FIREBASE TUTORIAL

Firebase is a Backend – as-a-Service. It provides variety of tools and services to help develop quality apps developed by Google. It is a NoSQL database program which stores data in JSON like documents.

Features:

1) **Authentication** – It supports authentication such as log in with Google, Facebook etc. You can add one or more sign-in methods in your app.

2) **Hosting** – It provides fast hosting for web apps. Firebase utilizes Superstatic and runs it as BrowserSync middleware. BrowserSync handles reloading the app across all connected devices.

3) **Realtime database** – Data is synced across all clients using WebSockets. You receive the updated data as soon as changes are saved.

4) **File Storage** – it provides a simple way to save any type of data. It has its own security rules to protect data.

5) **Test lab** – the app is tested on virtual and physical devices located in Google's data centers.

It has a bunch of other features as well:

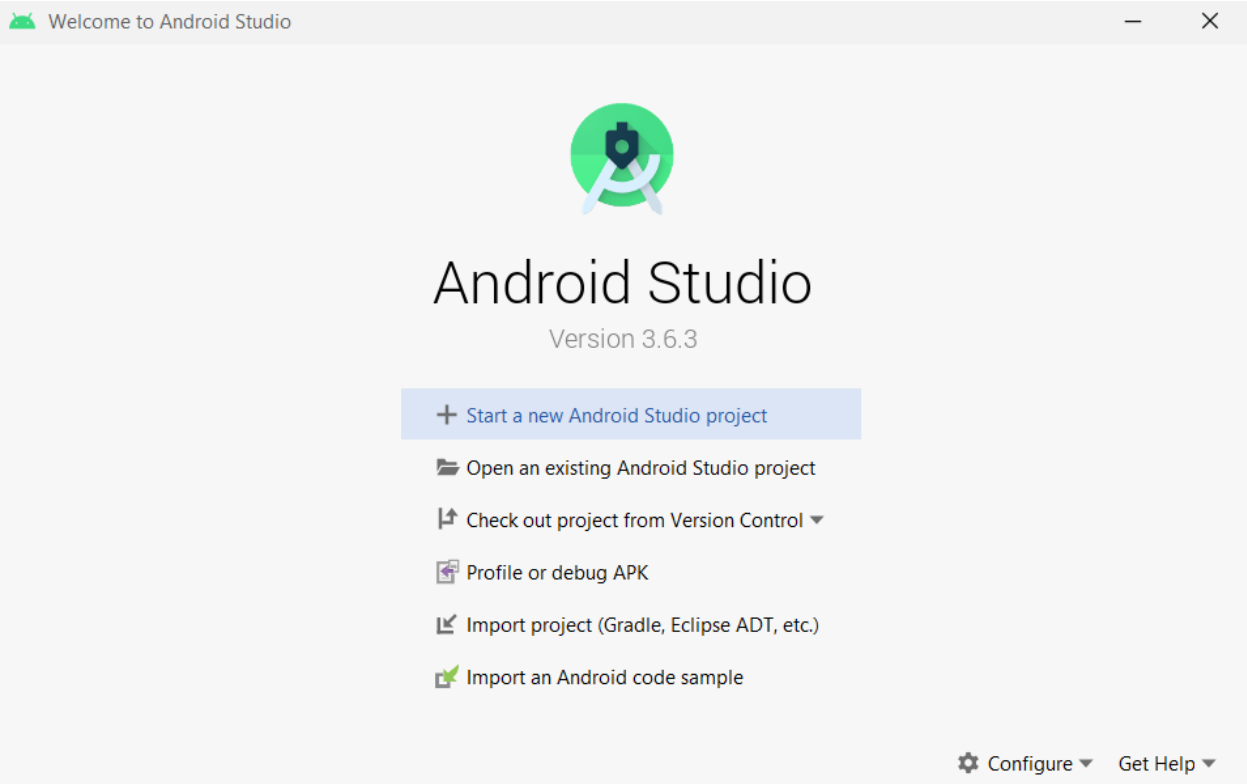
- Remote Config
- Crash
- Notifications
- AdMob

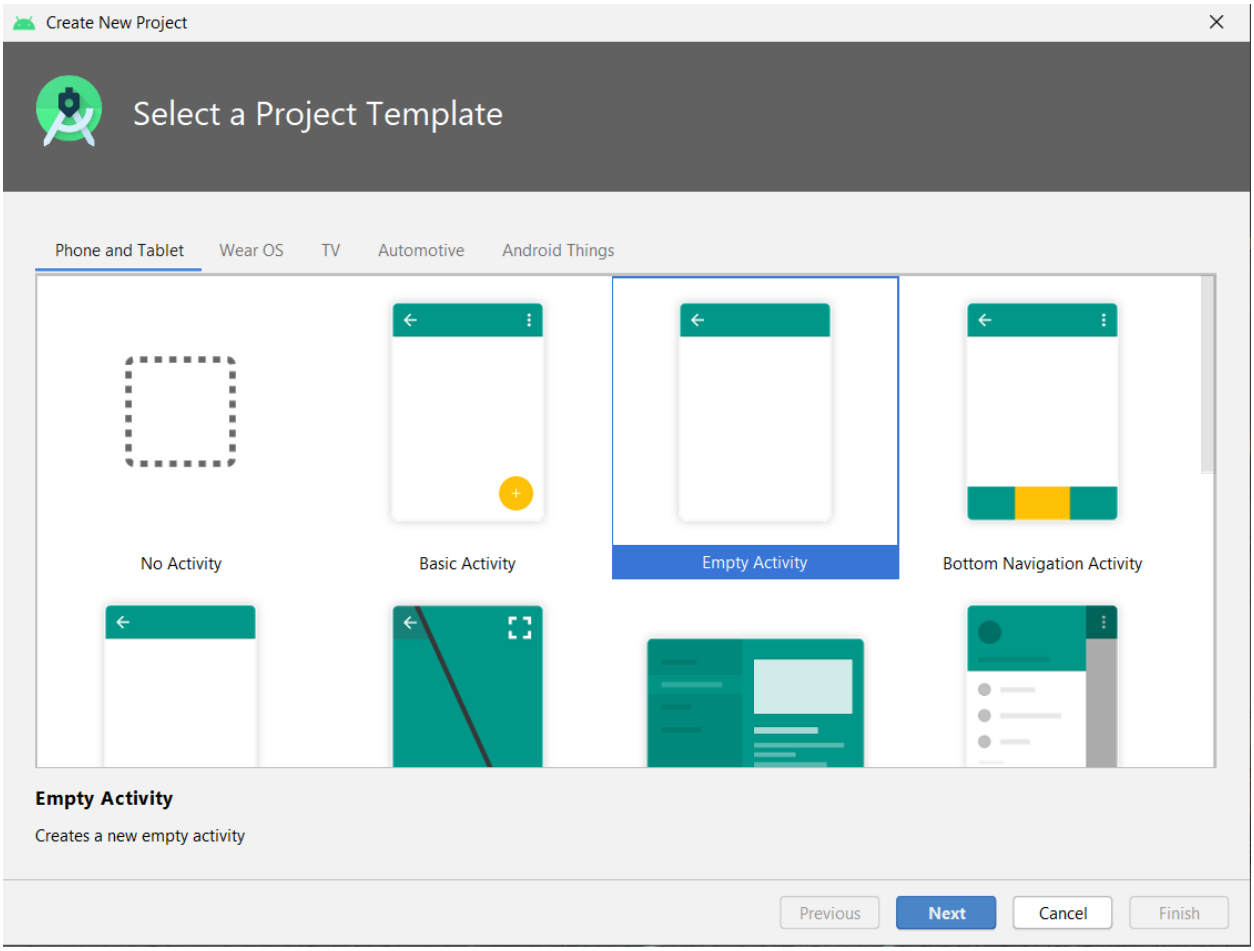
Cons:

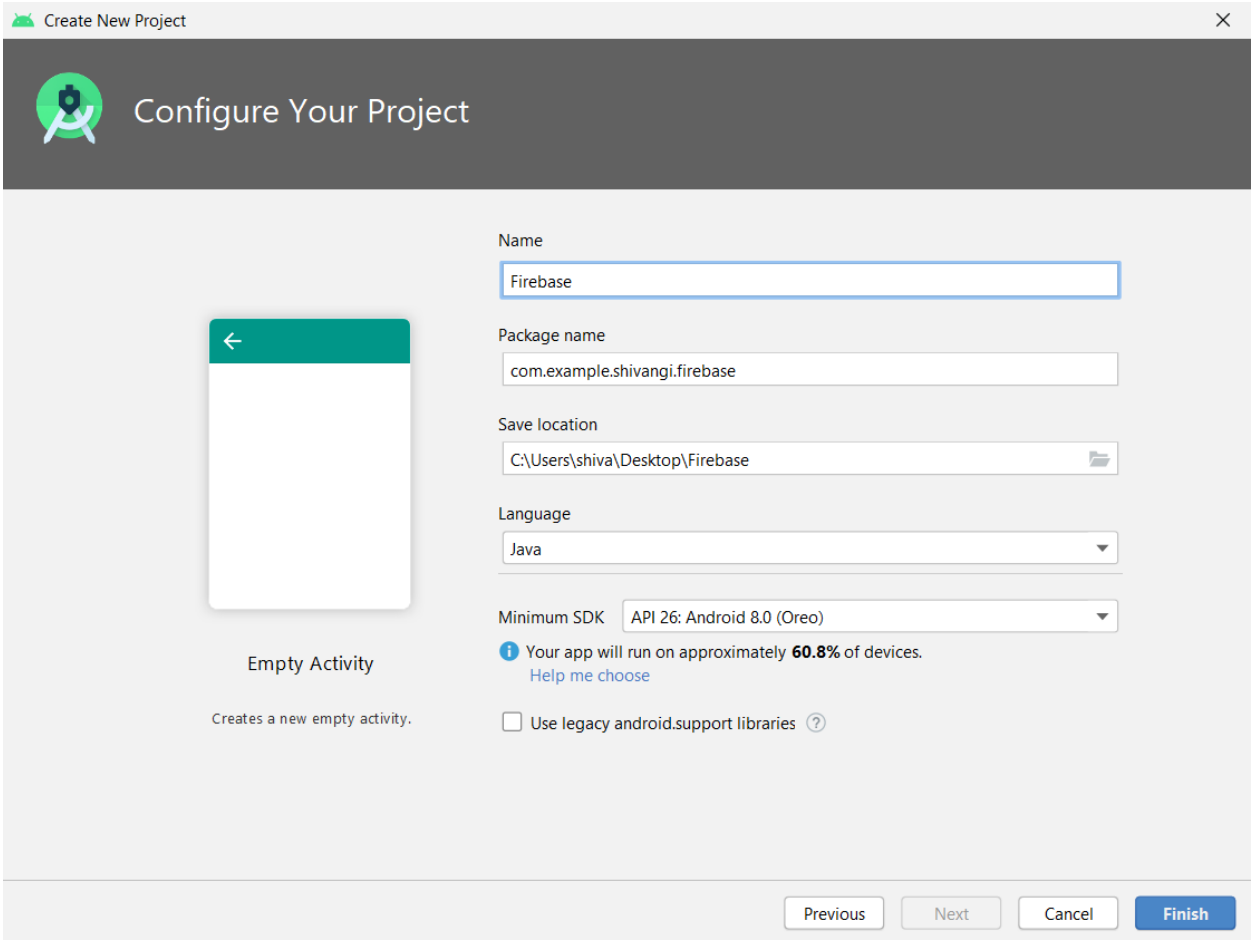
- Limited query abilities.
- Traditional relational data models are not applicable.

LET'S BEGIN!

Create a new project and name it Firebase.


Welcome to Android Studio
Android Studio
Version 3.6.3
Start a new Android Studio project
Open an existing Android Studio project
Check out project from Version Control
Profile or debug APK
Import project (Gradle, Eclipse ADT, etc.)
Import an Android code sample
Configure
Get Help


Create New Project
Select a Project Template
Phone and Tablet
Wear OS
TV
Automotive
Android Things
No Activity
Basic Activity
Empty Activity
Bottom Navigation Activity
Empty Activity
Creates a new empty activity
Previous
Next
Cancel
Finish



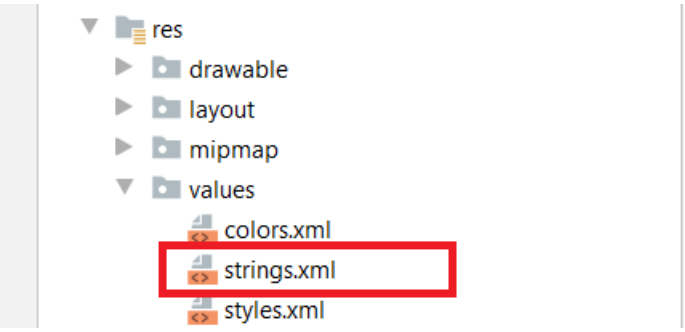
After the project syncs, the MainActivity.java looks like this.

activity_main.xml | MainActivity.java

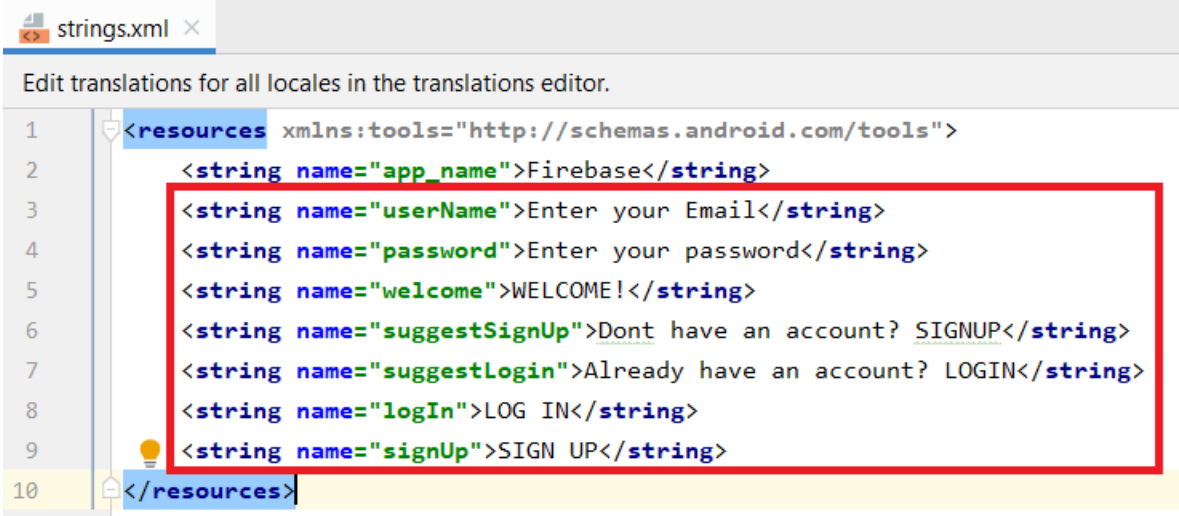
```java
package com.example.shivangi.firebase;

import ...

public class MainActivity extends AppCompatActivity {

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

First, we will add strings to our strings.xml file. We will use them in our layout so we don't hardcode the texts.

Go to your values folder in your res folder and click on the strings.xml file.



Add the following strings to your file.



Let's create our log in layout in our activity_main.xml file. The layout consists of two edit texts, one for email and the other one for password. A button to perform the log in operation when Clicked and a TextView that redirects to the sign up page if the user doesn't have an account.

In the button and TextView component, we have a red line in the onClick property. This is because the property is referring to a method in our MainActivity and we haven't yet created this method. To create it, click on the red line, a red bulb will appear on the left. Click on the Create 'onClick(View)' in MainActivity from the drop down list.

```
43      <Button
44          android:id="@+id/logIn"
45          android:layout_width="wrap_content"
46          android:layout_height="wrap_content"
47          android:layout_marginBottom="40dp"
48          android:onClick="onClick"
49          Suppress: Add tools:ignore="OnClick" attribute
50          Create 'onClick(View)' in 'MainActivity'        pOf="@+id/textViewSuggestSignUp"
51          app:layout_constraintEnd_toEndOf="parent"
52          app:layout_constraintHorizontal_bias="0.498"
53          app:layout_constraintStart_toStartOf="parent"
54          app:layout_constraintTop_toBottomOf="@+id/logInPassword"
55          app:layout_constraintVertical_bias="1.0" />
56
```

This will create the method for you in the MainActivity class.

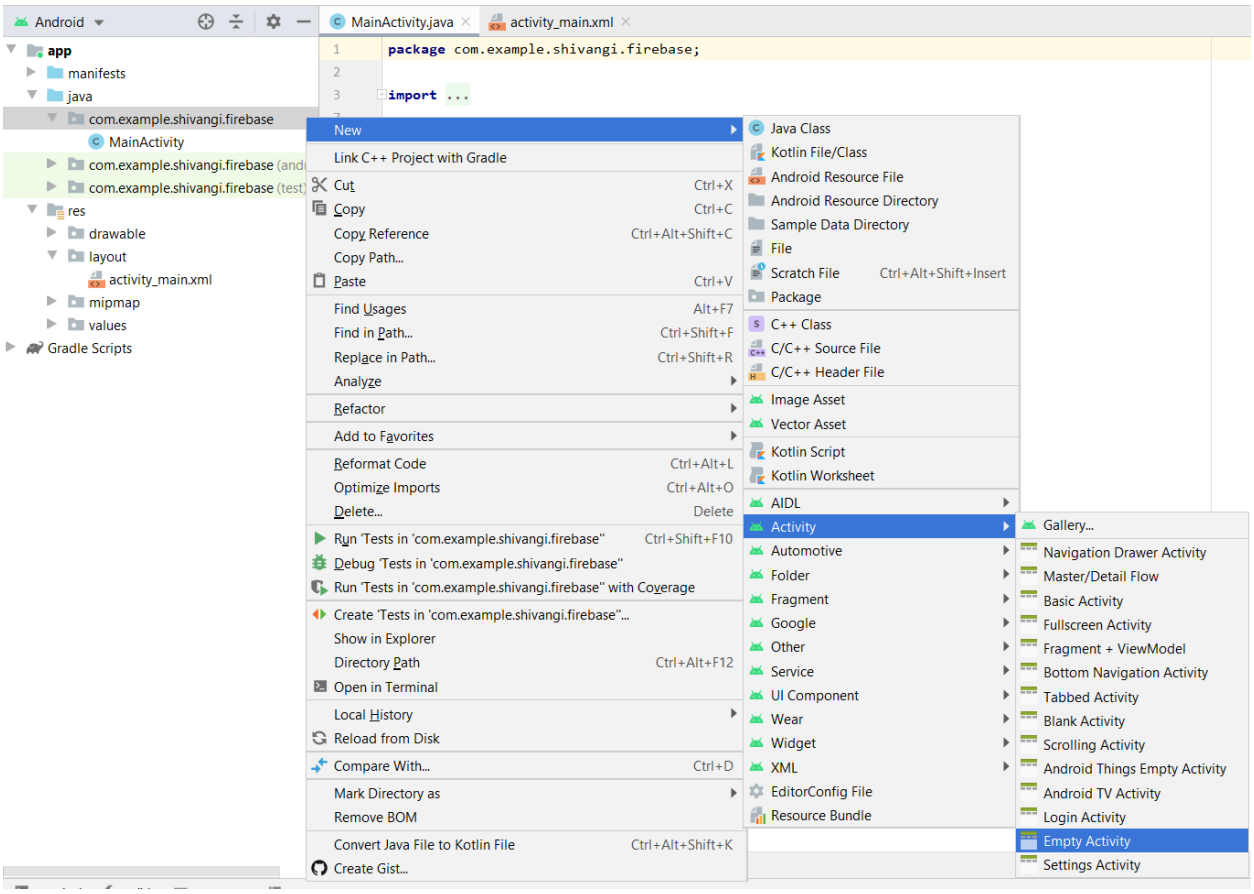
MainActivity.java | SignUp.java | activity_main.xml

```
1   package com.example.shivangi.firebase;
2
3   import ...
6
7   public class MainActivity extends AppCompatActivity {
8
9       @Override
10      protected void onCreate(Bundle savedInstanceState) {
11          super.onCreate(savedInstanceState);
12          setContentView(R.layout.activity_main);
13
14      }
15
16      public void onClick(View view) {
17
18      }
19  }
```

We will also create a new Empty Activity and name it SignUp. This activity will allow the user to sign up.

To create a new activity, right-click on your package folder and select **New** > **Activity** > **Empty Activity**.

Name the activity SignUp and click on Finish.



Your SignUp activity will look as follows:

MainActivity.java | activity_sign_up.xml | SignUp.java | activity_main.xml

```java
package com.example.shivangi.firebase;

import ...

public class SignUp extends AppCompatActivity {

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

Let's create a layout for our SignUp class. Add the following code in your activity_sign_up.xml file. The layout consists of two edit texts, one for email and the other one for password. A button to perform the sign up operation when Clicked and a TextView that redirects to the login page if the user already has an account. Add the onClick methods in the SignUp class.

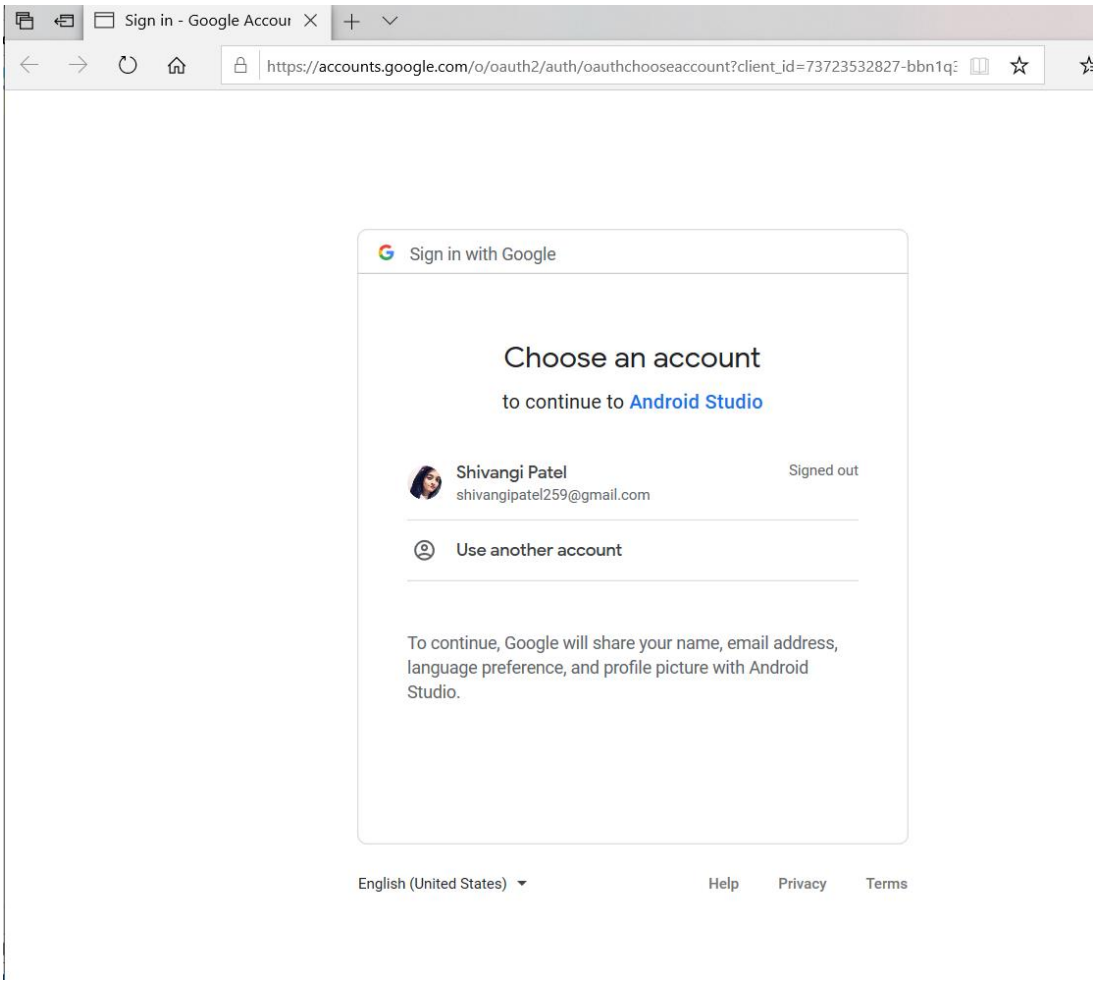
```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"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

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="71dp"
        android:layout_marginBottom="62dp"
        android:text="SIGN UP"
        android:textSize="24sp"
        app:layout_constraintBottom_toTopOf="@+id/signUpPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/signUpPersonName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="34dp"
        android:ems="10"
        android:hint="Enter your Email"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toTopOf="@+id/signUpPassword"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView2" />
```

```
35      <EditText
36          android:id="@+id/signUpPassword"
37          android:layout_width="wrap_content"
38          android:layout_height="wrap_content"
39          android:layout_marginBottom="38dp"
40          android:ems="10"
41          android:hint="Enter your password"
42          android:inputType="textPassword"
43          app:layout_constraintBottom_toTopOf="@+id/signUp"
44          app:layout_constraintStart_toStartOf="@+id/signUpPersonName"
45          app:layout_constraintTop_toBottomOf="@+id/signUpPersonName" />
46
47      <Button
48          android:id="@+id/signUp"
49          android:layout_width="wrap_content"
50          android:layout_height="wrap_content"
51          android:layout_marginBottom="39dp"
52          android:onClick="onClickSignUp"
53          android:text="SIGN UP"
54          app:layout_constraintBottom_toTopOf="@+id/suggestTxtView"
55          app:layout_constraintEnd_toEndOf="parent"
56          app:layout_constraintStart_toStartOf="parent"
57          app:layout_constraintTop_toBottomOf="@+id/signUpPassword" />
58
```

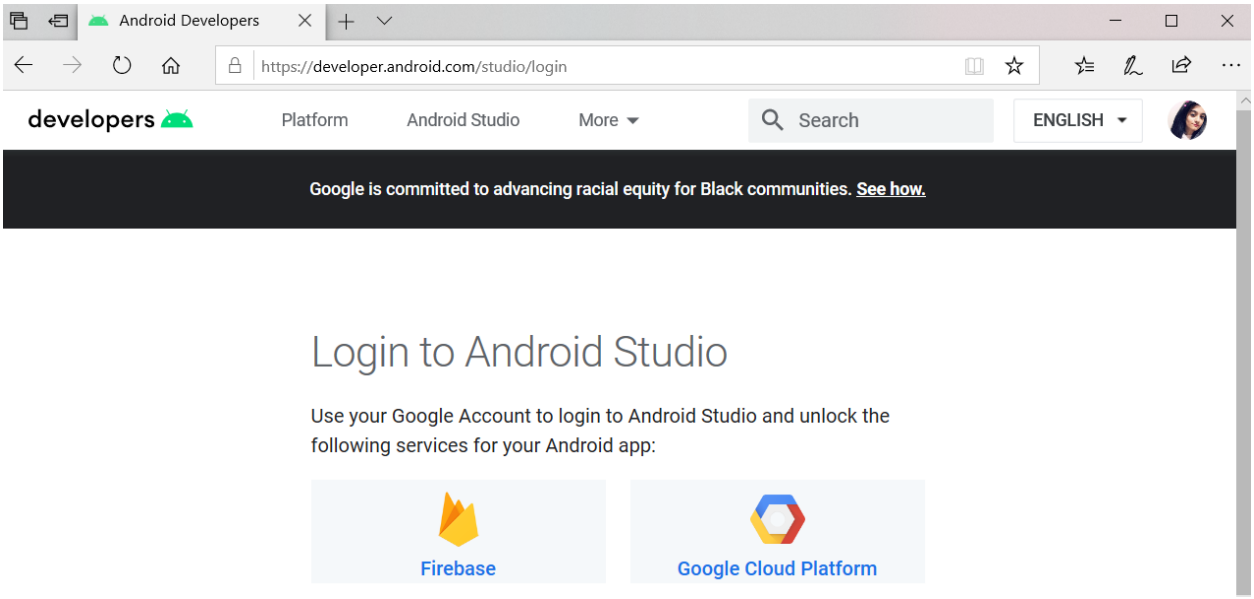
```
59      <TextView
60          android:id="@+id/suggestTxtView"
61          android:layout_width="211dp"
62          android:layout_height="wrap_content"
63          android:layout_marginBottom="276dp"
64          android:gravity="center_horizontal"
65          android:onClick="onClickSignUp"
66          android:text="Already have an account? LOGIN"
67          app:layout_constraintBottom_toBottomOf="parent"
68          app:layout_constraintEnd_toEndOf="parent"
69          app:layout_constraintStart_toStartOf="parent"
70          app:layout_constraintTop_toBottomOf="@+id/signUp" />
71
72  </androidx.constraintlayout.widget.ConstraintLayout>
```

```java
package com.example.shivangi.firebase;

import ...

public class SignUp extends AppCompatActivity {

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

    public void onClickSignUp(View view){
    }
}
```

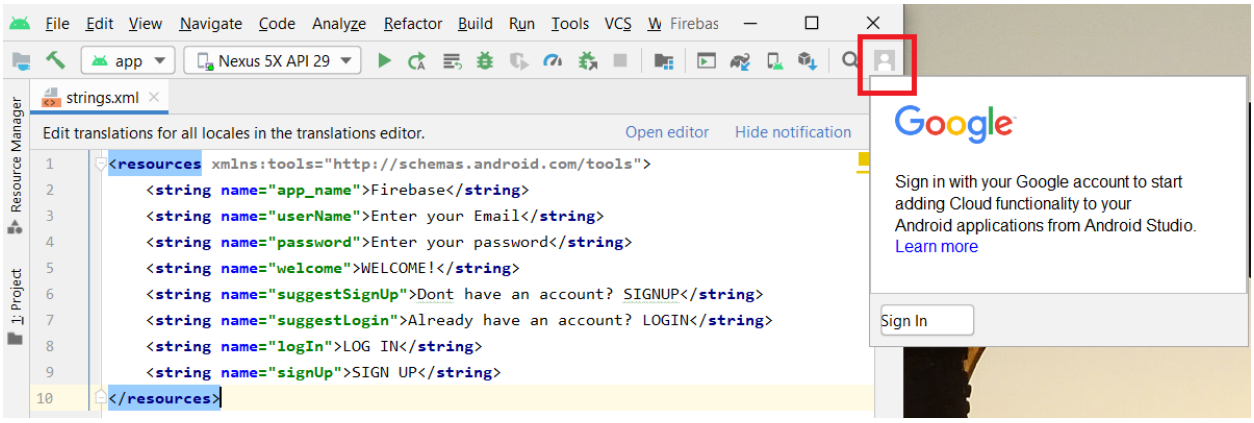
Now let's set up Firebase for our app. To register our app using firebase, go to Firebase console (https://console.firebase.google.com/) and sign in with your google account.



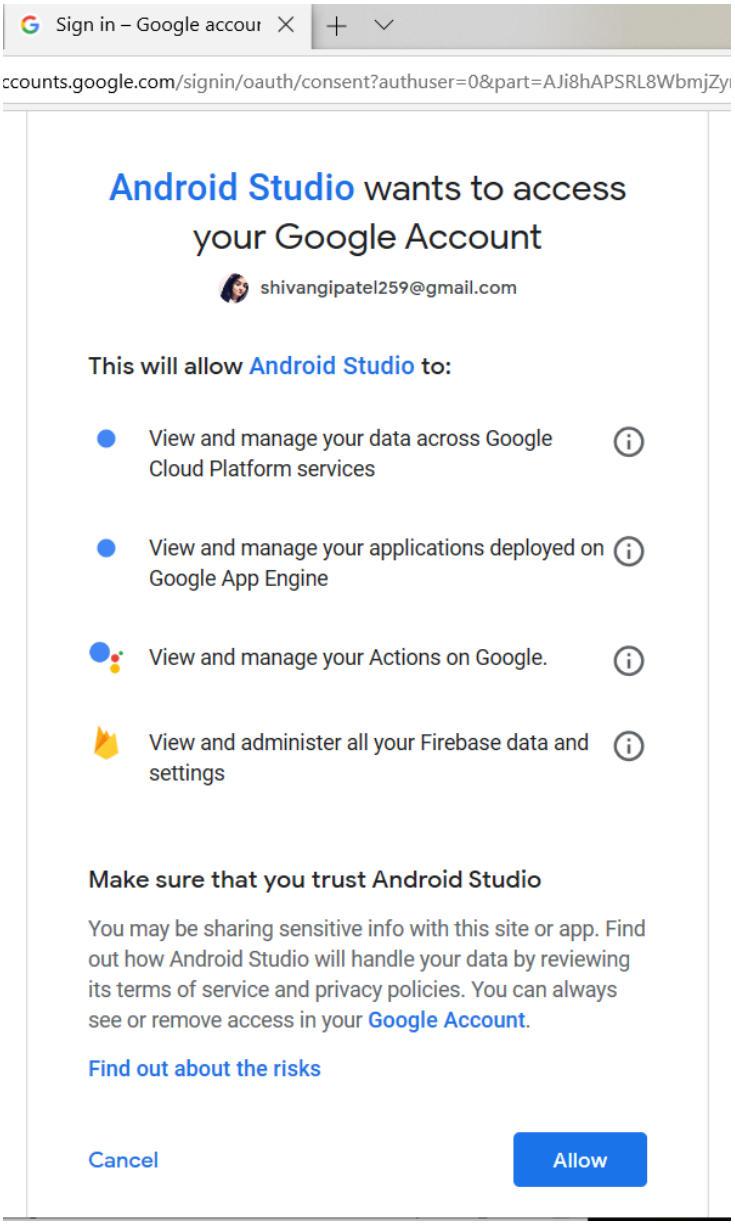
If you haven't signed in with your google account in android studio, the following will appear.



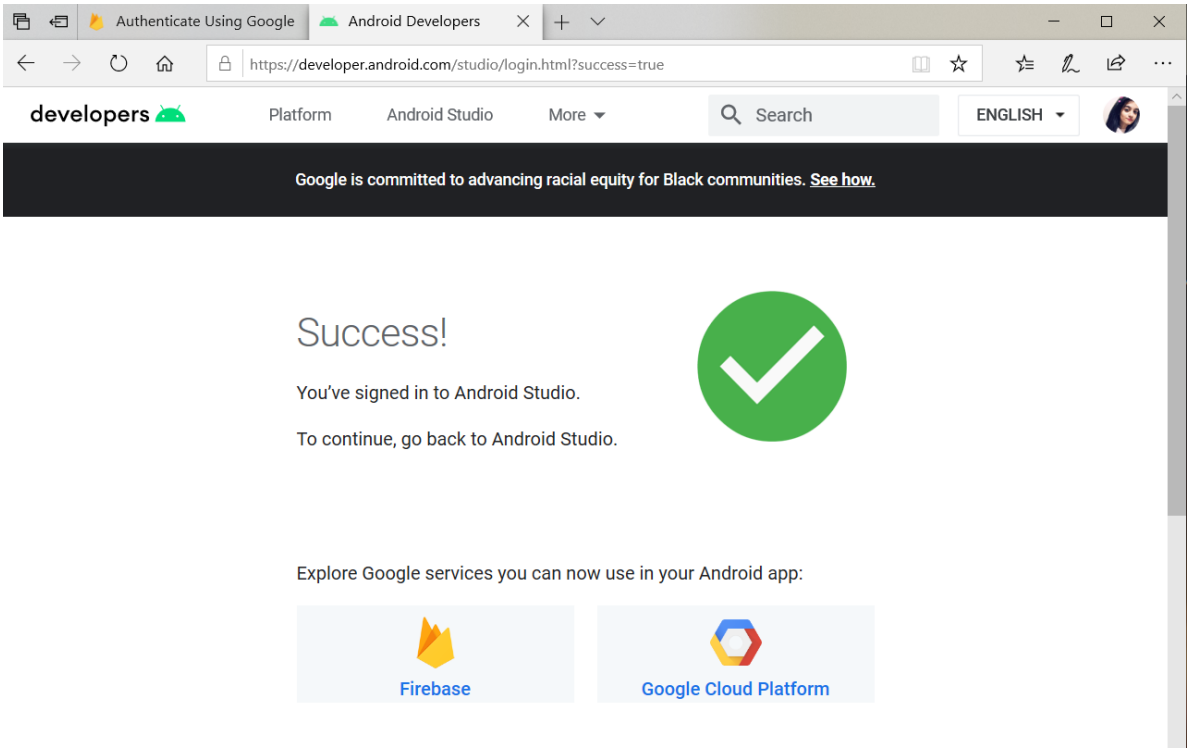
You can login to Android Studio by clicking on the icon at the top right corner.



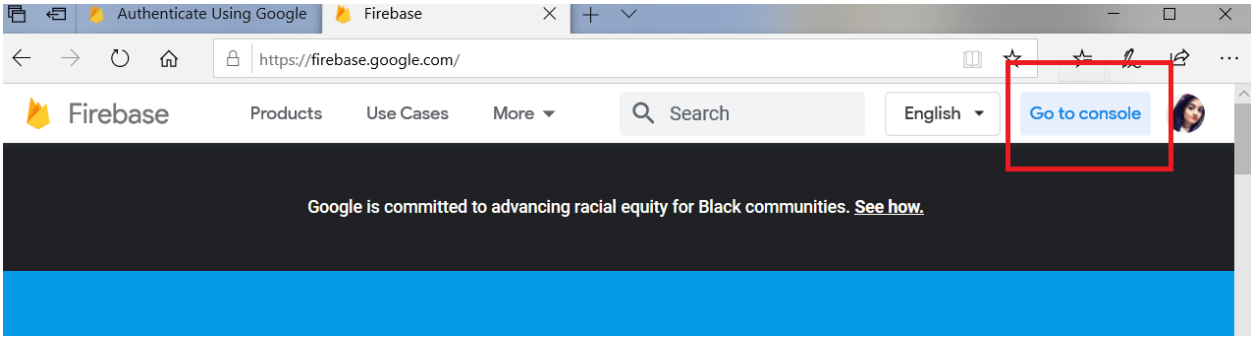
Click Sign In and sign In to your account on the redirected browser and grant access to Android Studio.



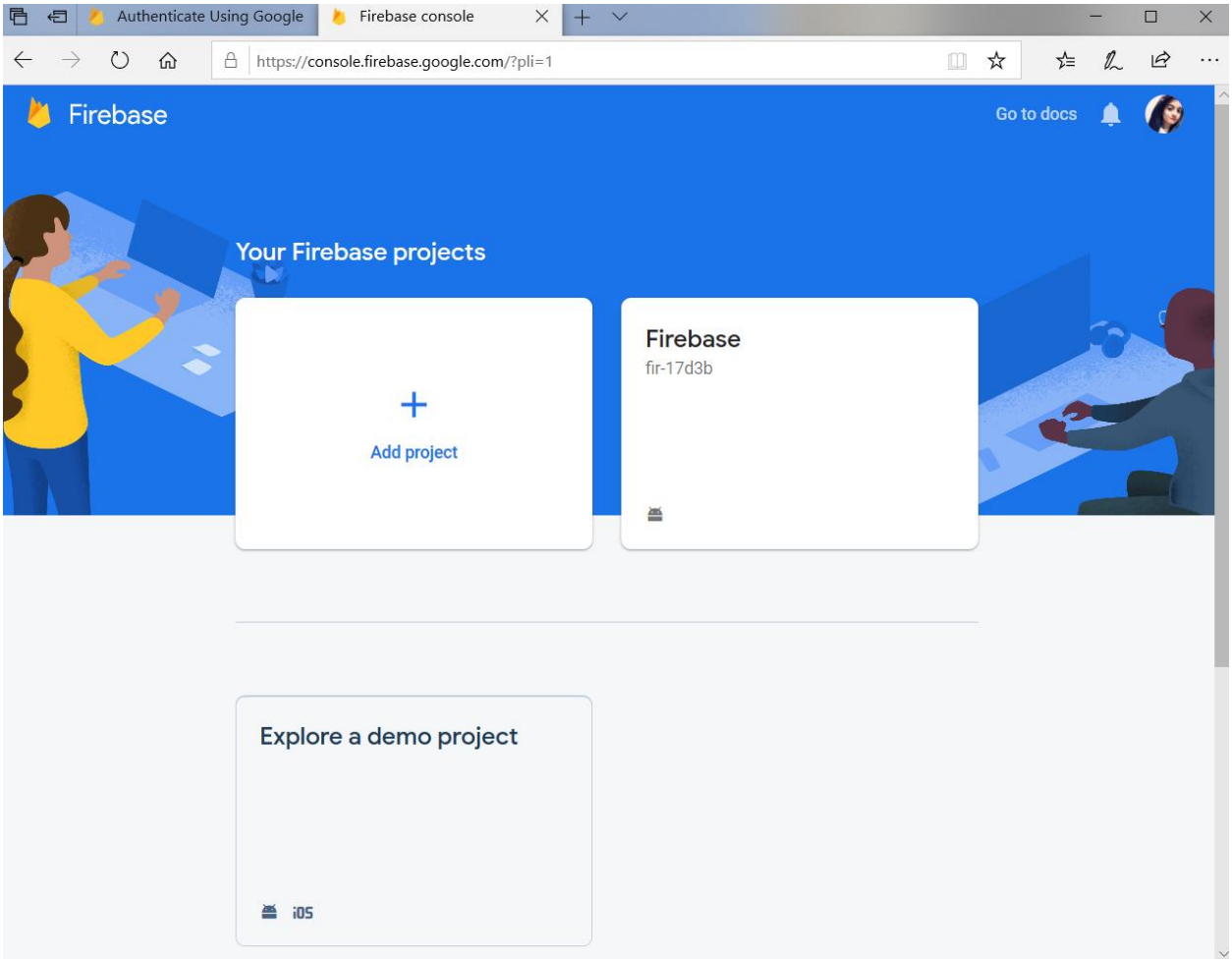
The following page will appear once you successfully sign in.



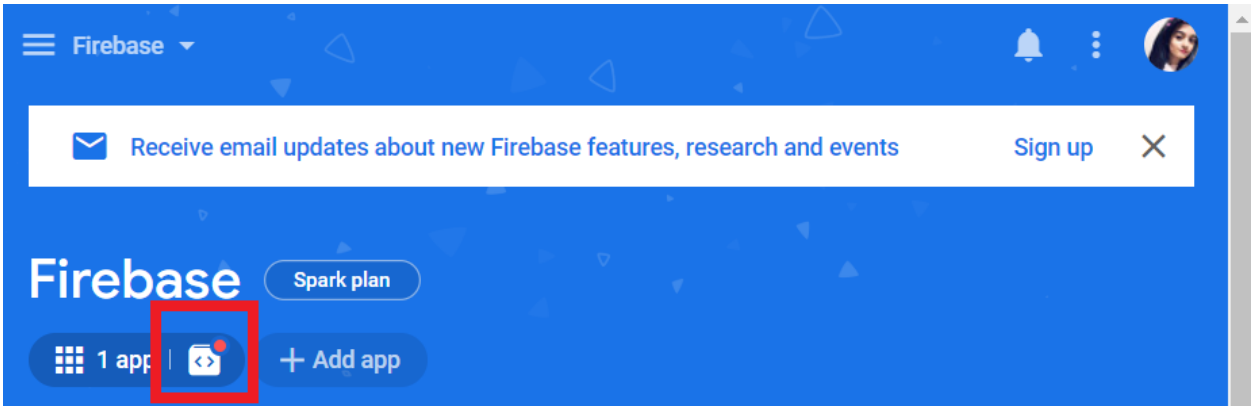
Click on Firebase. In the web page that appears, click on **Go to console**.



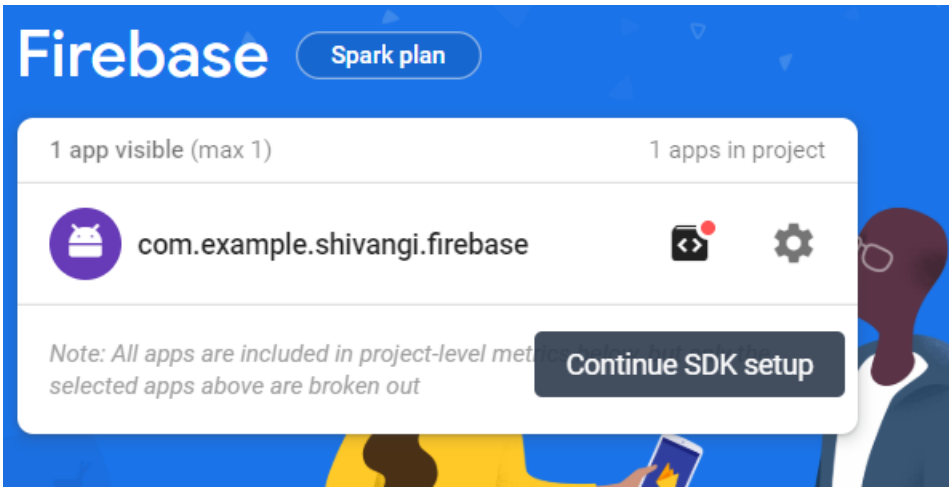
The following page will appear:



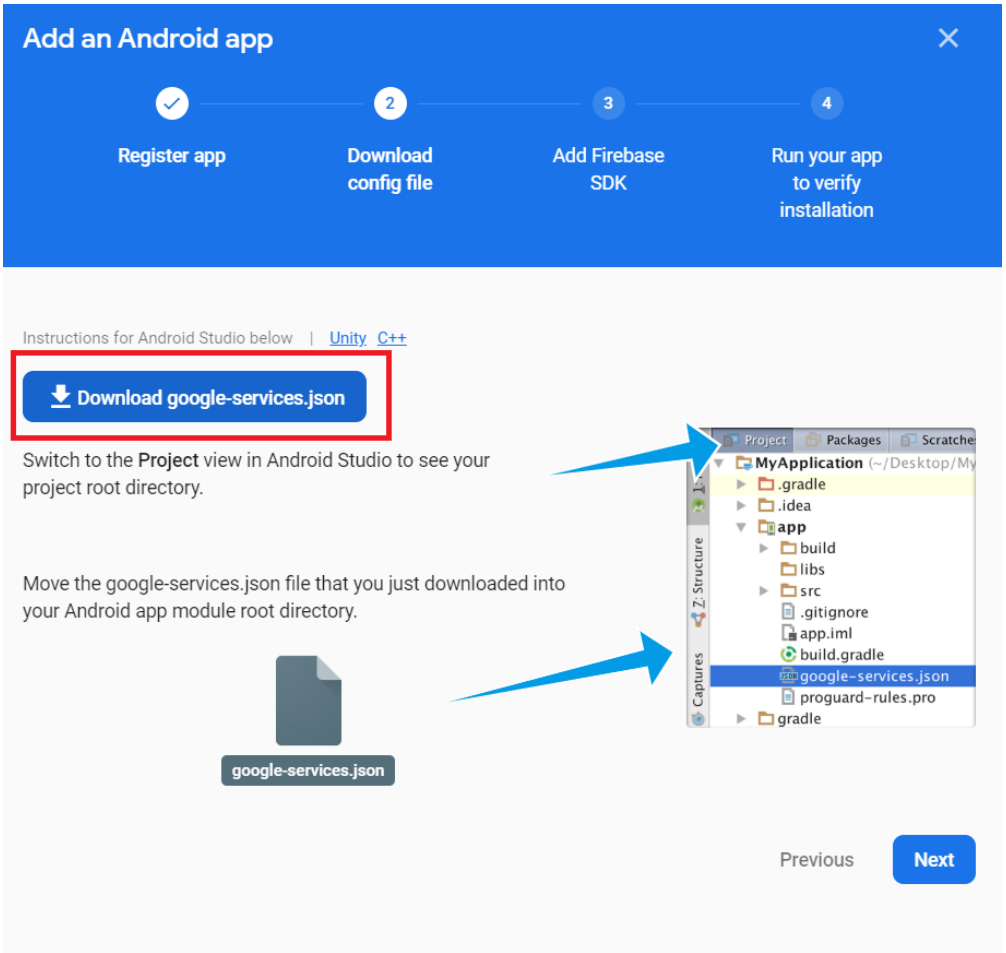
Click on the Firebase project. Then Click on the icon shown below.



In the tab that expands, click the **Continue SDK setup** button.

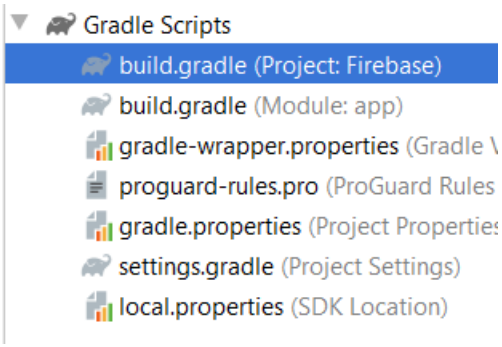


Download the google-service.json file and add it to your app folder under the project mode as shown in the image below. Click next.

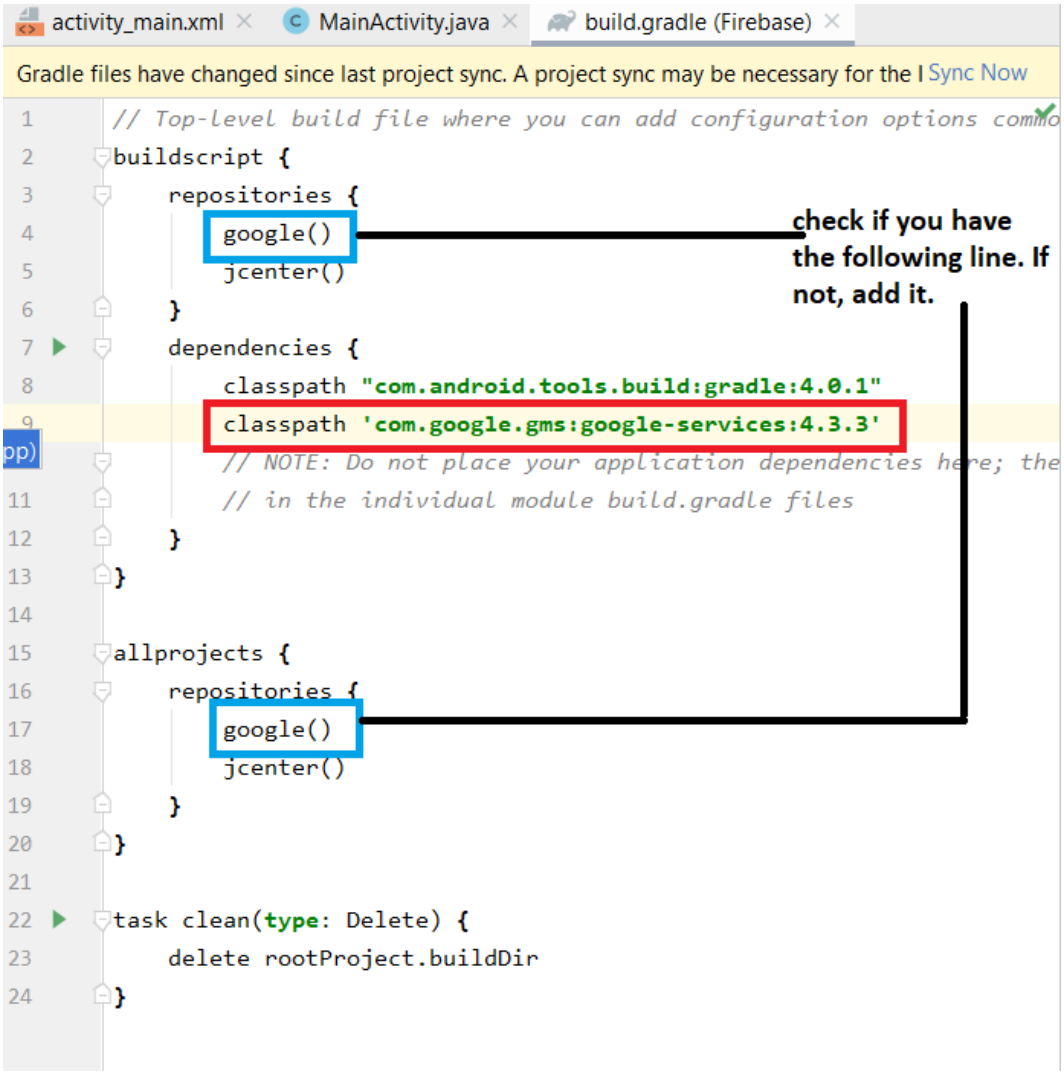


Follow the steps from the web browser to enable the Firebase products in your app.

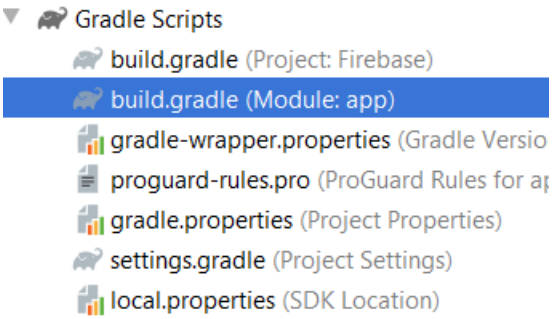
Click on the build.gradle (project: Firebase) file.



Add the classpath in your dependencies and check if you have google() in your repositories.

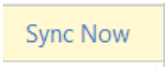


Then go to your build.gradle (Module: app) file and apply the plugins and dependencies :

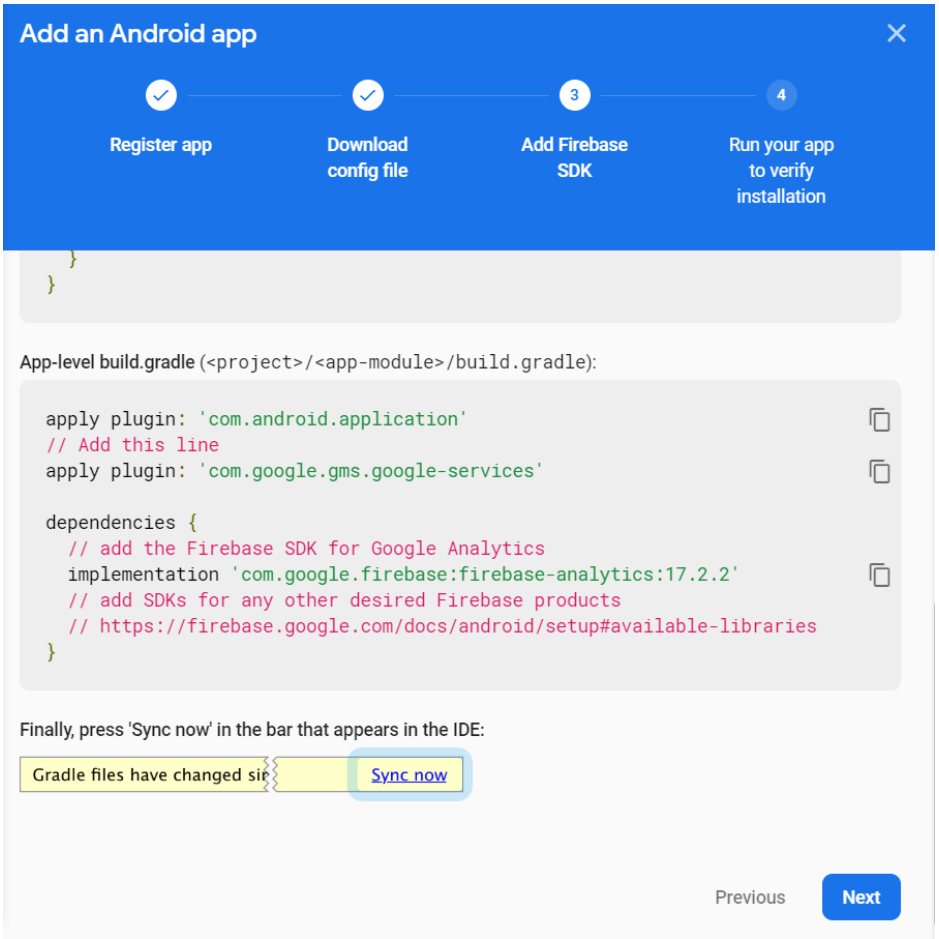



```
25  apply plugin: 'com.android.application'
26  apply plugin: 'com.google.gms.google-services'
27
28  dependencies {
29      implementation fileTree(dir: "libs", include: ["*.jar"])
30      implementation 'androidx.appcompat:appcompat:1.1.0'
31      implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
32      implementation 'com.google.firebase:firebase-auth:19.3.2'
33      testImplementation 'junit:junit:4.12'
34      androidTestImplementation 'androidx.test.ext:junit:1.1.1'
35      androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'
36      implementation 'com.google.firebase:firebase-analytics:17.2.2'
37
38  }
```

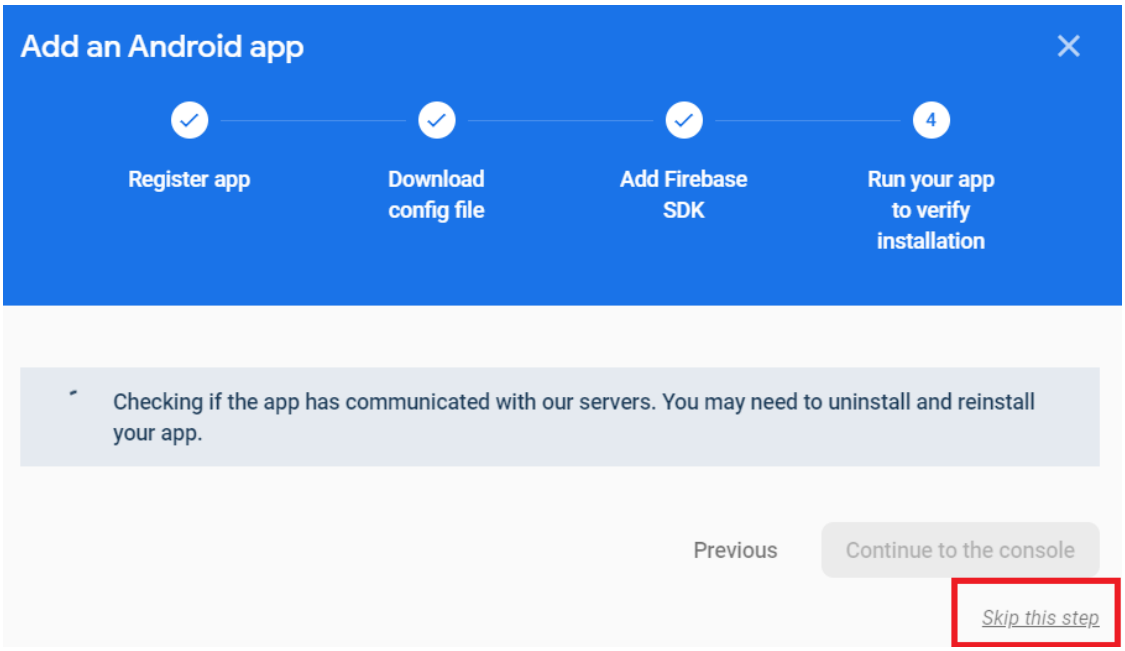
Once done, Sync the file by clicking on the SYNC NOW at the top right corner.



On the web browser, click next.

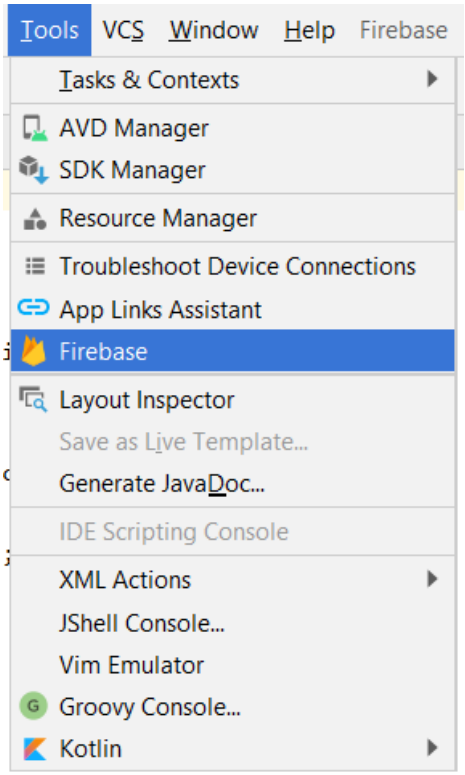


In the last step, you can skip the step.

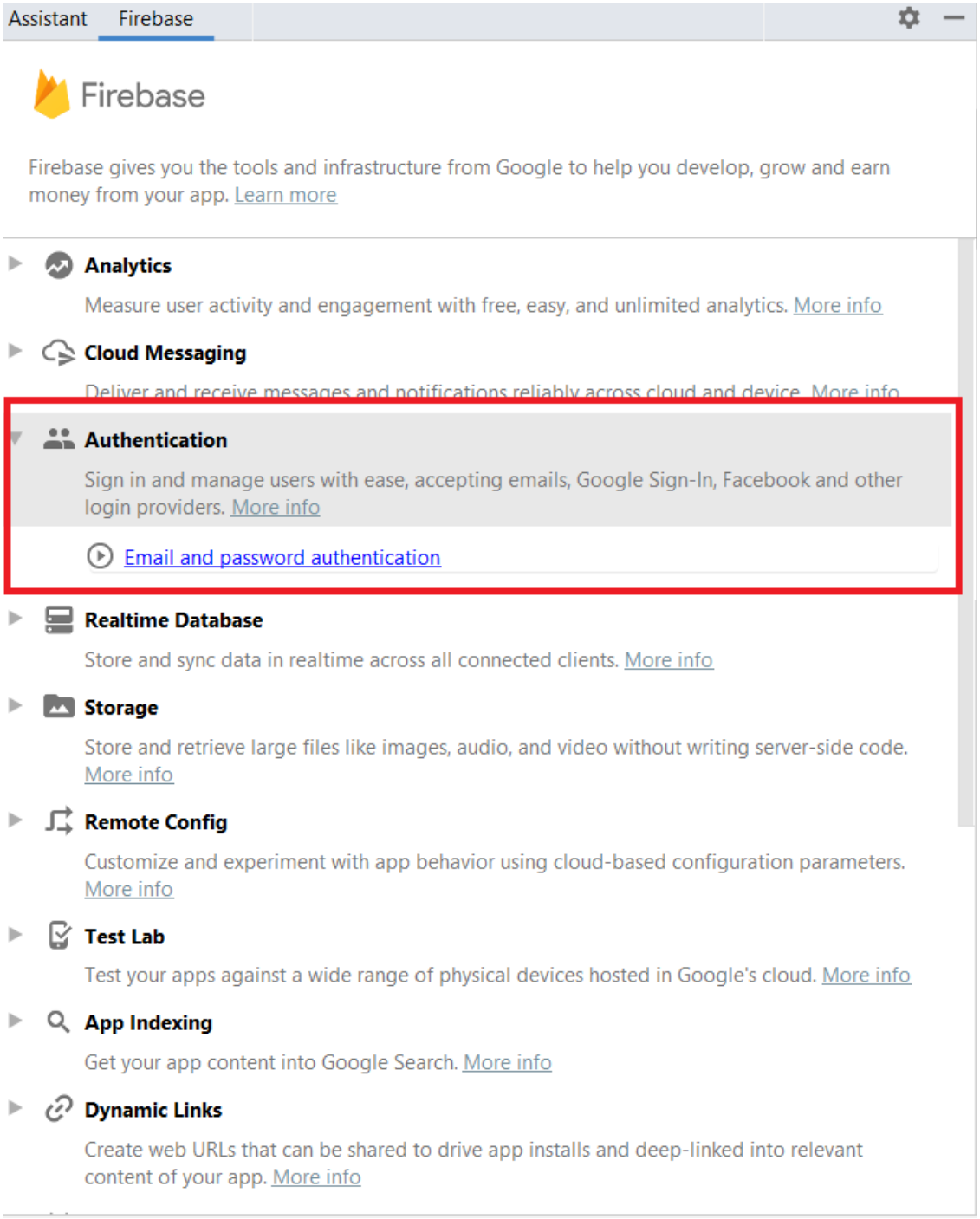


You can also do the following to connect and add Firebase dependencies to your app.

Click on Tools and select Firebase



An assistant panel will open on the right. This includes all the services provided by firebase. Click on Authentication and then on the Email and Password authentication link.



You can now easily connect to the Firebase and add the authentication to your app. Below step 2, the assistant provides you will codes that will help you create a user and log in operations.

Assistant Firebase

← Firebase › Authentication

You can use Firebase Authentication to let your users sign in with their email addresses and passwords, an
manage your app's password-based accounts. This tutorial helps you set up an email and password systen
access information about the user.

Launch in browser

1 Connect your app to Firebase

Connect to Firebase

2 Add Firebase Authentication to your app

Add Firebase Authentication to your app

To use an authentication provider, you need to enable it in the Firebase console . Go to the Sign-in M
page in the Firebase Authentication section to enable Email/Password sign-in and any other identity
you want for your app.

3 Check current auth state

Declare an instance of FirebaseAuth

```
private FirebaseAuth mAuth;
```

In the onCreate() method, initialize the FirebaseAuth instance.

```
// Initialize Firebase Auth
mAuth = FirebaseAuth.getInstance();
```

When initializing your Activity, check to see if the user is currently signed in.

```
@Override
public void onStart() {
```

Now we will declare member variables for our EditTexts and FirebaseAuth. We will initialize these variables in our onCreate method and get the Firebase instance.

```
public class MainActivity extends AppCompatActivity {

    EditText loginEmail, loginPassword;
    private FirebaseAuth mAuth;

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

        loginEmail = findViewById(R.id.logInPersonName);
        loginPassword = findViewById(R.id.logInPassword);

        mAuth = FirebaseAuth.getInstance();
    }
```

Add the following code to the onClick method to perform the required operations when clicked.

The TextView will redirect us to the SignUp class.

```
    public void onClick(View view) {
        switch (view.getId()){
            case R.id.logIn:
                break;
            case R.id.textViewSuggestSignUp:
                startActivity(new Intent( packageContext: this,SignUp.class));
                break;
        }
    }
```

Let’s first create a user. Go to your SignUp class and declare EditTexts and FirebaseAuth variables. Initialize these variables in your onCreate method.

```
public class SignUp extends AppCompatActivity {

    EditText userEmail, userPassword;
    private FirebaseAuth mAuth;

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

        userEmail = findViewById(R.id.signUpPersonName);
        userPassword = findViewById(R.id.signUpPassword);

        mAuth = FirebaseAuth.getInstance();
    }
```

Now we will create a register() method below the onCreate method. This method will get the input from the EditTexts and validate it. It will check if the values from the EditTexts are not null, a valid email pattern is added, and the minimum length of the password is 6.

Android provides us with a Pattern class that has a matcher method that matches character sequences against regular expression. We don't have to create a pattern for email as the pattern for email address is already provided.

A toast message is displayed to inform the user if the validations are not met.

```
45      private void register() {
46          String email = userEmail.getText().toString();
47          String password = userPassword.getText().toString();
48
49          if(email.isEmpty()){
50              Toast.makeText( context: SignUp.this, text: "Please enter an email!",Toast.LENGTH_SHORT).show();
51              return;
52          }
53
54          if(!Patterns.EMAIL_ADDRESS.matcher(email).matches()){
55              Toast.makeText( context: SignUp.this, text: "Please enter a valid email!",Toast.LENGTH_SHORT).show();
56              return;
57          }
58
59          if(password.isEmpty()){
60              Toast.makeText( context: SignUp.this, text: "Please enter a password!",Toast.LENGTH_SHORT).show();
61              return;
62          }
63
64          if(password.length() < 6){
65              Toast.makeText( context: SignUp.this, text: "The Minimum length of a password is 6!",Toast.LENGTH_SHORT).show();
66              return;
67          }
68      }
```

Firebase provides a createUserWithEmailAndPassword() method that takes the email and password as its parameter. An addOnCompleteListener() method is used to perform required actions depending on the situation. If an error occurred, then we can inform the user of the error.

Let's call the createUserWithEmailAndPassword() method by using the Firebase instance we initialized.

```
69          mAuth.createUserWithEmailAndPassword(email,password)
70                  .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
71              @Override
72              public void onComplete(@NonNull Task<AuthResult> task) {
73                  |
74              }
75          });
```

Add the following code. The onComplete method will determine if the task was successful. If it was, then it will inform the user on its success and clear the fields. If the email the user is trying to register is already in use, it will notify the user about it.

```
69          mAuth.createUserWithEmailAndPassword(email,password)
70                  .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
71              @Override
72              public void onComplete(@NonNull Task<AuthResult> task) {
73                  if(task.isSuccessful()){
74                      Toast.makeText( context: SignUp.this, text: "User registered successfully!",Toast.LENGTH_LONG).show();
75                      userEmail.setText("");
76                      userPassword.setText("");
77                  }
78                  else {
79                      if(task.getException() instanceof FirebaseAuthUserCollisionException){
80                          Toast.makeText( context: SignUp.this, text: "You are already registered!",Toast.LENGTH_SHORT).show();
81                      }
82                      else {
83                          Toast.makeText( context: SignUp.this, text: "Registration failed!",Toast.LENGTH_SHORT).show();
84                      }
85                  }
86              }
87          });
```

We have completed the registration method. Now we have to call this method when the user clicks the button.

In your onClickSignUp() method, we will add a switch method which will determine which component is calling it and perform the tasks accordingly. The signUp button will call the register method while the TextView will redirect user to the MainActivity.

Add the following code:

```
35 @    public void onClickSignUp(View view){
36          switch (view.getId()){
37              case R.id.signUp:
38                  register();
39                  break;
40              case R.id.suggestTxtView:
41                  startActivity(new Intent( packageContext: this,MainActivity.class));
42          }
43      }
44
```

Your final SignUp class will look as follows:

SignUp.java

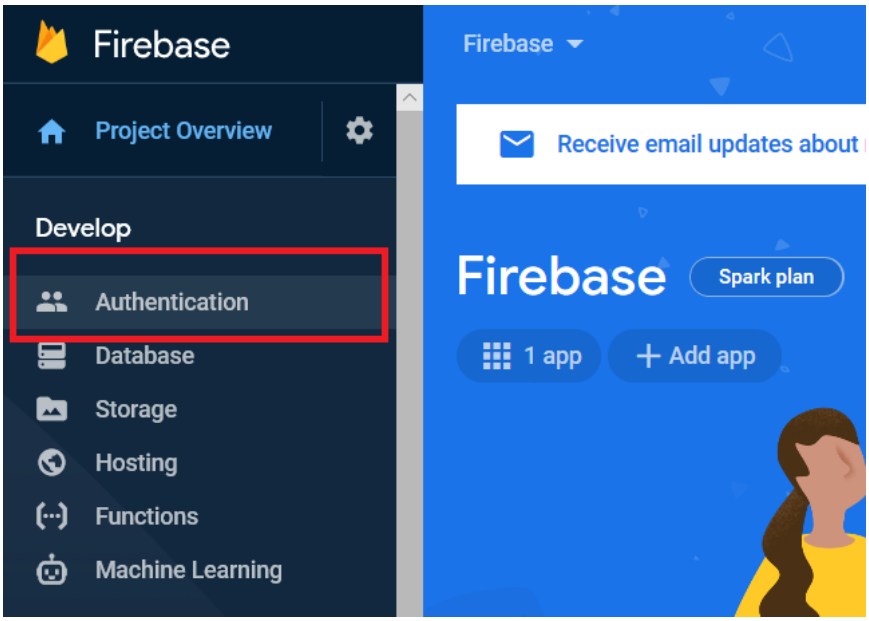
```java
package com.example.shivangi.firebase;

import ...

public class SignUp extends AppCompatActivity {

    EditText userEmail, userPassword;
    private FirebaseAuth mAuth;

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

        userEmail = findViewById(R.id.signUpPersonName);
        userPassword = findViewById(R.id.signUpPassword);

        mAuth = FirebaseAuth.getInstance();
    }

    public void onClickSignUp(View view){
        switch (view.getId()){
            case R.id.signUp:
                register();
                break;
            case R.id.suggestTxtView:
                startActivity(new Intent( packageContext: this,MainActivity.class));
        }
    }

    private void register() {
        String email = userEmail.getText().toString();
        String password = userPassword.getText().toString();
```

SignUp.java

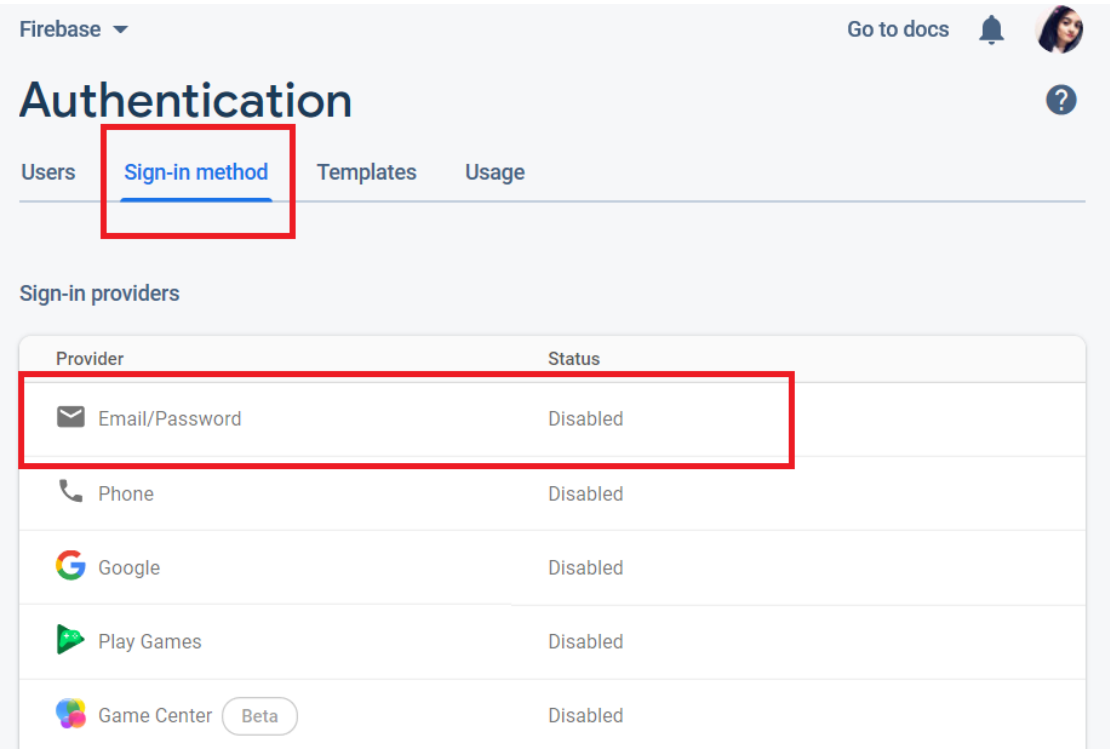
```java
    private void register() {
        String email = userEmail.getText().toString();
        String password = userPassword.getText().toString();

        if(email.isEmpty()){
            Toast.makeText( context: SignUp.this, text: "Please enter an email!",Toast.LENGTH_SHORT).show();
            return;
        }

        if(!Patterns.EMAIL_ADDRESS.matcher(email).matches()){
            Toast.makeText( context: SignUp.this, text: "Please enter a valid email!",Toast.LENGTH_SHORT).show();
            return;
        }

        if(password.isEmpty()){
            Toast.makeText( context: SignUp.this, text: "Please enter a password!",Toast.LENGTH_SHORT).show();
            return;
        }

        if(password.length() < 6){
            Toast.makeText( context: SignUp.this, text: "The Minimum length of a password is 6!",Toast.LENGTH_SHORT).show();
            return;
        }
```

```java
        mAuth.createUserWithEmailAndPassword(email,password)
                .addOnCompleteListener( activity: this, (task) -> {
                if(task.isSuccessful()){
                    Toast.makeText( context: SignUp.this, text: "User registered successfully!",Toast.LENGTH_LONG).show();
                    userEmail.setText("");
                    userPassword.setText("");
                }
                else {
                    if(task.getException() instanceof FirebaseAuthUserCollisionException){
                        Toast.makeText( context: SignUp.this, text: "You are already registered!",Toast.LENGTH_SHORT).show();
                    }
                    else {
                        Toast.makeText( context: SignUp.this, text: "Registration failed!",Toast.LENGTH_SHORT).show();
                    }
                }
        });
    }
}
```

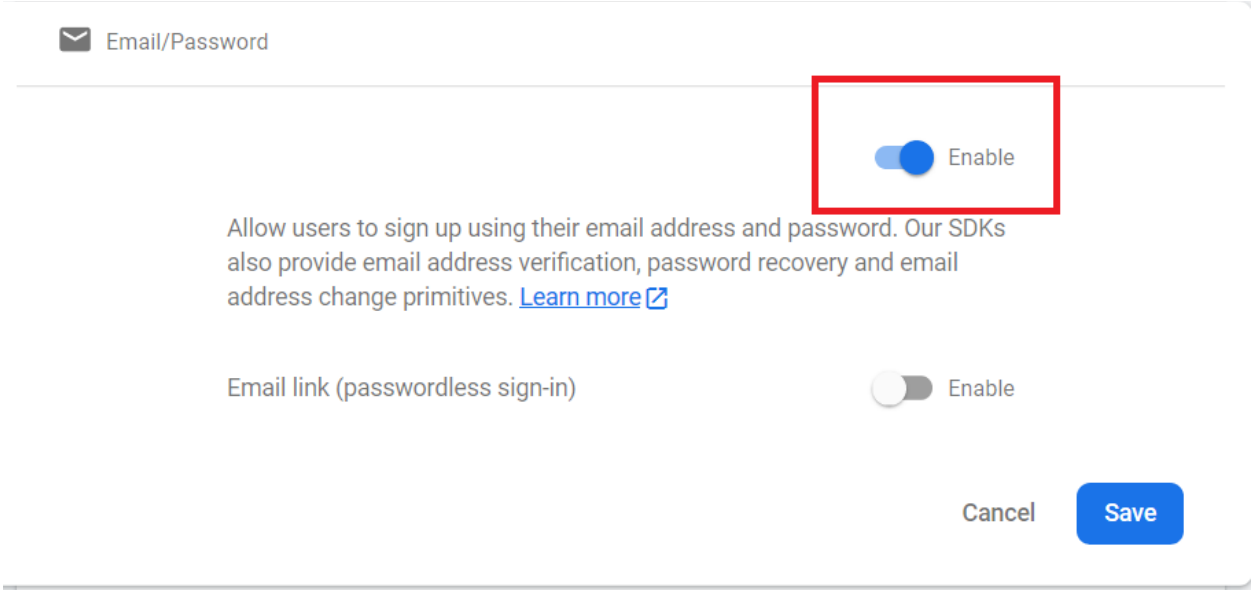
Before we run our app, we have to enable email and password sign in from our Firebase console. Go to your firebase console and click on the Authentication on the left navigation pane.

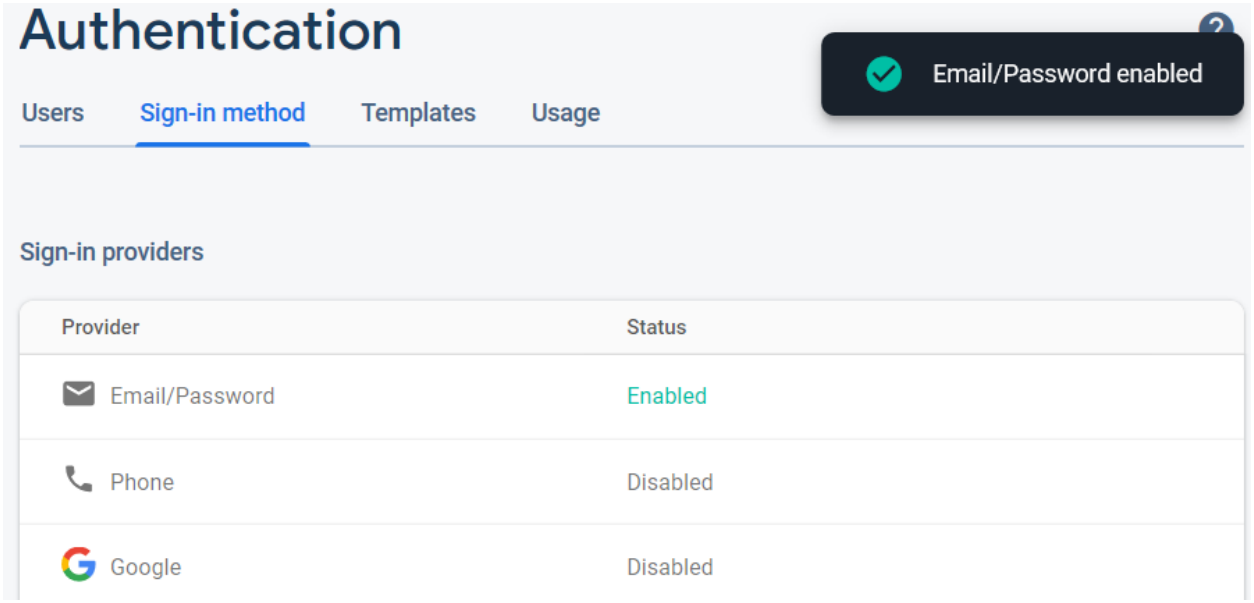


Click on the Sign-in method and then Email/Password option under Sign-in providers. As you can see it is currently disabled. When you click on it, a dialog box will appear.

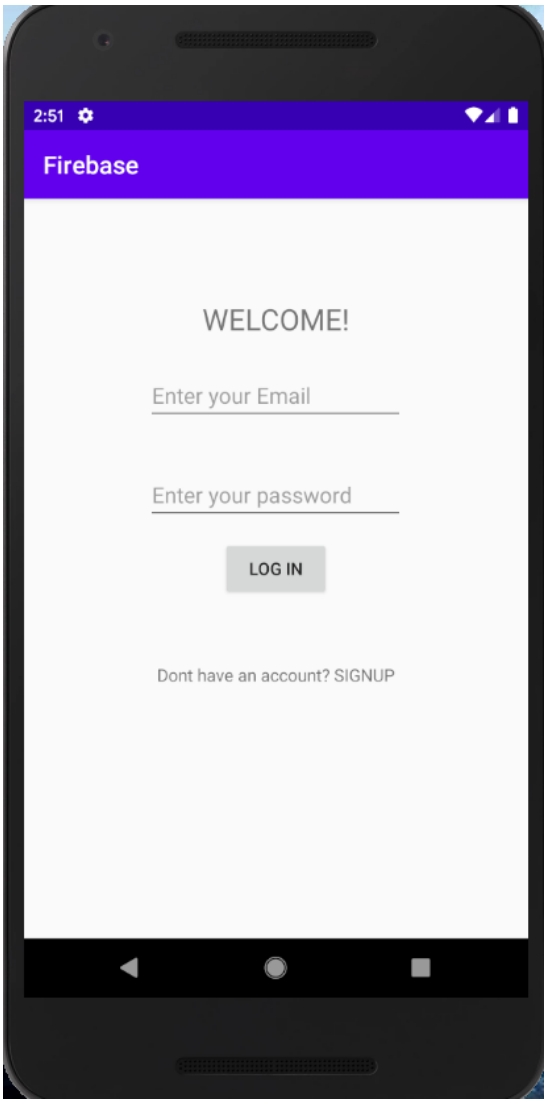


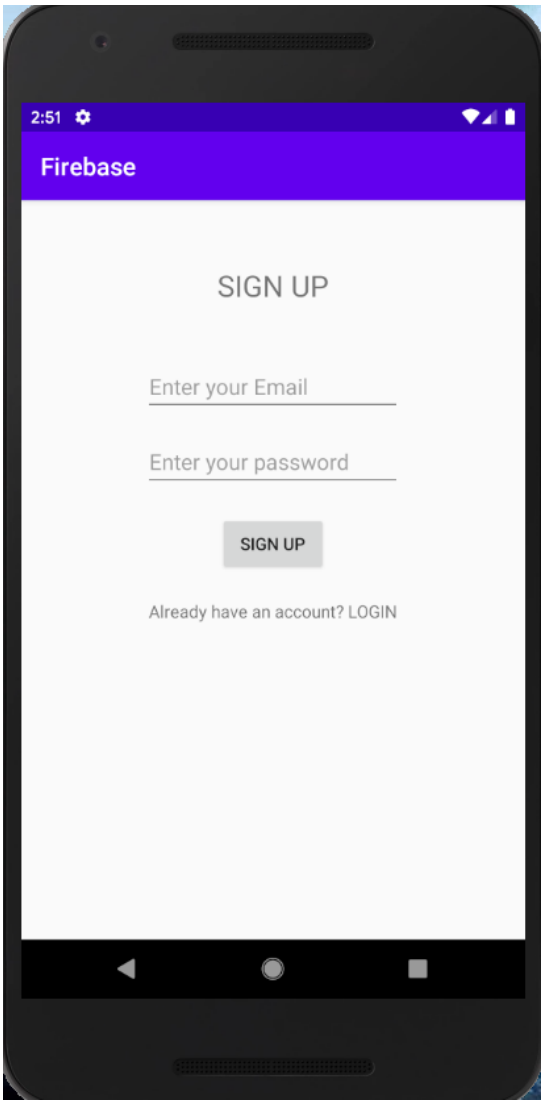
Enable the Email/ Password and click save.

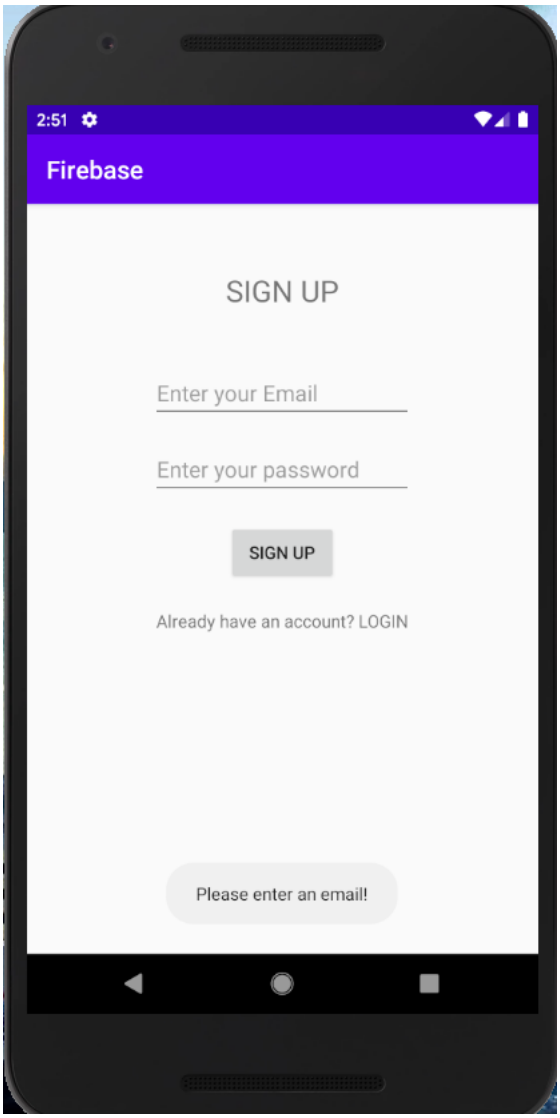


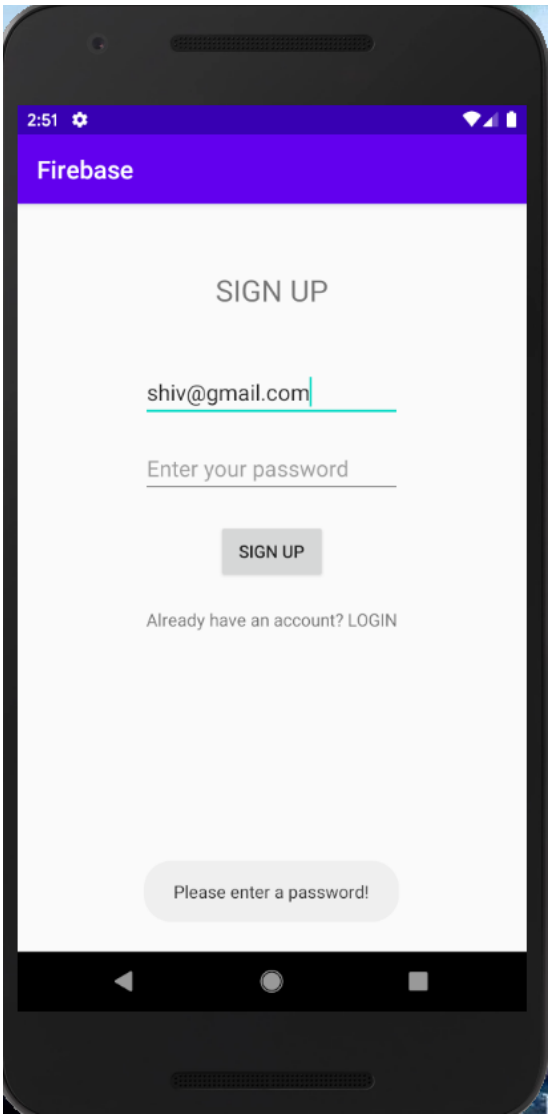


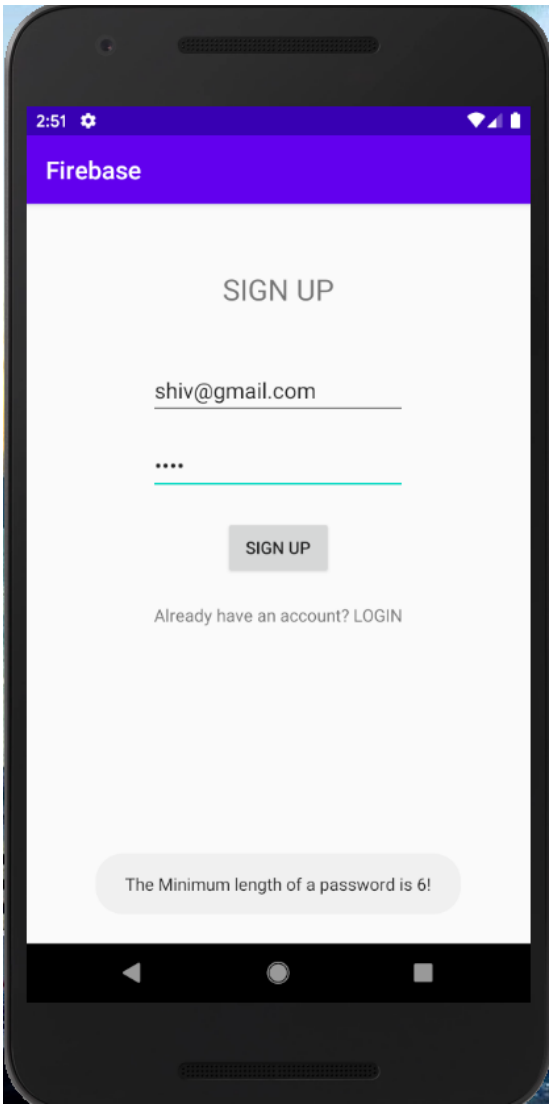
Run the app.

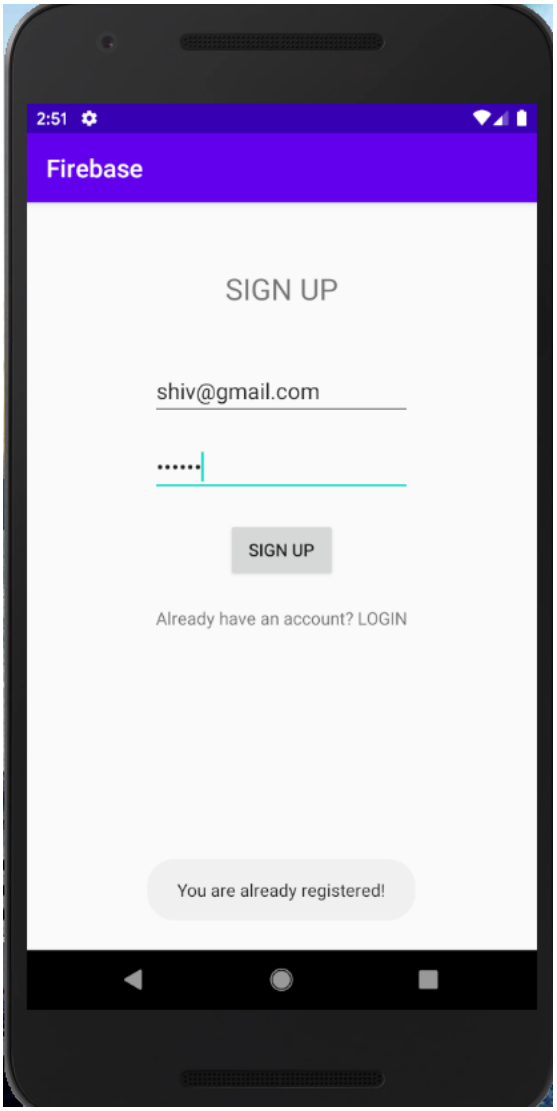

2:51
Firebase
WELCOME!
Enter your Email
Enter your password
LOG IN
Dont have an account? SIGNUP

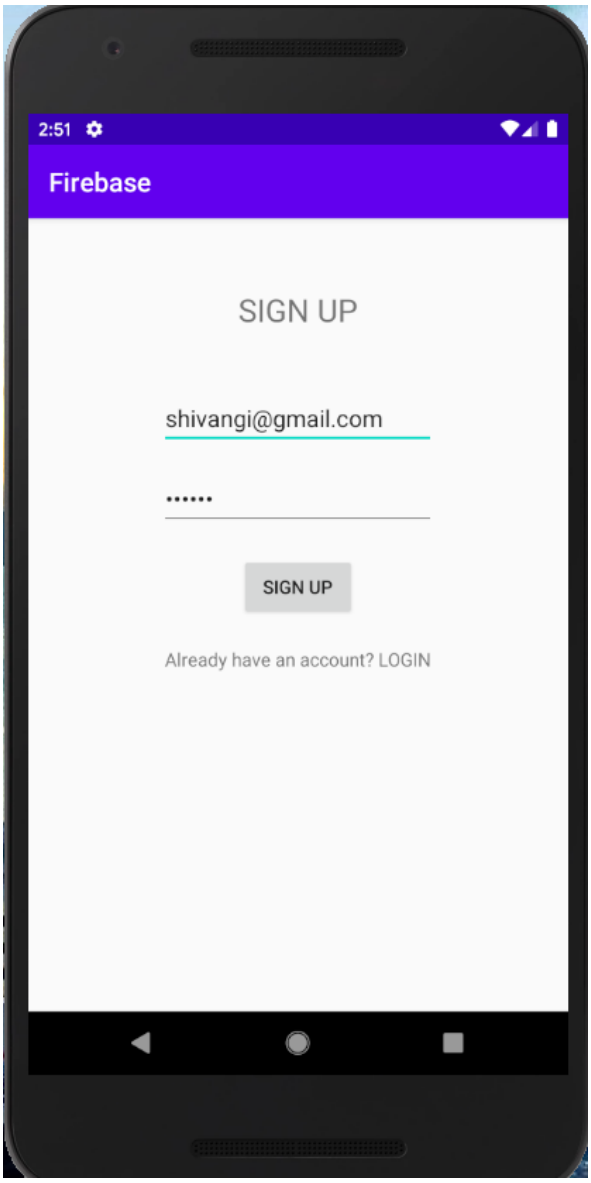

2:51
Firebase
SIGN UP
Enter your Email
Enter your password
SIGN UP
Already have an account? LOGIN

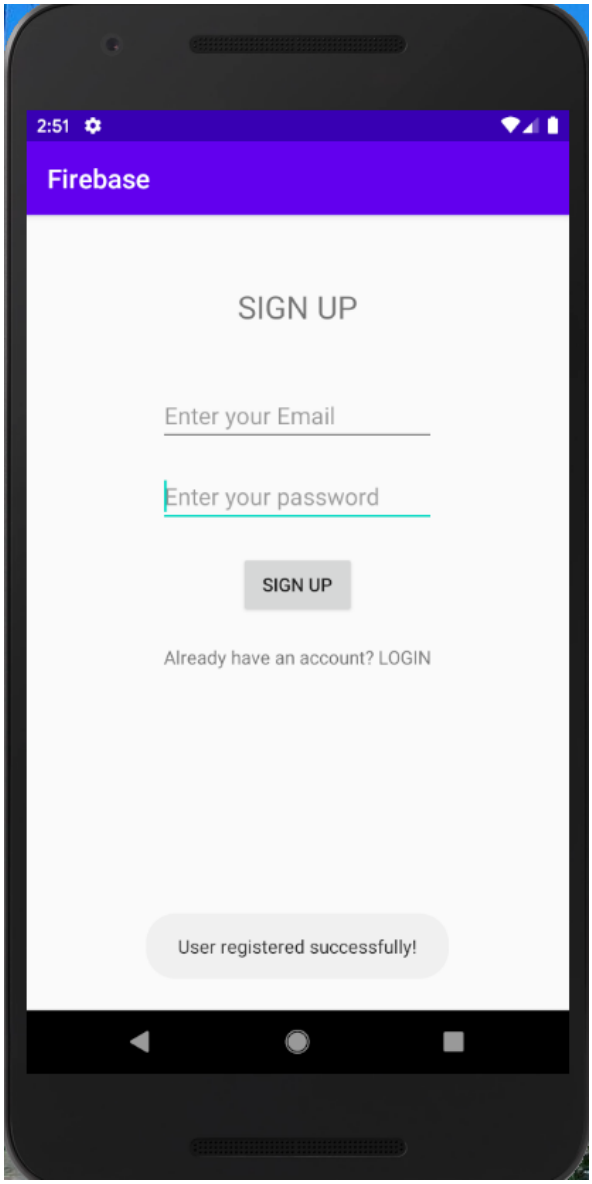

2:51
Firebase
SIGN UP
Enter your Email
Enter your password
SIGN UP
Already have an account? LOGIN
Please enter an email!


2:51
Firebase
SIGN UP
shiv@gmail.com
Enter your password
SIGN UP
Already have an account? LOGIN
Please enter a password!


2:51
Firebase
SIGN UP
shiv@gmail.com
SIGN UP
Already have an account? LOGIN
The Minimum length of a password is 6!


2:51
Firebase
SIGN UP
shiv@gmail.com
SIGN UP
Already have an account? LOGIN
You are already registered!


2:51
Firebase
SIGN UP
shivangi@gmail.com
SIGN UP
Already have an account? LOGIN


2:51
Firebase
SIGN UP
Enter your Email
Enter your password
SIGN UP
Already have an account? LOGIN
User registered successfully!

Now let's create operations for login in our MainActivity. Create a new method under the onCreate method called logIn. In this method we will get the input from the EditTexts and validate it. It will check if the values from the EditTexts are not null, a valid email pattern is added, and the minimum length of the password is 6.

You can simply copy the validation code from the register method we created in our SignUp class and change the editText variable to the one declared in your MainActivity and the SignUp.this to MainActivity.this

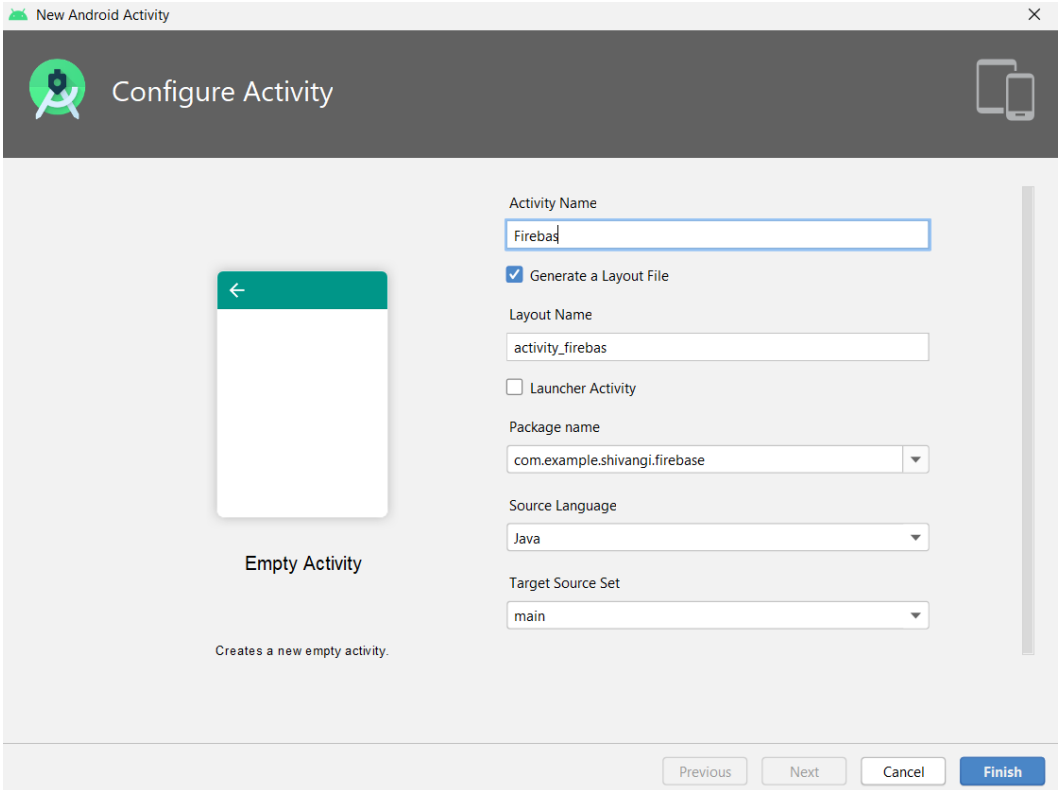
```
34      public void logIn() {
35          String email = loginEmail.getText().toString();
36          String password = loginPassword.getText().toString();
37
38          if(email.isEmpty()){
39              Toast.makeText( context: MainActivity.this, text: "Please enter an email!",Toast.LENGTH_SHORT).show();
40              return;
41          }
42
43          if(!Patterns.EMAIL_ADDRESS.matcher(email).matches()){
44              Toast.makeText( context: MainActivity.this, text: "Please enter a valid email!",Toast.LENGTH_SHORT).show();
45              return;
46          }
47
48          if(password.isEmpty()){
49              Toast.makeText( context: MainActivity.this, text: "Please enter a password!",Toast.LENGTH_SHORT).show();
50              return;
51          }
52
53          if(password.length() < 6){
54              Toast.makeText( context: MainActivity.this, text: "The Minimum length of a password is 6!",Toast.LENGTH_SHORT).show();
55              return;
56          }
57      }
```

Firebase provides a signInWithEmailAndPassword() method and it takes an email and a password as its parameters. Add the following code after your validations.

```
58          mAuth.signInWithEmailAndPassword(email,password)
59                  .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
60              @Override
61              public void onComplete(@NonNull Task<AuthResult> task) {
62
63              }
64          });
```

We want to open a new activity if the logIn is successful or display a toast if the something went wrong.

Create a new Empty activity and name it Firebase.



MainActivity.java Firebase.java

```java
package com.example.shivangi.firebase;

import ...

public class Firebase extends AppCompatActivity {

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

Now let's go back to our MainActivity and add the following code in the signInWithEmailAndPassword() method.

```
58          mAuth.signInWithEmailAndPassword(email,password)
59                  .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
60              @Override
61              public void onComplete(@NonNull Task<AuthResult> task) {
62                  if(task.isSuccessful()){
63                      startActivity(new Intent( packageContext: MainActivity.this, Firebase.class));
64                  } else{
65                      Toast.makeText( context: MainActivity.this, text: "Please enter a correct email and password!",Toast.LENGTH_SHORT).show();
66                  }
67              }
68          });
```

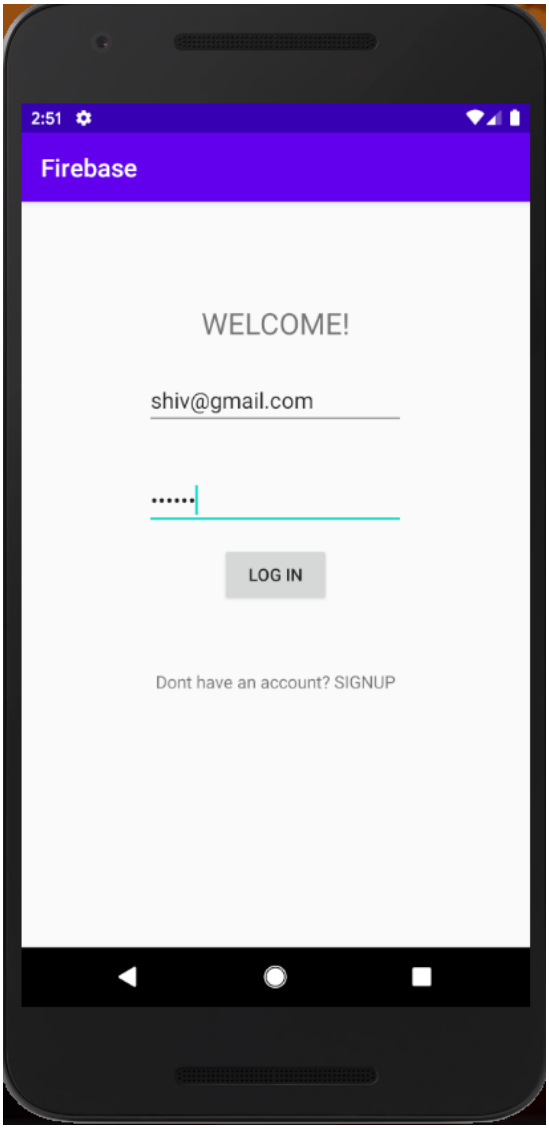
Your final MainActivity class looks as follows:

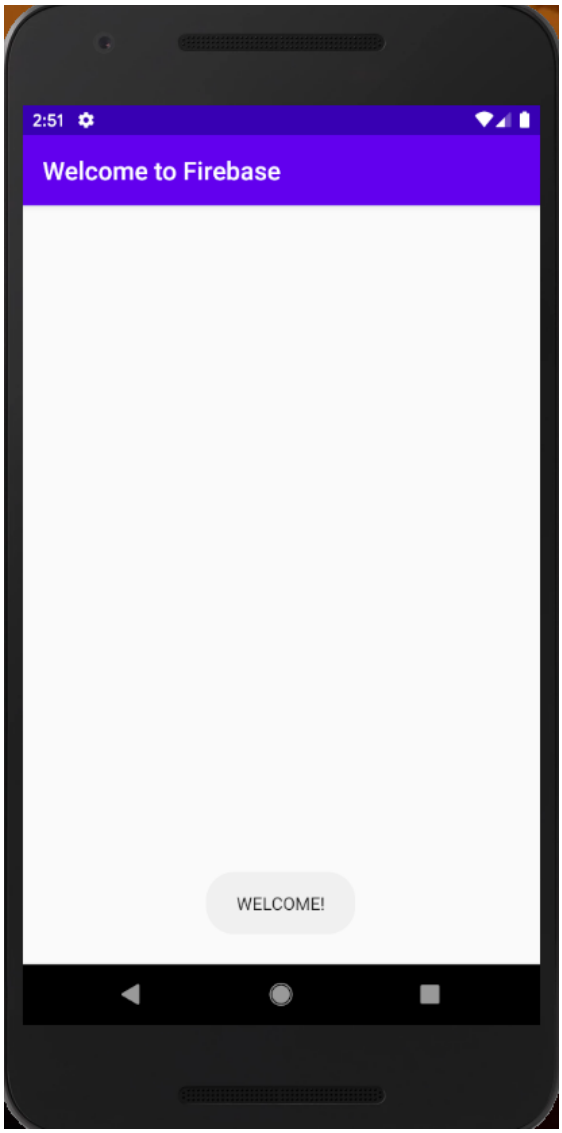
MainActivity.java

```
1   package com.example.shivangi.firebase;
2
3   import ...
37
38  public class MainActivity extends AppCompatActivity {
39
40      EditText loginEmail, loginPassword;
41      private FirebaseAuth mAuth;
42
43      @Override
44      protected void onCreate(Bundle savedInstanceState) {
45          super.onCreate(savedInstanceState);
46          setContentView(R.layout.activity_main);
47
48          loginEmail = findViewById(R.id.logInPersonName);
49          loginPassword = findViewById(R.id.logInPassword);
50
51          mAuth = FirebaseAuth.getInstance();
52      }
53
54      public void logIn() {
55          String email = loginEmail.getText().toString();
56          String password = loginPassword.getText().toString();
57
58          if(email.isEmpty()){
59              Toast.makeText( context: MainActivity.this, text: "Please enter an email!",Toast.LENGTH_SHORT).show();
60              return;
61          }
62
63          if(!Patterns.EMAIL_ADDRESS.matcher(email).matches()){
64              Toast.makeText( context: MainActivity.this, text: "Please enter a valid email!",Toast.LENGTH_SHORT).show();
65              return;
66          }
```

```
68          if(password.isEmpty()){
69              Toast.makeText( context: MainActivity.this, text: "Please enter a password!",Toast.LENGTH_SHORT).show();
70              return;
71          }
72
73          if(password.length() < 6){
74              Toast.makeText( context: MainActivity.this, text: "The Minimum length of a password is 6!",Toast.LENGTH_SHORT).show();
75              return;
76          }
77
78          mAuth.signInWithEmailAndPassword(email,password)
79                  .addOnCompleteListener( activity: this, (task) → {
82                  if(task.isSuccessful()){
83                      Toast.makeText( context: MainActivity.this, text: "WELCOME!",Toast.LENGTH_SHORT).show();
84                      startActivity(new Intent( packageContext: MainActivity.this, Firebase.class));
85                  } else{
86                      Toast.makeText( context: MainActivity.this, text: "Please enter a correct email and password!",Toast.LENGTH_SHORT).show();
87                  }
88          });
90      }
91
92  @   public void onClick(View view) {
93          switch (view.getId()){
94              case R.id.logIn:
95                  logIn();
96                  break;
97              case R.id.textViewSuggestSignUp:
98                  startActivity(new Intent( packageContext: this,SignUp.class));
99                  break;
100         }
101     }
102 }
```

Run your app:

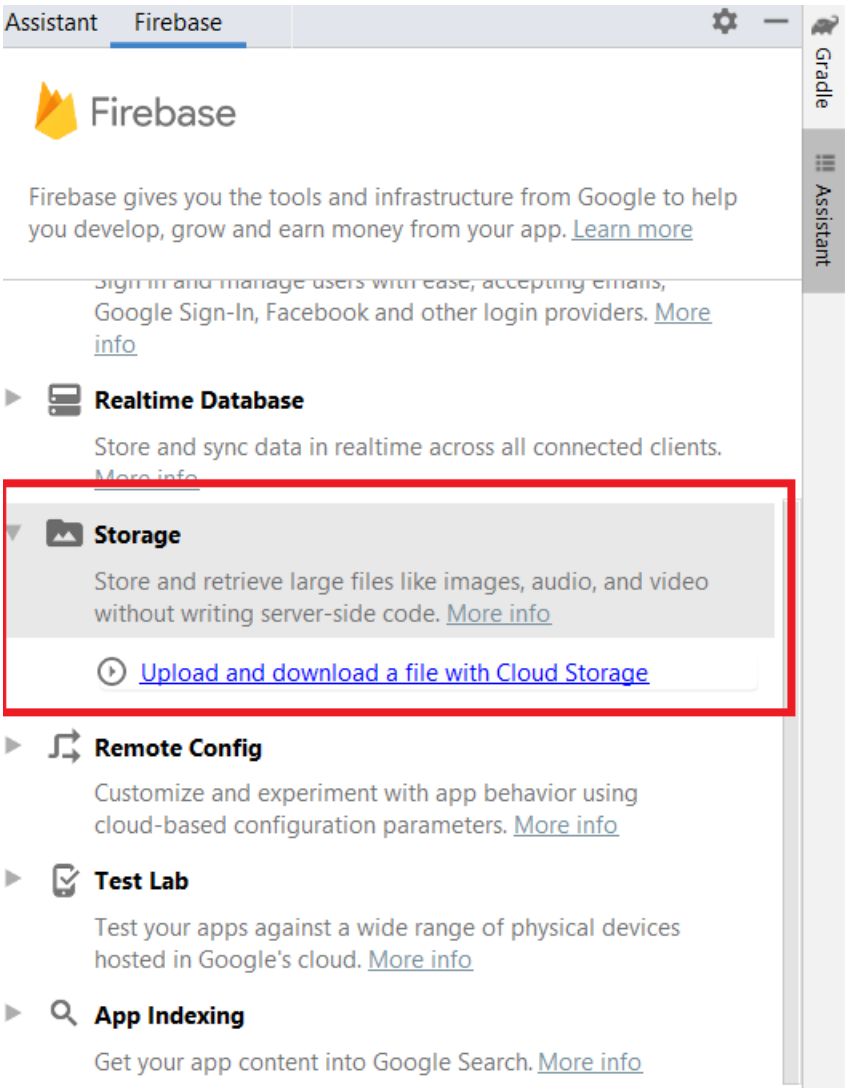

2:51
Firebase
WELCOME!
shiv@gmail.com
LOG IN
Dont have an account? SIGNUP


2:51
Welcome to Firebase
WELCOME!

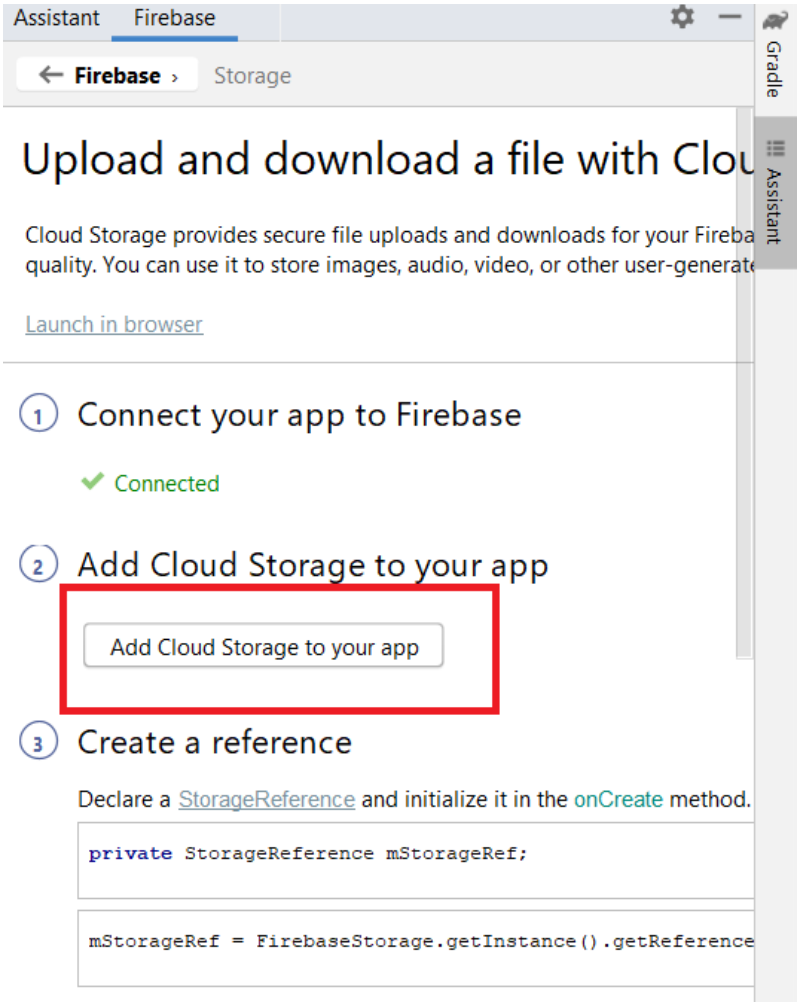
Firebase also provides storage for your data. Below I will show you how to upload your images in the firebase storage.

First, we will have to add Firebase storage to our app.

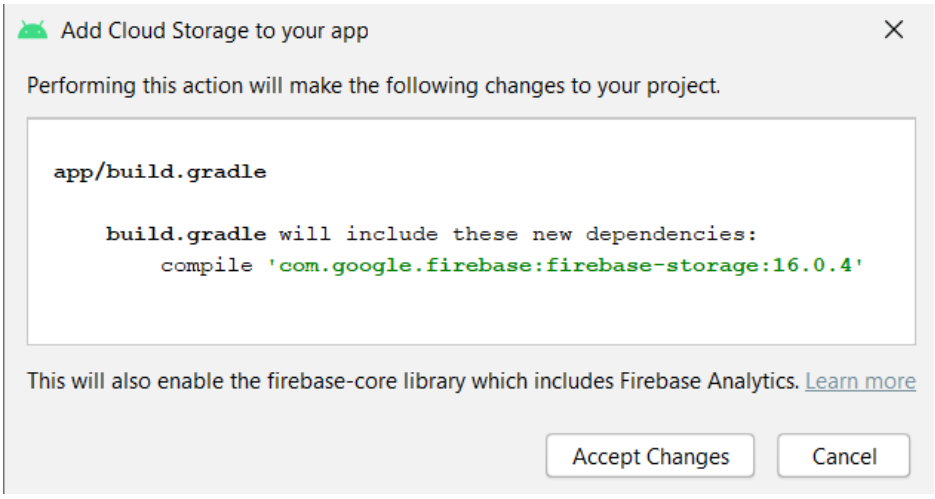
In you assistant tab, click on storage and then the upload and download a file with cloud storage link.



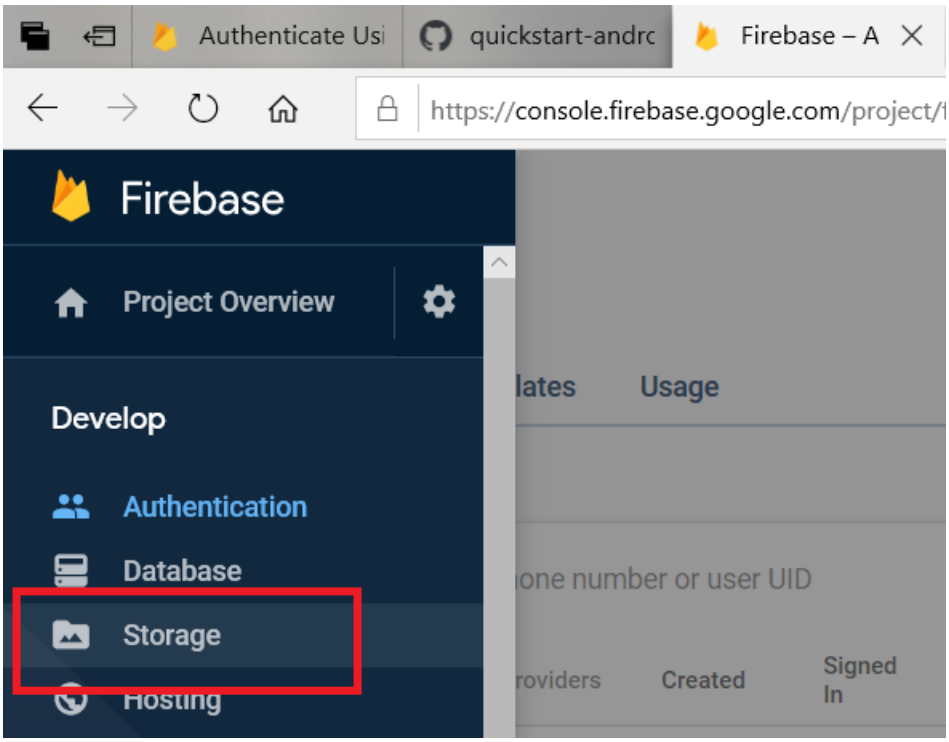
Click on the add cloud storage to your app button.



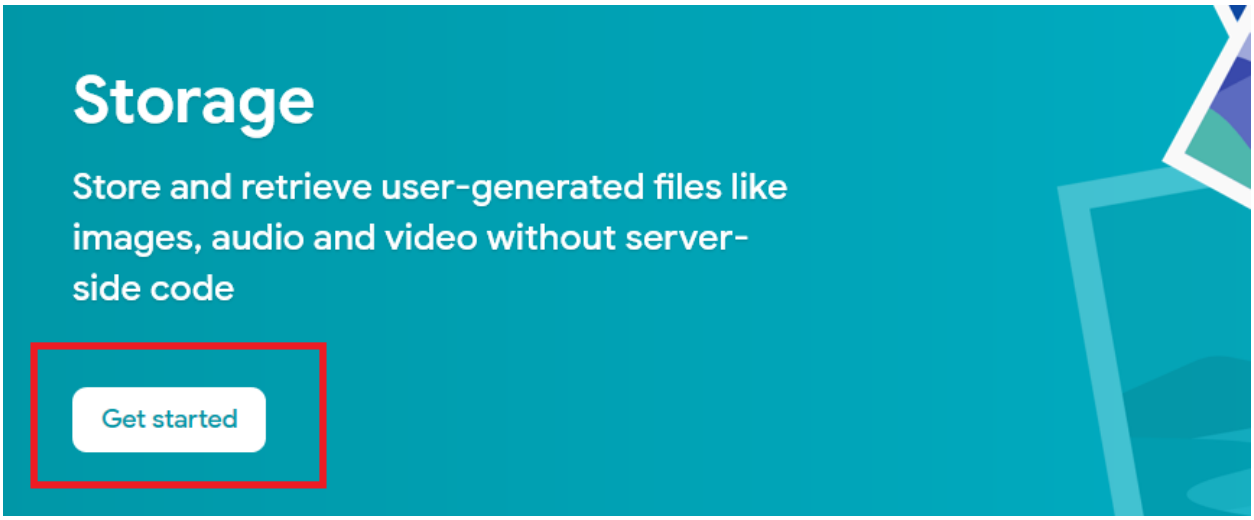
In the dialog box that appears, click Accept.



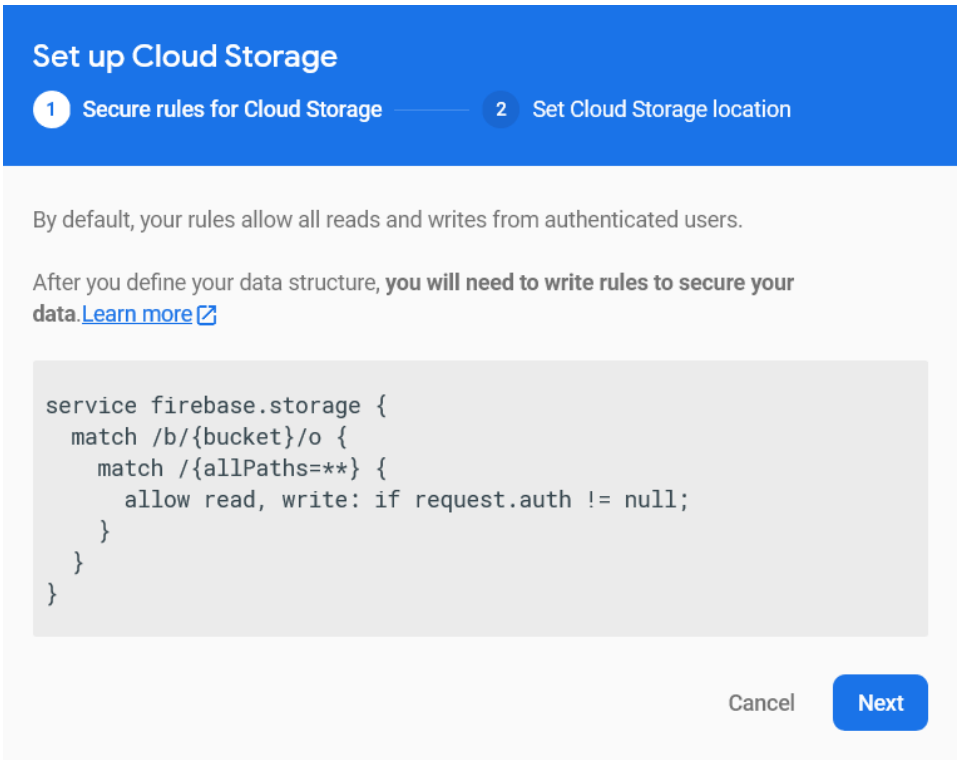
Go to your Firebase console and click on Storage from the navigation tab on the left.

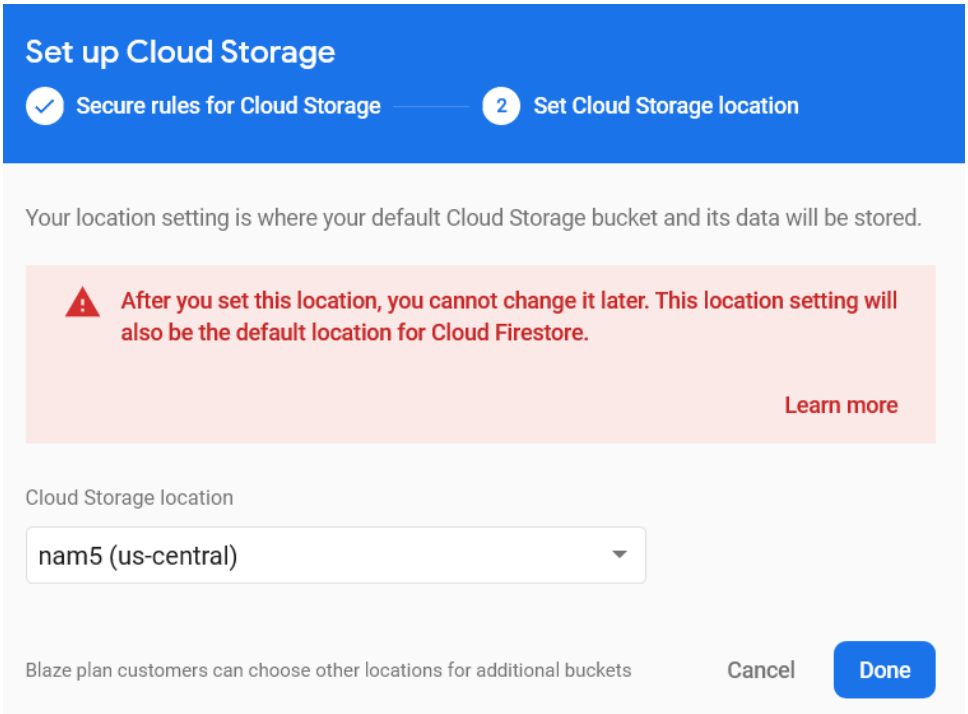


Click Get started on the page that appears.

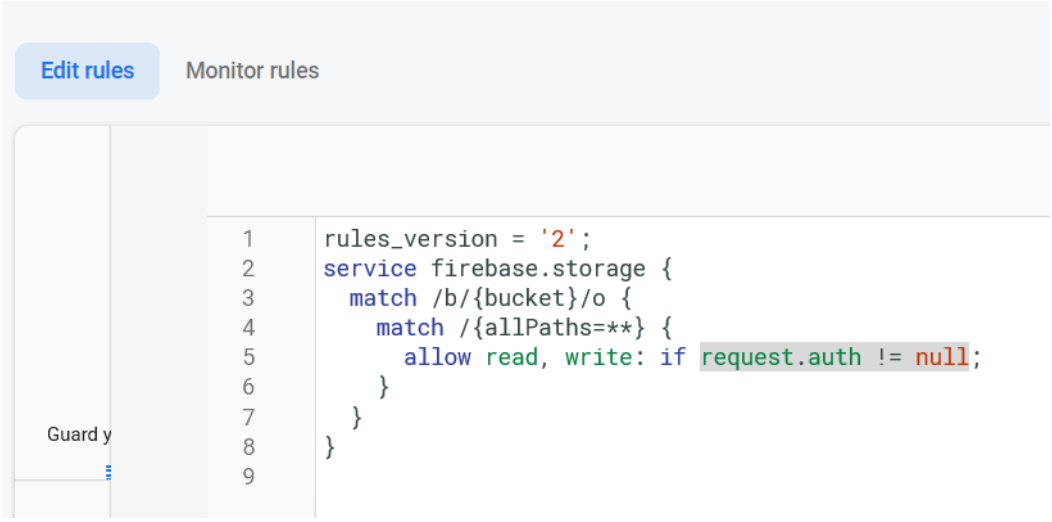


A dialog box will appear. Click next and then done.

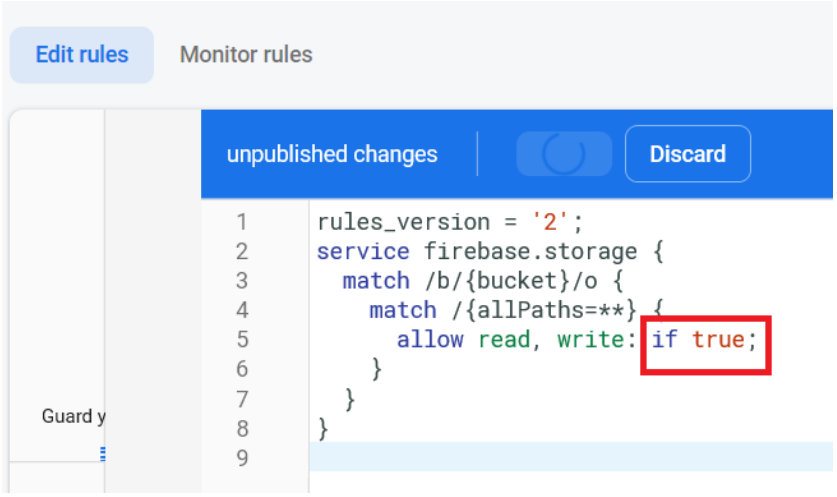


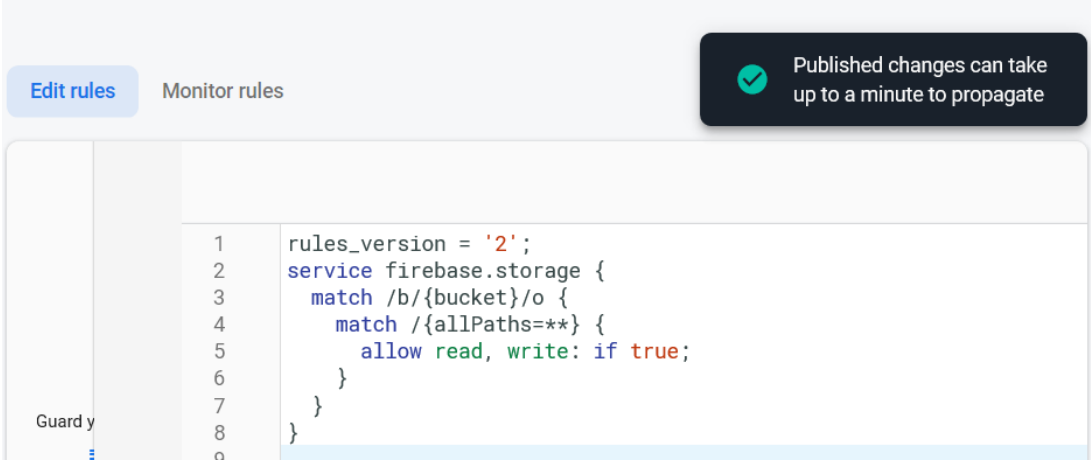


Click on the rules tab. The following Json file will appear.

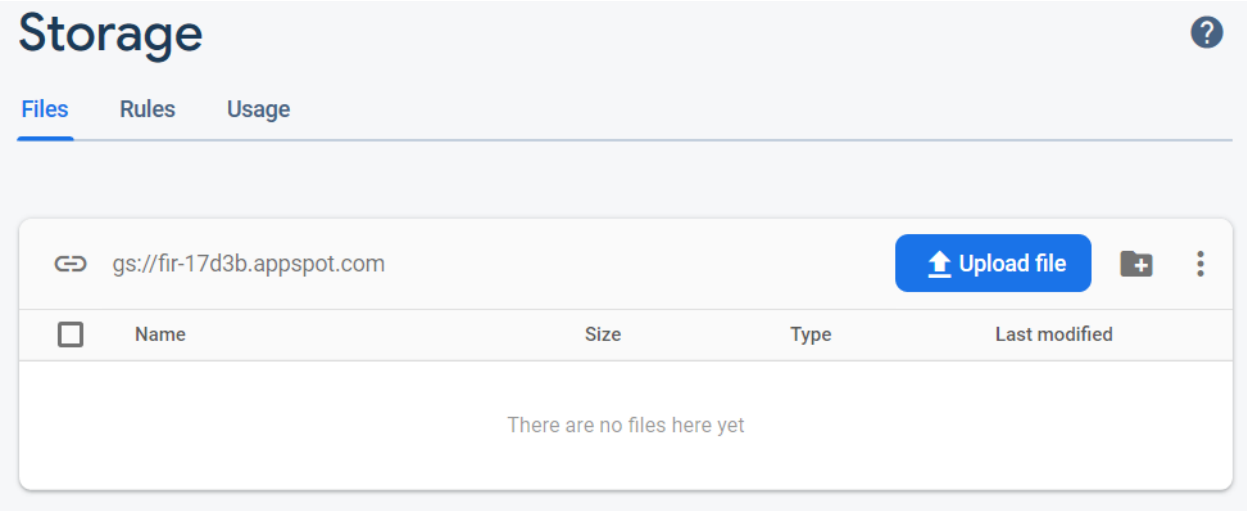


Edit rules to allow read, write: if true;





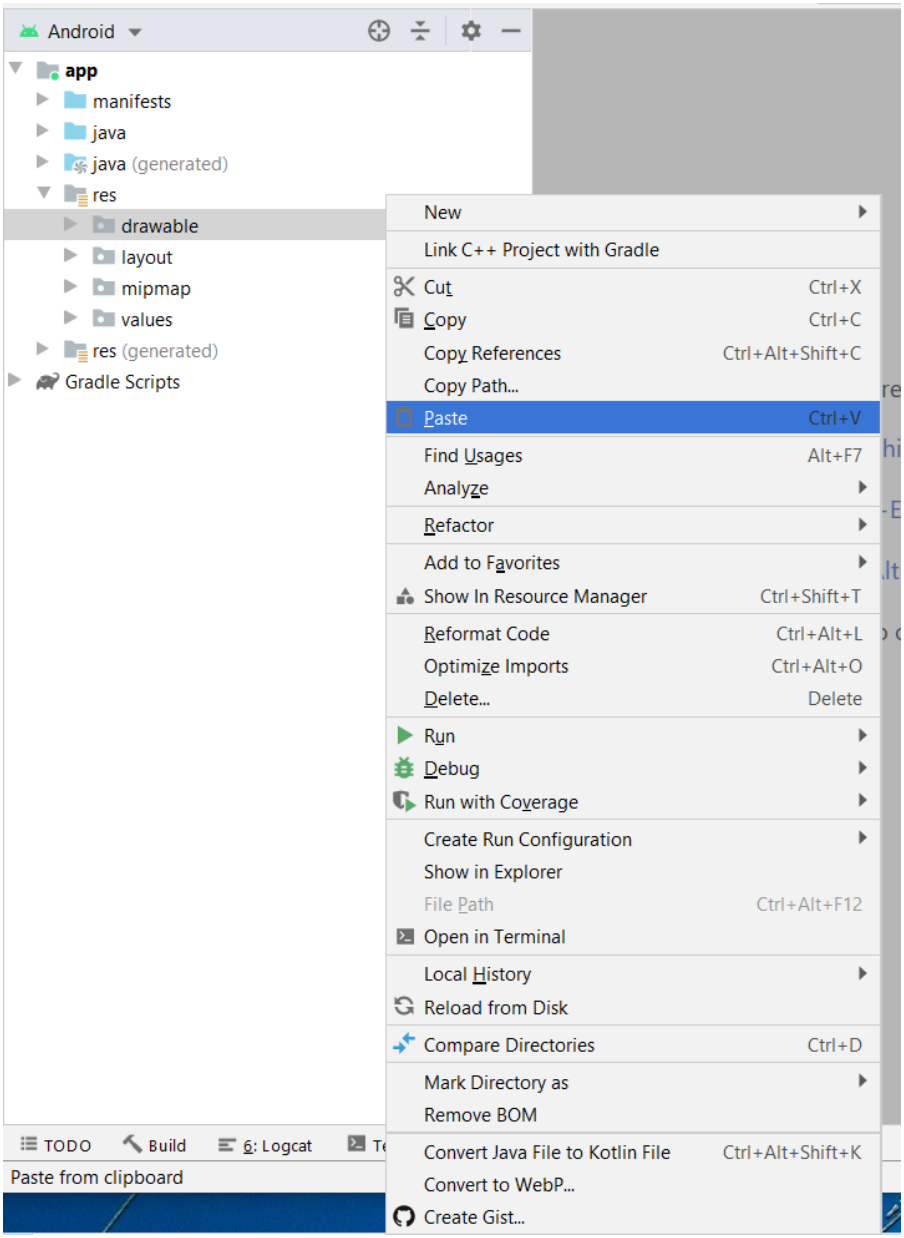
You can view your images in the files tab in your Storage.



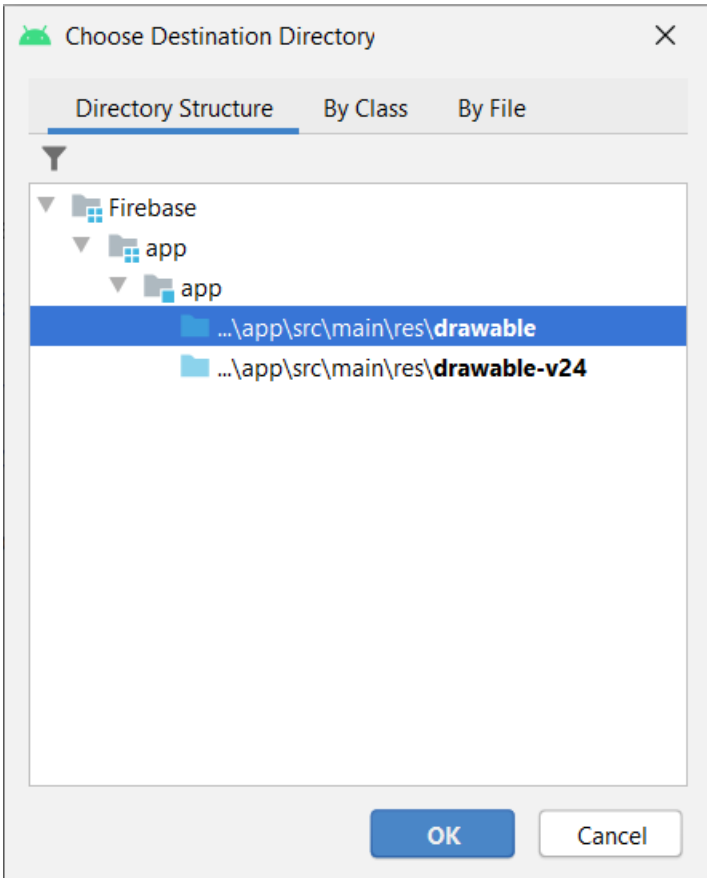
Now that we have set up the Firebase Storage, let's create our layout in the activity_firebase.xml.

Before we start creating the layout, we need to import an image to our drawable folder. I use the image below.

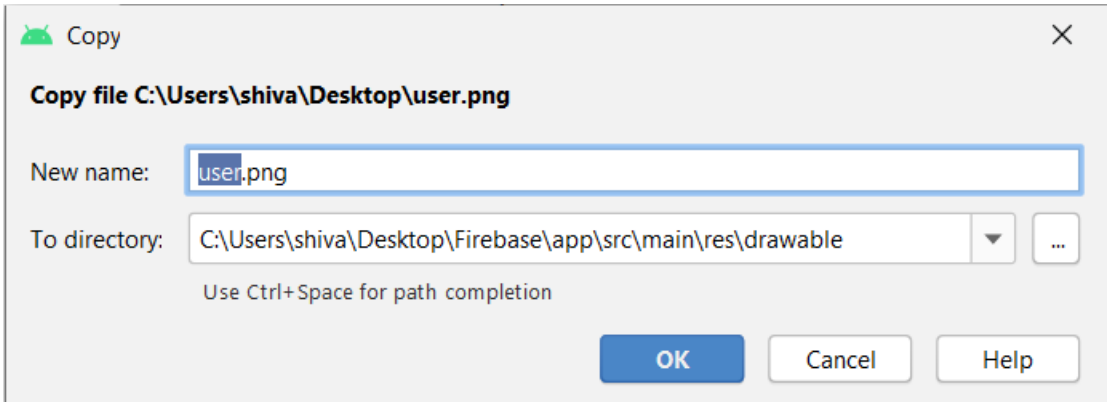
Copy the image. Right-click on the drawable folder under res folder.



A dialog box will appear. Click OK.



Another dialog box will appear. We will let the image name stay the same and click OK.



Our layout will have an ImageView and two buttons, one to choose a file and one to upload.

Add the following code:

Firebase.java | activity_firebase.xml

Code | Split | Des

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


    <Button
        android:id="@+id/btnChoose"
        android:layout_width="308dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="51dp"
        android:layout_marginEnd="51dp"
        android:layout_marginBottom="31dp"
        android:text="Choose file.."
        app:layout_constraintBottom_toTopOf="@+id/btnUpload"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView" />

    <Button
        android:id="@+id/btnUpload"
        android:layout_width="308dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="136dp"
        android:text="Upload"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnChoose" />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="308dp"
        android:layout_height="308dp"
        android:layout_marginTop="80dp"
        android:layout_marginBottom="44dp"
        app:layout_constraintBottom_toTopOf="@+id/btnChoose"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.495"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/user" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Let's move to our Firebase class.

We will declare the following g variables. We have a StorageReference variable that holds the reference to our storage location.

```
public class Firebase extends AppCompatActivity {

    Button choose, upload;
    ImageView image;
    StorageReference reference;
    public static final int REQUEST_CODE = 1;
    Uri uriImg;
```

In our onCreate method, we will instantiate our reference variable to get the firebase instance and the reference to our storage location. We then get access to our buttons and ImageView. Set onClickListeners for your buttons.

```
33      @Override
34      protected void onCreate(Bundle savedInstanceState) {
35          super.onCreate(savedInstanceState);
36          setContentView(R.layout.activity_firebase);
37
38          reference = FirebaseStorage.getInstance().getReference();
39
40          choose = findViewById(R.id.btnChoose);
41          upload = findViewById(R.id.btnUpload);
42          image = findViewById(R.id.imageView);
43
44          choose.setOnClickListener(new View.OnClickListener() {
45              @Override
46              public void onClick(View v) {
47              }
48          });
49
50          upload.setOnClickListener(new View.OnClickListener() {
51              @Override
52              public void onClick(View v) {
53              }
54          });
55
56      }
```

Let's create a chooseFile method. This method sets the intent type to image/* and action as ACTION_GET_CONTENT to get the images from the phone gallery.

```
60      private void chooseFile(){
61          Intent intent = new Intent();
62          intent.setType("image/*");
63          intent.setAction(Intent.ACTION_GET_CONTENT);
64          startActivityForResult(intent,REQUEST_CODE);
65      }
```

We override the onActivityResult() method to get the result back from the startActivityForResult().

```
91      public void onActivityResult(int requestCode, int resultCode, Intent data) {
92          super.onActivityResult(requestCode, resultCode, data);
93          if(requestCode == REQUEST_CODE && resultCode == RESULT_OK
94          && data != null && data.getData() != null){
95              uriImg = data.getData();
96              image.setImageURI(uriImg);
97          }
98      }
```

Now lets call the chooseFile() method in our onClickListener for our choose button.

```
44          choose.setOnClickListener(new View.OnClickListener() {
45              @Override
46              public void onClick(View v) {
47                  chooseFile();
48              }
49          });
```

We will create a getExtention method to return a file extension.

```
85      private String getExtention(Uri uri){
86          ContentResolver contentResolver = getContentResolver();
87          MimeTypeMap map = MimeTypeMap.getSingleton();
88          return map.getExtensionFromMimeType(contentResolver.getType(uri));
89      }
```

Now let's create an uploadFile method. In this method, we create a reference to our selected image location and use putFile() method to upload the image.

```
67      private void uploadFile(){
68          StorageReference ref = reference.child(getExtention(uriImg));
69
70          ref.putFile(uriImg)
71                  .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
72                      @Override
73                      public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
74                          Toast.makeText( context: Firebase.this, text: "Image Uploaded!",Toast.LENGTH_SHORT).show();
75                      }
76                  })
77                  .addOnFailureListener(new OnFailureListener() {
78                      @Override
79                      public void onFailure(@NonNull Exception exception) {
80                          Toast.makeText( context: Firebase.this, text: "Error Uploading the image!",Toast.LENGTH_SHORT).show();
81                      }
82                  });
83      }
```

Your Firebase class looks as follows:

Firebase.java

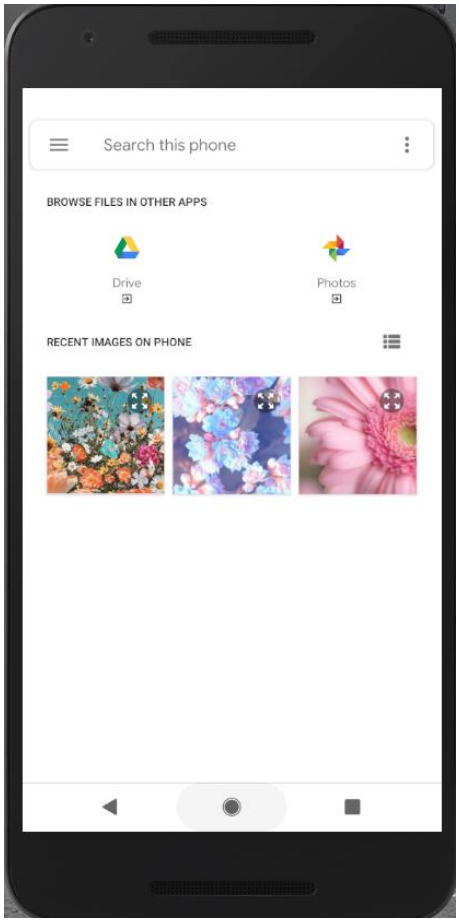
```
1   package com.example.shivangi.firebase;
2
3   import ...
24
25  public class Firebase extends AppCompatActivity {
26
27      Button choose, upload;
28      ImageView image;
29      StorageReference reference;
30      public static final int REQUEST_CODE = 1;
31      Uri uriImg;
32
33      @Override
34      protected void onCreate(Bundle savedInstanceState) {
35          super.onCreate(savedInstanceState);
36          setContentView(R.layout.activity_firebase);
37
38          reference = FirebaseStorage.getInstance().getReference();
39
40          choose = findViewById(R.id.btnChoose);
41          upload = findViewById(R.id.btnUpload);
42          image = findViewById(R.id.imageView);
43
44          choose.setOnClickListener(new View.OnClickListener() {
45              @Override
46              public void onClick(View v) {
47                  chooseFile();
48              }
49          });
```

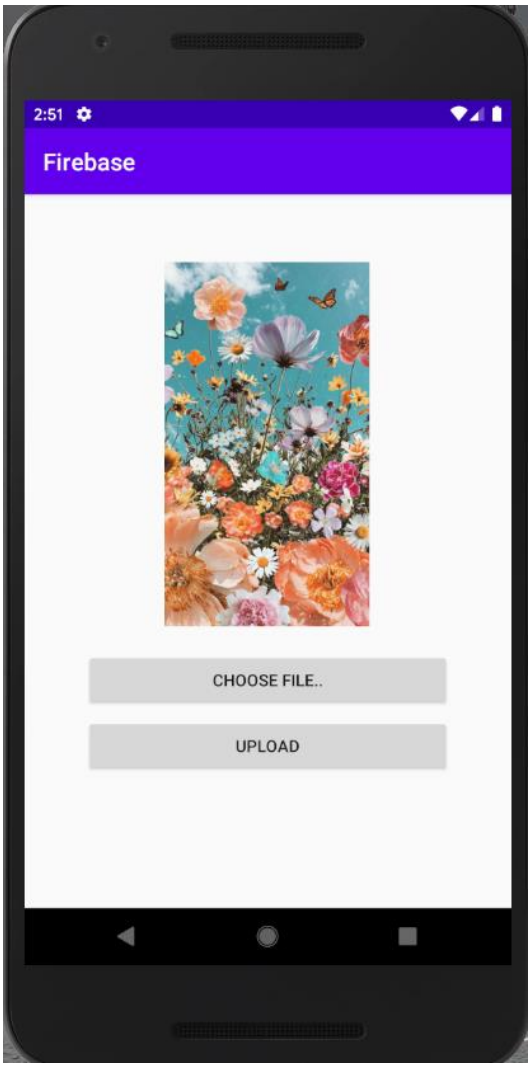
```java
        upload.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                uploadFile();
            }
        });

    }

    private void chooseFile(){
        Intent intent = new Intent();
        intent.setType("image/*");
        intent.setAction(Intent.ACTION_GET_CONTENT);
        startActivityForResult(intent,REQUEST_CODE);
    }

    private void uploadFile(){
        StorageReference ref = reference.child(getExtention(uriImg));

        ref.putFile(uriImg)
                .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
                    @Override
                    public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
                        Toast.makeText( context: Firebase.this, text: "Image Uploaded!",Toast.LENGTH_SHORT).show();
                    }
                })
                .addOnFailureListener(new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception exception) {
                        Toast.makeText( context: Firebase.this, text: "Error Uploading the image!",Toast.LENGTH_SHORT).show();
                    }
                });
    }
```

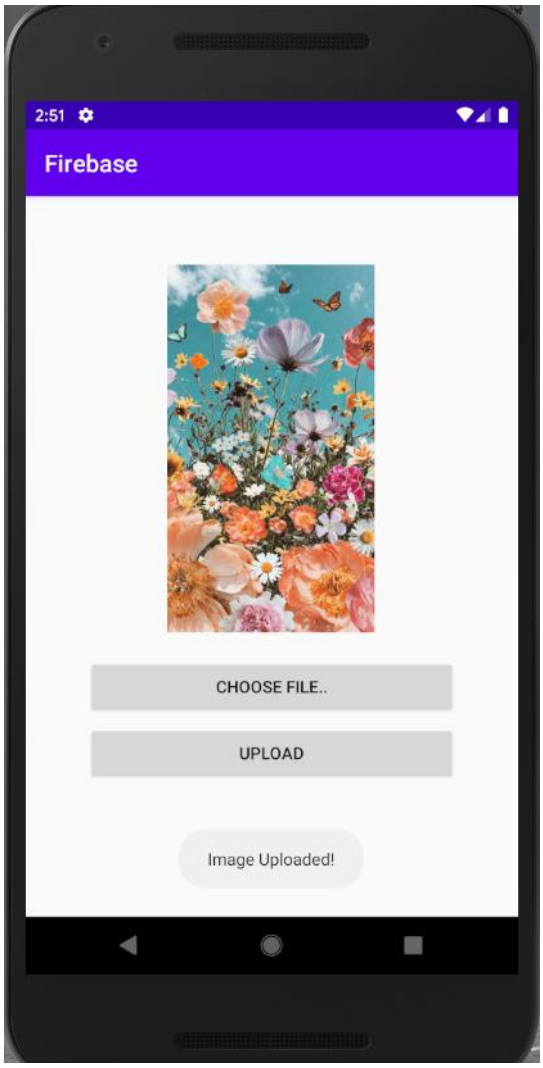
```java
    private String getExtention(Uri uri){
        ContentResolver contentResolver = getContentResolver();
        MimeTypeMap map = MimeTypeMap.getSingleton();
        return map.getExtensionFromMimeType(contentResolver.getType(uri));
    }

    public void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if(requestCode == REQUEST_CODE && resultCode == RESULT_OK
        && data != null && data.getData() != null){
            uriImg = data.getData();
            image.setImageURI(uriImg);
        }
    }
}
```

For this example, I have downloaded images in our emulator.

Run the app.




2:51
Firebase
CHOOSE FILE..
UPLOAD


2:51
Firebase
CHOOSE FILE..
UPLOAD
Image Uploaded!

RESOURCES:

https://howtofirebase.com/what-is-firebase-fcb8614ba442

https://firebase.google.com/docs/android/setup

https://firebase.google.com/docs/auth/android/start

https://developer.android.com/reference/java/util/regex/Pattern

https://developers.google.com/android/guides/client-auth

https://firebase.google.com/docs/auth/android/google-signin?utm_source=studio

https://firebase.google.com/docs/storage/android/upload-files

https://firebase.google.com/docs/storage/android/create-reference

https://www.geeksforgeeks.org/android-how-to-upload-an-image-on-firebase-storage/

https://stackoverflow.com/questions/8589645/how-to-determine-mime-type-of-file-in-android