Društvena mreža bazirana na anketama

Pleša, Matija

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

Sveučilišni studij

DRUŠTVENA MREŽA BAZIRANA NA ANKETAMA

Završni rad

Matija Pleša

Osijek, 2021.

SADRŽAJ

1. UVOD

Društvena povezanost ljudi je vrlo bitna stavka života svake osobe. Do lakšega povezivanja s ostatkom svijeta, te razmjene informacija i stavova doveo je napredak tehnologije. Tome su najviše doprinijeli internet, te društvene mreže. Društvena mreža je vrsta internetske usluge, koja se najčešće javlja u obliku platforme, prozora ili web stranice. To je internetski prozor koji služi za međusobno povezivanje korisnika [1]. Danas postoje stotine ovakvih servisa, a među najpoznatijima su: Facebook, Twitter, i Instagram.

Izrada mobilne aplikacije potaknuta je time da su društvene mreže vrlo bitne u povezivanju ljudi i njihovoj razmjeni stavova. Aplikacija se temelji na tome da korisnici postavljaju pitanja o vlastitim stavovima, te im drugi korisnici odgovaraju na ankete. Ideja aplikacije je da unutar iste korisnici mogu pregledavati sve ankete koje su ikada bile postavljenje, filtrirati pregled, te brisati ankete koje su postavili ako nitko do tada nije odgovorio na iste. Ankete su unesene u vanjsku bazu podataka kako bi bili sigurni da će ostati pohranjene na pravilan način i da će biti dostupne.

Rad je podijeljen na tri dijela: usporedba s već postojećim rješenjima, razvojno okruženje i programsko rješenje i primjena aplikacije. U prvom dijelu aplikacija se uspoređuje s već sličnim rješenjima. U drugom dijelu opisano je razvojno okruženje koje je korišteno za izradu same aplikacije, a u zadnjem djelu opisano je programsko rješenje aplikacije.

1.1. Zadatak završnog rada

Cilj je izraditi Android aplikaciju unutar koje će se korisnik moći registrirati, te kasnije prijaviti u aplikaciju. Samim time imat će mogućnost postavljanja anketa, brisanja istih, te pregledavanja anketa drugih korisnika i odgovaranja na ankete. Svaki korisnik imat će mogućnost uređivati tj. brisati vlastite ankete. Ankete, odgovori na iste i podaci potrebni za prijavu u aplikaciju bit će pohranjeni na vanjsku bazu podataka pomoću Firebase platforme.

2. USPOREDBA S VEĆ POSTOJEĆIM RJEŠENJIMA

Kako razvoj mobilnih aplikacija, te interneta napreduje, vrlo je teško osmisliti neku novu ideju i istu realizirati u vidu aplikacije. Na tržištu postoji već vrlo veliki broj sličnih a ponekad i gotovo istih aplikacija koje rješavaju neku problematiku, ali to nikada ne treba predstavljati zapreku prilikom realiziranja ideje. Nisu uvijek već postojeća rješenja uvijek najbolja i uvijek postoji prostora za napredak. Kao što je navedeno u uvodu cilj ovoga završnoga rada je izraditi aplikaciju koja će u vidu društvene mreže bazirane na anketama omogućiti korisnicima da razmjenjuju svoje stavove i iskustva u vidu anketa te odgovora na iste. Na trgovini play postoji nekolicina aplikacija koje se bave sličnom problematikom, a najpoznatije su: Pollie: Create Polls i Poll Everywhere. Navedene aplikacije će se usporediti s aplikacijom koja se izrađuje u ovom završnom radu u sljedećim poglavljima.

2.1. Pollie: Create Polls

Pollie: Create Polls je mobilna aplikacija koja je dostupna za preuzimanje na trgovina play za Android uređaje. Ova aplikacija pomaže prilikom izrade novih anketa. Ne zahtijeva registraciju i besplatna je za korištenje. Jedna od prednosti je što ima već gotove predloške ankete koje korisnik treba samo popuniti, te odabrati postavke ankete koje želi (npr. mogu li korisnici više puta glasati itd.). Glavni nedostatak aplikacije je što korisnici unutar aplikacije ne mogu komentirati ankete, već je ankete potrebno pomoću linka slati korisnicima [7]. Također nedostatak je što korisnici ne mogu uređivati ankete, već ako žele nešto promijeniti moraju ih ponovo izraditi.

2.2. Poll Everywhere

Poll Everywhere je također mobilna aplikacija koje je dostupna za preuzimanje na trgovina play za Android uređaje. Aplikacija zahtjeva registraciju i nije besplatna za korištenje. Pomoću aplikacije je moguće izrađivati ankete, također iz već gotovih predložaka. Prednost aplikacije je što je ankete moguće uređivati. Također je prednost što je moguće uživo diskutirati s drugim korisnicima o stavovima oko anketa i odgovora [8]. A glavni nedostatak je što je poprilično skupa za korištenje.

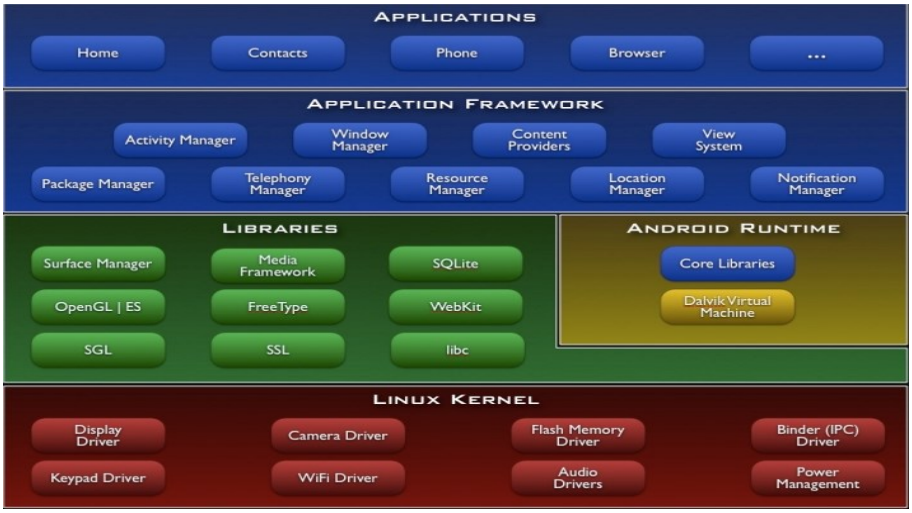
3. RAZVOJNO OKRUŽENJE

U ovom poglavlju će biti opisane korištene tehnologije, tj razvojno okruženje korišteno za razvoj same aplikacije završnog rada. Korištene tehnologije za izradu završnog rada su: Android Studio, programski jezik Java, opisni jezik XML, poslužitelj Firebase, te operacijski sustav Android.

3.1. Operacijski sustav Android

Operacijski sustav Android je otvoreni sustav za mobilne uređaje američke tvrtke Google Inc. Temeljen je na jezgri Linux. Android je prvo razvila tvrtka Android Inc. koju je Google naknadno kupio. Android je najprodavaniji operacijski sustav od 2011. godine za mobilne uređaje, te ima više od 2 milijarde mjesečno aktivnih korisnika. Trgovina Play ima preko 3.5 milijuna aplikacija. Android je zasnovan na Linux 2.6 jezgri je kod napisan u C/C++ programskom jeziku. Većina aplikacija za Android ipak je napisana u programskom jeziku Java [2]. Android sustav prema arhitekturi možemo podijeliti na tri razine kako je prikazano na slici 3.1 [9]:

- Linux jezgra
- C/C++ knjižnice
- Android Runtime



Slika 3.1 Razine Android arhitekture

3.2. Android Studio

Android Studio je utemeljen na JetBrains IntelliJ IDEA softveru, te predstavlja službeno okruženje za Android platformu. Objavljen je 2013. godine, dok je prva stabilna verzija izašla 2014. godine. Android Studio podržava Linux, MacOS i Windows operacijske sustave. Prednost Android Studia je što ima ugrađen emulator za testiranje aplikacija, te ima pristup vanjskim poslužiteljima poput GitHub-a i Google Cloud platformi. Također ima alat za ispravljanje pogrešaka i alat za praćenje performansi i resursa [3]. Android Studio razvojno okruženje nudi mnogo mogućnosti te je ovaj završni rad kompletno napravljen u ovom razvojnom okruženju.

Slika 3.2 Android Studio logotip

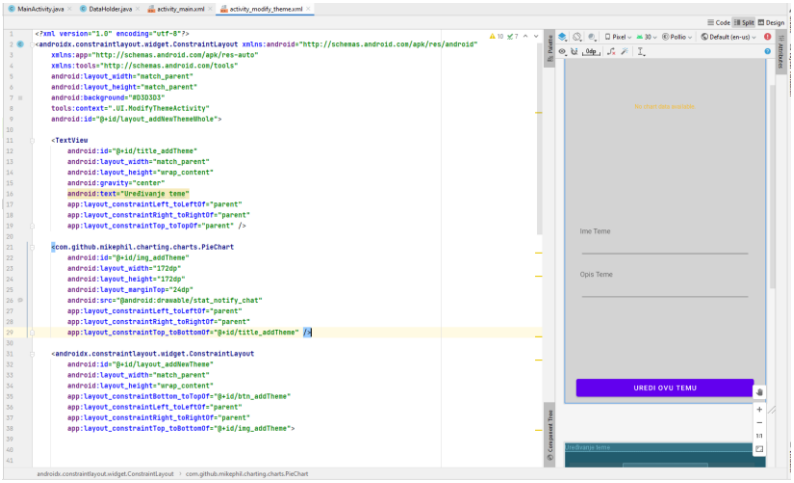
3.3. Programski jezik Java

Programski jezik Java objavljen je 1995. godine. Razvili su ga Patrick Naughton, James Gosling iz tvrtke Sun Microsystems. Danas je najkorišteniji programski jezik. Prema sintaksi je vrlo sličan programskom jeziku C++ i također je objektno orijentiran. Ne podržava višestruko nasljeđivanje, i polimorfizam (preopterećivanje operatora) te globalne i statičke varijable. U odnosu na C++ jezik, Java pruža bolju sigurnost i pouzdanost zbog Java Virtual Machine-a (JVM). Java Virtual Machine predstavlja zatvoreno okruženje unutar kojega se programi razvijaju brže i s manje pogrešaka [5].

Slika 3.3 Java logotip

3.4. Opisni jezik XML

XML (eXtensible Markup Language) predstavlja jezik za označavanje podataka. Format oznaka XML jezika je vrlo sličan formatu oznaka HTML (HyperText Markup Language) jezika. XML je standardiziran jezik za čiju se standardizaciju brine W3C (World Wide Web Consortium). XML je zapravo jezik čiji je princip realizacije jednostavan i svaka oznaka koja ga opisuje ima poznato i lako shvatljivo značenje [4]. Unutar Android Studio-a XML jezik se koristi za opis izgleda korisničkog sučelja te za predefinirano ponašanje elemenata.



Slika 3.4 Primjer XML koda i izgled korisničkog sučelja

3.5. Poslužitelj Firebase

Firebase je platforma razvijena od strane Google-a. Platforma nudi veliki broj mogućnosti, odnosno servisa kao što su: Google Analytics, Firebase Cloud Messaging, Cloud Firestore, Firebase Hosting, ML kit te za aplikaciju ovog završnog rada najvažniji servis Firebase Realtime Database. Firebase Realtime Database predstavlja udaljenu online bazu podataka u koju je moguće u stvarnome vremenu izmjenjivati podatke [6].

Slika 3.5 Firebase logotip

4. PROGRAMSKO RJEŠENJE I PRIMJENA APLIKACIJE

Razvoj same aplikacije kao i koraci u kreiranju ove aplikacije uz primjere i objašnjena bit će prikazani u ovom poglavlju. Prije samog kreiranja aplikacije potrebno je preuzeti Android Studio te se dobro upoznati s principima njegova rada. Isto tako vrlo je bitno poznavanje samog programskog jezika Java i opisnog jezika XML. Uz to potrebno je napraviti korisnički račun na Firebase platformi kako bi mogli koristi uslugu baze podataka u stvarnome vremenu (Firebase Realtime Database). Vrlo važan koncept za prikaz anketa je i RecyclerView koji omogućuje prikaz anketa, i komentara na ankete koji se povlače iz baze podataka. Također vrlo važni su i adapteri za prilagodbu prenesenih podataka između aplikacije i baze podataka.

4.1. Android Manifest

Obavezna stavka svake Android aplikacije je AndroidManifest.xml datoteka. Datoteka je pisana opisnim XML jezikom te su unutar nje sadržane bitne informacije o aplikaciji. Unutar ove datoteke dodajemo dozvole aplikaciji. Za izradu ove aplikacije bilo je potrebno dodati dozvolu za pristup internetu.

AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.example.pollio">
4
5      <uses-permission android:name="android.permission.INTERNET" />
6
```

Slika 4.1 Prikaz dodavanja dozvole za pristup internetu

```
android:icon="@drawable/like_icon"
android:label="@string/app_name"
```

Slika 4.2 Prikaz deklaracije imena i ikone aplikacije

```
    <activity android:name=".UI.ModifyThemeActivity"></activity>
    <activity android:name=".UI.NewThemeActivity" />
    <activity android:name=".UI.ThemeActivity" />
    <activity android:name=".UI.MainActivity"></activity>
    <activity android:name=".UI.StartActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```

Slika 4.3 Prikaz deklaracija komponenti

4.2. Knjižnice

Prije početka izrade same aplikacije potrebno je dodati neke od ključnih implementacija knjižnica kao što su: Firebase Realtime Database protokoli te dodatne knjižnice za ljepši izgled korisničkog sučelja.

```
dependencies {

    implementation 'androidx.appcompat:appcompat:1.3.1'
    implementation 'com.google.android.material:material:1.4.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.0'
    implementation 'org.jetbrains:annotations:21.0.1'
    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'

    implementation 'com.google.firebase:firebase-database:20.0.1'
    implementation 'com.google.android.material:material:1.5.0-alpha01'
    implementation 'com.github.PhilJay:MPAndroidChart:v3.1.0'
}
```

Slika 4.4 Implementacija vanjskih knjižnica

4.3. Početak korištenja aplikacije

Kako bi korisnik mogao pravilno koristiti aplikaciju prvo što je potrebno je da se registrira. Prilikom pokretanja aplikacije prva pokrenuta aktivnost je StartActivity. Unutar te aktivnosti tj. njezinog korisničkog sučelja korisnik ima ponuđena dva polja za unus korisničkog imena i korisničke lozinke. Pomoću metode setOnClickListener čeka se korisnikov klik na tipku Prijava. Taj korisnikov klik pokreće provjeru uvjeta uspješne registracije. Provjera uvjeta uspješne registracije se vrši tako da se provjerava dali je korisnik unio i korisničko ime i korisničku lozinku prije klika na tipku, sadrži li ime 3 ili više znaka i sadrži li lozinka 6 ili više znakova. Ako neki od uvjeta nije zadovoljen ispisuje se poruka koja opisuje grešku. Ako je korisnik unio sve kako uvjeti nalažu ime i lozinka se pohranjuju u novi objekt klase User s parametrima nickname za ime, password za lozinku i search_name za ime. Nakon toga se automatski izvodi pokušaj prijave korisnika. Za to je zaslužna metoda loginUser.

```
    btn_start.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            hideKeyboard( activity: StartActivity.this);

            if (TextUtils.isEmpty(txt_nickname.getText().toString()) || TextUtils.isEmpty(txt_password.getText().toString())) {
                infoSnackBar( info: "Ime i zaporka su obavezni!", btn_info: "U redu!");
            } else if (txt_nickname.getText().toString().length() < 3) {
                infoSnackBar( info: "Ime mora biti 3 ili više znakova!", btn_info: "U redu!");
            } else if (txt_password.getText().toString().length() < 6) {
                infoSnackBar( info: "Zaporka mora biti 6 ili više znakova!", btn_info: "U redu!");
            } else {
                loginUser(new User(txt_nickname.getText().toString(), txt_password.getText().toString(), txt_nickname.getText().toString().toLowerCase()));
            }
        }
    });
}
```

Slika 4.5 Provjera uvjeta uspješne registracije korisnika

```
public class User {
    private String nickname;
    private String password;
    private String search_name;

    public User(String nickname, String password, String search_name) {
        this.nickname = nickname;
        this.password = password;
        this.search_name = search_name;
    }

    public User() {
    }

    public String getNickname() { return nickname; }

    public void setNickname(String nickname) { this.nickname = nickname; }

    public String getPassword() { return password; }

    public void setPassword(String password) { this.password = password; }

    public String getSearch_name() { return search_name; }

    public void setSearch_name(String search_name) { this.search_name = search_name; }
}
```

Slika 4.6 Prikaz klase User s pripadajućim metodama

4.4. Registracija korisnika

Ako je korisnik prvi puta pristupio aplikaciji pokušaj prijave neće proći. Nakon toga korisniku će iskočiti poruka „Oooh, novi korisnik? Želite li se registrirati?“ Ako se korisnik odluči registrirati bit će registriran u bazu podataka s podacima koje je unio na početku. Za to je zaslužna metoda registerUser. U ovom dijelu aplikacija prvi puta dolazi u kontakt s bazom podataka na Firebase platformi. Za prijenos podataka se koriste predefinirani protokoli od strane Firebase servisa koji smo uključili u već spomenutim knjižnicama. Za pomoć oko rada s bazom podataka kreirana je klasa DataHolder. Klasa DataHolder sadrži privatni String naziva dbUrl koji sadrži URL (Uniform Resource Locator) adresu vanjske baze podataka.

```
private void registerUser(final User newUser) {
    Snackbar.make(layout_start, text: "Ooooh, novi korisnik? Želite li se registrirati?", Snackbar.LENGTH_LONG)
            .setAction( text: "Yes, please!", new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    DatabaseReference ref = mDatabase.child("Users");
                    ref.child(newUser.getSearch_name()).setValue(newUser);
                    startThisActivity(new Intent( packageContext: StartActivity.this, MainActivity.class), newUser);
                }
            })
            .setActionTextColor(getResources().getColor(R.color.teal_700))
            .show();
}
```

Slika 4.7 Prikaz metode registerUser

```
public class DataHolder {
    private String data;
    private final String dbUrl = "https://pollio-153d7-default-rtdb.europe-west1.firebasedatabase.app/";

    public String getDbUrl() { return this.dbUrl; }

    public String getData() { return data; }

    public void setData(String data) { this.data = data; }

    private static final DataHolder holder = new DataHolder();

    public static DataHolder getInstance() { return holder; }
}
```

Slika 4.8 Prikaz klase DataHolder s pripadajućim metodama

4.5. Prijava korisnika

Nakon uspješne registracije korisnik se automatski prijavljuje u aplikaciju i može krenuti s pregledom i objavljivanjem anketa. Ako je korisnik već bio registriran ranije, on tada unosi korisničko ime i lozinku te se putem tipke Prijava prijavljuje u aplikaciju. Uspješno prijavljivanje

u aplikaciju predstavlja korištenje Firebase protokola za provjeru ispravnosti unesenih podataka kako bi se provjerilo sadrži li baza podataka postojećih korisnika točno tog korisnika s istim korisničkim imenom i lozinkom te nakon toga šalje potvrdu o uspješnosti u pronalasku takvog korisnika. Neuspješnu prijavu predstavlja to da se korisnik ne nalazi u bazi podataka te mu se ne dozvoljava pristup aplikaciji uz poruku o neuspjeloj prijavi u aplikaciju.

```
private void loginUser(final User newUser) {
    DatabaseReference ref = mDatabase.child("Users");
    ref.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            boolean foundUser = false;
            for (DataSnapshot data : dataSnapshot.getChildren()) {
                User mUser = data.getValue(User.class);
                if (mUser.getSearch_name().equals(newUser.getSearch_name()) && !mUser.getPassword().equals(newUser.getPassword())) {
                    infoSnackBar( info: "Zaporka je netočna!",  btn_info: "U redu!");
                    foundUser = true;
                } else if (mUser.getSearch_name().equals(newUser.getSearch_name()) && mUser.getPassword().equals(newUser.getPassword())) {
                    startThisActivity(new Intent( packageContext: StartActivity.this, MainActivity.class), mUser);
                    foundUser = true;
                }
            }

            if (!foundUser) {
                registerUser(newUser);
            }
        }

        @Override
        public void onCancelled(@NonNull DatabaseError error) {
        }
    });
}
```

Slika 4.9 Prikaz metode loginUser

Prilikom klika na tipku prijava i uspješne prijave u aplikaciju. Pokreće se nova aktivnost MainActivity unutar koje je moguće vršiti pregled već objavljenih anketa, komentirati i ocjenjivati ih te postavljati svoje ankete, uređivati ih i brisati ih.

4.6. Glavni izbornik

Uspješnom registracijom i prijavom u aplikaciju, te pohranom podataka o korisniku u bazu podataka pokreće se glavna aktivnost tj. glavni izbornik. Glavni izbornik predstavlja najsloženiji dio programskog rješenja, ali vrlo jednostavnog i intuitivnog sučelja s vrlo jednostavnim i jasno vidljivim funkcionalnostima koje su pružene korisniku na samovoljno korištenje. Gotovo sve funkcionalnosti ove aplikacije se obavljaju interakcijom iz glavnog izbornika gdje su trenutno omogućene funkcionalnosti poput: Dodavanja nove ankete, Pregleda anketi za raspravu, Odabir

sortiranja anketa (Najnovije, Najstarijem Najviše glasova, Moje ankete prvo), Odabira zaslona za uređivanje i glasanja na ankete.

Ankete unutar glavnog izbornika su prikazane pomoću RecyclerView-a. RecyclerView omogućava fleksibilan prikaz više identičnih modela prikaza, kao što su u ovome slučaju ankete. Ankete su zapravo „djeca“ RecyclerView-a koja posjeduju svoj indeks ili poziciju na kojoj se nalaze. Anketama je najlakše upravljati pomoću liste za prikaz koristeći indekse anketa.

Glavna zadaća glavnog izbornika je zapravo prikaz anketa, te interakcija s njima. Za korištenje samog RecyclerView-a unutar glavnog izbornika kreiran je ThemeAdapter s pripadajućim ViewHolder-om. RecyclerView adapter tj. u ovom slučaju nazvan ThemeAdapter je potreban za povezivanje samih podataka ankete koje želimo prikazati i samog RecyclerView-a.

```
public class ThemeAdapter extends RecyclerView.Adapter<ThemeAdapter.ViewHolder> {
    private final List<Theme> mThemes;
    private final FirebaseDatabase tasksDBRef;
    private Context context;

    public ThemeAdapter(List<Theme> mThemes, FirebaseDatabase mRef) {
        this.mThemes = mThemes;
        this.tasksDBRef = mRef;
    }

    @Override
    public void onAttachedToRecyclerView(RecyclerView recyclerView) {
        super.onAttachedToRecyclerView(recyclerView);
        context = recyclerView.getContext();
    }

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {...}

    @Override
    public void onBindViewHolder(@NonNull ViewHolder holder, int position) {...}

    private void modifyTheme(Theme theme) {...}

    @Override
    public int getItemCount() { return mThemes.size(); }

    public class ViewHolder extends RecyclerView.ViewHolder {...}
}
```

Slika 4.10 Prikaz ThemeAdapter-a

Klasa ViewHolder je potrebna za povezivanje layout-a prikaza pojedine ankete sa RecylerView adapterom.

```
public class ViewHolder extends RecyclerView.ViewHolder {
    private final TextView themeTitle;
    private final TextView themeDesc;
    private final TextView themeLikes;
    private final TextView themeDislikes;
    private final PieChart themeIcon;
    private final ConstraintLayout themeView;
    private final TextView tv_AuthorMark;

    public ViewHolder(@NonNull View itemView) {
        super(itemView);

        themeTitle = itemView.findViewById(R.id.item_title);
        tv_AuthorMark = itemView.findViewById(R.id.tv_AuthorMark);
        themeDesc = itemView.findViewById(R.id.item_description);
        themeLikes = itemView.findViewById(R.id.item_likes);
        themeDislikes = itemView.findViewById(R.id.item_dislikes);
        themeIcon = itemView.findViewById(R.id.item_image);
        themeView = itemView.findViewById(R.id.item_layout);
    }
}
```

Slika 4.11 Prikaz ViewHolder-a

Za rad s anketama unutar aplikacije kreirana je klasa Theme, koja sadrži podatke o samoj anketi od imena, opisa, broja glasova za i protiv te imena autora ankete.

```
public class Theme implements Parcelable {
    private String themeName;
    private String themeDescription;
    private int themeLikes;
    private int themeDislikes;
    private String author;

    public Theme() {
    }

    public Theme(String author, String themeName, String themeDescription, int themeLikes, int themeDislikes) {
        this.themeName = themeName;
        this.themeDescription = themeDescription;
        this.themeLikes = themeLikes;
        this.themeDislikes = themeDislikes;
        this.author = author;
    }

    public String getAuthor() { return author; }

    public void setAuthor(String author) { this.author = author; }

    public int getThemeDislikes() { return themeDislikes; }

    public int getThemeLikes() { return themeLikes; }

    public String getThemeDescription() { return themeDescription; }

    public String getThemeName() { return themeName; }

    public void setThemeDescription(String themeDescription) {
        this.themeDescription = themeDescription;
    }

    public void setThemeDislikes(int themeDislikes) { this.themeDislikes = themeDislikes; }

    public void setThemeLikes(int themeLikes) { this.themeLikes = themeLikes; }

    public void setThemeName(String themeName) { this.themeName = themeName; }
```

Slika 4.12 Prikaz klase Theme

Nakon prikaza anketa implementirano je sortiranje pretraživanja. To se vrši pomoću metoda: sortByNewst, sortByOldest, sortByMostVoted i sortByAuthorsFirst. Metoda sortByNewst implementira sortiranje prikaza anketa tako da prikazuje prvo najnovije ankete. Metoda radi na principu da u novu listu newOrderThemes dodaje ankete redom iz baze podataka te im obrne poredak u listi i tako ih prikazuje unutar RecyclerView-a.

```
private void sortByNewest() {
    MenuState = 1;
    List<Theme> newOrderThemes = new ArrayList<>();
    newOrderThemes.addAll(themes);
    Collections.reverse(newOrderThemes);

    themeAdapter = new ThemeAdapter(newOrderThemes, database);
    rv_themes.setAdapter(themeAdapter);
}
```

Slika 4.13 Prikaz metode sortByNewest

Metoda sortByOldest implementira sortiranje prikaza anketa tako da prikazuje prvo najstarije ankete. Metoda sortByOldest radi na principu da redom iz baze podataka kako su bile upisane prikazuje ankete unutar RecylerView-a.

```
private void sortByOldest() {
    themeAdapter = new ThemeAdapter(themes, database);
    rv_themes.setAdapter(themeAdapter);
}
```

Slika 4.14 Prikaz metode sortByOldest

Metoda sortByMostVoted implementira sortiranje prikaza anketa tako da prikazuje prvo ankete s najviše glasova. Metoda sortByMostVoted radi na principu da uspoređuje broj glasova anketa te prvo ispisuje one koje ima veći broj glasova.

```
private void sortByMostVoted() {
    List<Theme> newOrderThemes = new ArrayList<>();
    newOrderThemes.addAll(themes);

    Collections.sort(newOrderThemes, new Comparator<Theme>() {
        @Override
        public int compare(Theme lhs, Theme rhs) {
            return Integer.valueOf(rhs.getThemeLikes() + rhs.getThemeDislikes()).compareTo(lhs.getThemeLikes() + lhs.getThemeDislikes());
        }
    });

    themeAdapter = new ThemeAdapter(newOrderThemes, database);
    rv_themes.setAdapter(themeAdapter);
}
```

Slika 4.15 Prikaz metode sortByMostVoted

Metoda sortByAuthorsFirst implementira sortiranje prikaza anketa tako da prikazuje prvo ankete koje je objavio trenutno prijavljeni autor. Metoda radi na principu da provjerava dali je korisničko ime prijavljenog autora isto s korisničkim imenom autora anketa iz baze podataka te prvo ispisuje ankete koje je objavio trenutno prijavljeni autor. Također u aplikaciji je implementirano da korisnik kod vlastitih tema u gornjem desnom kutu vidi slovo A te je tako siguran da su to njegove ankete kako bi ih u daljnjem radu mogao uređivati ili brisati.

```
private void sortByAuthorsFirst() {
    List<Theme> newOrderThemes = new ArrayList<>();

    List<Theme> tempThemes = new ArrayList<>();
    for (int i = 0; i < themes.size(); i++) {
        if (themes.get(i).getAuthor().equals(DataHolder.getInstance().getData())) {
            newOrderThemes.add(themes.get(i));
        } else {
            tempThemes.add(themes.get(i));
        }
    }
    newOrderThemes.addAll(tempThemes);

    themeAdapter = new ThemeAdapter(newOrderThemes, database);
    rv_themes.setAdapter(themeAdapter);
}
```

Slika 4.16 Prikaz metode sortByAuthorsFirst

Za odabir metode sortiranja pregleda anketa implementiran je padajući izbornik pomoću metode onCreateOptionsMenu te se unutar padajućeg izbornika mogu odabrati prethodno navedene metode sortiranja prikaza.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.main_menu, menu);

    for (int i = 0; i < menu.size(); i++) {
        menu.getItem(i).setEnabled(i != MenuState);
    }

    return true;
}
```

Slika 4.17 Prikaz metode onCreateOptionsMenu

Za mogućnost odabira metoda zaslužna je metoda onOptionsItemsSelected.

```
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    findViewById(R.id.menu_sort_icon).setEnabled(false);

    if (item.getItemId() == R.id.menu_sort_newest) {
        sortByNewest();
    } else if (item.getItemId() == R.id.menu_sort_oldest) {
        sortByOldest();
        MenuState = 2;
    } else if (item.getItemId() == R.id.menu_sort_mostVotes) {
        sortByMostVoted();
        MenuState = 3;
    } else if (item.getItemId() == R.id.menu_sort_author) {
        sortByAuthorsFirst();
        MenuState = 4;
    } else {
        return super.onOptionsItemSelected(item);
    }

    invalidateOptionsMenu();
    return true;
}
```

Slika 4.18 Prikaz metode onOptionsItemSelected

Klikom na neku od anketa otvara se nova aktivnost ThemeActivity a klikom na tipku „DODAJ NOVU TEMU“ otvara se nova aktivnost NewThemeActivity.

4.7. Dodavanje nove ankete

Dodavanje nove ankete vrši se u aktivnosti NewThemeActivity. Unutar ove aktivnosti postoji mogućnost upisa imena ankete te njenog opisa, a pritiskom na tipku „DODAJ NOVU TEMU“, anketa se dodaje u bazu podataka i prikazuje se na glavnom izborniku.

Prilikom klika na ranije spomenutu tipku vrši se provjera jesu li uneseni i ime i opis te dali je ime kraće od 35 znakova i opis od 100 znakova. Programsko rješenje za to je prikazno na slici ispod.

```
        @Override
        public void onClick(View v) {
            String title = et_addThemeTitle.getText().toString().trim();
            String description = et_addThemeDescription.getText().toString().trim();

            if (TextUtils.isEmpty(title) || TextUtils.isEmpty(description)) {
                infoSnackBar( info: "Oba polja su obavezna!", btn_info: "Gotcha!");
            } else if (title.length() > 35) {
                infoSnackBar( info: "Naslov teme ne smije biti duži od 35 znakova!", btn_info: "U redu!");
            } else if (description.length() > 100) {
                infoSnackBar( info: "Opis teme ne smije biti duži od 100 znakova!", btn_info: "U redu!");
            } else {
                addNewThemeToDB(title, description);
            }
        }
    });
}
```

Slika 4.19 Prikaz programskog rješenja za provjeru uvjeta

Za dodavanje ankete u bazu podataka pomoću preddefiniranih Firebase protokola kreirana je metoda addThemeToDB. Unutar metode se prvo provjerava postoji li već kreirana anketa s istim imenom. Ako ne postoji anketa se dodaje u bazu podataka i automatski prikazuje u glavnom izborniku.

```
private void addNewThemeToDB(String title, String description) {
    myRef.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            boolean titleExist = false;

            for (DataSnapshot snapshot : dataSnapshot.getChildren()) {
                Theme theme = snapshot.getValue(Theme.class);
                if (theme.getThemeName().equals(title)) {
                    infoSnackBar( info: "Identično pitanje već postoji, izaberite drugo.",  btn_info: "U redu!");
                    titleExist = true;
                }
            }

            if (!titleExist) {
                Theme newTheme = new Theme(DataHolder.getInstance().getData(), title, description,  themeLikes: 0,  themeDislikes: 0);
                myRef.child(title).setValue(newTheme);
                myRef.child(title).child("createdAt").setValue(ServerValue.TIMESTAMP);

                finish();
            }
        }
```

Slika 4.20 Prikaz metode addNewThemetoDB

4.8. Komentiranje i glasanje ankete

Komentiranje i glasanje ankete omogućeno je pomoću aktivnosti ThemeActivity. Unutar ove aktivnosti komentari i glasovi se prikazuje pomoću već objašnjenog principa s RecyclerView-om te pomoću njegovog adaptera koji ima naziv ThemeCommentAdapter i pripadajućeg ViewHolder-a. Pomoću tipki sa strelicom gore ili dolje se glasa za ili protiv te se glas automatski sprema u bazu podataka i prikazuje unutar RecyclerView-a. Pritiskom na tipku „GLASAJ“ i držanjem te tipke otvara se nova aktivnost GradeDialog unutar koje je moguće uz glas za ili protiv ostaviti i komentar na temu koji će se prikazati u izborniku s glasovima i komentarima. Metoda pushNewDataToDB kreirana je s ciljem da postavi komentar i glas autora u bazu podataka kako bi se on mogao pravilno ispisati unutar izbornika i kako bi ostao pohranjen.

```
private void pushNewDataToDB(Theme theme, FirebaseDatabase mDatabase, Dialog dialog) {
    EditText text_dialog = dialog.findViewById(R.id.text_dialog);

    DatabaseReference myRef = mDatabase.getReference( path: "Themes" + "/" + theme.getThemeName());
    myRef.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull @NotNull DataSnapshot snapshot) {
            if (isChosen == 1) {
                int grade = snapshot.child("themeLikes").getValue(Integer.class);
                snapshot.child("themeLikes").getRef().setValue(grade + 1);
            } else {
                int grade = snapshot.child("themeDislikes").getValue(Integer.class);
                snapshot.child("themeDislikes").getRef().setValue(grade + 1);
            }
        }

        @Override
        public void onCancelled(@NonNull @NotNull DatabaseError error) {

        }
    });

    DatabaseReference myRef2 = mDatabase.getReference( path: "ThemeComments" + "/" + theme.getThemeName());
    String key = myRef2.push().getKey();

    if (!TextUtils.isEmpty(text_dialog.getText().toString().trim())) {
        ThemeComment newComment = new ThemeComment(key, DataHolder.getInstance().getData(), text_dialog.getText().toString().trim(), isChosen,  likes: 0,  dislikes: 0);
        myRef2.child(key).setValue(newComment);
    } else {
        ThemeComment newComment = new ThemeComment(key, DataHolder.getInstance().getData(), isChosen,  likes: 0,  dislikes: 0);
        myRef2.child(key).setValue(newComment);
    }
}
```

Slika 4.21 Prikaz metode pushNewDataToDB

4.9. Uređivanje i brisanje ankete

Uređivanje i brisanje ankete moguće je samo ako je korisnik autor ankete koju želi izbrisati ili urediti. Kako je već objašnjeno autoru ankete vidljivo je veliko slovo A u gornjem desnom kutu pored ankete. Pritiskom na anketu i dugim držanjem otvara se izbornik unutar kojega korisnik može odabrati „UREDI“, „PREKID“ ili „OBRIŠI“. Pritiskom na obriši ankete se briše. Pritiskom na tipku uredi otvara se nova aktivnost ModifyThemeActivity.

4.10. Uređivanje ankete

Uređivanje ankete vrši se u aktivnosti ModifyThemeActivity. Unutar aktivnosti implementirana je mogućnost promjene imena i opisa anketa. Ako korisnik odluči promijeniti ime ili opis ankete u bazi podataka se mijenja anketa sa „starim“ imenom i opis te se upisuju novo ime i opis. Također brišu se i glasovi te komentari koje su ostali korisnici ostavili na „staru“ anketu. Za to je implementirana metoda pushChangesToDB.

```
private void pushChangesToDB(String title, String description) {
    boolean isChanges = false;

    if (TextUtils.isEmpty(title) || TextUtils.isEmpty(description)) {
        infoSnackBar( info: "Oba polja su obavezna!",  btn_info: "Gotcha!");
    } else if (title.length() > 35) {
        infoSnackBar( info: "Naslov teme ne smije biti duži od 35 znakova!",  btn_info: "U redu!");
    } else if (description.length() > 100) {
        infoSnackBar( info: "Opis teme ne smije biti duži od 100 znakova!",  btn_info: "U redu!");
    } else {
        if (!currentTheme.getThemeName().equals(title)) {
            currentTheme = new Theme(DataHolder.getInstance().getData(), title, description,  themeLikes: 0,  themeDislikes: 0);
            myRef.removeValue();
            DatabaseReference myRefParent = myRef2.getParent().getRef();
            myRef2.removeValue();
            myRefParent.child(currentTheme.getThemeName()).setValue(currentTheme);
            myRefParent.child(currentTheme.getThemeName()).child("createdAt").setValue(ServerValue.TIMESTAMP);
            isChanges = true;
        } else if (!currentTheme.getThemeDescription().equals(description)) {
            currentTheme.setThemeDescription(description);
            myRef2.child("themeDescription").setValue(description);
            isChanges = true;
        }

        if (!isChanges) {
            Snackbar.make(layout_addNewThemeWhole,  text: "Nikakve promjene nisu odrađene.", Snackbar.LENGTH_LONG)
                    .setAction( text: "Natrag", new View.OnClickListener() {
                        @Override
                        public void onClick(View v) { finish(); }
                    })
                    .setActionTextColor(getResources().getColor(R.color.teal_700))
                    .show();
        } else {
            initializeDataBase();
            infoSnackBar( info: "Promjene su objavljene!",  btn_info: "Hvala!");
        }
    }
```

Slika 4.22 Prikaz metode pushChangesToDB

Također za lakši prikaz glasova unutar aplikacije implementiran je PieChart graf. Koji se nalazi u glasačkom sučelju pojedine ankete. Tako da svaka anketa ima svoji graf s postotkom glasova za i protiv. Za to je bilo potrebno inicijalizirati novi PieChart graf i novu listu u kojoj je pohranjen broj glasova za i protiv. Prilikom inicijalizacije PieChart grafa predaju se vrijednosti pohranjene u listi te je naknadno odabrana boja za glasove.

```
List<PieEntry> valuesForPie = new ArrayList<>();
valuesForPie.add(new PieEntry(currentTheme.getThemeLikes()));
valuesForPie.add(new PieEntry(currentTheme.getThemeDislikes()));
PieDataSet pieDataSet = new PieDataSet(valuesForPie,  label: "Grades");
pieDataSet.setColors(Color.GREEN, Color.RED);
PieData pieData = new PieData(pieDataSet);
img_iconTheme.setData(pieData);
```

Slika 4.23 Prikaz programskog rješenja za PieChart graf

5. KORISNIČKO SUČELJE

U ovom poglavlju pokazat će se izgled korisničkog sučelja i opisati izgled svih već spomenutih i objašnjenih aktivnosti koje se nalaze unutar aplikacije.

5.1. Korisničko sučelje registracije i prijave

Prilikom pokretanja aplikacije otvara se početna aktivnost u kojoj je moguća registracija i prijava. Unutar ove aktivnosti nalazi ilustracija dvije osobe u razgovoru (ImageView XML element), gumb za prijavu (Button XML element) i dva prostora za upis korisničkog imena i lozinke (oboje su EditText XML elementi). Od korisnika se traži da unese ime i lozinku uz već spomenute uvijete te pritiskom na gumb „PRIJAVA“ ako nije registriran nudi mu se mogućnost registracije, a ukoliko je već postojeći korisnik prijavljuje se u aplikaciju i otvara se nova aktivnost glavnog izbornika.

Slika 5.1 Prikaz početne aktivnosti

5.2. Korisničko sučelje glavnog izbornika

Unutar glavnog izbornika omogućen je prikaz anketa, te pregledat broja glasova i njihov omjer u postotku. Također moguće je dodavati nove ankete te pristupiti aktivnosti komentiranja i glasanja ankete kao i aktivnosti uređivanja ankete ili je moguće obrisati anketu.

Slika 5.2 Prikaz glavnog izbornika

Kao što je već objašnjeno moguće je i filtrirati prikaz anketa pritiskom na tri vertikalne postavljenje točke u gornjem desnom kutu. Pritiskom se otvara padajući izbornik iz kojega je moguć odabir željenog načina sortiranja.

Slika 5.3 Prikaz padajućeg izbornika

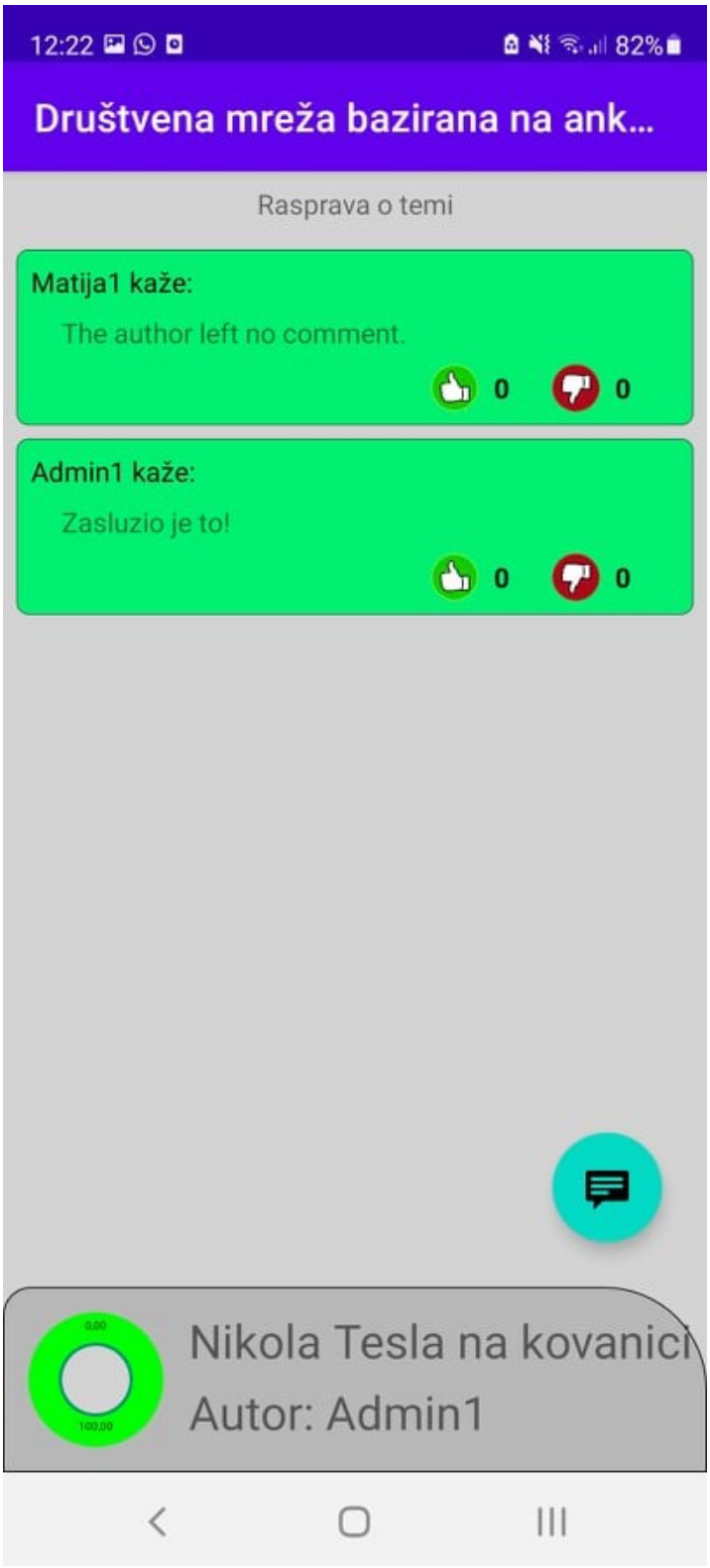
5.3. Korisničko sučelje dodavanja nove ankete

Pritiskom na gumb „DODAJ NOVU TEMU“ u glavnom izborniku otvara se nova aktivnost u kojoj je moguće dodati novu anketu.

Slika 5.4 Prikaz aktivnosti dodavanja nove ankete

Unutar ove aktivnosti moguće je odabrati ime ankete, te opis iste, i pritiskom na gumb „DODAJ NOVU TEMU“ anketa se dodaje u bazu podataka i automatski ispisuje unutar glavnog izbornika.

5.4. Korisničko sučelje komentiranja i glasanja



Slika 5.5 Prikaz aktivnosti komentiranja i glasanja

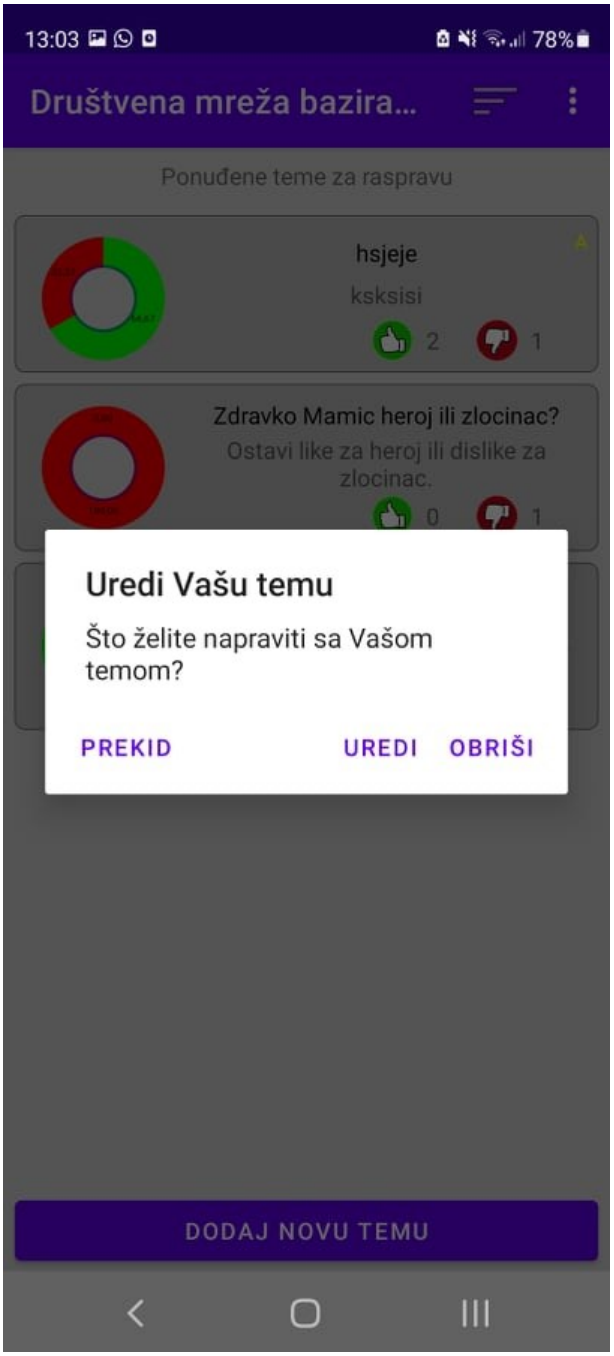
Unutar ove aktivnosti moguće je pregledavati komentare i glasove pojedine ankete. Do ove aktivnosti se dolazi klikom na anketu unutar glavnog izbornika. Također unutar ove ankete moguće je ostaviti vlastit komentar ili glasovati za ili protiv.

Slika 5.6 Prikaz mogućnosti glasanja i/ili komentiranja

5.5. Korisničko sučelje uređivanja i brisanja ankete

Kako bi korisnik bio u mogućnosti urediti ili obrisati anketu, on mora biti i autor iste. Svaki korisnik u gornjem desnom kutu ankete ima prikazno veliko tiskano slovo A ako je autor ankete.

Pritiskom i držanjem na anketu otvara se izbornik u kojem korisnik može odabrati želi li urediti ili obrisati anketu ili želi odustati od bilo kakve radnje.



Slika 5.7 Prikaz izbornika

Ako se korisnik odluči za brisanje, anketa se briše iz baze podataka. Ako se korisnik odluči za odabir uređivanja otvara se nova aktivnost unutar koje je moguće promijeniti ime i opis ankete (Ako korisnik uredi anketu prethodni glasovi i komentari se brišu).

Slika 5.8 Prikaz aktivnosti uređivanja ankete

6. ZAKLJUČAK

Kako je čovjek misaono i društveno biće mobilne aplikacije i društvene mreže za komuniciranje postale su dio svakodnevice u svim dijelovima života od privatnog do poslovnog. Ovaj rad zahtjeva predznanje i volju za istraživanjem i proučavanjem materijala te stvaranja vlastitih ideja. Aplikacija je velikom dijelu napravljena u Android Studiu, uz spomenuto okruženje korišten je i vanjski Firebase servis za bazu podataka. Cilj ovog završnog rada bio je napraviti aplikaciju koja će korisnicima omogućiti postavljanje anketa i mogućnost rasprave i ostavljanja mišljena. Korisničko sučelje je vrlo jednostavno i razumljivo kako ne bi predstavljalo prepreku pri korištenju. Također u daljnjem razvoju ove aplikacije moguće je nadograditi funkcionalnosti ili dodati nove ako je to potrebno.

LITERATURA

[1] Wikipedija, Društvena mreža, dostupno na:

https://hr.wikipedia.org/wiki/Dru%C5%A1tvena_mre%C5%BEa, (kolovoz 2021.)

[2] Wikipedija, Android operacijski sustav, dostupno na:

https://hr.wikipedia.org/wiki/Android_(operacijski_sustav), (kolovoz 2021.)

[3] Wikipedija, Android Studio, dostupno na:

https://en.wikipedia.org/wiki/Android_Studio, (kolovoz 2021.)

[4] Wikipedija, XML, dostupno na:

https://hr.wikipedia.org/wiki/XML, (kolovoz 2021.)

[5] Wikipedija, Java programski jezik, dostupno na:

https://hr.wikipedia.org/wiki/Java_(programski_jezik), (kolovoz 2021.)

[6] Wikipedija, Firebase, dostupno na:

https://en.wikipedia.org/wiki/Firebase, (kolovoz 2021.)

[7] Pollie: Create Polls, dostupno na:

https://pollie.app/, (kolovoz 2021.)

[8] Poll Everywhere, dostupno na:

https://www.polleverywhere.com/, (kolovoz 2021.)

[9] Android arhitektura, dostupno na:

https://www.researchgate.net/figure/Android-low-level-system-architecture_fig1_270576401, (kolovoz 2021.)

SAŽETAK

U ovom završnom radu razvijena je mobilna android aplikacija koja omogućava postavljane anketa na platformu, te razmjenu komentara i glasova, tj. stavova oko ankete s drugim korisnicima aplikacije. Korisnicima je omogućena registracija i prijava u aplikaciju te su s podacima koje unesu tijekom registracije predstavljeni u daljnjem korištenju. Glavni cilj aplikacije je omogućiti anonimnu objavu i komentiranje različitih anketa koje korisnici objavljuju. Aplikacija je izrađena u programskom okruženju Android Studio u Java programskom jeziku. Za bazu podataka je korišten Firebase servis i njegova baza podataka u stvarnom vremenu. U radu su objašnjeni ključne funkcionalnosti aplikacije te su algoritmi kao i izgled korisničkog sučelja popraćeni slikama kako bi se prikazalo što je objašnjeno u radu.

Ključne riječi: Android Studio, anketa, Java, mobilna aplikacija

ABSTRACT

Social network based on surveys

In this final paper, a mobile Android application was developed and it allows posting new polls on the platform and posting comments and votes on other users polls. Users must register and log in to the application in order to use it. The main objective of the application is to allow anonymous posting and commenting on various surveys published by users. The application was created in the Android Studio programming environment in the Java programming language. The Firebase service and its real-time database were used for the database. The paper explains the key functionalities of the application and the algorithms as well as the appearance of the user interface are accompanied by images to show what is explained in the paper.

Key words: Android studio, Java, mobile application, survey

ŽIVOTOPIS

Matija Pleša rođen je u Zagrebu 5.srpnja 1998. godine. U Kutini završava osnovnu školu i tehničku školu smjera tehničar za računarstvo s odličnim uspjehom. Nakon završetka tehničke škole 2017. godine upisuje se na preddiplomski studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Cilj mu je nakon završetka preddiplomskog studija upisati diplomskih studij na istoimenome fakultetu. Razvoj android aplikacija ga jako zanima, također kao i jezici Java i JavaScript, te baze podataka.

Matija Pleša

PRILOZI

Na optičkom disku u prilogu nalazi se završni rad u .docx i .pdf formatu, izvorni kod aplikacije te sama aplikacija.