Prerequisites & Integration Steps

- **Prerequisites:**

  1. **google-services.json** file from Firebase Console
  2. **Legacy Server Key** from Firebase Console
  3. **App Id** from Smartech panel

  Note: Minimum SDK version (minSdkVersion) available in the build.gradle file of the app should be at least 16 (Jelly Bean) or above.

- **Integration Steps:**

  - **Setting up FCM in the application**
    1. Add google-services.json file in app directory of the project.
    2. Adding dependencies given below in **build.gradle file of the project**.

    ```
    classpath 'com.google.gms:google-services:3.1.0'
    classpath 'com.android.tools.build:gradle:3.1.3'
    ```

    3. Adding below code in the **build.gradle file of the app**.

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

  - **Adding dependencies in the build.gradle file of the app**

    ```
    implementation 'in.netcore.smartechfcm:smartech-fcm:1.1.5'
    implementation 'com.google.firebase:firebase-messaging:11.6.2'
    implementation 'com.google.code.gson:gson:2.8.0'
    implementation 'com.firebase:firebase-jobdispatcher:0.8.5'
    ```

  - **To register device for push notifications**
    To register the device for receiving push notifications from Smartech panel**,** add given snippet inside the **onCreate method of the Application class**.

    ```
    NetcoreSDK.register(this, <app_id>);
    ```

- **To capture user login**
  To capture login activity of the user, add given snippet inside the activity/fragment when the user gets logged in successfully.

```
NetcoreSDK.setIdentity(context, <unique_user_identity>);
NetcoreSDK.login(context);
```

- **To capture user logout**
  To capture logout activity of the user, add given snippet inside the activity/fragment when the user gets logged out successfully.

```
NetcoreSDK.logout(context);
NetcoreSDK.clearIdentity(context);
```

  Note: Avoid calling ‘clearIdentity’ method if one wants to track user activity even if user has logged out of the application.

- **To capture custom activity**
  To capture custom activity performed by the user, add given snippet as per the requirement.

```
NetcoreSDK.track(context, <event_name>, payload.toString());
```

  e.g.

```
JSONObject jsonObject = new JSONObject();
JSONObject payload = new JSONObject();
try {
        payload.put("name", "Nexus 5");
        payload.put("prid", 2);
        payload.put("price", 15000);
        payload.put("prqt", 1);
        jsonObject.put("payload", payload);
        NetcoreSDK.track(context, “Add To Cart”, jsonObject.toString());
}
catch (JSONException e) { e.printStackTrace(); }
```

  Note: Prior implementation of custom activity tracking with event id will be also supported by the SDK.

- **To capture user attributes**
  To capture and map user attributes, add given snippet as per the requirement.

  ```
  NetcoreSDK.profile(context, <profile_object>);
  ```

  e.g.

  ```
  JSONObject profile = new JSONObject();
  try {
          profile.put("NAME", "John Doe");
          profile.put("AGE", 35);
          profile.put("MOBILE", 9876543210);
          NetcoreSDK.profile(context, profile);
  }
  catch (JSONException e) { e.printStackTrace(); }
  ```

  Note: Use attribute name in capital letters as shown above.

- **To use custom push notification icon**
  SDK uses launcher icon for push notifications by default and in order to change it, use a custom icon by adding given snippet inside **onCreate method of the Application class**.

  ```
  NetcoreSDK.setPushIcon(context, <path_to_drawable_icon>);
  ```

  e.g.

  ```
  NetcoreSDK.setPushIcon(context, R.drawable.ic_push_icon);
  ```

  Note: The notification icon being used should strictly be in **.png format** as per Google's guidelines. Preferable size for the push notification icons are mentioned below.

drawable-mdpi	: 24 x 24
drawable-hdpi	: 36 x 36
drawable-xhdpi	: 48 x 48
drawable-xxhdpi	: 72 x 72
drawable-xxxhdpi	: 96 x 96

- **To fetch delivered push notifications**
  To fetch delivered push notifications, add given snippet as per the requirement.

```
JSONArray notifications = NetcoreSDK.getNotifications(context);

OR

JSONArray notifications = NetcoreSDK.getNotifications(context, <count>);
```

  Note: The method returns a 'JSONArray' of delivered push notifications for the user. **By default the method returns last 10 delivered notifications**.

- **To implement push notifications in existing FCM class**
  To use push notifications from Smartech panel along with existing set up of the FCM class, add given snippet **inside the FCM receiver class**.

```
boolean pushFromSmartech =
NetcoreSDK.handleNotification(context, remoteMessage.getData());
```

  Note: The method returns a boolean value.

  - Returns **true**, if the push notification is received from the Smartech panel. It will also render the push notification without any extra efforts further.

  - Return **false**, if the push notification is not received from the Smartech panel. In this case, handle the push notification at your end as per the requirement.

- **To opt out user from being tracked (GDPR Policy)**
  If the end user wants to opt out of being tracked, add given snippet as per the requirement.

```
NetcoreSDK.optOut(context, <boolean_flag>);
```

  Note: The method accepts a boolean value.

  - If an end user wants to opt out, the flag should be passed as **true**. Once the user opts out, SDK will not be able to track that particular user further and no communications will be received by that user.

  - If an end user wants to opt in, the flag should be passed as **false**. Once the user opts in, SDK will be able to track that particular user further and next communications will be received by that user.

- **To implement location tracking**
  In order to track user location and pass it further to Smartech, add given snippet as per the requirement.

  ```
  NetcoreSDK.setUserLocation(context, <double_lat>, <double_long>);
  ```

  Note: The method mentioned above accepts 3 parameters including context, latitude & longitude. **Data type of 'latitude' & 'longitude' should compulsorily be 'Double'**. In case if any parameter is null, SDK will not be able to persist user location.

- **To set user identity**
  In order to identify a user, set unique user identity by adding given snippet as per the requirement.

  ```
  NetcoreSDK.setIdentity(context, <unique_user_identity>);
  ```

- **To clear user identity**
  In order to wipe out user identity from the SDK, add given snippet as per the requirement.

  ```
  NetcoreSDK.clearIdentity(context);
  ```

- **To set existing FCM token**
  To set existing FCM token of the application to the SDK, **add given snippet just before 'register' method in the Application class of the app**.

  ```
  NetcoreSDK.setPushToken(context, <token_string>);
  ```

  e.g.

  ```
  NetcoreSDK.setPushToken(context, <token_string>);
  NetcoreSDK.register(this, <app_id>);
  ```

- **To get GUID of the user**
  To obtain GUID of the user from the SDK, add given snippet as per the requirement.

  ```
  NetcoreSDK.getGUID(context);
  ```

- **To get FCM token of the user**
  To obtain the FCM token of the user from the SDK, add given snippet as per the requirement.

```
NetcoreSDK.getPushToken(context);
```

- **To implement deep link in the application**
  To implement deep link in the application, add given snippet **inside AndroidManifest.xml file with in the Activity Tag**.

```
<intent-filter>
  <action android:name = "android.intent.action.VIEW"/>
  <category android:name = "android.intent.category.DEFAULT"/>
  <category android:name =
"android.intent.category.BROWSABLE"/>
  <data android:scheme = "<scheme>" android:host= "<host>"/>
</intent-filter>
```

e.g.

```
<intent-filter>
  <action android:name = "android.intent.action.VIEW"/>
  <category android:name = "android.intent.category.DEFAULT"/>
  <category android:name =
"android.intent.category.BROWSABLE"/>
  <data android:scheme = "smartech" android:host= "products"/>
</intent-filter>
```