Kontrola kvalitete pruženih usluga u turizmu putem informacijsko komunikacijskih tehnologija

Ivić, Slaven

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:974828

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-17**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

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

Sveučilišni studij elektrotehnike

KONTROLA KVALITETE PRUŽENIH USLUGA U TURIZMU PUTEM INFORMACIJSKO KOMUNIKACIJSKIH TEHNOLOGIJA

Završni rad

Slaven Ivić

Osijek, 2017.

SADRŽAJ

1. UVOD ... 1

1.1. Zadatak završnog rada ... 1

2. KORIŠTENE TEHNOLOGIJE ZA IZRADU APLIKACIJE ... 2

2.1. Android operacijski sustav ... 2

2.2. Android Studio ... 3

2.3. Google Firebase i JavaScript Object Notation ... 5

3. IZRADA APLIKACIJE ... 7

3.1. Ideja i osnovni koncept ... 7

3.2. Izrada aplikacijskog sučelja ... 8

3.2.1. Definiranje potrebnog izgleda ... 8

3.2.2. Izrada sučelja ... 9

3.3. Povezivanje aplikacije s bazom ... 18

3.3.1. Povezivanje s bazom Google Firebase ... 18

3.3.2. Oblikovanje podataka za spremanje u bazu ... 19

3.3.3. Povezivanje sučelja i baze ... 20

3.4. Testiranje aplikacije ... 22

3.4.1. Testiranje na uređaju Huawei P9 Lite ... 22

3.4.2. Testiranje na uređaju Nexus 4 (AVD) ... 23

3.4.3. Testiranje na uređaju Nexus 9 (AVD) ... 24

3.4.4. Testiranje spremanja podataka (Firebase) ... 24

4. ANALIZA KORISNOSTI APLIKACIJE ... 26

5. ZAKLJUČAK ... 27

LITERATURA ... 28

SAŽETAK ... 29

ABSTRACT ... 30

ŽIVOTOPIS ... 31

PRILOZI ... 32

1. UVOD

Pri kontroli kvalitete pruženih usluga, kako u turizmu tako i u ostalim djelatnostima, značajne su postale ankete kao vid ispitivanja korisnika o njihovom zadovoljstvu danim uslugama, te povratne informacije dobivene tim putem. Pri tome je važna primjena aplikacija za mobilne uređaje, jer se pokazuju kao brz, jednostavan i jeftin način kanaliziranja informacija dobivenih od strane anketiranih osoba.

Rezultat ovoga rada će biti aplikacija koja će za svrhu imati zamijeniti standardno anketiranje posjetitelja Nacionalnog parka Paklenica putem anketnog upitnika. Njezin cilj je učiniti cijeli postupak kontrole kvalitete pruženih usluga jednostavnijim i bržim te korisnicima pristupačnijim, korištenjem informacijsko komunikacijskih tehnologija. Putem aplikacije će se prikupljati podaci (odgovori posjetitelja na anketna pitanja) te spremati u bazu, odakle se, po potrebi, mogu koristiti pri analizi i izradi statistike. Anketna pitanja koja aplikacija koristi, sastavila je Uprava Nacionalnog parka Paklenica te ih je potrebno implementirati u aplikacijsko sučelje. Pri izradi aplikacije potrebno je poznavati način rada Android operacijskog sustava, razvojno programsko okruženje *Android Studio*, Google-ovu mobilnu platformu *Firebase* za kreiranje baze podataka, proces anketiranja i prikupljanja podataka, te povezati te cjeline u Android aplikaciju odgovarajućeg izgleda i funkcionalnosti.

Drugo poglavlje završnog rada predstavlja teorijski dio korištenih tehnologija za izradu aplikacije - *Android operacijski sustav, Android Studio, Google Firebase i JavaScript Object Notation*. Trećim poglavljem se opisuje izrada aplikacije odnosno izrada njenog sučelja, povezivanje sa *Google Firebase-om* te testiranje aplikacije. Četvrto poglavlje predstavlja analizu korisnosti aplikacije kao primjene informacijsko komunikacijski tehnologija u turizmu i njen značaj za Nacionalni park Paklenica.

1.1. Zadatak završnog rada

Zadatak završnog rada je napraviti analizu korištenja informacijsko komunikacijskih tehnologija u svrhu kontrole kvalitete pruženih usluga u turizmu, te izraditi Android aplikaciju čija bi svrha bila anketiranje posjetitelja o kvaliteti usluge pružene pri posjeti Nacionalnom parku Paklenica.

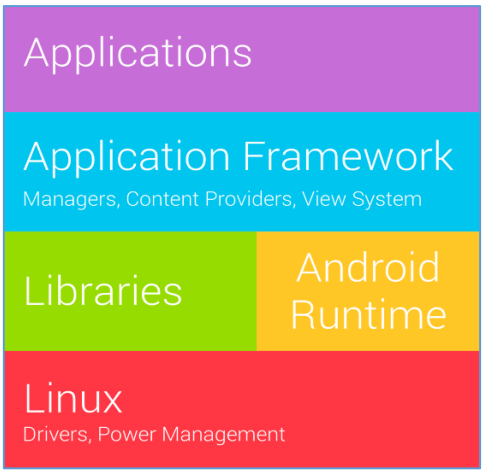
2. KORIŠTENE TEHNOLOGIJE ZA IZRADU APLIKACIJE

U ovom poglavlju opisane su korištene tehnologije koje su potrebne za izradu aplikacije, a to su: *Android* operacijski sustav, *Android Studio, Google Firebase* te *Java Script Object Notation*.

2.1. Android operacijski sustav

Android operacijski sustav je Google-ov operacijski sustav otvorenog koda za mobilne uređaje koji se temelji na Linux-ovom kernelu, dizajniran prvenstveno za uređaje s ekranom osjetljivim na dodir (*touchscreen devices*) poput tableta, pametnih telefona i slično. Android je modularan i prilagodljiv što je pridonijelo njegovoj popularizaciji, pogodan je za inovacije te korisnicima nudi jeftinije i bolje iskustvo uporabe mobilnih uređaja. [1]

Android operacijski sustav je pisan u C/C++ programskom jeziku a arhitektura se može podijeliti na nekoliko razina; na dnu je Linux 2.6 jezgra s upravljačkim programima gdje su najznačajniji upravljački programi za upravljanje napajanjem i upravljački programi za međuprocesnu komunikaciju; iznad jezgre su knjižnice (libraries) pisane u C/C++ programskom jeziku za grafičko sučelje, povezivanje s internetom, za snimanje i reproduciranje poznatih audio/video formata i sl. Sljedeći sloj služi za pokretanje aplikacija (*Android Runtime*) s dvije važne komponente – *Core librabries* koje sadrže jezgrene knjižnice u programskom jeziku Java, te *Dalvik Virtual Machine* koji pokreće aplikacije kao samostalne procese odnosno instance virtualnog stroja. Sljedeći je *Application Framework* (aplikacijski okvir) koji sadrži mehanizme za pisanje aplikacija. Zadnji sloj je sloj aplikacija koji je vidljiv krajnjem korisniku i koji, osim ugrađenih aplikacija, sadrži i aplikacije preuzete s *Android Marketa* (*Google Play trgovine*) čiji broj iz dana u dan raste. Aplikacije se mogu pisati u Java programskom jeziku korištenjem Android SDK (Software Development Kit), što najčešće i jest slučaj, ali također je moguće i pisanje aplikacija u C/C++ programskom jeziku pri čemu se koristi Android NDK (Native Code Development Kit). [1]

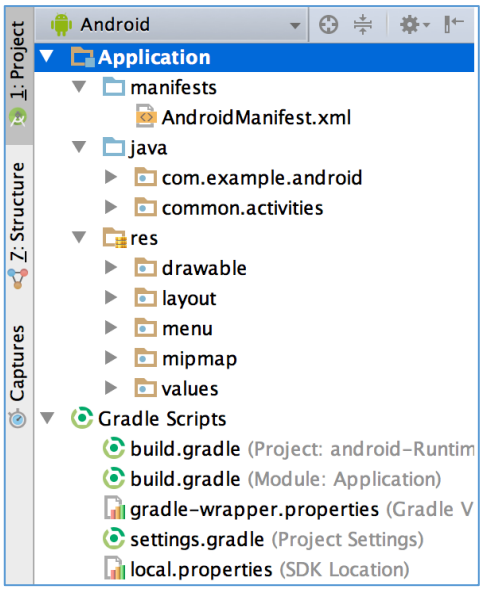


Sl. 2.1. *Arhitektura Androida [2]*

2.2. Android Studio

Android Studio je službeno integrirano razvojno okruženje (engl. *Integrated Development Environment - IDE*) za Android platformu posebno dizajniran za razvoj Android aplikacija. Android Studio se temelji na JetBrains' IntelliJ IDEA te je dostupan za preuzimanje na Windows, Linux ili Mac OS X operacijskim sustavima. [3]

Android Studio koristi fleksibilni *build* sustav (*Gradle*) kako bi kompajlirao i linkao programske datoteke te ih sažima u *.apk* datoteku koja se instalira na Android platformu. Svaki projekt u Android Studio-u sadržava jedan ili više modula; aplikacijski modul (*Android App module*) sastoji se od tri ključna dijela – ***manifest*** datoteke koja daje informacije o aplikaciji, opisuje komponente aplikacije, navodi biblioteke i slično, ***java*** datoteke koja sadrži java kodove te ***res*** datoteke koja sadrži resurse – *.xml* datoteke koje definiraju izgled aplikacije, vrijednosti, stringove i slično. [3]

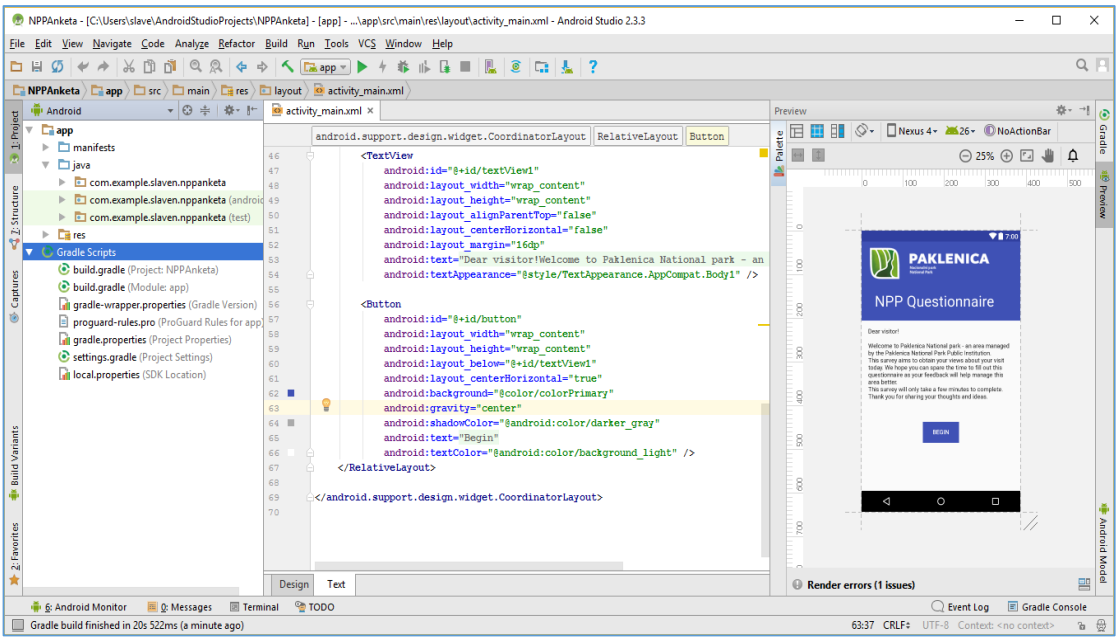


Sl. 2.2. *Dijelovi Android aplikacije [3]*

Izgled aplikacije definiraju *layout*-i odnosno *.xml* datoteke (*Extensible Markup Language – XML*) u Android Studio-u, pa je za dizajniranje izgleda aplikacije potrebno poznavati *XML* jezik za označavanje. Svaki *layout* sadrži određeni broj *XML* objekata kojima je potrebno odrediti parametre i razmještaj radi funkcionalnog izgleda aplikacije.

Android Studio također sadrži *Lint* alate za poboljšanje kvalitete koda, podršku za *Google Cloud* platformu, podršku za *GitHub* te *drag and drop* korisničko sučelje za jednostavno dizajniranje izgleda aplikacije. [3]

Za potrebe zadatka završnog rada koristi se Android Studio 2.2.3 čije je sučelje prikazano na slici 2.3.



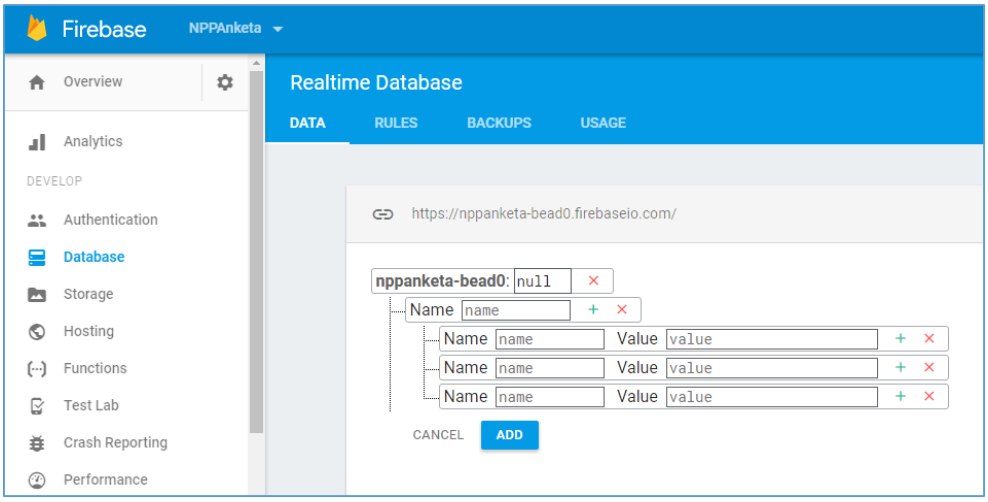
Sl. 2.3. *Izgled Android Studio razvojnog okruženja*

2.3. Google Firebase i JavaScript Object Notation

Firebase je Google-ova razvojna platforma za mobilne i web aplikacije. Firebase su osnovali 2011. Andrew Lee i James Tamplin te je prvotno nudio samo bazu podataka u stvarnom vremenu (*Realtime Database*) za spremanje i sinkronizaciju podataka na više različitih klijenata. Vremenom se ponuda proširila na mnoštvo proizvoda korisnih pri razvoju aplikacija - *Crash Reporting*, *Google Analytics*, *Cloud Storage*, *Hosting*, *Test Lab for Android*, *Performance Monitoring* te brojni drugi. [4]

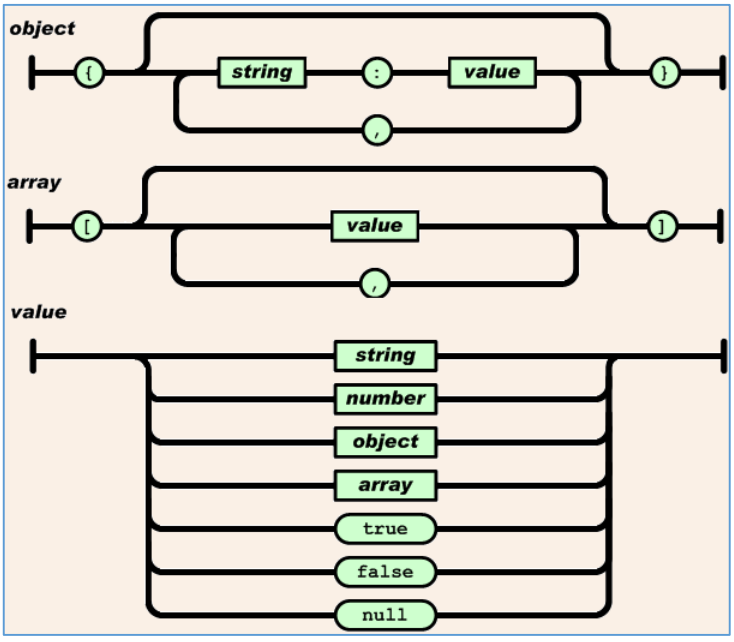
Za potrebe izrade završnog rada koristio se Firebase Realtime Database, kao potrebna baza za spremanje podataka iz aplikacije. To je *NoSQL* baza podataka koja se nalazi na *Firebase Cloud*-u i omogućava spremanje i razmjenu podataka između više korisnika. Osim jednostavnosti korištenja i jednostavnosti integriranja u Android projekt, Firebase Database nudi i *offline* (bezmrežno) korištenje, što znači da ukoliko korisnik ostane privremeno bez mreže i bez pristupa Firebase Cloud-u, podaci će se spremiti lokalno, te prvom sljedećom prilikom *uploadati* i spremiti u Firebase bazu. Format u kojem se podaci spremaju u bazu je *JavaScript Object Notation* (JSON).

Povezivanje Android Studio projekta i Firebase-a je omogućeno putem dostupnog alata u sklopu Android Studia - *Firebase Assistant*, koji kroz par koraka korisniku omogućava jednostavno i brzo povezivanje Android aplikacije i Firebase baze podataka.



Sl. 2.4. *Izgled Firebase konzole za upravljanje bazom podataka*

Kao što je već navedeno Firebase baza podataka ima JSON strukturu odnosno format koji je dizajniran za čitljivu razmjenu podataka; struktura može biti organizirana korištenjem *objekata* koji se označavaju vitičastim zagradama i koje mogu sadržavati druge objekte, polja ili podatke u obliku *naziv:vrijednost*, ili korištenjem polja koji se označavaju uglatim zagradama i koji su zapravo niz vrijednosti. Vrijednosti mogu biti string (s dvostrukim navodnicima), brojevi, *true*, *false*, *null*, ili drugi objekti i polja. Navedeno je prikazano slikom 2.5. [6]



Sl. 2.5. *JSON struktura [6]*

3. IZRADA APLIKACIJE

U ovom poglavlju opisan je cjelokupni proces izrade aplikacije, počevši od ideje i osnovnog koncepta, a uključuje izradu aplikacije u razvojnom okruženju *Android Studio*, povezivanje aplikacije sa *Google Firebase*-om te testiranje aplikacije na nekoliko uređaja.

3.1. Ideja i osnovni koncept

Ideja o primjeni Android aplikacije pri anketiranju posjetitelja o kvaliteti pružene usluge temelji se na činjenici da je takav oblik anketiranja korisnicima pristupačniji, brži i jednostavniji, što u konačnici znači veći odziv na ankete, a time i bolja povratna informacija te bolji uvid i kontrola kvalitete pruženih usluga. Također, primjena aplikacije kao zamjene za stari način anketiranja posjetitelja (papirnati oblik), osim efikasnosti znači i ekonomičnost u vidu uštede resursa – nema potrebe za printanjem anketnih materijala. Ovim načinom optimiraju i vrijeme obrade rezultata dobivenih anketom. Do sada su određene osobe duži period godine obavljale analizu prikupljenih podataka, što je značilo veću mogućnost pogreške, sporost u obradi podataka, ali i gubitak vremena koji su djelatnici mogli iskoristiti za važnije poslove. Prisutna je i marketinška komponenta u cilju promocije parka budući da ovakav oblik anketiranja nije čest u Republici Hrvatskoj (Nacionalni park Kornati ima sličan oblik anketiranja, no putem web aplikacije - Online anketa [5]).

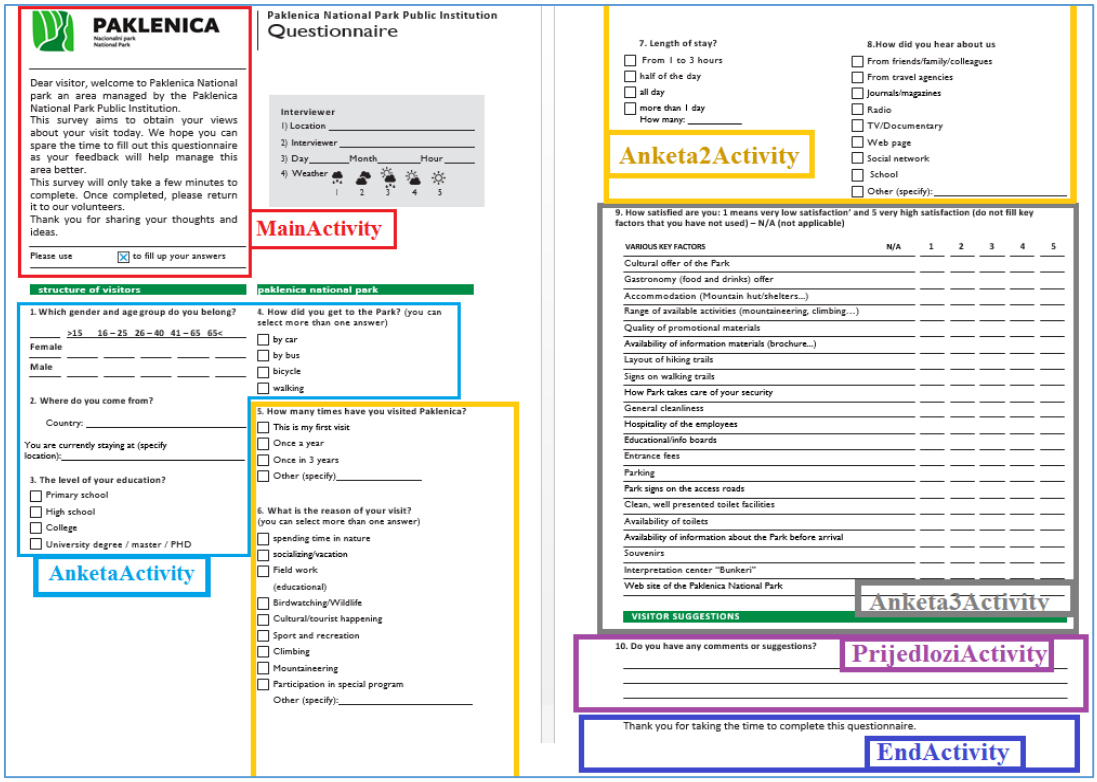
Osnovni koncept aplikacije je sljedeći – korisnik pokreće aplikaciju nakon čega se pojavljuje početni zaslon (*activity*) dobrodošlice, par uvodnih riječi te dugme za početak ankete; klikom na dugme za početak otvara se sljedeći zaslon koji sadrži prvi blok anketnih pitanja; izgled zaslona ovisi o pitanjima i formatu traženog odgovora na odgovarajuće pitanje; klikom na dugme za nastavak prikazuje se sljedeći blok pitanja sve do konačnog zaslona kojim se korisniku daje do znanja da su njegovi odgovori zabilježeni te sadrži zahvalu za sudjelovanje u anketi; u međuvremenu odgovori korisnika su spremljeni u Firebase bazu podataka. Android sustav, kao najpopularnija platforma, idealna je za realizaciju ovakve jednostavne android aplikacije koja će biti dostupna velikom broju korisnika.

3.2. Izrada aplikacijskog sučelja

3.2.1. Definiranje potrebnog izgleda

Kako bi se započelo s dizajniranjem aplikacijskog sučelja, potrebno je ponajprije definirati potreban izgled aplikacije. Očekivani izgled ankete je zadala Uprava Nacionalnog parka Paklenice (Sl. 3.1.). Pri izradi aplikacije potrebno ju je najprije podijeliti na određeni broj zaslona (*activity*), pri čemu svaki od njih ima vlastiti *layout* te se svaki ponaša kao zasebna cjelina koje su međusobno povezane – iz jednog zaslona se prelazi na sljedeći, klikom na odgovarajuće dugme pri čemu je redoslijed zaslona unaprijed definiran.

S obzirom na zadanu anketu, aplikacija je podijeljena na sljedeće cjeline: *MainActivity* (prvi zaslon nakon pokretanja aplikacije, zaslon uvoda i dobrodošlice), *AnketaActivity* (prvi dio anketnih pitanja), *Anketa2Activity* (drugi dio anketnih pitanja), *Anketa3Activity* (treći dio anketnih pitanja), *PrijedloziActivity* (komentari i prijedlozi posjetitelja), te *EndActivity* (završni zaslon zahvale za sudjelovanje u anketi).



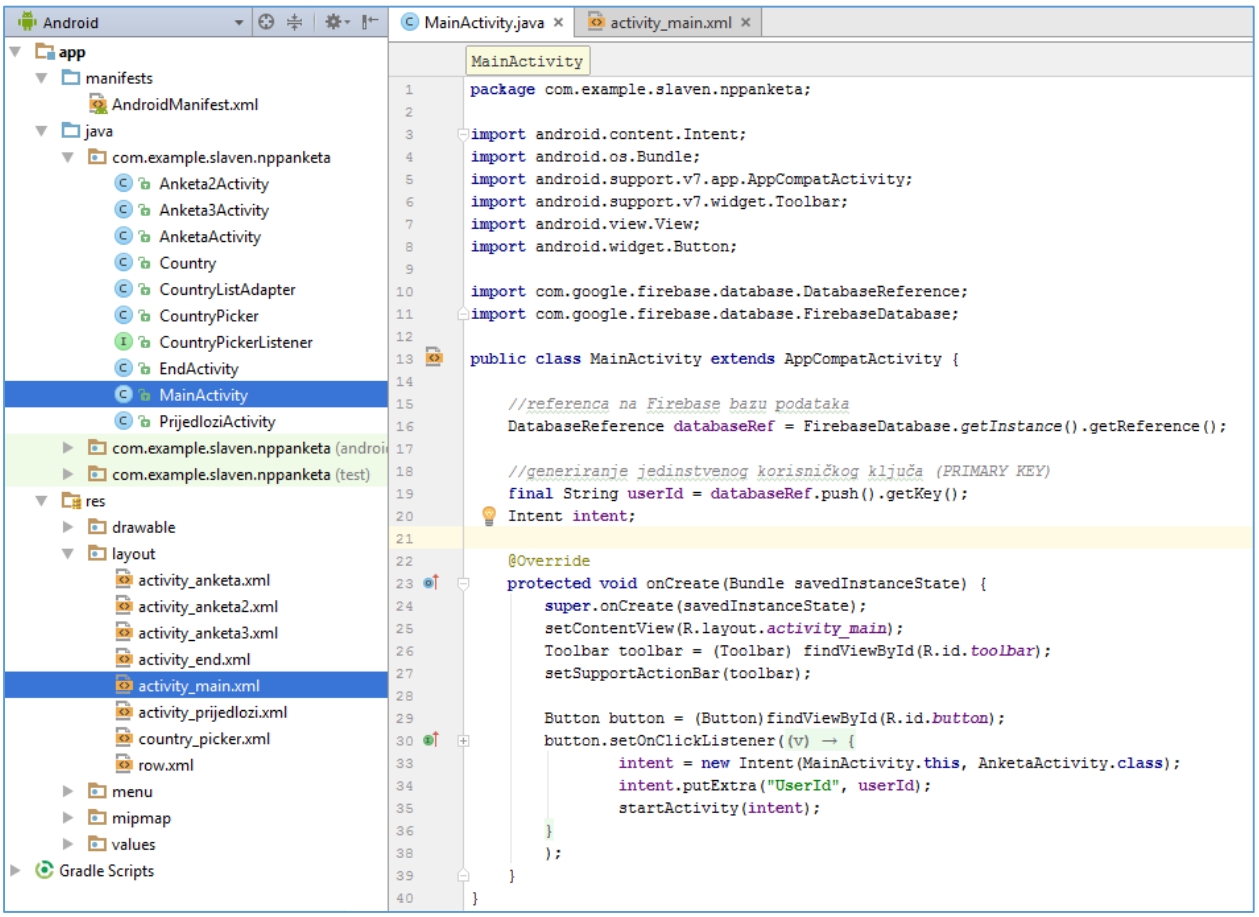
Sl. 3.1. *Analiza izgleda ankete i raspodjela zaslona*

Nakon provedene analize izgleda ankete i odabrane raspodjele dijelova aplikacije, izrada aplikacijskog sučelja je realizirana u razvojnom okruženju Android Studio kako je to i prije navedeno.

3.2.2. Izrada sučelja

Svaki *activity*, dakle, ima svoj *layout* kojim se definira sam izgled zaslona, te pripadajući *Java* kod, odnosno pripadajuću *Java* klasu (engl. *Java class*) koja radi sa objektima i podacima odgovarajućeg zaslona, komunicira sa drugim klasama i slično, odnosno sadrži programerski dio aplikacije. Dakle, pri izradi funkcionalnog sučelja aplikacije, potrebno je za svaki zaslon aplikacije napisati njegov *layout* u XML jeziku, te njegovu klasu u Java jeziku. Android Studio pri kreiranju svakog novog zaslona (*New Activity*) automatski generira i njegovu klasu (*Activity.class*) te pripadajući *layout* (*activity.xml*) što olakšava izradu sučelja.

Pri izradi prvog zaslona *MainActivity* generirana je njegova klasa (*MainActivity.class*) i njegov pripadajući *layout* (*activity_main.xml*).



Sl. 3.2. *Prikaz klase MainActivity*

Kao što je vidljivo na slici 3.2., u Android načinu prikaza projekta, kreirane klase se nalaze unutar mape *app/java/naziv_paketa(com.example.slaven.nppanketa)*, a *.xml* datoteke *layout-a* unutar mape *res/layout*.

MainActivity je klasa koja se pokreće s pokretanjem aplikacije pa je u *manifest* datoteci dodano obilježje, filter za dotičnu klasu (*.LAUNCHER*), što je prikazano na sljedećoj slici:

Sl. 3.3. *AndroidManifest.xml – kod za MainActivity*

Kao što je vidljivo na slici 3.3. unutar *manifest* datoteke, koja također koristi XML jezik za označavanje, moguće je definirati temu aplikacije ili temu pojedinog *activity-ja* kao što je to slučaj sa *MainActivity* i *EndActivity* koji sadrže logo nacionalnog parka te je odabrana tema *AppTheme.NoActionBar*.

Izgled *MainActivity-ja* definiran je XML jezikom u *activity_main.xml* datoteci; XML jezik opisuje ono što korisnik vidi na ekranu a u slučaju ovog zaslona, potrebno je bilo opisati nekoliko objekata – *TextView*, *ImageView*, *Button* te *RelativeLayout* i *LinearLayout* koji sadržavaju navedene objekte. *LinearLayout* koji je vertikalne orijentacije, sadrži dva *RelativeLayout-a*, jedan ispod drugog, od kojih jedan sadrži *TextView* za stilizirani prikaz naziva aplikacije i *ImageView*

koji sadrži logo nacionalnog parka, a drugi *TextView* za prikaz teksta uvodne riječi i *Button* za početak ankete tj. prelazak na sljedeći zaslon (*AnketaActivity*).

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="160dp"
        android:background="@color/colorPrimary"
        android:text="@string/app_name"
        android:textColor="@color/WhiteP"
        android:textSize="35sp"
        android:gravity="bottom"
        android:padding="20dp"
        />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="335dp"
        android:layout_height="wrap_content"
        tools:ignore="ContentDescription"
        app:srcCompat="@drawable/logo" />

</RelativeLayout>
```

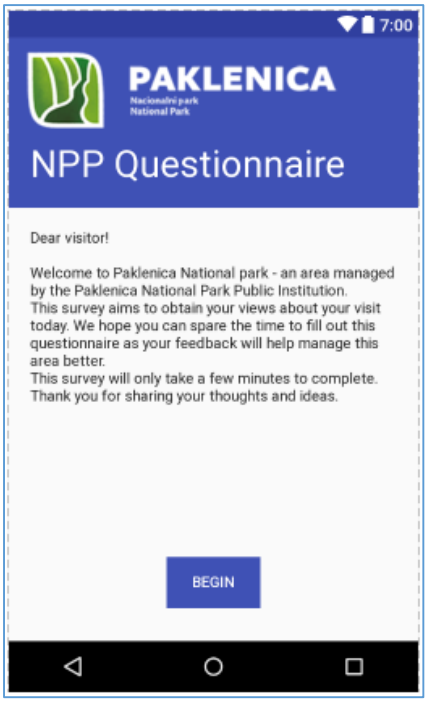
```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_margin="20dp"
        android:text="@string/text_main"
        android:textAppearance="@style/TextAppearance.AppCompat.Body1"
        />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:background="@color/colorPrimary"
        android:text="Begin"
        android:textColor="@android:color/background_light"
        android:layout_marginBottom="31dp"
        />
</RelativeLayout>
```

Sl. 3.4. *XML kod activity_main layout-a*

XML kod sa slike 3.4. za rezultat daje izgled *activity_main* layout-a (Sl. 3.5.). Kako bi navedeni *activity* imao određenu funkcionalnost, potrebno je napisati određeni Java kod unutar klase *MainActivity.class*.



Sl. 3.5 *Izgled activity_main layout-a*

Ključni dio Java koda u *MainActivity.class* je korištenje metode *onCreate( )* kojom se pri pokretanju activity-ja postavlja njegov izgled tako da se postavi odgovarajući layout iz resursa - *setContentView(R.layout.activity_main)*. Također, unutar iste metode postavljen je *listener* na dugmetu *Button* koji osluškuje i čeka klik na dugme; ukoliko korisnik klikne na dugme, izvršava se dio Java koda unutar *listener-a*, a to je u ovom slučaju pokretanje novog activity-a, odnosno *intent-a* (*MainActivity -> AnketaActivity*), što je prikazano na slici 3.6.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    Button button = (Button)findViewById(R.id.button);
    button.setOnClickListener(new View.OnClickListener(){
        @Override
        public void onClick(View v){
            intent = new Intent(MainActivity.this, AnketaActivity.class);
            startActivity(intent);
        }
    });
}
```

Sl. 3.6. *Metoda onCreate( ) unutar Java klase MainActivity*

Pokretanjem drugog activity-a odnosno zaslona (*AnketaActivity*), počinje anketa s prvim blokom pitanja (pitanja 1.-4.). Njegov layout čine *ScrollView* koji omogućuje korisniku listanje po zaslonu ukoliko je layout duži od prikaza na zaslonu (što ovdje jest slučaj), vertikalni *LinearLayout* koji se nalazi unutar *ScrollView-a* i koji sadrži pitanja poredana jedno ispod drugog u vidu *RelativeLayout-a*, dakle, svako pitanje ima svoj *RelativeLayout*. Prvo pitanje se sastoji od *TextView-a* koji prikazuje tekst prvog pitanja što se postiže dodavanjem sljedećeg koda:

```
android:text="@string/pitanje1"
-------------------------------------------------------------------------------
<string name="pitanje1">1.Which gender and age group do you belong?</string>
```

Sl. 3.7. *Postavljanje teksta*

Navedene linije koda, kao tekst postavlja unaprijed definiranu tekstualnu vrijednost (string) koja se nalazi u resursima (*res/values/strings.xml*). Ovo omogućuje izradu sučelja aplikacije na različitim jezicima, budući da Android u ovisnosti o postavkama mobilnog uređaja (postavke *jezika i regije*), automatski odlučuje koje će tekstualne vrijednosti koristiti odnosno iz kojih resursa će ih uzimati.

Kod prvog pitanja spol se odabire klikom na jedan od dva ponuđena *item-a* objekta *Spinner* koji se ponaša kao padajući izbornik u kome je moguće odabrati odgovarajući *item* s liste.

Starost se odabire putem *RadioButton-a* koji se nalaze unutar grupe (*RadioGroup*), što osigurava odabir isključivo jedne od ponuđenih opcija (dobne skupine). Korisnikov odabir se bilježi odgovarajućim Java kodom u klasi *AnketaActivity* (Sl. 3.8.).

```
//spremanje pitanja 1 - spol
int g=spinner.getSelectedItemPosition();
switch(g){
    case 0:  //određena akcija
        break;
    case 1:  //određena akcija
        break;
}
```

```
//spremanje podataka starosti - pitanje 1
RadioGroup rg = (RadioGroup)findViewById(R.id.radioGroup);
int a=rg.getCheckedRadioButtonId();
switch (a){
    case R.id.radioButton115:   //određena akcija
        break;
    case R.id.radioButton1625:  //određena akcija
        break;
    case R.id.radioButton2640:  //određena akcija
        break;
    case R.id.radioButton4165:  //određena akcija
        break;
    case R.id.radioButton66:    //određena akcija
        break;
    case -1:    //nije odabrana nijedna stavka
        break;
}
```

Sl. 3.8. *Bilježenje korisnikovog odabira (pitanje1, AnketaActivity)*

Kod drugog pitanja dva *EditText* objekta prikupljaju korisnikov tekstualni unos o lokaciji (zemlja odakle dolazi i grad u kojem je trenutno smješten). Tekst korisnikovog unosa moguće je dobiti korištenjem sljedećeg Java koda:

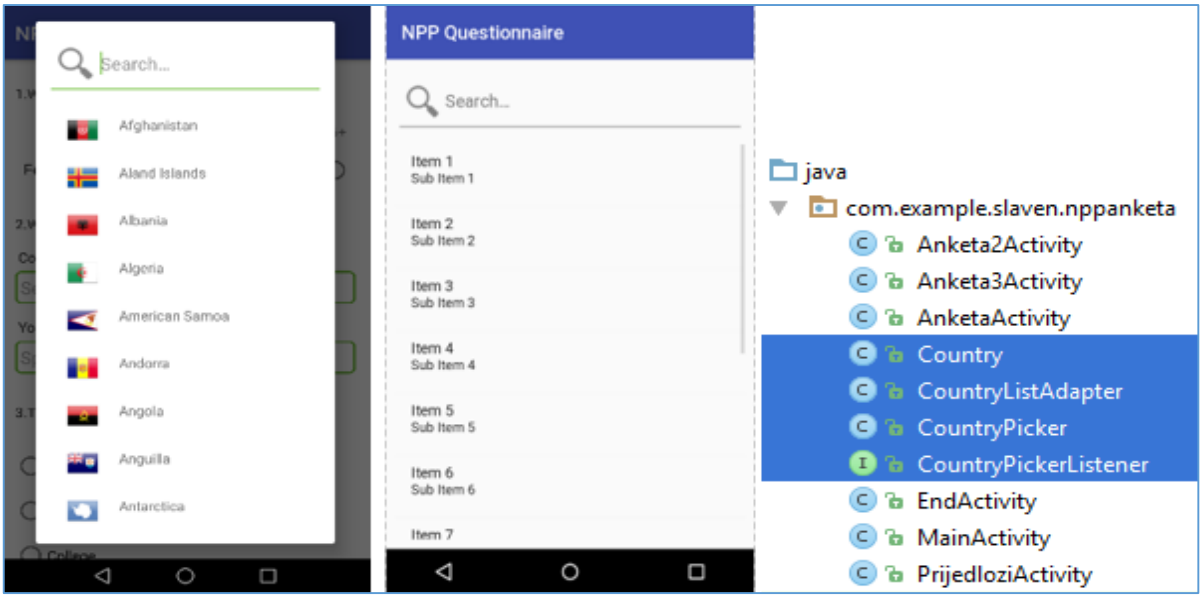
```
final EditText editText= (EditText) findViewById(R.id.editText);
        String tekst = editText.getText().toString();
```

Sl. 3.9. *Dobivanje korisnikovog unosa*

Navedeni kod prikazuje *id* objekta (*view)* – *R.id.editText*, što je zapravo *id* objekta *EditText* opisanog XML jezikom u datoteci *activity_anketa.xml*. Analogno je moguće zabilježiti bilo kakav korisnikov tekstualni unos i u sljedećim pitanjima.

Pri odabiru zemlje/države iz koje dolazi, korisniku je omogućen odabir zemlje putem *CountryPicker-a*. To je klasa kreirana posebno za odabir zemlje koja se poziva kada korisnik klikne na *EditText* u layout-u *AnketaActivity* pri čemu se pojavljuje *dialog*, prozor koji sadrži spisak zemalja svijeta s pripadajućim zastavama; korisnik tada može odabrati bilo koju zemlju sa spiska pri čemu će njegov odabir rezultirati ispisom naziva zemlje u objekt *EditText*. Za rad *CountryPicker-a* kreirane su i klase (Sl.3.10. pod *c)*): *CountryPickerListener.class* – sučelje, *Country.class* – klasa s atributima naziv i šifra zemlje, *CountryListAdapter.class* – klasa s metodama koje postavljaju izgled prozora za odabir zemlje, tako da se iz resursa (*res/drawable*)

postave ikone zastava zemalja te odgovarajući nazivi, a korisniku je omogućeno i traženje svoje zemlje putem tražilice – dodanog polja za unos teksta koji zatim sužava listu ponuđenih zemalja. Izgled prozora za odabir zemlje prikazan je na slici 3.10. pod *a).*



Sl. 3.10. *a) Izgled prozora za odabir zemlje b)layout country_picker.xml c)potrebne klase*

CountryPicker naravno ima i svoj layout na osnovu kojeg je i definiran izgled prozora za odabir zemlje; njegov layout, prikazan na slici 3.10. pod *b),* je jednostavan i sastoji se od *LinearLayout-a* unutar kojeg su tražilica *EditText* s ikonom traženja (povećalo) te *ListView*, objekt koji omogućuje prikaz liste koja sadrži više *item-a* a čiji je izgled definiran s layout-om *row.xml*. Spomenuti layout sadrži horizontalno orijentirani *LinearLayout* s dva objekta – *ImageView* za zastavu, te *TextView* za naziv zemlje.

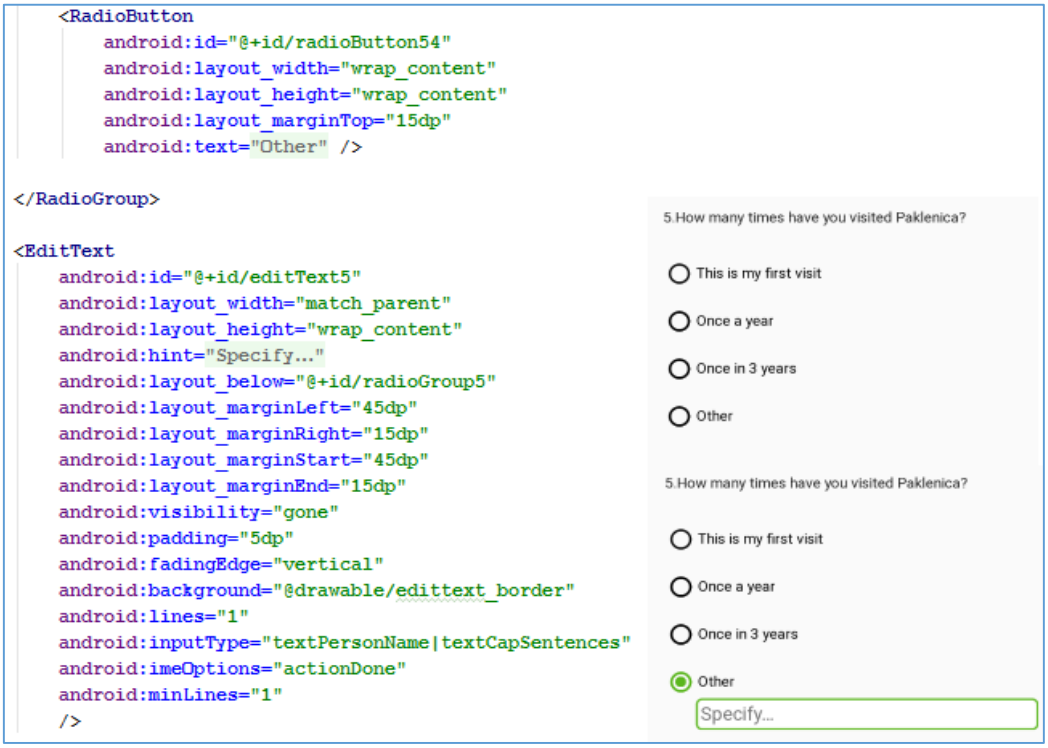
Za odabir mjesta, korisniku je omogućeno automatsko završavanje teksta, s obzirom da se radi o hrvatskim gradovima. To je postignuto uporabom *AutoCompleteTextView-a* unutar samog layout-a, te uporabom *adaptera* koji prima polje stringova kao parametar i prilikom korisnikovog unosa teksta nudi opcije sa liste (gradove) koji sadržavaju odgovarajuća slova (Sl. 3.11.).

```
AutoCompleteTextView textView = (AutoCompleteTextView) findViewById(R.id.editText2);
ArrayAdapter<String> adapter2 =
        new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, gradovi);
textView.setAdapter(adapter2);
```


Sl. 3.11. *AutoCompleteTextView za odabir mjesta*

Ostatak prvog bloka pitanja ne zahtjeva veću pozornost jer se s njima rukuje na način prethodno objašnjen, uz napomenu da četvrto pitanje daje mogućnost višestrukog odabira, pa se umjesto *RadioButton-a* koriste *CheckBox-ovi*, čije je stanje (označen/odabran-neoznačen/neodabran) moguće provjeriti ispitivanjem određenog uvjeta (*if(cb.isChecked())*).

Na dnu prvog bloka pitanja je dugme *Dalje* (*Next*) kojim se otvara sljedeći blok pitanja – *Anketa2Activity*. Kao što je to i ranije bio slučaj, i ovaj *layout* koristi objekte *RadioButton, RadioGroup,* te *CheckBox* u slučaju mogućnosti višestrukog odabira, kako bi prikupio korisnikove odgovore na pitanja. Jedina novost je opcija *Ostalo* (*Other*) pri čemu se pojavljuje XML jezikom opisani *EditText* te na taj način omogućuje korisniku unos teksta kao dodatnog pojašnjenja *Specify* (Sl. 3.12.)



Sl. 3.12. *Realizacija unosa teksta pri odabiru opcije „Ostalo“*

Navedeno je realizirano pomoću parametra *visibility* (vidljivost) objekta za unos teksta, kojemu se vrijednost mijenja kada korisnik odabere opciju *Ostalo* – iz *GONE* u *VISIBLE*. Kako bi aplikacija znala kada korisnik odabire navedenu opciju te u skladu s tim prikazivala/sakrivala objekt za unos teksta, u klasi *Anketa2Activity* je napisan odgovarajući Java kod – koristi se metoda *ostaloFunkcija( ),* koja za parametre prima *RadioButton-e* te pripadajući *EditText* kojem se mijenja parametar vidljivost u *VISIBLE* kada je odabrana zadnja opcija - *RadioButton Ostalo*(*rb4*), odnosno u *GONE* kada je odabrana bilo koja druga opcija danog pitanja (*rb1*,*rb2*,*rb3*). Metoda je prikazana na slici 3.13.

```
//pomoćna metoda
public void ostaloFunkcija (final RadioButton rb1, final RadioButton rb2, final RadioButton rb3,
                            final RadioButton rb4, final EditText ed){
    rb1.setOnClickListener((view) → { ed.setVisibility(View.GONE); });
    rb2.setOnClickListener((view) → { ed.setVisibility(View.GONE); });
    rb3.setOnClickListener((view) → { ed.setVisibility(View.GONE); });
    rb4.setOnClickListener((view) → { ed.setVisibility(View.VISIBLE); });
}
```

Sl. 3.13. *Java kod pomoćne metode ostaloFunkcija( )*

Također, kao što je slučaj sa svim anketnim zaslonima, na dnu se nalazi dugme (*Button* objekt) koji vodi u sljedeću „aktivnost" (eng. *activity*) – u ovom slučaju *Anketa3Activity.*

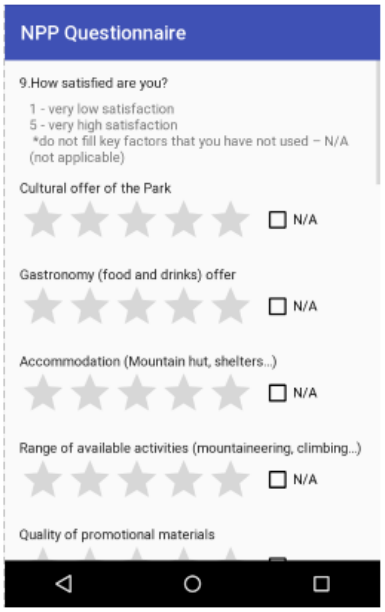
Treći blok pitanja - *Anketa3Activity* je zapravo *layout* koji predstavlja ocjenjivanje pojedinih stavki odnosno korisnikovo zadovoljstvo određenim sadržajem, uslugom, ponudom koju Park nudi – sigurnost, smještaj, čistoća, ljubaznost, kulturološka ponuda, parking i slično. To je postignuto korištenjem XML objekta *RatingBar* unutar layout-a; svaka stavka koja se ocjenjuje ima svoj *RelativeLayout* koji sadrži već spomenuti *RatingBar*, *TextView*, te *CheckBox* kojim je korisniku omogućena opcija *Not Applicable – N/A* u slučaju da korisnik određenu uslugu nije koristio. XML kod jedne od stavki prikazan je na slici 3.14., zajedno sa realiziranim zaslonom.

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="15dp"
    android:layout_marginStart="15dp">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:text="Gastronomy (food and drinks) offer"
        android:textAppearance="@style/TextAppearance.AppCompat"/>

    <RatingBar
        android:id="@+id/ratingBar2"
        style="@style/Widget.AppCompat.RatingBar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView2"
        android:numStars="5"
        android:stepSize="1" />
    <CheckBox
        android:id="@+id/radioButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@+id/ratingBar2"
        android:layout_toEndOf="@+id/ratingBar2"
        android:layout_below="@+id/textView2"
        android:layout_margin="8dp"
        android:text="N/A"/>

</RelativeLayout>
```



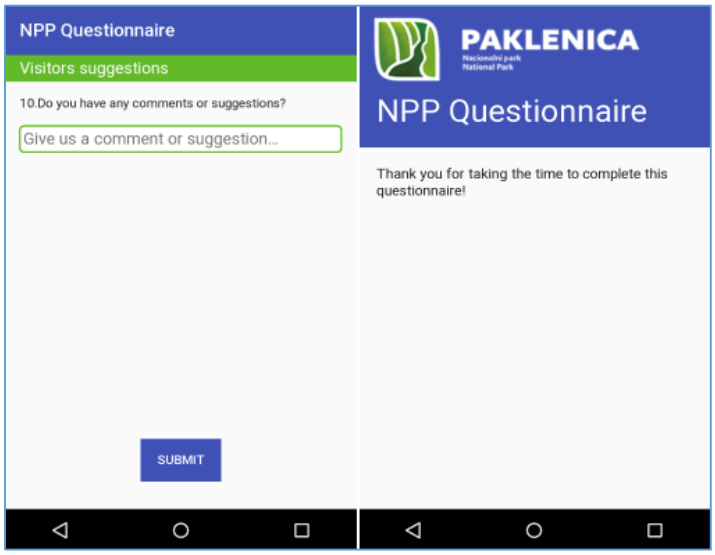
Sl. 3.14. *XML kod i layout Anketa3Activity-a*

Korisnikova ocjena određene stavke se može dobiti sljedećim Java kodom:

```
int ocjena=(int)rbar.getRating();
```

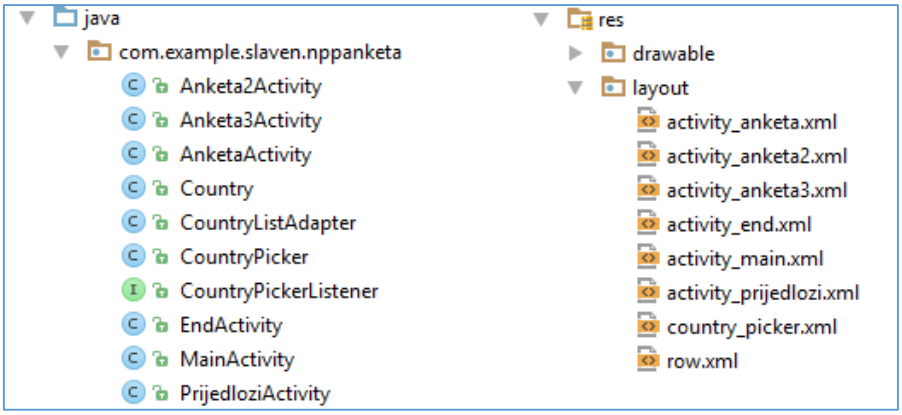
Sl. 3.15. *Dohvaćanje korisnikove ocjene*

Prelaskom na sljedeću aktivnost – *PrijedloziActivity*, korisniku je omogućeno davanje prijedloga i komentara Parku; to je realizirano pomoću objekta za unos teksta *EditText* i *Button-a* koji sprema uneseni tekst i završava anketu (engl. *Submit*). Tada se prikazuje posljednji zaslon aplikacije – *EndActivity*, kojim se korisniku zahvaljuje na izdvojenom vremenu za rješavanje ankete; *layout* zadnjeg activity-a *activity_end.xml* je isti kao i početni layout aplikacije (*activity_main.xml*) uz izmjenu teksta koji se prikazuje te on ne sadrži dugme za početak ankete. Layout-i *activity_end.xml* i *activity_prijedlozi.xml* prikazani su na slici 3.16.



Sl. 3.16. *PrijedloziActivity i EndActivity*

Ovime je izrada aplikacijskog sučelja gotova te se pristupilo drugoj fazi izrade aplikacije – povezivanje aplikacije sa bazom; popis svih potrebnih klasa i *layout*-a prikazan je na slici 3.17.

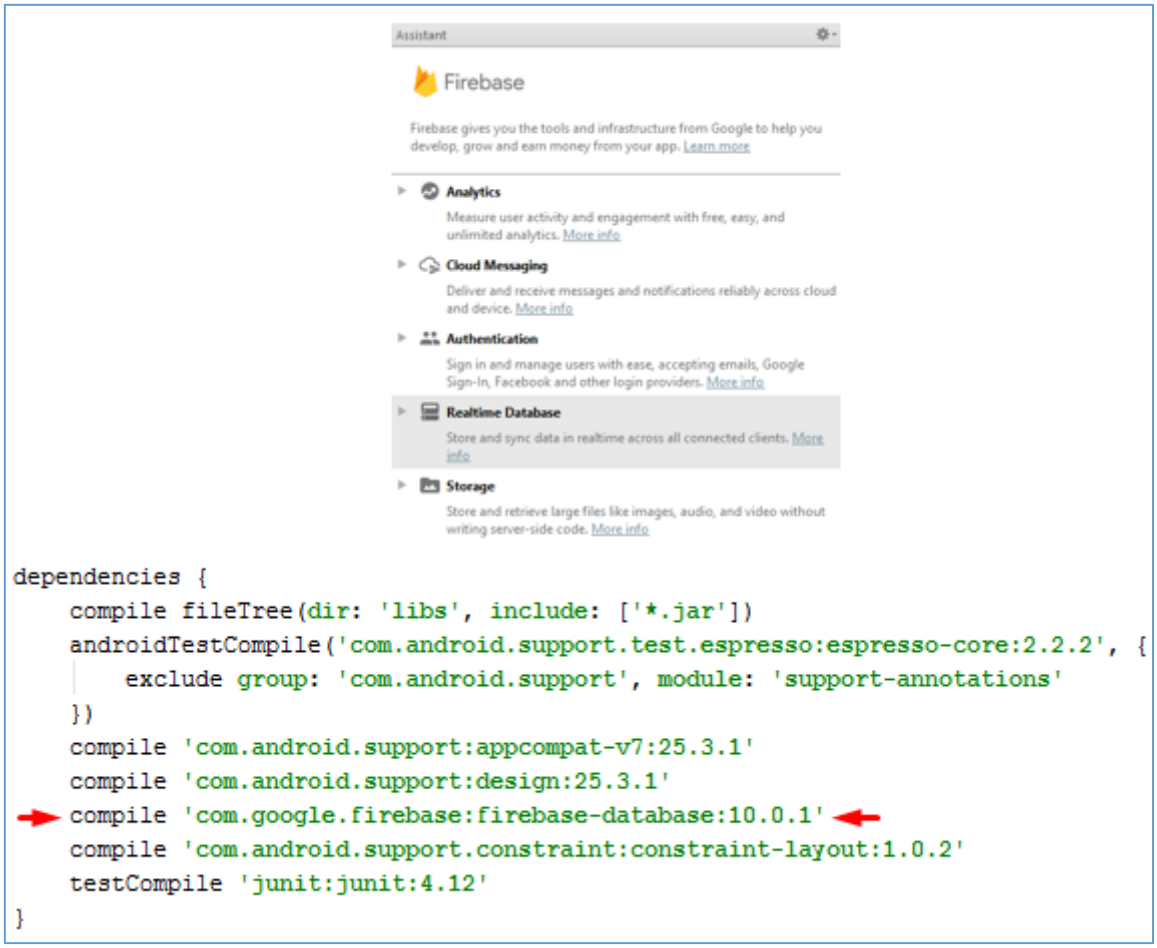


Sl. 3.17. *Prikaz korištenih klasa i layout-a*

3.3. Povezivanje aplikacije s bazom

3.3.1. Povezivanje s bazom Google Firebase

Za povezivanje Android projekta (aplikacije) sa *Google Firebase-om* potrebno je dodati određene zavisnosti (engl. *dependencies*) unutar *build.gradle* datoteke (*Module:app*), što je prikazano na slici 3.18. Android Studio (inačica 2.2.3) omogućava povezivanje projekta s Firebase-om putem alata *Firebase Assistant* (Sl. 3.18.), što je već navedeno u teorijskom dijelu o Firebase-u. Njime se omogućava jednostavno i brzo povezivanje Android aplikacije i Firebase baze podataka kroz par koraka – povezivanje projekta i Firebase-a putem Google računa (u sklopu izrade baze kreiran je Google račun za pristup bazi nppaklenicadb@gmail.com), postavljanje zavisnosti (*dependencies*), postavljanje pravila za bazu (nema potrebe za autentikacijom prilikom pisanja u bazu, kako bi se podaci svih korisnika koji ispunjavaju anketu mogli nesmetano spremati u bazu), te upute za Java kod kojim se piše i čitaju podaci iz Firebase baze.



Sl 3.18. *Firebase Assistant i potrebni dependencies*

Nakon što je Android projekt povezan sa bazom, moguće je pisati u bazu na sljedeći način:

```
//referenca na Firebase bazu podataka
DatabaseReference databaseRef = FirebaseDatabase.getInstance().getReference();
databaseRef.child ("Test")setValue("Hello, World!");
```

Sl. 3.19. *Pisanje podataka u bazu*

Svaka Java klasa će sadržavati ovaj Java kod (u donekle izmijenjenom obliku) kako bi mogla spremati korisnikove odgovore na pitanja u bazu podataka. Kakav je oblik podataka potreban za spremanje u bazu s obzirom da je cijela baza u JSON formatu, objašnjeno je u sljedećoj cjelini.

3.3.2. Oblikovanje podataka za spremanje u bazu

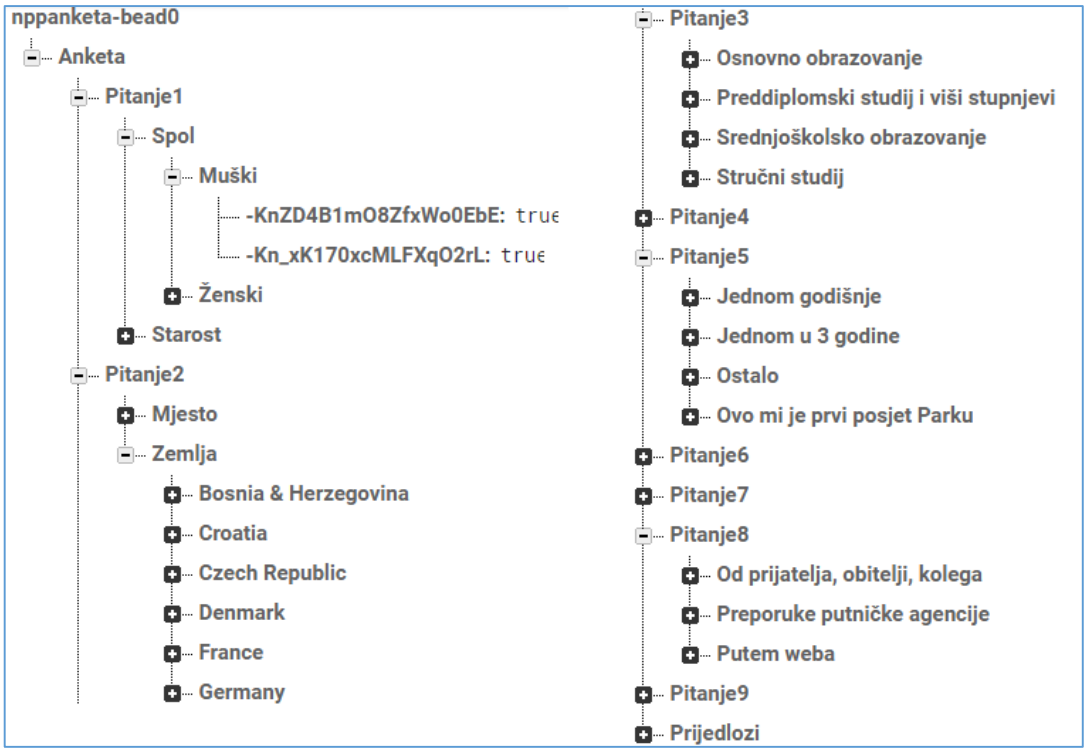
Pri spremanju podataka – korisnikovih odgovora na pitanja, baza će biti koncipirana na sljedeći način (JSON struktura): temeljni objekt se naziva *Anketa{}* koja sadržava objekte *Pitanja* - *Pitanje1{}, Pitanje2{},...,Pitanje10{}*. Dakle svako pitanje je objekt koji sprema određene vrijednosti. Kako bi korisnikov odabir bio zabilježen i jedinstven, Firebase nudi generiranje jedinstvenog ključa koji je zapravo string od 20 znakova s ogromnim brojem kombinacija kako bi se osigurala jedinstvenost ključa (primjer: „*-KnZD4B1mO8ZfxWo0EbE"*). Ključ se generira na početku ankete (*MainActivity.class*) te se prosljeđuje svakom sljedećem activity-ju pomoću sljedećeg Java koda:

```
//generiranje jedinstvenog korisničkog ključa (PRIMARY KEY)
final String userId = databaseRef.push().getKey();
intent.putExtra("UserId", userId);
startActivity(intent);

//dohvaćanje ključa
final String userId = getIntent().getExtras().getString("UserId");
```

Sl. 3.20. *Generiranje i dohvaćanje jedinstvenog ključa*

Dakle, svaka klasa anketnih activity-ja sadržavat će navedeni Java kod te referencu na Firebase bazu podataka kako bi omogućila ispravno spremanje podataka u bazu. Podaci se spremaju unutar odgovarajućeg pitanja (objekta), te unutar odgovarajućeg odgovora (objekta), kao vrijednost: „userId":*true*. Prikaz strukture baze podataka vidljiv je na slici 3.21.



Sl. 3.21. *Struktura baze podataka*

3.3.3. Povezivanje sučelja i baze

Nakon što je unutar svake anketne klase definirana referenca na bazu podataka te korisnički ključ, potrebno je još u ovisnosti o korisnikovom odabiru određene podatke spremiti u bazu. Taj postupak će biti prikazan na primjeru prvog bloka pitanja (*AnketaActivity.class*)

Dakle unutar *onClickListener-a* postavljenog na *Button-u* za prijelaz na sljedeći blok pitanja, provjerava se koji od ponuđenih odgovora je korisnik odabrao te se njegov odabir ili njegov tekstualni unos sprema u bazu pod odgovarajućim objektom. Navedeni proces je prikazan na slici 3.23. U slučajevima kada se od korisnika traži da odabere barem jedan od ponuđenih odgovora, uvedena je zastavica *f* kojom se signalizira da korisnik nije odgovorio na dato pitanje te ukoliko je to bio slučaj, korisniku će prijelaz na sljedeći blok pitanja biti onemogućen i ispisat će se poruka da je potrebno odgovoriti na sva pitanja. Taj uvjet (je li korisnik ostavio neko od pitanja neodgovorenim, odnosno je li zastavica postavljena na vrijednost *-1*) se provjerava na kraju klase unutar izvršavanja *onClick* metode (Sl. 3.22.).

```
if(f==-1){
    Toast.makeText(AnketaActivity.this, "Molim odgovorite na sva pitanja. Hvala!", Toast.LENGTH_SHORT).show();
}
else {
    Intent intent = new Intent(AnketaActivity.this, Anketa2Activity.class );
    intent.putExtra("UserId", userId);
    startActivity(intent);
}
```

Sl. 3.22. *Provjera uvjeta za odgovaranje na sva pitanja*

```
//spremanje pitanja 2 - država i mjesto
if (editText.getText().toString().trim().equals("") || editText2.getText().toString().trim().equals(""))
    f=-1;
else {
    databaseRef.child("Anketa/Pitanje2/Zemlja").child(editText.getText().toString()).child(userId).setValue(true);
    databaseRef.child("Anketa/Pitanje2/Mjesto").child(editText2.getText().toString()).child(userId).setValue(true);
}

//razina obrazovanja - pitanje 3
RadioGroup rg3 = (RadioGroup)findViewById(R.id.radioGroup3);
int e=rg3.getCheckedRadioButtonId();
switch (e){
    case R.id.radioButton31:
        databaseRef.child("Anketa/Pitanje3").child("Osnovno obrazovanje").child(userId).setValue(true);
        break;
    case R.id.radioButton32:
        databaseRef.child("Anketa/Pitanje3").child("Srednjoškolsko obrazovanje").child(userId).setValue(true);
        break;
    case R.id.radioButton33:
        databaseRef.child("Anketa/Pitanje3").child("Stručni studij").child(userId).setValue(true);
        break;
    case R.id.radioButton34:
        databaseRef.child("Anketa/Pitanje3").child("Preddiplomski studij i viši stupnjevi").child(userId).setValue(true);
        break;
    case -1:    f=-1; break;
}
```

Sl. 3.23. *Spremanje podataka u Firebase bazu*

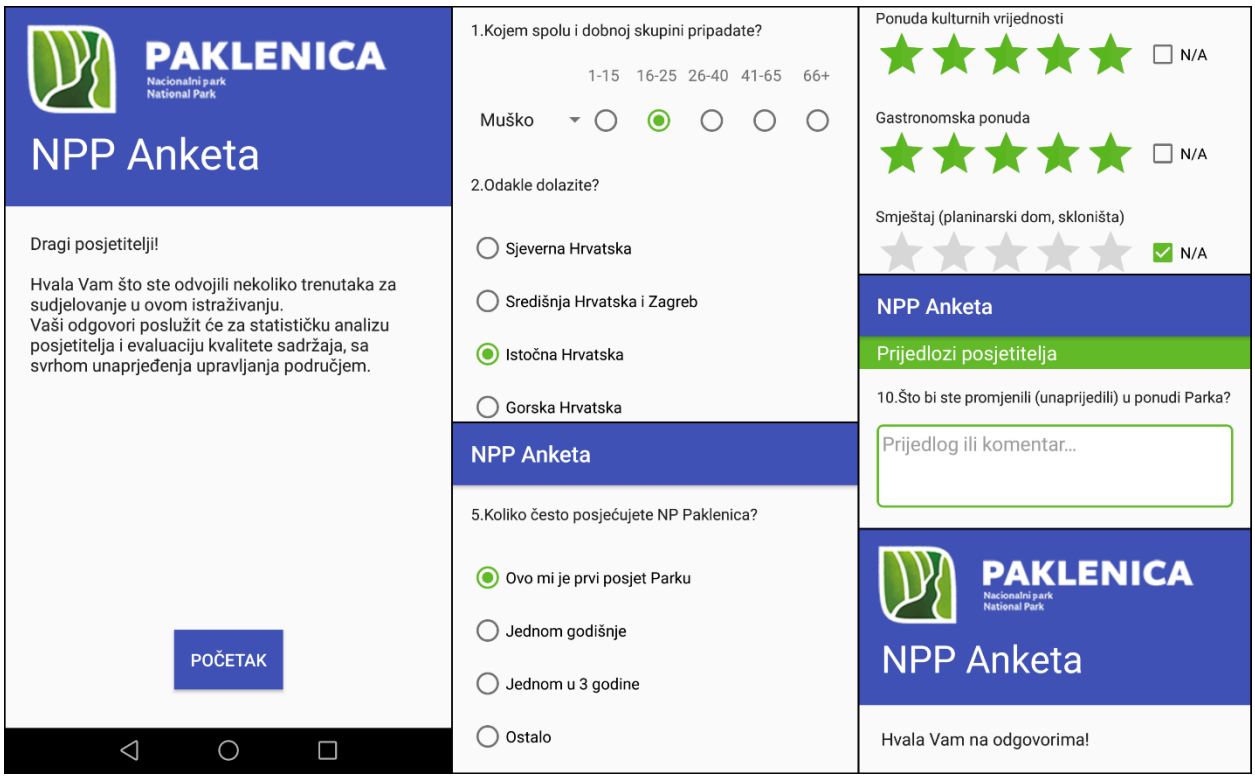
Ostale klase na sličan (ili gotovo isti) način spremaju svoje podatke (korisnikove odgovore) u bazu. Ovime je aplikacija uspješno povezana s bazom te je njena funkcija u potpunosti realizirana. Ocjena ispravnosti i funkcionalnosti aplikacije određena je njenim testiranjem.

3.4. Testiranje aplikacije

Testiranje rada aplikacije je izvršeno na nekoliko uređaja – dva virtualna Android uređaja (AVD – *Android Virtual Device*) kreirana pomoću Android emulatora (*Nexus 4, Nexus 9 tablet*) te stvarnom Android uređaju – Huawei P9 Lite. U nastavku su prikazi zaslona s tih uređaja prilikom rada aplikacije. Također, obavljeno je i testiranje komunikacije aplikacije i Firebase-a, odnosno spremanje podataka iz aplikacije u bazu (Sl. 3.27.).

3.4.1. Testiranje na uređaju Huawei P9 Lite

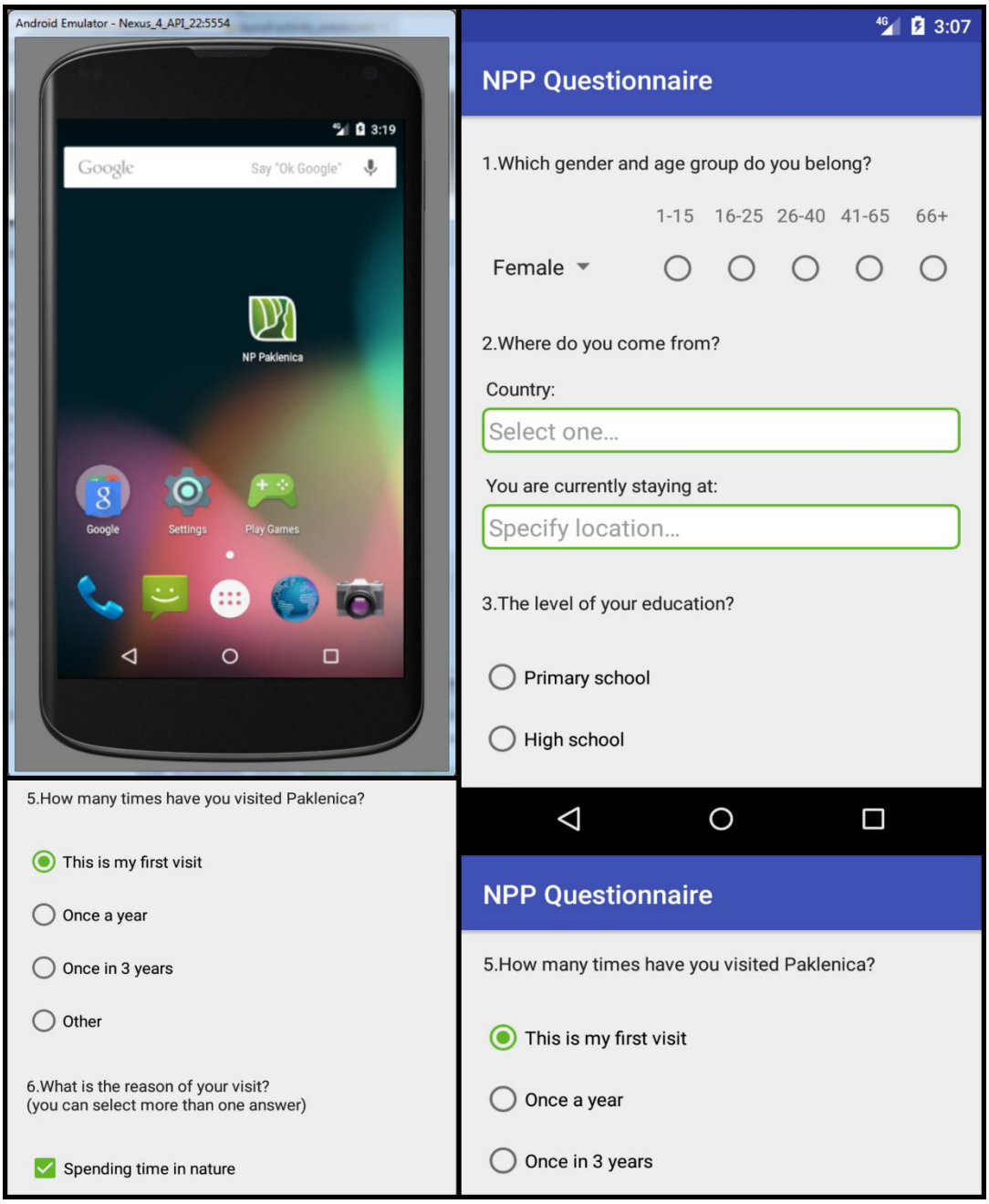
- Rezolucija: 1920x1080 (424 ppi)
- Zaslon: 5.2"
- Inačica sustava Android: 7.0 (Nougat)



Sl. 3.24. *Prikazi zaslona uređaja Huawei P9 Lite pri radu aplikacije*

3.4.2. Testiranje na uređaju Nexus 4 (AVD)

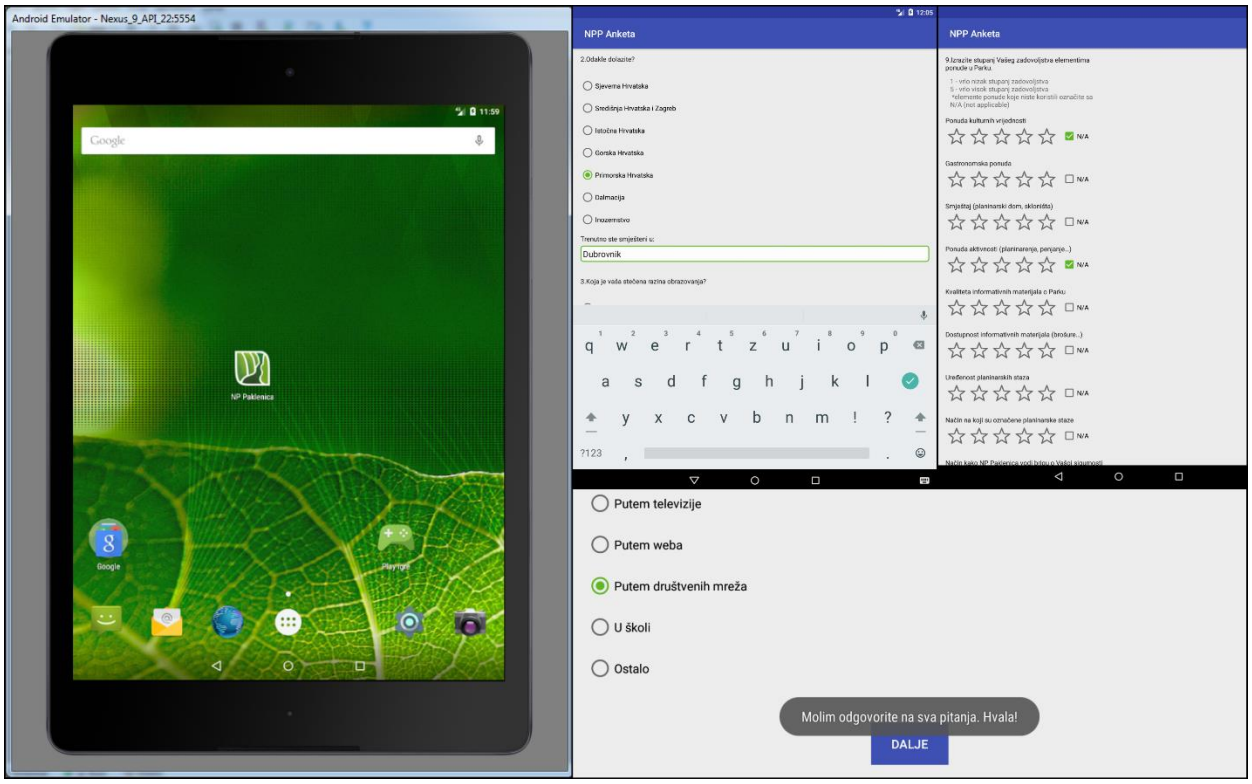
- Rezolucija: 1280x768 (318 ppi)
- Zaslon: 4.7"
- Inačica sustava Android: 5.1 (Lollipop)



Sl. 3.25. *Prikazi zaslona uređaja Nexus 4 pri radu aplikacije*

3.4.3. Testiranje na uređaju Nexus 9 (AVD)

- Rezolucija: 2048x1536 (281 ppi)
- Zaslon: 8.9"
- Inačica sustava Android: 5.1 (Lollipop)



Sl. 3.26. *Prikazi zaslona uređaja Nexus 9 (tablet) pri radu aplikacije*

3.4.4. Testiranje spremanja podataka (Firebase)

Aplikacija u stvarnom vremenu sprema podatke – korisnikove odgovore na pitanja u *Firebase* bazu podataka – koja je povezana s aplikacijom preko Google računa nppaklenicadb@gmail.com. Navedeni postupak spremanja podataka prikazan je na slici 3.21.

Sl. 3.27. *Prikaz spremanja podataka u bazu*

Ovime je potvrđena uspješnost realizacije Android aplikacije uz ispravnu funkcionalnost te je aplikaciju moguće praktično primijeniti kao oblik anketiranja posjetitelja Nacionalnog Parka Paklenica.

4. ANALIZA KORISNOSTI APLIKACIJE

Primjena informacijsko komunikacijskih tehnologija (engl. ICT) postala je trend u turizmu u novije vrijeme, budući da je to gospodarska grana u kojoj pravovremena dostupnost informacije igra veliku ulogu, a korištenje različitih aplikacija (web aplikacije, aplikacije za mobilne uređaje i slično) omogućuju upravljanje informacijama na dinamičniji i atraktivniji način koji donosi određene prednosti i pozitivne promjene poslovanja u turizmu.

Nacionalni park Paklenicu, kao jedan od najvećih nacionalnih parkova u Republici Hrvatskoj, godišnje posjeti preko 80.000 turista, koristeći pritom razne usluge koje Park nudi – planinarenje, penjanje, razgledanje prirodne i kulturne baštine, ugostiteljska ponuda i brojne druge aktivnosti i usluge u sklopu parka.[7] Kako bi se osigurala što bolja ponuda i razina kvalitete pruženih usluga potrebno je od posjetitelja – korisnika usluga, dobiti povratnu informaciju putem anketiranja. Anketiranje je način prikupljanja podataka kojim se, pomoću unaprijed sastavljenog upitnika, dobiva obavještenje o obilježjima statističkog skupa [8] – u ovom slučaju, aplikacijom se dobivaju informacije o kvaliteti pruženih usluga radi unapređivanja svih segmenata usluga koje Nacionalni park Paklenica pruža. Tako, na primjer, Park dobiva informaciju o zadovoljstvu posjetitelja kulturnom i gastronomskom ponudom parka, opsegu dostupnih aktivnosti, sigurnosti i čistoći unutar parka, ljubaznosti zaposlenika, cijeni ulaznica, parkingu i slično. Na temelju podataka koje aplikacija daje, Park može pravovremeno poduzeti odgovarajuće mjere kako bi podigao kvalitetu određene usluge, uvesti preraspodjelu osoblja na pojedinim radnim zadacima ili primijeniti nove ideje pružanja raznovrsnijih usluga. Također, analizom prikupljenih podataka putem aplikacije, moguće je dobiti informacije o strukturi turista koji posjećuju Park (odakle su, područje zanimanja, dob) te prilagoditi ponudu i poslovanje određenim zahtjevima (npr. uporaba određenog stranog jezika na natpisima u Parku, prilagođenost ponude i aktivnosti starijim ili mlađim osobama i slično). Njena korisnost se u konačnici ogleda kvalitetnijom uslugom Parka te kao rezultat toga, bolja posjećenost i bolje poslovanje.

Dakle, kako bi se optimiziralo poslovanje u turizmu i ispravile pogreške u radu, primjena ovakve aplikacije koja će anketirati korisnike usluga, na nov i zanimljiv način uz smanjenje troškova, pridonijeti inoviranju i praćenju trendova te pritom odgovoriti na zahtjeve turista, je neophodna.

5. ZAKLJUČAK

Sve veći tehnološki napredak današnjice znači i primjenu raznih tehnologija u gospodarstvu ali i svakodnevnom životu radi olakšavanja i ubrzavanja određenih procesa, usluga i zadataka. Upravo je to cilj primjene informacijsko komunikacijskih tehnologija – olakšati izvođenje određene aktivnosti u bilo kojoj grani gospodarstva ili području života. Tematika ovog završnog rada nedjeljivo je vezana upravo za primjenu informacijsko komunikacijskih tehnologija u turizmu – dakle izrada i uporaba aplikacije u svrhu kontrole kvalitete pruženih usluga. Aplikacija je realizirana u razvojnom okruženju Android Studio koristeći znanja stečena samostalnim radom ali i znanja stečena na kolegijima bliske tematike (*Programiranje, Baze Podataka*) – Java programski jezik, JSON, XML, rad sa Android operacijskim sustavom i razvijanje aplikacije unutar Android Studia.

Svrha aplikacije je anketiranje posjetitelja o kvaliteti usluge pri posjeti nacionalnog parka Paklenica, te kao takva zamjenjuje (ne nužno u potpunosti) dosadašnji proces i način anketiranja putem anketnih listova, uz očigledne prednosti brzine, jednostavnosti, efikasnosti i ekonomičnosti - ušteda na resursima (ljudskim i financijskim), koji su bili nužni kod starog načina anketiranja.

Testiranjem aplikacije je potvrđena njena ispravnost i funkcionalnost, te je omogućeno preuzimanje aplikacije sa *Google Play* trgovine za Android mobilne uređaje, koji su najzastupljeniji na ovim područjima, čime je osigurana dostupnost aplikacije odnosno ankete. Primjena informacijsko komunikacijskih tehnologija, odnosno u ovom slučaju, uporaba Android aplikacije za anketiranje posjetitelja ima za svrhu, između ostalog, povećanje broja korisnika ankete (što znači bolju povratnu informaciju o kvaliteti ponude i sadržaja parka, mogućnosti poboljšanja i slično), nudeći posjetiteljima zanimljiviji i moderniji način anketiranja te ujedno bržu i jednostavniju analizu prikupljenih odgovora.

Aplikaciju je moguće i dodatno poboljšati dodavanjem podrške i za druge jezike osim hrvatskog i engleskog, moguće je i poboljšanje dizajna, izrada sučelja prilagođenog tablet uređajima, izrada baze za spremanje podataka aplikacije koja bi više odgovarala postupku analize prikupljenih anketa ili izrada aplikacije koja bi sukladno aplikaciji za anketiranje, vršila analizu trenutno prikupljenih podataka ispisujući rezultate u odgovarajućem formatu.

LITERATURA

[1] Krajci, I.; Cummings, D.: „ History and Evolution of the Android OS“, Android on x86, Apress, 2013., str. 1-8.

[2] Cristopher Jones, Android Architecture, http://www.thejonesjournal.com/android/what-is-android-architecture (stranica posjećena: 21. lipnja 2017.)

[3] Android Developers, https://developer.android.com/studio/intro/index.html, (stranica posjećena 24. lipnja 2017.)

[4] Chris Esplin: „What is Firebase?“, https://howtofirebase.com/what-is-firebase-fcb8614ba442 (stranica posjećena: 29. lipnja 2017.)

[5] Nacionalni park Kornati, Online Anketa, http://www.np-kornati.hr/hr/turizam/2016-06-20-17-20-39 (stranica posjećena 22. lipnja 2017.)

[6] JSON.org, Introducing JSON, http://www.json.org/ (stranica posjećena 24. lipnja 2017.)

[7] Nacionalni park Paklenica, http://www.np-paklenica.hr/index.php/park/o-parku, (stranica posjećena 21. lipnja 2017.)

[8] Brajdić, I. (2002). VREDNOVANJE ANKETNIH UPITNIKA SA STAJALIŠTA ISPITANIKA-MENADŽERA U TURIZMU. Tourism and hospitality management, 8(1-2), 65-78. Preuzeto s http://hrcak.srce.hr/181670

SAŽETAK

Tema ovog rada je realizacija aplikacije koja za svrhu ima zamijeniti standardno anketiranje posjetitelja Nacionalnog parka Paklenica putem anketnog upitnika te učiniti cijeli postupak kontrole kvalitete pruženih usluga jednostavnijim i bržim te korisnicima pristupačnijim, korištenjem informacijsko komunikacijskih tehnologija. Na temelju anketnih pitanja dobivenih od strane Nacionalnog parka Paklenica, određen je izgled aplikacije. Aplikacija je izrađena u razvojnom programskom okruženju *Android Studio,* a baza za spremanje korisnikovih odgovora na anketna pitanja kreirana je pomoću Google-ove mobilne platforme *Firebase*. Aplikacija prikuplja odgovore korisnika na anketna pitanja, na temelju kojih je moguće napraviti analizu trenutne ponude Parka, kontrolirati kvalitetu pruženih usluga, i upravljati informacijama na dinamičniji i aktraktivniji način uz prednosti brzine, jednostavnosti i ekonomičnosti. Njena korisnost se u konačnici ogleda kvalitetnijom uslugom te kao rezultat toga, bolja posjećenost Parka.

Ključne riječi: Android operacijski sustav, Android Studio, Java, JSON, Google Firebase, Nacionalni park Paklenica, anketa, turizam

ABSTRACT

Quality control of provided services in tourism by means of information and communication technology

The topic of this paper is the development of an application for the purpose of replacing the survey of Paklenica National Park through a standard questionnaire and make the entire process of controlling the quality of provided services easier, faster and more accessible to users by using information communication technologies. The layout of the application was determined by the questionnaires received from the Paklenica National Park and the application was created in the *Android Studio* development environment; a database for saving user responses to survey questions was created using the Google Mobile *Firebase* platform. The application collects user feedback, based on which it is possible to analyze the current offer of the Park, control the quality of provided services, and manage information in a more dynamic and more attractive way with the advantages of speed, simplicity and cost-effectiveness. Its usefulness is ultimately reflected in better quality of the service and, as a result, more visitors.

Keywords: Android operating system, Android Studio, Java, JSON, Google Firebase, National Park Paklenica, questionnaire, tourism.

ŽIVOTOPIS

Slaven Ivić rođen je 9. listopada 1995. godine u Munchenu, Njemačka. 2002. godine je započeo svoje osnovnoškolsko obrazovanje u Osnovnoj školi Vladimira Nazora, Gornja Dubica, Bosna i Hercegovina. Nakon završena 4 razreda, nastavlja obrazovanje u OŠ Vladimira Nazora, Odžak. U Odžaku upisuje opću gimnaziju u Srednjoj školi Pere Zečevića. Kroz osnovnoškolsko i srednjoškolsko obrazovanje sudjeluje na raznim školskim, županijskim i državnim natjecanjima iz matematike. Nakon završene srednje škole, upisuje Elektrotehnički fakultet u Osijeku gdje je 2016. bio nagrađen dekanovom nagradom kao najbolji student na drugoj godini studija.

Slaven Ivić

PRILOZI

CD

- Elektronička verzija rada (dokument u .docx i .pdf formatu)
- Android Studio projekt
- Aplikacija (.apk datoteka)