

Push notification for Flutter apps using FCM and Node js (without firebase function).



Bassem Karbia

Sep 20 · 6 min read



Push notification for Flutter apps using FCM and Node js



If you wasted a lot of time on the web looking for how to use FCM with Flutter and node js without the Firebase function in the easiest way and you

haven't found what you need, now you are in the right place.

Also if you are a beginner don't worry i will explain all the process to you just you must have some basics in Flutter and a node js project.

We will do it in 3 steps:

*SetUp the firebase project

*Flutter part

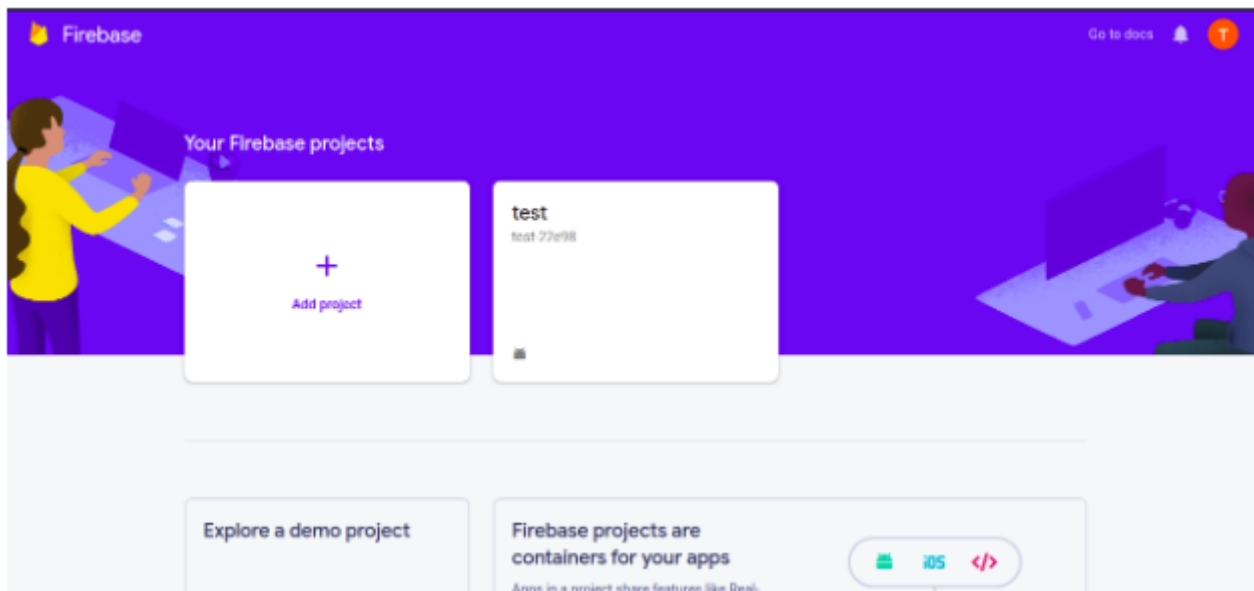
*Write the function of push notification on Node js

Note: you can do it whatever your database is, for my case i use mongoDB.

So lets start go go go....

*SetUp the firebase project

*Open <https://console.firebase.google.com/>

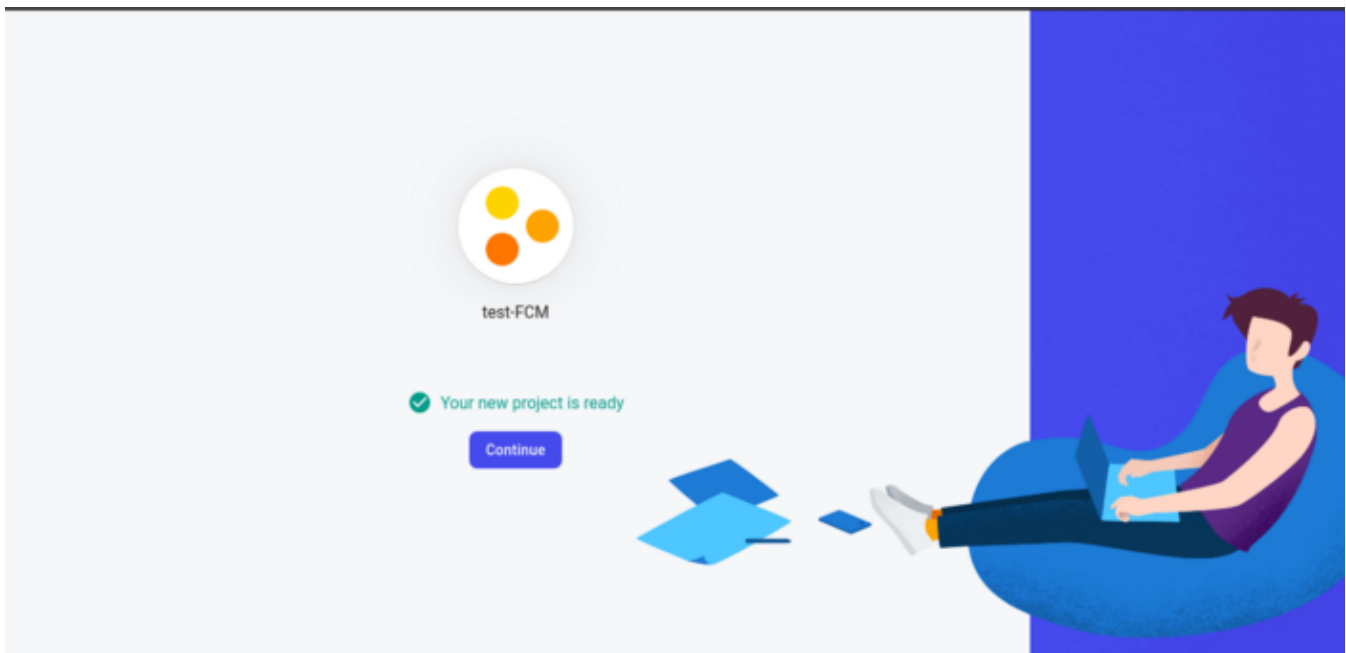


*Add a project(if you have one skip this step)

- enter the name of project



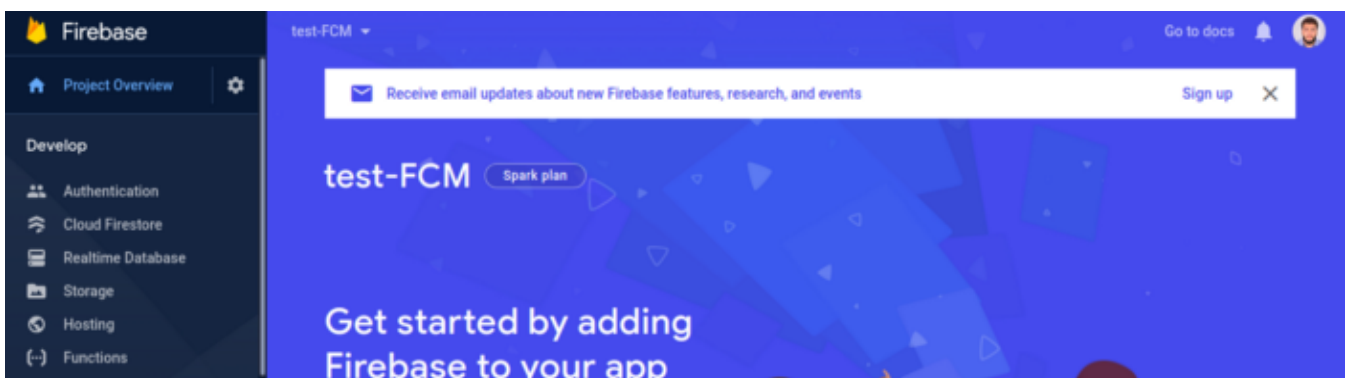
- click continue in the next two steps and wait until the project is created

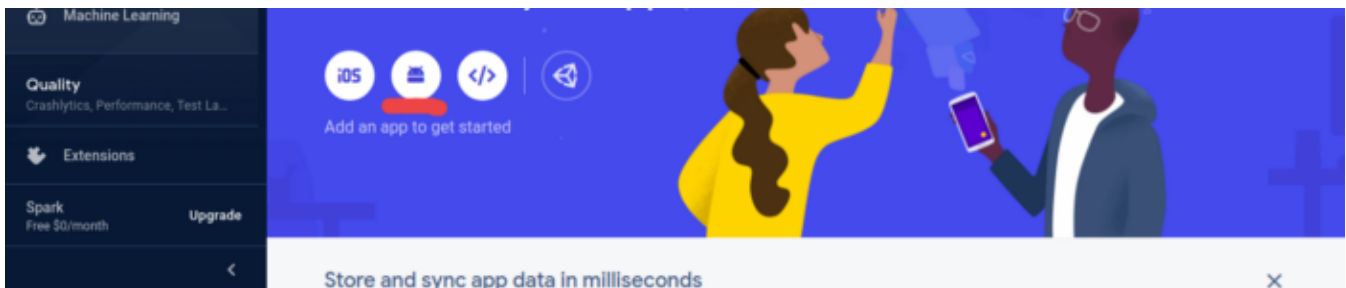


Setup Firebase project in Android

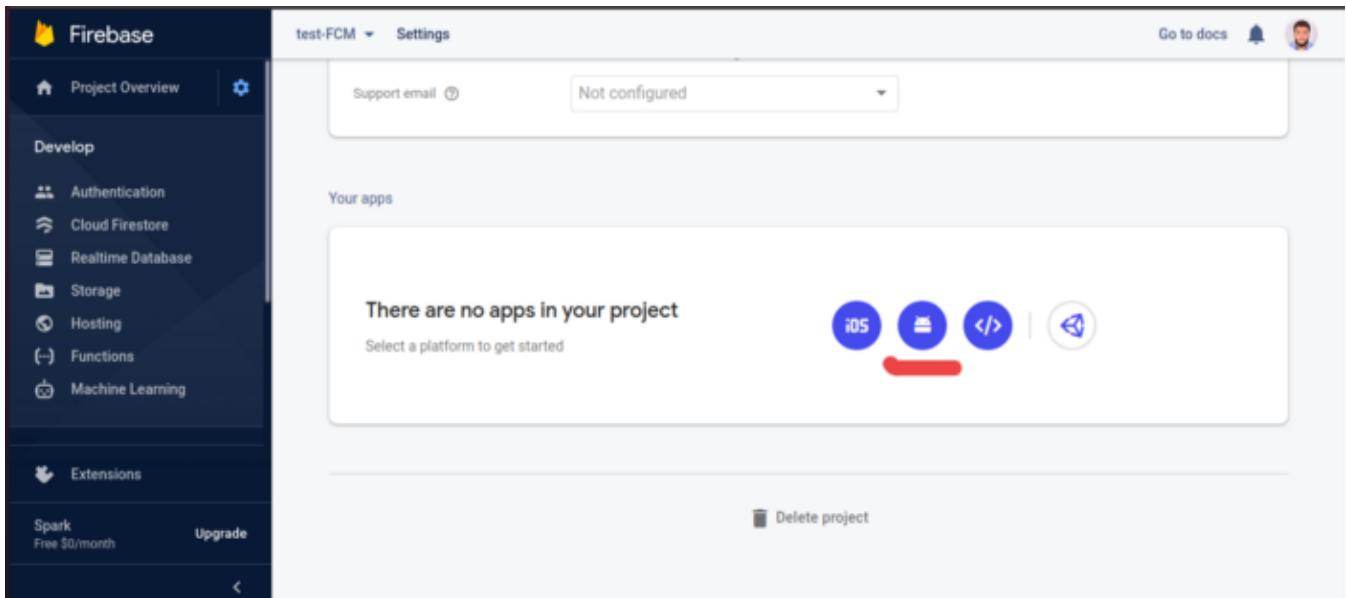
Step1:

Click to the android icon



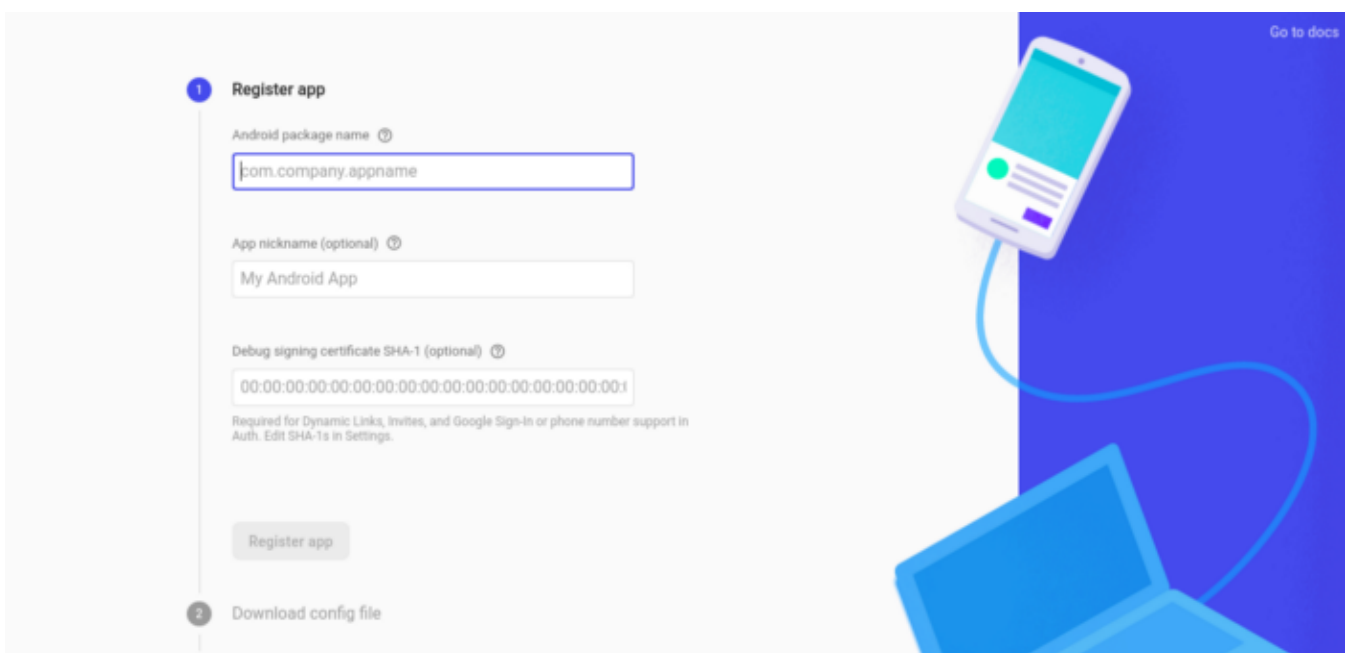


If you have a project so you will find this in the project setting



Step2:

Add Android package name



you find that in :

<project-name>/android/app/src/main/AndroidManifest.xml

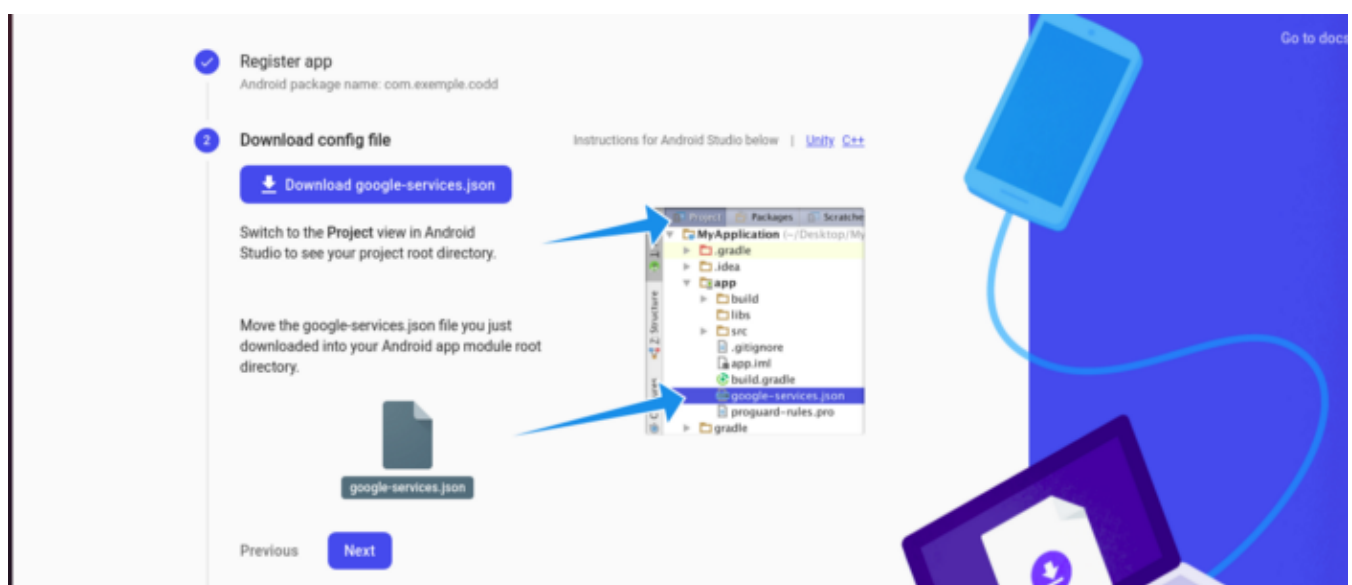
```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.test_notification">

    <!-- io.flutter.app.FlutterApplication is an android.app.Application
    that
        calls FlutterMain.startInitialization(this); in its onCreate
        method.
        In most cases you can leave this as-is, but you if you want
        to provide
        additional functionality it is fine to subclass or reimplement
        FlutterApplication and put your custom class here. -->
    <application
        android:name="io.flutter.app.FlutterApplication"
        android:label="@string/app_label"
        android:icon="@mipmap/ic_launcher">
```

For the App nickname and Debug signing certificate SHA-1 are not required so we don't have to fill them in our case.

Step3:

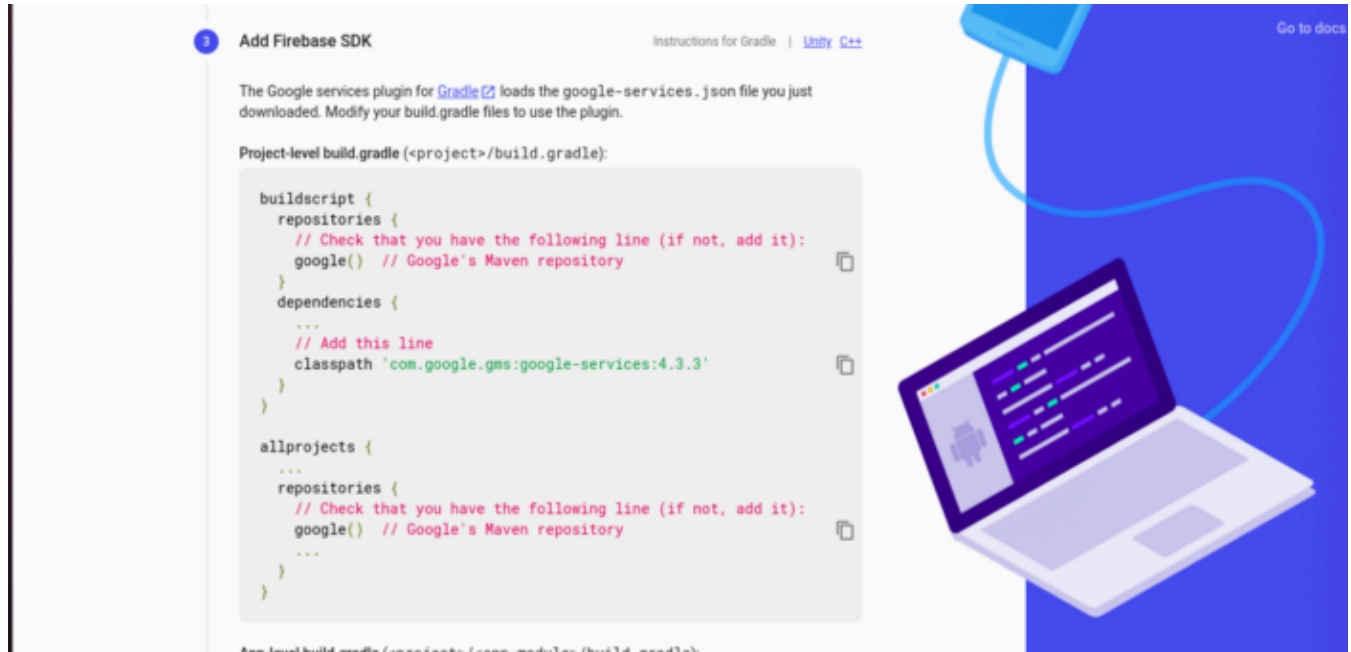
Download **google-service.json** and move to folder <project-name>/android/app and make sure the file name is exactly google-service.json



3 Add Firebase SDK

Step4:

This step is to Add Firebase SDK to your app



3 Add Firebase SDK [Instructions for Gradle](#) | [Unity](#) [C++](#) [Go to docs](#)

The Google services plugin for [Gradle](#) loads the `google-services.json` file you just downloaded. Modify your `build.gradle` files to use the plugin.

Project-level `build.gradle` (<project>/`build.gradle`):

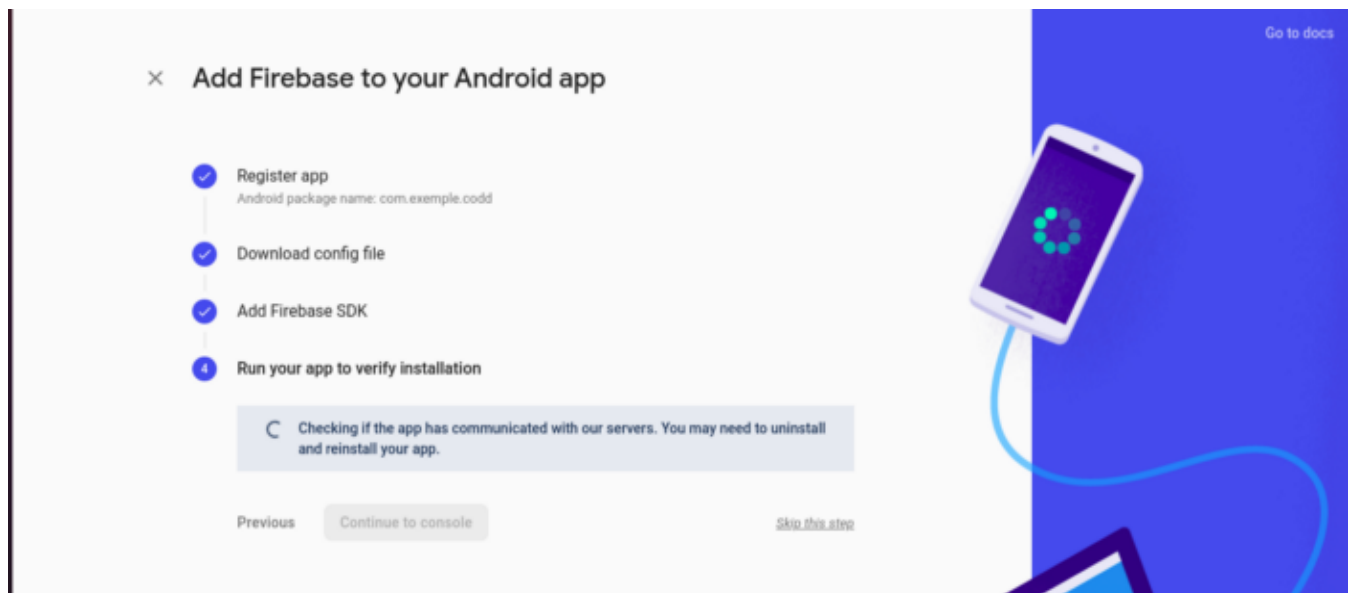
```
buildscript {
  repositories {
    // Check that you have the following line (if not, add it):
    google() // Google's Maven repository
  }
  dependencies {
    ...
    // Add this line
    classpath 'com.google.gms:google-services:4.3.3'
  }
}

allprojects {
  repositories {
    // Check that you have the following line (if not, add it):
    google() // Google's Maven repository
    ...
  }
}
```

App-level `build.gradle` (<project>/`app/build.gradle`):

In your <project-name>/`build.gradle` file check if these lines exist.

Step5:



Add Firebase to your Android app [Go to docs](#)

- ✓ Register app
Android package name: com.example.codd
- ✓ Download config file
- ✓ Add Firebase SDK
- 4 Run your app to verify installation

⌚ Checking if the app has communicated with our servers. You may need to uninstall and reinstall your app.

[Previous](#) [Continue to console](#) [Skip this step](#)

wait until firebase detect your app test device(open the network in your device and run the app)

Note: it is possible that firebase will not detect your test device, but that's okay, just click Skip this step (firebase will detect your device later)

Setup Firebase project in ios

For iOS the app are required to generate a certificate for the Apple Push Notification service (APNs) and enable background services in Xcode.

Rather than duplicating the contents of the documentation, I recommend following the official Firebase installation guide <https://firebase.google.com/docs/cloud-messaging/ios/certs>

(I just made captures not code so that you write the code yourself)

*Flutter part

Step1:

Add to your **pubspec.yaml** the [firebase_messaging.plygin](#)

```
dependencies:
  flutter:
    sdk: flutter
  firebase_messaging: ^7.0.0

# The following adds the Cupertino Icons font to your applicatio
n.# Use with the CupertinoIcons class for iOS style icon
  cupertino_icons: ^0.1.3

dev_dependencies:
  flutter_test:
    sdk: flutter
```

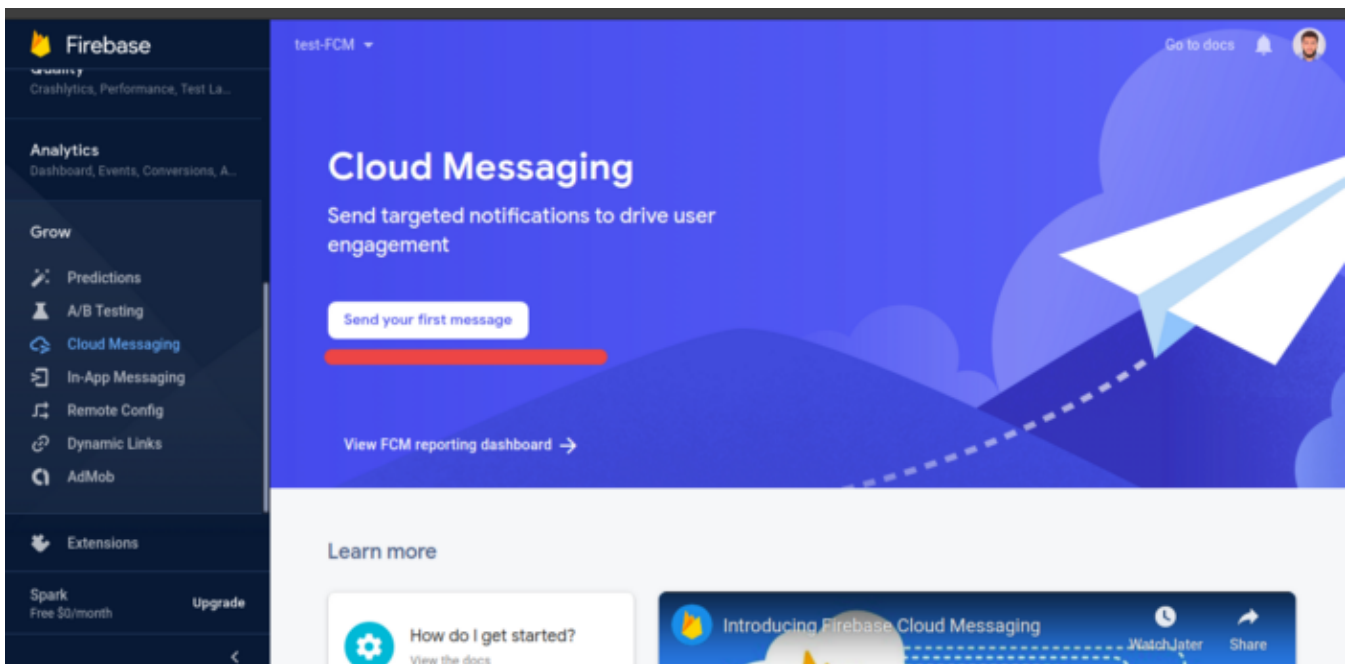
just follow the instructions you will find in the plugin documentation, there is some configuration you have to add it in **android/build.gradle** and **android/app/build.gradle**.

Now Add this in the **android/app/src/main/AndroidManifest.xml**:

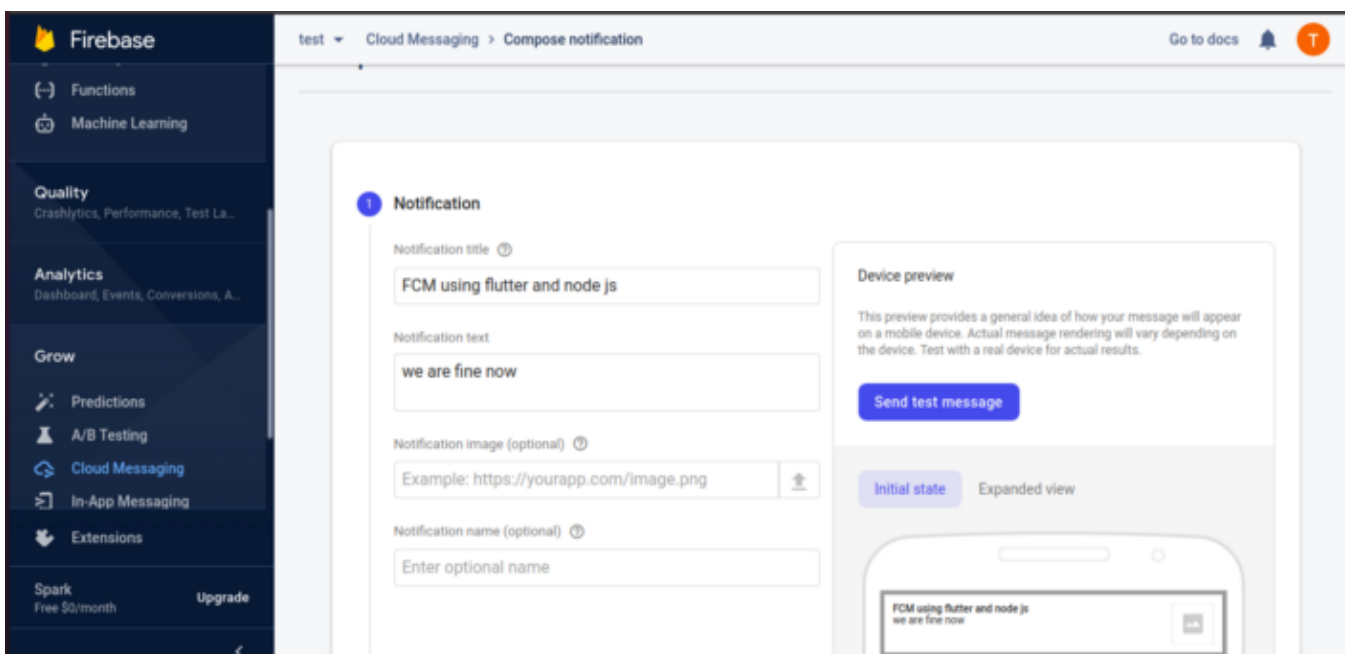
```
<intent-filter>
  <action android:name="FLUTTER_NOTIFICATION_CLICK" />
  <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

I will explain to you why we add this later ,just trust me 😊

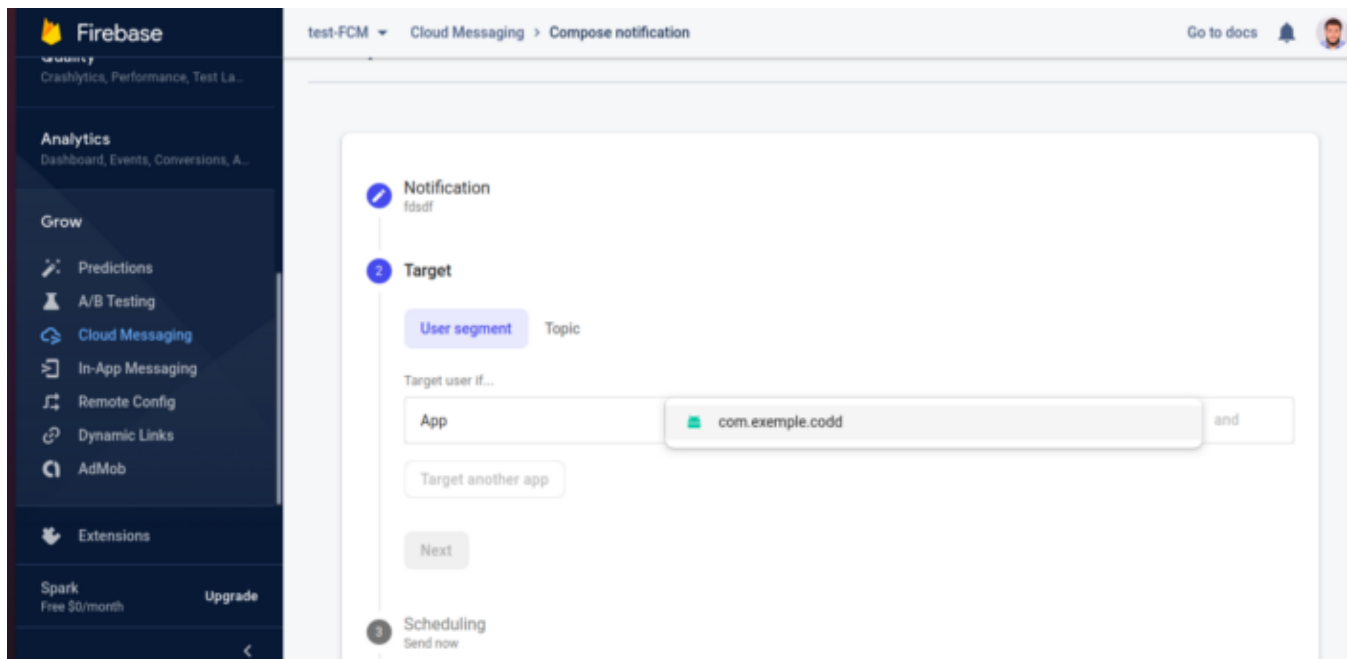
*Now we will try to send notification to the app from the firebase console



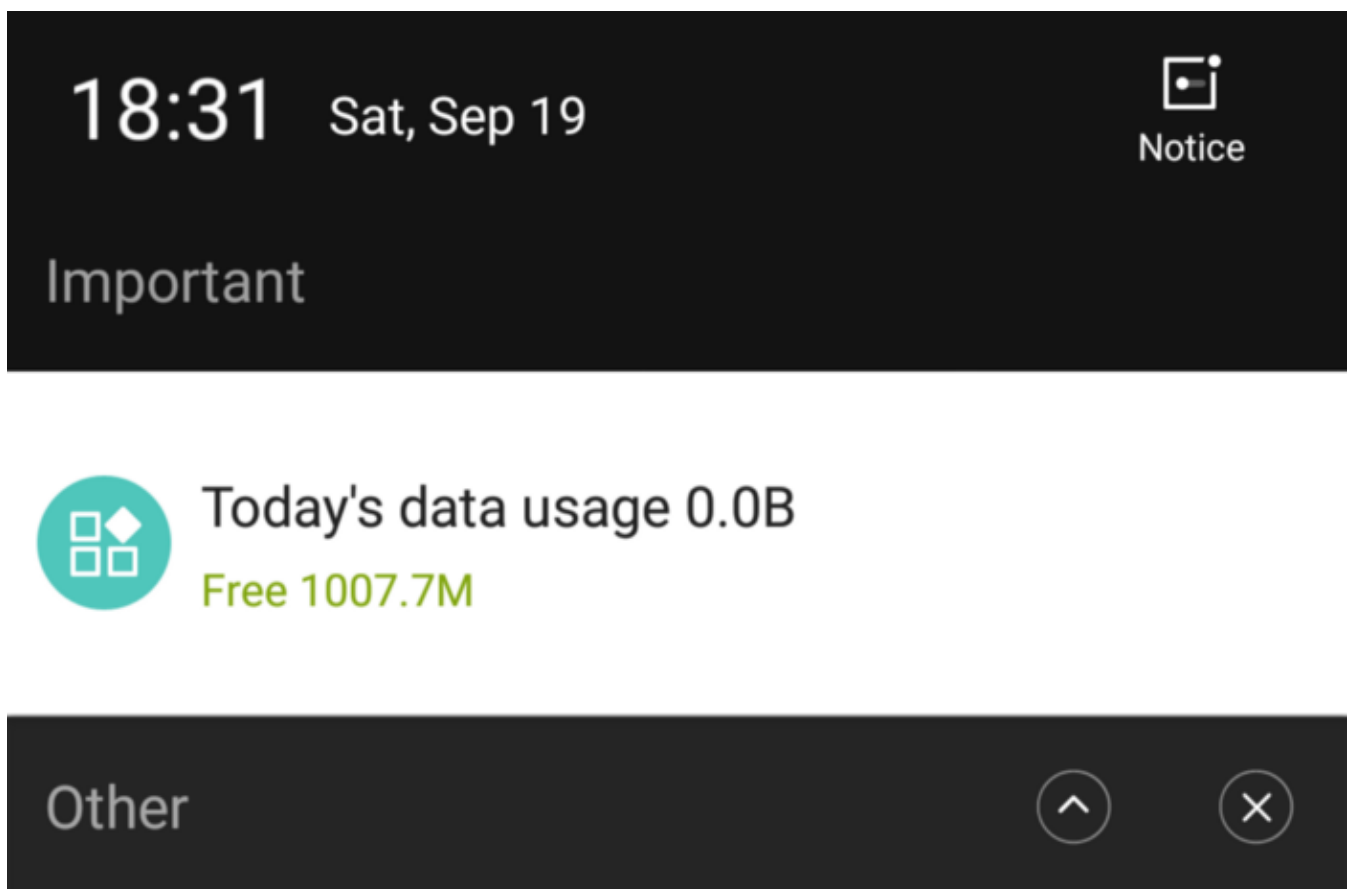
*Then we write the title and the text



*Choose our target And click **send**



Now check you device and you will see the notification like this



test_notification • now

FCM using flutter and node is

we are fine now



Step2:

Now we will handle the notification according to state of the app so there is three cases:

- 1/the app is open and running in the foreground. (we will use the **onMessage**).
- 2/he app is closed, but still running in the background (we will use the **onResume**).
- 3/the app is fully terminated (we will use the **onLaunch**).

So you remember now when i told you to add the **intent** in the **AndroidManifest.xml** file ,if yes so thank you for trusting me 😊

the intent Allow us to use (**onMessage,onResume,onLaunch**).

*Now change your main.dart file like this

```
import 'package:flutter/material.dart';
import 'package:test_notification/firebase_messaging.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  // This widget is the root of your application
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'FCM with flutter and node js',
      theme: ThemeData(
        primarySwatch: Colors.blue,
```

```

        primarySwatch : Colors.blue,
        visualDensity : VisualDensity.adaptivePlatformDensity,
      ),
      home:Scaffold(
        : Firebase_Messaging()
      body ),
    );
  }
}

```

*Create a new file called **firebase_messaging.dart** and write this code

```

import 'package:flutter/material.dart';
import 'package:firebase_messaging/firebase_messaging.dart';

class Firebase_Messaging extends StatefulWidget {
  @override
  _Firebase_MessagingState createState() =>
  _Firebase_MessagingState ();
}

class _Firebase_MessagingState extends State<Firebase_Messaging> {
  final FirebaseMessaging _fc =FirebaseMessaging();

  @override
  @override
  Widget build(BuildContext context) {
    return Container(

    );
  }
}

```

I just created a **stateful** widget and initialize a **FirebaseMessaging** const called **_fc**.

Step3:

Create a function **configureCallbacks()** and we called the **configure** function of **FirebaseMessaging** with the 3 parameters **onMessage,onResume,onLaunch** so the code look like this:

```
class _FireBase_MessagingStat extends State<FireBase_Messaging> {
  final FirebaseMessaging _fc =FirebaseMessaging();

  @override
  Widget build(BuildContext context) {
    return Container(

    );
  }

  void configureCallbacks() {
    _fc.configure(
      : (message) async{
onMessage
    },
    onResume: (message) async {
    },
    onLaunch: (message) async {
    }
    );
  }
}
```

Step4:

We will now add how the app acts when it receives a notification in the three cases so the **configureCallbacks** function look like this:

```
void configureCallbacks() {
  _fc.configure(
    onMessage: (message) async{
      showDialog(
        context: context,
        builder: (context) => AlertDialog(
          content: ListTile(
            subtitle: Text('onMessage'),
```

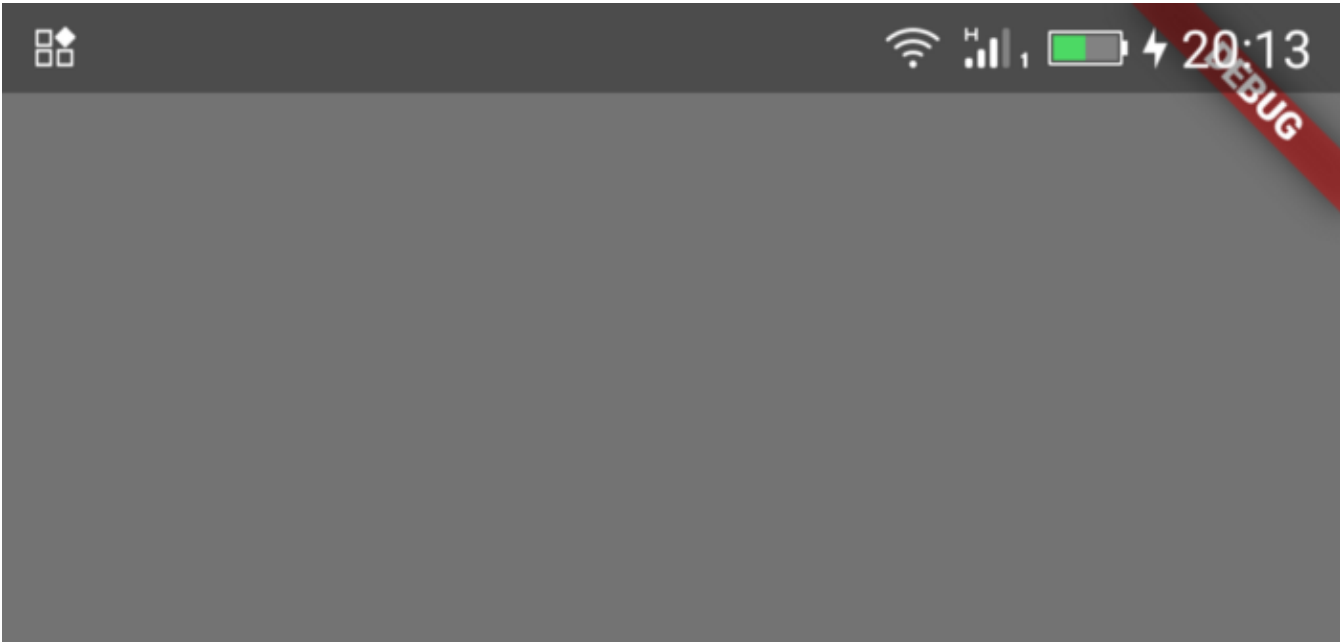
```

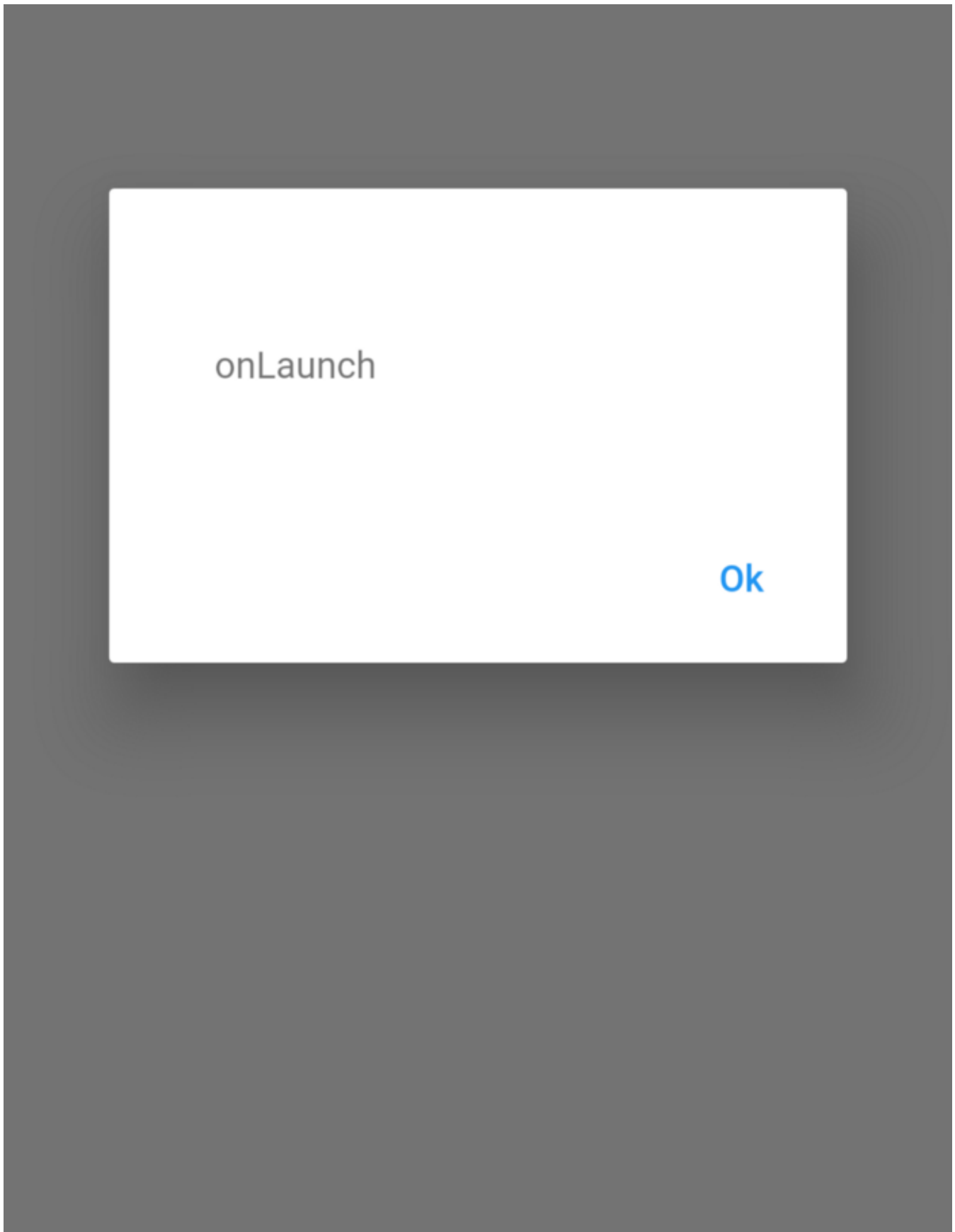
    ),
    actions: <Widget>[
      FlatButton(
        child: Text('Ok'),
        onPressed: () => Navigator.of(context).pop(),
      ),
    ],
  ),
);
},
onResume: (message) async {
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      content: ListTile(
        subtitle: Text('onResume'),
      ),
      actions: <Widget>[
        FlatButton(
          child: Text('Ok'),
          onPressed: () => Navigator.of(context).pop(),
        ),
      ],
    ),
  );
},
onLaunch: (message) async {
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      content: ListTile(
        subtitle: Text('onLaunch'),
      ),
      actions: <Widget>[
        FlatButton(
          child: Text('Ok'),
          onPressed: () => Navigator.of(context).pop(),
        ),
      ],
    ),
  );
}
);
}
);
}
}

```

Just i add a **showDialog** function ,you can add what you want depends on your case.

So now when we click on the notification the app acts like this:





Step5:

Add the **configureCallbacks** function to **initstate** and the code look like this:



```
@override
void initState() {
  super.initState();

  configureCallbacks();
}
@override
Widget build(BuildContext context) {
  return Container(

  );
}
```

Step6:

So now we need to synchronize between the app and our server ,I will explain to you how this happen ,we send a notification with a specific **topic** from our server to firebase(**we will create this function later in the node js part**) and we will make a listener in our app to this topic so it's a simple function of **FirebaseMessaging** called **subscribeToTopic**.

```
void subscribeToEvent() {
  _fc.subscribeToTopic('Events');
}
```

here just i create the function and the name of the topic is 'Events'

Step7:(the last step in the Flutter part)

Finally ouff ... 🏆🥳🥳

So just add the subscribe function to **initState** then the **initState** look like this:

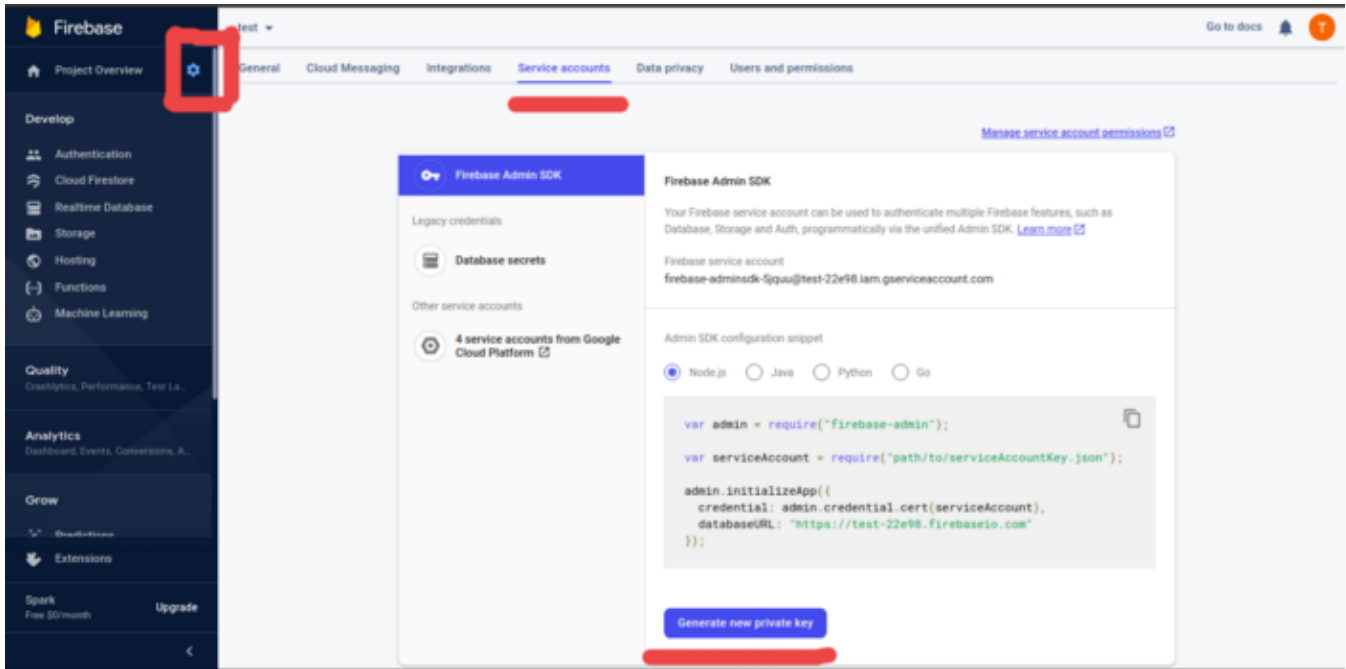
```
void initState() {
  super.initState();
  subscribeToEvent();
  configureCallbacks();
}
```

*Write the function of push notification on Node js

Step1:

download **firebase-service-account.json** place it in the root directory

You find this file in **project settings/service accounts** like this:



Step2:

Install fireBase admin on the project with :

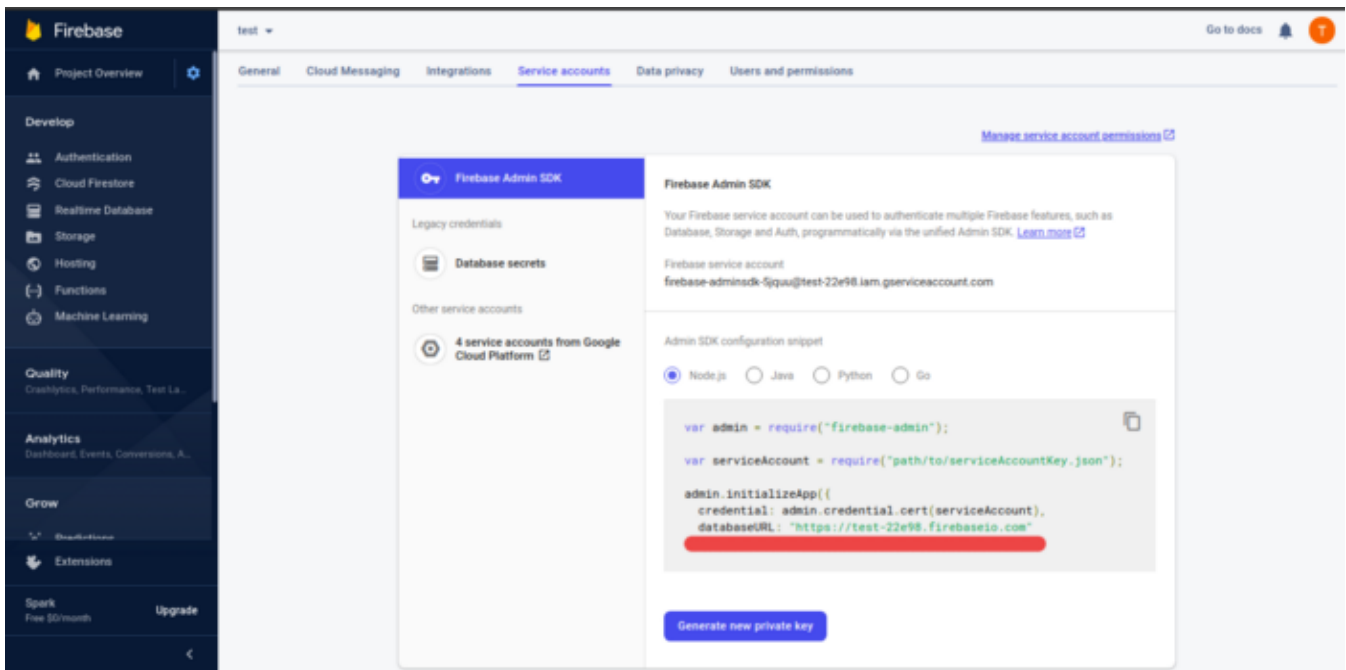
```
npm install — save firebase-admin
```

Step3:

Create a file called **firebaseNotification.js** and initialize firebase admin like this:

```
const admin = require("firebase-admin");
const serviceAccount = require("../firebase-service-account.json");
const FIREBASE_DATABASE_URL="Your firebase database url";
admin.initializeApp({
  credential: admin.credential.cert(serviceAccount),
  databaseURL: FIREBASE_DATABASE_URL
});
```


You find the firebase database url here:



Step4:

Create the function to send notification

```
export async function sendNotificationEvenetCreation(){
  try {
    var payload={notification:{title:'FCM using flutter and node js'
    ,body:'we are fine now'}
    ,data:{click_action:"FLUTTER_NOTIFICATION_CLICK"}}

    await admin.messaging().sendToTopic('Events',payload);

  } catch (error) {
    console.log(error);
  }
}
```

Make sure the topic name is the same as the one you wrote in the Flutter code.

Now just start the server and call the function so your app will receive the notification.

Note: we can also send notifications with device token and also customize the notification like the icon but for this article i just want to make it very simple.

You find me on Linkdin : [Bassem Karbia](#)

Thank you 😊😊

Flutter

Fcm

Nodejs

Push Notification

About

Help

Legal

Get the Medium app

