

Thank you for your purchase! To ensure seamless after-sales support, please activate your product using the link below:

[Activate Now](#)

Your activation is essential for us to provide you with the best assistance. We appreciate your cooperation.

Technology Used

- **Flutter** with Dart Language for Mobile App
- **ReactJS** with Typescript for Admin Panel
- **NodeJS** with Express Framework for API
- **MySQL** for Database

Please follow the below steps to set up the project on your server.

(We have provided the steps to set up using Visual Studio Code Editor. You can use other editors also. Steps may vary based on your editor.)

1. Setup Prerequisite (If not available)

- Install Visual Studio Code (VSCode) from [this link](#)
- Install NodeJS from [this link](#) (Minimum version 16.14.0)
- Install and set up Flutter from [this link](#)
- Install MySQL from [this link](#)
(You can choose the MySQL edition based on your needs)
- Install MySQL Workbench from [this link](#) (This is optional)

2. Setup Mobile App (Technology Flutter)

- Initial steps to set up and run mobile app
 - Open the **App** folder in the VSCode
 - Run the following commands in the VSCode Terminal

```
flutter clean  
flutter pub get
```

iii. Additional steps to set up for iOS (You can skip these steps if you don't want to set up for iOS)

1. In the VSCode terminal, go to the ios directory

(using the command **cd ios**)

2. Run the following command to install pods

```
pod install
```

iv. Connect your Android or iOS device with your machine

(To run on an Apple device, you must have an Apple computer)

v. Run the following command to run on an Android or iOS device

```
flutter run
```

vi. To upgrade the Flutter version run the following command in the VSCode Terminal

(Only if your Flutter version is lower than mentioned in this document)

```
flutter upgrade
```

b. Change API base URL

After the setup of your API and Admin panel, you have to change your API base URL, for that go to the file located at **lib\utils\global.dart**

```
const Map<String, dynamic> appParameters = {  
  "LIVE": {  
    "apiUrl": "YOUR-API-URL",  
    "imageUrl": "YOUR-IMAGE-BASE-URL",  
  },  
  "DEV": {  
    "apiUrl": "YOUR-API-URL",  
    "imageUrl": "YOUR-IMAGE-BASE-URL",  
  },  
};
```

c. Change Package Name/Bundle ID

An app's package name is a unique identifier that is automatically created when you create an app. The term used for iOS apps is "**bundle ID**" and for Android apps, it is "**package name**".

i. Set Package Name for Android App

1. Change the package name in the file located at **android/app/src/main/AndroidManifest.xml**

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
  package="com.demo.matrimonyapp"  
  <queries>  
    <provider android:authorities="com.facebook.katana.provider..."  
  </queries>
```

2. Change the package name in the file located at **android/app/src/debug/AndroidManifest.xml**

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
  package="com.demo.matrimonyapp"  
  <!-- Flutter needs it to communicate with the running application  
  to allow setting breakpoints, to provide hot reload, etc.
```

3. Change Package Name in file which is located at **android/app/src/Profile/AndroidManifest.xml**

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.demo.matrimonyapp">
    <!-- Flutter needs it to communicate with the running application
         to allow setting breakpoints, to provide hot reload, etc. -->
```

4. Change the Package Name in the file which is located at **android/app/build.gradle**

```
defaultConfig {
    // TODO: Specify your own unique Application ID (https://developer.android.com/studio/b
    applicationId "com.demo.matrimonyapp"
    minSdkVersion 21
    targetSdkVersion flutter.targetSdkVersion
```

5. Change the folder structure for the below path as per your package name.

android\app\src\main\java\com\demo\matrimonyapp

Ex. If your package name is com.app.matrimony

android\app\src\main\java\com\app\matrimony

6. Change Package Name in file which is located at **android\app\src\main\java\com\demo\matrimonyapp\MainActivity.java**

```
package com.demo.matrimonyapp;

import io.flutter.embedding.android.FlutterActivity;
```

- ii. Set Bundle ID for iOS App

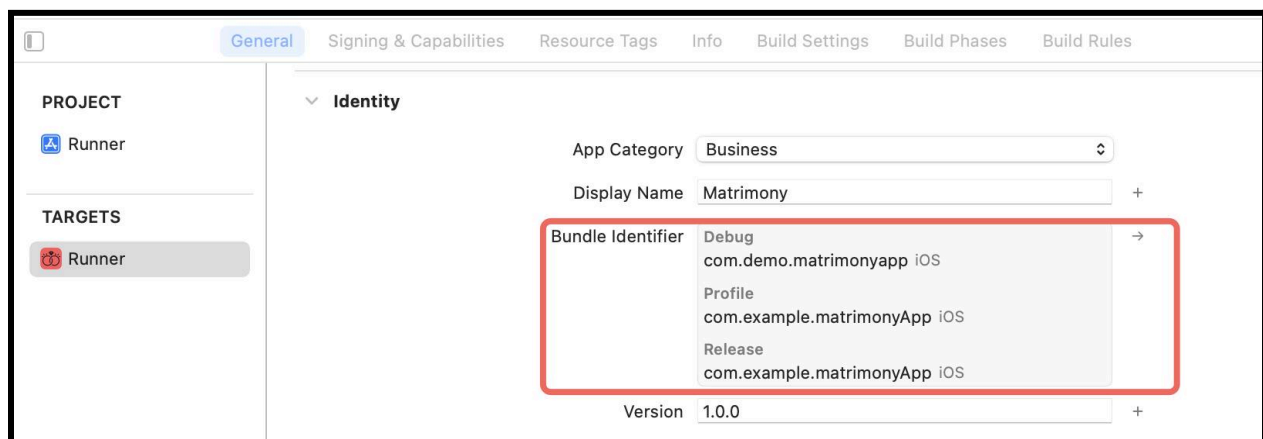
1. In VSCode

- a. Go to **ios/Runner/info.plist**
- b. Change the string of key **CFBundleIdentifier**

```
<key>CFBundleExecutable</key>
<string>$(EXECUTABLE_NAME)</string>
<key>CFBundleIdentifier</key>
<string>$(PRODUCT_BUNDLE_IDENTIFIER)</string>
<key>CFBundleInfoDictionaryVersion</key>
<string>5.0</string>
```

2. In XCode

- a. Right-click on the **iOS** folder and Choose Open in Xcode Option
- b. Click on the folder icon left side of the XCode window
- c. Select Runner.
- d. Select Target runner
- e. Go to identity
- f. Change Bundle Identifier




d. Create and set Keystore file for Android

- i. Create a keystore.jks file, if not exist, use the below command in the terminal

```
keytool -genkey -v -keystore "path\keystore.jks" -storetype JKS -keyalg RSA -keysize 2048 -validity 10000 -alias keystore
```

- ii. Fill in all the details asked while executing the above command
- iii. Recommended. After creating your keystore.jks file, please put it in the **android/app** folder
- iv. Create key.properties file in the **android** folder and add the details in the file as per the below screenshot.

```
storePassword=android  
keyPassword=android  
keyAlias=keystore  
storeFile=keystore.jks
```



NOTE:

- If you have changed any default value for any of these keys (storePassword, keyPassword, keyAlias, storeFile) while creating the keystore.jks file, then please also change them to key.properties file.
- If you place your keystore.jks file somewhere else in the project than mentioned in step 5.c.iii then please change storeFile key value accordingly.
- For more details please refer to [this link](#)

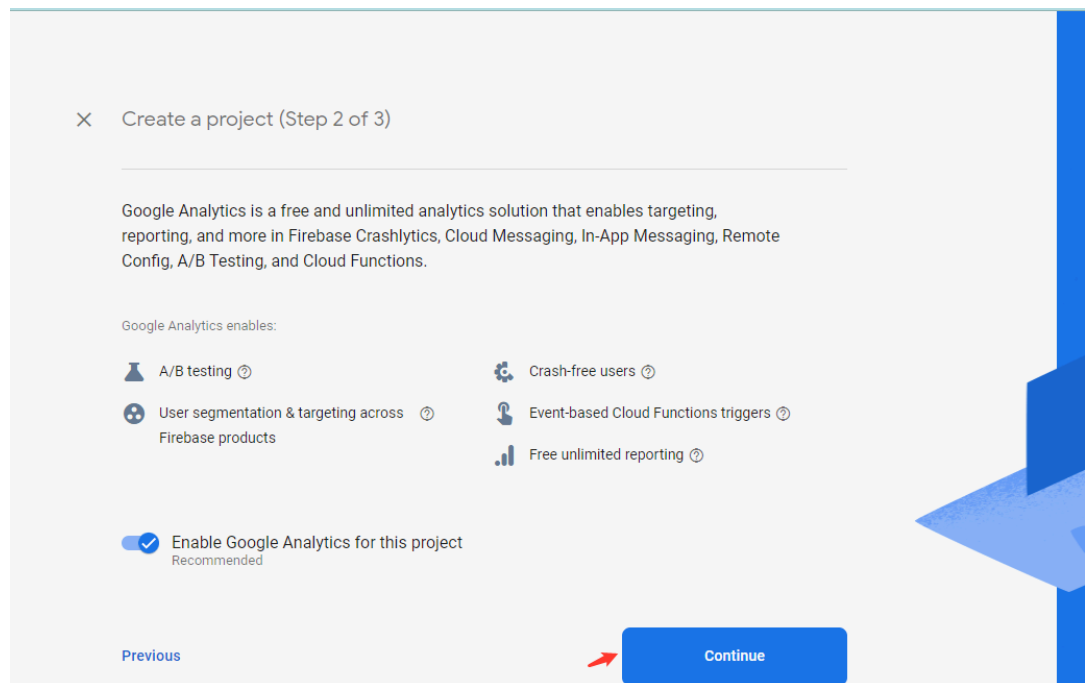
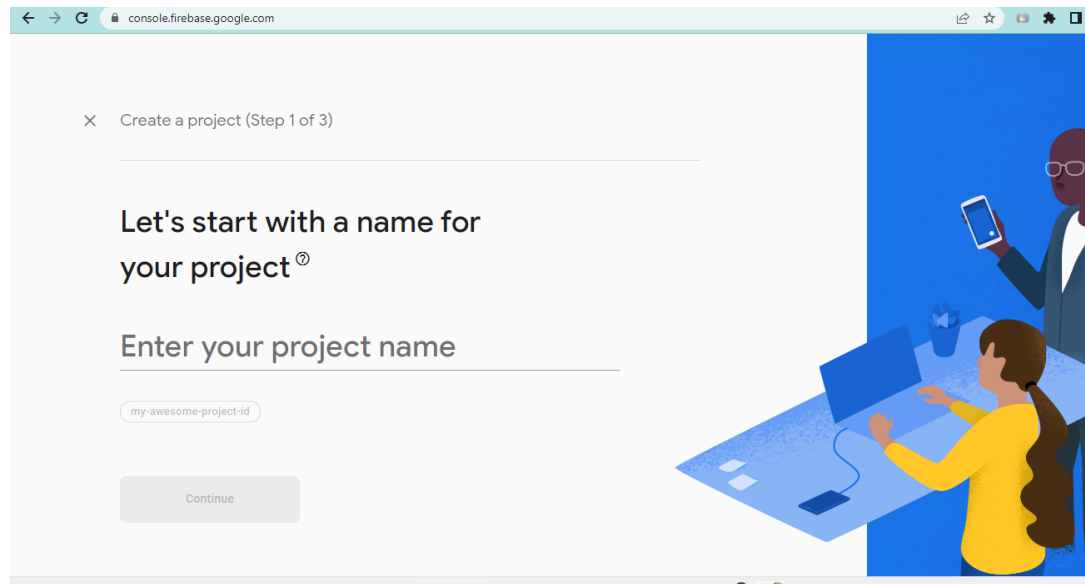
e. Create Firebase Account & Project

In this project, we are using the following Firebase services.

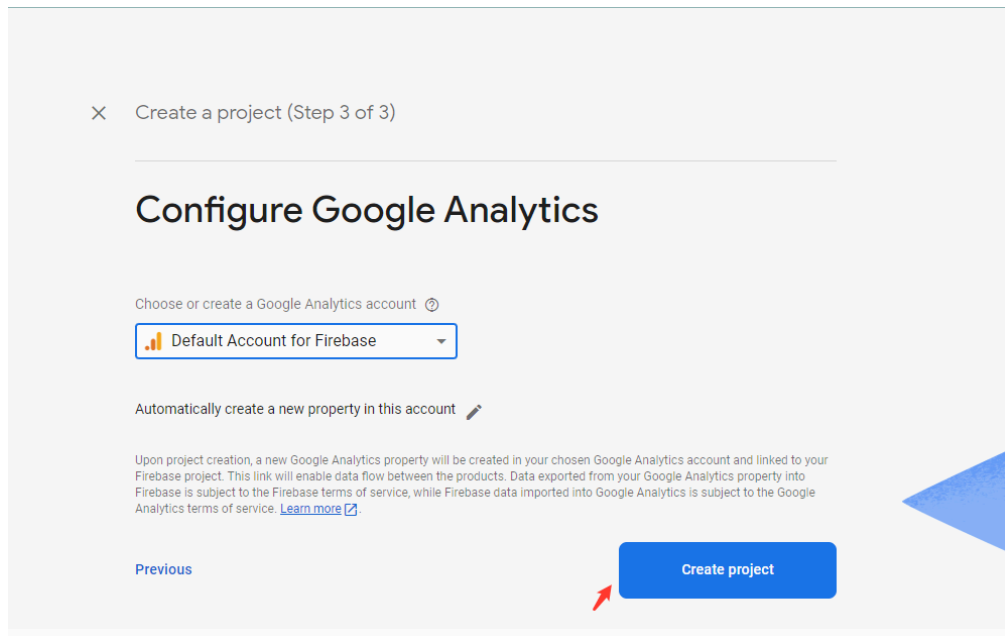
- i. Push Notification
- ii. Phone Authentication
- iii. Firestore Database
- iv. Firebase Analytics

For this, you need a Firebase account and a project set up in the Firebase. Please follow the below steps for this,

- i. Go to the [Firebase console](#)
- ii. Sign up if you don't have a Google Account or you want to create a new account for your project. Otherwise, sign in with your Google Account.
- iii. Click on **Add Project**
- iv. Enter your project name



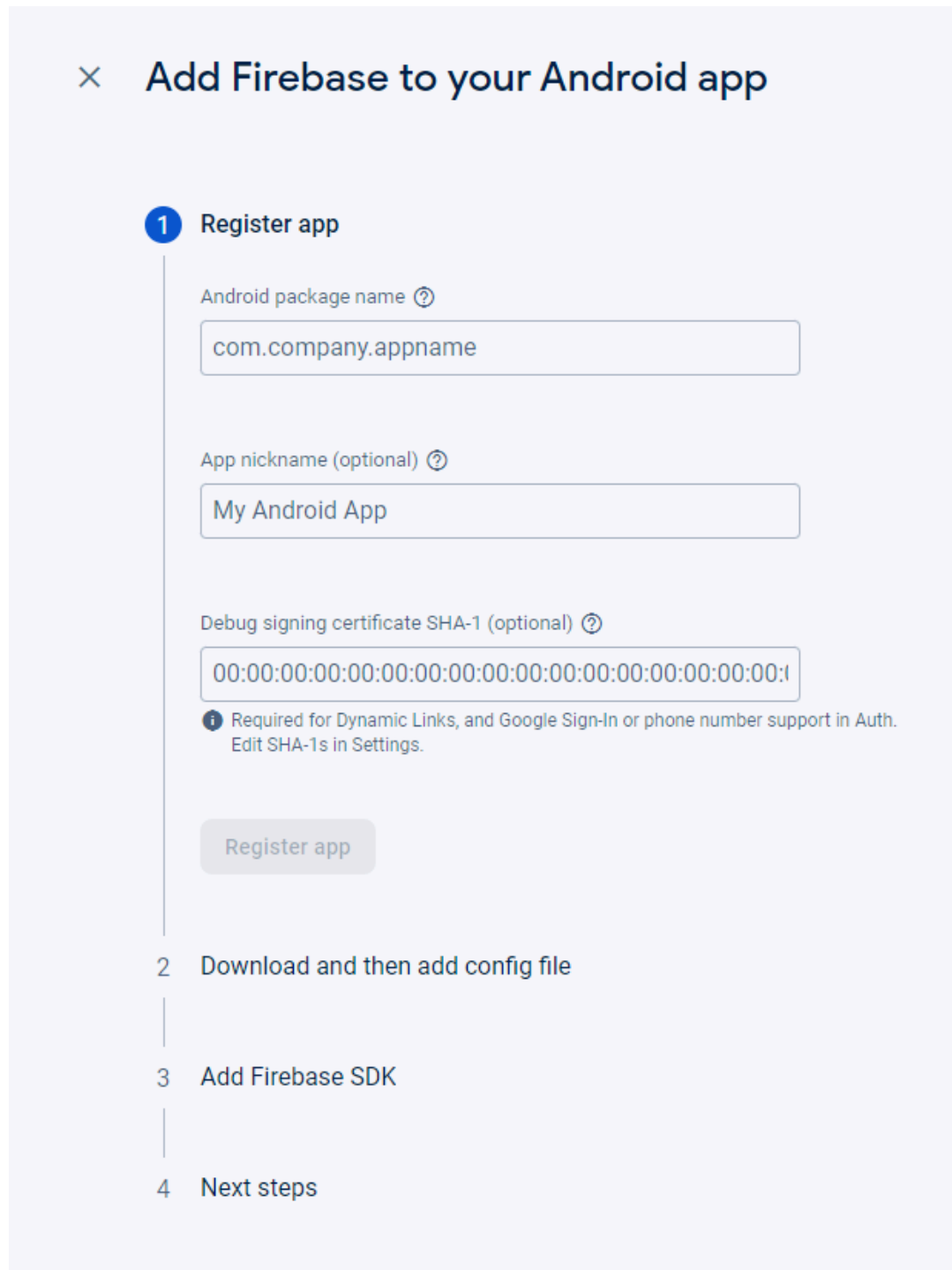
- v. Select Default Account for Firebase
(or you can create a new account)



vi. Create project

f. Set up Android App in Firebase Project

- i. Go to the Firebase console
 - ii. Select the project you created in step 5.d.vi
 - iii. Go to **Project Setting**
 - iv. In the **General** Tab click on the **Add App** button
 - v. Select **Android**
 - vi. Fill out the form and click on the **Register App Button**
- (Please check the below screenshot for reference)



- vii. You need SHA keys (SHA-1 and SHA-256) to add once you create the Android App in the above steps.
 - 1. To Generate debug SHA use the below command

```
keytool -list -v -keystore "Your directory path\debug.jks" -alias  
androiddebugkey -storepass android -keypass android
```

2. To Generate release SHA use the below command

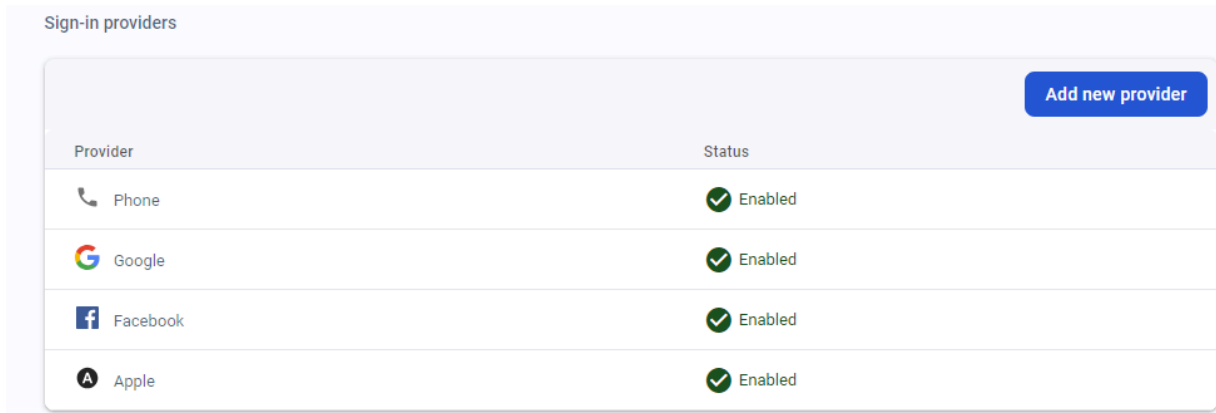
```
keytool -list -v -keystore "your directory path\keystore.jks" -alias  
androidreleasekey -storepass your store password -keypass your  
key password
```

After generating the debug and release SHA, you have to add them in the Firebase Console where you have created the Android app.

Please check the screenshot below for the reference.

[illegible]

- viii. Download the google-services.json file from Firebase project settings and paste it at **android/app** location.
- ix. Setup Authentication
 - Enable Sign in methods:
 - a. In the Firebase console's **Authentication** section, open the [Sign in method](#) page.
 - b. From the **Sign in method** page, enable the methods which are shown in the image.



g. Setup Firebase iOS App

- i. Go to the Firebase console
- ii. Select the project you created in step 5.d.vi
- iii. Go to **Project Setting**
- iv. In the **General** Tab click on the Add App button
- v. Select **iOS**
- vi. Fill out the form and click on the **Register App** Button

(Please check the below screenshot for reference)

×

Add Firebase to your Apple app

1

Register app

Apple bundle ID ?

com.company.appname

App nickname (optional) ?

My Apple app

App Store ID (optional) ?

123456789

Register app

2

Download config file

3

Add Firebase SDK

4

Add initialisation code

5

Next steps

- vii. Download the GoogleService-info.plist file from Firebase project settings and paste it at the **ios/Runner** location in the app

- viii. Go to the `ios\Runner\AppDelegate.m` file and replace “YOUR-API-KEY” with your Api key.

```
- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    [GMServices provideAPIKey:@"YOUR-API-KEY"];
    [GeneratedPluginRegistrant registerWithRegistry:self];
    // Override point for customization after application launch.
    return [super application:application didFinishLaunchingWithOptions:launchOptions];
}
```

- ix. Replace “YOUR-REVERSED-CLIENT-ID” with your reversed client ID. You can find this Id from the GoogleService-info.plist file you added from step 5.g.vii.

```
<string>Editor</string>
<key>CFBundleURLName</key>
<string>$(REVERSED_CLIENT_ID)</string>
<key>CFBundleURLSchemes</key>
<array>
    <string>YOUR-REVERSED-CLIENT-ID</string>
</array>
</dict>
<dict>
```

- x. XCode Project Setting

This step covers reviewing the most important settings in the XCode workspace. For detailed procedures and descriptions, see [Prepare for App Distribution](#)

1. Navigate to your target's settings in XCode:

- a. Open the default Xcode workspace in your project by running the below command in a terminal window from your Flutter project directory.

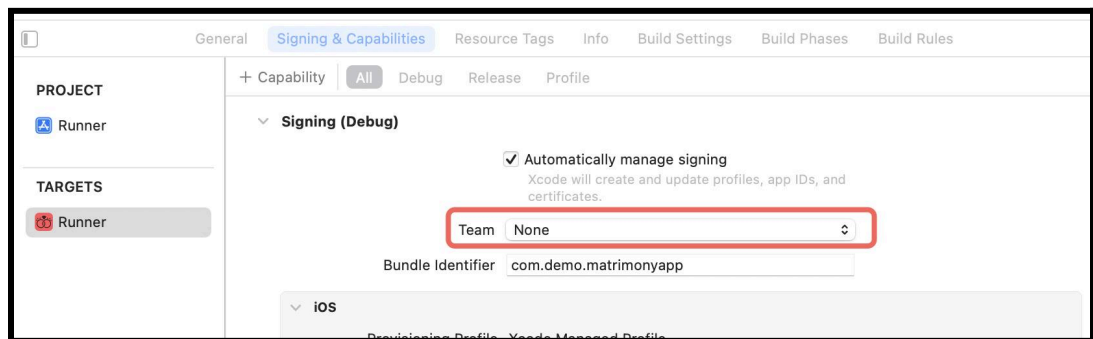
open ios/Runner.xcworkspace

- b. Select the Runner target in the Xcode navigator to view your app's settings.

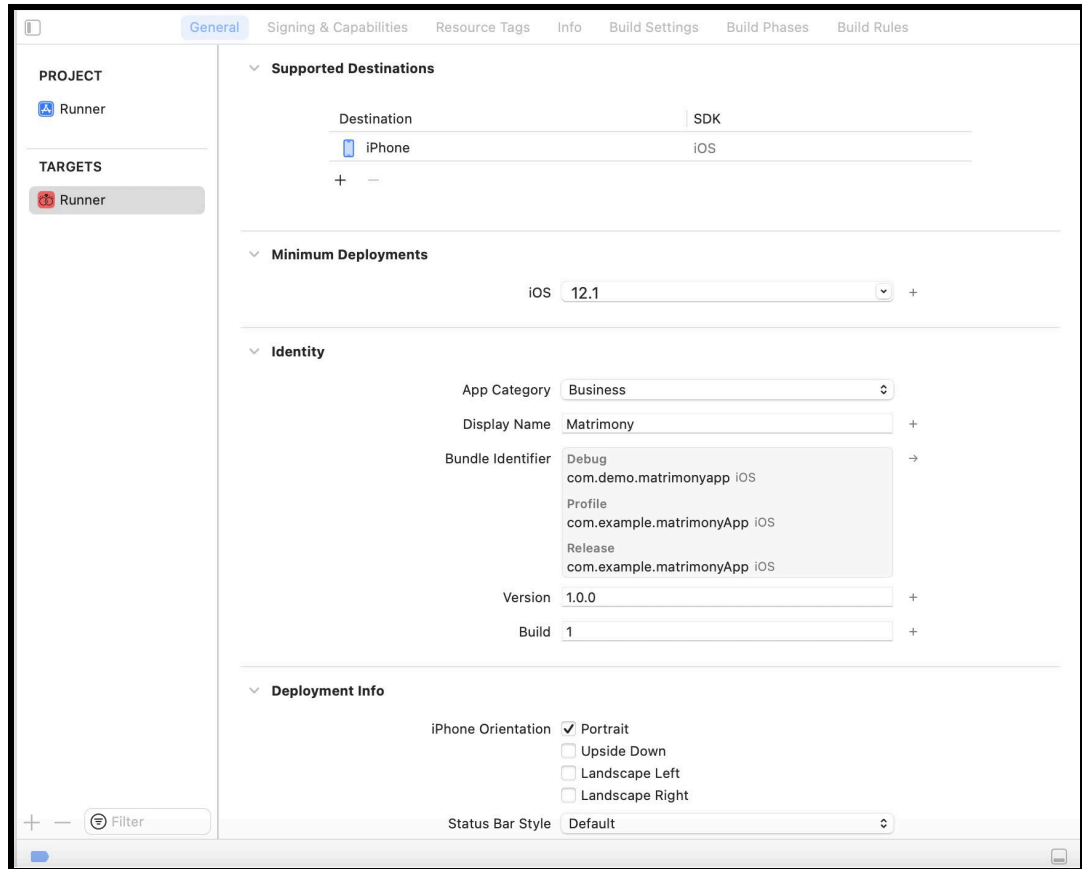
2. Verify the most important settings

- a. In the Identity section of the General tab

- i. **Display Name** (The display name of your app.)
 - ii. **Bundle Identifier** (The App ID you registered on App Store Connect.)
- b. In the Signing & Capabilities tab
- i. **Automatically manage signing** (Xcode should automatically manage app signing and provisioning. This is set true by default, which should be sufficient for most apps. For more complex scenarios, see the [Code Signing Guide](#))
 - ii. **Team** (Select the team associated with your registered Apple Developer account. If required, select Add Account..., then update this setting.)



- c. In the deployment section of the build settings tab:
- i. iOS Deployment Target
 - 1. The minimum iOS version that the app supports is 11.0.
 - 2. The General tab of your project settings should resemble the following:

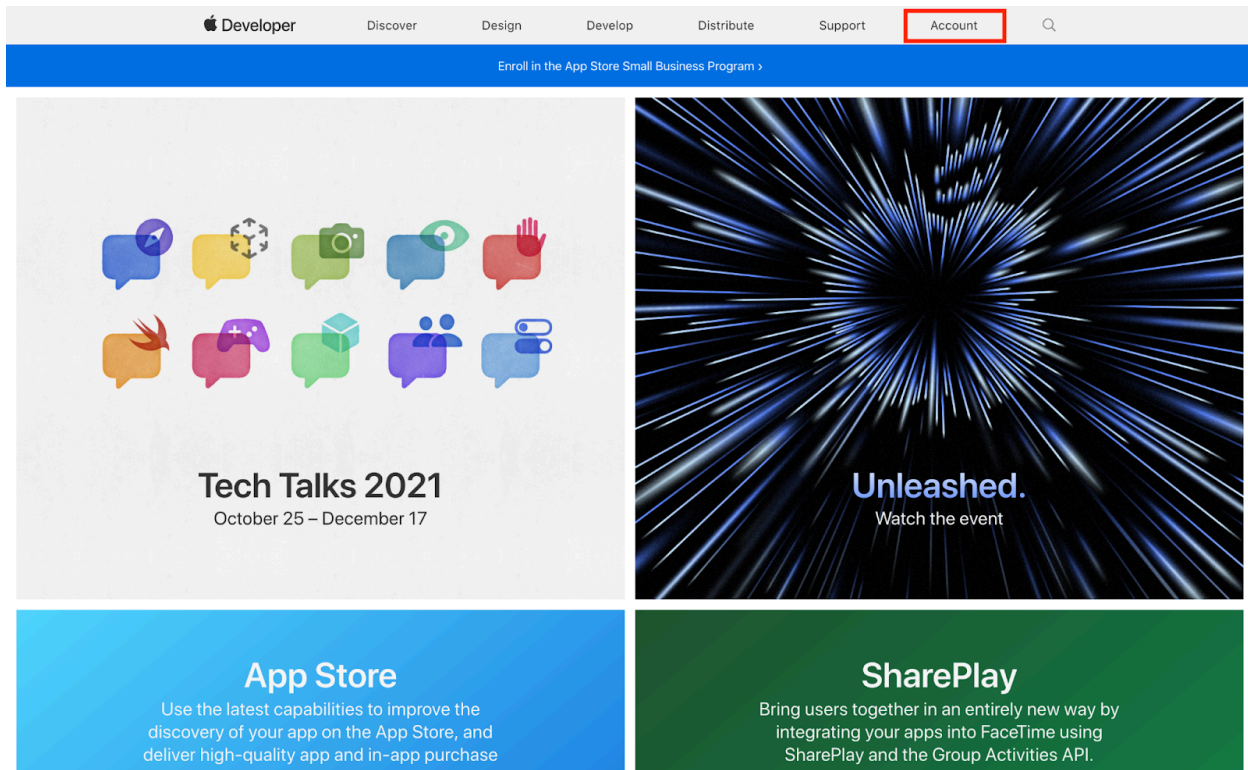


3. For a detailed overview of app signing, see [Create, export, and Delete signing certificates](#).

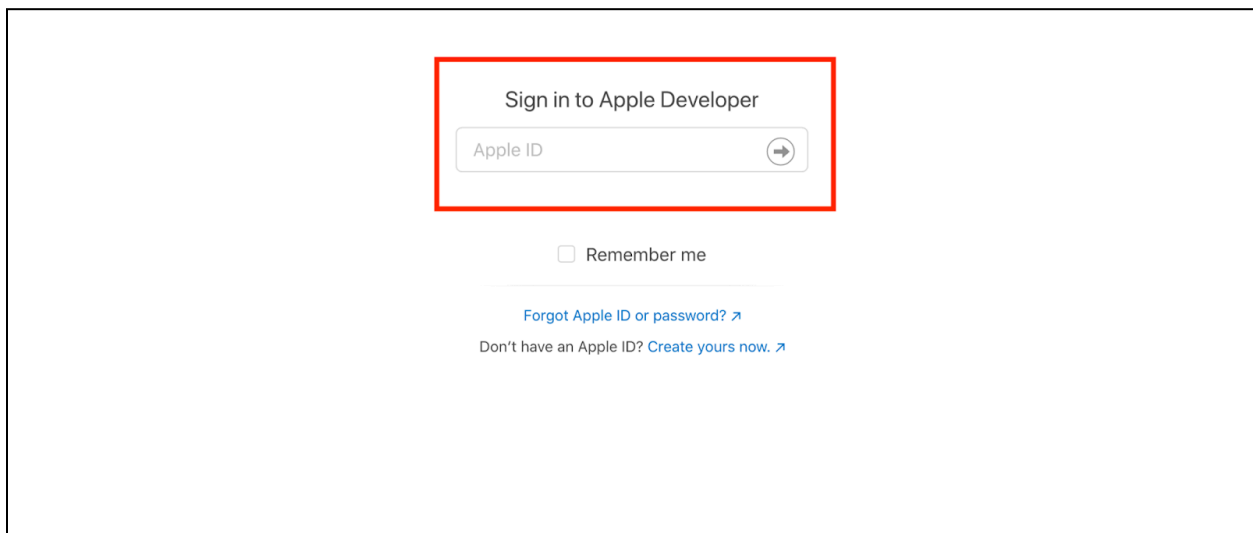
xi. Upload your APNs authentication key

If you don't already have an APNs authentication key, make sure to create one.

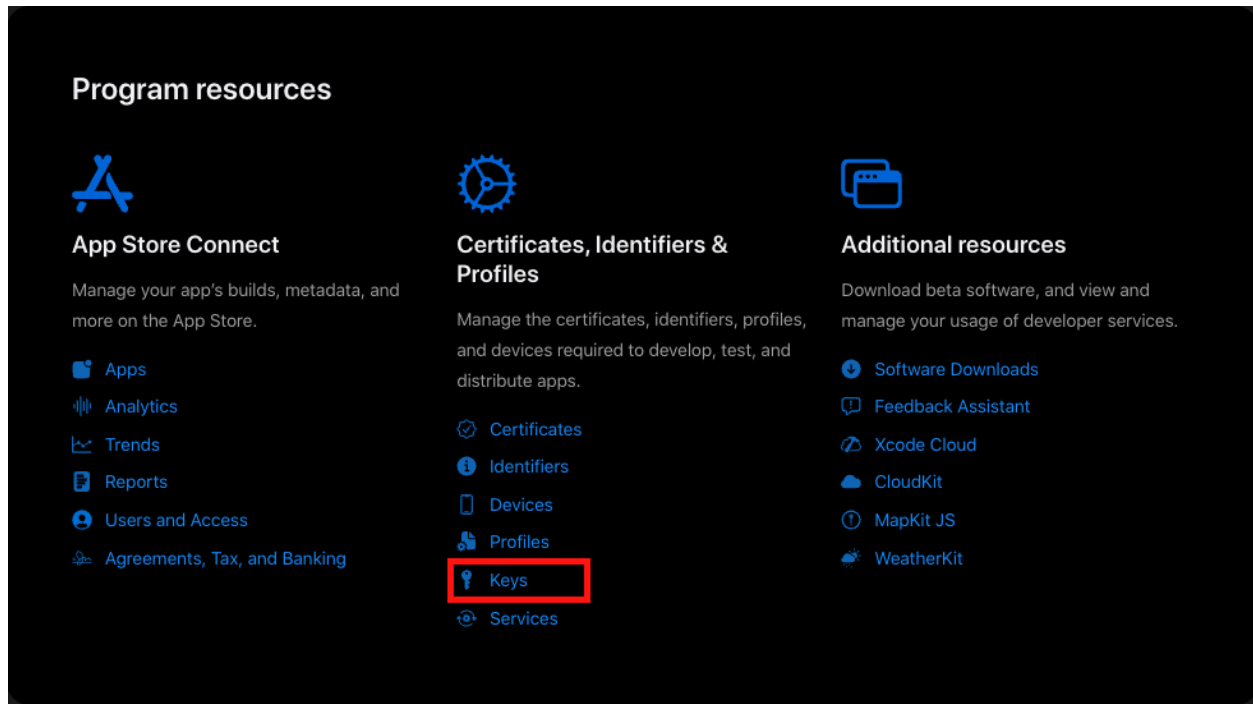
1. Go to <https://developer.apple.com> and click Account



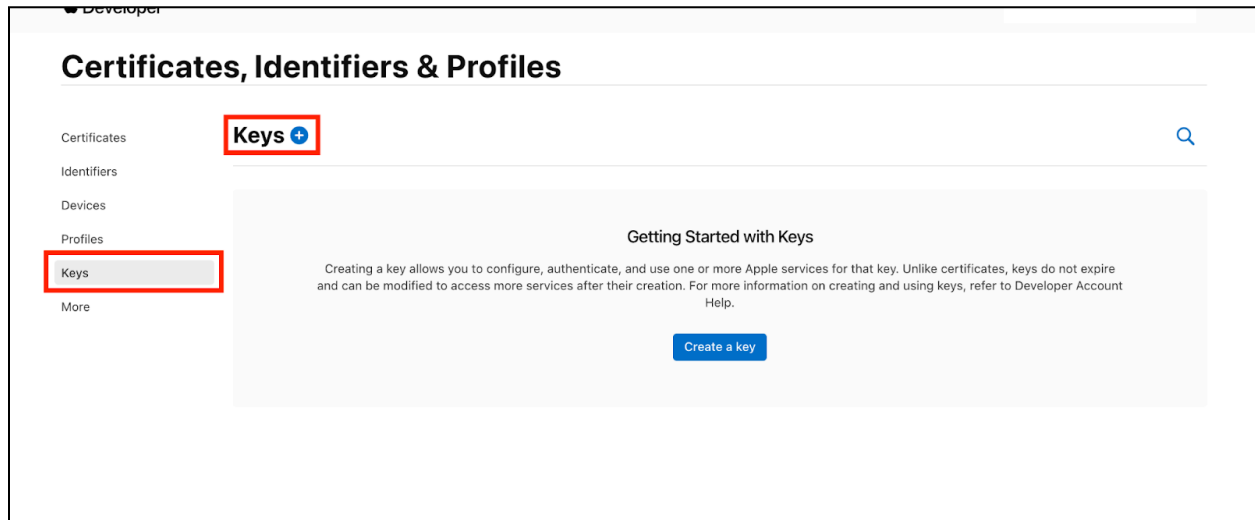
2. Log in with your Apple Developer account



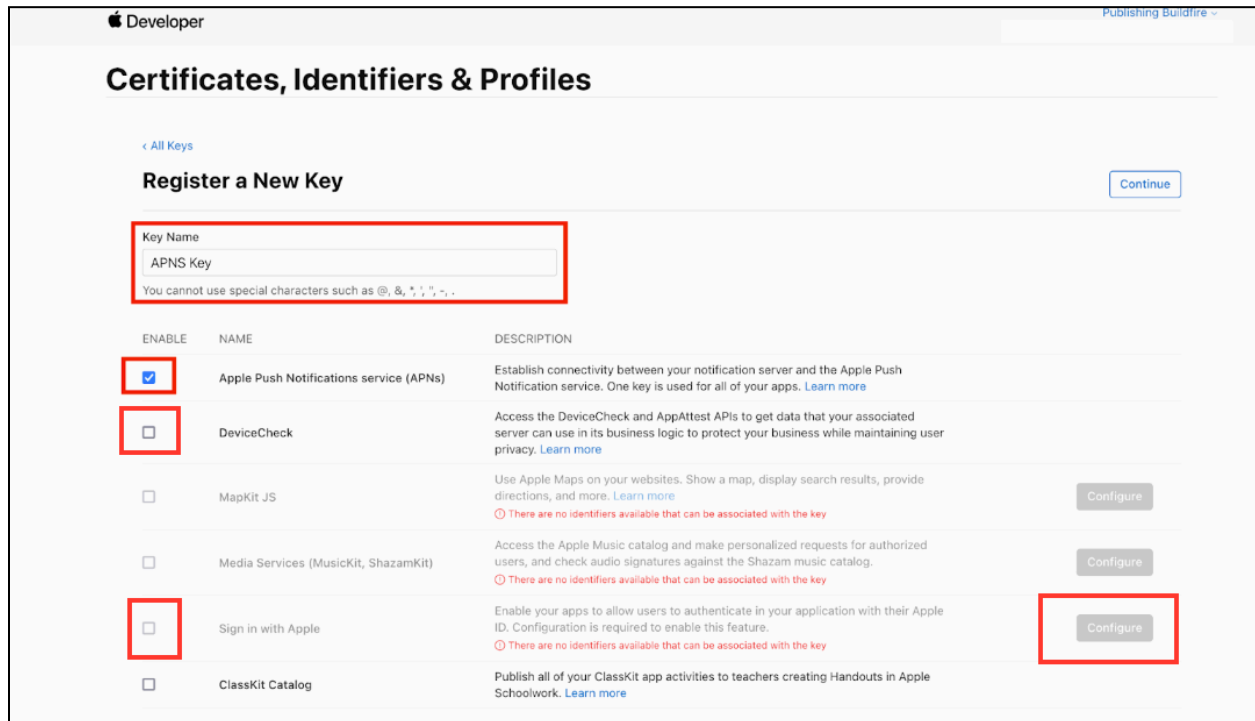
3. Click on Certificates, IDs & Profiles



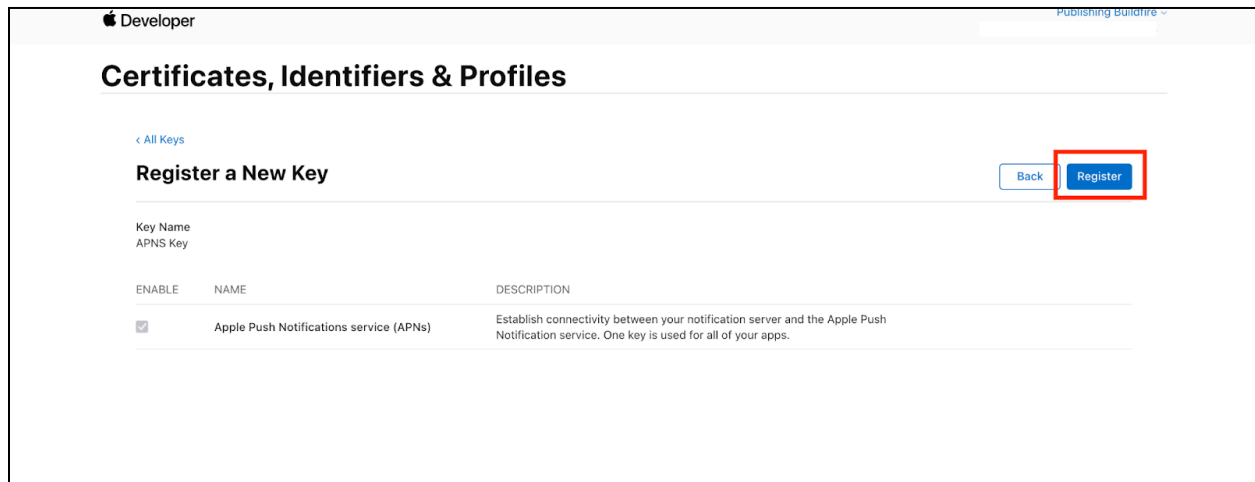
- Click on Keys and add a new key by clicking on the blue plus (+) icon next to the title Keys.



- On the next page, enter 'APNS Key' in the Key Name field and click the checkbox to enable Apple Push Notifications service (APNs), Device Check and Sign in with Apple. Also configure the Sign in with Apple.

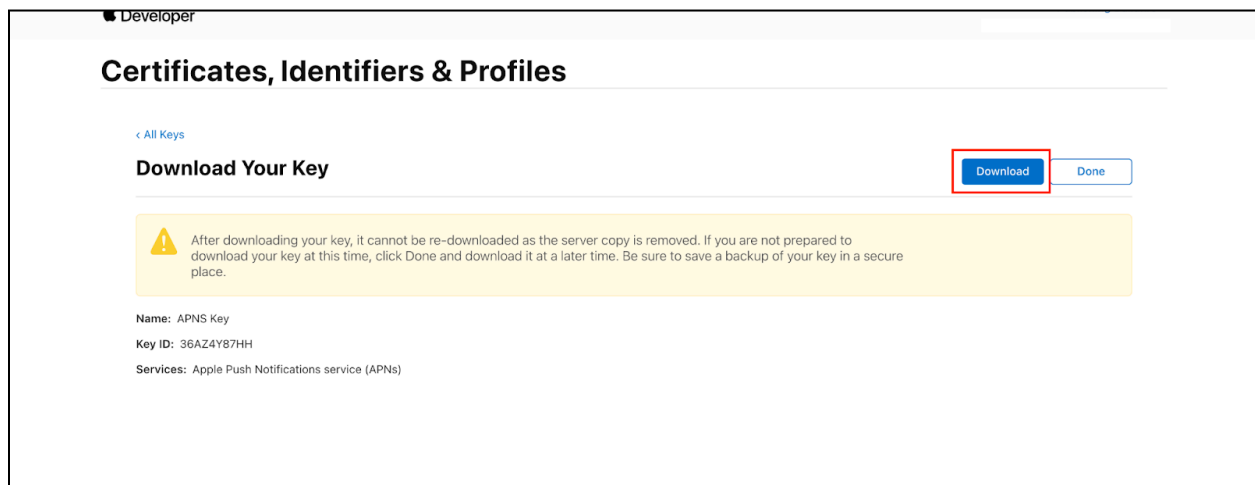


6. Click Register



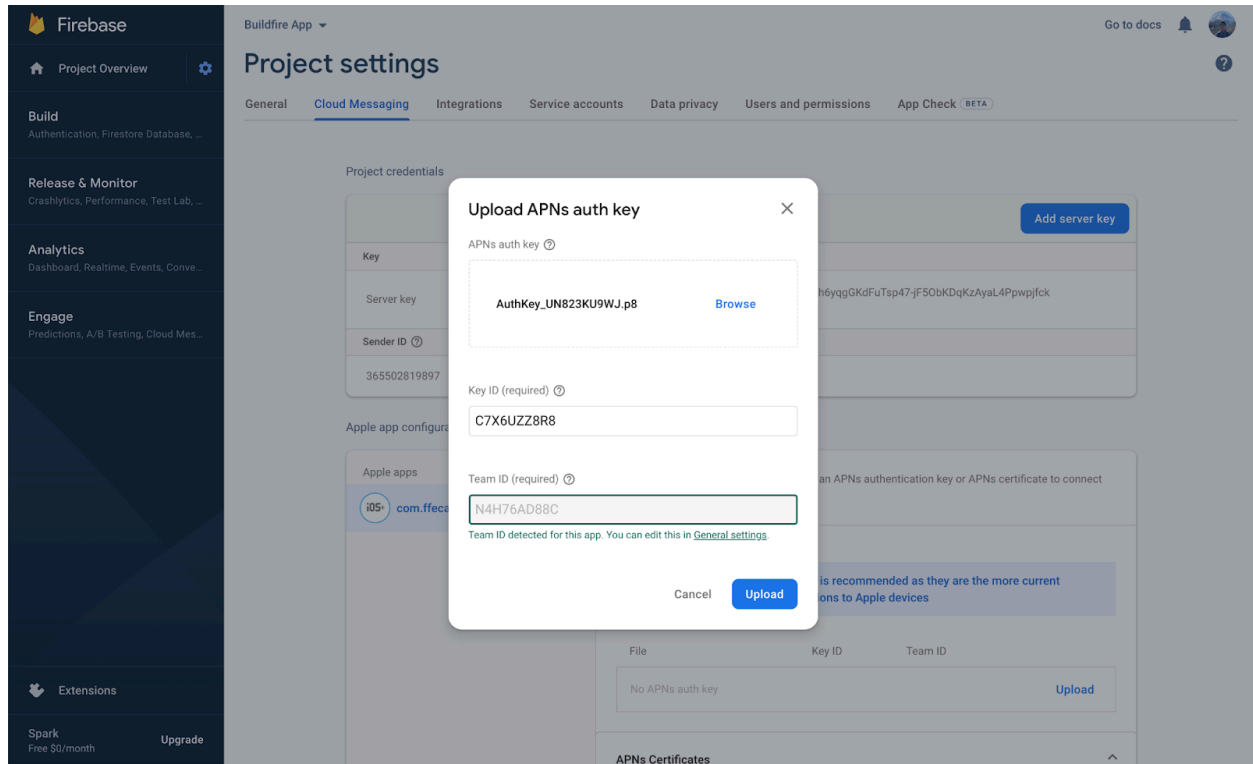
7. Click Download. This will download the APNs key that you will upload to Firebase. Please keep this page open to obtain the Key ID and Team ID for Firebase.

NOTE: Once the key has been downloaded, it cannot be retrieved again.

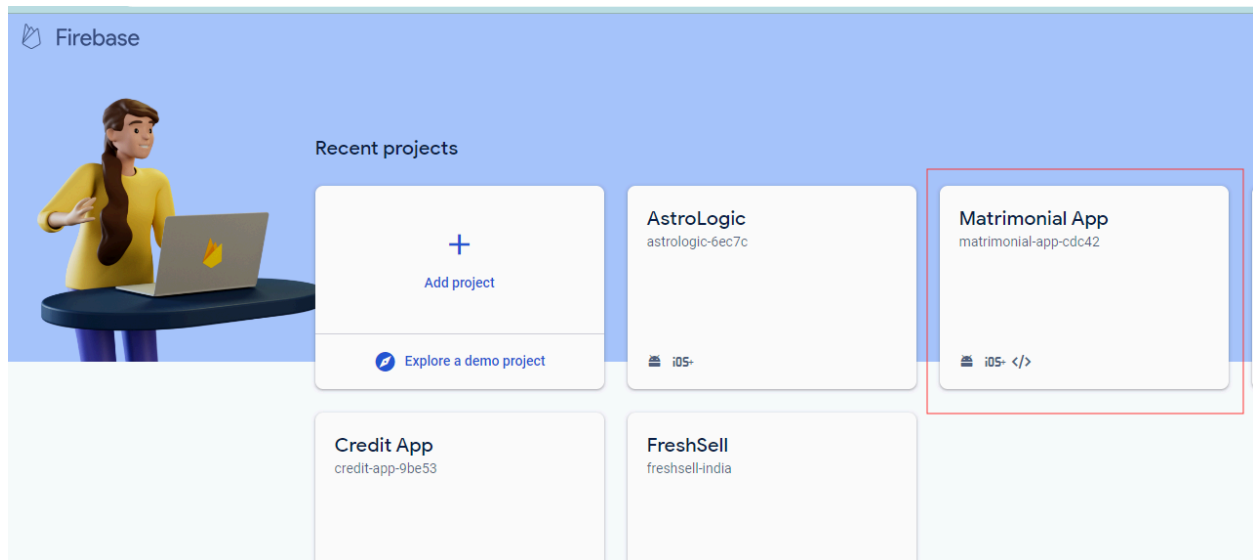


8. Now that you have the APNS key downloaded, you will need to upload this to Firebase. Open up a new browser tab or window and navigate to <https://console.firebase.google.com/>

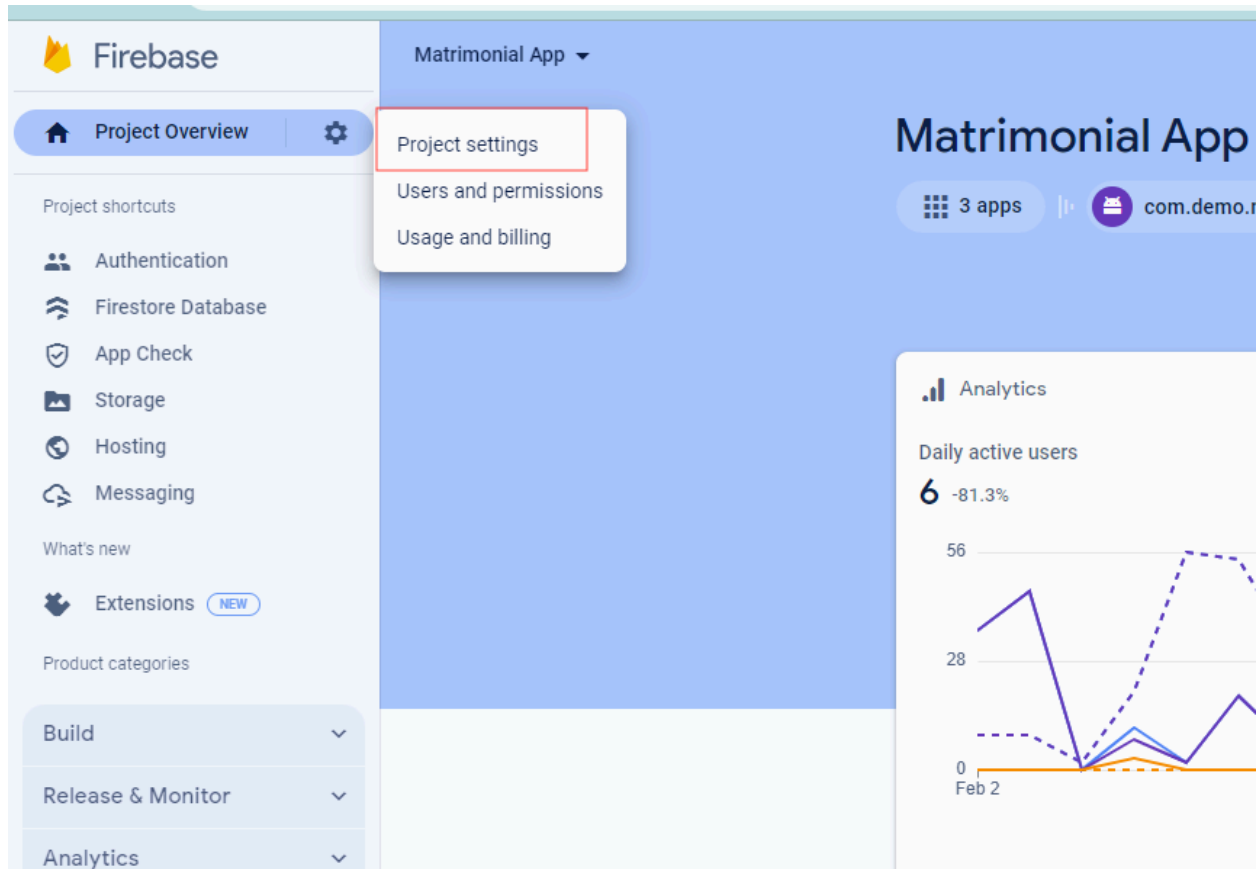
NOTE: The next few steps require you to have an iOS Firebase Certificate. If you have not done this yet, please check out our [iOS Firebase Certificate](#) article before continuing.



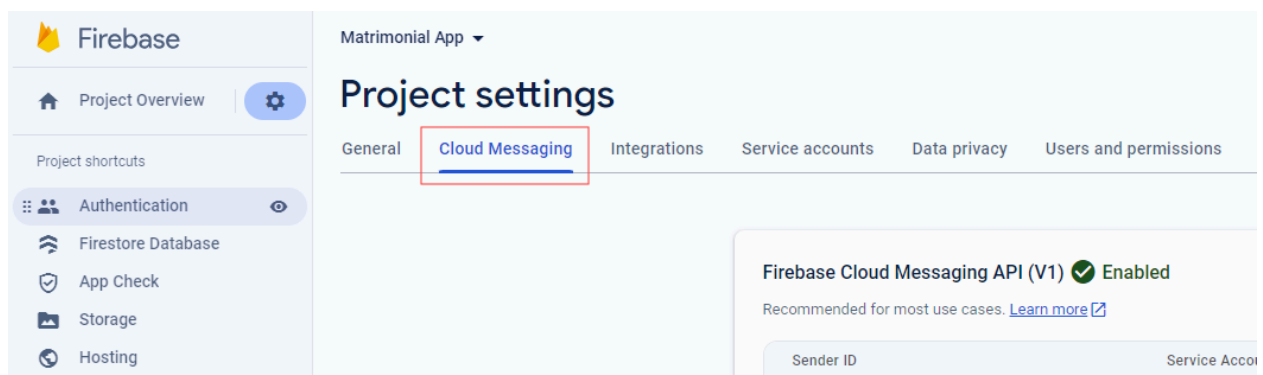
9. Click on your App project

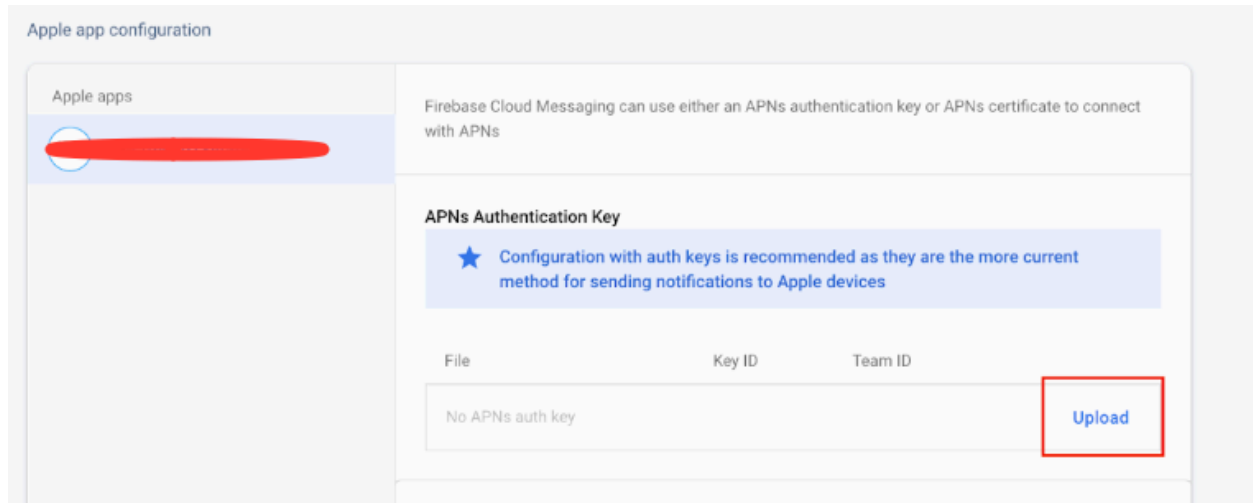


10. On the top left, click on the gear icon on the right side of Project Overview and select Project Settings

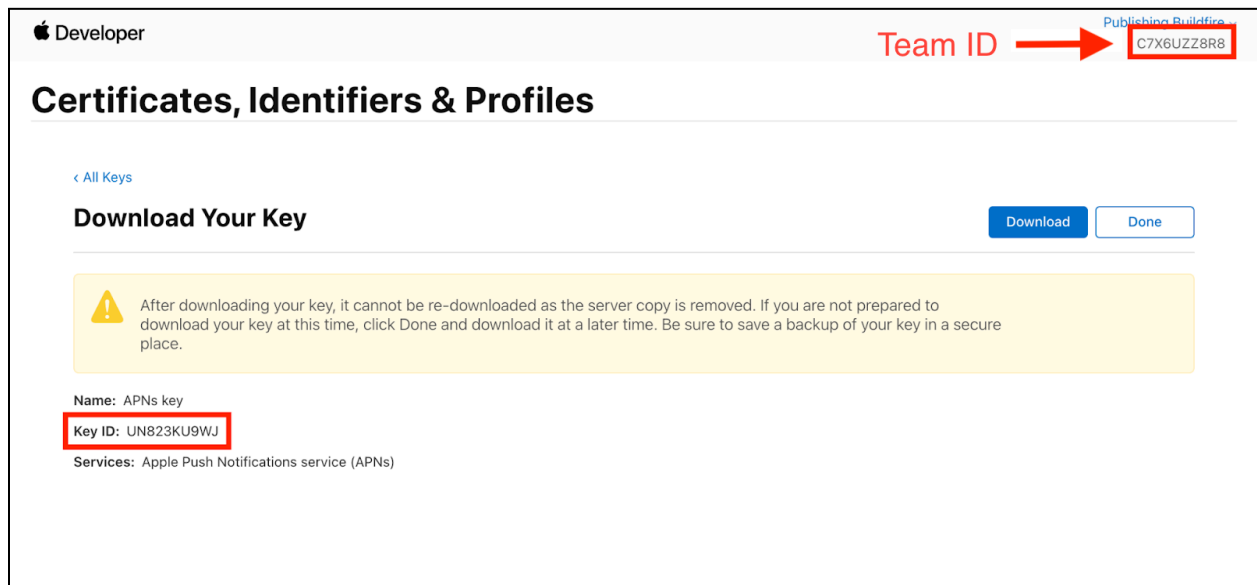


11. Click on the Cloud Messaging tab and in the Apple app configuration section, click Upload

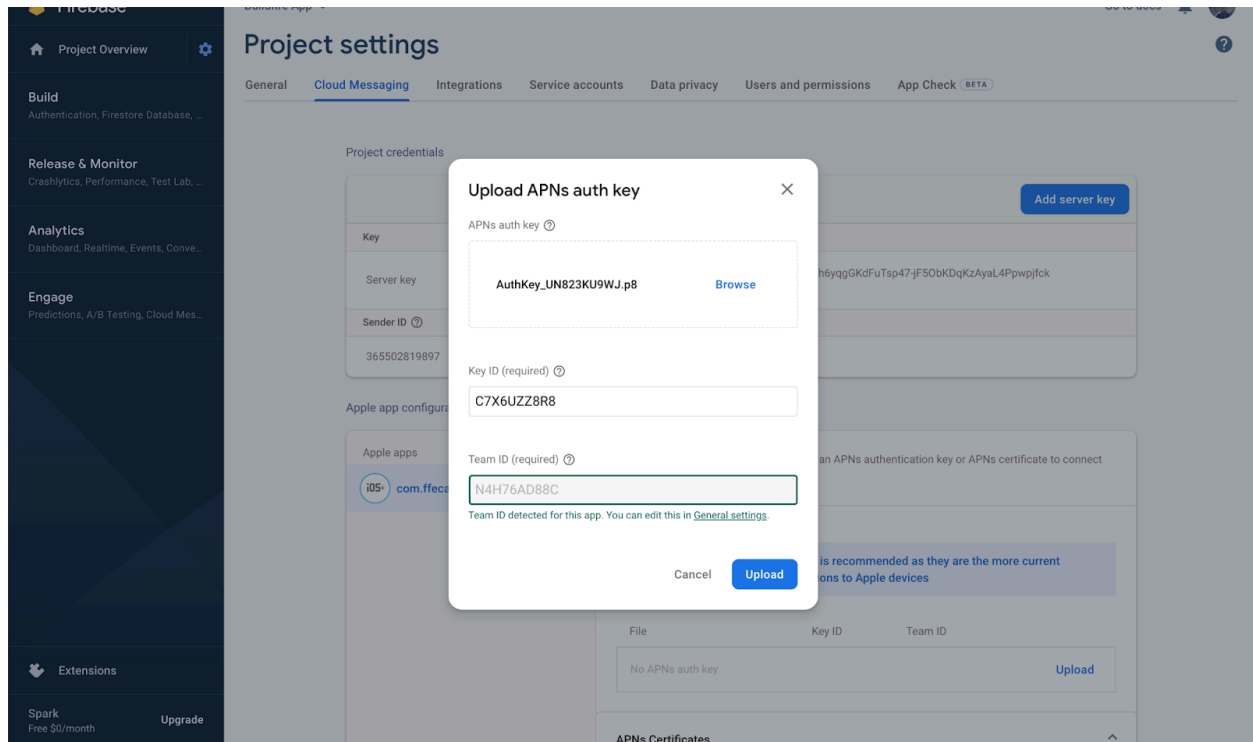




12. Here you will upload the key file by clicking Browse. Select the file that ends with .p8 that was downloaded in the previous steps. The file name will look like this: AuthKey_UN823KU9WJ.p8
13. Now you will have to copy the Key ID and Team ID by going back to your Apple Developer account. The Key ID is located below the key name and the Team ID is located in the top right corner, next to your Apple developer name.



14. Go back to the Firebase page and copy and paste the Key ID and Team ID. Lastly, click on the Upload button.



h. Configure the Firebase setting to the Project

Go to the **lib\firebase_option.dart** file

- For Android settings replace your credentials in the android method

```
static const FirebaseOptions android = FirebaseOptions(  
  apiKey: "Your key",  
  authDomain: "*Your Firebase Porject Id*.firebaseapp.com",  
  projectId: "Your Firebase Porject Id",  
  storageBucket: "*Your Firebase Porject Id*.appspot.com",  
  messagingSenderId: "Your message Sender Id",  
  appId: "Your Firebase App Id",  
);
```

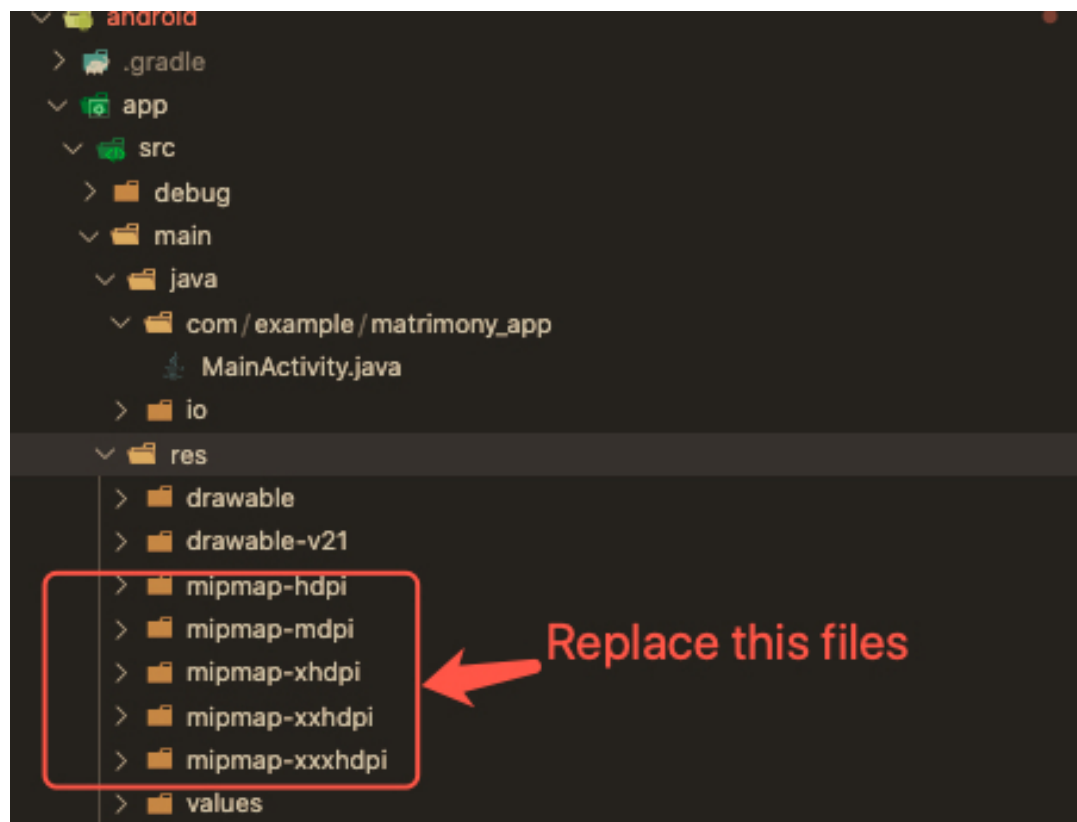
- For iOS settings replace your credentials in the ios method

```
static const FirebaseOptions ios = FirebaseOptions(  
    apiKey: "Your key",  
    authDomain: "*Your Firebase Porject Id*.firebaseapp.com",  
    projectId: "Your Firebase Porject Id",  
    storageBucket: "*Your Firebase Porject Id*.appspot.com",  
    messagingSenderId: "Your message Sender Id",  
    appId: "Your Firebase App Id",  
    androidClientId: 'Your Android Client Id',  
    iosClientId: 'Your Ios Client Id',  
    iosBundleId: 'com.demo.matrimonyapp',  
);
```

i. Change App Icon

i. For Android

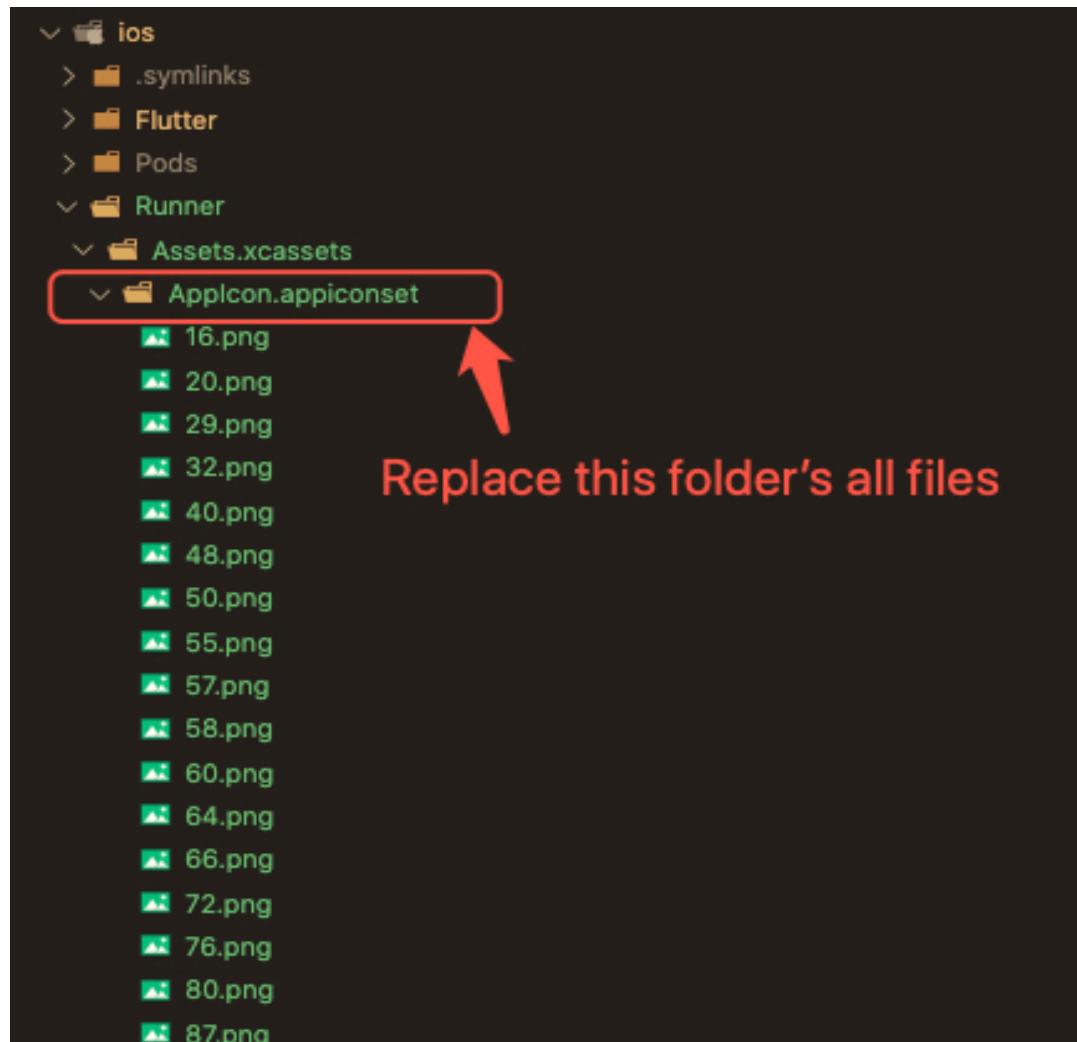
Replace the icons in the **android\app\src\main\res** folder as shown in the below image.



ii. For iOS

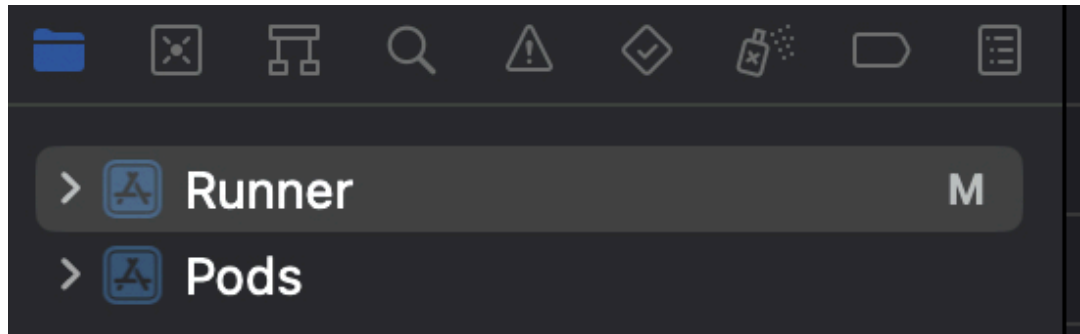
1. Replace the icons in the below folder as shown in the below image

ios\Runner\Assets.xcassets\AppIcon.appiconset

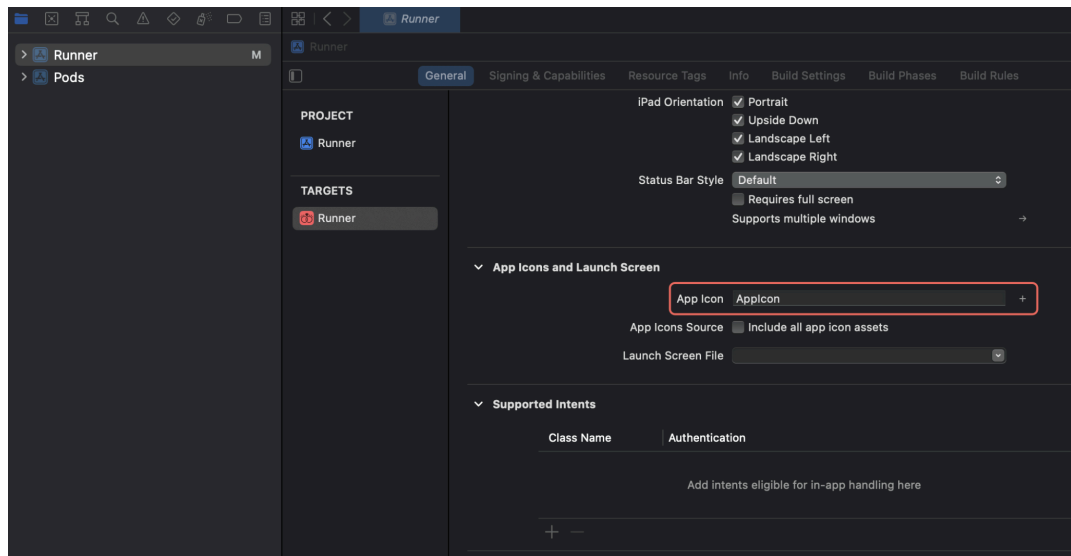


2. Change icons using XCode

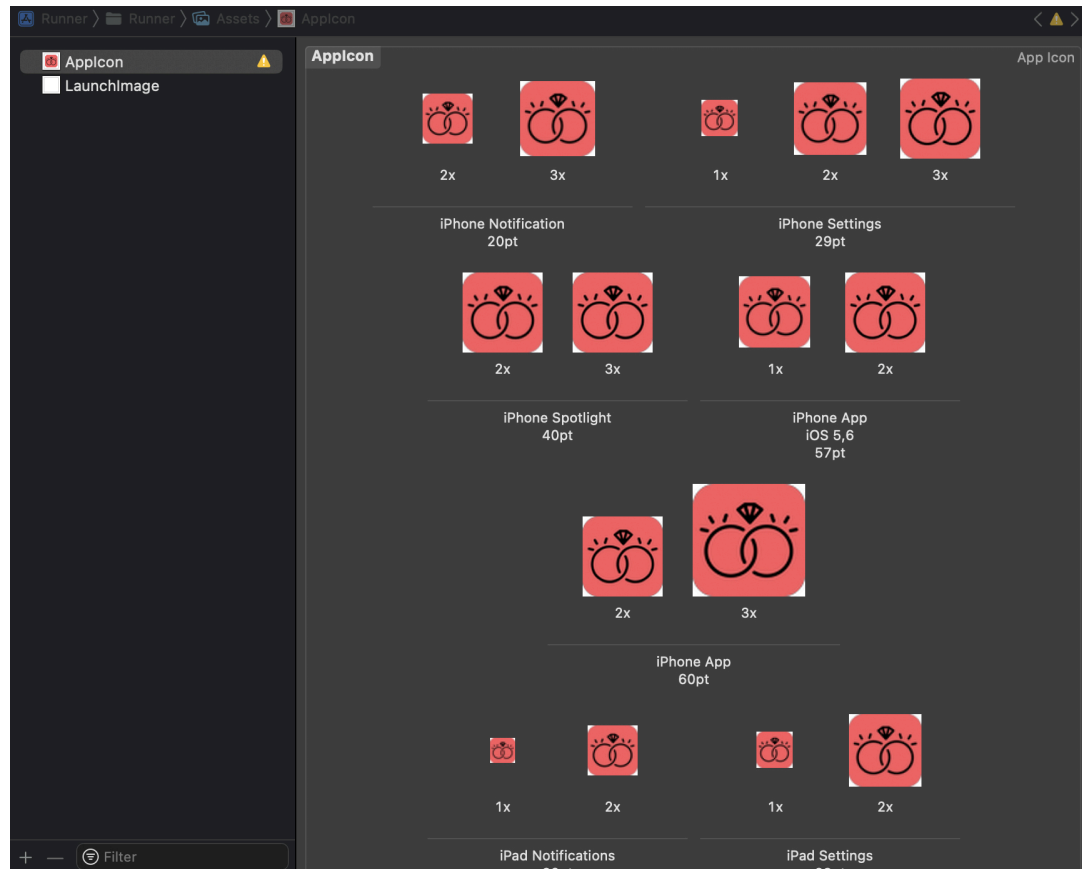
- a. Right-click on the iOS folder Choose Open in Xcode Option
- b. Click on the folder icon left side of the XCode window



- c. Select Runner.
- d. Select Target runner
- e. Go to App Icons And Launch Images
- f. Click the right arrow button of the app icons source



g. Replace all the icons according to their size



NOTE:

- If you want to generate the App icon bundle from any image you have, you can generate it from publicly available websites like

<https://www.appicon.co/>

j. Setup Deep Link

i. For Android

1. Open **android/app/src/main/AndroidManifest.xml** file.
2. Add the following metadata tag and intent filter inside the **<activity>** tag **.MainActivity**.
Replace **example.com** with your web domain.

```
>  
<meta-data android:name="flutter_deeplinking_enabled" android:value="true" />  
<intent-filter android:autoVerify="true">  
  <action android:name="android.intent.action.VIEW" />  
  <category android:name="android.intent.category.DEFAULT" />  
  <category android:name="android.intent.category.BROWSABLE" />  
  <data android:host="YOUR-DOMAIN" android:scheme="https" android:pathPrefix="/matrimony"/>  
</intent-filter>
```

3. Change android:launchMode to android:launchMode="singleTask"
4. Hosting assetlinks.json file

File content:

```
{  
  "relation": ["delegate_permission/common.handle_all_urls"],  
  "target": {  
    "namespace": "android_app",  
    "package_name": "com.example.deeplink_cookbook",  
    "sha256_cert_fingerprints": ["FF:2A:CF:7B:DD:CC:F1:03:3E:E8:B2:2  
7:7C:A2:E3:3C:DE:13:DB:AC:8E:EB:3A:B9:72:A1:0E:26:8A:F5:EC:A  
F"]  
  }  
}
```

Where,

- Set the **package_name** value to your Android application ID.
- Replace sha256_cert_fingerprints with yours
- Host the file at a URL that resembles the following:
<webdomain>/well-known/assetlinks.json
- Verify that your browser can access this file.

5. Share your link as follows:
(Link should be look a like as follow)
<https://<YOUR MAIN DOMAIN>/PassYourNeededParameter>

ii. For ios

1. Hosting apple-app-site-association file

Adjust IOS settings.

a. Hosting apple-app-site-association file :

- You need to host an **apple-app-site-association** file without any extension(No .json,.php,.aspx etc...) in the web domain(on Main

Domain Not on Sub Domain). This file tells the mobile browser which iOS application to open instead of the browser.

- The hosted file should have the following content:

```
{
  "applinks": {
    "apps": [],
    "details": [
      {
        "appID": "S8QB4VV633.com.example.deeplinkCookbook",
        "paths": ["*"]
      }
    ]
  }
}
```

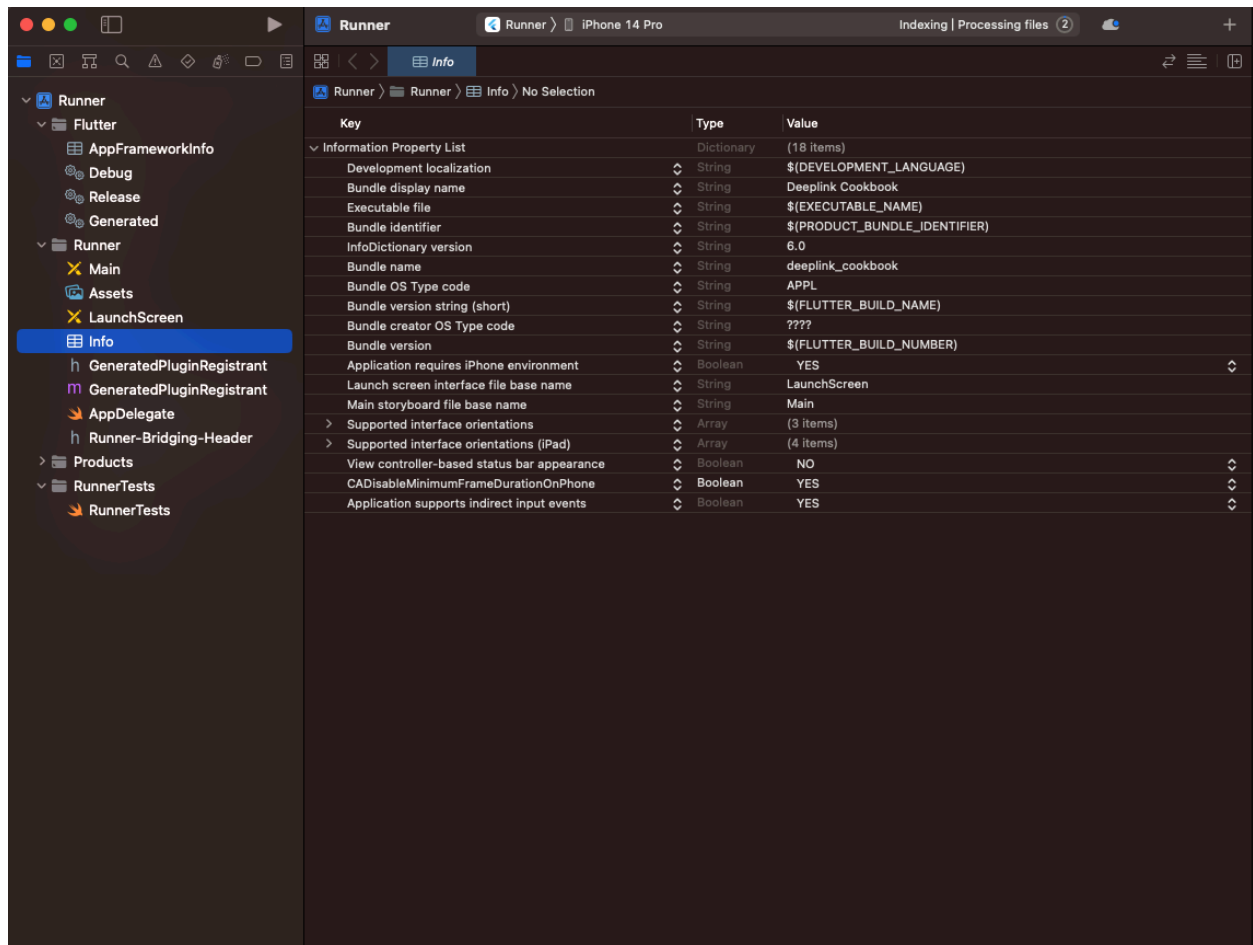
Replace “appID” with your ids.

“appID” format: <team id>.<bundle id>.

- Host the file at a URL that at the following: <YOUR MAIN WEB DOMAIN>/well-known/apple-app-site-association
- Verify that your browser can access OR download this file.
- **Note:** It might take up to 24 hours before Apple’s Content Delivery Network (CDN) requests the apple-app-site-association (AASA) file from your web domain. The universal link won’t work until the CDN requests the file. To bypass Apple’s CDN, check out the alternate mode section.

2. Adjust IOS settings:

- a. Launch Xocode.
- b. Open the **ios/Runner.xcworkspace** file inside the project’s **ios** folder.
- c. Navigate to the **Info.plist** file in the **ios/Runner** folder.

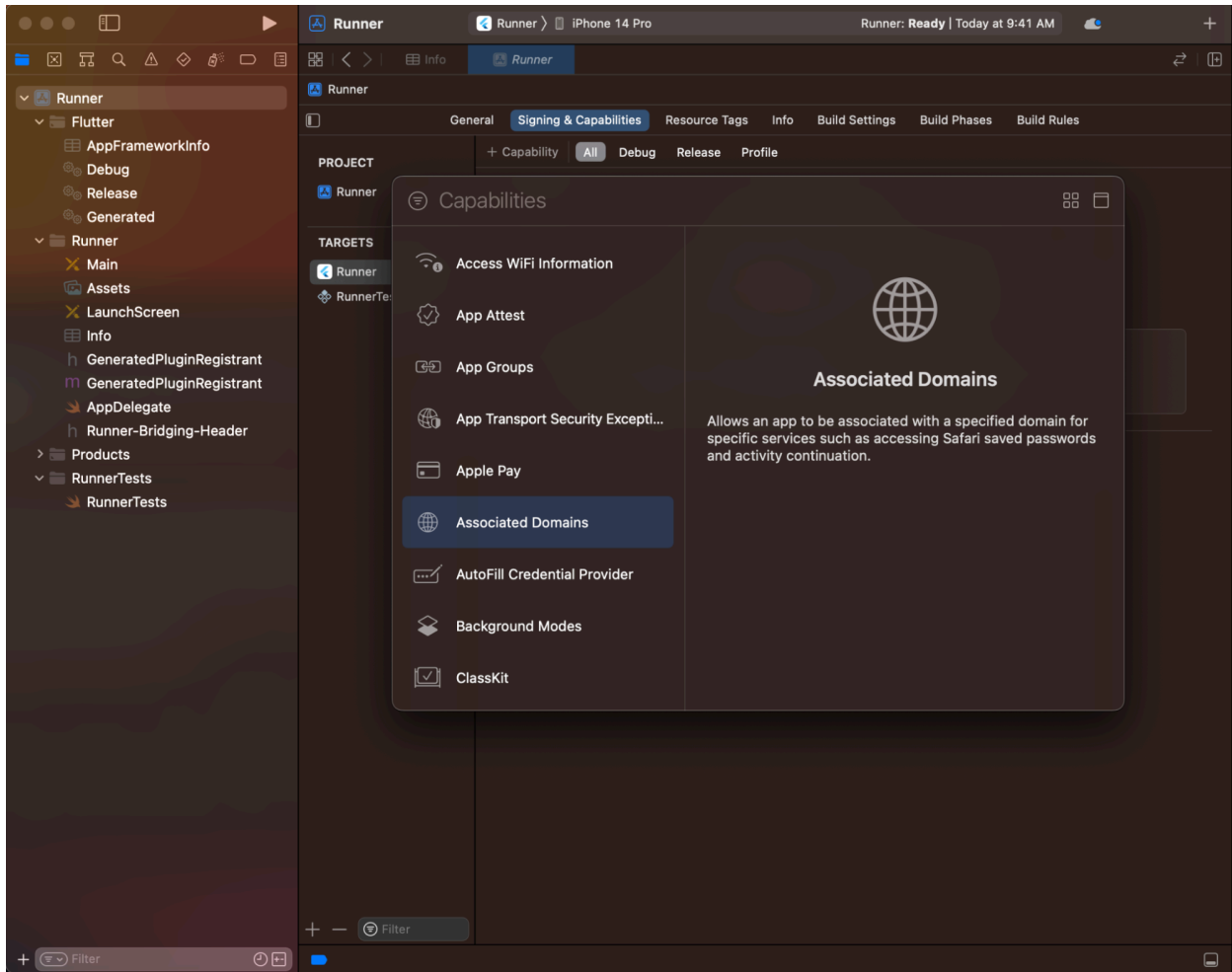


- d. In the **Info** property list, control-click on the list to add a row.
- e. Control-click the newly added row and turn on the Raw Keys and Values mode
- f. Update the key to **FlutterDeepLinkingEnabled** with a **Boolean** value set to

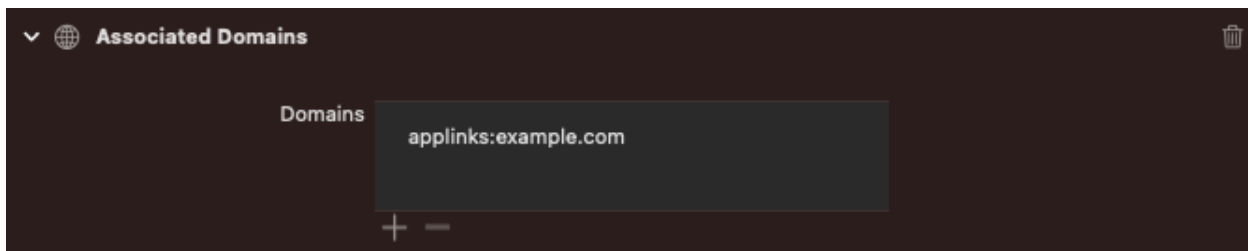


NOTE: The FlutterDeepLinkingEnabled property opts into Flutter's default deeplink handler. If you use third-party plugins, such as uni_links, setting this property will break these plugins. Skip this step if you prefer to use third-party plugins.

- g. Click the top-level Runner.
- h. Click Signing & Capabilities.
- i. Click + Capability to add a new domain.
- j. Click Associated Domains.



- k. In the Associated Domains section, click +.
- l. Enter **applinks:<web domain>**. Replace **<web domain>** with your own domain name.



k. Build Release for Android

- i. Open Project in VS Code
- ii. In Terminal Execute the below commands

```
flutter clean  
flutter pub get  
flutter build apk --release
```

- iii. After making the release, to generate the release bundle Execute the below command

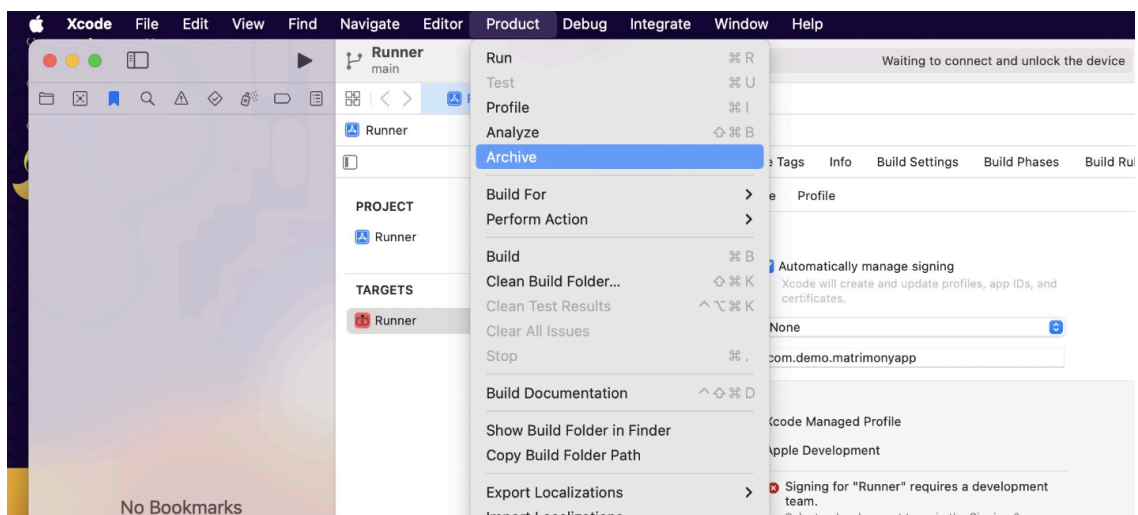
```
flutter build appbundle --release
```

- iv. Get the APK from the below path

build\app\outputs\flutter-apk\app-release.apk

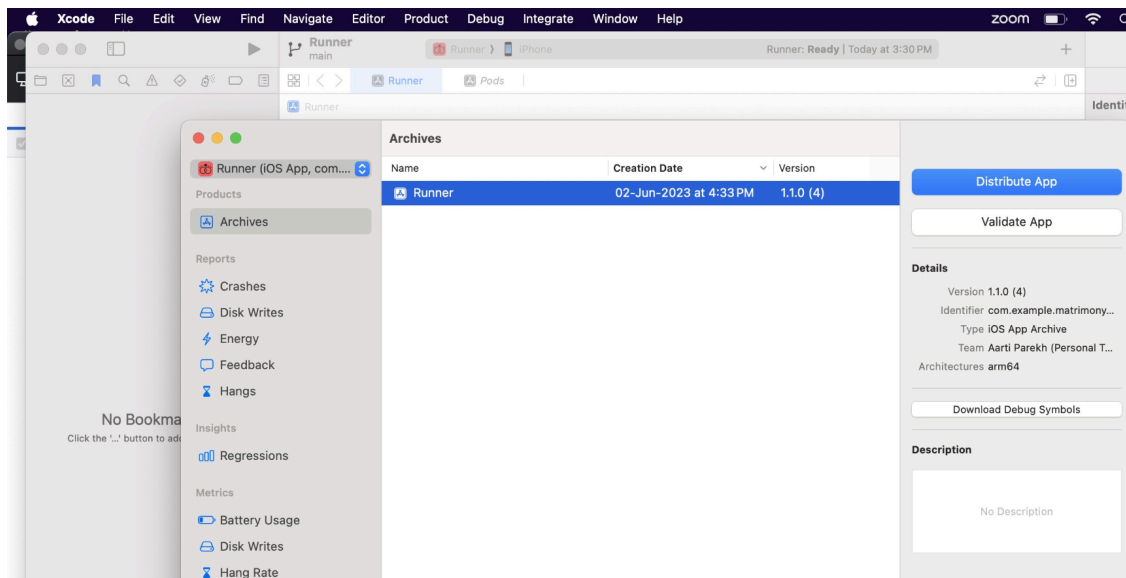
I. Build Release for iOS

- i. Open Project in XCode
- ii. Select **Archive** from the **Product Menu**



- iii. After successfully archiving select the **Organizer** option from the **Windows menu**

- iv. After clicking on it opens one popup for Archive, Click on the **Distribute App Button**



- v. After successfully done, you can upload this app to your Apple developer account in the TestFlight

- vi. To publish your app from TestFlight please follow [this link](#)

NOTE: While running/building the app, if you get any error **A Firebase App name “[DEFAULT]” already exists**, then add the param name when initializing the Firebase in the `lib/main.dart` file.

```
Future<void> main() async {  
  WidgetsFlutterBinding.ensureInitialized();  
  await Firebase.initializeApp(  
    name: 'Matrimony',  
    options: DefaultFirebaseOptions.currentPlatform,  
  );  
  FirebaseMessaging.onBackgroundMessage(_firebaseMessagingBackgroundHandler);  
}
```

m. Other Options for the Advanced User

i. Paths to the images used in the app

Images	Path	Screen Path
Splash screen	assets/images/splashImage.png	lib/views/splash/splash_screen.dart
Introduction screen	assets/images/introImg1.png	lib/views/introduction/intro_screen.dart
	assets/images/introImg2.png	lib/views/introduction/intro_screen.dart
	assets/images/introImg3.png	lib/views/introduction/intro_screen.dart
Verify Phone screen	assets/images/verifyPhone.png	lib/views/authentication/verify_phone_screen.dart
Dashboard screen	assets/images/appbarImg.png	lib/views/dashboard/dashboard_screen.dart
	assets/images/occupation.png	lib/views/dashboard/dashboard_screen.dart

ii. Fonts used in the app. If you want to change then you can make the changes in the **pubspec.yaml** file and the **Assets** folder.

Poppins	assets/fonts/Poppins-Bold.ttf
	assets/fonts/Poppins-Regular.ttf
	assets/fonts/Poppins-SemiBold.ttf
	assets/fonts/Poppins-Medium.ttf
	assets/fonts/Poppins-Light.ttf
	assets/fonts/Poppins-Thine.ttf

iii. Colors used in the app. If you want to change the colors you can make the changes in the file **lib/Theme/nativeTheme.dart**

#	Color code
Primary Color	#ff5a60
Primary Color Dark	#1C3079
Primary swatch color	#ff5a60
Text button - background color	#ff5a60
Card - color	white
Card - shadowColor	#EEEEEE

TextFormField - filled color	white
AppBar theme - color	#ff5a60
Checkbox - check color	white
Checkbox - fillColor	#ff5a60
scaffoldBackgroundColor	white
Dialog -background color	white

iv. Packages used in the app are listed below. You can find them in **pubspec.yaml** file.

Package Name - Version	Description
pinput - ^2.2.31	For the OTP text field
font_awesome_flutter - ^10.4.0	To access the icon of font-awesome
dots_indicator - 2.1.0	To display dots indicator to show a position
get - 4.6.5	Stat management
get_storage - ^2.1.1	A fast, extra light key value in memory, which backs up data to disk at each operation.
flutter_svg - ^1.0.3	For drawing SVG files.
fluttermtoast - ^8.2.1	For showing a toast message
connectivity_plus - ^3.0.4	To check the connectivity of the device
http - ^0.13.5	For consuming HTTP resources
shared_preferences - ^2.1.0	To store something locally
device_info_plus - ^8.2.0	To get device info
permission_handler - ^10.2.0	This plugin provides a cross-platform API to request and check permission
image_picker - ^0.8.7+3	For picking an image
cached_network_image - ^3.2.3	To show images from the internet
intl - ^0.17.0	To Provide internationalization and localization facilities, including message translation, plurals and genders, date/number formatting and parsing, and bidirectional text.
firebase_auth - 4.6.0	To use the Firebase authentication API.

cloud_firestore - ^4.5.1	To use the cloud Firestore API.
firebase_analytics - ^10.2.0	A Flutter plugin to use the Google Analytics for Firebase API.
firebase_storage - ^11.1.0	To use Firebase cloud storage API.
firebase_messaging - 14.4.0	To use Firebase cloud messaging API.
flutter_local_notifications ^8.1.1+2	A cross-platform plugin for displaying local notifications.
firebase_core - 2.12.0	To use the Firebase Core API, which enables connecting to multiple Firebase apps.
firebase_dynamic_links - ^5.1.0	To use Firebase dynamic API links.
firebase_performance - ^0.9.1	To use Firebase performance API.
firebase_crashlytics - ^3.3.5	To use Firebase Crashlytics API.
flutter_html - ^3.0.0-alpha.5	For rendering HTML and CSS as Flutter widgets.
google_sign_in - ^6.1.0	A secure authentication system for signing in with a Google account on Android or iOS.
flutter_facebook_auth - ^5.0.8	The easiest way to add Facebook login to your app.
sign_in_with_apple - ^4.3.0	To support login via an Apple ID.
google_maps_flutter - ^2.2.5	To provide a Google map widget.
geocoding - ^2.1.0	To provide easy geocoding and reverse geocoding features.
geolocator - ^9.0.2	To provide access to platform-specific locations.
email_otp - ^2.0.1	It generates OTP on the recipient's email which can be used.
aligned_dialog - ^0.0.6	It allows the user to open a dialog aligned with the associated widget.
flutter_holo_date_picker - ^1.1.0	It displays a date picker.
razorpay_flutter - ^1.3.4	For payment gateway
jiffy - ^5.0.0	It is a DateTime package for parsing, manipulating and formatting dates.
in_app_purchase - ^3.1.5	It supports in-app purchases underlying the Apple App Store and Google Play Store.
agora_rtc_engine - ^6.1.0	It provides real-time voice and video communications.
stop_watch_timer - ^2.0.0	Counter time in the app.
flutter_share - ^2.0.0	Share a message or links.

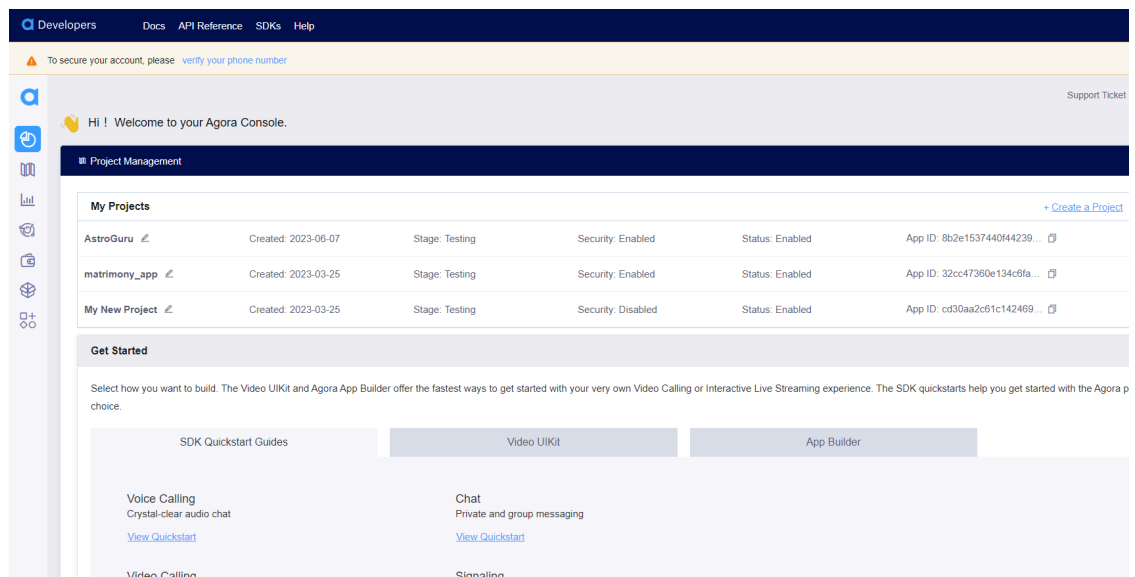
path_provider - ^2.0.14	It finds commonly used locations on the file system.
webview_flutter - ^3.0.4	It provides a webview widget.
file - ^6.1.4	It is a generic file system for abstraction for Dart.
awesome_notifications - ^0.7.4+1	Create a local notification and push notifications.
customizable_space_bar	AppBar which changes the content with scrolling rate. Enables to implement "Large Title"
phonepe_payment_sdk - ^2.0.1	For payment gateway
flutterwave_standard - ^1.0.7	For payment gateway
carousal_slider - ^4.2.1	It provides a carousal slider widget.
flutter_slidable - ^3.0.1	A Flutter implementation of slidable list item with directional slide actions that can be dismissed.

3. Third-Party Integration

a. Setup Agora

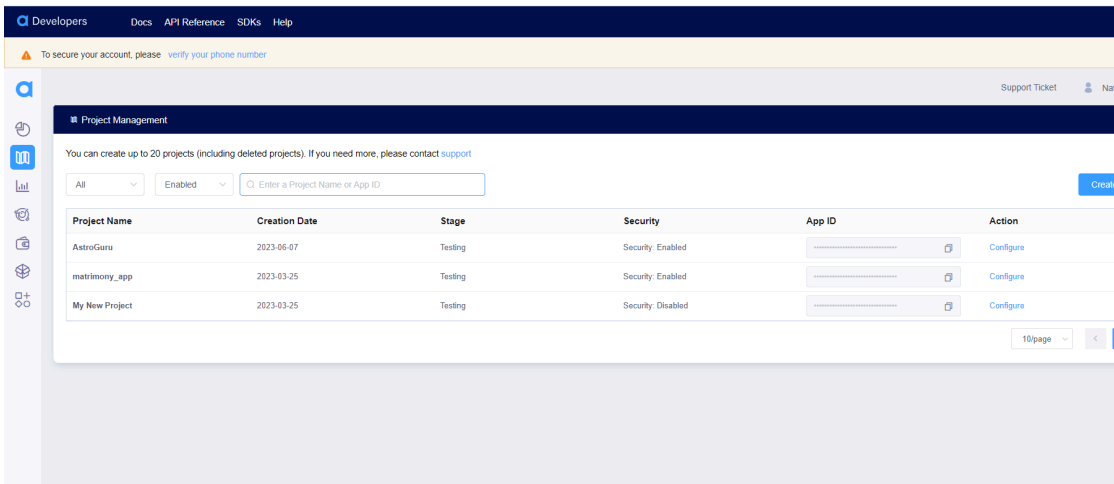
The Agora account is used to make a video call between 2 users in the mobile app.

i. Log in or sign up for the [Agora Console](#)

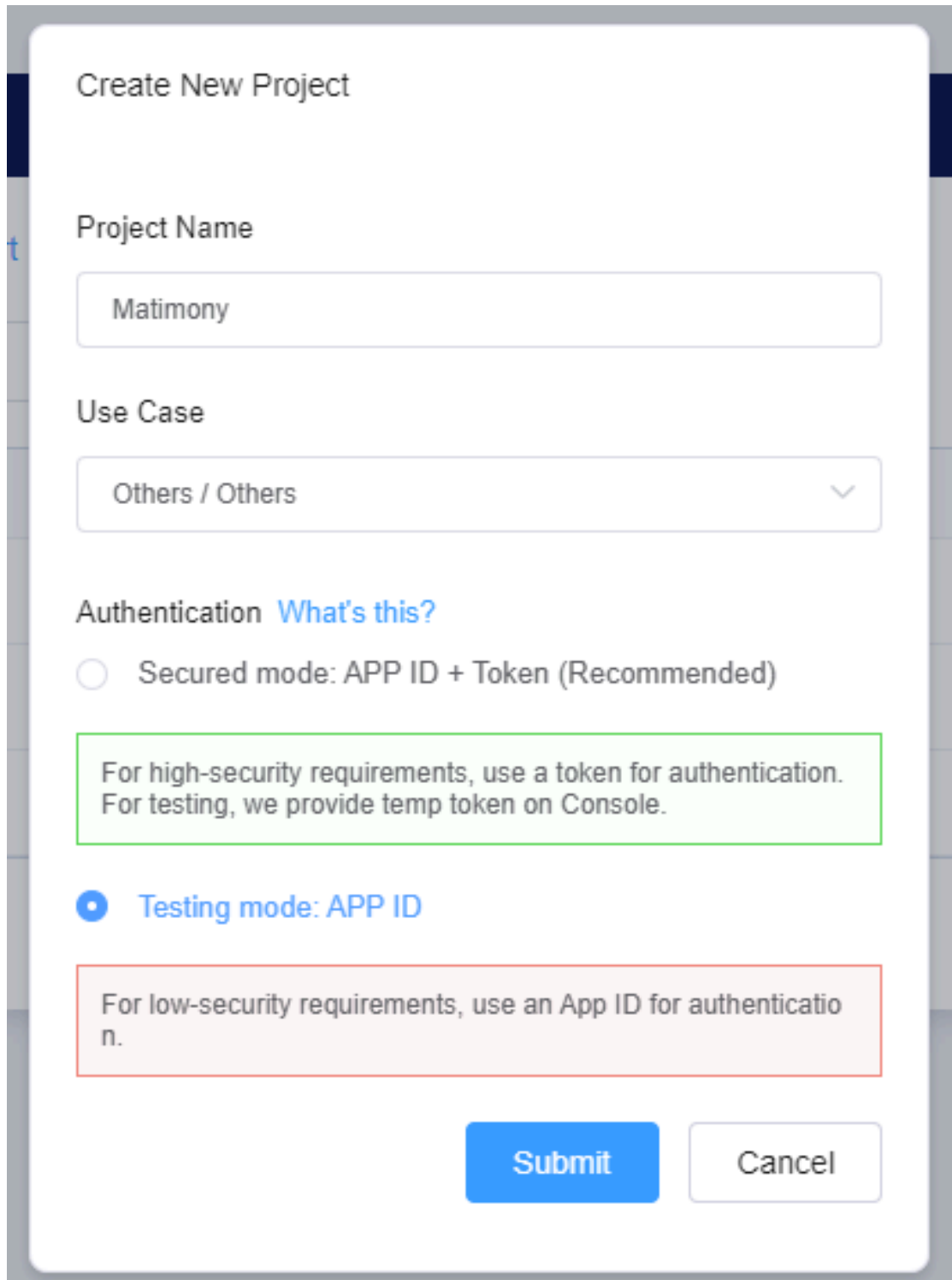


ii. Create Agora Project

1. From the side menu, Click on Project Management



2. Click on Create a Project Button



Create New Project

Project Name

Matimony

Use Case

Others / Others

Authentication [What's this?](#)

☐ Secured mode: APP ID + Token (Recommended)

For high-security requirements, use a token for authentication.
For testing, we provide temp token on Console.

☒ Testing mode: APP ID

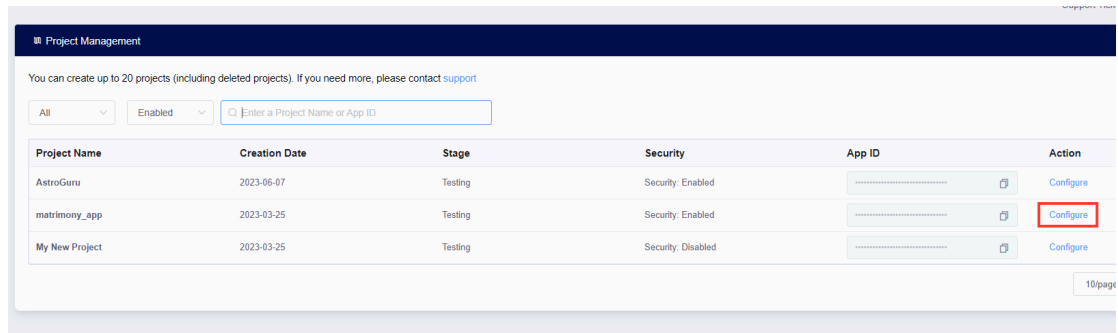
For low-security requirements, use an App ID for authentication.

Submit Cancel

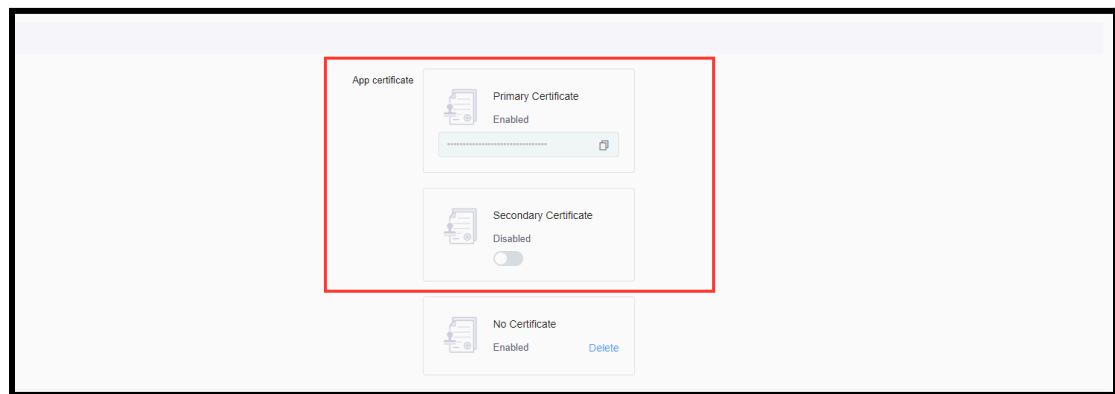
3. Give the project name select Other in Use Case and check Authentication in Testing Mode

iii. Config Project in Agora

4. Click on Configure button

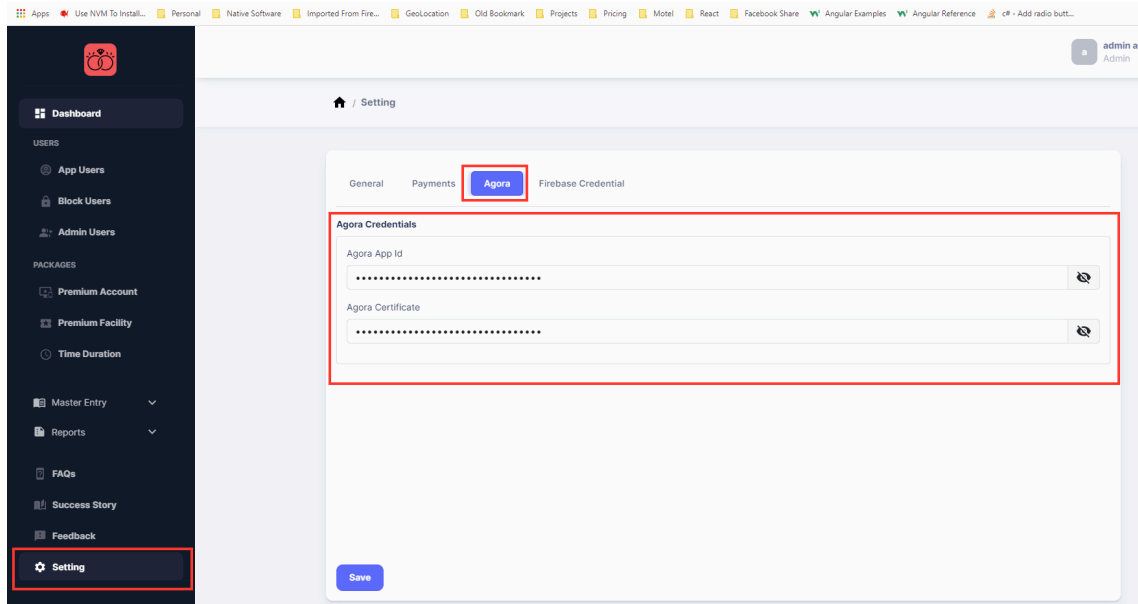


5. Enable primary certificate and Disable Secondary Certificate



iv. After Successfully Deploying the Admin panel on live (after completing step 4), Login into the Admin Panel and change the Agora key on the live

v. Change your **Agora credentials** from the **Agora** tab on the **Setting** page



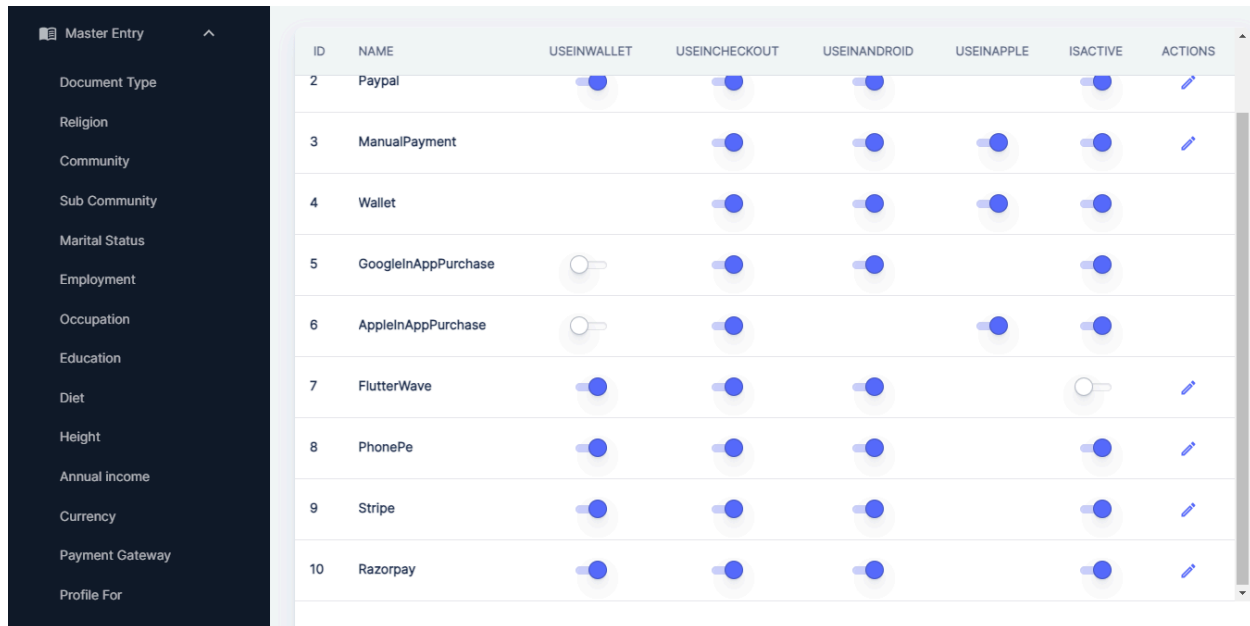
b. Razorpay Setup

- Setup RazorPay from [this link](#).
- After generating Razorpay KeyId and Razorpay Secret Key from the link, Setup them in the admin panel.
- In the **Admin Panel** Go to Master Entry from the menu and then click on the Payment Gateways, On this page change your credentials in the Razorpay Section.



c. Stripe Setup

- i. Setup Stripe from [this link](#).
- ii. After generating the Stripe Secret key from the link, set it up in the admin panel.
- iii. In the **Admin Panel** Go to Master Entry from the menu and then click on the Payment Gateways, On this page change your credentials in the Stripe Section.



ID	NAME	USEINWALLET	USEINCHECKOUT	USEINANDROID	USEINAPPLE	ISACTIVE	ACTIONS
2	Paypal	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit
3	ManualPayment	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit
4	Wallet	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
5	GoogleInAppPurchase	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
6	AppleInAppPurchase	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
7	FlutterWave	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Edit
8	PhonePe	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit
9	Stripe	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit
10	Razorpay	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit

d. PhonePe Setup

- i. Setup PhonePe from [this link](#).
- ii. Add PhonePe MerchantId, Redirect Mode, Salt Key, Salt Index, and Environment Value from the link in the admin panel.
- iii. In the **Admin Panel** Go to Master Entry from the menu and then click on the Payment Gateways, On this page change your credentials in the PhonePe Section.

ID	NAME	USEINWALLET	USEINCHECKOUT	USEINANDROID	USEINAPPLE	ISACTIVE	ACTIONS
2	Paypal	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	Edit
3	ManualPayment		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit
4	Wallet		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
5	GoogleInAppPurchase	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	
6	AppleInAppPurchase	<input type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
7	FlutterWave	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input type="checkbox"/>	Edit
8	PhonePe	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	Edit
9	Stripe	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	Edit
10	Razorpay	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	Edit

e. FlutterWave Setup

- Set up FlutterWave from [this link](#).
- Add FlutterWave MerchantId, Publickey, Secretkey, and Encryption key from the link in the admin panel.
- In the **Admin Panel** Go to Master Entry from the menu and then click on the Payment Gateways, On this page change your credentials in the FlutterWave Section.

ID	NAME	USEINWALLET	USEINCHECKOUT	USEINANDROID	USEINAPPLE	ISACTIVE	ACTIONS
2	Paypal	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	Edit
3	ManualPayment		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit
4	Wallet		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
5	GoogleInAppPurchase	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	
6	AppleInAppPurchase	<input type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
7	FlutterWave	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input type="checkbox"/>	Edit
8	PhonePe	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	Edit
9	Stripe	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	Edit
10	Razorpay	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	Edit

USEFUL LINKS

- To set up NodeJS with Typescript from scratch you can use [this link](#)
- To set up MySQL database you can use [this link](#)
- For more information on iOS refer to [this link](#)

This document was last updated on 23 October 2024.