**What is Firebase?**

[**https://blog.mindorks.com/pushing-notifications-in-android-using-fcm**](https://blog.mindorks.com/pushing-notifications-in-android-using-fcm)

Firebase is a platform for your devices to develop awesome apps using collections of services provided by firebase.

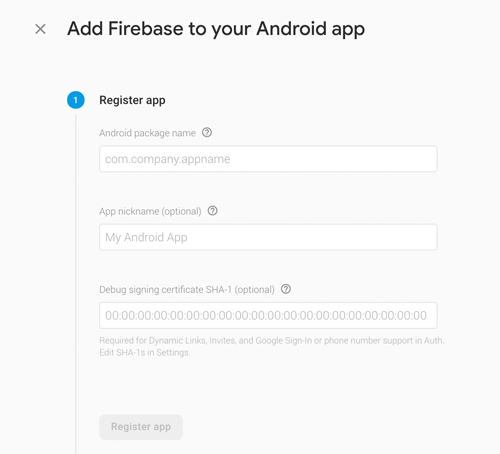
To integrate the Push Notification in our app, we need to use the Firebase Cloud Messaging Service of Firebase.

Now, Lets integrate the Service in our app,

#### Step 1:

First we need to setup out app to firebase console,

* First we need to create a project from [console.firebase.google.com](https://console.firebase.google.com/) by providing the Project Name and the Location
* Then we need to add out app, by sharing the app's package name ***(for eg: com.mindorks.app)*** , app's nick name, and a SHA1 key.



* To generate the SHA key we need to use ,

keytool -list -v \

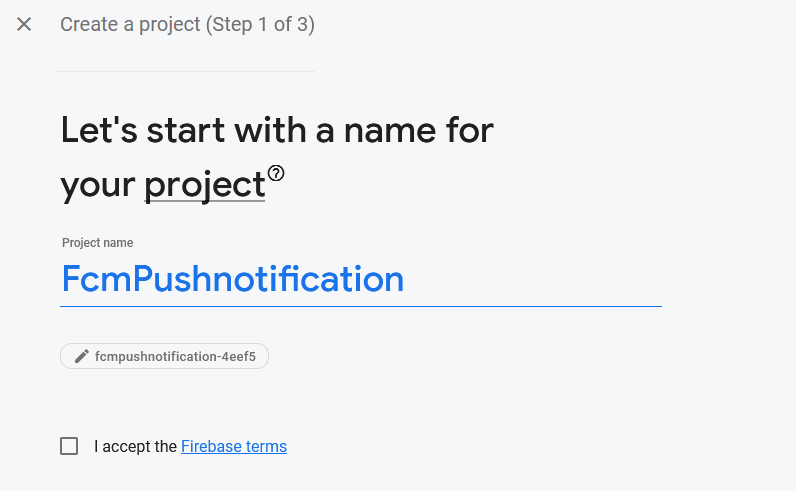
-alias androiddebugkey -keystore ~/.android/debug.keystore

and for Release SHA1 ,

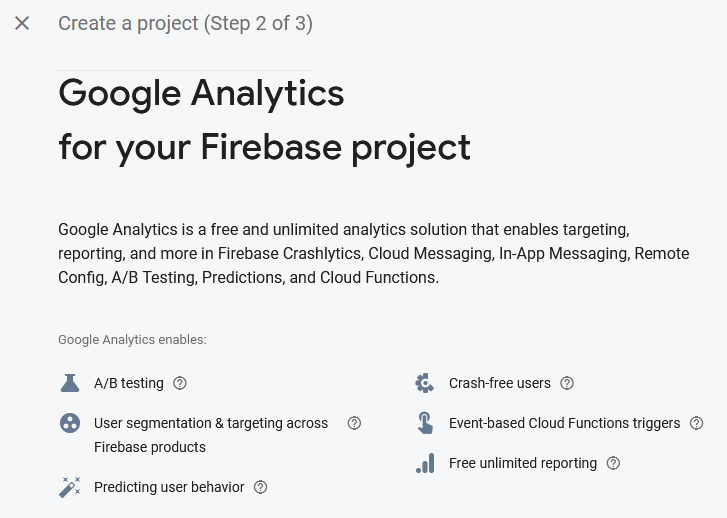
keytool -exportcert -list -v \

-alias <your-key-name> -keystore <path-to-production-keystore>

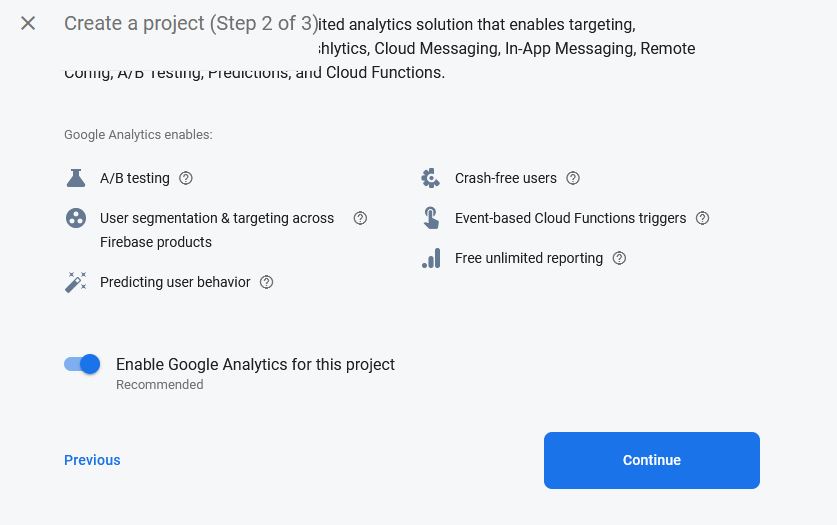
\* Once the Setup is done , we will get a google-serivces.json file ,

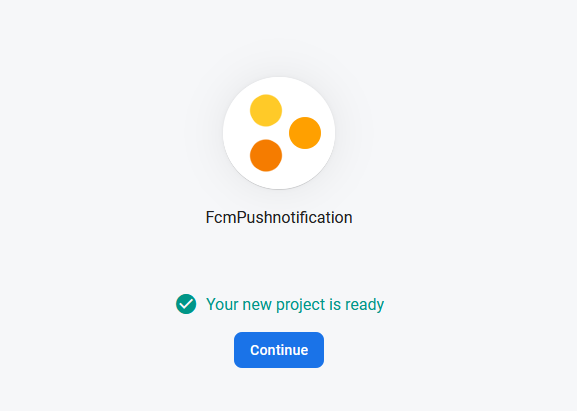


Project Unique Id - fcmpushnotification-4eef5

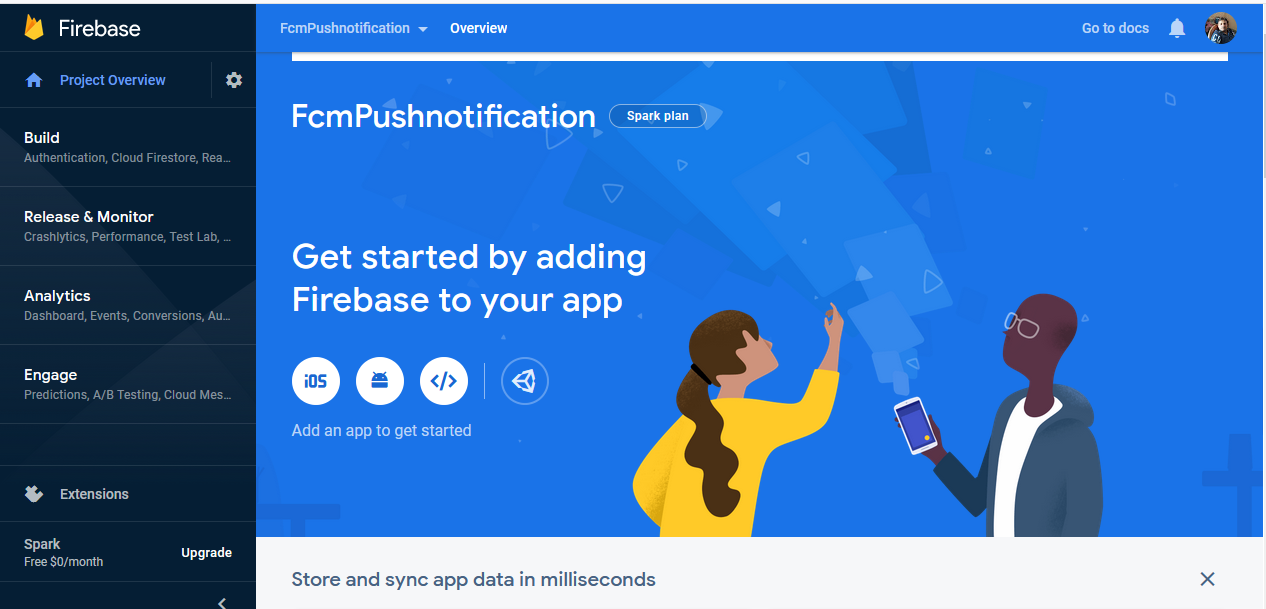


Step 2

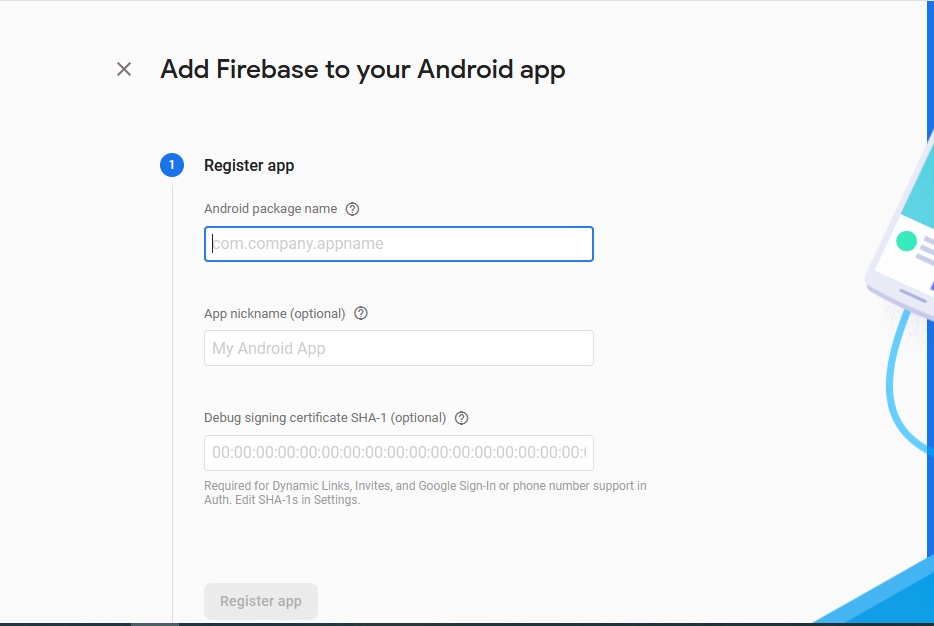




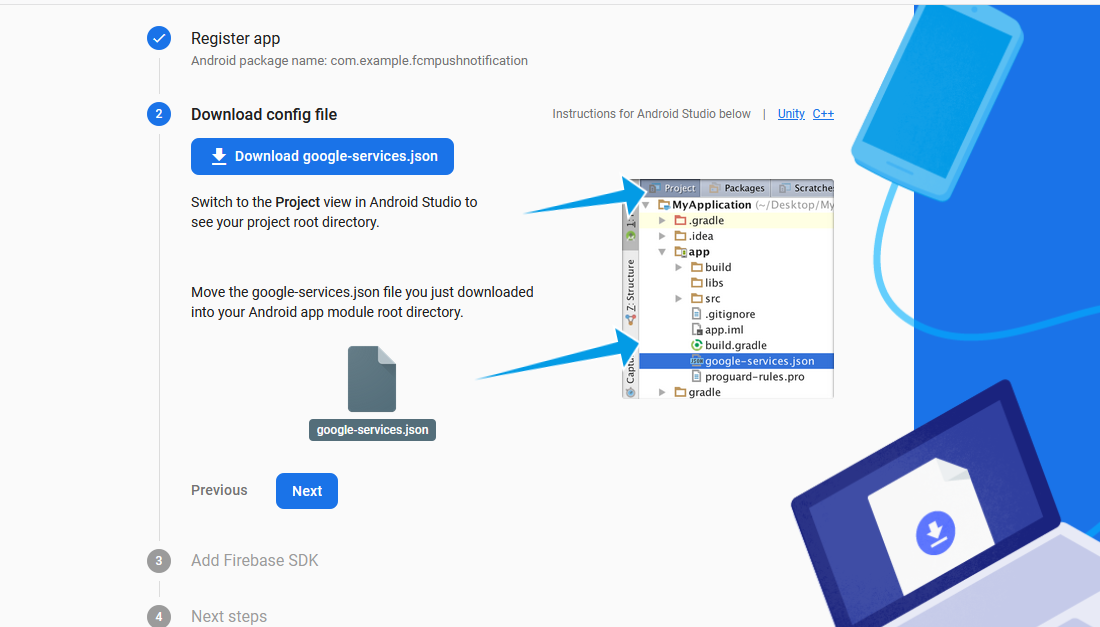
Now App Is generated and below Screen will come



Select “Android” button and give package name

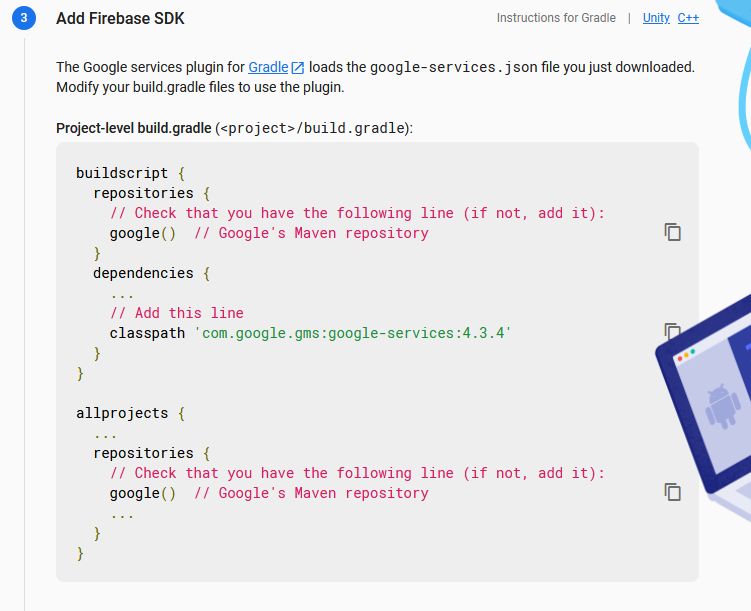


Enter your App Package name and click Register App

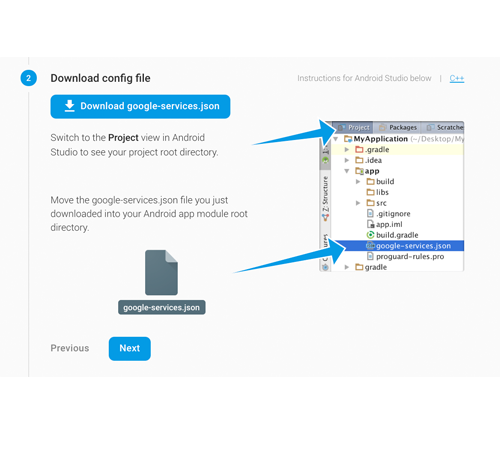


1.Download google-services.json file which will used in App

2.Click Next button



\* Once the Setup is done , we will get a google-serivces.json file ,



* We need to add the file in our project under app package.
* Once the json file is added we need to add the Firebase dependencies,

In the project's ***build.gradle,***

buildscript {

dependencies {

// Add this line

classpath 'com.google.gms:google-services:4.2.0'

}

}

and in app's ***build.gradle,***

dependencies {

// Add this line

implementation 'com.google.firebase:firebase-core:16.0.9'

}

and at the bottom of the app's ***build.gradle*** add,

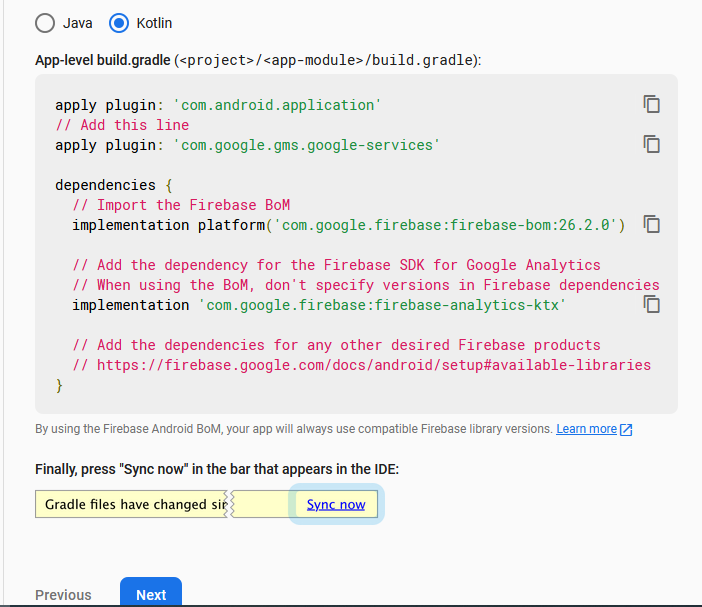
apply plugin: 'com.google.gms.google-services'

Now the setup of project is completely done and let's start with Firebase cloud Notification.



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:4.3.4'  
  }  
}  
  
allprojects {  
  ...  
  repositories {  
    // Check that you have the following line (if not, add it):  
    google()  // Google's Maven repository  
    ...  
  }  
}



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

apply plugin: 'com.android.application'  
// Add this line  
apply plugin: 'com.google.gms.google-services'  
  
dependencies {  
  // Import the Firebase BoM  
  implementation platform('com.google.firebase:firebase-bom:26.2.0')  
  
  // Add the dependency for the Firebase SDK for Google Analytics  
  // When using the BoM, don't specify versions in Firebase dependencies  
  implementation 'com.google.firebase:firebase-analytics-ktx'  
  
  // Add the dependencies for any other desired Firebase products  
  // https://firebase.google.com/docs/android/setup#available-libraries  
}

By using the Firebase Android BoM, your app will always use compatible Firebase library versions. [Learn more](https://firebase.google.com/docs/android/learn-more?authuser=1)

Finally, press "Sync now" in the bar that appears in the IDE:

Module/gradle

plugins **{** id 'com.android.application'  
 id 'kotlin-android'  
 id 'com.google.gms.google-services'  
**}**android **{** compileSdkVersion 30  
 buildToolsVersion "30.0.2"  
  
 defaultConfig **{** applicationId "com.example.fcmpushnotification"  
 minSdkVersion 19  
 targetSdkVersion 30  
 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* **}** kotlinOptions **{** jvmTarget = '1.8'  
 **}  
}**dependencies **{** implementation "org.jetbrains.kotlin:kotlin-stdlib:$kotlin\_version"  
 implementation 'androidx.core:core-ktx:1.3.2'  
 implementation 'androidx.appcompat:appcompat:1.2.0'  
 implementation 'com.google.android.material:material:1.2.1'  
 implementation 'androidx.constraintlayout:constraintlayout:2.0.4'  
 testImplementation 'junit:junit:4.+'  
 androidTestImplementation 'androidx.test.ext:junit:1.1.2'  
 androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'  
  
 *//Fcm  
 // Import the Firebase BoM* implementation platform('com.google.firebase:firebase-bom:26.2.0')  
 *// Add the dependency for the Firebase SDK for Google Analytics  
 // When using the BoM, don't specify versions in Firebase dependencies* implementation 'com.google.firebase:firebase-analytics-ktx'  
 *// Add the dependencies for any other desired Firebase products  
 // https://firebase.google.com/docs/android/setup#available-libraries***}**

Project / gradle

*// Top-level build file where you can add configuration options common to all sub-projects/modules.*buildscript **{** ext.kotlin\_version = "1.4.20"  
 repositories **{** google()  
 jcenter()  
 **}** dependencies **{** classpath "com.android.tools.build:gradle:4.1.1"  
 classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin\_version"  
 classpath 'com.google.gms:google-services:4.3.4'  
 *// NOTE: Do not place your application dependencies here; they belong  
 // in the individual module build.gradle files* **}  
}**allprojects **{** repositories **{** google()  
 jcenter()  
 **}  
}**task clean(type: Delete) **{** delete rootProject.buildDir  
**}**

#### Step 2:

Let's add the FCM dependency by adding the following in app level ***build.gradle***,

implementation 'com.google.firebase:firebase-messaging:18.0.0'

#### Step 3:

We need to create a service which extends ***FirebaseMessagingService***

class MyFirebaseMessagingService : FirebaseMessagingService()

and add it to the Andorid Manifest file using,

<service

android:name=".MyFirebaseMessagingService"

android:exported="false">

<intent-filter>

<action android:name="com.google.firebase.MESSAGING\_EVENT" />

</intent-filter>

</service>

FirebaseMessagingService is responsible for handling the messaging through Firebase Cloud Messaging

**Step 4:**

To communicate with the App of an user, the Firebase needs a specific identity of the user. This is handled by the Firebase Token which is unique for every user. To generate the token we need to call,

FirebaseInstanceId.getInstance().instanceId

.addOnCompleteListener(OnCompleteListener { task ->

if (!task.isSuccessful) {

Log.w(TAG, "getInstanceId failed", task.exception)

return@OnCompleteListener

}

// Get new Instance ID token

val token = task.result?.token

// Log and toast

val msg = getString(R.string.msg\_token\_fmt, token)

Log.d(TAG, msg)

Toast.makeText(baseContext, msg, Toast.LENGTH\_SHORT).show()

})

Here the token are volatile.The registration token changes when:

* The app deletes Instance ID
* The app is restored on a new device
* The user uninstalls/reinstall the app
* The user clears app data.

We need to geneate a new token ,

class MyFirebaseMessagingService : FirebaseMessagingService() {

override fun onMessageReceived(remoteMessage: RemoteMessage?) {

Log.d(TAG, "From: ${remoteMessage?.from}")

// Check if message contains a notification payload.

remoteMessage?.notification?.let {

Log.d(TAG, "Message Notification Body: ${it.body}")

//Message Services handle notification

}

}

override fun onNewToken(token: String) {

//handle token

}

}

When the new token is generated we need to update the token wherever we store the token. We need to store the token in our backend service by using some API for better communication between app and our backend services.

In ***OnMessageReceived*** , we handle the Data Payload which we receive from our backend server/Firebase console. We can use Notification Manager class to show it as notification for the app or we can use it however we want.

#### Step 5:

To Show the message as Notifiation we add in onMessageReceived(),

class MyFirebaseMessagingService : FirebaseMessagingService() {

override fun onMessageReceived(remoteMessage: RemoteMessage?) {

Log.d(TAG, "From: ${remoteMessage?.from}")

// Check if message contains a notification payload.

remoteMessage?.notification?.let {

Log.d(TAG, "Message Notification Body: ${it.body}")

//Message Services handle notification

val notification = NotificationCompat.Builder(this)

.setContentTitle(remoteMessage.from)

.setContentText(it.body)

.setSmallIcon(icon)

.build()

val manager = NotificationManagerCompat.from(applicationContext)

manager.notify(/\*notification id\*/0, notification)

}

}

override fun onNewToken(token: String) {

//handle token

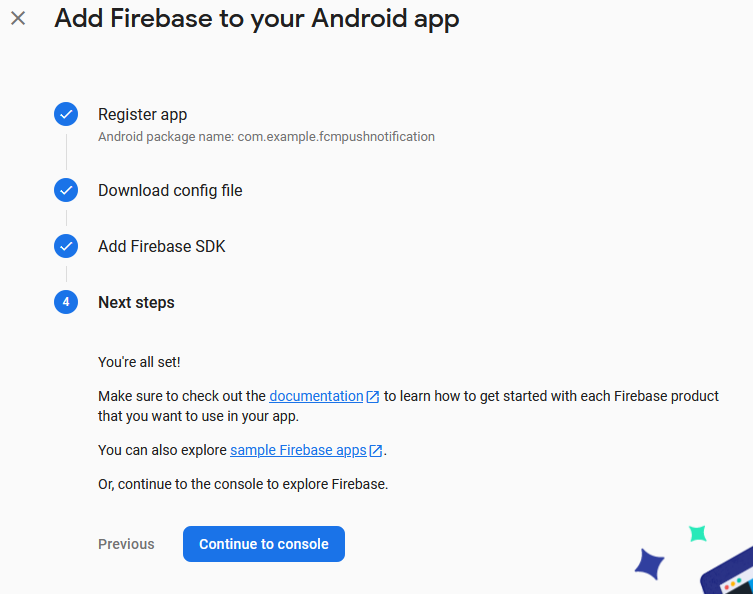
}

}

Run and check token generated

MyFirebaseMessagingService

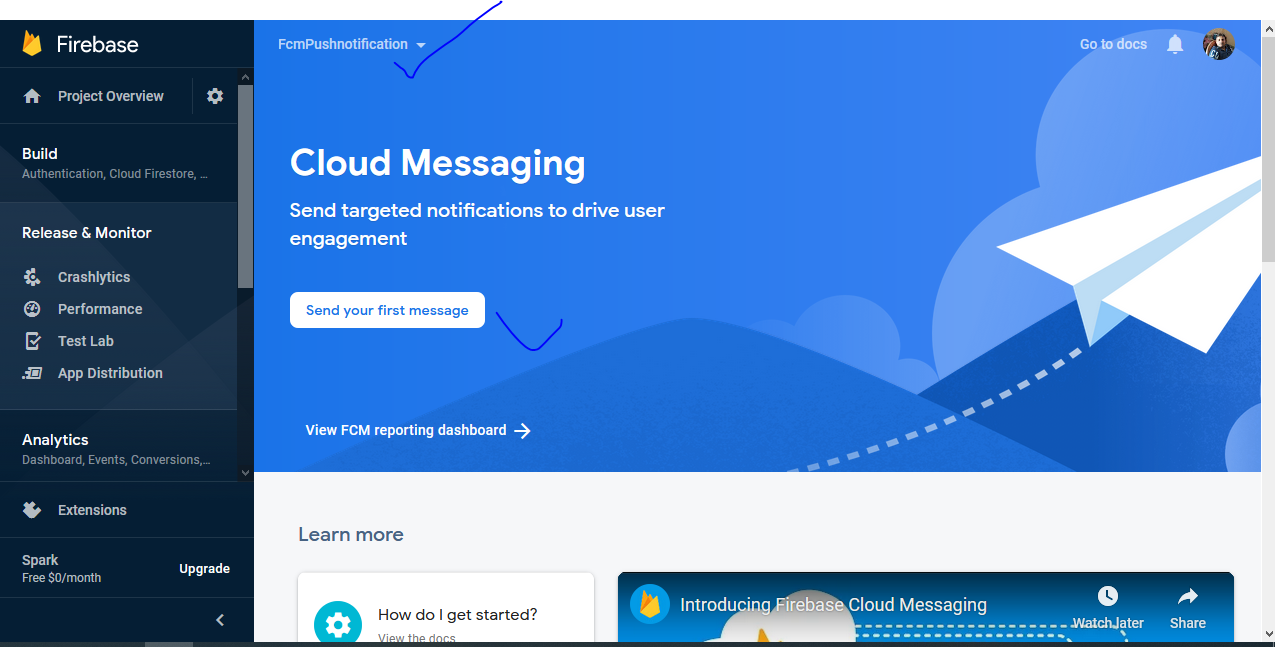
D/MyFirebaseMessagingServ: onNewToken: cP1cDZH6Qmint8AGXzz3jV:APA91bHLB5UIKUWFP95esxa8pH9ZcFP08-j0Ql-Za5JXoPauQs4k7913QIBVaGnnX9UWZvCkNpT68fAnCQIUXVxfDGV44rc1zLk1gDFQkYv4lbSSu2gffH\_GgsaRahysNUyBB8D9PsA0 -xyzywas

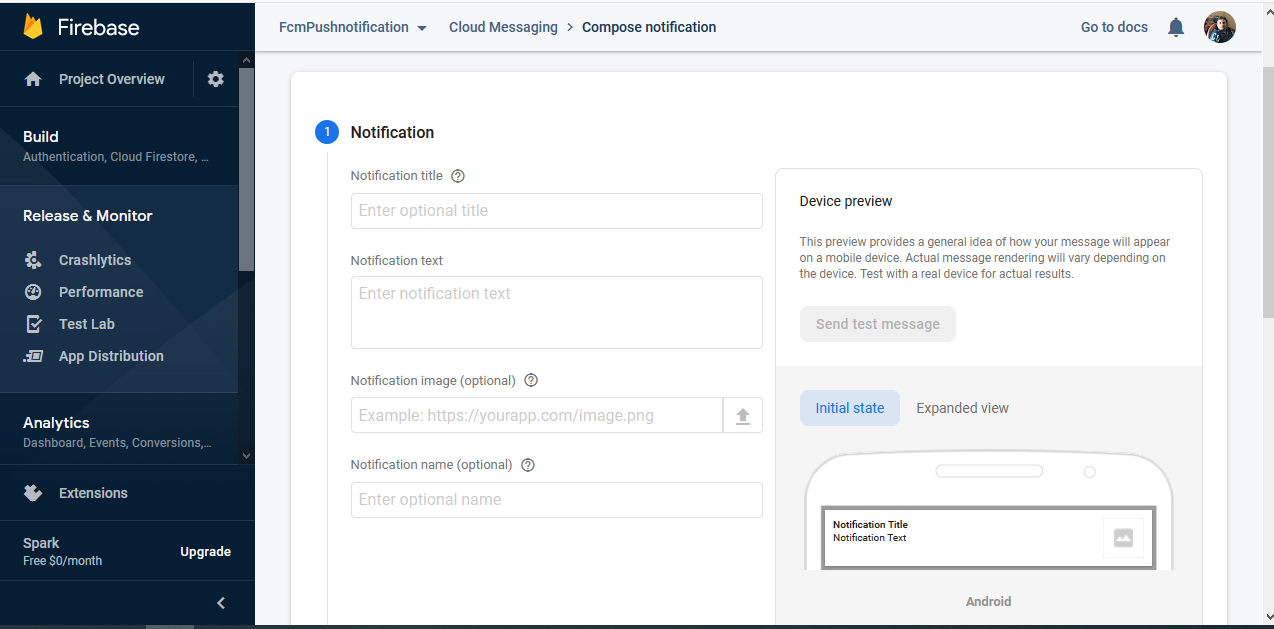


Now Send Test Message

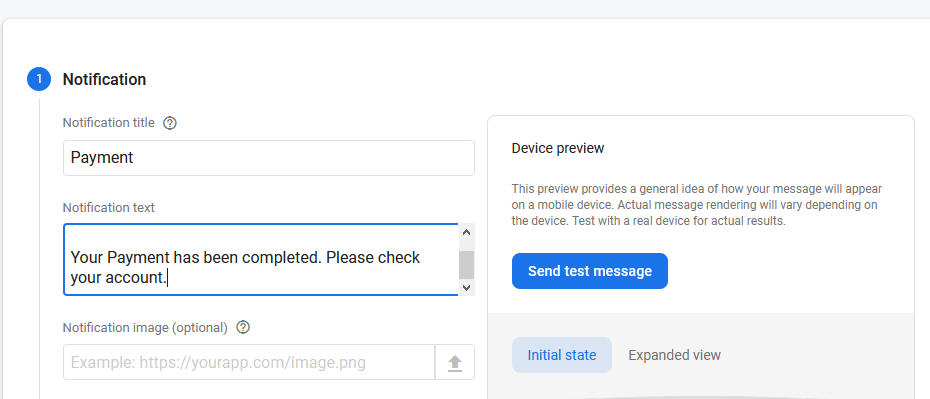
<https://console.firebase.google.com/u/1/project/fcmpushnotification-4eef5/notification>

Select your app and Click send your first message

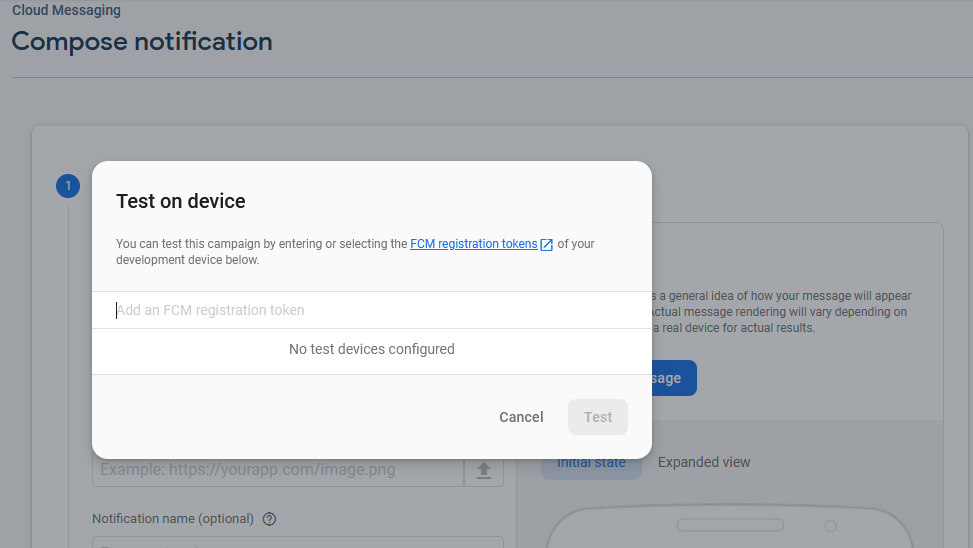




Enter Message Title and Message body and click send Message

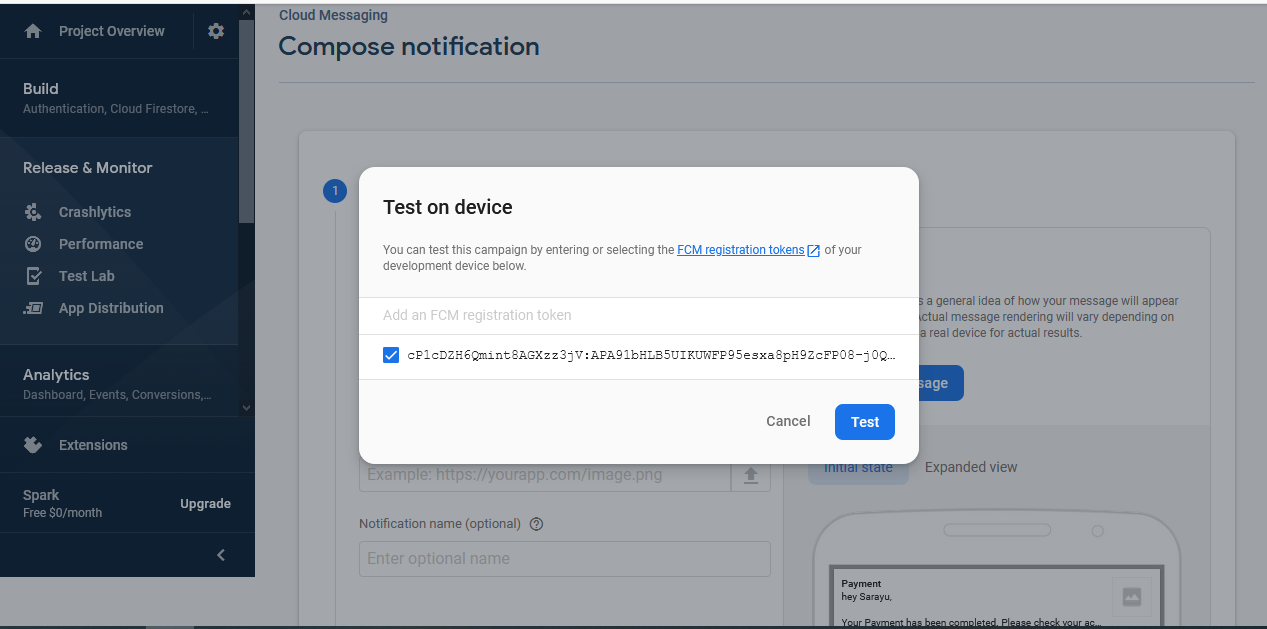


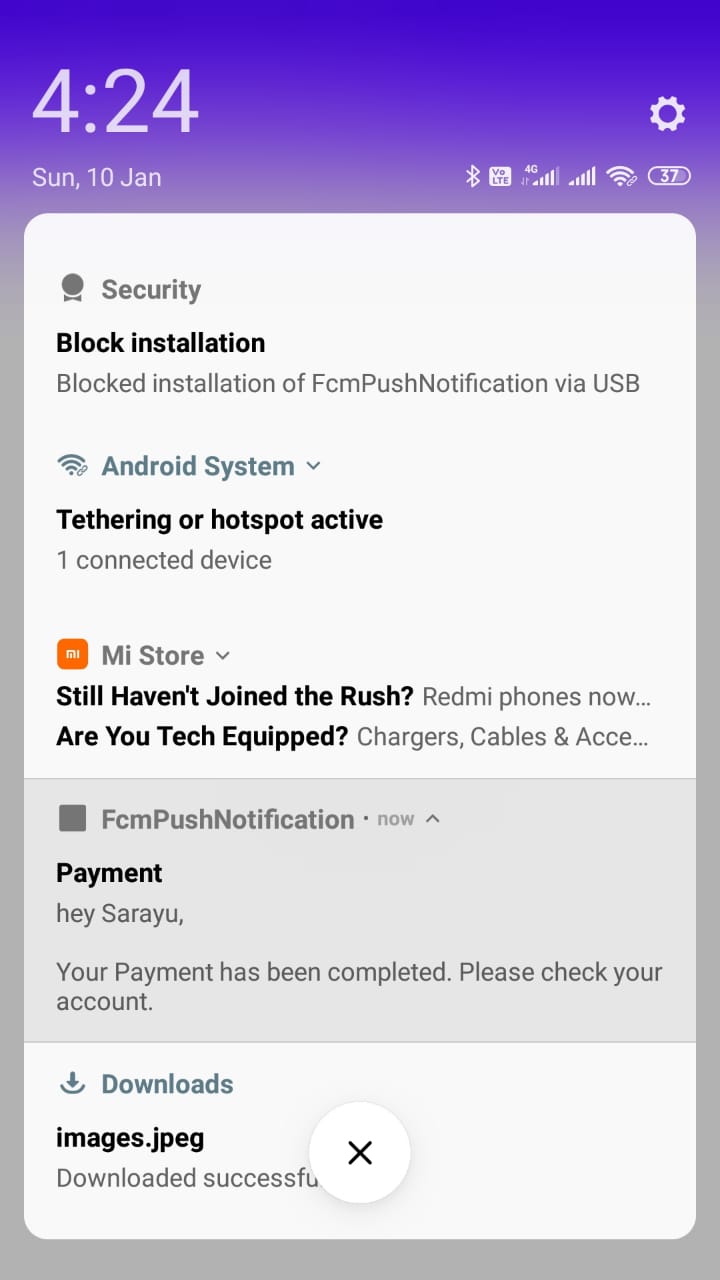
After clicking Send test Message . It will ask your token number



Provide your Token Number

cP1cDZH6Qmint8AGXzz3jV:APA91bHLB5UIKUWFP95esxa8pH9ZcFP08-j0Ql-Za5JXoPauQs4k7913QIBVaGnnX9UWZvCkNpT68fAnCQIUXVxfDGV44rc1zLk1gDFQkYv4lbSSu2gffH\_GgsaRahysNUyBB8D9PsA0





Now It is completed.

## 1. Why is this happening?

There are two types of messages in FCM (Firebase Cloud Messaging):

1. **Display Messages**: These messages trigger the onMessageReceived() callback only when your app is in **foreground**
2. **Data Messages**: Theses messages trigger the onMessageReceived() callback **even** if your app is in **foreground/background/killed**