An application that uses GUI components, Fonts, Colours

Expt 1 **Date: 18/08/2022**

Aim:

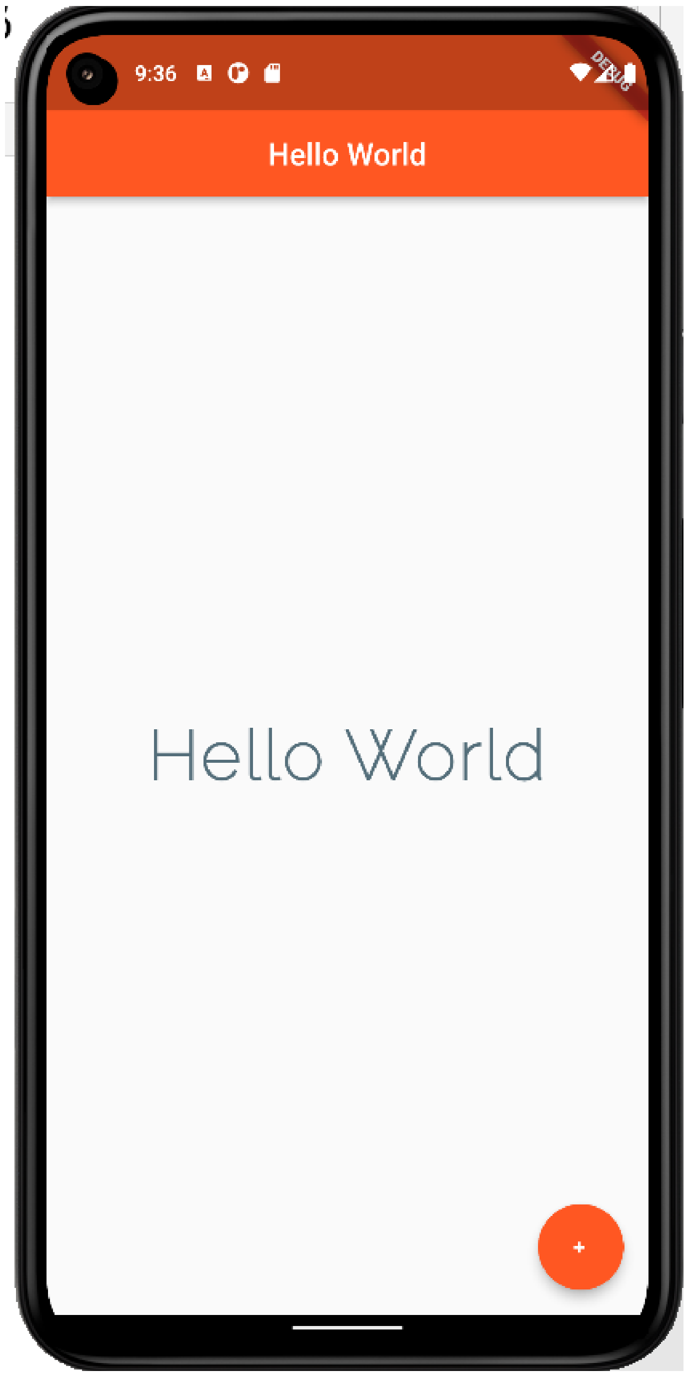
To create a mobile application that uses GUI components, fonts, and colours.

Code:

main.dart

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

void main() {
  runApp(MaterialApp(
    home: Home(),
  ));
}

class Home extends StatelessWidget {
  const Home({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
          title: Text("Hello World"),
          centerTitle: true,
          backgroundColor: Colors.deepOrange,
        ),
        body: Center(
          child: Text(
            "Hello World",
            style: TextStyle(
              fontSize: 45.0,
              fontWeight: FontWeight.bold,
              letterSpacing: 2.0,
              color: Colors.blueGrey[600],
              fontFamily: 'Raleway',
            ),
          ),
        ),
        floatingActionButton: FloatingActionButton(
          onPressed: () {},
          child: Text("+"),
          backgroundColor: Colors.deepOrange,
        ),
      );
  }
}
```

Output:



Result:

A mobile application which uses GUI components, fonts, and colours has been implemented successfully

An application that uses Layout Managers and Event Listeners

Expt 2 **Date: 25/08/2022**

Aim:

To create a mobile application that uses Layout Managers and Event Listeners

Code:

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


void main() {

 runApp(const MaterialApp(

   home: Home(),

 ));

}


class Home extends StatefulWidget {

 const Home({Key? key}) : super(key: key);


 @override

 State<Home> createState() => _HomeState();

}


class _HomeState extends State<Home> {

 int projects = 0;

 @override

 Widget build(BuildContext context) {

   return Scaffold(

     backgroundColor: Colors.black54,
     appBar: AppBar(

       title: Text("Profile"),

       backgroundColor: Colors.black12,

       centerTitle: true,

       elevation: 0.0,

     ),

     body: Padding(

       padding: EdgeInsets.fromLTRB(30.0, 40.0, 30.0, 0.0),

       child: Column(

         crossAxisAlignment: CrossAxisAlignment.start,

         children: <Widget>[

           Center(

             child: CircleAvatar(

               backgroundImage: AssetImage(''),

               radius: 50.0,

             ),

           ),

           SizedBox(

             height: 20.0,

           ),

           Text(

             "NAME",

             style: TextStyle(

               color: Colors.grey,

               letterSpacing: 2.0,

             ),

           ),

           SizedBox(
             height: 10.0,
           ),
           Text(
             "Cobra tate",
             style: TextStyle(
               color: Colors.limeAccent,
               letterSpacing: 2.0,
               fontSize: 28.0,
               fontWeight: FontWeight.bold,
             ),
           ),
           SizedBox(
             height: 20.0,
           ),
           Text(
             "PROJECTS",
             style: TextStyle(
               color: Colors.grey,
               letterSpacing: 2.0,
             ),
           ),
           SizedBox(
             height: 10.0,
           ),
           Text(
             "$projects",
             style: TextStyle(
               color: Colors.limeAccent,
               letterSpacing: 2.0,
               fontSize: 28.0,
               fontWeight: FontWeight.bold,
             ),
           ),
           SizedBox(
             height: 20.0,
           ),
           ElevatedButton(
             onPressed: () {
               setState(() {
                 projects++;
               });
             },
             onLongPress: () {
               setState(() {
                 projects *= 2;
               });
             },
             child: Icon(
               Icons.add,
             ),
             style: ElevatedButton.styleFrom(
               primary: Colors.green,
             ),
           ),
           SizedBox(
             height: 20.0,
           ),

           ElevatedButton(

             onPressed: () {

               setState(() {

                 if (projects > 0) projects--;

               });

             },

             onLongPress: () {

               setState(() {

                 if (projects > 0) projects ~/= 2;

               });

             },

             child: Icon(

               Icons.remove,

             ),

             style: ElevatedButton.styleFrom(

               primary: Colors.deepOrange,

             ),

           ),

           SizedBox(

             height: 20.0,

           ),

           Row(

             children: [

               Icon(

                 Icons.mail,

                 color: Colors.grey,

               ),
               SizedBox(
                 width: 20.0,
               ),
               Text(
                 "andrewtate@topg.com",
                 style: TextStyle(
                   color: Colors.grey,
                   fontWeight: FontWeight.bold,
                   fontSize: 15.0,
                 ),
               )
             ],
           )
         ],
       ),
     ),
   );
 }
}
```

Output:

Result:

An application that uses layout managers and event listeners has been implemented successfully.

Creation of Calculator Application

Expt 3 **Date: 01/09/2022**

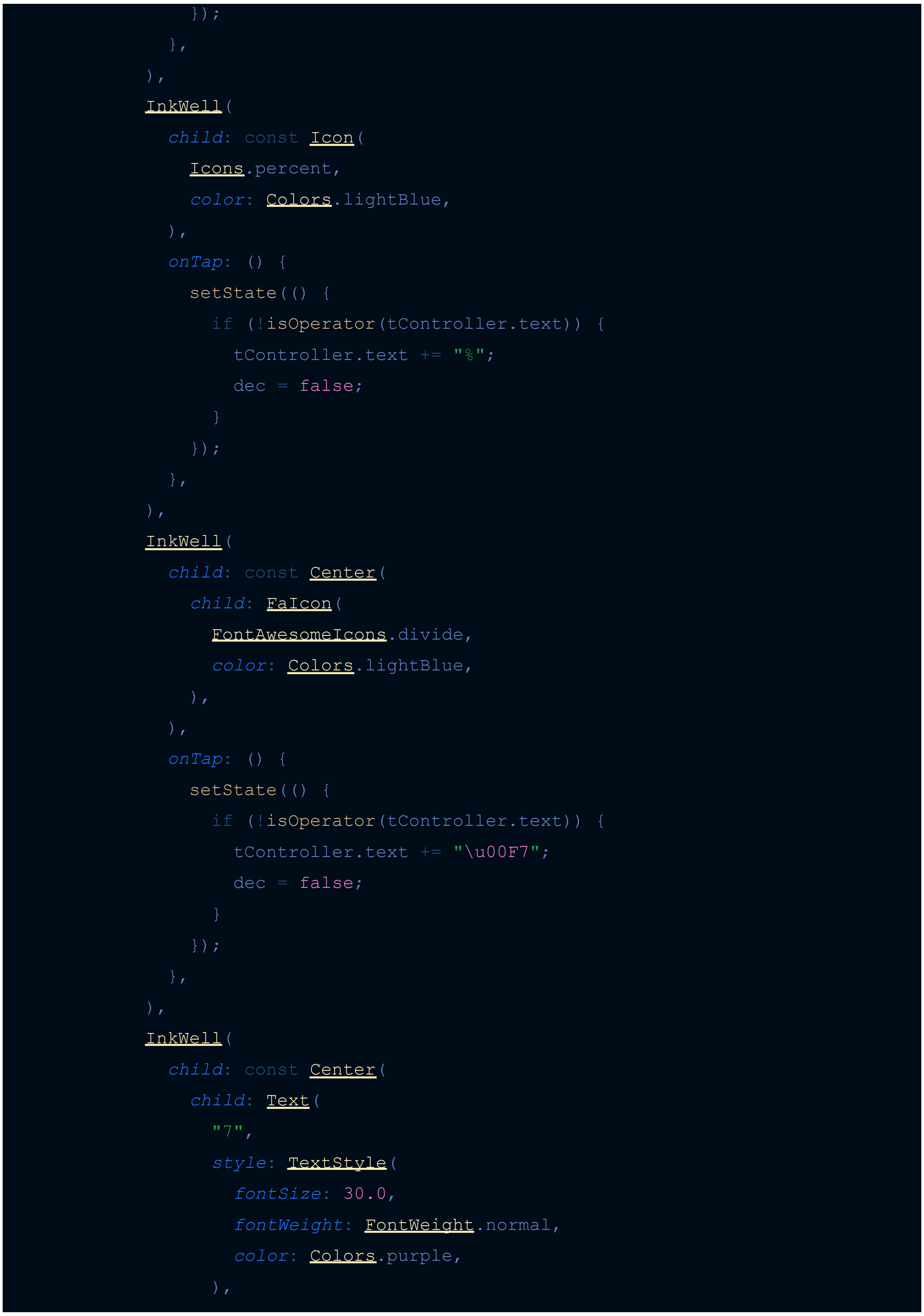
Aim:

To create a mobile calculator application

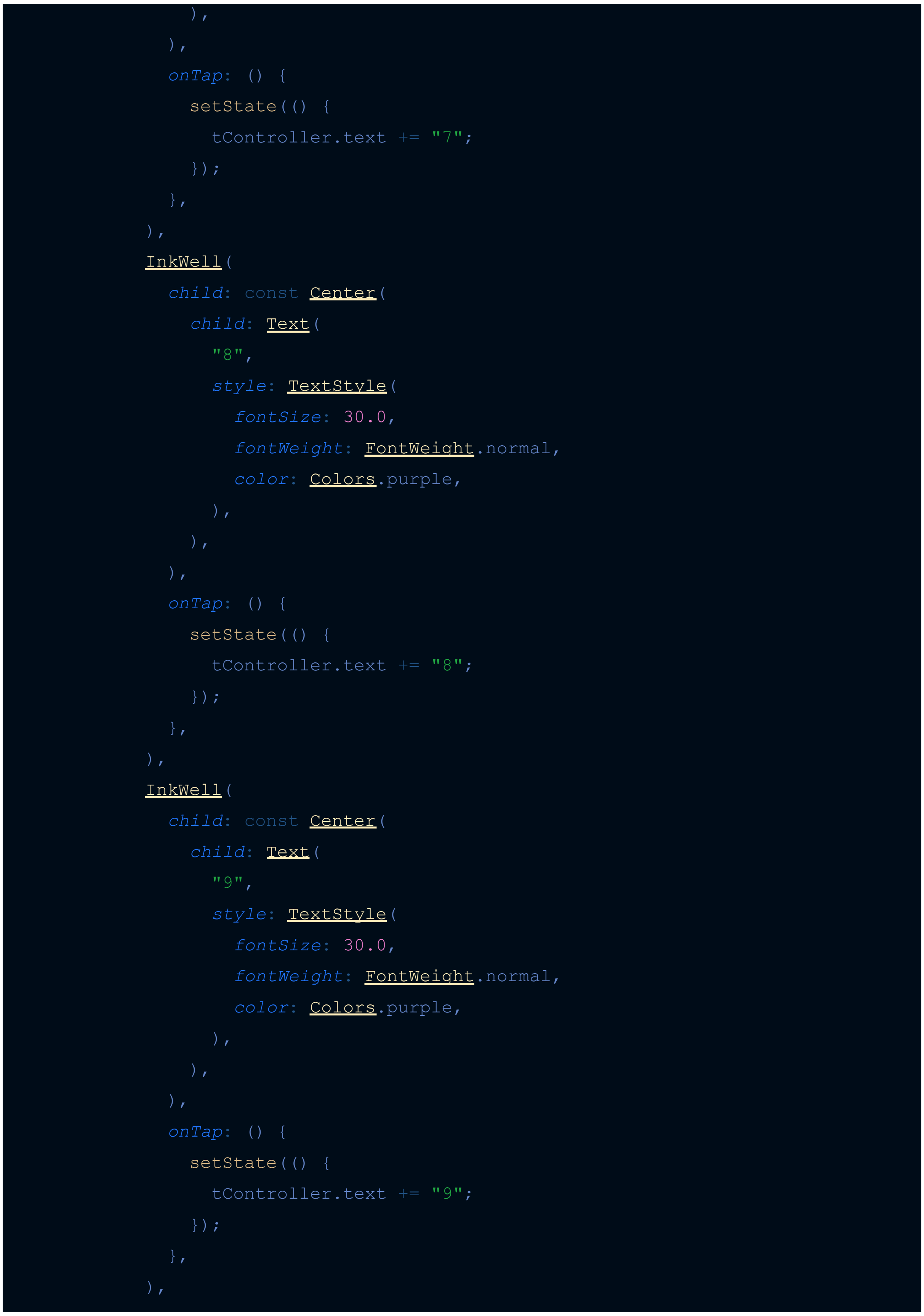
Code:

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

void main() {
 runApp(MaterialApp(
   home: const Home(),
   theme: ThemeData.dark(),
 ));
}

class Home extends StatefulWidget {
 const Home({Key? key}) : super(key: key);

 @override
 State<Home> createState() => _HomeState();
}

class _HomeState extends State<Home> {
 @override
 Widget build(BuildContext context) {
   return Scaffold(
     appBar: AppBar(
       title: const Text("Calculator"),
       backgroundColor: Colors.lightBlue,
     ),
     body: const Padding(
       padding: EdgeInsets.fromLTRB(20.0, 30.0, 20.0, 40.0),
       child: CalcBody(),
     ),
```

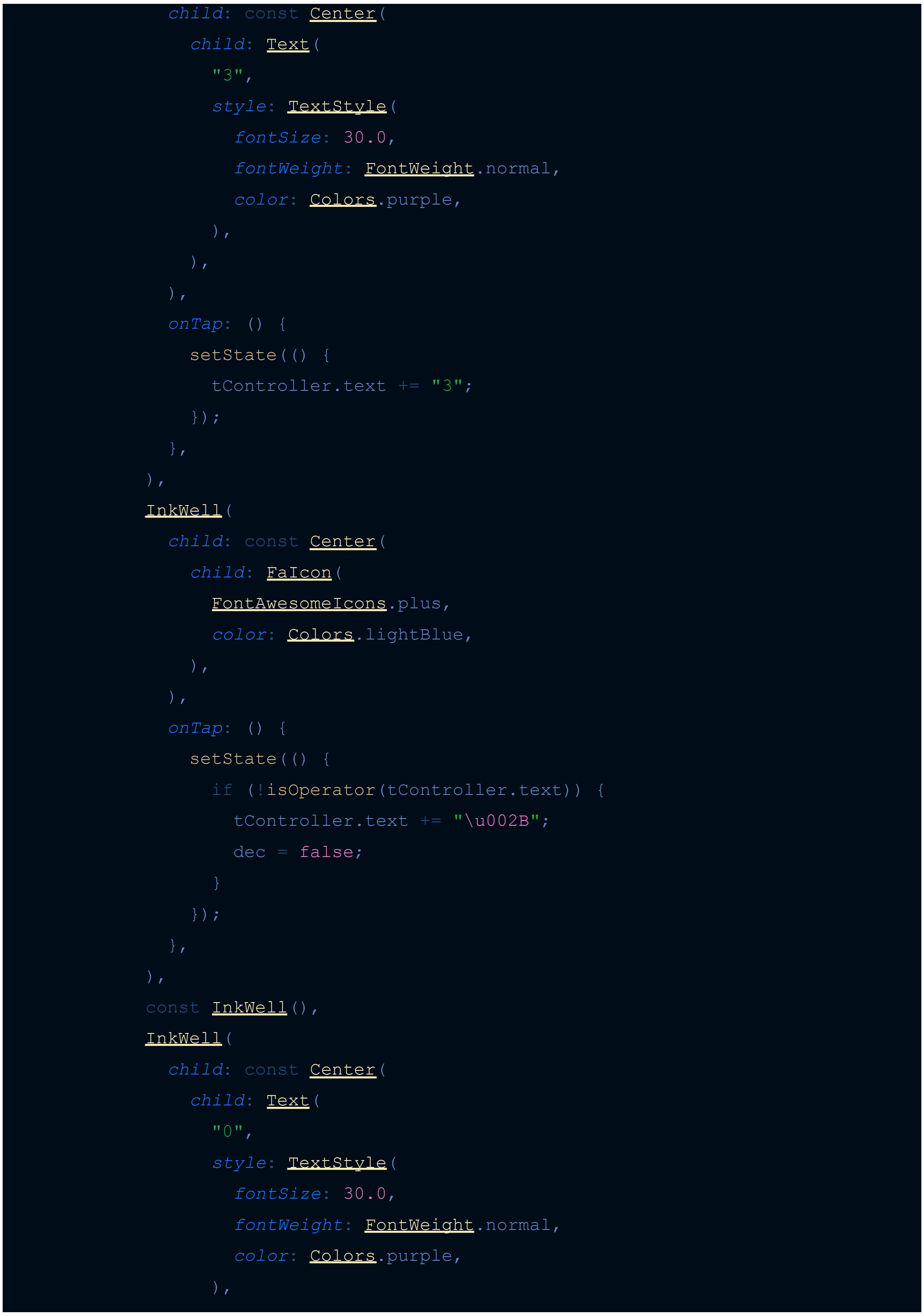
```dart
    );
  }
}

class CalcBody extends StatefulWidget {
  const CalcBody({Key? key}) : super(key: key);

  @override
  State<CalcBody> createState() => _CalcBodyState();
}

class _CalcBodyState extends State<CalcBody> {
  var tController = TextEditingController();
  bool dec = false;
  bool isOperator(String s) {
    if (s[s.length - 1] == "\u00F7" ||
        s[s.length - 1] == "\u00D7" ||
        s[s.length - 1] == "-" ||
        s[s.length - 1] == "\u002B" ||
        s[s.length - 1] == "." ||
        s[s.length - 1] == "%") {
      return true;
    }
    return false;
  }

  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        const SizedBox(
          height: 120.0,
        ),
        Container(
          padding: const EdgeInsets.all(22.0),
          child: TextField(
            textAlign: TextAlign.right,
            decoration: const InputDecoration(
              hintText: "0",
            ),
            style: const TextStyle(
              fontSize: 45.0,
            ),
            controller: tController,
            readOnly: true,
          ),
        ),
        const SizedBox(
          height: 10.0,
        ),
        Expanded(
          child: GridView.count(
            crossAxisSpacing: 10,
            crossAxisCount: 4,
            children: [
              InkWell(
                child: const Center(
                  child: Text(
                    "AC",
                    style: TextStyle(
                      fontSize: 20.0,
                      fontWeight: FontWeight.bold,
                      color: Colors.lightBlue,
                    ),
                  ),
                ),
                onTap: () {
                  setState(() {
                    tController.text = "";
                    dec = false;
                  });
                },
              ),
              InkWell(
                child: const Icon(
                  Icons.backspace,
                  color: Colors.lightBlue,
                ),
                onTap: () {
                  setState(() {
                    if (tController.text.isNotEmpty) {
                      tController.text = tController.text
                          .substring(0, tController.text.length - 1);
                    }
```



```dart
    });
  },
),
InkWell(
  child: const Icon(
    Icons.percent,
    color: Colors.lightBlue,
  ),
  onTap: () {
    setState(() {
      if (!isOperator(tController.text)) {
        tController.text += "%";
        dec = false;
      }
    });
  },
),
InkWell(
  child: const Center(
    child: FaIcon(
      FontAwesomeIcons.divide,
      color: Colors.lightBlue,
    ),
  ),
  onTap: () {
    setState(() {
      if (!isOperator(tController.text)) {
        tController.text += "\u00F7";
        dec = false;
      }
    });
  },
),
InkWell(
  child: const Center(
    child: Text(
      "7",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
```

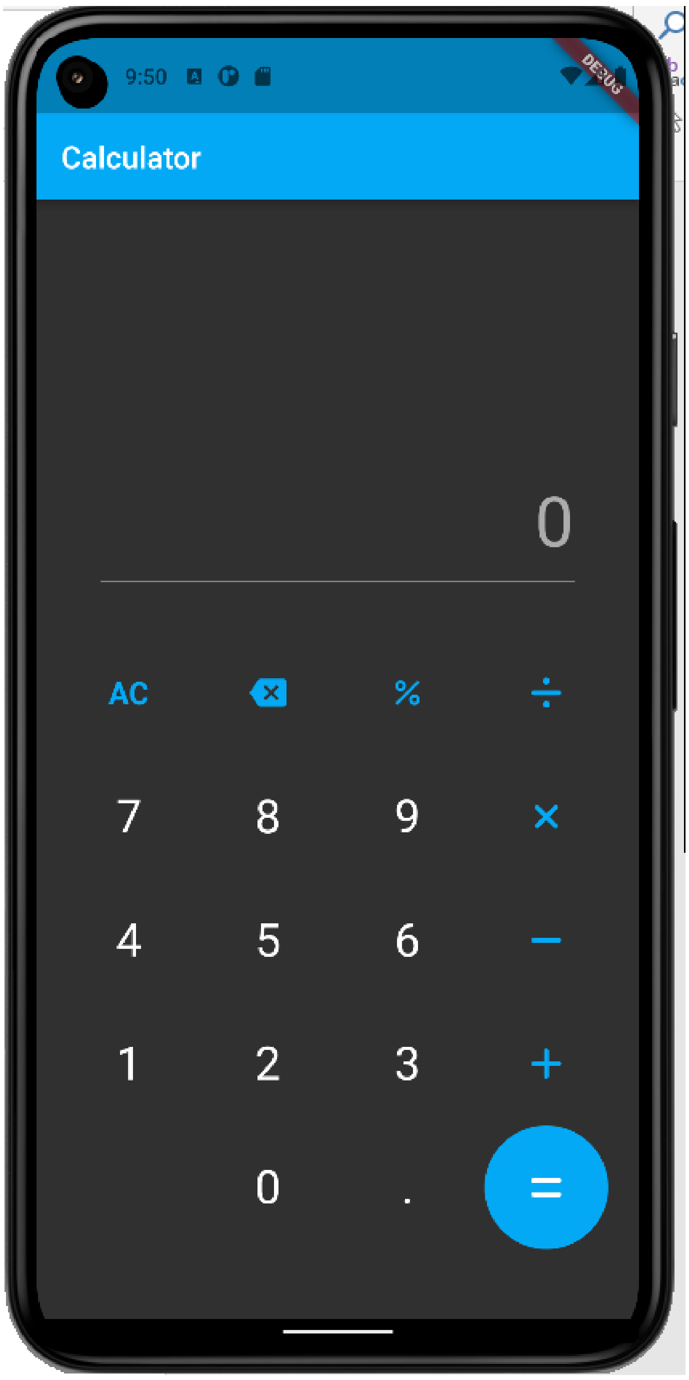


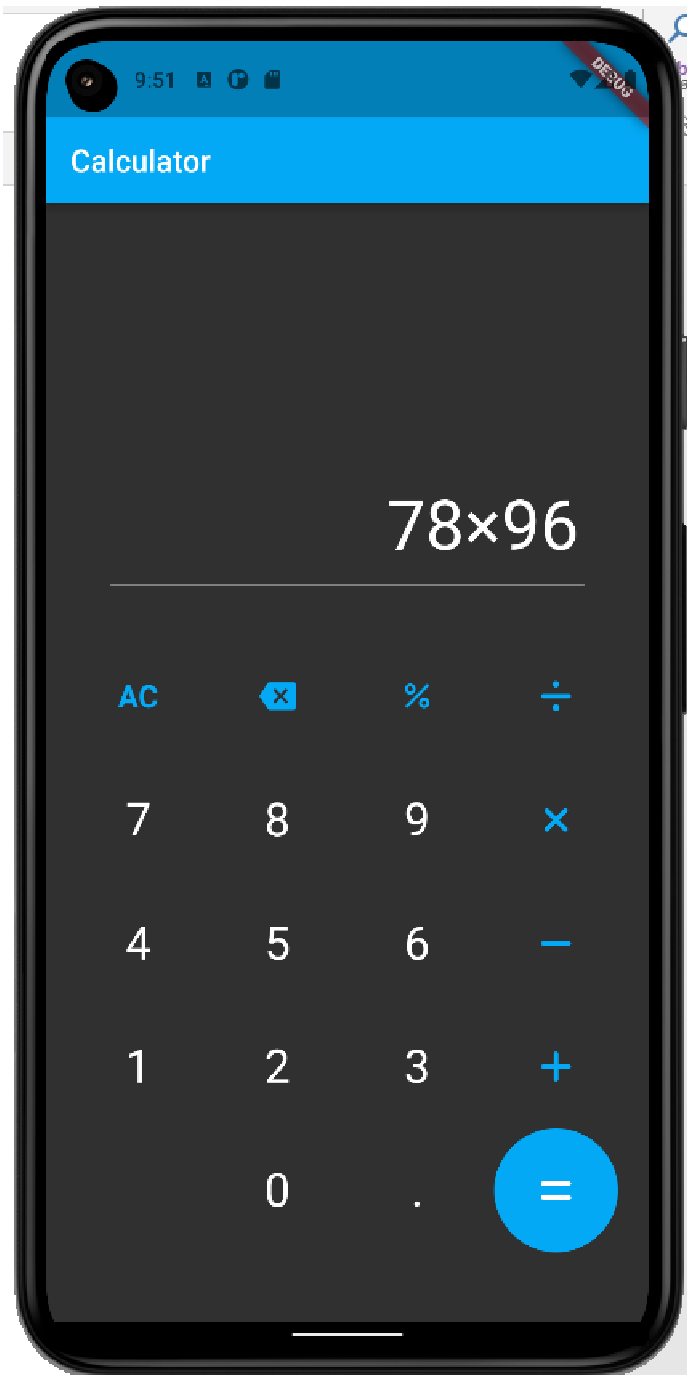
```
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "7";
    });
  },
),
InkWell(
  child: const Center(
    child: Text(
      "8",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "8";
    });
  },
),
InkWell(
  child: const Center(
    child: Text(
      "9",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "9";
    });
  },
),
InkWell(
  child: const Center(
    child: FaIcon(
      FontAwesomeIcons.xmark,
      color: Colors.lightBlue,
    ),
  ),
  onTap: () {
    setState(() {
      if (!isOperator(tController.text)) {
        tController.text += "\u00D7";
        dec = false;
      }
    });
  },
),
InkWell(
  child: const Center(
    child: Text(
      "4",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "4";
    });
  },
),
InkWell(
  child: const Center(
    child: Text(
      "5",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "5";
    });
  },
),
InkWell(
  child: const Center(
    child: Text(
      "6",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "6";
    });
  },
),
InkWell(
  child: const Center(
    child: FaIcon(
      FontAwesomeIcons.minus,
      color: Colors.lightBlue,
    ),
  ),
  onTap: () {
    setState(() {
      if (!isOperator(tController.text)) {
        tController.text += "-";
        dec = false;
      }
    });
  },
),
InkWell(
  highlightColor: Colors.grey,
  splashColor: Theme.of(context).canvasColor,
  child: Container(
    decoration: BoxDecoration(
      color: Theme.of(context).canvasColor,
      shape: BoxShape.circle,
    ),
    child: const Center(
      child: Text(
        "1",
        style: TextStyle(
          fontSize: 30.0,
          fontWeight: FontWeight.normal,
          color: Colors.purple,
        ),
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "1";
    });
  },
),
InkWell(
  child: const Center(
    child: Text(
      "2",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "2";
    });
  },
),
InkWell(
```

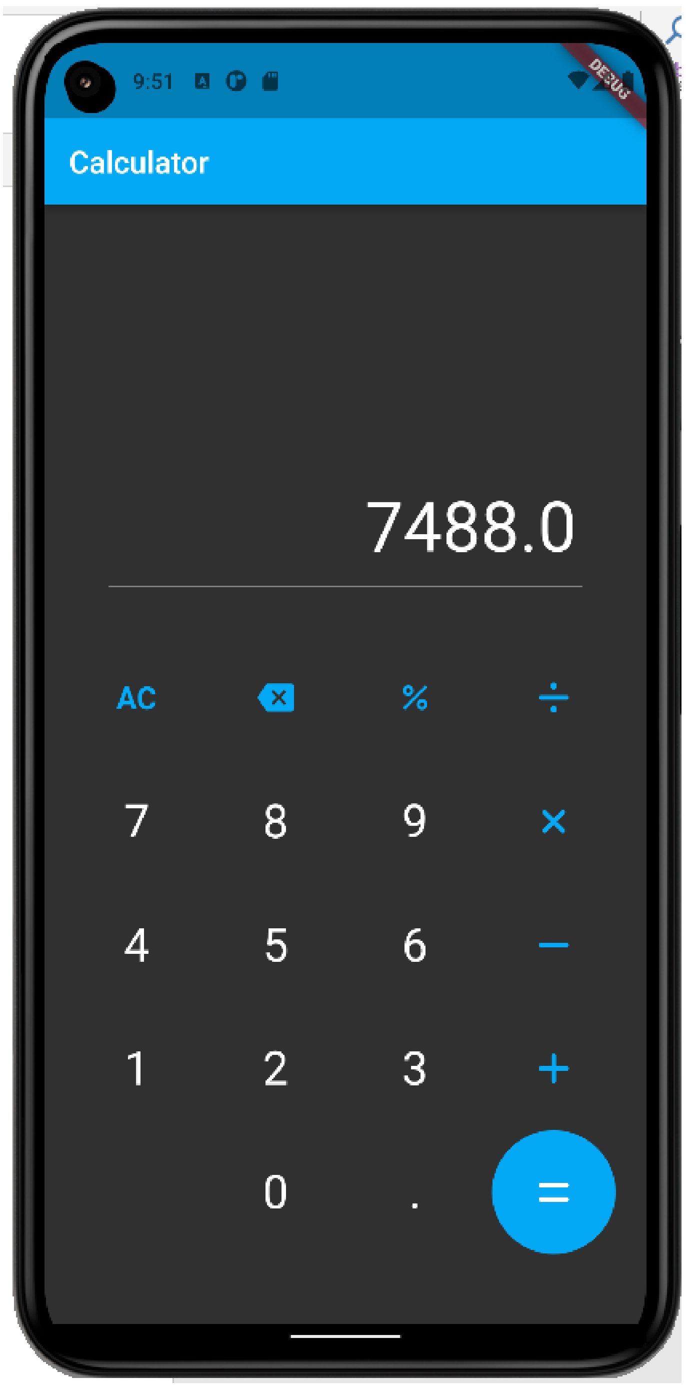


```
  child: const Center(
    child: Text(
      "3",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "3";
    });
  },
),
InkWell(
  child: const Center(
    child: FaIcon(
      FontAwesomeIcons.plus,
      color: Colors.lightBlue,
    ),
  ),
  onTap: () {
    setState(() {
      if (!isOperator(tController.text)) {
        tController.text += "\u002B";
        dec = false;
      }
    });
  },
),
const InkWell(),
InkWell(
  child: const Center(
    child: Text(
      "0",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      tController.text += "0";
    });
  },
),
InkWell(
  child: const Center(
    child: Text(
      ".",
      style: TextStyle(
        fontSize: 30.0,
        fontWeight: FontWeight.normal,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
    setState(() {
      if (!dec && !isOperator(tController.text)) {
        tController.text += ".";
        dec = true;
      }
    });
  },
),
InkWell(
  child: Container(
    decoration: const BoxDecoration(
      color: Colors.lightBlue,
      shape: BoxShape.circle,
    ),
    child: const Center(
      child: FaIcon(
        FontAwesomeIcons.equals,
        color: Colors.purple,
      ),
    ),
  ),
  onTap: () {
     String expression = '';
     for (int i = 0; i < tController.text.length; i++) {
       if (tController.text[i] == '×') {
         expression += '*';
       } else if (tController.text[i] == '÷') {
         expression += '/';
       } else {
         expression += tController.text[i];
       }
     }
     try {
       Parser p = Parser();
       Expression exp = p.parse(expression);
       ContextModel cm = ContextModel();
       double eval = exp.evaluate(EvaluationType.REAL, cm);
       setState(() {
         tController.text = '$eval';
       });
     } catch (e) {
       setState(() {
         tController.text = 'ERR';
       });
     }
   },
 ),
],
),
),
],
);
}
}
```

Output:


9:50
DEBUG
Calculator
0
AC
%
÷
7
8
9
×
4
5
6
−
1
2
3
+
0
.
=


9:51
DEBUG
Calculator
78×96
AC
%
÷
7
8
9
×
4
5
6
−
1
2
3
+
0
.
=



Result:

A calculator application for mobiles has been implemented successfully.

An application that draws basic graphical primitives on screen

Expt 4 **Date: 08/09/2022**

Aim:

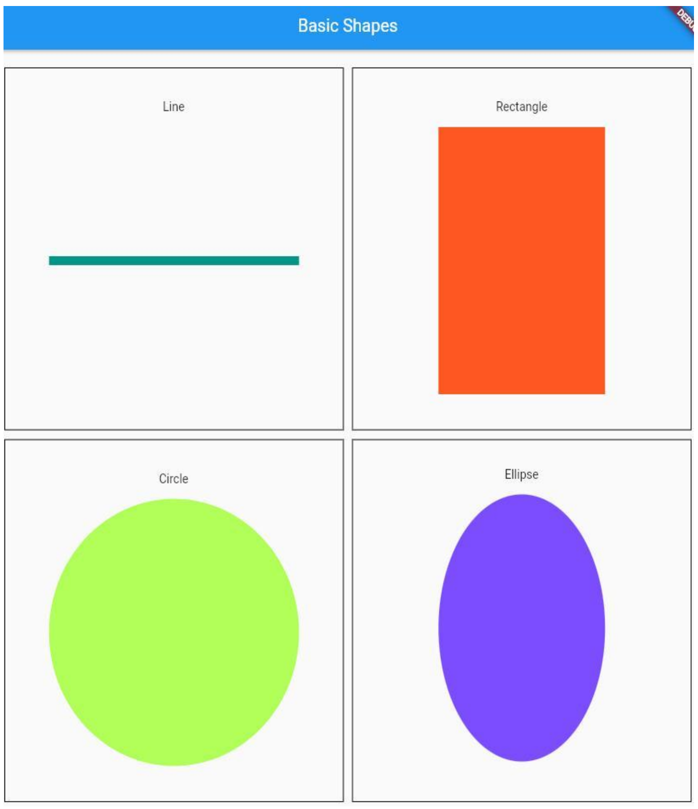
To create a mobile application that draws basic graphical primitives on screen.

Code:

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

void main() {
 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: 'Flutter Demo',
     theme: ThemeData(
       primarySwatch: Colors.blue,
     ),
     home: const MyHomePage(title: 'Week 4 Graphical Primitives'),
   );
 }
}

class MyHomePage extends StatefulWidget {
 const MyHomePage({super.key, required this.title});
 final String title;

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

class _MyHomePageState extends State<MyHomePage> {
 Widget build(BuildContext context) {
```

```
return Scaffold(
    appBar: AppBar(
      title: Text(widget.title),
    ),
    body: ListView(
      children: [
        Text('\nCircle\n'),
        Container(
            height: 100.0,
            width: 50.0,
            // ignore: prefer_const_constructors
            decoration: new ShapeDecoration(
              shape: const CircleBorder(side: BorderSide.none),
              color: Colors.pink,
            )),
        Text('\n\nRectangle\n\n'),
        Container(
          height: 150,
          width: 100,
          decoration: const BoxDecoration(
              color: Colors.red,
              borderRadius: BorderRadius.all(Radius.circular(10))),
        ),
        Text('\nSized Box\n'),
        SizedBox(
            height: 50,
            width: 10,
            child: Container(
              padding: EdgeInsets.all(10),
              decoration: BoxDecoration(
                  shape: BoxShape.rectangle,
                  borderRadius: BorderRadius.circular(10),
                  border: Border.all(color: Colors.pink)),
            )),
        Text('\n\nSquare\n\n'),
        IconButton(
          onPressed: (() {}),
          icon: const Icon(
            Icons.square,
            size: 150,
            color: Colors.black,
          ),
              )
            ],
          ));
  }
}
```

Output:



Result:

A mobile application that draws basic graphical primitives on screen has been implemented successfully.

An application that makes use of a database

Expt 5 **Date: / /2022**

Aim:

To create a mobile application that connects to a database and performs CRUD operations.

Code:

Main.dart

```
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'pages/home.dart';

void main() {
  runApp(const MaterialApp(
    home: Home(),
  ));
}
```

Home.dart

```
import 'dart:convert';

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

```dart
import '../schema.dart';

class Home extends StatefulWidget {
  const Home({Key? key}) : super(key: key);

  @override
  State<Home> createState() => _HomeState();
}

class _HomeState extends State<Home> {
  var tController1 = TextEditingController();
  var tController2 = TextEditingController();
  var tController3 = TextEditingController();
  var url="http://10.106.206.0:5000";

  static const _iconTypes = <IconData>[
    Icons.add,
    FontAwesomeIcons.rotate,
    Icons.delete,
    FontAwesomeIcons.eye,
    FontAwesomeIcons.solidEye,
  ];
  // Map<IconData,Function> iconMap={
  //   Icons.add:
  // }
  int curIcon=0;
  int decider=1;
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text(
          'Student Profile',
          style: TextStyle(
            fontSize: 20.0,
          ),
        ),
        centerTitle: true,
        backgroundColor: Colors.indigo,
      ),
      body: SingleChildScrollView(
        child: Form(
          key: _formKey,
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.center,
            children: <Widget>[
              Container(
                padding: const EdgeInsets.fromLTRB(32.0,32.0,32.0,0.0),
                child: TextFormField(
                  decoration: const InputDecoration(
                    hintText: "Reg No.",
                  ),
                  keyboardType: TextInputType.number,
                  controller: tController1,
                  validator: (String? value) {
                    if ( (value == null || value.isEmpty) &&curIcon%5!=4) {
                      return 'Please enter some text';
                    }
                    return null;
                  },
                ),
              ),
              const SizedBox(
                height: 10.0,
              ),
              Container(
                padding: const EdgeInsets.fromLTRB(32.0,32.0,32.0,0.0),
                child: TextFormField(
                  decoration: const InputDecoration(
                    hintText: "Name",
                  ),
                  controller: tController2,
                  validator: (String? value) {
                    if ( (value == null || value.isEmpty)
&&(curIcon%5!=4&&curIcon%5!=3&&curIcon%5!=2)) {
                      return 'Please enter some text';
                    }
                    return null;
                  },
                ),
              ),
              const SizedBox(
                height: 10.0,
              ),
              Container(
                padding: const EdgeInsets.all(32.0),
                child: TextFormField(
                  decoration: const InputDecoration(
                    hintText: "Marks",
                  ),
                  keyboardType: TextInputType.number,
                  controller: tController3,
                  validator: (String? value) {
                    if ( (value == null || value.isEmpty)
&&(curIcon%5!=4&&curIcon%5!=3&&curIcon%5!=2)){
                      return 'Please enter some text';
                    }
                    return null;
                  },
                ),
              ),
              const SizedBox(
                height: 10.0,
              ),
              GestureDetector(
                child: FloatingActionButton(
                  backgroundColor: Colors.indigo,
                  child: AnimatedSwitcher(
                      duration: const Duration(seconds: 2),
                      transitionBuilder: (Widget child, Animation<double>
animation) {
```

```
                        return ScaleTransition(scale: animation, child:
child);
                      },
                      child: Icon(
                        _iconTypes[curIcon%5],
                      )
                  ),
                  onPressed: () {
                    if (_formKey.currentState!.validate()) {
                      // Process data.
                      int opt=curIcon%5;
                      switch(opt){
                        case 0: {
                          addData();
                          showDialog(
                              context: context,
                              builder: (context) => AlertDialog(
                                title: const Text(
                                  "Insertion Done"
                                ),
                                content: const Text(
                                  "The record has been inserted"
                                ),
                                actions: [
                                  TextButton(
                                    onPressed: () => Navigator.pop(context,
'OK'),
                                    child: const Text('OK'),
                                  ),
                                ],
                              )
                          );
                        }
                        break;
                        case 1: {
                          updateData();
                          showDialog(
                              context: context,
                              builder: (context) => AlertDialog(
                                title: const Text(
                                    "Updating Done"
                                ),
                                content: const Text(
                                    "The record has been updated"
                                ),
                                actions: [
                                  TextButton(
                                    onPressed: () => Navigator.pop(context,
'OK'),
                                    child: const Text('OK'),
                                  ),
                                ],
                              ),
                          );
                        }
                        break;
                        case 2: {
                          deleteData();
                          showDialog(
                              context: context,
                              builder: (context) => AlertDialog(
                                title: const Text(
                                    "Record Deleted"
                                ),
                                content: const Text(
                                    "The record has been deleted"
                                ),
                                actions: [
                                  TextButton(
                                    onPressed: () => Navigator.pop(context,
'OK'),
                                    child: const Text('OK'),
                                  ),
                                ],
                              ),
                          );
                        }
                        break;
                        case 3: {
                          //viewOne();
                          showDialog(
                            context: context,
                            builder: (context) => AlertDialog(
                              title: const Text(
                                  "All the records in the DB"
                              ),
                              content:
FutureBuilder<Map<dynamic,dynamic>?>(
                                future: viewData(),
                                builder: (context,snapshot){
                                  if(snapshot.hasError){
                                    print("COD GOD!");
                                  }
                                  else
if(snapshot.connectionState==ConnectionState.waiting){
                                    return const
CircularProgressIndicator();
                                  }
                                  else if(snapshot.hasData){
                                    final Map<dynamic,dynamic>? viewOne =
snapshot.data;
                                    return Container(
                                      height: 300.0,
                                      width: 300.0,
                                      child: Text("Reg No.:
${viewOne?["reg_no"]} Name: ${viewOne?["name"]} Marks:
${viewOne?["marks"]}"),
                                    );
                                  }
                                  return Container();
                                },
                              ),
                              actions: [
                                TextButton(
                                  onPressed: () => Navigator.pop(context,
'OK'),

                                  child: const Text('OK'),
                                ),
                              ],
                            ),

                          );
                        }
                        break;
                        case 4: {
                          showDialog(
                              context: context,
                              builder: (context) => AlertDialog(
                                title: const Text(
                                  "All the records in the DB"
                                ),
                                content: FutureBuilder<List<dynamic>?>(
                                  future: viewAllData(),
                                  builder: (context,snapshot){
                                    if(snapshot.hasError){
                                      print("Mangathada Mariyatha");
                                    }
                                    else
if(snapshot.connectionState==ConnectionState.waiting){
                                      return const
CircularProgressIndicator();
                                    }
                                    else if(snapshot.hasData){
                                      final List<dynamic>?
ViewData=snapshot.data;

                                      return Container(
                                        height: 300.0,
                                        width: 300.0,
                                        child: ListView.builder(
                                          itemCount: ViewData?.length,
                                          itemBuilder: (BuildContext
context,int index){
                                            return Text("Reg No.:
${ViewData?[index]["reg_no"]} Name: ${ViewData?[index]["name"]} Marks:
${ViewData?[index]["marks"]}");
                                          },
                                        ),
                                      );
                                    }
                                    return Container();
                                  },
                                ),
                                actions: [
                                  TextButton(
                                    onPressed: () => Navigator.pop(context,
'OK'),

                                    child: const Text('OK'),
                                  ),
                                ],
                              ),
                          );
                        }
                        break;
                      }
                      setState(() {});
                    }
                  },
                ),
                onHorizontalDragStart: (d){},
                onHorizontalDragUpdate: (d){
                  setState(() {
                    int matter= (d.primaryDelta!).toInt();
                    decider=(matter>=0)?(1):(-1);
                  });

                },
                onHorizontalDragEnd: (details){
                  setState(() {
                    curIcon+=decider;
                  });
                },
              ),
            ],
          ),
        ),
      ),
      floatingActionButtonLocation:
FloatingActionButtonLocation.centerFloat,
    );
  }
  Future<List> viewAllData() async {
    Response response = await get(Uri.parse("${url}/view_all"));
    Map data=json.decode(response.body);
    List data1=data['result'];
    return data1;
  }
  Future<void> addData() async{
    Student s1=Student(regno: tController1.text,name:
tController2.text,marks: tController3.text);
    final response = await post(
      Uri.parse('${url}/add'),
      headers: <String, String>{
        'Content-Type': 'application/json; charset=UTF-8',
        'reg_no': s1.regno,
        'name': s1.name,
        'marks': s1.marks,
      });
  }
  Future<Map<dynamic,dynamic>> viewData() async{
    Response response = await get(
      Uri.parse("${url}/view"),
      headers: <String,String>{
        'Content-Type': 'application/json; charset=UTF-8',
        'reg_no': tController1.text,
      });
    Map data=json.decode(response.body);
    Map data1=data['result'];
    return data1;
  }
  Future<void> updateData() async{
    Student s1=Student(regno: tController1.text,name:
tController2.text,marks: tController3.text);
    Response response= await patch(
      Uri.parse("${url}/update"),
      headers: <String,String>{
        'Content-Type': 'application/json; charset=UTF-8',
        'reg_no': s1.regno,
        'name': s1.name,
        'marks': s1.marks,
      });
  }
  Future<void> deleteData() async{
    Student s1=Student(regno: tController1.text,name:
tController2.text,marks: tController3.text);
    Response response= await delete(
        Uri.parse("${url}/delete"),
        headers: <String,String>{
          'Content-Type': 'application/json; charset=UTF-8',
          'reg_no': s1.regno,
        });
  }

}
```

schema.dart

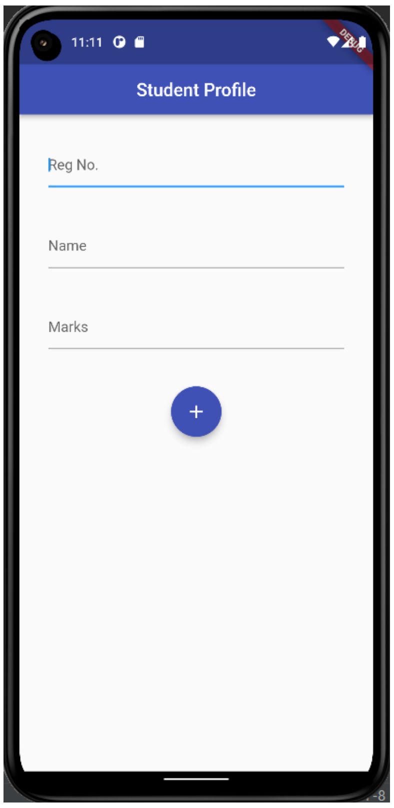
```
import 'package:flutter/material.dart';

class Student{
  String regno="-1";
  String name="Unknown";
  String marks="-1";
  Student({required this.regno,required this.name,required this.marks});
}
```

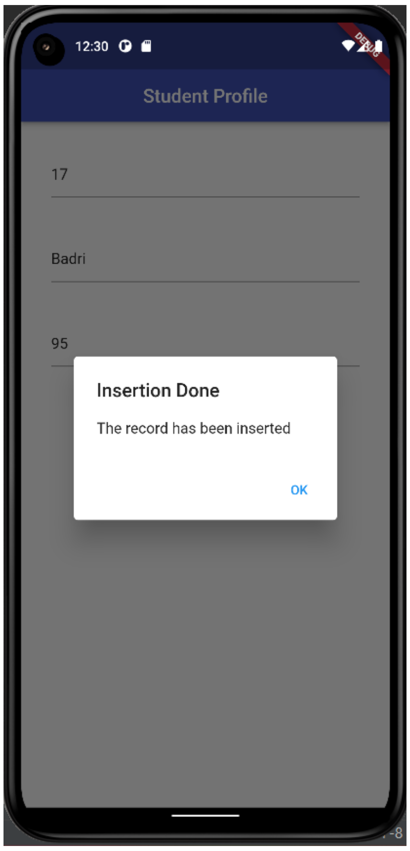
app.py

```
from flask import Flask, request, Response
import sqlite3, json

app = Flask(__name__)

db_locale="class.db"

@app.route("/view",methods=['GET'])
def view_rec():
```

```python
    if request.method=='GET':
        con=sqlite3.connect(db_locale)
        rno=request.headers["reg_no"]
        print(rno)
        c=con.cursor()
        sql_exec_str="SELECT * FROM student WHERE reg_no = ?"
        student_info=c.execute(sql_exec_str,[rno]).fetchall()
        con.commit()
        con.close()
        resp={}
        resp["reg_no"]=student_info[0][0]
        resp["name"]=student_info[0][1]
        resp["marks"]=student_info[0][2]
        fin_resp={}
        fin_resp["result"]=resp
        return json.dumps(fin_resp)
@app.route("/view_all",methods=['GET'])
def view_all_rec():
    if request.method=="GET":
        con=sqlite3.connect(db_locale)
        c=con.cursor()
        c.execute("""
            SELECT * FROM student
        """)
        students=c.fetchall()
        con.commit()
        con.close()
        res=[]
        final_res={}
        for student in students:
            resp={}
            resp["reg_no"]=student[0]
            resp["name"]=student[1]
            resp["marks"]=student[2]
            res.append(resp)
        final_res['result']=res
        return json.dumps(final_res)

@app.route("/add",methods=['POST'])
def add_rec():
    if request.method=='POST':
        con=sqlite3.connect(db_locale)
        c=con.cursor()
        c.execute("""
            INSERT INTO student(reg_no,name,marks)
            VALUES(?,?,?)
""",(request.headers["reg_no"],request.headers["name"],request.headers["marks"]
])
        )
        con.commit()
        con.close()
        resp={}
        return Response(status=200)

@app.route("/delete",methods=['DELETE'])
def delete_rec():
    if request.method=='DELETE':
        con=sqlite3.connect(db_locale)
        c=con.cursor()
        c.execute("""
            DELETE FROM student
            WHERE reg_no = ?
        """,([request.headers["reg_no"]])
        )
        con.commit()
        con.close()
        return Response(status=200)
@app.route("/update",methods=['PATCH'])
def update():
    if request.method=='PATCH':
        con=sqlite3.connect(db_locale)
        c=con.cursor()
        sql_exec_str="UPDATE student SET name = ?, marks=? WHERE reg_no =?"

c.execute(sql_exec_str,(request.headers['name'],request.headers['marks'],reque
st.headers['reg_no']))
        con.commit()
        con.close()
        return Response(status=200)
if __name__ == '__main__':
    app.run(host="0.0.0.0",port=5000)
```

Output:




12:30

Student Profile

17

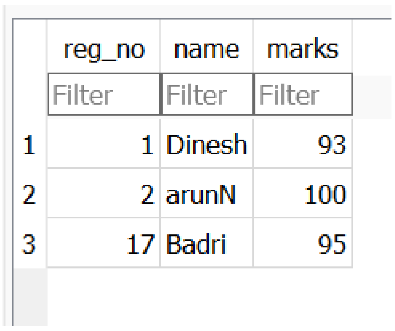
Badri

95

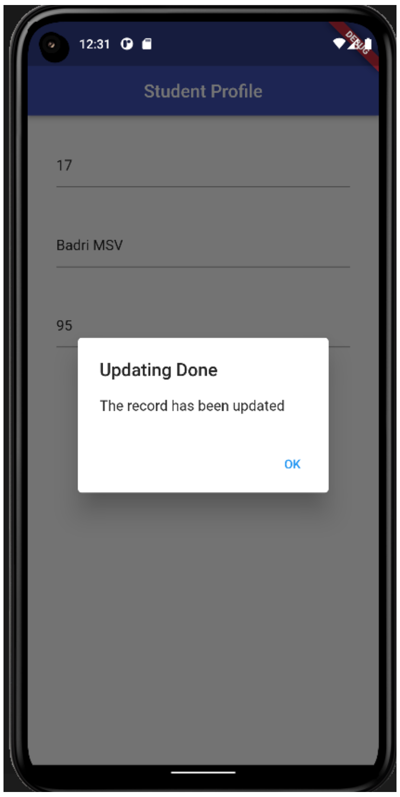
Insertion Done

The record has been inserted

OK



	reg_no	name	marks
	Filter	Filter	Filter
1	1	Dinesh	93
2	2	arunN	100
3	17	Badri	95


12:31

Student Profile

17

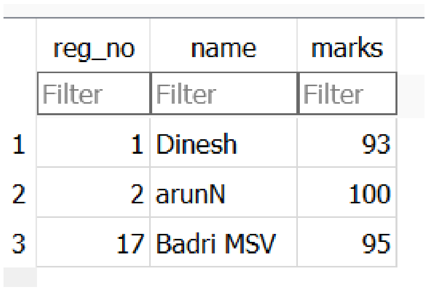
Badri MSV

95

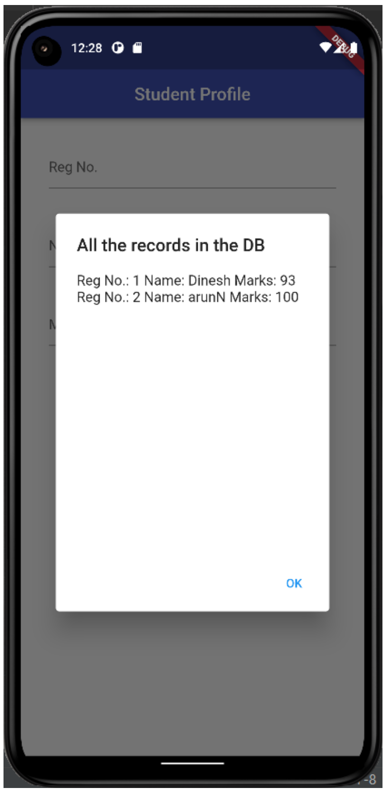
Updating Done

The record has been updated

OK

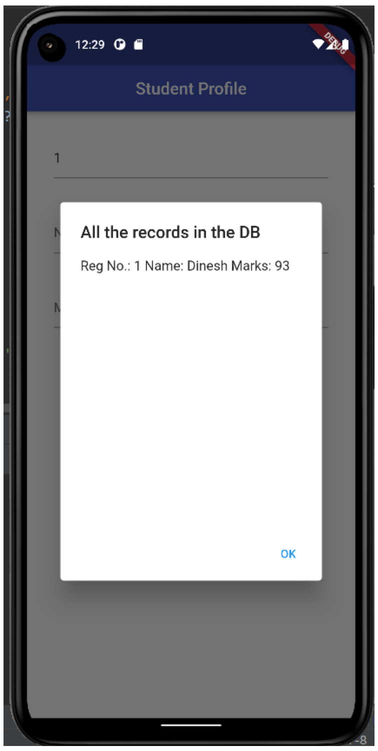


	reg_no	name	marks
	Filter	Filter	Filter
1	1	Dinesh	93
2	2	arunN	100
3	17	Badri MSV	95



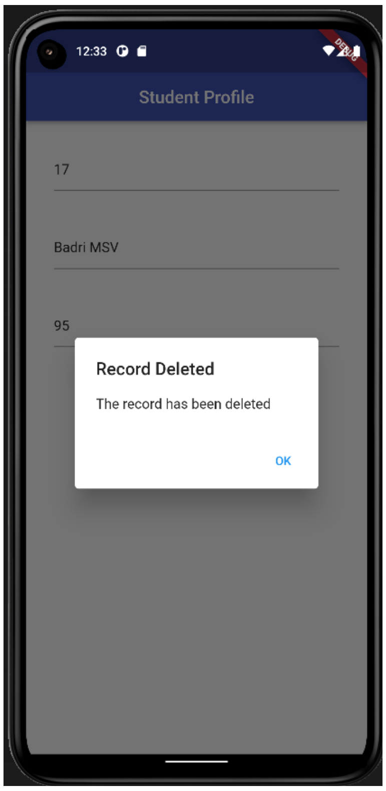
All the records in the DB

Reg No.: 1 Name: Dinesh Marks: 93
Reg No.: 2 Name: arunN Marks: 100



All the records in the DB

Reg No.: 1 Name: Dinesh Marks: 93


12:33

Student Profile

17

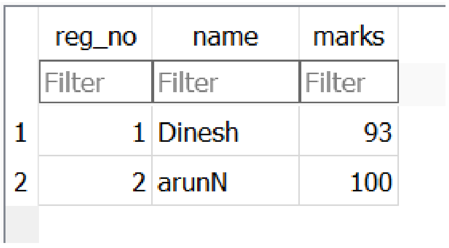
Badri MSV

95

Record Deleted

The record has been deleted

OK



	reg_no	name	marks
	Filter	Filter	Filter
1	1	Dinesh	93
2	2	arunN	100

Result:

CRUD operations are performed successfully upon connecting the mobile app to the database by using python Flask as the backend.

An application that makes use of RSS feed

Expt 6 **Date: / /2022**

Aim:

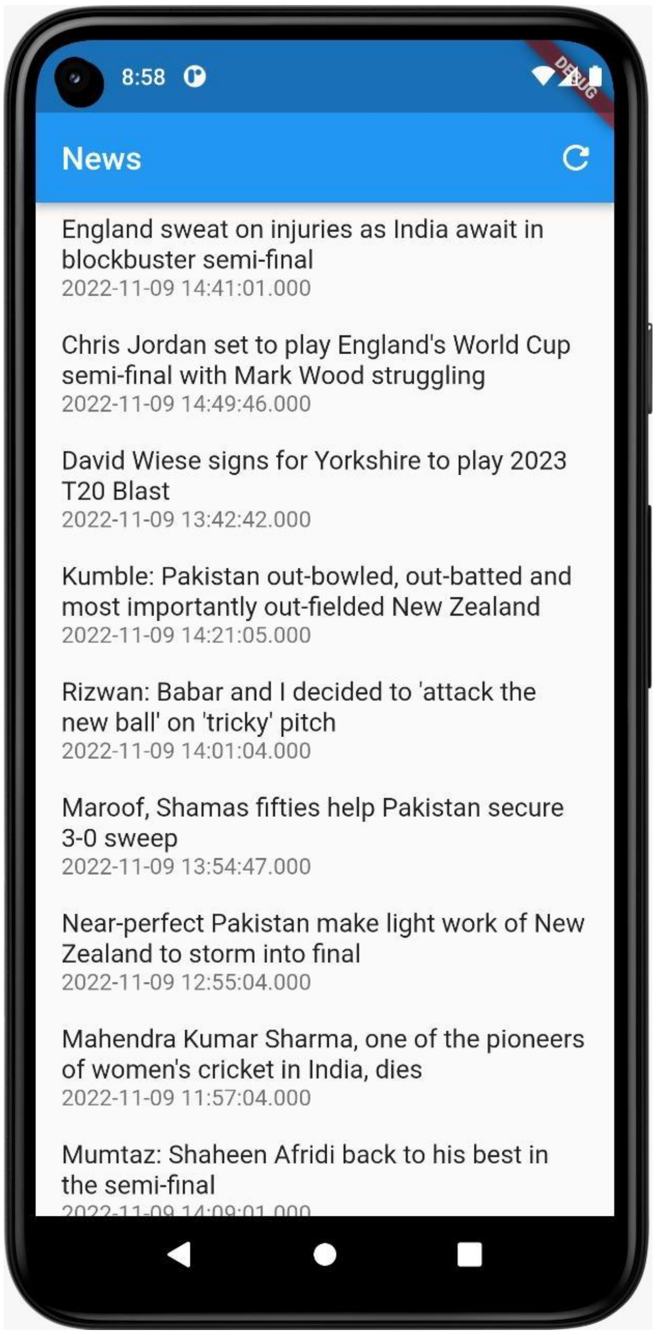
To create a mobile application that uses RSS feed.

Code:

Main.dart

```
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:webfeed/webfeed.dart';
import 'package:http/http.dart' as http;
import 'package:url_launcher/url_launcher.dart';
void main() {
  runApp(const RSSDemo());
}

class RSSDemo extends StatelessWidget {
  const RSSDemo({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return const MaterialApp(title: "RSS Feed", home: RSSMainPicture());
  }
}

class RSSMainPicture extends StatefulWidget {
  const RSSMainPicture({Key? key}) : super(key: key);

  @override
  State<RSSMainPicture> createState() => _RSSMainPictureState();
}

class _RSSMainPictureState extends State<RSSMainPicture> {
  late Future<RssFeed> result;
  Future<RssFeed> giver() async {
    var response =
    await
http.get(Uri.parse("https://www.espncricinfo.com/rss/content/story/feeds/0.
xml"));
    var channel = RssFeed.parse(response.body);
    return channel;
  }

  @override
  void initState() {
    super.initState();
    result = giver();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("News"),
        actions: [
          IconButton(onPressed: ()=>result=giver(), icon: const
Icon(Icons.refresh_rounded)),
        ],
      ),
      body: FutureBuilder<RssFeed?>(
        future: result,
        builder: (context,snapshot){
          if(snapshot.hasError){
            if(kDebugMode){
              print("Error");
            }
            return Container();
          }
          else if(snapshot.connectionState==ConnectionState.waiting){
            return const Center(
              child: CircularProgressIndicator(),
            );
          }
          else if(snapshot.hasData){
            var feed=snapshot.data!;
            var items=feed.items;
            return ListView.builder(
              itemCount: items?.length,
              itemBuilder: (context,index){
                var item=items![index];
                return GestureDetector(
                  onTap: () async{
                    if (!await launchUrl(Uri.parse(item.link!))) {
                      throw 'Could not launch ${item.link}';
                    }
                  },
                  child: ListTile(
                    // leading: CachedNetworkImage(
                    //   imageUrl: mediaImage!,
                    //   progressIndicatorBuilder: (context, url,
downloadProgress) =>
                    //       CircularProgressIndicator(value:
downloadProgress.progress),
                    //   errorWidget: (context, url, error) => const
Icon(Icons.error),
                    // ),
                    title: Text(item.title!),
                    subtitle: Text("${item.pubDate!}"),
                  ),
                );
              },
            );
          }
          return Container();
        },
      ),
    );
  }
}
```

Output:



England sweat on injuries as India await in blockbuster semi-final
2022-11-09 14:41:01.000

Chris Jordan set to play England's World Cup semi-final with Mark Wood struggling
2022-11-09 14:49:46.000

David Wiese signs for Yorkshire to play 2023 T20 Blast
2022-11-09 13:42:42.000

Kumble: Pakistan out-bowled, out-batted and most importantly out-fielded New Zealand
2022-11-09 14:21:05.000

Rizwan: Babar and I decided to 'attack the new ball' on 'tricky' pitch
2022-11-09 14:01:04.000

Maroof, Shamas fifties help Pakistan secure 3-0 sweep
2022-11-09 13:54:47.000

Near-perfect Pakistan make light work of New Zealand to storm into final
2022-11-09 12:55:04.000

Mahendra Kumar Sharma, one of the pioneers of women's cricket in India, dies
2022-11-09 11:57:04.000

Mumtaz: Shaheen Afridi back to his best in the semi-final
2022-11-09 14:09:01.000

Result:

RSS feed has been successfully integrated with the mobile app.

An application that implements multithreading

Expt 7 **Date: / /2022**

Aim:

To create a mobile application that implements multithreading.

Code:

main.dart

```
import 'home.dart';

import 'package:flutter/material.dart';


void main() {

 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: 'Flutter Demo',

     theme: ThemeData(

       primarySwatch: Colors.blue,

       brightness: Brightness.dark,
     ),

     home: const Home(),

   );

 }

}
```

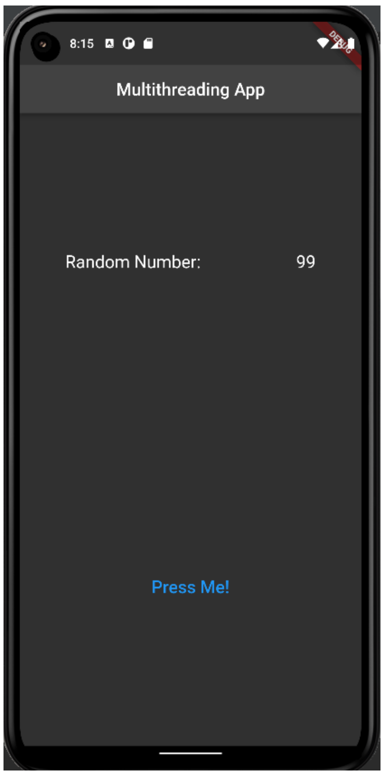
home.dart

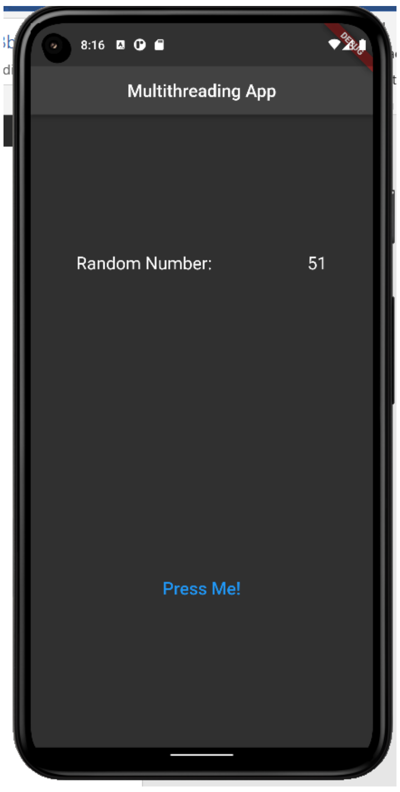
```
import 'dart:async';
import 'dart:math';

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

class Home extends StatefulWidget {
 const Home({Key? key}) : super(key: key);

 @override
 State<Home> createState() => _HomeState();
}

class _HomeState extends State<Home> {
 int randint = 99;
 static FutureOr<int> randGen(int cal) {
   var rng = Random();
   return rng.nextInt(100);
 }

 @override
 Widget build(BuildContext context) {
   return Scaffold(
     appBar: AppBar(
       title: Text(
         "Multithreading App",
       ),
       centerTitle: true,
```

```dart
    ),
    body: Column(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      children: <Widget>[
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceAround,
          children: [
            Text(
              "Random Number: ",
              style: TextStyle(
                fontSize: 20.0,
              ),
            ),
            Text(
              "${randint}",
              style: TextStyle(
                fontSize: 20.0,
              ),
            ),
          ],
        ),
        SizedBox(
          height: 20.0,
        ),
        TextButton(
          onPressed: () async {
            int result = await compute(randGen, randint);
            setState(() {
              randint = result;
            });
          },
          child: Text(
            "Press Me!",
            style: TextStyle(
              fontSize: 20.0,
            ),
          ),
        ),
      ],
    ),
  );
}
```

```
}
```

Output:





Result:
An android application that implements multithreading has been developed and executed successfully.

An application that uses GPS location information

Expt 8 **Date: / /2022**

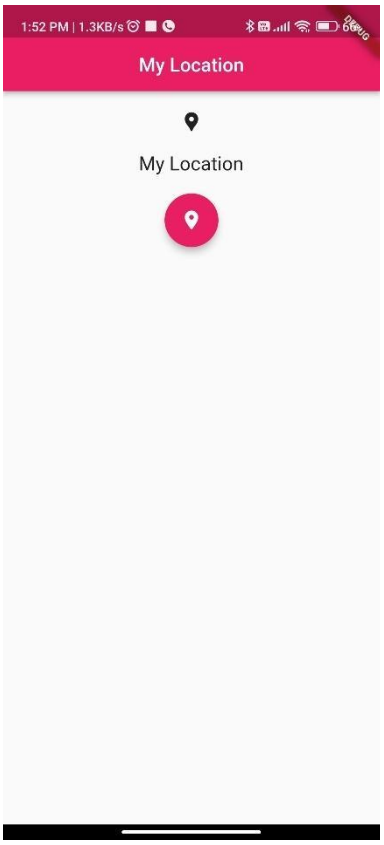
Aim:

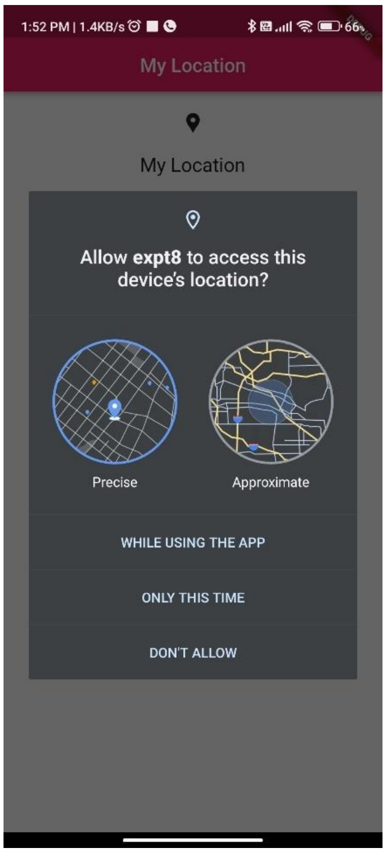
To create a mobile application that uses GPS location information.

Code:

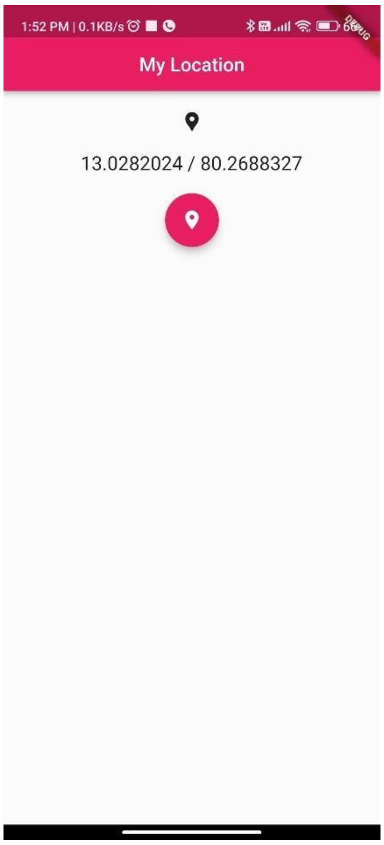
```dart
import 'package:flutter/material.dart';
import 'package:location/location.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: Colors.pink,
      ),
      home: const Home(),
    );
  }
}
class Home extends StatelessWidget {
  const Home({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text(
          "My Location"
        ),
        centerTitle: true,
      ),
      body: const LocationInfo(

      ),
      floatingActionButtonLocation:
FloatingActionButtonLocation.centerDocked,
    );
  }
}

class LocationInfo extends StatefulWidget {
  const LocationInfo({Key? key}) : super(key: key);

  @override
  State<LocationInfo> createState() => _LocationInfoState();
}

class _LocationInfoState extends State<LocationInfo> {
  String _myLoc ="My Location";
  Location location=new Location();
  late bool _serviceEnabled;
  late PermissionStatus _permissionGranted;
  late LocationData _locationData;
  bool _isListenLocation =false, _isGetLocation = false;

  @override
  Widget build(BuildContext context) {
    return Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: <Widget>[
        const SizedBox(
          height: 20.0,
        ),
        const Icon(
          Icons.location_pin,
        ),
        const SizedBox(
          height: 20.0,
        ),
        Center(
          child: Text(
            "$_myLoc",
            style: TextStyle(
              fontSize: 20.0,
            ),
          ),
        ),
        const SizedBox(
          height: 20.0,
        ),
        FloatingActionButton(
            child: Icon(
              Icons.location_on_sharp,
            ),
            onPressed: updateLoc,
        ),
      ],
    );
  }
  void updateLoc() async{
    _serviceEnabled = await location.serviceEnabled();
    if(!_serviceEnabled){
      _serviceEnabled = await location.requestService();
      if(_serviceEnabled)
        return;
    }
    _permissionGranted = await location.hasPermission();
    if(_permissionGranted == PermissionStatus.denied){
      _permissionGranted = await location.requestPermission();
      if(_permissionGranted != PermissionStatus.granted)
        return;
    }
    _locationData = await location.getLocation();
    setState(() {
      _isGetLocation = true;
    });
    if(_isGetLocation){
      _myLoc="${_locationData.latitude} / ${_locationData.longitude}";
    }
  }
}
```

Output:





Allow **expt8** to access this device's location?



13.0282024 / 80.2688327

Result:

A native application that uses GPS location has been developed and executed successfully.

An application that takes advantage of rich gesture-based UI handling

Expt 9 **Date: / /2022**

Aim:

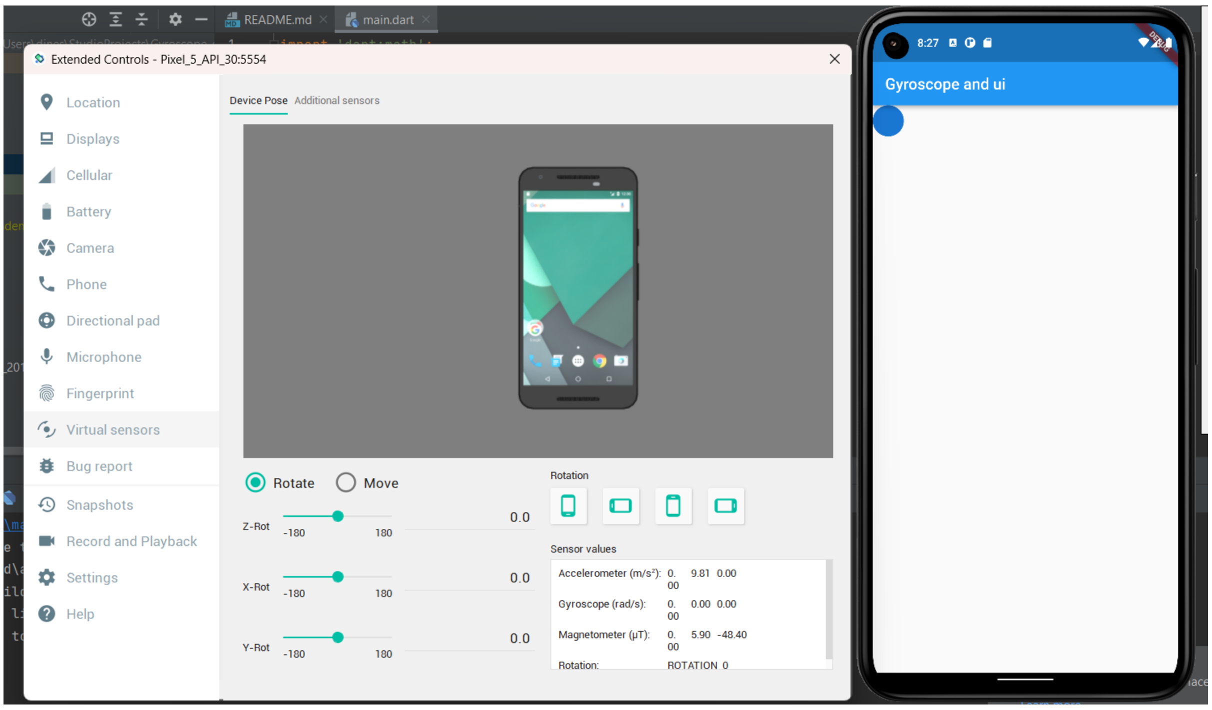
To create a mobile application that will take advantage of underlying phone functionality including rich gesture-based UI handling

Code:

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

void main() {
  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: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // Try running your application with "flutter run". You'll see the
        // application has a blue toolbar. Then, without quitting the app, try
        // changing the primarySwatch below to Colors.green and then invoke
        // "hot reload" (press "r" in the console where you ran "flutter run",
        // or simply save your changes to "hot reload" in a Flutter IDE).
        // Notice that the counter didn't reset back to zero; the application
        // is not restarted.
        primarySwatch: Colors.blue,
      ),
      home: const MyHomePage(title: 'Gyroscope and ui'),
    );
  }
}

class MyHomePage extends StatefulWidget {
```

```dart
  const MyHomePage({super.key, required this.title});

  // This widget is the home page of your application. It is stateful, meaning
  // that it has a State object (defined below) that contains fields that affect
  // how it looks.

  // This class is the configuration for the state. It holds the values (in this
  // case the title) provided by the parent (in this case the App widget) and
  // used by the build method of the State. Fields in a Widget subclass are
  // always marked "final".

  final String title;

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

class _MyHomePageState extends State<MyHomePage> {
  double _dx = 0,
      _dy = 0;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(widget.title),
      ),
      body: StreamBuilder<GyroscopeEvent>(
        stream: SensorsPlatform.instance.gyroscopeEvents,
        builder: (context, snapshot) {
          if (snapshot.hasData) {
            _dy = _dy + snapshot.data!.y * 10;
            _dx = _dx + snapshot.data!.x * 10;
          }
          return Stack(
            children: [
              Positioned(
                top: _dy,
                left: _dx,
                child: GestureDetector(
                  onPanUpdate: (details) {
                    setState(() {
                      _dy = max(0, _dy + details.delta.dy);
                      _dx = max(0, _dx + details.delta.dx);
                    });
                  },
                  child: const CircleAvatar(),
                ),
              )
            ],
          );
        },
      ),
    );
  }
}
```

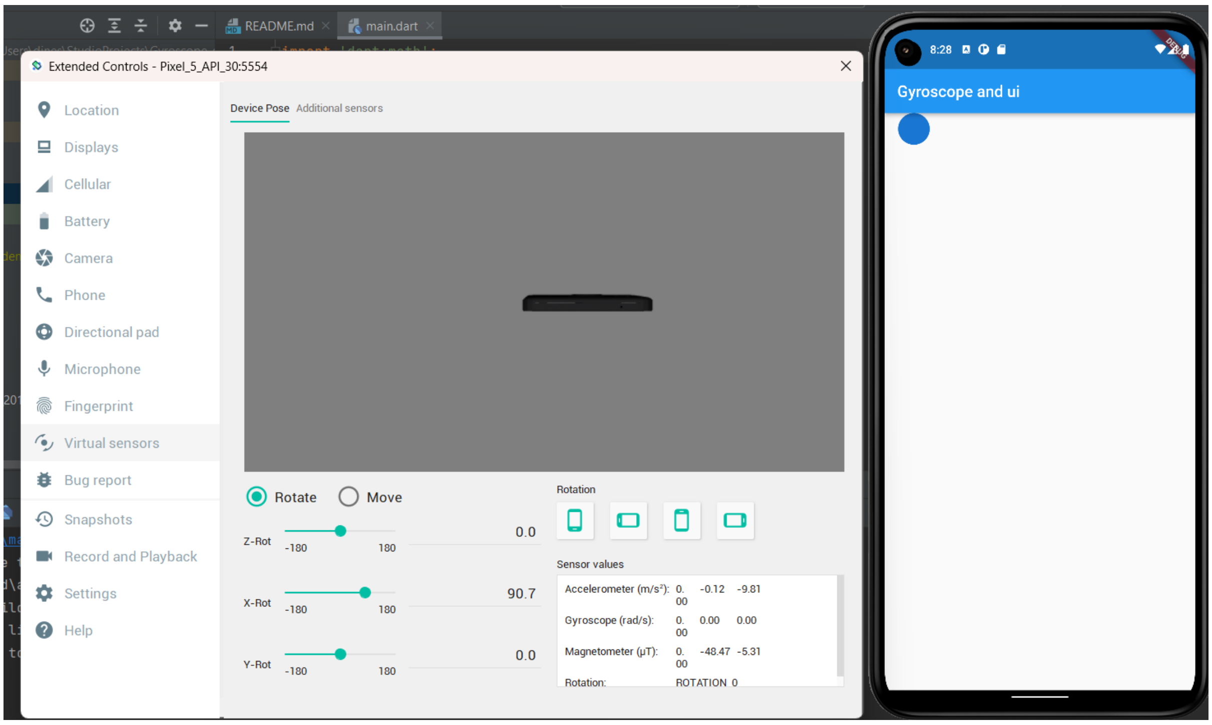
Output:



Rotate [✓] Move

```
Z-Rot  -180  180   0.0
X-Rot  -180  180   0.0
Y-Rot  -180  180   0.0
```

```
Accelerometer (m/s²):  0.   9.81  0.00
                       00
Gyroscope (rad/s):     0.   0.00  0.00
                       00
Magnetometer (μT):     0.   5.90  -48.40
                       00
Rotation:              ROTATION_0
```



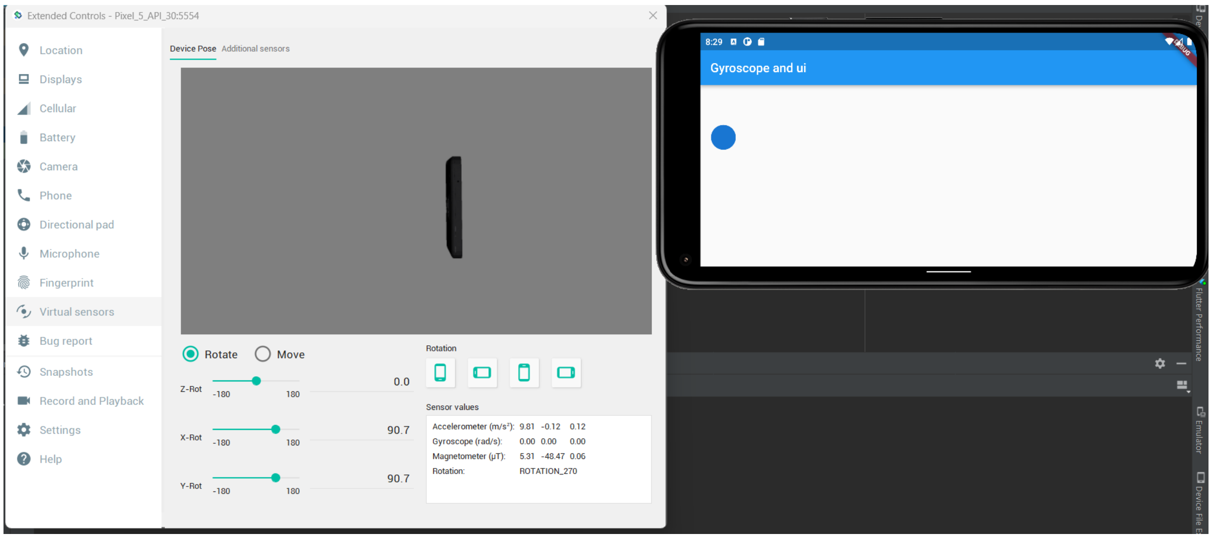
Rotate [✓] Move

Z-Rot -180 180 0.0

X-Rot -180 180 90.7

Y-Rot -180 180 0.0

```
Accelerometer (m/s²):  0.00  -0.12   -9.81
Gyroscope (rad/s):     0.00   0.00    0.00
Magnetometer (μT):     0.00  -48.47  -5.31
Rotation:              ROTATION_0
```



Rotate [✓] Move

Z-Rot -180 180 0.0

X-Rot -180 180 90.7

Y-Rot -180 180 90.7

```
Accelerometer (m/s²):  9.81  -0.12   0.12
Gyroscope (rad/s):     0.00   0.00   0.00
Magnetometer (μT):     5.31  -48.47  0.06
Rotation:              ROTATION_270
```

Result:

A mobile application that uses rich gestures to handle UI was developed and executed successfully.

An application that creates an alert upon user action

Expt 10 **Date: / /2022**

Aim:

To create an application that sends an alert upon user action.

Code:

main.dart

```dart
import 'package:expt10/pages/home.dart';
import 'package:flutter/material.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Experiment 10',
      theme: ThemeData.dark(),

      home: const Home(),
    );
  }
}
```

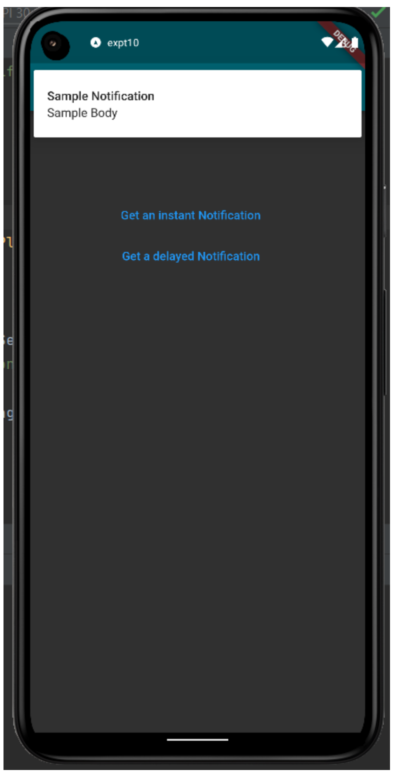
home.dart

```
import 'package:expt10/services/local_notification_service.dart';
import 'package:flutter/material.dart';

class Home extends StatefulWidget {
  const Home({Key? key}) : super(key: key);

  @override
  State<Home> createState() => _HomeState();
}

class _HomeState extends State<Home> {
  late final LocalNotificationService service;
  @override
  void initState(){
    service = LocalNotificationService();
    service.initialize();
    super.initState();
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text(
          "Local Notifications Expt"
        ),
        backgroundColor: const Color(0xff006473),
        centerTitle: true,
      ),
      body: Padding(
        padding: EdgeInsets.all(MediaQuery.of(context).size.width*0.25),
        child: Column(
          children: <Widget>[
            TextButton(
                onPressed: () async {
                  await service.showNotification(
                      id: 0,
                      title: "Sample Notification",
                      body: "Sample Body"
                  );
                },
                child: const Text(
                  "Get an instant Notification"
                ),
            ),
            TextButton(
              onPressed: () async {
                await service.showScheduledNotification(
                    id: 0,
                    title: "Sample Notification",
                    body: "Sample Body",
                    seconds: 4,
                );
              },
              child: const Text(
                  "Get a delayed Notification"
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

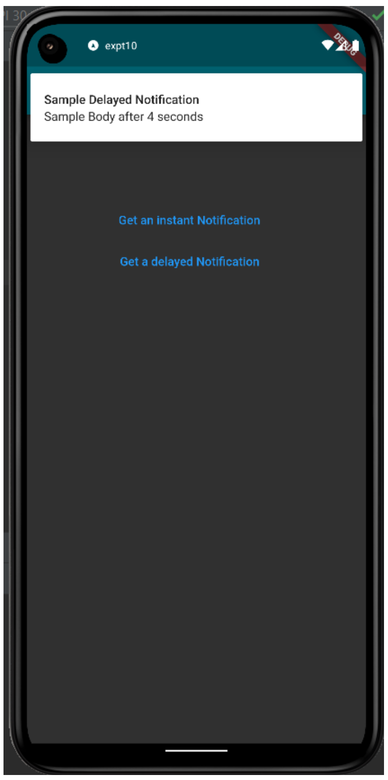
local_notification_service.dart

```
import 'package:flutter_local_notifications/flutter_local_notifications.dart';
import 'package:timezone/timezone.dart' as tz;
import 'package:timezone/data/latest.dart' as tz;

class LocalNotificationService {
  LocalNotificationService();

  final _localNotificationService = FlutterLocalNotificationsPlugin();

  Future<void> initialize() async{
    tz.initializeTimeZones();
    const AndroidInitializationSettings androidInitializationSettings =
    AndroidInitializationSettings('ic_stat_assistant_navigation');

    const DarwinInitializationSettings iosInitializationSettings =
        DarwinInitializationSettings(
          requestAlertPermission: true,
          requestBadgePermission: true,
          requestSoundPermission: true,
        );
    const InitializationSettings settings = InitializationSettings(
        android: androidInitializationSettings,
        iOS: iosInitializationSettings
    );

    await _localNotificationService.initialize(settings);
  }
  Future<NotificationDetails> _notificationDetails() async{
    const AndroidNotificationDetails androidNotificationDetails =
AndroidNotificationDetails(
        "channel_id", "channel_name",
      channelDescription: "Description",
      importance: Importance.max,
      priority: Priority.max,
      playSound: true,
    );
    const DarwinNotificationDetails darwinNotificationDetails =
DarwinNotificationDetails();
    return const NotificationDetails(android: androidNotificationDetails,iOS:
darwinNotificationDetails);
  }
  Future<void> showNotification({
    required int id,
    required String title,
    required String body}) async{
      final details = await _notificationDetails();
      await _localNotificationService.show(id, title, body, details);
  }
  Future<void> showScheduledNotification({
    required int id,
    required String title,
    required String body,
    required int seconds
  }) async{
    final details = await _notificationDetails();
    await _localNotificationService.zonedSchedule(
      id,
      title,
      body,
      tz.TZDateTime.from(DateTime.now().add(Duration(seconds: seconds)),
tz.local,),
      details,
      androidAllowWhileIdle: true,
      uiLocalNotificationDateInterpretation:
UILocalNotificationDateInterpretation.absoluteTime
    );
  }
}
```

Output:


expt10

Sample Notification
Sample Body


expt10

Sample Delayed Notification
Sample Body after 4 seconds

Result:

An application that sends an alert upon user action was developed and executed successfully.

An application that creates an alarm clock

Expt 11 **Date: / /2022**

Aim:

To create an application that creates an alarm clock.

Code:

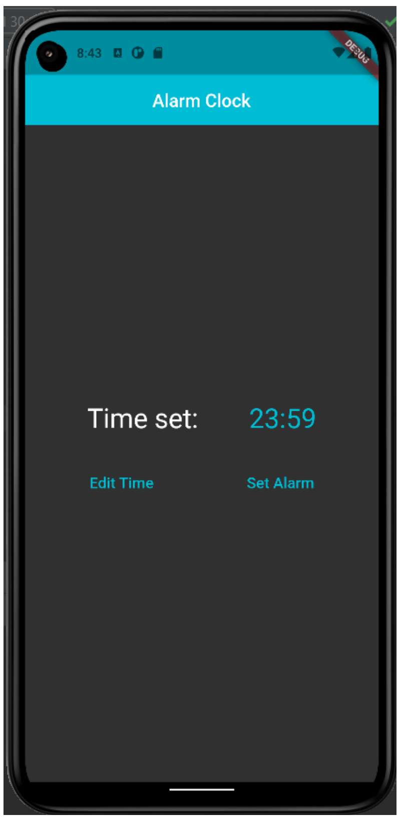
main.dart

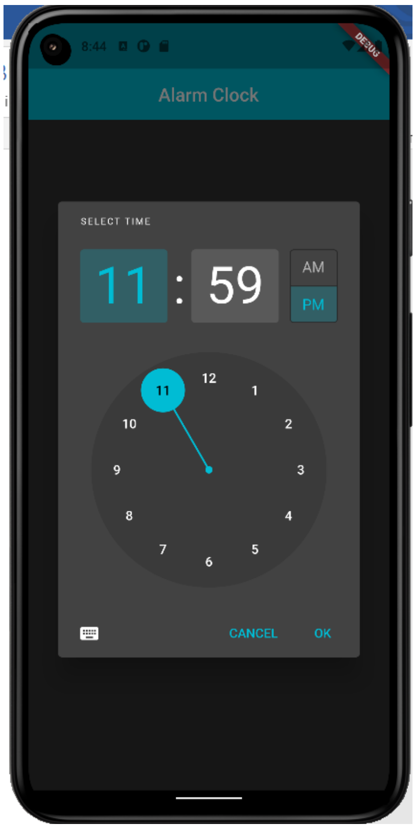
```dart
import 'package:flutter/material.dart';
import 'pages/home.dart';
void main() {
  runApp(const MyApp());
}

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

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: Colors.cyan,
        brightness: Brightness.dark,
      ),
      home: const Home(),
    );
  }
}
```

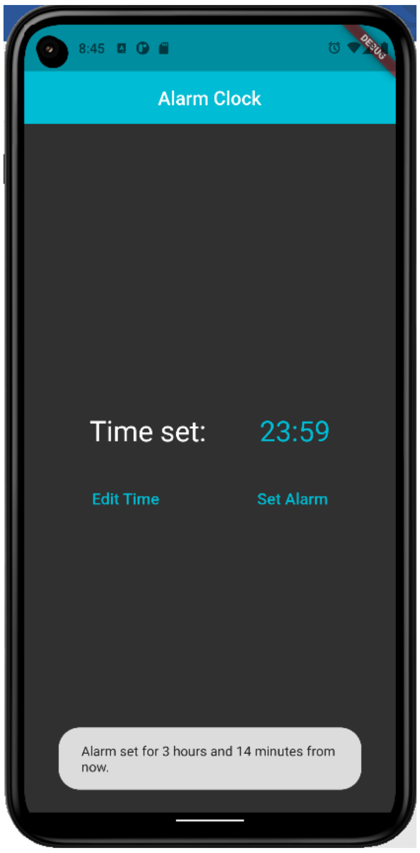
home.dart

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

class Home extends StatefulWidget {
  const Home({Key? key}) : super(key: key);

  @override
  State<Home> createState() => _HomeState();
}

class _HomeState extends State<Home> {
  TimeOfDay time= TimeOfDay(hour: 23, minute: 59);
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
          "Alarm Clock",
        ),
        centerTitle: true,
        elevation: 0.0,
        backgroundColor: Colors.cyan,
      ),
      body: Padding(
        padding: EdgeInsets.all(20),
        child: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              Row(
                mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                children: [
                  Text(
                    "Time set: ",
                    style: TextStyle(
                      fontSize: 30.0,
                    ),
                  ),
                  Text(

"${time.hour.toString().padLeft(2,'0')}:${time.minute.toString().padLeft(2,'0')}",
                    style: TextStyle(
                      fontSize: 30.0,
                      color: Colors.cyan,
                    ),
                  )
                ],
              ),
              SizedBox(
                height: 30.0,
              ),
              Row(
                mainAxisAlignment: MainAxisAlignment.spaceAround,
                children: [
                  TextButton(
                    onPressed: () async{
                      TimeOfDay? newTime = await showTimePicker(
                          context: context,
                          initialTime: time,
                      );
                      if(newTime == null) return;
                      setState(() {
                        time = newTime;
                      });
                    },
                    child: Text(
                      "Edit Time",
                      style: TextStyle(
                        fontSize: 17.0,
                      ),
                    ),
                  ),
                  TextButton(
                    onPressed: () {
                      FlutterAlarmClock.createAlarm(time.hour,time.minute);
                    },
                    child: Text(
                      "Set Alarm",
                      style: TextStyle(
                        fontSize: 17.0,
                      ),
                    ),
                  ),
                ],
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

Output:

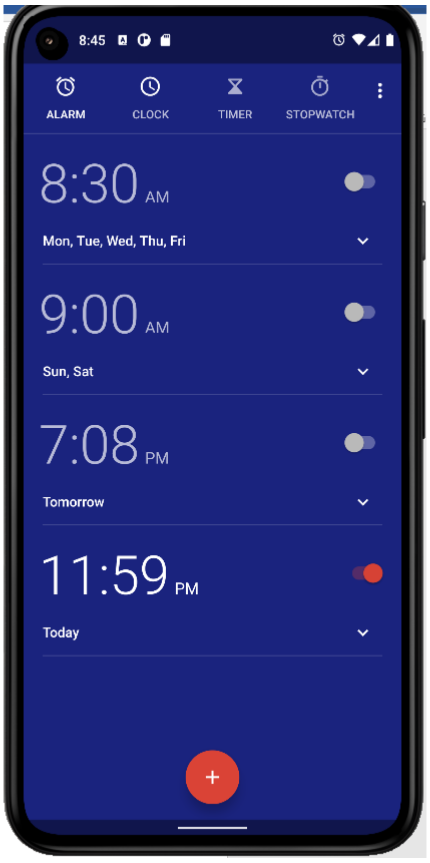

8:43
Alarm Clock
Time set:
23:59
Edit Time
Set Alarm


8:44
Alarm Clock
SELECT TIME
11
:
59
AM
PM
12
1
2
3
4
5
6
7
8
9
10
11
CANCEL
OK



Time set: 23:59

Alarm set for 3 hours and 14 minutes from now.



8:30 AM
Mon, Tue, Wed, Thu, Fri

9:00 AM
Sun, Sat

7:08 PM
Tomorrow

11:59 PM
Today

Result:

An application that creates an alarm clock is developed and tested successfully.

An application that performs REST-based API calls

Expt 12 **Date: / /2022**

Aim:

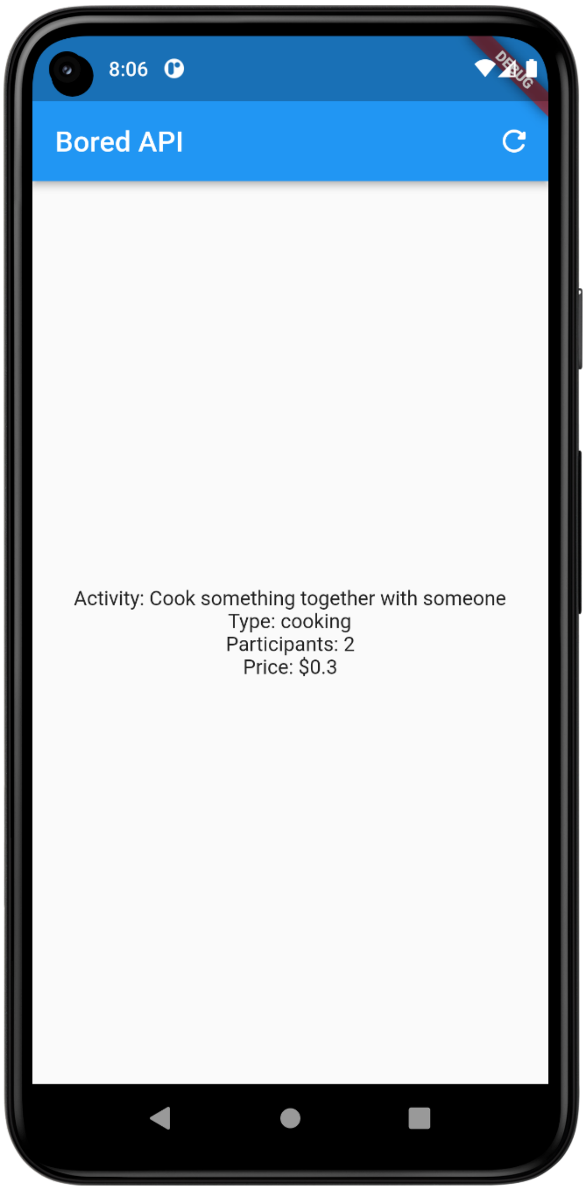
To create an application that performs REST-based API calls.

Code:

main.dart

```dart
import 'dart:convert';

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

void main() {
  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: 'Api Calls',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // Try running your application with "flutter run". You'll see the
        // application has a blue toolbar. Then, without quitting the app, try
        // changing the primarySwatch below to Colors.green and then invoke
        // "hot reload" (press "r" in the console where you ran "flutter run",
        // or simply save your changes to "hot reload" in a Flutter IDE).
        // Notice that the counter didn't reset back to zero; the application
        // is not restarted.
        primarySwatch: Colors.blue,
      ),
      home: const MyHomePage(title: 'Codeforces Problem Set'),
    );
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key, required this.title});

  // This widget is the home page of your application. It is stateful, meaning
  // that it has a State object (defined below) that contains fields that affect
  // how it looks.
  // This class is the configuration for the state. It holds the values (in this
  // case the title) provided by the parent (in this case the App widget) and
  // used by the build method of the State. Fields in a Widget subclass are
  // always marked "final".

  final String title;

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

class _MyHomePageState extends State<MyHomePage> {
  late Future<Map<String,dynamic>> info;
  @override
  void initState(){
    info=giver();
    super.initState();
  }

  Future<Map<String,dynamic>> giver() async{
    var response = await
http.get(Uri.parse("https://www.boredapi.com/api/activity"));
    Map<String,dynamic> result=json.decode(response.body);
    //print(result);
    return result;
  }
  @override
  Widget build(BuildContext context){
    return Scaffold(
      appBar: AppBar(
        title: const Text("Bored API"),
        actions: [
          IconButton(onPressed: ()=>setState(() {
            info=giver();
          }), icon: const Icon(Icons.refresh_rounded))
        ],
      ),
      body: FutureBuilder<Map<String,dynamic>>(
        future: info,
        builder: (context,snapshot){
          if(snapshot.connectionState==ConnectionState.waiting){
            return const Center(child: CircularProgressIndicator());
          }
          Map<String,dynamic> data={};
          if(snapshot.hasData){
            data=snapshot.data!;
            return Center(
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Text("Activity: ${data["activity"]}"),
                  Text("Type: ${data["type"]}"),
                  Text("Participants: ${data["participants"]}"),
                  Text("Price: \$${data["price"]}"),
                ],
              ),
            );
          }
          return Container();
        },
      ),
    );
  }
}
```

Output:



```
Activity: Cook something together with someone
Type: cooking
Participants: 2
Price: $0.3
```

Result:

An application that performs REST-based API calls is developed and tested successfully.