Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Jernej Jakofčič

Mobilna aplikacija za iskanje najcenejših bencinskih servisov in beleženje stroškov

diplomsko delo

visokošolski strokovni študijski program
prve stopnje
računalništvo in informatika

Mentor: viš. pred. dr. Aljaž Zrnec

Ljubljana, 2018

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

V Sloveniji imamo okoli 500 bencinskih servisov na različnih lokacijah. Cene naftnih derivatov se spreminjajo vsakih 14 dni, na nekaterih servisih pa se oblikujejo prosto. V okviru diplomske naloge zasnujte mobilno aplikacijo za Android, ki bo omogočala izbrati trenutno najcenejšega ponudnika naftnih derivatov na določeni lokaciji. Poleg tega naj aplikacija omogoča tudi beleženje plačil za naftne derivate in pošiljanje podatkov v oblak za poznejše analize. Pri izdelavi aplikacije uporabite Google-ove lokacijske storitve.

Zahvaljujem se mentorju viš. pred. dr. Aljažu Zrnecu za mentorstvo in vso strokovno pomoč. Iskreno se zahvaljujem tudi svoji družini za vso podporo ter prijateljem za pomoč, ki sem je bil deležen tekom študija.

Moji mami Bernardi.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
WCF	Windows Communication Foundation	Windowsova komunikacijska fundacija točnost
SQL	Structured Query Language	Strukturirani poizvedbeni jezik
GPS	Global Positioning Language	Globalni sistem pozicioniranja
JSON	JavaScript Object Notation	Standard za izmenjavo podatkov
IDE	Integrated Development Environment	Integrirano razvojno okolje
REST	Representational State Transfer	Pristop za komunikacijo s spletno storitvijo
CLI	Common Language Infrastrucutre	Specifikacija podjetja Microsoft za kodo in izvajalno okolje
SDK	Software Development Kit	Programski razvojni paket
ADT	Android Development Tools	Razvojna orodja Android
HTTP	Hypertext Transfer Protocol	Metoda za prenos informacij po spletu
HTML	Hyper Text Markup Language	Označevalni jezik za izdelavo spletnih strani
XML	Extensible Markup Language	Razširljivi označevalni jezik
URL	Uniform Resource Locator	Enolični krajevnik vira
DPI	Dots Per Pnch	Število pik na površini palca
OCR	Optical Character Recognition	Optično prepoznavanje znakov

Povzetek

Naslov: Mobilna aplikacija za iskanje najcenejših bencinskih servisov in beleženje stroškov

Avtor: Jernej Jakofčič

Vozniki za delovanje svojih motornih vozil potrebujejo gorivo, katerega cena se glede na ponudnika in lokacijo bencinskega servisa spreminja iz dneva v dan. Namen diplomske naloge je razviti mobilno aplikacijo za pametne telefone z operacijskimi sistemi Android, ki poišče najbližji in najcenejši bencinski servis glede na trenutno lokacijo, poleg tega pa omogoča tudi beleženje stroškov in s tem olajša iskanje cenovno in dostopno boljših bencinskih servisov. Potrebno je bilo pridobiti podatke o bencinskih servisih in se osredotočiti predvsem na tiste, ki so potrebni za iskanje najbližjega in najcenejšega bencinskega servisa, tj. ceno goriv in lokacijo. Na podlagi teh dveh lastnosti nam mobilna aplikacija omogoči navigacijo do bencinskega servisa, ki ustreza pogojem najbližjega in najcenejšega servisa.

Ključne besede: mobilna aplikacija, pametni telefon, Android, bencinski servis.

Abstract

Title: Mobile application for searching the cheapest filling station and recording costs

Author: Jernej Jakofčič

Motor vehicles need fuel for their operation. Prices of fuel are changing daily and they depend on the provider and location of the filling station. The purpose of this bachelor's degree is to develop a mobile application for smart phones with Android operating systems, which can find the nearest and cheapest filling station according to the current location of the user. Furthermore, the application enables us to record the costs and thus facilitate the search for more affordable and accessible filling stations. It was essential to obtain information on filling stations and primarily focus on those needed to find the nearest and cheapest one, those being the price of fuel and the location. Based on these two properties, the application enables us to navigate to the filling station that meets the conditions of the nearest and cheapest service.

Keywords: mobile application, smart phone, Android, filling station.

Poglavje 1

Uvod

Mobilna aplikacija je programska oprema, ki je prilagojena za delovanje na mobilni napravi, največkrat na telefonu ali tablici. Najbolj običajne aplikacije so že nameščene na mobilni napravi ob nakupu. Naprednejše mobilne aplikacije pa si je potrebno prenesti iz določenih spletnih trgovin, ki ponujajo mobilne aplikacije. Primer takšne trgovine je Google Play, kjer so na voljo najrazličnejše plačjive in neplačjive mobilne aplikacije, ki nam omogočajo pomoč pri delu, predvajanje multimedijskih vsebin, komunkacijsko podporo, igranje iger, prebiranje novic, navigacijo s pomočjo GPS in podobno.

1.1 Ozadje

Pametni mobilni telefon je naprava, brez katere si dandanes marsikdo ne predstavlja normalnega vsakdana. Omogoča nam uporabo različnih že vgrajenih mobilnih aplikacij, ki predstavljajo večino funkcionalnosti. Ena izmed bolj uporabljenih funkcionalnosti pametnih telefonov je GPS, ki s pomočjo ustreznih mobilnih aplikacij omogoča navigacijo do izbranih lokacij ali drugače daje navodila za pot do želene destinacije. Globalni sistem za pozicioniranje (GPS) deluje tako, da določa trenutno lokacijo in destinacijo s pomočjo satelitov. Aplikacije, ki uporabljajo GPS, prikazujejo našo lokacijo in destinacijo na zemljevidu, tako da je prikaz poti za doseg želenega cilja uporabniku čim

bolj razumljiv. Med uporabniki mobilnih aplikacij je zelo veliko takih, ki potrebujejo prikaz poti do najbližjega bencinskega servisa, da lahko napolnijo rezervoar svojega motornega vozila in se s tem izognejo težavam zaradi pomanjkanja goriva.

1.2 Namen in prispevek diplomskega dela

V Sloveniji imamo okoli 500 bencinskih servisov, pri katerih lahko kupimo različne naftne derivate. Večino trgovanja bencinskih servisov predstavlja trgovanje z naftnimi proizvodi, se pravi z nafto, bencinom, plinom in kurilnim oljem.

Cene naftnih derivatov se ves čas spreminjajo, zato je njihovim cenovnim spremembam težko slediti, če nismo vključeni v neprestano seznanjanje s spremembami le-teh. Cene neosvinčenega motornega 95-oktanskega in dizelskega goriva, ki se ne prodajajo na bencinskih servisih na prometnih površinah avtocest in hitrih cest v Sloveniji, se oblikujejo na osnovi določil uredbe o oblikovanju cen določenih naftnih derivatov. V skladu z uredbo se določajo najvišje prodajne cene teh naftnih derivatov, ki se spreminjajo vsakih 14 dni. Cene neosvinčenega 98- ali večoktanskega motornega bencina ter ekstra lahkega kurilnega olja se na vseh prodajnih mestih v Sloveniji določajo s strani trgovcev z naftnimi derivati. Enako je pri cenah neosvinčenega 95-oktanskega in dizelskega goriva na bencinskih servisih na prometnih površinah avtocest in hitrih cest v Sloveniji, kjer cene prosto določajo trgovci z naftnimi derivati [5].

Podatki o cenah so dostopni na spletnih straneh servisov [21, 20], kjer se nahajajo seznami vseh bencinskih servisov določenega trgovca z naftnimi derivati. Tudi lokacije bencinskih servisov najdemo na spletnih straneh ponudnikov.

Cilj diplomske naloge je izdelati mobilno aplikacijo za operacijski sistem Android, ki bo uporabnikom omogočala izbiro najbližjega in najcenejšega bencinskega servisa za nakup potrebnega goriva za motorno vozilo. Poleg

tega bo omogočeno tudi shranjevanje stroškov posameznega nakupa goriva za kasnejšo analizo. Podatki o cenah naftnih derivatov bodo posredovani v mobilno aplikacijo s pomočjo spletne storitve, ki se bo nahajala v oblaku in bo tako vključno s podatkovno bazo vedno dostopna mobilni aplikaciji. Pridobljeni podatki so ključnega pomena za delovanje aplikacije.

1.3 Struktura diplomskega dela

V drugem poglavju diplomskega dela se srečamo z opisom tehnologij in orodij, uporabljenih pri izdelavi aplikacije. Poleg predstavitve posameznih orodij je predstavljena tudi vloga posamezne tehnologije ali orodja pri izdelavi diplomske naloge.

V nadaljevanju je predstavljena izdelava in uporaba razlečenjevalnika podatkov, s katerim so bili pridobljeni podatki o bencinskih servisih, ki so shranjeni v podatkovni bazi in dostopni mobilni aplikaciji preko dostopanja do spletne storitve. Delovanje mobilne aplikacije temelji na uporabi teh podatkov.

Sledi predstavitev razvoja aplikacije, vse od pridobitve podatkov in izdelave same mobilne aplikacije do implementacije podatkovne baze in spletne storitve ter prenosa spletne storitve in podatkovne baze na strežnik.

V zaključku dela so podane sklepne ugotovitve in predlogi za izboljšanje aplikacije.

Poglavje 2

Pregled in analiza sorodnih aplikacij

Obstoječe mobilne aplikacije, ki omogočajo navigacijo do bencinskih servisov, večinoma ne vsebujejo cen naftnih derivatov, poleg tega pa nam ne omogočajo beleženja stroškov ob nakupu goriva. Funkcionalnost teh aplikacij povsem zadošča za pregled servisov v bližini in usmerjanje do njihove lokacije, a s pomanjkljivostjo, da ne predlaga najcenejšega in hkrati najbližjega bencinskega servisa. Sledi opis štirih mobilnih aplikacij, katerih analiza mi je služila kot pomoč pri razvoju mobilne aplikacije za iskanje najcenejšega in najbližjega bencinskega servisa.

2.1 Mobilna aplikacija DKV

DKV Euro Service je nemško podjetje, ki zagotavlja brezgotovinske storitve na evropskih cestah in je eno izmed vodilnih podjetij na področju logistike in transportnega sektorja. S kartico za gorivo ter aplikacijo DKV omogočajo, da uporabnik najde gorivo po ugodnih cenah in pri tem ostane mobilen [7].

DKV je primer mobilne aplikacije za pametne telefone z operacijskimi sistemi iOS, Windows in Android. Pridobimo jo lahko v trgovini z aplikacijami za operacijske sisteme Android Google Play, na App Storu, kjer lahko želene

aplikacije pridobijo uporabniki mobilnih naprav z operacijskim sistemom iOS, in pa na Windows Store, ki je trgovina za pametne telefone z operacijskim sistemom Windows. Uporabniki aplikacije lahko z njeno pomočjo poiščejo vse bencinske servise s partnerstvom DKV po vsej Evropi.

Aplikacija deluje tako, da poišče trenutno lokacijo s pomočjo GPS-a, vgrajenega v pametni telefon ali tablico, potem pa je potrebno podati svoje zahteve za iskanje bencinskega servisa. Na voljo imamo izbiro radija ali polmera, v katerem naj aplikacija poišče bencinske servise, nato sledi izbira goriva, ki ga potrebujemo, in pa izbira ponudnika naftnih derivatov. Na podlagi teh kriterijev aplikacija poišče servise v podanem radiju, do katerih imamo potem možnost usmerjanja oziroma navigacije.

Poleg pomanjkljivosti, ki so že bile omenjene, se pojavlja še ena bistvena pomanjkljivost, in sicer ta, da mobilna aplikacija omogoča samo iskanje servisov, ki so v partnerstvu z DKV, to pomeni, da prikaže samo določene servise in ne vseh, ki se morda lahko nahajajo v bližini trenutne lokacije uporabnika aplikacije.

2.2 Mobilna aplikacija OMV Stations

OMV je avstrijsko podjetje, ki se ukvarja z distribucijo, predelavo in proizvodnjo naftnih derivatov. Mobilna aplikacija OMV Stations je aplikacija ponudnika naftnih derivatov OMV. Omogoča nam vpogled v več kot 1450 bencinskih servisov po celotni Evropi. Spada med tiste mobilne aplikacije, ki uporabniku nudijo podatke o bencinskih servisih, navigacijo do bencinskih servisov s pomočjo GPS-a, prikaz vseh bencinskih servisov na določeni poti od začetne do končne lokacije in podatke o cenah naftnih derivatov pri ponudniku OMV, ki ima v Sloveniji nekaj več kot 100 bencinskih servisov. Poleg osnovnih funkcionalnosti nam aplikacija omogoča iskanje servisov z barom Viva Cafe in servise z bankomati, hkrati pa lahko želene servise dodamo na listo bolj priljubljenih bencinskih servisov in si s tem omogočimo hitrejši dostop do podatkov o določenem servisu [19].

Navedene storitve lahko uporabljamo v Sloveniji, Avstriji, Romuniji in Nemčiji. Tudi pri tej aplikaciji gre za omejeno iskanje bencinskih servisov, kajti gre samo za iskanje bencinskih servisov OMV, poleg tega pa nam aplikacija ne omogoči hitrega iskanja najcenejšega servisa.

2.3 Mobilna aplikacija Petrol

Petrol je slovensko trgovsko podjetje z naftnimi derivati, plinom in ostalimi energenti in hkrati njihov največji ponudnik, saj ima več kot 300 bencinskih servisov na območju celotne Slovenije. Petrol ima kar dve mobilni aplikaciji, ki omogočata lažje poslovanje in vsaka služi svojemu namenu. To sta aplikaciji Na poti in pa Moj Petrol.

2.4 Mobilna aplikacija Moj Petrol

Petrolova aplikacija Moj Petrol je na voljo samo uporabnikom plačilne kartice Petrol klub, s katero smo deležni določenih popustov pri nakupu. Aplikacija ponuja pregled porabe energentov ter njihovo naročanje, spremljanje kartičnega stanja in finančnega poslovanja, se pravi beleženja stroškov, ki smo si jih zadali ob nakupu. [17].

Aplikacija se poveže z računom, ki ga pridobimo ob naročilu kartice Petrol klub, in nam tako omogoča plačevanje s pomočjo aplikacije brez prisotnosti kartice, hkrati pa, kot je bilo omenjeno, tudi beleženje stroškov.

Pomanjkljivost aplikacije je zopet ta, da je njena uporabnost omejena samo na ponudnika naftnih derivatov, ki je njen lastnik, se pravi lahko z njo iščemo samo Petrolove bencinske servise, na katerih lahko plačujemo s to aplikacijo.

2.5 Mobilna aplikacija Na poti

Tudi pri uporabi aplikacije Na poti uporabnik potrebuje kartico Petrol klub. Tudi ta aplikacija ponuja podobne funkcionalnosti kot aplikacija Moj Petrol. Z mobilno aplikacijo Na poti lahko plačujemo račune na bencinskih servisih Petrol, poleg tega nam omogoča pregled računov za gorivo in spremljanje stroškov, hiter pregled zlatih točk ugodnosti in pa iskanje najbližjega Petrolovega bencinskega servisa [18].

Pri iskanju bencinskih servisov se aplikacija omejuje samo na bencinske servise ponudnika Petrol, prav tako je s plačevanjem z aplikacijo, ki je mogoče samo pri Petrolu.

Poglavje 3

Uporabljene tehnologije in programska oprema

Mobilna aplikacija je bila implementirana za mobilne naprave, ki uporabljajo operacijski sistem Android. Podatki o bencinskih servisih, ki se nahajajo v podatkovni bazi, so bili pridobljeni s pomočjo razčlenjevalnika. Za dostop do podatkovne baze, ki se nahaja v oblaku, uporablja mobilna aplikacija spletno storitev, ki se prav tako nahaja v oblaku. V nadaljevanju sledi predstavitev programskih orodij ter tehnologij, ki so bile uporabljene pri izdelavi mobilne aplikacije za iskanje najbližjega in najcenejšega bencinskega servisa.

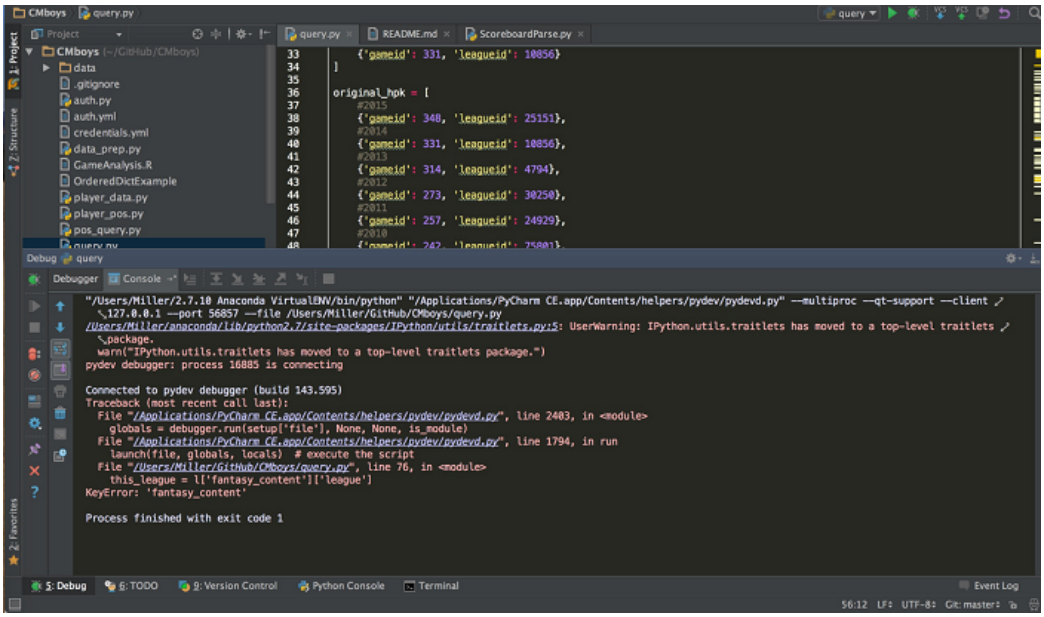
3.1 PyCharm Community

PyCharm je programska oprema ali bolje rečeno integrirano okolje za razvoj programske opreme (IDE) v programskem jeziku Python. PyCharm razvija podjetje JetBrains. Na voljo sta dve različici orodja PyCharm, in sicer Community različica, ki je brezplačna in dostopna vsem uporabnikom, ter Professional različica, ki je plačljiva.

Tehnična razlika med različicama je v tem, da prva podpira le osnovne funkcionalnosti razvojnega orodja za programiranje v Pythonu, ne omogoča pa podpore spletnemu razvijanju ter raznim ogrodjem, kot je na primer

Django. To ne pomeni, da aplikacije Django ni mogoče izdelati s pomočjo neplačljive različice, ampak to, da delo ni tako produktivno kot pri plačljivi verziji PyCharma [23].

S pomočjo programske opreme PyCharm in programskega jezika Python je bil izdelan razčlenjevalnik podatkov, ki je pridobil podatke o bencinskih servisih z njihovih spletnih strani. Slika 3.1 prikazuje razvojno okolje PyCharm, ki je razvojno orodje za programski jezik Python.



Slika 3.1: Razvojno okolje PyCharm [13].

3.1.1 IDE

IDE ali integrirano razvojno okolje je programsko orodje, ki programerjem pomaga pri razvoju. Običajno vsebuje urejevalnik izvorne kode, prevajalnik oziroma tolmač, orodje za avtomatizacijo izgradnje programa in običajno tudi razhroščevalnik. Ponavadi so v IDE vgrajeni tudi sistem za nadzor različic programov in orodja za tvorbo grafičnih uporabniških vmesnikov.

Prednost integriranih razvojnih orodij je, da so si v osnovi med seboj

podobna. Veliko podobnosti najdemo v grafičnem vmesniku in funkcionalnostih orodij, ki so vključena v IDE, zato se izkušenemu programerju ni težko privaditi na novo okolje, saj že ve, kaj lahko od IDE pričakuje [10].

3.1.2 Python

Najboljše je, da se učenja programskega jezika in programiranja lotimo na čim bolj zanimiv ter razumljiv način. Tukaj v veliko primerih nastopi programski jezik Python, ki je eden izmed bolj enostavnih programskih jezikov. Njegove najboljše karakteristike so, da je enostaven za učenje ter razumljiv in povsem primeren za razvoj preprostih in srednje zahtevnih programov. Primeren je tako za začetnike kot tudi za zahtevnejše uporabnike.

Ena izmed njegovih prednosti je ta, da deluje v vseh operacijskih sistemih brez spreminjanja kode, kar predstavlja bistveno prednost pred drugimi programskimi jeziki, npr. programskim jezikom C++. Python je interpreter oziroma tolmač, kar uporabnikom omogoča razvoj preprostih rešitev z malo vrsticami kode za probleme, ki se lahko pojavijo pri razvoju določenih aplikacij.

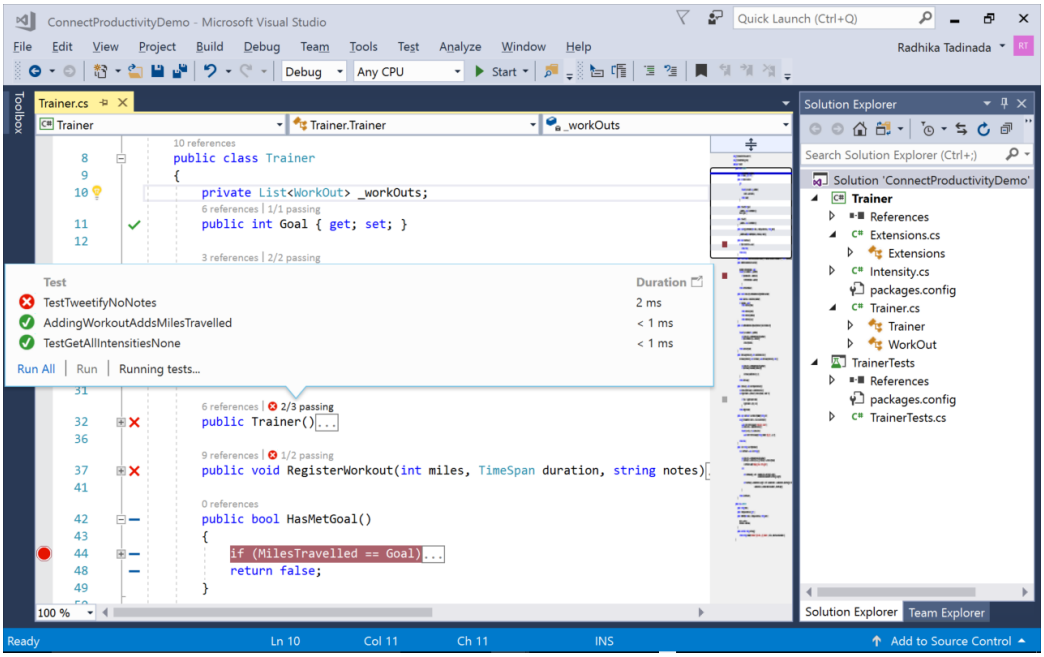
3.2 Microsoft Visual Studio

Visual Studio je tako kot PyCharm integrirano razvojno okolje, ki ga je razvilo in ga še vedno nadgrajuje podjetje Microsoft. Namenjen je predvsem razvoju spletnih strani, spletnih aplikacij, spletnih storitev, programov za operacijski sistem Windows in ostalih aplikacij, ki temeljijo na ogrodju .NET. Z Visual Studiom se med drugim razvijajo aplikacije, ki so prenosljive med različnimi platformami.

Microsoft ponuja več različic Visual Studia. Community različica je brezplačna za osebno in akademsko uporabo ter za uporabo v manjših skupinah. Professional različica je plačljiva različica Community in je namenjena podjetjem. Različica Enterprise, ki je prav tako plačljiva, ponuja vse funkcionalnosti Visual Studia. Pri vseh treh različicah imamo na voljo uporabo več

programskih jezikov. Najbolj standardni izmed njih je Visual Basic, sledi mu C++, ki je splošnonamenski programski jezik, in pa C# [24, 25].

V integriranem razvojnem okolju Microsoft Visual Studio je bila s programskim jezikom C# izdelana spletna storitev, ki skrbi za operacije s podatki med mobilno aplikacijo in podatkovno bazo. Tudi podatkovna baza Microsoft SQL je bila izdelana v Microsoft Visual Studiu. Spletna storitev in podatkovna baza sta bili iz Visual Studia naloženi v oblak Azure. Slika 3.2 prikazuje Microsoftovo razvojno orodje Visual Studio.



Slika 3.2: Razvojno okolje Visual Studio [16].

3.2.1 C#

C# je programski jezik, ki ga je razvila skupina Microsoftovih strokovnjakov za programske jezike leta 2001 v okviru razvoja ogrodja .NET. Gre za razmeroma nov programski jezik, ki je bil razvit kot osnovni programski jezik, ki podpira vse funkcionalnosti ogrodja .NET in je usmerjen v objektno

programiranje. Podoben je programskima jezikoma C in C++.

Dele programov, napisanih v programskem jeziku C#, odlikuje enostavnost povezovanja z ostalimi deli programov, ki so lahko izdelani v kateremkoli programskem jeziku, ki spada med programske jezike, skladne s CLI, poleg tega pa imajo programski jeziki tudi dobro prenosljivost med različnimi platformami. Programski jeziki CLI so programski jeziki, ki se uporabljajo za izdelavo različnih knjižnic in programov. Ti programi morajo biti skladni s specifikacijo CLI, kar pomeni, da se ti jeziki najprej prevedejo v standardiziran skupni jezik. Posledično lahko sestavimo aplikacijo iz več različnih delov, ki so lahko napisani v kateremkoli izmed teh programskih jezikov [6].

Primeri programskih jezikov, ki ustrezajo specifikaciji CLI:

- C# (najpogosteje uporabljen),
- C++,
- Visual Basic .NET,
- Windows PowerShell.

3.3 Postman

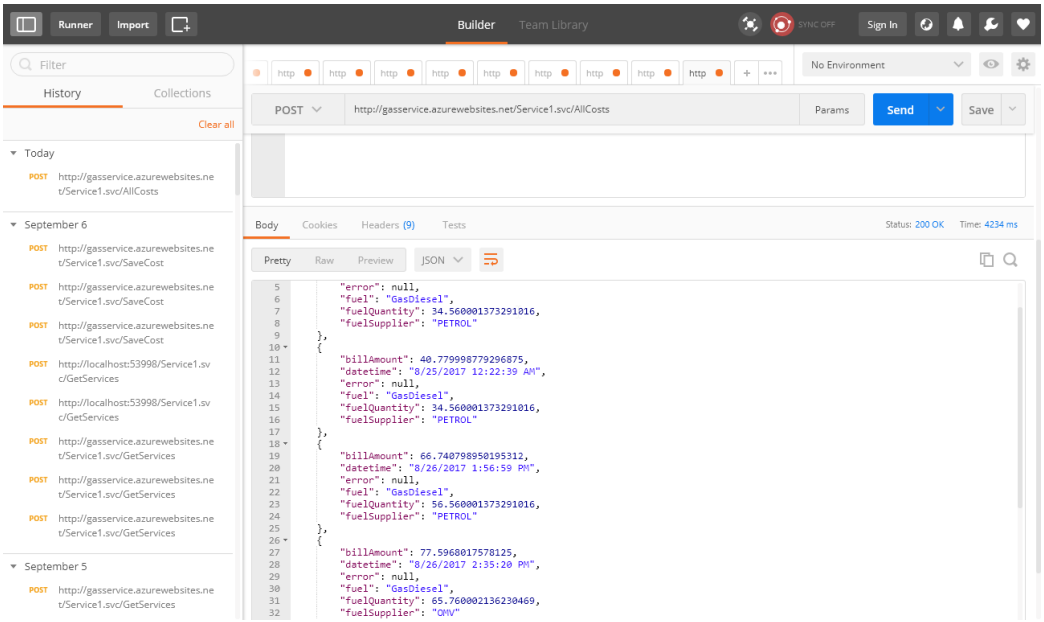
API-je (ang. Application Programming Interface) in spletne storitve je potrebno vedno stestirati, da se prepričamo, če na določeno zahtevo vrnejo želene podatke. Postman je eno izmed orodij, ki nam to omogoča in je eno izmed bolj popolnih razvojnih okolij za razvoj API-jev. Gre za aplikacijo, ki je bila razvita in prvič predstavljena oktobra 2012. Po predstavitvi je bila deležna hitrega vzpona števila uporabnikov, do danes pa je postala ena izmed bolj popularnih aplikacij v spletni trgovini Chrome Store, ki se lahko uporablja na operacijskih sistemih Mac, Windows in Linux [22].

Postman omogoča pošiljanje zahtev za določene podatke, sprejemanje odzivov in njihovo shranjevanje, dodajanje testov ter kreiranje in spremljanje poteka dela. Osnovna funkcionalnost aplikacije je, da sestavimo določeno

zahtevo v izbranem formatu (JSON, XML, itd.), izberemo vrsto zahteve (GET, POST, itd.) in vpišemo URL-naslov. Na ta naslov bo poslana zahteva in v primeru, da je sestavljena pravilno ter vsebuje ustrezne podatke, nam storitev, do katere smo dostopali s pomočjo zahteve, poslane s Postmana, vrne želene podatke.

Postman je različica aplikacije, ki jo lahko uporabljamo zastonj. Obstaja pa tudi različica Postman PRO, ki je plačljiva in je namenjena predvsem večjim skupinam razvijalcev, ker omogoča povezavo z različnimi orodji zunaj aplikacije. Slika 3.3 prikazuje Postman - razvojno okolje za API-je.

S pomočjo Postmana je bila testirana spletna storitev. Glede na zahteve, ki so bile poslane s Postmana, so se v njem pojavili odgovori na poslane zahteve. Na osnovi preverjanja podatkov v odgovorih spletne storitve in glede na stanje v podatkovni bazi je bila preverjena pravilnost delovanja spletne storitve.



Slika 3.3: Postman - razvojno okolje za API-je.

3.4 Microsoft Azure

Azure je izjemno zmogljiva Microsoftova storitev računalništva v oblaku. Gre za odprto in fleksibilno platformo javnega oblaka, ki tako posameznikom kot tudi podjetjem in organizacijam omogoča različne storitve, kot so dostop do spletnih aplikacij, spletnih storitev in podatkovnih baz, ki lahko gostujejo na platformi Azure. Podpira hiter razvoj in upravljanje aplikacij. Pri razvoju je mogoče uporabiti več različnih programskih jezikov kot tudi več različnih orodij. Ker gre za računalništvo v oblaku, je povsem preprosta tudi povezava aplikacije v oblaku z ostalimi odjemalci na lokaciji stranke.

Visoka dosegljivost in zanesljivost omogočata uporabnikom, da na platformi Azure izvajajo tudi aplikacije, ki za svoje izvajanje potrebujejo zahtevnejše in ugodnejše pogoje, poleg tega pa ponuja uporabnikom platforme tudi neomejene strežniške in shranjevalne kapacitete [3].

Pomembne lastnosti platforme Azure:

- avtomatizirano posodabljanje operacijskih sistemov,
- zagotavljenje sistema za deljenje bremen,
- zagotavljanje mehanizma za varovanje pred strojnimi napakami,
- nadgradnja aplikacij brez prekinitve delovanja le-teh,
- odprtost (programske knjižnice za odjemalce so prosto dostopne in gostujejo na platformi GitHub).

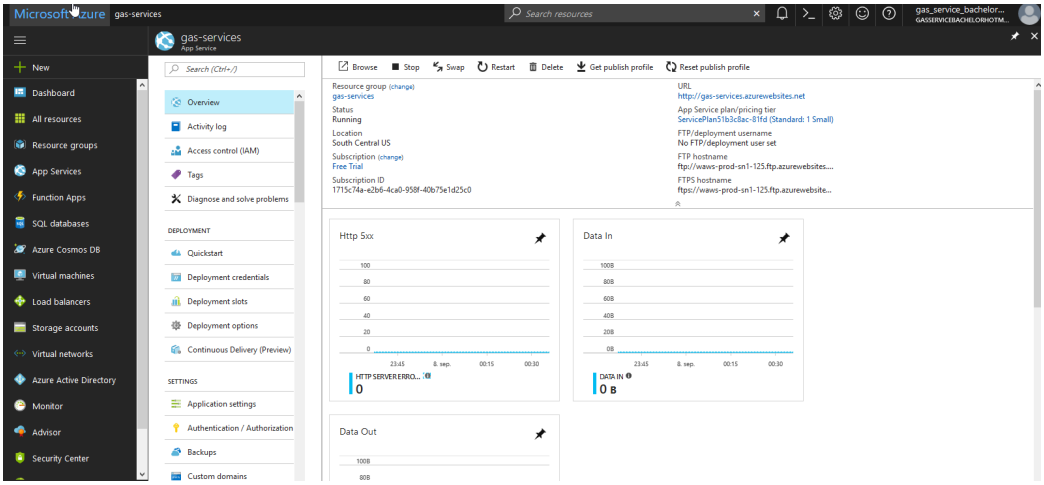
Poleg zgoraj naštetih lastnosti Azure zagotavlja tudi urejeno arhiviranje podatkov, restavriranje le-teh in lažjo integracijo z obstoječim okolji za virtualizacijo [4].

Gradniki platforme Azure:

- Microsoft SQL,

- Microsoft .NET,
- Live,
- Microsoft SharePoint,
- Microsoft Dynamics CRM.

Spletna storitev in podatkovna baza sta bili naloženi v oblak Azure, s čimer sta bili preko internetnega omrežja dostopni mobilni aplikaciji. Slika 3.4 prikazuje Azure - storitveno in infrastrukturno platformo v oblaku.



Slika 3.4: Azure - storitvena in infrastrukturna platforma v oblaku.

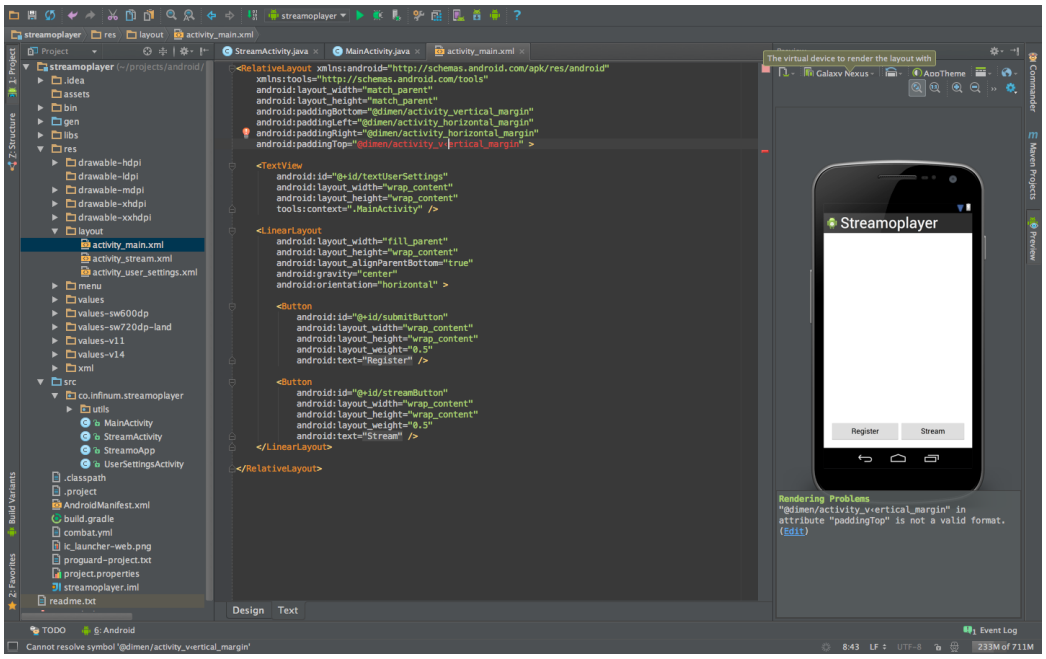
3.5 Android Studio

Android Studio je povsem brezplačno razvojno okolje, ki obsega programski razvojni paket (SDK) in omogoča programiranje tako preprostih kot tudi zahtevnejših aplikacij za operacijski sistem Android. Omogoča razvoj v programskih jezikih Java in C++.

Tako kot prej omenjena PyCharm in Visual Studio je tudi Android Studio integrirano razvojno okolje, le da je namenjen razvoju aplikacij za operacijski sistem Android. Razvil ga je Google, temelji pa na programski opremi

Intellij IDEA razvijalca JetBrains [11] in je neke vrste zamenjava za Eclipse ADT, ki je bilo prvotno orodje za razvoj mobilnih aplikacij Android. Znotraj razvojnega okolja vsebuje med drugim tudi urejevalnik postavitev grafičnih gradnikov aplikacije, ki jo razvijamo tako, da imamo poleg programiranja funkcionalnosti aplikacije Android tudi orodje za razvoj dejanskega izgleda aplikacije [2].

S pomočjo razvojnega okolja Android Studio in programskega jezika Java je bila razvita mobilna aplikacija za mobilne naprave Android, ki omogoča navigacijo do najbližjega bencinskega servisa in beleženje stroškov. Slika 3.5 prikazuje razvojno okolje Android Studio.



Slika 3.5: Razvojno okolje Android Studio [1].

3.5.1 Java

Java je objektno usmerjen programski jezik, katerega avtorji so zaposleni podjetja Sun Microsystems. Java vsebuje bogato standardno knjižnico pro-

gramskih struktur in funkcij, ki nam omogočajo delo z datotekami, dostopanje do podatkovnih baz, uporabo mrežne povezave, hkratno izvajanje več programov, grafične aplikacije, mobilne aplikacije in podobno.

Poznamo več vrst Jave:

- J2SE - Java za osebne računalnike,
- J2ME - Java za mobilne naprave, pametne televizorje,
- J2EE - poslovna različica Jave.

Javina dodatna vrednost je to, da se lahko v Javi napisana koda, ko je enkrat prevedena, zažene na vseh platformah, ki podpirajo Javo [12].

3.6 JSON

JSON je preprost format za izmenjavo podatkov, ki se uporablja za različne namene, med drugim za komunikacijo med spletnimi storitvami in njihovimi odjemalci, z uporabo protokola REST. Je enostaven za branje in pisanje. Temelji na JavaScriptu. JSON je tekstoven format, ki je v celoti neodvisen od programskega jezika. Njegova struktura temelji na dvojicah ime-vrednost. Ime je ključ, preko katerega se prepozna in pridobi vrednost. Vrednosti so lahko različnih podatkovnih tipov (String, Integer, Polje, Objekt ...) [14].

Format JSON lahko uporabljamo tudi pri aplikaciji Postman za sestavo zahteve, ko hočemo poslati določeno zahtevo za podatke.

Format JSON je bil uporabljen za izmenjavo podatkov med aplikacijo Postman in spletno storitvijo pri testiranju le-te. Prav tako pa si je s pomočjo JSON-a mobilna aplikacija izmenjavala podatke s spletno storitvijo. Slika 3.6 prikazuje urejen seznam vrednosti v formatu JSON.

Slika 3.6: JSON - urejen seznam vrednosti [15].

Poglavje 4

Razčlenjevalnik podatkov

V tem poglavju je skupaj z uporabljenimi knjižnicami opisana izdelava razčlenjevalnika podatkov.

Razčlenjevalnik podatkov (ang. Parser) je rešitev za pridobivanje in obdelavo podatkov. Z njegovo pomočjo lahko iz množice podatkov izluščimo samo tiste informacije, ki jih potrebujemo.

V okviru pridobivanja podatkov za izdelavo diplomske naloge je bil izdelan razčlenjevalnik s pomočjo programskega jezika Python v razvojnem okolju PyCharm. Gre za razčlenjevalnik, ki deluje s pomočjo različnih Pythonovih knjižnic, ki nam olajšajo delo pri iskanju želenih podatkov. Potrebno je bilo pridobiti naslove in koordinate (dolžino in širino) bencinskih servisov ter cene goriv na posameznih bencinskih servisih. Pridobljeni podatki so se s pomočjo knjižnice za povezavo s podatkovno bazo Microsoft SQL vnesli v podatkovno bazo. Vneseni podatki so bili tako pripravljeni za uporabo s strani spletne storitve.

4.1 Uporaba knjižnic Requests in Beautiful Soup

Za pridobivanje HTML-vsebine spletnih strani bencinskih servisov se je uporabljala Pythonova knjižnica Requests, ki omogoča pridobitev vsebine iz

določenega URL-naslova, ki ga podamo metodi *get()* knjižnice Requests. Za pridobitev HTML-vsebine spletne strani je bila uporabljena metoda *beautifulSoup()*, ki je del knjižnice Beautiful Soup. Slika 4.1 prikazuje pridobivanje vsebine spletne strani Petrola, kjer se nahaja seznam bencinskih servisov.

```
page = requests.get("http://www.petrol.si/bencinski-servisi/seznam")
soup = BeautifulSoup(page.content, 'html.parser')
```

Slika 4.1: Pridobivanje vsebine spletne strani Petrola, kjer se nahaja seznam bencinskih servisov.

Slika 4.2 prikazuje iskanje določenega elementa znotraj HTML-vsebine.

```
for line in soup.find_all("td"):
```

Slika 4.2: Iskanje določenega elementa znotraj HTML-vsebine.

Ko so bili iz HTML-ja posameznih spletnih strani bencinskih servisov pridobljeni podatki, med drugim tudi URL-naslovi za dostop do strani, kjer se nahajajo podatki o cenah goriv na posameznem servisu, je bilo potrebno sestaviti ustrezne URL-naslove, preko katerih se je dostopalo do slik s cenami goriv. Večinoma so bili enaki, spreminjale so se samo identifikacijske številke servisov, ki so služile za dostop do podatkov o cenah.

4.2 Uporaba knjižnic Urllib, Pytesseract in PIL

Optično prepoznavanje znakov je postopek pridobivanja podatkov iz slikovnih gradiv ali z drugimi besedami branje teksta s slike. Za pridobivanje cen goriv se je bilo potrebno poslužiti tega postopka, kajti na Petrolovi spletni strani so cene naftnih derivatov na posameznih bencinskih servisih prikazane

na sliki. Na tem mestu je bila uporabljena knjižnica Urllib in njena metoda *urlRetrieve()*, ki je pridobila sliko iz podanega URL-naslova ter jo shranila na disk. Nato je sledila uporaba metode *image()* knjižnice PIL, s katero se sliko odpre in se ji lahko spreminja lastnosti. Vsem slikam je bilo potrebno pred začetkom branja teksta povečati ločljivost in gostoto slikovnih točk (ang. Dots Per Inch), saj v nasprotnem primeru prepoznava teksta ni pravilna. Po transformaciji je bilo sliko potrebno shraniti nazaj na disk.

Zadnji postopek je predstavljal dejansko pridobivanje teksta s slike. Sliko s spremenjenimi lastnostmi je bilo potrebno zopet odpreti, tudi tokrat s pomočjo metode Image. Ko je bila slika zopet odprta, se je iz nje prebral tekst s pomočjo metode *image_to_string()* knjižnice Pytesseract. Pytesseract je Pythonova knjižnica, ki omogoča pridobivanje besedila iz slik. Uporabimo jo lahko na slikah različnih formatov, kot so: jpg, png, git, bmp in ostali. Slika 4.3 prikazuje pridobivanje teksta s slike.

```
for line in hrefs:
    page = requests.get("http://" + line)
    soup = BeautifulSoup(page.content, 'html.parser')

    for img in soup.find_all("img"):

        src = img.get("src")

        if "/sites/www.petrol.si/files/bencinskiServisi/" in src:

            url = "http://" + src.replace("/sites/", "")

            urllib.request.urlretrieve(url, "new.jpg")

            image = Image.open("new.jpg")
            im = image.resize((4320, 2952), Image.ANTIALIAS)
            im.save("repaired.jpg", quality=100, dpi=(600, 600))

            repaired_im = Image.open("repaired.jpg")
            text = pytesseract.image_to_string(repaired_im, lang='eng')
            text = text.replace('\n\n', '\n')
```

Slika 4.3: Postopek pridobivanja teksta iz slike.

4.3 Knjižnica Pymssql

Podatke, pridobljene s slik, je bilo potrebno shraniti v podatkovno bazo Microsoft SQL. Za to je bilo potrebno vzpostaviti povezave med razčlenjevalnikom in podatkovno bazo. Knjižnica Pymssql je preprost vmesnik za Python, ki omogoča povezavo s podatkovno bazo Microsoft SQL in izvajanje poizvedb nad podatkovno bazo. Ko so bili potrebni podatki o bencinskih servisih ustrezno izluščeni, so se s pomočjo te knjižnice shranili v podatkovno bazo Microsoft SQL.

Povezava s podatkovno bazo se odpre s pomočjo metode *connect()*, ki ji je potrebno podati podatke o podatkovni bazi, in sicer uporabniško ime in geslo uporabnika, ime podatkovne baze in ime serverja, ki je lahko kar povezovalni niz, ki vsebuje vse potrebne podatke za dostop do serverja.

Nato sledi kreiranje kazalca s pomočjo metode *cursor()*. Kazalec omogoča kodi Python izvajati poizvedbe SQL nad podatkovno bazo. Kazalec je pripet na povezavo do podatkovne baze, dokler ni izvedena metoda *close()*, ki zapre povezavo do baze. Izvajanje poizvedb SQL nam omogočata metodi *execute()* in *executemany()*, ki kot parameter sprejmejo poizvedbo SQL v obliki niza. To poizvedbo lahko oblikujemo tako, da ji pripnemo dodatne parametre, ki so potrebni za uspešno izvedbo poizvedbe. Z metodo *commit()* potrdimo spremembe, ki smo jih izvedli nad podatkovno bazo. Slika 4.4 prikazuje povezavo s podatkovno bazo in vstavljanje podatkov.

```
conn = pymssql.connect(server, username, password, dbname)

cursor = conn.cursor()

cursor.executemany(
    "INSERT INTO GasStations VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)",
    [(name, address, longitude, latitude, gas95, gas100, gasDiesel, gas, oil)]
)

cursor.commit()
conn.close()
```

Slika 4.4: Povezava s podatkovno bazo in vstavljanje podatkov.

Poglavje 5

Spletna storitev

V tem poglavju sledi opis implementacije spletne storitve WCF (ang. Windows Communication Foundation), podatkovne baze in nato še migracije obeh v oblak Azure.

Za komunikacijo mobilne aplikacije s podatkovno bazo je bilo potrebno implementirati spletno storitev, ki bo to omogočala. Gre za spletno storitev REST, kar pomeni, da gre za komunikacijo s pomočjo protokola HTTP in uporabo zahtev GET, POST, PUT, DELETE itd. Za izmenjavo podatkov med spletno storitvijo in odjemalcem se uporablja JSON, ki je sintaksa za shranjevanje in izmenjavo podatkov. To je tekstovni format, ki je popolnoma neodvisen od programskega jezika.

Razvoj spletne storitve je potekal v Microsoft Visual Studiu s programskim jezikom C#. Za shranjevanje in dostopanje do podatkov je bila prav tako v Visual Studiu ustvarjena podatkovna baza Microsoft SQL.

Vse metode, ki so implementirane znotraj spletne storitve, sprejmejo zahtevo HTTP POST, skupaj z ustreznimi podatki JSON, ki jih metode potrebujejo za izvrševanje poizvedb nad podatkovno bazo Microsoft SQL.

Za dostop do storitve potrebuje uporabnik uporabniško ime in geslo, ki ga pridobi ob uspešni registraciji v mobilno aplikacijo (odjemalec) Android. Ob registraciji mora podati svoje osnovne podatke, ki omogočajo prijavo v mobilno aplikacijo in uporabljanje njenih funkcionalnosti.

5.1 Podatkovna baza MSSQL

Podatkovna baza vsebuje pet tabel. Prva tabela je namenjena hranjenju podatkov o uporabnikih mobilne aplikacije. Geslo posameznega uporabnika je shranjeno v zakriptirani obliki s pomočjo zgoščevalne funkcije SHA-256, ki preslika vhodni niz v nekakšen prstni odtis vhodnega niza. Ostali podatki, ime, priimek, elektronski naslov in uporabniško ime, so shranjeni v navadni tekstovni obliki.

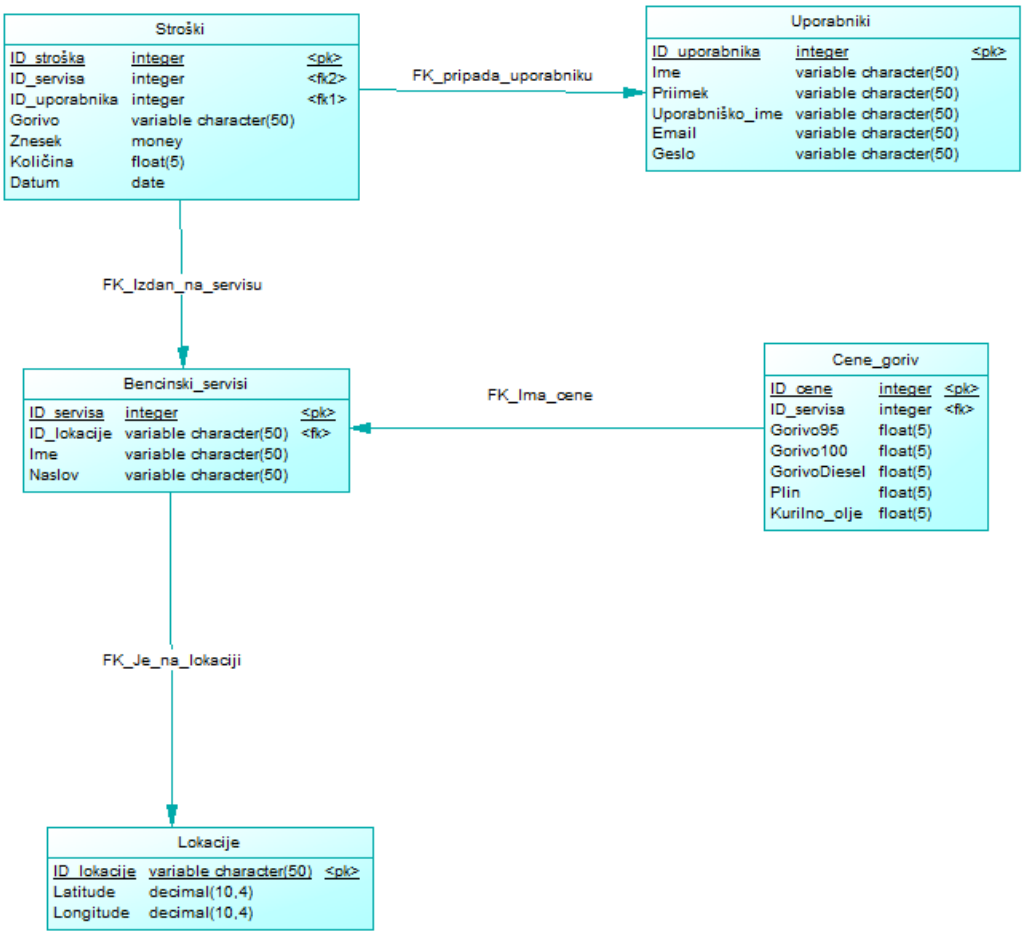
Sledi tabela bencinskih servisov, ki hrani osnovne podatke o posameznih bencinskih servisih, pridobljene s pomočjo razčlenjevalnika. Tabela vsebuje naslov in ime servisa.

Tretja tabela vsebuje podatke o lokaciji posameznega bencinskega servisa, zemljepisno širino (ang. Latitude) in zemljepisno dolžino (ang. Longitude).

Posamezne cene naftnih derivatov se med bencinskimi servisi razlikujejo. Četrta tabela je namenjena hranjenju cen vseh naftnih derivatov, ki jih lahko kupimo na večini bencinskih servisov po Sloveniji.

Zadnja tabela pa hrani podatke o stroških, ki jih je imel uporabnik z nakupom goriva na izbranem bencinskem servisu. Podatki o posameznem shranjenem strošku se navezujejo na točno določen bencinski servis s pomočjo identifikacijske številke. Preko uporabniškega imena pa se navezujejo tudi na uporabnika, ki je strošek shranil preko odjemalca.

Povezava do podatkovne baze je v spletni storitvi omogočena in implementirana s pomočjo razreda SqlConnection, ki mu podamo ConnestionString, ki predstavlja niz z vsemi podatki za povezavo z bazo. Izvajanje vseh poizvedb v zvezi s podatkovno bazo v okviru spletne storitve je implementirano s pomočjo razreda SqlCommand, ki mu je potrebno podati poizvedbo Microsoft SQL s parametri, ki jih lahko vključimo v poizvedbo preko tako imenovanega "bindinga". Binding je uporaba spremenljivk znotraj poizvedb SQL, ki se jim dinamično določajo vrednosti, ki so nato uporabljene v posameznih poizvedbah SQL. Poleg tega pa je razredu SqlCommand treba podati tudi niz s podatki za povezavo do podatkovne baze. Slika 5.1 prikazuje logični podatkovni model podatkovne baze.



Slika 5.1: Logični podatkovni model.

5.2 Pridobitev najbližjega in najcenejšega bencinskega servisa

Bistvena funkcionalnost spletne storitve je poleg omogočanja prijave, registracije in shranjevanja stroškov vračanje najbližjega in najcenejšega bencinskega servisa, če seveda spletna storitev prejme pravilno zahtevo za pridobitev servisa. Storitev mora dobiti zahtevo HTTP POST s podatki JSON, ki morajo vsebovati trenutno lokacijo (dolžino in širino) uporabnika mobilne aplikacije, tip goriva, ki ga potrebuje (bencin, dizel, plin ...) in želeno odda-

ljenost servisa, ki je lahko večja ali manjša od 50 kilometrov. Oddaljenost 50 kilometrov je bila izbrana zato, ker večini avtomobilov ob opozorilu za pomanjkanje goriva preostane približno 50 kilometrov vožnje s preostalim gorivom.

Poizvedba za pridobitev bencinskih servisov, ki se nahajajo v določenem polmeru oziroma radiju, se poslužuje formule Haversine, ki se uporablja za določanje razdalje med dvema podanima zemljepisnima dolžinama in širinama na Zemljini obli [9].

Bencinskim servisom, ki jih pridobimo s to poizvedbo, se nato primerja cena. V primeru, da je najcenejši samo en bencinski servis, je ta dokončno izbran, sicer je potrebna nadaljnja primerjava. Nato se primerja cene servisov in se izbere najcenejšega in hkrati najbližjega ter se ga vrne. Če se v določenem polmeru ne nahaja noben bencinski servis, ki ustreza iskalnim kriterijem, se vrne ustrezno sporočilo in je potrebno znova izbrati kriterije za iskanje ter poslati novo zahtevo HTTP POST.

Podatki o bencinskem servisu, ki jih vrne poizvedba, in nato odgovor na zahtevo HTTP POST so:

- naslov bencinskega servisa,
- oddaljenost od bencinskega servisa,
- cene posameznega goriva (95-oktanski bencin, dizel ...),
- koordinate bencinskega servisa (širina in dolžina),
- ime ponudnika storitev (Petrol, OMV ...).

Slika 5.2 prikazuje poizvedbo za pridobitev bencinskega servisa znotraj določenega polmera.

```
SqlCommand cmd = new SqlCommand("SELECT " +
    "Id, Name, Address, Gas95, Gas100, GasDiesel, Gas, Oil, Longitude, Latitude, " +
    "CAST(ACOS(SIN(RADIANS(Latitude)) * SIN(RADIANS(@uLat)) + COS(RADIANS(Latitude)) " +
    "* COS(RADIANS(@uLat)) * COS(RADIANS(Longitude) - RADIANS(@uLon))) * 6380 AS DECIMAL(10,2)) AS Distance " +
    "FROM GasStations WHERE " +
    "ACOS(SIN(RADIANS(Latitude)) * SIN(RADIANS(@uLat)) + COS(RADIANS(Latitude)) * " +
    "COS(RADIANS(@uLat)) * COS(RADIANS(Longitude) - RADIANS(@uLon))) * 6371 < @distance " +
    "ORDER BY Distance", conn);

cmd.Parameters.AddWithValue("@uLat", float.Parse(latitude, System.Globalization.CultureInfo.InvariantCulture));
cmd.Parameters.AddWithValue("@uLon", float.Parse(longitude, System.Globalization.CultureInfo.InvariantCulture));
cmd.Parameters.AddWithValue("@distance", distance);

using (SqlDataReader reader = cmd.ExecuteReader())
{
    while (reader.Read())
    {
        GasService gasService = new GasService();
        gasService.id = reader.GetInt32(reader.GetOrdinal("Id"));
        gasService.provider = reader.GetString(reader.GetOrdinal("Name")).ToString();
        gasService.address = reader.GetString(reader.GetOrdinal("Address")).ToString();
        gasService.gas95 = reader.GetString(reader.GetOrdinal("Gas95")).ToString();
        gasService.gas100 = reader.GetString(reader.GetOrdinal("Gas100")).ToString();
        gasService.gasDiesel = reader.GetString(reader.GetOrdinal("GasDiesel")).ToString();
        gasService.gas = reader.GetString(reader.GetOrdinal("Gas")).ToString();
        gasService.oil = reader.GetString(reader.GetOrdinal("Oil")).ToString();
        gasService.latitude = reader.GetString(reader.GetOrdinal("Latitude")).ToString();
        gasService.longitude = reader.GetString(reader.GetOrdinal("Longitude")).ToString();
        gasService.distance = reader.GetDecimal(reader.GetOrdinal("Distance"));
        list.Add(gasService);
    }
}
```

Slika 5.2: Poizvedba za pridobitev bencinskega servisa znotraj določenega polmera.

5.3 Testiranje spletne storitve

Testiranje je preizkus delovanja programa ali storitve z množico testnih vhodnih podatkov, ki vsebujejo robne primere. Na ta način se potrdi ali ovrže pravilnost delovanja. Pri testiranju moramo biti osredotočeni na iskanje napak v programu. Gre za verifikacijo in validacijo programa. Namen testiranja je pokazati, kaj program dela in kako se obnaša ob določenih pogojih. V okviru testiranja se lahko odkrijejo napake, ki se nanašajo na kodo, nepravilno prenašanje zahtev, nepravilno posredovanje odgovorov na zahtevo in podobno. Če pride do določene napake, se to napako identificira in odpravi. Včasih te napake pripeljejo do pojavitve odpovedi delovanja sistema, zato je testiranje korak, ki ga ne smemo preskočiti.

Testiranje spletne storitve je potekalo tako, da je bilo najprej potrebno zagnati spletno storitev (lokalno) s pomočjo orodja Microsoft Visual Studio.

Nato je sledilo pošiljanje zahtev HTTP POST na URL-naslov, na katerem je bila dosegljiva storitev. Za prijavo je bil to URL localhost/Service1.svc/Login. Če se je kje pojavil kakšen problem oziroma neustrezno delovanje spletne storitve, potem je bilo te napake potrebno odpraviti v programski kodi storitve.

Testiranje s pošiljanjem zahtev na URL-naslov storitve je bilo izvedeno s pomočjo aplikacije Postman. Potrebno je bilo nastaviti ustrezno metodo (POST) in vpisati ustrezen URL-naslov, ki se od metode do metode razlikuje. Na koncu je bila sestavljena vsebino zahtevka v obliki JSON in nastavljena avtorizacija, da je storitev preverila, ali zahteva prihaja od pravega uporabnika.

5.4 Gostovanje v oblaku Azure

Ko je bilo delovanje spletne storitve in njenih poizvedb nad podatkovno bazo Microsoft SQL uspešno stestirano, je sledila namestitev storitve in podatkovne baze na Microsoftov oblak Azure.

Uporaba oblaka Azure je z Microsoft Visual Studiom povsem preprosta. Visual Studio namreč omogoča prijavo v oblak z uporabniškim imenom Azure, nakar lahko izbrane projekte enostavno naložimo v oblak. Kreirati je potrebno spletno storitev in podatkovno bazo na Azuru, da si pripravimo prostor, kamor se bosta shranili storitev in podatkovna baza. Preden naložimo spletno storitev, je potrebno posodobiti povezovalni niz, ki vsebuje podatke za dostop do podatkovne baze, sicer storitev ne bo dostopala do baze na portalu Azure. V Visual Studiu se izbere želeni projekt in pritisne "Publish".

Ko sta spletna storitev in podatkovna baza nameščeni v oblak, lahko od kjerkoli dostopamo do nje s pomočjo URL-naslova, na katerem se spletna storitev nahaja. Če hočemo, da do storitve dostopamo tudi iz računalnika, moramo na platformi Azure dodati pravice za dostop. Potrebno je vpisati IP-naslov računalnika, s katerega bomo dostopali do spletne storitve.

Storitve, kot je računalništvo v oblaku, seveda niso brezplačne. Tudi

Microsoftova platforma Azure svojih storitev ne ponuja brezplačno, je pa mogoče dobiti enomesečni brezplačen dostop za spoznavanje portala in s tem izkoristiti priložnost za testiranje delovanja računalništva v oblaku.

Poglavje 6

Mobilna aplikacija Napolni

To poglavje opisuje načrtovanje, implementacijo in delovanje mobilne aplikacije. Mobilna aplikacija za iskanje najbližjega in najcenejšega bencinskega servisa in beleženja stroškov je bila razvita za mobilne naprave z operacijskim sistemom Android. Operacijski sistem Android je bil izbran zaradi njegove razširjenosti med uporabniki pametnih telefonov in tablic.

Aplikacija je bila razvita v programskem jeziku Java, grafični elementi aplikacije pa v jeziku XML. Uporabilo se je privzeto orodje za razvoj Androidovih aplikacij, Android Studio, ki omogoča programiranje funkcionalnosti aplikacije in izdelavo njene grafične podobe.

Za zaganjanje in testiranje mobilne aplikacije se je uporabljal mobilni telefon OnePlus, model X, ki ima nameščen operacijski sistem Android različice 6.0.1 Marshmallow, ki je šesta različica operacijskega sistema Android.

Primarna funkcionalnost aplikacije je iskanje najbližjega in najcenejšega bencinskega servisa glede na uporabnikovo trenutno lokacijo. Pri tem si aplikacija pomaga z razredom Location Manager, ki omogoča dostop do sistemskih lokacijskih storitev mobilne naprave. Te storitve omogočajo aplikacijam pridobivanje in posodabljanje trenutne lokacije mobilne naprave, kar se izvede vsako določeno časovno periodo. Ko aplikacija pridobi trenutno lokacijo mobilne naprave, pošlje zahtevo HTTP POST na URL-naslov spletne storitve, ki se nahaja na portalu Azure, za iskanje bencinskega servisa. Odgovor

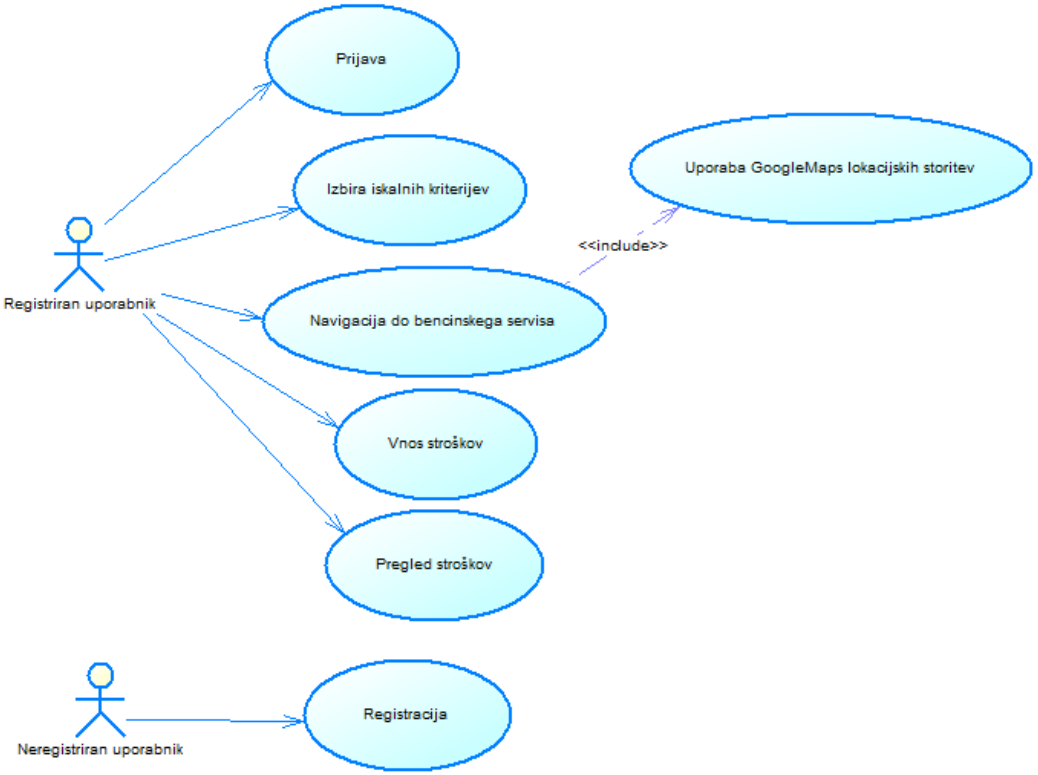
na zahtevo HTTP z vsebino JSON, ki prispe iz spletne storitve, vsebuje podatke o najbližjem in najcenejšem servisu. V primeru, da se v določenem iskanem polmeru ne nahaja noben bencinski servis, je uporabnik s pomočjo aplikacije ustrezno opozorjen in mora ponoviti iskanje z drugačnimi kriteriji. Ob uspešnem odgovoru spletne storitve s podatki o servisu v vsebini odgovora JSON je uporabnik preusmerjen na Google Maps, ki je prosto dostopen strežnik z geografskimi podatki in zemljevidi, ki ga ponuja podjetje Google in ima dosegljive zemljevide držav sveta, večjih mest, satelitske slike celega sveta ter pogled iz talne perspektive, kjer ima na voljo navigacijo od svoje trenutne lokacije do bencinskega servisa. Po končani navigaciji in pa morebitnem nakupu goriva ima uporabnik možnost vnosa stroška in pregleda dosedanjih zabeleženih stroškov. Za upravljanje z vsemi omenjenimi funkcionalnostmi se mora uporabnik pred tem registrirati in se prijaviti v mobilno aplikacijo.

6.1 Načrtovanje aplikacije

Mobilna aplikacija je bila načrtovana v sožitju s funkcionalnostmi, ki jih ponuja spletna storitev, da ji bodo te funkcionalnosti čim bolje služile ob delovanju. Najprej je sledila skica vseh zaslonov, ki jih bo potrebovala aplikacija in hkrati identifikacija funkcionalnosti, ki jih bo imel posamezen zaslon. Sledila je izbira orodja, ki se bo uporabilo za razvoj aplikacije.

Potrebno je bilo izbrati način povezave mobilne aplikacije s spletno storitvijo za omogočanje izmenjave podatkov med njima. Izbran je bil razred HttpUrlConnection, ki omogoča sestavljanje zahtev HTTP in povezovanje ter pošiljanje zahtev HTTP na določen URL-naslov.

Na koncu načrtovanja je prišla na vrsto izbira barv, ki so se uporabile pri izdelavi grafičnih vmesnikov. Slika 6.1 prikazuje diagram primerov uporabe aplikacije, ki prikazuje funkcionalnosti aplikacije.



Slika 6.1: Diagram primerov uporabe.

6.2 Razvoj aplikacije

Razvoj aplikacije je potekal v skladu z načrtom. Uporabljeno je bilo integrirano razvojno okolje Android Studio, ki omogoča programiranje funkcionalnosti aplikacije, ki jo razvijamo, in izdelavo grafične podobe aplikacije.

Najprej je bilo potrebno izdelati celotno grafično podobo aplikacije, in sicer zaslonske maske za:

- prijavo,
- registracijo,
- zamenjavo gesla,

- iskanje najbližjega in najcenejšega bencinskega servisa,
- navigacijo,
- beleženje stroškov,
- prikaz stroškov.

Pri zaslonih je šlo za postavljanje gradnikov, ki jih ponuja Android Studio, v okno, ki predstavlja izgled izbranega zaslona. Poleg tega je bilo tukaj potrebno urejanje elementov, uporabljenih na zaslonu v označevalnem jeziku XML.

Zasloni so nato potrebovali implementacijo njihovih funkcionalnosti. Pred začetkom implementacije funkcionalnosti je bilo potrebno v konfiguracijsko datoteko AndroidManifest.xml dodati potrebne pravice za aplikacijo, in sicer za dostopanje do internetnega omrežja in za pridobivanje trenutne lokacije mobilne naprave, sicer ju aplikacija ne more uporabljati.

Pri vseh zaslonih, razen pri zaslonu za navigacijo, je šlo za implementacijo pošiljanja zahteve HTTP POST z vsebino JSON, ki je vsebovala podatke za metodo spletne storitve, na katero je bila zahteva posredovana. Zahteve HTTP POST se formira s pomočjo razreda HttpUrlConnection, ki vsebuje metode za nastavitev avtorizacije, da spletna storitev ve, od kod prihaja določena zahteva, nastavitev vrste zahteve (zahteva HTTP POST), format vsebine, ki se posreduje v zahtevi (JSON), in še vsebino odgovora na zahtevo, ki ga je pripravljena sprejeti mobilna aplikacija (JSON). Slika 6.2 prikazuje pravice za dostopanje do internetega omrežja in pridobivanja lokacije mobilne naprave.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

Slika 6.2: Pravice za dostopanje do internetnega omrežja in pridobivanja lokacije mobilne naprave.

Razred JSONObject in njegova metoda Put služita za kreiranje vsebine JSON, ki se posreduje spletni storitvi. Za pošiljanje vsebine JSON zahteve HTTP POST pa se uporablja razred OutputStreamWriter, ki zakodira vsebino JSON v byjte in jo posreduje spletni storitvi s pomočjo metode Flush.

Pri sprejemanju vsebine odgovora JSON na zahtevo se uporabljata razreda BufferedInputStream in BufferedReader, s pomočjo katerih se vsebina vsebine odgovora JSON zapiše na seznam, s katerega se pridobi posamezne vrstice vsebine z metodo ReadLine.

Pridobivanju trenutne lokacije mobilne naprave služi razred Location Manager, ki omogoča dostop do sistemskih lokacijskih storitev mobilne naprave.

Trenutno lokacijo naprave se pridobi s pomočjo štirih metod, in sicer:

- *requestLoationUpdates()* - posodobitev lokacije naprave,
- *getLastKnownLocation()* - pridobitev zadnje lokacije naprave,
- *getLatitude()* - pridobitev zemljepisne širine,
- *getLongitude()* - pridobitev zemljepisne dolžine.

Za navigacijo je bilo potrebno uporabiti lokacijske storitve Google Maps. Kreirati je bilo potrebno povezavo do lokacijske storitve Google Maps s pomočjo pravilnega URL-naslova, ki vključuje začetno ter končno lokacijo in nam zažene možnost navigacije do bencinskega servisa, ki se nahaja na končni lokaciji. Slika 6.3 prikazuje sestavo URL-naslova za dostop do Google Maps.

```
String uri = "http://maps.google.com/maps?" +
        "saddr=" + latitude + "," + longitude +
        "&daddr=" + destLatitude + "," + destLongitude;
```

Slika 6.3: Sestava URL-naslova za dostop do Google Maps.

6.2.1 Lokacijske storitve Google Maps

Zemljevidi Google (ang. Google Maps) so storitev z geografskimi podatki in zemljevidi, ki jo ponuja podjetje Google. Prikazuje zemljevid sveta, večjih mest, vasi in cestnih povezav. Poleg tega pa zemljevidi Google uporabniku omogočajo tudi navigacijo do lokacije, ki si jo ta izbere. Na podlagi izbranega prevoznega sredstva (avto, javni prevozi, kolo, peš) se spreminja tudi čas potovanja do izbrane destinacije. Posebna funkcionalnost zemljevidov Google je Street view, ki omogoča sprehod po izbrani lokaciji v tridimenzionalnem pogledu po mestu, vasi ali kje drugje [8].

6.3 Delovanje aplikacije

6.3.1 Registracija, prijava in zamenjava gesla

Ko se aplikacija zažene, se nahajamo na zaslonu za prijavo, ki nam omogoča prijavo v aplikacijo, registracijo novega uporabnika ali zamenjavo trenutnega gesla. Ob prvi uporabi aplikacije prijava ni mogoča, saj je uporabnik še neregistriran, torej brez podatkov za prijavo (uporabniško ime in geslo). Osnovna funkcionalnost mobilnih aplikacij je registracija uporabnika, torej vnos podatkov o uporabniku, da lahko identificiramo osebe, ki uporabljajo aplikacijo.

Pri registraciji novega uporabnika mobilne aplikacije za iskanje najbližjega in najcenejšega servisa moramo podati svoje ime, priimek, elektronski naslov, uporabniško ime in geslo, kot je razvidno na sliki 5.3., ki predstavlja stran za registracijo.

Ob uspešni registraciji pridobimo uporabniško ime in geslo, s katerima se prijavimo ob naslednji uporabi aplikacije. Mogoča je tudi zamenjava gesla na strani za zamenjavo gesla. Funkcionalnost pride prav, če uporabnik pozabi geslo ali hoče zaradi kakšnega drugega razloga zamenjati svoje podatke za prijavo. Če uporabnik posodobi geslo, je preusmerjen na stran za prijavo v aplikacijo, kjer lahko vpiše svoje ime in geslo. Ob uspešni prijavi in prav tako ob uspešni registraciji je uporabnik preusmerjen na stran za iskanje ben-

cinskega servisa. Slika 6.4 prikazuje zaslonsko masko za registracijo novega uporabnika.

Slika 6.4: Zaslonska maska za registracijo novega uporabnika.

6.3.2 Iskanje najbližjega in najcenejšega bencinskega servisa

Na zaslonu za iskanje bencinskega servisa se nahaja kriterij za iskanje servisa. Kriterij vsebuje izbiro goriva, ki ga uporabnik potrebuje in katerega ceno se bo preverilo, poleg tega pa še radij oddaljenosti servisa. Mogoče je iskanje servisa znotraj 50-kilometrskega dosega ali iskanje dlje kot 50 kilometrov od trenutne lokacije. Slika 6.5 prikazuje zaslonsko masko za izbiro iskalnega kriterija.

Izbira iskalnega dosega je postavljena nad ali pod 50 kilometrov, kar je povprečna razdalja, ki jo še lahko prevozi motorno vozilo ob opozorilu na

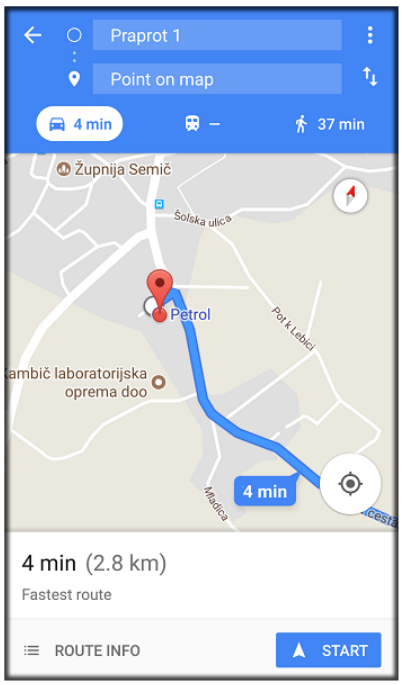
Slika 6.5: Zaslonska maska za izbiro iskalnega kriterija.

nizko stanje goriva v rezervoarju. Ob izbranem kriteriju in začetku iskanja s pritiskom na gumb Poišči servis se izbrani podatki skupaj s trenutno lokacijo posredujejo spletni storitvi, ki vrne ali najbližji in najcenejši servis ali pa opozorilo, da v izbranem dosegu ni nobenega bencinskega servisa. Če spletna storitev vrne podatke o servisu, potem se odpre zaslon, ki vsebuje lokacijske storitve Google Maps z možnostjo navigacije do izbranega bencinskega servisa. Poleg iskanja servisa imamo možnost preusmeritve na zaslon za pregled stroškov in na zaslon za vnos stroška ob nakupu goriva.

6.3.3 Navigacija

Zaslon za navigacijo se odpre ob uspešni pridobitvi podatkov bencinskega servisa od spletne storitve. Na zaslonu se izriše pot od trenutne lokacije do lokacije najbližjega in najcenejšega servisa. Izbira navigacije s pomočjo

lokacijske storitve Google Maps, ki velja za eno najbolj popularnih izbir za lokacijske storitve, tako pri uporabnikih Androida kot tudi iOSa, omogoča podrobna glasovna navodila, ki vključujejo prometna opozorila in navodila kdaj in kam zaviti, ter grafična navodila za želeno pot. Slika 6.6 prikazuje izris poti in možnost navigacije do bencinskega servisa.



Slika 6.6: Izris poti in možnost navigacije do bencinskega servisa.

Po uspešnem dosegu bencinskega servisa in morebitni napolnitvi rezervoarja se s pritiskom na gumb za vrnitev vrnemo na zaslon za iskanje servisa. Google Maps potrebuje za čim boljšo učinkovitost usmerjanja uporabnika ustrezno podatkovno povezavo, ki mu omogoča komunikacijo z napravami, ki omogočajo ustrezno določanje trenutne ter končne lokacije in s tem ustrezno delovanje aplikacije.

6.3.4 Beleženje stroškov

Po dosegu cilja imamo možnost, če se nam to zdi primerno, odpreti zaslonsko masko za vnos stroška, ki smo ga imeli ob morebitnem nakupu goriva. Naslov bencinskega servisa in njegov lastnik in s tem ponudnik uslug na bencinskem servisu ter cene posameznih goriv so še vedno na voljo ob koncu morebitne navigacije do servisa. Ob morebitni uporabi zaslona za beleženje stroška se zato samodejno izpiše naslov bencinskega servisa, pojavi se slika ponudnika goriv in pa gorivo, ki smo ga po vsej verjetnosti natočili, ker smo ga tudi iskali. Potreben je samo vnos količine goriva, ki je bila natočena, ob shranitvi stroška pa se izpiše znesek stroška, ki smo ga imeli z nakupom goriva, ter njegova shranitev v podatkovno bazo. Slika 6.7 prikazuje zaslonsko masko za vnos stroška.

Slika 6.7: Zaslonska maska za vnos stroška.

6.3.5 Pregled stroškov

Iz pregleda stroškov, ki smo jih imeli ob nakupu goriva ob napolnitvi rezervoarja ali ob kakšni drugi uslugi, ki se tiče nakupa goriva, se lahko razbere več stvari. Na zaslonu za pregled stroškov ima uporabnik možnost izbire kriterija za pregled stroškov.

Lahko izbira med naslednjimi možnostmi:

- pregled skupnega zneska vseh do sedaj zabeleženih stroškov,
- pregled skupnega zneska vseh stroškov, zabeleženih v tekočem letu,
- pregled skupnega zneska vseh stroškov, zabeleženih v zadnji polovici leta,
- pregled skupnega zneska vseh stroškov, zabeleženih v trenutnem mesecu.

Ob izbiri enega izmed naštetih kriterijev se na zaslonu aplikacije izpiše znesek trenutne skupne vsote vseh stroškov, ki so bili zabeleženi v izbranem obdobju.

6.3.6 Primer uporabe

Primer uporabe prikazuje uporabo aplikacije s strani uporabnika, ki še nima ustvarjenega uporabniškega računa za uporabo aplikacije. Ob zagonu aplikacije se prikaže stran s prijavo, ki omogoča prijavo, registracijo in zamenjavo gesla. Če uporabnik še nima uporabniškega računa, se usmeri na stran za registracijo z uporabo gumba "Registriraj" (povezava A), kjer se registrira. Po uspešni registraciji je preusmerjen nazaj na stran za prijavo, kjer vpiše svoje uporabniško ime in geslo ter pritisne gumb "Prijava". Preusmerjen je na glavno stran aplikacije (povezava B). Na glavni strani aplikacije izbere vrsto goriva in oddaljenost bencinskega servisa ter pritisne na gumb "Poišči servis" (povezava C). Pojavi se zemljevid Google z lokacijo najbližjega bencinskega

servisa z izbranim gorivom. Uporabnik uporabi navigacijo do servisa. Ko doseže cilj, pritisne gumb za vrnitev na glavno stran aplikacije, kjer izbere preusmeritev na stran za vnos stroška (povezava D). Na strani za vnos stroška vpiše količino goriva, ki jo je natočil, in ob pritisku na gumb "Shrani račun" se shrani vnesena količina in cena goriva, ki se izračuna na podlagi cene goriva na izbranem bencinskem servisu. S pritiskom gumba "Nazaj na iskanje" se uporabnik vrne na glavno stran. Na glavni strani se s pomočjo gumba "Pregled stroškov" preusmeri na stran za pregled stroškov (povezava E). Na strani za pregled stroškov izbere obdobje pregleda stroškov za trenutni mesec in prikaže se mu seštevek vseh stroškov, vnesenih v tekočem mesecu. Z gumbom "Nazaj na iskanje" se vrne na glavno stran. Na glavni strani pa pritisne na gumb "Odjava" in se tako odjavi in vrne na stran za prijavo v aplikacijo. Na strani za prijavo pritisne na gumb "Zamenjava gesla" (povezava F). Preusmerjen je na stran za zamenjavo gesla, kjer si z vpisom uporabniškega imena in pa trenutnega ter novega gesla spremeni geslo. Na koncu se vrne nazaj na prijavno stran s pomočjo gumba "Nazaj". Slika 6.8 prikazuje primer uporabe aplikacije.

Slika 6.8: Primer uporabe.

Poglavje 7

Sklepne ugotovitve

V sklopu diplomske naloge je bila razvita mobilna aplikacija za iskanje najcenejšega bencinskega servisa ter beleženje stroškov za kasnejšo analizo. Aplikacija vsebuje funkcionalnosti, ki uporabniku omogočajo iskanje najbližjega ter najcenejšega bencinskega servisa glede na trenutno lokacijo in pa glede na iskalne kriterije, ki jih uporabnik poda. Poleg tega omogoča tudi beleženje in pregled stroškov ob nakupih goriva.

Poleg mobilne aplikacije je bil izdelan tudi razčlenjevalnik podatkov, ki je služil za pridobitev podatkov o bencinskih servisih v Sloveniji. Podatki so bili vstavljeni v podatkovno bazo, kreirano s pomočjo Microsoft Visual Studia. Za pridobivanje in pošiljanje podatkov o bencinskih servisih, stroških in uporabnikih aplikacije v podatkovno bazo pa je bila mobilni aplikaciji v pomoč v Visual Studiu izdelana spletna storitev, ki je bila naložena na Microsoftov portal Azure skupaj s podatkovno bazo.

Mobilna aplikacija lahko služi uporabnikom motornih vozil za iskanje najcenejšega bencinskega servisa v Sloveniji s pomočjo mobilne naprave in njene funkcionalnosti GPS. Omogoča tudi shranjevanje vseh zneskov, ki jih je imel uporabnik ob nakupih goriva, ter njihov pregled po različnih obdobjih, ki jih uporabnik izbere v aplikaciji.

Opisana aplikacija predstavlja prvo različico. Obstaja več možnosti nadgradnje aplikacije, kot na primer razširitev iskalnega območja bencinskih ser-

visov na ostale evropske države. Ob tem bi bilo potrebno pridobiti podatke o lokacijah ter cenah na večini bencinskih servisov po Evropi, kar bi pomenilo zelo velik projekt. Za začetek bi se lahko lotili razširitve iskalnega kriterija na države, na katere meji Slovenija, nato pa bi iskanje postopoma razširili na območje celotne Evrope. Poleg tega bi se lahko nadgradilo pregled zabeleženih stroškov v bolj pregledno in razločno grafično upodobitev. Mogoča bi bila tudi nadaljnja analiza zbranih podatkov, kajti spletna storitev omogoča pridobivanje podatkov o lokaciji bencinskega servisa, na katerem je bil račun shranjen, višino zneska, vrsto goriva, uro shranitve računa in trgovca z naftnimi derivati. Iz naštetih podatkov bi bilo mogoče razbrati, kje in kdaj se proda največja količina določenega goriva, hkrati pa tudi, pri katerem ponudniku naftnih derivatov v Sloveniji se uporabniki motornih vozil največkrat odločijo za nakup goriva.

Literatura

[1] Android Studio. Dosegljivo: https://s3.amazonaws.com/infinum.web.production/repository_items/files/000/000/168/original/android-studio-3.png?1393599622. [Dostopano: 3. 9. 2017].

[2] Android Studio Features. Dosegljivo: https://developer.android.com/studio/features.html. [Dostopano: 3. 9. 2017].

[3] Azure platforma. Dosegljivo: https://www.nil.com/sl/microsoft/microsoft-azure/. [Dostopano: 3. 9. 2017].

[4] Azure lastnosti. Dosegljivo: https://www.arne.si/index.php/ponudba/microsoft-azure. [Dostopano: 3. 9. 2017].

[5] Cene goriv. Dosegljivo: http://www.energetika-portal.si/podrocja/energetika/cene-naftnih-derivatov/. [Dostopano: 7. 9. 2017].

[6] C#. Dosegljivo: https://sl.wikipedia.org/wiki/Programski_jezik_C_sharp#Razli.C4.8Dice. [Dostopano: 3. 9. 2017].

[7] DKV. Dosegljivo: https://www.dkv-euroservice.com/gb/global/about-dkv/company/. [Dostopano: 8. 9. 2017].

[8] Google Maps. Dosegljivo: https://sl.wikipedia.org/wiki/Google_Maps. [Dostopano: 8. 9. 2017].

[9] Haverine formula. Dosegljivo: https://en.wikipedia.org/wiki/Haversine_formula. [Dostopano: 14. 9. 2017].

[10] IDE. Dosegljivo: http://colos1.fri.uni-lj.si/ERI/RACUNALNISTVO/PROG_JEZIKI_ORODJA/prog_ide.html. [Dostopano: 2. 9. 2017].

[11] Intellij. Dosegljivo: http://www.omv.si/portal/01/si/omv_si/omv-slovenia/goriva/Cene. [Dostopano: 8. 9. 2017].

[12] Java. Dosegljivo: https://sl.wikipedia.org/wiki/Programski_jezik_java. [Dostopano: 3. 9. 2017].

[13] JetBrains PyCharm. Dosegljivo: https://i.stack.imgur.com/ByTTr.png. [Dostopano: 3. 9. 2017].

[14] Json. Dosegljivo: http://www.json.org/json-sl.html. [Dostopano: 3. 9. 2017].

[15] Json Example. Dosegljivo: https://forums.mulesoft.com/storage/temp/455-json-example.png. [Dostopano: 3. 9. 2017].

[16] Microsoft Visual Studio. Dosegljivo: https://www.visualstudio.com/wp-content/uploads/2017/02/LightBulb-LUT-1280x800.png. [Dostopano: 3. 9. 2017].

[17] Moj Petrol. Dosegljivo: http://www.petrol.si/mobilna-aplikacija-moj-petrol?utm_source=1val&utm_medium=bannerportalmojpetrol&utm_campaign=moj_petrol&utm_term=kw21. [Dostopano: 8. 9. 2017].

[18] Na poti. Dosegljivo: https://play.google.com/store/apps/details?id=si.petrol.napoti. [Dostopano: 8. 9. 2017].

[19] OMV. Dosegljivo: https://www.omv.com/portal/01/com/omv/OMV_Group/products/OMV_Stationfinder_App. [Dostopano: 8. 9. 2017].

[20] Omv Cene. Dosegljivo: http://www.omv.si/portal/01/si/omv_si/omv-slovenia/goriva/Cene. [Dostopano: 8. 9. 2017].

[21] Petrol Cene. Dosegljivo: http://www.petrol.si/na-poti/za-vozilo/goriva-q-max/goriva-q-max. [Dostopano: 8. 9. 2017].

[22] Postman. Dosegljivo: https://www.getpostman.com/postmana. [Dostopano: 9. 9. 2017].

[23] PyCharm. Dosegljivo: https://en.wikipedia.org/wiki/PyCharm. [Dostopano: 2. 9. 2017].

[24] Visual Studio. Dosegljivo: https://sl.wikipedia.org/wiki/Visual_Studio. [Dostopano: 3. 9. 2017].

[25] Visual Studio Compare. Dosegljivo: https://www.visualstudio.com/vs/compare/. [Dostopano: 3. 9. 2017].