

EXPERIMENT 6

Aim: To set up Flutter with Firebase for Android apps.

Theory:

Firebase is a comprehensive platform provided by Google for building mobile and web applications. It offers a wide range of tools and services to help developers build high-quality apps quickly and efficiently. Here's an overview of Firebase and its key components:

Key Components:

1. **Realtime Database:** Firebase Realtime Database is a NoSQL cloud database that allows developers to store and sync data in real-time across multiple clients. It's suitable for applications requiring real-time updates, such as chat apps, collaborative tools, and live data feeds.
2. **Cloud Firestore:** Firestore is Firebase's newer database solution that offers more powerful querying capabilities, scalability, and real-time updates. It's a flexible, scalable database for mobile, web, and server development.
3. **Authentication:** Firebase Authentication provides a secure and easy-to-use authentication system that supports various authentication methods, including email/password, phone number, Google Sign-In, Facebook Login, and more.
4. **Cloud Storage:** Firebase Cloud Storage allows developers to store and serve user-generated content, such as images, videos, and audio files, directly from Google's infrastructure. It offers scalable, secure, and reliable storage solutions with powerful SDKs for easy integration.
5. **Analytics:** Firebase Analytics offers powerful analytics features to help developers understand user behavior, measure app performance, and track key metrics. It provides insights into user engagement, retention, and conversion rates.

Prerequisites:

- A Google account to use Firebase.
- Developing for iOS will require XCode.
- To download and install Flutter.
- To download and install Android Studio and Visual Studio Code.
- It is recommended to install plugins for your code editor:
 - Flutter and Dart plugins installed for Android Studio.
 - Flutter extension installed for Visual Studio Code.

Creating a New Flutter Project

This tutorial will require the creation of an example Flutter app.

Once you have your environment set up for Flutter, you can run the following to create a new application:

```
flutter create FurnitureApp
```

Navigate to the new project directory:

```
cd FurnitureApp
```

Using `flutter create` will produce a demo application that will display the number of times a button is clicked.

Now that we've got a Flutter project up and running, we can add Firebase.


Creating a New Firebase Project


First, log in with your Google account to manage your Firebase projects. From within the Firebase dashboard, select the Create new project button and give it a name:

× Create a project (Step 1 of 3)

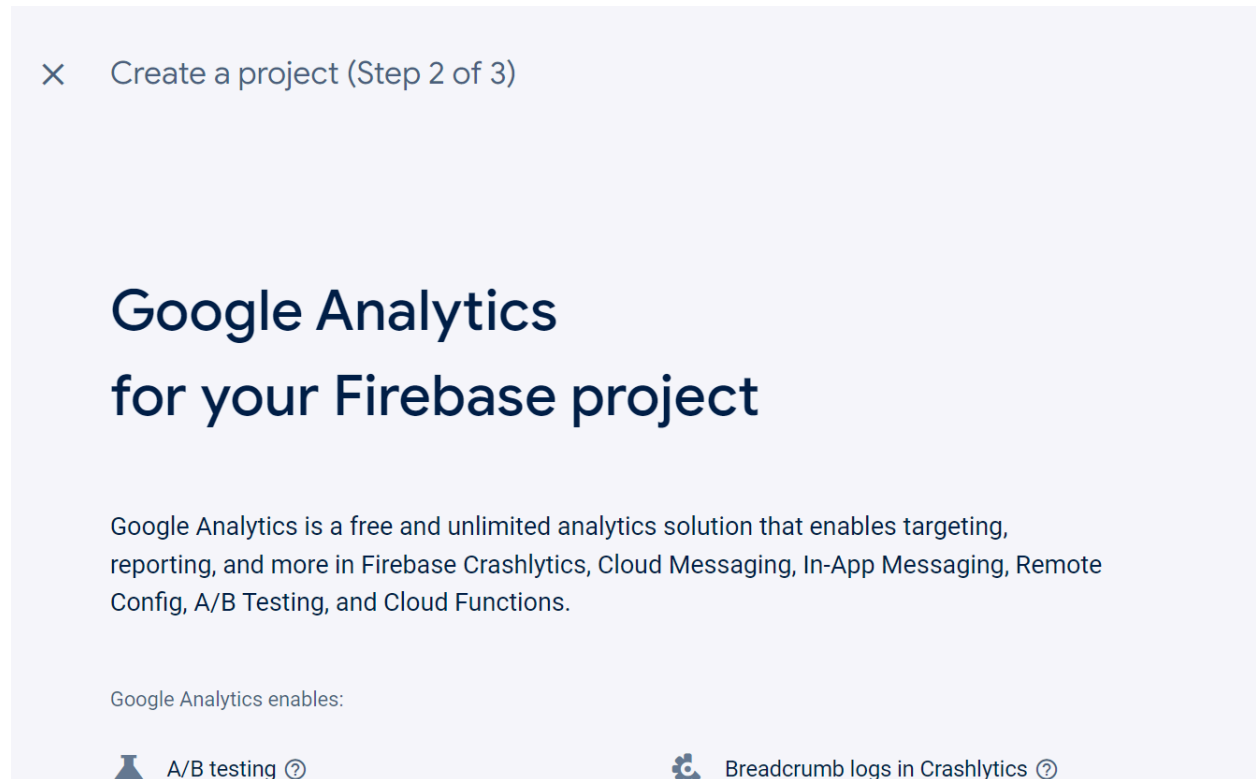
Let's start with a name for your project [?]

Project name

 furnitureapp-7ea6d

 Select parent resource

Next, we're given the option to enable Google Analytics. This tutorial will not require Google Analytics, but you can also choose to add it to your project.



If you choose to use Google Analytics, you will need to review and accept the terms and conditions prior to project creation.

After pressing Continue, your project will be created and resources will be provisioned. You will then be directed to the dashboard for the new project.

Adding Android support

Registering the App

In order to add Android support to our Flutter application, select the Android logo from the dashboard. This brings us to the following screen:

[illegible]

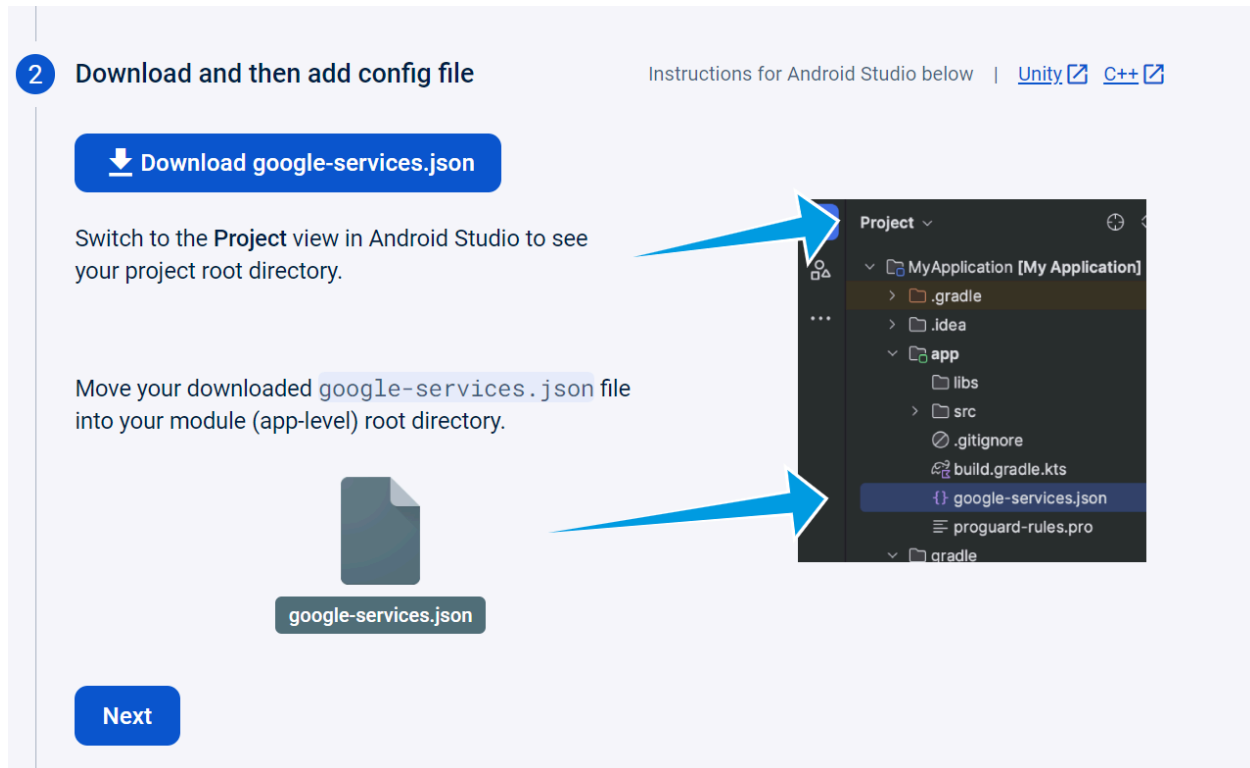
The most important thing here is to match up the Android package name that you choose here with the one inside of our application.

The structure consists of at least two segments. A common pattern is to use a domain name, a company name, and the application name

```
defaultConfig {  
    // TODO: Specify your own unique Application ID (https://developer.android.com/studio/run/application-id)  
    applicationId "com.furniture.furnitureapp"  
    // You can update the following values to match your application needs.  
    // For more information, see: https://docs.flutter.dev/deployment/android#deploying-your-application  
    minSdkVersion flutter.minSdkVersion  
    targetSdkVersion flutter.targetSdkVersion  
    versionCode flutterVersionCode.toInteger()  
    versionName flutterVersionName  
}
```

Downloading the Config File

The next step is to add the Firebase configuration file into our Flutter project. This is important as it contains the API keys and other critical information for Firebase to use.



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.

`google-services.json`

[Next](#)

Next, move the `google-services.json` file to the `android/app` directory within the Flutter project.

Adding the Firebase SDK

We'll now need to update our Gradle configuration to include the Google Services plugin.

Open `android/build.gradle` in your code editor and modify it to include the following:

```
buildscript {
  repositories {
    // Check that you have the following line (if not, add it):
    google() // Google's Maven repository
  }
  dependencies {
    ...
    // Add this line
    classpath 'com.google.gms:google-services:4.3.6'
  }
}

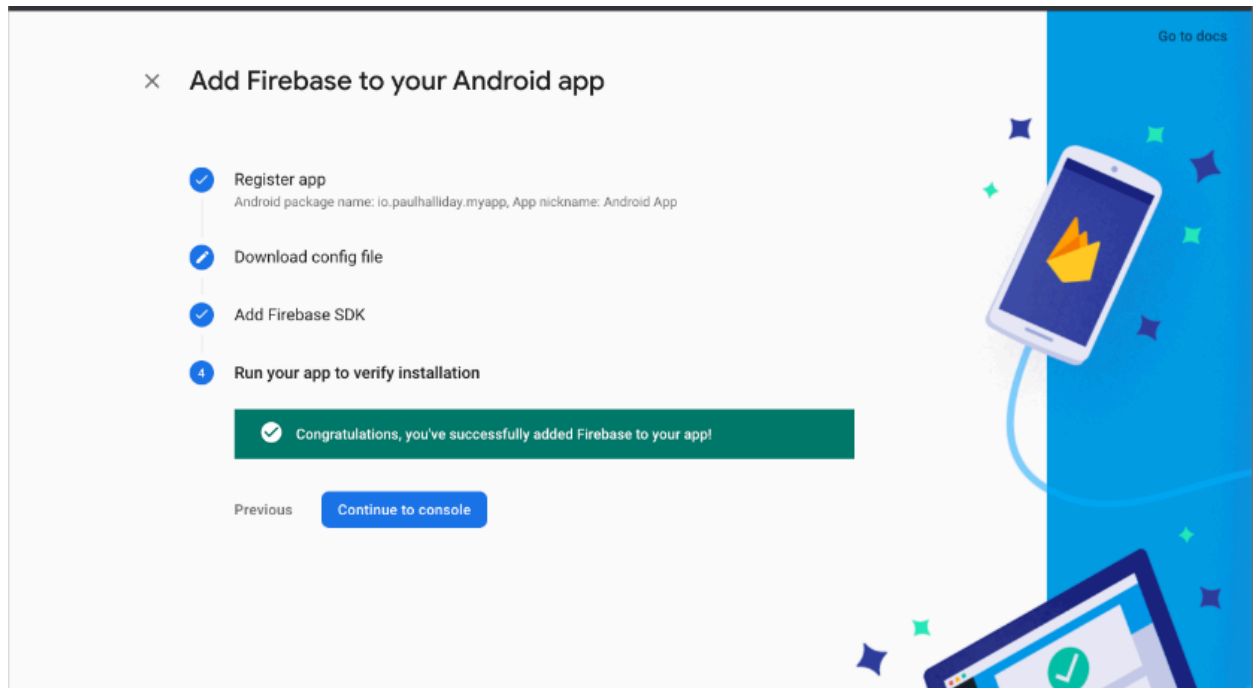
allprojects {
  ...
  repositories {
    // Check that you have the following line (if not, add it):
    google() // Google's Maven repository
    ...
  }
}
```

```
apply plugin: 'com.android.application'
// Add this line
apply plugin: 'com.google.gms.google-services'

dependencies {
  // Import the Firebase BoM
  implementation platform('com.google.firebase:firebase-bom:28.0.0')

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

With this update, we're essentially applying the Google Services plugin as well as looking at how other Flutter Firebase plugins can be activated such as Analytics.



Conclusion:

We have learned how to set up our Flutter application to be used with Firebase.

Flutter has official support for Firebase with the FlutterFire set of libraries.