B2X_CODES

Grocery Store

Complete Setup Guide

Flutter Installation

How to install flutter on your local machine?

Prerequisites

- Flutter SDK
- Dart
- Java 8 or above
- Android Studio
- Xcode (Mac)
- VS Code

How to install?

Flutter SDK

Download the following installation bundle to get the latest release of the Flutter SDK. Latest versions are given below for particular platform:

- Windows: https://flutter.dev/docs/development/tools/sdk/releases?tab=windows
- MacOs: https://flutter.dev/docs/development/tools/sdk/releases?tab=macos
- Linux: https://flutter.dev/docs/development/tools/sdk/releases?tab=linux

Please follow the Flutter installation guide on their official website and use the latest versions.
Official website: https://flutter.dev/docs/get-started/install

Android Studio

Download Android studio from the below given link for particular platform and follow the onscreen instructions for installation.
Official download link: https://developer.android.com/studio#downloads

Xcode (Only for MacOs)

Download Xcode from the below given link and follow the onscreen instructions for installation.
Official download link: https://developer.apple.com/xcode/

VS Code

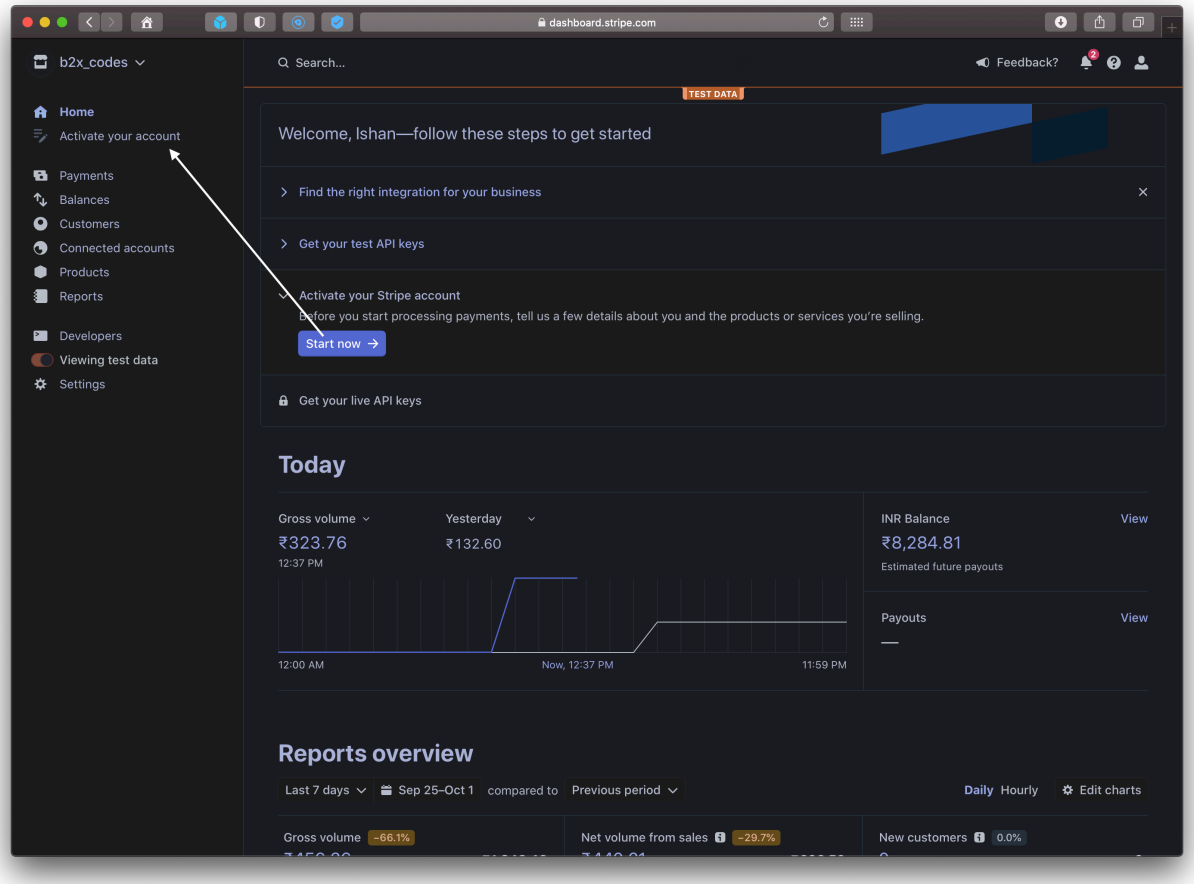
Download VS Code from the below given link and follow the onscreen instructions for installation.

Official download link: https://code.visualstudio.com/download

Stripe API Setup

The grocery app uses Stripe payment API for payments. Go to the below given link and create an account on Stripe and follow the instructions given below.
Stripe link: https://dashboard.stripe.com/register

Once you have created the account verify your email address and **Activate** the Stripe account by clicking on **Activate your account** found on left side bar of the stripe dashboard.
For activating you will have to fill out all the required information and once that is done submit and wait for the activation.

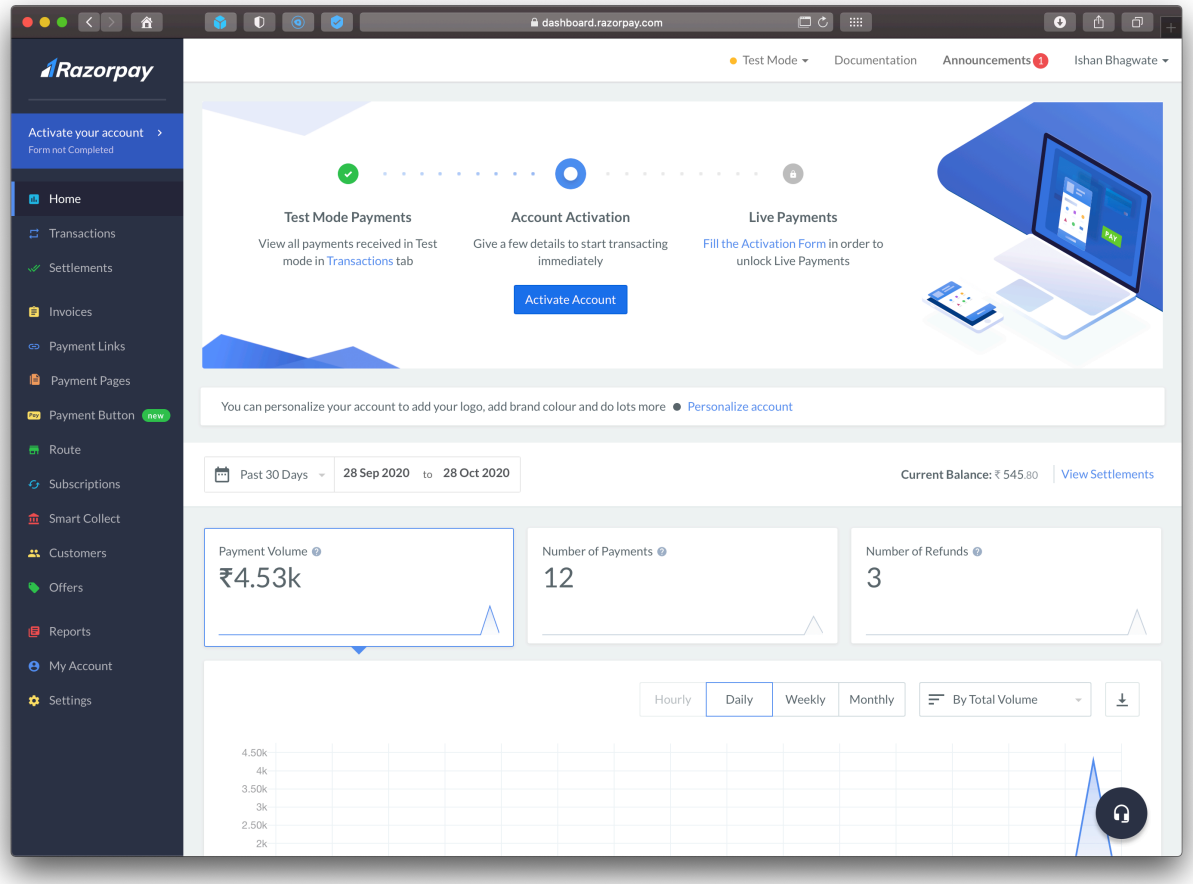


After the activation get your API keys (i.e: Publishable key & Secret key) then copy them into a notepad for a while (DO NOT SHARE THESE WITH ANYONE).

Razorpay API Setup

The grocery app uses Razorpay payment API for payments. Go to the below given link and create an account on Razorpay and follow the instructions given below.
Razorpay link: https://razorpay.com

Once you have created the account verify your email address and **Activate** the Razorpay account by clicking on **Activate your account** found on left side bar of the razorpay dashboard.
For activating you will have to fill out all the required information and once that is done submit and wait for the activation.



After the activation get your API keys (i.e: Key ID & Key Secret) then copy them into a notepad for a while (DO NOT SHARE THESE WITH ANYONE).

Firebase Functions Setup

Go to this link: https://firebase.google.com/docs/functions/get-started#set-up-node.js-and-the-firebase-cli
Now follow the given instructions to setup firebase functions. You will need to setup **node.js, npm** and **firebase-tools** in order to deploy the firebase functions. Once all these pre-requisites are completed follow the below given instructions:

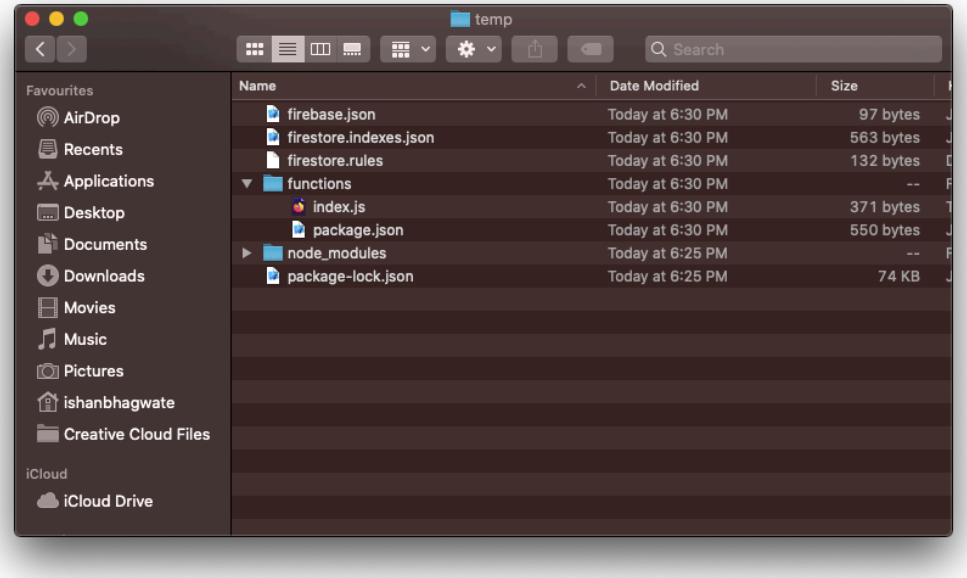
Initialise **firebase functions** by following the instructions in the given link:
https://firebase.google.com/docs/functions/get-started#initialize-your-project

Make sure to not enable “ES Lint” as it will give these errors.

You’ll be asked if you want to enable es lint after you run
$ firebase init functions

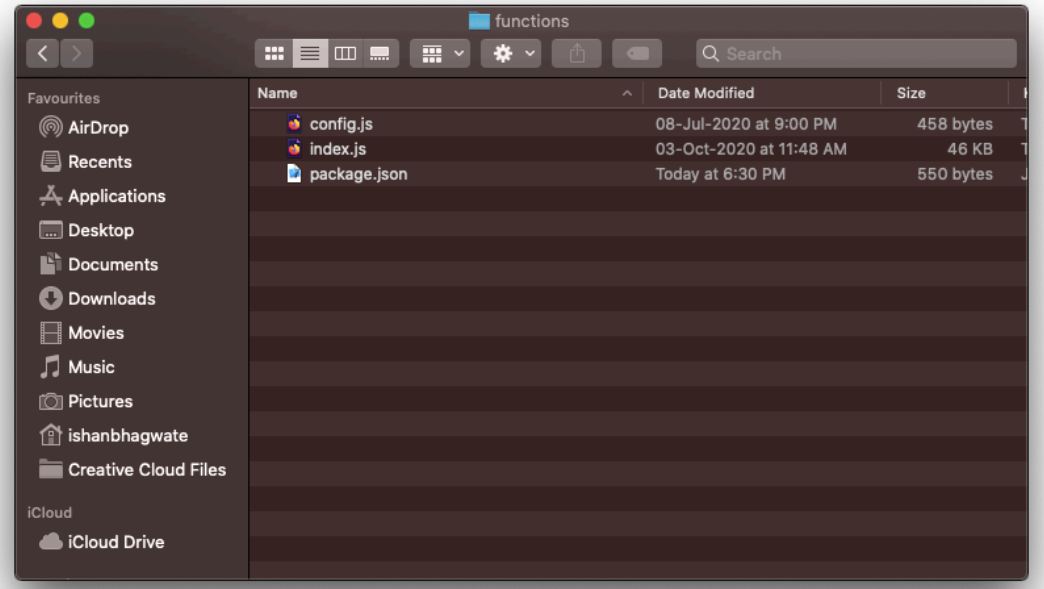
So just type - “N”

Once everything is setup you will have a folder similar to as shown:

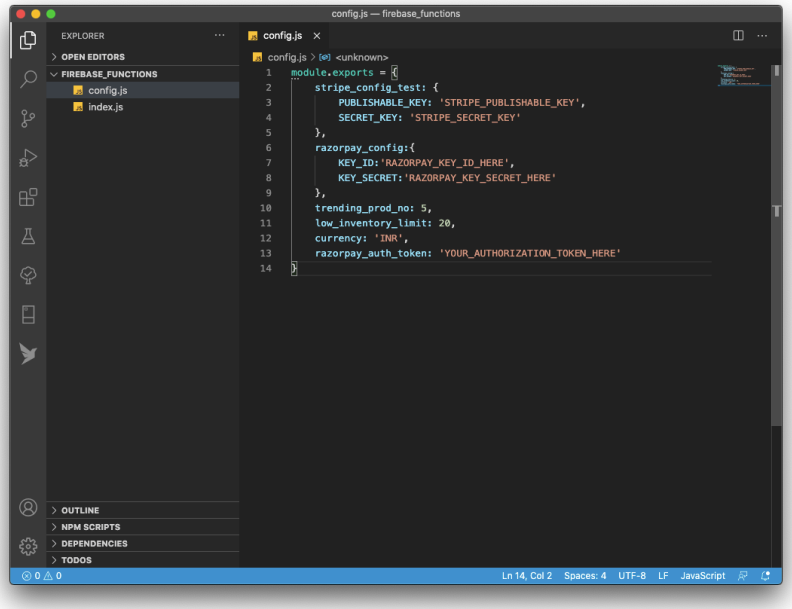


B2X_CODES

Open the **grocery_store_firebase_function** folder in the provided setup files folder and copy all the files i.e: **index.js, config.js and package.json** to the newly created firebase functions folder and paste it in **functions** folder and replace **index.js** and **package.json** as shown:



After that open **config.js** file and edit the following two values:
PUBLISHABLE_KEY and SECRET_KEY
Replace them with the **Stripe API** keys that we copied earlier.
Also now replace **Razorpay API** keys that we copied earlier.
Also you can modify **low_inventory_limit** and **trending_prod_no** values as per your need.
To get the razorpay_auth_token -> Download Postman and get the Authorization token for it.



Once all this is setup you are ready to deploy the function but first you will have to subscribe to Blaze plan in firebase as the functions require few features from it.
You can opt for Blaze plan from Firebase console.

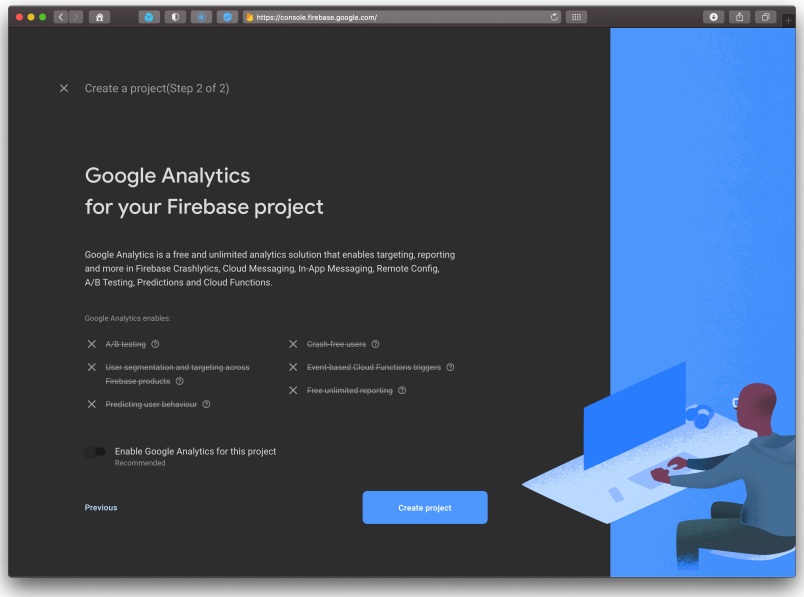
Now if you have done all this type the following command from the terminal in VS-Code:
—> firebase deploy

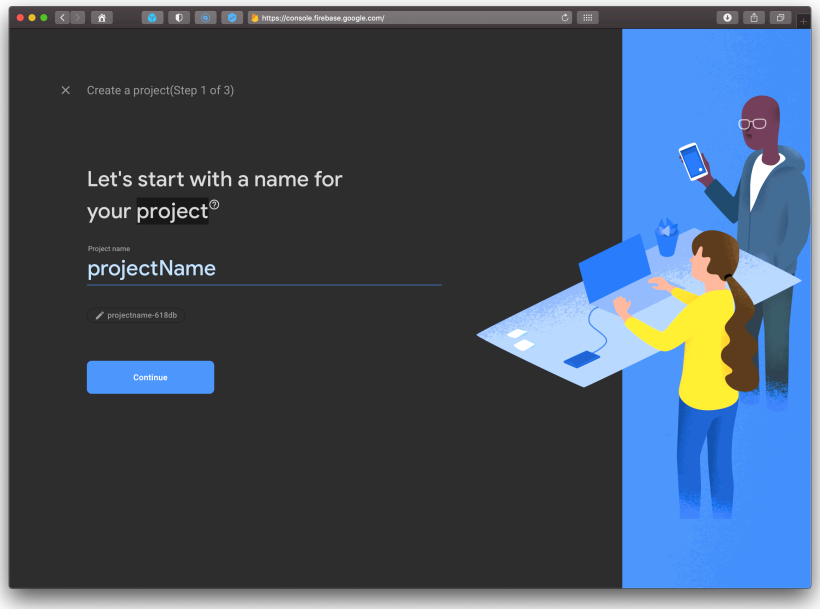
Firebase Setup

Grocery Store uses Firebase as backend, to setup firebase go to the link below and create a new google account that will be then used as firebase account for the app.
Link: https://firebase.google.com

Once you have created the account go to firebase console and click on **Add project**:
Link: https://console.firebase.google.com/

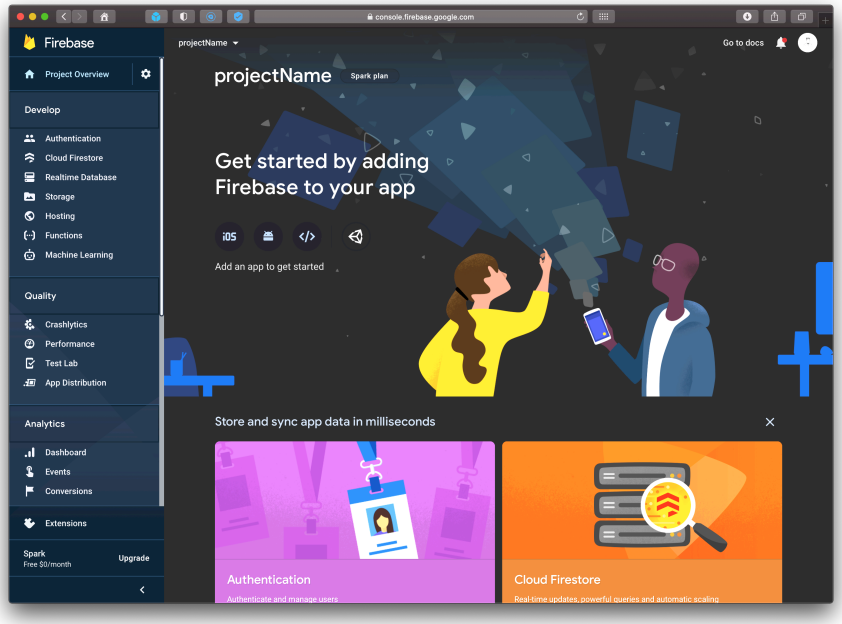
Follow the onscreen instructions to create project as shown below:



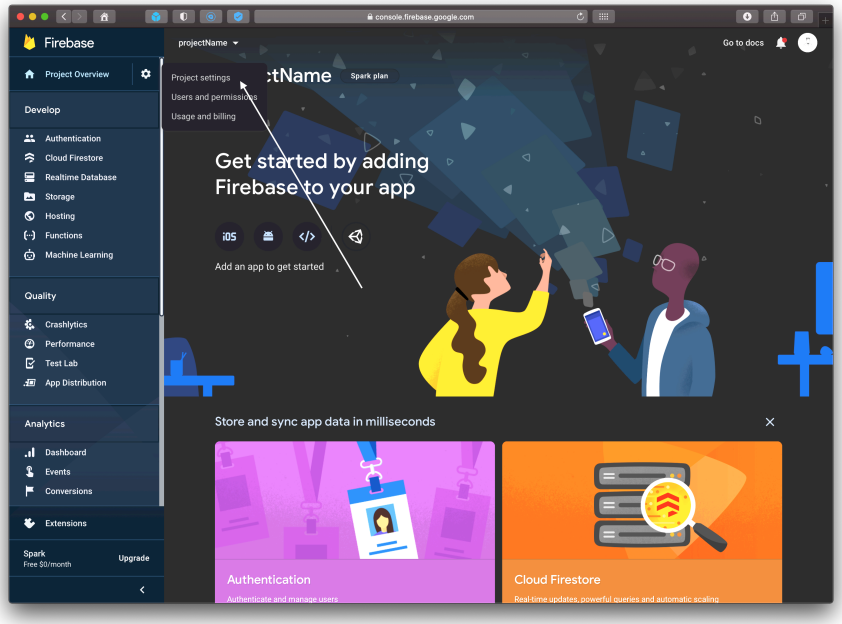


B2X_CODES

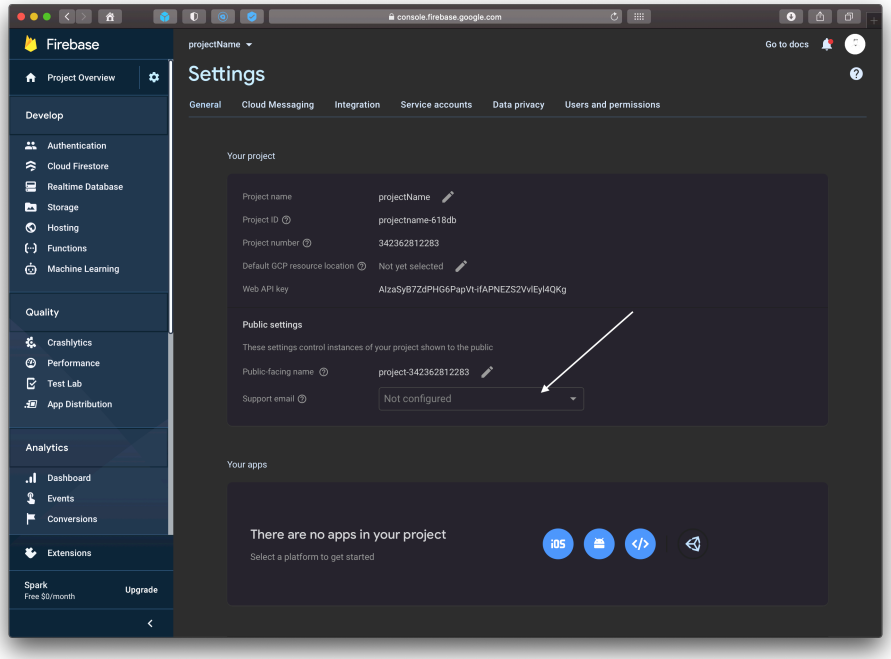
After creating the project go to project dashboard as shown below:



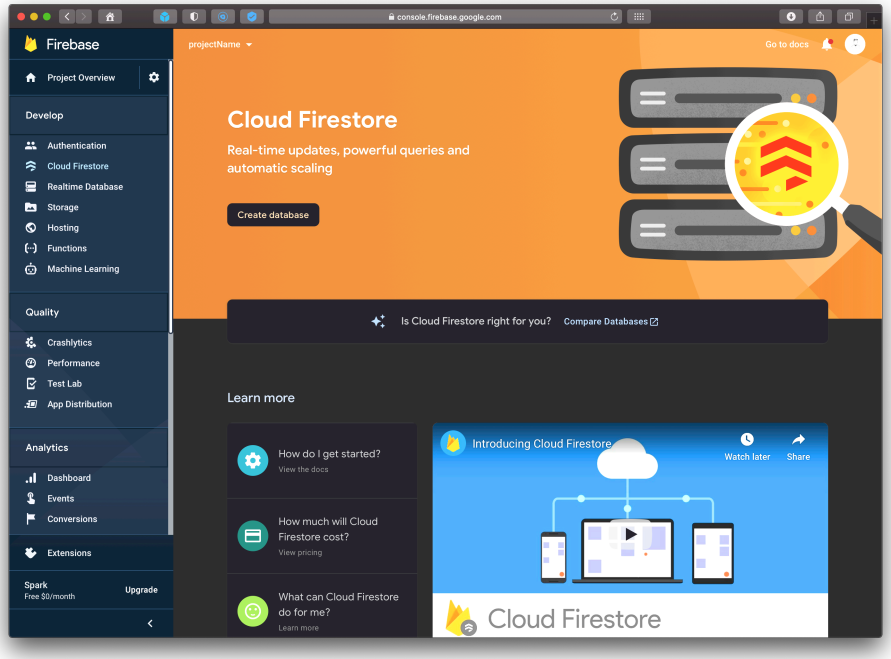
Go to **Project Settings**:



Now in **Project Settings** select your **Support email** as shown. This is IMPORTANT to setup otherwise app won't work as intended.

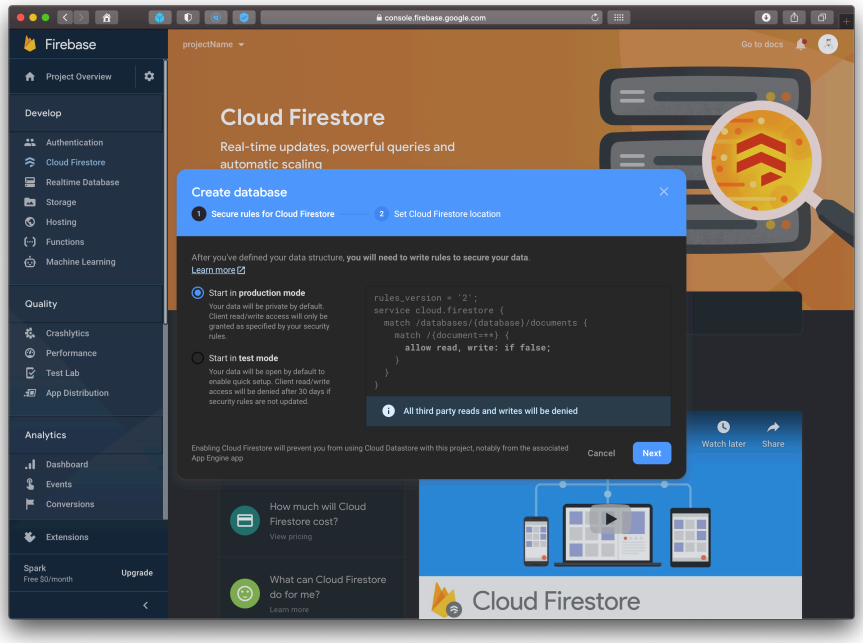


Click on **Cloud Firestore** shown in the left bar and click on **Create database** as shown:

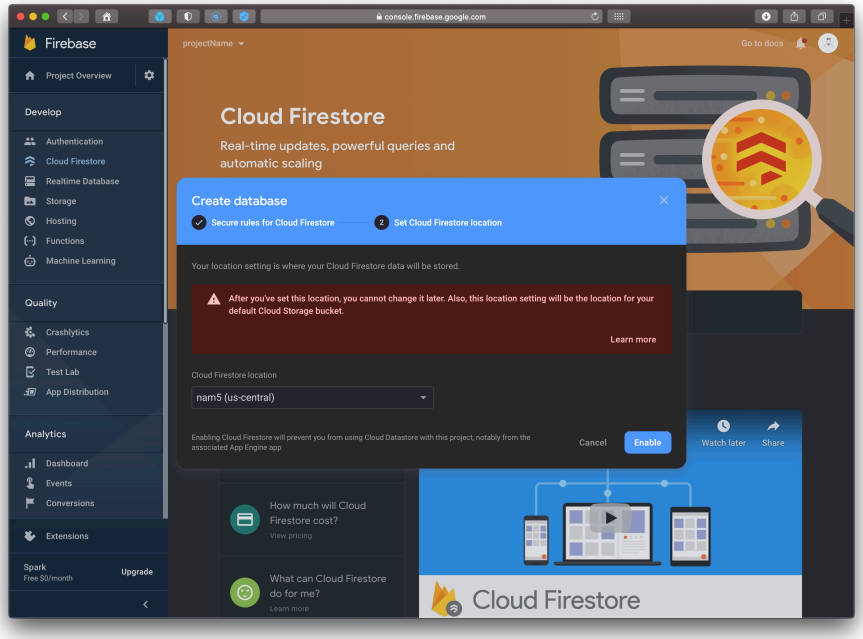


B2X_CODES

Click **Next** as shown:



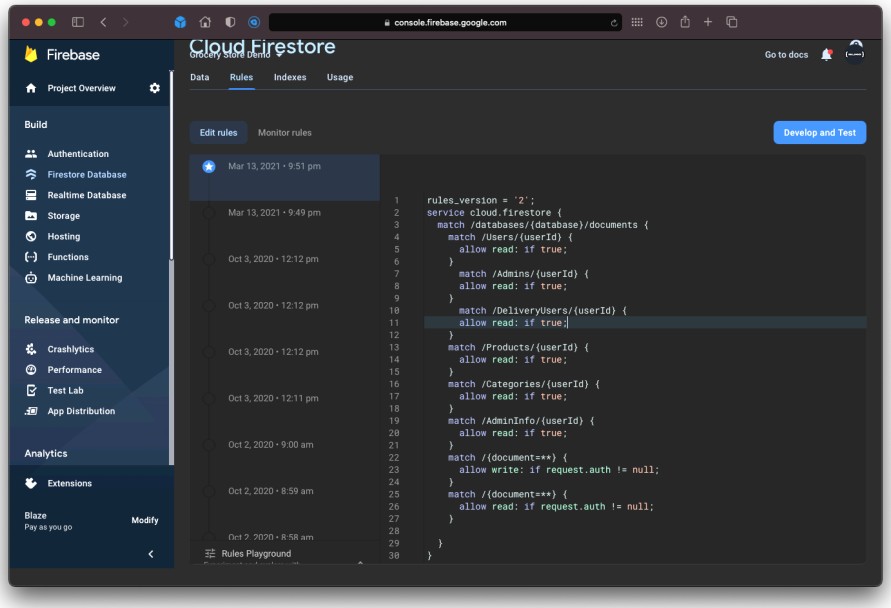
Now select **Cloud Firestore location** based on your region and click **Enable** as shown:
This may take some time to process so please wait for it to finish.



B2X_CODES

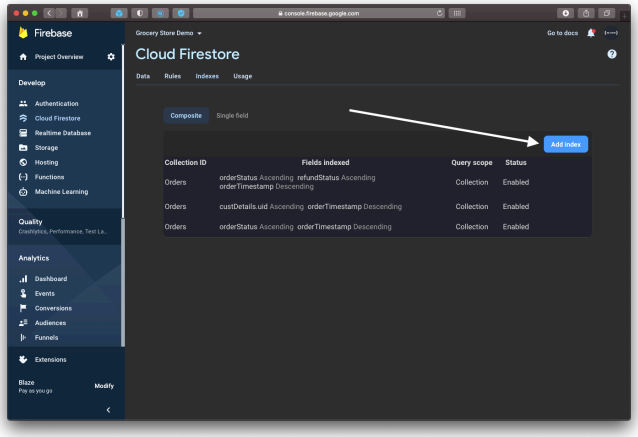
Now click on **Rules** and modify them as shown:
Copy the code given below and **paste** it in your **Rules** —>

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /Users/{userId} {
      allow read: if true;
    }
    match /Admins/{userId} {
      allow read: if true;
    }
    match /DeliveryUsers/{userId} {
      allow read: if true;
    }
    match /Products/{userId} {
      allow read: if true;
    }
    match /Categories/{userId} {
      allow read: if true;
    }
    match /AdminInfo/{userId} {
      allow read: if true;
    }
    match /{document=**} {
      allow write: if request.auth != null;
    }
    match /{document=**} {
      allow read: if request.auth != null;
    }
  }
}
```

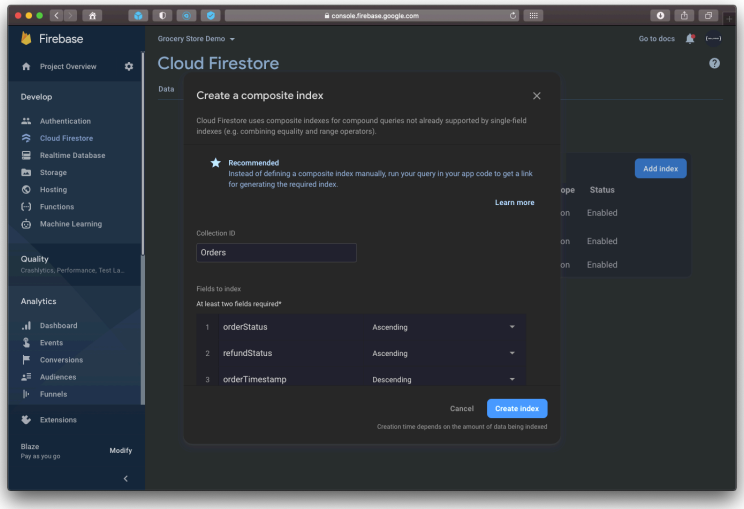


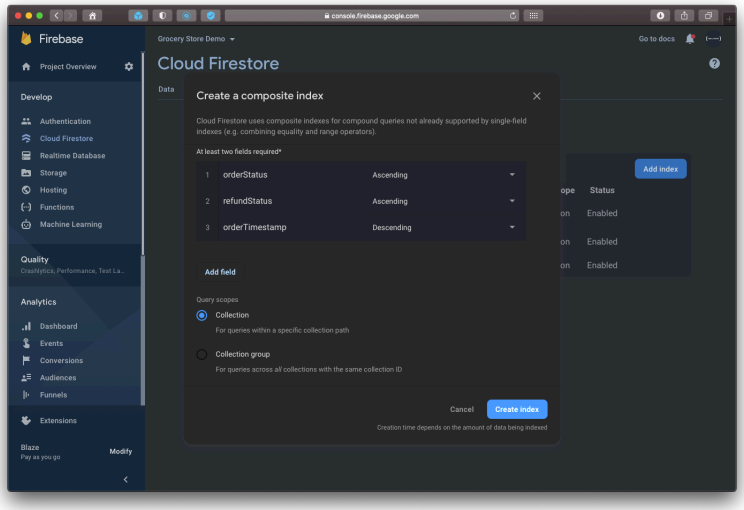
B2X_CODES

Now click on **Indexes** and follow the steps as shown:



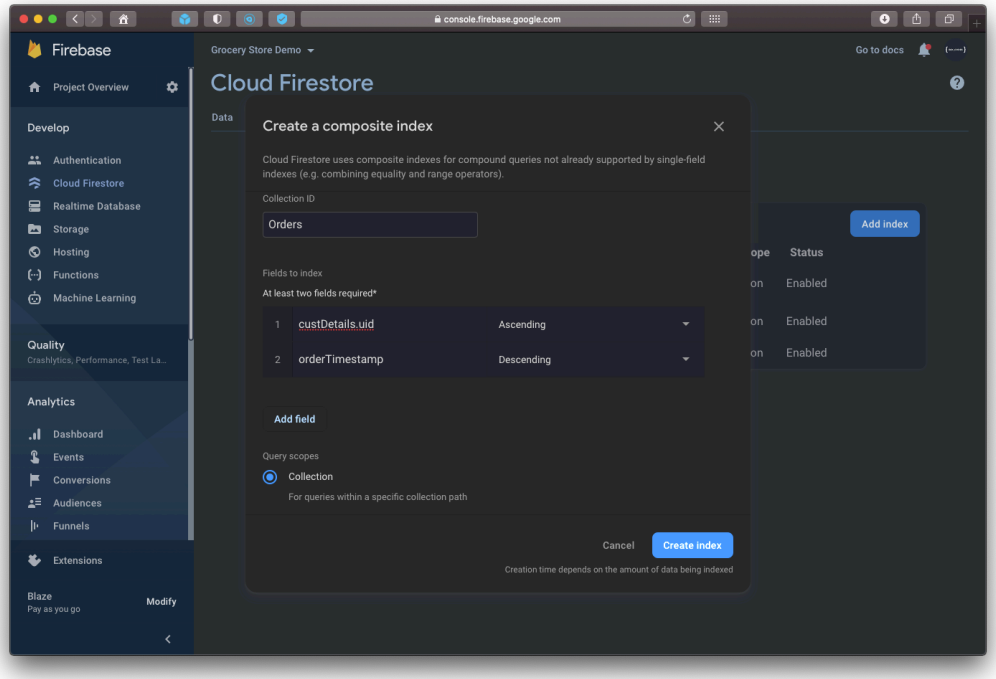
Click on **Add index** and then fill the form as shown below:



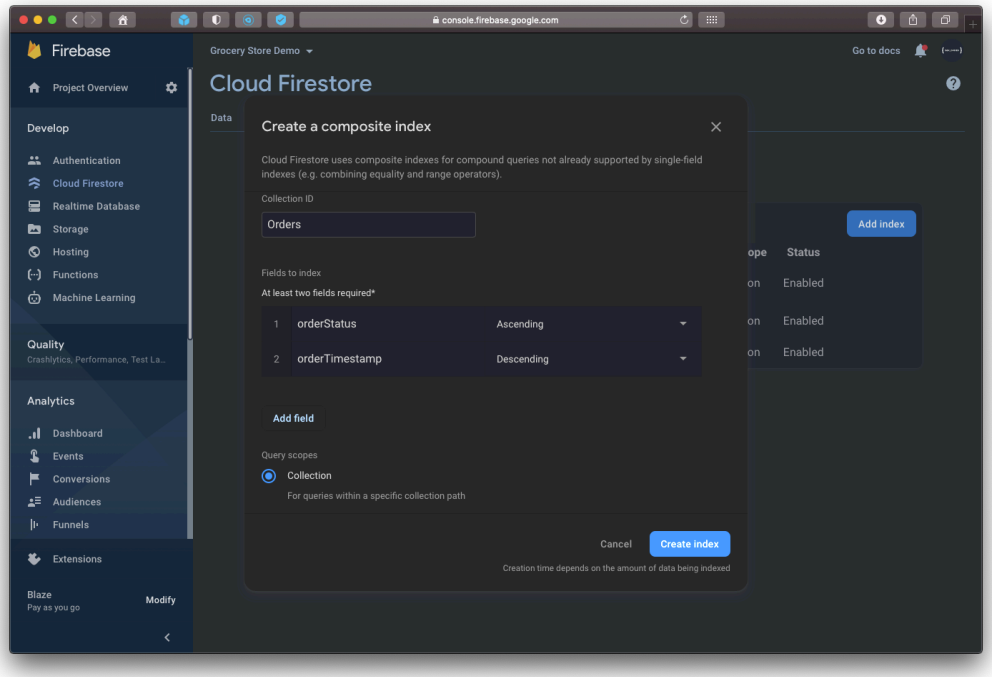


Now again click on **Add index** and follow the same process for these other two indexes:

2nd index details:

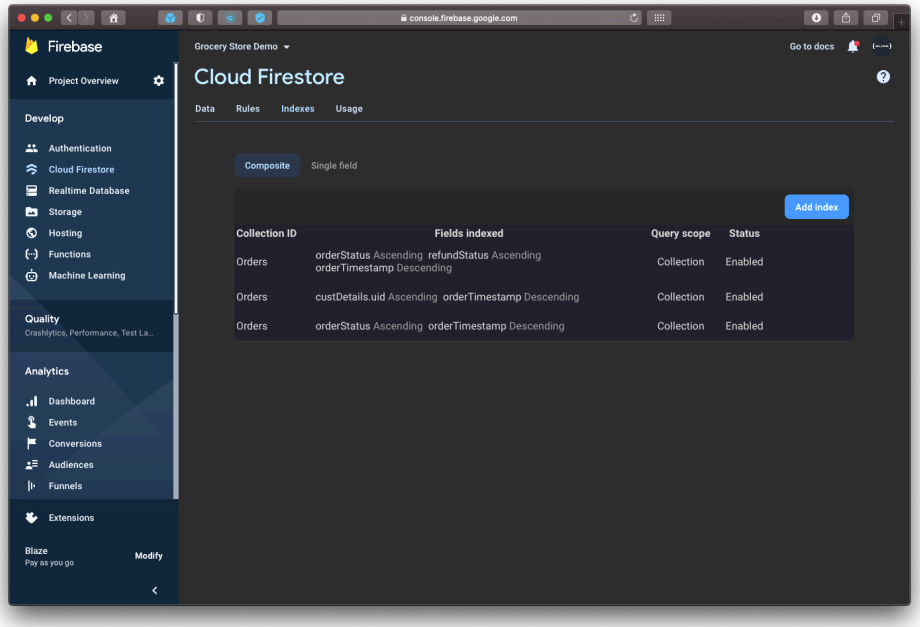


3rd index details:



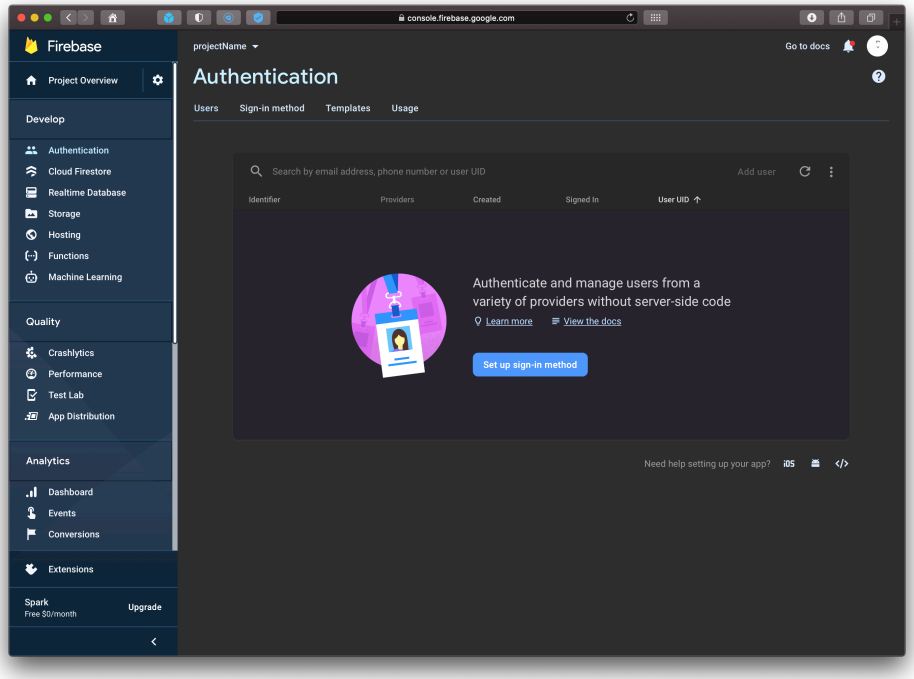
B2X_CODES

At last check if all the three **Indexes** look like given below or not. If yours look different please follow the process again.

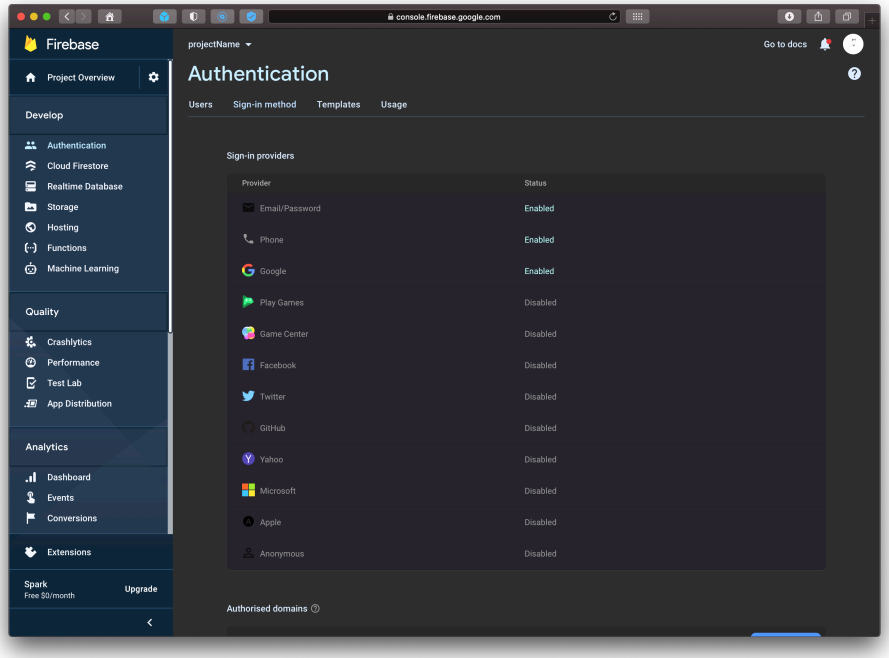


B2X_CODES

Click on **Authentication** shown in the left bar and click on **Setup sign-in method** as shown:

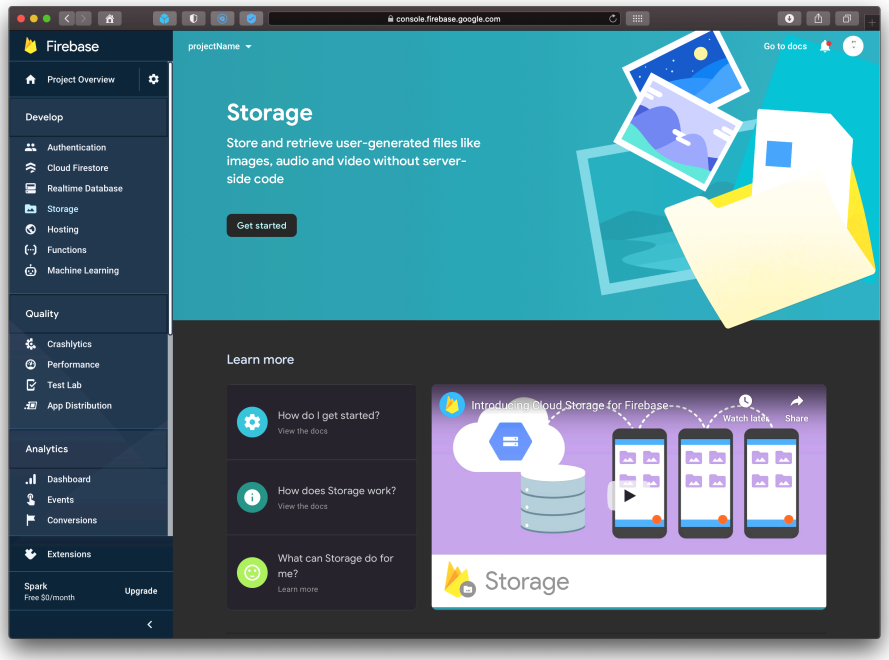


Now **Enable** the **Email/Password, Phone and Google** sign in providers as shown:

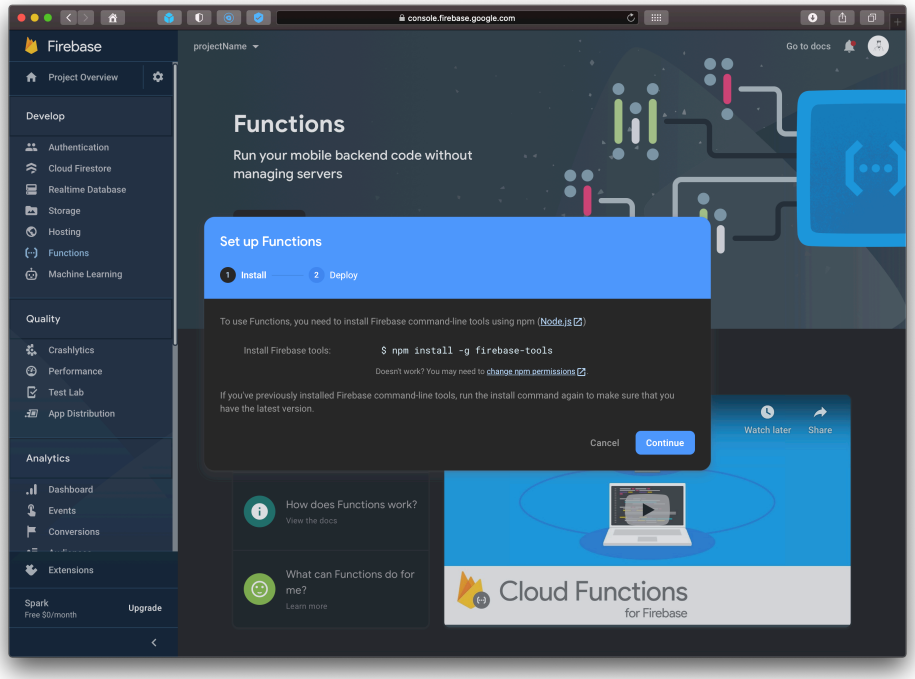


B2X_CODES

Click on **Storage** shown in the left bar and click on **Get started** -> **Next** -> **Done** as shown:

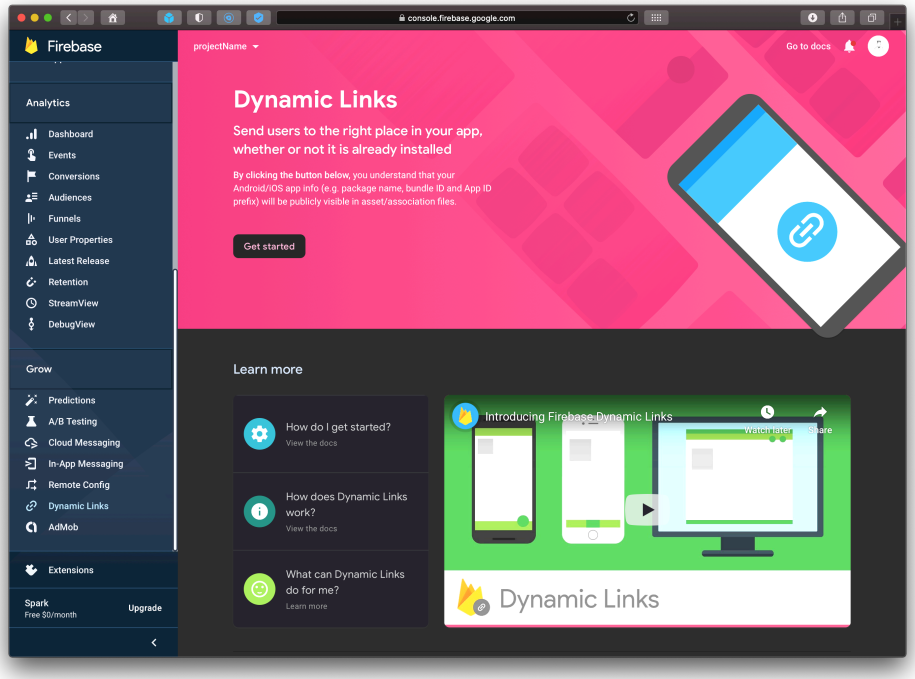


Click on **Functions** shown in the left bar and click on **Continue** -> **Finish** as shown:

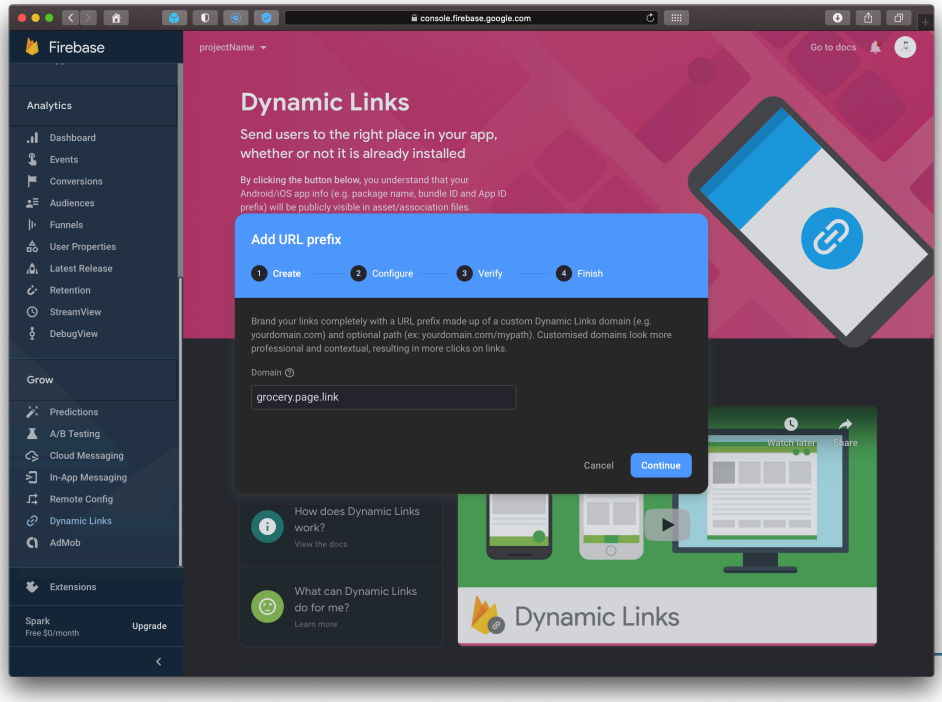


B2X_CODES

Click on **Dynamic Links** shown in the left bar and click on **Get started** as shown:

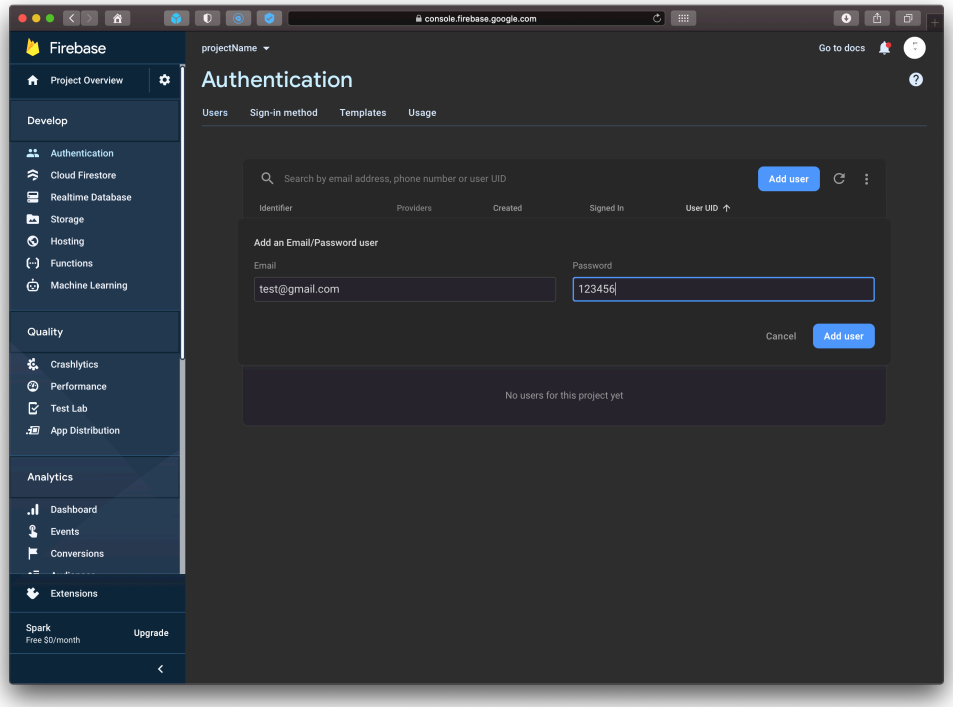


Now enter your domain or if you don't have one then simply type as follows:
YOUR_APP_NAME.page.link
Copy this URL to a notepad for later use.
Click **Continue** -> **Finish**



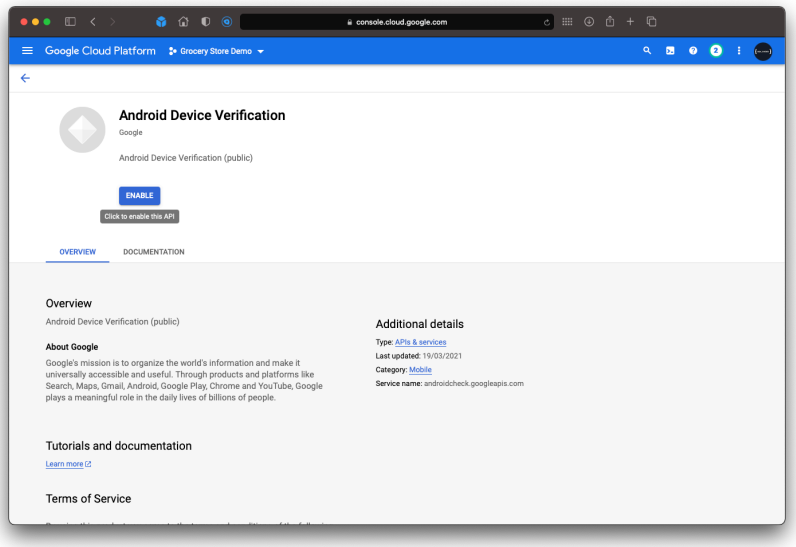
Admin Setup

Go to **Authentication** shown in the left bar and click on **Add user** then enter your **Email and Password** as shown:
This email and password will be used as the **PRIMARY ADMIN** for the Seller/Admin app so do not share these credentials with anyone.



Google Cloud Setup

Go to the following given URL and login with your email which you used for firebase account and follow the steps as shown:
URL: https://console.cloud.google.com/apis/library/androidcheck.googleapis.com

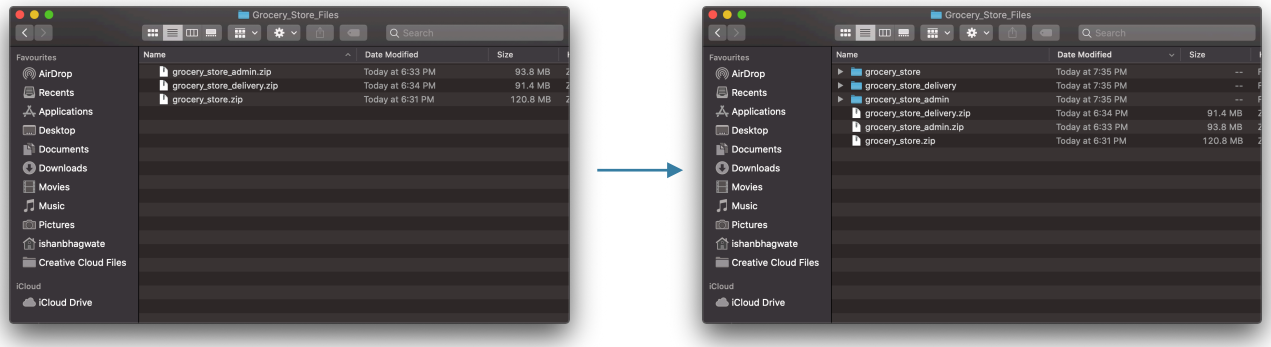


Now open the URL and click on ENABLE.
Make sure to check the appropriate Project is selected in the cloud console which can be seen in upper left side.

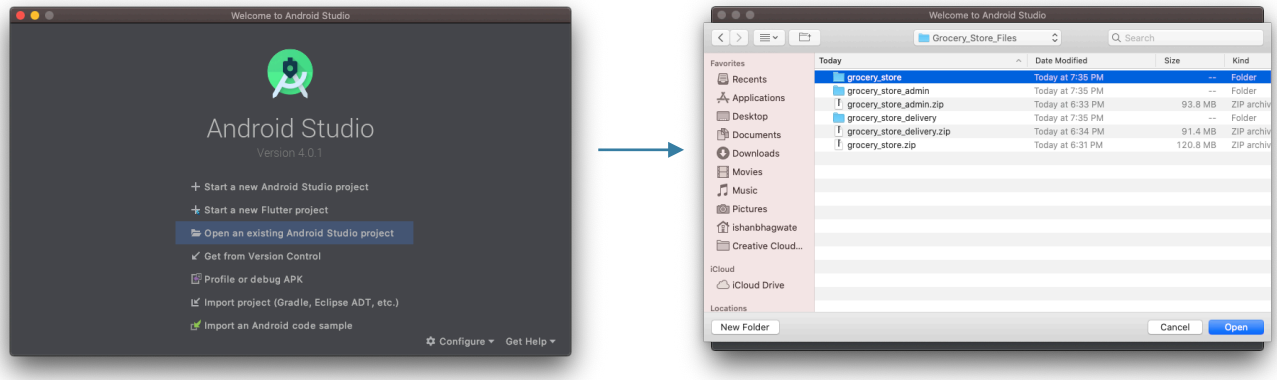
Building the App

This will help you setup and configure the codebase provided with step by step instructions.
Open the .zip file provided and extract all the data.
In that you will find 3 folders. These are all the app codebases i.e: **Grocery Store** (Customer app), **Grocery Store Admin** (Admin app), and **Grocery Store Delivery** (Delivery app).

Extract all three .zip files as shown:



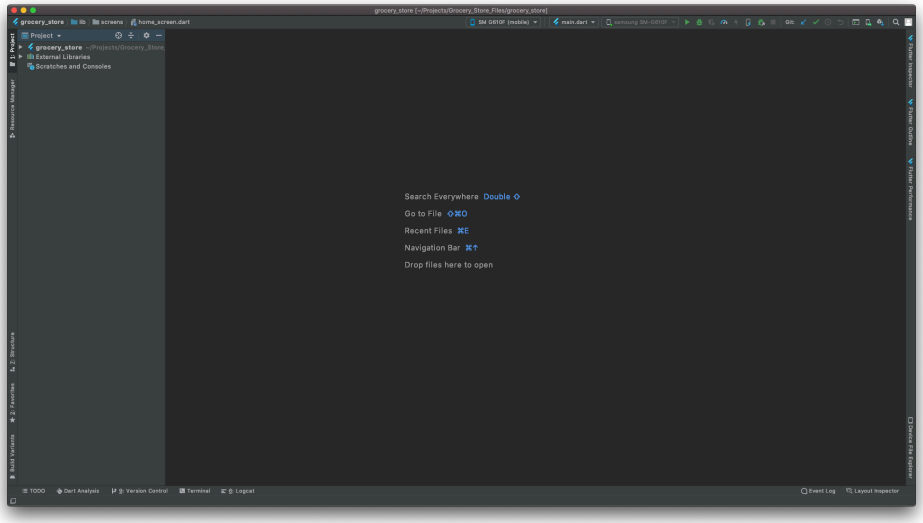
Now open **Android Studio** and click on **Open an existing Android Studio project** -> Navigate to the extracted folder and select **grocery_store** project folder as shown:



Now similarly open other two projects/folders as well in different **Android Studio** windows. Now when you open the projects for the first time Android Studio may ask to download some Gradle files so download them and proceed.

B2X_CODES

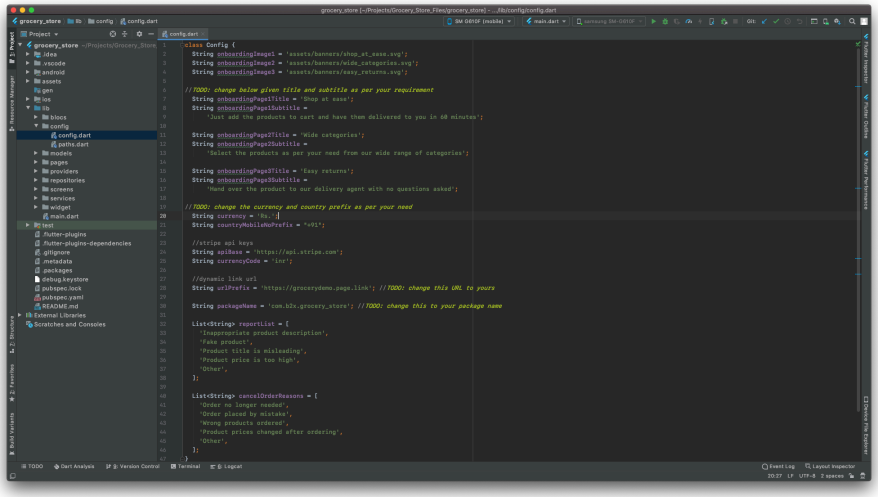
Once everything is up and running you will see a screen as shown:



Now click on the **grocery_store** shown in left bar and navigate to the **config.dart** file as shown:
In config.dart file you can configure your Grocery Store app as per your requirements. For exa. If you want to change your currency to $ then change the value as follows:
String currency = 'Rs.' —> **String currency = '$'**

Paste the link we copied earlier to notepad in **urlPrefix**.

Change the **packageName** to yours as follows and copy it to notepad:
com.b2x.grocery_store -> **com.yourCompanyName.grocery_store** (without any white spaces)

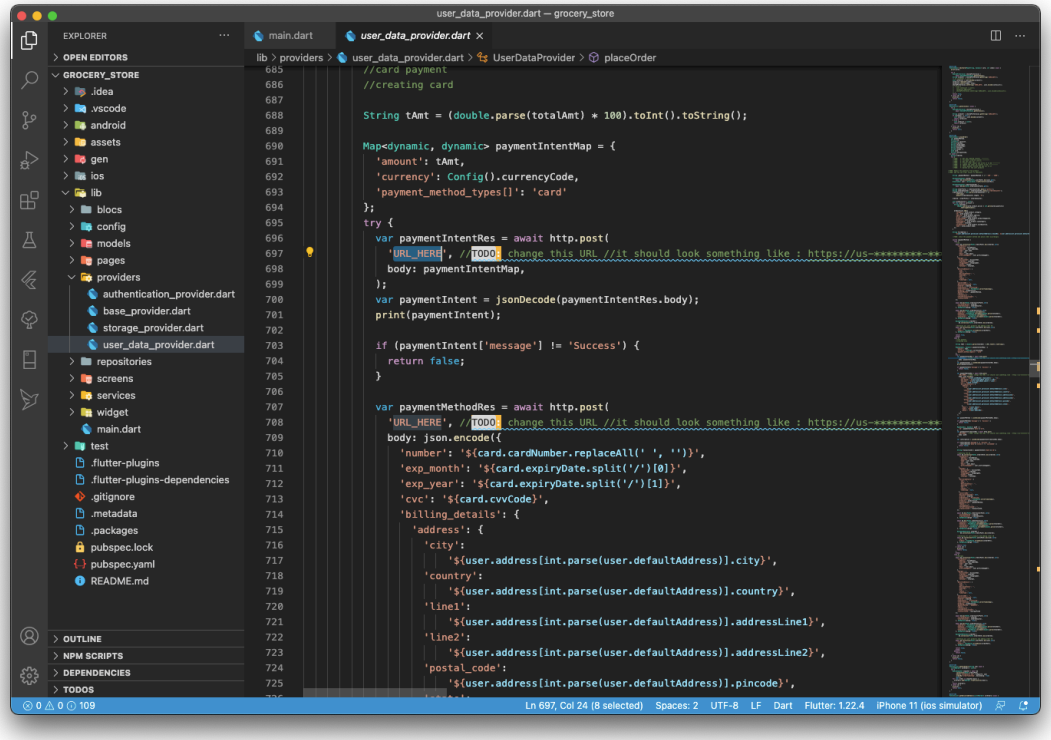


Similarly open the **config.dart** in other two apps as well and modify them as per the given instructions in those files.

Adding the URLs of Firebase Functions

Now we will add URLs of firebase functions that we deployed earlier in the setup to the apps codebase.
Open **Firebase Console** -> **Functions**
Here you will see all the functions deployed with their respective urls.

Follow the steps given below:
Open **grocery_store** app folder in **VSCode** or **Android Studio**
Open file: **lib** -> **providers** -> **user_data_provider.dart**
Now copy the respective URLs from firebase console to the codebase as shown below:
URLs that need to be replaced are on line no. 697, 708 & 746.

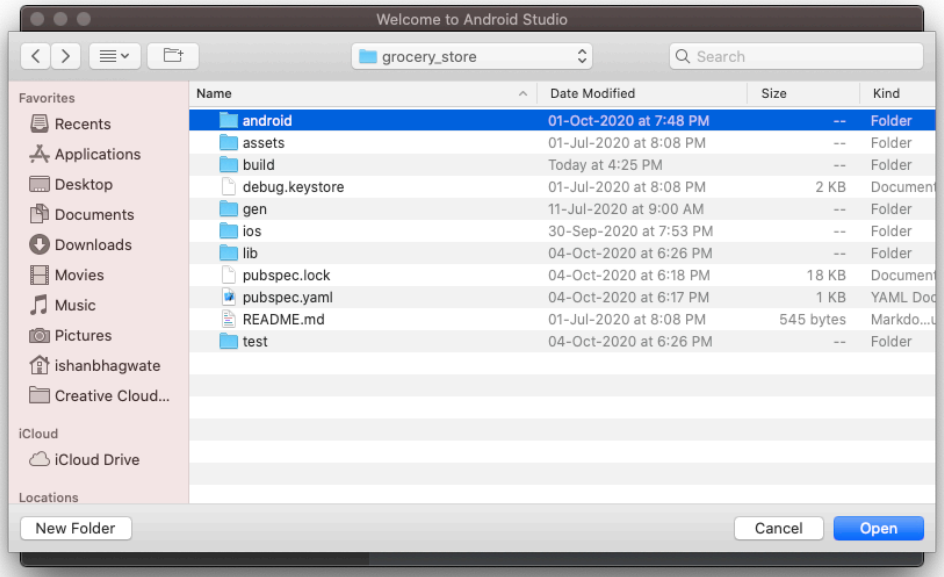


Follow the same process for **grocery_store_admin** app.
URLs that need to be replaced are on line no. 1399, 1662, 1701, 1850, 1881, 1940 & 2122.

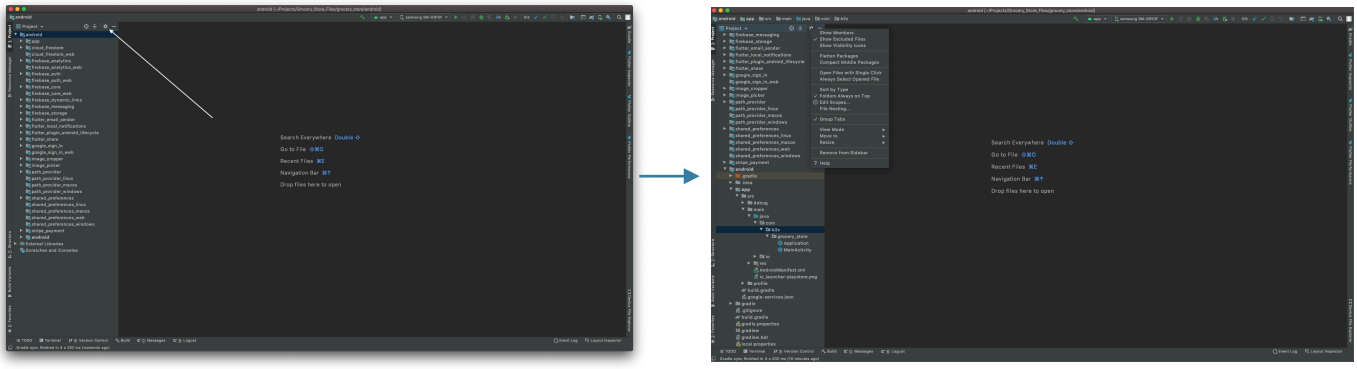
Changing the package name

1) Android:

Now in Android Studio click on **File** -> **Open** -> go to **grocery_store** folder, select **android** and click **Open** as shown:

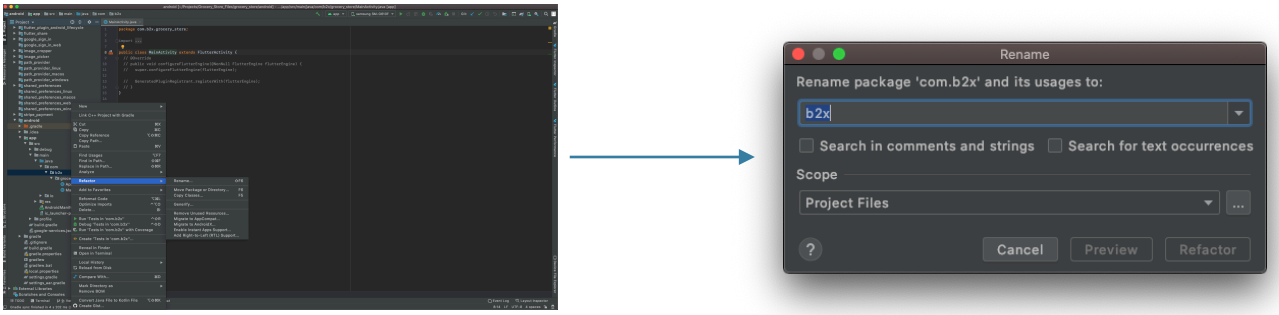


Click on the icon shown below and untick **Compact Middle Packages** as shown:



B2X_CODES

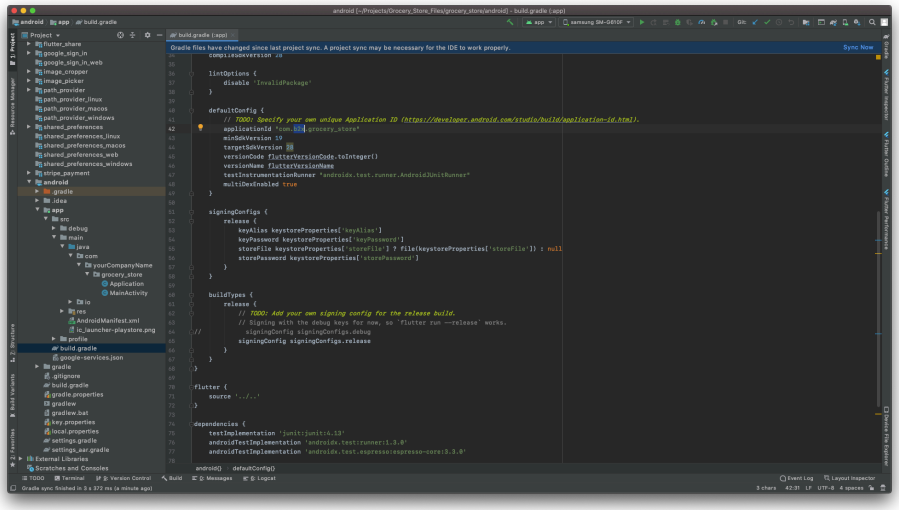
Now click on the **android** shown in left bar and navigate to the **MainActivity** file.
Right click on **b2x** -> **Refactor** -> **Rename** as shown:



Here change the package name to the name which we earlier copied to notepad.
It will look something like: **com.companyName.grocery_store**.

Copy **companyName** and paste in the above given window and click on **Refactor**.

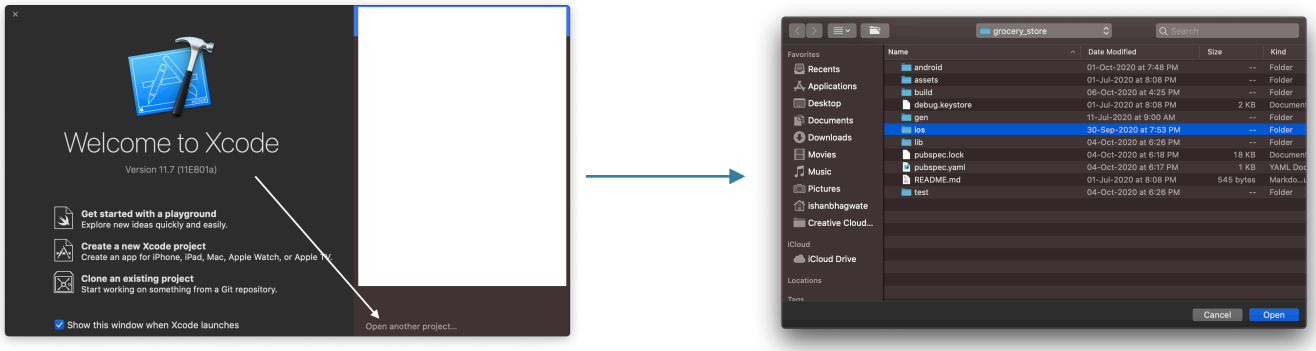
Go to **build.gradle** file as shown and change the **applicationId** to the above copied package name.



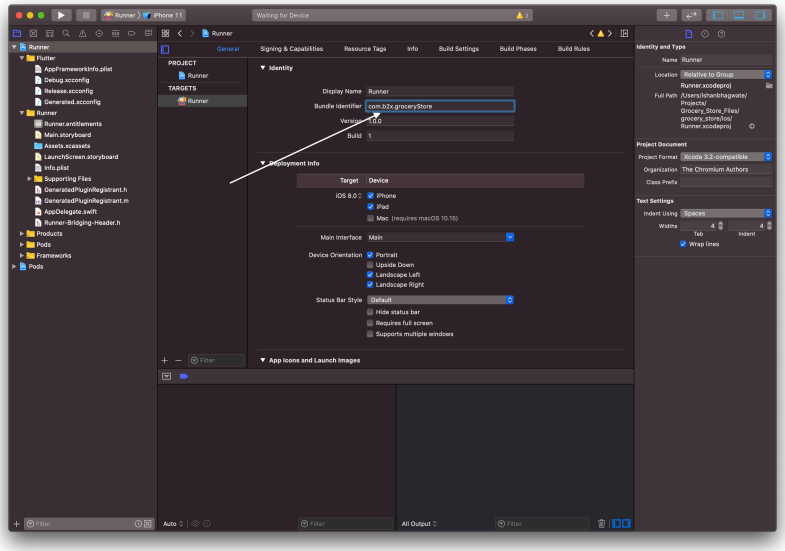
Now repeat this process of changing the package name for other two apps also.

2) iOS:

Open **Xcode** -> Click **Open Project** in right bottom corner -> Navigate to **grocery_store** folder -> Select **ios** folder and click **Open** as shown:



Now Double click on **Runner** and change the **Bundle identifier**.
Just replace **b2x** with **yourCompanyName**
It will look something like: **com.yourCompanyName.groceryStore** as shown:

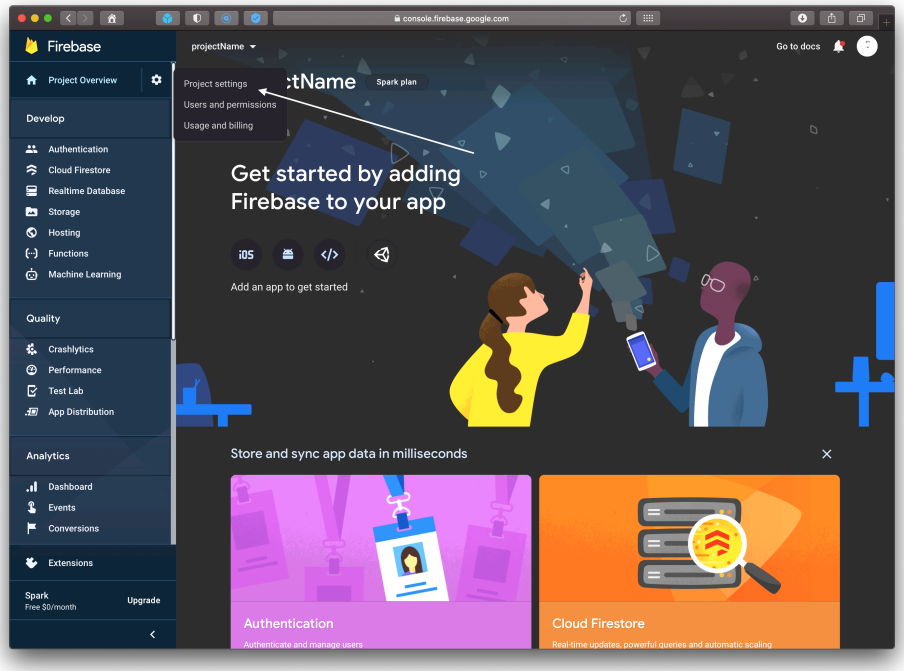


Copy the **bundle identifier** in a notepad.
Also please keep Android (**packageName**) and iOS (**bundle identifier)** separate as they are different.
Now repeat this process of changing the **bundle identifier** for other two apps as well.

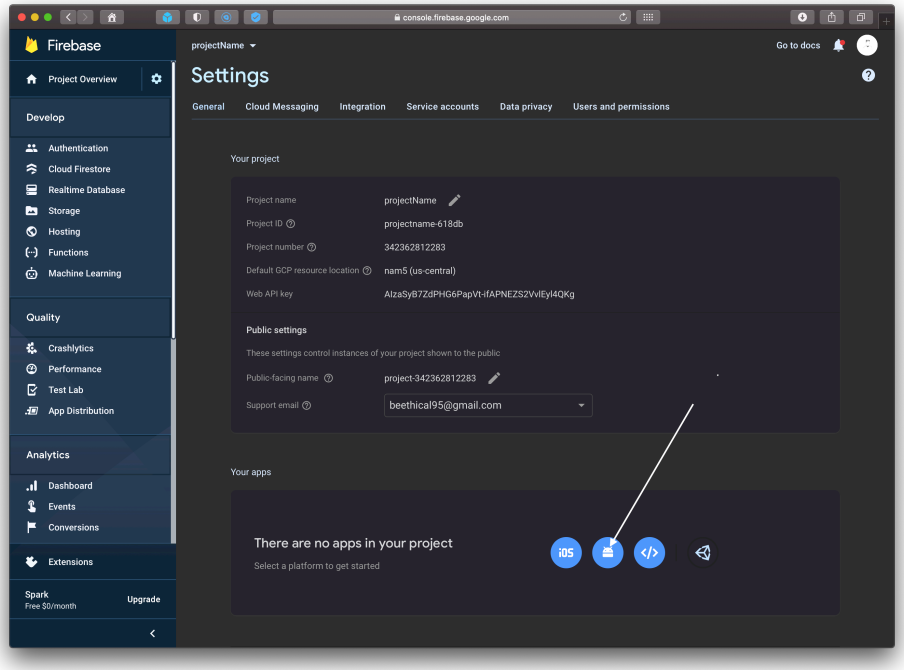
Connecting App to Firebase

1) Android

Open Firebase Console and click on **Setting icon** besides **Project Overview** -> **Project settings** as shown:



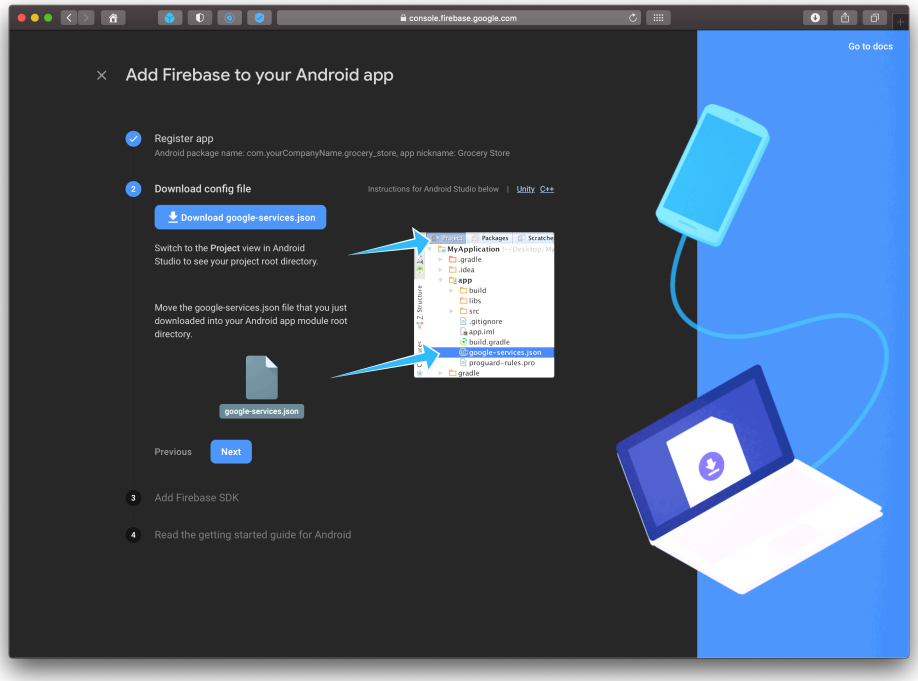
Now click on the **Android logo** as shown:



Now follow the on screen instructions as shown:

Here we will show you how to connect the Grocery Store app to firebase and same steps are to be performed for other two apps (i.e. Admin app & Delivery app) as well.

1) Paste the **packageName** that we copied earlier. (Paste the respective packageName i.e Grocery Store **packageName** for Grocery Store app and so on)
2) Type your app name eg: Grocery Store
3) Click on **Register app**

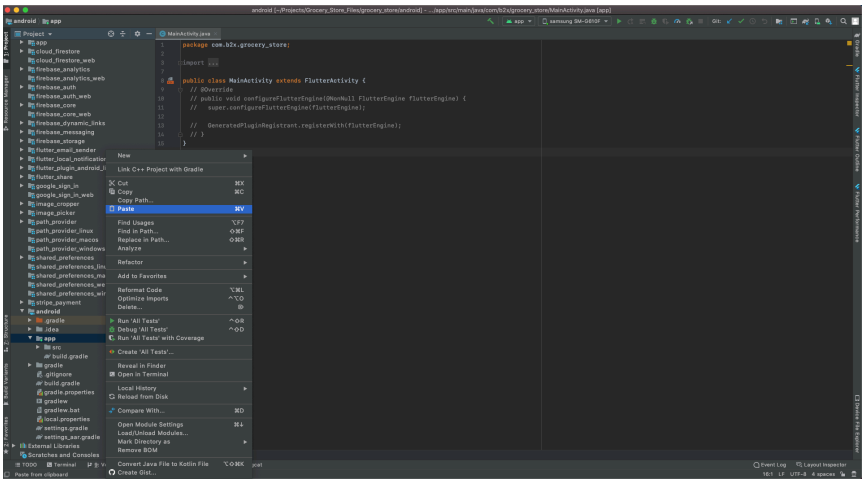


Click on **Download google-services.json -> Next**

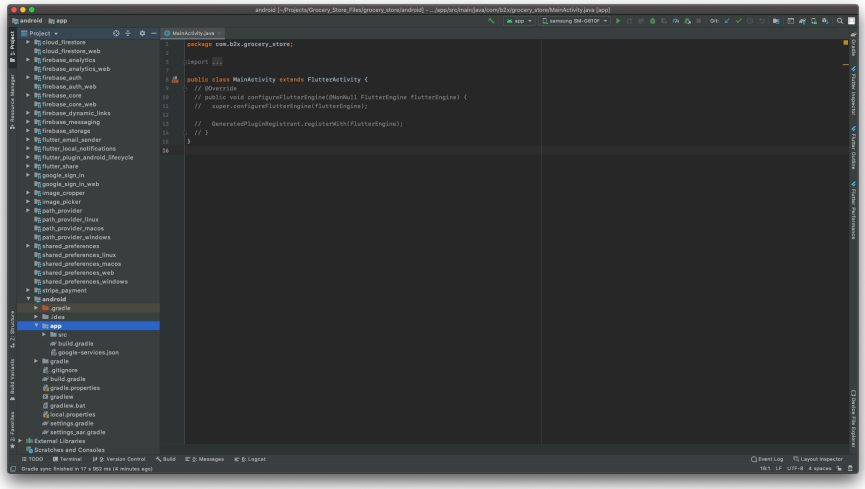
Open **Android Studio -> Click on Open -> Select the Project** if not opened already

B2X_CODES

Navigate to **app** folder as shown and paste the **google-services.json** file that we just downloaded by right-clicking on **app** folder -> **Paste**:



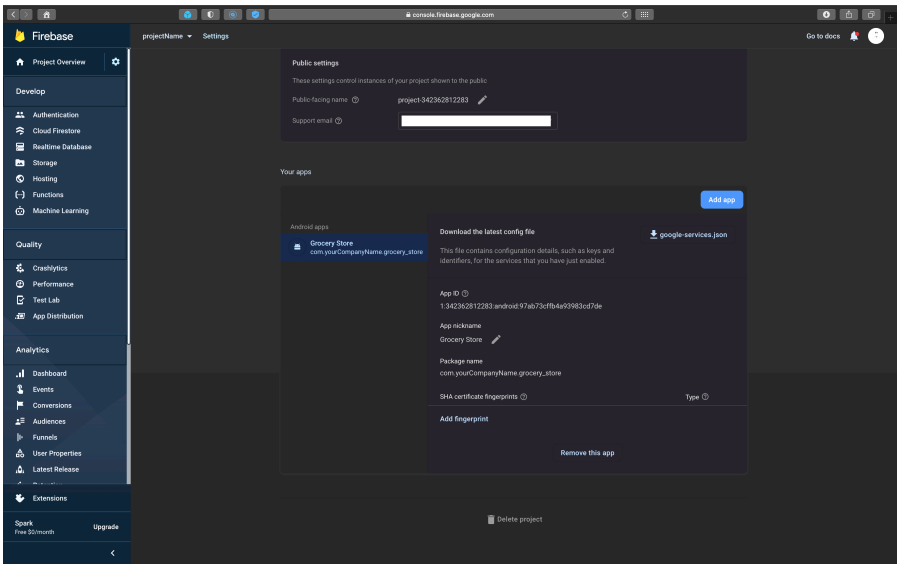
After pasting the file it should look something like this shown below:
Make sure to check if you have pasted the file in **app** folder and not in sub-folders otherwise it can cause issues later.



Now in the firebase console click **Next** -> **Continue to the console**

B2X_CODES

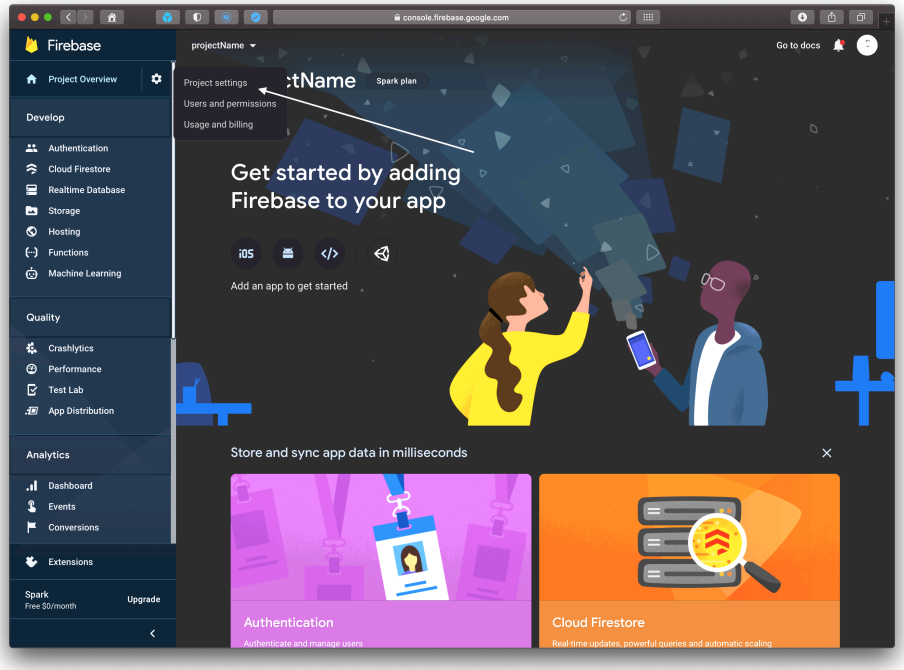
You can find the newly added app in the firebase console as shown:



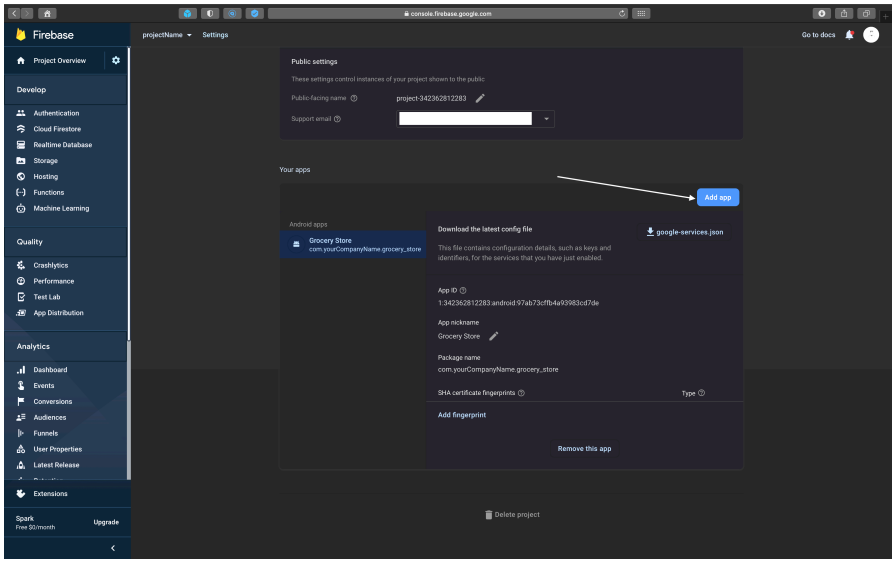
Similarly follow these same steps to add other 2 android apps as well.

2) iOS

Open Firebase Console and click on **Setting icon** besides **Project Overview** -> **Project settings** as shown:



Now click on the **Add app** button and select **iOS logo** as shown:

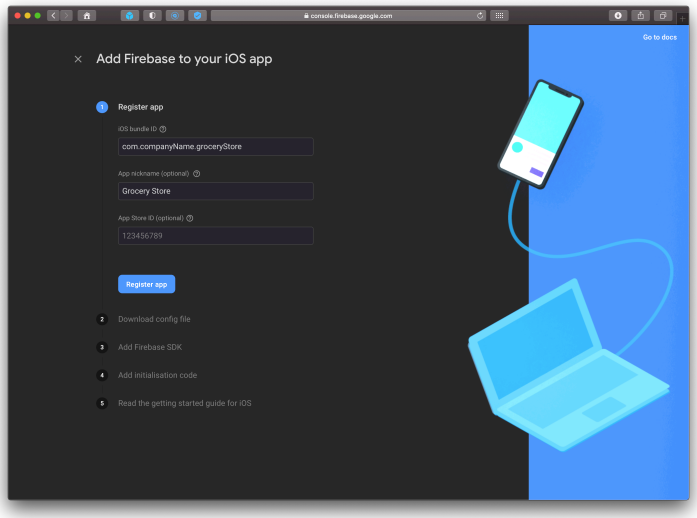


B2X_CODES

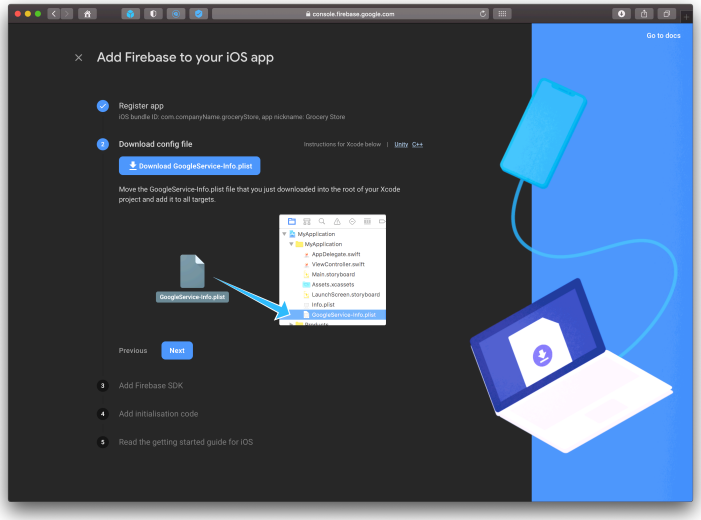
Now follow the on screen instructions as shown:

Here we will show you how to connect the Grocery Store app to firebase and same steps are to be performed for other two apps (i.e. Admin app & Delivery app) as well.

1) Paste the **bundle identifier** that we copied earlier. (Paste the respective **bundle identifier** i.e Grocery Store **bundle identifier** for Grocery Store app and so on)
2) Type your app name eg: Grocery Store
3) Click on **Register app**



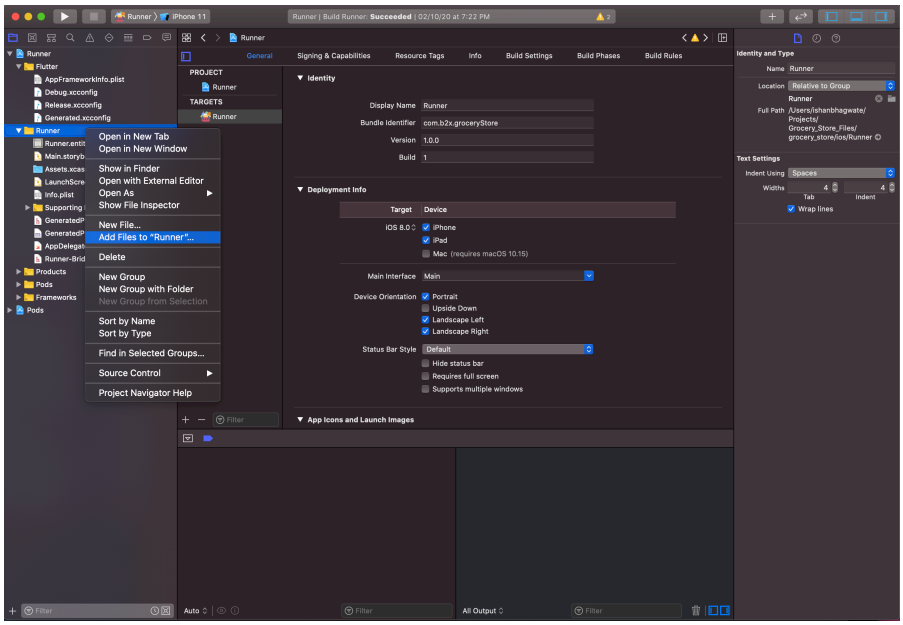
Click on **Download GoogleServiceInfo.plist -> Next**



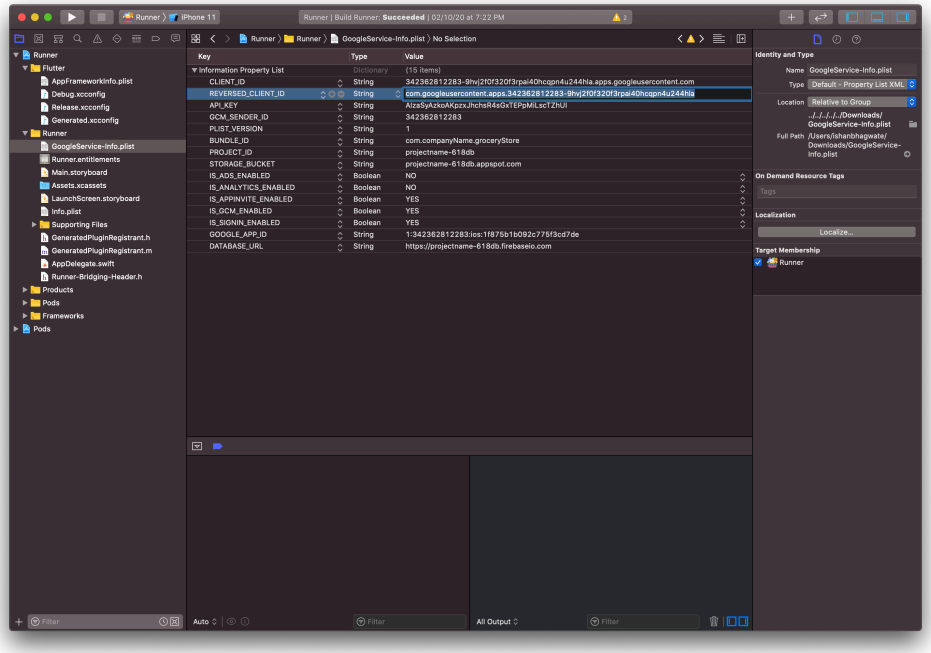
B2X_CODES

Now open **Xcode -> Click on Open -> Select the Project** if not opened already
Open **Xcode** -> Click **Open Project** in right bottom corner -> Navigate to **grocery_store (whichever app you are working with)** folder -> Select **ios** folder and click **Open** as shown:

In **Xcode** right-click on the inner **Runner** folder and click on **Add Files to "Runner"** as shown:
Now select the recently downloaded **GoogleServiceInfo.plist** file and click **Add.**

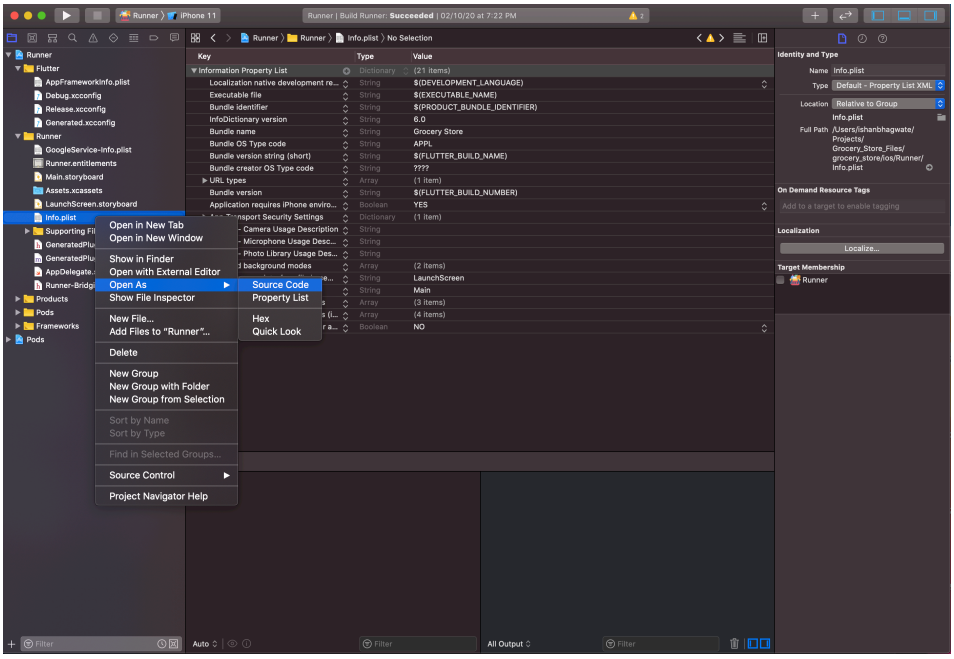


Double click on that file and copy **REVERSED_CLIENT_ID** value as shown:

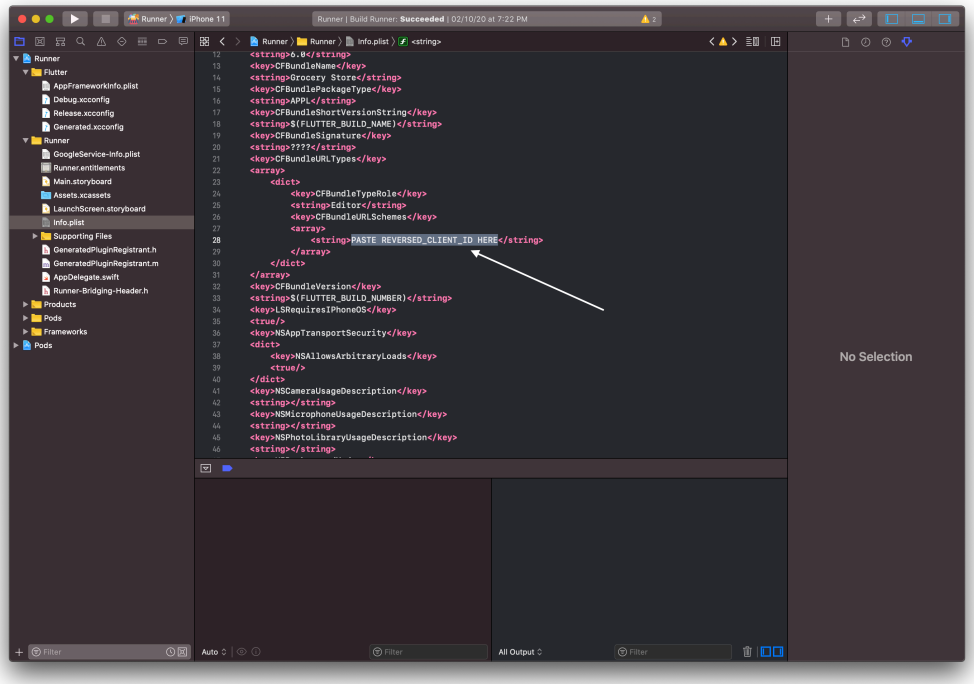


B2X_CODES

Now right click on **Info.plist file** and Select **Open As -> Source Code** as shown:



Paste the copied **REVERSED_CLIENT_ID** in the place of **PASTE REVERSED_CLIENT_ID HERE** as shown:



Similarly follow these same steps to add other 2 iOS apps as well.

Generating apps for release

1) Android

Here we will show how to create a release **.apk** file for **Grocery Store** app and same has to be done for other two apps as well.

Open **Android Studio** -> Click **Open existing Android Studio Project** -> Select **grocery_store** folder -> Click **Open** as shown:

Click on **Terminal** given in bottom bar as shown:
Now follow the instructions given on this https://flutter.dev/docs/deployment/android#create-a-keystore or shown in the below image to create a keystore.
Run the command as per your OS and follow the instructions as given.

Create a keystore

If you have an existing keystore, skip to the next step. If not, create one by running the following at the command line:

On Mac/Linux, use the following command:

```
keytool -genkey -v -keystore ~/key.jks -keyalg RSA -keysize 2048 -validity 10000 -alias key
```

On Windows, use the following command:

```
keytool -genkey -v -keystore c:\Users\USER_NAME\key.jks -storetype JKS -keyalg RSA -keysize
 2048 -validity 10000 -alias key
```

This command stores the `key.jks` file in your home directory. If you want to store it elsewhere, change the argument you pass to the `-keystore` parameter. **However, keep the `keystore` file private; don't check it into public source control!**

Note:

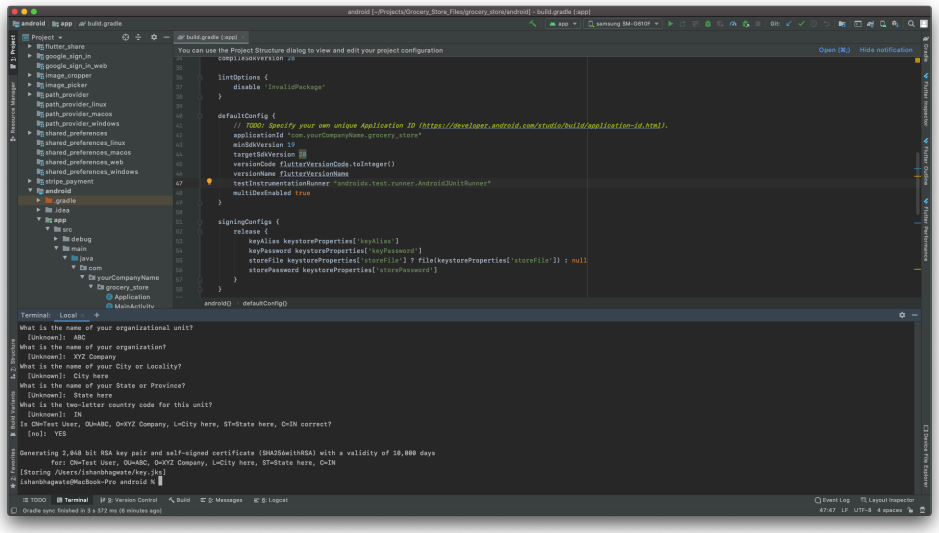
- The `keytool` command might not be in your path—it's part of Java, which is installed as part of Android Studio. For the concrete path, run `flutter doctor -v` and locate the path printed after 'Java binary at:'. Then use that fully qualified path replacing `java` (at the end) with `keytool`. If your 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.

1) Type the password
2) Retype the password
3) Fill the details as asked

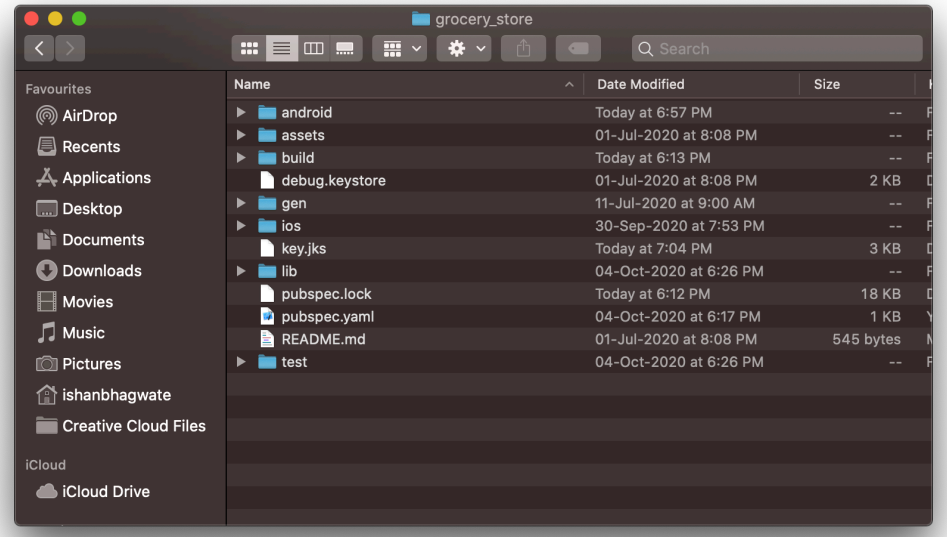
NOTE: Please keep these details safe and do not share or forget the password used here.

B2X_CODES

Now once you have filled all the details type **YES** and click **Enter**

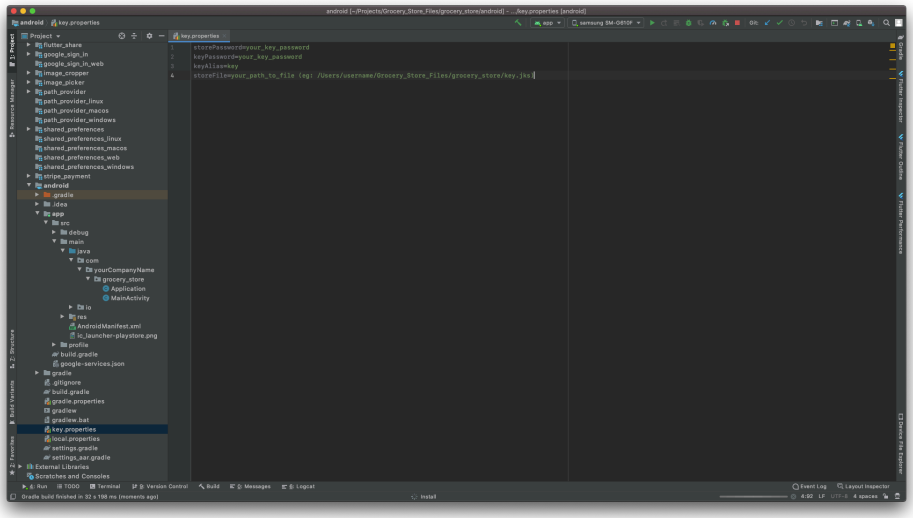


Now **cut** the file (i.e: **key.jks**) and **paste** in your particular project folder as shown:



B2X_CODES

In **Android Studio** open **key.properties** file and edit the values as per your inputted values as shown

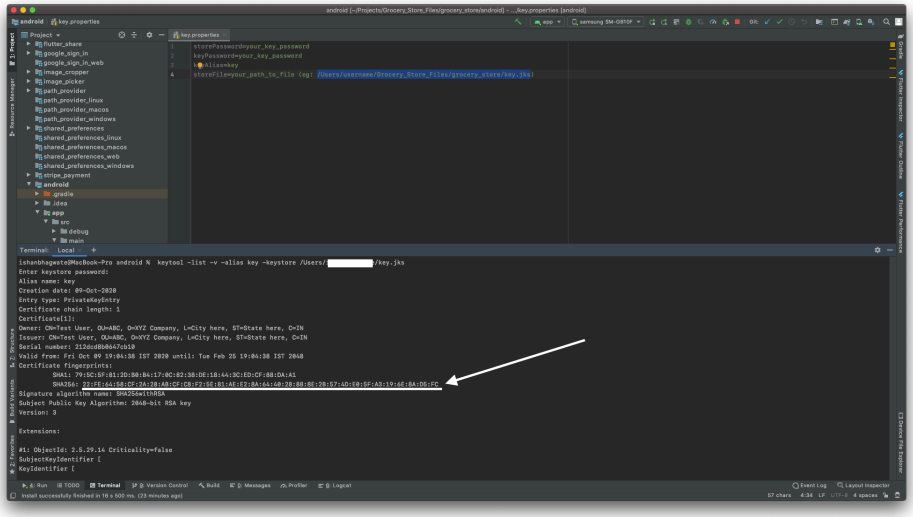


Open **Terminal** and type the following command to get the **SHA-256 fingerprint**:

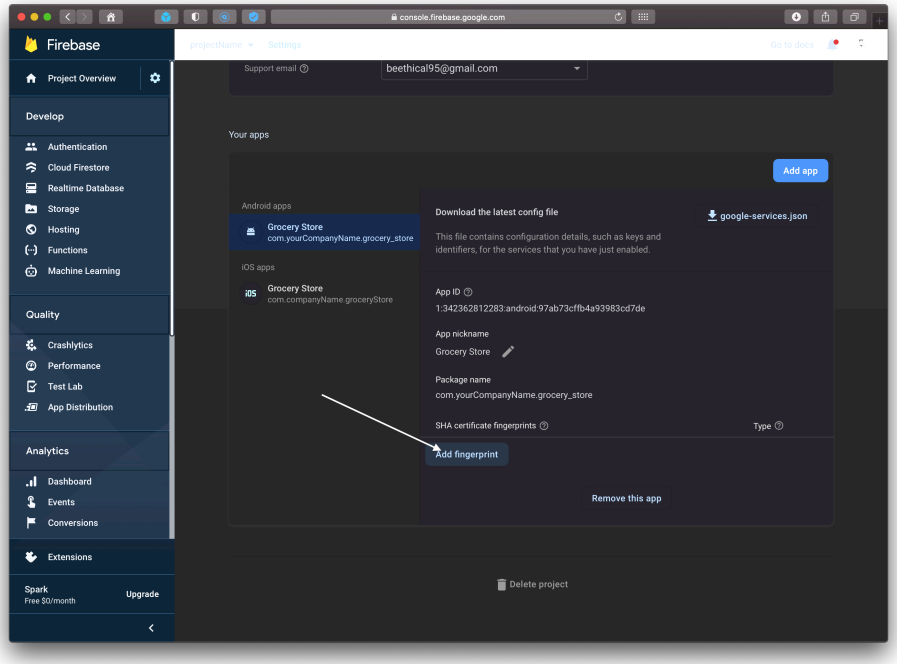
—> **keytool -list -v -alias key -keystore <path-to-production-keystore>**

For eg: keytool -list -v -alias key -keystore /Users/username/Grocery_Store_Files/grocery_store/key.jks

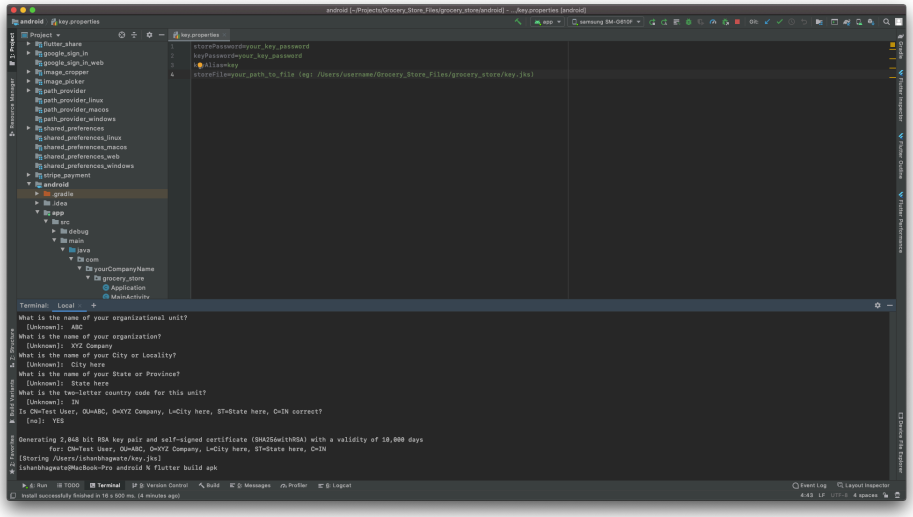
Now input the keystore password and copy the **SHA-256** fingerprint as shown:



Open **Firebase console** -> **Project Settings** -> Select the particular app -> Click **Add fingerprint** -> Paste the copied **SHA-256** fingerprint and click **Add**



Open **Terminal** and type the following command to generate the release **.apk** file as shown:
—> **flutter build apk —release**



Now repeat this process for generating **.apk** files for other two apps also.

2) iOS

Here we will show how to create a release **.ipa** file for **Grocery Store** app and same has to be done for other two apps as well.
NOTE: To generate .ipa file you will need to have an Apple developer account.
Follow the below given steps:

Step 1: *Change scheme destination to* `Generic IOS device`.

Step 2: *Click* `Product` > `Archive` > *once this is complete open up the Organiser and click the latest version.*

Step 3: *Click on* `Export...` *option from right side of organiser window.*

Step 4: *Select a method for export > Choose correct signing > Save to Destination.*

> ***Xcode 10.0***

Step 3: *From Right Side Panel Click on Distribute App.*

Step 4: *Select Method of distribution and click next.*

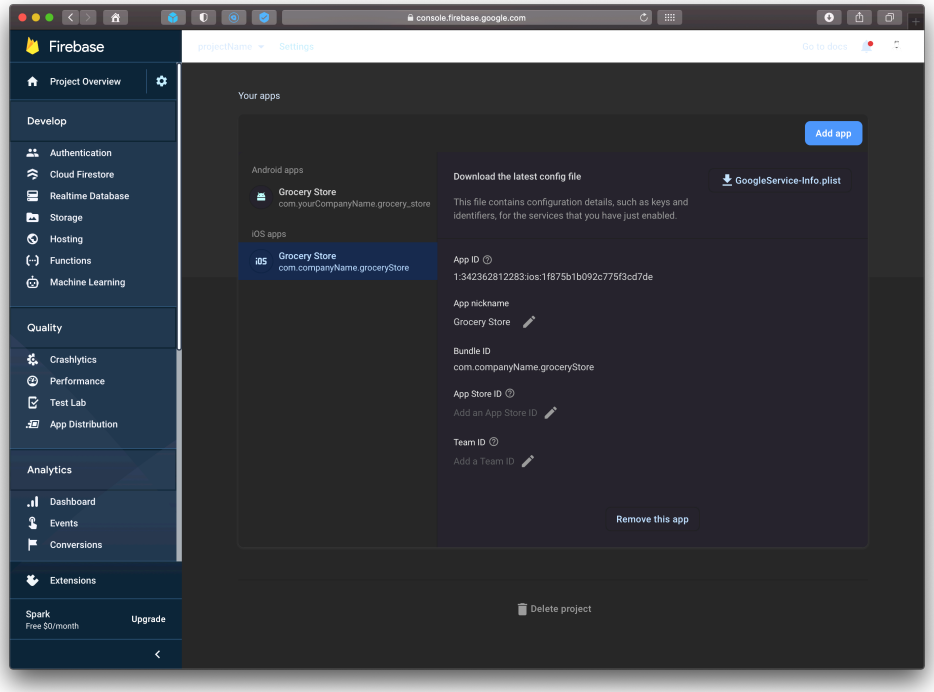
Step 5: *It Opens up distribution option window. Select* ***All compatible device variants*** and *click next.*

Step 6: *Choose signing certificate.*

Step 7: *It will open up Preparing archive for distribution window. it takes few min.*

Step 8: *It will open up Archives window. Click on export and save it.*

Open **Firebase console** -> **Project Settings** -> Select the particular app
Type in your **App Store ID** and **Team ID**

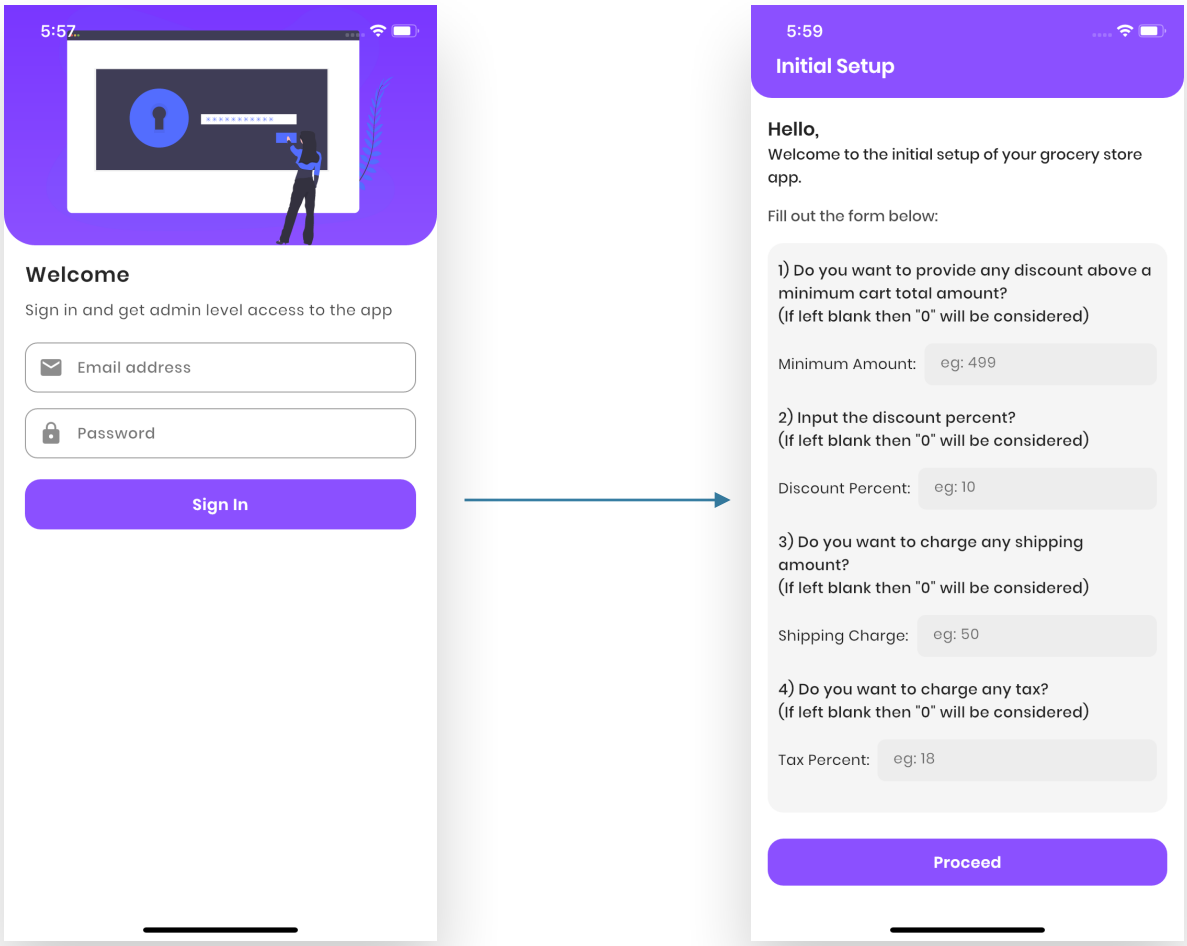


Now repeat this process for generating **.ipa** files for other two apps also.

App Setup

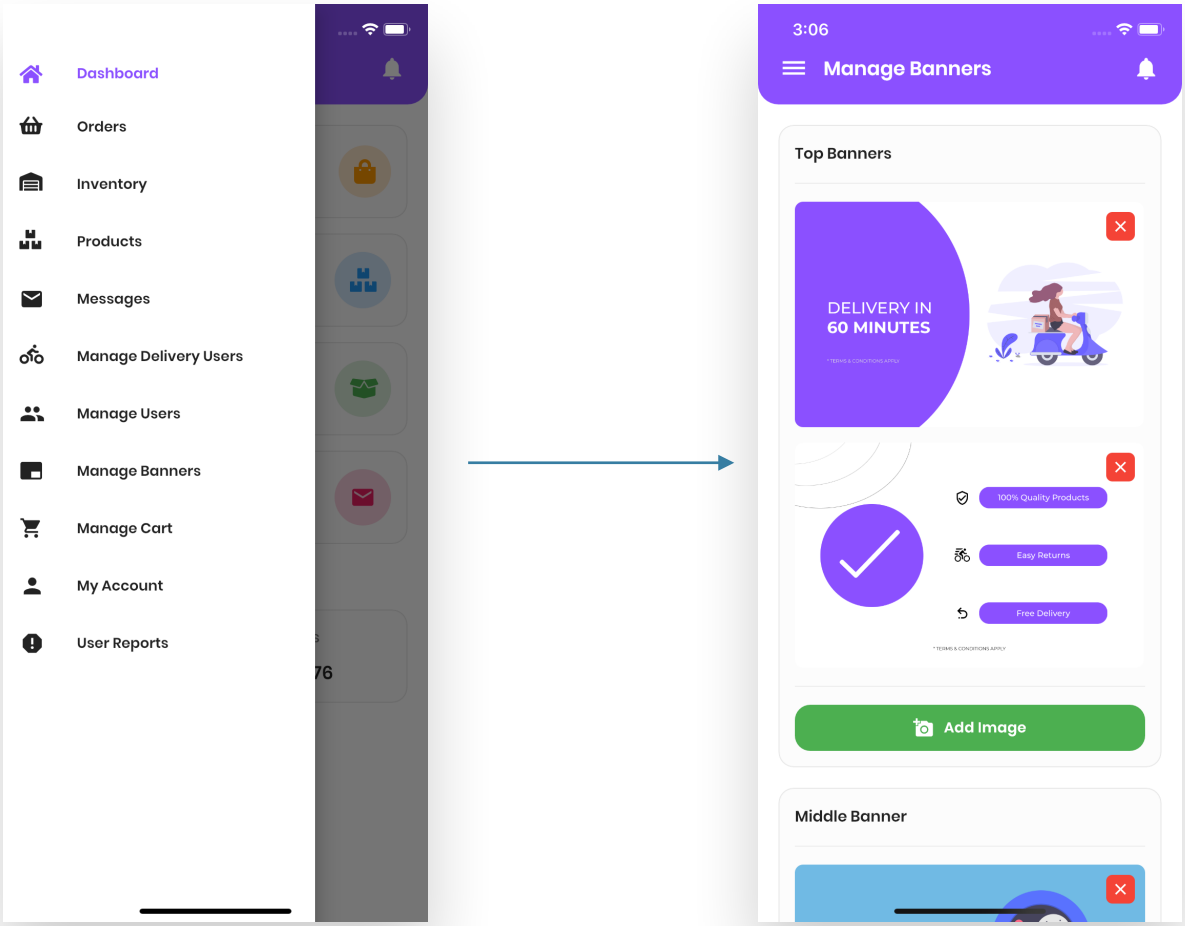
This will help you setup and configure the app with step by step instructions.

Open the **Grocery Store Admin** app and sign in with email and password you created earlier, after that you will be taken to initial setup screen. Fill out all the details and click **Proceed** as shown:



Now open **Inventory** and add **Categories.** If you don't add categories and go to **Banners** section it will not work as **Banners** need categories to be added first.

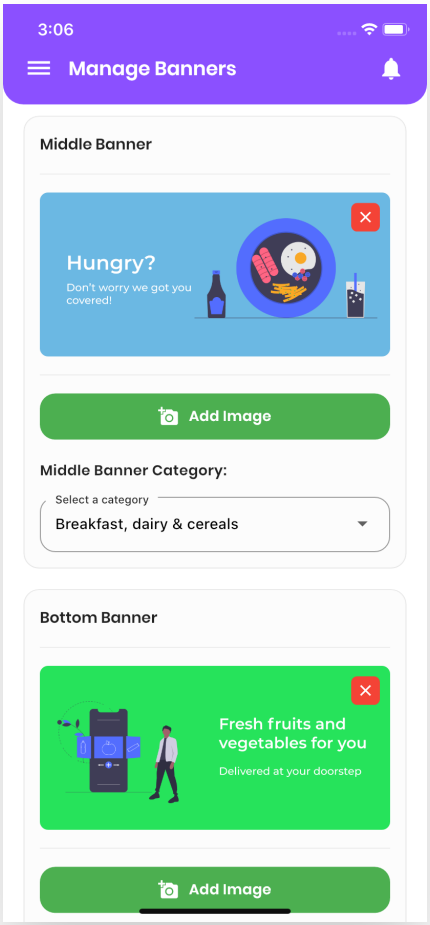
Once you have added categories open **Manage Banners** and set up all the required banners as shown:
Please don't run the Grocery Store app till setting up everything mentioned below.

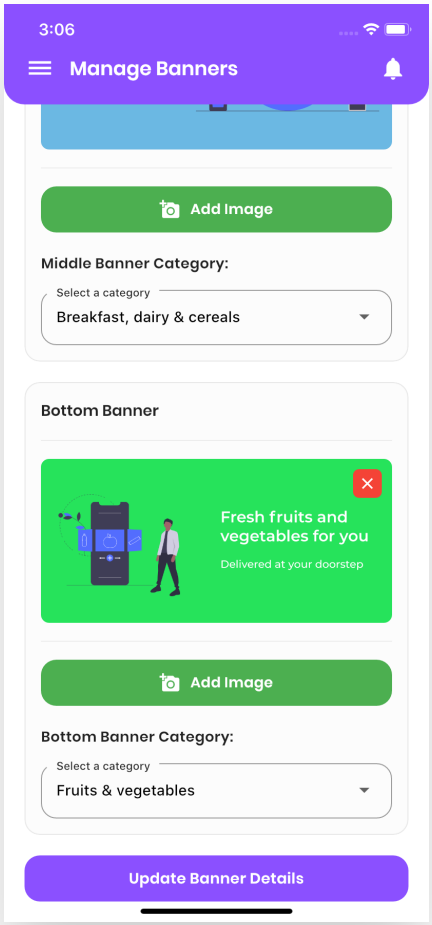


Add all the banners as shown in the screenshots, all the banner images can be found in **assets** folder provided to you.

Remember to add at-least one **Top Banner** image. Also adding **bottom banner** and **middle banner** is mandatory.

Bottom banner and **middle banner** requires category so make sure to add them as well and then click on **Update Banner Details**.





Use the images provided to you and setup the banners and once this is setup you can now run **Grocery Store** app.

If you need any assistance in setting up the platform kindly drop us a mail at b2xcodes@gmail.com.