ROOMIES DOCUMENTATION

Roomies is an app for audio chatting, users can create rooms open, private or social rooms, users can join in to public rooms to listen or participate in a topic conversation or create their own as well as create scheduled events to happen in future

Created: 7/18/2021
Latest update: 7/18/2021
By: Fredrick Mundia
Requirements 1. android studio /visual studio IDE ~ any stable version will work 2. Flutter sdk 2.7.0 or higher 3. Agora Account (free account will work) 3. nodejs (to generate agora token) 4. hosting server with nodejs hosting capability 5. Firebase

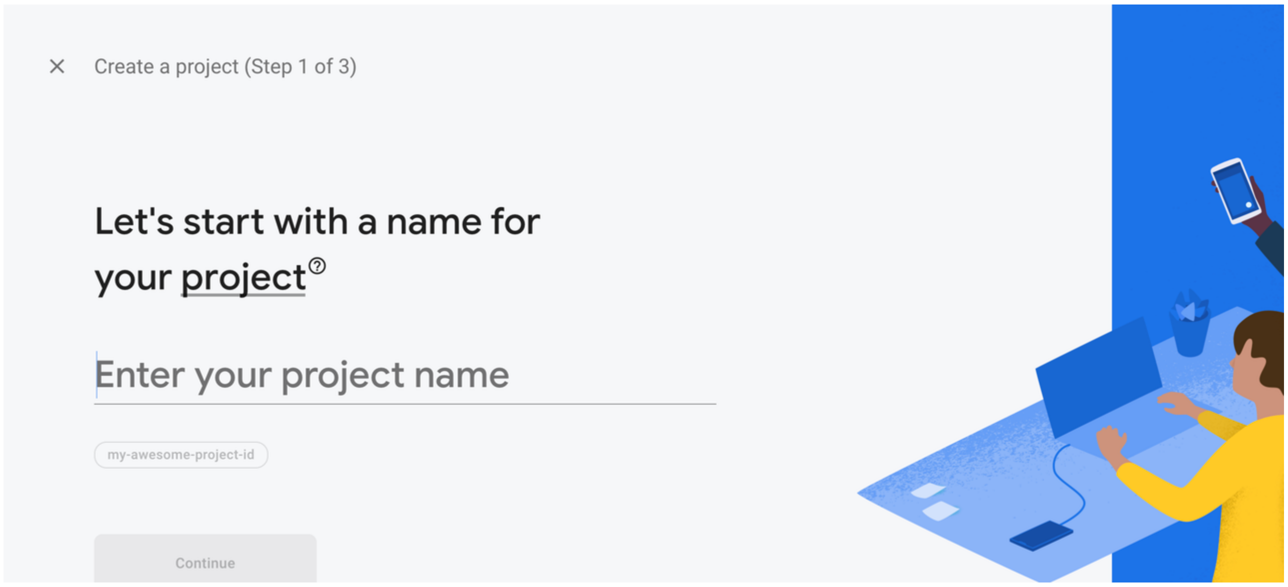
App Installation Guide

The installation is pretty easy, please follow the steps below:

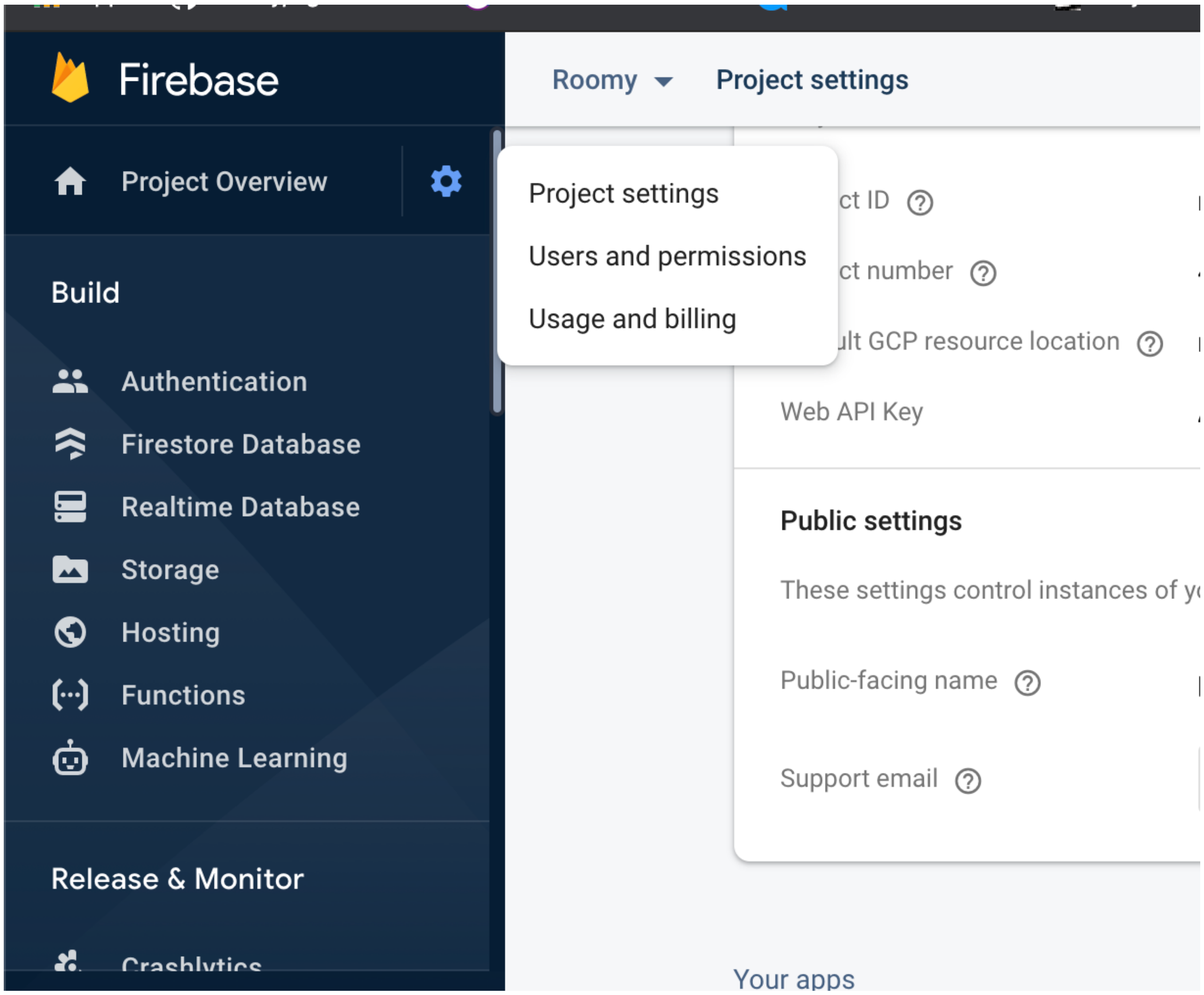
1. install android studio or Visual studio as the IDE to run this source code
2. install flutter sdk in your computer and configure with android studio or visual studio,
   note: there are different processes between windows and mac
   - windows: follow this link from flutter official website https://flutter.dev/docs/get-started/install/windows
   - mac : follow this link from flutter official website https://flutter.dev/docs/get-started/install/macos

3. import project from Roomies directory you downloaded from codecanyon

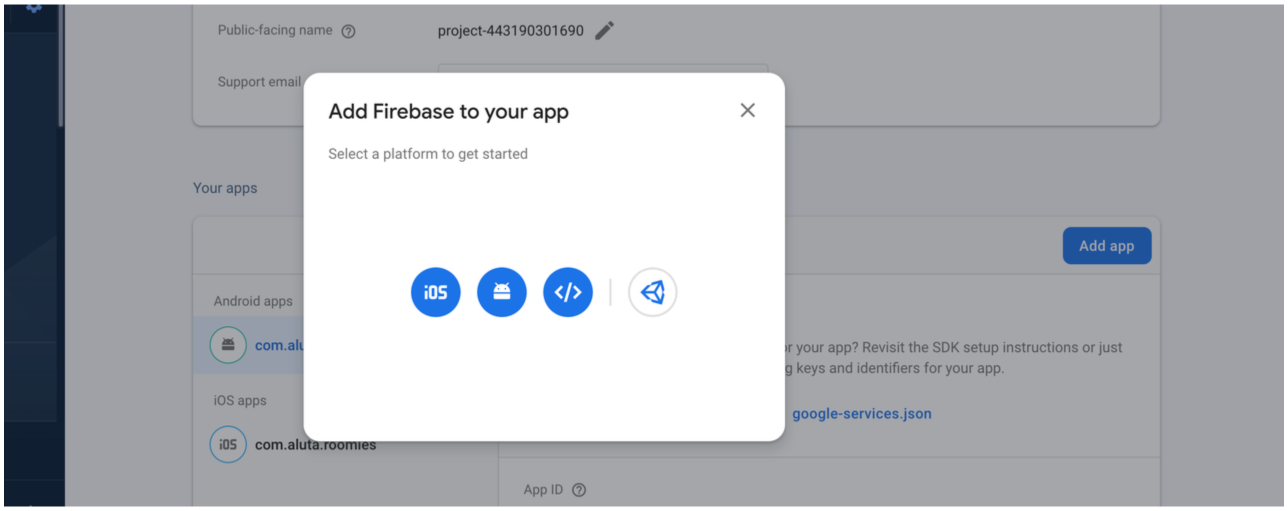
4. create firebase account and create your first project from the console https://firebase.google.com/


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

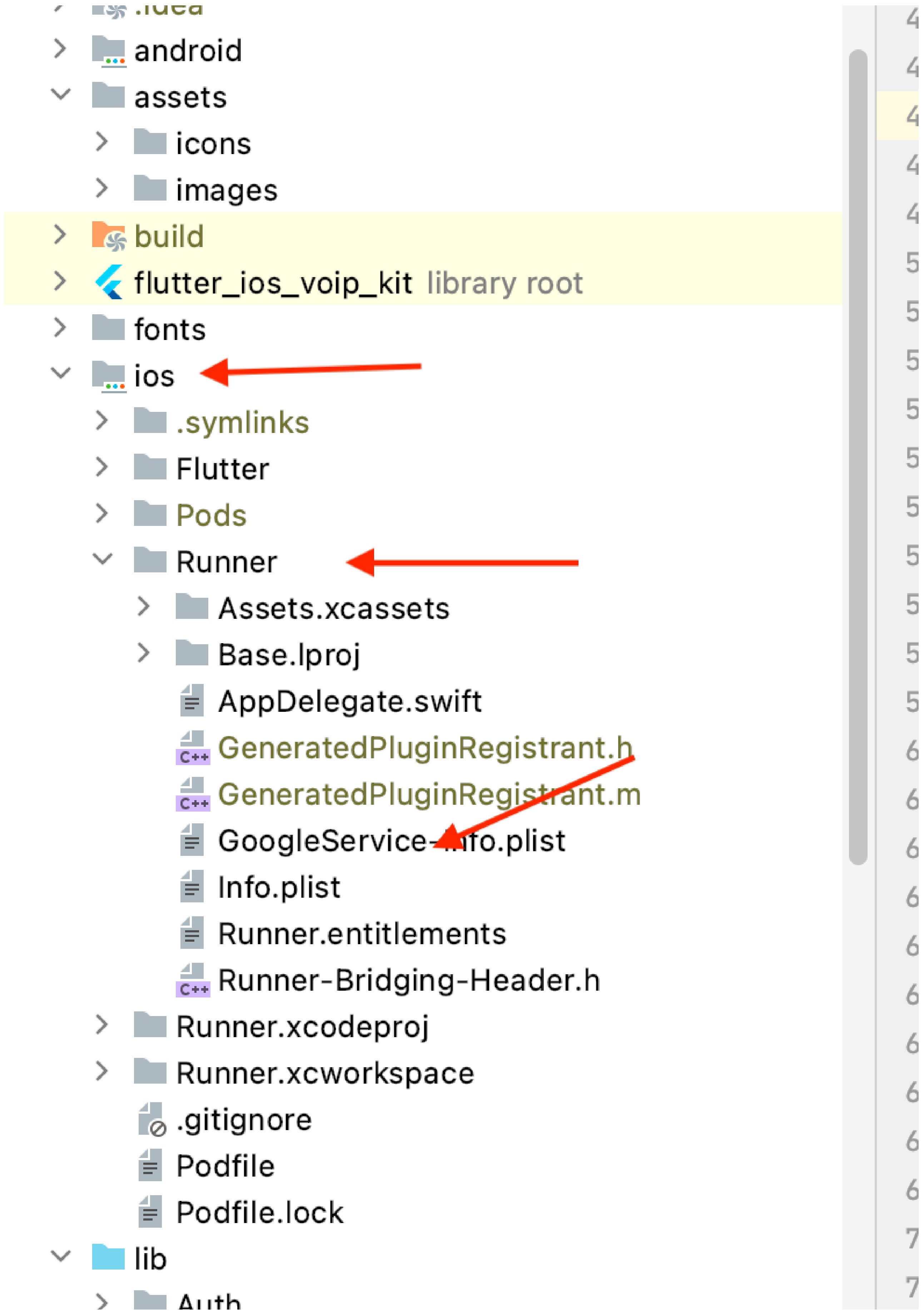
5. after creating the project, click on project settings inside your dashboard



8. scroll to the bottom of the settings page, choose android icon or ios icon depending with which platform you want to run the project on and enter app package name as “com.aluta.roomies”

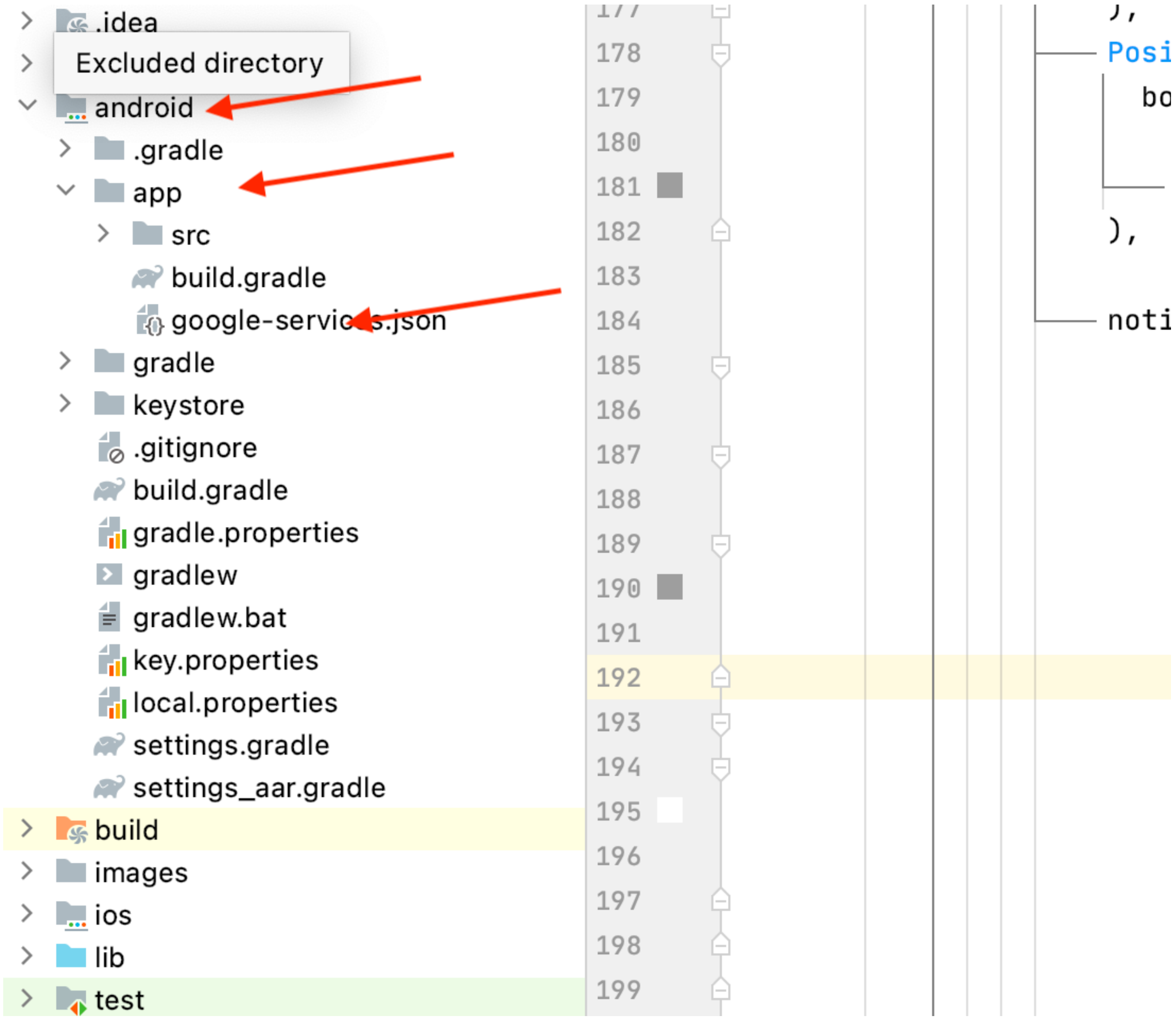


9. after you have created the app with the app bundle id “com.aluta.roomies”, download google-service.json for android or GoogleService-Info.plist for ios.
    a. for ios paste the “GoogleService-Info.plist” file to Roomies → ios → Runner



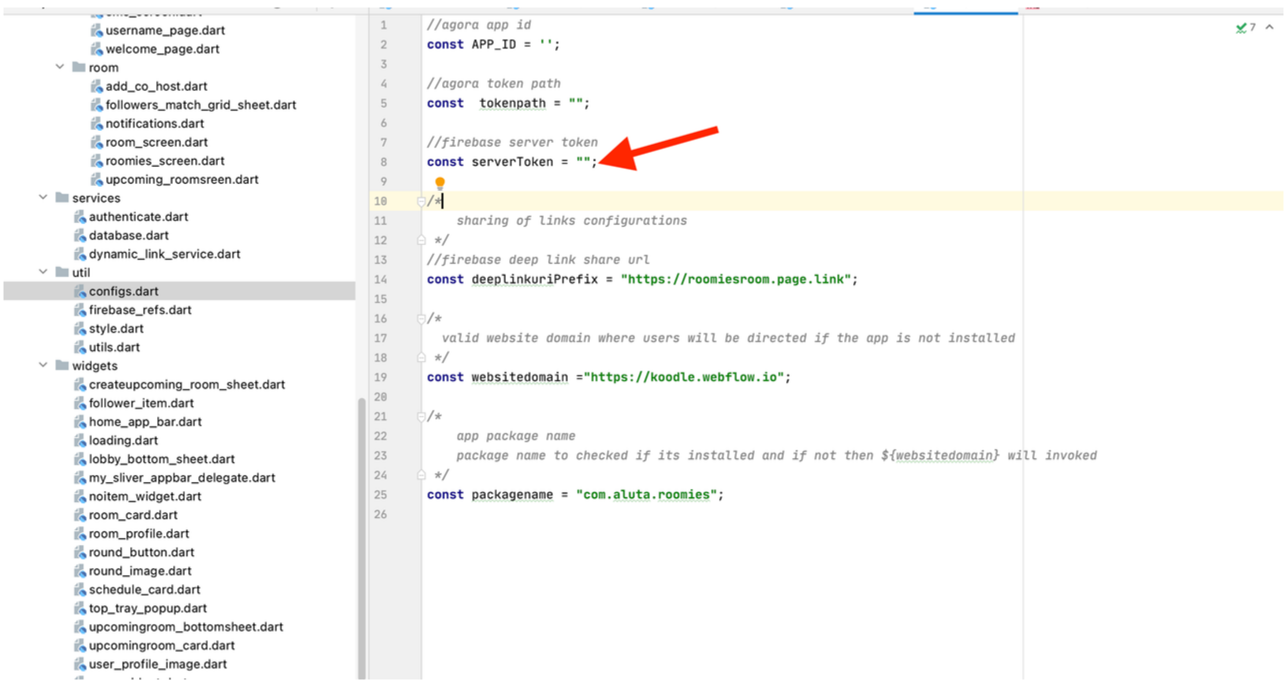
```
.idea
android
assets
  icons
  images
build
flutter_ios_voip_kit  library root
fonts
ios
  .symlinks
  Flutter
  Pods
  Runner
    Assets.xcassets
    Base.lproj
    AppDelegate.swift
    GeneratedPluginRegistrant.h
    GeneratedPluginRegistrant.m
    GoogleService-Info.plist
    Info.plist
    Runner.entitlements
    Runner-Bridging-Header.h
  Runner.xcodeproj
  Runner.xcworkspace
  .gitignore
  Podfile
  Podfile.lock
lib
  Auth
```

b. for android copy the file "google-service.json" to Roomies → android → app



.

10. last but not the list, go to your console, under **settings → cloud messaging → copy server** key and paste in the source code under **lib→utils→configs→serverToken**



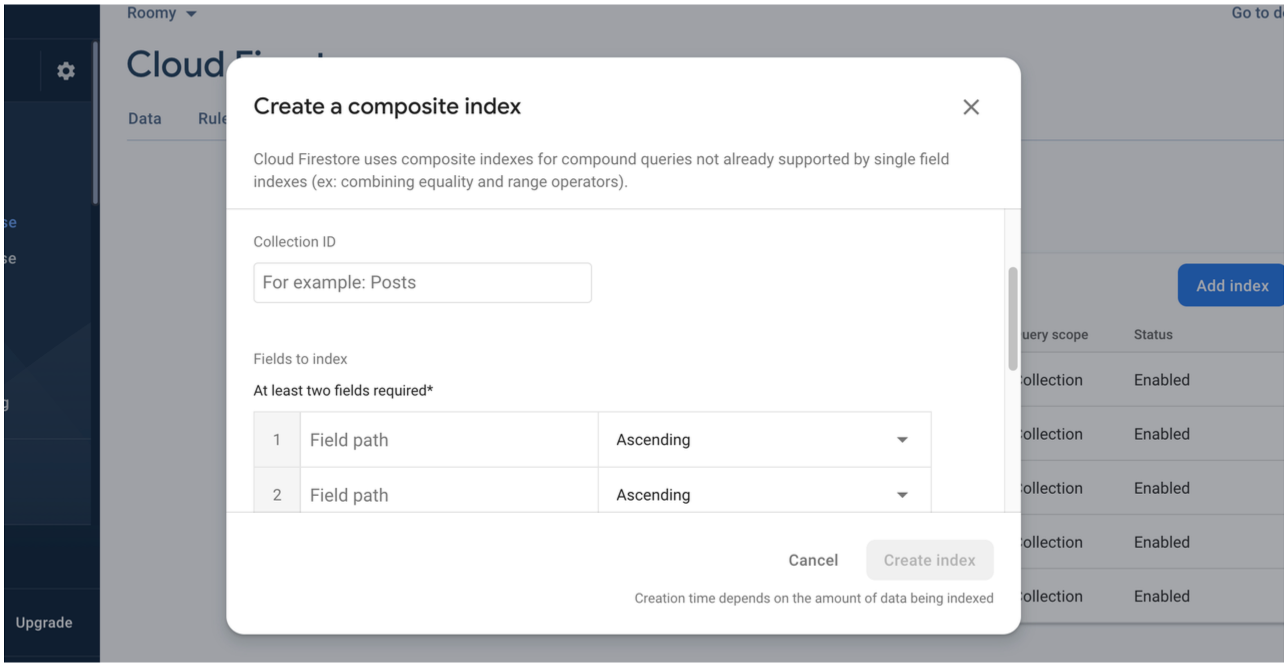
```
      username_page.dart
      welcome_page.dart
    room
      add_co_host.dart
      followers_match_grid_sheet.dart
      notifications.dart
      room_screen.dart
      roomies_screen.dart
      upcoming_roomsreen.dart
  services
    authenticate.dart
    database.dart
    dynamic_link_service.dart
  util
    configs.dart
    firebase_refs.dart
    style.dart
    utils.dart
  widgets
    createupcoming_room_sheet.dart
    follower_item.dart
    home_app_bar.dart
    loading.dart
    lobby_bottom_sheet.dart
    my_sliver_appbar_delegate.dart
    noitem_widget.dart
    room_card.dart
    room_profile.dart
    round_button.dart
    round_image.dart
    schedule_card.dart
    top_tray_popup.dart
    upcomingroom_bottomsheet.dart
    upcomingroom_card.dart
    user_profile_image.dart
```

```
//agora app id
const APP_ID = '';

//agora token path
const  tokenpath = "";

//firebase server token
const serverToken = "";

/*
    sharing of links configurations
 */
//firebase deep link share url
const deeplinkuriPrefix = "https://roomiesroom.page.link";

/*
  valid website domain where users will be directed if the app is not installed
 */
const websitedomain ="https://koodle.webflow.io";

/*
    app package name
    package name to checked if its installed and if not then ${websitedomain} will invoked
 */
const packagename = "com.aluta.roomies";
```

Firebase Indexing

for firebase queries to work properly, you have to create some query indexes below

open firestore console → firebase database → indexes → composite → add index



Create a composite index

Cloud Firestore uses composite indexes for compound queries not already supported by single field indexes (ex: combining equality and range operators).

Collection ID

For example: Posts

Fields to index

At least two fields required*

1	Field path	Ascending
2	Field path	Ascending

Cancel   Create index

Creation time depends on the amount of data being indexed

i. collection id : activities
   Fields to index
   - to → ascending
   - time → Descending
   Query Scope: collection

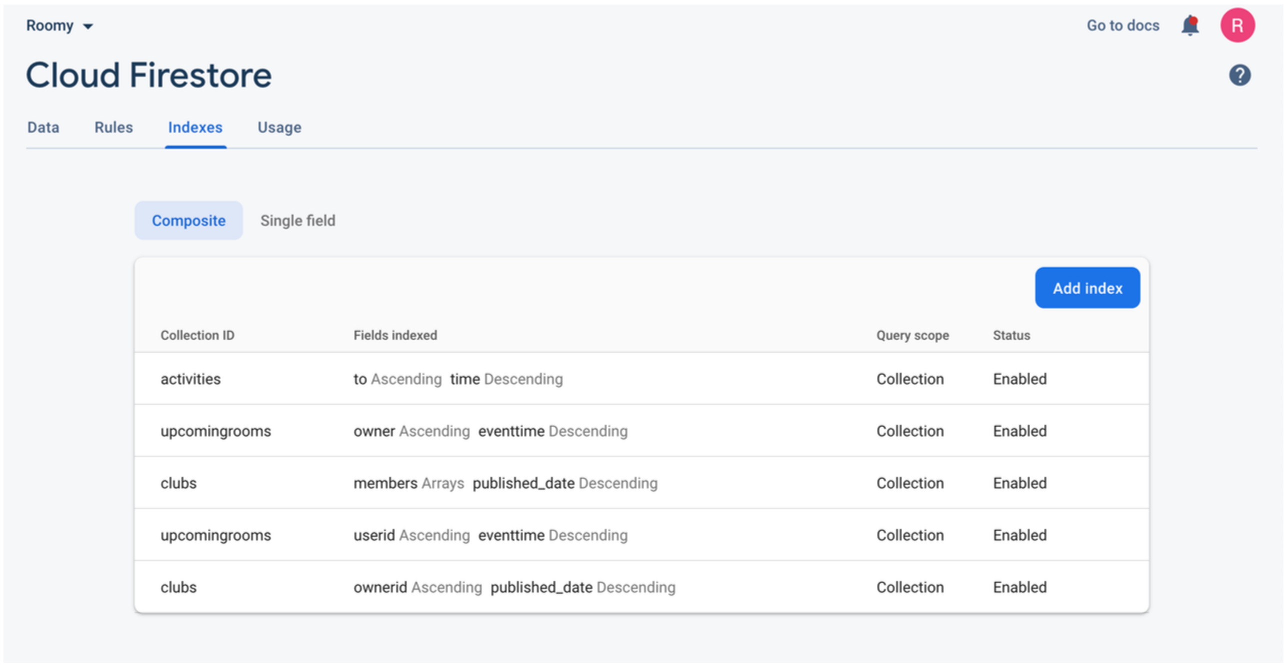
ii. collection id : upcomingrooms
   Fields to index
   - owner → ascending
   - eventtime → Descending
   Query Scope: collection

iii. collection id : clubs
   Fields to index
   - members → Arrays
   - published_date → Descending
   Query Scope: collection

iv. collection id : upcomingrooms
   Fields to index
   - userid → Ascending
   - eventtime → Descending
   Query Scope: collection

v. collection id : clubs
   Fields to index
   - ownerid → Ascending
   - published_date → Descending
   Query Scope: collection

this is how the final view of all indexes should look like

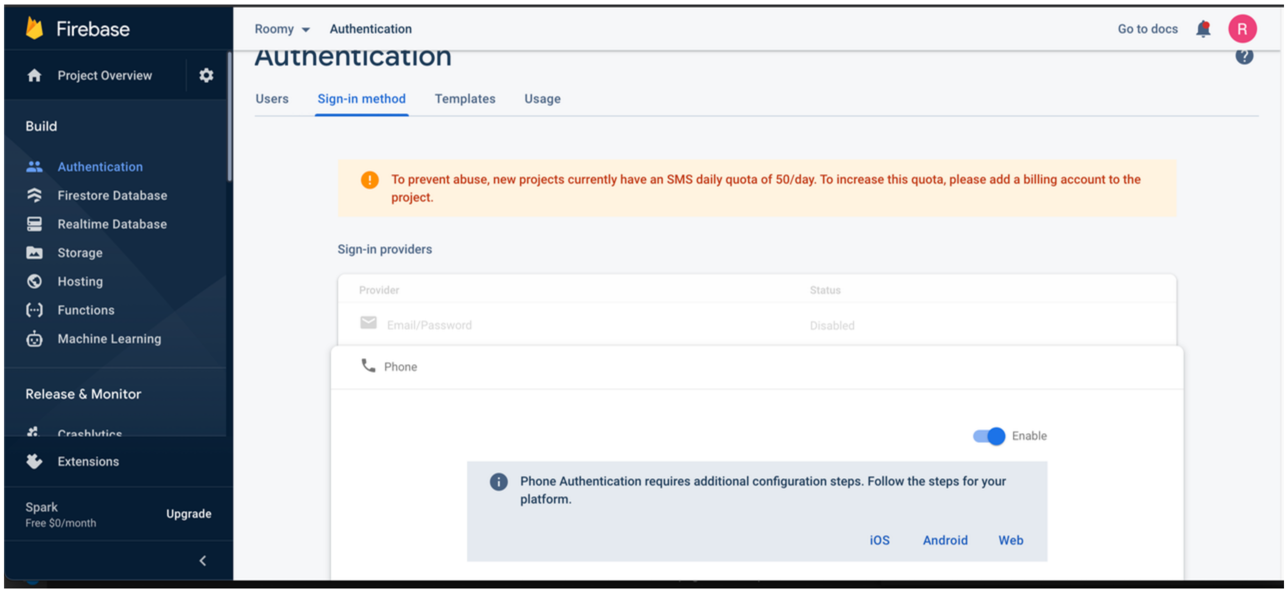


Collection ID	Fields indexed	Query scope	Status
activities	to Ascending time Descending	Collection	Enabled
upcomingrooms	owner Ascending eventtime Descending	Collection	Enabled
clubs	members Arrays published_date Descending	Collection	Enabled
upcomingrooms	userid Ascending eventtime Descending	Collection	Enabled
clubs	ownerid Ascending published_date Descending	Collection	Enabled

Firebase phone verification Configuration

firebase phone verification is a firebase option that allows you to authenticate users with their phone number preceded with the country code.

Open firebase console and go to **authentication → sign-in method → phone** and enable this option, refer to screenshot below



To prevent abuse, new projects currently have an SMS daily quota of 50/day. To increase this quota, please add a billing account to the project.

Sign-in providers

Provider	Status
Email/Password	Disabled
Phone	

Phone Authentication requires additional configuration steps. Follow the steps for your platform.

after that, your firebase phone verification is enabled, below are steps to set up in android and ios devices.

1.0 Android

Step 1.

In the Google Cloud Console, enable the Android DeviceCheck API for your project. The default Firebase API Key will be used, and needs to be allowed to access the DeviceCheck API. This is to help avoid recaptcha option in android

Step 2

generate and SHA-256 and SHA-1

go to your project, android directory and run this command in your terminal **./gradlew signingReport** if you are on a mac and for windows command prompt run this command **gradlew signingReport**

Step 3

add SHA-256 and SHA-1 generate in step 2 and open firebase console, firebase settings under your apps, select android app package, click on **add fingerprint** and add

SHA-256 and SHA-1 respectively

2.0 IOS

Step 1.

open project in xcode, under project target, select **Runner,** under **signing & capabilities,** add **background Modes** and select **Remote notifications,**

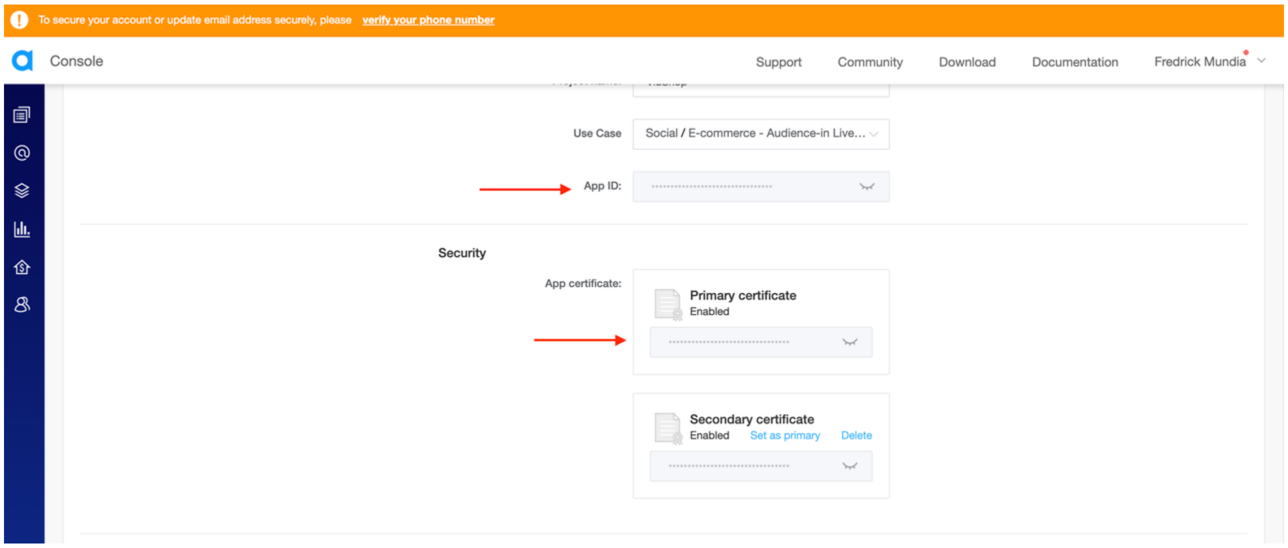
Step 2.

under **info** tab, open **URL Types**, add new section and change **identifier** field to your package name and **URL Schemes** field to **REVERSED_CLIENT_ID** which you can find it in your GoogleService-Info.plist you downloaded from firebase refer to (App **Installation Guide point 9a)**

Configuring Agora

Agora Real-Time Voice and Video Engagement platform is the platform we are using for users to hold rooms for conversations inside “Roomies” app, its a free to create an account and for the first 10,000 minutes of call. for more information you can read about them in this link https://www.agora.io/en/

1. create an account with agora here https://sso.agora.io/en/signup
2. create your application and give it any name of your choice,
3. copy appID, appCertificate and keep them safe, we will need them later in this guide



To secure your account or update email address securely, please verify your phone number

Use Case: Social / E-commerce - Audience-in Live...

App ID:

Security

App certificate:

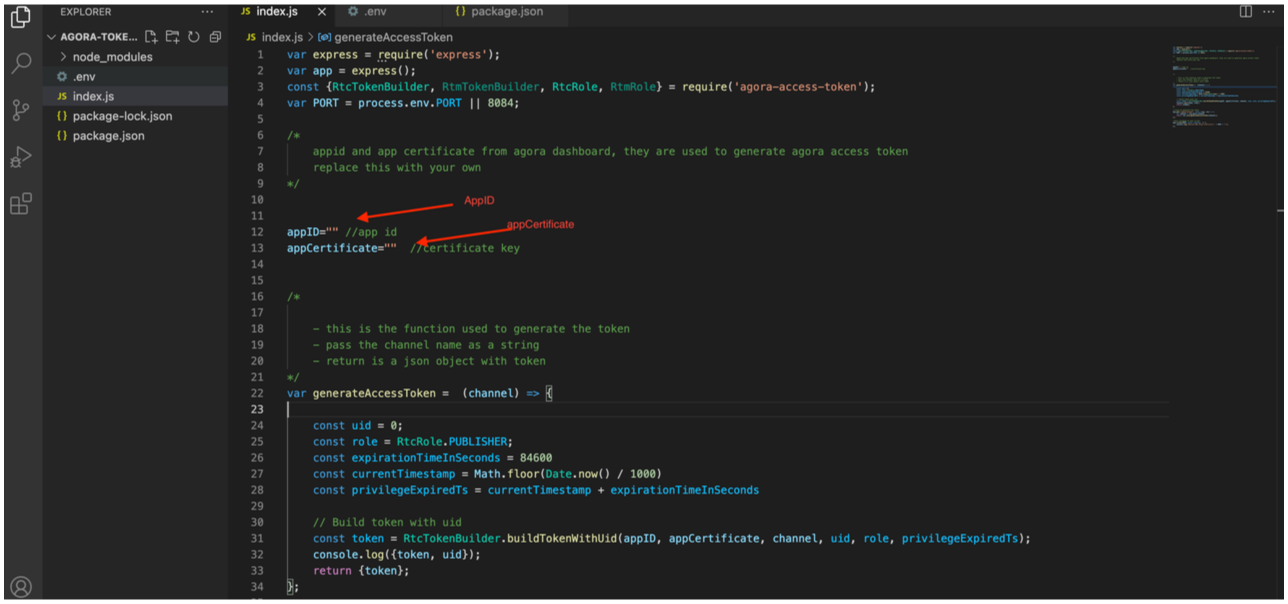
Primary certificate
Enabled

Secondary certificate
Enabled Set as primary Delete

Agora-token-generation

for agora to work in your app, you need to host "**agora-token-generation"** provided **Roomies** directory , this is a nodejs script used to generate a token that will enable users to create rooms, the script can be hosted on your server of choice provided it can support nodejs 10 and above

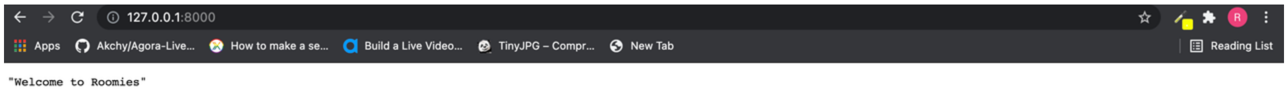
1. install nodejs v10 and above on your server of your choice
2. Open directory **agora-token-generation (**this folder has the codes to generate agora token to allow your app communicate with agora server and generate rooms)**,**
3. open the file **index.js** and replace **appID** and **appCertificate** with the once you copied from agora when you created your application (refer to Configuring Agora → point number. 3 above)



```javascript
var express = require('express');
var app = express();
const {RtcTokenBuilder, RtmTokenBuilder, RtcRole, RtmRole} = require('agora-access-token');
var PORT = process.env.PORT || 8084;

/*
    appid and app certificate from agora dashboard, they are used to generate agora access token
    replace this with your own
*/


appID="" //app id
appCertificate=""  //certificate key


/*

    - this is the function used to generate the token
    - pass the channel name as a string
    - return is a json object with token
*/
var generateAccessToken =  (channel) => {

    const uid = 0;
    const role = RtcRole.PUBLISHER;
    const expirationTimeInSeconds = 84600
    const currentTimestamp = Math.floor(Date.now() / 1000)
    const privilegeExpiredTs = currentTimestamp + expirationTimeInSeconds

    // Build token with uid
    const token = RtcTokenBuilder.buildTokenWithUid(appID, appCertificate, channel, uid, role, privilegeExpiredTs);
    console.log({token, uid});
    return {token};
};
```

4. upload **agora-token-generation** to your server and run **npm install** from the path where "**agora-token-generation"** is installed.

5. if your installation is completed successfully you will be able to access your script from your domain https://your-domain-name.com or from the path where you installed your nodejs script and see a screen like the one attached below



```
"Welcome to Roomies"
```

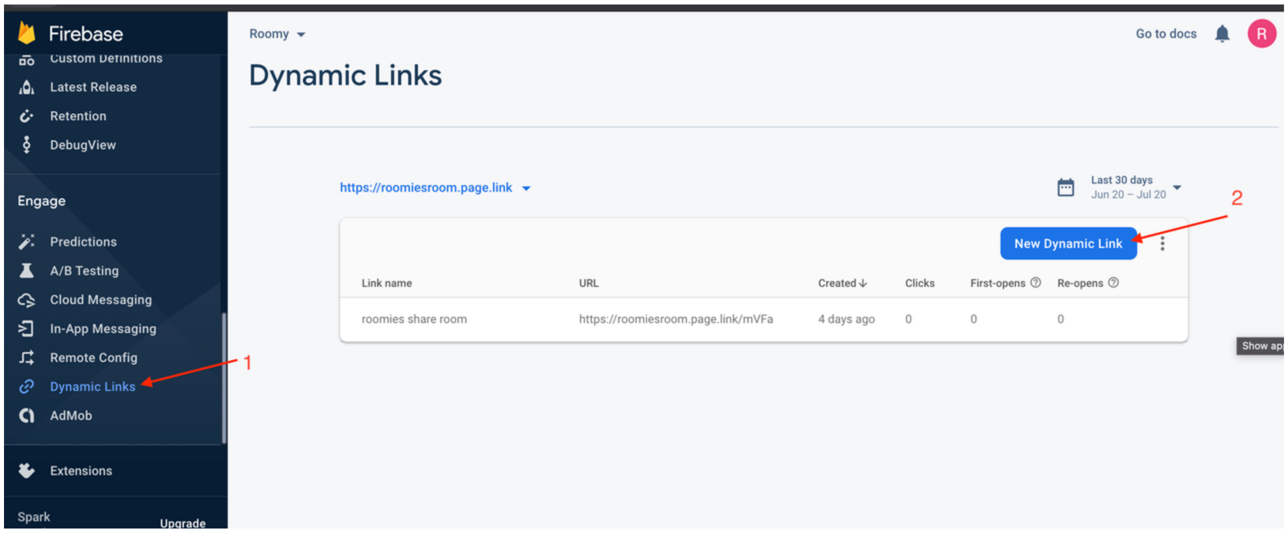
6. now go back to the project in android studio project you imported (ref from point number 4 of App **Installation Guide** section) open file lib → util → config.dart and update **tokenpath** with https://your-domain-name.com/generatetoken and

APP_ID with the one you copied from agora when you created your application (ref from point number. 3 of **Configuring Agora** section).

Configuring share link (Deep link using firebase)

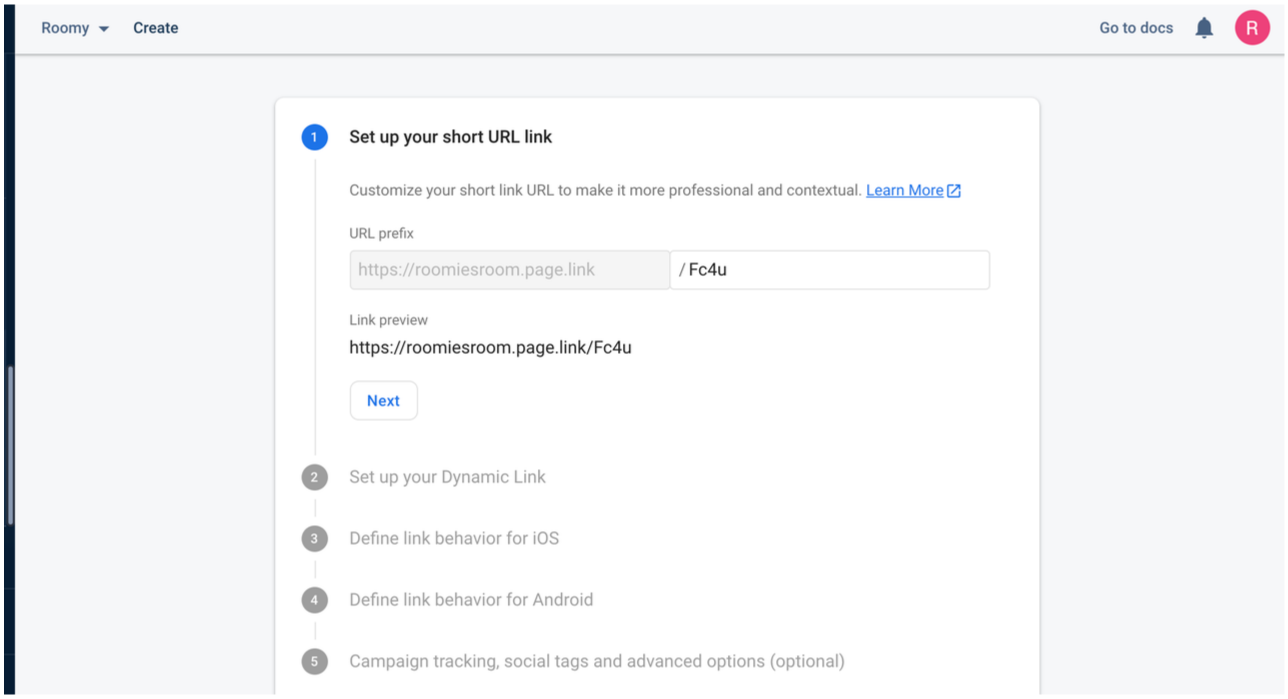
1. go to firebase console and click on the left side menu **Dynamic Links then New Dynamic**

fig.1



Link name	URL	Created	Clicks	First-opens	Re-opens
roomies share room	https://roomiesroom.page.link/mVFa	4 days ago	0	0	0

fig.2


Roomy Create Go to docs R

1 **Set up your short URL link**

Customize your short link URL to make it more professional and contextual. Learn More

URL prefix

https://roomiesroom.page.link / Fc4u

Link preview

https://roomiesroom.page.link/Fc4u

Next

2 Set up your Dynamic Link

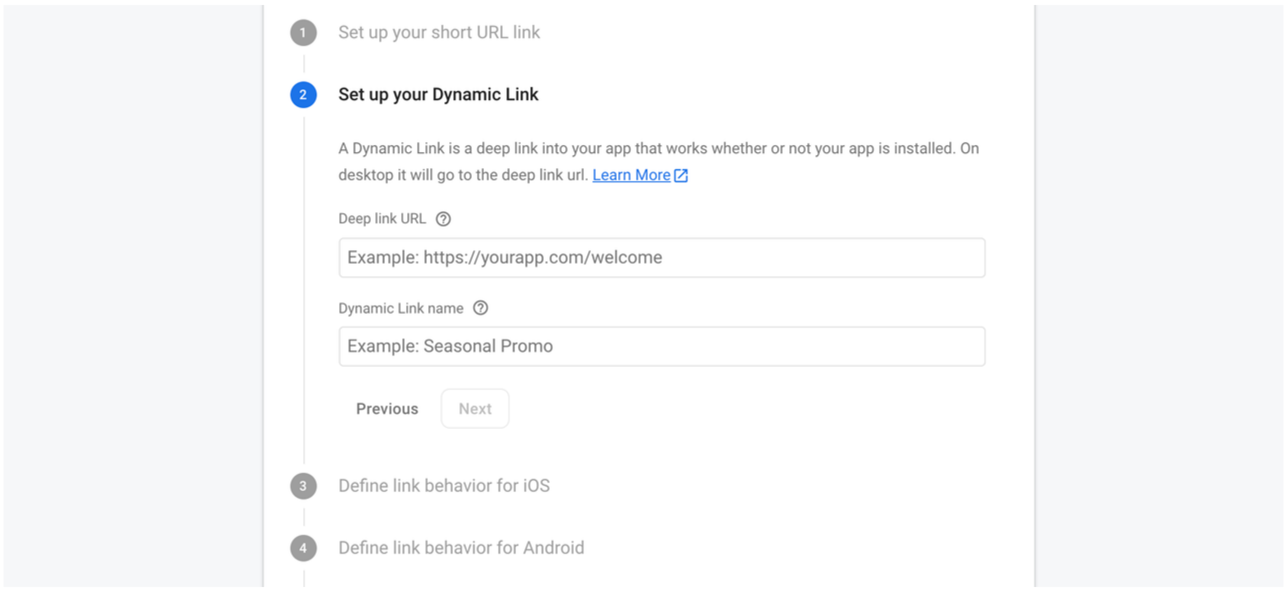
3 Define link behavior for iOS

4 Define link behavior for Android

5 Campaign tracking, social tags and advanced options (optional)

1. set up your short url link, click on Next

fig.3


1 Set up your short URL link

2 **Set up your Dynamic Link**

A Dynamic Link is a deep link into your app that works whether or not your app is installed. On desktop it will go to the deep link url. Learn More

Deep link URL

Example: https://yourapp.com/welcome

Dynamic Link name

Example: Seasonal Promo

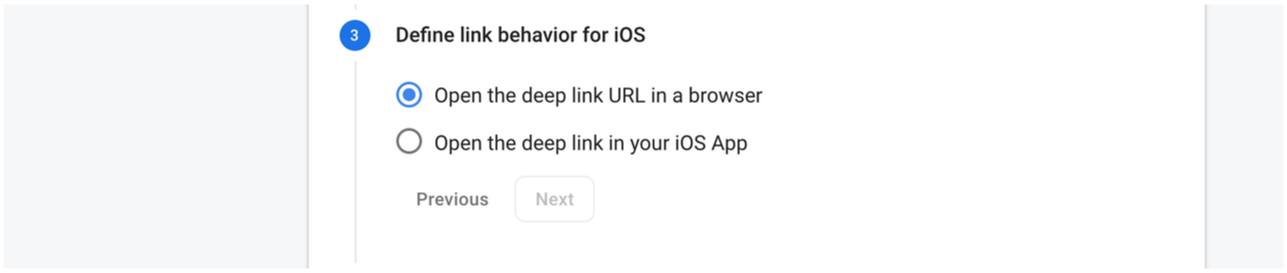
Previous Next

3 Define link behavior for iOS

4 Define link behavior for Android

Set up your dynamic link
1. deep link url - enter your valid domain name here
2. dynamic link name - enter any name here for example your app name

fig.4



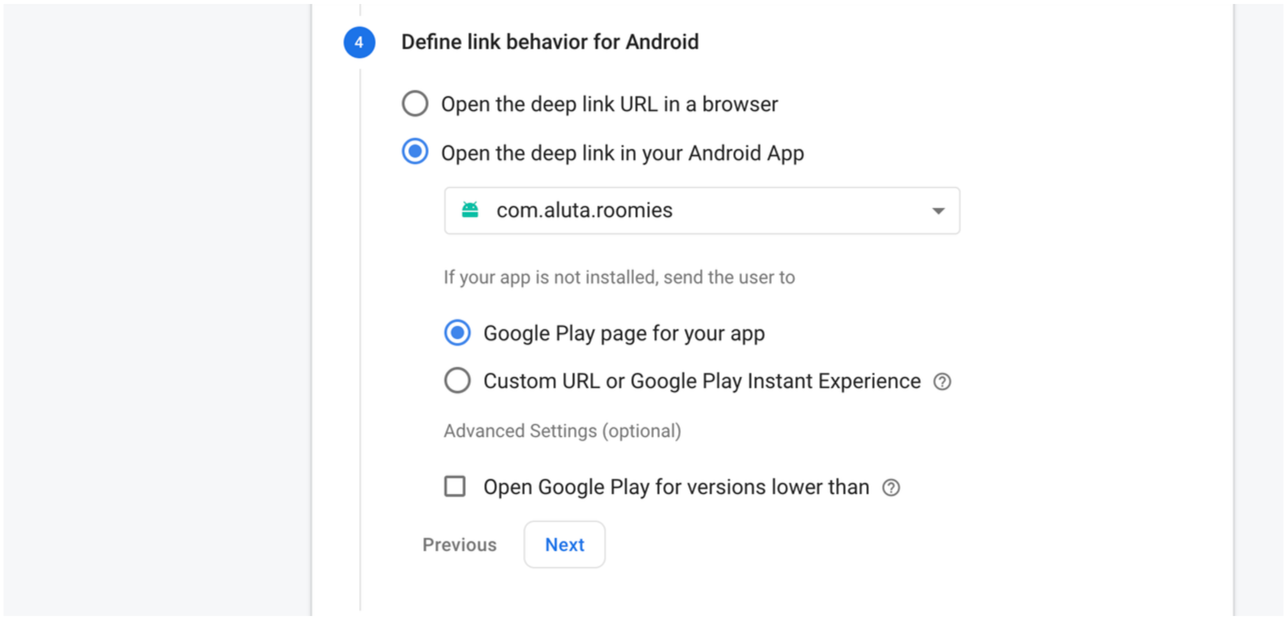
3 Define link behavior for iOS

[✓ Open the deep link URL in a browser]

Open the deep link in your iOS App

select where you want those users who dont have the app installed on ios devices will be direcred

fig.5



4 Define link behavior for Android

Open the deep link URL in a browser

[✓ Open the deep link in your Android App]

com.aluta.roomies

If your app is not installed, send the user to

[✓ Google Play page for your app]

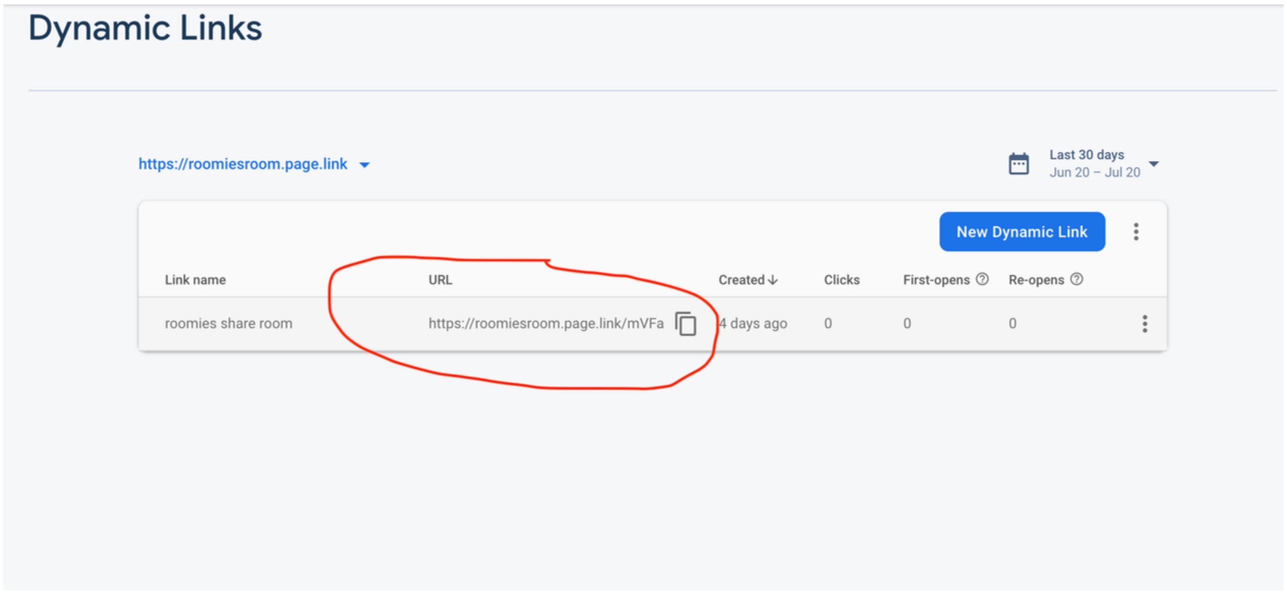
Custom URL or Google Play Instant Experience

Advanced Settings (optional)

☐ Open Google Play for versions lower than

select the app where those who dont have the app installed will be directed on android devices

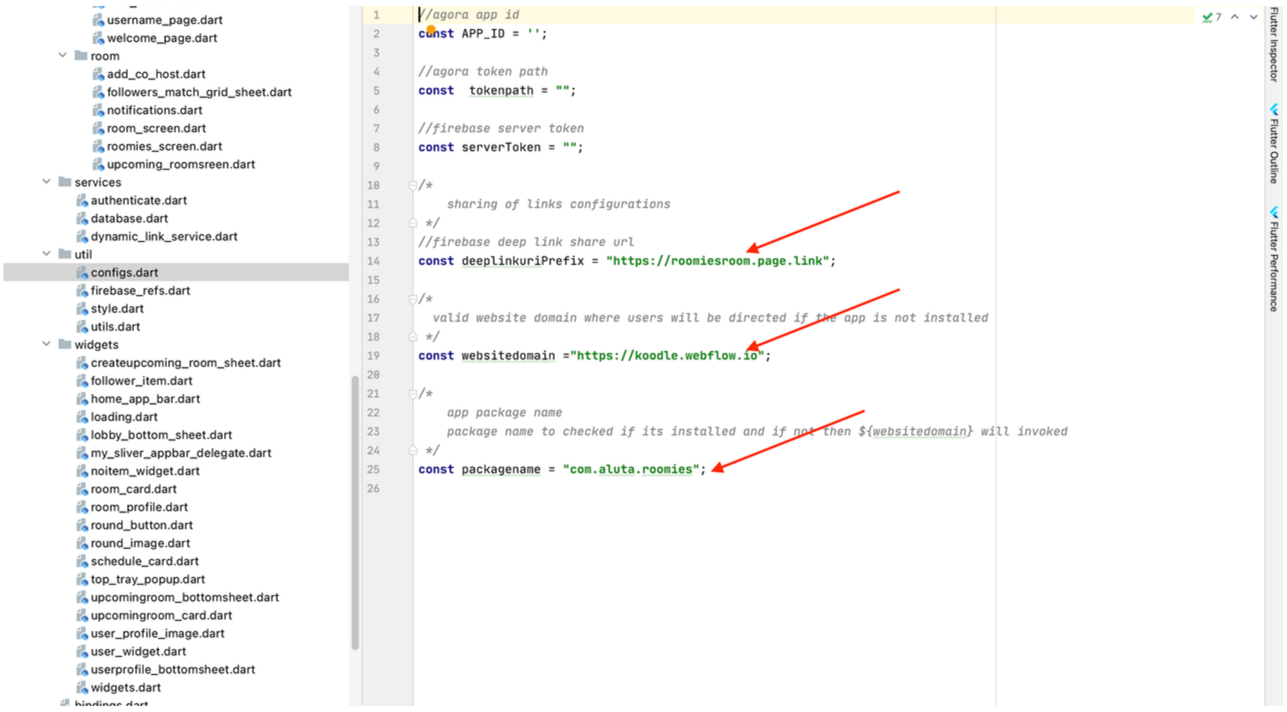
fig.6



Link name	URL	Created	Clicks	First-opens	Re-opens
roomies share room	https://roomiesroom.page.link/mVFa	4 days ago	0	0	0

*copy the url prefix without the last path, e.g https://roomiesroom.page.link and paste this link in your source code as shown in the picture below under **deeplinkuriPrefix** , under **websitedomain** replace with the domain you entered in firebase check fig.3 under deep link url, and package name to your app **packagename**, for the purpose of this documentation its **com.aluta.roomies** check fig.7 for more guidance*

fig.7



```dart
//agora app id
const APP_ID = '';

//agora token path
const  tokenpath = "";

//firebase server token
const serverToken = "";

/*
    sharing of links configurations
 */
//firebase deep link share url
const deeplinkuriPrefix = "https://roomiesroom.page.link";

/*
  valid website domain where users will be directed if the app is not installed
 */
const websitedomain ="https://koodle.webflow.io";

/*
    app package name
    package name to checked if its installed and if not then ${websitedomain} will invoked
 */
const packagename = "com.aluta.roomies";

```

lastly run the following commands

```
flutter clean
flutter pub get
flutter run
```

for ios build

```
flutter clean
pod deintegrate
pod install
flutter run
```

Vuala!! your setup process is complete and you can now run your project on your device or emulator and enjoy!