Experiment No: 06

How To Set Up Firebase with Flutter for iOS and Android Apps

Firebase is a great backend solution for anyone that wants to use authentication, databases, cloud functions, ads, and countless other features within an app.

In this article, you will create a Firebase project for iOS and Android platforms using

Flutter. **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 flutterfirebaseexample`

Navigate to the new project directory:

`cd flutterfirebaseexample`

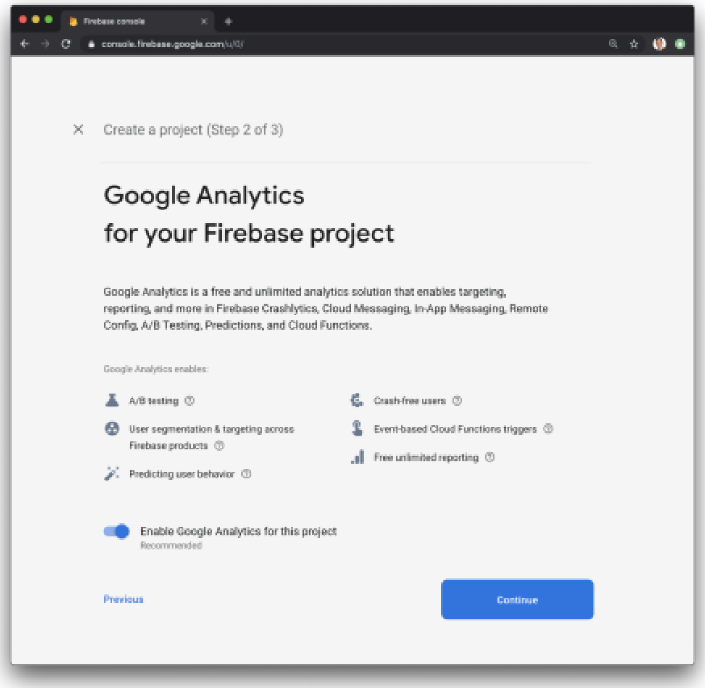
Using `flutter create` will produce a demo application that will display the number of times a button is clicked.

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:

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.



Create a project (Step 2 of 3)

Google Analytics
for your Firebase project

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, Predictions, and Cloud Functions.

Google Analytics enables:

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

Enable Google Analytics for this project ✓
Recommended

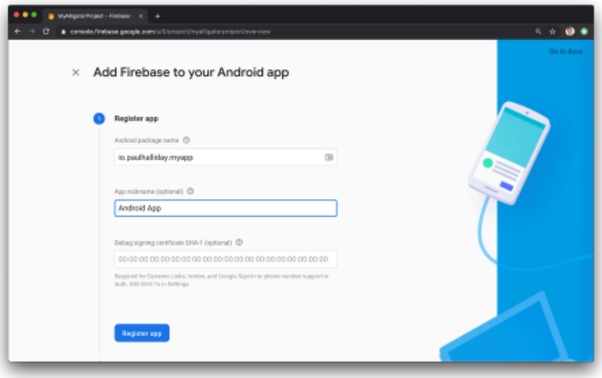
f you choose to use Google Analytics, you will need to review and accept the terms and conditions prior to project creation.

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:


MyAlligatorProject – Firebase
console.firebase.google.com/u/0/project/myalligatorproject/overview
Go to docs

Add Firebase to your Android app

1 Register app

Android package name: io.paulhalliday.myapp

App nickname (optional): Android App

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

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

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:

```
com.example.flutterfirebaseexample
```

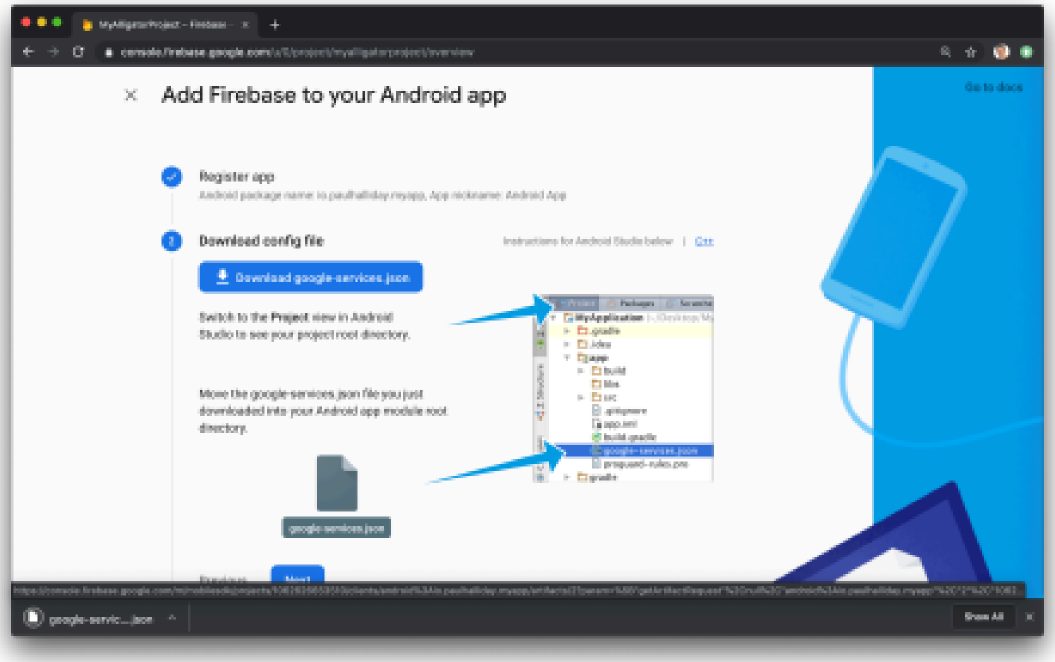
Once you've decided on a name, open `android/app/build.gradle` in your code editor and update the `applicationId` to match the Android package name:

android/app/build.gradle

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.

Select Download `google-services.json` from this page:


Add Firebase to your Android app
Register app
Download config file
Download google-services.json
Switch to the Project view in Android Studio to see your project root directory.
Move the google-services.json file you just downloaded into your Android app module root directory.
google-services.json
Next

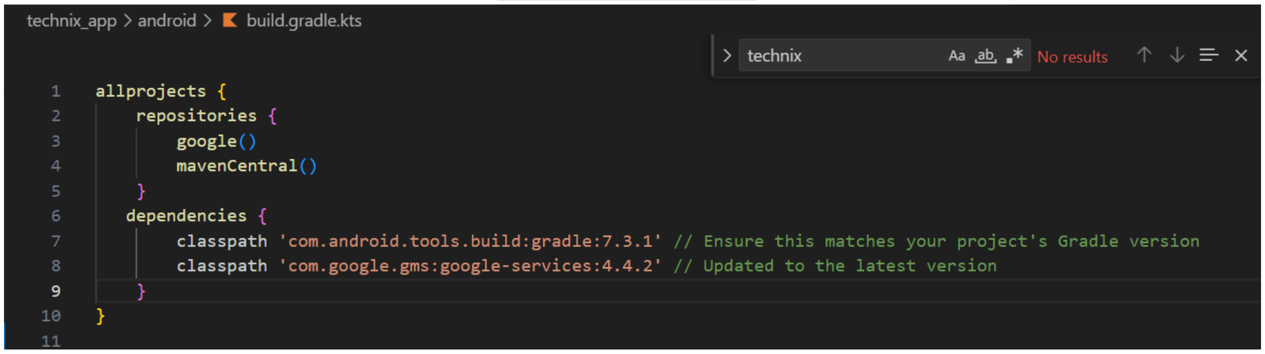
Next, move the `google-services.json` file to the `android/app` directory within the Flutter project.

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:

android/buiild.gradle



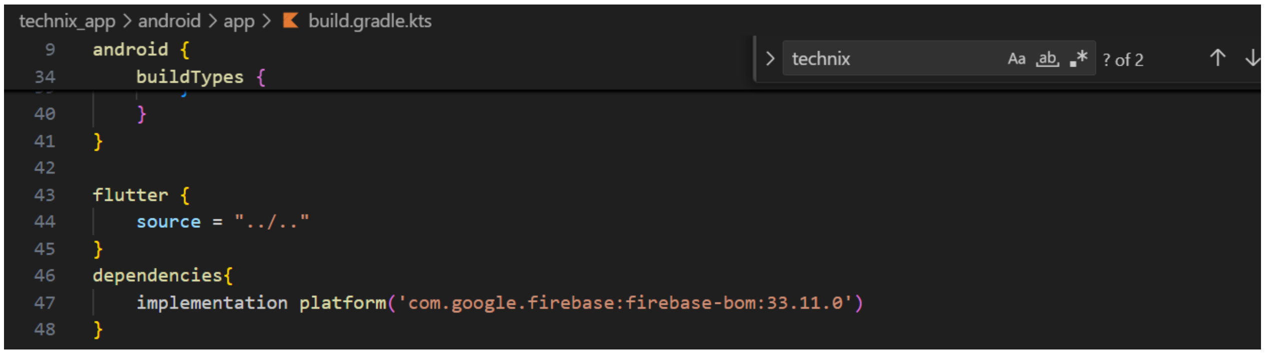
technix_app > android > build.gradle.kts

technix   No results

```
allprojects {
    repositories {
        google()
        mavenCentral()
    }
   dependencies {
        classpath 'com.android.tools.build:gradle:7.3.1' // Ensure this matches your project's Gradle version
        classpath 'com.google.gms:google-services:4.4.2' // Updated to the latest version
    }
}
```

Finally, update the app level file at `android/app/build.gradle` to include the following:

android/app/build.gradle



technix_app > android > app > build.gradle.kts

technix   ? of 2

```
android {
    buildTypes {
    }
}

flutter {
    source = "../.."
}
dependencies{
    implementation platform('com.google.firebase:firebase-bom:33.11.0')
}
```

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.

From here, run your application on an Android device or simulator. If everything has worked correctly, you should get the following message in the dashboard:
Next up, let's add iOS support!

Adding iOS Support

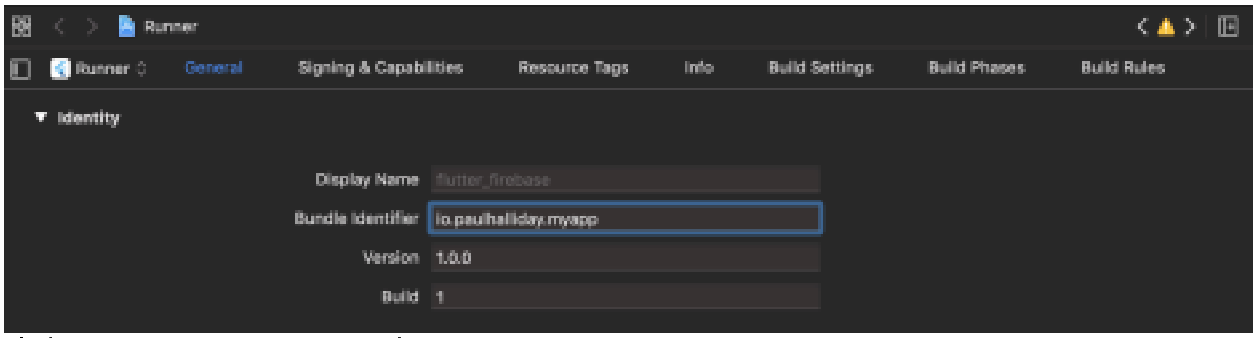
In order to add Firebase support for iOS, we have to follow a similar set of instructions.

Head back over to the dashboard and select Add app and then iOS icon to be navigated to the setup process.

Registering an App

Once again, we'll need to add an "iOS Bundle ID". It is possible to use the "Android package name" for consistency:

You'll then need to make sure this matches up by opening the iOS project up in Xcode at `ios/Runner/Runner.xcodeproj` and changing the Bundle identifier under General:

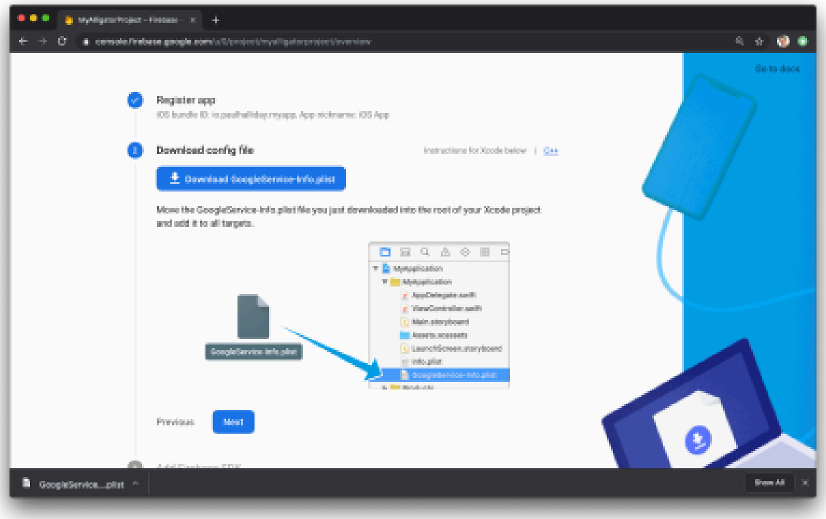


Display Name flutter_firebase
Bundle Identifier io.paulhalliday.myapp
Version 1.0.0
Build 1

Click Register app to move to the next screen.

Downloading the Config File

In this step, we'll need to download the configuration file and add this to our Xcode project.



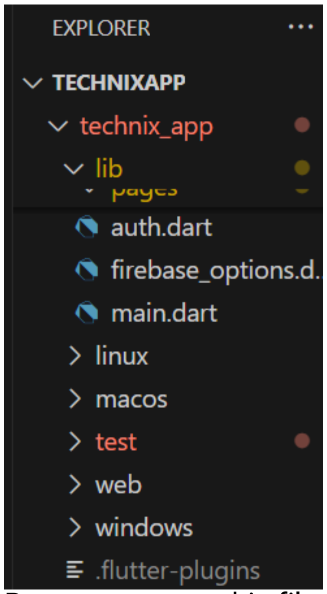
Register app
iOS bundle ID: io.paulhalliday.myapp, App nickname: iOS App

Download config file
Instructions for Xcode below | C++

Move the GoogleService-Info.plist file you just downloaded into the root of your Xcode project and add it to all targets.

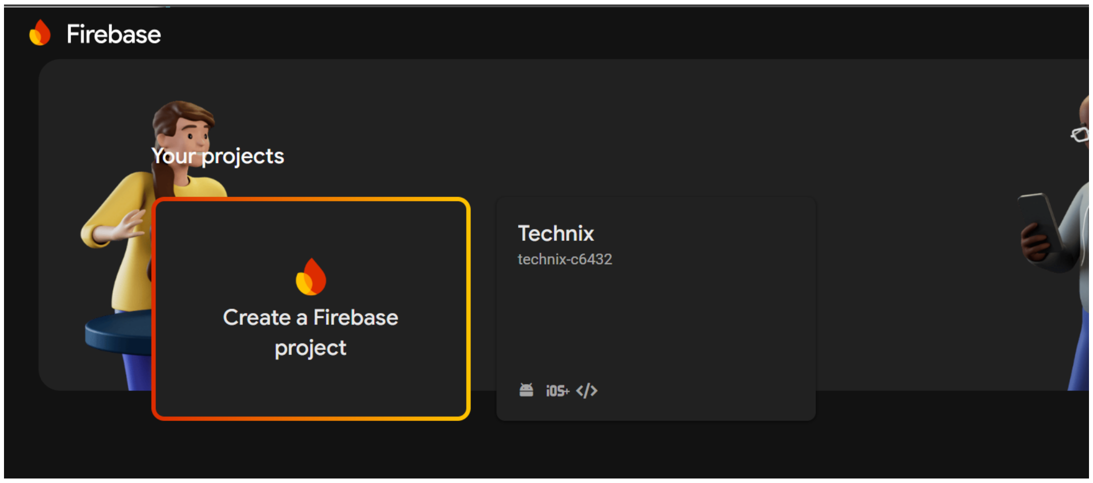
Add Firebase SDK

Download `GoogleService-Info.plist` and move this into the root of your Xcode project within `Runner`:



Be sure to move this file within Xcode to create the proper file references.
There are additional steps for installing the Firebase SDK and adding initialization code, but they are not necessary for this tutorial.

That's it!



Conclusion

In this article, you learned how to set up and ready our Flutter applications to be used with Firebase.

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