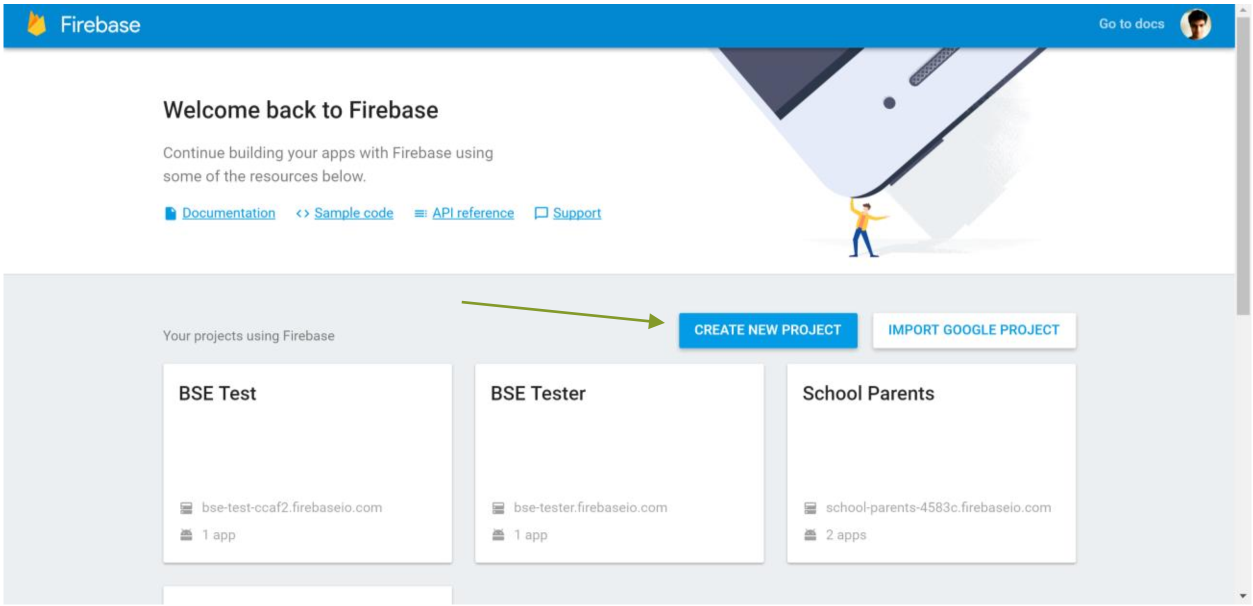
SETTING UP FCM MESSAGING SERVICE ON CLIENT SIDE (ANDROID)

WHAT IS FCM?

FCM stands for Firebase Cloud Messaging, and is an upgrade to the already existing Google Cloud Messaging. Some of the advantages include easier implementation on client end and also much faster than its predecessor. Here is a documentation of how to implement Firebase in your project.

STEP 1

Go to Firebase (https://console.firebase.google.com/). Login in using your Google account, or create a new Account if you do not have one. Once logged in, you will see a screen like the one below. Click on "Create New Project".



Welcome back to Firebase

Continue building your apps with Firebase using some of the resources below.

Your projects using Firebase

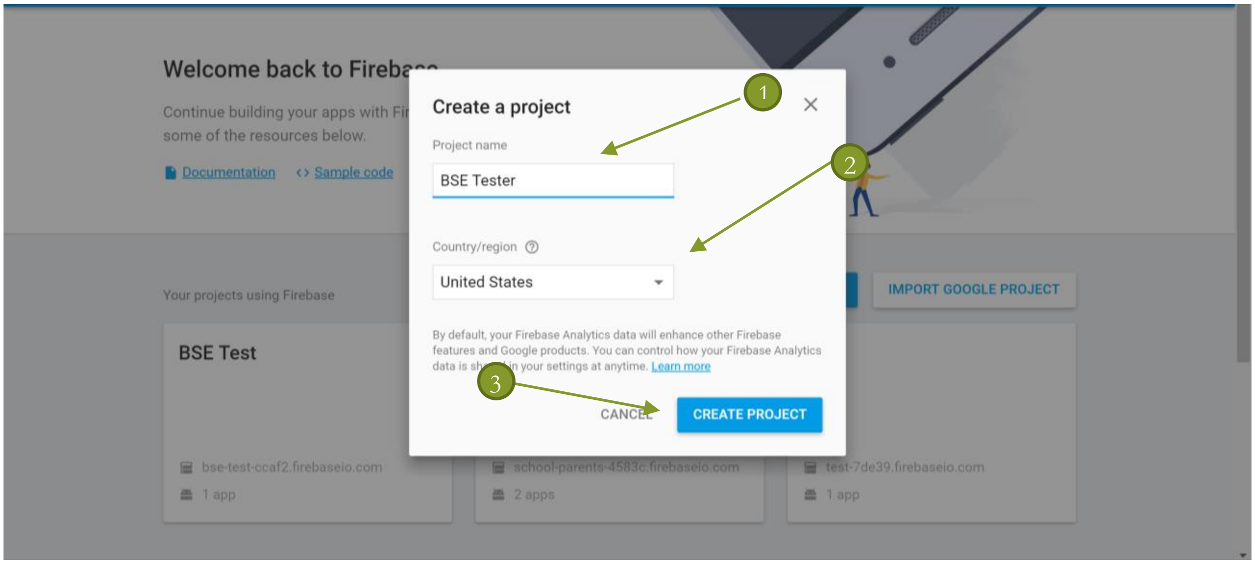
BSE Test
bse-test-ccaf2.firebaseio.com
1 app

BSE Tester
bse-tester.firebaseio.com
1 app

School Parents
school-parents-4583c.firebaseio.com
2 apps

STEP 2

Enter your Project Name and Country and Click “Create Project”.


Welcome back to Fireba
Continue building your apps with Fi
some of the resources below.
Documentation
<> Sample code
Your projects using Firebase
BSE Test
bse-test-ccaf2.firebaseio.com
1 app
school-parents-4583c.firebaseio.com
2 apps
test-7de39.firebaseio.com
1 app
IMPORT GOOGLE PROJECT
1
2
3
CANCEL
CREATE PROJECT

Create a project

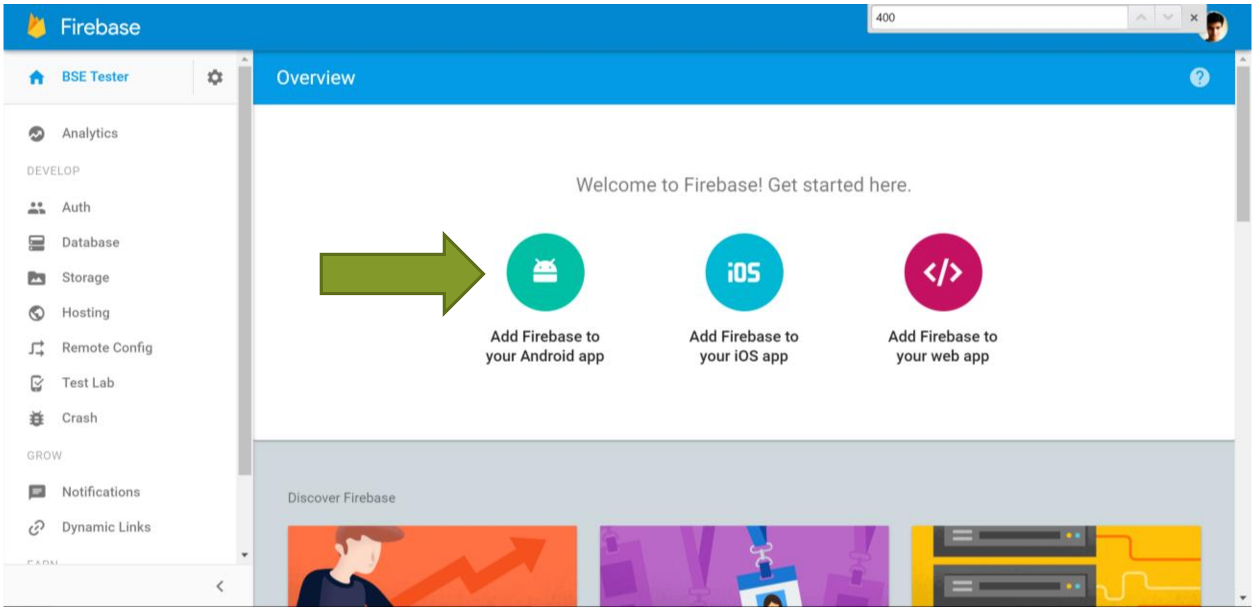
Project name: BSE Tester

Country/region: United States

By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is sh in your settings at anytime. Learn more

STEP 3

You will receive a screen like this. Click on the “Add Firebase to your Android App”.

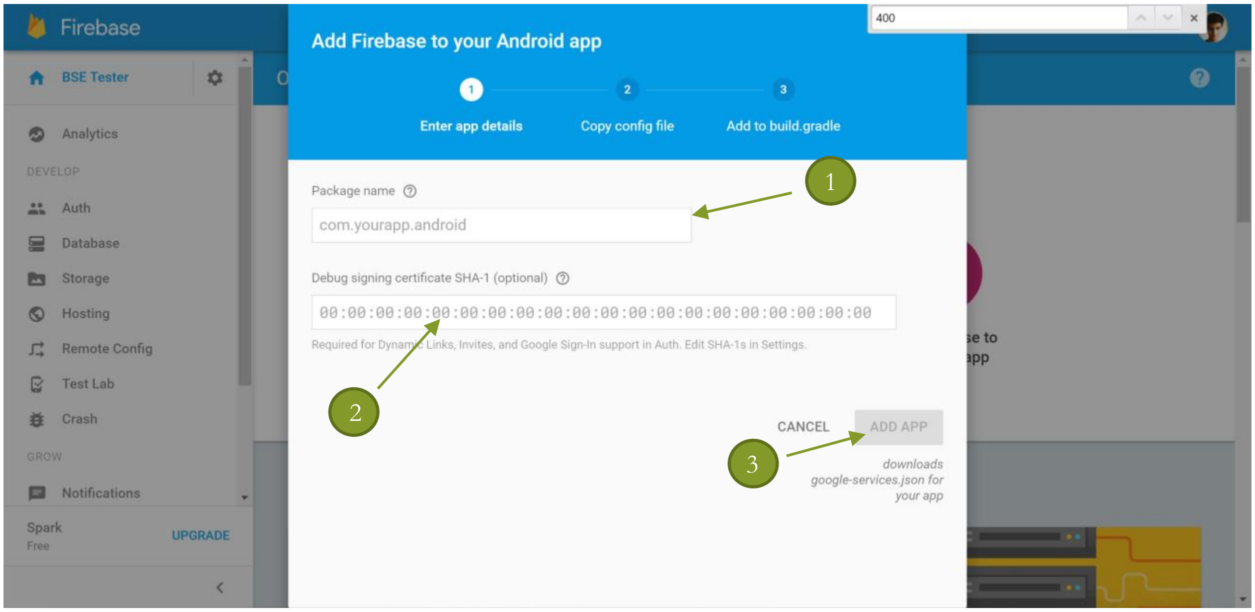


Welcome to Firebase! Get started here.

STEP 4

Enter the package name found in your Android Manifest. It is recommended that you also enter your SHA-1 certificate. To get this, please refer to https://chandruscm.wordpress.com/2015/12/29/how-to-obtain-sha1-signing-certificate-fingerprint-from-android-studio/

After you have entered the details, press "Add App"



Add Firebase to your Android app

Package name: com.yourapp.android

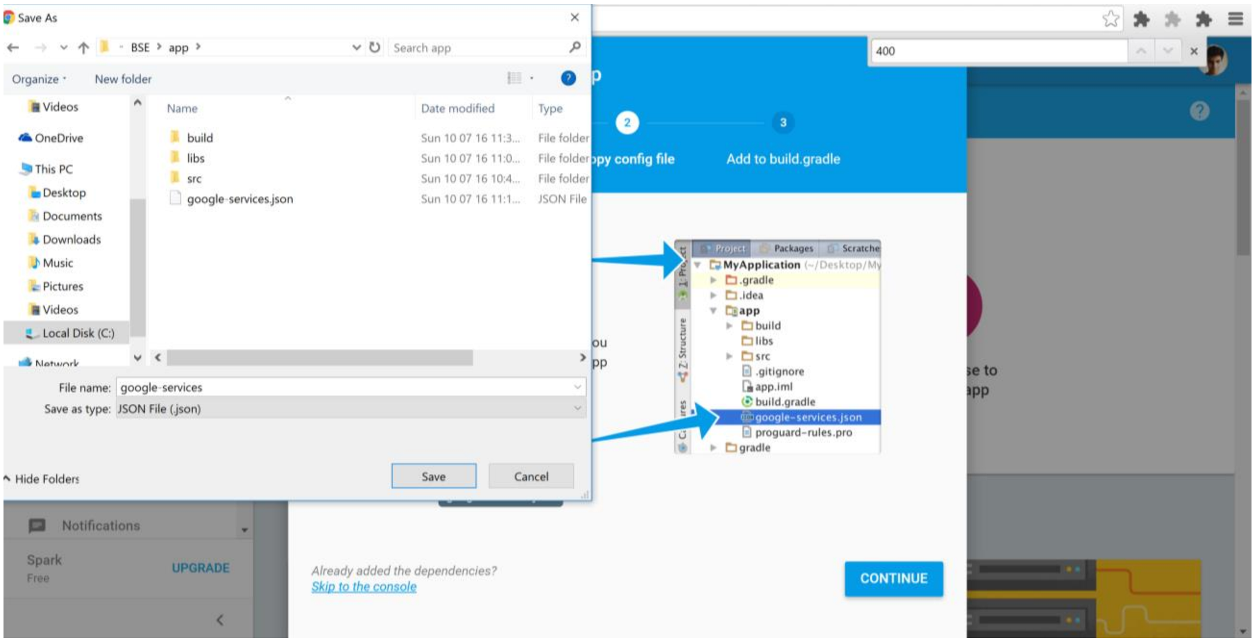
Debug signing certificate SHA-1 (optional): 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

Required for Dynamic Links, Invites, and Google Sign-In support in Auth. Edit SHA-1s in Settings.

downloads google-services.json for your app

STEP 4

Google will automatically ask you to download a google-services.json file. Download the .json file and copy it into the app folder of your project. Typically this is found under C:/Users/(your user nae)/AndroidStudioProjects/(project name). Then click “Continue”.



Save As

BSE > app

Name	Date modified	Type
build	Sun 10 07 16 11:3...	File folder
libs	Sun 10 07 16 11:0...	File folder
src	Sun 10 07 16 10:4...	File folder
google-services.json	Sun 10 07 16 11:1...	JSON File

File name: google-services

Save as type: JSON File (.json)

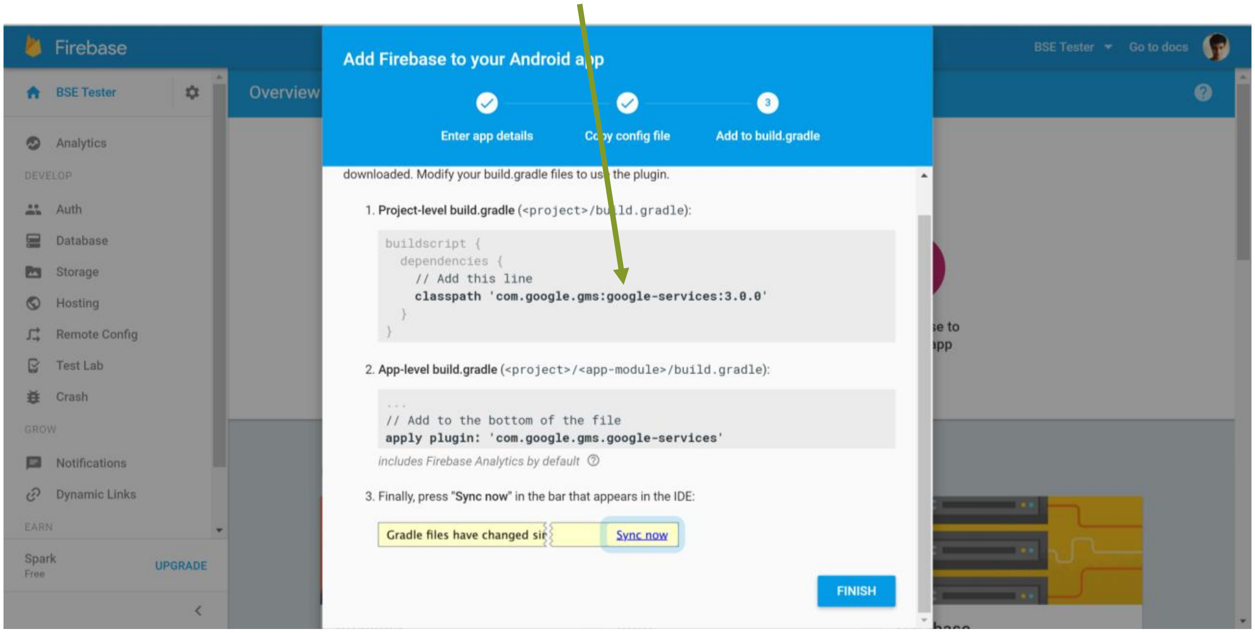
STEP 5

Add this dependency to your app-level build.gradle file:

```
dependencies {
     compile 'com.google.firebase:firebase-messaging:9.2.0'

     compile 'com.google.android.gms:play-services:9.2.0'
}

//Add this to the end of your file

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

Note: Please ensure the firebase-messaging:x-x-x is updated. To get the latest version number, please refer to the one used in https://firebase.google.com/docs/android/setup. To get the latest version number of play-services:x-x-x, please refer to the version used in https://developers.google.com/android/guides/setup

Add the following to your project-leve build.grade file, then click "Finish".



Add Firebase to your Android app

downloaded. Modify your build.gradle files to use the plugin.

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

```
buildscript {
  dependencies {
    // Add this line
    classpath 'com.google.gms:google-services:3.0.0'
  }
}
```

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

```
...
// Add to the bottom of the file
apply plugin: 'com.google.gms.google-services'
```

includes Firebase Analytics by default

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

Gradle files have changed si Sync now

Note: Do not forget to press sync-now after changing the Gradle file for it to make the changes compatible in the IDE.

STEP 6

Edit your Android Manifest to add these 2 services. It is important that these are under your application tag. Find more information on https://firebase.google.com/docs/cloud-messaging/android/client

```
<application…>

...

<service
    android:name=".MyFirebaseMessagingService">
    <intent-filter>
        <action android:name="com.google.firebase.MESSAGING_EVENT"/>
    </intent-filter>
</service>

  <service
    android:name=".MyFirebaseInstanceIDService">
    <intent-filter>
        <action android:name="com.google.firebase.INSTANCE_ID_EVENT"/>
    </intent-filter>
</service>

...

</application>
```

Also, you can delete all the GCM services such as:

```
<!-- [START
gcm_permission] -
->
    <uses-permission
android:name="com.google.android.c2dm.permission.RECEIVE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <!-- [END gcm_permission] -->
<receive
r
          android:name="com.google.android.gms.gcm.GcmReceiver"
          android:exported="true"
          android:permission="com.google.android.c2dm.permission.SEND" >
          <intent-filter>
              <action android:name="com.google.android.c2dm.intent.RECEIVE"
/>
```

```
                <category android:name="gcm.play.android.samples.com.gcmquickstart" />
            </intent-filter>
        </receiver>
        <!-- [END gcm_receiver] -->


        <!-- [START gcm_listener] -->
        <service
            android:name="gcm.play.android.samples.com.gcmquickstart.MyGcmListenerService"
            android:exported="false" >
            <intent-filter>
                <action android:name="com.google.android.c2dm.intent.RECEIVE" />
            </intent-filter>
        </service>
        <!-- [END gcm_listener] -->
        <!-- [START instanceId_listener] -->
        <service
            android:name="gcm.play.android.samples.com.gcmquickstart.MyInstanceIDListenerService"
            android:exported="false">
            <intent-filter>
                <action android:name="com.google.android.gms.iid.InstanceID"/>
            </intent-filter>
        </service>
        <!-- [END instanceId_listener] -->
        <service
            android:name="gcm.play.android.samples.com.gcmquickstart.RegistrationIntentService"
            android:exported="false">
        </service>
```

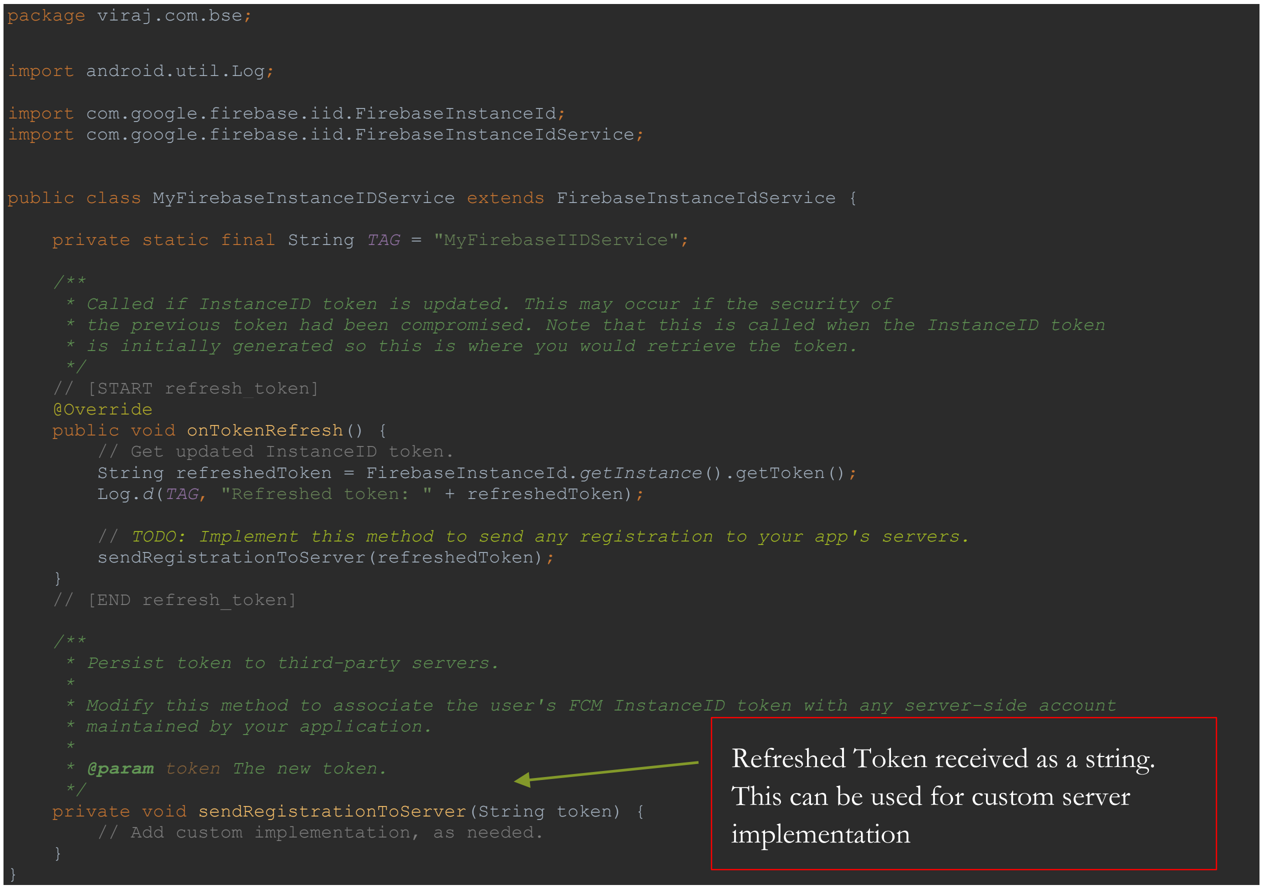
STEP 7

Create a new class called MyFirebaseInstanceIDService in your project and copy the code below. This class is intended to notify any changes in the token of your device, in a case where the security of the token is compromised.

The registration token may change 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.

The refreshed token is passed on to the sendRegistrationToServer() method. Through this, you implement a custom processing of the token, such as updating the token in your server to ensure the notifications are received.



```java
package viraj.com.bse;


import android.util.Log;

import com.google.firebase.iid.FirebaseInstanceId;
import com.google.firebase.iid.FirebaseInstanceIdService;


public class MyFirebaseInstanceIDService extends FirebaseInstanceIdService {

    private static final String TAG = "MyFirebaseIIDService";

    /**
     * Called if InstanceID token is updated. This may occur if the security of
     * the previous token had been compromised. Note that this is called when the InstanceID token
     * is initially generated so this is where you would retrieve the token.
     */
    // [START refresh_token]
    @Override
    public void onTokenRefresh() {
        // Get updated InstanceID token.
        String refreshedToken = FirebaseInstanceId.getInstance().getToken();
        Log.d(TAG, "Refreshed token: " + refreshedToken);

        // TODO: Implement this method to send any registration to your app's servers.
        sendRegistrationToServer(refreshedToken);
    }
    // [END refresh_token]

    /**
     * Persist token to third-party servers.
     *
     * Modify this method to associate the user's FCM InstanceID token with any server-side account
     * maintained by your application.
     *
     * @param token The new token.
     */
    private void sendRegistrationToServer(String token) {
        // Add custom implementation, as needed.
    }
}
```

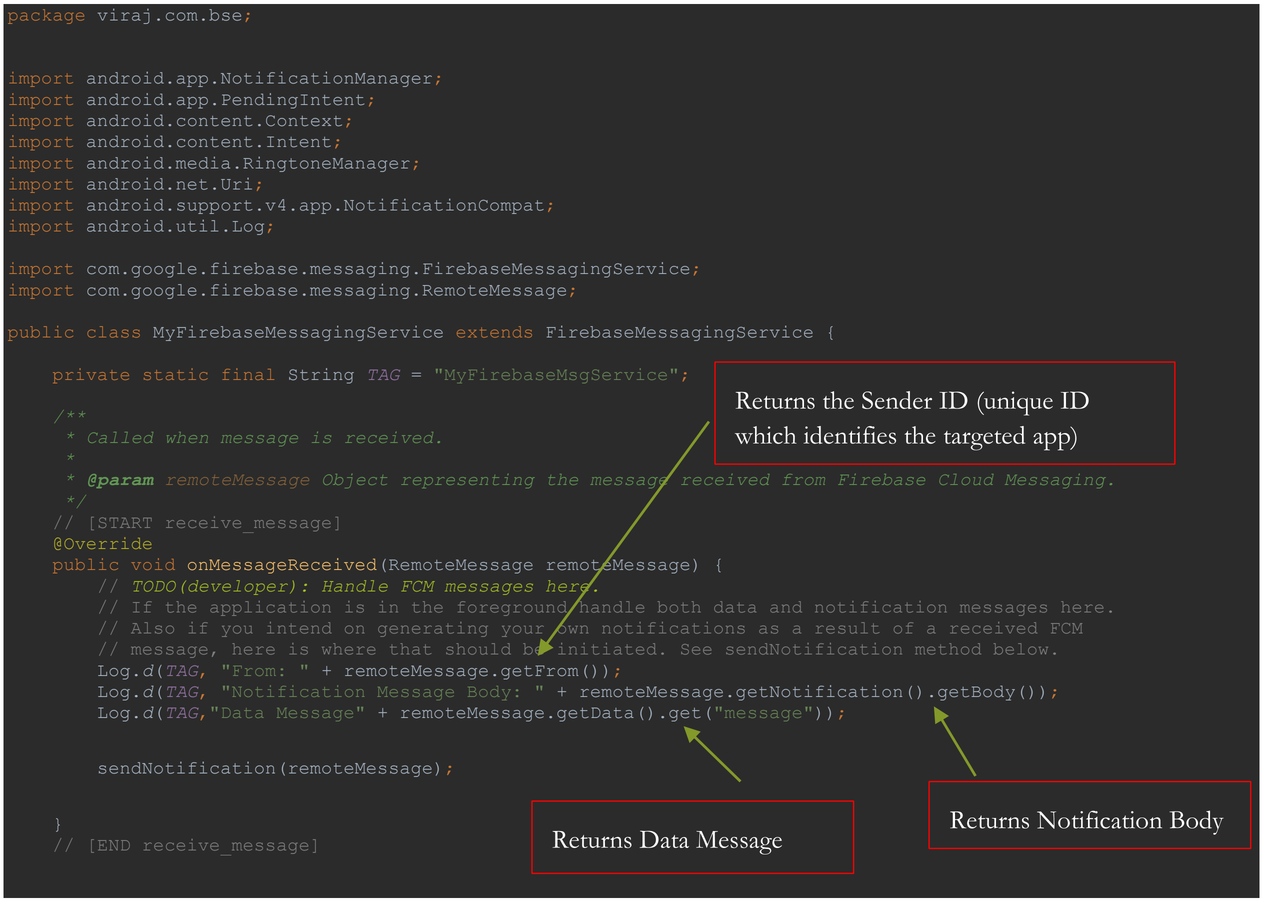
Refreshed Token received as a string. This can be used for custom server implementation

Code Referenced From: https://github.com/firebase/quickstart-android/blob/master/messaging/app/src/main/java/com/google/firebase/quickstart/fcm/MyFirebaseInstanceIDService.java

STEP 8

Create a new classs called MyFirebaseMessagingService in your project and copy the code below. This service is used to process incoming messages. This is a little different from the implementation of Google Cloud Messaging as in the latter we used to receive a data Bundle while here we receive a message of class RemoteMessage. We can retrieve the message from remoteMessage.getData.get("message"). We can also pass different data through different tags in the data json file while sending the message. For more information, please read https://developers.google.com/cloud-messaging/http-server-ref.

In this, we can process the message through the onMessageRecieved() method. You can also set what kind of a notification you would like to generate, in case you want to.



```java
package viraj.com.bse;


import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;
import android.support.v4.app.NotificationCompat;
import android.util.Log;

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

public class MyFirebaseMessagingService extends FirebaseMessagingService {

    private static final String TAG = "MyFirebaseMsgService";

    /**
     * Called when message is received.
     *
     * @param remoteMessage Object representing the message received from Firebase Cloud Messaging.
     */
    // [START receive_message]
    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {
        // TODO(developer): Handle FCM messages here.
        // If the application is in the foreground handle both data and notification messages here.
        // Also if you intend on generating your own notifications as a result of a received FCM
        // message, here is where that should be initiated. See sendNotification method below.
        Log.d(TAG, "From: " + remoteMessage.getFrom());
        Log.d(TAG, "Notification Message Body: " + remoteMessage.getNotification().getBody());
        Log.d(TAG,"Data Message" + remoteMessage.getData().get("message"));


        sendNotification(remoteMessage);


    }
    // [END receive_message]
```

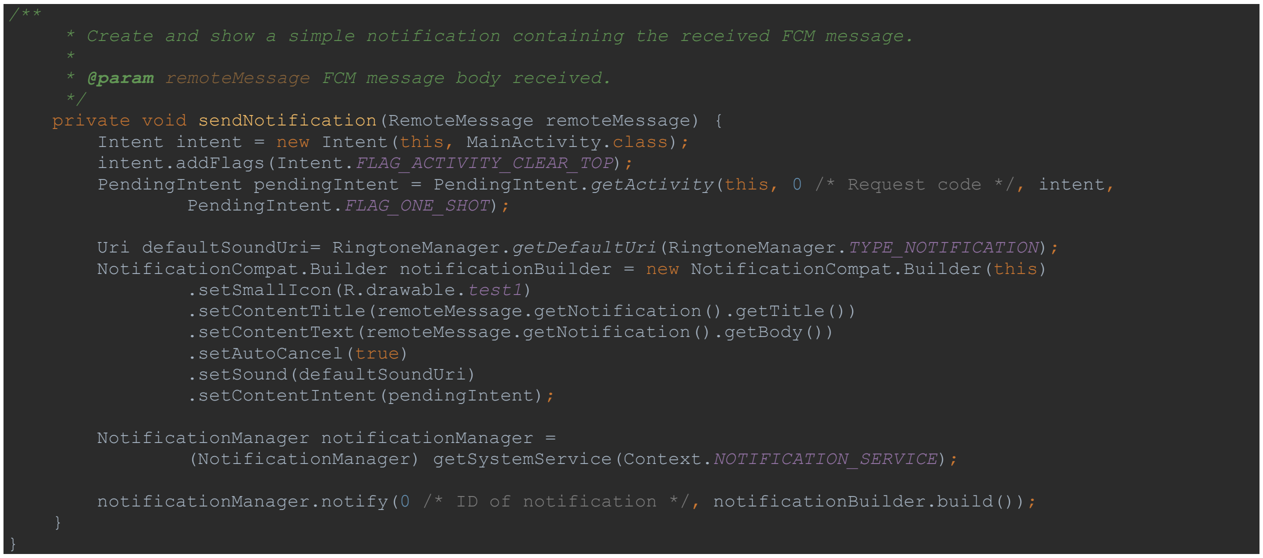
Returns the Sender ID (unique ID which identifies the targeted app)

Returns Data Message

Returns Notification Body

The sendNotification() is intended to build a notification for the user. This method accepts a RemoteMessage and extracts all the notification data and passes it through the Notification Builder.

In the initial code, it passed a string of value remoteMessage.getNotification().getBody() but this has been changed so that we can specify more information in the notification, such as Title, the icon etc.



```
/**
     * Create and show a simple notification containing the received FCM message.
     *
     * @param remoteMessage FCM message body received.
     */
    private void sendNotification(RemoteMessage remoteMessage) {
        Intent intent = new Intent(this, MainActivity.class);
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0 /* Request code */, intent,
                PendingIntent.FLAG_ONE_SHOT);

        Uri defaultSoundUri= RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
        NotificationCompat.Builder notificationBuilder = new NotificationCompat.Builder(this)
                .setSmallIcon(R.drawable.test1)
                .setContentTitle(remoteMessage.getNotification().getTitle())
                .setContentText(remoteMessage.getNotification().getBody())
                .setAutoCancel(true)
                .setSound(defaultSoundUri)
                .setContentIntent(pendingIntent);

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

        notificationManager.notify(0 /* ID of notification */, notificationBuilder.build());
    }
}
```

Code Referenced From:

https://github.com/firebase/quickstart-android/blob/master/messaging/app/src/main/java/com/google/firebase/quickstart/fcm/MyFirebaseMessagingService.java