

IONIC 3 FIREBASE 3 SHOPPING CART APP

Introduction

Ionic 3 FireBase 3 Shopping cart app is eCommerce mobile app for managing products and selling products online integrated with PayPal and Stripe payment systems. This app is integrated with firebase and there is no requirement of hosting server. This app has two parts of admin and customer that can be separated to separate apps of admin and customers

[You can also read online updated documents](#)

Admin Features:

1. Add and manage categories
2. Add and manage products
3. Add and manage customers
4. Add and manage vendors
5. Add and manage brands
6. Add and manage images
7. Create and manage orders
8. Configure and Manage PayPal payment method
9. Configure and Manage Stripe payment method
10. Manage Banners and Currency

Customer Features:

1. Browse categories and products
2. Create account
3. Login with Google Authentication, Facebook Authentication, Twitter Authentication, FireBase Authentication
4. Add products to cart
5. Manage cart
6. Checkout

7. Make payment
8. View and edit profile

Installation

Summary

1. Setup ionic environment
2. Create an ionic App
3. Install Firebase
4. Connect your App and Firebase
5. Firebase Authentication

1. Setup ionic environment

Skip this if you have already setup ionic environment

Before we begin we need a recent version of Node.js. [Download the installer](#) for Node.js 6 or greater

We need to install the latest version of the CLI and Cordova. Install the Ionic CLI and Cordova for native app development

```
$ npm install -g ionic cordova
```

2. Create an ionic App

```
$ ionic start myApp
```

```
$ cd myApp
```

Replace myApp/src folder with downloaded src folder. Better delete src folder in myApp/ directory. Copy and paste downloaded src folder in its place myApp/ directory

3. Install Plugins:

Social Sharing:

```
ionic cordova plugin add cordova-plugin-x-socialsharing  
npm install --save @ionic-native/social-sharing
```

Native Storage:

```
ionic cordova plugin add cordova-plugin-nativestorage  
npm install --save @ionic-native/native-storage
```

Paypal:

```
ionic cordova plugin add com.paypal.cordova.mobilesdk  
npm install --save @ionic-native/paypal
```

Stripe:

```
ionic cordova plugin add cordova-plugin-stripe  
npm install --save @ionic-native/stripe
```

Translate:

```
npm install @ngx-translate/core @ngx-translate/http-loader --save
```

3. Install Firebase

```
npm install firebase --save  
npm install promise-polyfill --save-exact
```

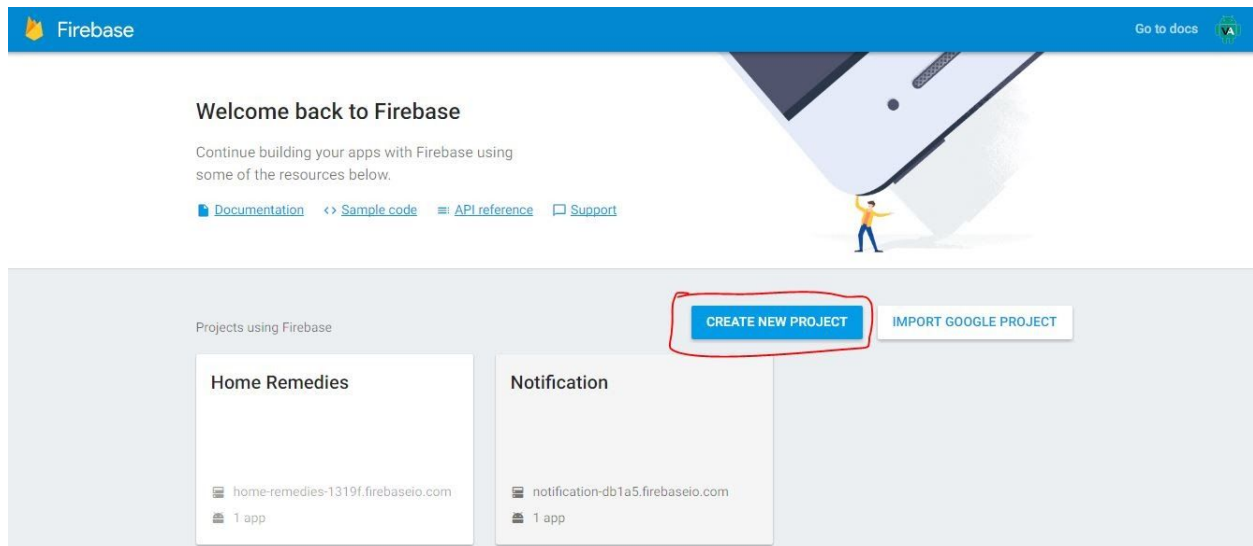
4. Connect your App and Firebase

Go to `app.component.ts` and initialize Firebase, for that you need following config data from firebase console:

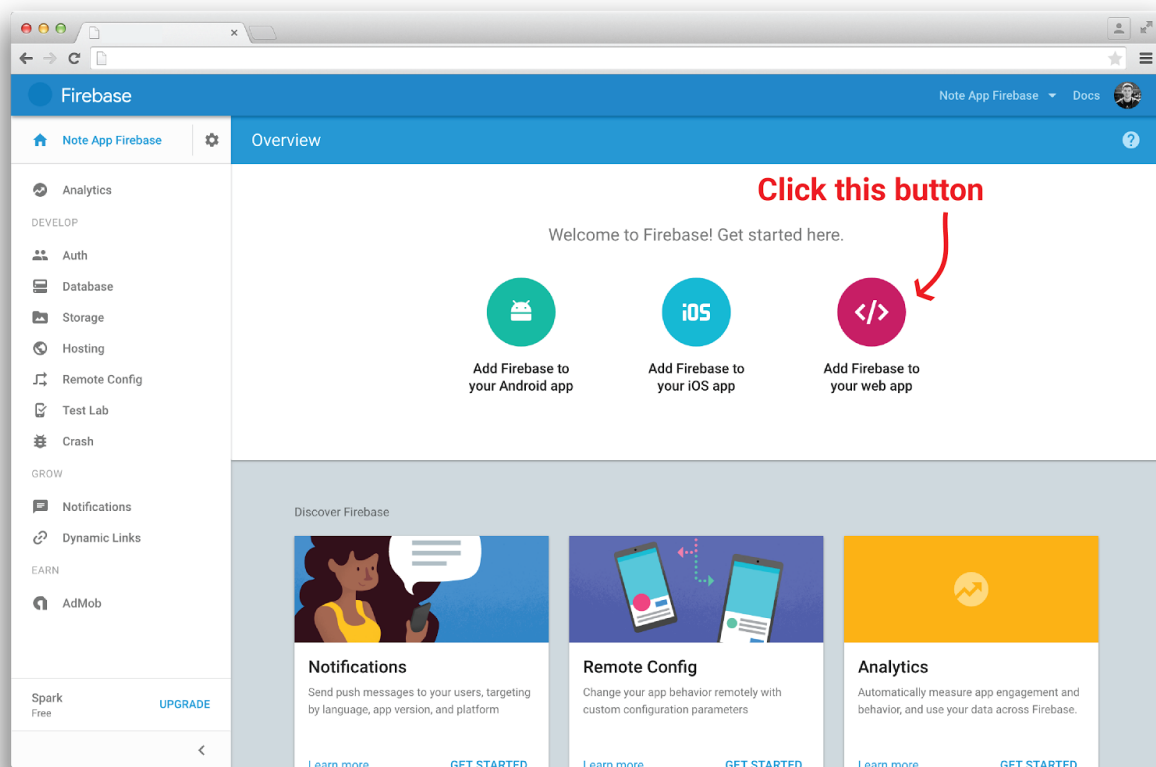
```
apiKey: "<API_KEY>",
authDomain: "<PROJECT_ID>.firebaseapp.com",
databaseURL: "https://<DATABASE_NAME>.firebaseio.com",
storageBucket: "<BUCKET>.appspot.com",
messagingSenderId: "<SENDER_ID>",
```

You can get that data inside your [Firebase console](#):

1. We need to create firebase project in the [Firebase console](#). If you already have a Firebase project, click Add App from the project overview page.



2. Click Add Firebase to your web app.



3. Note the initialization config variables, which you will use in a minute.

Add Firebase to your web app



Copy and paste the snippet below at the bottom of your HTML, before other `script` tags.

```
<script src="https://www.gstatic.com/firebasejs/3.6.3/firebase.js"></script>
<script>
  // Initialize Firebase
  var config = {
    apiKey: "F8pv2IHW8Gc",
    authDomain: ".firebaseapp.com",
    databaseURL: "https://.firebaseio.com",
    storageBucket: ".appspot.com",
    messagingSenderId: ""
  };
  firebase.initializeApp(config);
</script>
```

COPY

Check these resources to
learn more about Firebase for
web apps:

[Get Started with Firebase for Web Apps](#)

[Firebase Web SDK API Reference](#)

[Firebase Web Samples](#)

3. Copy and paste the config variables from the snippet to inside the constructor in `app.component.ts` file

```
firebase.initializeApp({

  apiKey: "",

  authDomain: "",

  databaseURL: "",

  storageBucket: "",

  messagingSenderId: ""

});
```

4. Firebase Authentication

We have used 4 authentication methods in our app.

1. **Google Login**
2. **Facebook Login**
3. **Twitter Login**
4. **Firebase Login**

1. Google Authentication:

Step #1: Get your credentials from Google.

Step #2: Install the Google native plugin.

Step #3: Login using Google and Firebase.

Step #1: Get your credentials from Google.

For IOS:

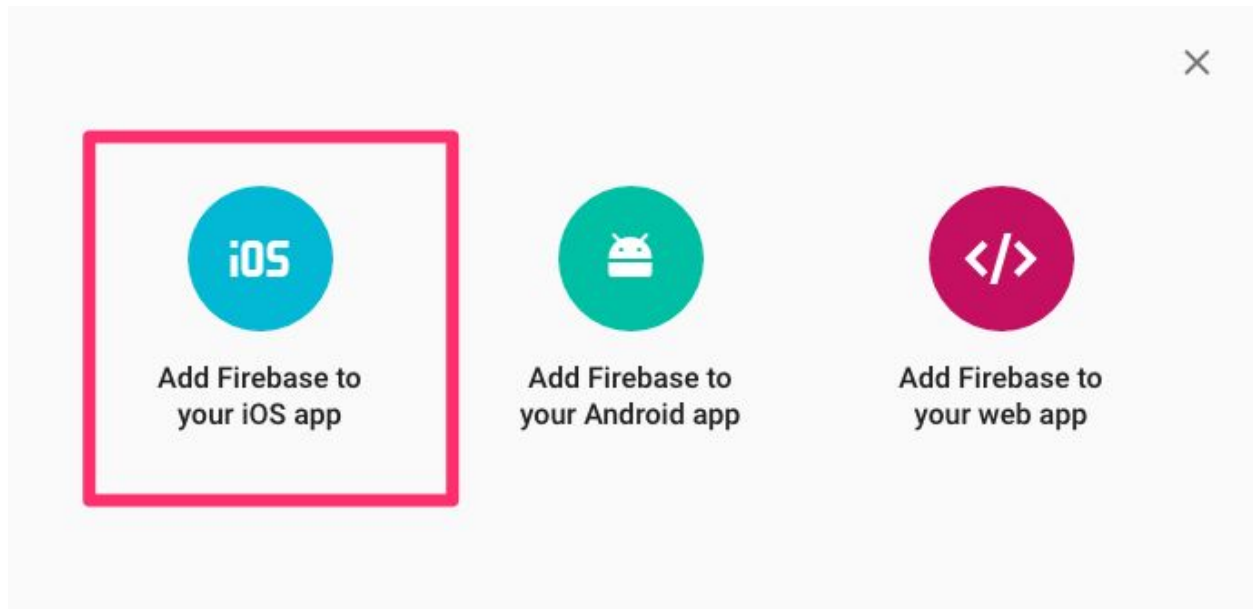
We need to create both ios and android app in firebase console.

let's start with the iOS one,

Go to console.firebase.google.com.

Choose your project.

Inside your project Add ios app.



It will ask you an iOS Bundle ID, to get it, go into your app's config.xml file and copy the package id you're using, it's the one that looks like (id="com.ionicframework.something").

Click REGISTER APP. It will generate GoogleService-Info.plist file

Click on Download GoogleService-Info.plist file, which is a config file for iOS.

After downloading GoogleService-Info.plist file click on Continue > Continue > FINISH

Copy GoogleService-Info.plist file to your project root directory

For Android:

Once we add iOS app, we need to add Android app.

Click ADD APP

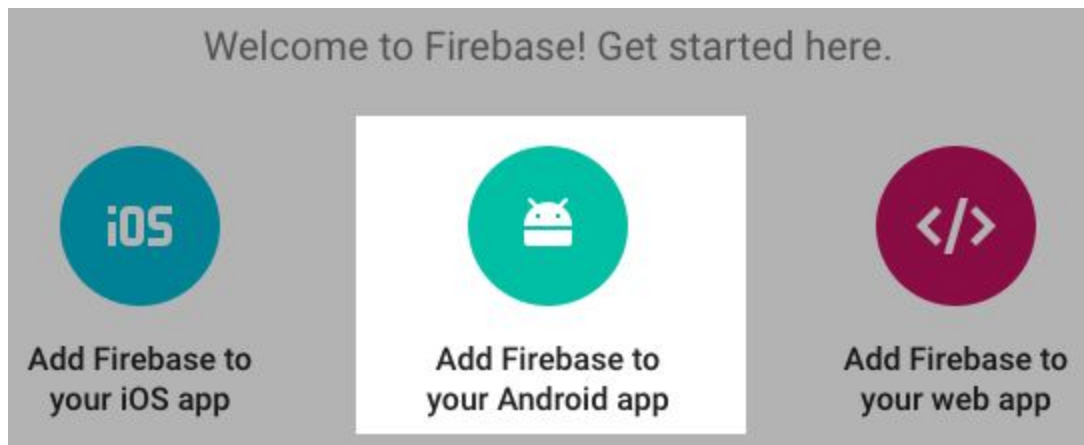
It will ask you an Android Bundle ID, to get it, go into your app's config.xml file and copy the package id you're using, it's the one that looks like (id="com.ionicframework.something").

Click REGISTER APP. It will generate google-services .json file

Click on Download google-services .json file, which is a config file for Android.

After downloading google-services .json file click on Continue > FINISH

Copy google-services .json file to your project root directory



Step #2: Install the Googleplus Cordova Plugin.

Note: We will get **CLIENT_ID** in google-services .json file

Run following command to install Googleplus cordova plugin.

```
$ ionic cordova plugin add cordova-plugin-googleplus --variable  
REVERSED_CLIENT_ID=CLIENT_ID  
  
$ npm install --save @ionic-native/google-plus  
  
$ cordova prepare.
```

Step #3: Ionic Google Login with Firebase.

Add client id in login.ts file ('<Your web client ID>').

```
loginUser(): void {  
    this.googlePlus.login({
```

```

    'webClientId': '<Your web client ID>',

    'offline': true

  }).then( res => {
firebase.auth().signInWithCredential(firebase.auth.GoogleAuthProvider.credential(res.idToken))

    .then( success => {

        console.log("Firebase success: " + JSON.stringify(success));

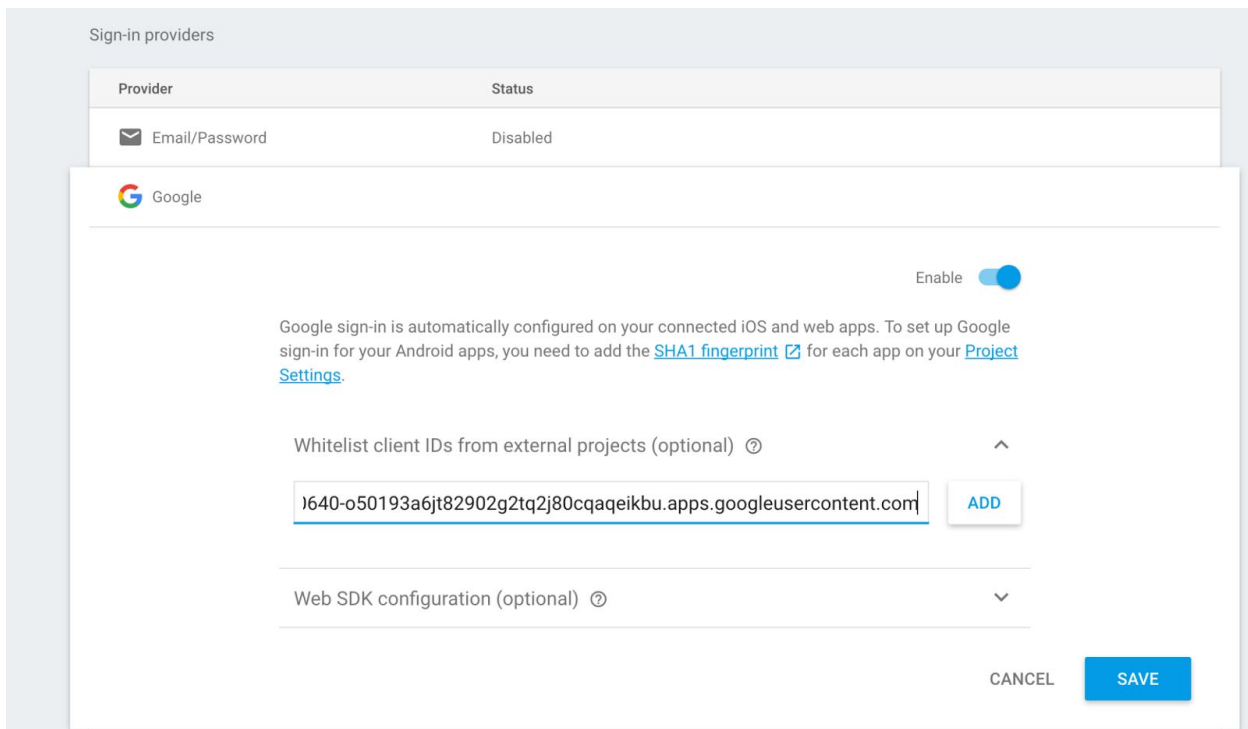
    }).catch( error => console.log("Firebase failure: " + JSON.stringify(error)));

  }).catch(err => console.error("Error: ", err)); }

```

Go to Firebase console-

Add client id here.



Sign-in providers

Provider	Status
Email/Password	Disabled

Google

Enable ☒

Google sign-in is automatically configured on your connected iOS and web apps. To set up Google sign-in for your Android apps, you need to add the [SHA1 fingerprint](#) for each app on your [Project Settings](#).

Whitelist client IDs from external projects (optional) [?] ^

Web SDK configuration (optional) [?] v

CANCEL

Facebook Authentication:

Create a Facebook App

We need to register for a Facebook application, Go to <https://developers.facebook.com>

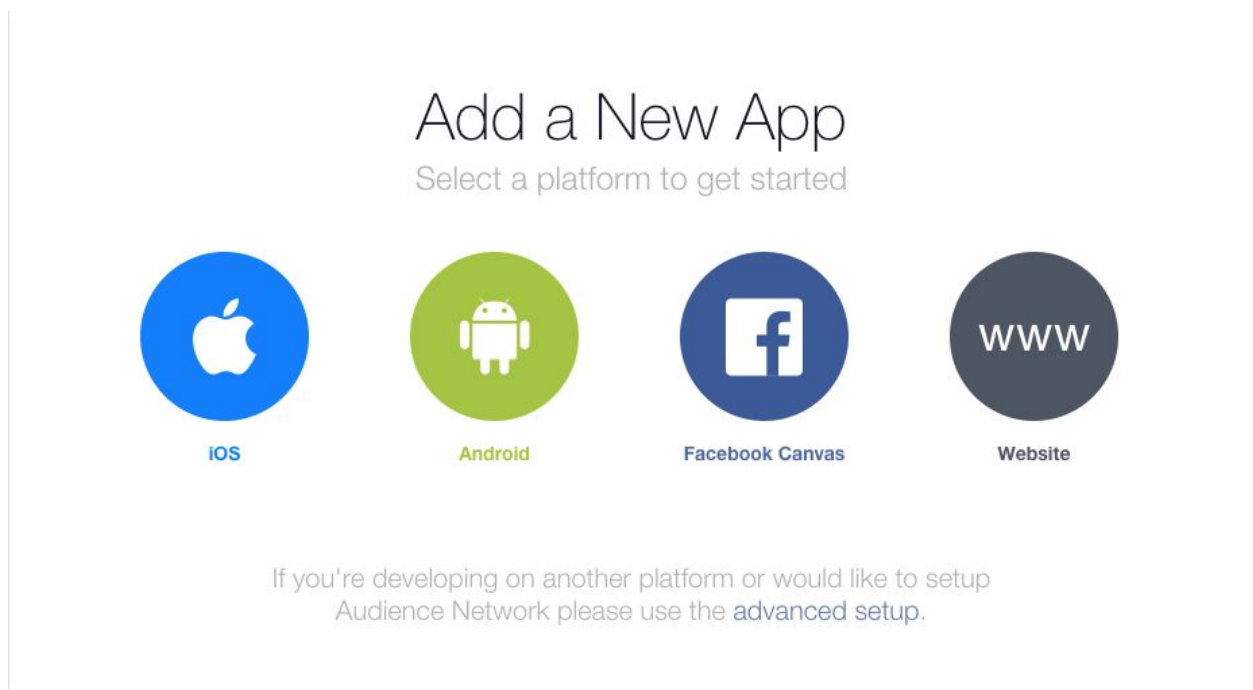
Create a Facebook for Developers account

click on the button Create a New App

It will ask which kind of app you want to create

we just need the ID for the plugin.

You will name your app and generate an ID



Facebook Create new App

The screenshot shows the 'Create a New App ID' form on the Facebook Developers website. The form is titled 'Create a New App ID' with the subtitle 'Get started integrating Facebook into your app or website'. It contains three input fields: 'Display Name' with the value 'La mia app', 'Namespace' with the placeholder 'A unique identifier for your app (optional)', and 'Categoria' with a dropdown menu showing 'Scegli una categoria'. At the bottom, there is a checkbox for 'By proceeding, you agree to the Facebook Platform Policies' and two buttons: 'Annulla' and 'Create App ID'.

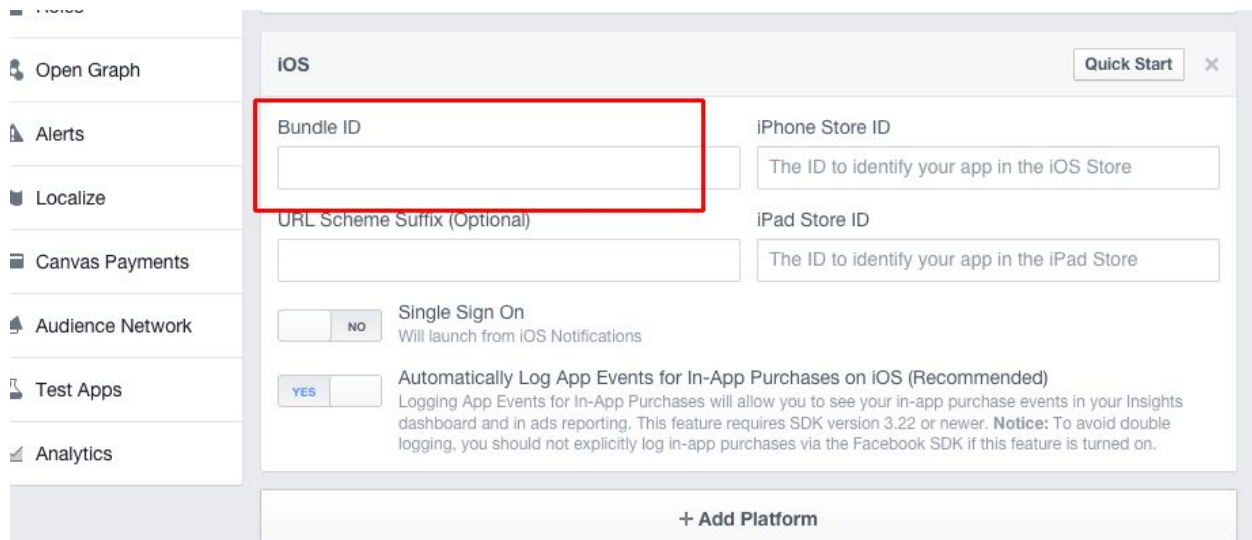
Facebook App Id.

Here you will get AppId and Appsecret

The screenshot shows the Facebook App Dashboard for an app named 'My Website'. The dashboard is divided into two main sections: a left sidebar with navigation links and a main content area. The sidebar includes links for 'My Website', 'Dashboard', 'Settings', 'Status & Review', 'App Details', 'Roles', 'Open Graph', 'Alerts', 'Localize', 'Payments', 'Audience Network', 'Test Apps', and 'Insights'. The main content area displays the app's name 'My Website' with a logo, the App ID '1419459171658979', and the App Secret (masked with dots). Below this, there is a 'Facebook Login' section with a 'Trend' graph showing 'Active Users' over time. The graph includes checkboxes for 'Monthly Active Users', 'Weekly Active Users', and 'Daily Active Users'. The x-axis of the graph shows dates from April 15 to May 11.

You'll get a prompt asking you which platform, just choose iOS and you'll see this

Now go back to your app's dashboard on Facebook and click under settings click on add platform.

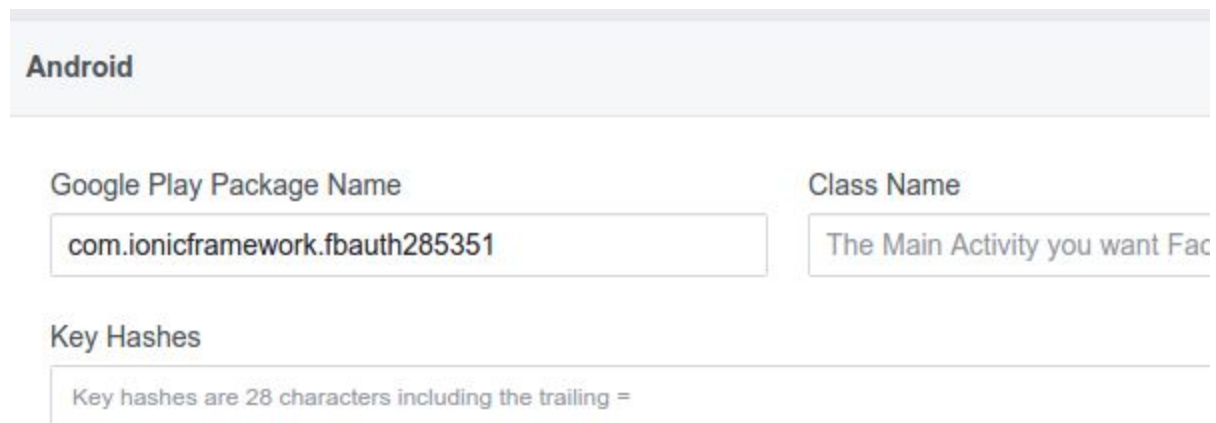


The screenshot shows the Facebook app dashboard with a sidebar on the left containing links: Open Graph, Alerts, Localize, Canvas Payments, Audience Network, Test Apps, and Analytics. The main content area displays the 'iOS' platform configuration modal. The modal has a 'Quick Start' button in the top right corner. The 'Bundle ID' field is highlighted with a red rectangular box. Below it is the 'URL Scheme Suffix (Optional)' field. To the right of the 'Bundle ID' field are the 'iPhone Store ID' and 'iPad Store ID' fields, each with a descriptive text: 'The ID to identify your app in the iOS Store' and 'The ID to identify your app in the iPad Store' respectively. Below these fields are two toggle switches: 'Single Sign On' (set to 'NO') and 'Automatically Log App Events for In-App Purchases on iOS (Recommended)' (set to 'YES'). The 'Automatically Log App Events' toggle has a detailed description: 'Logging App Events for In-App Purchases will allow you to see your in-app purchase events in your Insights dashboard and in ads reporting. This feature requires SDK version 3.22 or newer. Notice: To avoid double logging, you should not explicitly log in-app purchases via the Facebook SDK if this feature is turned on.' At the bottom of the modal is a '+ Add Platform' button.

Go to your app's config.xml file and copy bundle I'd paste it Bundle I'D field and save

Android

Now go back to your app's dashboard on Facebook and click under settings click on add platform

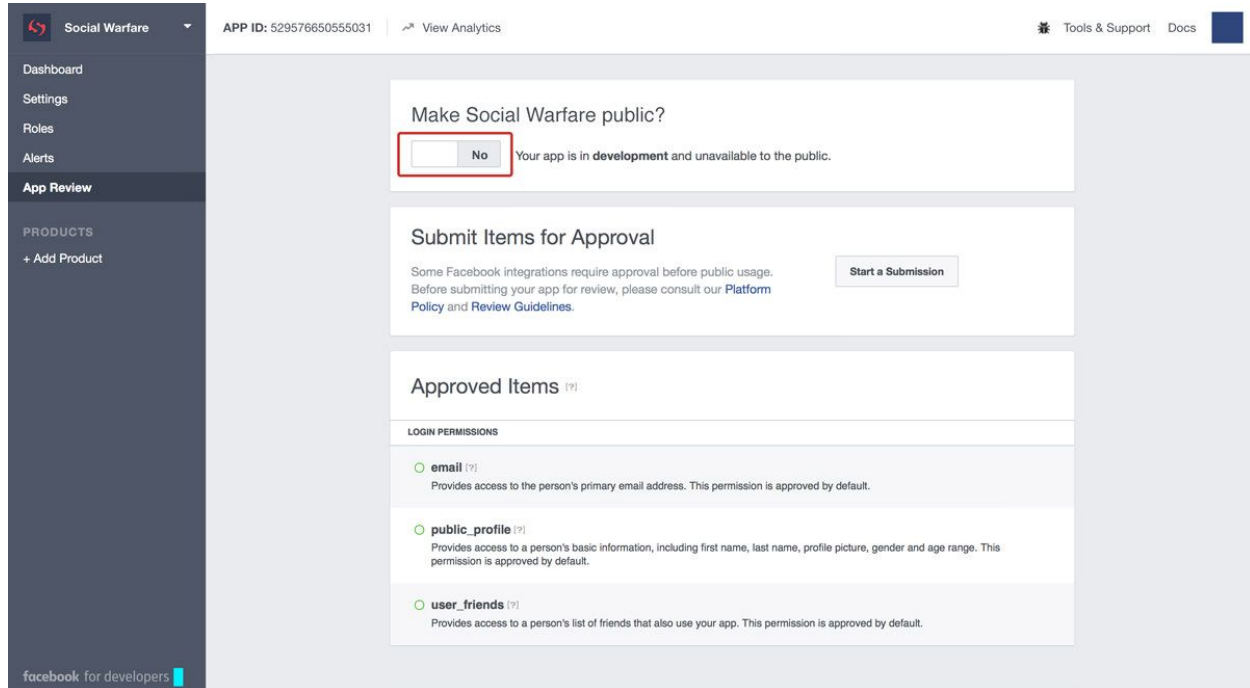


The screenshot shows the Facebook app dashboard with the 'Android' platform configuration modal. The modal has a title 'Android' at the top. Below the title are two fields: 'Google Play Package Name' and 'Class Name'. The 'Google Play Package Name' field contains the text 'com.ionicframework.fbauth285351'. The 'Class Name' field contains the text 'The Main Activity you want Fac'. Below these fields is a 'Key Hashes' section with a text input field containing the text 'Key hashes are 28 characters including the trailing ='.

Go to your app's config.xml file and copy bundle I'd paste it Bundle I'D field and save

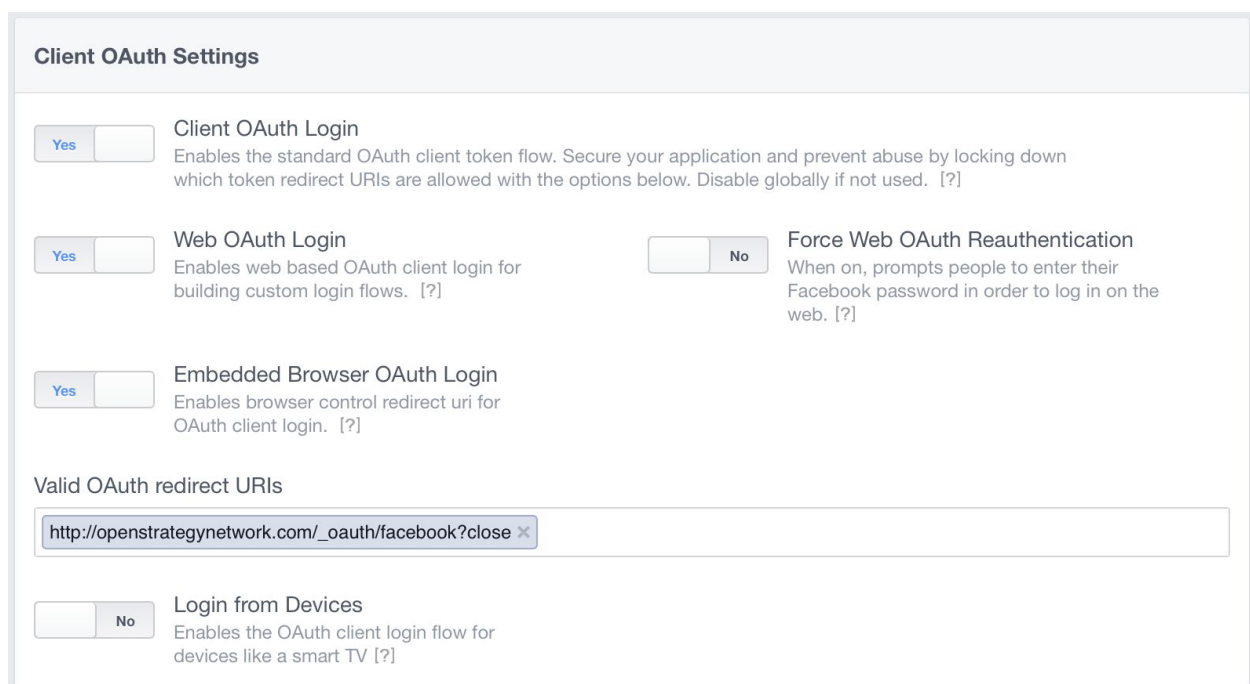
After that you need make your app public. Go to App review inside you have enable it.

Enable Firabase

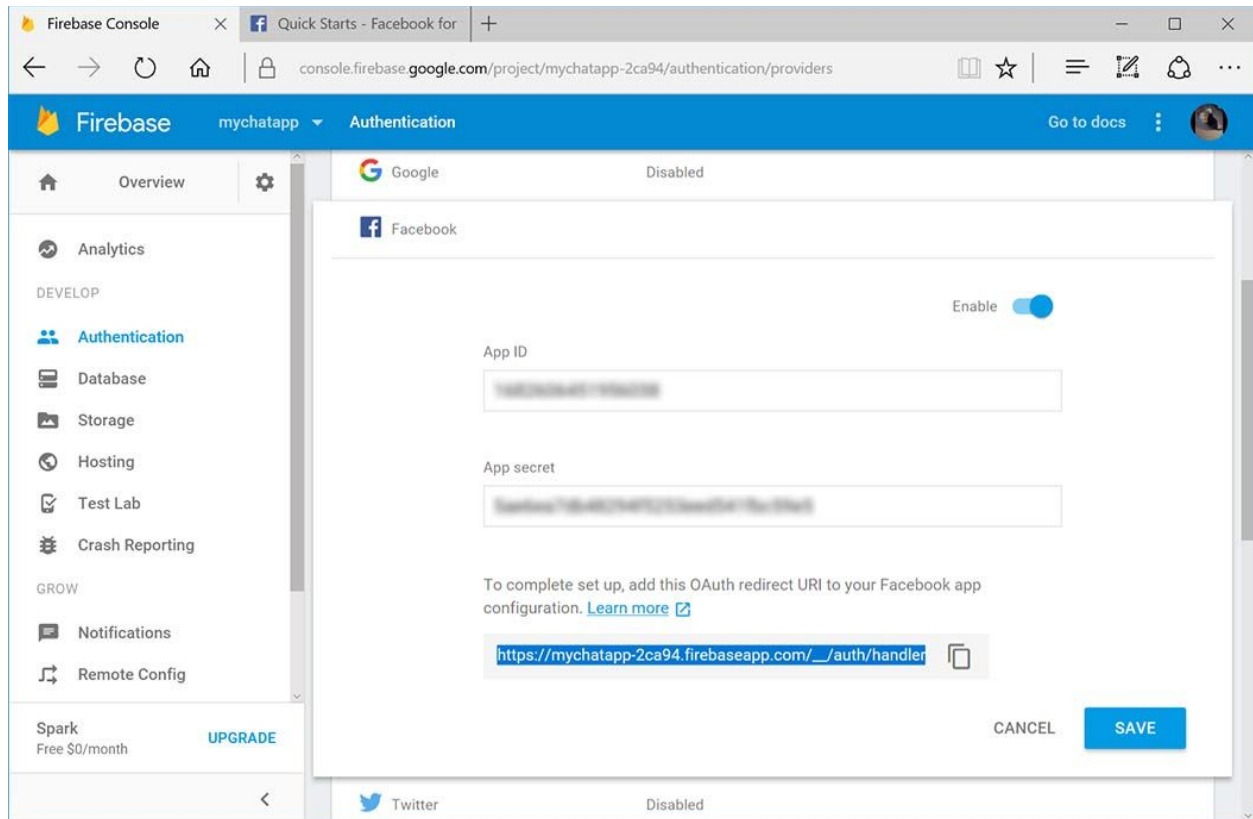


Now you need to go add product inside that choose Facebook Login.

After that facebook login setting add Valid OAuth redirect URIs



You can get this one from here. Copy this Valid OAuth redirect URIs and paste it there.



Now go back to your app's dashboard on Facebook and note down App ID and App Secret


Go to your Firebase Console and enable Facebook Authentication

Go to -

console.firebase.google.com/project/YOURPROJECTNAMEHERE/authentication/providers

Enable Facebook, it's going to ask you for your App ID and App Secret. Copy and paste it from your app's dashboard on Facebook

Install Cordova Facebook Plugin


 Facebook

Enable ☒

App ID

App secret

To complete set up, add this OAuth redirect URI to your Facebook app configuration. [Learn more](#)

https://moaningmyrtle-2d294.firebaseio.com/_/auth/handler 

CANCEL **SAVE**

APP_NAME and APP_ID From Facebook Dashboard

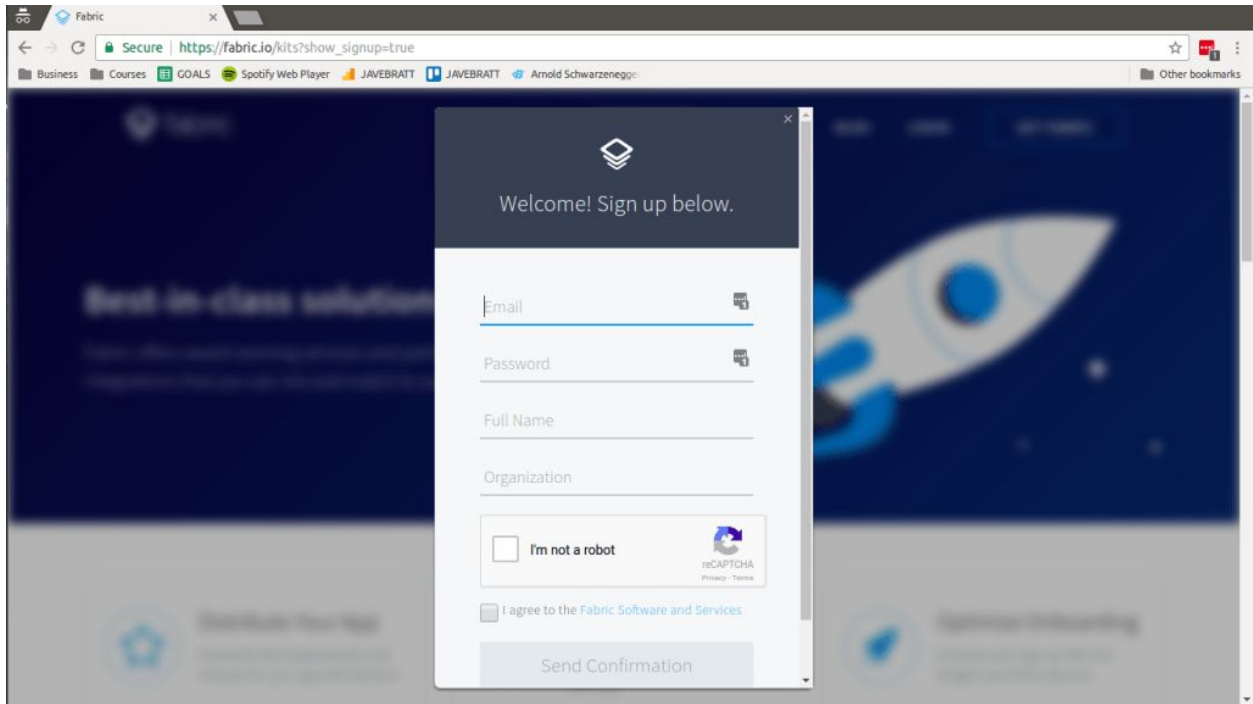
```
$ ionic cordova plugin add cordova-plugin-facebook4 --variable APP_ID="123456789"
--variable APP_NAME="myApplication"
$ npm install --save @ionic-native/facebook
```

Twitter Authentication:

- Step #1: Set up your Fabric Account.
- Step #2: Get your Fabric API Key.
- Step #3: Create your app in Fabric.
- Step #4: Install the Twitter Connect Plugin
- Step #5: Enable Twitter Authentication in Firebase.

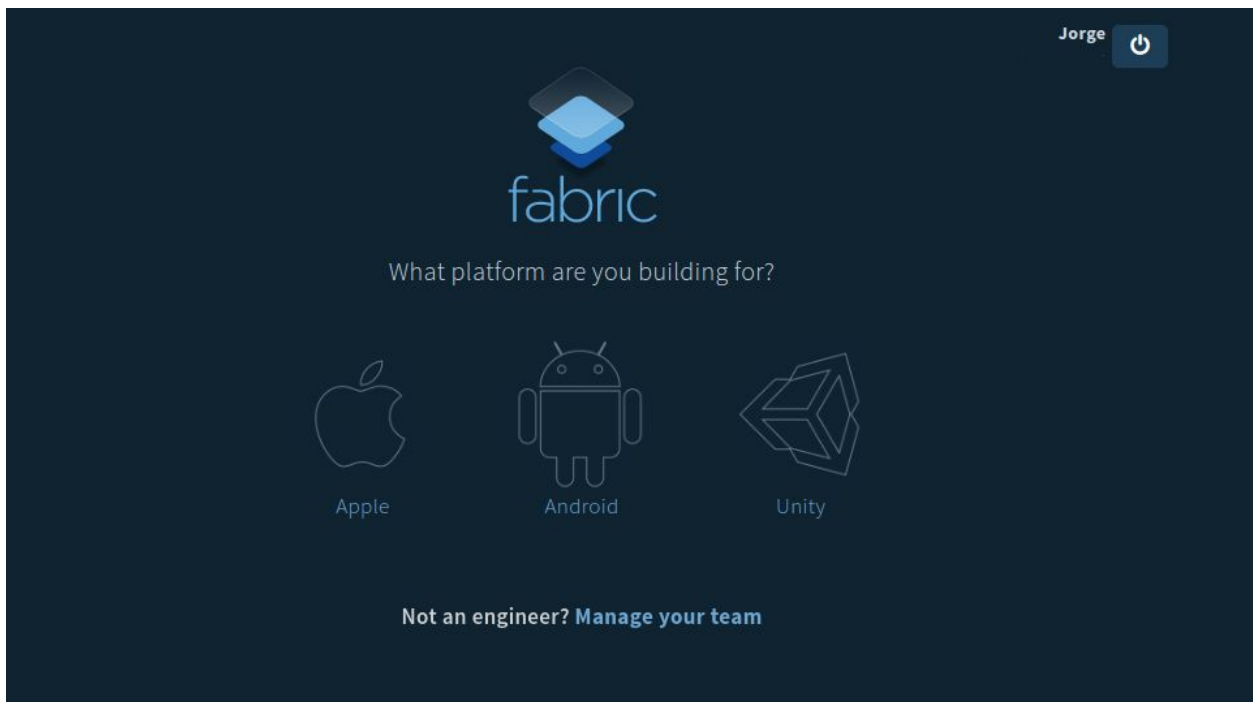
Step#1: Set up your Fabric Account.

Create an account in [Fabric website](https://fabric.io)



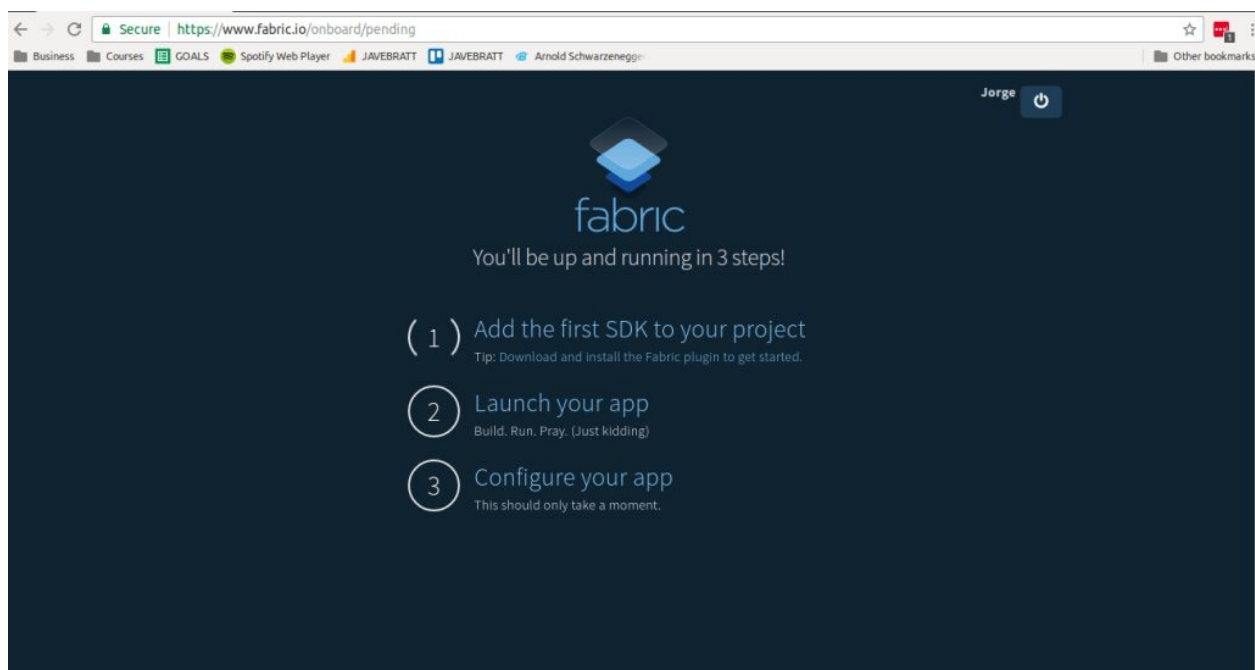
A screenshot of a web browser showing the Fabric website's sign-up page. The browser's address bar displays the URL https://fabric.io/kits?show_signup=true. A modal form is centered on the screen with the heading "Welcome! Sign up below." and the Fabric logo. The form contains the following fields: "Email", "Password", "Full Name", and "Organization". Below these fields is a reCAPTCHA section with a checkbox labeled "I'm not a robot" and a reCAPTCHA logo. At the bottom of the form is a checkbox labeled "I agree to the Fabric Software and Services" and a "Send Confirmation" button. The background of the website is dark blue with a rocket ship illustration.

Select either iOS or android platform



When you pick the platform you'll go through an on-boarding process (*yup, they actually think this is good*), just read through every single message and follow the instructions step by step


Once everything is installed, you need to build and run the app, once you do, it will send a signal to Fabric letting them know that you have successfully installed and ran the SDK, which will let you pass this "pending" page





Step #2: Get your Fabric API key

Fabric uses an API key and build secret to authenticate your app:

1. Login to Fabric account and open <https://fabric.io/kits/android/crashlytics/install>
2. Find the metadata code block in AndroidManifest.xml
3. Find your API Key pre filled in the code.

SummaryInstallCode Examples



Add Your API Key

AndroidManifest.xml2 AdditionsJAVEBRATT

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity android:name=".MainActivity" android:label="@string/app_name" >
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
    <meta-data
      android:name="io.fabric.ApiKey"
      android:value="" />
  </application>
  <uses-permission android:name="android.permission.INTERNET" />
</manifest>
```

Account Provisioning and Terms of Service

INSTALL TWITTER VIA GRADLE

- 1 Add the Kit to Your build.gradle
- 2 Add Your API Key
- 3 Account Provisioning and Terms of Service
- 4 Initialize Your Kit
- 5 Build

Want this done automatically?
Get our Android plugin

Step #3: Create a Twitter app

Go to this link <https://apps.twitter.com/>. Here you need to sign up.

After you need to create app.

Create an application

Application Details

Name: *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description: *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website: *

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL:

Where should we return after successfully authenticating? For [@Anywhere applications](#), only the domain specified in the callback will be used. [OAuth 1.0a](#) applications should explicitly specify their `oauth_callback` URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

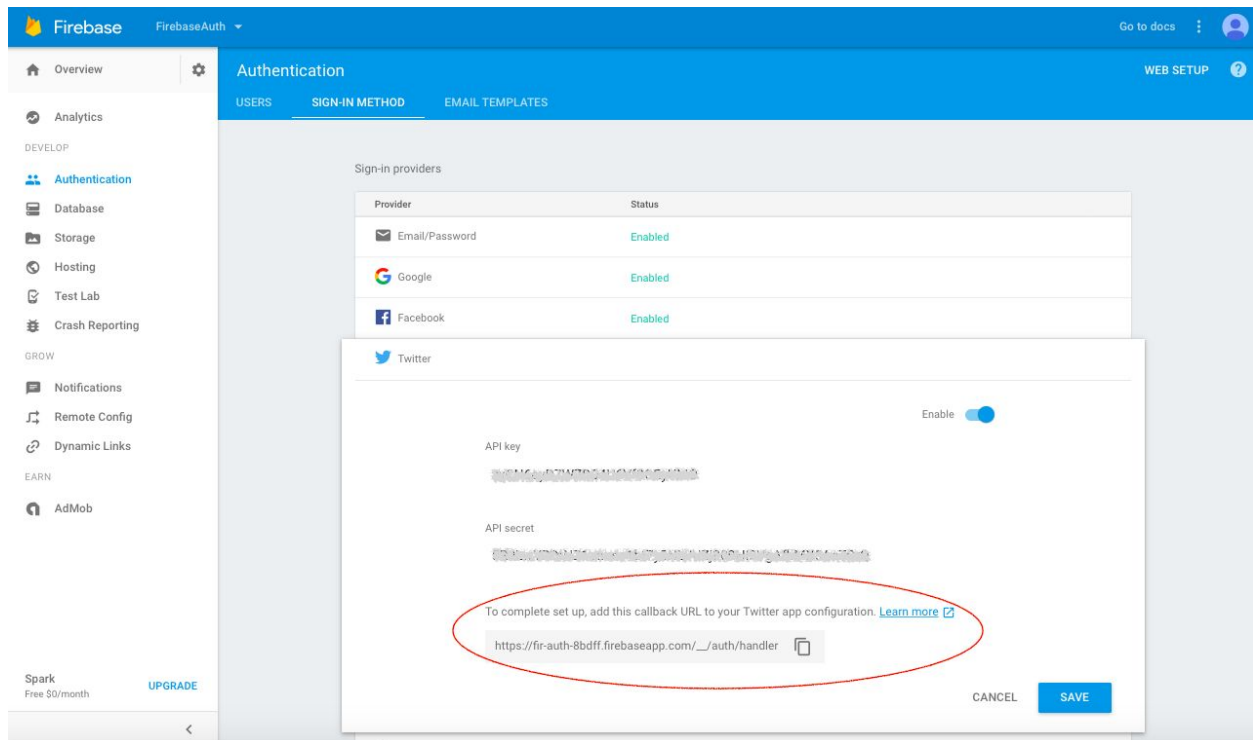
Add your app name.

Add description.

Website URL.

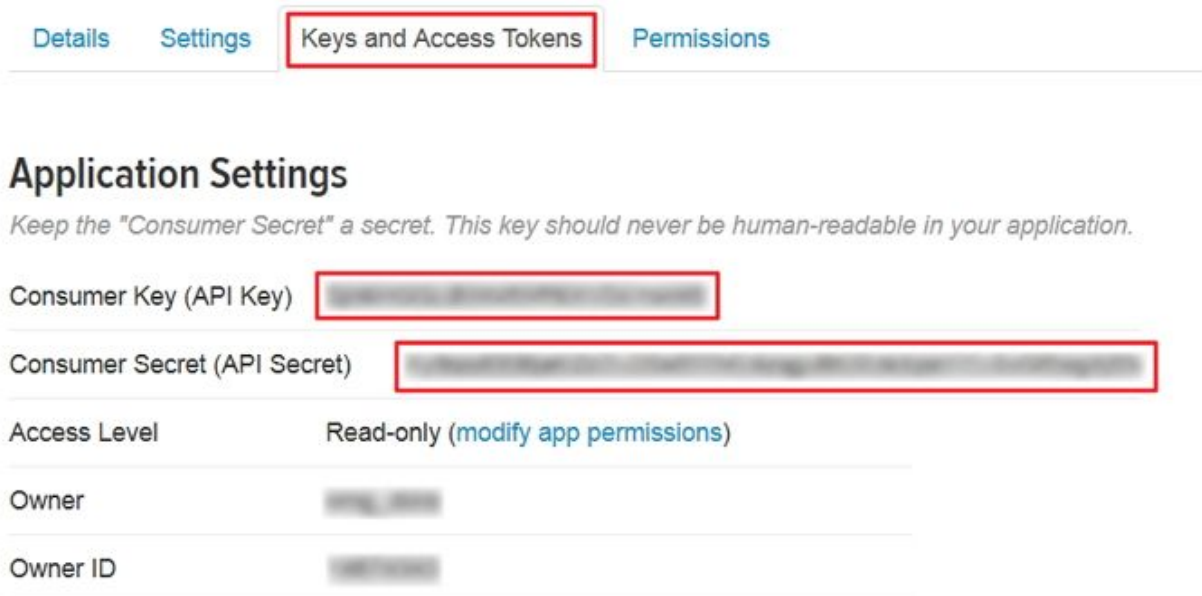
Callback URL

Note:- Callback URL get from firebase



Copy this URL and it Callback URL.

And you can get the consumer key and consumer secret from app Keys And Access Tokens.



Step #4: Install the Twitter Connect Plugin

```
$ ionic cordova plugin add twitter-connect-plugin --variable FABRIC_KEY=fabric_API_key
```

```
$ npm install --save @ionic-native/twitter-connect
```

Note:- How you will get fabric API key in above tutorial is there

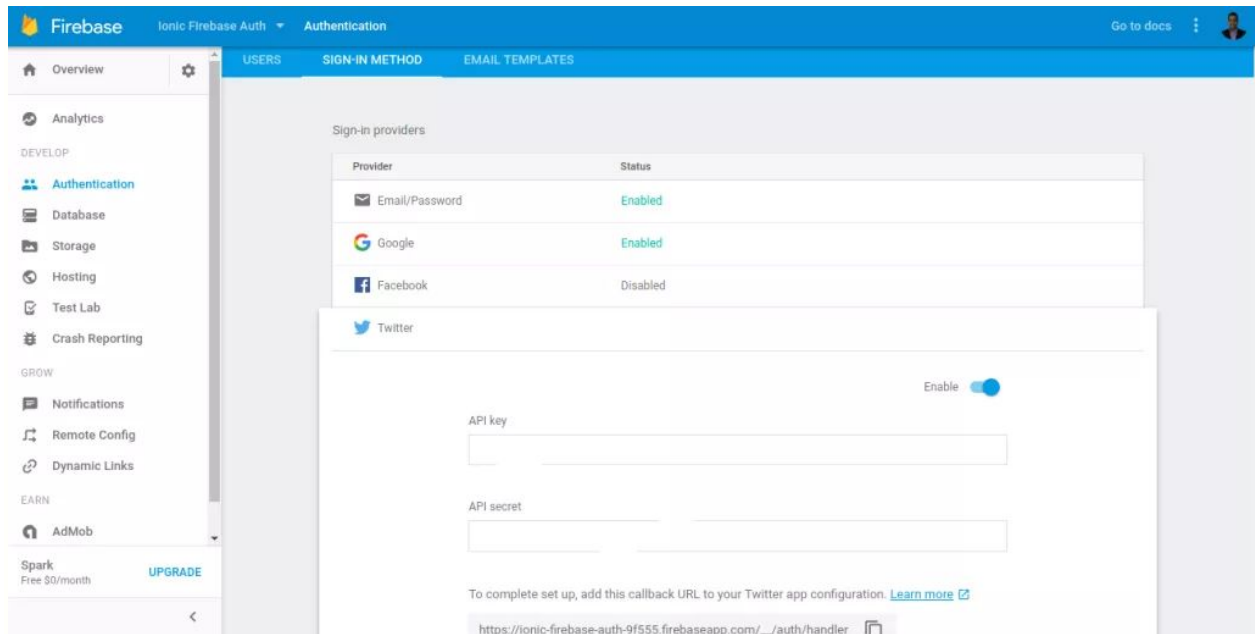
Once the plugin is installed, you'll need to do some configuration, go ahead and open the **config.xml** file that's in the project root, and **right before the closing `</widget>` tag**, add this:

```
<preference name="TwitterConsumerKey" value="<Twitter Consumer Key>" />
<preference name="TwitterConsumerSecret" value="<Twitter Consumer Secret>"
```

Step #5: Enable Twitter Authentication in Firebase.

Now you need to tell your Firebase app to allow users to Sign-In with Twitter, for that go to your Firebase Console

Choose your app and inside the Authentication Tab go to "Sign-In Method" and enable Twitter, it's going to ask you for an API Key and Secret, you'll use the same you just used, the ones for the app you created in Fabric.



Add Push Notification:

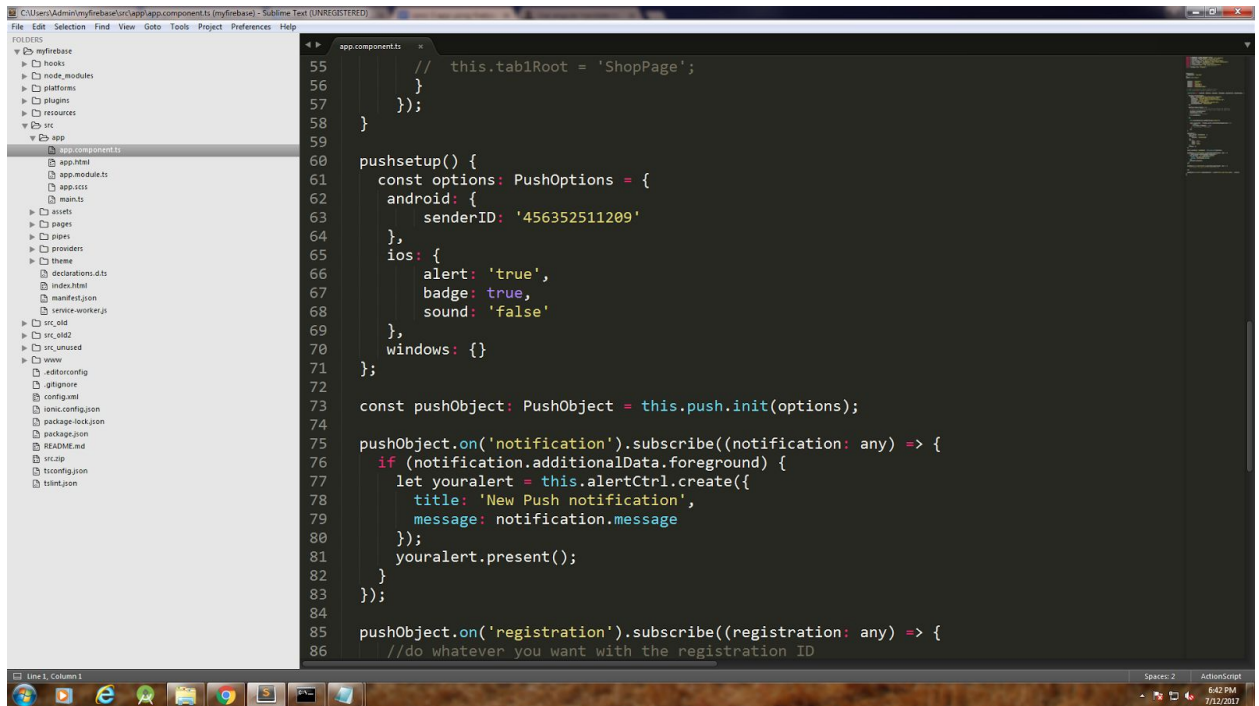
Install this plugin:

Note:- You will get `SENDER_ID` from **GoogleService-Info.plist** file.

```
ionic cordova plugin add phonegap-plugin-push --variable  
SENDER_ID=XXXXXXXXXX
```

```
npm install --save @ionic-native/push
```

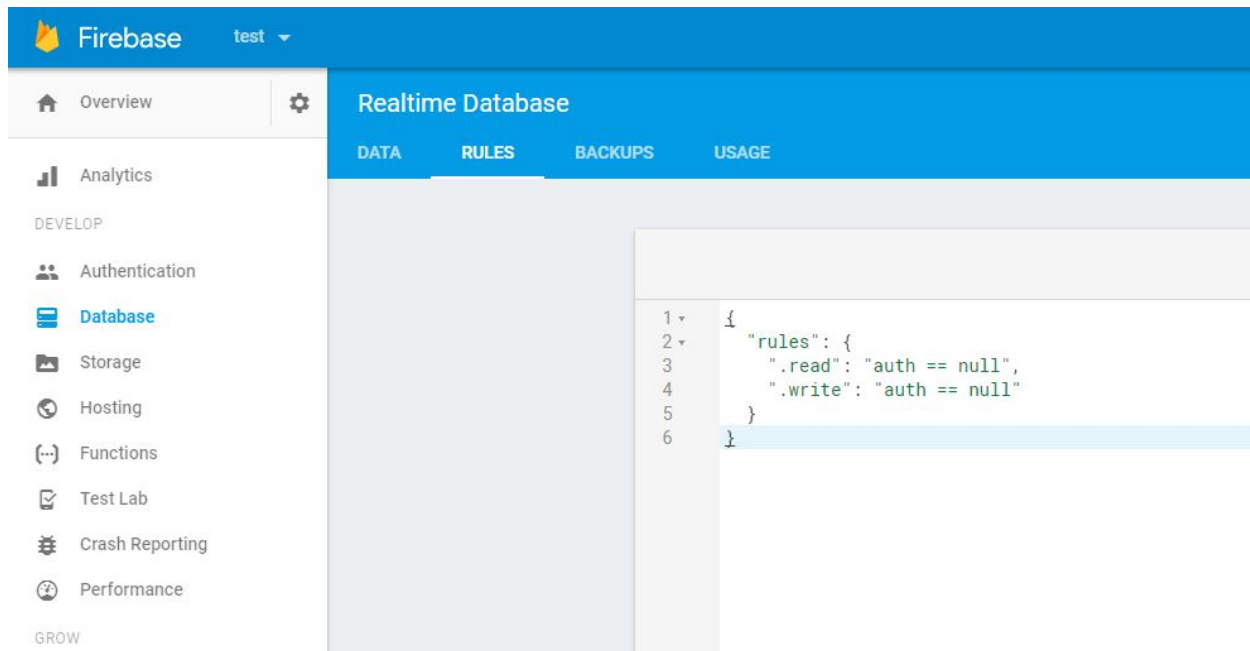
Add sender Id in `app.component.ts` file.



Firestore rules:

Change Firestore database rules to make contents available for customers

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

If you want secure you data you can also use firebase rule we have created. Here write permission is disable for users other than admin

You have to register and make registered user as admin before using below firebase below rule. Replace **Admin UID** in below firebase rule with actual Admin User uid

```
{
  "rules": {
    "Category_List":{
      ".read": true,
      ".write": "root.child('Customer-Role').child(auth.uid).child('role').val() ===
'Admin'"
    },
  },
}
```

```

"product-List":{
    ".read": true,

    ".write": "root.child('Customer-Role').child(auth.uid).child('role').val() ===
'Admin'"
},

"Vendor-List":{
    ".read": true,

    ".write": "root.child('Customer-Role').child(auth.uid).child('role').val() ===
'Admin'"
},

"Brand-List":{
    ".read": true,

    ".write": "root.child('Customer-Role').child(auth.uid).child('role').val() ===
'Admin'"
},

"Banners":{
    ".read": true,

    ".write": "root.child('Customer-Role').child(auth.uid).child('role').val() ===
'Admin'"
},

"Setting":{
    ".read": true,

```

```

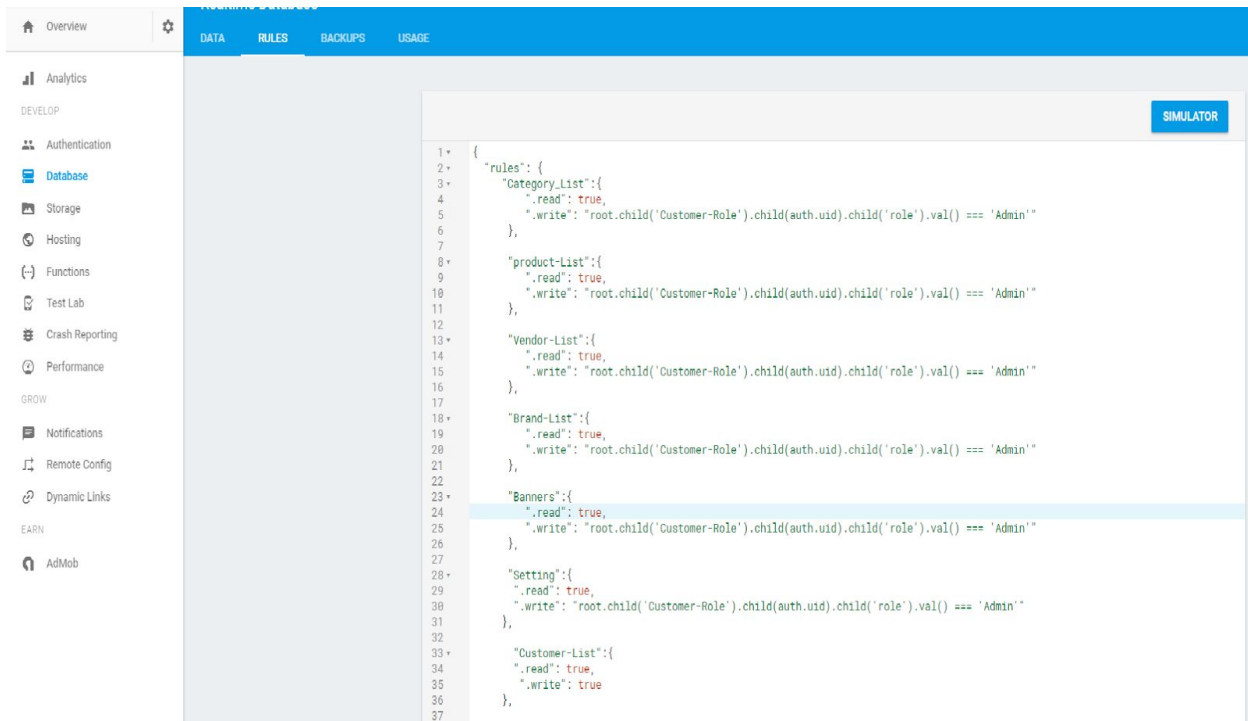
    ".write": "root.child('Customer-Role').child(auth.uid).child('role').val() === 'Admin'"
  },

  "Customer-List":{
    ".read": true,
    ".write": true
  },

  "Customer-Role":{
    ".read": true,
    ".write": "auth.uid === 'Admin UID'"
  },

  "Order-List":{
    ".read": true,
    ".write": true
  }
}
}

```



Admin Panel

You can build separate app with admin functionality for managing your shop. With this you can do following activities

1. Add and manage categories
2. Add and manage products
3. Add and manage customers
4. Add and manage vendors
5. Add and manage brands
6. Add and manage images
7. Create and manage orders
8. Configure and Manage PayPal payment method
9. Configure and Manage Stripe payment method
10. Manage Banners and Currency

Add and manage categories

Add and manage products

Add and manage customers

Add and manage vendors

Add and manage brands

Add and manage images

Create and manage orders

Configure and manage payments

Manage banners and currency

Separating customer app

1. Delete admin folder
2. Remove **tab4Root = 'AdminPage'**; in `src/app/app.component.ts` file
3. Remove **<ion-tab [root]="tab4Root" tabTitle="Admin" tabIcon="person"></ion-tab>** in `src/app/app.html` file
4. Delete End Admin Only Functions section in `src/providers/service.ts` file. This is optional and you will find any error if this is not deleted

Integrating Wordpress Blog

1. Upload the json-api folder to the /wp-content/plugins/directory or install directly through the plugin installer. [JSON API By Dan Phiffer](#) (You have to upload attached json-api to show author image in blog post)
2. Activate the plugin through the 'Plugins' menu in WordPress or by using the link provided by the plugin installer.
3. Activate Controllers Wordpress Settings->JSON API->Controllers. (core, post, respond, widgets)
4. To integrate wordpress blog add wordpress website url in myApp/src/providers/config.ts file