CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION

The lack of a specific social media platform makes it difficult for students to collaborate, share resources, and communicate effectively in modern academic environments. This project fills this gap by creating Spotlight, a cutting-edge Android application with a feature-rich suite catered to the particular requirements of student communities that improves the collegiate experience.

1.2 OBJECTIVES

The main objectives of the project are:

- ✓ Boost Connectivity: Establish a platform that makes it easier for college students to network and connect with one another.
- ✓ Facilitate Resource Sharing: Encourage the sharing of information by offering resources for exchanging scholarly findings and information.
- ✓ Promote Event Engagement: Through specialized listings and promotions, promote attendance at college events and activities.
- ✓ Encourage Collaborative initiatives: Give students the freedom to work together on extracurricular and academic initiatives to promote creativity and cooperation.
- ✓ Optimize Organization Management: Give clubs and organizations of students specialized tools so they can interact with members and run events effectively.

1.3 EXPECTED OUTCOMES

The Spotlight social media application aims to create a dynamic and connected college community. Improved connectivity will benefit users by encouraging deep connections and knowledge exchange. The platform will act as a vibrant centre for student organization management, cooperative project development, and event participation. Spotlight strives to provide an inclusive and engaging social space that maximizes the college experience by ensuring that students can easily connect, cooperate, and contribute to a vibrant academic and social ecosystem. It does this through its user-friendly design and real-time interaction capabilities.

CHAPTER 2
FUNDAMENTALS OF TECHNOLOGIES USED

2.1 ANDROID

Google created Android, a popular mobile operating system known for its adaptability, easy-to-use interface, and open-source status. Since its 2008 release, Android has developed into a powerful force in the smartphone market, powering a wide variety of gadgets. Android, which is well-known for being highly customizable, provides developers with an adaptable platform on which to construct creative applications utilizing the Java and Kotlin programming languages. Android's app store, Google Play Store, offers millions of apps with a wide range of features and interests. The secret to Android's popularity is its versatility—it works with everything from smart TVs and wearables to smartphones and tablets. In order to give consumers a seamless digital experience, the operating system places a strong emphasis on seamless interaction with Google services.

Linux kernel

The fundamental component of the Linux operating system, the kernel acts as a link between software and hardware. It was created by Linus Torvalds and the open-source community and offers basic features including memory allocation, device drivers, system calls, and process management. It serves as the basis for many Linux distributions due to its open and modular design, which facilitates extensive modification and adaption. The reputation of Linux as a strong and adaptable operating system for a variety of computing devices, from servers and embedded systems to personal computers, is largely due to the kernel's dependability, security, and efficiency.

2.2 ACTIVITY LIFE CYCLE

An essential component of developing Android apps is the Activity Lifecycle, which lists the different stages an activity might go through. An activity's lifetime comprises important phases like creation, starting, resuming, pausing, ending, and destruction. An activity is a single screen with a user interface. An activity goes through a number of transitions upon launch, depending on system events and user involvement. To handle these changes and guarantee appropriate resource allocation and release, data permanence, and a smooth user experience, developers employ lifecycle techniques. Developers may react to changes in the app's status, including handling interruptions or adjusting for various device setups, thanks to the lifecycle. Developing dependable and responsive Android applications that provide a consistent and dependable user experience across a range of devices and settings requires an understanding of and skill with the Activity Lifecycle.

2.3 XML IN ANDROID STUDIO

XML, or Extensible Markup Language, is essential for specifying the layout and style of user interfaces in Android Studio. Layout files, which define the arrangement and appearance of different UI elements in an Android application, are created using XML. Data binding is then used to link these layout files, which are kept in the "res/layout" directory, to matching Java or Kotlin code. The hierarchical structure of XML enables developers to define relationships between various parts, specify properties, and designate user interface components. In addition to XML code, Android Studio has a visual editor that makes UI design more approachable with a WYSIWYG (What You See Is What You Get) methodology. Developers can now design visually appealing and responsive user interfaces with a distinct division between the presentation layer and the functionality of the program thanks to the combination of XML and visual tools.

2.4 FIREBASE STORAGE

A cloud-based solution called Firebase Storage makes it possible for online and mobile applications to upload and download files in a safe and scalable manner. It provides developers with a dependable and effective method of storing and retrieving user-generated material, including documents, videos, and photographs, by integrating with the larger Firebase platform. Firebase Storage offers developers a smooth way to manage storage requirements for their apps by streamlining the management of multimedia assets with features like access control, automatic scaling, and simple SDK connection.

CHAPTER 3

REQUIREMENT SPECIFICATIONS

3.1 MOBILE REQUIRMENTS

- A smartphone running Android version 11 or higher
- A USB cable to link the tablet or phone to a PC.
- Constant internet connectivity.

3.2 COMPUTER REQUIREMENTS

- Android studio JDK 17.0.1 and above
- Gradle 7.5 and above
- 64 - bit Microsoft® Windows® 8/10/11 or MacOS® 10.14 (Mojave) or higher
- 8GB RAM minimum
- 8 GB of available disk space minimum (IDE + Android SDK + Android Emulator)
- Intel Core i5 minimum
- 2.1 GHz processor
- Firebase for backend

CHAPTER 4

DESIGN

4.1 FLOWCHART

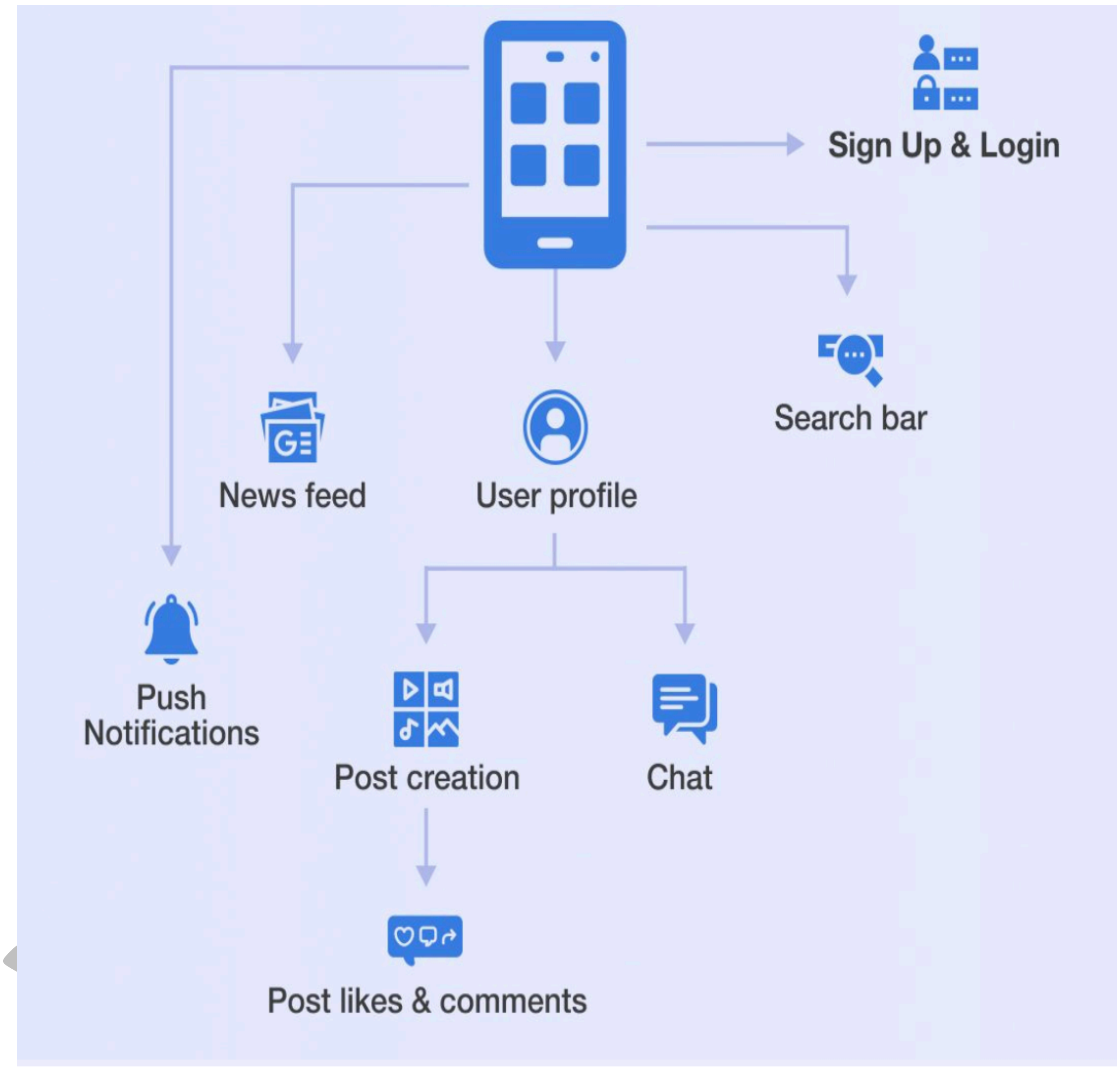


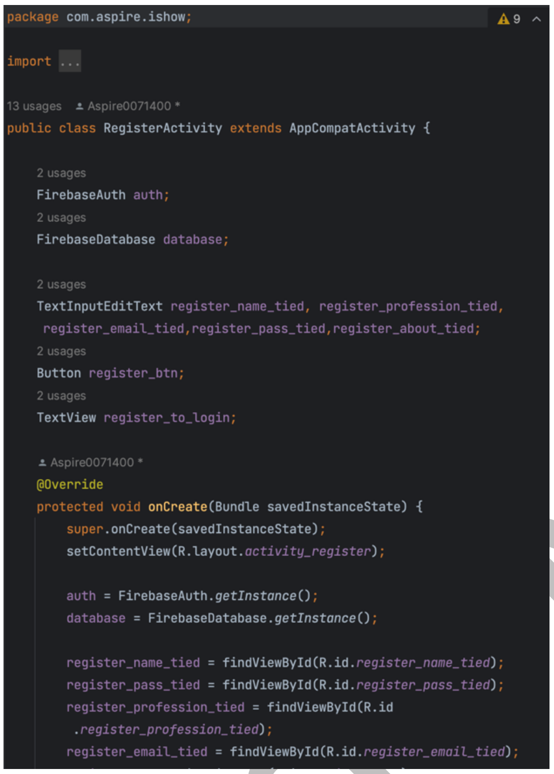
Fig 4.1 Flowchart

CHAPTER 5

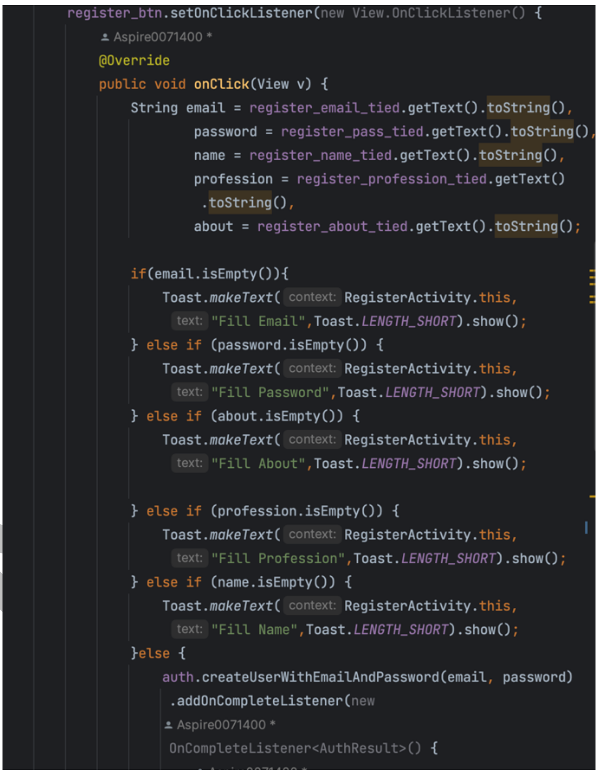
IMPLEMENTATION

5.1 REGISTRATION ACTIVITY

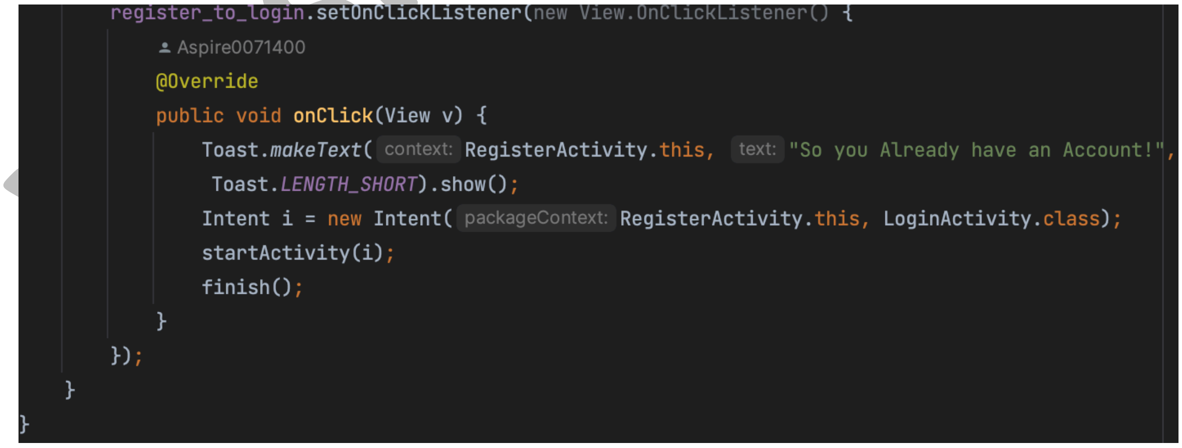
It allows new users to register themselves to the application.



```
package com.aspire.ishow;                                              ⚠9 ^

import ...

13 usages   Aspire0071400 *
public class RegisterActivity extends AppCompatActivity {

    2 usages
    FirebaseAuth auth;
    2 usages
    FirebaseDatabase database;

    2 usages
    TextInputEditText register_name_tied, register_profession_tied,
     register_email_tied,register_pass_tied,register_about_tied;
    2 usages
    Button register_btn;
    2 usages
    TextView register_to_login;

    Aspire0071400 *
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);

        auth = FirebaseAuth.getInstance();
        database = FirebaseDatabase.getInstance();

        register_name_tied = findViewById(R.id.register_name_tied);
        register_pass_tied = findViewById(R.id.register_pass_tied);
        register_profession_tied = findViewById(R.id
         .register_profession_tied);
        register_email_tied = findViewById(R.id.register_email_tied);
```



```
        register_btn.setOnClickListener(new View.OnClickListener() {
            Aspire0071400 *
            @Override
            public void onClick(View v) {
                String email = register_email_tied.getText().toString(),
                        password = register_pass_tied.getText().toString(),
                        name = register_name_tied.getText().toString(),
                        profession = register_profession_tied.getText()
                         .toString(),
                        about = register_about_tied.getText().toString();

                if(email.isEmpty()){
                    Toast.makeText( context: RegisterActivity.this,
                     text: "Fill Email",Toast.LENGTH_SHORT).show();
                } else if (password.isEmpty()) {
                    Toast.makeText( context: RegisterActivity.this,
                     text: "Fill Password",Toast.LENGTH_SHORT).show();
                } else if (about.isEmpty()) {
                    Toast.makeText( context: RegisterActivity.this,
                     text: "Fill About",Toast.LENGTH_SHORT).show();

                } else if (profession.isEmpty()) {
                    Toast.makeText( context: RegisterActivity.this,
                     text: "Fill Profession",Toast.LENGTH_SHORT).show();
                } else if (name.isEmpty()) {
                    Toast.makeText( context: RegisterActivity.this,
                     text: "Fill Name",Toast.LENGTH_SHORT).show();
                }else {
                    auth.createUserWithEmailAndPassword(email, password)
                     .addOnCompleteListener(new
                     Aspire0071400 *
                     OnCompleteListener<AuthResult>() {
```

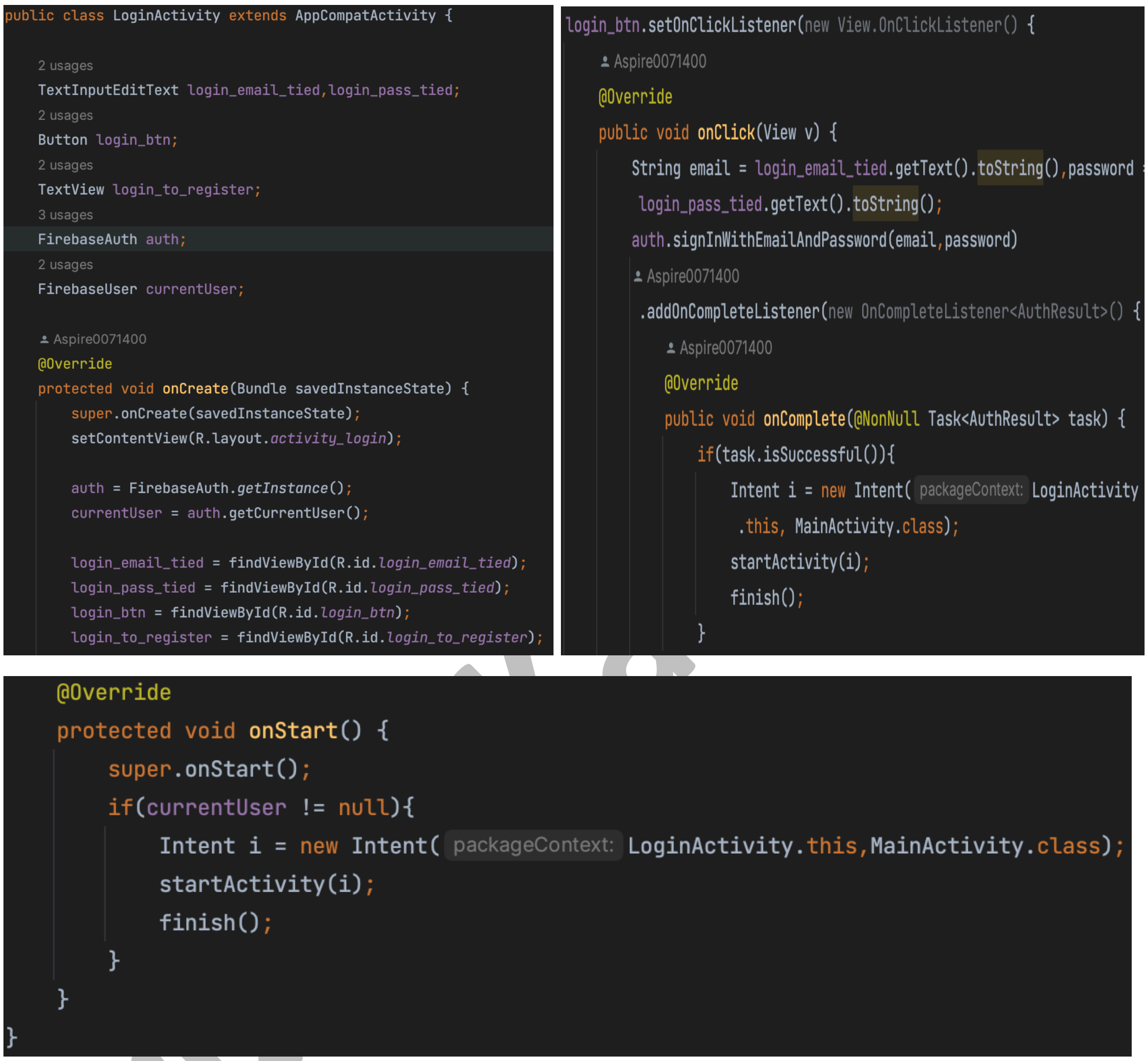


```
        register_to_login.setOnClickListener(new View.OnClickListener() {
            Aspire0071400
            @Override
            public void onClick(View v) {
                Toast.makeText( context: RegisterActivity.this,  text: "So you Already have an Account!",
                 Toast.LENGTH_SHORT).show();
                Intent i = new Intent( packageContext: RegisterActivity.this, LoginActivity.class);
                startActivity(i);
                finish();
            }
        });
    }
}
```

Fig 5.1 Registration activity

5.2 LOGIN ACTIVITY

Functions to allow registered users to Login to the application.



```
public class LoginActivity extends AppCompatActivity {

    2 usages
    TextInputEditText login_email_tied,login_pass_tied;
    2 usages
    Button login_btn;
    2 usages
    TextView login_to_register;
    3 usages
    FirebaseAuth auth;
    2 usages
    FirebaseUser currentUser;

    Aspire0071400
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        auth = FirebaseAuth.getInstance();
        currentUser = auth.getCurrentUser();

        login_email_tied = findViewById(R.id.login_email_tied);
        login_pass_tied = findViewById(R.id.login_pass_tied);
        login_btn = findViewById(R.id.login_btn);
        login_to_register = findViewById(R.id.login_to_register);
```

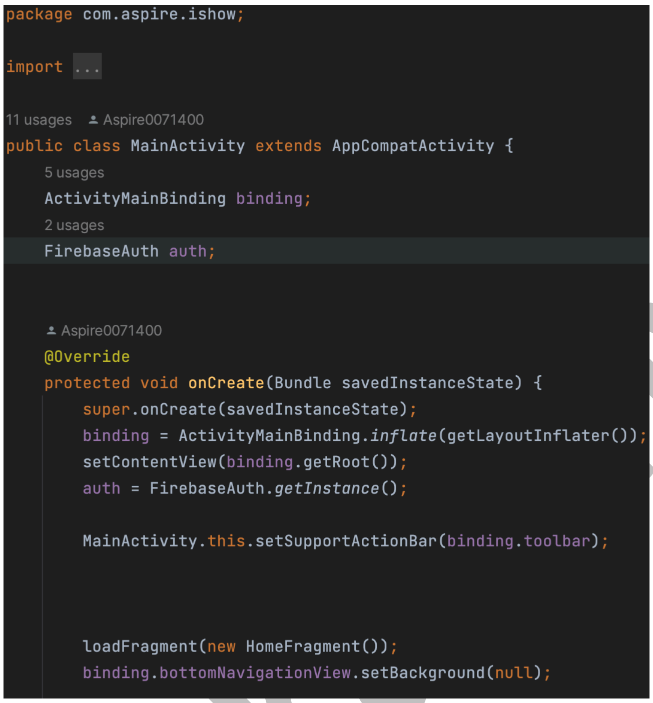
```
login_btn.setOnClickListener(new View.OnClickListener() {
    Aspire0071400
    @Override
    public void onClick(View v) {
        String email = login_email_tied.getText().toString(),password
         login_pass_tied.getText().toString();
        auth.signInWithEmailAndPassword(email,password)
        Aspire0071400
         .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            Aspire0071400
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    Intent i = new Intent( packageContext: LoginActivity
                     .this, MainActivity.class);
                    startActivity(i);
                    finish();
                }
```

```
    @Override
    protected void onStart() {
        super.onStart();
        if(currentUser != null){
            Intent i = new Intent( packageContext: LoginActivity.this,MainActivity.class);
            startActivity(i);
            finish();
        }
    }
}
```

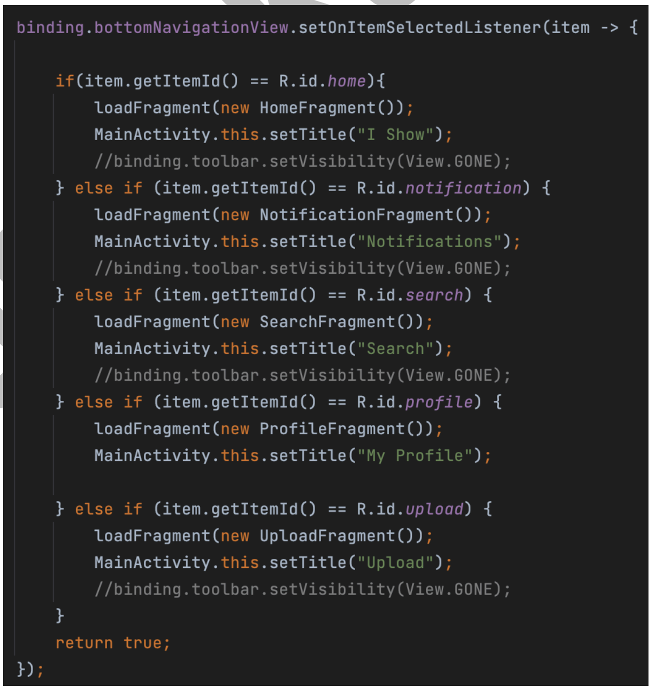
Fig 5.2 Login Activity

5.3 MAIN ACTIVITY

The main activity window of the application.



```
package com.aspire.ishow;

import ...

11 usages  Aspire0071400
public class MainActivity extends AppCompatActivity {
    5 usages
    ActivityMainBinding binding;
    2 usages
    FirebaseAuth auth;


    Aspire0071400
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());
        auth = FirebaseAuth.getInstance();

        MainActivity.this.setSupportActionBar(binding.toolbar);



        loadFragment(new HomeFragment());
        binding.bottomNavigationView.setBackground(null);
```

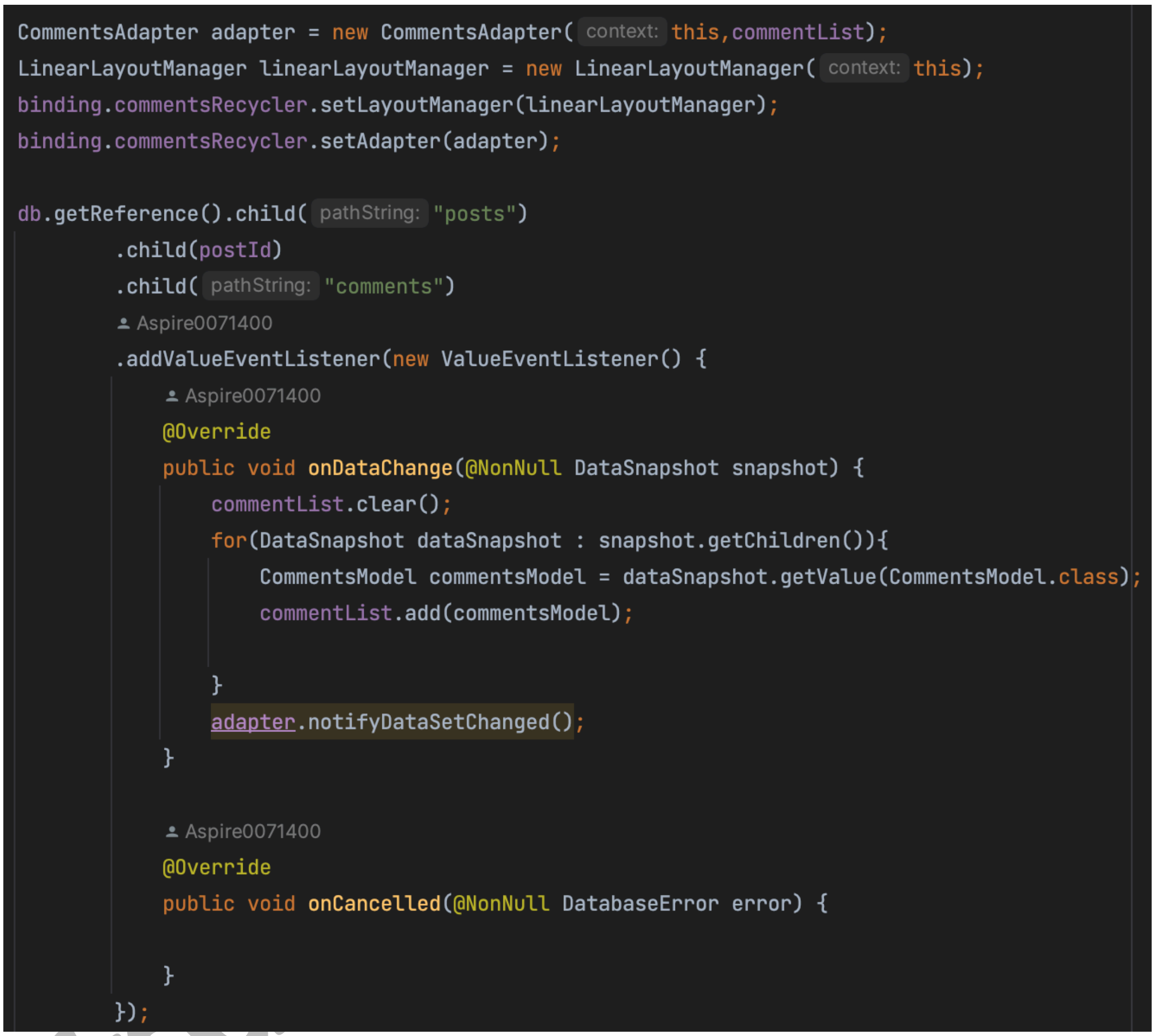


```
binding.bottomNavigationView.setOnItemSelectedListener(item -> {

    if(item.getItemId() == R.id.home){
        loadFragment(new HomeFragment());
        MainActivity.this.setTitle("I Show");
        //binding.toolbar.setVisibility(View.GONE);
    } else if (item.getItemId() == R.id.notification) {
        loadFragment(new NotificationFragment());
        MainActivity.this.setTitle("Notifications");
        //binding.toolbar.setVisibility(View.GONE);
    } else if (item.getItemId() == R.id.search) {
        loadFragment(new SearchFragment());
        MainActivity.this.setTitle("Search");
        //binding.toolbar.setVisibility(View.GONE);
    } else if (item.getItemId() == R.id.profile) {
        loadFragment(new ProfileFragment());
        MainActivity.this.setTitle("My Profile");

    } else if (item.getItemId() == R.id.upload) {
        loadFragment(new UploadFragment());
        MainActivity.this.setTitle("Upload");
        //binding.toolbar.setVisibility(View.GONE);
    }
    return true;
});
```

Fig 5.3 MAIN Activity

5.4 COMMENT ACTIVITY

Used to comment on the posts.

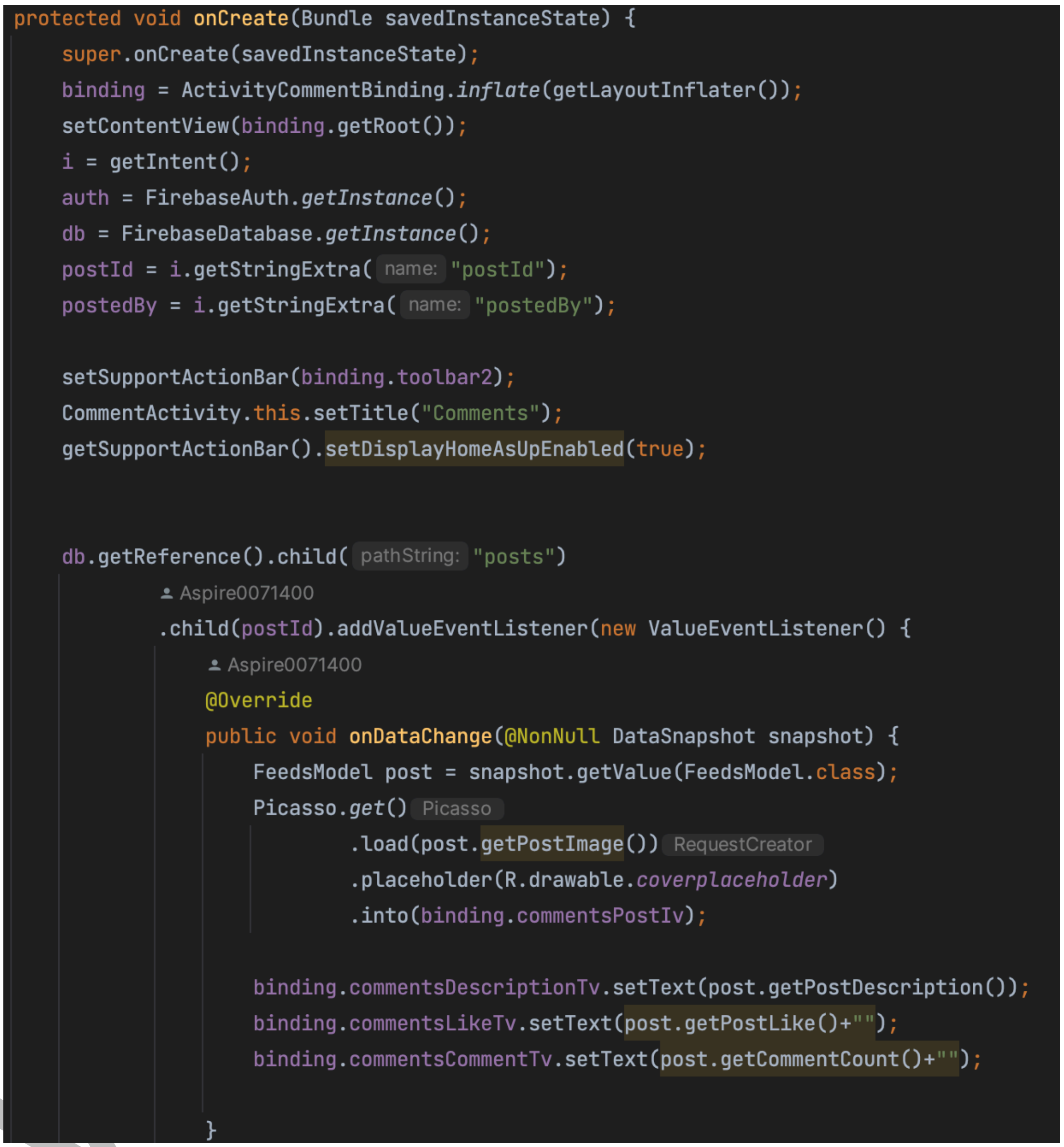


```
CommentsAdapter adapter = new CommentsAdapter( context: this,commentList);
LinearLayoutManager linearLayoutManager = new LinearLayoutManager( context: this);
binding.commentsRecycler.setLayoutManager(linearLayoutManager);
binding.commentsRecycler.setAdapter(adapter);

db.getReference().child( pathString: "posts")
        .child(postId)
        .child( pathString: "comments")
        Aspire0071400
        .addValueEventListener(new ValueEventListener() {
            Aspire0071400
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                commentList.clear();
                for(DataSnapshot dataSnapshot : snapshot.getChildren()){
                    CommentsModel commentsModel = dataSnapshot.getValue(CommentsModel.class);
                    commentList.add(commentsModel);

                }
                adapter.notifyDataSetChanged();
            }

            Aspire0071400
            @Override
            public void onCancelled(@NonNull DatabaseError error) {

            }
        });
```

Fig 5.4 Comment Activity

5.5 LIKE ACTIVITY

Used to like the uploaded post on the application.

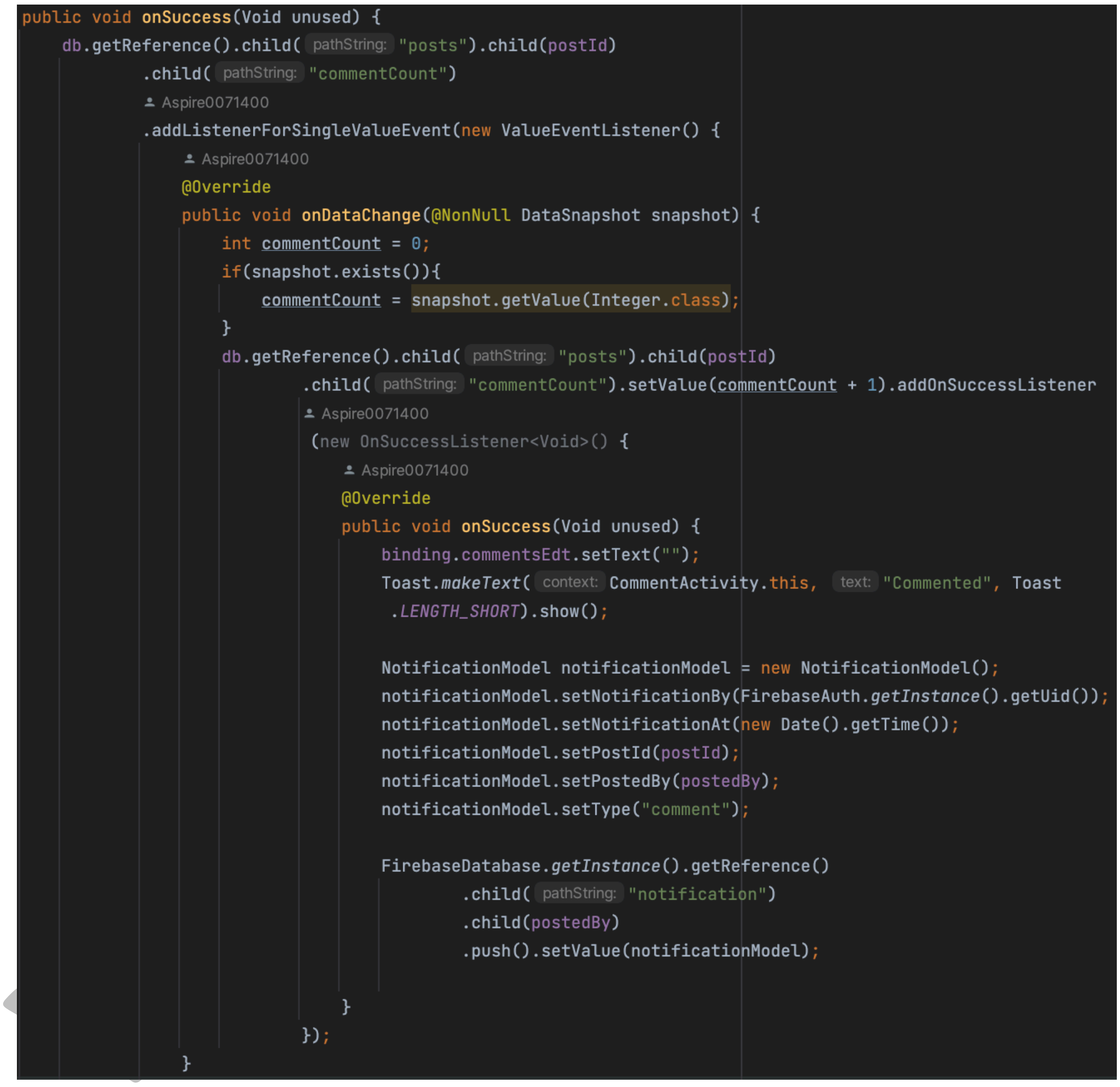


```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    binding = ActivityCommentBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());
    i = getIntent();
    auth = FirebaseAuth.getInstance();
    db = FirebaseDatabase.getInstance();
    postId = i.getStringExtra( name: "postId");
    postedBy = i.getStringExtra( name: "postedBy");

    setSupportActionBar(binding.toolbar2);
    CommentActivity.this.setTitle("Comments");
    getSupportActionBar().setDisplayHomeAsUpEnabled(true);


    db.getReference().child( pathString: "posts")
            Aspire0071400
            .child(postId).addValueEventListener(new ValueEventListener() {
                Aspire0071400
                @Override
                public void onDataChange(@NonNull DataSnapshot snapshot) {
                    FeedsModel post = snapshot.getValue(FeedsModel.class);
                    Picasso.get() Picasso
                            .load(post.getPostImage()) RequestCreator
                            .placeholder(R.drawable.coverplaceholder)
                            .into(binding.commentsPostIv);

                    binding.commentsDescriptionTv.setText(post.getPostDescription());
                    binding.commentsLikeTv.setText(post.getPostLike()+"");
                    binding.commentsCommentTv.setText(post.getCommentCount()+"");

                }
```

Fig 5.5 Like Activity

5.6 NOTIFICATION ACTIVITY

It shows all the latest notifications related to the uploaded posts to the user on the applications.

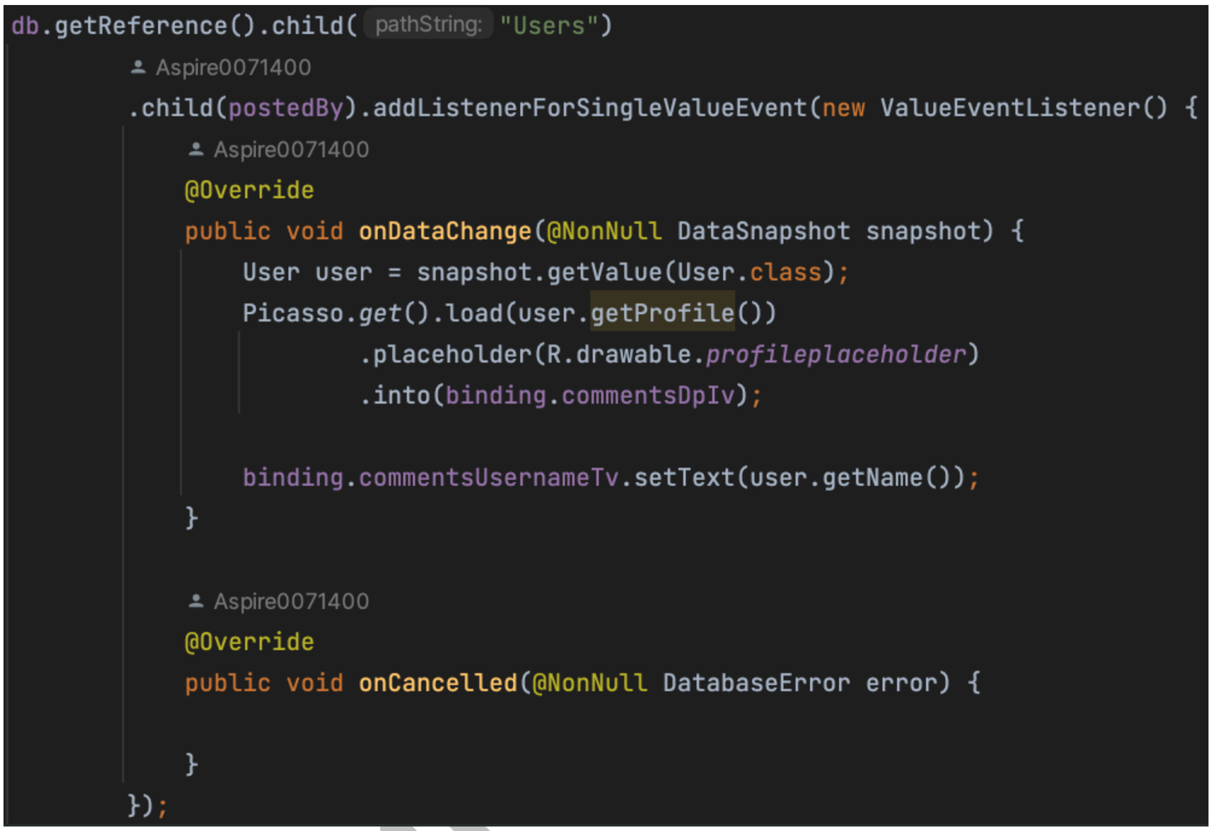


```
public void onSuccess(Void unused) {
    db.getReference().child( pathString: "posts").child(postId)
            .child( pathString: "commentCount")
            Aspire0071400
            .addListenerForSingleValueEvent(new ValueEventListener() {
                Aspire0071400
                @Override
                public void onDataChange(@NonNull DataSnapshot snapshot) {
                    int commentCount = 0;
                    if(snapshot.exists()){
                        commentCount = snapshot.getValue(Integer.class);
                    }
                    db.getReference().child( pathString: "posts").child(postId)
                            .child( pathString: "commentCount").setValue(commentCount + 1).addOnSuccessListener
                            Aspire0071400
                            (new OnSuccessListener<Void>() {
                                Aspire0071400
                                @Override
                                public void onSuccess(Void unused) {
                                    binding.commentsEdt.setText("");
                                    Toast.makeText( context: CommentActivity.this,  text: "Commented", Toast
                                     .LENGTH_SHORT).show();

                                    NotificationModel notificationModel = new NotificationModel();
                                    notificationModel.setNotificationBy(FirebaseAuth.getInstance().getUid());
                                    notificationModel.setNotificationAt(new Date().getTime());
                                    notificationModel.setPostId(postId);
                                    notificationModel.setPostedBy(postedBy);
                                    notificationModel.setType("comment");

                                    FirebaseDatabase.getInstance().getReference()
                                            .child( pathString: "notification")
                                            .child(postedBy)
                                            .push().setValue(notificationModel);

                                }
                            });
                }
```

Fig 5.6 Notification Activity

5.7 USER PROFILE

It shows all logged in user's profile displaying the details of the user..



```
db.getReference().child( pathString: "Users")
        Aspire0071400
        .child(postedBy).addListenerForSingleValueEvent(new ValueEventListener() {
            Aspire0071400
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                User user = snapshot.getValue(User.class);
                Picasso.get().load(user.getProfile())
                        .placeholder(R.drawable.profileplaceholder)
                        .into(binding.commentsDpIv);

                binding.commentsUsernameTv.setText(user.getName());
            }

            Aspire0071400
            @Override
            public void onCancelled(@NonNull DatabaseError error) {

            }
        });
```

Fig 5.7 User Profile

CHAPTER 6

RESULTS

6.1 REGISTRATION ACTIVITY

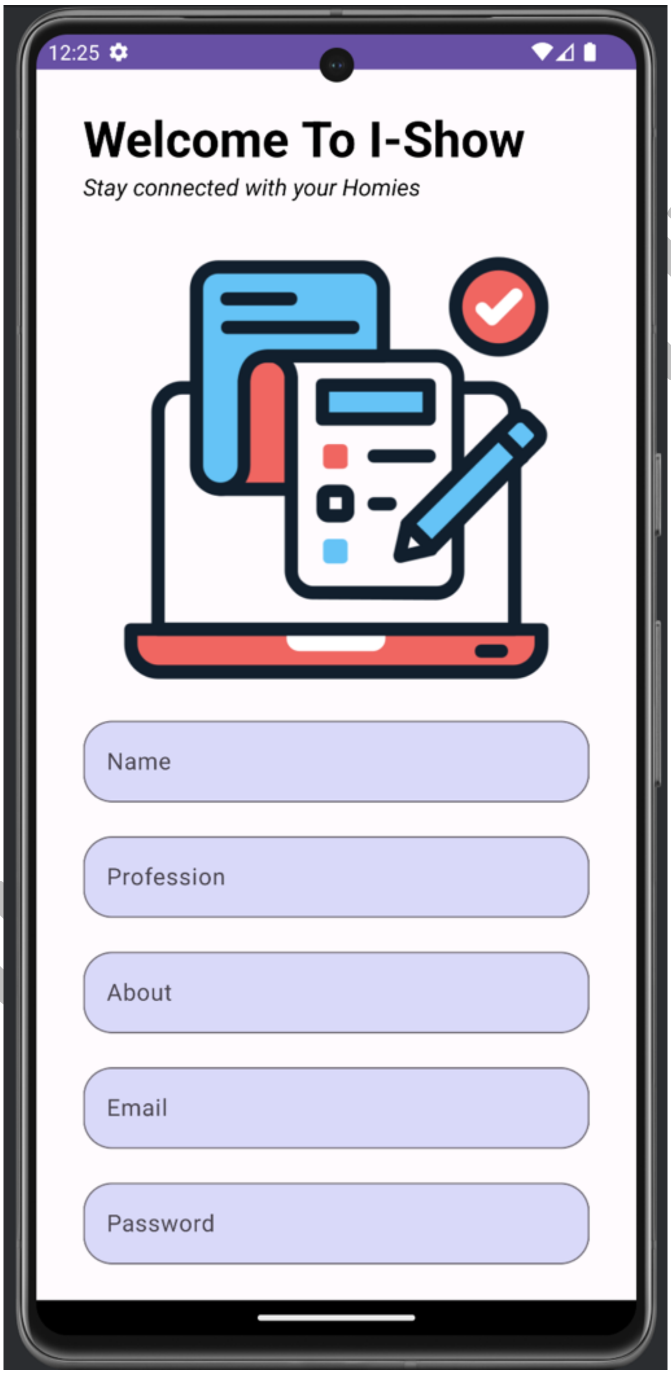


Fig 6.1 REGISTRATION ACTIVITY

6.2 LOGIN ACTIVITY

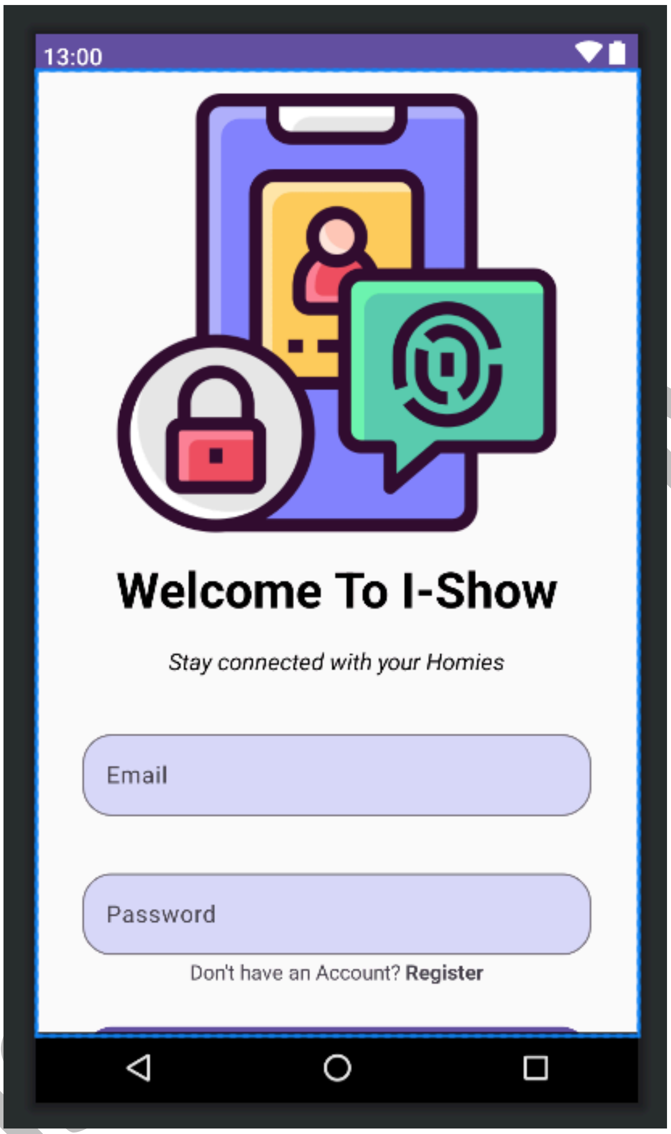

13:00
Welcome To I-Show
Stay connected with your Homies
Email
Password
Don't have an Account? Register

Fig 6.2 LOGIN ACTIVITY

6.3 UPLOAD STORY

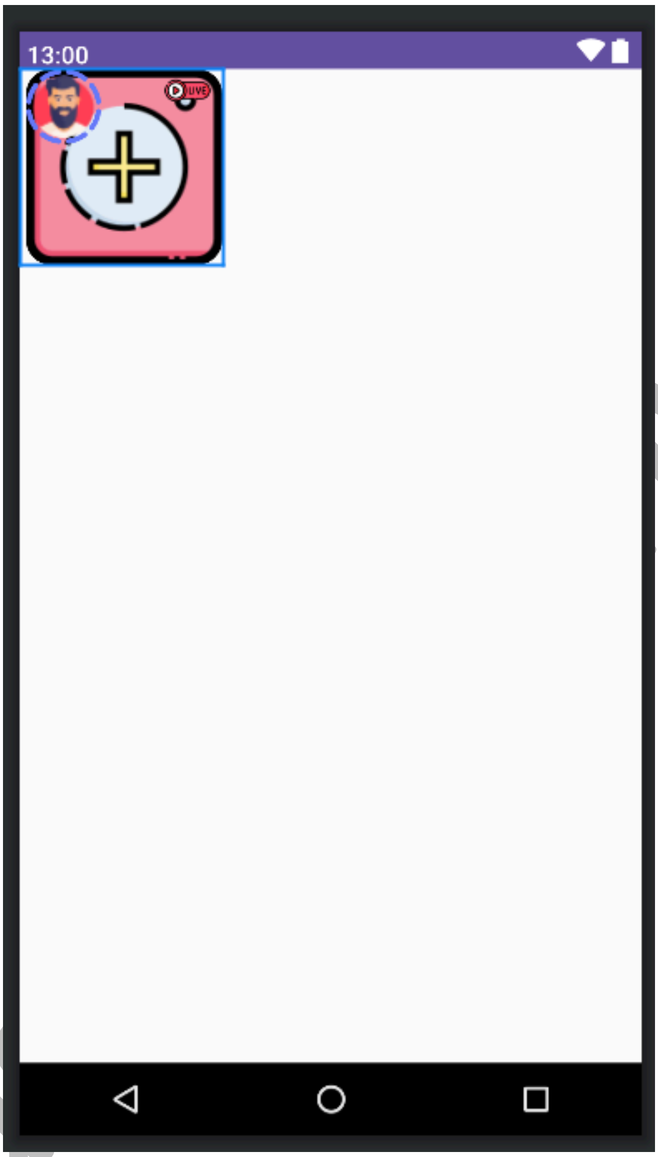


Fig 6.3 UPLOAD STORY

6.4 UPLOAD POST

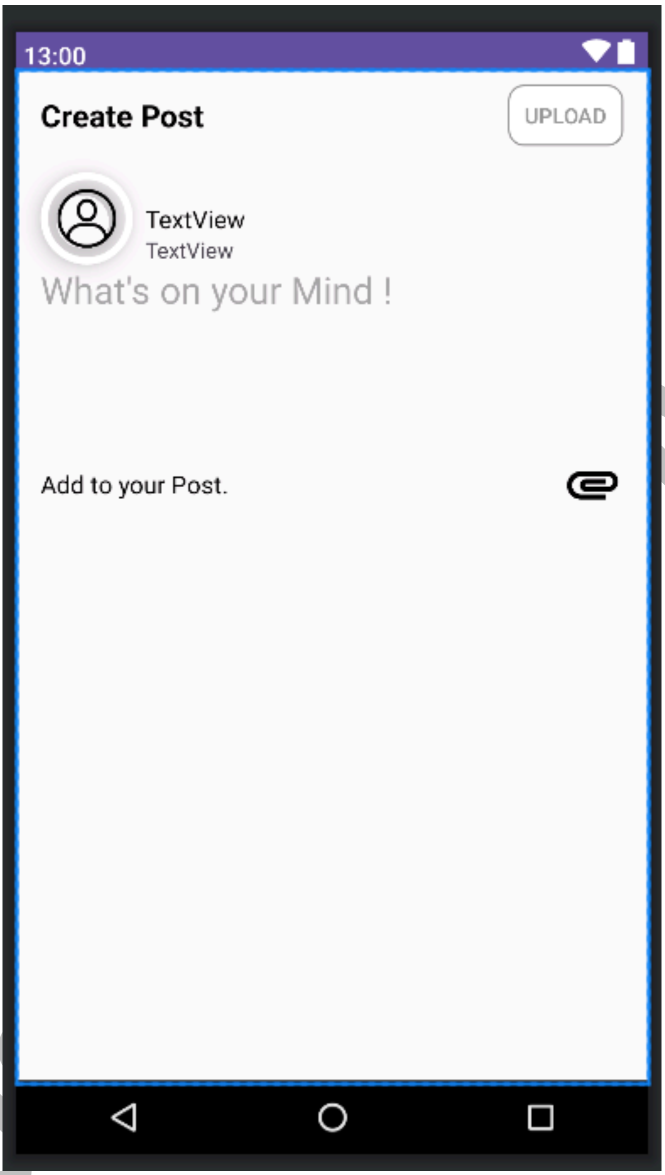
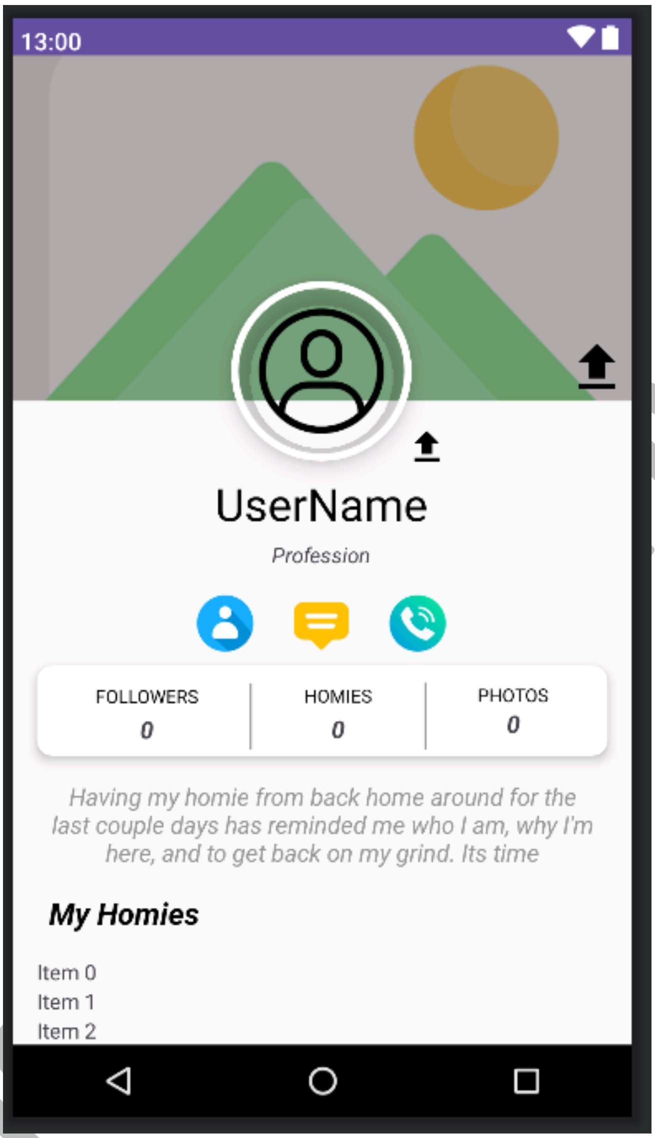


Fig 6.4 UPLOAD POST

6.5 USER PROFILE


13:00

UserName

Profession

FOLLOWERS	HOMIES	PHOTOS
0	*0*	*0*

Having my homie from back home around for the last couple days has reminded me who I am, why I'm here, and to get back on my grind. Its time

My Homies

Item 0
Item 1
Item 2

Fig 6.5 USER PROFILE

6.6 LIKE AND COMMENT ON POST

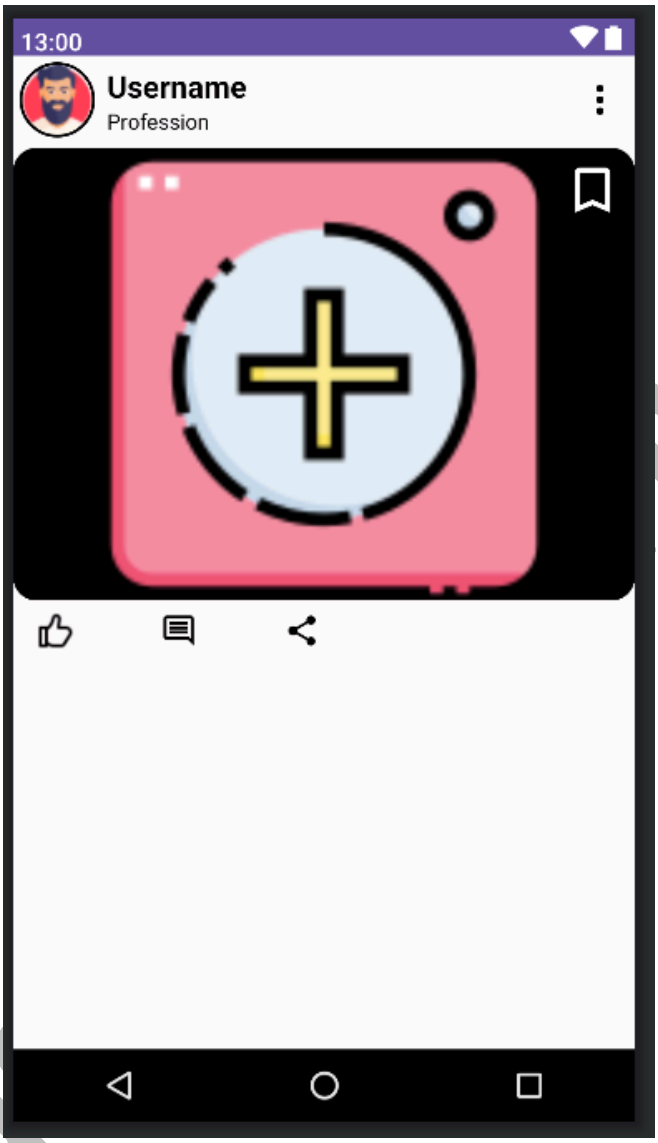
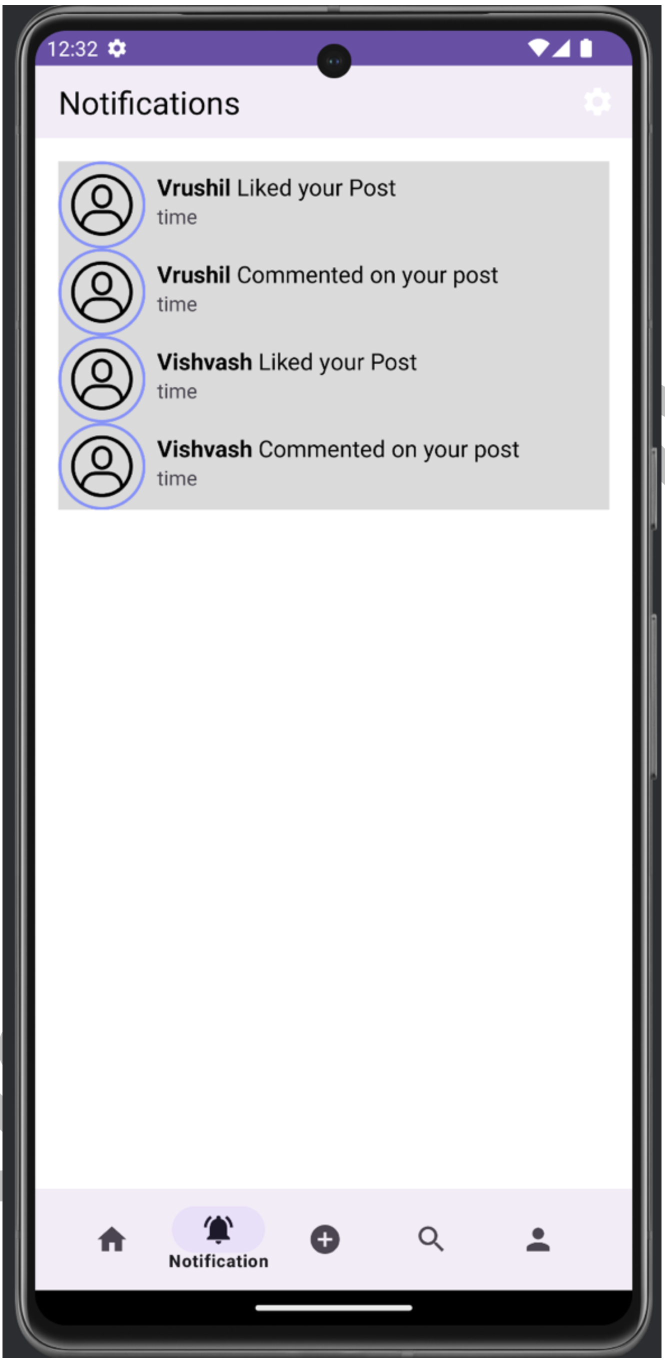


Fig 6.6 LIKE AND COMMENT ON POST

6.7 NOTIFICATION


12:32

Notifications

Vrushil Liked your Post
time

Vrushil Commented on your post
time

Vishvash Liked your Post
time

Vishvash Commented on your post
time

Fig 6.7 NOTIFICATION

CHAPTER 7

CONCLUSION

To sum up, the Spotlight social media project offers a thorough and creative solution designed to meet the changing needs of campus communities. The project successfully alters the college experience by introducing elements including collaboration project spaces, resource sharing, event participation, connection tools, and particular functionality for student groups. The application offers a frictionless and safe environment for students to interact, communicate, and engage in meaningful ways. It was developed on the Android platform utilizing Java and Firebase technologies. Real-time interactions, a user-friendly design, and support for a variety of activities all help to create a lively and welcoming campus community. By developing a social media platform that goes beyond conventional bounds and becomes an essential component of students' academic and social journeys, Spotlight not only meets but exceeds its original goals. Spotlight is a living example of the transformative power of technology in creating connections and cooperation within the distinctive and dynamic terrain of college life, thanks to its extensive feature set and dedication to boosting community participation.

REFERENCES

1. https://developer.android.com/
2. https://firebase.google.com/docs
3. https://www.geeksforgeeks.org/best-way-to-become-android-developer-a-complete-roadmap/
4. https://www.geeksforgeeks.org/how-to-create-a-social-media-app-on-android-studio/
5. https://developer.android.com/reference/app-actions/built-in-intents/social/create-social-media-connection
6. https://developer.android.com/jetpack/androidx/releases/navigation