



# Documentation

**Author :** Active IT zone

**Software Framework :** Flutter

**Addon For:** Active eCommerce CMS

**Provided by :** codecanyon



## **Documentation**

1. What are the prerequisites?
2. How to run Flutter Application in Android Studio?
3. How to configure the App according to your setup?
4. How to change the package name?
5. How to build the App for testing (build and apk) ?
6. How to generate play store uploadable files for release?
7. How to generate app store uploadable files? (This section will be available soon)
8. How to Update for Android?
9. How to configure social login?
10. How to configure push notification?
11. How to configure google map ?

## 1. What are the prerequisites?

This Flutter app can be hosted into Google Play Store + Apple Appstore as your branded eCommerce CMS app. The app will communicate with your hosted eCommerce CMS web application through APIs. That means the prerequisite to publish the eCommerce Mobile application is to have the eCommerce CMS Web application in the latest version always.

Flutter version must be : [Flutter 1.22.4 • channel stable](#)

Dart version must be : [Dart 2.10.4](#)

Make sure your flutter and dart versions are correct. Follow the flutter documentation from <https://flutter.dev/docs/get-started/install> to install the given version of flutter in your pc/mac.

## 2. How to run Android Application in Android Studio?

- Install Android Studio from <https://developer.android.com/studio>
- Extract the source\_code.zip. You will find this inside the main zip.
- Open the folder in your android studio.
- Even if you are building an app for ios, use android studio for the build.
- Then in your android studio terminal run:

`flutter pub get` \*\* You need this to get all 3rd party packages from pub.dev

## 3. How to configure the App according to your setup?

### 1. App Config:

This helps you connect your app to your server.

Open lib/app\_config.dart

You can change the [copyright\\_text](#), [app\\_name](#), [purchase\\_code](#), [HTTPS](#), [DOMAIN\\_PATH](#) variable.

Do not change the other variables.

Make sure that `purchase_code` is given. Otherwise your app will not work properly.

If your site does not have https or your are using a local machine as server (localhost) the make `HTTPS = false`;

Your `DOMAIN_PATH` is your site url without any protocol. (see screenshot below)

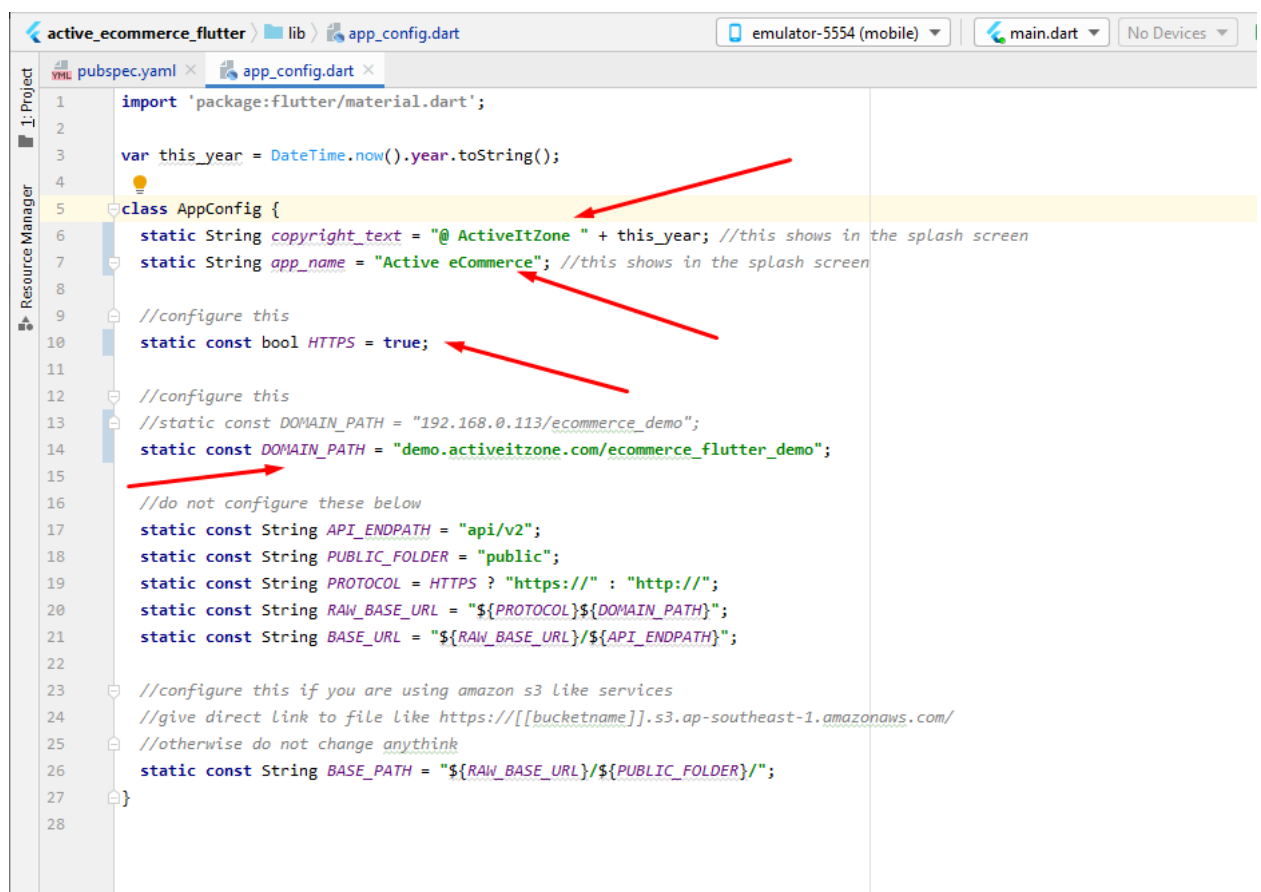
If you are using localhost , `DOMAIN_PATH` should be "`your_ip_address/your_project`";

**\*\* "localhost/your\_project" will not not work \*\***

Normally you do not have to change the `BASE_PATH`. Keep it as given.

But if you are using s3 for image uploading your `BASE_PATH` should be :

`BASE_PATH = "https://[[bucketname]].s3. [[regeion]].amazonaws.com/";`



```
1 import 'package:flutter/material.dart';
2
3 var this_year = DateTime.now().year.toString();
4
5 class AppConfig {
6   static String copyright_text = "@ ActiveItZone " + this_year; //this shows in the splash screen
7   static String app_name = "Active eCommerce"; //this shows in the splash screen
8
9   //configure this
10  static const bool HTTPS = true;
11
12  //configure this
13  //static const DOMAIN_PATH = "192.168.0.113/ecommerce_demo";
14  static const DOMAIN_PATH = "demo.activeitzone.com/ecommerce_flutter_demo";
15
16  //do not configure these below
17  static const String API_ENDPATH = "api/v2";
18  static const String PUBLIC_FOLDER = "public";
19  static const String PROTOCOL = HTTPS ? "https://" : "http://";
20  static const String RAW_BASE_URL = "${PROTOCOL}${DOMAIN_PATH}";
21  static const String BASE_URL = "${RAW_BASE_URL}/${API_ENDPATH}";
22
23  //configure this if you are using amazon s3 like services
24  //give direct link to file like https://[[bucketname]].s3.ap-southeast-1.amazonaws.com/
25  //otherwise do not change anything
26  static const String BASE_PATH = "${RAW_BASE_URL}/${PUBLIC_FOLDER}/";
27 }
28
```

The screenshot shows the `app_config.dart` file in an IDE. Red arrows point to the following lines: line 5 (class definition), line 6 (copyright\_text), line 7 (app\_name), line 10 (HTTPS), line 14 (DOMAIN\_PATH), and line 26 (BASE\_PATH).

## 2. Theme Config:

This helps you change your app's colors according to your theme/branding

Open `lib/my_theme.dart`

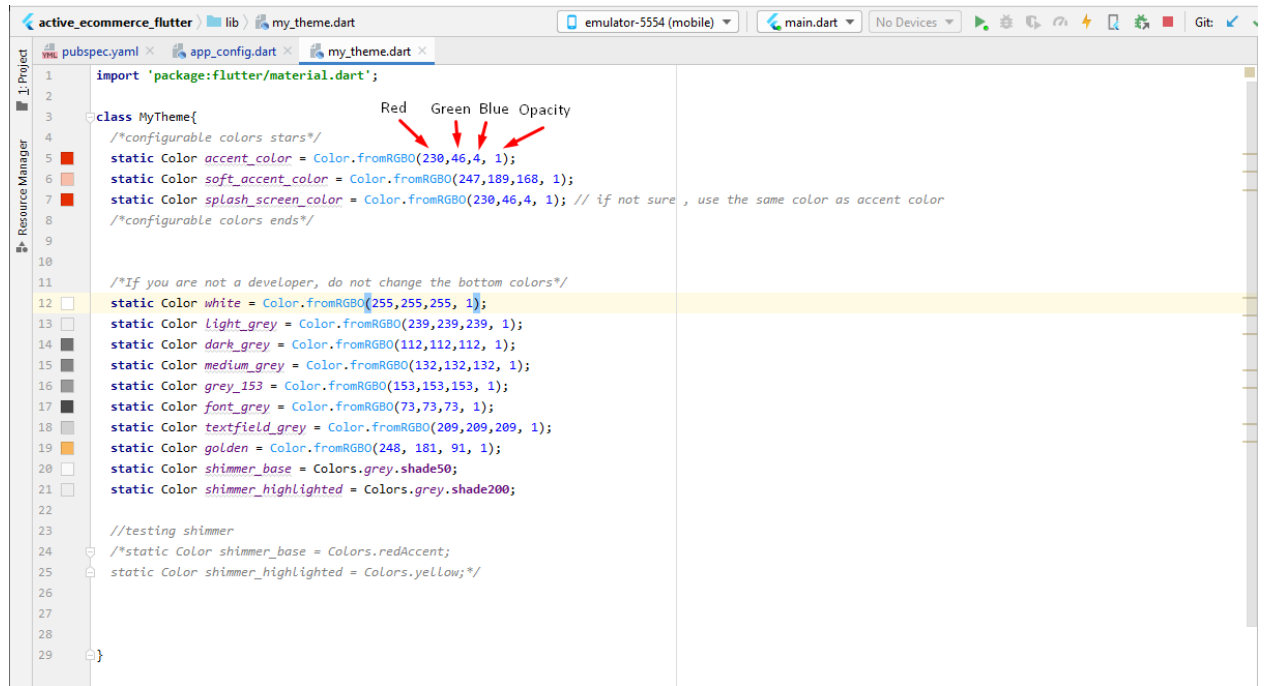
You can change the `accent_color`, `soft_accent_color`, `splash_screen_color` variable.

Flutter by default does not support hex color. Do not change the other variables.

Use <https://www.rapidtables.com/convert/color/hex-to-rgb.html> To get the RGB value if you do not already know your theme's RGB color.

You should keep the Opacity value 1 (Opacity can be 0, 0.1, 0.2, ..... ,0.9 ,1)

See the screenshot below.



### 3. Configure the launcher icon:

This helps you change your app's launcher icon.

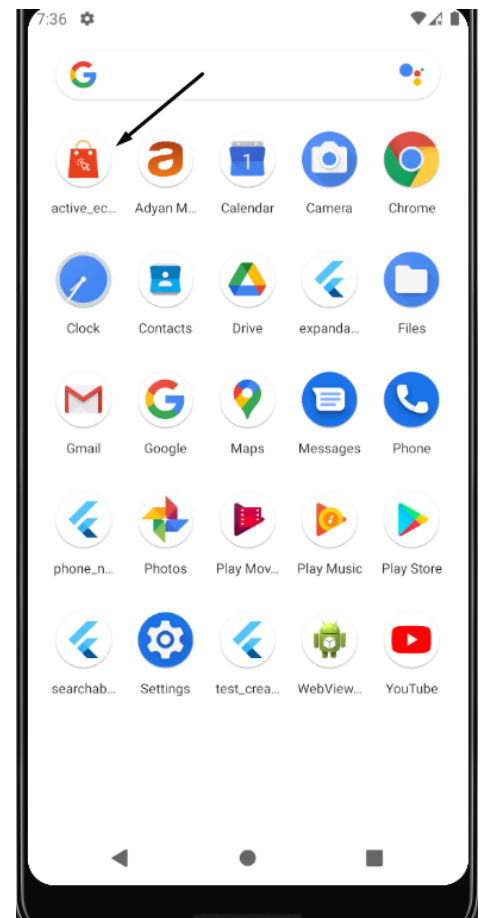
Change the `app_logo.png` in `assets` folder with your own logo. Your file name should also be `app_logo.png` and it should be a `512x512 png` image and the image format should be the same.

After replacing the file , **uninstall** your app from your emulator. Otherwise the logo will not be changed.

Then in your android studio terminal run:

`flutter pub get`

Then run :

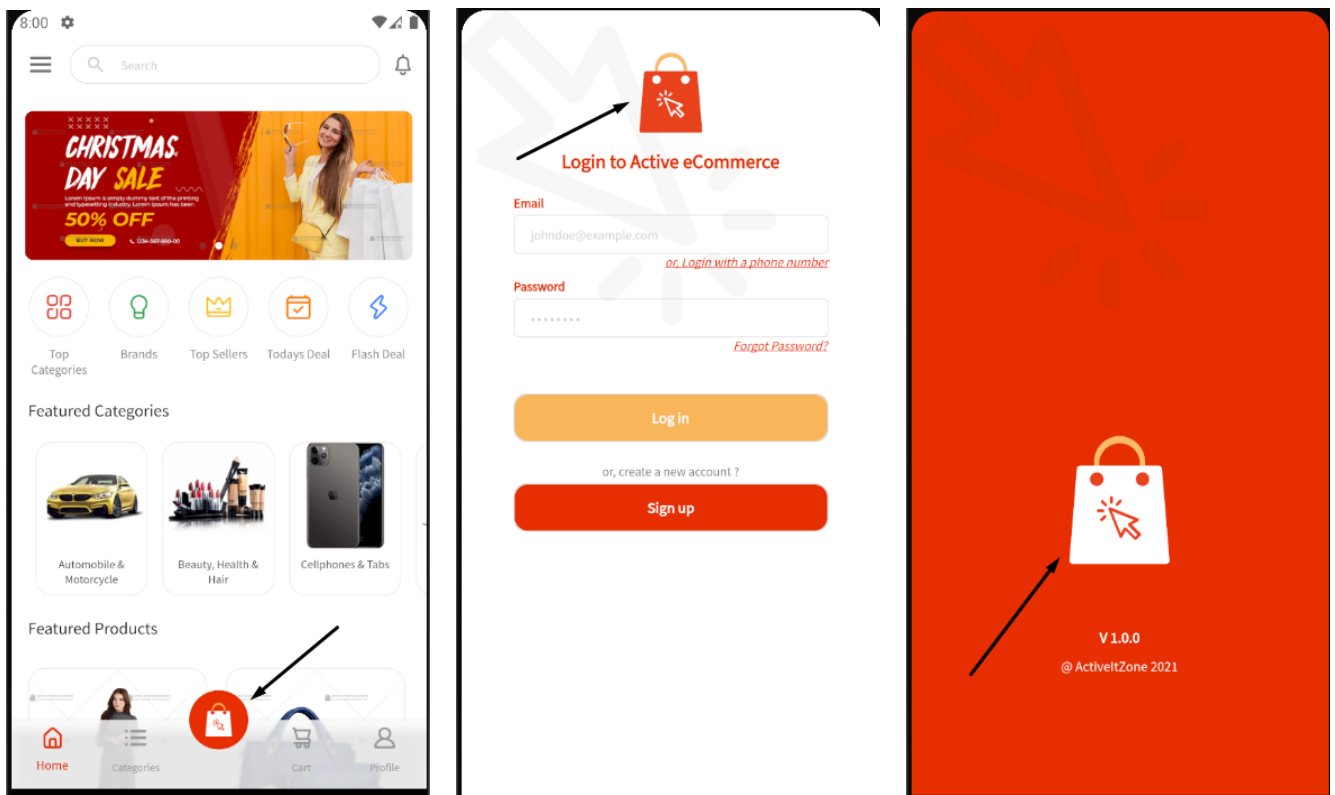


`flutter pub run flutter_launcher_icons:main`

Then run your app (shift +10). The app will be installed again with your given launcher icon.

#### 4. Configure other logos:

In the asset folders we have other logos that you may want to change according to your branding.



This logos will be found in :

`assets/square_logo.png (50x64)`

`assets/login_registration_form_logo.png (512x512)`

`assets/splash_screen_logo.png (512x512)`

Change this logo with your own logo. File name , image format and size should be the same for each logo.

Then in your android studio terminal run:

`flutter pub get`

Then restart your app (shift +10). You should see your own logo in these places.

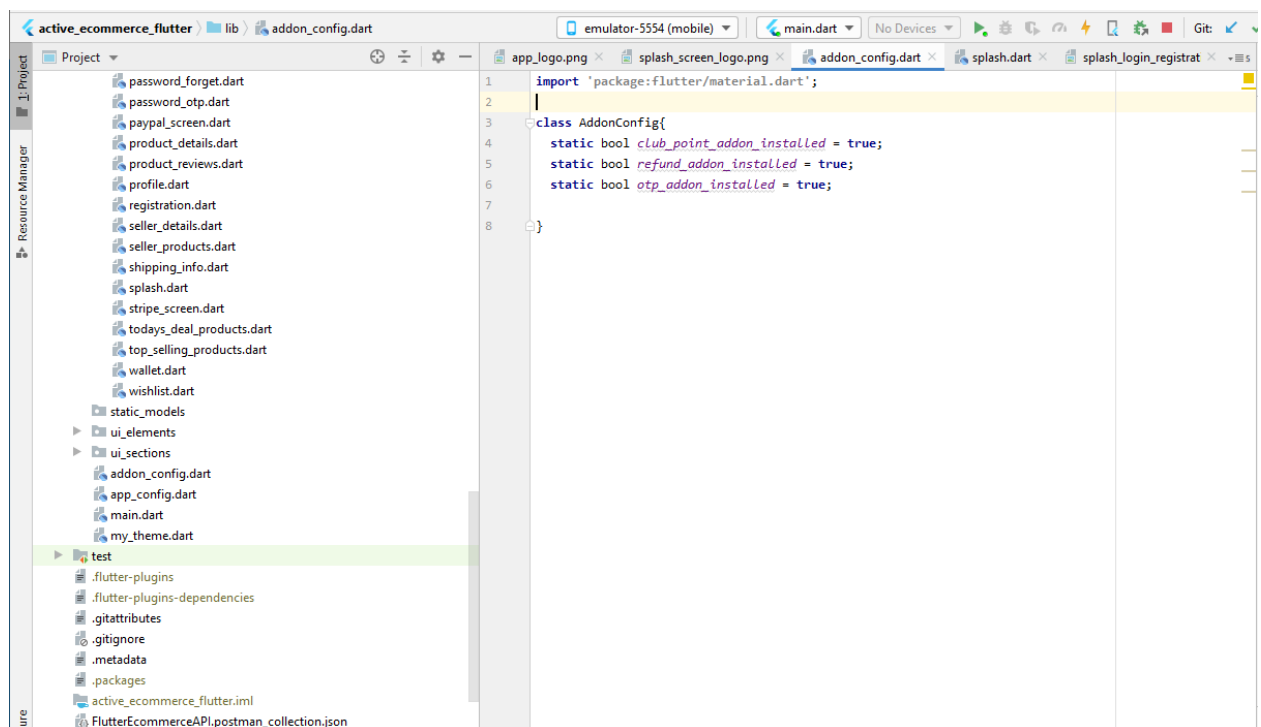
#### 5. Addon Config:

This helps you with certain sections in app that shows according to the addons

Open lib/addon\_config.dart

You can change the `club_point_addon_installed`, `refund_addon_installed`, `otp_addon_installed` variable.

For each variable : if you have the installed the related addon in your server : make it `true`, otherwise make it `false`



#### 4. How to change the package name ?

This is very important. Your app cannot have the same package name as other app. If it does, the playstore will not accept it as an unique application. So rename your app according to your business/brand name. Try to write an unique package name.

Naming convention : <https://docs.oracle.com/javase/tutorial/java/package/namingpkgs.html>

For example

Let's say your package is : `com.onatcipli.networkUpp`

And your app name is `"Network Upp"`

Then ,

Run this command inside your flutter project root.

Run the command in android studio terminal :

```
flutter pub run rename --bundleId com.onatcipli.networkUpp
```

```
flutter pub run rename --appName "Network Upp"
```

Try uninstalling the app from the emulator , then run the commands and then restart the app.

If it does not work, first uninstall, then restart the app then run the commands.

**\*\*In case the above do not work:**

### **In Android**

for **package name** just change in build build.gradle only (andddroid/app/build.gradle)

```
defaultConfig {  
    applicationId "bundleId com.onatcipli.networkUpp"  
    .....  
}
```

### **For iOS**

Change the bundle identifier from your Info.plist file inside your ios/Runner directory.

```
<key>CFBundleIdentifier</key>  
<string>bundleId com.onatcipli.networkUpp</string>
```

If you face issues consult a flutter developer.



#### 4. How to Build the app for testing (build an apk) ?

<https://flutter.dev/docs/deployment/android> see the doc for reference

In terminal run : `flutter build apk`

It will build an apk and show the folder. You can then install it in your phone to test, or share to multiple users for testing .

#### 5. How to generate play store uploadable files for release?

<https://flutter.dev/docs/deployment/android> see the doc for reference

Signing the app:

To publish on the Play Store, you need to give your app a digital signature. Use the following instructions to sign your app.

Go through the screenshots below carefully to understand how to generate key and use it for the released signed app:

##### **Note:**

- The `keytool` command might not be in your path—it's part of Java, which is installed as part of Android Studio. For the concrete path, run `flutter doctor -v` and locate the path printed after 'Java binary at:'. Then use that fully qualified path replacing `java` (at the end) with `keytool`. If your path includes space-separated names, such as `Program Files`, use platform-appropriate notation for the names. For example, on Mac/Linux use `Program\ Files`, and on Windows use `"Program Files"`.
- The `-storetype JKS` tag is only required for Java 9 or newer. As of the Java 9 release, the keystore type defaults to `PKS12`.

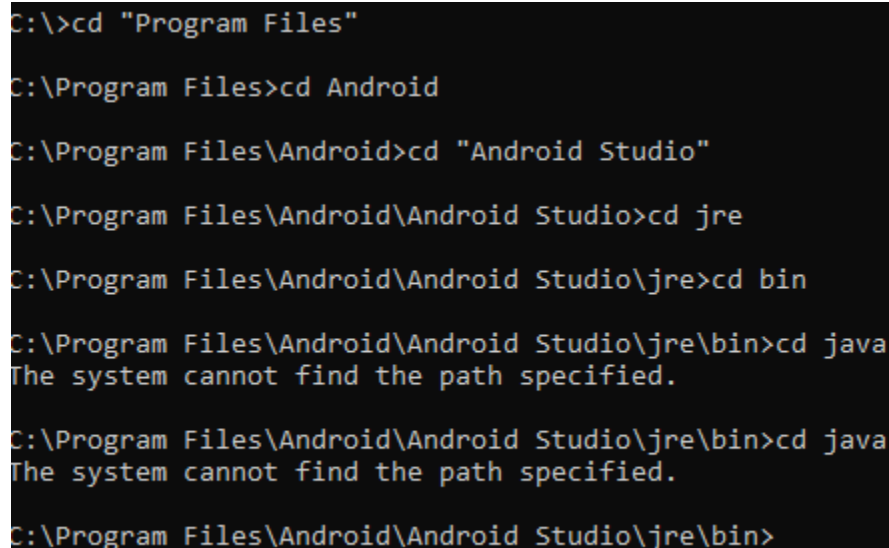
```
C:\flutter_projects\active_ecommerce_flutter>flutter doctor -v
[v] Flutter (Channel stable, 1.22.4, on Microsoft Windows [Version 10.0.19041.867], locale en-US)
    • Flutter version 1.22.4 at C:\flutter
    • Framework revision 1aafb3a8b9 (5 months ago), 2020-11-13 09:59:28 -0800
    • Engine revision 2c956a31c0
    • Dart version 2.10.4

[v] Android toolchain - develop for Android devices (Android SDK version 30.0.1)
    • Android SDK at C:\Users\User\AppData\Local\Android\sdk
    • Platform android-30, build-tools 30.0.1
    • Java binary at: C:\Program Files\Android\Android Studio\jre\bin\java
    • Java version OpenJDK Runtime Environment (build 1.8.0_242-release-1644-b01)
    • All Android licenses accepted.

[v] Android Studio (version 4.0)
    • Android Studio at C:\Program Files\Android\Android Studio
    • Flutter plugin installed
    • Dart plugin version 193.7547
    • Java version OpenJDK Runtime Environment (build 1.8.0_242-release-1644-b01)

[v] VS Code (version 1.53.2)
```

Find binary path



```
C:\>cd "Program Files"
C:\Program Files>cd Android
C:\Program Files\Android>cd "Android Studio"
C:\Program Files\Android\Android Studio>cd jre
C:\Program Files\Android\Android Studio\jre>cd bin
C:\Program Files\Android\Android Studio\jre\bin>cd java
The system cannot find the path specified.
C:\Program Files\Android\Android Studio\jre\bin>cd java
The system cannot find the path specified.
C:\Program Files\Android\Android Studio\jre\bin>
```

Then generate and store the key (image on next page)



```
Microsoft Windows [version 10.0.19041.807]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\User>cd ..
C:\Users>cd ..
C:\>cd "Program Files"
C:\Program Files>cd Android
C:\Program Files\Android>cd "Android Studio"
C:\Program Files\Android\Android Studio>cd jre
C:\Program Files\Android\Android Studio\jre>cd bin
C:\Program Files\Android\Android Studio\jre\bin>cd java
The system cannot find the path specified.
C:\Program Files\Android\Android Studio\jre\bin>cd java
The system cannot find the path specified.
C:\Program Files\Android\Android Studio\jre\bin>keytool.exe -genkey -v -keystore c:\flutter_projects\active_ecommerce_flutter\key.jks -storetype JKS -keyalg RSA -keysize 2048 -validity 10000 -alias key
Enter keystore password:
Re-enter new password:
What is your first and last name?
[Unknown]: Test
What is the name of your organizational unit?
[Unknown]: Test
What is the name of your organization?
[Unknown]: Test
What is the name of your City or Locality?
[Unknown]: Test
What is the name of your State or Province?
[Unknown]: Test
What is the two-letter country code for this unit?
[Unknown]: us
Is CN=Test, OU=Test, O=Test, L=Test, ST=Test, C=us correct?
[no]: yes
Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days
for: CN=Test, OU=Test, O=Test, L=Test, ST=Test, C=us
Enter key password for <key>
(RETURN if same as keystore password):
Re-enter new password:
[Storing c:\flutter_projects\active_ecommerce_flutter\key.jks]
```

active\_ecommerce\_flutter

	Name	Date modified	Type	Size
	.dart_tool	4/1/2021 5:58 PM	File folder	
	.git	4/1/2021 7:41 PM	File folder	
	.idea	4/1/2021 9:44 PM	File folder	
	android	3/28/2021 9:59 PM	File folder	
	assets	4/1/2021 5:53 PM	File folder	
	build	4/1/2021 9:01 PM	File folder	
	dummy_assets	3/31/2021 8:41 PM	File folder	
	ios	11/29/2020 7:19 PM	File folder	
	lib	4/1/2021 5:29 PM	File folder	
	test	11/29/2020 7:19 PM	File folder	
	.flutter-plugins	4/1/2021 9:00 PM	FLUTTER-PLUGINS...	3 KB
	.flutter-plugins-dependencies	4/1/2021 9:00 PM	FLUTTER-PLUGINS...	8 KB
	.gitattributes	11/29/2020 9:09 PM	Text Document	1 KB
	.gitignore	11/29/2020 7:19 PM	Git Ignore Source ...	1 KB
	.metadata	11/29/2020 7:19 PM	METADATA File	1 KB
	.packages	4/1/2021 8:59 PM	PACKAGES File	15 KB
	active_ecommerce_flutter.iml	1/17/2021 6:49 PM	IML File	1 KB
	FlutterEcommerceAPI.postman_collectio...	3/25/2021 8:38 PM	JSON Source File	39 KB
	key.jks	4/1/2021 9:56 PM	JKS File	3 KB
	pubspec.lock	4/1/2021 8:59 PM	LOCK File	26 KB
	pubspec.yaml	4/1/2021 8:45 PM	Yaml Source File	3 KB
	README.md	11/29/2020 7:19 PM	Markdown Source...	1 KB

flutter\_projects > active\_ecommerce\_flutter >

Name	Date modified	Type	Size
.dart_tool	4/1/2021 10:38 PM	File folder	
.git	4/1/2021 10:51 PM	File folder	
.idea	4/3/2021 3:50 PM	File folder	
android	4/1/2021 10:44 PM	File folder	
assets	4/1/2021 5:53 PM	File folder	
build	4/1/2021 10:39 PM	File folder	
dummy_assets	3/31/2021 8:41 PM	File folder	
ios	11/29/2020 7:19 PM	File folder	
lib	4/1/2021 5:29 PM	File folder	
test	11/29/2020 7:19 PM	File folder	
.flutter-plugins	4/3/2021 3:51 PM	FLUTTER-PLUGINS...	3 KB
.flutter-plugins-dependencies	4/3/2021 3:51 PM	FLUTTER-PLUGINS...	8 KB
.gitattributes	11/29/2020 9:09 PM	Text Document	1 KB
.gitignore	11/29/2020 7:19 PM	Git Ignore Source ...	1 KB
.metadata	11/29/2020 7:19 PM	METADATA File	1 KB
.packages	4/1/2021 10:38 PM	PACKAGES File	15 KB
active_ecommerce_flutter.iml	1/17/2021 6:49 PM	IML File	1 KB
FlutterEcommerceAPI.postman_collectio...	3/25/2021 8:38 PM	JSON Source File	39 KB
key.jks	4/1/2021 9:56 PM	JKS File	3 KB
pubspec.lock	4/1/2021 10:38 PM	LOCK File	26 KB
pubspec.yaml	4/1/2021 8:45 PM	Yaml Source File	3 KB
README.md	11/29/2020 7:19 PM	Markdown Source...	1 KB

Then copy the key.jks from the root folder and paste it in the android/app folder

This PC > Local Disk (C:) > flutter\_projects > active\_ecommerce\_flutter > android > app >

Name	Date modified	Type	Size
src	11/29/2020 7:19 PM	File folder	
build.gradle	4/3/2021 3:50 PM	GRADLE File	3 KB
key.jks	4/1/2021 9:56 PM	JKS File	3 KB

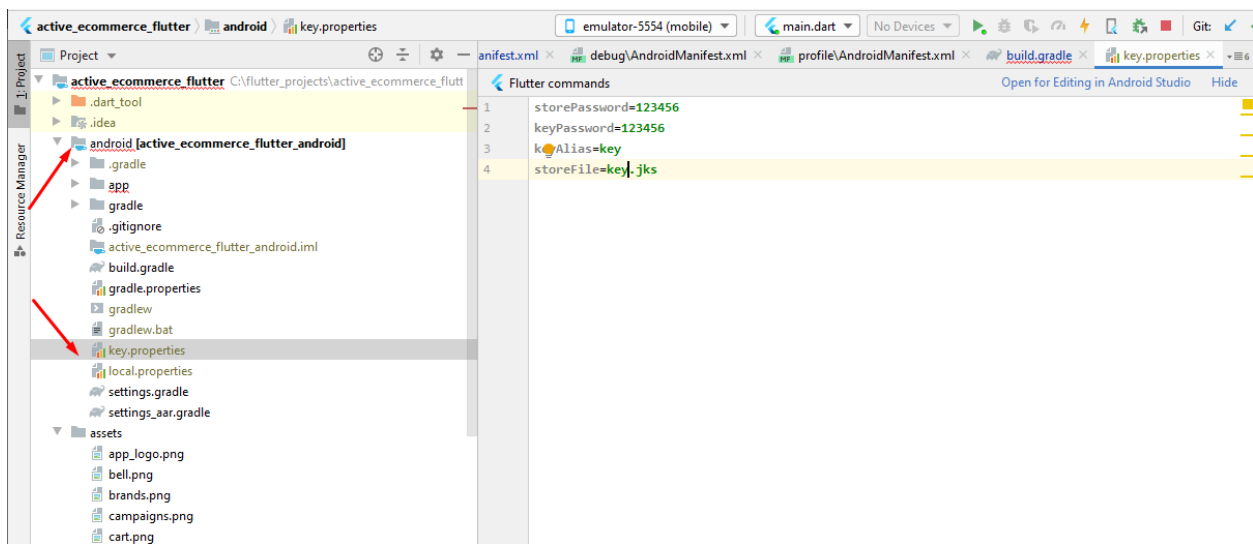
## Reference the keystore from the app

Create a file named `<your app dir>/android/key.properties` that contains a reference to your keystore:

```
storePassword=<password from previous step>
keyPassword=<password from previous step>
keyAlias=key
storeFile=<location of the key store file, such as /Users/<user name>/key.jks>
```

**\*\* If you lose the jks file , you will not be able to release a new update your app in playstore\*\***

Create new file `key.properties` in android folder . Enter the information



Read this

## Configure signing in gradle

Configure signing for your app by editing the `<your app dir>/android/app/build.gradle` file.

1. Add code before `android` block:

```
android {  
  ...  
}
```

With the keystore information from your properties file:

```
def keystoreProperties = new Properties()  
def keystorePropertiesFile = rootProject.file('key.properties')  
if (keystorePropertiesFile.exists()) {  
    keystoreProperties.load(new FileInputStream(keystorePropertiesFile))  
}  
  
android {  
  ...  
}
```

Load the `key.properties` file into the `keystoreProperties` object.

2. Add code before `buildTypes` block:

```
buildTypes {  
    release {  
        // TODO: Add your own signing config for the release build.  
        // Signing with the debug keys for now,  
        // so 'flutter run --release' works.  
        signingConfig signingConfigs.debug  
    }  
}
```

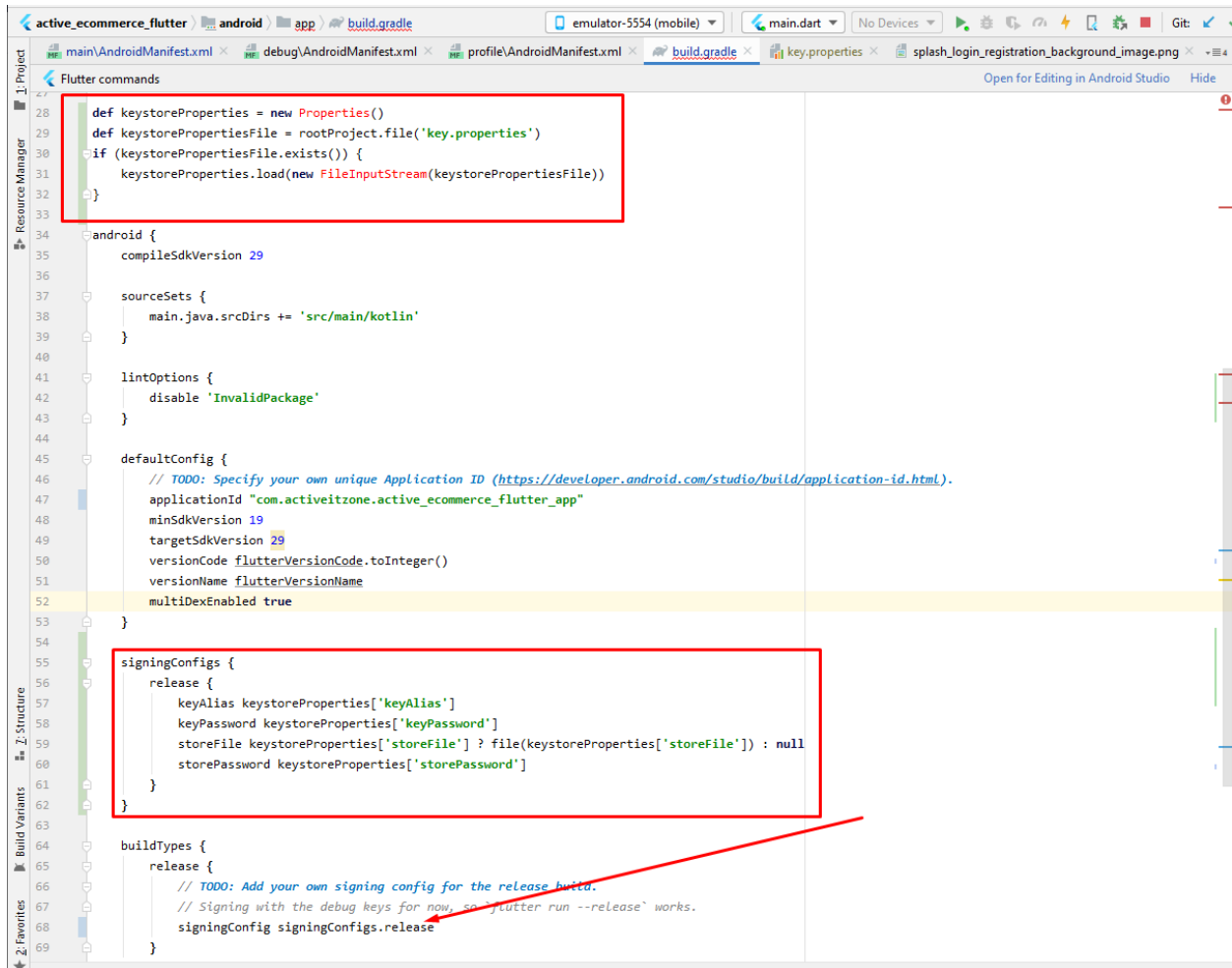
With the signing configuration info:

```
signingConfigs {  
    release {  
        keyAlias keystoreProperties['keyAlias']  
        keyPassword keystoreProperties['keyPassword']  
        storeFile keystoreProperties['storeFile'] ? file(keystoreProperties['storeFile']) : null  
        storePassword keystoreProperties['storePassword']  
    }  
}  
buildTypes {  
    release {  
        signingConfig signingConfigs.release  
    }  
}
```

Configure the `signingConfigs` block in your module's `build.gradle` file.

Release builds of your app will now be signed automatically.

in app/build.gradle do necessary changes



**Note:** You may need to run **flutter clean** after changing the gradle file. This prevents cached builds from affecting the signing process.

Now you are almost done

In your terminal run : **flutter build appbundle**

The release bundle for your app is created at **<your app dir>/build/app/outputs/bundle/release/app.aab**.

Upload this app.aab file to your google play console



## 5. How to update for android? **\*\*Read all the points carefully before doing anything**

- This section will help you if you are here for the update and have already generated the signed release apk/appbundle the last time and already have the keytool and the manifest file ready in your old project folder.
- If you are installing and building the release file for the first time this section is not for you.
- Extract the source\_code.zip. You will find this inside the main zip.
- Open the folder in your android studio.
- \*\*Remember to open this in a separate folder than your old project.
- Even if you are building an app for ios, use android studio for the build.
- Then in your android studio terminal run:  
`flutter pub get`
- This will fetch all the necessary packages
- If you are updating, you must have build the key.jks previously
- Copy the key.jks , key.properties, and the manifest file from your old project and paste in the correct locations
- See the previous screenshots for the file locations
- If you are missing your old project, you have to configure key.properties, and the manifest file like described in the installation.
- As our source code is made ready for the fresh installation , you will have to do all your configuration (like domain path, app color, package name etc ) shown in the previous steps.
- But do not create a new key.jks, you have to update your app with the existing key
- If you have somehow lost your previous key , you have to release a totally new app to the play store. You will not be able to release an update.
- In your terminal run : `flutter build appbundle`
- The release bundle for your app is created at `<your app dir>/build/app/outputs/bundle/release/app.aab`.

- Upload this app.aab file to your google play console

## 9. How to configure social login?

In lib/social\_config.dart make necessary changes.

**static final allow\_google\_login = false;** make it true if you need to show the google button

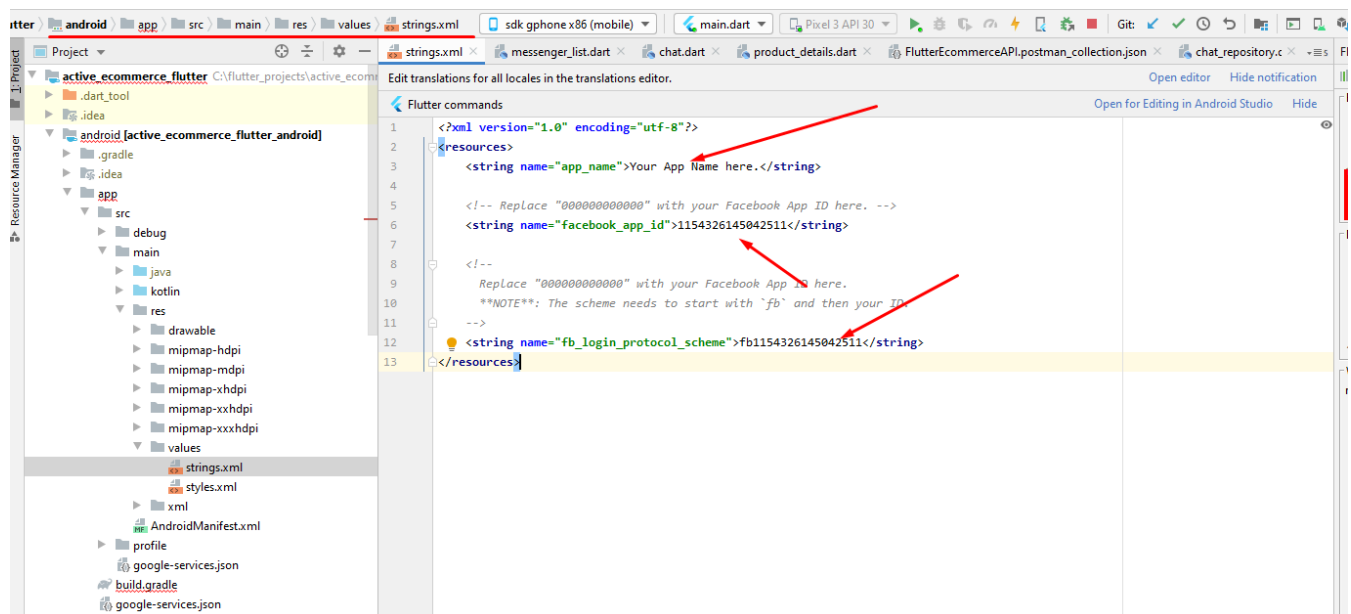
**static final allow\_facebook\_login = false;** make it true if you need to show the facebook button

**Facebook:** Package Used

[https://pub.dev/packages/flutter\\_facebook\\_login](https://pub.dev/packages/flutter_facebook_login)

See its documentation and steps

Configure the below file with facebook credentials



Remember to configure the facebook console properly.

- Make sure valid Outh uri is given.
- Enable Browser OAuth Login
- Also provide the privacy and support urls

Android

Quick Start

×

Google Play Package Name

com.activeitzone.active\_ecommerce\_flutter\_app

×

Class Name

Main

Key Hashes

Key hashes are 28 characters including the trailing = and are limited to the following characters: [a-zA-Z0-9+/=]

Amazon Appstore URL (Optional)

Ex. http://www.amazon.com/dp/B004GJDQT8

No

Single Sign On

Will launch from Android Notifications

Yes

Log In-App Events Automatically (Recommended)

Turning this toggle on automatically logs in-app events, including Purchase, Start Trial and Subscribe, that are processed through the GooglePlay store. To automatically log Purchase events, use Facebook SDK for Android v4.36 or higher. For Subscribe and Start Trial events, use Facebook SDK for Android v5.1. **Note:** When this toggle is turned on, you should stop manually logging in-app Purchase, StartTrial, and Subscribe events on Android, otherwise you will see duplicate reporting. [Learn More](#)

No

Google Play Credentials

Use your Google Developers credentials to reduce fraudulent in-app purchases.

Android

Quick Start

×

Google Play Package Name

com.activeitzone.active\_ecommerce\_flutter\_app

×

Class Name

Main

Key Hashes

Key hashes are 28 characters including the trailing = and are limited to the following characters: [a-zA-Z0-9+/=]

Amazon Appstore URL (Optional)

Ex. http://www.amazon.com/dp/B004GJDQT8

No

Single Sign On

Will launch from Android Notifications

Yes

Log In-App Events Automatically (Recommended)

Turning this toggle on automatically logs in-app events, including Purchase, Start Trial and Subscribe, that are processed through the GooglePlay store. To automatically log Purchase events, use Facebook SDK for Android v4.36 or higher. For Subscribe and Start Trial events, use Facebook SDK for Android v5.1. **Note:** When this toggle is turned on, you should stop manually logging in-app Purchase, StartTrial, and Subscribe events on Android, otherwise you will see duplicate reporting. [Learn More](#)

No

Google Play Credentials

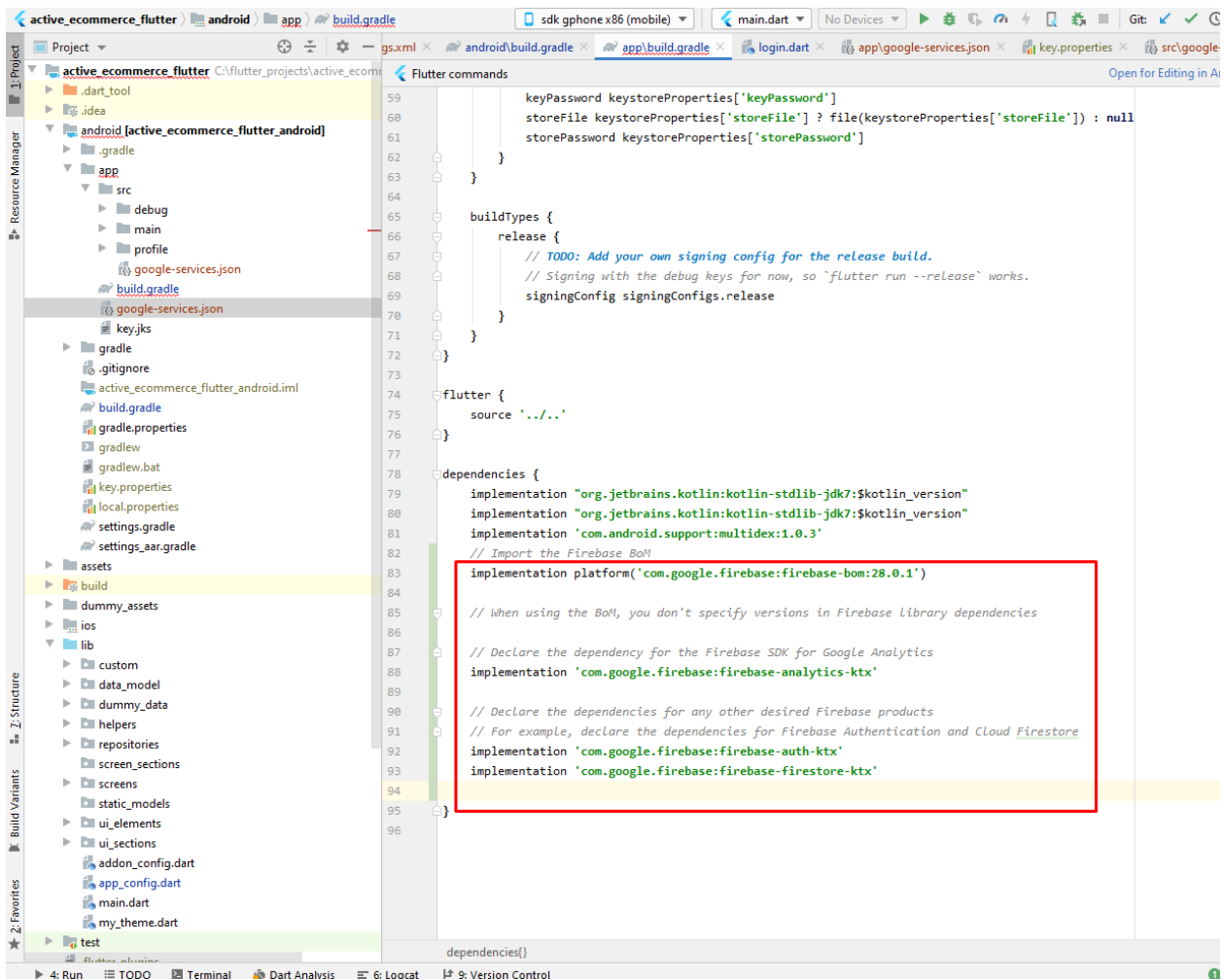
Use your Google Developers credentials to reduce fraudulent in-app purchases.

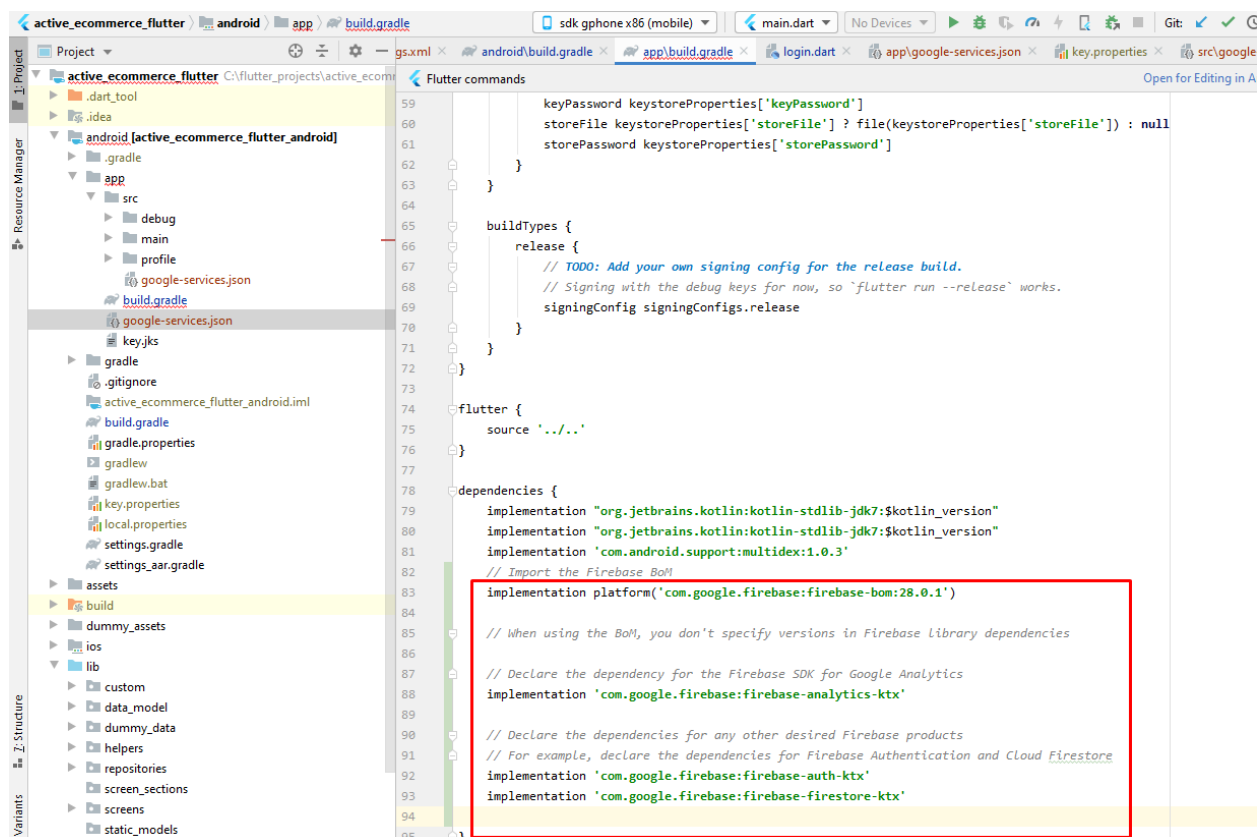
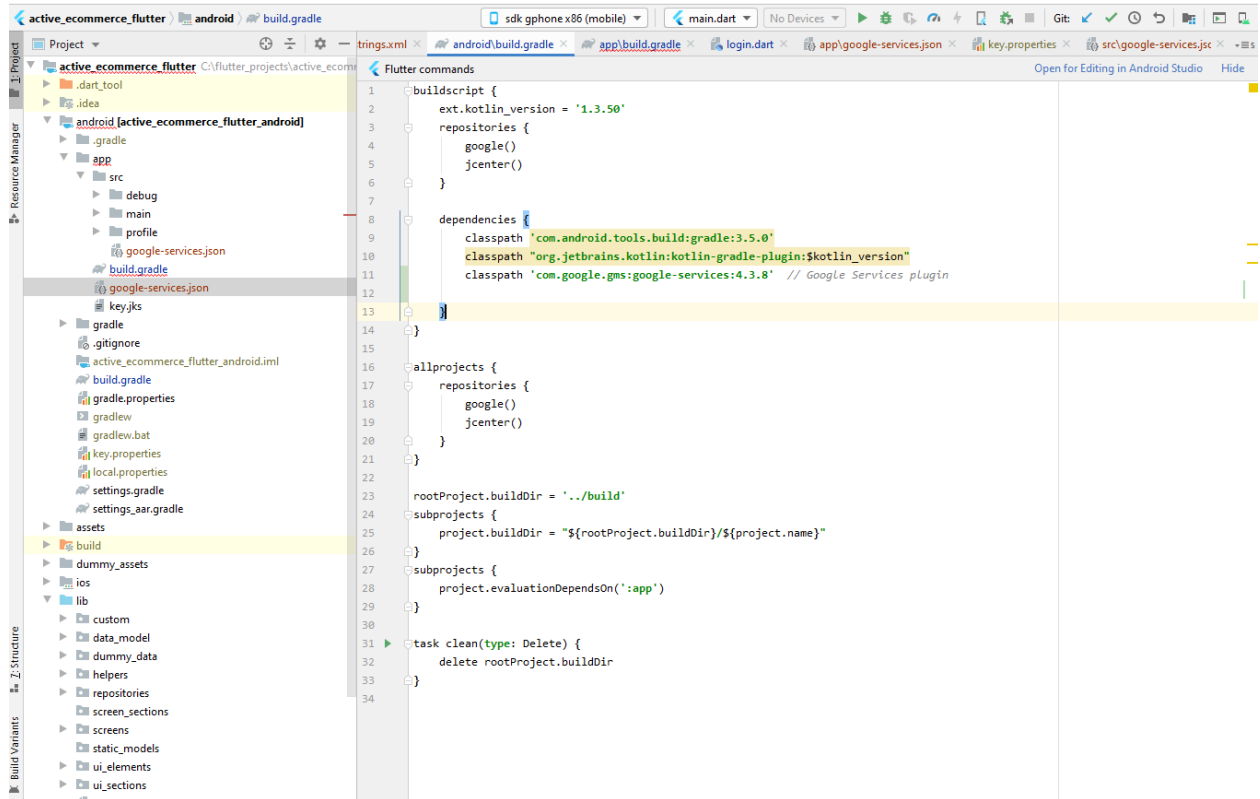
## Google: Package Used

[https://pub.dev/packages/google\\_sign\\_in](https://pub.dev/packages/google_sign_in)

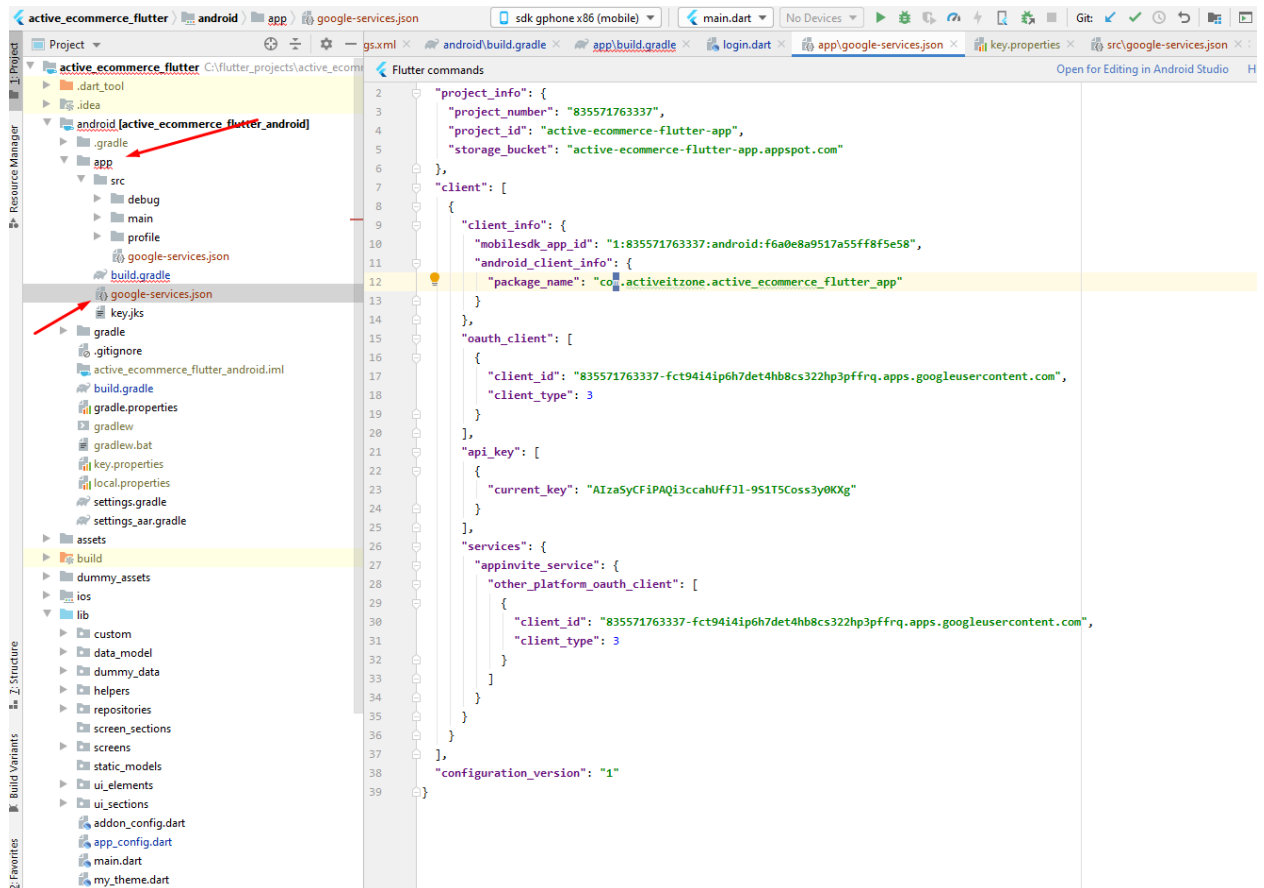
See its documentation and steps from the link.

These files below are already configured:





You will need to generate your own google-services.json. Do not use ours - it will not work for you

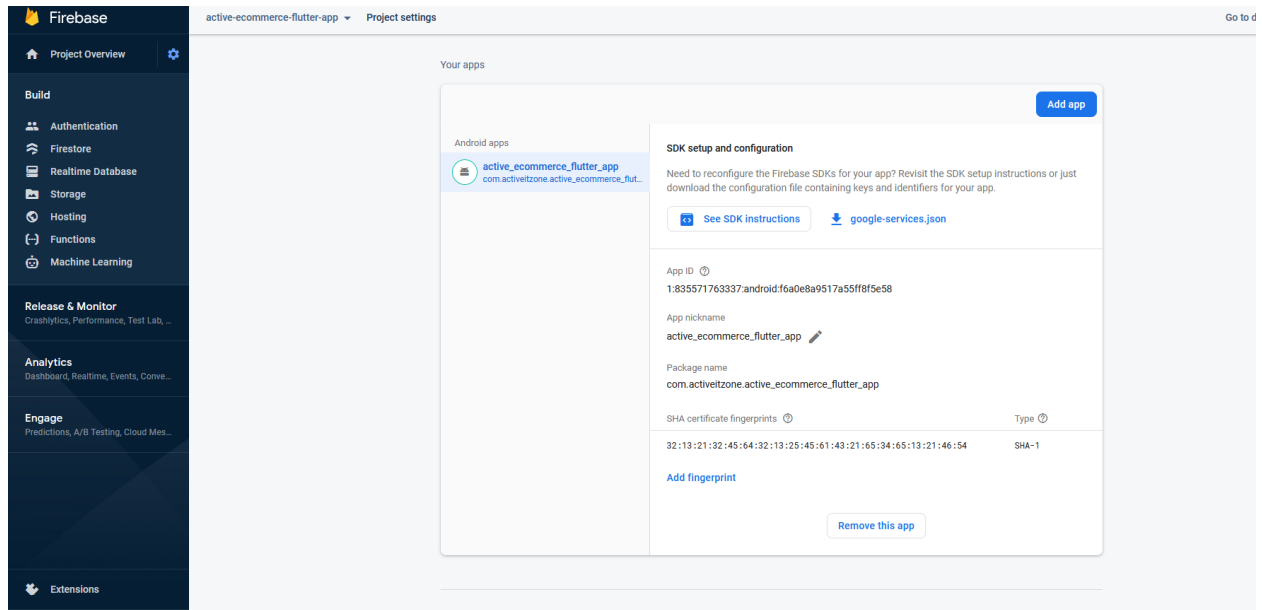


Firebase console:

<https://console.firebase.google.com/u/0/>

Follow the guideline from here [https://pub.dev/packages/google\\_sign\\_in](https://pub.dev/packages/google_sign_in)

You need to create an app. You need to provide your fingerprints here (sha1 and sha 256)



You will find your signature/fingerprints from here ( Provided that you already have generated the key). You will also need the path of your key.jks. You may have already kept it in the root folder.

```
C:\Program Files\Android\Android Studio\jre\bin>keytool -list -v -keystore C:\flutter_projects\active_ecommerce_flutter\key.jks -alias key -storepass 123456 -keypass 123456
Alias name: key
Creation date: Apr 1, 2021
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
  Owner: CN=Test, OU=Test, O=Test, L=Test, ST=Test, C=us
  Issuer: CN=Test, OU=Test, O=Test, L=Test, ST=Test, C=us
  Serial number: 656ab3f5
  Valid from: Thu Apr 01 21:56:13 BDT 2021 until: Mon Aug 17 21:56:13 BDT 2048
  Certificate fingerprints:
    MD5: B4:6B:55:48:BF:D9:E1:1B:43:2D:76:3D:99:1A:D0:B8
    SHA1: B1:3A:53:CF:F8:0A:07:1F:9B:6E:14:8E:24:69:7C:EC:03:05:2F
    SHA256: 92:09:F0:BF:56:F8:14:AB:AD:CB:C6:43:1D:79:FA:3F:66:2E:D8:2D:66:FD:5F:BE:08:10:88:06:FA:37:46:A1
  Signature algorithm name: SHA256withRSA
  Subject Public Key Algorithm: 2048-bit RSA key
  Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
```

You will also need to enable the people api if needed.

Google Cloud Platform Select a project Search products and resources

## Google People API

Google

Provides access to information about profiles and contacts.

[ENABLE](#) [TRY THIS API](#)

Click to enable this API

[OVERVIEW](#) [DOCUMENTATION](#)

### Overview

Provides access to information about profiles and contacts.

#### About Google

Google's mission is to organize the world's information and make it universally accessible and useful. Through products and platforms like Search, Maps, Gmail, Android, Google Play, Chrome and YouTube, Google plays a meaningful role in the daily lives of billions of people.

#### Additional details

Type: [SaaS & APIs](#)  
Last updated: 3/19/21  
Category: [Social](#)  
Service name: people.googleapis.com

### Tutorials and documentation

[Learn more](#)

### Terms of Service

## oAuth Consent screen

Google Cloud Platform Active Commerce Flutter Search products and resources

APIs & Services

- Dashboard
- Library
- Credentials**
- OAuth consent screen
- Domain verification
- Page usage agreements

### Create OAuth client ID

A client ID is used to identify a single app to Google's OAuth servers. If your app runs on multiple platforms, each will need its own client ID. See [Setting up OAuth 2.0](#) for more information.

Application type \*  
Android

[Learn more](#) about OAuth client types

Name \*  
Android client 1

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

Package name \*  
com.activeitzone.active\_ecommerce\_flutter\_app

From your AndroidManifest.xml file.

SHA-1 certificate fingerprint \*  
B1:3A:33:CF:F8:9A:07:17:1F:9B:6E:14:8E:24:69:7C:EC:03:D5:2F

SHA-1 signing certificate fingerprint restricts usage to your Android apps. [Learn more](#)

Use this command to get the fingerprint.

```
$ keytool -keystore path-to-debug-or-production-keystore -list -v
```

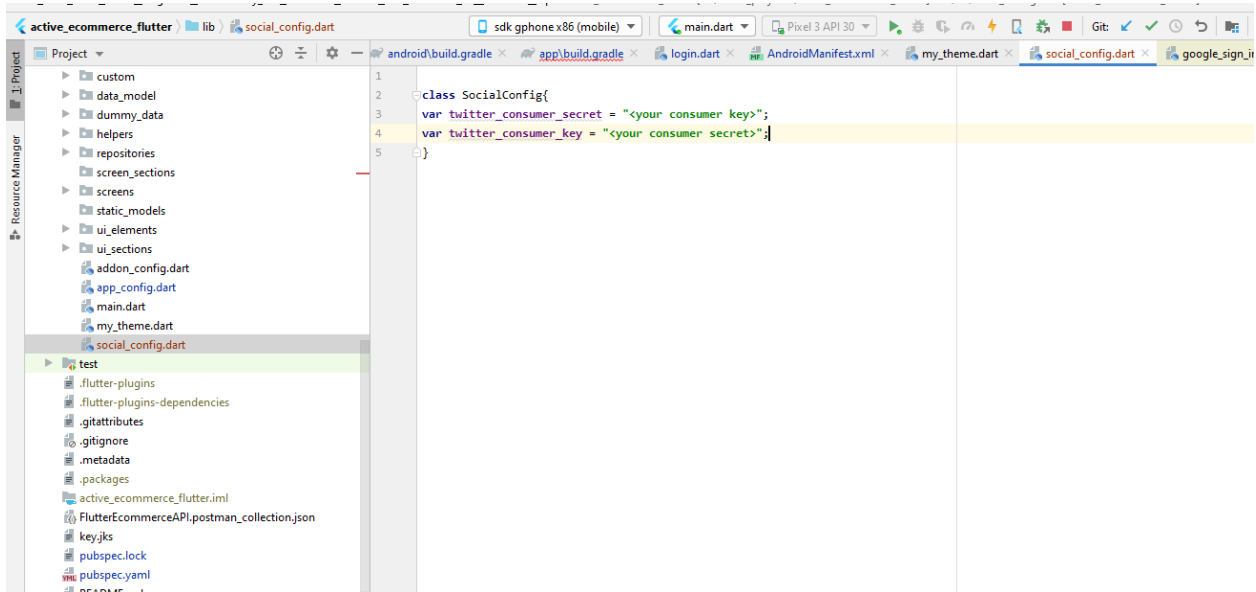
[CREATE](#) [CANCEL](#)



**Twitter** : package used

[https://pub.dev/packages/flutter\\_twitter\\_login/install](https://pub.dev/packages/flutter_twitter_login/install)

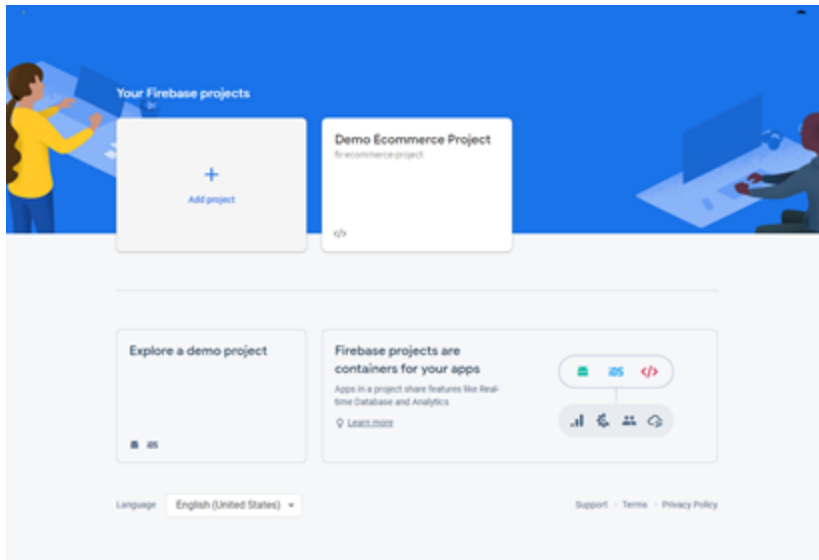
Just put correct values to the social config file.



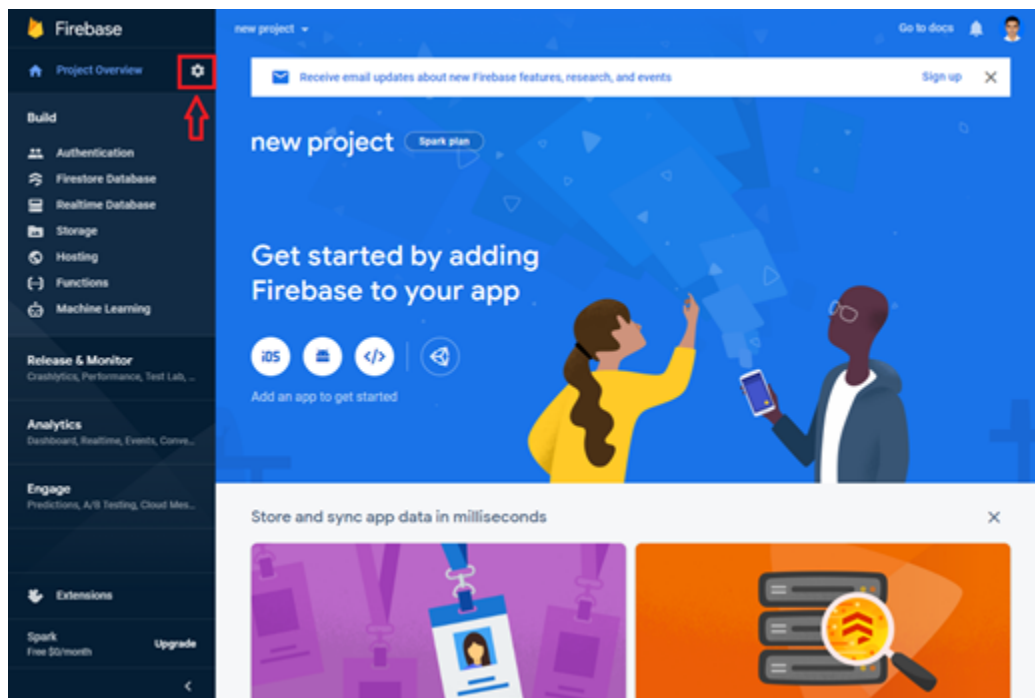
## 10. How to configure push notification?

To use firebase follow the procedure which are mentioned below

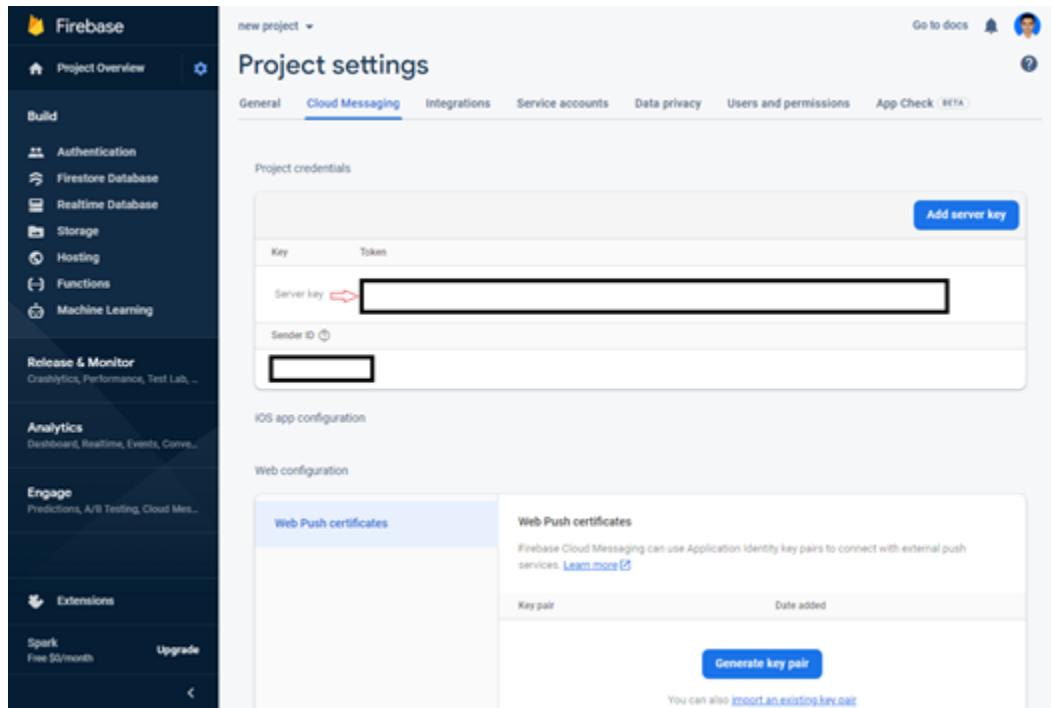
1. Go to this URL to create project <https://console.firebase.google.com/u/0/>  
If you already have a project then continue with that.



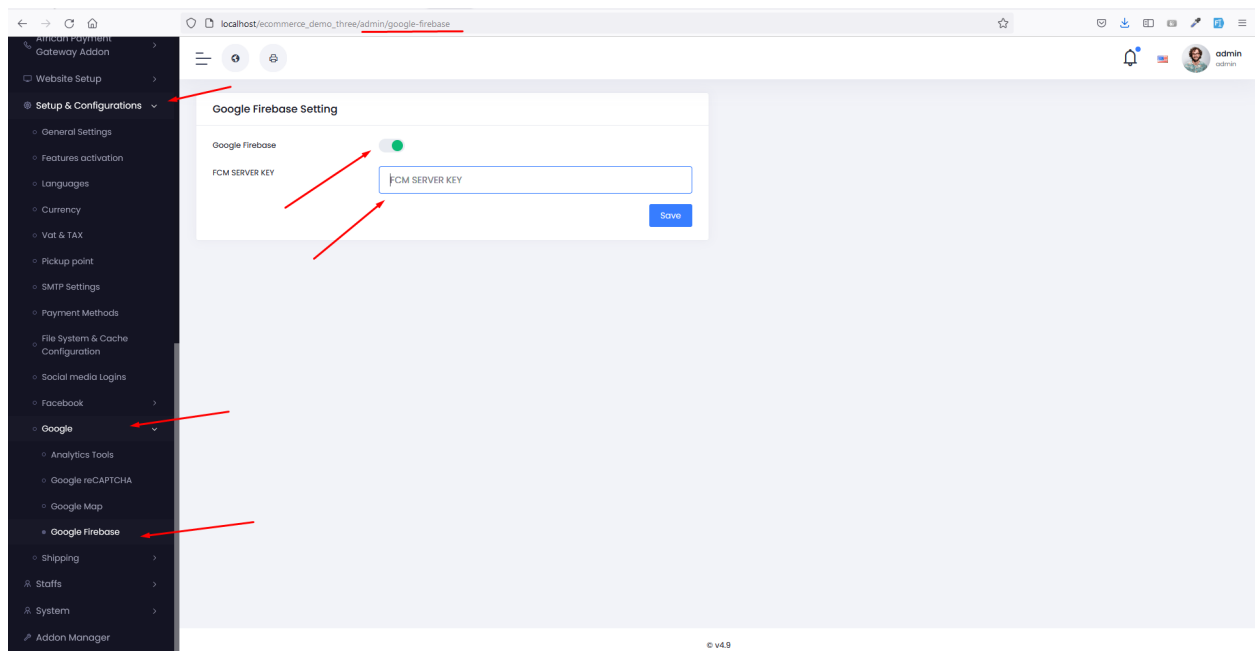
2. Now go to project settings to get server key



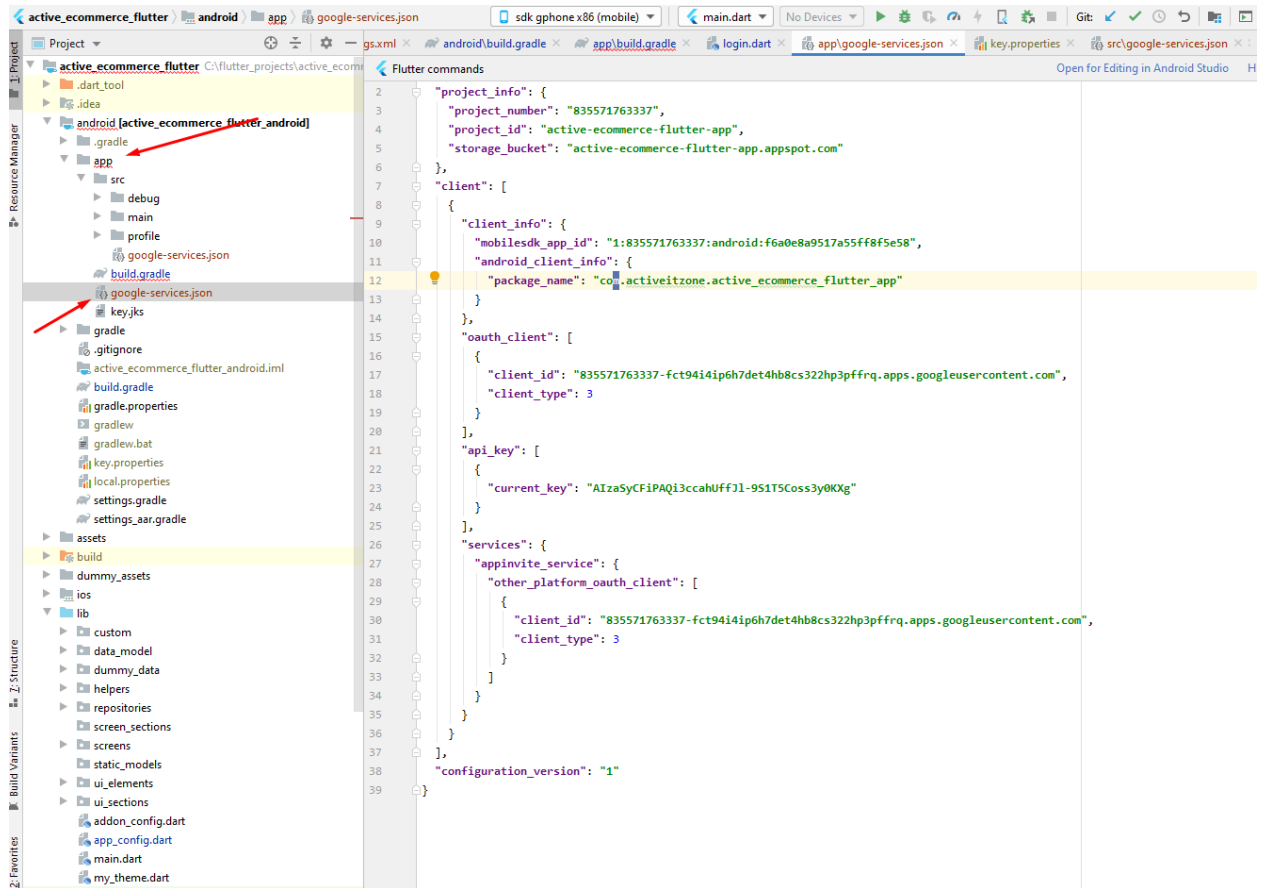
3. To get server key click on Cloud Messaging option



4. Turn on the switch and put the server key in admin panel



5. You will need to generate your own google-services.json. Do not use ours - it will not work for you

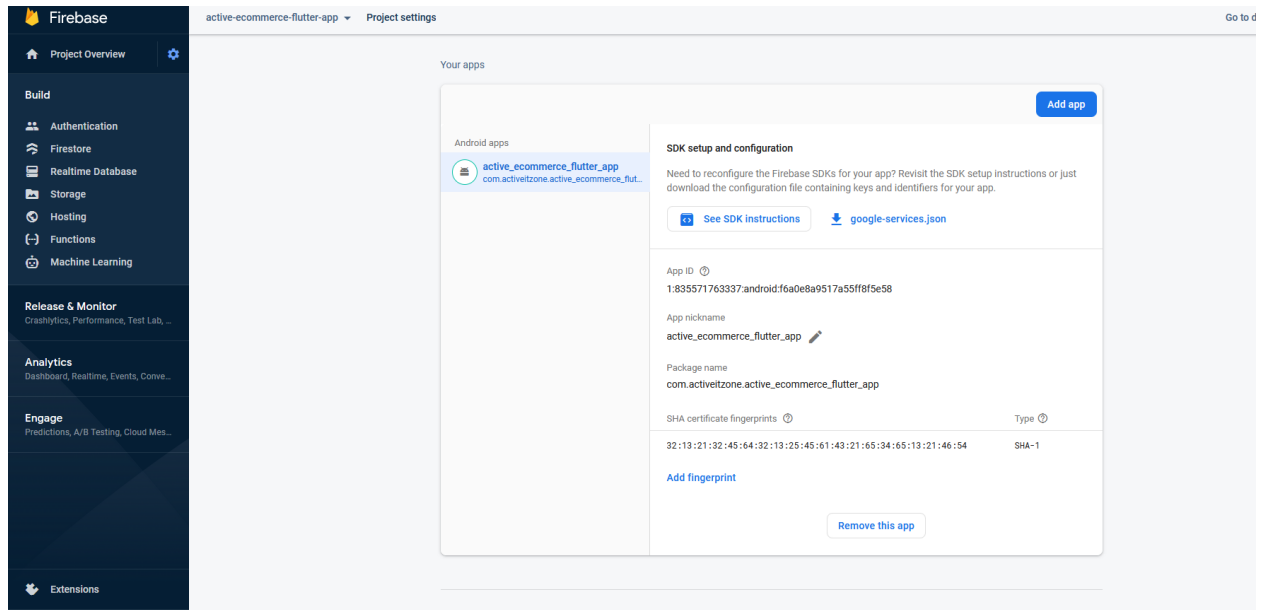


Firestore console:

<https://console.firebase.google.com/u/0/>

Follow the guideline from here [https://pub.dev/packages/google\\_sign\\_in](https://pub.dev/packages/google_sign_in)

You need to provide your fingerprints here (sha1 and sha 256)



You will find your signature/fingerprints from here ( Provided that you already have generated the key). You will also need the path of your key.jks. You may have already kept it in the root folder.

```
C:\Program Files\Android\Android Studio\jre\bin>keytool -list -v -keystore C:\flutter_projects\active_ecommerce_flutter\key.jks -alias key -storepass 123456 -keypass 123456
Alias name: key
Creation date: Apr 1, 2021
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
  Owner: CN=Test, OU=Test, O=Test, L=Test, ST=Test, C=us
  Issuer: CN=Test, OU=Test, O=Test, L=Test, ST=Test, C=us
  Serial number: 656ab3f5
  Valid from: Thu Apr 01 21:56:13 BDT 2021 until: Mon Aug 17 21:56:13 BDT 2048
  Certificate fingerprints:
    MD5: B4:6B:55:48:BF:D9:E1:1B:43:2D:76:3D:99:1A:D0:B8
    SHA1: B1:3A:53:CF:F8:0A:07:17:1F:9B:6E:14:8E:24:69:7C:EC:03:05:2F
    SHA256: 92:09:F0:BF:56:F8:14:AB:AD:CB:C6:43:1D:79:FA:3F:66:2E:D8:2D:66:FD:5F:BE:08:10:88:06:FA:37:46:A1
  Signature algorithm name: SHA256withRSA
  Subject Public Key Algorithm: 2048-bit RSA key
  Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
```

7. Although most of the configuration for android is done you can check guidelines from here.

<https://firebase.google.com/docs/cloud-messaging/android/client>

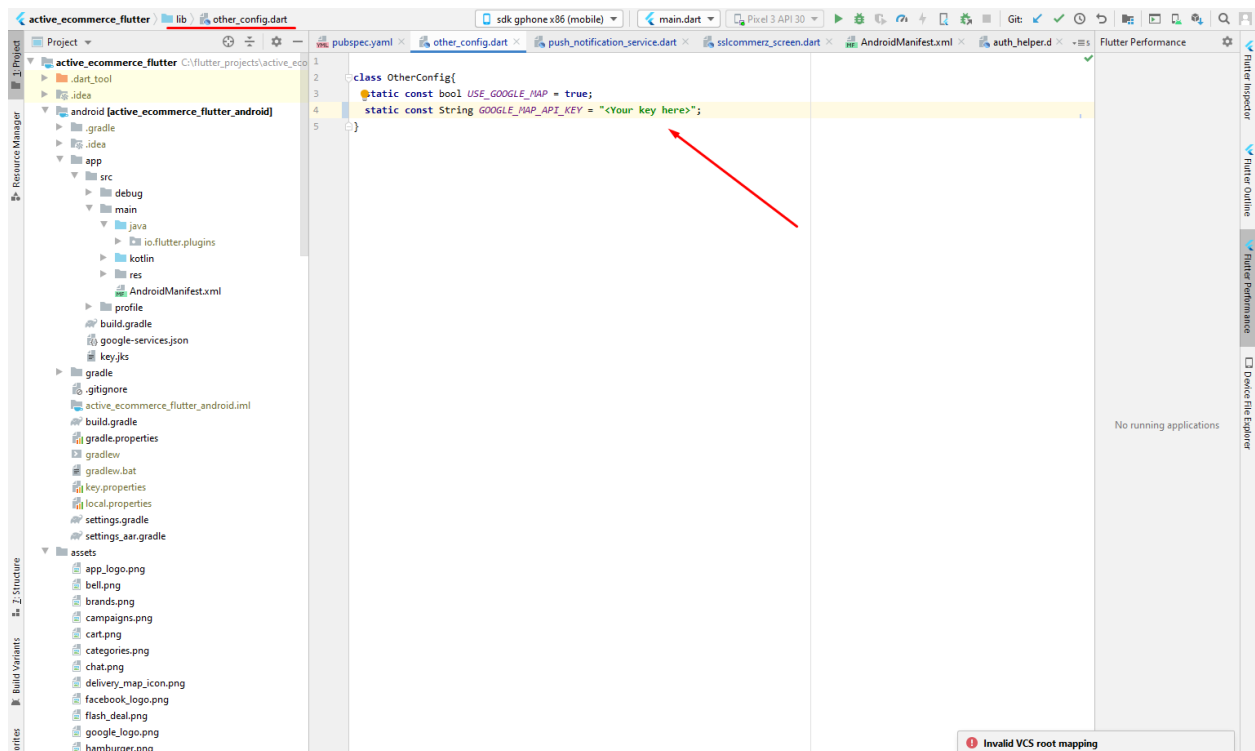
8. For ios follow this <https://firebase.google.com/docs/cloud-messaging/ios/client>

9. Push notification is a little bit tricky , so follow the guidelines properly. Learn more about how a firebase application connects with your mobile app from google searching if needed.

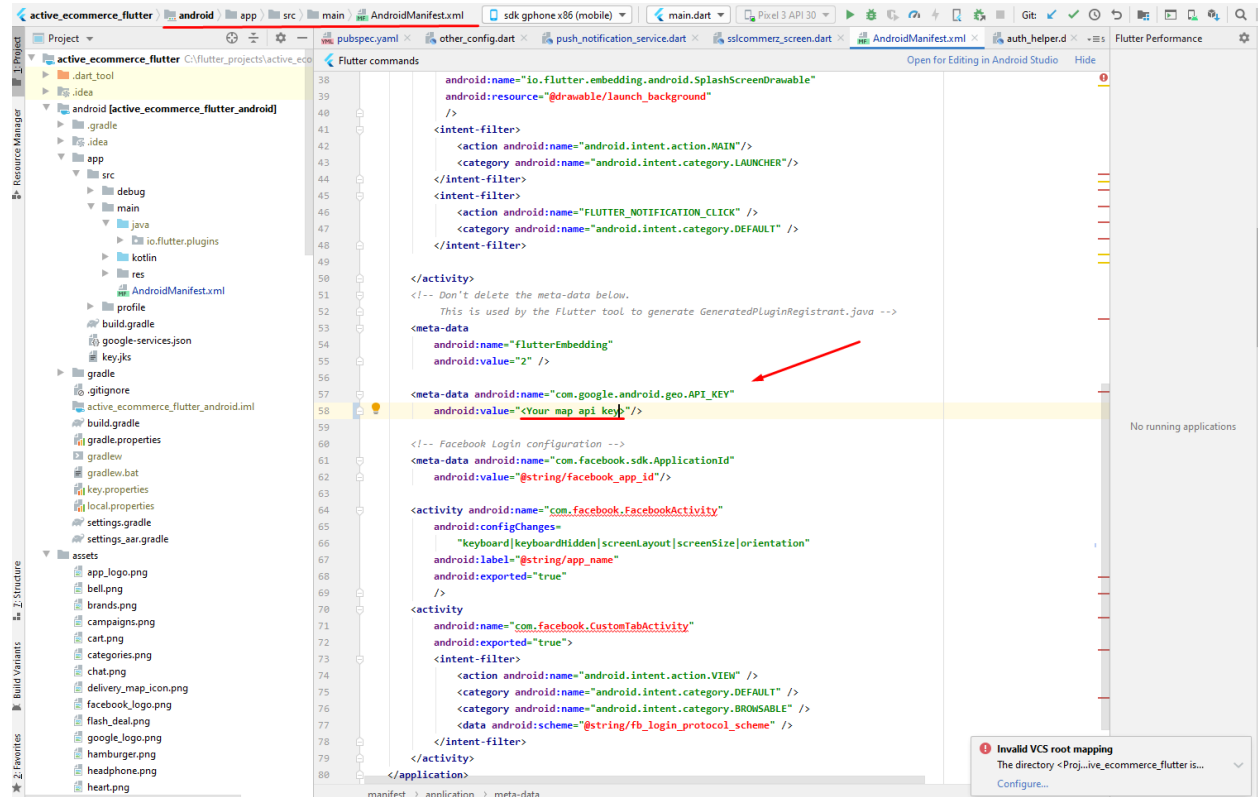
## 11. How to configure google map? (Read the whole thing before implementing)

1. Go to <https://console.developers.google.com/> and generate api keys separately for ios and android. No restrictions are needed

1. In lib/other\_config.dart make, use google map = true and put google map api key



2. In main `AndroidManifest.xml` put the map api key

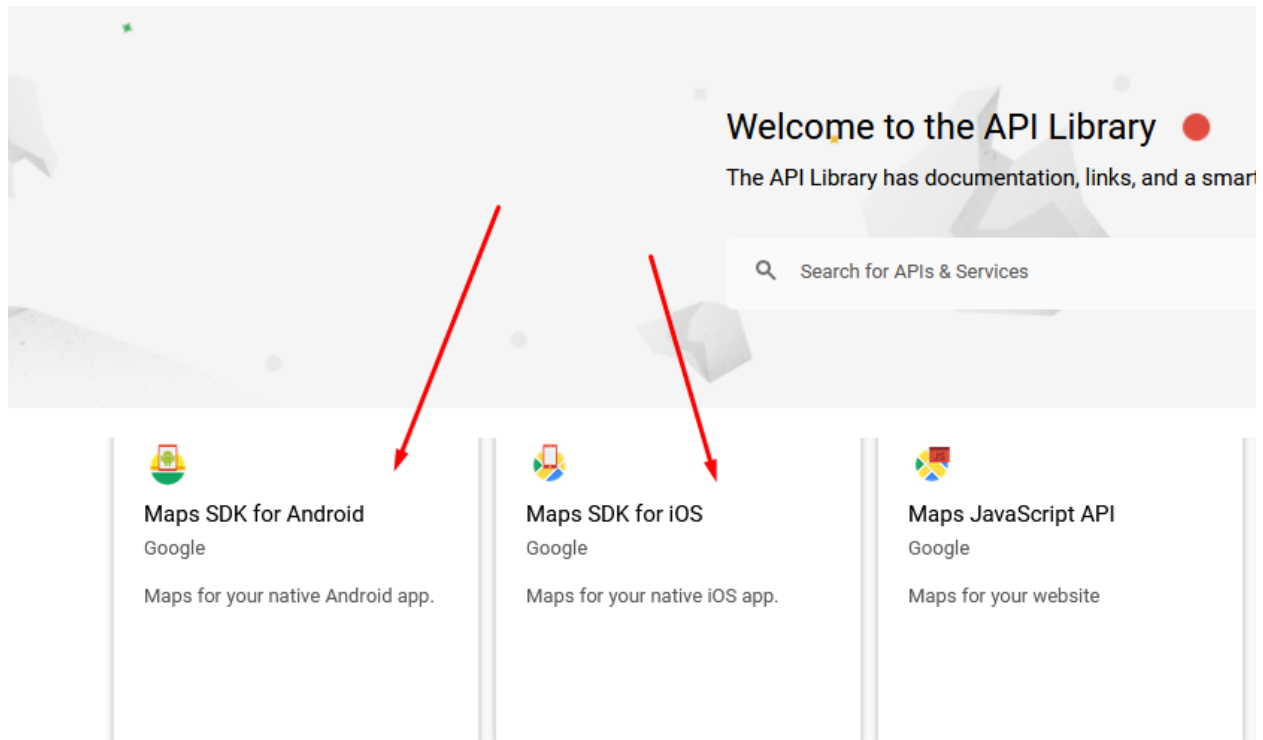


### 3. For ios follow this

<https://blog.logrocket.com/adding-google-maps-to-a-flutter-app/#addinggooglemapstoflutterio>

**S**

4. Enable android and ios api. These are free.



Machine learning

5. In the customer app we are searching location via text .And while setting pin to location taking information from the location. For these we would need these apis enabled.

Unfortunately these api are not free, you will need to add card.If you do not want to spend money you cannot use google map in the customer app



### Geocoding API

Google Enterprise API

Convert between addresses and geographic coordinates.

MANAGE



API Enabled



### Places API

Google Enterprise API

Get detailed information about 100 million places

MANAGE



API Enabled