Implementiranje mobilnog marketinga u promocijski miks integrirane marketinške komunikacije putem android aplikacije namijenjene posjetiteljima Nacionalnog parka Paklenica

Tivanovac, Matija

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni diplomski studij računarstva

IMPLEMENTIRANJE MOBILNOG MARKETINGA U PROMOCIJSKI MIKS INTEGRIRANE MARKETINŠKE KOMUNIKACIJE PUTEM ANDROID APLIKACIJE NAMIJENJENE POSJETITELJIMA NACIONALNOG PARKA PAKLENICA

Diplomski rad

Matija Tivanovac

Osijek, 2020.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 09.10.2020.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime studenta:	Matija Tivanovac
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D 895 R, 25.09.2019.
OIB studenta:	51646219792
Mentor:	Prof. dr. sc. Dominika Crnjac Milić
Sumentor:	Izv. prof. dr. sc. Krešimir Nenadić
Sumentor iz tvrtke:	Ivana Adžić
Predsjednik Povjerenstva:	Doc.dr.sc. Zdravko Krpić
Član Povjerenstva 1:	Prof. dr. sc. Dominika Crnjac-Milić
Član Povjerenstva 2:	Izv. prof. dr. sc. Alfonzo Baumgartner
Naslov diplomskog rada:	Implementiranje mobilnog marketinga u promocijski miks integrirane marketinške komunikacije putem android aplikacije namijenjene posjetiteljima Nacionalnog parka Paklenica
Znanstvena grana rada:	**Informacijski sustavi (zn. polje računarstvo)**
Zadatak diplomskog rada:	Definirati osnovne pojmove i teorijske pretpostavke mobilnog marketinga kao sastavnog dijela integrirane marketinške komunikacije, te izraditi android aplikaciju kao primjer implementiranja modernih tehnologija u turizmu, a u svrhu individualiziranih marketinških komunikacija. Izraditi android aplikaciju u svrhu dojave nalazišta Natura 2000 vrsta. U aplikaciji bi postojao popis Natura 2000 vrsta za nacionalni park Paklenica i park prirode Velebit (fotografije, zanimljivosti i sl.). Aplikacija bi funkcionirala na slijedeći način; ako korisnik zabilježi vrstu (fotografira ju), aplikacija ima pristup lokaciji i ti se podaci, zajedno s vremenom opažanja, šalju na predodređeni e-mail upravi Nacionalnog parka. Uprava te podatke bilježi i koristi u svrhu
Prijedlog ocjene pismenog dijela ispita (diplomskog 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: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	09.10.2020.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

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

IZJAVA O ORIGINALNOSTI RADA

Osijek, 14.10.2020.

Ime i prezime studenta:	Matija Tivanovac
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D 895 R, 25.09.2019.
Turnitin podudaranje [%]:	8%

Ovom izjavom izjavljujem da je rad pod nazivom**: Implementiranje mobilnog marketinga u promocijski miks integrirane marketinške komunikacije putem android aplikacije namijenjene posjetiteljima Nacionalnog parka Paklenica**

izrađen pod vodstvom mentora Prof. dr. sc. Dominika Crnjac Milić

i sumentora Izv. prof. dr. sc. Krešimir Nenadić

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:

Table of Contents

1. UVOD

Cilj ovog diplomskog rada je izrada Android mobilne aplikacije za Nacionalni park Paklenicu. Aplikacija bi imala bitnu ulogu u promociji parka, ali bi se mogla koristiti i za istraživačke svrhe u smislu praćenja rasprostranjenosti životinjskih i biljnih vrsta na području parka. Cilj aplikacije je omogućiti posjetiteljima nacionalnog parka pregled ugroženih i zanimljivih biljnih i životinjskih vrsta i način obavještavanja uprave gdje i kada su tu vrstu vidjeli.

Mobilni uređaji kao i aplikacije koje se na njima pokreću još uvijek se razvijaju. Prema [1] istraživanja su pokazala da je broj korisnika mobilnih uređaja toliko porastao da je prije nekoliko godina bio veći od broja korisnika osobnih računala za nekoliko stotina tisuća korisnika. Kao posljedica toga mobilne aplikacije danas predstavljaju bitan dio naše svakodnevice i čine omiljeni način na koji korisnici vrše interakciju s pružateljima usluga i proizvoda. Tvrtke zato ulažu sve više sredstava u mobilni marketing.

Iz spomenutog se mogu pretpostaviti potencijalne koristi Android aplikacije za nacionalni park. Nacionalni park Paklenica je jedan od prvih i jedan od najposjećenijih nacionalnih parkova u Republici Hrvatskoj. Utemeljen 1949. godine, Nacionalni park Paklenica se prostire na 95 km^2, a obuhvaća područje poznatih kanjona Velike i Male Paklenice. Nacionalni park štiti najočuvaniji i najveći šumski kompleks na području Dalmacije, a turistima je atraktivan zbog „svoje jedinstvene prirodne osnove, izuzetnih geomorfoloških oblika i veličanstvenih šuma“ [2]. Turisti park u cijelosti mogu doživjeti samo pješačenjem po mnogobrojnim planinarskim stazama. To iskustvo istraživanja i osobnog doživljaja nacionalnog parka ova aplikacija nastoji produbiti, učiniti zanimljivijim što će u konačnici privući veći broj posjetitelja ali i potaknuti postojeće ljubitelje parka na ponovnu posjetu. Aplikacija posjetiteljima omogućava interaktivno iskustvo putem aktivnog sudjelovanja tijekom istraživanja svih dobrobiti parka i pruža edukaciju o istaknutim prisutnim vrstama biljnog i životinjskog svijeta čime također potiče ekološku osviještenost i ističe potrebu za očuvanjem zaštićenih i ugroženih vrsta.

Osim općeg uvoda i cilja ovog rada opisanog u ovom poglavlju, u drugom poglavlju opisane su slične aplikacije i uspoređene sa zadanom. U trećem poglavlju dane su kratke teorijske osnove potrebne za bolje razumijevanje izrađene aplikacije. Četvrto poglavlje sadrži u potpunosti opisanu izradu zadane aplikacije, nakon čega slijedi njeno testiranje na virtualnom i fizičkom mobilnom uređaju. Zatim je dan teorijski uvid u mobilni marketing i promociju i moguću primjenu ove aplikacije u te svrhe. Na kraju se nalazi zaključak o korisnosti aplikacije u svrhu promocije parka i proučavanja ugroženih biljnih i životinjskih vrsta.

1.1. Zadatak diplomskog rada

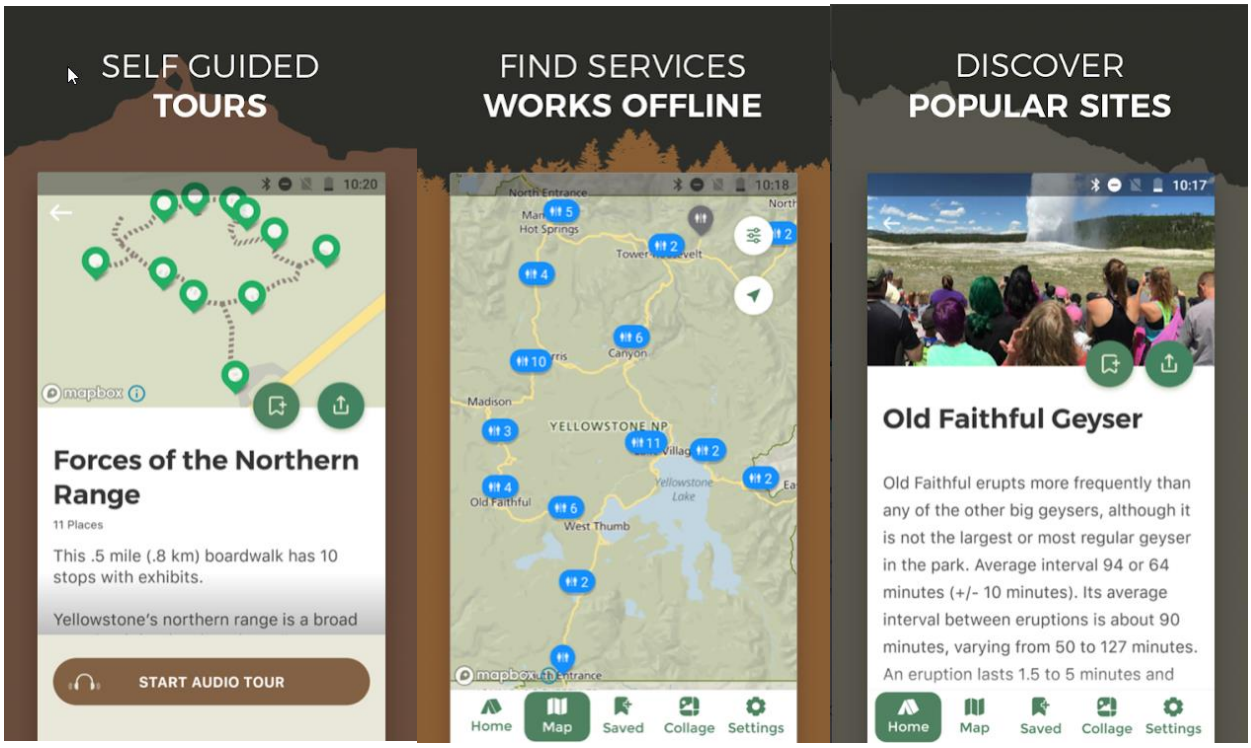
Definirati osnovne pojmove i teorijske pretpostavke mobilnog marketinga kao sastavnog dijela integrirane marketinške komunikacije, te izraditi android aplikaciju kao primjer implementiranja modernih tehnologija u turizmu, a u svrhu individualiziranih marketinških komunikacija. Izraditi android aplikaciju u svrhu dojave nalazišta Natura 2000 vrsta. U aplikaciji bi postojao popis Natura 2000 vrsta za nacionalni park Paklenica i park prirode Velebit (fotografije, zanimljivosti i sl.). Aplikacija bi funkcionirala na slijedeći način; ako korisnik zabilježi vrstu (fotografira ju), aplikacija ima pristup lokaciji i ti se podaci, zajedno s vremenom opažanja, šalju na predodređeni e-mail upravi Nacionalnog parka. Uprava te podatke bilježi i koristi u svrhu praćenja stanja životinjskih i biljnih vrsta. Simbolična nagrada bi bila dodijeljena od strane Nacionalnog parka svakom N-tom opažaču. (npr. 100-tom opažaču) Aplikacija bi bila korisna i zanimljiva posjetiteljima Nacionalnog parka Paklenica, jer sudjeluju u praćenju stanja ciljanih vrsta biljaka i životinja, a korisna bi bila i upravi Nacionalnog parka jer bi dobivali informacije o rasprostranjenosti važnih vrsta biljaka i životinja.

2. POSTOJEĆE SLIČNE REALIZACIJE

U ovom poglavlju navedene su neke od postojećih, sličnih aplikacija i opisane su njihove međusobne sličnosti i razlike. Opisi su potkrijepljeni slikama pojedinih aplikacija.

2.1. NPS Yellowstone

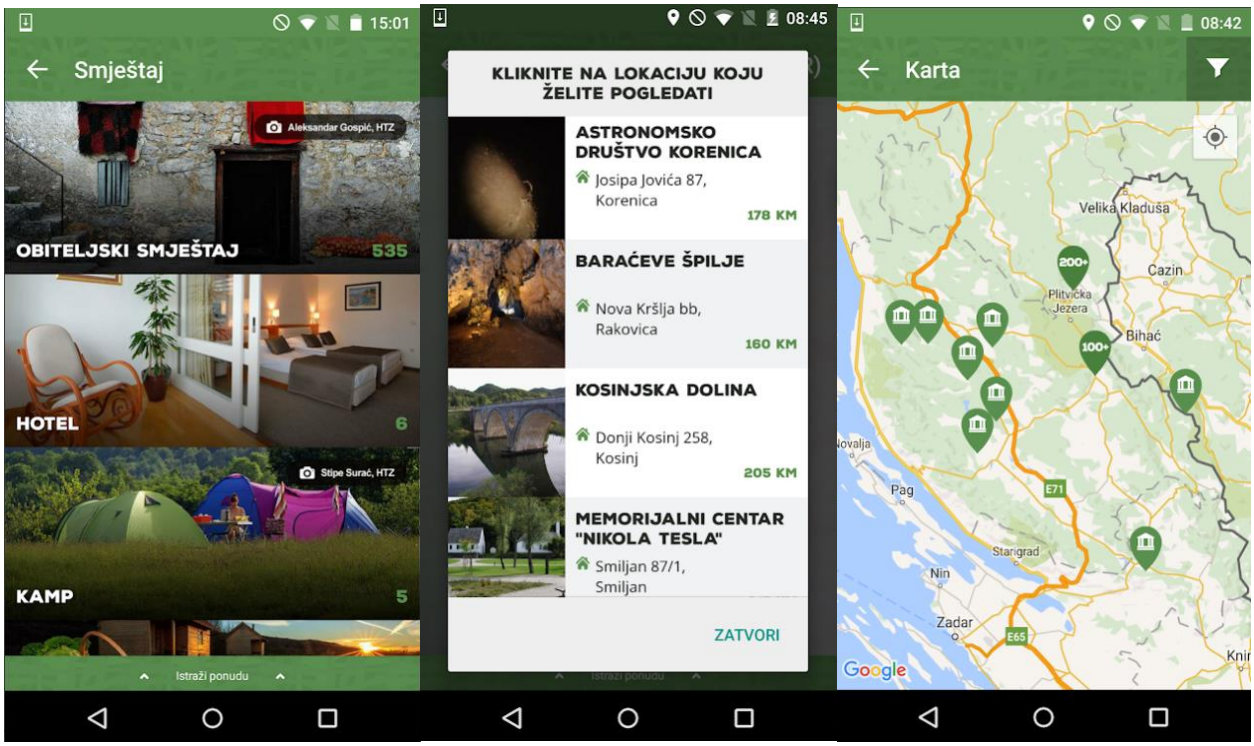
NPS Yellowstone je službena aplikacija nacionalnog parka Yellowstone. Ona je dio NPS (*National Park Service*) sustava koji se sastoji od web stranica i aplikacija za neke od nacionalnih parkova u Americi. Sve ove aplikacije imaju iste funkcionalnosti s razlikom u sadržaju koji ovisi o parku. Aplikacije sadrže kartu koja ima označene bitne znamenitosti parka i usluge, te smjernice. Moguće je vidjeti detalje i opis svake znamenitosti, spremiti najdraže te dobiti upozorenja kada se korisnik nalazi u blizini neke znamenitosti. Također je implementiran kalendar s događanjima i novostima te audio vodič. Aplikacije je prilično opsežna, međutim, korisnicima nudi samo informativan sadržaj. Ne postoji nikakva mogućnost interakcije između korisnika i parka niti pruža istraživačke i znanstvene mogućnosti aplikacije izrađene u sklopu ovog rada. Primjer je vidljiv na slici 2.1.



Sl. 2.1. NPS Yellowstone aplikacija [3]

2.2. Plitvička jezera

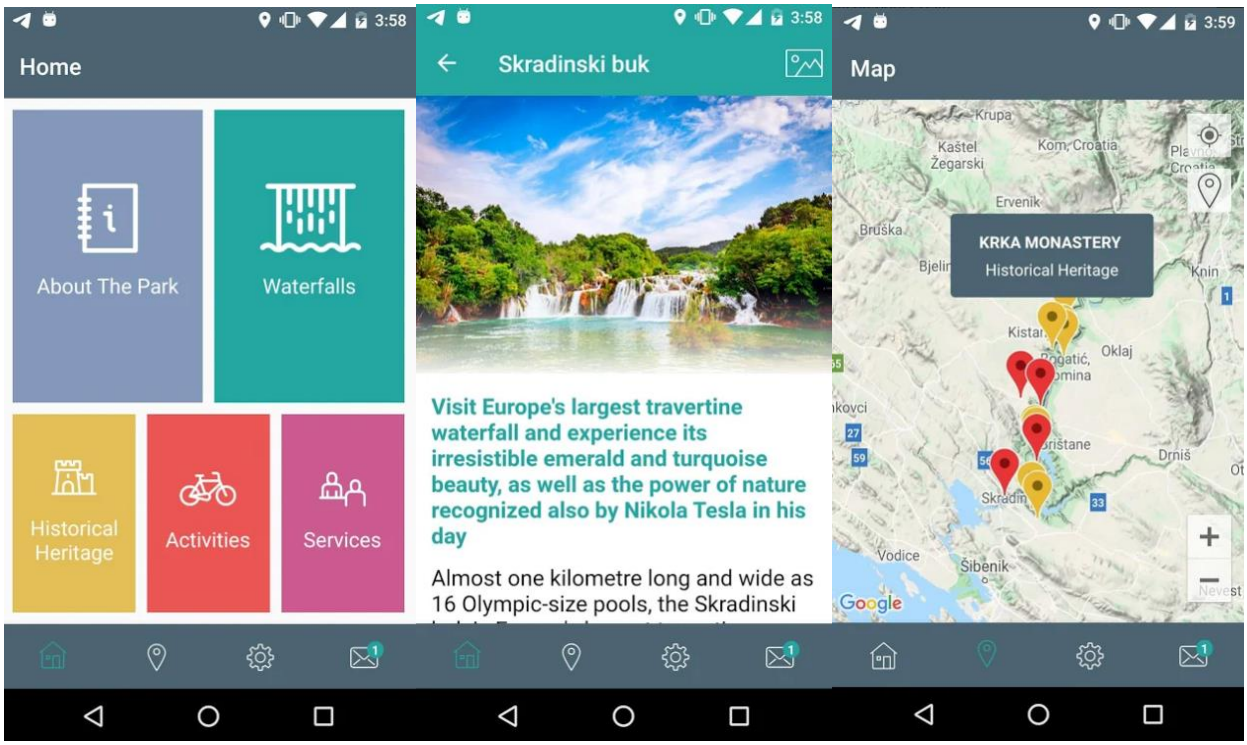
Aplikacija Plitvička jezera je službena aplikacija ovog nacionalnog parka, izrađena uz pomoć vanjske tvrtke. Sastoji se od karte koja ima označene sve znamenitosti i usluge parka. Detalje i opise svake znamenitosti moguće je pretraživati prema opisu i udaljenostima od trenutne lokacije. Sadrži kalendar događanja i lociranje znamenitosti pomoću proširene stvarnosti. Aplikacija je vrlo opširna i pruža korisniku sve potrebne informacije, ali u usporedbi s aplikacijom ovog rada ne nudi korisnicima mogućnost da svojim korištenjem aplikacije pomognu u istraživanju i praćenju ugroženih vrsta na području parka. Primjer aplikacije nalazi se na slici 2.2.



Sl. 2.2. Plitvička jezera aplikacija [4]

2.3. NP Krka

NP Krka je službena aplikacija Nacionalnog parka Krka. Sadrži kartu na kojoj su označene sve znamenitosti i usluge. Korisniku pruža sve informacije o događajima, aktivnostima i atrakcijama. Sadrži galeriju sa slikama i prikaz svih vodopada na području parka. Aplikacija pruža korisniku sve potrebne informacije, ali ne nudi interaktivan sadržaj koji bi korisniku omogućio sudjelovanje i pružanje pomoći prilikom istraživanja ugroženih vrsta na području parka poput aplikacije izrađene u sklopu ovog rada. Primjer NP Krka aplikacije prikazan je na slici 2.3.



Sl. 2.3. NP Krka aplikacija [5]

3. TEORIJSKE OSNOVE OPERACIJSKOG SUSTAVA I KORIŠTENIH TEHNOLOGIJA I ALATA

Ovo poglavlje daje kratke teorijske osnove koje su potrebne kako bi se bolje razumjeli alati korišteni za izradu zadane aplikacije i operacijski sustav koji ju pokreće.

3.1. Android operacijski sustav

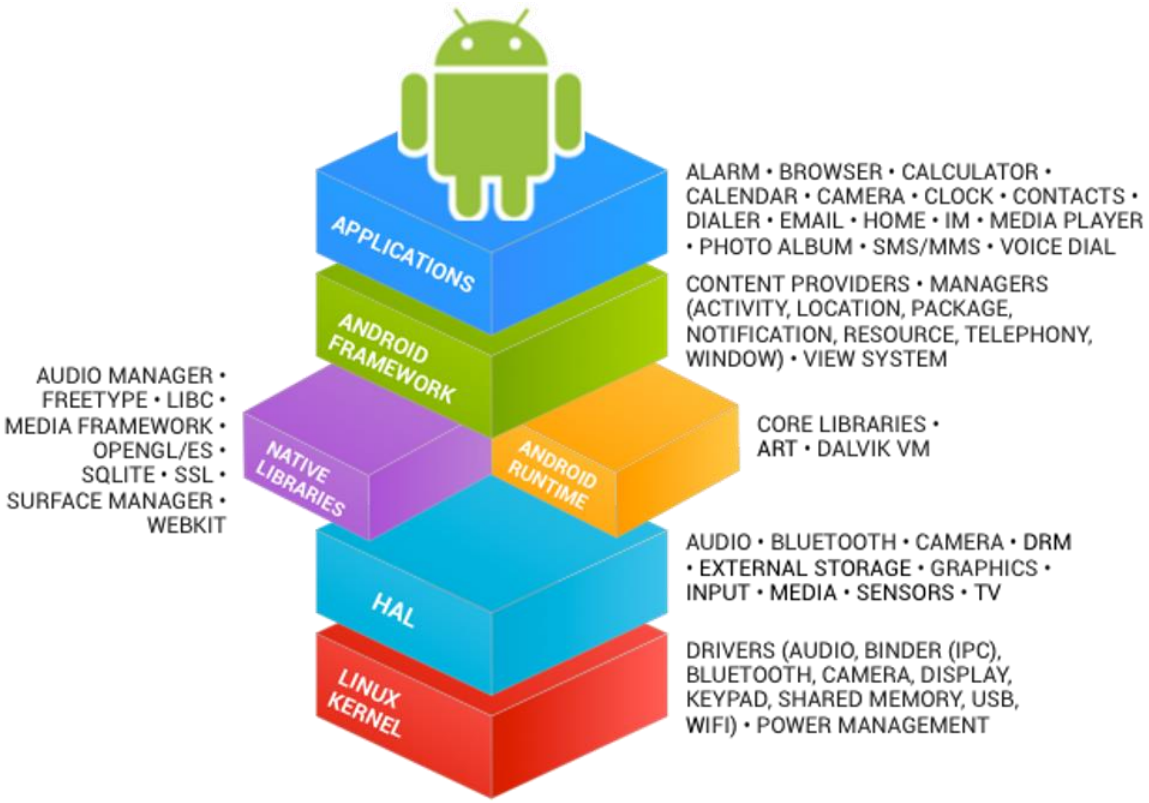
Prema [6] Android operacijski sustav za mobilne uređaje započela je razvijati tvrtka Android Inc. Tvrtka je osnovana 2003. godine, nekoliko godina prije najave prvog iPhone uređaja tvrtke Apple, čak prije nego je riječ "smartphone" postala široko korištena u javnosti. Osnivači tvrtke, Rich Miner, Nick Sears, Chris White i Andy Rubin, željeli su razviti pametnije mobilne uređaje koji bi bolje pratili korisničku lokaciju i sklonosti. Google je kupio Android tvrtku 2005. godine. Razvoj operacijskog sustava nastavili su Rubin i ostali osnivači koji su ostali raditi pod novim vodstvom.

Android operacijski sustav je temeljen na prilagođenoj Linux jezgri i ostalom "open source" softveru. Kao otvorena platforma nije vezan za proizvođača hardvera, a arhitektura mu je modularna i prilagodljiva, što ga čini pogodnim za inovacije i doprinosi njegovoj popularnosti i raširenosti. Prema [7] Android je najprodavaniji operacijski sustav za pametne telefone i tablete i ima više instalacija od svih operacijskih sustava. Operacijski sustav je od svoga otkrivanja 2007. godine prošao kroz mnoge promjene i višestruke inačice softvera, a najnovija verzija je trenutno 11.

Korisničko sučelje je jedno od glavnih obilježja Android sustava. Dizajnirano je za izravno upravljanje putem dodira u obliku stvarnih pokreta poput dodirivanja, tapkanja i sl. Za dodatne korisničke pokrete, poput okretanja uređaja na njegovu stranu, koriste se razni senzori pomoću kojih se sučelje prilagođava trenutnoj situaciji. Druga bitna obilježja su aplikacije koje proširuju funkcionalnosti uređaja, a pisane su najčešće koristeći službeno Android okruženje za razvoj softvera i Java programski jezik te upravljanje pohranom bitno radi produljenja trajanja baterije uređaja.

Prema [8], [9] operacijski sustav Android je napisan u C/C++ programskim jezicima a sastoji se od slojevite arhitekture kao što je vidljivo na slici 3.1. Android jezgra temelji se na modificiranoj Linux jezgri verzije 3.x ili 4.x u kod novijih uređaja. Jezgra upravlja bitnim dijelovima poput memorije, procesa, mrežne komunikacije, a sadrži i upravljačke programe (driver) proizvođača hardvera. Sloj za apstrakciju sklopovlja je sljedeća razina. On sadrži brojne biblioteke koje pružaju

sučelje za odgovarajuće hardverske komponente. Sljedeći slojevi, Android Runtime sloj i nativne C/C++ biblioteke koriste se za pokretanje aplikacija i pružanje funkcionalnosti C/C++ biblioteka istima. Razina aplikacijskog okvira (Application Framework) pruža osnove za razvoj Android aplikacija i omogućava upotrebu aplikacijskog programskog sučelja (API). Posljednji, aplikacijski sloj, vidljiv je korisniku, a sadrži osnovne i ugrađene aplikacije, te aplikacije koje je korisnik preuzeo sa Google Play trgovine.



Sl. 3.1. Arhitektura Android sustava [10]

3.2. Android Studio

Prema [11] Android Studio je službena integrirana razvojna okolina (IDE – *Integrated Development Environment*) za Android operacijski sustav. Nastao je na temelju JetBrains IntelliJ IDEA softvera i posebno je prilagođen izradi Android mobilnih aplikacija. Prva stabilna inačica

je postala dostupna 2014. godine nakon čega je zamijenio Eclipse IDE, općenitiju Java razvojnu okolinu koja se prije koristila i koja podržava različite programske jezike. Neke od značajki Android razvojne okoline su podrška za fleksibilni Gradle build sustav, Lint alati za praćenje performansi, kompatibilnosti i sličnog, čarobnjaci s predlošcima za uobičajene Android dizajne i komponente, alat za obradu sučelja s *drag-and-drop* mogućnostima, podrška Google Cloud platforme i emulator virtualnih uređaja. Java je potrebna za pisanje programa, Android SDK omogućava pokretanje tih programa na Android sustavu, a Android Studio ima ulogu objediniti sve to na jednom mjestu za korisnika.

3.3. Java programski jezik

Prema [12], [13] Java je robustan, objektno orijentiran programski jezik kojeg je stvorio James Gosling, tvrtka Sun Microsystems 1995. godine. Java postoji u 2 paketa: JRE (Java Runtime Environment) koji sadrži biblioteke Java klasa i Java virtualni stroj čime omogućuje pokretanje programa u Javi, te JDK (Java Development Kit) koji još pruža i razvojne alate. Glavna obilježja Java jezika su: jednostavnost, distribuiranost, prenosivost, visoke performanse, sigurnost, višenitnost, ali i jedan od nedostataka – sporost. Java je neovisna o platformi na kojoj se izvodi, jer se programi izvršavaju unutar virtualnog stroja (Java Virtual Machine - JVM), tako da prevoditelj prvo prevede programski kod u *bytecode*, a JVM ga zatim interpretira i izvršava. Java uključuje i automatsko upravljanje memorijom čime je pojednostavljeno stvaranje i uništavanje objekata, te automatsko skupljanje smeća što znači da oslobađa memoriju kada objekti nemaju aktivne pokazivače na memoriju.

4. MOBILNA APLIKACIJA

Ovo poglavlje detaljno opisuje izradu zadane mobilne aplikacije od koncepta pa sve do testiranja. Na kraju poglavlje opisano je testiranje same aplikacije uz prikaz rada pomoću slika sa fizičkog i virtualnog uređaja na kojima je pokrenuta

4.1. Ideja i osnovni koncept

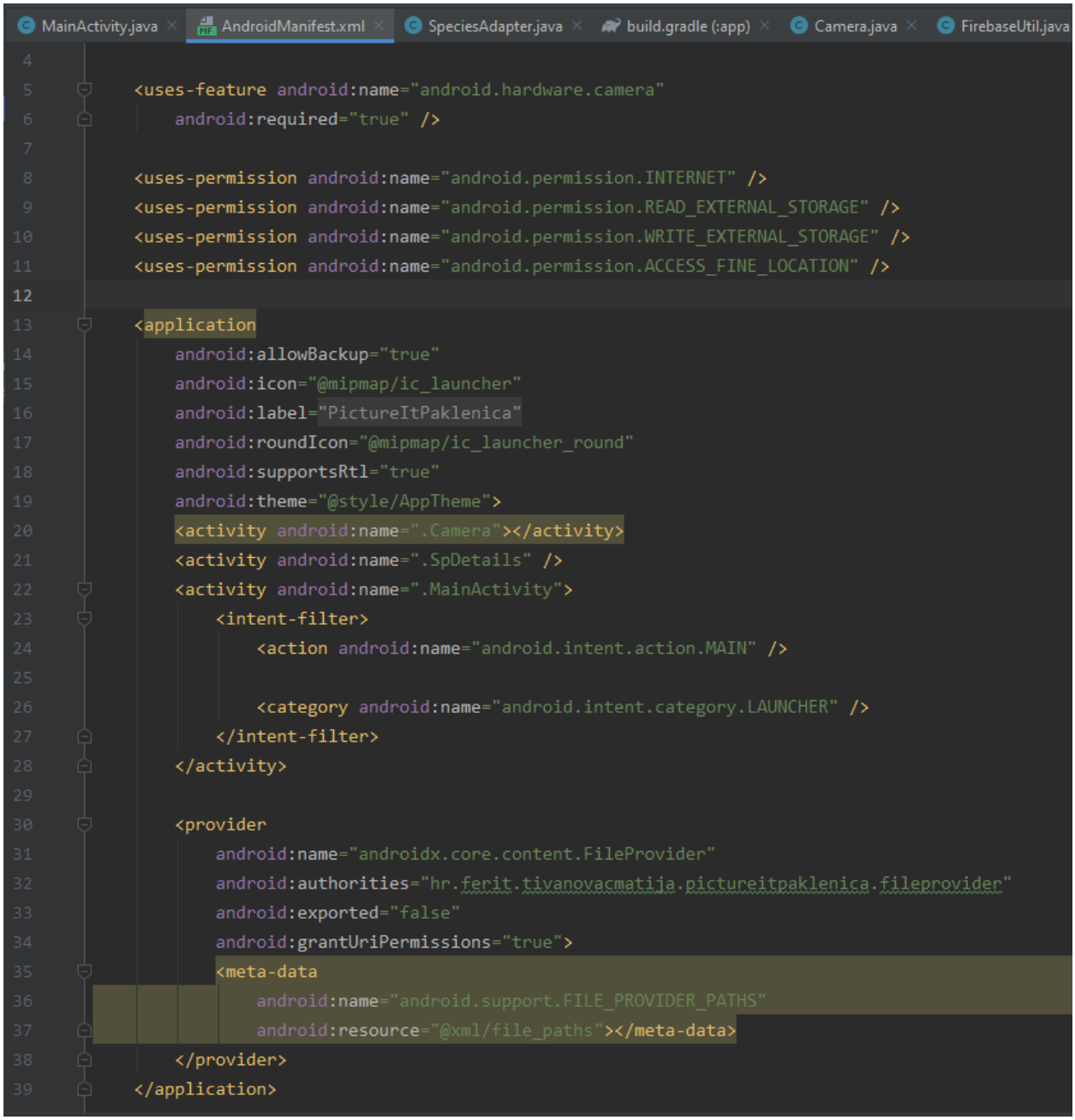
Ideja o izradi Android aplikacije u svrhu promocije i istraživanja za Nacionalni park Paklenicu proizlazi iz činjenice da su mobilni uređaji danas vrlo raširena i dostupna tehnologija i time pružaju mogućnost masovnog marketinga i prikupljanja podataka. Aplikacija će pružiti interaktivno iskustvo posjetiteljima prilikom posjete, a upravi parka mogućnost praćenja biljnih i životinjskih vrsta.

Aplikacija je koncipirana tako da bude što jednostavnija korisnicima za uporabu. Prilikom prvog pokretanja aplikacije od korisnika se traži da se registrira ili prijavi ako se već registrirao. Zatim se na zaslonu prikazuje poruka dobrodošlice, nakon čega korisnik može vidjeti listu sa slikama, nazivima i latinskim nazivima biljnih i životinjskih vrsta koje se nalaze na području parka. Odabirom određene vrste moguće je vidjeti više detalja o njoj, a pritiskom na ikonu fotoaparata korisnik dolazi na zaslon gdje može poslati elektroničku poštu upravi parka koja sadrži ime vrste, vrijeme, sliku i trenutnu lokaciju uređaja u trenutku pritiska na gumb za slanje. S početnog ekrana dugim pritiskom na vrstu iz liste također je moguće pokrenuti slikanje slike pomoću postojeće aplikacije za kameru dostupne na Android sustavima. Aplikacija zatim pita korisnika želi li prihvatiti sliku ili slikati novu. Ukoliko korisnik odabere prvu opciju, slika se, zajedno sa trenutnim vremenom i lokacijom uređaja šalje na predodređenu adresu elektroničke pošte koju je osigurala uprava nacionalnog parka.

4.2. Izrada aplikacije

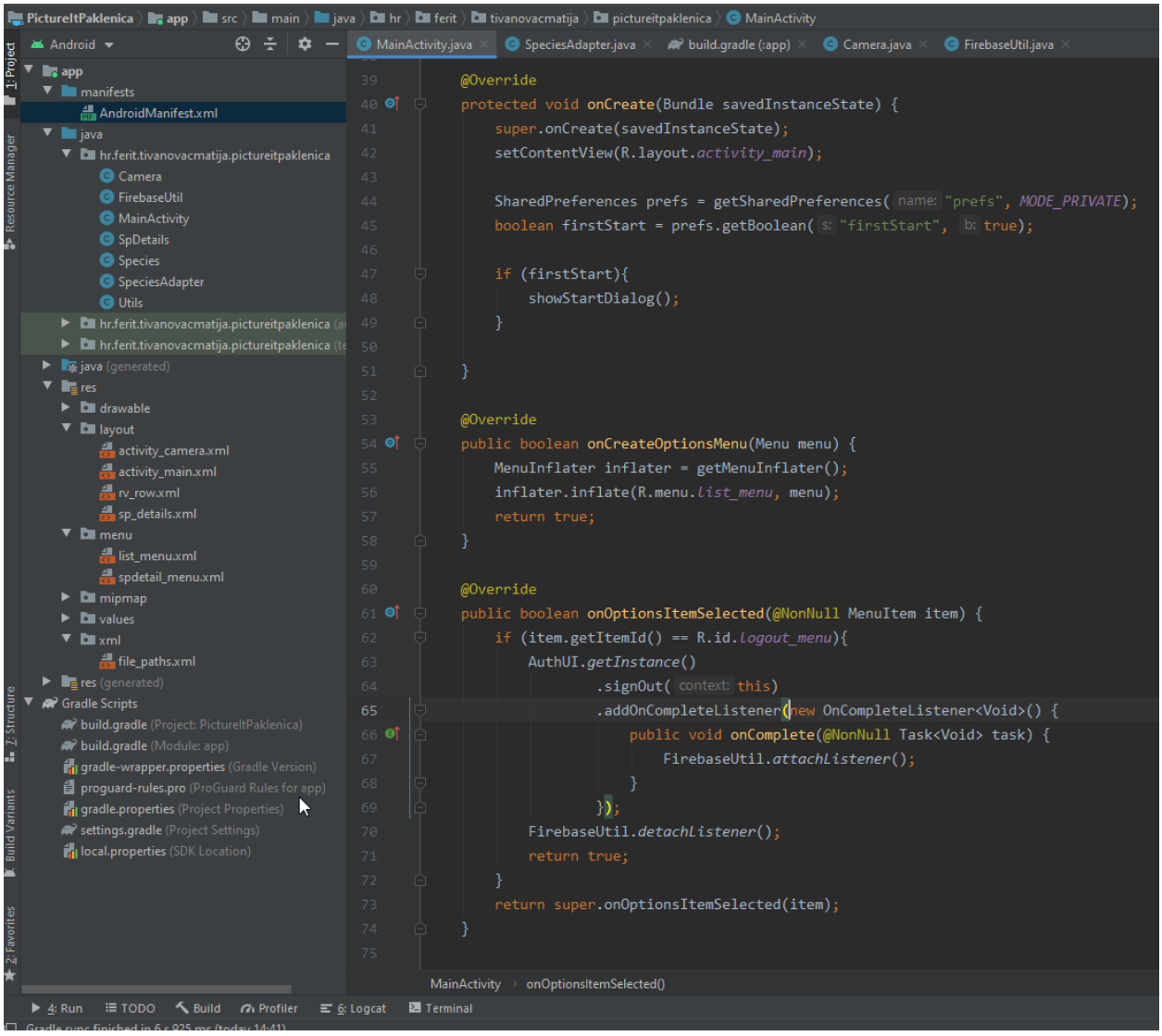
Android aplikacije se sastoje od nekoliko osnovnih komponenta: aktivnost, servis, primatelji broadcast-a i Content provider, također svaka aplikacija ima svoju Manifest datoteku i odgovarajuće Intent komponente koje omogućavaju prelazak s jedne aktivnosti na drugu. Aktivnost predstavlja jedan zaslon aplikacije. Početak izrade aplikacije je stvaranje novog projekta u Android studiju koji sadrži jednu praznu aktivnost. Sve komponente aplikacije prikazane su kao datoteke unutar stvorenog projekta.

Jedna od bitnijih datoteka kod svakog Android projekta je Manifest datoteka. Ona je pisana XML opisnim jezikom i sadrži metapodatke o aplikaciji, definira strukturu aplikacije, aktivnosti, servise, pružatelje sadržaja, dozvole, određuje verziju i ikonu aplikacije i sl. Na slici 4.1. vidljiva je Manifest datoteka projekta sa zatraženim dopuštenjima, aktivnostima i definiranim načinom pohrane podataka.



Sl. 4.1. Manifest datoteka

Sljedeće mape koje sadrže bitne datoteke su java i res. Android sustav razdvaja funkcionalnost od prikaza pa svaka aktivnost ima datoteku klase unutar java mape koja ju predstavlja i sadrži programski kod, te XML datoteku koja definira izgled (layout) aktivnosti koja se nalazi u res mapi. Unutar res mape nalaze se i ostali "resursi" aplikacije poput slika, ikona, tekstualnih vrijednosti (string), boja i sl. Aplikacija se sastoji od sedam java klasa koje definiraju sve funkcionalnosti aplikacije. „MainActivity“ klasa sadrži listu biljnih i životinjskih vrsta. „SpDetails“ klasa prikazuje detalje o vrsti, a „Camera“ klasa omogućava slikanje i slanje emaila s potrebnim podatcima. Ostale četiri klase su tzv. pomoćne klase. „FireBaseUtil“ odrađuje vezu s Firebase bazom podataka, „Species“ klasa omogućava stvaranje objekata za svaku vrstu. „Utils“ klasa pomaže pri pohrani slike prilikom slikanja, a „SpeciesAdapter“ popunjava listu vrsta s biljkama i životinjama. Spomenute datoteke vidljive su na slici 4.2.



Sl. 4.2. Datoteke unutar projekta

4.2.1. Firebase prijava, baza podataka i skladište podataka

Pri pokretanju aplikacije uvijek se traži od korisnika da se logira ili registrira. Zato je potrebno projekt povezati s *Google Firebase* uslugom i sa autentifikacijskim dijelom Firebase usluge. U projekt se prvo dodaje Firebase configuracijska datoteka „*google-services.json*“, zatim se dodaje dodatak (*plugin*) za Google usluge unutar *build.gradle* datoteke i unutar *app/buid.gradle* i zavisnosti (*dependencies*) za željene *Firebase* usluge. U ovom slučaju dodane su zavisnosti za bazu podataka, skladište (*storage*) podataka, analizu i autentifikaciju podataka i korisnika kao što se vidi na slici 4.3.

```
    implementation 'com.google.firebase:firebase-analytics:17.5.0'
    implementation 'com.google.firebase:firebase-database:19.4.0'
    implementation 'com.google.firebase:firebase-storage:19.2.0'
    implementation 'com.firebaseui:firebase-ui-auth:6.2.0'
    implementation 'androidx.cardview:cardview:1.0.0'
    implementation 'androidx.recyclerview:recyclerview:1.1.0'

}

apply plugin: 'com.google.gms.google-services'
```

Sl. 4.3. Firebase zavisnosti

Firebase nudi dva načina autentifikacije: automatski i prilagođeni. Za implementiranje automaskog načina prijave putem *FirebaseUI* potrebno je dodati intent za prijavu kao na slici 4.4. Intent je dodan unutar metode „signIn“ u „FirebaseUtil“ klasi u kojoj se odrađuje cjelokupno upravljanje i povezivanje sa Firebase-om. Spomenuta metoda se poziva ovisno o tome je li korisnik prijavljen ili ne što se provjerava pomoću osluškivača „AuthStateListener“ objekta. Metodu za prijavu pozivamo nakon provjere da korisnik nije prijavljen kao na slici 4.4.

```
mAuthListener = new FirebaseAuth.AuthStateListener() {
    @Override
    public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
        if (firebaseAuth.getCurrentUser() == null) {
            FirebaseUtil.signIn();
        }
    }
};
private static void signIn() {
    // Choose authentication providers
    List<AuthUI.IdpConfig> providers = Arrays.asList(
            new AuthUI.IdpConfig.EmailBuilder().build());

    // Create and launch sign-in intent
    caller.startActivityForResult(
            AuthUI.getInstance() AuthUI
                    .createSignInIntentBuilder() AuthUI.SignInIntentBuilder
                    .setAvailableProviders(providers)
                    .build(),
            RC_SIGN_IN);
}
```

Sl. 4.4. Metoda za prijavu korisnika i njeno pozivanje

Osluškivač se dodaje unutar „*onResume*“ metode u početnoj „MainActivity“ aktivnosti, a uklanja unutar „*onPause*“ metode. FirebaseUI nudi i metode za odjavu korisnika. Odjava putem „*AuthUI.getInstance().signOut*“ odjavljuje korisnika preko izbornika u početnoj aktivnosti što pokazuje slika 4.5.

```java
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if (item.getItemId() == R.id.logout_menu){
        AuthUI.getInstance()
                .signOut( context: this)
                .addOnCompleteListener(new OnCompleteListener<Void>() {
                    public void onComplete(@NonNull Task<Void> task) {
                        FirebaseUtil.attachListener();
                    }
                });
        FirebaseUtil.detachListener();
        return true;
    }
    return super.onOptionsItemSelected(item);
}
```

Sl. 4.5. Odjava korisnika u početnoj aktivnosti

FirebaseUtil klasa sadrži potrebne instance baze podataka i skladišta podataka kao i spajanje sa skladištem. Skladište sadrži samo slike vrsta, dok se u bazi podataka nalaze poveznice na te slike. Samo prijavljeni korisnici mogu pristupiti Firebase podatcima što je određeno pravilima pristupa baze i skladišta vidljivo na slici 4.6.

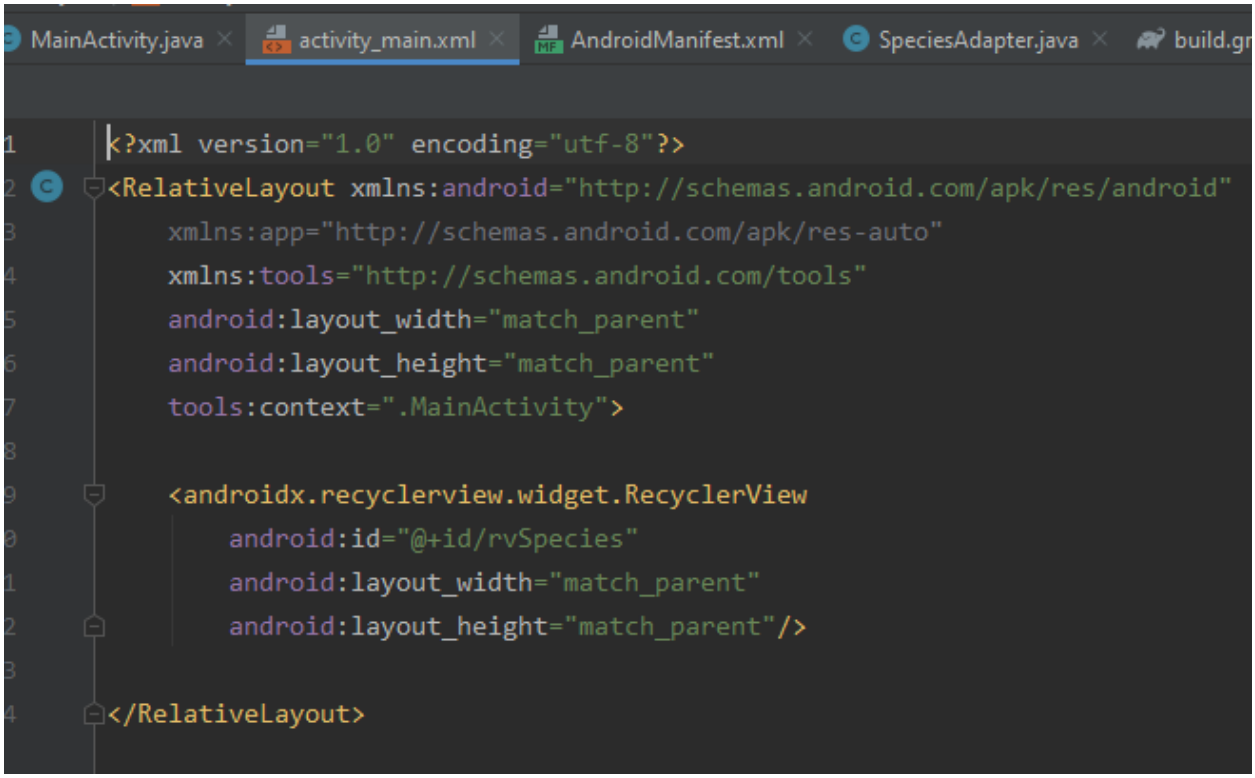
```
{
  "rules": {
    ".read": "auth != null",
    ".write": "auth != null"
  }
}
```

```
rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

Sl. 4.6. Pravila pristupa baze i skladišta

4.2.2. Popis životinja i biljaka

Ova aktivnost sadrži popis biljnih i životinjskih vrsta nabrojanih unutar Natura 2000 popisa. Za prikaz popisa koristi se *RecyclerView* kontrola koja ima naprednije mogućnosti *ListView*-a. Spomenuta kontrola prikazuje listu podataka koja se može pomicati vertikalno i horizontalno. Lista sadrži elemente čiji izgled se može prilagoditi po želji. Temeljna funkcionalnost RecyclerView-a je efikasno upravljanje memorijom. RecyclerView unosi u memoriju samo one elemente liste koji su vidljivi na ekranu, a ostale redom prilikom pomicanja liste. XML datoteka „MainActivity“ klase sadrži samo RecyclerView kontrolu što je vidljivo na slici 4.7.



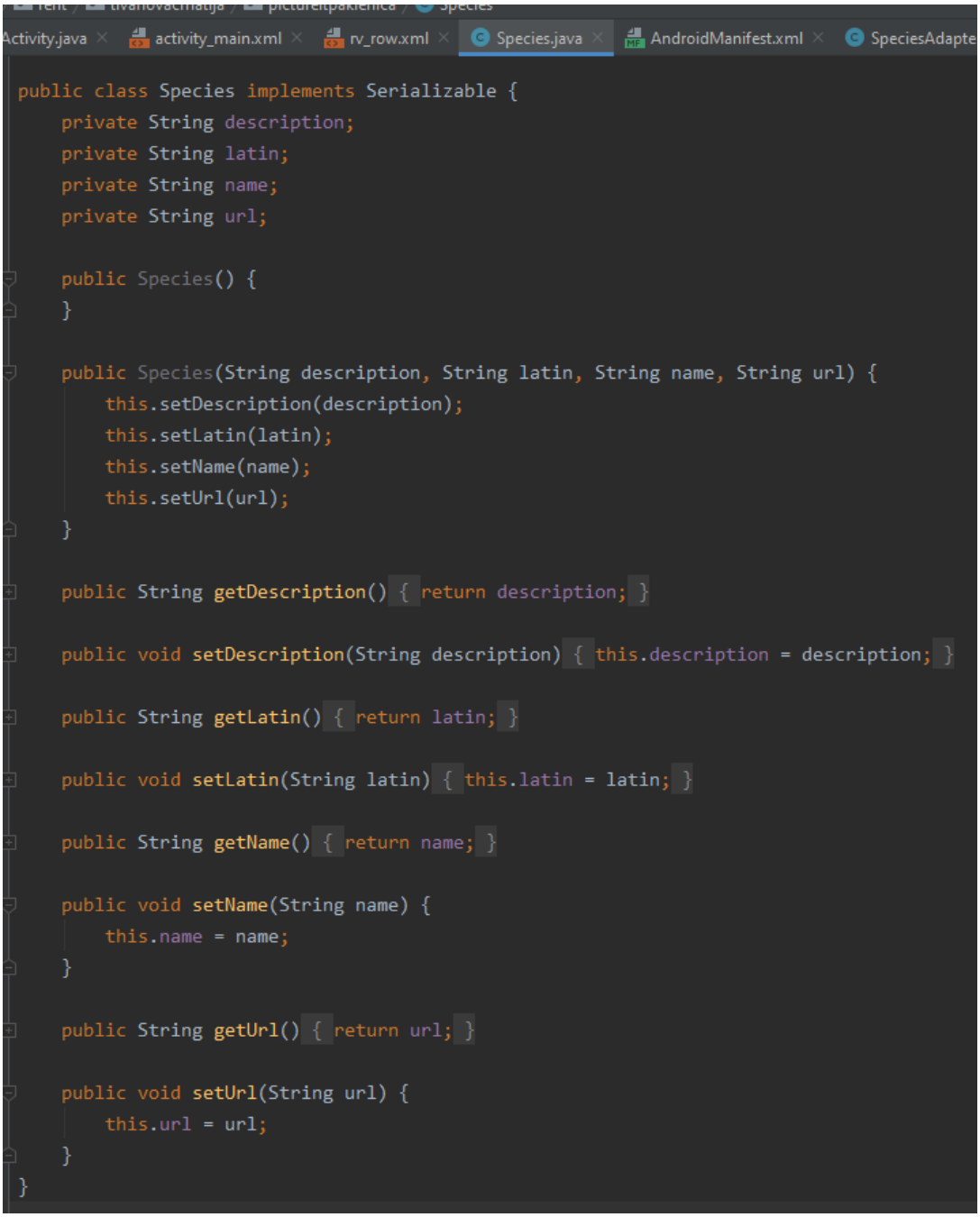
Sl. 4.7. XML datoteka „activity_main“

Zasebna XML datoteka „rv_row.xml“ određuje kako će izgledati pojedini element liste i sastoji se od jednog ImageView-a i dva TextView-a koji prikazuju sliku vrste, njen naziv i latinski naziv. Ove tri komponente nalaze se unutar CardView kontrole koja omogućuje da taj element u listi bude prikazan kao kartica. Kod za spomenuto vidljiv je na slici 4.8.

```
<androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:cardCornerRadius="3dp"
    app:cardElevation="3dp"
    app:cardBackgroundColor="#fff9bf"
    app:cardUseCompatPadding="true">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="5dp">

        <ImageView
            android:id="@+id/ivSpecies"
            android:layout_width="120dp"
            android:layout_height="120dp"
            android:src="@mipmap/ic_launcher"/>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/tvName"
            android:text="TextView"
            android:textSize="22sp"
            android:textColor="@color/colorPrimary"
            android:textStyle="bold"
            android:layout_toRightOf="@id/ivSpecies"
            android:layout_toEndOf="@id/ivSpecies"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="25dp"
            android:layout_marginStart="10dp" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/tvLatin"
            android:textStyle="italic"
            android:text="TextView"
            android:textSize="18sp"
            android:textColor="#7a7a7a"
            android:layout_toRightOf="@id/ivSpecies"
            android:layout_toEndOf="@id/ivSpecies"
            android:layout_marginLeft="10dp"
            android:layout_marginStart="10dp"
            android:layout_marginTop="5dp"
            android:layout_below="@id/tvName" />

    </RelativeLayout>

</androidx.cardview.widget.CardView>
```

Sl. 4.8. Izgled „rv_row.xml“ datoteke

Kako bi elemente povezali u listu potreban je oblikovni obrazac Adapter. Adapter omogućava ponovno korištenje onih *View* objekata (elemenata liste) koji se ne vide na ekranu. Također je potrebno i korištenje oblikovnog obrasca *ViewHolder* koji smanjuje potrebu za pozivanjem metode *findViewById()*, tj. metode za pronalazak elemenata u java klasi. Potrebno je stvoriti vlastitu klasu *Species* koja će predstavljati jednu vrstu s njenim podatcima: nazivom, latinskim nazivom, opisom i poveznicom na sliku u skladištu. Kod za spomenuto prikazan je na slici 4.9.



Sl. 4.9. *Species* klasa predstavlja vrstu

Unutar klase „*SpeciesAdapter*“ se preko *ViewHolder* objekta i LayoutManager-a podatci unose u listu i prikazuju na željeni način. *Bind* metoda prima objekt vrste i povezuje pojedinu View kontrolu sa podatcima o vrsti. Unutar „onChildAdded“ metode koju će okinuti svaki element iz baze kada se Activity pokrene preko *DataSnaphot* instance pridružujemo podatke iz Firebase baze klasi „Species“ i zatim dodajemo tu vrstu u polje Species objekata. U *onCreateViewHolder* dohvaća se „rv_row“ izgled, a u *onCreateViewHolder* dohvaća se vrsta na trenutnoj poziciji i dodaje se holder-u. Metoda *getItemCount* broji Species objekte u polju i vraća njegovu veličinu. Nakon toga je potrebno u *onChildCreated* obavijestiti da je dodan element tako da se metodi *notifyIteminserted* preda pozicija što je zapravo veličina polja minus 1. Ovaj postupak je vidljiv na slici 4.10.

```
        @Override
        public void onChildAdded(@NonNull DataSnapshot snapshot, @Nullable String previousChildName) {
            Species sp = snapshot.getValue(Species.class);
            ssp.add(sp);
            notifyItemInserted( position: ssp.size()-1);
        }
        @Override
        public void onChildChanged(@NonNull DataSnapshot snapshot, @Nullable String previousChildName) {}
        @Override
        public void onChildRemoved(@NonNull DataSnapshot snapshot) {}
        @Override
        public void onChildMoved(@NonNull DataSnapshot snapshot, @Nullable String previousChildName) {}
        @Override
        public void onCancelled(@NonNull DatabaseError error) {}
    };
    mDatabaseReference.addChildEventListener(mChildListener);
}
@NonNull
@Override
public SpeciesViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    Context context = parent.getContext();
    View itemView = LayoutInflater.from(context).inflate(R.layout.rv_row, parent,  attachToRoot: false);
    return new SpeciesViewHolder(itemView);
}
@Override
public void onBindViewHolder(@NonNull SpeciesViewHolder holder, int position) {
    Species species = ssp.get(position);
    holder.bind(species);
}
@Override
public int getItemCount() { return ssp.size(); }
```

Sl. 4.10. Metode u adapter klasi

Za prikaz slika u listi korištena je metoda *showImage* unutar *bind* metode. Za olakšani prikaz i manipulaciju slikama korištena je Picasso vanjska biblioteka što je vidljivo na slici 4.11.

```
private void showImage(String url){
    if (url != null && url.isEmpty() == false){
        Picasso.get() Picasso
                .load(url) RequestCreator
                .resize( targetWidth: 80,  targetHeight: 80)
                .centerCrop()
                .into(ivSpecies);
    }
}
```

Sl. 4.11. Prikaz i obrada slika

Kako bi korisnik mogao pritiskom na vrstu vidjeti njezine detalje i opis potrebno je dodati osluškivač na klik unutar ViewHolder-a i implementirati *onClick* metodu. U toj metodi potrebno je dohvatiti poziciju adaptera, poziciju vrste čiji se detalji žele pogledati i zatim preko intent-a pozvati novi Activity. Naravno u novi Activity potrebno je predati podatke o vrsti pomoću *putExtra* metode kojoj se predaje vrsta što se vidi na slici 4.12. Ovdje je odmah implementiran na isti način i dugi pritisak na vrstu preko kojega se otvara aktivnost za slikanje i slanje emaila.

```
@Override
public void onClick(View view) {
    int position = getAdapterPosition();
    Species selectedSp = ssp.get(position);
    Intent intent = new Intent(view.getContext(), SpDetails.class);
    intent.putExtra( name: "Species", selectedSp);
    view.getContext().startActivity(intent);
}
```

Sl. 4.12. Metoda *onClick*

S druge strane, „sp_details“ aktivnost sadrži uvećanu sliku iz liste ispod koje se prikazan naziv vrste, latinski naziv i opis vrste koja je odabrana u listi. Svi ti elementi se nalaze unutar ScrollView-a koji omogućuje pomicanje ekrana vertikalno ako se ne vide svi elementi na ekranu kao što se vidi na slici 4.13.

```
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
<androidx.constraintlayout.widget.ConstraintLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".SpDetails">
    <ImageView
        android:id="@+id/ivSpecies"
        android:layout_width="311dp"
        android:layout_height="225dp"
        android:layout_marginTop="30dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@mipmap/ic_launcher" />
    <TextView
        android:id="@+id/tvName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="36dp"
        android:text="TextView"
        android:textSize="27sp"
        android:textStyle="bold"
        android:textColor="@color/colorPrimary"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ivSpecies" />
    <TextView
        android:id="@+id/tvLatin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:text="TextView"
        android:textSize="20sp"
        android:textStyle="italic"
        android:textColor="@color/colorPrimary"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/tvName" />
    <TextView
        android:id="@+id/tvDescription"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="20dp"
        android:layout_marginTop="65dp"
        android:layout_marginEnd="20dp"
        android:layout_marginBottom="40dp"
        android:justificationMode="inter_word"
        android:text="TextView"
        android:textSize="20sp"
        android:textColor="@color/colorPrimary"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/tvLatin" />
</androidx.constraintlayout.widget.ConstraintLayout>
</ScrollView>
```

Sl. 4.13. Datoteka „sp_details.xml“

Kako bi se prikazali svi podatci o vrsti u java klasi „SpDetails“ potrebno je primiti *Intent* koje je proslijeđen s liste i primiti vrstu kako bi se mogli prikazati njeni podatci kao na slici 4.14.

```
Intent receivedIntent = getIntent();
Species sp = (Species) receivedIntent.getSerializableExtra( name: "Species");
tvName.setText(sp.getName());
tvLatin.setText(sp.getLatin());
tvDescription.setText(sp.getDescription());
showImage(sp.getUrl());
```

Sl. 4.14. Prihvaćanje podataka u aktivnosti

Na ekranu s podatcima o vrsti nalazi se i izbornik u traci aplikacije u gornjem desnom kutu. Izbornik se sastoji od jedne ikone u obliku fotoaparata, pritiskom na koju se pokreće aktivnost za slikanje i slanje elektroničke pošte. Kod za izbornik vidi se na slici 4.15.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/camera_menu"
        android:icon="@drawable/ic_camera"
        app:showAsAction="ifRoom"
        android:title="Camera"/>

</menu>
```

Sl. 4.15. Izgled izbornika

Prilikom pokretanja aktivnosti za slikanje potrebno joj je proslijediti podatke o vrsti čije je detalje korisnik upravo gledao. Pritisak na izbornik i otvaranje nove aktivnosti implementirano je u *onOptionsItemSelected* metodi kao na slici 4.16.

```java
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if (item.getItemId() == R.id.camera_menu){
        Intent intent = getIntent();
        intent.setComponent(new ComponentName ( pkg: this, Camera.class));
        startActivity(intent);
        return true;
    }
    return super.onOptionsItemSelected(item);

}
```

Sl. 4.16. Otvaranje nove aktivnosti putem izbornika

4.2.3. Slikanje fotografije i lokacija

Posljednji dio aplikacije je aktivnost koja prima podatke ili od aktivnosti s listom vrsta ili od aktivnosti s detaljima vrste, detektira lokaciju uređaja i omogućava slikanje slike i njeno slanje putem email-a. Budući da se pristupa lokaciji i omogućava pohrana slike na uređaj, potrebno je od korisnika zatražiti dopuštenja za navedeno. U Manifest datoteku se unose zahtjevi za pisanje i čitanje vanjske memorije, pristup preciznoj lokaciji (putem GPS-a) i poslužitelju za memoriju, *FileProvider*, što je vidljivo na slici 4.17.

```xml
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<provider
    android:name="androidx.core.content.FileProvider"
    android:authorities="hr.ferit.tivanovacmatija.pictureitpaklenica.fileprovider"
    android:exported="false"
    android:grantUriPermissions="true">
    <meta-data
        android:name="android.support.FILE_PROVIDER_PATHS"
        android:resource="@xml/file_paths">
    </meta-data>
</provider>
```

Sl. 4.17. Dopuštenja za pristup memoriji i lokaciji

Aktivnost se sastoji od jednog ImageView-a i jednog gumba. Pritiskom na ImageView otvara se aplikacija kamere koja je instalirana na uređaj, a pritiskom na gumb otvara se izbornik aplikacija za slanje emaila, nakon čega se otvara i odabrana email aplikacija. Kod aktivnosti vidljiv je na slici 4.18.

```
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Camera">
    <ImageView
        android:id="@+id/ivPicture"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_marginTop="160dp"
        android:src="@drawable/ic_camera_place"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <Button
        android:id="@+id/btnSend"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="64dp"
        android:background="@color/colorPrimary"
        android:padding="15dp"
        android:text="Šalji email"
        android:textColor="#FFF"
        android:textSize="20sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ivPicture" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Sl. 4.18. Izgled aktivnosti za slikanje

Kako bi se pozvala kamera uređaja potrebno je prvo zatražiti dopuštenje za to od korisnika. Zatim je potrebno dodati osluškivač *setOnClickListener* na ImageView. Unutar osluškivača se putem Intenta poziva kamera. Prije implementiranja slikanja, potrebno je napraviti metodu koja će kameri osigurati datoteku u koju će se pohraniti slika. Za tu svrhu koristi se pomoćna „Utils“ klasa koja sadrži *createImageFile* metodu. Metoda stvara ime slike pomoću trenutnog datuma i dohvaća

direktorij, mapu za pohranu. Na kraju vraća datoteku sa ekstenzijom .jpg. Radi boljeg korisničkog iskustva, u „Utils“ klasi je potrebno napraviti još jednu metodu imena *notifyGalleryAboutPic*. Ova metoda će pomoću Intenta potražiti novu datoteku i obavijestiti aplikaciju za pregled slika, „Gallery“, da je dodana nova slika što je vidljivo na slici 4.19.

```
public class Utils {
    public static void notifyGalleryAboutPic (Context context, String imagePath){

        Intent intent = new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE);
        File file = new File(imagePath);
        Uri contentUri = Uri.fromFile(file);
        intent.setData(contentUri);
        context.sendBroadcast(intent);
    }

    public static File createImageFile() throws IOException{

        String timeStamp = new SimpleDateFormat( pattern: "yyyyMMdd_HHmmss", Locale.ENGLISH).format(new Date());
        String imageFileName = "IMG_" + timeStamp;

        File storageDir = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES);
        File imageFile = new File(storageDir, child: imageFileName + ".jpg");
        return imageFile;
    }
}
```

Sl. 4.19. Klasa „Utils“

Dakle, u klasi „Camera“ poziva se kamera uređaja putem Intenta i stvara se datoteka za sliku putem *createImageFile* metode. Ukoliko je stvorena datoteka, dohvaća se njena putanja za pohranu te se u nju pohranjuje nova slika što je vidljivo na slici 4.20.

```
ivPicture.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        if (takePictureIntent.resolveActivity(getPackageManager()) != null){

            File photoFile = null;
            try {
                photoFile = Utils.createImageFile();
            } catch (IOException e) {
                e.printStackTrace();
            }

            if (photoFile != null){
                mPhotoPath = photoFile.getAbsolutePath();
                mPhotoURI = FileProvider.getUriForFile( context: Camera.this,  authority: "hr.ferit.tivanovacmatija.pictureitpaklenica.fileprovider", photoFile);
                takePictureIntent.putExtra(MediaStore.EXTRA_OUTPUT, mPhotoURI);
                startActivityForResult(takePictureIntent,  requestCode: 21);
```

Sl. 4.20. Slikanje i pohrana slike

Zatim se poziva nova metoda *onActivityResult* koja dohvaća umanjenu sliku koju kamera vraća nakon slikanja. Tu sliku postavlja u ImageView i poziva metodu za obavještavanje galerije da je stvorena nova slika koja će se sada nalaziti i u galeriji uređaja što se vidi na slici 4.21.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    if (requestCode == 21 && resultCode == RESULT_OK) {
        Bitmap bitmap = BitmapFactory.decodeFile(mPhotoPath);
        getLocation();
        ivPicture.setImageBitmap(bitmap);
        btnSend.setEnabled(true);
        Toast.makeText( context: Camera.this,  text: "Image saved", Toast.LENGTH_SHORT).show();
        Utils.notifyGalleryAboutPic( context: Camera.this, mPhotoPath);
    }
    super.onActivityResult(requestCode, resultCode, data);

}
```

Sl. 4.21. Postavljanje umanjene slike i obavještavanje galerije

Lokacija uređaja se dohvaća prilikom pokretanja kamere i tada se od korisnika i traži dopuštenje za lokaciju. Ukoliko korisnik dopusti poziva se metoda *getLocation*. Ona koristi

fusedLocationProviderClient i *Geocoder* kako bi od uređaja dobila lokaciju i iz nje izvukla zemljopisnu širinu i dužinu i pohranila ih što pokazuje slika 4.22.

```
private void getLocation() {
    fusedLocationProviderClient.getLastLocation().addOnCompleteListener((task) -> {
            Location location = task.getResult();
            if (location != null){
                try {
                    Geocoder geocoder = new Geocoder( context: Camera.this, Locale.getDefault());
                    List<Address> addresses = geocoder.getFromLocation(location.getLatitude(), location.getLongitude(),  maxResults: 1);
                    lat = addresses.get(0).getLatitude();
                    longitude = addresses.get(0).getLongitude();
                }catch (IOException e){
                    e.printStackTrace();
                }
            }
    });
}
```

Sl. 4.22. Dohvaćanje zemljopisne širine i dužine

Nakon slikanja, dohvaćanja slike i lokacije potrebno je iz primljenog Intenta od liste ili aktivnosti s detaljima vrste dohvatiti podatke o vrsti što se radi na isti način opisan u klasi „SpDetails“. Također je potrebno dohvatiti podatke o prijavljenom korisniku pomoću *getCurrentUser* metode i poslati sve te podatke putem emaila. Slanje emaila obavlja se pritiskom na gumb „ŠALJI EMAIL“ preko Intenta pomoću „ACTION_SEND“. Potrebno je postaviti tip poruke i njezin sadržaj. Preko Intenta se postavlja odabrana adresa primatelja, naslov poruke, sadržaj poruke sa svim podatcima o vrsti, korisniku i lokaciji i na kraju se dodaje slika kao prilog što je vidljivo na slici 4.23. Zatim se pokreće izbornik aplikacija koje će poslati poruku. Android sustav nudi sve aplikacije koje podržavaju ovakav tip poruke. Potrebno je odabrati email aplikaciju po želji nakon čega se ona i otvara sa svim podatcima već unesenim. Korisniku je preostalo još samo poslati poruku.

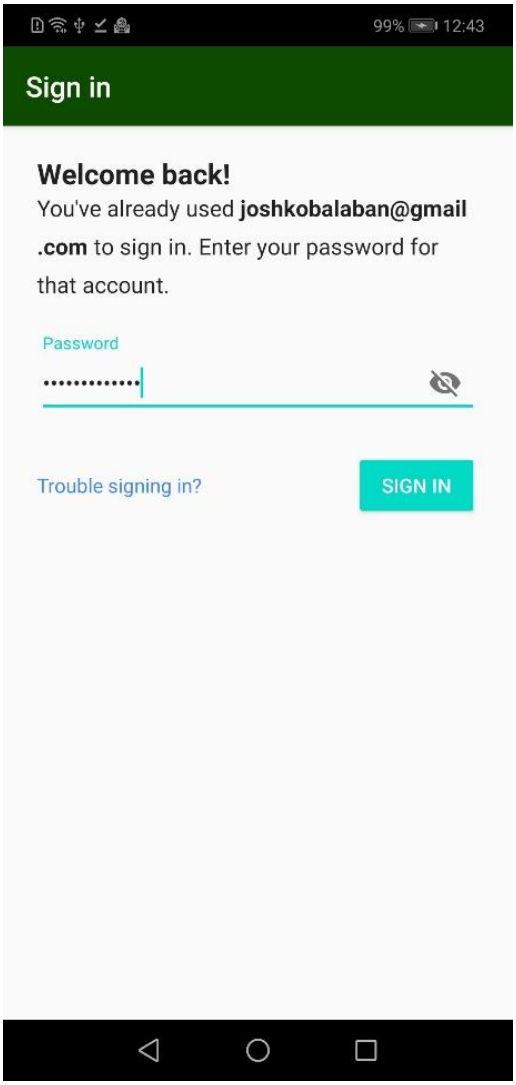
```
btnSend.setOnClickListener((view) -> {
        FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
        if (user != null) {
            userName = user.getDisplayName();
            userEmail = user.getEmail();
        }
        if (btnSend.isEnabled()) {
            Intent i = new Intent(Intent.ACTION_SEND);
            i.setType("message/rfc822");
            i.putExtra(Intent.EXTRA_EMAIL, new String[]{"paklenicapark@gmail.com"});
            i.putExtra(Intent.EXTRA_SUBJECT,  value: "PictureItPaklenica image");
            i.putExtra(Intent.EXTRA_TEXT,  value: userName + "\n" + userEmail + "\n\n" + spName + "\n\n" + lat + ", " + longitude);
            i.putExtra(Intent.EXTRA_STREAM, mPhotoURI);
            try {
                startActivity(Intent.createChooser(i,  title: "Send mail..."));
            } catch (android.content.ActivityNotFoundException ex) {
                Toast.makeText( context: Camera.this,  text: "There are no email clients installed.", Toast.LENGTH_SHORT).show();
            }
        }
```

Sl. 4.23. Slanje emaila

5. TESTIRANJE APLIKACIJE

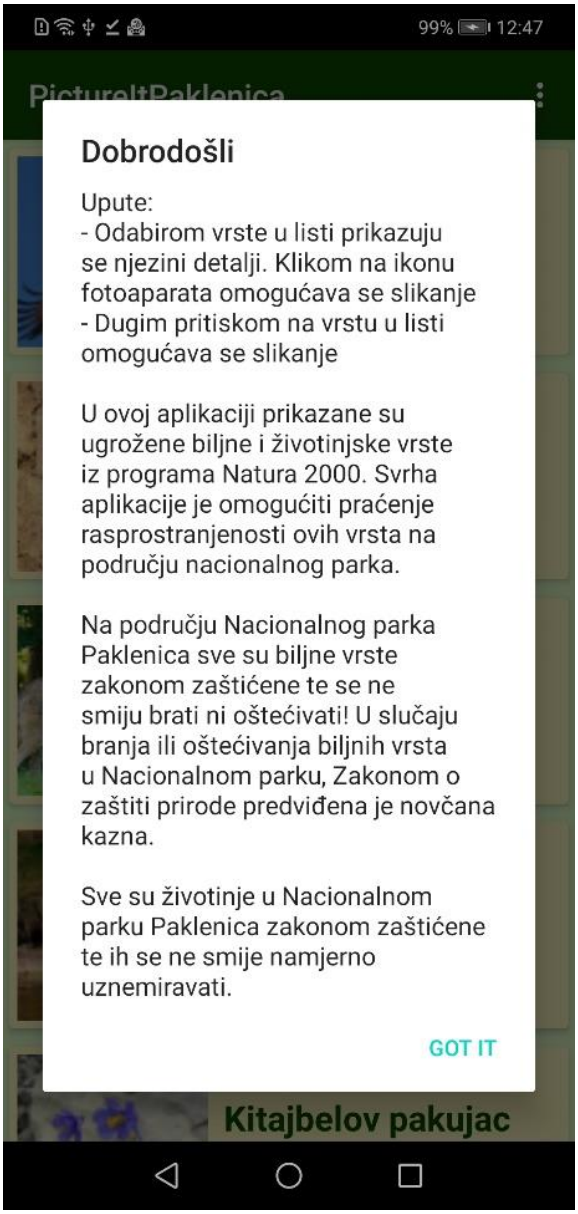
Aplikaciju je nakon izrade potrebno testirati kako bi se provjerila ispravnost i funkcionalnost cjelokupne aplikacije. Aplikacija je testirana na stvarnom uređaju Huawei P20 Lite sa rezolucijom ekrana 2280 x 1080 piksela i gustoćom piksela 432 ppi (*pixels per inch*). Koristi inačicu 9 Android sustava, API 28.

Prilikom svakog pokretanja, aplikacija provjerava je li korisnik prijavljen. Ukoliko nije, traži unos adrese elektroničke pošte. Ako je korisnik registriran slijedi unos lozinke za dovršetak prijave kao što je vidljivo na slici 5.1. Ako korisnik nije registriran, nakon unosa adrese elektroničke pošte, mora napisati svoje ime i prezime i postaviti lozinku.



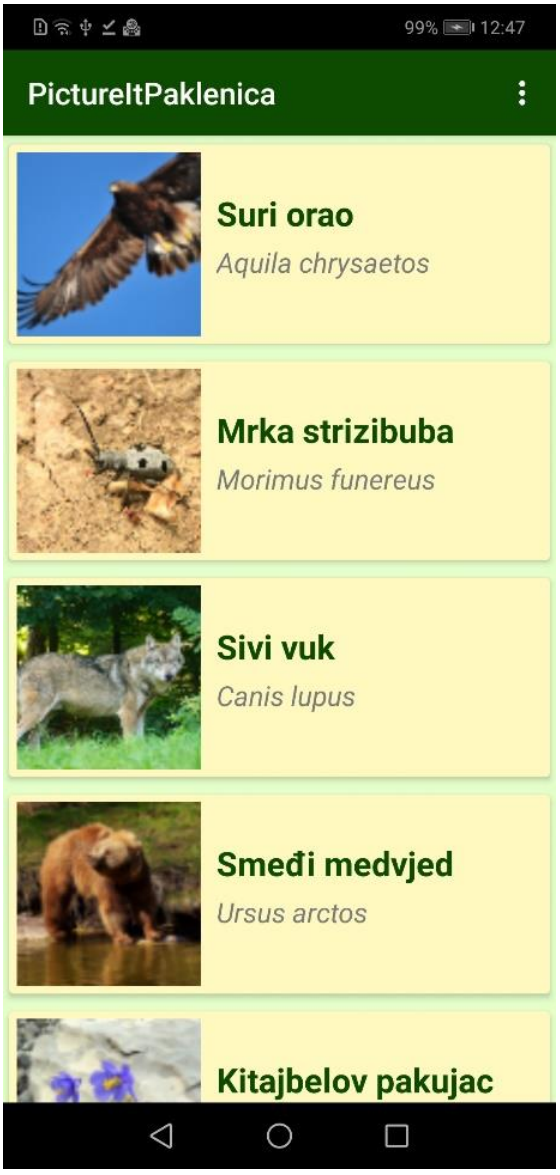
Sl. 5.1. Prijava registriranog korisnika

Ako je aplikacija pokrenuta prvi puta nakon instalacije, prikazuje se *Alert Dialog*, odnosno, prozor s porukom koja sadrži kratke upute i upozorenje o uznemiravanju životinja i biljaka u parku. Nakon toga, poruka se više neće pojavljivati, osim ako se aplikacija obriše i ponovno instalira. Poruka je vidljiva na slici 5.2.



Sl. 5.2. Poruka dobrodošlice

Ako je korisnik prijavljen i vidio je početnu poruku, svaki puta kada pokrene aplikaciju prvo će se prikazati popis životinja i biljaka vidljiv na slici 5.3.. Ovdje korisnik ima pregled svih vrsta koje uprava parka želi prikazati. Popis se može micati gore i dolje i prihvaća dugi ili kratki pritisak na pojedini element. U gornjem desnom kutu nalazi se izbornik preko kojega se korisnik može odjaviti iz aplikacije.

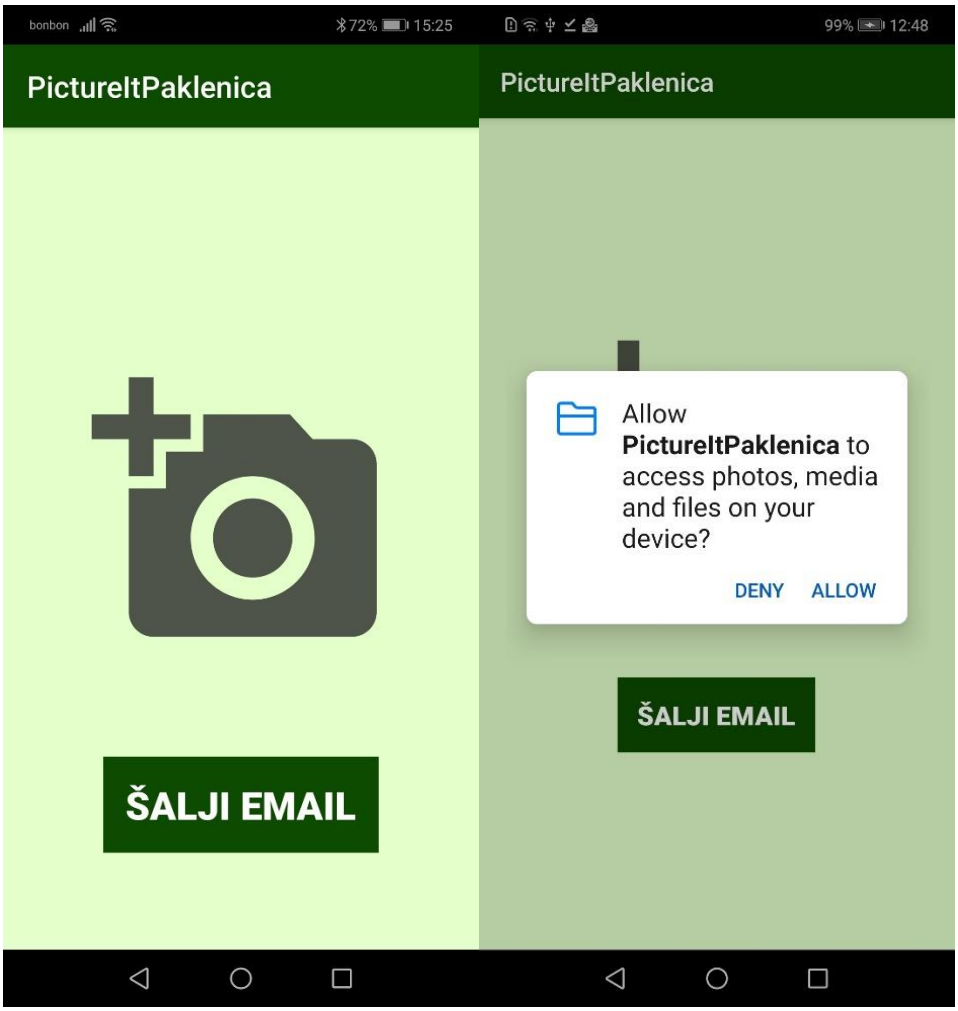


Sl. 5.3. Popis životinja i biljaka

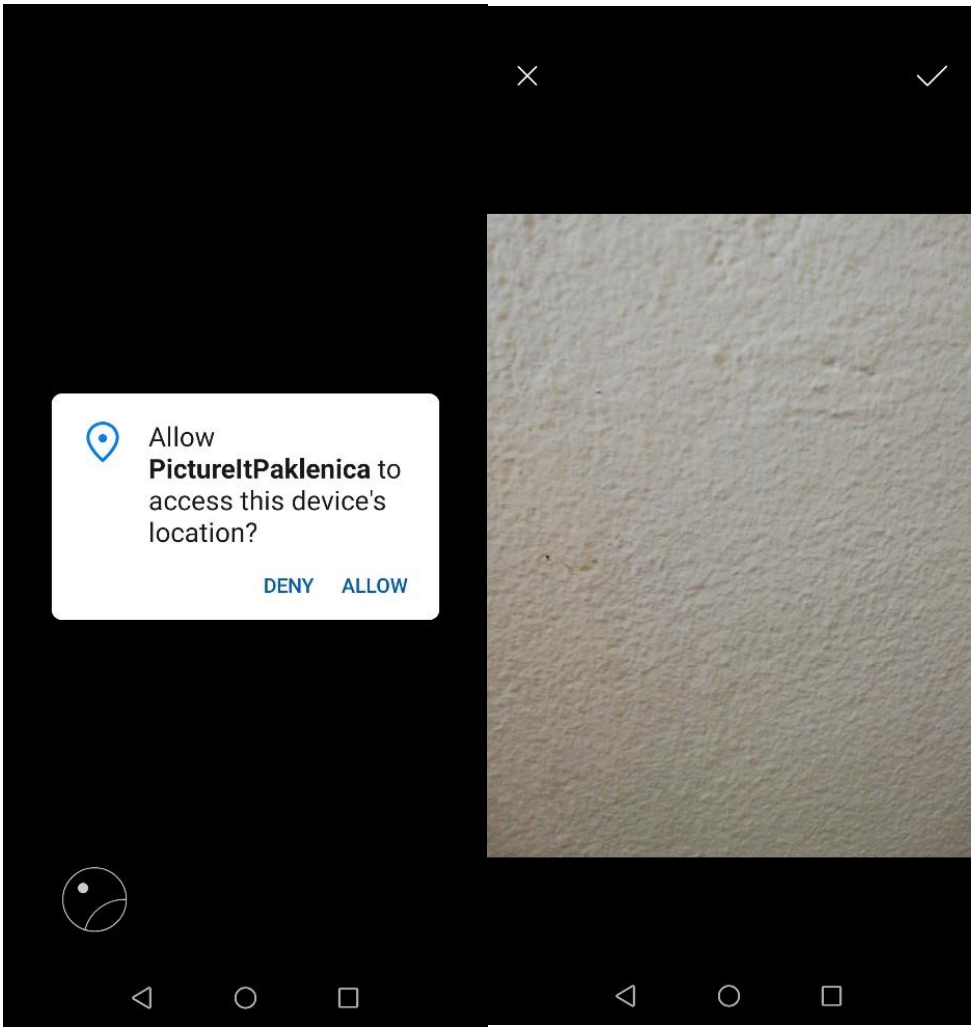
Kratkim pritiskom na element (vrstu) iz liste prikazuju se detaljniji podatci o njoj što je vidljivo na slici 5.4. Pojavljuje se uvećana slika iz liste sa hrvatskim i latinskim nazivom, kao i kratki tekst koji detaljnije opisuje vrstu ili govori o nekoj zanimljivosti specifičnoj za nju. Također, u gornjem desnom kutu, nalazi se ikona fotoaparata koja korisnika vodi do posljednje funkcionalnosti aplikacije.

Sl. 5.4. Detalji o vrsti

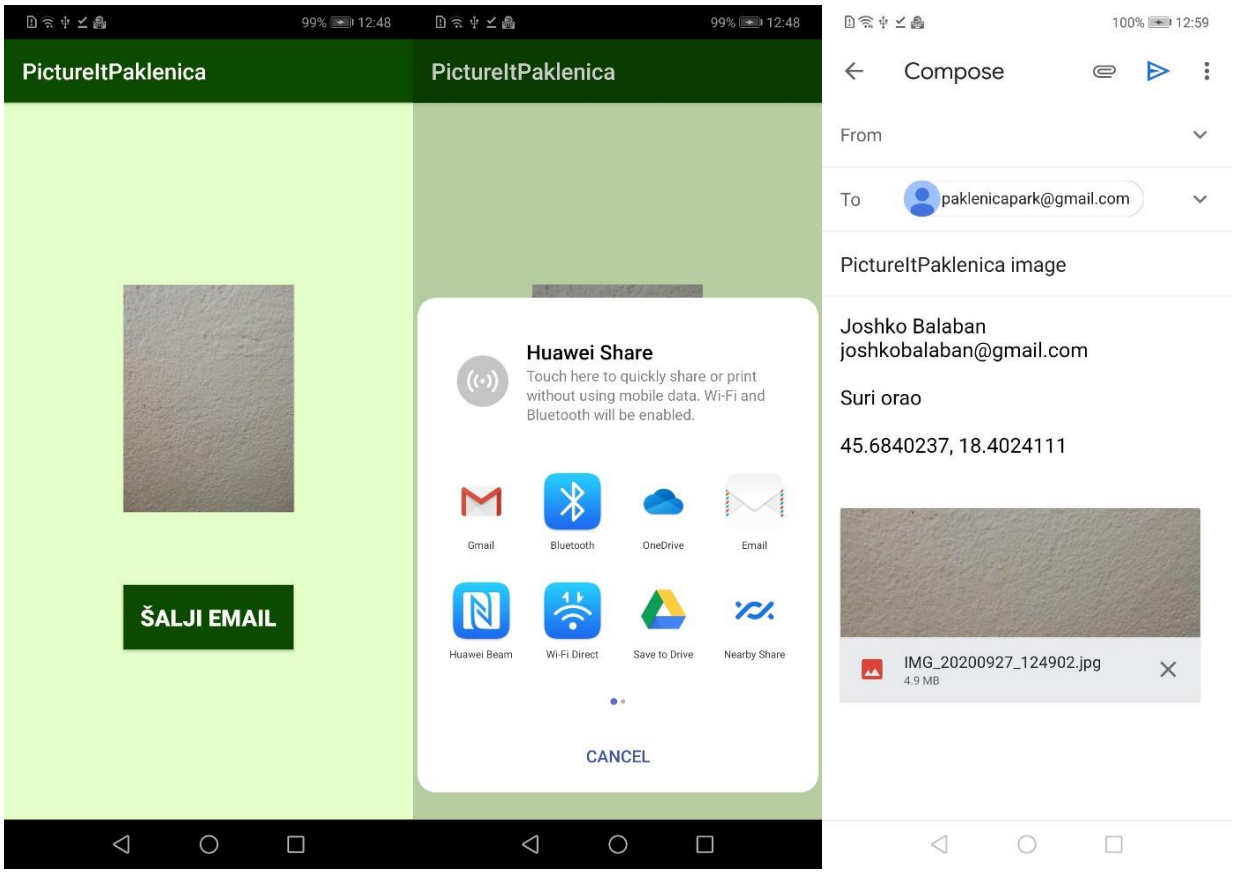
Posljednji dio aplikacije omogućuje korisniku slikanje i slanje elektroničke pošte. Na taj dio je moguće doći pritiskom na spomenutu ikonu fotoaparata ili dugim pritiskom na vrstu iz popisa. Ovdje se nalaze ikona za slikanje i gumb za slanje elektroničke pošte koji je prije slikanja onemogućen što je vidljivo na slici 5.5. Prvo što korisnik mora odobriti aplikaciji je pristup fotografijama i datotekama na uređaju što je prikazano na slici 5.5. Zatim korisnik može pokrenuti kameru. Nakon što korisnik aplikaciji dopusti i pristup lokaciji uređaja, moguće je slikati sliku. Sliku je potrebno potvrditi ili odbaciti kao što je vidljivo na slici 5.6. Potvrđivanjem slike ona se prikazuje u umanjenoj verziji na ekranu i korisniku je omogućeno slanje elektroničke pošte pritiskom na gumb i odabirom aplikacije za slanje što je vidljivo na slici 5.7. Nakon što se odabrana aplikacija otvori, nova poruka je odmah popunjena sa svim informacijama poput: korisničkog imena i prezimena, adrese elektroničke pošte, nazivom biljke ili životinje i lokacijom uređaja što je vidljivo na slici 5.7. Korisnik na kraju samo mora poslati elektroničku poštu i ona će doći na već određenu adresu uprave parka.



Sl. 5.5. Prikaz ekrana prije slikanja

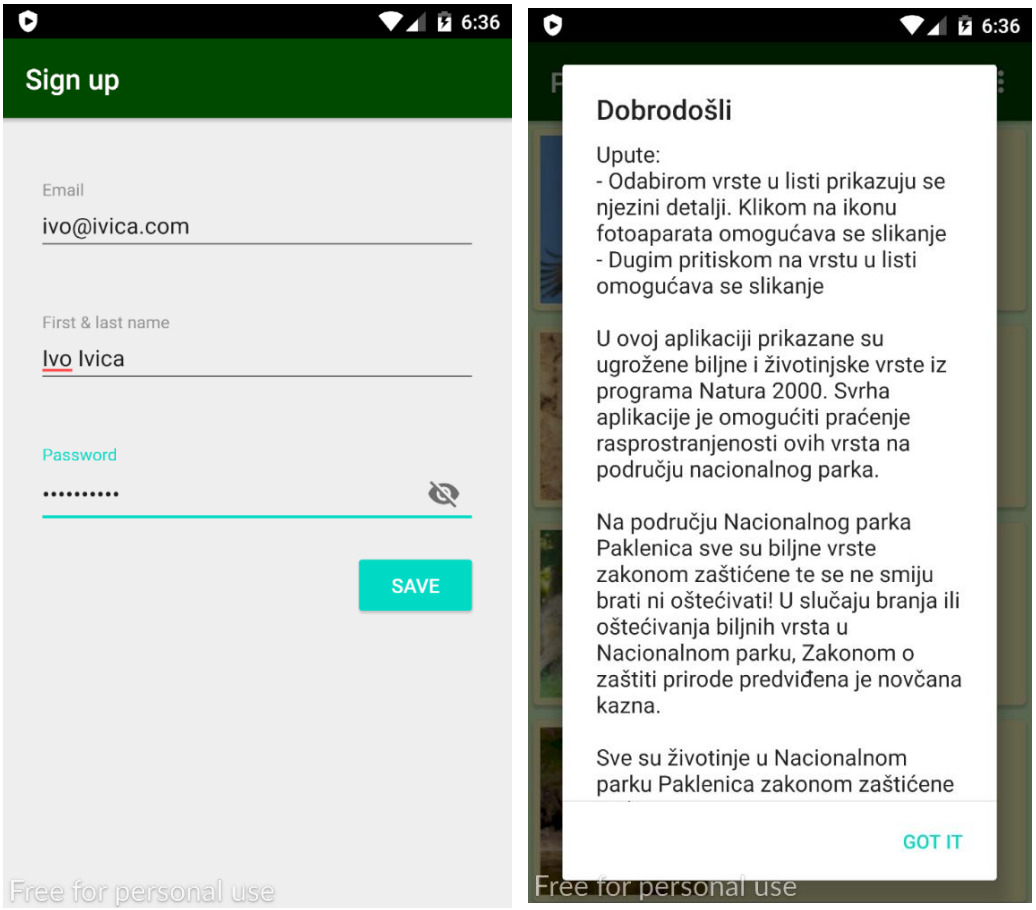


Sl. 5.6. Slikanje i odabir slike

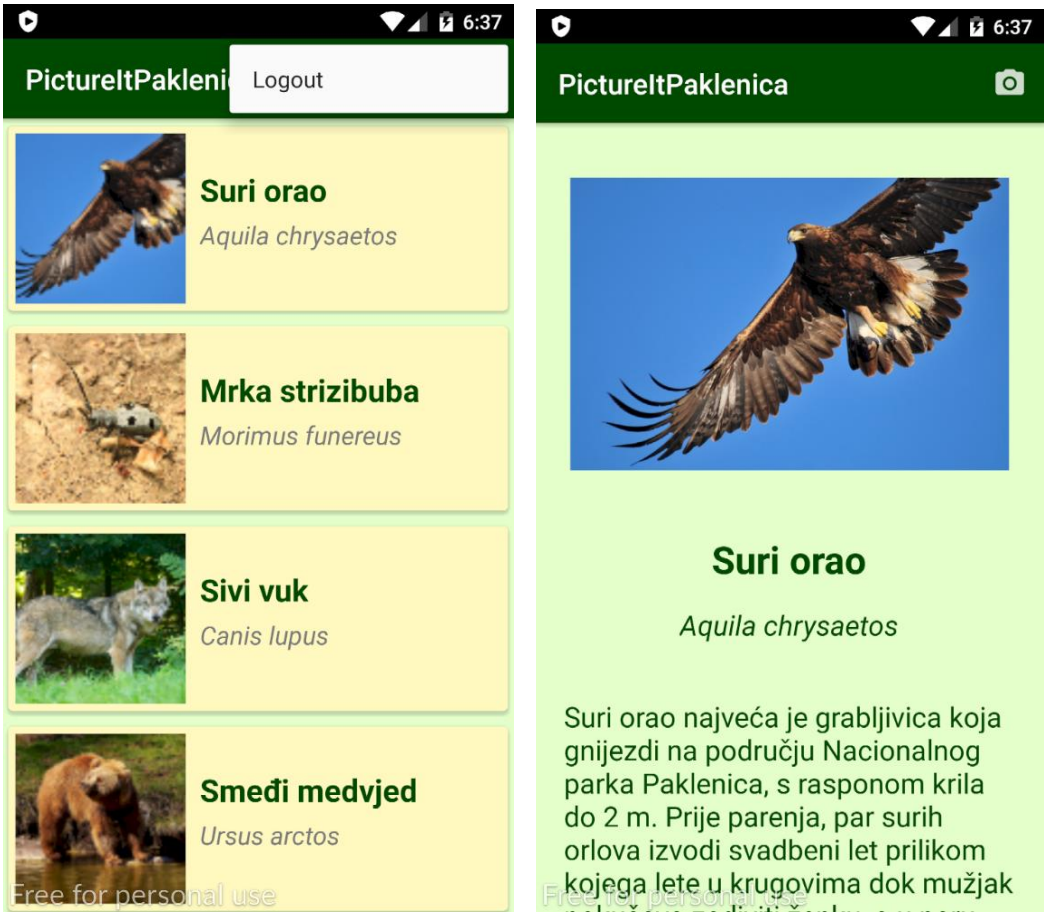


Sl. 5.7. Slanje elektroničke pošte nakon slikanja

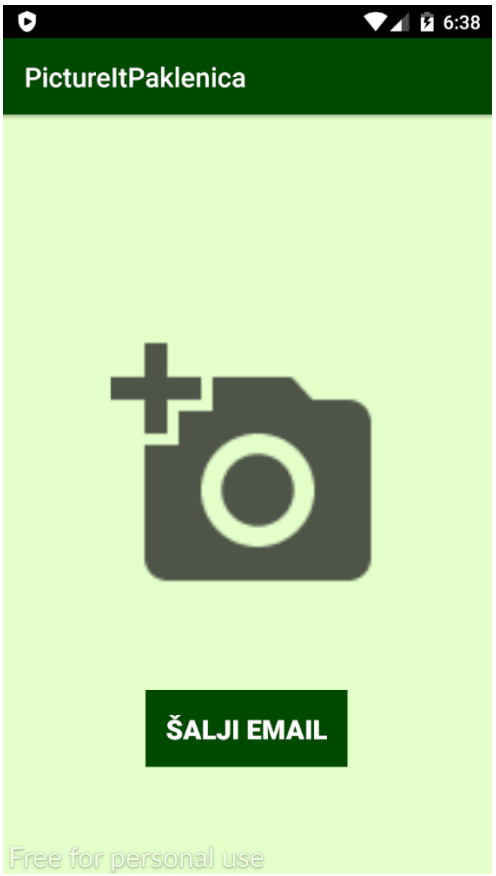
Testiranje se provodi i na virtualnom mobilnom uređaju Samsung Galaxy S6 putem AVD (*Android Virtual Device*) alata Android Studio razvojne okoline. Virtualni uređaj koristi 5.1 verziju Android operacijskog sustava tj. API 22. Ekran uređaja ima rezoluciju 1440 x 2560 sa gustoćom piksela 640 ppi. Aplikacija je pokrenuta uspješno i potpuno je funkcionalna što pokazuju slike 5.8., 5.9. i 5.10.



Sl. 5.8. Testiranje aplikacije na virtualnom mobilnom uređaju



Sl. 5.9. Testiranje aplikacije na virtualnom mobilnom uređaju



Sl. 5.10. Testiranje aplikacije na virtualnom mobilnom uređaju

6. MOBILNI MARKETING I PROMOCIJA PUTEM APLIKACIJE

Prema [14] marketing je aktivnost, set instrukcija i procesi stvaranja, komuniciranja, pružanja i razmjene ponuda koje imaju neku vrijednost za klijente, partnere, kupce i društvo u cijelosti. U svojoj osnovi marketing pokušava spojiti proizvod i usluge neke kompanije s kupcima koji žele pristup tim proizvodima. Kao rezultat toga povećava se profitabilnost tvrtke. Dakle glavni cilj marketinga je dobiti pozornost potencijalne publike i održati taj odnos s njom. Proizvod i promocija su temelji marketinškog miksa. Proizvod bi u slučaju nacionalnog parka bila usluga ulaza u park i obilazak parka. Promocija se odnosi na pitanja kada i gdje prenijeti marketinšku poruku ciljanoj publici. Budući da je uporaba mobilnih uređaja u današnje vrijeme jako raširena, mobilni marketing igra veliku ulogu u promociji i pristupu klijentima. Korisnici mobitela gotovo uvijek imaju svoj uređaj sa sobom, što znači da se promocija može odvijati svugdje i uvijek.

Prema [15] mobilni marketing se može definirati kao korištenje mobilnog medija kao komunikacijskog i zabavnog kanala između pružatelja usluga/proizvoda i krajnjeg korisnika. Ono je jedino sredstvo koje omogućava spontanu, direktnu, interaktivnu i ciljanu komunikaciju bilo kada i bilo gdje. Osobe koje se bave mobilnim marketingom ne ciljaju publiku prema demografiji već prema ponašanju i na odgovarajući način prilagođavaju marketinšku strategiju. Mobilni marketing je postao sastavni dio integrirane marketinške komunikacije koja nastoji objediniti sve metode i strategije marketinga kako bi one surađivale zajedno i uklonile nedostatke koje svaka ima pojedinačno.

Prema [16] integrirana marketinška komunikacija je koncept koji osigurava da su svi oblici komunikacije i poruka pažljivo povezani i konzistentni. To podrazumijeva horizontalnu integraciju gdje bi sve poslovne funkcije (produkcijske, financijske, distribucijske…) trebale surađivati. Integracija podataka i unutarnja integracija označavaju objedinjavanje svih podataka od različitih odjela i informiranje i motiviranje svih zaposlenika. Vanjska integracija, s druge strane, podrazumijeva uključivanje vanjskih agencija. Razvojem tehnologije integrirana marketinška komunikacija dobiva novu sferu koja zamjenjuje prijašnju jednosmjernu komunikaciju sa mogućnošću interaktivne komunikacije sa klijentima.

Zbog svoje interaktivnosti i dostupnosti mobilna aplikacija predstavlja vrlo važan alat u promocijskom miksu. U sklopu nacionalnog parka mobilna aplikacija može unaprijediti postojeću uslugu (obilazak parka) tako da ju učini zanimljivijom i pruži joj dodatnu svrhu i cilj. Uprava parka ima mogućnost putem aplikacije organizirati natjecanja i ponuditi posjetiteljima nagrade za razne

aktivnosti, primjerice nagraditi osobu s najviše poslanih važećih slika biljnih ili životinjskih vrsta ili nagraditi osobu koja je poslala stotu sliku. U aplikaciju je moguće implementirati razne oglase ili zatražiti plaćanje aplikacije kako bi se mogla skinuti na uređaj. Ovakva mobilna aplikacija omogućila bi upravi nacionalnog parka interaktivnu komunikaciju s gotovo svim posjetiteljima neovisno o demografskim obilježjima.

7. ZAKLJUČAK

U današnje vrijeme visoke tehnologije vrlo je malo osoba koje ne posjeduju mobilni uređaj. Oni koji ga posjeduju provode veliki dio njihove svakodnevice koristeći ga. Prema tome, mobilni uređaji su jako rašireni i potrebno ih je integrirati u poslovne, istraživačke i ostale strategije. Cilj ovog rada bio je izraditi mobilnu aplikaciju za najrašireniji i korišteniji operacijski sustav, Android. Aplikacija podržava više inačica Android sustava što bi trebalo većini posjetitelja parka omogućiti njeno korištenje.

Aplikacija je namijenjena posjetiteljima Nacionalnog parka Paklenica gdje će se koristiti u svrhe promocije i praćenja ugroženih vrsta na području parka. Funkcionalnosti koje aplikacija pruža će uz spomenuto, omogućiti i uporabu aplikacije u svrhe poboljšanja kvalitete doživljaja parka prilikom posjete, ali i edukaciju korisnika o potrebama zaštite i očuvanja ugroženih biljnih i životinjskih vrsta.

Aplikacija koristi udaljenu bazu podatak što znači da ju je u određenoj mjeri moguće izmijeniti i dodavati novi sadržaj bez programskog znanja i pisanja koda. Također aplikacija ima velik potencijal za dodatna unaprjeđenja poput implementiranja sustava pretraživanja vrsta kako bi se brže pronašla željena životinja ili biljka. Moguće je uvesti i lokalizaciju, kako bi aplikacija bila razumljiva na više jezika i pokrila veću publiku. U aplikaciju bi se potencijalno mogao ugraditi sustav umjetne inteligencije koji bi prilikom slikanja prepoznavao o kojoj biljnoj ili životinjskoj vrsti se radi čime bi se poboljšao istraživački i znanstveni aspekt aplikacije. U svrhu poboljšanja promocijske vrijednosti aplikacije moguće je omogućiti sustav nagrađivanja korisnika za razne aktivnosti i pravovremenog obavještavanja korisnika o dobivenoj nagradi ili novim informacijama distribuiranim putem mobilne aplikacije. U usporedbi sa većinom ostalih postojećih aplikacija vezanih uz nacionalne parkove, jedina osim informativnog sadržaja pruža korisnicima interaktivan sadržaj i mogućnosti da službenicima parka i upravi pomognu u istraživanju ugroženih vrsta na području parka.

LITERATURA

[1] Mobile VS. Desktop Internet Usage, https://www.broadbandsearch.net/blog/mobile-desktop-internet-usage-statistics – 10.9.2020

[2] Nacionalni park Paklenica, https://www.np-paklenica.hr/hr/park-hr/o-parku – 10.9.2020

[3] Google Play, NPS Yellowstone, ažurirano 16.4.2020., https://play.google.com/store/apps/details?id=gov.nps.yell – 29.9.2020

[4] Google Play, Plitvička jezera, ažurirano 25.6.2020., https://play.google.com/store/apps/details?id=com.plavatvornica.plitvice – 29.9.2020

[5] Google Play, NP Krka, ažurirano 15.1.2020., https://play.google.com/store/apps/details?id=com.lilcodelab.npkrka – 29.9.2020

[6] J. Callaham, The history of android: the evolution of the biggest mobile OS in the world, 13.9.2020., https://www.androidauthority.com/history-android-os-name-789433/ – 15.9.2020.

[7] Mobile Operating System Market Share Worldwide, https://gs.statcounter.com/os-market-share/mobile/worldwide – 10.9.2020

[8] S. Gunasekera, Android Apps Security, Apress, 2012., Berkeley, CA

[9] N. Elenkov, Android Security Internals, No Starch Press, Inc., 2015., San Francisco

[10] J. Ross, Andorid Development: Lecture Notes, ažurirano 24.5.2017., https://info448-s17.github.io/lecture-notes/img/introduction/android_architecture.png

[11] Android Developers https://developer.android.com/studio/intro – 1.9.2020.

[12] H. Schildt, Java 2: The Complete Reference, McGraw-Hill/Osborne Media, 2002.

[13] Oracle – The Java Tutorials https://docs.oracle.com/javase/tutorial/index.html – 1.9.2020.

[14] A. Twin, Marketing, Investopedia, ažurirano 17.8.2020., https://www.investopedia.com/terms/m/marketing.asp – 3.9.2020.

[15] M. Leppäniemi, H. Karjaluoto, Mobile Marketing: From Marketing Strategy To Mobile Marketing Campaign Implementation, International Journal of Mobile Marketing, No. 1, Vol. 3, str. 50 – 61., lipanj 2008.

[16] Integrated Marketing Communications, Multimediamarketing, https://multimediamarketing.com/mkc/marketingcommunications/ – 5.9.2020.

SAŽETAK

Cilj ovog rada je izrada mobilne aplikacije koja korisnicima daje pregled ugroženih biljnih i životinjskih vrsta koje se mogu vidjeti u Nacionalnom parku Paklenica te omogućuje korištenje kamere mobilnog uređaja u svrhu praćenja ugroženih vrsta. Ovaj rad sadrži teorijsku osnovu Android operacijskog sustava, Android Studio razvojne okoline, Java programskog jezika, opis razvoja mobilne aplikacije i opis njenog korištenja u svrhu promocije parka. Na kraju je dano objašnjenje marketinga i moguće korištenje aplikacije u te svrhe, usporedba sa sličnim rješenjima te zaključak cjelokupnog rada

Ključne riječi: Android Studio, Android operacijski sustav, Java, marketing, Nacionalni park Paklenica

ABSTRACT

The implementation of mobile marketing in a promotional mix of integrated marketing communications using an Android application intended for the visitors of the Paklenica National Park

The aim of this paper is to create a mobile application which will give its users an overview of the endangered plant and animal species in the Paklenica National Park and enables the use of the device camera for the purpose of tracking endangered species. This paper contains a theoretical background for the Android operating system, Android Studio development environment, Java programming language, instructions for developing the mobile application and a description of its use as a promotional tool for the park. The final part explains what marketing is and the possible use of the application for that purpose, followed by a comparison with similar solutions and the conclusion of the entire paper.

Keywords: Android Studio, Android operating system, Java, marketing, Paklenica National Park

ŽIVOTOPIS

Matija Tivanovac rođen je 15.12.1994. godine u Osijeku. Živi u Belišću gdje stječe osnovnoškolsko obrazovanje u Osnovnoj školi Ivana Kukuljevića Belišće. Godine 2009. upisuje opću gimnaziju u Valpovu. Godine 2013. završava srednju školu s odličnim uspjehom i položenom državnom maturom, te upisuje preddiplomski sveučilišni studij računarstva na Elektrotehničkom fakultetu u Osijeku. Od ostalih znanja i vještina posjeduje odlično znanje engleskog jezika i vrlo dobro znanje njemačkog jezika, vješt je u radu na računalu i ima vozačku dozvolu B kategorije.