Mobilna aplikacija za provjeru znanja

Barić, Davor

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:200:914094

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-01-02**

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 preddiplomski studij računarstva

MOBILNA APLIKACIJA ZA PROVJERU ZNANJA

Završni rad

Davor Barić

Osijek, 2017.

SADRŽAJ:

1. UVOD 1

1.1. Zadatak završnog rada 1

2. TEORIJSKA PODLOGA 2

2.1. Android kao operacijski sustav 2

2.2. Android Studio 3

2.3. Programski jezik Java 4

2.4. XML u Android Studiu 5

2.5. Firebase 6

3. ANDROID APLIKACIJA ZA PROVJERU ZNANJA 7

3.1. Ideja i model 7

3.2. Aplikacija 8

3.2.1. Početni zaslon 10

3.2.2. Administracijski dio 13

3.2.3. Korisnički dio 14

3.2.4. Krajnji rezultat 19

3.3. Testiranje aplikacije 21

4. ZAKLJUČAK 22

LITERATURA 23

SAŽETAK 24

ABSTRACT 25

ŽIVOTOPIS 26

PRILOZI 27

1. UVOD

Tema završnog rada je izraditi Android aplikaciju za provjeru znanja. Kako bi studenti, učenici i svi oni koji na neki način nešto uče mogli provjeriti svoja znanja. Ovakva aplikacija je upravo takva da sami sebe ispitaju, odnosno provjere svoje znanje iz određenog područja.

Prilikom izrade aplikacije nužno je bilo koristiti znanja stečena na fakultetu iz kolegija „Programiranje 1“, „Programiranje 2“ te „Objektno orijentirano programiranje“. Za izradu aplikacije korišteno je razvojno okruženje Android Studio koje koristi objektno orijetirani programski jezik Java.

Na početku završnog rada dan je teorijski osvrt na tehnologije i jezike koji su korišteni prilikom izrade aplikacije. Nakon toga će se dati vizualni opis nastanka aplikacije, te dijelovi koda koji su sastavni dio ove mobilne aplikacije.

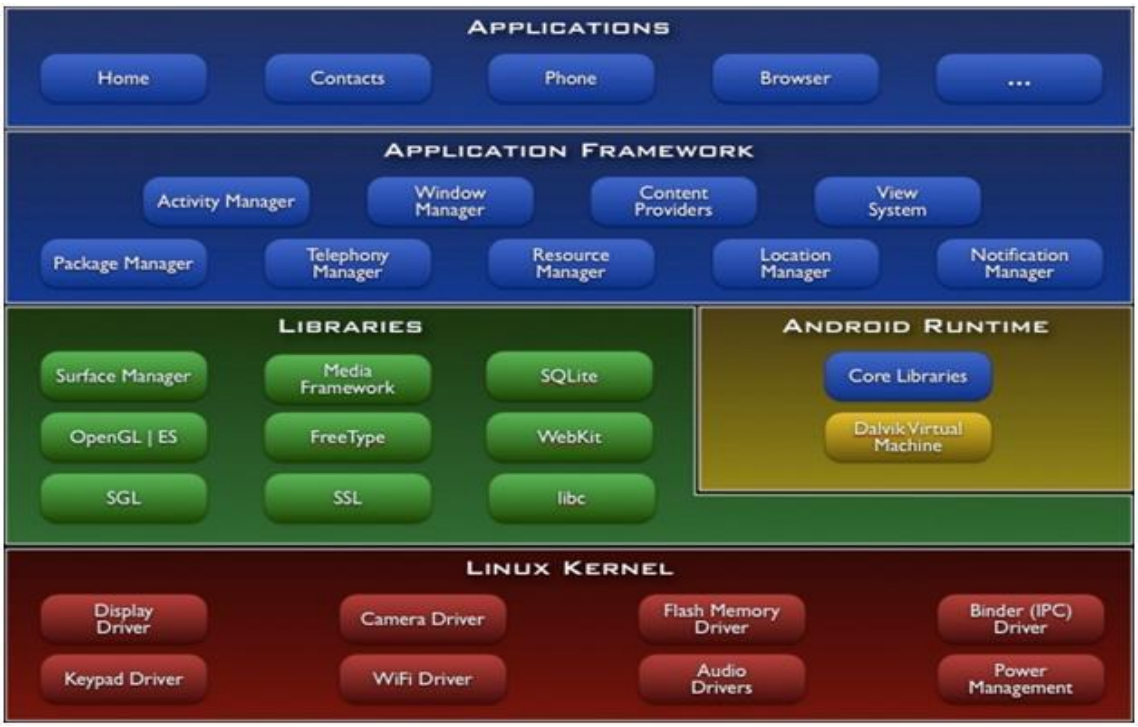
1.1. Zadatak završnog rada

Zadatak je ovog završnog rada izraditi aplikaciju za Android operacijski sustav za samoprovjeru znanja stečenih tijekom obrađenog gradiva ili gradiva koje će se tek učiti te ispitivati. Korisniku će biti ponuđen test te mogućnost odgovaranja na pitanja koje je administrator postavio. Kao programsko okruženje, za izradu aplikacije korišten je Android Studio, te je kod aplikacije pisan u programskom jeziku Java s korištenjem Firebase *realtime* baze podataka.

2. TEORIJSKA PODLOGA

2.1. Android kao operacijski sustav

Google Inc dizajnirao je prvi operacijski sustav otvorenog koda kojeg danas nazivamo Android. Android kao operacijski sustav baziran je na Linux *kernelu* te je prvotno izrađen za uređaje sa zaslonom na dodir kao što su pametni telefoni i tableti. Andy Rubin, Rich Miner, Nick Sears i Chris White su u listopadu 2003. godine osnovali Android Inc. s namjerom da razvijaju programe za pametne mobilne uređaje. Dvije godine kasnije, tvrtku Android Inc kupio je Google. Zbog svoje prilagodljivosti Android se danas sve više koristi na raznim uređajima. Primjer toga su satovi, televizori pa danas sve više i automobili. Android operacijski sustav pisan je u C/C++ programskom jeziku te je zansovan na jezgri Linux 2.6. Kao što je vidljivo na slici 2.1. jezgra Linux 2.6. je ujedno i prva razina operacijskog sustava. Iznad Linux jezgre slijede biblioteke zaslužne za pokretanje aplikacija. Nakon toga slijedi aplikacijski okvir koji sadrži sve potrebne mehanizme za pisanje aplikacija. Također aplikacijski okvir omogućava upotrebu API-ja (*eng. Application Progamming Interface*). Najviša razina operacijskog sustava je upravo ono što je vidljivo krajnjem korisniku poput ugrađenih aplikacija kao što su kalendar, SMS program, web preglednik pa sve do aplikacija koje korisnik može pronaći na Android Marketu. Današnji broj aplikacija na Android Marketu je prerasto 250 000.

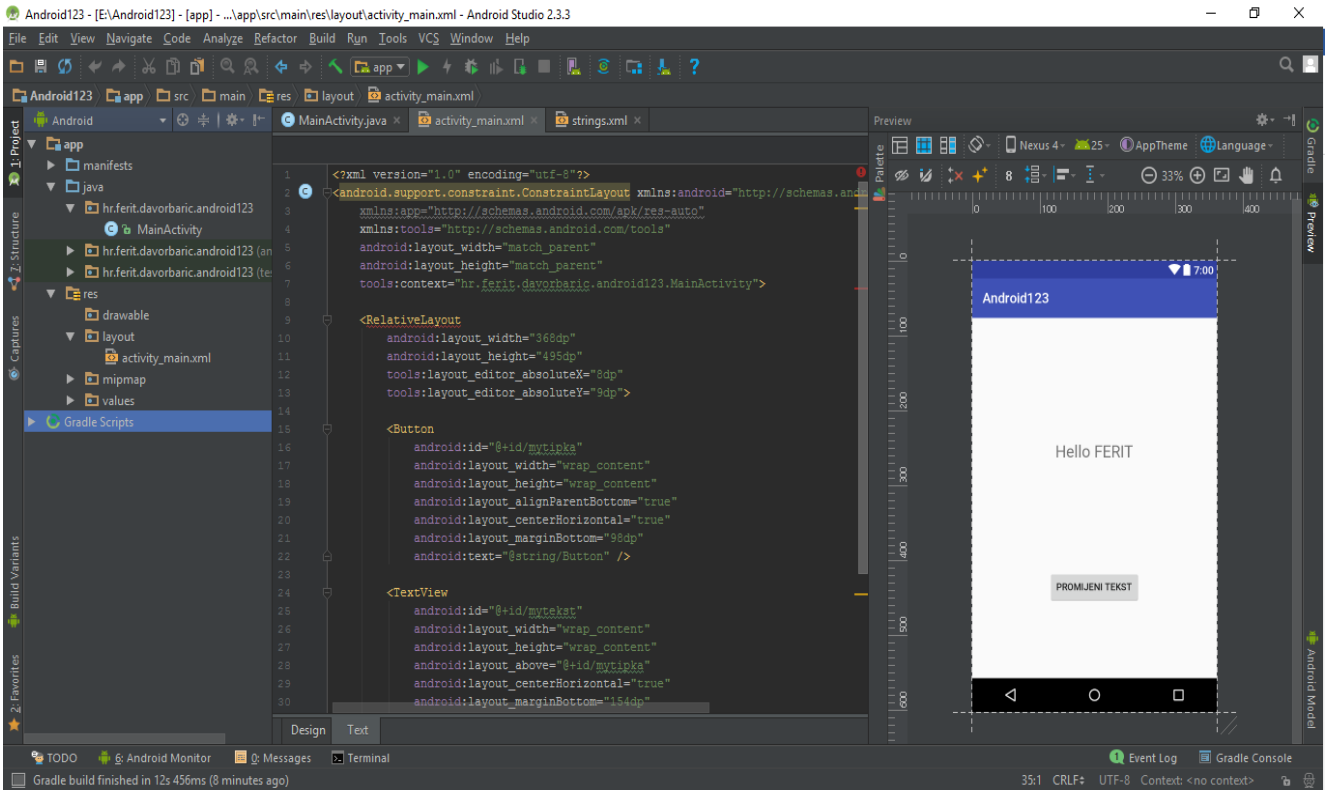


Sl. 2.1. Građa Android operacijskog sustava

2.2. Android Studio

Android Studio je relativno novo integrirano programsko okruženje za ravoj Android aplikacija. Sam Android Studio najavljen je 16.svibnja 2013. na Google I/O konferenciji, no prva stabilna verzija je objavljena nakon malo više od godinu dana, točnije u prosincu 2014. godine, počevši od verzije 1.0. Android Studio, koji je baziran na JetBrain's IntelliJ IDEA softveru, posebno je dizajniran za razvoj mobilnih aplikacija odnosno Android aplikacija. Android Studio je zamijenio Eclipse koji je bio donedavno primarno razvojno okruženje za Android. Neke od značajki Android Studia su: mogućnost jednostavnog kreiranja izgleda korisničkog sučelja pomoću tzv. *drag and drop* načina, podrška za *Gradle build* sustav, ugrađena podrška za Google *Cloud* platformu, *Android Virutal Device* (emulator) koji služi za pokretanje aplikacije na virtualnom uređaju te mnoge druge značajke.

Android Studio je počeo sa verzijom 1.0. za koju su bile potrebne nešto slabije specifikacije računala na kojem pokrećemo razvojno okruženje. Danas je Android Studio već dosegao verziju 2.3.3. te su potrebne znatno veće specfikacije računala. Za usporedbu, verzija 1.0 mogla se pokretati sa minimalno 3 GB radne memorije iako je 4 GB bilo preporučeno, dok verzija 2.0 preporučava 8 GB radne memorije te dodatnih 1GB za Android emulator. Za verziju 1.0 bilo je dovoljno imati 1 GB prostora na disku za Android SDK (eng. *Software Development Kit)* dok je danas za verziju 2.0 potrebno imati 500 MB za Android Studio te barem 1,5 GB za Android SDK (eng. *Software Development Kit).* Ovo razvojno okruženje je moguće pokretati na operacijskim sustavima kao što su Windows 7 ili noviji, Mac OS X 10.9.5 ili noviji, GNOME ili KDE desktop.



Sl. 2.2. Izgled sučelja Android Studia

2.3. Programski jezik Java

Početkom 1990-tih tim predvođen Jamesom Goslingom u tvrtci Sun Microsystems razvio je objektno orijentirani programski jezik nazvan Java. Njihova ideja bila je napraviti programski jezik koji će biti baziran na C++-u, ali sa nešto jednostavnijom sintaksom i jednostavnijom kontrolom memorije. Java je danas jedan od najkorištenijih programskih jezika jer broji u svijetu oko devet milijuna korisnika. Kao što je i rečeno, programski jezik Java bazirana je na C++ te kao osnovni koncepti koriste se klase i objekti, odnosno sav programski kod koji se piše je napisan upravo unutar klasa ili razreda. Osnovni tipovi podataka koje nalazimo u Javi svakako su int, boolean, byte, string i slični. Iako je Java bazirana na C++, memorija nekog objekta automatski se čisti što nije slučaj kod C++ gdje moramo koristiti naredbu *delete* kako bismo izbrisali stvoreni objekt.

2.4. XML u Android Studiu

XML predstavlja kraticu za jezik za označavanje podataka odnosno *EXtensible Markup Language.* Prvotna ideja ovog jezika je bila napraviti takav program gdje će se korisni sadržaj nalaziti u uokvirenim odgovarajućim oznakama, te isto tako stvoriti jezik koji bi bio jednostavan za čitanje najprije ljudima, a zatim i računalnim programima. XMLov format velike sličnosti ima s oznakama HTML jezika. Danas XML pronalazi primjenu u mnogim namjenama kao što su: razmjena podataka, pohrana podataka, izradu novih jezika za označavanje, izgled sučelja i mnogih drugih. Kao jezik, XML je standardiziran te se za njegovu standardizaciju brine Word Wide Web Consortium.

XML predstavlja vizualni izgled aplikacije u Android Studiu, te pomoću njega definiramo sve one stvari koje su korisnicima aplikacije vidljive kada pokrenu aplikaciju. XML definira čitavi izgled aplikacije od boja, pozadina, vrsta tekstova, gumbova, veličine fonta pa sve do postupka na koji način će *layouti* biti postavljeni na zaslonu.

Primjer XML naredbi:

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="prijava"
    android:textSize="20dp"
    android:layout_centerInParent="true"
    android:id="@+id/login_btn"  />
```

Iz primjera je vidljivo da svaki sadržaj počinje s uokvirenom odgovarajućom oznakom <, a završava se />. Ovo je samo jedan primjer kreiranja gumba gdje mu je dodan tekst koji će pisati na njemu, a to je „prijava“ te još neka svojstva poput veličine slova, njegovog položaja na *layoutu,* visine i širine gumba te njegovog ID preko kojeg pristupamo njemu u java kodu.

2.5. Firebase

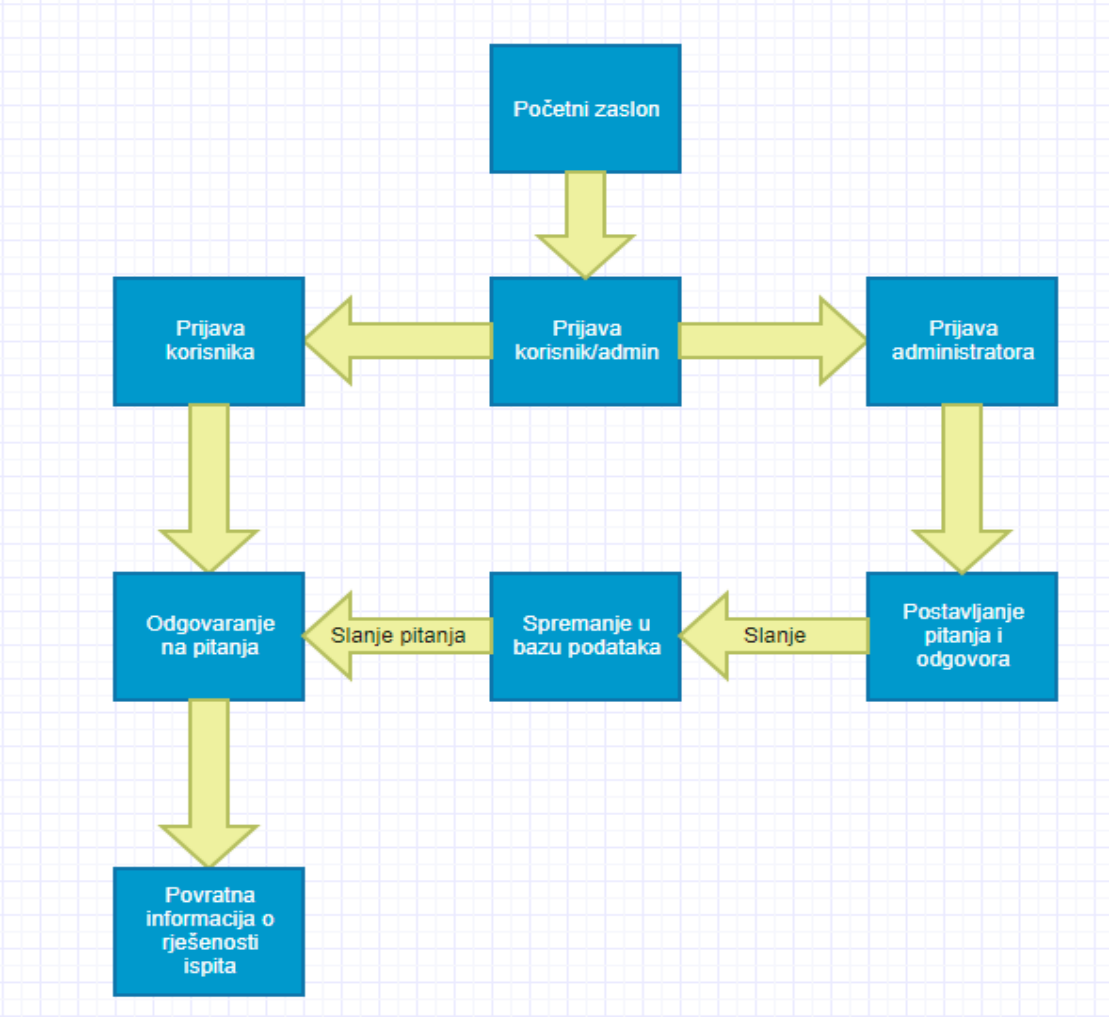
Firebase predstavlja platformu za razoj mobilnih i web aplikacija koja je razvijena od strane tvrtke Firebase Inc. 2011. godine. Tu platformu je 2014. godine otkupio Google. Neke od usluga koje koristi Firebase su: Firebase Analytics, Firebase Realtime Database, Firebase Storage, Firebase Hosting i još mnogo drugih. U ovom završnom radu kao uslugu ćemo koristiti Firebase Realtime Database odnosno bazu podataka u stvarnom vremenu. Firebase Realtime Database omogućuje pohranu i sinkornizaciju podataka između korisnika i uređaja u stvarnom vremenu. Dobra strana ovakve baze podataka je to što se ažurirani podaci sinkroniziraju sa povezanim uređajima u vrlo brzom vremenu i također ostaju dostupni ako nestane aplikacija. Firebase je integriran alat u Android Studio te ga nalazimo pod *tools* u Android Studio. Potrebno je imati Google korisnički račun kako bi mogli pristupiti i pokreniti Firebase.

3. ANDROID APLIKACIJA ZA PROVJERU ZNANJA

3.1. Ideja i model

Mnogi fakulteti koriste neke od metoda ispitivanja kako bi provjerili znanje i pripremljenost svojih studenta. Neka ispitivanja svode se na znanje stečeno prijašnjim učenjem dok neka ispitivanja očekuju da se studenti pripreme za nešto novo što će tek naučiti, te da im samim time bude lakše savladati novo gradivo. Studentima, učenicima i ostalima koje se ispituje ponekad je teško pronaći motivaciju kako bi sami sebe preispitali. Danas gotovo svatko koristi pametni telefon te provodi veliku većinu svoga vremena koristeći ga. Tako je nastala ideja da se realizira aplikacija za pametni mobilni uređaj kojim će studenti moći provjeriti svoje znanje prije nego što pristupe ispitivanju. Takvi načini provođenja ispitivanja, odnosno samoprovjere kod korisnika, stvaraju veću želju za učenjem te ostvaruju bolje efekte jer odgovaraju na postavljena pitanja i stječu više znanja, nego učenjem nekih nepotrebnih stvari. Korisnik će po završetku samoprovjere dobiti okvirnu informaciju koji je njegov rezultat u odnosu na broj pitanja koja su mu postavljena, te će moći procijeniti svoju spremnost za ispit koji ga očekuje.

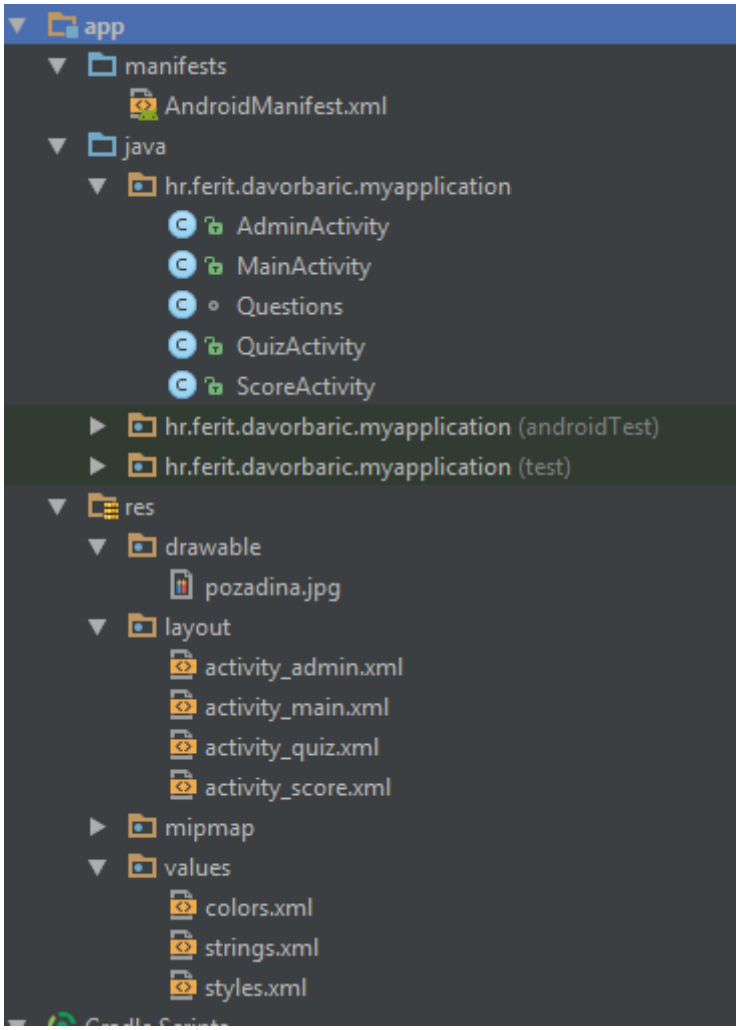
Svaka aplikacija ima svoj model kako će izgledati. Tako ćemo prikazati u obliku dijagrama slijedni prikaz odvijanja ove aplikacije. Prvi dio aplikacije je svakako prijava. Aplikacija nudi mogućnost prijave za korisnika ili za administratora putem odgovarajućih korisničkih imena i lozinki ovisno o tome tko pristupa aplikaciji. U slučaju da pristupa administrator, on će imati mogućnost na svome sučelju postavljati pitanja, odgovore te točan odgovor koji će se zatim poslati u bazu i tamo spremiti. Ulaskom u korisnikovo sučelje pred korisnika se otvara ispit s pitanjima koje je administrator postavio te na koje korisnik odgovora na temelju svoga znanja. Na kraju aplikacije korisnik dobiva povratnu informaciju o tome je li položio ovaj „pripremni“ ispit ili nije.



Sl. 3.1.Slijedni prikaz aplikacije

3.2. Aplikacija

Svaka aplikacija ima svoja glavna obilježja i datoteke od kojih se sastoji. Tako u Android Studiu imamo nekoliko mapa kao što su manifest, resursi i java, što je vidljivo na slici 3.2. Manifest datoteka je pisana XML opisnim jezikom te su u toj datoteci definirane sve aktvnosti, dozvole, ikona i verzija aplikacije te struktura aplikacije. Gotovo svaki element aplikacije je definiran u ovoj datoteci. U mapi „resursi“ možemo naći sve slike i druge slične podatke vezane za izgled. Također, u mapi „resursi“ su definirani *layouti* koji su zaslužni za izgled Android aplikacije.



Sl. 3.2. Sadržaj projekta u Android Studiu

U mapi „java“ možemo vidjeti da je definirano pet klasa i to „AdminActivity“, „MainActivity“, „QuizActivity“, „ScoreActivity“ te „Questions“ u kojima je pisan java kod. Nešto kasnije će funkcionalnost svih tih klasa biti pojedinačo objašnjena. U mapi „res“, koja predstavlja resurse, imamo četiri podmape. U prvoj, odnosno mapi „drawable“, spremljena je pozadina koja je korištena u ovoj aplikaciji. U drugoj su spremljeni svi *layouti* korišteni u ovoj aplikaciji, *layouti* koji zapravo predstavljaju svaki zaslon zasebno. U mapi „mipmap“ je spremljena ikona koju koristi aplikacija. Posljednja podmapa je „values“ u kojoj možemo naći XML datoteke koje definiraju izgled ove aplikacije poput boja, stilova teksta i ostalih.

3.2.1. Početni zaslon

Kao i kod svake druge aplikacije prilikom pokretanja aplikacije otvara se početni zaslon kao što je vidljivo na slici 3.3. Na zaslonu je korisniku preko dva tekstualna okvira omogućen unos korisničkog imena (id_student) i pripadajuće lozinke (pw_student). Sučelje također sadrži i gumb „Prijava“ (login_btn) kojim se ulazi u administracijsko ili korisničko sučelje ovisno o tome koji su podaci upisani u gornja dva tekstualna okvira.

Sl. 3.3. Početni zaslon aplikacije

Kod za izgled početnog zaslona aplikacije pisan je u „activity_main.xml“ datoteci. U svim XML datotekama bilo je potrebno pisati kod, a ne koristiti tzv. *drag and drop* način slaganja elemenata na zaslon, kako bi se sadržaj na zaslonu isto prikazivao na različitim veličinama ekrana mobilnih uređaja. Vidljivo je da je korišten relativni *layout*. Kod takvog tipa *layouta* elementi se slažu u odnosu jedan na drugi i mogu biti postavljeni bilo gdje na zaslonu, dok kod linearnog elemente slažemo jedan ispod drugoga. Svakom elementu su pridružena poneka svojstva poput veličine teksta, stila unosa u polje, pozicije u odnosu na drugi element te njegovog ID preko kojeg pristupamo njemu u „MainActivity.java“ klasi.

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@drawable/pozadina"/>

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/id_student"
        android:hint="Korisnicko ime"
        android:textSize="24dp"
        android:layout_above="@+id/pw_student"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="24dp"/>

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/pw_student"
        android:textSize="24dp"
        android:hint="Lozinka"
        android:inputType="textPassword"
        android:layout_above="@+id/login_btn"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="40dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="prijava"
        android:textSize="20dp"
        android:layout_centerInParent="true"
        android:id="@+id/login_btn"/>

</RelativeLayout>
```

Sl. 3.4. XML kod početnog zaslona aplikacije

U „activity_main.xml“ datoteci dan je samo vizualni izgled početnog zaslona. Kako bi taj zaslon bio funkcionalan, potrebno je napisati java kod u „MainActivity.java" klasi koji će obaviti određenu radnju koja je potrebna, a dan je na slici 3.5.

```
public class MainActivity extends AppCompatActivity {

    private Button login_btn;
    private EditText studentIdEt, studentPwEt;

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

        studentIdEt= (EditText) findViewById(R.id.id_student);
        studentPwEt= (EditText) findViewById(R.id.pw_student);
        login_btn = (Button) findViewById(R.id.login_btn);

        login_btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (studentIdEt.getText().toString().equals("student") && studentPwEt.getText().toString().equals("student")){
                    startActivity(new Intent(MainActivity.this, QuizActivity.class));
                }else if(studentIdEt.getText().toString().equals("admin") && studentPwEt.getText().toString().equals("admin")){
                    startActivity(new Intent(MainActivity.this, AdminActivity.class));
                }else {
                    Toast.makeText(MainActivity.this, "Pogresni korisnicki podaci", Toast.LENGTH_LONG).show();
                }

                studentIdEt.setText("");
                studentPwEt.setText("");
            }
        });
    }
}
```

Sl. 3.5. Java kod početnog zaslona

Kako bi se obavila radnja prijave korisnika ili administratora, potrebno je postaviti osluškivač na gumb „prijava“ kojem smo dodijelili ID login_btn. Postavljamo osluškivač „setOnClickListener()“ na gumb login_btn te pozivamo metodu „onClick()“ u kojoj definiramo što će se dogoditi pritiskom na gumb „prijava“. Ako je tekst koji korisnik unese u polje gdje piše korisničko ime i lozinka jednak „admin“, pokreće se nova aktivnost „startActvitiy (new Intent (MainActivity.this, AdminActivity.class));“, odnosno prelazimo na novi zaslon i otvara se „AdminAcitvity.class“ u kojemu su definirane radnje za administracijski dio. Unosom lozinke i korisničkog imena, koje je jednako „student“, pritiskom na gumb „prijava“ prelazi se na novu aktivnost „QuizActivity.class“. U slučaju da korisnik unese pogrešnu lozinku ili korisničko ime, dobit će obavijest o tome da je upisao pogrešne korisničke podatke.

3.2.2. Administracijski dio

Nakon upisivanja korisničkog imena „admin“ te lozinke „admin“ pritiskom na gumb „prijava“ otvara se administracijsko sučelje. U ovome dijelu administratoru je omogućen unos pitanja, odgovora te točnog odgovora u Firebase bazu podataka. Izgled administracijskog sučelja opisan je „activity_admin.xml“ datotekom koja je opisana na sličan način kao i početni zaslon. Sastoji se od sedam tekstualnih okvira (rb_pitanje_et, pitanje_et, odga_et, odgb_et, odgc_et, odgd_et te odgovor_et) te jednoga gumba (dodaj_btn). Definirana je također i klasa „Questions.class“ koja sadrži konstruktore, „get()“ i „set()“ za pitanje, odgovore i točan odgovor preko kojih vršimo „komunikaciju“ između administratora, baze podataka i na kraju korisnika.

Sl. 3.6. Izgled administracijskog sučelja

U „AdminActivity.java“ klasi potrebno je bilo definirati objekt adminQuestions klase Questions kako bi preko njega mogli pitanja i odgovore upisane u administracijskom dijelu proslijediti bazi podataka. Na gumb dodaj_btn postavljen je osluškivač „setOnClickListener()“,

te se prilikom klika admina na taj gumb poziva metoda „onClick()“ koja uzima tekst upisan u sedam tekstualnih okvira te ih sprema u objekt adminQuestions klase „Questions.class“. Nakon toga se sva vrijednost koja je zapisana u objekt adminQuestions, odnosno u sedam teksutalnih okvira, šalje u Firebase bazu podataka. Po završetku slanja prvog pitanja, odgovora i točnog odgovora, polja se ispražnjavaju te su slobodna za unos drugog pitanja.

```
private DatabaseReference databaseReference;
private EditText rb_pitanje_et, pitanje_et, odga_et, odgb_et, odgc_et, odgd_et, odgovor_et;
private Button dodaj_btn;
private Questions adminQuestions;

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

    rb_pitanje_et = (EditText) findViewById(R.id.rb_pitanje_et);
    pitanje_et = (EditText) findViewById(R.id.pitanje_et);
    odga_et = (EditText) findViewById(R.id.odga_et);
    odgb_et = (EditText) findViewById(R.id.odgb_et);
    odgc_et = (EditText) findViewById(R.id.odgc_et);
    odgd_et = (EditText) findViewById(R.id.odgd_et);
    odgovor_et = (EditText) findViewById(R.id.odgovor_et);
    dodaj_btn = (Button) findViewById(R.id.dodaj_btn);


    dodaj_btn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            adminQuestions = new Questions(pitanje_et.getText().toString(), odga_et.getText().toString(),odgb_et.getText().toString(),
                    odgc_et.getText().toString(),odgd_et.getText().toString(),odgovor_et.getText().toString());
            databaseReference = FirebaseDatabase.getInstance().getReference();
            databaseReference.child(rb_pitanje_et.getText().toString()).setValue(adminQuestions);

            rb_pitanje_et.setText("");
            pitanje_et.setText("");
            odga_et.setText("");
            odgb_et.setText("");
            odgc_et.setText("");
            odgd_et.setText("");
            odgovor_et.setText("");
        }
    });
}
}
```

3.7. Java kod administracijskog sučelja

3.2.3. Korisnički dio

Upisivanjem korisničkog imena „student“ te lozinke „student“ i pritiskom gumba „prijava“ na početnom zaslonu, otvara se korisnički dio. Izgled korisničkog sučelja opisan je u „activity_quiz.xml“ datoteci. Datoteka sadrži jedan tekstualni okvir u koji se postavlja pitanje te četiri gumba na koje postavljamo odgovore. Korisnik odgovara, te putem *Toast* obavijesti na dnu zaslona dobiva infomaciju je li odgovor točan ili netočan.

Sl. 3.8. Izgled korisničkog sučelja

Kako bi svaki korisnik koji pristupa aplikaciji dobio različita pitanja, bilo je potrebno napraviti metodu koja će od svih pitanja koja su u bazi pohranjena odabrati pet i to slučajnim odabirom. Kao što je vidljivo iz koda, bilo je potrebno napraviti dvije *for* petlje, i to prvu takvu da uzima deset pitanja iz baze, zatim tih deset pitanja „izmiješa“ preko metode „Collections.shuffle()“. Na kraju od tih promiješanih deset pitanja, druga *for* petlja uzima onih pet kojih će se proslijediti u metodu „queryDatabase()“, odnosno Firebase bazu podataka na koje će korisnik na kraju i odgovarati. Funkcionalnost metode „queryDatabase()“ je objašnjena kasnije.

```
private Button a_btn, b_btn, c_btn, d_btn;
private TextView questionTv;
private int rez = 0;
private int rMaxQ = 10;
private int rSelectedQ = 5;
private DatabaseReference databaseReference;
private Questions questions;
private int i = 1;

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

    questionTv = (TextView) findViewById(R.id.pitanje_tv);
    a_btn = (Button) findViewById(R.id.a_btn);
    b_btn = (Button) findViewById(R.id.b_btn);
    c_btn = (Button) findViewById(R.id.c_btn);
    d_btn = (Button) findViewById(R.id.d_btn);


    ArrayList<Integer> rmrq = new ArrayList<Integer>();
    final ArrayList<Integer> rsrq = new ArrayList<Integer>();
    for (int i = 1; i < rMaxQ; i++) {
        rmrq.add(i);
    }
    Collections.shuffle(rmrq);

    for (int i = 0; i < rSelectedQ; i++) {
        rsrq.add(rmrq.get(i));
    }

    queryDatabase(rsrq.get(0));
```

Sl. 3.9. Java kod odabira pitanja

Metoda „QueryDatabase()“ prima cjelobrojni podatak i to redni broj pitanja koji je izabran „*radnom*“ metodom odabira, tj. slučajnim odabirom preko metode „Collections.shuffle()“. U objektu questions klase Questions dohvaćamo svu vrijednost koja je pohranjena u konstruktorima u klasi „Questions.class“ te na svaki gumb (a_btn, b_btn, c_btn, d_btn) se postavlja taj isti tekst koji je dohvaćen iz klase „Questions. class“ za svaki element posebno.

```
private void queryDatabase(int qNum){
    databaseReference = FirebaseDatabase.getInstance().getReference().child(String.valueOf(qNum));
    databaseReference.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            questions = dataSnapshot.getValue(Questions.class);
            questionTv.setText(questions.getQ());
            a_btn.setText(questions.getA());
            b_btn.setText(questions.getB());
            c_btn.setText(questions.getC());
            d_btn.setText(questions.getD());
        }
```

Sl. 3.10. Java kod metode „queryDatabese()“

Na svaki od odgovora odnosno gumbova potrebno je postaviti osluškivač „setOnClickListener()“ te kada se dogodi klik izvršava se metoda „onClick()“ u kojoj se prvo provjerava je li zadnje pitanje, ako je, provjerava se točnost odgovora pod „a“ putem metode „checkAnswers()“ koja je opisana nešto kasnije. Zatim se pokreće nova aktivnost, odnosno novi zaslon i to „ScoreAcitvity.class“ tj. zaslon sa rezultatom i to preko „startAcitvity(intent)“. U objekt „bundle“ spremamo rezultat, te ćemo preko objekta „bundle“ poslati rezultat na zaslon sa rezultatom. U slučaju da to nije posljednje pitanje, provjerava se prvo točnost odgovora pod „a“, a zatim se postavlja novo pitanje koje je spremljeno u bazu podataka i tako sve dok se ne postavi pet pitanja. Kod je potrebno ponoviti četiri puta za sva četiri gumba (a_btn, b_btn, c_btn, d_btn). Na slici 3.11. i 3.12 je dan kod za dva gumba i to a_btn i c_btn. Sličan kod vrijedi za b_btn i d_btn.

```
a_btn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (i>4){
            checkAnswers("a");
            Intent intent = new Intent(QuizActivity.this, ScoreActivity.class);
            Bundle bundle = new Bundle();
            bundle.putInt("rezultat", rez);
            intent.putExtras(bundle);
            startActivity(intent);
            finish();
        }else{
            checkAnswers("a");
            queryDatabase(rsrq.get(i++));
        }

    }
});
```

Sl. 3.11. Java kod provjere klika na a_btn

```
c_btn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (i>4){
            checkAnswers("c");
            Intent intent = new Intent(QuizActivity.this, ScoreActivity.class);
            Bundle bundle = new Bundle();
            bundle.putInt("rezultat", rez);
            intent.putExtras(bundle);
            startActivity(intent);
            finish();
        }else{
            checkAnswers("c");
            queryDatabase(rsrq.get(i++));
        }
    }
```

Sl.3.12. Java kod provjere klika na c_btn

Kao što je vidiljivo na slici, u svakoj metodi „onClick()“ imamo metodu „checkAnswers()“. Metoda „checkAnswers()“ provjerava je li odgovor točan ili netočan. U slučaju da je točan, varijabla „rez“ će se povećati za 1 te će se pojaviti obavijest „Točan odgovor“. U slučaju da je odgovor netočan, korisniku će se pojaviti obavijest „Netočan odgovor“.

```
private void checkAnswers(String ans) {
    if (ans.equals(questions.getAns())){
        rez += 1;
        Toast.makeText(this, "Tocan odgovor", Toast.LENGTH_SHORT).show();
    }else
    {
        Toast.makeText(this, "Netocan odgovor", Toast.LENGTH_SHORT).show();
    }
}
```

Sl. 3.13. Java kod metode „checkAnswers()“

3.2.4. Krajnji rezultat

Korisnik po završetku testa dobiva povratnu informaciju o riješenosti i to u numeričkom i tekstualnom obliku. Posljednji korisnički zaslon dan je „activity_score.xml“ datotekom. Datoteka se sastoji od jednog tekstualnog okvira u koji spremamo rezultat.

Sl. 3.14. Izgled zaslona sa rezultatom

U „ScoreActivity.java“ klasi se vidi da je preko objekta „bundle“ dohvaćen rezultat iz „activity_quiz.java“ klase te je spremljen u cjelobrojnu varijablu „rez“. Potom smo u tekstualni okvir, koji smo definirali u „activity_score.xml“ datoteci, spremili rezultat. Ako je broj točnih odgovora bio veći ili jednak 3, ispisat će se tekst s brojem točnih odgovora i povratna

informacija kao „PROLAZ“ , a u slučaju da je broj točnih odgovora manji od 3, stići će povratna informacija kao „PAD“.

```
public class ScoreActivity extends AppCompatActivity{

    private TextView rezultatTv;

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

        Bundle bundle = getIntent().getExtras();
        int rez = bundle.getInt("rezultat");


        rezultatTv  = (TextView) findViewById(R.id.rezultat_tv);
        if (rez >= 3)
        {
            rezultatTv.setText("Rezultat je: " + rez +  " \n  PROLAZ! :) ");
        }else
            {
            rezultatTv.setText("Rezultat je: " + rez +  " \n  PAD! :(  ");
        }
    }
}
```

Sl. 3.15. Java kod zaslona sa rezultatom

Također, moguće je bilo postaviti i da je prolaz ispita na manjem broju točnih odgovora, no postavljeno je na 3 kako bi se korisnik što bolje pripremio za odgovaranje na pitanja, te odgovorio što više točnih odgovora i samim time bolje naučio određeno gradivo.

3.3. Testiranje aplikacije

Pri izradi korišten je pametni mobilni uređaj Lenovo K3 Note kako bi se aplikacija mogla testirati, iako Android Studio omogućuje i testiranje na virtualnom uređaju. Lenovo K3 Note ima dijagonalu ekrana 5.5“ te samim time preglednost za ovu aplikaciju je bila više nego dobra. Aplikacija je također testirana i na uređaju Xiaomi Redmi 3 koji ima dijagonalu ekrana 5.0“ kako bi se dokazalo da je aplikaciju moguće pokrenuti na različitim dimenzijama ekrana i verzijama operacijskog sustava. Aplikacija sama po sebi nije zahtjevna te ne zahtjeva posebno dobre specifikacije mobilnih uređaja. Jedini zahtjev ove aplikacije je povezanost na internet kako bi se pitanja mogla učitati iz baze u koju je administrator spremio pitanja.

4. ZAKLJUČAK

Mnogi u svijetu danas teško pronalaze motivaciju za učenjem. S ciljem kako bi se olakšao način učenja i shvaćanja određenog gradiva, odlučeno je napraviti aplikaciju koja će pomoći pri tome. Kako veliki broj ljudi danas koristi pametne mobilne uređaje, i kako je Android kao operacijski sustav jedan od najraširenijih u svijetu, tako i na našem području, došla je ideja da se napravi aplikacija baš za taj operacijski sustav. Glavni problem ove aplikacije bilo je upoznavanje sa sintaksom u Android Studiu te samim Android Studiom kao razvojnim okruženjem. Sama aplikacija vrlo je jednostavna za korištenje, iako se od korisnika očekuje što bolja pripremljenost za ispit kako bi ostvario što bolji rezultat.

Aplikacija je izrađena u Android Studiu te je također korištena *Firebase* baza podataka za pohranu pitanja i odgovora. Od programskih alata za razvoj ovakve aplikacije bilo je potrebno usvojiti i poznavati osnovne koncepte objektno orijentiranog programiranja te programskog jezika Java. Aplikaciju je moguće pokrenuti na bilo kojem uređaju koji ima Android platformu.

Kao poboljšanje ove aplikacije u budućnosti može se poraditi na proširenju baze podataka sa pitanjima i odgovorima te na vizualnom izgledu korisničkog i administracijskog sučelja.

LITERATURA

[1] https://en.wikipedia.org/wiki/Android_Studio (posjećeno: 10. lipnja 2017.)

[2] https://en.wikipedia.org/wiki/Android_(operating_system) (posjećeno: 11. lipnja 2017.)

[3] https://en.wikipedia.org/wiki/Java_(programming_language) (posjećeno: 11. lipnja 2017.)

[4] https://en.wikipedia.org/wiki/XML (posjećeno: 21. lipnja 2017.)

[5] https://firebase.google.com (posjećeno: 29. lipnja 2017.)

[6] https://en.wikipedia.org/wiki/Firebase (posjećeno: 1. srpnja 2017.)

SAŽETAK

Cilj rada bio je izraditi aplikaciju za Android operacijski sustav koja će pomoći prvenstveno studentima i učenicima u savladavanju novog ili već naučenog gradiva. Aplikacija nudi lakši način savladavanja gradiva jer mnogi svakodnevno koriste pametne mobilne uređaje. U teorijskom dijelu rada obuhvaćen je opis Androida kao operacijskog sustava te Android Studio kao razvojno okruženje aplikacije. Aplikacija se sastoji od četiri zaslona i to početnog, korisničkog, administracijskog i zaslona s rezultatom. Početni zaslon služi za unos korisničkih podataka s kojima pristupamo ili u administracijsko sučelje ili u korisničko. Administracijski zaslon služi za unos pitanja i odgovora u bazu podataka, a korisnički za odgovoranje na pitanja koja su postavljena u administracijskom dijelu. Prilikom završetka odgovaranja pojavljuje se zaslon s rezultatom na kojemu korisnik dobiva povratnu informaciju s rezultatom koji je ostvario.

Ključne riječi: Android Studio, Java, Android, Firebase, odgovori, pitanja

ABSTRACT

The main goal of this paper was to develop an application for Android operating system that will help students and pupils with their studying. This application provides an easier way of studying, because many people use smart phones every day. The theoretical part of this work includes a description of Android as an operating system, and Android Studio as the development environment for the application. The application has four screens: main, user, administrator and the result screen. In the main screen we enter users' data used for accessing either the administrator interface or the users' interface. The administrator screen is used for entering questions and answers into the database and the users' screen is used for answering the questions that are asked in the administrator part. A screen with the result appears after having answered the questions, and the user gets feedback along with the result he had accomplished.

Keywords: Android Studio, Java, Android, Firebase, questions, answers

ŽIVOTOPIS

Davor Barić rođen je 24.srpnja 1995. godine u Osijeku, Hrvatska. Živi u Đakovu, na adresi Augusta Šenoe 35. 2002. godine započinje osnovnoškolsko obrazovanje u OŠ Ivana Gorana Kovačića u Đakovu. Nakon završenog osnovnoškolskog obrazovanja, 2010. godine upisuje Srednju strukovnu školu Braće Radića u Đakovu, smjer računalni tehničar za strojarstvo. U srednjoj školi se iskazuje na županijskim i državnim natjecanjima iz područja strojarstva. 2013. godine u Osijeku osvaja 2. mjesto na županijskom natjecanju iz predmeta Tehnička mehanika, te samim time osigurava i nastup na državnom natjecanju u Zagrebu iz istog predmeta gdje osvaja 7. mjesto. Godinu dana kasnije osvaja 3. mjesto u Osijeku na županijskom natjecanju iz predmeta Strojarske konstrukcije. Preddiplomski sveučilišni studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku upisuje 2014. godine, koji i dalje pohađa. Od vještina i znanja posjeduje određeno znanje u govoru, čitanju i pisanju engleskog jezika, vozačku dozvolu B kategorije. Posjeduje vještine stečene srednjoškolskim obrazovanjem u dizajniranju i crtanju pomoću CAD alata kao što su AutoCAD i CATIA. Također posjeduje znanje u radu na računalu, rad s Microsoft Office alatima, osnovno znanje programskih jezika C, C++ i C# te rad s bazama podataka.

PRILOZI

CD

- Android Studio projekt
- Pisani rad u pdf. i docx. formatu