Mobile Programming Practical

Practical No. 1

Program to demonstrate the features of Dart language.

Dart is an open-source general-purpose programming language developed by Google. It supports application development in both client and server-side. But it is widely used for the development of android apps, iOS apps, IoT(Internet of Things), and web applications using the Flutter Framework.

DartPad is an online code editor for the Dart language. In addition to executing regular Dart programs, it can run Flutter programs and show graphic output.

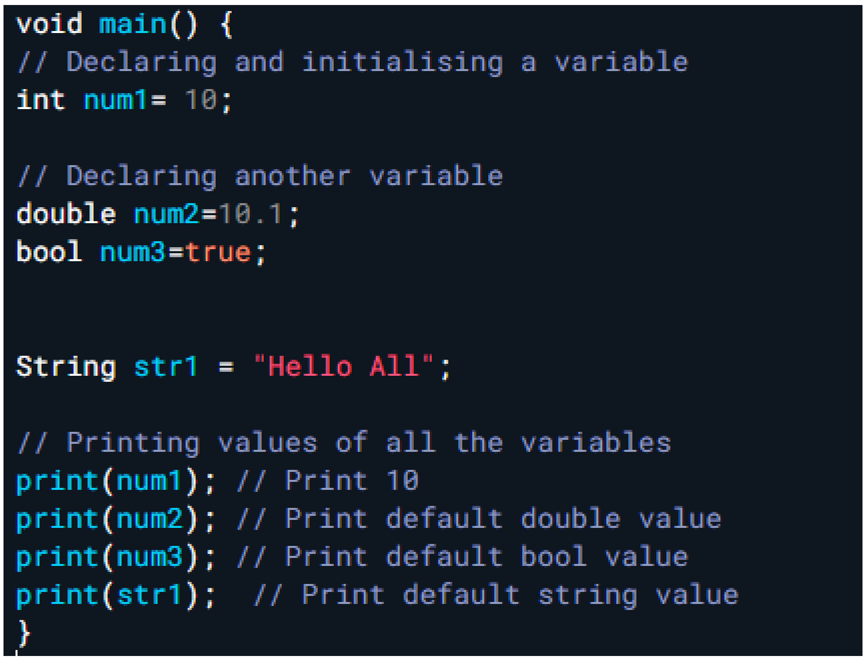
```
void main() {
  print("Hello World!");
}
```

The main() function is a predefined method in Dart. This method acts as the entry point to the application. A Dart script needs the main() method for execution. print() is a predefined function that prints the specified string or value to the standard output i.e. the terminal.

Type of the variable:

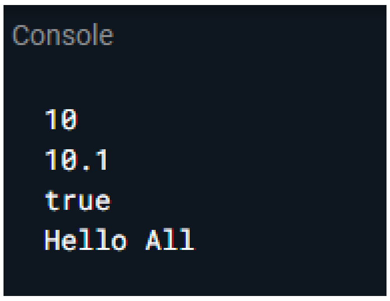
1. Integer
2. Double
3. String
4. Booleans
5. Lists
6. Map

CODE:



```
void main() {
// Declaring and initialising a variable
int num1= 10;

// Declaring another variable
double num2=10.1;
bool num3=true;


String str1 = "Hello All";

// Printing values of all the variables
print(num1); // Print 10
print(num2); // Print default double value
print(num3); // Print default bool value
print(str1);  // Print default string value
}
```

OUTPUT:

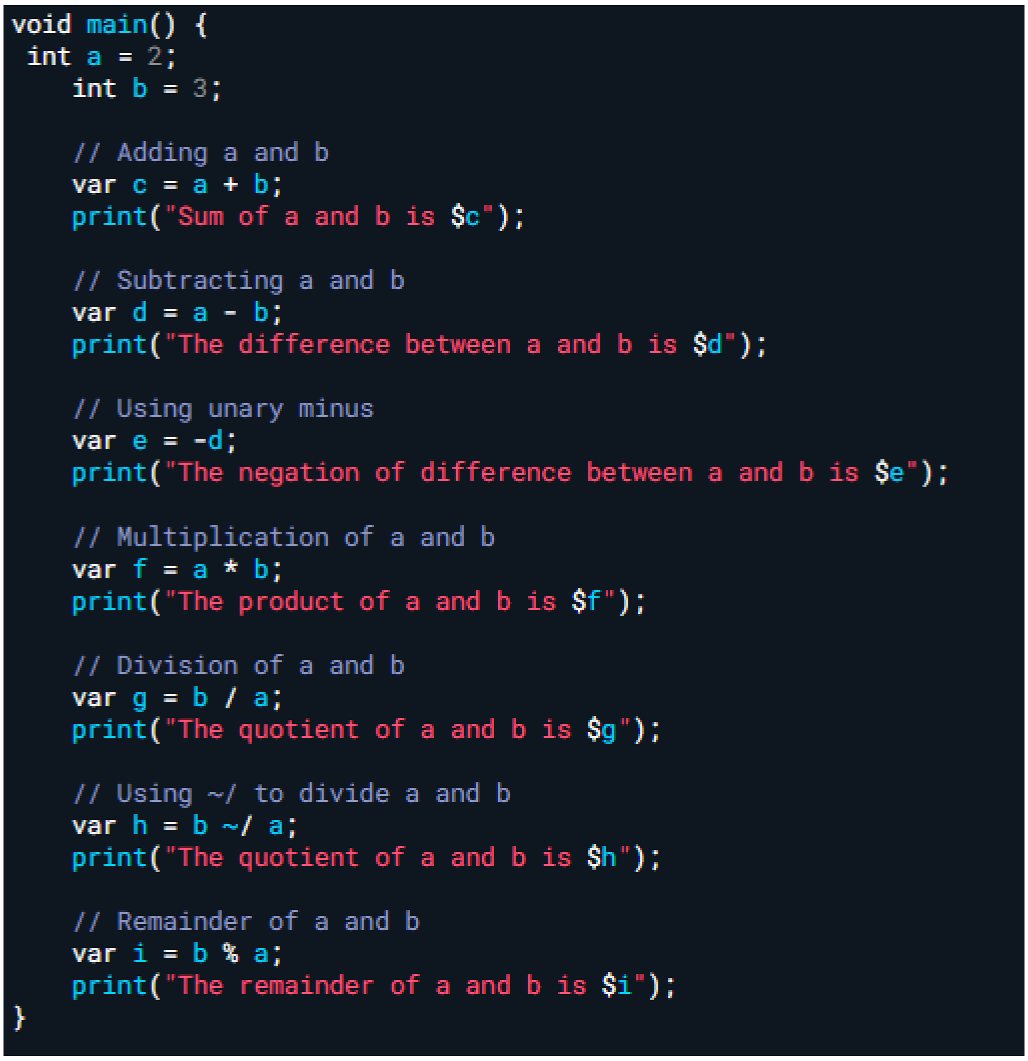


```
Console

  10
  10.1
  true
  Hello All
```

Different types of operators in Dart:

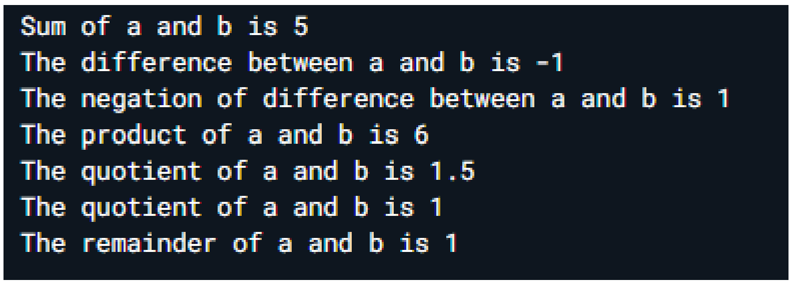
1. Arithmetic Operators
2. Relational Operators
3. Type Test Operators
4. Bitwise Operators
5. Assignment Operators
6. Logical Operators
7. Conditional Operator
8. Cascade Notation Operator

CODE:



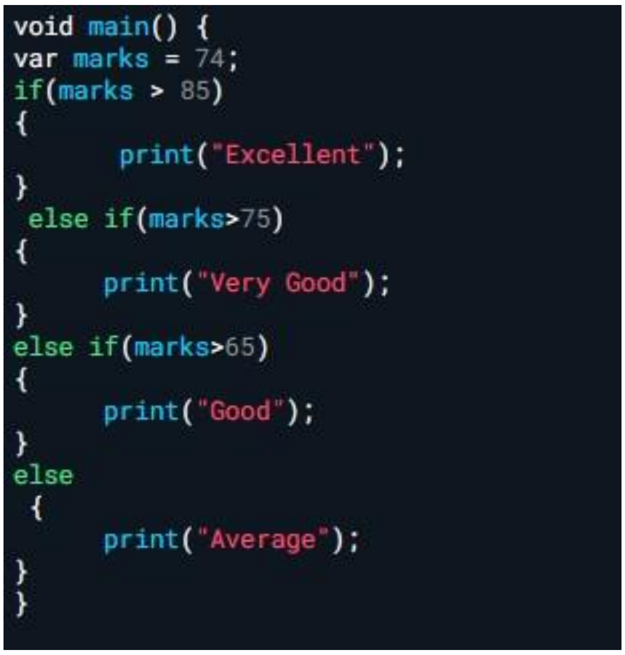
```dart
void main() {
 int a = 2;
    int b = 3;

    // Adding a and b
    var c = a + b;
    print("Sum of a and b is $c");

    // Subtracting a and b
    var d = a - b;
    print("The difference between a and b is $d");

    // Using unary minus
    var e = -d;
    print("The negation of difference between a and b is $e");

    // Multiplication of a and b
    var f = a * b;
    print("The product of a and b is $f");

    // Division of a and b
    var g = b / a;
    print("The quotient of a and b is $g");

    // Using ~/ to divide a and b
    var h = b ~/ a;
    print("The quotient of a and b is $h");

    // Remainder of a and b
    var i = b % a;
    print("The remainder of a and b is $i");
}
```

OUTPUT:



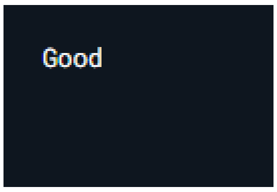
```
Sum of a and b is 5
The difference between a and b is -1
The negation of difference between a and b is 1
The product of a and b is 6
The quotient of a and b is 1.5
The quotient of a and b is 1
The remainder of a and b is 1
```

Decision-making statements are those statements which allow the programmers to decide which statement should run in different conditions.



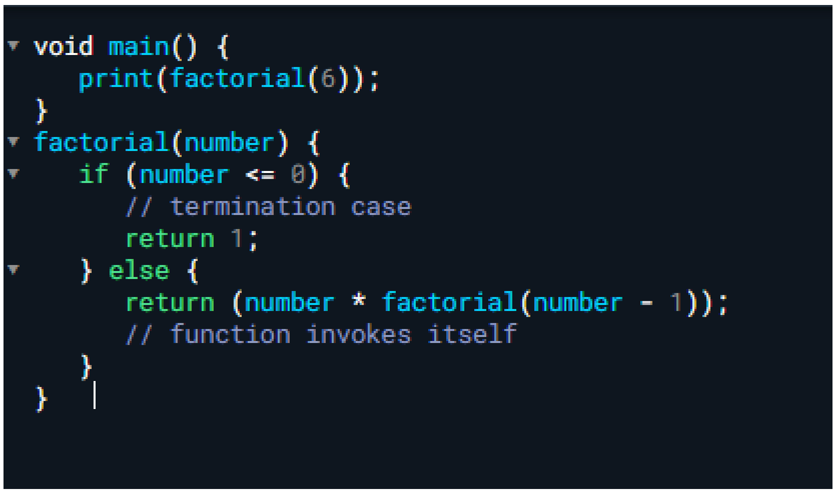
```
void main() {
var marks = 74;
if(marks > 85)
{
       print("Excellent");
}
 else if(marks>75)
{
      print("Very Good");
}
else if(marks>65)
{
      print("Good");
}
else
 {
      print("Average");
}
}
```

Output:



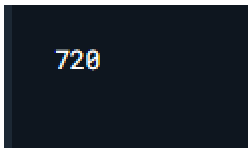
```
Good
```

Function is a set of statements that take inputs, do some specific computation and produces output. Functions are created when certain statements are repeatedly occurring in the program and a function is created to replace them. Functions make it easy to divide the complex program into smaller sub-groups and increase the code reusability of the program.



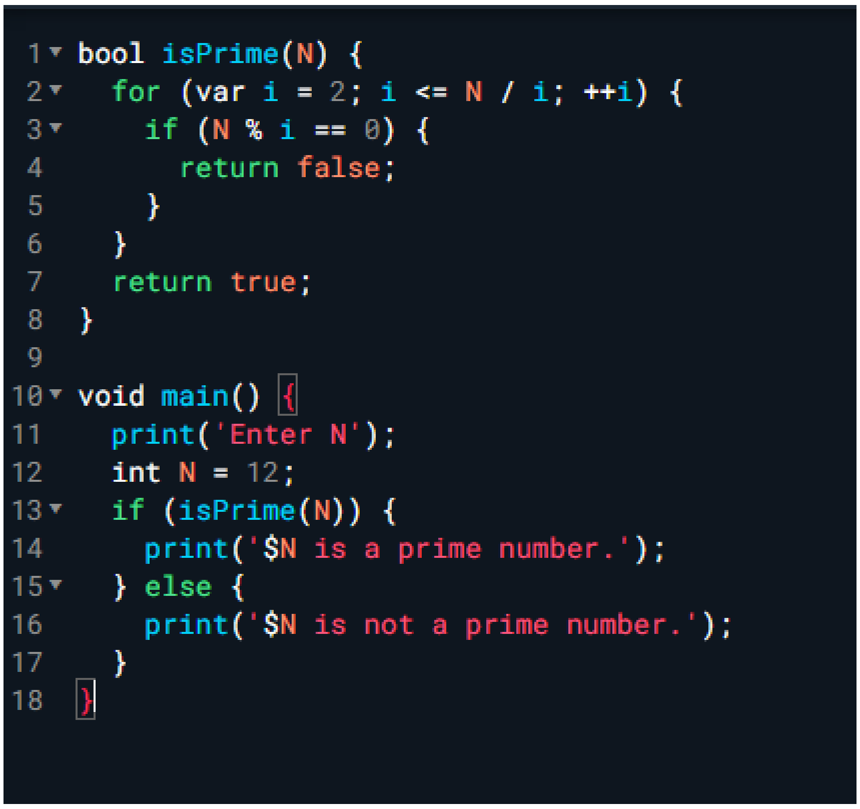
```
void main() {
   print(factorial(6));
}
factorial(number) {
   if (number <= 0) {
      // termination case
      return 1;
   } else {
      return (number * factorial(number - 1));
      // function invokes itself
   }
}
```

Output:



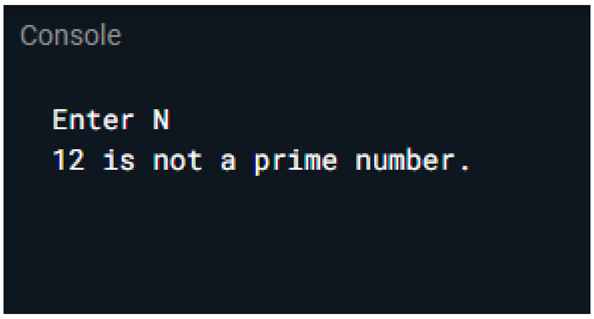
```
720
```

DART Prime code



```dart
bool isPrime(N) {
  for (var i = 2; i <= N / i; ++i) {
    if (N % i == 0) {
      return false;
    }
  }
  return true;
}

void main() {
  print('Enter N');
  int N = 12;
  if (isPrime(N)) {
    print('$N is a prime number.');
  } else {
    print('$N is not a prime number.');
  }
}
```

OUTPUT



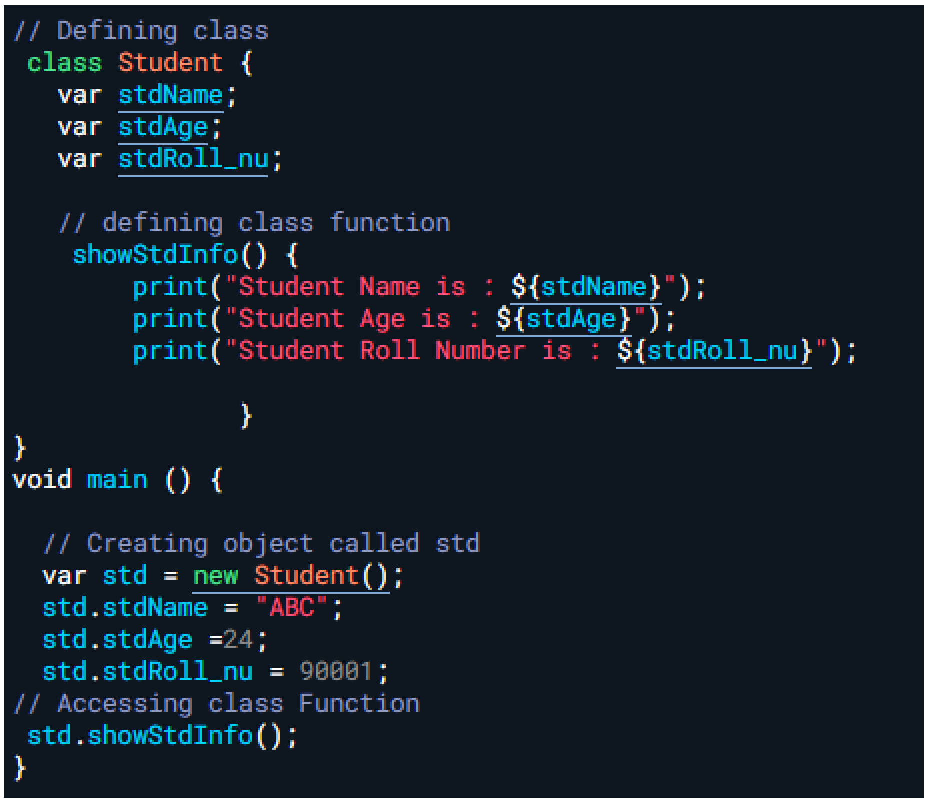
Console

```
Enter N
12 is not a prime number.
```

Dart is an object-oriented language. It supports object-oriented programming features like classes, interfaces, etc. A class in terms of OOP is a blueprint for creating objects. A class encapsulates data for the object. Dart gives built-in support for this concept called class.

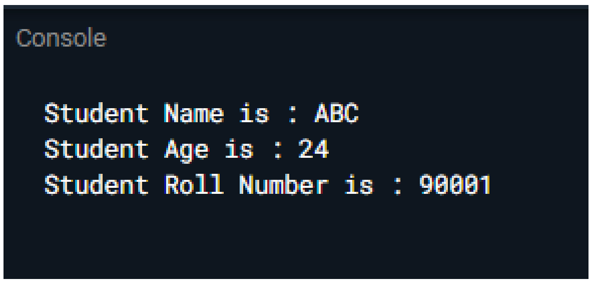
Declaring a Class

Use the class keyword to declare a class in Dart. A class definition starts with the keyword class followed by the class name; and the class body enclosed by a pair of curly braces.



```
// Defining class
 class Student {
   var stdName;
   var stdAge;
   var stdRoll_nu;

   // defining class function
    showStdInfo() {
        print("Student Name is : ${stdName}");
        print("Student Age is : ${stdAge}");
        print("Student Roll Number is : ${stdRoll_nu}");

               }
}
void main () {

  // Creating object called std
  var std = new Student();
  std.stdName = "ABC";
  std.stdAge =24;
  std.stdRoll_nu = 90001;
// Accessing class Function
 std.showStdInfo();
}
```

OUTPUT:



```
Console

  Student Name is : ABC
  Student Age is : 24
  Student Roll Number is : 90001
```

Practical No. 2

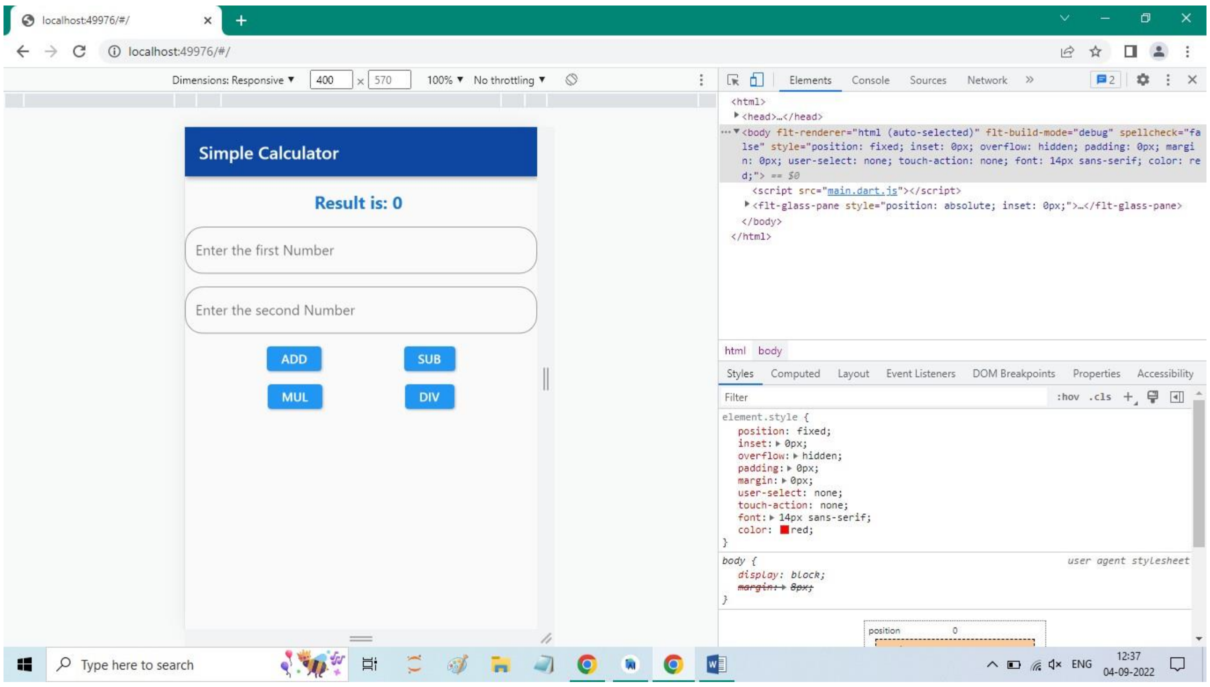
Designing the mobile app to implement different widgets.

Code:

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

void main() {
  runApp(MaterialApp(
    debugShowCheckedModeBanner: false,
    home: MyApp(),
  ));
}

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

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

class _MyAppState extends State<MyApp> {
  TextEditingController controller1 = TextEditingController();
  TextEditingController controller2 = TextEditingController();
  int? num1 = 0,
      num2 = 0,
      result = 0;

  add() {
    setState(() {
      num1 = int.parse(controller1.text);
      num2 = int.parse(controller2.text);
      result = num1! + num2!;
    });
  }

  sub() {
    setState(() {
      num1 = int.parse(controller1.text);
      num2 = int.parse(controller2.text);
      result = num1! - num2!;
    });
  }
```

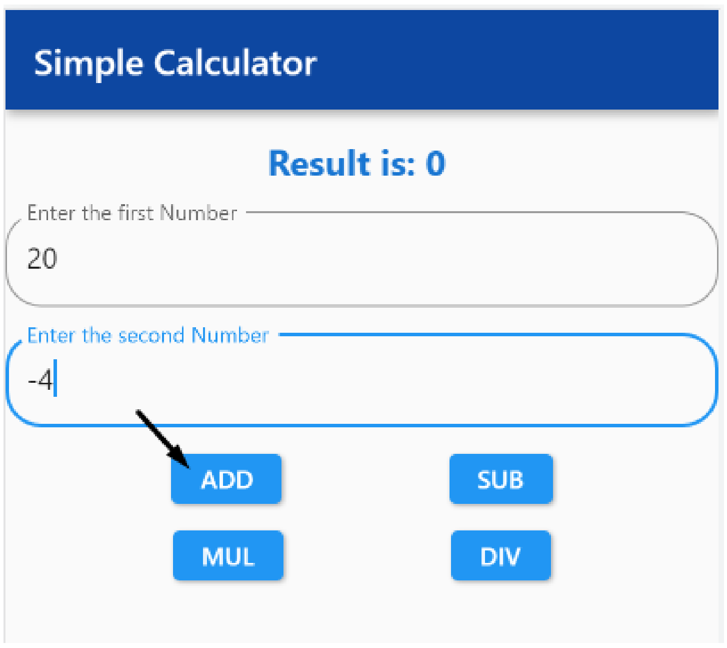
```
mul() {
  setState(() {
    num1 = int.parse(controller1.text);
    num2 = int.parse(controller2.text);
    result = num1! * num2!;
  });
}

div() {
  setState(() {
    num1 = int.parse(controller1.text);
    num2 = int.parse(controller2.text);
    result = num1! ~/ num2!;
  });
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Simple Calculator'),
      backgroundColor: Colors.blue.shade900,
    ),
    body: Column(
      children: [
        SizedBox(
          height: 15,
        ),
        Text('Result is: $result', style: TextStyle(fontSize: 20,
            color: Colors.blue.shade700
        ),),
        SizedBox(
          height: 15,
        ),
        TextField(
          controller: controller1,
          decoration: InputDecoration(
              labelText: "Enter number", border: OutlineInputBorder(
              borderRadius: BorderRadius.circular(20)
          )
          ),
        ),
        SizedBox(
          height: 15,
        ),
        TextField(
          controller: controller2,
          decoration: InputDecoration(
            labelText: "Enter number", border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(20)
          )
          ),
        ),
        SizedBox(
          height: 15,
        ),
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            ElevatedButton(onPressed: () {
              add();
              controller1.clear();
              controller2.clear();
            }, child: Text('ADD')),
            ElevatedButton(onPressed: () {
              sub();
            }, child: Text('SUB'))
          ],
        ),
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            ElevatedButton(onPressed: () {
              mul();
            }, child: Text('MUL')),
            ElevatedButton(onPressed: () {
              div();
            }, child: Text('DIV')),
          ],
        )
      ],
    ),
  );
 }
}
```

Output:



```
<html>
 ▸<head>…</head>
 ▾<body flt-renderer="html (auto-selected)" flt-build-mode="debug" spellcheck="false" style="position: fixed; inset: 0px; overflow: hidden; padding: 0px; margin: 0px; user-select: none; touch-action: none; font: 14px sans-serif; color: red;"> == $0
    <script src="main.dart.js"></script>
   ▸<flt-glass-pane style="position: absolute; inset: 0px;">…</flt-glass-pane>
  </body>
</html>
```

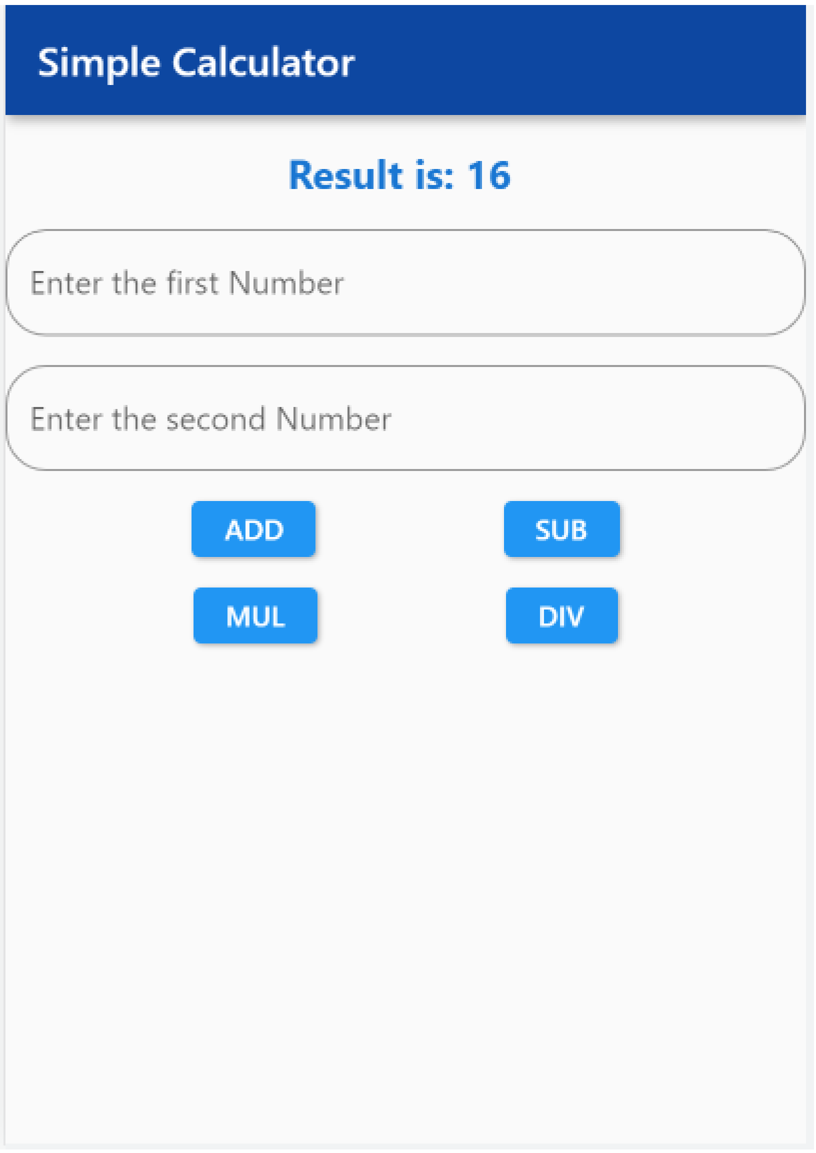
```
element.style {
   position: fixed;
   inset: ▸ 0px;
   overflow: ▸ hidden;
   padding: ▸ 0px;
   margin: ▸ 0px;
   user-select: none;
   touch-action: none;
   font: ▸ 14px sans-serif;
   color: 🟥red;
}
body {                                user agent stylesheet
   display: block;
   ~~margin: ▸ 8px;~~
}
```


Simple Calculator

Result is: 0

Enter the first Number: 20

Enter the second Number: -4


Simple Calculator

Result is: 16

Enter the first Number

Enter the second Number

Practical No. 3

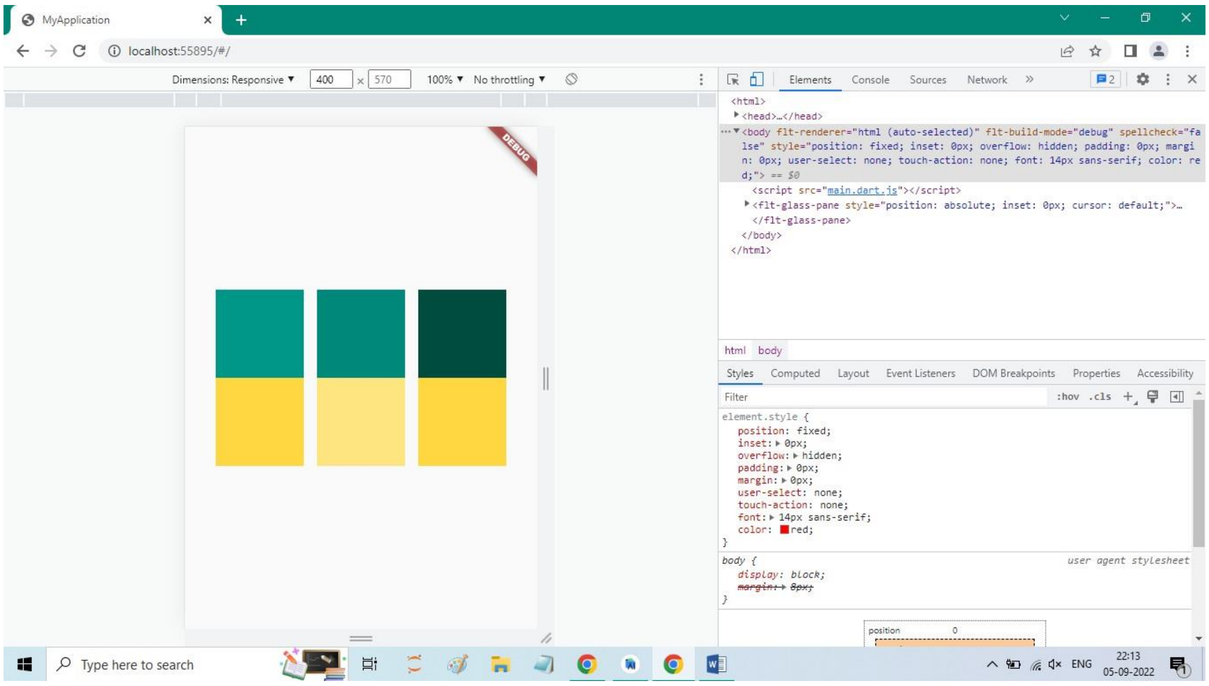
Designing the mobile app to implement different Layouts.

Date: **Sign:**

Code:

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

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

class Demoapp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
        title: 'My Application',
        debugShowCheckedModeBanner: true,
        home: Scaffold(
            body: Padding(
                padding: const EdgeInsets.all(20.0),
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
                    Row(
                      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                      children: [
                        Container(height: 100, width: 100, color: Colors.teal,),
                        Container(
                            height: 100, width: 100, color: Colors.teal[600]),
                        Container(
                            height: 100, width: 100, color: Colors.teal[900]),
                      ],
                    ),
                    Row(
                      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                      children: [
                        Container(
                          height: 100, width: 100, color: Colors.amberAccent,),
                        Container(height: 100,
                            width: 100,
                  color: Colors.amberAccent[100]),
                Container(height: 100,
                  width: 100,
                  color: Colors.amberAccent[200]),
              ],
            )
          ],
        )
      )

    )
  );
 }
}
```

Output:



```
<html>
 <head>…</head>
 <body flt-renderer="html (auto-selected)" flt-build-mode="debug" spellcheck="false" style="position: fixed; inset: 0px; overflow: hidden; padding: 0px; margin: 0px; user-select: none; touch-action: none; font: 14px sans-serif; color: red;"> == $0
   <script src="main.dart.js"></script>
  <flt-glass-pane style="position: absolute; inset: 0px; cursor: default;">…
   </flt-glass-pane>
 </body>
</html>
```

```
element.style {
   position: fixed;
   inset: 0px;
   overflow: hidden;
   padding: 0px;
   margin: 0px;
   user-select: none;
   touch-action: none;
   font: 14px sans-serif;
   color: red;
}
body {                                   user agent stylesheet
   display: block;
   ~~margin: 8px;~~
}
```

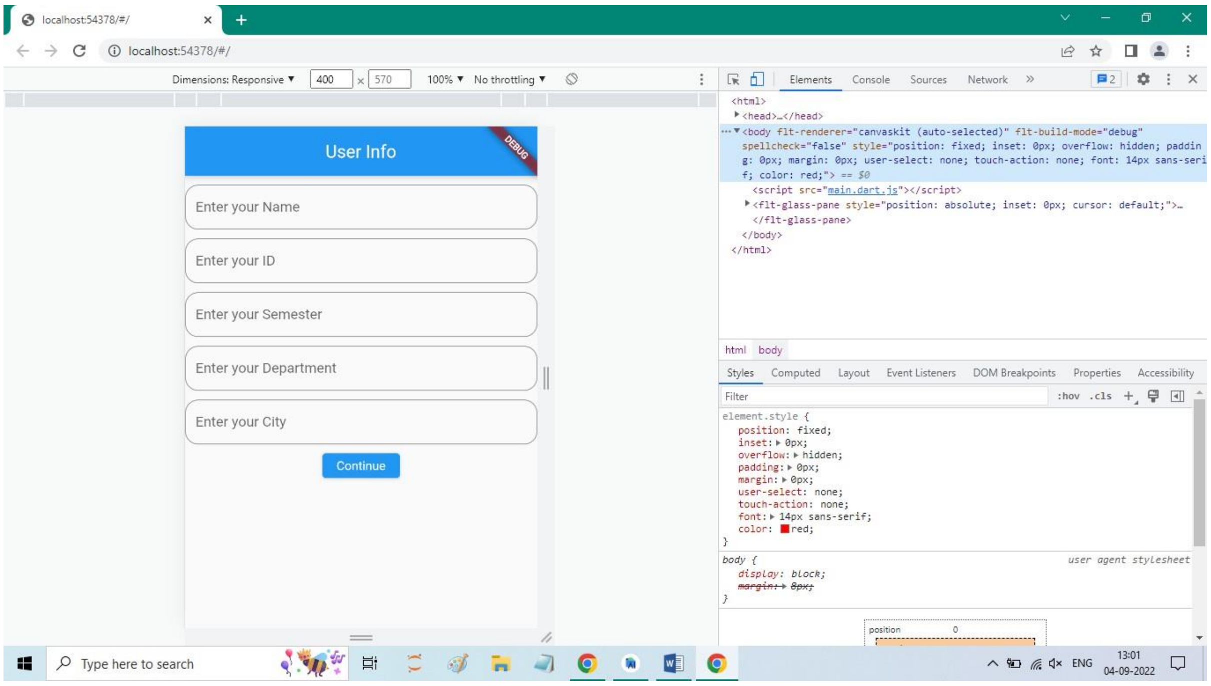
Practical No. 4

Designing the mobile app to implement the routing.

Code:

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

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

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

   @override
   Widget build(BuildContext context) {
     TextEditingController name = TextEditingController();
     TextEditingController id = TextEditingController();
     TextEditingController semester = TextEditingController();
     TextEditingController dept = TextEditingController();
     TextEditingController city = TextEditingController();
     return Scaffold(
        appBar: AppBar(
            title: Text("User Info"),
            centerTitle: true,
            ), // AppBar
        body: Column(
           children: [
           SizedBox(height: 10),
           TextField(
                  cotroller: name,
                  decoration: InputDecoration(
                     labelText: " Enter your name",
                     border: OutLineInputBorder(
                            borderRadius: BorderRadius.circular(15)
                            ) // OutlineInputBorder
                  ), //Input Decoration
        ), // TextField

        SizedBox(height: 10),
           TextField(
            cotroller: id,
            decoration: InputDecoration(
              labelText: " Enter your ID",
              border: OutLineInputBorder(
                   borderRadius: BorderRadius.circular(15)
                   ) // OutlineInputBorder
            ), //Input Decoration
     ), // TextField

     SizedBox(height: 10),
       TextField(
            cotroller: semester,
            decoration: InputDecoration(
              labelText: " Enter your Semester",
              border: OutLineInputBorder(
                   borderRadius: BorderRadius.circular(15)
                   ) // OutlineInputBorder
            ), //Input Decoration
     ), // TextField

     SizedBox(height: 10),
       TextField(
            cotroller: dept,
            decoration: InputDecoration(
              labelText: " Enter your Department",
              border: OutLineInputBorder(
                   borderRadius: BorderRadius.circular(15)
                   ) // OutlineInputBorder
            ), //Input Decoration
     ), // TextField

     SizedBox(height: 10),
       TextField(
            cotroller: city,
            decoration: InputDecoration(
              labelText: " Enter your City",
              border: OutLineInputBorder(
                   borderRadius: BorderRadius.circular(15)
                   ) // OutlineInputBorder
            ), //Input Decoration
     ), // TextField

     SizedBox(height: 10,),
     ElevatedButton(onPressed: () {
          Navigator.push(context, MaterialPageRoute(builder: (context)=>NextScreen(
                name: name.text,
                id: id.text,
                semester: semester.text,
                dept: dept.text,
                city: city.text,
                ))).when.Complete(() => { //NextScreen, MaterialPageRoute
                name:clear(),
                id.clear(),
                semester.clear(),
                dept.clear(),
                city.clear()
                });
       }, child: Text("continue")) //ElevatedButton
     ],
   ), //Column
  ); // Scaffold
 }
}

class NextScreen extends StatelessWidget {
   String? name, id, semester, dept, city;
   NextScreen({
       this.name, this.id, this.semester, this.dept, this.city
  });

 @override
 Widget build(BuildContext context) {
   return Scaffold(
       body: Column(
         children: [
                Text("Name:" +name.toString()),
                Text("Id:" +id.toString()),
                Text("Semester:" +semester.toString()),
                Text("Department:" +name.toString()),
                Text("City:" +city.toString()),
       ],
      ), //Column
    ); // Scaffold
  }
}
```

Output:



```
<html>
 ▸<head>…</head>
…▾<body flt-renderer="canvaskit (auto-selected)" flt-build-mode="debug"
  spellcheck="false" style="position: fixed; inset: 0px; overflow: hidden; paddin
  g: 0px; margin: 0px; user-select: none; touch-action: none; font: 14px sans-seri
  f; color: red;"> == $0
    <script src="main.dart.js"></script>
   ▸<flt-glass-pane style="position: absolute; inset: 0px; cursor: default;">…
    </flt-glass-pane>
  </body>
</html>
```

```
element.style {
   position: fixed;
   inset: ▸ 0px;
   overflow: ▸ hidden;
   padding: ▸ 0px;
   margin: ▸ 0px;
   user-select: none;
   touch-action: none;
   font: ▸ 14px sans-serif;
   color: 🟥red;
}
body {                                          user agent stylesheet
   display: block;
   ~~margin: ▸ 8px;~~
}
```

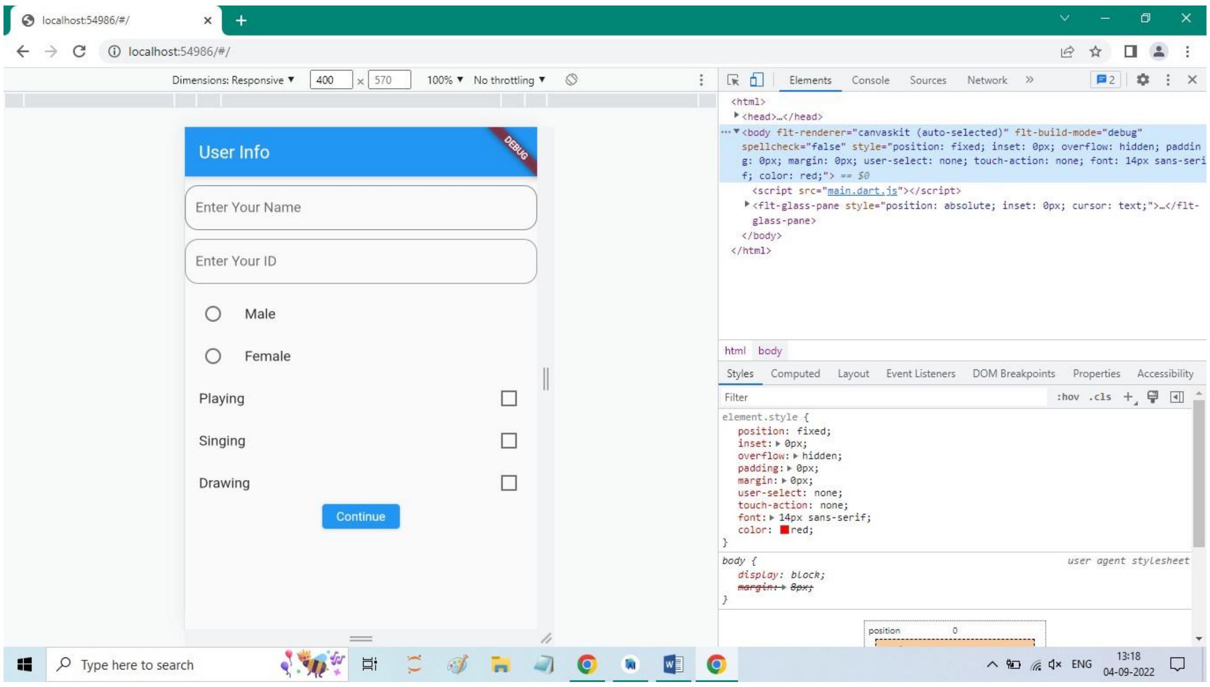
Practical No. 5

Designing the mobile app to implement the state management.

Code:

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

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

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

  @override
  State<HomeScreen> createState() => _HomeScreenState();
 }

class _HomeScreenState extends State<HomeScreen> {
  TextEditingController name = TextEditingController();
  TextEditingController id = TextEditingControler();
  String genderValue = "";
  bool hobby1 = false;
  bool hobby2 = false;
  bool hobby3 = false;
  String strhobby1 = "",
  String strhobby2 = "",
  String strhobby3 = "",
  final formKey = GlobalKey<FormState>(); // formValidation

  @overrride
  Widget build(BuildContext context) {
      return Scaffold(
        appBar: AppBar(
              title: Text("User Info"),
              ),
        body: Form(
              key: formKey,
              child: Column(
                children: [
                  SizedBox(height: 10,),
                TextFormField(
                      controller: name,
                      validator: (value){
                         if(value!.isEmpty) {
                                return 'Please Enter Your Name';
                           }
                         return null;
                      },
                      decoration: InputDecoration(
                                labelText: "Enter Your Name",
                                border: OutlineInputBorder(
                                   borderRadius: BorderRadius.circular(15)
                                ) //OutlineInputBorder
                      ), //InputDecoration
            ), // TextFormField
            SizedBox(height: 10,),
            TextFormField(
              Controller: id,
              validator: (value) {
                      if(value!.isEmpty) {
                         return 'Please Enter your ID';
                      }
                      return null;
               },
              decoration: InputDecoration(
                   labelText: "Enter your ID",
                   border: OutLineInputBorder(
                        borderRadius: BorderRadius.circular(15)
                      ) //OutLineInputBorder
                  ), //InputDecoration
            ), //TextFormField
            SizedBox(height: 10,),
            RadioListTile(value: 'Male',groupValue: genderValue, onChanged: (val)
 {
            setState(() {
               genderValue = val.toString();
            });
     },
            title: Text("Male"),), // RadioListTitle
            RadioListTile(value: 'Female',groupValue: genderValue, onChanged:
 (val) {
            setState(() {
               genderValue = val.toString();
            });
     },
                title: Text("Female"),), // RadioListTitle

                CheckboxListTile(value: hobby1, onChanged: (value) {
                  setState(() {
                        hobby1 = !hobby1;
                        if(hobby1) {
                          strhobby1 = 'Playing';
                        }
                    });
                },
                title: Text("Playing"),), //checkboxListTile
                CheckboxListTile(value: hobby2, onChanged: (value) {
                  setState(() {
                        hobby2 = !hobby2;
                        if(hobby2) {
                          strhobby2 = 'Singing';
                        }
                    });
                },
                title: Text("Singing"),), //checkboxListTile
                CheckboxListTile(value: hobby3, onChanged: (value) {
                  setState(() {
                        hobby3 = !hobby3;
                        if(hobby3) {
                          strhobby3 = 'Drawing';
                        }
                    });
                },
                title: Text("Drawing"),), //checkboxListTile

                ElevatedButton(onPressed: () {
                  if(formKey.currentState!.validate()) {
                        if(genderValue !="") {
                          Navigator.push(context, MaterialPageRoute(builder: (context) =>
  NextScreen (
                                name: name.text,
                                id:id.text,
                                gender:genderValue,
                                hobbies: '${strhobby1.toString()}
                                       ${strhobby2.toString()}
                                       ${strhobby3.toString()},
                                ))); // NextScreen //MaterialPageRoute
                          }
                        }
                        }, child: Text("Contiue"))// ElevatedButton
                        ],
                    ), //Column
                ), //Form
            ); //Scaffold
        }
    }

class NextScreen extends StatelessWidget {
   String ? name,id,gender,hobbies;
   NextScreen ({
        this.name, this.id, this.gender,this.hobbies
        });
  @override
  Widget build(BuildContext context) {
        return Scaffold(
          body: Column(
                children: [
                        Text("Name: " +name.toString()),
                        Text("Id: " +id.toString()),
                        Text("Gender: " +gender.toString()),
                        Text("Hobbies: " +hobbies.toString()),
                ],
          ), // Column
        ); //Scaffold
    }
}
```

Output:


localhost:54986/#/
localhost:54986/#/
Dimensions: Responsive ▾ 400 x 570 100% ▾ No throttling ▾
Elements Console Sources Network
DEBUG
Type here to search
ENG 13:18 04-09-2022

User Info

Enter Your Name

Enter Your ID

○ Male

○ Female

Playing ☐

Singing ☐

Drawing ☐

Continue

```
<html>
 ▸<head>…</head>
…▾<body flt-renderer="canvaskit (auto-selected)" flt-build-mode="debug"
  spellcheck="false" style="position: fixed; inset: 0px; overflow: hidden; paddin
  g: 0px; margin: 0px; user-select: none; touch-action: none; font: 14px sans-seri
  f; color: red;"> == $0
    <script src="main.dart.js"></script>
   ▸<flt-glass-pane style="position: absolute; inset: 0px; cursor: text;">…</flt-
    glass-pane>
  </body>
</html>
```

html body

Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility

```
element.style {
   position: fixed;
   inset: ▸ 0px;
   overflow: ▸ hidden;
   padding: ▸ 0px;
   margin: ▸ 0px;
   user-select: none;
   touch-action: none;
   font: ▸ 14px sans-serif;
   color: 🟥red;
}
body {                                        user agent stylesheet
   display: block;
   ~~margin: ▸ 8px;~~
}
```

position 0

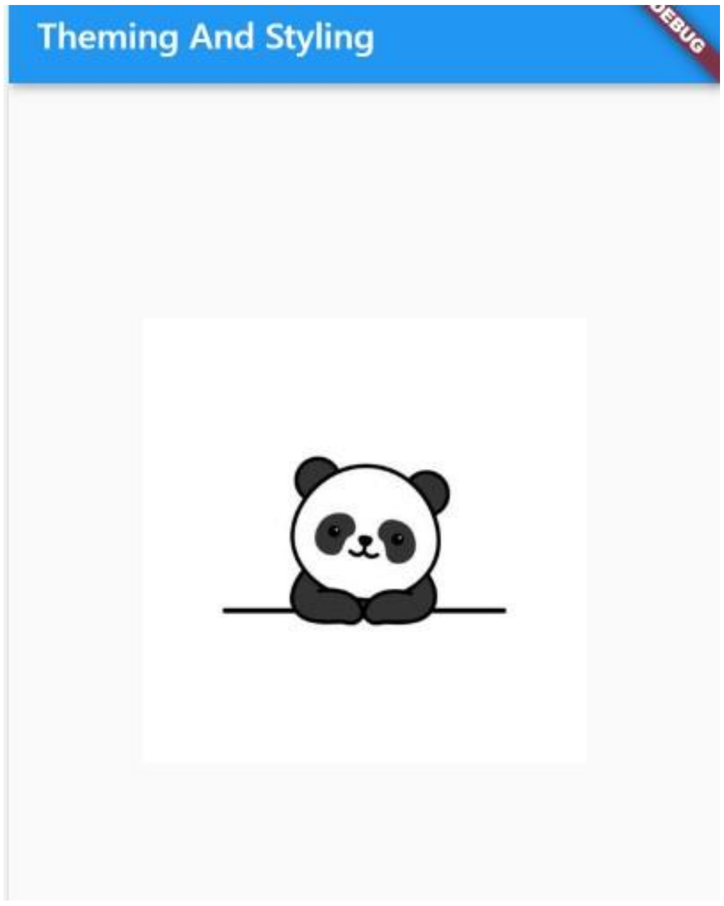
Practical No.6

Designing the mobile app to implement the theming and styling.

Code:

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

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

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

  @override
  Widget build(BuildContext cotext) {
   return Scaffold(
     appBar: AppBar(
           title: Text('Theming and Styling'),
           ), //AppBar
     body: Center(
           child: Column(
                   mainAxisAlignment: MainAxisAlignment.saceEvenly,
                   children: [
                           Image.network('copy-any-google-image-link'
height:250,width: 250,)
                           ],
                   ), //Column
           ) // Center
   );
  }
}
```

Output:



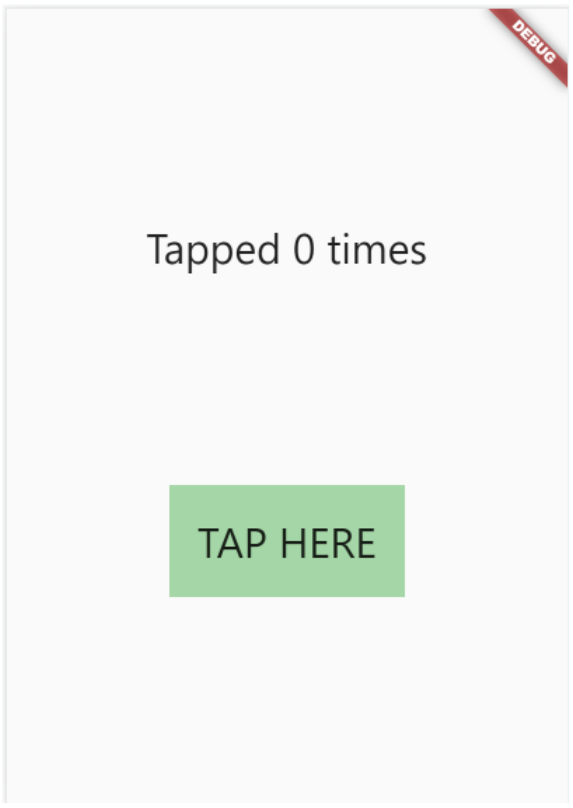
Practical No.7

Designing the mobile app to implement Gestures.

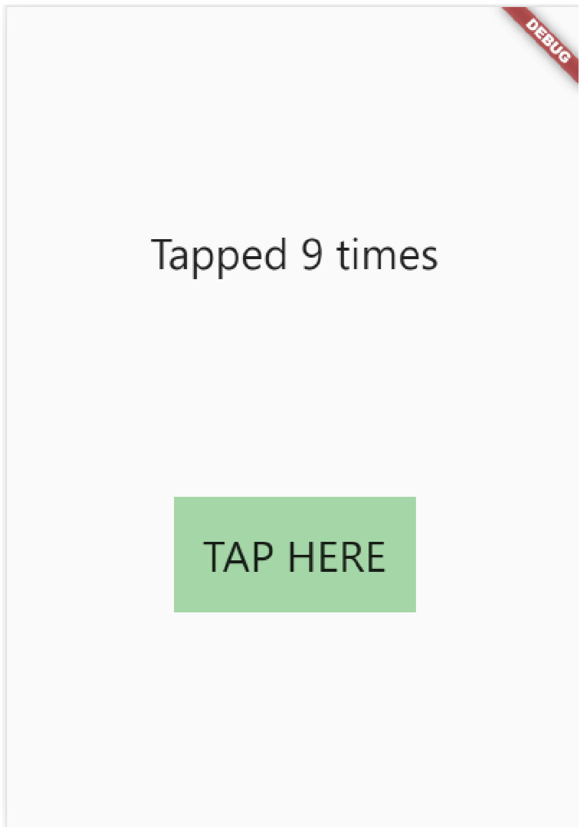
Code:

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

void main() {
  runApp(MaterialApp(home: MyApp()));
}

class MyApp extends StatefulWidget {
  const Mypp({Key? key}) : super(key: key);
  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  int numberOfTimesTapped = 0;
  @override
  Widget build(BuildContext context) {
 return Scaffold(
   body: Center(
          child: Column(
                 mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                 children: [
                   Text('Tapped '+ numberOfTimesTapped.toString() + 'times', style:
TextStyle                              (fontSize:30)),
                   GestureDetector(
                          onTap:() {
                          setState((){
                          numberOfTimesTapped++;
                          });
                              },
                   child: Container(
                          padding: EdgeInsets.all(20),
                          color: Colors.green[200],
                              child: Text('TAP HERE', style:TextStyle(fontSize: 30), )),
//Container
                    )//GestureDetector
                ],
            ), // Column
        ), // Center
 ); //Scaffold
   }
}
```

Output:



```
Tapped 0 times
```


DEBUG
Tapped 9 times
TAP HERE

Practical No.8

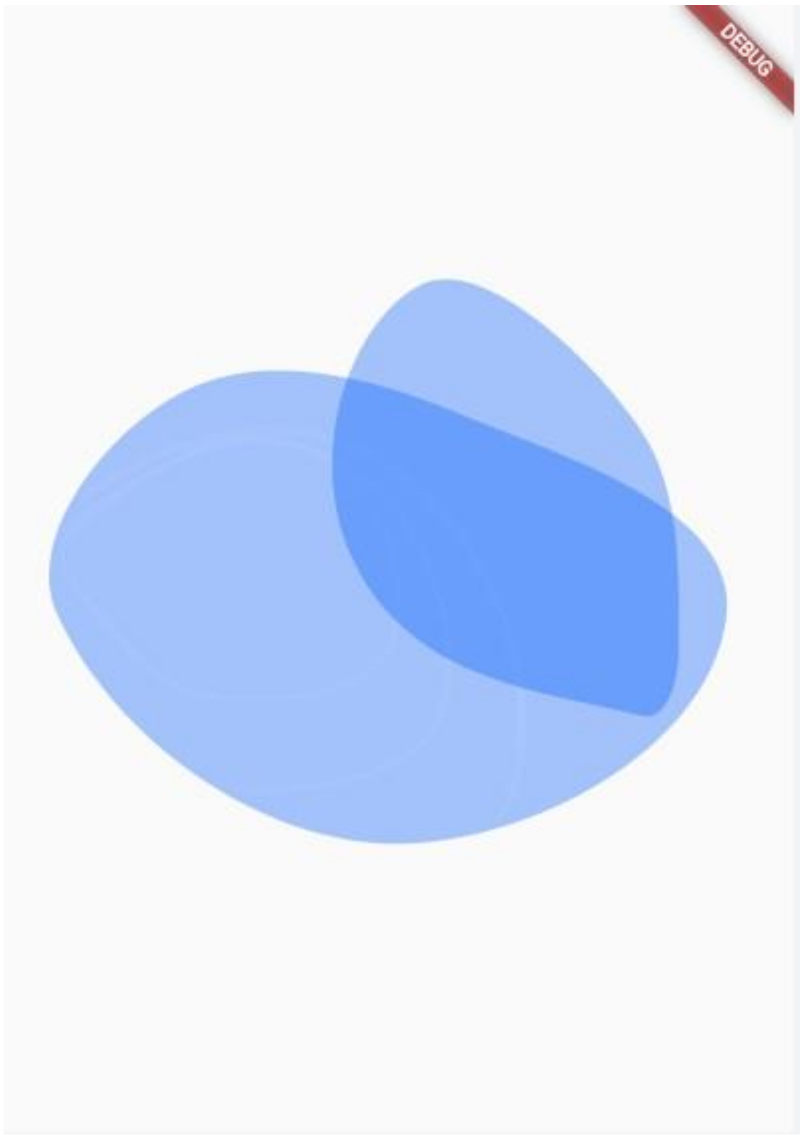
Designing the mobile app to implement the Animation

Code:

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

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

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

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

class _MyAppState extends State<MyApp> with SingleTickerProvideStateMixin
{
//controller
  late final AnimationController _controller;
  @override
  void initState() {
 super.initState();
 _controller = AnimationController(duration:Duration(secods: 10), vsync: this);
  }

  @override
  void dispose() {
 super.dispose();
 _controller.dispose();
 }
```

```
  bool bookmark = false;
  Widget build(BuildContext context) {
 return Scaffold(
   body: Center(
          child: GestureDetector(
                 onTap: () {
                   if(bookmark == false)
                    {
                         bookmark = true;
                         _controller.forward();
                     }
                   else
                   {
                         bookmark = false;
                         _controller.reverse();
                    }
                 },
          child:
Lottie.network('https://assets9.lottieflies.com/packages/lf20_3le10jj4.json',
          controller: _controller
          )), // GestureDetector
 ), //Center
   ); //Scaffold
  }
}
```

Output:

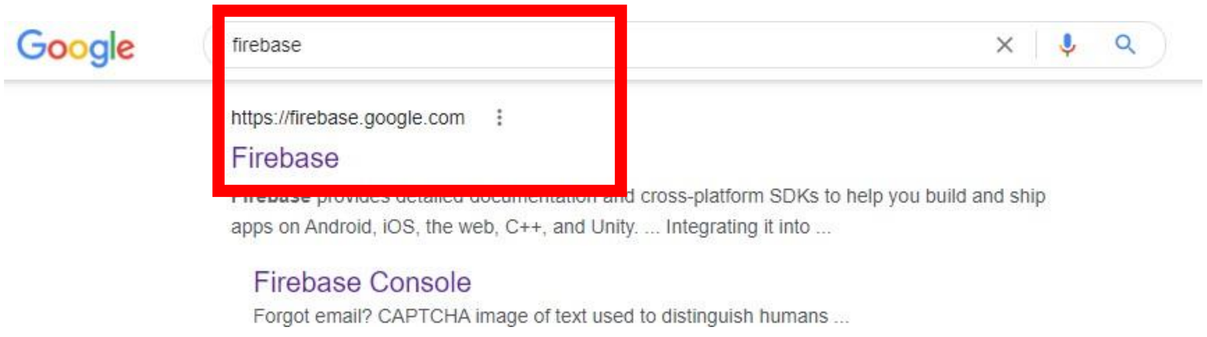

DEBUG

Practical No.9

Designing the mobile app working with Firebase Date: Sign:

---

1. Search on Google " Firebase ". Click on the website https://firebase.google.com/



firebase

https://firebase.google.com

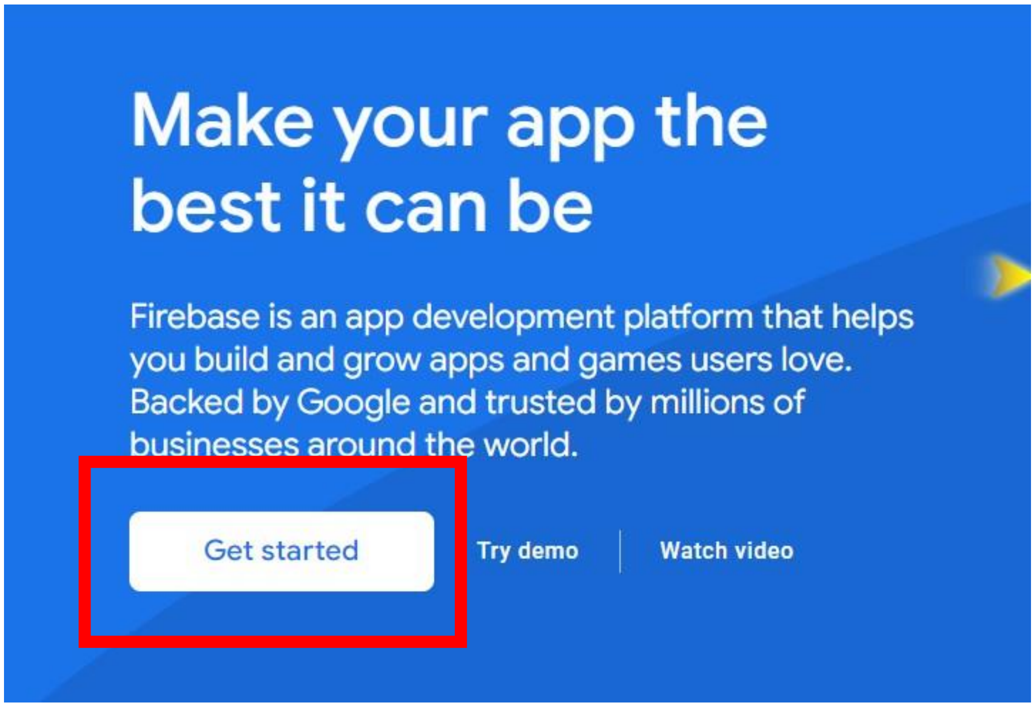
Firebase

Firebase provides detailed documentation and cross-platform SDKs to help you build and ship apps on Android, iOS, the web, C++, and Unity. ... Integrating it into ...

Firebase Console

Forgot email? CAPTCHA image of text used to distinguish humans ...

2. Click on Get Started button.

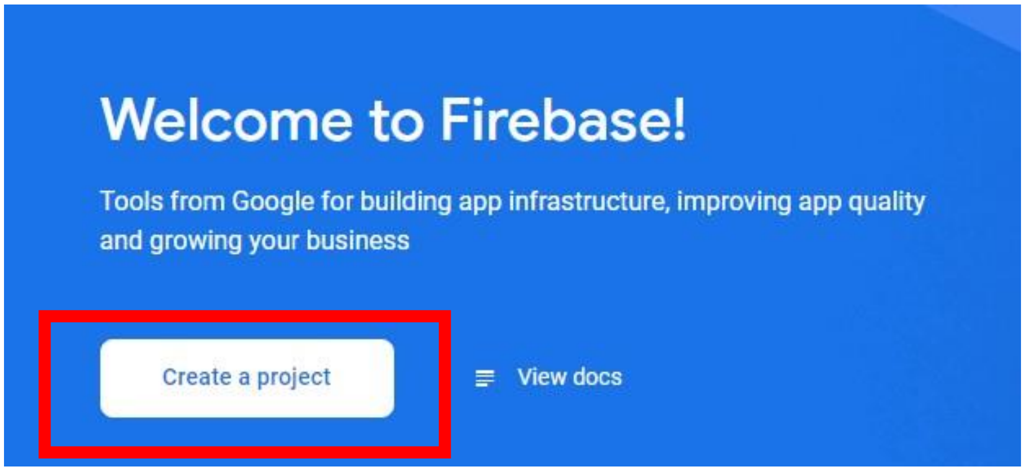


Make your app the best it can be

Firebase is an app development platform that helps you build and grow apps and games users love. Backed by Google and trusted by millions of businesses around the world.

3. Login through your email id.

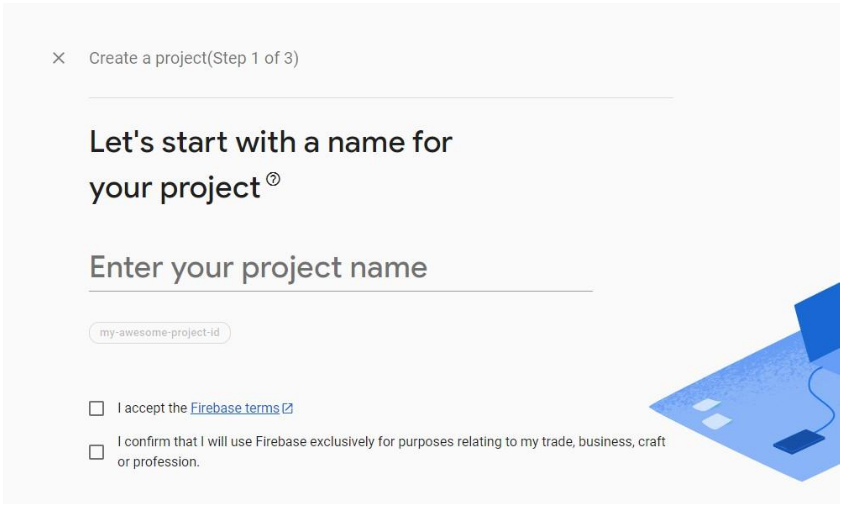
4. After login, click on Create a project button.



Welcome to Firebase!

Tools from Google for building app infrastructure, improving app quality and growing your business

5. Enter your project name and enable the two checkboxes and then and click on Continue button.



Create a project(Step 1 of 3)

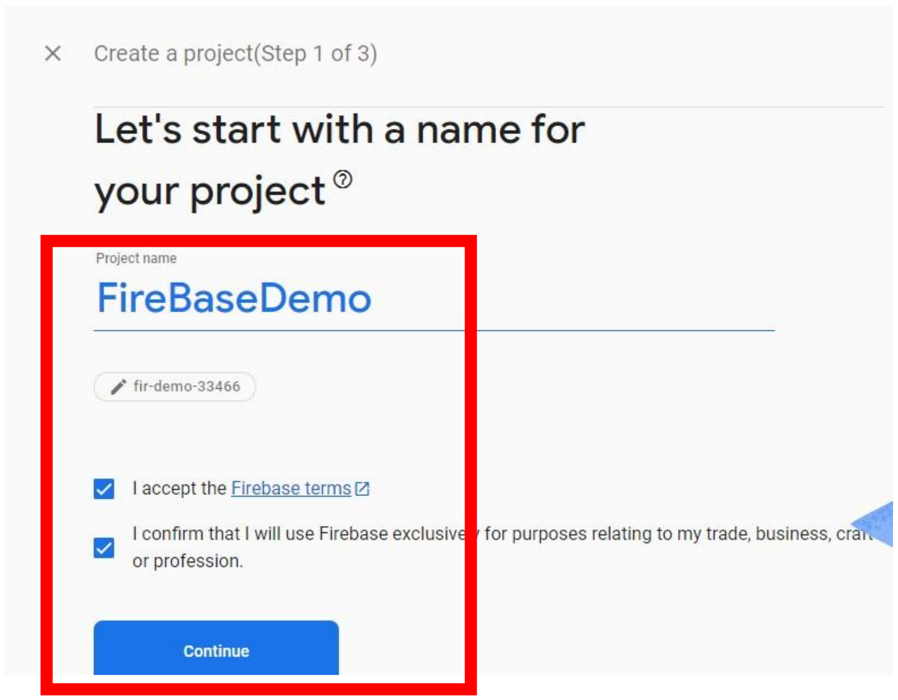
Let's start with a name for your project

Enter your project name

my-awesome-project-id

I accept the Firebase terms

I confirm that I will use Firebase exclusively for purposes relating to my trade, business, craft or profession.



Create a project(Step 1 of 3)

Let's start with a name for your project

Project name

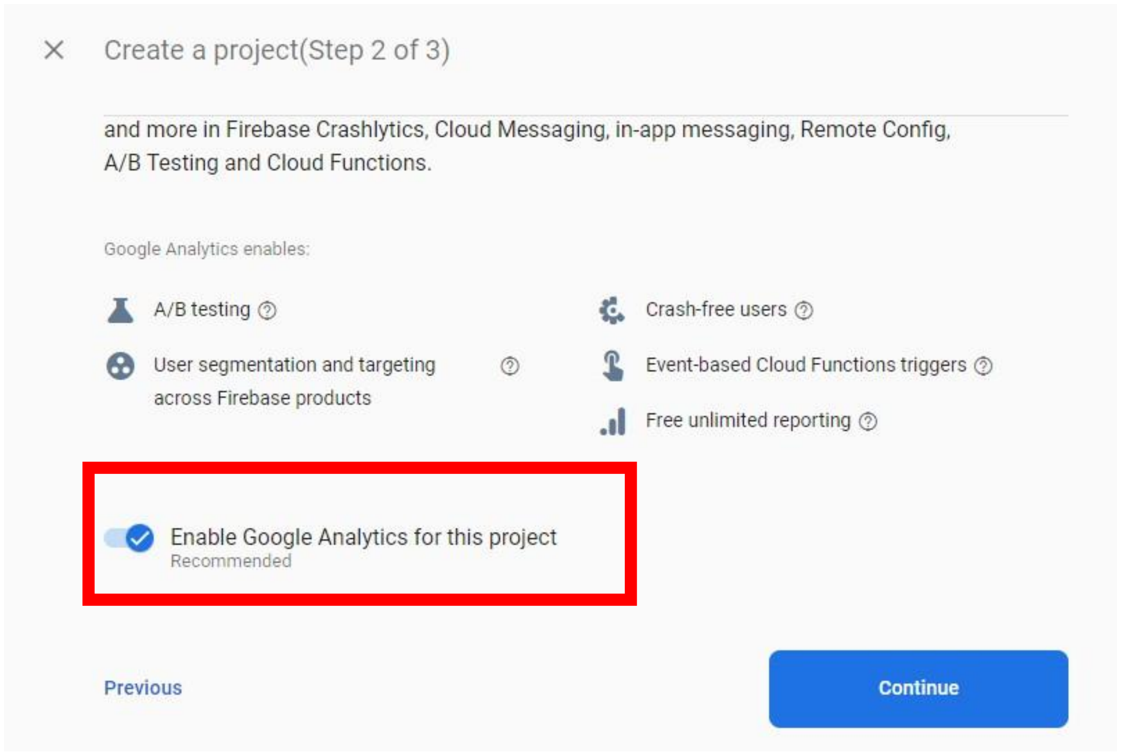
FireBaseDemo

fir-demo-33466

✓ I accept the Firebase terms

✓ I confirm that I will use Firebase exclusively for purposes relating to my trade, business, craft or profession.

6. Enable Google Analytics for this project and then click on Continue button.



Create a project(Step 2 of 3)

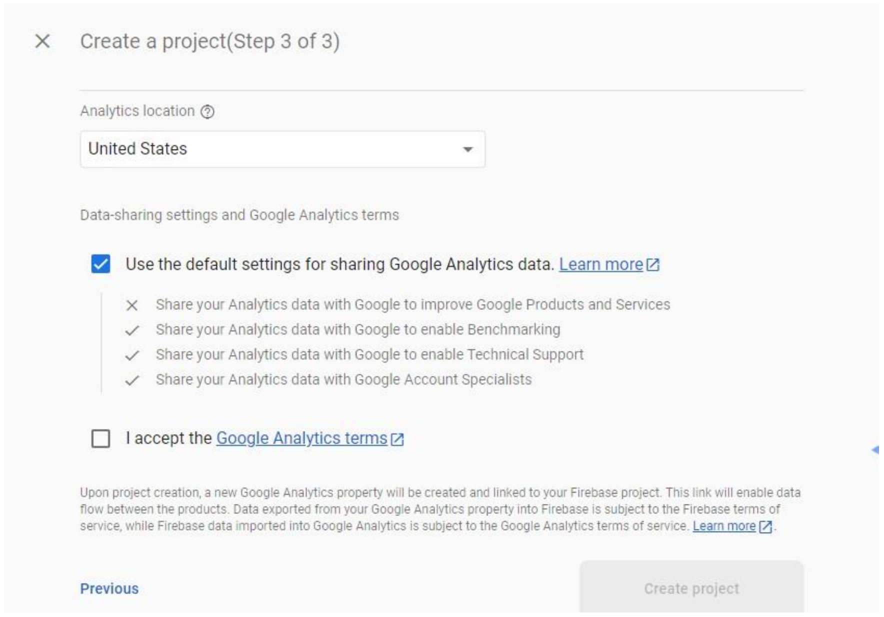
and more in Firebase Crashlytics, Cloud Messaging, in-app messaging, Remote Config, A/B Testing and Cloud Functions.

Google Analytics enables:

- A/B testing
- User segmentation and targeting across Firebase products
- Crash-free users
- Event-based Cloud Functions triggers
- Free unlimited reporting

✓ Enable Google Analytics for this project
Recommended

7. Select a Analytics location of your choice and enable(tick) the two checkboxes then click on Create Project button.



Create a project(Step 3 of 3)

Analytics location

United States

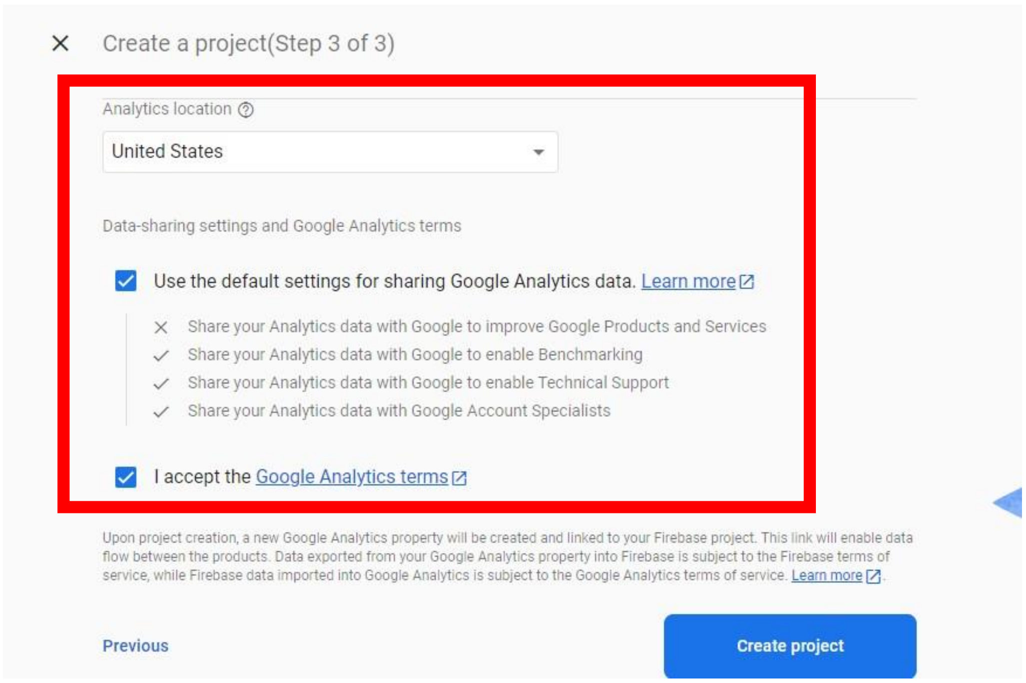
Data-sharing settings and Google Analytics terms

☑ Use the default settings for sharing Google Analytics data. Learn more

- × Share your Analytics data with Google to improve Google Products and Services
- ✓ Share your Analytics data with Google to enable Benchmarking
- ✓ Share your Analytics data with Google to enable Technical Support
- ✓ Share your Analytics data with Google Account Specialists

☐ I accept the Google Analytics terms

Upon project creation, a new Google Analytics property will be created and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. Learn more.



Create a project(Step 3 of 3)

Analytics location

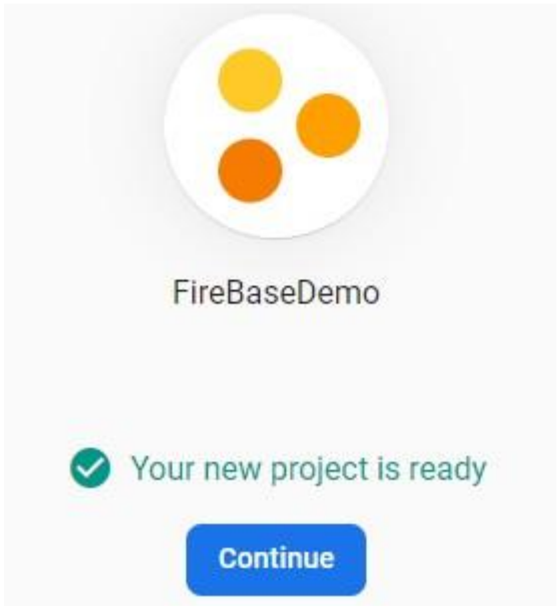
United States

Data-sharing settings and Google Analytics terms

- [✓] Use the default settings for sharing Google Analytics data. Learn more
  - × Share your Analytics data with Google to improve Google Products and Services
  - ✓ Share your Analytics data with Google to enable Benchmarking
  - ✓ Share your Analytics data with Google to enable Technical Support
  - ✓ Share your Analytics data with Google Account Specialists
- [✓] I accept the Google Analytics terms

Upon project creation, a new Google Analytics property will be created and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. Learn more.

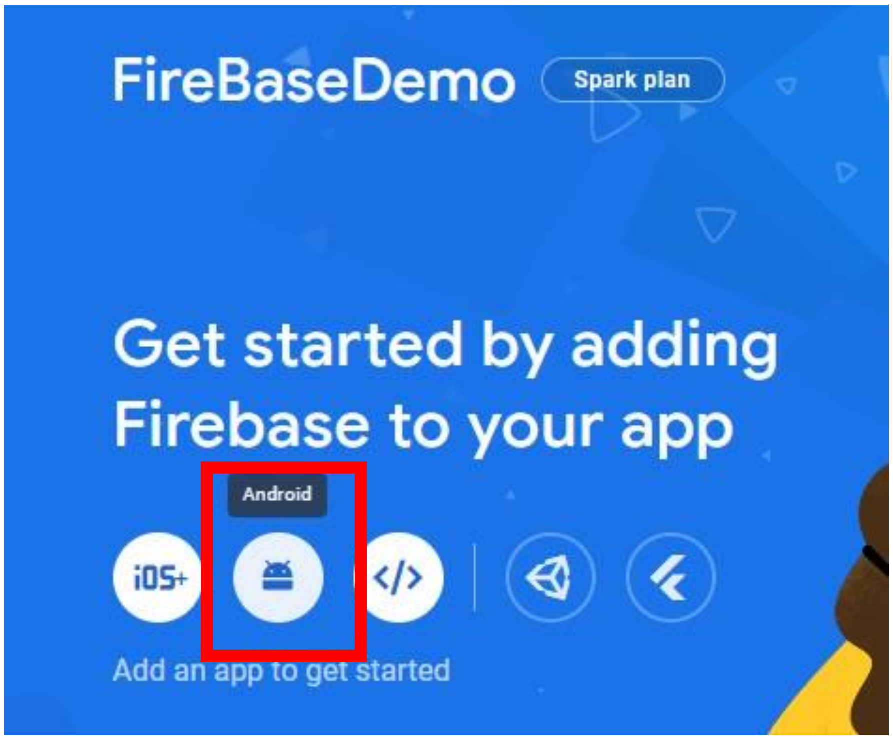
8. Project will now be created. Click on Continue.



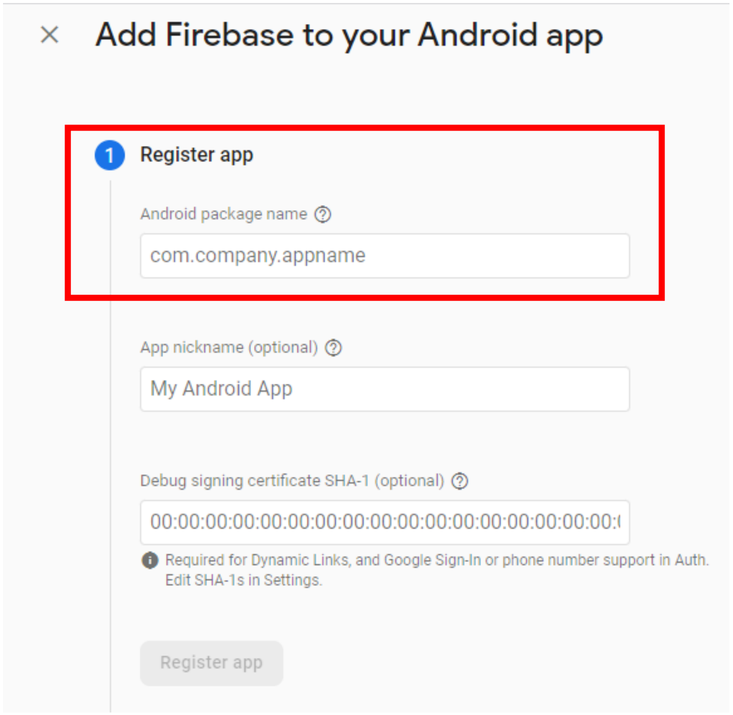
FireBaseDemo

Your new project is ready

9. Click on Android button as you are making an Android based application.


FireBaseDemo
Spark plan
Get started by adding Firebase to your app
Android
iOS+
Add an app to get started

10. Enter Android Package Name.



Add Firebase to your Android app

1 Register app

Android package name: com.company.appname

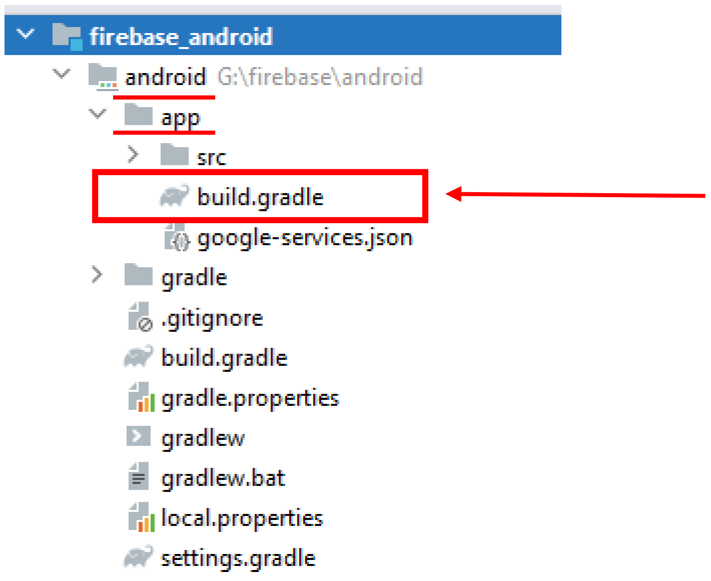
App nickname (optional): My Android App

Debug signing certificate SHA-1 (optional): 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:0

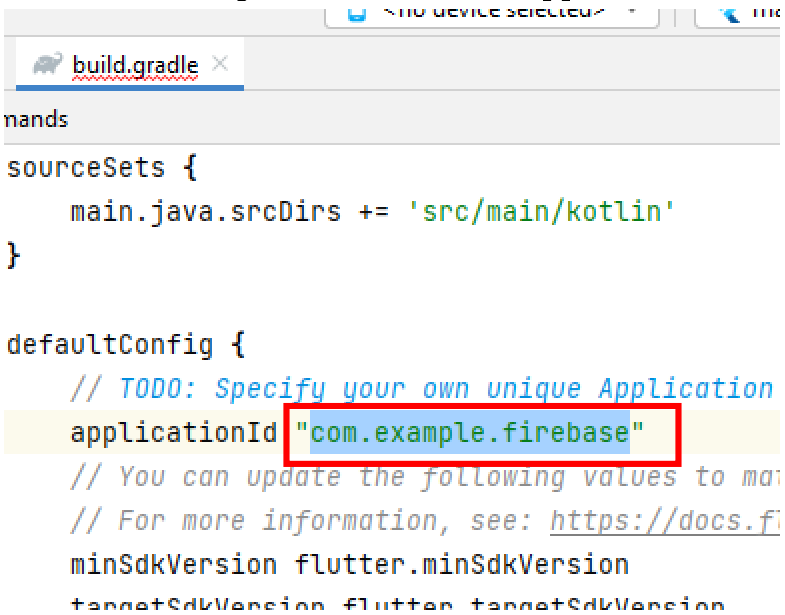
Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

Follow the below steps to find your Android Package Name:

- Open your Android Studio project.
- Open the app folder which is in Android folder.
- Open the build.gradle file in Android Studio. (Double click on build.gradle file to open it)

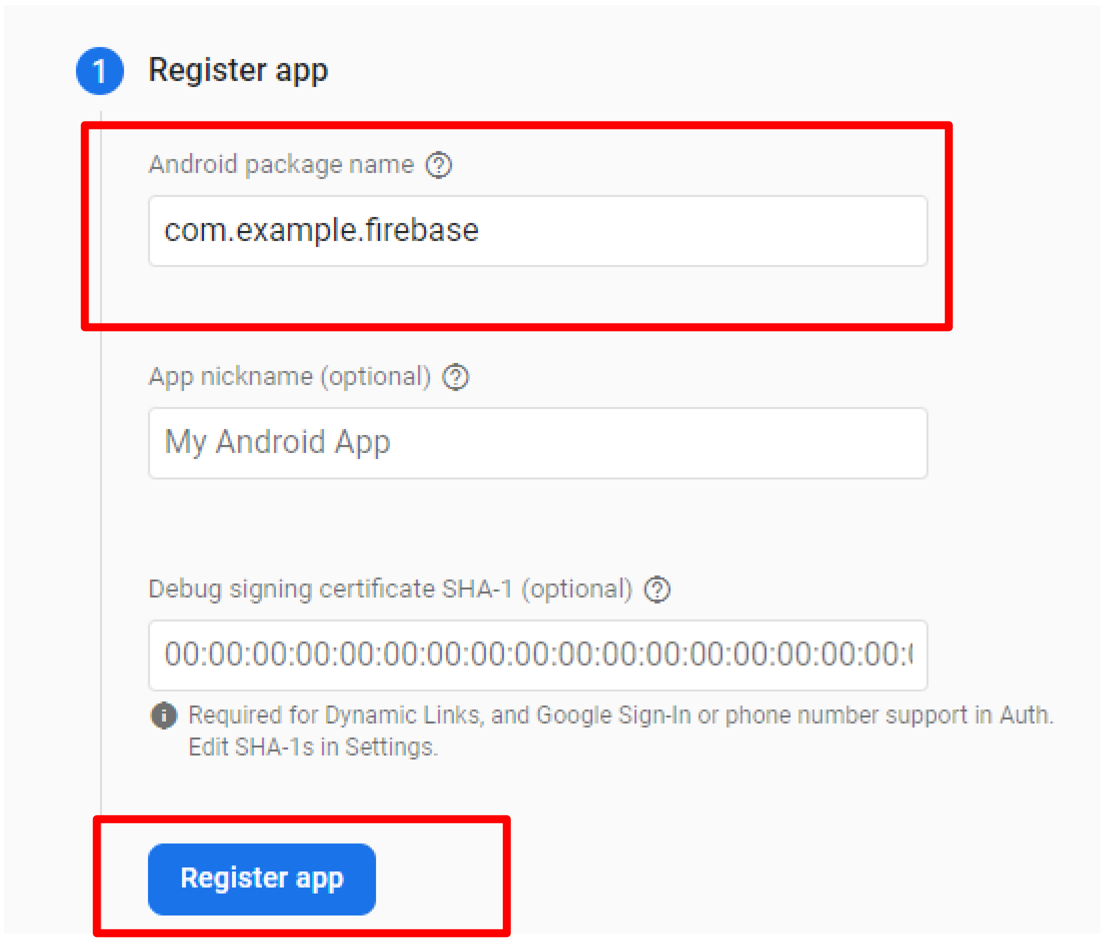


- In build.gradle file , search for the defaultConfig section.
- In defaultConfig section check the applicationID.



```
sourceSets {
    main.java.srcDirs += 'src/main/kotlin'
}

defaultConfig {
    // TODO: Specify your own unique Application
    applicationId "com.example.firebase"
    // You can update the following values to ma
    // For more information, see: https://docs.f
    minSdkVersion flutter.minSdkVersion
    targetSdkVersion flutter.targetSdkVersion
```

- The applicationID is your Android Package Name. (Eg : Here , the Android Package name is : com.example.firebase)
- Copy it without quotes and paste in android package name field.
- Only mention Android Package Name. Other is optional.
- Click on Register App button.


1 Register app

Android package name

com.example.firebase

App nickname (optional)

My Android App

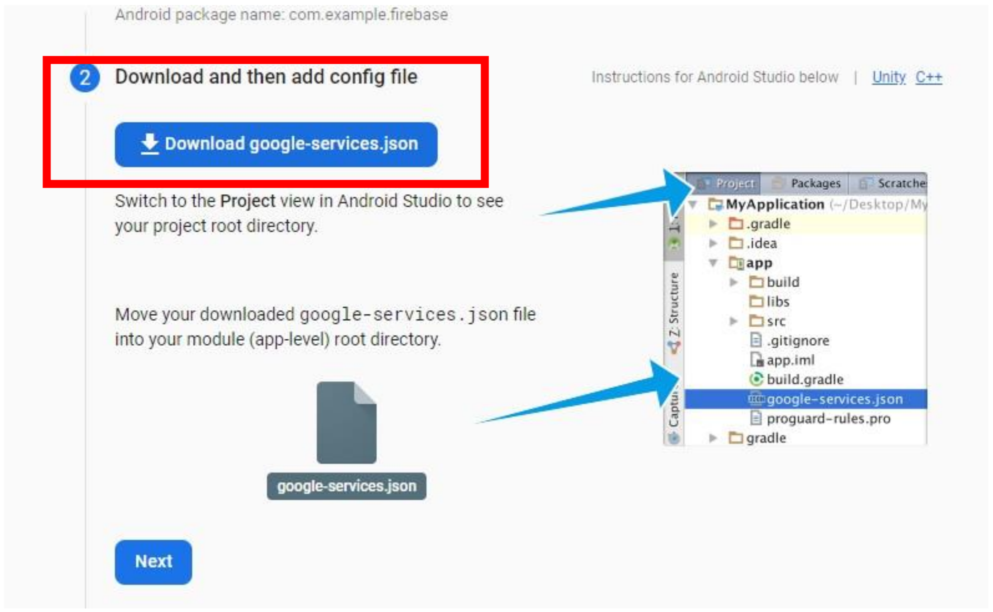
Debug signing certificate SHA-1 (optional)

00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:0

Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

11. Download the google-services.json file and copy that file to app folder that is inside Android folder and click on Next button.

…\android\app .



Android package name: com.example.firebase

2 Download and then add config file

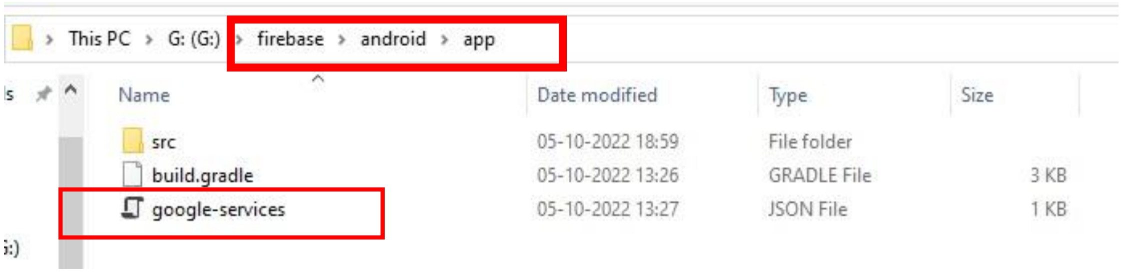
Instructions for Android Studio below | Unity C++

Download google-services.json

Switch to the **Project** view in Android Studio to see your project root directory.

Move your downloaded `google-services.json` file into your module (app-level) root directory.

Next

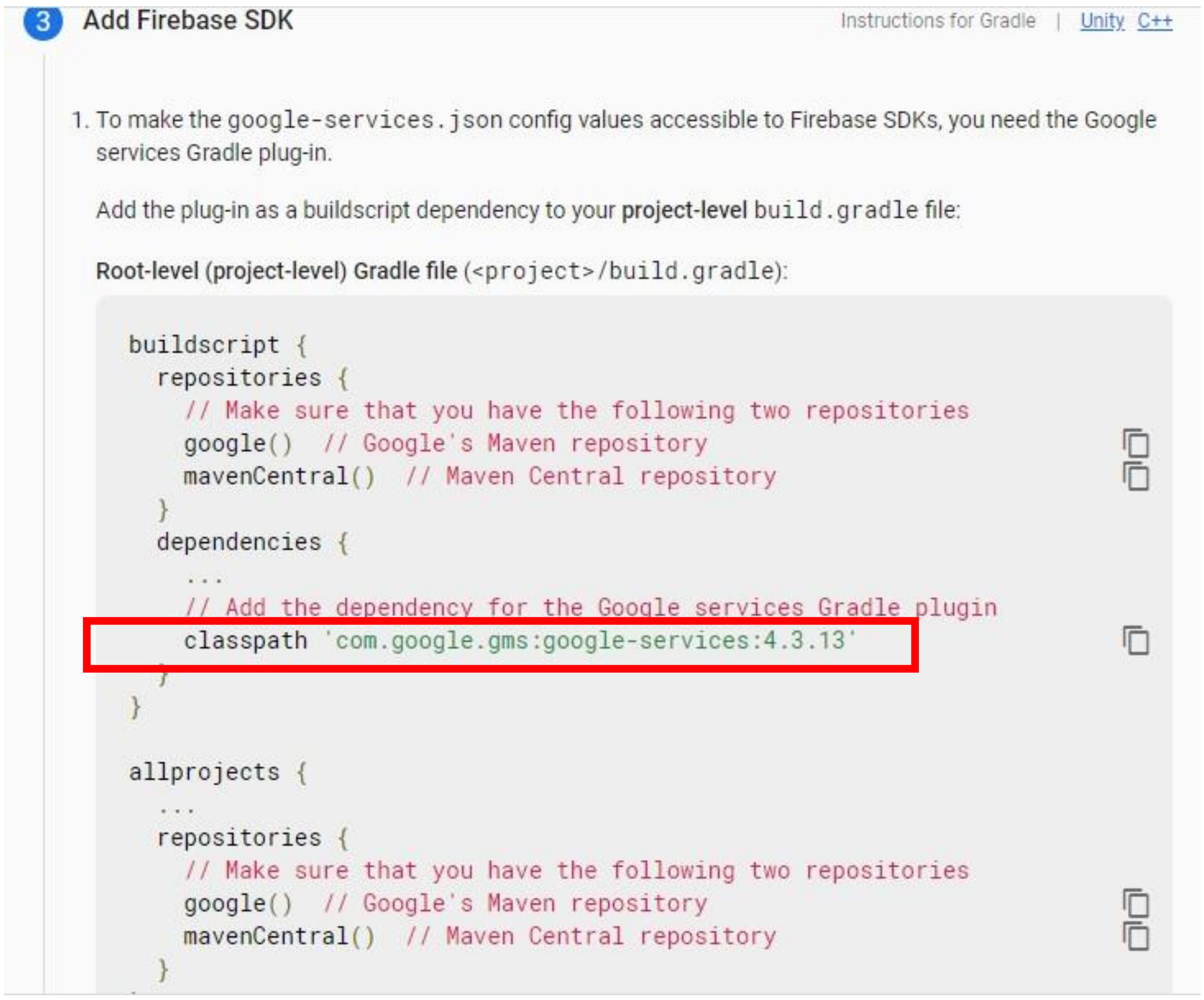


This PC › G: (G:) › firebase › android › app

Name	Date modified	Type	Size
src	05-10-2022 18:59	File folder	
build.gradle	05-10-2022 13:26	GRADLE File	3 KB
google-services	05-10-2022 13:27	JSON File	1 KB

12. Go to Android Studio project and open build.gradle file and add classpath in dependencies section.

(Note : do not go to app--> build gradle. Both are different)

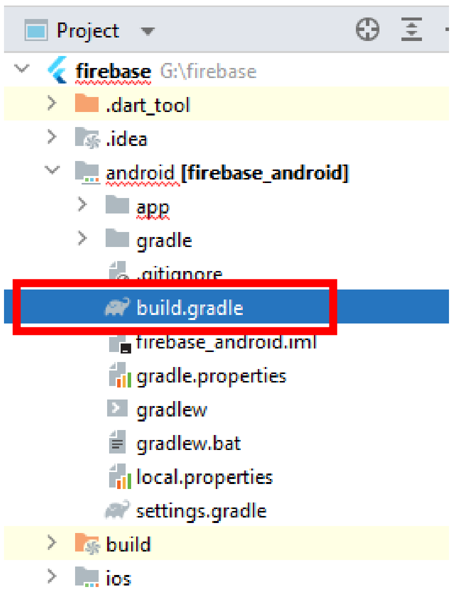


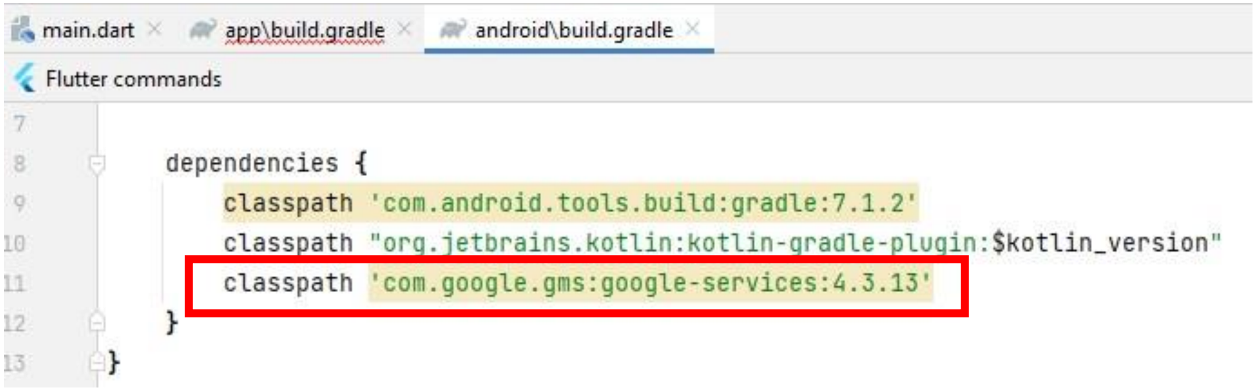
1. To make the `google-services.json` config values accessible to Firebase SDKs, you need the Google services Gradle plug-in.

   Add the plug-in as a buildscript dependency to your **project-level** `build.gradle` file:

   Root-level (project-level) Gradle file (`<project>/build.gradle`):

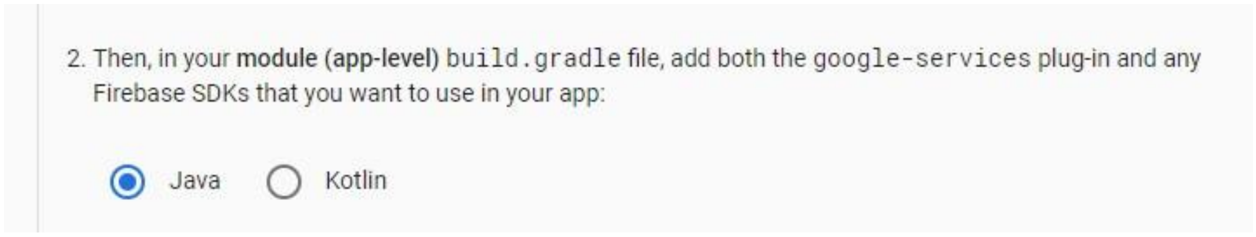
```
buildscript {
  repositories {
    // Make sure that you have the following two repositories
    google()  // Google's Maven repository
    mavenCentral()  // Maven Central repository
  }
  dependencies {
    ...
    // Add the dependency for the Google services Gradle plugin
    classpath 'com.google.gms:google-services:4.3.13'
  }
}

allprojects {
  ...
  repositories {
    // Make sure that you have the following two repositories
    google()  // Google's Maven repository
    mavenCentral()  // Maven Central repository
  }
```





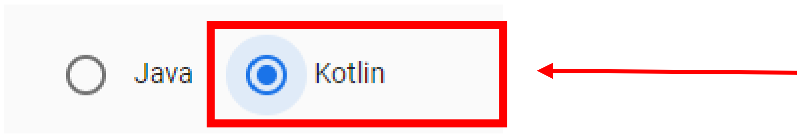
```
dependencies {
    classpath 'com.android.tools.build:gradle:7.1.2'
    classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
    classpath 'com.google.gms:google-services:4.3.13'
}
}
```

13. Now click on Kotlin radio button.



2. Then, in your **module (app-level)** `build.gradle` file, add both the `google-services` plug-in and any Firebase SDKs that you want to use in your app:

[✓ Java] Kotlin

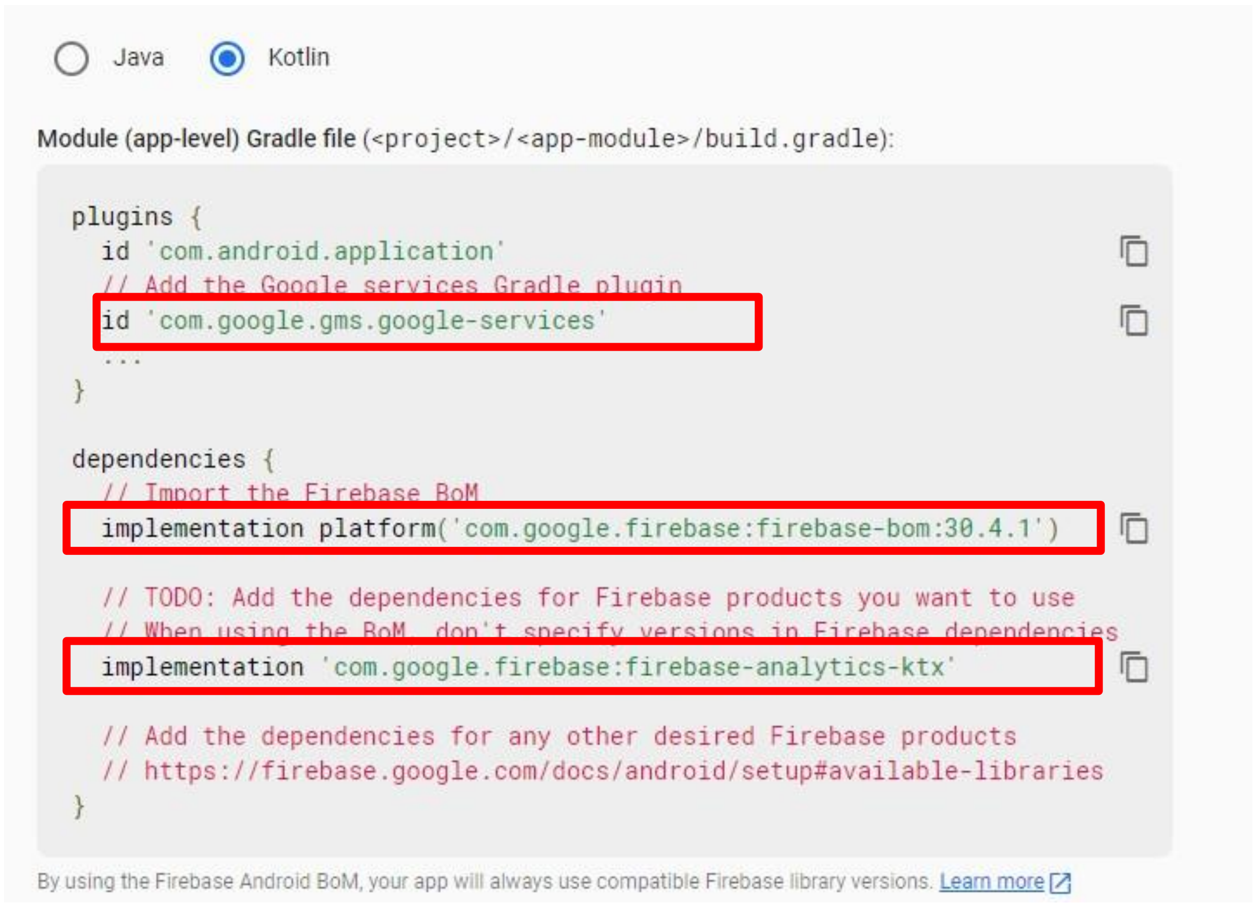


Java [✓ Kotlin]

14. Now open the build.gradle file that is present inside the app folder (android\app\build.gadle).
Copy the google service plugin and paste it in plugin section in build.gradle file.
Format of pasting plugin:

Apply plugin : paste_plugin_here

Note: Remove word id and only paste that is within quotes.

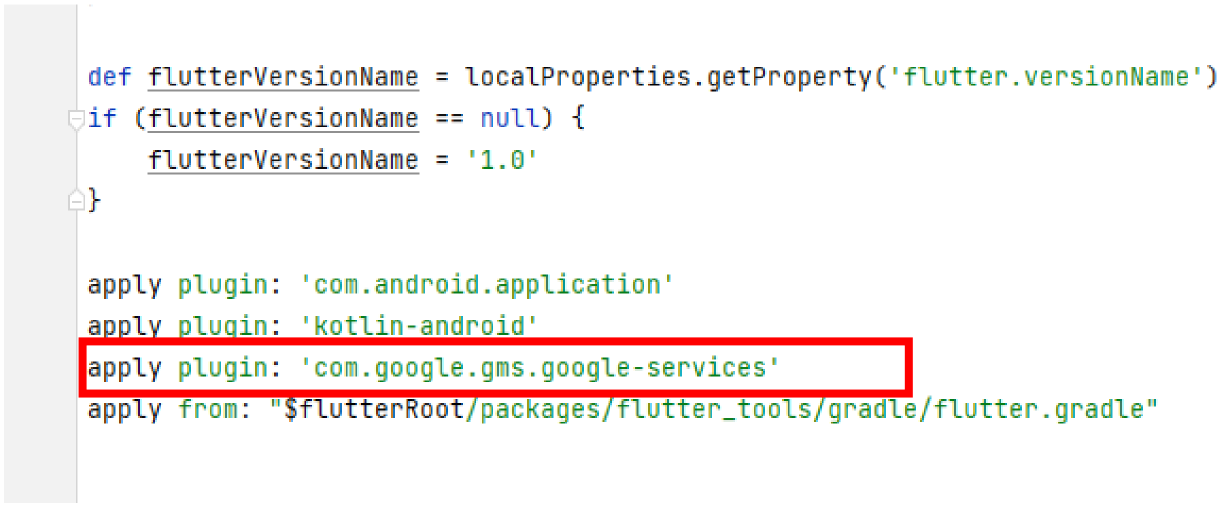


Java [✓ Kotlin]

Module (app-level) Gradle file (`<project>/<app-module>/build.gradle`):

```
plugins {
  id 'com.android.application'
  // Add the Google services Gradle plugin
  id 'com.google.gms.google-services'
  ...
}

dependencies {
  // Import the Firebase BoM
  implementation platform('com.google.firebase:firebase-bom:30.4.1')

  // TODO: Add the dependencies for Firebase products you want to use
  // When using the BoM, don't specify versions in Firebase dependencies
  implementation 'com.google.firebase:firebase-analytics-ktx'

  // Add the dependencies for any other desired Firebase products
  // https://firebase.google.com/docs/android/setup#available-libraries
}
```

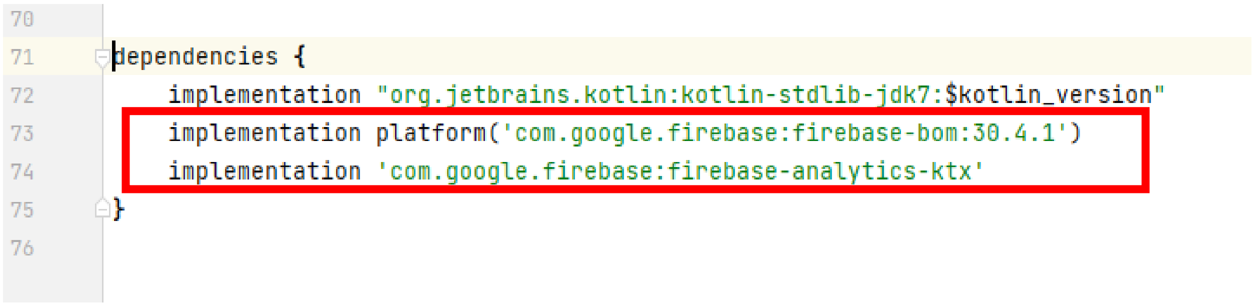
By using the Firebase Android BoM, your app will always use compatible Firebase library versions. Learn more



```
def flutterVersionName = localProperties.getProperty('flutter.versionName')
if (flutterVersionName == null) {
    flutterVersionName = '1.0'
}

apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
apply plugin: 'com.google.gms.google-services'
apply from: "$flutterRoot/packages/flutter_tools/gradle/flutter.gradle"
```

Then,

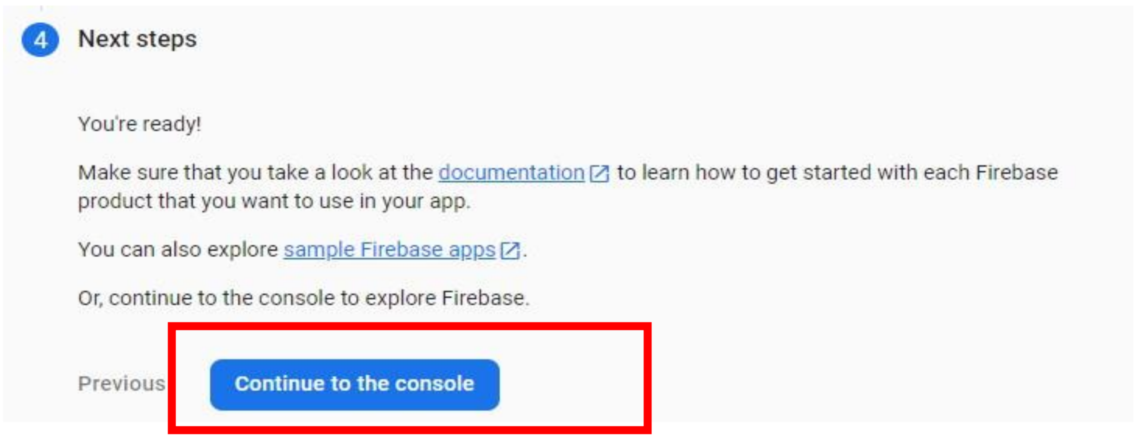
Copy both the dependencies and paste them in dependencies section.



```
dependencies {
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation platform('com.google.firebase:firebase-bom:30.4.1')
    implementation 'com.google.firebase:firebase-analytics-ktx'
}
```

Click on Next button.

15. Click on Continue to the console button



4 Next steps

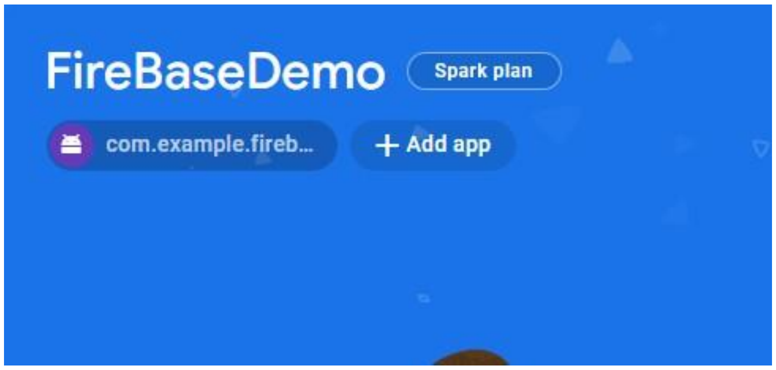
You're ready!

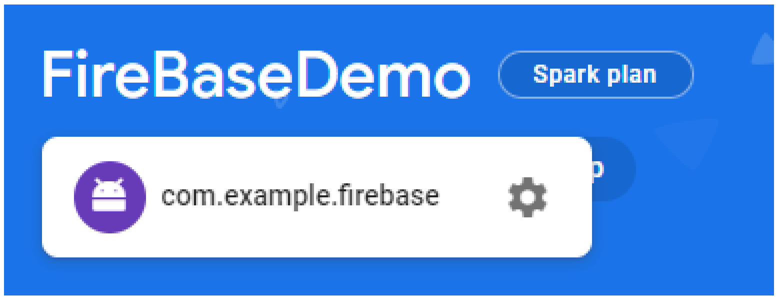
Make sure that you take a look at the documentation to learn how to get started with each Firebase product that you want to use in your app.

You can also explore sample Firebase apps.

Or, continue to the console to explore Firebase.

16. Finally your Android studio project name must be displayed in the format : com.example.your_flutter_project_name


FireBaseDemo
Spark plan
com.example.fireb...
+ Add app


FireBaseDemo
Spark plan
com.example.firebase