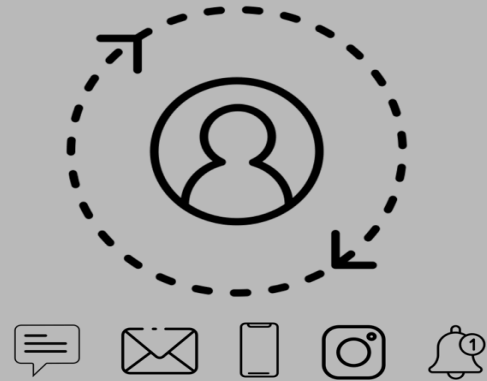


Appice

AI & ML DRIVEN

CUSTOMER DATA & ENGAGEMENT PLATFORM



Integration document

Flutter Development

[This document will help you integrate the appICE SDK in your Android/iOS Projects.]

[Table of Contents](#)

Setting up your app on //appice.io	4
Sign-up	4
Setup your App	4
Initializing appiceflutter plugin in your project	4
Android	5
iOS	9
Sample code	12
Verify integration on appICE panel	23

Setting up your app on //appice.io

Sign-up

You need to sign-up and create an account with appICE.io.

1. Visit <https://panel.appice.io/login> and click on the Register button.
2. This brings you to the “Sign up” page.
3. Provide your Name, Email address and your chosen password and create your account.
4. Login using your username and password credentials once your account is approved by the administrator.

Setup your App

To start the process of integrating appICE in your app, you will first need to setup your app on the appICE dashboard.

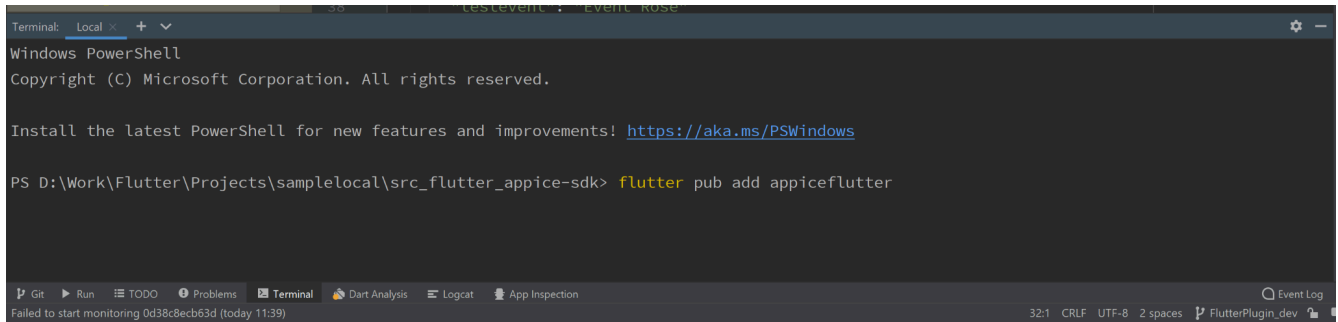
1. Click on Setup New App in the left panel.
2. Provide the link to your mobile app on the Google Play Store as well as Apple store if you have both versions of your app.
Click Next to go to the next page.
3. Now you have access to the SDK's for Flutter including the instructions for integration. To make it easier, we have also provided the option to provide your developer's email address so that those instructions can be emailed to them directly.
Click Next to go to the next page.
4. Now you are all ready to start receiving data from your mobile app once the developer completes the integration and publishes the app again on the app store.

Initializing appiceflutter plugin in your project

Open Terminal and add

Create a new project using android studio.

To add appiceflutter plugin in your project, create a new flutter project and open it in AndroidStudio.



Add Plugin

Open terminal and add 3 plugins:

- ## 1. Add appice plugin

```
flutter pub add appiceflutter
```

Android

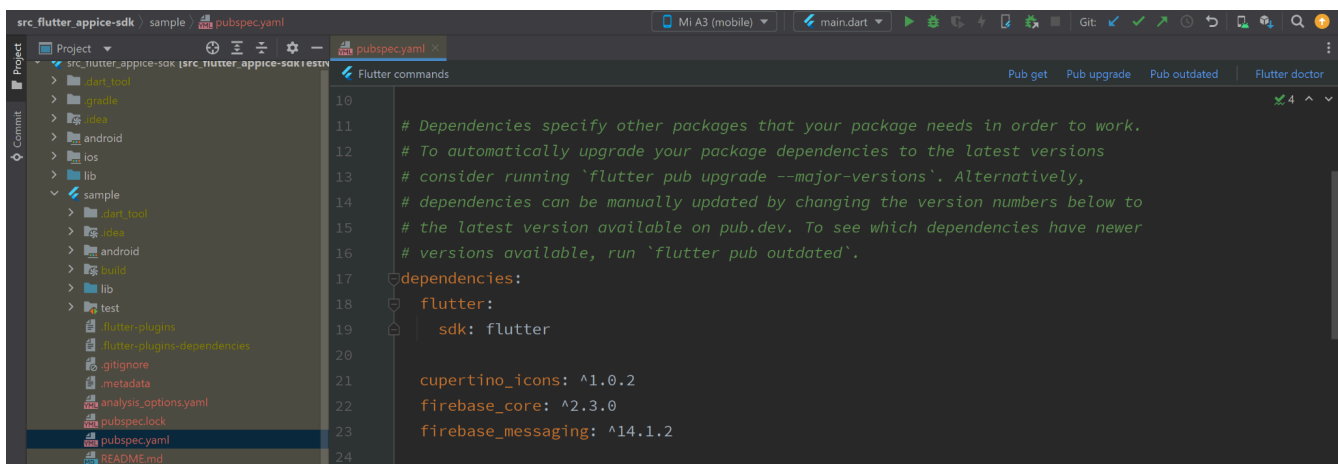
- ## 2. Add firebase messaging plugin

```
flutter pub add firebase_messaging
```

- ### 3. Add firebase core plugin

```
flutter pub add firebase_core
```

This will add a line like this to your package's pubspec.yaml (and run an implicit flutter pub get):



AD_ID Permission

Apps targeting Android 13 and above, you must add the `com.google.android.gms.permission.AD_ID` permission in the `AndroidManifest.xml`

`..\android\app\src\main\AndroidManifest.xml`

```
<uses-permission android:name="com.google.android.gms.permission.AD_ID"/>
```

Add Multidex Support

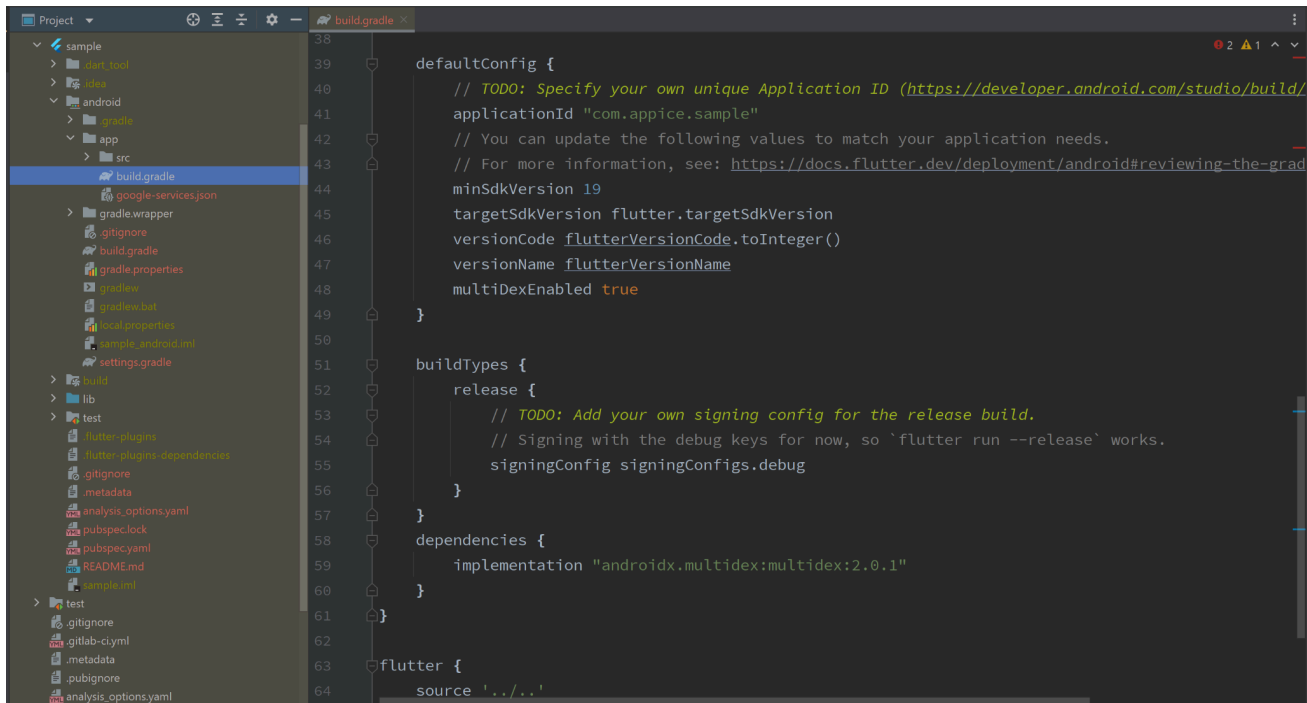
Open project > app > build.gradle

Inside defaultConfig, add this:

`multiDexEnabled true`

Inside dependencies, add this:

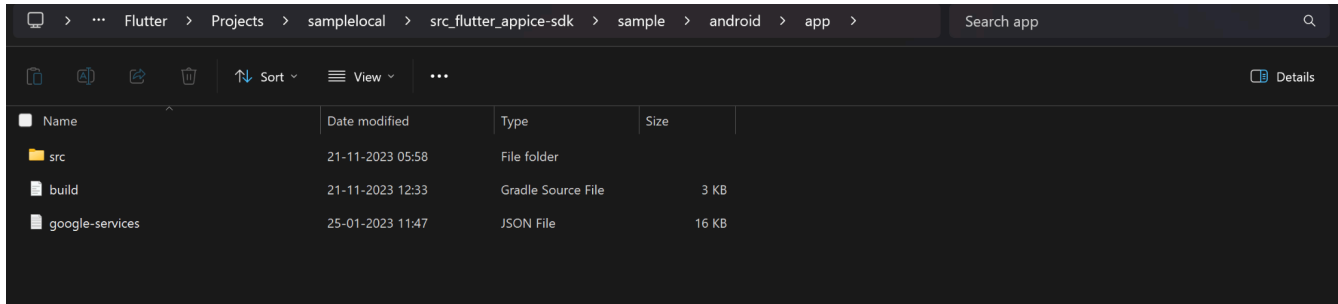
`implementation "androidx.multidex:multidex:2.0.1"`



Add Firebase Google services.json

Add google services.json file into the given location

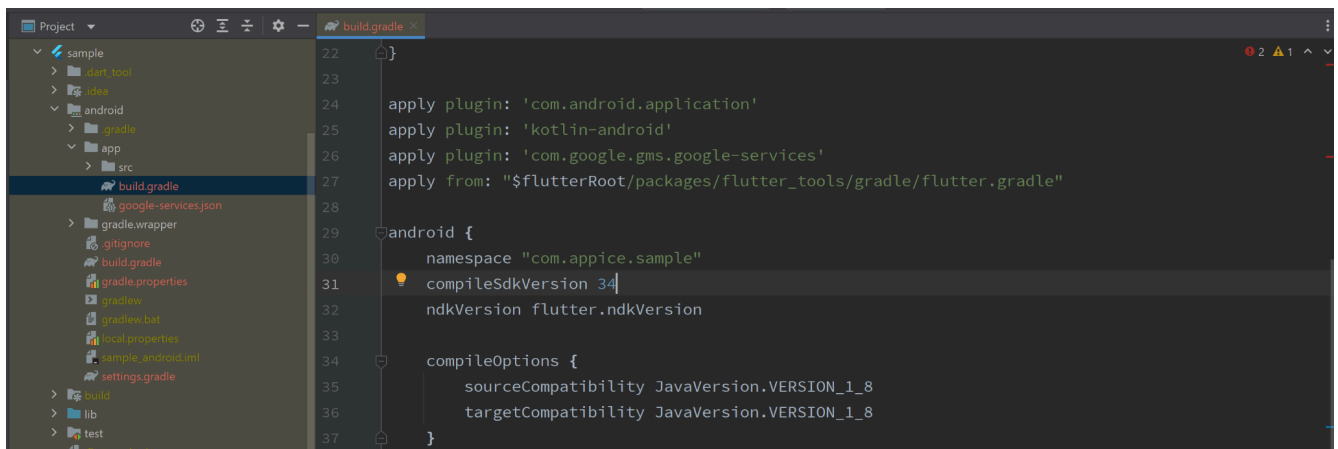
`..sample\android\app`



Change compileSdkVersion to 34

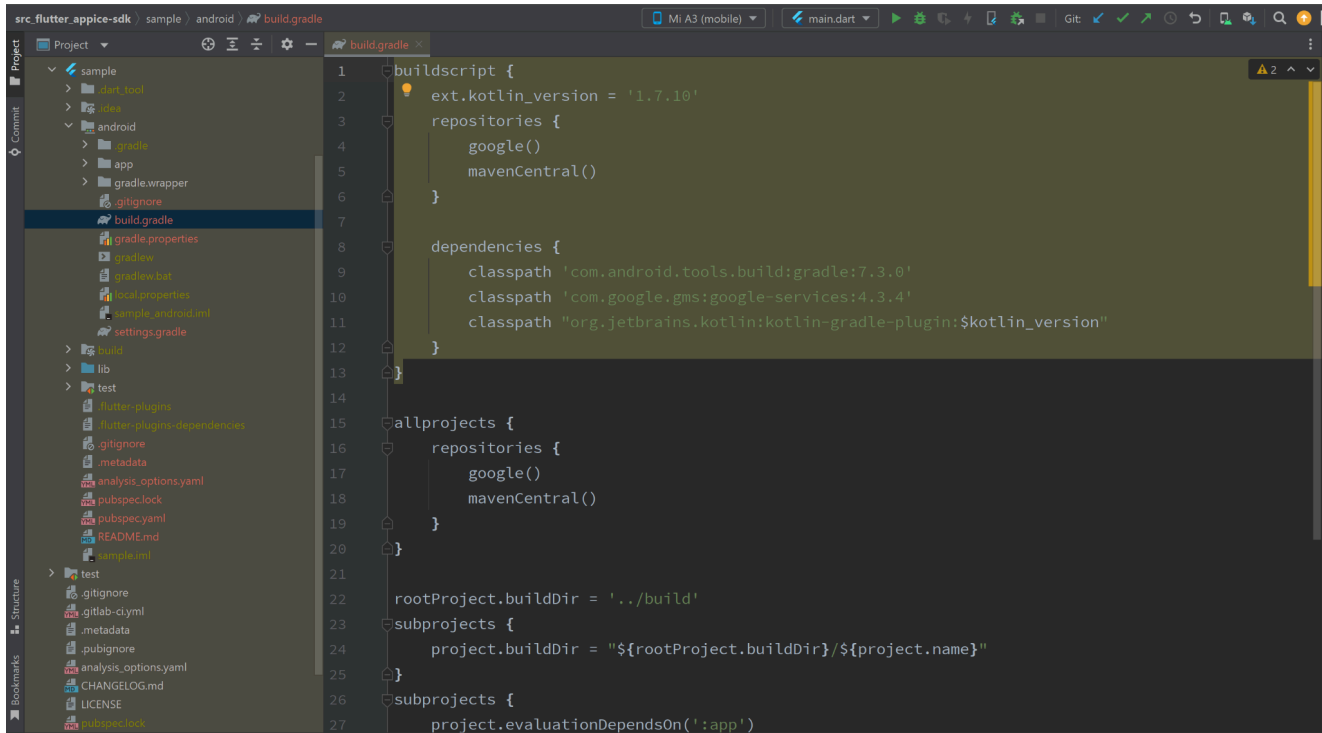
Change the compileSdkVersion into the given location

\\sample\\android\\app\\build.gradle
compileSdkVersion 34



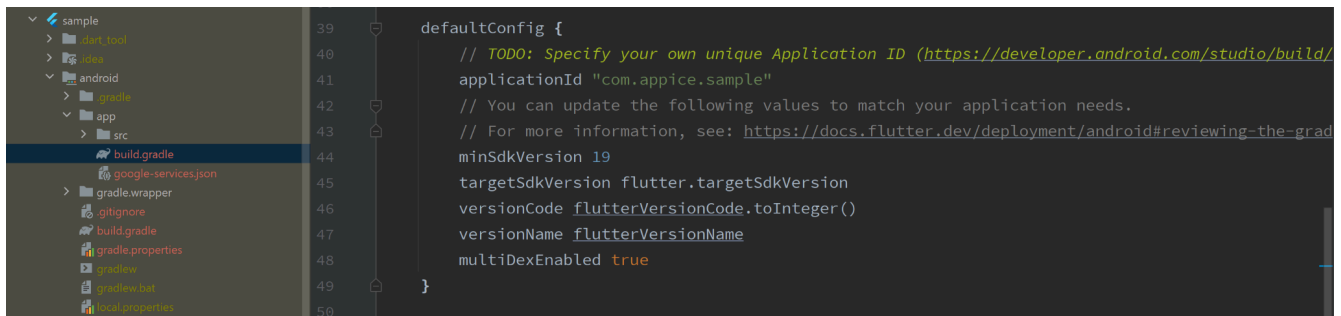
Add classpath dependency

\\sample\\android\\build.gradle
classpath 'com.google.gms:google-services:4.3.4'

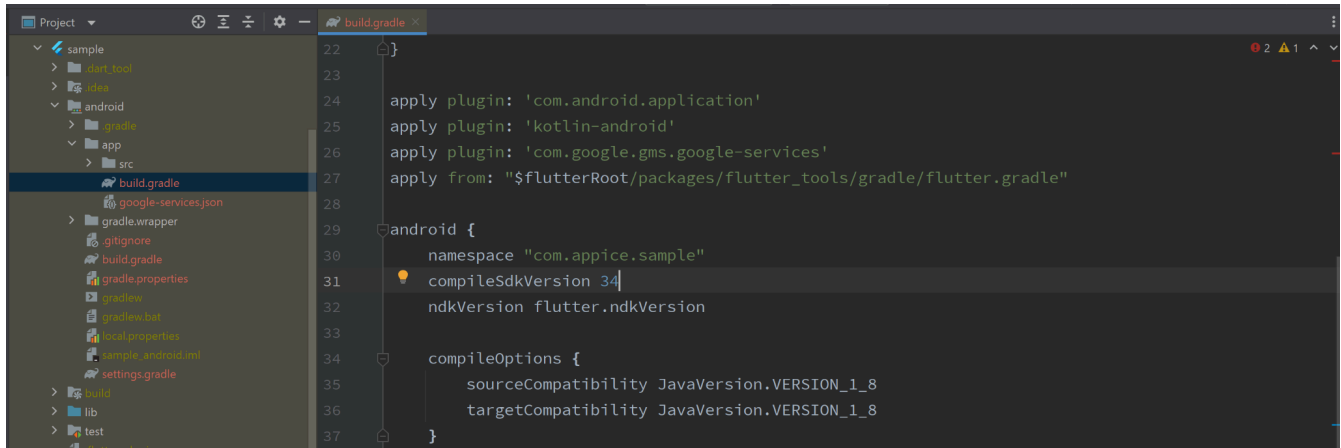


Apply plugin and change minSdkVersion 19

\\sample\\android\\app\\build.gradle
minSdkVersion 19



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



iOS

Step 1: open PodFile updated pod file as

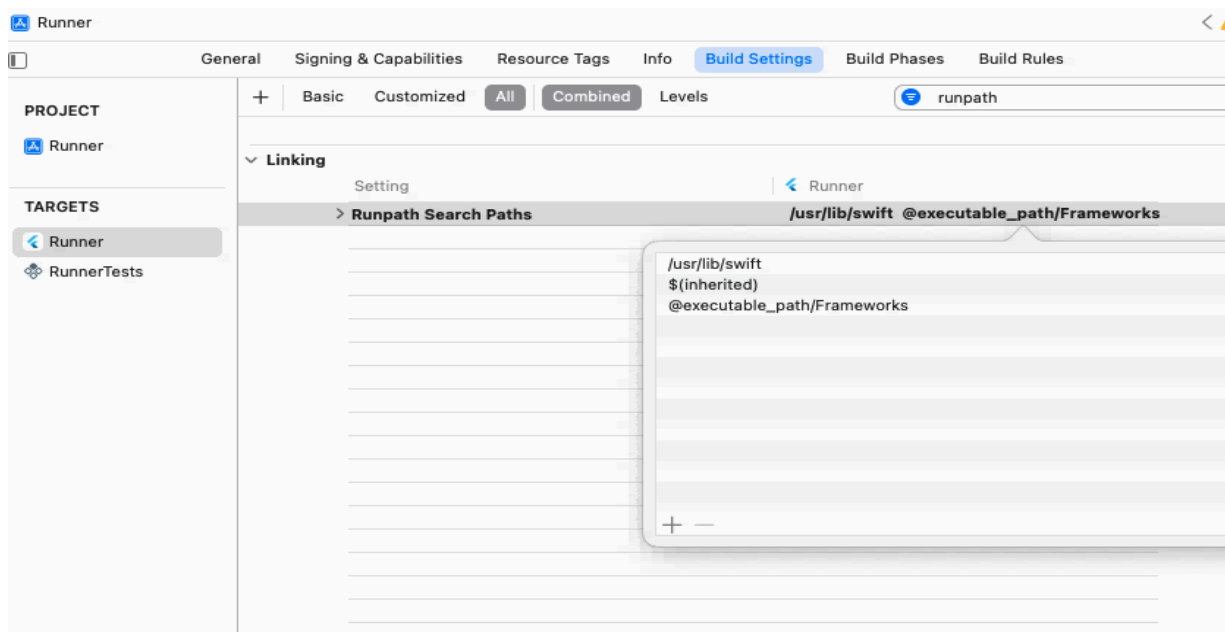
```
target 'Runner' do
  use_frameworks! :linkage => :static
  use_modular_headers!
```

```
  flutter_install_all_ios_pods File.dirname(File.realpath(__FILE__))
end
```

Step 2: pod deintegrate

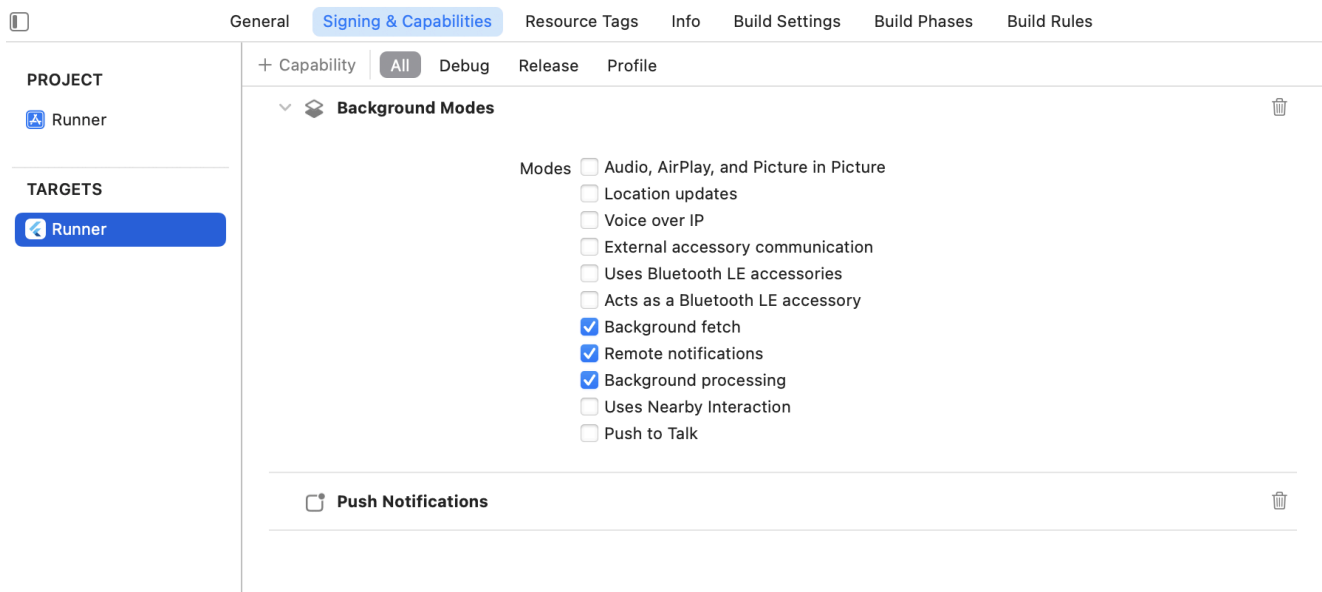
Step 3: pod install

Step 4: In xcode build settings- add `/usr/lib/swift`



Push Notification setting for IOS

1. Setting enable in capability section of app



2. Add below code in AppDelegate.m file

```
- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    [GeneratedPluginRegistrant registerWithRegistry:self];
    UNUserNotificationCenter *center =
        [UNUserNotificationCenter currentNotificationCenter];
    center.delegate = self;

    return [super application:application didFinishLaunchingWithOptions:launchOptions];
}

-(void)application:(UIApplication*)application
didRegisterForRemoteNotificationsWithDeviceToken:(NSData*)deviceToken
{
    NSLog(@"token is : %@",deviceToken);
    [AppiceflutterPlugin setPushToken:deviceToken];
}

-(void)application:(UIApplication *)application didReceiveRemoteNotification:(NSDictionary *)userInfo
fetchCompletionHandler:(void (^)(UIBackgroundFetchResult))completionHandler{
    NSLog(@"received payload =%@",userInfo);
    [AppiceflutterPlugin pushNotificationReceived:userInfo];
}

- (void)userNotificationCenter:(UNUserNotificationCenter *)center
didReceiveNotificationResponse:(UNNotificationResponse *)response
withCompletionHandler:(void (^)(void))completionHandler {
```

```
NSLog(@"received payload clicked=%@",response.notification.request.content.userInfo);
```

```
[AppiceflutterPlugin pushNotificationClicked:response.notification.request.content.userInfo];  
completionHandler();  
}
```

```
- (void)userNotificationCenter:(UNUserNotificationCenter *)center  
willPresentNotification:(UNNotification *)notification  
withCompletionHandler:  
    (void (^)(UNNotificationPresentationOptions options))  
    completionHandler {  
completionHandler(UNNotificationPresentationOptionSound | UNNotificationPresentationOptionBadge |  
    UNNotificationPresentationOptionAlert);  
}
```

3. Add below UNUserNotificationCenterDelegate in AppDelegate.h file

```
@interface AppDelegate : FlutterAppDelegate<UNUserNotificationCenterDelegate>
```

Initialise flutter SDK

Now in your Dart code, you can use:

```
import 'package:appiceflutter/appiceflutter.dart';
```

main.dart

```
// Initialize the AppICE SDK
```

```
Future<void> appICEInit() async {  
    final List<String> certs = <String>[""];  
    await Appiceflutter.initSDK(  
        "app_id",  
        "app_key",  
        "api_key",  
        "region",  
        "baseurl",  
        certs);  
}
```

```
// Set user information
```

```
Future<void> setUser() async {  
    Map map = <dynamic, dynamic>{};  
    map[Appiceflutter.name] = 'Test';  
    map[Appiceflutter.email] = 'Test@email';  
}
```

```
map[Appiceflutter.phone] = '45674557455';
map[Appiceflutter.age] = 23;
map[Appiceflutter.gender] = 'M';
map[Appiceflutter.educationType] = 'Education';
Appiceflutter.setUser(map);
print(map);
}
```

// Set user IDs

```
Future<void> setUserId() async {
  List<String> arr = <String>[];
  arr.add("354345YGJKSH");
  arr.add("YHGJKSGH787JH");
  Appiceflutter.setUserId(arr);
}
```

//Receive the payload from PUSH Notification

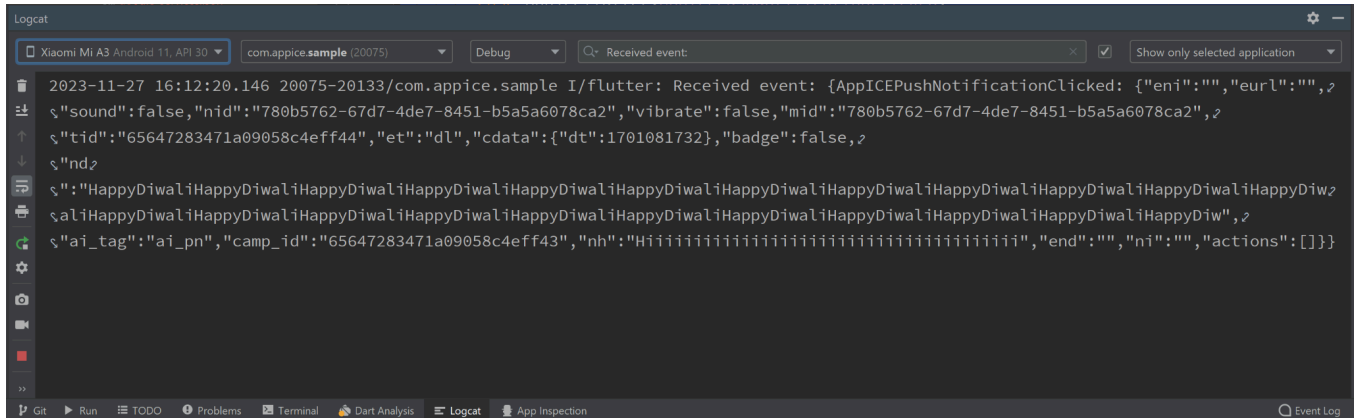
@override

```
void initState() {
  super.initState();
  //initialise the SDK
  applCEInit();
}
```

//Receive the payload from PUSH and INAPP

```
Appiceflutter.myChannel.setMethodCallHandler((call) async {
  switch (call.method) {
    case Appiceflutter.AppICEPushNotificationClicked:
      print('Received event: ${call.arguments}');
      //Appiceflutter.onPushClicked(call.arguments);
      break;
    case Appiceflutter.AppICEInAppClicked:
      print('Received event from AppICEInAppClicked : ${call.arguments}');
      //Appiceflutter.onPushClicked(call.arguments);
      break;
  }
});
```

}



Sample code

Copy and paste the below code to your main.dart:

```
import 'dart:async';
import 'package:appiceflutter/appiceflutter.dart';
import 'package:appiceflutter/appiceinboxmessage.dart';
import 'package:flutter/material.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatefulWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {

  @override
  void initState() {
    super.initState();
    //initialise the SDK
    appICEInit();

    //Receive the payload from PUSH and INAPP
    Appiceflutter.myChannel.setMethodCallHandler((call) async {
      switch (call.method) {
        case Appiceflutter.AppICEPushNotificationClicked:
          print('Received event: ${call.arguments}');
          //Appiceflutter.onPushClicked(call.arguments);
          break;
      }
    });
  }
}
```

```
case Appiceflutter.AppICEInAppClicked:  
  print('Received event from AppICEInAppClicked : ${call.arguments}');  
  //Appiceflutter.onPushClicked(call.arguments);  
  break;
```

```
  }  
});  
}
```

// Record an Event

```
Future<void> sendEvent() async {  
  Map<String, String> map = {  
    "onClick": "true",  
    "name": "Rahulk",  
    "testevent": "Event Rose"  
  };  
  await Appiceflutter.tagEvent('EventName', map);  
  print("inside send event");  
}
```

```
Future<void> appICEInit() async {  
  final List<String> certs = <String>[""];  
  await Appiceflutter.initSDK(  
    "5bebe93c25d705690ffbc758",  
    "9e9ec60197c8373a11ac15ce4dae80e973608ab2",  
    "e5aa4b293f3188fac9cb1a4017963604",  
    "US",  
    "https://devapi.appice.io",  
    certs);  
}
```

// Set user information

```
Future<void> setUser() async {  
  Map map = <dynamic, dynamic>{};  
  map[Appiceflutter.name] = 'Test';  
  map[Appiceflutter.email] = 'Test@email';  
  map[Appiceflutter.phone] = '45674557455';  
  map[Appiceflutter.age] = 23;  
  map[Appiceflutter.gender] = 'M';  
  map[Appiceflutter.educationType] = 'Education';  
  Appiceflutter.setUser(map);  
  print(map);  
}
```

// Set user IDs

```
Future<void> setUserId() async {  
  List<String> arr = <String>[];  
  arr.add("354345YGJKSH");
```

```
arr.add("YHGJKSGH787JH");
Appiceflutter.setUserId(arr);
print('UserId Set');
}

// Set custom variable with string
Future<void> setCustommVarString() async {
  await Appiceflutter.setCustomVariableWithString('mobNo', 'Developer');
}

// Receive push notification
Future<void> pushNotificationReceived() async {
  await Appiceflutter.pushNotificationReceived("message");
}

// Set custom variable with boolean
Future<void> setCustommVarBoolean() async {
  await Appiceflutter.setCustomVariableWithBoolean('isAwesome', true);
}

// Set custom variable with integer
Future<void> setCustommVarInt() async {
  await Appiceflutter.setCustomVariableWithInteger('device_id', 1234567890);
}

// Stop ApplICE context
Future<void> stopContext() async {
  await Appiceflutter.stopContext();
}

// Set custom variable with long
Future<void> setCustommVarLong() async {
  await Appiceflutter.setCustomVariableWithLong('myLong', 1254343423243567892);
}

// Set custom variable with float
Future<void> setCustommVarFloat() async {
  await Appiceflutter.setCustomVariableWithFloat('myFloat', 12.0);
}

Future<void> getSessionTimeout() async {
  var data = await Appiceflutter.getSessionTimeout();
  // print("getSessionTimeout Value Received: $data");
}

// Validate ApplICE integration
Future<void> validateIntegration() async {
  var flg = await Appiceflutter.validateIntegration();
```

```
    print('$flg');
  }

// Set session timeout
Future<void> setSessionTimeout() async {
  await Appiceflutter.setSessionTimeout(18000);
  print('Timeout Set');
}

// Set as a test device
Future<void> setAsTestDevice() async {
  await Appiceflutter.setAsTestDevice(true);
}

// Get test device status
Future<void> getIsTestDevice() async {
  await Appiceflutter.getIsTestDevice();
}

// Remove as a test device
Future<void> removeAsTestDevice() async {
  await Appiceflutter.removeAsTestDevice();
}

// Open Play Service update
Future<void> openPlayServiceUpdate() async {
  await Appiceflutter.openPlayServiceUpdate();
}

// Get AppICE SDK version
Future<void> getSdkVersion() async {
  var intv = await Appiceflutter.getSdkVersion();
  print('$intv');
}

// Get AppICE SDK int version
Future<void> getSdkIntVersion() async {
  await Appiceflutter.getSdkIntVersion();
}

// Set device ID
Future<void> setDeviceId() async {
  String id = "abc";
  await Appiceflutter.setDeviceId(id);
  print('Device id set');
}

// Check if it's a geofence campaign
```



```
Future<void> isGeofenceCamp() async {
  String id = "454HHFkljl";
  await Appiceflutter.isGeofenceCamp(id);
}

// Get device ID
Future<void> getDeviceId() async {
  var data=await Appiceflutter.getDeviceId();
  print("Device id Received: $data");
}

// Get Android ID
Future<void> getAndroidId() async {
  await Appiceflutter.getAndroidId();
}

// Get AppICE SDK app key
Future<void> getAppKey() async {
  await Appiceflutter.getAppKey();
}

// Get AppICE SDK API key
Future<void> getApiKey() async {
  await Appiceflutter.getApiKey();
}

// Get current context
Future<void> getCurrentContext() async {
  await Appiceflutter.getCurrentContext();
}

// Get AppICE SDK app ID
Future<void> getAppId() async {
  var appid = await Appiceflutter.getAppId();
  print('$appid');
}

// Set alias
Future<void> setAlias() async {
  await Appiceflutter.setAlias("False_Name");
}

// Synchronize AppICE inbox
Future<void> synchronizeInbox() async {
  try {
    Map<String, dynamic> result = await Appiceflutter.synchronizeInbox(5000); // Example timeout
    value
    print("Received values from synchronizeInbox: $result");
  }
}
```

```
} catch (e) {
  print("Error in handleSynchronizelInbox: $e");
}
}

// Get message by campaign ID with user ID
Future<void> getMessageByCampaignIdWithUserId() async {
  String msgId = '123';
  String usrId = '123';
  await Appiceflutter.getMessageByCampaignIdWithUserId(msgId, usrId);
}

// Get message by campaign ID
Future<void> getMessageByCampaignId() async {
  String msgId = '63bd1779d58e53bc7333c42b';
  var data=await Appiceflutter.getMessageByCampaignId(msgId);
  print("Received getMessageByCampaignId : $data");
}

// Get message count
Future<void> getMessageCount() async {
  int msgType = 1; // 1=all
  var data = await Appiceflutter.getMessageCount(msgType);
  print("Received message count : $data");
}

// Get message count data with user ID
Future<void> getMessageCountDataWithUserId() async {
  int msgType = 123;
  List<String> arr = ['myUser'];
  await Appiceflutter.getMessageCountDataWithUserId(msgType, arr);
}

// Get inbox message data
Future<void> getInboxMessageData() async {
  int msgType = 1;
  List<AppICEInboxData>? list =
  await Appiceflutter.getInboxMessageData(msgType);
}

// Get inbox message for ID
Future<void> getInboxMessageForId() async {
  String messageId = '9e16e82ba1af3946ce92b44f4c50a3579e2e121b';
  var data=await Appiceflutter.getInboxMessageForId(messageId);
  print("getInboxMessageForId received: $data");
}

// Update inbox message
```

```
Future<void> updateInboxMessage() async {
  int Type = 1;
  String messageId = 'GivA5k6VJqqvK3aZRXyez2qgoAOLzJvCui';
  var data=await Appiceflutter.updateInboxMessage(Type, messageId);
  print("Update Inbox received: $data");
}

// Get alias
Future<void> getAlias() async {
  var data=await Appiceflutter.getAlias();
  print("getAlias received: $data");
}

// Get user data
Future<void> getUser() async {
  try {
    Map<String, dynamic>? userData = await Appiceflutter.getUser();
    print("User data received: $userData");
  } catch (e) {
    print("Error fetching user data: $e");
  }
}

// Set child ID
Future<void> setChildId() async {
  String child = "CHILD1234";
  await Appiceflutter.setChildId(child);
}

// Set installer
Future<void> setInstaller() async {
  String install = "INSTALL2344";
  await Appiceflutter.setInstaller(install);
}

// Get custom variable
Future<void> getCustomVariable() async {
  try {
    Map<String, dynamic>? userData = await Appiceflutter.getCustomVariable();
    print("CustomVariable received: $userData");
  } catch (e) {
    print("Error fetching CustomVariable: $e");
  }
}

// Remove custom variable
Future<void> removeCustomVariable() async {
  String cvar = "Remove Custom Variable";
```

```
    await Appiceflutter.removeCustomVariable(cvar);
}

// Set small icon path
Future<void> setSmallIcon() async {
    String path = "path to small icon";
    await Appiceflutter.setSmallIcon(path);
}

// Handle open link URL
Future<void> handleOpenLinkUrl() async {
    String path = "https://panel.appice.io/login";
    await Appiceflutter.handleOpenLinkUrl(path);
}

// Check if payload has an external URL
Future<void> isPayloadHaveExternalUrl() async {
    String path = "https://panel.appice.io/login";
    await Appiceflutter.isPayloadHaveExternalUrl(path);
}

// Check if device is registered
Future<void> isDeviceRegistered() async {
    await Appiceflutter.isDeviceRegistered("Device Registered");
}

// Check if SSL pinning is enabled
Future<void> isSSLPinningEnabled() async {
    await Appiceflutter.isSSLPinningEnabled();
}

// Get SDK configuration information
Future<void> getSdkConfigInfo() async {
    await Appiceflutter.getSdkConfigInfo("getSdkConfigInfo");
}

// Get limited tracking status
Future<void> getIsLimitedTracking() async {
    await Appiceflutter.getSdkConfigInfo("getIsLimitedTracking");
}

// Get custom device ID
Future<void> customDeviceId() async {
    await Appiceflutter.getSdkConfigInfo("customDeviceId");
}

// Get advertising ID
Future<void> getAdvertisingId() async {
```

```
    await Appiceflutter.getAdvertisingId();
  }

// Set session ID
Future<void> setSessionID() async {
  await Appiceflutter.setSessionID();
}

// Get base URL
Future<void> getBaseUrl() async {
  await Appiceflutter.getBaseUrl();
}

// Get user ID
Future<void> getUserId() async {
  var data = await Appiceflutter.getUserId();
}

// Get installer information
Future<void> getInstaller() async {
  await Appiceflutter.getInstaller();
}

// Set referrer
Future<void> setReferrer() async {
  await Appiceflutter.setReferrer("myreferrer");
}

// Get referrer information
Future<void> getReferrer() async {
  await Appiceflutter.getReferrer();
}

// Set install referrer
Future<void> setInstallReferrer() async {
  await Appiceflutter.setInstallReferrer("install_referrrer");
}

// Get install referrer information
Future<void> getInstallReferrer() async {
  await Appiceflutter.getInstallReferrer();
}

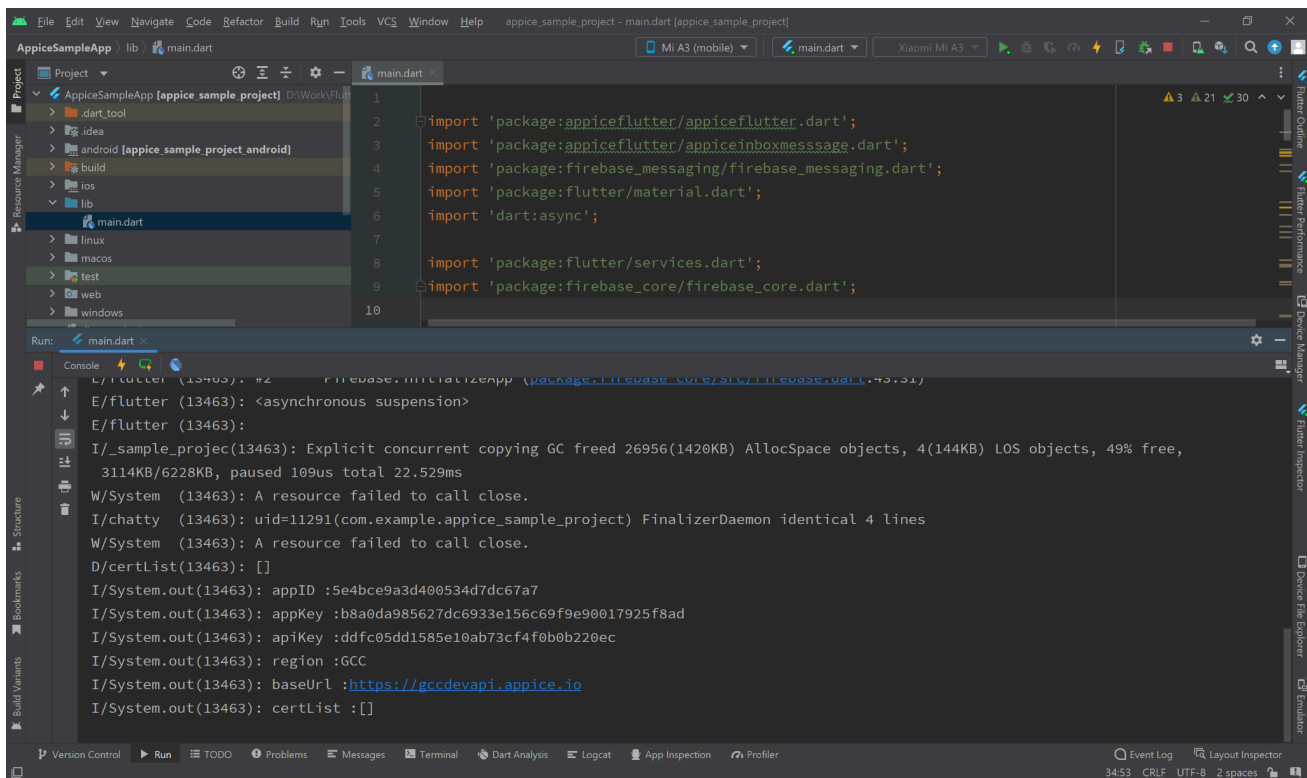
// Check if Semusi Sensing is enabled
Future<void> isSemusiSensing() async {
  await Appiceflutter.isSemusiSensing();
}
```

```
Widget cButton(String btnName,Function()? method){
  return
  Container(
    margin: const EdgeInsets.all(2),
    child: SizedBox(
      width: double.infinity ,
      child: TextButton(
        child: Text(btnName, style: const TextStyle(fontSize: 15.0)),
        onPressed: method
      ),
    ),
  );
}

@override
Widget build(BuildContext context) {
  return MaterialApp(
    home: Scaffold(
      appBar: AppBar(title: const Text('Sample App')),
      body: Center(child: ListView(children: <Widget>[
        cButton("INIT", () => appICEInit()),
        cButton("STOP CONTEXT", () => stopContext()),
        cButton("getAppId", () => getAppId()),
        cButton("TagEvent", () => sendEvent()),
        cButton("SetUser", () => setUser()),
        cButton("getUSer", () => getUser()),
        cButton("SetUserId", () => setUserId()),
        cButton('getUserId', () => getUserId()),
        cButton("setCustomString", () => setCustommVarString()),
        cButton("setCustomBoolean", () => setCustommVarBoolean()),
        cButton("setCustomInt", () => setCustommVarInt()),
        cButton("setCustomLong", () => setCustommVarLong()),
        cButton("setCustomFloat", () => setCustommVarFloat()),
        cButton("getSessionTimeout", () => getSessionTimeout()),
        cButton("setSessionTimeout", () => setSessionTimeout()),
        cButton("setAsTestDevice", () => setAsTestDevice()),
        cButton("removeAsTestDevice", () => removeAsTestDevice()),
        cButton("getIsTestDevice", () => getIsTestDevice()),
        cButton("isSemusiSensing", () => isSemusiSensing()),
        cButton("getSdkVersion", () => getSdkVersion()),
        cButton("validateIntegration", () => validateIntegration()),
        cButton("Synchronise Inbox", () => synchronizeInbox()),
        cButton('getMessageCount', () => getMessageCount()),
        cButton('getMessageByCampaignId', () => getMessageByCampaignId()),
        cButton('getInboxMessageData', () => getInboxMessageData()),
        cButton('getInboxMessageForId', () => getInboxMessageForId()),
        cButton('updateInboxMessage', () => updateInboxMessage()),
        cButton('openPlayServiceUpdate', () => openPlayServiceUpdate()),
```

```
cButton('setDeviceId', () => setDeviceId()),
cButton('getDeviceId', () => getDeviceId()),
cButton('setAlias', () => setAlias()),
cButton('getAlias', () => getAlias()),
cButton('getAndroidId', () => getAndroidId()),
cButton('getAppKey', () => getAppKey()),
cButton('getAPIKey', () => getApiKey()),
cButton('getCurrentContext', () => getCurrentContext()),
cButton('setChildId', () => setChildId()),
cButton('getReferrer', () => getReferrer()),
cButton('setReferrer', () => setReferrer()),
cButton('setInstallReferrer', () => setInstallReferrer()),
cButton('getInstallReferrer', () => getInstallReferrer()),
cButton('setInstaller', () => setInstaller()),
cButton('getInstaller', () => getInstaller()),
cButton('getCustomVariable', () => getCustomVariable()),
cButton('removeCustomVariable', () => removeCustomVariable()),
cButton('setSmallIcon', () => setSmallIcon()),
cButton('isDeviceRegistered', () => isDeviceRegistered()),
cButton('setSessionID', () => setSessionID()),
cButton('getBaseUrl', () => getBaseUrl()),
cButton('pushNotificationReceived', () => pushNotificationReceived()),
cButton('isSSLPinningEnabled', () => isSSLPinningEnabled()),
cButton('handleOpenLinkUrl', () => handleOpenLinkUrl()),
cButton('isPayloadHaveExternalUrl', () => isPayloadHaveExternalUrl()),
cButton('isGeofenceCamp', () => isGeofenceCamp()),
cButton('getSdkConfigInfo', () => getSdkConfigInfo()),
cButton('getIsLimitedTracking', () => getIsLimitedTracking()),
cButton('getAdvertisingId', () => getAdvertisingId()),
cButton('customDeviceId', () => customDeviceId()),
]
)),
));
}
```

Logs when you run the application



Verify integration on appICE panel

1. Login to panel.appice.io to see these values. Click on your app and go to its dashboard.
2. Click on 'Active Users → View Detail' to see details of your phone app.
3. Clicking on 'View Detail' takes you to the App Users page. Search your phone's AdvertisingId in the Search box to search for your phone.
4. Click on the searched device Id to see details:
It shows like this.

Home

Dashboard

Setup Apps

Users

Acquisition

Activities

Engagement

Churn

Documentation

7a48b968-50f3-4d5e-9bc1-ed8d0b634aee

Demographics

Gender : N/A

User Attributes

Variable	Value
Model	Xiaomi MI A3
Carrier	Vodafone IN
App Version	1.0.0

Competing Apps

Interests

Recent Activity

01/02/23 13:11:34

Session_Start

01/02/23 12:58:28

Campaign_Received

campid : 63ae76d48502b09b94891832

01/02/23 12:58:28

Campaign_Received

campid : 63aebb2c67ae0cd53588e403

01/02/23 12:58:28

Campaign_Received

campid : 63ae75fd8502b09b9489182f

01/02/23 12:58:28

Session_Start

01/02/23 12:58:28

Campaign_Received

campid : 63ae75fd8502b09b9489182f

01/02/23 12:58:28

Campaign_Received

campid : 63ae76d48502b09b94891832

Recency

Seen 1 mins 34 secs ago

Monetary

0

Campaign Engagement

0

CAC / LTV

0