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)

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

b. Install NodeJS from this link

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 API (Technology NodeJS)

a. Open the **Builds** folder and take WebAPI folder, upload it on your hosting server

b. Execute **npm i** command in server Terminal.

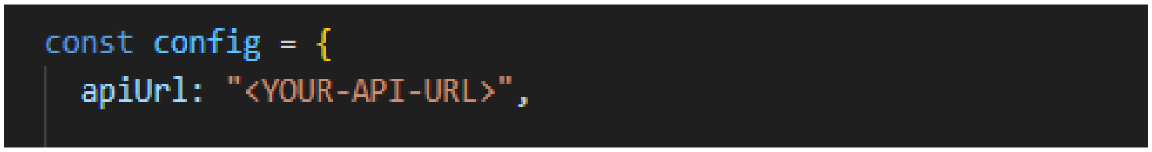
3. Setup Admin Panel (Technology ReactJS with TypeScript)

a. Open VSCode

b. Install ReactJS from this link (If not installed)

c. Open the **AdminPanel** folder into the VSCode from the ZIP file.

d. Update "build\variable.json" File in the **AdminPanel** folder with the following content.



```
const config = {
  apiUrl: "<YOUR-API-URL>",
```

- Replace <YOUR-API-URL> with your Api url

e. Upload build to your Server

f. For further configuration please check 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.

g. Setup Third-Party Libraries

i. Setup Agora in the project please follow

- For that please check step **6. a**.

ii. Setup Razorpay in the project please follow

- For that please check step **6. b**.

iii. Setup Stripe in the project please follow

- For that please check step **6. c**.

iv. Setup PhonePe in the project please follow

- For that please check step **6. d**.

v. Setup Flutterwave in the project please follow

- For that please check step **6.e**

4. Setup Mobile 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
```

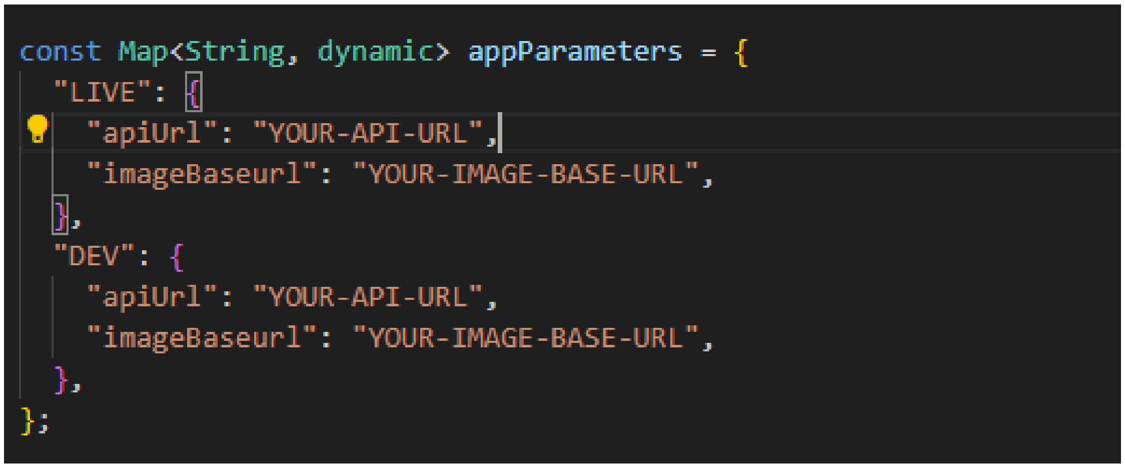
vi. To upgrade the Flutter version run the following command in the VSCode Terminal

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

```
flutter upgrade
```

b. Change API base URL

After the setup of your 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",
    "imageBaseurl": "YOUR-IMAGE-BASE-URL",
  },
  "DEV": {
    "apiUrl": "YOUR-API-URL",
    "imageBaseurl": "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/AndoidManifest.xml**
2. Change the package name in the file located at **android/app/src/debug/AndoidManifest.xml**
3. Change Package Name in file which is located at **android/app/src/Profile/AndoidManifest.xml**
4. Change the folder structure for the below path as per your package name.

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

ii. Set Bundle ID for iOS App

1. In VsCode Go to **ios/Runner/info.plist**
2. In Xcode Change in your Target Runner's General Tab

- For more information please refer to **Matrimony Setup Document v1.4.pdf** point no **5.c**.

d. Create and set Keystore file for Android

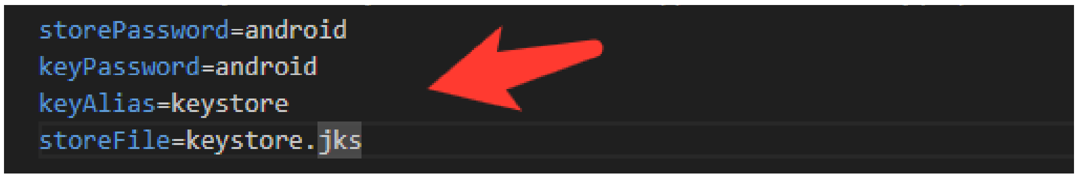
i. Create a keystore.jks file, if not exist, use the below command in the terminal

```
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 Setup

1. To create a firebase project go to this Firebase console.
2. Set up Firebase for Your Flutter App

- Go to your Firebase console dashboard.
- Click on the “Add app” button (Android/iOS) and follow the instructions to register your app.
- **For Android:**
  - Enter your Android package name(com.example.myapp) and click on “Register app”.
  - Download the **‘google-services.json’** file and place it inside your Flutter app’s **‘android/app’** directory.
  - Add the Firebase SDK to your app by following the instructions provided on the Firebase console.
- **For iOS:**
  - Enter your iOS bundle ID and click on “Register app”.
  - Download the **‘GoogleServices-Info.plist’** file and add it to your Flutter app’s **‘ios/Runner’** directory.
  - Follow the additional setup instruction provided on the Firebase console.
- If you need any help please refer to our **Matrimony Setup Document v1.4.pdf** point no **5.f, 5.g and 5.h**.

f. Change App Icon

- To change the app icon in a Flutter app for both Android and iOS platforms, you can follow these steps:s
- **For Android:**
  1. **Prepare Your App Icon:** Create your app icon in the required sizes. Android app icons typically come in various sizes (e.g., 48x48, 72x72, 96x96, 144x144, and 192x192 pixels) and densities (mdpi, hdpi, xhdpi, xxhdpi, xxxhdpi).
  2. **Replace the Default Icon:** In your Flutter project, navigate to the android/app/src/main/res directory.
  3. **Replace Icons:** Replace the default icons in the respective density folders with your new app icons. Make sure to maintain the correct folder structure and file naming conventions. For example, ic_launcher.png for the launcher icon.
  4. **Update Android Manifest:** Open android/app/src/main/AndroidManifest.xml and ensure that the android:icon attribute of the <application> tag points to the correct icon resource. It should be something like @mipmap/ic_launcher.
- **For iOS:**

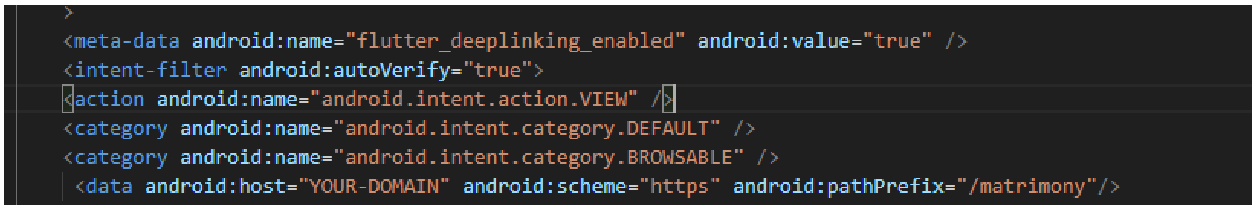
1. **Prepare Your App Icon:**Create your app icon in the required sizes. iOS app icons typically come in various sizes (e.g., 60x60, 120x120, 180x180, and so on) for different device resolutions.
2. **Replace the Default Icon:** In your Flutter project, navigate to the ios/Runner/Assets.xcassets directory.
3. **Replace Icons:** Replace the default icons in the AppIcon.appiconset folder with your new app icons. Make sure to maintain the correct size and file format (usually PNG).
4. **Update Info.plist:** Open ios/Runner/Info.plist and ensure that the CFBundleIconFile property refers to your app icon filename without the extension. For example, AppIcon.
5. **Build and Run:** Build and run your Flutter app on an iOS device or simulator to see the updated app icon

- If you need any help with any of the above points please refer to our **Matrimony Setup Document v1.4.pdf** point no **5.i**.

g. Setup Deep Link

i. For Android

- Open **android/app/src/main/AndroidManifest.xml** file.
- 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"/>
```

- Change android:launchMode to android:launchMode="singleTask"
- 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.
    - Share your link as follows:
      (Link should be look a like as follow)
      https://<YOUR MAIN DOMAIN>/PassYourNeededParameter

ii. For Ios

- **Hosting apple-app-site-association file**
  Adjust IOS settings.
    - 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:

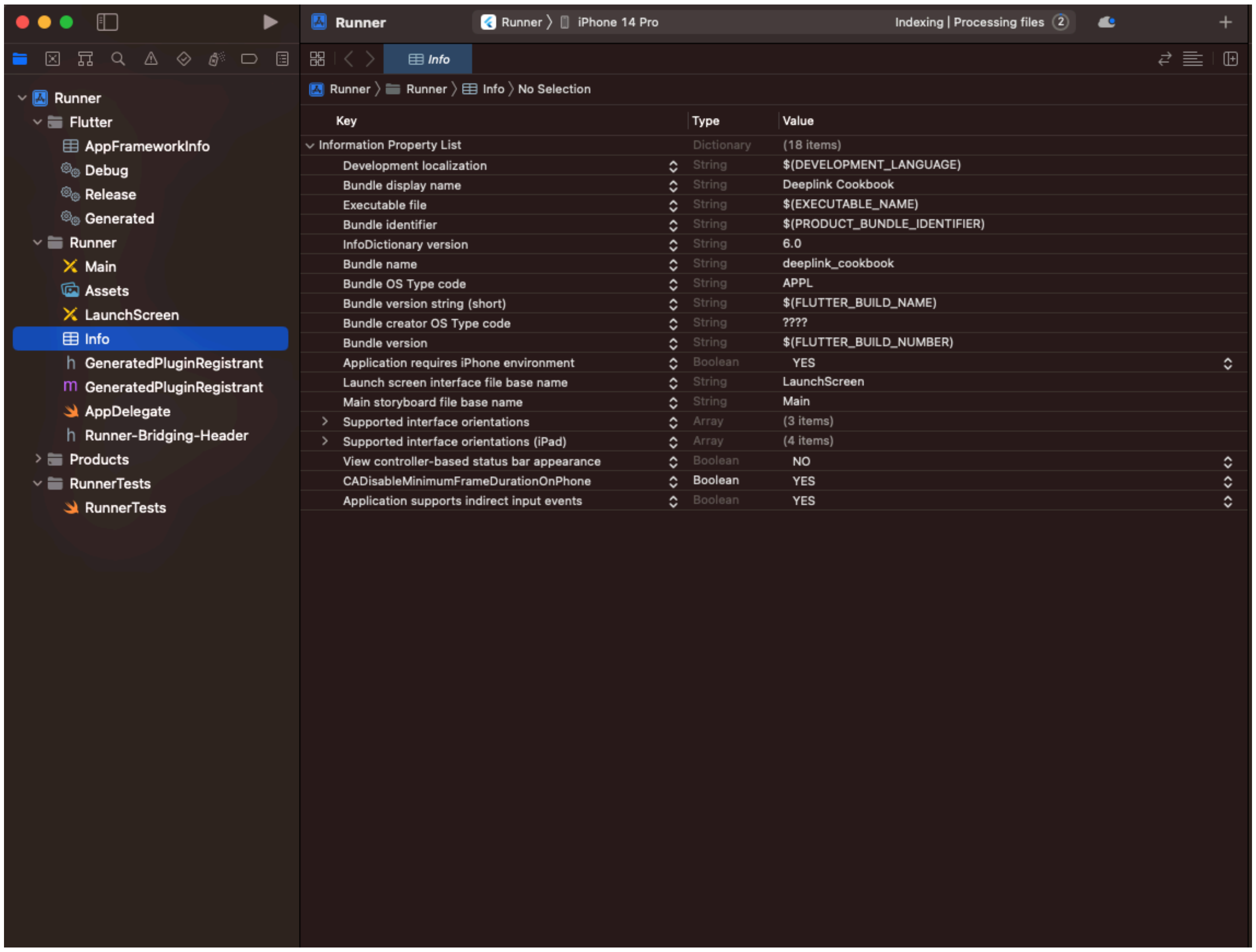
```
{
 "applinks": {
     "apps": [],
     "details": [
     {     "appID":"S8QB4VV633.com.example.deeplinkCookbook",
       "paths": ["*"]
     }
    ]
```

```
            }
           }
```

Replace “appID” with your ids.

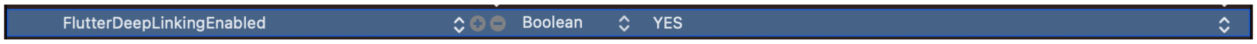
“appID” format: <team id>.<bundle id>.

- 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.
    - **Adjust IOS settings:**
        - Launch Xocode.
        - Open the **ios/Runner.xcworkspace** file inside the project’s **ios** folder.
        - Navigate to the **Info.plist** file in the **ios/Runner** folder.



Key	Type	Value
Information Property List	Dictionary	(18 items)
Development localization	String	$(DEVELOPMENT_LANGUAGE)
Bundle display name	String	Deeplink Cookbook
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	deeplink_cookbook
Bundle OS Type code	String	APPL
Bundle version string (short)	String	$(FLUTTER_BUILD_NAME)
Bundle creator OS Type code	String	????
Bundle version	String	$(FLUTTER_BUILD_NUMBER)
Application requires iPhone environment	Boolean	YES
Launch screen interface file base name	String	LaunchScreen
Main storyboard file base name	String	Main
Supported interface orientations	Array	(3 items)
Supported interface orientations (iPad)	Array	(4 items)
View controller-based status bar appearance	Boolean	NO
CADisableMinimumFrameDurationOnPhone	Boolean	YES
Application supports indirect input events	Boolean	YES

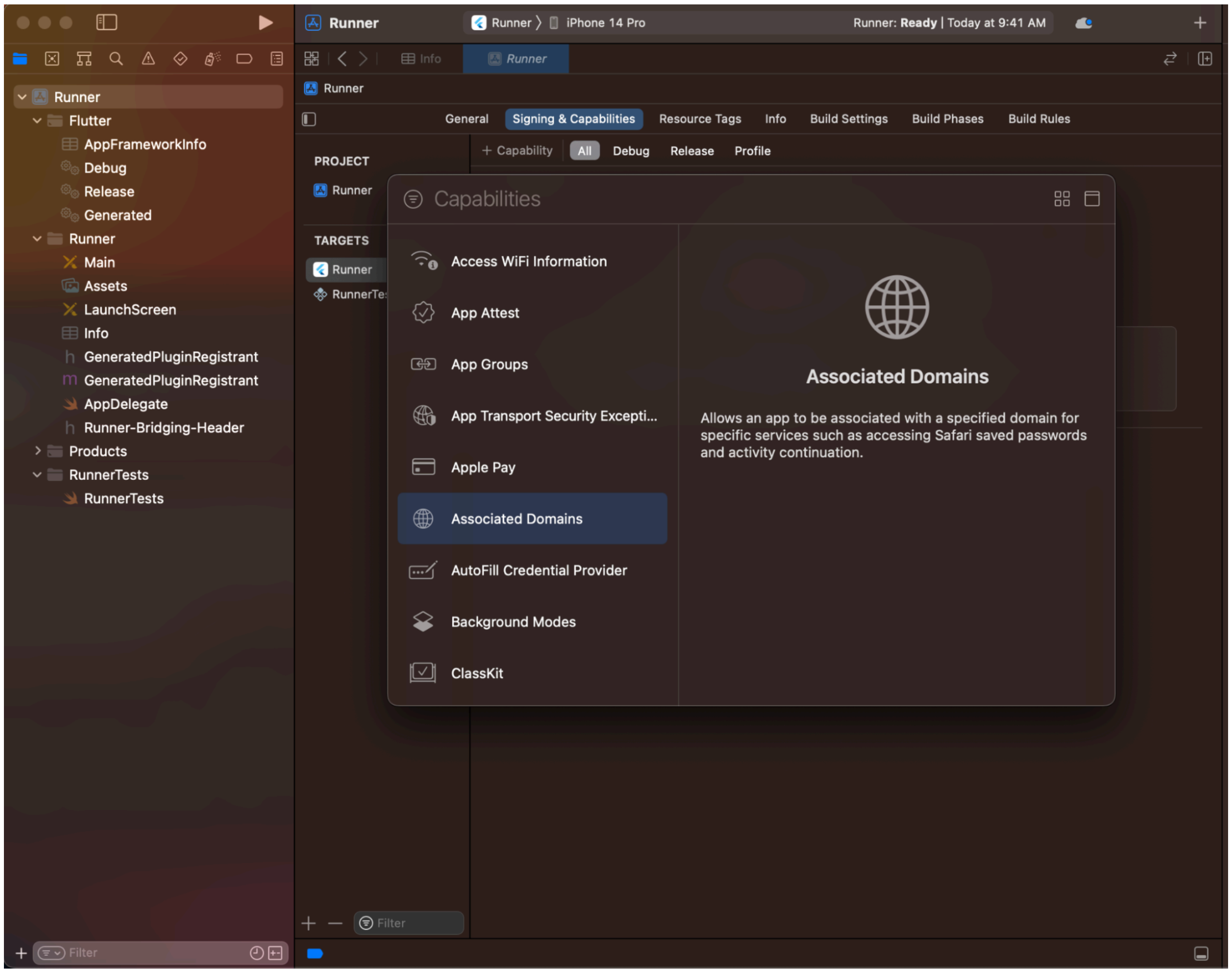
- In the **Info** property list, control-click on the list to add a row.
- Control-click the newly added row and turn on the Raw Keys and Values mode
- Update the key to **FlutterDeepLinkingEnabled** with a **Boolean** value set to



Key	Type	Value
FlutterDeepLinkingEnabled	Boolean	YES

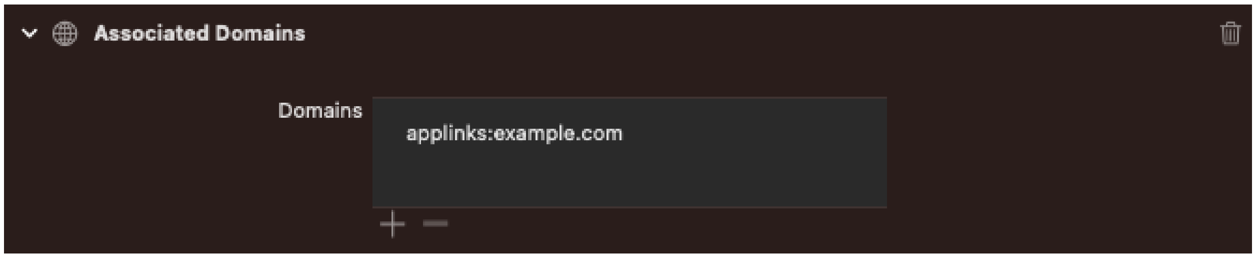
NOTE: The FlutterDeepLinkingEnabled property opts into Flutter's default deeplink handler. If you use third-party plugins, such as uni_links, setting this property will break these plugins. Skip this step if you prefer to use third-party plugins.

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



Allows an app to be associated with a specified domain for specific services such as accessing Safari saved passwords and activity continuation.

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



Domains: applinks:example.com

h. 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.

5. Third-Party Integration

a. Setup Agora

The Agora account is used to make a video call between two 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
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.

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

b. Razorpay Setup

i. Setup RazorPay from this_link.

ii. After generating **Razorpay KeyId** and **Razorpay Secret Key** from the link, Setup them 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 **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.

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.

e. FlutterWave Setup

i. Set up FlutterWave from this_link.

ii. Add FlutterWave **MerchantId**, **Publickey**, **Secretkey**, and **Encryption key** 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 **FlutterWave** Section.

NOTE: For more details regarding Third Party Integrations please follow Detailed Documentation **Matrimony Setup Document v1.4**.

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 05 April 2024.