## Technology Used

- **Flutter** with Dart Language for Mobile App
- **Angular 16** 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)

a. Install Visual Studio Code (VSCode) from this link

b. Install NodeJS from this link (Minimum version 16.14.0)

c. Install and set up Flutter from this link

d. Install MySQL from this link

(You can choose the MySQL edition based on your needs)

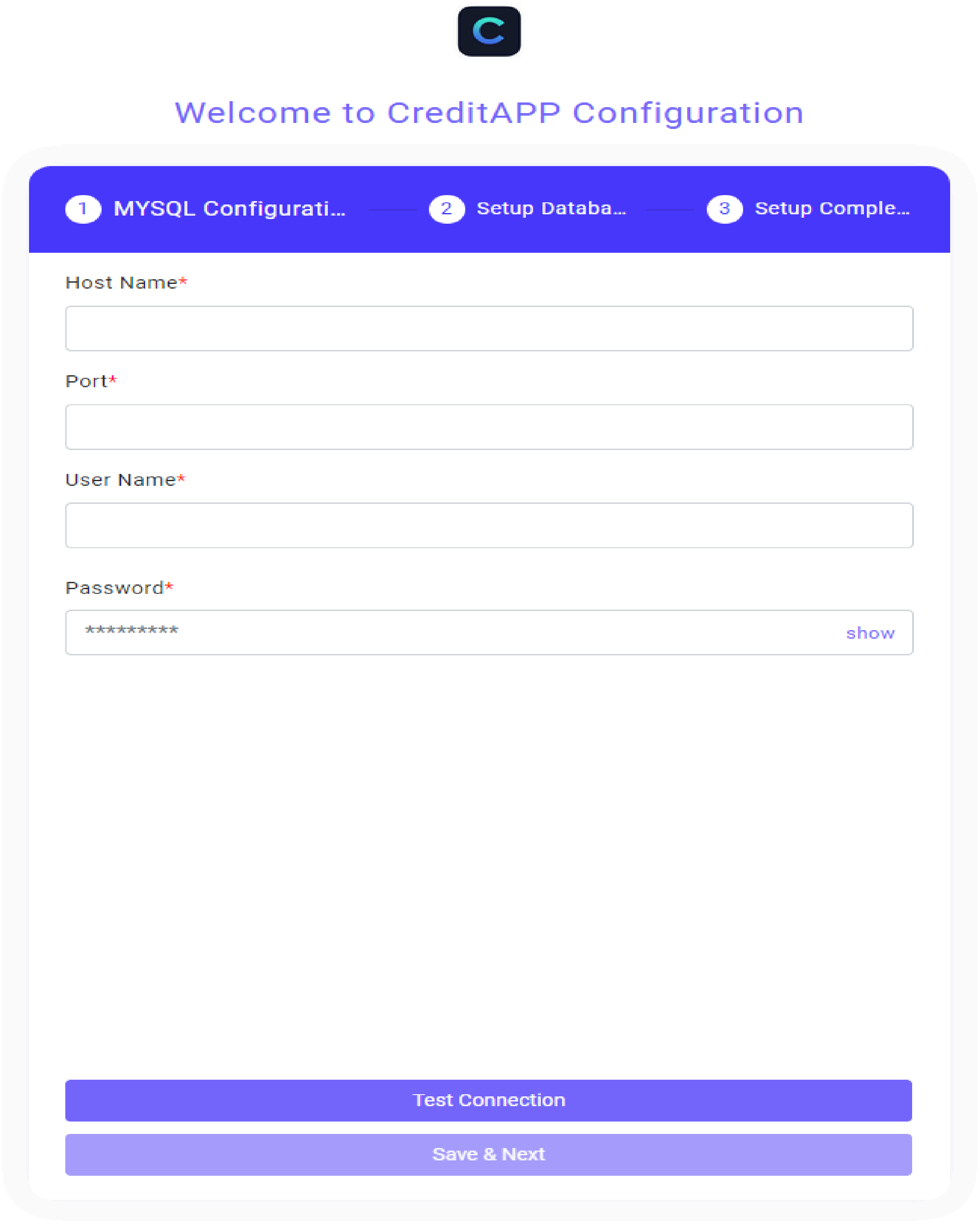
e. Install MySQL Workbench from this link (This is optional)

## 2. Setup the Database (Technology MySQL)

Database configuration could be set up through the admin panel.

**Note** :- Please ensure to restart the API server after completing the database configuration step in the admin panel before proceeding with the login process.

**Step 1: Connect with MYSQL Server**



Welcome to CreditAPP Configuration

Host Name*

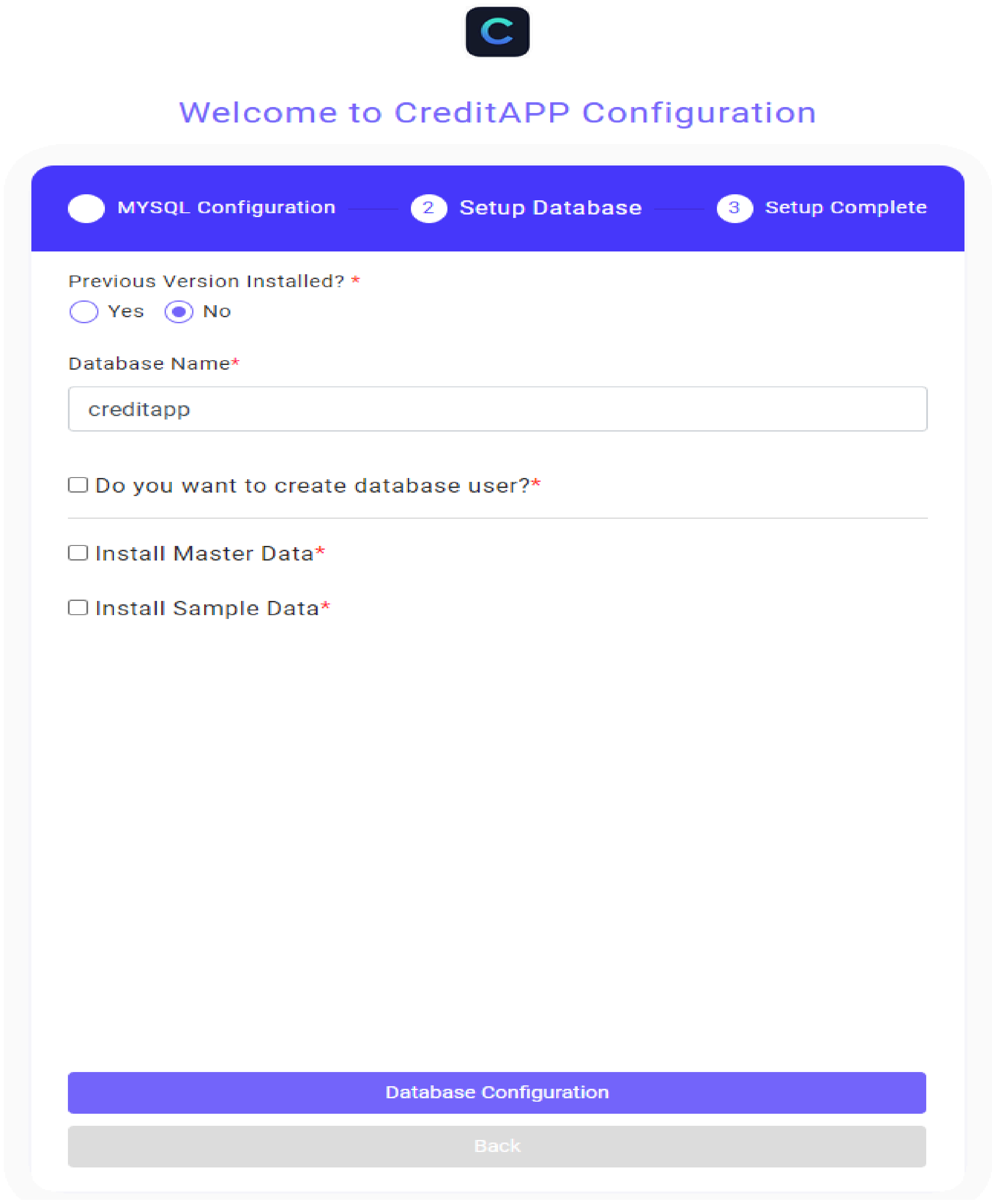
Port*

User Name*

Password*

**********

## Step 2: Setup Database



Welcome to CreditAPP Configuration

Previous Version Installed? *
Yes [✓ No]

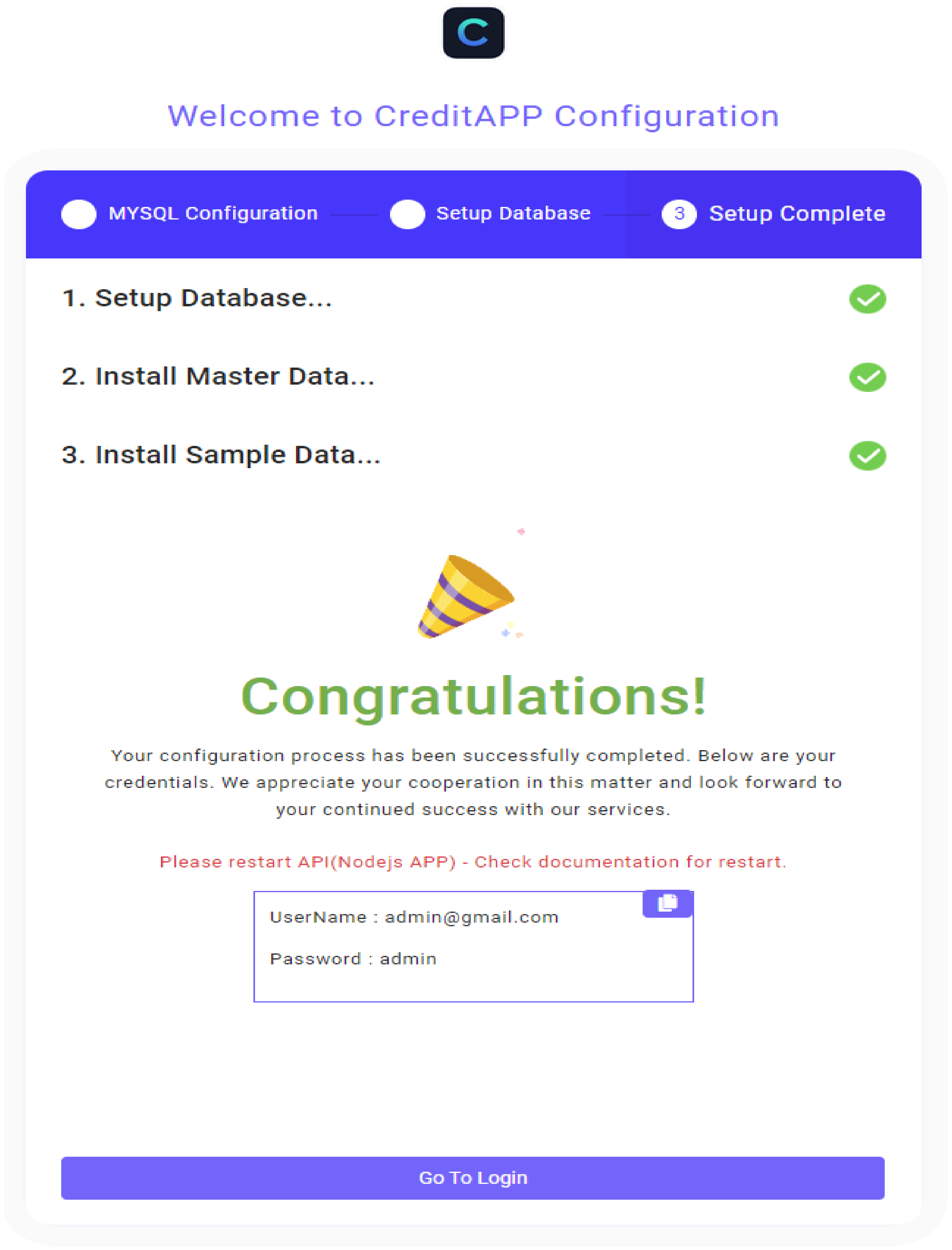
Database Name*
creditapp

☐ Do you want to create database user?*

☐ Install Master Data*

☐ Install Sample Data*

## Step 3: Complete Setup



1. Setup Database... ✓

2. Install Master Data... ✓

3. Install Sample Data... ✓

Congratulations!

Your configuration process has been successfully completed. Below are your credentials. We appreciate your cooperation in this matter and look forward to your continued success with our services.

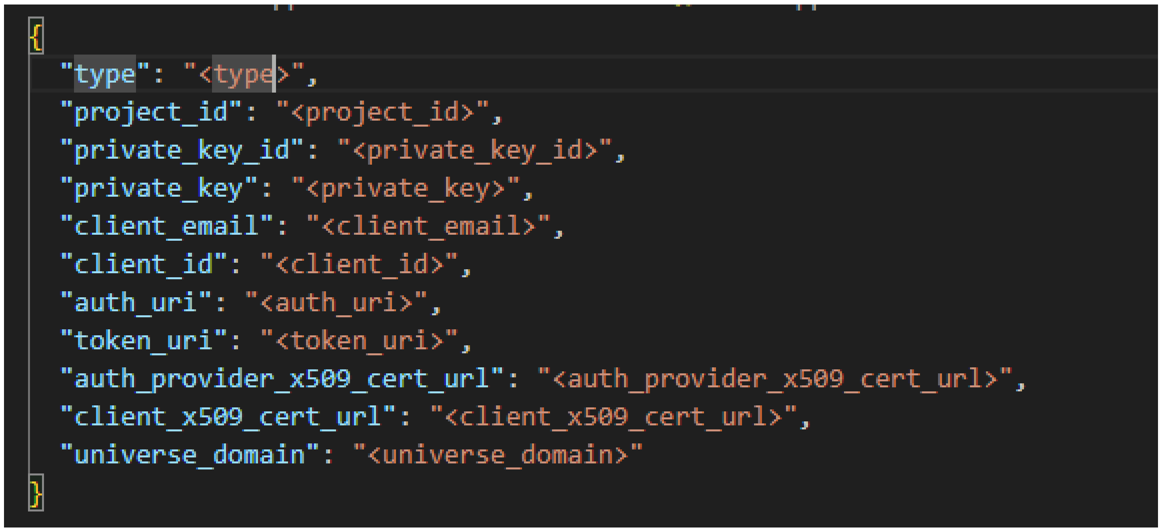
Please restart API(Nodejs APP) - Check documentation for restart.

UserName : admin@gmail.com

Password : admin

## 3. **Setup the API, Admin Panel, Website** (Technology NodeJS, Angular)

Update credit-app-9be53-firebase-adminsdk-dbci3-1141581325.json file



```
{
  "type": "<type>",
  "project_id": "<project_id>",
  "private_key_id": "<private_key_id>",
  "private_key": "<private_key>",
  "client_email": "<client_email>",
  "client_id": "<client_id>",
  "auth_uri": "<auth_uri>",
  "token_uri": "<token_uri>",
  "auth_provider_x509_cert_url": "<auth_provider_x509_cert_url>",
  "client_x509_cert_url": "<client_x509_cert_url>",
  "universe_domain": "<universe_domain>"
}
```

- Open the folder and upload it on your hosting server.
- Execute npm i command in server Terminal.
- For Accessing API use url

**https://<YOUR_DOMAIN_NAME>/api**

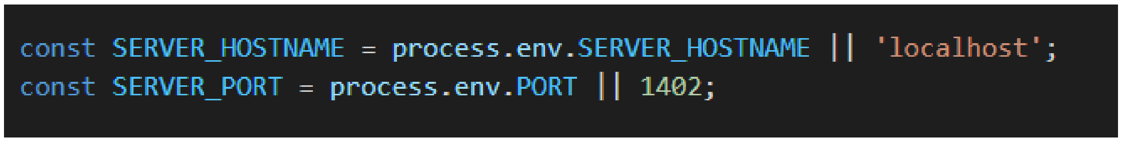
- For Accessing Admin use url

**https://<YOUR_DOMAIN_NAME>/admin**

- For Accessing API use url

**https://<YOUR_DOMAIN_NAME>**

a. Setup API Server (advanced user setup)



```
const SERVER_HOSTNAME = process.env.SERVER_HOSTNAME || 'localhost';
const SERVER_PORT = process.env.PORT || 1402;
```

b. Setup AWS S3 Bucket for Storing documents

i. To Store data in an S3 bucket, you need to generate AWS Security Key Access Credentials first.

1. Login to your **AWS Management Console**
2. Click on your username and select **My Security credential**
3. Then select **Access Keys**
4. Then Click on **Create New Access Key**
5. After that, you can either copy the Access Key ID and Secret Access Key from this window or you can download it as a .csv file

ii. Please replace “AWS_ID_S3_BUCKET” With your AWS Account ID in admin panel(AWS CREDENTIAL- AWS_ID)

iii. Please replace “AWS_SECRET_S3_BUCKET” With your AWS Account SECRET in admin panel(AWS CREDENTIAL- AWS_SECRET)

## 4. **Setup Customer App** (Technology Flutter)

### a. Initial steps to set up and run mobile app

i. Open the **App** folder in the VSCode

ii. 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
```

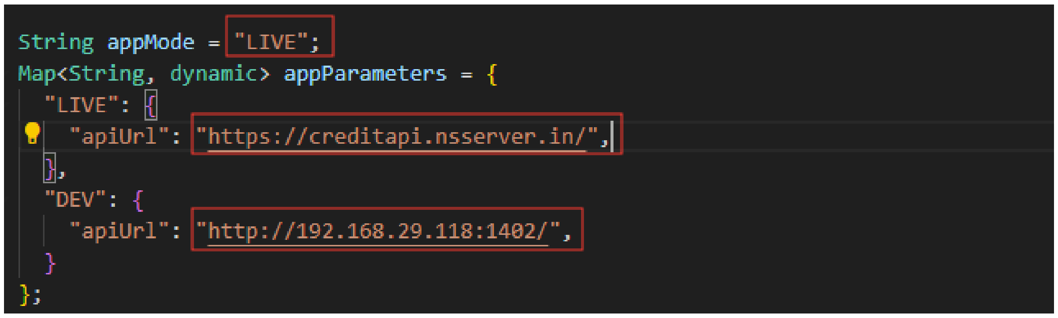
vi. To upgrade the Flutter version run the following command in the VSCode Terminal

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

```
flutter upgrade
```

## b. Change API base URL

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

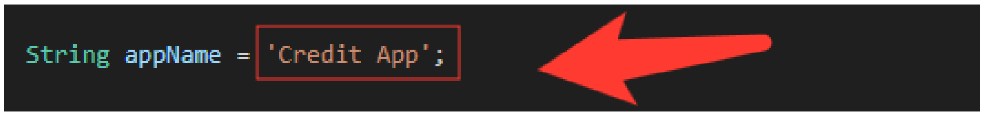


```
String appMode = "LIVE";
Map<String, dynamic> appParameters = {
  "LIVE": {
    "apiUrl": "https://creditapi.nsserver.in/",
  },
  "DEV": {
    "apiUrl": "http://192.168.29.118:1402/",
  }
};
```

## c. Change App Name

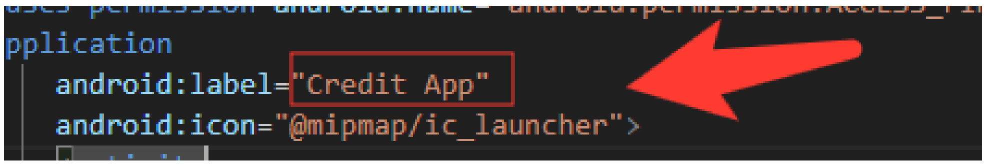
i. Change the app name in the Android App

1. Change the app name in the file located at **lib\utils\global.dart**



```
String appName = 'Credit App';
```

2. Change the app name in the file located at **android/app/src/main/AndoidManifest.xml**



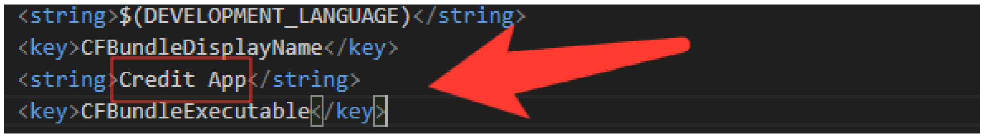
```
pplication
    android:label="Credit App"
    android:icon="@mipmap/ic_launcher">
```

ii. Change the app name in the iOS App

1. In VSCode

   a. Go to **ios/Runner/info.plist**

   b. Change string of key **CFBundleDisplayName**



```
<string>$(DEVELOPMENT_LANGUAGE)</string>
<key>CFBundleDisplayName</key>
<string>Credit App</string>
<key>CFBundleExecutable</key>
```

2. In XCode

   a. Right-click on the **iOS** folder and Choose Open in Xcode Option

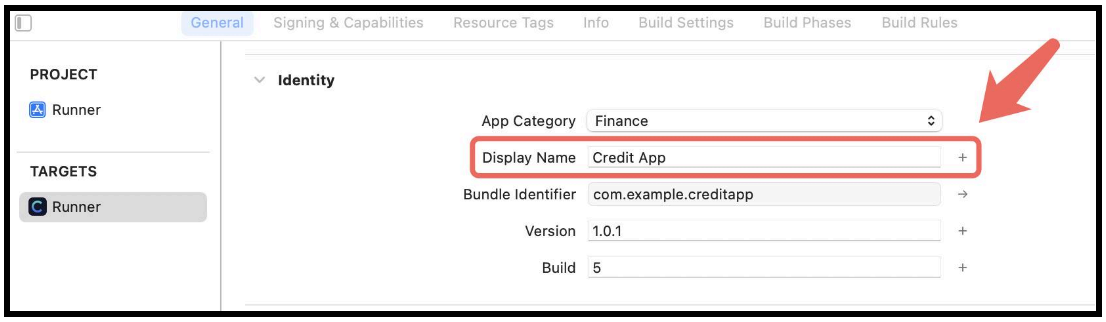
   b. Click on the folder icon left side of the XCode window

   c. Select Runner.

   d. Select Target runner

   e. From the General Tab Go to identity

   f. Change Display Name



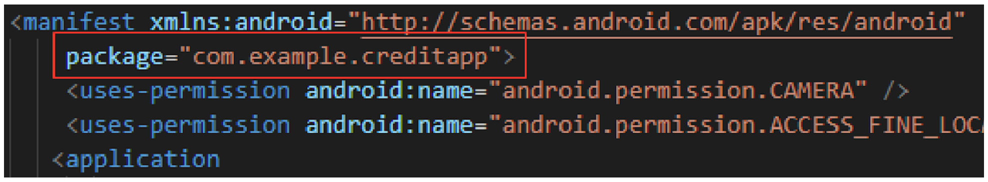
App Category: Finance
Display Name: Credit App
Bundle Identifier: com.example.creditapp
Version: 1.0.1
Build: 5

d. 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**".

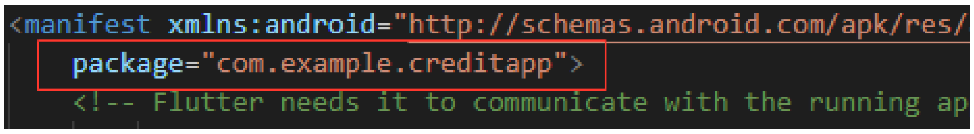
i. Set Package Name for Android App

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



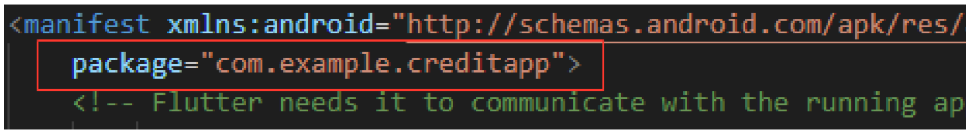
```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.creditapp">
    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOC
  <application
```

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



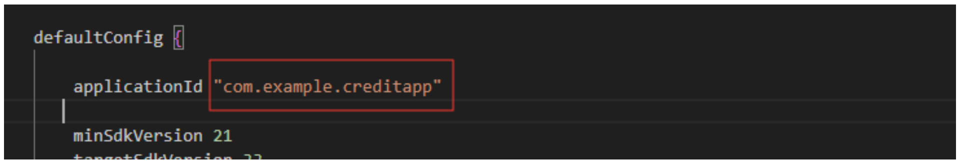
```
<manifest xmlns:android="http://schemas.android.com/apk/res/
    package="com.example.creditapp">
    <!-- Flutter needs it to communicate with the running ap
```

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



```
<manifest xmlns:android="http://schemas.android.com/apk/res/
    package="com.example.creditapp">
    <!-- Flutter needs it to communicate with the running ap
```

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

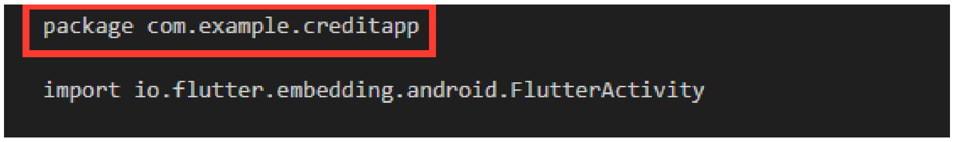


```
defaultConfig {
    applicationId "com.example.creditapp"

    minSdkVersion 21
```

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

**android\app\src\main\kotlin\com\example\creditapp\**

6. Change Package Name in the file which is located at **android\app\src\main\kotlin\com\example\creditapp\MainActivity.kt**



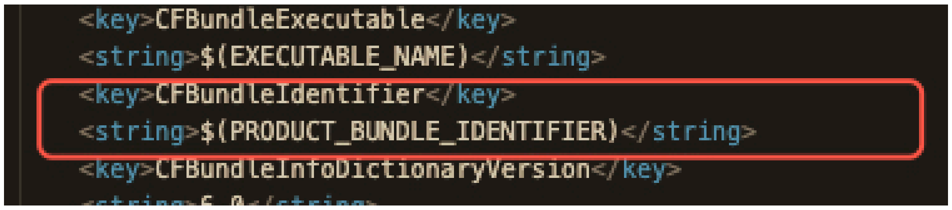
```
package com.example.creditapp

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>
```

2. In XCode

   a. Right-click on the **iOS** folder and Choose Open in Xcode Option

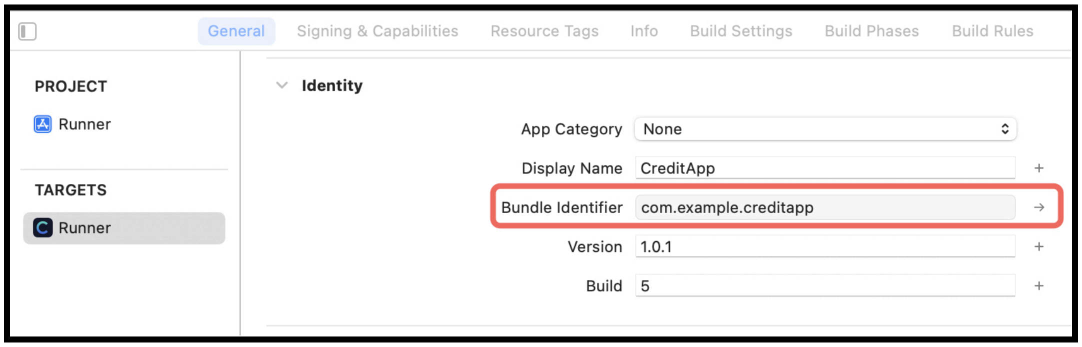
   b. Click on the folder icon left side of the XCode window

   c. Select Runner.

   d. Select Target runner

   e. In general, Tab Go to identity

   f. Change Bundle Identifier



App Category: None

Display Name: CreditApp

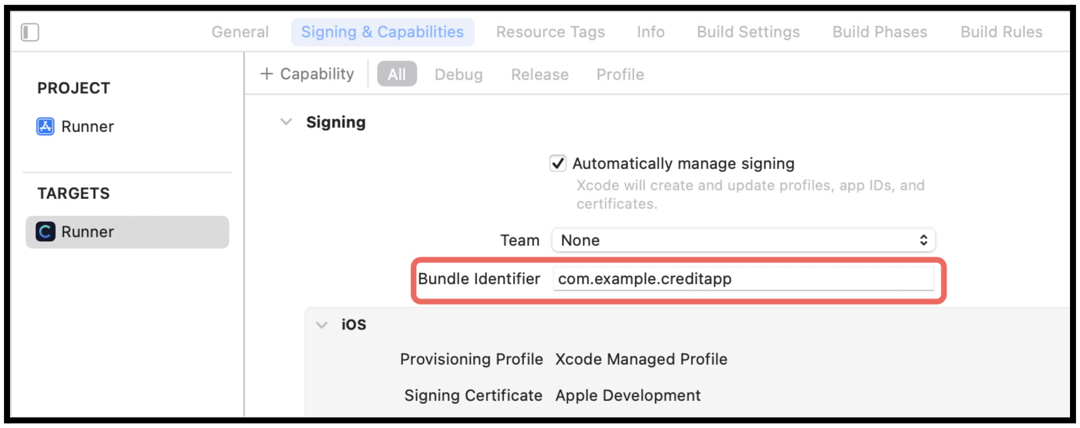
Bundle Identifier: com.example.creditapp

Version: 1.0.1

Build: 5

g. In Signing & Capabilities Go to Signing

h. Change Bundle Identifier



✓ Automatically manage signing

Xcode will create and update profiles, app IDs, and certificates.

Team: None

Bundle Identifier: com.example.creditapp

Provisioning Profile: Xcode Managed Profile

Signing Certificate: Apple Development

e. Create and set Keystore file for Android

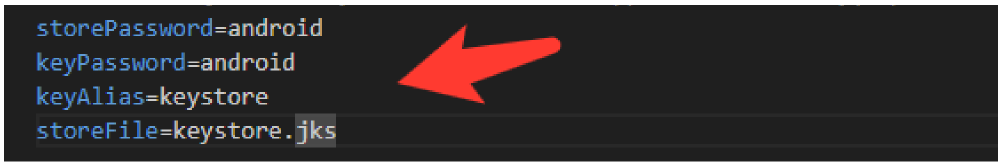
i. Create keystore.jks file if not exist using the below command in the terminal

```
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 a 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 the 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

## f. Create Firebase Account & Project

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

i. Push Notification

ii. Phone Authentication

iii. Firebase Analytics

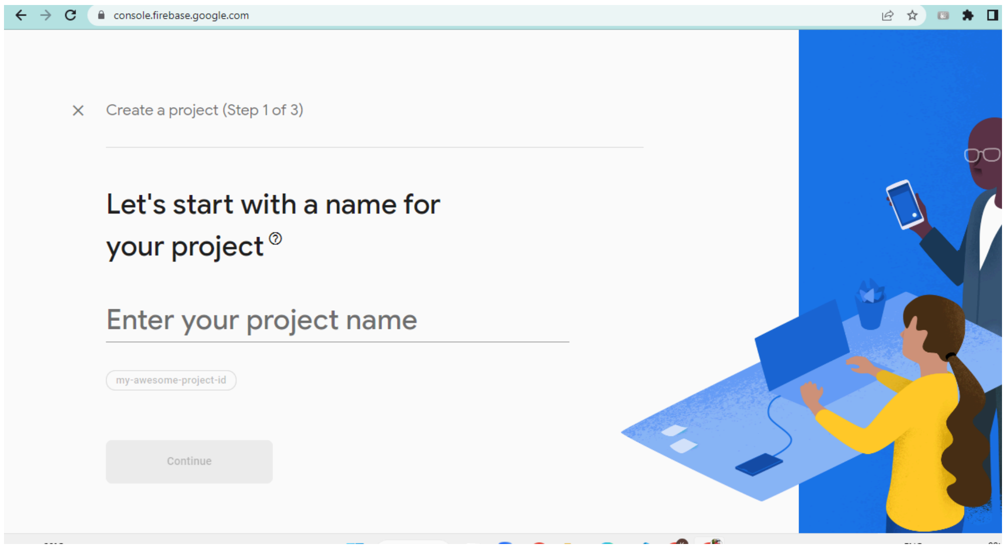
For this, you need a Firebase account and a project set up in the Firebase. Please follow the below steps for this,

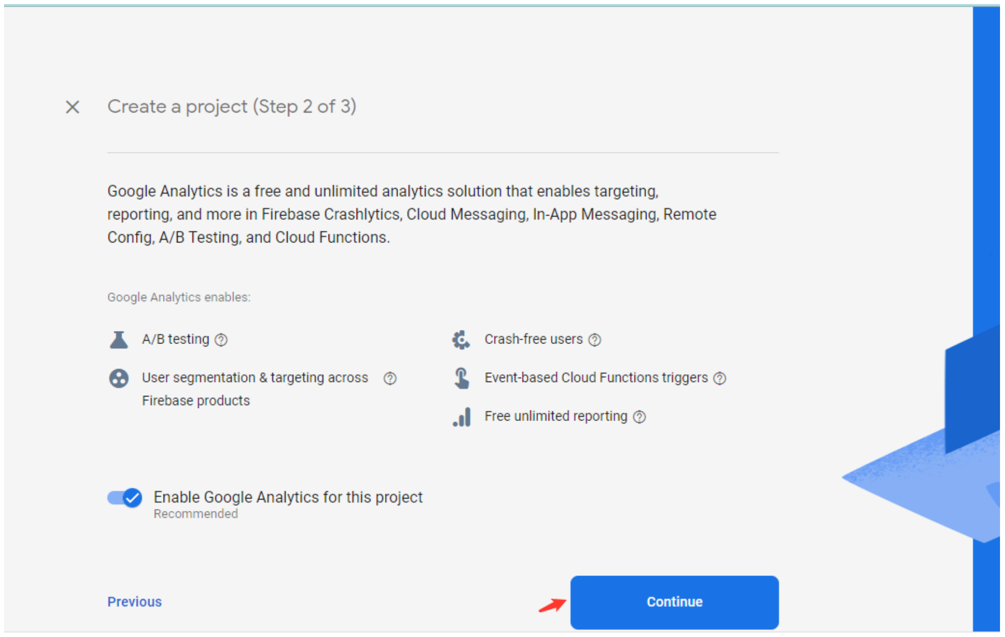
i. Go to the Firebase console

ii. Sign up if you don't have a Google Account or 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


console.firebase.google.com
Create a project (Step 1 of 3)
Let's start with a name for your project
Enter your project name
my-awesome-project-id
Continue


Create a project (Step 2 of 3)

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, and Cloud Functions.

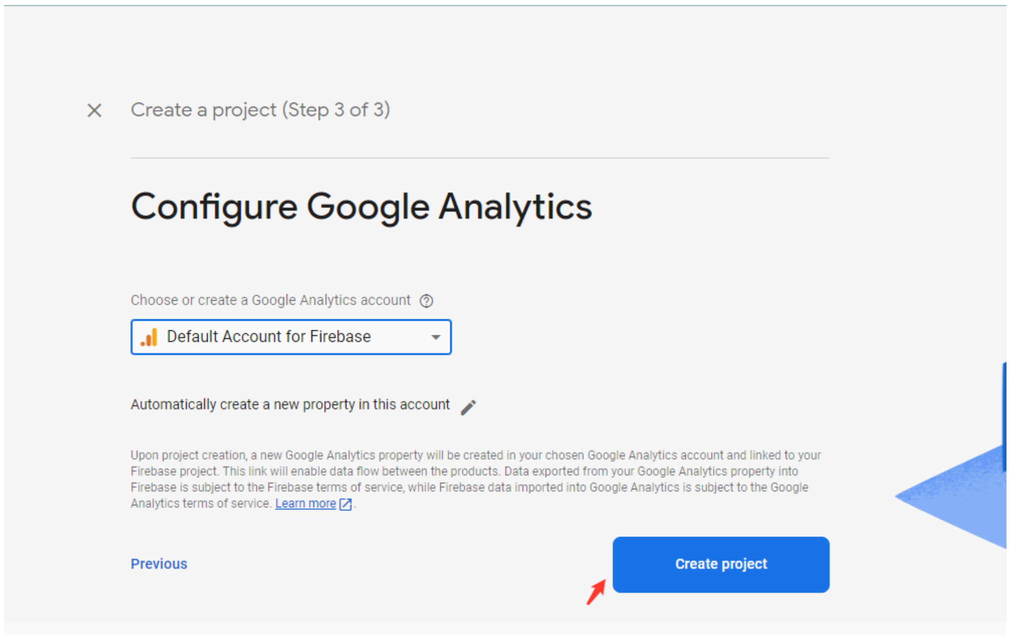
Google Analytics enables:

- A/B testing
- User segmentation & targeting across Firebase products
- Crash-free users
- Event-based Cloud Functions triggers
- Free unlimited reporting

Enable Google Analytics for this project: ✓ (Recommended)

v. Select Default Account for Firebase

(or you can create a new account)



Create a project (Step 3 of 3)

Configure Google Analytics

Choose or create a Google Analytics account: Default Account for Firebase

Automatically create a new property in this account

Upon project creation, a new Google Analytics property will be created in your chosen Google Analytics account and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. Learn more.

vi. Create project

g. Set up Android App in Firebase Project

i. Go to the Firebase console

ii. Select the project you created in step 5.d.vi

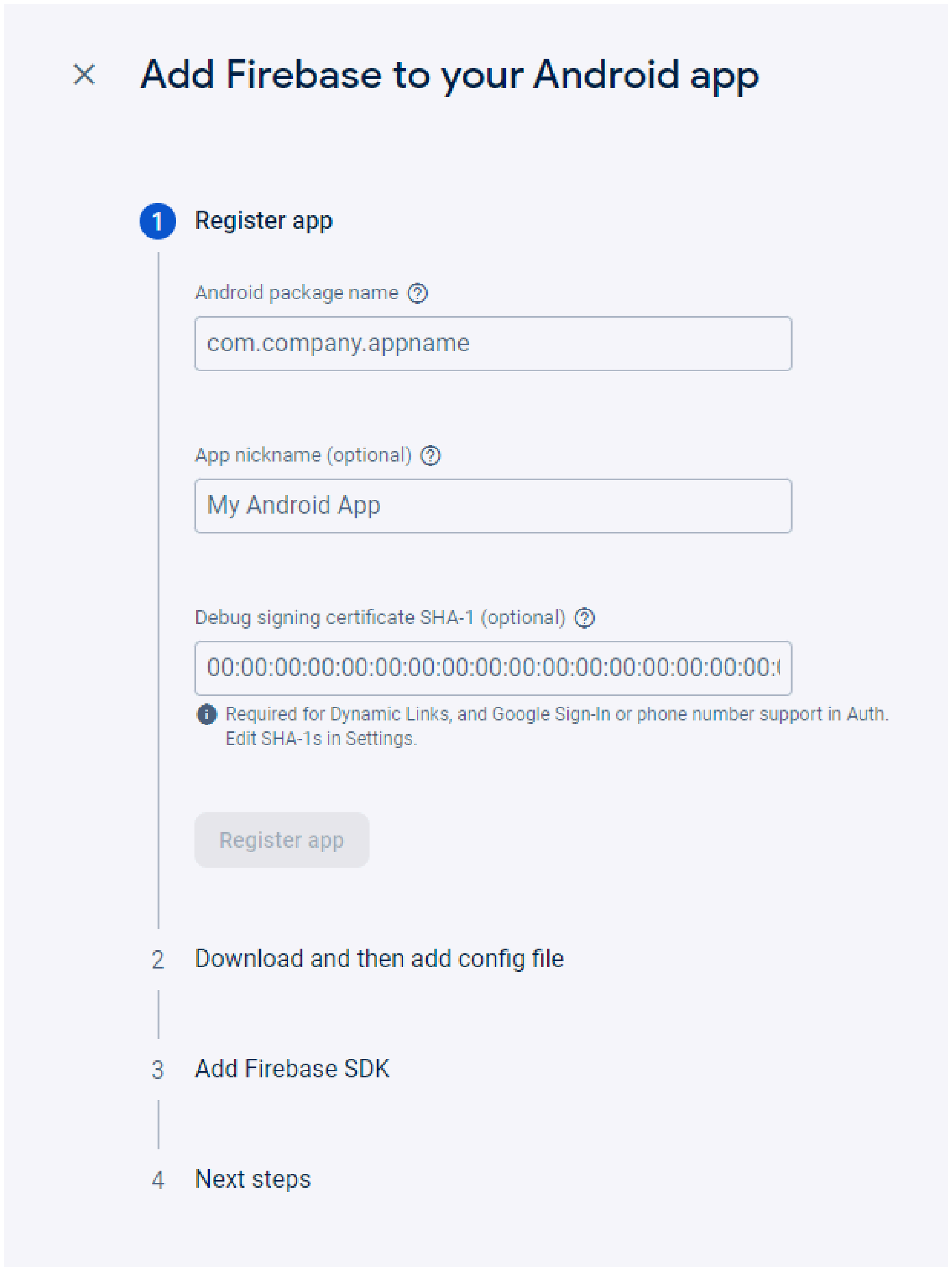
iii. Go to **Project Setting**

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)


Add Firebase to your Android app

1 Register app

Android package name

com.company.appname

App nickname (optional)

My Android App

Debug signing certificate SHA-1 (optional)

00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:0

Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

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

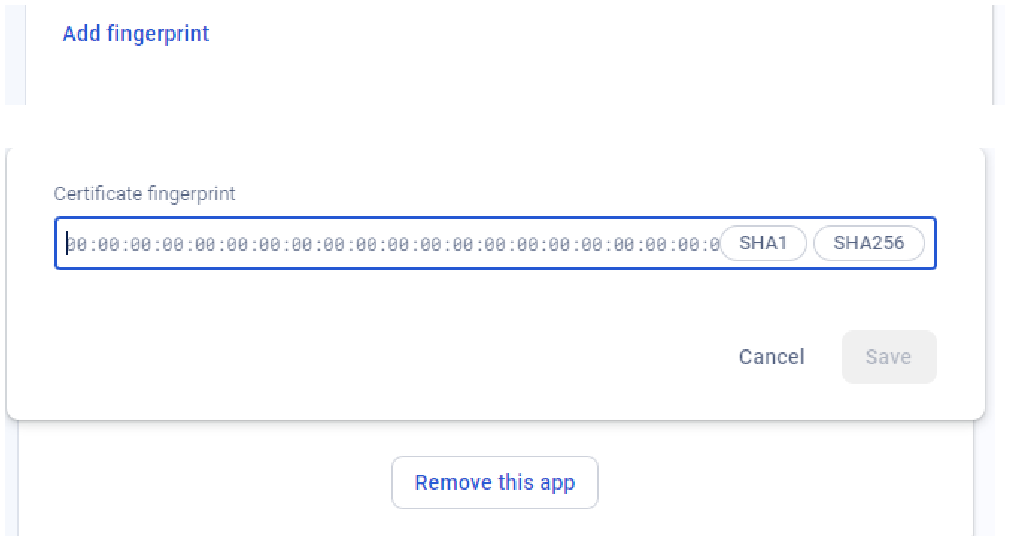
```
keytool -list -v -keystore "Your directory path\debug.jks" -alias androiddebugkey -storepass android -keypass android
```

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.



viii. Download the google-services.json file from Firebase project settings and paste it at the **android/app** location.

ix. Add Firebase SDK Add the plugin as a build script dependency to your project-level build.gradle file:

```
buildscript {
  repositories {
    // Make sure that you have the following two repositories
    google()  // Google's Maven repository
    mavenCentral()  // Maven Central repository
  }
  dependencies {
    ...
    // Add the dependency for the Google services Gradle plugin
    classpath 'com.google.gms:google-services:4.3.15'
  }
}

allprojects {
  ...
  repositories {
    // Make sure that you have the following two repositories
    google()  // Google's Maven repository
    mavenCentral()  // Maven Central repository
  }
}
```

x. Then, in your module (app-level) build.gradle file, add both the google-services plugin and any Firebase SDKs that you want to use in your app:

```
plugins {
  id 'com.android.application'
  // Add the Google services Gradle plugin
  id 'com.google.gms.google-services'
  ...
}

dependencies {
  // Import the Firebase BoM
  implementation platform('com.google.firebase:firebase-bom:31.5.0')

  // TODO: Add the dependencies for Firebase products you want to use
  // When using the BoM, don't specify versions in Firebase dependencies
  implementation 'com.google.firebase:firebase-analytics-ktx'

  // Add the dependencies for any other desired Firebase products
  // https://firebase.google.com/docs/android/setup#available-libraries
}
```

### h. Setup Firebase iOS App

i. Go to the Firebase console

ii. Select the project you created in step 5.d.vi

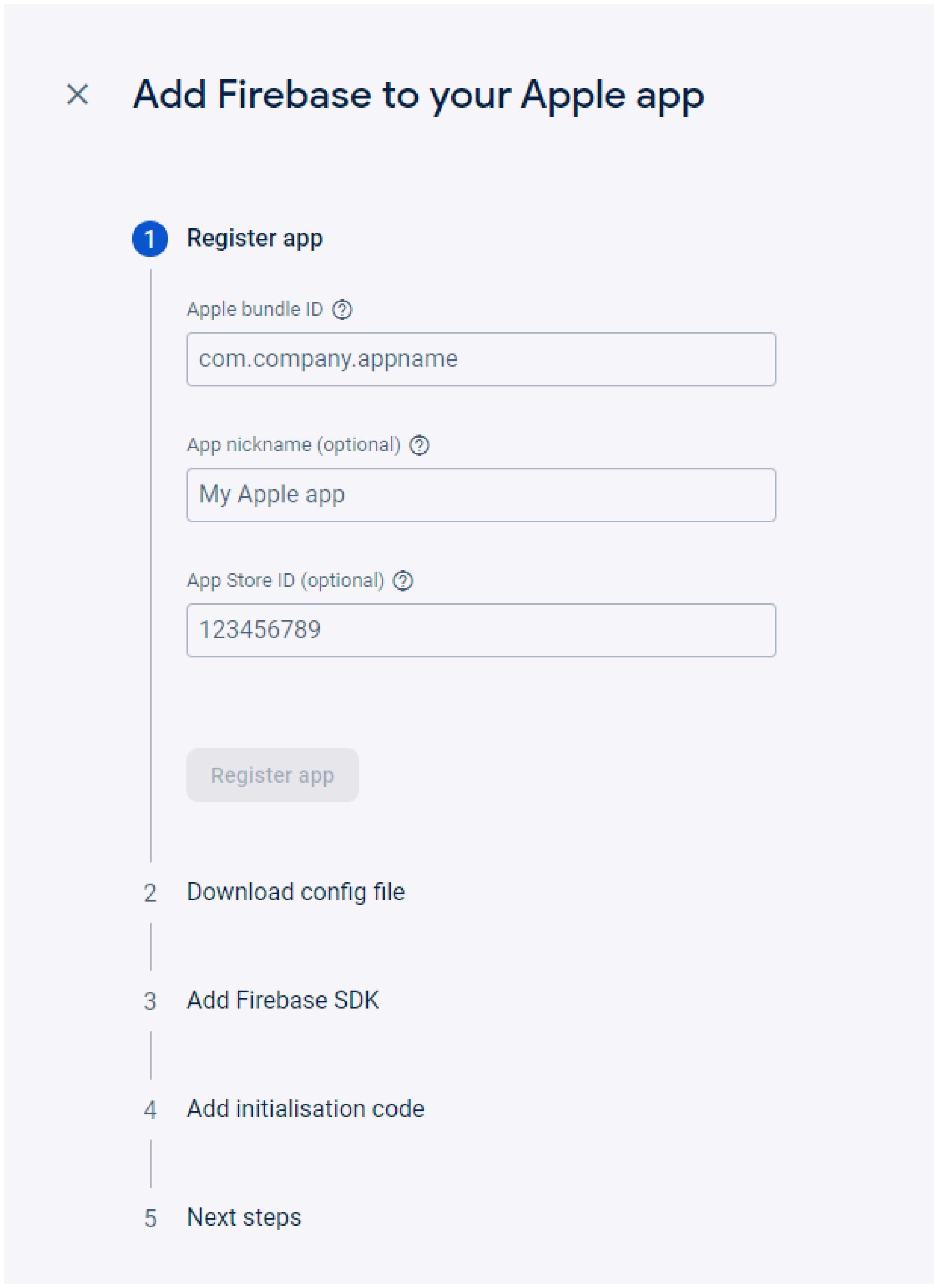
iii. Go to **Project Setting**

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)



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

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

3. 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. To view your app's settings, select the Runner target in the Xcode navigator.

4. Verify the most important settings

    a. In the Identity section of the General tab

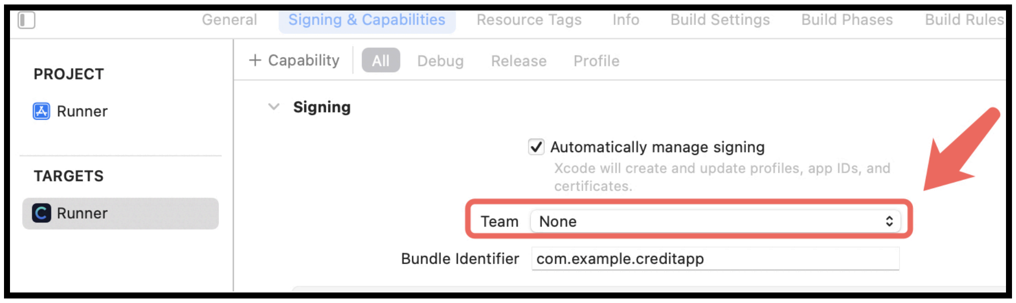
        i. **Display Name** (The display name of your app.)

        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.)



Signing

✓ Automatically manage signing
Xcode will create and update profiles, app IDs, and certificates.

Team: None

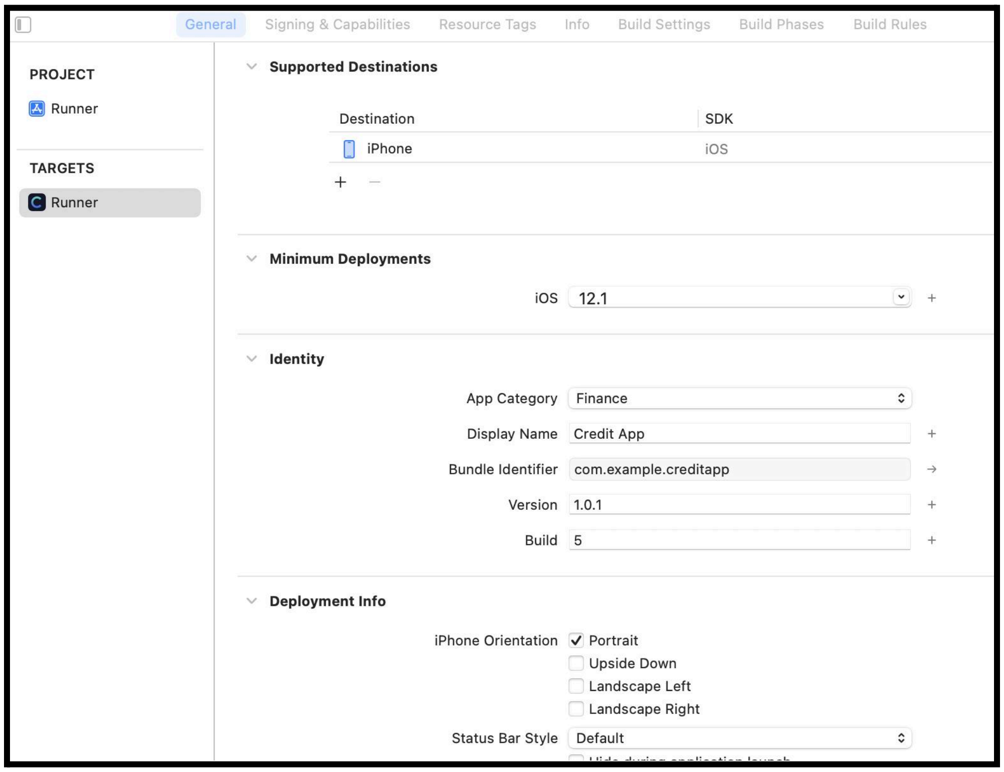
Bundle Identifier: com.example.creditapp

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:



Supported Destinations

| Destination | SDK |
|---|---|
| iPhone | iOS |

Minimum Deployments

iOS: 12.1

Identity

| | |
|---|---|
| App Category | Finance |
| Display Name | Credit App |
| Bundle Identifier | com.example.creditapp |
| Version | 1.0.1 |
| Build | 5 |

Deployment Info

iPhone Orientation: ✓ Portrait, Upside Down, Landscape Left, Landscape Right

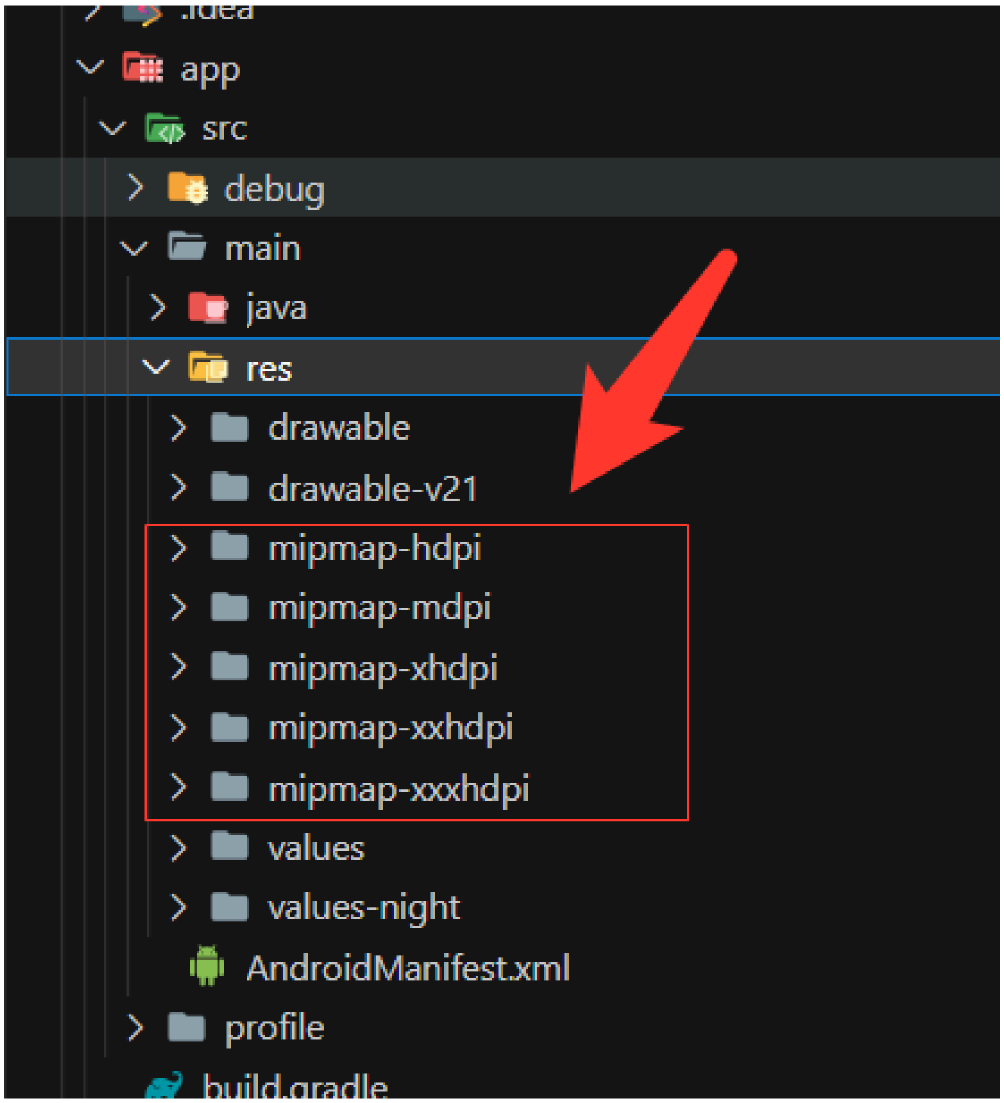
Status Bar Style: Default

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

i. Change App Icon

i. For Android

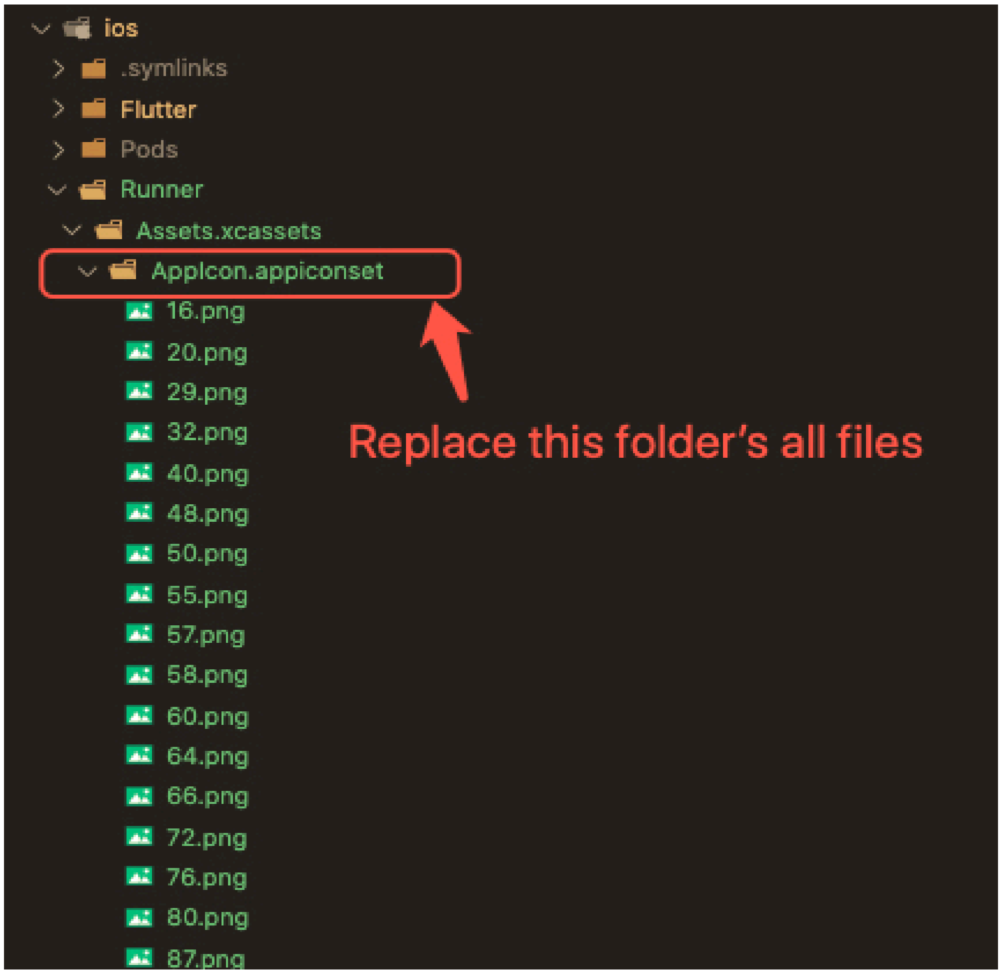
Replace the icons in the **android\app\src\main\res** folder as shown in the below image.



ii. For iOS

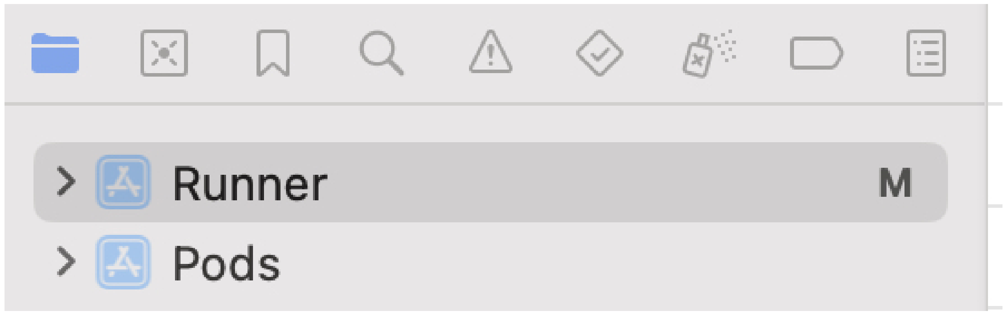
1. Replace the icons in the below folder as shown in the below image

**ios\Runner\Assets.xcassets\AppIcon.appiconset**



```
ios
  .symlinks
  Flutter
  Pods
  Runner
    Assets.xcassets
      AppIcon.appiconset
        16.png
        20.png
        29.png
        32.png
        40.png
        48.png
        50.png
        55.png
        57.png
        58.png
        60.png
        64.png
        66.png
        72.png
        76.png
        80.png
        87.png
```

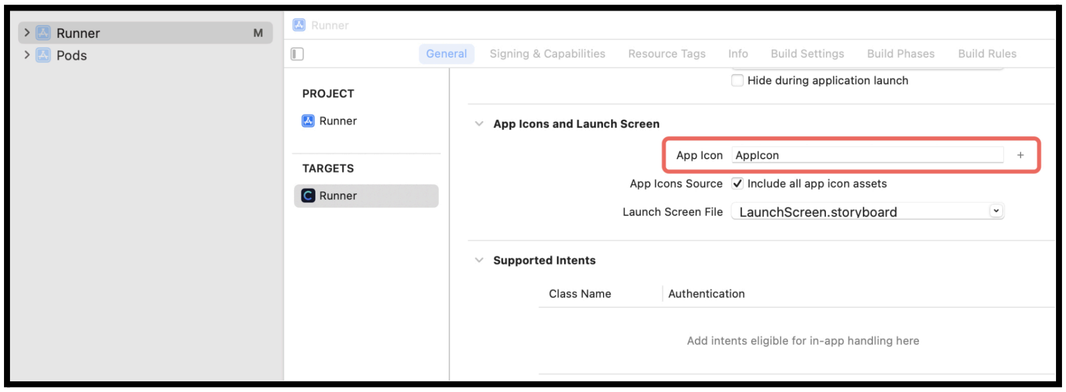
2. Change icons using XCode
   a. Right-click on the iOS folder Choose Open in Xcode Option
   b. Click on the folder icon on the left side of the XCode window



   c. Select Runner.
   d. Select Target runner

e. Go to App Icons And Launch Images

f. Click the right arrow button of the app icon source



Hide during application launch

App Icons and Launch Screen

App Icon: AppIcon

App Icons Source: ✓ Include all app icon assets

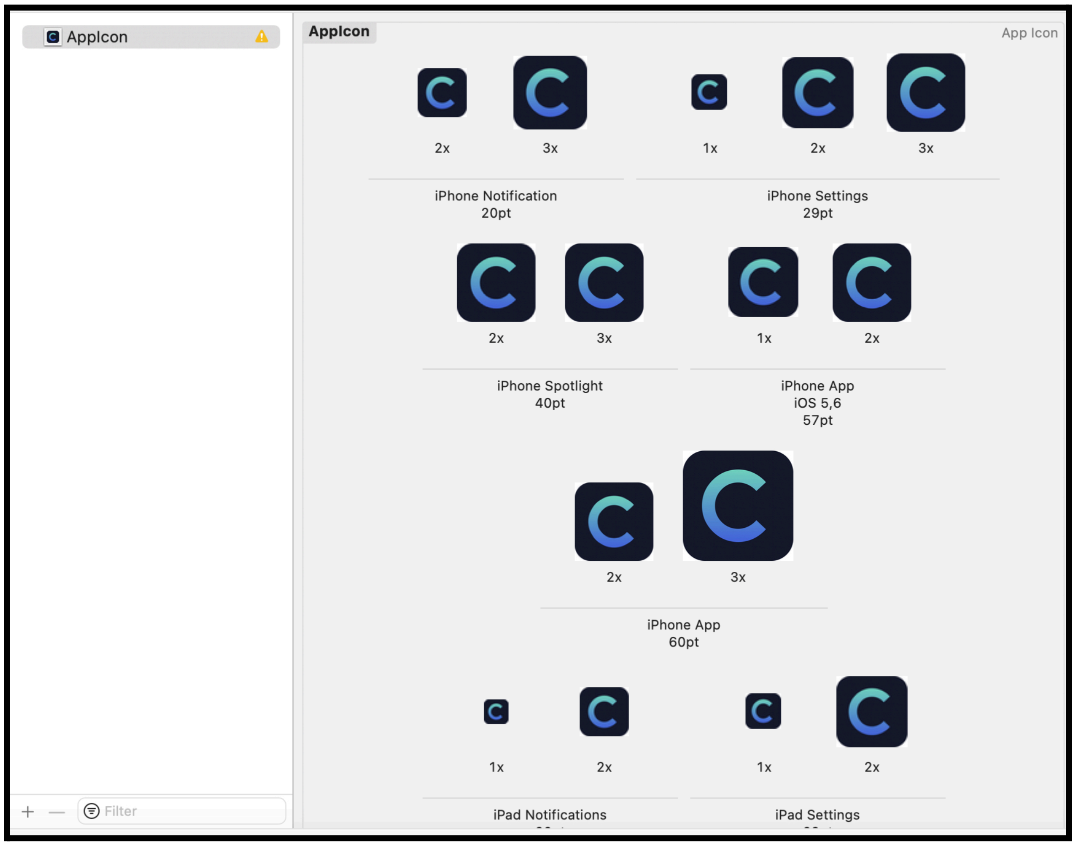
Launch Screen File: LaunchScreen.storyboard

Supported Intents

| Class Name | Authentication |
|---|---|

Add intents eligible for in-app handling here

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. 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
```

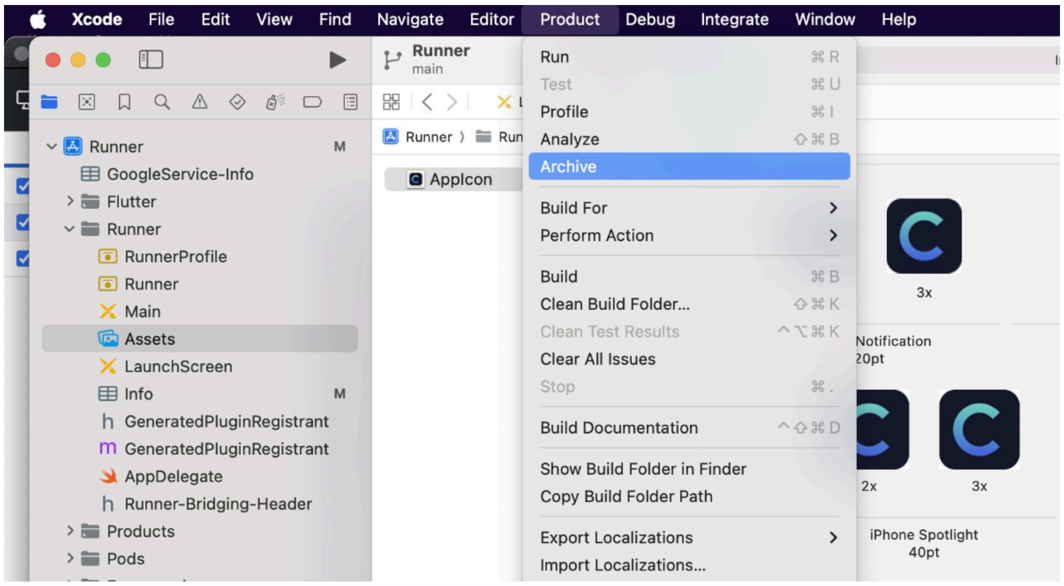
iv. Get the APK from the below path

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

k. Build Release for iOS

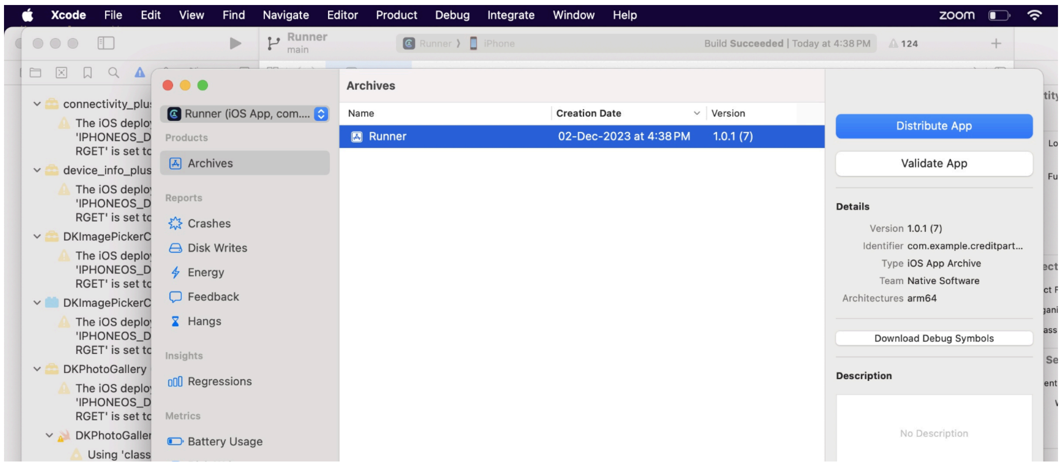
i. Open Project in XCode

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



| Name | Creation Date | Version |
|---|---|---|
| Runner | 02-Dec-2023 at 4:38 PM | 1.0.1 (7) |

Details
Version 1.0.1 (7)
Identifier com.example.creditpart...
Type iOS App Archive
Team Native Software
Architectures arm64

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]

## I. Other Options for the Advanced User

i. Paths to the images used in the app

| Images | Path | Screen Path |
|---|---|---|
| Splash screen | assets/test_logo.png | lib/views/splash/splash_screen.dart |
| Introduction screen | assets/intro_image1.png | lib/views/introduction/introduction_screen.dart |
| | assets/intro_image2.png | lib/views/introduction/introduction_screen.dart |
| | assets/intro_image3.png | lib/views/introduction/introduction_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, you can make the changes in the **pubspec.yaml** file and the **Assets** folder.

| | |
|---|---|
| Roboto | assets/fonts/Roboto-Black.ttf |
| | assets/fonts/Roboto-Bold.ttf |
| | assets/fonts/Roboto-Regular.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 | #8080ff |
| Primary swatch color | #8080ff |
| Text button - background color | #8080ff |
| Card - Color | white |
| Card - shadow color | white |
| Appbar theme - color | white |
| Checkbox - check color | white |
| Checkbox - fill color | #8080ff |
| 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 - ^2.0.5 | For the OTP text field |
| font_awesome_flutter - ^10.4.0 | To access the icon of font-awesome |
| dotted_border - 2.0.0+3 | A flutter package to easily add dotted borders around widgets |
| cupertino_icons - ^1.0.5 | For UI design |
| scratcher - ^2.3.0 | For Scratch card |
| convex_bottom_bar - ^3.2.0 | For BottomAppBar |
| share_plus - ^7.0.0 | A Flutter plugin to share content from your Flutter app via the platform's share dialog. |
| dots_indicator - 2.1.2 | To display dots indicator to show a position |

| | |
|---|---|
| flutter_document_picker - 5.2.1 | The picked document is copied |
| file_picker - ^5.3.0 | Pick single or multiple files |
| sms_autofill - ^2.3.0 | The SMS autofill is provided by default |
| mobile_number - 2.1.1 | For fetching the device's mobile number or list SIM card data |
| otp_autofill - 2.1.0 | For OTP autofill using User Consent API and Retriever API |
| open_file - ^3.3.1 | to open files |
| image_cropper - ^4.0.1 | Flexible image cropping |
| flutter_image_compress - ^2.0.3 | Compresses image as a native plugin. |
| 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 - ^2.0.5 | For drawing svg files. |
| connectivity_plus - ^4.0.0 | To check the connectivity of a device |
| http - ^0.13.6 | For consuming HTTP resources |
| shared_preferences - ^2.1.1 | To store something locally |
| loading_animation_widget - ^1.2.0+4 | Loading animation or loading spinner or loader. |
| shimmer - 2.0.0 | Easy way to add shimmer effect |
| carousel_slider - ^4.2.1 | A carousel slider widget |
| number_to_words - 1.0.0 | To convert numbers to words by |
| device_info_plus - ^9.0.0 | To get device info |
| permission_handler - ^10.2.0 | This plugin provides a cross-platform API to request and check permission |
| date_format - ^2.0.7 | A simple API to format dates |
| page_view_indicator - 2.0.0 | Customizable indicators for your PageViews. |
| image_picker - ^0.8.7+5 | For picking an image |
| cached_network_image - ^3.2.3 | To show images from the internet |
| flutter_custom_clippers - ^2.1.0 | custom clippers to help you achieve various custom shapes. |

| | |
|---|---|
| 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. |
| firebase_analytics - ^10.4.0 | A Flutter plugin to use the Google Analytics for Firebase API. |
| firebase_messaging - 14.6.0 | To use Firebase cloud messaging API. |
| flutter_local_notifications - ^14.0.0+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. |
| flutter_html - ^3.0.0-beta.1 | For rendering HTML and CSS as Flutter widgets. |
| flutter_rating_bar - ^4.0.1 | supporting any fraction of the rating. |
| google_sign_in - ^6.1.0 | A secure authentication system for signing in with a Google account on Android or iOS. |
| url_launcher - ^6.1.11 | web, phone, SMS, and email schemes. |
| fl_chart - 0.62.0 | For Line Charts, Bar charts, Pie charts, Scatter charts, and Radar charts. |
| radial_menu - 0.0.1 | For radial menu opening and revealing icons in a circle |
| Introduction_screen - ^3.1.8 | Introduction Screen allows you to have a screen on an app's first launch to, for example, explain your app |
| provider - ^6.0.5 | A wrapper around InheritedWidget to make them easier to use and more reusable |
| path_provider - ^2.0.15 | It finds commonly used locations on the file system. |
| webview_flutter - ^4.2.0 | It provides a webview widget. |
| flutter_typeahead - ^4.3.8 | A highly customizable typeahead (autocomplete) text input field for Flutter |
| substring_highlight - ^1.0.33 | Highlight Flutter text at the character level for simple and customizable search term highlighting. |
| file - ^6.1.4 | It is a generic file system for abstraction for Dart. |
| confetti - ^0.7.0 | For celebration animation |

## 5. Setup Partner App (Technology Flutter)

### a. Initial steps to set up and run mobile app

i. Open the **App** folder in the VSCode

ii. 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
```

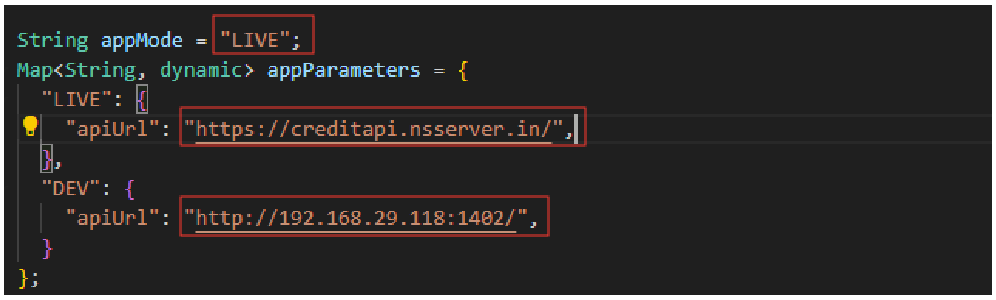
vi. To upgrade the Flutter version run the following command in the VSCode Terminal

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

```
flutter upgrade
```

### b. Change API base URL

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

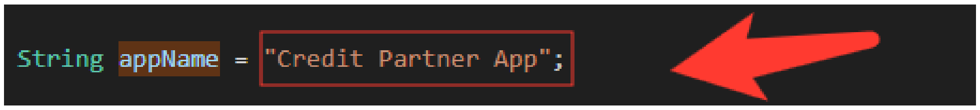


```
String appMode = "LIVE";
Map<String, dynamic> appParameters = {
  "LIVE": {
    "apiUrl": "https://creditapi.nsserver.in/",
  },
  "DEV": {
    "apiUrl": "http://192.168.29.118:1402/",
  }
};
```

c. Change App Name

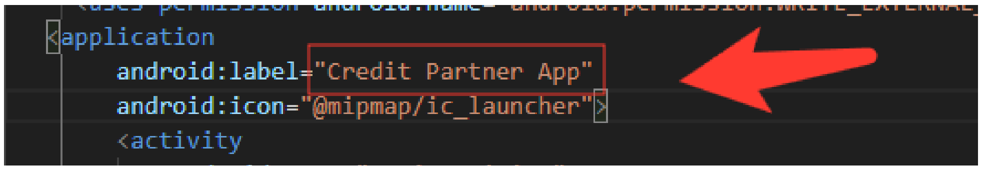
i. Change the app name in the Android App

1. Change the app name in the file located at **lib\utils\global.dart**



```
String appName = "Credit Partner App";
```

2. Change the app name in the file located at **android/app/src/main/AodidManifest.xml**



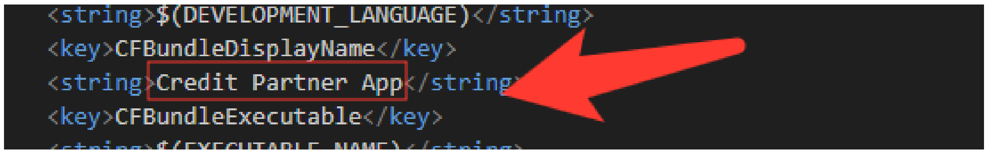
```
<application
    android:label="Credit Partner App"
    android:icon="@mipmap/ic_launcher">
    <activity
```

ii. Change the app name in the iOS App

1. In VSCode

a. Go to **ios/Runner/info.plist**

b. Change string of key **CFBundleDisplayName**



```
<string>$(DEVELOPMENT_LANGUAGE)</string>
<key>CFBundleDisplayName</key>
<string>Credit Partner App</string>
<key>CFBundleExecutable</key>
```

2. In XCode

a. Right-click on the **iOS** folder and Choose Open in Xcode Option

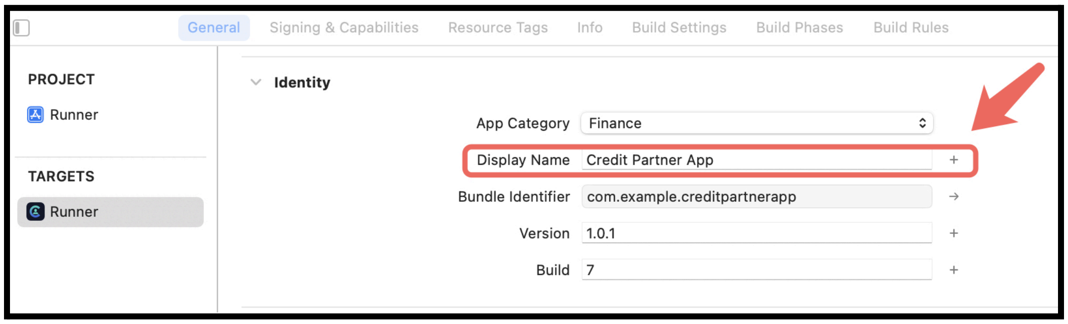
b. Click on the folder icon left side of the XCode window

c. Select Runner.

d. Select Target runner

e. From the General Tab Go to the identity

f. Change Display Name



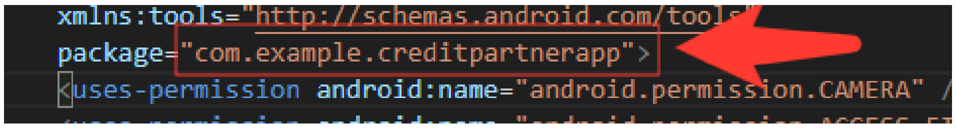
App Category: Finance
Display Name: Credit Partner App
Bundle Identifier: com.example.creditpartnerapp
Version: 1.0.1
Build: 7

## d. 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**".

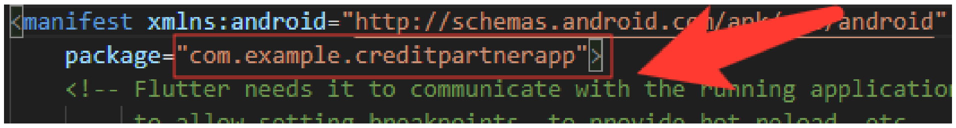
i. Set Package Name for Android App

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



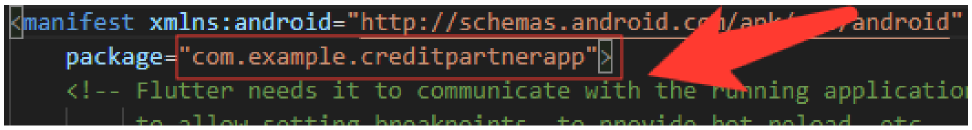
```
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.creditpartnerapp">
    <uses-permission android:name="android.permission.CAMERA" /
```

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



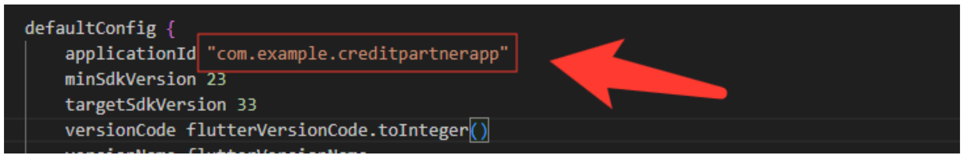
```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.creditpartnerapp">
    <!-- Flutter needs it to communicate with the running applicatio
```

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



```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.creditpartnerapp">
    <!-- 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**

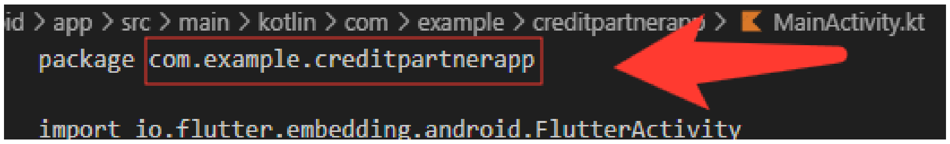


```
defaultConfig {
    applicationId "com.example.creditpartnerapp"
    minSdkVersion 23
    targetSdkVersion 33
    versionCode flutterVersionCode.toInteger()
```

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

   **android\app\src\main\kotlin\com\example\creditpartnerapp\**

6. Change Package Name in file which is located at **android\app\src\main\kotlin\com\example\creditpartnerapp\MainActivity.kt**



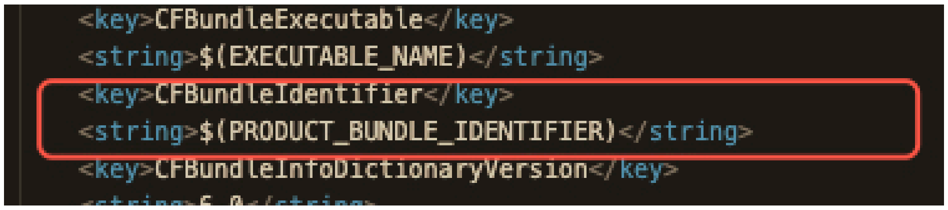
```
id > app > src > main > kotlin > com > example > creditpartnerapp > MainActivity.kt
 package com.example.creditpartnerapp

 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>6.0</string>
```

2. In XCode

   a. Right-click on the **iOS** folder and Choose Open in Xcode Option

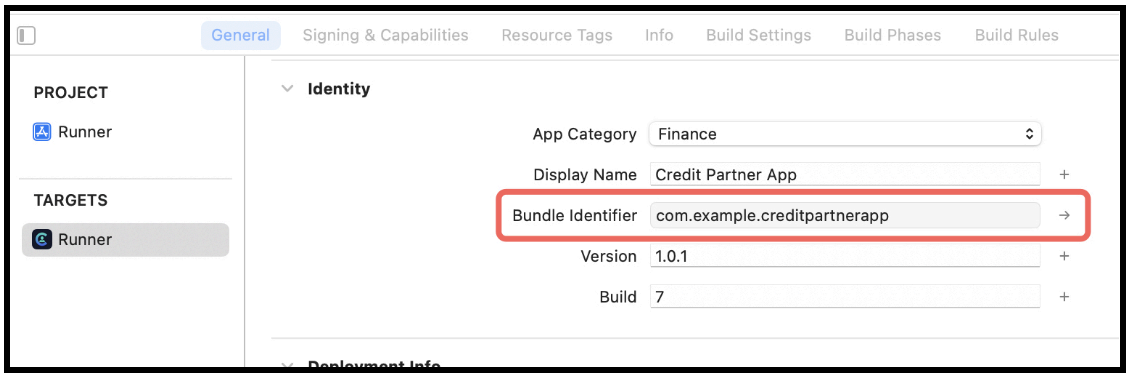
   b. Click on the folder icon on the left side of the XCode window

c. Select Runner.

d. Select Target runner

e. Go to identity

f. Change Bundle Identifier



Identity

App Category: Finance

Display Name: Credit Partner App

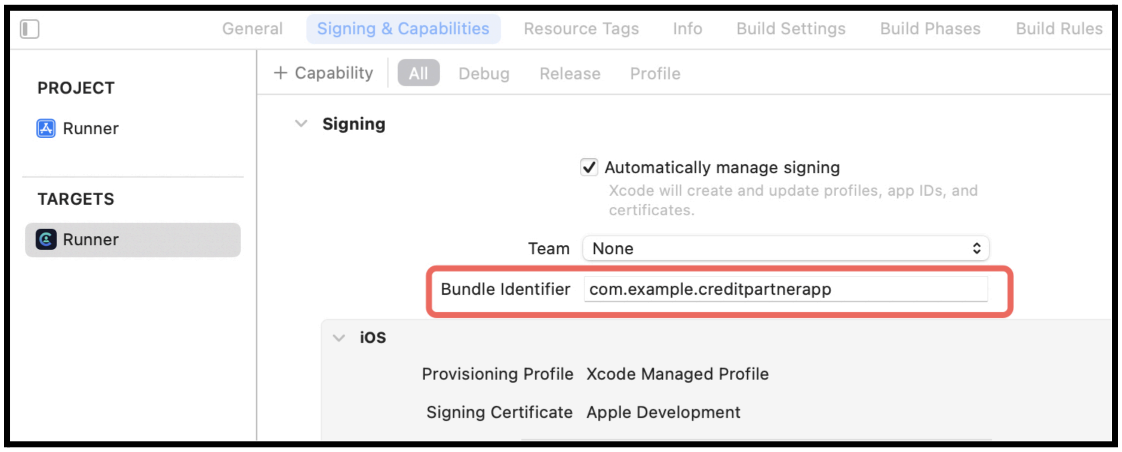
Bundle Identifier: com.example.creditpartnerapp

Version: 1.0.1

Build: 7

g. In Signing & Capabilities Go to Signing

h. Change Bundle Identifier



Signing

✓ Automatically manage signing

Xcode will create and update profiles, app IDs, and certificates.

Team: None

Bundle Identifier: com.example.creditpartnerapp

iOS

Provisioning Profile: Xcode Managed Profile

Signing Certificate: Apple Development

e. Create and set Keystore file for Android

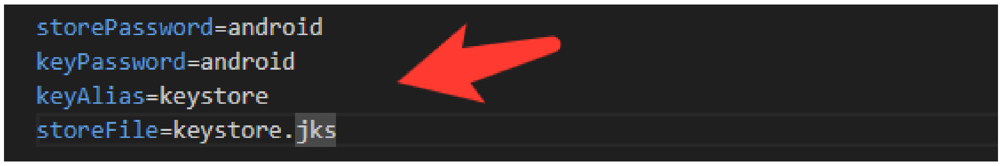
i. Create a keystore.jks file if it does not exist using the below command in the terminal

```
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 a 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 the 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

## f. Create Firebase Account & Project

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

i. Push Notification

ii. Phone Authentication

iii. Firebase Analytics

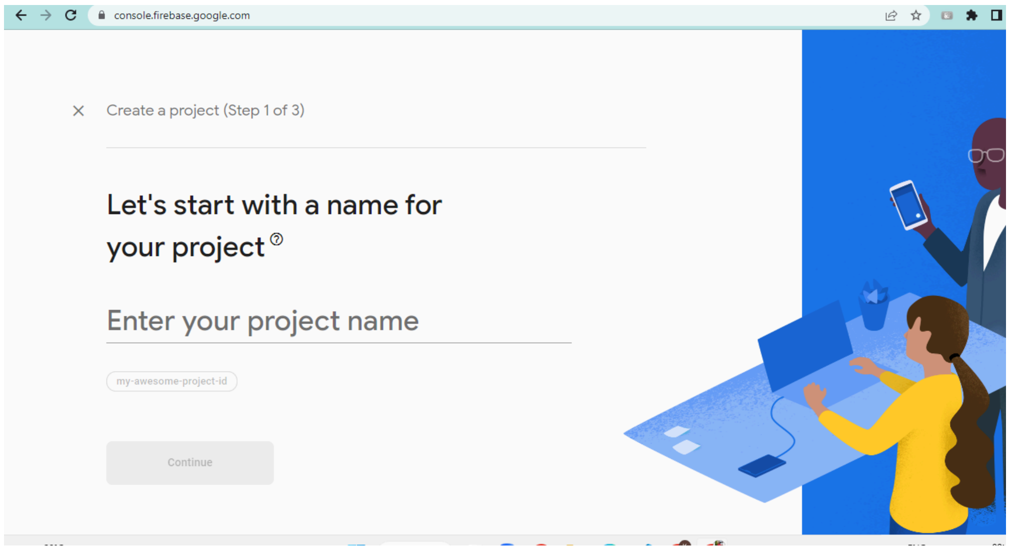
For this, you need a Firebase account and a project set up in the Firebase. Please follow the below steps for this,

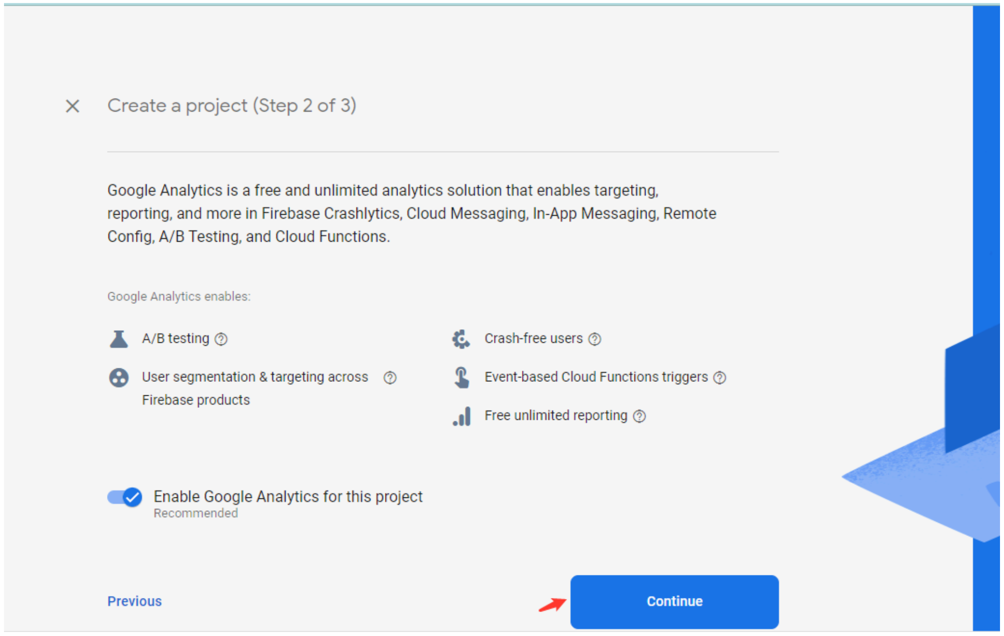
i. Go to the Firebase console

ii. Sign up if you don't have a Google Account or 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


console.firebase.google.com
Create a project (Step 1 of 3)
Let's start with a name for your project
Enter your project name
my-awesome-project-id
Continue



Create a project (Step 2 of 3)

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, and Cloud Functions.

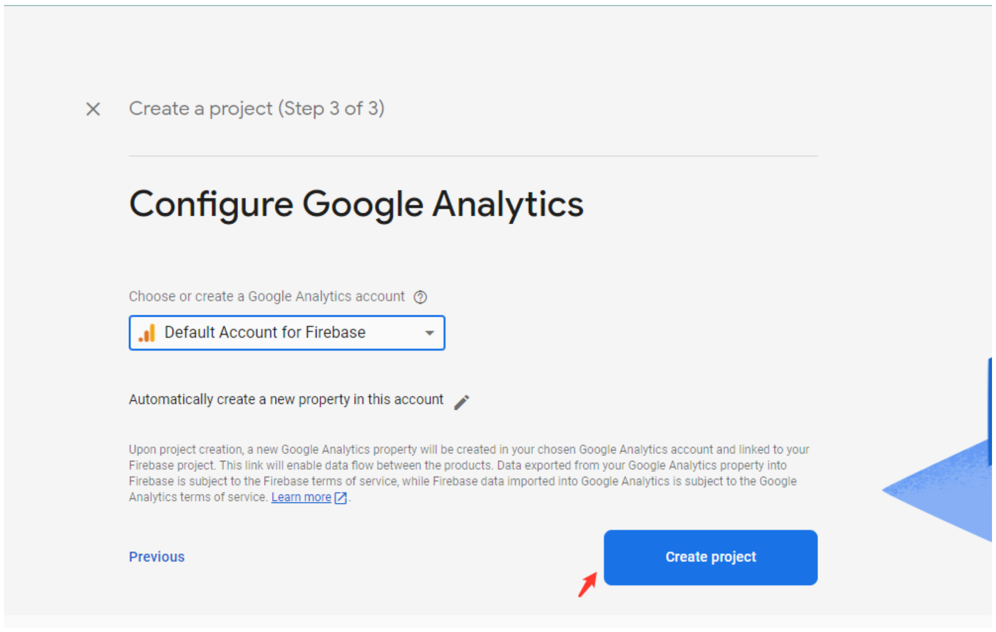
Google Analytics enables:

- A/B testing
- User segmentation & targeting across Firebase products
- Crash-free users
- Event-based Cloud Functions triggers
- Free unlimited reporting

✓ Enable Google Analytics for this project
Recommended

v. Select Default Account for Firebase

(or you can create a new account)


Create a project (Step 3 of 3)

Configure Google Analytics

Choose or create a Google Analytics account

Default Account for Firebase

Automatically create a new property in this account

Upon project creation, a new Google Analytics property will be created in your chosen Google Analytics account and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. Learn more

vi. Create project

g. Set up Android App in Firebase Project

i. Go to Firebase console

ii. Select the project you created in step 5.d.vi

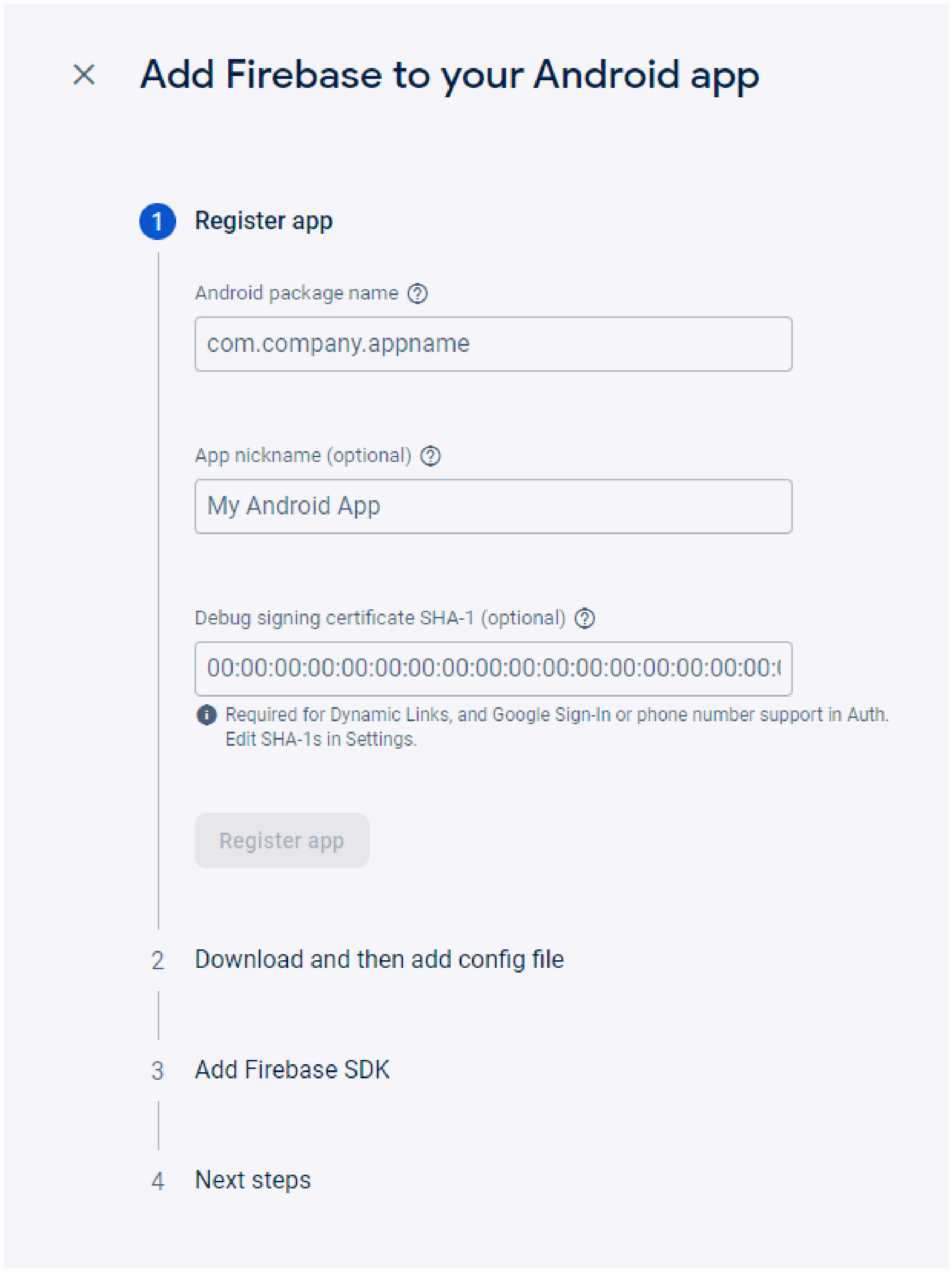
iii. Go to **Project Setting**

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)



Add Firebase to your Android app

1 Register app

Android package name: com.company.appname

App nickname (optional): My Android App

Debug signing certificate SHA-1 (optional): 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:0

Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

vii. You need SHA keys (SHA-1 and SHA-256) to add once you create the Android App in the above steps.

5. To Generate debug SHA use the below command

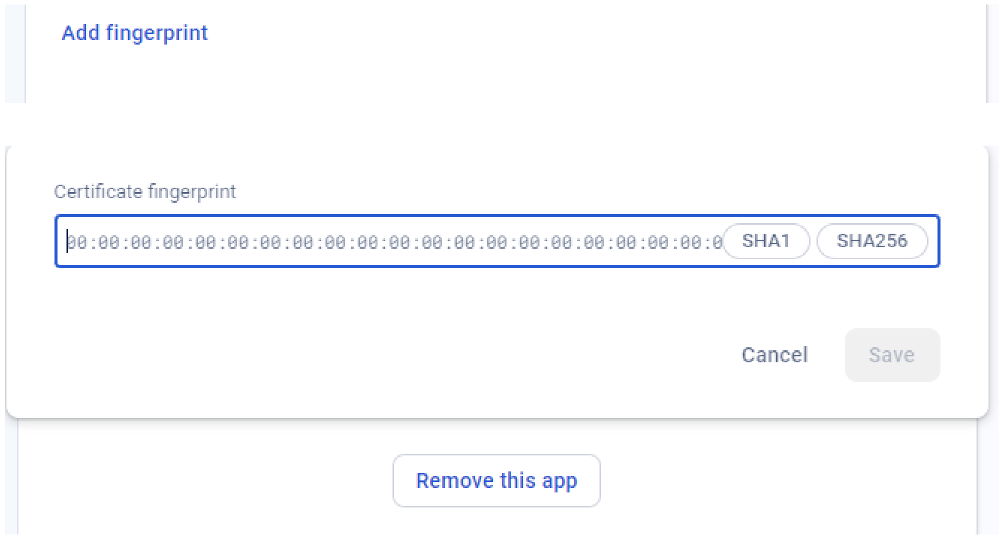
```
keytool -list -v -keystore "Your directory path\debug.jks" -alias androiddebugkey -storepass android -keypass android
```

6. 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.



viii. Download the google-services.json file from Firebase project settings and paste it at the **android/app** location.

ix. Add Firebase SDK Add the plugin as a build script dependency to your project-level build.gradle file:

```
buildscript {
  repositories {
    // Make sure that you have the following two repositories
    google()  // Google's Maven repository
    mavenCentral()  // Maven Central repository
  }
  dependencies {
    ...
    // Add the dependency for the Google services Gradle plugin
    classpath 'com.google.gms:google-services:4.3.15'
  }
}

allprojects {
  ...
  repositories {
    // Make sure that you have the following two repositories
    google()  // Google's Maven repository
    mavenCentral()  // Maven Central repository
  }
}
```

x. Then, in your module (app-level) build.gradle file, add both the google-services plugin and any Firebase SDKs that you want to use in your app:

```
plugins {
  id 'com.android.application'
  // Add the Google services Gradle plugin
  id 'com.google.gms.google-services'
  ...
}

dependencies {
  // Import the Firebase BoM
  implementation platform('com.google.firebase:firebase-bom:31.5.0')

  // TODO: Add the dependencies for Firebase products you want to use
  // When using the BoM, don't specify versions in Firebase dependencies
  implementation 'com.google.firebase:firebase-analytics-ktx'

  // Add the dependencies for any other desired Firebase products
  // https://firebase.google.com/docs/android/setup#available-libraries
}
```

### h. Setup Firebase iOS App

i. Go to the Firebase console

ii. Select the project you created in step 5.d.vi

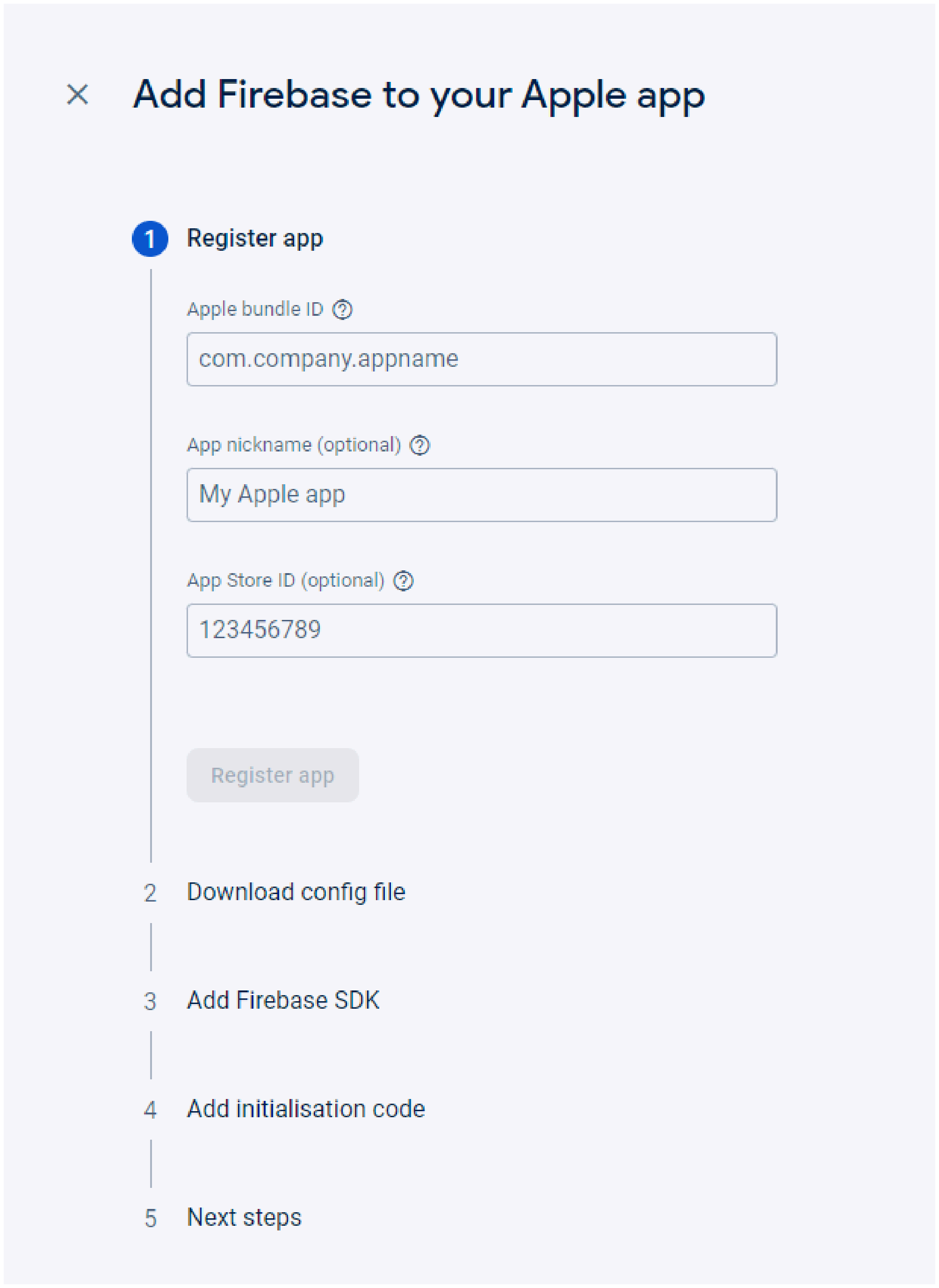
iii. Go to **Project Setting**

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

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

viii. 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]()

7. 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. To view your app's settings, select the Runner target in the Xcode navigator.

8. Verify the most important settings

   a. In the Identity section of the General tab

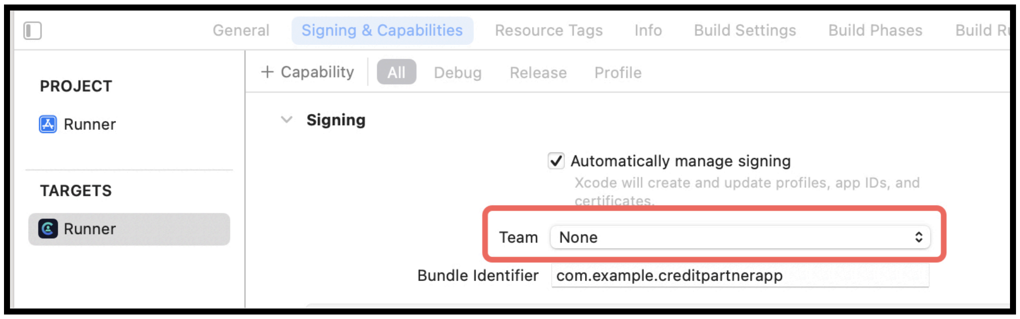
      i. **Display Name** (The display name of your app.)

      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](Code Signing Guide))

      ii. **Team** (Select the team associated with your registered Apple Developer account. If required, select Add Account…, then update this setting.)



✓ Automatically manage signing
Xcode will create and update profiles, app IDs, and certificates.

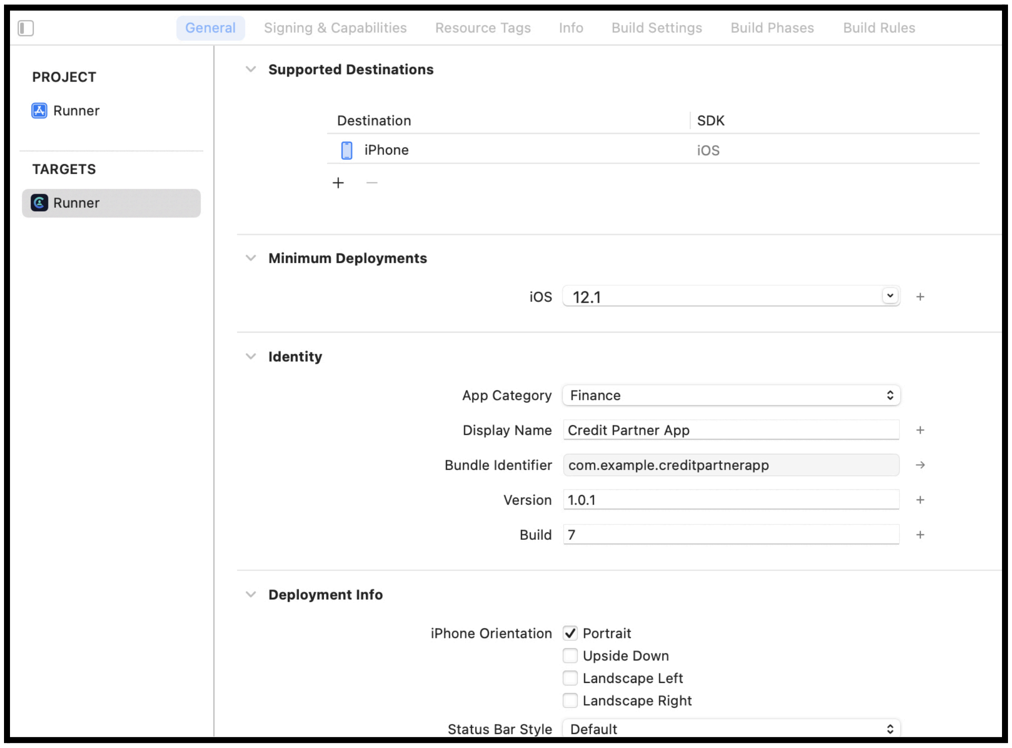
Team: None

Bundle Identifier: com.example.creditpartnerapp

   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:



Supported Destinations

| Destination | SDK |
|---|---|
| iPhone | iOS |

Minimum Deployments

iOS 12.1

Identity

App Category: Finance
Display Name: Credit Partner App
Bundle Identifier: com.example.creditpartnerapp
Version: 1.0.1
Build: 7

Deployment Info

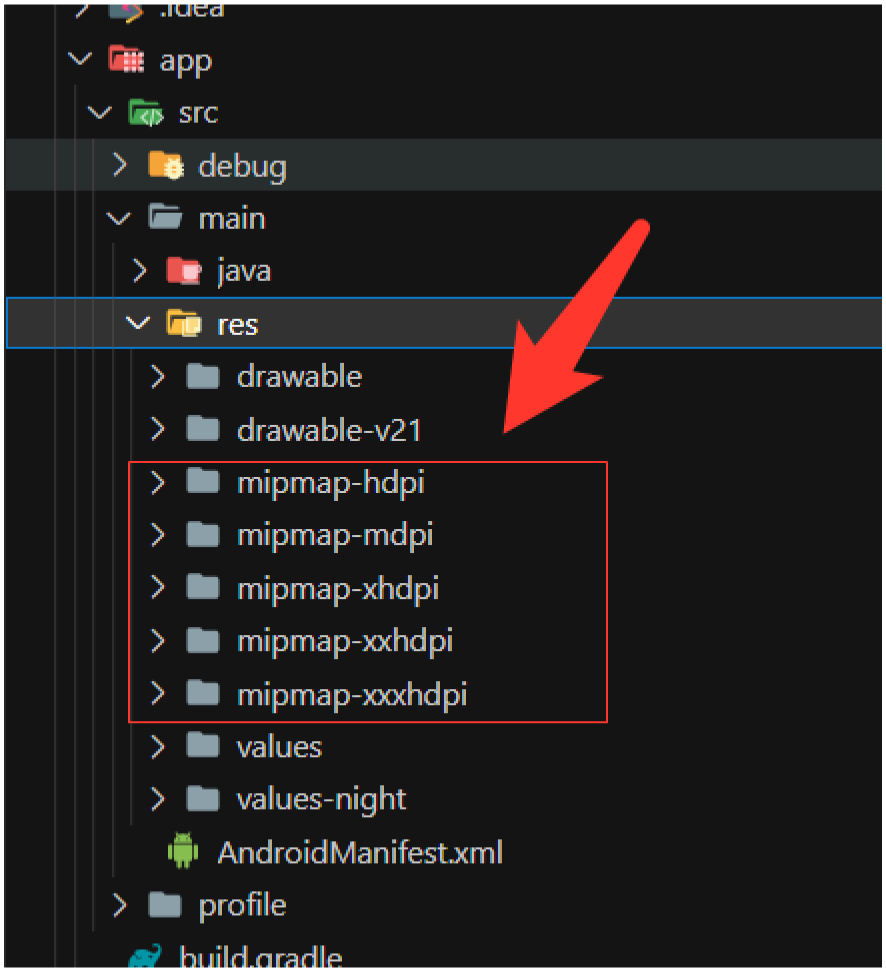
iPhone Orientation: [✓ Portrait] Upside Down, Landscape Left, Landscape Right
Status Bar Style: Default

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

i. Change App Icon

i. For Android

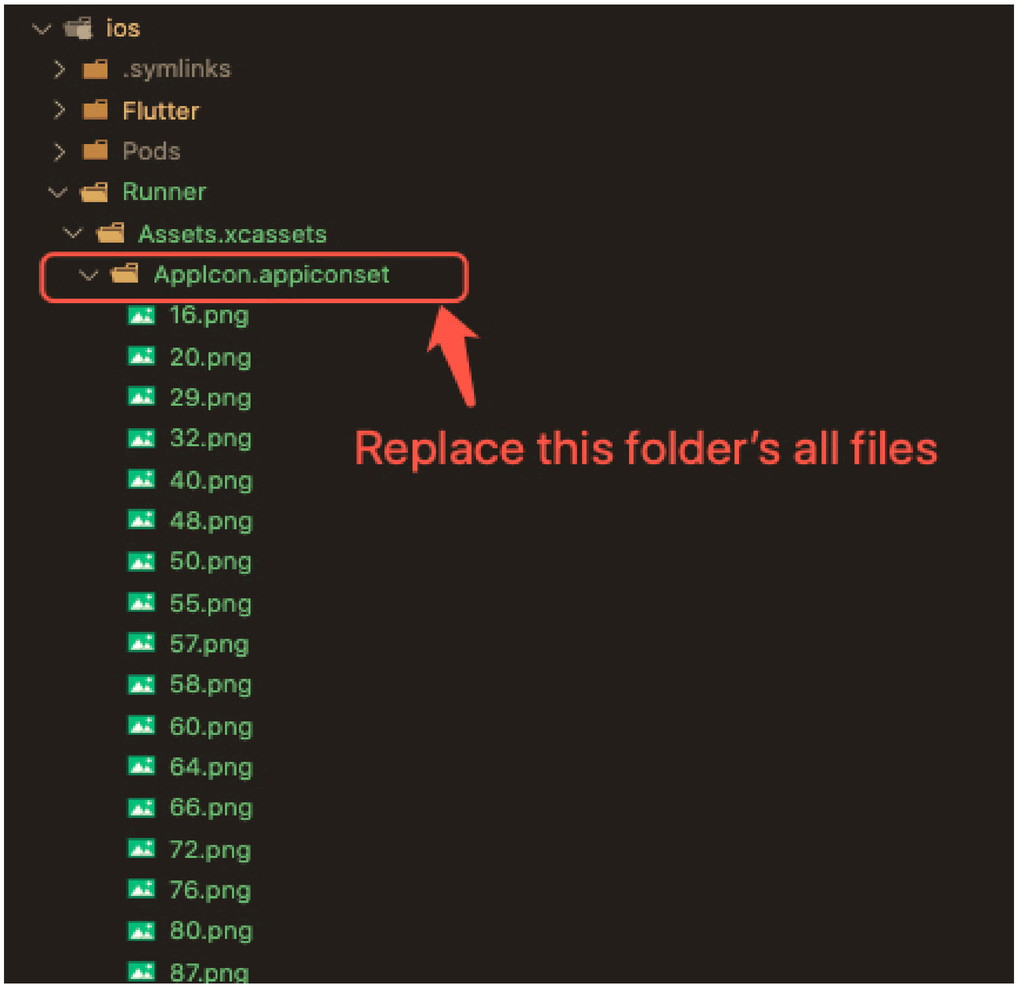
Replace the icons in the **android\app\src\main\res** folder as shown in the below image.



ii. For iOS

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

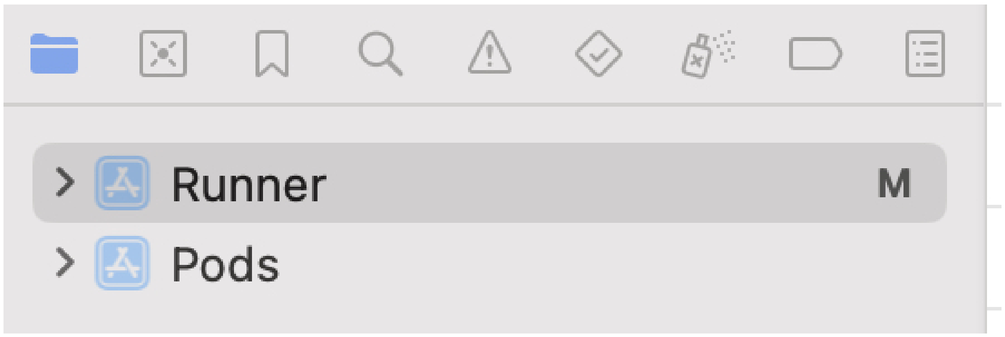
   **ios\Runner\Assets.xcassets\AppIcon.appiconset**



2. Change icons using XCode

   a. Right-click on the iOS folder Choose Open in Xcode Option

   b. Click on the folder icon on the left side of the XCode window

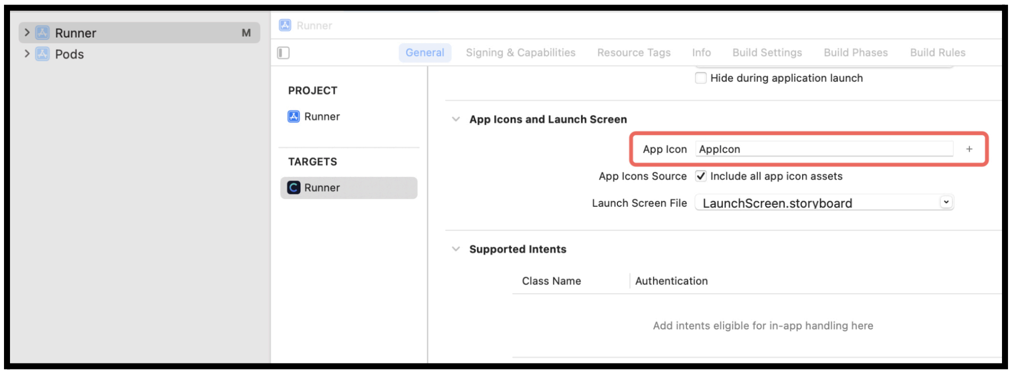


   c. Select Runner.

d. Select Target runner

e. Go to App Icons And Launch Images

f. Click the right arrow button of the app icon source



Hide during application launch

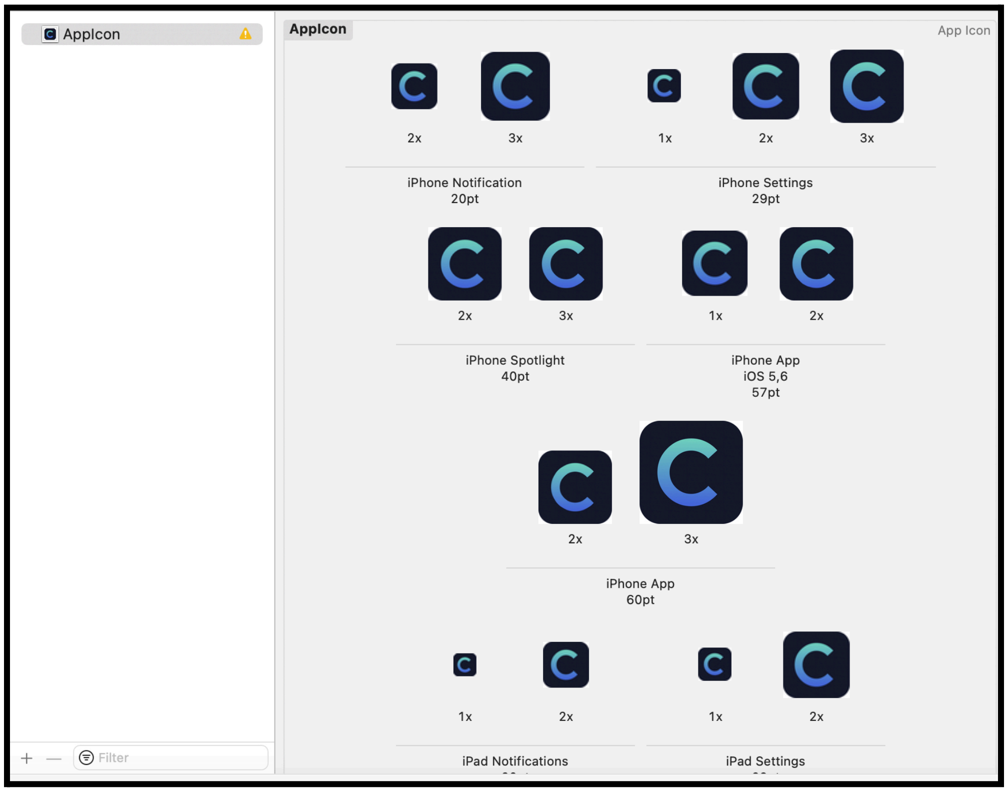
App Icons and Launch Screen

App Icon: AppIcon

App Icons Source: ✓ Include all app icon assets

Launch Screen File: LaunchScreen.storyboard

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. 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
```

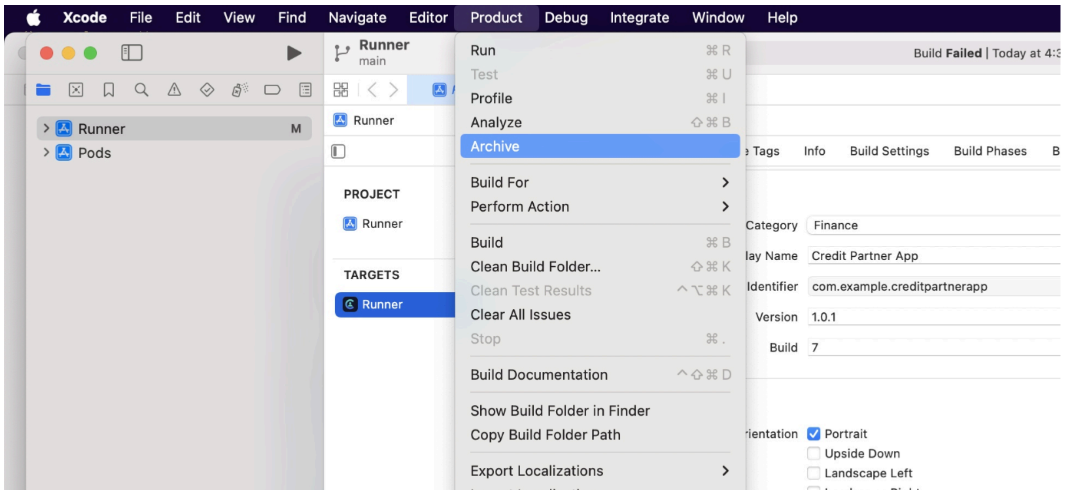
iv. Get the APK from the below path

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

## k. Build Release for iOS

i. Open Project in XCode

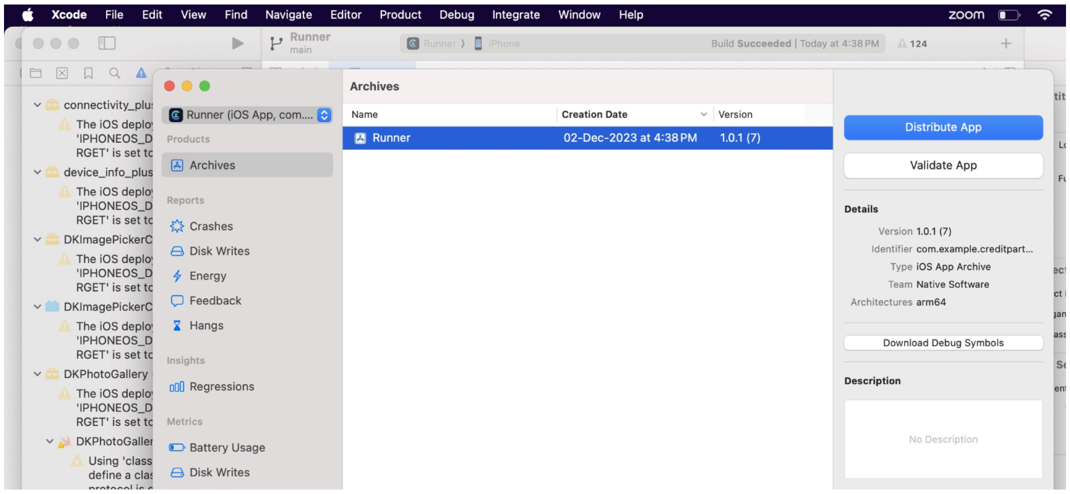
ii. Select **Archive** from the **Product Menu**



Category: Finance
Display Name: Credit Partner App
Identifier: com.example.creditpartnerapp
Version: 1.0.1
Build: 7
Orientation: ✓ Portrait, Upside Down, Landscape Left

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



| Name | Creation Date | Version |
|---|---|---|
| Runner | 02-Dec-2023 at 4:38 PM | 1.0.1 (7) |

Details
Version 1.0.1 (7)
Identifier com.example.creditpart...
Type iOS App Archive
Team Native Software
Architectures arm64

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]

## I. Other Options for the Advanced User

i. Paths to the images used in the app

| Images | Path | Screen Path |
|---|---|---|
| Splash screen | assets/web-logo.png | lib/views/splash/splash_screen.dart |
| Introduction screen | assets/intro_image1.png | lib/views/introduction/introduction_screen.dart |
| | assets/intro_image2.png | lib/views/introduction/introduction_screen.dart |
| | assets/intro_image3.png | lib/views/introduction/introduction_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, you can make the changes in the **pubspec.yaml** file and the **Assets** folder.

| | |
|---|---|
| Roboto | assets/fonts/Roboto-Black.ttf |
| | assets/fonts/Roboto-Bold.ttf |

| | |
|---|---|
| | assets/fonts/Roboto-Regular.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 | #32D6D8 |
| Primary swatch color | #32D6D8 |
| Text button - background color | #32D6D8 |
| Card - Color | white |
| Card - shadow color | grey[200] |
| Appbar theme - color | white |
| Checkbox - check color | white |
| Checkbox - fill color | #32D6D8 |
| 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 - ^2.0.5 | For the OTP text field |
| font_awesome_flutter - ^10.4.0 | To access the icon of font-awesome |
| Material_design_icons_flutter - 6.0.7096 | The Material Design Icons designed by the community for Flutter |
| dotted_border - 2.0.0+3 | A flutter package to easily add dotted borders around widgets |
| cupertino_icons - ^1.0.5 | For UI design |
| scratcher - ^2.3.0 | For Scratch card |
| convex_bottom_bar - ^3.2.0 | For BottomAppBar |
| share_plus - ^7.0.0 | A Flutter plugin to share content from your Flutter app via the platform's share dialog. |
| dots_indicator - 2.1.2 | To display dots indicator to show a position |

| | |
|---|---|
| flutter_document_picker - 5.2.1 | The picked document is copied |
| file_picker - ^5.3.0 | Pick single or multiple files |
| sms_autofill - ^2.3.0 | The SMS autofill is provided by default |
| mobile_number - 2.1.1 | For fetching the device's mobile number or list SIM card data |
| otp_autofill - 2.1.0 | For OTP autofill using User Consent API and Retriever API |
| open_file - ^3.3.1 | to open files |
| image_cropper - ^4.0.1 | Flexible image cropping |
| flutter_image_compress  - ^2.0.3 | Compresses image as native plugin. |
| 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 - ^2.0.5 | For drawing svg files. |
| connectivity_plus - ^4.0.0 | To check the connectivity of the device |
| http -  ^0.13.6 | For consuming HTTP resources |
| shared_preferences -  ^2.1.1 | To store something locally |
| shimmer - 2.0.0 | Easy way to add shimmer effect |
| carousel_slider - ^4.2.1 | A carousel slider widget |
| number_to_words - 1.0.0 | For convert number to words by |
| device_info_plus -  ^9.0.0 | To get device info |
| step_progress_indicator - ^1.0.2 | Bar indicator made of a series of selected and unselected steps |
| permission_handler -  ^10.2.0 | This plugin provides a cross-platform API to request and check permission |
| date_format - ^2.0.7 | A simple API to format dates |
| Syncfusion_flutter_pdfviewer - 20.3.61-beta | Flutter PDF Viewer library is used to display a PDF document seamlessly and efficiently. |
| page_view_indicators - 2.0.0 | Customizable indicators for your PageViews. |
| image_picker - ^0.8.7+5 | For picking an image |
| cached_network_image -  ^3.2.3 | To show the image from internet |
| flutter_custom_clippers - ^2.1.0 | custom clippers to help you achieve various custom shapes. |

| | |
|---|---|
| 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. |
| firebase_analytics - ^10.4.0 | A Flutter plugin to use the Google Analytics for Firebase API. |
| Firebase_dynamic_links - ^5.3.1 | For Google Dynamic Links for Firebase |
| firebase_messaging - 14.6.0 | To use Firebase Cloud Messaging API. |
| flutter_local_notifications - ^14.0.0+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. |
| flutter_html - ^3.0.0-beta.1 | For rendering HTML and CSS as Flutter widgets. |
| fl_chart - 0.62.0 | For Line Chart, Bar Chart, Pie Chart, Scatter Chart, and Radar Chart. |
| Introduction_screen - ^3.1.8 | Introduction Screen allows you to have a screen on an app's first launch to, for example, explain your app |
| provider - ^6.0.5 | A wrapper around InheritedWidget to make them easier to use and more reusable |
| path_provider - ^2.0.15 | It finds commonly used locations on the file system. |
| webview_flutter - ^4.2.0 | It provides a webview widget. |
| store_redirect - ^2.0.2 | To redirect users to an app page in the Google Play Store and Apple App Store. |
| substring_highlight - ^1.0.33 | Highlight Flutter text at the character level for simple and customizable search term highlighting. |
| file - ^6.1.4 | It is a generic file system for abstraction for Dart. |
| confetti - ^0.7.0 | For celebration animation |
| flutter_downloader - ^1.10.3 | For easy to download files. |

**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 Augustl 2024.