Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Luka Žnidaršič

Mobilna aplikacija za določanje lokacij z optimalnim vremenom

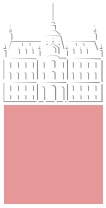
DIPLOMSKO DELO
VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

doc. dr. Miha Moškon
MENTOR

Ljubljana, 2019

Univerza
v Ljubljani

Fakulteta *za računalništvo*
in informatiko



Tematika naloge:

Kandidat naj v svojem delu analizira spletne storitve za pridobivanje vremenskih podatkov in te uporabi pri razvoju lastne aplikacije, ki bo uporabniku predlagala optimalno lokacijo v izbranem časovnem in prostorskem intervalu glede na vremenske razmere. Aplikacija naj bo namenjena mobilnim platformam, na katerih teče operacijski sistem Android.

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Spodaj podpisani izjavljam, da sem avtor dela, da slednje ne vsebuje materiala, ki bi ga kdorkoli predhodno že objavil ali oddal v obravnavo za pridobitev naziva na univerzi ali drugem visokošolskem zavodu, razen v primerih kjer so navedeni viri.
S svojim podpisom zagotavljam, da:

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

— Luka Žnidaršič, Ljubljana, marec 2019.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Luka Žnidaršič

Mobilna aplikacija za določanje lokacij z optimalnim vremenom

POVZETEK

Vreme močno vpliva na naše vsakdanje življenje. Vpliva na naše načrtovanje izletov, vikendov ali pa celo dopusta. Zato se povečuje število storitev, ki nam ponujajo vedno več in bolj natančne vremenske informacije. Na voljo nam je vedno več načinov za iskanje vremenskih razmer glede na izbrane lokacije. Po drugi strani storitev, ki bi določile lokacije na podlagi podanih vremenskih razmer, praktično ni.

Cilj te diplomske naloge je bil razviti program, ki uporabnikom omogoča iskanje najprimernejših lokacij za izbrani dan in zaželjeno vreme. Program je bil razvit za mobilne naprave, na katerih teče operacijski sistem Android. Razvit je bil s pomočjo orodja Android Studio. Za upravljanje z zemljevidi je bila uporabljena storitev Google Maps, za pridobivanje vremenskih informacij in napovedi pa je bila uporabljena storitev OpenWeatherMap. Za komunikacijo z OpenWeatherMap storitvijo je bila uporabljena Android knjižnica Volley, za preslikavo prejetih podatkov v Java objekte pa je bila uporabljena knjižnica Jackson, pridobljena z orodjem Apache Maven.

Končni rezultat diplomskega dela je aplikacija, ki je zgrajena iz treh aktivnosti. V aktivnosti za nastavitve si uporabnik izbere željene vremenske razmere. V aktivnosti za izbiro si izbere dan, ki ga zanima, ter polmer in center območja, v katerem išče željene vremenske razmere. V aktivnosti za rezultate pa so prikazane najboljše lokacije, razvrščene po ujemanju z željenimi vremenskimi razmerami na izbrani dan. Uporabnik si lahko ogleda tudi podrobnejše vremenske informacijo o lokacijah, ali pa jih razvrsti glede na ujemanje z izbranimi željenimi vremenskimi parametri.

Ključne besede: mobilna aplikacija, vreme, določanje optimalnih lokacij, Android

University of Ljubljana
Faculty of Computer and Information Science

Luka Žnidaršič

Mobile application for finding locations with optimal weather conditions

ABSTRACT

Weather heavily impacts our daily lives. It can affect the planning of our free time activities, weekends or even our vacations. Because of that, more and more weather services are offering more and more weather information. There is an increasing amount of services that allow us to find the weather conditions for chosen places, but we lack a way to find the locations where we will be able to find the weather we desire.

The goal of this thesis was to develop a program, which would allow its users to search for the optimal locations for the chosen day and weather conditions. The program was developed for mobile devices which run the Android operating system. It was developed using the Android Studio. For managing the maps, we used Google Maps and for acquiring weather information and forecasts, OpenWeatherMap service was used. For communication with the OpenWeatherMap service, we used the Android Volley library, and for mapping the acquired weather data to Java objects, Jackson library was used, which was acquired from Apache Maven.

The final result of this thesis is an application which is composed of three activities. In the settings activity, the user chooses his desired weather conditions. In the choice activity, the user chooses the days of interest and the radius and center of the area in which they wish to find the desired weather conditions. In the results activity the optimal locations are displayed, sorted by their matching with the desired weather conditions on the chosen day. The user can also view more details about weather for each of the locations or sort them by matching with specific desired weather parameters.

Key words: mobile application, weather, finding optimal locations, Android

ZAHVALA

Zahvaljujem se doc. dr. Mihi Moškonu za pomoč pri ustvarjanju diplomske naloge, svojim staršem, da so mi stali ob strani na moji poti izobrazbe in svoji punci, ki me je vztrajno vzpodbujala in podpirala.

— Luka Žnidaršič, Ljubljana, marec 2019.

KAZALO

1 Uvod

Vreme ima zelo pomemben vpliv na naše vsakdanje življenje. Lahko vpliva na planiranje našega prostega časa, izletov, vikendov ali pa celo celih dopustov. Ponudnikov vremenskih storitev je vedno več in ponujajo vedno bolj podrobne podatke. Avtomatizirani pristopi, ki bi poiskali lokacije, kjer nam bo na voljo vreme čim bolj podobno takemu, kakršnega si želimo, žal še niso bili razviti. Lahko bi sicer iskali vreme za vse kraje znotraj željenega območja in časovnega intervala, vendar bi bilo to zelo časovno potratno in neučinkovito. Zato smo se odločili, da bomo naredili program, ki to avtomatizira in nam predstavi rezultate na jasen in pregleden način.

Skozi delo bodo predstavljene tehnologije in orodja, ki so bila uporabljena pri razvoju programa, opis strukture in delovanja, primer uporabe in ideje za izboljšavo.

2 Tehnologije in orodja

Za razvoj so bile uporabljene brezplačne storitve, orodja in viri, ki so izdani pod "Creative Commons" licenco. Opisani so v nadaljevanju poglavja.

2.1 Android

Android je odprtokodni operacijski sistem zgrajen na Linux jedru [1]. Primarno je namenjen mobilnim telefonom in tablicam, vendar lahko teče tudi na ostalih napravah, kot so naprimer pametne ure, televizije, hladilniki, osebni računalniki itd. Razvija in vzdržuje ga Google in je trenutno eden izmed najbolj široko uporabljenih mobilnih operacijskih sistemov. Android aplikacije se primarno razvijajo v programskem jeziku Java, za načrtovanje vizualnih elementov pa se najpogosteje uporablja XML (angl. *eXtensible Markup Language*) [2]. Glavni gradniki Android aplikacij so aktivnosti, ki so razdeljene na Java objekt, ki poskrbi za logiko aktivnosti, in XML del, ki poskrbi za izgled aktivnosti in postavitve vizualnih elementov [3].

2.1.1 Logika aktivnosti in Java

Java je odprtokodni, splošen, objektno orientiran, imperativni programski jezik [4]. Je eden izmed najpopularnejših programskih jezikov in lahko teče na skoraj vseh platformah. Uporablja se za lokalne programe, zaledje pri spletnem programiranju in je ključen del operacijskega sistema Android in večine njegovih aplikacij.

2.1.2 Izgled aktivnosti in XML

Android programi v večini uporabljajo XML za načrtovanje vizualnih elementov v aktivnostih [2]. Pri grajenju aplikacije se XML zapis pretvori v Java objekte. XML ne podpira samo urejanja izgleda vsake aktivnosti posebej, ampak omogoča tudi grajenje modularnih elementov, ki jih lahko nato uporabimo v večih aktivnostih.

2.2 Google Maps SDK za Android

Z Google Maps SDK (angl. *Software Development Kit*) lahko v program dodamo zemljevide osnovane na storitvah Google Maps in uporabljamo Googlova geometrijska orodja. Google Maps API (angl. *Application Programming Interface*) samodejno upravlja z dostopom do Google Maps strežnikov, prenosom podatkov z njih, prikazom zemljevidov in odzive na interakcije z zemljevidom [5]. Poleg tega nam omogoča tudi dodajanje elementov, kot so poljubne oznake, krogi, črte in poljubne slike, na zemljevid.

Za upravljanje in prikaz zemljevidov bi bili lahko namesto Google Maps uporabljeni drugi SDK-ji. Dobre izbire so bile tudi TomTom, Mapbox, HERE in Mapfit. Vsi omenjeni kandidati ponujajo podobne funkcionalnosti kot SDK Google Maps, vendar sem se na koncu odločil za slednjega zaradi njegovih geometrijskih orodij in obstoječo dobro integriranostjo z okoljem Android in osnovnimi Android knjižnicami.

2.3 Volley

Volley je knjižnica za Android, ki omogoča preprosto in hitro upravljanje z zahtevki HTTP [6]. Z Volley se lahko klice HTTP izvaja sočasno brez prekinitve osnovne niti programa. Na preprost način lahko določimo kako naj se program odzove pri uspešnem klicu in kako pri napakah.

2.4 Storitev OpenWeatherMap

OpenWeatherMap je vremenska storitev, ki ponuja vremenske podatke za cel svet. Vremenski podatki se lahko pridobivajo za lokacije glede na ime mesta, poštno številko, identifikacijsko številko mesta ali pa točne koordinate [7]. Vremenski podatki pri brezplačnem računu so na voljo za trenuten čas ali pa v obliki 5-dnevne napovedi z vremenskimi podatki na vsake tri ure. Komunikacija s strežniki OpenWeatherMap se v programu upravlja s knjižnico Volley.

Ker pri planiranju izletov in dopustov ponavadi želimo pridobiti podatke za nekaj dni vnaprej, je bila v programu uporabljena 5-dnevna napoved s podatkom o vremenu na vsake tri ure, do katerega lahko pridemo preko klica API. Poleg vremenskih podatkov ta klic vrne tudi nekaj podatkov o lokaciji. Storitev lahko vrne podatke v obliki XML ali pa JSON (angl. *JavaScript Object Notation*). Zaradi večje preglednosti pri razvoju in manjše porabe podatkov je bil v programu uporabljen format JSON.

Potencialni kandidati za pridobivanje podatkov o vremenu so bili poleg storitve OpenWeatherMap še AccuWeather, The Weather Channel, Dark Sky, API APIXU Weather, World Weather Online in Weatherbit.io. Ker so vse te storitve visoke kvalitete, je bil glavni faktor, ki je vplival na končno odločitev ta, kakšne funkcionalnosti storitve ponujajo brezplačnemu uporabniku in na koliko klicev v podanem časovnem intervalu je storitev omejena.

- OpenWeatherMap ponuja trenutne vremenske razmere in 5-dnevno napoved z vremenskimi razmerami za vsako tretjo uro. Dovoljuje 60 brezplačnih API klicev na minuto.
- AccuWeather ima za brezplačne uporabnike na voljo le zelo omejene funkcionalnosti, zato je bil izločen iz izbire.
- The Weather Channel razvijalcem ponuja 500 brezplačnih API klicev dnevno, medtem ko jih OpenWeatherMap ponuja 60 na minuto, kar je skupaj 86400 dnevno. To je vodilo do izključitve The Weather Channel iz izbire.
- Dark Sky ponuja 1000 brezplačnih API klicev dnevno. Ker je to še vedno veliko manj kot OpenWeatherMap, je bil tudi Dark Sky izločen iz izbire.
- API APIXU Weather sicer nima dnevne omejitve, ima pa omejitev 5000 brezplačnih API klicev na mesec, kar je veliko manj kot 2592000, kot jih lahko na mesec iz-

vedemo z OpenWeatherMap. To je vodilo do eliminacije API APIXU Weather iz izbire.

- World Weather Online je bil elimininiran iz izbire, ker brezplačnim uporabnikom ponuja le preizkusno obdobje šestdesetih dni.
- Weatherbit.io ponuja 45 klicev na minuto in 5-dnevno napoved z vremenskimi podatki za vsako drugo uro. Ker je za optimalno delovanje programa veliko bolj pomembno koliko klicev na minuto lahko izvedemo, kot pa časovni interval med posameznimi vremenskimi podatki, smo se na koncu odločili za uporabo OpenWeatherMap.

2.5 Format JSON

JSON je človeku lahko berljiv format za izmenjavo podatkov. Podatki v formatu JSON so zapisani v obliki parov ključ-vrednost, baziran pa je na programskem jeziku JavaScript [8]. Večinoma se ta format uporablja za shranjevanje in prenos objektov.

2.6 API Jackson

API Jackson je JSON razčlenjevalnik in zapisovalnik, ki omogoča preslikavo podatkov iz JSON v objekte Java, ki ustrezajo strukturi danega objekta JSON [9]. Jackson je razdeljen na podknjižnice, ki ponujajo dodatne funkcionalnosti. Te so Core, Databind in Annotations.

2.6.1 Core

Jackson Core vsebuje osnove, kot so drevesni model, razčlenjevanje in zapisovanje podatkov v formatu JSON. Funkcionalnosti ostalih knjižnic se zanašajo na Core.

2.6.2 Databind

Jackson Databind omogoča preslikavo formata JSON v objekte Java in pretvorbe v obe smeri med obema.

2.6.3 Annotations

Jackson Annotations omogoča večjo kontrolo nad preslikavo formata JSON v objekte Java. Omogoča nam naprimer preimenovanje atributov, kar je lahko včasih nujno po-

trebno, saj Java ne podpira vseh imen atributov JSON.

2.7 Apache Maven

Apache Maven je orodje za upravljanje projektov. Baziran je na projektnem objektnem modelu (angl. *Project Object Model*, POM) in s pomočjo centralne zbirke knjižnic in dokumentacije poenostavi strukturo projekta in upravlja grajenje projekta [10].

2.8 Git

Git je brezplačen in odprtokoden sistem za nadzor različic [11]. Je hiter, omogoča pregledno dokumentiranje napredka in je dobro integriran z velikim številom razvojnih okolij.

3 Opis rešitve

Program za določanje geografskih lokacij z optimalnimi vremenskimi razmerami je razdeljen na več ključnih delov, in sicer:

- podatkovni objekti,
- pomočniški objekti,
- aktivnosti,
- dialog za razvrščanje in
- dialog za podrobnosti.

Njihove podrobnejše opise podajamo v nadaljevanju poglavja.

3.1 Podatkovni objekti

Podatkovni objekti skupaj predstavljajo popolno preslikavo vremenskih podatkov, ki jih dobimo v obliki objekta JSON s strežnika OpenWeatherMap. Korenski objekt, poimenovan WeatherResponse, vsebuje strežnikovo odzivno kodo in sporočilo, objekt City s

podatki o mestu, za katerega smo zahtevali podatke, število vremenskih podatkov in seznam objektov WeatherData z vremenskimi podatki. V objekt je bila dodana tudi metoda za iskanje vremenskih podatkov za točno določen čas, vendar je s pomočjo Jackson Annotations označena tako, da je pri preslikovanju iz objekta JSON prezrta.

Objekt City vsebuje identifikacijsko številko mesta, ime mesta, ime države, populacijo mesta in objekt Coord, ki vsebuje zemljepisno dolžino in širino mesta.

Objekt WeatherData vsebuje:

- čas vremenske napovedi,
- seznam objektov Weather,
- objekt Clouds,
- objekt Wind,
- objekt Rain,
- objekt Snow in
- objekt Main.

Seznam objektov Weather predstavlja seznam vseh možnih vremenskih pojavov (jasno, deževno, nevihtno itd.), kjer je prvi objekt pogosto edini in predstavlja glavni vremenski pojav. Objekt Clouds vsebuje podrobnejše podatke o napovedani oblačnosti, objekt Wind podrobnejše podatke o vetru, objekt Rain podrobnejše podatke o dežju, objekt Snow pa podrobnejše podatke o snegu. Objekt Main nam pove najpomembnejše podatke, kot so povprečna, minimalna in maksimalna temperatura, pritisk, vlaga itd.

3.2 Pomočniški objekti

Pomočniški objekti so vsi objekti, ki pomagajo pri računanju, sortiranju, pridobivanju in shranjevanju podatkov ter skrbijo za komunikacijo s strežnikom OpeanWeatherMap, ampak ne igrajo aktivne vloge pri interakciji z uporabnikom ali upravljanjem z grafičnim vmesnikom. Ti objekti so:

- GeometryHelper,
- PreferencesHelper,

- WeatherAPIHelper,
- WeatherIconsHelper in
- WeatherSortHelper.

3.2.1 GeometryHelper

GeometryHelper je pomočniški objekt, ki zagotavlja vse geometrijske izračune. Najpomembnejša metoda, ki se nahaja v tem objektu, je metoda getDispersedPoints. Ta metoda prejme zemljepisne koordinate in polmer območja v kilometrih, s pomočjo katerih zgenerira 30 točk, ki so naključno razpršene po površini kroga s centrom v podanih koordinatah in podanim polmerom. Da pride do teh podatkov, ta metoda uporabi naključno seme, iz katerega v zanki tridesetkrat generira naključni kot od 0 do 360 stopinj in kvadratno korenjeno vrednost med 0 in 1, pomnoženo s polmerom [12]. Nato uporabi ta kot in oddaljenost, da s pomočjo Google Maps pomočniškega objekta SphericalUtils pridobi koordinate za novo točko, ki je za tako razdaljo oddaljena od izbrane lokacije v smeri izračunanega kota. Preden točko doda v seznam rezultatov, preveri, če je katerikoli že obstoječi točki bližja od desetine premera. Če je, še enkrat ponovi izračun in to ponavlja, dokler ne dobi točke, ki je primerno oddaljena od vseh ostalih. Minimalna razdalja desetine premera je bila izbrana zato, ker so si pri tej razdalji točke na zemljevidu dovolj narazen, da se ne prekrivajo ne vizualno in ne po območju za dotik.

3.2.2 PreferencesHelper

PreferencesHelper je objekt, ki je zadolžen za shranjevanje in branje podatkov iz vgrajene shrambe parov ključ-vrednost SharedPreferences. Za pravilno delovanje zahteva, da se mu pri konstrukciji poda aplikacijski kontekst. S tem zagotovimo, da so lahko podani podatki naloženi v vseh aktivnostih, ne samo v tisti, v kateri se kontekst shrani. Objekt se uporablja za shranjevanje željenih vremenskih razmer v aktivnosti za nastavitve in njihovem branju pri nalaganju najboljših rezultatov v aktivnosti za rezultate.

3.2.3 WeatherAPIHelper

WeatherAPIHelper je objekt, ki je zadolžen za komunikacijo z OpenWeatherMap strežnikom. Ta objekt se uporablja za pridobitev vremenskih podatkov v izbirni aktivnosti. Pri konstrukciji mu podamo seznam koordinat, katerega potem uporabi pri metodi getWeather, da za vsako lokacijo izvede klic na strežnik za pridobitev vremenskih podatkov. Za

komunikacijo s strežnikom uporablja knjižnico Volley, ki za vsak klic vzpostavi upravljalca za primer uspešnega odziva in upravljalca za primer napake. Upravljalci za uspešne klice shranjujejo vse rezultate in štejejo koliko odzivov je bilo obdelanih, upravljalci za napake pa le beležijo koliko odzivov je bilo obdelanih. Ko se obdela toliko odzivov, kot je bilo podanih lokacij v seznamu koordinat pri konstrukciji, bo upravljalec, ki je prejel zadnji odziv, poklical izbirno aktivnost za nadaljnje izvajanje. Vsak klic strežniku poda zemljepisno širino in dolžino lokacije, za katero želi pridobiti vremenske podatke, hkrati pa tudi sporoči, da naj bodo podatki v metričnih enotah, natančnega tipa in v formatu JSON.

3.2.4 WeatherIconsHelper

WeatherIconsHelper je objekt, ki je zadolžen za pridobivanje ikon. Uporablja se v aktivnosti za rezultate, kjer pomaga pridobiti ikone za vreme, moč vetra in zastavo države. Za pridobitev vremenskih ikon in ikon za moč vetra mu moramo pri konstrukciji podati kontekst, s pomočjo katerega lahko dostopa do grafičnih virov aplikacije. Zastave držav pridobi tako, da sprejme dvomestno kodo države in z njeno pomočjo poišče pravo zastavo Unicode za to državo. Za vremenske ikone so bile uporabljene Amedia Weather Icons [13], ki so bile popravljene na prave velikosti za uporabo v aplikacijah Android.

3.2.5 WeatherSortHelper

WeatherSortHelper pomaga sortirati vremenske objekte in se uporablja v aktivnosti za rezultate in v dialogu za izbiro načina sortiranja. Vremenske objekte sortira po bližini željenim vremenskim razmeram, nastavljenim v aktivnosti za nastavitve. Sortirani so lahko glede na temperaturo, hitrost vetra, smer vetra ali pa glede na vse skupaj, kjer se upošteva tudi željen tip vremena.

Vsako sortiranje najde 6 najboljših rezultatov. Sortiranje po temperaturi poišče rezultate, ki so po temperaturi najbližje izbrani temperaturi. Sortiranje po hitrosti vetra ima 3 nivoje: šibek, srednji in močan veter. Če imamo izbran šibek veter, poišče, rezultate s čim nižjo hitrostjo vetra, če imamo izbran močan veter, poišče rezultate s čim večjo hitrostjo vetra, če pa imamo izbran srednji veter pa poišče rezultate, kjer je hitrost vetra najbližje 12 m/s, kar se nahaja na sredini šeste stopnje (od dvanajstih) na Beaufortovi lestvici jakosti vetrov [14]. Sortiranje po smeri vetra poišče rezultate, ki so po smeri vetra najbližje izbrani smeri. Pri računanju za vsak rezultat izračuna razliko v kotu v smeri urinega kazalca in v obratno smer urinega kazalca in upošteva nižjo vrednost. Če

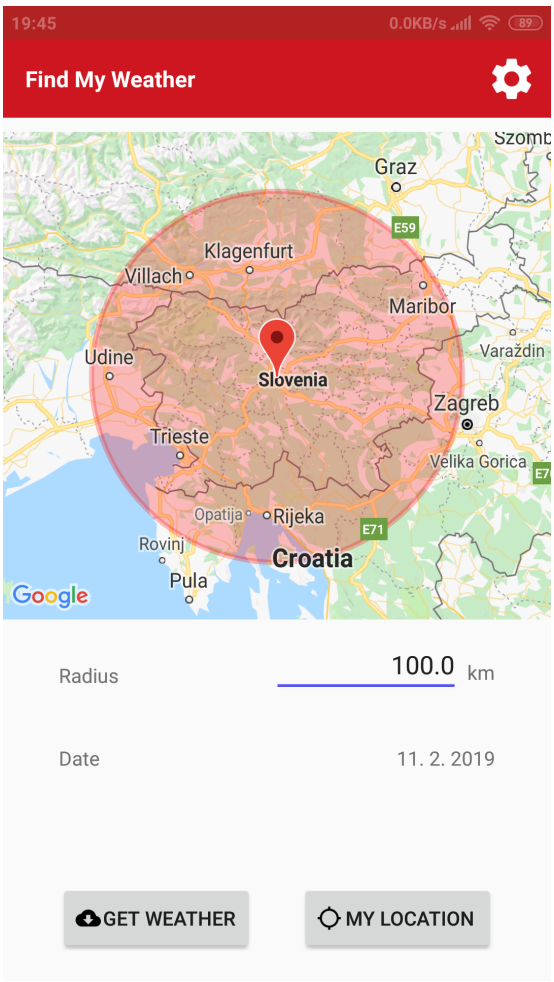
je v aktivnosti za nastavitve bila izbrana smer vetra “Any”, je sortiranje po smeri vetra onemogočeno.

Sortiranje po vseh željenih razmerah skupaj predpostavi, da so nam vse izbrane vremenske razmere enako pomembne. Pri tej vrsti sortiranja se uporablajo tudi tipi vremena, kateri so razporejeni od lepšega proti slabšem vremenu v sledečem vrstnem redu: jasno, atmosferično (megla, dim, prah itd.), prš, deževje, nevihta, sneg. Najprej se izvede prehod preko vseh rezultatov in izračuna največje razlike za vsakega izmed parametrov od izbranih vremenskih razmer. Nato za vsakega od rezultatov izračuna razliko za vsak parameter od izbranih vremenskih razmer in jih deli s prej izračunanimi največjimi razlikami, da dobi relativno razliko za vsak parameter. Nato ustvari povprečje teh relativnih razlik. Na koncu vrne rezultate, ki so najbližje izbranim parametrom po povprečju relativnih razlik. Če je v aktivnosti za nastavitve bila izbrana smer vetra “Any”, je smer vetra prezrta pri sortiranju.

3.3 Izbirna aktivnost

Ko zaženemo aplikacijo, se najprej srečamo z izbirno aktivnostjo (glej sliko 3.1). Ta aktivnost služi izbiri parametrov, glede na katere bi radi iskali svoje željene vremenske razmere. Ob zagonu aplikacija najprej preveri razpoložljivost različnih lokacijskih storitev in s pomočjo najboljše, ki nam je na voljo, poskusi poiskati našo lokacijo. Ko natančnost lokacije doseže zadostno natančnost, je naša lokacija prikazana na zemljevidu. Če nimamo omogočene nobene lokacijske storitve, nas aplikacija na to opozori. Če se želimo kasneje spet vrniti na našo lokacijo ali pa jo posodobiti po večjih premikih, lahko prej opisan proces ponovimo s pritiskom na gumb “My location”. Če želimo izbrati poljubno lokacijo, lahko to preprosto naredimo s pritiskom na zemljevid na željeno lokacijo.

Poleg lokacije moramo izbrati še polmer območja okoli izbrane lokacije, znotraj katerega želimo izvesti iskanje. Polmer se vpisuje v kilometrih in med pisanjem popravlja nivo povečave zemljevida, tako da je vidno celotno območje. Ker ima Google Maps težave s prikazovanjem prevelikih krogov v bližini polov, je bil polmer območja omejen na največ 1000 km. Takšna velikost nam še vedno omogoča pokritje zelo širokega območja, medtem ko nam tudi zagotovi, da se ne bomo srečevali z vizualnimi napakami pri izbiri območja. Spodnja meja polmera območja je bila določena zaradi vremenskih razlogov. Ker se na območju manjšem od 20 km zelo redko vremenske razmere opazno razlikujejo, je bil pol-



Slika 3.1 Izbirna aktivnost uporabniškega vmesnika aplikacije, ki omogoča izbiro lokacije in določitev parametrov za iskanje.

mer območja omejen na najmanj toliko. Pri tej velikosti polmera si zagotovimo zadostno variacijo v vremenskih razmerah znotraj območja, medtem ko je taka razdalja še vedno popolnoma razumna in dosegljiva za večino ljudi. Če imamo izbran prenizek ali previsok polmer območja, nas program na to opozori in velikost polmera območja popravi na najvišjo ali najnižjo dovoljeno vrednost, odvisno od tega, ali smo vpisali previsoko ali prenizko vrednost.

Zadnja stvar, ki jo moramo izbrati je dan, za katerega želimo poiskati določene vremenske razmere. V večini primerov nam je dovoljena izbira kateregakoli dneva od trenutnega dneva do štirih dni kasneje. Če je ura že preko 21.00, nam je onemogočena izbira

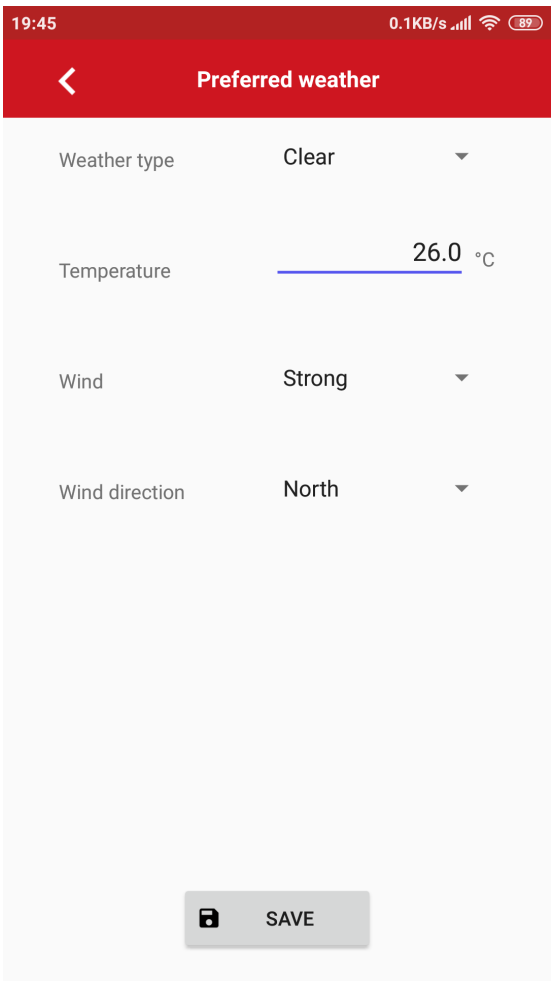
trenutnega dneva, ker so vremenski podatki na voljo le za vsako tretjo uro, kjer je 21.00 zadnja ura, za katero lahko pridobimo podatke. Od časov, za katere so nam na voljo vremenski podatki, je 15.00 najbolj reprezentativen za celoten dan. Če je izbran katerikoli sledeči dan ali pa današnji dan do 15.00, nam bo pri rezultatih prikazano vreme za 15. uro. Če je izbran trenutni dan, ampak po 15. uri, bo pri rezultatih prikazano vreme za prvo naslednjo uro, ki je večkratnik števila 3.

Ko zaključimo izbiro, pritisnemo na gumb "Get weather". Ob tem dejanju aplikacija najprej preveri ali ima dostop do spleta. Če ga nima, nas na to opozori in ne nadaljuje procesa. Če pa je povezava na splet omogočena, se pokliče GeometryHelper, ki generira 30 enakomerno porazdeljenih lokacij znotraj izbranega območja. Nato se s pomočjo WeatherAPIHelper pridobi vremenske podatke za vse generirane lokacije. Vremenski podatki so nato s pomočjo Jackson Databind pretvorjeni v objekte Java. Ker se lahko lokacije vremenskih podatkov rahlo razlikujejo od generiranih lokacij, program še enkrat preveri, če so vse lokacije znotraj izbranega območja in če so vse še vedno zadostno oddaljene druga od druge. Vse, ki ustrezajo pogojem, so dodane na končen seznam, ostale pa so prezrte. Po tem nas program preusmeri na aktivnost za rezultate, kateri poda vse podatke, ki jih ta potrebuje za pravilen prikaz rezultatov. Če pri prodobivanju ali pretvorbi vremenskih podatkov pride do kakšne napake, nas program na to opozori. Če je podatkov premalo, nas program na to opozori in nas ne spusti v naslednjo aktivnost.

Če želimo popraviti željene vremenske razmere, lahko dostopamo do aktivnosti za nastavitve s pritiskom na gumb z ikono za nastavitve, ki se nahaja v zgornjem desnem kotu.

3.4 Aktivnost za nastavitve

Do aktivnosti za nastavitve, ki je prikazana na sliki 3.2, lahko dostopamo iz katerekoli druge aktivnosti. V tej aktivnosti si lahko ogledamo in izberemo kakšne vremenske razmere želimo iskati. Na voljo so nastavitve tipa vremena, temperature, moči vetra in smeri vetra. Če se odločimo, da se želimo vrniti na prejšnjo aktivnost, nas bo program vrnil na aktivnost, s katere smo prišli. Če pa se odločimo za shranjevanje nastavitev, nas program ponovno pošlje na izbirno aktivnost. Za shranjevanje podatkov poskrbi objekt PreferencesHelper.

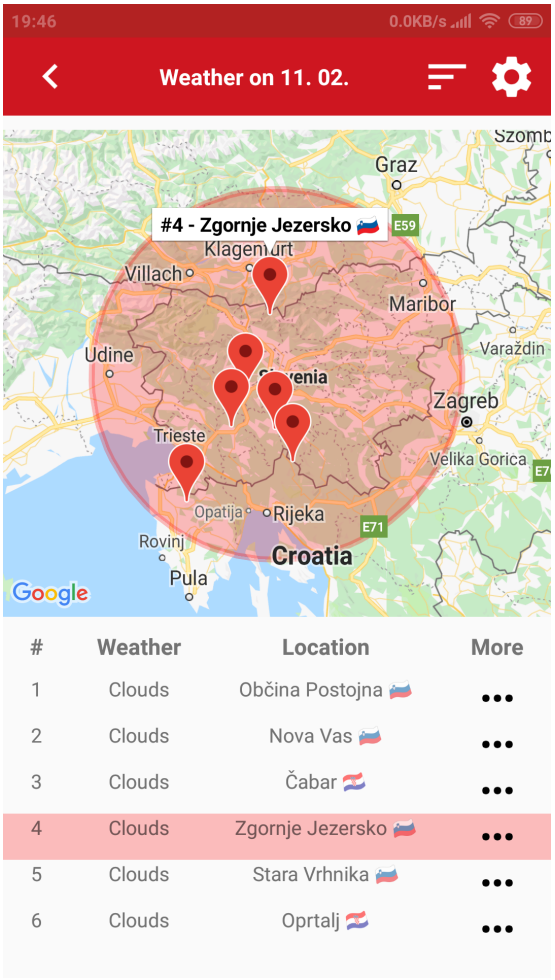


Slika 3.2 Uporabniški vmesnik aktivnosti za nastavitve preko katerega lahko bolj natančno določimo željene vremenske razmere.

3.5 Aktivnost z rezultati

Aktivnost z rezultati (glej sliko 3.3) nam prikaže 6 najboljših rezultatov iz izbranega območja, razvrščene glede na željene vremenske razmere in izbrani dan, ki je prikazan v orodni vrstici na vrhu zaslona. Na zemljevidu so prikazane njihove lokacije, v tabeli pa njihovi vremenski in lokacijski podatki. Število rezultatov je bazirano na primernem razmerju med velikostjo zemljevida in tabele, na velikost katere vpliva velikost besedila v tabeli. Ob pritisku na tabelo se bo obarvala izbrana vrstica in prikazal opis nad oznako na zemljevidu. Ob pritisku na oznako na zemljevidu se bo prikazal njen opis in

označila njena vrstica v tabeli. Ob pritisku na označeno vrstico v tabeli ali pa na območje zemljevida brez oznak bo program zemljevid in tabelo vrnil v začetno, neoznačeno stanje. Ob prvem prikazu so rezultati sortirani glede na vse željene parametre. Tako kot v izbirni aktivnosti je tudi tu možen dostop do aktivnosti za nastavitve s pomočjo gumba z ikono za nastavitve v skrajnem zgornjem desnem kotu.

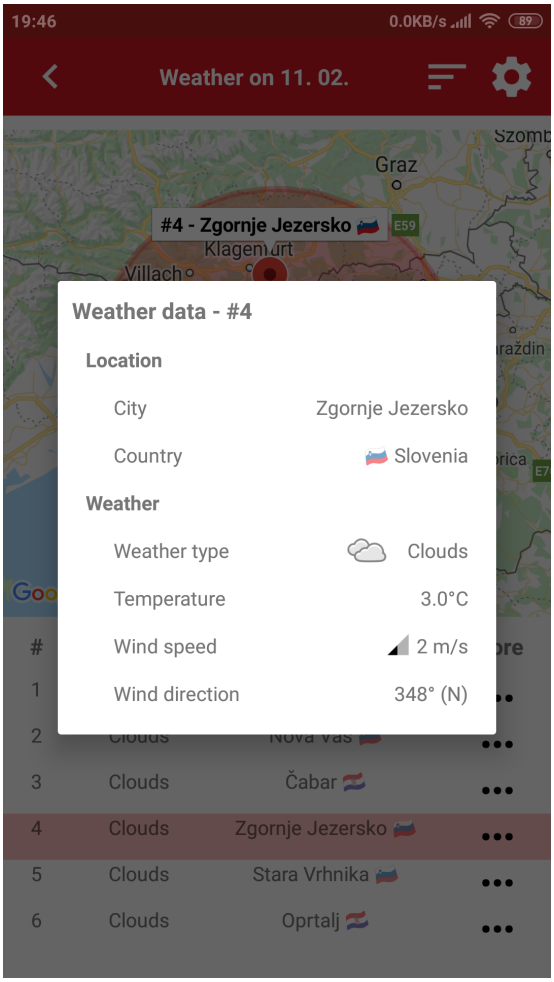


Slika 3.3 Uporabniški vmesnik aktivnosti, ki prikazuje rezultate iskanja lokacije.

3.5.1 Podrobnosti o lokaciji in vremenu

Tabela v vsaki vrstici poleg vrstnega števila lokacije, tipa vremena in lokacije, vsebuje tudi gumb za prikaz več podrobnosti o lokaciji in vremenu na njej. Ob pritisku na ta gumb se nam odpre dialog s podrobnostmi (glej sliko 3.4). Na vrhu je prikazano

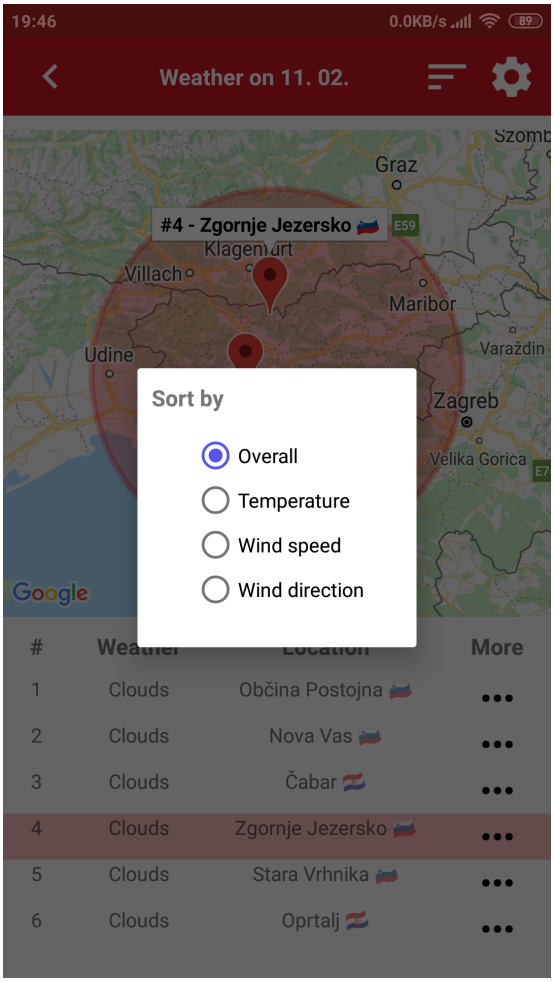
zaporedno število lokacije. Temu sledijo podatki o lokaciji, kjer sta prikazana ime mesta in ime države skupaj z ikono države, pridobljeno s pomočjo objekta WeatherIconsHelper. Temu sledi območje s podatki o vremenskih razmerah. Najprej je prikazan tip vremena, skupaj z ustrezno ikono, pridobljeno s pomočjo objekta WeatherIconsHelper. Temu sledi temperatura, izražena v stopinjah Celzijev. Nato je prikazana moč vetra z ustrezno ikono, ki je tudi pridobljena s pomočjo objekta WeatherIconsHelper. Zadnji podatek nam pove smer vetra v stopinjah in poenostavljeni smeri neba v oklepajih.



Slika 3.4 Dialog s podrobnostmi o vremenu na izbrani lokaciji.

3.5.2 Sortiranje rezultatov

Če si uporabnik želi ogledati rezultate sortirane na drugačen način od privzetega, lahko pritisne na gumb za sortiranje, ki se nahaja levo od gumba za nastavitve. To dejanje odpre dialog, kjer lahko uporabnik določi način sortiranja podatkov (glej sliko 3.5). Če je kot željena smer vetra izbrana vrednost "Any", opcija sortiranja po smeri vetra ni dostopna. Ob pritisku na katerokoli izmed možnosti za sortiranje, se na novo zgradi seznam najboljših rezultatov s pomočjo objekta WeatherSortHelper. Po tem se dialog zapre in na zemljevidu in v tabeli se prikažejo novi najboljši rezultati glede na izbran način sortiranja. Pri odprtju tega dialoga je označen tip sortiranja, po katerem so razvrščeni rezultati.



Slika 3.5 Dialog za sortiranje rezultatov iskanja.

4 Primer uporabe

V tem poglavju je opisano kako bi program uporabil amaterski jadralec, da bi prišel do kraja s čimboljšimi razmerami za jadranje.

4.1 Izbor željenih vremenskih razmer

Ob zagonu aplikacije bi najprej obiskal nastavitve, da bi si nastavil željene vremenske razmere. Neprofesionalni jadralci se večinoma izogibajo sunkovitih vetrov in si želijo čim jasnejšega in toplejšega vremena. Dober primer stalnega, nesunkovitega vetra na Jadranu je jugo, ki piha jugozahodno in dosega srednje moči. Amaterski jadralec bi se zato odpravil v nastavitveno aktivnost in za željene vremenske pogoje nastavil jasen tip vremena, jugovzhodni veter srednje moči in temperaturo 30 °C. Nato bi shranil podatke in se vrnil na izbirno aktivnost.

4.2 Izbira lokacije, območja in časa

Amaterski jadralec bo želel najti lokacijo na jadranu in se bo želel držati bližine Splita, ki je znan po tem, da ima stabilno vreme, ki redko hitro preide v nevihte. Zaradi tega bo

jadralčeva izbrana centralna lokacija morje pri Splitu. Za polmer območja si bo izbral 30 km, ker je to še vedno relativno kratka vožnja za potencialno občutno boljše vreme. Nato si bo izbral dan, na katerega bo želel jadrati. Ko bo izbral časovni in prostorni interval, za katerega se zanima, bo s pritiskom na gumb "Get weather" prešel na aktivnost z rezultati.

4.3 Pregled in analiza rezultatov

Po uspešni pridobitvi podatkov, ga bo program preusmeril na aktivnost z rezultati, kjer mu bodo najprej predstavljeni rezultati, sortirani po vseh izbranih vremenskih razmerah. Vsak predstavljen rezultat si bo dobro ogledal in če ne bo našel lokacije s primerno temperaturo ali močjo vetra, bo podatke sortiral glede na temperaturo ali moč vetra in si ponovno ogledal rezultate. Če mu bo smer vetra zelo pomembna, bo podatke sortiral še po smeri vetra in si ogledal rezultate.

4.4 Odločitev

Ko bo jadralec pregledal dovolj rezultatov, bo našel tistega, ki ga bo najbolj zanimal in se pripravil za jadranje ob tistem času in na tistem kraju. Če ne bo našel lokacije s primernimi vremenskimi razmerami za izbrani dan, bo nastavil drugačne vremenske razmere ali drug dan in postopek ponovil.

5 Zaključek

Opisani program predstavlja mobilno aplikacijo za naprave, na katerih teče operacijski sistem Android. Bodočim uporabnikom omogoča iskanje najboljših lokacij, kjer bodo našli željeno vreme ob izbranem času. Program uporablja SDK Google Maps in API OpenWeatherMap in zaradi tega ponuja zanesljive vremenske in lokacijske rezultate. Hkrati podpira funkcionalnosti, ki pohitrijo iskanje, kot je naprimer pridobitev lastne lokacije, sortiranje rezultatov na različne načine in prikaz vremenskih in lokacijskih podrobnosti z uporabniku prijaznimi ikonami za učinkovito predstavitev podatkov.

Z več časa, sredstvi ali več raziskave bi se program dalo še nadalnje izboljšati. Nekaj pomembnih izboljšav, ki bi se jih dalo narediti je:

- stalna radialna porazdelitvena mreža za lokacije,
- pridobitev vremena za večje število lokacij,
- terensko zavedna generacija lokacij,
- prilagodljivo sortiranje,
- iskalna vrstica za lokacije,

- več vremenskih preferenc in podrobnosti,
- izbira ure in časovni pasovi in
- internacionalizacija.

5.1 Stalna radialna porazdelitvena mreža za lokacije

Ena izmed bolj izstopajočih spornih točk v programu je uporaba naključnosti za generiranje enakomerne porazdelitve točk po zemljevidu. Naključnost ima zelo nizko možnost generiranja popolnoma enakomerne porazdelitve, kateri se približuje z večjim številom generiranih točk. Pri trideset točkah in dodatnimi pogoji za preprečitev prevelike bližine med točkami je mogoče doseči praktično uporabno približno enakomerno porazdelitev, vendar to še vedno ne zagotovi konsistence med rezultati pri večkratnemu iskanju pri enakih nastavitvah.

Z uporabo naprednejše matematike bi bilo možno razviti stalno radialno porazdelitveno mrežo, ki bi se raztegovala glede na velikost območja in bi vedno zagotovila enakomerno relativno porazdelitev točk znotraj poljubnega območja.

5.2 Pridobitev vremena za večje število lokacij

Manj točk kot uporabimo za pridobitev podatkov, manj jasno predstavo dobimo o vremenskih razmerah v izbranem območju, kar tudi lahko premočno obteži anomalije v podatkih. Čeprav API OpenWeatherMap ponuja še največ klicev na časovni interval, je to še vedno zgolj 60 klicev na minuto. Da se to lahko prenese v celo število celotnih pridobivanj podatkov o vremenu, želimo, da vsako pridobivanje vremena kliče vreme za število lokacij, ki je deljitelj števila 60. Možnost 60 lokacij bi pomenila le eno pridobivanje vremenskih podatkov na minuto, kar bi pri zmotah pomenilo predolgo čakanja za naslednji klic, zaradi česar sem to možnost izločil. V nadaljevanju je bilo preizkušenih 20 in 30 klicev na pridobivanje vremena. 30 lokacij je vračalo občutno boljše porazdelitve in rezultate od 20 lokacij na pridobivanje vremena, zato je bila končna odločitev 30 klicev.

Če bi bil za program uporabljen vremenski API s plačljivim računom, ki bi omogočil veliko večje število klicev, ne bi bil omejitven faktor količina klicev, ampak čas prenosa vremenskih podatkov s strežnika. Ker je čas prenosa podatkov na sodobnih mobilnih napravah praviloma zelo majhen, bi se dalo količino klicev na eno iskanje občutno povečati.

5.3 Terensko zavedna generacija lokacij

V končni aplikaciji generiranje točk znotraj območja interesa predpostavi, da je optimalna porazdelitev točk enakomerna porazdelitev. V tem kontekstu bi bilo veliko bolje uporabiti porazdelitev, ki odraža teren v območju interesa. Če bi lahko identificirali območja z enakim terenom, kjer so vremenske razmere večino časa enake, in območja, kjer se teren hitro spreminja in je na manjši površini pogosto prisotnih več različnih tipov vremena, bi lahko za manj raznolika območja uporabili manj točk, kar bi nam sprostilo več točk za uporabo na območjih z več različnimi tereni. S tem bi uporabniku podali jasnejšo sliko o vremenskih razmerah znotraj območja interesa.

5.4 Prilagodljivo sortiranje

Algoritem za sortiranje vremenskih podatkov pri uporabi vseh izbranih podatkov predpostavi, da so uporabniku vsi parametri enako pomembni. Lahko bi se izvedla raziskava o tem, kako pomembni so ljudem različni vremenski parametri. Po rezultatih te raziskave bi lahko pridobljene podatke uporabili kot uteži pri sortiranju vremenskih podatkov. Poleg tega bi lahko omogočili uporabniško prilagajanje pomembnosti vsakega izmed vremenskih parametrov, s čimer bi si lahko vsak uporabnik sortiral podatke tako, kot bi jih sam hotel.

5.5 Iskalna vrstica za lokacije

Za tiste manj pripravljene ali manj geografsko zavedne sta lahko gumb za trenutno lokacijo in zemljevid premalo za izbiro lokacije, ki jih zanima. Da bi bilo iskanje lokacij lažje, bi lahko dodali v izbirno aktivnost iskalno vrstico, ki bi predlagala rezultate s pomočjo API Google Places.

5.6 Več vremenskih preferenc in podrobnosti

Nekateri ljudje bi si lahko želeli podrobnejše izbiranje željenih vremenskih razmer. Za njih bi v nastavitveno aktivnost lahko dodali več vremenskih parametrov in jih integrirali v sortirne algoritme v objektu WeatherSortHelper. Prav tako bi lahko več podrobnosti dodali v okno z vremenskimi in lokacijskimi podrobnostmi.

5.7 Izbira ure in časovni pasovi

Končna aplikacija prikazuje vremenske podatke za 15. uro izbranega dneva za vse lokacije, ker je ob tem času največ ljudi budnih in ker vreme ob tem času večinoma najboljše predstavlja vreme dneva. Če je izbran trenutni dan in je čas že prek 15. ure, je prikazano vreme za naslednji večkratnik števila 3 (ker so vremenski podatki na voljo le za vsako 3. uro v dnevu) od trenutnega časa. Da bi omogočili podrobnejšo izbiro časovnega intervala, bi lahko uporabnikom dali možnost, da bi sami izbrali uro, za katero jih vreme zanima.

Aplikacija prikazuje vreme za točno določen čas, ki pa ni prilagojen na časovni pas lokacije. Zaradi časovnih razlik med različnimi časovnimi pasovi, bi lahko naredili časovno korekcijo podatkov glede na časovno območje, kjer bi to bilo mogoče, drugje pa opozorili na možno neveljavnost podatkov.

5.8 Internacionalizacija

Za boljšo internacionalizacijo aplikacije bi se lahko dodalo več jezikov in implementiralo možnost izbire sistema enot (metrični/imperialni).

Literatura

[1] Android Description, Dosegljivo: https://techterms.com/definition/android, [Dostopano: 8. 2. 2019].

[2] Android Fundamentals, Dosegljivo: https://developer.android.com/guide/components/fundamentals, [Dostopano: 8. 2. 2019].

[3] Z. Mednieks, B. Meike, L. Dornin, M. Nakamura, Programming Android, 2nd Edition, O'Reilly Media, 2012.

[4] About the Java Technology, Dosegljivo: https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html, [Dostopano: 9. 2. 2019].

[5] Maps SDK for Android Overview, Dosegljivo: https://developers.google.com/maps/documentation/android-sdk/intro, [Dostopano: 9. 2. 2019].

[6] Volley Overview, Dosegljivo: https://developer.android.com/training/volley/, [Dostopano: 10. 2. 2019].

[7] OpenWeatherMap Introduction, Dosegljivo: https://openweathermap.org/guide, [Dostopano: 10. 2. 2019].

[8] JSON Introduction, Dosegljivo: https://www.json.org/, [Dostopano: 10. 2. 2019].

[9] FasterXML Projects, Dosegljivo: http://fasterxml.com/projects.html, [Dostopano: 10. 2. 2019].

[10] What is Maven, Dosegljivo: https://maven.apache.org/what-is-maven.html, [Dostopano: 10. 2. 2019].

[11] About Git, Dosegljivo: https://git-scm.com/, [Dostopano: 10. 2. 2019].

[12] Programming.Guide - Generating a random point inside a circle, Dosegljivo: https://programming.guide/random-point-within-circle.html, [Dostopano: 19. 2. 2019].

[13] Amedia Weather Icons - GitHub, Dosegljivo: https://github.com/amedia/meteo-icons, [Dostopano: 12. 2. 2019].

[14] Beaufort Wind Scale, Dosegljivo: https://www.spc.noaa.gov/faq/tornado/beaufort.html, [Dostopano: 16. 2. 2019].

[15] FindMyWeather - GitHub, Dosegljivo: https://github.com/BassGuitarPanda/FindMyWeather, [Dostopano: 25. 2. 2019].

A Priloga

A.1 Koda v repozitoriju GitHub

Celoten projekt Android je dostopen na repozitoriju GitHub [15].