
	}).change();
</script>
```

Ispis 5: Funkcija kontrole polja plin

Sortiranje auto oglasa se obavlja tako da trenutno povučene auto oglase funkcija sortira prema korisničkom odabiru, što je vidljivo na ispisu 6. Mogućnosti sortiranja su po datumu objavljivanja auto oglasa, po cijeni automobila, po broju prijeđenih kilometara automobila, po snazi motora automobila, po godini proizvodnje automobila te po najpopularnijim rezultatima koji ovise o broju pregleda auto oglasa. Sortiranja se mogu primijeniti ulazno i silazno.

Vrijednost navedenih mogućih parametara sortiranja, s klijentske strane definirana je brojevima od nula do deset. Na ispisu 6 može se vidjeti povezanost broja i parametra po kojem će se auto oglasi sortirati u slučaju odabira od strane korisnika. Funkcija `home(request)` također upravlja paginacijom te stranice. Kôd se može vidjeti u ispisu 7.

```
if 'sortiranje' in request.GET:
        context['sortiranje'] = int(request.GET['sortiranje'])
        if int(request.GET['sortiranje']) == 0:
            svi_oglasi = svi_oglasi.order_by('-
datum_objavljivanja')
        elif int(request.GET['sortiranje']) == 1:
            svi_oglasi = svi_oglasi.order_by('datum_objavljivan
ja')
        elif int(request.GET['sortiranje']) == 2:
            svi_oglasi = svi_oglasi.order_by('-broj_pregleda')
        elif int(request.GET['sortiranje']) == 3:
            svi_oglasi = svi_oglasi.order_by('cijena')
        elif int(request.GET['sortiranje']) == 4:
            svi_oglasi = svi_oglasi.order_by('-cijena')
        elif int(request.GET['sortiranje']) == 5:
            svi_oglasi = svi_oglasi.order_by('kilometraza')
        elif int(request.GET['sortiranje']) == 6:
            svi_oglasi = svi_oglasi.order_by('-kilometraza')
        elif int(request.GET['sortiranje']) == 7:
            svi_oglasi = svi_oglasi.order_by('tehpod__snaga')
        elif int(request.GET['sortiranje']) == 8:
            svi_oglasi = svi_oglasi.order_by('-tehpod__snaga')
        elif int(request.GET['sortiranje']) == 9:
            svi_oglasi = svi_oglasi.order_by('tehpod__godina_pr
oizvodnje')
        elif int(request.GET['sortiranje']) == 10:
            svi_oglasi = svi_oglasi.order_by('-
tehpod__godina_proizvodnje')
```

Ispis 6: Sortiranje auto oglasa

```
paginator = Paginator(svi_oglasi, 10)
page_number = request.GET.get('page')
page_obj = paginator.get_page(page_number)
```

Ispis 7: Postavljanje paginacije

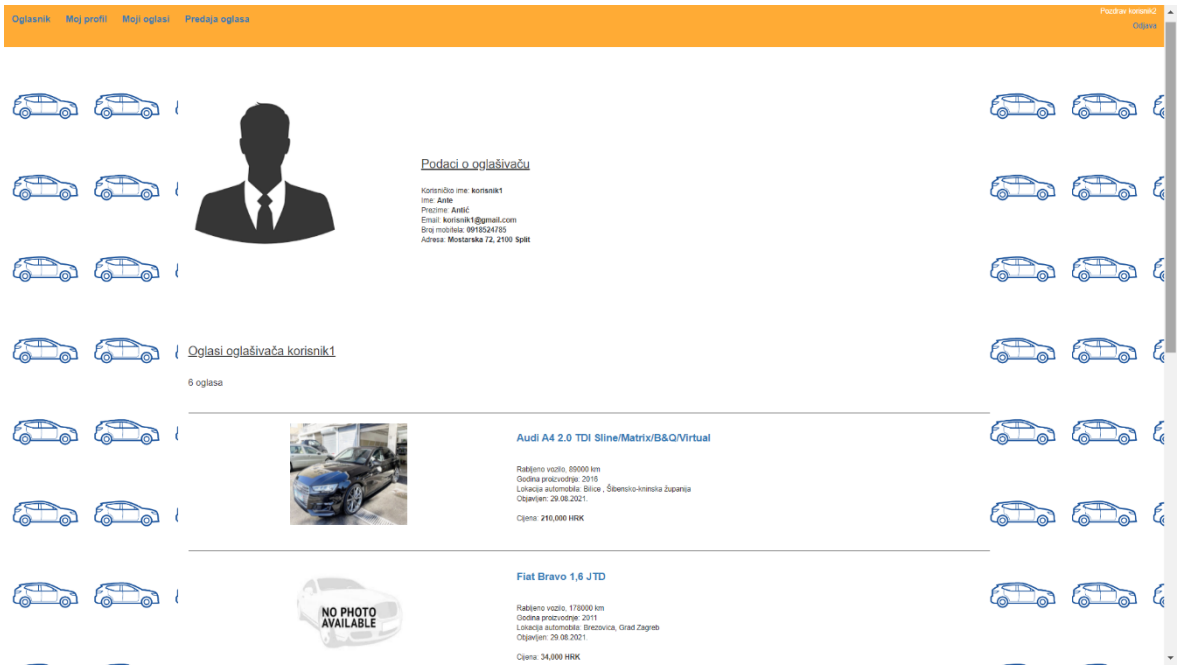
Administrator, na stranici za pregled svih korisnika, pored opcije da klikom na korisničko ime ode na profil korisnika (administratorsko sučelje) ima i opciju pregleda svih auto oglasa tog korisnika. Registrirani korisnik ima mogućnost pregleda svih svojih auto oglasa.

Ove dvije radnje kontrolira funkcija `moji_oglasi_aktivni(request,id)` gdje je `id` primarni ključ korisnika. Auto oglasi se prikazuju na različitim stranicama s obzirom na aktivnost (zbog preglednosti), pa tako na pregledu aktivnih auto oglasa postoji veza za pregled neaktivnih auto oglasa kojom upravlja funkcija `moji_oglasi_neaktivni(request,id)` i obrnuto. Kôd funkcije `moji_oglasi_neaktivni(request,id)` može se vidjeti na ispisu 8. Funkcija filtrira oglase po prodavaču i po aktivnosti te ih sortira po datumu objavljivanja silazno.

```
@login_required
def moji_oglasi_neaktivni(request, id):
    oglasi = Oglas.objects.filter(prodavac__id = id, jeli_akti-
van = False).order_by('-datum_objavljivanja')
    prodavac_id = id
    broj_oglasa = oglasi.count()
    context = {
        'oglasi' : oglasi,
        'broj_oglasa' : broj_oglasa,
        'prodavac_id' : prodavac_id
    }
    return render(request, 'moji oglasi neaktivni.html', context)
```

Ispis 8. Funkcija neaktivnih oglasa

Verzija pregleda profila, prikazana na slici 6, na kojoj su osobni podaci i aktivni auto oglasi omogućena je svim korisnicima aplikacije. Pristupa joj se preko veze na stranici za pregled auto oglasa te njom upravlja funkcija `korisnik(request,id)` gdje je `id` primarni ključ oglašivača.



Slika 6: Pregled profila i aktivnih oglasa omogućen svim korisnicima

6. Zaključak

Ovim završnim radom izrađena je web aplikacija za oglašavanje automobila korištenjem web razvojnog okvira Django koji je pisan u programskom jeziku Python. Napretkom i razvojem tehnologije u današnje vrijeme digitalizacija procesa prodaje je neophodna kako bi se prodaja što lakše i jednostavnije odvila između oglašivača i zainteresiranih kupaca.

Tematika ovog završnog rada je dovoljno široka da se ostavlja prostora za moguće nadogradnje i proširenja web aplikacije. Neka od mogućih proširenja koja bi pridonijela povezanosti korisnika aplikacije su razgovor korisnika (engl. *chat*) i komentari na auto oglasu. Mogu se implementirati kao zasebni modeli s tekstualnim atributom. Komentar bi imao vezu prema auto oglasu i prema prijavljenom korisniku, a razgovor samo prema prijavljenom korisniku. Lakši ponovni pronalazak željenog auto oglasa omogućila bi opcija za dodavanje auto oglasa u favorite. Favoriti se mogu implementirati kao zasebni model s vezom prema auto oglasima i prema korisnicima. Kako bi se upotpunio proces kupnje može se nadodati opcija za generiranje kupoprodajnog ugovora te omogućiti plaćanje. Generiranje kupoprodajnog ugovora može se implementirati kao zasebni model koji bi popunjavao podatke o kupcu, prodavaču i auto oglasu iz baze podataka. Plaćanje se može implementirati kroz integraciju nekog od sustava plaćanja koji dozvoljava Django.

Umjesto korištenja Django predložaka, za razvoj korisničkog sučelja mogli su se koristiti neki od klijentskih radnih okvira poput Angulara i VueJSa ili biblioteka poput ReactJSa. Njihovo korištenje pridonijelo bi izgledu korisničkog sučelja (moderniji izgled) i olakšalo samu izradu, povećavajući interakciju unutar aplikacije i općenito ugodnije korisničko iskustvo.

Literatura

1. Django, https://www.djangoproject.com/ (posjećeno 18.08.2021.)

2. Welcome to Python.org, https://www.python.org/ (posjećeno 18.08.2021.)

3. MySQL, https://www.mysql.com/ (posjećeno 18.08.2021.)

4. XAMPP Installers and Downloads for Apache Friends, https://www.apachefriends.org/index.html (posjećeno 18.08.2021.)

5. phpMyAdmin, https://www.phpmyadmin.net/ (posjećeno 18.08.2021.)

6. jQuery, https://jquery.com/ (posjećeno 18.08.2021.)

7. Bootstrap, https://getbootstrap.com/ (posjećeno 18.08.2021.)

8. HTML, https://html.com/ (posjećeno 18.08.2021.)

9. CSS: Cascading Style Sheets | MDN, https://developer.mozilla.org/en-US/docs/Web/CSS (posjećeno 18.08.2021.)