Mobilna Android aplikacija za potporu obilasku turističkih znamenitosti

Lukac, Ivan

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

Sveučilišni preddiplomski studij

MOBILNA ANDROID APLIKACIJA ZA POTPORU OBILASKU TURISTIČKIH ZNAMENITOSTI

Završni rad

Ivan Lukac

Osijek, 2021.

SADRŽAJ

1. UVOD

Razvoj informacijskih i komunikacijskih tehnologija općenito, te razvoj i korištenje mobilnih uređaja uvelike su poboljšali razvoj turizma i njegovo promoviranje. Informacije su dostupne bilo kome na svijetu sa skoro bilo koje lokacije na svijetu. Gotovo na svim najpoželjnijim turističkim lokacijama, uvijek preko mobilnog uređaja imamo pristup internetu, pa turist u svakom trenutku posjeduje sve dostupna znanja vezana za turističke sadržaje u blizini kojih se nalazi. Tehnologija ne služi samo turistima, već i turističkim radnicima i ugostiteljima. Oni uz pomoć tih tehnologija mogu prikupiti vrlo važne podatke o posjećenosti turističkih lokacija i atrakcija, kao i informacije o posjećenosti pojedinih mjesta ili interesu turista. Na temelju toga, mogu planirati i stvarati svoju turističku ponudu.

Glavni zadatak ovog završnog rada je programski ostvariti mobilnu aplikaciju za turistički obilazak Tvrđe u gradu Osijeku. Smisao aplikacije je pružiti turistima pomoć pri obilasku Tvrđe. U jednostavnom korisničkom sučelju se na trenutnim lokacijama trebaju prikazati korisne informacije o znamenitosti u blizini lokacije, a na temelju prethodno kreiranog profila korisnika rangirati te informacije po važnosti i interesima korisnika. Cilj ovog rada je analizirati već postojeća rješenja, istražiti ih, te na temelju njih, stvoriti novo, po mogućnosti prikladnije rješenje, odnosno mobilnu aplikaciju za turistički obilazak Tvrđe. Ta aplikacija treba omogućiti novo turističko iskustvo na temelju korisnički kreiranog profila i korištenih mehanizama za prikaz informacija korisniku.

Drugim poglavljem definiran je obilazak turističkih sadržaja, izazovi obilaska turističkih znamenitosti i potpora računalnih tehnologija u turizmu i obilasku turističkih sadržaja. U drugom poglavlju još su prikazana postojeća slična rješenja mobilnih aplikacija za turizam. U trećem poglavlju navedeni su svi funkcijski i nefunkcijski zahtjevi na mobilnu aplikaciju, opisane su sve funkcionalnosti mobilne aplikacije i arhitektura mobilne aplikacije. Četvrtim poglavljem prikazano je programsko rješenje ove mobilne aplikacije na strani korisnika i na poslužiteljskoj strani. Nakon toga će se u petom poglavlju prikazati način korištenja aplikacije korisniku, te ispitati i analizirati njen rad na nekoliko karakterističnih korisničkih slučajeva.

1.1. Zadatak završnog rada

U teorijskom dijelu završnog rada treba opisati i analizirati zahtjeve i izazove pri obilasku turističkih znamenitosti prema unaprijed utvrđenoj ruti ili prema slobodnom obilasku. Također, na temelju prikaza postojećih rješenja i postupaka, te uzimajući u obzir mogućnosti geolokacije

i profil korisnika, treba predložiti model i arhitekturu mobilne Android aplikacije koja pri kretanju rutom korisniku omogućuje otkrivanje turističkih znamenitosti i davanje dodatnih informacija o znamenitosti. Nadalje, treba opisati postupak rangiranja informacija i stvaranja preporuka prema korisniku, te potrebne programske tehnologije, jezike i razvojne okvire. U praktičnom dijelu rada, treba razviti mobilnu aplikaciju s bazom podataka na temelju predloženog modela i postupka prikaza informacija. Mobilna aplikacija treba omogućiti unos profila korisnika, izbor rute, prikaz informacija i preporuka korisniku ovisno o geolokaciji, kao I informacije o obavljenom obilasku. Mobilnu aplikaciju potrebno je ispitati i analizirati za različite profile korisnika i različite lokacije, odnosno turističke znamenitosti u Tvrđi u Osijeku.

2. POTPORA RAČUNALNIH TEHNOLOGIJA OBILASKU TURISTIČKIH SADRŽAJA

Nalazimo se u razdoblju u kojemu nove tehnologije ulaze u sve vrste posla i zabave, pa tako i u turizam. Velika većina turista sada ima pametni telefon. Zbog toga, razvoj mobilnih aplikacija za turizam je snažno popraćen i korisniku se omogućava dolazak novih informacija u istom trenutku u kojem ih on zatraži. Kako u ostatku svijeta, tako i ovdje u Osijeku, cilj je implementirati što više novih tehnologija u turistički razvitak i turističku ponudu grada. Tako je i razvoj ove mobilne aplikacije pokušaj doprinosa osječkom turizmu. Može rezultirati pozitivnim razvojem turizma i dobrom povratnom informacijom turista koji koriste ovu aplikaciju.

2.1. Obilazak turističkih sadržaja

Prema [1], turizam je gospodarska djelatnost koja obuhvaća skup odnosa i poslova vezanih za putovanje zbog odmora i uživanja. Bazira se ne na stalnom već na privremenom boravištu turista. Do porasta turističkih pokazatelja, došlo je uslijed povećane potrebe za odmorom i promjenom klime te pojačanog buđenja smisla za ljepotom krajolika ili pojedinih društvenih događanja. Zbog niza koji proizlaze iz turističkog razvoja, turizam je postao iznimno važna komponenta gospodarstva države.

Turistički obilazak je unaprijed planirano kretanje turista s ciljem obilaska što više znamenitosti tog područja koje posjećuje, na što efikasniji način. Može biti već viđeni obilazak koji je turist čuo kao preporuku osobe koja je već posjetila to mjesto, ili koju je turist vidio na televiziji, društvenim mrežama, časopisu itd.. Također, turistički obilazak ne mora biti unaprijed planiran, može biti i slobodni obilazak gdje turist obilazi znamenitosti bez određenog plana i određenog vremenskog prozora za obilazak tog sadržaja.

Turistički obilazak ne mora biti samo slušanje turističkog vodiča ili čitanje informacija o nekoj znamenitosti iz raznih priručnika i knjižica, turistički obilazak se također može vršiti pomoću mobilne ili web aplikacije, ali budući da je turistički obilazak najčešće obavljen pješke, onda je praktičnije korištenje mobilne aplikacije.

Prema [2], turističke zajednice su organizacije koje se osnivaju radi promicanja turističkih aktivnosti te turizma kao važne gospodarske grane u državama koje se bave turizmom. Jedan od ciljeva je i zastupanje gospodarskih interesa ljudi koji se bave ugostiteljskim uslugama, uslugama u turizmu ili obavljaju drugu djelatnost neposredno povezanu s turizmom.

2.2. Izazovi obilazaka turističkih sadržaja i potpora računalnih tehnologija

Turistički obilazak uvelike ovisi o turistu. Ako je turist mlada osoba željna dugih šetnji, cjelodnevnih obilaženja turističkih znamenitosti te ima neograničene novčane resurse i vrijeme, takav turistički obilazak neće imati nikakve mane, izazove ili smetnje za turista. No, u većini slučajeva to nije tako, turisti dolaze na turističke destinacije na određeni dio vremena (tjedan dana prosječno) i s određenim novčanim budžetom. Također, turisti najčešće nisu voljni hodati dulje rute za obilazak znamenitosti ako se one mogu učinkovitije obići za manje vremena i uz manje hoda. U tom slučaju, ljudi koji su dobro upoznati s određenom turističkom destinacijom mogu olakšati turistima obilazak stvaranjem preddefiniranih ruta za obilazak turističkih sadržaja. Prema [2], pomoću mobilnih uređaja je moguće ostvariti izvrstan turistički doživljaj preko ugrađenog GPS uređaja, stalne internetske veze za otkrivanje novih informacija o znamenitostima, te kamere pomoću kojih se mogu sačuvati uspomene o turističkom obilasku Budući da je došlo vrijeme u kojem svaki turist posjeduje Smartphone, mogući je razvoj mobilnih aplikacija kojima će se olakšati turistu obilazak turističkog sadržaja gdje će to moći obaviti za manje vremena i bit će odmorniji za ostale aktivnosti za vrijeme svog odmora u tom mjestu. Prema [3], aktivni IOS ili Android uređaj ima više od 700 milijuna ljudi u svijetu. Tako se u posljednjih nekoliko godina razvilo više aplikacija koje olakšavaju turistima obilazak nekog sadržaja, npr. aplikacija koja definira rute za korisnika. Ruta može biti obavljena autom, biciklom ili pješke. Uz to, turističke zajednice gradova i mjesta u velikom broju slučajeva imaju Facebook, Twitter i Instagram stranicu, te Youtube kanal što olakšava novim turistima kratki uvid u ono što bi mogli vidjeti te bi to moglo privući nove turiste.

2.3. Stanje u području i postojeća slična rješenja

U ovom odjeljku su prikazana slična rješenja na platformi Google Play. To znači da su prikazane aplikacije za Android OS, budući da je to ciljano tržište na koje se pokušava plasirati s ovom aplikacijom. Aplikacija koja je prikazana je najviše za određivanje ruta prema korisniku te pronalaženje smještaja za turiste u određenim turističkim destinacijama.

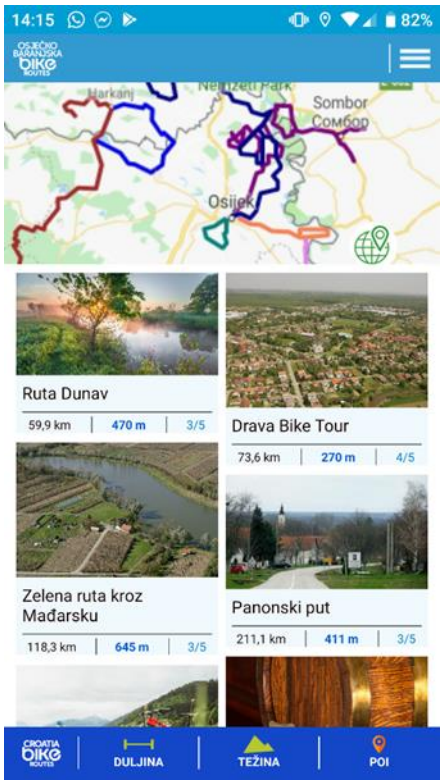
2.3.1. Stanje u području obilazaka turističkih znamenitosti

Što se tiče korištenja mobilnih aplikacija za pomaganje turističkom obilasku, već sada postoji velik broj aplikacija s kojima se turisti mogu služiti kako bi poboljšali svoj turistički doživljaj.

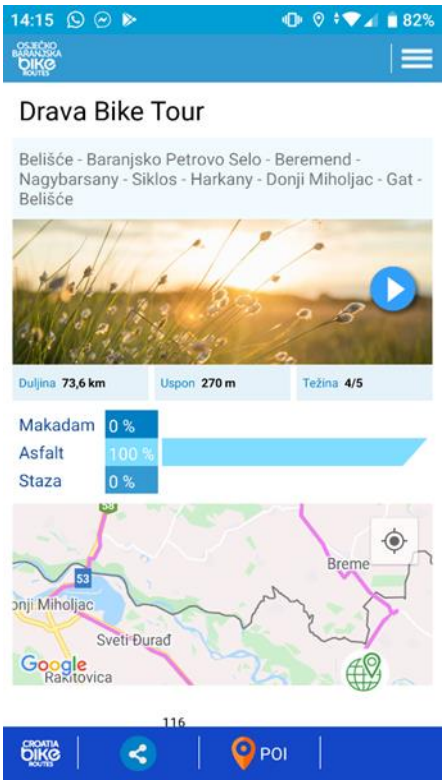
Zasigurno će saznati neke dodatne informacije koje vrlo vjerojatno ne bi saznali da nisu imali aplikaciju na svom uređaju. Kao što je ranije napisano, najviše je istraživano aplikacija na platformi Google Play, no to ne znači da je to jedini izvor aplikacija. Mnogi ljudi koriste iPhone mobilne uređaje koji se ne oslanja na Google Play nego na App Store. Činjenica je da i na App Storeu postoji veliki broj aplikacija koje povećavaju korisnikov turistički doživljaj. Sve aplikacije su bazirane na geolokaciji i bazi podataka koja sadrži informacije o mjestima i znamenitostima. Ovisno o širini korištenja aplikacije je i velika baza podataka gdje su sve informacije o velikom broju mjesta koja se mogu posjetiti. Tvorci ovakvog tipa aplikacija često surađuju s lokalnim hotelima i ljudima koji iznajmljuju apartmane kako bi korisnici aplikacije mogli što lakše naći smještaj. Još jedan koristan segment u ovakvim aplikacijama su predefinirane rute kojima poslužitelj aplikacije daje korisniku savjet što bi sve trebao obići u tom gradu. Također mnoge aplikacije nude opciju da korisnik ostavi svoj komentar ili savjet za buduće posjetitelje nekog odredišta.

2.3.2. Osječko-baranjska Bike Routes

Osjecko-baranjska Bike Routes je mobilna aplikacija koja cilja sportski i avanturistički turizam i turiste. Aplikacija nudi ponudu biciklističkih ruta te omogućuje davanje informacija korisniku uz pojedine znamenitosti uz određenu rutu. U ovoj aplikaciji se mogu saznati i ostali parametri rute kao što je recimo duljina i vrsta podloge po kojoj će se voziti. Pregled aplikacije prikazan je na slikama 2.1 i 2.2.



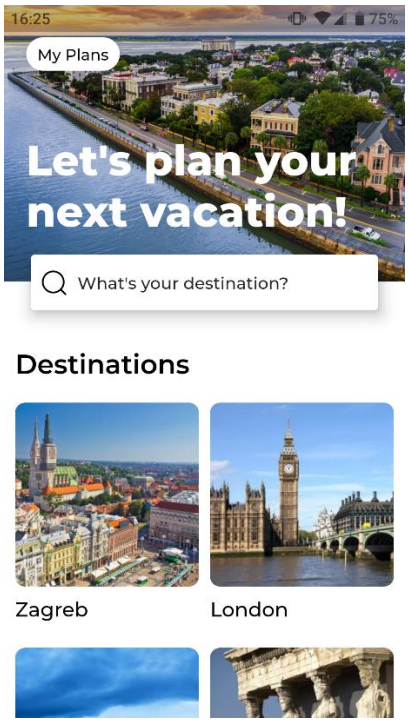
Slika 2.1. Prikaz popisa ruta aplikacije



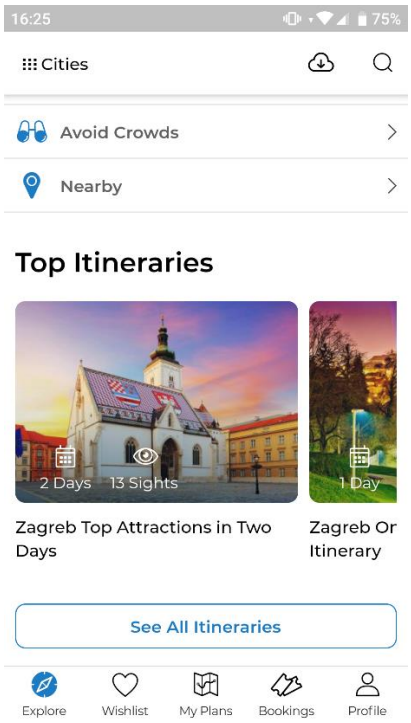
Slika 2.2. Prikaz rute u aplikaciji

2.3.3. Visit A City

Ova mobilna aplikacija koja omogućuje korisniku planiranje svog cjelokupnog odmora. Nudi informacije o lokalitetima i znamenitostima gradova i mjesta. Ima poprilično veliku bazu podataka, pa stoga ima veliki broj informacija koje se mogu saznati. Aplikacija nudi opciju stvaranja to-do liste gdje korisnik bilježi mjesta koja treba posjetiti. U ovoj aplikaciji pokazuje se još mnoštvo dobrih svojstava, npr. izbjegavanje gužve i još mnogi. Rad aplikacije je omogućen i bez internetske veze. Sučelje aplikacije vidljivo je na slikama 2.3 i 2.4.



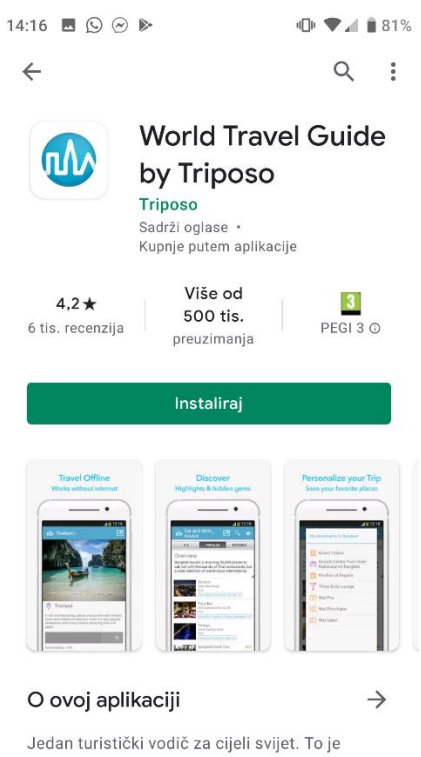
Slika 2.3. Početni zaslon aplikacije



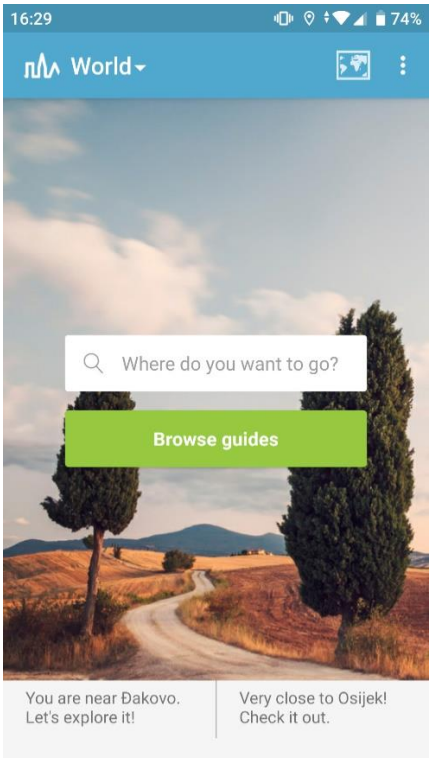
Slika 2.4. Informacije u aplikaciji o gradu Zagrebu

2.3.4. World Travel Guide By Triposo

Aplikacija koju je proizveo Triposo je duže vremena na tržištu i nudi mnoštvo opcija. Razvijena je primarno za ljude koji prvi put dolaze u novi grad ili mjesto, kako bi se lakše snašli. Ova aplikacija za razliku od ostalih nudi i rezerviranje smještaja hotela. Uz to, nudi i 50000 destinacija diljem cijelog svijeta. Rad ove aplikacije je omogućen i bez internetske veze. Prema [3], Triposo je besplatni mobilni vodič koji je dostupan za iOS i Android uređaje. Svaki vodič u aplikaciji sadrži informacije o obilasku znamenitosti, noćnom životu, restoranima i sličnom. Prikaz sučelja aplikacije dan je na slikama 2.5 i 2.6.



Slika 2.5. Prikaz ponude aplikacije u Google Play



Slika 2.6. Početni zaslon aplikacije

Prethodno opisane aplikacije su vrlo korisne za poboljšavanje turističkog doživljaja. Sve tri mobilne aplikacije sadrže niz dodatnih informacija o znamenitostima koje se nalaze na mjestu koje korisnik obilazi. Aplikacije se oslanjaju na gelokaciju uređaja, mogu raditi i bez nje, ali doživljaj aplikacije je potpun ako korisnik odobri korištenje njegove lokacije. Također, jedna od funkcionalnosti aplikacija su preddefinirane rute. To su rute kojima se korisniku izlaže popis mjesta koje bi bilo dobro obići ili koji su prijašnji gosti dobro ocijenili. Jedna od aplikacija ima mogućnost rezerviranja smještaja za goste, ali to nije jedna od glavnih funkcionalnosti ovakvih aplikacija, već samo pokoja aplikacija ima tu mogućnost.

3. MODEL I ARHITEKTURA APLIKACIJE ZA TURISTIČKI OBILAZAK TVRĐE

3.1. Funkcijski i nefunkcijski zahtjevi na mobilnu aplikaciju za turistički obilazak Tvrđe

Radi lakšeg prikaza rada aplikacije u ovome poglavlju će biti objašnjene funkcionalnosti aplikacije te će biti prikazan dijagram rada kako bi rad aplikacije bio zorno prikazan. Pri ulasku u aplikaciju korisnik vidi zaslon s dva gumba. Gumb „Kreiraj Profil“ vodi korisnika na zaslon na kojem će kreirati profil, a gumb „Započni obilazak“ vodi korisnika na zaslon za dohvaćanje lokacije. Za potpuni doživljaj aplikacije, korisnik bi prvo trebao stvoriti korisnički profil. Tako da, ako i pritisne gumb „Započni obilazak“, može pristupiti zaslonu za kreiranje profila pritiskom na gumb u desnom kutu zaslona. Kada korisnik kreira profil, ako želi, može ga spremiti u bazu podataka jednostavnim gumbom, a ako ne želi, ne mora.

Funkcijski zahtjevi na aplikaciju su:

- Pri ulasku u aplikaciju korisnik bira hoće li odmah kreirati profil ili će koristiti aplikaciju bez korisničkog profila
- Kreiranje korisničkog profila
- Spremanje profila u bazu podataka
- Korisnik može obilaziti prethodno definiranu rutu na temelju njegovog profila
- Korisnik može obilaziti slobodnu rutu
- Dohvaćanje korisnikove gelokacije
- Opcija spremanja željene lokacije na listu
- Stvaranje „mrvica“ od svih mjesta koje je posjetio
- Prikazivanje željenih lokacija na Google Maps activityu
- Prikazivanje liste svih spremljenih lokacija koje su posjećene

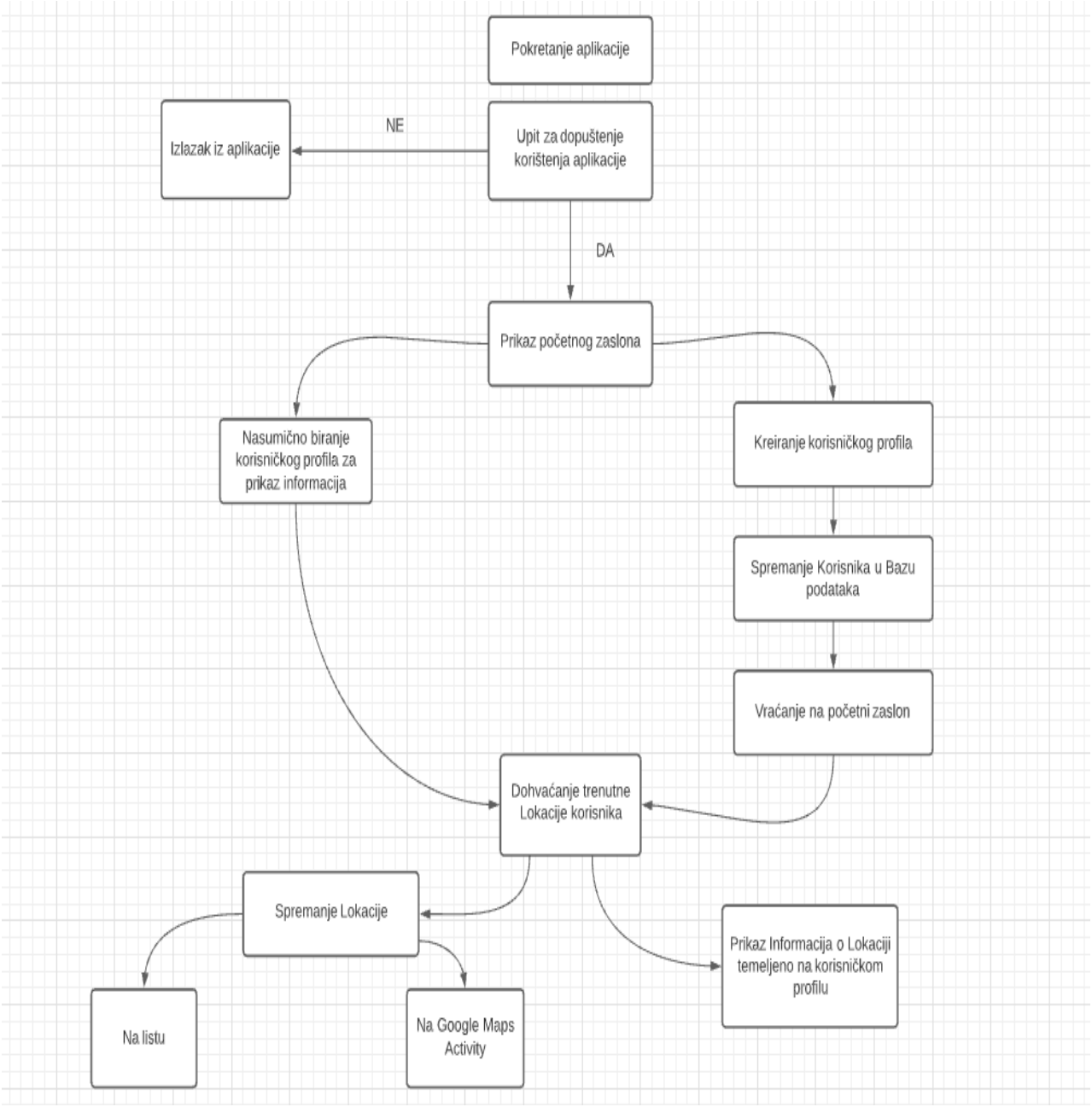
U sljedećem dijelu bit će opisani nefunkcijski zahtjevi na aplikaciju. Prema [4], nefunkcijske karakteristike aplikacije ili sustava su skupno nazvane kvaliteta usluge (eng. *Quality of Service*). Kada tim programera razvija aplikaciju, oni jamče kvalitetu usluge. Definicija kvalitete koju jamči proizvođač usluge naziva se ugovor o razini usluge (eng. *Service Level Agreement*).

Ova aplikacija ima minimalni SDK 23. To znači da je API 23 što odgovara inačici 6.0. Androida. Korisnici koji imaju inačicu Androida ispod 6.0. na svom uređaju neće moći koristiti ovu aplikaciju. Ciljani SDK je 30, što znači da korisnici Android inačice 11.0. i ispod su u

mogućnosti koristiti aplikaciju. Android 11.0. je za vrijeme pisanja ovog rada najnovija inačica Android OS-a, ako ne bude novih inačica ove aplikacije kad budu izlazile sljedeće inačice Android OS-a korisnici koji imaju najnoviji OS neće moći koristiti ovu aplikaciju. Vrijeme koje je potrebno za pokretanje te vrijeme odziva aplikacije je manje od 1 sekunde.

Pri ulasku u aplikaciju korisnik mora odobriti uporabu lokacije korisnikova uređaja. Bez toga, ova aplikacija neće raditi. Nakon što korisnik odobri korištenje lokacije, dohvaća se trenutna geolokacija uređaja za vrijeme kraće od jedne sekunde. U jednom dijelu aplikacije sve dosad spremljene lokacije mogu se pokazati na zaslonu *Google Mapsa*. Kako bi se to postiglo, programer mora implementirati *Google Maps* API ključ. Kako bi došao do tog API ključa, korisnik mora imati Google račun te preko Google Cloud Platform dolazi do njega. Nakon što je programer nabavio ključ, dodaje ga u Manifest.xml ovog projekta i može implementirati svoj kod potreban za rad aplikacije.

Na slici 3.1 vidljiv je dijagram rada aplikacije. Tim dijagramom su opisani postupci koje korisnik može napraviti u ovoj mobilnoj aplikaciji i funkcionalnosti same aplikacije. Iz dijagrama je vidljivo da za potpuni doživljaj aplikacije potrebno je kreirati korisnički profil ili ga barem nasumce odabrati kako bi se informacije o znamenitosti mogle rangirati.



Slika 3.1 Dijagram rada aplikacije

3.2. Potrebni postupci i programski pristupi za ostvarenje aplikacije

U ovom dijelu bit će opisani algoritmi odlučivanja za određene postupke u aplikaciji te još neke pozadinske radnje koje nisu vidljive korisniku.

3.2.1. Postupak dohvaćanja lokacije korisnikovog uređaja

Prema [5], prvi korak dohvaćanja lokacije je traženje dopuštenja od korisnika za uporabu lokacije njegovog uređaja. Kada to korisnik dopusti onda se automatski pali GPS uređaja te se

preko tog GPS-a dolazi do lokacije. Za dohvaćanje aplikacije programer mora implementirati nekoliko predefiniranih objekata pomoću kojih će doći do lokacije. Na početku je potrebno stvoriti objekt klase *FusedLocationProviderClient* koji služi kao početna točka za korištenje gelokacije uređaja. Nadalje, stvara se objekt klase *LocationRequest* i preko njega se prati ima li aplikacija dopuštenje za praćenje lokacije korisnikovog uređaja. Nakon toga implementira se objekt klase *LocationCallback* koji se koristi za primanje obavijesti kada dođe do promjene lokacije ili se više nije u stanju pratiti lokacija. Pomoću ugrađene metode *getLastLocation()* dobije se trenutna lokacija korisnikovog uređaja ako imamo dopuštenje za korištenje GPS-a. Tu lokaciju se nadalje sprema u varijablu *currentLocation* koja se koristi u daljnjem kodu.

3.2.2. Spremanje i prikazivanje željenih lokacija

Jedna od zanimljivijih funkcionalnosti ove lokacije je spremanje lokacije koju je korisnik već posjetio. U prethodnom dijelu opisano je kako se dohvaća lokacija, a sada će biti opisano kako spremiti lokaciju na globalnu listu kako bi korisnik mogao vidjeti koje lokacije je sve obišao i želi ih zapamtiti. Ukoliko korisnik lokaciju ne smatra zanimljivom, istu i ne sprema. Kako bi uopće spremanje lokacija bilo moguće, stvorena je globalna lista lokacija. Lokacija se sprema na način da korisnik pritisne gumb „New Pin“ te se tada lokacija sprema na prethodno stvorenu globalnu listu. Po želji, korisnik može vidjeti listu spremljenih lokacija pritiskom na gumb „Show Pin List“ gdje se korisniku prikazuje novi zaslon s jednostavnom listom adresa tih lokacija. Prikaz lokacija nije praktičan, iz razloga pojavljivanja dugačkog i nerazumljivog niza znakova. Ovim načinom, korisnik može točno vidjeti mjesta koja je posjetio.

Drugi način prikaza spremljenih lokacija je pomoću *Google Maps* zaslona gdje će se za svaku spremljenu lokaciju prikazati „pribadača“. Do *Google Maps* zaslona dođe se tako da se napravi posebna klasa gdje se aktivira *Google Maps API* ključ. Pozivom metoda *getLatitude()* i *getLongitude()* na svaki element spremljene liste lokacija stvara se „pribadača“ na zaslonu.

3.2.3. Kreiranje korisničkog profila i preddefinirane rute

Za uspješno korištenje ove aplikacije potrebno je prvo kreirati korisnički profil prema kojemu će se formirati korisnikov doživljaj u ovoj aplikaciji. Za stvaranje korisničkog profila potrebni su parametri preko kojih programer odlučuje o vrsti profila koji će se stvoriti. Bitno je razlikovati parametre bitni za algoritam stvaranja profila od parametara bitnih za bazu podataka u koju se sprema korisnički profil.

Parametri korišteni za kreiranje profila u ovoj aplikaciji su:

- Spol: muško, žensko
- Godine: 18-35, 36-60, 60+
- Interesi: povijest, društvo, hrana i piće, raznoliko
- Ime i prezime

Na temelju spola, godina i interesa je stvoren algoritam za kreiranje korisničkog profila te stvaranje rute za turistički obilazak. Rutu korisnik može, ali i ne mora obići. Prema [6], planiranje rute je u današnje vrijeme postalo iznimno teško. Stoga je potrebno da turist odredi nekoliko točaka interesa preko kojih će čovjek ili algoritam moći kreirati rutu koja bi odgovarala tom turistu. Uz točke interesa, važno je znati neke informacije o turistu (npr. godine) kako bi se mogla prilagoditi dužina i trajanje rute obilaska. Prema [6], važno je da turist kreira svoje točke interesa pomoću kojih će kreirati korisnički profil ili odlučiti što želi posjetiti.

Važno je znati, algoritam za kreiranje profila i rute neće biti zadovoljavajući za svakog korisnika. Preferencije nekih korisnika nije moguće otkriti s nekoliko parametara, a nema ni smisla tražiti puno više parametara kako se ne bi ulazilo u korisnikovu privatnost. Također, korisnik gubi volju za korištenjem aplikacije ako mora popuniti mnogo parametara za stvaranje svog profila. Algoritam je kreiran tako da ne bude isključivi odabir, npr. osoba koja ima 18-35 godina, zanima je društvo, neće dobiti samo *cafe* barove i noćne klubove za predefiniranu rutu nego će tu biti i muzeja i drugih znamenitosti. Također jedan od važnih parametara je i duljina rute. Ciljano je napravljeno da starijim osobama ruta po Tvrđi bude što kraća za hodanje. Međutim, mlađi ljudi znaju vrlo često brzo izgubiti interes ako je predugačka ruta, stoga je kreiran algoritam da rute budu što kraće a kako bi se ipak obišao veći broj znamenitosti i upoznala Tvrđa iz turističkog oka.

Prema [7], postoje dvije vrste obilazaka, obilazak u jednom ili u više pokušaja. U ovoj aplikaciji su kreirani obilasci iz jednog pokušaja. Algoritam je temeljen na grupama *radio buttona* gdje korisnik klikne na svoj odabir i kada pritisne gumb „Create Profile“ algoritam počinje sa svojim radom gdje za predane godine, spol i interese stvara korisnički profil. Primjerice, za mušku osobu starosti 18-35 godina i interesira se za društvo algoritam će stvoriti profil *PartyMan* i preddefiniranu rutu koja se sastoji od jednog noćnog kluba, *cafe* bara, pivnice, znamenitosti te udruge. Korisnički profili koji se mogu stvoriti su: *PartyMan*, *PartyWoman*, *Historian, Gourmand, All rounder*.

3.2.4. Postupak rangiranja podataka o turističkoj znamenitosti

Prema [8], turistički obilasci postaju sve više i više personalizirani, za razliku od prošlih vremena kada su grupna putovanja bila puno popularnija. Turist današnjice želi imati kreiranu rutu za obilazak prema svojim interesima. Također, želi i saznati činjenice o nekoj znamenitosti bazirane na njegovim interesima.

U ovoj aplikaciji, po literaturnim podacima opisano je 10 znamenitosti sa zanimljivim informacijama. Naravno, sve informacije neće svima biti zanimljive, stoga su korišteni profili koje korisnik kreira ulaskom u aplikaciju ili početkom korištenja aplikacije. Prema tim profilima, može biti 4 tipa rangiranja informacija o znamenitosti. Naravno, neke znamenitosti imaju takovu povijest i takove informacije da jednostavno ne može postojati rangiranje informacija ako su one u nekom širem smislu jednako zanimljive svima ili su sve informacije iste vrste. Primjerice, informacije o pojedinom sakralnom objektu u Tvrđi će biti jednako zanimljive *Partymanu* i *Gourmandu*. Stoga, ne postoji rangiranje informacija već je tekst isti za sve profile.

Za korisnički profil *Partyman,* prvo su prikazane informacije o npr. ljudima koji posjećuju tu znamenitost, je li ta znamenitost mjesto gdje se ljudi sastaju u većem broju te niz i drugih informacija koje su vezane uz osobe ili društvo.

Za korisnički profil *Historian*, prvo su prikazane povijesne informacije, informacije o nastanku tog mjesta, te niz godina koje su značajne u povijesti ili su odgovorne za rast/pad te znamenitosti.

Za korisnički profil *Gourmand* se vrlo često ispostavilo da su informacije vrlo slične sa profilom *All Rounder* zato što veliki broj znamenitosti nije u svezi s kulinarstvom i gastronomijom, ali kod onih znamenitosti koji imaju, te informacije su na prvom mjestu.

Kod korisničkog profila *All Rounder* najvažnije je nepostojanje više informacija iste vrste i postojanje svih vrsta informacija u istom obimu. Također, bitno je da informacije iste vrste nisu poredane jedne za drugom kako korisnik ne bi osjetio dosadu predugim čitanjem nečega što nije svojstveno njegovoj volji.

3.2.5. Model baze podataka

U ovoj aplikaciji koristi se baza podataka *Room* čija implementacija će biti opisana u daljnjem tekstu ovog rada. U ovom dijelu se opisuje kako i zašto se koristi. Baza podataka se koristi kako bi se mogao spremiti već kreiran korisnički profil, da korisnik ne mora pri svakom ulasku u

aplikaciju ponovno kreirati isti korisnički profil. Kao što je prethodno opisano, korisnički profil se sastoji od imena i prezimena, spola, starosti i interesa. Nakon kreiranja profila, korisnik dobije ime profila i preddefiniranu preporučenu korisničku rutu. Stvoreni korisnički profil i rutu moguće je spremiti u bazu podataka. Stoga, za ovu bazu je potrebna jedna tablica koja je nazvana *user*. Tablica se sastoji od imena, prezimena, spola, godina, interesa, kategorije profila te preddefinirane rute. Do podataka za tablicu *user* u bazi podataka se dolazi pomoću elemenata *editText, radio button* i *textview*. Pritiskom na gumb „Save User“, trenutni korisnik se sprema u bazu podataka na način da se povlače podaci iz prethodno navedena 3 elementa. Pritiskom na gumb „Show users“, prikazuje se jednostavna lista svih prethodno spremljenih korisničkih profila. U poglavlju 5 potpoglavlju 5.2 ovog rada prikazano je kako to izgleda u aplikaciji.

3.3. Arhitektura mobilne aplikacije za turističku šetnju Tvrđom

U ovom poglavlju se opisuje raspodjela aplikacije na dio koji je vidljiv korisniku (eng. *frontend*) i na dio koji se izvršava u pozadini aplikacije (eng. *backend*).

Korisnički dio ove aplikacije čine:

- *Google Maps* zaslon - prozor na kojemu se prikazuju „pribadače“ prethodnih lokacija na zaslonu
- Zaslon za dohvaćanje lokacije - zaslon na kojemu se vidi geografska širina i dužina trenutne lokacije te adresa i nekoliko gumbova
- Zaslon za kreiranje profila - zaslon na kojemu se nalaze elementi *editText, textview* i *radio button* za kreiranje profila
- Početni zaslon aplikacije - zaslon s pozadinskom slikom Tvrđe te gumbima koji vode na dohvaćanje lokacije i kreiranje profila
- Zaslon sa spremljenim lokacijama - zaslon na kojemu su prikazane sve dosad spremljene adrese lokacija u obliku jednostavne liste

Poslužiteljski dio ove aplikacije čine:

- Room database – baza podataka u koju se sprema korisnički profil koji korisnik želi spremiti
- Dohvaćanje lokacije- postupak dohvaćanja trenutne korisnikova geolokacije, uz njegovo dopuštenje
- Algoritam odlučivanja rangiranja informacija o znamenitosti – algoritam pomoću kojeg se odlučuje koja će se informacija prva prikazati za određeni korisnički profil

- Algoritam odlučivanja kreiranja profila - algoritam koji na temelju odabranih elemenata kreira korisnički profil

4. PROGRAMSKO RJEŠENJE MOBILNE APLIKACIJE ZA TURISTIČKI OBILAZAK TVRĐE

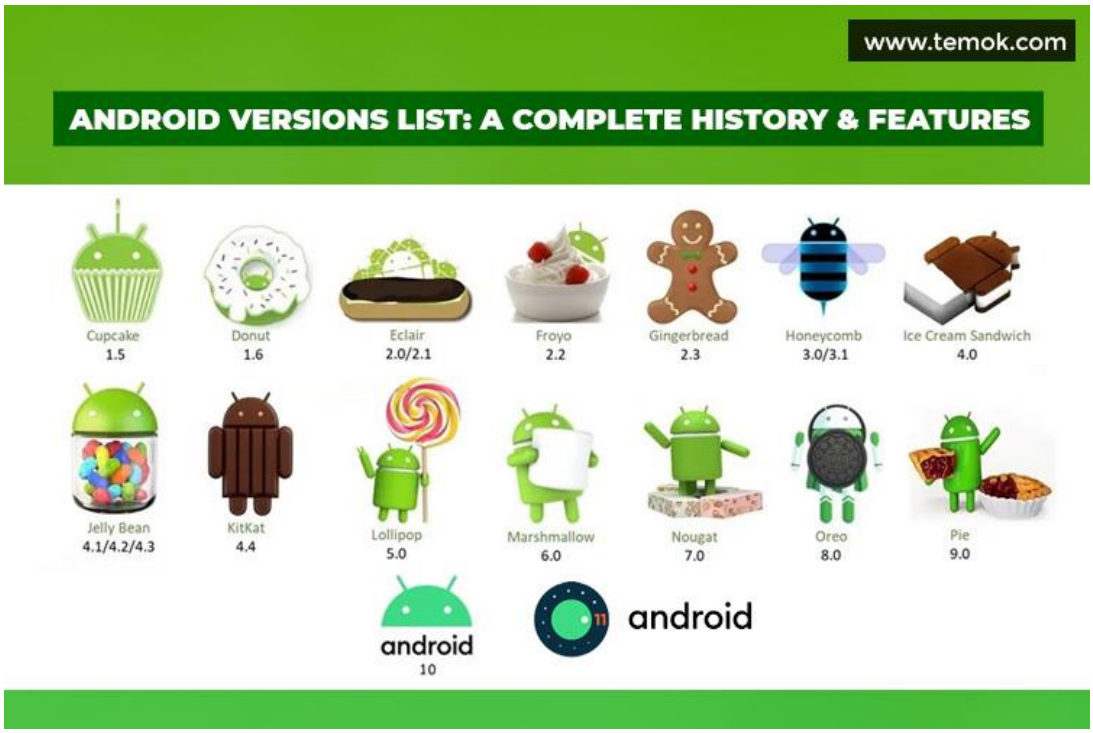
4.1. Korištene programske tehnologije, jezici i razvojne okoline

U ovome dijelu rada teorijski su opisani alati korišteni za razvoj aplikacije za turistički obilazak Tvrđe.

4.1.1. Operacijski sustav Android

Prema [9], Android OS je otvoreni operacijski sustav za mobitele, tablete, televizore i automobile. Trenutno je najpopularniji operacijski sustav za mobitele na svijetu. Prema [10], Android OS je nastao 2003. godine kada su Andy Rubin, Rich Miner, Nick Sears i Chris White osnovali tvrtku Android Inc. kako bi razvijali aplikacije za pametne mobitele. Primarno su htjeli stvoriti operacijski sustav za kamere ali su shvatili da je to tržište premalo, pa su stvorili operacijski sustav za mobilne uređaje. Android Inc. je u poslovnoj suradnji s Googleom, ali ne i u njegovu vlasništvu. Služi kao prevoditelj između korisnika i uređaja, npr. kada korisnik želi fotografirati nešto, sustav mu pruža gumb koji će korisnik stisnuti i onda će sustav narediti uređaju kako da to napravi. Budući da je Android OS otvoreni operacijski sustav, svaki proizvođač mobitela može prilagođavati operacijski sustav svojim korisnicima, ali je najcjenjeniji „stock android“ koji je izvorni Android kako ga je i Google napravio. Zadnja inačica Android OS je Android 11. Sve inačice Android OS-a nazivane su imenima slatkiša (Lollypop, KitKat, Marshmallow,…).

Na slici 4.1 su prikazane sve do sada postojeće inačice Androida.

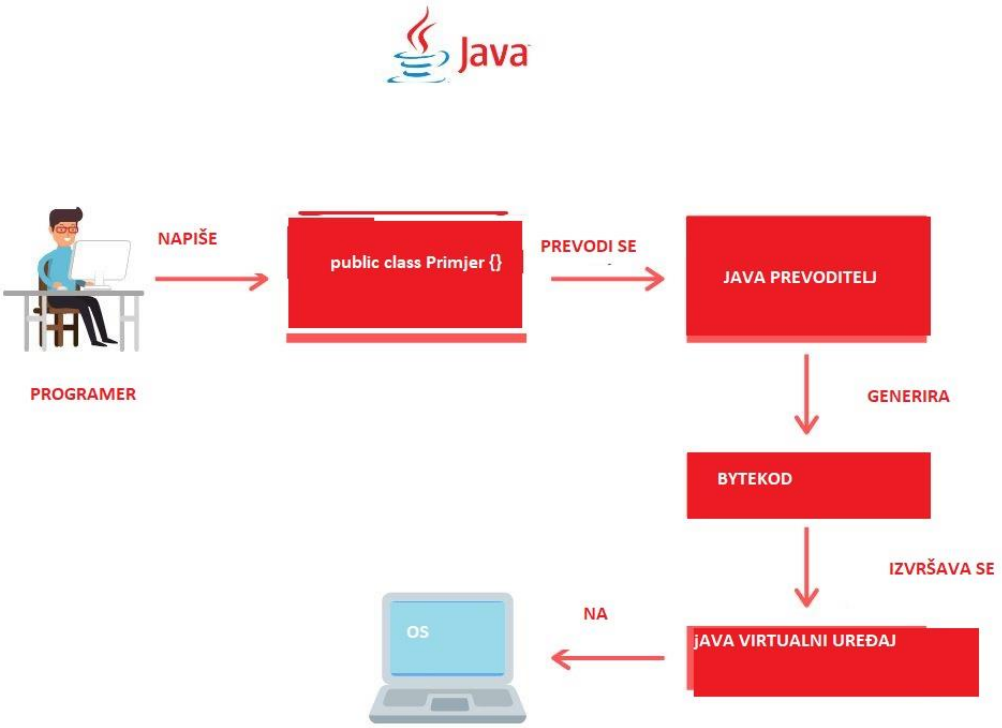


Slika 4.1 Sve dosadašnje inačice Androida [11]

4.1.2. Programski jezik Java

Prema [12], Java je objektno orijentirani programski jezik koji je razvila tvrtka Sun Microsystems. Razvili su ga James Gosling, Mike Sheridan i Patrick Naughton u lipnju 1991. godine. Java je široko upotrijebljeni programski jezik zato što se temelji na konceptu „jednom napiši, pokreni bilo gdje“. Kod napisan u programskom jeziku Java može se pokrenuti na bilo kojem uređaju koji ima *Java Virtual Machine* (JVM). S obzirom da se može pokrenuti na JVM, za razliku od ostalih programskih jezika, Java se ne mora prilagođavati ostalim platformama nego ostaje u svom izvornom obliku. Trenutno, programski jezik Java jedan je od najkorištenijih programskih jezika zato što ga koristi devet milijuna ljudi. Također, programski jezik Java trenutno je jako popularan zato što se Android aplikacije pišu u Javi ili Kotlinu. Aplikacija za turistički obilazak Tvrđe je napisana u Javi.

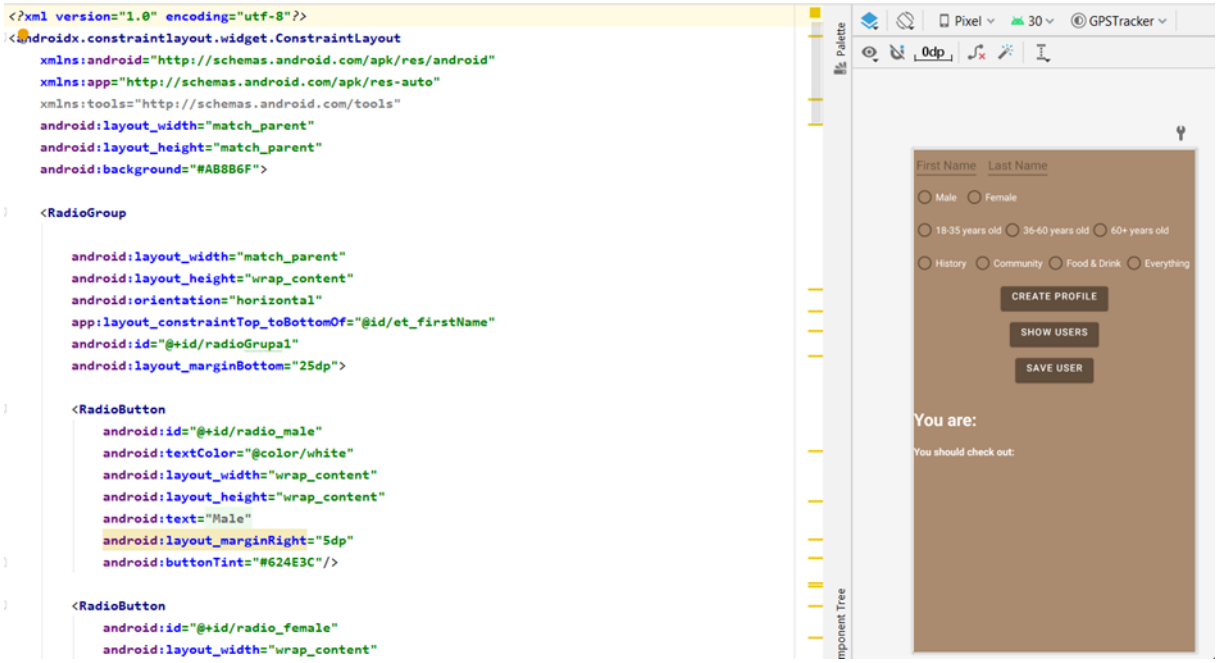
Na slici 4.2 dan je prikaz izvršavanja programskog koda u Javi.



Slika 4.2 Prikaz izvršenja Java koda [13]

4.1.3. XML

Prema [14], XML znači *Extensible Markup Language*. To je markup jezik koji se koristi za označavanje podataka te koristi oznake pomoću kojih se organiziraju stranice ili aplikacije. XML se koristi za dizajn aplikacije ili zaslona koji se želi prikazati. Vrlo je sličan HTML-u , ali se HTML više koristi za dizajn web stranica dok se XML koristi za dizajn aplikacija u razvojnoj okolini Android Studio. Jedna od prednosti XML-a je dohvaćanje podataka iz resursa i prikazivanje na zaslonu. Prikaz se može podijeliti na prikaz koda ili zaslona ili oboje. XML se može pisati u više različitih layouta, npr. Constraint, Linear, Relative. U te layoute se postavljaju elementi kao što su *editText*, *textview*, *list*, *button*, itd. Svim elementima se pridjeljuje pozicija te *id*, a može se promijeniti tekst, boja i ostali atributi. Na sljedećoj slici se vidi dio xml koda u načinu rada *split*.



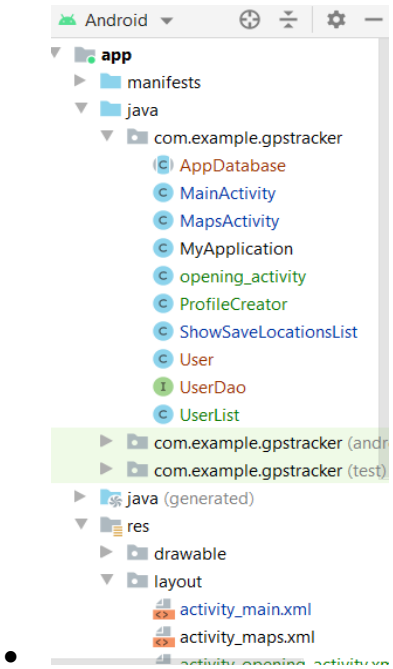
Slika 4.3 Prikaz xml koda i dizajna u načinu rada *split*

4.1.4. Razvojna okolina Android Studio

Prema [15], Android Studio je razvojna okolina koja se koristi za razvoj Android aplikacija. Mogu se razvijati aplikacije za mobitel, TV - uređaj, automobil te tablet. Temelji se na integriranom razvojnom okruženju (IDE). Android Studio dostupan je za sve popularne operacijske sustave Windows, Linux, macOS. Moguće je pisati pomoću dva programska jezika, a to su Java i Kotlin. Prije je Java bila zastupljenija, ali trenutno je Kotlin „popularniji“ među programerima Android aplikacija. Neke od funkcionalnosti Android Studija su: gradle sustav, refaktoriranje koda, integracija sa Githubom i Gitlabom. Virtualni uređaj (emulator) koristi se za testiranje aplikacija kako ih se ne bi moralo instalirati na fizički uređaj te tamo testirati.

Struktura projekta u Android Studiju je:

- Gradle - sustav koji se koristi za izgradnju projekta
- Manifest - mjesto gdje se nalazi AndroidManifest.xml, koristi se za traženje raznih dopuštenja u aplikaciji
- Java - sadrži sve dijelove programa koji sadrže Java programski kod
- Res - mjesto gdje se nalaze svi resursi aplikacije, a to su xml datoteke, slike, stringovi, itd.



Slika 4.4. Prikaz strukture projekta

4.2. Programsko rješenje na strani korisnika

U ovome dijelu rada sve prethodno opisane funkcionalnosti su potkrijepljene programskim kodom aplikacije. Prikazana su programska rješenja i vidljiva na korisničkoj strani i na koje korisnik može utjecati.

4.2.1. Google Maps activity

Pritiskom na gumb „SHOW MAP“ na zaslonu za dohvaćanje lokacije, korisnik dolazi do ovog zaslona. Na ovom zaslonu su prikazane „pribadače“ svih lokacija koje je korisnik odlučio spremiti na globalnu listu. Lokacije koje je korisnik spremio se spremaju na globalnu listu te se onda iz te globalne liste prikazuju na *Google Maps* zaslonu kao što je prikazano na slici 4.5. Za korištenje ovog zaslona potreban je *Google Maps* API ključ koji može koristiti svaka osoba koja ima kreiran Google Račun. Do API ključa dolazi se u *Google Cloud Platform* konzoli. Nakon što se implementira API ključ, programer implementira svoj željeni kod te je zaslon spreman na korištenje. Na slici 4.5 bit će prikazano korištenje *Google Maps* zaslona. Ako niti jedna lokacija nije spremljena, na tom zaslonu bit će prikazana početna lokacija grada Sydneya.

```
public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;
    List<Location> savedLocations;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync( onMapReadyCallback: this);
        MyApplication myApplication=(MyApplication)getApplicationContext();
        savedLocations=myApplication.getMyLocations();
    }
```

Slika 4.5 Dio programskog koda klase MapsActivity

Na slici 4.6 prikazano je stvaranje novih „pribadača“ na *Google Maps* zaslonu unutar for petlje gdje se predaju lokacije s globalne liste i stvaraju se novi objekti klase *MarkerOptions* tako da im se pomoću metoda *getLatitude()* i *getLongitude()* predaju parametri geografske širine i dužine. Postupak kreiranja Google Maps zaslona preuzet je s [16], a pretvaranje lokacija s globalne liste u pribadače s [17].

```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;
    LatLng sydney = new LatLng( v: -34,  v1: 151);
    LatLng lastLocationPlaced=sydney;
    for(Location location : savedLocations){
        LatLng latLng=new LatLng(location.getLatitude(),location.getLongitude());
        MarkerOptions markerOptions =new MarkerOptions();
        markerOptions.position(latLng);
        //markerOptions.title("Lat: "+location.getLatitude()+" Lon: "+location.getLongitude());
        mMap.addMarker(markerOptions);
        lastLocationPlaced=latLng;
    }
    mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(lastLocationPlaced, v: 12));


    // Add a marker in Sydney and move the camera
    //LatLng sydney = new LatLng(-34, 151);
    //mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));
    //mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
}
```

Slika 4.6 Programski kod za stvaranje pribadače na Google Maps zaslonu

4.2.2. Početni zaslon aplikacije

Pri ulasku u aplikaciju korisnik može vidjeti pozadinsku fotografiju Tvrđe te dva gumba koji vode na sljedeće zaslone. Prelazak na drugi zaslon se radi pomoću objekta klase Intent što je prikazano programskim kodom na slici 4.7.

```
public class opening_activity extends AppCompatActivity {
    Button btn_zapocniObilazak, btn_kreirajProfil;
    TextView tv_naslov;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_opening_activity);
        btn_zapocniObilazak=findViewById(R.id.btn_zapocniObilazak);
        btn_kreirajProfil=findViewById(R.id.btn_kreirajProfil);
        tv_naslov=findViewById(R.id.textView);
        btn_zapocniObilazak.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i=new Intent( packageContext: opening_activity.this,MainActivity.class);
                startActivity(i);
            }
        });
        btn_kreirajProfil.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i=new Intent( packageContext: opening_activity.this,ProfileCreator.class);
                startActivity(i);
            }
        });
    }
}
```

Slika 4.7 Programski kod klase opening_activity za početni zaslon

4.2.3. Zaslon s dohvaćanjem lokacije

Na zaslonu s dohvaćanjem lokacije korisnik može vidjeti geografsku širinu i dužinu, te adresu mjesta na kojem se trenutno nalazi. U gornjem desnom kutu zaslona korisnik ima „plus“ koji je zapravo gumb pomoću kojeg može otići na zaslon za kreiranje novog korisničkog profila. Uz to, korisnik može vidjeti jedan *switch* pomoću kojeg može odrediti želi li da se geolokacija njegovog uređaja i dalje prati. Ispod tog elementa, nalazi se nekoliko *radio buttona*. Oni služe za podešavanje potrošnje baterije pri određivanju lokacije i za preciznost određivanja lokacije korisnika. Uz te elemente, postoji još tri gumba koji služe za dodatne funkcionalnosti aplikacije kao što su: spremanje lokacije, prikazivanje spremljenih lokacija na jednostavnoj listi, te prikazivanje spremljenih lokacija na *Google Maps* zaslonu. Nakon toga, nalazi se jedna skupina

radio buttona za odabir prethodno definiranog korisničkog profila te gumb pomoću kojeg se prikazuju dodatne informacije o trenutnoj lokaciji korisnika. Na slici 4.8 je prikazan dio XML koda kojim se postiže dizajn ovog zaslona.

```
<Button
    android:id="@+id/btn_showMap"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:insetTop="1dp"
    android:insetBottom="1dp"
    android:text="Show Map"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/btn_showPinList"
    android:backgroundTint="#624E3C"
    app:layout_constraintBottom_toTopOf="@id/tv_labelProfile"/>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintTop_toBottomOf="@id/btn_showMap"
    app:layout_constraintBottom_toTopOf="@id/radio_profiles"
    android:text="Please select your profile:"
    app:layout_constraintStart_toStartOf="parent"

    android:layout_marginBottom="10dp"
    android:textSize="15sp"
    android:textColor="@color/white"
    android:id="@+id/tv_labelProfile"
    />
```

Slika 4.8 Kreiranje gumba „SHOW MAP“ i textview „Please select your profile“

4.2.4. Zaslon za kreiranje profila

Korisnik može doći do ovog zaslona na dva načina, može odmah na početnom zaslonu pritisnuti gumb za kreiranje novog profila ili može na zaslonu za dohvaćanje lokacije stisnuti „plus“ u gornjem desnom kutu te doći do ovog zaslona. Na samom zaslonu je prikazano više skupina radio buttona pomoću kojih korisnik odabire karakteristike svog profila na temelju koji se taj profil formira. Uz skupine grafičkih oznaka radio buttona se nalaze i dva edittexta gdje korisnik upisuje svoje ime i prezime kako bi se kada korisnik spremi profil u bazu podataka znalo čiji je profil. Osim toga, na zaslonu se nalazi tri gumba na temelju kojih se kreira profil, sprema u bazu podataka, te se profili u toj bazi podataka prikazuju u obliku jednostavne liste. Na dnu zaslona nalaze se dva *textviewa* koji prikazuju korisnikov profil i rutu koja je preporučena za korisnika. Na slici 4.9 će biti prikazan dio XML koda za kreiranje takvog zaslona.

```
<RadioGroup
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    app:layout_constraintTop_toBottomOf="@id/radioGrupa1"
    android:id="@+id/radioGrupa2"
    android:layout_marginBottom="25dp"
   >

    <RadioButton
        android:textColor="@color/white"
        android:id="@+id/radio_age1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:buttonTint="#624E3C"
        android:text="18-35 years old" />

    <RadioButton
        android:textColor="@color/white"
        android:id="@+id/radio_age2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:buttonTint="#624E3C"
        android:text="36-60 years old" />
    <RadioButton
        android:textColor="@color/white"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/radio_age3"
        android:text="60+ years old"
        android:buttonTint="#624E3C"/>
```

Slika 4.9 Radio grupa za godine korisnika

4.2.5. Zaslon sa spremljenim lokacijama

Pritiskom na gumb „SHOW SAVED LOCATIONS“ dolazi se do ovog zaslona. Na zaslonu se prikazuje jednostavna lista svih adresa lokacija koje je korisnik spremio. Budući da, ako se lokacija pokuša prikazati kao string, dobije se niz korisniku besmislenih znakova, u ovoj aplikaciji je prikazana lista adresa koje je korisnik spremio. Prikazivanje adresa je puno praktičnije, jer svi ljudi u svakodnevnom životu koriste adrese mjesta. Sve lokacije su se spremale u globalnu listu iz koje su onda dohvaćane i u *for* petlji pretvarane u adrese što se može vidjeti na sljedećoj slici. Lokacije su pretvorene u adrese pomoću predefinirane klase *Geocoder* i ugrađene metode *getFromLocation* gdje se uz predane geografsku širinu i dužinu, te maksimalan broj rezultata može dobiti adresa te lokacije.

```
public class ShowSaveLocationsList extends AppCompatActivity {
    ListView lv_savedLocations;
    String TAG="ShowSaveLocationsList";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_show_save_locations_list);
        MyApplication myApplication=(MyApplication)getApplicationContext();
        Geocoder geocoder=new Geocoder( context: this);
        List<Location> savedLocations=myApplication.getMyLocations();
        lv_savedLocations = findViewById(R.id.lv_wayPoints);
        List<Address> addresses=new ArrayList<~>();
        List<String> trueAddresses=new ArrayList<~>();
        for (int i=0;i<savedLocations.size();i++) {
            try {
                trueAddresses.add(geocoder.getFromLocation(savedLocations.get(i).getLatitude(), savedLocations.get(i).getLongitude(),  maxResults: 1)
            } catch (Exception e) {
                Log.e( tag: "Exception",  msg: "Unable to get address");
            }
        }
        lv_savedLocations.setAdapter(new ArrayAdapter<String>( context: this, android.R.layout.simple_list_item_1,trueAddresses));
        Log.d(TAG,  msg: "onCreate: ");

    }
```

Slika 4.10 Programski kod klase za spremanje lokacija na listu

4.3. Programsko rješenje na strani poslužitelja

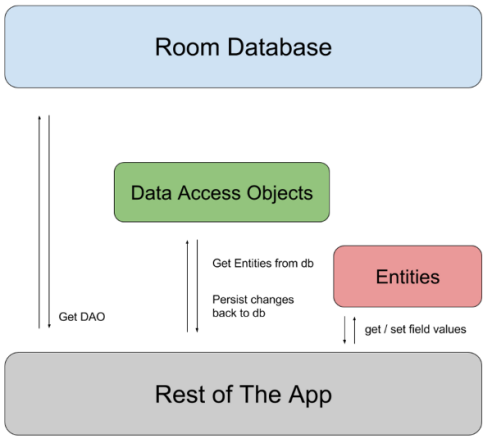
4.3.1. Baza podataka Room

Prema [18], Room je lokalna baza podataka koja služi za spremanje strukturiranih podataka. Baza podataka Room idealna je za ovu aplikaciju zato što ne zahtjeva pristup internetu što je pogotovo bitno za strane turiste od kojih neki nemaju pristup internetu. Baza podataka Room se koristi češće nego *SQLite* baza podataka zato što je jednostavnija za korištenje, a ima sve funkcionalnosti kao i *SQLite*.

Baza podataka Room sastoji se od tri temeljne komponente:

- *Database class* - drži bazu podataka i služi kao pristupna točka za poveznicu aplikacije i baze podataka
- *Data entities* - prikazuje tablicu u bazi podataka
- *Data access objects* - klasa koja pruža metode kojima korisnik može manipulirati tablicama u bazi podataka

Na slici 4.11 prikazan je način rada baze podataka Room.



Slika 4.11 Način rada baze podataka Room [18]

Za korištenje baze podataka Room prvo je potrebno implementirati ovisnosti u skriptu *gradle*.

```
dependencies {
    def room_version = "2.3.0"

    implementation "androidx.room:room-runtime:$room_version"
    annotationProcessor "androidx.room:room-compiler:$room_version"
```

Slika 4.12 Ovisnosti za korištenje Room baze podataka

Nakon toga, potrebno je stvoriti klasu koja će predstavljati tablicu u bazi podataka. U ovoj aplikaciji je potrebna samo jedna tablica, ali u slučaju da je potrebno više tablica toliko će biti i klasa. Tablica koja je stvorena naziva se *User* i njeni elementi su ime, prezime, spol, godine, interes, korisnički profil i preddefinirana ruta. Programski kod za stvaranje tablice *User* prikazan je na slici 4.13.

```java
public class User {
    @PrimaryKey(
            autoGenerate = true
    )
    public int uid;

    @ColumnInfo(name = "first_name")
    public String firstName;

    @ColumnInfo(name = "last_name")
    public String lastName;
    @ColumnInfo(name="Years")
    public String years;
    @ColumnInfo(name="Sex")
    public String sex;
    @ColumnInfo(name="Interest")
    public String interest;
    @ColumnInfo(name="Profile")
    public String profile;
    @ColumnInfo(name="Route")
    public String route;

    public User(String firstName, String lastName, String years, String sex, String interest, String profile, String route) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.years = years;
        this.sex = sex;
        this.interest = interest;
        this.profile = profile;
        this.route = route;
    }
```

Slika 4.13 Programski kod klase User

Nakon kreiranja tablice, potrebno je napraviti DAO i definirati metode *insertAll()* i *delete()* za manipuliranje tablicom kao što je prikazano na slici 4.14.

```java
@Dao
public interface UserDao {
    @Query("SELECT * FROM user")
    List<User> getAll();


    @Insert
    void insertAll(User user);

    @Delete
    void delete(User user);
}
```

Slika 4.14 Sučelje UserDao

Nakon toga, potrebno je kreirati klasu za bazu podataka. Klasa *AppDatabase* mora biti apstraktna, te mora imati metodu koja nema parametara i vraća instancu klase DAO, kao što je vidljivo na slici 4.15.

```
@Database(entities = {User.class}, version = 1)
public abstract class AppDatabase extends RoomDatabase {
    public abstract UserDao userDao();
    private static final String DB_NAME = "appDatabase.db";
    private static volatile AppDatabase instance;

    static synchronized AppDatabase getInstance(Context context) {
        if (instance == null) {
            instance = create(context);
        }
        return instance;
    }

    protected AppDatabase() {};

    private static AppDatabase create(final Context context) {
        return Room.databaseBuilder(
                context,
                AppDatabase.class,
                DB_NAME).allowMainThreadQueries().build();
    }
```

Slika 4.15 Programski kod apstraktne klase AppDatabase

Nakon što su svi dosadašnji postupci napravljeni, potrebno je kreirati instancu baze podataka što je i učinjeno na slici 4.16. Model kreiranja Room baze podataka u ovoj mobilnoj aplikaciji temeljen je na [18].

```
AppDatabase db;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.profile_creator);
    db=AppDatabase.getInstance(this);
```

Slika 4.16 Instanciranje baze podataka

4.3.2. Dohvaćanje lokacije

Prema [15], *Location* je predefinirana klasa koja predstavlja geografsku lokaciju nekog mjesta. Može imati geografsku širinu, dužinu, visinu, brzinu i točnost određivanja. Ima veliki broj ugrađenih metoda, ali neke koje će bit korištene u ovoj aplikaciji su *getLatitude()*, *getLongitude()*, itd. Kako bi se došlo do lokacije nekog uređaja prvo se mora zatražiti korisnikovo dopuštenje za pristup geolokaciji njegovog uređaja. Postoje dva pristupa lokaciji:

izravni i pozadinski. U ovoj aplikaciji je korišten izravni pristup. Na početku je potrebno u manifestu zatražiti pristup lokaciji kao što je prikazano na slici 4.17.

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

Slika 4.17 Kreiranje dopuštenja za dohvaćanje lokacije

Nakon što su napisana dopuštenja za korištenje aplikacije u *manifestu*. Potrebno je kreirati instancu klase *FusedLocationProviderClient*. *FusedLocationProviderClient* je jedan od lokacijskih API-ja u *Google Play Services*. Pomoću njega dolazi se do tehnologije za određivanje lokacije uređaja te pruža jednostavan API na kojem se mogu definirati zahtjevi poput visoke preciznosti određivanja lokacije ili male potrošnje baterije. Na slici 4.18 vidljivo je stvaranje instance klase *FusedLocationProviderClient*.

```
FusedLocationProviderClient fusedLocationProviderClient;
RadioButton rb_historian, rb_partyman, rb_gourmand, rb_allrounder;
ImageView iv_popup;
boolean updateOn=false;
LocationRequest locationRequest;
LocationCallback locationCallBack;
Location currentLocation;
List<Location> savedLocations;

fusedLocationProviderClient= LocationServices.getFusedLocationProviderClient( activity: this);
```

Slika 4.18 Instanciranje FusedLocationProviderClienta

Nakon toga, može se zatražiti posljednja poznata lokacija uz prethodni uvjet da je korisnik dopustio pristup lokaciji njegovog uređaja kao što je prikazano na slici 4.19. Kao što je vidljivo na slici 4.19, prvo se provjerava pomoću metode *checkSelfPermission()* ima li aplikacija dopuštenje za korištenje korisnikove lokacije. Ako ima, poziva se metoda *updateLocationParameters()* kojoj se predaje trenutna lokacija kao parametar, a kao rezultat poziva metode *updateLocationParameters()* ažurira se *textview* s geografskom širinom, dužinom i adresom.

```
private void refreshLocation(){
    fusedLocationProviderClient= LocationServices.getFusedLocationProviderClient( activity: this);
    if(ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_COARSE_LOCATION)== PackageManager.PERMISSION_GRANTED){
        fusedLocationProviderClient.getLastLocation().addOnSuccessListener( activity: this, new OnSuccessListener<Location>() {
            @Override
            public void onSuccess(Location location) {
                if(location!=null) {
                    updateLocationParameters(location);
                    currentLocation = location;
                }
            }
        });
    }else {
        if(Build.VERSION.SDK_INT>=23){
            requestPermissions(new String[] {Manifest.permission.ACCESS_COARSE_LOCATION}, requestCode: 1);
        }
    }
}
```

Slika 4.19 Programski kod za zatraživanje dohvaćanja lokacije korisnika

Kao što je već navedeno, sada je moguće postaviti određene zahtjeve kao što su vrijeme osvježavanja lokacije uređaja, potrošnja baterije uređaja, preciznost određivanja lokacije, itd.

Kako bi bilo moguće postaviti sve zahtjeve, potrebno je kreirati objekt klase *LocationRequest*. U toj klasi su ugrađene metode *setInterval()* koja služi za postavljanje vremenskog intervala u milisekundama za željeno vrijeme osvježavanja lokacije u aplikaciji, *setFastestInterval()* koje je najbrže moguće postavljeno vrijeme osvježavanja lokacije. Također moguće je postaviti i prioritet pomoću metode *setPriority()*. Tom metodom se određuje preciznost određivanja lokacije uređaja te potrošnja baterije uređaja dok određuje lokaciju. Može biti:

- PRIORITY_BALANCED_POWER_ACCURACY
- PRIORITY_HIGH_ACCURACY
- PRIORITY_LOW_POWER

U ovoj aplikaciji su korišteni prioriteti potrošnje baterije uređaja i preciznosti *balanced power, high accuracy* i *low power* budući da je korisniku omogućeno koje prioritete želi postaviti. Na slici 4.20 vide se vrijednosti koje su postavljene za ovu aplikaciju.

```
locationRequest=new LocationRequest();
locationRequest.setInterval(20000);
locationRequest.setFastestInterval(3000);
locationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURACY);
```

Slika 4.20 Postavljanje vrijednosti intervala

```
locationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURACY);
if(rb_low.isChecked())
    locationRequest.setPriority(LocationRequest.PRIORITY_LOW_POWER);
if(rb_high.isChecked()){
    locationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
}
```

Slika 4.21 Mijenjanje prioriteta potrošnje baterije pri dohvaćanju lokacije

Kao što su postavljeni intervali osvježavanja lokacije korisnika, tako je sada potrebno napisati programsko rješenje za osvježavanje lokacije korisnika.

Osvježavanje lokacije se postiže tako da na prethodno stvorenom *fusedLocationProviderClient* se poziva metoda *requestLocationUpdates()*. Toj metodi potrebno je proslijediti objekte klase *LocationCallback* i *LocationRequest*. *LocationCallback* se koristi za primanje informacije, ako se lokacija uređaja promijenila ili ako se više ne može pratiti, a pomoću objekta klase *LocationRequest* zatražuje se kvaliteta usluge dohvaćanja lokacije.

```
@SuppressLint("MissingPermission")
private void startLocationUpdates() {
    fusedLocationProviderClient.requestLocationUpdates(locationRequest,locationCallBack,  looper: null);
    refreshLocation();
    Toast.makeText( context: this, text: "Location is being tracked",Toast.LENGTH_LONG);
}
```

Slika 4.22 Programski kod metode startLocationUpdates

```
locationCallBack=(LocationCallback) onLocationResult(locationResult) -> {
        super.onLocationResult(locationResult);
        Location location=locationResult.getLastLocation();
        if (location != null) {
            updateLocationParameters(location);
        }
```

Slika 4.23 Dohvaćanje zadnje dostupne lokacije

Ako je potrebno zaustaviti praćenje lokacije, pozvat će se metoda *removeLocationUpdates*() kojoj će se proslijediti objekt klase *LocationCallback* kao parametar kao što je prikazano na slici 4.24. Pozivom prethodno navedene metode dolazi do prestanka praćenja geolokacije korisnika. Svi postupci rada s lokacijom uređaja pronađeni su na [5].

```
@SuppressLint("MissingPermission")
private void stopLocationUpdates() {
    tv_lat.setText("##");
    tv_lon.setText("##");
    tv_address.setText("****");
    fusedLocationProviderClient.removeLocationUpdates(locationCallBack);
    new AlertDialog.Builder( context: this).setTitle("Location is not being tracked").setMessage("Do you want " +
            "to turn Location tracking back on?").setPositiveButton( text: "YES", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            sw_locationupdates.setChecked(true);
            refreshLocation();
        }
    }).setNegativeButton( text: "NO", listener: null).show();
}
```

Slika 4.24 Programski kod metode stopLocationUpdates

4.3.3. Programska implementacija postupka rangiranja informacija o znamenitosti

U ovoj aplikaciji opisano je 10 lokacija u Tvrđi. Tekstovi o pojedinoj lokaciji su vrlo slični za svaki korisnički profil, ali je rangiranje informacija drugačije. Naravno, za neke lokacije je jednostavno nemoguće različito rangirati informacije zato što će npr. *Gourmanda* i *Partymana* jednako zanimati informacije o nekoj starijoj građevini koja ima ponajviše povijesnih informacija. Zato u nekim slučajevima rangiranje nije bilo potrebno.

Korisnik dolazi do informacija o određenoj lokaciji pritiskom na gumb „SHOW MORE“ kada dođe na lokaciju opisane znamenitosti. Kada korisnik pritisne gumb, pojavi se skočni prozor na kojem će biti slika tog mjesta te tekst s informacija o tom mjestu. Prije toga, potrebno je kreirati korisnički profil. Kada se kreira korisnički profil, potrebno je na zaslonu za dohvaćanje lokacije odabrati jedan *radio button* za svoj korisnički profil. Na temelju profila i trenutne lokacije, formira se tekst s informacija o toj znamenitosti.

```
if(distance(currentLocation.getLatitude(),currentLocation.getLongitude(), lat2: 45.56164, lng2: 18.69831)<=0.02){
    ((ImageView)popupWindow.getContentView().findViewById(R.id.iv_slika)).setImageResource(R.drawable.vodena_vrata);
    if(rb_historian.isChecked()){
        ((TextView)popupWindow.getContentView().findViewById(R.id.tv_popup_text)).setText("Vodena vrata iznimno su popularna turistička osječka a
    }
    if(rb_gourmand.isChecked()){
        ((TextView)popupWindow.getContentView().findViewById(R.id.tv_popup_text)).setText("Vodena vrata iznimno su popularna turistička osječka a
    }
    if(rb_partyman.isChecked()){
        ((TextView)popupWindow.getContentView().findViewById(R.id.tv_popup_text)).setText("Vodena vrata iznimno su popularna turistička osječka a
    }
    if(rb_allrounder.isChecked()){
        ((TextView)popupWindow.getContentView().findViewById(R.id.tv_popup_text)).setText("Vodena vrata iznimno su popularna turistička osječka a
    }
}
```

Slika 4.25 Programski kod prikazivanja dodatnih informacija za različite profile

Pomoću metode *distance* je uspoređivana trenutna lokacija s koordinatama te znamenitosti. Ovaj problem se mogao riješiti pomoću klase *Geofencing*, ali ovom metodom je bilo jednostavnije. Ako se ispuni uvjet da se opisana znamenitost nalazi u krugu od 20 metara od korisnikove trenutne lokacije i ako je označen jedan od *radio buttona* za profil, pomoću metoda *setText()* i *setImageResource()* mijenja se slika i tekst skočnog prozora. Metodi distance su predana četiri parametra (dva parametra za svaku lokaciju) te se u metodi onda uspoređuje jesu li te lokacije na određenoj udaljenosti. Ako nisu predani, neće se ništa dogoditi, a ako jesu, pritiskom na gumb „SHOW MORE“ će se prikazati skočni prozor s dodatnim informacijama o znamenitosti.

4.3.4. Programska implementacija za kreiranje korisničkog profila

Kao što je prethodno objašnjeno u tekstu, kreiranje korisničkog profila se temelji na imenu, prezimenu, spolu, godinama te interesu. Zato su kreirane radio skupine s gumbovima vezanim za ta područja, te je onda pomoću uvjeta stvoren korisnički profil te ruta koja je predložena za taj korisnički profil. Potrebno je upisati ime, prezime, te odabrati *radio buttone* koji odgovaraju istinitim vrijednostima vezanim uz korisnika te pritisniti gumb „Create Profile“ te će stvoriti profil te predefinirana ruta predložena za korisnika.

```
if(rb_female.isChecked() && rb_age3.isChecked() && rb_history.isChecked()){
    tv_profile.setText("You are: HISTORIAN");
    tv_route.setText("You should check out: Vodena Vrata Tvrđa, Muzej školjaka i vodenog svijeta, Crkva Sv. Križa, Muzej Slavonije, Crkva sv.Mihaela Ar
}
if(rb_everything.isChecked()){
    tv_profile.setText("You are: ALL ROUNDER");
    tv_route.setText("You should check out: Vodena vrata Tvrđe, Club Memories, Kružni pil Presvetog Trojstva, Pivnice i sobe Merlon, Muzej Slavonije");
}
if(rb_foodie.isChecked()){
    tv_profile.setText("You are: GOURMAND");
    tv_route.setText("You should check out: Pivnice i sobe Merlon, Club Memories, Restoran Slavonska kuća, Petar Pan, Kružni pil Presvetog Trojstva");
}
if(rb_male.isChecked()&& rb_age2.isChecked() && rb_history.isChecked()){
    tv_profile.setText("You are: HISTORIAN");
    tv_route.setText("You should check out: Vodena Vrata Tvrđa, Muzej školjaka i vodenog svijeta, Fort Pub, Muzej Slavonije, Restoran Slavonska Kuća");
}
if(rb_male.isChecked() && rb_age3.isChecked() && rb_history.isChecked()){
    tv_profile.setText("You are: HISTORIAN");
    tv_route.setText("You should check out: Muzej školjaka i vodenog svijeta, Muzej Slavonije, Pivnica i sobe Merlon, Vodena vrata Tvrđa, Arheološki mu
}
if(rb_male.isChecked()&& rb_age1.isChecked() && rb_history.isChecked()){
    tv_profile.setText("You are: HISTORIAN");
    tv_route.setText("You should check out: Vodena Vrata Tvrđa, Muzej školjaka i vodenog svijeta, Fort Pub, Muzej Slavonije, Arheološki muzej");
}
if(rb_male.isChecked()&&rb_age1.isChecked()&&rb_community.isChecked()){
    tv_profile.setText("You are: PARTYMAN");
    tv_route.setText("You should check out: Club Memories, Q club, Pivnica Bure bar, Kružni pil Presvetog Trojstva, Aeroklub Osijek");
}
```

Slika 4.26 Programski kod prikazivanja dodatnih informacija ovisno o lokaciji i profilu

U programskom kodu za kreiranje korisničkog profila korištene su metode *isChecked()* da se vidi koji *radio button* je označen te *setText()* kako bi se mogao novostvoreni korisnički profil i ruta prikazati. To je ostvareno pomoću naredbi *if* gdje se na označene određene *radio buttone* stvara novi korisnički profil i ruta.

5. PRIKAZ NAČINA RADA APLIKACIJE, ISPITIVANJE I ANALIZA REZULTATA

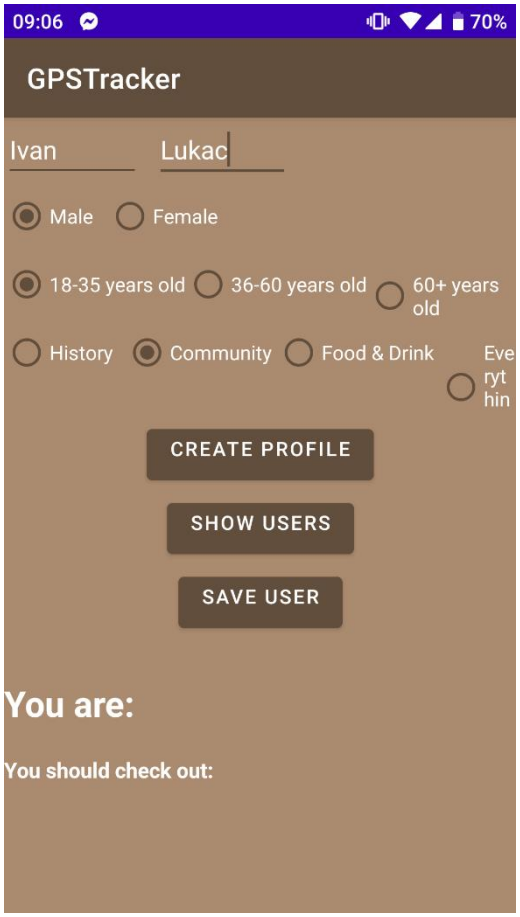
U ovom dijelu rada prikazan je rad mobilne aplikacije za turistički obilazak Tvrđe i dane su upute za njeno korištenje. Prikazat će se različiti slučajevi ispitivanja rada aplikacije kako bi se ispitale sve mogućnosti u aplikaciji. Prvo će biti prikazane upute za korištenje aplikacije kako bi čitatelj rada mogao znati pravilan način za uporabu aplikacije. Zatim će biti prikazan niz ispitivanja kojima će se provjeravati funkcionalnost aplikacije. Nakon toga će se prokomentirati rezultati ispitivanja.

5.1. Upute za korištenje aplikacije za turistički obilazak Tvrđe

Pri ulasku u aplikaciju korisnik nailazi na početni zaslon s dva gumba i pozadinskom fotografijom Tvrđe.

Slika 5.1 Početni zaslon aplikacije

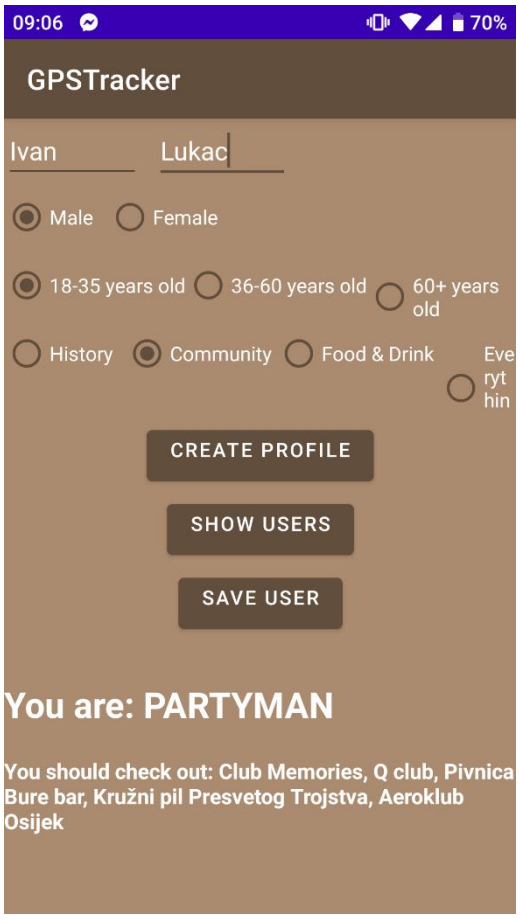
Za pravilnu uporabu aplikacije, potrebno je prvo kreirati korisnički profil. Nije obavezno na početku, ali kasnije je potrebno za pravi doživljaj aplikacije. Pritiskom na gumb „KREIRAJ PROFIL“, on vodi korisnika na novi zaslon.



Slika 5.2 Zaslon za kreiranje profila

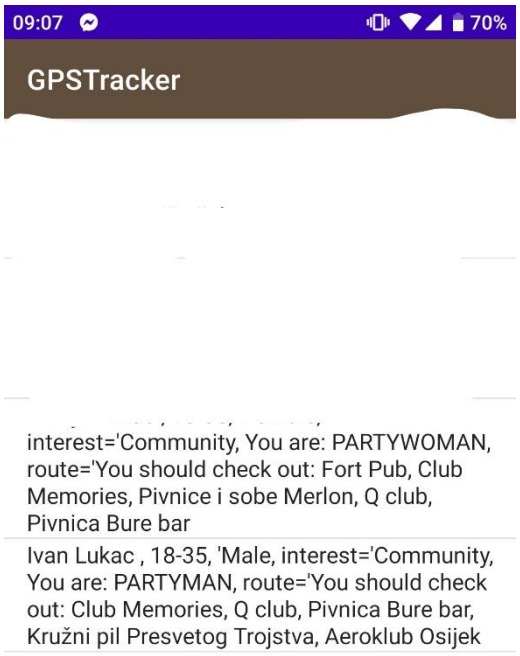
Na ovom zaslonu potrebno je upisati svoje podatke za točno kreiranje korisničkog profila, kao što je prikazano na slici 5.3. Točnost tih podataka neće se provjeravati, ali je u interesu korisnika da postavi točne podatke kako bi aplikacija bila maksimalno prilagođena njemu.

Nakon što je ispunio sve podatke, korisnik treba stisnuti gumb „CREATE PROFILE“ koji će onda prethodno opisanim algoritmima generirati profil na temelju danih informacija, kao što je prikazano na slici 5.4.



Slika 5.3 Kreirani profil i ruta

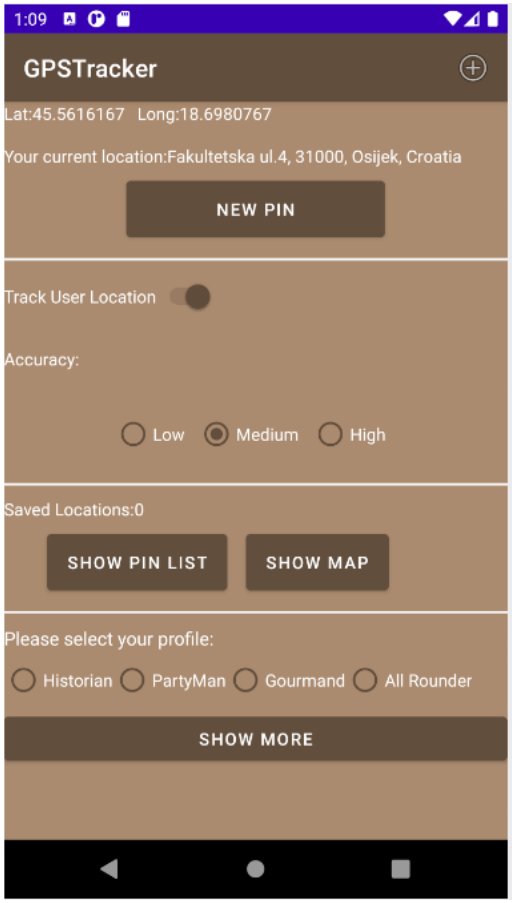
Ako želi, korisnik može spremiti svoj profil. To će postići pritiskom na gumb „SAVE USER“ preko kojega će se kreirani korisnički profil spremiti u postojeću bazu podataka. Listu svih spremljenih korisničkih profila moguće je vidjeti pritiskom na gumb „SHOW USERS“ kao što je prikazano na slici 5.5.



Slika 5.4 Spremljeni korisnički profili

Nakon uspješnog kreiranja korisničkog profila, dolazi se na zaslon za dohvaćanje lokacije. Bitno je reći, neke značajke ove aplikacije mogu se koristiti i izvan Tvrđe, kao što je spremanje lokacije koje je korisnik posjetio i prikazivanje istih. No za potpuni doživljaj aplikacije, korisnik bi trebao koristiti aplikaciju kada se nalazi u Tvrđi.

Nakon što je korisnik kreirao profil i spremio ga, dolazi do zaslona za dohvaćanje lokacije koji je prikazan na slici 5.6 i na kojem će izvršavati niz sljedećih postupaka.



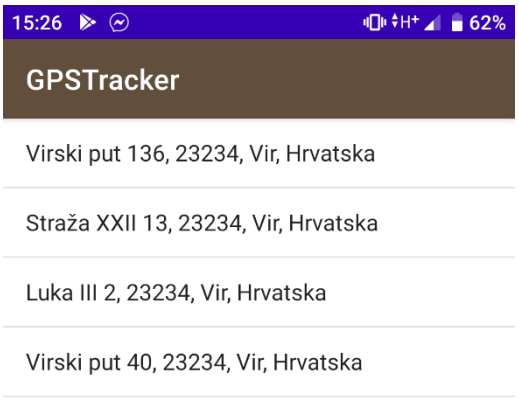
Slika 5.6 Zaslon za dohvaćanje lokacije

Nakon što je korisnik došao do određene znamenitosti u Tvrđi, treba odabrati jedan gumb iz skupine *radio button* koji označava profil koji je dodijeljen korisniku. Nakon toga, potrebno je pritisnuti gumb „SHOW MORE“ pomoću kojeg će doći do skočnog prozora kao što je prikazano na sljedećoj slici 5.7.

Slika 5.5 Skočni prozor s dodatnim informacijama

U skočnom prozoru se nalazi slika znamenitosti te tekst na hrvatskom jeziku koji se može pomicati (eng. *Scrollable*). Za svaki profil je u velikom broju slučajeva drugačiji tekst, osim na mjestima gdje to jednostavno nema smisla.

Ako se korisniku svidjela znamenitost, pomoću gumba „NEW PIN“ može spremiti lokaciju tog mjesta na globalnu listu lokacija. Pritiskom na gumb „SHOW SAVED LOCATIONS“ prikazat će se sve lokacije koje je korisnik spremio u obliku adresa na jednostavnoj listi. Korisnik može spremiti sve lokacije na svijetu, ne samo one u Tvrđi, kao što je prikazano na slici 5.8.



Slika 5.6 Lista spremljenih lokacija

Također, moguće je prikazati sve spremljene lokacije s globalne liste na *Google Maps* zaslonu. To će se postići stiskom na gumb „SHOW MAP“ što će biti prikazano na sljedećoj slici. Za svaku lokaciju na globalnoj listi stvara se „pribadača“ koja se prikazuje na zaslonu kao što je i vidljivo na slici 5.9.

Slika 5.7 Spremljene lokacije prikazane na Google Maps zaslonu

Kao što je prethodno opisano, neke funkcionalnosti aplikacije rade i izvan lokaliteta Tvrđe, kao što je na primjer spremanje lokacija na globalnu listu i njihovo prikazivanje.

5.2. Uvjeti i korisnički slučajevi ispitivanja rada aplikacije

U ovome dijelu rada isproban je rad aplikacije u određenim uvjetima i slučajevima kako bi se ispitala ispravnost i kvaliteta aplikacije. Nadalje će biti opisani slučajevi ispitivanja i dobiveni rezultati.

5.2.1. Definiranje uvjeta ispitivanja aplikacije

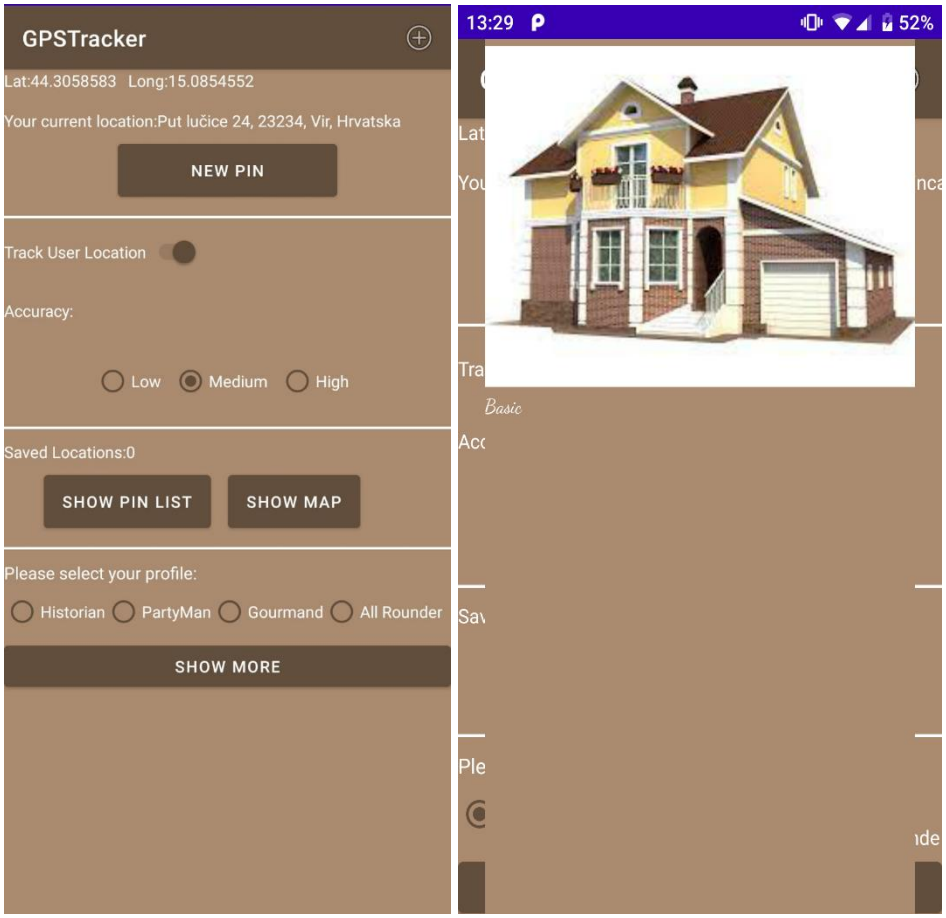
Ne postoje nikakvi uvjeti za kreiranje korisničkog profila, osim popunjavanja svih polja radio buttona. Zato nije potrebno ispitivanje kreiranja profila. Više će se ispitivati funkcionalnosti aplikacije vezane uz lokaciju. Prvo će se provjeravati hoće li aplikacija pritiskom na gumb „SHOW MORE“ prikazati više informacija samo na mjestu znamenitosti za koju je napravljen opis ili za sve lokacije izbacuje dodatan tekst. To ispitivanje je nazvano Ispitivanje točnog

dohvaćanja lokacije. Nakon toga će se testirati hoće li se sve lokacije (i one van Tvrđe) spremiti na globalnu listu lokacija. Taj slučaj će se zvati Ispitivanje spremanja i prikazivanja lokacija. Posljednji test bit će provjera hoće li se promjenom korisničkog profila promijeniti tekst o istoj znamenitosti, a dobit će ime Ispitivanje promjene teksta s promjenom profila.

5.2.2. Slučajevi korištenja aplikacije

Prvo se obavlja Ispitivanje točnog dohvaćanja lokacije i prikazivanje dodatnih informacija o znamenitosti. Usporedit će se mjesto u Tvrđi te mjesto izvan Tvrđe i tekstovi koji će biti prikazani za njih.

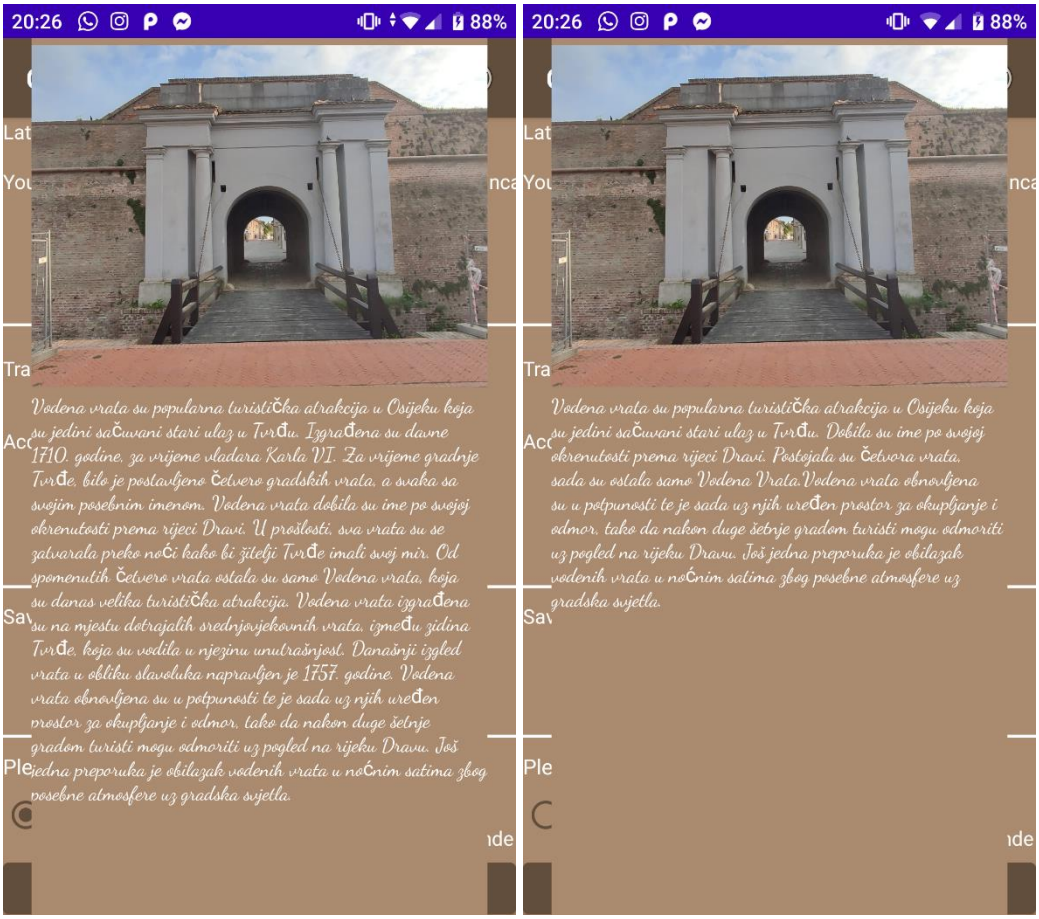
U prvom slučaju lokacija uređaja je postavljena na mjesto na otoku Viru što predstavlja slučaj lokacije izvan Tvrđe. U nastavku će se vidjeti rezultat ispitivanja na slici 5.10.



Slika 5.9 i 5.10. Prikazivanje dodatnih informacija za mjesto izvan Tvrđe

Kao što se vidi iz priloženih slika, kada se postavi lokacija izvan Tvrđe ili jedna od lokacija u Tvrđi koja nije opisana u aplikaciji, skočni prozor će prikazivati jednu generičku sliku i tekst basic.

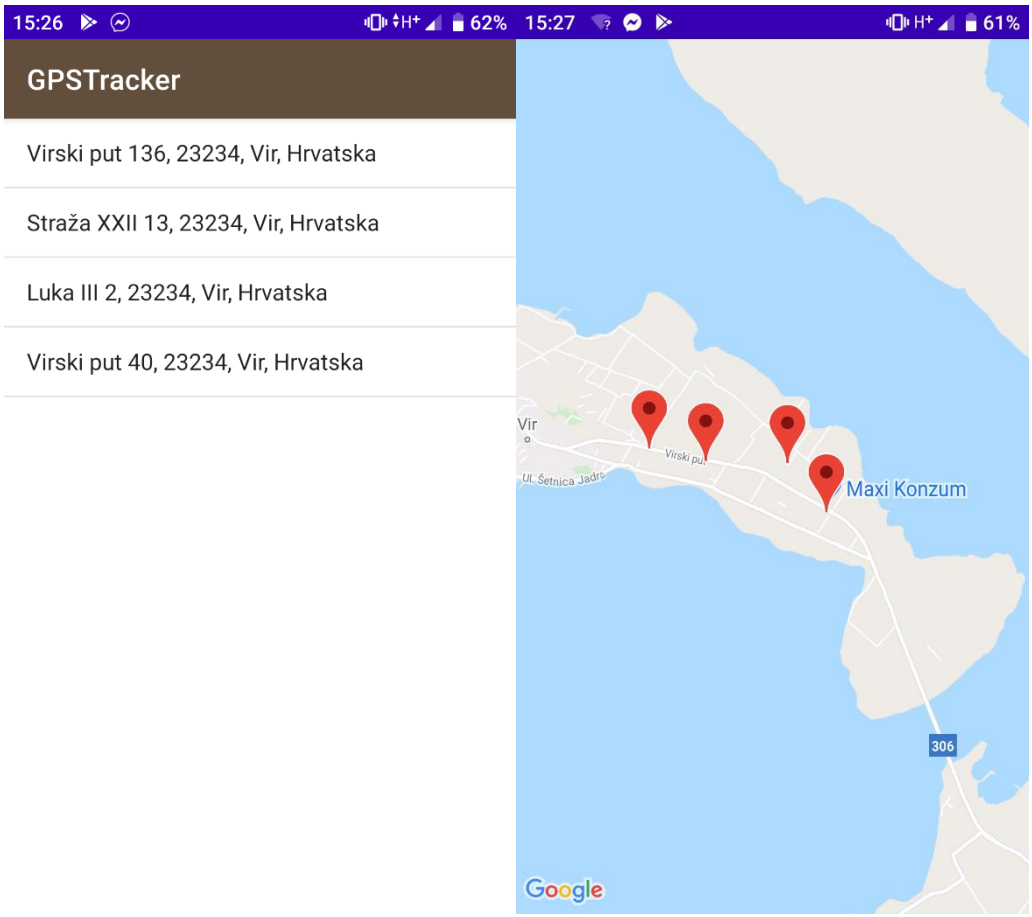
Drugi slučaj će biti jedna od opisanih lokacija u Tvrđi. Lokacija uređaja bit će podešena na Vodena Vrata u Tvrđi koja su opisana u aplikaciji i skočni prozor bi trebao dati dodatne informacije i sliku. Ovim slučajem provjerava se dohvaćanje lokacije u aplikaciji te opisivanje znamenitosti koje se nalaze u Tvrđi.



Slika 5.11 i 5.12 Prikazivanje dodatnih informacija za mjesto u Tvrđi

Kao što je vidljivo iz prikazanih slika, kada se dođe na adresu na kojoj se nalaze Vodena Vrata u Tvrđi, pritiskom na gumb „SHOW MORE“ će biti prikazan dodatni tekst o znamenitosti i slika koja prikazuje tu znamenitost. Ovim ispitivanjem je utvrđeno da je Ispitivanje točnog dohvaćanja lokacije i dodantih informacija u tekstu uspješno izvedeno.

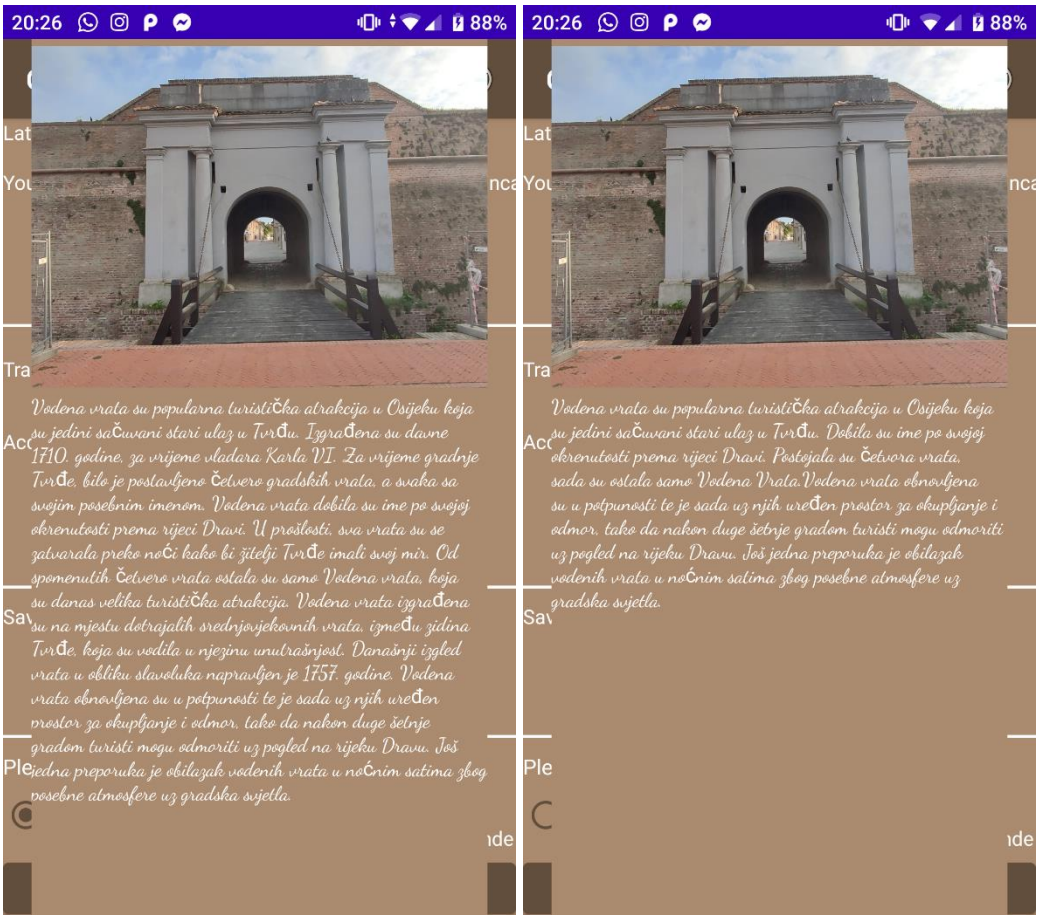
Druga funkcionalnost koja će se testirati je mogućnost spremanja lokacije koje se nalaze izvan Tvrđe na globalnu listu lokacije te kasnije prikazati na jednostavnu listu adresa tih lokacija. Ovim postupkom se provjerava spremanje lokacija na globalnu listu i prikazivanje lokacija, na jednostavnoj listi i na *Google Maps* zaslonu pomoću *Google Maps* API ključa.



Slika 5.13 i 5.14 Prikazivanje spremljenih lokacija

Kao što je vidiljivo na slikama, sve adrese su na otoku Viru (izvan Tvrđe) su spremljene na globalnu listu te prikazane na jednostvanoj listi adresa tih lokacija te na zaslonu Google Mapsa. Ovim slučajem je potvrđena ispravnost Ispitivanja spremanja i prikazivanja lokacija.

Nakon toga, ispituje se promjena teksta o znamenitosti s obzirom na promjenu korisničkog profila. Predmet ispitivanja bit će Vodena Vrata u Tvrđi. Na slikama 5.15 i 5.16 su prikazana dva različita profila te različiti tekstovi o istoj znamenitosti. Ovim slučajem provjerava se hoće li se s promjenom korisničkog profila promijeniti i tekst o znamenitosti.



Slika 5.15 i 5.16 Prikazivanje dodatnih informacija o istom mjestu za različite profile

Kao što je vidljivo na slikama, kada se označe različiti profili, dolazi da različitog teksta o znamenitosti. Na prvoj slici je označen profil *Historian*, a na drugoj *Partyman*. Pomoću ovih slika potvrđena je točnost algoritma za rangiranje informacija o znamenitosti temeljeno na korisničkom profilu i geolokaciji, čime je potvrđeno Ispitivanje promjene teksta s promjenom profila.

5.3. Analiza rada aplikacije

Aplikacija je vrlo jednostavna za korištenje, dizajn aplikacije je vrlo jednostavan i sve funkcionalnosti su lako dostupne i vrlo lako se koriste. Početni zaslon je dodan ponajviše radi dizajna i „*ljepote*“ aplikacije. Kreiranje korisničkog profila je napravljeno što jednostavnije i nije potrebno davati točne brojeve npr. godina, što nije ugodno svim korisnicima. Kada se kreira korisnički profil, velikim slovima piše koji korisnički profil je dodijeljen i preporučena ruta za obilazak. Kada se dođe do zaslona za dohvaćanje lokacije na njemu se jasno vidi trenutna adresa na kojoj se korisnik nalaze te geografska širina i dužina. Postoji opcija namještanja točnosti

određivanja lokacije te potrošnja baterije uređaja pri određivanju lokacije pomoću *radio buttona*. Također, vrlo lako se može ugasiti praćenje lokacije ako korisnik ne želi da se zna njegova trenutna lokacija. Dohvaćanje lokacije se odvija u vremenu kraćem od jedne sekunde i sve funkcionalnosti se ostvaruju vrlo brzo. Provjeravalo se točno dohvaćanje lokacije, točno kreiranje korisničkog profila, spremanje željenih lokacija te prikaz istih, prikazivanje dodatnih informacija o znamenitosti, drukčije rangiranje informacija o znamenitosti temeljeno na različitim profilima. Za vrijeme ispitivanja ispravnosti rada ove aplikacije, u svim testovima su dobiveni očekivani rezultati kao što je prikazano u radu i za svako testiranje su priložene slike kao dokaz uspješnosti ispitivanja. Na slikama 5.9 i 5.11 vidljivo je točno dohvaćanje gelokacije korisničkog uređaja. Na slikama 5.15 i 5.16 vidi se da se promjenom korisničkog profila mijenja i tekst o određenoj znamenitosti. Sve lokacije koje je korisnik želio spremiti, mogao je vrlo lako prikazati kao što se vidi na slikama 5.13 i 5.14, i to ne samo lokacije unutar Tvrđe, nego bilo gdje na svijetu. Kao što je prikazano, ispitivanjem aplikacije na slici 5.4 pomoću označenih *radio buttona* kreira se određeni korisnički profil i preporučena ruta. Kao što je potvrđeno trima ispitivanjima (*Ispitivanje točnog dohvaćanja lokacije*, *Ispitivanje spremanja i prikazivanja lokacija* i *Ispitivanje promjene teksta s promjenom profila*), korištenje aplikacije pokazalo se vrlo dobrim, nema neočekivanih kvarova i/ili problema. Aplikacija nije zahtjevna i veliki broj uređaja ju može koristiti bez ikakvih problema s velikom brzinom.

6. ZAKLJUČAK

Godinama je turizam jedna od gospodarskih grana s najvećim rastom i svake godine sve je veće zanimanje za obilaskom novih mjesta. Kako raste broj ljudi koji posjećuju druga mjesta, tako raste i broj ljudi koji koriste pametne uređaje u svakodnevnom životu. Stoga je došlo do potrebe za što većim uvođenjem tehnologije u turizam. Uz već postojeća rješenja, cilj ovog rada bio je ostvariti mobilnu aplikaciju koja će olakšati korisnicima dolazak do novih informacija dostupnih na jednom mjestu. U ovom konkretnom slučaju, na 10 lokacija unutar Tvrđe korisnici mogu saznati mnoštvo informacija o opisanim znamenitostima. Ova mobilna aplikacija nudi korisniku kreiranje vlastitog profila te stvaranje preporučene rute na temelju njegovog profila. Također, u ovoj aplikaciji moguće je saznati dodatne informacije o znamenitosti koju korisnik obilazi. Informacije o toj znamenitosti rangiraju se na temelju njegovog profila, odnosno korisnikovog interesa.

Aplikacija je napravljena u programskom okruženju Android Studio. Pomoću XML-a dizajnirana je aplikacija, dok su sve funkcionalnosti aplikacije ostvarene pomoću programskog jezika Java. Za spremanje podataka vezanih za profil korisnika korištena je baza podataka Room. Za korištenje *Google Maps* zaslona bio je potreban *Google Maps* API. Zamišljeni funkcionalni i nefunkcionalni zahtjevi ove aplikacije uspješno su ostvareni tako što se s njima omogućilo dohvaćanje lokacije, kreiranje korisničkog profila, rangiranje dodatnih informacija o opisanim znamenitostima. Aplikacija je ispitana za više slučajeva koji su opisani i prikazani u petom poglavlju. Nakon provedena tri ispitivanja, utvrđeno je da je aplikacija funkcionalna, ali kao i svugdje, moguće ju je poboljšati s nekoliko dodatnih funkcionalnosti kao što je na primjer proširenje područja na primjerice cijeli grad Osijek ili regiju Slavoniju.

LITERATURA

[1] Proleksis Enciklopedija, Turizam https://proleksis.lzmk.hr/49440/ [Pristupljeno 11 lipanj 2021].

[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. Smirnov, A. Kashevnik, N. Shilov, N. Teslya, A. Shabaev, Mobile Application for Guiding Tourist Activities: Tourist Assistant - TAIS, St. Petersburg, SPIIRAS, str. 94-96

[4] Fakultet elektrotehnike i računarstva, Prezentacija: Vrednovanje nefunkcijskih obilježja raspodijeljenih sustava, str. 6, https://www.fer.unizg.hr/_download/repository/RS-2020_10.pdf [Pristupljeno 3 srpanj 2021]

[5] Android Developers, Build location-aware apps, 24 svibanj 2021. https://developer.android.com/training/location. [Pristupljeno 1 lipanj 2021].

[6] V. Kasapakis, C. Konstantopoulos, K. Mastakas, D. Gavalas, Scenic Athens: A Personalized Scenic Route, *IEEE Symposium on Computers and Communication (ISCC)*, 2016.,str. 1-2

[7] S. Zhu, H. Alghamdi, A. El Saddik, E-Tourism: Mobile Dynamic Trip Planner, *IEEE International Symposium on Multimedia*, 2016. str. 1

[8] W. Gao, Z. Yia, X. Chen, Y. Yang, User-based Collaborative Filtering for Tourist Attraction, *IEEE International Conference on Computational Intelligence & Communication Technology*, 2015. str. 1-3

[9] C. Schmidt, What is Android? Here is a Complete Guide for Beginners, 24 lipanj 2016. https://www.nextpit.com/what-is-android. [Pristupljeno 1 srpanj 2021].

[10] J. Alabaster, Android founder: We Aimed to Make a Camera OS, 16 travanj 2013. https://www.pcworld.com/article/2034723/android-founder-we-aimed-to-make-a-camera-os.html. [Pristupljeno 1 srpanj 2021].

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

[12] J. Byous, Java Technology: The Early Years, 2003., https://web.archive.org/web/20090311011509/http://java.sun.com/features/1998/05/birthday.html [Pristupljeno 1 srpanj 2021].

[13] How Java Works, https://python-tricks.com/how-java-works/. [Pristupljeno 2 srpanj 2021].

[14] »XML-Overview, https://www.tutorialspoint.com/xml/xml_overview.htm. [Pristupljeno 2 srpanj 2021].

[15] Android Developers, Android, https://developer.android.com/. [Pristupljeno 2 srpanj 2021].

[16] Adding a map with marker, Google, https://developers.google.com/maps/documentation/android-sdk/map-with-marker. [Pristupljeno 15 lipanj 2021].

[17] S. Sluiter, *Android Studio Tutorial - Build a GPS App,* freeCodeCamp.org, 2021.

[18] Android Developers, Save Data in a local database using Room, https://developer.android.com/training/data-storage/room. [Pristupljeno 2 lipanj 2021].

[19] Android Developers, Build location aware apps,]https://developer.android.com/training/location. [Pristupljeno 4 travanj 2021].

ŽIVOTOPIS

Ivan Lukac rođen je 4.11.1999. u Osijeku. Nakon pohađanja Osnovne škole Ivan Goran Kovačić u Đakovu. 2014. godine upisuje opću gimnaziju Antun Gustav Matoš u Đakovu te 2018. završava gimnaziju te upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, smjer računarstvo. 2019. godine polaže ispit za turističkog vodiča za četiri lokalne županije. Govori dva strana jezika, engleski i njemački, te poznaje mnoge programske jezike kao što su C, C++, Java i C#. Voli se baviti rukometom i košarkom, te svirati klavir.

_____

Potpis autora

SAŽETAK

U ovom završnom radu osmišljena je i programski ostvarena Android mobilna aplikacija za obilazak turističkih sadržaja u Tvrđi. Ova aplikacija je napravljena u Android Studiju gdje je za funkcionalnosti aplikacije korišten programski jezik Java, a za dizajn opisni jezik XML. Aplikacija omogućuje korisniku stvaranje profila i preporučene rute za obilazak te mogućnost dohvata novih informacija o određenim znamenitostima prema kreiranom korisničkom profilu. Korisnički profil moguće je spremiti u bazu podataka i prikazati na jednostavnoj listi. Korisniku se nudi opcija spremanja bilo koje lokacije koju je posjetio na globalnu listu lokacija. Korisniku je omogućeno prikazati spremljene lokacije na listi ili na zaslonu *Google Mapsa*. Ispitivanjem funkcionalnosti mobilne aplikacije utvrđena je njena ispravnost na primjerima korištenja. Cilj rada je omogućiti mobilnu aplikaciju koja turistima olakšava obilaske i smanjuje troškove obilaska znamenitosti. Također, svrha ove aplikacije je stvaranje kvalitetnog turističkog obilaska za niz različitih ljudi koji posjećuju Tvrđu uz minimalne napore turista i rješavanje problema nepersonaliziranih turističkih obilazaka ponudom rangiranja informacija temeljenih na korisnikovim sklonostima.

Ključne riječi: Android mobilna aplikacija, geolokacija, preporuka rute, turistički obilazak, turizam.

ABSTRACT

Mobile Android Application for support the visiting of touristic attractions

In this Bachelor's thesis, an Android mobile application for touring tourist facilities in the Fortress was designed and programmatically realized. This application was made in the Android Studio where the Java programming language was used for the application's functionalities, and the XML descriptive language for the design of the application. The application allows the user to create a profile and recommended tour routes, as well as the ability to retrieve new information about certain attractions according to the created user profile. The user profile can be saved in a database and displayed in a simple list. The user is offered the option of saving any location he has visited to the global list of locations. The user is allowed to view saved locations in a list or on the Google Maps screen. By examining the functionality of the mobile application, its correctness was determined on examples of use. The aim of the paper is to provide a mobile application that makes it easier for tourists to visit and reduces the cost of sightseeing. Also, the purpose of this application is to create a quality tour for a number of different people who visit the Fortress with minimal effort of tourists and solve the problem of non-personalized tours by offering ranking information based on user preferences.

Keywords: Android mobile application, geolocation, route recommendation, sightseeing tour, tourism.

PRILOZI

Prilog 1. Završni rad u formatu .docx

Prilog 2. Završni rad u formatu .pdf

Prilog 3. Projekt izrađene mobilne aplikacije