Mobilna Android aplikacija za kooperativno stvaranje i preporučivanje ruta obilazaka turističkih znamenitosti

Čačić, Mislav

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

MOBILNA ANDROID APLIKACIJA ZA KOOPERATIVNO STVARANJE I PREPORUČIVANJE RUTA OBILAZAKA TURISTIČKIH ZNAMENITOSTI

Završni rad

Mislav Čačić

Osijek, 2022.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 14.09.2022.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Mislav Čačić
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	4499, 09.10.2019.
OIB Pristupnika:	26788970424
Mentor:	Prof.dr.sc. Goran Martinović
Sumentor:	,
Sumentor iz tvrtke:	Mario Levanić
Naslov završnog rada:	Mobilna Android aplikacija za kooperativno stvaranje i preporučivanje ruta obilazaka turističkih znamenitosti
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rad:	U teorijskom dijelu završnog rada, uz osvrt na postojeća rješenja, treba analizirati i opisati izazove i mogućnosti preporučivanja turističkih znamenitosti i stvaranja ruta obilazaka od strane samih posjetitelja. Uzimajući u obzir sklonosti korisnika, mogućnosti korištenja geolokacije i dodjeljivanja multimedijskih i tekstualnih sadržaja geolokaciji, te kooperativno i na sadržaju zasnovano stvaranje preporuka, treba predložiti model i arhitekturu mobilne Android aplikacije. Mobilna aplikacija treba za izabrano područje omogućiti izbor inicijalne rute obilaska, dodavanje znamenitosti u rute na odgovarajuće geolokacije, dodavanje multimedijskih i tekstualnih sadržaja znamenitosti, te preporučivanje novih ruta obilaska zasnovano sadržajnom i na kooperativnom preporučivanju i izboru optimalne rute. Nakon opisa potrebnih programskih tehnologija, jezika i razvojne okoline, u praktičnom dijelu rada treba razviti mobilnu aplikaciju s bazom podataka koristeći predloženi model i arhitekturu. Mobilnu aplikaciju potrebno je ispitati i analizirati za različite slučajeve korištenja i izabrano područje obilaska. Tema rezervirana za: Mislav Čačić Mentor iz tvrtke: Mario Levanić, Informatika Fortuno d.o.o., Vinkovci
Prijedlog ocjene završnog rada:	Izvrstan (5)

Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 2 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	14.09.2022.
Datum potvrde ocjene od strane Odbora:	21.09.2022.

Potvrda mentora o predaji konačne verzije rada:	*Mentor elektronički potpisao predaju konačne verzije.*
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 27.09.2022.

Ime i prezime studenta:	Mislav Čačić
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	4499, 09.10.2019.
Turnitin podudaranje [%]:	11

Ovom izjavom izjavljujem da je rad pod nazivom**: Mobilna Android aplikacija za kooperativno stvaranje i preporučivanje ruta obilazaka turističkih znamenitosti**

izrađen pod vodstvom mentora Prof.dr.sc. Goran Martinović

i sumentora ,

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. POTPORA RAČUNALNIH TEHNOLOGIJA U KOOPERATIVNOM STVARENJU I PREPORUČIVANJU RUTA OBILASKA TURISTIČKIH ZNAMENITOSTI 3

2.1. Turizam 3

2.2. Izazovi obilaska turističkih znamenitosti i potpora računalnih tehnologija 4

2.3. Pregled područja teme i aktualna rješenja 4

2.3.1. Pregled područja teme 5

2.3.2. Places Near Me 5

2.3.3. Polarsteps 6

2.3.4. Visit A City 7

3. MODEL I ARHITEKTURA APLIKACIJE U KOOPERATIVNOM STVARANJU I PREPORUČIVANJU RUTA OBILASKA TURISTIČKIH ZNAMENITOSTI 9

3.1. Funkcijski i nefunkcijski zahtjevi za mobilnu aplikaciju za kooperativno stvaranje i preporučivanje ruta obilazaka turističkih znamenitosti 9

3.2. Potrebni procesi i programski pristup za ostvarivanje aplikacije 10

3.2.1. Dohvaćanje lokacije mobilnog uređaja korisnika 10

3.2.2. Postupak kreiranja rute kretanja ovisno o preferencijama korisnika, kooperativno stvaranje i preporučivanje ruta obilaska 11

3.2.3. Baza podataka 12

3.3. Arhitektura mobilne Android aplikacije 12

4. PROGRAMSKO RJEŠENJE MOBILNE APLIKACIJE ZA KOOPERATIVNO STVARANJE I PREPORUČIVANJE RUTA OBILAZAKA TURISTIČKIH ZNAMENITOSTI 14

4.1. Korištene programske tehnologije, razvojne okoline i jezici kako bi se aplikacija napravila 14

4.1.1. Operacijski sustav Android 14

4.1.2. Razvojno okruženje Android Studio 15

4.1.3. Programski jezik Kotlin 16

4.1.4. XML 17

4.1.5. Firebase 17

4.2. Programska rješenja na strani korisnika 18

4.2.1. Prikaz rute i Google Maps Activity 18

4.2.2. Početni zaslon ... 20
4.2.3. Dohvaćanje lokacije korisnika ... 21
4.2.4. Dodavanje nove znamenitosti i kooperativno stvaranje preporuka na osnovu ocjene znamenitosti ... 22
4.2.5. Izgradnja rute ... 23
4.2.6. Pregled detalja profila ... 24
4.2.7. Prijava ili registracija ... 25

4.3. Programska rješenja na strani poslužitelja ... 25

4.3.1. Baza podataka ... 25
4.3.2. Sortiranje lokacija po blizini ... 26
4.3.3. Dohvaćanje lokacije korisnika ... 27

5. PRIKAZ RADA APLIKACIJE, ANALIZA RADA I ISPITIVANJE REZULTATA ... 28

5.1. Upute za korištenje aplikacije za kooperativno stvaranje i preporučivanje ruta obilazaka turističkih znamenitosti ... 28

5.2. Uvjeti ispitivanja i korisnički slučajevi ... 33

5.2.1. Definiranje uvjeta za ispitivanje aplikacije ... 33
5.2.2. Slučajevi korištenja ... 34

5.3. Analiza rada ostvarene mobilne aplikacije ... 37

6. ZAKLJUČAK ... 38

LITERATURA ... 39

ŽIVOTOPIS ... 41

SAŽETAK ... 42

ABSTRACT ... 43

PRILOZI ... 44

1. UVOD

U današnje vrijeme, razvojem tehnologije i njezinih mogućnosti, postoji prilika gotovo u svakom dijelu svijeta, u svakom trenutku dobiti traženu informaciju putem interneta, mobilnih prijenosnih uređaja, laptopa itd. Prednosti napretka tehnologije jest što korisnicima olakšava veliku većinu svakodnevnih obveza, obavljanje poslova, prikupljanje informacija slično. Brze i jednostavne informacije od velike su važnosti u turizmu i njegovom promoviranju. Tehnologija u ovom slučaju ne donosi korist samo turistima, već i turističkim djelatnicima, kako bi mogli oglasiti svoje ponude i programe, saznati što turisti najviše žele posjetiti u određenom mjestu, što turiste od znamenitosti grada najviše zanima te po tome stvoriti i napraviti svoju ponudu koju će plasirati turistima.

Glavni zadatak ovog završnog rada jest napraviti mobilnu Android aplikaciju za stvaranje i preporučivanje ruta obilazaka turističkih znamenitosti u gradu Osijeku. Smisao aplikacije jest, po određenim odabirima korisnika, preporučiti znamenitost, napraviti rutu obilazaka znamenitosti i pružiti informacije o znamenitosti turistima. Cilj ovog završnog rada jest analizirati rješenja koja već postoje, proučiti ih te na osnovu analize napraviti jednostavnije, pristupačnije i bolje rješenje od postojećeg. Također, pomoću kreiranih računa korisnika i njihovih preferencija i odabira stvoriti najprikladniju rutu obilazaka znamenitosti koje odgovaraju njihovim odabirima.

U poglavlju 2 opisane su tehnološke potpore za ostvarivanje aplikacije i slična postojeća rješenja. U poglavlju 3 opisan je model i arhitektura aplikacije, funkcijski i nefunkcijski zahtjevi aplikacije i potrebni procesi za ostvarivanje aplikacije. Poglavlje 4 opisuje korištene programske tehnologije, razvojne okoline te kroz programski kod i opis objašnjava rad aplikacije. U poglavlju 5 opisan je način korištenja aplikacije, te je prikazano ispitivanje aplikacije s analizom rezultata ispitivanja.

1.1. Zadatak završnog rada

U teorijskom dijelu završnog rada, uz osvrt na postojeća rješenja, analizirat će se i opisati izazovi i mogućnosti preporučivanja turističkih znamenitosti i stvaranja ruta obilazaka od strane samih posjetitelja. Uzimajući u obzir sklonosti korisnika, mogućnosti korištenja geolokacije i dodjeljivanja multimedijskih i tekstualnih sadržaja geolokaciji, te kooperativno i na sadržaju zasnovano stvaranje preporuka, treba predložiti model i arhitekturu mobilne Android aplikacije. Mobilna aplikacija treba, za izabrano područje, omogućiti izbor inicijalne rute obilaska, dodavanje znamenitosti u rute na odgovarajuće geolokacije, dodavanje multimedijskih i tekstualnih sadržaja

znamenitosti, te preporučivanje novih ruta obilaska zasnovano sadržajnom i na kooperativnom preporučivanju i izboru optimalne rute. Nakon opisa potrebnih programskih tehnologija, jezika i razvojne okoline, u praktičnom dijelu rada treba razviti mobilnu aplikaciju s bazom podataka koristeći predloženi model i arhitekturu. Mobilnu aplikaciju potrebno je ispitati i analizirati za različite slučajeve korištenja i izabrano područje obilaska.

2. POTPORA RAČUNALNIH TEHNOLOGIJA U KOOPERATIVNOM STVARENJU I PREPORUČIVANJU RUTA OBILASKA TURISTIČKIH ZNAMENITOSTI

U 21. stoljeću gotovo svaka osoba ima pametni telefon te je zbog toga razvoj mobilnih aplikacija važan u svim segmentima života pa tako i u turizmu. Samim razvojem mobilnih aplikacija znatno se olakšava čovjekov život na dnevnoj bazi s jednostavnim zadacima koje aplikacije mogu obaviti, pa i onih težih.

Kako bi se olakšalo promoviranje turizma i obilasci turističkih ruta u gradu Osijeku, napravljena je i ova aplikacija. Cilj je aplikacije da u konačnici pomogne razvoju turizma, olakšavanju odluka posjetitelja što vidjeti od znamenitosti te pomoć turističkim djelatnicima kako bi mogli napraviti turističku ponudu poput obilaska grada, te usluge organiziranog prijevoza na određene lokacije.

2.1. Turizam

Prema [1], turizam je ukupnost odnosa i pojava koji proizlaze iz putovanja i boravka posjetitelja nekog mjesta, ako je takvo putovanje poduzeto radi odmora i uživanja te se njime ne zasniva stalno prebivalište i ne poduzima se neka gospodarska djelatnost. Do porasta turizma došlo je zbog čovjekove potrebe za odmorom i opuštanjem nakon posla, posjećivanjem znamenitosti i prirodnih krajolika ili zbog javnih društvenih događaja poput koncerta, festivala i slično. Zbog prihoda koje turizam donosi pojedinoj državi, ali i pojedincima u ugostiteljskom sektoru, postao je jedna od glavnih gospodarskih grana svake zemlje.

Turistički obilazak može biti planiran, ponovni turistov obilazak mjesta koje je već posjetio, obilazak na sugestiju bliske osobe turistu, posjet mjestu koje je turist vidio na televiziji, društvenim mrežama i slično. Također, turistički obilazak može biti i neplaniran, gdje turist nije unaprijed odredio rutu obilaska ili vremensko trajanje obilaska.

Zbog toga je korisno da svako turističko mjesto ima svoju web ili mobilnu aplikaciju koja pomaže turistu kako bi odredio rutu obilaska znamenitosti prema svojim željama i odabirima, te kako bi vrijeme boravka što bolje iskoristio i preporučio potencijalnim budućim posjetiteljima destinacije.

2.2. Izazovi obilaska turističkih znamenitosti i potpora računalnih tehnologija

Svaki turist i turistički obilazak je različit. Na primjer, ukoliko je turist osoba mlađe životne dobi s neograničenim vremenom i novčanim resursima, te osoba koja može imati cjelodnevni obilazak znamenitosti, takva osoba neće imati problema ili smetnji prilikom obilaska znamenitosti. Međutim, realnost je drugačija, najčešće suprotna od primjera. Prosječan posjet turista destinaciji traje između pet i sedam dana, ograničenog je budžeta i dnevnog vremena za obilazak znamenitosti; odnosno turisti žele obići što više sadržaja, u što manjem vremenskom roku i sa što manje pretjeranog fizičkog umaranja.

Za rješenje navedenih problema od pomoći može biti lokalna osoba ili osoba koja vrlo dobro poznaje određenu turističku destinaciju tako što turistima predlaže unaprijed definiranu rutu obilaska turističkih atrakcija. Prema [2], s pametnim uređajem koji ima ugrađen GPS može se postići odličan turistički doživljaj te, pomoću internetske veze, radi pronalaženje informacija o znamenitosti i pomoću kamere kako bi cjelokupni doživljaj i ovjekovječili slikom.

Skoro svaka osoba danas ima pametni uređaj i pristup internetu. Razvojem mobilnih i web aplikacija turistu se može znatno olakšati određivanje rute obilaska znamenitosti tako što to mobilni uređaj i aplikacija naprave za njega, dok turistu ostaje više vremena i energije za ostale aktivnosti o kojima se mora brinuti prilikom obilaska mjesta.

Prema [3], broj aktivnih pametnih uređaja iznosi 6,64 milijarde. Zbog toga nije ni čudo što se broj mobilnih aplikacija stalno povećava u svim segmentima pa i u turizmu. Međutim, postoji puno više aplikacija koje prikazuju turističke znamenitosti i informacije o njima, nego aplikacija koje kreiraju rutu za obilazak istih.

2.3. Pregled područja teme i aktualna rješenja

U ovom će dijelu biti prikazana postojeća rješenja slična ovom radu na Google Play platformi, a razlog prikaza dostupnih aplikacija na ovoj platformi jest ciljana skupina za koju je namijenjena ova aplikacija. Budući da se radi o Android aplikaciji i aplikaciji namijenjenoj za Android OS, bit će izostavljene slične postojeće aplikacije na drugim operacijskim sustavima.

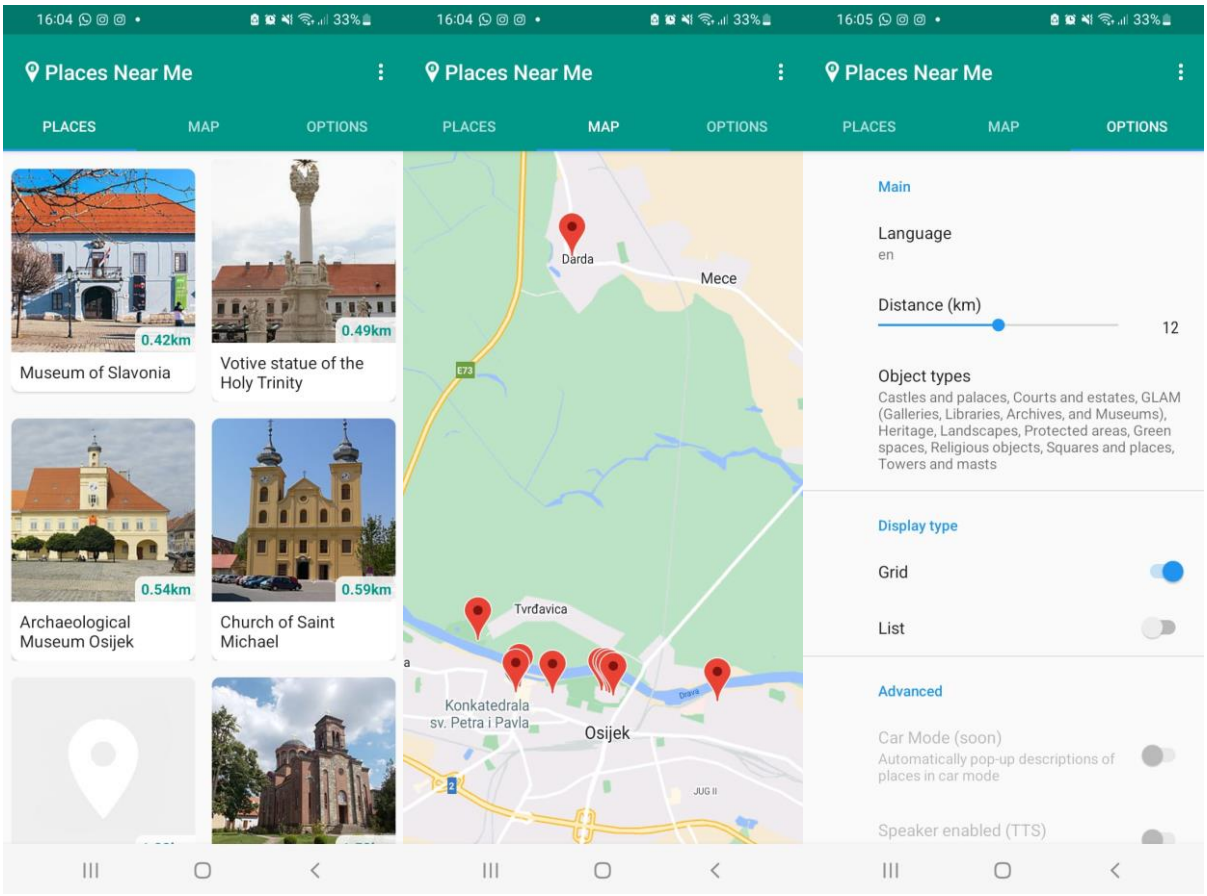
2.3.1. Pregled područja teme

Pregledom Google Play platforme može se utvrditi kako zapravo i ne postoji veliki broj aplikacija koje kreiraju put obilaska znamenitosti. Većina turističkih aplikacija zapravo služi za rezerviranje smještaja ili prijevoza. Postoji mali broj aplikacija koje pokazuju znamenitosti i atrakcije, ali ne slažu rutu obilaska istih.

Za korisnike iOS sustava situacija je ista. Međutim, oni nisu ciljana skupina u ovom radu i za ovu aplikaciju. Kreatori svih pregledanih aplikacija baziraju se uglavnom na geolokaciji i bazama podataka. Aplikacije su, uglavnom na svjetskoj razini napravljene uz pomoć lokalnih turističkih zajednica i hotela u određenom mjestu. Gotovo sve aplikacije imaju mogućnost ostavljanja komentara i/ili dodavanje novih turističkih atrakcija.

2.3.2. Places Near Me

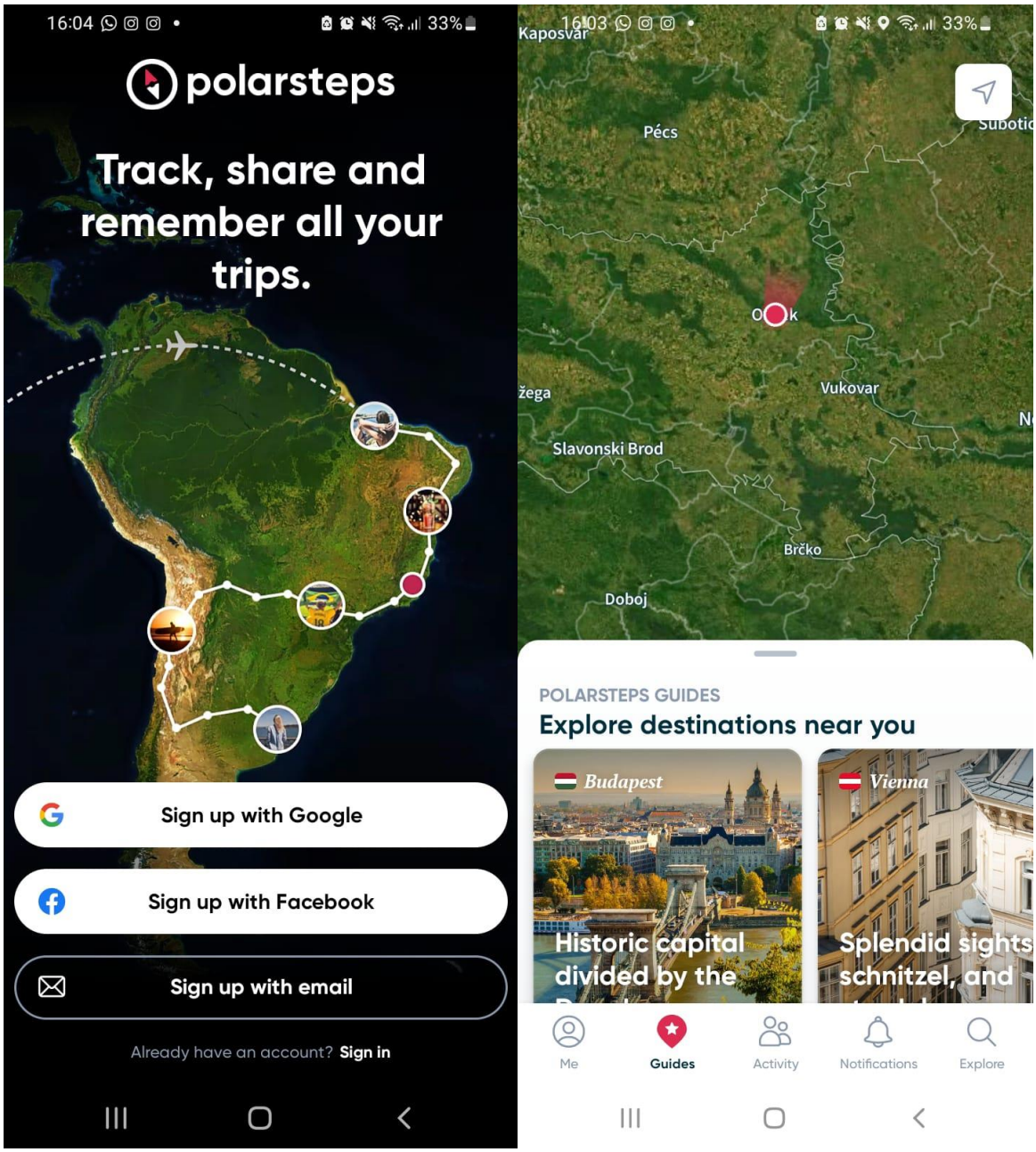
Places Near Me mobilna je Android aplikacija koja je najsličnija aplikaciji u ovom radu. Aplikacija nudi prikaz znamenitosti u određenom mjestu i u određenom kilometarskom radijusu. Također, aplikacija prikazuje udaljenost od trenutne lokacije do određene turističke znamenitosti. Aplikaciji nedostaje kreiranje rute obilaska ponuđenih znamenitosti i neka vrsta sortiranja (po razdoblju izgradnje, po stilu izgradnje itd.). Na slici 2.1 vidljiv je izgled aplikacije i neke funkcionalnosti.



Slika 2.1. Prikaz izgleda aplikacija Places Near Me

2.3.3. Polarsteps

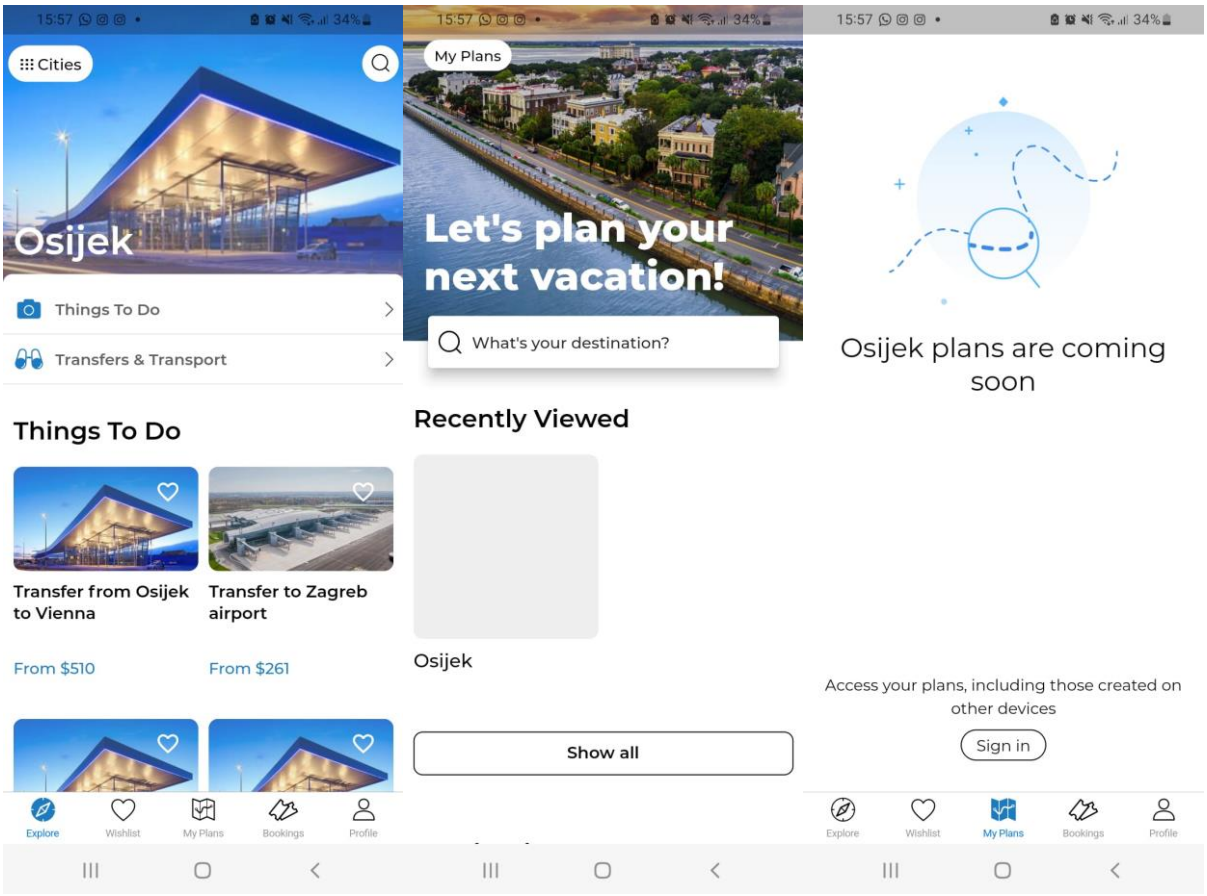
Aplikacija nudi kreiranje vlastite rute kretanja, ali tako da korisnik sam određuje kretanje i obilazak mjesta. Nadalje, aplikacija nudi rutu kretanja prijevoznim sredstvom, ali ne nudi rutu obilaska znamenitosti na toj relaciji. Aplikacija nudi već spremljene prijedloge putovanja s predloženim turističkim atrakcijama, restorana i smještaja, ali ne nudi rutu kretanja. Na slici 2.2 prikazan je izgled aplikacije i neke funkcionalnosti.



Slika 2.2. Prikaz izgleda aplikacija Polarsteps

2.3.4. Visit A City

Aplikacija omogućuje planiranje turistovog cijelog boravka u mjestu. Nudi mogućnost organiziranja i planiranja prijevoza i obilaska popularnih atrakcija grada. Ova aplikacija pokriva svjetsku razinu turističkih sadržaja, ali za neka određena mjesta nema ponuđenog sadržaja, kao primjerice za grad Osijek. Postoji mogućnost stvaranja to-do popisa. Na slici 2.3. prikazan je izgled aplikacije i neke funkcionalnosti.



Slika 2.3. Prikaz izgleda aplikacija Visit A City

3. MODEL I ARHITEKTURA APLIKACIJE U KOOPERATIVNOM STVARANJU I PREPORUČIVANJU RUTA OBILASKA TURISTIČKIH ZNAMENITOSTI

3.1. Funkcijski i nefunkcijski zahtjevi za mobilnu aplikaciju za kooperativno stvaranje i preporučivanje ruta obilazaka turističkih znamenitosti

Kako bi se jednostavnije prikazao rad i funkcionalnosti aplikacije, u ovom će dijelu biti prikazan dijagram rada aplikacije. Prilikom otvaranja aplikacije korisnik može vidjeti zaslon s dva gumba. Prvi gumb „Prijavi se“ vodi korisnika na zaslon za prijavu na već postojeći korisnički račun u bazi podataka, dok gumb „Registriraj se“ vodi korisnika na zaslon za izradu novog korisničkog računa koji se sprema u bazu podataka.

Funkcijski zahtjevi na aplikaciji su:

- Ulaskom u aplikaciju korisnik odlučuje hoće li se logirati na već postojeći korisnički račun ili će kreirati novi korisnički račun
- Prijava u korisnički račun
- Kreiranje novog računa
- Spremanje računa i podataka u bazu podataka
- Korisnik može izabrati neku od ponuđenih opcija u izborniku
- Korisnik može odabrati spremljene znamenitosti i odabrati vremensko razdoblje koje ga zanima
- Korisnik može dodati znamenitost u bazu podataka
- Korisnik može odabrati najposjećenije znamenitosti u vremenskom razdoblju koje ga zanima
- Korisnik može sam odrediti rutu obilaska znamenitosti po vlastitom izboru
- Dohvaćanje korisnikove geolokacije
- Prikaz znamenitosti na Google Maps-u

Nefunkcijski zahtjevi su, prema [4], zahtjevi koji specificiraju kriterije koji se koriste za prosuđivanje rada sustava, a ne njegovo specifično ponašanje. Odnosno, ako funkcijski zahtjev definira što sustav radi, nefunkcijski zahtjev definira kako sustav treba raditi. Također, on je jamac kvalitete usluge (naziva se ugovor o razini usluge) koji se stvara prilikom izrade aplikacije.

Aplikacija za kooperativno stvaranje i preporučivanje ruta obilaska turističkih znamenitosti cilja na SDK 31, odnosno korisnici Android inačice 12.0 ili niže, u mogućnosti su koristiti ovu

aplikaciju, ali minimalni SDK joj je 19, odnosno API 19 što odgovara inačici Android 4.4 (KitKat). Navedeni Android 12.0 je najnovija inačica Android OS-a za vrijeme pisanja ovog rada. Vrijeme odziva i pokretanja aplikacije je manje od 1s.

Kako bi se mogla kreirati odgovarajuća ruta obilaska znamenitosti korisnik treba odobriti dozvolu za uporabu lokacije uređaja, inače aplikacija neće raditi u punoj mogućnosti. Nakon odobrenja lokacije, dohvaća se lokacija mobilnog uređaja.

Kako bi se mogle stvarati preporuke za obilazak znamenitosti programer mora dohvatiti API ključ od Google Maps-a, kojeg korisnik predaje (bez njegovog znanja) preko Google računa i Google Cloud Platform-e. Nakon što ga je programer nabavio koristi ga za implementiranje koda potrebnog za rad aplikacije.

Na slici 3.1 prikazan je dijagram rada aplikacije. Na dijagramu je opisan postupak što sve korisnik može napraviti te funkcionalnosti aplikacije.

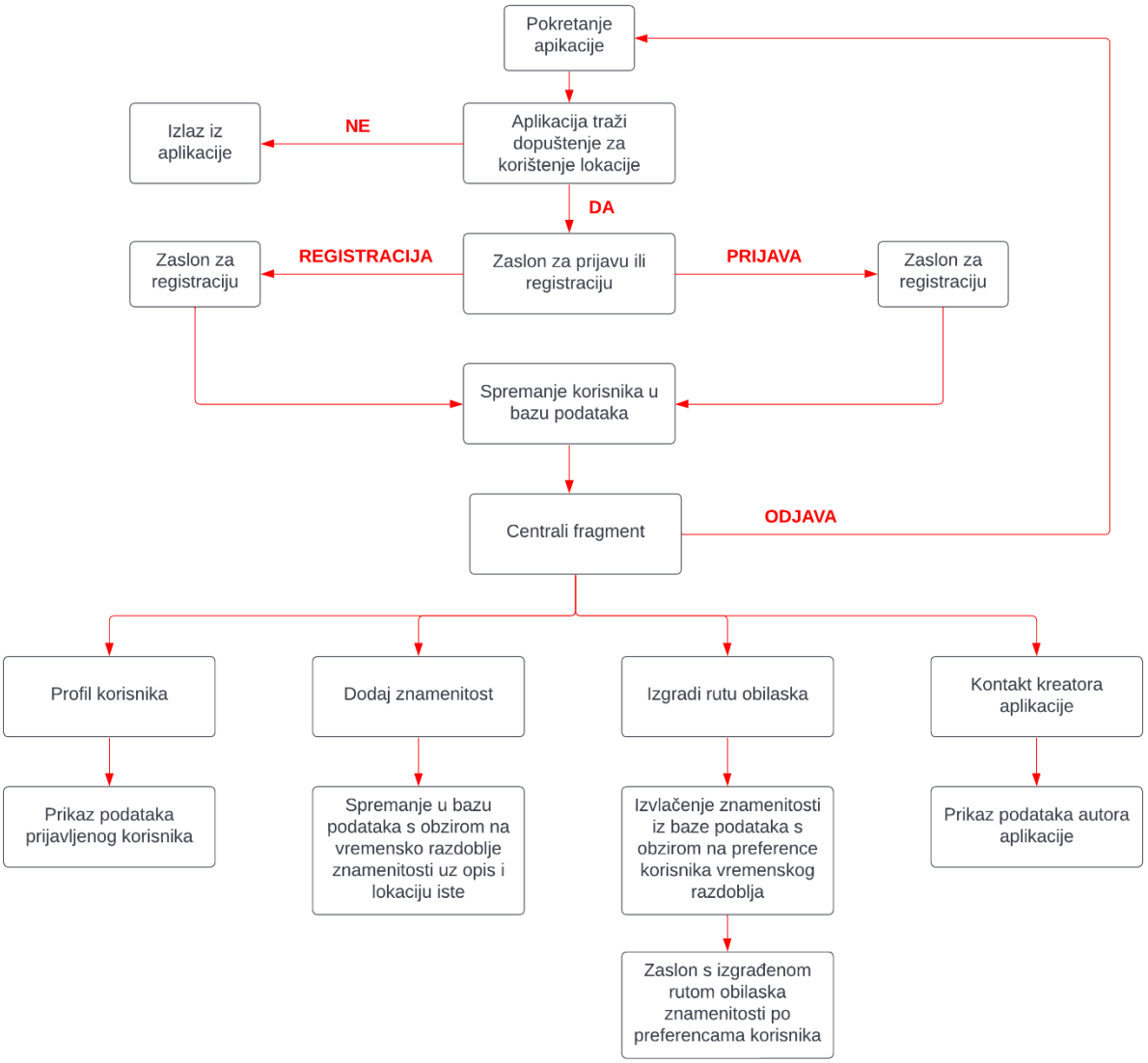
Zbog urednosti i lakšeg čitanja dijagrama rada izostavljene su povratne veze iz svakog zaslona. Na primjer, iz svakog zaslona do kojeg se dođe preko centralnog putem tipke za nazad (engl. *back button*).

3.2. Potrebni procesi i programski pristup za ostvarivanje aplikacije

Pod navedenim podnaslovom opisat će se algoritam za odlučivanje i pozadinske radnje koje su nevidljive korisniku i za koje on ne zna da se događaju.

3.2.1. Dohvaćanje lokacije mobilnog uređaja korisnika

Prema [5], prvotno je bitno korisniku omogućiti odabir hoće li dopustiti aplikaciji da upotrebljava njegovu geolokaciju ili neće. Ukoliko korisnik dopusti upotrebu lokacije, mobilni uređaj uključuje GPS uređaj i preko satelita se dohvaća lokacija. Pomoću platforme od Google Maps-a (API-a, odnosno alata za razvoj softver-a), te pomoću SDK-a (kako bi programeri mogli u svoje aplikacije ugraditi Google Maps ili primati podatke/informacije). Zbog teme ovog završnog rada, lokacija i Google Maps najbitnije su stavke aplikacije kako bi se mogle kreirati rute kretanja i spremati lokacije znamenitosti u bazu podataka.



Slika 3.1. Dijagram tijeka rada aplikacije

3.2.2. Postupak kreiranja rute kretanja ovisno o preferencijama korisnika, kooperativno stvaranje i preporučivanje ruta obilaska

Svaki turist danas želi imati isplaniranu rutu kretanja i obilaska nekog mjesta tako da ima što manje "praznog hoda", odnosno da u ograničenom vremenu koje provodi na određenoj lokaciji obiđe što veći broj znamenitosti, atrakcija itd. koje turista zanimaju.

Aplikacija daje korisniku mogućnost odabira vremenskog razdoblja kroz povijest te mu aplikacija vraća listu znamenitosti i najpovoljniju rutu obilaska znamenitosti s obzirom na njegovu lokaciju i odabire. Prvotno su predefinirane neke znamenitosti, njihov opis, ocjena i lokacija gdje se nalaze,

međutim, i korisnik sam može unijeti znamenitost (kako bi proširio listu znamenitosti koje se nalaze u tom vremenskom razdoblju u mjestu u kojem se nalazi).

Korisnik može izabrati preporučenu rutu na osnovu ocjena znamenitosti drugih korisnika i njihovog iskustva odnosno svaki korisnik prilikom ocjenjivanja posjećenih znamenitosti preporučuje znamenitosti drugom korisniku. Na taj način se doprinosi kooperativnom stvaranju i preporučivanju ruta i znamenitosti.

Primjerice, ukoliko korisnika zanimaju znamenitosti iz doba secesije, odabrat će to vremensko razdoblje te dobiti listu i kratki opis za svaku pojedinačnu znamenitost iz tog vremenskog razdoblja. Ukoliko je korisnik osoba koja živi u nekom turističkom mjestu, aplikaciju može koristiti kako bi dodao znamenitosti i informacije o istima koje se ne nalaze prethodno u bazi podataka aplikacije te time može povećati listu znamenitosti i na taj način obogatiti turistovo iskustvo, a time povećati šanse povratka turista u nekom mjesto ili turist može preporučiti mjesto prijateljima, obitelji itd. zbog dobrog vlastitog iskustva.

3.2.3. Baza podataka

Aplikacija radi pomoću baze podataka Firebase. Baza podataka koristi se radi spremanja korisničkih račun prilikom registracije ili kako bi se korisnik prijavio na svoj račun ukoliko on postoji. Prilikom registracije korisnik mora unijeti e-mail i lozinku pomoću kojih se kasnije prijavljivljuje te nadimak. Za ostvarivanje korišten je FireBase Authentication. Za spremanje znamenitosti korišten je FireBase Storage za pohranu slika te FireBase Realime Database za pohranu podataka o znamenitosti. Baza podataka izgleda poput stabla gdje se podaci spremaju šifrirani nasumičnom kombinacijom brojeva i slova kao djeca (engl. *child*) pod nazivom znamenitosti (engl. *sights*) roditeljskom (engl. *parent*) koji je u ovom slučaju baza podataka.

Za prikaz znamenitosti iz baze podataka, ovisno o odabiru korisnika, povlače se slike i opisi znamenitosti koji se nalaze u tom vremenskom razdoblju te se kreira ruta obilaska tih znamenitosti.

3.3. Arhitektura mobilne Android aplikacije

U navedenom poglavlju opisat će se korisniku vidljiv dio (engl. *frontend*) i korisniku nevidljiv dio, odnosno ono što se događa u pozadini aplikacije (engl. *backend*). Arhitektura aplikacije je MVVM model (Model-View-View-Model), radi jednostavnosti čitanja koda i bolje implementacije.

Dio vidljiv korisniku sadrži:

- Početni zaslon – zaslon s dva gumba koji vode na određeni zaslon ovisno o stisnutom gumbu (engl. *button*), odnosno na prijavu ili registraciju korisnika
- Glavni zaslon – zaslon s pozadinskom slikom, *editText* i *textView* sadržajima koji vode na ostale zaslone aplikacije, te s gumbom za odjavu koji vodi nazad na početni zaslon
- Profilni zaslon – prikaz podataka korisnika koji je trenutno prijavljen u aplikaciju
- Zaslon za dodavanje znamenitosti – zaslon koji služi kako bi se spremila nova znamenitost u bazu podataka
- Zaslon za izgradnju rute – zaslon s mogućnosti odabira vremenskog razdoblja i prikaz liste znamenitosti te gumb za prikaz rute obilaska
- Zaslon s kontaktom – zaslon s podacima za kontaktiranje autora aplikacije
- *Google Maps* zaslon – zaslon na kojem se nalazi karta mjesta gdje se koristi aplikacija i prikazuje oznake lokacija gdje se nalaze znamenitosti

Dio nevidljiv korisniku:

- *Firebase database* – baza podataka za spremanje korisnikovih informacija i svih podataka o znamenitosti
- Dohvaćanje lokacije – dohvaća korisnikovu geolokaciju nakon dopuštenja korisnika
- Algoritam za kreiranje optimalne rute obilaska znamenitosti – algoritam pomoću kojeg se određuje najpovoljnija ruta obilaska znamenitosti po korisnikovim odabirima i njegovoj geolokaciji

4. PROGRAMSKO RJEŠENJE MOBILNE APLIKACIJE ZA KOOPERATIVNO STVARANJE I PREPORUČIVANJE RUTA OBILAZAKA TURISTIČKIH ZNAMENITOSTI

4.1. Korištene programske tehnologije, razvojne okoline i jezici kako bi se aplikacija napravila

U ovom dijelu rada teorijski su predstavljeni alati koji su korišteni za razvoj aplikacije za kooperativno stvaranje i preporučivanje ruta obilaska turističkih znamenitosti.

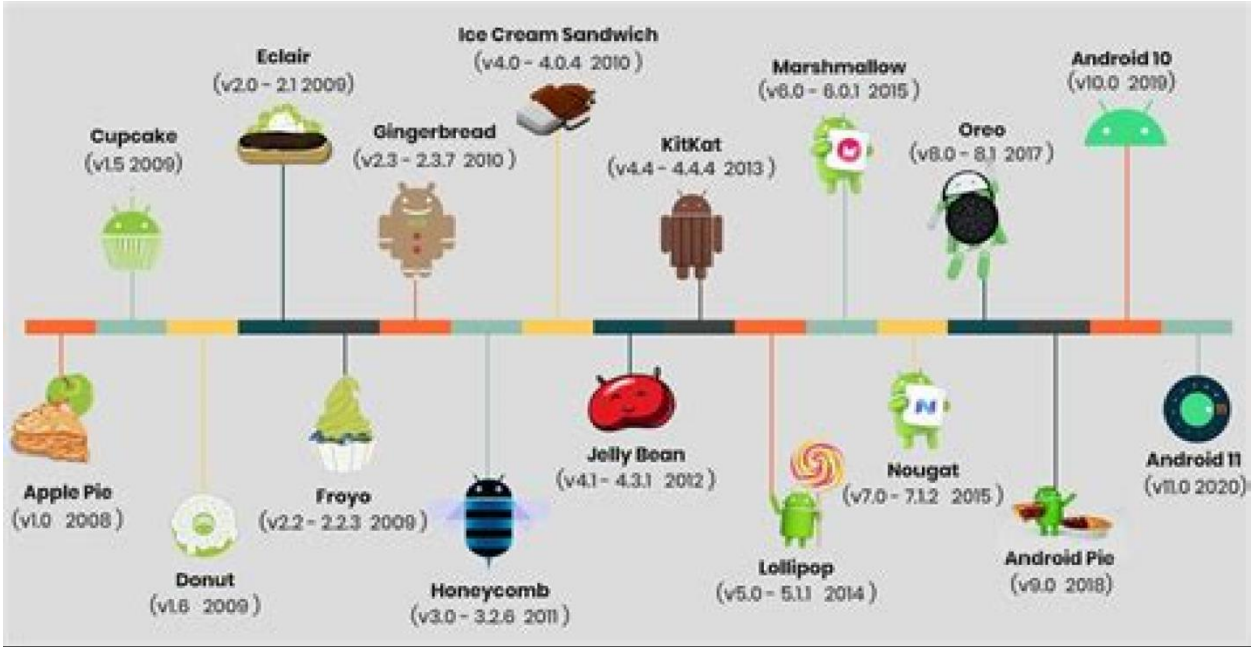
4.1.1. Operacijski sustav Android

Prema [6], Android operacijski sustav jedan je od najčešće korištenih sustava za mobitele, televizore, tablete itd. Temelji se na Linux jezgri kojeg je razvio Google.

Zbog otvorenosti programskog koda postao je najbrže rastući mobilni operacijski sustav te je zbog svoje otvorenosti postao omiljen i za potrošače i za programere. Još jedan razlog tome je što svaki proizvođač ili korisnik može svojevoljno sam promijeniti kod i prilagoditi ga sebi ili svojim korisnicima.

Zapravo, operacijski sustav Android je posrednik između korisnika i uređaja na kojima se koriste i pružaju razne mogućnosti preko kojih korisnik može upravljati uređajem. Na primjer, ukoliko korisnik želi pojačati zvuk stisnut će gumb, a sustav će narediti uređaju da to i napravi. Inače, sve inačice Android sustava nazvane su po slatkim proizvodima, odnosno slatkišima (KitKat, Oreo, Donut itd.).

Na slici 4.1 prikazane su inačice Androida, međutim nedostaje inačica 12.0 [7].

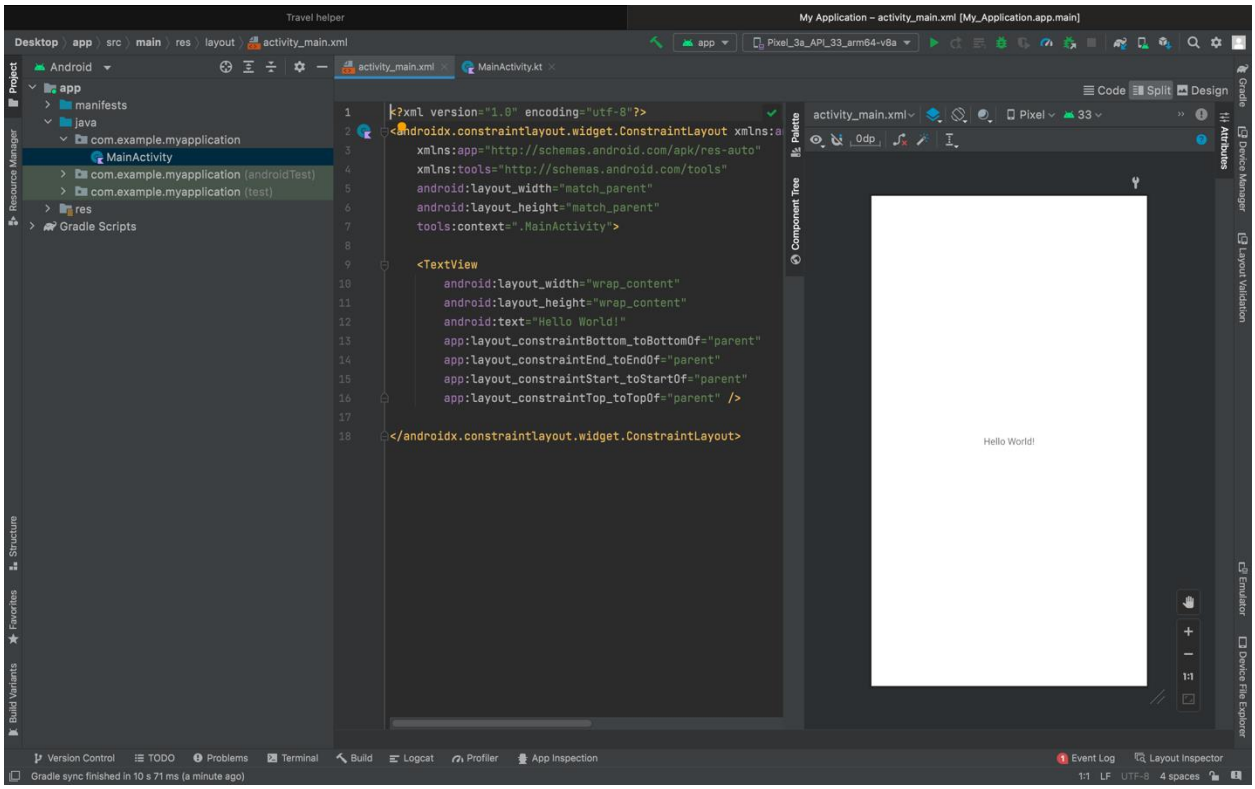


Slika 4.1. Dosadašnje inačice Androida [7]

4.1.2. Razvojno okruženje Android Studio

Android Studio integrirano je razvojno okruženje (IDE) za razvoj Android aplikacija i temelji se na ItelliJ IDEA, odnosno Java integriranom razvojnom okruženju za softver koji uključuje alate za uređivanje koda i razvoj za programere. Služi za stvaranje android aplikacija koje se mogu koristit na gotovo svim pametnim uređajima poput telefona, televizora, tableta itd. Dostupan je za sve operacijske sustave, a moguće je pisati u dva programska jezika. To su trenutno popularniji Kotlin i Java. Postoje brojne funkcionalnosti Android Studija, a neke od njih su: gradle sustav, emulator, refaktoracija koda, integracija s GitHub i GitLab i drugima. Kako bi aplikacija bila dostupna na Trgovini Google Play potrebno ju je prevoditi [8].

Na slici 4.2 prikazano je razvojno okruženje Android Studio prilikom kreiranja novog projekta. Vidljivo je kako program automatski kreira *Main activity* kao Kotlin i XML *file*.

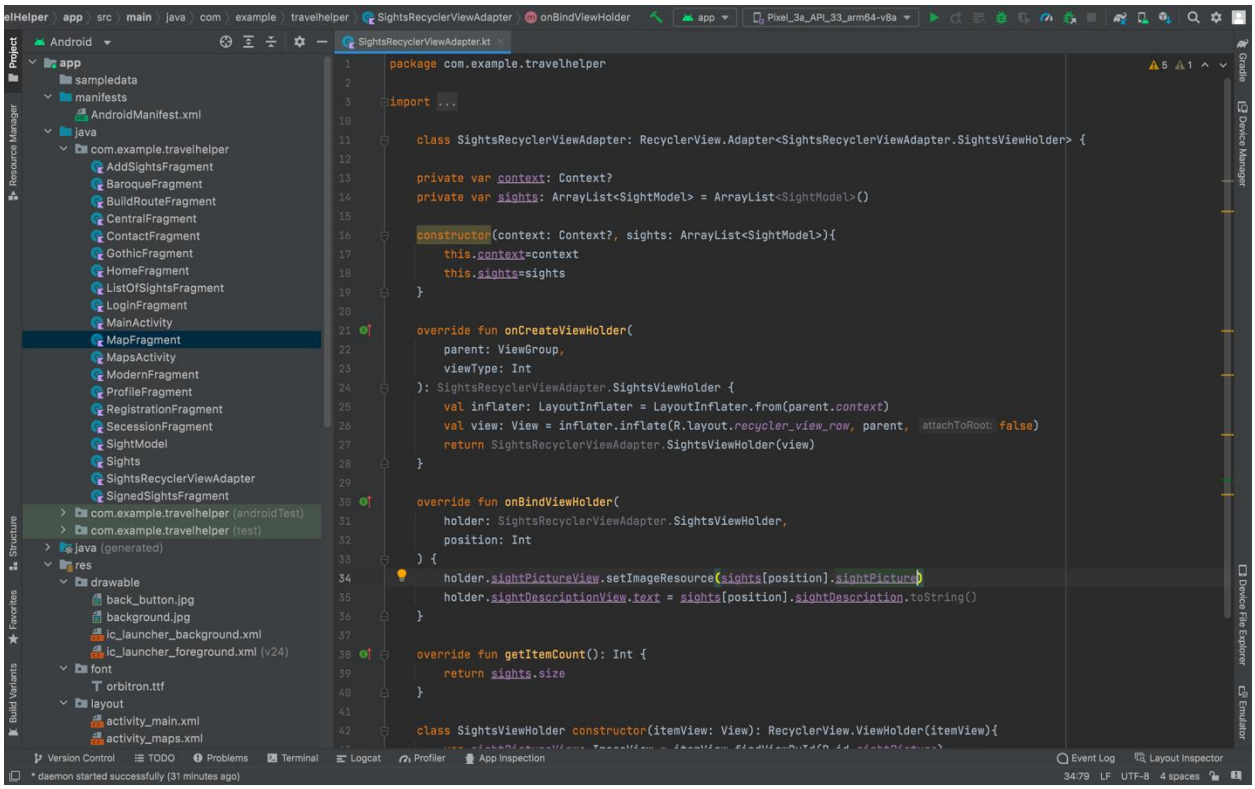


Slika 4.2. Razvojno okruženje Android Studio

4.1.3. Programski jezik Kotlin

Prema [9], Kotlin je moderni jezik sa statičkim sustavom tipova koji popravlja veliki broj Java problema, kao što su naprimjer iznimke nultog pokazivača, pretjerana opširnost koda i druge. Napravljen je na primjeru Swifta, Scala, C# itd. Dobra stvar kod Kotlina je što ima aktivnu zajednicu koja poboljšava jezik iz dana u dan. Također, Kotlin je siguran, pametan, snažan i svestran programski jezik koji ima značajnu interoperabilnost s programskim jezicima Java i JavaScript.

Kotlin se može koristiti u razne svrhe poput izrade Android aplikacija, pozadine (engl. *backend*), spajanje s bazom podataka itd. Na slici 4.3 prikazan je primjer programskog jezika Kotlin u Android aplikaciji iz rada.



Slika 4.3. Primjer programskog jezika Kotlin

4.1.4. XML

Prema [10], eXtensible Markup Language (skraćeno XML) je označni (engl. *markup*) jezik koji je dizajniran za pohranu i prijenos podataka te kako bi se organizirale stranice, aplikacije i slično. Jednostavnije rečeno, XML se koristi kako bi se dizajnirao zaslon mobitela na aplikaciji ili izgled web stranice/aplikacije, međutim on nije zamjena za HTML.

XML je samoopisni jezik, a njegova je prednost što se istovremeno može prikazati i kod i zaslon i može biti pisan u više vrsta *layouta*. U te se *layoute* dodaju elementi zaslona poput teksta, gumbova, lista, slika itd. Pristup elementima vrši se pomoću ID-a elementa. Svakom elementu se mogu mijenjati značajke poput boje, veličine, fonta itd. XML je neovisan o platformi i o programskom jeziku preko kojeg će se koristiti *layout*.

4.1.5. Firebase

Firebase kao platforma služi za izgradnju, poboljšanje i razvoj aplikacija te pokriva značajan dio usluga koje olakšavaju programeru rad jer ih ne mora sam napraviti. Primjer toga jest autentifikacija, baza podataka, pohrana datoteka, *push* poruke i brojne druge.

Usluge Firebase pohranjene su u oblaku (engl. *cloud*) računala i zbog toga nema potrebe za pisanjem međusloja koda između usluga na poslužiteljskoj strani i aplikacija. Zbog toga se upiti na bazu mogu direktno slati iz klijentske aplikacije [11].

4.2. Programska rješenja na strani korisnika

U dijelu programskog rješenja vidljivog korisniku bit će opisane sve funkcionalnosti koje su vidljive korisniku i na koje korisnik može utjecati. Sve funkcionalnosti bit će popraćene kodom aplikacije.

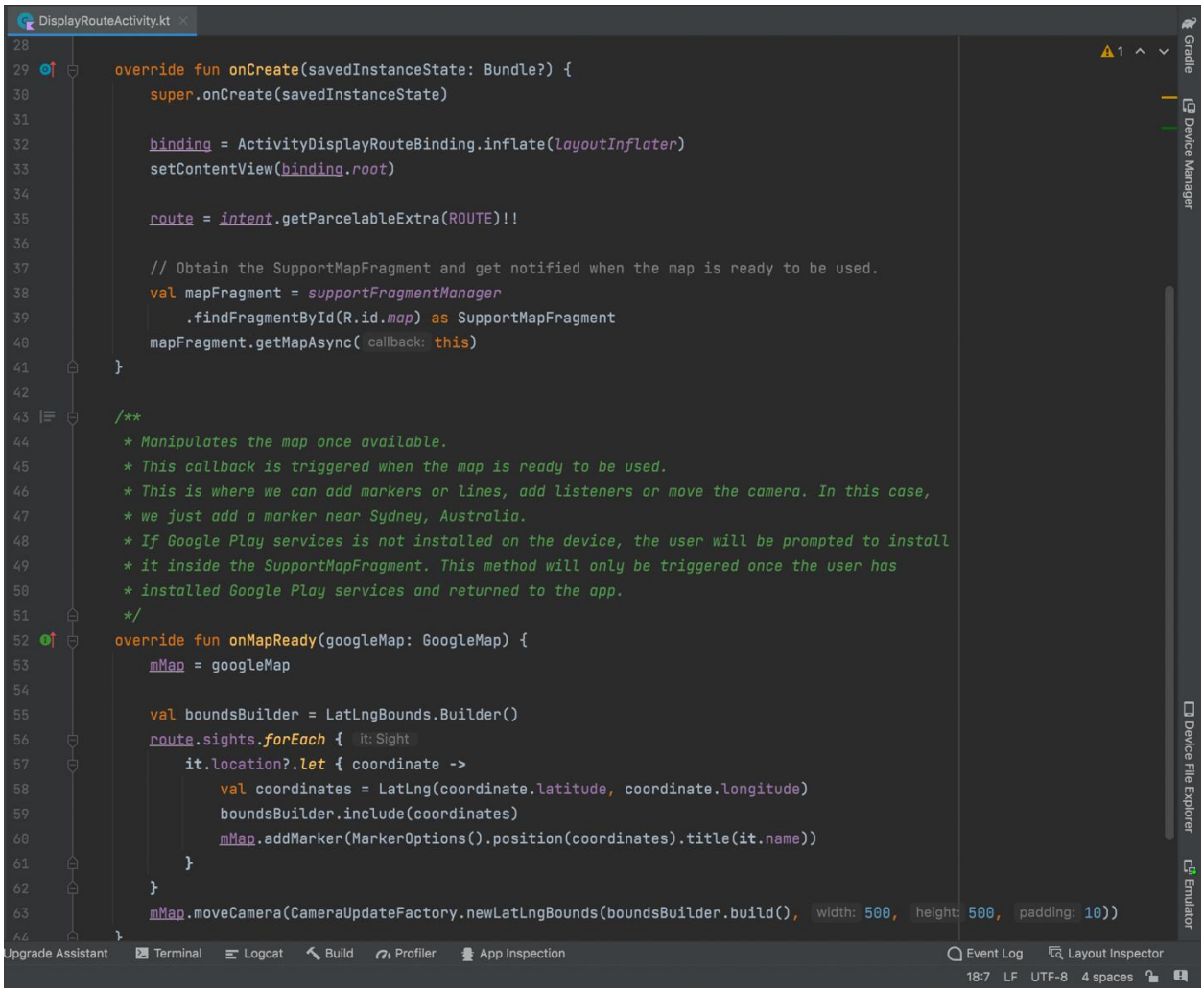
4.2.1. Prikaz rute i Google Maps Activity

Korištenje Google Maps-a zahtjeva implementaciju Google Maps API ključa koji se traži preko Google korisničkog računa na platformi zvanoj Google Cloud Platform.

Google Maps otvara se nakon pritiska gumba u zaslonu za kreiranje rute naziva „CREATE ROUTE“. Na zaslonu se tada prikazuju lokacije korisnikove odabrane znamenitosti u obliku pribadača.

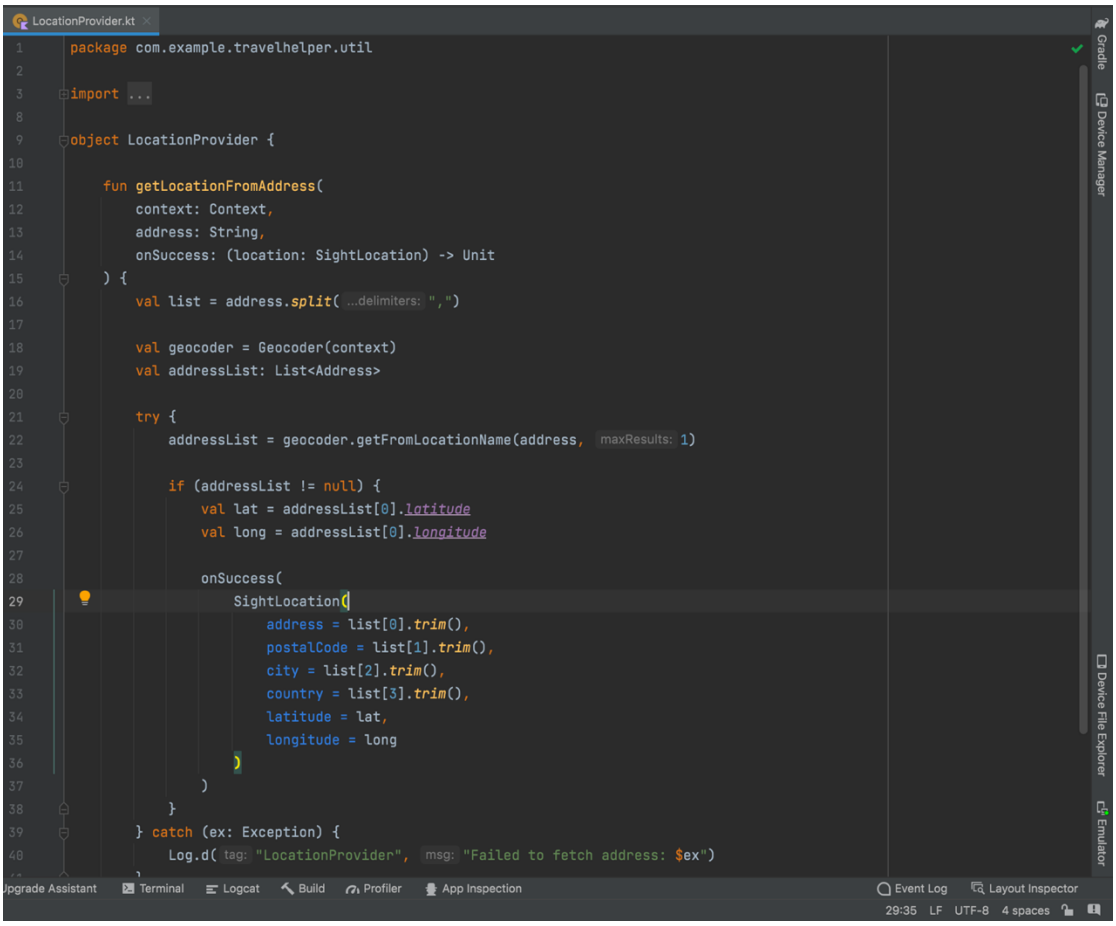
Te lokacije, odnosno adrese znamenitosti, spremljene su u bazu podataka, odakle se povlači samo adresa i prikazuje unutar Google Maps-a.

Na slici 4.4 prikazano je korištenje aktivnosti (engl. *activity*) Google Mapsa.



Slika 4.4. Prikaz korištenja Google Maps Activity

Dohvaćanje znamenitosti vrši se pomoću dohvaćanja adrese znamenitosti i geografske širine i dužine te adrese na kojoj se znamenitost nalazi. Prvotno joj se mora dodati kontekst i adresu. Za dohvaćanje koordinata iz adrese koristi se funkcija *geocoder*, što je Google-ova klasa. Klasa ima funkciju *getFromLocationName* kojoj se za rad predaje adresa i broj rezultata kojeg funkcija treba vratiti. Funkcija se nalazi u *try-catch* petlji kako aplikacija ne bi prestala raditi ukoliko se upiše kriva adresa, zbog nedostatka interneta ili slično. Programski kod za to prikazan je na slici 4.5.



Slika 4.5. Prikaz koda pomoću kojeg se dohvaća lokacija znamenitosti

4.2.2. Početni zaslon

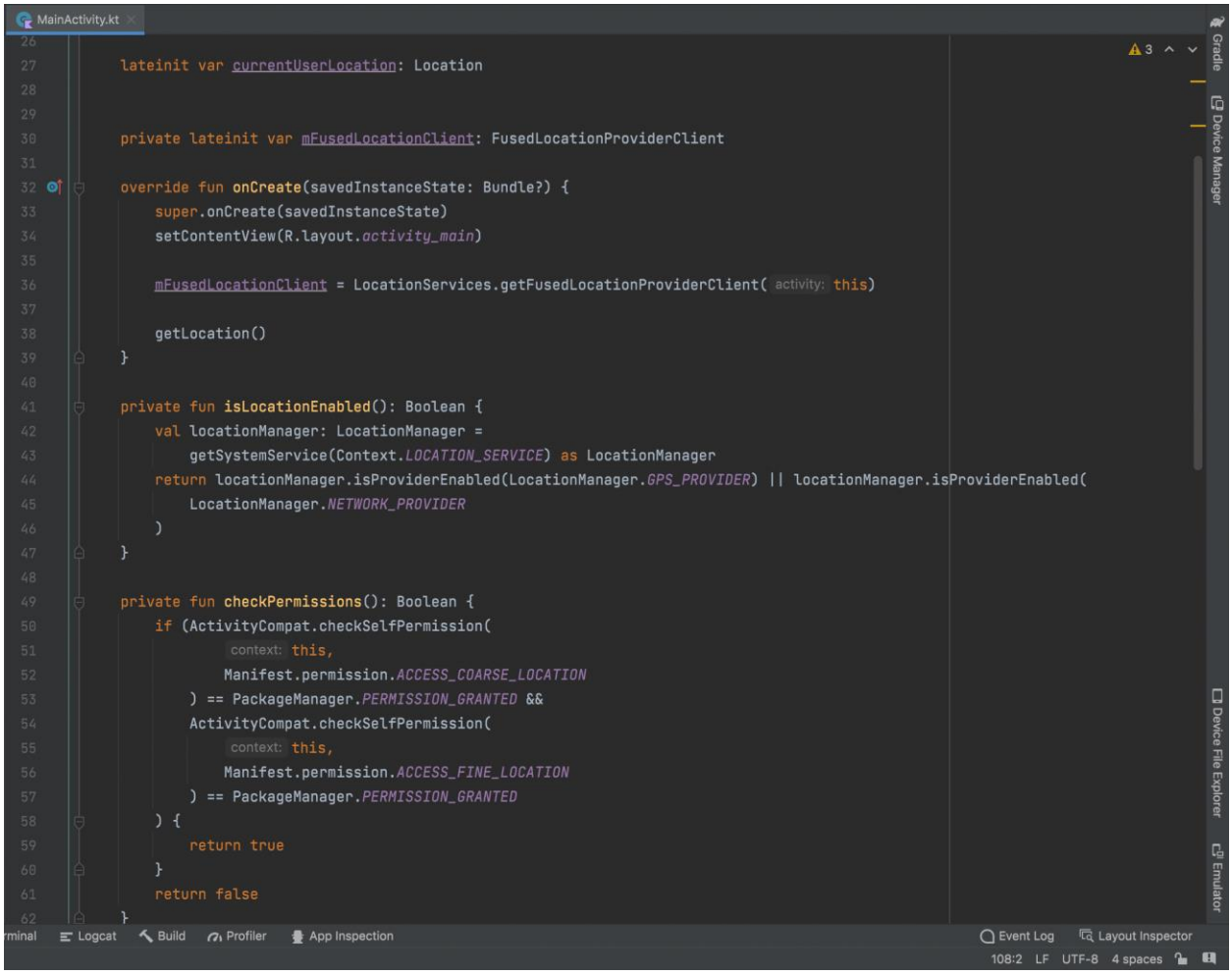
Mijenjanje zaslona izvedeno je putem funkcije *navigateTo*(fragment: Fragment). Funkcija prima argument fragment koji predstavlja fragment na koji se korisnik želi navigirati. Također, uz prebacivanje na novi zaslon, funkcija sprema stari zaslon na *backstack* kako bi se omogućio povratak na prošli zaslon pritiskom na *back* strelicu. Rad ove funkcije prikazan je slikom 4.6.

Slika 4.6. Prikaz *navigateTo* funkcije

4.2.3. Dohvaćanje lokacije korisnika

Kako bi aplikacija ispravno radila potrebno je koristiti lokaciju korisnika, koju prvotno mora odobriti. Za ostvarivanje mogućnosti potrebno je inicijalizirati lokacijske usluge. Funkcija *isLocationEnabled* provjerava je li korisnik odobrio korištenje lokacije te vraća vrijednost *true* ukoliko je, odnosno *false* ukoliko nije. Provjera dopuštenja vrši se u funkciji *checkPermissions*. Kada korisnik dopusti aplikaciji korištenje njegove geolokacije, funkcija *getLocation* dohvaća korisnikovu geografsku dužinu i geografsku širinu [12].

Traženje dopuštenja za uporabu lokacije prikazan je kodom na slici 4.7.



Slika 4.7. Traženje dopuštenja za korištenje lokacije

4.2.4. Dodavanje nove znamenitosti i kooperativno stvaranje preporuka na osnovu ocjene znamenitosti

Dodavanje znamenitosti vrši se pomoću funkcije *addSight* koja unesene podatke dohvaća i sprema u bazu podataka pod jedinstvenom ID vrijednosti. Referenca na tablicu znamenitosti (engl. *Sights*) u bazi podataka naziva se *dbSights*. Funkcija kreira jedinstveni ID za svaku novu znamenitost te ju sprema u bazu podataka. Pomoću reference *storageSights* na Firebase Storage sprema se slika nove znamenitosti u bazu podataka.

Funkcija *addSight* prikazana je slikom 4.8.

```
48    fun addSight(imageUri: Uri, sight: Sight) {
49        sight.id = dbSights.push().key
50
51        storageSights.child(sight.id!!).putFile(imageUri).addOnSuccessListener {  it: UploadTask.TaskSnapshot!
52            dbSights.child(sight.id!!).setValue(sight).addOnCompleteListener { task ->
53                if (task.isSuccessful) _addSightSuccess.value = Unit else _addSightFail.value =
54                    task.exception?.message
55            }
56        }
57    }
```

Slika 4.8. Dodavanje nove znamenitosti

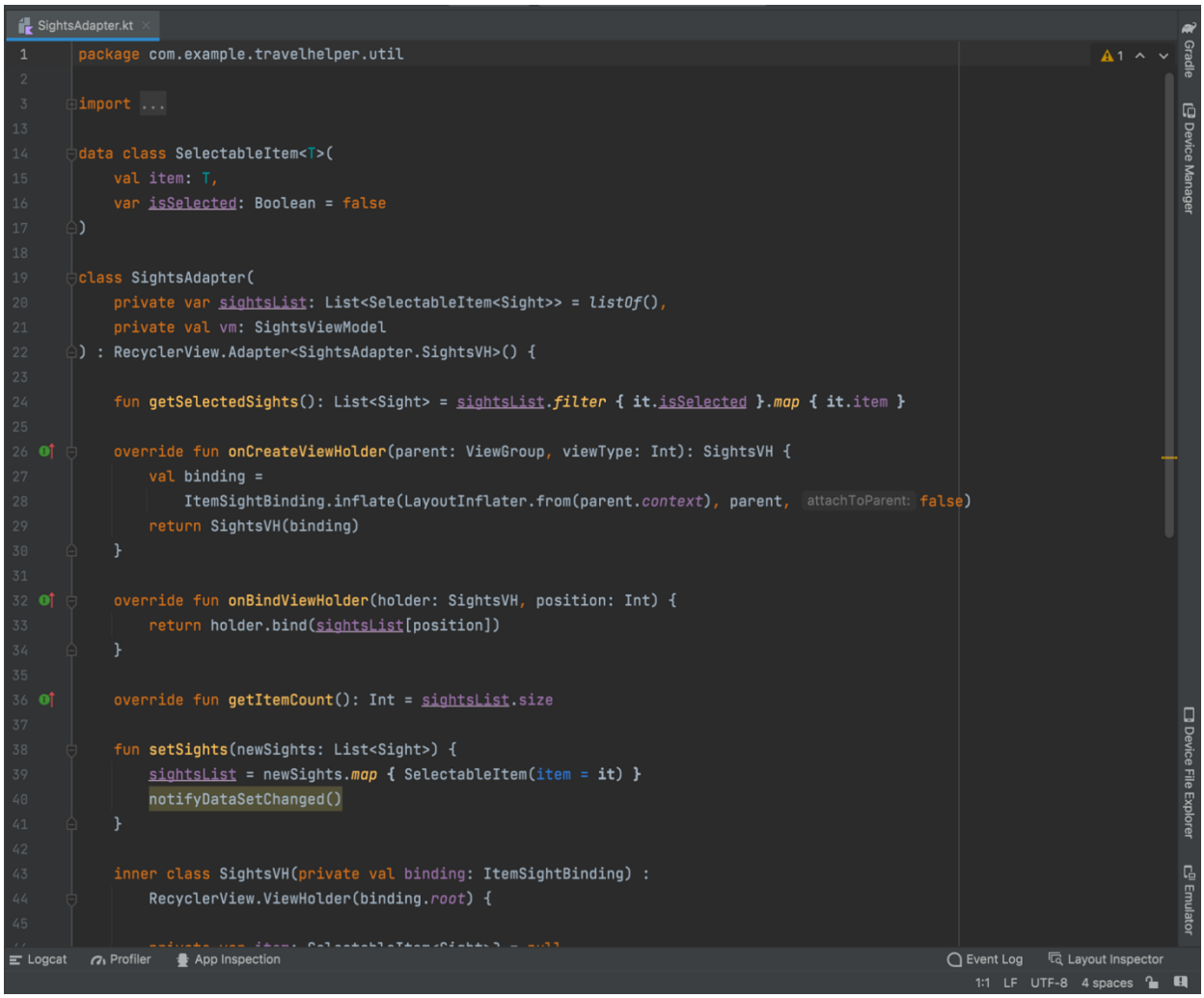
Prilikom dodavanja nove znamenitosti korisnik ima mogućnost ocjenjivanja iste preko XML-ove komponente naziva *RatingBar*. Od komponente se uzima atribut *rating* te se sprema u bazu podataka skupa s ostalim podacima. Ocjenjivanje znamenitosti služi za kooperativno stvaranje i preporučivanje rute obilazaka turističkih znamenitosti. Na taj način korisnik ima mogućnost preporučiti rutu drugim korisnicima s obzirom na ocjenu znamenitosti koju je dao prilikom dodavanja nove znamenitosti. Svaki korisnik zbog ocjena znamenitosti i u ovisnosti o udaljenosti može stvoriti rutu obilaska znamenitosti po svojim osobnim interesima. *RoundToInt* zaokružuje vrijednost kako ne bih moglo biti pola ocjene s decimalnim iznosom, već zaokruženi iznos od 1 do 5.

Navedeno je prikazano slikom 4.9.

Slika 4.9. Prikaz metode da ocjenjivanje znamenitosti

4.2.5. Izgradnja rute

Korisnik rutu izgrađuje odabirom znamenitosti koje ga zanimaju te nakon pritiska gumba povlači se adresa znamenitosti iz baze i prikazuje se na mapi. Funkcija *getSelectedSights* provjerava koje je znamenitosti korisnik odabrao iz baze podataka. Nakon odabira, pritiskom na gumb *Build route*, odabrane se rute spremaju u listu te predaju *Intent* objektu koji služi kako bi se predale odabrane znamenitosti novoj započetoj aktivnosti *activityu BuildRouteActivity*. *BuildRouteAcitivity* implementira Google Maps API te postavlja markere s odabranim lokacijama na mapu. Programski kod za izgradnju rute prikazan je na slici 4.10.



Slika 4.10. Prikaz programskog kod izgradnju rute

4.2.6. Pregled detalja profila

Korisnik ima mogućnost vidjeti svoje podatke na zaslonu s profilom. Funkcija *setupProfile* prikazuje podatke trenutno prijavljenog korisnika. Programski kod funkcije prikazan je slikom 4.11.

```
47      private fun setupProfile() {
48          val currentUser = vm.getCurrentUser()
49          binding.userEmail.text = currentUser?.email
50          binding.usernameOnProfile.text = currentUser?.displayName
51      }
```

Slika 4.11. Funkcija *setupProfile*

4.2.7. Prijava ili registracija

Korisnik ima mogućnost napraviti novi korisnički račun ili se prijaviti na postojeći, ukoliko ga je prvotno napravio. Napravljeni novi račun sprema se u bazu podataka sa svim podacima, a kod prijave se provjerava postoji li u bazi podataka korisnički račun s podacima koji su uneseni za prijavu. Programski kod za prijavu i registraciju prikazan je na slici 4.12 i 4.13.

```
42    fun registerUser(email: String, password: String, username: String) {
43        auth.createUserWithEmailAndPassword(email, password).addOnCompleteListener {  it: Task<AuthResult!>
44            if (it.isSuccessful) {
45                _registrationSuccess.value = Unit
46                setupUsername(username = username)
47            } else {
48                _registrationFail.value = Unit
49            }
50        }
51    }
```

Slika 4.12. Programski kod za registraciju

```
36    fun loginUser(email: String, password: String) {
37        auth.signInWithEmailAndPassword(email, password).addOnCompleteListener {  it: Task<AuthResult!>
38            if (it.isSuccessful) _loginSuccess.value = Unit else _loginFail.value = Unit
39        }
40    }
```

Slika 4.13. Programski kod za prijavu

4.3. Programska rješenja na strani poslužitelja

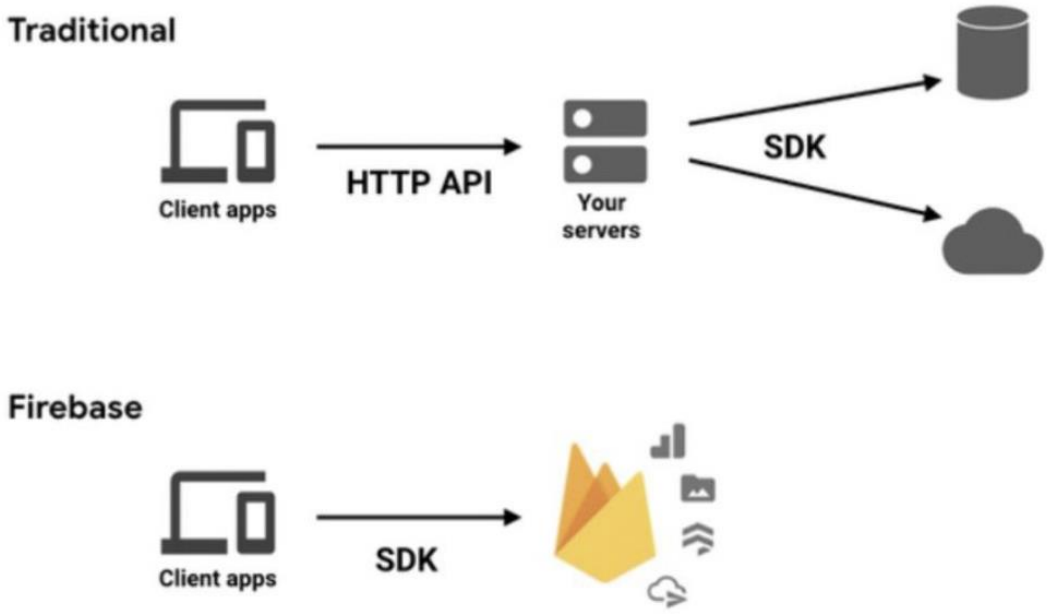
4.3.1. Baza podataka

Za ostvarivanje svih funkcionalnosti ove aplikacije korištena je baza podataka Firebase. Za pohranu slika korištena je Cloud Storage koja je Firebase-ova usluga za pohranu podataka kao što su fotografije, video zapisi, audio zapisi itd.

Za rad sa spremljenim znamenitostima korištena je Realtime Database gdje se spremaju detalji svake znamenitosti - naziva, opisa, adrese i ocjene znamenitosti te ID kodom na sliku znamenitosti na koju se referenciraju prethodno nabrojani podaci.

Kako bi se mogli kreirati korisnički računi ili prijavljivati na postojeće, korištena je Firebase usluga Authentication, gdje se spremaju podaci korisnika i dohvaćaju isti prilikom prijave na račun.

Na slici 4.14 [13] prikazana je struktura baze podataka Firebase gdje se vidi kako za rad s Firebase bazom podataka nema potrebe za pisanjem posredničkog koda. Implementacija baze podataka vrši se povezivanjem usluga Firebase s aplikacijom za koju se koristi na stranici Firebase. Pisanje koda za implementaciju nije potrebno.



Slika 4.14. Struktura baze podataka Firebase [13]

4.3.2. Sortiranje lokacija po blizini

Nakon što korisnik odabere vremensko razdoblje znamenitosti koje ga zanima, aplikacija povlači znamenitosti iz baze podataka za odabrano vremensko razdoblje i prikazuje ih sortirane po blizini. Aplikacija dohvaća trenutnu korisnikovu lokaciju te uspoređuje s lokacijama znamenitosti koje su spremljene. Prilikom pronalaska najbliže znamenitosti postavlja ju kao prvu za prikaz te traži sljedeću najbližu toj prvotnoj lokaciji. Na slici 4.15 prikazan je kod sortiranja znamenitosti.

```
91      private fun sortSightsByNearestLocation(sights: List<Sight>, currentLocation: Location): List<Sight> {
92          val map = mutableMapOf<Location, Sight>(). also { map ->
93              sights.forEach { sight ->
94                  map[Location( provider: ""). also {  it: Location
95                      it.longitude = sight.location!!.longitude
96                      it.latitude = sight.location.latitude
97                  }] = sight
98              }
99          }
100         val sorted = map.toSortedMap(compareBy { it.distanceTo(currentLocation) })
101         return sorted.values.toList()
102     }
```

Slika 4.15. Programski kod za sortiranje znamenitosti po lokaciji

4.3.3. Dohvaćanje lokacije korisnika

Za mogućnost korištenja svih metoda povezanih s lokacijom potrebno je implementirati Google Maps API ključ. Prilikom pokretanja aplikacije od korisnika se traži dopuštenje za korištenje njegove geolokacije. Nakon davanja potrebnih dopuštenja, pokreće se poziv za dohvaćanje lokacije te, ukoliko je uspješan, vraća instancu objekta *Location* iz kojeg se dohvaćaju koordinate korištenjem funkcija *getLatitude()* i *getLongitude()*. Programski kod za upit o dozvoli korištenja lokacije prikazan je slikom 4.16.

```
33      private fun fetchLocation() {
34
35          val task: Task<Location> = fusedLocationProviderClient.lastLocation
36
37          if(ActivityCompat.checkSelfPermission( context: this, android.Manifest.permission.ACCESS_FINE_LOCATION)
38          != PackageManager.PERMISSION_GRANTED && ActivityCompat
39                  .checkSelfPermission( context: this, android.Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED
40          ){
41              ActivityCompat.requestPermissions( activity: this, arrayOf(android.Manifest.permission.ACCESS_FINE_LOCATION),  requestCode: 101)
42              return
```

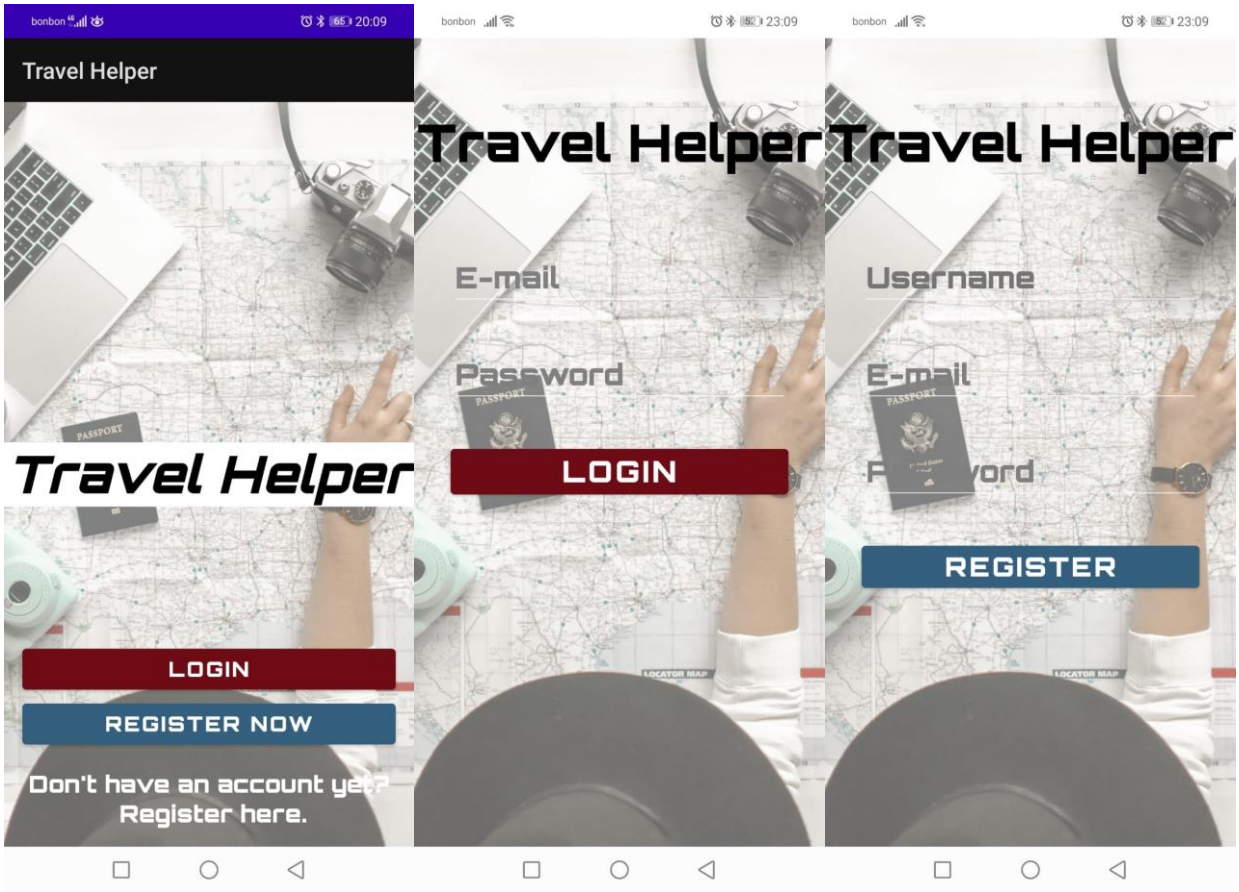
Slika 4.16. Programski kod za upit o dozvoli korištenja lokacije

5. PRIKAZ RADA APLIKACIJE, ANALIZA RADA I ISPITIVANJE REZULTATA

U ovom podnaslovu ukratko su opisane upute za korištenje aplikacije i opisan je rad aplikacije. Također, prikazano je korištenje i rad aplikacije te su ispitane sve njezine mogućnosti. Na kraju ispitivanja aplikacije bit će prokomentirani rezultati ispitivanja.

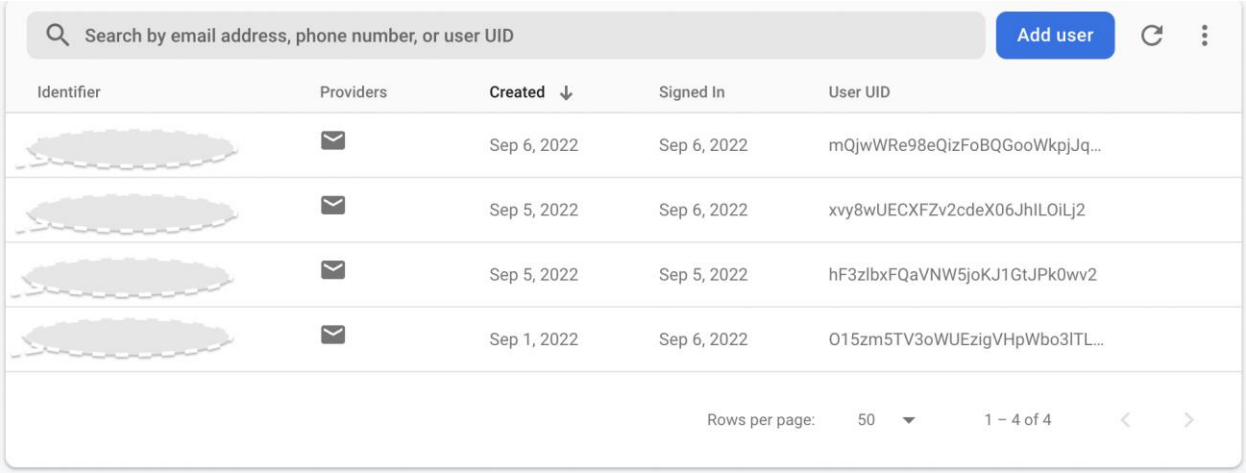
5.1. Upute za korištenje aplikacije za kooperativno stvaranje i preporučivanje ruta obilazaka turističkih znamenitosti

Prilikom pokretanja aplikacije korisnik mora dopustiti aplikaciji pristup lokaciji. U suprotnom aplikacija prestaje s radom. Također, korisnik prilikom ulaska u aplikaciju vidi dva gumba koja vode na zaslone za prijavu (engl. *login*) ili za registraciju korisnika (engl. *register now*) te pozadinskom slikom. Svi ovi zasloni prikazani su na slici 5.1.



Slika 5.1. Početni zaslon, te zaslon za prijavljivanje i registraciju

Na zaslonu za prijavu korisnika potrebno je upisati podatke s kojima se korisnik registrirao u aplikaciju ili se registrirati s novim podacima koje sam izabere. Svi korisnički računi spremaju se u bazu podataka (slika 5.2).



Slika 5.2. Spremljeni registrirani korisnici

Nakon prijavljivanja na već postojeći korisnički račun ili nakon pravljenja novog korisničkog računa (registracije) otvara se glavni zaslon preko kojeg se dolazi do drugih zaslona koji su zapravo funkcija i smisao aplikacije (slika 5.3).

Pritiskom gumba za odjavu korisnik više nije prijavljen i aplikacija se vraća na početni zaslon.

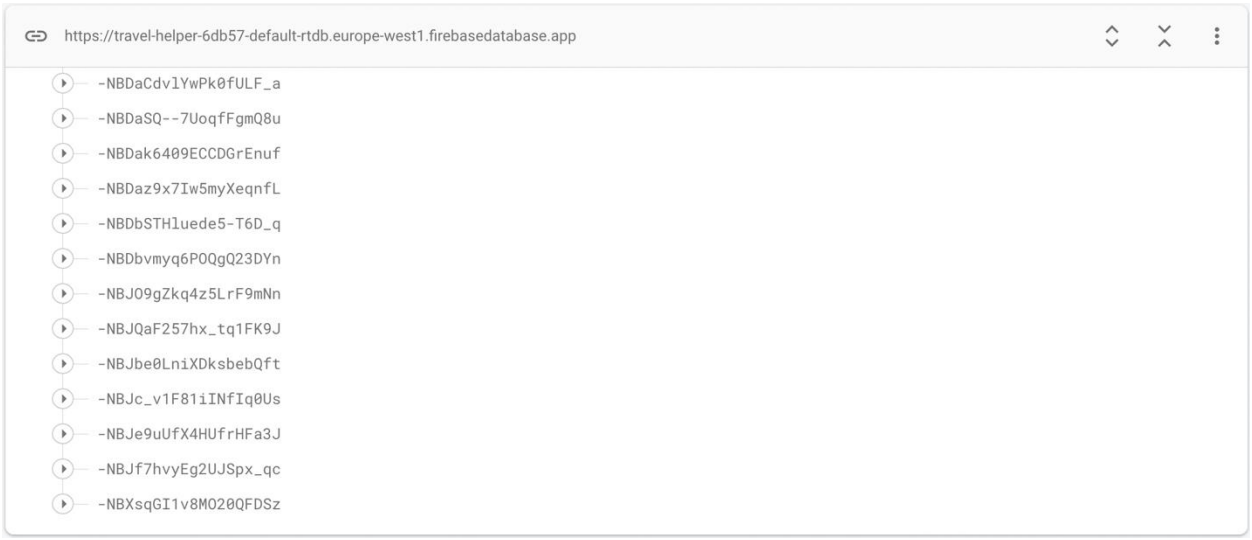
Korisnik iz glavnog zaslona može doći do zaslona profila gdje se nalaze podaci trenutno prijavljenog korisnika, zaslona odakle korisnik može dodati novu znamenitost, zaslona za kreiranje rute te zaslona s kontaktom kreatora aplikacije.

Kod zaslona za dodavanje znamenitosti, korisnik ima padajući izbornik gdje odabire vremensko razdoblje u koje želi spremiti znamenitost. Uz odabir razdoblja korisnik dodaje sliku iz galerije, upisuje naziv znamenitosti, opis znamenitosti te adresu znamenitosti. Važno je napomenuti kako se adresa upisuje u obliku: naziv ulice i broj, poštanski broj, grad, država (npr. "Trg Svetog Trojstva, 31000, Osijek, Hrvatska). Na gumb za učitavanje (engl. *upload*) korisnik sprema novu znamenitost u bazu podataka. Sve navedeno prikazano je na slici 5.4.

Slika 5.3. Prikaz glavnog zaslona

Slika 5.4. Dodavanje nove znamenitosti u bazu podataka

Baza podataka sa spremljenim znamenitostima prikazana je na slici 5.5.



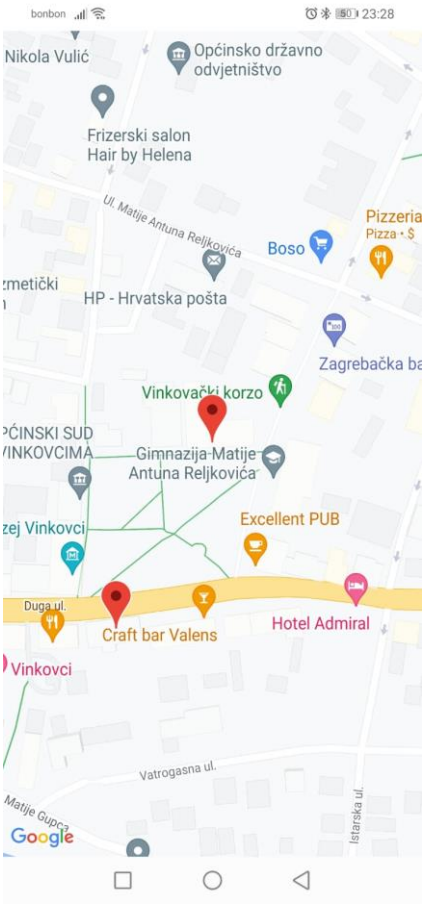
Slika 5.5. Prikaz spremljenih znamenitosti u bazi podataka

Na slici 5.6 prikazan je zaslon za pravljenje rute i prikaza znamenitosti iz baze podataka. Korisnik može birati vremensko razdoblje u kojem se nalaze znamenitosti. Prilikom izbora razdoblja prikazuju se sve znamenitosti sa slikom, nazivom, opisom i adresom koje se nalaze u bazi podataka i odgovaraju izabranom vremenskom razdoblju. Korisnik može klikom na znamenitost odabrati koje ga zanimaju te pritiskom na gumb za prikaz mape, tj. kreiranje rute (engl. *create route*).

Slika 5.6. Prikaz zaslona za prikaz znamenitosti i izgradnju rute

Znamenitosti su prikazane redoslijedom od najbliže korisnikovoj trenutnoj lokaciji, do iduće najbliže prvotnoj znamenitosti i tako do zadnje. Prikaz kreirane rute prikazan je na slici 5.7 sa znamenitostima koje je korisnik odabrao i iz vremenskog razdoblja koje je korisnik odabrao.

Sve funkcije aplikacije rade u potpunosti jednako na svim lokacijama u Republici Hrvatskoj. Kako bi aplikacija radila i za druge zemlje, potrebno je doraditi upis lokacije.



Slika 5.7. Prikaz kreiranje rute

5.2. Uvjeti ispitivanja i korisnički slučajevi

U nastavku je ispitivan rad aplikacije radi provjere ispravnosti i kvalitete aplikacije. Prikazani su rezultati ispitivanja aplikacije u ispitivanju.

5.2.1. Definiranje uvjeta za ispitivanje aplikacije

Uvjet kako bi se aplikacija mogla koristiti jest dopuštanje dohvaćanja lokacije mobilnog uređaja korisnika te kreiranje korisničkog profila/prijavljivanje u postojeći profil. U ispitivanju fokus je stavljen na dodavanje novih znamenitosti i dohvaćanje istih iz baze podataka, te kreiranja rute korisnika i učitavanje lokacije.

Prvotno se ispitivanje naziva Spremanje znamenitosti i ono ispituje sprema li se znamenitost sa svim podacima u bazu podataka nakon pritiska gumba „upload“. Nakon toga provjerava se je li nova znamenitost dodana te prikazana u određenoj kategoriji u kojoj se sprema odlaskom na zaslon za kreiranje rute i odabirom vremenskog razdoblja u koje je spremljena znamenitost te provjeru

spremanja adrese preko ekrana s implementiranom kartom, a ispitivanje se naziva Provjera spremanja nove znamenitosti i njezin prikaz.

5.2.2. Slučajevi korištenja

U testu Spremanja znamenitosti ispitan je rad aplikacije kada se unese znamenitost i njezini podaci. Ukoliko su podaci ispravno uneseni (dodana slika, naziv, opis i pravilno upisana adresa) aplikacija sprema znamenitost u bazu podataka, međutim, ako nije pravilno uneseno aplikacija ne reagira na pritisak gumba „upload“. Navedeno je prikazano slikom 5.8. Greška kod unosa na slici jest nepravilan unos adrese lokacije.

Slika 5.8. Pokušaj spremanja nepravilno unesenih podataka

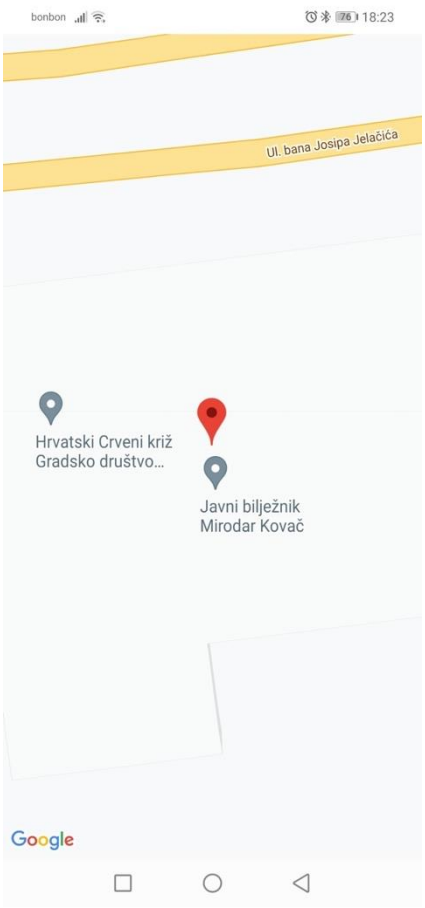
Ukoliko su podaci ispravno uneseni aplikacija ih sprema u bazu podataka što je prikazano na slici 5.9.

Slika 5.9. Uspješno spremanje znamenitosti u bazu podataka

U testu Provjera spremanja nove znamenitosti i njezin prikaz testira se prikazuje li aplikacija novo spremljenu znamenitost u dobu kojem bi trebala te kako je spremljena sa svojim podacima i njezinu lokaciju preko Google Maps API ključa. Na slici 5.10 vidljivo je kako je tek spremljena znamenitost sa slike 5.9 prikazana iz baze podataka. Također je vidljivo kako je ocjena nužna za kooperativno stvaranje preporuka ruta spremljena u bazu podataka i prikazana na zaslon. Ocjena znamenitosti nužna je kako bi svaki korisnik mogao vidjeti preporuku za pojedinu znamenitost od korisnika koji je dodao znamenitost.

Slika 5.10. Prikaz prethodno spremljene znamenitosti

Na slici 5.11 prikazana je znamenitost preko „Google Maps“ zaslona. U ovom slučaju vidljivo je također da aplikacija radi pravilno i ono što bi trebala.



Slika 5.11. Prikaz znamenitosti u zaslonu s kartom

5.3. Analiza rada ostvarene mobilne aplikacije

Aplikacija TravelHelper napravljena je za vrlo jednostavno korištenje, modernog izgleda i intuitivnog načina rada svih funkcija. Samo registriranje i prijava, dodavanje znamenitosti ili kreiranje rute vrlo je jednostavno. Veličina fonta, boje i pozadina prilagođene su tako da sve bude jasno vidljivo. Od korisnika se traži dopuštenje za sve radnje koje mogu narušiti njegovu privatnost te ih on može odbiti, međutim, aplikacija onda neće raditi. Aplikacija radi brzo te nema puno "praznog hoda" (engl. *delay*) između funkcija aplikacije. Tijekom ispitivanja aplikacije sve su funkcije testirane i dale su očekivane rezultate. Na slici 5.9 vidljivo je kako aplikacija uspješno sprema znamenitosti u bazu podataka te je na slikama 5.10 i 5.11 vidljivo kako funkcije prikazivanja svih znamenitosti i kreiranje rute obilaska rade ispravno. Tijekom testiranja aplikacije i korištenja iste nema neočekivanih grešaka ili kvarova aplikacije. Aplikacija nije zahtjevna prema mobilnim uređajima, ne zauzima puno memorije te je zbog toga velika većina korisnika može koristiti bez ikakvog problema velikom brzinom.

6. ZAKLJUČAK

Turizam kao velika i važna ekonomska grana u gotovo svim državama svijeta raste velikom brzinom te s tim turističko posjećivanje gradova i država i broj turista također raste. Gotovo svaki čovjek danas posjeduje pametni uređaj i služi se njime u različite svrhe. Zbog ta dva razloga dolazi do potrebe da i turizam prati tehnologiju. Zbog toga postoji sve više raznovrsnih aplikacija i stranica koja turistu omogućavaju isplanirati obilazak turističkog mjesta koje želi posjetiti. Iako trenutno postoje slična rješenja i aplikacije, rijetko koja korisniku omogućava posjetiti spomenike i znamenitosti iz razdoblja koja njega zanimaju i koja se nalaze u njegovim interesima. Tako korisnik na dlanu ruke ima mogućnost saznati što ga zanima o znamenitosti iz njezinog opisa. S obzirom na to da aplikacija nije ograničena mjestom korištenja, može se jednostavno proširiti i upotrebljavati u bilo kojem mjestu na svijetu.

Mobilna aplikacija TravelHelper je napravljena u programskom jeziku Kotlin, dok je izgled, odnosno dizajn aplikacije napravljen u XML-u, a sve to u programskom okruženju Android Studio. Za spremanje podataka, od korisničkih računa do svih podataka o znamenitosti, korištena je baza podataka FireBase. Korištenje svih lokacijskih usluga ostvareno je putem Goole Maps i API ključa. Zamišljeni funkcionalni i nefunkcionalni zahtjevi aplikacije uspješno su ostvareni, a primjer toga je dohvaćanje lokacije mobilnog uređaja korisnika, spremanje u bazu podataka i čitanje iz iste, kreiranje profila i prijava korisnika i druge. Nakon ispitivanja aplikacije ustanovljena je ispravnost i funkcionalnost aplikacije. Također, aplikacija ima potencijal i mjesta za nadogradnje novim funkcionalnostima.

LITERATURA

[1] turizam. Hrvatska enciklopedija, mrežno izdanje. Leksikografski zavod Miroslav Krleža, 2021. dostupno na: http://www.enciklopedija.hr/Natuknica.aspx?ID=62763 [29.06.2022.]

[2] V. Ponciano, I. M. Pires, F. R. Ribeiro, N. M. Garcia, Mobile application for Inclusive Tourism, Chaves, 16th Iberian Conference on Information Systems and Technologies, 2021, str. 1

[3] A. Turner, How Many Smarthphones Are In The World?: How Many People Have Smarthpones In The World [online], BankMyCell, 2022., dostupno na: https://www.bankmycell.com/blog/how-many-phones-are-in-the-world [29.06.2022.]

[4] V. Ng, Non-Functional Requirement of the Mobile Development system [online], Medium, 10.03.2022., dostupno na: https://medium.com/@vishwasng/non-functional-requirement-of-the-mobile-development-system-e0ed98f2a872 [29.06.2022.]

[5]Location-Aware Application, Techopedia, 02.12.2014., dostupno na: https://www.techopedia.com/definition/30836/location-aware-application [26.07.2022.]

[6] K. Chinetha, J. Daphney Joann i A. Shalini, An Evolution of Android Operating System and Its Version, International Journal of Engineering and Applied Sciences, vol. 2, no. 2, velj. 2015., dostupno na: https://media.neliti.com/media/publications/257997-an-evolution-of-android-operating-system-2d35484a.pdf [29.06.2022.]

[7] Android Versions, https://www.temok.com/blog/android-version-list/ [29.06.2022.]

[8]Android Studio, TechTarget, lis. 2018., dostupno na: https://www.techtarget.com/searchmobilecomputing/definition/Android-Studio [29.06.2022.]

[9] M. Moskala i I. Wojda, Android Development with Kotlin, Say hello to Kotlin, Packt Publishing Ltd, 30. kol 2017., str.8, dostupno na: https://books.google.hr/books?hl=hr&lr=&id=PJZGDwAAQBAJ&oi=fnd&pg=PP1&dq=kotlin&ots=3KhhcFKWxC&sig=nFtXQ9ZZfR-5AbKq3neJ-i-DG5w&redir_esc=y#v=onepage&q=kotlin&f=false [29.06.2022.]

[10] What is XML?, JavaTpoint, dostupno na: https://www.javatpoint.com/what-is-xml [29.06.2022.]

[11] D. Stevenson, What is Firebase? The complete story, abridged, Medium (2018, rujan). Dostupno na: https://medium.com/firebase-developers/what-is-firebase-thecomplete-story-abridged-bcc730c5f2c0 [01.09.2022.]

[12] Get Current Location in Android Studio using Kotlin, Techpass Master, dostupno na: https://techpassmaster.com/get-current-location-in-android-studio-using-kotlin/ [01.09.2022.]

[13] yogeshdotnet, Google images, dostupno na: https://yogeshdotnet.com/wp-content/uploads/firebase-introduction-in-details-1024x600.jpg [01.09.2022.]

ŽIVOTOPIS

Mislav Čačić rođen je 01.01.1999. godine u Osijeku. Prilikom završetka Osnovne škole Bartola Kašića u Vinkovcima 2014. godine upisuje smjer Tehničar za mehatroniku u Tehničkoj školi Ruđera Boškovića u Vinkovcima. 2018. godine završava srednjoškolsko obrazovanje i iste godine upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Govori engleski i njemački. Poznaje brojne programske jezike kao što su C, C++, C#, Kotlin, Python, Swift. U slobodno vrijeme voli se družiti s prijateljima i odlaziti na treninge u teretanu.

_____

Potpis autora

SAŽETAK

U Završnom radu osmišljena je i napravljena Android aplikacija za kooperativno stvaranje i preporučivanje ruta obilaska turističkih znamenitosti koja je ostvarena u Android Studiju. Funkcionalnosti aplikacije ostvarene su programskim jezikom Kotlin, dok je opisni jezik za dizajn i izgled aplikacije korišten XML. Aplikacija imena TravelHelper korisniku daje mogućnost stvaranja profila ili registriranja, dodavanja nove znamenitosti u bazu podataka, izvlačenja podataka radi prikaza istih, ocjenjivanja znamenitosti i kreiranja rute kretanja preko Google Mapsa. Kroz ocjenjene znamenitosti korisnik ima mogućnost stvarati rutu prema ocjenama i preporukama drugih korisnika aplikacije i vlastitim interesima.. Na primjerima je utvrđeno kako je aplikacija u potpunosti ispravna i funkcionalna u svim funkcijama. Cilj je aplikacije omogućiti turistu lakše pronalaženje sadržaja u nekom gradu koji ga zanima i koji bi htio posjetiti te kako bi osobe koje žive u turističkim mjestima i od turizma mogle dodati zanimljivosti svoga mjesta te na taj način privući turiste na posjet. Drugi cilj mobilne aplikacije jest pomoći turistu u odabiru sadržaja koje želi posjetit prilikom svog boravka u nekom mjestu.

Ključne riječi: Android mobilna aplikacija, geolokacija, kooperativno stvaranje i preporučivanje rute, turizam, znamenitosti.

ABSTRACT

In the final paper, an Android application for the cooperative creation and recommendation of tourist sightseeing routes was designed and built, which was realized in the Android Studio. The functionality of the application was realized using the programming language Kotlin while the descriptive language used for the design and appearance of the application was XML. The TravelHelper application gives the user the ability to create a profile or register, add a new landmark to the database, retrieve data for display, rate landmarks and create a route via Google Maps. Through the rated sights, the user has the possibility to create a route according to the ratings and recommendations of other users of the application and their own interests. The example show that the application is completely correct and functional in all functions. The goal of the application is to make it easier for tourists to find content in a city that they are interested in and would like to visit, and so that people who live in tourist places and from tourism can add interesting things to their place and thus attract tourists to visit. Another goal of the mobile application is to help the tourist in choosing the content he wants to visit during his stay in a place.

Keywords: Android mobile application, geolocation, cooperative route creation and recommending, tourism, sights.

PRILOZI

Prilog 1. Završni rad u formatu .docx

Prilog 2. Završni rad u formatu .pdf

Prilog 3. Programski kod mobilne aplikacije