Mobile Application Development Practical File

A PRACTICAL FILE SUBMITTED IN PARTIAL FULFILLMENT OF COURSE

Mobile Application Development Laboratory

BACHELORS OF TECHNOLOGY

Department of Information Technology

SUBMITTED BY

Jaskirat Singh — 1921151 —2005007

SUBMITTED TO

Prof. Ranjodh Kaur

Session - Aug-Dec 2022

GURU NANAK DEV ENGINEERING COLLEGE

LUDHIANA-141006, INDIA

Contents

List of Figures

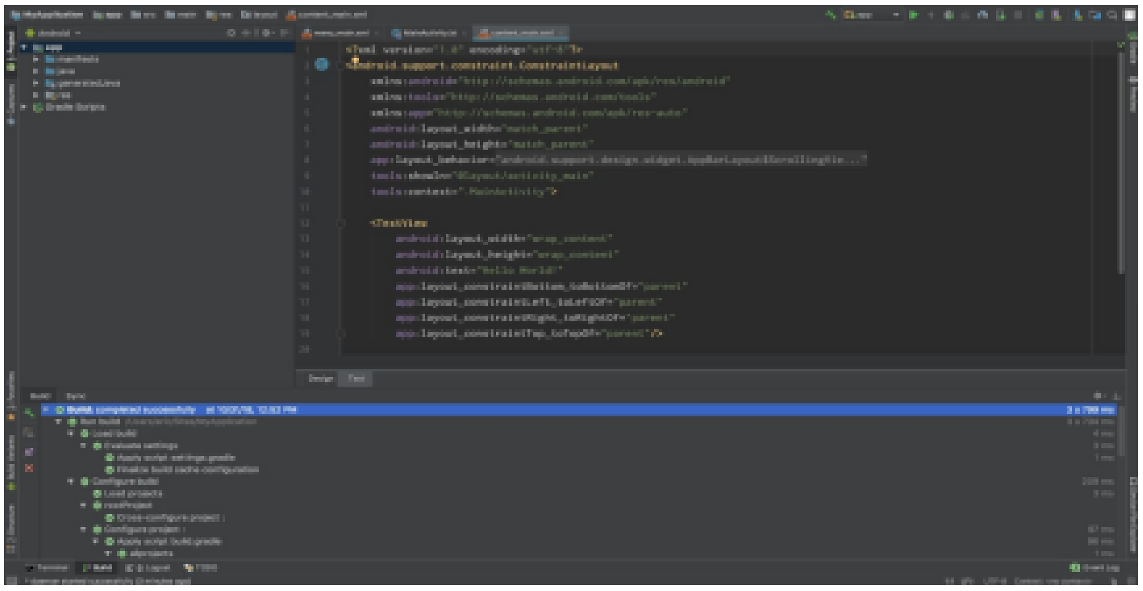
1 To study design aspects of development environment like Android, IOS.

1.0.1 Android

Android's Java environment can be broken down into a handful of key sections. When you understand the contents in each of these sections, the Javadoc reference material that ships with the SDK becomes a real tool and not just a pile of seemingly unrelated material. You might recall that Android isn't a strictly Java ME software environment, but there's some commonality between the Android platforms and other Java development platforms. The next few sections review some of the Java packages (core and optional) in the Android SDK and where you can use them. Android Studio is the official IDE for Android development, and with a single download it includes everything you need to begin developing Android apps as you can see below

- IntelliJ IDE + Android Studio plugin
- Android SDK Tools
- Android Platform-tools
- Android Emulator with an Android system image including Google Play Services

Android Studio provides tools for building apps on every type of Android device. Code editing, debugging, performance tooling, a flexible build system, and an instant build or deploy system are included in Android studio.



```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="android.support.design.widget.AppBarLayout$ScrollingVie..."
    tools:showIn="@layout/activity_main"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>
```

Figure 1.1: Android Studio

1. Command Line Tools:- The Android SDK tools available from the SDK Manager provide additional command-line tools to help you during your Android development. The tools are classified into two groups: SDK tools and platform tools. SDK tools are platform independent and are required no matter which Android platform you are developing on. Platform tools are customized to support the features of the latest Android platform.

2. SDK Tools:- The SDK tools are installed with the SDK starter package and are periodically updated. The SDK tools are required if you are developing Android applications. The most important SDK tools include the Android SDK Manager (Android sdk), the AVD Manager (Android AVD) the emulator (emulator), and the Dalvik Debug Monitor Server (DDMS).

3. Virtual Device Tools:-
    - Android Virtual Device Manager
    - Android Emulator (emulator)
    - mksdcard

4. Development Tools:- Hierarchy Viewer (hierarchyviewer) - Provides a visual representation of the layout's View hierarchy with performance information for each node in the layout, and a magnified view of the display to closely examine the pixels in your layout.

5. SDK Manager:- SDK Manager lets you manage SDK packages, such as installed platforms and system images. sqlite3 - Lets you access the SQLite data files created and used by Android applications.

6. Debugging tools:-
    - Android Monitor
    - adb
    - Dalvik Debug Monitor Server (DDMS)
    - Device Monitor
    - systrace

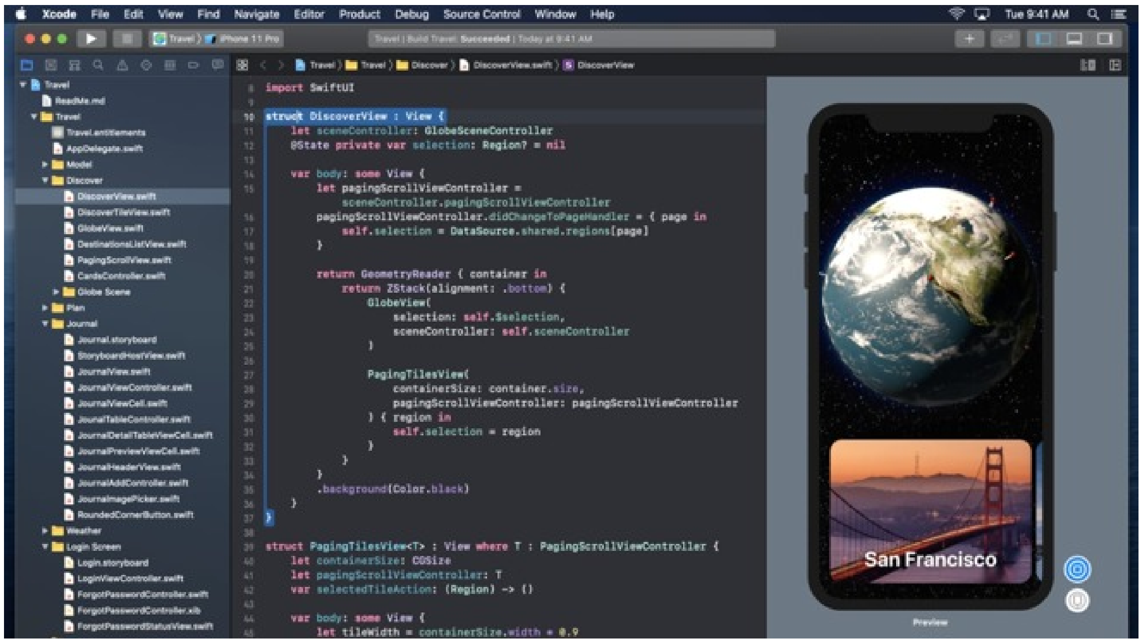
7. Build Tools:-
    - apksigner
    - JOBB
    - ProGuard
    - zipalign

1.0.2 iOS

iOS application development is the process of making mobile applications for Apple hardware, including iPhone, iPad and iPod Touch. The software is written in the Swift programming language or Objective-C and then deployed to the App Store for users to download.

Requirements for iOS development environment:-

- An Apple Mac computer running the latest version of macOS.
- Xcode, which is the integrated development environment (IDE) for macOS, available as a free download from the Mac App Store.
- An active Apple Developer account, which requires a USD 99 annual fee.



```swift
import SwiftUI

struct DiscoverView : View {
    let sceneController: GlobeSceneController
    @State private var selection: Region? = nil

    var body: some View {
        let pagingScrollViewController =
            sceneController.pagingScrollViewController
        pagingScrollViewController.didChangeToPageHandler = { page in
            self.selection = DataSource.shared.regions[page]
        }

        return GeometryReader { container in
            return ZStack(alignment: .bottom) {
                GlobeView(
                    selection: self.$selection,
                    sceneController: self.sceneController
                )

                PagingTilesView(
                    containerSize: container.size,
                    pagingScrollViewController: pagingScrollViewController
                ) { region in
                    self.selection = region
                }
            }
        }
        .background(Color.black)
    }
}

struct PagingTilesView<T> : View where T : PagingScrollViewController {
    let containerSize: CGSize
    let pagingScrollViewController: T
    var selectedTileAction: (Region) -> ()

    var body: some View {
        let tileWidth = containerSize.width * 0.9
```

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Travel | Build Travel: Succeeded | Today at 9:41 AM
San Francisco
Preview

Figure 1.2: xCode iOS IDE

iOS development programming languages:-

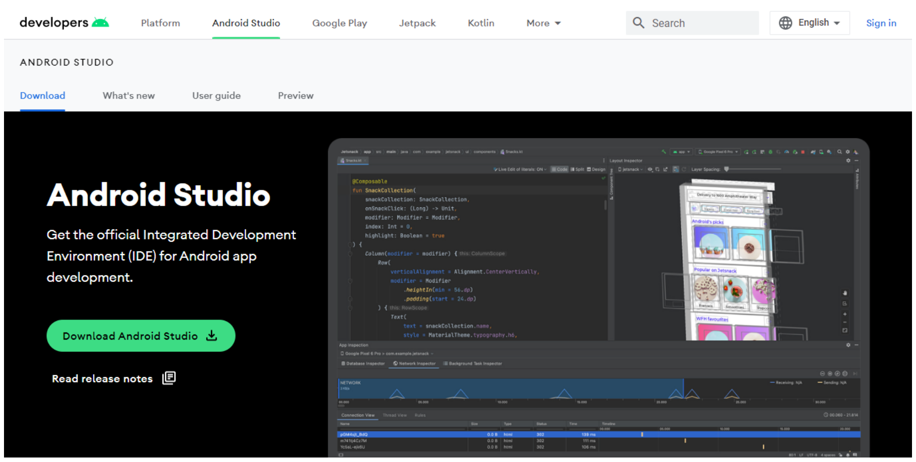
- **Objective-C:** Developed in the early 1980s, Objective-C was the primary programming language for all Apple products for decades. Derived from the C language, Objective-C is an object-oriented programming language centered on passing messages to different processes (as opposed to invoking a process in traditional C programming). Many developers choose to maintain their legacy applications written in Objective-C instead of integrating them into the Swift framework, which was introduced in 2014.
- **Swift:** The Swift programming language is the new "official" language of iOS. While it has many similarities to Objective-C, Swift is designed to use a simpler syntax and is more focused on security than its predecessor. Because it shares a run time with Objective-C, you can easily incorporate legacy code into updated apps. Swift is easy to learn, even for people just beginning to program. Because Swift is faster, more secure and easier to use than Objective-C, you should plan to use it to develop your iOS app unless you have a compelling reason to stick with Objective-C.

One of the major advantages of iOS app development is the extensive collection of developer resources available to you. Because of the standardization, functionality and consistency of iOS app development, Apple is able to release native APIs and libraries as kits that are stable, feature-rich and easy to use. You can use these iOS SDKs to seamlessly integrate your app into Apple's existing infrastructure.

For example, if you're working on an app controller for a smart toaster oven, you can use HomeKit to standardize the communication between the toaster and the phone. There are kits for game development (such as SpriteKit, GameplayKit and ReplayKit), health apps, maps, cameras, as well as Siri, Apple's virtual assistant.

2 Android Development Environment: To setup Android studio2 and study its basic components.

1. Head over to https://developer.android.com/studiodownloads to get the Android Studio executable or zip file.
2. Click on the Download Android Studio Button.

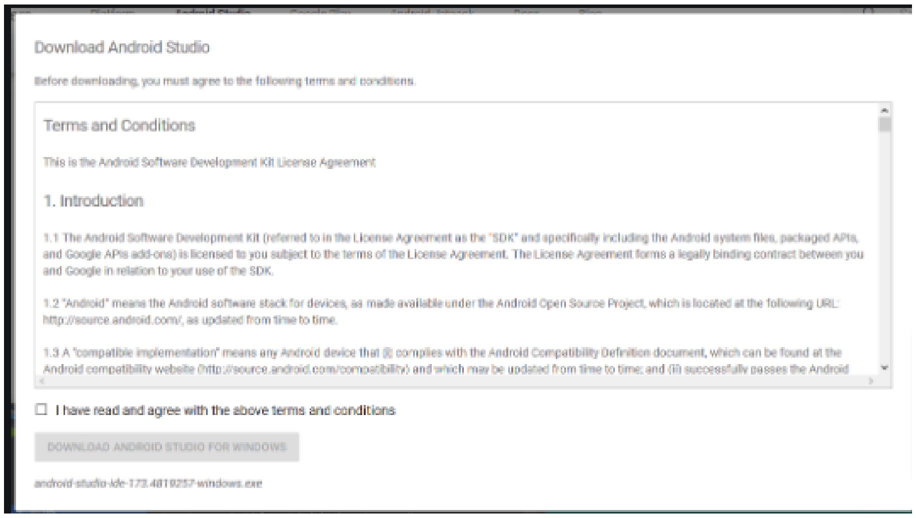


Android Studio

Get the official Integrated Development Environment (IDE) for Android app development.

Figure 2.1: Official Android Studio download page

Click on the "I have read and agree with the above terms and conditions" checkbox followed by the download button.



Download Android Studio

Before downloading, you must agree to the following terms and conditions.

Terms and Conditions

This is the Android Software Development Kit License Agreement

1. Introduction

1.1 The Android Software Development Kit (referred to in the License Agreement as the "SDK" and specifically including the Android system files, packaged APIs, and Google APIs add-ons) is licensed to you subject to the terms of the License Agreement. The License Agreement forms a legally binding contract between you and Google in relation to your use of the SDK.

1.2 "Android" means the Android software stack for devices, as made available under the Android Open Source Project, which is located at the following URL: http://source.android.com/, as updated from time to time.

1.3 A "compatible implementation" means any Android device that (i) complies with the Android Compatibility Definition document, which can be found at the Android compatibility website (http://source.android.com/compatibility) and which may be updated from time to time; and (ii) successfully passes the Android

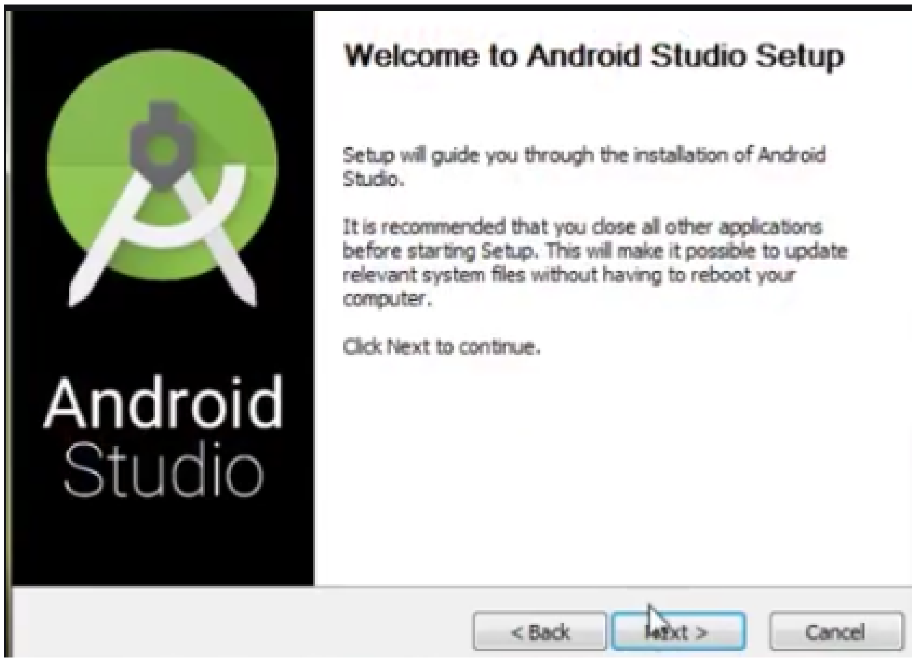
☐ I have read and agree with the above terms and conditions

android-studio-ide-173.4819257-windows.exe

Figure 2.2: Terms and conditions

Click on the Save file button in the appeared prompt box and the file will start downloading.

3. After the downloading has finished, open the file from downloads and run it. It will prompt the following dialog box.



Welcome to Android Studio Setup

Setup will guide you through the installation of Android Studio.

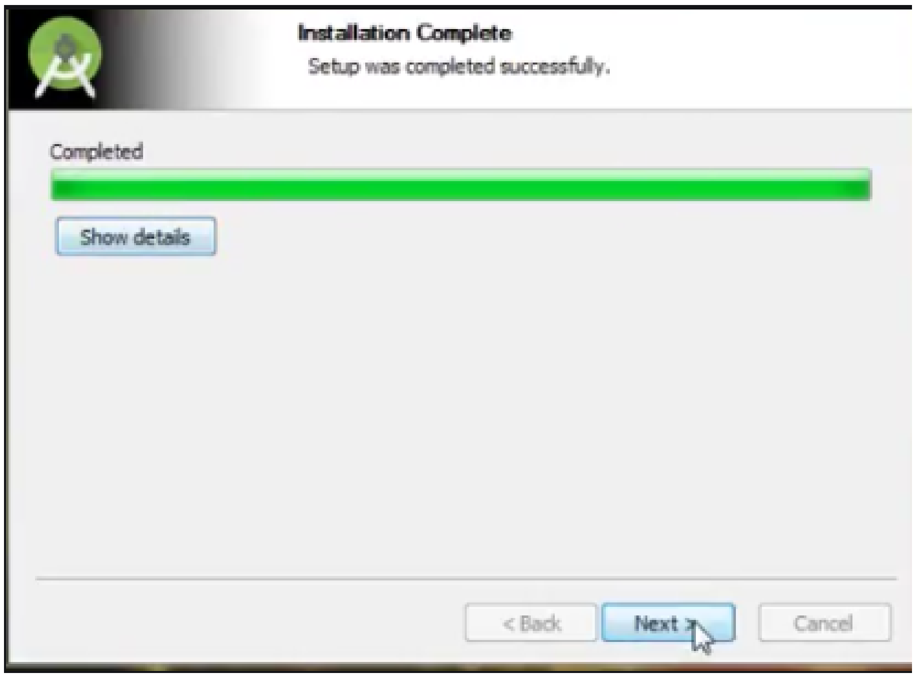
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Figure 2.3: Download completed

Click on next. In the next prompt, it'll ask for a path for installation. Choose a path and hit next.

4. It will start the installation, and once it is completed, it will be like the image shown below.

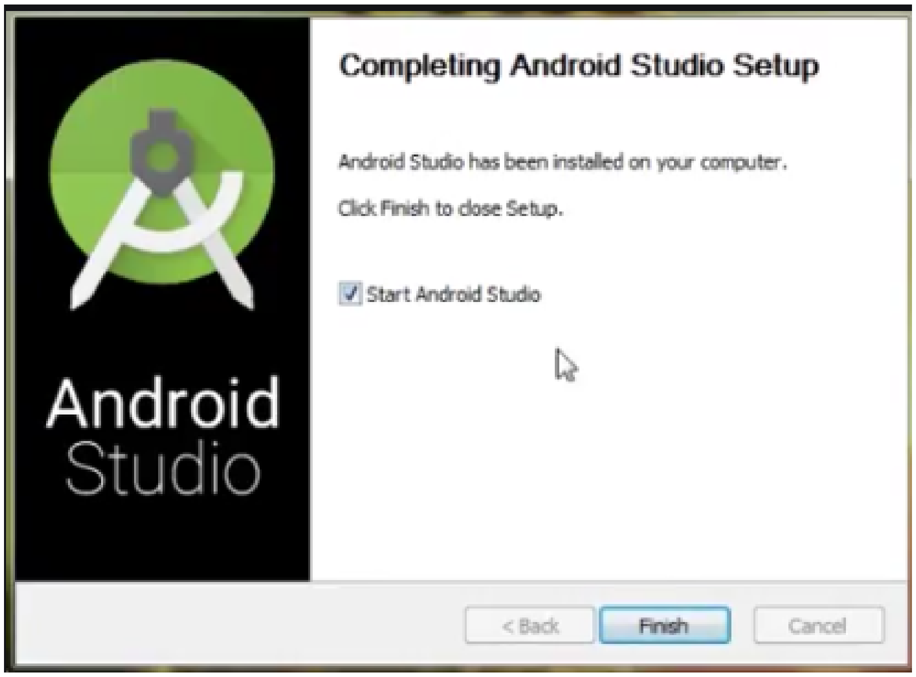


Installation Complete
Setup was completed successfully.

Completed

Figure 2.4: Installation process

Click on next.



Completing Android Studio Setup

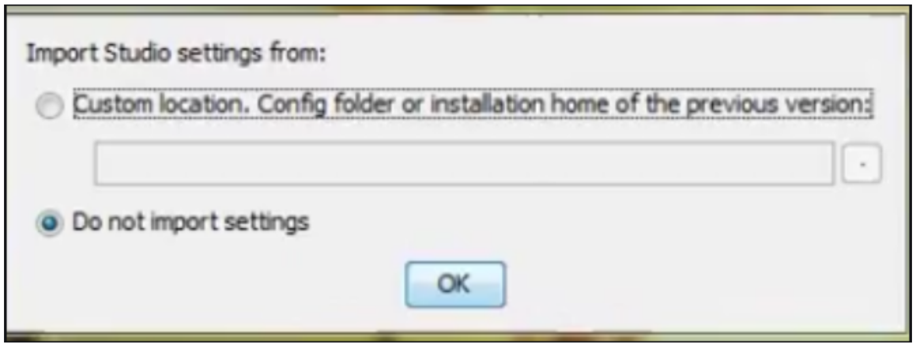
Android Studio has been installed on your computer.

Click Finish to close Setup.

[✓] Start Android Studio

Figure 2.5: Installation process

5. Once “Finish” is clicked, it will ask whether the previous settings need to be imported [if the android studio had been installed earlier], or not. It is better to choose the ‘Don’t import Settings option’.



Import Studio settings from:

○ Custom location. Config folder or installation home of the previous version:

● Do not import settings

Figure 2.6: Installation process

Click on OK.

6. This will start the Android Studio.

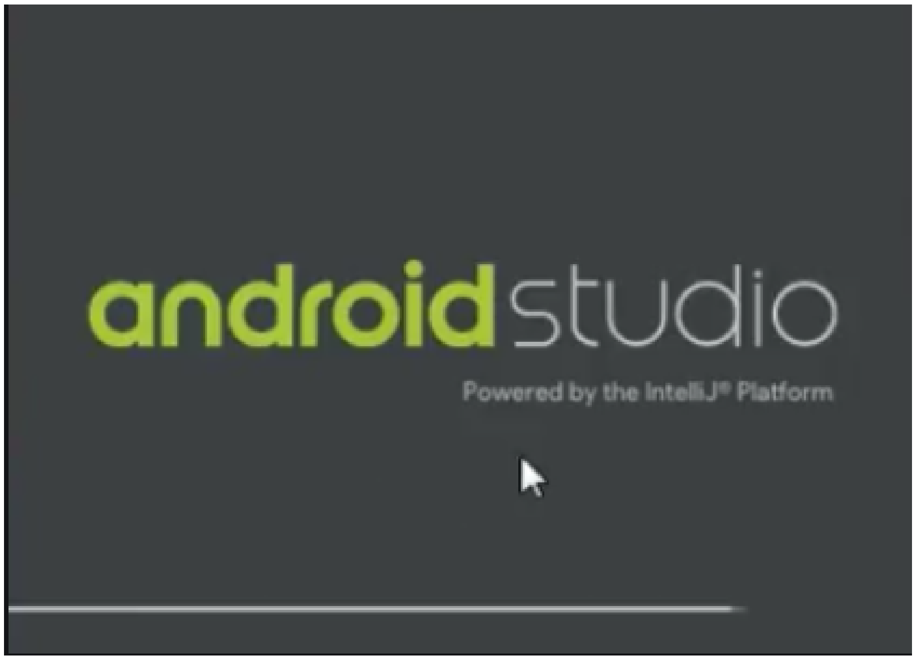
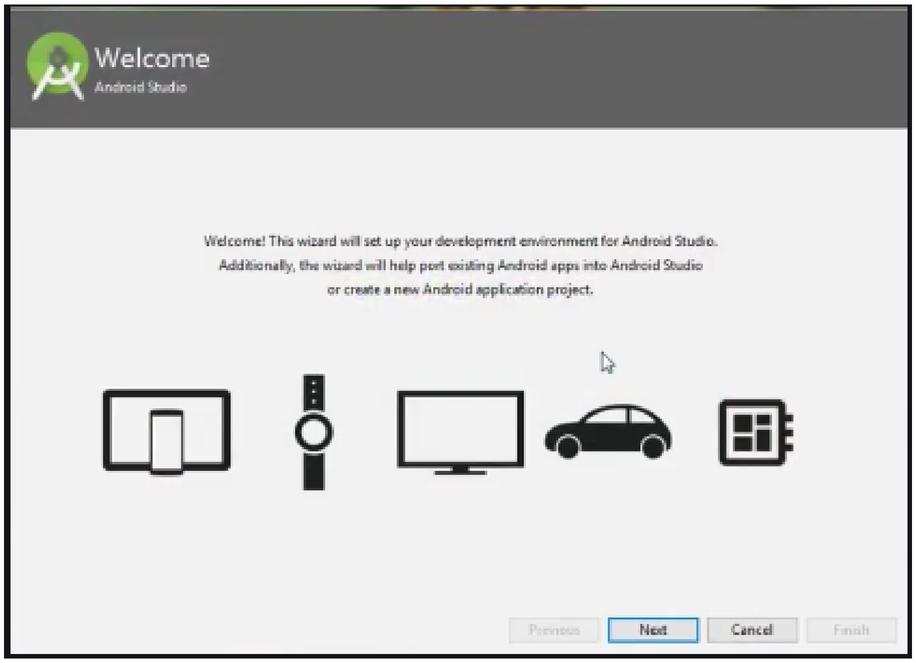


Figure 2.7: Installation completed

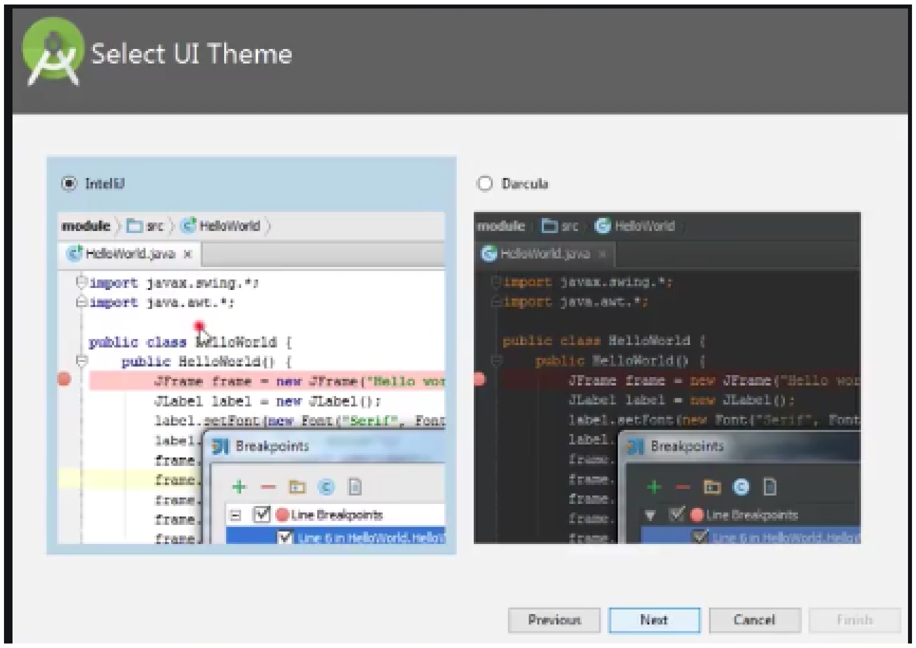
7. After it has found the SDK components, it will redirect to the Welcome dialog box.



Welcome! This wizard will set up your development environment for Android Studio.
Additionally, the wizard will help port existing Android apps into Android Studio
or create a new Android application project.

Figure 2.8: Welcome screen

8. Choose Standard and click on Next. Now choose the theme, whether the Light theme or the Dark one. The light one is called the IntelliJ theme whereas the dark theme is called Dracula. Choose as required. Click on Next.

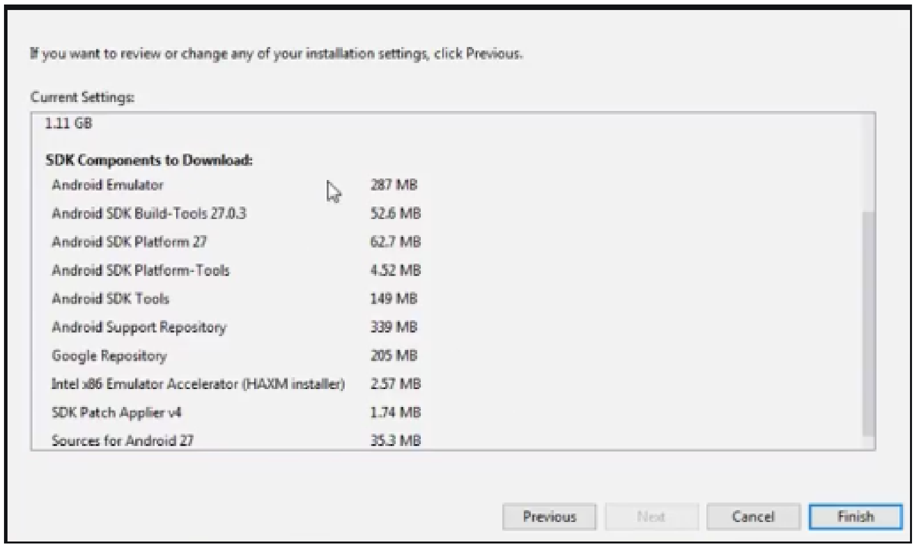


```
import javax.swing.*;
import java.awt.*;

public class HelloWorld {
    public HelloWorld() {
        JFrame frame = new JFrame("Hello wor
        JLabel label = new JLabel();
        label.setFont(new Font("Serif", Font
        label.
        frame.
        frame.
        frame.
        frame.
        frame.
```

Figure 2.9: Theme selection

9. Now it is time to download the SDK components.



If you want to review or change any of your installation settings, click Previous.

Current Settings:

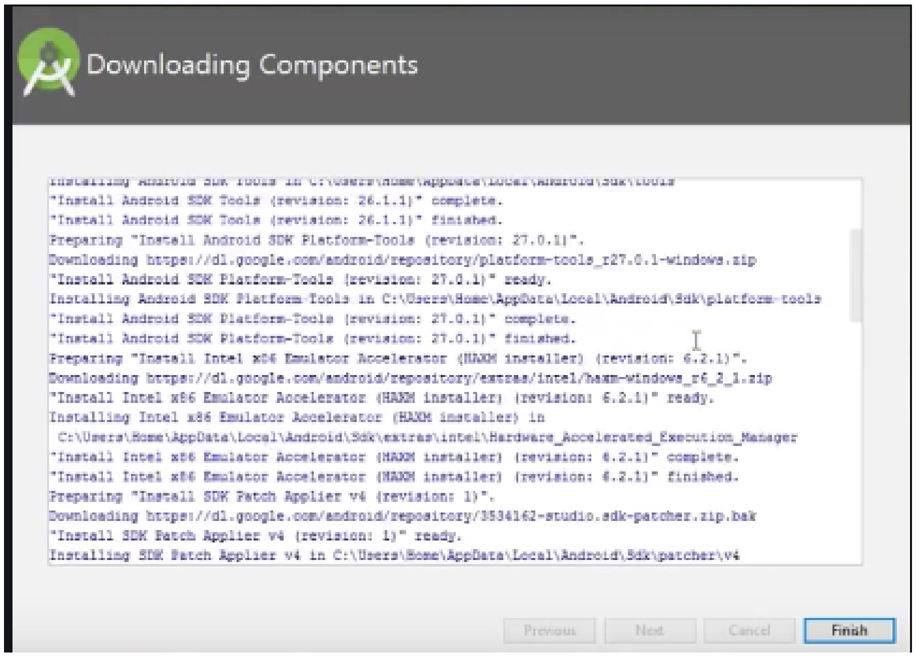
1.11 GB

SDK Components to Download:

Android Emulator	287 MB
Android SDK Build-Tools 27.0.3	52.6 MB
Android SDK Platform 27	62.7 MB
Android SDK Platform-Tools	4.52 MB
Android SDK Tools	149 MB
Android Support Repository	339 MB
Google Repository	205 MB
Intel x86 Emulator Accelerator (HAXM installer)	2.57 MB
SDK Patch Applier v4	1.74 MB
Sources for Android 27	35.3 MB

Figure 2.10: Installation process

Click on Finish. Components begin to download let it complete.



```
Installing Android SDK Tools in C:\Users\Home\AppData\Local\Android\Sdk\tools
"Install Android SDK Tools (revision: 26.1.1)" complete.
"Install Android SDK Tools (revision: 26.1.1)" finished.
Preparing "Install Android SDK Platform-Tools (revision: 27.0.1)".
Downloading https://dl.google.com/android/repository/platform-tools_r27.0.1-windows.zip
"Install Android SDK Platform-Tools (revision: 27.0.1)" ready.
Installing Android SDK Platform-Tools in C:\Users\Home\AppData\Local\Android\Sdk\platform-tools
"Install Android SDK Platform-Tools (revision: 27.0.1)" complete.
"Install Android SDK Platform-Tools (revision: 27.0.1)" finished.
Preparing "Install Intel x86 Emulator Accelerator (HAXM installer) (revision: 6.2.1)".
Downloading https://dl.google.com/android/repository/extras/intel/haxm-windows_r6_2_1.zip
"Install Intel x86 Emulator Accelerator (HAXM installer) (revision: 6.2.1)" ready.
Installing Intel x86 Emulator Accelerator (HAXM installer) in
 C:\Users\Home\AppData\Local\Android\Sdk\extras\intel\Hardware_Accelerated_Execution_Manager
"Install Intel x86 Emulator Accelerator (HAXM installer) (revision: 6.2.1)" complete.
"Install Intel x86 Emulator Accelerator (HAXM installer) (revision: 6.2.1)" finished.
Preparing "Install SDK Patch Applier v4 (revision: 1)".
Downloading https://dl.google.com/android/repository/3534162-studio.sdk-patcher.zip.bak
"Install SDK Patch Applier v4 (revision: 1)" ready.
Installing SDK Patch Applier v4 in C:\Users\Home\AppData\Local\Android\Sdk\patcher\v4
```

Figure 2.11: Installation process

10. Click on Start a new Android Studio project to build a new app.

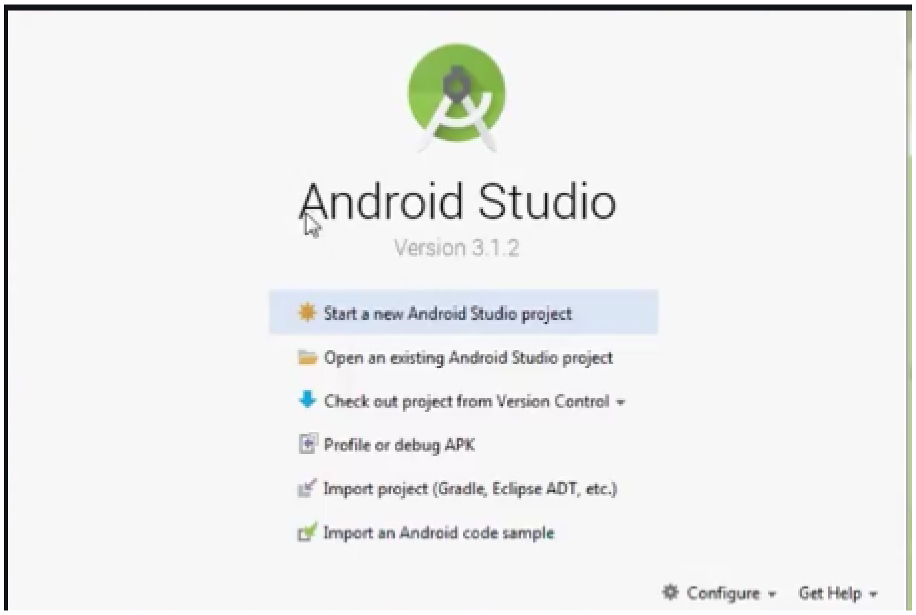

Android Studio
Version 3.1.2
Start a new Android Studio project
Open an existing Android Studio project
Check out project from Version Control
Profile or debug APK
Import project (Gradle, Eclipse ADT, etc.)
Import an Android code sample
Configure
Get Help

Figure 2.12: First Project

3 Android User Interface Design: To study various XML files needed for interface design.

XML stands for Extensible Markup Language. XML is a markup language much like HTML used to describe data. It is derived from Standard Generalized Markup Language(SGML). In Android, the XML is used to implement UI-related data, and it's a lightweight markup language that doesn't make layout heavy. XML only contains tags, while implementing they need to be just invoked. Syntax for XML tag:-

```
<tag_name>Hello World!</tag_name>
```

Different XML files serve different purposes in Android Studio. The list of various XML files in Android Studio with their purposes is discussed below.

1. **Layout XML files in android:** The Layout XML files are responsible for the actual User Interface of the application. It holds all the widgets or views like Buttons, TextViews, EditTexts, etc. which are defined under the ViewGroups. The Location of the layout files in Android is:

   ```
   app -> src -> main -> res -> layout
   ```

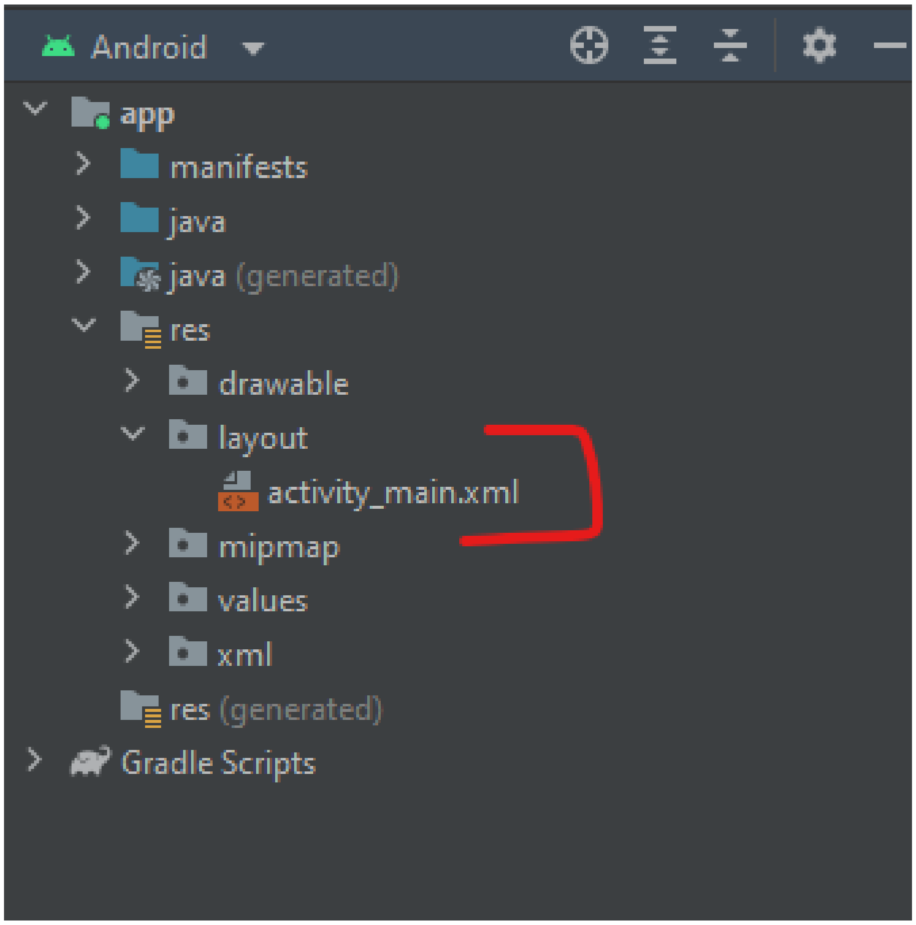


Figure 3.1: Layout XML location

2. **AndroidManifest.xml file:** This file describes the essential information about the application's, like the application's package names which matches code's namespaces, a component of the application like activities, services, broadcast receivers, and content providers. Permission required by the user for the application features also mentioned in this XML file. Location of the AndroidManifest.xml file:

   ```
   app -> src -> main
   ```

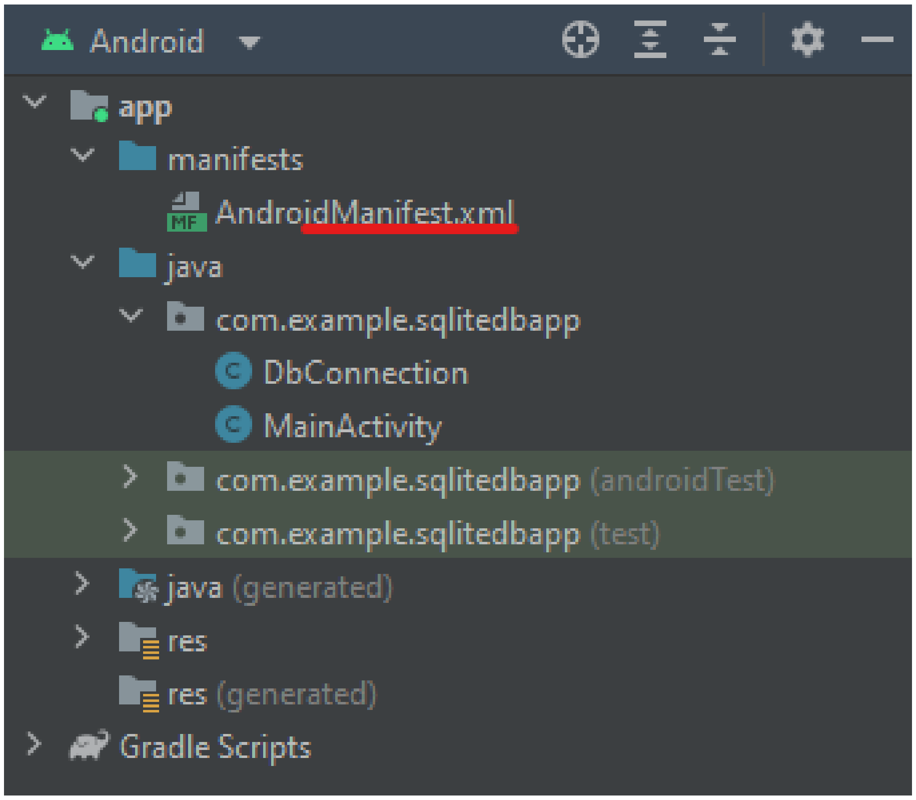


Figure 3.2: Manifest XML location

3. **strings.xml file:** This file contains texts for all the TextViews widgets. This enables reusability of code, and also helps in the localization of the application with different languages. The strings defined in these files can be used to replace all hardcoded text in the entire application. Location of the strings.xml file

   ```
   app -> src -> main -> res -> values
   ```

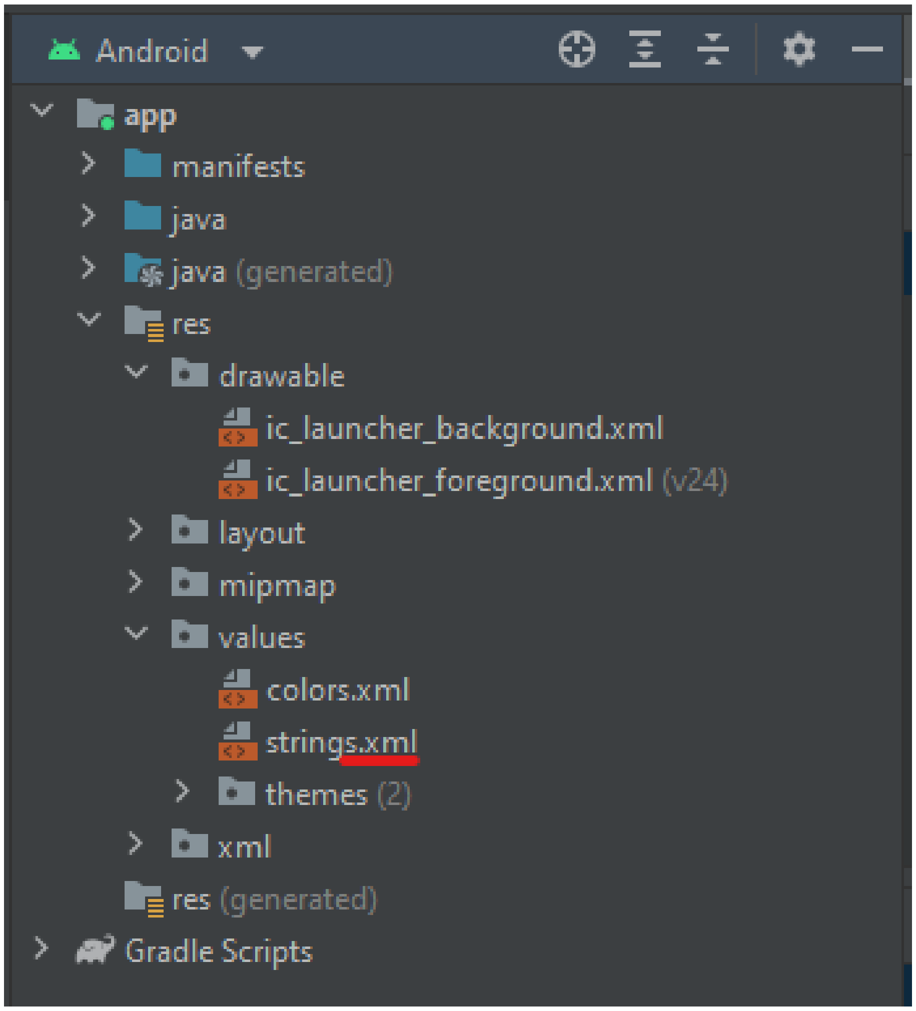
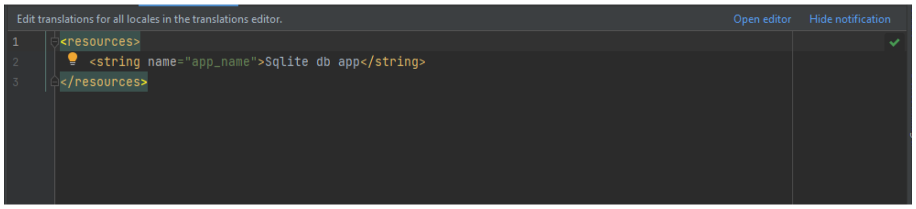


Figure 3.3: String XML location


Edit translations for all locales in the translations editor.
Open editor
Hide notification

```xml
<resources>
    <string name="app_name">Sqlite db app</string>
</resources>
```

Figure 3.4: String XML Code

4. **themes.xml file:** This file defines the base theme and customized themes of the application. It also used to define styles and looks for the UI(User Interface) of the application. By defining styles we can customize how the views or widgets look on the User Interface. Location of styles.xml file.

   `app -> src -> main -> res -> values`

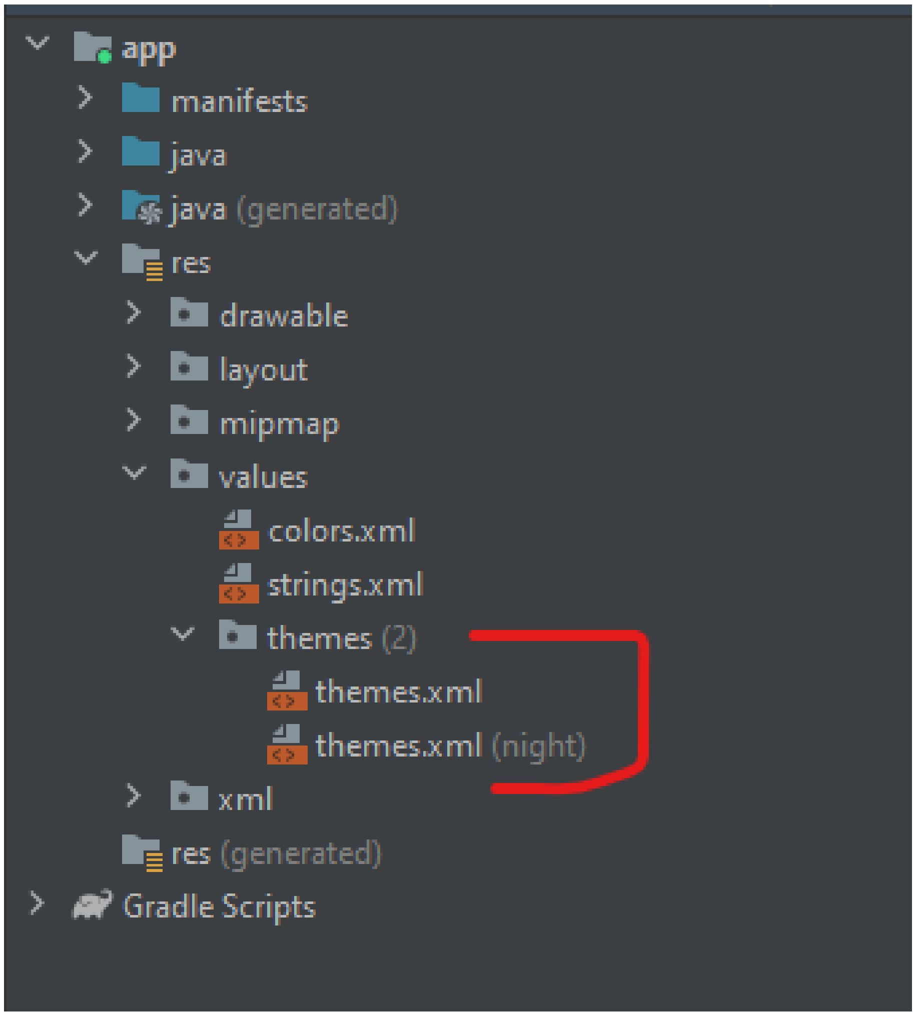
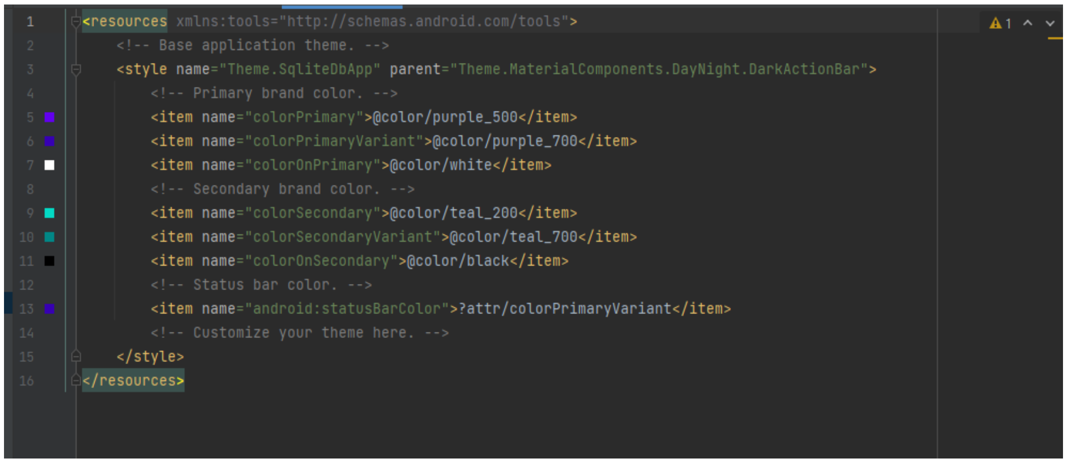


Figure 3.5: Themes XML location



```xml
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.SqliteDbApp" parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
</resources>
```

Figure 3.6: Themes XML code

5. **Drawable XML files:** These are the XML files that provide graphics to elements like custom background for the buttons and its ripple effects, also various gradients can be

created. This also holds the vector graphics like icons. Using these files custom layouts can be constructed for EditTexts. Location for the Drawable files are:

```
app -> src -> main -> res -> drawable
```

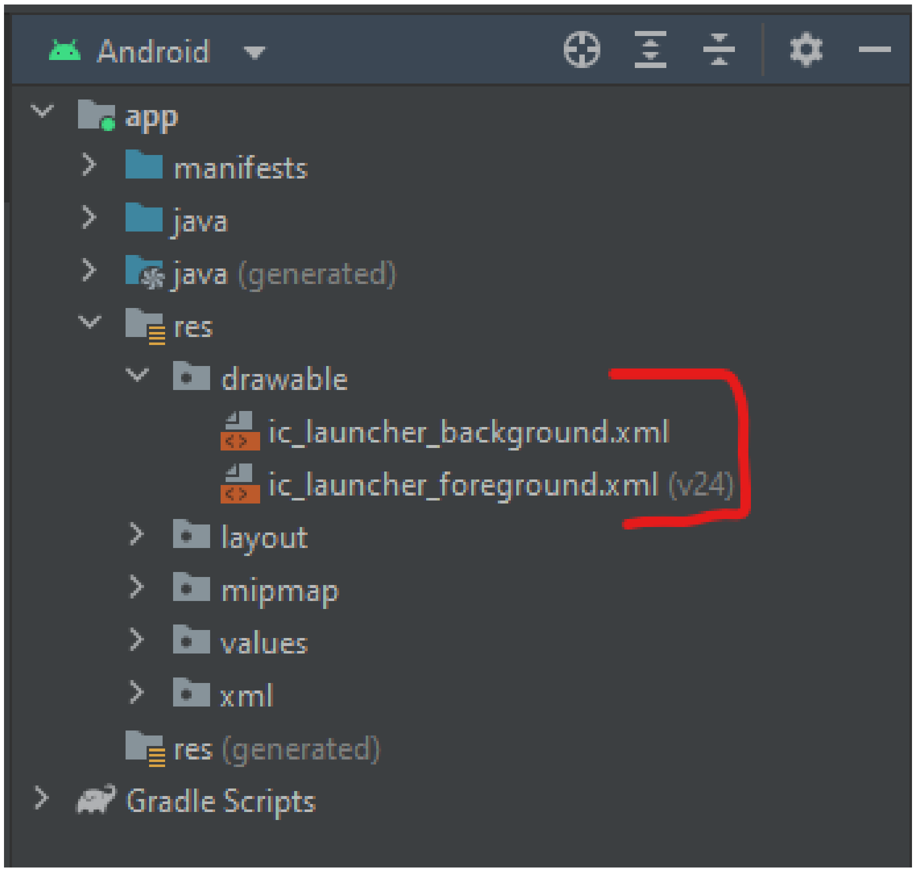
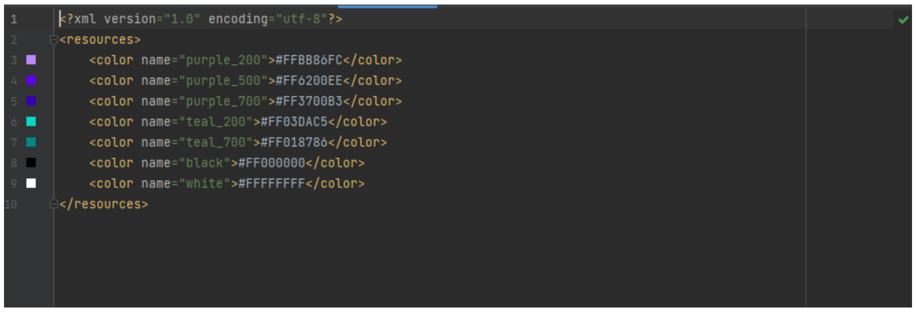


Figure 3.7: Drawable XML location

6. **colors.xml file:** The colors.xml file is responsible to hold all the types of colors required for the application. It may be primary brand color and its variants and secondary brand color and its variants. The colors help uphold the brand of the applications. So the colors need to be decided cautiously as they are responsible for the User Experience. The colors need to be defined in the hex code format. Location of colors.xml file:

```
app -> src -> main -> res -> values
```



```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

Figure 3.8: colors XML code

7. **dimens.xml file:** As the file name itself suggests that the file is responsible to hold the entire dimensions for the views. it may be the height of the Button, padding of the views, the margin for the views, etc. The dimensions need to in the format of pixel density(dp) values. Which replaces all the hard-coded dp values for the views. This file needs to be created separately in the values folder. Location to create dimens.xml file:

   ```
   app -> src -> main -> res -> values
   ```

4 Android User Interface Design: To implement different type of layouts like relative, grid, linear and table.

4.0.1 UI development - XML Code for Relative and Table Layout

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TableLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">


    <ImageView
        android:id="@+id/imageView4"
        android:layout_width="match_parent"
        android:layout_height="40pt"
        android:layout_marginBottom="20pt"
        android:layout_marginTop="20pt"
        app:srcCompat="@drawable/android" />

    <ImageView
        android:id="@+id/imageView5"
        android:layout_width="match_parent"
        android:layout_height="40pt"
        android:layout_marginBottom="20pt"
        app:srcCompat="@drawable/android" />

    <ImageView
        android:id="@+id/imageView6"
        android:layout_width="match_parent"
        android:layout_height="40pt"
        app:srcCompat="@drawable/android" />

        <ImageView
            android:id="@+id/imageView7"
            android:layout_width="match_parent"
            android:layout_height="40pt"
            android:layout_marginBottom="20pt"
            android:layout_marginTop="20pt"
            app:srcCompat="@drawable/android" />

        <ImageView
            android:id="@+id/imageView8"
            android:layout_width="match_parent"
            android:layout_height="40pt"
            android:layout_marginBottom="20pt"
            app:srcCompat="@drawable/android" />

        <ImageView
            android:id="@+id/imageView9"
            android:layout_width="match_parent"
            android:layout_height="40pt"
            app:srcCompat="@drawable/android" />


    </TableLayout>

</ScrollView>

        <RelativeLayout
            android:id="@+id/rl1"
            android:layout_width="65dp"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"

    android:layout_marginRight="60dp"
    android:background="#FF2212">

    <ImageView
        android:id="@+id/imageView4"
        android:layout_width="30dp"
        android:layout_height="match_parent"
        android:layout_marginLeft="12dp"
        android:layout_marginTop="15dp"
        android:foregroundGravity="center_vertical|center_horizontal"
        app:srcCompat="@drawable/star" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center_horizontal"
        android:text="Rating 4.3"
        android:textAlignment="center"
        android:textAllCaps="false"
        android:textColor="#FFF9F9" />
</RelativeLayout>

<RelativeLayout
    android:id="@+id/rl2"
    android:layout_width="65dp"
    android:layout_height="wrap_content"
    android:layout_marginRight="60dp"
    android:background="#FF2212">

    <ImageView
        android:id="@+id/imageView5"
        android:layout_width="40dp"
        android:layout_height="match_parent"
        android:layout_marginLeft="12dp"
        android:layout_marginTop="10dp"
        android:foregroundGravity="center_vertical|center_horizontal"
        app:srcCompat="@drawable/fire" />

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:text="150 Kcal"
        android:textAlignment="center"
        android:textAllCaps="false"
        android:textColor="#FFF9F9" />
</RelativeLayout>

<RelativeLayout
    android:id="@+id/rl3"
    android:layout_width="65dp"
    android:layout_height="wrap_content"
    android:layout_marginRight="60dp"
    android:background="#FF2212">

    <ImageView
        android:id="@+id/imageView6"
        android:layout_width="40dp"
        android:layout_height="match_parent"
        android:layout_marginLeft="12dp"
        android:layout_marginTop="10dp"
        android:foregroundGravity="center_vertical|center_horizontal"
        app:srcCompat="@drawable/timer" />
```

```
            <TextView
                android:id="@+id/textView5"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="5dp"
                android:text="5-10 Min"
                android:textAlignment="center"
                android:textAllCaps="false"
                android:textColor="#FFF9F9" />
        </RelativeLayout>
    </LinearLayout>

    <TextView
        android:id="@+id/textView"
        android:layout_width="335dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="20dp"
        android:layout_marginRight="20dp"
        android:text="This paneer patty burger uses 100% quality cottage cheese with sliced tomatoes ,
             cucumbers, vegetables and onions ...
" />

    <Button
        android:id="@+id/button3"
        android:layout_width="341dp"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="20dp"
        android:backgroundTint="#FB2212"

        android:text="Add to Cart"
        app:cornerRadius="20dp" />

</LinearLayout>

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    <TableLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:background="#8BC34A">


        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:background="#8BC34A">

            <ImageButton
                android:id="@+id/imageButton"
                android:layout_width="90pt"
                android:layout_height="80pt"
                android:backgroundTint="@color/white"
                app:srcCompat="@drawable/androidlogo" />

            <ImageButton
                android:id="@+id/imageButton1"
                android:layout_width="85pt"
                android:layout_height="80pt"
                android:backgroundTint="@color/white"
                app:srcCompat="@drawable/androidlogo" />

        </TableRow>
        <TableRow>

            <ImageButton
                android:id="@+id/imageButton2"
                android:layout_width="90pt"
                android:layout_height="80pt"
                android:backgroundTint="@color/white"
                app:srcCompat="@drawable/androidlogo" />

            <ImageButton
                android:id="@+id/imageButton3"
                android:layout_width="85pt"
                android:layout_height="80pt"
                android:backgroundTint="@color/white"
                app:srcCompat="@drawable/androidlogo" />

        </TableRow>

        <TableRow>

            <ImageButton
                android:id="@+id/imageButton4"
                android:layout_width="90pt"
                android:layout_height="80pt"
                android:backgroundTint="@color/white"
                app:srcCompat="@drawable/androidlogo" />

            <ImageButton
                android:id="@+id/imageButton5"
                android:layout_width="85pt"
                android:layout_height="80pt"
                android:backgroundTint="@color/white"
                app:srcCompat="@drawable/androidlogo" />

        </TableRow>


    </TableLayout>
```

4.0.2 Java Code

```
package com.example.multi_layout;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

4.0.3 Output Screen

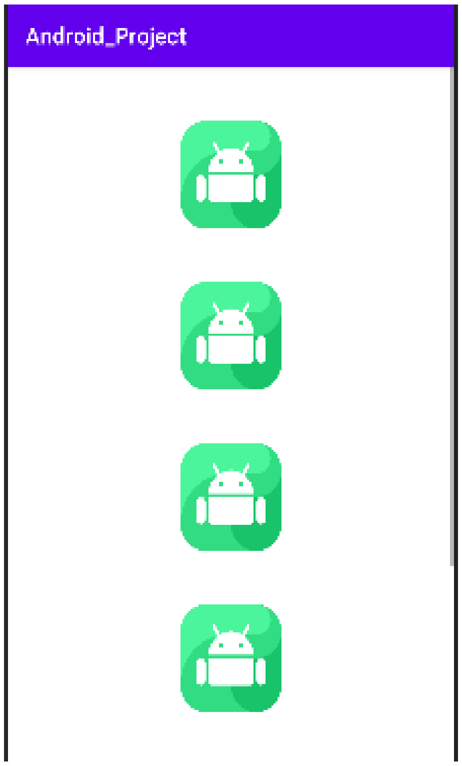


Figure 4.1: Relative layout example

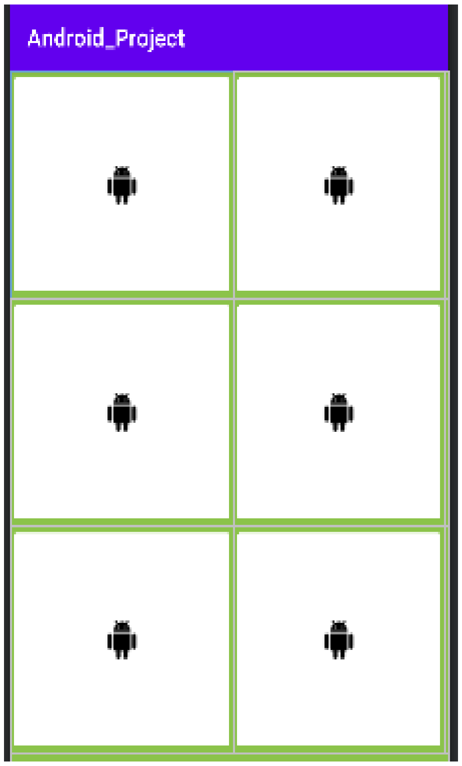


Figure 4.2: Table layout example

5 Apps Interactivity in Android: To incorporate element of interactivity using Android Fragment and Intent Class

5.0.1 UI development (XML files)

1. **activity_main4.xml:**

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    tools:context=".MainActivity4">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#050505"
        android:orientation="horizontal">

        <Button
            android:id="@+id/btn1"
            android:layout_width="108dp"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:layout_weight="1"
            android:backgroundTint="#FBC8C8"
            android:text="Songs"
            android:textColor="#9C27B0" />

        <Button
            android:id="@+id/btn2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:layout_weight="1"
            android:backgroundTint="#FBC8C8"
            android:text="Playlist"
            android:textColor="#9C27B0" />

        <Button
            android:id="@+id/btn3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:layout_weight="1"
            android:backgroundTint="#FBC8C8"
            android:text="Artists"
            android:textColor="#9C27B0" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="685dp"
        android:orientation="vertical">

        <androidx.fragment.app.FragmentContainerView
            android:id="@+id/fragmentContainerView"
            android:name="com.example.spotifyform.Fragment1"
            android:layout_width="match_parent"
            android:layout_height="match_parent" />
    </LinearLayout>
```

```
</LinearLayout>
```

2. fragment_1.xml:-

```
    <?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ff1324"
    tools:context=".Fragment1">


    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="center"
        android:gravity="center"
        android:textColor="#ffff"
        android:text="Fragment 1" />

</FrameLayout>
```

3. fragment_2.xml:-

```
   <?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#3F51B5"
    tools:context=".Fragment2">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="center"
        android:gravity="center"
        android:text="Fragment 2"
        android:textColor="#ffff" />
</FrameLayout>
```

4. fragment_3.xml

```
   <?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FFC107"
    tools:context=".Fragment3">

    <TextView
        android:id="@+id/textView4"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="center"
        android:gravity="center"
        android:text="Fragment 3"
        android:textColor="#ffff" />
```

```
</FrameLayout>
```

5.0.2 Java Code

1. MainActivity4.java

```java
package com.example.spotifyform;

import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.FragmentManager;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity4 extends AppCompatActivity {
    Button btn1, btn2, btn3;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main4);

        //Fragment Manager to manage code
        //FragmentManager fragmentManager = getSupportFragmentManager();

        btn1=findViewById(R.id.btn1);
        btn2=findViewById(R.id.btn2);
        btn3=findViewById(R.id.btn3);

        ///Code for Songs Button to Switch Fragments(Fragment1)
        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                FragmentManager fragmentManager = getSupportFragmentManager();
                fragmentManager.beginTransaction()
                        .replace(R.id.fragmentContainerView,Fragment1.class, null)
                        .setReorderingAllowed(true).addToBackStack("name")
                        .commit();
            }
        });
        ///Code for Playlist Button to Switch Fragments(Fragment2)
        btn2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                FragmentManager fragmentManager = getSupportFragmentManager();
                fragmentManager.beginTransaction()
                        .replace(R.id.fragmentContainerView,Fragment2.class, null)
                        .setReorderingAllowed(true).addToBackStack("name")
                        .commit();
            }
        });
        ///Code for Artists Button to Switch Fragments(Fragment3)
        btn3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                FragmentManager fragmentManager = getSupportFragmentManager();
                fragmentManager.beginTransaction()
                        .replace(R.id.fragmentContainerView,Fragment3.class, null)
                        .setReorderingAllowed(true).addToBackStack("name")
                        .commit();
            }
        });
    }
}
```

2. Fragment1.java:

```
package com.example.spotifyform;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.fragment.app.Fragment;


public class Fragment1 extends Fragment {



    public Fragment1() {
        // Required empty public constructor
    }


    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_1, container, false);
    }
}
```

3. Fragment2.java:

```
package com.example.spotifyform;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.fragment.app.Fragment;


public class Fragment2 extends Fragment {


    public Fragment2() {
        // Required empty public constructor
    }




    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_2, container, false);
    }
}
```

4. Fragment3.java:

```java
package com.example.spotifyform;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.fragment.app.Fragment;


public class Fragment3 extends Fragment {


    public Fragment3() {
        // Required empty public constructor
    }


    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_3, container, false);
    }
}
```

5.0.3 Output Screens:-

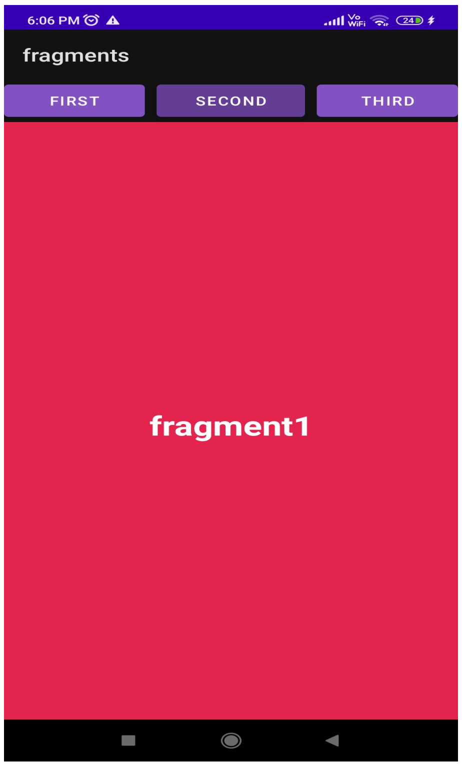


Figure 5.1: Fragment 1

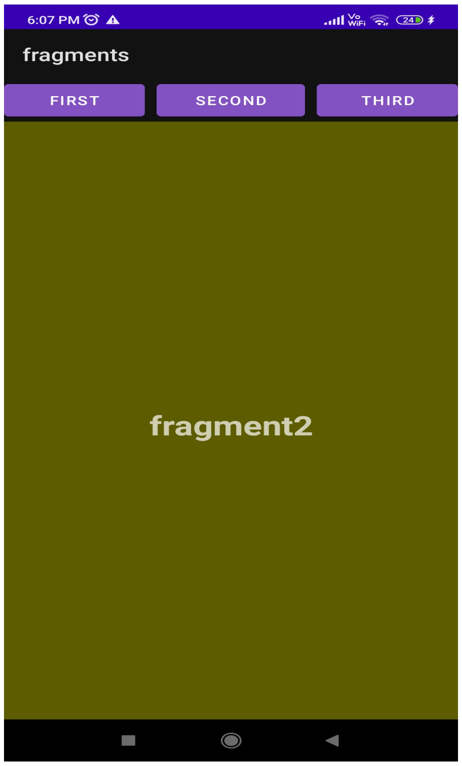


Figure 5.2: Fragment 2

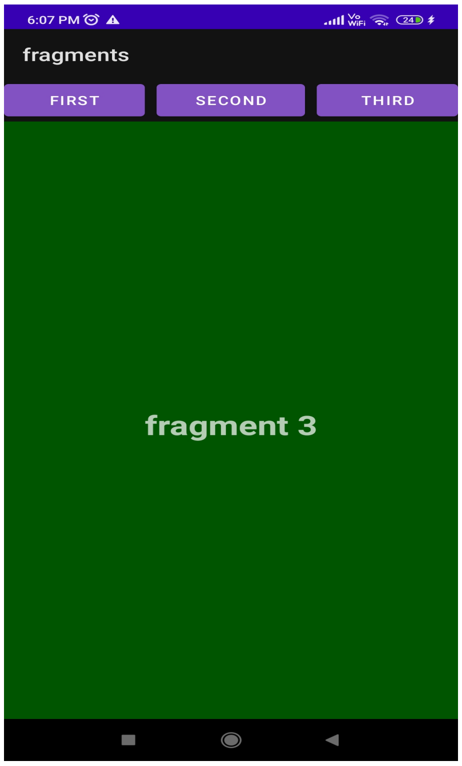


Figure 5.3: Fragment 3

6 Persistent Data Storage: To perform database connectivity of android app using SQLite.

Login and Signup form is created using SQLite database.

6.0.1 UI Development (XML code)

1. SignUp and signIn form:

```
    <?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:orientation="vertical"
    android:layout_height="match_parent"
    android:padding="10pt">


<!-- <ImageView-->
<!-- android:id="@+id/imageView2"-->
<!-- android:layout_width="80dp"-->
<!-- android:layout_height="100dp"-->
<!-- android:layout_gravity="center"-->
<!-- app:srcCompat="@drawable/img" />-->

    <ImageView
        android:id="@+id/imageView3"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:layout_gravity="center"
        app:srcCompat="@drawable/img" />

    <Button
        android:id="@+id/button2"
        android:layout_width="300dp"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:backgroundTint="#5768C5"
        android:text="SIGN UP WITH FACEBOOK"
        app:cornerRadius="35pt" />


        <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Or" />

        <TextView
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
            android:layout_marginBottom="15pt"
        android:textSize="18dp"
        android:text="Sign up with your email address"
        android:textStyle="bold"/>

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical"
            android:paddingTop="10pt">


    <EditText
        android:id="@+id/editTextTextPersonName"
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_gravity="center"
        android:layout_marginBottom="20px"
        android:inputType="textPersonName"
        android:hint="Email" />

    <EditText
        android:id="@+id/editTextTextPersonName2"
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_marginBottom="20px"
        android:layout_gravity="center"
        android:inputType="textPersonName"
        android:hint="Confirm email" />

    <EditText
        android:id="@+id/editTextTextPassword"
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_gravity="center"
        android:layout_marginBottom="20px"
        android:inputType="textPersonName"
        android:hint="Password" />


    <EditText
        android:id="@+id/editTextTextUserName"
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_gravity="center"
        android:layout_marginBottom="20px"
        android:inputType="textPersonName"
        android:hint="What should we call you?" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="350dp"
        android:layout_gravity="center"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10pt"
        android:text="Date of Birth:" />

<!-- <EditText-->
<!-- android:id="@+id/editTextDate"-->
<!-- android:layout_width="300dp"-->
<!-- android:layout_gravity="center"-->
<!-- android:layout_height="wrap_content"-->
<!-- android:ems="10"-->
<!-- android:inputType="date" />-->


    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="49dp"
        android:orientation="horizontal"
        android:padding="1pt">
    <!-- <TableRow>-->

    <Spinner
        android:id="@+id/spinner"
        android:layout_width="180dp"
        android:layout_height="44dp"
        android:layout_marginLeft="10pt"
        android:hint="Month" />

    <EditText
        android:id="@+id/editTextText"
        android:layout_width="75dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Date"
        android:inputType="textPersonName" />

    <EditText
        android:id="@+id/editTextYear"
        android:layout_width="75dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Year"
        android:inputType="textPersonName" />


    <!-- </TableRow>-->

</LinearLayout>


<TableLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_marginTop="3pt">

    <TableRow>

    <RadioButton
        android:id="@+id/radioButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:layout_marginRight="5pt"
        android:text="Male" />
    <RadioButton
        android:id="@+id/radioButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:layout_marginRight="5pt"
        android:text="Female" />
    <RadioButton
        android:id="@+id/radioButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Others" />

    </TableRow>
</TableLayout>

        <CheckBox
            android:id="@+id/checkBox"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
```

```
                android:textSize="6pt"
                android:layout_marginBottom="8pt"
                android:text="Share my registration data with Spotify's content providers for marketing
                    purposes." />

            <CheckBox
                android:id="@+id/checkBox1"
                android:layout_width="500px"
                android:layout_gravity="center"
                android:layout_height="wrap_content"
                android:text="I'm no robot" />
<!-- android:textSize="6pt"-->

    <TextView
        android:id="@+id/textView3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="4pt"
        android:textColor="@color/black"
        android:text="By clicking on Sign Up, you agree to Spotify's Term and Conditions of Use." />


    <TextView
        android:id="@+id/textView4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="4pt"
        android:textColor="@color/black"
        android:text="To learn more about ho Spotify collects, uses, shares and protects your personal
            data please read Spotify's Privacy Please." />

            <Button
                android:layout_width="600px"
                android:layout_height="match_parent"
                android:padding="4pt"
                android:layout_gravity="center"
                android:backgroundTint="#37C63F"
                android:text="Sign Up"
                android:textColor="@color/white"
                android:textSize="8pt"
                app:cornerRadius="10pt" />
        </LinearLayout>

    </ScrollView>

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:srcCompat="@drawable/img" />


    <!-- <ScrollView-->
<!-- android:layout_width="match_parent"-->
<!-- android:layout_height="match_parent">-->

<!-- <LinearLayout-->
<!-- android:layout_width="match_parent"-->
<!-- android:layout_height="wrap_content"-->
<!-- android:orientation="vertical">-->


<!-- -->
<!-- </LinearLayout>-->
<!-- </ScrollView>-->
</LinearLayout>
```

2. CRUD form: activity_form.xml

```
<?xml version="1.0" encoding="utf-8"?>

 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:orientation="vertical"
    android:layout_height="match_parent"
    android:padding="10pt">


    <EditText
        android:id="@+id/editTextTextPersonName"
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_gravity="center"
        android:layout_marginBottom="20px"
        android:inputType="textPersonName"
        android:hint="Email" />

    <EditText
        android:id="@+id/editTextTextPersonName2"
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_marginBottom="20px"
        android:layout_gravity="center"
        android:inputType="textPersonName"
        android:hint="Confirm email" />

    <EditText
        android:id="@+id/editTextTextPassword"
        android:layout_width="350dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_gravity="center"
        android:layout_marginBottom="20px"
        android:inputType="textPersonName"
        android:hint="Password" />

 <Button
    android:id="@+id/Inbtn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Insert" />
 <Button
    android:id="@+id/delbtn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Delete" />
 <Button
    android:id="@+id/upbtn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Update" />
 <Button
    android:id="@+id/vwbtn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="View" />


 <!-- <EditText-->
<!-- android:id="@+id/editTextTextUserName"-->
<!-- android:layout_width="350dp"-->
<!-- android:layout_height="wrap_content"-->
```

```
<!-- android:ems="10"-->
<!-- android:layout_gravity="center"-->
<!-- android:layout_marginBottom="20px"-->
<!-- android:inputType="textPersonName"-->
<!-- android:hint="What should we call you?" />-->

<!-- <TextView-->
<!-- android:id="@+id/textView2"-->
<!-- android:layout_width="350dp"-->
<!-- android:layout_gravity="center"-->
<!-- android:layout_height="wrap_content"-->
<!-- android:layout_marginBottom="10pt"-->
<!-- android:text="Date of Birth:" />-->

<!-- <Spinner-->
<!-- android:id="@+id/spinner"-->
<!-- android:layout_width="180dp"-->
<!-- android:layout_height="44dp"-->
<!-- android:layout_marginLeft="10pt"-->
<!-- android:hint="Month" />-->

<!-- <EditText-->
<!-- android:id="@+id/editTextText"-->
<!-- android:layout_width="75dp"-->
<!-- android:layout_height="wrap_content"-->
<!-- android:ems="10"-->
<!-- android:hint="Date"-->
<!-- android:inputType="textPersonName" />-->

<!-- <EditText-->
<!-- android:id="@+id/editTextYear"-->
<!-- android:layout_width="75dp"-->
<!-- android:layout_height="wrap_content"-->
<!-- android:ems="10"-->
<!-- android:hint="Year"-->
<!-- android:inputType="textPersonName" />-->
    </LinearLayout>
```

6.0.2 Java Code

1. MainActivity.java

```
package com.example.ui_practice;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_intro);
        ImageView next;
        next=findViewById(R.id.next);

        next.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent iNext= new Intent(MainActivity.this, Login.class);
                startActivity(iNext);
```

```java
                finish();
            }
        });
    }
}
```

2. SignUp.java

```java
package com.example.ui_practice;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class SignUp extends AppCompatActivity {
    private EditText name, email, password;
    private Button register;
    private FirebaseAuth auth;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sign_up);

        name = findViewById(R.id.name);
        email = findViewById(R.id.email);
        password = findViewById(R.id.password);
        register = findViewById(R.id.register_btn);
        auth = FirebaseAuth.getInstance();

        register.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String email_txt = email.getText().toString();
                String password_txt = password.getText().toString();
                String name_txt = name.getText().toString();
                if (TextUtils.isEmpty(email_txt) || TextUtils.isEmpty(password_txt) || TextUtils.isEmpty(
                     name_txt))
                    Toast.makeText(SignUp.this, "Empty Credentials!", Toast.LENGTH_SHORT).show();
                else if (password_txt.length() < 6)
                    Toast.makeText(SignUp.this, "Password Too short!", Toast.LENGTH_SHORT).show();
                else {
                    registerUser(email_txt, password_txt);
                }

            }
        });
    }

        private void registerUser(String email, String password) {
            auth.createUserWithEmailAndPassword(email, password).
                    addOnCompleteListener(SignUp.this, new OnCompleteListener<AuthResult>() {
                        @Override
                        public void onComplete(@NonNull Task<AuthResult> task) {
                            if(task.isSuccessful()){
                                Toast.makeText(SignUp.this, "Registration Completed!", Toast.LENGTH_SHORT).
                                     show();
                                startActivity(new Intent(SignUp.this, Login.class));
                                finish();
                            }

                            else
                                Toast.makeText(SignUp.this, "Registration Failed!", Toast.LENGTH_SHORT).
                                     show();
                        }
                    });
        }




}
```

3. Login.java

```
package com.example.ui_practice;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class Login extends AppCompatActivity {
    private TextView register;
    private EditText login_email, login_password;
    private Button login_btn;
    private FirebaseAuth auth;

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

        //getting all ids from R class
        register=findViewById(R.id.register_link);
        login_email=findViewById(R.id.login_email);
        login_password=findViewById(R.id.login_pass);
        login_btn=findViewById(R.id.login_btn);
        auth= FirebaseAuth.getInstance();

        register.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent iRegister=new Intent(Login.this, SignUp.class);
                startActivity(iRegister);
                finish();
            }
        });

        login_btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String email=login_email.getText().toString();
                String password=login_password.getText().toString();

                if(TextUtils.isEmpty(email) )
                    login_email.setError("Please provide correct email");
                if(TextUtils.isEmpty(password) )
                    login_password.setError("Please provide password");

                loginUser(email, password);


            }
        });
    }

    private void loginUser(String email, String password) {
        auth.signInWithEmailAndPassword(email, password).addOnSuccessListener(new OnSuccessListener<
             AuthResult>() {
            @Override
            public void onSuccess(AuthResult authResult) {
                Toast.makeText(Login.this, "Login Successful", Toast.LENGTH_SHORT).show();
                startActivity(new Intent(Login.this, EndActivity.class));
                finish();
            }
        });
    }
}
```

6.0.3 Output Screens:

1. Register and Login Form:-

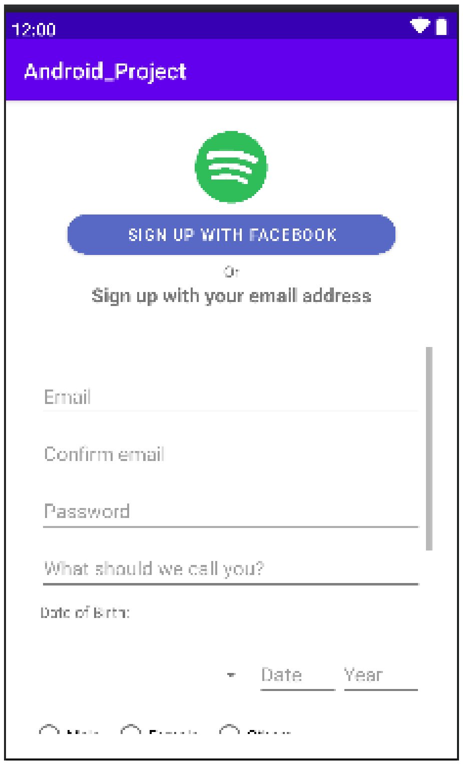


Figure 6.1: Login and Registration form

2. Data Insert, Update, Delete form

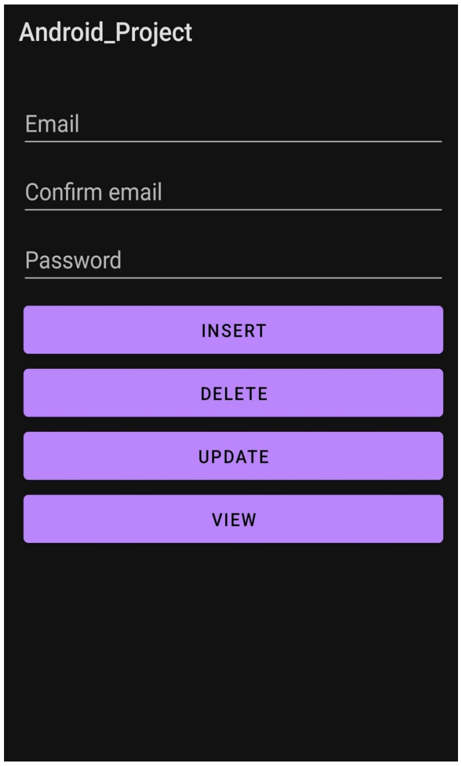


Figure 6.2: CRUD Form

7 Android Services and Threads: To implement the concept of multithreading using Android Service class

7.0.1 XML Code

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="250dp"
        android:layout_height="250dp"
        android:layout_margin="50dp"
        android:layout_gravity="center"/>

    <Button
        android:layout_marginTop="60pt"
        android:id="@+id/b1"
        android:layout_width="70pt"
        android:layout_height="25pt"
        android:layout_margin="10dp"
        android:layout_gravity="center"
        android:text="Show Image 1"/>

    <Button
        android:id="@+id/b2"
        android:layout_width="70pt"
        android:layout_height="25pt"
        android:layout_margin="10dp"
        android:layout_gravity="center"
        android:text="Show Image 2"/>



</LinearLayout>
```

7.0.2 Java Code

```java
package com.example.calculator_app;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity5 extends AppCompatActivity
{
    ImageView img;
    Button bt1,bt2;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.multithread);
        bt1 = (Button)findViewById(R.id.b1);
        bt2= (Button) findViewById(R.id.b2);
        img = (ImageView)findViewById(R.id.imageView);

        bt1.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v)
            {
                new Thread(new Runnable()
                {
                    @Override
                    public void run()
                    {
                        img.post(new Runnable()
                        {
                            @Override
                            public void run()
                            {
                                img.setImageResource(R.drawable.calculator);
                            }
                        });
                    }
                }).start();
            }
        });

        bt2.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v)
            {
                new Thread(new Runnable()
                {
                    @Override
                    public void run()
                    {
                        img.post(new Runnable()
                        {
                            @Override
                            public void run()
                            {
                                img.setImageResource(R.drawable.android);
                            }
                        });
                    }
                }).start();
            }
        });
    }
}
```

7.0.3 Output Screens:

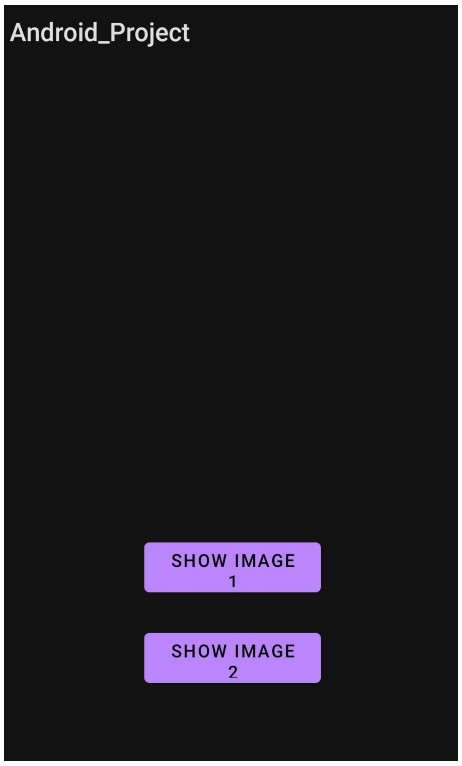


Figure 7.1

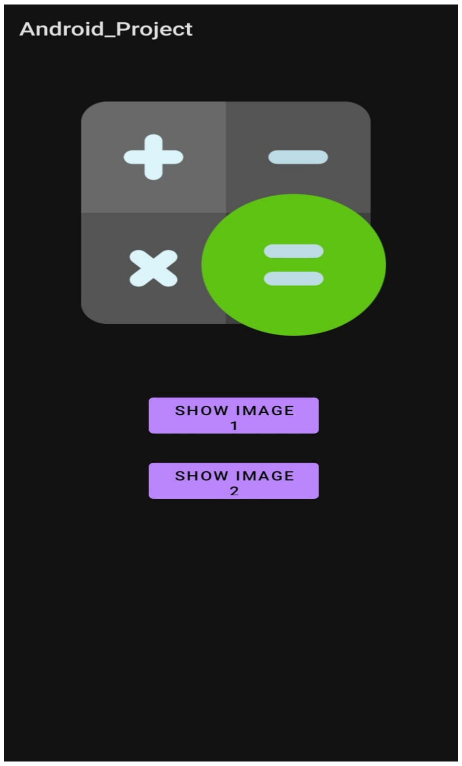


Figure 7.2

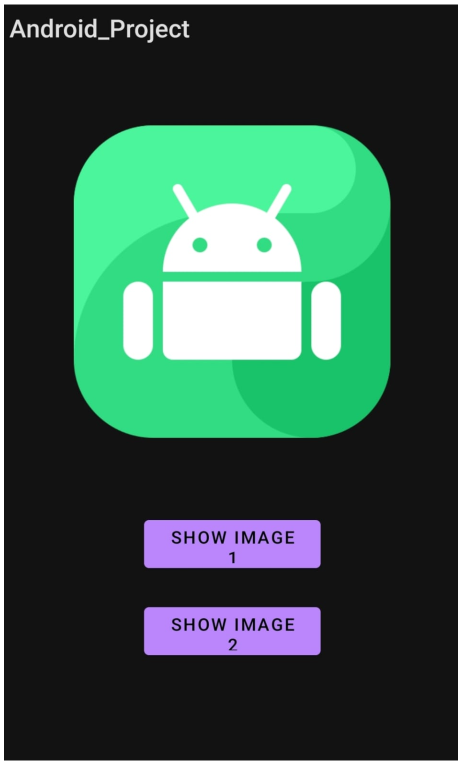


Figure 7.3

8 Android Security and Debugging: To implement concept of permission and perform request for permission to access different hardware components of mobile

1. Declare the permission in the Android Manifest file: In Android, permissions are declared in the AndroidManifest.xml file using the uses-permission tag.

```
<!--Declaring the required permissions-->
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.CAMERA" />
```

2. Modify activity_main.xml file to Add two buttons to request permission on button click: Permission will be checked and requested on button click. Open the activity_main.xml file and add two buttons to it.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical">


   <!--Button to request storage permission-->
   <ImageButton
      android:id="@+id/storage"
      android:layout_width="100pt"
      android:layout_height="100pt"
      app:srcCompat="@drawable/camera"
      android:layout_gravity="center"
      android:layout_marginTop="20pt"
      android:padding="8dp"
      android:layout_centerHorizontal="true"/>
<!-- android:layout_below="@id/toolbar"-->
<!-- android:layout_centerHorizontal="true"/>-->

   <!--Button to request camera permission-->




   <ImageButton
      android:id="@+id/camera"
      android:layout_width="100pt"
      android:layout_height="100pt"
      android:layout_marginTop="16dp"
      app:srcCompat="@drawable/storage"
      android:layout_gravity="center"
      android:padding="8dp"
      android:layout_below="@id/storage"
      android:layout_centerHorizontal="true"/>


</LinearLayout>
```

3. Check whether permission is already granted or not. If permission isn't already granted, request the user for the permission: In order to use any service or feature, the permis-

sions are required. Hence we have to ensure that the permissions are given for that. If not, then the permissions are requested.

Check for permissions: Beginning with Android 6.0 (API level 23), the user has the right to revoke permissions from any app at any time, even if the app targets a lower API level. So to use the service, the app needs to check for permissions every time.

4. MainActivity.java code:

```
package com.example.calculator_app;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

public class MainActivity6 extends AppCompatActivity {

   // Defining Buttons
   private ImageButton storage, camera;

   // Defining Permission codes.
   // We can give any value
   // but unique for each permission.
   private static final int CAMERA_PERMISSION_CODE = 100;
   private static final int STORAGE_PERMISSION_CODE = 101;

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

      storage = findViewById(R.id.storage);
      camera = findViewById(R.id.camera);

      // Set Buttons on Click Listeners
      storage.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v)
         {
            checkPermission(Manifest.permission.WRITE_EXTERNAL_STORAGE, STORAGE_PERMISSION_CODE);
         }
      });

      camera.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v)
         {
            checkPermission(Manifest.permission.CAMERA, CAMERA_PERMISSION_CODE);
         }
      });
   }

   // Function to check and request permission.
   public void checkPermission(String permission, int requestCode)
   {
        if (ContextCompat.checkSelfPermission(MainActivity6.this, permission) == PackageManager.
             PERMISSION_DENIED) {

            // Requesting the permission
            ActivityCompat.requestPermissions(MainActivity6.this, new String[] { permission },
                 requestCode);
        }
        else {
            Toast.makeText(MainActivity6.this, "Permission already granted", Toast.LENGTH_SHORT).show();
        }
    }

    // This function is called when the user accepts or decline the permission.
    // Request Code is used to check which permission called this function.
    // This request code is provided when the user is prompt for permission.

    @Override
    public void onRequestPermissionsResult(int requestCode,
                                           @NonNull String[] permissions,
                                           @NonNull int[] grantResults)
    {
        super.onRequestPermissionsResult(requestCode,
                permissions,
                grantResults);

        if (requestCode == CAMERA_PERMISSION_CODE) {
            if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                Toast.makeText(MainActivity6.this, "Camera Permission Granted", Toast.LENGTH_SHORT) .show
                     ();
            }
            else {
                Toast.makeText(MainActivity6.this, "Camera Permission Denied", Toast.LENGTH_SHORT) .show()
                     ;
            }
        }
        else if (requestCode == STORAGE_PERMISSION_CODE) {
            if (grantResults.length > 0
                    && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                Toast.makeText(MainActivity6.this, "Storage Permission Granted", Toast.LENGTH_SHORT).show
                     ();
            } else {
                Toast.makeText(MainActivity6.this, "Storage Permission Denied", Toast.LENGTH_SHORT).show()
                     ;
            }
        }
    }
}
```

8.0.1 Output Screens

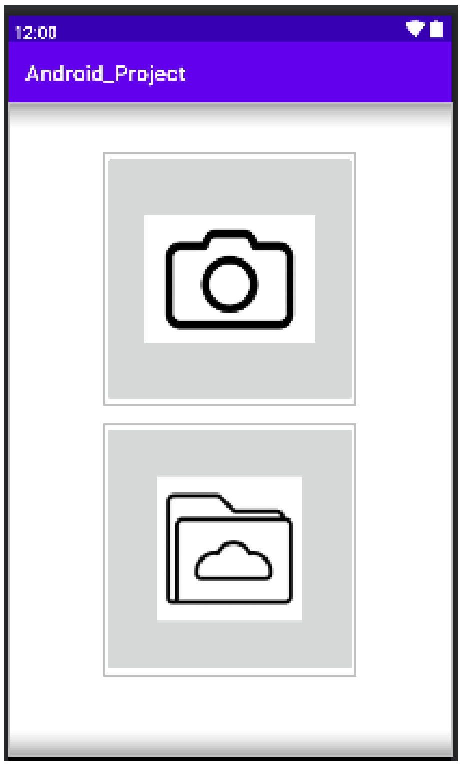


Figure 8.1: Permission Example using camera and storage permission

9 Android Security and Debugging: To perform debugging and testing of android app using tools like Logcat, Android debug bridge, DDMS

To perform debugging and testing of an Android app, you can use a variety of tools such as Logcat, Android Debug Bridge (ADB), and Dalvik Debug Monitor Server (DDMS). Here is a brief overview of how to use each of these tools:

1. Logcat: This is a command-line tool that displays system log messages. It is useful for debugging your app, as well as for understanding the system behavior. You can use Logcat to display log messages from your app, as well as from the Android system. To access Logcat, you can use the adb logcat command or the Android Studio Logcat tool.
2. Android Debug Bridge (ADB): This is a command-line tool that allows you to communicate with an Android device or emulator. You can use ADB to install, uninstall, and run your app on a device or emulator, as well as to execute shell commands on a device. ADB is included in the Android SDK Platform-Tools package, which you can download from the Android website.
3. Dalvik Debug Monitor Server (DDMS): This is a graphical tool that provides a number of debugging features, such as the ability to capture logcat output, view heap and thread information, and simulate incoming calls and SMS messages. DDMS is included in the Android SDK Tools package, which you can download from the Android website.

To use Logcat to debug your Android app, you can follow these steps:

1. Make sure that your device is connected to your computer and is in developer mode.
2. Open a command prompt or terminal window and navigate to the platform-tools directory within your Android SDK installation.
3. Type the following command to display log messages from your app:
   adb logcat com.your.package.name:V *:S
4. Replace com.your.package.name with the package name of your app.
   The :V flag tells Logcat to show verbose-level log messages, and the *:S flag tells it to suppress all log messages from other apps.
5. Run your app on the device or emulator and reproduce the issue you are trying to debug.
6. Look for log messages in the Logcat output that are related to the issue you are trying to debug. Pay attention to log messages with the tag E (error) or W (warning), as these may indicate problems with your app.

To use ADB to install and run your app on a device or emulator, you can follow these steps:

1. Make sure that your device is connected to your computer and is in developer mode.
2. Open a command prompt or terminal window and navigate to the platform-tools directory within your Android SDK installation.
3. Type the following command to install your app on the device:
   adb install com.your.package.name.apk
4. Replace com.your.package.name.apk with the path to your app's APK file.
5. Replace com.your.package.name.apk with the path to your app's APK file.
6. Replace com.your.package.name.apk with the path to your app's APK file.
7. Replace com.your.package.name.apk with the path to your app's APK file.
8. Once the installation is complete, you can type the following command to start your app: adb shell am start -n com.your.package.name/.MainActivity
9. Replace com.your.package.name/.MainActivity with the package name and main activity of your app.

To use DDMS to debug your Android app, you can follow these steps:

1. Open Android Studio and click on the "DDMS" button in the toolbar.
2. In the DDMS window, select the device or emulator that you want to debug.
3. Use the various tabs in the DDMS window to access debugging features such as logcat output, heap and thread information, and simulated incoming calls and SMS messages.
4. Run your app on the device or emulator and use DDMS to debug any issues you encounter

10 Project

10.0.1 XML Code

```
\begin{lstlisting}[language=XML]
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical">

   <ImageView
      android:id="@+id/imageView"
      android:layout_width="250dp"
      android:layout_height="250dp"
      android:layout_margin="50dp"
      android:layout_gravity="center"/>

   <Button
      android:layout_marginTop="60pt"
      android:id="@+id/b1"
      android:layout_width="70pt"
      android:layout_height="25pt"
      android:layout_margin="10dp"
      android:layout_gravity="center"
      android:text="Show Image 1"/>

   <Button
      android:id="@+id/b2"
      android:layout_width="70pt"
      android:layout_height="25pt"
      android:layout_margin="10dp"
      android:layout_gravity="center"
      android:text="Show Image 2"/>



</LinearLayout>
```

10.0.2 Java Code

```
package com.example.calculator_app;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.EditText;

public class MainActivity2 extends AppCompatActivity {

   String oldNumber ="";
   String op="+";
   EditText ed1;
   boolean isNew = true;

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

```
    ed1 = findViewById(R.id.editText);

}


public void handleNumberEvent(View view) {
    if(isNew)
        ed1.setText("");
    isNew = false;
    String number = ed1.getText().toString();
    switch(view.getId()){
        case R.id.bu1:
            number += "1";
            break;
        case R.id.bu2:
            number += "2";
            break;
        case R.id.bu3:
            number += "3";
            break;
        case R.id.bu4:
            number += "4";
            break;
        case R.id.bu5:
            number += "5";
            break;
        case R.id.bu6:
            number += "6";
            break;
        case R.id.bu7:
            number += "7";
            break;
        case R.id.bu8:
            number += "8";
            break;
        case R.id.bu9:
            number += "9";
            break;
        case R.id.bu0:
            number += "0";
            break;
        case R.id.buDot:
            number += ".";

        case R.id.buPlusMinus:
            number += "-"+number;
            break;

    }
    ed1.setText(number);
}

public void operatorEvent(View view){

    isNew = true;
    oldNumber = ed1.getText().toString();
    switch (view.getId()){
        case R.id.buPlus: op = "+"; break;
        case R.id.buMinus: op = "-"; break;
        case R.id.buMultiply: op = "*"; break;
        case R.id.buDivide: op = "/"; break;
    }

}

public void equalEvent(View view) {
    String newNumber = ed1.getText().toString();
    double result = 0.0;
```

```
        switch(op){
            case "+":
                result = Double.parseDouble(oldNumber) + Double.parseDouble(newNumber);
                break;
            case "-":
                result = Double.parseDouble(oldNumber) - Double.parseDouble(newNumber);
                break;
            case "*":
                result = Double.parseDouble(oldNumber) * Double.parseDouble(newNumber);
                break;
            case "/":
                result = Double.parseDouble(oldNumber) / Double.parseDouble(newNumber);
                break;
        }
        ed1.setText(result +"");
    }

    public void acEvent(View view) {
        ed1.setText("0");
        isNew = true;
    }

    public void percentEvent(View view) {
        double no = Double.parseDouble(ed1.getText().toString())/100;
        ed1.setText(no + "");
        isNew = true;

    }
}
```

10.0.3 Output Screens:

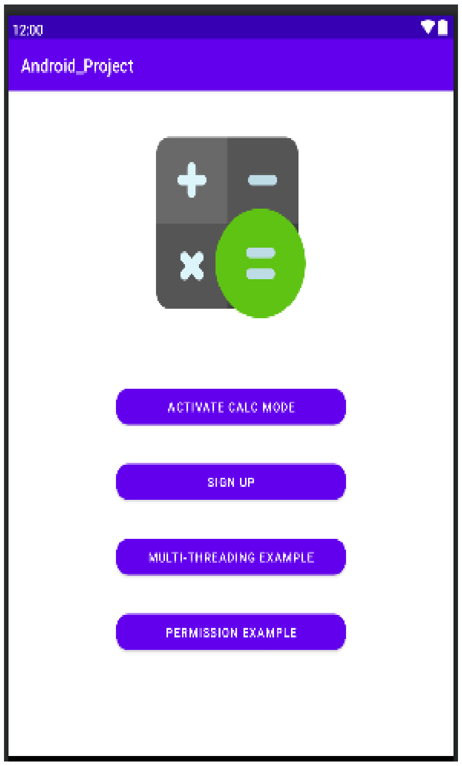


Figure 10.1: Home Page

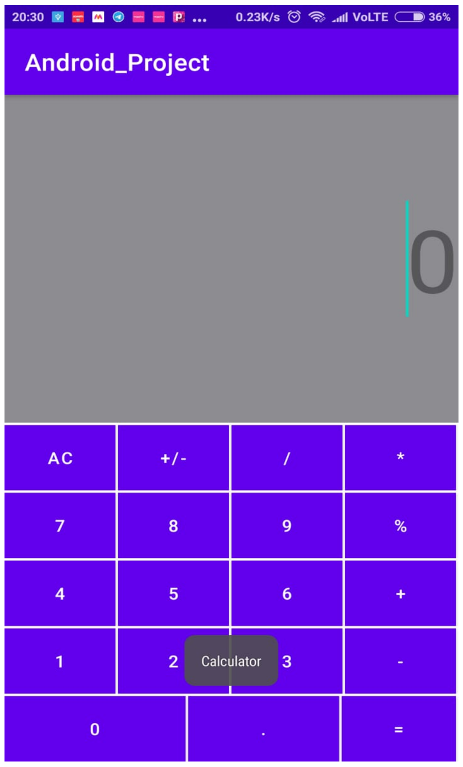


Figure 10.2: Calculator App