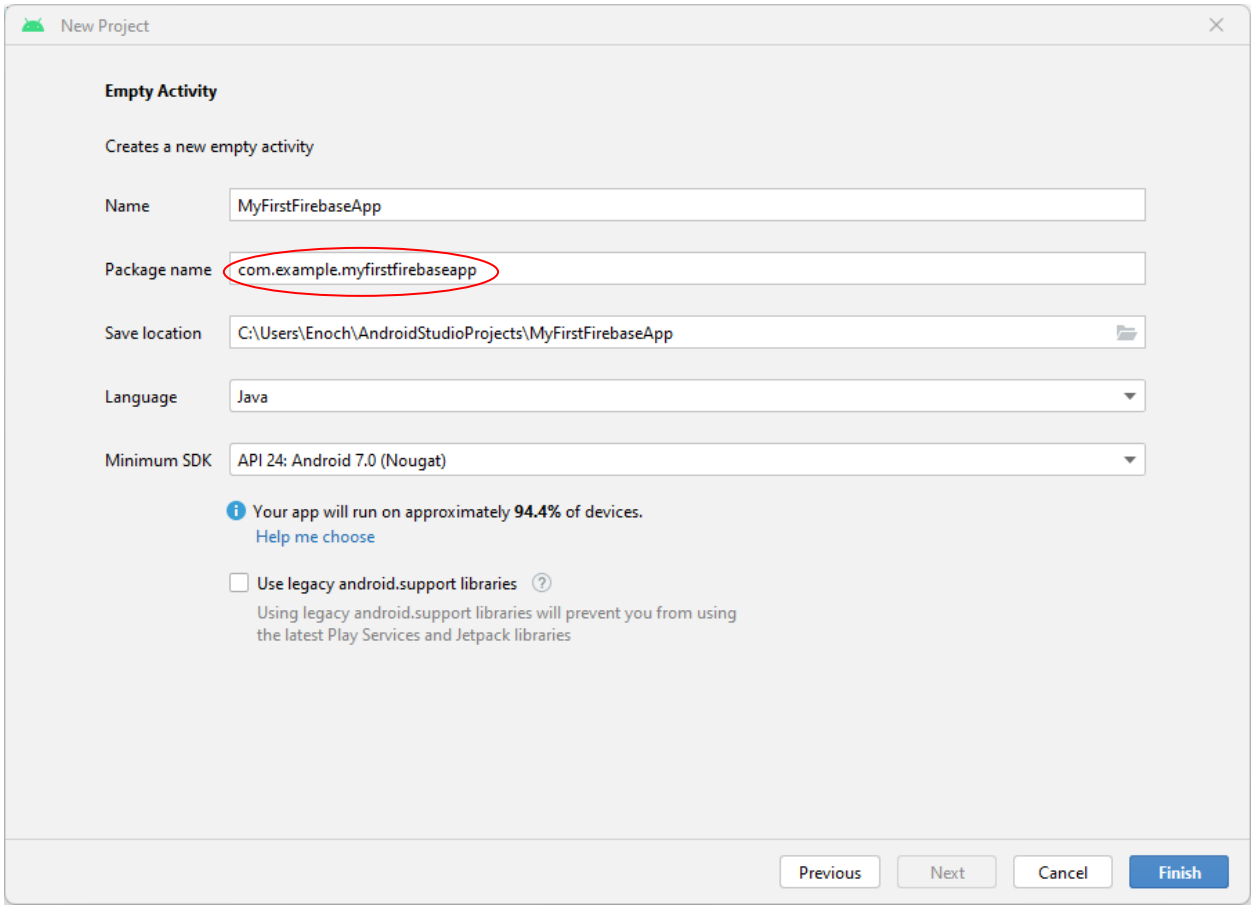
# Push Notification

This document shows how to send push notification to an Android device using Google's Firebase Cloud Messaging (FCM) service.

## *Android Studio*

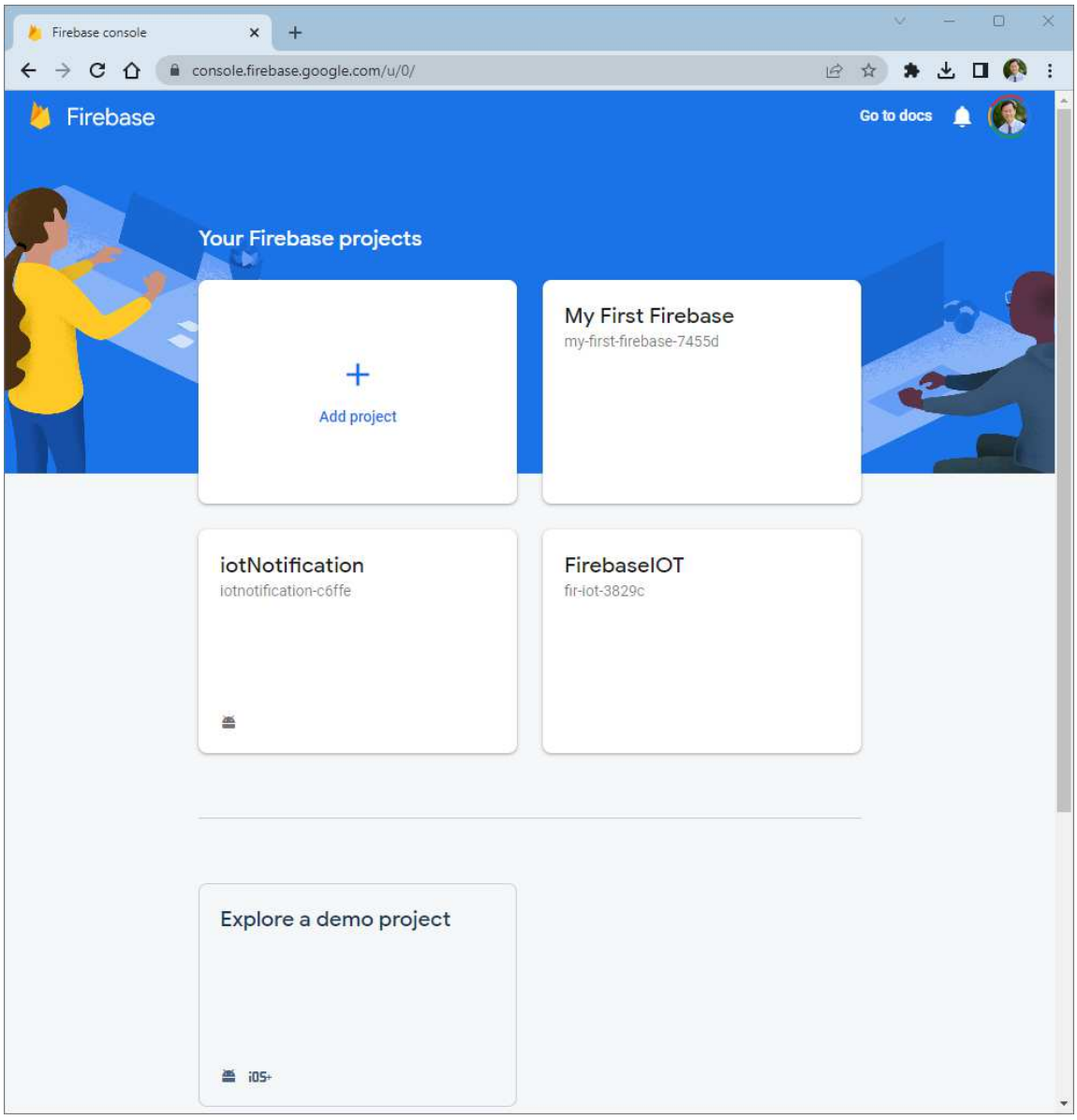
1) Create a new Empty project in Android Studio and name it MyFirstFirebaseApp. Note the Package name which is all in lowercase letters; it will be used in Step.



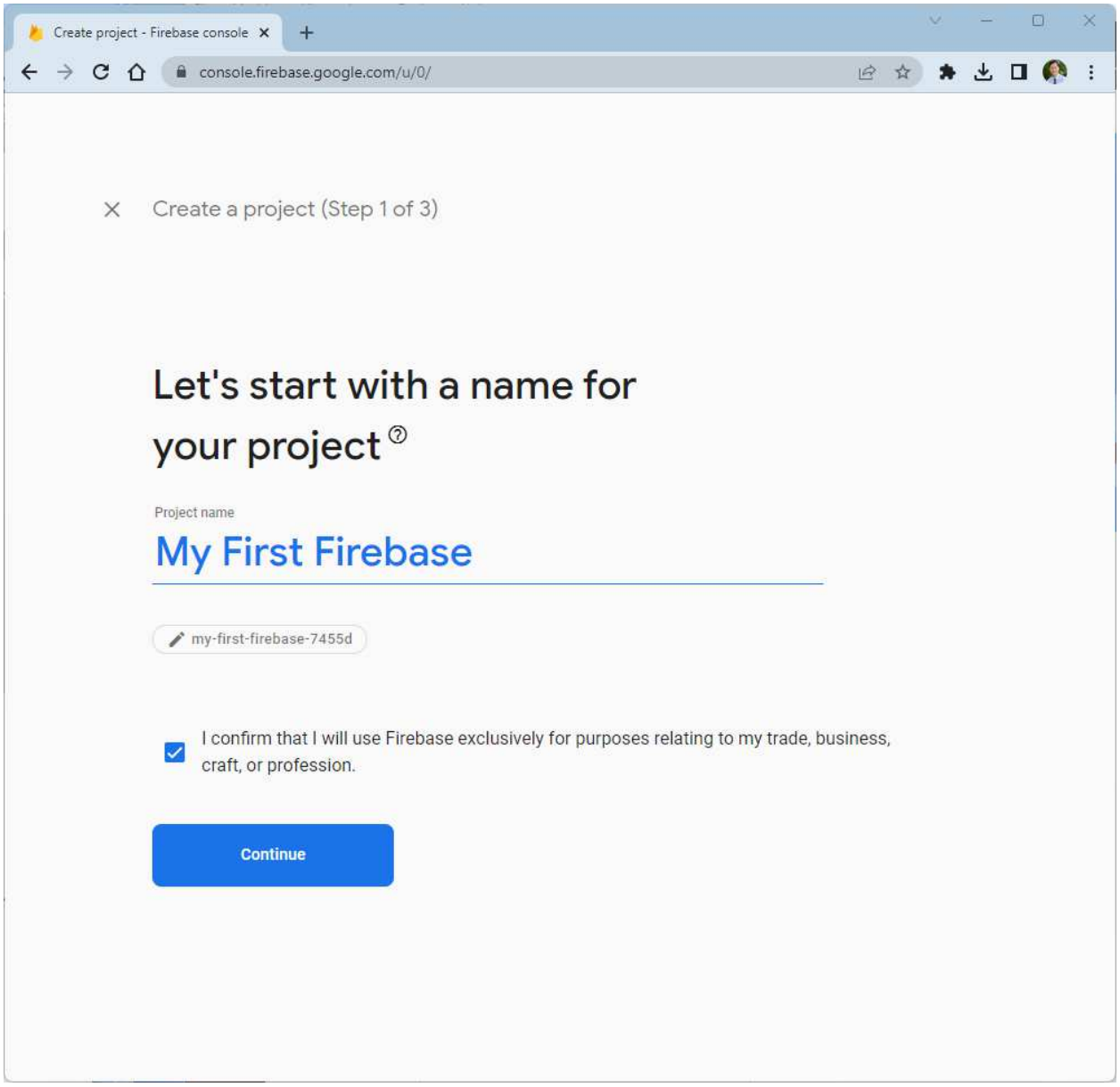
## *Setting up Firebase for your Android project*

2) Go to the Firebase console https://console.firebase.google.com/. Log in using your Google account.

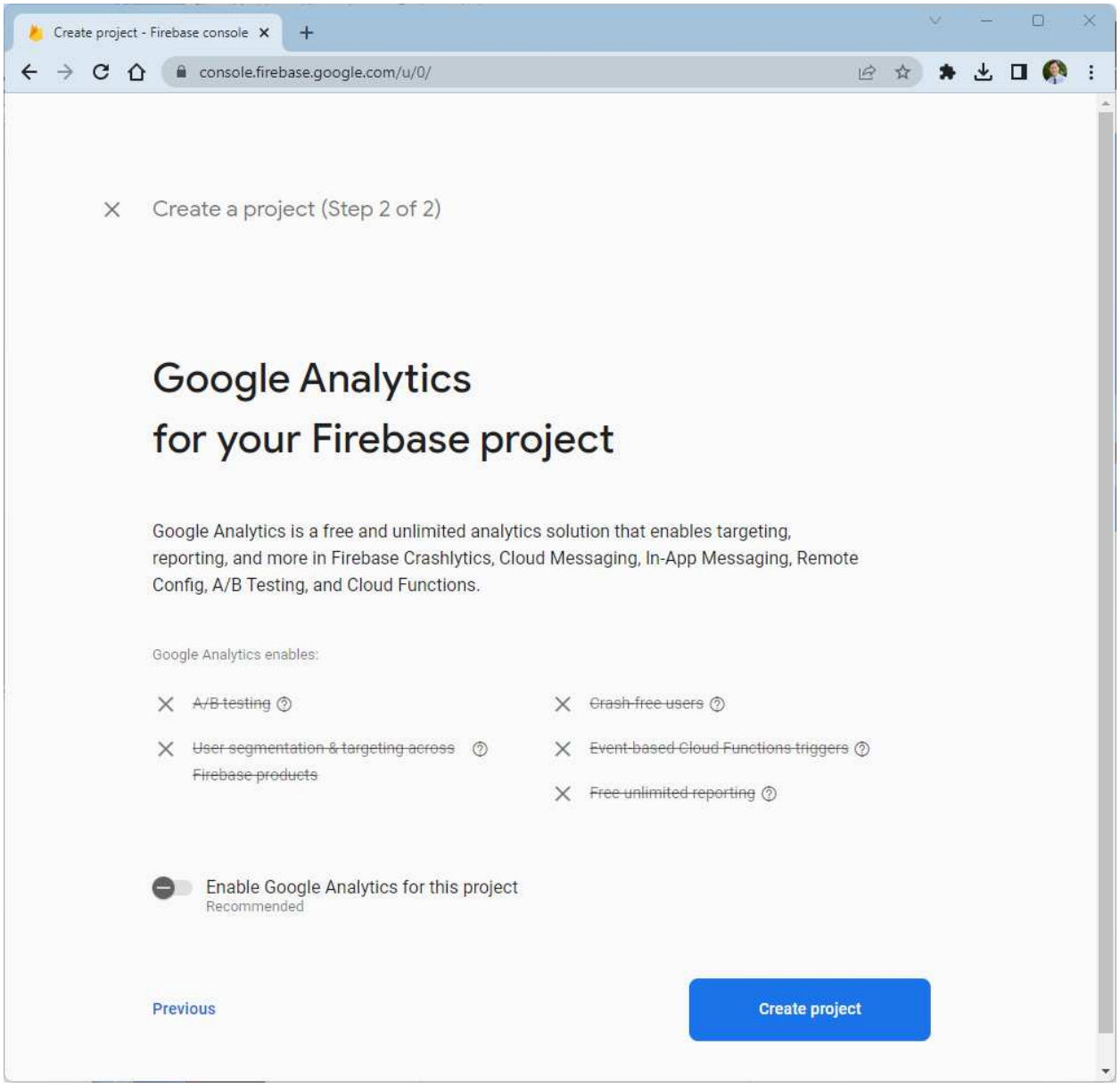
3) Click on **Add project** if you don't already have a project. If you already have a project, then just click on your project button and go to step 7)



4) In **Create a project Step 1 of 3**, type in a name for your project. Note that this name is NOT your Android package name so it can be different. Check the confirm box, and then click **Continue**.

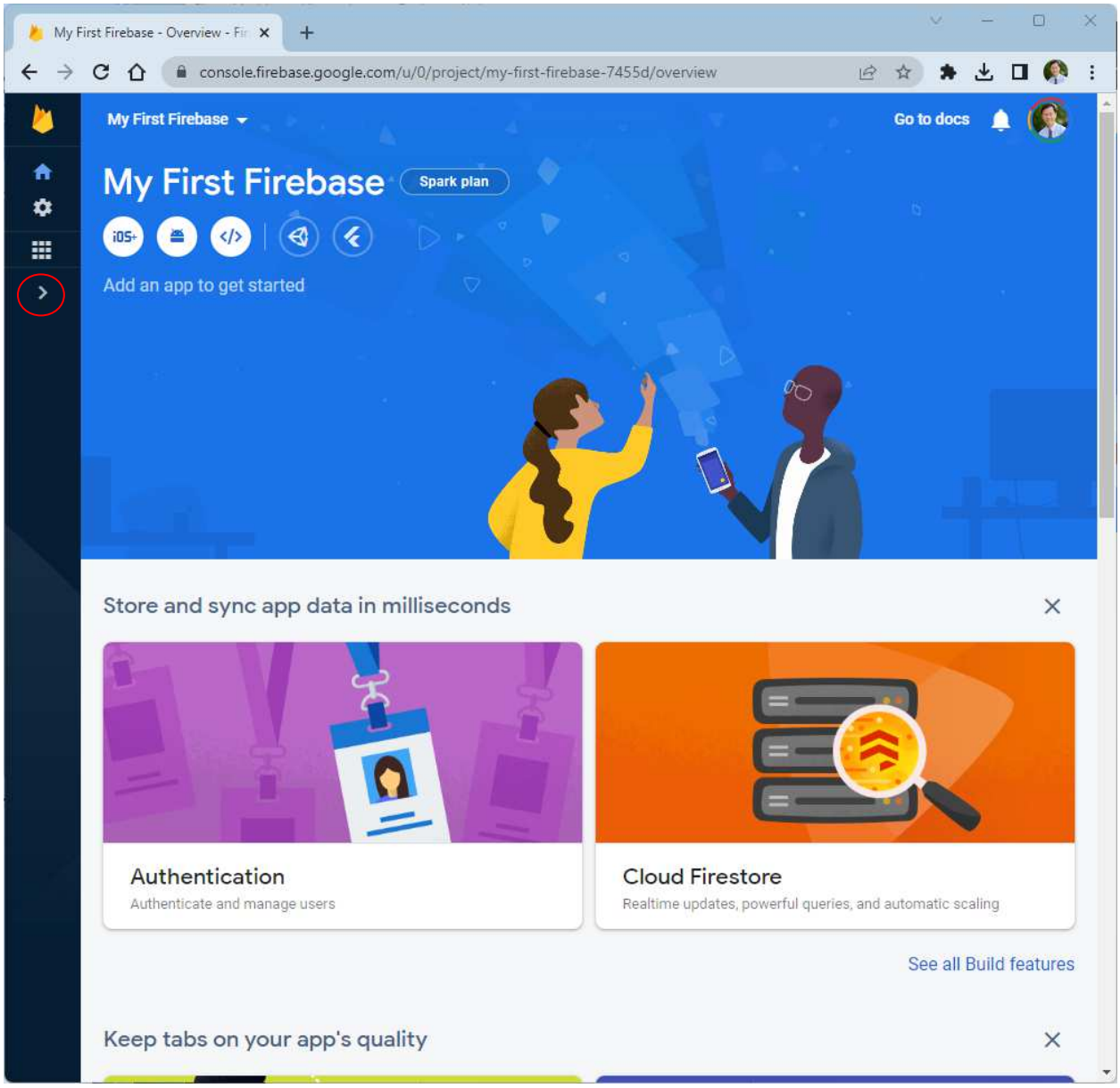


5) In Step 2 of 3, turn off the **Enable Google Analytics for this project**. This will change the Step 2 of 3 to Step 2 of 2. Click **Create project**.

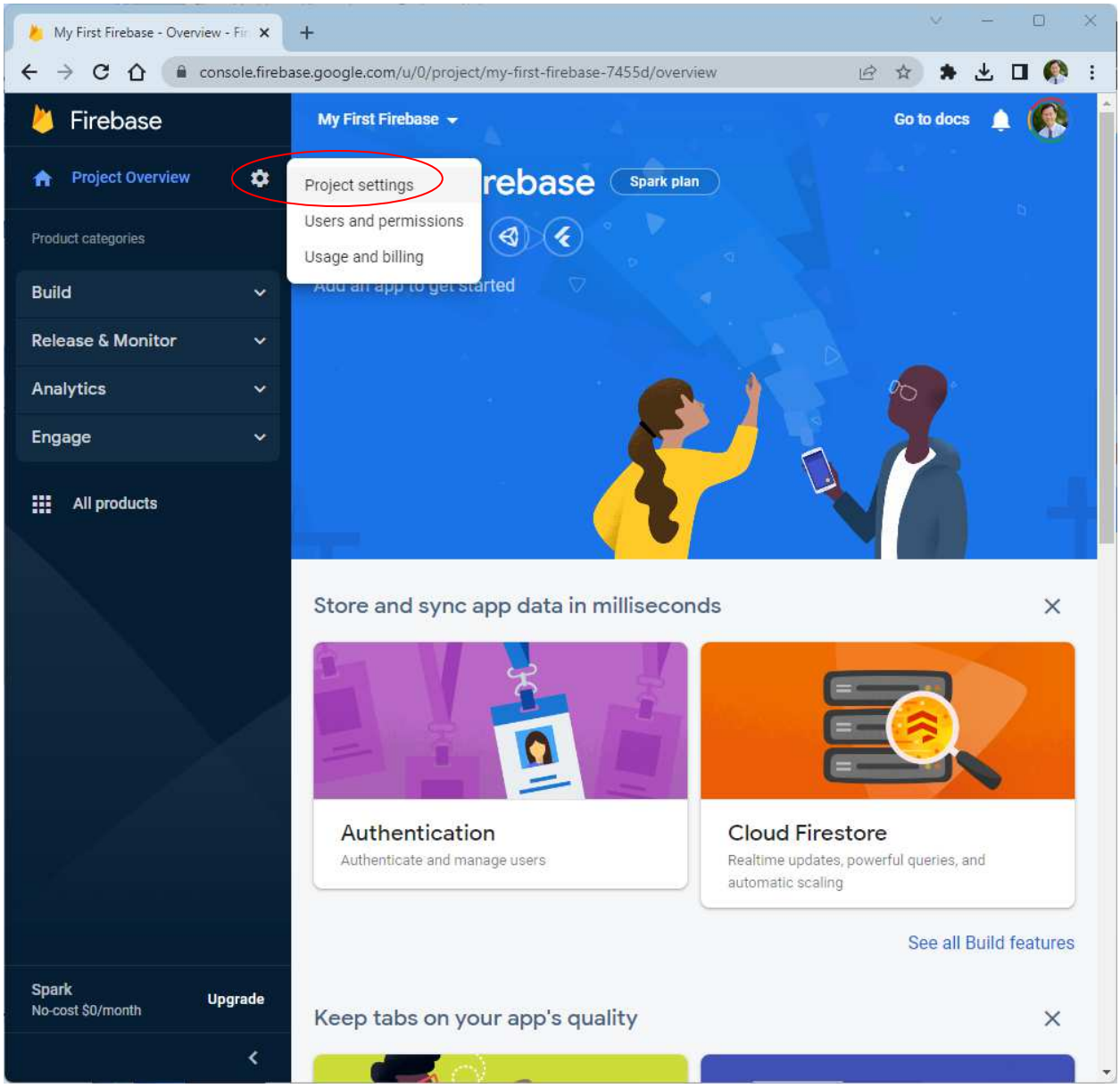



6) Wait for it to create the project, then click **Continue** when it says your new project is ready. It will show your project's console page.

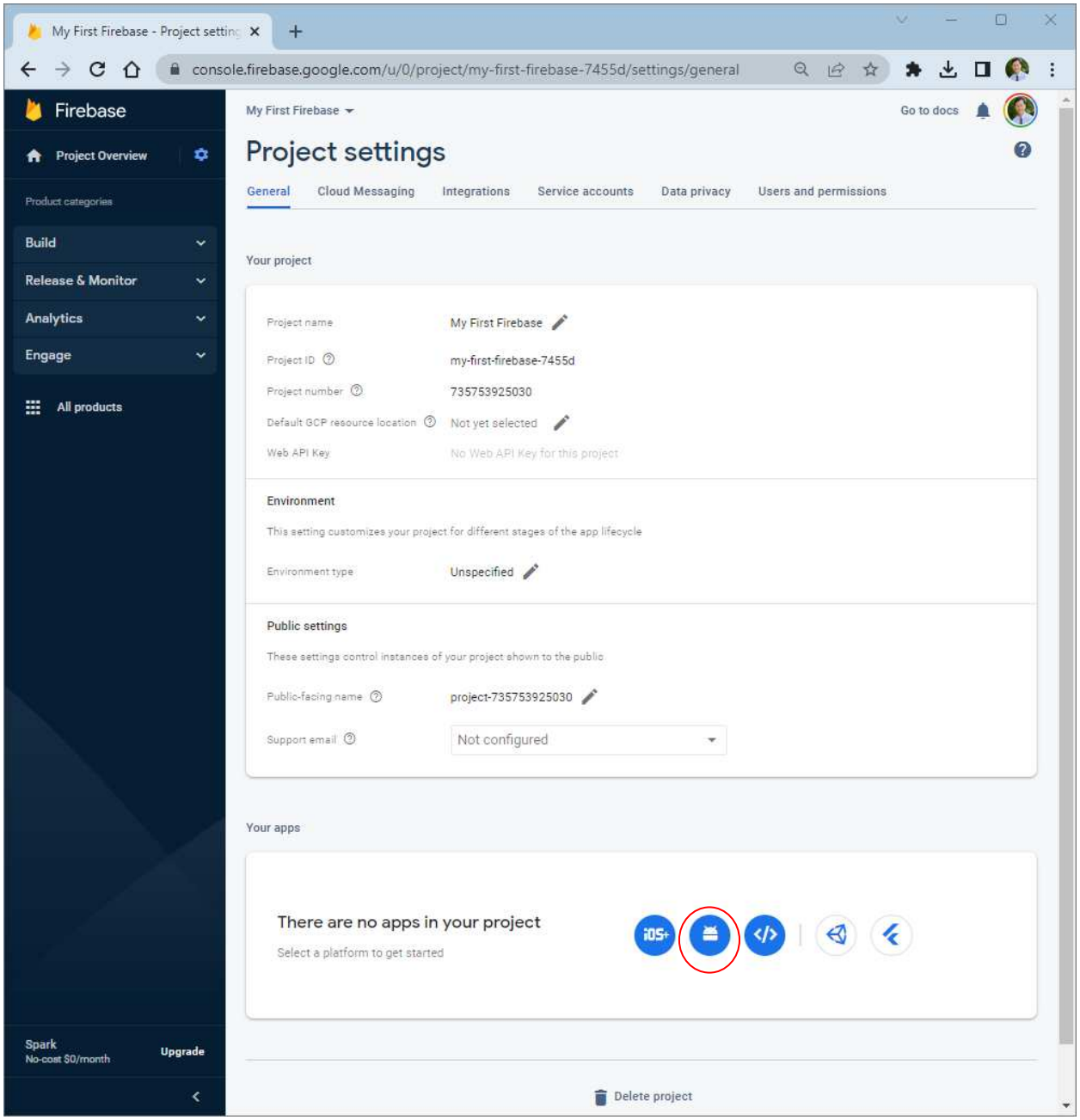
7) You can click on the greater-than sign > on the left sidebar to expand the sidebar if it is not already expanded.



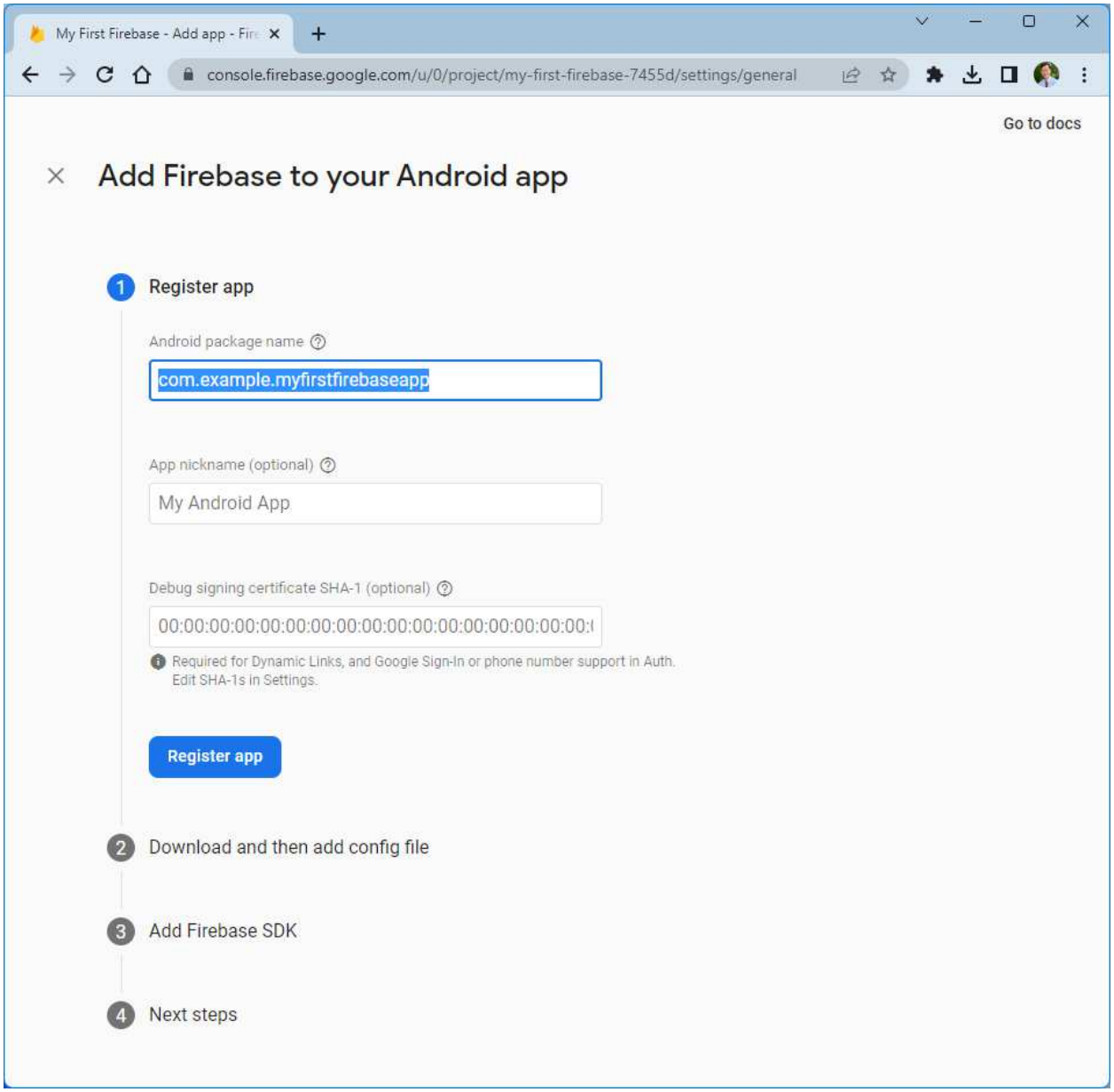
8) Click on the **gear** icon next to **Project Overview** in the sidebar and then click on **Project settings**.



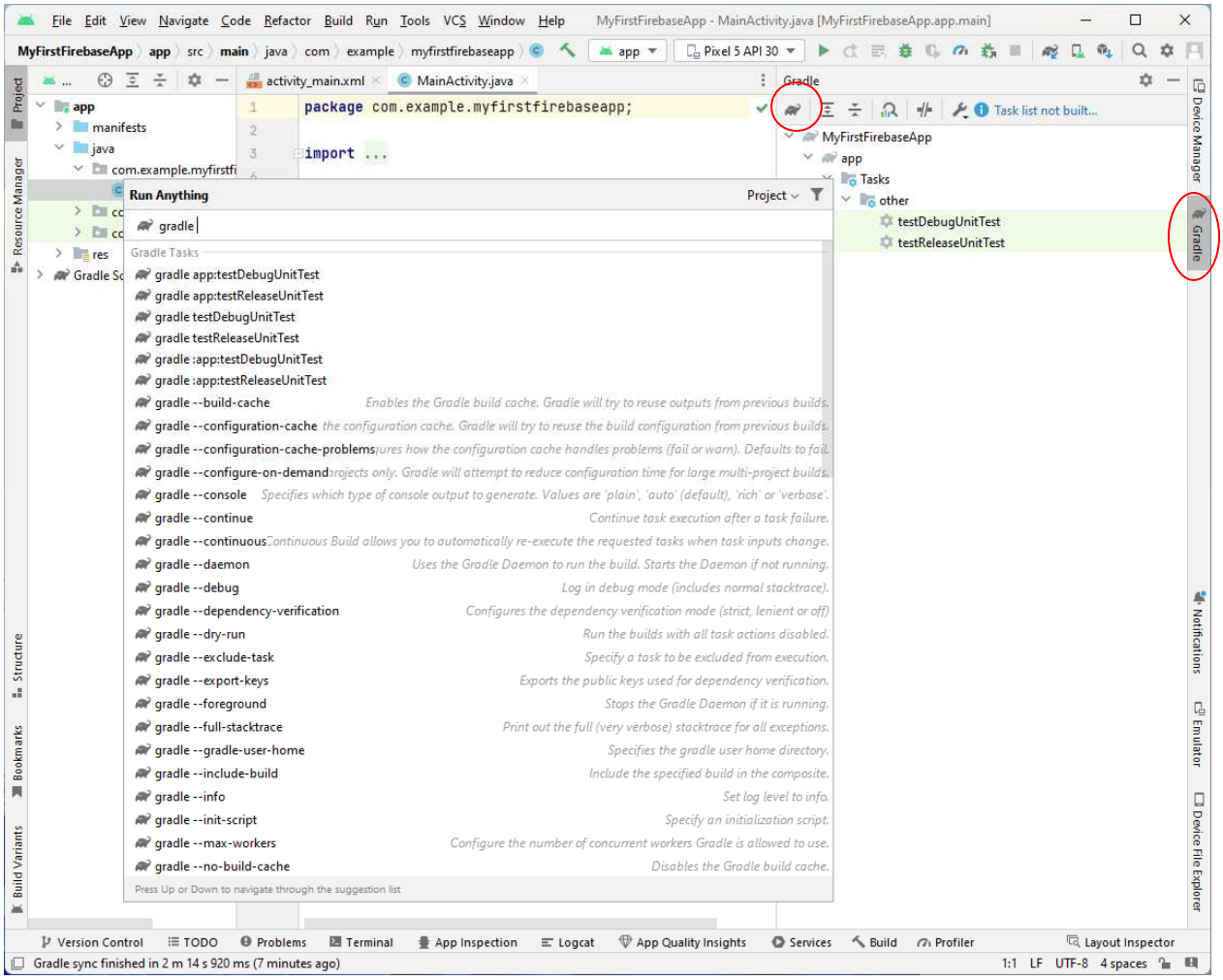
9) On the Project settings page, select the **General tab**. At the bottom of the page it'll say there are no apps in your project if this is the first time you come here. You need to add an app by clicking on the platform that you want to work with. Click on the **Android icon** platform. If you have already added an app then you will see that app listed as shown in the figure in step 21) and you can continue with step 21).



10) Go through the steps in the **Add Firebase to your Android app** window. In Step 1 **Register app**, type in your Android package name, all in lowercase. Make sure that this name matches exactly the one that you used in Step 1) when you created your Android project in Android Studio.

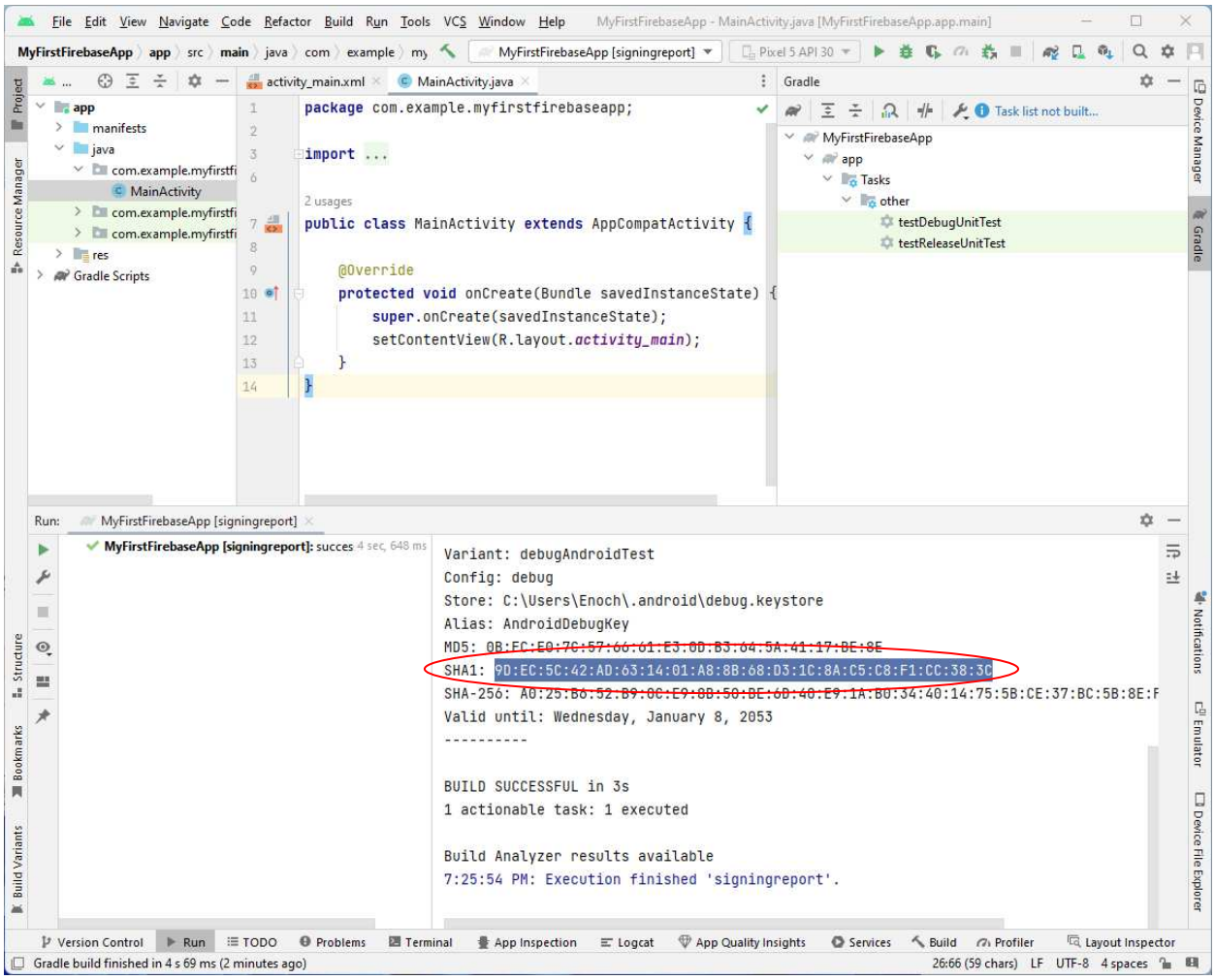


11) (Optional step) To get the **Debug signing certificate SHA-1**, go to your Android Studio project and click on the **Gradle** tab on the right edge of the IDE. Then click on **Execute Gradle Task** button  .



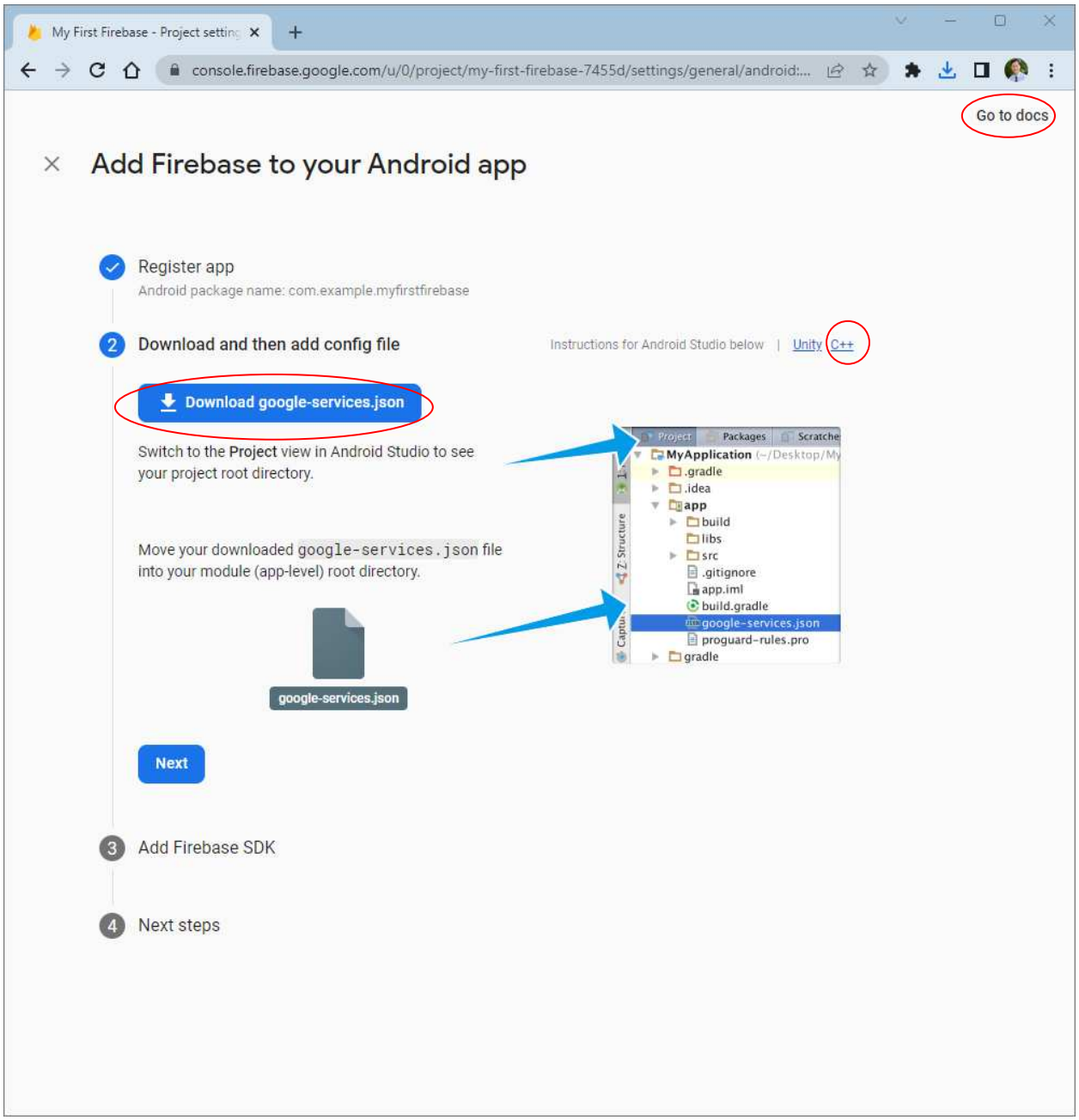

In the popup window, type in **signingreport** and press **Enter** to generate the report.


After the report is successfully generated you will see the SHA1 key in the report window. You can copy this key to the Firebase field in Step 10).

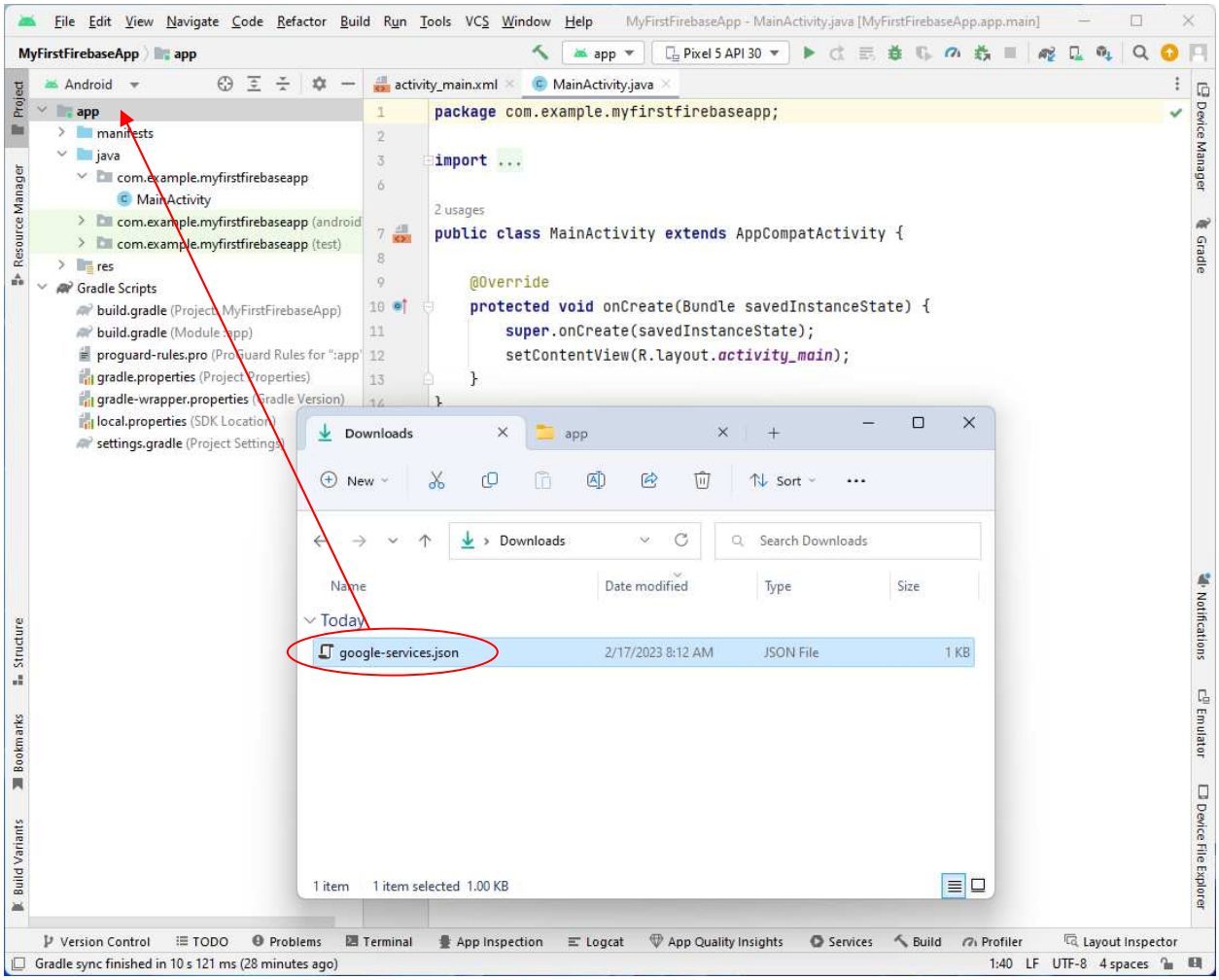

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
MyFirstFirebaseApp - MainActivity.java [MyFirstFirebaseApp.app.main]
MyFirstFirebaseApp app src main java com example my
MyFirstFirebaseApp [signingreport]
Pixel 5 API 30
activity_main.xml
MainActivity.java
Gradle
Task list not built...
MyFirstFirebaseApp
app
Tasks
other
testDebugUnitTest
testReleaseUnitTest
manifests
java
com.example.myfirstfi
MainActivity
com.example.myfirstfi
com.example.myfirstfi
res
Gradle Scripts
1 package com.example.myfirstfirebaseapp;
2
3 import ...
6
2 usages
7 public class MainActivity extends AppCompatActivity {
8
9 @Override
10 protected void onCreate(Bundle savedInstanceState) {
11 super.onCreate(savedInstanceState);
12 setContentView(R.layout.activity_main);
13 }
14 }
Run: MyFirstFirebaseApp [signingreport]
MyFirstFirebaseApp [signingreport]: succes 4 sec, 648 ms
Variant: debugAndroidTest
Config: debug
Store: C:\Users\Enoch\.android\debug.keystore
Alias: AndroidDebugKey
MD5: 0B:EC:E0:7C:57:66:61:E3:0D:B3:64:5A:41:17:BE:8E
SHA1: 9D:EC:5C:42:AD:63:14:01:A8:8B:68:D3:1C:8A:C5:C8:F1:CC:38:3C
SHA-256: A0:25:B6:52:B9:06:E9:8B:50:BE:6B:40:E9:1A:B0:34:40:14:75:5B:CE:37:BC:5B:8E:F
Valid until: Wednesday, January 8, 2053
---
BUILD SUCCESSFUL in 3s
1 actionable task: 1 executed
Build Analyzer results available
7:25:54 PM: Execution finished 'signingreport'.
Version Control Run TODO Problems Terminal App Inspection Logcat App Quality Insights Services Build Profiler Layout Inspector
Gradle build finished in 4 s 69 ms (2 minutes ago)
26:66 (59 chars) LF UTF-8 4 spaces
Project
Resource Manager
Structure
Bookmarks
Build Variants
Device Manager
Gradle
Notifications
Emulator
Device File Explorer

12) Back in the Firebase console, click **Register app** to go to the next step.

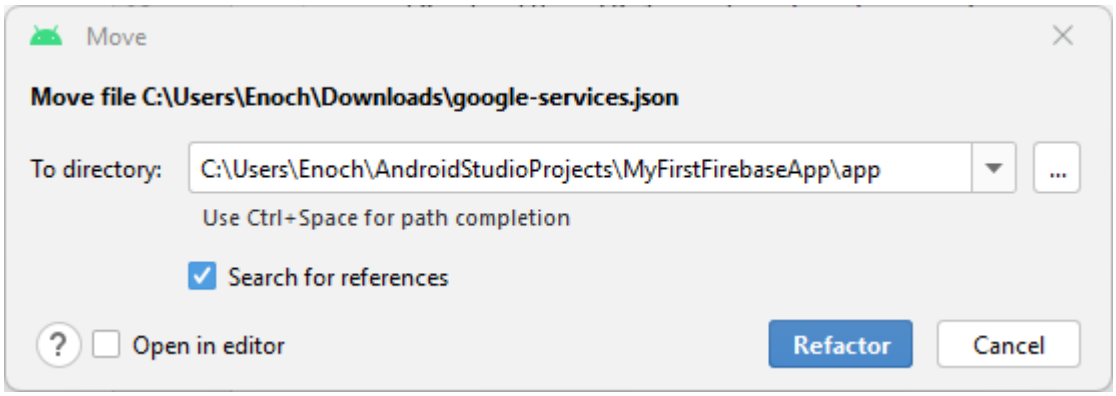
13) In Step 2 **Download and then add config file**, click on the **Download google-services.json** button to get the file.



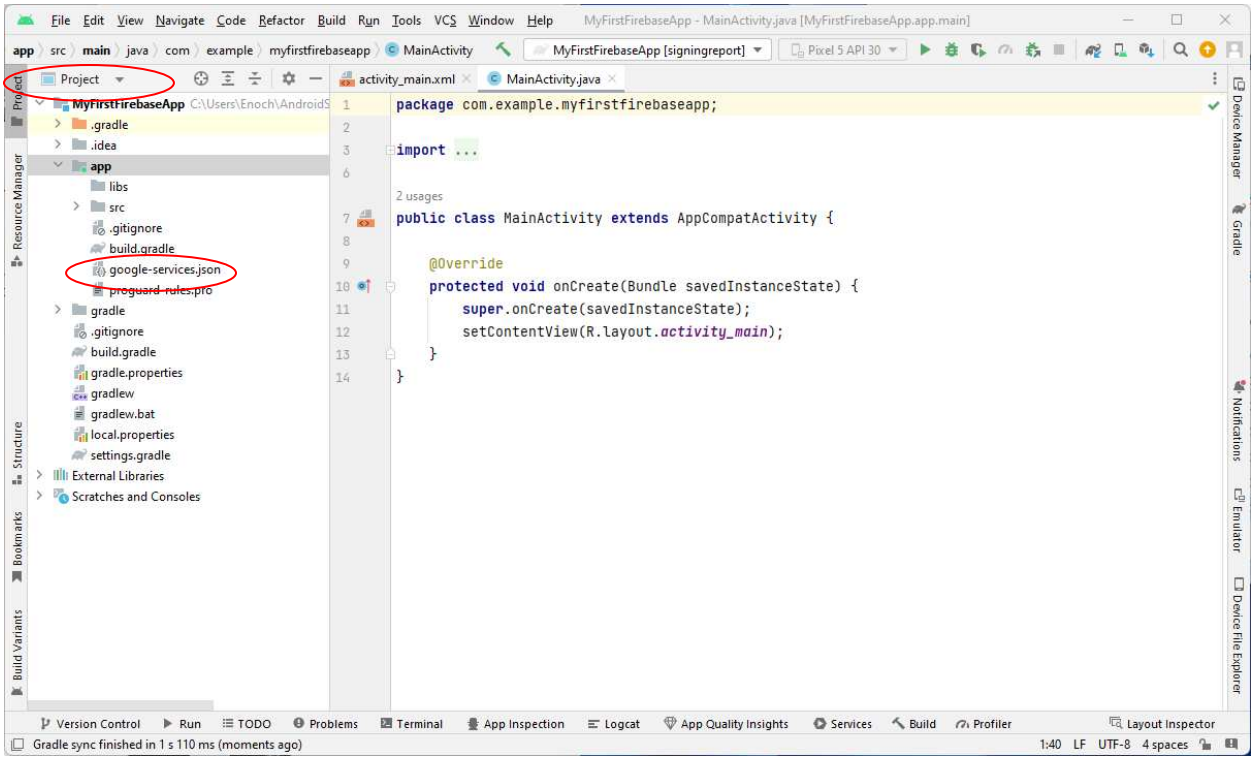
14) Back in Android Studio, drag the file **google-services.json** that you have downloaded into the **app** folder in your project.



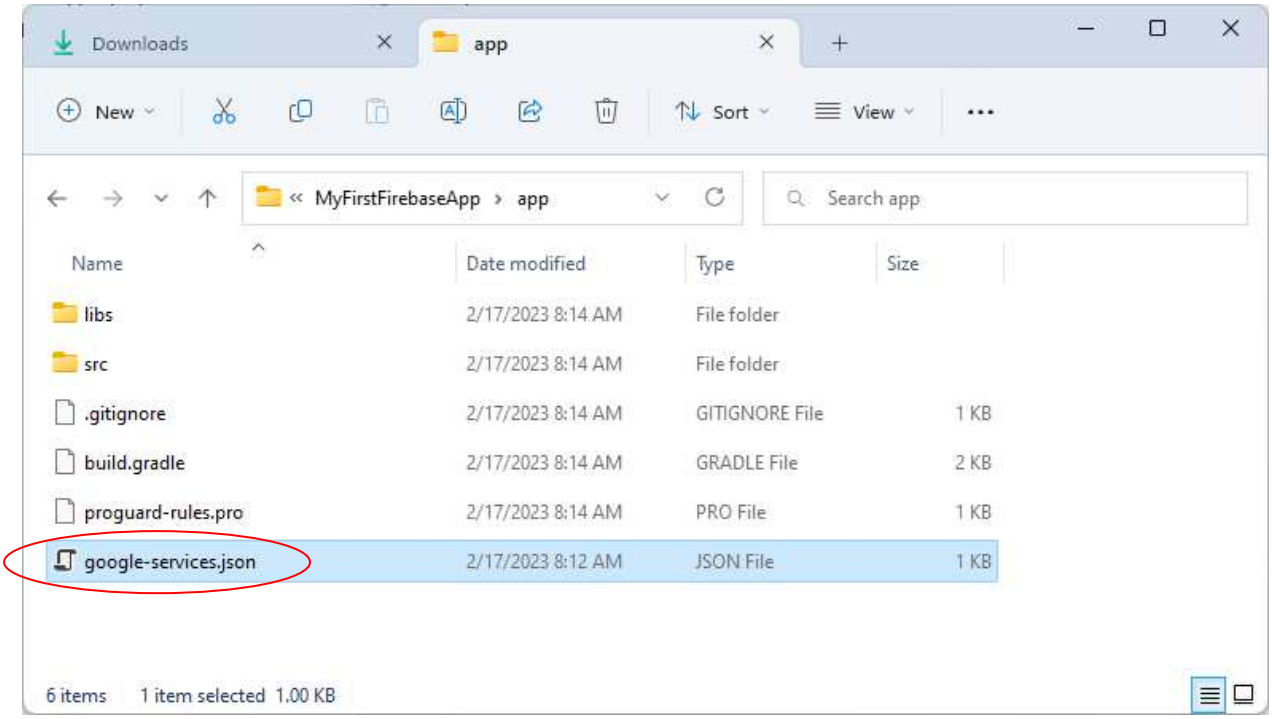

15) Click the **Refactor** button.



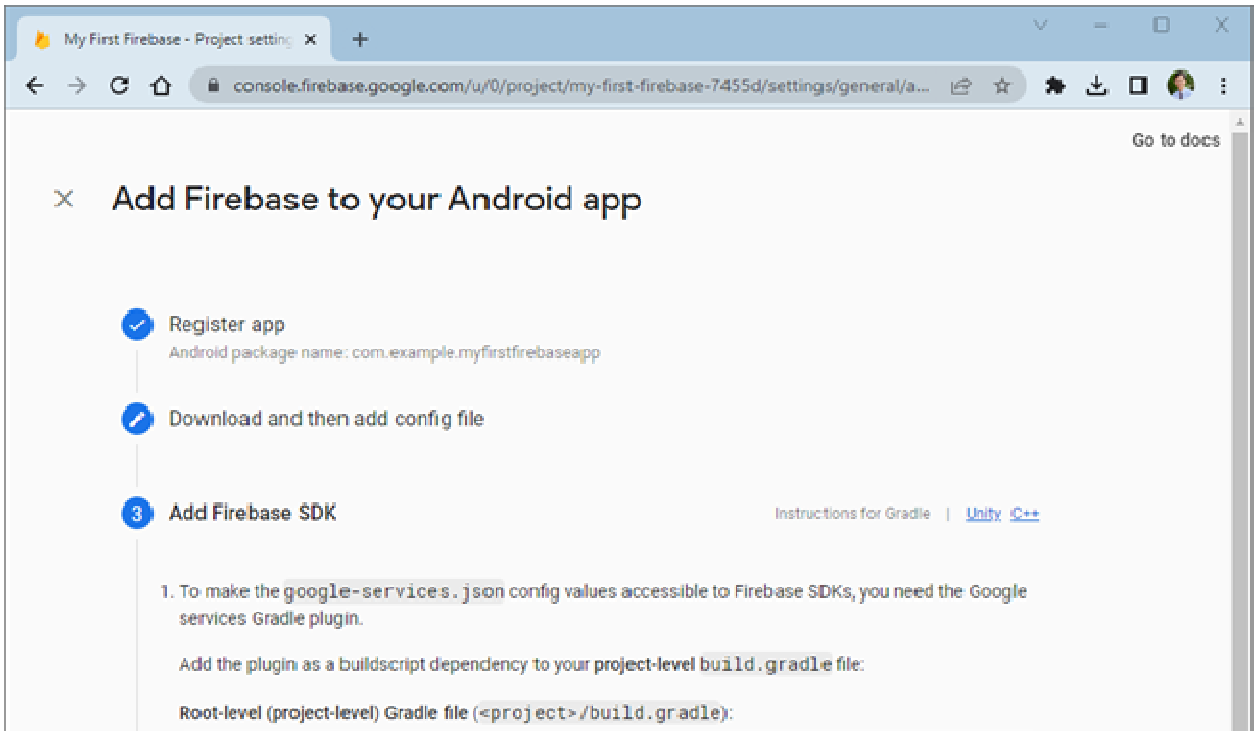
You should see the file **google-services.json** in your project's app directory.




You can also see it here in the File Manager.



16) Back in the Firebase console, click **Next** to go to Step 3 and follow the instructions to **Add Firebase SDK** to your Android project.



17) For Step 3 Point 1, copy the three lines: buildscript, dependencies, classpath **'com.google.gms:google-services:4.3.15'** and paste it into the **build.gradle (Project)** file before the **plugins** section as shown below.

~~You need to change the line to follow the format as in the id lines like this~~

~~id 'com.google.gms.google-services' version '4.3.15' apply false~~

```
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
  }
}
```

```
buildscript {
    dependencies {
        classpath 'com.google.gms:google-services:4.3.15'
    }
}
// Top-level build file where you can add configuration options common to
all sub-projects/modules.
plugins {
    id 'com.android.application' version '7.4.1' apply false
    id 'com.android.library' version '7.4.1' apply false
//    id 'com.google.gms.google-services' version '4.3.15' apply false
}
```

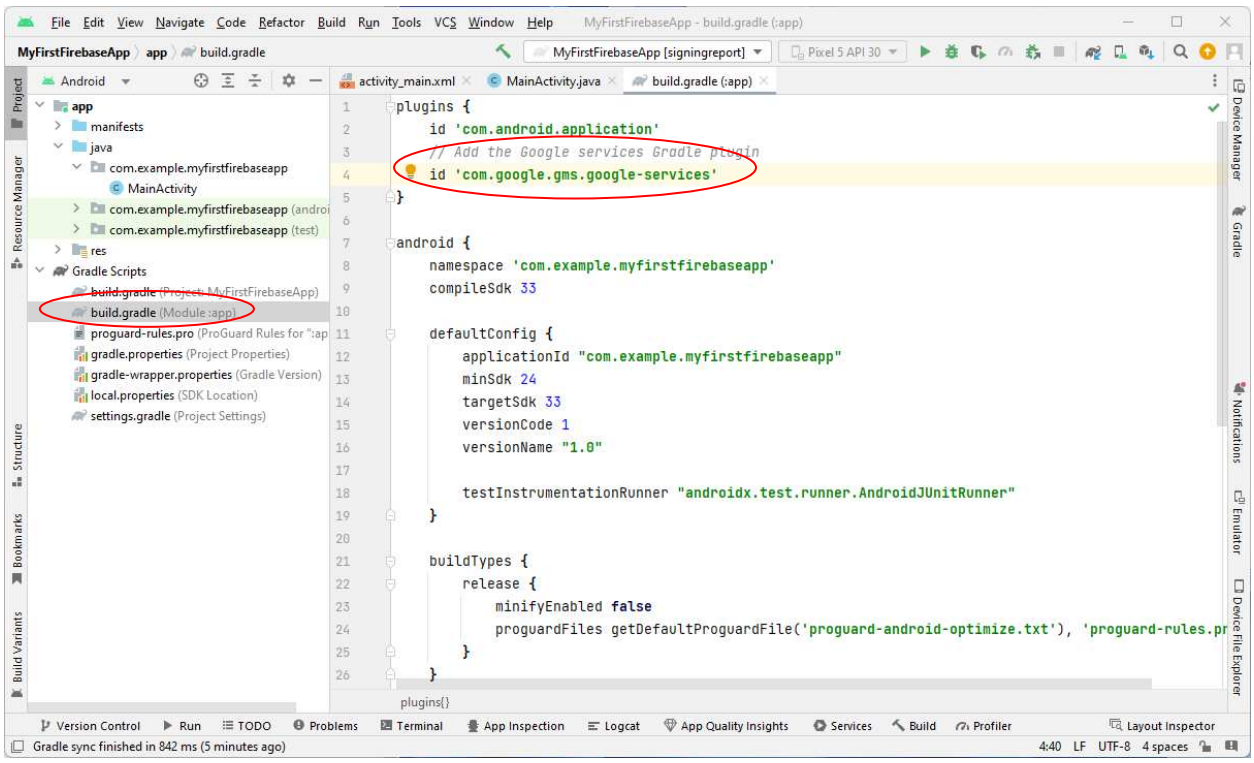
18) For Step 3 Point 2, copy the id **`'com.google.gms.google-services'`** and paste it into the **build.gradle (Module:app)** file under the **plugins** section.

2. 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:

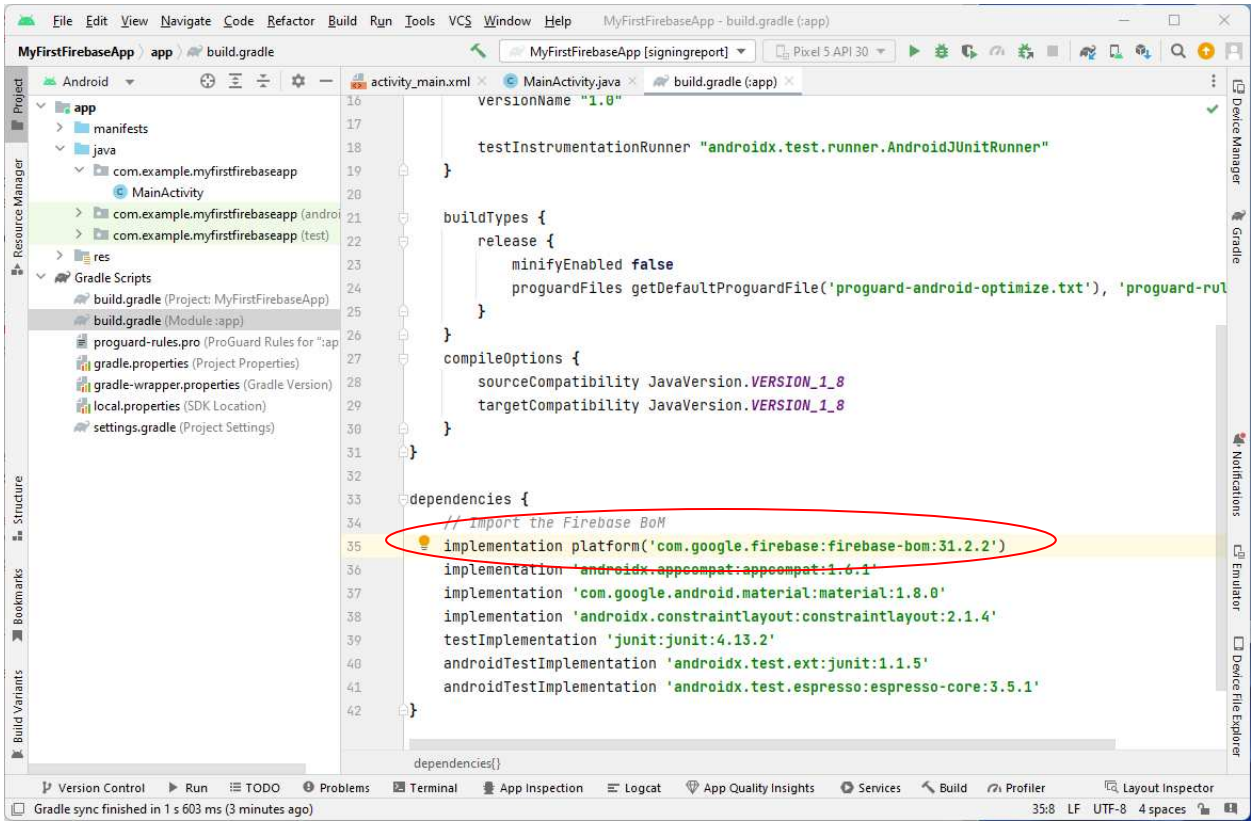
**Module (app-level) Gradle file** (`<project>/<app-module>/build.gradle`):

```
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:31.2.2')

  // TODO: Add the dependencies for Firebase products you want to use
  // When using the BoM, don't specify versions in Firebase dependencies
  // https://firebase.google.com/docs/android/setup#available-libraries
}
```

By using the Firebase Android BoM, your app will always use compatible Firebase library versions. Learn more



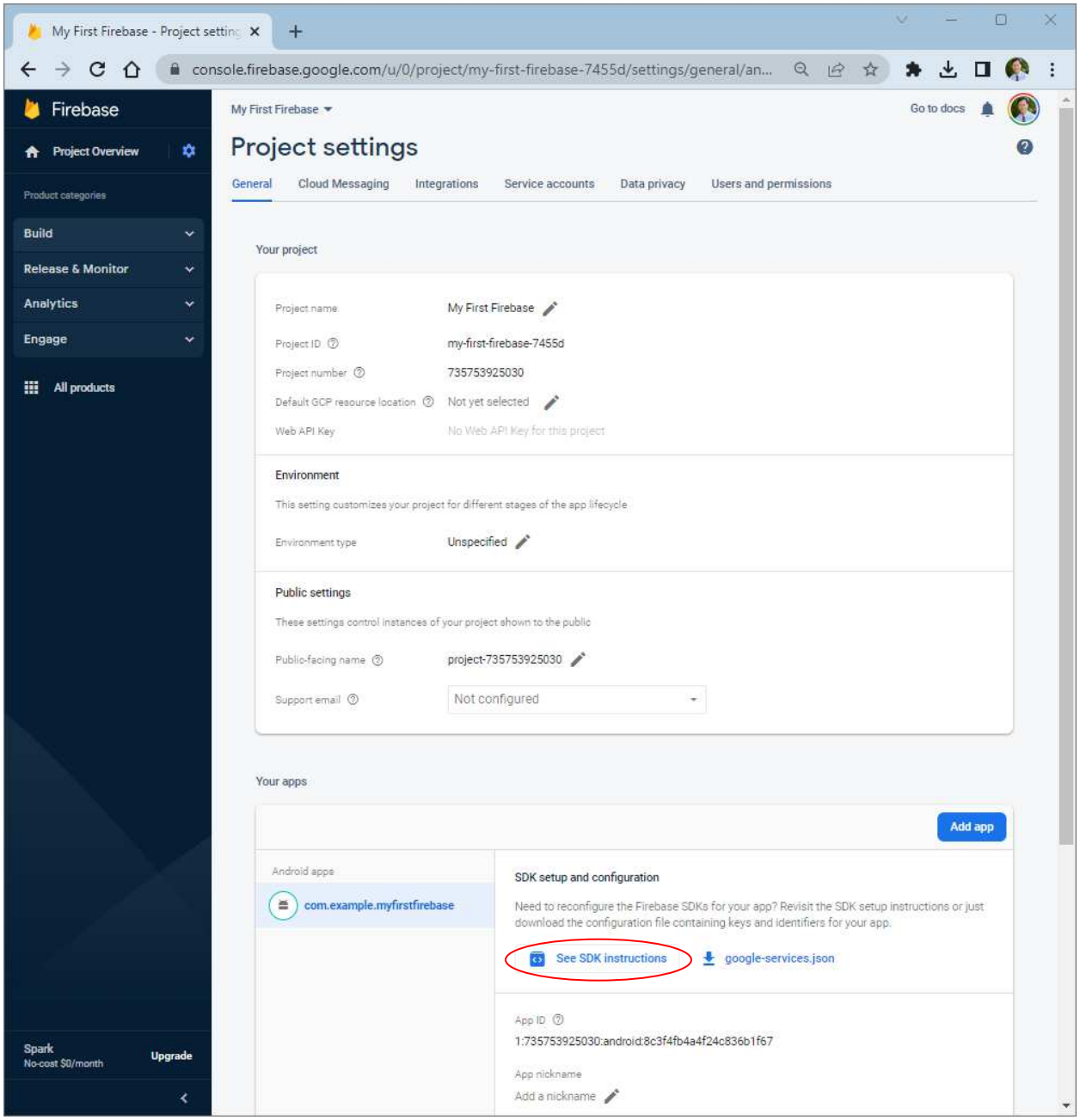
Also copy the implementation platform(`'com.google.firebase:firebase-bom:31.2.2'`) and paste it into **build.gradle (Module:app)** file under the **dependencies** section.




19) For Step 3 Point 3, after adding the above plugin and desired SDKs, click **Sync Now** to sync your Android project with Gradle files. If the sync is successful then you can click on **Next** and you're done.

20) Step 4, click on **Continue to console** to return to the Project console.

21) In the Firebase **Project settings** page you will see your app listed. You can click on the **See SDK instructions** button to go through the steps again or click on the **download google-services.json** file.



## *Android Studio project*

22) build.gradle (Project) file.

Add the **buildscript** section.

```
buildscript {
    dependencies {
        classpath 'com.google.gms:google-services:4.3.15'
    }
}
// Top-level build file where you can add configuration options common to
all sub-projects/modules.
plugins {
    id 'com.android.application' version '7.4.1' apply false
    id 'com.android.library' version '7.4.1' apply false
}
```

23) build.gradle (App) file.

Add the following three lines:

in plugins

```
id 'com.google.gms.google-services'
```

in dependencies:

```
implementation platform('com.google.firebase:firebase-
bom:31.2.2')

implementation 'com.google.firebase:firebase-messaging:23.1.1'
```

```
plugins {
    id 'com.android.application'
    id 'com.google.gms.google-services'
}

android {
    namespace 'com.example.myfirstfirebaseapp'
    compileSdk 33

    defaultConfig {
        applicationId "com.example.myfirstfirebaseapp"
        minSdk 24
        targetSdk 33
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner
```

```
"androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-
optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {
    implementation platform('com.google.firebase:firebase-
bom:31.2.2')
    implementation 'androidx.appcompat:appcompat:1.6.1'
    implementation 'com.google.android.material:material:1.8.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    implementation 'com.google.firebase:firebase-
messaging:23.1.1'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
}
```

24) AndroidManifest.xml

Add the **<service** section

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyFirstFirebaseApp"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
```

```
        <service
            android:name=".MyFirebaseMessagingService"
            android:stopWithTask="false"
            android:exported="true">
            <intent-filter>
                <action
android:name="com.google.firebase.MESSAGING_EVENT" />
            </intent-filter>
        </service>

    </application>

</manifest>
```

25) MyFirebaseMessagingService.java

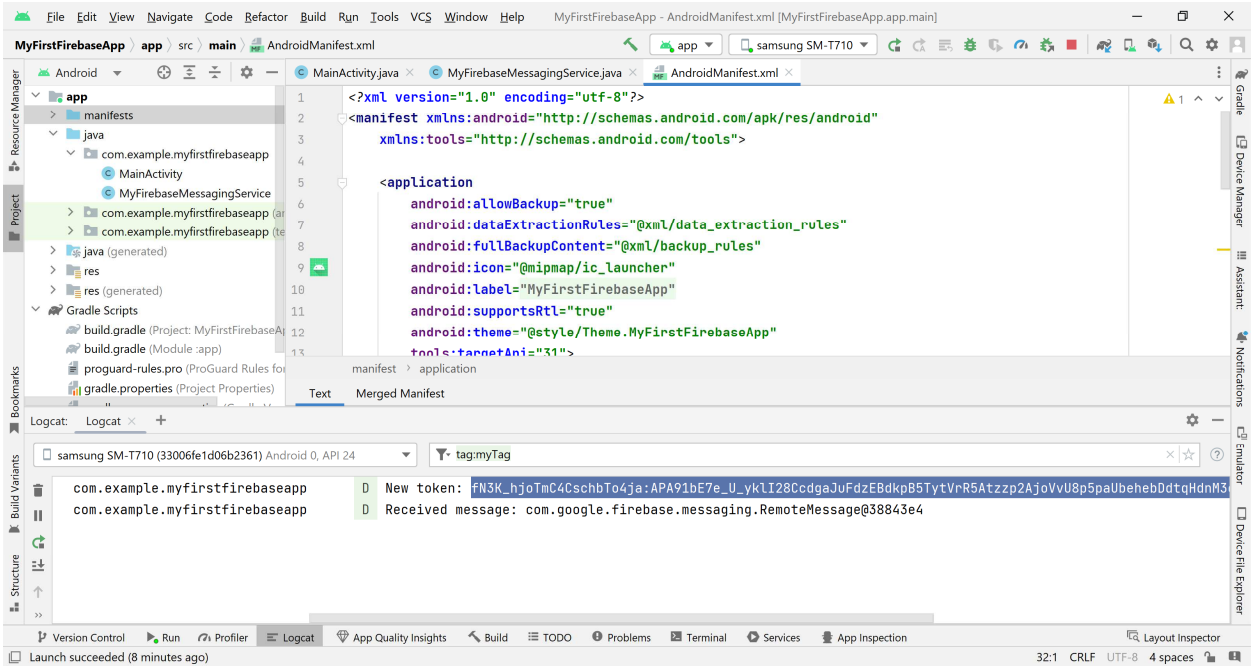
Add a new class java file named **MyFirebaseMessagingService** with the following code.

```
package com.example.myfirstfirebaseapp;

import android.util.Log;
import androidx.annotation.NonNull;
import com.google.firebase.messaging.FirebaseMessagingService;
import com.google.firebase.messaging.RemoteMessage;
import java.util.Objects;

public class MyFirebaseMessagingService extends FirebaseMessagingService {

    // this method is called when app is first ran after install
    @Override
    public void onNewToken(@NonNull String token) {
        Log.d("MyTag", "New token: "+token);

    }

    // this method is called when app is in the foreground when a
notification arrives
    @Override
    public void onMessageReceived(@NonNull RemoteMessage remoteMessage) {
        Log.d("MyTag", "Received message \n  Title: "+
Objects.requireNonNull(remoteMessage.getNotification()).getTitle()+"\n
Body: "+remoteMessage.getNotification().getBody());
    }

}
```

26) MainActivity.java

```
package com.example.myfirstfirebaseapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        FirebaseMessaging.getInstance().subscribeToTopic("alarms"); //
subscribe to topic "alarms"
    }
}
```

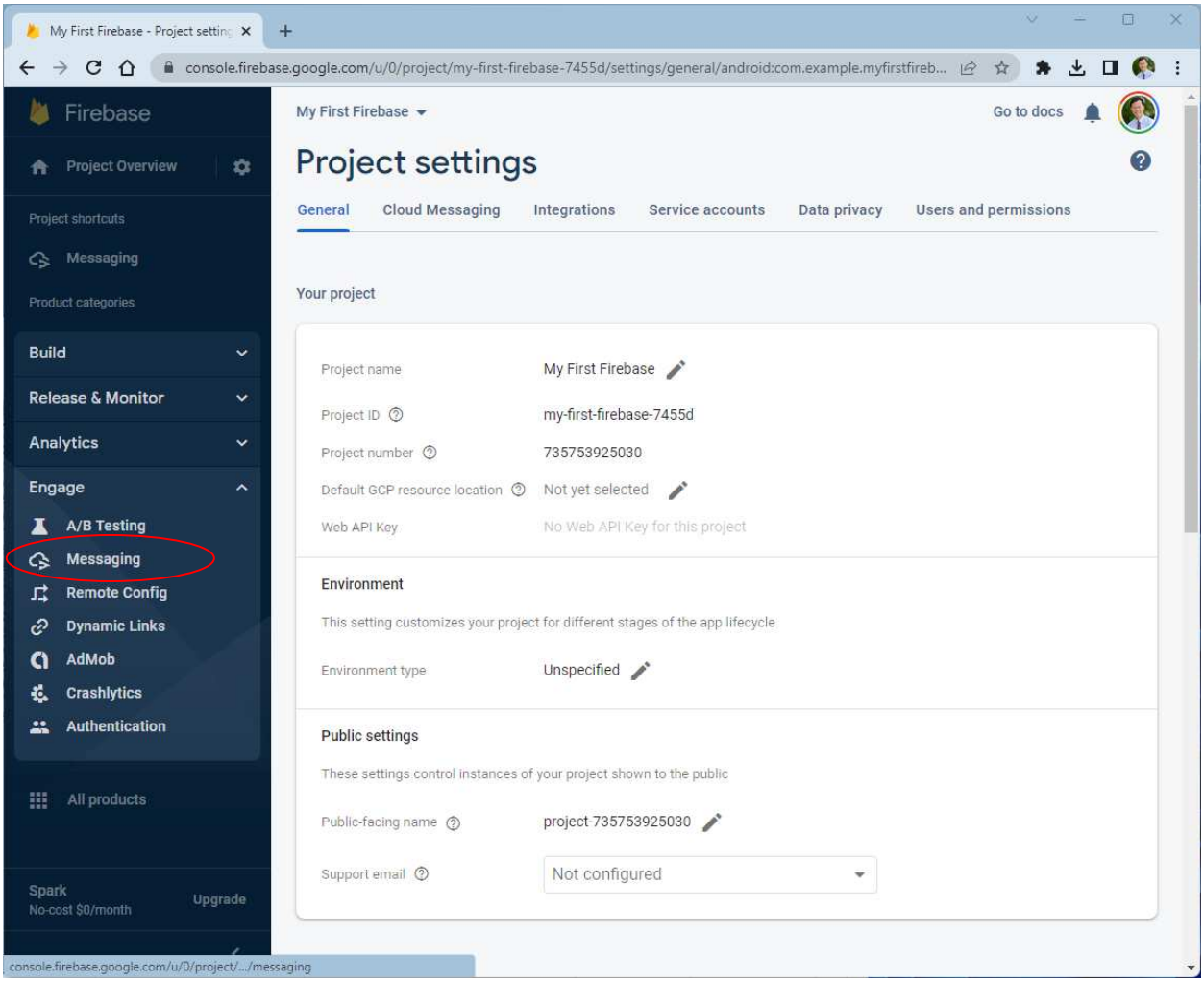
27) Run the app on a device and look at the log. It will print out the new token. Copy the entire token string. It is very long.



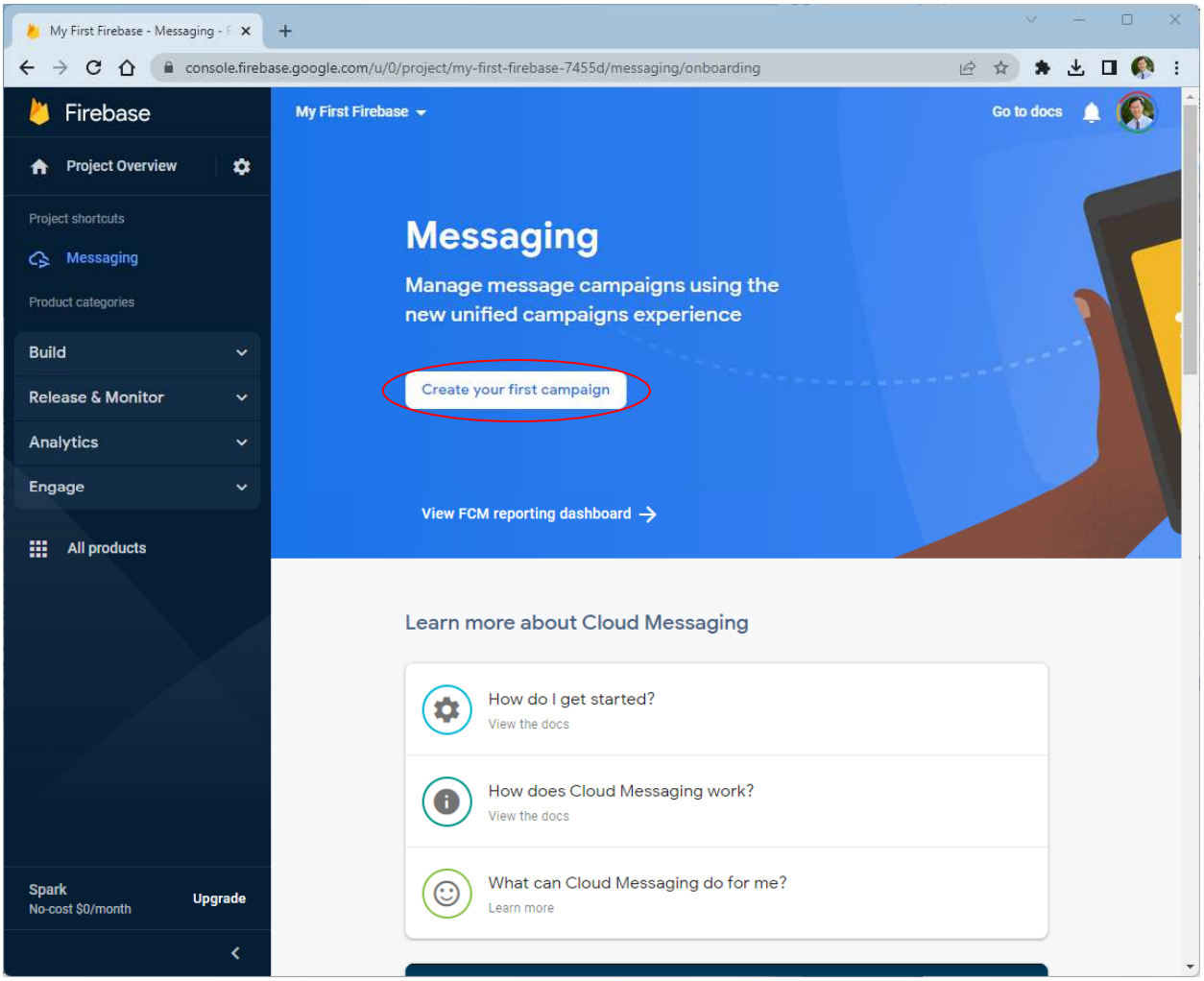

28) Put the app to the background. THIS IS VERY IMPORTANT!

## *Firebase Console*

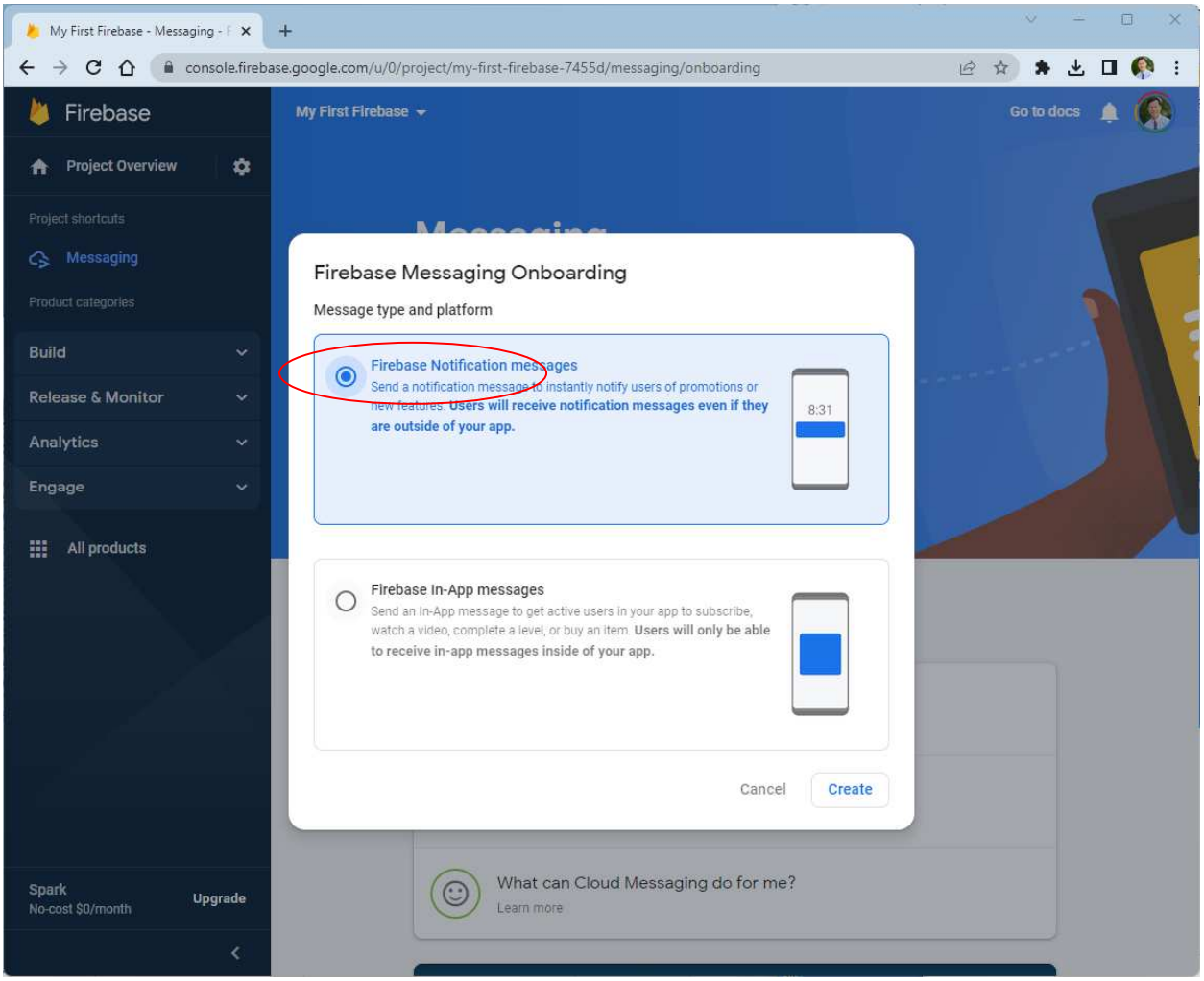
29) Make sure that you are in the correct project. Click on **Messaging** on the sidebar.



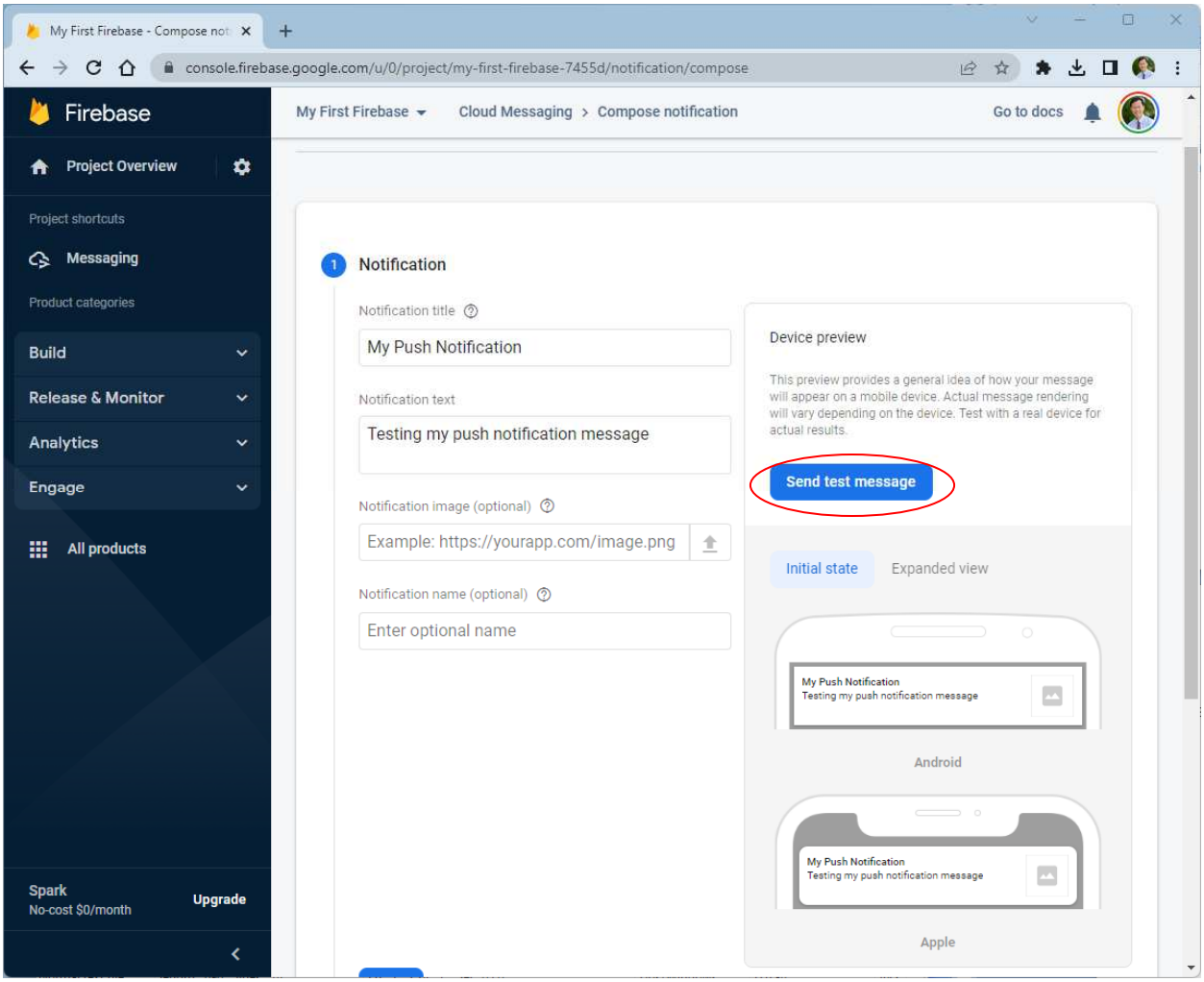
30) Click on **Create your first campaign**.



31) Select **Firebase Notification messages**, and click **Create**.



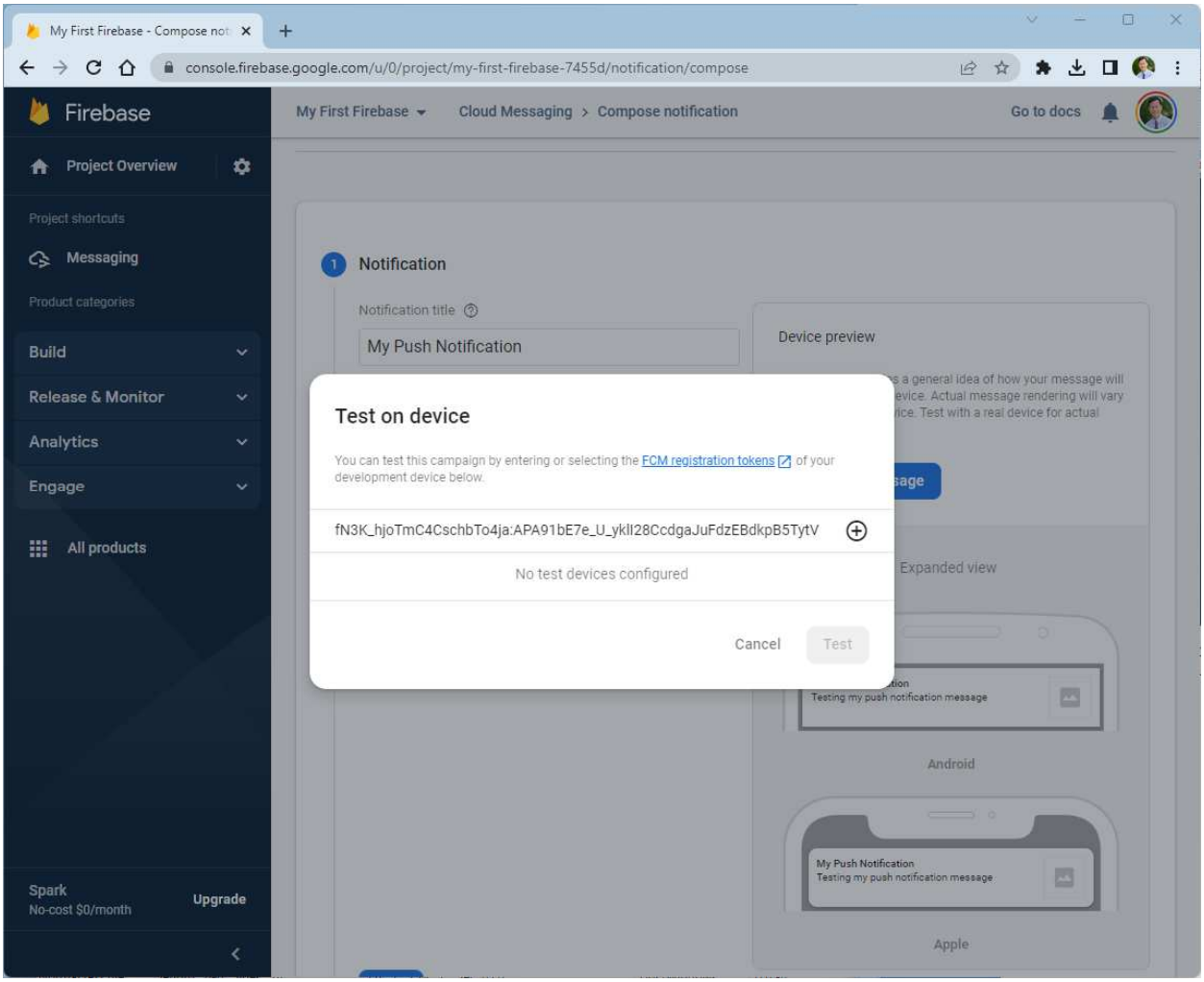
32) Type in the notification title and text, then click the **Send test message** button.

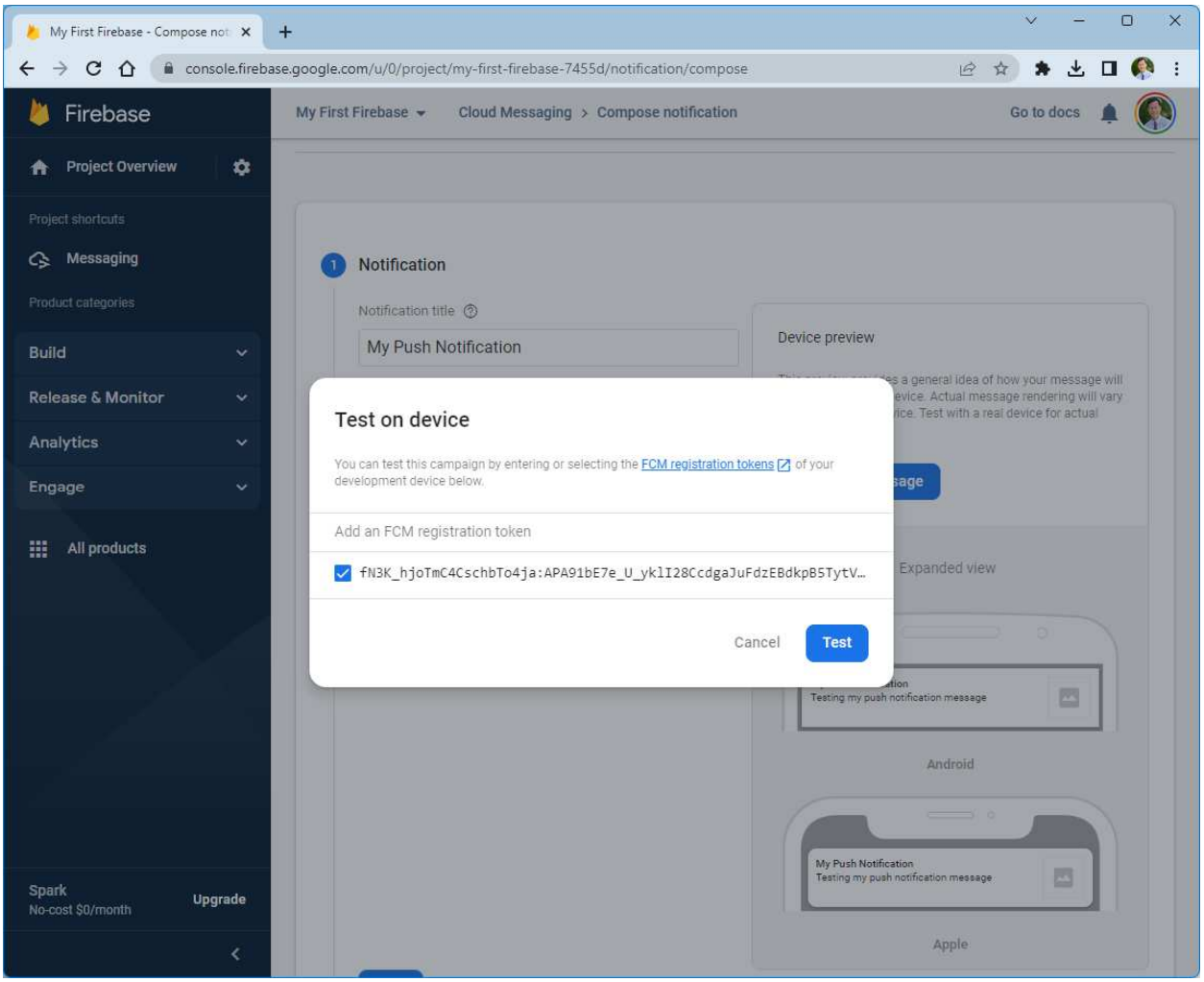


33) In the **Test on device** popup window, paste the token string that you copied from the Android Studio run log in step 26).

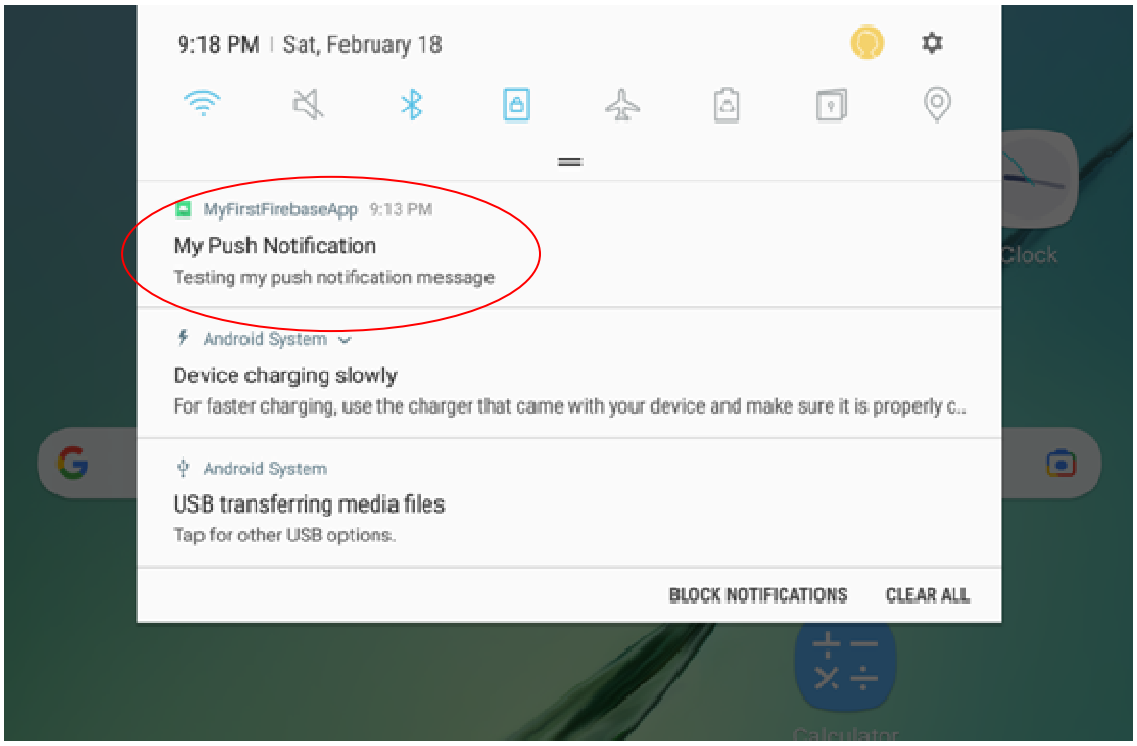
Click the **+** sign.

Click the **Test** button.

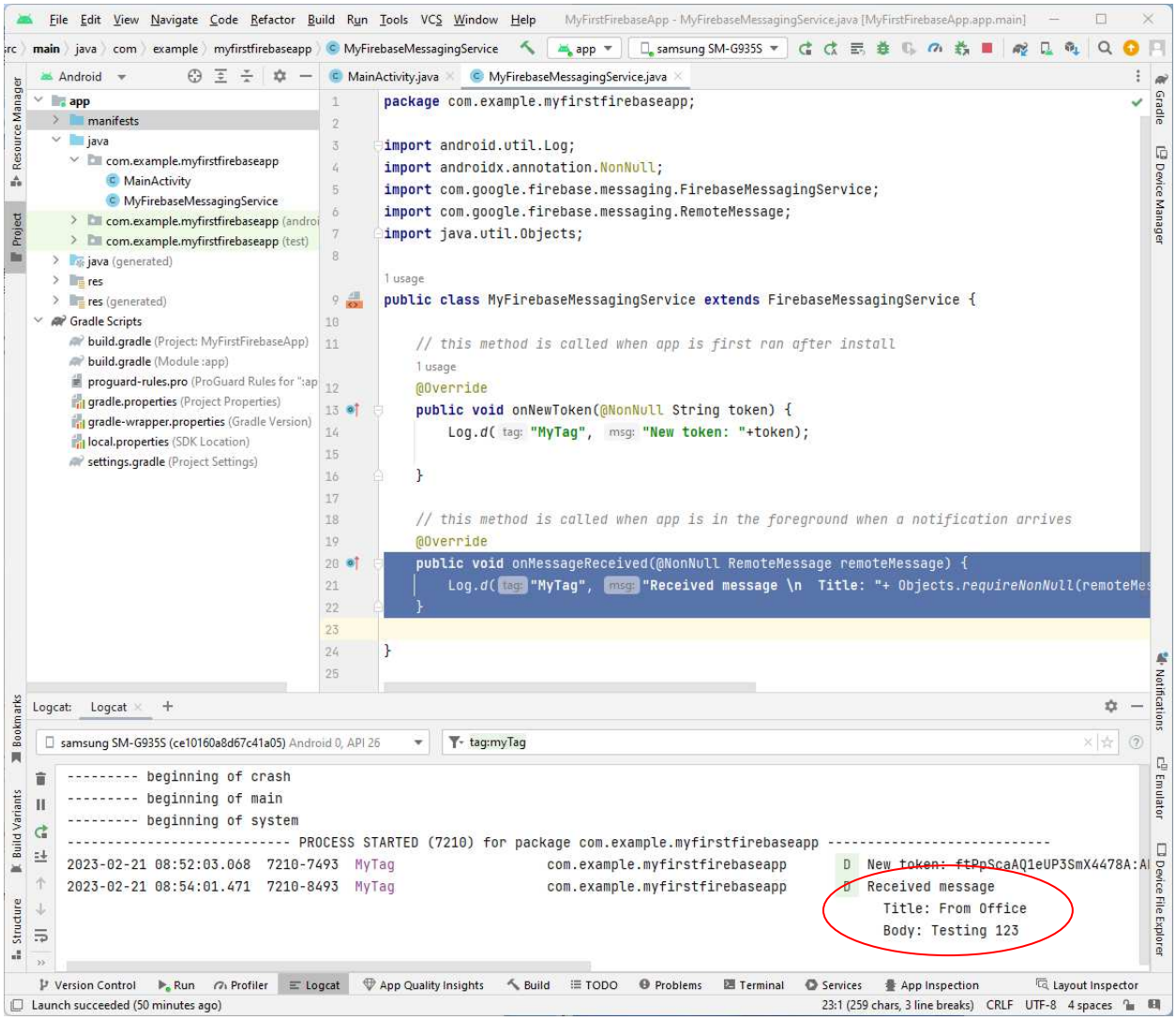



My First Firebase - Compose not
console.firebase.google.com/u/0/project/my-first-firebase-7455d/notification/compose
Firebase
My First Firebase
Cloud Messaging > Compose notification
Go to docs
Project Overview
Project shortcuts
Messaging
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Notification
Notification title
My Push Notification
Device preview
Test on device
You can test this campaign by entering or selecting the FCM registration tokens of your development device below.
Add an FCM registration token
fN3K_hjoTmC4CschbTo4ja:APA91bE7e_U_yklI28CcdgaJuFdzEBdkpB5TytV…
Cancel
Test
Expanded view
Testing my push notification message
Android
My Push Notification
Testing my push notification message
Apple

34) After a few seconds you should see the notification message on your device. Your app MUST be in the background to get this notification.



If your app is running in the foreground and your device is connected to Android Studio then you will see the log message from the onMessageReceived method.

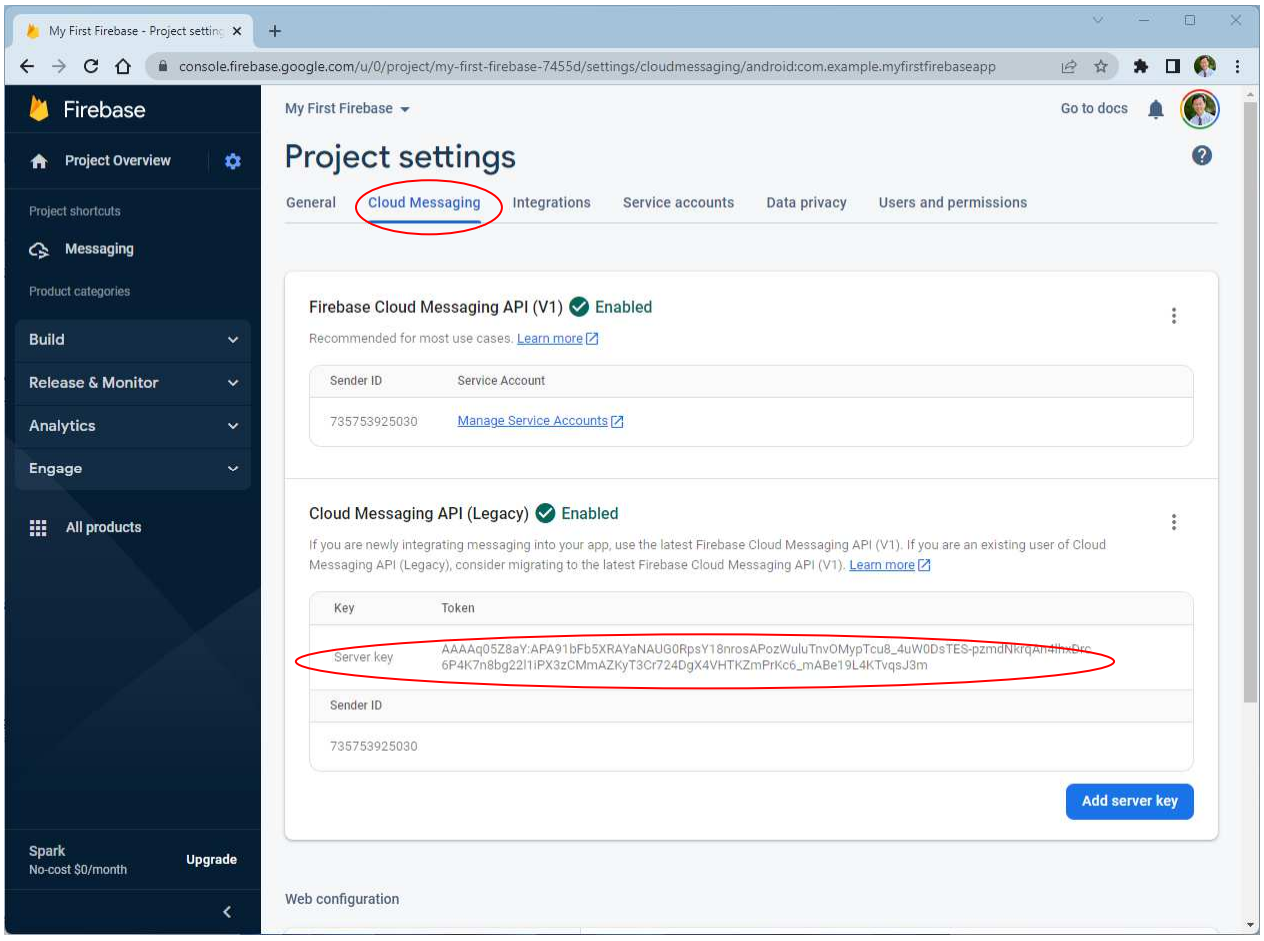



35) From Step 32) if instead of doing the **Send test message** but continue with Points 2 and 3, then it will take about a minute for the notification to arrive.

## *Sending Push Notification from PHP*

https://www.w3schools.in/php/examples/send-push-notification-using-fcm

36) Get the server key from **Firebase Console -> Project Settings -> Cloud Messaging -> Server key**




37) Get the token from running the client app on the mobile device from Step 26). This is shown only from running the app for the first time after installation.

38) Create the following **push_notification.php** file. Change the $server_key and $token to match your setup. Don't need the $token if using topics.

and upload it to your Ubuntu server.

```
<?php
// Reference: https://firebase.google.com/docs/cloud-messaging/http-server-ref
```

```
// https://firebase.google.com/docs/cloud-messaging/js/first-
message#send_a_notification_message
// https://www.w3schools.in/php/examples/send-push-notification-using-fcm


//grab information from HTML
$title = $_POST['title'];
$message = $_POST['message'];
//$title = $_GET['title'];
//$message = $_GET['message'];


// Get the server key from the Firebase Console -> Project Settings -> CLOUD
MESSAGING -> Server key
// https://console.firebase.google.com/u/0/project/iotnotification-
c6ffe/settings/cloudmessaging/android:com.mygadgets2.iotalarm

// My server key for MyFirstFirebase project on Firebase
$server_key =
"AAAAq05Z8aY:APA91bFb5XRAYaNAUG0RpsY18nrosAPozWuluTnvOMypTcu8_4uW0DsTES-
pzmdNkrqAn4lhxDrc6P4K7n8bg22l1iPX3zCMmAZKyT3Cr724DgX4VHTKZmPrKc6_mABe19L4KTvqsJ3m";

// Do not need to use tokens anymore 2023/03/05
// Using subscription to topics instead// unique token for mobile device
// change this for the mobile device you want to send message to
// get this from running the app on the mobile device

$token =
"fN3K_hjoTmC4CschbTo4ja:APA91bE7e_U_yklI28CcdgaJuFdzEBdkpB5TytVrR5Atzzp2AjoVvU8p5paU
behebDdtqHdnM3ougFiW9yA8cIQyEgyoNNBfb5uEGqcpOYrtsiRMvDxt-8Gq-VitCt26JMGLNld4F4vh";


sendPushNotification($title, $message, $token, $server_key);


////////////////////////////////////////////////////////
function sendPushNotification($title, $message, $token, $server_key) {

      echo "Sending\r\n<br>";
      echo " Title      = ".$title . "\r\n<br>";
      echo " Message    = ".$message . "\r\n<br>";
      echo " Token      = ".$token . "\r\n<br>";
      echo " Server key = ".$server_key . "\r\n<br>";


      //Firebase HTTP API URL
      $path_to_fcm = 'https://fcm.googleapis.com/fcm/send';


      $fields = array(
            // 'to' => $token,  // send to one device with this token id
            'to' = > '/topics/alarms', // send to all subscribed to the topic
"alarms"
            // 'notification' => array ('title'=>$title,'body'=>$message),   // for
sending notification payload
            'data' = > array('title' = > $title, 'body' = > $message),
      // for sending data payload
```

```
            'android_channel_id' = > 'Alarm default channel',         // need to
match in pushNotification method in MyFirebaseMessagingService.java file
            'priority' = > 'high'
            );

      $headers = array(
            'Authorization:key='.$server_key,
            'Content-Type:application/json'
            );

      $payload = json_encode($fields);
      $curl_session = curl_init();
      // need to install php-curl package if you do not see the following echo
message
      echo "after curl init\r\n<br>";
      curl_setopt($curl_session, CURLOPT_URL, $path_to_fcm);
      curl_setopt($curl_session, CURLOPT_POST, true);
      curl_setopt($curl_session, CURLOPT_HTTPHEADER, $headers);
      curl_setopt($curl_session, CURLOPT_RETURNTRANSFER, true);
      curl_setopt($curl_session, CURLOPT_SSL_VERIFYHOST, 0);
      curl_setopt($curl_session, CURLOPT_SSL_VERIFYPEER, false);
      curl_setopt($curl_session, CURLOPT_IPRESOLVE, CURL_IPRESOLVE_V4);
      curl_setopt($curl_session, CURLOPT_POSTFIELDS, $payload);

      $result = curl_exec($curl_session);
      if ($result == = FALSE) {
            die('FCM Send Error: '.curl_error($curl_session));
      }
      curl_close($curl_session);
      echo "\r\n<br> curl send successful \r\n<br>";
}     // end of function

?>
```

39) If you don't see the "after curl init" message from the echo then you need to install the php-curl package.

```
$curl_session = curl_init();
echo "after curl init";
```

To install php-curl. The version number in php7.4-curl might be different.

```
sudo apt update
sudo apt install php7.4-curl
sudo phpenmod curl
sudo systemctl restart apache2.service
```

40) Here's the webpage to test the push_notificaiton.php send.

```
<!DOCTYPE html>
<html>
<body>

<h2>Send Notifications</h2>
```

```
<p>Clicking on submit will execute the push_notification.php file
<br>
<form action="push_notification.php"  method="POST">
  Title:<br>
  <input type="text" name="title">
  <br>
  Message:<br>
  <input type="text" name="message">
  <br><br>
  <input type="submit" value="Submit">
</form>


</body>
</html>
```

41) After a few seconds you should see the notification message on your device. Your app MUST be in the background to get this notification.

## *Advanced Complete Receiver Code*

This push notification receiver code will receive the notification when the app is in background or forground. The Firebase payload in the push_notification.php code must send a data message and not a notification message for this to work.

42) Push_notification.php

```
<?php
// Reference: https://firebase.google.com/docs/cloud-messaging/http-server-ref
// https://firebase.google.com/docs/cloud-messaging/js/first-
message#send_a_notification_message
// https://www.w3schools.in/php/examples/send-push-notification-using-fcm


//grab information from HTML
$title = $_POST['title'];
$message = $_POST['message'];
//$title = $_GET['title'];
//$message = $_GET['message'];


// Get the server key from the Firebase Console -> Project Settings -> CLOUD
MESSAGING -> Server key
// https://console.firebase.google.com/u/0/project/iotnotification-
c6ffe/settings/cloudmessaging/android:com.mygadgets2.iotalarm

// My server key for MyFirstFirebase project on Firebase
$server_key =
"AAAAq05Z8aY:APA91bFb5XRAYaNAUG0RpsY18nrosAPozWuluTnvOMypTcu8_4uW0DsTES-
pzmdNkrqAn4lhxDrc6P4K7n8bg22l1iPX3zCMmAZKyT3Cr724DgX4VHTKZmPrKc6_mABe19L4KTvqsJ3m";

// Do not need to use tokens anymore 2023/03/05
// Using subscription to topics instead// unique token for mobile device
// change this for the mobile device you want to send message to
// get this from running the app on the mobile device

$token =
"fN3K_hjoTmC4CschbTo4ja:APA91bE7e_U_yklI28CcdgaJuFdzEBdkpB5TytVrR5Atzzp2AjoVvU8p5paU
behebDdtqHdnM3ougFiW9yA8cIQyEgyoNNBfb5uEGqcpOYrtsiRMvDxt-8Gq-VitCt26JMGLNld4F4vh";


sendPushNotification($title, $message, $token, $server_key);


////////////////////////////////////////////////////////
function sendPushNotification($title, $message, $token, $server_key) {

   echo "Sending\r\n<br>";
   echo " Title      = ".$title . "\r\n<br>";
   echo " Message    = ".$message . "\r\n<br>";
   echo " Token      = ".$token . "\r\n<br>";
   echo " Server key = ".$server_key . "\r\n<br>";
```

```
    //Firebase HTTP API URL
    $path_to_fcm = 'https://fcm.googleapis.com/fcm/send';


    $fields = array(
        // 'to' => $token,  // send to one device with this token id
        'to' = > '/topics/alarms', // send to all subscribed to the topic "alarms"
        // 'notification' => array ('title'=>$title,'body'=>$message),   // for
sending notification payload
        'data' = > array('title' = > $title, 'body' = > $message),             // for
sending data payload
        'android_channel_id' = > 'Alarm default channel',         // need to match in
pushNotification method in MyFirebaseMessagingService.java file
        'priority' = > 'high'
        );

    $headers = array(
        'Authorization:key='.$server_key,
        'Content-Type:application/json'
        );

    $payload = json_encode($fields);
    $curl_session = curl_init();
    // need to install php-curl package if you do not see the following echo message
    echo "after curl init\r\n<br>";
    curl_setopt($curl_session, CURLOPT_URL, $path_to_fcm);
    curl_setopt($curl_session, CURLOPT_POST, true);
    curl_setopt($curl_session, CURLOPT_HTTPHEADER, $headers);
    curl_setopt($curl_session, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($curl_session, CURLOPT_SSL_VERIFYHOST, 0);
    curl_setopt($curl_session, CURLOPT_SSL_VERIFYPEER, false);
    curl_setopt($curl_session, CURLOPT_IPRESOLVE, CURL_IPRESOLVE_V4);
    curl_setopt($curl_session, CURLOPT_POSTFIELDS, $payload);

    $result = curl_exec($curl_session);
    if ($result == = FALSE) {
        die('FCM Send Error: '.curl_error($curl_session));
    }
    curl_close($curl_session);
    echo "\r\n<br> curl send successful \r\n<br>";
}   // end of function

?>
```

43) MainActivity.java

```
package com.example.myfirstfirebaseapp;

import androidx.appcompat.app.AppCompatActivity;

import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    private String token, timeStamp, title, body;
    TextView token_tv, timeStamp_tv, title_tv, body_tv;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        FirebaseMessaging.getInstance().subscribeToTopic("alarms"); //
subscribe to topic "alarms"
    }

    @Override
    protected void onResume() {
        super.onResume();
        token_tv = findViewById(R.id.token);
        timeStamp_tv = findViewById(R.id.timeStamp);
        title_tv = findViewById(R.id.title);
        body_tv = findViewById(R.id.body);
        retrieveData();
        token_tv.setText(token);
        timeStamp_tv.setText(timeStamp);
        title_tv.setText(title);
        body_tv.setText(body);
    }

    // retrieve data from Shared Preferences
    private void retrieveData() {
        // read from shared preferences
        SharedPreferences sharedPreferences =
getSharedPreferences("MySharedPref", MODE_PRIVATE);
        // token = getSharedPreferences("MyData",
MODE_PRIVATE).getString("token", "empty");
        token = sharedPreferences.getString("token", "empty");
        timeStamp = sharedPreferences.getString("timeStamp", "");
        title = sharedPreferences.getString("title", "");
        body = sharedPreferences.getString("body", "");
    }
}
```

44) MyFirebaseMessagingService.java

```
package com.example.myfirstfirebaseapp;

import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.media.RingtoneManager;
import android.net.Uri;
import android.os.Build;
import android.util.Log;
import androidx.annotation.NonNull;
import androidx.core.app.NotificationCompat;

import com.google.firebase.messaging.FirebaseMessagingService;
import com.google.firebase.messaging.RemoteMessage;
```

```
import java.text.SimpleDateFormat;
import java.util.Date;

public class MyFirebaseMessagingService extends FirebaseMessagingService {

    // this method is called when a new FCM registration token is generated
    // a new token is generated when the app is first installed
    // this unique token is needed in the push_notification.php code to send
push notifications to the app

    @Override
    public void onNewToken(@NonNull String token) {
        Log.d("MyTag", "New token: "+token);
        getSharedPreferences("MyData",
MODE_PRIVATE).edit().putString("token", token).apply();  // optional save
token to shared preferences
        sendRegistrationTokenToServer(token);   // optional save token on
your app server
    }

    // There are two types of messages, data messages and notification
messages.
    // For Data messages (using the key 'data' in the payload) this
onMessageReceived method is called whether the app is in the foreground or
background.
    // For Notification messages (using the key 'notification' in the
payload) this onMessageReceived method is called
    // only when the app is in the foreground. When the app is in the
background an automatically generated notification is displayed.
    // Messages containing both notification and data payloads are treated as
notification messages.
    // When the user taps on the notification they are returned to the app.
    // The Firebase console always sends notification messages.
    // For more see: https://firebase.google.com/docs/cloud-
messaging/concept-options


    @Override
    public void onMessageReceived(@NonNull RemoteMessage remoteMessage) {
        String title = "", body = "";

        // check if message contains a data payload
        if (remoteMessage.getData().size() >0) {
            title = remoteMessage.getData().get("title");
            body = remoteMessage.getData().get("body");
            Log.d("MyTag", " Data payload: Title: " + title + " Body: " +
body);
            showNotification(title, body);
        }

        // check if message contains a notification payload
        if (remoteMessage.getNotification() != null) {
            title = remoteMessage.getNotification().getTitle();
            body = remoteMessage.getNotification().getBody();
            Log.d("MyTag", " Notification payload: Title: "+ title+" Body: "
+body);
```

```
            showNotification(title, body);
        }

        String timeStamp = new SimpleDateFormat("HH:mm:ss a EEE MM-dd-
yyyy").format(new Date());  // get timestamp
        timeStamp = timeStamp.replace("AM", "am").replace("PM", "pm");  //
change to lower case

        // optional save message to shared preferences
        SharedPreferences sharedPreferences =
getSharedPreferences("MySharedPref", MODE_PRIVATE);
        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.putString("timeStamp", timeStamp);
        editor.putString("title", title);
        editor.putString("body", body);
        editor.apply();
    }

    // create and show push notification on device
    private void showNotification(String messageTitle, String messageBody) {
        Intent intent = new Intent(this, MainActivity.class);
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0,
intent, PendingIntent.FLAG_IMMUTABLE);

        String channelId = "my_notification_id";
        Uri defaultSoundUri =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
        NotificationCompat.Builder notificationBuilder = new
NotificationCompat.Builder(this, channelId)
                .setSmallIcon(R.drawable.notification_ic)
                .setContentTitle(messageTitle)
                .setContentText(messageBody)
                .setAutoCancel(true)
                .setSound(defaultSoundUri)
                .setColor(1234)
                .setContentIntent(pendingIntent);

        NotificationManager notificationManager =
                (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            // need to match the channelId in the push_notification.php $field
payload
            NotificationChannel channel = new NotificationChannel(channelId,
"Alarm default channel", NotificationManager.IMPORTANCE_DEFAULT);
            notificationManager.createNotificationChannel(channel);
        }

        notificationManager.notify(0, notificationBuilder.build());
    }

    // optional send the FCM registration token to your app server
    // to manage this apps tokens on the server side
    // the push_notification.php code can then get the token from the mySQL
database
```

```
    private void sendRegistrationTokenToServer(String token) {
        // the Firebase token format is
        //
fN3K_hjoTmC4CschbTo4ja:APA91bHl1lpDscrpTZgR_m3leoDhJmKxqZmSAuOqr5EZyFnE6pBtXJ
1PcCDHUvMncwXxYaVLoprNMF8oJKykJTD1VbD4Egq59kNN0c_SRf4wgjJj7517hytC1l7QuoyZ97O
wGsW61yoD
        // the first part up to the : is the unique device ID that does not
change for the device
        // the second part after the : is the token that changes after each
new install of app
        String[] data = {"deviceID", "token"};
        data[0] = token.substring(0,token.indexOf(":"));
        data[1] = token.substring(token.indexOf(":")+1);
        new UploadToken().execute(data);    // send token to server
    }
}
```

45) UploadToken.java

Optional. This is only needed for saving the Firebase token to a mySQL database in my server.

```
// Executes the PHP script at the given URL
//
http://hwang.lasierra.edu/~enoch/CPTG%20384%20Mobile%20App/Push%20Notificatio
n/save_token.php
// Uses POST to send the FCM registration token to the save_token.php script
// which saves the token to the mySQL database FirebaseTokensDB in the Tokens
table

package com.example.myfirstfirebaseapp;

import android.os.AsyncTask;
import android.util.Log;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

public class UploadToken extends AsyncTask<String, Void, Boolean> {

    public UploadToken() {
    }

    // execute PHP script
    @Override
    protected Boolean doInBackground(String... data) {
        String deviceID = data[0];
        String token = data[1];

        String myPostData = "deviceID=" + deviceID+ "&token=" + token;
        try {
            URL url = new
URL("http://hwang.lasierra.edu/~enoch/CPTG%20384%20Mobile%20App/Push%20Notifi
```

```
cation/save_token.php");
            HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
            // send POST data to PHP script
            connection.setRequestMethod("POST");
            connection.getOutputStream().write(myPostData.getBytes());
            // get response from PHP script
            if (connection.getResponseCode() == HttpURLConnection.HTTP_OK) {
                Log.d("MyTag", "Connection ok");
                String line;
                StringBuilder builder = new StringBuilder();
                BufferedReader reader = new BufferedReader(new
InputStreamReader(connection.getInputStream()));
                while ((line = reader.readLine()) != null) {
                    builder.append(line);
                }
                Log.d("MyTag", "PHP response: "+builder.toString());
            } else {
                Log.d("MyTag", "HTTP return code:
"+connection.getResponseCode());
            }
            return true;
        } catch (MalformedURLException e) {
            Log.d("MyTag", "Malformed URL exception: "+e);
        } catch (IOException e) {
            Log.d("MyTag", "IO exception: "+e);
        }
        return false;
    }
}
```