//read_admin_doc_first

Travel Hour App Doc (v3.0)

20-10-20 Last updated

"It's an offline document. There is an online documentation here. We are highly recommending the online version. Because it will always be updated."

Introduction

So you are currently on the flutter beta channel. You can build this app from this channel too. If you want to build this app on the flutter stable channel, you can. It's recommended to use the stable channel. If you want to switch on the stable channel, just run **flutter channel stable** and then **flutter upgrade**. That's it. It will go back to the stable channel just like before. If you don't have **a mac device** and **an apple developer account**, you can skip iOS setups. Because you need Xcode App which is only available for mac devices and others setup requires an apple developer account.

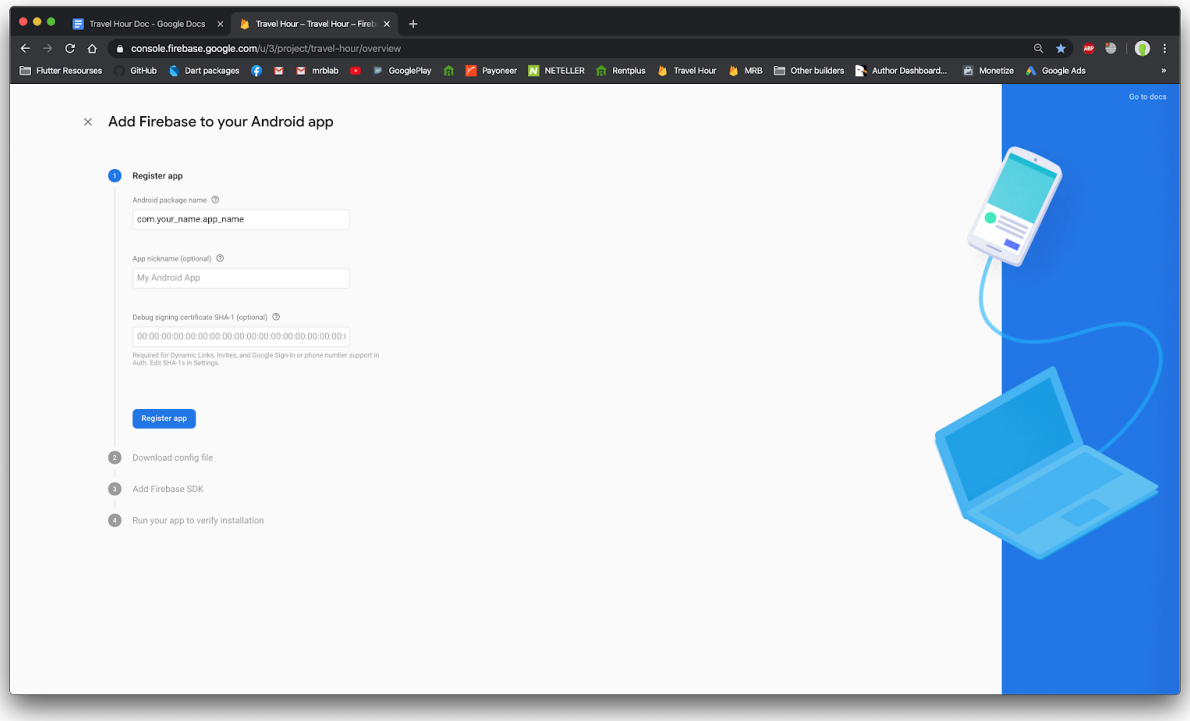
Project Setup

1. Open the **travel_hour** folder on your IDE (VSCode or Android Studio).
2. Run this following command on the terminal to get all the packages from the server.

```
flutter pub get
```

Firebase Setup for Android

1. Go to the **firebase console > Project overview page**. Click on add app and then android icon. Enter your android package name. Your package name should be like **com.your_name.your_app_name** . Like our package name is **com.mrblab.travel_hour**. You can use the same package name for android & iOS. iOS doesn't support **underspace** in the package name. So, keep in mind that if you want to use same package name for both android & iOS.



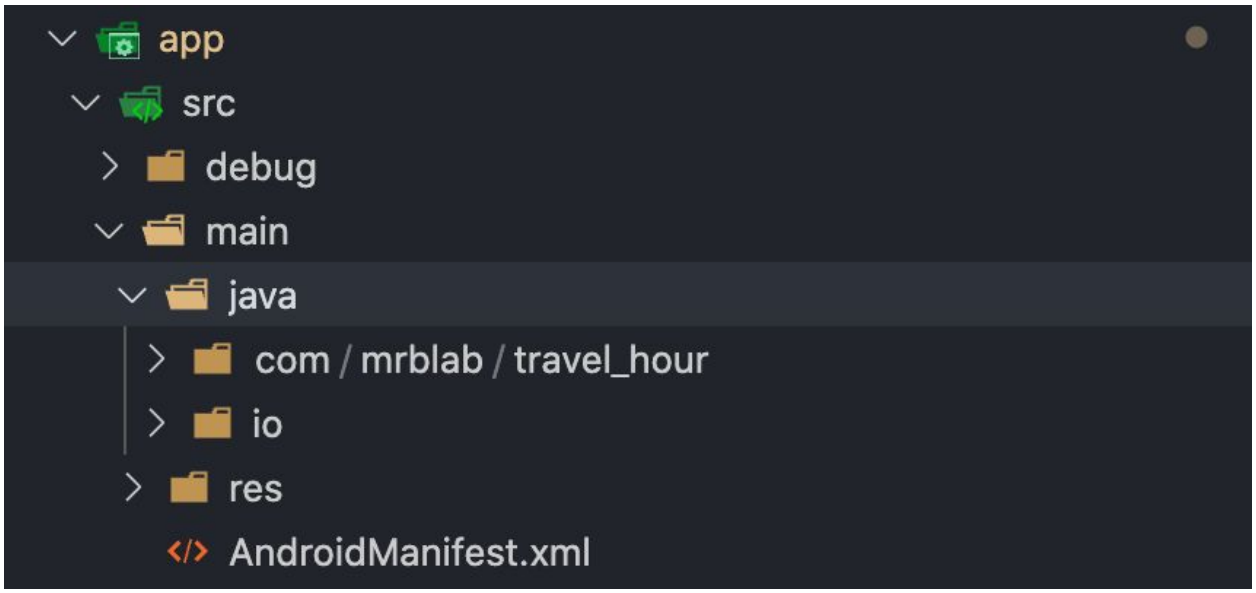
2. Click on the register app and skip other steps by clicking next.

3. Go to your IDE and now you **have to change the package name** of your app. Go to

- **android>app>build.gradle,**
- **android/app/src/debug/AndroidManifest.xml,**
- **android/app/src/main/AndroidManifest.xml,**
- **android/app/src/main/java/com/mrblab/travel_hour/MainActivity.java,**
- **android/app/src/profile/AndroidManifest.xml**

files and then find & replace **com.mrblab.travel_hour** by **your_package_name**.

4. Now you have to rename two folders. Go to **android/app/src/main/java/com** and rename **mrblab** by **your_name** and and inside this folder rename **travel_hour** by your **app_name.** Remember this should be according to your android package name.



That's it. Your android package name setup is complete.

5. Now, you need to generate **2 signing certificates** for google sign in feature.

To generate a debug certificate, run this command on your terminal from your app root directory.

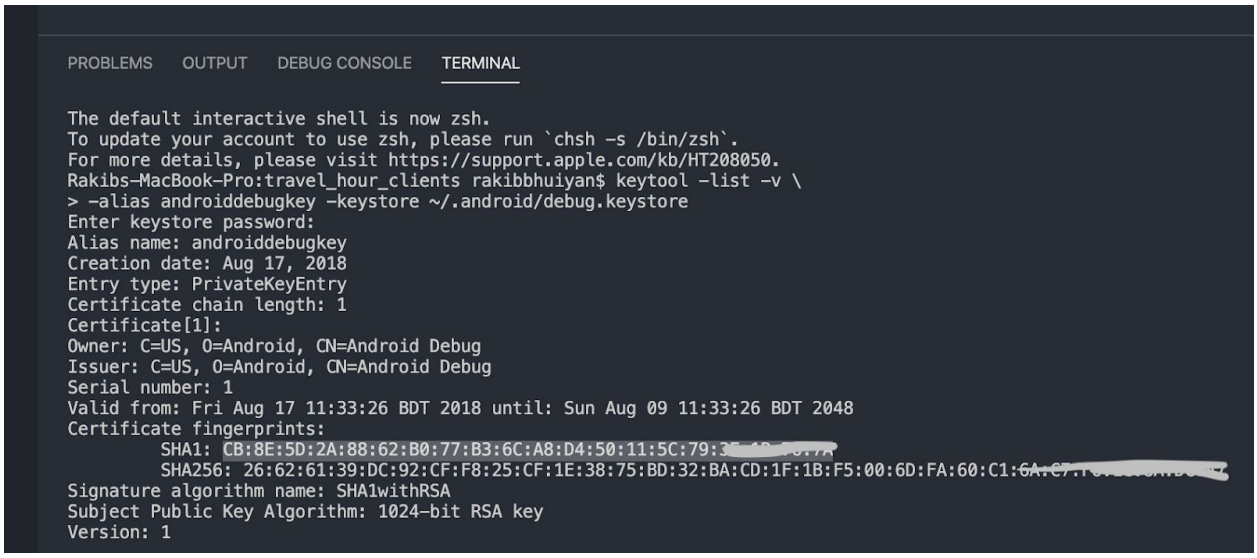
For Mac Users, run

```
keytool -list -v \
-alias androiddebugkey -keystore ~/.android/debug.keystore
```

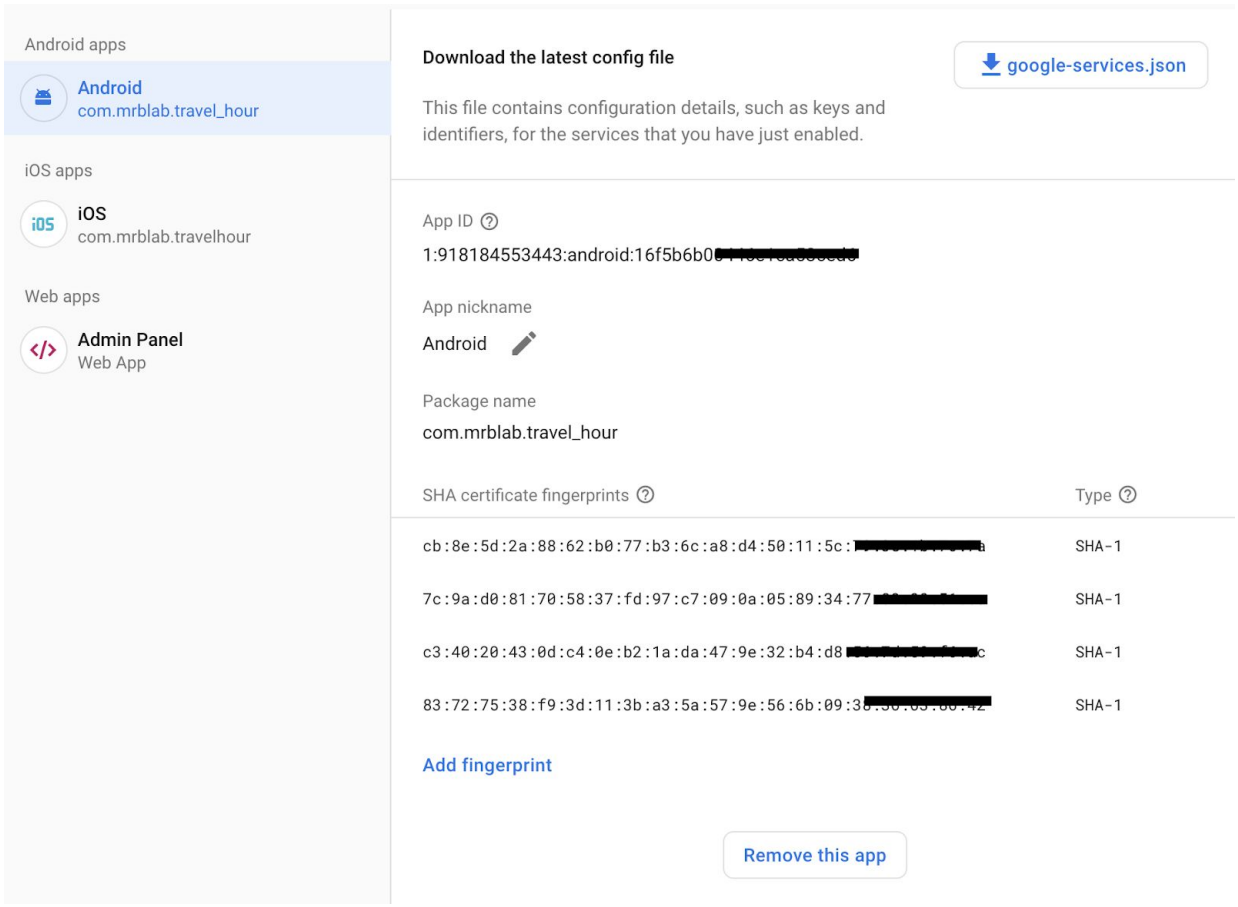
For Windows users, run

```
keytool -list -v \
-alias androiddebugkey -keystore %USERPROFILE%\.android\debug.keystore
```

Use **android** a s a debug password when the terminal asks for a password.



Copy the SHA1 certificate code and go to **Firebase Console>Your Project>Project Settings** and click on the **android icon** and then add the **SHA1** code by clicking **add fingerprint** button.



To generate a release certificate, You have to generate a keystore file. To generate a keystore file, run this command below from the root of your project directory.

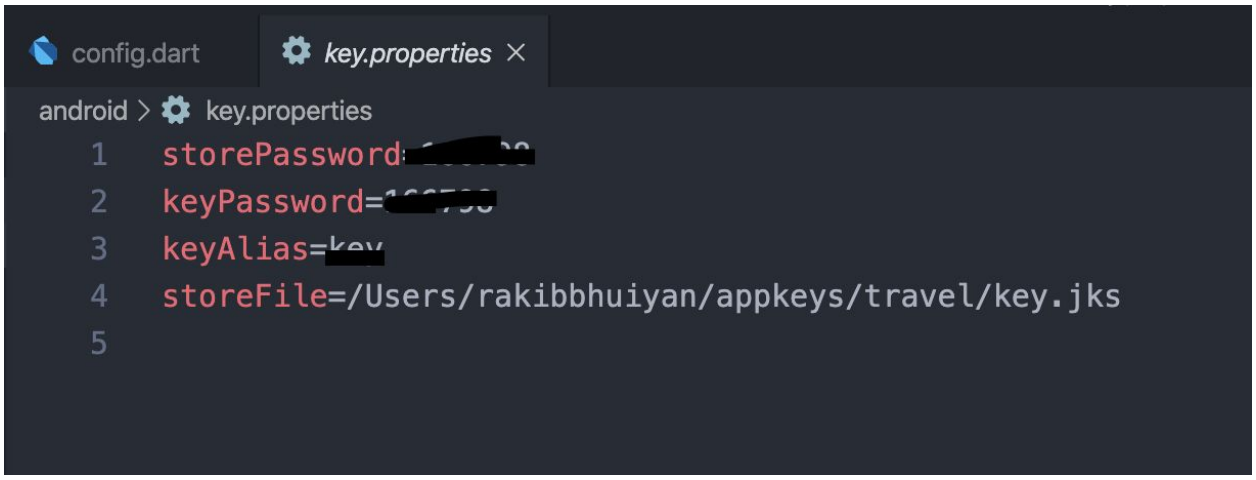
For Mac users, run

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

For Windows users, run

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

Enter your details and remember **alias key** name and **password**. After this, you will get a **.jks** keystore file. Locate this file and move the file into the **android/app** folder and copy the path by right clicking on the **key.jks** file Then go to **android/key.properties** file and replace the path of the keystore file of yours. Then also replace the **password and key alias name** which you have inputted to generate the keystore file.



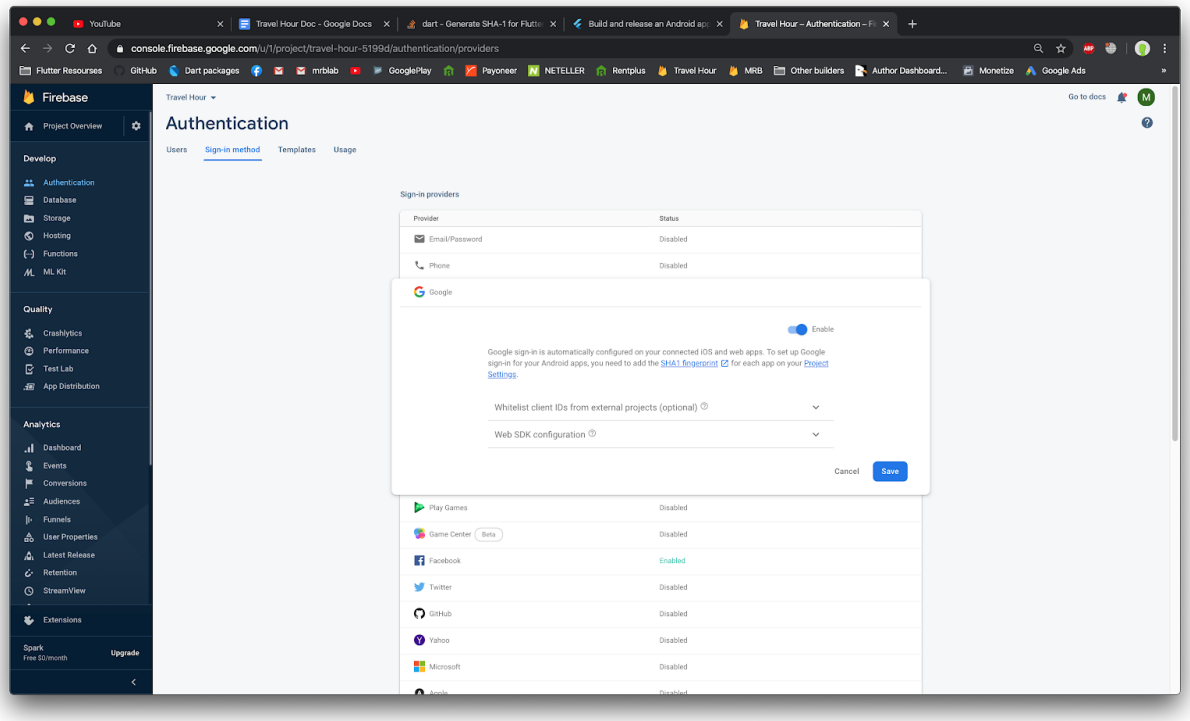
Now you can generate a release certificate, To do that, run with replacing your **alias_name** and **keystore_location**.

```
keytool -list -v -keystore keystore_location -alias alias_name
```

After that you will get a **SHA1** code. Copy that code and add to your firebase console project settings where you previously added a debug **SHA1** code.

Google Sign In Setup :

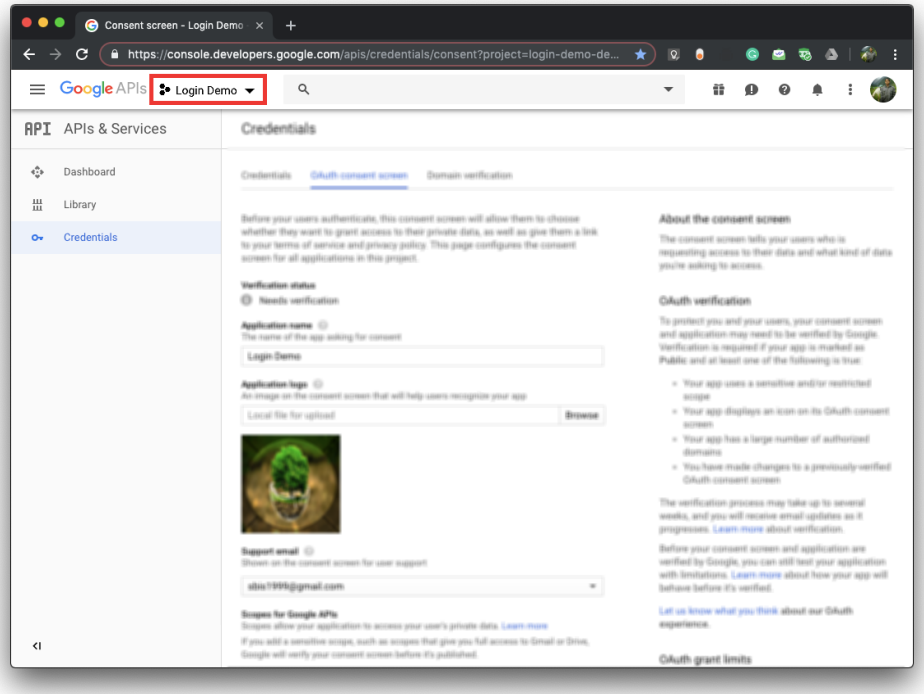
5. Now you have to set up google sign in. To do that, Go to **firebase console>your project>authentication>Sign-in-method** and click on **google** and the enable and save it.



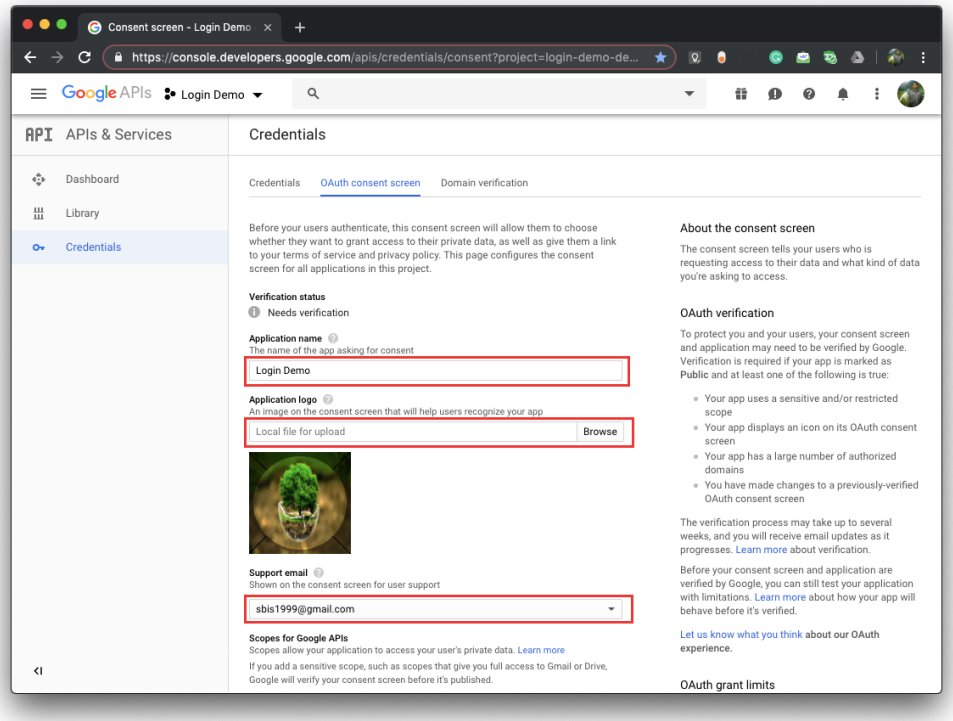
6. You have to configure some stuff for google sign in. Go to this [url](#).

7. Make sure you are signed in with the same account with which you have created the Firebase project.

8. Also, make sure that on the top-left corner, your project is selected for which you are filling this consent.

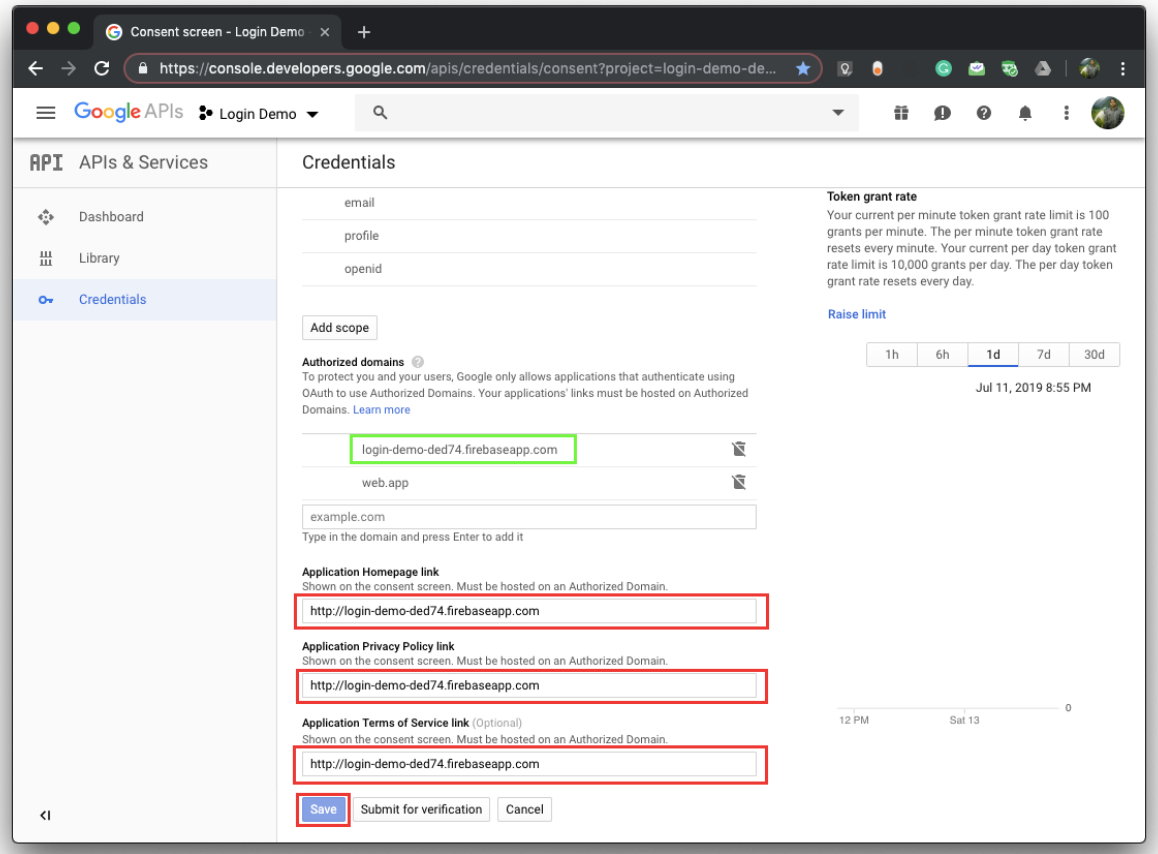


9. Go to Credentials → OAuth consent screen tab and start filling the form.
10. Enter “Application name”, “Application logo” & “Support email”.



11. Then, scroll down and fill the “Application Homepage link”, “Application Privacy Policy link” and “Application Terms of Services link”.

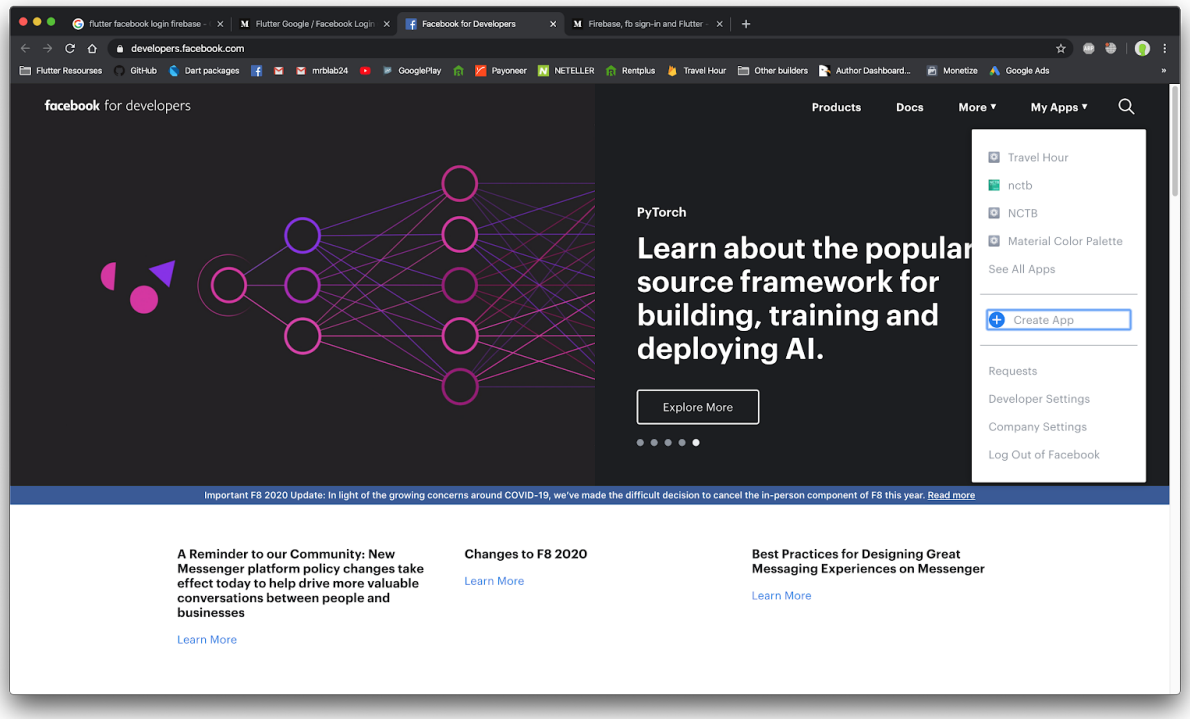
12. In all these places, you have to enter the same link starting with **http://** then your app domain name which I have marked with green below.



13. Click on Save. That's it. You have completed your google signin setup.

Facebook Sign In Setup :

14. Now you have set up facebook sign in. To do that, Go to this url.
15. Go My apps > Create App.
16. Enter App name and email and go to Dashboard tab.



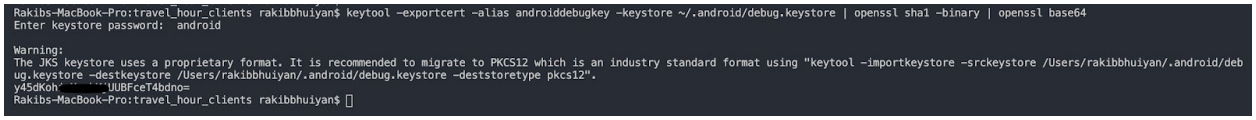
17. Scroll down on the right pane until you reach '**Add a product**', select **Facebook Login**
18. You will be redirected to the quick start page.
19. Select **Android**. Skip 1 & 2.
20. Enter **your_package_name** in the package option and enter **your_package_name.MainActivity** in the activity option.
21. For the next step, you need to generate two **hash ids**. To do that, run the following commands on terminal. For Mac Users,

```
keytool -exportcert -alias androiddebugkey -keystore ~/.android/debug.keystore | openssl sha1 -binary | openssl base64
```

For Window users,

```
keytool -exportcert -alias androiddebugkey -keystore
"C:\Users\USERNAME\.android\debug.keystore" |
"PATH_TO_OPENSSL_LIBRARY\bin\openssl" sha1 -binary |
"PATH_TO_OPENSSL_LIBRARY\bin\openssl" base64
```

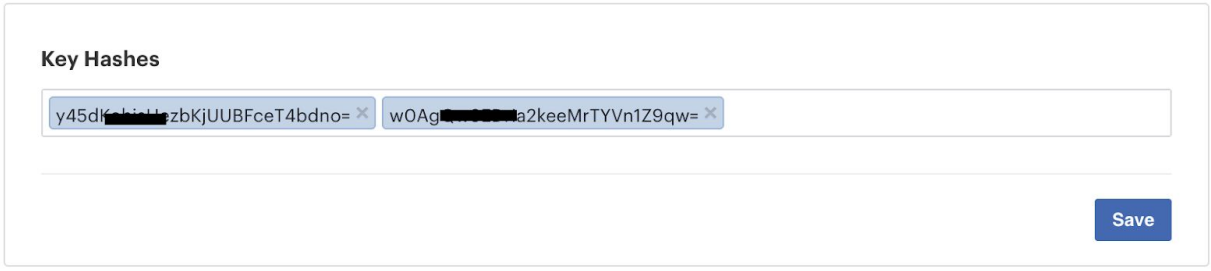
Use **android** as a debug password. After that you will get a **hash id** like this.



22. For release hash id, run this following command by replacing your **alias key name** and **keystore location**. You can get these from your **android/key.properties** file.

```
keytool -exportcert -alias YOUR_RELEASE_KEY_ALIAS -keystore YOUR_RELEASE_KEY_PATH
| openssl sha1 -binary | openssl base64
```

23. After that you will get another **hash id**. Now, copy and paste them in the next steps of facebook developer site. Like this,



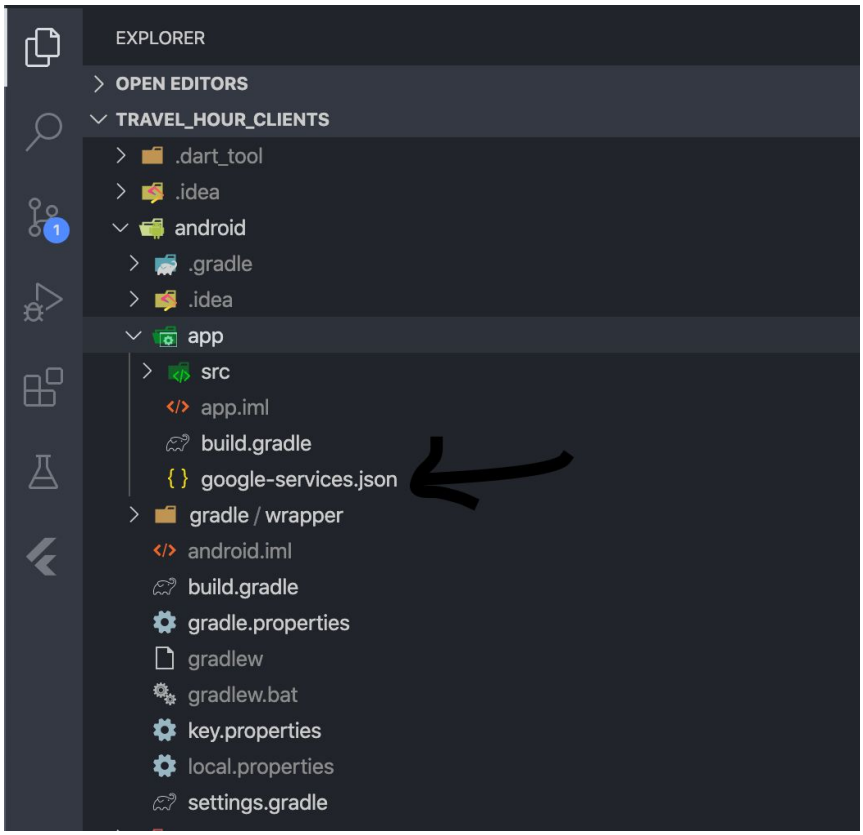
24. Skip all the steps by clicking next.

25. Now go to **settings** tab & copy both **app id** and app **secret key**.

26. Now go to firebase console > your project > authentication > Sign-in-method and click on **facebook**, enable it and paste both app id and app secret key and save it.

27. Now go to project settings and click on **android icon** and download **google-service.json** file.

28. Now go to **android/app** directory and paste the **google-service.json** file here.

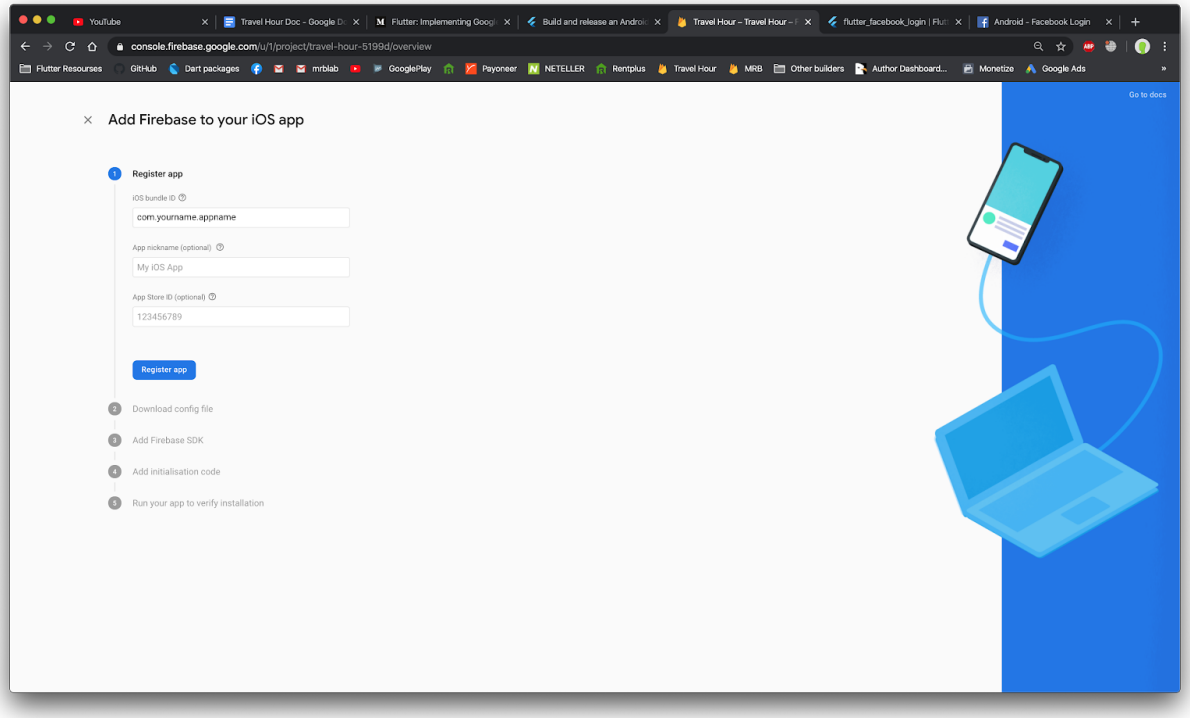


29. Now go to **android/app/src/main/res/values/strings.xml** this directory and change **the app name**, **app_id** and **fb+app_id**.

30. That's it. Android setup for Firebase database, Google Sign in & Facebook login setup is complete.

Firebase Setup for iOS

1. Go to the firebase console > Project overview page. Click on **add app** and then **android icon**. Enter **your package name**. You can use the same package name that you have used for android.



2. Click on the **register app** and skip others by clicking next.

3. Now go to project settings and click on ios and download the

GoogleService-info.plist file.

4. The go to **ios/Runner** directory and paste the file here.

5. Now, Open **iOS folder** on Xcode by right clicking on iOS folder from VSCode or Android Studio and go to runner folder and move the **GoogleService-info.plist** file here. You will get a popup and click yes or confirm the popup message.

5. Now, open the **GoogleService-info.plist** file from your IDE or from xcode Xcode and copy the **REVERSED_CLIENT_ID**. (See the picture below)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>CLIENT_ID</key>
    <string>918184553443-fl09a8l241nk6eslp4c5lcoaalkjj8d3.apps.googleusercontent.com</string>
    <key>REVERSED_CLIENT_ID</key>
    <string>com.googleusercontent.apps.918184553443-fl09a8l241nk6eslp[redacted]</string>
    <key>ANDROID_CLIENT_ID</key>
    <string>918184553443-1tal16hutlomnip48g42k3am93re1j7a.apps.googleusercontent.com</string>
    <key>API_KEY</key>
    <string>AIzaSyBdctrSAqjNZFTjh_WIP83Mtcksq0qb5Zg</string>
    <key>GCM_SENDER_ID</key>
    <string>918184553443</string>
    <key>PLIST_VERSION</key>
    <string>1</string>
    <key>BUNDLE_ID</key>
    <string>com.mrblab.travelhour</string>
    <key>PROJECT_ID</key>
    <string>travel-hour-5199d</string>
    <key>STORAGE_BUCKET</key>
    <string>travel-hour-5199d.appspot.com</string>
    <key>IS_ADS_ENABLED</key>
    <false></false>
```

6. Go to **ios/Runner/Info.plist** file and replace the **REVERSED_CLIENT_ID** here.

```xml
<key>CFBundleURLTypes</key>
<array>
    <dict>
        <key>CFBundleURLSchemes</key>
        <array>
            <string>fb19318634199******</string>  <!-- fb+app_id -->
        </array>
    </dict>
    <dict>
        <key>CFBundleTypeRole</key>
        <string>Editor</string>
        <key>CFBundleURLSchemes</key>
        <array>
            <string>com.googleusercontent.apps.918184553443-fl09a8l241nk6eslp4c5l******</string> <!-- google reversed client id -->
        </array>
    </dict>
</array>
<key>CFBundleVersion</key>
<string>$(FLUTTER_BUILD_NUMBER)</string>
<key>FacebookAppID</key>
<string>1931863419*****</string>  <!-- fb app id -->
<key>FacebookDisplayName</key>
<string>Travel Hour</string>   <!-- app name on facebook -->
<key>FirebaseAppDelegateProxyEnabled</key>
<false/>
<key>GADApplicationIdentifier</key>
<string>ca-app-pub-3940256099942544~3347511713</string>
<key>LSApplicationQueriesSchemes</key>
```

So, your Firebase & Google Sign In for iOS setup is complete.

Facebook Sign In for iOS :

8. Now go to developers.facebook.com again and navigate to your project and click facebook login > quick setup > ios icon.

9. Skip 1 & go to step 2.

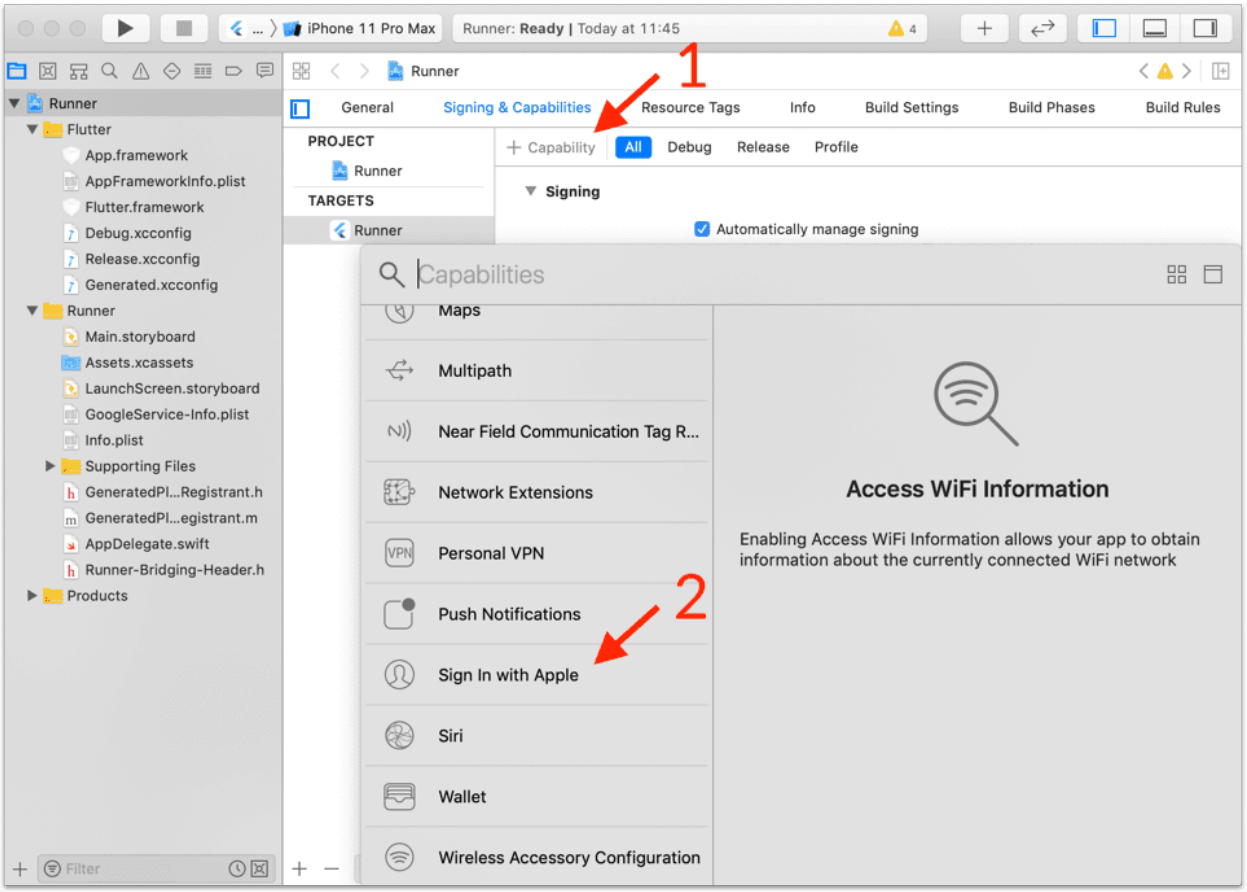
10.Enter **your package name** in the **bundle ID** option & skip others by clicking next.

11. That's it. iOS setup for Facebook Sign In is complete.

Sign In with Apple (Only for iOS)

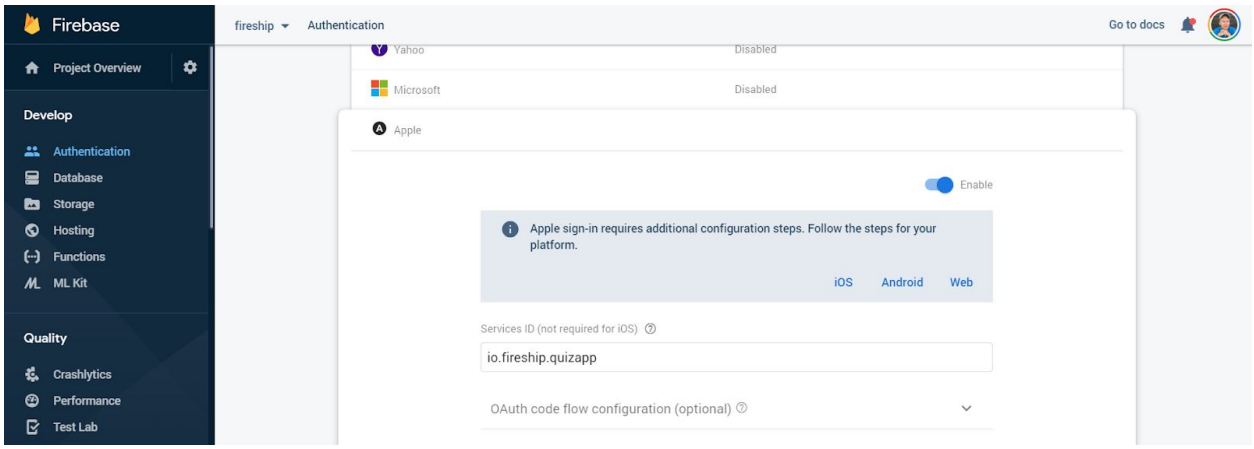
To do that, you need an paid apple developer account and xcode app on your mac.

1. From your IDE, Right click on the **ios folder** and click on **open on xcode** and then go to **runner > sign in & capability** tab.



2. Add **Sign in with Apple**. We already did this in the project. If this is not available in the project, do this by yourself.

3. No go to your firebase console > your project > authentication page and enable apple sign in option.



4. That's it. One more thing, when you configure Firebase push notification for iOS in the next step, make sure you have also tik on the **Sign In with Apple** option in the identifier on apple developer page.

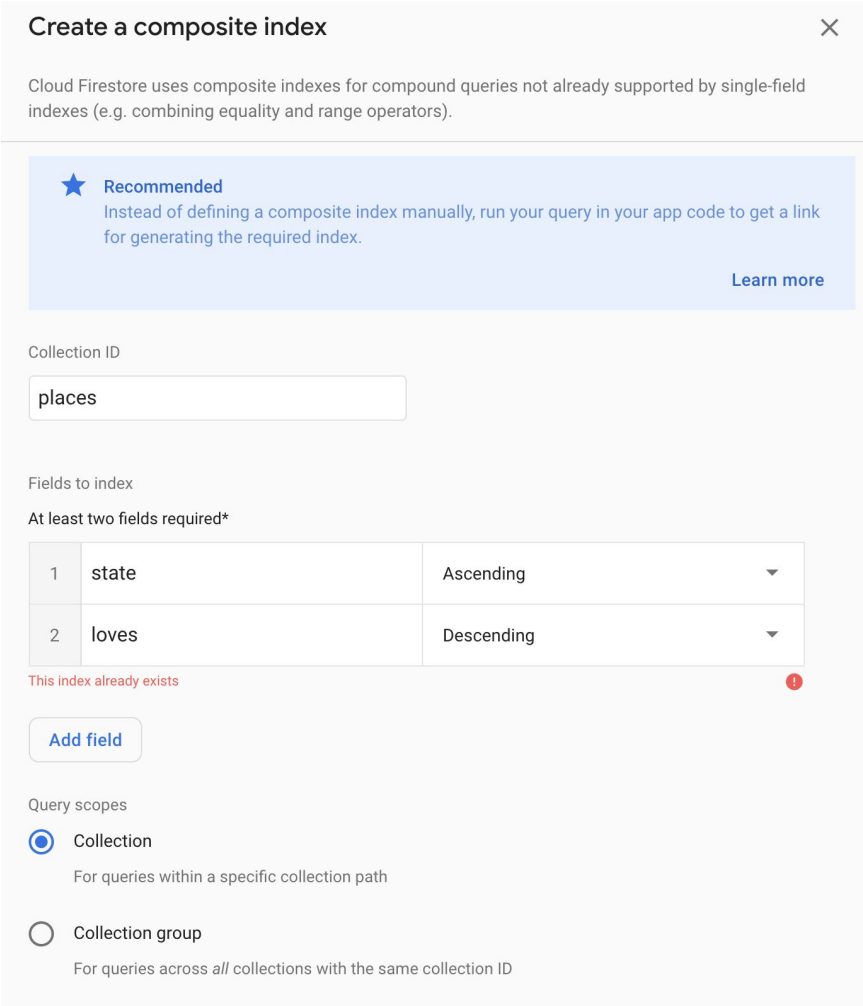
Firebase Database Rules Setup :

Go to your firebase console > project overview > database > cloud firestore > rules and then edit the rules like this and click publish. That's it.

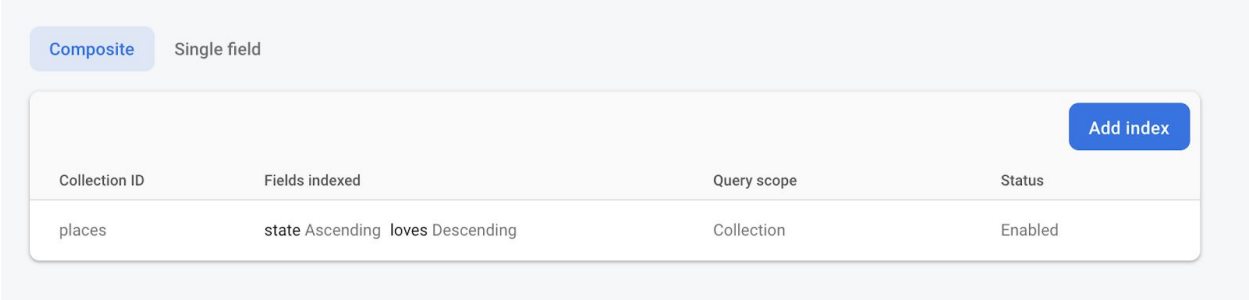
```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if true;
    }
  }
}
```

Firebase Database Index Setup :

Now go to the right tab (Indexes) of the same page. Click on **Add Index** and then create an index by following the picture below.



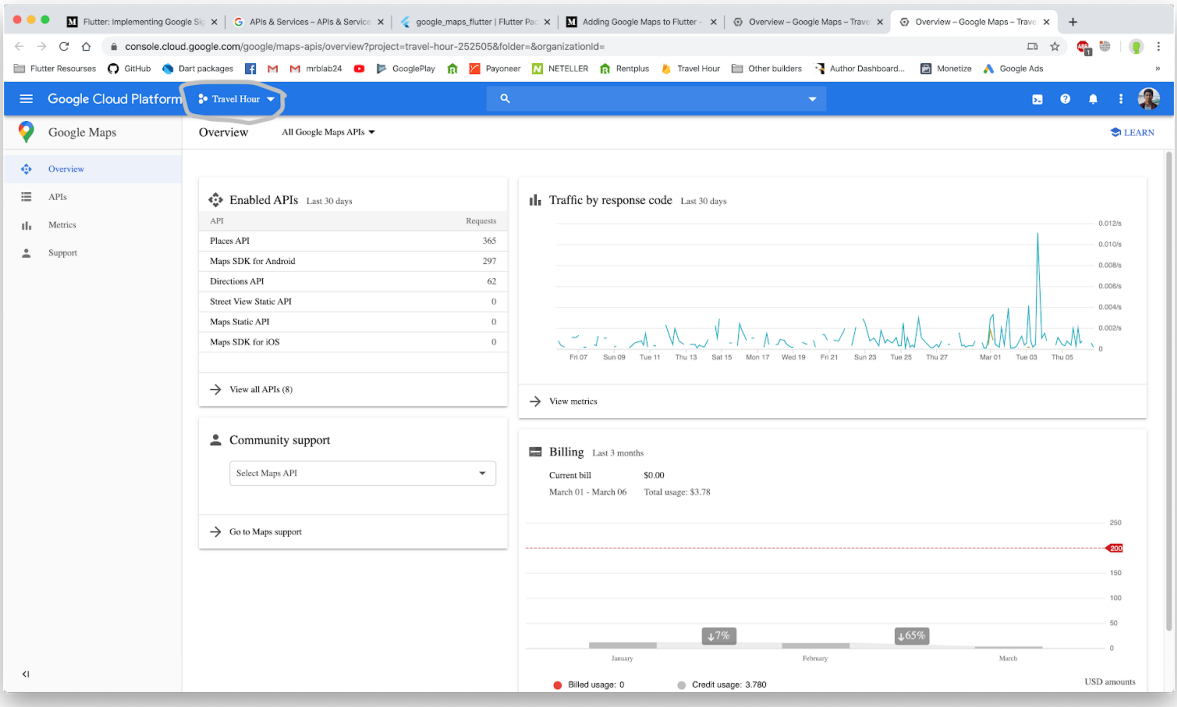
After setup, the index will look like this picture below.



That's it. Your database setup is complete.

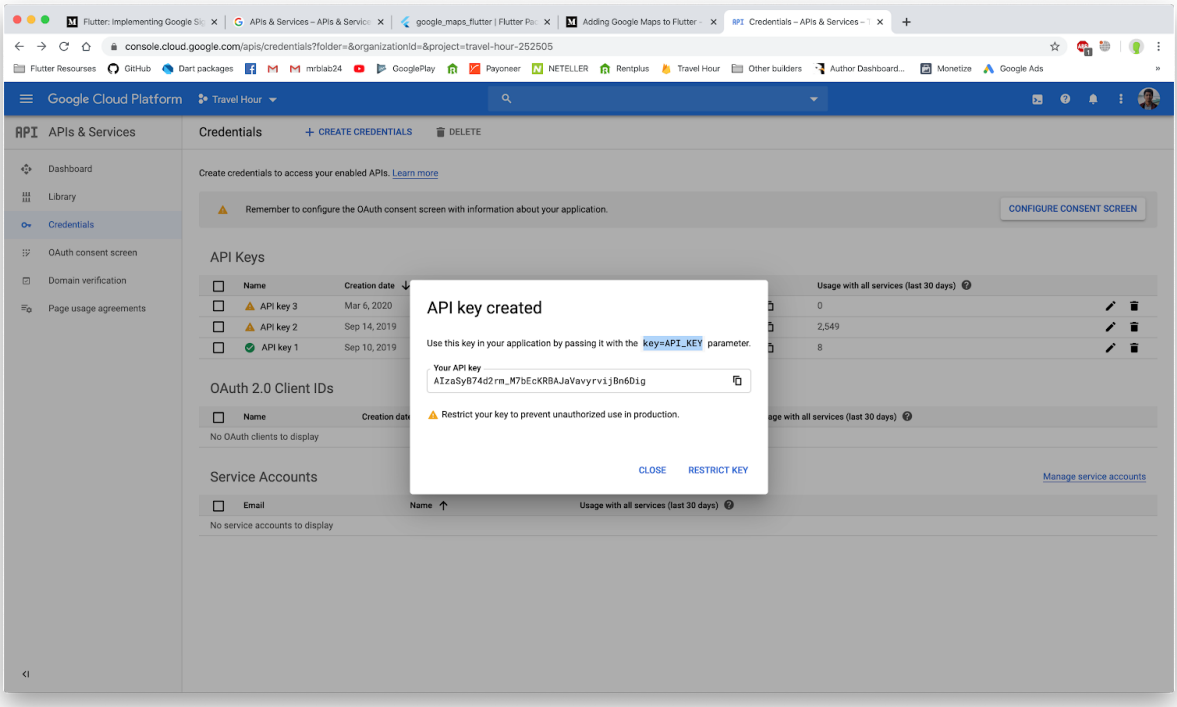
Google Map Setup

1. First of all, go to this [url](#).
2. Make sure you have selected your project.



3. Go to the APIs tab and enable these (4) APIs.
    - Directions API
    - Places API
    - Maps SDK for Android
    - Maps SDK for iOS

4. Click on the menu button (Top Left) and select APIs & Services.
5. Go to the Credentials Tabs and click create credential then click **API key**.



6. Copy that **API key** and go to **lib/config/config.dart** file and replace with your google map **API key**.

7. Most importantly, you have to add billing on your Google Cloud Console. If not, Google Map API key won't work.

Google Map Android Setup

1. Go to **android/app/src/main/AndroidManifest.xml** and specify your API key here.

```
</activity>
    <meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="AIzaSyA-fE7BtlehhXxsO_sWnHpa********"/>   <!-- Paste your Google API key here-->
```

Google Map iOS Setup

1. Go to **ios/Runner/AppDelegate.m** and specify your **google map API key** here.

```objc
#include "AppDelegate.h"
#include "GeneratedPluginRegistrant.h"
#import "GoogleMaps/GoogleMaps.h"

@implementation AppDelegate

- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
  [GeneratedPluginRegistrant registerWithRegistry:self];
  // Override point for customization after application launch.
  [GMSServices provideAPIKey: @"AIzaSyA-fE7BtlehhXxsO_sWnHpayWen*******"];     // Paste your google map API key here
  return [super application:application didFinishLaunchingWithOptions:launchOptions];
}

@end
```

Finally your google map setup is complete.

Firebase Push Notifications Setup :

1. For Android, You don't have to do anything. We already integrated the procedures in the project.

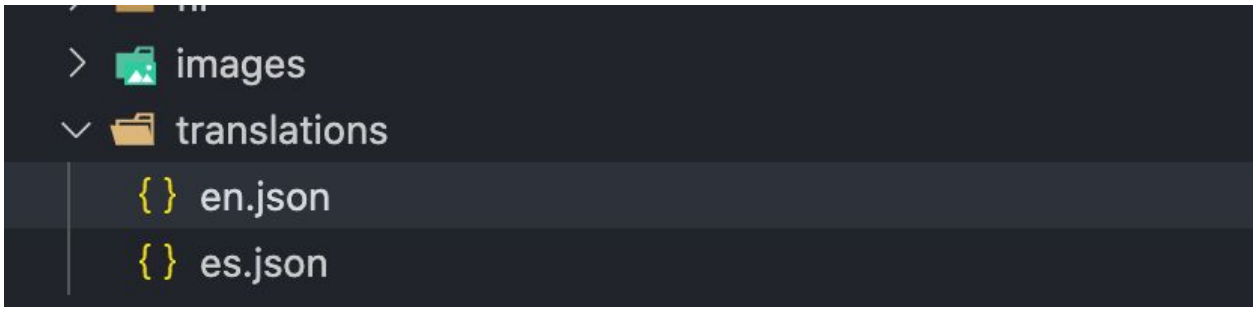
2. For iOS, Go to this link and follow the instructions. This is a well written doc from Flutter Team.

Multi-language setup :

You can skip this setup now. This is not a mandatory setup to run this app.

So, we have used two languages in this app. English & Spanish. You can as much you can. We are assuming that you want to add your own country language. You need to know about your two letter language code. Like, English language code is **en** and Spanish language code is **es** . You can search for your language code on google.

1. First go to the **assets/translations** folder from your IDE. Add a .json file here with **your_language_code.json** name. No go to **assets/translations/en.json** file and copy everything from this file and paste to **your_language_code.json** file.



2. Now, Rename the all right side strings. Do not edit left side strings. These are the keys. Look at the **es.json** file and you will understand what to do.

3. No go to **lib/main.dart** file and add your language code to **supportedLocals**.

```
Run | Debug
void main()async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(
    EasyLocalization(
      supportedLocales: [Locale('en'), Locale('es')],
      path: 'assets/translations',
      fallbackLocale: Locale('en'),
      startLocale: Locale('en'),
      useOnlyLangCode: true,
      child: MyApp(),
    ) // EasyLocalization
  );
}
```

Your code will be look like this :

```
supportedLocales: [Locale('en'), Locale('es'), Locale('your_language_code')],
```

4. You can edit the **startLocale** by replacing **en** by **your_language_code** if you want to add your default language to your language.

5. For Android, you don't have to do anything.
6. For iOS, go to **ios/Runner/Info.plist** and add your language code in a string.

```
11    <key>CFBundleInfoDictionaryVersion</key>
12    <string>6.0</string>
13    <key>CFBundleLocalizations</key>
14    <array>
15        <string>en</string>
16        <string>es</string>
17    </array>
18    <key>CFBundleName</key>
19    <string>Travel Hour</string>
20    <key>CFBundlePackageType</key>
21    <string>APPL</string>
```

7. Add **<string>your_language_code</string>** inside the array.
8. Now go to lib/config/config.dart and add your language name in the list with comma.

```
41  //Language Setup
42
43  final List<String> languages = [
44    'English',
45    'Spanish'
46  ];
47
```

9. That's it. Your multi-language setup is complete. You can add as many languages you want by following these steps.

Ads Setup :

You can skip this setup for now and can configure later. We have added test unit ids to test ads.

So, We have used both admob and facebook ads in this app. But you can use only one at a time. Either admob or facebook ads. Admob ads applied by default. So, if you want to add admob ads, you just have to put your ads id. That's it. We recommend you to use admob ads because they provide higher revenue than any other ads networks. Use facebook ads if only your admob account is suspended. By the way, that was just a suggestion from us. The choice is yours. We have used only **interstitial ads** with this app which is most suitable for this type of apps and which revenue is higher. So you just need an interstitial ad unit id.

1. You can control ads from the admin panel. We have added an option to turn of/on ads at any time you want.

2. You can also set after how many times user clicks to show an ad. In default we have added clicksCount for **5 times**. That means when a user goes to a screen and if the **clicksCount multiple by 5**, then an ad will be shown. Then means after 5,10,15...clicks an ad will be shown if the ads loaded properly. We have applied ads into 5 screens.

   place_details.dart, blog_details.dart, guide.dart, hotel.dart, restaurant.dart.

   That means ClickCount will be applied only for these 5 screens. To edit the clickCount times, go to **lib/config/config.dart** file and edit the amount.

```
50
51  final int userClicksAmountsToShowEachAd  = 5;
52
```

Admob Setup :

1. Go to **android/app/src/main/AndroidManifest.xml** file and replace with your **admob app id** of yours which you will get from your admob account.

```
78        <!---admob section-->
79
80        <meta-data
81            android:name="com.google.android.gms.ads.APPLICATION_ID"
82            android:value="ca-app-pub-3940256099942544~3347511713"/>
83
```

2. Go to **lib/config/config.dart** file and replace **admobAppId** and **admobInterstitialAdId** value by yours.

```
53    //-- admob ads --
54    final String admobAppId = 'ca-app-pub-3940256099942544~3347511713';
55    final String admobInterstitialAdId = 'ca-app-pub-3940256099942544/1033173712';
56
```

That's it. Your Admob setup is complete.

Facebook Ads Setup :

Ignore this if you have added admob.

Before we do this, we need to inform you that iOS won't support facebook ads in this version. We have found some problems and issues with the fb ads package and decided to disable this for iOS. So you have to disable **facebook_audience_network** package first from the **pubspec.yaml** file.

1. First, as we previously said that you need to remove admob first to apply fb ads on your app. To do that, go to pubspec.yaml file and remove **firebase_admob** line enable **facebook_audience_network** and then run **flutter pub get** on the terminal.

2. Now, go to the **android/app/src/main/AndroidManifest.xml** and remove these 3 lines.

```
78      <!---admob section-->
79
80      <meta-data
81          android:name="com.google.android.gms.ads.APPLICATION_ID"
82          android:value="ca-app-pub-3940256099942544~3347511713"/>
83
```

3. Now go to **lib/config/config.dart** file and replace the values of **fbInterstitialAdIdAndroid** by yours which you will get from facebook monetization manager.

```
57  //fb ads
58  final String fbInterstitalAdIDAndroid = '193186341991913_351138796196666';
59  //final String fbInterstitalAdIDiOS = '193186341991913_351139692863243';
60
```

4. Then go to **lib/blocs/ads_bloc.dart** file and and remove admob ads section and enable fb ads section. To enable a disabled line on VSCode, first select line and then click **Command+/** for mac and **Controll+/** for windows. Your ads_boc.dart should be look like these : (You can ignore the disabled lines)

```dart
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:facebook_audience_network/ad/ad_interstitial.dart';
import 'package:facebook_audience_network/facebook_audience_network.dart';
import 'package:flutter/foundation.dart';
import 'package:travel_hour/config/config.dart';

class AdsBloc extends ChangeNotifier {




  int _clickCounter = 0;
  int get clickCounter => _clickCounter;

  bool _adsEnabled = false;
  bool get adsEnabled => _adsEnabled;


  Future checkAdsEnable () async {
    FirebaseFirestore firestore = FirebaseFirestore.instance;
    await firestore.collection('admin').doc('ads').get()
      .then((DocumentSnapshot snap) {
      bool _enabled = snap.data()['ads_enabled'];
      _adsEnabled = _enabled;
      print(_adsEnabled);

      notifyListeners();
    }).catchError((e){
      print('error : $e');
    });
  }

  void increaseClickCounter(){
    _clickCounter ++;
    print('Clicks : $_clickCounter');
    notifyListeners();
  }


  void enableAds (){
    if(_adsEnabled == true){
      //loadAdmobInterstitialAd();
      loadFbAd();
    }
  }


  initiateAds (){
    increaseClickCounter();
    //showAdmobInterstitialAd();
    showFbAdd();
  }
```

```
117  bool _fbadloaded = false;
118  bool get fbadloaded => _fbadloaded;
119
120
121  Future loadFbAd() async{
122    print('loading');
123    FacebookInterstitialAd.loadInterstitialAd(
124      placementId:
125      Config().fbInterstitalAdIDAndroid,
126      //Platform.isAndroid ? Config().fbInterstitalAdIDAndroid : Config().fbInters
127      listener: (result, value) {
128        print(result);
129        if (result == InterstitialAdResult.LOADED){
130          _fbadloaded = true;
131          print('ads loaded');
132          notifyListeners();
133        }else if(result == InterstitialAdResult.DISMISSED && value["invalidated"]
134          _fbadloaded = false;
135          print('ads dismissed');
136          loadFbAd();
137          notifyListeners();
138        }
139
140      }
141    );
142  }
143
144  Future showFbAdd() async{
145    if(_clickCounter % Config().userClicksAmountsToShowEachAd == 0){
146      if(_fbadloaded == true){
147        print('showing');
148      await FacebookInterstitialAd.showInterstitialAd();
149      _fbadloaded = false;
150      notifyListeners();
151    }
152    }
153  }
154
155  Future destroyFbAd() async{
156    if (_fbadloaded == true) {
157      FacebookInterstitialAd.destroyInterstitialAd();
158      _fbadloaded = false;
159      notifyListeners();
160    }
161  }
162
163
164  initFbAd () async{
165    await FacebookAudienceNetwork.init();
166  }
167  // fb ads ---------- end ---------
168
```

5. That's it. Your fb ads setup is successful.

Other Setup

1. Go to **lib/models/config.dart** file and change all of your details.
2. Change App name,
3. Country name,
4. To change the **splash icon**, you have to upload your own splash icon. The icon should be in the **.png** format and make sure you have renamed it to **splash** . Go to **lib/assets/images** folder and drop the icon here and replace with our icon.
5. mapAPI key you have changed before if not change again,
6. Support Email,
7. Privacy policy url (Ignore this for now if you are not going to release now).
8. Your website url (same as before)
9. iOS app ID (Only for iOS and you can ignore this for now)
10. **Special State 1& 2** change is mandatory. You have to add two State/Category name that you have already added in the admin panel.
11. Replace the latitude & longitude of the **initialCameraPosition** by your Country Latitude & Longitude.

```
class Config {
  final String appName = 'Travel Hour';
  final String mapAPIKey = 'AIzaSyA-fE7BtlehhXxsO_sWnHpayWen9Dcg3Bs';
  final String countryName = 'Bangladesh';
  final String splashIcon = 'assets/images/splash.png';
  final String supportEmail = 'mrblab24@gmail.com';
  final String privacyPolicyUrl = 'https://www.freeprivacypolicy.com/privacy/view/0
  final String ourWebsiteUrl = 'https://codecanyon.net/user/mrblab24/portfolio';
  final String iOSAppId = '1535904378';

  final String specialState1 = 'Sylhet';
  final String specialState2 = 'Chittagong';


  final CameraPosition initialCameraPosition = CameraPosition(
    target: LatLng(23.777176,90.399452),
    zoom: 10,
  ); // CameraPosition

```

Change App Name for Android

1. Go to **android/app/src/main/AndroidManifest.xml** file and Change your app name.

```
<application
    android:name="io.flutter.app.FlutterApplication"
    android:label="Travel Hour"
    android:icon="@mipmap/ic_launcher">
```

Change App Name for iOS

1. Go to **ios/Runner/Info.plist file** and Change your app name.

```
<dict>
    <key>CFBundleDevelopmentRegion</key>
    <string>$(DEVELOPMENT_LANGUAGE)</string>
    <key>CFBundleExecutable</key>
    <string>$(EXECUTABLE_NAME)</string>
    <key>CFBundleIdentifier</key>
    <string>$(PRODUCT_BUNDLE_IDENTIFIER)</string>
    <key>CFBundleInfoDictionaryVersion</key>
    <string>6.0</string>
    <key>CFBundleName</key>
    <string>Travel Hour</string>
    <key>CFBundlePackageType</key>
    <string>APPL</string>
    <key>CFBundleShortVersionString</key>
    <string>$(FLUTTER_BUILD_NAME)</string>
```

Change the App Icon :

1. Go to the asset folder and delete the default icon **(icon.png**).
2. Now upload your app icon as png in the **assets/images** folder and rename it to **icon.png**
3. Now run the following command on the terminal,

```
flutter pub get
flutter pub run flutter_launcher_icons:main
```

That's it.. For more info, visit this site.

So Your Setup is 100% complete now. Now run this following command to clean the project.

Run the app :

```
flutter clean
```

And After that run the following command to run this app on your physical or emulator devices. (If you enabled fb ads in this project, you have to disable the **facebook_audience_network** package from **pubspec.yaml** file and procedures from the **lib/blocs/ads_bloc.dart** file)

```
flutter run
```

Test if everything is okay or not.

To Release the Android App on Google Play Store:

You have done all the things that require an android release. To Test the release android app, run the following command on the terminal.

```
flutter build apk --split-per-abi
```

You will get 3 apk files from the **build/app/output/apk/release** folder. You can test the **v7** version of the apk file. If you want to publish the app in the google play store, don't upload any of the following files. Use an **appbundle** file which is recommended by Google. To generate an appbundle, run the following command on terminal :

```
flutter build appbundle
```

After that, you will get an **.aab** file in the **build/app/output/appbundle/release** folder.

Now you can upload this .aab file to the google play store.

To Release the iOS app on App Store :

Follow the official doc from flutter team here.

That’s it. We know that you are so tired right now. Take some rest. Everything is complete now.

So, That's it. If you face any problem, please contact us. If you are not a flutter developer and can't set up the project, we will help you to customize this project. This will cost additional charge. Thanks.

If you love our work then don't forget to submit a review on envato market. Thanks

MRB Lab

Contact: mrblab24@gmail.com