DMI COLLEGE OF ENGINEERING

PALANCHUR CHENNAI-600123

DEPARTMENT OF INFORMATION TECHNOLOGY

LABORATRY MANUAL

IT 3681-MOBILE APPLICATIONS DEVELOPMENT LABORATORY

YEAR/SEM : III/VI

TABLE OF CONTENTS

Ex.No:1 Study and installation of Flutter/Kotlin multi-platform environment

Introduction

In general, developing a mobile application is a complex and challenging task. There are many frameworks available to develop a mobile application. Android provides a native framework based on Java language and iOS provides a native framework based on Objective-C / Swift language.

However, to develop an application supporting both the OSs, we need to code in two different languages using two different frameworks. To help overcome this complexity, there exists mobile frameworks supporting both OS. These frameworks range from simple HTML based hybrid mobile application framework (which uses HTML for User Interface and JavaScript for application logic) to complex language specific framework (which do the heavy lifting of converting code to native code). Irrespective of their simplicity or complexity, these frameworks always have many disadvantages, one of the main drawback being their slow performance.

In this scenario, Flutter – a simple and high performance framework based on Dart language, provides high performance by rendering the UI directly in the operating system's canvas rather than through native framework.

Flutter also offers many ready to use widgets (UI) to create a modern application. These widgets are optimized for mobile environment and designing the application using widgets is as simple as designing HTML.

To be specific, Flutter application is itself a widget. Flutter widgets also supports animations and gestures. The application logic is based on reactive programming. Widget may optionally have a state. By changing the state of the widget, Flutter will automatically (reactive programming) compare the widget's state (old and new) and render the widget with only the necessary changes instead of re-rendering the whole widget.

We shall discuss the complete architecture in the coming chapters.

Features of Flutter

Flutter framework offers the following features to developers –

- Modern and reactive framework.
- Uses Dart programming language and it is very easy to learn.
- Fast development.
- Beautiful and fluid user interfaces.
- Huge widget catalog.
- Runs same UI for multiple platforms.
- High performance application.

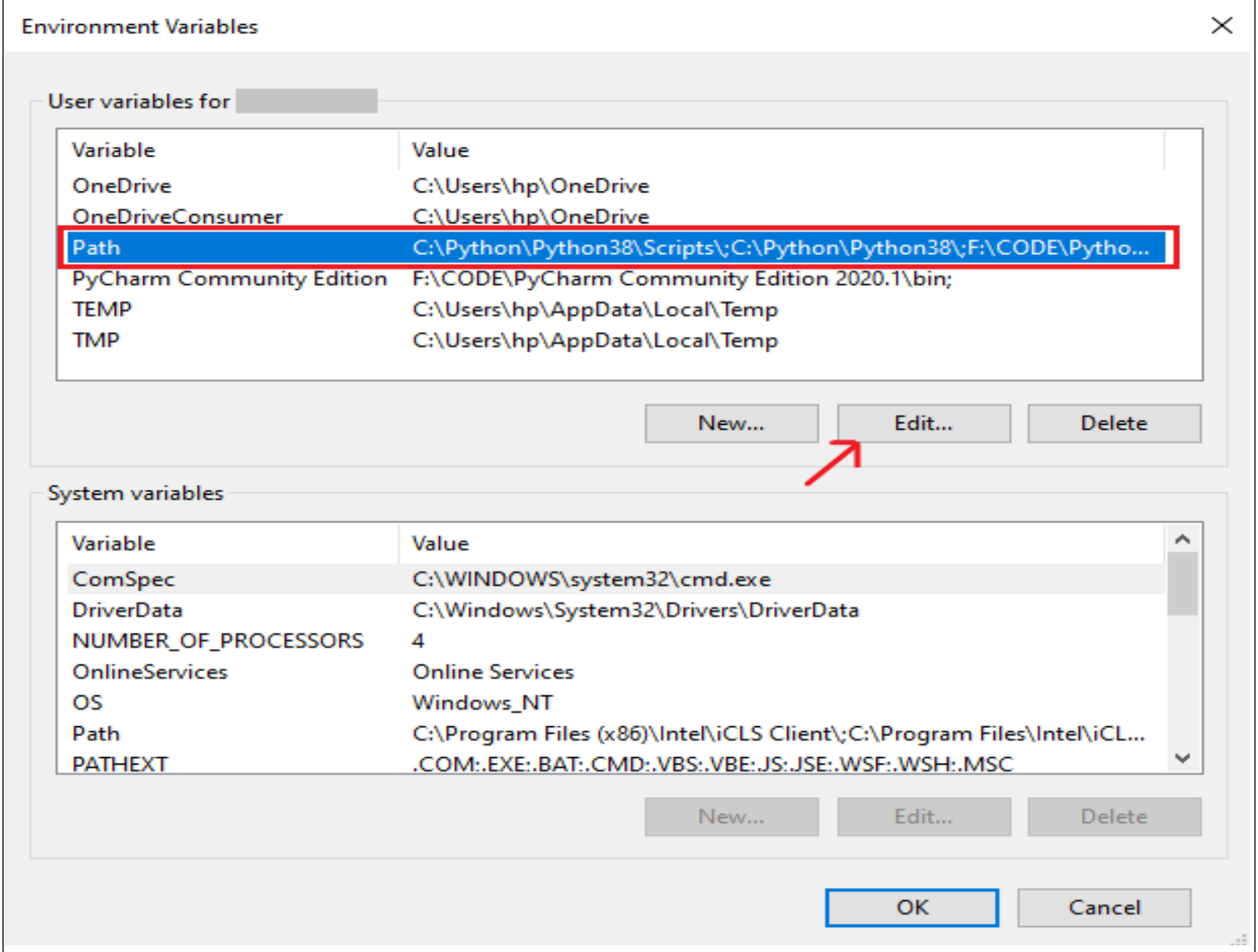
Installation in Windows

In this section, let us see how to install *Flutter SDK* and its requirement in a windows system.

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.

Step 2 – Unzip the zip archive in a folder, say C:\flutter\

Step 3 – Update the system path to include flutter bin directory.



Step 4 – Flutter provides a tool, flutter doctor to check that all the requirement of flutter development is m

Step 5 − Running the above command will analyze the system and show its report as shown below −

```
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, v1.2.1, on Microsoft Windows
[Version 10.0.17134.706], locale en-US)
[√] Android toolchain - develop for Android devices (Android SDK
version 28.0.3)
[√] Android Studio (version 3.2)
[√] VS Code, 64-bit edition
(version 1.29.1) [!] Connected
device
! No devices available
! Doctor found issues in 1 category.
```

The report says that all development tools are available but the device is not connected. We can fix this by connecting an android device through USB or starting an android emulator.

Step 6 − Install the latest Android SDK, if reported by flutter doctor

Step 7 − Install the latest Android Studio, if reported by flutter doctor

Step 8 − Start an android emulator or connect a real android device to the system.

Step 9 − Install Flutter and Dart plugin for Android Studio. It provides startup template to create new Flutter application, an option to run and debug Flutter application in the Android studio itself, etc.,

- Open Android Studio.
- Click File → Settings → Plugins.
- Select the Flutter plugin and click Install.
- Click Yes when prompted to install the Dart plugin.
- Restart Android studio.

Installation in MacOS

To install Flutter on MacOS, you will have to follow the following steps −

Step 1 − Go to URL, https://flutter.dev/docs/get-started/install/macos and download latest Flutter SDK. As of April 2019, the version is 1.2.1 and the file is flutter_macos_v1.2.1- stable.zip.

Step 2 − Unzip the zip archive in a folder, say /path/to/flutter

Step 3 − Update the system path to include flutter bin directory (in ~/.bashrc file).

```
> export PATH = "$PATH:/path/to/flutter/bin"
```

Step 4 − Enable the updated path in the current session using below command and then verify it as well.

```
source ~/.bashrc
```

source
$HOME/.bash_profile
echo $PATH

Flutter provides a tool, flutter doctor to check that all the requirement of flutter development is met. It is similar to the Windows counterpart.

Step 5 − Install latest XCode, if reported by flutter doctor

Step 6 − Install latest Android SDK, if reported by flutter doctor

Step 7 − Install latest Android Studio, if reported by flutter doctor

Step 8 − Start an android emulator or connect a real android device to the system to develop android application.

Step 9 − Open iOS simulator or connect a real iPhone device to the system to develop iOS application.

Step 10 − Install Flutter and Dart plugin for Android Studio. It provides the startup template to create a new Flutter application, option to run and debug Flutter application in the Android studio itself, etc.,

- Open Android Studio
- Click **Preferences → Plugins**
- Select the Flutter plugin and click Install
- Click Yes when prompted to install the Dart plugin.
- Restart Android studio.

Result:

Thus, installation of Flutter/Kotlin multi-platform environment successfully completed.

Ex.No:2 Develop an application that uses Widgets, GUI components, Fonts, and Colors.

Aim:

To develop a Simple Android Application that uses GUI components, Font and Colors.

Procedure:

1.Set Up Android Studio:

Download and install Android Studio.Set up your development environment by installing necessary SDKs and tools.

2.Design User Interface (UI):

Use Android Studio's layout editor to design your app's UI.Drag and drop widgets like TextView, EditText, Button, etc., onto your layout.Customize layout using constraints or other layout types.

3.Customize Fonts and Colors:

Add custom font files to the res/font directory.Define font families in res/font/font_family.xml.Define colors in res/values/colors.xml.

4.Implement Functionality:

Write the logic for your app's functionality in the associated Java/Kotlin file.Set up event listeners for user interactions.Handle these interactions and perform desired actions.

5.Test and Deploy:

Test your app on different devices using Android Virtual Device (AVD) or physical devices.Debug and fix any issues.Once ready, deploy your app to the Google Play Store or other distribution channels.

GUI.dart

```
import 'package:flutter/material.dart';
class GUI extends StatefulWidget {
  const GUI({super.key});

  @override
  State<GUI> createState() => _GUIState();
}
class _GUIState extends State<GUI> {
  TextEditingController textfield1Controller=TextEditingController();
  Color customTextColor=Colors.black;
  double customFontSize=14;
  FontWeight customFontWeight=FontWeight.normal;
  bool isBold=false;
  @override
  void initState() {
    super.initState();
    // TODO: implement initState
    textfield1Controller.text="Sample Text";
  }
  @override
  Widget build(BuildContext context) {
```

```
    return  Scaffold(
      appBar: AppBar(title: const Text("GUI Example"),centerTitle: true,),
      body:  SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.only(left:20,right:20),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              // Output
              const SizedBox(height: 30,),
              Container(
               width: double.infinity,
               decoration: BoxDecoration(
                 borderRadius: BorderRadius.circular(5),
                 border: Border.all(color: Colors.green,width: 2)
               ),
               child: Padding(
                 padding: const EdgeInsets.all(8.0),
                 child: Column(
                   mainAxisSize: MainAxisSize.min,
                   crossAxisAlignment: CrossAxisAlignment.start,
                   children: [
                     const Text("Output",style: TextStyle(color: Colors.green,fontWeight: FontWeight.w500)),
                     const SizedBox(height:5,),
                     Text(textfield1Controller.text,style: TextStyle(color: customTextColor,fontSize:
customFontSize,fontWeight: customFontWeight),),
                     const SizedBox(height: 30,),
                   ],

                 ),
               ),
             ),

              const SizedBox(height: 30,),

              // Input
              Container(
                width: double.infinity,
                decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(5),
                    border: Border.all(color: Colors.blue,width: 2)
                ),
                child: Padding(
                  padding: const EdgeInsets.all(8.0),
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    mainAxisSize: MainAxisSize.min,
                    children: [
                      const Text("Input",style: TextStyle(color: Colors.blue,fontWeight: FontWeight.w500),),
                      const SizedBox(height:5,),
                      Container(
                          height: 150,
```

```
child: Padding(
                    padding: const EdgeInsets.all(5),
                    child: TextField(
                      controller: textfield1Controller,
                      style: TextStyle(fontSize: 14),
                      decoration: InputDecoration(border: InputBorder.none,),
                      onChanged: (value) {
                        setState(() {

                        });
                      },
                                    ),
                  )),
              const SizedBox(height: 10,),

            ],
          ),
        ),
      ),
      const SizedBox(height: 30,),

      // Operation
      Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Expanded(
            flex:1,
            child: Container(

                child: Text("Selected Font Color",style: TextStyle(color: Colors.black, fontSize: 13),)),
          ),
                      Expanded(
            flex:1,
            child: Container(
              decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(5),
                border: Border.all(color:Colors.black),
              ),
              width: 160,
              height: 45,
              child: Center(
                child: DropdownButton<Color>(

                  style: TextStyle(fontWeight: FontWeight.normal,fontSize: 12,color: Colors.black, ),
                  dropdownColor: Colors.white,

                  iconSize: 0,
                  value: customTextColor,
                  onChanged: (Color? newValue) {
                    setState(() {
                      customTextColor = newValue!;
                    });
```

```
  },

underline: Container(
                    color: Colors.white,
                  ),
                  alignment: Alignment.centerRight,
                  items: [
                    DropdownMenuItem(value: Colors.black,child: Center(child: Text("Black"))),
                    DropdownMenuItem(value: Colors.red,child: Center(child: Text("Red"))),
                    DropdownMenuItem(value: Colors.blue,child: Center(child: Text("Blue"))),
                    DropdownMenuItem(value: Colors.green,child: Center(child: Text("Green"))),
                    DropdownMenuItem(value: Colors.yellow,child: Center(child: Text("Yellow"))),
                    DropdownMenuItem(value: Colors.orange,child: Center(child: Text("Orange"))),
                    DropdownMenuItem(value: Colors.pink,child: Center(child: Text("Pink"))),

                  ],
                ),
              ),
            ),
          ),
        ],
      ),
      const SizedBox(height: 15,),
      Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Expanded(
            flex:1,
            child: Container(

                child: Text("Selected Font Size",style: TextStyle(color: Colors.black, fontSize: 13),)),
          ),

          Expanded(
            flex:1,
            child: Container(
              decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(5),
                border: Border.all(color:Colors.black),
              ),
              width: 160,
              height: 45,
              child: Center(
                child: DropdownButton<double>(

                  style: TextStyle(fontWeight: FontWeight.normal,fontSize: 12,color: Colors.black, ),
                  dropdownColor: Colors.white,

                  iconSize: 0,
                  value: customFontSize,
                  onChanged: (double? newValue) {
                    setState(() {
                      customFontSize = newValue!;
```

```
        });
                  },
                  underline: Container(
                    color: Colors.white,
                  ),

alignment: Alignment.centerRight,
                  items: [
                    DropdownMenuItem(value: 10.00,child: Center(child: Text("10"))),
                    DropdownMenuItem(value: 12.00,child: Center(child: Text("12"))),
                    DropdownMenuItem(value: 14.00,child: Center(child: Text("14"))),
                    DropdownMenuItem(value: 16.00,child: Center(child: Text("16"))),
                    DropdownMenuItem(value: 18.00,child: Center(child: Text("18"))),
                    DropdownMenuItem(value: 20.00,child: Center(child: Text("20"))),
                    DropdownMenuItem(value: 30.00,child: Center(child: Text("30"))),

                  ],
                ),
              ),
            ),
          ),
        ],
      ),
      const SizedBox(height: 15,),
      Row(
        children: [
          Expanded(
            flex: 1,
            child: const Text("BOLD"),
          ),
          Expanded(
            flex: 1,
            child: Switch(
              activeColor: Colors.black,
              hoverColor: Colors.black,
              value: isBold,
              onChanged: (value) {
                isBold = value;

                if(isBold==true){
                  customFontWeight=FontWeight.bold;
                }
                else{
                  customFontWeight=FontWeight.normal;
                }
                setState(() {


                });
              },
            ),
          ),
        ],
```

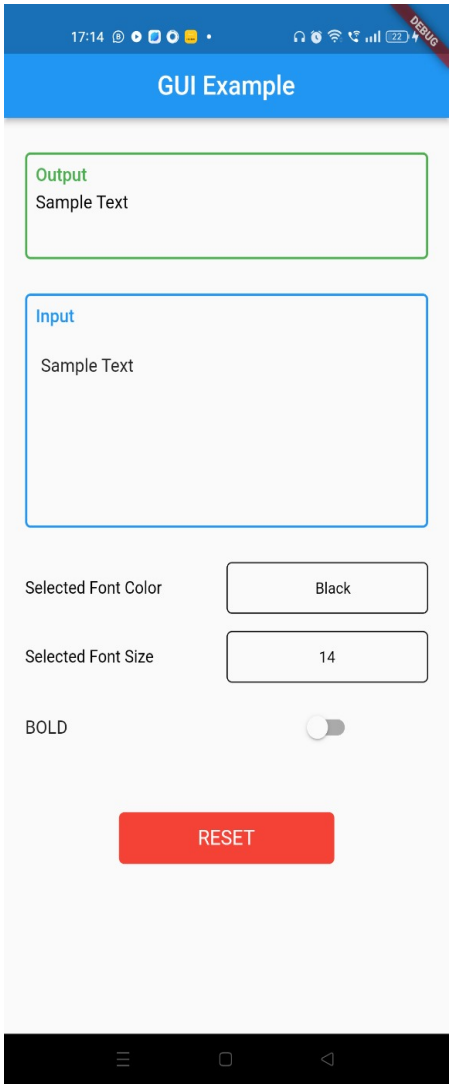
```
    ),
          const SizedBox(height: 50,),


/Submit
          Center(
            child: InkWell(
              onTap: (){
                customFontWeight=FontWeight.normal;
                customFontSize=14;
                textfield1Controller.text="Sample Text";
                customTextColor=Colors.black;
                setState(() {

                });
              },
              child:Container(
                width:205,
                height: 45,

                decoration: BoxDecoration(
                    color: Colors.red,
                  borderRadius: BorderRadius.circular(5)
                ),
                child: Center(
                  child: Text(
                    "RESET",style: TextStyle(color: Colors.white,fontSize: 16),
                  ),
                ),
              )
              ),
            ),
            const SizedBox(height: 30,),
          ],
        ),
      ),
    ),
  );
 }
}
```

Main.dart

```
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart' show Firebase;
import 'package:flutter/services.dart';
import 'package:textrecognition/EarnPointsFragment.dart';
import 'package:textrecognition/GUI.dart';

Future<void> main() async{
  // WidgetsFlutterBinding.ensureInitialized();
  // await SystemChrome.setPreferredOrientations([DeviceOrientation.portraitUp]);
  // // await Firebase.initializeApp();

  runApp(const MyApp());
```

```
}

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

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Text Recognizer',


   home:  GUI(),
    );
  }
}
```

OUTPUT:



Result:

Thus a Simple Android Application that uses GUI components, Font and Colors is developed and executed successfully.

Ex.No:3 Develop a native calculator application.

Aim:

To develop a Simple Android Application for Native Calculator.

Procedure:

1.Set Up Android Studio:

Download and install Android Studio.Set up your development environment by installing necessary SDKs and tools.

2.Design User Interface (UI):

Use Android Studio's layout editor to design your app's UI.Drag and drop widgets like TextView, EditText, Button, etc., onto your layout.Customize layout using constraints or other layout types.

3.Customize Fonts and Colors:

Add custom font files to the res/font directory.Define font families in res/font/font_family.xml.Define colors in res/values/colors.xml.

4.Implement Functionality:

Write the logic for your app's functionality in the associated Java/Kotlin file.Set up event listeners for user interactions.Handle these interactions and perform desired actions.

5.Test and Deploy:

Test your app on different devices using Android Virtual Device (AVD) or physical devices.Debug and fix any issues.Once ready, deploy your app to the Google Play Store or other distribution channels.

Program

Main.dart

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

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

class MyApp extends StatelessWidget {
@override
Widget build(BuildContext context) {
        return MaterialApp(
        debugShowCheckedModeBanner: false,
        home: HomePage(),
        ); // MaterialApp
}
}

class HomePage extends StatefulWidget {
@override
```

```
_HomePageState createState() => _HomePageState();


}

class _HomePageState extends State<HomePage> {
var userInput = '';
var answer = '';

// Array of button
final List<String> buttons = [
        'C',
        '+/-',
        '%',
        'DEL',
        '7',
        '8',
        '9',
        '/',
        '4',
        '5',
        '6',
        'x',
        '1',
        '2',
        '3',


        '-',
        '0',

        ' ',
        '.',
        '=',
        '+',
];

@override
Widget build(BuildContext context) {
        return Scaffold(
        appBar: new AppBar(
                title: new Text("Calculator"),
        ), //AppBar
        backgroundColor: Colors.white38,
        body: Column(
                children: <Widget>[
                Expanded(
                        child: Container(
                        child: Column(
                                mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                                children: <Widget>[
                                        Container(
                                        padding: EdgeInsets.all(20),
                                        alignment: Alignment.centerRight,
                                        child: Text(
                                                userInput,
                                                style: TextStyle(fontSize: 18, color: Colors.white),
                                        ),
                                        ),
```

```
Container(
                                        padding: EdgeInsets.all(15),
                                        alignment: Alignment.centerRight,


child: Text(
                                                answer,
                                                style: TextStyle(
                                                        fontSize: 30,
                                                        color: Colors.white,
                                                        fontWeight: FontWeight.bold),
                                        ),
                                        )
                                ]),
                        ),
                ),
                Expanded(
                        flex: 3,
                        child: Container(
                        child: GridView.builder(
                                itemCount: buttons.length,
                                gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
                                        crossAxisCount: 4),
                                itemBuilder: (BuildContext context, int index) {
                                        // Clear Button
                                        if (index == 0) {
                                        return MyButton(
                                                buttontapped: () {
                                                setState(() {



                                                        userInput = '';
                                                          answer = '0';
                                                });
                                                },




                                                buttonText: buttons[index],
                                                color: Colors.blue[50],
                                                textColor: Colors.black,
                                        );
                                        }

                                        // +/- button
                                        else if (index == 1) {
                                        return MyButton(
                                                buttonText: buttons[index],
                                                color: Colors.blue[50],
                                                textColor: Colors.black,
                                        );
                                        }
                                        // % Button
                                        else if (index == 2) {
                                        return MyButton(
                                                buttontapped: () {
```

```
                    setState(() {
                        userInput += buttons[index];
                    });
                    },
                    buttonText: buttons[index],


                      color: Colors.blue[50],
                    textColor: Colors.black,
                );
                }
                // Delete Button
                else if (index == 3) {
                return MyButton(
                    buttontapped: () {
                    setState(() {
                        userInput =
                            userInput.substring(0, userInput.length - 1);
                    });
                    },
                    buttonText: buttons[index],
                    color: Colors.blue[50],
                    textColor: Colors.black,
                );
                }
                // Equal_to Button
                else if (index == 18) {
                return MyButton(
                    buttontapped: () {
                    setState(() {
                        equalPressed();
                    });
                    },


                  buttonText: buttons[index],
                    color: Colors.orange[700],
                    textColor: Colors.white,
                );
                }




                // other buttons
                else {
                return MyButton(
                    buttontapped: () {
                    setState(() {
                        userInput += buttons[index];
                    });
                    },
                    buttonText: buttons[index],
                    color: isOperator(buttons[index])
                        ? Colors.blueAccent
                        : Colors.white,
```

```
                                        textColor: isOperator(buttons[index])
                                                ? Colors.white
                                                : Colors.black,
                                        );
                                        }
                                }), // GridView.builder
                        ),


),

],
        ),
        );
}

bool isOperator(String x) {
        if (x == '/' || x == 'x' || x == '-' || x == '+' || x == '=') {
        return true;
        }
        return false;
}

// function to calculate the input operation
void equalPressed() {
        String finaluserinput = userInput;
        finaluserinput = userInput.replaceAll('x', '*');

        Parser p = Parser();
        Expression exp = p.parse(finaluserinput);
        ContextModel cm = ContextModel();
        double eval = exp.evaluate(EvaluationType.REAL, cm);
        answer = eval.toString();
}
}
```

Button.dart

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



/ creating Stateless Widget for buttons
class MyButton extends StatelessWidget {

// declaring variables
final color;
final textColor;
final String buttonText;
final buttontapped;



//Constructor
MyButton({this.color, this.textColor, this.buttonText, this.buttontapped});

@override
Widget build(BuildContext context) {
        return GestureDetector(
```

```
onTap: buttontapped,
        child: Padding(
                padding: const EdgeInsets.all(0.2),
                child: ClipRRect(
                // borderRadius: BorderRadius.circular(25),
                child: Container(
                        color: color,
                        child: Center(
                        child: Text(


                                buttonText,

                                style: TextStyle(

                                color: textColor,
                                fontSize: 25,
                                fontWeight: FontWeight.bold,
                                ),
                        ),
                        ),
                ),
                ),
        ),
        );
}
}
```

Adding dependencies in pubspec.yaml file

```yaml
dependencies:
  flutter:
    sdk: flutter



  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2
  math_expressions: ^2.4.0

dev_dependencies:
  flutter_test:
    sdk: flutter
```

Output:

Result:

Thus a Simple Android Application for Native Calculator is developed and executed successfully

Ex.No:4 Develop a gaming application that uses 2-D animations and gestures

Aim:

To develop a gaming application that uses 2-D animations and gestures

Procedure:

1.Set Up Android Studio:

Download and install Android Studio.Set up your development environment by installing necessary SDKs and tools.

2.Design User Interface (UI):

Use Android Studio's layout editor to design your app's UI.Drag and drop widgets like TextView, EditText, Button, etc., onto your layout.Customize layout using constraints or other layout types.

3.Customize Fonts and Colors:

Add custom font files to the res/font directory.Define font families in res/font/font_family.xml.Define colors in res/values/colors.xml.

4.Implement Functionality:

Write the logic for your app's functionality in the associated Java/Kotlin file.Set up event listeners for user interactions.Handle these interactions and perform desired actions.

5.Test and Deploy:

Test your app on different devices using Android Virtual Device (AVD) or physical devices.Debug and fix any issues.Once ready, deploy your app to the Google Play Store or other distribution channels.

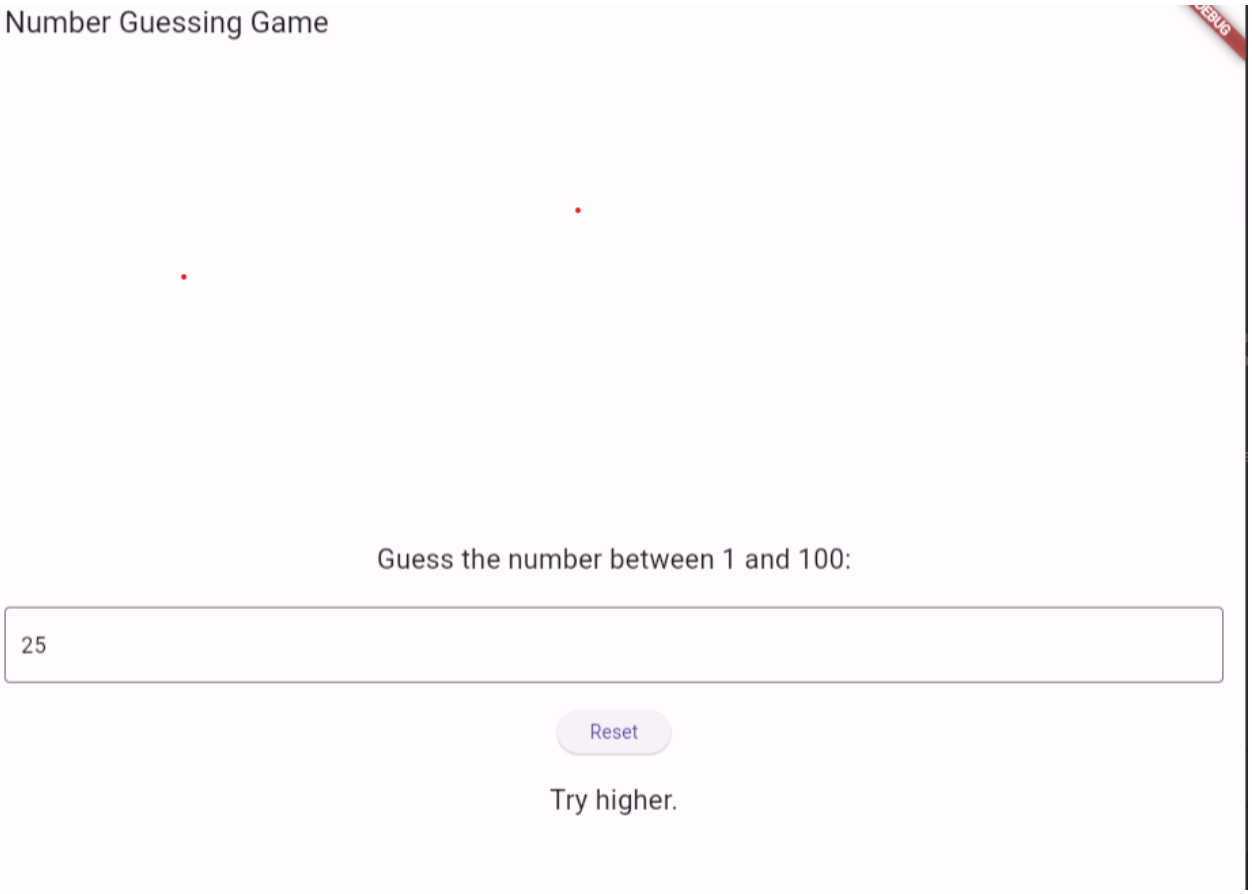
Code:

```
import 'dart:math';

import 'package:flutter/material.dart';

void main() {
  runApp(NumberGuessingGame());
}

class NumberGuessingGame extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: NumberGuessingScreen(),
    );
  }
}

class NumberGuessingScreen extends StatefulWidget {
```

```
  @override
  _NumberGuessingScreenState createState() => _NumberGuessingScreenState();
}

class _NumberGuessingScreenState extends State<NumberGuessingScreen> {
  final _random = Random();
  int _targetNumber = 0;
  int? _guess;
  String _message = '';
  int _attempts = 0;

  @override
  void initState() {
    super.initState();
    _resetGame();
  }

  void _resetGame() {
    _targetNumber = _random.nextInt(100) + 1;
    _guess = null;
    _message = '';
    _attempts = 0;
  }

  void _checkGuess() {
    if (_guess == _targetNumber) {
      setState(() {


        _message = 'Congratulations! You guessed the number in $_attempts attempts.';
      });
    } else if (_guess! < _targetNumber) {
      setState(() {
        _message = 'Try higher.';
      });
    } else {
      setState(() {
        _message = 'Try lower.';
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Number Guessing Game'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
```

```
          children: [
            Text(
              'Guess the number between 1 and 100:',
              style: TextStyle(fontSize: 20.0),
            ),
            SizedBox(height: 20.0),
            TextField(
              keyboardType: TextInputType.number,
              onChanged: (value) => _guess = int.tryParse(value),
              onSubmitted: (_) {
                setState(() {
                  _attempts++;
                  _checkGuess();
                });
              },
              decoration: InputDecoration(
                hintText: 'Enter your guess',
                border: OutlineInputBorder(),
              ),
            ),
            SizedBox(height: 20.0),
            ElevatedButton(
              onPressed: () {
                setState(() {
                  _resetGame();
                });
              },
              child: Text('Reset'),
            ),
            SizedBox(height: 20.0),
            Text(
              _message,
              style: TextStyle(fontSize: 20.0),
            ),
          ],
        ),
      ),
    );
  }
}
```

OUTPUT:


Number Guessing Game
Guess the number between 1 and 100:
25
Reset
Try higher.

Result:

Thus, the program was executed successfully.

Ex.No: 5 Develop a movie rating application (similar to IMDB)

Aim:

To develop a movie rating application.

Procedure:

1.Set Up Android Studio:

Download and install Android Studio.Set up your development environment by installing necessary SDKs and tools.

2.Design User Interface (UI):

Use Android Studio's layout editor to design your app's UI.Drag and drop widgets like TextView, EditText, Button, etc., onto your layout.Customize layout using constraints or other layout types.

3.Customize Fonts and Colors:

Add custom font files to the res/font directory.Define font families in res/font/font_family.xml.Define colors in res/values/colors.xml.

4.Implement Functionality:

Write the logic for your app's functionality in the associated Java/Kotlin file.Set up event listeners for user interactions.Handle these interactions and perform desired actions.

5.Test and Deploy:

Test your app on different devices using Android Virtual Device (AVD) or physical devices.Debug and fix any issues.Once ready, deploy your app to the Google Play Store or other distribution channels.

MainActivity.java

```
package com.example.radiobutton; import
    android.os.Bundle;
import android.app.Activity; import
android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import  android.widget.RadioGroup.OnCheckedChangeListener; import
android.widget.TextView;
import android.widget.Toast;
public class MainActivity extends Activity
```

```
{
private RadioGroup radioGroup;
private RadioButton sound, vibration,
silent; private Button button;
private TextView textView;
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
radioGroup = (RadioGroup) findViewById(R.id.myRadioGroup);
radioGroup.setOnCheckedChangeListener(new
OnCheckedChangeListener()
 {


@Override
public void onCheckedChanged(RadioGroup group, int checkedId)
 {
// find which radio button is selecet if(checkedId == R.id.silent)
{
Toast.makeText(getApplicationContext(), "choice: Silent", Toast.LENGTH_SHORT).show();

}
 else if(checkedId ==R.id.sound)
{
Toast.makeText(getApplicationContext(), "choice: Sound", Toast.LENGTH_SHORT).show();

}
else
 {
Toast.makeText(getApplicationContext(), "choice: Vibration",
Toast.LENGTH_SHORT).show();

}
}
});
sound = (RadioButton) findViewById(R.id.sound);
```

```
vibration=(RadioButton)findViewById(R.id.vibrate);

silent = (RadioButton) findViewById(R.id.silent);
textView = (TextView) findViewById(R.id.textView1);
button = (Button)findViewById(R.id.button1);
button.setOnClickListener(new OnClickListener()


{
@Override
public void onClick(View v) {
int selectedId = radioGroup.getCheckedRadioButtonId();

//findwhichradioButtonischeckedbyid
if(selectedId ==sound.getId())
{
textView.setText("You chose 'Sound' option");
}
else if(selectedId == vibration.getId())

        {
        textView.setText("You chose 'Vibration' option");
        }
        else
        {
        textView.setText("Youchose 'Silent' option");
        }
        }
        });
        }


        }
```

MainActivity.xml

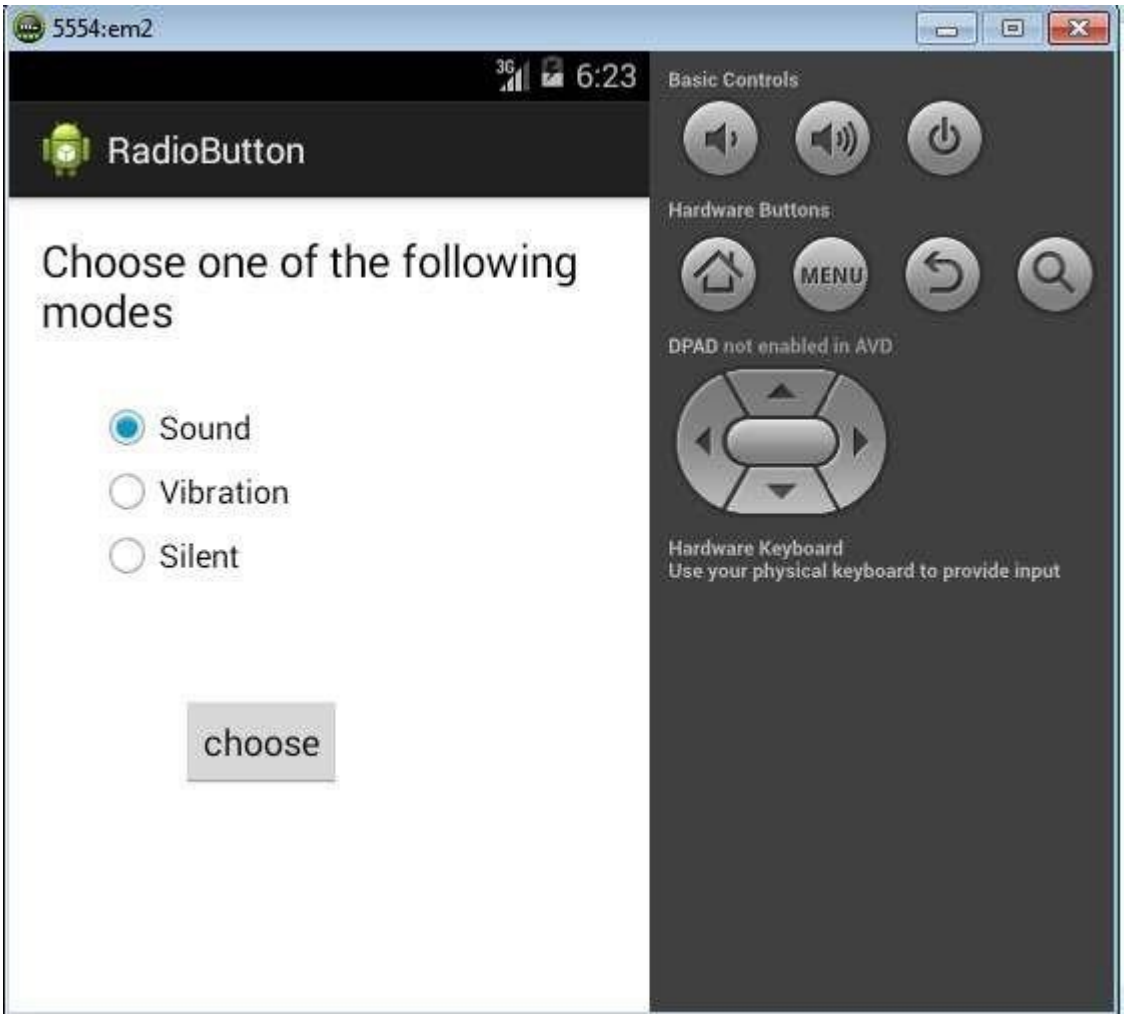
```
        <RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical
_margin"
android:paddingLeft="@dimen/activity_horizontal_
margin"
android:paddingRight="@dimen/activity_horizontal
_margin"
android:paddingTop="@dimen/activity_vertical_ma


rgin" tools:context=".MainActivity" >

<RadioGroup
android:id="@+id/myRadio
Group"
android:layout_width="wrap
_content"
android:layout_height="wrap_
content"
android:layout_alignParentLef
t="true"
android:layout_below="@+id/
textView1"
android:layout_marginLeft="
27dp"
android:layout_marginTop="
28dp" >

<RadioButton

android:id="@+id/sound"
android:layout_width="wrap_
content"

—
android:layout_height="wrap_
content"
android:checked="true"
```

```
android:text="Sound" />

<RadioButton
android:id="@+id/vibrate"
android:layout_width="wra
p_content"
android:layout_height="wra
p_content"
android:text="Vibration"
/>



<RadioButton
android:id="@+id/silent"
android:layout_width="wra
p_content"

android:layout_height="wra
p_content"
android:text="Silent" />

</RadioGroup>
<TextView
android:id="@+id/textVie
w1"

android:layout_width="wra"

p_content"
android:layout_height="wra
p_content"

android:layout_alignParentL
eft="true"
android:layout_alignParentT
op="true"

android:text="Choose one of the following modes"

android:textAppearance="?android:attr/textAppearanceLarge" />
<Button
```

```
android:id="@+id/bu
t ton1"

android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/myRadioGr
oup"
android:layout_below="@+id/myRadioGro
up" android:layout_marginLeft="42dp"
android:layout_marginTop="53dp"
android:text="choose" />

</RelativeLayout
```

OUTPUT:



Result:

Thus, the program was executed and implemented successfully

Ex.No: 6 Develop an application to connect to a web service and to retrieve data with HTTP

Aim:

To develop an application to connect to a web service and to retrieve data with HTTP

Procedure:

1.Set Up Android Studio:

Download and install Android Studio.Set up your development environment by installing necessary SDKs and tools.

2.Design User Interface (UI):

Use Android Studio's layout editor to design your app's UI.Drag and drop widgets like TextView, EditText, Button, etc., onto your layout.Customize layout using constraints or other layout types.

3.Customize Fonts and Colors:

Add custom font files to the res/font directory.Define font families in res/font/font_family.xml.Define colors in res/values/colors.xml.

4.Implement Functionality:

Write the logic for your app's functionality in the associated Java/Kotlin file.Set up event listeners for user interactions.Handle these interactions and perform desired actions.

5.Test and Deploy:

Test your app on different devices using Android Virtual Device (AVD) or physical devices.Debug and fix any issues.Once ready, deploy your app to the Google Play Store or other distribution channels.

Program Code:

```
import 'dart:async';
import 'dart:convert';

import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;

Future<Album> fetchAlbum() async {
  final response = await http
      .get(Uri.parse('https://jsonplaceholder.typicode.com/albums/10'));

  if (response.statusCode == 200) {

    return Album.fromJson(jsonDecode(response.body) as Map<String, dynamic>);
  } else {
    throw Exception('Failed to load album');
  }
}

class Album {
  final int userId;
  final int id;
  final String title;

  const Album({
    required this.userId,
    required this.id,
    required this.title,
```

```
  });

  factory Album.fromJson(Map<String, dynamic> json) {
    return switch (json) {
      {
      'userId': int userId,
      'id': int id,
      'title': String title,
      } =>
          Album(
            userId: userId,
            id: id,
            title: title,
          ),
      _ => throw const FormatException('Failed to load album.'),
    };
  }
}

void main() => runApp(const MyApp());

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



  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  late Future<Album> futureAlbum;

  @override
  void initState() {
    super.initState();
    futureAlbum = fetchAlbum();
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Fetch Data Example',
      theme: ThemeData(
          colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
          scaffoldBackgroundColor: Colors.blue
      ),
      home: Scaffold(
        appBar: AppBar(
          title: const Text('Fetch Data Example'),
          backgroundColor: Colors.blue,
        ),
        body: Stack(
            children: [
              Container(
                child: Center(
                  child: FutureBuilder<Album>(
                    future: futureAlbum,
                    builder: (context, snapshot) {
```

```
        if (snapshot.hasData) {
                  return Text(snapshot.data!.title,style: TextStyle(fontSize: 50,color: Colors.red),);
                } else if (snapshot.hasError) {
                  return Text('${snapshot.error}');
                }

                // By default, show a loading spinner.
                return const CircularProgressIndicator();
              },

            ),
          ),
        )


      ]
    ),
  ),
 );
 }
}
```

OUTPUT:

Result:

Thus, the program was implemented and executed successfully

Ex.No: 7. Develop a simple shopping application

Aim:

To develop a simple shopping application

Procedure:

1.Set Up Android Studio:

Download and install Android Studio.Set up your development environment by installing necessary SDKs and tools.

2.Design User Interface (UI):

Use Android Studio's layout editor to design your app's UI.Drag and drop widgets like TextView, EditText, Button, etc., onto your layout.Customize layout using constraints or other layout types.

3.Customize Fonts and Colors:

Add custom font files to the res/font directory.Define font families in res/font/font_family.xml.Define colors in res/values/colors.xml.

4.Implement Functionality:

Write the logic for your app's functionality in the associated Java/Kotlin file.Set up event listeners for user interactions.Handle these interactions and perform desired actions.

5.Test and Deploy:

Test your app on different devices using Android Virtual Device (AVD) or physical devices.Debug and fix any issues.Once ready, deploy your app to the Google Play Store or other distribution channels.

Main.dart

```
import 'dart:ui_web';


import 'package:flutter/material.dart';


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


class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
```

```
    title: 'Shopping Cart',
    theme: ThemeData(
      primarySwatch: Colors.blue,
    ),
    home: ShoppingCart(),
  );
 }
}

class ShoppingCart extends StatefulWidget {
 @override
 _ShoppingCartState createState() => _ShoppingCartState();
}

class _ShoppingCartState extends State<ShoppingCart> {
 List<Product> products = [
  Product(name: 'Product 1', price: 10,image="assets\img.png"),
  Product(name: 'Product 2', price: 20,),
  Product(name: 'Product 3', price: 30),
  Product(name: 'Product 4', price: 30),
  Product(name: 'Product 5', price: 30),
  Product(name: 'Product 6', price: 50),
 ];

 Map<Product, int> cart = {};

 void addToCart(Product product) {
  if (cart.containsKey(product)) {
    cart[product] = cart[product]! + 1;
  } else {
    cart[product] = 1;
  }
```

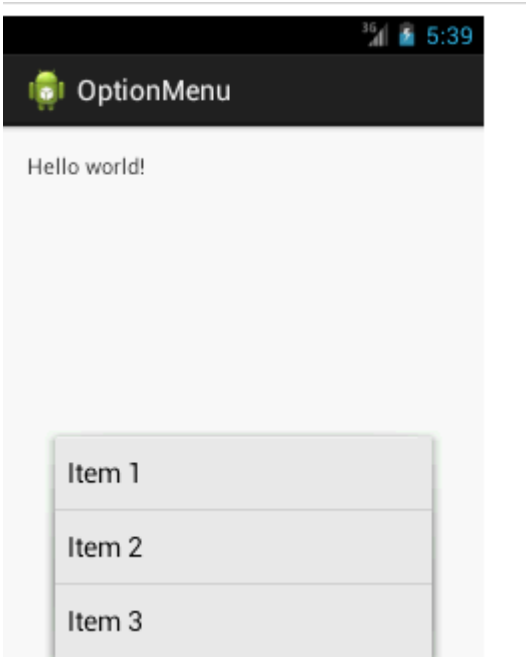
```
setState(() {});
 }

 void removeFromCart(Product product) {
  if (cart.containsKey(product)) {
    if (cart[product]! > 1) {
      cart[product] = cart[product]! - 1;
    } else {
      cart.remove(product);
    }
    setState(() {});
  }
 }

 double getTotalPrice() {
return cart.entries.fold(0, (total, entry) => total + (entry.key.price * entry.value));
 }

 @override
 Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Shopping Cart'),
    ),
    body: ListView.builder(
      itemCount: products.length,
      itemBuilder: (context, index) {
        final product = products[index];
        return ListTile(leading: Image.network('https://example.com/image.png'),
          title: Text(product.name),
          subtitle: Text('\$${product.price.toStringAsFixed(2)}'),
          trailing: ElevatedButton(
```

```
      onPressed: () => addToCart(product),
          child: Text('Buy'),
        ),
      );
    },
  ),
  floatingActionButton: FloatingActionButton(
    onPressed: () {
      Navigator.push(
        context,
        MaterialPageRoute(builder: (context) => CartScreen(cart: cart)),
      );
    },
    child: Icon(Icons.shopping_cart),
  ),

 );

}
}

class Product {
  final String name;
  final double price;
  Product({required this.name, required this.price});
}

class CartScreen extends StatelessWidget {
  final Map<Product, int> cart;

  const CartScreen({Key? key, required this.cart}) : super(key: key);
```

```
@override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: Text('Cart'),
   ),
   body: ListView.builder(
    itemCount: cart.length,
    itemBuilder: (context, index) {
     final product = cart.keys.toList()[index];
     final quantity = cart[product];
     return ListTile(image.assets('assets/img.png')
      title: Text(product.name),
      subtitle: Text('\$${product.price.toStringAsFixed(2)} x $quantity'),
      trailing: IconButton(
       icon: Icon(Icons.remove),

 onPressed: () => _removeFromCart(context, product),
      ),
     );

},
   ),
   bottomNavigationBar: BottomAppBar(
    child: Padding(
     padding: const EdgeInsets.all(16.0),
     child: Text(
      'Total: \$${getTotalPrice(cart).toStringAsFixed(2)}',
      style: TextStyle(fontSize: 20.0, fontWeight: FontWeight.bold),
     ),
    ),
   ),
```

```
    );
  }

  double getTotalPrice(Map<Product, int> cart) {
    return cart.entries.fold(0, (total, entry) => total + (entry.key.price * entry.value));
  }
  void _removeFromCart(BuildContext context, Product product) {
    _ShoppingCartState state = context.findAncestorStateOfType<_ShoppingCartState>()!;
    state.removeFromCart(product);
  }
}
```

OUTPUT:



Result:

Thus, the program was implemented and executed successfully

Ex.No: 8 **Design a web server supporting push notifications.**

Aim:

To develop a web server supporting push notifications.

Algorithm:

1. Create a New Android Project:
   - Click New in the toolbar.
   - In the window that appears, open the Android folder, select Android Application Project, and click next.
   - Provide the application name and the project name and then finally give the desired package name.
   - Choose a launcher icon for your application and then select Blank Activity and then click Next
   - Provide the desired Activity name for your project and then click Finish.
2. Create a New AVD (Android Virtual Device):
   - click Android Virtual Device Manager from the toolbar.
   - In the Android Virtual Device Manager panel, click New.
   - Fill in the details for the AVD. Give it a name, a platform target, an SD card size, and a skin (HVGA is default).
   - Click Create AVD and Select the new AVD from the Android Virtual Device Manager and click Start.
3. Design the layout by adding a text box and a command button.
4. Run the application.
5. If the entered E-mail doesn't match the given E-mail id, then an alert will be displayed.
6. If the entered E-mail id matches with the provided mail-id then login is successful.
7. Close the Android project.

PROGRAM CODE:

Main.dart

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:push_notification/navigate.dart';

import 'firebase_options.dart';
import 'notification.dart';

void main() async{
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );
  await PushNotificationService().initialize();
  await PushNotificationService().getToken();
  runApp(MyApp());
}

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

  @override
  State<MyApp> createState() => _MyAppState();
```

```
}

class _MyAppState extends State<MyApp> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(child: Text("Notifiication Test")),
    );
  }
}
```

Notification.dart

```
import 'package:firebase_messaging/firebase_messaging.dart';

class PushNotificationService {
  FirebaseMessaging _fcm = FirebaseMessaging.instance;

  Future initialize() async {
    FirebaseMessaging.onMessage.listen((RemoteMessage message) {
      print('Got a message whilst in the foreground!');
      print('Message data: ${message.data}');

      if (message.notification != null) {
        print('Message also contained a notification: ${message.notification}');
      }
    });
  }

  Future<String?> getToken() async {
    String? token = await _fcm.getToken();
    print('Token: $token');
    return token;
  }
}
```

Firebase.dart

```
// File generated by FlutterFire CLI.
// ignore_for_file: type=lint
import 'package:firebase_core/firebase_core.dart' show FirebaseOptions;
import 'package:flutter/foundation.dart'
    show defaultTargetPlatform, kIsWeb, TargetPlatform;

/// Default [FirebaseOptions] for use with your Firebase apps.
///
/// Example:
/// ```dart
/// import 'firebase_options.dart';
/// // ...
/// await Firebase.initializeApp(
///   options: DefaultFirebaseOptions.currentPlatform,
/// );
/// ```
class DefaultFirebaseOptions {
  static FirebaseOptions get currentPlatform {
    if (kIsWeb) {
      return web;
    }
    switch (defaultTargetPlatform) {
      case TargetPlatform.android:
        return android;
      case TargetPlatform.iOS:
        return ios;
      case TargetPlatform.macOS:
        return macos;
      case TargetPlatform.windows:
        return windows;
      case TargetPlatform.linux:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for linux - '
          'you can reconfigure this by running the FlutterFire CLI again.',
        );
      default:
        throw UnsupportedError(
          'DefaultFirebaseOptions are not supported for this platform.',
        );
    }
  }

  static const FirebaseOptions web = FirebaseOptions(
    apiKey: 'AIzaSyDo8Fu14QO3xmYzDhMMaLBqIBTEY5loaYU',
    appId: '1:603150362822:web:8f67879b819ea54918cfc7',
    messagingSenderId: '603150362822',
    projectId: 'push-notification-f04c0',
    authDomain: 'push-notification-f04c0.firebaseapp.com',
    storageBucket: 'push-notification-f04c0.appspot.com',
  );

  static const FirebaseOptions android = FirebaseOptions(
    apiKey: 'AIzaSyAumRRc9Guv3WgLRprxHZs9B2iuSUuf19M',
    appId: '1:603150362822:android:8bc9ca501f35343118cfc7',
    messagingSenderId: '603150362822',
    projectId: 'push-notification-f04c0',
    storageBucket: 'push-notification-f04c0.appspot.com',
  );

  static const FirebaseOptions ios = FirebaseOptions(
```

```
    apiKey: 'AIzaSyCoATukJQgpLqtqMFFP8A0SJSD14Zvz6-g',
    appId: '1:603150362822:ios:edadc8822450e85a18cfc7',
    messagingSenderId: '603150362822',
    projectId: 'push-notification-f04c0',
    storageBucket: 'push-notification-f04c0.appspot.com',
    iosBundleId: 'com.example.pushNotification',
  );

  static const FirebaseOptions macos = FirebaseOptions(
    apiKey: 'AIzaSyCoATukJQgpLqtqMFFP8A0SJSD14Zvz6-g',
    appId: '1:603150362822:ios:edadc8822450e85a18cfc7',
    messagingSenderId: '603150362822',
    projectId: 'push-notification-f04c0',
    storageBucket: 'push-notification-f04c0.appspot.com',
    iosBundleId: 'com.example.pushNotification',
  );

  static const FirebaseOptions windows = FirebaseOptions(
    apiKey: 'AIzaSyDo8Fu14QO3xmYzDhMMaLBqIBTEY5loaYU',
    appId: '1:603150362822:web:5af224ea389860bf18cfc7',
    messagingSenderId: '603150362822',
    projectId: 'push-notification-f04c0',
    authDomain: 'push-notification-f04c0.firebaseapp.com',
    storageBucket: 'push-notification-f04c0.appspot.com',
  );
}
```

OUTPUT:

Result:

Thus, the program was implemented and executed successfully.

Ex.No: 9 Develop an application by integrating Google maps

AIM:

To develop an android application that uses Google Map location information.

ALGORITHM:

1. Create a New Android Project:
   - Click New in the toolbar.
   - In the window that appears, open the Android folder, select Android Application Project, and click next.
   - Provide the application name and the project name and then finally give the desired package name.
   - Choose a launcher icon for your application and then select Blank Activity and then click Next
   - Provide the desired Activity name for your project and then click Finish.
2. Create a New AVD (Android Virtual Device):
   - click Android Virtual Device Manager from the toolbar.
   - In the Android Virtual Device Manager panel, click New.
   - Fill in the details for the AVD. Give it a name, a platform target, an SD card size, and a skin (HVGA is default).
   - Click Create AVD and Select the new AVD from the Android Virtual Device Manager and click Start.
3. Design the graphical layout.
4. Run the application.
5. The requested data is retrieved from the database named myFriendsDb.
6. Close the Android project.

PROGRAM CODE

UseGps.java

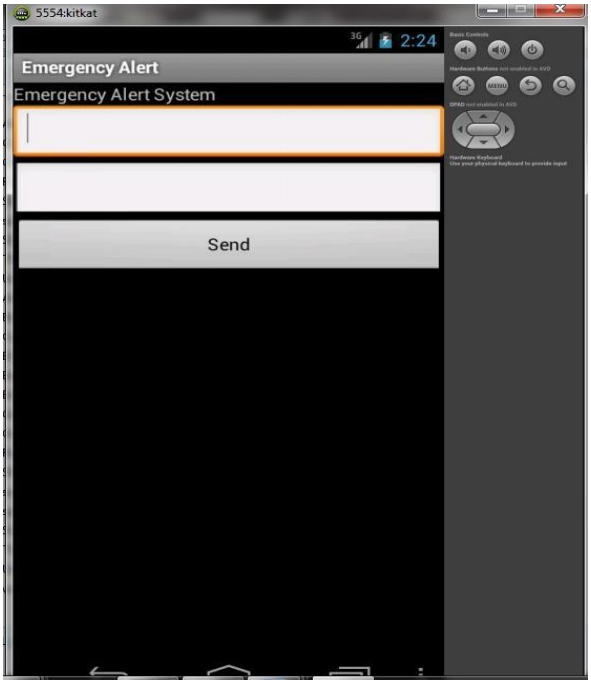
```
package com.emergency;
import android.app.Activity;
import
android.content.Context;
import android.location.Location;
import
android.location.LocationListener;
import
android.location.LocationManager;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

```
public class UseGps extends Activity
{
        Button
        buttonSend;
        EditTexttextSMS;
        EditTexttextlon;
public void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
buttonSend = (Button) findViewById(R.id.buttonSend);
textSMS = (EditText) findViewById(R.id.editTextSMS);
textlon = (EditText) findViewById(R.id.textlon);
LocationManagermlocManager                                                    =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
LocationListenermlocListener = new MyLocationListener();
mlocManager.requestLocationUpdates( LocationManager.GPS_PROVIDER, 0, 0,
mlocListener);
}
public class MyLocationListener implements LocationListener
{
public void onLocationChanged(Location loc)
{
loc.getLatitude();
loc.getLongitude();
Double lat=loc.getLatitude();
Double lon=loc.getLongitude();
textSMS.setText(lat.toString());
textlon.setText(lon.toString());
}
public void onProviderDisabled(String provider)
{
Toast.makeText( getApplicationContext(),"Gps Disabled",Toast.LENGTH_SHORT ).show();
}
public void onProviderEnabled(String provider)
{
Toast.makeText( getApplicationContext(), "Gps Enabled", Toast.LENGTH_SHORT).show();
}
}
}
```

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/a
ndroid" android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
>
<TextView
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Emergency Alert System"
   />

<EditText
android:id="@+id/editTextSMS"
android:layout_width="fill_parent"
android:layout_height="wrap_conte
nt" android:gravity="top" />

<EditText
android:id="@+id/textlon"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:gravity="top" />

<Button
android:id="@+id/buttonSend"
android:layout_width="fill_parent"
android:layout_height="wrap_conte
nt" android:text="Send" />

</LinearLayout>
```

OUTPUT:



Result:

Thus, the program for android application that makes use of Google Map was executed successfully.

Ex.No: 10 **Mini Projects involving Flutter/Kotlin multi-platform AIM:**

To Write a mobile application that creates alarm clock.

PROCEDURE:

Create a new Android Application

1. In Eclipse go to **File->New->Project**
2. Select an **Android Project** from the Android Folder and press **Next.**
3. Fill in the details of your Android application.
   a. **Project Name**: The project name and folder that Eclipse will store the project files
   b. **Build Target:** The version of the Android SDK that will be used when you build your program. Select a platform that is equal to or lower than the target chosen for the AVD.
   c. **Application Name:** This is the name of the application.
   d. **Package Name**: The namespace that all of the source code will reside under.
   e. **Create Activity:** The name for that class stub that is generated by the plugin.
4. The values that are used in this example are:
   a. **Project Name**: Alarm
   b. **Build Target:** 2.3.3
   c. **Application Name:** Alarm
   d. **Package Name**: com. Alarm.example
   e. **Create Activity:** Alarm
5. Click on **Finish. Coding the Application**

1. Open **AndroidMainfest.xml** which is located in **res->values-> AndroidMainfest.xml.** This file will hold all of the text that our layout will use.
2. Click on the **AndroidMainfest**.xml at the bottom to bring up the raw xml file.

Editing the the java code

1. Open SampleApp.java from the left hand side.
2. Save the files.

Running the Application

1. Click on the green circle with the white arrow.
2. Choose the AVD that we created in a previous step.
3. The android AVD will load and the program will run.

PROGRAMS

FileName :MainActivity.java

```
Package
com.lab.alarmclock;
import
java.util.Calendar;
import
android.app.Activity;
import
android.app.AlarmManager;
import
android.app.PendingIntent;
import
android.content.Context;
import
android.content.Intent;
import android.os.Bundle;
import android.view.View;
import
android.view.View.OnClickListene
r; import android.widget.Button;
import
android.widget.TimePicker;

public class AlarmActivity extends Activity
                        {

        private TimePicker
        timepicker; private Context
        context; private Button
        btnSetAlarm;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                // TODO Auto-generated method stub
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);

               context = this;

                timepicker = (TimePicker) findViewById(R.id.timepicker);

                btnSetAlarm = (Button) findViewById(R.id.btnSetAlarm);
                btnSetAlarm.setOnClickListener(new OnClickListener() {
```

```
                @Override
                public void onClick(View v) {
                        // TODO Auto-generated method stub
Calendar calendar = Calendar.getInstance(); calendar.set(Calendar.HOUR_OF_DAY,
        timepicker.getCurrentHour()); calendar.set(Calendar.MINUTE,
                timepicker.getCurrentMinute());

                        Intent myIntent = new Intent(context,
                        AlarmReceiver.class); PendingIntent pendingIntent =
                        PendingIntent.getBroadcast(
                                context, 0, myIntent, 0);

                        AlarmManager alarmManager =
 (AlarmManager) getSystemService(ALARM_SERVICE);
                        alarmManager.set(AlarmManager.RTC,
                                calendar.getTimeInMillis(), pendingIntent);

                }
        });

};
```

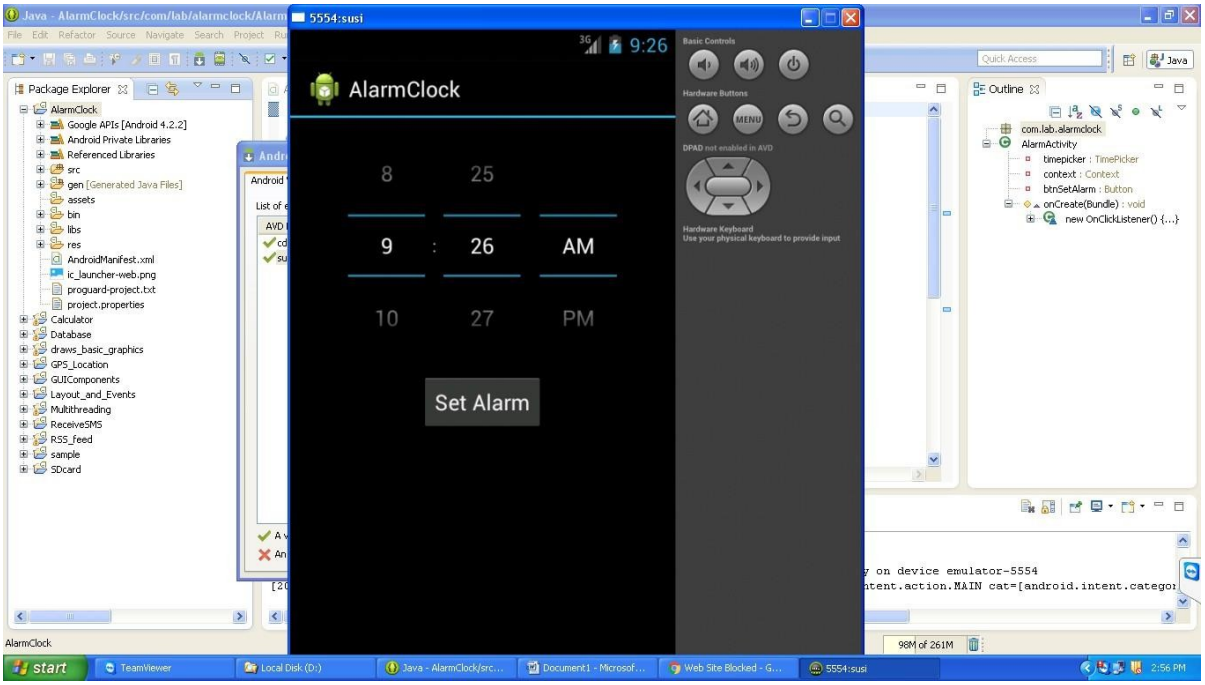
File Name: Alaram Reciever.java

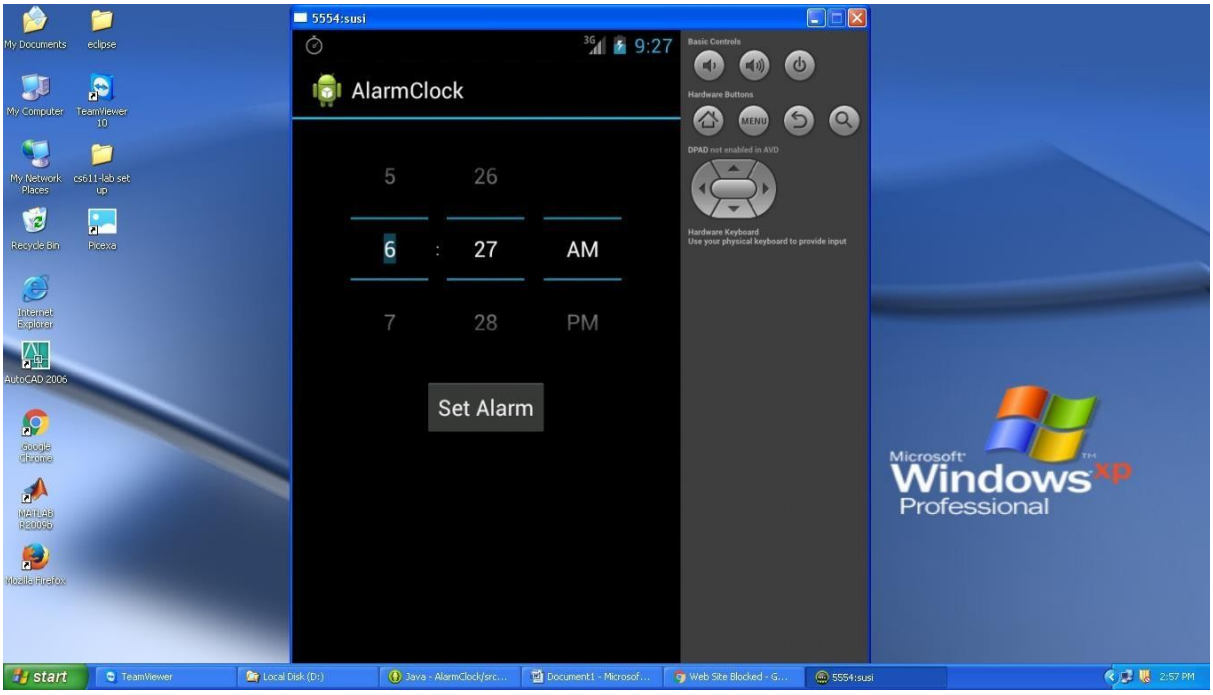
```
package com.lab.alarmclock;

import
android.content.Context;
import
android.content.Intent;
import
android.media.Ringtone;
import
android.media.RingtoneManager;
import android.net.Uri;
import
android.support.v4.content.WakefulBroadcastReceiver;
import android.util.Log;
import android.widget.Toast;

public class AlarmReceiver

extends

        WakefulBroadcastReceiver { @Override
        public void onReceive(Context context, Intent intent) {
                // TODO Auto-generated method stub

                Log.e("alarmreceiver", "alarmreceiver");
                Toast.makeText(context, "alarmreceiver", Toast.LENGTH_LONG).show();

                Uri alarmUri = RingtoneManager
                                .getDefaultUri(RingtoneManager.TYPE_ALARM);

                Ringtone ringtone = RingtoneManager.getRingtone(context,

                alarmUri); ringtone.play();

        }
}
```

File Name: Androidmainfest.xml

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

<manifest
   xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.lab.alarmclock"

   android:versionCode="1"
   android:versionName="1.0"
   >

   <uses-permission android:name="android.permission.WAKE_LOCK" />

   <uses-sdk
      android:minSdkVersion="8"
      android:targetSdkVersion="2
      1"
      />


<application
android:allowBackup="true "
      android:icon="@drawable/ic_launc
      he                                r"
      android:label="@string/app_name"
      android:theme="@style/AppThem
      e" >

      <activity
         android:name=".AlarmActivity"
         android:label="@string/app_name
         "
         >

         <intent-filter>
            <action android:name="android.intent.action.MAIN" />


            <category android:name="android.intent.category.LAUNCHER" />

         </intent-filter>
      </activity>
      <receiver android:name=".AlarmReceiver" />

   </application>

</manifest>
```

OUTPUT:


5554:susi
9:26
AlarmClock
8
25
9
:
26
AM
10
27
PM
Set Alarm
Basic Controls
Hardware Buttons
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input
Package Explorer
AlarmClock
Google APIs [Android 4.2.2]
Android Private Libraries
Referenced Libraries
src
gen [Generated Java Files]
assets
bin
libs
res
AndroidManifest.xml
ic_launcher-web.png
proguard-project.txt
project.properties
Calculator
Database
draws_basic_graphics
GPS_Location
GUIComponents
Layout_and_Events
Multithreading
ReceiveSMS
RSS_feed
sample
SDcard
Outline
com.lab.alarmclock
AlarmActivity
timepicker : TimePicker
context : Context
btnSetAlarm : Button
onCreate(Bundle) : void
new OnClickListener() {...}
start
TeamViewer
Local Disk (D:)
Document1 - Microsof...
Web Site Blocked - G...
2:56 PM


5554:susi
9:27
AlarmClock
5
26
6
:
27
AM
7
28
PM
Set Alarm
Basic Controls
Hardware Buttons
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input
My Documents
eclipse
My Computer
TeamViewer 10
My Network Places
cs611-lab set up
Recycle Bin
Picasa
Internet Explorer
AutoCAD 2006
Google Chrome
MATLAB R2009a
Mozilla Firefox
Microsoft
Windows xp
Professional
start
TeamViewer
Local Disk (D:)
Document1 - Microsof...
Web Site Blocked - G...
2:57 PM

RESULT:

Thus the mobile application that creates alarm clock