Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko

Dejan Orter

RAZVOJ MOBILNE IN SPLETNE APLIKACIJE LOCATOR

Diplomsko delo

Maribor, junij 2017

RAZVOJ MOBILNE IN SPLETNE APLIKACIJE LOCATOR

Diplomsko delo

Študent(ka):	Dejan Orter
Študijski program:	Visokošolski študijski program
	Računalništvo in informacijske tehnologije
Mentor(ica):	doc. dr. Matej Črepinšek
Lektor(ica):	dip. Sloveniska (UNI) Petra Potočnik

Univerza v Mariboru

Fakulteta za elektrotehniko, računalništvo in informatiko
Smetanova ulica 17
2000 Maribor, Slovenija

FERI

Številka: E1074611
Datum in kraj: 22. 03. 2017, Maribor

Na osnovi 330. člena Statuta Univerze v Mariboru (Statut UM – UPB 11, Ur. l. RS, št. 44/2015) izdajam

SKLEP O ZAKLJUČNEM DELU

1. **Dejanu Orterju,** študentu študijskega programa prve stopnje VS RAČUNALNIŠTVO IN INFORMACIJSKE TEHNOLOGIJE, se dovoljuje izdelati zaključno delo.

2. Tema zaključnega dela je pretežno s področja Inštituta za računalništvo.

3. **MENTOR: doc. dr. Matej Črepinšek**

4. **Naslov zaključnega dela:**
   RAZVOJ MOBILNE IN SPLETNE APLIKACIJE LOCATOR

5. **Naslov zaključnega dela v angleškem jeziku:**
   DEVELOPMENT OF THE MOBILE AND WEB APPLICATION LOCATOR

6. Rok za izdelavo in oddajo zaključnega dela je 30. 09. 2017. Zaključno delo je potrebno izdelati skladno z "Navodili za izdelavo zaključnega dela" in ga v treh izvodih (dva trdo vezana izvoda in en v spiralo vezan izvod) oddati v pristojnem referatu članice. Hkrati se odda tudi izjava mentor-ja/-ice (in morebitnega somentor-ja/-ice) o ustreznosti zaključnega dela.

Pravni pouk: Zoper ta sklep je možna pritožba na Senat članice v roku 10 delovnih dni od dneva prejema sklepa.

Dekan:
red. prof. dr. Borut Žalik

UNIVERZA V MARIBORU FAKULTETA ZA ELEKTROTEHNIKO, RAČUNALNIŠTVO IN INFORMATIKO

Obvestiti:
- kandidata,
- mentor-ja/-ico,
- odložiti v arhiv.

Razvoj mobilne in spletne aplikacije Locator

Ključne besede: Android aplikacije, mobilne aplikacije, lokacija, GPS

UDK: **621.397.7-026.26(043.2)**

Povzetek

Cilj diplomske naloge je izdelava Android aplikacije Locator, ki staršem ali skrbnikom omogoča lažji nadzor nad otroki. Aplikacija je sestavljena iz dveh delov. Iz primarne aplikacije, ki se izvaja v ozadju na otrokovem telefonu ter intervalno pošilja otrokovo lokacijo na strežnik ali na mobilni telefon preko sporočila SMS. Primarno aplikacijo upravljamo s pomočjo ukazov SMS, da pa si vseh ukazov ni potrebno zapomniti, nam pomaga pomožna aplikacija, ki služi kot uporabniški vmesnik in s pomočjo katere upravljamo aplikacijo. Ogled lokacij, ki so bile poslane na strežnik, je mogoč v pomožni aplikaciji.

Development of the mobile and web application Locator

Key words: Android application, Mobile application, location, GPS

UDK: **621.397.7-026.26(043.2)**

Abstract

The aim of this thesis is to create Android application Locator that allows parents or guardians to easily control their children. The application is made of two parts. From the primary application running in background on the child's phone and intermittently sending the child's location to the server or to the mobile phone via SMS. The primary application is managed by SMS commands. The secondary application serves as user interface through which we control the primary application. Secondary application is used to view the locations that were sent to the server.

Kazalo vsebine

1. Uvod ... 1

2. Uporabljene tehnologije ... 3

2.1 Android ... 3

2.2 Java ... 4

2.3 HTML ... 4

2.4 CSS ... 5

2.5 JavaScript ... 5

2.6 jQuery ... 6

2.7 PHP ... 6

2.8 AJAX ... 6

2.9 JSON ... 6

2.10 OpenStreetMaps ... 7

2.11 Osmdroid ... 8

3. Uporabljena orodja ... 9

3.1 Android Studio ... 9

3.2 Notepad++ ... 9

3.3 FileZilla ... 9

3.4 XAMPP ... 9

4. Ideja ... 10

4.1 Pregled že obstoječih aplikacij ... 10

4.1.1 Opis pregledanih aplikacij ... 10

4.1.2 Primerjave pregledanih aplikacij ... 13

4.2 Načrt ... 15

4.2.1 Predvidene funkcionalnosti ... 15

4.2.2 Deljenje s spletnim strežnikom 15
4.2.3 Deljenje s pomočjo sporočila SMS 16
4.2.4 Nastavljanje aplikacije preko sporočila SMS 16
5. Implementacija 17
5.1 Prijava in nastavitve 17
5.2 Pridobivanje pravic 18
5.3 Pošiljanje sporočil SMS 18
5.4 Prejemanje in obdelava sporočil SMS 19
5.5 Alarmi 21
5.6 Pridobivanje lokacije 22
5.7 Pošiljanje podatkov na strežnik 22
5.8 Podatkovna baza 23
5.9 Testiranje 24
6. StrežniškI del 24
6.1 Registracija 24
6.2 Registracija z že obstoječim računom 25
6.3 Vstavljanje prejete lokacije v bazo 26
7. Spletni del 26
7.1 Uporabniški vmesnik 26
7.2 Prijava 27
7.3 Prikaz lokacij 27
8. POMOŽNA aplikacija 29
8.1 Prijava 29
8.2 Vnos številke ter gesla za prejem 29
8.3 Vmesnik z zavihki 30

8.4 Fragment SMS 31

8.5 Fragment WEB 32

8.6 Fragment MAP 33

9. Sklep 35

10. Viri 37

Kazalo slik

Slika 2.1: Android arhitektura 4
Slika 2.2: OpenStreetMap 7
Slika 2.3: Osmdroid izgled mape 8
Slika 5.1: Nastavitve in prijava 17
Slika 5.2: Dialog za pridobivanje pravic 18
Slika 5.3 Metoda za pošiljanje sporočil 19
Slika 5.4 izsek kode kjer preverjam če je prispeli SMS namenjen aplikaciji 19
Slika 5.5 Izsek kode kjer nastavim alarm za pošiljanje na strežnik 21
Slika 5.6 Metoda za preverjanje internetnega dostopa 21
Slika 5.7 Izsek kode kjer zahtevamo lokacijo ter preverjamo status naprave 22
Slika 5.8 Izsek kode kjer sestavim vsebino sporočila SMS 22
Slika 5.9 Obvestilo, ki se prikaže ob komunikaciji z bazo 22
Slika 5.10: Sestava objekta JSON, ki ga pošljemo metodi za komunikacijo s strežnikom ... 23
Slika 5.11: Izsek kode kjer izvedem http zahtevo na strežnik 23
Slika 5.12 Relacije v bazi 24
Slika 6.1: PHP koda za registracijo 25
Slika 6.2: PHP koda za pridobivanje že obstoječega uporabnika 25
Slika 6.3: Vstavljanje lokacije v bazo 26
Slika 7.1: Koda naslovne strani index.html 27
Slika 7.2: Lokacije, prikazane na spletni strani 28
Slika 9.1: Prikaz AlertDialoga 30
Slika 9.2: Nastavitve parametrov AlertDialoga 30
Slika 9.3 Ustvarjanje Vmesnika z zavihki 31
Slika 9.4: Izgled fragmenta SMS 32
Slika 9.5: Izgled fragmenta WEB 33
Slika 9.6: MapView v Layout-u 34
Slika 9.7: Inicializacija MapView-a 34

Kazalo tabel

Tabela 4.1: Primerjava lastnosti aplikacij 13

Tabela 4.2: Primerjava plačljivih z brezplačnimi..... 13

Tabela 4.3: Moje ocene aplikacij 14

Uporabljene kratice

HTML – Hyper Text Markup Language

CSS – Cascading Style Sheets

JS – Java Script

XML – eXtensible Markup Language

AJAX – Asynchronous JavaScript And XML

JSON – JavaScript Object Notation

OSM – Open Street Maps

IDE – Integrated Development Environment

SFPT – Secure File Transfer Protocol

OS – Operacijski sistem

1. UVOD

V vsakdanjem življenju nas tehnologija obdaja že skoraj na vsakem koraku, prav tako ima skoraj vsakdo pametni telefon z zaslonom na dotik. Mobilni telefoni so zelo priročni, saj so zelo majhni in jih lahko imamo ves čas pri sebi. Edina negativna lastnost današnjih mobilnih telefonov je zmogljivost baterije, ki pa se zadnje čase izboljšuje. V današnjem svetu obstaja vse več razvijalcev, ki razvijajmo različne aplikacije z namenom, da nam pomagajo pri vsakodnevnih opravilih. Eden izmed vodilnih operacijskih sistemov na mobilnih telefonih je Android operacijski sistem. Ker se tehnologija zelo hitro razvija, je potrebno ves čas slediti novim tehnologijam.

Tudi najmlajši otroci velikokrat uporabljajo pametne telefone. Večinoma za igranje iger, kar starši dandanes dojemajo kot težavo, saj so nekateri otroci z igranjem iger že zasvojeni. Vendar se lahko pri otrocih pametni telefoni uporabljajo tudi v koristne namene, kamor lahko prištevamo učenje barv, jezika, geografije, abecede itd., in sicer prav s pomočjo raznoraznih aplikacij. Prav tako lahko s pomočjo telefona spremljamo, kje se otroci nahajajo. Izdelava aplikacije, ki bi omogočala spremljanje otroka bo tudi cilj pričujoče diplomske naloge.

Prikazali bomo izdelavo aplikacije, ki je lahko nameščena na otrokovem telefonu in intervalno pošilja lokacijo na strežnik ali na mobilni telefon v obliki sporočil SMS. Aplikacijo je mogoče nastavljati preko ukazov SMS ali s pomočjo uporabniške aplikacije, pri kateri parametre ukazov nastavimo preko uporabniškega vmesnika, v ozadju pa se pošlje SMS z ustreznim ukazom. Do aplikacije ima dostop samo starš, ne pa tudi otrok, ki prav tako ne bo mogel zaustaviti delovanja aplikacije. Lokacije, ki bodo poslane v obliki sporočil SMS, bodo podale povezave do Google Mapsa, kjer bo prikazana lokacija. Poleg koordinat lokacije se pošlje tudi trenutna hitrost.

V spletni aplikaciji so v seznamu prikazane lokacije, poslane na strežnik. Te lokacije so razvrščene po datumu. Poleg same povezave do strani Google Maps z lokacijo je v seznamu tudi hitrost in ponudnik, preko katerega smo pridobili lokacijo. Ponudnik je lahko GPS ali NETWORK. Takrat, ko GPS ni na voljo, se lokacija pridobi preko NETWORK ponudnika. Hitrost je prikazana le, če je ponudnik GPS, s katerim dosežemo tudi večjo natančnost. V spletno aplikacijo se je potrebno prijaviti z uporabniškim imenom in geslom.

Uporabniška aplikacija služi kot nekakšen vmesnik, kar pomeni, da se ni potrebno učiti ukazov za upravljanje aplikacije na daljavo. V uporabniški aplikaciji nastavimo parametre, s pomočjo katerih se potem sestavi primerno sporočilo SMS z ukazom, ki se pošlje v ozadju. Uporabimo pa jo lahko tudi za pregled lokacij, ki so bile poslane na strežnik in so prikazane kot pini na zemljevidu.

V diplomski nalogi najprej opišemo uporabljene tehnologije in orodja. Nato predstavimo izdelavo primarne Android aplikacije, ki je osrednji del diplomske naloge. V nadaljevanju bomo opisali še izdelavo spletne aplikacije ter dodali dokumentacijo. Na koncu sledi opis izdelave uporabniškega dela aplikacije.

2. UPORABLJENE TEHNOLOGIJE

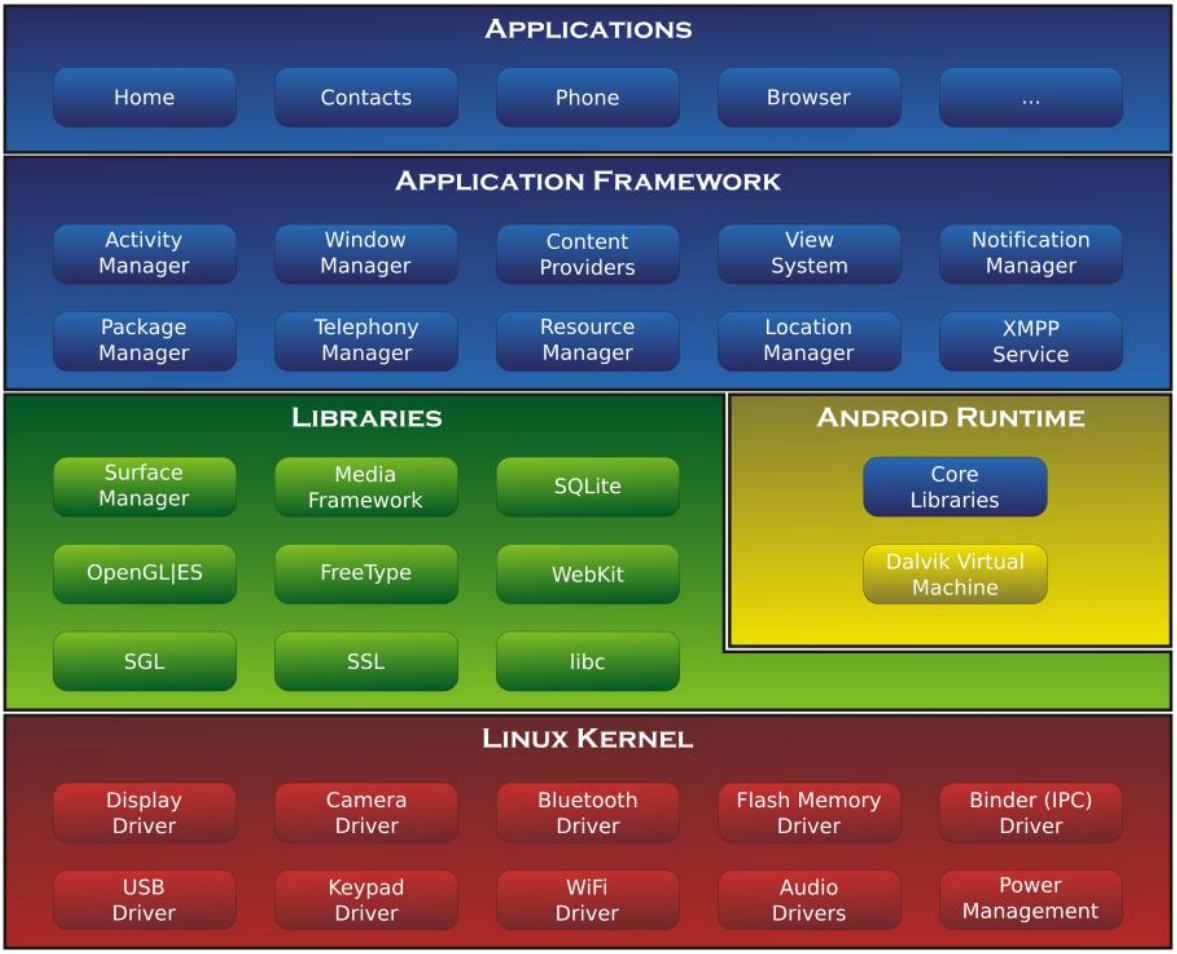
Aplikacija je narejena za Andoid telefone, ki temeljijo na programskem jeziku Java. Spletna aplikacija je narejena z orodji HTML, CSS, JavaScript, komunikacija z bazo pa z jezikom PHP. V uporabniško aplikacijo je vgrajen zemljevid OSM.

2.1 Android

Android je odprtokodna platforma za razvoj mobilnih aplikacij, ki ga je Google odkupil leta 2005. Od prve verzije iz leta 2008 do današnje verzije (verzija 7.0 »Nougat«) şe je platforma zeli izboljšala. Temelji na direktni interakciji z uporabo gest in potez, kot so npr. dotik, poteg in podobno [1].

Zgradba temelji na Linux jedru in je bila primarno zasnovana za mobilne telefone na dotik. Sedaj se uporablja tudi na mnogih drugih področjih, kot so televizije, ure itd. Arhitektura OS je sestavljena iz petih komponent. Zgradbo prikazuje Slika 2.1 [2].

- Od verzije 4.4 se za prevajanje kod uporablja JIT (Just in time compiler).
- Knjižnice se uporabljajo za dostop do strojnih komponent naprave, do njih uporabnik nima dostopa.
- Aplikacijsko ogrodje (ang. »Framework«) je tisti del arhitekture, v katerem se nahajajo sistemske aplikacije.
- Aplikacije imajo končnico .apk. Ko aplikacijo zapremo, se zapre tudi proces, kar omogoča optimalno rabo pomnilnika.



Slika 2.1: Android arhitektura

Vir: https://en.wikipedia.org/wiki/Android_%28operating_system%29#Software_stack

2.2 Java

Programski jezik Java je postopkoven programski jezik. Zanj skrbi podjetje Oracle – Sun Microsystems, ki ga vzdržuje in posodablja. Z Javo lahko programiramo tudi aplikacije na mobilnih in pametnih telefonih [3]. Jave ne smemo zamenjevati z jezikom JavaScript, ki ima zgolj podobno ime.

2.3 HTML

HTML je namenjen izdelavi spletnih strani. Razvil ga je Tim Berners-Lee, ko je delal v CERN-u [4]. HTML nam pove, kako naj se spletna stran prikaže glede na zaporedje ukazov oz. značk. Pred značko se vedno nahaja znak <, za značko pa >, kot npr. <značka>. Za

prikaz spletne strani potrebujemo spletni brskalnik. To je program, ki bere ukaze v jeziku HTML in glede na to prikazuje vsebino. V HTML-u obstajata dve vrsti značk, in sicer samostojne (<značka>) ter začetne in končne (<značka> in </značka>) [5].

Za pisanje HTML dokumentov ne potrebujemo posebnega orodja. Pišemo lahko v katerem koli urejevalniku besedil, vendar nam uporaba orodja, namenjenega pisanju HTML-a, zelo olajša delo.

2.4 CSS

CSS oz. Cascading Style Sheets je mehanizem, ki pove spletnemu brskalniku, kako naj prikaže HTML dokument. S skupnimi močmi sta ga izdala Håkon Wium in Bert Bos.

CSS strukturira vsebino HTML dokumenta in ji da želeno obliko. S pomočjo CSS-a omejimo ponavljanje kode. Značkam lahko določimo razrede ter razrede oblikujemo. CSS nam omogoča določanje poravnave, barve pisav, odmikov na straneh, obrob, pozicije elementov ter veliko drugih dejavnosti [6].

2.5 JavaScript

JavaScript (JS) je programski jezik. Večinoma se uporablja za izdelavo spletnih strani, nekoliko pa tudi drugje. Skripte lahko vgradimo ali vključimo v HTML strani ter tako naredimo stran bolj dinamično. Tako lahko z malo truda svoje strani izboljšate in olepšate.

Izvedba JS-ja je neodvisna od gostitelja, saj nima vgrajenih vhodno-izhodnih funkcij. JS je odprt jezik, kar pomeni, da ga lahko uporablja vsakdo, brez da bi potreboval licenco [7].

JS ima veliko podobnih lastnosti kot programski jezik Java, vendar ni »prevedena Java«. Osnovna sintaksa je precej podobna programskemu jeziku C. Izvaja se na odjemalčevi strani [8].

2.6 jQuery

jQuery je preprosta, a zmogljiva JavaScript knjižnica. Pripravil jo je John Resig, prvič pa je bila objavljena januarja 2006. Knjižnica je prosto dostopna in odprtokodna JavaScript knjižnica, ki je izdana pod MIT licenco.

jQuery uporabljamo za hitrejšo in enostavnejšo navigacijo po dokumentu, upravljanje dogodkov in še veliko več. Namen knjižnice je pisanje krajše in bolj pregledne kode, saj je njegova filozofija napiši manj, naredi več [9].

2.7 PHP

PHP je zelo razširjen skriptni jezik, ki se uporablja za razvoj spletnih strani in ga je razvil danski programer Lerdorf leta 1995.

Najbolj je podoben jezikoma C in Perl [10]. PHP kodo lahko v HTML dokument vključimo preko ločene datoteke, lahko pa pišemo kar v HTML dokumentu med značkama <?php in ?>. Uporabili smo ga za komunikacijo s podatkovno bazo.

2.8 AJAX

AJAX oz. Asinhroni JavaSvript in XML se uporabljata za prenos podatkov med spletnim brskalnikom in strežnikom. S pomočjo AJAX-a dosežemo dinamično osveževanje dela spletne strani, kar pomeni, da se osveži samo del spletne strani in ne celotna stran [11]. Navadno se podatki izmenjujejo v formatu JSON.

2.9 JSON

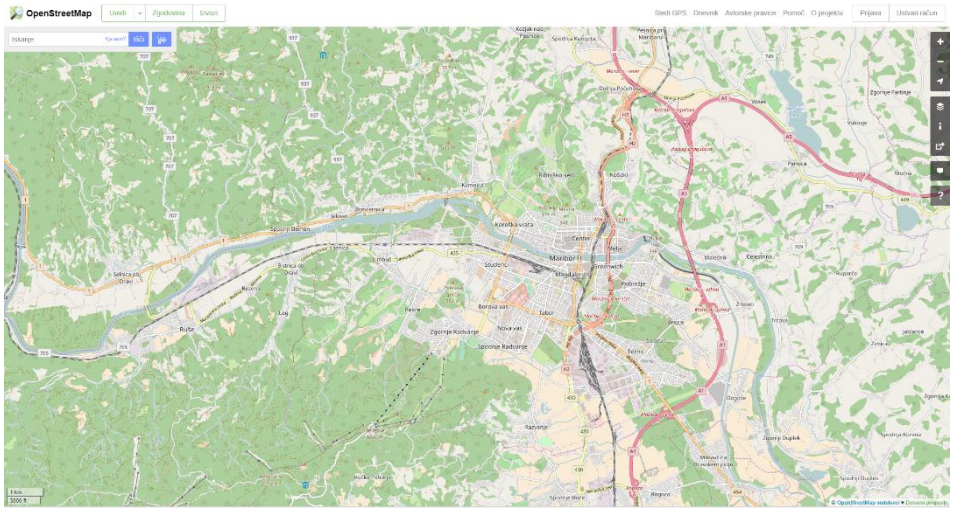
JSON je tekstovni format, ki je enostaven za branje in pisanje. Idealen je za izmenjavo podatkov, saj se v vseh programskih jezikih predstavlja enako [12].

2.10 OpenStreetMaps

OpenStreetMaps (OSM) je projekt, ki ga je leta 2004 ustanovil Steve Coast. Pristop je podoben kot pri Wikipediji, saj lahko vsi dodajamo vsebino. Torej lahko celotna skupnost po celem svetu vstavlja in vzdržuje podatke o cestah, poteh, restavracijah, avtobusnih postajah in še veliko več. Sodelujočih je iz dneva v dan več, s čimer se skupnost veča in zemljevid postaja vedno boljši. Danes OSM uporablja že na tisoče spletnih strani in mobilnih aplikacij.

OSM temelji na lokalnem znanju sodelujočih, ki uporabljajo zračne posnetke, GPS naprave in terenske karte ter s tem zagotavljajo, da je OSM ves čas natančen in posodobljen. V skupnosti lahko najdemo tudi dnevnike uporabnikov, bloge itd.

OSM so prosto dostopni podatki, ki jih lahko uporabljamo v različne namene, dokler zasluge pripisujemo OSM-u in sodelujočim na projektu. OSM upravlja fundacija OpenStreetMap v imenu skupnosti oz. vseh sodelujočih. Partnerji, ki podpirajo gostovanje, pa so: Center UCL VR, Impreial Collage, Bytemark Hosting ter ostali. Na sliki Slika 2.2 vidimo OSM zemljevid [13].



Slika 2.2: OpenStreetMap
Vir: https://www.openstreetmap.org

2.11 Osmdroid

Glavni avtor knjižnice Osmdroid je Nicolas Gramlich. Knjižnica omogoča delo z OpenStreetMaps in je brezplačna nadomestitev za Android MapView. Januarja 2016 je izšla različica 5.1. Na Google Play je tudi aplikacija Osmdroid, ki demonstrira uporabo Osmdroid knjižnice (Slika 2.3) [14].

Slika 2.3: Osmdroid izgled mape

Vir: https://play.google.com/store/apps/details?id=org.osmdroid&hl=en

3. UPORABLJENA ORODJA

Primarno orodje za izdelavo aplikacije je Andorid, v katerem se razvija večina Android aplikacij. Spletni del aplikacije je napisan v programu Notepad++. Datoteke smo na strežnik naložili s programom FileZilla.

3.1 Android Studio

Android Studio je uradno razvijalno okolje za razvijanje Android aplikacij. Je prosto dostopno programsko orodje. Temelji na IntelliJ IDEA. Poleg odličnih urejevalnikov kod in razvijalnih orodij nam ponuja še veliko drugih funkcionalnosti.

3.2 Notepad++

Notepad je urejevalnik besedil, ki ga lahko uporabljamo tudi za pisanje HTML-a, CSS-a, JavaScripta, PHP-a in drugih kod. Je eno izmed najosnovnejših orodij za pisanje kode (nima samodejnega zapolnjevanja besed), vendar je za naše potrebe dovolj zmogljiv.

3.3 FileZilla

Je programsko orodje, preko katerega lahko s pomočjo grafičnega vmesnika nalagamo datoteke na strežnik. Datoteke smo preko SFTP-a naložili na zakupljen prostor na strežniku.

3.4 XAMPP

V procesu razvijanja aplikacije smo PHP izvajali na lokalnem strežniku, in sicer s pomočjo razvijalnega okolja XAMPP. XAMPP je najbolj popularno brezplačno razvijalno okolje za PHP, preko katerega je enostavno namestiti strežnik Apache in MySQL podatkovno bazo. PhpMyAdmin se uporablja za kreiranje baze in tabel v bazi. Ima še nekaj drugih funkcionalnosti, ki pa jih nismo potrebovali.

4. IDEJA

Ideja projekta je zajemala izdelavo Android aplikacije, ki bi avtomatično sporočala koordinate telefona na strežnik ali na mobilni telefon v obliki sporočila SMS, nastavljati pa bi jo bilo mogoče tudi s pomočjo sporočila SMS v smislu komand in izvrševanja. Ideja se nam je zdela zelo zanimiva, saj je takšna aplikacija zelo uporabna, sploh v današnjih časih, ko ima že skoraj vsakdo pametni telefon.

4.1 Pregled že obstoječih aplikacij

Najprej je bilo potrebno narediti temeljit pregled aplikacij (Junij 2017), ki že obstajajo, nato pa smo se lahko odločili, ali je aplikacijo sploh smiselno razvijati. V nadaljevanju bomo na kratko opisali aplikacije, ki smo jih v ta namen našli na Marketu (Send my location, Send my GPS location, Moja lokacija na zemljevidu, My location, Glympse – Share my location, Moje GPS koordinate, My location #2, Show my location, MyPosition, Share my position, Mobile location tracker, My location #3, Upravitelj naprav Android), in za vsako naredili tudi seznam njenih funkcionalnosti. Naredili smo tudi več primerjav, kot je na primer primerjava lastnosti aplikacij (Tabela 4.1), primerjava brezplačnih aplikacij s plačljivimi (Tabela 4.2) in primerjava najbolje ocenjenih aplikacij. Na koncu smo podali še svoje ocene aplikacij (Tabela 4.3). Prišli smo do sklepa, da je aplikacijo smiselno razviti.

4.1.1 Opis pregledanih aplikacij

- Send my location: Aplikacija je namenjena seznanjanju želenih ljudi z lokacijo, kjer se trenutno nahajamo, in sicer brez nepotrebnega besednega sporazumevanja, pač pa samo s posredovanjem lokacije. Lokacijo lahko pošljemo v obliki sporočila SMS ali e-maila, lahko pa tudi preko družbenih omrežij (Facebook, Twiter) ali sporočilnih storitev (WhatsApp, Skype). Lokacijo moramo poslati sami, ročno. Aplikacija nima možnosti pošiljanja na časovne intervale. Aplikacija je zastonj in ima oceno 4,1, podalo pa jo je 568 uporabnikov [15].

- Send my GPS location: Aplikacija je namenjena odkrivanju trenutne lokacije in naslova ter deljenju koordinat s prijatelji v obliki sporočila SMS, e-maila, Facebooka, Twitterja in podobnega. Omogoča tudi pretvarjanje GPS koordinat med dvema formatoma. Zemljevid ni vgrajen, pač pa se ob kliku na zemljevid odpre Google Zemljevid. Omogoča nastavljanje intervala za osveževanje lokacije, ki jo lahko pošljemo samo ročno. Aplikacija je zastonj in ima oceno 4,2, podalo pa jo je 262 uporabnikov [16].
- Moja lokacija na zemljevidu: S pomočjo aplikacije lahko izvemo naš trenutni naslov in lokacijo ter koordinate delimo s prijatelji in družino tako, da lahko vidijo, kje smo. Shranimo lahko svoje priljubljene kraje ter poiščemo naslov, kamor želimo priti. Omogoča tudi Street View, če imamo primerno verzijo Androida. Aplikacija je plačljiva in ocenjena z oceno 4,2 od skupaj 19.138 ocen [17].
- My location: Aplikacija te obvešča o trenutni lokaciji ter naslovu, če je na voljo internet. Lokacijo lahko delimo preko sporočila SMS, e-maila in družbenih omrežij. Omogoča tudi najdbo telefona v primeru izgube, in sicer tako, da pošljemo sporočilo SMS z določenim geslom, nato pa dobimonazaj trenutno lokacijo telefona. To pomeni, da lahko aplikacijo aktiviramo preko drugega telefona z sporočilom SMS ali ročno. Aplikacija je plačljiva, ocenjena je z oceno 4,2, podalo pa jo je 466 uporabnikov [18].
- Glympse – Share GPS location: Aplikacija omogoča enostavno deljenje lokacije z družino in prijatelji. Deli se lahko v obliki sporočila SMS, e-maila, družbenih omrežij. Zraven se lahko doda tudi sporočilo, poleg tega pa lahko na zemljevidu označiš, kam si namenjen. Nastavlja se lahko tudi dolžina časa deljenja tvoje lokacije z drugimi. Možno je shraniti priljubljene lokacije. Aplikacija beleži tudi zgodovino lokacij, ki jih je uporabnik prejel in poslal. Aplikacija ni plačljiva, ocenjena je z oceno 4,5, ki jo je podalo 95.322 uporabnikov [19].
- My GPS coordinates: Aplikacija omogoča preprosto deljenje GPS lokacije v obliki sporočila SMS, e-maila in družbenih omrežij. Prejemnik prejme povezavo do Google Zemljevida s trenutno lokacijo. Zemljepisna širina in dolžina sta prikazani v decimalnem formatu in formatu s stopinjami. Izpisana je tudi točnost signala in

kdaj je bil prejet zadnji signal. Aplikacija je zastonj, ocenjena je z oceno 4,2, ki jo je podalo 3.657 uporabnikov [20].

- Show my location: Aplikacija nam prikaže zemljepisno širino in dolžino ter naslov trenutnega nahajališča. Naslov lahko delimo z drugimi, in sicer v obliki sporočila SMS, e-maila in Facebooka. Aplikacija je zastonj in je ocenjena z oceno 4,2, ki jo je podalo 26 uporabnikov [21].
- Share my position: Enostavna aplikacija za deljenje trenutne lokacije v obliki sporočila SMS, e-maila in družbenih omrežij. Pošlje se povezava do Google Zemljevida z našimi koordinatami. Aplikacija deluje preko GPS podatkov, internetna povezava pa ob tem ni potrebna. Aplikacija je brezplačna in je ocenjena z oceno 4,2, ki jo je podalo 279 uporabnikov [22].
- Mobile location tracker Aplikacija spremlja našo lokacijo in shranjuje lokacije v Location History. Trenutno lokacijo lahko delimo v obliki sporočila SMS, e-maila in družbenih omrežij. Lokacija se pošlje kot povezava do Google zemljevida z našimi koordinatami. Omogoča tudi, da najdemo pot od točke do točke ali določen naslov. Podpira več oblik zemljevida (običajna, hibridna, satelitska, reliefna). Aplikacija je brezplačna in ima 128.993 ocen s povprečjem 4,1 [23].
- My location #2: Aplikacija omogoča, da preko GPS-a ali Mobilnih podatkov/WiFi-ja najdemo lokacijo, kjer se nahajamo. Lokacijo lahko delimo v obliki sporočila SMS, e-maila in družbenih omrežij. Pošlje se naslov, povezava do Google zemljevida ter koordinate. Ima dva pogleda zemljevida (navaden ali satelitski). Ocenjena je z oceno 4,1, ki jo je podalo3.684 uporabnikov. Je brezplačna [24].
- Poišči mojo napravo: Aplikacija omogoča, da s pomočjo spletne aplikacije sledimo telefonu, vedeti moramo le geslo in uporabniško ime. Namen aplikacije je najdba telefona v primeru izgube le-tega in sledenje telefonu. Aplikacija je brezplačna, ima 441.974 ocen, katerih povprečje znaša 4,3 [25].

4.1.2 Primerjave pregledanih aplikacij

Tabela 4.1 prikazuje primerjavo lastnosti vseh aplikacij, Tabela 4.2 pa prikazuje funkcionalnosti brezplačnih aplikacij s funkcionalnostmi plačljivih aplikacij. Naše ocene aplikacij prikazuje Tabela 4.3.

Tabela 4.1: Primerjava lastnosti aplikacij

Lastnost / ***Aplikacija***	**Cena**	**Število namestitev**	**Ocena**	**Število ocen**
Send my location	0 €	100.000–500.000	4,1	568
Send my GPS location	0 €	50.000–100.000	4,2	262
Moja lokacija na zemljevidu	1,19 €–2,99 €	1.000.000–5.000.000	4,2	19.138
My location	1,09 €	100.000–500.000	4,2	466
Glympse – Share my location	0 €	5.000.000–10.000.000	4,5	95.322
Moje GPS koordinate	0 €	1.000.000–5.000.000	4,2	3.657
Show my location	0 €	1.000–5.000	4.2	26
Share my position	0 €	50.000–100.000	4,2	279
Mobile location tracker	0 €	10.000.000–50.000.000	4,1	122.993
My location #2	0 €	100.000–500.000	4,1	3.684
Poišči mojo napravo	0 €	10.000.000–50.000.000	4,3	441.974

Tabela 4.2: Primerjava plačljivih z brezplačnimi

Aplikacija / ***Funkcionalnost***	**Moja lokacija na zemljevidu (plačljiva)**	**My location (plačljiva)**	**Send my location (brezplačna)**	**Glympse – Share GPS location (brezplačna)**
Deljenje lokacije	✓	✓	✓	✓
Intervalno deljenje lokacije	X	X	X	X

Oddaljena aktivacija	✗	✓	✗	✗
Deljenje dlje časa	✗	✗	✗	✓
Prikaz trenutne lokacije na zemljevidu	✓	✓	✓	✓
Število podprtih pogledov zemljevida	3+Street wiew	?	2	3+prikaz prometa
Iskanje naslova	✓	✗	✗	✗
Shranjevanje priljubljenih/zgodovine	✗	✗	✗	✓

Tabela 4.3: Moje ocene aplikacij

Aplikacija	Ocena	Komentar
Glympse – Share my location	4,4	Aplikacija je zelo uporabna, saj se lahko nastavi čas, kar pomeni, da se lahko določi časovna omejitev spremljanja lokacije. Podpira tudi vse vrste zemljevidov. Ne moremo je aktivirati preko sporočila SMS.
My location #2	4,2	Ne omogoča oddaljenega aktiviranja in avtomatskega pošiljanja. Omogoča prikaz hitrosti, poleg tega pa ima veliko izbiro nastavitev.
Mobile location tracker	4,1	Ne omogoča oddaljenega aktiviranja in avtomatskega pošiljanja. Omogoča iskanje naslova, ki ga uporabnik označi na zemljevidu, in iskanje poti med točkama.
Send my location	3,8	Ne omogoča oddaljene aktivacije, niti ne pošiljanja dlje časa. Sicer je aplikacija uporabna za deljenje lokacije s prijatelji.
Show my location	3,8	Nima vgrajenega zemljevida. Pri naslovu se izpiše tudi poštna številka. Če se deli preko Facebooka, se lahko doda tudi sliko.
Send my GPS location	3,5	Nima vgrajenega zemljevida, ne omogoča oddaljene aktivacije, pošiljanja dlje časa. Vgrajen ima pretvornik med sistemoma.
Share my position	3,4	Nima vgrajenega zemljevida, izpišeta se samo zemljepisna širina in dolžina.
Moje GPS koordinate	3,4	Nima vgrajenega zemljevida, izpišejo se samo

		koordinate. Zelo slab prevod.
Moja lokacija na zemljevidu	/	Aplikacije nismo namestili, ker je plačljiva.
My location	/	Aplikacije nismo namestili, ker je plačljiva. V opisu piše, da se lahko aplikacija aktivira preko sporočila SMS.

4.2 Načrt

oNa podlagi ugotovitev smo se odločili, da je aplikacijo smiselno izdelati. Sledil je korak priprave načrta o izgledu aplikacije in o fukcionalnostih, ki naj bi jih omogočala.

4.2.1 Predvidene funkcionalnosti

- Deljenje lokacije s spletnim strežnikom.
- Deljenje lokacije preko sporočila SMS.
- Nastavljanje aplikacije preko sporočila SMS.
- Vnos telefonskih številk, ki lahko aplikacijo nastavljajo preko sporočila SMS.
- Izvajanje v ozadju.
- Nastavljanje intervala pošiljanja.
- Možnost pošiljanja povprečne hitrosti.

4.2.2 Deljenje s spletnim strežnikom

Aplikacija bi se ves čas izvajala v ozadju in bi se začela izvajati takoj, ko se telefon vklopi. Ko bi dobila informacijo, da je na voljo internetni dostop, bi na strežnik začela pošiljati trenutne koordinate. Interval pošiljanja bi se nastavil v nastavitvah aplikacije, do katerih bi imel dostop le tisti, ki je aplikacijo namestil, uporabnik telefona pa ne (morda zaščita z geslom, ki se ga določi ob namestitvi). Pošiljanje na strežnik bi bilo mogočenastavljati tudi preko sporočila SMS (vklop/izklop, interval, pošiljanje hitrosti).

4.2.3 Deljenje s pomočjo sporočila SMS

Pri deljenju s pomočjo sporočila SMS bi bilo najprej potrebno vnesti telefonske številko/-e, s katere lahko prihajajo ukazi in kamor se pošiljajo sporočila SMS z lokacijo. Številke se bi ponovno vnašale v nastavitvah aplikacije, do katerih bi imel dostop upravitelj aplikacije. V nastavitvah bi lahko nastavili tudi privzete vrednosti (interval, pošiljanje povprečne hitrosti uporabnika), da bi se lahko potem poslal ukaz za začetek pošiljanja. Aplikacija bi ves čas preverjala, ali je z določene telefonske številke prišlo sporočilo SMS z ukazom, zatem bi aplikacija začela pošiljati sporočila SMS z GPS koordinatami.

4.2.4 Nastavljanje aplikacije preko sporočila SMS

Aplikacijo bi bilo mogoče nastavljati preko drugega telefona s pomočjo ukazov preko sporočila SMS. Po pošiljanju določenega ukaza bi aplikacija začela oddajati lokacijo. V ukazu bi lahko aktivirali aplikacijo s privzetimi nastavitvami, ob tem bi lahko dodali parametre, kot so interval pošiljanja in hitrost premikanja. Lahko bi poslali ukaz za enkraten SMS s trenutno lokacijo. Omenjen ukaz bi se lahko uporabil tudi z namenom končanja pošiljanja sporočil SMS. Na začetku ukaza bi morala stati ključna beseda, s katero bi aplikaciji sporočili, da gre za SMS z ukazom (npr. Where he is – WHI). Primer oblike ukaza za aktivacijo: KLJUČNA BESEDA – NASTAVITVE ZA (sms ali web) - VKLOP/IZKLOP – INTERVAL-POŠILJANJE HITROSTI. Primer ukaza: WHI-SMS-START-5-KM (nastavljaj sms, začni pošiljanje, vsakih 5 minut, pošiljaj tudi hitrost).

5. IMPLEMENTACIJA

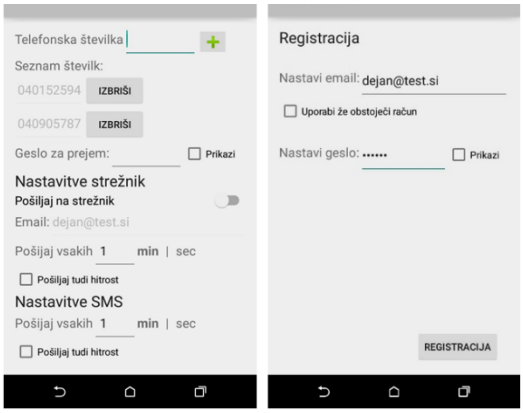
V tem poglavju bo prikazana implementacija mobilne aplikacije Locator. Vsak del aplikacije bo podrobno opisan, ob tem bodo dodane tudi slike kod.

5.1 Prijava in nastavitve

Najprej smo pripravili osnutek izgleda aplikacije, vendar ta še zdaleč ni bil enak sedanjemu izgledu. Treba je bilo dodati mnogo elementov, na katere na začetku nismo pomislili. V Android Studiu je lahko dodajanje elementov na grafični vmesnik dokaj enostavno in poteka po principu »potegni in spusti«, kar pomeni, da samo povlečemo elemente, ki jih želimo imeti na vmesniku.

Na aktivnost (Activity) prijave smo dodali elemente, ki bodo omogočali registracijo in prijavo, torej vnosno polje za uporabniško ime in geslo ter gumb za prijavo. Po prijavi se odpre aktivnost Nastavitve.

Nastavitve vsebujejo možnost dodajanja telefonskih številk, s katerih lahko aplikacija prejema sporočila SMS, in gesla, s katerim lahko s katere koli številke prejmemo sporočilo SMS in ga obdelamo. Nastavimo lahko tudi privzete nastavitve za sporočila SMS (interval, enota intervala, pošiljanje hitrost) kot tudi privzete nastavitve za pošiljanje na strežnik. V nastavitvah lahko vklopimo tudi pošiljanje na strežnik, ne moremo pa vklopiti pošiljanja prek sporočila SMS. To se lahko vklopi samo preko sporočila SMS. Oba grafična vmesnika prikazuje Slika 5.1.



Slika 5.1: Nastavitve in prijava

5.2 Pridobivanje pravic

Preden lahko začnemo dostopati do uporabnikovega telefona, moramo pridobiti ustrezne pravice. Nekatere, manj občutljive pravice se lahko dodajo v datoteko AndroidManifest. Tiste, ki so bolj občutljive narave, pa je od API levela 23 potrebno zahtevati v času izvajanja. Šele ko pridobimo pravice, lahko z uporabnikovega telefona pošiljamo sporočila SMS ter dostopamo do lokacije. Ko želimo npr. poslati sporočilo SMS, se uporabniku pokaže dialog Slika 5.2), kar pomeni, da z metodo onRequestPremissionsResult() po pridobitvi pravic pošljemo SMS.

Slika 5.2: Dialog za pridobivanje pravic

5.3 Pošiljanje sporočil SMS

Naslednji korak je zajemal izdelavo razreda, v katerem se bodo pošiljala sporočila SMS. Za ta razred smo potrebovali pravico SEND_SMS. Odsek kode za pošiljanje sporočil SMS prikazuje Slika 5.3. Ustvarili smo dve metodi: eno za pošiljanje napake in drugo za pošiljanje koordinat. Obe metodi prejmeta dva parametra, ki sta vsebina sporočila in številka, na katero naj se sporočilo pošlje. Pri pošiljanju napake se pošlje opis napake (če na primer GPS ni vklopljen, se pošlje sporočilo »GPS ni vklopljen«). Sporočilo se pošlje s pomočjo SmsManeger in metode sendTextMessage. Razreda SmsManeger na fakulteti nismo spoznali, tako da nam je bilo raziskovanje omenjenega zelo zanimivo.

```
public void poslji_SMS(String vsebina, String telefonska_stevilka)
{
    try {
        SmsManager smsManager = SmsManager.getDefault();
        smsManager.sendTextMessage(telefonska_stevilka, null,vsebina, null, null);
        Log.d(TAG, "Koordinate poslane!");
    }
    catch (Exception e)
    {
        Log.d(TAG, "SMS ni poslan!");
    }
```

Slika 5.3 Metoda za pošiljanje sporočil

5.4 Prejemanje in obdelava sporočil SMS

Ustvarili smo BroadcastReciever, ki bo vsa prihodna sporočila prejemal na telefon. Če je intent.getAction() enak SMS_RECEVED, potem vemo, da je bilo prejeto sporočilo SMS. V tem primeru preberemo in shranimo sporočilo ter telefonsko številko pošiljatelja. Nato sporočilo razdelimo na podlagi znaka; če je na prvem mestu razdeljenega sporočila geslo, ki ga imamo nastavljenega v nastavitvah, potem odstranimo geslo ter obdelamo ostalo vsebino sporočila SMS; če na prvem mestu ni gesla, potem preverimo številko (kot prikazuje Slika 5.4), če je številka v seznamu številk, prav tako obdelamo vsebino sporočila SMS.

```
String[] razdeljenaVsebina= vsebinaSporocila.split("-");

if(razdeljenaVsebina[0].equals(SPMprejemnik.pridobiId("geslo_za_prejem")))
{
    String vsebinaSporocilaBrezGesla = "";
    for(int i=1; i<razdeljenaVsebina.length; i++)
    {
        vsebinaSporocilaBrezGesla+=razdeljenaVsebina[i]+"-";
    }
    vsebinaSporocilaBrezGesla = vsebinaSporocilaBrezGesla.substring(0, vsebinaSporocilaBrezGesla.length()-1);
    obdelajSMS(ctx, stevilkaBrez386, vsebinaSporocilaBrezGesla);
}
else
{
    String[] stevilke = SPMprejemnik.pridobiId("telefonska_stevilka").split("-");
    for (int i = 0; i < stevilke.length; i++) {
        if (posiljateljSporocila.equals("+386" + stevilke[i].substring(1, 9)))
        {
            obdelajSMS(ctx, stevilkaBrez386, vsebinaSporocila);
        }
    }
}
```

Slika 5.4 izsek kode kjer preverjam če je prispeli SMS namenjen aplikaciji

Za tem smo si zamislili zaporedje tekstovnih nizov, ki bodo pomenili nek ukaz. Odločili sem se, da bodo posamezni deli ukaza med seboj ločeni z znakom – (minus). Najprej moramo aplikaciji sporočiti, da je sporočilo namenjeno njej. Za ta del ukaza smo izbrali niz LOCATOR. Potem pride na vrsto ukaz, ki nam pove, ali je nastavitev za SMS pošiljanje koordinat ali za WEB pošiljanje koordinat, tako da sta SMS ali WEB naslednji ključni besedi. Če nato dodamo še ključno besedo START, to pomeni najosnovnejši ukaz, ki pomeni začetek pošiljanja bodisi preko sporočila SMS bodisi preko strežnika s privzetimi nastavitvami, kot so nastavljene v aktivnosti Nastavitve. Ko želimo začeti pošiljanje preko sporočila SMS s privzetimi nastavitvami, mora ukaz izgledati tako: LOCATOR-SMS-START (ne sme vsebovati presledkov, ločen je z minusi, lahko je napisan z velikimi ali malimi črkami).

Pri ostalih ukazih ključni besedi START dodamo še določene parametre, kot so interval pošiljanja, enota intervala in pošiljanje informacije o hitrosti. Če dodamo kakšnega od teh parametrov, se pri SMS pošiljanju ta parameter ne vzame iz privzetih nastavitev, ampak se upošteva parameter, napisan v sporočilu SMS. Seveda mora biti sintaksa pravilna. Naslednja ključna beseda je lahko interval ali zahteva za pošiljanje hitrosti. Interval lahko znaša najmanj 30 sekund in mora biti celo število. Če pošljemo ukaz LOCATOR-SMS-START-5, se bo SMS pošiljal vsakih 5 enot, ki so določene v privzetih nastavitvah. Tudi pošiljanje hitrosti se bo upoštevalo glede na privzete nastavitve. Za pošiljanje s privzetimi nastavitvami in s hitrostjo moramo poslati ukaz LOCATOR-SMS-START-KM. V tem primeru se vse nastavitve vzamejo iz privzetih nastavitev, razen hitrosti, ki jo tukaj eksplicitno določimo oz. določimo dejavnik, da se mora pošiljati tudi podatek o hitrosti. Pošljemo lahko tudi interval in zahtevo po pošiljanju hitrosti, pri čemer moramo biti pozorni, da na prvem mestu stoji interval, šele nato pa zahteva po pošiljanju hitrosti. Ta ukaz izgleda tako: LOCATOR-SMS-START-5-KM. Obstaja še nekaj daljših ukazov, ki so poleg že omenjenih opisani v dokumentaciji.

Pri pošiljanju v obliki sporočil SMS obstaja tudi poseben ukaz za enkratno pošiljanje lokacije, ki ima naslednjo obliko: LOCATOR-SMS-1X. Za prenehanje pošiljanja sporočil SMS se mora poslati sporočilo z ukazom LOCATOR-SMS-STOP. Ukazi za pošiljanje na strežnik so skoraj enaki, vendar mora na drugem mestu stati beseda WEB. Ukaz enkrat za strežnik ne obstaja. Pri strežniku se parametri, ki jih pošljemo, spremenijo tudi v privzetih nastavitvah, medtem ko se pri pošiljanju preko sporočil SMS ti parametri v privzetih nastavitvah ne spremenijo.

V prejemniku sporočil SMS se s pomočjo razreda AlarmManeger in metode setRepeating() ustvarjajo alarmi, ki se prožijo na čas, ki je določen z intervalom in enoto intervala. Slika 5.5 prikazuje del kode, kjer se nastavi alarm za pošiljanje na strežnik. Tukaj se alarmi izklapljajo oz. prekličejo z metodo cencel().

```
public void zacniIzvajanjeWeb(Context ctx, int interval, String stevilka, Boolean posiljajTudiHitrost)
{
    Long time = new GregorianCalendar().getTimeInMillis();

    Intent intentAlarm;
    intentAlarm = new Intent(ctx, LokacijaIzvajajWeb.class);
    intentAlarm.putExtra("posiljajTudiHitrost", posiljajTudiHitrost);
    intentAlarm.putExtra("stevilka", stevilka);

    alarmManagerWeb = (AlarmManager) ctx.getSystemService(Context.ALARM_SERVICE);
    alarmManagerWeb.setRepeating(AlarmManager.RTC_WAKEUP, time, interval, PendingIntent.getBroadcast(ctx, 1, intentAlarm, PendingIntent.FLAG_UPDATE_CURRENT));
}
```

Slika 5.5 Izsek kode kjer nastavim alarm za pošiljanje na strežnik

5.5 Alarmi

Ob sprožitvi alarma se kliče nov razred, ki razširja BroadcastReciever. Vse, kar se mora zgoditi, ko se alarm vklopi, mora biti napisano v metodi onRecieve(). Ustvarili smo dva tovrstna razreda: enega za pošiljanje sporočil SMS in drugega za pošiljanje na strežnik. V obeh razredih pridobimo lokacijo, ki jo pošljemo z izbrano metodo. Tukaj tudi preverjamo, če je GPS vklopljen in če obstaja internetna povezava za pošiljanje na strežnik (odsek kode prikazuje Slika 5.6).

```
public static boolean isOnline(Context ctx) {
    ConnectivityManager cm = (ConnectivityManager) ctx.getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo netInfo = cm.getActiveNetworkInfo();
    return netInfo != null && netInfo.isConnectedOrConnecting();
}
```

Slika 5.6 Metoda za preverjanje internetnega dostopa

5.6 Pridobivanje lokacije

Za pridobivanje lokacije je potrebno ustvariti LocationManegerja, preko katerega kličemo metodo requestLocationUpdates(), ki zahteva lokacijo (primer zahtevka prikazuje Slika 5.7). Kot parameter je potrebno poslati tudi LocationListenerja, ki ga ustvarimo pred klicem metode.

```
if (Utils.isOnline(ctx)) {
    if (locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER)) {
        try {
            locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 1000, 1, locationListener);
```

Slika 5.7 Izsek kode kjer zahtevamo lokacijo ter preverjamo status naprave

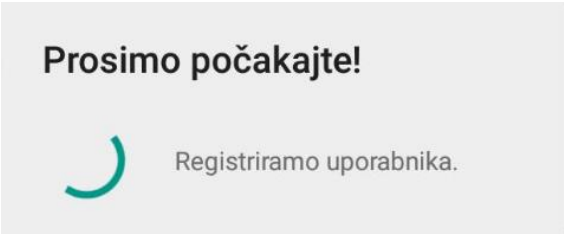
LocationListener ima metodo onLocationChanged, ki se kliče z metodo requestLocationUpdates in prejme objekt Location, ki ima metode getLatitude(), getLongitude() in getSpeed(). Te vrnejo geografsko širino, dolžino in hitrost, s pomočjo katerih sestavimo niz (kot prikazuje Slika 5.8), na katerega lahko prejemnik sporočila SMS klikne, zatem pa se mu na Google zemljevidu odpre zemljevid z mestom lokacije.

```
final LocationListener locationListener = new LocationListener() {
    public void onLocationChanged(Location location) {
        mLocation = location;
        if(zePoslano)
        {
            Log.d("Poslane koordinate", "Ze poslano true");
        }
        else {
            zePoslano = true;
            poslji(location.getLatitude(), location.getLongitude(), location.getTime(), String.valueOf(location.getSpeed()));
        }
```

Slika 5.8 Izsek kode kjer sestavim vsebino sporočila SMS

5.7 Pošiljanje podatkov na strežnik

Za vsako komunikacijo s strežnikom smo ustvarili AsyncTask, ki se izvaja v ozadju na drugi niti, saj lahko komunikacija s strežnikom traja dlje časa. Na grafičnem



Slika 5.9 Obvestilo, ki se prikaže ob komunikaciji z bazo

vmesniku v tem času prikažemo dialog (Slika 5.9), kjer uporabnika obveščamo, kaj se dogaja.

V AsyncTasku kličemo metodo, ki je odgovorna za komunikacijo s strežnikom. V metodi se ustvarita HttpClient in HttpRequest na strežniku. Metoda prejme objekt JSON, ki ga pred tem sestavimo v AsyncTasku in vsebuje podatke, ki se bodo poslali na strežnik.

JSON pretvorimo v StringEntity, nato HttpRequestu nastavimo Entity na ta pretvorjen JSON. V nadaljevanju ustvarimo HttpResponse, ki kliče metodo execute(), da izvedemo zahtevo. Odgovor strežnika pretvorimo v niz, le-tega pa v format JSON, ki ga metoda vrne kot odgovor strežnika. Na sliki (Slika 5.10) vidimo, kako se sestavi objekt JSON, naslednja slika (SSlika 5.11) pa prikazuje zahtevek na strežnik.

```
JSONObject jsonobj = new JSONObject();
jsonobj.put("email", String.valueOf(params[0]));
jsonobj.put("password_hash", String.valueOf(params[1]));
rezultat = Utils.vrniOdgovorStreznika("register", jsonobj);
```

Slika 5.10: Sestava objekta JSON, ki ga pošljemo metodi za komunikacijo s strežnikom

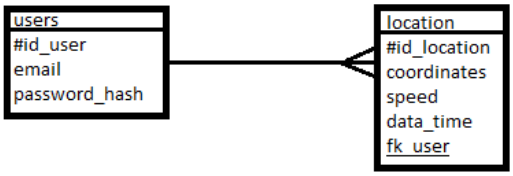
```
HttpClient httpclient = new DefaultHttpClient();
HttpPost httppost = new HttpPost(url);
Log.d(TAG_SERVIS, "Zahtevek na " + url);

try {
    // dodam post parametre
    StringEntity se = new StringEntity(parametri.toString());
    se.setContentType("application/json;charset=utf-8");
    se.setContentEncoding(new BasicHeader(HTTP.CONTENT_TYPE, "application/json;charset=utf-8"));
    httppost.setEntity(se);
    //izvedem zahtevek
    HttpResponse response = httpclient.execute(httppost);
    String responseText = EntityUtils.toString(response.getEntity());
```

Slika 5.11: Izsek kode kjer izvedem http zahtevo na strežnik

5.8 Podatkovna baza

V podatkovni bazi smo ustvarili dve tabeli. Prva tabela je bila tabela uporabnikov, ki je vsebovala id uporabnika (id_user), ki je tudi primarni ključ in se sam povečuje. Poleg tega vsebuje še e-mail (e-mail) oz. uporabniško ime in geslo (password_hash) v md5 obliki.

Druga tabela je bila tabela lokacij. Kot primarni ključ v tej tabeli je id lokacije (id_location), ki se avtomatsko povečuje. Ostali parametri so koordinate (coordinates), hitrost (speed), datum in čas (date_time) ter tuji ključ uporabnika (fk_user). Na sliki (SSlika 5.12) je grafični prikaz relacij v bazi.



Slika 5.12 Relacije v bazi

5.9 Testiranje

Aplikacija je bila testirana na mobilni napravi HTC Desire 820. Na omenjeni napravi so bili preizkušeni vsi ukazi SMS, ki so delovali pravilno. Nastavili smo intervalno pošiljanje na strežnik, se z mobilno napravo sprehodili po Mariboru ter sproti preverjali natančnost lokacij, za katere smo ugotovili, da so bile točne. Lokacije so bile manj natančne, ko smo izklopili GPS.

6. STREŽNIŠKI DEL

Ko smo na strežnik preko http zahteve pridobili podatke, smo jih morali shraniti v bazo. Podatki so v formatu JSON, servisi pa so napisani v programskem jeziku PHP.

6.1 Registracija

Servis register.php (Slika 6.1) skrbi za registracijo novega uporabnika. Ko se preko http zahteve pošljejo podatki v formatu JSON iz aplikacije na strežnik, jih na strežniku dekodiramo s pomočjo json_decode. Nato vzpostavimo povezavo z bazo ter vstavimo podatke. Kot rezultat pošljemo niz »ok«, če je bila registracija uspešna, sicer pošljemo niz »error«.

```php
<?php
$prijeti_podatki = file_get_contents('php://input');
$prejeti_json = json_decode($prijeti_podatki, true);
$email = $prejeti_json['email'];
$password_hash = $prejeti_json['password_hash'];
$is_removed = false;
try {
    $db_hostname = 'localhost';
    $db_username = 'netjein_dejan';
    $db_password = 'VATTO]m]n;=lRwgT';
    $db_name = 'netjein_dejan_sms';
    $db = new PDO("mysql:host=$db_hostname;dbname=$db_name", $db_username, $db_password);
    $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $stmt = $db->prepare("INSERT INTO users (email,password_hash) VALUES (:email,:password_hash)");
    $stmt->bindParam(':email', $email);
    $stmt->bindParam(':password_hash', $password_hash);
    $stmt->execute();

    $stmt1 = $db->prepare("SELECT * FROM users WHERE email=:email");
    $stmt1->bindParam(':email', $email);
    $stmt1->execute();
    if($user = $stmt1->fetch()) {
        $rezultat = array('rezultat' => 'ok', 'id_user' => $user['id_user']);
        echo json_encode($rezultat);
    }
    else {
        $rezultat = array('rezultat' => 'error', 'sporocilo' => 'Nepravilno uporabniško ime in/ali geslo');
        echo json_encode($rezultat);
    }
    $db = null;
}
catch(PDOException $e) {
    $rezultat = array('rezultat' => 'error', 'sporocilo' => $e->getMessage());
    echo json_encode($rezultat);
}
```

Slika 6.1: PHP koda za registracijo

6.2 Registracija z že obstoječim računom

Če uporabnik v aplikaciji izbere, da se želi registrirati z že obstoječim računom, se kliče servis get_user.php (Slika 6.2). Slednji prav tako kot pri registraciji najprej dekodira podatke in se nato poveže z bazo. Potem v bazi poišče uporabnika s tem e-mailom. Če ga ni, vrne rezultat, da uporabniškega imena ni našel. V primeru, da uporabniško ime najde, preveri še geslo. V primeru pravilnega gesla je prijava uspešna in vrne rezultat »ok«.

```php
$stmt1 = $db->prepare("SELECT * FROM users WHERE email=:email");
$stmt1->bindParam(':email', $email);
$stmt1->execute();
if($user = $stmt1->fetch()) {
    if($user['password_hash'] == $password_hash)
    {
        $rezultat = array('rezultat' => 'ok', 'id_user' => $user['id_user']);
        echo json_encode($rezultat);
    }
    else
    {
        $rezultat = array('rezultat' => 'error', 'sporocilo' => 'Nepravilno geslo!');
        echo json_encode($rezultat);
    }
}
else {
    $rezultat = array('rezultat' => 'error', 'sporocilo' => 'Uporabniškega imena ni v bazi!');
    echo json_encode($rezultat);
}
```

Slika 6.2: PHP koda za pridobivanje že obstoječega uporabnika

6.3 Vstavljanje prejete lokacije v bazo

Za vstavljanje lokacije, ki se pošlje na strežnik, skrbi servis send_location.php. Tudi pri tem servisu najprej dekodiramo podatke, se nato povežemo z bazo in jih vstavimo. Če se hitrost na strežnik ne pošilja, ima hitrost vrednost –1. Med prejetimi podatki je tudi id uporabnika aplikacije, ki ga pri vstavljeni lokaciji nastavimo na tuj ključ uporabnika (Slika 6.3). Tudi tukaj se kot odgovor v primeru nemotenega delovanja pošlje »ok« oz. »error«, če pride do kakšne napake.

```
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$stmt = $db->prepare("INSERT INTO location (coordinates, speed, date_time, fk_user) VALUES (:coordinates, :speed, NOW(), :fk_user)");
//$stmt = $db->prepare("INSERT INTO lokacija (koordinate, hitrost, datum, tk_uporabnik) VALUES (:coordinates, :speed, NOW(), :fk_user)");
$stmt->bindParam(':coordinates', $coordinates);
$stmt->bindParam(':speed', $speed);
$stmt->bindParam(':fk_user', $fk_user);
$stmt->execute();
```

Slika 6.3: Vstavljanje lokacije v bazo

7. SPLETNI DEL

Spletni del aplikacije je bil razvit z namenom pregleda nad lokacijami, poslanimi na strežnik. V spletno aplikacijo se je potrebno prijaviti z istim uporabniškim imenom in geslom kot v glavno aplikacijo. Po prijavi se odpre spletna stran s seznamom lokacij, poslanih na strežnik.

7.1 Uporabniški vmesnik

Najprej smo implementirali dve osnovni html strani ter s pomočjo CSS oblikovali videz (nastavili ozadje, velikost pisav, barve …). Na naslovni strani (index.html, Slika 7.1) je prijavni obrazec, kamor se oseba prijavi s podatki, identičnimi podatkom, s katerimi se oseba vpiše v Android aplikacijo. Po uspešni prijavi se odpre uporabniški pogled (userView.html), kjer so izpisane vse lokacije uporabnika, ki so bile poslane na strežnik.

```
<head>
    <meta charset="UTF-8">
    <title>SMS Android - Prijava</title>
    <link rel = "stylesheet" type="text/css" href="index.css"></link>
    <script src = "http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.12.2.min.js"></script>
    <script src = "http://www.myersdaily.org/joseph/javascript/md5.js"></script>
</head>
<body>
    <center>
        <h1>Prijava v Spletni SMS <br> Android</h1>
        <div class="field">
            <div class="divSingIn">
                <p class="TextView">Email: <input id="email" class="editFields" width="auto" type="email " name="email" value=""><br></p>
                <p class="TextView">Geslo: <input id="password" class="editFields" type="password" name="password" value=""></p>
                <p id="notice"><br></p>
                <input id="BsingIn" name="BsingIn" type="button" value="Prijava" class="BsingIn" onclick="singIn()">
            </div>
        <div>
    </center>
</body>
```

Slika 7.1: Koda naslovne strani index.html

7.2 Prijava

Ob kliku na gumb prijava se kliče JavaScript funkcija (vključen tudi jQuery), ki pridobi podatke iz HTML vnosnih polj ter pošlje zahtevek AJAX na strežnik. Podatki se preko AJAX-a na strežnik pošljejo v formatu JSON. V sign_in.php najprej s pomočjo $_POST pridobimo prejete podatke, se povežemo z bazo in v primeru, da uporabnik obstaja ter da je geslo pravilno, vrnemo id uporabnika, sicer vrnemo »error«. Nato, če odgovor ni enak »error«, odpremo userView.html.

7.3 Prikaz lokacij

S preusmeritvijo na userView.html, se ob dogodku onLoad kliče JavaScript funkcija, ki iz localStorage pridobi id in nato izvede klic AJAX na strežnik. V get_location_of_user.php se najprej vzpostavi povezava z bazo ter poišče uporabnik z id-jem iz localStorage. Nato se kreira tabela, v kateri sta izpisana datum in čas ter koordinate in hitrost uporabnika na posamezni lokaciji, kot je prikazano na Slika 7.2.

Lokacije

50 ▾ Refresh Sing Out

Datum	Koordinate	Hitrost	Provider
03.07.2017 - 10:41:05	http://www.google.com/maps/place/46.6182871,15.2247882	30m/s	GPS
03.07.2017 - 10:41:05	http://www.google.com/maps/place/46.6182871,15.2287882	30m/s	GPS
03.07.2017 - 10:41:05	http://www.google.com/maps/place/46.6182871,15.2987882	44m/s	GPS
03.07.2017 - 10:41:05	http://www.google.com/maps/place/47.6182871,15.2987882	42m/s	GPS
03.07.2017 - 10:41:05	http://www.google.com/maps/place/46.6182871,16.2987882	42m/s	GPS
03.01.2017 - 10:41:37	Lokacija ni na voljo!	Ni podatka	/
03.01.2017 - 10:41:05	http://www.google.com/maps/place/46.6132871,15.2247882	Ni podatka	NETWORK

Slika 7.2: Lokacije, prikazane na spletni strani

8. POMOŽNA APLIKACIJA

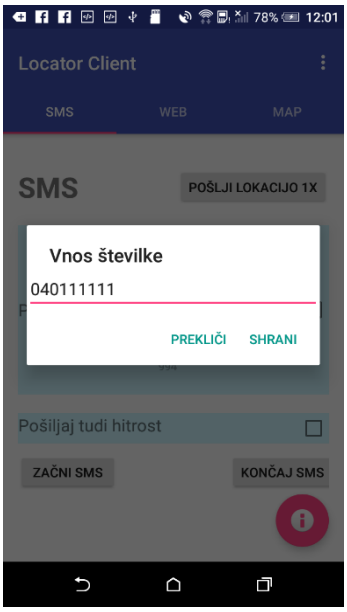
Ker so bili ti ukazi za povprečnega uporabnika preveč zahtevni, smo izdelali še odjemalca, ki služi kot uporabniški vmesnik ter namesto uporabnika pošilja sporočila, kar pomeni, da uporabniku ni treba poznati vseh zapletenih ukazov.

8.1 Prijava

Z namenom povezave s strežnikom se moramo prijaviti tudi v odjemalca. Na tak način pridobivamo lokacije, ki so poslane na strežnik. Prikazane so na OpenStreetMap, (slednji je opisan pri uporabljenih tehnologijah). Tako uporabniku ni potrebno obiskati spletne strani ter lokacije pregledovati z Google zemljevidom. Prijava je implementirana podobno kot pri osnovni aplikaciji, vendar obstaja nekaj majhnih razlik.

8.2 Vnos številke ter gesla za prejem

Po prijavi je potrebno vnesti številko, na katero se naj pošiljajo sporočila SMS. Številka se doda v meniju s pomočjo razreda AlertDialog. Razred predstavlja dialog, kjer lahko uporabnik vnese zahtevane podatke. AlertDialog se ustvari s konstruktorjem AlertDialog.Builder, kjer mu nato nastavimo naslov z metodo setTitle() ter ostale parametre, kot je prikazano na eni izmed slik (Slika 8.2). Z metodama setPositiveButton() in setNegativeButton()nastavimo dogodke po kliku na pozitivni oz. negativni gumb. Na koncu moramo poklicati še metodo show(), ki prikaže dialog, kot vidimo na naslednji sliki (SSlika 8.1). Tudi za vnos gesla za prejem smo izbrali enak način.



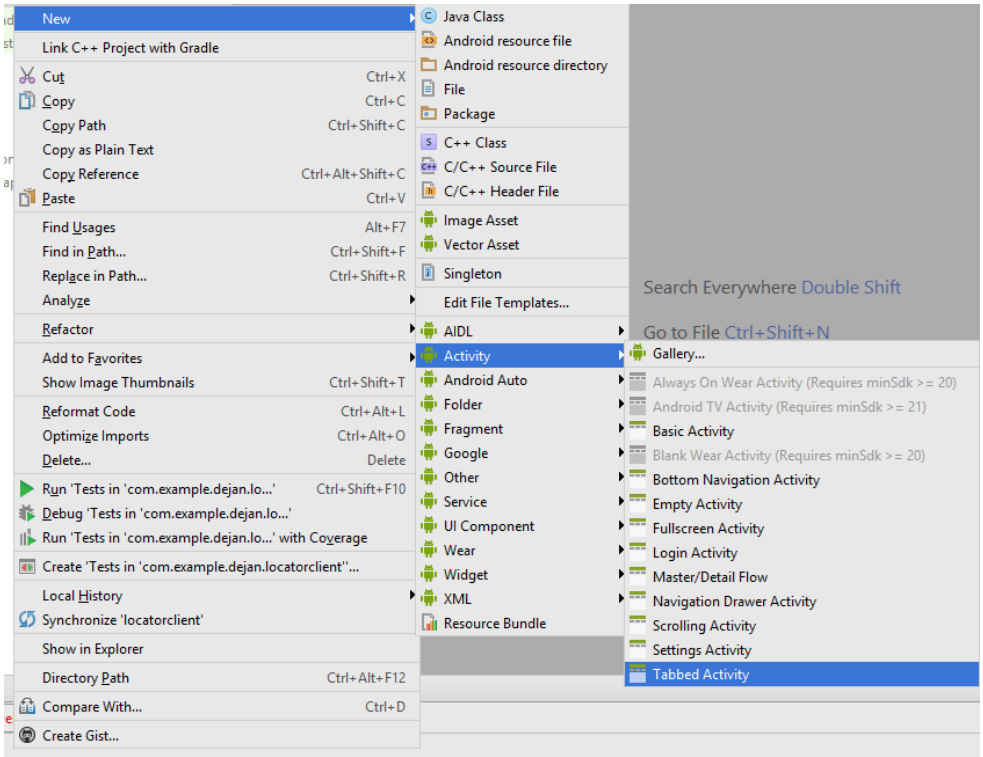
Slika 8.1: Prikaz AlertDialoga

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);
builder.setTitle("Vnos gesla: ");

final EditText input = new EditText(this);
input.setText(SPMuser.getStringId("gesloZaPrejem"));
input.setInputType(InputType.TYPE_CLASS_TEXT );
builder.setView(input);

builder.setPositiveButton("Shrani", (dialog, which) → {
        String geslo = input.getText().toString();
            SPMuser.saveStringId("gesloZaPrejem", geslo);
});
builder.setNegativeButton("Prekliči", (dialog, which) → {
        dialog.cancel();
});

builder.show();
```

Slika 8.2: Nastavitve parametrov AlertDialoga

8.3 Vmesnik z zavihki

Zaradi boljšega izgleda in bolj praktične uporabe smo se odločili za izdelavo vmesnika z zavihki. Android Studio za takšen vmesnik omogoča predlogo, s pomočjo katere ga je veliko lažje izdelati. Kako priti do predloge, prikazuje slika (Slika 8.3). Ko ustvarjamo predlogo, moramo izbrati tudi način premikanja med zavihki. Na izbiro imamo premikanje s potegi levo in desno ali pa premikanje s pomočjo orodne vrstice, ki jo imamo zgoraj in v kateri so zavihki, po katerih se lahko premikamo. Lahko imamo tudi spustni seznam, v katerem so zavihki. Odločili smo se za orodno vrstico z vsebujočimi zavihki.



Slika 8.3 Ustvarjanje Vmesnika z zavihki

Ko ustvarimo takšen vmesnik, nam Android Studio ustvari dve postavitvi, in sicer vmesnik, na katerem bodo fragmenti ter vmesnik z enim fragmentom. Dodati smo morali toliko fragmentov, kolikor želimo imeti zavihkov. Pri tem smor kopirali predpripravljen fragment ter ga primerno preimenovali. V osnovi imamo razred, ki razširja razred Fragment kar v vmesniku, na katerem so fragmenti. Zaradi preglednosti smo raje dali vsak razred v svojo datoteko in ga poimenovali enako kot postavitve ter jih povezali s postavitvami. Nato lahko vsako postavitev primerno uredimo.

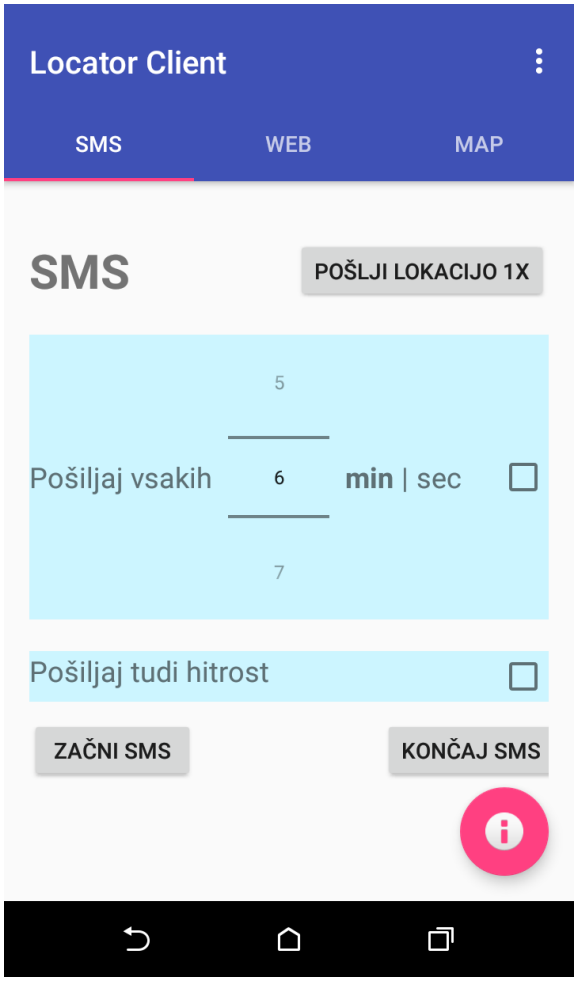
8.4 Fragment SMS

Po tem, ko smo ustvarili želene zavihke, je bilo potrebno vsakega posebej napolniti z gradniki. Ker je namen aplikacije pošiljanje sporočil SMS s pomočjo vmesnika, smo morali uporabiti gradnike, s katerimi so se lahko določili parametri ukazov.

Najprej smo dodali NumberPicker, ki nam omogoča izbiro števila, ki bo služilo kot parameter za izbiro časovnosti (torej na koliko časa naj se na strežnik pošilja sporočila SMS oz. lokacija). Dodamo ga lahko kar s pomočjo “povleci in spusti”, ki ga omogoča

Android Studio. Nato nastavimo parametre, kot so najmanjša možna vrednost in največja možna vrednost.

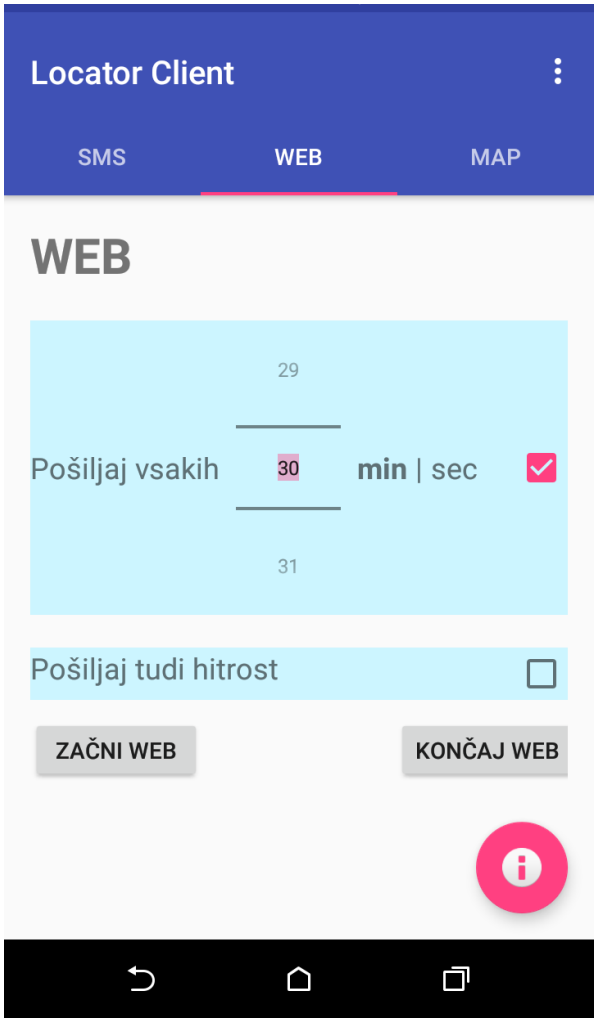
Enako kot z NumberPicker-jem je tudi s CheckBox-om, samo dodamo ga na fragment, nastavimo lastnosti in nato napišemo metodo setOnCheckedChangeListener(), v kateri definiramo, kaj se zgodi, ko ga odkljukamo. Izgled fragmenta SMS prikazuje spodnja slika Slika 8.4).



Slika 8.4: Izgled fragmenta SMS

8.5 Fragment WEB

Tudi na fragmentu WEB so zelo podobni gradniki, saj oba fragmenta opravljata zelo podobno vlogo. Slika 8.5 prikazuje izgled fragmenta WEB.



Slika 8.5: Izgled fragmenta WEB

8.6 Fragment MAP

Fragment map je namenjen prikazu zadnjih lokacij, ki so bile poslane na strežnik. Na njem je Spinner, na katerem je seznam lokacij. Pod njim je OpenStreetMap, kjer so z ikono označene lokacije, ki jih ob zagonu pridobimo s strežnika. S klikom na posamezno lokacijo se nam prikažeta datum in ura, ko je bila lokacija poslana na strežnik.

Da smo lahko uporabil OpenStreetMap, smo morali najprej dodati knjižnico osmdroid-android-4.1.jar. Nato smo v »Layout« dodali MapView, kot je prikazano na spodnji sliki Slika 8.6). Za tem je bilo potrebno inicializirati MapView in MapContorler ter nastaviti osnovne parametre, ki so naslednji: kje naj bo središče prikaza, kakšna naj bo povečava (Slika 8.7).

```
<org.osmdroid.views.MapView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/mapview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_marginBottom="70dp"
    android:clickable="true" />
```

Slika 8.6: MapView v Layout-u

```
mMapView = (MapView) rootView.findViewById(R.id.mapview);
mMapView.setTileSource(TileSourceFactory.MAPNIK);
mMapController = (MapController) mMapView.getController();
mMapController.setZoom(13);
String[] prvi_lat_long = vse_lat_long_posebej[0].split(",");
GeoPoint gPt = new GeoPoint(Double.parseDouble(prvi_lat_long[0]), Double.parseDouble(prvi_lat_long[1]));
mMapController.setCenter(gPt);
mMapView.setBuiltInZoomControls(false);
mMapView.setMultiTouchControls(true);
```

Slika 8.7: Inicializacija MapView-a

Vtiče dodamo tako, da v seznam dodajamo OverlayItem-e, kamor vstavimo napis, ki naj se prikaže ob kliku na vtič, ter koordinate kot GeoPoint, ki sprejme geografsko širino ter geografsko dolžino. Nato ustvarimo Overlay, ki sprejme kontekst ter seznam OverlayItem-ov. Na koncu moramo še MapView-u dodati Overlay z metodo getOverlays().add().

9. SKLEP

V diplomski nalogi smo izdelali aplikacijo Locator, s katero smo zadovoljni, sajdeluje tako, kot smo si zastavili. Intervali, ki zbujajo aplikacijo, da se pošlje lokacijo, so dokaj točni, vendar mora biti za pošiljanje lokacije na strežnik vzpostavljena internetna povezava. Aplikacija se izvaja v ozadju ter ne zavzema preveč virov. Upravljamo jo preko ukazov SMS ali s pomočjo uporabniške aplikacije, ki sporočila SMS pošilja namesto nas. Pri nastavljanju aplikacije preko ukazov SMS, si je bilo potrebno izmisliti smiselne ukaze, ki so omogočali vse nastavitve. Koordinate GPS so natančne in se posodabljajo pravočasno, medtem ko parameter trenutne hitrosti ni tako natančen. Aplikacija se začne izvajati samodejno po ponovnem zagonu telefona, če je bila v izvajanju tudi pred ponovnim zagonom. Metoda, ki pregleda sporočila SMS, samodejno izpusti sporočila, za katere zazna, da niso namenjena aplikaciji.

Za upravljanje aplikacije s pomočjo sporočil SMS mora biti v program vnešena telefonska številka administratorja programa. Sicer se jo lahko upravlja tudi z geslom, ki je nastavljeno v aplikaciji z namenom, da lahko aplikacijo upravljamo s strani več telefonov, a le v primeru poznavanja gesla.

Pri uporabniški aplikaciji je mogoče enostavno nastaviti vse parametre, zato ni potrebe po učenju ukazov. Lokacije na strežniku se učinkovito prikazujejo na zemljevidu Open Street Maps. Spletna aplikacija, ki ni bila v ospredju, pa prav tako omogoča enostaven pregled lokacij na strežniku.

Dostop otroka do aplikacije je onemogočen s pomočjo gesla, ki ga starš nastavi po namestitvi aplikacije, vendar lahko otrok v nastavitvah vseeno onemogoči aplikacijo, kar prepreči pošiljanje lokacije. Prav tako se pošiljanje na strežnik ne izvaja, če ima otrok izklopljen prenos podatkov in Wi-Fi.

Največ težav nam je povzdročalo pridobivanje trenutnih koordinat, saj koordinate niso bile primerno posodobljene. Dosegli smo tudi, da aplikacija deluje še po ponovnem zagonu sistema, kar nam je vzelo nekoliko več časa kot ostale funkcionalnosti.

Za napredek je še veliko prostora, saj bi se lahko aplikacija uporabljala za veliko drugih namenov.

10. VIRI

[1] Hashimi S., Komatineni S., MacLean D. Pro Android 2. Apress (2010)

[2] Wikipedija, Android. Dostopno na: https://sl.wikipedia.org/wiki/Android_(operacijski_sistem) (13.07.2017)

[3] Wikipedija, programski jezik Java. Dostopno na: https://sl.wikipedia.org/wiki/Programski_jezik_java (13.07.2017)

[4] Wikipedija, zgodovina HTML. Dostopno na: https://en.wikipedia.org/wiki/HTML#Development (13.07.2017)

[5] Fdv Ljubljana, osnove HTML. Dostopno na: http://mrvar.fdv.uni-lj.si/sola/VIS2/html/htmlslo.htm (13.07.2017)

[6] Wikipedija, CSS. Dostopno na: https://sl.wikipedia.org/wiki/CSS (13.07.2017)

[7] Mozilla, About JavaScript. Dostopno na: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript (13.07.2017)

[8] Wikipedija, JavaScript. Dostopno na: https://sl.wikipedia.org/wiki/JavaScript (13.07.2017)

[9] Wikipedija, jQuery. Dostopno na: https://en.wikipedia.org/wiki/JQuery (13.07.2017)

[10] Wikipedija, PHP. Dostopno na: https://en.wikipedia.org/wiki/PHP (13.07.2017)

[11] Tutorials point AJAX. Dostopno na: https://www.tutorialspoint.com/ajax/what_is_ajax.htm (13.07.2017)

[12] JSON. Dostopno na: http://www.json.org/json-sl.html (13.07.2017)

[13] About OpenStreetMaps. Dostopno na: https://www.openstreetmap.org/about (13.07.2017)

[14] About Osmdroid. Dostopno na: http://wiki.openstreetmap.org/wiki/Osmdroid (13.07.2017)

[15] Send my Location. Dostopno na: https://play.google.com/store/apps/details?id=bg.angelov.send.my.location (31.07.2017)

[16] Send my GPS location. Dostopno na: https://play.google.com/store/apps/details?id=net.mypapit.mobile.myposition (31.07.2017)

[17] Moja lokacija na zemljevidu. Dostopno na: https://play.google.com/store/apps/details?id=com.location.test (31.07.2017)

[18] My location. Dostopno na: https://play.google.com/store/apps/details?id=com.mobilekidunia.mylocation (31.07.2017)

[19] Glympse – Share GPS location. Dostopno na: https://play.google.com/store/apps/details?id=com.glympse.android.glympse (31.07.2017)

[20] My GPS coordinates. Dostopno na: https://play.google.com/store/apps/details?id=com.gpscoordinatesandlocation (31.07.2017)

[21] Show my location. Dostopno na: https://play.google.com/store/apps/details?id=com.ravindra.whereim (31.07.2017)

[22] Share my position. Dostopno na: https://play.google.com/store/apps/details?id=it.frusso.android.sharemyposition (31.07.2017)

[23] Mobile location tracker. Dostopno na: https://play.google.com/store/apps/details?id=com.crazyapps.mobilelocationtracker (31.07.2017)

[24] My location #2. Dostopno na: https://play.google.com/store/apps/details?id=com.themelisx.mylocation (31.07.2017)

[25] Poišči mojo napravo. Dostopno na:

https://play.google.com/store/apps/details?id=com.google.android.apps.adm
(31.07.2017)

FERI

IZJAVA O AVTORSTVU

Spodaj podpisani/-a Dejan Orter

z vpisno številko E1074611

sem avtor/-ica diplomskega dela z naslovom: Razvoj mobilne in spletne aplikacije Locator

(naslov diplomskega dela)

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom (naziv, ime in priimek) doc. dr. Matej Črepinšek, univ. dipl. inž. rač. in inf.

  in somentorstvom (naziv, ime in priimek)

- 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 DKUM.

V Mariboru, dne 22.08.2017

Podpis avtorja/-ice:

Orter Dej

FERI

IZJAVA O USTREZNOSTI ZAKLJUČNEGA DELA

Podpisani mentor/-ica : doc. dr. Matej Črepinšek, univ. dipl. inž. rač. in inf.

(ime in priimek mentor-ja/-ice)

in somentor/-ica (eden ali več, če obstajajo):

(ime in priimek somentor-ja/-ice)

Izjavlja-m/-va/-mo, da je študent/-ka

Ime in priimek: Dejan Orter

ID številka: 1002142121

vpisna številka: E1074611

na študijskem programu: Računalništvo in informacijske tehnologije

izdelal/-a zaključno delo z naslovom:

Razvoj mobilne in spletne aplikacije Locator

Development of the mobile and web application Locator

(naslov zaključnega dela v slovenskem jeziku)

v skladu z odobreno temo zaključnega dela, navodili o pripravi zaključnih del in mojimi (najinimi/našimi) navodili.

Preveril/-a/-i in pregledal/-a/-i sem/sva/smo poročilo o preverjanju podobnosti vsebin z drugimi deli (priloga) in potrjujem/potrjujeva/potrjujemo, da je zaključno delo ustrezno.

Datum in kraj: Maribor, 22.08.2017 Podpis mentor-ja/-ice:

Datum in kraj: Maribor, 22.08.2017 Podpis somentor-ja/-ice (če obstaja):

FERI

IZJAVA O ISTOVETNOSTI TISKANE IN ELEKTRONSKE VERZIJE ZAKLJUČNEGA DELA IN OBJAVI OSEBNIH PODATKOV DIPLOMANTOV

Ime in priimek avtorja-ice: Dejan Orter

Vpisna številka: E1074611

Študijski program: Računalništvo in informacijske tehnologije

Naslov zaključnega dela: Razvoj mobilne in spletne aplikacije Locator

Mentor: *doc. dr. Matej Črepinšek, univ. dipl. inž. rač. in inf.*

Somentor: _____

Podpisani-a Dejan Orter izjavljam, da sem za potrebe arhiviranja oddal elektronsko verzijo zaključnega dela v Digitalno knjižnico Univerze v Mariboru. Zaključno delo sem izdelal-a sam-a ob pomoči mentorja. V skladu s 1. odstavkom 21. člena Zakona o avtorskih in sorodnih pravicah dovoljujem, da se zgoraj navedeno zaključno delo objavi na portalu Digitalne knjižnice Univerze v Mariboru.

Tiskana verzija zaključnega dela je istovetna z elektronsko verzijo elektronski verziji, ki sem jo oddal za objavo v Digitalno knjižnico Univerze v Mariboru.

Zaključno delo zaradi zagotavljanja konkurenčne prednosti, varstva industrijske lastnine ali tajnosti podatkov naročnika: _____ ne sme biti javno dostopno do _____(datum odloga javne objave ne sme biti daljši kot 3 leta od zagovora dela).

Podpisani izjavljam, da dovoljujem objavo osebnih podatkov, vezanih na zaključek študija (ime, priimek, leto in kraj rojstva, datum zaključka študija, naslov zaključnega dela), na spletnih straneh in v publikacijah UM.

Datum in kraj: Maribor, 22.08.2017 Podpis avtorja-ice: Orter Dejan

Podpis mentorja: _____
(samo v primeru, če delo ne sme biti javno dostopno)

Podpis odgovorne osebe naročnika in žig:
(samo v primeru, če delo ne sme biti javno dostopno) _____