**Cloud Firestore for flutter:**

[**https://firebase.google.com/docs/flutter/setup**](https://firebase.google.com/docs/flutter/setup)

**Prerequisites (Android Setup)**

* Install your preferred editor or IDE.
* Make sure that your app meets the following requirements:
  + Targets API level 16 (Jelly Bean) or later
  + Uses Gradle 4.1 or later
* Set up a device or emulator for running your app. Emulators must use an emulator image with Google Play.
* Install Flutter for your specific operating system, including the following:
  + Flutter SDK
  + Supporting libraries
  + Platform-specific software and SDKs
* Sign into Firebase using your Google account.

Step 1: Create a Firebase project

Before you can add Firebase to your Flutter app, you need to create a Firebase project to connect to your app.

Step 2: Register your app with Firebase

**Important:** If Flutter app is to be released on both iOS and Android, register *both* the iOS and Android versions of the app *with the same Firebase project*.

1. In the center of the Firebase console's project overview page, click the **Android** icon to launch the setup workflow.

If you've already added an app to your Firebase project, click **Add app** to display the platform options.

1. Enter your app's package name in the **Android package name** field.

Make sure that you enter the ID that your app is actually using. You cannot add or modify this value after you register your app with your Firebase project.

* + A package name is sometimes referred to as an application ID.
  + Find this package name in your module (app-level) Gradle file, usually app/build.gradle (example package name: com.yourcompany.yourproject).

1. (Optional) Enter other app information as prompted by the setup workflow.
   * **App nickname**: An internal, convenience identifier that is only visible to you in the Firebase console
   * **Debug signing certificate SHA-1**: A SHA-1 hash is required by Firebase Authentication (when using Google Sign In or phone number sign in) and Firebase Dynamic Links.
2. Click **Register app**.

Step 3: Add a Firebase configuration file

1. Click **Download google-services.json** to obtain your Firebase Android config file (google-services.json).
   * You can download your Firebase Android config file again at any time.
   * Make sure the config file is not appended with additional characters, like (2).
2. Move your config file into the android/app directory of your Flutter app.
3. To enable Firebase services in your Android app, add the google-services plugin to your Gradle files, as follows:
   * In your **root-level (project-level)** Gradle file (android/build.gradle), add rules to include the Google Services Gradle plugin. Check that you have Google’s Maven repository, as well.

buildscript {  
  
    repositories {  
      **// Check that you have the following line (if not, add it):  
      google()  // Google's Maven repository**  
    }  
  
    // ...  
  
    dependencies {  
      // ...  
  
      **// Add the following line:  
      classpath 'com.google.gms:google-services:4.3.4'  // Google Services plugin**  
    }  
}  
  
allprojects {  
    // ...  
  
    repositories {  
      **// Check that you have following line (if not, add it):  
      google()  // Google's Maven repository**  
      // ...  
    }  
}

* + In your **module (app-level)** Gradle file (usually android/app/build.gradle), apply the Google Services Gradle plugin.

**// Add the following line:  
apply plugin: 'com.google.gms. google-services’ // Google Services plugin**  
  
android {  
  // ...  
}  
  
// ...

1. Run flutter packages get.
2. Back in the Firebase console setup workflow, click **Next** to skip the remaining steps.

Step 4: Add FlutterFire plugins

Flutter uses plugins to provide access to a wide range of platform-specific services, such as Firebase APIs. Plugins include platform-specific code to access services and APIs on each platform.

Firebase is accessed through a number of different libraries, one for each Firebase product (for example: Realtime Database, Authentication, Analytics, or Storage). Flutter provides a set of Firebase plugins, which are collectively called **FlutterFire**.

Since Flutter is a multi-platform SDK, each FlutterFire plugin is applicable for both iOS and Android. So, if you add any FlutterFire plugin to your Flutter app, it will be used by both the iOS and Android versions of your Firebase app.

[**iOS Setup**](https://firebase.google.com/docs/flutter/setup?platform=ios)