



## **Edelivery User Documentation**

## ➤ Install Android Studio

You can check steps for download and install android studio for mac/windows/linux from [here](#)  
[download the latest version of Android Studio](#).

Here are the steps for installing studio in ubuntu

To install Android Studio on ubuntu, proceed as follows:

1. Unpack the .zip file you downloaded to an appropriate location for your applications, such as within /usr/local/ for your user profile, or /opt/ for shared users.  
To launch Android Studio, open a terminal, navigate to the android-studio/bin/ directory, and open terminal there execute studio.sh by entering `./studio.sh`
2. Select whether you want to import previous Android Studio settings or not, then click **OK**.
3. The Android Studio Setup Wizard guides you through the rest of the setup, which includes downloading Android SDK components that are required for development.
4. The following video shows each step of the recommended setup procedure. <https://developer.android.com/studio/videos/studio-install-linux.mp4>

## ➤ Changes In Projects(android)

### 1.Open Project in Android Studio

File->open

### 2.Change package name

-Find <PROJECT\_ROOT>\app\build.gradle and Change package Name  
(replace your package name here)

```
applicationId "com.elluminatiinc.edelivery"  
//replace your package name here
```

### 3.Change BASE\_URL in this build.gradle file

-You can set BASE\_URL according to flavours.  
-Default flavour is “developer” so for running your project in your sever you need to change your developer BASE\_URL

```
productFlavors {  
    production {  
        flavorDimensions "default"  
        buildConfigField "String", "BASE_URL", "https://edelivery.appemporio.net/v3/"  
        // if IMAGE_URL is not S3 bucket url then it will ne same as BASE_URL  
        buildConfigField "String", "IMAGE_URL", "https://edelivery.appemporio.net/v3/"  
    }  
  
    staging {  
        flavorDimensions "default"  
        buildConfigField "String", "BASE_URL", "https://edeliverydemo.appemporio.net/"  
        buildConfigField "String", "IMAGE_URL", "https://edeliverydemo.appemporio.net/"  
    }  
  
    developer {  
        flavorDimensions "default"  
        buildConfigField "String", "BASE_URL", "https://apiedeliverynew.appemporio.net/v3/"  
        buildConfigField "String", "IMAGE_URL", "https://apiedeliverynew.appemporio.net/v3/"  
    }  
  
    local {  
        flavorDimensions "default"  
        buildConfigField "String", "BASE_URL", "http://192.168.0.160:8000/v3/"  
    }  
}
```

```
buildConfigField "String", "IMAGE_URL", "\"http://192.168.0.160:8000/v3/\""  
}
```

#### 4.Change in build.gradle file

-Change version code and version number to 1

```
versionCode 1  
versionName "1.0.0"
```

#### 5. Change package name in provider\_paths.xml file(both app)

-For FileProvider to request content URIs for the images/ subdirectory of your private file area.

-We are using this for getting content uri of image captured through camera

-Goto eber->src->res->xml->provider\_paths.xml

```
<?xml version="1.0" encoding="utf-8"?>  
    <paths>  
        <external-path  
            name="my_images"  
            path="Android/data/com.edelivery.user/files/Pictures" />  
            //replace your package name here  
        </external-path>  
    </paths>
```

#### 6.Change your Theme color

File Goto : utis->AppColor.java

```
public static int COLOR_THEME = Color.parseColor("#00AFC2");//add your color hex
```

## 7.Change drawable xml icon color

File name : eber -> src -> main -> res ->drawable

## 8.Change drawable icon color

Ex. solace screen, image logo, notification icon etc...

File name : eber -> src -> main -> res

drawable-hdpi

drawable-xhdpi

drawable-xxhdpi

## 9.Change font

-Add your font file in edelivery -> src -> main ->assets ->fonts

-Change in your all custom view (button, text, Edittext etc)

```
private boolean setCustomFont(Context ctx, String asset) {
    try {
        if (typeface == null) {
            // Log.i(TAG, "asset:: " + "fonts/" + asset);
            typeface = Typeface.createFromAsset(ctx.getAssets(),
                "fonts/Roboto_Regular_0.ttf");
        }
    } catch (Exception e) {
        AppLog.handleException(TAG, e);
        // Log.e(TAG, "Could not get typeface: " + e.getMessage());
        return false;
    }
    setTypeface(typeface);
    return true;
}
```

## 10. Change include subdomain in

- Goto src->main->res->xml->network\_security\_config.xml
- For opting-out from cleartext network traffic  
(e.g. HTTP, FTP, WebSockets, XMPP, IMAP, SMTP)

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
  <domain-config cleartextTrafficPermitted="true">

    <domain includeSubdomains="true">192.168.0.141</domain> //Add Your Base Url
  </domain-config>
</network-security-config>
```

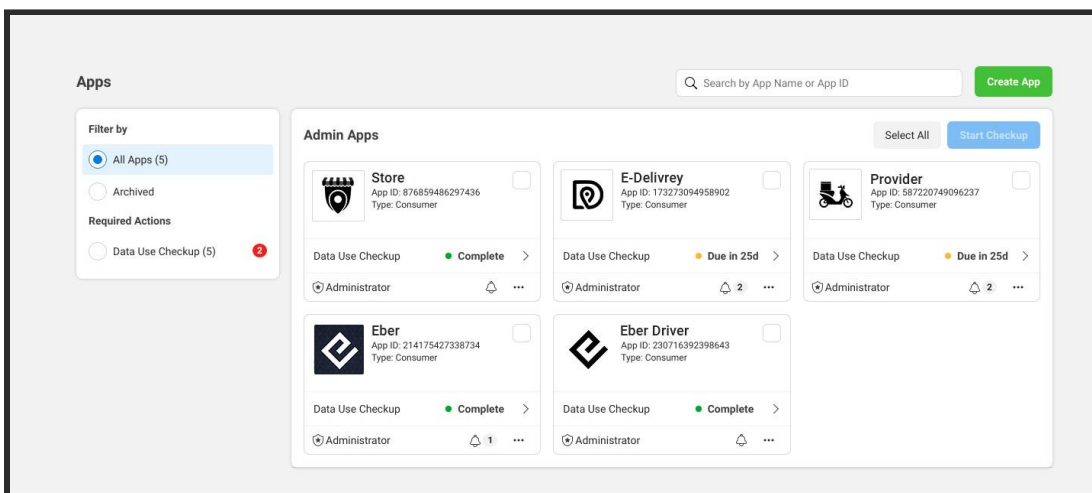
## ★ FACEBOOK\_APP\_ID and FB\_LOGIN\_PROTOCOL\_SCHEME (facebook Login)

For enabling facebook social login

Create facebook account after open facebook developer site

<https://developers.facebook.com/apps/>

### 1. Create a New App



2. Click on customer after click next

The screenshot shows the 'Create an App' dialog with the 'Type' tab selected. Under 'Select an app type', the 'Customer' option is highlighted with a blue border and a blue dot. The other options are 'Business', 'Sports', and 'Gaming'.

**Create an App** [X] Cancel

**Type** Details

**Select an app type**  
The app type can't be changed after your app is created. [Learn more](#)

- Business**  
Create or manage business assets such as Page, Event, Group, Ads, Messenger, WhatsApp and Instagram Graph APIs using the available business permissions, features and products.
- Customer** (selected)  
Connect consumer products, and permissions, like Facebook Login and Instagram Basic Display to your app.
- Sports**  
Create an HTML5 game hosted on Facebook.
- Gaming**  
Connect an off-platform game to Facebook Login.

3. Add your app name

Provide the Development and Release Key Hashes for Your App

<https://developers.facebook.com/docs/facebook-login/android/#6--provide-the-development-and-release-key-hashes-for-your-app>

The screenshot shows the 'Create an App' dialog with the 'Details' tab selected. The 'Display name' field is empty. The 'App Contact Email' field contains 'eber.support@elluminatiinc.com'. The 'Business Account' dropdown is set to 'No Business Account selected'. At the bottom, there are 'Previous' and 'Create App' buttons.

**Create an App** [X] Cancel

**Type** **Details**

**Add details**

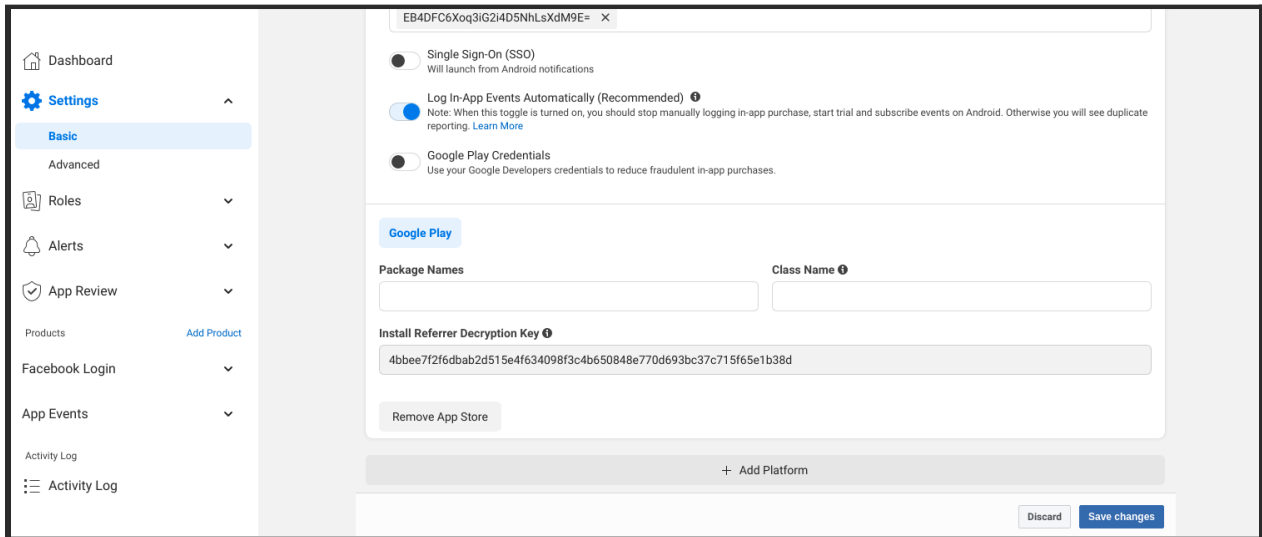
**Display name**  
This is the app name associated with your app ID. You can change this later.

**App Contact Email**  
This email address is used to contact you about potential policy violations, app restrictions or steps to recover the app if it's been deleted or compromised.

**Business Account · Optional**  
To access certain permissions or features, apps need to be connected to a Business Account.

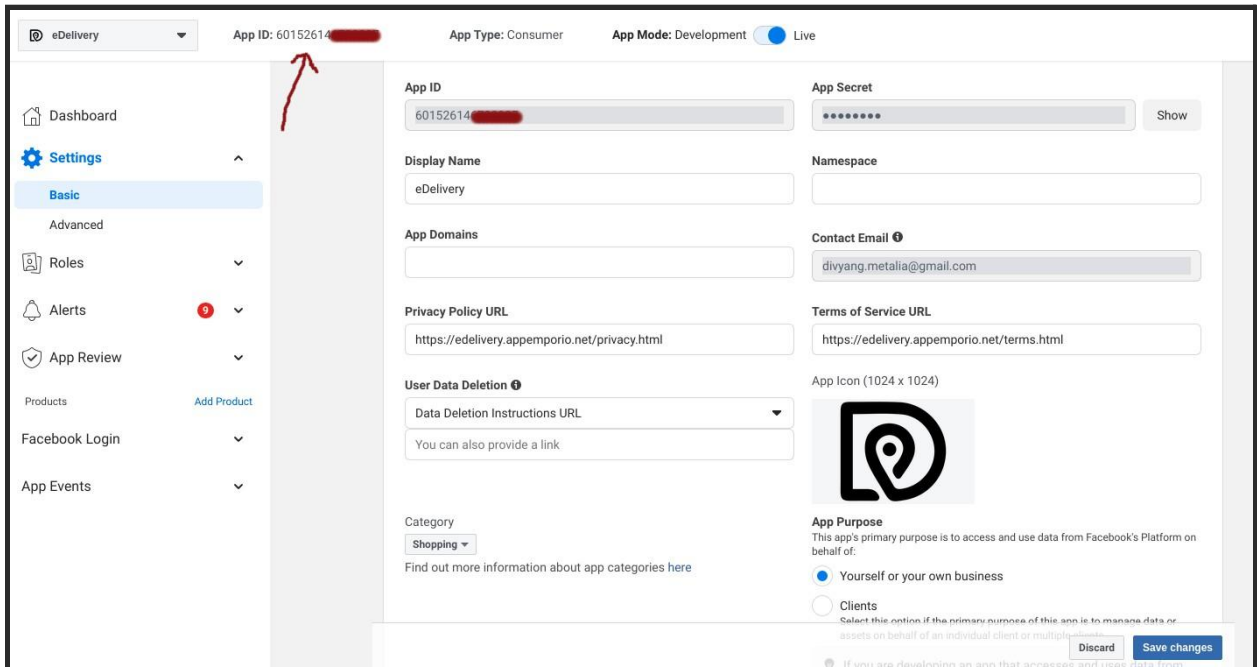
By proceeding, you agree to the [Facebook Platform Terms](#) and [Developer Policies](#). [Previous] [Create App]

4. Click on settings -> basic -> + Add Platform -> after select android



This screenshot shows the 'Basic' configuration tab for an app. The left sidebar contains navigation links: Dashboard, Settings (selected), Basic (selected), Advanced, Roles, Alerts, App Review, Products (with an 'Add Product' link), Facebook Login, App Events, Activity Log, and another Activity Log link. The main content area is titled with a browser tab 'EB4DFC6Xoq3IG2i4D5NhLsXdM9E= X'. It features three toggle switches: 'Single Sign-On (SSO)' (disabled), 'Log In-App Events Automatically (Recommended)' (enabled), and 'Google Play Credentials' (disabled). Below these is a 'Google Play' section with input fields for 'Package Names' and 'Class Name', and a text field for 'Install Referrer Decryption Key' containing a long alphanumeric string. A 'Remove App Store' button is present. At the bottom, there is a '+ Add Platform' button and 'Discard' and 'Save changes' buttons.

5. Now you get one app id



This screenshot displays the 'App Details' page for an app named 'eDelivery'. The top header shows 'App ID: 60152614', 'App Type: Consumer', and 'App Mode: Development' with a 'Live' toggle. The left sidebar is identical to the previous screenshot. The main content area is divided into two columns. The left column includes fields for 'App ID' (60152614), 'Display Name' (eDelivery), 'App Domains', 'Privacy Policy URL' (https://edelivery.appemporio.net/privacy.html), 'User Data Deletion' (with a dropdown menu), and 'Category' (Shopping). The right column includes 'App Secret' (masked with dots), 'Namespace', 'Contact Email' (divyang.metalia@gmail.com), 'Terms of Service URL' (https://edelivery.appemporio.net/terms.html), 'App Icon' (1024 x 1024), and 'App Purpose' (with radio buttons for 'Yourself or your own business' and 'Clients'). A red arrow points to the 'App ID' field. At the bottom, there are 'Discard' and 'Save changes' buttons.



After past this key in FACEBOOK\_APP\_ID (ex.12345678901) and  
FB\_LOGIN\_PROTOCOL\_SCHEME is (ex. fb12345678901) for more details check this  
video

<https://www.youtube.com/watch?v=qAN9KYhOSec>

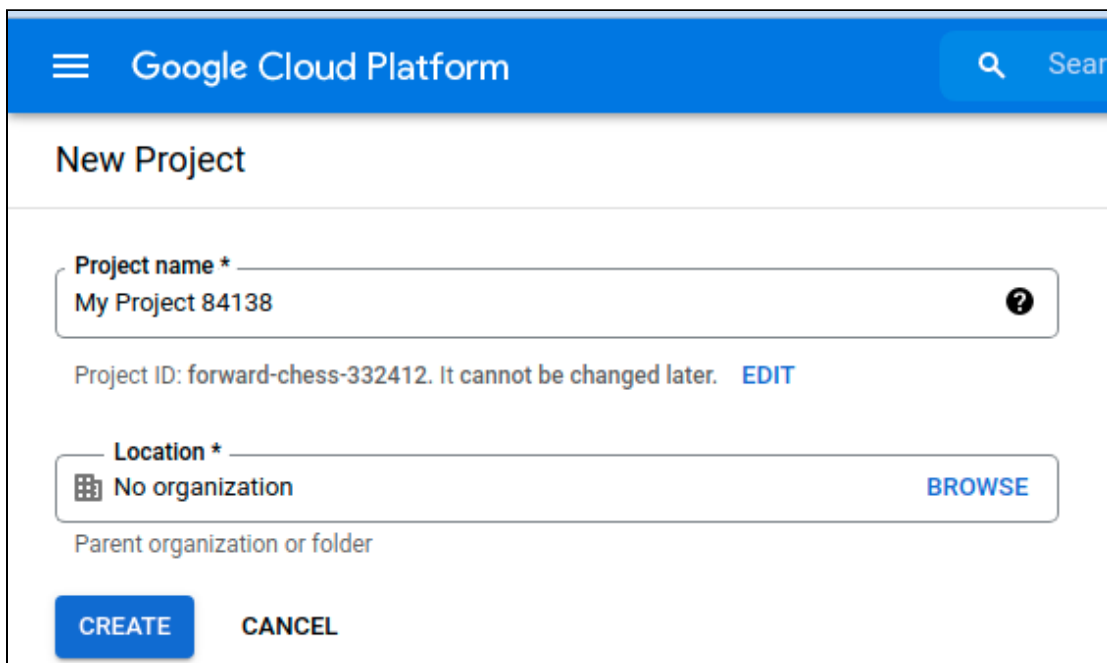
```
resValue "string", "FACEBOOK_APP_ID", "601526146700337"  
resValue "string", "FB_LOGIN_PROTOCOL_SCHEME", "fb601526146700337"
```

## ● Google Cloud Console (Google Apis)

### - Create Google Cloud Project

For Using Google Apis (Google Map Api, Geocoding Api, Distance matrix Api etc) In our project we need to create project in google cloud console

1. Open the [Google Cloud Console](#).
2. Next to "Google Cloud Platform," click the Down arrow . A dialog listing current projects appears.
3. Click **New Project**. The New Project screen appears.
4. In the **Project Name** field, enter a descriptive name for your project. If you're executing a quickstart, use "Quickstart."
5. Click **Organization** and select your organization.
6. In the **Location** field, click **Browse** to display potential locations for your project.
7. Click a location and click **Select**.
8. Click **Create**. The console navigates to the Dashboard page and your project is created within a few minutes.



The screenshot shows the 'New Project' dialog in the Google Cloud Platform console. The header bar is blue with the 'Google Cloud Platform' logo and a search bar. The dialog title is 'New Project'. It contains two main input fields: 'Project name \*' with the value 'My Project 84138' and a help icon, and 'Location \*' with the value 'No organization' and a 'BROWSE' button. Below the location field, it says 'Parent organization or folder'. At the bottom, there are 'CREATE' and 'CANCEL' buttons. A project ID is displayed: 'Project ID: forward-chess-332412. It cannot be changed later. EDIT'.

For further information on GCP projects, refer to [Creating and managing projects](#).

## - Activate Billing

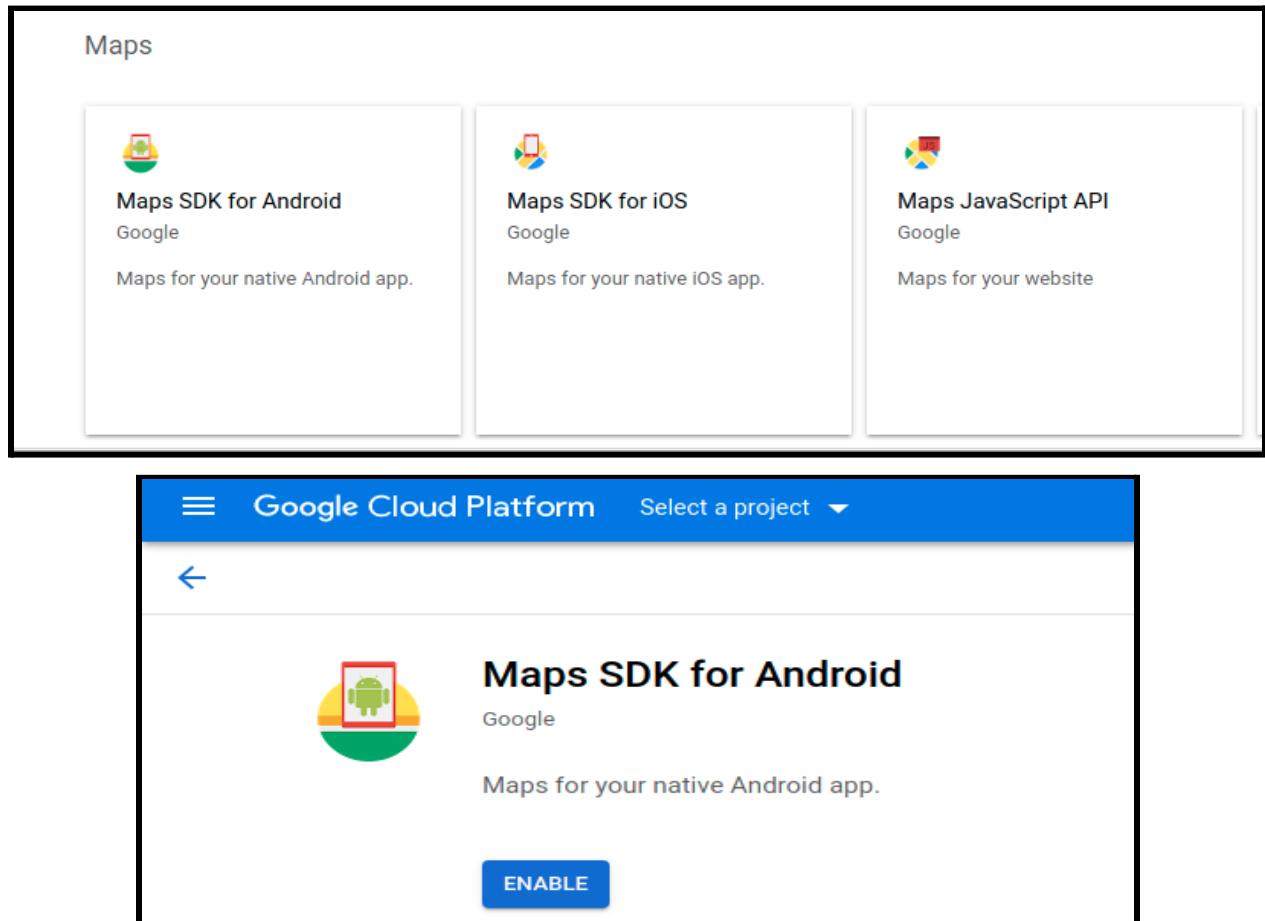
After successfully registering for a trial account you will be entitled to ~\$300 free credits that you can spend within the Google Cloud Platform (GCP). However It would recommend to set up billing by adding a valid credit / debit card.

You can create a Billing Account [here](#) and its worthing remembering that one billing account can be used across multiple GCP projects.

## - Enable a Google Workspace API

1. Open the [Google Cloud Console](#).
2. Next to "Google Cloud Platform," click the Down arrow and select a project.
3. In the top-left corner, click Menu > **APIs & Services**.
4. Click **Enable APIs and Services**. The **Welcome to API Library** page appears.
5. In the search field, enter the name of the API you want to enable.  
For example, type "Map API" to find the Gmail API. If you are enabling an API for a quickstart, refer to the quickstart's Prerequisites section for the API to enable.
6. Click the API to enable. The API page appears.
7. Click **Enable**. The Overview page appears.
8. To enable an additional API, repeat steps 3 - 7.

For Example:



## ★ Make these libraries enable

### → Maps SDK for Android

With the Maps SDK for Android, add maps to your [Android app](#) including [Wear OS](#) apps using Google Maps data, map displays, and map gesture responses.  
on web pages and mobile devices.Geolocation API

For more detail :-

<https://developers.google.com/maps/documentation/android-sdk/overview>

### → Geocoding API

**Geocoding** is the process of converting addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739), which you can use to place markers on a map, or position the map.  
The Geocoding API provides a direct way to access these services via an HTTP request.

For more detail :-

<https://developers.google.com/maps/documentation/geocoding/overview>

### → Distance Matrix API

The Distance Matrix API is a service that provides travel distance and time for a matrix of origins and destinations.

For more detail :-

<https://developers.google.com/maps/documentation/distance-matrix/overview>

### → Directions API

Provide directions for multiple transportation modes, featuring real-time traffic information.

For more detail :-

<https://developers.google.com/maps/documentation/directions>

### → Places API

The Places API is a service that returns information about places using HTTP requests. Places are defined within this API as establishments, geographic locations, or prominent points of interest.

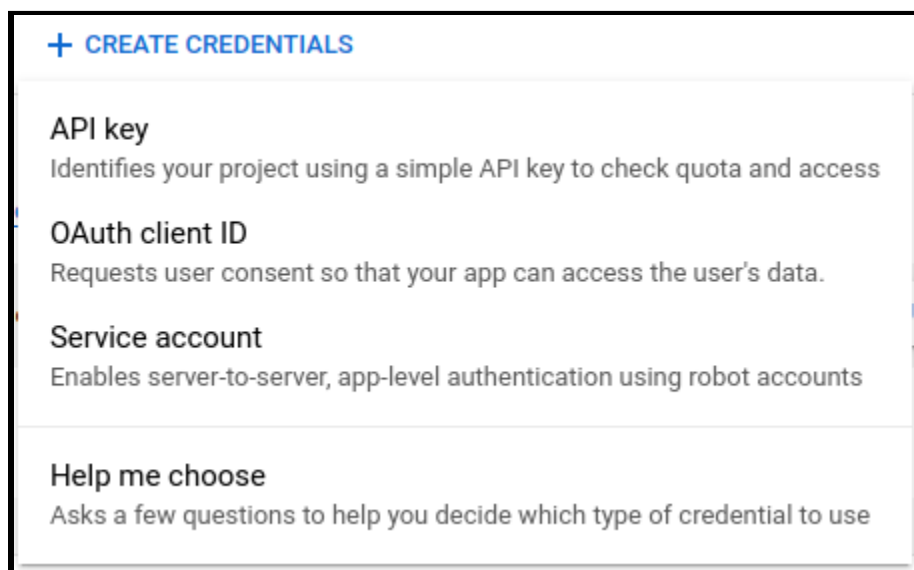
→ For more detail :-

<https://developers.google.com/maps/documentation/places/web-service/overview>

For more information on apis you can refer : <https://developers.google.com/maps/documentation>

### - Create Api key

1. Go to the Google Maps Platform > Credentials page.  
[Go to the Credentials page](#)
2. On the Credentials page, click Create credentials > API key.  
The API key created dialog displays your newly created API key.
3. Click Close.  
The new API key is listed on the Credentials page under API keys.  
(Remember to restrict the API key before using it in production.)



- After paste this key in project build.gradle file

Goto GOOGLE\_ANDROID\_API\_KEY

```
resValue "string", "GOOGLE_ANDROID_API_KEY", "AIzaSyDNFD-7eoMliwhzW7U45WMeHyXbXW6rQfQ"  
//paste your google key here
```

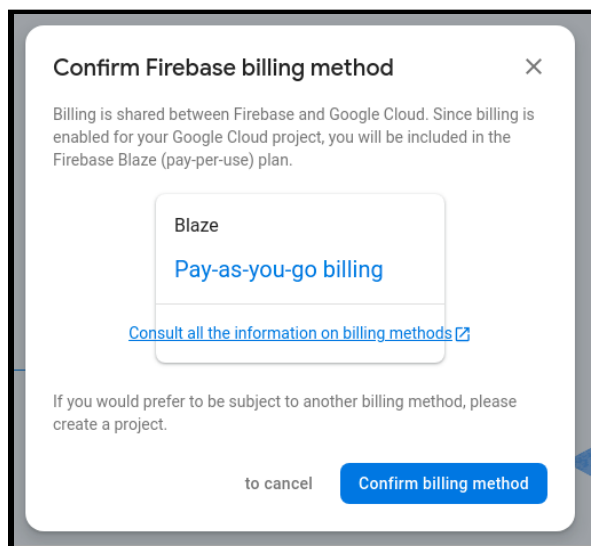
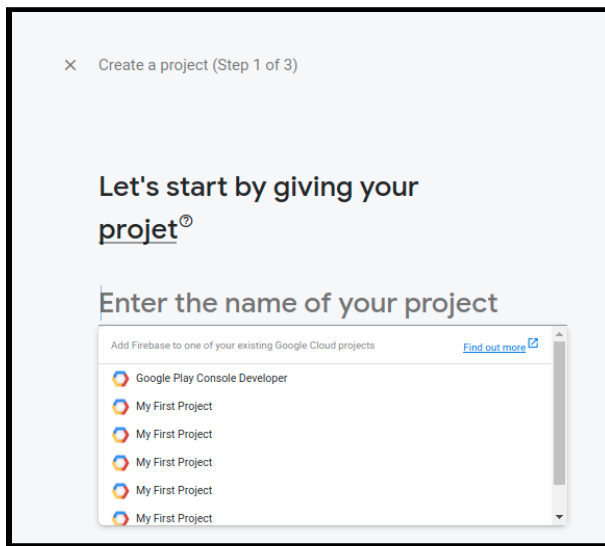
## ★ Firebase Account

Firebase provides many utilities like cloud messaging, Crashalytics ,Analytics , RealTime Databases ,In-App Messaging , Dynamic Links etc.

You can learn more about firebase products from <https://firebase.google.com/>

## - Create Project in FirebaseConsole

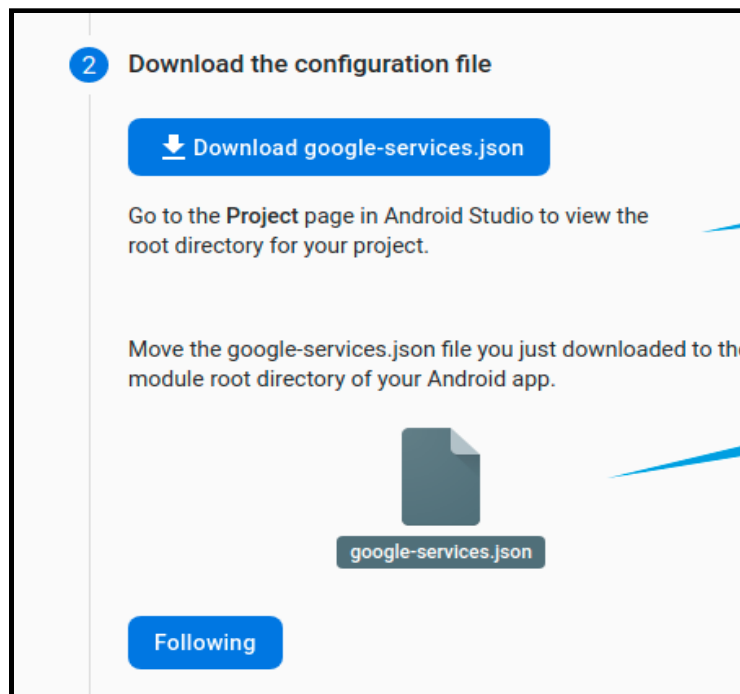
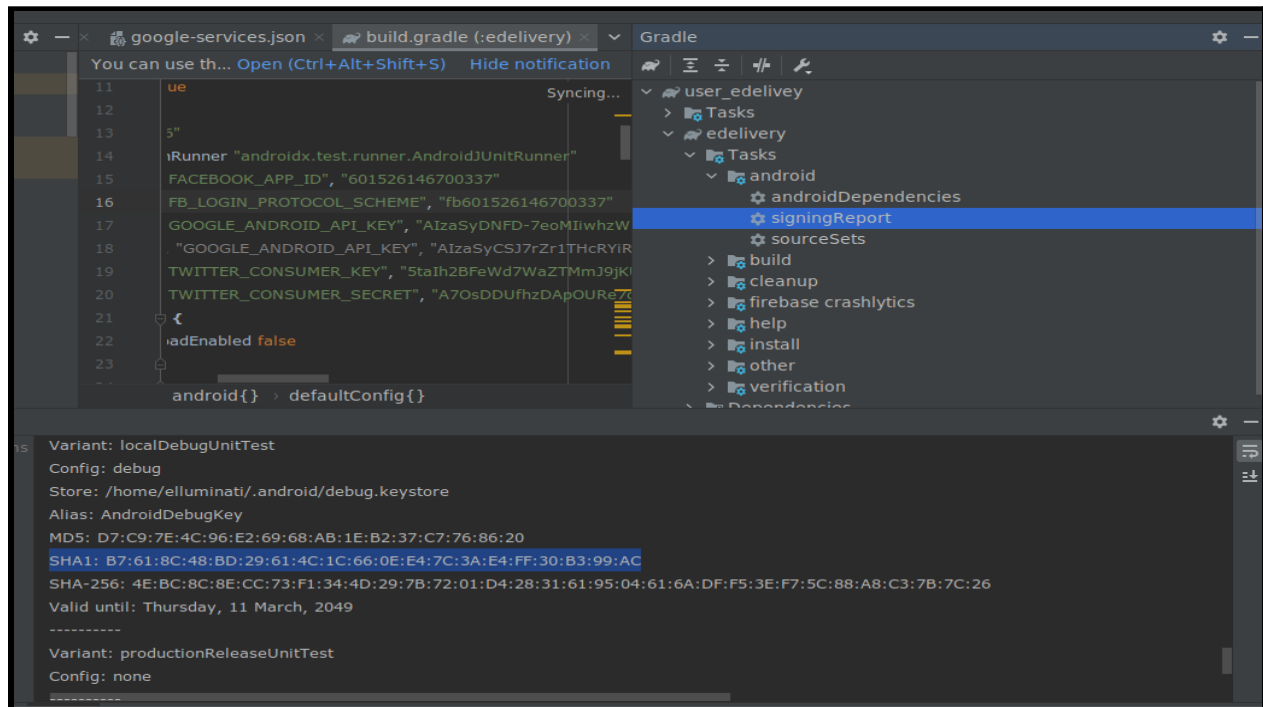
- Goto Firebase console <https://console.firebase.google.com/u/4/>
- Click Add Project
- You can see Google cloud projects you created in <https://console.cloud.google.com> here , Select Your Project and continue
- Unselect switch. You can set it up later .Continue to create project



- Confirm Your Billing Method
- In the next steps, you will be asked whether to set up Google Analytics.

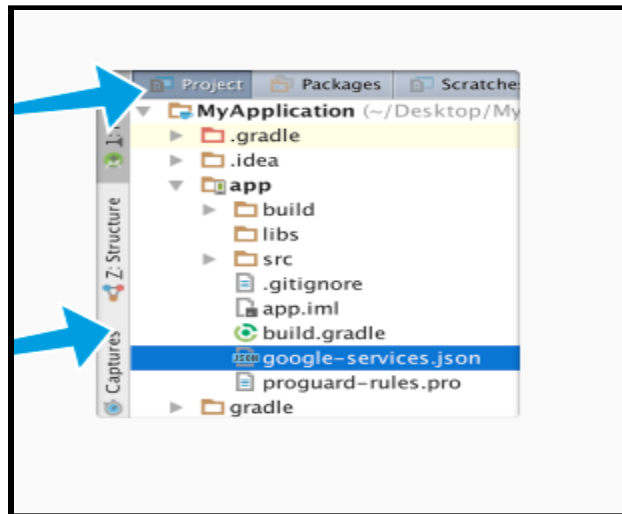


→ You can get your sha1 from here





→ Add this **google-services.json** file to the module root directory of your Android app. Refer below image.



### - Create RealTimeDatabase

Store and sync data with our NoSQL cloud database. Data is synced across all clients in real time, and remains available when your app goes offline.

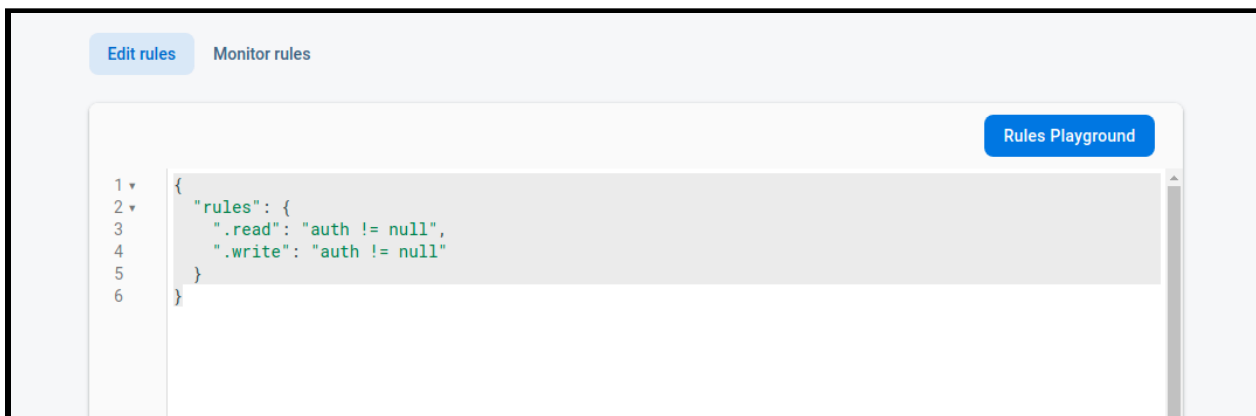
Firebase Realtime Database Security Rules determine who has read and write access to your database, how your data is structured, and what indexes exist.

For more info check this <https://firebase.google.com/docs/database>

We are using Firebase realTimeDatabase for sending, retrieving, storing chat data

→ GoTo [Firebase console](#)

→ Side menu -> Realtime Database -> Create Database -> select locked mode -click rules -> read true and write true -> click on publish



## ★ For Crashlytics:

- GoTo [Firebase console](#)
- Side menu -> Crashlytics
- You can learn how to integrate crashlytics from [here](#)



## ★ Build project

Check build variants (check which have BASE\_URL)

1. Select target device
2. Run project