

Isumiro



Isumiro
ONLINE MARKETPLACE

Documentation

Android and iOS App

1. Introduction

Thank you for your interest in Isumiro.

This guide was implemented to help you set up this project successfully. For the process to go smoothly, it's essential to follow all the steps in this file.

Isumiro is a classified ads platform for iOS and Android made entirely with Flutter. Flutter is a mobile application framework created by Google to make cross-platform applications with one single codebase.

Isumiro uses Firebase as the backend service. Firebase, another product by Google, is a backend service that offers an online database, authentication service, storage, and much more.

This project includes one client app with some admin features and Firebase functions to run on the server.

2. Project Structure

The project is structured in these main different pages (these are the screens the user will interact with):

- **SpashPage**: this page displays the logo and checks if the user is logged in or not. If logged in, it redirects to the dashboard page, otherwise, it redirects to the login page.
- **LoginPage**: this page signs in the user using phone authentication.
- **RegistrationPage**: this page will be accessed when the user logs in for the first time and will allow him to complete his profile.
- **DashboardPage (bottom navigation)**: this is a holder of other pages, and is used to display the bottom navigation menu.
 - **HomePage**: this page displays in order from top to bottom, a search bar, a few categories that might interest the user, and a grid of items susceptible of interesting the user.
 - **ChatPage**: it displays a list of conversations started by the user in order to buy something, or by another user interested in the item the user is selling.
 - **SellPage**: this page allows the user to add an item for sale, by choosing a category, giving a description of the product, and uploading the pictures.
 - **AdPage**: this is a page listing the user's items on sale.
 - **AccountPage**: this is a page used by the user to manage his account.
- **SingleChatPage**: this page allows two users to exchange about an item in order to conclude a sale.
- **ItemPage (editable)**: this is a detail page of an item uploaded on the app.

- **SearchPage**: this page is used to search for items on the app.
- **DisabledPage**: To be displayed when the user's account has been disabled.
- **UserPage**: this page will display all the ads running for a single user.

3. Prerequisite

Some development knowledge will be needed in order to install Isumiro successfully. Here's what's needed:

- A. IDE for mobile and web development, we recommend VSCode
- B. Flutter SDK and JDK with path setup on your local machine
- C. Basic knowledge about Google and Firebase.
- D. Basic knowledge of Google Admob

4. Environment Setup

To run the project, some environment setups are needed:

- A. You'll have to download and install Flutter on your system. You can check the documentation on the following link: [Install | Flutter](#)
- B. You'll need to create an account with Firebase by following this link: [Firebase | Google's Mobile and Web App Development Platform](#)
- C. You'll need to create an account with Google Admob from this link: [Google AdMob - Earn More With Mobile App Monetization](#)
- D. You'll have to download and install Node.js which you can do by following this link: [Node.js \(nodejs.org\)](#)

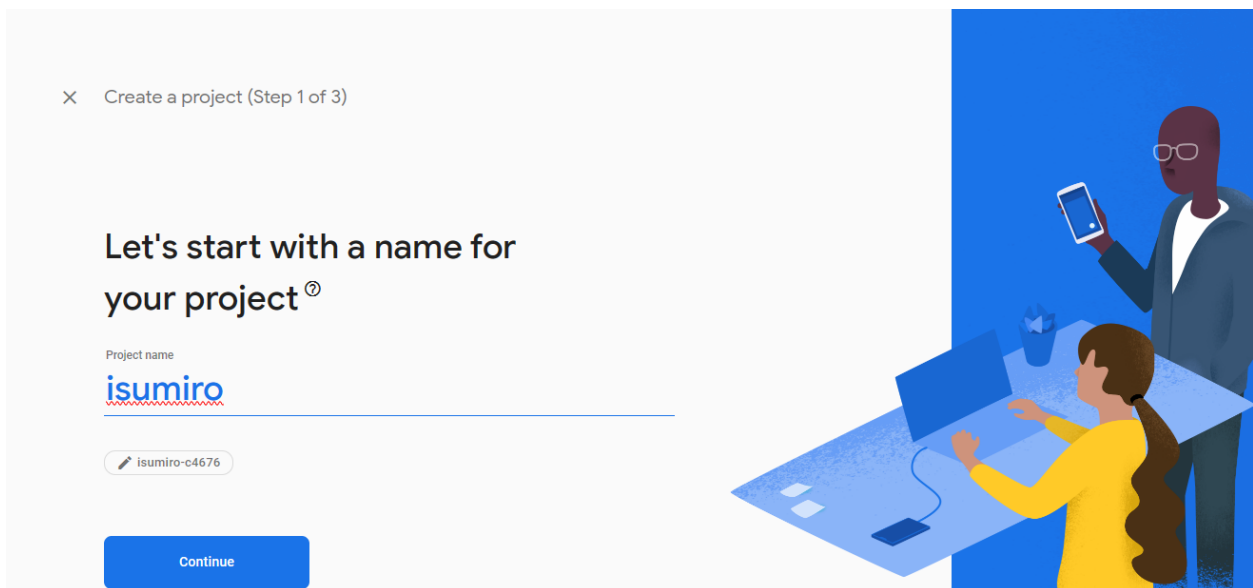
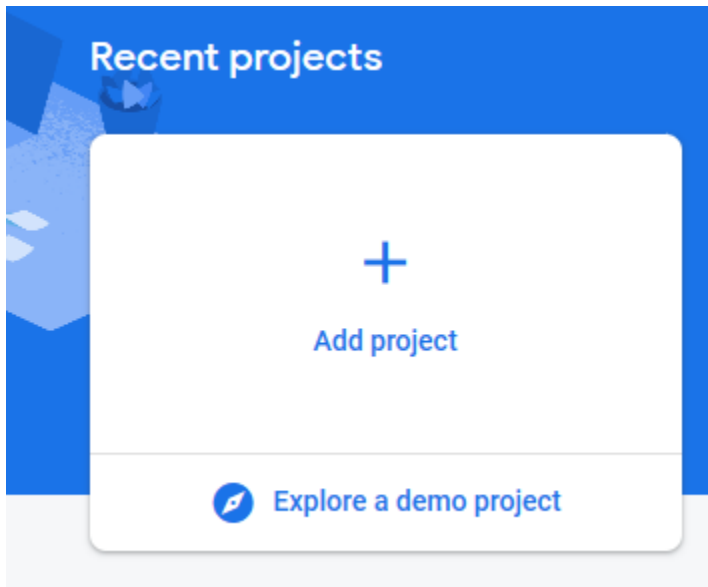
5. Basic Setup

- A. Download the compressed file provided and unzip it on your machine.
- B. Unzip isumiro file containing the Flutter project
- C. Open isumiro folder in VS Code or Android Studio

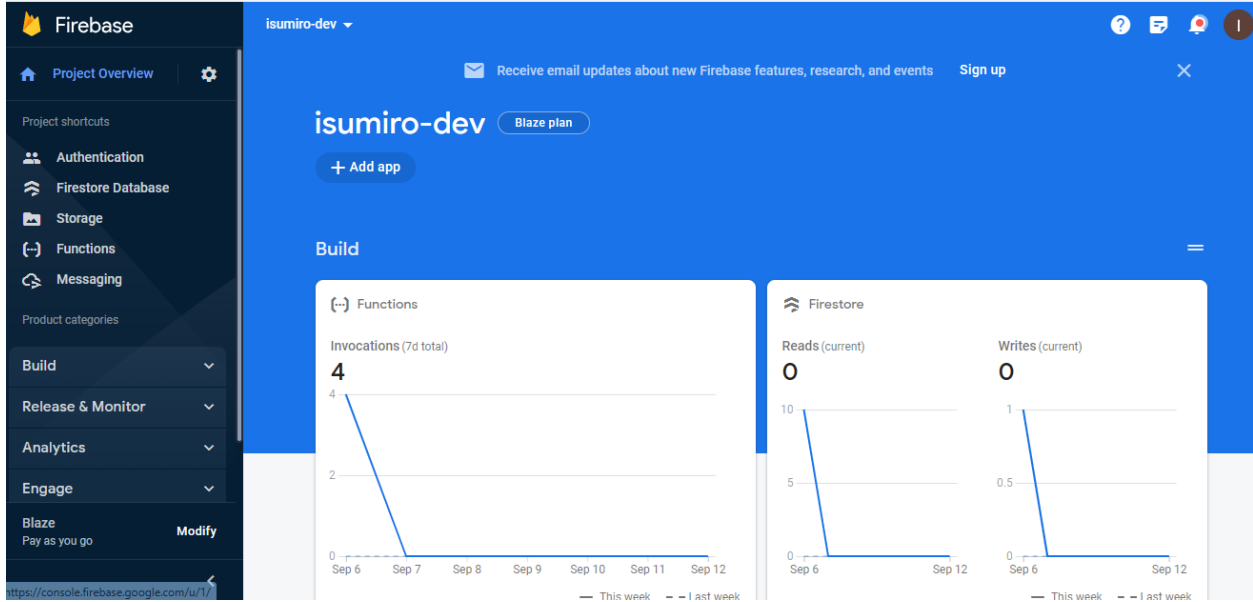
D. Unzip isumiro_functions file which will be needed later

6. Firebase Setup

A. Go on and create a new Firebase project



B. Once created, You'll get to the overview of your Firebase project



C. Upgrade your project from spark to blaze as you'll need Firebase functions. If you don't you won't be able to use cloud functions

Build

Release & Monitor

Analytics

Engage

Blaze
Pay as you go

Modify

isumiro-dev

Blaze plan

+ Add app

Build

(--) Functions

Invocations (7d total)

4

4

2

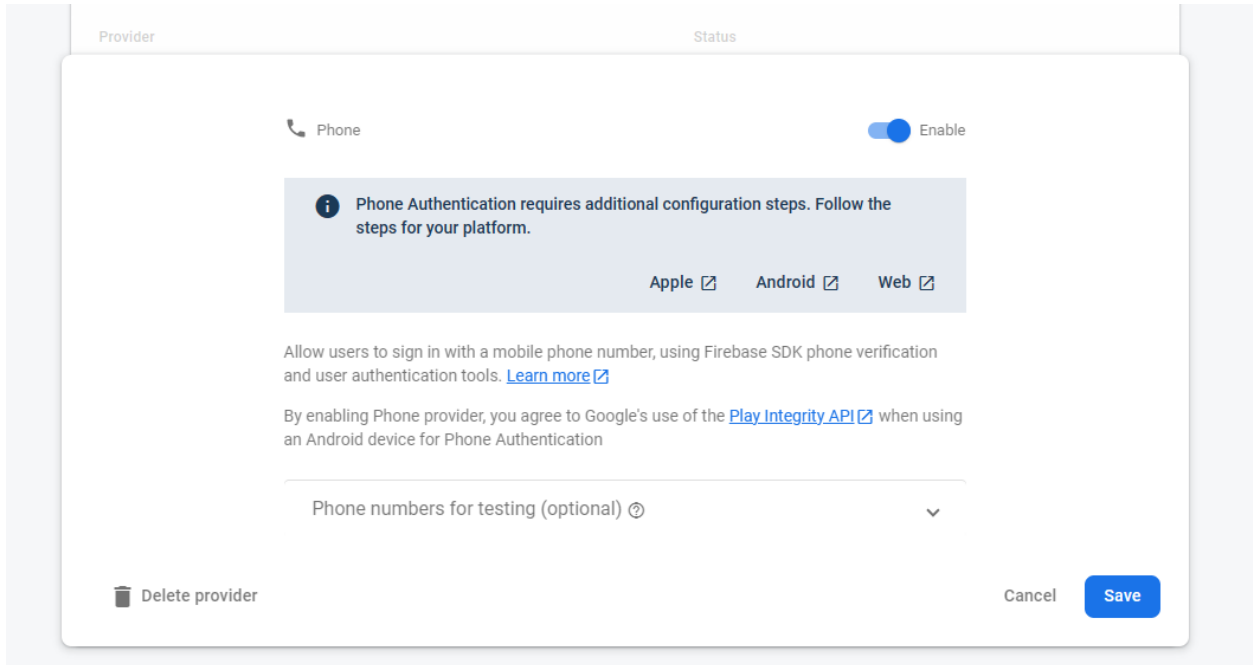
0

Sep 6 Sep 7 Sep 8 Sep 9 Sep 10 Sep 11 Sep 12

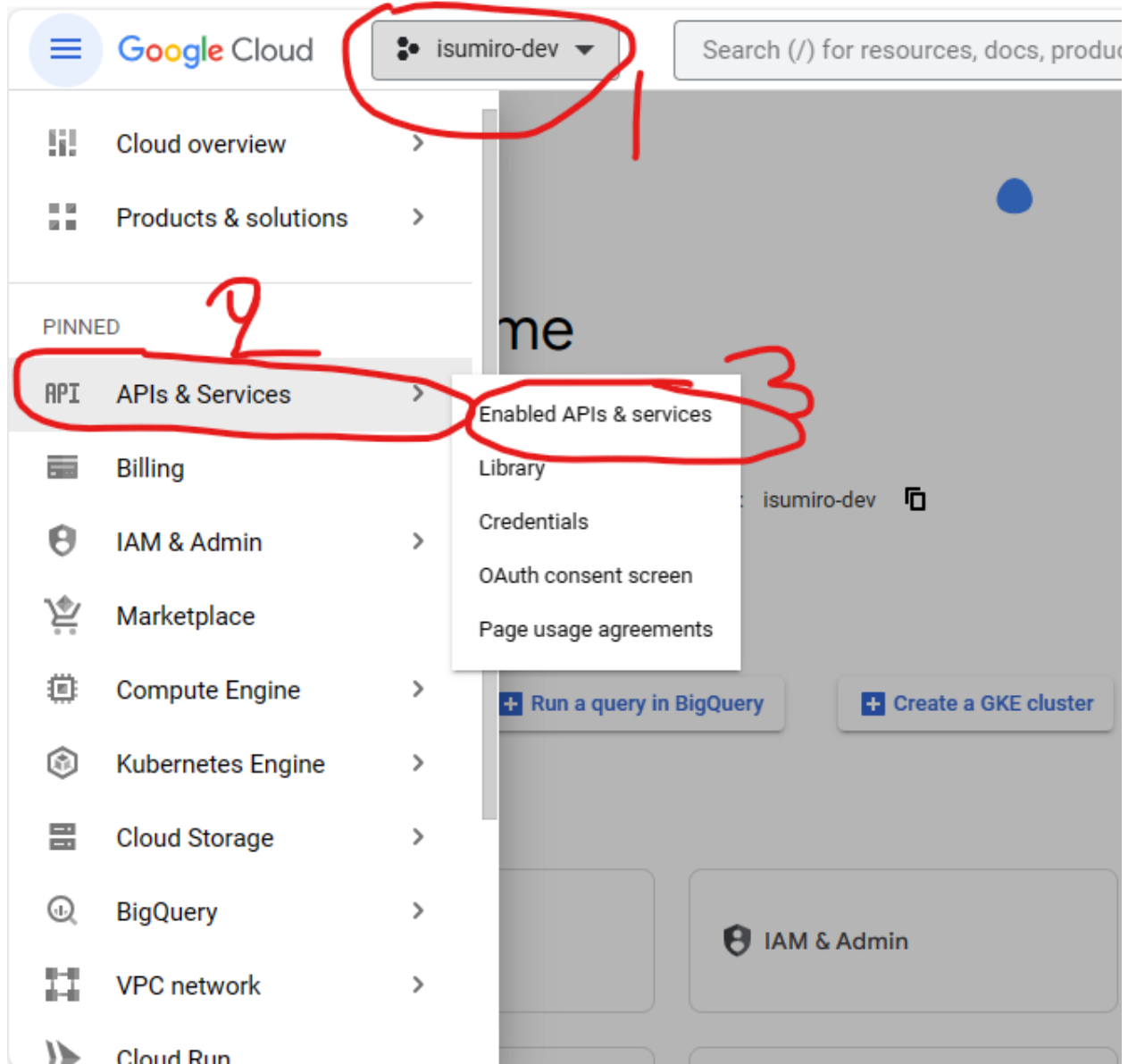
— This week - - Last week

https://console.firebase.google.com/u/1/project/isumiro-dev/authentication/users

D. Head to **Build > Authentication** to get started and enable phone authentication as it's the one used by the app for authentication and click save.



- E. Head to [Google Cloud console](#) and select your project
- F. Click on the navigation menu and select **APIs & services** and then select **enable api and services**



API APIs & Services

APIs & Services **+ ENABLE APIS AND SERVICES**

Enabled APIs & services

- Library
- Credentials
- OAuth consent screen
- Page usage agreements

Traffic

Errors

Median latency

1 hour 6 hours 12 hours ✓ 1 d

https://console.cloud.google.com/?authuser=1&project=isumiro-dev

G. Enable api " Play Integrity API".

Google Cloud isumiro

API API Library

play integrity

API Library > "play integrity"

Filter Type to filter


1 result

Visibility ^

Public (1)

Category ^

Mobile (1)

 **Google Play Integrity API**
Google

The Play Integrity API helps you check that you're interacting with your genuine app on a genuine Android device powered by Google Play services. The Play Integrity API has replaced SafetyNet Attestation and Android Device Verification.



Google Play Integrity API

Google

Check that interactions are coming from your genuine app running on a genuine Android device.

ENABLE

TRY THIS API ↗

OVERVIEW

SUPPORT

RELATED PRODUCTS

Overview

The Play Integrity API helps you check that you're interacting with your genuine app on a genuine Android device powered by Google Play services. The Play Integrity API has replaced SafetyNet Attestation and Android Device Verification.

[Learn more](#) ↗

Additional details

Type: [SaaS & APIs](#)

Last product update: 12/23/22

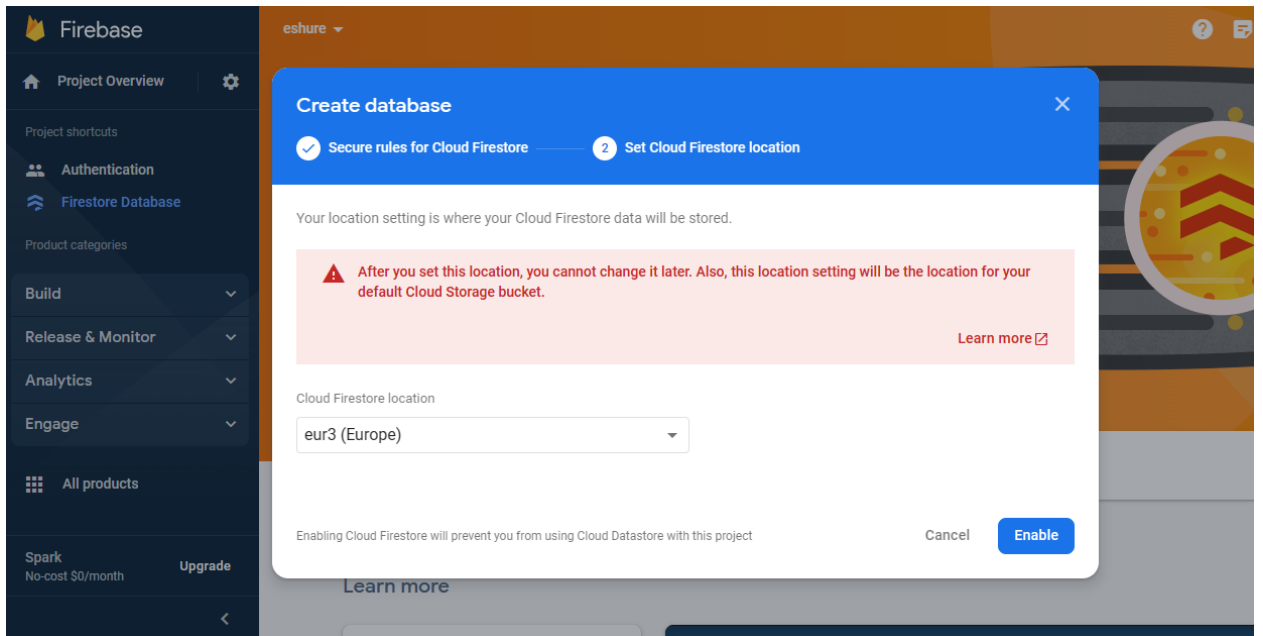
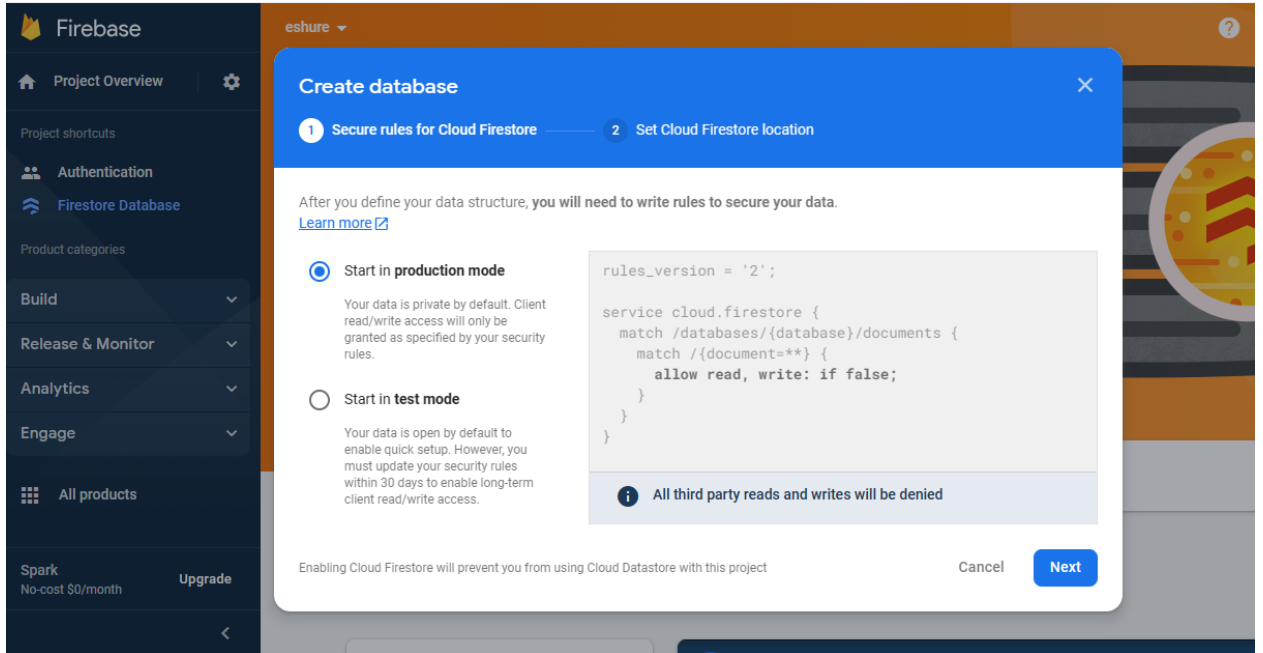
Category: [Mobile](#)

Service name: playintegrity.googleapis.com

Support

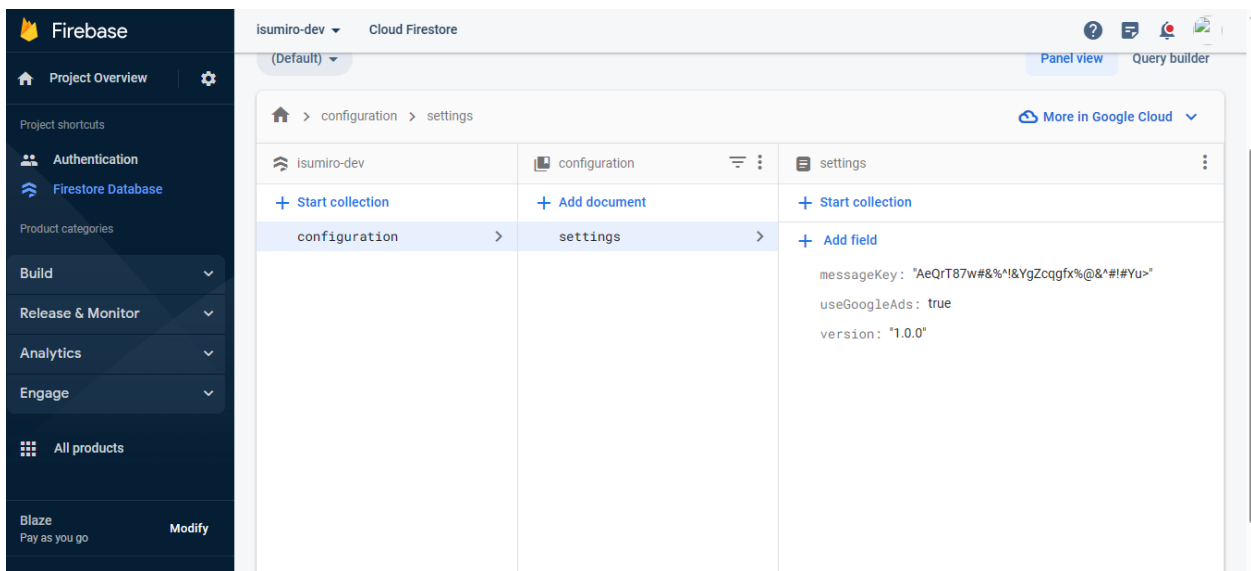
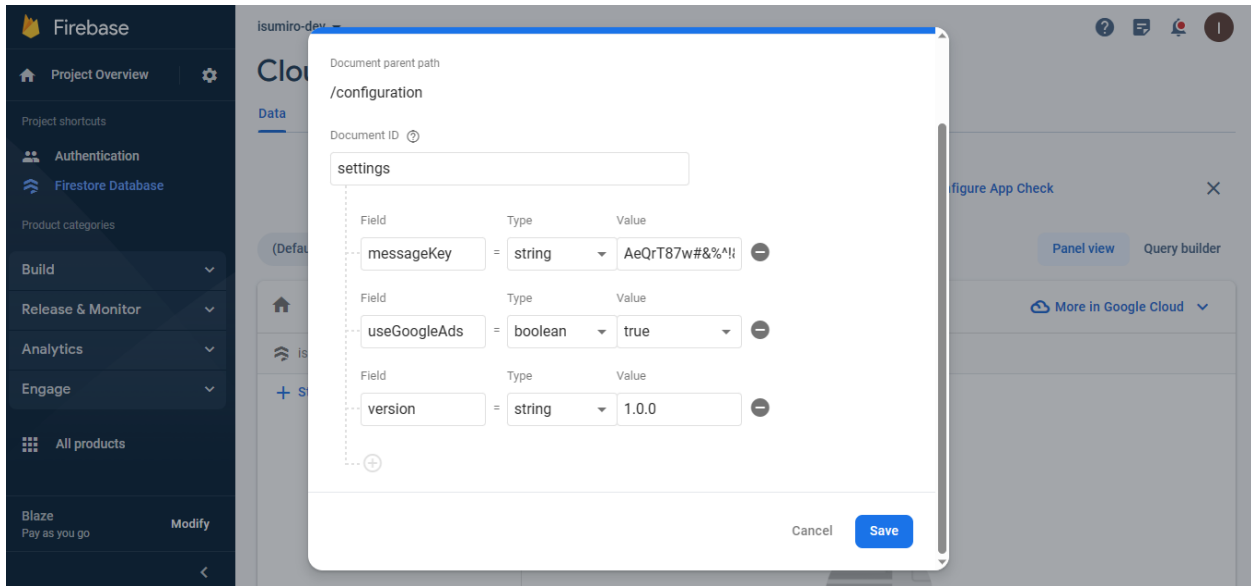
[Learn more](#) ↗

- H. That will make sure that phone authentication works without any issue
- I. Head to **Build > Firestore Database** where you'll need to create a configuration collection and a settings document. Create first your database in production mode and choose your database location.



- J. Then Create a collection called **configuration** and add a document with an ID called **settings**. Inside the settings document, add the following fields:
- messageKey**: a string value that will be used to encrypt chat messages. The string must have 32 characters! Here's an example of a value you can put it:
AeQrT87w#&%^!&YgZcqgfx%&^#!#Yu>

- b. **useGoogleAds**: a bool value that will decide whether to display or not Google ads in the app. Set it to **false** for now.
- c. **version**: a string value that will dictate whether users are forced to upgrade their app, which will happen when the value is greater than the actual value of the user app. For now, input **1.0.0**



K. Head to **Build > Storage** and click Get Started. Start in production mode and set up your storage location.

- L. Now We will configure Firebase functions, Firestore rules and indexes, and Storage rules. Unzip the download file from Envato and unzip the isumiro_functions file. Open the decompressed folder isumiro_functions in VSCode.
 - a. In the terminal run **npm install -g firebase-tools**
 - b. Then run **firebase login** to login into your account
 - c. Run **firebase init** and enter Y when it asks you to override the previous project

```
You're about to initialize a Firebase project in this directory:
C:\Users\Guy\Documents\FILES\PROGRAMMING\Envato\isumiro_functions
Before we get started, keep in mind:
* You are initializing within an existing Firebase project directory
? Are you ready to proceed? (Y/n) Y
```

- d. On the list of features to configure, select, **firestore, functions, and storage** using the space key

```
? Which Firebase features do you want to set up for this directory? Press Space to select features, then Enter
( ) Realtime Database: Configure a security rules file for Realtime Database and (optionally) provision default
(*) Firestore: Configure security rules and indexes files for Firestore
(*) Functions: Configure a Cloud Functions directory and its files
>( ) Hosting: Configure files for Firebase Hosting and (optionally) set up GitHub Action deploys
( ) Hosting: Set up GitHub Action deploys
(*) Storage: Configure a security rules file for Cloud Storage
( ) Emulators: Set up local emulators for Firebase products
(Move up and down to reveal more choices)
```

- e. In the next step select **Use an existing project** and choose the project we previously created in Firebase

```
First, let's associate this project directory with a Firebase project.
You can create multiple project aliases by running firebase use --add,
but for now we'll just set up a default project.
```

```
? Please select an option: (Use arrow keys)
> Use an existing project
Create a new project
Add Firebase to an existing Google Cloud Platform project
Don't set up a default project
```

- f. When asked what to call Firestore rules, just press enter
- g. When asked what to call Firestore indexes, just press enter
- h. When asked to override any of the files, select no and continue

- i. When asked whether to initialize a new codebase or override it, select **Overwrite**, and press enter

```
? What file should be used for Firestore indexes? firestore.indexes.json
=== Functions Setup
Detected existing codebase(s): default
? Would you like to initialize a new codebase, or overwrite an existing one?
  Initialize
> Overwrite
```

- j. Choose JavaScript as the language for cloud functions and press enter

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

? Would you like to initialize a new codebase, or overwrite an existing one? Overwrite
Overwriting codebase default...

? What language would you like to use to write Cloud Functions? (Use arrow keys)
> JavaScript
  TypeScript
  Python
```

- k. Just press enter when it asks to use ESLint
- l. When asked to override **functions/package.json**, Enter N and press enter

```
? Would you like to initialize a new codebase, or overwrite an existing one? Overwrite
Overwriting codebase default...

? What language would you like to use to write Cloud Functions? JavaScript
? Do you want to use ESLint to catch probable bugs and enforce style? No
? File functions/package.json already exists. Overwrite? (y/N) N
```

- m. When asked to override **functions/index.js**, Enter N and press enter

```
Overwriting codebase default...
? What language would you like to use to write Cloud Functions? JavaScript
? Do you want to use ESLint to catch probable bugs and enforce style? No
? File functions/package.json already exists. Overwrite? No
i Skipping write of functions/package.json
? File functions/index.js already exists. Overwrite? (y/N) N
```

- n. When asked to override **functions/.gitignore**, Enter N and press enter
- o. When asked to install dependencies enter Y and press enter and the dependencies will start installing.

```
? Do you want to use ESLint to catch probable bugs and enforce style? No
? File functions/package.json already exists. Overwrite? No
i Skipping write of functions/package.json
? File functions/index.js already exists. Overwrite? No
i Skipping write of functions/index.js
? File functions/.gitignore already exists. Overwrite? No
i Skipping write of functions/.gitignore
? Do you want to install dependencies with npm now? (Y/n) Y
```

- p. When asked what to call **Storage rules**, just press enter
- q. Finally, run **firebase deploy**. You may need to run this command 2 times to be successful

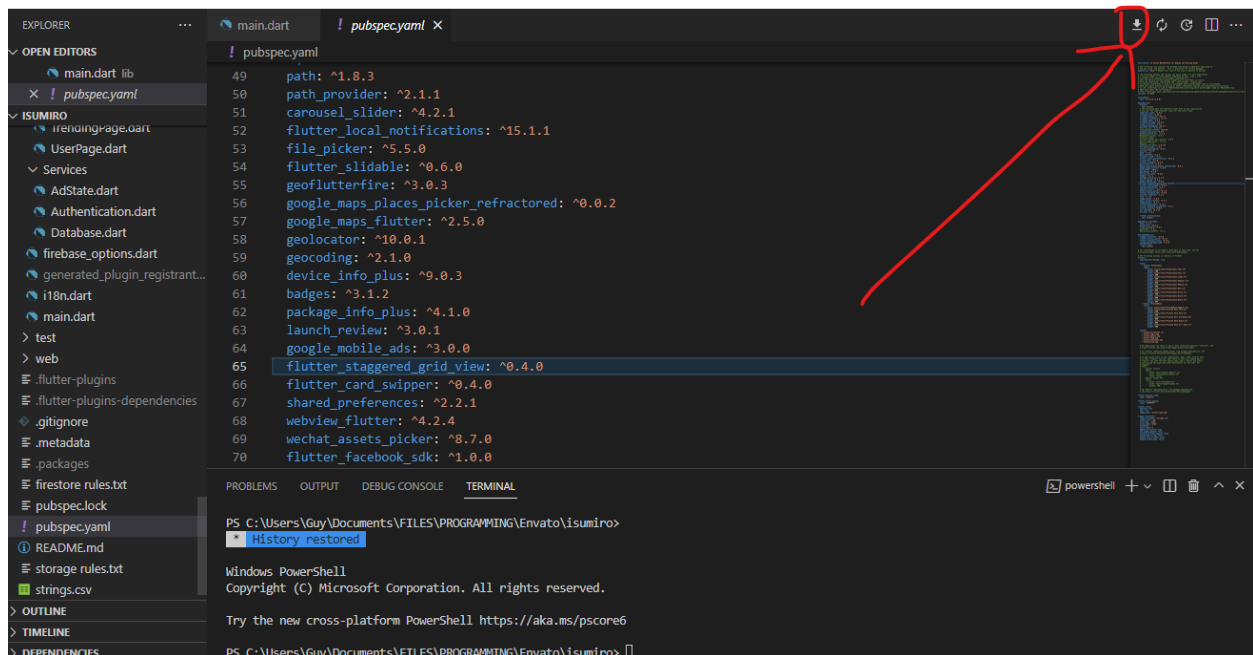
```
i deploying storage, firestore, functions
i firebase.storage: checking storage.rules for compilation errors...
+ firebase.storage: rules file storage.rules compiled successfully
i firestore: reading indexes from firestore.indexes.json...
i cloud.firestore: checking firestore.rules for compilation errors...
+ cloud.firestore: rules file firestore.rules compiled successfully
i functions: preparing codebase default for deployment
i functions: ensuring required API cloudfunctions.googleapis.com is enabled...
```

- M. Head to Functions and click Get Started. You should see the functions we installed in the previous step

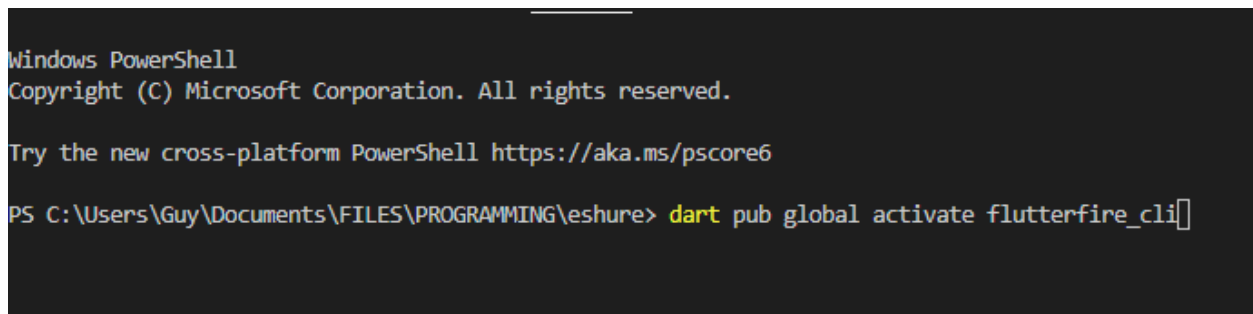
7. Flutter Setup

Now that we've set up our Firebase projects, open the Flutter project in VS Code and follow these steps to link it to Firebase.

A. Open **pubspec.yaml** file and run **flutter pub get** to download all the dependencies



B. Run **dart pub global activate flutterfire_cli** to install the flutterfire cli



- C. Now run **flutterfire configure** . This will create the Firebase files needed to launch in your production environment.

```
* History restored

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

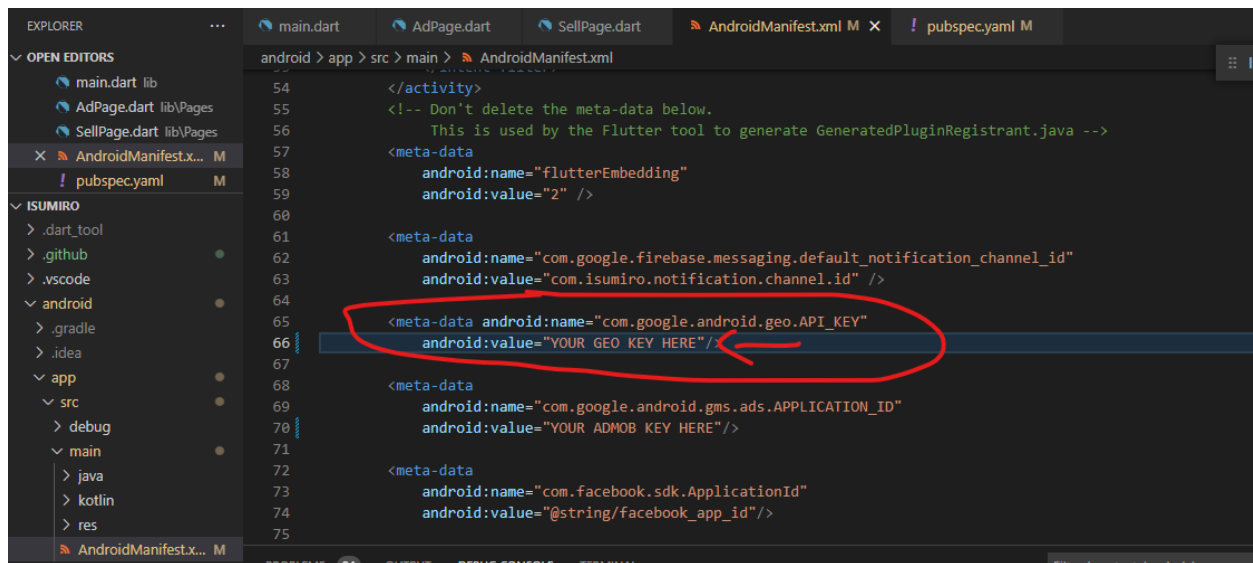
PS C:\Users\Guy\Documents\FILES\PROGRAMMING\Envato\isumiro> flutterfire configure
.: Fetching available Firebase projects...
```

- D. In the next step, select which project to link
- E. Make sure that Android and iOS platforms are only selected and press enter
- F. 3 files will be created: **firebase_options.dart** in **lib** folder, **google-services.json** in **android/app** folder and **firebase_app_id_file.json** in **ios** folder
- G. For the next step to work, You should add Java path to Windows environment variables
- H. Open a terminal and run **keytool -list -v -alias androiddebugkey -keystore %USERPROFILE%\android\debug.keystore** to generate the SHA1 and SH256 of your system. If it doesn't work, make sure java sdk is installed with the path environment well set up. If it still doesn't work, try this:
- Go to this path or wherever you have your keytool.exe file like **C:\Program Files\Java\jre7\bin** for example
 - Hold shift and right click -> then press Open command window here
 - The terminal will pop up, paste then the above keytool command

8. Geolocalisation setup

This app relies heavily on localization as each ad is associated with its location. The location of the user is picked up when the app is open. The app will then try to show relevant ads in a small radius close to the user. If no ads are found, the radius will get larger and larger until ads are found.

- A. Get an API key at [Google Maps Platform - Location and Mapping Solutions](#)
- B. Enable Google Map SDK for each platform.
 - a. Go to [Google Cloud console](#)
 - b. Choose the project that you want to enable Google Maps on. Here it's going to be the Firebase project created earlier
 - c. Select the navigation menu and then select "Google Maps".
 - d. Select "APIs" under the Google Maps menu.
 - e. To enable Google Maps for Android, select "Maps SDK for Android" in the "Additional APIs" section, then select "ENABLE".
 - f. To enable Google Maps for iOS, select "Maps SDK for iOS" in the "Additional APIs" section, then select "ENABLE".
 - g. Make sure the APIs you enabled are under the "Enabled APIs" section.
- C. For more details, see [Getting started with Google Maps Platform | Google for Developers](#).
- D. On Android. Specify your API key in the application manifest **android/app/src/main/AndroidManifest.xml**



```
54 </activity>
55 <!-- Don't delete the meta-data below.
56      This is used by the Flutter tool to generate GeneratedPluginRegistrant.java -->
57 <meta-data
58     android:name="FlutterEmbedding"
59     android:value="2" />
60
61 <meta-data
62     android:name="com.google.firebase.messaging.default_notification_channel_id"
63     android:value="com.isumiro.notification.channel.id" />
64
65 <meta-data android:name="com.google.android.geo.API_KEY"
66     android:value="YOUR GEO KEY HERE" />
67
68 <meta-data
69     android:name="com.google.android.gms.ads.APPLICATION_ID"
70     android:value="YOUR ADMOB KEY HERE"/>
71
72 <meta-data
73     android:name="com.facebook.sdk.ApplicationId"
74     android:value="@string/facebook_app_id"/>
75
```

- E. On iOS, specify your API key in the application delegate ios/Runner/AppDelegate.swift

```
4
5 @UIApplicationMain
6 @objc class AppDelegate: FlutterAppDelegate {
7   override func application(
8     _ application: UIApplication,
9     didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?
10  ) -> Bool {
11    GeneratedPluginRegistrant.register(with: self)
12    GMServices.provideAPIKey("YOUR GEO KEY")
13    FlutterDownloaderPlugin.setPluginRegistrantCallback(registerPlugins)
14    if #available(iOS 10.0, *) {
15      UNUserNotificationCenter.current().delegate = self as? UNUserNotificationCenterDelegate
16    }
17    return super.application(application, didFinishLaunchingWithOptions: launchOptions)
18  }
19
20 private func registerPlugins(registry: FlutterPluginRegistry) {
21   if (!registry.hasPlugin("FlutterDownloaderPlugin")) {
22     FlutterDownloaderPlugin.register(with: registry.registrar(forPlugin: "FlutterDownloaderPlugin"))
23   }
24 }
```

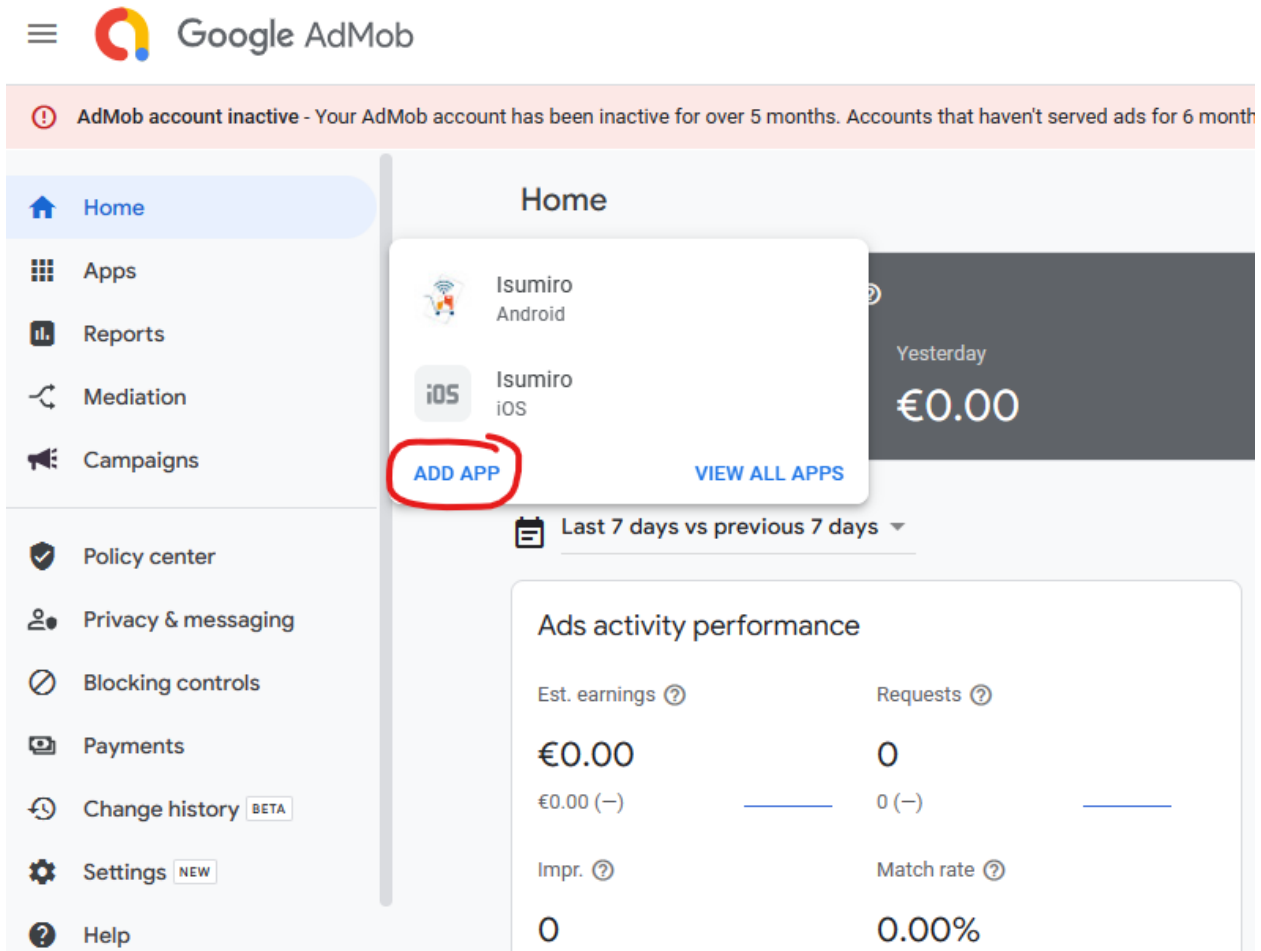
F. Go to **lib/Helpers/Utils.dart** and locate a function called **getMapsKey()**. Insert your key there too.

```
219 return text;
220 }
221
222 String getLanguage(BuildContext context) {
223   String language = Localizations.localeOf(context).languageCode;
224   return language;
225 }
226
227 String? getMapsKey() {
228   String? key;
229   if (Platform.isAndroid) {
230     key = "YOUR GEO KEY";
231   } else if (Platform.isIOS) {
232     key = "YOUR GEO KEY";
233   }
234
235   return key;
236 }
237
238 Future<List<Currency>> getCurrencies() async {
239   List<Currency> list = [];
240
241   String text = await rootBundle.loadString('assets/currencies.csv');
```

9. AdMob setup

This document just covers how to link your Flutter project to Admob as it's up to you where and what kind of banner you want to show. If you don't want to show Google Banner you can simply set the useGoogleAds field in the settings document in the configuration collection to false.

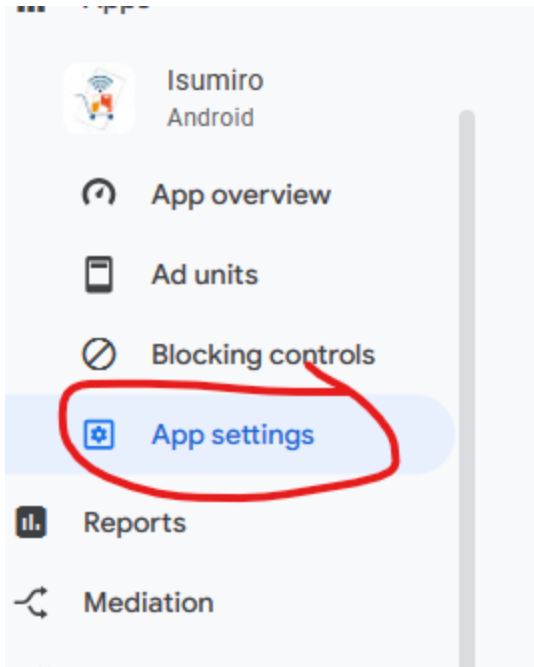
- A. Head to [Google AdMob - Earn More With Mobile App Monetization](#) and create an Account
- B. Click on Apps in the side menu and create an Android app and an iOS App



The screenshot shows the Google AdMob dashboard. At the top, there is a notification: "AdMob account inactive - Your AdMob account has been inactive for over 5 months. Accounts that haven't served ads for 6 months". Below this is a navigation sidebar with options: Home, Apps, Reports, Mediation, Campaigns, Policy center, Privacy & messaging, Blocking controls, Payments, Change history (BETA), Settings (NEW), and Help. The main content area is titled "Home" and displays a summary for two apps: "Isumiro Android" and "Isumiro iOS". A red circle highlights the "ADD APP" button. To the right, there is a performance summary for "Yesterday" showing "€0.00". Below this is a section for "Last 7 days vs previous 7 days" with a dropdown menu. The "Ads activity performance" section shows a table with the following data:

Est. earnings	Requests
€0.00	0
€0.00 (-)	0 (-)
Impr.	Match rate
0	0.00%

- C. Click on the Android app and go to app settings. There you should see the App ID. Copy it



D. On Android, Specify your API key in the application manifest
android/app/src/main/AndroidManifest.xml

A screenshot of an IDE showing the AndroidManifest.xml file. The file is open in the editor, and the following XML code is visible:

```
56 <!-- Don't delete the meta-data below.
57      This is used by the Flutter tool to generate GeneratedPlug
58 <meta-data
59     android:name="flutterEmbedding"
60     android:value="2" />
61
62 <meta-data
63     android:name="com.google.firebase.messaging.default_notific
64     android:value="com.isumiro.notification.channel.id" />
65
66 <meta-data android:name="com.google.android.geo.API_KEY"
67     android:value="YOUR GEO KEY"/>
68
69 <meta-data
70     android:name="com.google.android.gms.ads.APPLICATION_ID"
71     android:value="YOUR ADMOB KEY"/>
72
73 <meta-data
74     android:name="com.facebook.sdk.ApplicationId"
75     android:value="@string/facebook_app_id"/>
76
77 <provider
```

The meta-data entry for the AdMob application ID (lines 69-71) is highlighted with a red circle.

- E. Click on the iOS app and go to app settings. There you should see the App ID. Copy it
- F. In your app's **ios/Runner/Info.plist** file, add a **GADApplicationIdentifier** key with a string value of your AdMob app ID, as identified in the AdMob web interface:

```
ios > Runner > Info.plist
56 </array>
57 </dict>
58 </array>
59 <key>FacebookAppID</key>
60 <string>375012400917680</string>
61 <key>FacebookDisplayName</key>
62 <string>Isumiro</string>
63 <key>FacebookAutoLogAppEventsEnabled</key>
64 <true/>
65 <key>FacebookAdvertiserIDCollectionEnabled</key>
66 <true/>
67 <key>GADApplicationIdentifier</key>
68 <string>YOUR_ADMOB_KEY</string>
69 <key>SKAdNetworkItems</key>
70 <array>
71 <dict>
72 <key>SKAdNetworkIdentifier</key>
73 <string>cstr6suwn9.skadnetwork</string>
74 </dict>
75 </array>
76 <key>UISupportedInterfaceOrientations</key>
77 <array>
78 <string>UIInterfaceOrientationPortrait</string>
```

- G. To display banners, you'll have to create Ad units and add corresponding unit IDs wherever a banner has been placed in the app or by adding your own banner.
- H. If you want to know more about how to configure Admob, check this link: [Get started | Flutter | Google for Developers](#)

10. Other Installation

- You can change your app package name by using a package named [change_app_package_name | Dart Package \(pub.dev\)](#) which will make the job easier for you. All you have to run is **flutter pub run change_app_package_name:main com.new.package.name** where **com.new.package.name** is your new package name. Note however that after doing this you will have to redo the flutter setup done earlier

```
PS C:\Users\Guy\Documents\FILES\PROGRAMMING\eshure> flutter pub run change_app_package_name:main com.new.package.name
```

- You can upload your own logo in the assets folder and use [flutter_launcher_icons | Dart Package \(pub.dev\)](#) package to change the icons all over the app.
- [flutter_app_name | Dart Package \(pub.dev\)](#) package will help you to change the name of the app as well and [flutter_native_splash | Flutter Package \(pub.dev\)](#) to change the splash screen.
- If you wish to use **GitHub CI/CD** for deployment, you'll need to configure the **android-main-release.yaml** file in the **.github/workflows** folder to create releases on Github. Check this post for more info: [Deploy your Flutter App to Firebase App Distribution using GitHub Actions - Android \(bernos.dev\)](#)