GUJARAT TECHNOLOGICAL UNIVERSITY (GTU)

Competency-focused Outcome-based Green Curriculum-2021(COGC-2021)
Semester -V

Course Title: Mobile Application Development
(Course Code: 4351604)

Diploma programme in which this course is offered	Semester in which offered
Information Technology	5th Semester

1. RATIONALE

Mobile applications have become an essential component of businesses across industries in today's technology-driven world. Mobile Applications provide enormous opportunities for businesses to reach and engage with their customers. As a result, there is a growing demand for skilled Mobile application developers capable of developing innovative and robust applications that meet the needs of both businesses and end users. This demand paves the way for the development of a dedicated course on Mobile application development that focuses on industrial applications and meets market demand. This course develops necessary skills in students, after learning this course; students will be able to develop user-friendly mobile applications.

2. COMPETENCY

The purpose of this course is to help the student to attain the following industry-identified competencies through various teaching-learning experiences:

- **Develop user-friendly Mobile applications, design intuitive user interfaces, understand and implement various app components, effectively debug and troubleshoot issues, adapt to emerging technologies and continuously enhance their skills to meet the demands of the industry.**

3. COURSE OUTCOMES (COs)

The student will develop underpinning knowledge and adequate skills of competency for developing various mobile applications after attaining the following course outcomes.

a) Understand the fundamentals of Android app development.
b) Design Android user interfaces using various layouts, views, and widgets.
c) Connect an Android app to SQLite, Firebase, and MySQL databases.
d) Understand the working of APIs in Android app development.
e) Develop basic mobile applications using the Flutter framework.
f) Understand the steps involved in publishing an Android app to the Google Play Store.

4. TEACHING AND EXAMINATION SCHEME

Teaching Scheme (In Hours)			Total Credits (L+T/2+P/2)	Examination Scheme				
				Theory Marks		Practical Marks		Total Marks
L	T	P	C	CA	ESE	CA	ESE	
0	0	4	2	0	0	25	25	50

***Legends: L**-Lecture; **T** – Tutorial/Teacher Guided Theory Practice; **P** -Practical; **C** – Credit, **CA** - Continuous Assessment; **ESE** -End Semester Examination.*

5. SUGGESTED PRACTICAL EXERCISES

The following practical outcomes (PrOs) are the subcomponents of the COs. These PrOs need to be attained to achieve the COs.

Sr. No.	Practical Outcomes (PrOs)	Unit No.	Approx. Hrs. required
1	Install Android Studio, set up the Android development environment, and create a simple "Hello World" app.	I	02
2	Develop a simple app that demonstrates the activity lifecycle.	I	02
3	Develop a simple calculator app that takes user input and performs basic arithmetic operations like addition, subtraction, multiplication, and division.	I	02
4	Develop an Android application that uses LinearLayout to arrange UI components vertically or horizontally.	II	02
5	Develop an Android application that uses RelativeLayout to arrange UI components relative to each other.	II	02
6	Develop an Android application that uses ScrollView to display a long list of items.	II	02
7	Develop an Android application that uses ListView and Custom Adapter to display a list of images with text.	II	02
8	Develop an Android application that uses the Navigation Drawer to display a side menu.	II	02
9	Develop an Android application that uses the bottom navigation bar to switch between different tabs.	II	02
10	Develop an Android application that uses an Intent to pass	III	02

Sr. No.	Practical Outcomes (PrOs)	Unit No.	Approx. Hrs. required
	data between different activities.		
11	Develop an Android application that uses Services to perform background tasks.	III	02
12	Develop an Android application that uses Broadcast Receivers to receive and handle system-level broadcasts.	III	02
13	Develop an Android application that uses Content Providers to share data between different apps and components.	III	02
14	Develop an Android application that uses Content Providers to read system-level data, such as contacts and calendar events.	III	02
15	Create an application that creates a database using SQLiteOpenHelper Class and performs Insert and Read from the SQLite database.	IV	02
16	Create an application to Update and Delete data from the SQLite database using SQLiteOpenHelper class.	IV	02
17	Perform Firebase Integration to your Android application and store the data in the Firebase Database.	IV	02
18	Create an application to retrieve data from the Firebase Database and display it in the RecyclerView.	IV	02
19	Connect an Android application to the MySQL database using PHP, and insert the data in the database table.	IV	02
20	Perform insertion of data to MySQL database using PHP from an Android application.	IV	02
21	Perform reading of the data from the MySQL database using PHP in the JSON format and display on the screen of an Android application.	IV	02
22	Integrate Google maps API to your Android application and display your current location in the app.	V	02
23	Integrate Google maps API to your Android application and find the distance of any nearby location from your current location and display it.	V	02
24	Develop an application which performs Login using the	V	02

Sr. No.	Practical Outcomes (PrOs)	Unit No.	Approx. Hrs. required
	Google account of the user.		
25	Install Flutter SDK, configure the development environment and display “Hello World” in the centre of the screen.	VI	02
26	Develop a Flutter app to get two numbers from the user and display addition on the screen after clicking a button.	VI	02
27	Develop a Flutter app for Login using static data. If the User ID and Password are correct then clicking a login button should open a new screen showing Username at the center of the new screen.	VI	02
28	Demonstrate publishing an Android app on the Google Play Store following the policies and guidelines.	VII	02

Note

i. *More **Practical Exercises** can be designed and offered by the respective course faculty to develop the industry-relevant skills/outcomes to match the COs. The above table is only a suggestive list**.***

ii. ***Faculty can suggest students to develop a complete mobile application performing the practicals as listed above. Functionalities can be selected such that after completing the app, all the PrOs listed above can be achieved.***

iii. *The following are some **sample** ‘Process’ and ‘Product’ related skills (more may be added/deleted depending on the course) that occur in the above listed **Practical Exercises** of this course required which are embedded in the COs and ultimately the competency.*

S. No.	Sample Performance Indicators for the PrOs	Weightage in %
1	User Interface Design.	20
2	Coding methodology.	30
3	Testing and debugging of the program.	20
4	Correctness of the program.	20
5	Submission within the time limit.	10
Total		**100**

6. MAJOR EQUIPMENT/ INSTRUMENTS/SOFTWARE REQUIRED

This major equipment/instrument/software with broad specifications for the PrOs is a guide to procure them by the administrators. This will ensure the conduction of practicals in all institutions across the state in the proper way so that the desired skills are developed in students.

S. No.	Equipment Name with Broad Specifications	PrO. No.
1	Computer system with minimum 8 GB RAM, intel core-i5 processor and 128 GB SSD(recommended).	All
2	Android Studio, Xampp Server.	

7. AFFECTIVE DOMAIN OUTCOMES

The following ***sample*** Affective Domain Outcomes (ADOs) are embedded in many of the above-mentioned COs and PrOs. More could be added to fulfill the development of this competency.

a) Work as a leader/ team member.
b) Follow ethical practices.

The ADOs are best developed through the laboratory/field-based exercises. Moreover, the level of achievement of the ADOs according to Krathwohl's 'Affective Domain Taxonomy' should gradually increase as planned below:

i. 'Valuing Level' in 1st year
ii. 'Organization Level' in 2nd year.
iii. 'Characterization Level' in 3rd year.

8. UNDERPINNING THEORY

Only the major underpinning theory is formulated as higher-level UOs of *revised Bloom's taxonomy* in the development of the COs, and competency is not missedby the students and teachers. If required, more such higher-level UOs could be included by the course teacher to focus on the attainment of COs and competency.

Unit	Unit Outcomes (UOs)	Topics and Sub-topics
Unit – I **Introduction to Android App Development**	1a. Explain Android OS, its architecture and versions. 1b. Understand Android SDK and development environment 1c. Explain Activity Life cycle. 1d. Explain event-driven programming	1.1 Introduction to Android 1.2 Android OS features 1.3 Versions of an Android OS 1.4 Android SDK 1.5 Android Virtual Device 1.6 Activity in Android 1.7 Activity Life Cycle 1.8 Event-driven programming in Android

Unit – II **Building User Interfaces**	2a. Designing UIs using XML layout files 2b. Explain Layout types and view groups 2c. Explain Views and widgets. 2d. Create menus, Navigation Drawer and Bottom Navigation Bar 2e. Display the data in the ListView and RecyclerView using Adapter	2.1 Designing UI 2.2 XML for Designing, Padding, Margin 2.3 Layouts - ViewGroup in Android 2.4 Views and Widgets – Button, TextView, EditText, ImageView, ScrollView 2.5 Menus in Android, Navigation Drawer, Bottom Navigation Bar 2.6 Adapter, ListView, RecyclerView
Unit-III **Android App Components**	3a. Explain Intents, Services, Broadcast Receivers, and Content Providers 3b. Understand background tasks and Threads.	3.1 Intent in Android 3.2 Services, Broadcast receivers 3.3 Content providers in Android 3.4 Toast in Android 3.5 Threads and background tasks in Android
Unit-IV **Database Connectivity**	4a. Develop application using SQLite databases 4b. Use Firebase for database operations. 4c. Explain how to connect an Android app to MySQL databases via PHP and JSON.	4.1 SQLite Database 4.2 SQLiteOpenHelper Class 4.3 Create, open and close the database using SQLiteOpenHelper 4.4 Insert, Read, Update and delete the data from the table using SQLiteOpenHelper.
		4.5 Integration of Firebase to application. 4.6 Insert, Read, Update and Delete data to the Firebase database. 4.7 Database operations using PHP in MySQL from an Android application.
Unit-V **Working with APIs**	5a. Explain working with the APIs and parsing the JSON data into Android applications. 5b. Demonstrate working with Google Maps API to display maps and locations. 5c. Perform integration with Google Sign-In to authenticate users.	5.1 API - Application Programming Interface 5.2 Working with APIs 5.3 JSON Parsing in Android 5.4 Google Maps 5.5 Map Activity 5.6 Configure Google API console 5.7 Google Maps API for locations 5.8 User Authentication using Google Account

Unit-VI **Introduction to Flutter**	6a.Explain the Flutter framework and its advantages. 6b. Develop a basic Flutter app.	6.1 Introduction to Flutter 6.2 Flutter SDK 6.3 Advantages and Disadvantages of Flutter 6.4 Flutter application development configuration 6.5 Basic UI components like Text, TextField, Buttons in Flutter 6.6 Navigation and Routing in Flutter
Unit-VII **Publishing App on Google Play**	7a. Prepare an app to release and generate a signed APK. 7b. Describe the Google Play Store guidelines and policies. 7c. Explain the procedure to upload and publish an app on Google Play.	7.1 Overview of Publish an app 7.2 Prepare for release 7.3 Version your app 7.4 Sign your app 7.5 Google Play store guidelines and Policies 7.6 Test your app 7.7 Upload your app

***Note**: The UOs need to be formulated at the 'Application Level' and above of Revised Bloom's Taxonomy' to accelerate the attainment of the COs and the competency.*

9. SUGGESTED STUDENT ACTIVITIES

Other than laboratory learning, the following are the suggested student-related ***co-curricular*** activities which can be undertaken to accelerate the attainment of the various outcomes in this course: Students should conduct the following activities in groups and prepare reports of about 5 pages for each activity, also collect/record physical evidence for their (student's) portfolio which will be useful for their placement interviews:

a) Prepare a report based on practicals performed in the laboratory.
b) Undertake micro-projects in teams.
c) Give a seminar on any relevant topics.

10. SUGGESTED SPECIAL INSTRUCTIONAL STRATEGIES (if any)

These are sample strategies, which the teacher can use to accelerate the attainment of the various outcomes in this course:

a) Massive open online courses (***MOOCs***) may be used to teach various topics/subtopics.
b) Guide student(s) in undertaking micro-projects.
c) About ***20% of the topics/sub-topics*** which are relatively simpler or descriptive in nature are to be given to the students for ***self-learning***, but to be assessed using different assessment methods.

d) For Practicals, Faculty can suggest students to develop a complete mobile application performing all the practicals listed in Section 5.
e) With respect to ***section No.9***, teachers need to ensure to create opportunities and provisions for ***co-curricular activities***.
f) Guide students on how to address issues of society, environment and sustainability using the knowledge of this course.
g) More focus should be given on practical work which will be carried out in laboratory sessions. If possible, some theory sessions may be conducted in labs so that theory and practical can go hand in hand.
h) Arrange a Mobile application development/UI development competition by making groups of a maximum of three students each and award the winning group.

11. SUGGESTED MICRO-PROJECTS

Only one micro-project is planned to be undertaken by a student that needs to be assigned to him/her in the beginning of the semester. In the first four semesters, the micro-project is group-based. However, in the fifth and sixth semesters, it should preferably be ***individually*** undertaken to build up the skill and confidence in every student to become a problem solver so that s/he contributes to the projects of the industry. In special situations where groups have to be formed for micro-projects, the number of students in the group should ***not exceed Four.***

The micro-project could be industry application based, internet-based, workshop-based, laboratory-based or field-based. Each micro-project should encompass two or more COs which are in fact, an integration of PrOs, UOs and ADOs. Each student will have to maintain a dated work diary consisting of individual contributions in the project work and give a seminar presentation of it before submission. The total duration of the micro-project should not be less than ***16 (sixteen) student engagement hours*** during the course. The student ought to submit a micro-project by the end of the semester to develop the industry-oriented COs.

A suggestive list of micro-projects is given here. This has to match the competency and the COs. Similar micro-projects could be added by the concerned course faculty:

Case Study 1: Develop a Mobile Application to manage and store students' data with the following functionalities:

1. Registration
2. Login
3. Search details by student enrollment no
4. Update and delete the data

Case Study 2: Develop a Mobile Application to buy and sell products with the following functionality:

1. Registration and Login
2. Add Product by Seller
3. View Orders by Seller
4. View Products by Buyer
5. Order Products by Buyer

12. SUGGESTED LEARNING RESOURCES

S. No.	Title of Book	Author	Publication with place, year and ISBN
1	Head First Android Development	Dawn Griffiths, David Griffiths	,.Inc ,Reilly Media'O ISBN: 9781492076476
2	Android Programming for Beginners	John Horton	Ingram short title ISBN-10:1789538505
3	Flutter and Dart Cookbook	Richard Rose	Shroff/O'Reilly,Navi Mumbai ISBN: 9789355422446

13. SOFTWARE/LEARNING WEBSITES

- Android Development Kit: https://developer.android.com
- https://flutter.dev/
- https://developers.google.com/apis-explorer
- https://onlinecourses.swayam2.ac.in/nou21_ge41/preview
- https://www.javatpoint.com/android-tutorial
- https://www.tutorialspoint.com/android/index.htm
- https://www.geeksforgeeks.org/android-tutorial/
- https://play.google.com/console/about/

14. PO-COMPETENCY-CO MAPPING

Semester V	Mobile Application Development (Course Code:4351604)						
	POs and PSOs						
Competency & Course Outcomes	PO 1 Basic & Discipline specific knowledge	PO 2 Problem Analysis	PO 3 Design/ development of solutions	PO 4 Engineering Tools, Experimentation And Testing	PO 5 Engineering practices for society, sustainability & environment	PO 6 Project Management	PO 7 Life-long learning
Competency Develop user-friendly Mobile applications, design intuitive user interfaces, understand and implement various app components, effectively debug and troubleshoot issues, adapt to emerging technologies and continuously enhance their skills to meet the demands of the industry.							
Course Outcomes a) Understand the fundamentals of Android app development.	2	-	-	1	2	-	1
b) Design Android user interfaces using various layouts, views, and widgets.	2	2	3	2	-	-	-

c) Connect an Android app to SQLite, Firebase, and MySQL databases.	2	2	3	2	-	-	2
d) Understand the working of APIs in Android app development.	2	-	3	2	-	-	2
e) Develop basic mobile applications using the Flutter framework.	2	2	2	2	-	-	-
f) Understand the steps involved in publishing an Android app to the Google Play Store.	-	-	-	1	2	-	2

Legend: '**3'** for high, '**2**' for medium, **'1'** for low or **'-'** for the relevant correlation of each competency, CO, with PO/ PSO

15. COURSE CURRICULUM DEVELOPMENT COMMITTEE

GTU Resource Persons

S. No.	Name and Designation	Institute	Email
1	Roshan R. Rohit Lecturer in I.T.	Govt. Polytechnic for Girls, Surat.	roshanrohit2989@gmail.com
2	Chinkit D. Suthar Lecturer in I.T.	Govt. Polytechnic, Himatnagar.	cdsuthar91@gmail.com

★ activity_main.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"
    android:padding="16dp">

    <EditText
        android:id="@+id/editTextNumber1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberDecimal"
        android:hint="Enter number 1"/>

    <EditText
        android:id="@+id/editTextNumber2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberDecimal"
        android:hint="Enter number 2"/>

    <Button
        android:id="@+id/buttonAdd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="+"/>

    <Button
        android:id="@+id/buttonSubtract"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="-"/>

    <Button
        android:id="@+id/buttonMultiply"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="*"/>

    <Button
        android:id="@+id/buttonDivide"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="/"/>

    <TextView
        android:id="@+id/textViewResult"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
    android:paddingTop="16dp"
    android:text="Result: "
    android:textSize="18sp"/>
</LinearLayout>
```

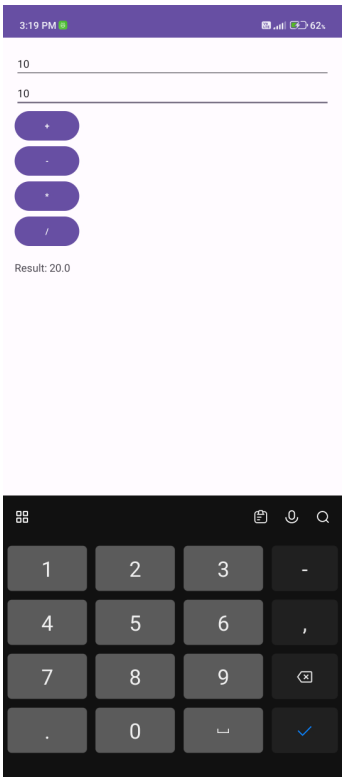
- ★ MainActivity. java

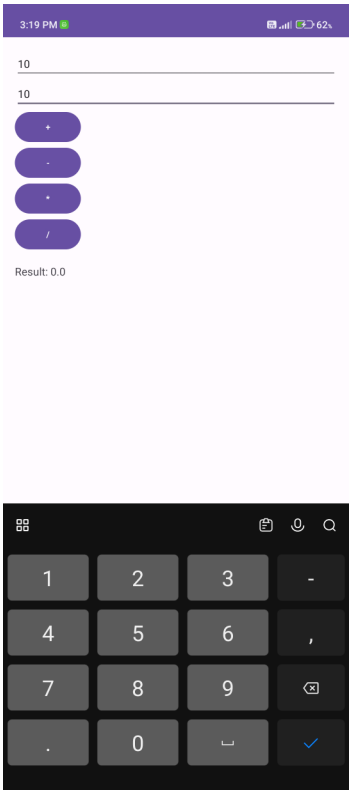
```
package com.example.aidrago;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

  private EditText input1, input2;
  private TextView resultText;

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

    input1 = findViewById(R.id.editTextNumber1);
    input2 = findViewById(R.id.editTextNumber2);
    resultText = findViewById(R.id.textViewResult);

    Button addButton = findViewById(R.id.buttonAdd);
    Button subtractButton = findViewById(R.id.buttonSubtract);
    Button multiplyButton = findViewById(R.id.buttonMultiply);
    Button divideButton = findViewById(R.id.buttonDivide);

    addButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        calculate('+');
      }
    });

    subtractButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        calculate('-');
      }
    });

    multiplyButton.setOnClickListener(new View.OnClickListener() {
```

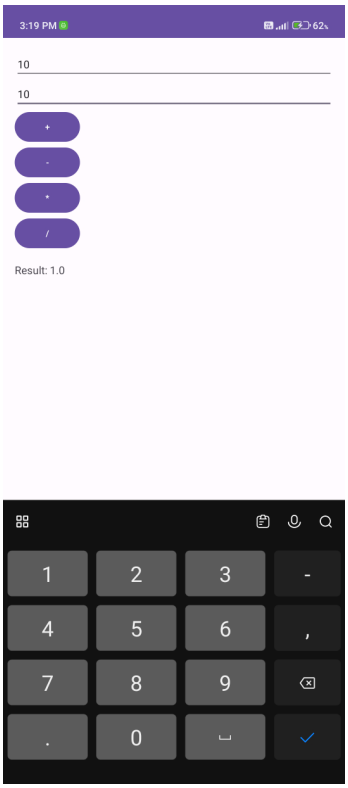
```
        @Override
        public void onClick(View v) {
            calculate('*');
        }
    });

    divideButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            calculate('/');
        }
    });
}

private void calculate(char operation) {
    try {
        double num1 = Double.parseDouble(input1.getText().toString());
        double num2 = Double.parseDouble(input2.getText().toString());
        double result = 0;

        switch (operation) {
            case '+':
                result = num1 + num2;
                break;
            case '-':
                result = num1 - num2;
                break;
            case '*':
                result = num1 * num2;
                break;
            case '/':
                if (num2 != 0) {
                    result = num1 / num2;
                } else {
                    Toast.makeText(this, "Cannot divide by zero!", Toast.LENGTH_SHORT).show();
                    return;
                }
                break;
        }

        resultText.setText("Result: " + result);
    } catch (NumberFormatException e) {
        Toast.makeText(this, "Invalid input", Toast.LENGTH_SHORT).show();
    }
}
}
```

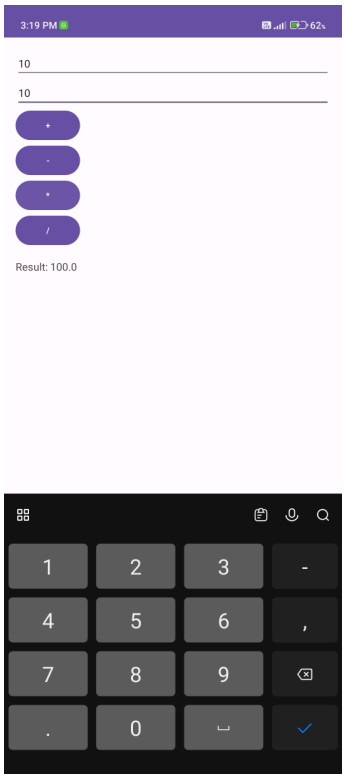
★ Output

4351604(MAD)









216120316028

★ activity_main.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"
   android:gravity="center">
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   android:gravity="center">

   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Button 1"/>

   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Button 2"/>

   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Button 3"/>
</LinearLayout>

<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="horizontal"
   android:gravity="center">

   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Button 1"/>

   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Button 2"/>

   <Button
```

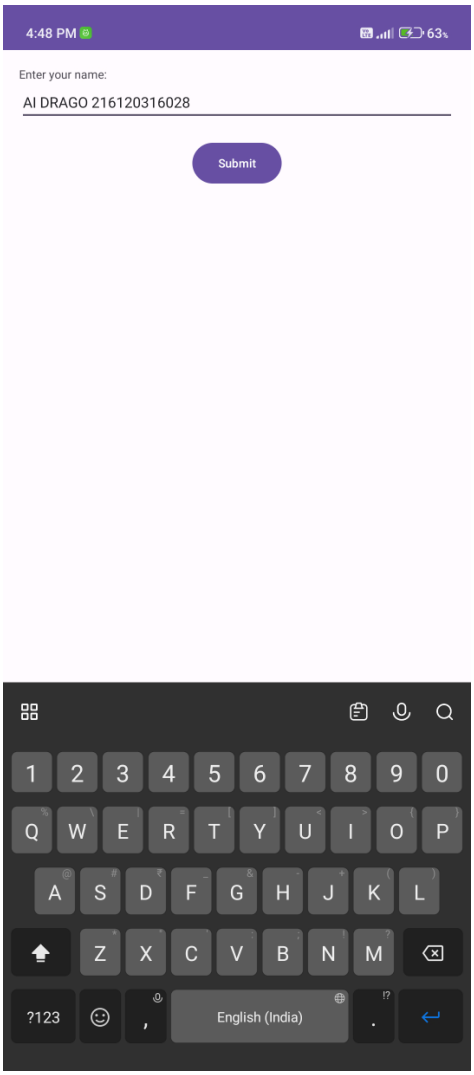
```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button 3"/>
</LinearLayout>


</LinearLayout>
```

- **Output**

★ activity_main.xml

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

<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"

   android:layout_width="match_parent"

   android:layout_height="match_parent">

   <TextView

      android:id="@+id/textView"

      android:layout_width="wrap_content"

      android:layout_height="wrap_content"

      android:text="Enter your name:"

      android:layout_marginTop="16dp"

      android:layout_marginStart="16dp"/>

   <EditText

      android:id="@+id/editText"

      android:layout_width="match_parent"

      android:layout_height="wrap_content"

      android:layout_below="@id/textView"

      android:layout_marginStart="16dp"

      android:layout_marginEnd="16dp"/>

   <Button

      android:id="@+id/button"

      android:layout_width="wrap_content"

      android:layout_height="wrap_content"

      android:text="Submit"

      android:layout_below="@id/editText"

      android:layout_centerHorizontal="true"

      android:layout_marginTop="16dp"/>

</RelativeLayout>
```

• Output

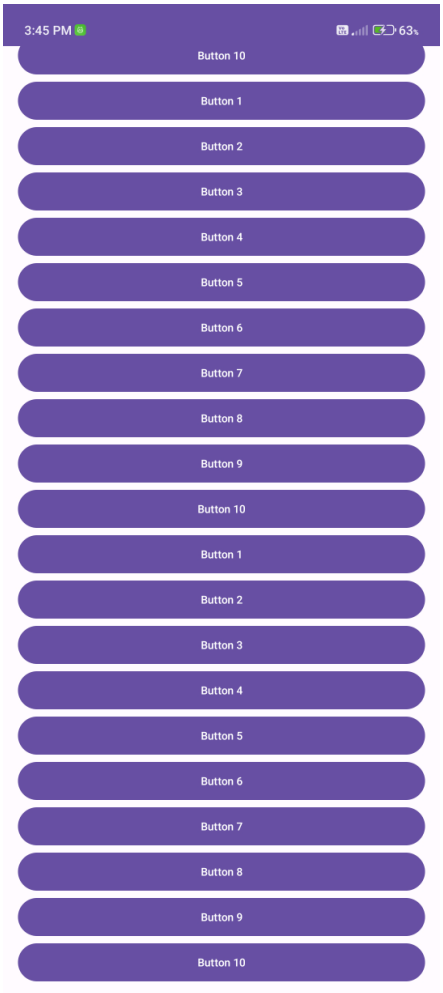


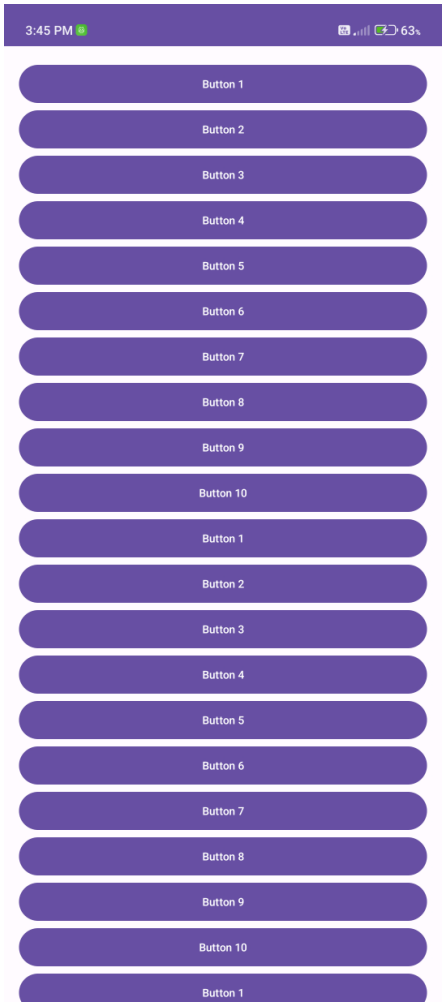
★ activity_main.xml

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

   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="vertical"
      android:padding="16dp">

      <Button
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="Button 1"/>

      <Button
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="Button 2"/>

      <Button
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="Button 3"/>

      <Button
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="Button 4"/>

      <Button
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="Button 5"/>

      <Button
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="Button 6"/>

      <Button
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="Button 7"/>

      <Button
         android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:text="Button 8"/>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Button 9"/>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Button 10"/>

.....

    </LinearLayout>
</ScrollView>
```

- **Output**





- **activity_main.xml:**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    tools:context=".MainActivity">
    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

- **List_item.xml:**

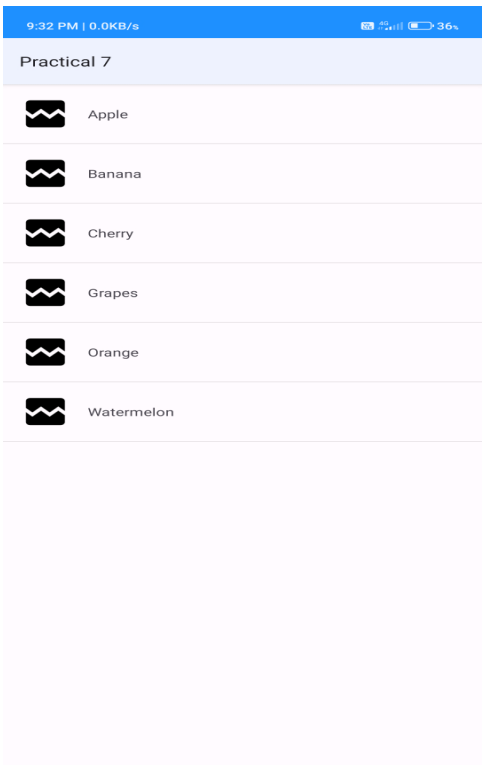
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:padding="16dp">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:scaleType="centerCrop" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginStart="16dp"
        android:textSize="18sp" />
</LinearLayout>
```

- **MainActivity.java:**

```
package com.example.practical7;

import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ListView;
import androidx.appcompat.app.AppCompatActivity;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    ListView listView;
    String[] fruitNames;
    int[] fruitImages;
```

```
    ArrayAdapter<String> adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        listView = findViewById(R.id.listView);

        fruitNames = new String[]{"Apple", "Banana", "Cherry", "Grapes", "Orange",
"Watermelon"};
        adapter = new ArrayAdapter<String>(this, R.layout.list_item, R.id.textView,
fruitNames) {
            @Override
            public View getView(int position, View convertView, ViewGroup parent) {
                if (convertView == null) {
                    convertView = getLayoutInflater().inflate(R.layout.list_item, null);      }

                ImageView imageView = convertView.findViewById(R.id.imageView);
                TextView textView = convertView.findViewById(R.id.textView);

                imageView.setImageResource(R.drawable.image);
                textView.setText(fruitNames[position]);

                return convertView;      }
        };
        listView.setAdapter(adapter);    }
}
```

★ Output:



- **activity_main.xml:**

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayou
t 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:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:openDrawer="start">
    <androidx.coordinatorlayout.widget.CoordinatorLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <com.google.android.material.appbar.AppBarLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <androidx.appcompat.widget.Toolbar
                android:id="@+id/toolbar"
                android:layout_width="match_parent"
                android:layout_height="?attr/actionBarSize"
                android:background="?attr/colorPrimary"
                app:popupTheme="@style/ThemeOverlay.AppCompat.Light"
                app:layout_scrollFlags="scroll|enterAlways" />
        </com.google.android.material.appbar.AppBarLayout>
    </androidx.coordinatorlayout.widget.CoordinatorLayout>
    <com.google.android.material.navigation.NavigationView
        android:id="@+id/nav_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        app:menu="@menu/nav_drawer" />

</androidx.drawerlayout.widget.DrawerLayout>
```

- **nav_drawer.xml:**

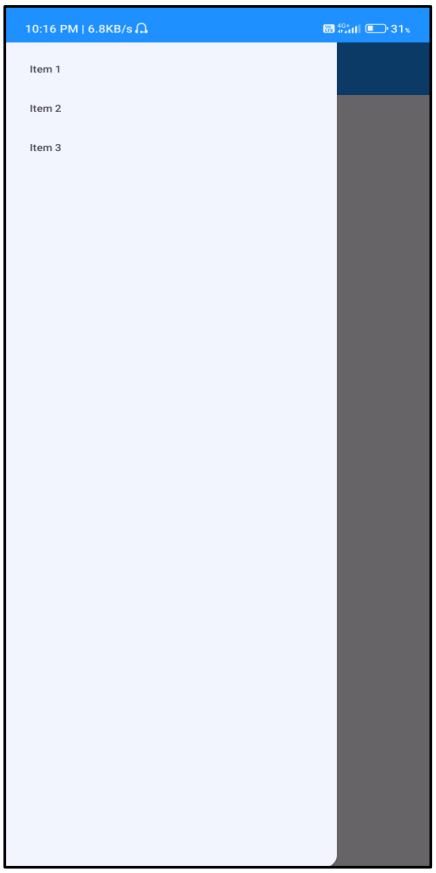
```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/menu_item1"
        android:title="Item 1" />
    <item
        android:id="@+id/menu_item2"
        android:title="Item 2" />
    <item
        android:id="@+id/menu_item3"
        android:title="Item 3" />
</menu>
```

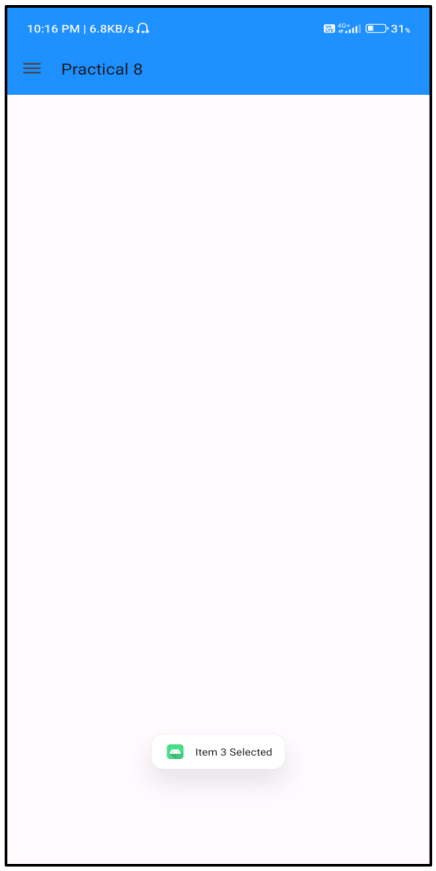
- **MainActivity.java:**

```
package com.example.practical8;

import android.os.Bundle;
import android.view.MenuItem;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBarDrawerToggle;
import androidx.core.view.GravityCompat;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;

import com.google.android.material.navigation.NavigationView;

public class MainActivity extends AppCompatActivity {
    DrawerLayout drawerLayout;
    ActionBarDrawerToggle toggle;

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

        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        drawerLayout = findViewById(R.id.drawer_layout);
        toggle = new ActionBarDrawerToggle(this, drawerLayout,
toolbar,R.string.app_name, R.string.app_name);
        drawerLayout.addDrawerListener(toggle);
        toggle.syncState();

        NavigationView navigationView = findViewById(R.id.nav_view);
        navigationView.setNavigationItemSelectedListener(new
NavigationView.OnNavigationItemSelectedListener() {
            @Override
            public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
                // Handle menu item clicks here
                int id = menuItem.getItemId();
                if (id == R.id.menu_item1) {
                    Toast.makeText(MainActivity.this, "Item 1 Selected",
Toast.LENGTH_SHORT).show();
                } else if (id == R.id.menu_item2) {
                    Toast.makeText(MainActivity.this, "Item 2 Selected",
Toast.LENGTH_SHORT).show();
                } else if (id == R.id.menu_item3) {
                    Toast.makeText(MainActivity.this, "Item 3 Selected",
Toast.LENGTH_SHORT).show();
                }
                drawerLayout.closeDrawer(GravityCompat.START);
                return true;
            }
        });
    }
```

```
}
```

★ **Output:**


10:16 PM | 6.8KB/s
Item 1
Item 2
Item 3


10:16 PM | 6.8KB/s
Practical 8
Item 3 Selected

- **activity_main.xml:**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottom_navigation"
        android:layout_width="match_parent"
        android:layout_height="70dp"
        android:padding="10dp"
        android:layout_alignParentBottom="true"
        app:menu="@menu/bottom_nav" />
</RelativeLayout>
```

- **bottom_nav.xml:**

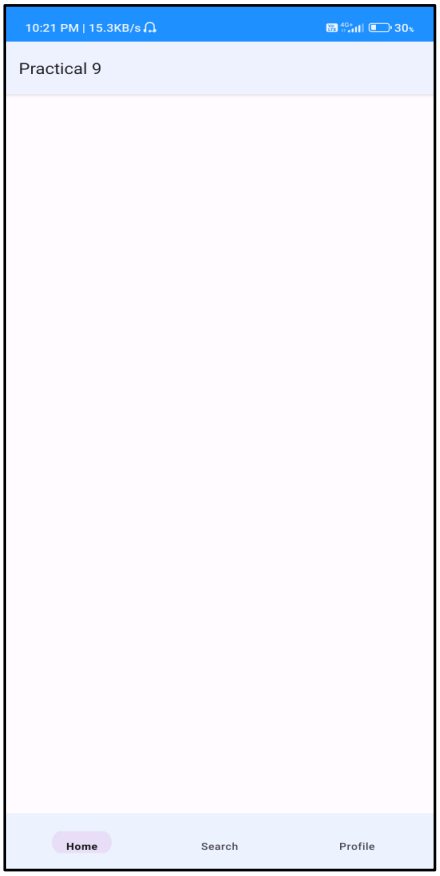
```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/menu_item1"
        android:title="Home" />
    <item
        android:id="@+id/menu_item2"
        android:title="Search" />
    <item
        android:id="@+id/menu_item3"
        android:title="Profile" />
</menu>
```

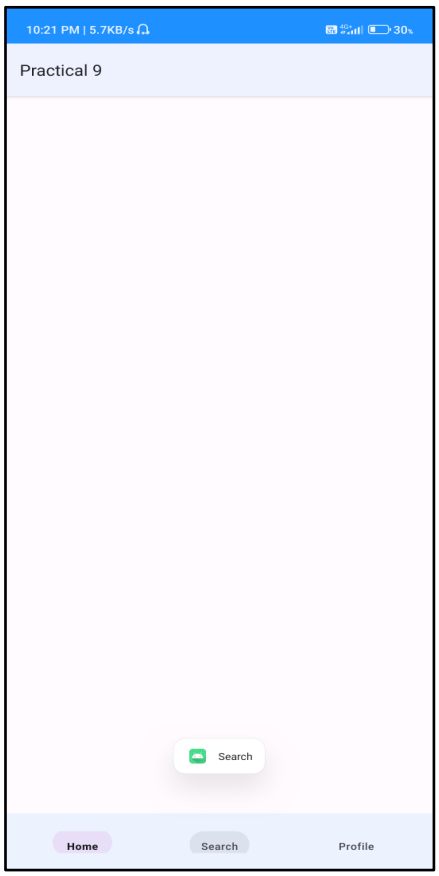
- **MainActivity.java:**

```
package com.example.practical9;

import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
```

```
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import com.google.android.material.bottomnavigation.BottomNavigationView;

public class MainActivity extends AppCompatActivity {

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

      BottomNavigationView bottomNavigationView =
findViewById(R.id.bottom_navigation);

      bottomNavigationView.setOnNavigationItemSelectedListener(new
BottomNavigationView.OnNavigationItemSelectedListener() {
         @Override
         public boolean onNavigationItemSelected(@NonNull MenuItem item) {
            if (item.getItemId() == R.id.menu_item1) {
               Toast.makeText(MainActivity.this, "Home",
Toast.LENGTH_SHORT).show();
            } else if (item.getItemId() == R.id.menu_item2) {
               Toast.makeText(MainActivity.this, "Search",
Toast.LENGTH_SHORT).show();

            } else if (item.getItemId() == R.id.menu_item3) {
               Toast.makeText(MainActivity.this, "Profile",
Toast.LENGTH_SHORT).show();

            }
            return false;
         }
      });
   }
}
```

★ Output:


10:21 PM | 15.3KB/s
Practical 9
Home
Search
Profile


10:21 PM | 5.7KB/s
Practical 9
Search
Home
Search
Profile

- **activity_main.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="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    android:padding="40dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Enter Name"
        android:layout_marginBottom="20dp"
        android:textSize="28sp"/>
    <EditText
        android:id="@+id/name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Your Name"
        android:layout_marginBottom="40dp"
        android:textSize="18sp"  />

    <Button
        android:id="@+id/submit"
        android:layout_width="200dp"
        android:layout_height="60dp"
        android:text="Submit"
        android:textSize="25dp"/>

</LinearLayout>
```

- **MainActivity.java:**

```
package com.example.practical10;


import androidx.appcompat.app.AppCompatActivity;


import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;


public class MainActivity extends AppCompatActivity {


    EditText name;
```

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


        name = findViewById(R.id.name);
        submit = findViewById(R.id.submit);


        submit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String Name = name.getText().toString();
                startActivity(new
Intent(MainActivity.this,NextActivity.class).putExtra("Name", Name));
            }
        });


    }
}
```

★ activity_next.xml:

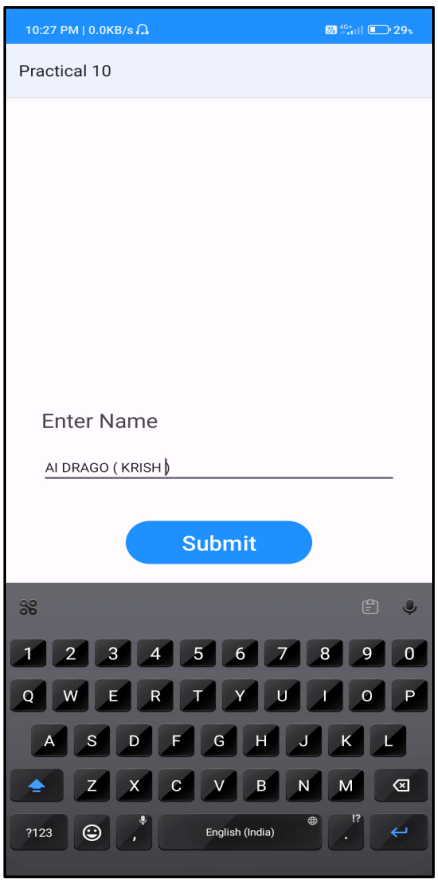
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"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

    <TextView
        android:id="@+id/txtname"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="28sp"/>

</LinearLayout>
```

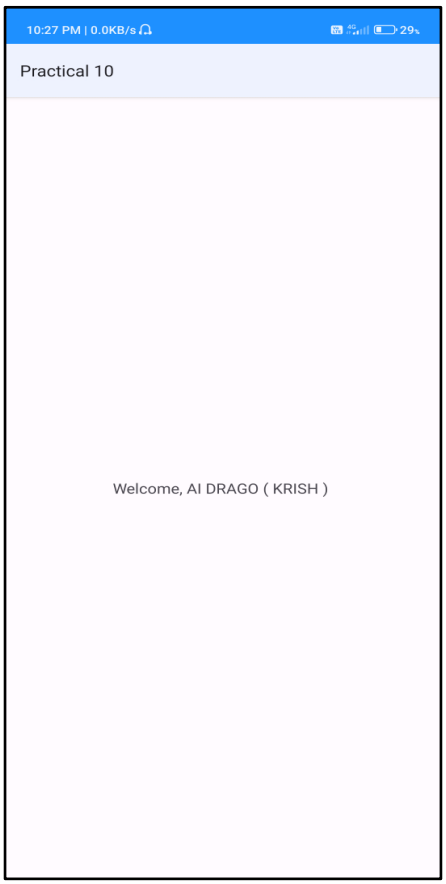
★ NextActivity.java:

```
package com.example.practical10;
```

```
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.widget.TextView;

public class NextActivity extends AppCompatActivity {

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

        txt = findViewById(R.id.txtname);

        Intent intent = getIntent();
        String Name = intent.getStringExtra("Name");
        txt.setText("Welcome, "+ Name);
    }
}
```

★ Output:





Aim: Develop an Android application that uses Services to perform background tasks.

In Android, a Service is a fundamental component that allows you to run background tasks or processes independently of the user interface. Services are particularly useful for performing long-running or background tasks without affecting the responsiveness of your app. In this response, I'll explain in detail what Services are and how to perform background tasks using them in an Android app.

What is an Android Service?

A Service in Android is a type of component that runs in the background, independent of the user interface. It doesn't have a user interface, making it ideal for tasks that need to run quietly without user interaction. Services are commonly used for tasks such as:

1. **Fetching data from a remote server**: Downloading data from the internet in the background.
2. **Playing music or audio**: Managing the playback of audio, even when the app is not in the foreground.
3. **Location tracking**: Continuously monitoring the device's location.
4. **File operations**: Copying, uploading, or processing files without user intervention.
5. **Periodic data synchronization**: Ensuring that app data is up-to-date in the background.

How to Create and Use an Android Service for Background Tasks:

To create and use a Service for background tasks in your Android app, follow these steps:

1. Create a Service Class:

You need to create a class that extends the **Service** class or one of its subclasses. For most use cases, you can use the base **Service** class. Here's a simple

example:

Java code

```
public class MyBackgroundService extends Service {
  @Override
  public IBinder onBind(Intent intent) {
    // Not used in this example.
    return null;
  }
  @Override
  public int onStartCommand(Intent intent, int flags, int startId) {
    // This method is called when the Service starts.
    // Perform your background task here.

    // Return a value that determines how the system should handle your Service.
    return START_STICKY;
  }
}
```

2. Start and Stop the Service:

You can start and stop a Service using **startService()** and **stopService()** (or **stopSelf()**) methods. For example:

Java code

```
// Start the service
Intent serviceIntent = new Intent(context, MyBackgroundService.class);
startService(serviceIntent);
// Stop the service
stopService(serviceIntent);
```

3. Handle Background Tasks:

Within the **onStartCommand()** method, you should perform your background

task. It's essential to use a separate thread or an **AsyncTask** if the task is time-consuming or might block the main UI thread to keep your app responsive. For example:

Java code:

```
@Override
public int onStartCommand(Intent intent, int flags, int startId) {
   // Perform a background task in a separate thread.
   new Thread(new Runnable() {
      public void run() {
         // Your background task code here.
      }
   }).start();

   return START_STICKY;}
```

4. Stop the Service:

Services don't automatically stop themselves once the task is complete. You should explicitly stop the Service when it's no longer needed. Typically, you do this in your **onStartCommand()** method after the task is finished:

javaCopy code

```
@Override
public int onStartCommand(Intent intent, int flags, int startId) {
   // Perform a background task here.

   // Stop the service when the task is complete.
   stopSelf();

   return START_NOT_STICKY;
```

```
}
```

Service Lifecycle:

Services have a lifecycle similar to Activities, but they are not directly tied to a user interface. The main lifecycle methods for a Service are:

- **onCreate()**: Called when the Service is created.
- **onStartCommand()**: Called when the Service is started.
- **onBind()**: Called when a client binds to the Service (not commonly used in background tasks).
- **onDestroy()**: Called when the Service is no longer needed and is being destroyed.

Types of Services:

There are three types of Services in Android:

1. **Foreground Service**: A foreground service is a Service that has a visible notification and is considered a high-priority task. It's used for tasks that the user is aware of, such as ongoing music playback.
2. **Background Service**: A background service is a Service that performs tasks in the background without displaying a notification. These are suitable for tasks like periodic data synchronization.
3. **Bound Service**: A bound service is a Service that allows other components (typically Activities) to bind to it and interact with it. This is used when a client needs to communicate with the Service.

Remember to declare your Services in the AndroidManifest.xml file to ensure that the system can start and stop them as needed.

Aim: Develop an Android application that uses Broadcast Receivers to receive and handle system-level broadcasts.

Broadcast Receivers in Android are a fundamental component of the Android operating system that allows apps to receive and respond to system-level broadcasts or messages. These broadcasts are sent by the Android system, other apps, or the device itself to inform various components and apps about system events or changes in the device's state. Broadcast Receivers play a crucial role in enabling apps to respond to events without the need for continuous polling or user interaction. In this response, I'll explain in detail what Broadcast Receivers are, how they work, and how they receive and handle system-level broadcasts.

What is a Broadcast Receiver?

A Broadcast Receiver is a component in an Android app that allows the app to listen for and respond to broadcasts. Broadcasts are messages that are sent by the Android system, other apps, or the device itself. They are a mechanism for inter-component communication within the Android system, and they are often used to notify apps about various events or state changes. Broadcast Receivers enable apps to react to these broadcasts, even if the app is not currently running.

Broadcast Receivers can be registered in two ways:

1. **Static Registration**: This is defined in the AndroidManifest.xml file, and it allows the app to receive broadcasts even when the app is not currently running. This is useful for receiving system-level broadcasts that may need to wake up your app or perform a background task.
2. **Dynamic Registration**: This is done programmatically within your app's code, allowing you to register or unregister a Broadcast Receiver during the runtime of your app. Dynamic registration is often used for scenarios where you only need to listen for broadcasts while your app is active.

How Broadcast Receivers Work:

When a broadcast is sent, the Android system examines the registered Broadcast Receivers to determine which components should receive the broadcast. Here's how it works:

1. **Intent**: A broadcast message is sent as an **Intent**. An **Intent** is a simple data structure containing information such as the action to be performed and the data to be used. It specifies what the broadcast is about.
2. **Intent Filters**: Broadcast Receivers are associated with specific **Intent Filters** in the AndroidManifest.xml or programmatically in code. These filters define which broadcasts the Broadcast Receiver should respond to based on the broadcast's action, category, and data. If an Intent matches the filter criteria, the Broadcast Receiver will be triggered.
3. **Broadcast Dispatcher**: When an **Intent** is broadcast, the Android system's Broadcast Dispatcher examines the Intent's action and filters it against the registered Broadcast Receivers. If there's a match, the corresponding Broadcast Receiver's **onReceive()** method is invoked.
4. **onReceive() Method**: The **onReceive()** method is implemented in the Broadcast Receiver class. It is where you specify the code that should be executed when the broadcast is received. This method is called on the main thread, so it's important to keep it lightweight and perform time-consuming tasks in the background.

Handling System-Level Broadcasts:

System-level broadcasts are broadcasts that are sent by the Android system to inform apps about various events and state changes on the device. Some common examples of system-level broadcasts include:

- **Battery low**: Sent when the device's battery is running low.
- **Screen on/off**: Sent when the screen is turned on or off.
- **Network connectivity changes**: Sent when the device's network connectivity status changes.

- **Package installation/uninstallation**: Sent when an app is installed or uninstalled.

To receive and handle system-level broadcasts, you need to define a Broadcast Receiver with an appropriate Intent Filter. Here's a simplified example of how you would handle the "Battery low" broadcast:

Xml code:

```
<receiver android:name=".BatteryReceiver">
<intent-filter>
<action android:name="android.intent.action.BATTERY_LOW" />
</intent-filter>
</receiver>
```

In the **BatteryReceiver** class, you would implement the **onReceive()** method to handle the broadcast:

Java code:

```
public class BatteryReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        // Handle the battery low broadcast here
        // You can perform tasks like showing a notification or taking action to conserve battery.
    }
}
```

This Broadcast Receiver will receive and handle the "Battery low" broadcast sent by the Android system.

Remember that to receive certain system-level broadcasts, your app may require specific permissions, and you should handle these broadcasts responsibly to ensure a positive user experience and optimize the app's performance.

Aim: Develop an Android application that uses Content Providers to share data between different apps and components.

Content Providers are a fundamental component of the Android operating system that allows apps to securely share data with other apps and components. They act as a bridge for inter-process communication and enable structured access to data, ensuring that data is shared in a controlled and secure manner. In this response, I'll explain in detail what Content Providers are and how to use them to share data between different apps and components in Android.

What is a Content Provider?

A Content Provider is a component in Android that provides a consistent and structured way to access, manage, and share data. They are often used for sharing data between different apps or components, such as Activities, Services, and Broadcast Receivers, in a controlled and secure manner. Content Providers are a part of the Android Content Provider framework, and they typically expose data through a content URI (Uniform Resource Identifier).

Some common use cases for Content Providers include:

- Sharing data between apps: Apps can access and manipulate data from other apps, such as contact information, media files, or database records.
- Data access and synchronization: Content Providers are used to access device-specific data, like contacts or call logs.
- Data storage and management: They can be used to store and retrieve data within an app, acting as a data source for an app's user interface.

How to Create and Use a Content Provider:

Here are the steps to create and use a Content Provider for sharing data between different apps and components:

1. Define a Content Provider:

To create a Content Provider, you need to define a class that extends the **ContentProvider** class. This class is responsible for managing access to your data. You will need to override several methods to define how data is queried, inserted,

updated, and deleted.

Java code

```
public class MyContentProvider extends ContentProvider {
 // Implement required methods for data management.
}
```

2. Define a Content URI:

A Content URI is a unique identifier that allows other apps to access your Content Provider. It typically consists of the authority (usually the app's package name) and a path segment that identifies the data.

Java code:

content://com.example.myapp/mydata

3. Implement Data Access Methods:

You need to override methods in your Content Provider to define how data is accessed. Some common methods include:

- **query()**: To retrieve data based on a set of criteria.
- **insert()**: To add new data.
- **update()**: To modify existing data.
- **delete()**: To remove data.

For example, to handle a query request, you would implement the **query()** method:

Java Code:

```
@Override
public Cursor query(Uri uri, String[] projection, String selection, String[] selectionArgs, String sortOrder) {
   // Implement how data is retrieved based on the provided parameters.
   // Return a Cursor with the query results.
}
```

4. Declare the Content Provider in the Manifest:

To make your Content Provider accessible to other apps, you need to declare it in your app's AndroidManifest.xml file.

Xml Code:

```
<provider
   android:name=".MyContentProvider"
   android:authorities="com.example.myapp"
   android:exported="true"
/>
```

The **android:authorities** attribute defines the unique authority for your Content Provider. It should be unique to your app.

5. Access Data from Other Apps:

To access the data exposed by your Content Provider from another app or component, you would use a content resolver. A content resolver is a class that provides a higher-level interface to Content Providers. Here's how you can query data from another app:

Java code:

```
Uri contentUri = Uri.parse("content://com.example.myapp/mydata");

ContentResolver contentResolver = context.getContentResolver();
Cursor cursor = contentResolver.query(contentUri, projection, selection, selectionArgs,
sortOrder);
```

6. Permissions and Security:

Security is a crucial aspect of Content Providers. You can define permissions in your AndroidManifest.xml to control which apps or components can access your Content Provider. You can also enforce permission checks in your Content Provider's methods to ensure secure data access.

Limitations and Considerations:

- Content Providers are not suitable for sharing complex data structures. They are primarily designed for simple data sharing.
- If you intend to share data between your app's own components (e.g., between Activities and Services), consider using other mechanisms like shared preferences or data-binding, as Content Providers are typically used for inter-app communication.
- Ensure that you implement robust error handling in your Content Provider to handle various scenarios, including permission denial.

Aim: Develop an Android application that uses Content Providers to read system-level data, such as contacts and calendar events.

Content Providers are a fundamental component in the Android operating system that allows apps to access and share data, including system-level data such as contacts and calendar events, in a structured and secure manner. Content Providers act as intermediaries for data access, providing a standardized interface for reading, writing, and querying data. In this response, I'll explain Content Providers and how to use them to read system-level data like contacts and calendar events.

What is a Content Provider?

A Content Provider is an Android component that facilitates structured data access and sharing between different apps and components. Content Providers serve as a layer of abstraction that separates the underlying data storage (e.g., databases, files, or network resources) from the requesting app. They provide a consistent way to access data through content URIs, which are similar to URLs for data sources. Content Providers are a core part of the Android Content Provider framework and are used extensively to share and access data.

Using Content Providers to Read System-Level Data:

System-level data, such as contacts and calendar events, is typically stored in Content Providers provided by the Android system itself. To read this data, you need to:

1. Identify the Appropriate Content Provider:

Android provides several system-level Content Providers to access common data sources. For example:

- **Contacts**: The system-level Contacts Provider stores information about contacts, including names, phone numbers, email addresses, etc.
- **Calendar**: The Calendar Provider stores events and scheduling data, including calendar events and reminders.

2. Determine the Required Permissions:

Access to system-level Content Providers often requires specific permissions

declared in your AndroidManifest.xml file. To read contacts, you typically need the **READ_CONTACTS** permission, and for calendar events, you need the **READ_CALENDAR** permission.

3. Use a Content Resolver:

To interact with a Content Provider and access the data it exposes, you need to use a Content Resolver. A Content Resolver is a class that provides a high-level interface to interact with Content Providers.

Java Code :

```
ContentResolver contentResolver = getContentResolver();
```

4. Query the Content Provider:

You can query a Content Provider by specifying a content URI and relevant parameters. For example, to query contacts:

```
Uri contactsUri = ContactsContract.Contacts.CONTENT_URI;

String[] projection = {ContactsContract.Contacts._ID,
ContactsContract.Contacts.DISPLAY_NAME};

String selection = null;

String[] selectionArgs = null;

String sortOrder = null;


Cursor cursor = contentResolver.query(contactsUri, projection, selection,
selectionArgs, sortOrder);
```

- **contactsUri** is the content URI for contacts.
- **projection** specifies the columns you want to retrieve.
- **selection** and **selectionArgs** allow you to filter the results.
- **sortOrder** specifies the sorting order of the results.

5. Iterate Through the Results:

Once you have queried the Content Provider, you can iterate through the cursor to access the retrieved data.

```
if (cursor != null && cursor.moveToFirst()) {
   do {
      String contactName =
cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME)
);
      // Handle the contact data as needed.
   } while (cursor.moveToNext());
   cursor.close();
}
```

6. Handle Permissions and Errors:

Ensure that you have requested and been granted the necessary permissions. Also, handle any exceptions or errors gracefully when interacting with the Content Provider.

Example for Reading Calendar Events:

To read calendar events, you can use the **CalendarContract** Content Provider. Here's a simplified example:

```
Uri eventsUri = CalendarContract.Events.CONTENT_URI;
String[] projection = {CalendarContract.Events.TITLE,
CalendarContract.Events.DTSTART};
String selection = null;
String[] selectionArgs = null;
String sortOrder = null;

Cursor cursor = contentResolver.query(eventsUri, projection, selection, selectionArgs,
sortOrder);

if (cursor != null && cursor.moveToFirst()) {
   do {
      String eventTitle =
cursor.getString(cursor.getColumnIndex(CalendarContract.Events.TITLE));
```

```
        long eventStartTime =
cursor.getLong(cursor.getColumnIndex(CalendarContract.Events.DTSTART));
        // Handle the event data as needed.
    } while (cursor.moveToNext());
    cursor.close();
}
```

In both examples, you query the Content Provider using the appropriate content URI and specify the desired data columns and any relevant filtering criteria.

Remember to handle exceptions, permissions, and errors when working with Content Providers. Ensure that you have the necessary permissions declared in your AndroidManifest.xml and that you request them at runtime when necessary.

- **activity_main.xml:**

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <EditText
        android:id="@+id/nameInput"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter a name"
        android:layout_margin="16dp"/>

    <Button
        android:id="@+id/insertButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Insert"
        android:layout_below="@+id/nameInput"
        android:layout_margin="16dp"/>

    <TextView
        android:id="@+id/resultTextView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/insertButton"
        android:layout_margin="16dp"/>
</RelativeLayout>
```

- **MainActivity.java:**

```
package com.example.practical15;


import android.app.Activity;

import android.content.ContentValues;

import android.database.Cursor;

import android.database.SQLException;

import android.database.sqlite.SQLiteDatabase;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;


public class MainActivity extends Activity {
```

```
DatabaseHelper dbHelper;
SQLiteDatabase database;
EditText nameInput;
Button insertButton;
TextView resultTextView;

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

    dbHelper = new DatabaseHelper(this);
    database = dbHelper.getWritableDatabase();

    nameInput = findViewById(R.id.nameInput);
    insertButton = findViewById(R.id.insertButton);
    resultTextView = findViewById(R.id.resultTextView);

    insertButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            String name = nameInput.getText().toString();
            if (!name.isEmpty()) {
                ContentValues values = new ContentValues();
                values.put(DatabaseHelper.COLUMN_NAME, name);
                database.insert(DatabaseHelper.TABLE_NAME, null, values);
                nameInput.setText(""); // Clear the input field
                displayData();
            }
        }
    });

    displayData();
}
```

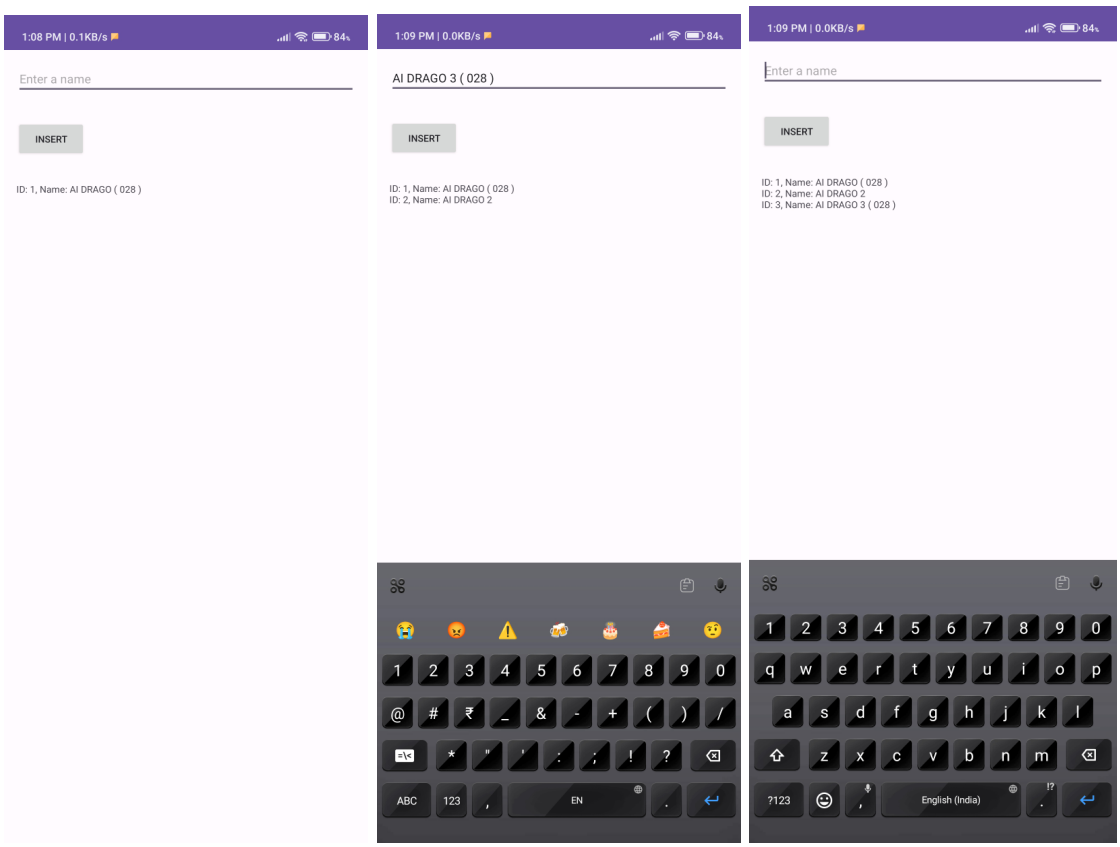
```
    private void displayData() {
        StringBuilder data = new StringBuilder();
        Cursor cursor = database.query(DatabaseHelper.TABLE_NAME, null, null, null,
null, null, null);
        cursor.moveToFirst();
        while (!cursor.isAfterLast()) {
            data.append("ID: ").append(cursor.getLong(0)).append(", Name:
").append(cursor.getString(1)).append("\n");
            cursor.moveToNext();
        }
        cursor.close();
        resultTextView.setText(data.toString());
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        database.close();
    }
}
```

★ DabaseHelper.java:

```
package com.example.practical15;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DatabaseHelper extends SQLiteOpenHelper {

    private static final String DATABASE_NAME = "Practical15";
    private static final int DATABASE_VERSION = 1;
    public static final String TABLE_NAME = "Student";
    public static final String COLUMN_ID = "id";
    public static final String COLUMN_NAME = "name";

    private static final String TABLE_CREATE =
            "CREATE TABLE " + TABLE_NAME + " (" +
                    COLUMN_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
                    COLUMN_NAME + " TEXT);";
```

```
    public DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(TABLE_CREATE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);
    }
}
```

★ Output:



activity_main.xml:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center">
    <EditText
        android:id="@+id/nameInput"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter a name"
        android:layout_margin="16dp"/>
    <Button
        android:id="@+id/insertButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/nameInput"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="16dp"
        android:layout_marginBottom="16dp"
        android:text="Insert"
        android:layout_centerHorizontal="true"/>
    <Button
        android:id="@+id/updateButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Update"
        android:layout_below="@+id/insertButton"
        android:layout_margin="16dp"
        android:layout_centerHorizontal="true"/>
    <Button
        android:id="@+id/deleteButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Delete"
```

```
        android:layout_below="@+id/updateButton"
        android:layout_margin="16dp"
        android:layout_centerHorizontal="true"/>

    <TextView
        android:id="@+id/resultTextView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/deleteButton"
        android:layout_margin="16dp" />
</RelativeLayout>
```

MainActivity.java:

```
package com.example.practical16;

import android.app.Activity;
import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity {

    private DatabaseHelper dbHelper;
    private SQLiteDatabase database;
    private EditText nameInput;
    private Button insertButton;
    private Button updateButton;
    private Button deleteButton;
    private TextView resultTextView;
```

```
String name;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    dbHelper = new DatabaseHelper(this);
    database = dbHelper.getWritableDatabase();
    nameInput = findViewById(R.id.nameInput);
    insertButton = findViewById(R.id.insertButton);
    updateButton = findViewById(R.id.updateButton);
    deleteButton = findViewById(R.id.deleteButton);
    resultTextView = findViewById(R.id.resultTextView);

    insertButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            insertData();
        }
    });
    updateButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            updateData();
        }
    });

    deleteButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            deleteData();
        }
    });

    displayData();
}
```

```
    private void insertData() {
        name = nameInput.getText().toString();
        if (!name.isEmpty()) {
            ContentValues values = new ContentValues();
            values.put(DatabaseHelper.COLUMN_NAME, name);
            long newRowId = database.insert(DatabaseHelper.TABLE_NAME, null, values);
            if (newRowId != -1) {
                nameInput.setText(""); // Clear the input field
                displayData();
            } else {
                Toast.makeText(this, "Insert failed", Toast.LENGTH_SHORT).show();
            }
        } else {
            Toast.makeText(this, "Name cannot be empty", Toast.LENGTH_SHORT).show();
        }
    }
    private void updateData() {
        name = nameInput.getText().toString();
        // Update the first record in the database
        ContentValues values = new ContentValues();
        values.put(DatabaseHelper.COLUMN_NAME,name+" (Updated Name)");
        int rowsAffected = database.update(DatabaseHelper.TABLE_NAME, values, "id=?", new
String[]{"1"});
        if (rowsAffected > 0) {
            displayData();
        } else {
            Toast.makeText(this, "Update failed", Toast.LENGTH_SHORT).show();
        }
    }
    private void deleteData() {
        // Delete the first record in the database
        int rowsDeleted = database.delete(DatabaseHelper.TABLE_NAME, "id=?", new
String[]{"1"});
        if (rowsDeleted > 0) {
            displayData();
        } else {
            Toast.makeText(this, "Delete failed", Toast.LENGTH_SHORT).show();
```

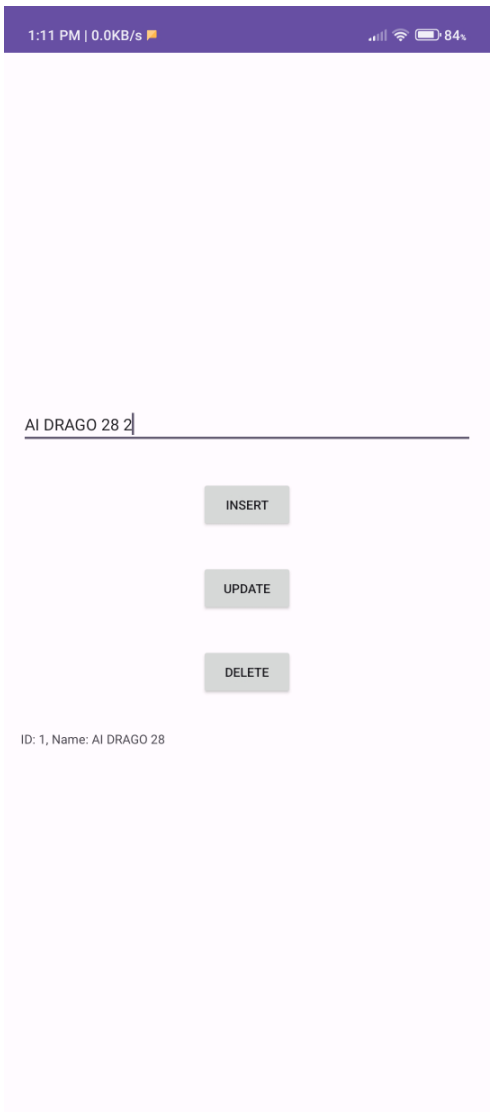
```
        }
    }
    private void displayData() {
        StringBuilder data = new StringBuilder();
        Cursor cursor = database.query(DatabaseHelper.TABLE_NAME, null, null, null, null, null,
null);
        cursor.moveToFirst();
        while (!cursor.isAfterLast()) {
            data.append("ID: ").append(cursor.getLong(0)).append(", Name:
").append(cursor.getString(1)).append("\n");
            cursor.moveToNext();
        }
        cursor.close();
        resultTextView.setText(data.toString());
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        database.close();
    }
}
```

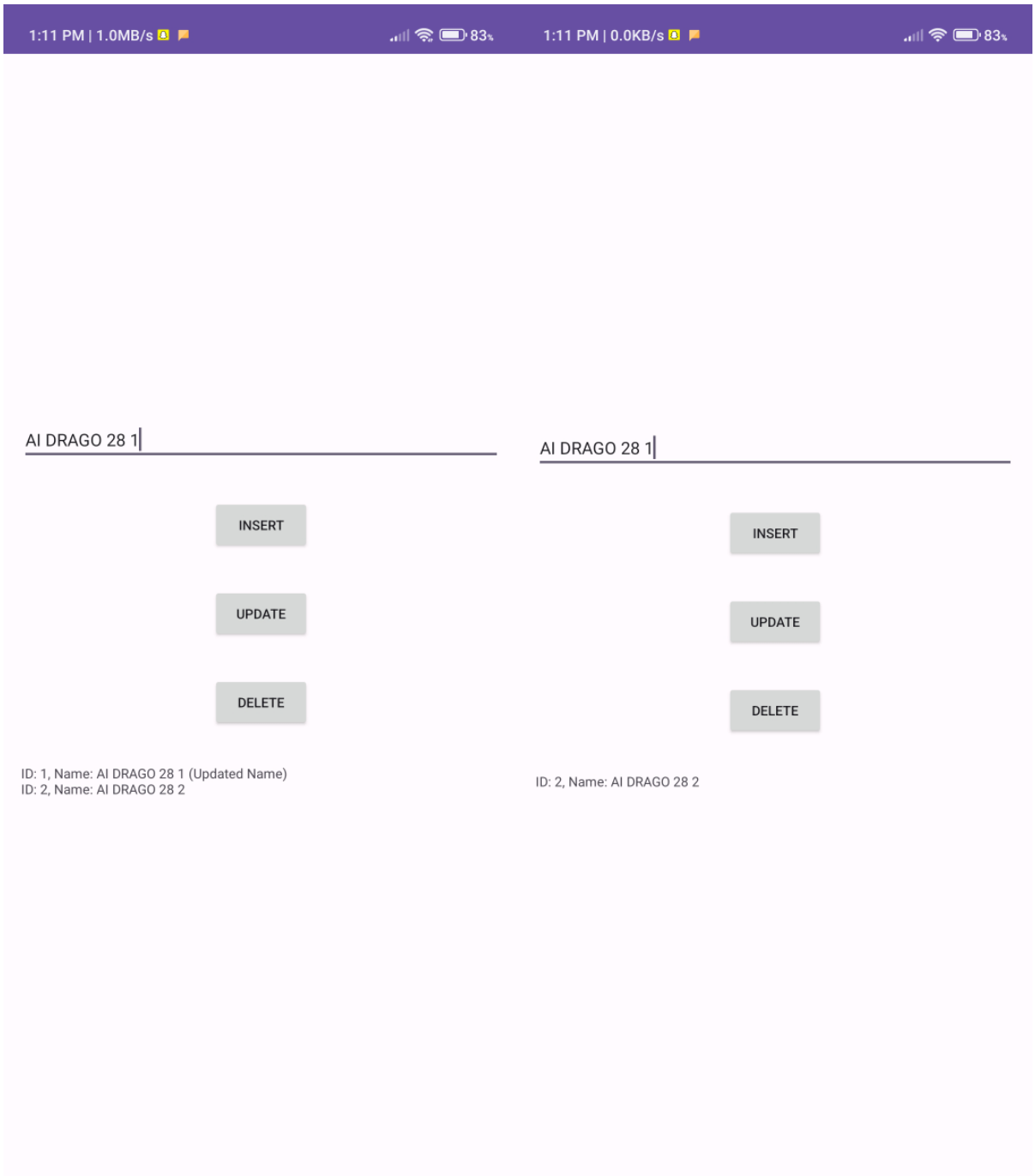
DabaseHelper.java:

```
package com.example.practical16;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DatabaseHelper extends SQLiteOpenHelper {

    private static final String DATABASE_NAME = "Practical15";
    private static final int DATABASE_VERSION = 1;
    public static final String TABLE_NAME = "Student";
    public static final String COLUMN_ID = "id";
    public static final String COLUMN_NAME = "name";

    private static final String TABLE_CREATE =
            "CREATE TABLE " + TABLE_NAME + " (" +
                    COLUMN_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
                    COLUMN_NAME + " TEXT);";

    public DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
```

```
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(TABLE_CREATE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);
    }
}
```

Output:




1:11 PM | 1.0MB/s
83%
AI DRAGO 28 1
INSERT
UPDATE
DELETE
ID: 1, Name: AI DRAGO 28 1 (Updated Name)
ID: 2, Name: AI DRAGO 28 2
1:11 PM | 0.0KB/s
83%
AI DRAGO 28 1
INSERT
UPDATE
DELETE
ID: 2, Name: AI DRAGO 28 2

- **MainActivity. java:**

```
package com.example.p14;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.FirebaseApp;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.HashMap;
import java.util.Map;

public class MainActivity extends AppCompatActivity {
   EditText nameEditText;
   Button storeButton;
   FirebaseFirestore db;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      FirebaseApp.initializeApp(this);
      db = FirebaseFirestore.getInstance();

      nameEditText = findViewById(R.id.nameEditText);
      storeButton = findViewById(R.id.storeButton);

      storeButton.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View view) {
            // Get the user's input from the EditText
            String userName = nameEditText.getText().toString().trim();


            if (!userName.isEmpty()) {
               // Create a new user document with the name and store it in Firestore
               Map<String, Object> user = new HashMap<>();
               user.put("name", userName);
               Log.d("if", userName);

               db.collection("users")
                     .add(user)
                     .addOnSuccessListener(new OnSuccessListener<DocumentReference>() {
```

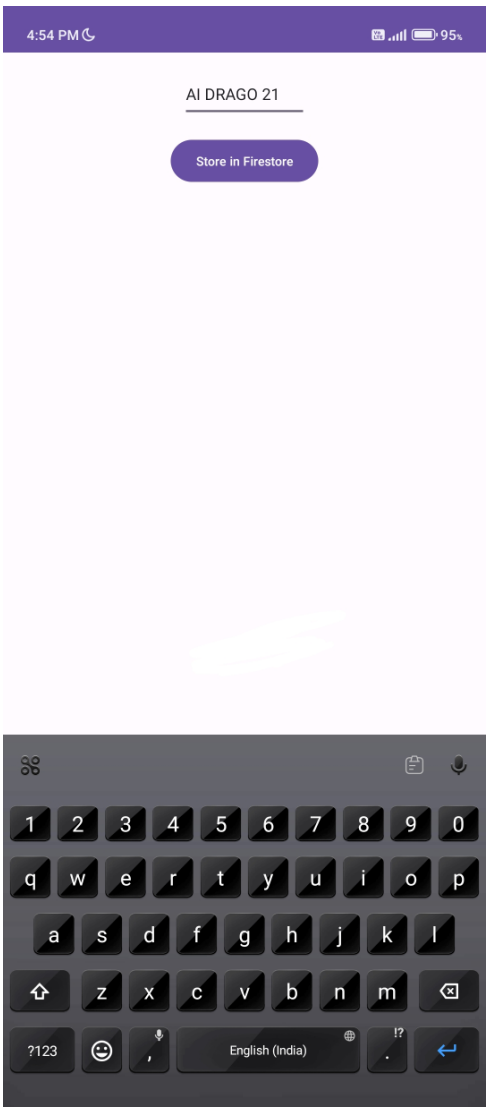
```
                    @Override
                    public void onSuccess(DocumentReference documentReference) {
                        Log.d("success", userName);
                        nameEditText.setText(""); // Clear the EditText
                    }
                })
                .addOnFailureListener(new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception e) {
                        Log.d("fail", userName);
                    }
                });
            }
        }
    });
  }
}
```

- **main_activity.xml:**

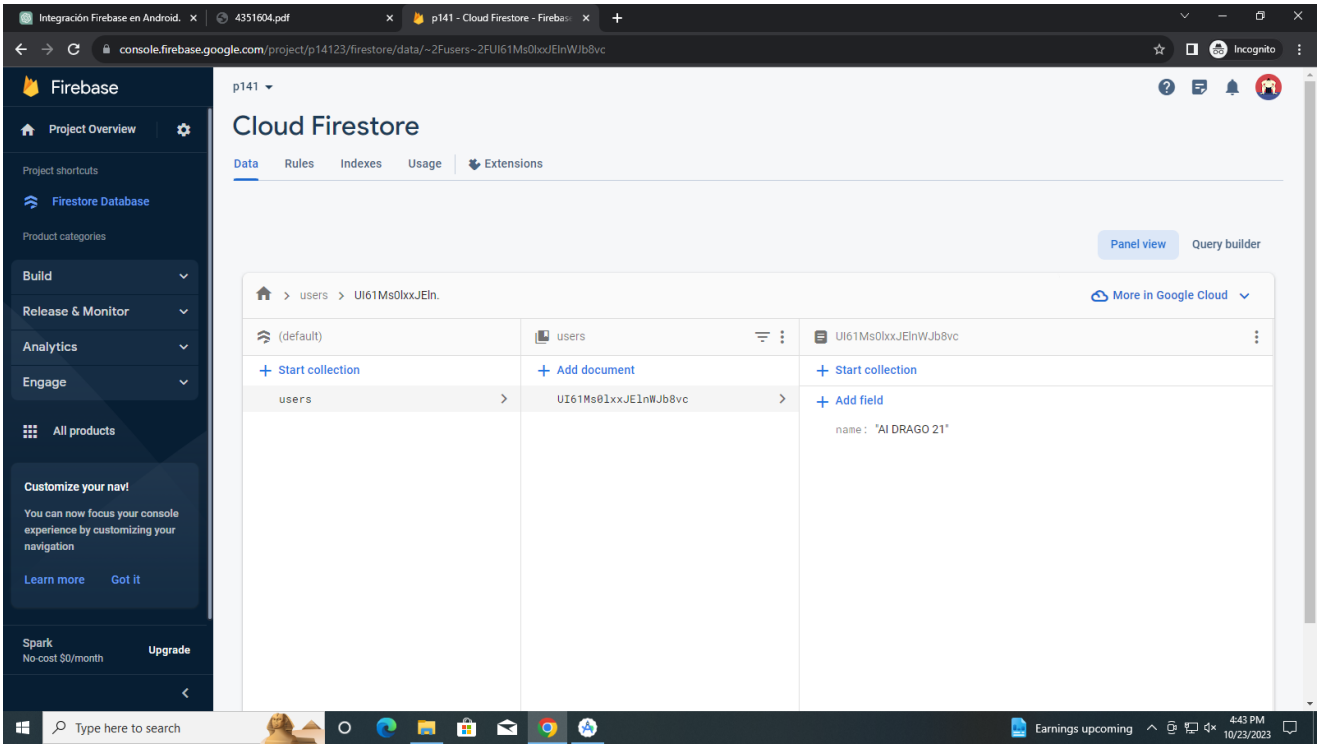
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
 xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity">

   <EditText
      android:id="@+id/nameEditText"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_margin="16dp"
      android:hint="Enter your name"
      android:minHeight="48dp"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent" />

   <Button
      android:id="@+id/storeButton"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Store in Firestore"
      app:layout_constraintTop_toBottomOf="@id/nameEditText"
      app:layout_constraintStart_toStartOf="parent"
```

```
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_margin="16dp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintTop_toBottomOf="@id/storeButton"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintBottom_toBottomOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

- ★ **Output**



- ★ Firebase Database :



- Mainactivity.java

java:package com.example.p14;

```
import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.QueryDocumentSnapshot;
import com.google.firebase.firestore.QuerySnapshot;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class MainActivity extends AppCompatActivity {

    private EditText nameEditText;
    private Button storeButton;
    private RecyclerView recyclerView;
    private UserAdapter userAdapter;
    private List<User> userList;
    private FirebaseFirestore db;

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

        db = FirebaseFirestore.getInstance();
        userList = new ArrayList<>();

        nameEditText = findViewById(R.id.nameEditText);
        storeButton = findViewById(R.id.storeButton);
        recyclerView = findViewById(R.id.recyclerView);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));
        userAdapter = new UserAdapter(userList);
        recyclerView.setAdapter(userAdapter);

        storeButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
```

```
            String userName = nameEditText.getText().toString().trim();
            if (!userName.isEmpty()) {
                Map<String, Object> user = new HashMap<>();
                user.put("name", userName);

                db.collection("users")
                        .add(user)
                        .addOnSuccessListener(new OnSuccessListener<DocumentReference>() {
                            @Override
                            public void onSuccess(DocumentReference documentReference) {
                                nameEditText.setText(""); // Clear the EditText after storing
                                retrieveDataFromFirestore();
                            }
                        })

                        .addOnFailureListener(new OnFailureListener() {
                            @Override
                            public void onFailure(@NonNull Exception e) {
                                // Handle failure
                            }
                        });
            }
        }
    });

    retrieveDataFromFirestore(); // Initial data retrieval from Firestore
}

private void retrieveDataFromFirestore() {
    db.collection("users")
            .get()
            .addOnSuccessListener(new OnSuccessListener<QuerySnapshot>() {
                @Override
                public void onSuccess(QuerySnapshot queryDocumentSnapshots) {
                    userList.clear();
                    for (QueryDocumentSnapshot documentSnapshot : queryDocumentSnapshots) {
                        String userName = documentSnapshot.getString("name");
                        userList.add(new User(userName));
                    }
                    userAdapter.notifyDataSetChanged();
                }
            })
            .addOnFailureListener(new OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception e) {
                    // Handle failure
                }
            });
}
}
```

- **activity_main.xml:**

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

    <EditText
        android:id="@+id/nameEditText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="Enter your name"
        android:minHeight="48dp"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="16dp"/>

    <Button
        android:id="@+id/storeButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Store in Firestore"
        android:layout_below="@id/nameEditText"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="16dp"/>

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/storeButton"
        android:layout_marginTop="16dp"/>
</RelativeLayout>
```

- **User.java:**

```
package com.example.p14;

public class User {
    private String name;
```

```
    public User(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }
}
```

- item_item.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="wrap_content"
    android:orientation="vertical"
    android:padding="16dp">

    <TextView
        android:id="@+id/nameTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="18sp"
        android:textStyle="bold"/>

</LinearLayout>
```

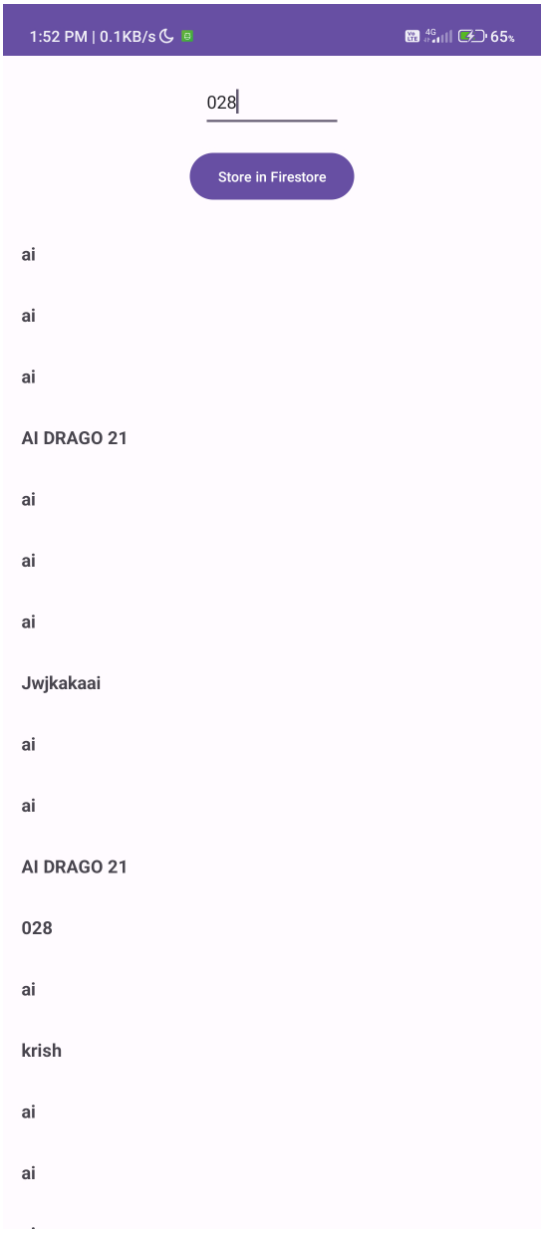
- **userAdapter.java:**

```
package com.example.p14;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
import java.util.List;

class UserAdapter extends RecyclerView.Adapter<UserAdapter.UserViewHolder> {
```

```
    private List<User> userList;

    public UserAdapter(List<User> userList) {
        this.userList = userList;
    }

    @NonNull
    @Override
    public UserViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_user, parent, false);
        return new UserViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull UserViewHolder holder, int position) {
        User user = userList.get(position);
        holder.bind(user);
    }

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

    static class UserViewHolder extends RecyclerView.ViewHolder {
        private TextView nameTextView;

        public UserViewHolder(@NonNull View itemView) {
            super(itemView);
            nameTextView = itemView.findViewById(R.id.nameTextView);
        }

        public void bind(User user) {
```

```
            nameTextView.setText(user.getName());
        }
    }
}
```

★ **Output :**



- **activity_main.xml:**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/edittextData"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="30dp"
        android:hint="Enter Data" />

    <Button
        android:id="@+id/insertButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="16dp"
        android:text="Insert Data" />

    <Button
        android:id="@+id/retrieveButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="16dp"
        android:text="Retrieve Data" />

    <TextView
        android:id="@+id/dataTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text=""
        android:textSize="18sp" />

</LinearLayout>
```

- **MainActivity.java:**

```
package com.example.practical19;

import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
import java.io.BufferedReader;
```

```
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;

public class MainActivity extends AppCompatActivity {

    private EditText data;
    private TextView dataTextView;
    Button insertButton;
    Button retrieveButton;

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

        data = findViewById(R.id.edittextData);
        dataTextView = findViewById(R.id.dataTextView);
        insertButton = findViewById(R.id.insertButton);
        retrieveButton = findViewById(R.id.retrieveButton);

        insertButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Perform data insertion
                new InsertData().execute();
            }
        });

        retrieveButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Perform data retrieval
                new RetrieveData().execute();
            }
        });
    }

    private class InsertData extends AsyncTask<Void, Void, String> {
        @Override
        protected String doInBackground(Void... params) {
            try {


                String url = "http://192.168.43.45/ data_operation.php";
                URL obj = new URL(url);
                HttpURLConnection con = (HttpURLConnection) obj.openConnection();

                con.setRequestMethod("POST");
                con.setDoOutput(true);

                String dataToInsert = data.getText().toString();
                String dataStr = "data=" + URLEncoder.encode(dataToInsert, "UTF-8");
                con.getOutputStream().write(dataStr.getBytes("UTF-8"));
```

```
            int responseCode = con.getResponseCode();

            BufferedReader in = new BufferedReader(new
InputStreamReader(con.getInputStream()));
            String inputLine;
            StringBuilder response = new StringBuilder();

            while ((inputLine = in.readLine()) != null) {
               response.append(inputLine);
            }

            in.close();

            return response.toString();
         } catch (Exception e) {
            e.printStackTrace();
            return "Error: " + e.getMessage();
         }
      }

      @Override
      protected void onPostExecute(String result) {
         dataTextView.setText(result);
      }
   }

   private class RetrieveData extends AsyncTask<Void, Void, String> {
      @Override
      protected String doInBackground(Void... params) {
         try {
            String url = "http://192.168.43.45/ data_operation.php";
            URL obj = new URL(url);
            HttpURLConnection con = (HttpURLConnection) obj.openConnection();

            con.setRequestMethod("GET");

            int responseCode = con.getResponseCode();

            BufferedReader in = new BufferedReader(new
InputStreamReader(con.getInputStream()));
            String inputLine;
            StringBuilder response = new StringBuilder();

            while ((inputLine = in.readLine()) != null) {
               response.append(inputLine);
            }

            in.close();

            return response.toString();
         } catch (Exception e) {
            e.printStackTrace();
            return "Error: " + e.getMessage();
```

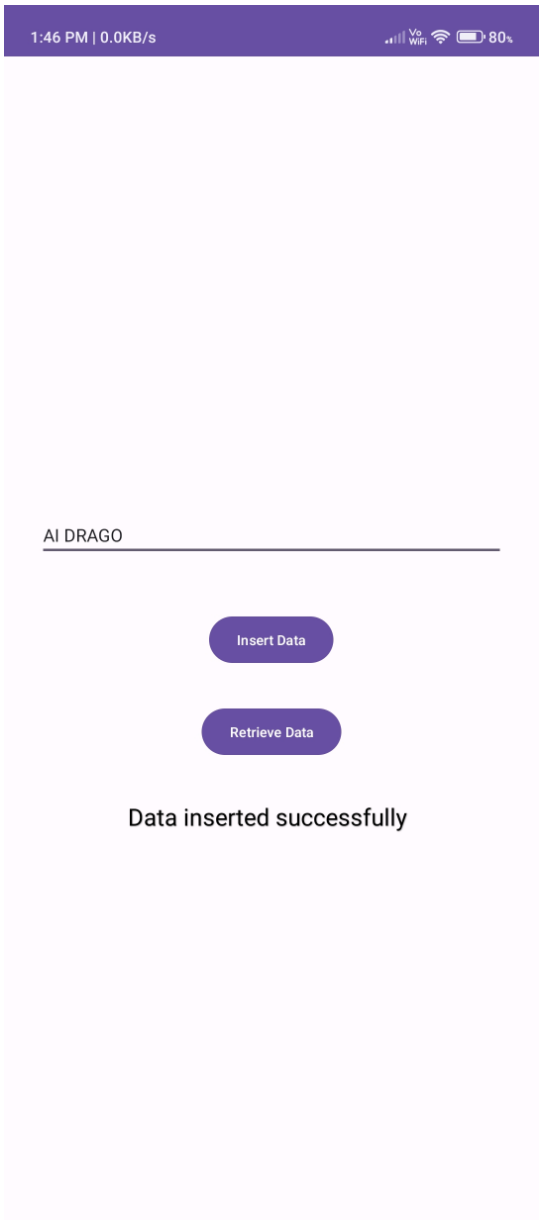
```
            }
        }

        @Override
        protected void onPostExecute(String result) {
            dataTextView.setText(result);
        }
    }
}
```

★ **data_operation.php:**

```
<?php
$dbHost = "localhost";
$dbUser = "root";
$dbPassword = "";
$dbName = "mydb";

// Create a connection to the database
$conn = new mysqli($dbHost, $dbUser, $dbPassword, $dbName);

// Check the connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Data insertion
    $data = $_POST['data'];

    $query = "INSERT INTO Practical19 (data) VALUES (?)";
    $stmt = $conn->prepare($query);
    $stmt->bind_param("s", $data);

    if ($stmt->execute()) {
        echo "Data inserted successfully";
    } else {
        echo "Error: " . $conn->error;
```

```
    }

    $stmt->close();
}
elseif ($_SERVER['REQUEST_METHOD'] === 'GET') {
    // Data retrieval
    $query = "SELECT data FROM Practical19";
    $result = $conn->query($query);

    $data = array();
    while ($row = $result->fetch_assoc()) {
        $data[] = $row['data'];
    }

    header('Content-Type: application/json');
    echo json_encode($data);
}

$conn->close();
?>
```

★ **Output:** Data insert in database and retrieval using PHP in the JSON format.


1:46 PM | 0.0KB/s
80%
Al DRAGO
Insert Data
Retrieve Data
Data inserted successfully


1:48 PM | 0.0KB/s
80%
216120316028
Insert Data
Retrieve Data
["Al DRAGO","216120316028"]

- **activity_main.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="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity">

<Button
    android:id="@+id/mapbtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Click here"
    android:textSize="25sp"/>


</LinearLayout>
```

- **MainActivity.java:**

```
package com.example.Practical22;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {

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

        btn = findViewById(R.id.mapbtn);
        Intent intent = new Intent(MainActivity.this, maps.class);
        startActivity(intent);
    }
}
```

- **activity_maps.xml:**

```
<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
```

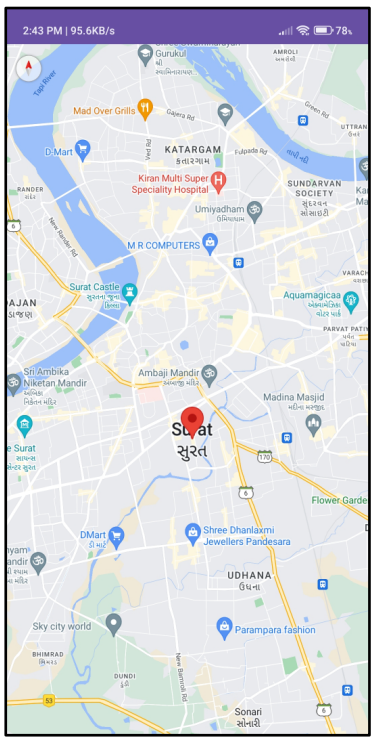
```
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".maps" />
```

★ maps.java:

```
package com.example.Practical22;

import androidx.fragment.app.FragmentActivity;

import android.os.Bundle;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.example.p23.databinding.ActivityMapsBinding;

public class maps extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;
    private ActivityMapsBinding binding;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityMapsBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        // Obtain the SupportMapFragment and get notified when the map is ready to be
```

```
used.
        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        LatLng marker = new LatLng(21.170240, 72.831060);
        mMap.addMarker(new MarkerOptions().position(marker).title("Surat"));
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(marker, 15.0f));
    }
}
```

★ Output:



★ activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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="match_parent"
    tools:context=".MainActivity">

    <fragment
        android:id="@+id/map"
        android:name="com.google.android.gms.maps.SupportMapFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MapsActivity" />

    <EditText
        android:id="@+id/latitudeInput"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="16dp"
        android:layout_above="@+id/longitudeInput"
        android:hint="Enter Latitude"
        android:inputType="numberDecimal" />

    <EditText
        android:id="@+id/longitudeInput"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_above="@+id/calculateDistanceButton"
        android:layout_marginStart="16dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="16dp"
        android:hint="Enter Longitude"
        android:inputType="numberDecimal" />
    <Button
        android:id="@+id/calculateDistanceButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:layout_centerHorizontal="true"
        android:layout_alignParentBottom="true"
        android:text="Calculate Distance" />
</RelativeLayout>
```

★ MainActivity.java:

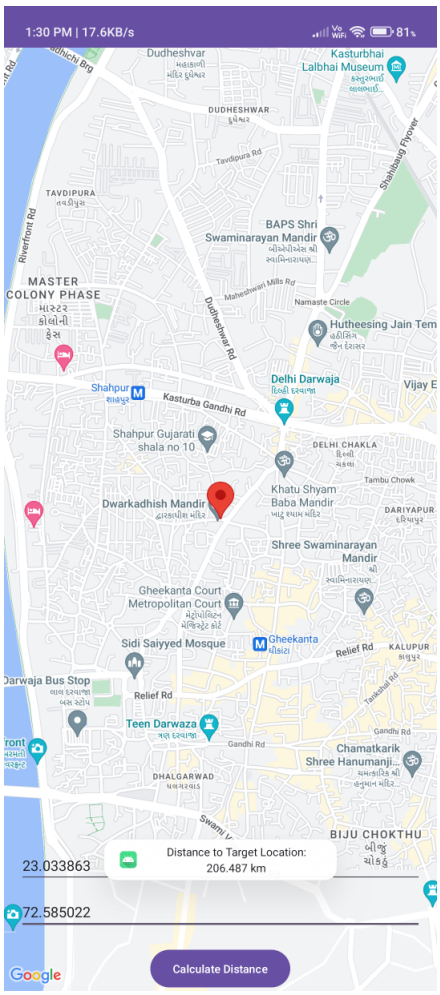
```
package com.example.Practical23;
```

```
import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

public class MainActivity extends AppCompatActivity implements
OnMapReadyCallback {

    private GoogleMap mMap;
    private FusedLocationProviderClient fusedLocationClient;
    private EditText latitudeInput, longitudeInput;
    private LatLng targetLocation;

    private static final int LOCATION_PERMISSION_REQUEST_CODE = 123;

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

        latitudeInput = findViewById(R.id.latitudeInput);
        longitudeInput = findViewById(R.id.longitudeInput);

        Button calculateDistanceButton = findViewById(R.id.calculateDistanceButton);
        calculateDistanceButton.setOnClickListener(view -> calculateDistance());

        fusedLocationClient = LocationServices.getFusedLocationProviderClient(this);

        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        // Add your map initialization code here if needed
    }
```

```
    private void calculateDistance() {
        if (mMap == null) {
            Toast.makeText(this, "Map not yet ready. Please wait.",
Toast.LENGTH_SHORT).show();
            return;
        }

        if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},
LOCATION_PERMISSION_REQUEST_CODE);
            return;
        }

        try {
            double latitude = Double.parseDouble(latitudeInput.getText().toString());
            double longitude = Double.parseDouble(longitudeInput.getText().toString());
            targetLocation = new LatLng(latitude, longitude);

            fusedLocationClient.getLastLocation()
                    .addOnSuccessListener(this, location -> {
                        if (location != null) {
                            LatLng currentLocation = new LatLng(location.getLatitude(),
location.getLongitude());

                            float[] results = new float[1];
                            Location.distanceBetween(
                                    currentLocation.latitude, currentLocation.longitude,
                                    targetLocation.latitude, targetLocation.longitude,
                                    results);

                            double distanceInMeters = results[0];
                            double distanceInKilometers = distanceInMeters / 1000;

                            Toast.makeText(this, "Distance to Target Location: " +
distanceInKilometers + " km", Toast.LENGTH_LONG).show();

                            mMap.clear();
                            mMap.addMarker(new
MarkerOptions().position(targetLocation).title("Target Location"));

mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(targetLocation, 15.0f));
                        } else {
                            Toast.makeText(this, "Location not available. Make sure location
services are enabled.", Toast.LENGTH_SHORT).show();
                        }
                    });
        } catch (NumberFormatException e) {
            Toast.makeText(this, "Invalid latitude or longitude input.",
Toast.LENGTH_SHORT).show();
        }
    }
```

```
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        if (requestCode == LOCATION_PERMISSION_REQUEST_CODE) {
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                calculateDistance();
            } else {
                Toast.makeText(this, "Location permission is required to calculate
distance.", Toast.LENGTH_SHORT).show();
            }
        }
    }
}
```

★ Output:



★ activity_main.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="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity">

    <com.google.android.gms.common.SignInButton
        android:id="@+id/sign_in_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```

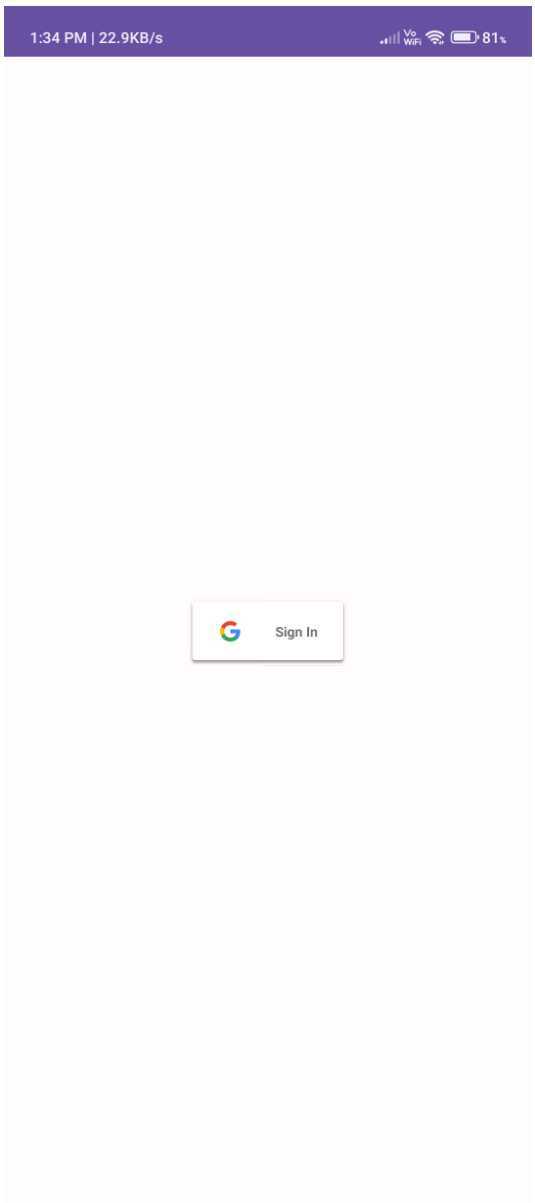
★ MainActivity.java:

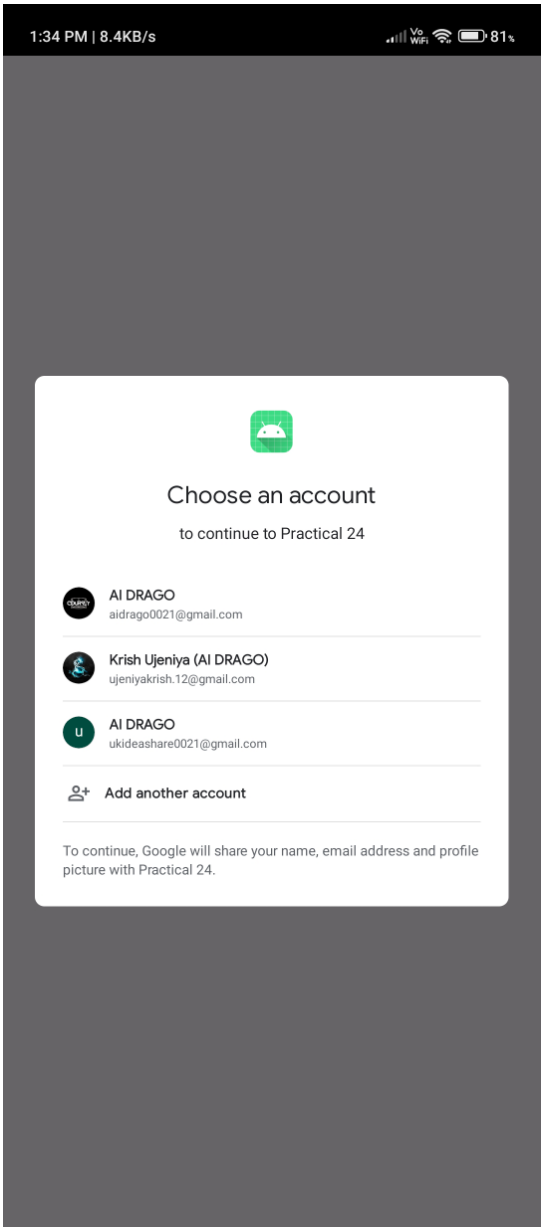
```
package com.example.practical24;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import com.google.android.gms.auth.api.Auth;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;
import com.google.android.gms.auth.api.signin.GoogleSignInOptions;
import com.google.android.gms.auth.api.signin.GoogleSignInResult;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.firebase.auth.AuthCredential;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.auth.GoogleAuthProvider;
import com.google.firebase.firestore.FirebaseFirestore;

public class MainActivity extends AppCompatActivity {

    private static final int RC_SIGN_IN = 9001;
    private FirebaseAuth mAuth;
    private GoogleApiClient mGoogleApiClient;

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

        mAuth = FirebaseAuth.getInstance();

        GoogleSignInOptions gso = new
```

```
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
            .requestIdToken("AIzaSyDxqbMqc7l0siIlu1nMkaPgqp7VB9ypV3Y")
            .requestEmail()
            .build();

        mGoogleApiClient = new GoogleApiClient.Builder(this)
            .addApi(Auth.GOOGLE_SIGN_IN_API, gso)
            .build();

        findViewById(R.id.sign_in_button).setOnClickListener(view ->
signInWithGoogle());
    }

    private void signInWithGoogle() {
        Intent signInIntent = Auth.GoogleSignInApi.getSignInIntent(mGoogleApiClient);
        startActivityForResult(signInIntent, RC_SIGN_IN);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == RC_SIGN_IN) {
            GoogleSignInResult result =
Auth.GoogleSignInApi.getSignInResultFromIntent(data);
            handleSignInResult(result);
        }
    }

    private void handleSignInResult(GoogleSignInResult result) {
        if (result.isSuccess()) {
            GoogleSignInAccount account = result.getSignInAccount();
            firebaseAuthWithGoogle(account);
        } else {
            Toast.makeText(this, "Google Sign-In failed. Please try again.",
Toast.LENGTH_SHORT).show();
        }
    }

    private void firebaseAuthWithGoogle(GoogleSignInAccount account) {
        AuthCredential credential =
GoogleAuthProvider.getCredential(account.getIdToken(), null);
        mAuth.signInWithCredential(credential)
            .addOnCompleteListener(this, task -> {
                if (task.isSuccessful()) {
                    FirebaseUser user = mAuth.getCurrentUser();
                    if (user != null) {
                        String uid = user.getUid();
                        FirebaseFirestore db = FirebaseFirestore.getInstance();
                        Toast.makeText(this, "Login successful",
Toast.LENGTH_SHORT).show();
                    }
                } else {
                    Toast.makeText(this, "Firebase authentication failed.",
Toast.LENGTH_SHORT).show();
```

```
                }
            });
        }
    }
```

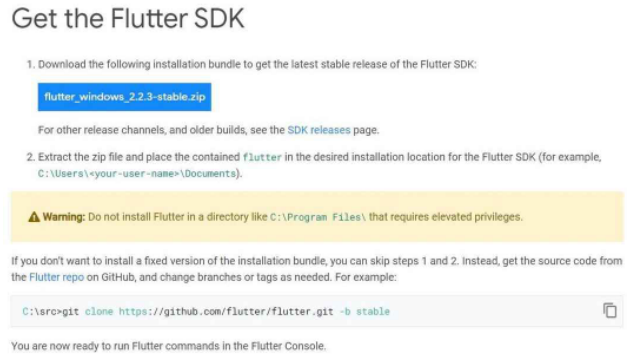
★ Output:


1:34 PM | 22.9KB/s
Sign In

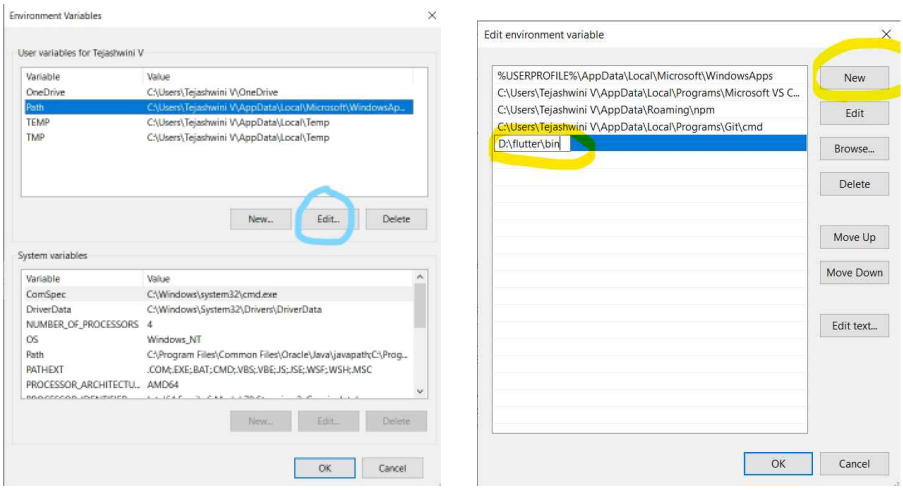

1:34 PM | 8.4KB/s
Choose an account
to continue to Practical 24
AI DRAGO
aidrago0021@gmail.com
Krish Ujeniya (AI DRAGO)
ujeniyakrish.12@gmail.com
AI DRAGO
ukideashare0021@gmail.com
Add another account
To continue, Google will share your name, email address and profile picture with Practical 24.

- Installation Step of Flutter :

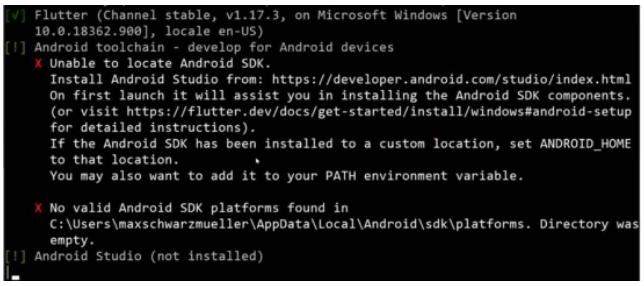
Step 1 – Go to URL, https://flutter.dev/docs/get-started/install/windows and download the latest Flutter SDK. As of April 2019, the version is 1.2.1 and the file is flutter_windows_v1.2.1-stable.zip. Unzip the zip archive in a folder, say C:\flutter\



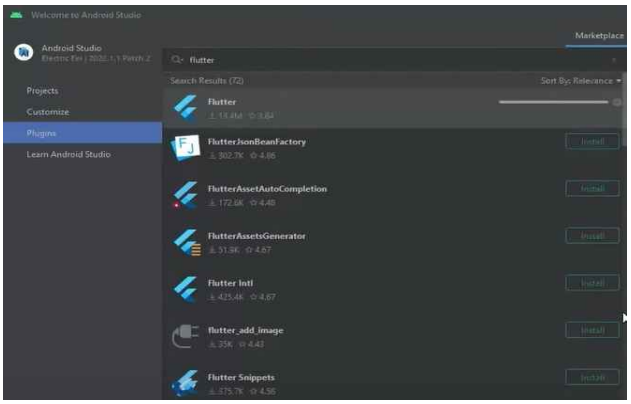
Step 2 – Update the system path to include flutter bin directory.



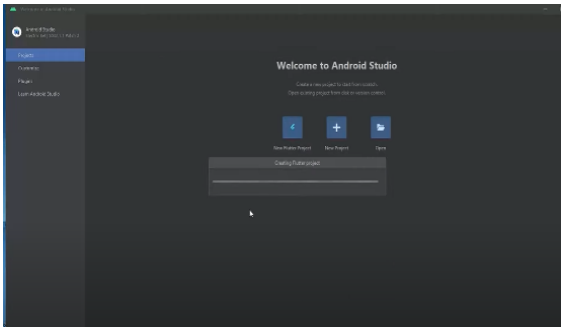
Step 3 – Flutter provides a tool, flutter doctor to check that all the requirement of flutter development is met.

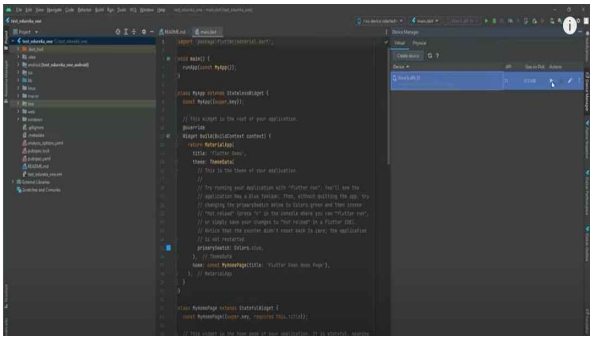


Step 4 – Install Flutter and Dart plugin for Android Studio.



Step 5 - Create New Flutter Project.





- **Sample Hello World program in Flutter :**

★ **main. dart :**

```
import 'package:flutter/material.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const MyHomePage(),
    );
  }
```

```
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key});


  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
        title: Text("Hello World by AI DRAGO(028)"),
      ),
      body: Center(
          child: Text(
              'Hello World',style: TextStyle(fontSize: 40),
            ),
        ),
      );
  }
}
```

★ **Output :**

- ★ main. dart :

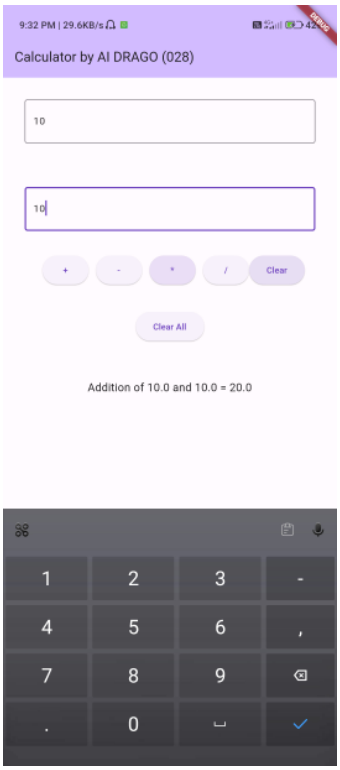
```
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {

    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const MyHomePage(),
    );
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key});

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  TextEditingController firstController = TextEditingController();
  TextEditingController secondController = TextEditingController();
  String calculationResult = '';
  FocusNode firstTextFieldFocus = FocusNode();
  FocusNode secondTextFieldFocus = FocusNode();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
        title: Text(' Calculator by AI DRAGO(028)'),
      ),
```

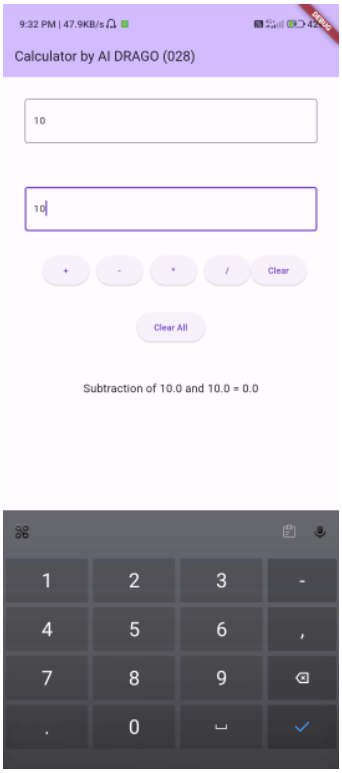
```
      body: SingleChildScrollView(
        child: Column(
          children: [
            Column(
              children: <Widget>[
                Padding(
                  padding: const EdgeInsets.all(30.0),
                  child: Center(
                    child: TextField(
                      controller: firstController,
                      focusNode: firstTextFieldFocus,
                      decoration: InputDecoration(
                        border: OutlineInputBorder(),
                        hintText: 'Enter First Number:-',
                      ),keyboardType: TextInputType.numberWithOptions(decimal: true), // Allow
numeric input
                    ),
                  ),
                ),
                SizedBox(width: 10,),
                Padding(
                  padding: const EdgeInsets.all(30.0),
                  child: Center(
                    child: TextField(
                      controller: secondController,
                      focusNode: secondTextFieldFocus,

                      decoration: InputDecoration(
                        border: OutlineInputBorder(),
                        hintText: 'Enter Second Number:-',
                      ),keyboardType: TextInputType.numberWithOptions(decimal: true),
 // Allow numeric input

                    ),
                  ),
                ),
                ],),

              Row(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  SizedBox(width: 10,),
                  ElevatedButton(
                    onPressed: () {
                      double first = double.tryParse(firstController.text) ?? 0;
                      double second = double.tryParse(secondController.text) ?? 0;
                      String result = calc(first, second, "+");
                      setState(() {
                        calculationResult = result;
                      });
```

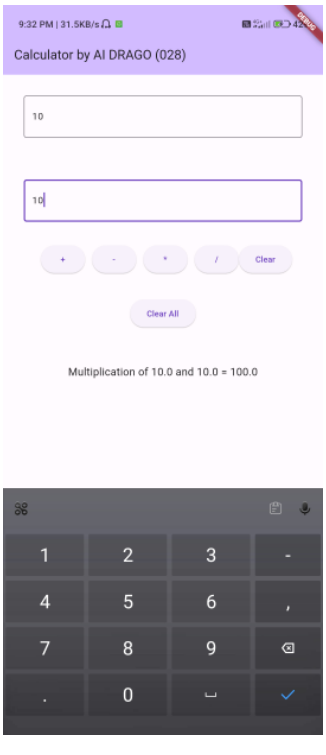
```
  },
  child: Text("+", style: TextStyle(fontSize: 14)),
),
SizedBox(width: 10,),
ElevatedButton(
  onPressed: () {
    double first = double.tryParse(firstController.text) ?? 0;
    double second = double.tryParse(secondController.text) ?? 0;
    String result = calc(first, second, "-");
    setState(() {
      calculationResult = result;
    });
  },
  child: Text("-", style: TextStyle(fontSize: 14)),
),
SizedBox(width: 10,),
ElevatedButton(
  onPressed: () {
    double first = double.tryParse(firstController.text) ?? 0;
    double second = double.tryParse(secondController.text) ?? 0;
    String result = calc(first, second, "*");
    setState(() {
      calculationResult = result;
    });
  },
  child: Text("*", style: TextStyle(fontSize: 14)),
),
SizedBox(width: 10,),
ElevatedButton(
  onPressed: () {
    double first = double.tryParse(firstController.text) ?? 0;
    double second = double.tryParse(secondController.text) ?? 0;
    String result = calc(first, second, "/");
    setState(() {
      calculationResult = result;
    });
  },
  child: Text("/", style: TextStyle(fontSize: 14)),
),
ElevatedButton(
  onPressed: () {
    setState(() {
      if(firstTextFieldFocus.hasFocus){
        firstController.clear();
      }else if(secondTextFieldFocus.hasFocus){
        secondController.clear();
      }
    });
  },
```

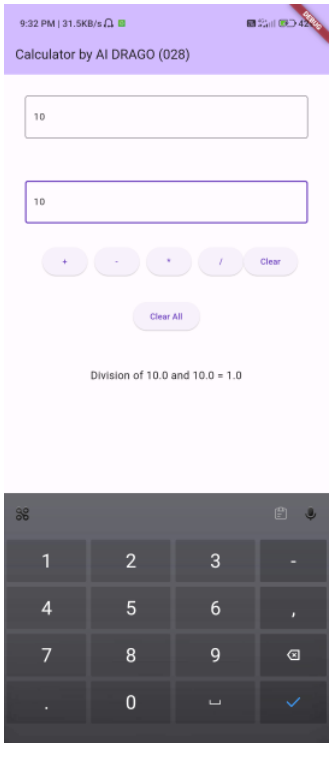
```
              child: Text("Clear", style: TextStyle(fontSize: 14)),
            ),
          ],
        ),
        SizedBox(height: 30,),
        ElevatedButton(
          onPressed: () {
            setState(() {
              firstController.clear();
              secondController.clear();
            });
          },
          child: Text("Clear All", style: TextStyle(fontSize: 14)),
        ),

            SizedBox(height: 50,),
            Center(
              child: Text(
                calculationResult,
                style: TextStyle(fontSize: 18,),
              ),
            ),

      ],
    ),

  ),
 );
}

 String calc(double first_no, double second_no, String operation) {
  if (operation == "+") {
   return "Addition of $first_no and $second_no = ${first_no + second_no}";
  } else if (operation == "-") {
   return "Subtraction of $first_no and $second_no = ${first_no - second_no}";
  } else if (operation == "*") {
   return "Multiplication of $first_no and $second_no = ${first_no * second_no}";
  } else if (operation == "/") {
   return "Division of $first_no and $second_no = ${first_no / second_no}";
  } else {
   return "Invalid Selection!!";
  }
 }
}
```

★ Output :


9:32 PM | 29.6KB/s
Calculator by AI DRAGO (028)
10
10
+
-
*
/
Clear
Clear All
Addition of 10.0 and 10.0 = 20.0


9:32 PM | 47.9KB/s
Calculator by AI DRAGO (028)
10
10
+
-
*
/
Clear
Clear All
Subtraction of 10.0 and 10.0 = 0.0


9:32 PM | 31.5KB/s
Calculator by AI DRAGO (028)
10
10
+
-
*
/
Clear
Clear All
Multiplication of 10.0 and 10.0 = 100.0


9:32 PM | 31.5KB/s
Calculator by AI DRAGO (028)
10
10
+
-
*
/
Clear
Clear All
Division of 10.0 and 10.0 = 1.0

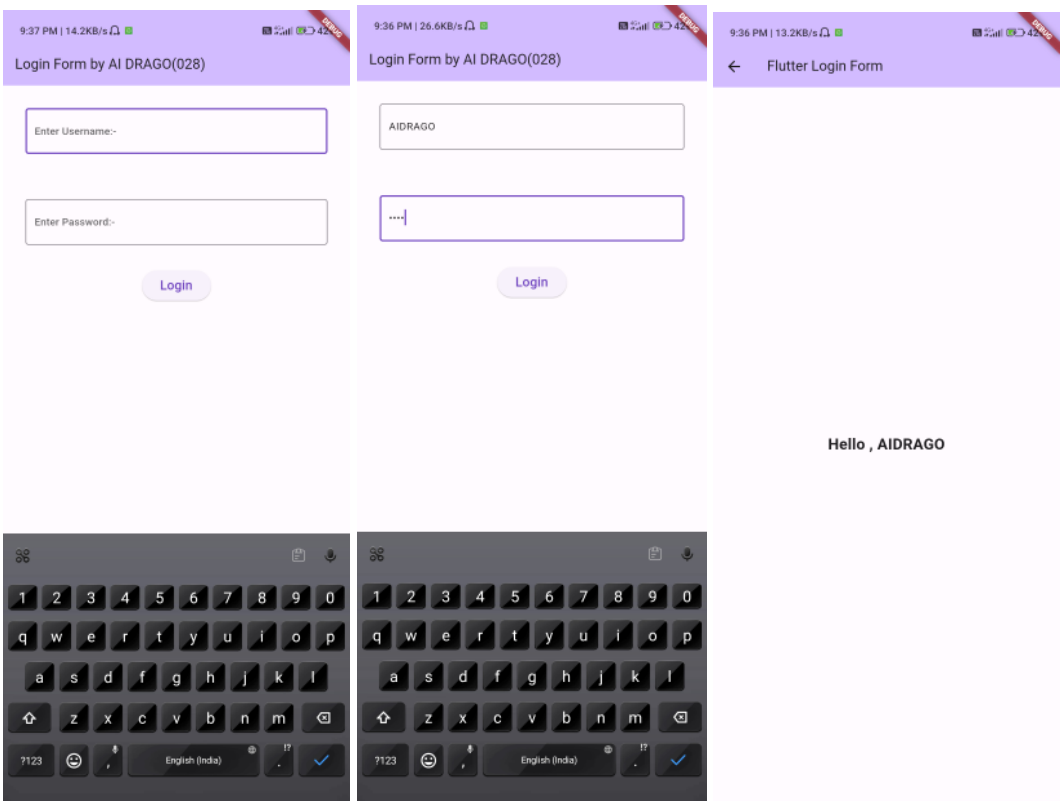
★ main.dart:-

```
import 'dart:ui';

import 'package:flutter/material.dart';
import 'package:flutter/services.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {

    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const MyHomePage(),
    );
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key});

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  TextEditingController userName = TextEditingController();
  TextEditingController passwordController = TextEditingController();
  FocusNode userNameFieldFocus = FocusNode();
  FocusNode passwordTextFieldFocus = FocusNode();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
```

```
      title: Text(' Login Form by AI DRAGO(028)'),
    ),
    body: SingleChildScrollView(
      child: Column(
        children: [
          Padding(
            padding: const EdgeInsets.all(30.0),
            child: Center(
              child: TextField(
                controller: userName,
                focusNode: userNameFieldFocus,
                decoration: InputDecoration(
                  border: OutlineInputBorder(),
                  hintText: 'Enter Username:-',
                ),
              ),
            ),
          ),
          SizedBox(width: 10,),
          Padding(
            padding: const EdgeInsets.all(30.0),
            child: Center(
              child: TextField(
                obscureText: true,
                controller: passwordController,
                focusNode: passwordTextFieldFocus,

                decoration: InputDecoration(
                  border: OutlineInputBorder(),
                  hintText: 'Enter Password:-',
                ),
              ),
            ),
          ),
          ElevatedButton(onPressed: () {
            to_homeScreen(userName.text, passwordController.text);
          },
              child: Text('Login',style: TextStyle(fontSize: 20),)),
        ],
      ),

    ),
  );
}

 void to_homeScreen (String user,String pass){
   if(user=="AIDRAGO" && pass=="1234"){
     Navigator.push(
       context,
```

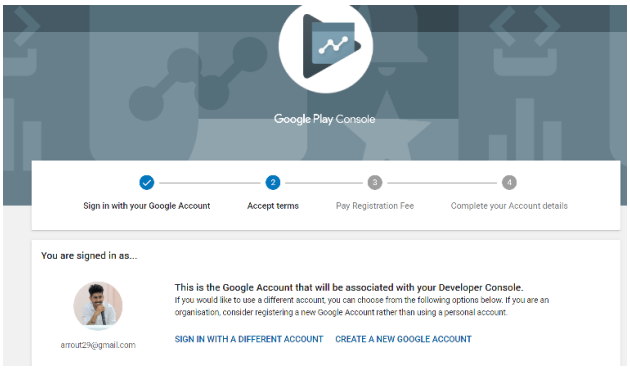
```
      MaterialPageRoute(
        builder: (context) => DisplayDataPage(user), // Pass data here
      ),
    );
   }else{
    showIncorrectCredentialsDialog();
   }
  }
  void showIncorrectCredentialsDialog() {
   showDialog(
    context: context,
    builder: (context) {
     return AlertDialog(
      title: Text('Incorrect Credentials'),
      content: Text('Please check your username and password.'),
      actions: <Widget>[
       TextButton(
        child: Text('OK'),
        onPressed: () {
         Navigator.of(context).pop();
        },
       ),
      ],
     );
    },
   );
  }
}

class DisplayDataPage extends StatelessWidget {
  final String data;

  DisplayDataPage(this.data);

  @override
  Widget build(BuildContext context) {
   return Scaffold(
    appBar: AppBar(
     backgroundColor: Theme.of(context).colorScheme.inversePrimary,
     title: Text('Flutter Login Form'),
    ),
    body: Center(
     child: Text("Hello , ${data}", style: TextStyle(fontSize: 24,fontWeight: FontWeight.bold),), /
/ Display the received data
    ),
   );
  }
}
```

★ **Output :**


9:37 PM | 14.2KB/s
Login Form by AI DRAGO(028)
Enter Username:-
Enter Password:-
Login
9:36 PM | 26.6KB/s
Login Form by AI DRAGO(028)
AIDRAGO
Login
9:36 PM | 13.2KB/s
Flutter Login Form
Hello , AIDRAGO

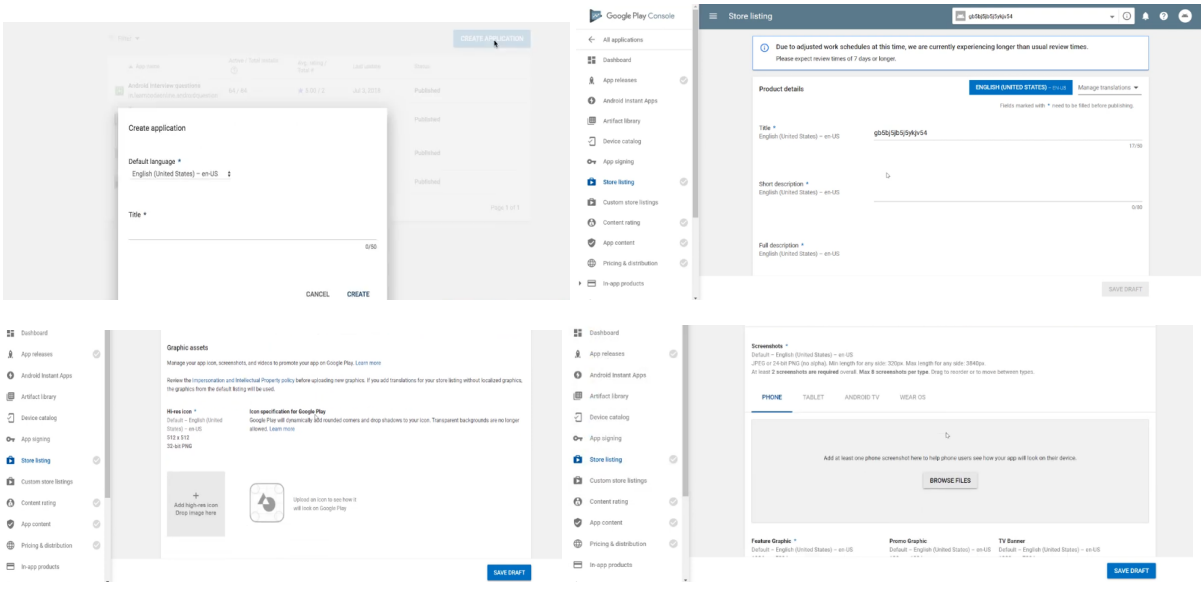
- ★ **Steps to Deploy your apk on play store :**

Step 1: Create a Developer Account on Google Play Console.

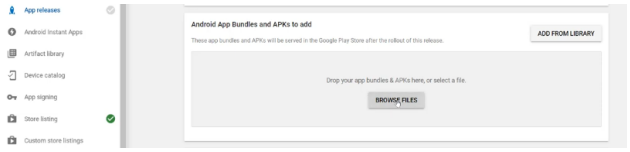


Step 2: Sign in and Click on "Create Application".

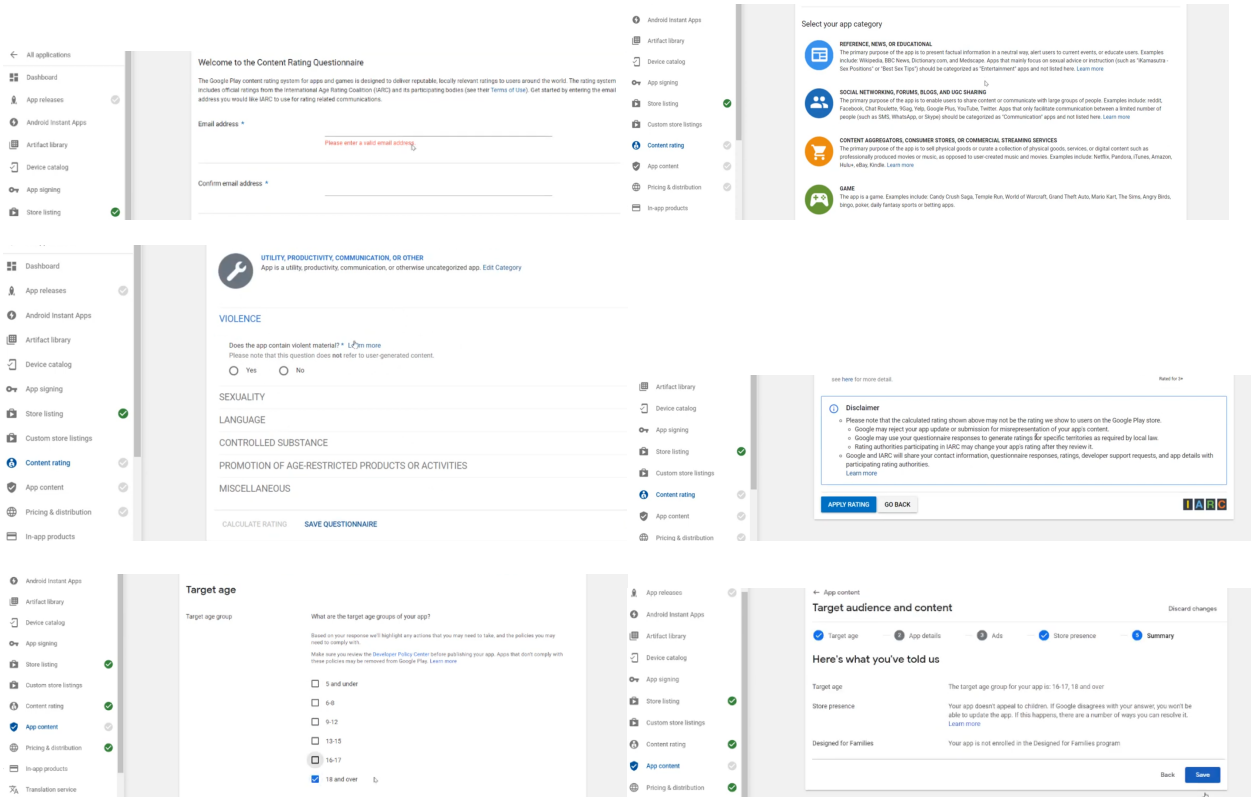
Step 3: Fill App Details (Title, Description, etc.) and Upload Assets (Icons, Screenshots).



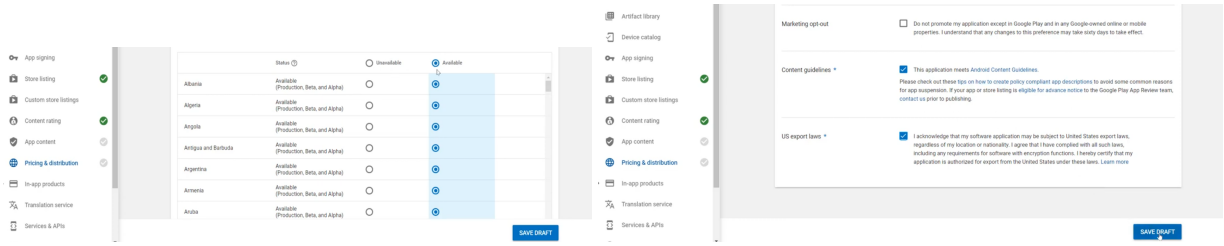
Step 4: Upload Your APK File.



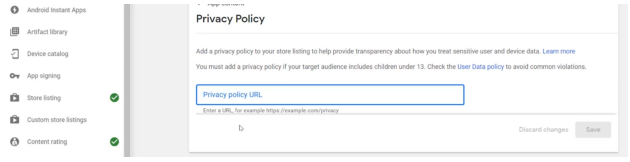
Step 5: Set Content Rating and Target Audience.



Step 6: Choose Pricing & Distribution Options.

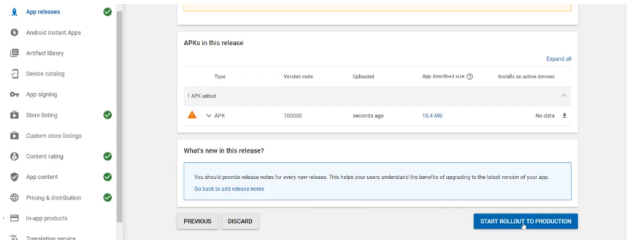


Step 7: Provide Privacy Policy and Submit the App.



Step 8: Wait for Google's Review and Approval.

Step 9: Once Approved, Click "Start Rollout to Production".



Step 10: Your App is Live on Google Play Store!