

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Simon Perčič

**Informacijska rešitev za iskanje in
ocenjevanje restavracij**

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

MENTOR: doc. dr. Dejan Lavbič

Ljubljana, 2014

Rezultati diplomskega dela so intelektualna lastnina avtorja in Fakultete za računalništvo in informatiko Univerze v Ljubljani. Za objavlanje ali izkoriščanje rezultatov diplomskega dela je potrebno pisno soglasje avtorja, Fakultete za računalništvo in informatiko ter mentorja.

Besedilo je oblikovano z urejevalnikom besedil $\text{\LaTeX}$.



Št. naloge: 01971 / 2013
Datum: 5.11.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **SIMON PERČIČ**

Naslov: **INFORMACIJSKA REŠITEV ZA ISKANJE IN OCENJEVANJE
RESTAVRACIJ
INFORMATION SYSTEM FOR FINDING AND EVALUATING
RESTAURANTS**

Vrsta naloge: DIPLOMSKO DELO UNIVERZITETNEGA ŠTUDIJA

Tematika naloge:

Na področju turizma so na voljo številne informacijske rešitve, ki uporabnikom olajšajo spoznavanje novih krajev, omogočajo spoznavanje kulinarčnih skrivnosti območja kjer se nahajajo idr. Nekatere izmed njih podpirajo tudi priporočila prijateljev, ki jih uporabniki pogosto v veliki meri upoštevajo. Pri pregledu stanja identificirajte težave obstoječih rešitev in v okviru diplomske naloge razvijte sistem, ki temelji na obstoječih podatkih o restavracijah, ki jih najdete na spletu, t.j. njihovih ocenah in drugih razpoložljivih podatkih. Uporabniku ponudite možnost podrobnega ocenjevanja restavracij, kjer pri modelu ocenjevanja opredelite različne kriterije, za katere lahko uporabnik poda oceno ter upoštevajte tudi priporočila prijateljev. Rešitev naj omogoča tudi poljubno sortiranje rezultatov po opredeljenih kriterijih. Rezultat naj bo predstavljen v obliki delujočega prototipa mobilne aplikacije, ki komunicira s strežniškim sistemom in je dovolj skalabilna tudi v primeru naraščajočega števila uporabnikov.

Mentor:

doc. dr. Dejan Lavbič



Dekan:

prof. dr. Nikolaj Zimic

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Spodaj podpisani Simon Perčič, z vpisno številko **63070195**, sem avtor diplomskega dela z naslovom:

Informacijska rešitev za iskanje in ocenjevanje restavracij

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Dejana Lavbiča,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 12. februarja 2014

Podpis avtorja:

Kazalo

Povzetek

Abstract

1	Uvod	1
2	Pregled sorodnih rešitev	3
2.1	Foursquare	4
2.2	TripAdvisor	5
2.3	Odpiralni Časi	7
2.4	Google Maps	8
2.5	Ostale storitve	9
3	Uporabljene tehnologije	11
3.1	Android aplikacija	11
3.2	Strežniški sistem	12
4	Predlog informacijske podpore iskanju in ocenjevanju restavracij	15
4.1	Iskanje, razvrščanje in filtriranje restavracij	17
4.2	Podrobnejši podatki o restavraciji	22
4.3	Dodajanje restavracije	27
4.4	Prijava in registracija	29
4.5	Uporabniški profil	31
4.6	Sheme prehodov med zaslonskimi maskami	37

KAZALO

5	Implementacija rešitve	43
5.1	Android aplikacija	43
5.2	Strežniški sistem	51
6	Zaključek	75
6.1	Nadaljnje delo	76

Povzetek

V okviru diplomske naloge smo zasnovali in razvili celotno informacijsko rešitev za iskanje in ocenjevanje restavracij. Na tem področju že obstajajo druge rešitve, a po naši raziskavi nobena izmed njih ne ponuja vseh funkcionalnosti, ki jih omogoča naša rešitev.

Razvili smo lastno rešitev, ki omogoča ocenjevanje in razvrščanje restavracij po posameznih kriterijih (kakovost hrane, prijaznost osebja, hitrost postrežbe in primernost cene). Naša rešitev poleg tega omogoča omejevanje iskanja restavracij glede na izbrano oddaljenost od uporabnikove lokacije z uporabo zunanje storitve Google Maps, ki omogoča izračun razdalje med različnimi točkami. S pomočjo zunanje storitve Foursquare smo za vsako restavracijo pridobili dodatne podatke, ki so prikazani na podrobnem pogledu restavracije. Uporabniki se lahko v našo storitev prijavijo in v sistem dodajo nove restavracije. Podprto je tudi iskanje in dodajanje prijateljev z namenom razvrščanja restavracij glede na njihove ocene.

Sistem je sestavljen iz aplikacije za mobilni operacijski sistem Android in strežniškega sistema, ki je postavljen na platformi Windows Azure.

Ključne besede: ocenjevanje restavracij, Android, Windows Azure, Foursquare, Google Maps, pridobivanje podatkov

Abstract

In this thesis we have designed and developed an information system for finding and evaluating restaurants. There are already some existing services available in this field, however based on our investigation none of them provides all of the functionalities that are included in our solution.

We have developed our own solution that allows evaluating and sorting restaurants by different criteria (food quality, staff friendliness, speed of service and price/performance ratio). Our solution also provides the option to filter restaurants based on the maximum distance from the user's location, which the user can select in the application. The filtering is achieved using the external service Google Maps, which provides the functionality of calculating distances between different locations. Using the external service Foursquare, we have obtained additional data for every restaurant, which is shown on restaurant details view. Using our application, users can login into their accounts and add new restaurants to the service. Users can also search and add other users as friends in order to sort the restaurants by friends' ratings.

The system consists of a mobile application for the Android mobile operating system and a server-side application that runs on the Windows Azure platform.

Keywords: evaluating restaurants, Android, Windows Azure, Foursquare, Google Maps, data scraping

Poglavje 1

Uvod

V vsakdanjem življenju se poleg zapletenih strokovnih vprašanj dnevno srečujemo z navidez popolnoma preprostimi odločitvami. Ena izmed osnovnih človeških potreb je potreba po prehranjevanju. Skoraj vsak dan se tako srečamo z odločitvijo, kam naj gremo jesti. Ko smo v poznani okolici, navadno že imamo približno idejo, kam lahko gremo jesti. V nepoznani okolici pa postane odločitev težja, saj nimamo ustreznih informacij in izkušenj, ki bi nam pomagale pri tej odločitvi.

Dandanes ima čedalje več ljudi vedno pri sebi priročno napravo z dovolj zmogljivim procesorjem, primerno velikim zaslonom, modulom za natančno določanje uporabnikove lokacije in dostopom do interneta. Opisane lastnosti pametnega mobilnega telefona bi lahko uporabili za pridobivanje ustreznih informacij pri iskanju primerne restavracije v naši trenutni okolici. Na tem področju že obstaja nekaj podobnih storitev, a po naši raziskavi nobena izmed njih ne ponuja ocenjevanja restavracij po posameznih kriterijih (recimo po oceni kakovosti hrane, prijaznosti osebja, hitrosti postrežbe itd.). Ker nas pri iskanju restavracij zanima tudi, katere restavracije na seznamu restavracij so nam dovolj blizu, bo naša rešitev omogočala omejevanje radija iskanja glede na našo trenutno lokacijo. Poleg tega bo naša rešitev omogočala dodajanje prijateljev, tako da bo mogoče pregledovanje lestvice restavracij glede na ocene prijateljev. Tako bomo dobili predloge za restavracije od ljudi, ki

jim najbolj zaupamo, torej od naših prijateljev in znancev.

Namen tega diplomskega dela je zasnovati in implementirati celoten sistem za iskanje in ocenjevanje gostinskih lokalov in restavracij. Sistem bo sestavljen iz aplikacije, ki teče na pametnem mobilnem telefonu, in iz strežniškega sistema, ki skrbi za hranjenje podatkov in poslovno logiko sistema. Strežniški sistem mora biti hiter, stabilen in skalabilen, da se zagotovi pričakovan nivo storitve. Strežniški sistem bo uporabljal zunanjo spletno storitev za pridobivanje dodatnih podatkov o restavracijah. Poleg tega bo uporabljal tudi zunanjo lokacijsko storitev za pomoč pri določanju razdalje med uporabnikom in zeleno lokacijo. Cilj je implementirati sistem, ki bo dovolj stabilen in robusten za uporabo v resničnem svetu.

Naše diplomsko delo je razdeljeno na štiri dele. V prvem delu bomo raziskali že obstoječe rešitve, ki so namenjene iskanju ali ocenjevanju restavracij. Naredili bomo kritično primerjavo njihovih funkcionalnosti v primerjavi z našo rešitvijo. V naslednjem poglavju bomo predstavili zanimivejše in pomembnejše programske tehnologije, ki smo jih uporabili pri razvoju našega sistema. V tretjem delu bomo podali predlog informacijske podpore za iskanje in ocenjevanje restavracij. V tem poglavju bomo tudi podrobno opisali vse funkcionalnosti iz uporabniškega vidika. V zadnjem delu bomo opisali še tehnične podrobnosti implementacije celotnega sistema, tako podrobnosti aplikacije kot tudi strežniškega sistema.

Poglavje 2

Pregled sorodnih rešitev

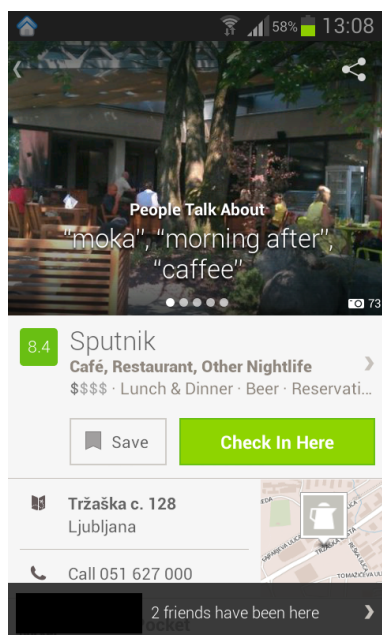
Trenutno je na voljo veliko storitev, kjer lahko uporabniki z uporabo mobilne naprave najdejo želeno restavracijo.

Kljub temu po našem pregledu ob času pisanja ni na voljo nobene storitve, ki bi bila namenjena ocenjevanju restavracij po posameznih kriterijih (recimo po oceni kakovosti hrane, prijaznosti osebja, ambientu, oceni primernosti cene, hitrosti postrežbe itd.). V večini obstoječih rešitev lahko restavracije ocenjujemo samo s splošno oceno, ali z načinom ‘všeč mi je’ in ‘ni mi všeč’ (ang. like, dislike).

Poleg tega večina storitev ne nudi dodajanja prijateljev z namenom pridobivanja priporočil glede na njihove ocene. Pridobivanje priporočil naših prijateljev je zelo pomembno, saj se v vsakdanjem življenju mnogokrat izkaže, da najbolj zaupamo prav ljudem, ki jih tudi v resničnem življenju poznamo, ne pa naključnim uporabnikom spleta.

Nekatere izmed spodaj opisanih storitev omogočajo pregledovanje in razvrščanje seznama restavracij, pri drugih pa je možno le iskanje po imenu, brez pregleda seznama restavracij.

V nadaljevanju bodo opisane storitve za ocenjevanje restavracij, ki so v zasnovi in namenu najbolj podobne naši rešitvi. Pri teh smo se omejili na tiste, ki ponujajo mobilno aplikacijo na operacijskem sistemu Android. Kljub tej omejitvi so zajete vse pomembne storitve.



Slika 2.1: Pogled restavracije v mobilni aplikaciji Foursquare

2.1 Foursquare

Foursquare [21] je storitev, namenjena iskanju lokacij in prijavljanju (ang. check in) na teh lokacijah. Med te lokacije spadajo tudi restavracije. Uporabniki lahko dodajajo nove lokacije (ang. venue), se prijavijo na lokaciji, iščejo lokacije po kategoriji ali imenu, urejajo podatke določene lokacije, ji dodajajo slike, jo ocenijo in komentirajo. Seznam lokacij je možno razvrstiti po dveh parametrih, po razdalji od uporabnika in po najboljšem ujemanju. Najboljše ujemanje pomeni, da bodo lokacije razvrščene glede na ocene drugih uporabnikov in glede na ocene uporabnikovih prijateljev. Seznam je prav tako mogoče filtrirati po določenih Foursquare-ovih parametrih, kot so: prikaz lokacij, ki smo jih shranili, prikaz lokacij kamor smo se/se nismo prijavili, kamor so se prijavili naši prijatelji...

Uporabniki lahko restavracije (ali druge lokacije) ocenjujejo le po principu 'všeč mi je' in 'ni mi všeč' ali pa tako, da napišejo komentar brez ocene. Skupna ocena za določeno restavracijo je tako določena glede na razmerje

med uporabniki, ki so restavracijo ocenili z ‘všeč mi je’ in uporabniki, ki so jo ocenili z ‘ni mi všeč’. Uporabniki lahko tudi vidijo komentarje ostalih uporabnikov in po želji označijo da se z njimi strinjajo. Komentarji, s katerimi se strinja največ uporabnikov, bodo višje na seznamu vseh komentarjev. Na sliki 2.1 je viden pogled posamezne restavracije v aplikaciji Foursquare.

Uporabniki lahko tudi dodajo ostale uporabnike kot prijatelje in s tem dobijo obvestila o njihovih prijavah. Poleg tega zna storitev predlagati najboljše lokacije glede na ocene drugih uporabnikov in prijateljev (glede na število prijateljev, katerim je določena lokacija všeč, so bili na določeni lokaciji...).

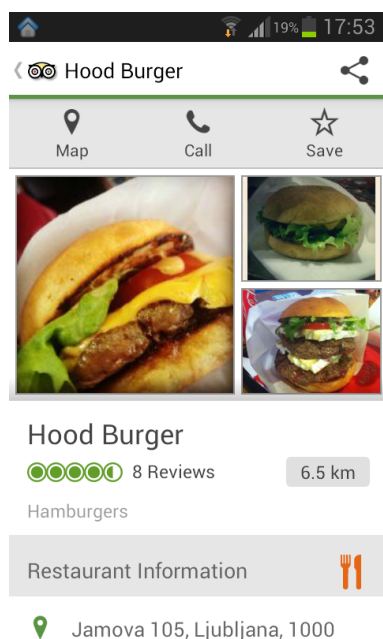
V primerjavi z rešitvijo, ki je tema te diplomske naloge, Foursquare ne ponuja možnosti ocenjevanja restavracij po posameznih kriterijih, niti skupne ocene s številčno oceno. Prav tako na Foursquare-u ni mogoče videti lestvice restavracij glede na oceno posameznega kriterija in oceno prijateljev. Naša rešitev poleg tega omogoča še iskanje v točno določenem radiju od uporabnikove trenutne lokacije.

2.2 TripAdvisor

TripAdvisor [26] je največja stran za načrtovanje potovanj na svetu [27]. Stran tako med drugim ponuja ocene nastanitev, restavracij in znamenitosti za kraje po celem svetu.

Pri opisu se bomo omejili na iskanje in ocenjevanje restavracij, saj je to tisti del storitve TripAdvisor, ki je podoben temi diplomske naloge. Uporabniki aplikacije TripAdvisor lahko pregledujejo seznam restavracij, ki je razvrščen po skupni oceni, razdalji od uporabnika ali po kombinaciji obeh faktorjev (torej najboljše bližnje restavracije). Uporabniki lahko rezultate omejijo po cenovnem razredu (4 razredi, od najcenejšega do najdražjega) in po tipu restavracije (italijanska, indijska, kitajska...).

Za izbrano restavracijo lahko uporabniki vidijo opise z ocenami ostalih uporabnikov, ki so ocenjeni s skupno oceno od 1 do 5. Ti opisi so razvrščeni



Slika 2.2: Pogled restavracije v mobilni aplikaciji TripAdvisor

od najnovejšega do najstarejšega, glede na datum objave. Uporabniki lahko tudi dodajajo svoje opise z ocenami. Tako kot pri Foursquare-u lahko tudi tu uporabniki dodajajo svoje slike izbrane restavracije. Uporabniki se lahko v aplikacijo prijavijo tudi s svojim Facebook profilom. V tem primeru je na seznamu restavracij pri vsaki restavraciji označeno, ali jo je ocenil, opisal ali všečkal kateri izmed uporabnikovih Facebook prijateljev. Pogled posamezne restavracije v aplikaciji TripAdvisor je viden na sliki 2.2.

V primerjavi z našo rešitvijo, TripAdvisor prav tako kot Foursquare ne ponuja možnosti ocenjevanja restavracij po posameznih kriterijih in posledično pregleda lestvice restavracij glede na posamezen kriterij. Aplikacija prav tako ne omogoča iskanja v določenem radiju od uporabnikove trenutne lokacije. Poleg tega uporabniki ne morejo dodati novih restavracij, to lahko stori le njen lastnik preko spletne strani TripAdvisor.com.

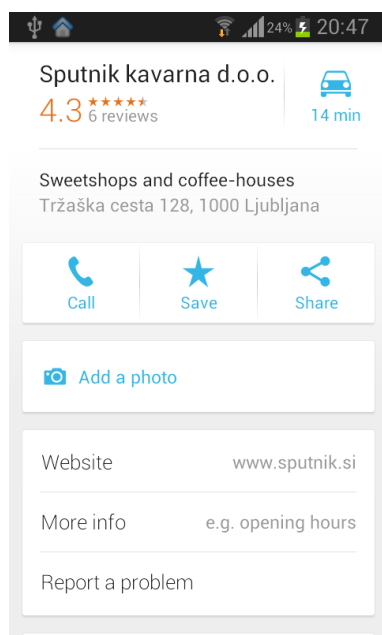


Slika 2.3: Pogled lokacije v mobilni aplikaciji Odpiralni Časi

2.3 Odpiralni Časi

Odpiralni Časi [28] je slovenska storitev, namenjena pridobivanju informacij o odpiralnih časih podjetij. Stran vsebuje tako lokacije, ki imajo opredeljene odpiralne čase (trgovine, banke, gostinske lokale, restavracije...), kot tudi lokacije, ki nimajo opredeljenega odpiralnega časa (bankomati, Bicikelj postaje, WIFI točke...). Uporabniki lahko dodajajo nove lokacije in jim dodajo delovni čas, kontaktne podatke in slike.

Seznam lokacij je razvrščen le glede na trenutno oddaljenost od uporabnika, ne pa tudi po oceni. Filtrirati ga je mogoče tako, da prikazuje vse ali pa samo odprte lokacije. Podrobni pogled posamezne lokacije prikazuje njen odpiralni čas, informacijo o tem ali je lokacija trenutno odprta ali ne, njene osnovne podatke (naslov, opis, telefonsko številko, spletno stran...) in njene slike. Na podstrani posamezne lokacije lahko vidimo tudi opise in ocene uporabnikov, kjer so ocene podane z lestvico od 1 do 5. Poleg tega lahko vsak doda tudi svoj opis in oceno za določeno lokacijo. Na sliki 2.3 je viden pogled



Slika 2.4: Pogled lokacije v mobilni aplikaciji Google Maps

posamezne lokacije.

V primerjavi z rešitvijo, ki je tema te diplomske naloge, aplikacija Odpiralni Časi ne omogoča ocenjevanja restavracij po posameznih kriterijih in pregleda lestvice restavracij glede na njihovo oceno (tudi ne glede na splošno oceno). Storitev tudi ne omogoča dodajanja prijateljev, poleg tega mobilna aplikacija ne omogoča prijave v uporabniški profil.

2.4 Google Maps

Aplikacija Google Maps je primarno namenjena prikazu zemljevida in iskanju naslovov. Omogoča pa tudi iskanje in ocenjevanje različnih lokacij in podjetij, med njimi tudi restavracij.

Aplikacija Google Maps omogoča pregledovanje osnovnih podatkov o lokaciji (ime, naslov, spletna stran, telefonska številka...) in opisov z ocenami, ki so jih dodali ostali uporabniki. Vidna je tudi skupna ocena (med 1 in 5)

vseh uporabnikov. Omogočeno je tudi dodajanje nove ocene in/ali opisa. Pogled lokacije je viden na sliki 2.4. Google Maps ne omogoča pregleda lestvice restavracij in razvrščanja ter filtriranja tega seznama.

V primerjavi z našo rešitvijo Google Maps omogoča le ocenjevanje in pregled skupne ocene uporabnikov, ne pa vseh ostalih funkcionalnosti. Naša rešitev tako omogoča še pregled lestvice restavracij in razvrščanje ter filtriranje tega seznama, ocenjevanje restavracij po posameznih kriterijih, prijavo v uporabniški profil, iskanje ter dodajanje prijateljev in dodajanje novih restavracij.

2.5 Ostale storitve

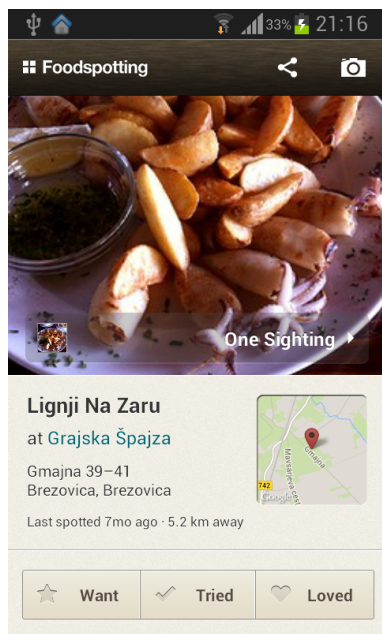
Poleg zgoraj naštetih storitev so na voljo še druge storitve, ki so specializirane za restavracije in druge gostinske lokale. Nekatere izmed njih so v svoji zasnovi in namenu precej drugačne od teme diplomske naloge, druge pa so namenjene za tuja območja oziroma ne delujejo v Sloveniji. V nadaljevanju bo na kratko opisanih nekaj izmed njih.

2.5.1 Foodspotting

Storitev Foodspotting [29] je namenjena deljenju in iskanju fotografij jedi iz restavracij in drugih gostinskih lokalov. Uporabniki lahko tako ocenijo posamezno fotografijo s tremi izbirami: želim (ang. want), poizkusil (ang. tried) in vzljubil (ang. loved). Na sliki 2.5 je primer prikaza posamezne jedi.

2.5.2 Urbanspoon

Urbanspoon [30] je storitev, namenjena iskanju in ocenjevanju restavracij. Poleg osnovnih podatkov o restavraciji so na voljo še ocene uporabnikov na način ‘všeč mi je’ in ‘ni mi všeč’, opisi uporabnikov in povezave na opise z ocenami kritikov hrane ter blogov.



Slika 2.5: Pogled posamezne jedi v mobilni aplikaciji Foodspotting

Storitev je v času pisanja namenjena samo restavracijam, ki se nahajajo v ZDA, Kanadi, Združenem kraljestvu, Avstraliji, Novi Zelandiji in na Irskem.

2.5.3 Yelp

Storitev Yelp [31] je namenjena iskanju in ocenjevanju lokalnih podjetij, med katere sodijo tudi restavracije. Vsako podjetje je predstavljeno z osnovnimi podatki in opisi s številčnimi ocenami uporabnikov (od 1 do 5). Uporabniki se lahko tudi prijavijo na določeno lokacijo (ang. check in), tako kot na storitvi Foursquare.

V času pisanja storitev ni vsebovala slovenskih lokacij, prav tako ni bilo možno dodati slovenskih podjetij. Poleg tega storitev v času pisanja ni na voljo v slovenskem Google Play Store in s tem ni na voljo slovenskim Android uporabnikom.

Poglavje 3

Uporabljene tehnologije

V tem poglavju bodo predstavljene zanimivejše in pomembnejše tehnologije, ki smo jih uporabili pri implementaciji naše storitve za ocenjevanje restavracij. Opisane bodo tako tehnologije, ki smo jih uporabili pri razvoju Android aplikacije, kot tiste, ki smo jih uporabili pri razvoju strežniškega sistema.

3.1 Android aplikacija

3.1.1 Google Play Services

Google Play Services [11] knjižnica omogoča Android aplikacijam uporabo Google storitev, kot so Google Maps, Google+, Google Ads...

V naši aplikaciji to knjižnico uporabljamo za prikaz restavracije oziroma večih restavracij na Google Maps zemljevidu.

3.1.2 AQuery

AQuery oziroma Android-query [8] je odprtokodna knjižnica za Android aplikacije. Med drugim omogoča enostavno ustvarjanje asinhronih spletnih zahtevkov, asinhrono nalaganje slik preko HTTP protokola, razčlenjevanje XML dokumentov in enostavno manipulacijo gradnikov uporabniškega vmesnika Android aplikacije.

V okviru naše aplikacije smo knjižnico uporabili pri implementaciji ogrodja za komunikacijo s strežnikom (opisanega v razdelku 5.1.3), kjer knjižnica skrbi za ustvarjanje asinhronih zahtevkov na strežnik.

3.1.3 Android Support Library

Android Support Library [10] je Android knjižnica, ki omogoča povratno združljivost novih funkcionalnosti Android ogrodja. To pomeni, da lahko tudi na starejših verzijah Android operacijskega sistema uporabljamo določene funkcije Android ogrodja, ki so na voljo šele v novejših verzijah operacijskega sistema. Tako lahko uporabljamo nove funkcije na vseh napravah, ki podpirajo Android od verzije 1.6 (API nivo v4) naprej.

V okviru naše aplikacije je ta knjižnica zelo pomembna za delovanje celotne aplikacije, saj lahko uporabljamo Android fragmente (opisane v razdelku 5.1.1) na Android operacijskih sistemih od verzije 2.3.3 (API nivo v10) naprej, čeprav je bil način grajenja uporabniškega vmesnika s fragmenti na voljo šele v verziji 3.0 (API nivo v11) operacijskega sistema.

3.1.4 Google Gson

Google Gson [9] je Java knjižnica, ki omogoča deserializacijo JSON nizov v Java objekte in tudi obratno, torej serializacijo Java objektov v JSON nize.

V naši aplikaciji smo jo uporabili pri pošiljanju zahtevkov na strežnik (serializacija Java objektov v JSON nize) in pri prejemanju rezultatov s strežnika (deserializacija JSON nizov v Java objekte).

3.2 Strežniški sistem

3.2.1 NHibernate

NHibernate [32] je odprtokodna .NET knjižnica za delo s podatkovno bazo. Nudi programsko ogrodje za preslikavo objektno orientiranega domenskega

modela na tradicionalno relacijsko bazo (ORM, ang. object-relational mapping). Primarna funkcija NHibernate je preslikava med .NET razredi in tabelami v bazi. Poleg tega nudi tudi metode za podatkovne poizvedbe in shranjevanje podatkov v podatkovno bazo. Opis funkcionalnosti je povzet po [33].

Knjižnico smo v strežniškem sistemu uporabili za enostavnejši dostop do podatkovne baze.

3.2.2 Protobuf-net

Protobuf-net [34] je odprtokodna .NET knjižnica za serializacijo in deserializacijo podatkov, ki deluje po Googlovem serializacijskem mehanizmu Protocol Buffers [35]. Protocol Buffers je mehanizem za serializacijo strukturiranih podatkov, ki je neodvisen od programskega jezika in platforme ter omogoča razširljivost. Razširljivost v tem kontekstu pomeni, da lahko serializacijski shemi dodamo nove vrednosti, a bomo kljub temu ohranili kompatibilnost s prejšnjimi verzijami serializiranih podatkov. Ker gre za binarno serializacijo, ki ni berljiva ljudem, je tak način serializacije primeren za komunikacijske protokole, shranjevanje podatkov itd. V primerjavi z drugimi načini serializacije (XML, JSON itd.) je Protocol Buffers veliko hitrejši pri serializaciji in deserializaciji podatkov, poleg tega pa serializirani podatki zasedejo manj prostora. Primerjava hitrosti delovanja in učinkovitosti serializacije je dostopna na [36]. Format serializacije je opisan na [37].

V našem strežniškem sistemu smo knjižnico uporabili za predpomnjenje rezultatov zahtev na Google Maps API, opisano v razdelku 5.2.5.

Poglavje 4

Predlog informacijske podpore iskanju in ocenjevanju restavracij

Lastno storitev za ocenjevanje restavracij smo razvili zato, ker nobena izmed že obstoječih sorodnih storitev ne omogoča določenih funkcionalnosti pri pregledu in ocenjevanju restavracij.

Naša rešitev za pregledovanje in ocenjevanje restavracij omogoča naslednje funkcije:

- ocenjevanje restavracij po različnih kriterijih,
- razvrščanje restavracij glede na različne kriterije ocen,
- filtriranje restavracij glede na razdaljo od uporabnika,
- prikaz restavracij v obliki seznama ali na zemljevidu,
- prikaz podrobnejših podatkov o restavraciji, njenih ocen in podatkov iz zunanjih virov,
- dodajanje nove restavracije (s predlaganjem naslova glede na trenutno lokacijo),

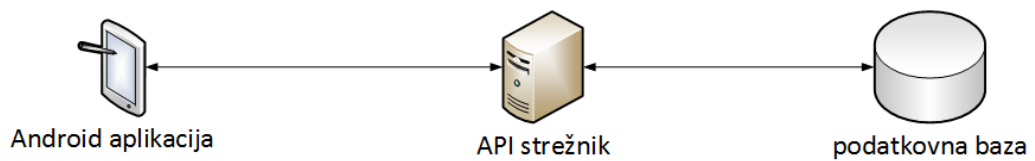
- prijava in registracija uporabnika,
- pregled uporabniškega profila in zgodovine ocen,
- iskanje in dodajanje prijateljev,
- razvrščanje restavracij glede na ocene prijateljev.

Naša rešitev v primerjavi z že obstoječimi rešitvami omogoča več funkcionalnosti. Nobena izmed že obstoječih storitev ne omogoča vseh spodaj naštetih funkcionalnosti, ki pa jih omogoča naša rešitev:

- ocenjevanje restavracij po posameznih kriterijih (ocena hrane, ocena primernosti cene, ocena hitrosti postrežbe in ocena prijaznosti osebja restavracije),
- pregled seznama restavracij glede na zgoraj naštete kriterije,
- filtriranje seznama restavracij glede na trenutno razdaljo od uporabnikove lokacije,
- dodajanje prijateljev z namenom razvrščanja seznama restavracij glede na njihove ocene.

Naš sistem za ocenjevanje restavracij je sestavljen iz dveh delov - iz aplikacije na mobilnem telefonu uporabnika in iz zalednega strežniškega sistema. Slednji skrbi za shranjevanje podatkov in za vso poslovno logiko sistema. Aplikacija preko interneta dostopa do spletnega vmesnika (APIja) na strežniku in s tem pridobi vse potrebne podatke za delovanje. Osnovna shema komunikacije med aplikacijo na mobilnem telefonu in strežnikom je prikazana na sliki 4.1.

V nadaljevanju bodo podrobno opisane vse funkcionalnosti, ki jih omogoča naša aplikacija. Funkcionalnosti bodo predstavljene v obliki zaslonских mask, ki sestavljajo Android aplikacijo.



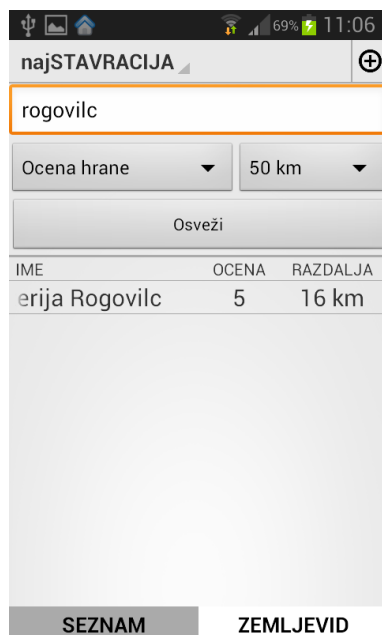
Slika 4.1: Shema komunikacije med Android aplikacijo in strežnikom



Slika 4.2: Iskanje restavracij

4.1 Iskanje, razvrščanje in filtriranje restavracij

Zaslonska maska, vidna na sliki 4.2, omogoča uporabnikom pregledovanje seznama restavracij in je začetna zaslonka maska pri odpiranju aplikacije. Restavracije na tem seznamu lahko uporabniki dodatno omejijo in jih razvrstijo.



Slika 4.3: Iskanje restavracij po imenu

4.1.1 Omejevanje rezultatov

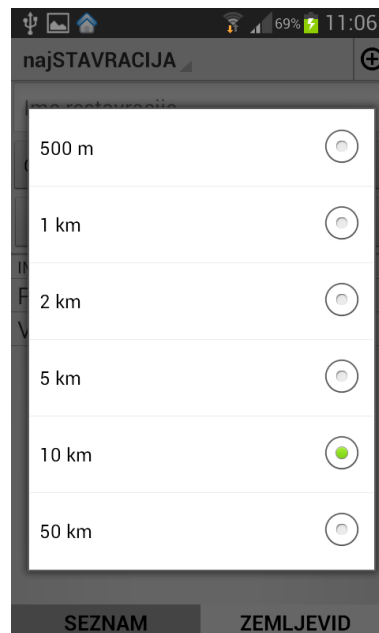
Uporabniki lahko restavracije omejijo glede na njihovo ime ali glede na trenutno razdaljo od uporabnika (v primeru, da je lokacija uporabnika znana).

Iskanje po imenu restavracije

Uporabniki lahko iščejo restavracije po imenu tako, da v vnosno polje vnesejo polno ali delno ime restavracije. Prikazan bo seznam vseh restavracij, ki v imenu vsebujejo vpisani niz. Prikaz iskanja je viden na sliki 4.3.

Prikaz bližnjih restavracij

Uporabnik lahko iz spustnega seznama, vidnega na sliki 4.4, omeji prikaz restavracij na tiste, ki so dovolj blizu trenutni lokaciji glede na izračunano pot s pomočjo storitve Google Maps. Izračun razdalje je možen le, če je uporabnikova trenutna lokacija znana. Ta se pridobi iz Android naprave



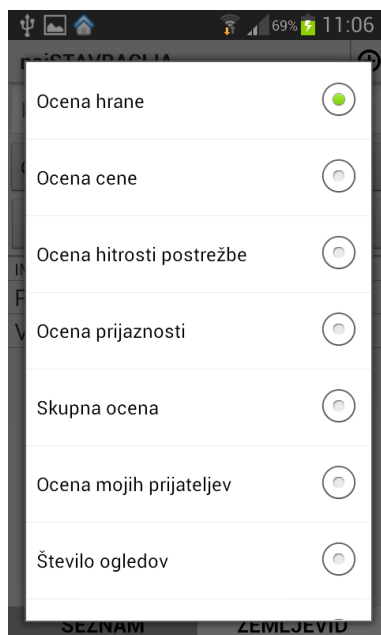
Slika 4.4: Omejevanje največje razdalje do restavracije

s pomočjo GPS modula, v primeru da tega modula naprava nima, je ta izklopljen ali nima signala, pa se lokacijo pridobi s pomočjo triangulacije med baznimi postajami za mobilno omrežje in WIFI signali [5]. Uporabnik lahko največjo razdaljo do restavracije omeji na naslednje razdalje: 500 m, 1 km, 2 km, 5 km, 10 km in 50 km.

4.1.2 Razvrščanje rezultatov

Uporabniki lahko rezultate razvrstijo glede na želen kriterij razvrščanja. Kriteriji razvrščanja so vidni na sliki 4.5 in so sledeči:

- povprečna ocena hrane v restavraciji,
- povprečna ocena primernosti cene hrane v restavraciji,
- povprečna ocena hitrosti postrežbe osebja restavracije,

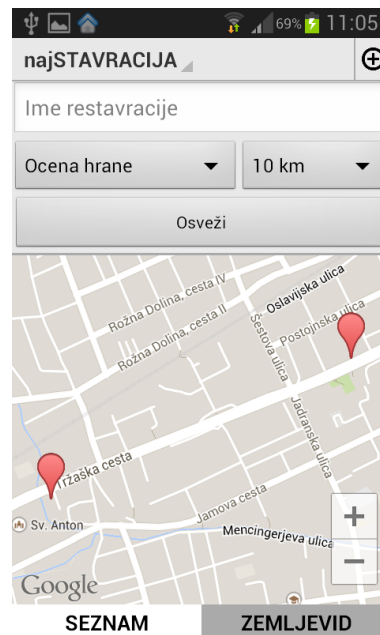


Slika 4.5: Razvrščanje restavracij

- povprečna ocena prijaznosti osebja restavracije oziroma vzdušja v restavraciji,
- skupna povprečna ocena po vseh zgoraj naštetih kriterijih,
- skupna povprečna ocena restavracije uporabnikovih prijateljev (možna samo, če je uporabnik prijavljen v aplikacijo in je preko nje našel in dodal vsaj enega prijatelja),
- število ogledov restavracije preko aplikacije,
- število oddanih ocen restavracije preko aplikacije.

4.1.3 Prikaz rezultatov iskanja

Rezultati iskanja so lahko prikazani v obliki seznama ali zemljevida.



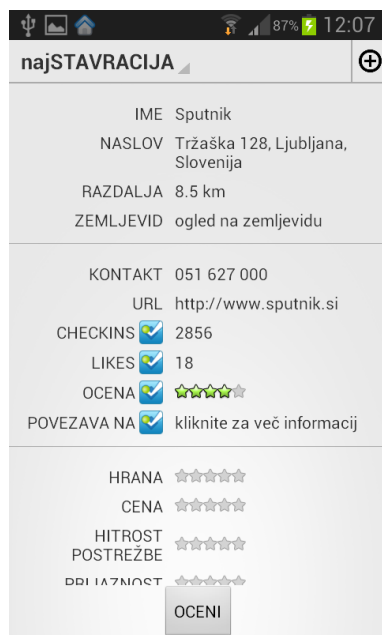
Slika 4.6: Prikaz rezultatov iskanja restavracij na zemljevidu

Seznam

Privzeto so restavracije prikazane v obliki seznama, razvrščenega glede na izbran kriterij razvrščanja. V seznamu je vsaka restavracija predstavljena z imenom, vrednostjo kriterija razvrščanja (bodisi oceno po določenem kriteriju, številom ogledov ali številom ocen) in trenutno razdaljo do restavracije (v primeru, da je uporabnikova lokacija znana). Z izbiro posamezne restavracije se prikaže pogled podrobnejših podatkov o restavraciji, opisan v podpoglavju 4.2.

Zemljevid

Poleg seznama restavracij lahko uporabniki rezultate iskanja vidijo tudi na zemljevidu restavracij, kot je razvidno iz zaslonske maske na sliki 4.6. Na zemljevidu je vsaka restavracija prikazana v obliki značke (ang. marker). Stopnja povečave zemljevida in center zemljevida sta nastavljena tako, da so naenkrat vidne vse značke restavracij, tako da naknadno premikanje oziroma

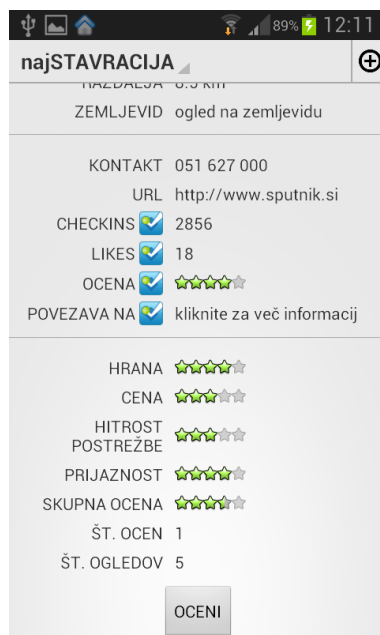


Slika 4.7: Podrobnejši podatki o restavraciji - 1. del

spreminjanje povečave ni potrebno. Po želji pa lahko uporabnik vseeno naknadno premika zemljevid in spreminja stopnjo povečave, da si podrobneje ogleda želeno regijo na zemljevidu. Ob pritisku na posamezno značko se odpre oblaček, v katerem je izpisano ime restavracije. Ob nadaljnjem pritisku na oblaček se prikaže pogled podrobnejših podatkov o restavraciji, opisan v nadaljevanju.

4.2 Podrobnejši podatki o restavraciji

Zaslonska maska, vidna na slikah 4.7 in 4.8, omogoča prikaz podrobnejših podatkov o posamezni restavraciji in prikaz lokacije restavracije na zemljevidu. Prijavljeni uporabnik lahko na tem pogledu poleg tega odpre dialog za ocenjevanje restavracije.



Slika 4.8: Podrobnejši podatki o restavraciji - 2. del

4.2.1 Osnovni podatki o restavraciji

Osnovni podatki, ki se prikažejo na tej zaslonski maski so:

- ime restavracije,
- naslov restavracije,
- trenutna razdalja do restavracije (v primeru, da je uporabnikova lokacija znana).

4.2.2 Ocene restavracije in metapodatki

Na tej zaslonski maski so poleg osnovnih podatkov o restavraciji vidne še naslednje povprečne ocene restavracije glede na določen kriterij ter metapodatki o restavraciji:

- povprečna ocena hrane v restavraciji,

- povprečna ocena primernosti cene hrane v restavraciji,
- povprečna ocena hitrosti postrežbe osebja restavracije,
- povprečna ocena prijaznosti osebja restavracije oziroma vzdušja v restavraciji,
- skupna povprečna ocena po vseh zgoraj naštetih kriterijih,
- število ogledov restavracije preko aplikacije,
- število oddanih ocen restavracije preko aplikacije.

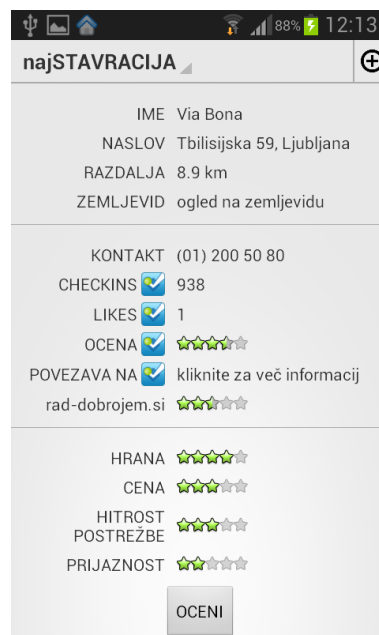
4.2.3 Podatki o restavraciji iz zunanjih virov

Na tej zaslonski maski so prav tako vidni tudi podatki o restavraciji iz zunanjih virov oziroma zbirk podatkov.

Foursquare

V primeru, da je restavracija na voljo v spletni storitvi Foursquare, se iz tega vira prikažejo spodaj našteti podatki:

- telefonska številka restavracije (ob dotiku na polje se odpre klicni zaslon Android naprave z vpisano številko),
- spletni naslov restavracije (ob dotiku na polje se ta naslov odpre v privzetem brskalniku Android naprave),
- povezava na spletni meni restavracije (ob dotiku na polje se ta naslov odpre v privzetem brskalniku Android naprave),
- število prijav preko spletne storitve Foursquare,
- število všečkov na spletni storitvi Foursquare,
- povezava na stran restavracije znotraj spletne storitve Foursquare (ob dotiku na polje se ta spletna stran odpre v privzetem brskalniku Android naprave),



Slika 4.9: Podrobnejši podatki o restavraciji - ocena spletne strani www.rad-dobrojem.si

- povprečna ocena restavracije med uporabniki spletne storitve Foursquare.

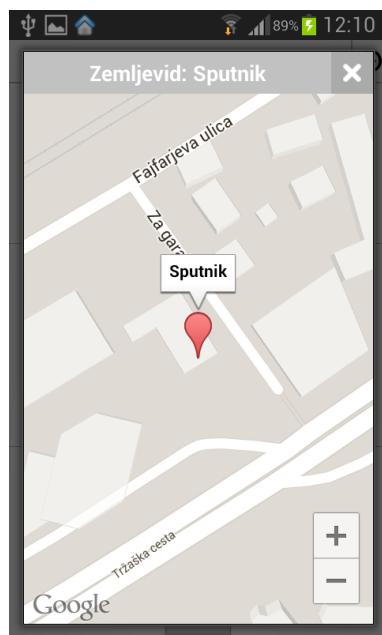
Prikažejo se samo tisti podatki, ki so dodani v storitev Foursquare.

Rad-dobrojem.si

V primeru, da je bila restavracija dodana iz spletne strani <http://www.rad-dobrojem.si>, se na tej zaslonski maski prikaže tudi ocena ocenjevalca iz te strani. Primer zaslonske maske z oceno je viden na sliki 4.9. Dodajanje restavracij iz omenjene spletne strani je podrobno opisano v razdelku 5.2.6.

4.2.4 Prikaz restavracije na zemljevidu

S pritiskom na povezavo 'ogled na zemljevidu' se odpre modalno okno z zemljevidom, vidno na sliki 4.10. Na tem zemljevidu je z značko označena

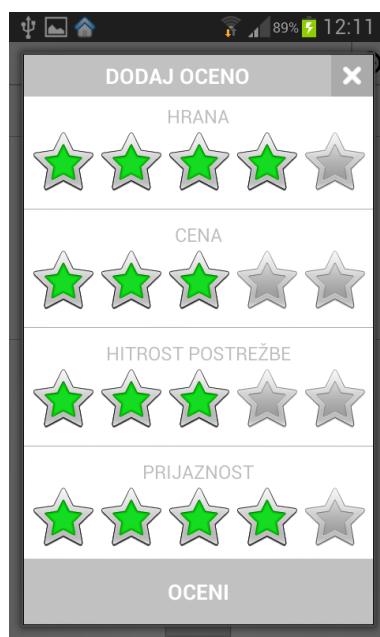


Slika 4.10: Prikaz restavracije na zemljevidu

izbrana restavracija. Zemljevid je nastavljen tako, da je značka na sredini zemljevida, povečava pa je nastavljena tako, da se vidi le neposredno okolico restavracije.

4.2.5 Ocenjevanje restavracije

S pritiskom na gumb 'Oceni' se prikaže modalno okno (vidno na sliki 4.11), kjer lahko uporabniki dodajo oceno restavraciji. V tem oknu se prikažejo štiri vnosna polja za ocenjevanje po posameznih kriterijih. Uporabnik določi oceno za hrano, oceno za primernost cene, oceno za hitrost postrežbe in oceno za prijaznost osebja v restavraciji. Uporabnik mora pri ocenjevanju podati ocene za vse štiri kriterije. Izbrane ocene se shranijo v bazi strežniškega sistema.



Slika 4.11: Ocenjevanje restavracije

4.3 Dodajanje restavracije

Zaslonska maska za dodajanje nove restavracije je zaradi praktičnosti dostopna iz celotne aplikacije. S pritiskom na gumb ‘+’ v zgornjem desnem kotu aplikacije se odpre zaslonska maska, ki omogoča uporabnikom dodajanje nove restavracije. Zaslonska maska je vidna na sliki 4.12.

Zaslonsko masko sestavljajo: 2 vnosni polji (za vpis imena in naslova restavracije), potrditveni gumb in seznam že obstoječih bližnjih restavracij.

V primeru, da je uporabnikova trenutna lokacija znana, se v vnosno polje za vpis naslova restavracije samodejno vpiše uporabniku najbližji naslov. Naslov se pridobi iz GPS koordinat s pomočjo obratnega geokodiranja, podrobeneje opisanega v razdelku 5.2.3. Izpolnjeni naslov lahko uporabnik po potrebi tudi spremeni.

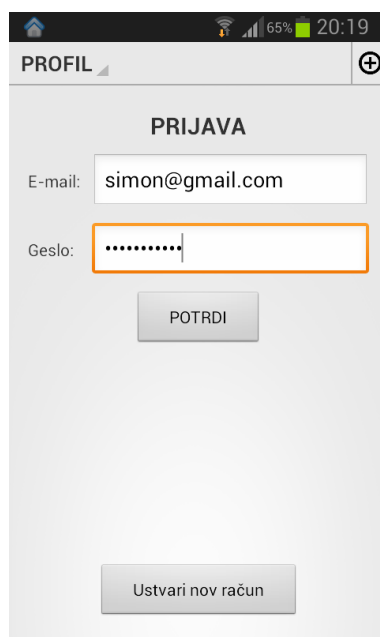
V seznamu bližnjih restavracij te zaslonske maske se prikažejo restavracije, ki so uporabniku trenutno najbližje. Seznam je urejen v naraščajočem vrstnem redu glede na razdaljo do restavracije. Filtrirati ga je možno z

The screenshot shows a mobile application interface with a status bar at the top displaying icons for home, signal, and battery (64%), along with the time 20:23. The app's title bar reads 'najSTAVRACIJA' with a plus icon on the right. Below the title bar, there are two input fields: 'IME' (Name) containing 'Cantina Mexicana' and 'NASLOV' (Address) containing 'Wolfova ulica 4, Ljubljana, SI'. The 'IME' field is highlighted with an orange border. Below these fields is a grey button labeled 'DODAJ'. At the bottom, there is a list area with headers 'IME' and 'RAZDALJA' (Distance), but it is currently empty.

Slika 4.12: Dodajanje nove restavracije

vpisom imena restavracije v vnosno polje. Namen seznama je prikazati oziroma predlagati restavracije, ki so podobne novi restavraciji, da bi se čimbolj izognili dodajanju restavracij, ki v bazi podatkov že obstajajo. S tem se izognemo neželenemu podvajanju podatkov, ki bi privedlo do večje fragmentacije in slabše preglednosti restavracij. S klikom na že obstoječo restavracijo v tem seznamu restavracij se prikaže pogled s podrobnimi podatki o restavraciji, opisan v podpoglavju 4.2.

V primeru, da nobena izmed predlaganih restavracij ne ustreza restavraciji, katero želi uporabnik dodati, lahko uporabnik doda novo restavracijo. To stori tako, da v vnosno polje vnese polno ime restavracije, po potrebi popravi predlagani naslov in s klikom na potrditveni gumb restavracijo shrani v sistem.



Slika 4.13: Prijava v uporabniški profil

4.4 Prijava in registracija

S prijavo v uporabniški profil je uporabniku na voljo več funkcionalnosti aplikacije kot brez prijave. Dodatne funkcionalnosti, ki so na voljo samo prijavljenim uporabnikom so:

- pregled uporabniškega profila,
- iskanje, dodajanje in pregledovanje prijateljev,
- ocenjevanje restavracij,
- pregled ocenjenih restavracij,
- razvrščanje restavracij glede na ocene prijateljev.

The screenshot shows a mobile application interface for profile registration. At the top, there's a status bar with icons for USB, home, Wi-Fi, signal, 69% battery, and time 11:04. Below is a header with 'PROFIL' and a plus icon. Two progress indicators, '1. KORAK' and '2. KORAK', are shown, with '1. KORAK' highlighted by a red bar. The main title is 'Registracija novega računa'. The form contains three input fields: 'E-mail' with the value 'simon@gmail.com', 'Geslo' (password) with masked characters, and 'Potrdi geslo' (confirm password) also with masked characters. An orange border highlights the 'Potrdi geslo' field. At the bottom is a grey button labeled 'NAPREJ'.

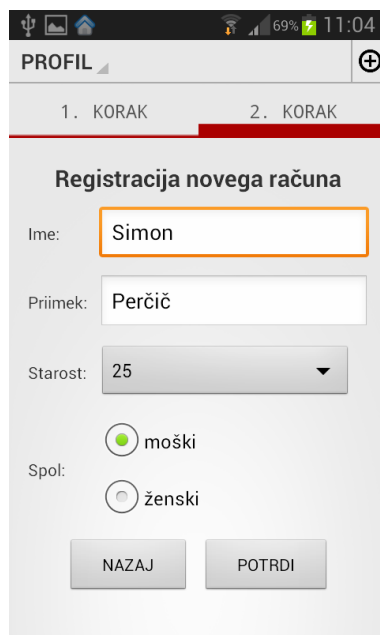
Slika 4.14: Registracija novega uporabniškega profila - 1. korak

4.4.1 Prijava v uporabniški profil

Uporabnik se lahko v uporabniški profil prijavi z uporabo temu namenjene zaslonske maske, ki je vidna na sliki 4.13. Ta vsebuje dve vnosni polji, kamor uporabnik vpiše elektronski naslov in svoje geslo. Uporabnik se nato prijavi s pritiskom na gumb 'potrdi'.

4.4.2 Registracija novega uporabniškega profila

V primeru, da uporabnik še nima ustvarjenega uporabniškega profila, lahko tega ustvari z uporabo zaslonske maske za registracijo novega uporabniškega profila, vidno na slikah 4.14 in 4.15. Ta zaslonska maska je sestavljena iz dveh korakov: v prvem uporabnik vpiše svoje identifikacijske podatke, v drugem pa svoje osebne podatke. V prvem koraku uporabnik tako vpiše svoj elektronski naslov, geslo in potrditev gesla. V drugem koraku pa uporabnik vpiše svoje ime, priimek, starost in spol. Registracijo uporabnik potrdi s pritiskom na gumb 'potrdi'.



The screenshot shows a mobile application interface for account registration. At the top, there's a status bar with icons for USB, home, signal, and battery (69%), and the time 11:04. Below that is a header with 'PROFIL' and a plus icon. A progress bar indicates '1. KORAK' and '2. KORAK', with '2. KORAK' being the active step. The main title is 'Registracija novega računa'. The form contains the following fields: 'Ime:' with the value 'Simon', 'Priimek:' with the value 'Perčič', 'Starost:' with a dropdown menu showing '25', and 'Spol:' with two radio buttons, 'moški' (selected) and 'ženski'. At the bottom, there are two buttons: 'NAZAJ' and 'POTRDI'.

Slika 4.15: Registracija novega uporabniškega profila - 2. korak

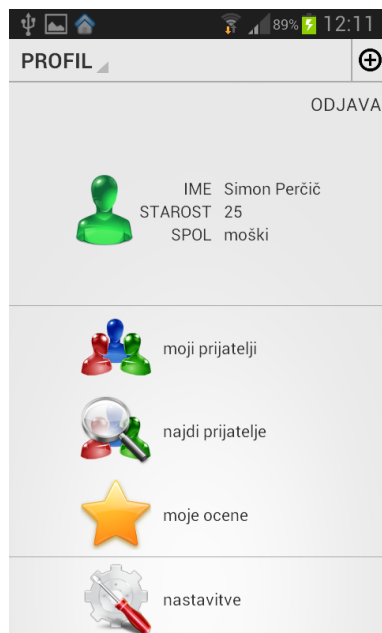
4.5 Uporabniški profil

Aplikacija omogoča tudi pregled in urejanje uporabniškega profila ter nastavitvev, poleg tega lahko uporabnik tudi dodaja prijatelje.

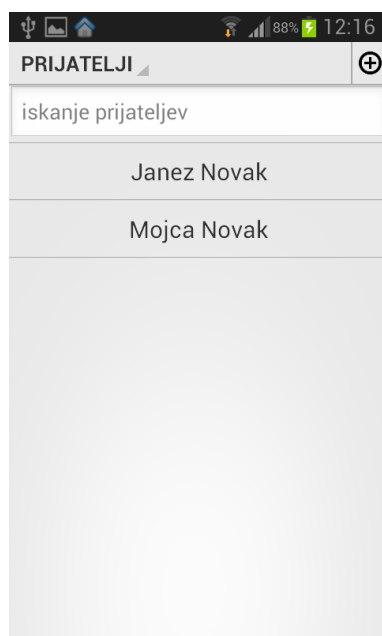
Na zaslonski maski vidni na sliki 4.16 lahko uporabnik vidi svoje podatke (ime, starost in spol), se odjavi iz aplikacije ter dostopa do zaslonskih mask za upravljanje s prijatelji, za pregled ocenjenih restavracij in za spreminjanje nastavitvev uporabniškega profila.

4.5.1 Pregled lastnih prijateljev

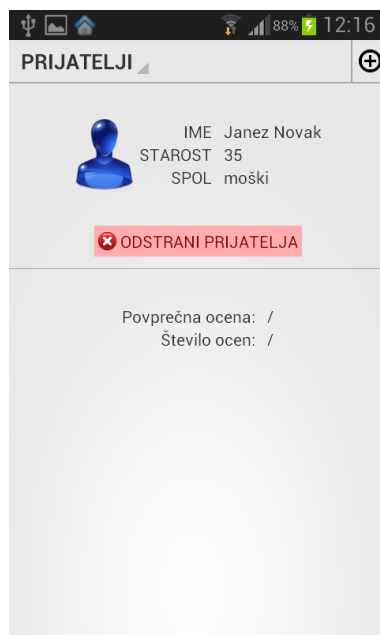
S pritiskom na gumb z napisom 'moji prijatelji' se odpre zaslonska maska s seznamom uporabnikov, ki jih ima uporabnik dodane kot prijatelje. Ta zaslonska maska je vidna na sliki 4.17. Seznam je mogoče filtrirati z vpisom imena ali priimka v vnosno polje na vrhu maske. S pritiskom na vrstico v seznamu se odpre pogled prijateljevega profila, opisan v razdelku 4.5.2.



Slika 4.16: Uporabniški profil



Slika 4.17: Pregled lastnih prijateljev



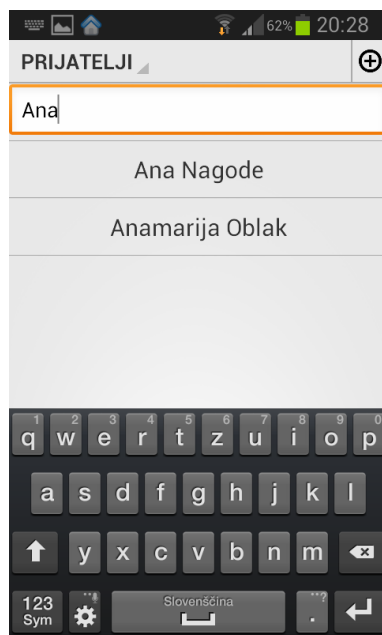
Slika 4.18: Pogled prijateljevega profila

4.5.2 Pogled prijateljevega profila

Ta zaslonska maska prikazuje podatke o uporabnikovem prijatelju (ime, priimek, starost in spol) ter podatka o povprečni oceni pri ocenjevanju restavracij in o številu ocenjenih restavracij. Na zaslonski maski je na voljo tudi gumb z napisom 'odstrani prijatelja', s katerim lahko iz seznama prijateljev odstranimo uporabnika. Zaslonska maska je vidna na sliki 4.18.

4.5.3 Iskanje prijateljev

S pritiskom na gumb z napisom 'najdi prijatelje' se prikaže zaslonska maska (vidna na sliki 4.19), ki omogoča iskanje drugih uporabnikov. Za iskanje po uporabnikih je potrebno vpisati iskalni niz v vnosno polje, v spodnjem seznamu pa se nato prikažejo uporabniki, ki v svojem imenu ali priimku vsebujejo vpisani niz. Z izbiro posameznega rezultata iskanja se prikaže pogled profila drugega uporabnika, opisan v razdelku 4.5.4.



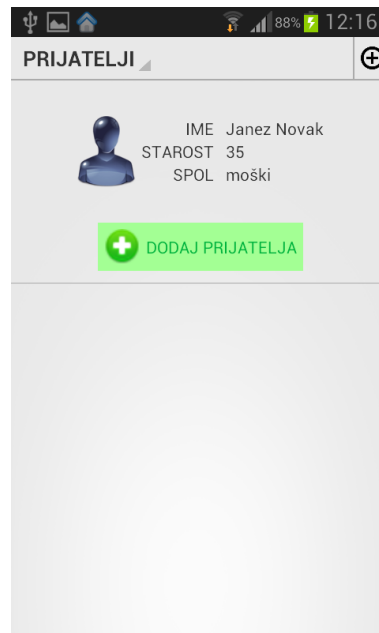
Slika 4.19: Iskanje prijateljev

4.5.4 Pogled profila drugega uporabnika

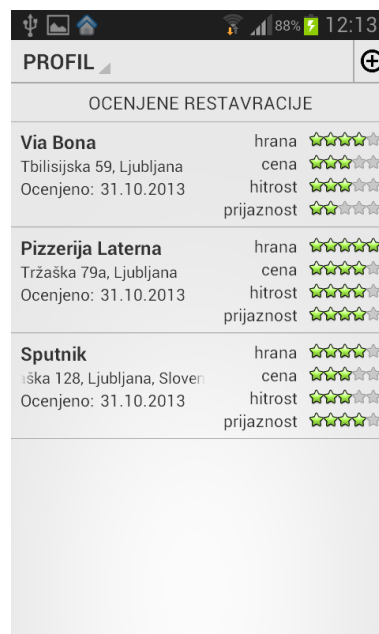
Ta zaslonska maska je zelo podobna zaslonski maski, opisani v razdelku 4.5.2. Za razliko od prejšnje zaslonske maske, na tej zaslonski maski nista vidna podatka o povprečni oceni pri ocenjevanju restavracij in o številu ocenjenih restavracij. Ta dva podatka sta vidna šele ko uporabnika dodamo kot prijatelja. To lahko storimo s pritiskom na gumb z napisom 'dodaj prijatelja'. Zaslonska maska je vidna na sliki 4.20.

4.5.5 Pregled ocenjenih restavracij

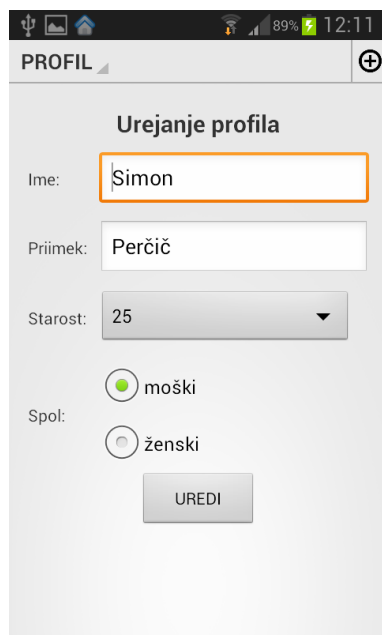
Uporabnik lahko s pritiskom na gumb z napisom 'moje ocene' odpre zaslonsko masko, ki prikazuje seznam lastnih ocenjenih restavracij (vidno na sliki 4.21). V seznamu so prikazani: ime restavracije, njen naslov, datum dodane ocene in ocene po posameznih kriterijih (ocena hrane, ocena primernosti cene, ocena hitrosti postrežbe in ocena prijaznosti osebja restavracije). Z izbiro posamezne vrstice seznama se prikaže pogled s podrobnimi podatki



Slika 4.20: Pogled profila drugega uporabnika



Slika 4.21: Pregled ocenjenih restavracij



Slika 4.22: Urejanje uporabniškega profila

o restavraciji, opisan v podpoglavju 4.2.

4.5.6 Urejanje uporabniškega profila

Uporabnik lahko z uporabo te zaslonske maske spreminja osebne podatke, ki jih je vpisal ob registraciji uporabniškega profila. Uporabnik lahko tako ureja svoje ime in priimek, spremeni vpisano starost in po želji tudi zamenja vpisan spol. S pritiskom na gumb 'uredi' se vpisani podatki preverijo in nato shranijo. Zaslonska maska je dostopna z izbiro gumba 'nastavitve' v uporabniškem profilu in je vidna na sliki 4.22.

4.5.7 Odjava iz profila

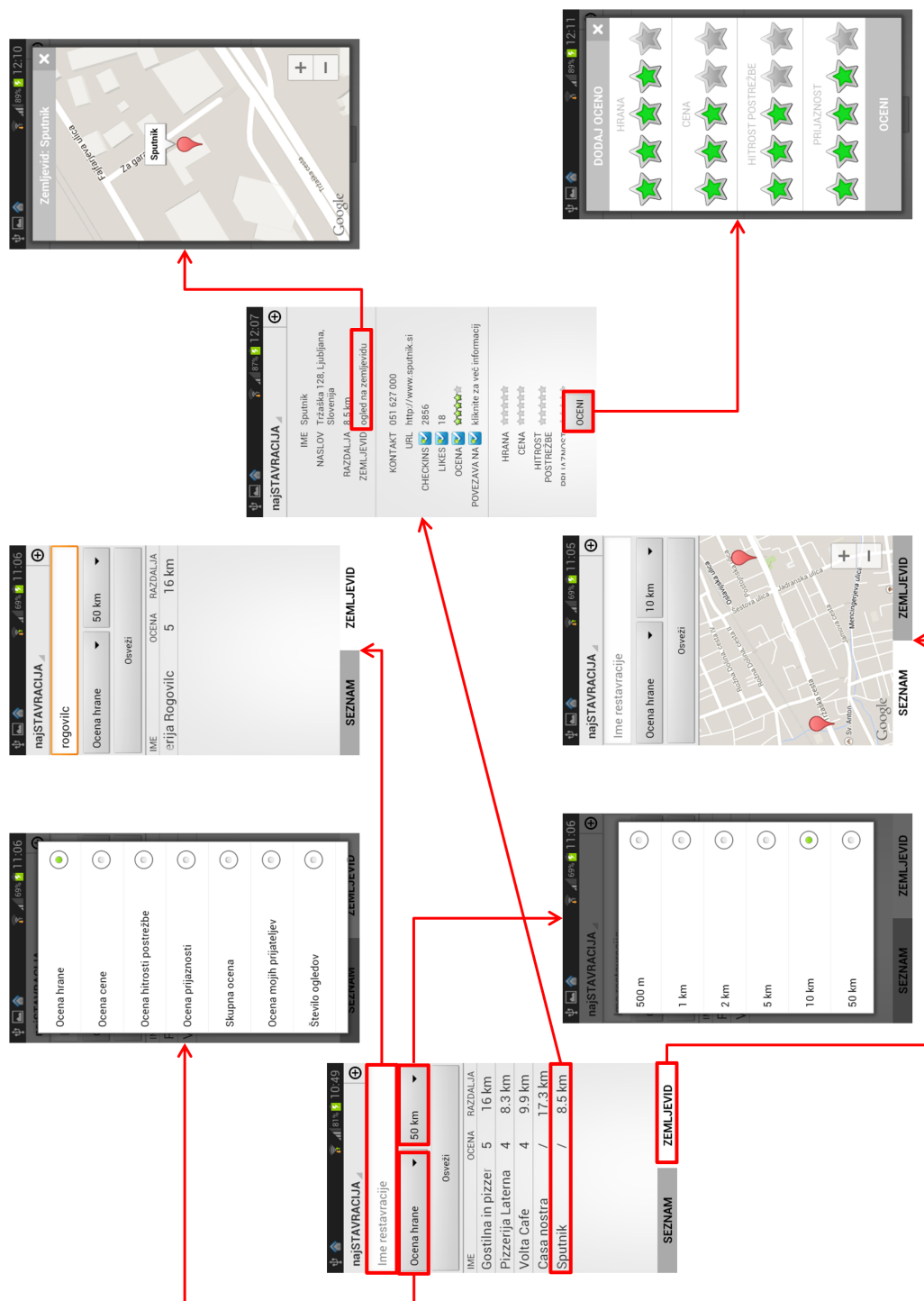
Uporabnik se lahko iz svojega profila tudi odjavi in s tem iz mobilne naprave izbriše shranjene avtentikacijske podatke. Za uporabo funkcij, ki zahtevajo prijavljenega uporabnika, se bo moral tako ponovno prijaviti ali ustvariti nov

uporabniški profil. Uporabnik se lahko odjavi s pritiskom na napis ‘odjava’ v uporabniškem profilu. Po uspešni odjavi se prikaže zaslonska maska za prijavo v profil, opisana v razdelku 4.4.1.

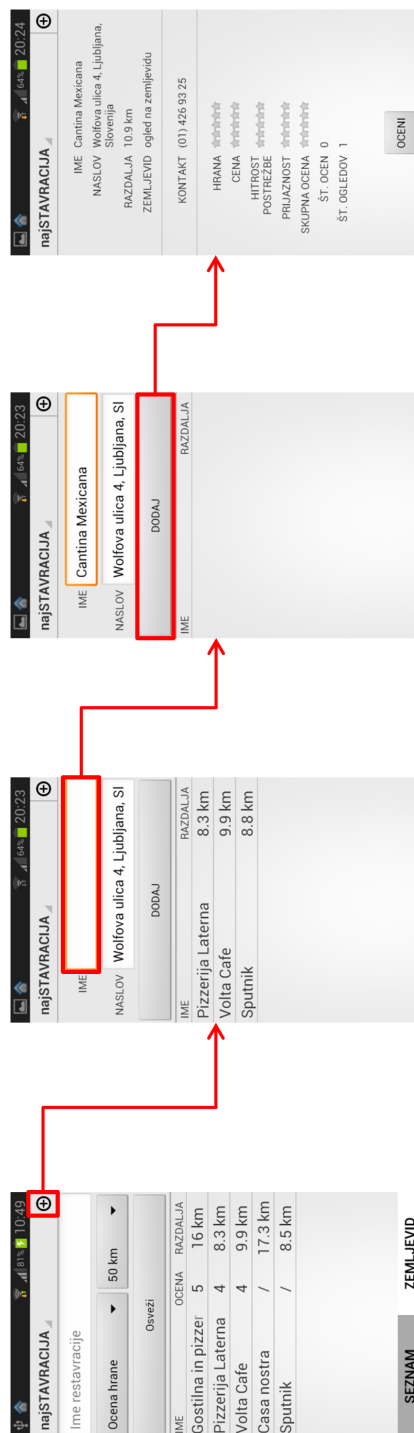
4.6 Sheme prehodov med zaslonskimi maskami

V nadaljevanju bodo predstavljene sheme prehodov med različnimi zaslonskimi maskami Android aplikacije. Na shemah so z rdečimi pravokotniki označene nekatere izmed akcij, ki jih lahko sprožijo uporabniki. Rdeče puščice prikazujejo, kaj so posledice teh akcij (torej na katero zaslonsko masko nas te akcije pripeljejo). V shemah bodo predstavljene le pomembnejše zaslonske maske in prehodi med njimi.

Na shemi 4.23 so vidni prehodi med zaslonskimi maskami pregleda restavracij, na shemi 4.24 so vidni prehodi med zaslonskimi maskami dodajanja restavracij, shema 4.25 prikazuje prehode med zaslonskimi maskami uporabniškega profila, shema 4.26 pa prikazuje prehode med zaslonskimi maskami iskanja in dodajanja prijateljev.



Slika 4.23: Shema prehodov med različnimi zaslonскими maskami pregleda restavracij



Slika 4.24: Shema prehodov med različnimi zaslonскими maskami dodajanja restavracij



Slika 4.26: Shema prehodov med različnimi zaslonskimi maskami iskanja in dodajanja prijateljev

Poglavje 5

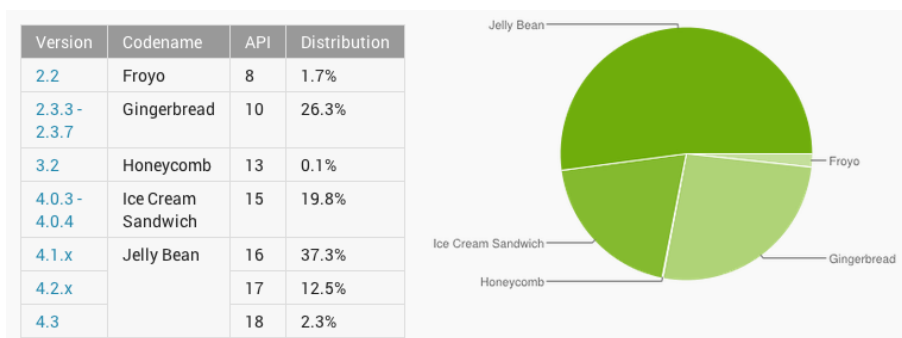
Implementacija rešitve

V prejšnjem poglavju smo podali predlog naše rešitve, v tem poglavju pa bomo predstavili še določene tehnične podrobnosti implementacije.

5.1 Android aplikacija

Android mobilno aplikacijo smo razvili tako, da podpira Googlov operacijski sistem Android od verzije 2.3.3 naprej (oznaka Gingerbread, API nivo v10). Seznam API nivojev je na voljo na [1].

Po Googlovih uradnih podatkih [2] je delež aktivnih Android naprav, ki imajo verzijo operacijskega sistema 2.3.3 ali višjo, enak 98.3%. Odločili smo se, da je ta delež dovolj velik, da nam starejših verzij operacijskega sistema ni potrebno podpirati. Distribucija različnih verzij operacijskega sistema je prikazana na sliki 5.1. Delež je izmerjen na podlagi vseh naprav, ki so se povezale na Google Play Store v zadnjih sedmih dneh. Google Play Store podpira samo Android operacijski sistem od verzije 2.2 naprej, sodeč po Googlovih podatkih pa je delež naprav s starejšimi verzijami operacijskega sistema manjši od 1% (ta delež je izmerjen na podlagi vseh naprav, ki so se povezale na njihove strežnike).



Slika 5.1: Distribucija različnih verzij Android operacijskega sistema

5.1.1 Struktura mobilne aplikacije

Pri zasnovi mobilne aplikacije smo se odločili, da se bodo vse zaslonske maske obnašale kot medsebojno neodvisni moduli. Android aktivnost (ang. activity) [3] predstavlja posamezno aktivnost, ki jo lahko izvede uporabnik. Zaradi tega je aktivnost tisti gradnik aplikacije, ki služi kot okvir (ang. container) za vse ostale gradnike uporabniškega vmesnika. V našem primeru je v aplikaciji samo ena Android aktivnost, ki je zadolžena za prikazovanje vseh ostalih zaslonskih pogledov.

Ostali zaslonski pogledi so implementirani modularno, torej delujejo neodvisno eden od drugega. Vsak zaslonski pogled skrbi za prikaz in uporabniško interakcijo z določeno funkcionalnostjo uporabniškega vmesnika. Vsak zaslonski pogled je implementacija Android fragmenta [4], torej neodvisnega dela uporabniškega vmesnika. Vsak fragment ima tako kot aktivnost svoj življenjski cikel, ki je v določeni meri odvisen od življenjskega cikla aktivnosti. V nadaljevanju pomeni fragment posamezno zaslonsko masko.

Glavna aktivnost v naši aplikaciji skrbi za:

- prikazovanje trenutno aktivnega fragmenta,
- navigacijo med različnimi fragmenti,
- menjavanje prikazanega fragmenta glede na zahteve iz ostalih fragmentov,

- pridobivanje uporabnikove lokacije,
- sprožanje omrežnih zahtev na strežniški sistem.

Na shemah 5.2 in 5.3 je prikazan okvirni potek menjave aktivnega fragmenta. Opis primera: uporabnik na seznamu restavracij izbere neko restavracijo, da bi videl njene podrobne podatke.

Potek menjave aktivnega fragmenta:

1. trenutno aktivni fragment (A) glavni aktivnosti aplikacije sporoči zahtevo po menjavi aktivnega fragmenta z drugim fragmentom (B);
2. glavna aktivnost aplikacije kličočemu fragmentu (A) sporoči zahtevo po shranjevanju lastnih podatkov, saj bo instanca kličočega fragmenta (A) odstranjena;
3. glavna aktivnost aplikacije kreira novo instanco drugega fragmenta (B) in ga označi za trenutno aktivnega.

Na shemi 5.4 je prikazan potek ustvarjanja in odstranjevanja modalnega fragmenta. Modalni fragment je tisti fragment, ki ne prekrije celotnega zaslona, ampak je v vlogi dialoga. Opis primera: uporabnik na podrobnem pogledu restavracije izbere akcijo za ocenjevanje restavracije.

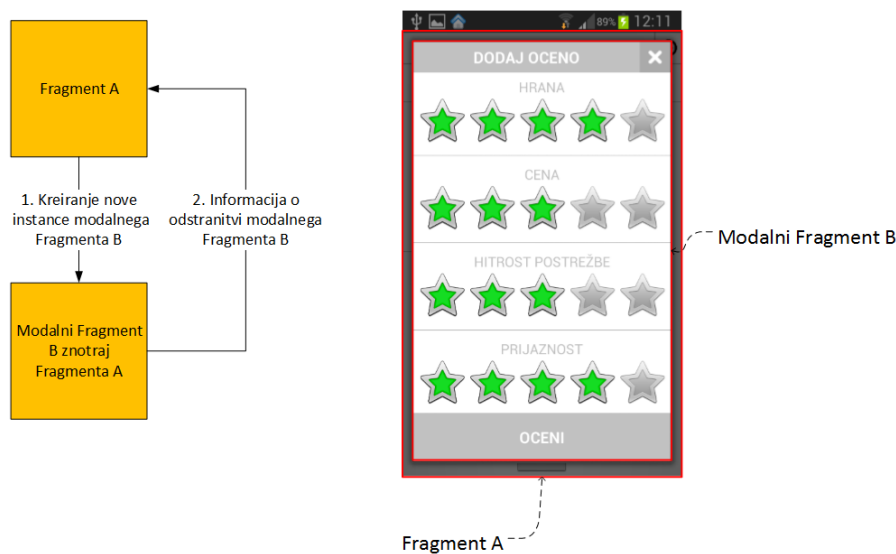
Potek:

1. fragment (A) naredi novo instanco modalnega fragmenta (B);
2. po koncu uporabe modalni fragment (B) fragmentu (A) sporoči, da bo sam odstranjen (s tem fragment (A) na primer izve kdaj lahko osveži lastne podatke).

5.1.2 Pridobivanje uporabnikove lokacije

Informacija o trenutni lokaciji uporabnika je zelo pomembna za boljše delovanje celotnega sistema. Zaradi tega smo implementirali sistem, ki skrbi





Slika 5.4: Potek ustvarjanja in odstranjevanja modalnega fragmenta

za to, da je v vsakem trenutku na voljo čim boljša informacija o uporabnikovi trenutni lokaciji. Za pridobivanje lokacij imamo na voljo dva vira: GPS senzor in Androidov sistem za pridobivanje lokacije iz omrežij (Android Network Location Provider). Slednji določa lokacijo glede na moč signalov WIFI omrežij in baznih postaj mobilnega omrežja [5].

V naši aplikaciji uporabljamo oba vira, saj sta si komplementarna. Implementirani sistem za določanje najboljšega približka trenutne uporabnikove lokacije potrebujemo zato, ker sta si oba vira uporabnikove lokacije med seboj različna. GPS senzor vrača natančnejše rezultate, a ne deluje v zaprtih prostorih, poleg tega pa za začetno pridobivanje lokacije potrebuje nekaj časa. Po drugi strani je Androidov sistem za pridobivanje lokacije iz omrežij hitrejši, deluje tudi v zaprtih prostorih, a je veliko manj natančen. Razlaga strategij za pridobivanje uporabnikove lokacije je povzeta po [5].

Naš sistem je načrtovan tako, da interno hrani trenutno najboljši približek lokacije. Ko iz enega izmed obeh virov dobi novo lokacijo, to primerja s trenutno najboljšo lokacijo in le-to posodobi, če se izkaže da je nova lokacija boljše. Pravila, po katerih se to zgodi, so razvidna iz diagrama poteka na

sliki 5.5. Starost lokacije je pomembna zato, ker se je lahko uporabnik od takrat, ko smo dobili prejšnjo lokacijo, že premaknil. Zaradi tega imajo bistveno novejša lokacije prednost pred starejšimi, četudi so manj natančne. Ocena natančnosti (povzeto po [6]) vsake lokacije je podana v metrih in je definirana sledeče: če iz podane koordinate narišemo krog, ki ima radij enak podani natančnosti v metrih, lahko s približno 68% verjetnostjo trdimo, da je resnična lokacija v tem radiju. Ta izračun predpostavlja, da so napake pri pridobivanju lokacije naključne z normalno porazdelitvijo. Verjetnost 68% ustreza enemu standardnemu odklonu (povzeto po [7]).

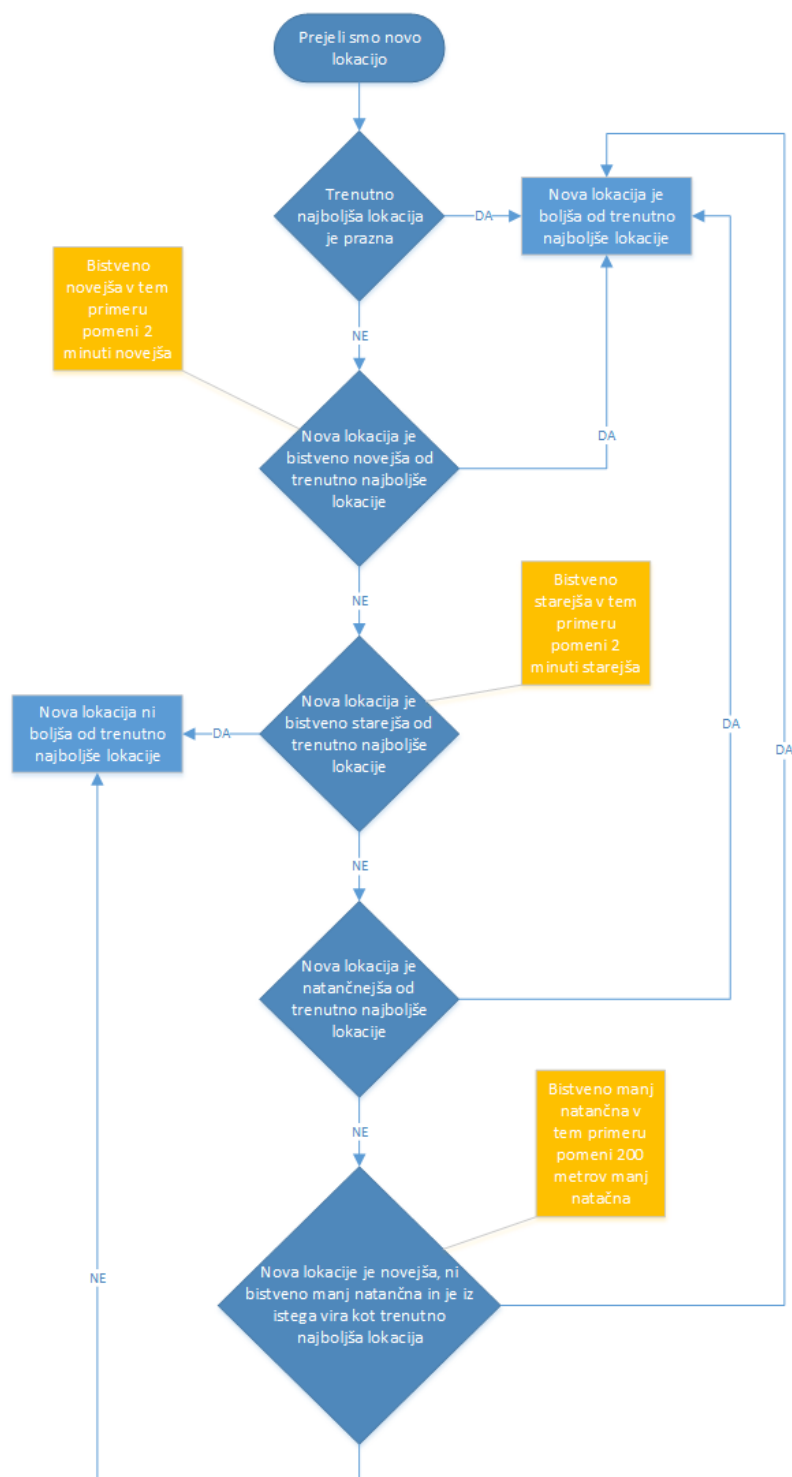
5.1.3 Ogrodje za komunikacijo s strežnikom

Komunikacija s strežnikom je kritičen del Android aplikacije, saj je brez povezave s strežnikom aplikacija neuporabna. Komunikacija poteka preko AJAX klicev med Android aplikacijo in strežniškim sistemom. Podatki se prenašajo v obliki JSON objektov.

Za potrebe aplikacije smo implementirali lastno ogrodje za komunikacijo s strežniškim delom sistema. Ogrodje je zasnovano in implementirano tako, da je njegova uporaba čim enostavnejša.

Ogrodje skrbi za naslednje funkcionalnosti:

- dodajanje uporabnikove lokacije v strežni zahtevek,
- dodajanje avtentikacijskih podatkov uporabnika v strežni zahtevek (samo če je uporabnik prijavljen v aplikacijo),
- deserializacija prejetega JSON rezultata v Java objekt,
- serializacija poslanih podatkov v JSON objekt,
- preverjanje prejetega rezultata s strežnika in prikaz morebitne prejete napake,
- odjavljanje uporabnika iz aplikacije v primeru, da uporabniški žeton ni veljaven,



Slika 5.5: Diagram poteka za preverjanje primernosti nove uporabnikove lokacije

```

AddRestaurantRequest request = new AddRestaurantRequest();
request.Name = restaurantName;
request.Address = restaurantAddress;

getMainActivity().doAjaxRequest(
    ApiUrlConsts.ADD_RESTAURANT,
    request,
    new RatingsAjaxCallback<AddRestaurantResponse>(AddRestaurantFragment.this, AddRestaurantResponse.class) {

        @Override
        public void activityCallback(AddRestaurantResponse response) {
            FragmentHelper.showRestaurantDetails(getMainActivity(), response.Id);
        }
    });

```

Slika 5.6: Primer kode za klic na strežnik

- prikaz in skrivanje dialoga za nalaganje podatkov.

Ogrodje za asinhron klic na strežnik uporablja knjižnico AQuery, opisano v razdelku 3.1.2. Klici na strežnik morajo biti asinhroni in se izvajati na niti (ang. thread) iz ozadja, da uporabniški vmesnik ostane odziven tudi v primeru, ko klic strežnika traja dlje časa.

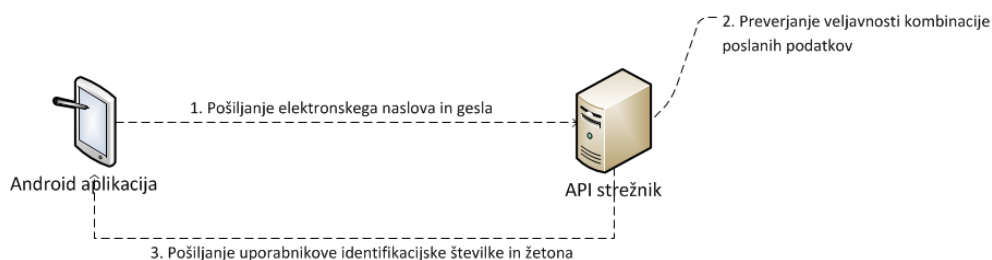
Na sliki 5.6 je prikazana koda za primer strežniškega klica dodajanja nove restavracije.

5.1.4 Avtentikacija

Uporabniki se lahko v aplikaciji prijavijo in s tem uporabljajo dodatne funkcionalnosti. Prijava uporabnika je sestavljena iz identifikacije in avtentikacije uporabnika.

Identifikacija uporabnika pomeni, da lahko določimo za katerega uporabnika gre. Avtentikacija pa pomeni, da preverimo, ali se uporabnik prijavlja s pravilnimi podatki.

Ob prijavi uporabnika se na strežnik pošlje vpisan elektronski naslov in geslo. Strežnik nato preveri veljavnost kombinacije elektronskega naslova in podanega gesla. V primeru, da je kombinacija veljavna, strežnik iz teh dveh podatkov izračuna uporabniški žeton, si ga shrani in ga skupaj z uporabniško identifikacijsko številko v rezultatu klica vrne Android aplikaciji. Android aplikacija si ta uporabniški žeton in uporabnikovo identifikacijsko



Slika 5.7: Shema avtentikacije uporabnika

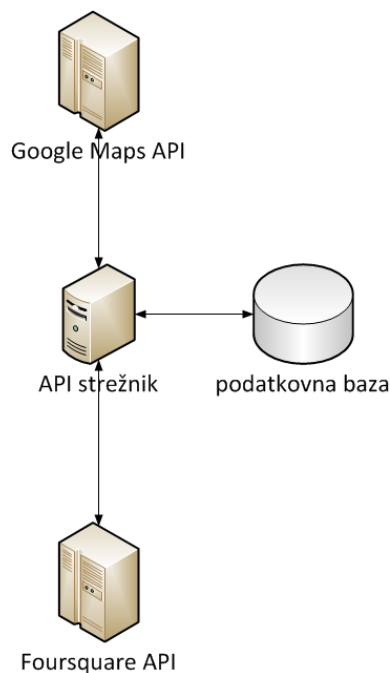
številko shrani v trajno shrambo na Android napravo. Osnovni princip trajne shrambe deluje na podlagi shranjevanja parov ključ/vrednost in je opisan na [12]. Shema avtentikacije je prikazana na sliki 5.7.

Pridobljena uporabnikova identifikacijska številka in žeton se nato pri vsakem zahtevku pošiljata na strežnik. Strežnik lahko tako preveri veljavnost uporabnikovih avtentikacijskih podatkov in v rezultatu klika vrne status njihove veljavnosti. V primeru, da avtentikacijski podatki niso veljavni, strežnik v rezultatu vrne napako. Aplikacija v takem primeru samodejno odjavi prijavljenega uporabnika.

5.2 Strežniški sistem

Android aplikacija je za svoje delovanje odvisna od strežnika, ki skrbi za hranjenje podatkov, poslovno logiko in pridobivanje podatkov iz zunanjih storitev. Strežniški sistem preko APIja nudi svoje podatke Android aplikaciji. API nudi metode za avtentikacijo uporabnika, registracijo in spreminjanje podatkov o uporabniku, pridobivanje podatkov o restavracijah, izračun razdalje med uporabnikom in restavracijami, dodajanje novih restavracij, ocenjevanje restavracij, pridobivanje podatkov o uporabnikih, dodajanje uporabnikov za prijatelje itd.

Strežniški sistem preko javno dostopnih APIjev pridobiva podatke iz zunanjih storitev, v našem primeru sta to Foursquare API in Google Maps API. Njuna uporaba bo podrobneje opisana v nadaljevanju. Na sliki 5.8 je vidna



Slika 5.8: Shema komunikacije med strežnikom in zunanji storitvi

shema komunikacije med našim strežnikom in zunanji storitvi.

5.2.1 Windows Azure

Zaradi želje po prihodnji skalabilnosti in stabilnosti delovanja je strežniški sistem zasnovan za delovanje na Microsoftovi oblaki platformi Windows Azure [38]. Za implementacijo smo uporabili .NET Framework 4 in programski jezik C#.

Windows Azure je Microsoftova oblaka platforma in infrastruktura. Omogoča poganjanje in upravljanje s storitvami v oblaku (ang. cloud service), spletnimi stranmi (ang. website) in virtualnimi sistemi (ang. virtual machine). Poleg tega omogoča tudi shranjevanje podatkov, uporabo podatkovnih baz in nekatere druge storitve.

Za potrebe našega strežniškega sistema uporabljamo Azure-jeve oblake storitve, ki spadajo v skupino platforme kot storitve (ang. Platform as a

Service, PaaS [41]).

Platforma Windows Azure nam tako omogoča, da razvijemo storitev in jo naložimo, nato pa platforma sama skrbi za poganjanje, uravnoteženje obremenitve (ang. load balancing), strojno infrastrukturo, posodobitve operacijskega sistema, strojne napake, izpade elektrike, omrežno povezljivost itd. Tako se lahko popolnoma osredotočimo na razvoj aplikacije.

Windows Azure podpira več instanc oblačnih storitev, katerim lahko tudi nastavljamo zmogljivost. Spletni zahtevki (in posledično obremenitve) se porazdelijo med več instanc s pomočjo sistema za uravnoteženje obremenitve. Tako je poskrbljeno za skalabilnost, saj lahko enostavno in učinkovito spreminjamo zmogljivost naše storitve, tako da bodisi povečamo število instanc naše storitve, bodisi povečamo zmogljivost obstoječim instancam. V primeru, da uporabljamo vsaj dve instanci oblačne storitve, nam platforma zagotavlja 99.95% dosegljivost naše storitve [39].

Za naš strežniški sistem uporabljamo 2 instanci oblačne storitve zmogljivostnega razreda 'XS', kjer imamo na voljo 1 GHz CPE, 768 MB delovnega pomnilnika in 20 GB shrambe na trdem disku strežnika [40].

Platforma oblačnih storitev nam omogoča tudi dva okolja delovanja, in sicer produkcijsko (ang. production) in uprizoritveno (ang. staging). Uprizoritveno je namenjeno testiranju, ko pa smo zadovoljni z delovanjem, lahko uprizoritveno okolje spremenimo v produkcijsko. Produkcijsko okolje je tisto, ki je dostopno končnim uporabnikom naše storitve.

Opis oblačnih storitev Microsoftove oblačne platforme Windows Azure je dostopen na [39].

5.2.2 Shema podatkovne baze

V nadaljevanju bo predstavljena shema podatkovne baze in opis posameznih entitet.

Kot je razvidno iz sheme na sliki 5.9, imamo v naši podatkovni bazi več različnih entitet, ki so: uporabnik, restavracija, lokacija, ocena, Foursquare podatki, lokalizacijski niz in predpomnjena vrednost.



Slika 5.9: Shema podatkovne baze strežniškega sistema

Entiteta **restavracija** predstavlja določeno restavracijo. Vsebuje podatek o imenu, številu ogledov, oceni spletne strani www.rad-dobrojem.si, datumu, ko je bila dodana v sistem, referenco na entiteto uporabnika, ki jo je dodal, referenco na entiteto o Foursquare podatkih in referenco na entiteto o lokaciji.

Entiteta **ocena** vsebuje podatke o posamezni oddani oceni. Ti podatki so: ocena hrane, ocena primernosti cene, ocena hitrosti, ocena prijaznosti, datum oddane ocene, referenca na uporabnika, ki jo je dodal, in referenca na restavracijo, kateri pripada.

Entiteta **lokacija** vsebuje podatke o posamezni lokaciji za neko restavracijo. Vsebuje naslov v tekstovni obliki in geografske koordinate (širina in dolžina).

Entiteta **Foursquare podatki** vsebuje podatke o neki restavraciji, pridobljene iz strani Foursquare [21]. Vsebuje spletni naslov restavracije, Foursquare oceno, telefonsko številko restavracije, število Foursquare prijav (ang. check-in), število Foursquare všečkov (ang. like), spletni naslov do Foursquare-ove predstavitve restavracije in spletni naslov do menija restavracije.

Entiteta **uporabnik** vsebuje podatke o registriranemu uporabniku, kot so: elektronski naslov, ime, priimek, starost, spol, geslo itd. S pomočjo dodatne entitete dobimo tudi informacijo o tem, kateri uporabnik ima dodanega drugega uporabnika kot prijatelja.

Entiteta **lokalizacijski niz** vsebuje podatke o posameznem lokalizacijskem nizu, saj je sistem razvit z mislijo na lokalizacijo v različne jezike. Posamezni lokalizacijski niz je sestavljen iz lokalizacijskega ključa, lokalizirane tekstovne vrednosti in tri črkovne oznake jezika, v katerem je lokalizirana vrednost.

Entiteta **predpomnjena vrednost** predstavlja entiteto, ki je predpomnjena zaradi želje po zmanjšanju števila zahtev na Google Geocoding API, opisanega v razdelku 5.2.3, in Google Distance Matrix API, opisanega v razdelku 5.2.3. Implementirani sistem predpomnjenja je podrobneje opisan v

razdelku 5.2.5. Entiteta vsebuje podatek o ključu, vrednosti, datumu ustvarjanja in tipu predpomnjene vrednosti.

5.2.3 Google Maps API

Naša aplikacija je v veliki meri odvisna od delovanja lokacijskih storitev. V naši aplikaciji lokacijske storitve potrebujemo za:

- pridobivanje geografskih koordinat iz naslova (pri dodajanju nove restavracije v sistem),
- pridobivanje predlogov za naslov restavracije glede na trenutno lokacijo uporabnika (pred dodajanjem nove restavracije),
- pridobivanje razdalje do restavracije (pri pregledovanju restavracij).

Lokacijske storitve nam omogočajo Google Maps API Web Services [13], ki so množica javno dostopnih HTTP spletnih storitev. Poleg ostalih funkcionalnosti te storitve omogočajo pridobivanje geografskega naslova na podlagi koordinat, pridobivanje koordinat iz podanega naslova ter izračun razdalje in časa potovanja med različnimi koordinatami. V nadaljevanju bomo podrobneje opisali uporabljene storitve.

Google Geocoding API

Google Geocoding API [14] omogoča storitvi geokodiranja (ang. geocoding) in obratnega geokodiranja (ang. reverse geocoding).

Geokodiranje (ang. geocoding) je postopek pretvarjanja naslovov iz človeku prijazne tekstualne oblike (primer: 'Tržaška cesta 25, Ljubljana') v geografske koordinate (primer: geografska širina: 46.0448994, geografska dolžina: 14.4892307). Koordinate potrebujemo za prikazovanje točk na zemljevidu, izračun razdalje med dvema lokacijama in ostale operacije. V bazi podatkov je skupaj z vsako restavracijo shranjena tudi njena lokacija v obliki geografskih koordinat in tekstualnega naslova.

```
{
  "results" : [
    {
      "address_components" : [...],
      "formatted_address" : "Tržaška cesta 25, Univerza v Ljubljani,
                             1000 Ljubljana, Slovenija",
      "geometry" : {
        "location" : {
          "lat" : 46.0448994,
          "lng" : 14.4892307
        },
        ...
      },
      ...
    },
    ...
  ],
  "status" : "OK"
}
```

Slika 5.10: Skrajšan rezultat klica za geokodiranje na Google Geocoding API

V našem strežniškem sistemu uporabljamo geokodiranje pri dodajanju nove restavracije v sistem.

Novo restavracijo lahko uporabnik doda preko Android aplikacije, kot je opisano v podpoglavju 4.3. Pri dodajanju uporabnik vpiše naslov restavracije, ki se na strežniški strani s pomočjo Google Geocoding API pretvori v geografske koordinate, ki se nato shranijo v podatkovno bazo.

Nove restavracije se v podatkovno bazo dodajo tudi pri luščenju restavracij iz spletne strani www.rad-dobrojem.si [18], po postopku, opisanem v razdelku 5.2.6. Iz spletne strani se pridobijo naslovi restavracij, ki se nato pretvorijo v geografske koordinate in shranijo v podatkovno bazo.

Primer klica na Google Geocoding API:

```
http://maps.googleapis.com/maps/api/geocode/json?
address=Tr%C5%BEa%C5%A1ka%20cesta%2025%2C%20Ljubljana&
language=sl&sensor=false
```

Skrajšan rezultat klica na Google Geocoding API je viden na sliki 5.10.

Obratno geokodiranje (ang. reverse geocoding) je postopek, pri katerem se geografske koordinate pretvorijo v človeku prijazen tekstualni naslov.

Primer: geografska širina: 46.0448994 in geografska dolžina: 14.4892307 se pretvorita v 'Tržaška cesta 25, Ljubljana'. Postopek obratnega geokodiranja je približek, ki poizkuša geografskim koordinatam določiti geografski opis v okviru določene tolerance. Zaradi tega je za neko lokacijo možnih več rezultatov, saj lahko neka lokacija predstavlja specifičen naslov, regijo, mesto in državo obenem. Google Geocoding API tako vrne več rezultatov, razvrščenih od najbolj do najmanj specifičnega.

Postopek obratnega geokodiranja smo uporabili pri predlaganju naslova nove restavracije glede na uporabnikovo trenutno lokacijo. Ko uporabnik želi dodati novo restavracijo, kot je to opisano v podpoglavju 4.3, mu aplikacija predlaga naslov, ki je najbližji njegovi trenutni lokaciji. Uporabnikova lokacija v obliki koordinat se na strežniški strani z uporabo obratnega geokodiranja Google Geocoding APIja pretvori v naslov, ki se nato prikaže na zaslonski maski Android aplikacije.

Primer klica:

```
http://maps.googleapis.com/maps/api/geocode/json?  
latlng=46.0448994,14.4892307&language=sl&sensor=false
```

Skrajšan rezultat klica za obratno geokodiranje je viden na sliki 5.11.

Iz primera je razvidno, da je Google Geocoding API za podane geografske koordinate vrnil več rezultatov, razvrščenih po njihovi specifičnosti. Tako je med drugim vrnil: 'Tržaška cesta 25, 1000 Ljubljana, Slovenija', 'Vič, Slovenija', '1000 Ljubljana, Slovenija' in 'Slovenija'.

Pri naši rešitvi smo se vedno omejili na prvi (torej najbolj specifični) vrnjeni rezultat.

Google Distance Matrix API

Google Distance Matrix API [16] omogoča izračun razdalje in časa trajanja poti med različnimi geografskimi točkami. Izračunana pot je priporočena pot, kot je prikazana na spletni aplikaciji Google Maps [19]. V naši aplikaciji uporabljamo izračun razdalje med dvema točkama pri prikazu seznama restavracij, opisanega v podpoglavju 4.1, in pri prikazu trenutne razdalje do

```
{
  "results" : [
    {
      ...,
      "formatted_address" : "Tržaška cesta 25, Univerza v Ljubljani,
                           1000 Ljubljana, Slovenija",
      "geometry" : {
        "location" : {
          "lat" : 46.0448994,
          "lng" : 14.4892307
        },
        ...
      },
      "types" : [ "street_address" ]
    },
    {
      ...,
      "formatted_address" : "Vič, Slovenija",
      "geometry" : {
        ...,
        "location" : {
          "lat" : 46.025078,
          "lng" : 14.4641067
        },
        ...
      },
      "types" : [ "administrative_area_level_3", "political" ]
    },
    ...,
    {
      ...,
      "formatted_address" : "1000 Ljubljana, Slovenija",
      "geometry" : {
        ...,
        "location" : {
          "lat" : 46.0514878,
          "lng" : 14.5060823
        },
        ...
      },
      "types" : [ "postal_town" ]
    },
    ...,
    {
      ...,
      "formatted_address" : "Slovenija",
      "geometry" : {
        ...,
        "location" : {
          "lat" : 46.151241,
          "lng" : 14.995463
        },
        ...
      },
      "types" : [ "country", "political" ]
    }
  ],
  "status" : "OK"
}
```

Slika 5.11: Skrajšan rezultat klica za obratno geokodiranje na Google Geo-coding API

```
{
  "destination_addresses" : [ "Tržaška cesta 77a,
                               1000 Ljubljana, Slovenija" ],
  "origin_addresses" : [ "Tržaška cesta 25, 1000 Ljubljana, Slovenija" ],
  "rows" : [
    {
      "elements" : [
        {
          "distance" : {
            "text" : "2,6 km",
            "value" : 2566
          },
          "duration" : {
            "text" : "5 min",
            "value" : 299
          },
          "status" : "OK"
        }
      ]
    }
  ],
  "status" : "OK"
}
```

Slika 5.12: Rezultat klica na Google Distance Matrix API

izbrane restavracije, opisano v podpoglavju 4.2.

Primer klica:

```
http://maps.googleapis.com/maps/api/distancematrix/json
?origins=46.0448994,14.4892307&destinations=46.0422956,14.4792024
&language=sl&sensor=false
```

Rezultat klica je viden na sliki 5.12.

Kot vidimo iz rezultata klica, nam Google Distance Matrix API vrne razdaljo med točkama (v metrih) in čas poti med točkama (v sekundah). Poleg tega nam vrne tudi prikazne vrednosti za oba rezultata.

Pri prikazu seznama restavracij se za vsako restavracijo, ki ustreza iskalnim pogojem, izračuna razdalja med uporabnikovo lokacijo in restavracijo. V primeru, da je razdalja manjša od največje dovoljene razdalje, se restavracijo doda na seznam restavracij.

Glede na to, da je restavracij v bazi lahko veliko, smo morali najti način, da zmanjšamo število klicev na Google Distance Matrix API. Za zmanjšanje števila klicev uporabljamo dve tehniki:

- filtriranje restavracij glede na zračno razdaljo med uporabnikovo lokacijo in restavracijo, opisano v nadaljevanju,
- predpomnjenje rezultatov izračuna razdalje iz Google Distance Matrix APIja, opisano v razdelku 5.2.5.

Izračun zračne razdalje med dvema točkama

Preden naredimo klic na Google Distance Matrix API, izračunamo zračno razdaljo med uporabnikovo lokacijo in izbrano restavracijo. Za izračun zračne razdalje med dvema točkama na sferi uporabljamo haversine formulo, opisano na [20]. V primeru, da je že zračna razdalja izven podane omejitve radija, lahko predpostavimo, da bo dejanska razdalja (ki ni v ravni črti) zagotovo večja. V tem primeru restavracije ne dodamo na seznam in tako niti ne kličemo Googlovega APIja. V nasprotnem primeru, torej da je razdalja do restavracije še znotraj podane omejitve, pa naredimo klic na Google Distance Matrix API, da dobimo dejansko razdaljo, katero potem primerjamo s podano omejitvijo največje dovoljene razdalje.

5.2.4 Foursquare API

Foursquare [21] je spletna platforma, s pomočjo katere uporabniki dodajajo lokacije in se na njih prijavijo (ang. check in). Poleg tega lahko uporabniki lokacije tudi ocenjujejo in komentirajo.

V naši aplikaciji uporabljamo Foursquare API [22] za pridobivanje dodatnih podatkov o restavracijah. Ti se dopolnijo, ko uporabnik preko Android aplikacije doda novo restavracijo v sistem.

Za pridobivanje dodatnih podatkov o restavraciji je najprej potrebno najti restavracijo v Foursquare sistemu, nato pa pridobiti njene podrobne podatke. Oba koraka sta opisana v nadaljevanju.

Iskanje pripadajoče restavracije v sistemu Foursquare

Pri dodajanju nove restavracije je najprej potrebno najti pripadajočo restavracijo v sistemu Foursquare, če takšna sploh obstaja. Pripadajočo restavracijo

najdemo s pomočjo metode Venue Search [23] v Foursquare APIju.

Metodi podamo 3 parametre:

- geografske koordinate lokacije restavracije,
- iskalni niz (podamo polno ime restavracije),
- identifikacijski niz kategorije lokacij, znotraj katere želimo iskati.

Primer klica:

```
https://api.foursquare.com/v2/venues/search  
?ll=46.0422956,14.4792024  
&query=Pizzerija%20Laterna  
&categoryId=4d4b7105d754a06374d81259  
&oauth_token=OAUTHTOKEN  
&v=20131130
```

Skrajšan rezultat klica je viden na sliki 5.13.

Foursquare API pri iskanju restavracij vrne več rezultatov, razvrščenih po ustreznosti. V našem strežniškem sistemu vedno vzamemo prvi rezultat, saj je za tega najbolj verjetno, da je pravi.

Pridobivanje podatkov o restavraciji iz sistema Foursquare

Po tem, ko smo v sistemu Foursquare našli ustrezno restavracijo, lahko pridobimo vse podatke za to restavracijo. To storimo s klicem metode Venue Detail [24] v Foursquare APIju.

Metodi podamo identifikacijski niz Foursquare lokacije (ang. venue), katerega dobimo kot del rezultata pri iskanju restavracije, opisanega v prejšnjem razdelku.

Primer klica:

```
https://api.foursquare.com/v2/venues/4b309b04f964a52068fa24e3  
?oauth_token=OAUTHTOKEN  
&v=20131130
```

Skrajšan rezultat klica je viden na sliki 5.14.

```

{
  ...,
  "response":{
    "venues":[
      {
        "id":"4b309b04f964a52068fa24e3",
        "name":"Pizzerija Laterna",
        "contact":{
          "phone":"012574692",
          "formattedPhone":"(01) 257 46 92"
        },
        "location":{
          "address":"Tržaška c. 79a",
          "lat":46.042198091395,
          "lng":14.479147706062482,
          "distance":11,
          "postalCode":"1000",
          "cc":"SI",
          "city":"Ljubljana",
          "state":"Slovenia",
          "country":"Slovenia"
        },
        "categories":[
          {
            "id":"4bf58dd8d48988d1ca941735",
            "name":"Pizza Place",
            "pluralName":"Pizza Places",
            "shortName":"Pizza",
            "primary":true
          }
        ],
        "verified":false,
        "stats":{
          "checkinsCount":766,
          "usersCount":149,
          "tipCount":12
        },
        "specials":{
          "count":0,
          "items":[
            ]
        },
        "hereNow":{
          "count":0,
          "groups":[
            ]
        },
        "referralId":"v-1385833857"
      },
      ...
    ]
  }
}

```

Slika 5.13: Skrajšan rezultat klica na Foursquare API za iskanje restavracij

```
{
  ...,
  "response": {
    "venue": {
      "id": "4b309b04f964a52068fa24e3",
      "name": "Pizzerija Laterna",
      "contact": {
        "phone": "012574692",
        "formattedPhone": "(01) 257 46 92"
      },
      "location": {
        "address": "Tržaška c. 79a",
        "lat": 46.042198091395,
        "lng": 14.479147706062482,
        "postalCode": "1000",
        "cc": "SI",
        "city": "Ljubljana",
        "state": "Slovenia",
        "country": "Slovenia"
      },
      "canonicalUrl": "https://foursquare.com/v/pizzerija-laterna/4b309b04f964a52068fa24e3",
      "categories": [
        ...
      ],
      "verified": false,
      "stats": {
        "checkinsCount": 766,
        "usersCount": 149,
        "tipCount": 12
      },
      "likes": {
        "count": 2,
        ...
      },
      "like": false,
      "dislike": false,
      "rating": 7.01,
      "shortUrl": "http://4sq.com/8YDN7A",
      ...
    }
  }
}
```

Slika 5.14: Skrajšan rezultat klica na Foursquare API za pridobivanje podatkov o restavraciji

S pomočjo te metode lahko iz sistema Foursquare dobimo dodatne podatke o restavraciji.

Iz rezultata klica metode vzamemo naslednje podatke, če so na voljo v rezultatu:

- telefonska številka restavracije,
- spletni naslov restavracije,
- povezava na spletni meni restavracije,
- število prijav (ang. check-in) preko spletne storitve Foursquare,
- število všečkov (ang. likes) na spletni storitvi Foursquare,
- povezava na stran restavracije znotraj spletne storitve Foursquare,
- povprečna ocena restavracije med uporabniki spletne storitve Foursquare.

5.2.5 Predpomnjenje rezultatov zahtev na Google Maps API

Kot smo opisali v razdelku 5.2.3, naš sistem za delovanje potrebuje zunanjo storitev Google Maps API. Zaradi želje po višji hitrosti delovanja smo želeli čimbolj zmanjšati število zahtev na zunanjo storitev.

Drugi razlog za zmanjšanje števila zahtev je v omejitvah največjega števila zahtev na Google Maps API. Uporaba Google Geocoding APIja je omejena na 2500 zahtev na 24 ur [15]. Uporaba Google Distance Matrix APIja pa je omejena na 100 elementov na zahtevo, 100 elementov na 10 sekund in 2500 elementov na 24 ur. Element je v tem kontekstu definiran kot število začetnih točk pomnoženo s številom ciljnih točk za domeno izračuna razdalje med dvema lokacijama [17].

Kot že omenjeno, smo število zahtev na Google Geocoding API, opisanega v razdelku 5.2.3, in Google Distance Matrix API, opisanega v razdelku 5.2.3, dosegli s predpomnjenjem rezultatov zahtev obeh storitev.

Predpomnjenje smo implementirali za tri tipe klicev na Google Maps API: klic za geokodiranje naslova, klic za obratno geokodiranje koordinat in klic za izračun razdalje med dvema koordinatama.

Pri **klicu za geokodiranje naslova** smo kot ključ za shranjevanje predpomnjene vrednosti uporabili naslov v tekstovni obliki. Predpomnjena vrednost je v tem primeru geografska koordinata naslova.

Pri **klicu za obratno geokodiranje koordinat** smo kot ključ za shranjevanje predpomnjene vrednosti uporabili vrednosti geografskih koordinat. Predpomnjena vrednost je v tem primeru naslov lokacije v tekstovni obliki.

Pri **klicu za izračun razdalje med koordinatama** smo kot ključ za shranjevanje predpomnjene vrednosti uporabili točki v obliki koordinat, med katerima želimo izračunati razdaljo. Predpomnjena vrednost je v tem primeru izračunana razdalja med obema točkama v metrih.

Dvonivojsko predpomnjenje

V strežniškem sistemu smo implementirali sistem dvonivojskega predpomnjenja. Dvonivojsko predpomnjenje je sestavljeno iz predpomnjenja v delovnem pomnilniku strežnika na prvem nivoju in shranjevanja v podatkovno bazo na drugem nivoju.

Pri vsaki zahtevi se najprej preveri ali je vrednost s podanim ključem na voljo na prvem nivoju predpomnilnika. V primeru, da vrednost obstaja, se jo uporabi kot rezultat. V primeru, da vrednost na prvem nivoju ne obstaja, se preveri drugi nivo predpomnilnika. Tukaj se ponovno vrne vrednost, če ta obstaja. V nasprotnem primeru se naredi klic na spletno storitev Google Maps API in rezultat shrani na oba nivoja predpomnilnika.

Prvi nivo predpomnjenja ima zadrževanje vrednosti nastavljeno na nekaj deset minut, po tem času so predpomnjene vrednosti samodejno odstranjene. Drugi nivo pa ima zadrževanje vrednosti nastavljeno na nekaj ur. Tako smo dosegli, da so vrednosti na prvem nivoju hranjene manj časa kot na drugem nivoju. To je pomembno zato, ker je strežniški sistem zasnovan za uporabo na distribuiranih strežniških sistemih (kot je tudi uporabljeni Windows

Azure), kjer je lahko naenkrat aktivnih več instanc strežniškega sistema. V takem primeru je drugi nivo predpomnjenja (podatkovna baza) skupen večim instancam strežniškega sistema, zato mora imeti drugi nivo nastavljen daljši čas zadrževanja predpomnjenih vrednosti kot prvi nivo predpomnjenja, ki je lasten vsaki instanci.

Za shranjevanje serializiranih vrednosti v podatkovno bazo na drugem nivoju smo uporabili knjižnico Protobuf-net, podrobneje opisano v razdelku 3.2.2.

5.2.6 Luščenje podatkov iz spletne strani www.rad-dobrojem.si

Za začetni uvoz restavracij v naš sistem smo se odločili izluščiti podatke o restavracijah in njihovih ocenah iz spletne strani www.rad-dobrojem.si [18]. Omenjena spletna stran vsebuje ocene, opise, slike in osnovne podatke o ocenjenih restavracijah. Poleg ocen in opisov profesionalnih ocenjevalcev so na strani tudi ocene uporabnikov spletne strani.

Na strani je tako na voljo veliko podatkov za posamezno restavracijo, za potrebe naše rešitve smo se pri luščenju podatkov omejili samo na ime restavracije, njen naslov in skupno oceno.

V nadaljevanju bo opisan postopek luščenja naštetih podatkov.

Pridobivanje seznama restavracij

Na spletni strani www.rad-dobrojem.si so restavracije v Sloveniji razdeljene po naslednjih pokrajinah, kot je prikazano na sliki 5.15:

- Dolenjska in Bela krajina,
- Gorenjska,
- Koroška,
- Ljubljana z okolico,
- Prekmurje,



Slika 5.15: Razdelitev restavracij po pokrajinah na www.rad-dobrojem.si

- Primorska in Notranjska,
- Štajerska.

Vsaka spletna stran določene pokrajine vsebuje seznam restavracij iz izbrane pokrajine, ki je razdeljen na več spletnih strani. Primer razdelitve je prikazan na sliki 5.16.

Primer spletnega naslova (URL) za določeno pokrajino in stran:

<http://www.rad-dobrojem.si/index.php?key=ocekat&p=2&id=1>.

Slika 5.16: Razdelitev restavracij določene pokrajine po straneh na www.rad-dobrojem.si



Slika 5.17: Restavracija iz seznama restavracij na spletni strani www.rad-dobrojem.si

V spletnem naslovu parameter **p** določa indeks spletne strani v rezultatih, parameter **id** pa določa identifikacijsko številko izbrane pokrajine.

Seznam restavracij smo pridobili tako, da smo za vsako pokrajino najprej prenesli HTML stran prve spletne strani seznama rezultatov in iz nje izluščili podatke, ki nas zanimajo. Postopek pridobivanja podatkov o posamezni restavraciji bo opisan v nadaljevanju. Po prenosu določene strani iz seznama rezultatov smo postopoma zviševali vrednost parametra **p**, ki označuje številko strani, in postopek ponovili. Postopek smo ponavljali, dokler je rezultat poizvedbe vseboval seznam restavracij. Opisani postopek smo opravili za vse pokrajine, ki nas zanimajo.

Luščenje podatkov o posamezni restavraciji

Za posamezno restavracijo iz spletne strani www.rad-dobrojem.si smo pridobili podatek o njenem imenu, naslovu in oceni.

Podatka o imenu in oceni smo pridobili iz seznama restavracij. Primer restavracije v seznamu restavracij je prikazan na sliki 5.17. Ime smo pridobili tako, da smo iz naslova vrstice vzeli podniz do prve vejice. Za navedeni primer je to 'Restavracija Most'. Oceno pa smo pridobili tako, da smo sešteli vse polne 'zvezdice' in morebitno polovično 'zvezdico' ter seštevek delili s številom vseh 'zvezdic', vključno s praznimi. Nato smo vrednost normalizirali tako, da smo jo množili z številom 5.

Naslov restavracije je na žalost na voljo šele, ko odpremo celoten opis restavracije. Naslov smo lahko izluščili le tako, da smo pridobili celotno HTML stran z opisom restavracije in poiskali naslov v rubriki 'Dodatni podatki'.

Dodatni podatki

Kako do tja: ob Mesarskem mostu nasproti tržnice

Regija: Ljubljana z okolico

Značilnost:
slovensko

Naslov: Petkovškovo nabrežje 21, Ljubljana

Telefon: +386 (0)1 232 8183

www: www.restavracija-most.si

E-pošta: most@t-2.net

Zaprto: nikoli; odprto od ponedeljka do sobote med 11. in 23. uro, v nedeljo med 11. in 22. uro (v zimskem času do 17. ure)

Lastnik: Čarli J. d.o.o.

Kuhar: Slavko Kečkeš

Cene: hladne predjedi 8 do 10, juhe 3 do 4,50, tople predjedi 7 do 16, glavne jedi 9 do 24, sladice 3,80 do 6; degustacijski meni 22 € (3 hodi) in 38 € (4 hodi)

Jedi:
internacionalizirana slovenska kuhinja

Dodatno:
dnevna ponudba

Leto obiska: avgust 2012

Slika 5.18: Dodatni podatki o restavraciji na spletni strani www.raddobrojem.si

Primer je prikazan na sliki 5.18.

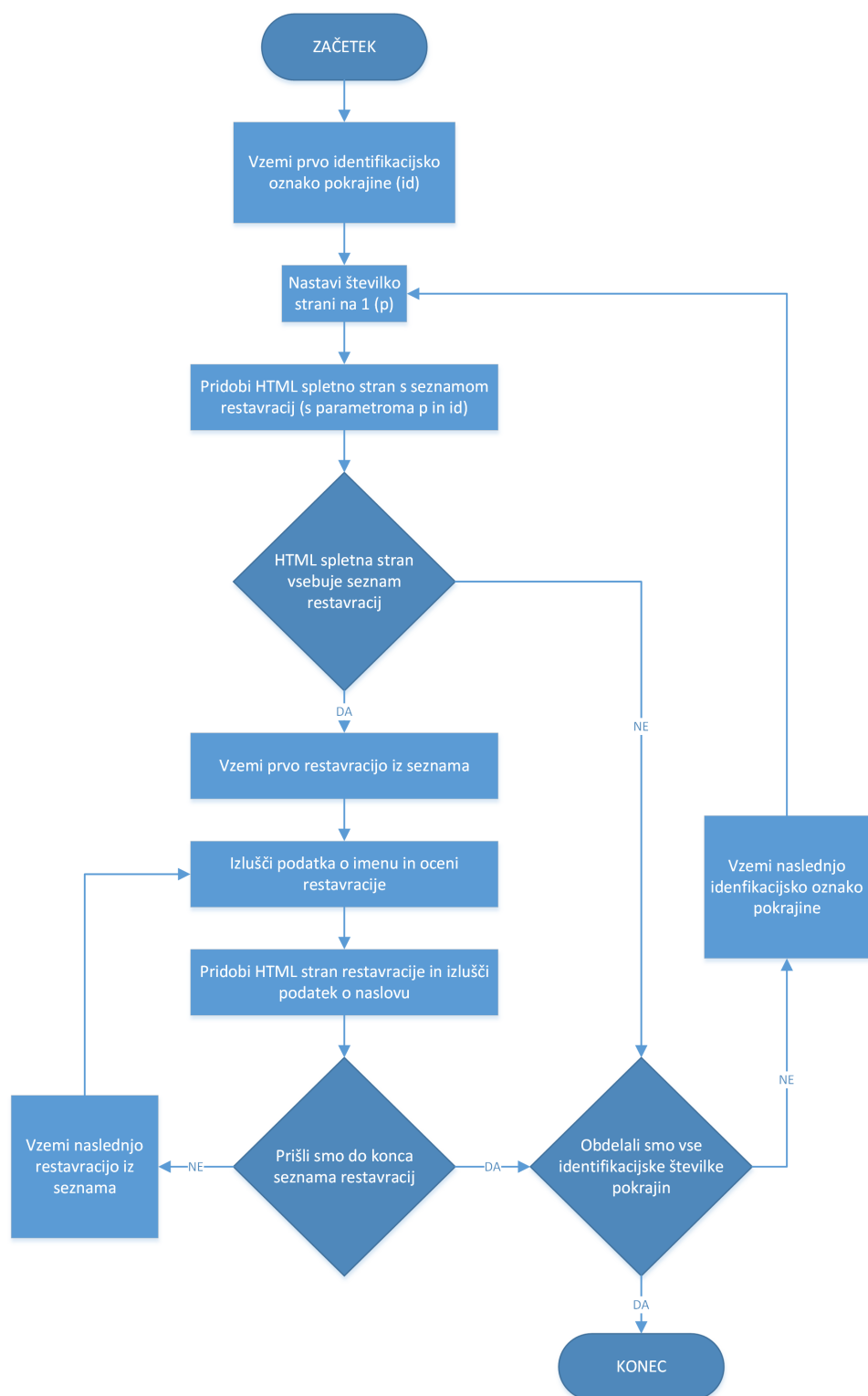
Diagram poteka 5.19 prikazuje potek luščenja restavracij iz spletne strani.

5.2.7 Opis APIja

API del strežniškega sistema je zasnovan kot ASP.NET MVC 4 projekt, ki vsebuje več kontrolerjev (ang. controller), od katerih vsak nudi metode, ki so vezane na določeno vsebinsko funkcionalnost. Celoten API je zasnovan po principu RPC (Remote Procedure Call [25], slovensko klic oddaljene metode), kjer končne točke (v našem primeru metode) opravljajo določene akcije.

Restaurant Controller

Kontroler za restavracije nudi metode, ki so povezane s pridobivanjem podatkov o restavracijah in njihovim upravljanjem. Nudi naslednje metode:



Slika 5.19: Diagram poteka luščenja restavracij iz spletne strani www.rad-dobrojem.si

- Restavracija (Restaurant) - ta metoda vrne podrobne podatke za podano restavracijo.
- Seznam restavracij (RestaurantList) - metoda vrne seznam restavracij, ki je filtriran in urejen glede na podane parametre.
- Seznam najbližjih restavracij (NearestRestaurantList) - metoda vrne seznam restavracij, ki so najbližje uporabniku. Rezultate je mogoče filtrirati le po imenu restavracije.
- Dodajanje nove restavracije (AddRestaurant) - ta metoda omogoča dodajanje nove restavracije. Vhodna parametra sta ime in naslov restavracije.
- Ocenjevanje restavracije (RateRestaurant) - metoda omogoča ocenjevanje podane restavracije po posameznih kriterijih (hrana, hitrost, cena in prijaznost).
- Pridobi naslov iz lokacije (GetAddressFromLocation) - metoda za podano lokacijo v koordinatah vrne najbližji naslov (po postopku obratnega geokodiranja, opisanega v razdelku 5.2.3).

Account Controller

Kontroler za uporabniški račun nudi metode, ki so povezane z upravljanjem uporabniškega računa. Nudi naslednji metodi:

- Prijava (Login) - ta metoda sprejme uporabniški email naslov in geslo in preveri ali je podana kombinacija veljavna.
- Registracija (Register) - ta metoda sprejme vse podatke, ki so potrebni za kreiranje novega uporabnika in iz njih ustvari nov uporabniški račun.

User Controller

Kontroler za uporabniški profil nudi metode, ki so povezane s pridobivanjem in urejanjem podatkov o uporabniškem profilu ter iskanjem, pregledom in upravljanjem prijateljev. Nudi naslednje metode:

- Moj profil (MyProfile) - metoda vrne podatke o trenutnem uporabniku (ime, priimek, starost in spol).
- Posodobi profil (UpdateProfile) - ta metoda omogoča spreminjanje uporabniških podatkov (ime, priimek, starost in spol).
- Moje ocene (MyRatings) - metoda vrne seznam vseh restavracij, katere je ocenil uporabnik. Vsaka restavracija ima poleg imena in naslova še podatek o oceni za vsak kriterij in podatek o datumu oddane ocene.
- Moji prijatelji (MyFriends) - ta metoda vrne seznam vseh uporabnikovih prijateljev.
- Iskanje uporabnikov (SearchUsers) - metoda omogoča iskanje uporabnikov glede na vpisani niz. V rezultatu so uporabniki, pri katerih je podani niz vsebovan v imenu, priimku ali elektronskem naslovu.
- Uporabnikov profil (UserProfile) - metoda vrne podatke za podanega uporabnika (ime, priimek, starost in spol). V primeru, da je podani uporabnik prijatelj trenutnega uporabnika, se vrnejo še dodatni podatki: podana povprečna ocena ocenjenih restavracij in število ocenjenih restavracij.
- Dodaj prijatelja (AddFriend) - ta metoda omogoča dodajanje podanega uporabnika kot prijatelja.
- Odstrani prijatelja (RemoveFriend) - ta metoda omogoča odstranjevanje podanega uporabnika iz seznama prijateljev.

Poglavje 6

Zaključek

V diplomski nalogi smo najprej opisali želeno informacijsko rešitev za problem iskanja in ocenjevanja restavracij. V nadaljevanju smo pregledali že obstoječe rešitve v tej domeni in naredili kritično primerjavo z našo rešitvijo. Nato smo podali naš predlog informacijske podpore za iskanje in ocenjevanje restavracij, skupaj s podrobnim opisom vseh funkcionalnosti iz uporabniškega vidika. Na koncu smo opisali še tehnične podrobnosti implementacije celotnega sistema, tako aplikacije kot strežniškega sistema. Poleg tega smo predstavili še zanimivejše uporabljene tehnologije, ki smo jih uporabili pri razvoju.

Osrednji namen diplomske naloge je bil implementacija celotne informacijske rešitve za iskanje in ocenjevanje restavracij. Naša informacijska rešitev je sestavljena iz mobilne aplikacije za operacijski sistem Android in zalednega strežniškega sistema. Naš sistem smo preko javno dostopnih APIjev povezali z dvema zunanjima storitvama, Foursquare API in Google Maps API. Iz prvega smo pridobili dodatne podatke o posamezni restavraciji, drugega pa smo uporabili za pridobivanje naslova iz koordinat in obratno ter izračun razdalje med uporabnikom in restavracijo.

Prednost naše rešitve pred že obstoječimi je v možnosti ocenjevanja in iskanja restavracij po posameznih ocenjevalnih kriterijih, ki so v našem primeru: ocena hrane, ocena primernosti cene, ocena prijaznosti osebja in ocena

hitrosti postrežbe. Poleg tega naša rešitev omogoča iskanje v določenem radiju od uporabnikove lokacije in razvrščanje restavracij glede na ocene prijateljev.

Končna implementacija rešitve je sistem, ki je dovolj stabilen in robusten za uporabo v resničnem svetu.

6.1 Nadaljnje delo

Razvoj naše informacijske rešitve se lahko nadaljuje z izboljšavami na področju uporabniške izkušnje.

V trenutni verziji naše rešitve morajo uporabniki ustvariti nov profil za uporabo nekaterih naprednih funkcionalnosti. Ta del aplikacije bi lahko izboljšali tako, da bi uporabnikom omogočili prijavo preko svojega Facebook [42] profila. Tako uporabnikom ne bi bilo potrebno ustvarjati novega profila, saj bi lahko uporabili že obstoječi Facebook profil. S tem bi naša aplikacija lahko pridobila tudi uporabnikovo sliko, katera bi lahko bila vidna v naši aplikaciji. Poleg tega bi lahko naša aplikacija na ta način pridobila tudi seznam uporabnikovih prijateljev na Facebooku. Tako uporabniku ne bi bilo potrebno ponovno iskati in dodajati ostalih uporabnikov kot svojih prijateljev preko naše aplikacije.

Poleg vsebinskih funkcionalnosti bi lahko za izboljšanje uporabniške izkušnje izboljšali tudi vizualni izgled aplikacije. S tem bi povečali krog ljudi, ki bi aplikacijo uporabljali, saj je izgled aplikacije končnim uporabnikom dandanes zelo pomemben.

Širši krog uporabnikov naše informacijske rešitve bi dosegli tudi tako, da bi razvili še aplikacijo za Applov mobilni operacijski sistem iOS. Ta je konkurenčen operacijskemu sistemu, za katerega smo razvili našo aplikacijo, torej Googlovemu operacijskemu sistemu Android.

Literatura

- [1] Seznam Android API nivojev. Dostopno na:
<http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels>

- [2] Primerjava distribucije različnih verzij Android operacijskega sistema. Dostopno na:
<http://developer.android.com/about/dashboards/index.html>

- [3] Opis Android Activity razreda. Dostopno na:
<http://developer.android.com/reference/android/app/Activity.html>

- [4] Opis Android Fragment razreda. Dostopno na:
<http://developer.android.com/reference/android/app/Fragment.html>

- [5] Opis strategije pridobivanja uporabnikove lokacije. Dostopno na:
<http://developer.android.com/guide/topics/location/strategies.html>

- [6] Razlaga ocene natančnosti pri pridobivanju lokacij. Dostopno na:
[http://developer.android.com/reference/android/location/Location.html#getAccuracy\(\)](http://developer.android.com/reference/android/location/Location.html#getAccuracy())

- [7] Razlaga standardnega odklona v verjetnosti in statistiki. Dostopno na:
http://en.wikipedia.org/wiki/Standard_deviation

- [8] Android knjižnica AQuery. Dostopno na:
<https://code.google.com/p/android-query>

-
- [9] Java knjižnica Google Gson. Dostopno na:
<https://code.google.com/p/google-gson>
 - [10] Android knjižnica Android Support Library. Dostopno na:
<http://developer.android.com/tools/support-library/index.html>
 - [11] Android knjižnica Google Play Services. Dostopno na:
<http://developer.android.com/google/play-services/index.html>
 - [12] Opis principa shranjevanja podatkov v Android aplikacijah. Dostopno na:
<http://developer.android.com/training/basics/data-storage/shared-preferences.html>
 - [13] Opis Google Maps API Web Services. Dostopno na:
<https://developers.google.com/maps/documentation/webservices/>
 - [14] Opis Google Geocoding API. Dostopno na:
<https://developers.google.com/maps/documentation/geocoding/>
 - [15] Omejitve Google Geocoding API. Dostopno na:
<https://developers.google.com/maps/documentation/geocoding/#Limits>
 - [16] Opis Google Distance Matrix API. Dostopno na:
<https://developers.google.com/maps/documentation/distancematrix/>
 - [17] Omejitve Google Distance Matrix API. Dostopno na:
<https://developers.google.com/maps/documentation/distancematrix/#Limits>
 - [18] Spletna stran z ocenami restavracij www.rad-dobrojem.si Dostopno na:
<http://www.rad-dobrojem.si/>
 - [19] Spletna različica Google Maps. Dostopno na:
<https://maps.google.com/>
 - [20] Opis haversine formule za izračun razdalje med dvema točkama na sferi. Dostopno na:
http://en.wikipedia.org/wiki/Haversine_formula

-
- [21] Spletna stran Foursquare. Dostopno na:
<https://foursquare.com/>
 - [22] Foursquare API. Dostopno na:
<https://developer.foursquare.com/docs/>
 - [23] Foursquare Venues Search API. Dostopno na:
<https://developer.foursquare.com/docs/venues/search>
 - [24] Foursquare Venue Detail API. Dostopno na:
<https://developer.foursquare.com/docs/venues/venues>
 - [25] Opis klica oddaljene metode. Dostopno na:
http://en.wikipedia.org/wiki/Remote_procedure_call
 - [26] Spletna stran TripAdvisor. Dostopno na:
<http://www.tripadvisor.com/>
 - [27] Opis strani TripAdvisor. Dostopno na:
http://www.tripadvisor.com/PressCenter-c6-About_Us.html
 - [28] Spletna stran Odpiralni Časi. Dostopno na:
<http://odpiralnicasi.com/>
 - [29] Spletna stran Foodspotting. Dostopno na:
<http://www.foodspotting.com/>
 - [30] Spletna stran Urbanspoon. Dostopno na:
<http://www.urbanspoon.com/>
 - [31] Spletna stran Yelp. Dostopno na:
<http://www.yelp.com/>
 - [32] .NET knjižnica NHibernate. Dostopno na:
<http://nhforge.org/>
 - [33] Opis .NET knjižnice NHibernate. Dostopno na:
<http://en.wikipedia.org/wiki/NHibernate>

-
- [34] .NET knjižnica Protobuf-net. Dostopno na:
<https://code.google.com/p/protobuf-net/>
- [35] Googlov mehanizem za serializacijo in deserializacijo podatkov Protocol Buffers. Dostopno na:
<https://developers.google.com/protocol-buffers/>
- [36] Primerjava učinkovitosti knjižnice Protobuf-net in nekaterih ostalih načinov serializacije. Dostopno na:
<https://code.google.com/p/protobuf-net/wiki/Performance>
- [37] Opis binarnega formata mehanizma Protocol Buffers. Dostopno na:
<https://developers.google.com/protocol-buffers/docs/encoding>
- [38] Microsoftova oblačna platforma Windows Azure. Dostopno na:
<http://www.windowsazure.com/en-us/>
- [39] Opis oblačne storitve Microsoftove oblačne platforme Windows Azure. Dostopno na:
<http://www.windowsazure.com/en-us/manage/services/cloud-services/what-is-a-cloud-service/>
- [40] Izračun cene oblačne storitve Windows Azure. Dostopno na:
<http://www.windowsazure.com/en-us/pricing/calculator/?scenario=cloud>
- [41] Opis platforme kot storitve. Dostopno na:
http://en.wikipedia.org/wiki/Platform_as_a_service
- [42] Spletna stran Facebook. Dostopno na:
<https://www.facebook.com/>