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 ?
12. How to configure default language for mobile app?
13. How to configure multiple languages for mobile app?
14. My data is not changing. How to clear cache?

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 3.0.2 • channel stable**
Dart version must be : **Dart 2.17.3**
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 scource_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?

A. 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)

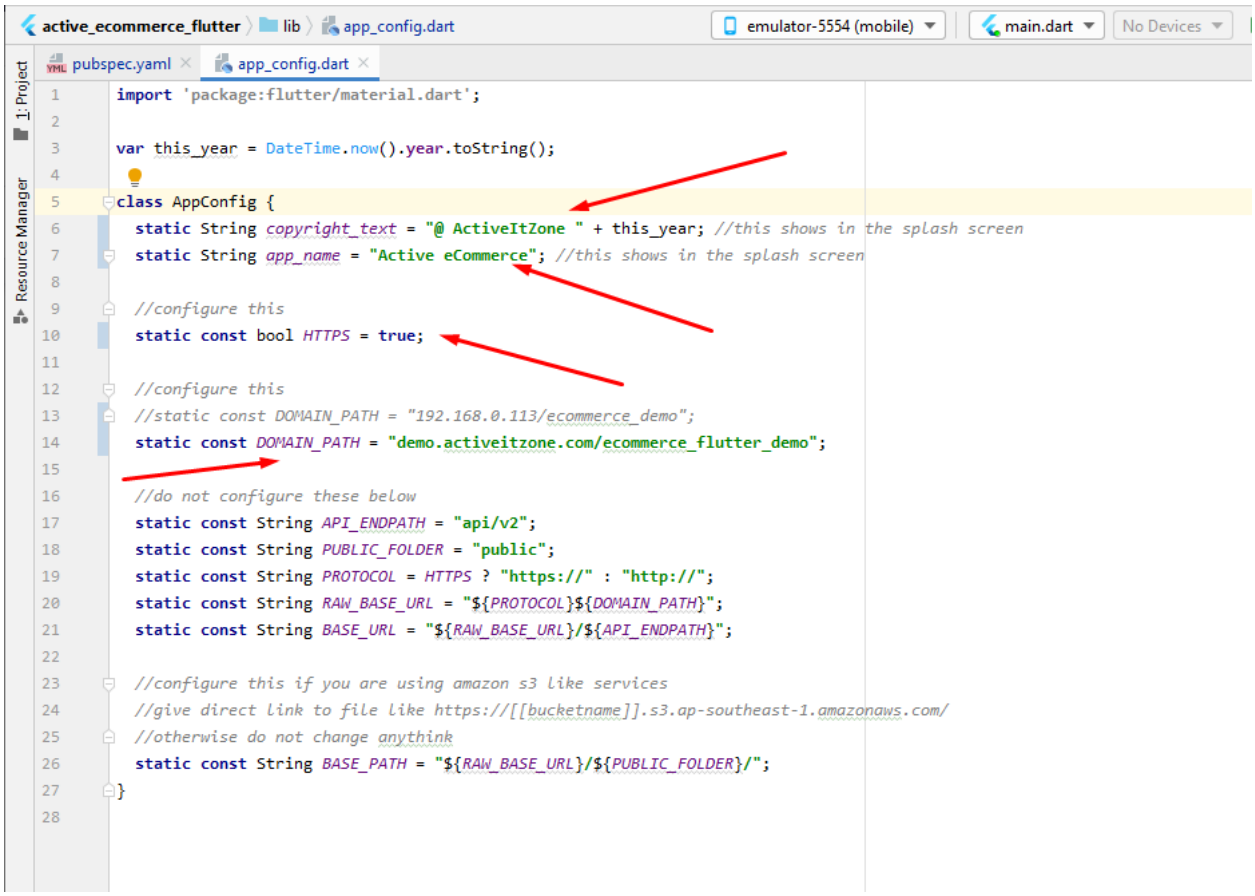
If you are using localhost , DOMAIN_PATH should be “your_ip_address/your_project”;

** “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/";



B. Theme Config:

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

Open lib/my_theme.dart

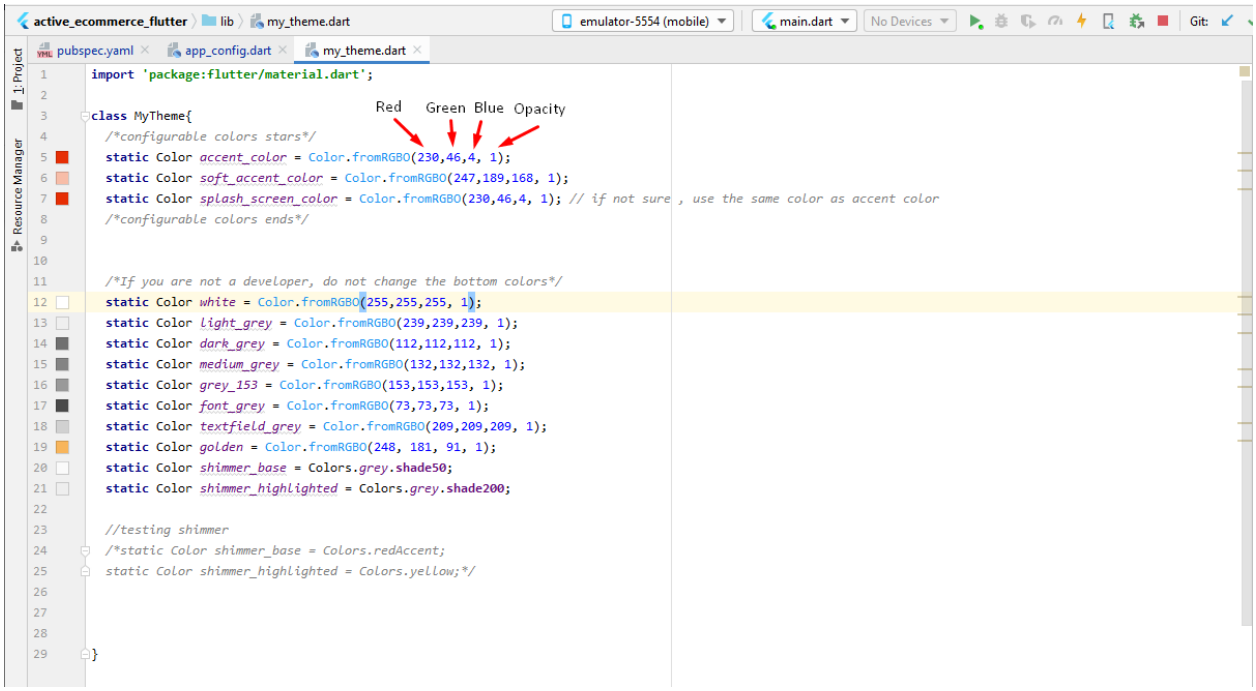
You can change the accent_color, soft_accent_color, splash_screen_color variable.

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.

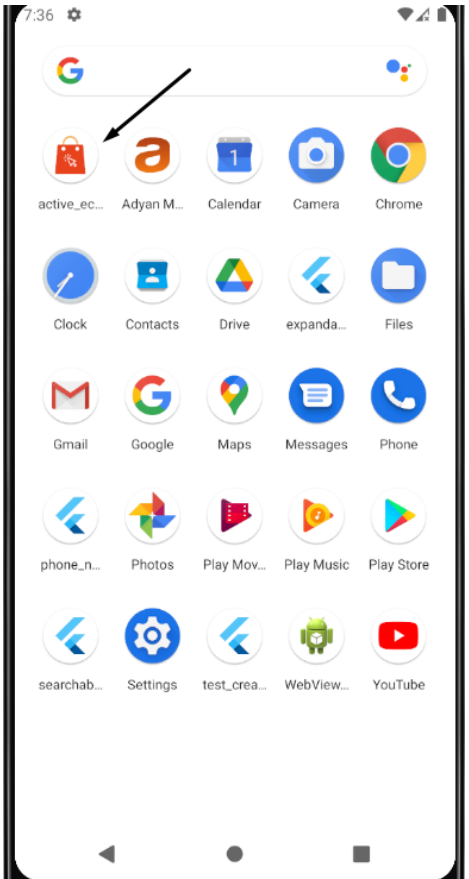



```dart
import 'package:flutter/material.dart';

class MyTheme{
  /*configurable colors stars*/
  static Color accent_color = Color.fromRGBO(230,46,4, 1);
  static Color soft_accent_color = Color.fromRGBO(247,189,168, 1);
  static Color splash_screen_color = Color.fromRGBO(230,46,4, 1); // if not sure , use the same color as accent color
  /*configurable colors ends*/


  /*If you are not a developer, do not change the bottom colors*/
  static Color white = Color.fromRGBO(255,255,255, 1);
  static Color light_grey = Color.fromRGBO(239,239,239, 1);
  static Color dark_grey = Color.fromRGBO(112,112,112, 1);
  static Color medium_grey = Color.fromRGBO(132,132,132, 1);
  static Color grey_153 = Color.fromRGBO(153,153,153, 1);
  static Color font_grey = Color.fromRGBO(73,73,73, 1);
  static Color textfield_grey = Color.fromRGBO(209,209,209, 1);
  static Color golden = Color.fromRGBO(248, 181, 91, 1);
  static Color shimmer_base = Colors.grey.shade50;
  static Color shimmer_highlighted = Colors.grey.shade200;

  //testing shimmer
  /*static Color shimmer_base = Colors.redAccent;
  static Color shimmer_highlighted = Colors.yellow;*/


}
```

C. 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

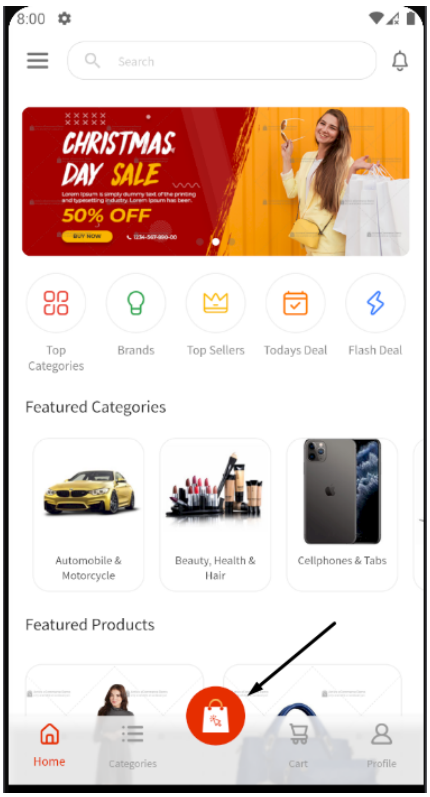
Then run :

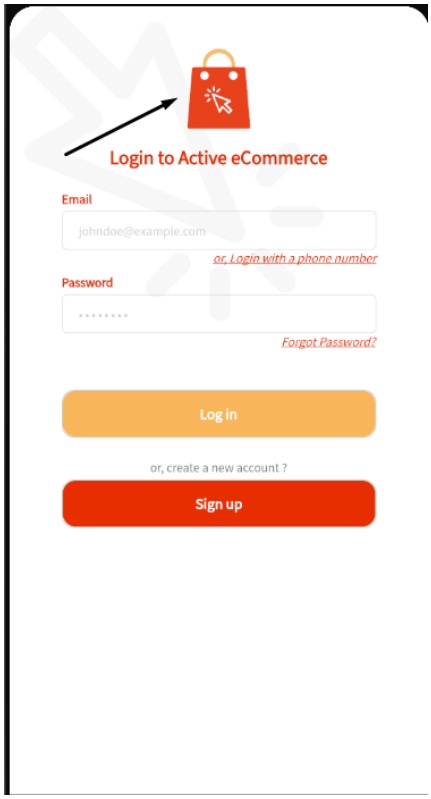
flutter pub run flutter_launcher_icons:main

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

D. 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.

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 (anddroid/app/build.gradle)

defaultConfig {

applicationId "bundleId com.onatcipli.networkUpp"

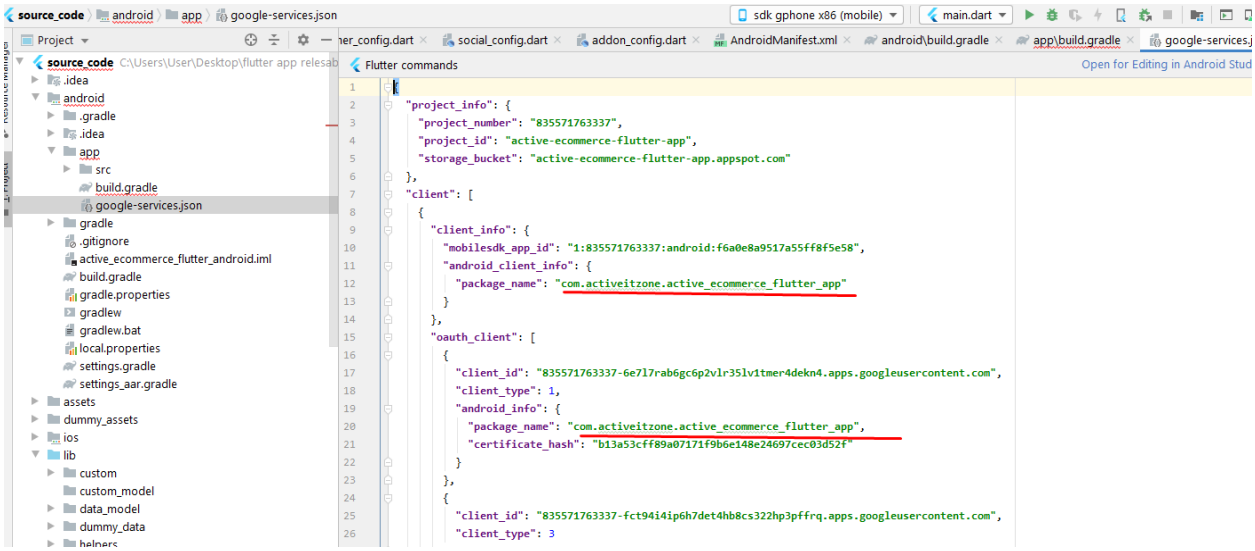
……

}

**However in one place you must have to manually change your package name.

Open android/app/google-services.json , and change your package name manually.

Otherwise you will get a build error , even on emulator on debug build



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.

5. 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 .

6. 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 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
[√] 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


[√] 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.

[√] 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)

[√] VS Code (version 1.53.2)
```

Find binary path

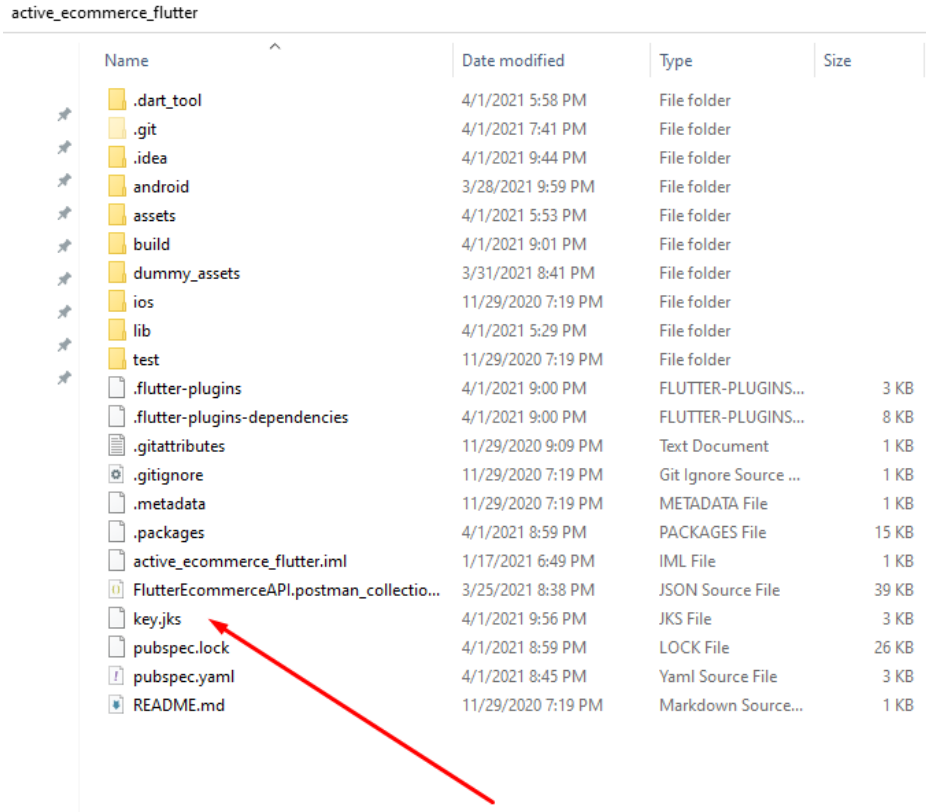
```
C:\>cd "Program Files"

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>
```

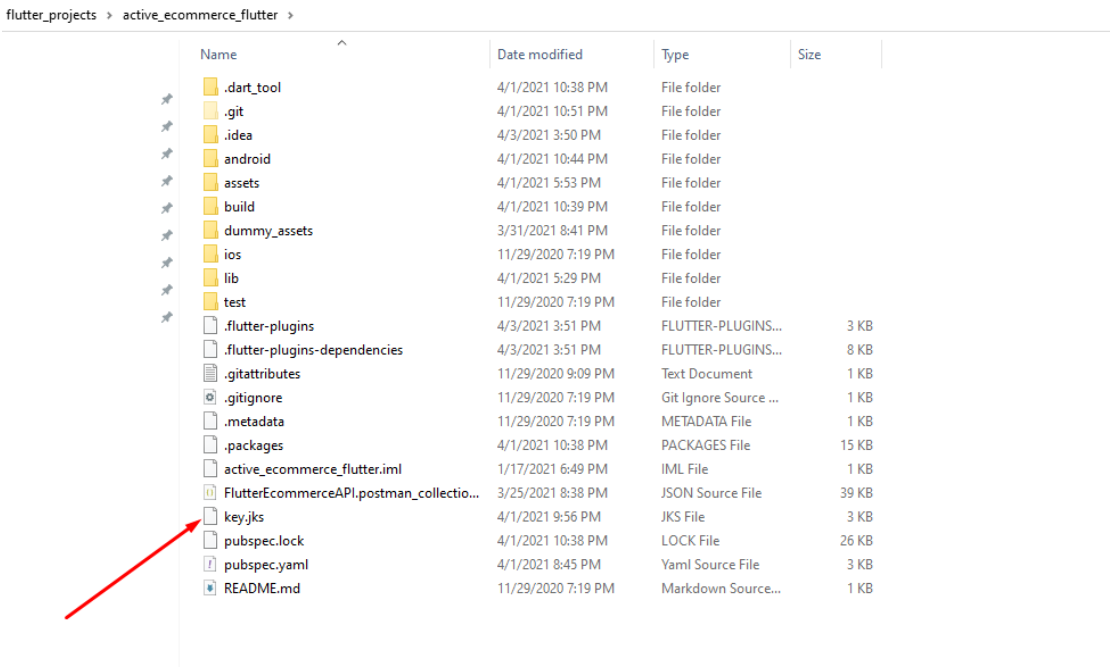
Then generate and store the key (image on next page)

```
Microsoft Windows [Version 10.0.19041.867]
(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]
```

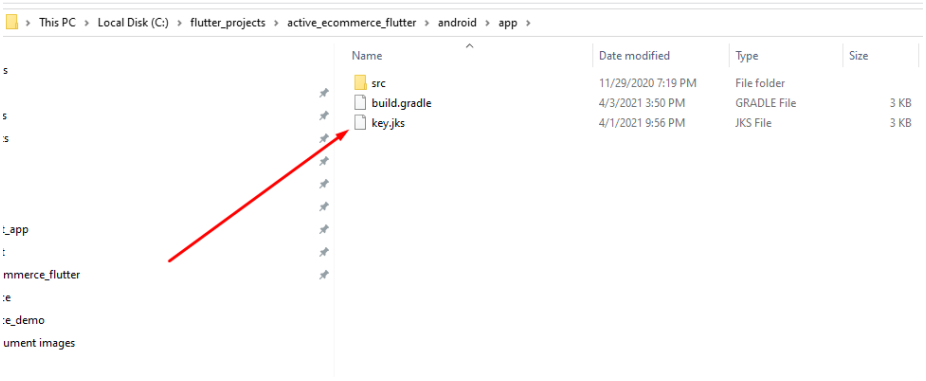
Binary location

File path to save the key





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



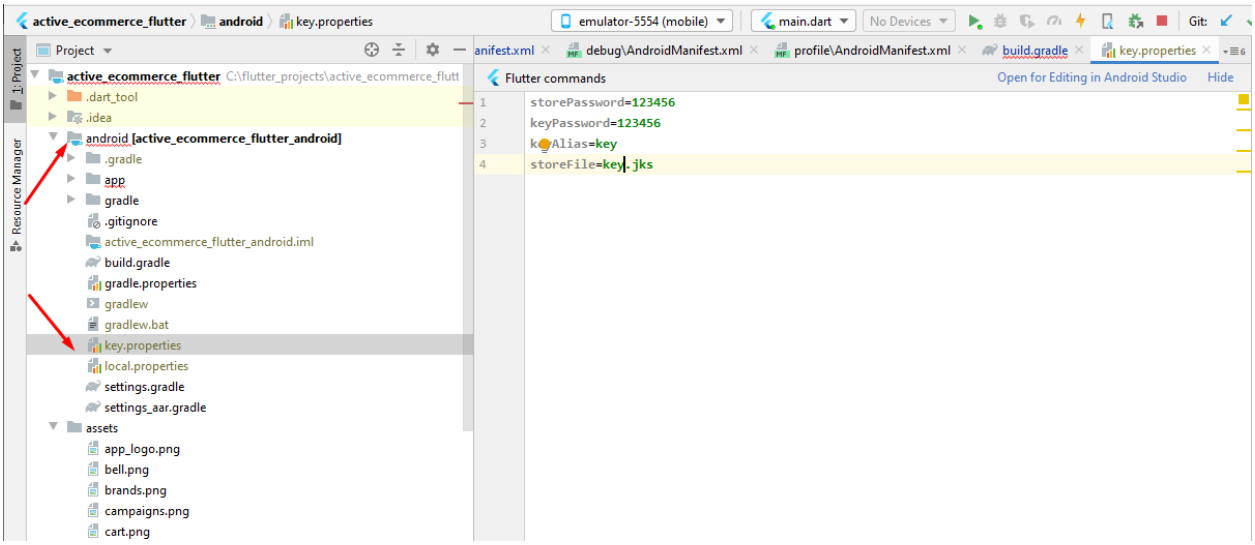
Reference the keystore from the app

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
    }
}
```

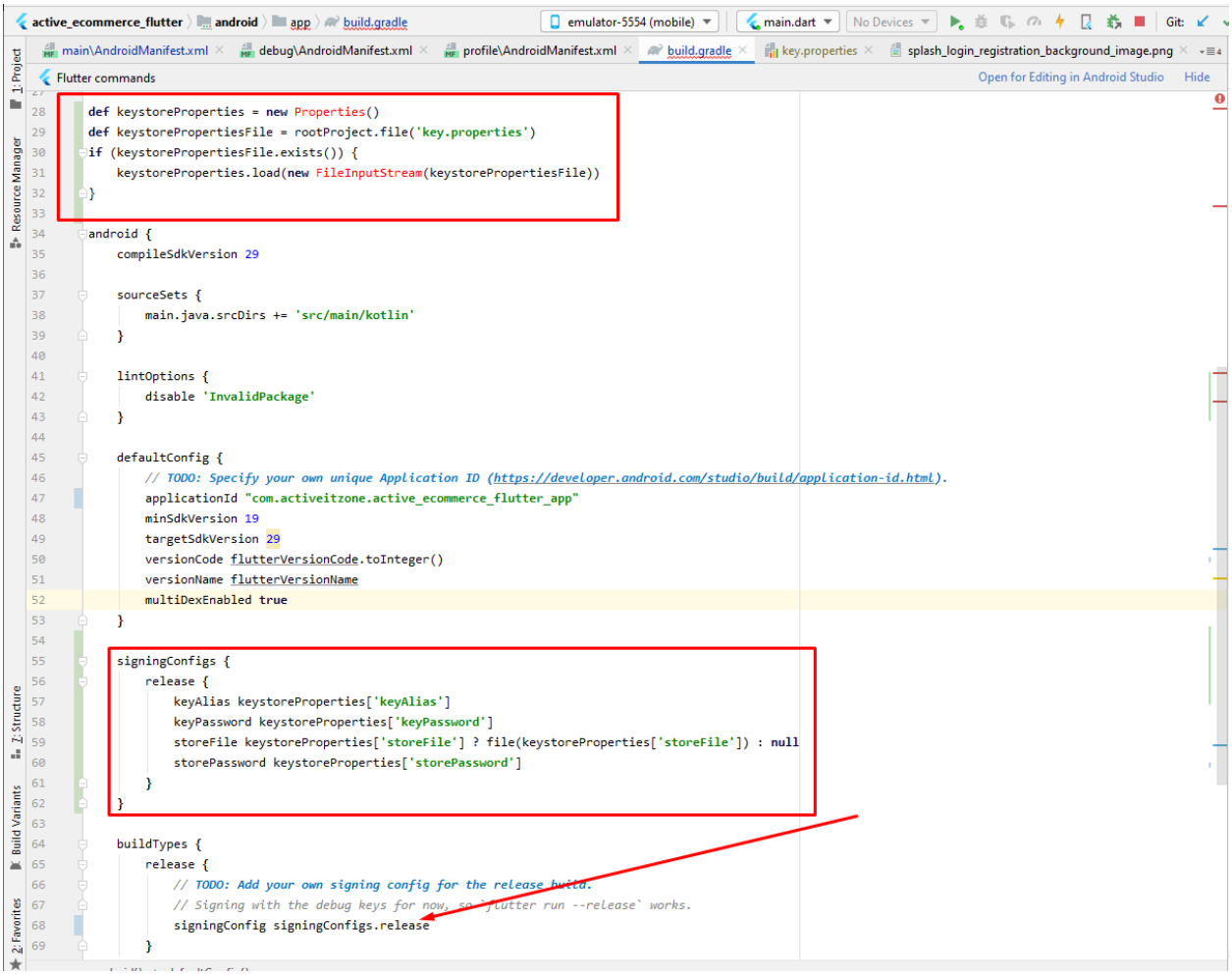
With the signing configuration info:

```
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

8. 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 scource_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?

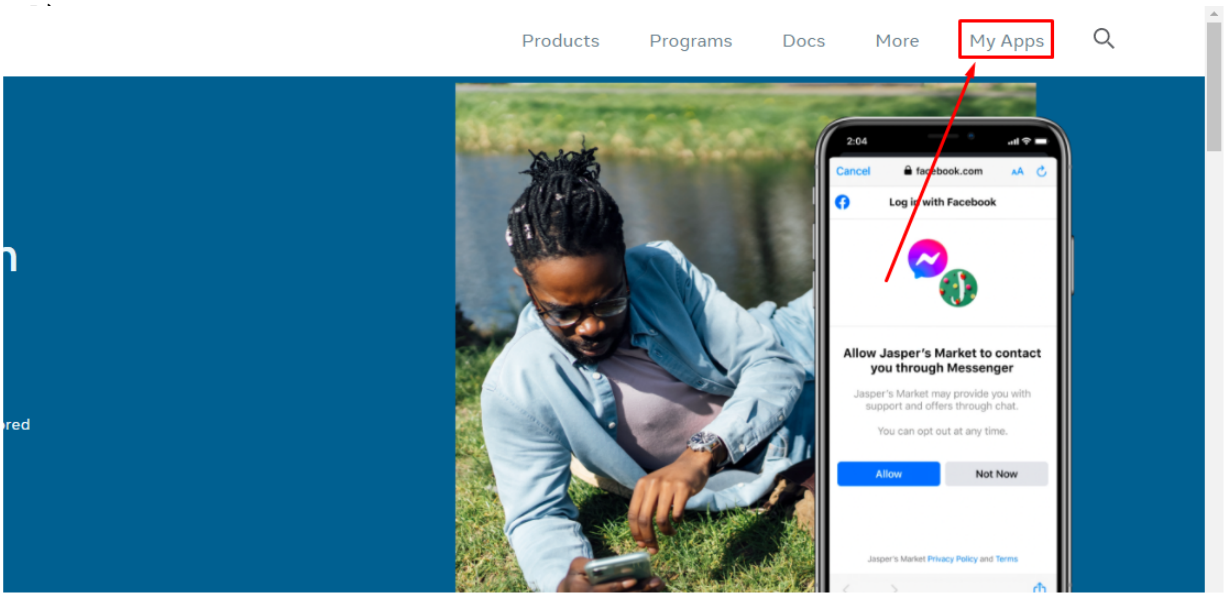
I) Facebook: Package Used

https://pub.dev/packages/flutter_facebook_auth

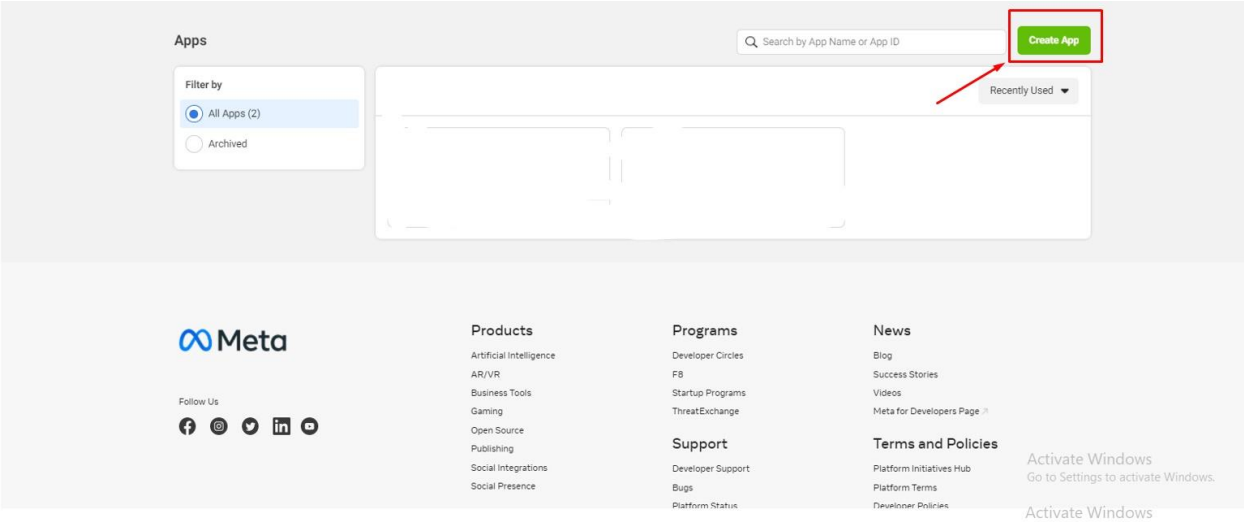
See its documentation and steps

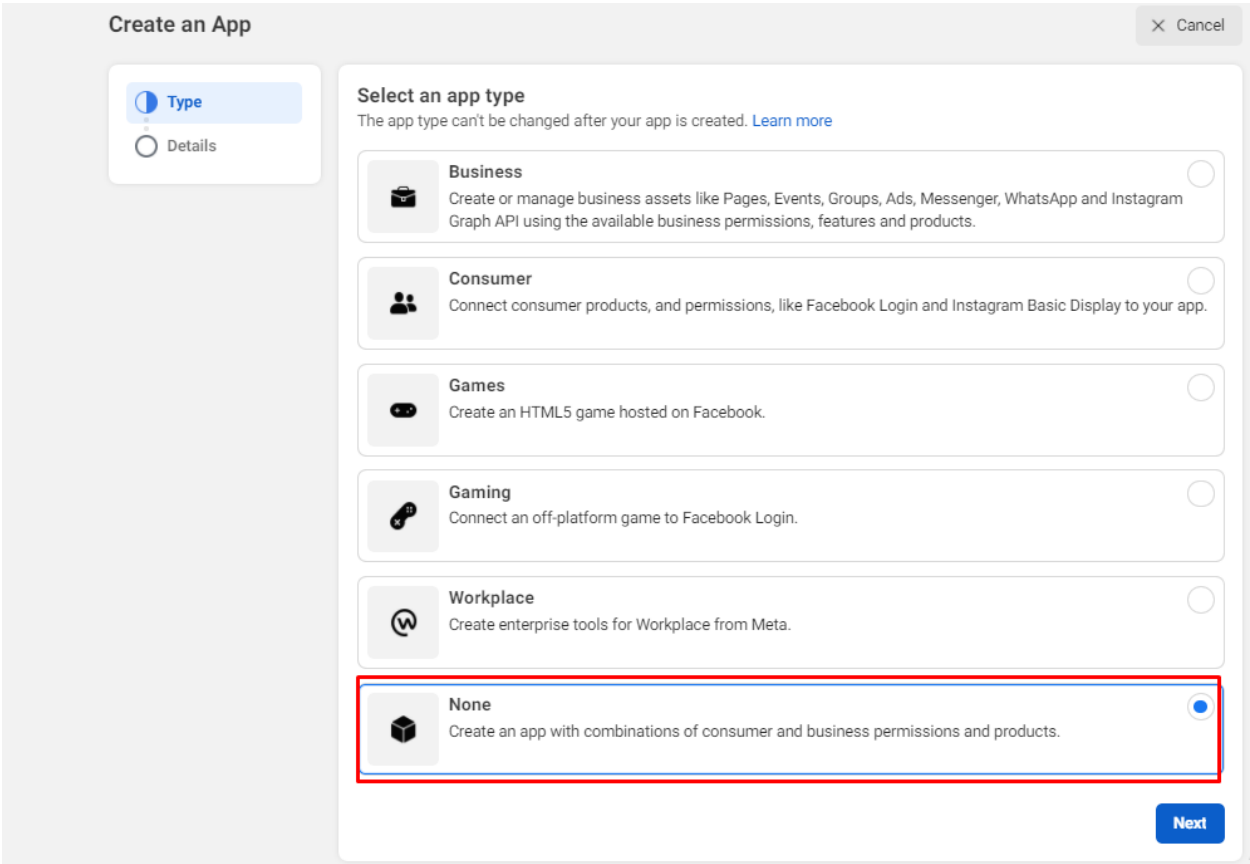
How to create a facebook app ?

A) First you have to login facebook in your browser.
B) Goto https://developers.facebook.com/
C) Goto My Apps.



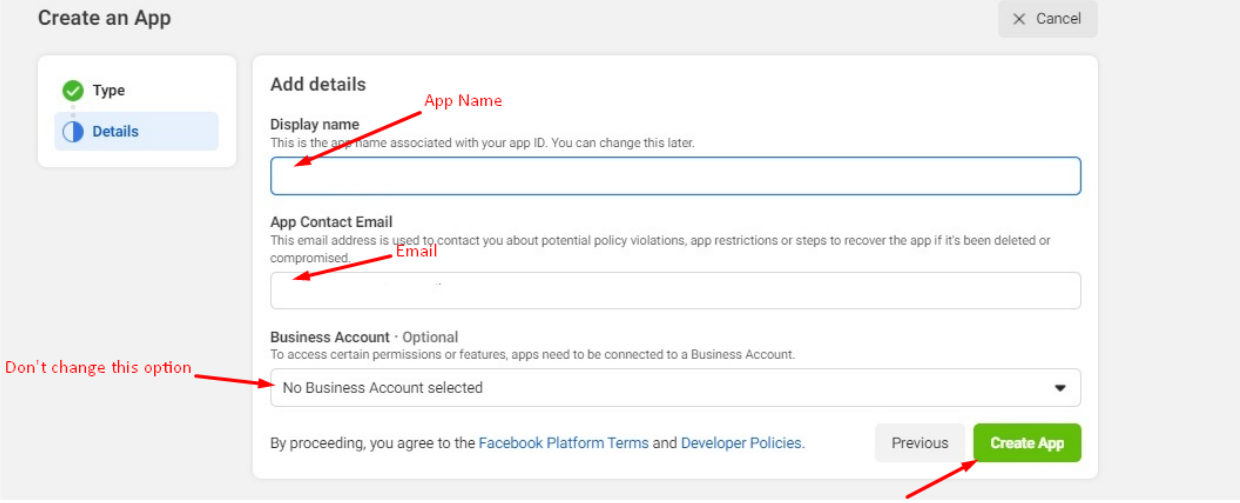
E) Create an app.





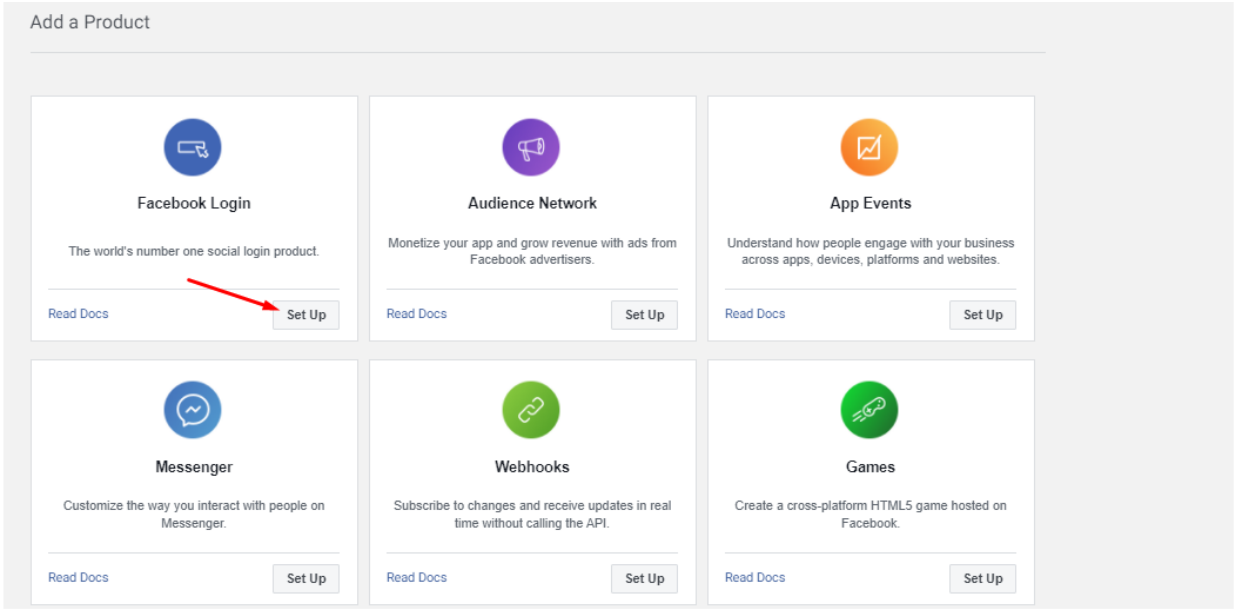
F) Add your app details

- App Name.
- App Contact Email.
- Business Account. (Don't Change it).

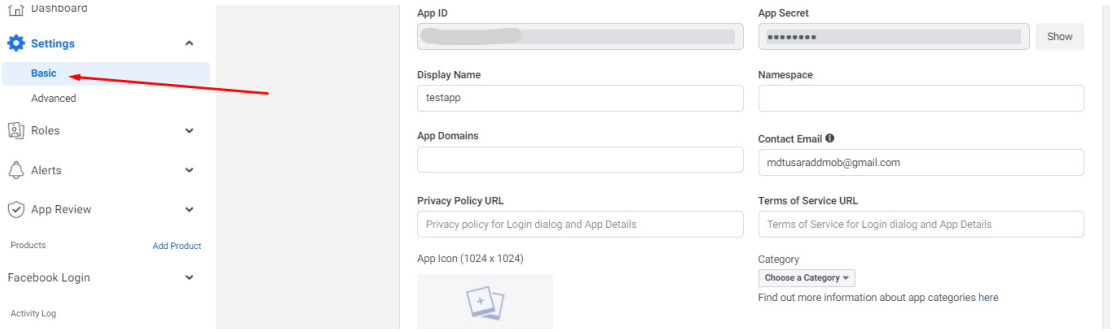


G) Add product "Facebook Login".

- ❖ Click the "Set Up" button.



H) Click the “Basic” Option.



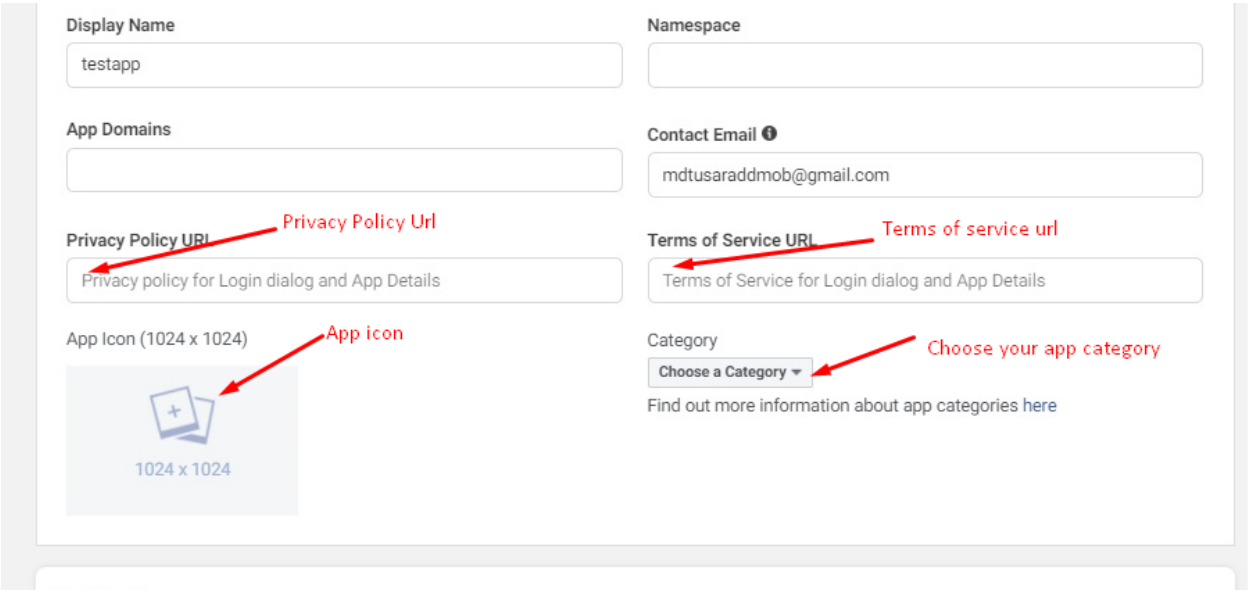
I) Add some information and save it.

I. This app privacy policy url.

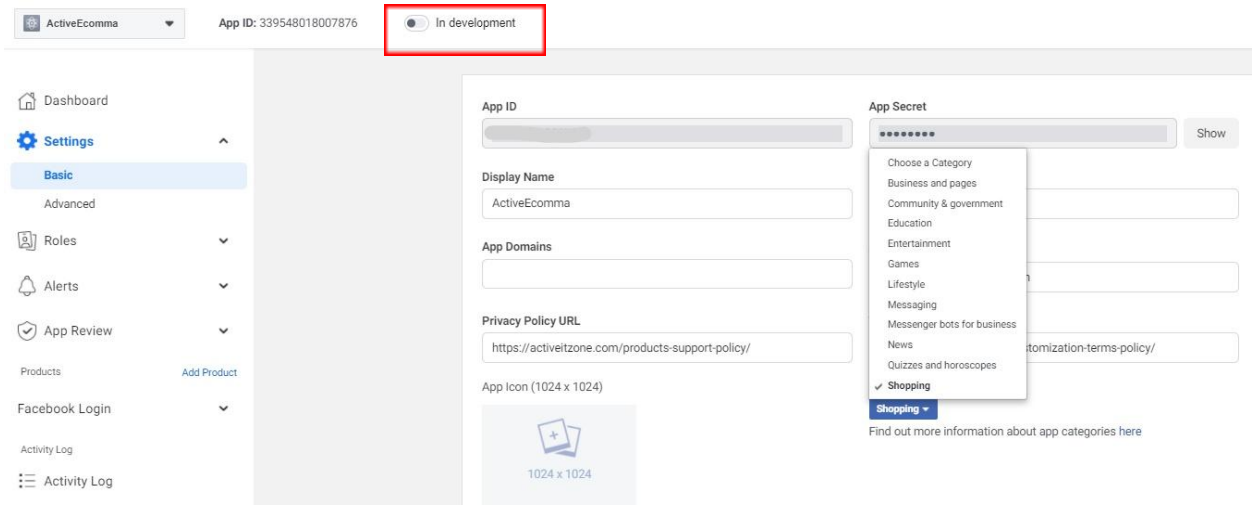
II. This app terms of service url.

III. This app icon (Icon size also 1024 X 1024 Or 512 X 512).

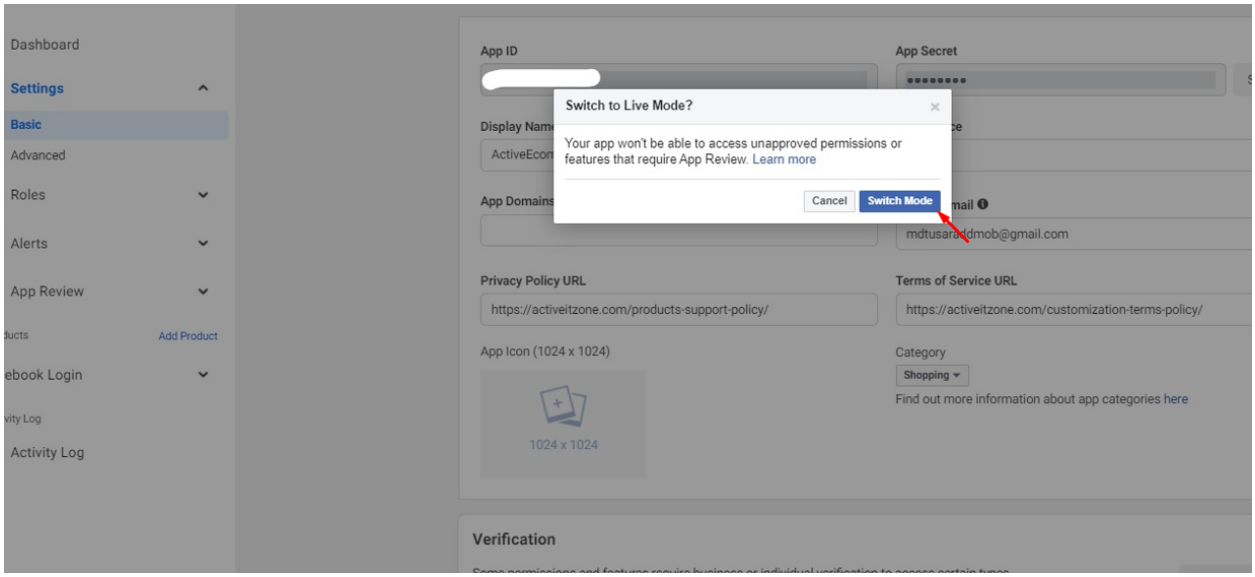
IV. This app category.



J) Now activate your app.



K) Click the “Switch Mode” Button.



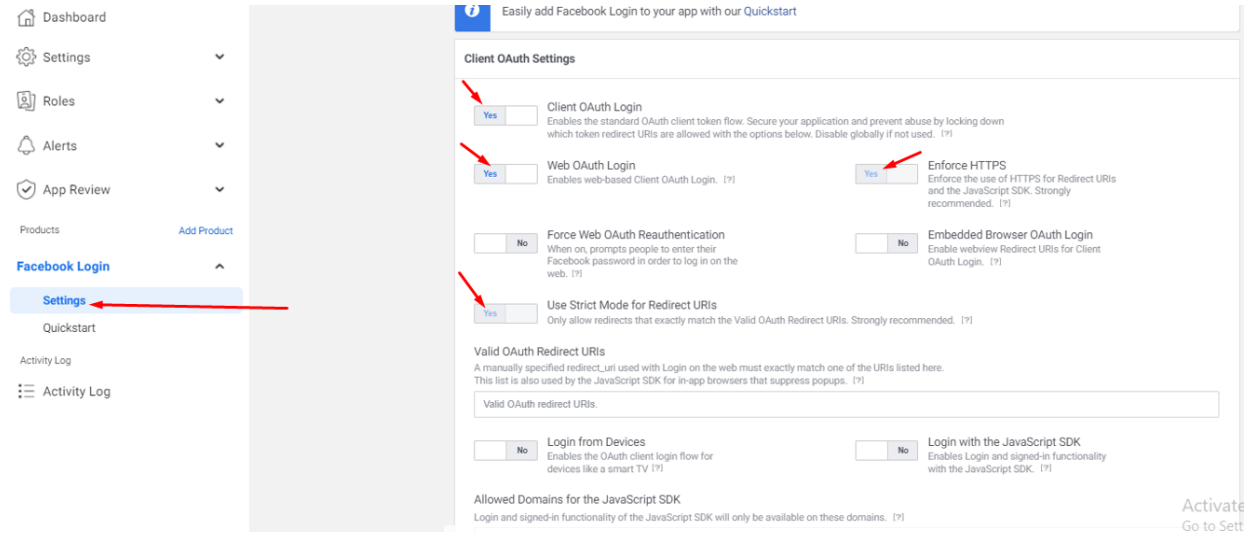
L) Make sure your app facebook login settings are on these options.

A) Client OAuth Login.

B) Web OAuth Login.

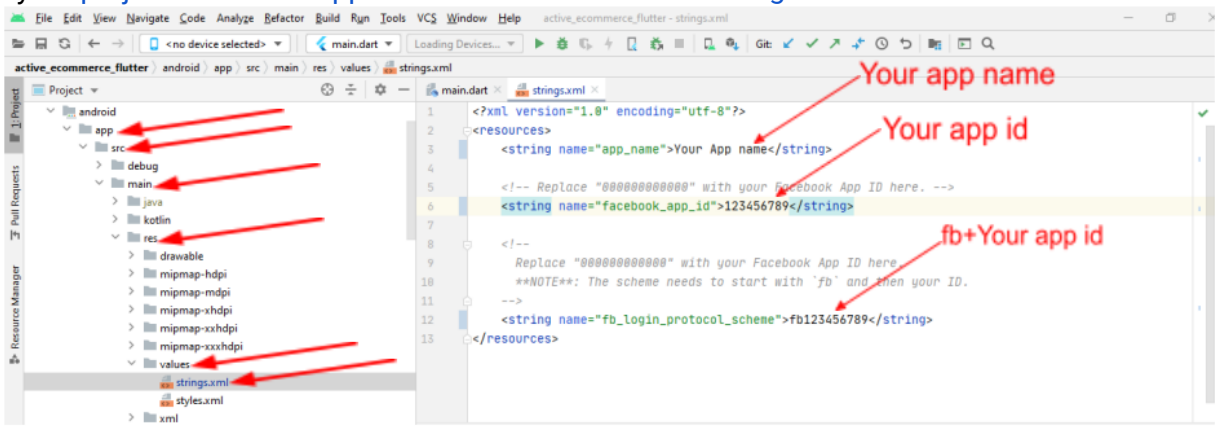
C) Enforce HTTPS

D) Use Strict Mode for Redirect URIs



M) Add your facebook app id, app name and fb login protocol scheme (***NOTE**: The scheme needs to start with `fb` and then your ID.example: fb123456789*)

into your project->android->app->src->main->res->values->string.xml file.



Google: Package Used

https://pub.dev/packages/google_sign_in

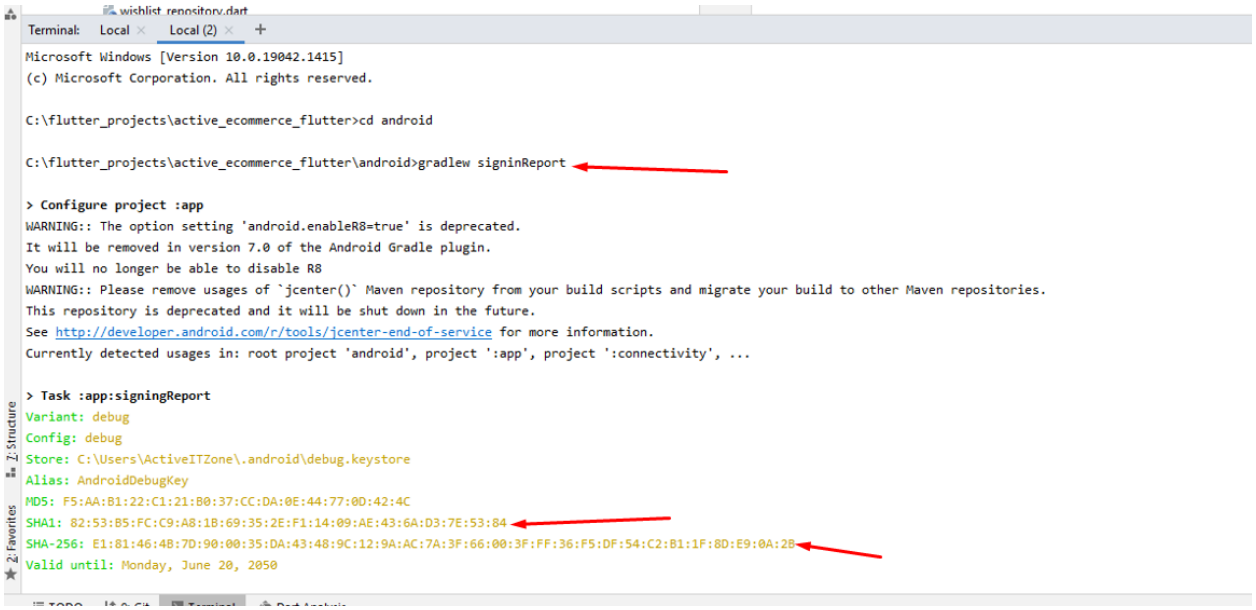
See its documentation and steps from the link.

1. Create a firebase project.
    A) First you need to have a google account and login in your browser.
    B) Goto this link: https://console.firebase.google.com/
    C) Click the “Add Project” Button.
    D) Add your project name and continue.
    E) Off the “disable google analytics” and Click the “Continue” Button.
    F) Click the “Continue” Button.

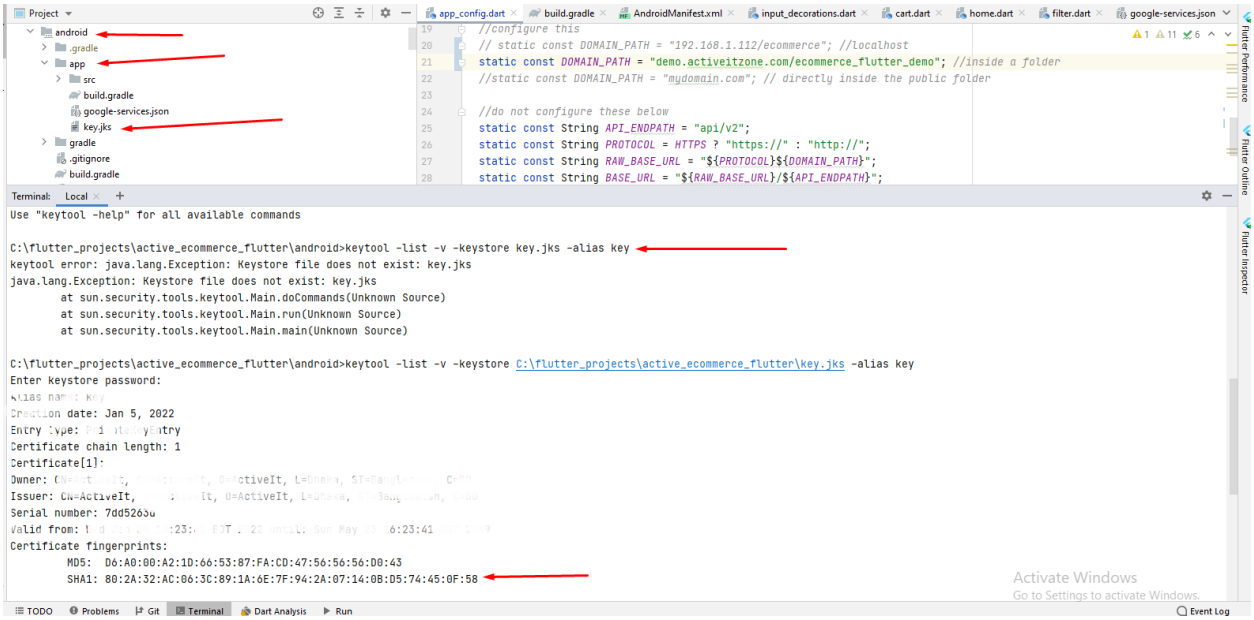
Add Android app on your Firebase project

1. Click the android icon.
   Add some informations:
    A) Android package name
    B) App nickname
    C) Add signing certificate SHA-1 and SHA-256

   How to get SHA-1 code in your project ?
    A. Goto android studio and open your project then open terminal and ensure that it shows your project directory in the terminal.
    B. Then drive into the android folder(cd android).
    C. i)For debug mode: write this command on your android studio terminal “gradlew signinReport” then press enter.

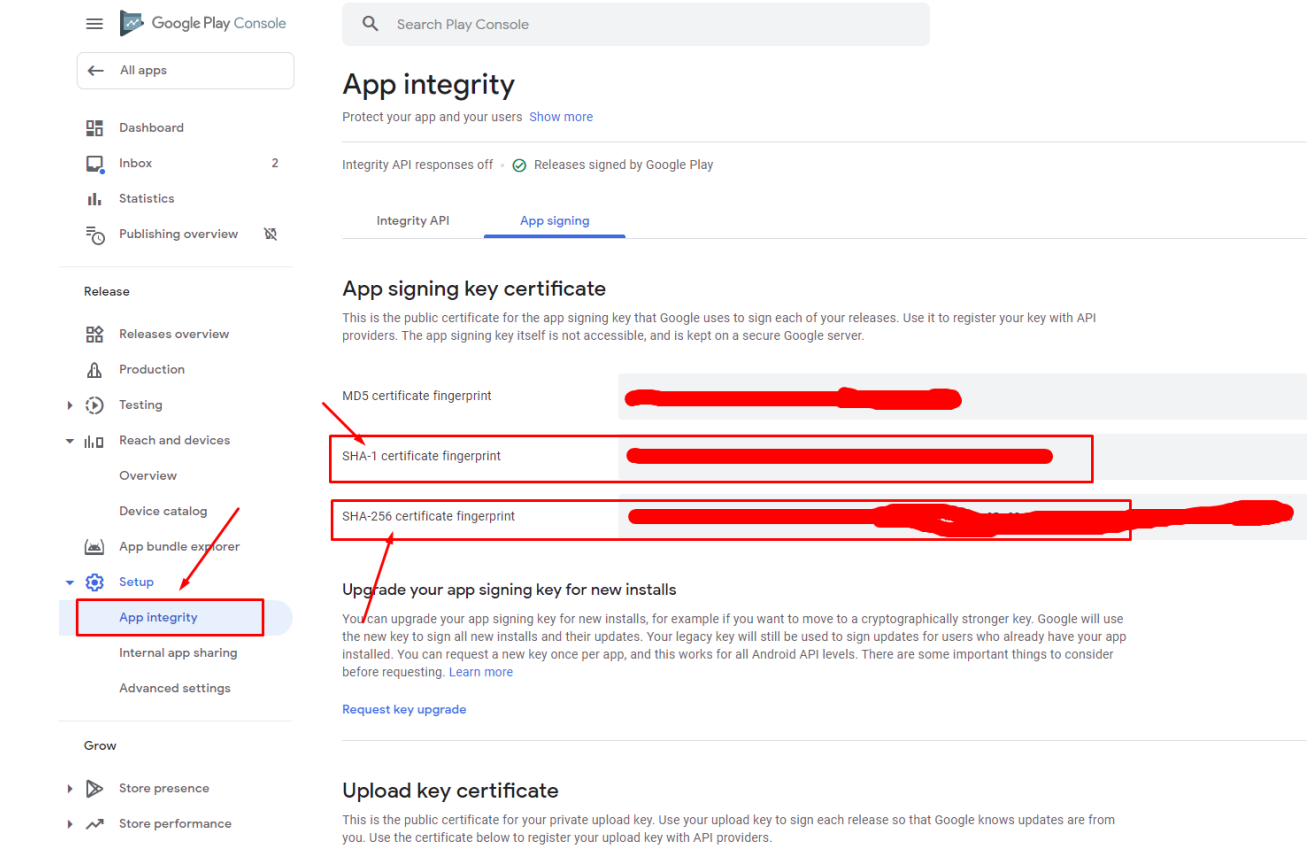


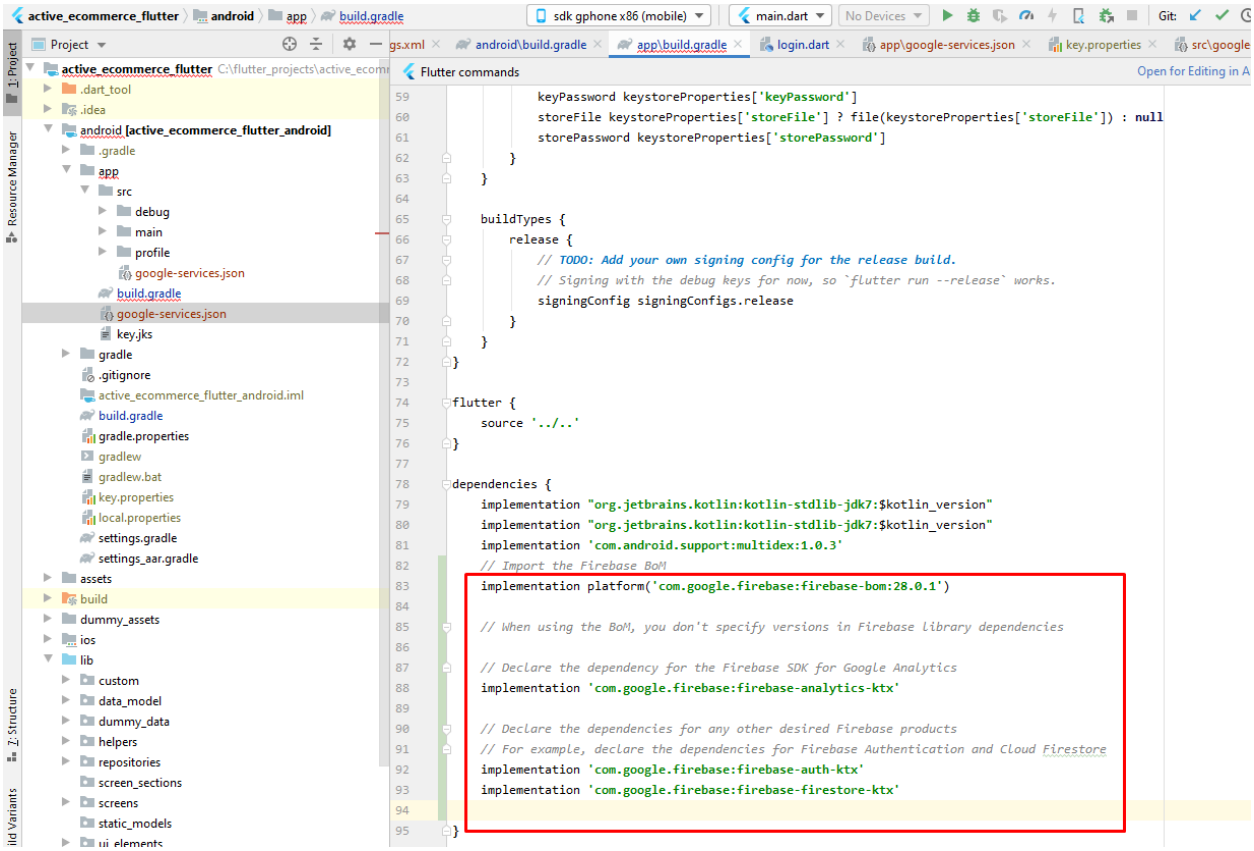
ii) For release mode: write this command on your android studio terminal “keytool -list -v -keystore YOUR_PROJECT_DIRECTORY\key.jks -alias key” then press enter.

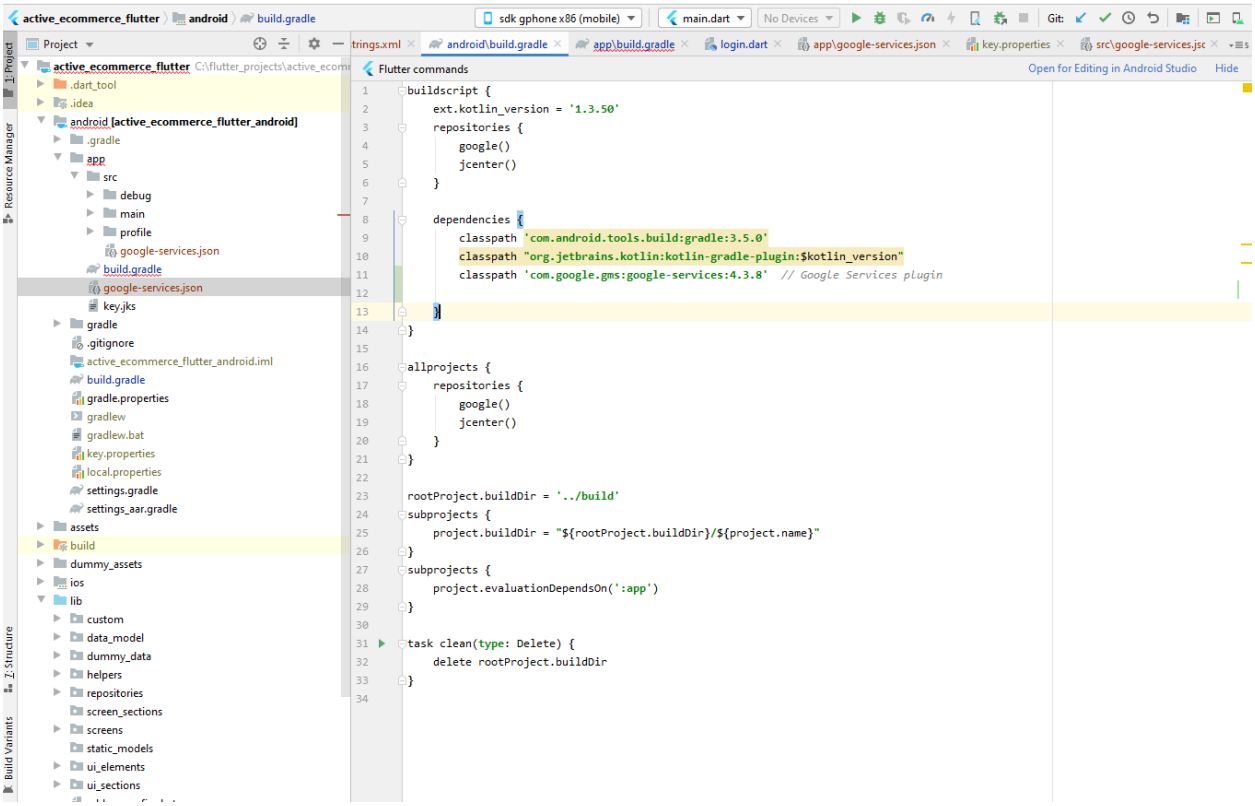


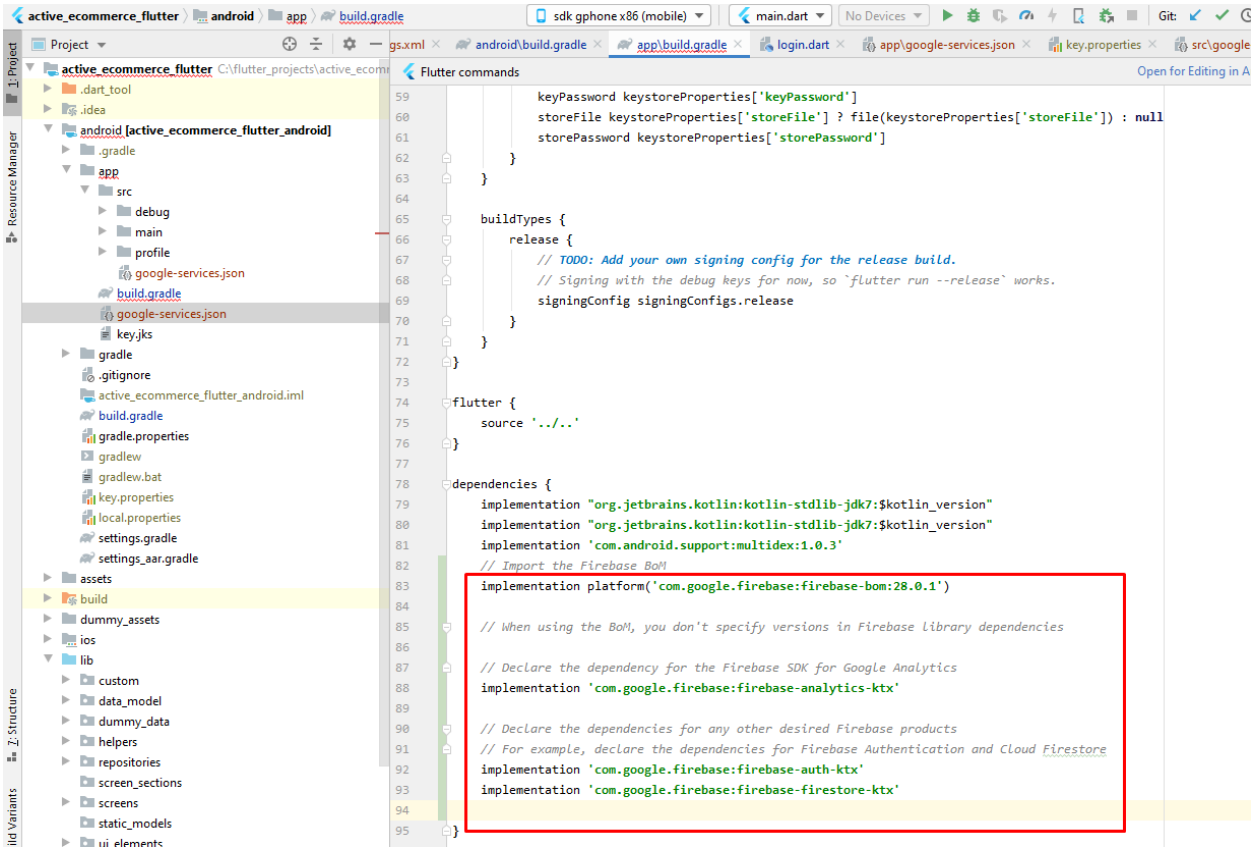
2. Click the “Register App” button
3. Download config file and add this file into your project->android->app folder
4. Add firebase SDK
5. Click the “Continue to console” button

6) After upload your app on playstore you need to add signing certificate SHA-1 and SHA-256 in firebase project setting : Open your google play console->Your App->Setup->App integrity

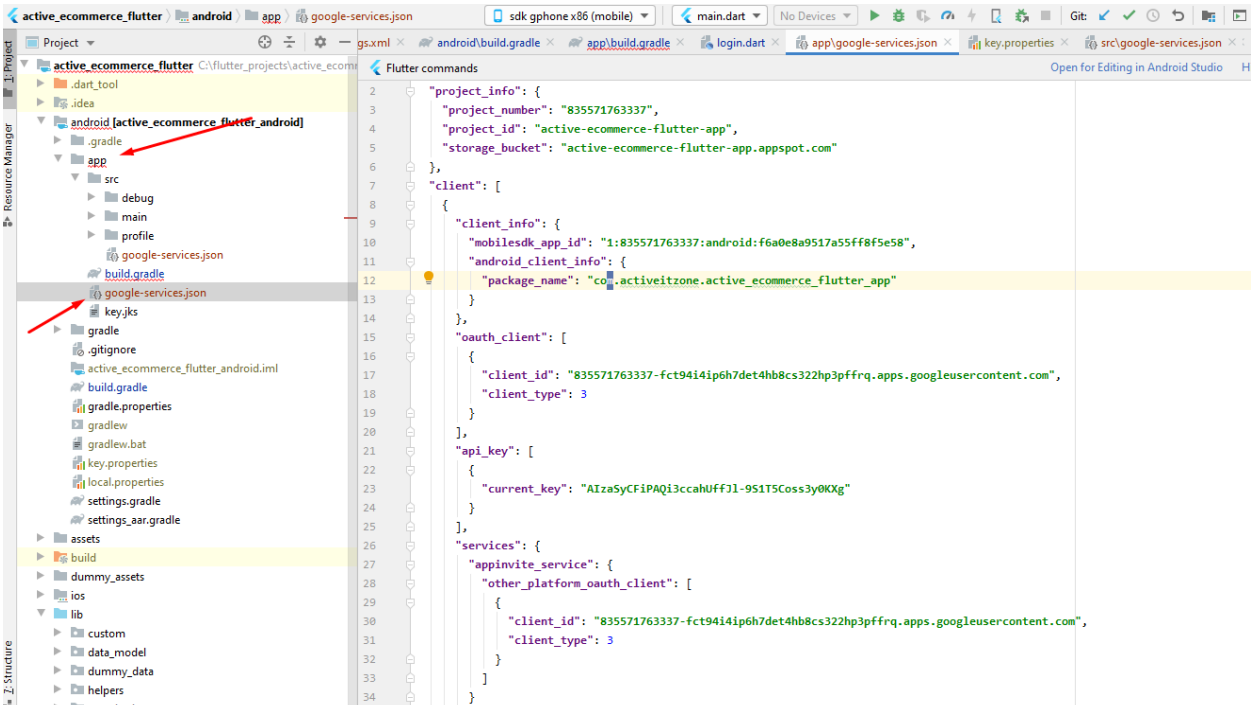

Google Play Console
Search Play Console
All apps
Dashboard
Inbox 2
Statistics
Publishing overview
Release
Releases overview
Production
Testing
Reach and devices
Overview
Device catalog
App bundle explorer
Setup
App integrity
Internal app sharing
Advanced settings
Grow
Store presence
Store performance
App integrity
Protect your app and your users Show more
Integrity API responses off
Releases signed by Google Play
Integrity API
App signing
App signing key certificate
This is the public certificate for the app signing key that Google uses to sign each of your releases. Use it to register your key with API providers. The app signing key itself is not accessible, and is kept on a secure Google server.
MD5 certificate fingerprint
SHA-1 certificate fingerprint
SHA-256 certificate fingerprint
Upgrade your app signing key for new installs
You can upgrade your app signing key for new installs, for example if you want to move to a cryptographically stronger key. Google will use the new key to sign all new installs and their updates. Your legacy key will still be used to sign updates for users who already have your app installed. You can request a new key once per app, and this works for all Android API levels. There are some important things to consider before requesting. Learn more
Request key upgrade
Upload key certificate
This is the public certificate for your private upload key. Use your upload key to sign each release so that Google knows updates are from you. Use the certificate below to register your upload key with API providers.


active_ecommerce_flutter
android
app
build.gradle
sdk gphone x86 (mobile)
main.dart
No Devices
Git:
Project
Flutter commands
Open for Editing in A
active_ecommerce_flutter C:\flutter_projects\active_ecomm
.dart_tool
.idea
android [active_ecommerce_flutter_android]
.gradle
app
src
debug
main
profile
google-services.json
build.gradle
google-services.json
key.jks
gradle
.gitignore
active_ecommerce_flutter_android.iml
build.gradle
gradle.properties
gradlew
gradlew.bat
key.properties
local.properties
settings.gradle
settings_aar.gradle
assets
build
dummy_assets
ios
lib
custom
data_model
dummy_data
helpers
repositories
screen_sections
screens
static_models
ui_elements
keyPassword keystoreProperties['keyPassword']
storeFile keystoreProperties['storeFile'] ? file(keystoreProperties['storeFile']) : null
storePassword keystoreProperties['storePassword']
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.release
flutter {
source '../..'
dependencies {
implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
implementation 'com.android.support:multidex:1.0.3'
// Import the Firebase BoM
implementation platform('com.google.firebase:firebase-bom:28.0.1')
// When using the BoM, you don't specify versions in Firebase library dependencies
// Declare the dependency for the Firebase SDK for Google Analytics
implementation 'com.google.firebase:firebase-analytics-ktx'
// Declare the dependencies for any other desired Firebase products
// For example, declare the dependencies for Firebase Authentication and Cloud Firestore
implementation 'com.google.firebase:firebase-auth-ktx'
implementation 'com.google.firebase:firebase-firestore-ktx'


active_ecommerce_flutter
android
build.gradle
sdk gphone x86 (mobile)
main.dart
No Devices
Git:
Project
Flutter commands
Open for Editing in Android Studio
Hide
android\build.gradle
app\build.gradle
login.dart
app\google-services.json
key.properties
buildscript {
ext.kotlin_version = '1.3.50'
repositories {
google()
jcenter()
}
dependencies {
classpath 'com.android.tools.build:gradle:3.5.0'
classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
classpath 'com.google.gms:google-services:4.3.8' // Google Services plugin
}
}
allprojects {
repositories {
google()
jcenter()
}
}
rootProject.buildDir = '../build'
subprojects {
project.buildDir = "${rootProject.buildDir}/${project.name}"
}
subprojects {
project.evaluationDependsOn(':app')
}
task clean(type: Delete) {
delete rootProject.buildDir
}

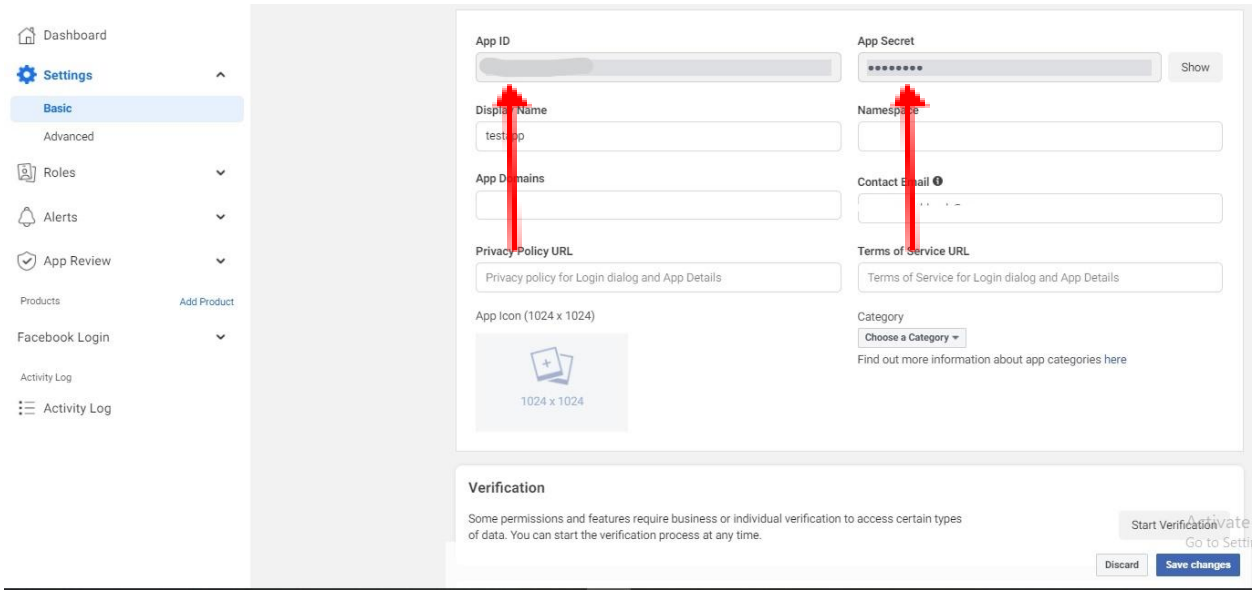

active_ecommerce_flutter
android
app
build.gradle
sdk gphone x86 (mobile)
main.dart
No Devices
Git:
Project
Flutter commands
Open for Editing in A
android\build.gradle
app\build.gradle
login.dart
app\google-services.json
key.properties
keyPassword keystoreProperties['keyPassword']
storeFile keystoreProperties['storeFile'] ? file(keystoreProperties['storeFile']) : null
storePassword keystoreProperties['storePassword']
}
}
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.release
}
}
}
flutter {
source '../..'
}
dependencies {
implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
implementation 'com.android.support:multidex:1.0.3'
// Import the Firebase BoM
implementation platform('com.google.firebase:firebase-bom:28.0.1')
// When using the BoM, you don't specify versions in Firebase library dependencies
// Declare the dependency for the Firebase SDK for Google Analytics
implementation 'com.google.firebase:firebase-analytics-ktx'
// Declare the dependencies for any other desired Firebase products
// For example, declare the dependencies for Firebase Authentication and Cloud Firestore
implementation 'com.google.firebase:firebase-auth-ktx'
implementation 'com.google.firebase:firebase-firestore-ktx'
}

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



Firebase Authentication :

1. Goto signin method and then enable Google and facebook.
2. For facebook you need a facebook app id and app secret, previously we created. Add these documents.



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

Follow the guideline from here https://pub.dev/packages/google_sign_in

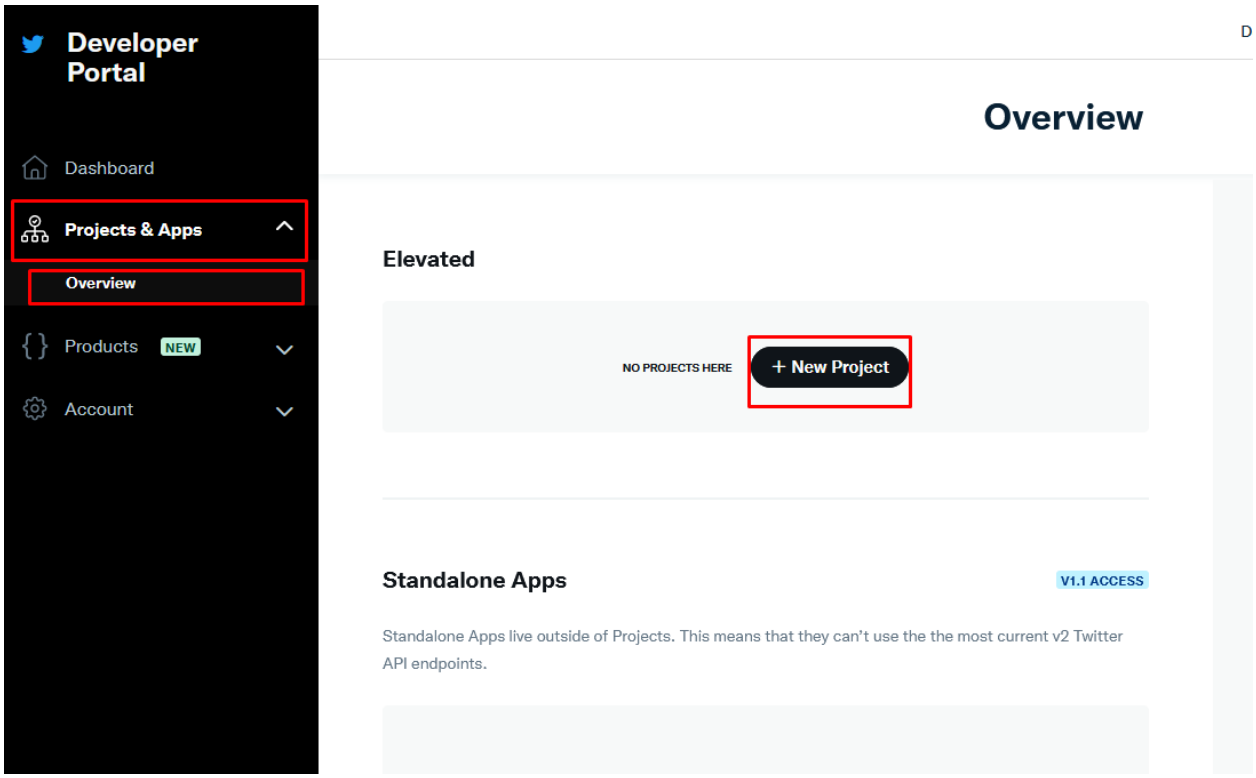
Twitter : package used

https://pub.dev/packages/twitter_login

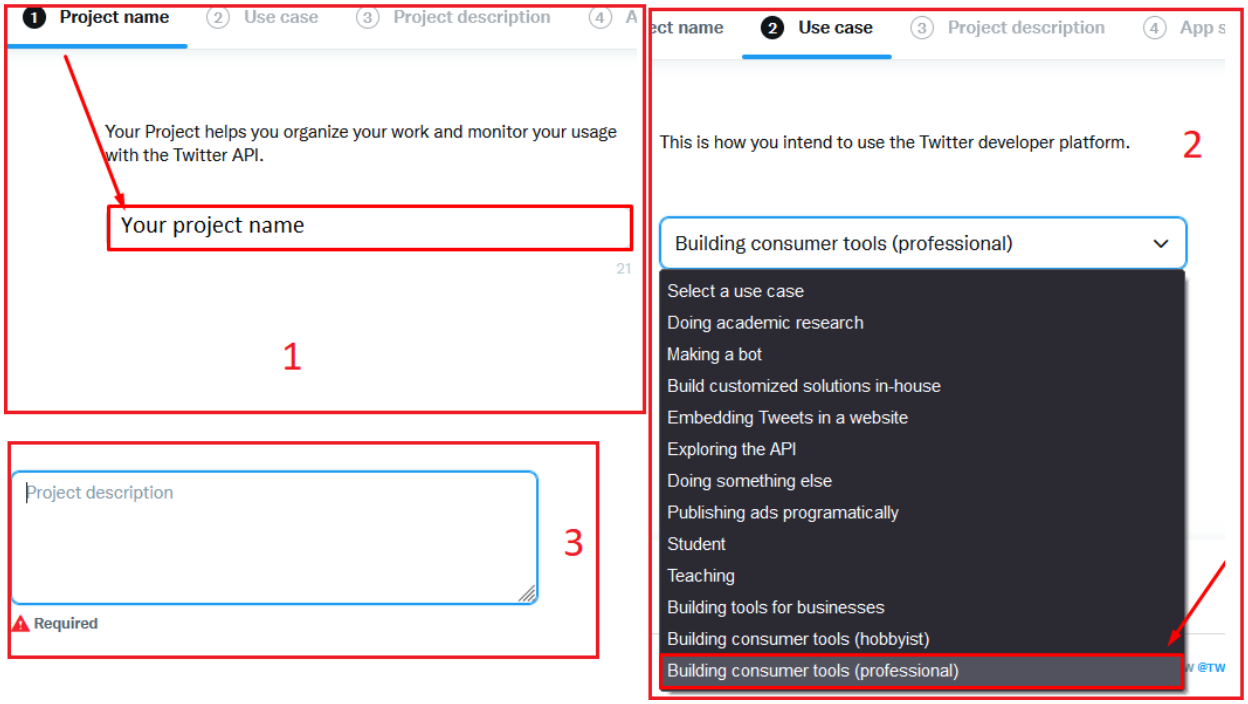
How to create a Twitter app.

1.1. First you have to login twitter in your browser.

1.2. Go to https://developer.twitter.com/
1.3. Go to Developer Portal.
1.4. Go to Project & Apps->Overview create a new project.



1.5. Filup some information for your project. Project Name, Use case, Project description.

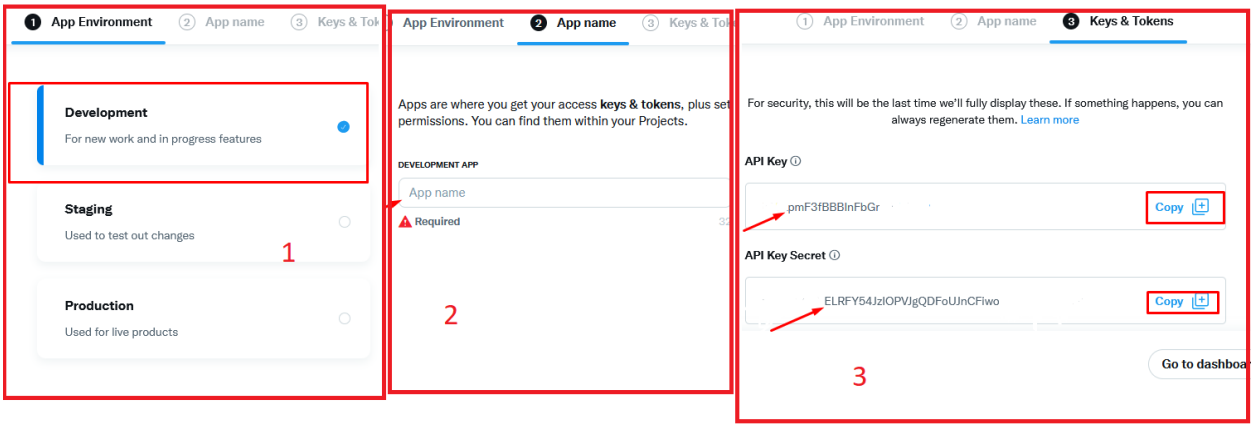


2. Setup your app

2.1. Select your app environment.

2.2. Enter your app name.

2.3. Collect your **API Key** and **API Key Secret.**(you will have to collect it.)



3. Setup user authentication setting.

3.1. Go to Project & Apps-> select your project and then select your created app.

3.2. Find **User authentication settings.**

3.3. Click the **Setup** button.

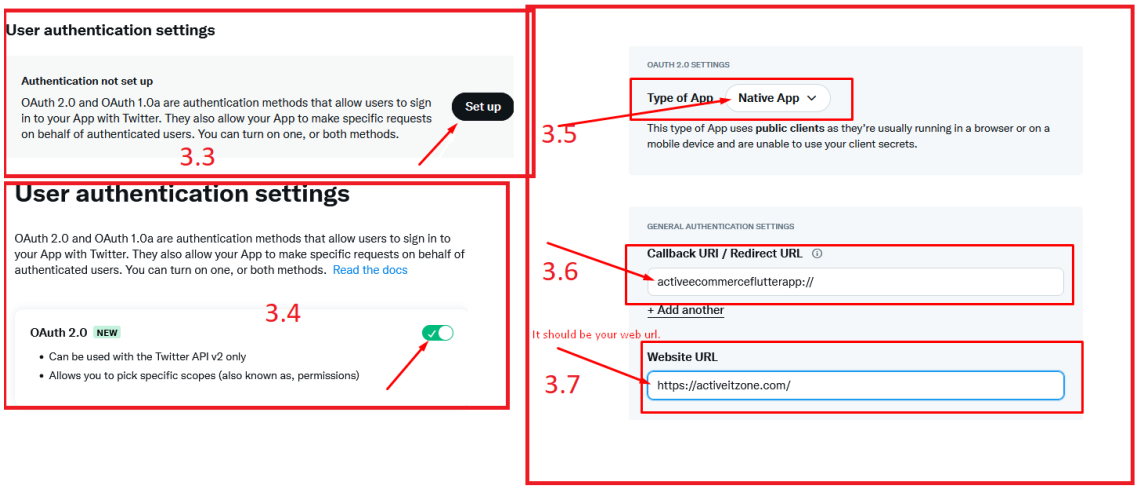
3.4. Enable OAuth 2.0

3.5. Select **Type of App.(Native app)**

3.6. Callback URI.(**`activeecommerceflutterapp://`**).

3.7. Web site URL.(It's your web URL).

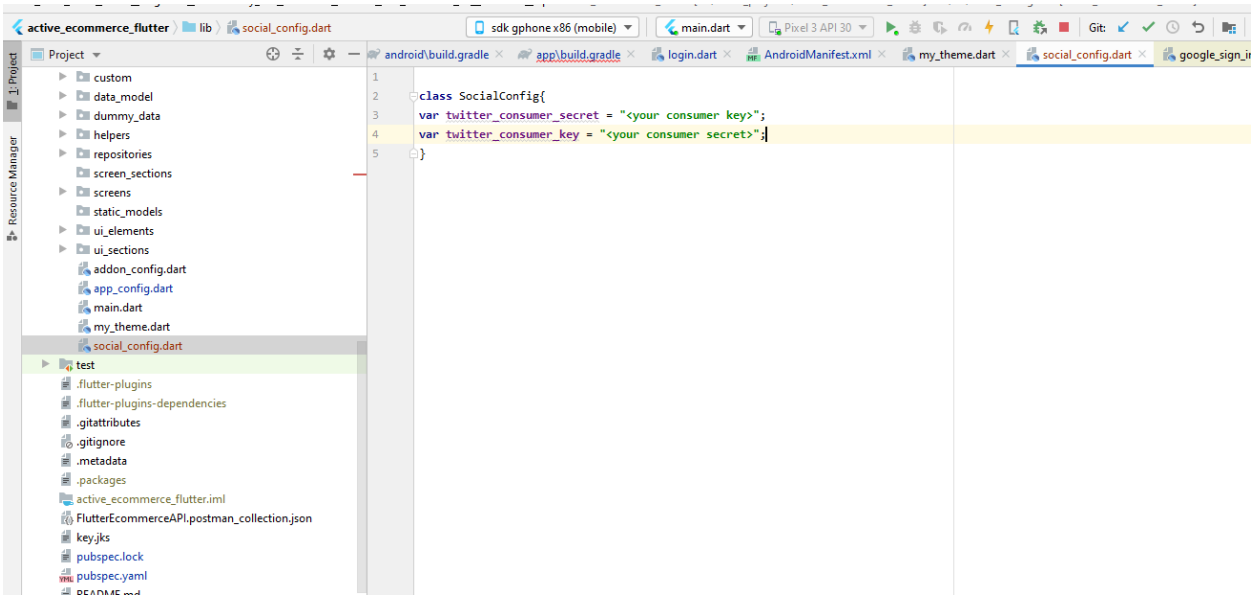
3.8. Save it.



4. Setup in your flutter e-commerce code.

4.1. Go to your flutter project->lib->social_config.dart

4.2. Enter your `twitter_consumer_secret` **and** `twitter_consumer_key`.

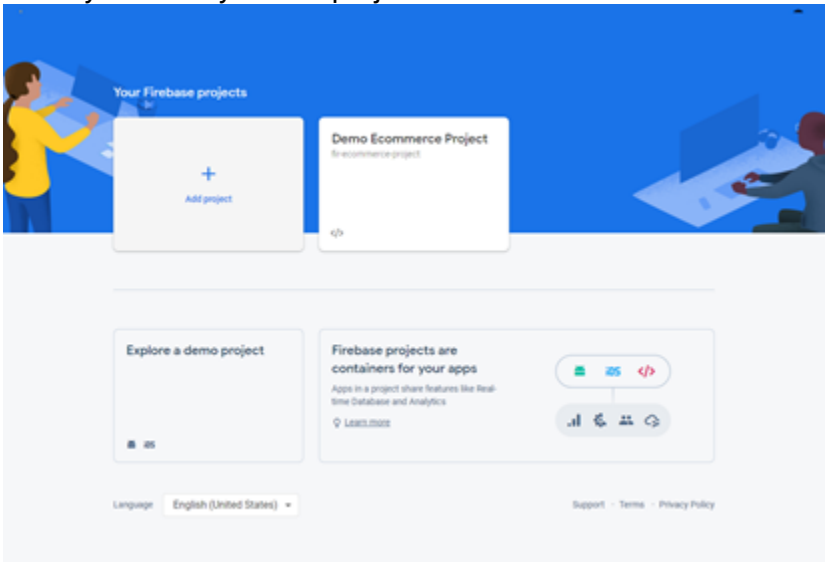


10. How to configure push notification?

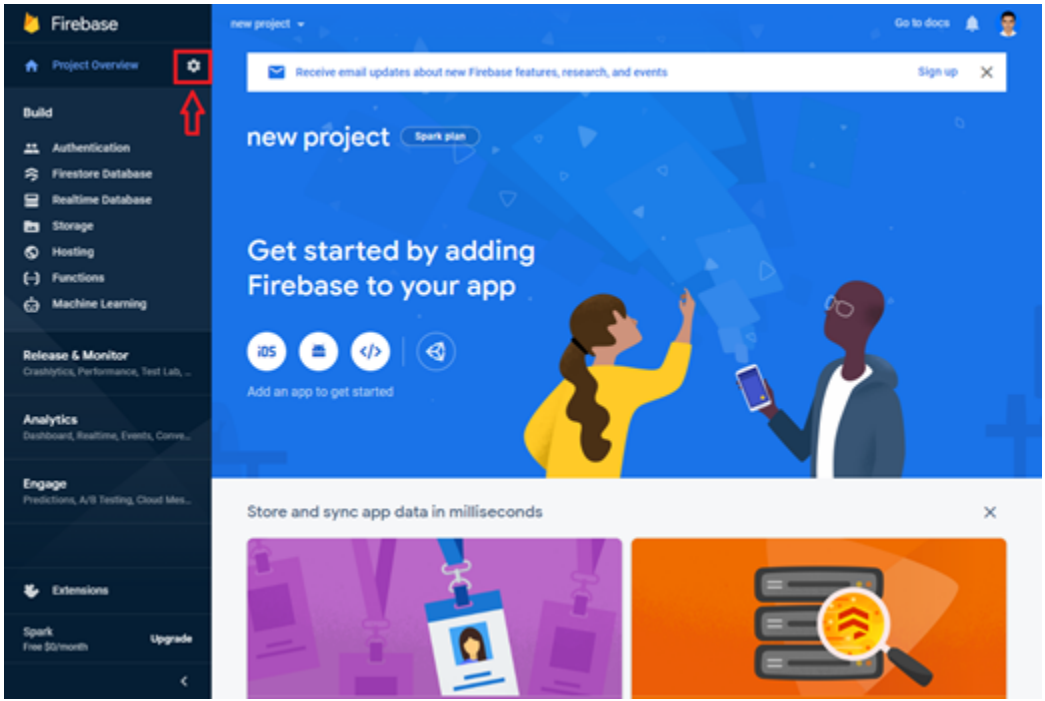
To use firebase follow the procedure which are mentioned below

1. Go to this URL to create project https://console.firebase.google.com/u/0/

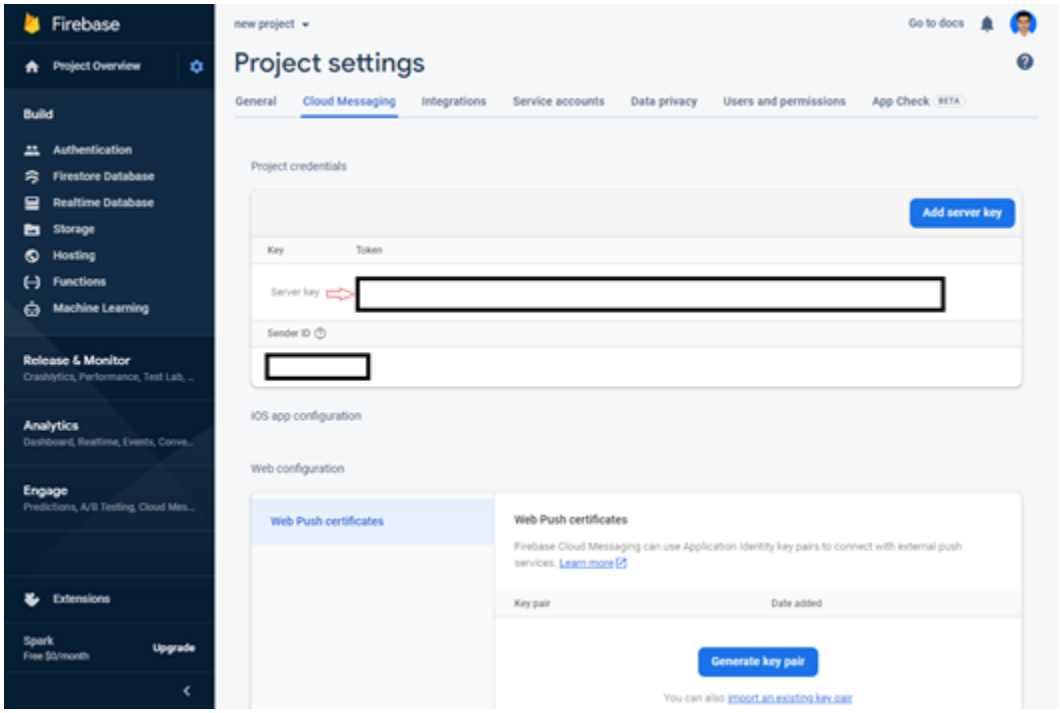
   If you already have a project then continue with that.



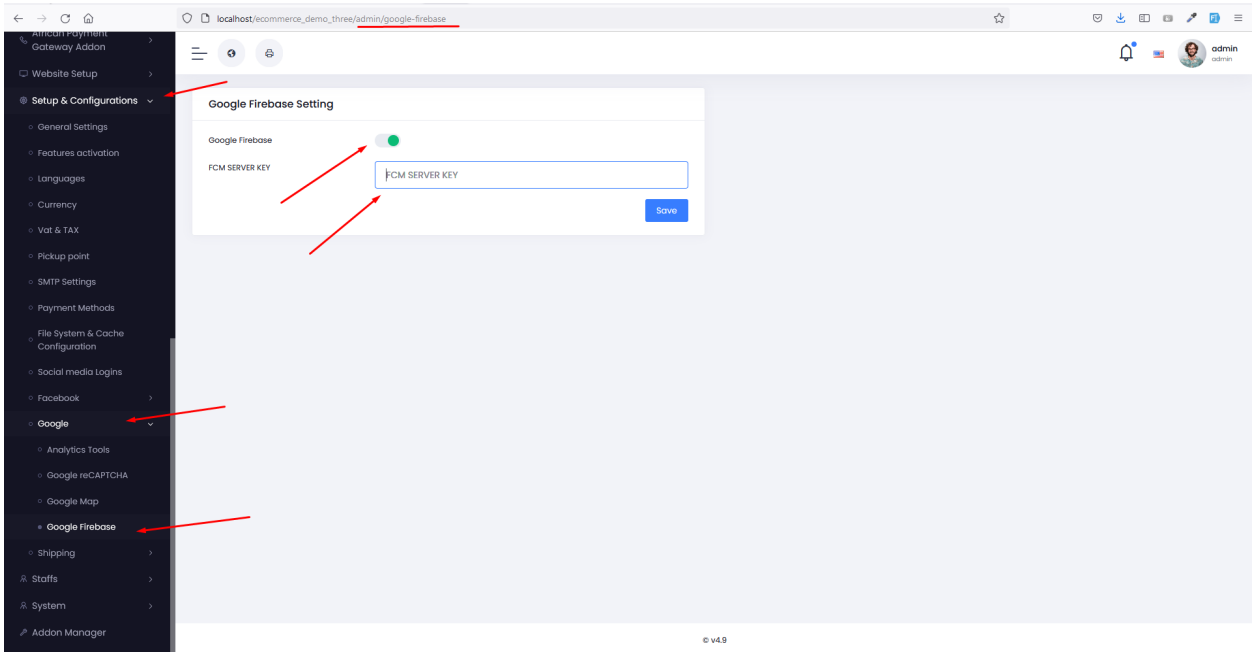
2. Now go to project settings to get server key



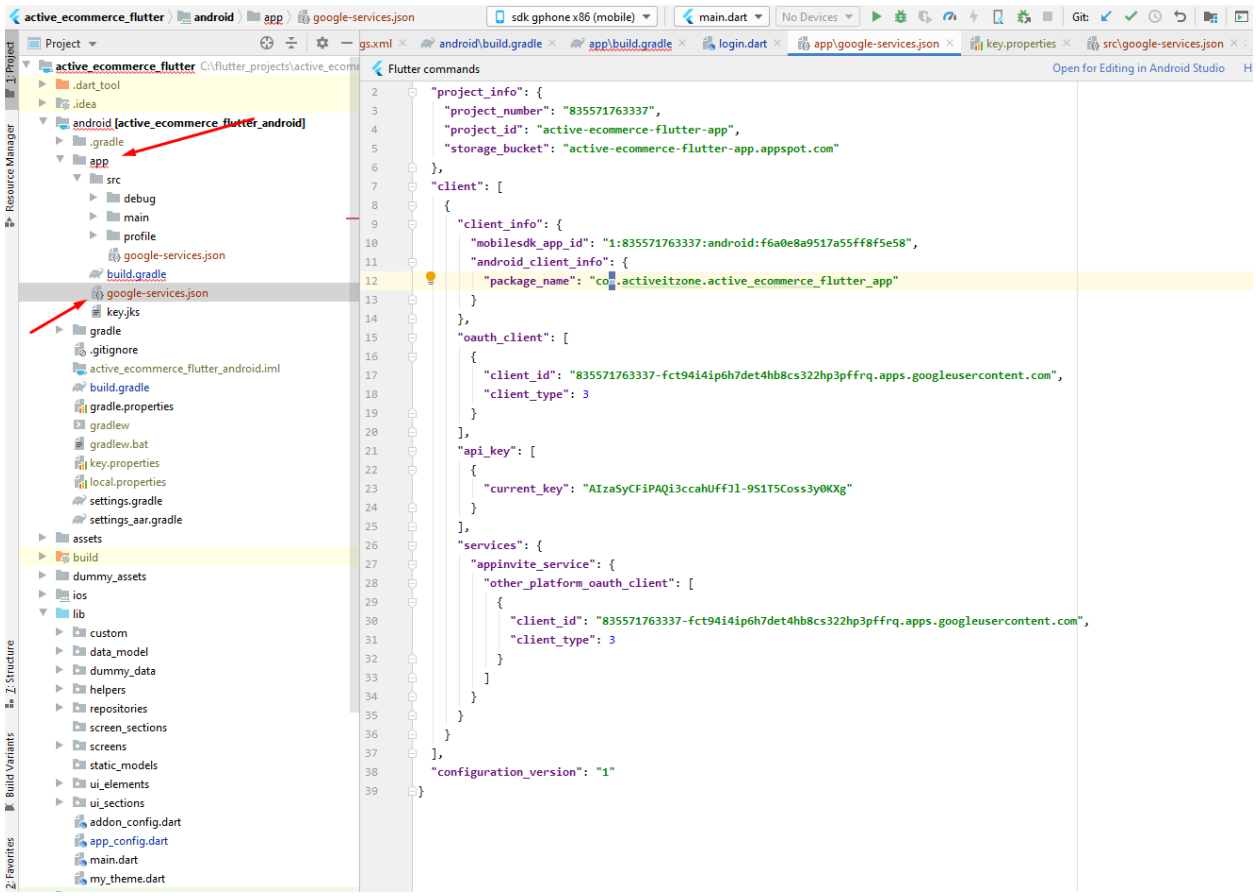
3. To get server key click on Cloud Messaging option



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

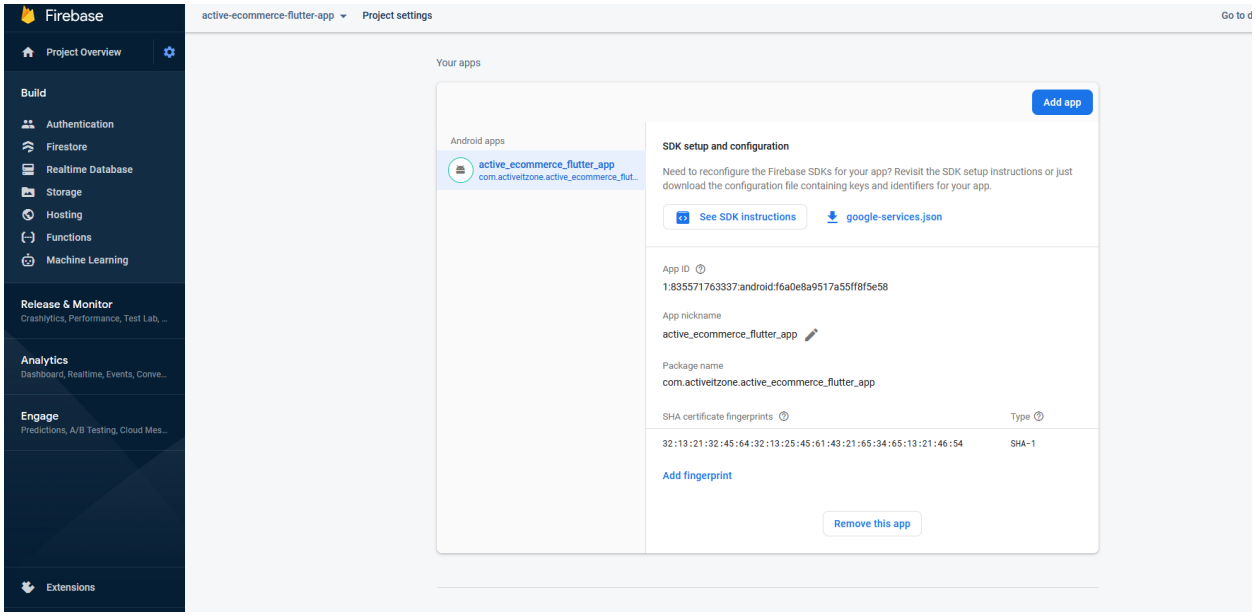


Firebase console:

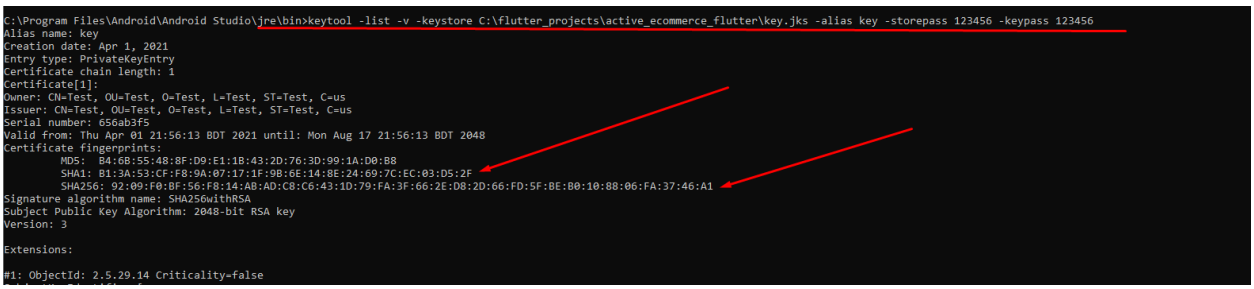
https://console.firebase.google.com/u/0/

Follow the guideline from here 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.



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

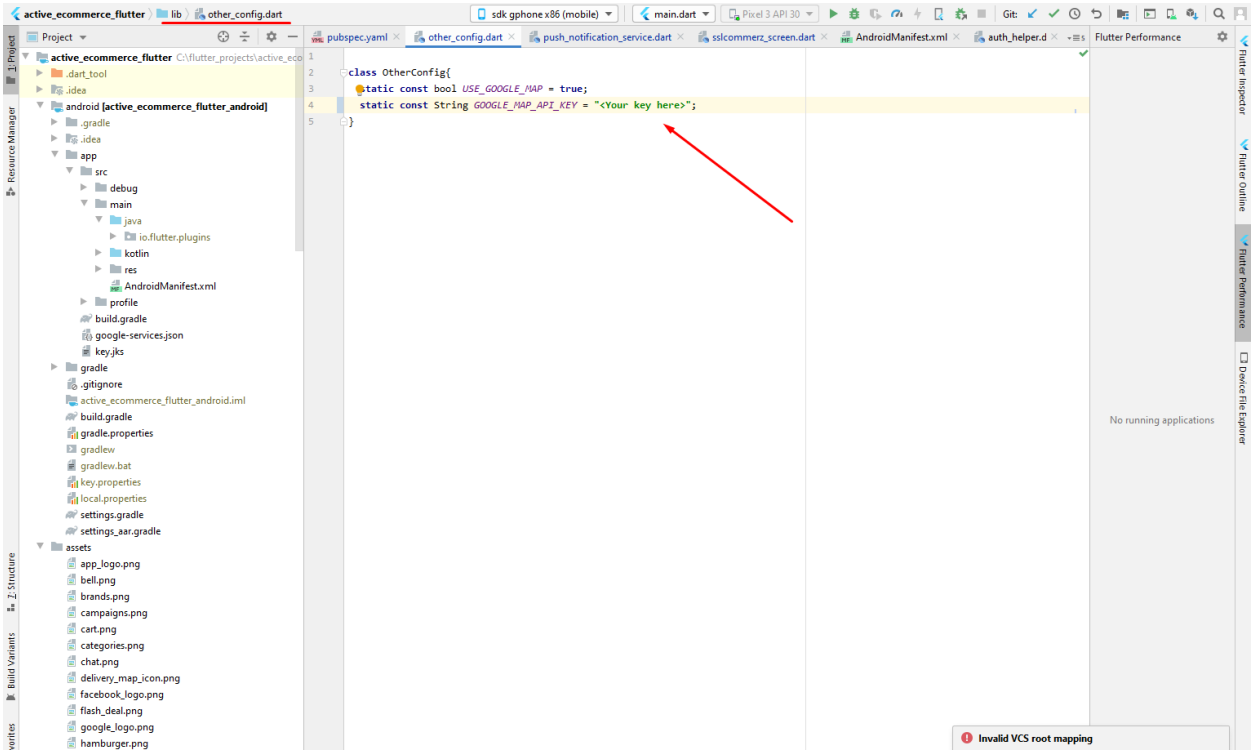
8. For ios follow this https://firebase.google.com/docs/cloud-messaging/ios/client

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.

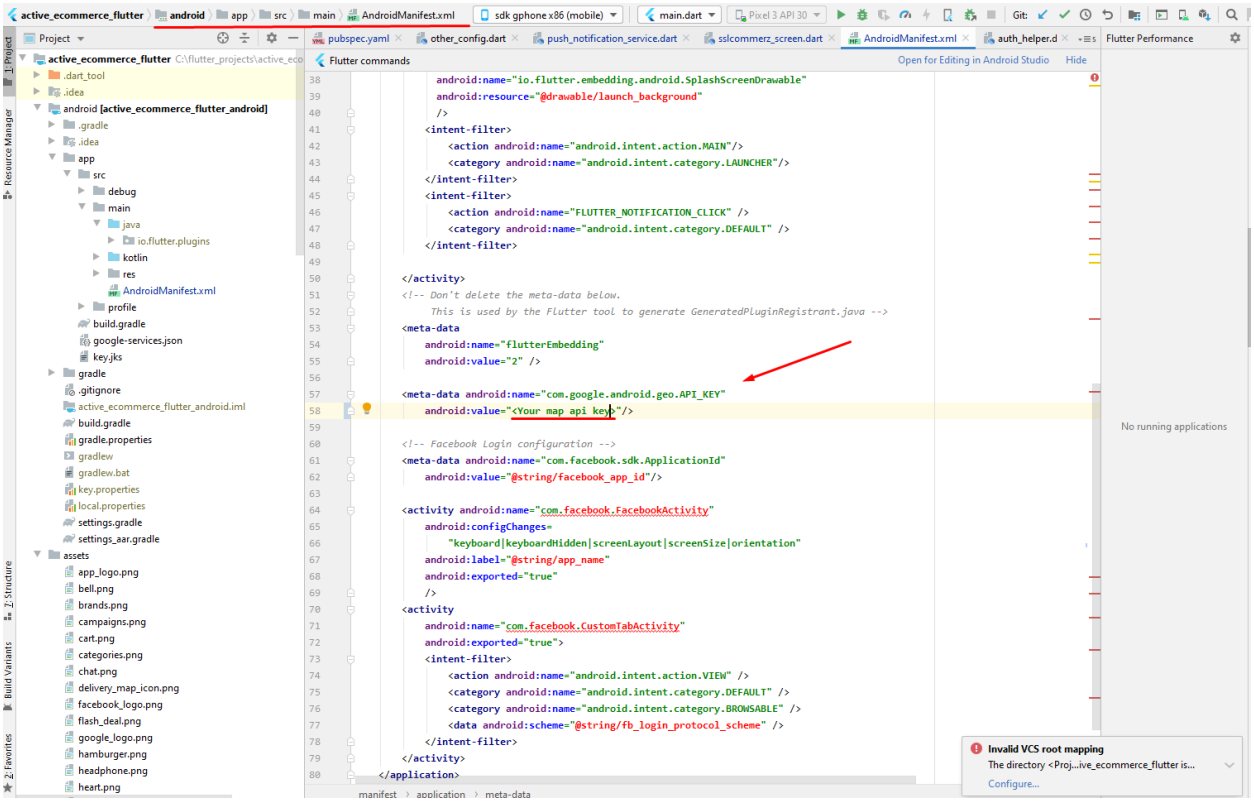
11. How to configure google map? (Read the whole thing before implementing)

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



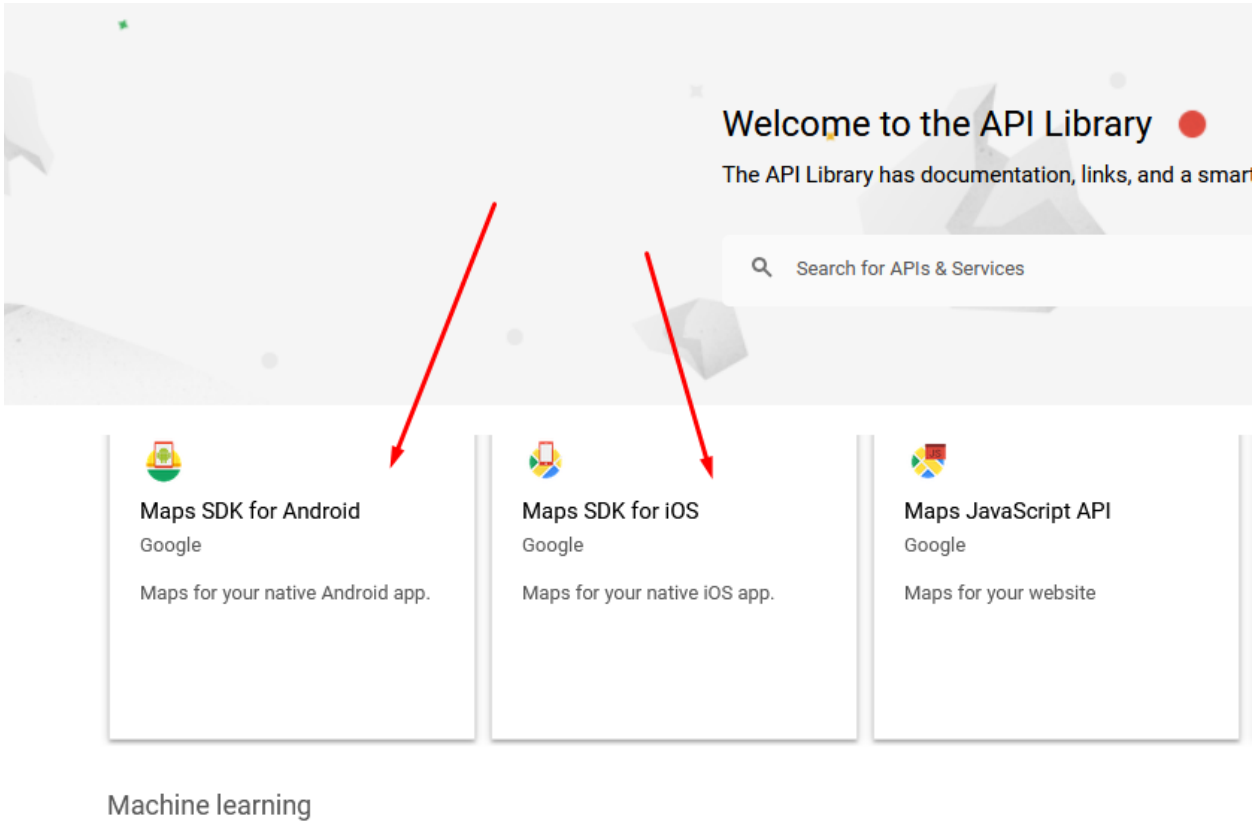
2. In main AndroidManifest.xml put the map api key



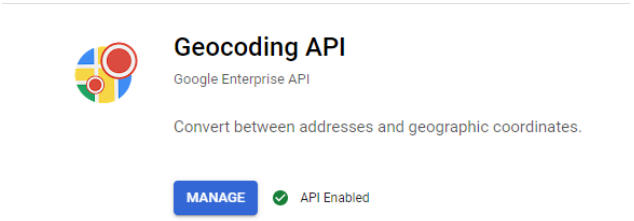
3.For ios follow this

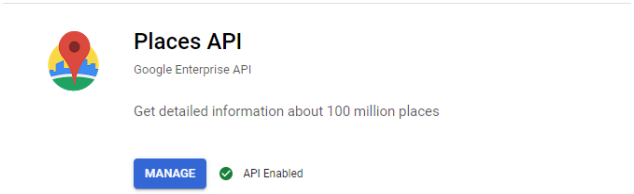
https://blog.logrocket.com/adding-google-maps-to-a-flutter-app/#addinggooglemapstoflutterios

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



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





12. How to configure the default language for mobile apps?

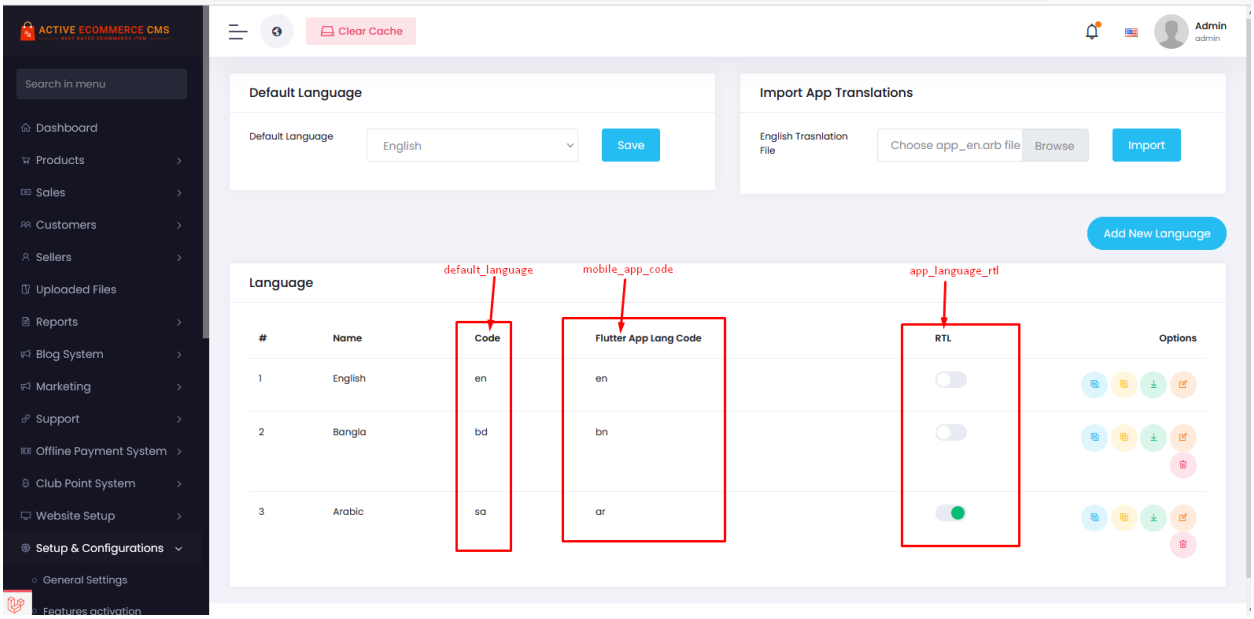
Go to your flutter project->lib->app_config.dart

Change variables value `//Default language config`

```
static String default_language ="en";
static String mobile_app_code ="en";
static bool app_language_rtl =false;
```

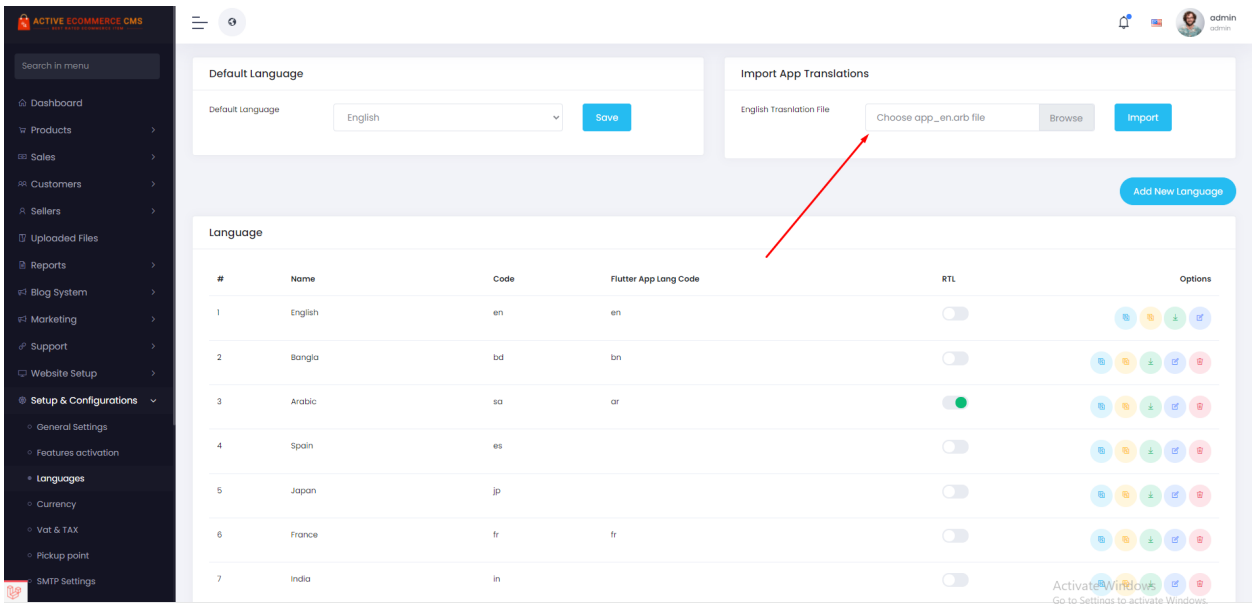
```
import 'package:flutter/material.dart';

var this_year = DateTime.now().year.toString();

class AppConfig {
  static String copyright_text = "@ ActiveItZone " + this_year; //this shows in the splash
  static String app_name = "Active eCommerce"; //this shows in the splash screen
  static String purchase_code = ""; //enter your purchase code for the app from codecanyon
  //static String purchase_code = ""; //enter your purchase code for the app from codecany

  //Default language config
  static String default_language ="en";
  static String mobile_app_code ="en";
  static bool app_language_rtl =false;

  //configure this
  static const bool HTTPS = false;
```

This value you can find in your admin panel. Go to your admin panel->setup & configurations->languages.

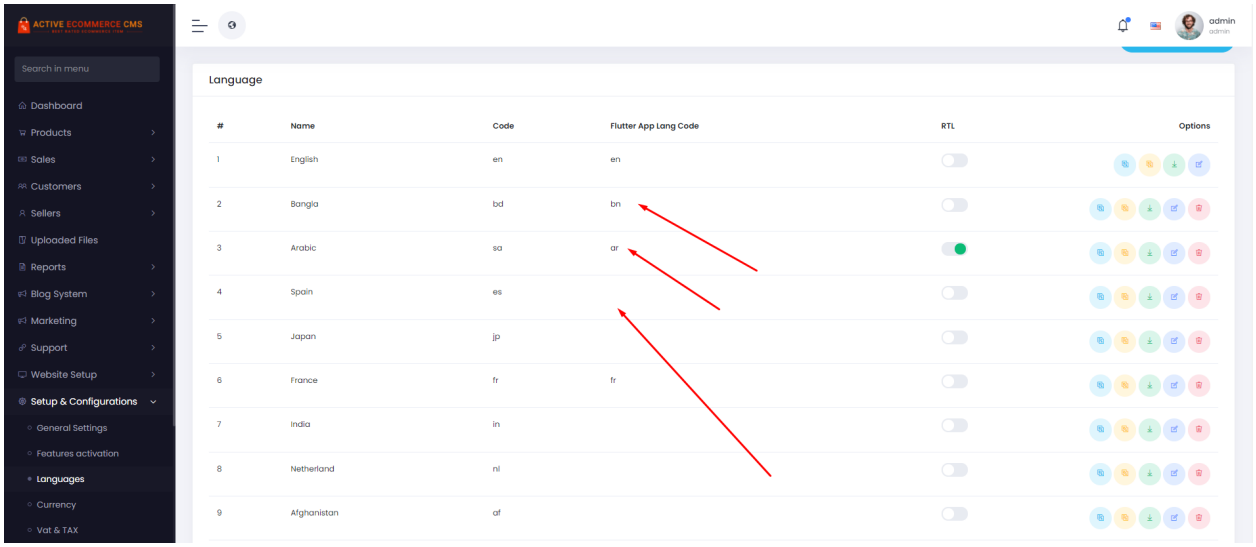


13. How to configure multiple languages for mobile app? (Read the whole thing before implementing)

1. In your **lib/l10n** folder you will see an **app_en.arb** file. This is your main translation + interpretation file. Never delete this.NEVER.
2. If you want another language file you can copy the app_en.arb file and make another language file like app_fr.arb and so on. But we will suggest that you use our translation generator from admin panel.
3. Always make sure your language code is valid. https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes Use iso 639-1 codes. By default flutter localization uses 78 major language codes from here.
4. Upload **app_en.arb** in the admin panel.It will fetch strings from the file and uploads to your database.

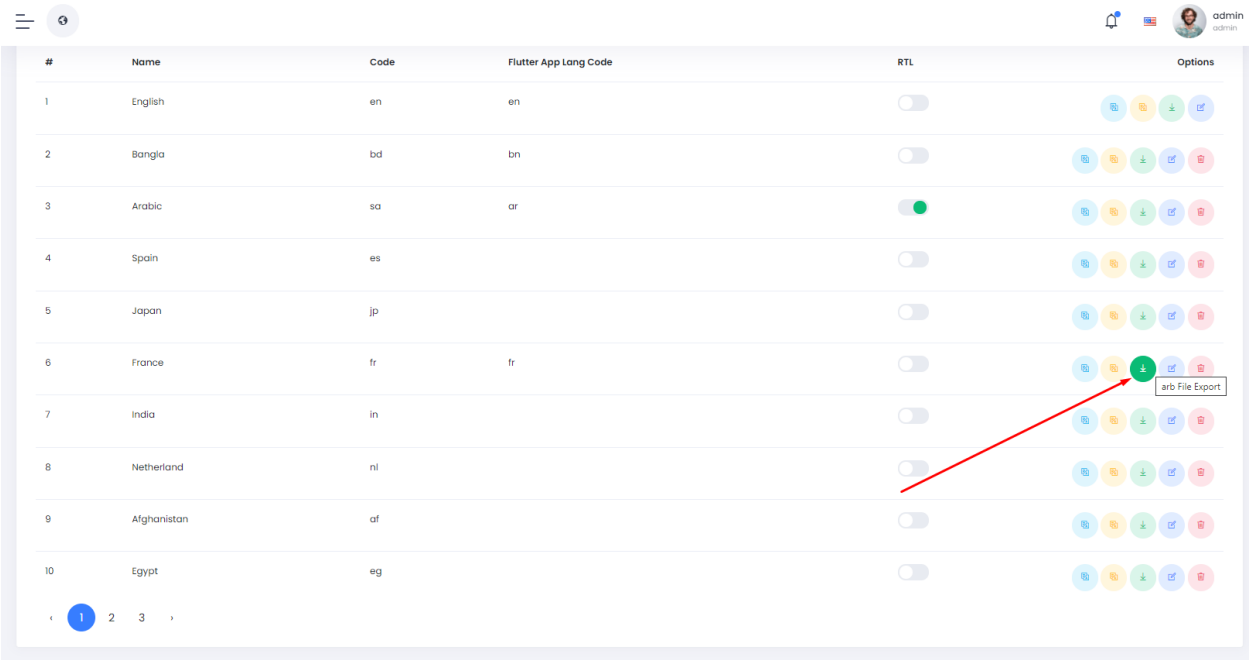


5. Make sure while adding/editing a language , your flutter app language code exists. The code must be in iso 639-1format. Without a valid code , you will not see a translated output in app.



6.Then translate your app strings like you did for your web. You can use google chrome's translation extension and the copy button for a faster output. See, our documentation on translation is provided with the cms. Remember the translations for web and app are kept separate, so even if you did create the translation, for the web , you have to create it for the mobile app too.

7.Once all the strings are converted for a particular language , say for example french, you can download the app_fr.arb file from the panel and put this arb file in your flutter apps **lib/l10n** folder along with your **app_en.arb** file.You can also change the main **app_en.arb** file this way but we encourage you **not to do it** . If you face any error due to app_en.arb file changes , we will not provide you any support.
Make sure the file you pasted in the **lib/l10n** is not empty.If you provide an empty file you will get errors.



8. For the same language , your language code for app and web can be different. This is not an issue. But you have to make sure the code for the app is in 639-1 format.

9.The language list to the app is shown from the backend api, so if you are using a lot of languages , make sure you provide translation for all of them.If you don't , by default the text from app_en.arb will be shown.

13. How to remove cache data.

To enrich user experience we have cached (Mostly for a day) a lot of api responses. If you think your app data is not changing even after your data has been changed from the backend, try clearing cache from the admin panel.There is a big red button on the top navbar in the admin panel to clear cache.