

Univerzitet u Kragujevcu  
Fakultet inženjerskih nauka



## Seminarski rad iz predmeta Programiranje mobilnih aplikacija

### Tema:

Razvoj Android aplikacije sa bazom podataka za evidentiranje  
osobe u matičnu knjigu rođenih

Student:  
Nikola Mitrevski

Predmetni profesor:  
dr Nenad Grujović  
Predmetni asistent:  
Vukašin Slavković

Kragujevac 2021.

**Sadržaj:**

1.	Uvod.....	2
2.	Specifikacije aplikacije.....	3
3.	Realizacija.....	8
3.1	Baza podataka.....	9
3.2	Xml layout fajlovi.....	10
3.3	Java klase.....	21
3.3.1	Klasa „MainActivity” .....	21
3.3.2	Klasa „UserActivity” .....	22
3.3.3	Klasa „RegistrationActivity” .....	25
3.3.4	Klasa „UpdateActivity” .....	29
3.3.5	Klasa „UpdateActivity2” .....	35
3.3.6	Klasa „CustomAdapter” .....	38
3.3.7	Klasa „CustomAdapter2” .....	41
3.3.8	Klasa „MainAdminActivity” .....	43
3.3.9	Klasa „AdminActivity” .....	45
3.3.10	Klasa „AdminActivity2” .....	47
3.3.11	Klasa „SignUpActivity” .....	49
3.3.12	Klasa „Check” .....	50
3.3.13	Klasa „DatabaseHelper” .....	51
4	Literatura.....	53

## **1. Uvod**

U ovom dokumentu biće objašnjena android aplikacija koja omogućava upis deteta u matičnu knjigu rođenih, onom roditelju koji se registruje, ali pre svega toga potrebno je definisati sledeće pojmove: android, operativni sistem, aplikacija i baza podataka.

Android je operativni sistem, koji programerima daje mogućnost pravljenja željenih aplikacija za mobilne uređaje, kao što su mobilni telefoni, "tablet" računari, "netbooks" itd.

Operativni sistem je skup programa i rutina odgovoran za kontrolu i upravljanje uređajima i računarskim komponentama kao i za obavljanje osnovnih radnji.

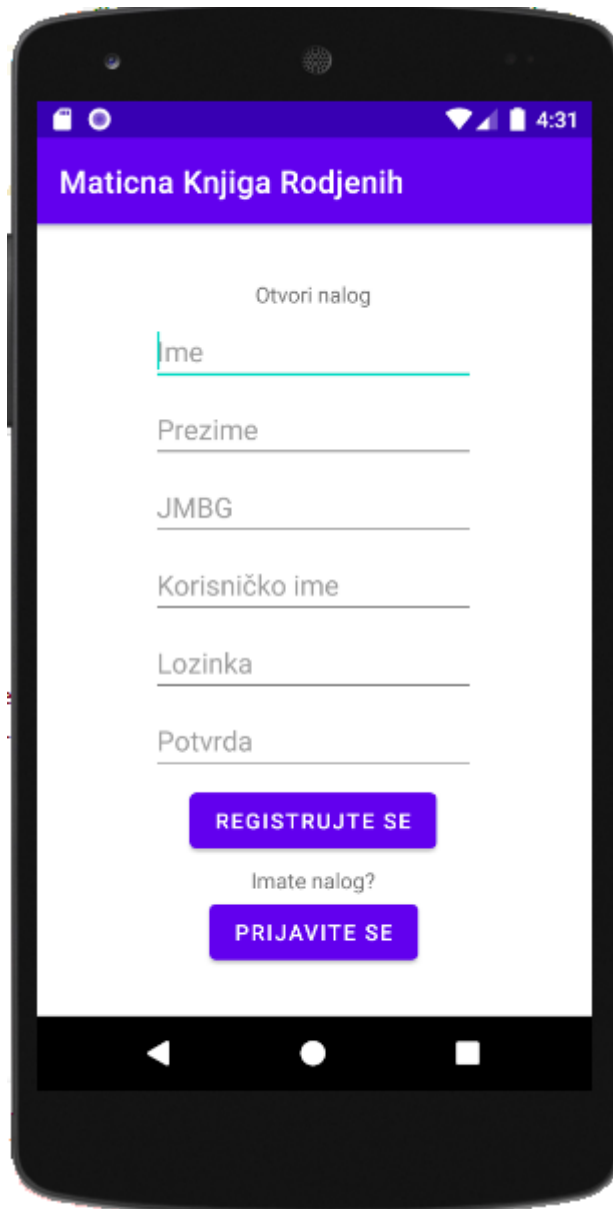
Aplikacije su softveri, dizajnirani za pomoć korisnicima da izvrše jedan ili više određenih zadataka. Primeri aplikacija za mobilne uređaje su: igre, kamera, beleške, sat,...

Baza podataka predstavlja kolekciju međusobno povezanih podataka koji su organizovani u tabele i druge strukture podataka i može da se koristi za jednu ili više aplikacija.

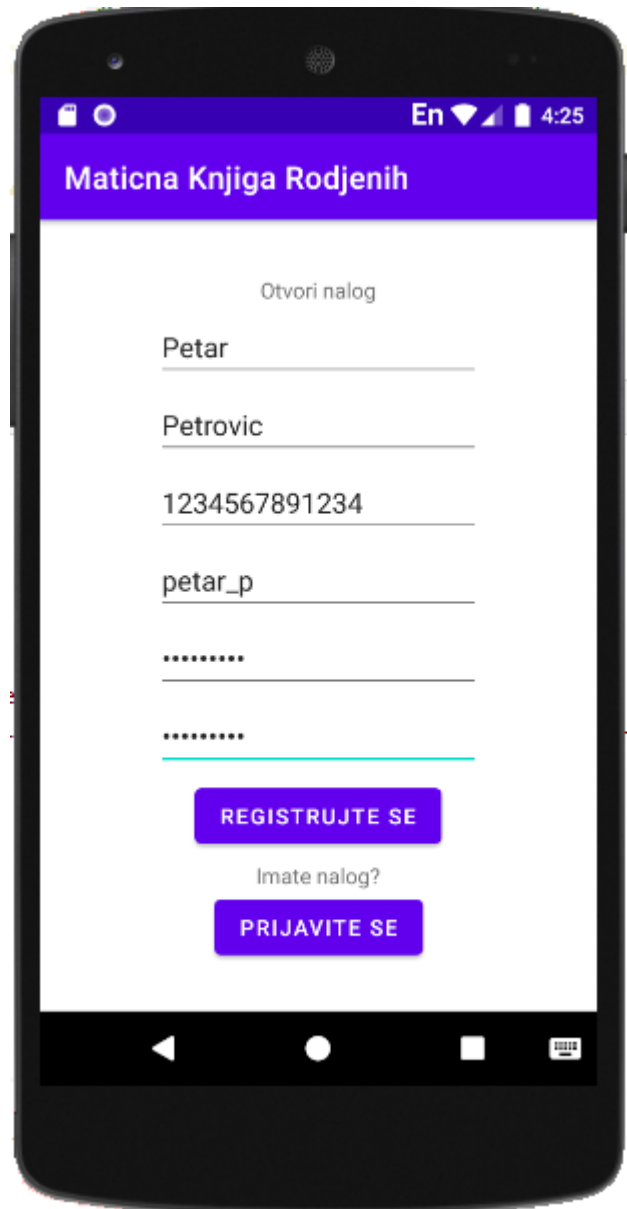
Podaci mogu biti različitog tipa: tekstualni, numerički, slike, audio i video zapisi i sl.

## 2. Specifikacije aplikacije

Aplikacija razvijena u okviru ovog projekta kao što je rečeno u uvodnom delu, služi da korisniku tj. roditelju omogući da upiše svoje dete u matičnu knjigu rođenih, nakon njegove registracije. Roditelj koji je upisao svoje dete na ovaj način, ima mogućnost da dobije izvod iz matične knjige rođenih u pdf formatu.



Slika 1 Prikaz prazne aktivnosti "SignUpActivity"



Slika 2 Prikaz popunjene aktivnosti "SignUpActivity"

Maticna Knjiga Rodjenih

Pol majke (m ili z/ž)

JMBG majke

VREME ROĐENJA MAJKE

DATUM ROĐENJA MAJKE

Mesto rođenja majke

Opština rođenja majke

Država rođenja majke

Državljanstvo majke

Prebivalište majke

Adresa majke

DODAJ

Slika 3 Prikaz prazne aktivnosti "RegistrationActivity"

Maticna Knjiga Rodjenih

z

1234567891836

7:0

5.11.1992

Beograd

Beograd

Republika Srbija

Srpsko

Velika Plana

Milosa Velikog 42

DODAJ

Slika 4 Prikaz popunjene aktivnosti "RegistrationActivity"



## REPUBLIKA SRBIJA

## IZVOD IZ MATIČNE KNJIGE ROĐENIH

## Podaci o detetu:

ime	prezime	jmbg	pol	vreme rođenja
Petar	Petrovic	1234567891235	m	10:50
datum rođenja	mesto rođenja	opština rođenja	država rođenja	državljanstvo
18.1.2013	Smederevska Palanka	Smederevska Palanka	Republika Srbija	Srpsko

## Podaci o ocu:

ime	prezime	jmbg	pol	vreme rođenja	datum rođenja
Nikola	Mitrevski	1234567891234	m	20:35	18.1.2021
mesto rođenja	opština rođenja	država rođenja	državljanstvo	prebivalište	adresa
Smederevo	Smederevo	Republika Srbija	Srpsko	Velika Plana	29 oktobar 778

## Podaci o majki:

ime	prezime	jmbg	pol	vreme rođenja	datum rođenja
Nikolija	Mitrevska	1234567891239	z	10:48	18.1.2021
mesto rođenja	opština rođenja	država rođenja	državljanstvo	prebivalište	adresa
Beograd	Beograd	Republika Srbija	Srpsko	Velika Plana	29 oktobar 778

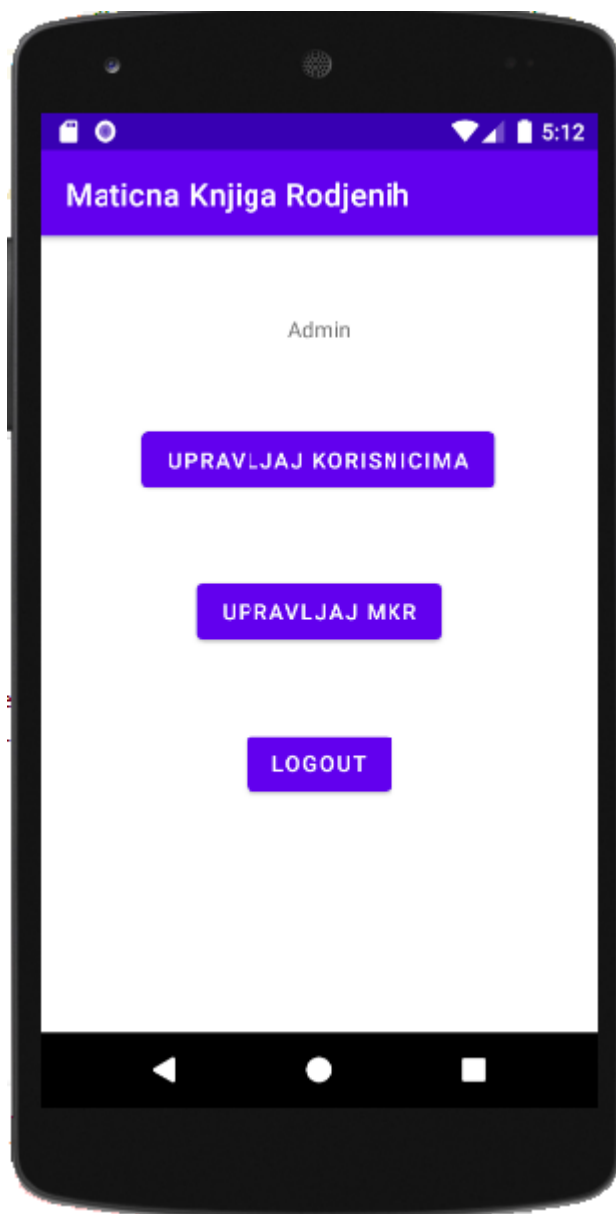
Datum izdavanja: 19/01/21

Vreme izdavanja: 16:47:40

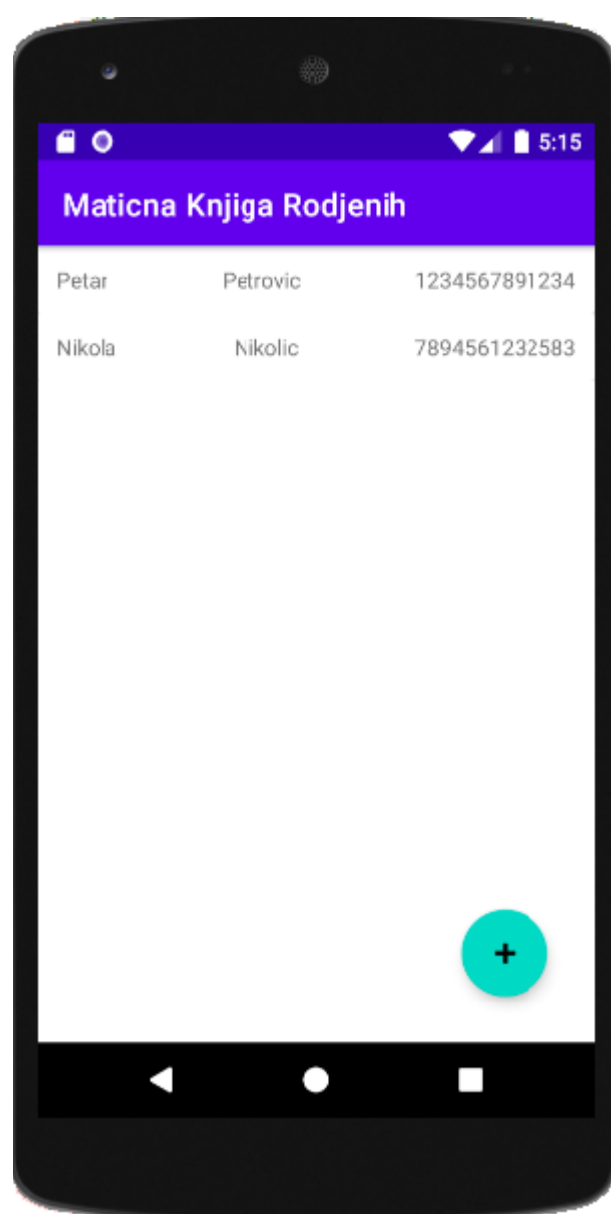

**NIKOLA MITREVSKI**  
 nikola.mitrevski1998@gmail.com

Slika 5 Prikaz primera kreiranog PDF fajla

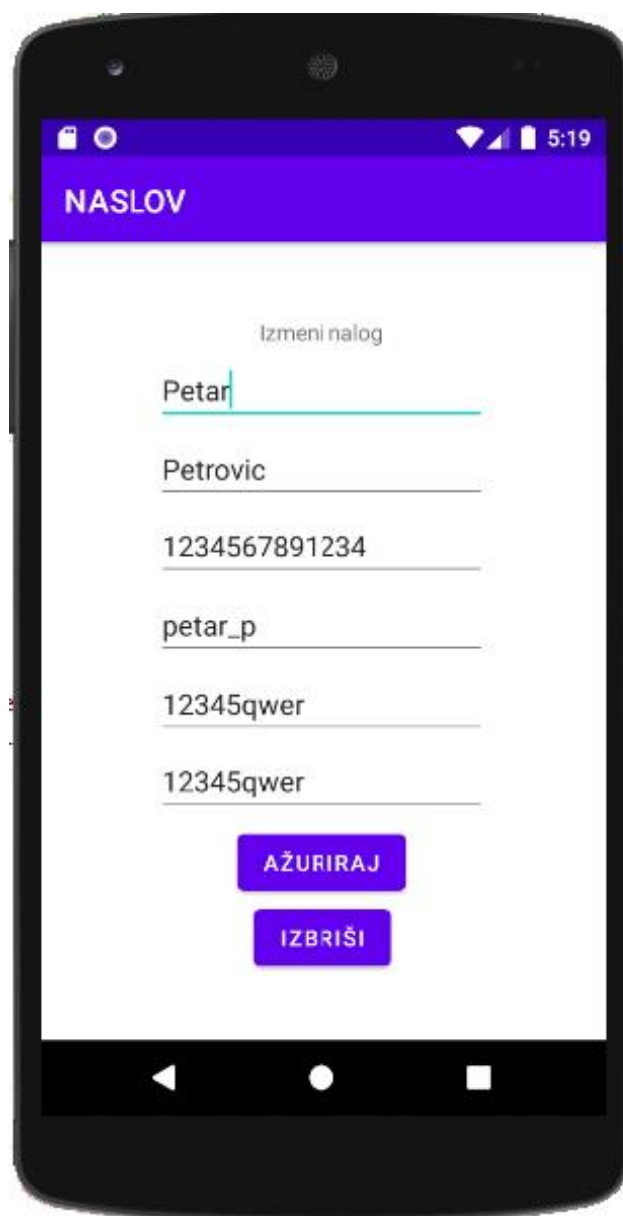
Takođe implementirana je i administratorska strana aplikacije, gde korisnik ulogavan kao administrator može da manipuliše svim podacima koji se nalaze u aplikaciji u bazi podataka.



Slika 6 Prikaz aktivnosti "MainAdminActivity"



Slika 7 Prikaz aktivnosti "AdminActivity"



Slika 8 Prikaz popunjene aktivnosti "UpdateActivity2"



### 3. Realizacija

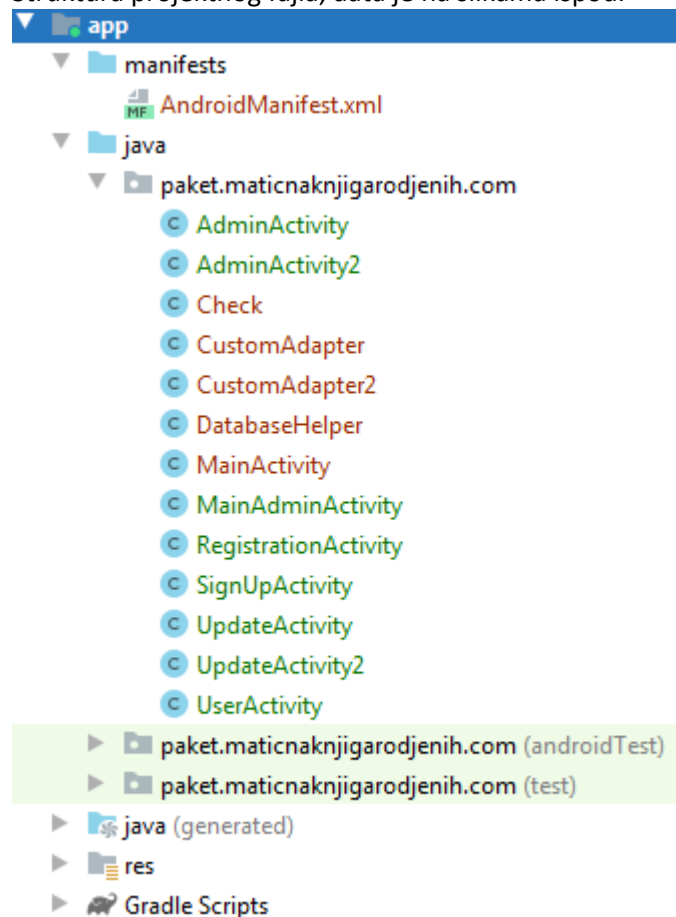
Ova aplikacija se sastoji iz jednostavne baze podataka i klijentske Android aplikacije.

Ona je napravljena u „Android Studio” razvojnom okruženju u kome su pisane Java klase koje su povezane sa određenim xml fajlovima.

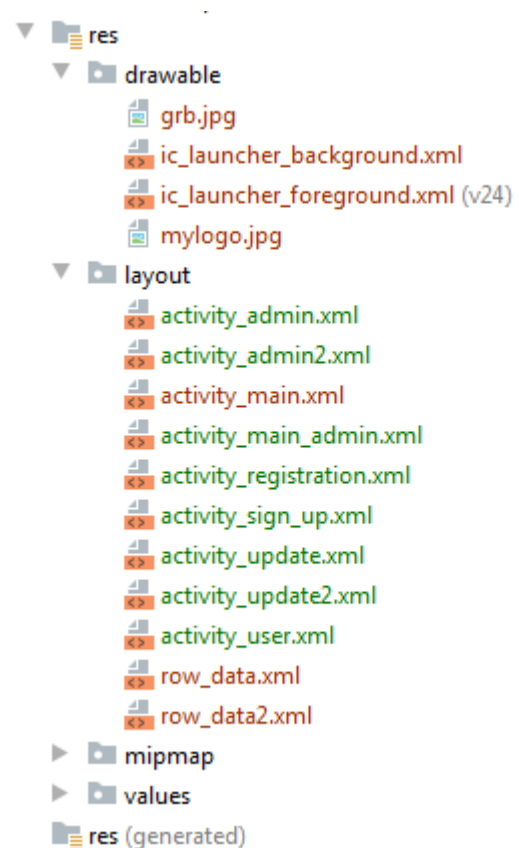
Xml fajlovi definišu izgled grafičkog korisničkog interfejsa.

Java klase koje su napisane su namenjene za rad sa bazom podataka i za upravljanje grafičkim korisničkim interfejsom.

Struktura projektnog fajla, data je na slikama ispod.



Slika 9 Prikaz strukture projektnog fajla (1 deo)



Slika 10 Prikaz strukture projektnog fajla (2 deo)

### 3.1 Baza podataka

Baza podataka koja je napravljena za ovu aplikaciju zove se "baza.db" i sastoji se od dve tabele: "maticnaKnjigaRodjenih" i "korisnici".

maticnaKnjigaRodjenih	korisnici
Column	Column
imeDeteta TEXT	ime TEXT
prezimeDeteta TEXT	prezime TEXT
jmbgDeteta TEXT	jmbg TEXT
polDeteta TEXT	korisnickolme TEXT
vremeRodjenjaDeteta TEXT	lozinka TEXT
datumRodjenjaDeteta TEXT	
mestoRodjenjaDeteta TEXT	
opstinaRodjenjaDeteta TEXT	
drzavaRodjenjaDeteta TEXT	
drzavljanstvoDeteta TEXT	
imeOca TEXT	
prezimeOca TEXT	
jmbgOca TEXT	
polOca TEXT	
vremeRodjenjaOca TEXT	
datumRodjenjaOca TEXT	
mestoRodjenjaOca TEXT	
opstinaRodjenjaOca TEXT	
drzavaRodjenjaOca TEXT	
drzavljanstvoOca TEXT	
prebivalisteOca TEXT	
adresaOca TEXT	
imeMajke TEXT	
prezimeMajke TEXT	
jmbgMajke TEXT	
polMajke TEXT	
vremeRodjenjaMajke TEXT	
datumRodjenjaMajke TEXT	
mestoRodjenjaMajke TEXT	
opstinaRodjenjaMajke TEXT	
drzavaRodjenjaMajke TEXT	
drzavljanstvoMajke TEXT	
prebivalisteMajke TEXT	
adresaMajke TEXT	

Slika 12 Prikaz strukture tabele "korisnici"

Slika 11 Prikaz strukture tabele "maticnaKnjigaRodjenih"

Tabela "maticnaKnjigaRodjenih" ima ukupno 34 kolona, od kojih kolona koja se zove "jmbgDeteta" predstavlja primarni ključ i vrednosti u toj koloni moraju biti jedinstvene, nesmeju postojati duplikati vrednosti.

Tabela "korisnici" ima ukupno 5 kolona, od kojih kolona koja se zove "korisnickoIme" predstavlja primarni ključ i vrednosti u toj koloni moraju biti jedinstvene, nesmeju postojati duplikati vrednosti.

Sve kolone su tekstualnog tipa(TEXT) u obe tabele.

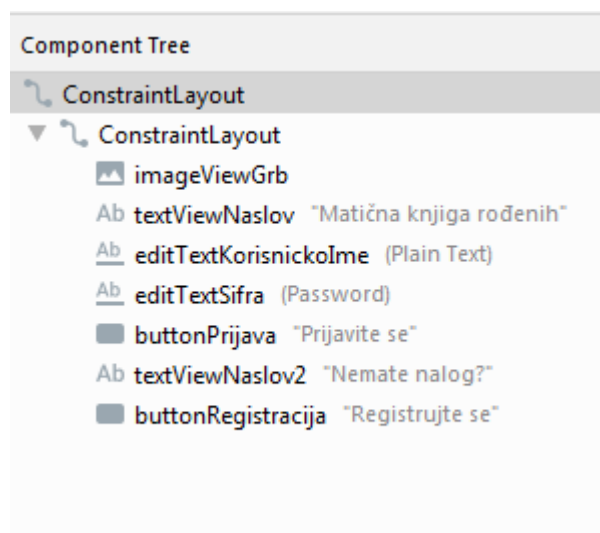
Ove dve tabele nisu fizički povezane(ne postoji strani ključ) ali jesu logički u kodu, biće objašnjeno kasnije.

### 3.2 Xml layout fajlovi

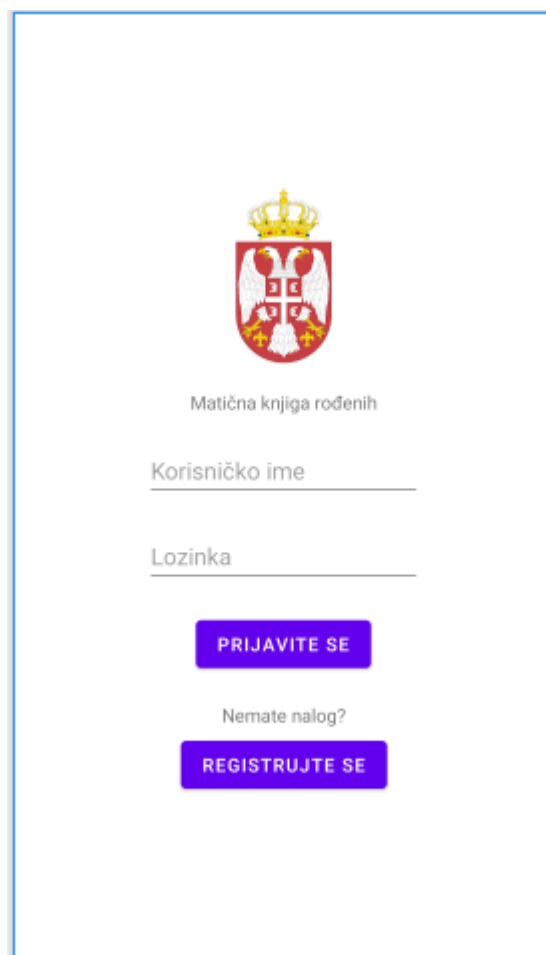
Za potrebe ove aplikacije, kreirano je ukupno 11 xml layout fajlova.

Glavni xml layout fajl se zove „activity main“, jer je to layout koji se prikaže prvi, kada se pokrene aplikacija.

Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi ovaj layout, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.

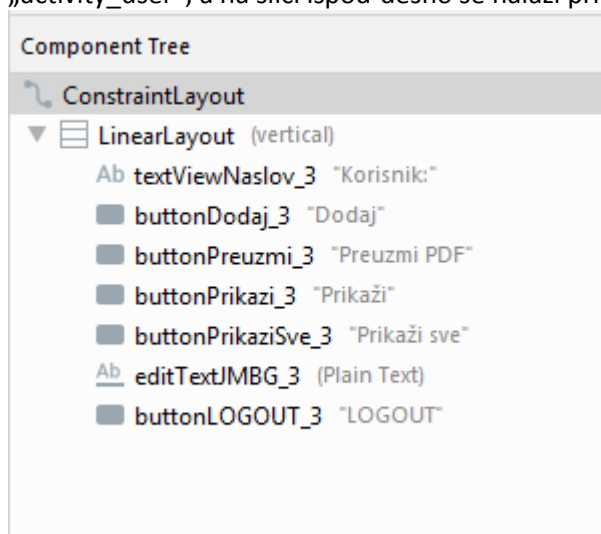


Slika 13 Prikaz stabla komponenta za aktivnost "MainActivity"

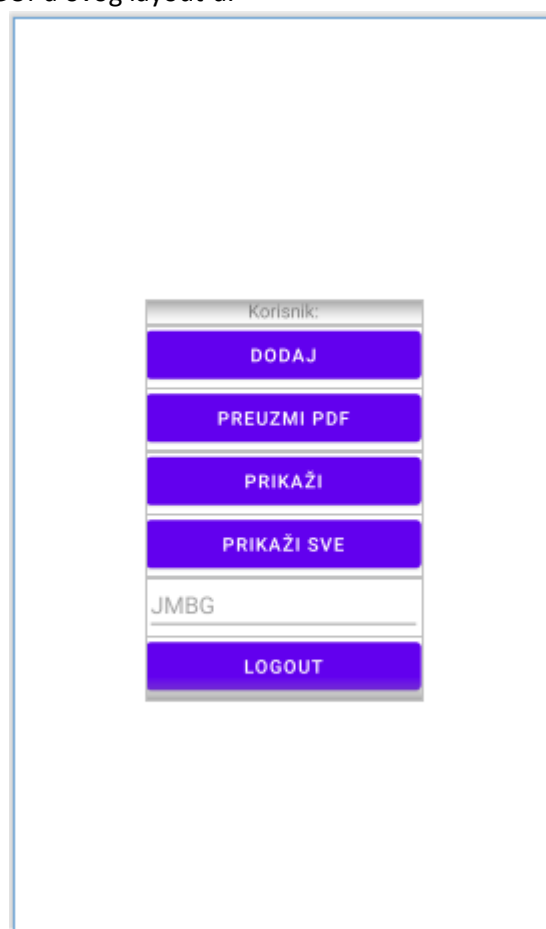


Slika 14 Prikaz GUI-a aktivnosti "MainActivity"

Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „activity\_user“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.

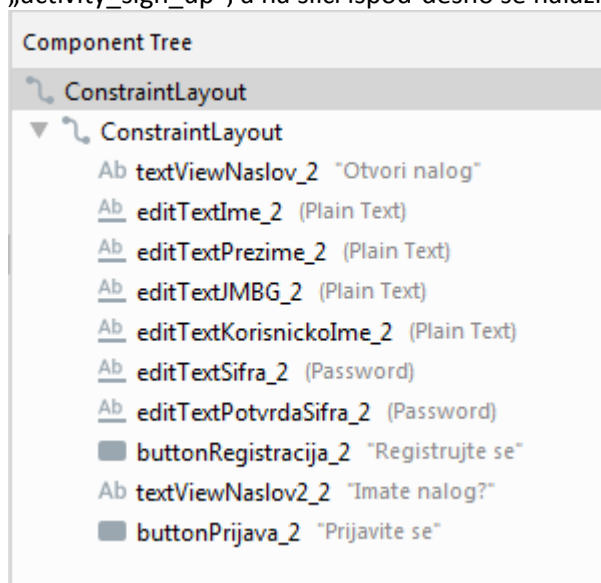


Slika 15 Prikaz stabla komponentata za aktivnost "UserActivity"

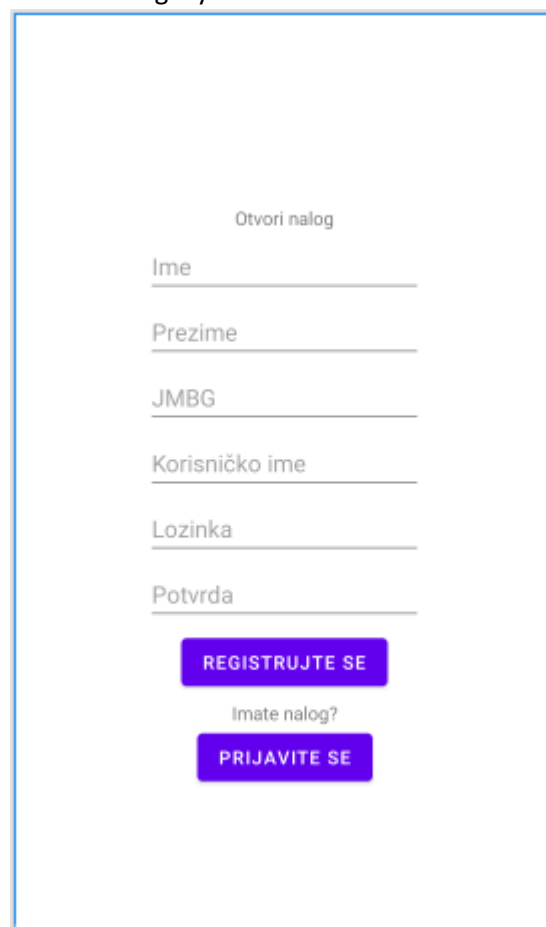


Slika 16 Prikaz GUI-a aktivnosti "UserActivity"

Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „activity\_sign\_up“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.

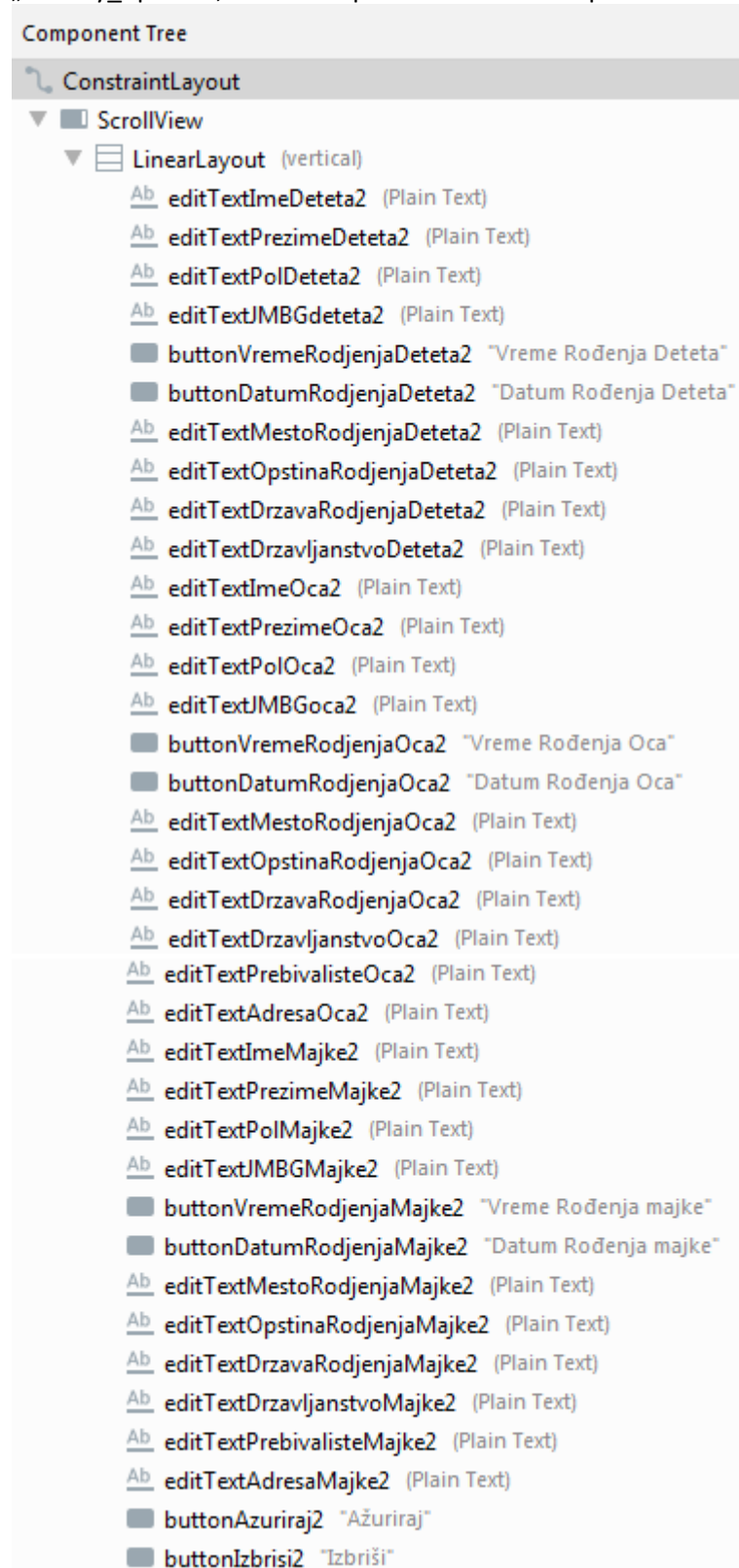


Slika 17 Prikaz stabla komponenata za aktivnost "SignUpActivity"



Slika 18 Prikaz GUI-a aktivnosti "SignUpActivity"

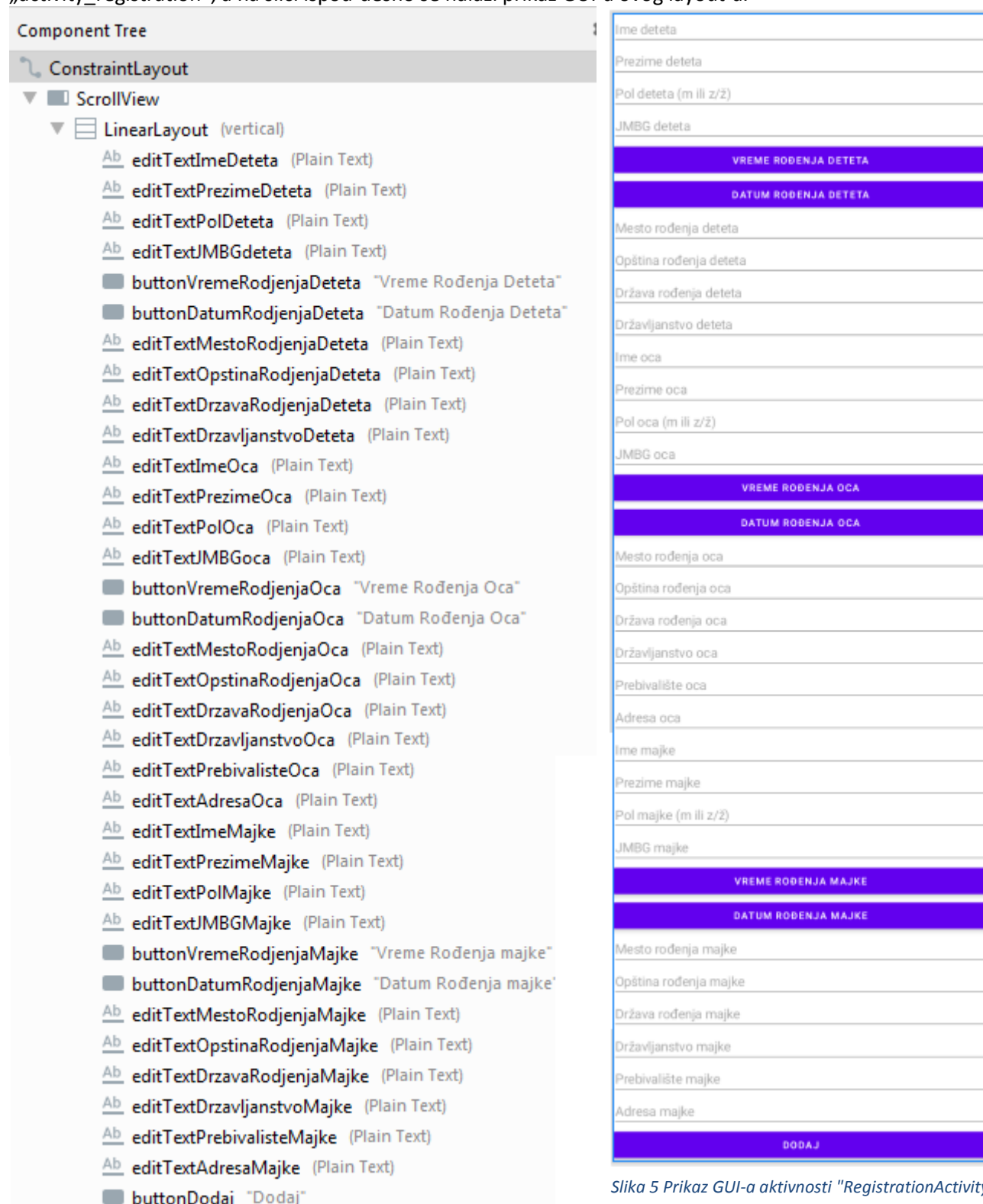
Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „activity\_update“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.



Slika 19 Prikaz stabla komponentata za aktivnost "UpdateActivity"

Slika 20 Prikaz GUI-a aktivnosti "UpdateActivity"

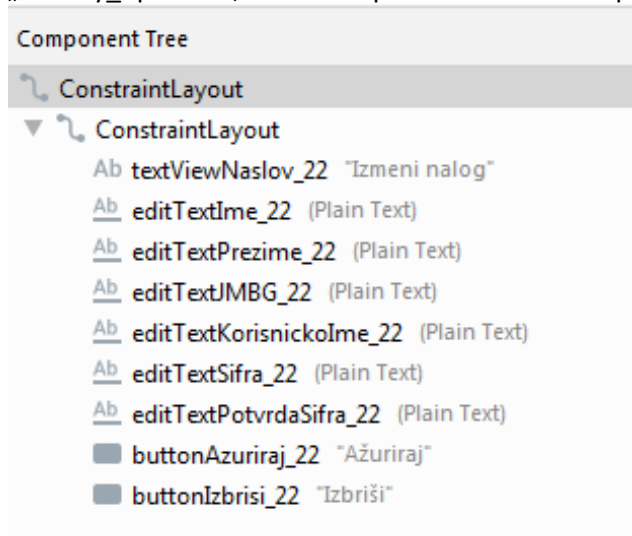
Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „activity\_registration“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.



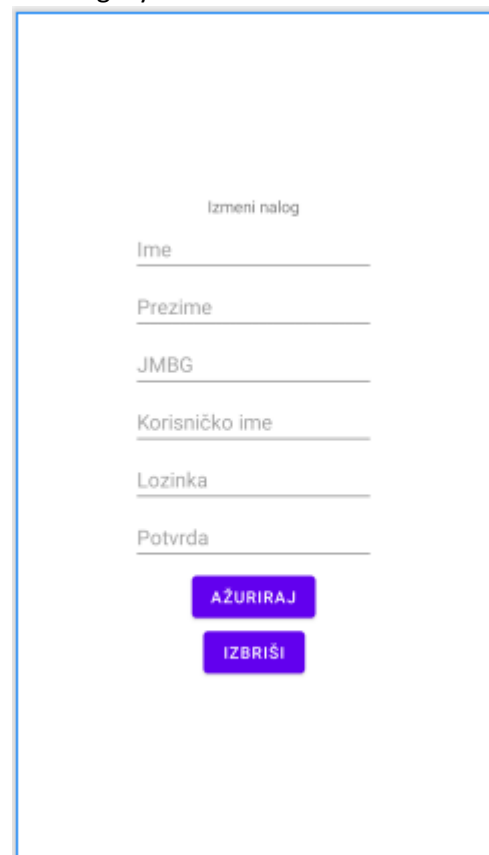
Slika 5 Prikaz GUI-a aktivnosti "RegistrationActivity"

Slika 61 Prikaz stabla komponenata za aktivnost "RegistrationActivity"

Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „activity\_update2“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.



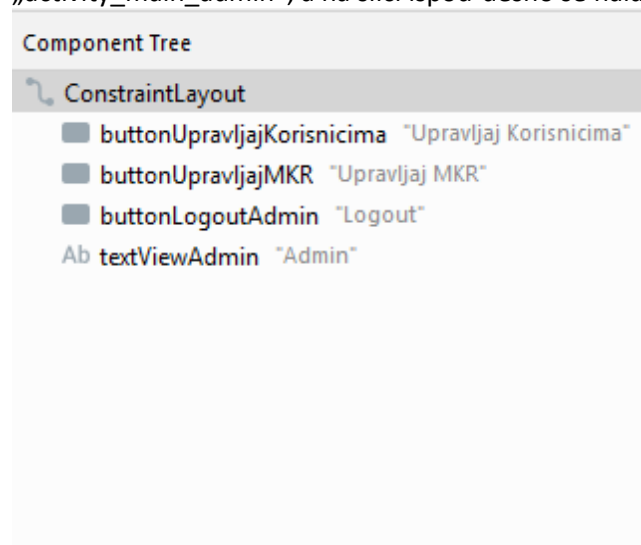
Slika 23 Prikaz stabla komponenata za aktivnost "UpdateActivity2"



Slika 24 Prikaz GUI-a aktivnosti "UpdateActivity2"



Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „activity\_main\_admin“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.

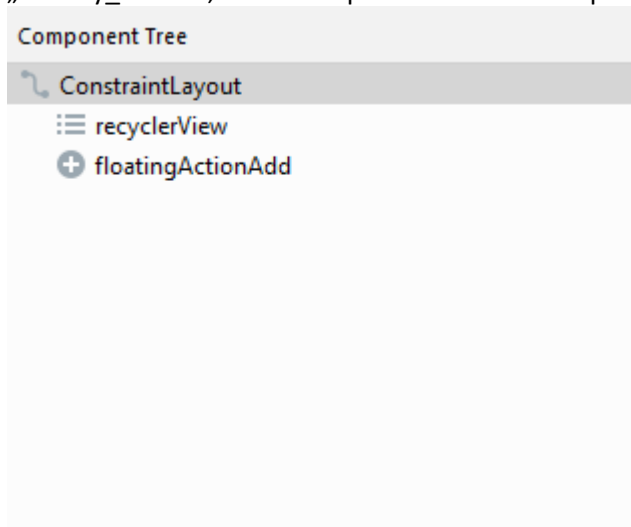


Slika 25 Prikaz stabla komponentata za aktivnost "MainAdminActivity"

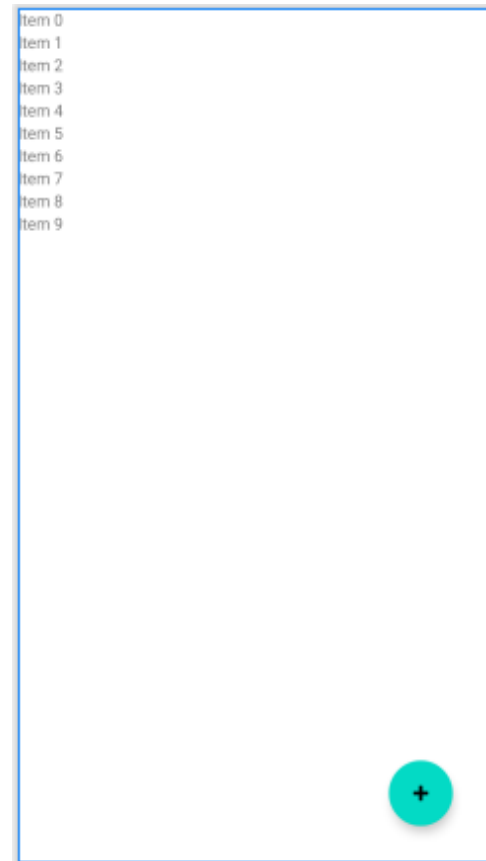


Slika 26 Prikaz GUI-a aktivnosti "MainAdminActivity"

Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „activity\_admin“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.

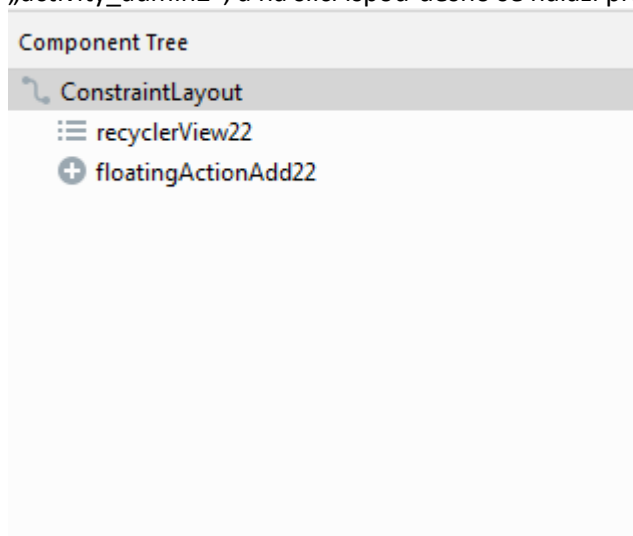


Slika 27 Prikaz stabla komponenata za aktivnost "AdminActivity"

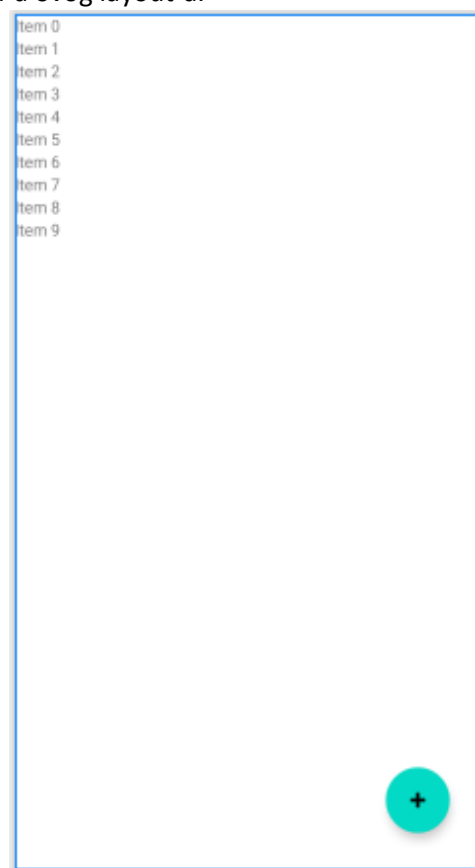


Slika 28 Prikaz GUI-a aktivnosti "AdminActivity"

Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „activity\_admin2“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.

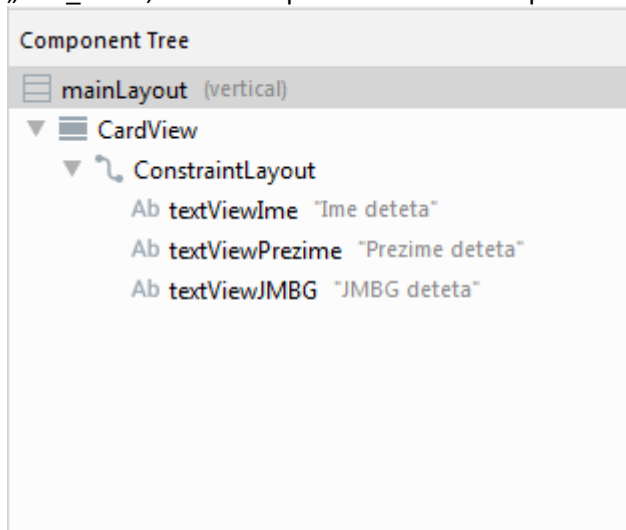


Slika 29 Prikaz stabla komponenata za aktivnost "AdminActivity2"

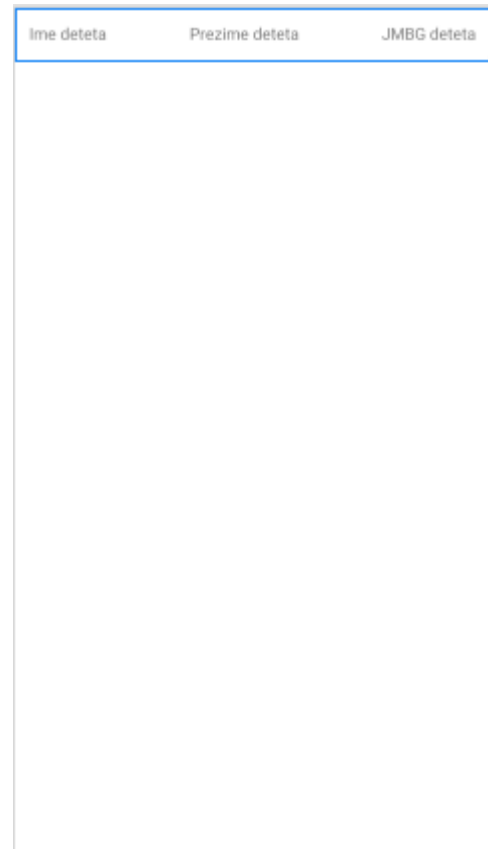


Slika 30 Prikaz GUI-a aktivnosti "AdminActivity2"

Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „row\_data“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.

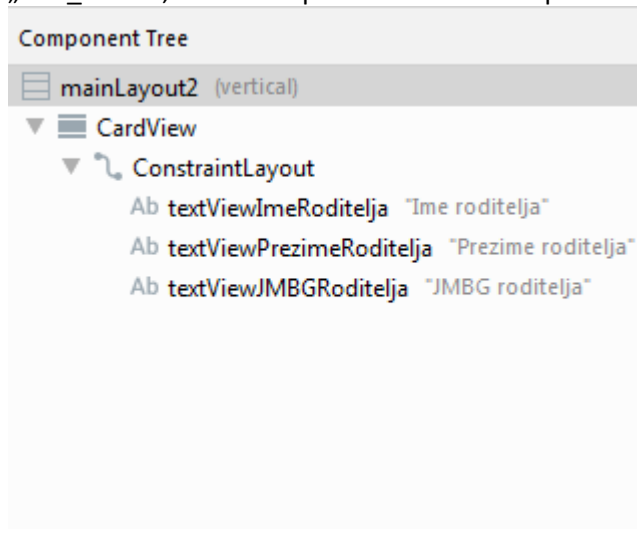


Slika 31 Prikaz stabla komponenata za aktivnost "RowData"



Slika 32 Prikaz GUI-a aktivnosti "RowData"

Na slici ispod-levo se nalazi strukturni prikaz svih grafičkih komponenti koje koristi layout „row\_data2“, a na slici ispod-desno se nalazi prikaz GUI-a ovog layout-a.



Slika 33 Prikaz stabla komponenata za aktivnost "RowData2"



Slika 34 Prikaz GUI-a aktivnosti "RowData2"

### 3.3 Java klase

#### 3.3.1 Klasa „MainActivity”

Klasa „MainActivity” nasleđuje klasu „AppCompatActivity”, slika 35.

```
12 public class MainActivity extends AppCompatActivity {
```

Slika 35 Prikaz klase "MainActivity" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_main.xml, slika 36” i namenjana je za preuzimanje korisničkih podataka prilikom prijavljivanja korisnika i prelazak u aktivnost „UserActivity”, „MainAdminActivity” ili „SignUpActivity”, u zavisnosti od događaja.

```
25 setContentView(R.layout.activity_main);
```

Slika 36 Prikaz naziva layout-a koji koristi klasa "MainActivity"

Mogući događaji su: pritisak na taster „PRIJAVITE SE” i pritisak na taster „REGISTRUJTE SE”.

U prvom delu ove klase se nalaze potrebne deklaracije, slika 37.

```
13 EditText editTextKorisnikIme, editTextSifra;
14 Button buttonPrijava, buttonRegistracija;
15
16 String korisnickoIme, sifra, jmbgRoditelja;
17 boolean rezultat;
18 Bundle extra;
19
20 DatabaseHelper databaseHelper;
```

Slika 37 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana(override) metoda „onCreate”, koja se poziva automatski prilikom startovanja programa, slika 38.

```
22 @Override
23 protected void onCreate(Bundle savedInstanceState) {...}
```

Slika 38 Prikaz override metode "onCreate"

Unutar metode „onCreate” je postavljen osluškivač na tastere „buttonPrijava” i „buttonRegistracija”, gde će osluškivač u zavisnosti od događaja pozvati odgovarajuću nadglasanu(override) metodu „onClick” koja će se izvršiti, slika 39.

```
32 buttonPrijava.setOnClickListener(new View.OnClickListener() {
33     @Override
34     public void onClick(View v) {...}
62 });
63
64 buttonRegistracija.setOnClickListener(new View.OnClickListener() {
65     @Override
66     public void onClick(View v) {...}
70 });
```

Slika 39 Prikaz osluškivača na tastere

Metoda „onClick“ koju poziva osluškivač na „buttonPrijava“, čuva korisničke ulaze iz EditText polja(slika 40) u obliku stringa, te ulaze prosleđuje metodi „validateUserInKorisnici“(slika 41), kako bi se izvršila odgovarajuća validacija i u zavisnosti od rezultata te validacije se izvršava određen deo koda koji je smešten u selekciji if(slika 42).

```
36      korisnickoIme = editTextKorisnickoIme.getText().toString();
37      sifra = editTextSifra.getText().toString();
```

Slika 40 Prikaz vraćanja vrednosti EditText polja i čuvanja, unutar metode „onClick“ koju poziva osluškivač na „buttonPrijava“

```
41      rezultat = databaseHelper.validateUserInKorisnici(korisnickoIme, sifra);
```

Slika 41 Prikaz poziva metode "validateUserInKorisnici" unutar metode „onClick“ koju poziva osluškivač na „buttonPrijava“

```
43      if(rezultat) {
44          jmbgRoditelja = databaseHelper.getJMBGRoditeljaFromKorisnici(korisnickoIme);
45
46          extra = new Bundle();
47          extra.putString("korisnickoIme", korisnickoIme);
48          extra.putString("jmbgRoditelja", jmbgRoditelja);
49
50          Intent intent = new Intent( packageContext: MainActivity.this, UserActivity.class);
51          intent.putExtras(extra);
52          startActivity(intent);
53      }
54      else if(korisnickoIme.equals("admin") && sifra.equals("12345")) {
55          Intent intent = new Intent( packageContext: MainActivity.this, MainAdminActivity.class);
56          startActivity(intent);
57      }
58      else {
59          Toast.makeText( context: MainActivity.this, text: "Izabrali ste nepostojeće korisničko ime ili lozinku", Toast.LENGTH_SHORT).show();
60      }
61  }
62  };
```

Slika 42 Prikaz selekcija if unutar metode „onClick“ koju poziva osluškivač na „buttonPrijava“

Ako je rezultat validacije jednak „true“ tada se ulazi u if selekciju i prva važna stvar koja se radi je spremanje podataka za prosleđivanje iz jedne aktivnosti u drugu i druga važna stvar je promena aktivnosti.

Ako rezultat validacije nije jednak „true“ tada se proverava uslov else-if selekcije i ako je uslov zadovoljen, tada se ulazi u telo else-if selekcije i prelazi se u drugu aktivnost.

Ako nisu zadovoljeni uslovi selekcija if i else-if, tada se prikazuje „toast“ poruka.

### 3.3.2 Klasa „UserActivity“

Klasa „UserActivity“ nasleđuje klasu „AppCompatActivity“, slika 43.

```
34 public class UserActivity extends AppCompatActivity {
```

Slika 43 Prikaz klase "UserActivity" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_user.xml“(slika 44) i namenjena je za preuzimanje korisničkog podatka sa ulaza, formiranje PDF fajla, prikazivanje određenih podataka preko „alert“ dialoga i prelazak u aktivnost „RegistrationActivity“.

```
51 setContentView(R.layout.activity_user);
```

Slika 44 Prikaz naziva layout-a koji koristi klasa "UserActivity"

U prvom delu ove klase se nalaze potrebne deklaracije, slika 45.

```

35     TextView textViewNaslov;
36     Button buttonDodaj, buttonPreuzmi, buttonPrikazi, buttonPrikaziSve, buttonLOGOUT;
37     EditText editTextJMBG;
38     String korisnickoIme, jmbgRoditelja, jmbgDeteta;
39     Bundle extra;
40
41     DatabaseHelper databaseHelper;
42
43     Bitmap bmp, scaledbmp; // koristimo za dodavanje slike u pdf file
44     Bitmap bmp2, scaledbmp2; // koristimo za dodavanje slike u pdf file
45     Date dateObj;           // koristimo za kreiranje trenutnog datuma i vremena u pdf file
46     DateFormat dateFormat; // koristimo za kreiranje trenutnog datuma i vremena u pdf file

```

Slika 45 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana(override) metoda „onCreate“, koja se poziva automatski prilikom prelaska iz aktivnosti „MainActivity“ u aktivnost „UserActivity“, slika 46.

```

49     protected void onCreate(Bundle savedInstanceState) {

```

Slika 46 Prikaz override metode "onCreate"

Unutar metode „onCreate“ su postavljeni osluškivači na tastere „buttonDodaj“, „buttonPreuzmi“, „buttonPrikazi“, „buttonPrikaziSve“ i „buttonLOGOUT“, gde će osluškivač u zavisnosti od događaja pozvati odgovarajuću nadglasanu(override) metodu „onClick“ koja će se izvršiti, slika 47.

```

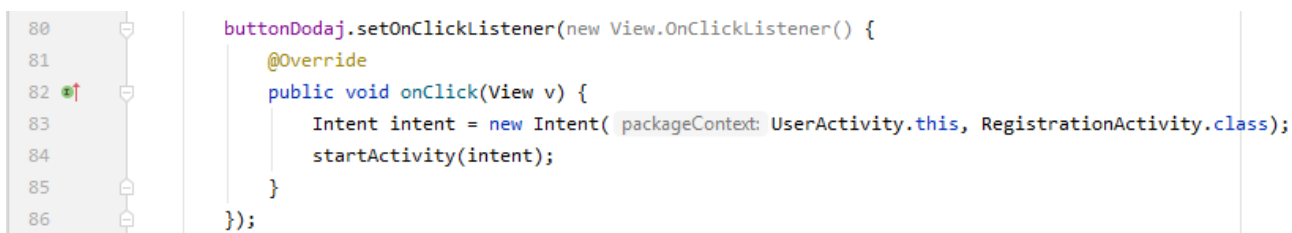
80     buttonDodaj.setOnClickListener(new View.OnClickListener() {
81         @Override
82         public void onClick(View v) {
86     });
87
88     buttonPreuzmi.setOnClickListener(new View.OnClickListener() {
89         @Override
90         public void onClick(View v) {
364     });
365
366     buttonPrikazi.setOnClickListener(new View.OnClickListener() {
367         @Override
368         public void onClick(View v) {
394     });
395
396     buttonPrikaziSve.setOnClickListener(new View.OnClickListener() {
397         @Override
398         public void onClick(View v) {
422     });
423
424     buttonLOGOUT.setOnClickListener(new View.OnClickListener() {
425         @Override
426         public void onClick(View v) { finish(); }
429     });

```

Slika 47 Prikaz osluškivača na tastere

Osluškivač na „buttonDodaj“ poziva nadglasanu(override) metodu „onClick“ koja omogućava prelazak u aktivnost „RegistrationActivity“, slika 48.





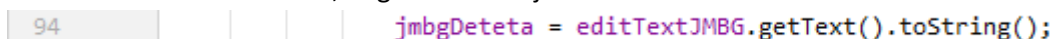
```

80
81
82 buttonDodaj.setOnClickListener(new View.OnClickListener() {
83     @Override
84     public void onClick(View v) {
85         Intent intent = new Intent( packageContext: UserActivity.this, RegistrationActivity.class);
86         startActivity(intent);
87     }
88 });

```

Slika 48 Prikaz definicije nadglasane metode "onClick" koju poziva osluškivač tastera "buttonDodaj"

Osluškivač na „buttonPreuzmi“ poziva nadglasanu(override) metodu „OnClick“ koja omogućava preuzimanje korisničkog ulaza iz EditText-a(slika 49), zatim prosleđivanje te vrednosti metodi „readOneDataFromMaticnaKnjigaRodjenih“ koja vraća „cursor“ sa učitanim podacima iz baze podataka(slika 50), gde se ti podaci smeštaju u PDF fajl koji će kreirati i sačuvati ova metoda, negde na uređaju.

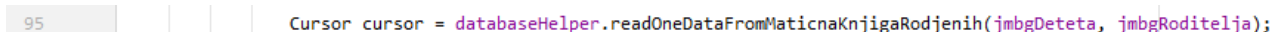


```

94 jmbgDeteta = editTextJMBG.getText().toString();

```

Slika 49 Prikaz vraćanja vrednosti EditText polja i čuvanje, unutar metode „onClick“ koju poziva osluškivač na „buttonPreuzmi“



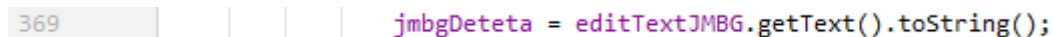
```

95 Cursor cursor = databaseHelper.readOneDataFromMaticnaKnjigaRodjenih(jmbgDeteta, jmbgRoditelja);

```

Slika 50 Prikaz poziva metode "readOneDataFromMaticnaKnjigaRodjenih", unutar metode „onClick“ koju poziva osluškivač na „buttonPreuzmi“

Osluškivač na „buttonPrikazi“ poziva nadglasanu(override) metodu „OnClick“ koja omogućava preuzimanje korisničkog ulaza iz EditText-a(slika 51), prosleđivanje te vrednosti metodi „readOneDataFromMaticnaKnjigaRodjenih“ koja vraća „cursor“ sa učitanim podacima iz baze podataka(slika 52) i poziv metode showMessage(slika 53), koja pomoću „alert“ dijaloga prikazuje podatke o jednom detetu(slika 54).

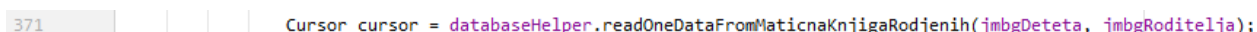


```

369 jmbgDeteta = editTextJMBG.getText().toString();

```

Slika 51 Prikaz vraćanja vrednosti EditText polja i čuvanje, unutar metode „onClick“ koju poziva osluškivač na „buttonPrikazi“

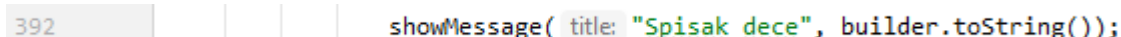


```

371 Cursor cursor = databaseHelper.readOneDataFromMaticnaKnjigaRodjenih(jmbgDeteta, jmbgRoditelja);

```

Slika 52 Prikaz poziva metode "readOneDataFromMaticnaKnjigaRodjenih", unutar metode „onClick“ koju poziva osluškivač na „buttonPrikazi“

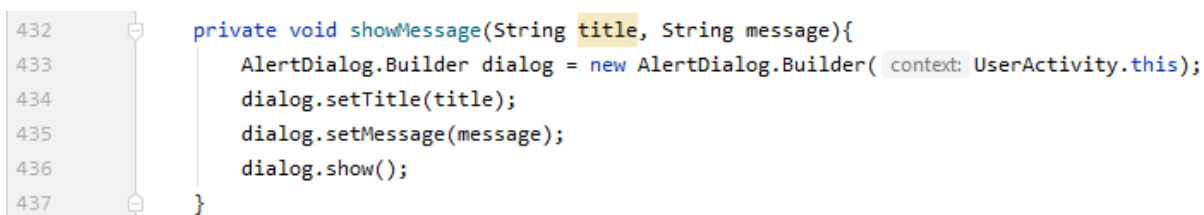


```

392 showMessage( title: "Spisak dece", builder.toString());

```

Slika 53 Prikaz poziva metode "showMessage", unutar metode „onClick“ koju poziva osluškivač na „buttonPriikazi“



```

432 private void showMessage(String title, String message){
433     AlertDialog.Builder dialog = new AlertDialog.Builder( context: UserActivity.this);
434     dialog.setTitle(title);
435     dialog.setMessage(message);
436     dialog.show();
437 }

```

Slika 54 Prikaz definicije metode "showMessage"

Osluškivač na „buttonPrikaziSve“ poziva nadglasanu(override) metodu „OnClick“ koja poziva metodu „readDataFromMaticnaKnjigaRodjenih“ koja vraća „cursor“ sa učitanim podacima iz baze podataka(slika 55) i metodu showMessage(slika 56), koja pomoću „alert“ dijaloga prikazuje podatke o jednom ili više deteta(slika 54).

```
399 Cursor cursor = databaseHelper.readDataFromMaticnaKnjigaRodjenih(jmbgRoditelja);
```

Slika 55 Prikaz poziva metode "readDataFromMaticnaKnjigaRodjenih", unutar metode „onClick“ koju poziva osluškivač na "buttonPrikaziSve"

```
420 showMessage( title: "Spisak dece", builder.toString());
```

Slika 56 Prikaz poziva metode "showMessage", unutar metode „onClick“ koju poziva osluškivač na "buttonPrikaziSve"

Osluškivač na „buttonLOGOUT“ završava aktivnost(slika 57).

```
427 finish();
```

Slika 57 Prikaz poziva metode "finish" unutar metode „onClick“ koju poziva osluškivač na "buttonLOGOUT"

### 3.3.3 Klasa „RegistrationActivity“

Klasa "RegistrationActivity" nasleđuje klasu „AppCompatActivity“, slika 58.

```
18 public class RegistrationActivity extends AppCompatActivity {
```

Slika 58 Prikaz klase "RegistrationActivity" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_registration.xml“(slika 59) i namenjana je za preuzimanje korisničkih podataka sa ulaza prilikom registracije deteta i smeštanje istih u bazu podataka.

```
47 setContentView(R.layout.activity_registration);
```

Slika 59 Prikaz naziva layout-a koji koristi klasa "RegistrationActivity"

U prvom delu ove klase se nalaze potrebne deklaracije, slika 60.

```

19  EditText editTextImeDeteta, editTextPrezimeDeteta, editTextPolDeteta, editTextJMBGdeteta, editTextMestoRodjenjaDeteta;
20  EditText editTextOpstinaRodjenjaDeteta, editTextDrzavaRodjenjaDeteta, editTextDrzavljanstvoDeteta;
21  Button buttonVremeRodjenjaDeteta, buttonDatumRodjenjaDeteta;
22  DatePickerDialog.OnDateSetListener setListener;
23  int Dsat, Dminut;
24  int Dgodina, Dmesec, Ddan;
25  String vremeRodjenjaDeteta, datumRodjenjaDeteta;
26
27  EditText editTextImeOca, editTextPrezimeOca, editTextPolOca, editTextJMBGOca, editTextMestoRodjenjaOca, editTextOpstinaRodjenjaOca;
28  EditText editTextDrzavaRodjenjaOca, editTextDrzavljanstvoOca, editTextPrebivalisteOca, editTextAdresaOca;
29  Button buttonVremeRodjenjaOca, buttonDatumRodjenjaOca;
30  DatePickerDialog.OnDateSetListener setListener2;
31  int Osat, Ominut;
32  int Ogodina, Omesec, Odan;
33  String vremeRodjenjaOca, datumRodjenjaOca;
34
35  EditText editTextImeMajke, editTextPrezimeMajke, editTextPolMajke, editTextJMBGMajke, editTextMestoRodjenjaMajke, editTextOpstinaRodjenjaMajke;
36  EditText editTextDrzavaRodjenjaMajke, editTextDrzavljanstvoMajke, editTextPrebivalisteMajke, editTextAdresaMajke;
37  Button buttonVremeRodjenjaMajke, buttonDatumRodjenjaMajke;
38  DatePickerDialog.OnDateSetListener setListener3;
39  int Msat, Mminut;
40  int Mgodina, Mmesec, Mdan;
41  String vremeRodjenjaMajke, datumRodjenjaMajke;
42
43  Button buttonDodaj;
44  DatabaseHelper databaseHelper;
45  String[] podatak;

```

Slika 60 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana(override) metoda „onCreate“, koja se poziva automatski prilikom prelaska iz aktivnosti „UserActivity“ ili „AdminActivity“ u aktivnost „RegistrationActivity“, slika 61.

```

48  protected void onCreate(Bundle savedInstanceState) {...}

```

Slika 61 Prikaz override metode "onCreate"

Unutar metode „onCreate“ je postavljen osluškivač na tastere „buttonVremeRodjenjaDeteta“, „buttonDatumRodjenjaDeteta“, „buttonVremeRodjenjaOca“, „buttonDatumRodjenjaOca“, „buttonVremeRodjenjaMajke“, „buttonDatumRodjenjaMajke“ i „buttonDodaj“, gde će osluškivač u zavisnosti od događaja pozvati odgovarajuću nadglasanu(override) metodu „onClick“ koja će se izvršiti, slika 62.

```

94      buttonVremeRodjenjaDeteta.setOnClickListener(new View.OnClickListener() {
95          @Override
96          public void onClick(View v) {...}
113      });
114
115      Calendar calendar = Calendar.getInstance();
116      Dgodina = calendar.get(Calendar.YEAR);
117      Dmesec = calendar.get(Calendar.MONTH);
118      Ddan = calendar.get(Calendar.DAY_OF_MONTH);
119
120      buttonDatumRodjenjaDeteta.setOnClickListener(new View.OnClickListener() {
121          @Override
122          public void onClick(View v) {...}
130      });
131      setListener = new DatePickerDialog.OnDateSetListener() {...};
142
143      buttonVremeRodjenjaOca.setOnClickListener(new View.OnClickListener() {
144          @Override
145          public void onClick(View v) {...}
162      });
163
164      Ogodina = calendar.get(Calendar.YEAR);
165      Omesec = calendar.get(Calendar.MONTH);
166      Odan = calendar.get(Calendar.DAY_OF_MONTH);
167
168      buttonDatumRodjenjaOca.setOnClickListener(new View.OnClickListener() {
169          @Override
170          public void onClick(View v) {...}
178      });
179      setListener2 = new DatePickerDialog.OnDateSetListener() {...};
190
191      buttonVremeRodjenjaMajke.setOnClickListener(new View.OnClickListener() {
192          @Override
193          public void onClick(View v) {...}
210      });
211
212      Mgodina = calendar.get(Calendar.YEAR);
213      Mmesec = calendar.get(Calendar.MONTH);
214      Mdan = calendar.get(Calendar.DAY_OF_MONTH);
215
216      buttonDatumRodjenjaMajke.setOnClickListener(new View.OnClickListener() {
217          @Override
218          public void onClick(View v) {...}
226      });
227      setListener3 = new DatePickerDialog.OnDateSetListener() {...};
238
239      buttonDodaj.setOnClickListener(new View.OnClickListener() {
240          @Override
241          public void onClick(View v) {...}
283      });

```

Slika 62 Prikaz osluškivača na tastere klase

Osluškivač na „buttonVremeRodjenjaDeteta“, „buttonDatumRodjenjaDeteta“, „buttonVremeRodjenjaOca“, „buttonDatumRodjenjaOca“, „buttonVremeRodjenjaMajke“ i „buttonDatumRodjenjaMajke“ poziva odgovarajuću nadglasanu metodu „onClick“ koja je namenjene za funkcionisanje datePickerDialog-a i timePickerDialog-a.

Osluškivač na „buttonDodaj“ poziva nadglasanu(override) metodu „onClick“ koja preuzima korisničke podatke iz editText polja i smešta ih u vidu stringa u niz stringova(slika 63), gde se taj niz zatim prosleđuje kao parametar pozivu metode „addDataInMaticnaKnjigaRodjenih“, koja je namenjena za smeštanje podataka u bazu podataka(slika 64).

```

246      podatak[0] = editTextImeDeteta.getText().toString();
247      podatak[1] = editTextPrezimeDeteta.getText().toString();
248      podatak[2] = editTextJMBGdeteta.getText().toString();
249      podatak[3] = editTextPolDeteta.getText().toString();
250      podatak[4] = vremeRodjenjaDeteta;
251      podatak[5] = datumRodjenjaDeteta;
252      podatak[6] = editTextMestoRodjenjaDeteta.getText().toString();
253      podatak[7] = editTextOpstinaRodjenjaDeteta.getText().toString();
254      podatak[8] = editTextDrzavaRodjenjaDeteta.getText().toString();
255      podatak[9] = editTextDrzavljanstvoDeteta.getText().toString();
256      podatak[10] = editTextImeOca.getText().toString();
257      podatak[11] = editTextPrezimeOca.getText().toString();
258      podatak[12] = editTextJMBGOca.getText().toString();
259      podatak[13] = editTextPolOca.getText().toString();
260      podatak[14] = vremeRodjenjaOca;
261      podatak[15] = datumRodjenjaOca;
262      podatak[16] = editTextMestoRodjenjaOca.getText().toString();
263      podatak[17] = editTextOpstinaRodjenjaOca.getText().toString();
264      podatak[18] = editTextDrzavaRodjenjaOca.getText().toString();
265      podatak[19] = editTextDrzavljanstvoOca.getText().toString();
266      podatak[20] = editTextPrebivalisteOca.getText().toString();
267      podatak[21] = editTextAdresaOca.getText().toString();
268      podatak[22] = editTextImeMajke.getText().toString();
269      podatak[23] = editTextPrezimeMajke.getText().toString();
270      podatak[24] = editTextJMBGMajke.getText().toString();
271      podatak[25] = editTextPolMajke.getText().toString();
272      podatak[26] = vremeRodjenjaMajke;
273      podatak[27] = datumRodjenjaMajke;
274      podatak[28] = editTextMestoRodjenjaMajke.getText().toString();
275      podatak[29] = editTextOpstinaRodjenjaMajke.getText().toString();
276      podatak[30] = editTextDrzavaRodjenjaMajke.getText().toString();
277      podatak[31] = editTextDrzavljanstvoMajke.getText().toString();
278      podatak[32] = editTextPrebivalisteMajke.getText().toString();
279      podatak[33] = editTextAdresaMajke.getText().toString();

```

Slika 63 Prikaz vraćanja vrednosti i čuvanja, unutar metode „onClick“ koju poziva osluškivač na "buttonDodaj"

```

281      databaseHelper.addDataInMaticnaKnjigaRodjenih(podatak);

```

Slika 64 Prikaz poziva metode "addDataInMaticnaKnjigaRodjenih", unutar metode „onClick“ koju poziva osluškivač na "buttonDodaj"

### 3.3.4 Klasa „UpdateActivity”

Klasa „UpdateActivity” nasleđuje klasu „AppCompatActivity”, slika 65.

```
27 public class UpdateActivity extends AppCompatActivity {
```

Slika 65 Prikaz klase "UpdateActivity" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_update.xml”(slika 66) i namenjena je za preuzimanje korisničkih podataka iz baze podataka i ažuriranje istih.

```
57 setContentView(R.layout.activity_update);
```

Slika 66 Prikaz naziva layout-a koji koristi klasa "UpdateActivity"

U prvom delu ove klase se nalaze potrebne deklaracije, slika 67.

```
28 EditText editTextImeDeteta, editTextPrezimeDeteta, editTextPolDeteta, editTextJMBGdeteta, editTextMestoRodjenjaDeteta;
29 EditText editTextOpstinaRodjenjaDeteta, editTextDrzavaRodjenjaDeteta, editTextDrzavljanstvoDeteta;
30 Button buttonVremeRodjenjaDeteta, buttonDatumRodjenjaDeteta;
31 DatePickerDialog.OnDateSetListener setListener;
32 int Dsat = 0, Dminut = 0;
33 int Dgodina = 0, Dmesec = 0, Ddan = 0;
34 String vremeRodjenjaDeteta = "", datumRodjenjaDeteta = "";
35
36 EditText editTextImeOca, editTextPrezimeOca, editTextPolOca, editTextJMBGOca, editTextMestoRodjenjaOca, editTextOpstinaRodjenjaOca;
37 EditText editTextDrzavaRodjenjaOca, editTextDrzavljanstvoOca, editTextPrebivalisteOca, editTextAdresaOca;
38 Button buttonVremeRodjenjaOca, buttonDatumRodjenjaOca;
39 DatePickerDialog.OnDateSetListener setListener2;
40 int Osat = 0, Ominut = 0;
41 int Ogodina = 0, Omesec = 0, Odan = 0;
42 String vremeRodjenjaOca = "", datumRodjenjaOca = "";
43
44 EditText editTextImeMajke, editTextPrezimeMajke, editTextPolMajke, editTextJMBGMajke, editTextMestoRodjenjaMajke, editTextOpstinaRodjenjaMajke;
45 EditText editTextDrzavaRodjenjaMajke, editTextDrzavljanstvoMajke, editTextPrebivalisteMajke, editTextAdresaMajke;
46 Button buttonVremeRodjenjaMajke, buttonDatumRodjenjaMajke;
47 DatePickerDialog.OnDateSetListener setListener3;
48 int Msat = 0, Mminut = 0;
49 int Mgodina = 0, Mmesec = 0, Mdan = 0;
50 String vremeRodjenjaMajke = "", datumRodjenjaMajke = "";
51
52 Button buttonAzuriraj, buttonIzbrisi;
53 DatabaseHelper db;
54 String[] podatak = new String[34];
55 String imeDeteta = "", prezimeDeteta = "", jmbgDeteta = "";
```

Slika 67 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana(override) metoda „onCreate”, koja se poziva automatski prilikom prelaska iz aktivnosti „CustomAdapter” u aktivnost „UpdateActivity”, slika 68.

```
58 protected void onCreate(Bundle savedInstanceState) { ... }
```

Slika 68 Prikaz override metode "onCreate"

```

107
108
109
126
127
128
129
130
131
132
133
134
135
143
144
155
156
157
158
175
176
177
178
179
180
181
182
183
191
192
203
204
205
206
223
224
225
226
227
228
229
230
231
239
240
251
252
253
254
255
256
257
296
297
298
299
300
303

buttonVremeRodjenjaDeteta.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {...}
});

Calendar calendar = Calendar.getInstance();
Dgodina = calendar.get(Calendar.YEAR);
Dmesec = calendar.get(Calendar.MONTH);
Ddan = calendar.get(Calendar.DAY_OF_MONTH);

buttonDatumRodjenjaDeteta.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {...}
});
setListener = new DatePickerDialog.OnDateSetListener() {...};

buttonVremeRodjenjaOca.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {...}
});

Ogodina = calendar.get(Calendar.YEAR);
Omesec = calendar.get(Calendar.MONTH);
Odan = calendar.get(Calendar.DAY_OF_MONTH);

buttonDatumRodjenjaOca.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {...}
});
setListener2 = new DatePickerDialog.OnDateSetListener() {...};

buttonVremeRodjenjaMajke.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {...}
});

Mgodina = calendar.get(Calendar.YEAR);
Mmesec = calendar.get(Calendar.MONTH);
Mdan = calendar.get(Calendar.DAY_OF_MONTH);

buttonDatumRodjenjaMajke.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {...}
});
setListener3 = new DatePickerDialog.OnDateSetListener() {...};

ActionBar actionBar = getSupportActionBar();
actionBar.setTitle("NASLOV"); // promeniti

buttonAzuriraj.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {...}
});

buttonIzbrisi.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) { confirmDialog(); }
});

```

Slika 69 Prikaz osluškivača na tastere



Osluškivač na „buttonVremeRodjenjaDeteta“, „buttonDatumRodjenjaDeteta“, „buttonVremeRodjenjaOca“, „buttonDatumRodjenjaOca“, „buttonVremeRodjenjaMajke“ i „buttonDatumRodjenjaMajke“ poziva odgovarajuću nadglasanu metodu „onClick“ koja je namenjene za funkcionisanje datePickerDialog-a i timePickerDialog-a, slika 69.

Osluškivač na „buttonAzuriraj“ poziva nadglasanu(override) metodu „onClick“ koja preuzima korisničke podatke iz editText polja i smešta ih u vidu stringa u niz stringova(slika 70), gde se taj niz zatim prosleđuje kao parametar pozivu metode „updateDataInMaticnaKnjigaRodjenih“, koja je namenjena za ažuriranje podataka u bazu podataka(slika 71).

```

246 podatak[0] = editTextImeDeteta.getText().toString();
247 podatak[1] = editTextPrezimeDeteta.getText().toString();
248 podatak[2] = editTextJMBGdeteta.getText().toString();
249 podatak[3] = editTextPolDeteta.getText().toString();
250 podatak[4] = vremeRodjenjaDeteta;
251 podatak[5] = datumRodjenjaDeteta;
252 podatak[6] = editTextMestoRodjenjaDeteta.getText().toString();
253 podatak[7] = editTextOpstinaRodjenjaDeteta.getText().toString();
254 podatak[8] = editTextDrzavaRodjenjaDeteta.getText().toString();
255 podatak[9] = editTextDrzavljanstvoDeteta.getText().toString();
256 podatak[10] = editTextImeOca.getText().toString();
257 podatak[11] = editTextPrezimeOca.getText().toString();
258 podatak[12] = editTextJMBGOca.getText().toString();
259 podatak[13] = editTextPolOca.getText().toString();
260 podatak[14] = vremeRodjenjaOca;
261 podatak[15] = datumRodjenjaOca;
262 podatak[16] = editTextMestoRodjenjaOca.getText().toString();
263 podatak[17] = editTextOpstinaRodjenjaOca.getText().toString();
264 podatak[18] = editTextDrzavaRodjenjaOca.getText().toString();
265 podatak[19] = editTextDrzavljanstvoOca.getText().toString();
266 podatak[20] = editTextPrebivalisteOca.getText().toString();
267 podatak[21] = editTextAdresaOca.getText().toString();
268 podatak[22] = editTextImeMajke.getText().toString();
269 podatak[23] = editTextPrezimeMajke.getText().toString();
270 podatak[24] = editTextJMBGMajke.getText().toString();
271 podatak[25] = editTextPolMajke.getText().toString();
272 podatak[26] = vremeRodjenjaMajke;
273 podatak[27] = datumRodjenjaMajke;
274 podatak[28] = editTextMestoRodjenjaMajke.getText().toString();
275 podatak[29] = editTextOpstinaRodjenjaMajke.getText().toString();
276 podatak[30] = editTextDrzavaRodjenjaMajke.getText().toString();
277 podatak[31] = editTextDrzavljanstvoMajke.getText().toString();
278 podatak[32] = editTextPrebivalisteMajke.getText().toString();
279 podatak[33] = editTextAdresaMajke.getText().toString();

```

Slika 70 Prikaz vraćanja vrednosti i čuvanja, unutar metode „onClick“ koju poziva osluškivač na "buttonAzuriraj"

```

294 db.updateDataInMaticnaKnjigaRodjenih(podatak);

```

Slika 71 Prikaz poziva metode "updateDataInMaticnaKnjigaRodjenih", unutar metode „onClick“ koju poziva osluškivač na "buttonAzuriraj"



Osluškivač na „buttonIzbrisi“ poziva nadglasanu(override) metodu „OnClick“ koja poziva pomoćnu metodu „confirmDialog“, slika 72.

```

298 buttonIzbrisi.setOnClickListener(new View.OnClickListener() {
299     @Override
300     public void onClick(View v) {
301         confirmDialog();
302     }
303 });

```

Slika 72 Prikaz definicije override metode "onClick", koju poziva osluškivač na "buttonIzbrisi"

U trećem delu ove klase se nalaze pomoćne metode „getAndSetIntentData“, „dataToList“ i „confirmDialog“, slika 73.

```

308 void getAndSetIntentData() {...}
357
358 void dataToList(String jmbg) {...}
402
403 void confirmDialog() {...}

```

Slika 73 Prikaz pomoćnih metoda

Metoda „dataToList“ poziva metodu „readOneDataFromMaticnaKnjigaRodjenih“ koja vraća cursor sa pročitanim podacima iz baze podataka(slika 74), a zatim „dataToList“ metoda vadi podatke iz cursora i čuva ih u niz stringova(slika 75).

```

359 Cursor cursor = db.readOneDataFromMaticnaKnjigaRodjenih(jmbg);

```

Slika 74 Prikaz poziva metode "readOneDataFromMaticnaKnjigaRodjenih" koju poziva metoda "dataToList"

```

365         podatak[0] = cursor.getString( columnIndex: 0);
366         podatak[1] = cursor.getString( columnIndex: 1);
367         podatak[2] = cursor.getString( columnIndex: 2);
368         podatak[3] = cursor.getString( columnIndex: 3);
369         podatak[4] = cursor.getString( columnIndex: 4);
370         podatak[5] = cursor.getString( columnIndex: 5);
371         podatak[6] = cursor.getString( columnIndex: 6);
372         podatak[7] = cursor.getString( columnIndex: 7);
373         podatak[8] = cursor.getString( columnIndex: 8);
374         podatak[9] = cursor.getString( columnIndex: 9);
375         podatak[10] = cursor.getString( columnIndex: 10);
376         podatak[11] = cursor.getString( columnIndex: 11);
377         podatak[12] = cursor.getString( columnIndex: 12);
378         podatak[13] = cursor.getString( columnIndex: 13);
379         podatak[14] = cursor.getString( columnIndex: 14);
380         podatak[15] = cursor.getString( columnIndex: 15);
381         podatak[16] = cursor.getString( columnIndex: 16);
382         podatak[17] = cursor.getString( columnIndex: 17);
383         podatak[18] = cursor.getString( columnIndex: 18);
384         podatak[19] = cursor.getString( columnIndex: 19);
385         podatak[20] = cursor.getString( columnIndex: 20);
386         podatak[21] = cursor.getString( columnIndex: 21);
387         podatak[22] = cursor.getString( columnIndex: 22);
388         podatak[23] = cursor.getString( columnIndex: 23);
389         podatak[24] = cursor.getString( columnIndex: 24);
390         podatak[25] = cursor.getString( columnIndex: 25);
391         podatak[26] = cursor.getString( columnIndex: 26);
392         podatak[27] = cursor.getString( columnIndex: 27);
393         podatak[28] = cursor.getString( columnIndex: 28);
394         podatak[29] = cursor.getString( columnIndex: 29);
395         podatak[30] = cursor.getString( columnIndex: 30);
396         podatak[31] = cursor.getString( columnIndex: 31);
397         podatak[32] = cursor.getString( columnIndex: 32);
398         podatak[33] = cursor.getString( columnIndex: 33);

```

Slika 75 Prikaz vraćanja vrednosti cursor-a i čuvanja, unutar metode "dataToList"

Metoda „onCreate“ poziva metodu „getAndSetIntentData“, gde ona na početku poziva metodu „dataToList“(slika 76),a nakon toga uzima podatke iz niza stringova i smešta ih u EditText polja(slika 77).

```

311         dataToList(jmbgDeteta);

```

Slika 76 Prikaz poziva metode "dataToList" koju poziva metoda "getAndSetIntentData"

```
313 editTextImeDeteta.setText(podatak[0]);
314 editTextPrezimeDeteta.setText(podatak[1]);
315 editTextJMBGdeteta.setText(podatak[2]);
316 editTextPolDeteta.setText(podatak[3]);
317 buttonVremeRodjenjaDeteta.setText(podatak[4]); // proveriti
318 vremeRodjenjaDeteta = podatak[4];
319 buttonDatumRodjenjaDeteta.setText(podatak[5]); // proveriti
320 datumRodjenjaDeteta = podatak[5];
321 editTextMestoRodjenjaDeteta.setText(podatak[6]);
322 editTextOpstinaRodjenjaDeteta.setText(podatak[7]);
323 editTextDrzavaRodjenjaDeteta.setText(podatak[8]);
324 editTextDrzavljanstvoDeteta.setText(podatak[9]);
325 editTextImeOca.setText(podatak[10]);
326 editTextPrezimeOca.setText(podatak[11]);
327 editTextJMBGOca.setText(podatak[12]);
328 editTextPolOca.setText(podatak[13]);
329 buttonVremeRodjenjaOca.setText(podatak[14]); // proveriti
330 vremeRodjenjaOca = podatak[14];
331 buttonDatumRodjenjaOca.setText(podatak[15]); // proveriti
332 datumRodjenjaOca = podatak[15];
333 editTextMestoRodjenjaOca.setText(podatak[16]);
334 editTextOpstinaRodjenjaOca.setText(podatak[17]);
335 editTextDrzavaRodjenjaOca.setText(podatak[18]);
336 editTextDrzavljanstvoOca.setText(podatak[19]);
337 editTextPrebivalisteOca.setText(podatak[20]);
338 editTextAdresaOca.setText(podatak[21]);
339 editTextImeMajke.setText(podatak[22]);
340 editTextPrezimeMajke.setText(podatak[23]);
341 editTextJMBGMajke.setText(podatak[24]);
342 editTextPolMajke.setText(podatak[25]);
343 buttonVremeRodjenjaMajke.setText(podatak[26]); // proveriti
344 vremeRodjenjaMajke = podatak[26];
345 buttonDatumRodjenjaMajke.setText(podatak[27]); // proveriti
346 datumRodjenjaMajke = podatak[27];
347 editTextMestoRodjenjaMajke.setText(podatak[28]);
348 editTextOpstinaRodjenjaMajke.setText(podatak[29]);
349 editTextDrzavaRodjenjaMajke.setText(podatak[30]);
350 editTextDrzavljanstvoMajke.setText(podatak[31]);
351 editTextPrebivalisteMajke.setText(podatak[32]);
352 editTextAdresaMajke.setText(podatak[33]);
```

Slika 77 Prikaz setovanja EditText polja na određene vrednosti, unutar metode „getAndSetIntentData“

Metoda „confirmDialog“ je namenjena za prikazivanje „alert“ dijaloga i za brisanje određenih podataka iz baze podataka pozivom metode „deleteData“, ali samo ako je data pozitivna potvrda „alert“ dijaloga, slika 78.

```

403 void confirmDialog() {
404     imeDeteta = editTextImeDeteta.getText().toString();
405     prezimeDeteta = editTextPrezimeDeteta.getText().toString();
406     jmbgDeteta = editTextJMBGdeteta.getText().toString();
407
408     AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
409     builder.setTitle("Obrisati dete iz matične knjige rođenih");
410     builder.setMessage("Da li želite da obrišete " + imeDeteta + prezimeDeteta + ", JMBG: " + jmbgDeteta + " ?");
411     builder.setPositiveButton( text: "Da", new DialogInterface.OnClickListener() {
412         @Override
413         public void onClick(DialogInterface dialog, int which) {
414             db.deleteData(jmbgDeteta);
415             finish();
416         }
417     });
418     builder.setNegativeButton( text: "Ne", new DialogInterface.OnClickListener() {
419         @Override
420         public void onClick(DialogInterface dialog, int which) {
421
422         }
423     });
424     builder.create().show();
425 }

```

Slika 78 Prikaz definicije metode "confirmDialog"

### 3.3.5 Klasa „UpdateActivity2“

Klasa "UpdateActivity2" nasleđuje klasu „AppCompatActivity“, slika 79.

```

15 public class UpdateActivity2 extends AppCompatActivity {

```

Slika 79 Prikaz klase "UpdateActivity2" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_update2.xml“ (slika 80) i namenjena je za preuzimanje korisničkih podataka iz baze podataka i ažuriranje istih.

```

26 setContentView(R.layout.activity_update2);

```

Slika 80 Prikaz naziva layout-a koji koristi klasa "UpdateActivity2"

U prvom delu ove klase se nalaze potrebne deklaracije, slika 81.

```

16 EditText editTextIme, editTextPrezime, editTextKorisnickoIme, editTextLozinka, editTextPotvrdaLozinke, editTextJMBG;
17 Button buttonAzuriraj, buttonIzbrisi;
18
19 DatabaseHelper db;
20
21 String ime = "", prezime = "", korisnickoIme = "", lozinka = "", potvrda = "", jmbg = "", jmbgRoditelja = "";

```

Slika 81 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana (override) metoda „onCreate“, koja se poziva automatski prilikom prelaska iz aktivnosti „CustomAdapter2“ u aktivnost „UpdateActivity2“, slika 82.

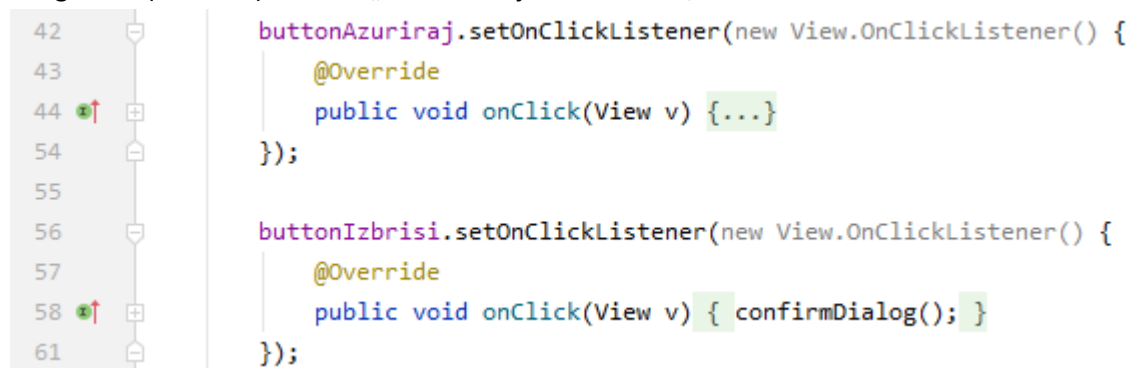
```

24 protected void onCreate(Bundle savedInstanceState) {

```

Slika 82 Prikaz override metode "onCreate"

Unutar metode „onCreate“ je postavljen oslušivač na tastere „buttonAzuriraj“ i „buttonIzbrisi“, gde će oslušivač u zavisnosti od događaja pozvati odgovarajuću nadglasanu(override) metodu „onClick“ koja će se izvršiti, slika 83.



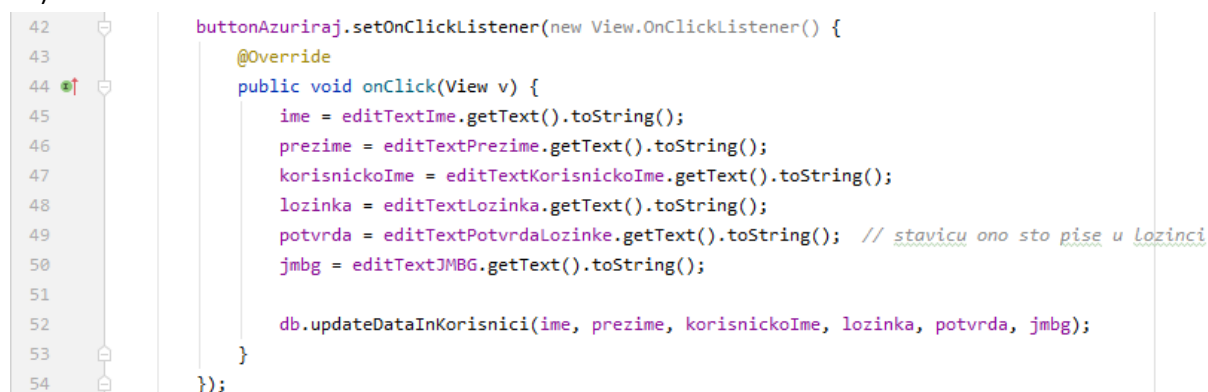
```

42 buttonAzuriraj.setOnClickListener(new View.OnClickListener() {
43     @Override
44     public void onClick(View v) {...}
54 });
56 buttonIzbrisi.setOnClickListener(new View.OnClickListener() {
57     @Override
58     public void onClick(View v) { confirmDialog(); }
61 });

```

Slika 83 Prikaz oslušivača na tastere

Oslušivač na „buttonAzuriraj“ poziva nadglasanu(override) metodu „onClick“ koja preuzima korisničke podatke iz editText polja i prosleđuje ih kao parametar pozivu metode „updateDataInKorisnici“, koja je namenjena za ažuriranje podataka u bazu podataka(slika 84).



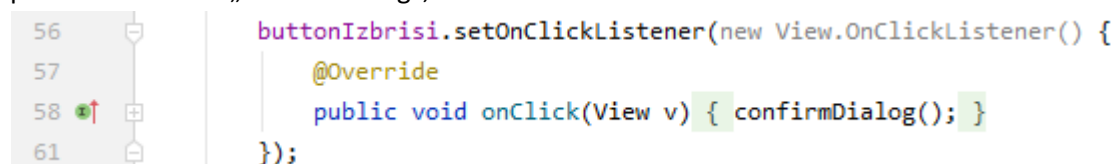
```

42 buttonAzuriraj.setOnClickListener(new View.OnClickListener() {
43     @Override
44     public void onClick(View v) {
45         ime = editTextIme.getText().toString();
46         prezime = editTextPrezime.getText().toString();
47         korisnickoIme = editTextKorisnickoIme.getText().toString();
48         lozinka = editTextLozinka.getText().toString();
49         potvrda = editTextPotvrdaLozinke.getText().toString(); // stavicu ono sto pise u lozinci
50         jmbg = editTextJMBG.getText().toString();
52         db.updateDataInKorisnici(ime, prezime, korisnickoIme, lozinka, potvrda, jmbg);
53     }
54 });

```

Slika 84 Prikaz definicije override metode "onClick" koju poziva oslušivač na "buttonAzuriraj"

Oslušivač na „buttonIzbrisi“ poziva nadglasanu(override) metodu „onClick“ koja poziva pomoćnu metodu „confirmDialog“, slika 85.



```

56 buttonIzbrisi.setOnClickListener(new View.OnClickListener() {
57     @Override
58     public void onClick(View v) { confirmDialog(); }
61 });

```

Slika 75 Prikaz definicije override metode „onClick“ koju poziva oslušivač na "buttonIzbrisi"

U trećem delu ove klase se nalaze pomoćne metode „getAndSetIntentData“, „dataToList“ i „confirmDialog“, slika 86.

```

66      void getAndSetIntentData() {...}
81
82      void dataToList(String jmbgRoditelja) {...}
98
99      void confirmDialog() {...}

```

Slika 86 Prikaz pomoćnih metoda

Metoda „dataToList“ poziva metodu „readOneDataFromKorisnici“ koja vraća cursor sa pročitanim podacima iz baze podataka(slika 87), a zatim „dataToList“ metoda vadi podatke iz cursora i čuva ih u promenljive(slika 88).

```

83      Cursor cursor = db.readOneDataFromKorisnici(jmbgRoditelja);

```

Slika 87 Prikaz poziva metode "readOneDataFromKorisnici", unutar metode "dataToList"

```

89      ime = cursor.getString( columnIndex: 0);
90      prezime = cursor.getString( columnIndex: 1);
91      jmbg = cursor.getString( columnIndex: 2);
92      korisnickoIme = cursor.getString( columnIndex: 3);
93      lozinka = cursor.getString( columnIndex: 4);
94      potvrda = cursor.getString( columnIndex: 4);

```

Slika 88 Prikaz vraćanja vrednosti cursor-a i čuvanja, unutar metode „dataToList“

Metoda „onCreate“ poziva metodu „getAndSetIntentData“, gde ona na početku poziva metodu „dataToList“(slika 89), a nakon toga uzima podatke iz promenljivih i smešta ih u EditText polja(slika 90).

```

69      dataToList(jmbgRoditelja);

```

Slika 89 Prikaz poziva metode "dataToList" unutar metode "getAndSetIntentData"

```

71      editTextIme.setText(ime);
72      editTextPrezime.setText(prezime);
73      editTextKorisnickoIme.setText(korisnickoIme);
74      editTextLozinka.setText(lozinka);
75      editTextPotvrdaLozinke.setText(potvrda);
76      editTextJMBG.setText(jmbg);

```

Slika 90 Prikaz setovanja EditText polja na određene vrednost, unutar metode „getAndSetIntentData“

Metoda „confirmDialog“ je namenjena za prikazivanje „alert“ dijaloga i za brisanje određenih podataka iz baze podataka pozivom metode „deleteDataFromKorisnici“, ali samo ako je data pozitivna potvrda „alert“ dijaloga, slika 91.

```

99  void confirmDialog() {
100      String imeRoditelja = editTextIme.getText().toString();
101      String prezimeRoditelja = editTextPrezime.getText().toString();
102      String jmbgRoditelja = editTextJMBG.getText().toString();
103
104      AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
105      builder.setTitle("Obrisati nalog roditelja");
106      builder.setMessage("Da li želite da obrišete " + imeRoditelja + " " + prezimeRoditelja + ", JMBG: " + jmbgRoditelja + " ?");
107      builder.setPositiveButton( text: "Da", new DialogInterface.OnClickListener() {
108          @Override
109          public void onClick(DialogInterface dialog, int which) {
110              db.deleteDataFromKorisnici(jmbgRoditelja);
111              finish();
112          }
113      });
114      builder.setNegativeButton( text: "Ne", new DialogInterface.OnClickListener() {
115          @Override
116          public void onClick(DialogInterface dialog, int which) {
117
118          }
119      });
120      builder.create().show();
121  }

```

Slika 91 Prikaz definicije metode "confirmDialog"

### 3.3.6 Klasa „CustomAdapter“

Klasa „CustomAdapter“ nasleđuje klasu „RecyclerView.Adapter“, slika 92.

```

18  public class CustomAdapter extends RecyclerView.Adapter<CustomAdapter.MyViewHolder> {

```

Slika 92 Prikaz klase "CustomAdapter" kako nasleđuje klasu "RecyclerView.Adapter"

Povezana je sa xml fajlom „row\_data.xml“ (slika 93) i namenjena je za kreiranje korisničkog adaptera koji se koristi za podešavanja RecyclerView u klasi „AdminActivity“ (slika 94).

```

38  View view = inflater.inflate(R.layout.row_data, parent, attachToRoot: false);

```

Slika 93 Prikaz kreiranja objekta klase View

```

54  recyclerView.setAdapter(customAdapter);

```

Slika 94 Prikaz podešavanja adapter-a

U prvom delu ove klase se nalaze potrebne deklaracije, slika 95.

```

20  private Context context;
21  private ArrayList imeDeteta, prezimeDeteta, jmbgDeteta;
22
23  private Activity activity;

```

Slika 95 Prikaz deklaracija polja



U drugom delu ove klase se nalazi konstruktor za inicijalizaciju navedenih deklaracija, slika 96.

```

25 CustomAdapter(Activity activity, Context context, ArrayList imeDeteta, ArrayList prezimeDeteta, ArrayList jmbgDeteta) {
26     this.context = context;
27     this.imeDeteta = imeDeteta;
28     this.prezimeDeteta = prezimeDeteta;
29     this.jmbgDeteta = jmbgDeteta;
30     this.activity = activity;
31 }

```

Slika 96 Prikaz definicije konstruktora klase

U trećem delu ove klase se nalaze nadglasane(override) metode „onCreateViewHolder“, „onBindViewHolder“ i „getItemCount“, slika 97.

```

33 @NonNull
34 @Override
35 public CustomAdapter.MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {...}
42
43 @Override
44 public void onBindViewHolder(@NonNull CustomAdapter.MyViewHolder holder, int position) {...}
62
63 @Override
64 public int getItemCount() {
65     return jmbgDeteta.size();
66 }

```

Slika 97 Prikaz override metoda

RecyclerView poziva metodu „onCreateViewHolder“, koja kreira i inicijalizuje ViewHolder, ali ne popunjava sadržaj prikaza, tj. ViewHolder nije još uvek vezan za određene podatke, slika 98.

```

35 public CustomAdapter.MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
36
37     LayoutInflater inflater = LayoutInflater.from(context);
38     View view = inflater.inflate(R.layout.row_data, parent, attachToRoot: false);
39
40     return new MyViewHolder(view);
41 }

```

Slika 98 Prikaz definicije override metode "onCreateViewHolder"



RecyclerView poziva metodu „onBindViewHolder“ kako bi ViewHolder povezao sa podacima, koje prvo metoda preuzme, a zatim koristi za popunjavanje ViewHolder-a, slika 99.

```

44 public void onBindViewHolder(@NonNull CustomAdapter.MyViewHolder holder, int position) {
45     holder.imeDeteta.setText(String.valueOf(imeDeteta.get(position)));
46     holder.prezimeDeteta.setText(String.valueOf(prezimeDeteta.get(position)));
47     holder.jmbgDeteta.setText(String.valueOf(jmbgDeteta.get(position)));
48
49     holder.mainLayout.setOnClickListener(new View.OnClickListener() {
50         @Override
51         public void onClick(View v) {
52             Intent intent = new Intent(context, UpdateActivity.class);
53             intent.putExtra( name: "imeDeteta", String.valueOf(imeDeteta.get(position)));
54             intent.putExtra( name: "prezimeDeteta", String.valueOf(prezimeDeteta.get(position)));
55             intent.putExtra( name: "jmbgDeteta", String.valueOf(jmbgDeteta.get(position)));
56
57             activity.startActivityForResult(intent, requestCode: 1);
58         }
59     });
60 }
61

```

Slika 99 Prikaz definicije override metode "onBindViewHolder"

RecyclerView poziva metodu „getItemCount“ kako bi dobio veličinu skupa podataka, slika 100.

```

64 public int getItemCount() {
65     return jmbgDeteta.size();
66 }

```

Slika 100 Prikaz definicije override metode "getItemCount"

ViewHolder koji se kreira i inicijalizuje se zove MyViewHolder, slika 101.

```

68 public class MyViewHolder extends RecyclerView.ViewHolder {
69
70     TextView imeDeteta, prezimeDeteta, jmbgDeteta;
71
72     LinearLayout mainLayout;
73
74     public MyViewHolder(@NonNull View itemView) {
75         super(itemView);
76
77         imeDeteta = itemView.findViewById(R.id.textViewIme);
78         prezimeDeteta = itemView.findViewById(R.id.textViewPrezime);
79         jmbgDeteta = itemView.findViewById(R.id.textViewJMBG);
80
81         mainLayout = itemView.findViewById(R.id.mainLayout);
82     }
83 }

```

Slika 101 Prikaz definicije klase "MyViewHolder"

### 3.3.7 Klasa „CustomAdapter2”

Klasa „CustomAdapter2” nasleđuje klasu „RecyclerView.Adapter”, slika 102.

```
18 public class CustomAdapter2 extends RecyclerView.Adapter<CustomAdapter2.MyViewHolder> {
```

Slika 102 Prikaz klase "CustomAdapter2" kako nasleđuje klasu "RecyclerView.Adapter"

Povezana je sa xml fajlom „row\_data2.xml”(slika 103) i namenjena je za kreiranje korisničkog adaptera koji se koristi za podešavanja RecyclerView u klasi „AdminActivity2”(slika 104).

```
38 View view = inflater.inflate(R.layout.row_data2, parent, attachToRoot: false);
```

Slika 103 Prikaz kreiranja objekta klase View

```
54 recyclerView.setAdapter(customAdapter);
```

Slika 104 Prikaz podešavanja adapter-a

U prvom delu ove klase se nalaze potrebne deklaracije, slika 105.

```
20 private Context context;
21 private ArrayList imeRoditelja, prezimeRoditelja, jmbgRoditelja;
22
23 private Activity activity;
```

Slika 105 Prikaz deklaracija polja

U drugom delu ove klase se nalazi konstruktor za inicijalizaciju navedenih deklaracija, slika 106.

```
25 CustomAdapter2(Activity activity, Context context, ArrayList imeRoditelja, ArrayList prezimeRoditelja, ArrayList jmbgRoditelja) {
26     this.context = context;
27     this.imeRoditelja = imeRoditelja;
28     this.prezimeRoditelja = prezimeRoditelja;
29     this.jmbgRoditelja = jmbgRoditelja;
30     this.activity = activity;
31 }
```

Slika 106 Prikaz definicije konstruktora klase

U trećem delu ove klase se nalaze nadglasane(override) metode „onCreateViewHolder”, „onBindViewHolder” i „getItemCount”, slika 107.

```
33 @NonNull
34 @Override
35 public CustomAdapter2.MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {...}
42
43 @Override
44 public void onBindViewHolder(@NonNull CustomAdapter2.MyViewHolder holder, int position) {...}
62
63 @Override
64 public int getItemCount() { return jmbgRoditelja.size(); }
```

Slika 107 Prikaz override metoda

RecyclerView poziva metodu „onCreateViewHolder“, koja kreira i inicijalizuje ViewHolder, ali ne popunjava sadržaj prikaza, tj. ViewHolder nije još uvek vezan za određene podatke, slika 108.

```

35 public CustomAdapter2.MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
36
37     LayoutInflater inflater = LayoutInflater.from(context);
38     View view = inflater.inflate(R.layout.row_data2, parent, attachToRoot: false);
39
40     return new MyViewHolder(view);
41 }

```

Slika 108 Prikaz definicije override metode "onCreateViewHolder"

RecyclerView poziva metodu „onBindViewHolder“ kako bi ViewHolder povezao sa podacima, koje prvo metoda preuzme, a zatim koristi za popunjavanje ViewHolder-a, slika 109.

```

44 public void onBindViewHolder(@NonNull CustomAdapter2.MyViewHolder holder, int position) {
45     holder.imeRoditelja.setText(String.valueOf(imeRoditelja.get(position)));
46     holder.prezimeRoditelja.setText(String.valueOf(prezimeRoditelja.get(position)));
47     holder.jmbgRoditelja.setText(String.valueOf(jmbgRoditelja.get(position)));
48
49     holder.mainLayout2.setOnClickListener(new View.OnClickListener() {
50         @Override
51         public void onClick(View v) {
52             Intent intent = new Intent(context, UpdateActivity2.class);
53             intent.putExtra("imeRoditelja", String.valueOf(imeRoditelja.get(position)));
54             intent.putExtra("prezimeRoditelja", String.valueOf(prezimeRoditelja.get(position)));
55             intent.putExtra("name: jmbgRoditelja", String.valueOf(jmbgRoditelja.get(position)));
56
57             activity.startActivityForResult(intent, requestCode: 1);
58         }
59     });
60 }
61 }

```

Slika 109 Prikaz definicije override metode "onBindViewHolder"

RecyclerView poziva metodu „getItemCount“ kako bi dobio veličinu skupa podataka, slika 110.

```

64 public int getItemCount() { return jmbgRoditelja.size(); }

```

Slika 110 Prikaz definicije override metode "getItemCount"

ViewHolder koji se kreira i inicijalizuje se zove MyViewHolder, slika 111.

```

68 public class MyViewHolder extends RecyclerView.ViewHolder {
69
70     TextView imeRoditelja, prezimeRoditelja, jmbgRoditelja;
71
72     LinearLayout mainLayout2;
73
74     public MyViewHolder(@NonNull View itemView) {
75         super(itemView);
76
77         imeRoditelja = itemView.findViewById(R.id.textViewImeRoditelja);
78         prezimeRoditelja = itemView.findViewById(R.id.textViewPrezimeRoditelja);
79         jmbgRoditelja = itemView.findViewById(R.id.textViewJMBGRoditelja);
80
81         mainLayout2 = itemView.findViewById(R.id.mainLayout2);
82     }
83 }

```

Slika 111 Prikaz definicije klase "MyViewHolder"

### 3.3.8 Klasa „MainAdminActivity“

Klasa "MainAdminActivity" nasleđuje klasu „AppCompatActivity“, slika 112.

```

10 public class MainAdminActivity extends AppCompatActivity {

```

Slika 112 Prikaz klase "MainAdminActivity" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_main\_admin.xml“ (slika 113) i namenjena je za prelazak u aktivnost „AdminActivity“ ili „AdminActivity2“.

```

16 setContentView(R.layout.activity_main_admin);

```

Slika 113 Prikaz naziva layout-a koji koristi klasa "AdminActivity2"

U prvom delu ove klase se nalaze potrebne deklaracije, slika 114.

```

11 Button buttonUpravljaKorisnicima, buttonUpravljaMKR, buttonLogoutAdmin;

```

Slika 114 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana (override) metoda „onCreate“, koja se poziva automatski prilikom prelaska iz aktivnosti „MainActivity“ u aktivnost „MainAdminActivity“, slika 115.

```

14 protected void onCreate(Bundle savedInstanceState) {...}

```

Slika 115 Prikaz override metode "onCreate"

Unutar metode „onCreate“ je postavljen oslušivač na tastere „buttonUpravljajKorisnicima“, „buttonUpravljajMKR“ i „buttonLogoutAdmin“, gde će oslušivač u zavisnosti od događaja pozvati odgovarajuću nadglasanu(override) metodu „onClick“ koja će se izvršiti, slika 116.

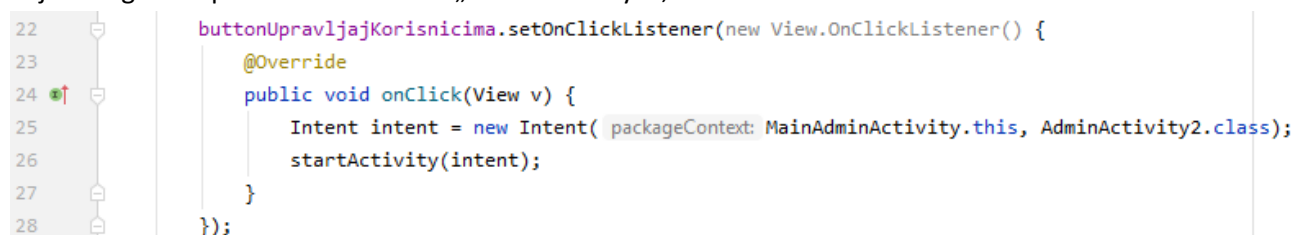


```

22 buttonUpravljajKorisnicima.setOnClickListener(new View.OnClickListener() {
23     @Override
24     public void onClick(View v) {...}
28 });
29
30 buttonUpravljajMKR.setOnClickListener(new View.OnClickListener() {
31     @Override
32     public void onClick(View v) {...}
36 });
37
38 buttonLogoutAdmin.setOnClickListener(new View.OnClickListener() {
39     @Override
40     public void onClick(View v) { finish(); }
43 });
  
```

Slika 116 Prikaz oslušivača na tastere

Oslušivač na „buttonUpravljajKorisnicima“ poziva nadglasanu(override) metodu „onClick“ koja omogućava prelazak u aktivnost „AdminActivity2“, slika 117.

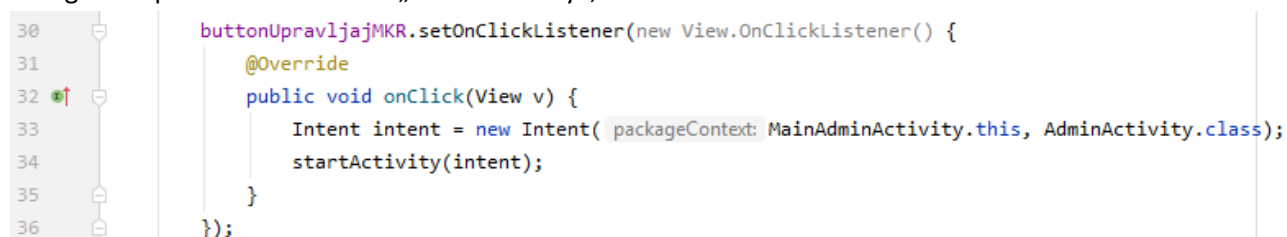


```

22 buttonUpravljajKorisnicima.setOnClickListener(new View.OnClickListener() {
23     @Override
24     public void onClick(View v) {
25         Intent intent = new Intent( packageContext: MainAdminActivity.this, AdminActivity2.class);
26         startActivity(intent);
27     }
28 });
  
```

Slika 117 Prikaz definicije override metode „onClick“ koju poziva oslušivač na "buttonUpravljajKorisnicima"

Oslušivač na „buttonUpravljajMKR“ poziva nadglasanu(override) metodu „onClick“ koja omogućava prelazak u aktivnost „AdminActivity“, slika 118.

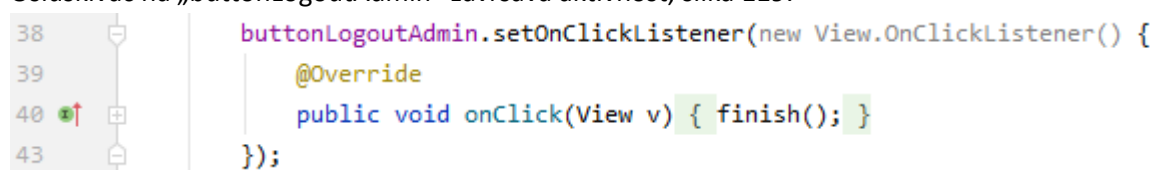


```

30 buttonUpravljajMKR.setOnClickListener(new View.OnClickListener() {
31     @Override
32     public void onClick(View v) {
33         Intent intent = new Intent( packageContext: MainAdminActivity.this, AdminActivity.class);
34         startActivity(intent);
35     }
36 });
  
```

Slika 118 Prikaz definicije override metode „onClick“ koju poziva oslušivač na "buttonUpravljajMKR"

Oslušivač na „buttonLogoutAdmin“ završava aktivnost, slika 119.



```

38 buttonLogoutAdmin.setOnClickListener(new View.OnClickListener() {
39     @Override
40     public void onClick(View v) { finish(); }
43 });
  
```

Slika 119 Prikaz definicije override metode „onClick“ koju poziva oslušivač na "buttonLogoutAdmin"

### 3.3.9 Klasa „AdminActivity”

Klasa „AdminActivity” nasleđuje klasu „AppCompatActivity”, slika 120.

```
18 public class AdminActivity extends AppCompatActivity {
```

Slika 120 Prikaz klase "AdminActivity" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_admin.xml”(slika 121) i namenjana je za prelazak u aktivnost „RegistrationActivity” i za podešavanje RecyclerView-a.

```
31 setContentView(R.layout.activity_admin);
```

Slika 121 Prikaz naziva layout-a koji koristi klasa "AdminActivity"

U prvom delu ove klase se nalaze potrebne deklaracije, slika 122.

```
19 RecyclerView recyclerView;
20 FloatingActionButton floatingActionButton;
21
22 DatabaseHelper databaseHelper;
23
24 ArrayList<String> imeDeteta, prezimeDeteta, jmbgDeteta;
25
26 CustomAdapter customAdapter;
```

Slika 122 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana(override) metoda „onCreate”, koja se poziva automatski prilikom prelaska iz aktivnosti „MainAdminActivity” u aktivnost „AdminActivity”, slika 123.

```
29 protected void onCreate(Bundle savedInstanceState) {...}
```

Slika 123 Prikaz override metode "onCreate"

Unutar metode „onCreate” je postavljen osluškivač na taster „floatingActionButton”, gde će osluškivač pozvati nadglasanu(override) metodu „onClick” koja će se izvršiti, slika 124.

```
36 floatingActionButton.setOnClickListener(new View.OnClickListener() {
37     @Override
38     public void onClick(View v) {...}
42 });
```

Slika 124 Prikaz osluškivača na taster

Osluškivač na „floatingActionButton“ poziva nadglasanu(override) metodu „OnClick“ koja omogućava prelazak u aktivnost „RegistrationActivity“, slika 125.

```

36 floatingActionButton.setOnClickListener(new View.OnClickListener() {
37
38     @Override
39     public void onClick(View v) {
40         Intent intent = new Intent( packageName: AdminActivity.this, RegistrationActivity.class );
41         startActivity(intent);
42     }
});

```

Slika 125 Prikaz definicije override metode „onClick“ koju poziva osluškivač na "floatingActionButton"

Metoda „dataToList“ sadrži poziv metode „readDataFromMaticnaKnjigaRodjenih“ koja vraća cursor sa podacima učitanim iz baze podataka, gde se ti podaci nakon toga čuvaju u listama, slika 126.

```

58 void dataToList() {
59     Cursor cursor = databaseHelper.readDataFromMaticnaKnjigaRodjenih();
60
61     if (cursor.getCount() == 0){
62         Toast.makeText( context: this, text: "Nema podataka", Toast.LENGTH_SHORT).show();
63     } else {
64         while (cursor.moveToNext()) {
65             imeDeteta.add(cursor.getString( columnIndex: 0));
66             prezimeDeteta.add(cursor.getString( columnIndex: 1));
67             jmbgDeteta.add(cursor.getString( columnIndex: 2));
68         }
69     }
70 }

```

Slika 126 Prikaz definicije metode "dataToList"

Nakon vraćanja iz pozvane metode, kreira se korisnički adapter i podešava se RecyclerView, slika 127.

```

52 customAdapter = new CustomAdapter( activity: AdminActivity.this, context: this, imeDeteta, prezimeDeteta, jmbgDeteta);
53
54 recyclerView.setAdapter(customAdapter);
55 recyclerView.setLayoutManager(new LinearLayoutManager( context: AdminActivity.this));

```

Slika 127 Prikaz kreiranja korisničkog adaptera i prikaz podešavanja RecyclerView

### 3.3.10 Klasa „AdminActivity2”

Klasa „AdminActivity2” nasleđuje klasu „AppCompatActivity”, slika 128.

```
18 public class AdminActivity2 extends AppCompatActivity {
```

Slika 88 Prikaz klase "AdminActivity2" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_admin2.xml”(slika 129) i namenjana je za prelazak u aktivnost „SignUpActivity” i za podešavanje RecyclerView-a.

```
31 setContentView(R.layout.activity_admin2);
```

Slika 929 Prikaz naziva layout-a koji koristi klasa "AdminActivity2"

U prvom delu ove klase se nalaze potrebne deklaracije, slika 130.

```
19 RecyclerView recyclerView;
20 FloatingActionButton floatingActionButton;
21
22 DatabaseHelper databaseHelper;
23
24 ArrayList<String> imeRoditelja, prezimeRoditelja, jmbgRoditelja;
25
26 CustomAdapter2 customAdapter;
```

Slika 130 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana(override) metoda „onCreate”, koja se poziva automatski prilikom prelaska iz aktivnosti „MainAdminActivity” u aktivnost „AdminActivity2”, slika 131.

```
29 protected void onCreate(Bundle savedInstanceState) {...}
```

Slika 1031 Prikaz override metode "onCreate"

Unutar metode „onCreate” je postavljen osluškivač na taster „floatingActionButton”, gde će osluškivač pozvati nadglasanu(override) metodu „onClick” koja će se izvršiti, slika 132.

```
36 floatingActionButton.setOnClickListener(new View.OnClickListener() {
37     @Override
38     public void onClick(View v) {...}
42 });
```

Slika 132 Prikaz osluškivača na taster



Osluškivač na „floatingActionButton“ poziva nadglasanu(override) metodu „OnClick“ koja omogućava prelazak u aktivnost „SignUpActivity“, slika 133.

```

36 floatingActionButton.setOnClickListener(new View.OnClickListener() {
37     @Override
38     public void onClick(View v) {
39         Intent intent = new Intent( packageContext: AdminActivity2.this, SignUpActivity.class );
40         startActivity(intent);
41     }
42 });

```

Slika 133 Prikaz definicije override metode „onClick“ koju poziva osluškivač na "floatingActionButton"

Takođe unutar metode „onCreate“ postoji poziv metode „dataToList“, slika 134.

```

50 dataToList();

```

Slika 134 Prikaz poziva metode "dataToList" unutar metode "onCreate"

Metoda „dataToList“ sadrži poziv metode „readDataFromKorisnici“ koja vraća cursor sa podacima učitanim iz baze podataka, gde se ti podaci nakon toga čuvaju u listama, slika 135.

```

58 void dataToList() {
59     Cursor cursor = databaseHelper.readDataFromKorisnici();
60
61     if (cursor.getCount() == 0){
62         Toast.makeText( context: this, text: "Nema podataka", Toast.LENGTH_SHORT).show();
63     } else {
64         while (cursor.moveToNext()) {
65             imeRoditelja.add(cursor.getString( columnIndex: 0));
66             prezimeRoditelja.add(cursor.getString( columnIndex: 1));
67             jmbgRoditelja.add(cursor.getString( columnIndex: 2));
68         }
69     }
70 }

```

Slika 135 Prikaz definicije metode "dataToList"

Nakon vraćanja iz pozvane metode, kreira se korisnički adapter i podešava se RecyclerView, slika 136.

```

52 customAdapter = new CustomAdapter2( activity: AdminActivity2.this, context: this, imeRoditelja, prezimeRoditelja, jmbgRoditelja);
53
54 recyclerView.setAdapter(customAdapter);
55 recyclerView.setLayoutManager(new LinearLayoutManager( context: AdminActivity2.this));

```

Slika 136 Prikaz kreiranja korisničkog adaptera i prikaz podešavanja RecyclerView

### 3.3.11 Klasa „SignUpActivity“

Klasa „SignUpActivity“ nasleđuje klasu „AppCompatActivity“, slika 137.

```
11 public class SignUpActivity extends AppCompatActivity {
```

Slika 137 Prikaz klase "SignUpActivity" kako nasleđuje klasu "AppCompatActivity"

Povezana je sa xml fajlom „activity\_sign\_up.xml“ (slika 138) i namenjana je za preuzimanje korisničkih podataka sa ulaza prilikom registracije korisnika i smeštanje istih u bazu podataka.

```
22 setContentView(R.layout.activity_sign_up);
```

Slika 138 Prikaz naziva layout-a koji koristi klasa "SignUpActivity"

U prvom delu ove klase se nalaze potrebne deklaracije, slika 139.

```
12 EditText editTextIme, editTextPrezime, editTextKorisnickoIme, editTextLozinka, editTextPotvrdaLozinke, editTextJMBG;
13 Button buttonRegistracija, buttonPrijava;
14
15 DatabaseHelper databaseHelper;
16
17 String ime = "", prezime = "", korisnickoIme = "", lozinka = "", potvrda = "", jmbg = "";
```

Slika 139 Prikaz deklaracija polja

U drugom delu ove klase se nalazi nadglasana(override) metoda „onCreate“, koja se poziva automatski prilikom prelaska iz aktivnosti „MainActivity“ ili „AdminActivity2“ u aktivnost „SignUpActivity“, slika 140.

```
20 protected void onCreate(Bundle savedInstanceState) {...
```

Slika 1140 Prikaz override metode "onCreate"

Unutar metode „onCreate“ je postavljen osluškivač na tastere „buttonRegistracija“ i „buttonPrijava“, gde će osluškivač u zavisnosti od događaja pozvati odgovarajuću nadglasanu(override) metodu „onClick“ koja će se izvršiti, slika 141.

```
33 buttonRegistracija.setOnClickListener(new View.OnClickListener() {
34     @Override
35     public void onClick(View v) {...}
47 });
48
49 buttonPrijava.setOnClickListener(new View.OnClickListener() {
50     @Override
51     public void onClick(View v) { finish(); }
54 });
```

Slika 141 Prikaz osluškivača na tastere

Osluškivač na „buttonRegistracija“ poziva nadglasanu(override) metodu „OnClick“ koja preuzima korisničke podatke iz editText polja i smešta ih u promenjive u vidu stringova, gde se zatim te promenljive prosleđuju kao parametri pozivu metode „addDataInKorisnici“, koja je namenjena za smeštanje podataka u bazu podataka(slika 142).

```

33 buttonRegistracija.setOnClickListener(new View.OnClickListener() {
34     @Override
35     public void onClick(View v) {
36         databaseHelper = new DatabaseHelper( context: SignUpActivity.this);
37
38         ime = editTextIme.getText().toString();
39         prezime = editTextPrezime.getText().toString();
40         jmbg = editTextJMBG.getText().toString();
41         korisnickoIme = editTextKorisnickoIme.getText().toString();
42         lozinka = editTextLozinka.getText().toString();
43         potvrda = editTextPotvrdaLozinke.getText().toString();
44
45         databaseHelper.addDataInKorisnici(ime, prezime, jmbg, korisnickoIme, lozinka, potvrda);
46     }
47 });

```

Slika 142 Prikaz definicije override metode „onClick“ koju poziva osluškivač na "addDataInKorisnici"

Osluškivač na „buttonLOGOUT“ završava aktivnost(slika 143).

```

49 buttonPrijava.setOnClickListener(new View.OnClickListener() {
50     @Override
51     public void onClick(View v) { finish(); }
54 });

```

Slika 1243 Prikaz definicije override metode „onClick“ koju poziva osluškivač na "buttonLOGOUT"

### 3.3.12 Klasa „Check“

Klasa „Check“ sadrži pet metoda, slika 144.

```

3 public class Check {
4     private static boolean isStringOnlySlova(String str) {...}
20
21     private static boolean isStringOnlyBrojevi(String str) {...}
37
38     private static boolean isStringOnlyBrojeviIslova(String str) {...}
54
55     @ public static String validateData1(String[] podatak) {...}
180
181     public static String validateData2(String ime, String prezime, String jmbg, String korisnickoIme, String lozinka, String potvrda) {...}
205 }

```

Slika 144 Prikaz svih metoda klase "Check"

Prve tri metode su namenjene za ispitivanje da li je String sastavljen samo od slova, samo od brojeva ili od slova i brojeva, a druge dve metode su namenjene za proveru korisničkih ulaznih podataka, gde one to rade pomoću pozivanja prve tri metode za ispitivanje Stringova.

### 3.3.13 Klasa „DatabaseHelper“

Klasa „DatabaseHelper“ je namenjena za stvaranje baze podataka, dodavanje podataka u bazu, ažuriranje podataka iz baze, kao i brisanje podataka iz baze.

Stvaranje baze podataka se dešava unutar tela konstruktora, slika 145.

```

58      public DatabaseHelper(Context context) {
59          super(context, DATABASE_NAME, factory: null, version: 1);
60          this.context = context;
61      }

```

Slika 145 Defenicija konstruktora klase DatabaseHelper

Stvaranje tabela baze podataka, se dešava unutar nadglasane(override) metode „onCreate“, slika 146.

```

63      @Override
64      public void onCreate(SQLiteDatabase db) {...}

```

Slika 146 Prikaz override metode "onCreate"

Nadogradnja(uprage) tabela baze podataka se dešava unutar nadglasane(override) metode „onUpgrade“, slika 147.

```

112     @Override
113     public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {...}

```

Slika 147 Prikaz override metode "onUpgrade"

Pored navedenih metoda, klasa „DatabaseHelper“ sadrži metode sa slike 148.

```

119     public void addDataInMaticnaKnjigaRodjenih(String [] podatak) {...}
173
174     public void addDataInKorisnici(String ime, String prezime, String jmbg, String korisnickoIme, String lozinka, String potvrda) {...}
199
200     public boolean validateUserInKorisnici(String korisnickoIme, String lozinka) {...}
221
222     public String getJMBGroditeljaFromKorisnici(String korisnickoIme) {...}
238
239     public Cursor readOneDataFromMaticnaKnjigaRodjenih(String jmbgDeteta, String jmbgRoditelja) {...}
252
253     public Cursor readDataFromMaticnaKnjigaRodjenih(String jmbgRoditelja) {...}
266
267     public Cursor readDataFromMaticnaKnjigaRodjenih() {...}
280
281     public Cursor readDataFromKorisnici() {...}
294
295     public void updateDataInMaticnaKnjigaRodjenih(String [] podatak) {...}
349
350     public void updateDataInKorisnici(String ime, String prezime, String korisnickoIme, String lozinka, String potvrda, String jmbg) {...}
375
376     public void deleteData(String jmbgDeteta) {...}
387
388     public void deleteDataFromKorisnici(String jmbgRoditelja) {...}
399
400     public Cursor readOneDataFromMaticnaKnjigaRodjenih(String jmbgDeteta) {...}
413
414     public Cursor readOneDataFromKorisnici(String jmbgRoditelja) {...}

```

Slika 148 Prikaz ostalih metoda

Metoda „addDataInMaticnaKnjigaRodjenih“ je namenjena za dodavanje primljenih podataka u tabelu koja se zove „maticnaKnjigaRodjenih“.

Metoda „addDataInKorisnici“ je namenjena za dodavanje primljenih podataka u tabelu koja se zove „korisnici“.

Metoda „validateUserInKorisnici“ je namenjena za proveru da li se primljeni podaci nalaze u tabelu koja se zove „korisnici“.

Metoda „getJMBGroditeljaFromKorisnici“ je namenja za vraćanje podatka iz tabele „korisnici“, u koliko primljen podatak postoji u toj tabeli.

Metoda „readOneDataFromMaticnaKnjigaRodjenih“ je namenja za vraćanje podataka iz tabele „maticnaKnjigaRodjenih“, u koliko primljeni podaci postoje u toj tabeli.

Metoda „readDataFromMaticnaKnjigaRodjenih“ je namenja za vraćanje podataka iz tabele „maticnaKnjigaRodjenih“, u koliko primljen podatak postoji u toj tabeli.

Metoda „readDataFromMaticnaKnjigaRodjenih“ je namenja za vraćanje svih podataka iz tabele „maticnaKnjigaRodjenih“.

Metoda „readDataFromKorisnici“ je namenja za vraćanje svih podataka iz tabele „korisnici“.

Metoda „updateDataInMaticnaKnjigaRodjenih“ je namenja za ažuriranje podataka tabele „maticnaKnjigaRodjenih“, u koliko jedan od primljenih podataka postoji u toj tabeli.

Metoda „updateDataInKorisnici“ je namenja za ažuriranje podataka tabele „korisnici“, u koliko jedan od primljenih podataka postoji u toj tabeli.

Metoda „deleteData“ je namenja za brisanje podatka iz tabele „maticnaKnjigaRodjenih“, u koliko primljen podatak postoji u toj tabeli.

Metoda „deleteDataFromKorisnici“ je namenja za brisanje podatka iz tabele „korisnici“, u koliko primljen podatak postoji u toj tabeli.

Metoda „readOneDataFromMaticnaKnjigaRodjenih“ je namenja za vraćanje podataka iz tabele „maticnaKnjigaRodjenih“, u koliko primljen podatak postoji u toj tabeli.

Metoda „readOneDataFromKorisnici“ je namenja za vraćanje podataka iz tabele „korisnici“, u koliko primljen podatak postoji u toj tabeli.

#### **4 Literatura**

1. programiranje mobilnih aplikacija, četvrta godina, prvi semestar, računarska tehnika i softversko inženjerstvo, moodle portal: <http://moodle.fink.rs>, 27.01.2021 (18:00)