Aim: 10: Create a User Profile Interface using Fire-base.

In this Experiment, you will create a Flutter app that allows the app user to create their Accounts **(user name & password)** to access the app service. You will create an authentication procedure depending on ***Firebase authentication service.***

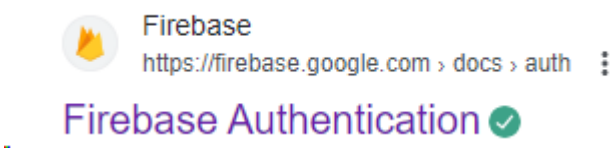
You will create the startup interface which includes the "**New User Account**" and "**Login**" buttons. When the user taps the "**New User Account**" button he/she will move to the New Account interface which will be used to create the new app user account. Also, if the user taps the **Login** button, he/she can login to the app using the account which he/she has created in the **New Account** interface. To do this, you should configure your Flutter app to use Firebase authentication service.

Follow the Steps:

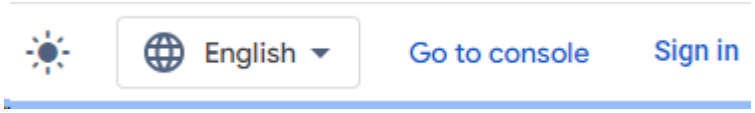
- Open **Android Studio**
- Click **File → New → New Flutter Project**
- Select **Flutter Application**, and then click **Next**.
- Type: **lab10** for Project Name, and create a new folder: Lab10 for Project Location. Click **Next**.
- Type: **Android CMR.com** for Company domain, then click **Finish.**

Configure Your Flutter App to Use Fire-Base Services:

1. Go to: **https://console.firebase.google.com**.
   Or Go to Google search → Firebase Login → Click on



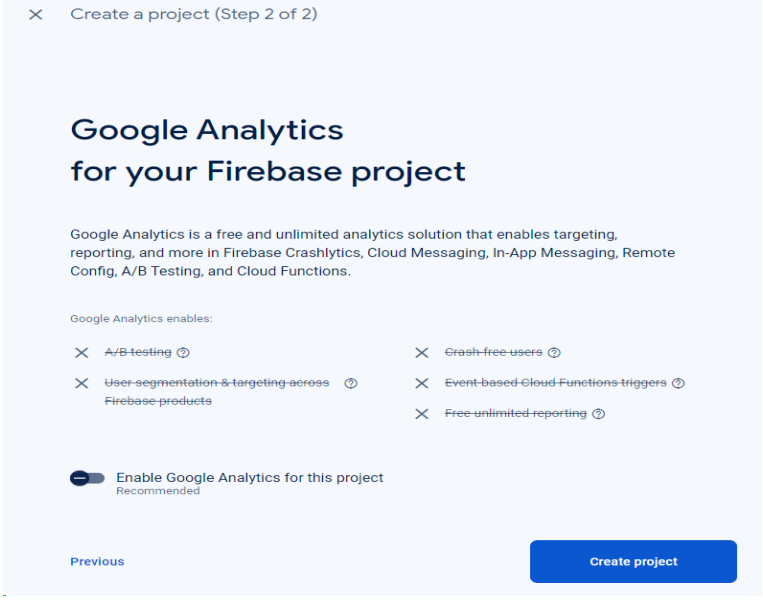
2. Go to Firebase Console on top right side



3. Sign into Firebase using your Google account.
4. Click **Get Started**, then click **Create a project**. Fill out your project name "**Lab 09**" or any other name as illustrated in the following figure.

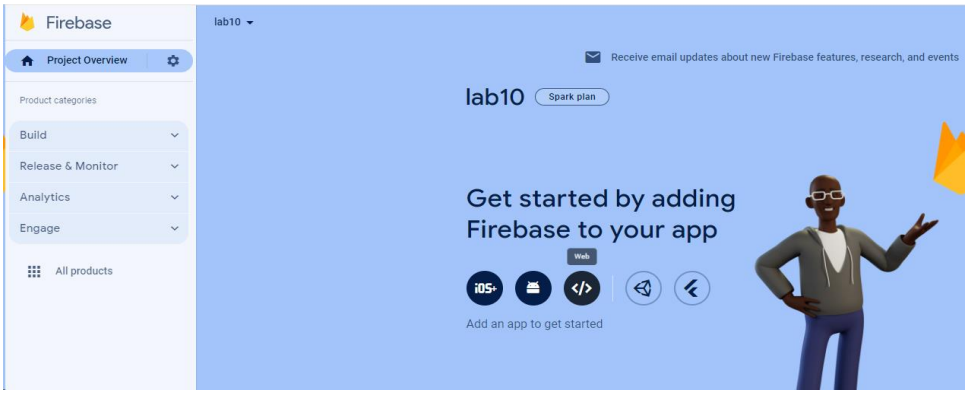
5. Check **I accept the Firebase terms**, and then click **Continue**.

Click **Create Project.** Then within seconds, you should get the following message as illustrated in the following figure. It displays your project that has been created on Google Firebase as illustrated in the following figure. Click **Continue.**

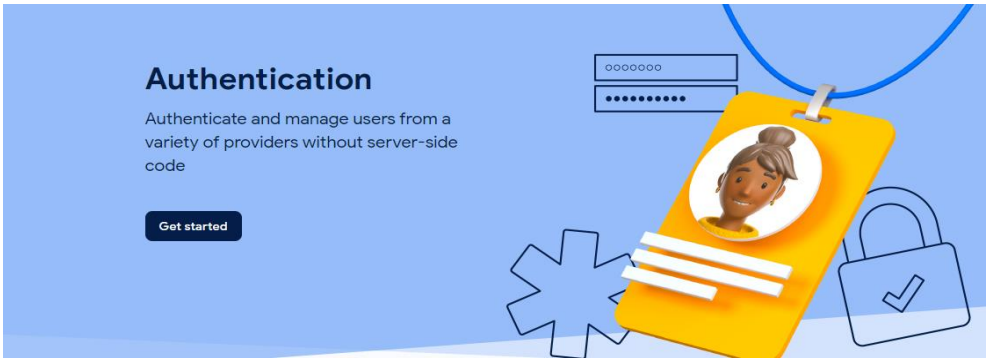


Click on Enable Google Analytics for this project and disable it → Click on Create Project

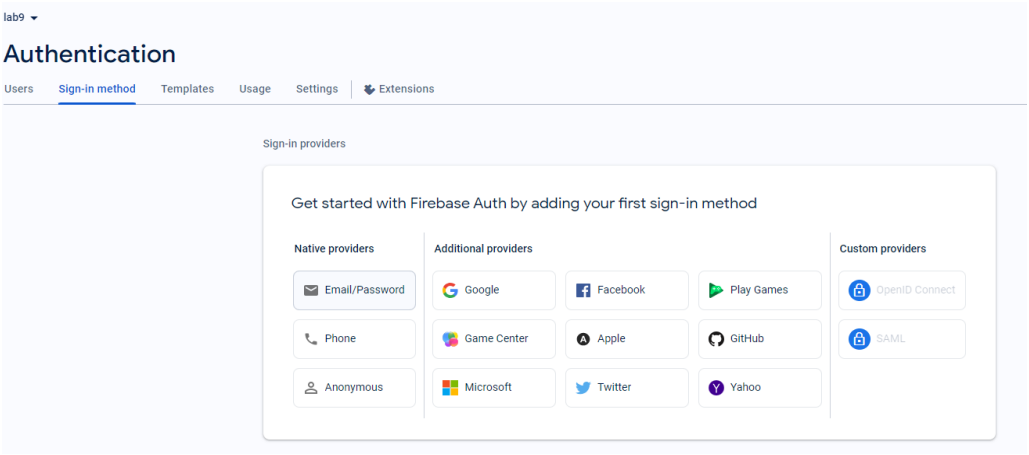
To add Firebase configurations to your Android files, click on the **Android icon** in the following figure:



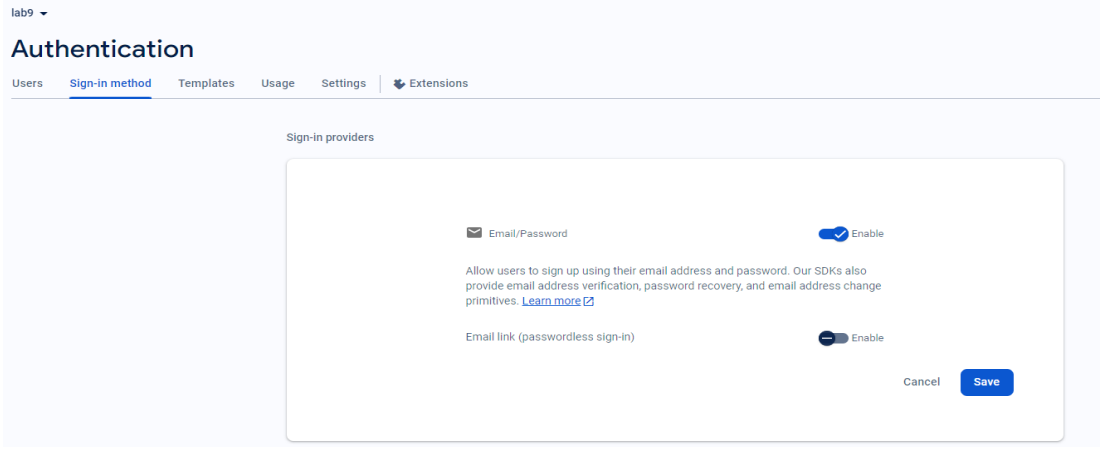
Click on **Firebase Authentication** → Click on Get Started



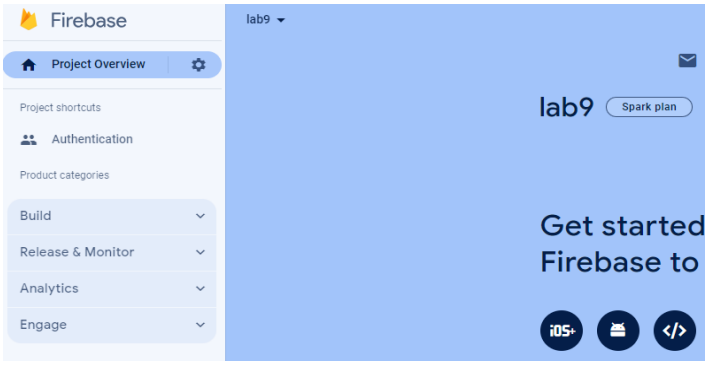
Click on E-Mail/Password



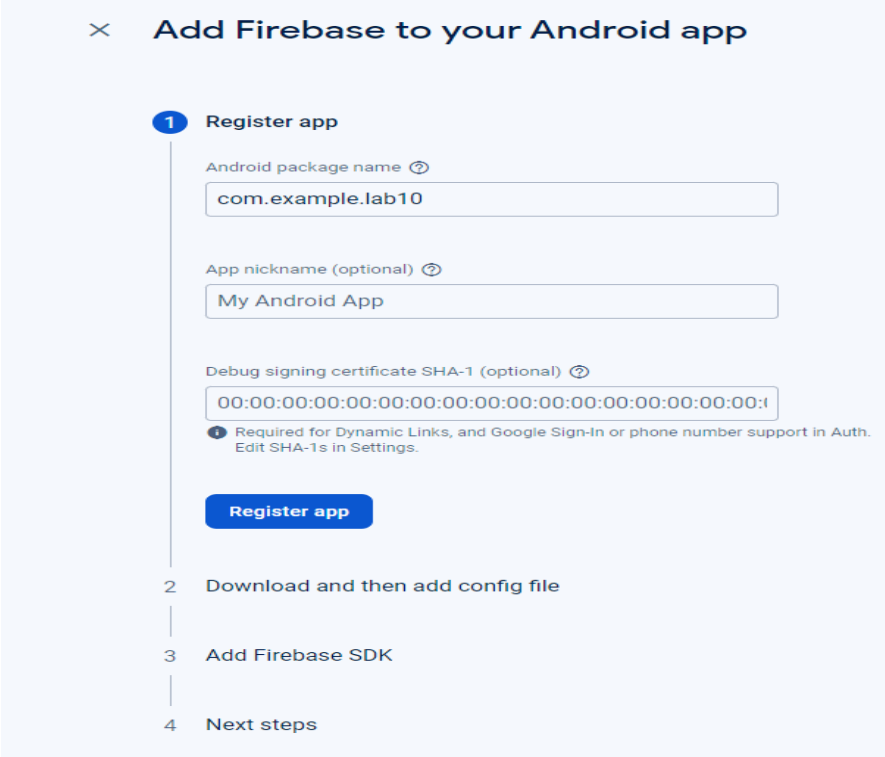
Enable E-Mail/Password and Save



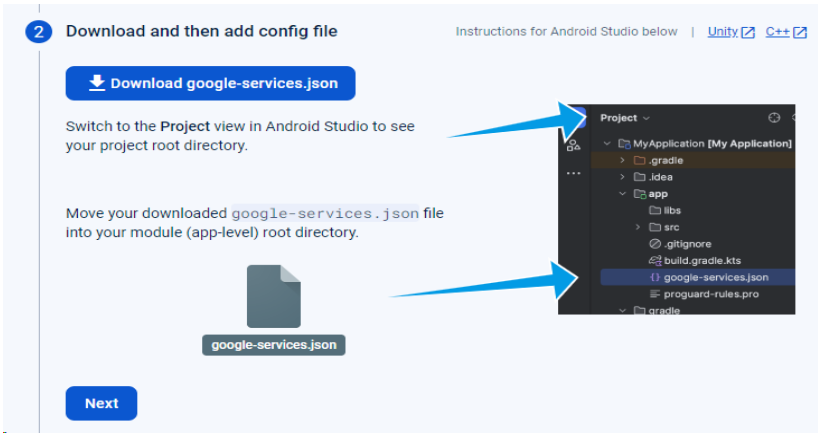
Click **on Project Overview** and select Android



Copy the package name of your project from Flutter Project → App → Src → build.gradle and paste it.



➔ Click on **Register app**



➔ Click **Download Google-service. json** button to download the JSON configuration file.

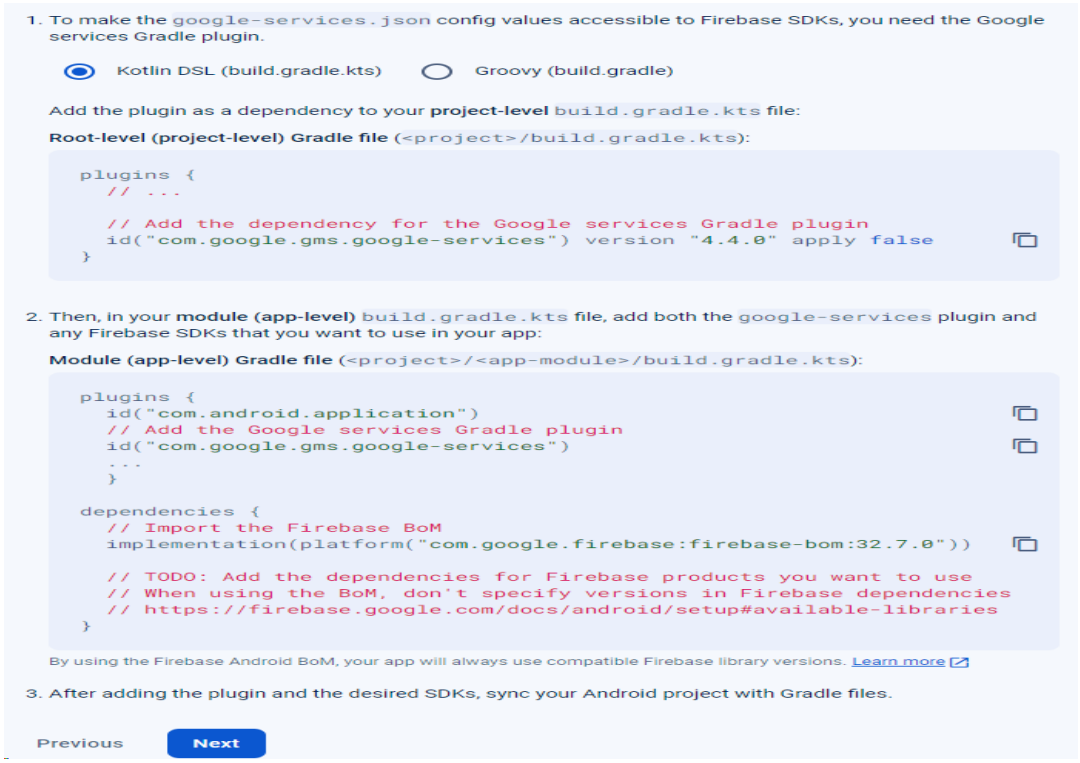
➔ Open your download folder and should find this file name: **google-services. json** without any number.

➔ Move this file: **Google-services. json** from your download folder to the **app** folder using the drag and drop technique. This **app** folder is in **Android Studio** in the following path:

Project name (**lab10**) → **android (lab10_android)** → **app**, then click **OK.**

➔ Now, return to the Firebase web site to complete the setup steps.

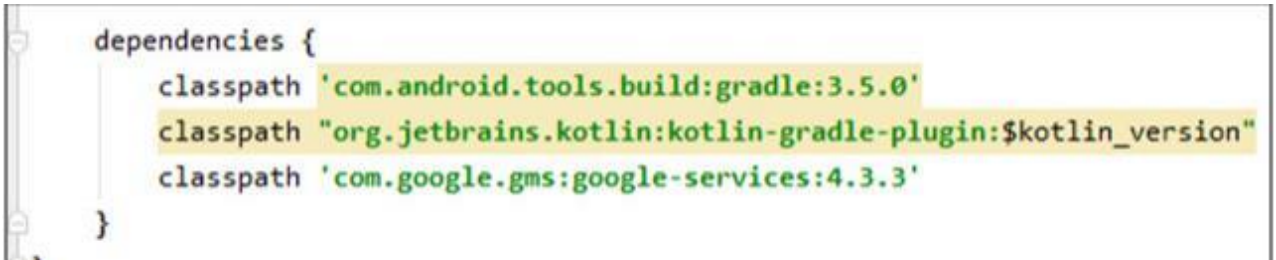
➔ Click **Next.** In the Add Firebase SDK step, copy the line illustrated in the following figure & click on the copy icon.



➔ Go to **Android Studio**, open the **build.gradle** file which is in the following path:

Your: **Project name** → **android (lab10_android)** → **build.gradle**

Paste this class path within the dependencies braces as illustrated in the following figure:



→ Now, return to the Firebase web site and from the configuration wizard, copy the other two lines :

apply plugin: 'com.android.application'

apply plugin: 'com.google.gms.google-services'

→ Go to Android Studio, open the **build.gradle** file which is in the following path:

Your Project name → **android** (**lab10_android**)→ **app** → **build.gradle**

Then, paste the other two lines as separate lines at the end of this file as illustrated in the following figure:

```
dependencies {
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test:runner:1.1.1'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
}
apply plugin: 'com.android.application'
apply plugin: 'com.google.gms.google-services'
```

Also, in the same build.gradle file add: multiDexEnabled true as illustrated in the grey

Highlighted part of the following configuration:

defaultConfig **{**

applicationId **"com.androidatc.lab_09"**

minSdkVersion 16

targetSdkVersion 28

versionCode flutterVersionCode.toInteger()

versionName flutterVersionName

testInstrumentationRunner **"androidx.test.runner.AndroidJUnitRunner"**

multiDexEnabled **true**

}

➔ Return to the Firebase web site, click **Next**.

➔ Then click the **Continue to console** button. You should get the following figure:

➔ Now, you should configure your **app settings** or add Firebase plug-in services to your app

By configuring: **pubspec.yaml** file for **Firebase authentication and database**.

Click the **cloud_firestore** plug-in , click the **Installing** tab and copy the dependencies value : cloud_firestore: ^0.13.4 or the latest update value which you will find at the time you perform this lab. The following figure displays the current **cloud_ firestore** configuration:

➔ Open your **pubspec.yaml** file in your Android Studio and paste this value under dependencies.

➔ Click **Back** on your web browser **toolbar** to get the "**Available Flutter Fire plugins**:" web page again or go to the web link: https://github.com/FirebaseExtended/ flutterfire ,

then click the : **firebase_auth** , click the **Installing** tab , then copy the existing dependencies value: **firebase_auth: ^0.15.4**

➔ Paste this value in your **pubspec.yaml** file under the dependencies.

➔ Back on your web browser to the "**Available FlutterFire plugins**:" web page or the web link: https://github.com/FirebaseExtended/flutterfire, then click the : **firebase_database**

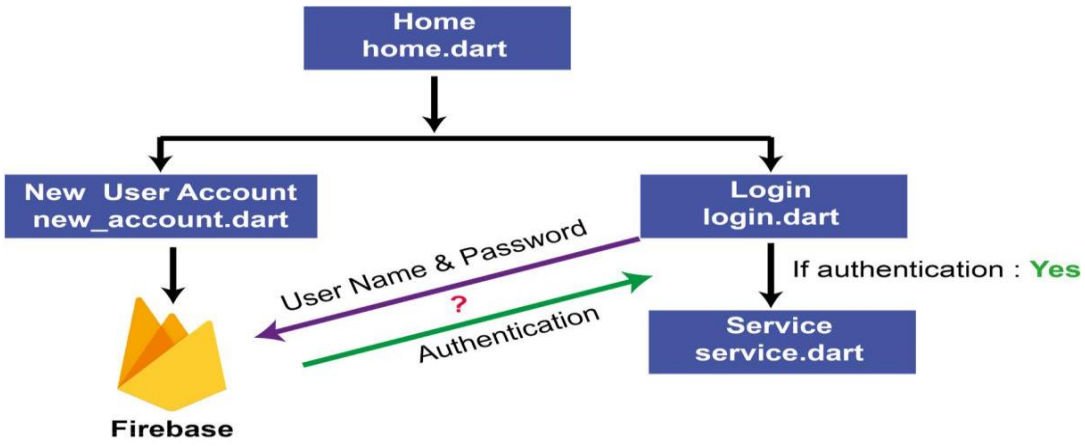
➔ Click the **Installing** tab, then copy the existing dependencies value:

firebase_database: ^3.1.3

➔ Paste this value in your **pubspec.yaml** file under the dependencies. The dependencies of the **pubspec.yaml** file should be as illustrated in the following figure:

```
19  dependencies:
20    flutter:
21      sdk: flutter
22    firebase_core: ^0.4.4
23    firebase_auth: ^0.15.4
24    firebase_database: ^3.1.1
25    cloud_firestore: ^0.13.3
```

➔ Now, at the top of the **pubspec.yaml** file content, click **Packages Get** to incorporate all of those settings into your Flutter project.



In Pubspec.yaml

Under Cupertino_icons: ^1.0.2

Paste the following: _

Go to **Google search →Flutter Fire →** Click on https://firebase.flutter.dev/

Firebase_core: ^v2.24.2

Firebase_auth: ^v4.15.3

Firebase_analytics: ^v10.7.4

Firebase_database: ^v4.15.3

Cloud_firestore: ^v10.3.8

Main.dart:

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:lab10/Login.dart';
void main() async{
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(MyApp());
}
class MyApp extends StatelessWidget {
  final Future<FirebaseApp> _initialization = Firebase.initializeApp();
  @override
```

```
  Widget build(BuildContext context) {

    return FutureBuilder(

        future: _initialization,
        builder: (context, snapshot) {
          // Check for Errors
          if (snapshot.hasError) {
            print("Something Went Wrong");
          }
          if (snapshot.connectionState == ConnectionState.waiting) {
            return Center(child: CircularProgressIndicator());
          }
          return MaterialApp(
            title: 'Flutter Firebase EMail Password Auth',
            theme: ThemeData(
              primarySwatch: Colors.deepOrange,
            ),
            debugShowCheckedModeBanner: false,
            home: Login(),
          );
        });
  }
}
```

Login. dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:lab10/SignUp.dart';
import 'package:lab10/Dashboard.dart';

class Login extends StatefulWidget {
  Login({Key? key}) : super(key: key);
```

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


class _LoginState extends State<Login> {
  final _formKey = GlobalKey<FormState>();

  var email = "";
  var password = "";
  // Create a text controller and use it to retrieve the current value
  // of the TextField.
  final emailController = TextEditingController();
  final passwordController = TextEditingController();

  userLogin() async {
    try {
      await FirebaseAuth.instance
          .signInWithEmailAndPassword(email: email, password: password);
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(
          builder: (context) => Dashboard(),
        ),
      );
    } on FirebaseAuthException catch (e) {
      if (e.code == 'user-not-found') {
        print("No User Found for that Email");
        ScaffoldMessenger.of(context).showSnackBar(
          SnackBar(
            backgroundColor: Colors.orangeAccent,
            content: Text(
              "No User Found for that Email",
              style: TextStyle(fontSize: 18.0, color: Colors.black),
```

```
        ),
      ),
    );
  }


   else if (e.code == 'wrong-password') {
    print("Wrong Password Provided by User");
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        backgroundColor: Colors.orangeAccent,
        content: Text(
          "Wrong Password Provided by User",
          style: TextStyle(fontSize: 18.0, color: Colors.black),
        ),
      ),
    );
  }
 }
}

@override
void dispose() {
 // Clean up the controller when the widget is disposed.
 emailController.dispose();
 passwordController.dispose();
 super.dispose();
}
@override
Widget build(BuildContext context) {
 return Scaffold(
  appBar: AppBar(
   title: Text("User Login"),
  ),
```

```
body: Form(
  key: _formKey,
  child: Padding(




  padding: EdgeInsets.symmetric(vertical: 20, horizontal: 30),
   child: ListView(
    children: [
     Container(
      margin: EdgeInsets.symmetric(vertical: 10.0),
      child: TextFormField(
       autofocus: false,
       decoration: InputDecoration(
        labelText: 'Email: ',
        labelStyle: TextStyle(fontSize: 20.0),
        border: OutlineInputBorder(),
        errorStyle:
        TextStyle(color: Colors.redAccent, fontSize: 15),
       ),
       controller: emailController,
       validator: (value) {
        if (value == null || value.isEmpty) {
         return 'Please Enter Email';
        } else if (!value.contains('@')) {
         return 'Please Enter Valid Email';
        }
        return null;
       },
      ),
     ),

Container(
```

```
  margin: EdgeInsets.symmetric(vertical: 10.0),
  child: TextFormField(
    autofocus: false,
    obscureText: true,
    decoration: InputDecoration(



      labelText: 'Password: ',
      labelStyle: TextStyle(fontSize: 20.0),
      border: OutlineInputBorder(),
      errorStyle:
      TextStyle(color: Colors.redAccent, fontSize: 15),
    ),
    controller: passwordController,
    validator: (value) {
      if (value == null || value.isEmpty) {
        return 'Please Enter Password';
      }
      return null;
    },
  ),
),
Container(
  margin: EdgeInsets.only(left: 60.0),
  child: Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      ElevatedButton(
        onPressed: () {

          // Validate returns true if the form is valid, otherwise false.
          if (_formKey.currentState!.validate()) {
            setState(() {
```

```
            email = emailController.text;
            password = passwordController.text;
          });
          userLogin();
        }
      },



child: Text(
          'Login',
          style: TextStyle(fontSize: 18.0),
        ),
      ),
      TextButton(
        onPressed: () => {
        },
        child: Text(
          'Forgot Password ?',
          style: TextStyle(fontSize: 14.0),
        ),
      ),
    ],
  ),
),
Container(
  child: Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Text("Don't have an Account? "),
      TextButton(
        onPressed: () => {
          Navigator.pushAndRemoveUntil(
            context,
```

```
                    PageRouteBuilder(
                      pageBuilder: (context, a, b) => Signup(),
                      transitionDuration: Duration(seconds: 0),
                    ),
                        (route) => false)
                },
                child: Text('Signup'),
              ),

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

Signup. Dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:create_user_profile/Login.dart';

class Signup extends StatefulWidget {
  Signup({Key? key}) : super(key: key);
  @override
  _SignupState createState() => _SignupState();
}
class _SignupState extends State<Signup> {
  final _formKey = GlobalKey<FormState>();
  var email = "";
```

```
var password = "";
var confirmPassword = "";

// Create a text controller and use it to retrieve the current value
// of the TextField.
final emailController = TextEditingController();
final passwordController = TextEditingController();
final confirmPasswordController = TextEditingController();



@override
void dispose() {
  // Clean up the controller when the widget is disposed.
  emailController.dispose();
  passwordController.dispose();
  confirmPasswordController.dispose();
  super.dispose();
}

registration() async {
  if (password == confirmPassword) {
    try {
      UserCredential userCredential = await FirebaseAuth.instance
          .createUserWithEmailAndPassword(email: email, password: password);
      print(userCredential);
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
          backgroundColor: Colors.redAccent,
          content: Text(
            "Registered Successfully. Please Login..",
            style: TextStyle(fontSize: 20.0),
          ),
        ),
```

```
    );
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(
        builder: (context) => Login(),
      ),
    );
  } on FirebaseAuthException catch (e) {
    if (e.code == 'weak-password') {
      print("Password Provided is too Weak");
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
          backgroundColor: Colors.orangeAccent,
          content: Text(
            "Password Provided is too Weak",
            style: TextStyle(fontSize: 18.0, color: Colors.black),
          ),
        ),
      );
    } else if (e.code == 'email-already-in-use') {
      print("Account Already exists");
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
          backgroundColor: Colors.orangeAccent,
          content: Text(
            "Account Already exists",
            style: TextStyle(fontSize: 18.0, color: Colors.black),
          ),
        ),
      );
    }
  }
} else {
  print("Password and Confirm Password doesn't match");
```

```
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        backgroundColor: Colors.orangeAccent,
        content: Text(
          "Password and Confirm Password doesn't match",
          style: TextStyle(fontSize: 16.0, color: Colors.black),
        ),
      ),
    );
  }
}



@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text("User SignUp"),
    ),
    body: Form(
      key: _formKey,
      child: Padding(
        padding: EdgeInsets.symmetric(vertical: 20, horizontal: 30),
        child: ListView(
          children: [
            Container(
              margin: EdgeInsets.symmetric(vertical: 10.0),
              child: TextFormField(
                autofocus: false,
                decoration: InputDecoration(
                  labelText: 'Email: ',
                  labelStyle: TextStyle(fontSize: 20.0),
                  border: OutlineInputBorder(),
                  errorStyle:
```

```
      TextStyle(color: Colors.redAccent, fontSize: 15),
    ),
    controller: emailController,
    validator: (value) {
      if (value == null || value.isEmpty) {
        return 'Please Enter Email';
      } else if (!value.contains('@')) {
        return 'Please Enter Valid Email';
      }
      return null;
    },
  ),
),

Container(
  margin: EdgeInsets.symmetric(vertical: 10.0),
  child: TextFormField(
    autofocus: false,
    obscureText: true,
    decoration: InputDecoration(
      labelText: 'Password: ',
      labelStyle: TextStyle(fontSize: 20.0),
      border: OutlineInputBorder(),
      errorStyle:
      TextStyle(color: Colors.redAccent, fontSize: 15),
    ),
    controller: passwordController,
    validator: (value) {
      if (value == null || value.isEmpty) {
        return 'Please Enter Password';
      }
      return null;
    },
  ),
```

```
),
Container(
  margin: EdgeInsets.symmetric(vertical: 10.0),
  child: TextFormField(
    autofocus: false,
    obscureText: true,
    decoration: InputDecoration(
      labelText: 'Confirm Password: ',
      labelStyle: TextStyle(fontSize: 20.0),




     border: OutlineInputBorder(),
      errorStyle:
      TextStyle(color: Colors.redAccent, fontSize: 15),
    ),
    controller: confirmPasswordController,
    validator: (value) {
      if (value == null || value.isEmpty) {
        return 'Please Enter Password';
      }
      return null;
    },
  ),
),
Container(
  child: Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      ElevatedButton(
        onPressed: () {
          // Validate returns true if the form is valid, otherwise false.
```

```
          if (_formKey.currentState!.validate()) {
            setState(() {
              email = emailController.text;
              password = passwordController.text;
              confirmPassword = confirmPasswordController.text;
            });
            registration();
          }
        },
        child: Text(
          'Sign Up',
          style: TextStyle(fontSize: 18.0),
        ),
      ),
    ],
  ),
),
Container(
  child: Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Text("Already have an Account? "),
      TextButton(
          onPressed: () => {
            Navigator.pushReplacement(
              context,
              PageRouteBuilder(
                pageBuilder:
                    (context, animation1, animation2) =>
                    Login(),
                transitionDuration: Duration(seconds: 0),
              ),
            )
          },
```

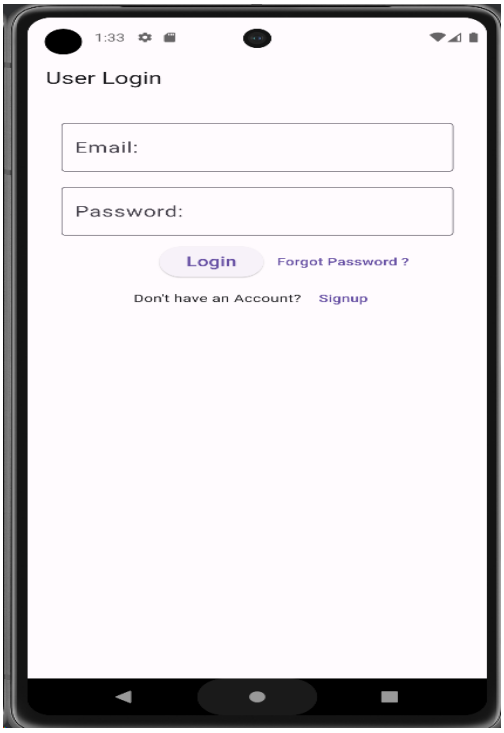
```
                    child: Text('Login'))
                ],
              ),
            )
          ],
        ),
      ),
    ),
  );
 }
}
```

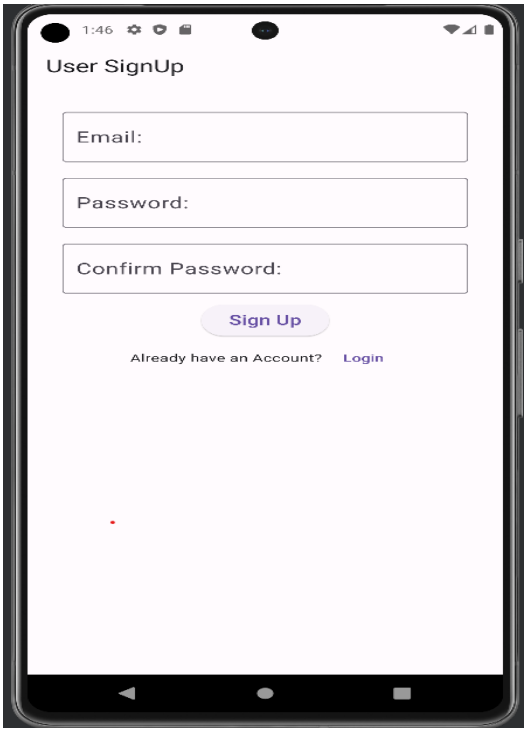
Dashboard. dart

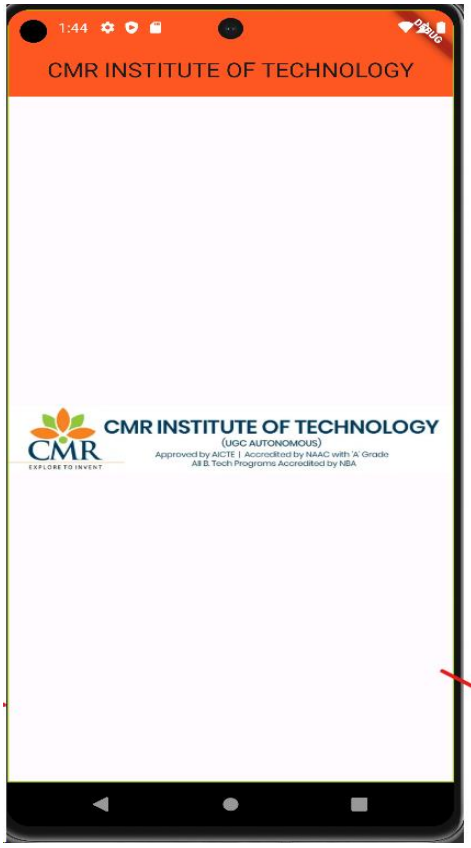
```
import 'package:flutter/material.dart';

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

 @override
 Widget build(BuildContext context) {
  // Material App
  return MaterialApp(
    home:Scaffold(
      appBar: AppBar(
        title: Text("CMR INSTITUTE OF TECHNOLOGY"),
        centerTitle: true,
        backgroundColor: Colors.deepOrange,
```

```
        ),
        body: Center(
          child: Image(
            image: NetworkImage('https://i0.wp.com/cmrithyderabad.edu.in/wp-content/uploads/2021/09/cropped-CMR-IT-logo-1.webp?w=731&ssl=1'),
          ),
        )
      ));
  }
}
```

Output:-


1:33
User Login
Email:
Password:
Login
Forgot Password ?
Don't have an Account?
Signup


1:46
User SignUp
Email:
Password:
Confirm Password:
Sign Up
Already have an Account?
Login


1:44
CMR INSTITUTE OF TECHNOLOGY
CMR
EXPLORE TO INVENT
CMR INSTITUTE OF TECHNOLOGY
(UGC AUTONOMOUS)
Approved by AICTE | Accredited by NAAC with 'A' Grade
All B. Tech Programs Accredited by NBA