**5. Android Push Notifications using Firebase Cloud Messaging**

In android, **Push Notification** is used to send an important messages to the mobile devices about our mobile app. By using Push Notifications we can let people know about the events happening in our app by sending a notifications to the mobile devices which contains our app. For example, shopping app’s which will send a notifications about the sale happening in their sites.

In android, by using Firebase Cloud Messaging (FCM) we can easily push a notifications to the mobile devices which will contain our mobile app.

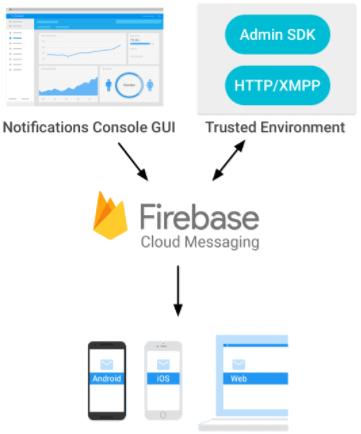
**What is Firebase Cloud Messaging?**

**Firebase Cloud Messaging** (FCM) is a cross platform (Android, iOS, Mobile Web) messaging solution which is used to send notification messages to the mobile devices at no cost.

By using firebase we can easily send messages to any device or schedule a messages to send in user’s local time zone based on our requirements.

We don’t need to write much coding to send a notifications and firebase will provide a detailed engagement and conversion tracking for notification messages.

Following is the pictorial representation of how firebase will work with android applications to send or receive messages.



Now we will see how to use firebase cloud messaging in our android applications to push notifications based on our requirements.

**Android Push Notifications Example**

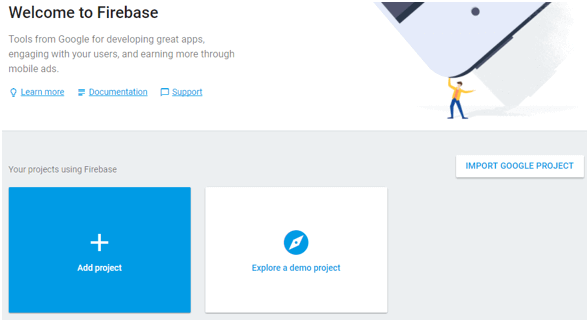
Create a new android application using android studio and give names as **PushNotification**.

Once we are done with creation of new app, we need to integrate a Firebase in our android applications to push notifications.

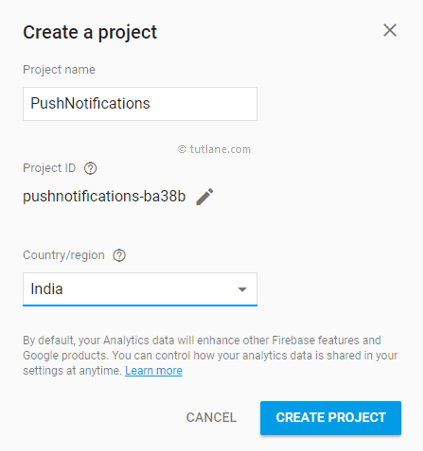
**Add Firebase to our Android App**

To add Firebase to our android app, first we need to create a Firebase project and Firebase configuration file for our app.

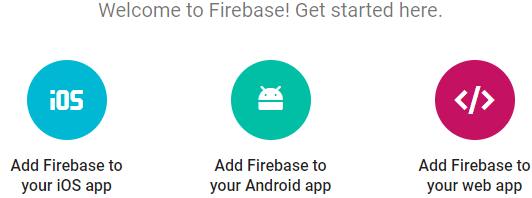
To create a Firebase project. Open [Firebase Console](https://console.firebase.google.com/?authuser=0) and it will ask you to login with Google mail account. Once we are logged in, select **Add Project** like as shown below.



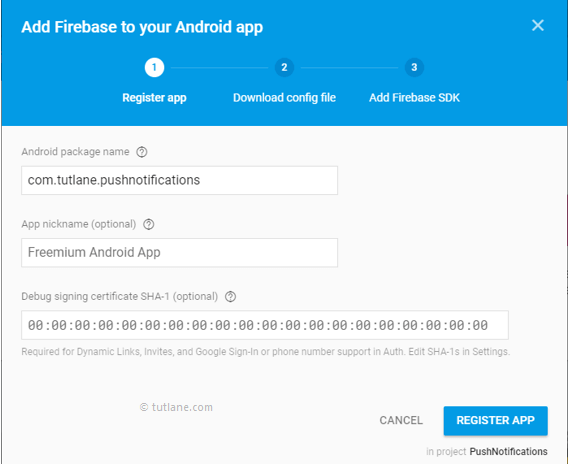
Once we click on **Add Project**, a new popup will open in that enter required details like project name, country and click on **Create Project** like as shown below.



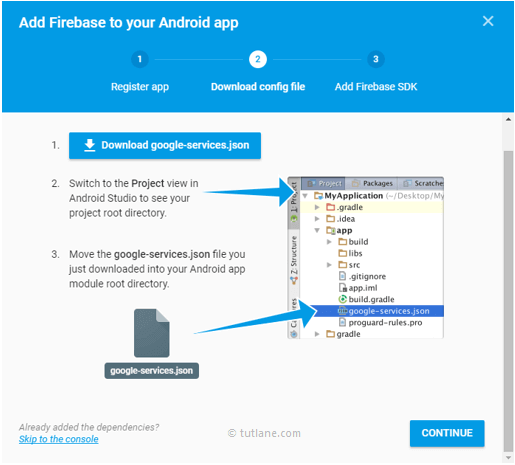
Once we click on Create Project, our project will be created. Now we need to add Firebase to our android app for that we need to select **Add Firebase to your Android App** option from the available options like as shown below.



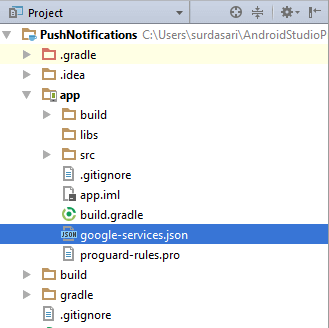
Once we select the required platform, a new popup will open in that we need to enter our app **package name**. This package name must be same as which we defined in our android **AndroidMaifest.xml** file and click **Register App** button like as shown below.



Once we register our app, we will get **google-services.json** file to download and add it to our project **app** folder like as shown below.



Once we add the downloaded **google-services.json** file in our project **app** folder that will be like as shown below.



Now we need to modify our application **build.gradle** file in project level (**<project>/build.gradle**) to include our firebase plugin file (**google-services.json**) like as shown below.

// Top-level build file where you can add configuration options common to all sub-projects/modules.  
  
buildscript {  
    repositories {  
        jcenter()  
    }  
    dependencies {  
        classpath 'com.android.tools.build:gradle:2.3.3'  
           // Add this line  
        classpath 'com.google.gms:google-services:3.1.0'  
     }  
}

Now open **app** level **build.gradle** (<project>/<app-module>/build.gradle) and make following changes at the bottom of file.

apply plugin: 'com.android.application'  
  
android {  
….  
}  
  
dependencies {  
    compile fileTree(dir: 'libs', include: ['\*.jar'])  
    ….  
  // Add this line  
compile 'com.google.firebase:firebase-messaging:9.4.0' }  
  
// Add this line to the bottom of file  
apply plugin: 'com.google.gms.google-services'

After adding the required dependencies click on **Sync Now** option in the right side to sync our project with required files.

Once we are done with registering our app with firebase, we need to integrate firebase cloud messaging in our android application to send notifications.

**Integrate Firebase Cloud Messaging**

To integrate FCM (firebase cloud messaging) in android app, we need to create a new class file **FireBaseMessaging.java**in **\java\com.tutlane.pushnotifications** path and write the code like as shown below.

**FireBaseMessaging.java**

package com.tutlane.pushnotifications;  
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 com.google.firebase.messaging.FirebaseMessagingService;  
import com.google.firebase.messaging.RemoteMessage;  
  
/\*\*  
 \* Created by surdasari on 21-09-2017.  
 \*/  
  
public class FireBaseMessaging extends FirebaseMessagingService {  
    @Override  
    public void onMessageReceived(RemoteMessage rMsg) {  
        sendNotification(rMsg.getNotification());  
    }  
  
    private void sendNotification(RemoteMessage.Notification rNotfy) {  
        Uri soundUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE\_NOTIFICATION);  
        Intent rintent = new Intent(this, MainActivity.class);  
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, rintent, 0);  
        NotificationCompat.Builder builder = new NotificationCompat.Builder(this)  
                .setContentTitle(rNotfy.getTitle())  
                .setContentText(rNotfy.getBody())  
                .setAutoCancel(true)  
                .setSmallIcon(R.drawable.ic\_notification)  
                .setSound(soundUri)  
                .setContentIntent(pendingIntent);  
        NotificationManager notificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION\_SERVICE);  
        notificationManager.notify(0, builder.build());  
    }  
}

If we observe above code, we are extending our class (**FireBaseMessaging.java**) file behaviour using **FirebaseMessagingService** class. The **FirebaseMessagingService** class is the base class to communicate with firebase messaging and it provide a functionality to display the notifications automatically.

Now we need to register our newly created class files in **AndroidManifest.xml** file like as shown below

**AndroidManifest.xml**

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
    package="com.tutlane.pushnotifications">  
    <uses-permission android:name="android.permission.INTERNET"/>  
    <application  
        android:allowBackup="true"  
        android:icon="@mipmap/ic\_launcher"  
        android:label="@string/app\_name"  
        android:roundIcon="@mipmap/ic\_launcher\_round"  
        android:supportsRtl="true"  
        android:theme="@style/AppTheme">  
        <activity android:name=".MainActivity">  
            <intent-filter>  
                <action android:name="android.intent.action.MAIN" />  
                <category android:name="android.intent.category.LAUNCHER" />  
            </intent-filter>  
        </activity>  
  
        <service  
            android:name=".FireBaseMessaging">  
            <intent-filter>  
                <action android:name="com.google.firebase.MESSAGING\_EVENT"/>  
            </intent-filter>  
        </service>

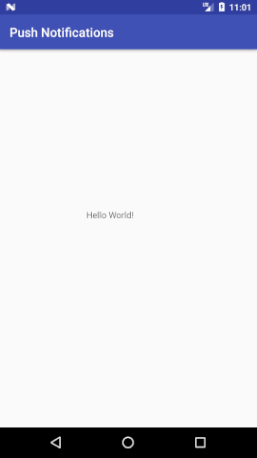
    </application>  
</manifest>

If we observe above code, we registered our **FireBaseMessaging** class in **AndroidManifest.xml** file and added a permission to access **INTERNET** because firebase cloud messaging required an INTERNET access to show the notifications.

Generally, during the launch of our [activity](https://www.tutlane.com/tutorial/android/android-activity-lifecycle), **onCreate()** callback method will be called by android framework to get the required layout for an [activity](https://www.tutlane.com/tutorial/android/android-activity-lifecycle).

**Output of Android Push Notification Example**

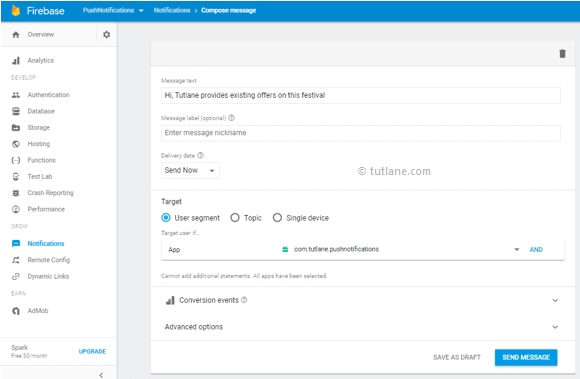
When we run above example using android virtual device (AVD) we will get a result like as shown below.



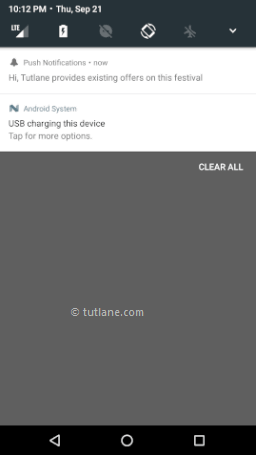
**Push Notifications using Firebase Console**

To publish or push notifications, we need to visit **Firebase Console** and open our app by clicking on it. After that, click on [Notifications](https://www.tutlane.com/tutorial/android/android-notifications-bigtextstyle-bigpicturestyle-inboxstyle) tab in the left panel. In case, if you visit first time, click on **Send Your First Message** otherwise click on **New Message**.

After that enter **Message Text**, select **Target** as **User Segment** then select your app from the list and click on **Send Message** like as shown below.



Once we click on Send Message button, we will get a notification in our app like as shown below.



This is how we can push notifications using firebase cloud messaging in android applications based on our requirements.