Introduction To MPOS SAAS

POS Inventory Management Web App With Firebase,Subscription Packages, PayPal and Invoice Printing

MPOS SAAS - Flutter POS Inventory Management App with Firebase is as the name says is dominating and powerful Flutter app contains complete app templates, ready to use screens, widgets, illustrations and stunning screens covering POS Inventory web app. MPOS Saas - Flutter POS Inventory Management Web App with Firebase is developed with the highest quality, ease of reusing widgets, fast, and completely user-friendly interface. You can easily use these to make your Flutter POS Inventory Management Web App. MPOS Saas- Flutter POS Inventory Management Web App with Firebase makes the developer job easy to achieve the modern look and feel of the web application. It saves your hustle and time to develop a perfect design UI for modern-day-use web applications. There is a Subscription System for generating revenue using PayPal (More coming soon).

Getting started

This guide will get you started with the setup of the application on your local machine.Now that you have set up your server, it's time to set up the mobile application to consume the data from your website. This guide will help you to connect the application with your server.

> Please make sure that you follow the steps mentioned in this section very carefully.
> If you miss some steps it may lead to unwanted behavior or errors.

Prerequisite:

- Android Studio: 2024.1.1
- Flutter version: 3.22.3 (min)

Install Flutter

Before you can start testing your application, you will need to install Flutter on your machine.

> Please install Flutter version +3 to avoid any dependency issues.

Setting up an Android Emulator or an iOS Simulator which will run your application on your local machine is part of the flutter installation.

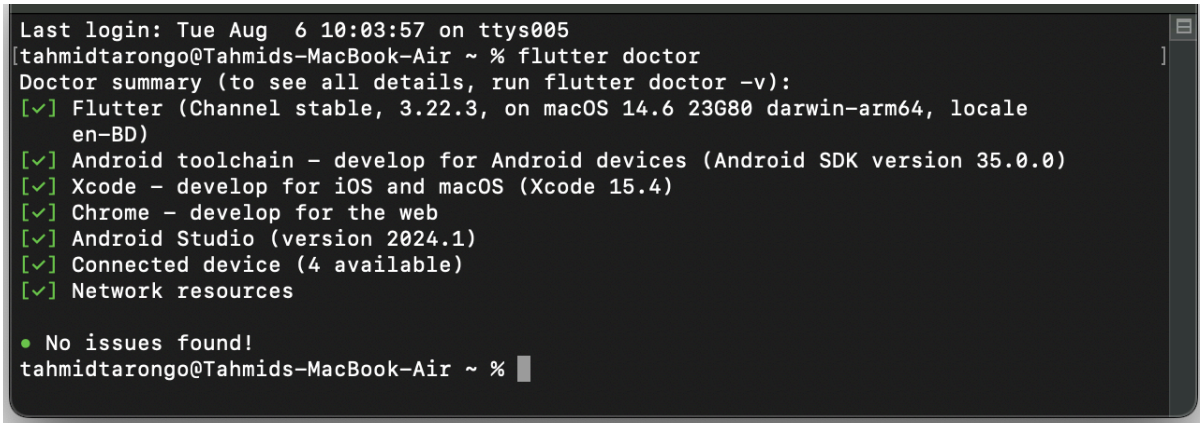
After you've installed flutter then please follow the steps from the following link to setup your dev machine: **https://flutter.dev/docs/get-started/install**

> Please make sure that you follow each and every step that is mentioned in the flutter installation guide properly

Set Up an editor
The recommended editor is Android Studio. Follow the guide from the link to setup an editor:
https://flutter.dev/docs/get-started/editor

Flutter Doctor
After you've followed through all the steps from the Flutter installation guide above, you need to check your installation for any issues that might cause some problem. You can do this by following the command below:

```
flutter doctor
```



Note that the summary above has all tick marks. If you see a cross (X) in-front of an option then follow the steps mentioned below that option to complete your installation. If you see [!] No Connected device (0 available) do not worry, you will open an iOS Simulator or an Android Emulator later in the setup process which will resolve this.

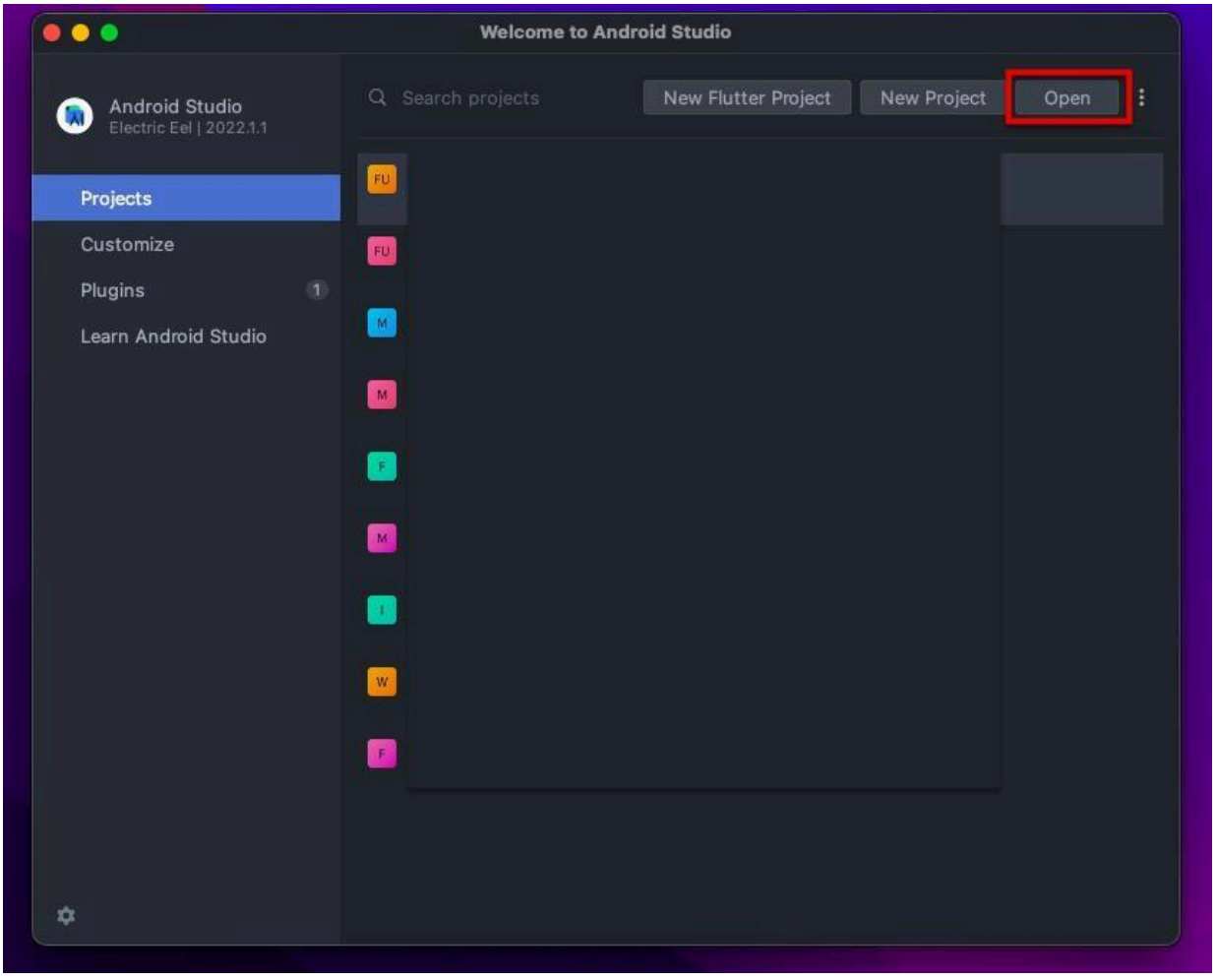
Modify App

Once you installed Android Studio on your machine, you can start changing the required files for App from android studio by following this guide.

Open app in Android Studio

To modify the required settings for your POS Saas applications, you need to open it in Android Studio. Follow the steps below to open the application in android studio.

1. Open Android Studio on your machine.
2. From the Welcome to Android Studio screen, click on Open.

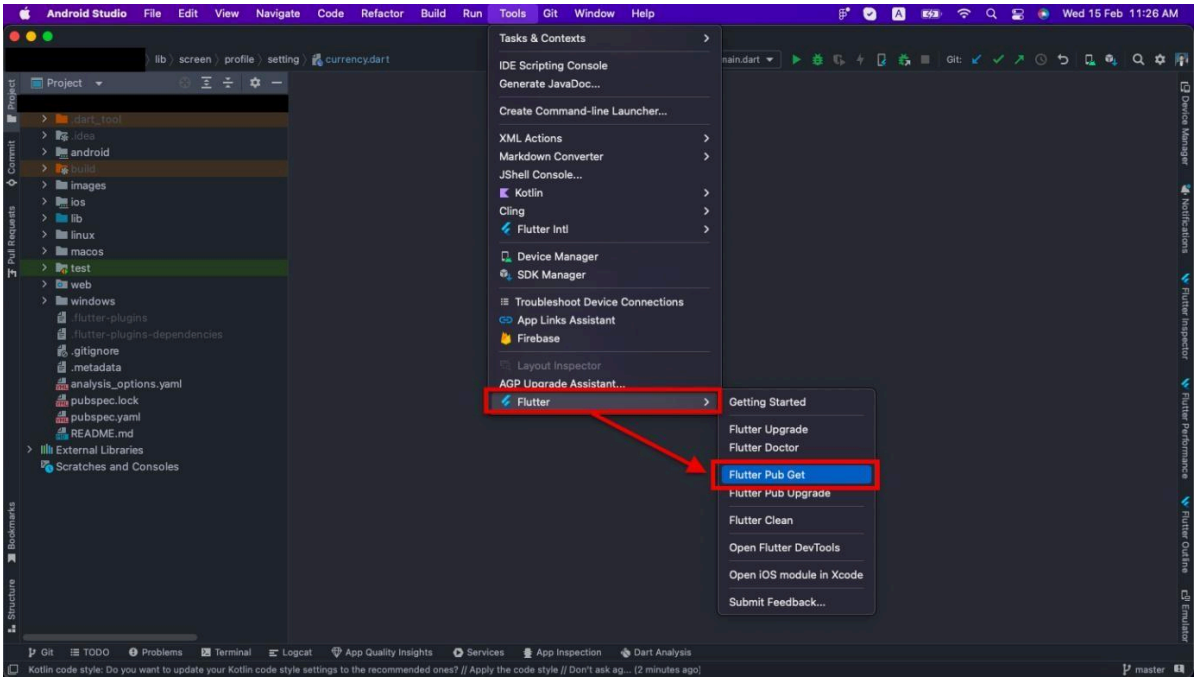


3. Choose the Pos Saas - App / Pos Saas Web/ Pos Saas super admin directory and open it. Note that you have to set the web first. It is located at download or the folder you selected when it was downloaded from the marketplace. This will open the application source code in android studio.

Now you are ready to make the necessary changes in the application files by following the guide below.

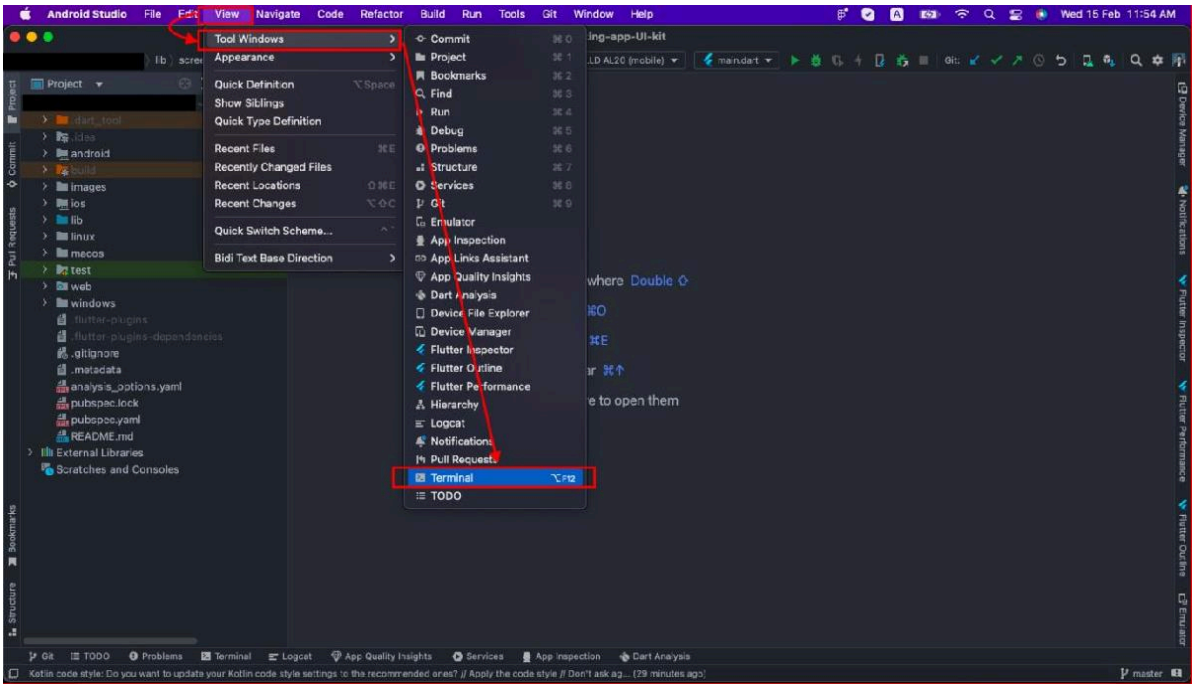
Get Flutter Packages

Now you have to open tool option from the topbar and from the opening menu you have to choose Flutter and then Flutter Pub Get & press or select that.
Also if required, you have to give the sdk path, & allow for the configuration.



You can do it from your terminal also. For that you have to open the terminal from the topbar.

Select:
View → Tool Windows → Terminal as shown in the Next figure.



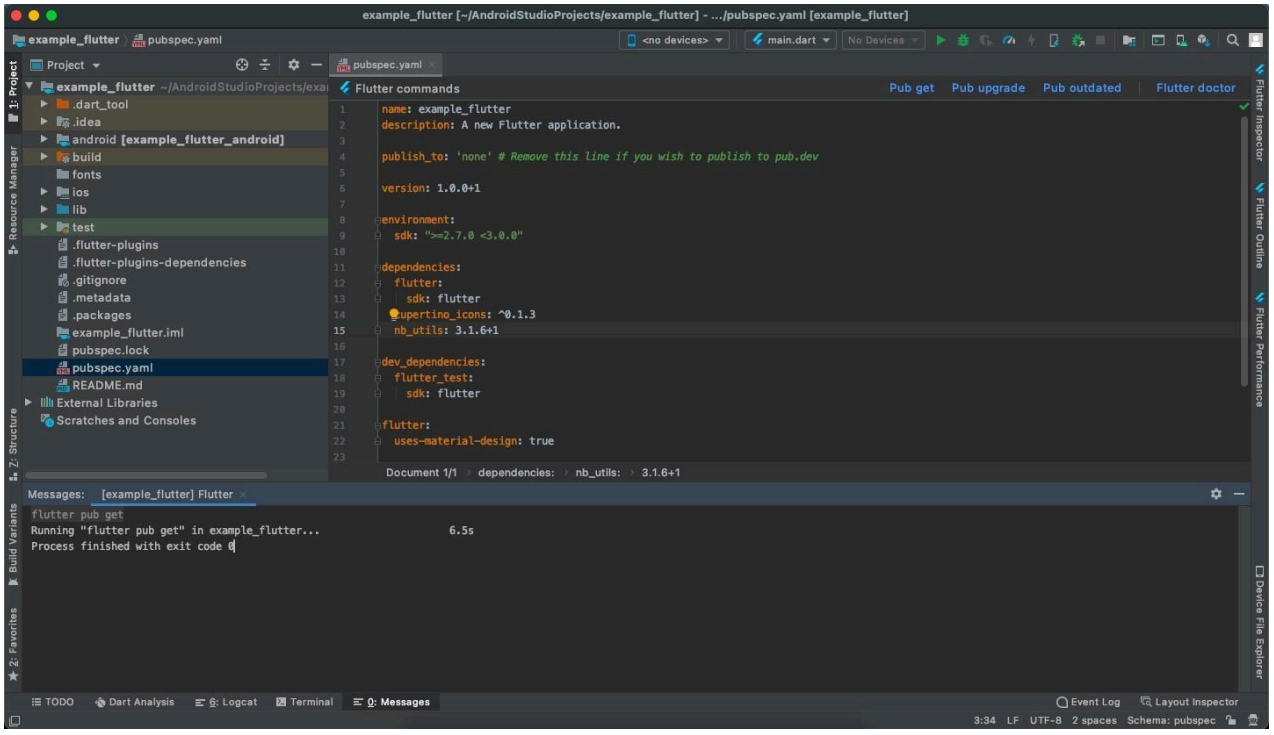
Now type flutter **pub get** on terminal and press enter.

This will get the dependencies for the application.

Make sure that you are connected to the internet while performing this task

Build and Run App

All below steps must be followed to build and run applications.



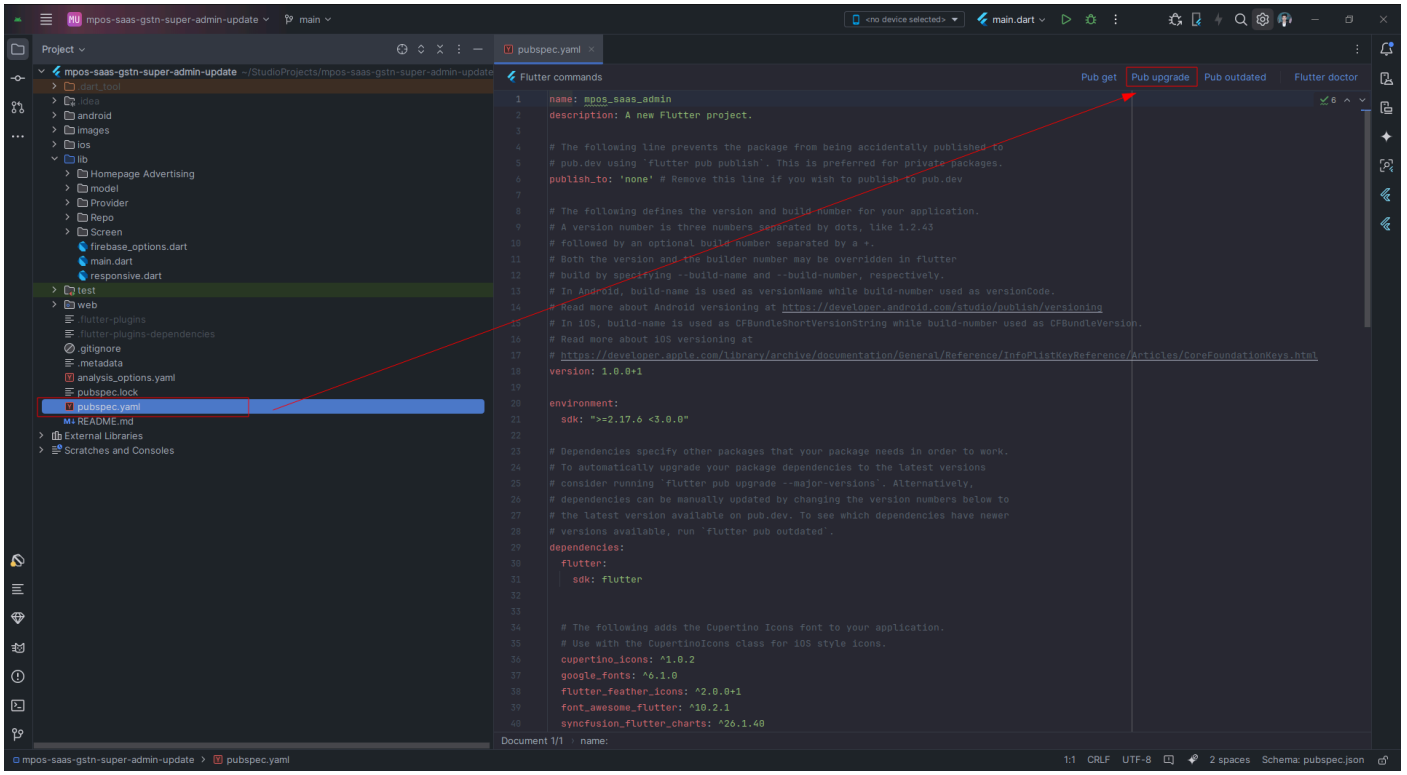
1. Locate the main Android Studio toolbar.
2. In the target selector (<no device selected>), select an Android device for running the app. Or if you want to

run the web app select the available browser from the dropdown menu. If none are listed as available, select Tools > Android > AVD Manager and create one there. For details, see Managing AVDs.

3. Click the run icon in the toolbar, or invoke the menu item Run > Run.

After the app build completes, you'll see the app on your device.

Project Version Update



To update your project version, go to the **pubspec.yaml** file. And click on the **Pub upgrade** button at the top.
It will automatically update all the packages version.Make sure your pc is connected with the internet.

To update your project version and dependencies in a Flutter project, follow these steps:

1. Open the **pubspec.yaml** File:
    - Locate the pubspec.yaml file in the root directory of your Flutter project.
    - Open it in your preferred text editor or IDE.
2. Update the Project Version:
    - Find the version field at the top of the pubspec.yaml file.
    - Change the version number to your desired version. For example: version: 1.1.0+2
    - The version format is major.minor.patch+build.

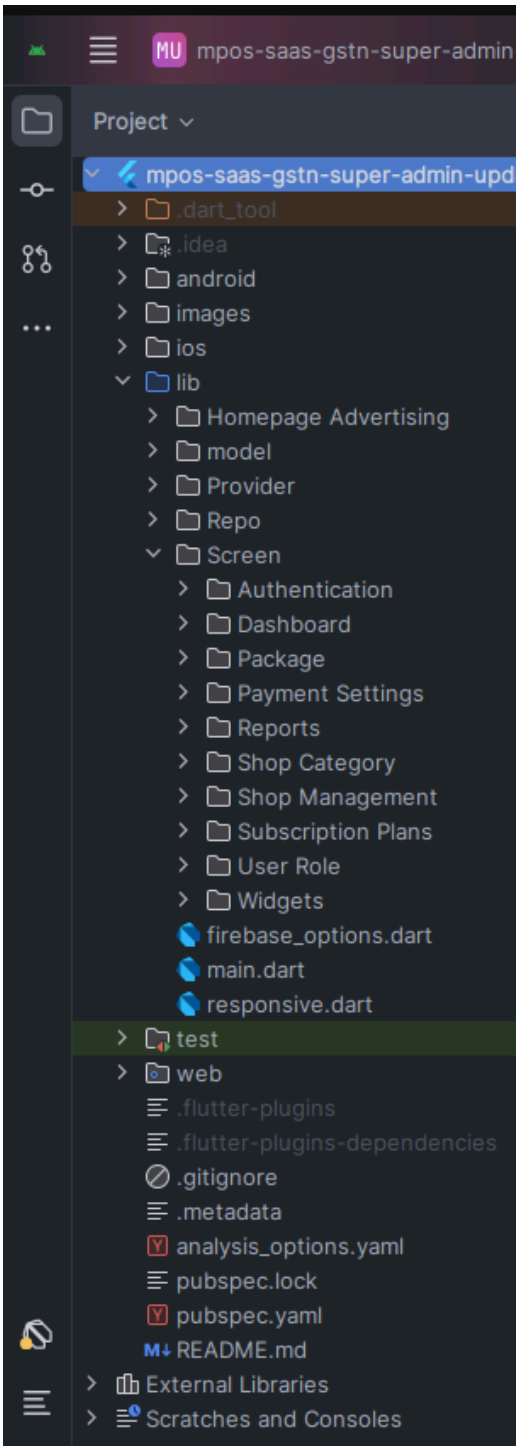
Run pub upgrade:

- Ensure your computer is connected to the internet.
- In your IDE, locate and click the Pub upgrade button. This button is typically found at the top of the editor or in the pubspec.yaml tab.
- Alternatively, you can run the following command in your terminal from the project directory:

flutter pub upgrade
This command updates all packages to the latest versions allowed by the constraints in **pubspec.yaml.**

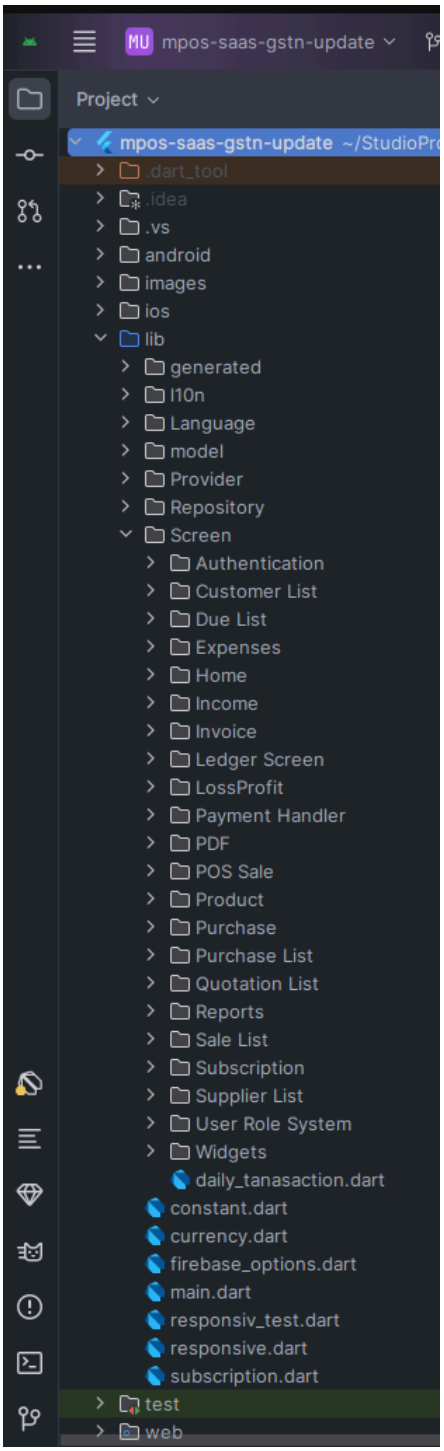
Project Structure:

MPOS SAAS Superadmin



- **images**: Apps/widgets/snippets images as well as appIcon are stored here.
- **lib**: Applications main files and directories are located here.
    - **Screen**: Full app's UI codes are stored here.
    - **Widgets**: All the reusable widgets are stored here
    - **firebase_option.dart:** all the firebase credentials are stored here.

MPOS SAAS Web

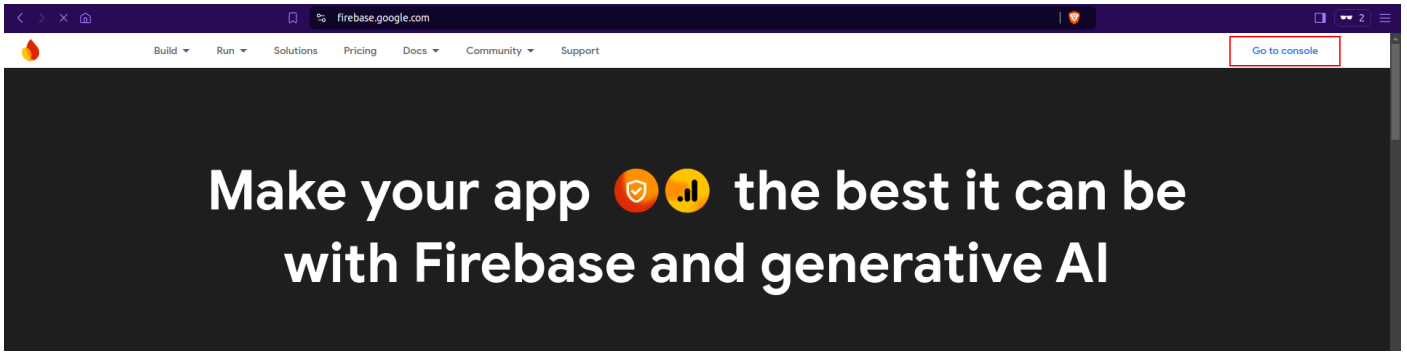


- **images**: Apps/widgets/snippets images as well as appIcon are stored here.
- **lib**: Applications main files and directories are located here.
    - **Screen**: Full app's UI codes are stored here.
    - **Widgets**: All the reusable widgets are stored here
    - **firebase_option.dart:** all the firebase credentials are stored here.

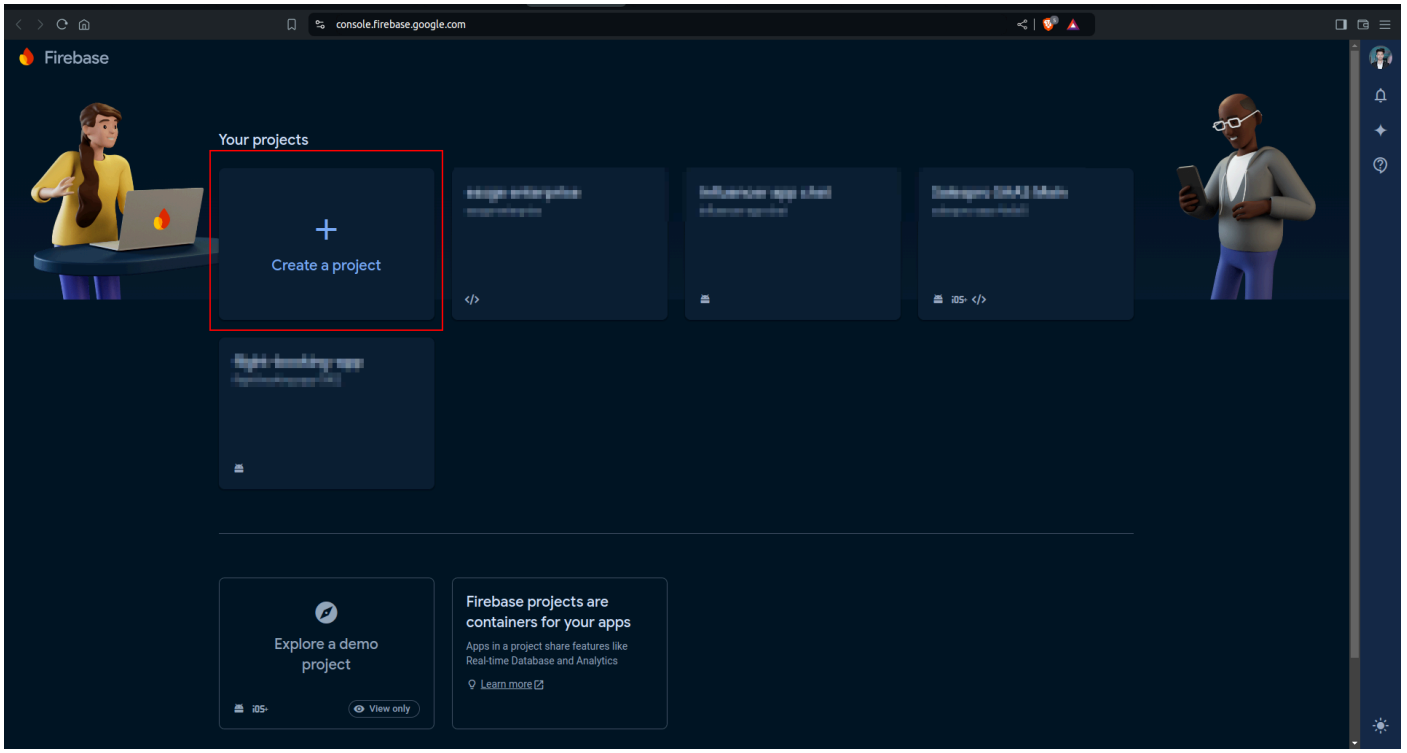
Firebase Setup

Step 1

First of all you have to go to https: console.firebase.google.com/
and click on Go to the console which is on the right side of the top bar menu.(if not signed in)

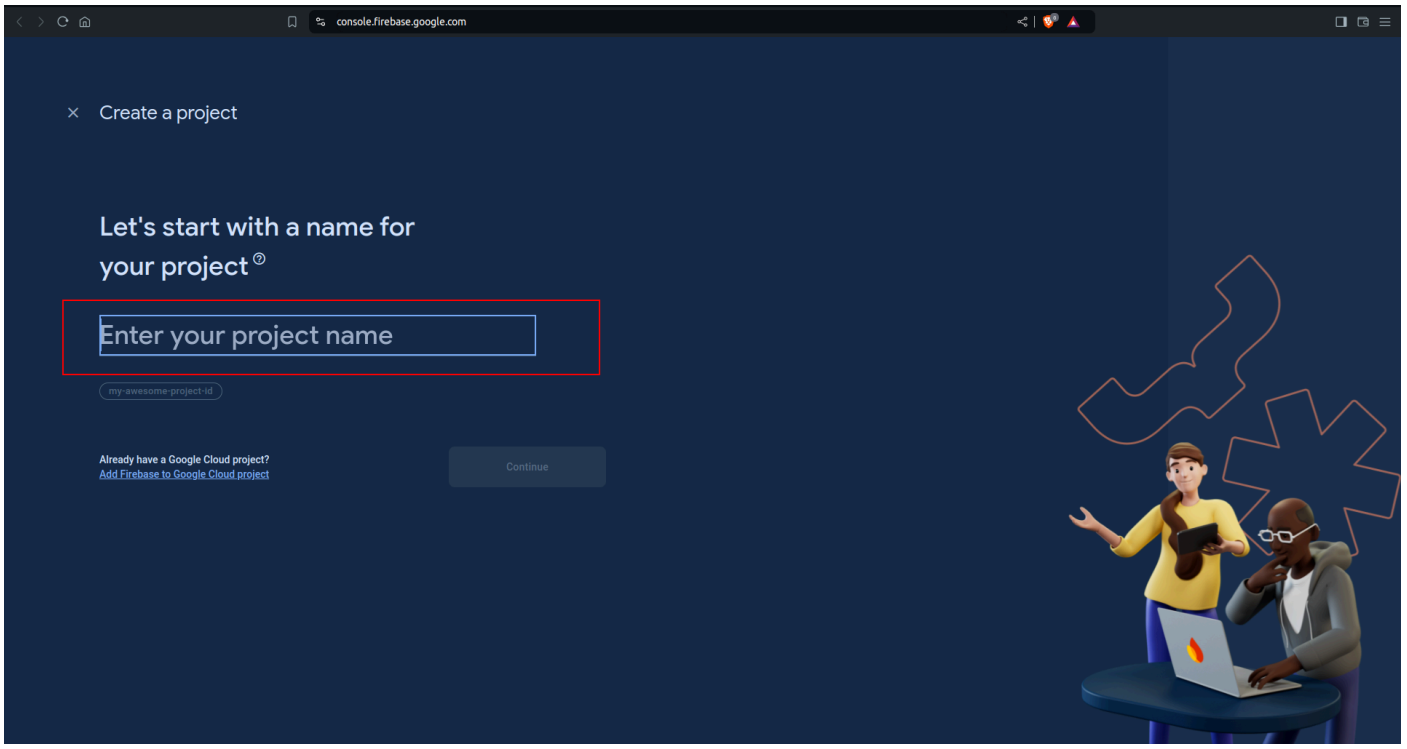


After tapping on the Go to console a new page will be open.

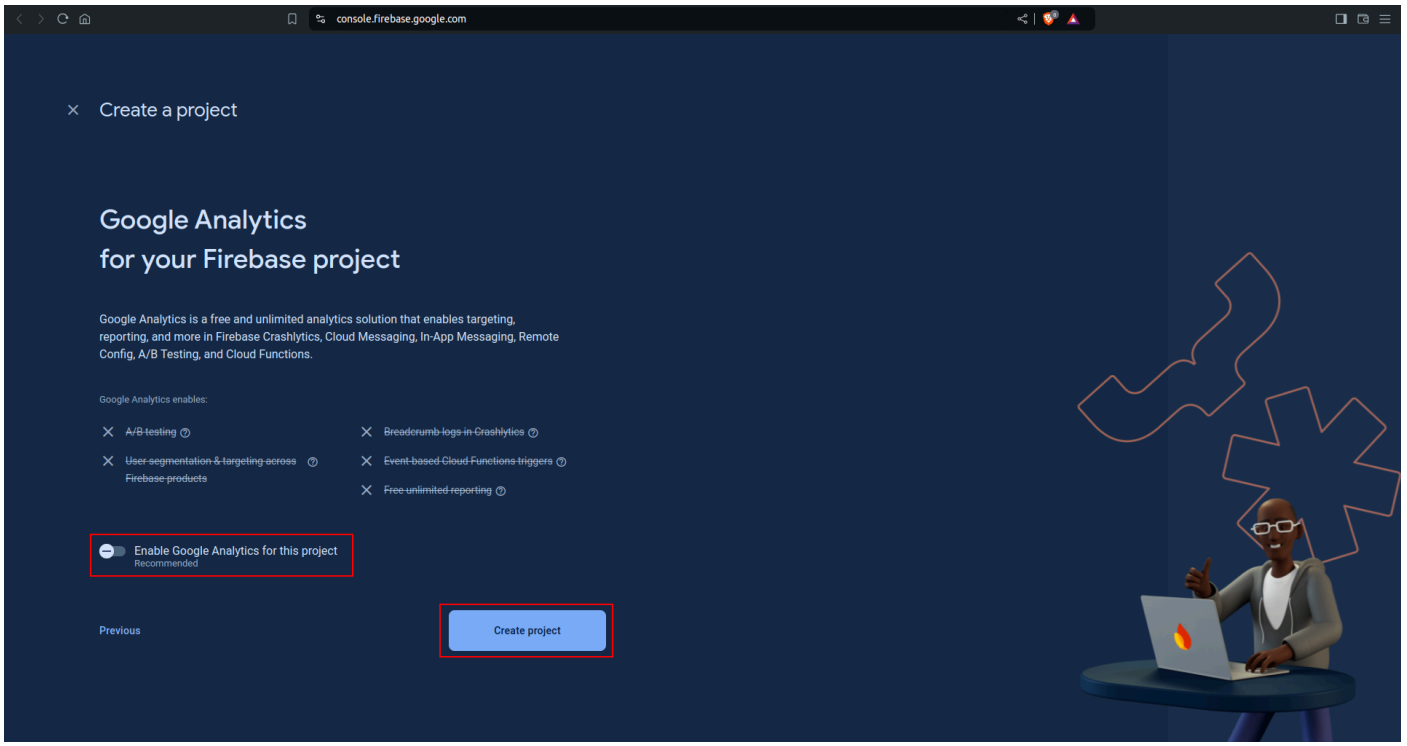


Now you have to create a new project which you want to connect with the app. Click on **Create a project** button
to create a new one.

A screen will open like this.

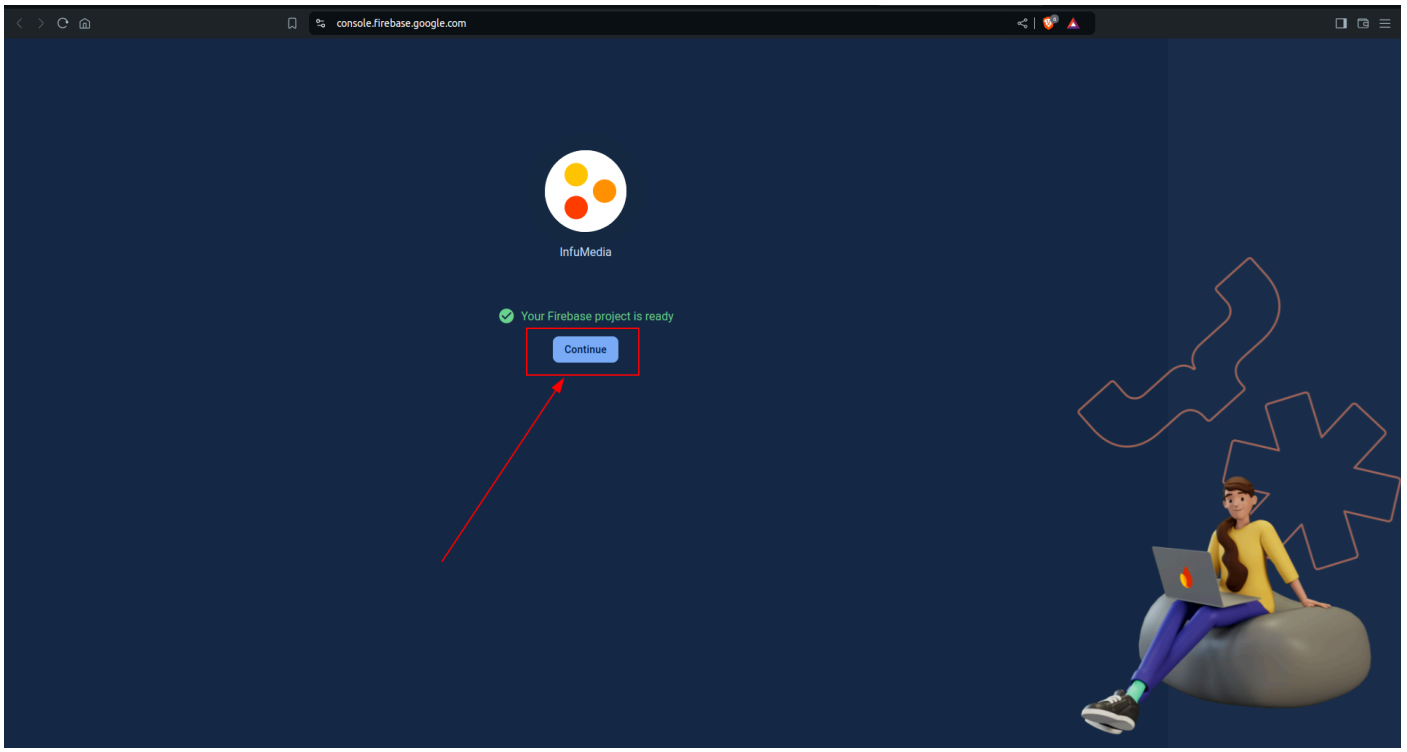


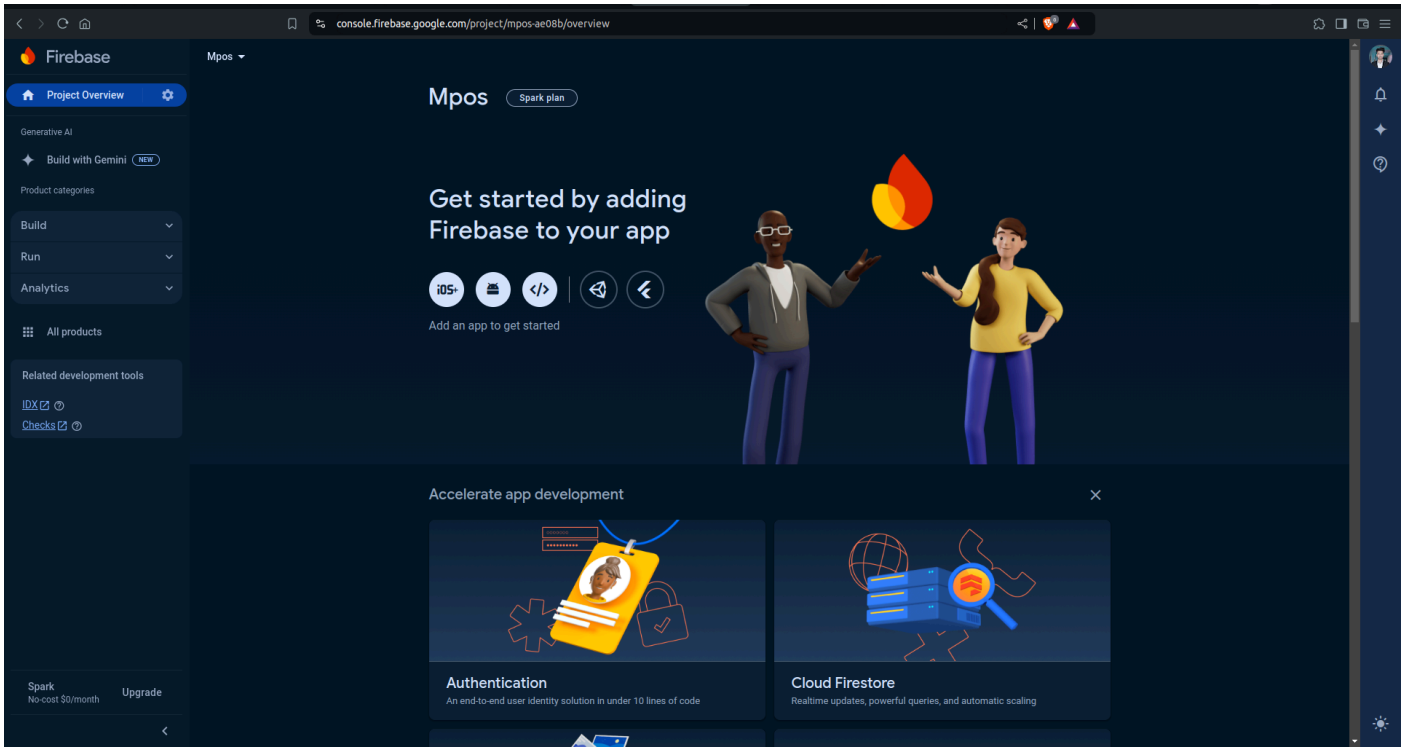
Here you have to enter a name for your project. then press continue to go to the next step.



Here if you want you can enable or disable the google analytics for your project. We recommended you to disable it for now and press **Create project**.

Here a new project is created for you. Press continue to view the project.

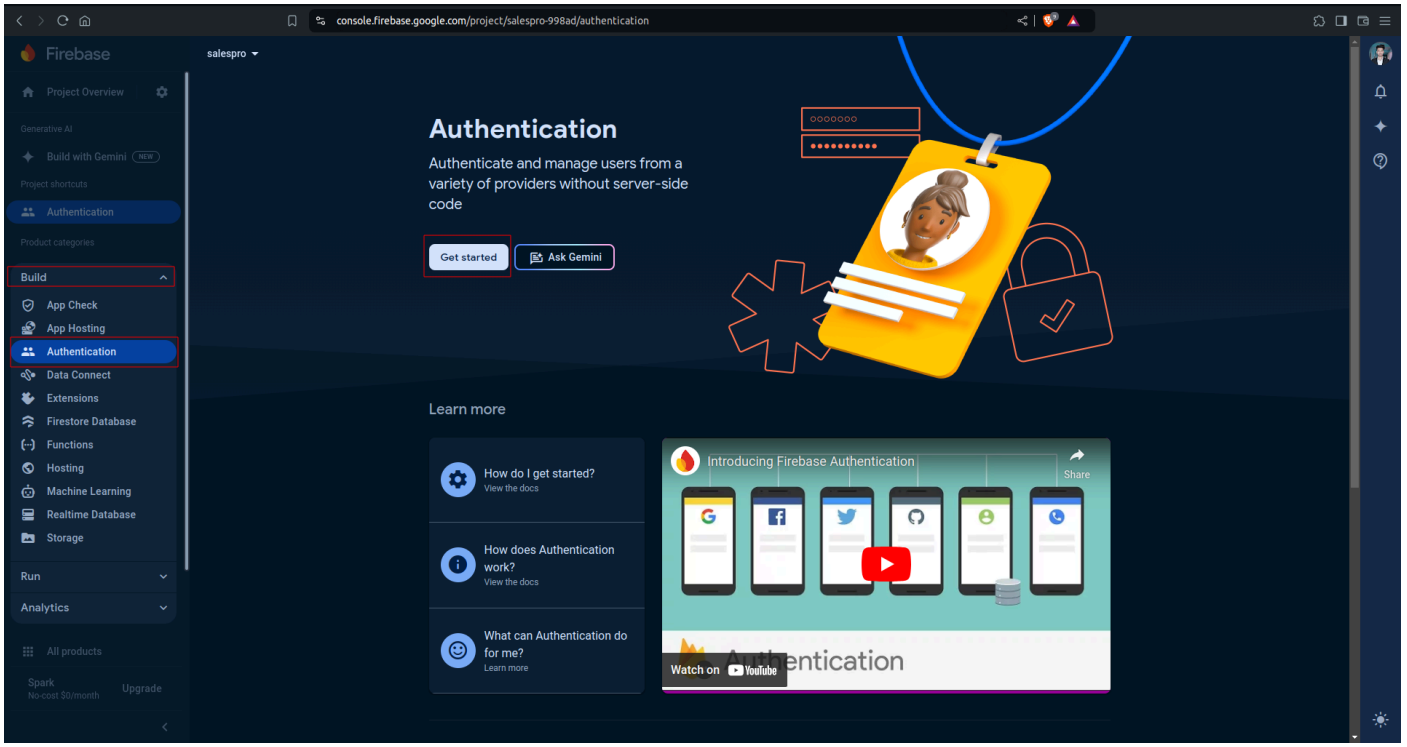




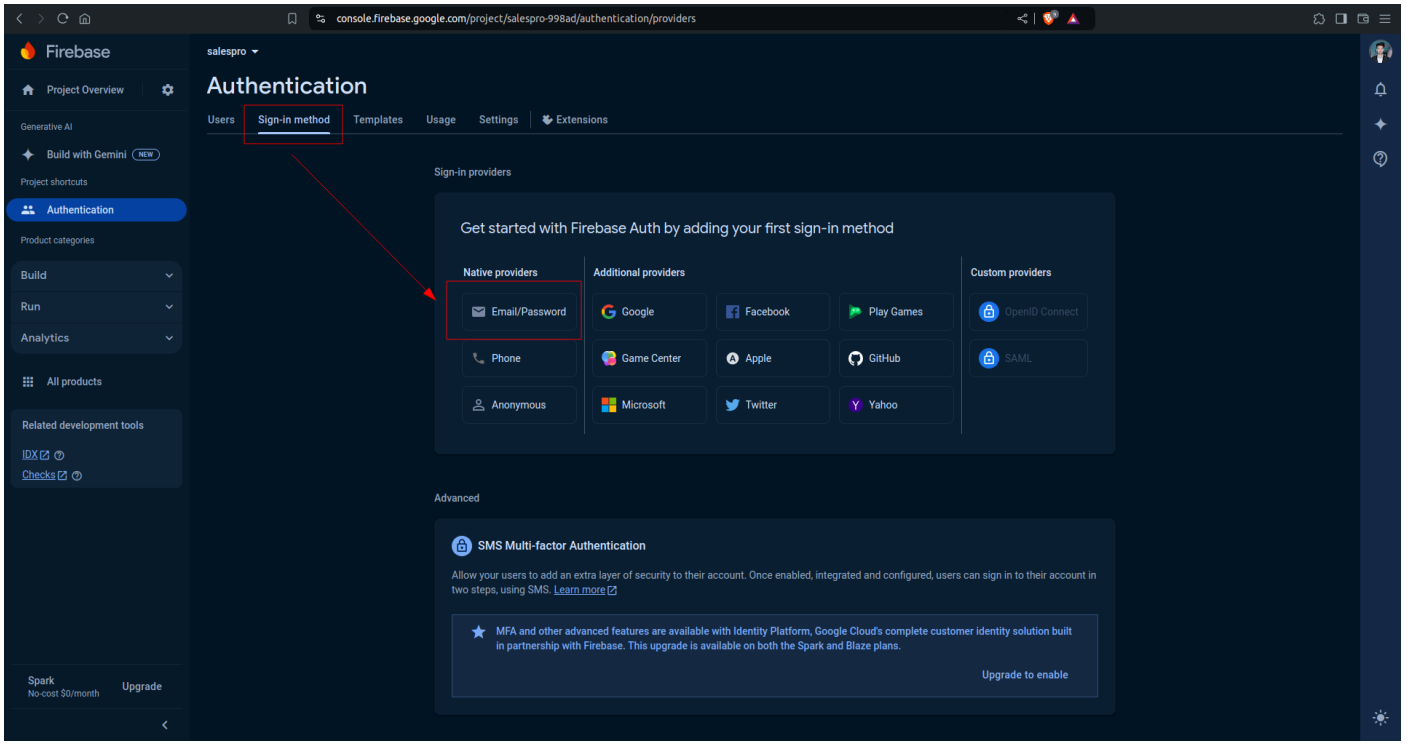
Step 2

> Note that It is required to complete all of the steps one by one before generate the credentials for the app.

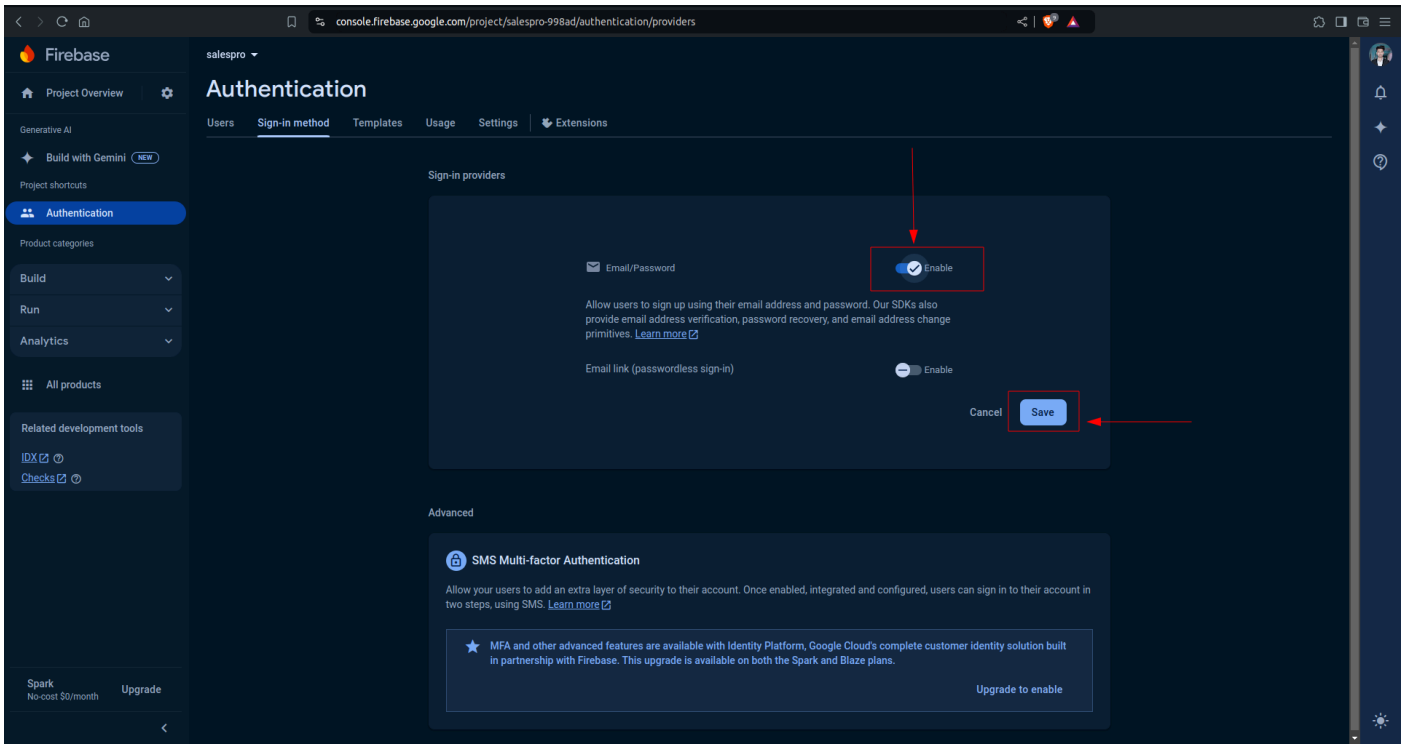
Now from the **sidebar** click on the **Build** section, then select the **Authentication** from the list inside the **build** section. This will open a screen like this.



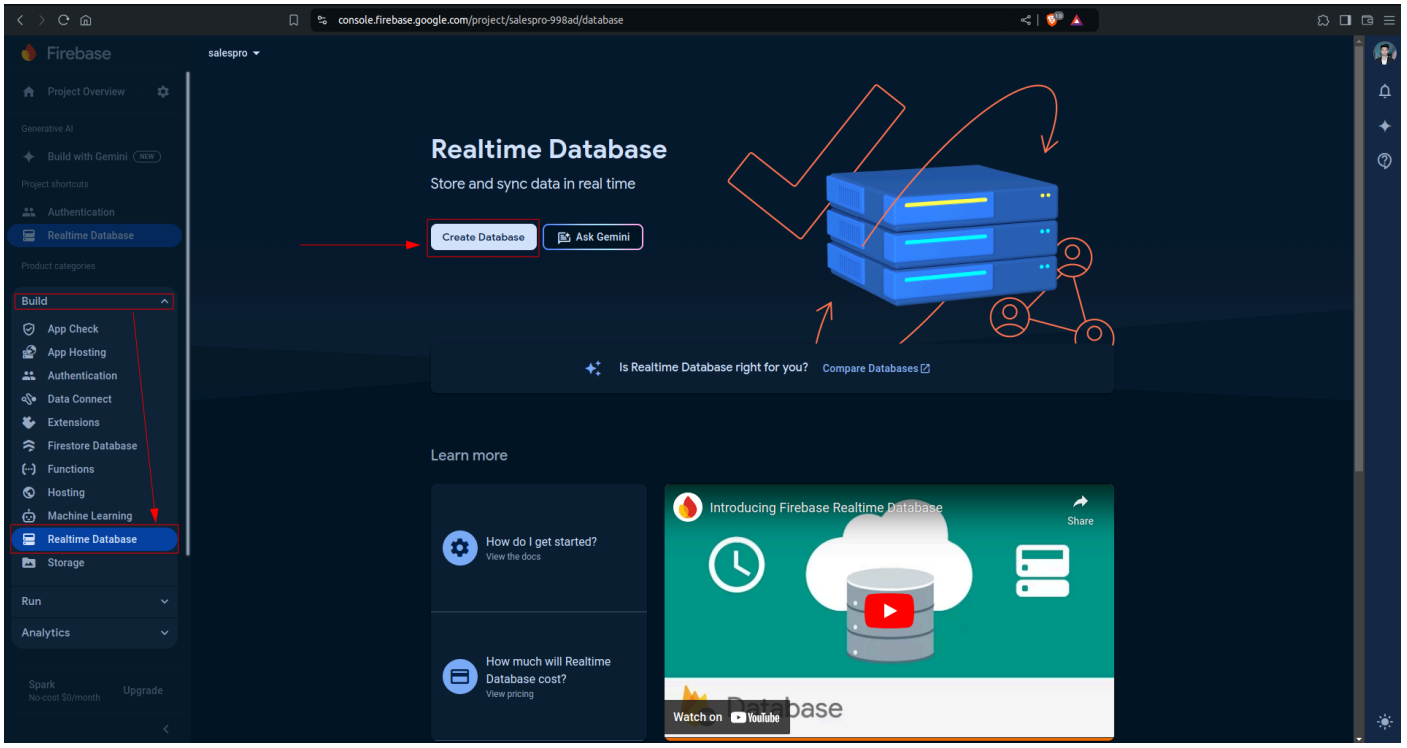
Now press on the **Sign-in method** and select **Email/Password**.



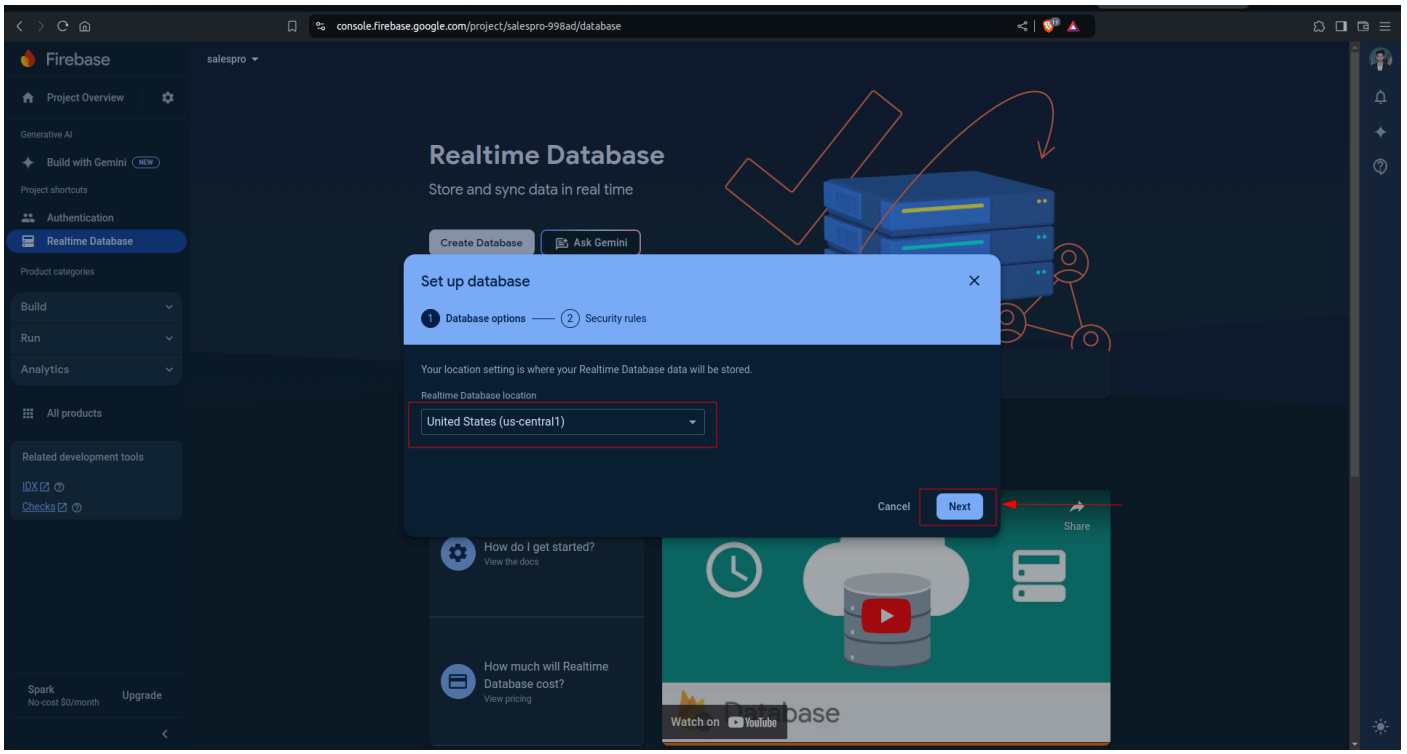
And here enable the **switch** icon and hit the **Save** button.



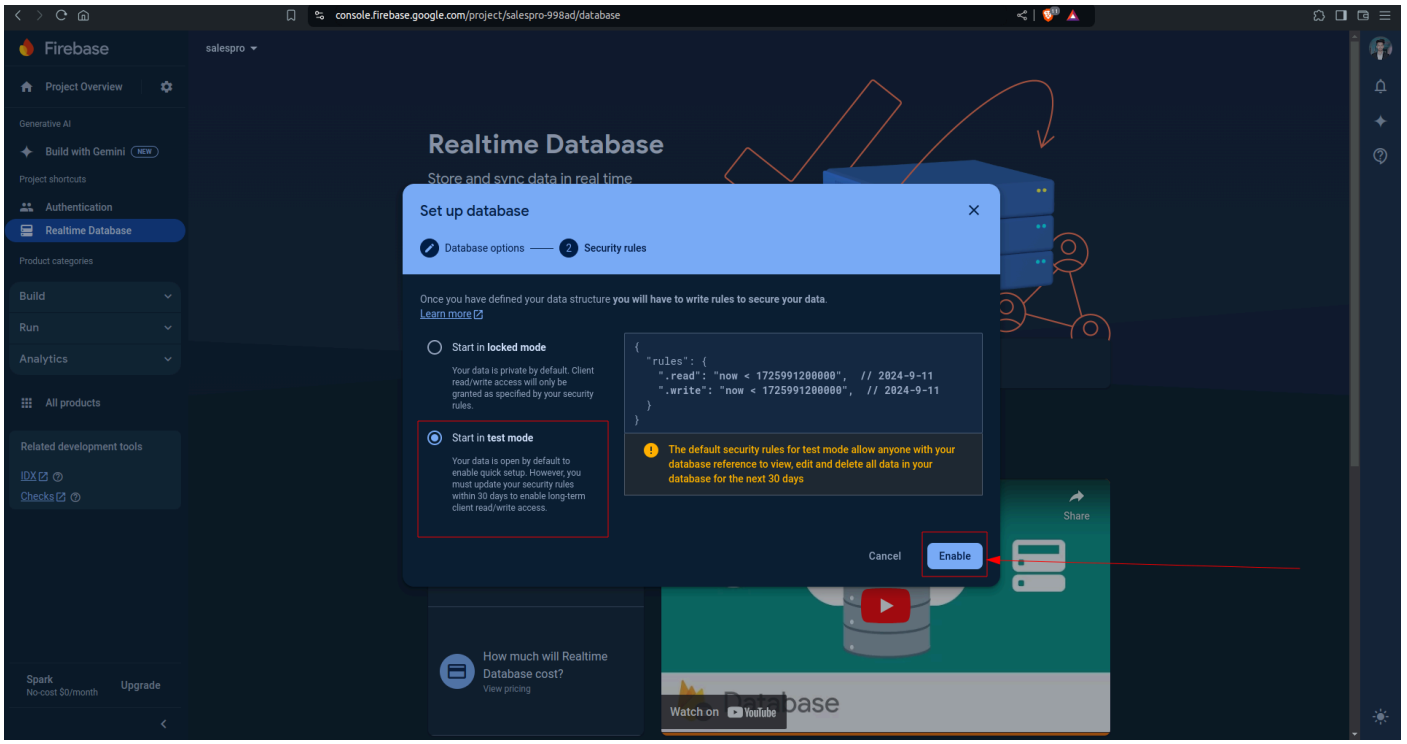
Now again from the **sidebar** click on the **Realtime Databse** section, then press the **Create Database** button**.**



Here a popup will arise, and **select any location** from the dropdown and press **Next** button.

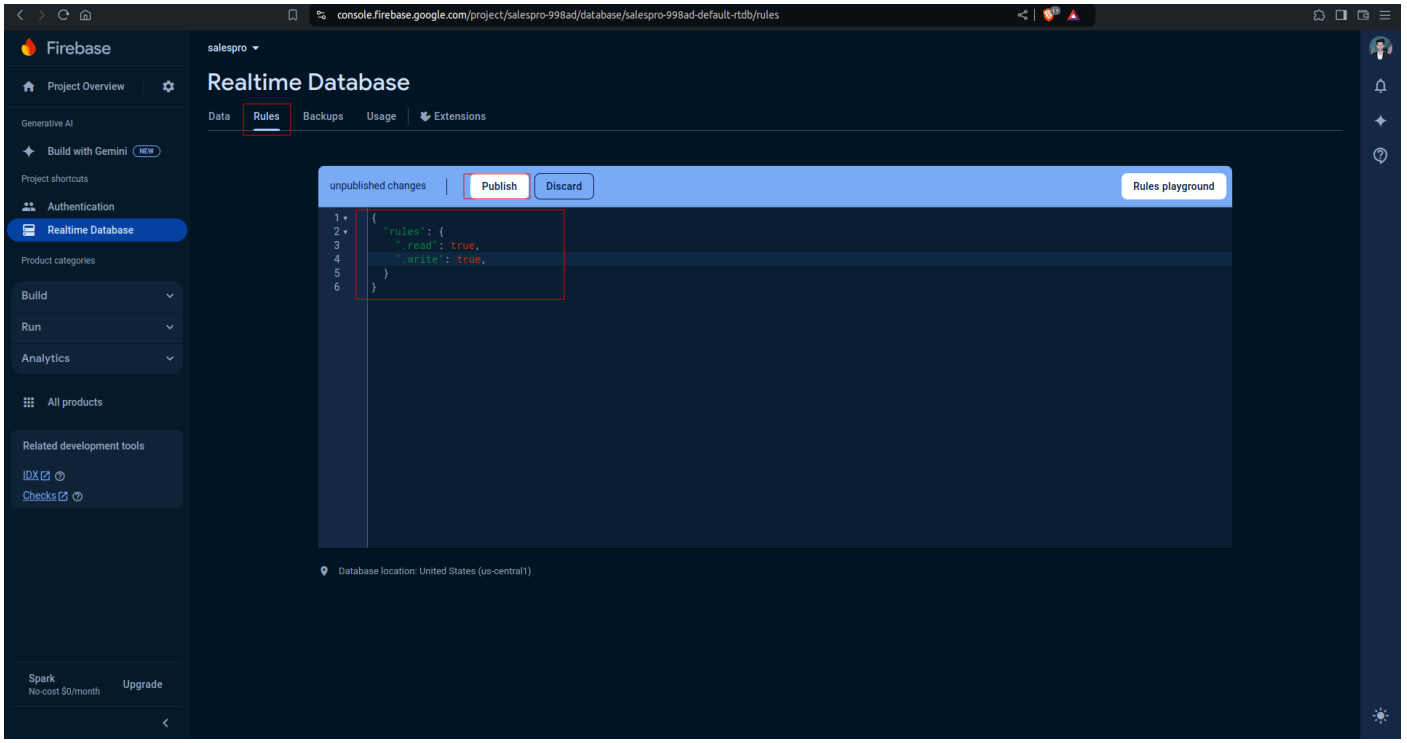


And select **test mode** and press **Enable**.



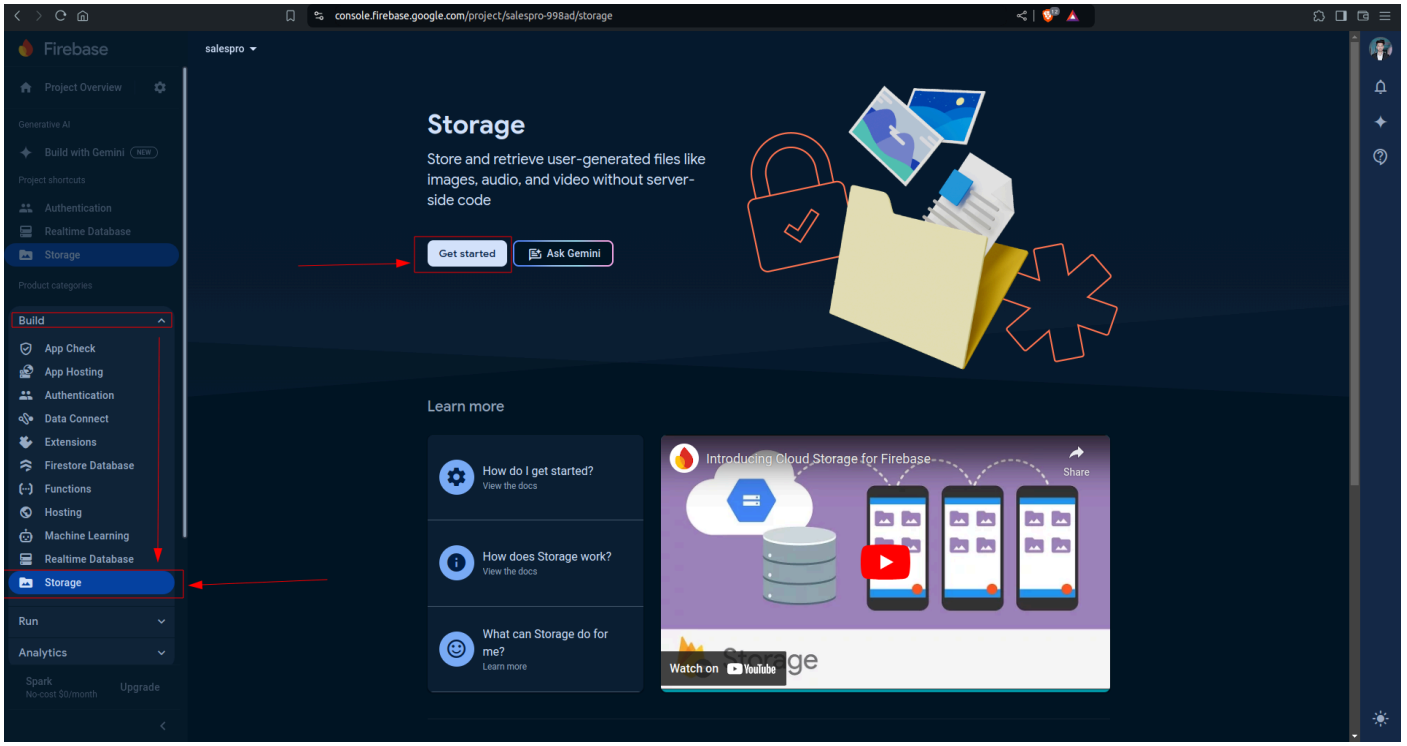
Now come to the **Rules** tab of that screen and make the **read/write rules true** like the image.
Or remove all the code and just paste this

```
{
  "rules": {
    ".read": true,
    ".write": true,
  }
}
```

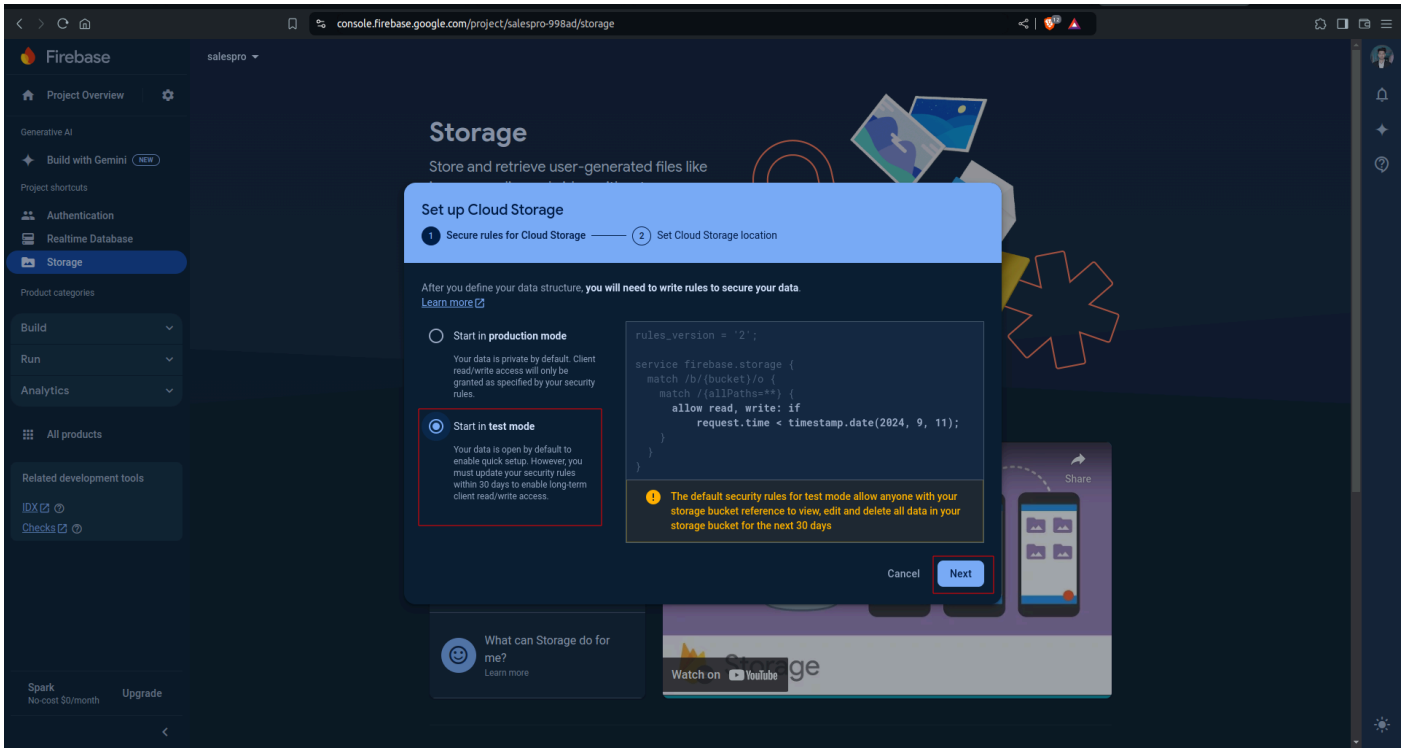


Make sure you copy and paste the same code as it is. And then press the **Publish** button.

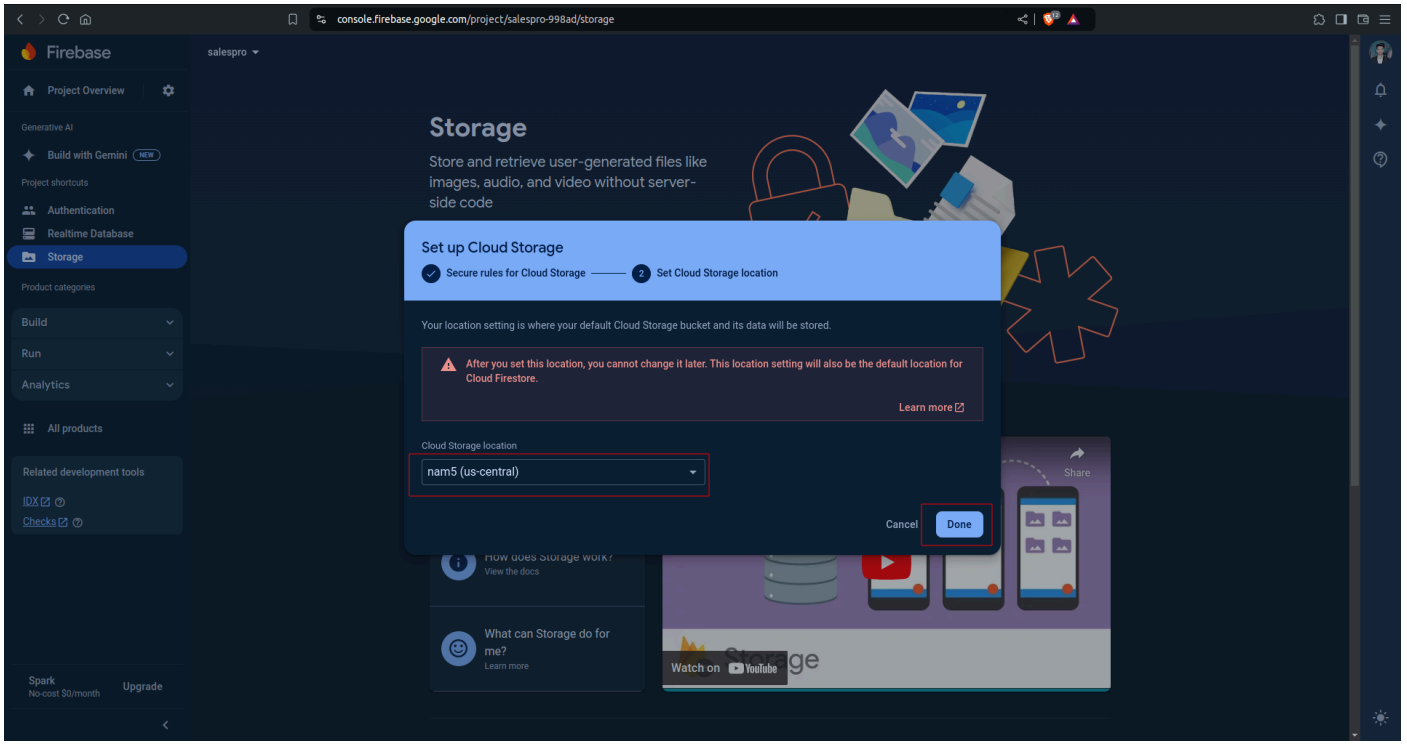
Now again from the **sidebar** click on the **Storage** section, then press the **Get Started** button.



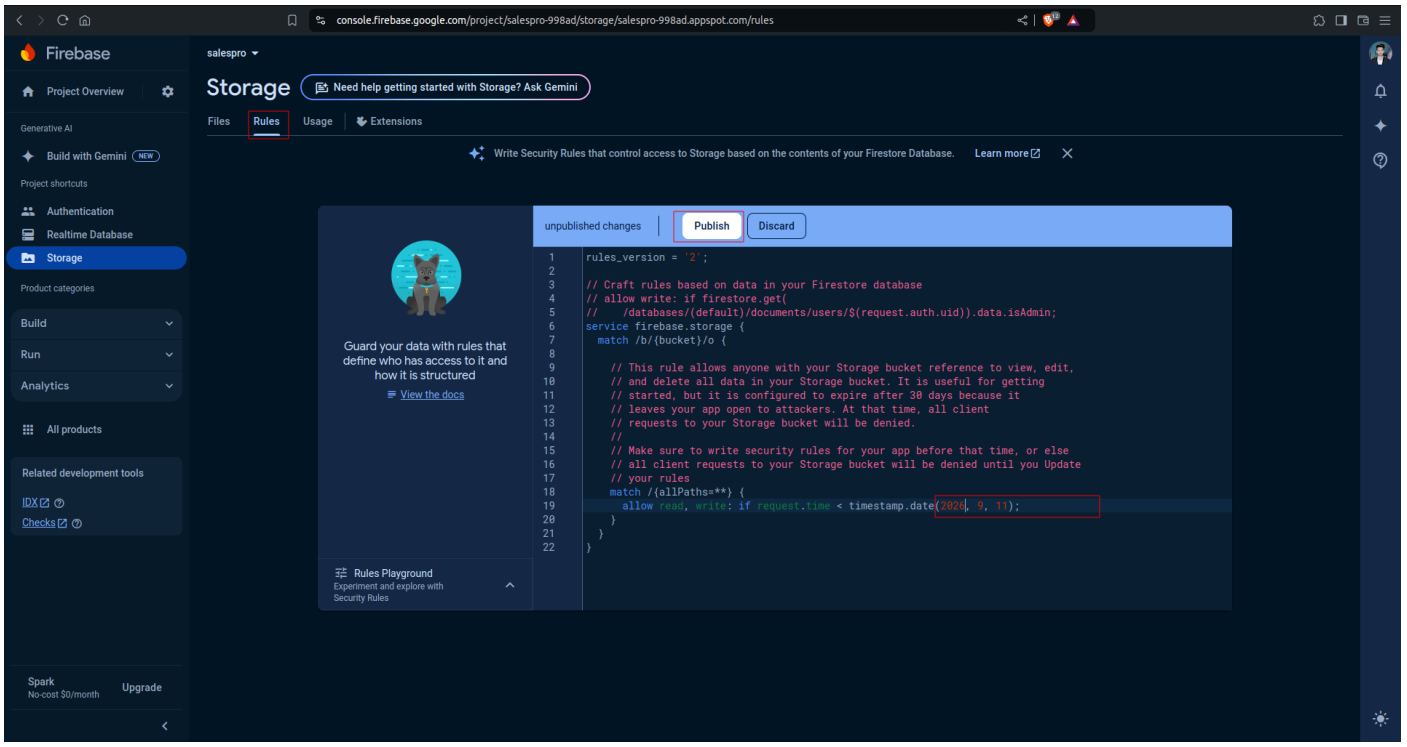
Here a popup will arise, and **Start in test mode** and press **Next** button.



Here **select any location** from the dropdown and press the **Done** button.



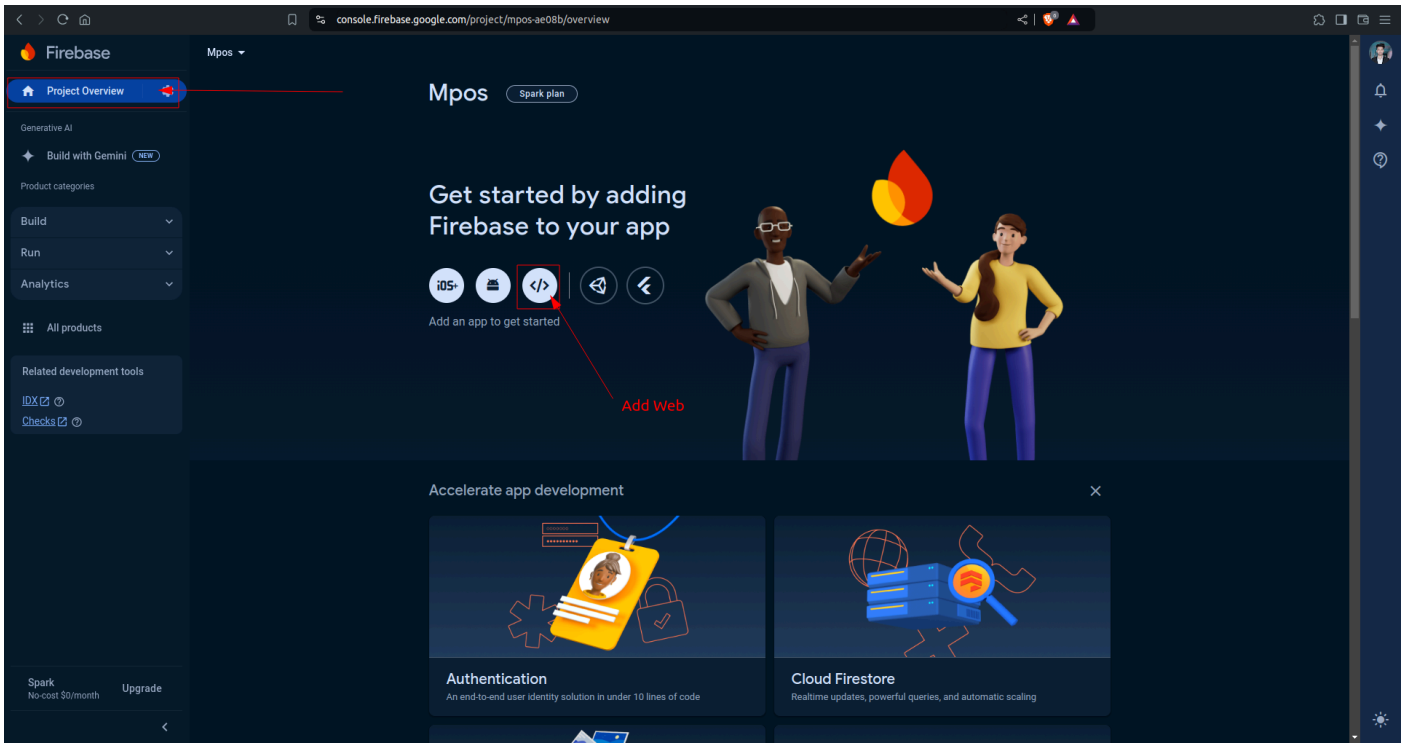
Now come to the **Rules** tab on that screen (Storage).



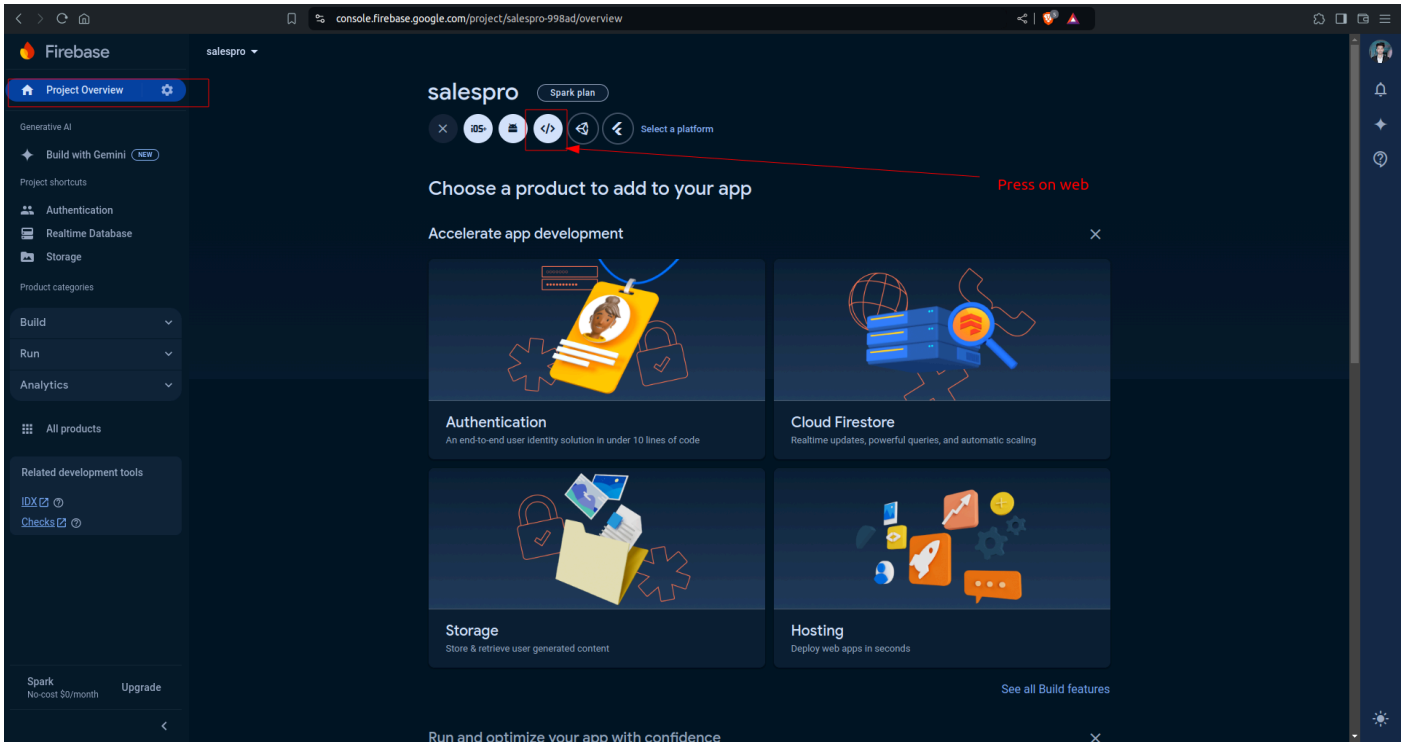
Here increase the time stamp as you want and press the publish button.

Add Superadmin to Firebase

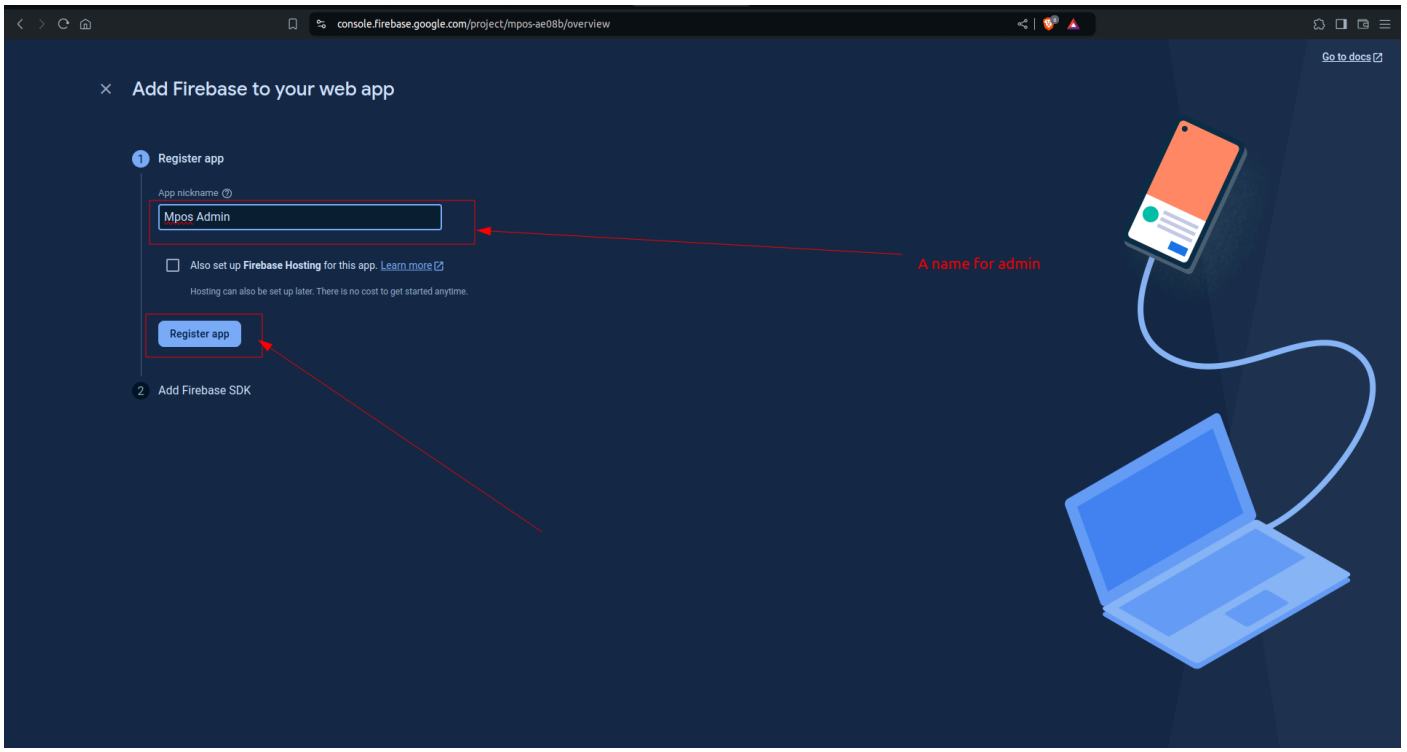
Go back to the **Project overview** screen and here you have to add the web app .



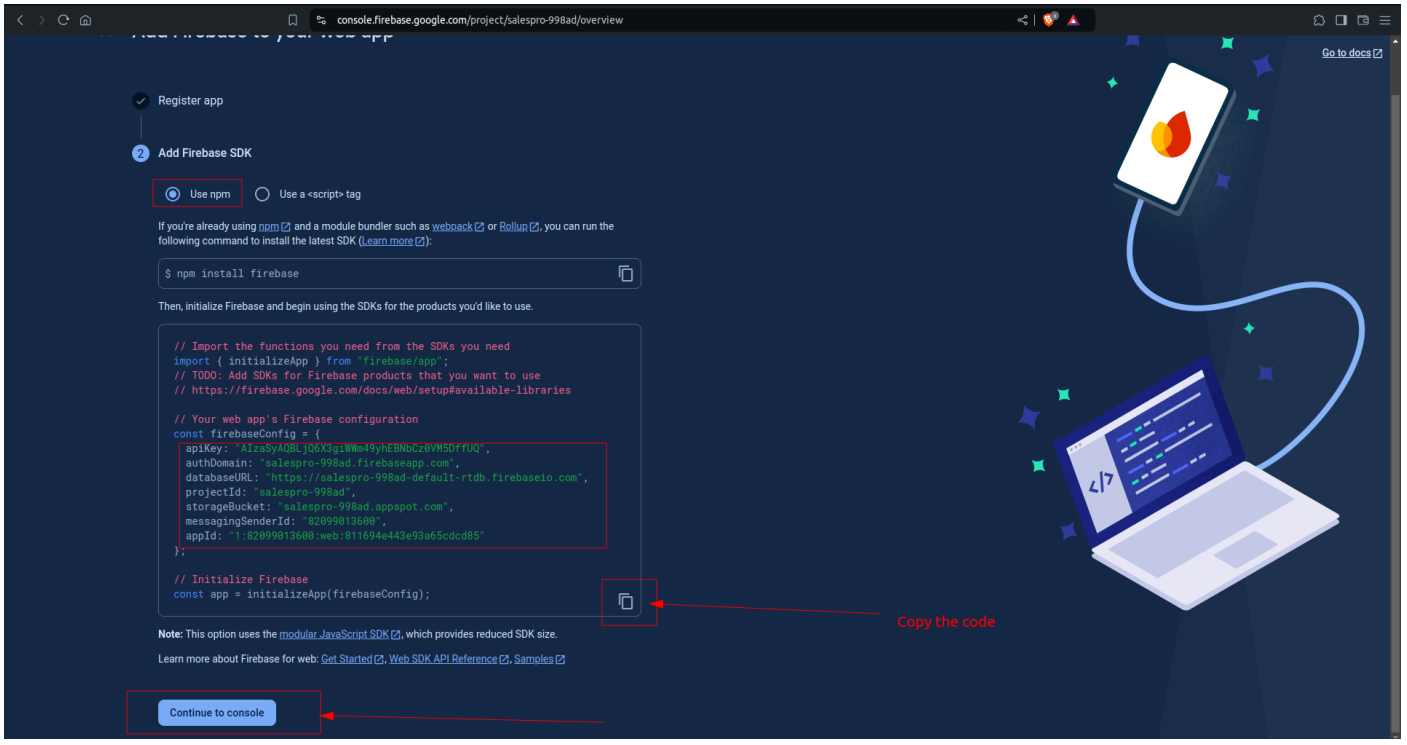
Press on the **Web** button to connect the admin with firebase.



Give a name for your salespro admin panel and press the **Register app** button.

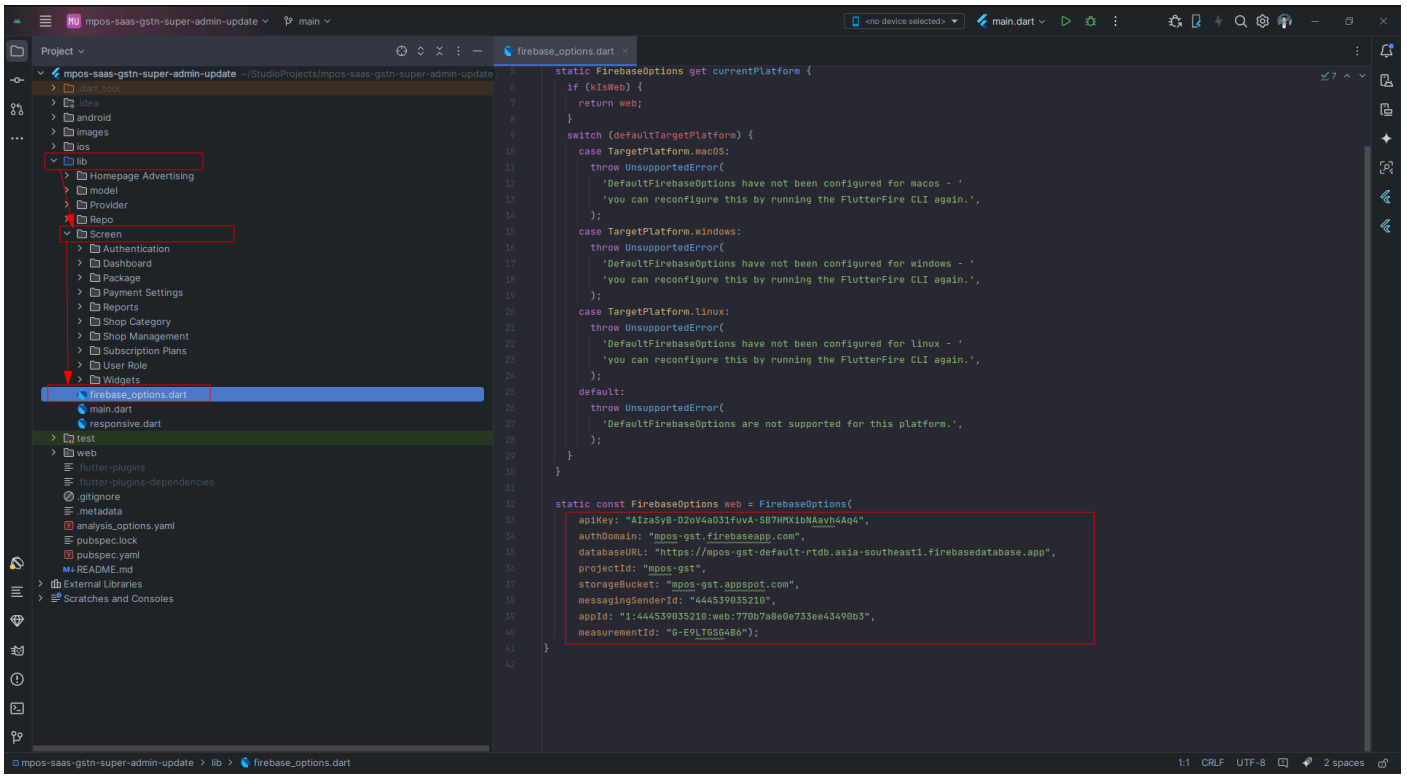


Here make sure you are on the **Use npm** radio button. Then copy the code inside the **`firebaseConfig`**
Or you can copy them all and keep them in a safe place. And press **Continue the console** button.

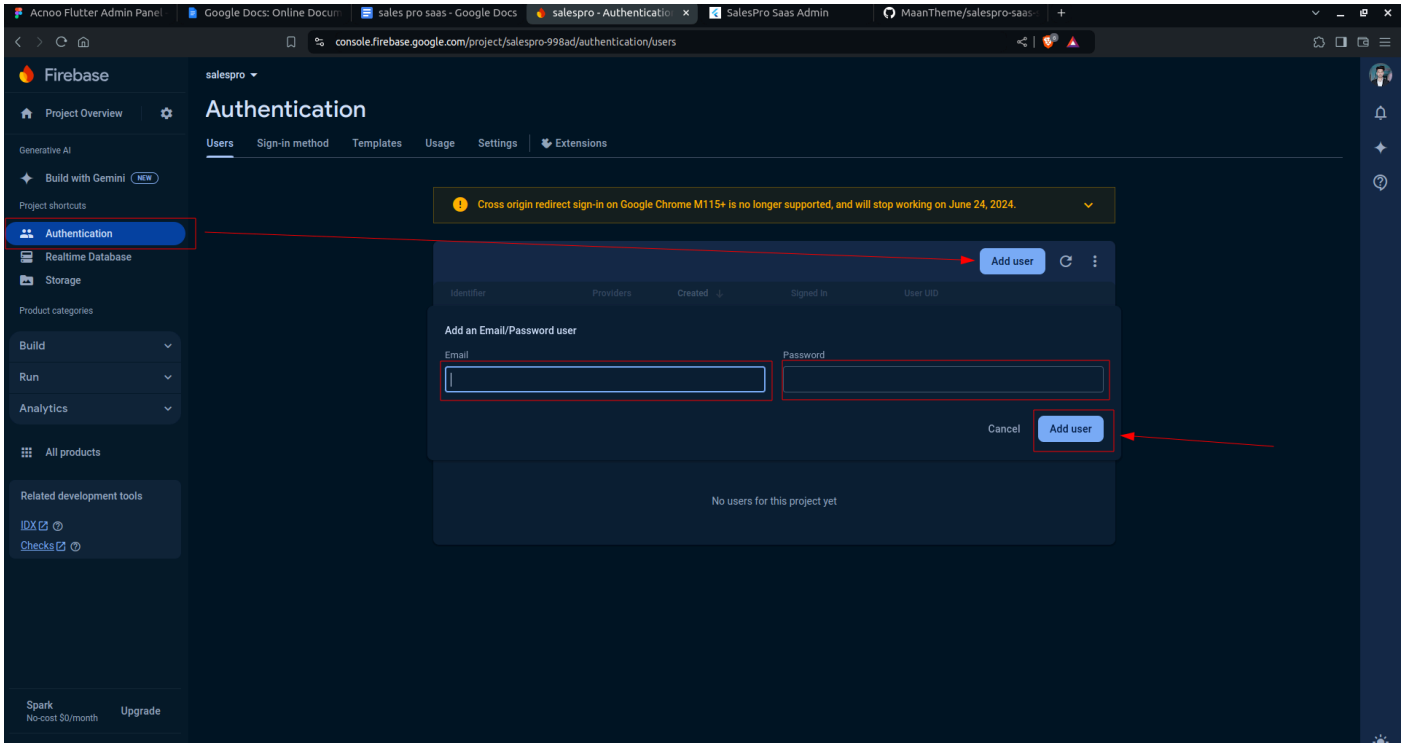


Now go back to the android studio and open the salepro superadmin and the **firebase_options.dart** folder.
Path: **/mpos-saas-gstn-update/lib/Screen/firebase_options.dart**

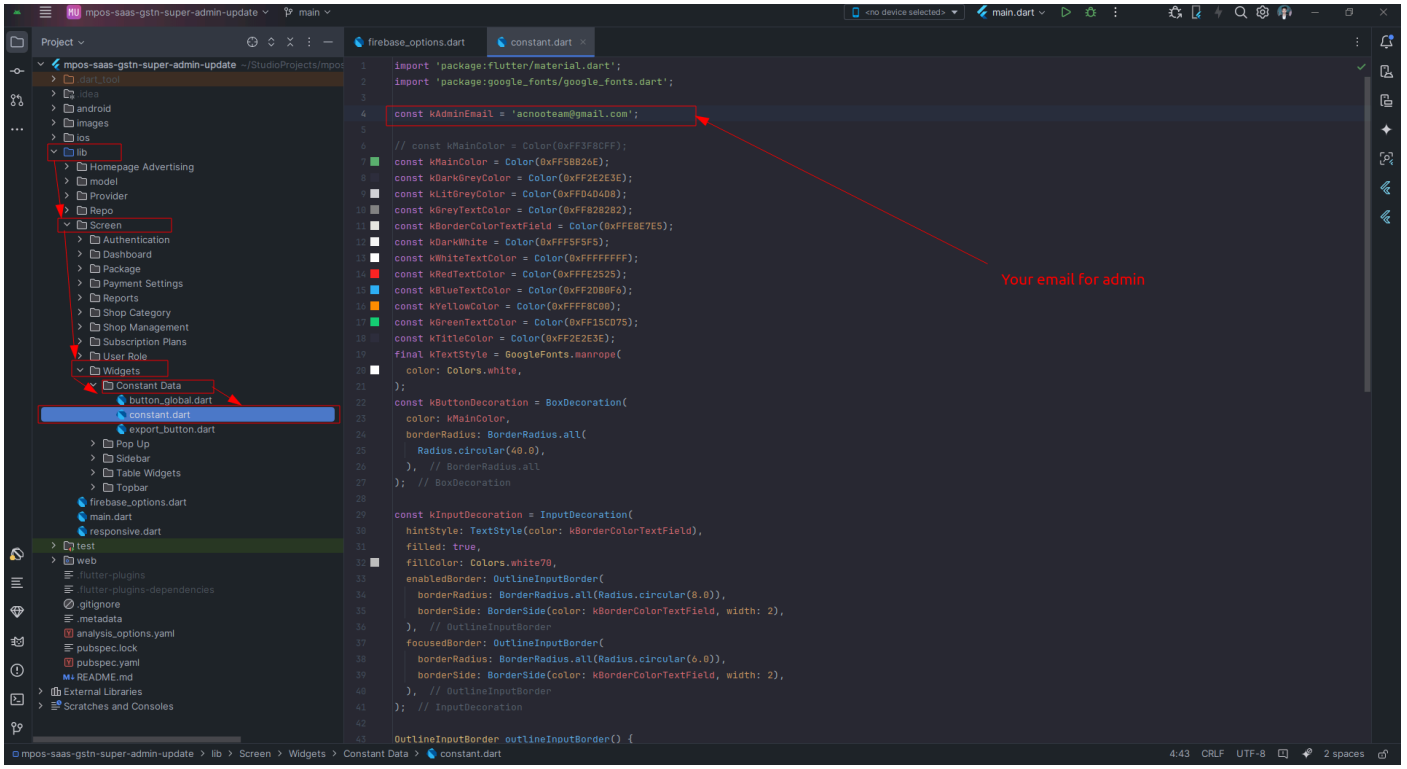
Now replace the credentials which you copied from the **firebase console.** And paste them, and make sure you have pasted all the credentials properly and valid.



Again go back to the **Firebase console,** and Select the **Authentication** from the sidebar and press **Add User.** Here give an email and a password for your **admins** login credentials. And Press **Add User** button under the text box.
Note that this will need every time you login to your admin panel.

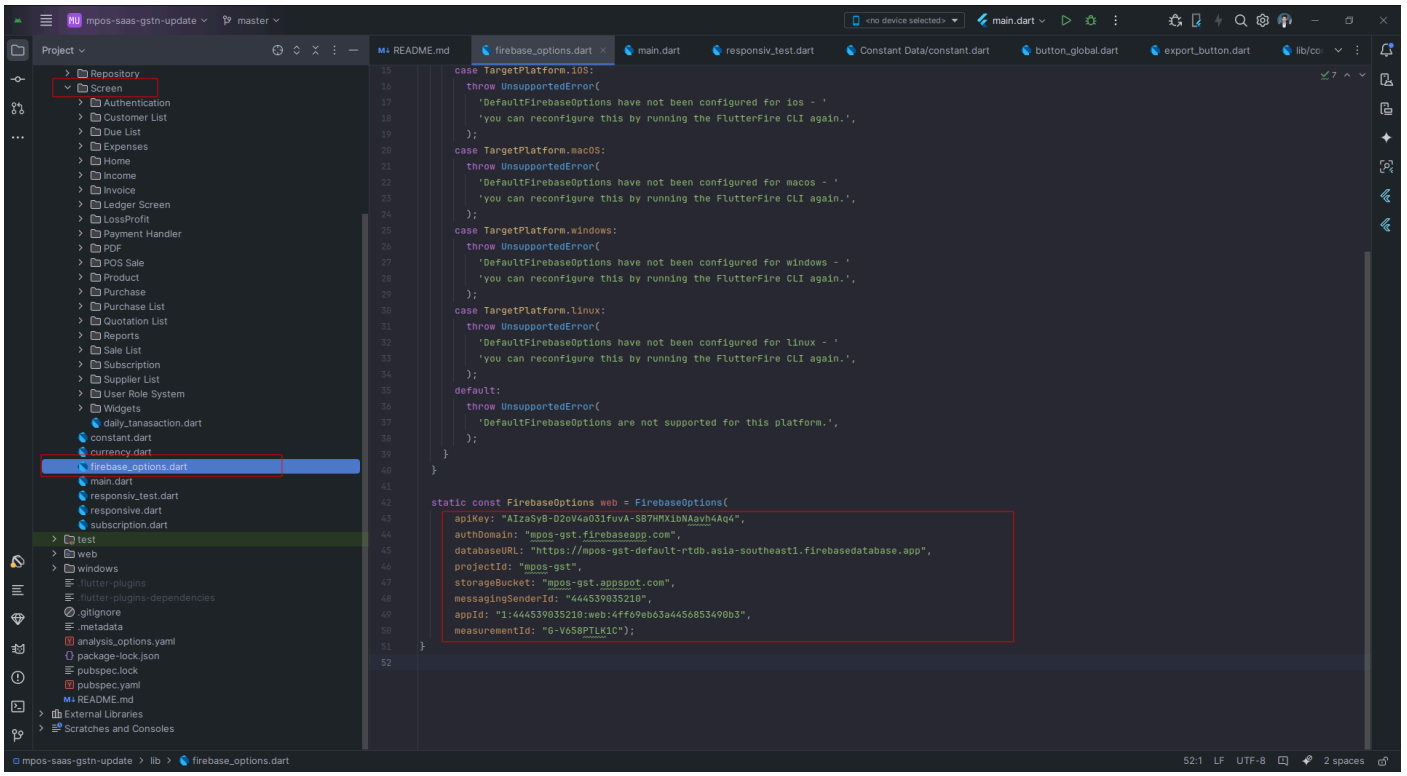


Now go back to the android studio and open the salepro superadmin and the **constant.dart** folder.
Path: **/mpos-saas-gstn-super-admin-update/lib/Screen/Widgets/Constant Data/constant.dart**



Firebase setup is complete for Mpos SAAS Superadmin.

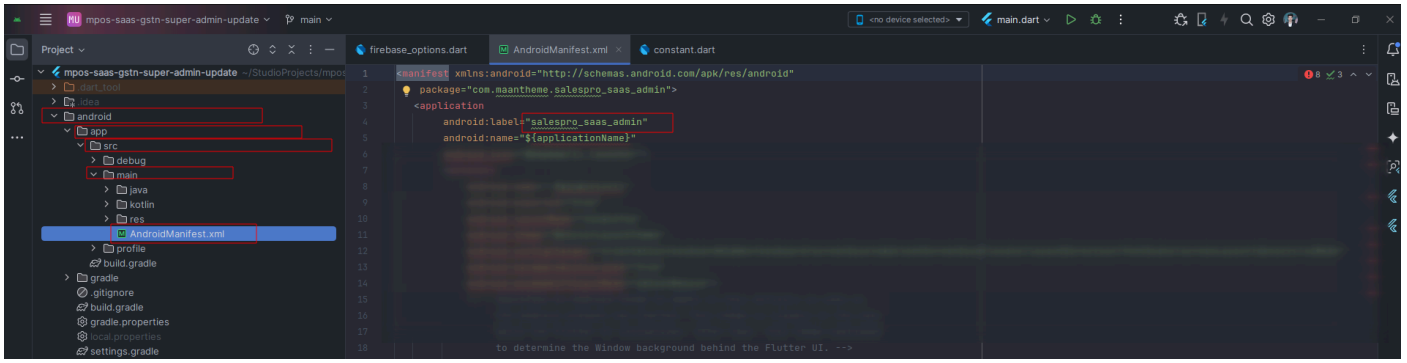
Now replace the same credentials to the **Mpos Web App** which you copied from the **firebase console.**



Change the app name

MPOS Super Admin

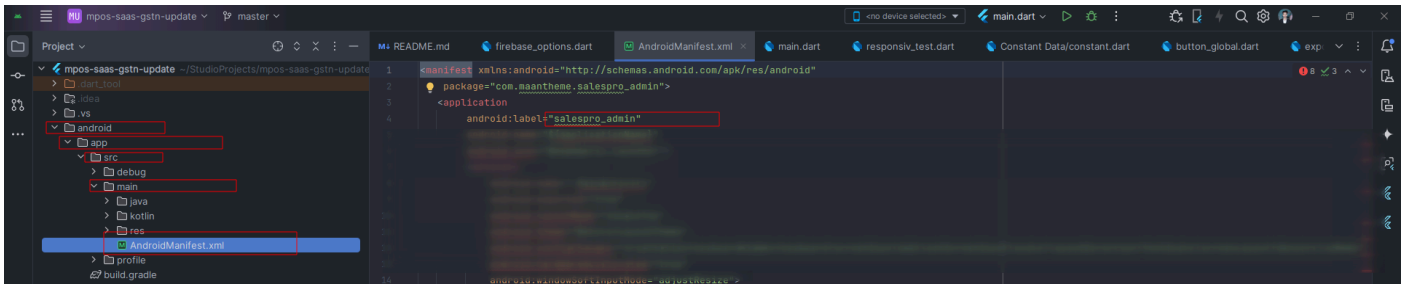
To change the app name you have to go to the **AndroidManifest.xml** file.
path: Project/android/app/src/main/AndroidManifest.xml



After opening the file change the **android:label =”AppName”**
And build the app. The App name will be changed.

MPOS Web App

To change the app name you have to go to the **AndroidManifest.xml** file.
path: Project/android/app/src/main/AndroidManifest.xml



After opening the file change the **android:label =”AppName”**
And build the app. The App name will be changed.

Change Logo & logo inside admin panel

Select Images:

- Choose the image that you want to use for the main logo.
- Ensure the image is appropriate and formatted correctly for the app's display requirements.

Rename Images:

- Rename the selected image with the following filenames:
    - logo: For MPOS Admin “**mpos.png” &**
    - logo: For web App **“mlogo.png”**
- Make sure to keep the file extension as .**png**.

Replace Existing Images:

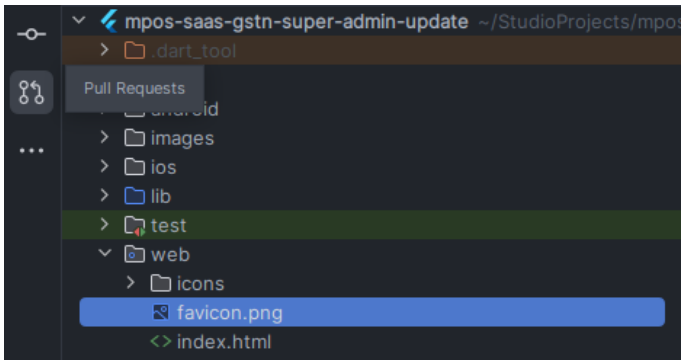
- Locate the folder where the current logo & avatar images are stored. This is typically named images
- Copy and paste your newly renamed images into this folder.
- If prompted, allow the new images to replace the existing ones.

Change favicon:

Rename Images:

- Rename the selected image with the following filenames:
    - Splash logo: **favicon.png**
- Make sure to keep the file extension as .**png**.

Replace Existing Images, **icons** folder inside the **web**

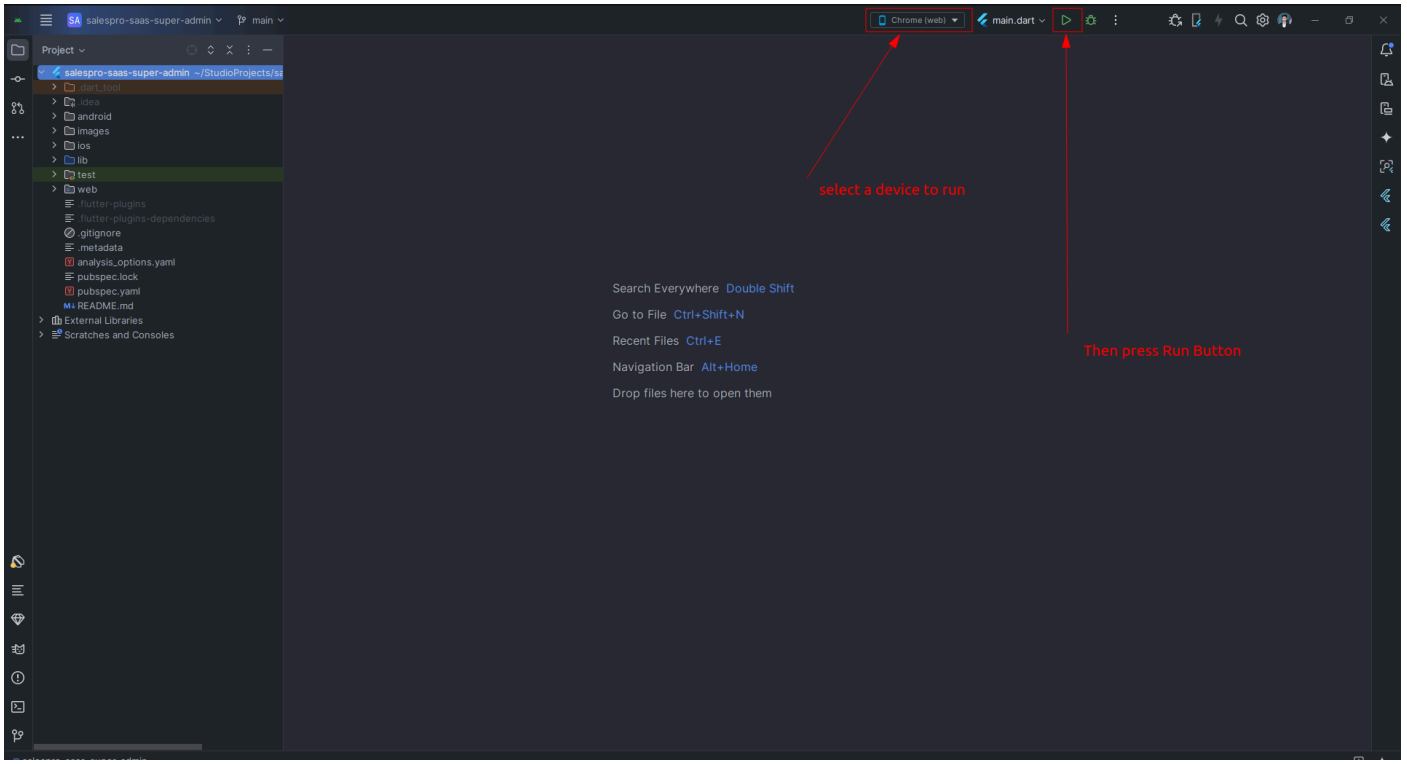


Restart the App:

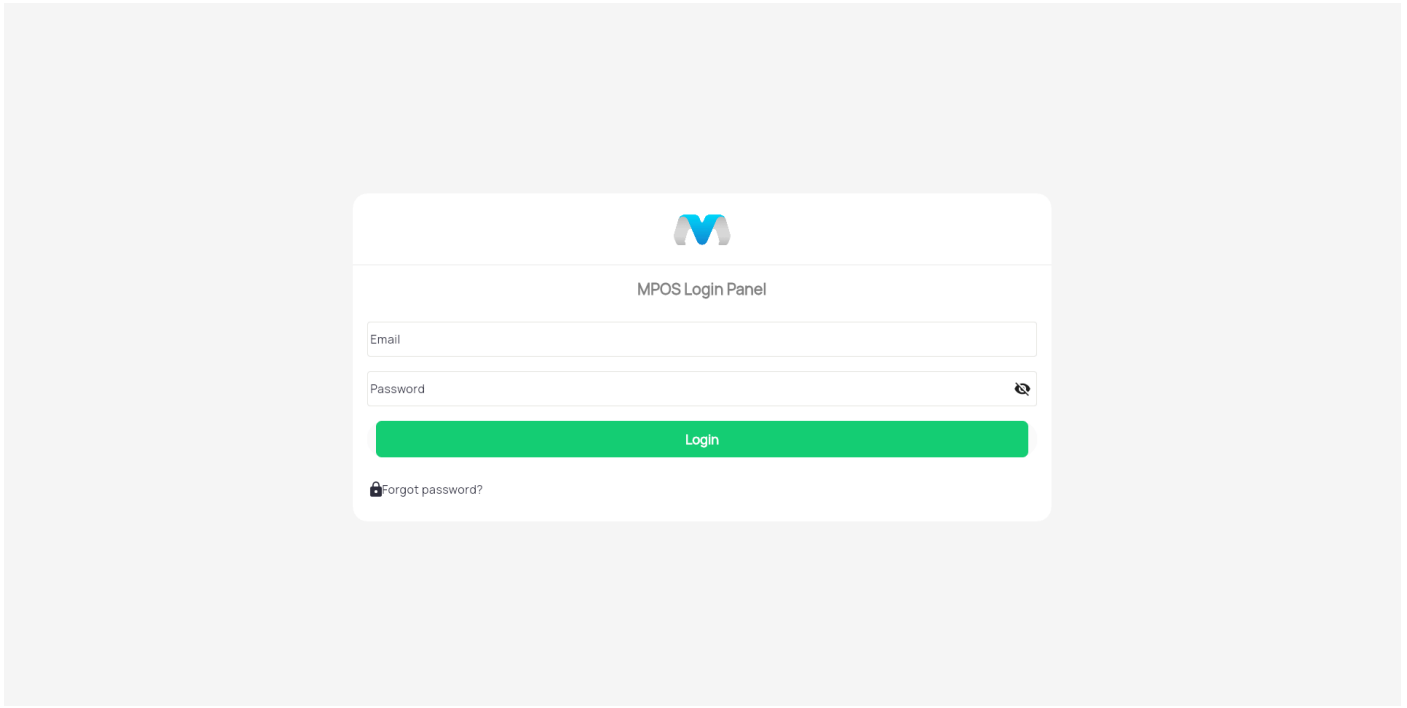
- Close the app completely to ensure all changes take effect.

Restart the app and verify that the new onboarding images are displayed correctly.

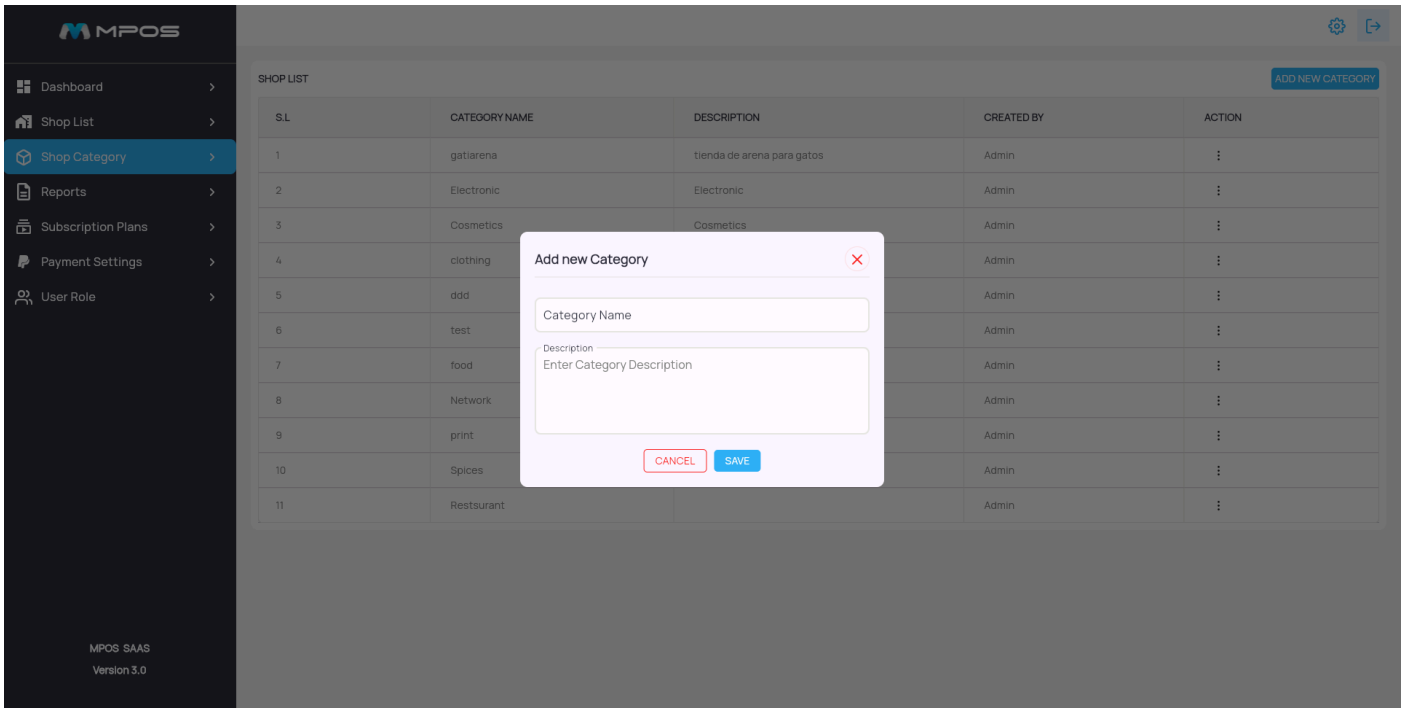
Now Run the Super Admin First.



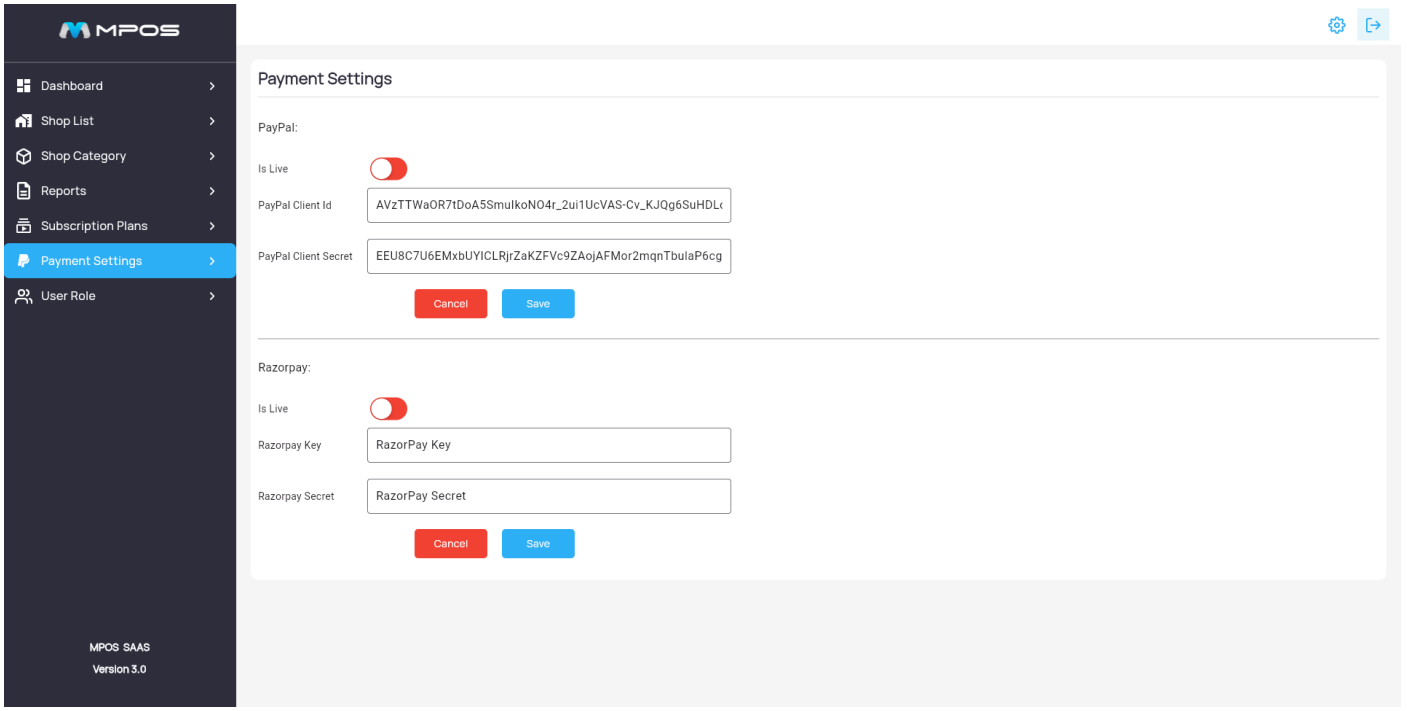
After opening, sign in to the admin with your email which you have added for admin.



Now Add some categories for the users.



Add Payment credentials



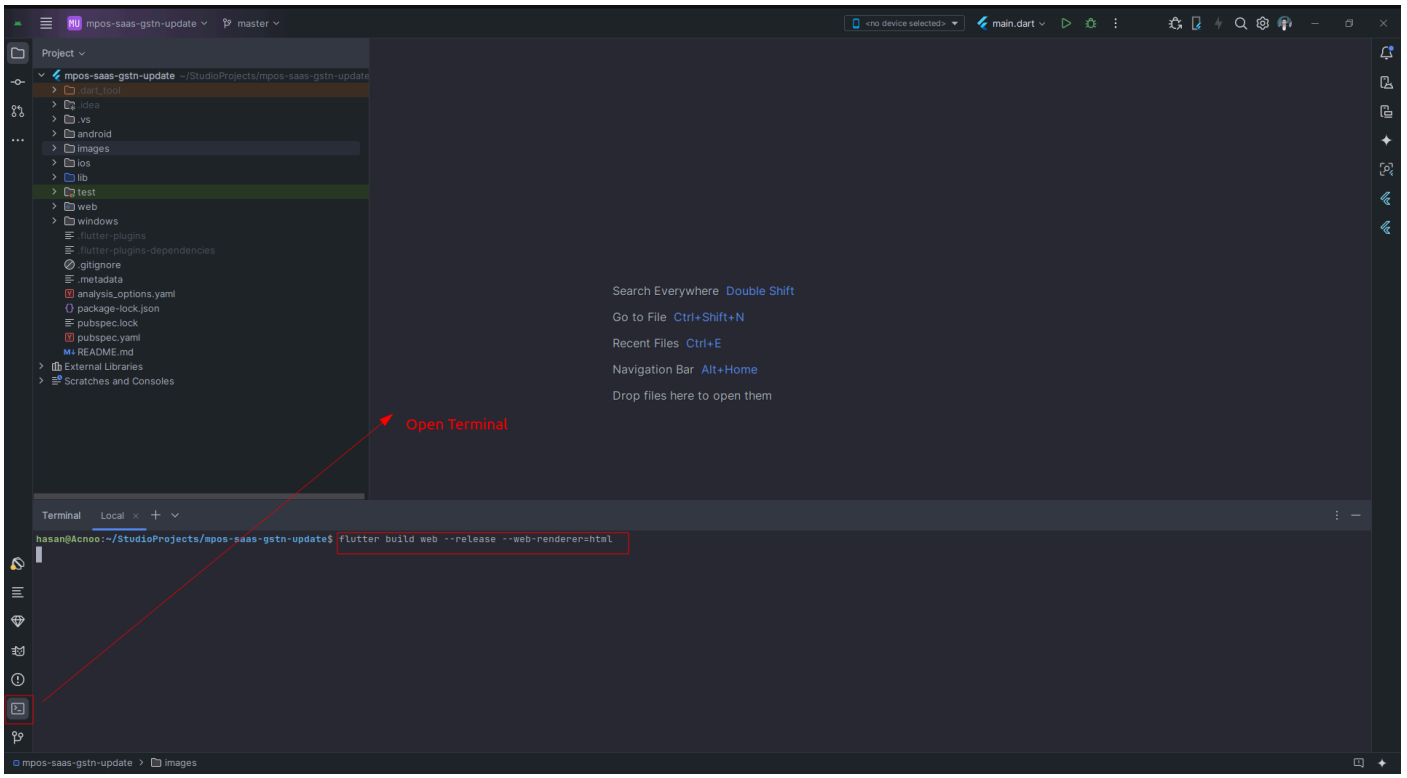
That's all for MPOS Saas Super Admin

Build File For Server Upload

To build the web application to upload the server, you have to open the terminal section.
With android studio you can open it from the bottom bar of the environment.
In the terminal write the command. Same for Admin and the web app.

flutter build web --release --web-renderer=html

and press enter.



It will take some time to build the file.After build the app, you have to go to the directory where you store the file. And open the web folder which is inside the build folder.

Path: /YourFolderName/mpos-saas-gstn-update/web

Now compress the web folder and extract it to the server

Help & Support

Support skype: https://join.skype.com/invite/kEPqImF1Vfqk
Support email: acnooteam@gmail.com

You can expect an answer within 24-48 hours, usually as soon as possible in the order they were received.