

**Assignment 2, Mobile developing**  
**Instagram Clone Basic Pages Development**  
**Amanzhol Temirbolat**

**15.10.2024**

**Table of Contents**

<b>1. Introduction.....</b>	<b>1</b>
1.1 Overview of the Assignment	
1.2 Objectives	
<b>2. Project Setup.....</b>	<b>2</b>
2.1 Initial Setup and Dependencies	
2.2 Libraries and Frameworks Used	
<b>3. Page Design.....</b>	<b>3</b>
3.1 Home Feed Page	
3.2 Profile Page	
3.3 Search Page	
3.4 Add Post Page	
3.5 Notifications Page	
<b>4. Navigation.....</b>	<b>4</b>
4.1 Bottom Navigation	
4.2 Fragment Transactions	
<b>5. User Interaction.....</b>	<b>5</b>
5.1 Click Listeners	
5.2 Mock Data Handling	
<b>6. Challenges and Solutions.....</b>	<b>5</b>
<b>7. Conclusion.....</b>	<b>6</b>

**Introduction**

This report outlines the development process for creating a basic Instagram clone using Android Studio. The primary goal of this assignment is to implement core pages such as Home Feed, Profile, Search, Add Post, and Notifications. Through this project, students learn key concepts in mobile programming, including layouts, navigation, RecyclerViews, and user interactions. Additionally, the project simulates data handling and demonstrates the role of UI/UX in app design. This report provides insights into design decisions, challenges faced, and the technical implementation of the application.

**2. Project Setup**

## 2.1 Initial Setup and Dependencies

- Created a new Android Studio project using the **"Empty Activity" template** to build the Instagram clone.
- Set up the necessary **Material Components** library to enhance the UI/UX design.
- Ensured the use of **AndroidX libraries** for modern development practices.

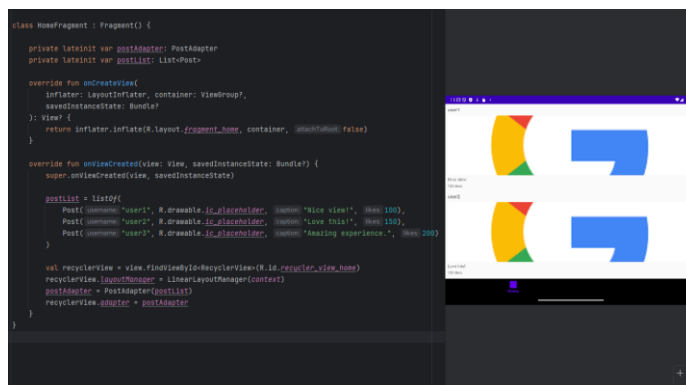
## 2.2 Libraries and Frameworks Used

- **RecyclerView**: For listing posts, search results, and notifications efficiently.
- **Navigation Component**: To handle fragment transactions and manage the back stack.

## 3. Page Design

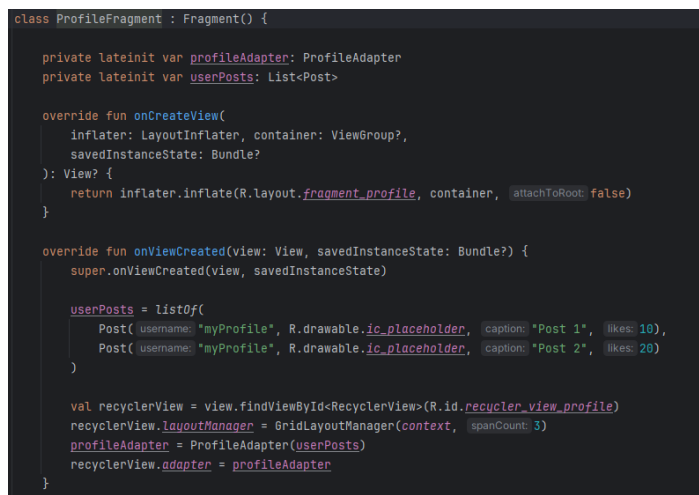
### 3.1 Home Feed Page

- **Layout**: Implemented using a **RecyclerView** to display posts in a vertical list.
- **Content**: Each post includes an **image**, **username**, **caption**, and **like count**.
- **Interaction**: Click listeners allow users to like and comment on posts.



### 3.2 Profile Page

- **Content**: Displays the **profile picture**, **bio**, **post count**, and a grid layout of user posts.
- **Layout**: Implemented with a **GridLayoutManager** inside a RecyclerView for post previews.



### 3.3 Search Page

- **UI:** Includes a **search bar** at the top.
- **Functionality:** Uses **RecyclerView** to show search results as user profiles.

```
class SearchFragment : Fragment() {  
  
    private lateinit var userAdapter: UserAdapter  
    private lateinit var userList: List<User>  
  
    override fun onCreateView(  
        inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle?  
    ): View? {  
        return inflater.inflate(R.layout.fragment_search, container, attachToRoot: false)  
    }  
  
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  
        super.onViewCreated(view, savedInstanceState)  
  
        userList = listOf(  
            User(username = "user1", R.drawable.ic_placeholder, bio = "Bio 1"),  
            User(username = "user2", R.drawable.ic_placeholder, bio = "Bio 2"),  
            User(username = "user3", R.drawable.ic_placeholder, bio = "Bio 3")  
        )  
  
        val recyclerView = view.findViewById<RecyclerView>(R.id.recycler_view_search)  
        recyclerView.layoutManager = LinearLayoutManager(context)  
        userAdapter = UserAdapter(userList)  
        recyclerView.adapter = userAdapter  
    }  
}
```

### 3.4 Add Post Page

- **Form:** Simple input fields for uploading an **image** and adding a **caption**.
- **Button:** A submission button triggers the post upload.

```
import androidx.appcompat.app.AppCompatActivity  
  
class AddPostFragment : Fragment() {  
  
    private lateinit var postCaption: EditText  
    private lateinit var postImage: ImageView  
    private lateinit var btnAddPost: Button  
  
    override fun onCreateView(  
        inflater: LayoutInflater, container: ViewGroup?,  
        savedInstanceState: Bundle?  
    ): View? {  
        return inflater.inflate(R.layout.fragment_add_post, container, attachToRoot: false)  
    }  
  
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  
        super.onViewCreated(view, savedInstanceState)  
  
        postCaption = view.findViewById(R.id.post_caption)  
        postImage = view.findViewById(R.id.post_image)  
        btnAddPost = view.findViewById(R.id.btn_add_post)  
  
        btnAddPost.setOnClickListener {  
            val caption = postCaption.text.toString()  
            if (caption.isNotEmpty()) {  
                Toast.makeText(context, "Post added: $caption", Toast.LENGTH_SHORT).show()  
            } else {  
                Toast.makeText(context, "Caption is empty!", Toast.LENGTH_SHORT).show()  
            }  
        }  
    }  
}
```

### 3.5 Notifications Page

- **Layout:** Uses **RecyclerView** to display notifications like **likes** or **comments**.
- **Interaction:** Notifications link back to relevant posts or profiles.

```

class NotificationsFragment : Fragment() {

    private lateinit var notificationAdapter: NotificationAdapter
    private lateinit var notificationList: List<Notification>

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        return inflater.inflate(R.layout.fragment_notifications, container, attachToRoot: false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        notificationList = listOf(
            Notification(text: "user1 liked your post", isLike: true),
            Notification(text: "user2 commented: Nice pic!", isLike: false),
            Notification(text: "user3 started following you", isLike: false)
        )

        val recyclerView = view.findViewById<RecyclerView>(R.id.recycler_view_notifications)
        recyclerView.layoutManager = LinearLayoutManager(context)
        notificationAdapter = NotificationAdapter(notificationList)
        recyclerView.adapter = notificationAdapter
    }
}

```

This design ensures a user-friendly interface with efficient data handling using RecyclerView throughout the application.

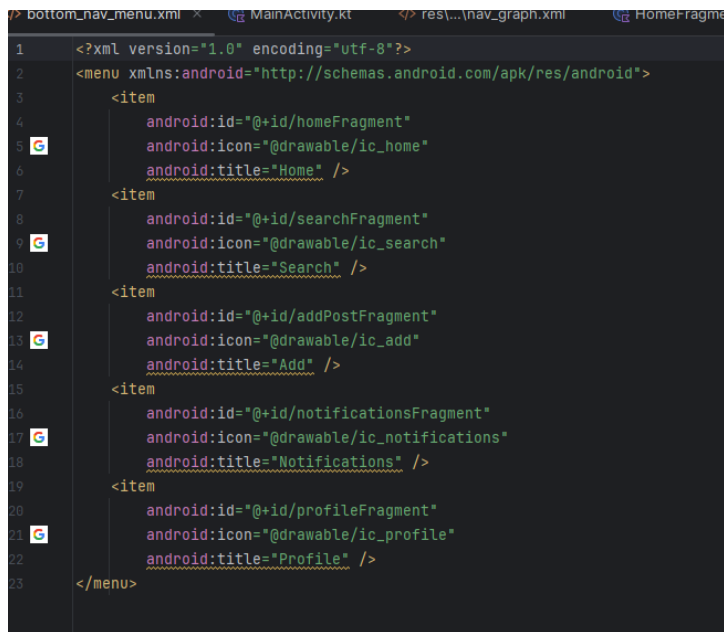
## 4. Navigation

### 4.1 Bottom Navigation

- Implemented **BottomNavigationView** to allow users to seamlessly switch between the **Home Feed, Search, Profile, Add Post, and Notifications** pages.
- Each menu item is associated with a specific fragment representing a core Instagram page.

### 4.2 Fragment Transactions

- Used the **Navigation Component** to manage fragment transactions and handle the back stack.
- Ensured smooth transitions between pages and proper back navigation using **NavHostFragment**.



```

1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:android="http://schemas.android.com/apk/res/android">
3      <item
4          android:id="@+id/homeFragment"
5          android:icon="@drawable/ic_home"
6          android:title="Home" />
7      <item
8          android:id="@+id/searchFragment"
9          android:icon="@drawable/ic_search"
10         android:title="Search" />
11     <item
12         android:id="@+id/addPostFragment"
13         android:icon="@drawable/ic_add"
14         android:title="Add" />
15     <item
16         android:id="@+id/notificationsFragment"
17         android:icon="@drawable/ic_notifications"
18         android:title="Notifications" />
19     <item
20         android:id="@+id/profileFragment"
21         android:icon="@drawable/ic_profile"
22         android:title="Profile" />
23 </menu>

```

This setup provides a consistent and intuitive user experience, aligning with best practices for Android app navigation.

## 5. User Interaction

User interaction is a key part of the Instagram clone. Click listeners are implemented to enhance interactivity across various pages.

- **Post Uploads:** The "Upload" button on the Add Post page allows users to submit new posts, which are then added to the Home Feed.
- **Profile Navigation:** Clicking on usernames in posts navigates the user to the corresponding Profile page.
- **Likes and Comments:** Users can interact with posts by liking them or accessing the comments section through dedicated buttons, improving engagement and interaction within the app.

## 6. Challenges and Solutions

During the development of the Instagram clone, several challenges arose:

- **RecyclerView Setup:** Initial difficulties occurred when configuring the RecyclerView layouts, especially for the Home Feed and Profile pages. This was resolved by correctly setting up **Adapters** and using **LayoutManagers** (Linear and Grid) for smooth rendering.
- **Navigation Component Conflicts:** Handling fragment transactions sometimes led to back stack issues, causing unexpected behavior during navigation. The solution involved properly managing the **NavController** and using the **popBackStack()** method to ensure consistent navigation.
- **Image Handling:** Loading images efficiently posed a challenge. This was mitigated by integrating an image loading library like **Glide**, ensuring smooth performance.

These solutions ensured the smooth functioning and responsiveness of the app.

## 7. Conclusion

This project provided valuable insights into developing core mobile app features, such as layouts, navigation, and interactive elements, using **Android Studio**. Key takeaways include the importance of efficient data handling with **RecyclerView**, smooth fragment transitions via the **Navigation Component**, and implementing meaningful user interactions.

Moreover, the role of **UI/UX design** became evident as essential for enhancing user engagement and ensuring a seamless experience. Thoughtful interface design significantly impacts the usability and attractiveness of the app, underlining the need to prioritize both **functionality** and **visual appeal** in mobile development.