**Galerieslafayette Mobile App Analytics Plan**

**PREPARED FOR**

**Galerieslafayette - Android & iOS Applications**

30 June 2025

Version 1.0

**ABOUT THIS DOCUMENT**

This document is written specifically for Galerieslafayette India Mobile App Analytics Tracking for Android and iOS Application. No part of this document, in either substance or meaning, shall be used or distributed to another vendor for this or any other project without prior permission from Galerieslafayette India.

**DOCUMENT VERSION**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Description** |
| 30 June 2025 | 1.0 | Siva | * Initial Draft |

**Adobe Documentation Links:**   
  
**Get the Adobe Experience Platform Mobile SDK:** https://developer.adobe.com/client-sdks/home/getting-started/get-the-sdk/  
  
**Lifecycle for Edge Network Implementation:** https://developer.adobe.com/client-sdks/edge/lifecycle-for-edge-network/

**Send events to Edge Network (requires Edge Network extension):** https://developer.adobe.com/client-sdks/home/getting-started/track-events/#send-events-to-edge-network

**TABLE OF CONTENTS**

[● Mobile App Analytics Install Instructions](#_Toc202253869)

[a. Android:](#_Toc202253870)

[b. iOS](#_Toc202253871)

[● Configure and Install Lifecycle for Edge Network](#_Toc202253872)

[Configure a Rule to forward Lifecycle data to Platform](#_Toc202253873)

[● Track State:](#_Toc202253874)

[c. Android](#_Toc202253875)

[d. iOS](#_Toc202253876)

[● Analytics Variables for Track Action (Custom Tracking):](#_Toc202253877)

[● Sign in Success: Track the “User Signin Success” button Clicks](#_Toc202253878)

[● Add to Favorite Items Added](#_Toc202253879)

[● Folder Creation (My Folders) – “Add all items” listed in the user folders with SKU numbers and product names](#_Toc202253880)

[● Scheduled Appointments Success](#_Toc202253881)

[● Adobe Target: Adobe Target helps test, personalize, and optimize mobile app experiences based on user behavior and mobile context. You can deliver interactions that engage and convert through iterative testing and rules-based and AI-powered personalization.](#_Toc202253882)

# Mobile App Analytics Install Instructions

***Here are the installation instructions documentation links for the Android and iOS platforms for reference:***  
[1. Add dependencies to your project](https://developer.adobe.com/client-sdks/home/getting-started/get-the-sdk/#1-add-dependencies-to-your-project)

[2. Add initialization code](https://developer.adobe.com/client-sdks/home/getting-started/get-the-sdk/#2-add-initialization-code)

[3. Ensure app permissions (Android only)](https://developer.adobe.com/client-sdks/home/getting-started/get-the-sdk/#3-ensure-app-permissions-android-only)

[4. Lifecycle metrics](https://developer.adobe.com/client-sdks/home/getting-started/enable-debug-logging/#lifecycle-metrics)  
  
***Documentation for Custom Codes for Track State and Track Action Tracking codes:***

5. [Track app states and screens (for Adobe Analytics)](https://developer.adobe.com/client-sdks/home/getting-started/track-events/#track-app-states-and-screens-for-adobe-analytics)

6. [Track user actions (for Adobe Analytics)](https://developer.adobe.com/client-sdks/home/getting-started/track-events/#track-user-actions-for-adobe-analytics)

|  |  |  |
| --- | --- | --- |
| **Platform** | **Android** | **iOS** |
| Development - Environment File ID | a3d08833fce3/13c5c32a5683/launch-175c67b044ae-development | a3d08833fce3/13c5c32a5683/launch-175c67b044ae-development |
| Production - Environment File ID | a3d08833fce3/13c5c32a5683/launch-59fa265903e7 | a3d08833fce3/13c5c32a5683/launch-59fa265903e7 |

## Android:

**1. Add the dependency to your project (build.gradle):** Each extension needs to be added as a dependency to the mobile application project. The following examples will add the Mobile Core and Profile extensions.  
  
We'll use [Gradle](https://gradle.org/) to manage dependencies. Refer to this [link](https://developer.adobe.com/client-sdks/documentation/manage-gradle-dependencies) to lock dependency versions.

Add the dependencies to build.gradle for each extension.

implementation platform('com.adobe.marketing.mobile:sdk-bom:3.+')

implementation 'com.adobe.marketing.mobile:edgeidentity'

implementation 'com.adobe.marketing.mobile:assurance'

implementation 'com.adobe.marketing.mobile:core'

implementation 'com.adobe.marketing.mobile:identity'

implementation 'com.adobe.marketing.mobile:lifecycle'

implementation 'com.adobe.marketing.mobile:signal'

implementation 'com.adobe.marketing.mobile:userprofile'

**2. Add Initialization Code:** Next you'll need to import SDK libraries into your project and register them for initialization. Extensions are registered with Mobile Core so that they can dispatch and listen for events.

The following code snippets demonstrate how you can import and register the Mobile Core and Profile extensions. You can also see, for reference, how Identity, Lifecycle, Signal, Profile, and other extensions are imported and registered.

After you register the extensions, call the start API in Mobile Core to initialize the SDK as shown below. This step is required to boot up the SDK for event processing.

import java.util.Arrays;

import java.util.List;

import com.adobe.marketing.mobile.AdobeCallback;

import com.adobe.marketing.mobile.Assurance;

import com.adobe.marketing.mobile.Extension;

import com.adobe.marketing.mobile.Identity;

import com.adobe.marketing.mobile.Lifecycle;

import com.adobe.marketing.mobile.LoggingMode;

import com.adobe.marketing.mobile.MobileCore;

import com.adobe.marketing.mobile.Signal;

import com.adobe.marketing.mobile.UserProfile;

import com.adobe.marketing.mobile.edge.identity.Identity;

...

import android.app.Application;

...

public class MainApp extends Application {

...

@Override

public void on Create(){

super.onCreate();

MobileCore.setApplication(this);

MobileCore.setLogLevel(LoggingMode.DEBUG);  
  
List<Class<? extends Extension>> extensions = Arrays.asList(

com.adobe.marketing.mobile.edge.identity.Identity.EXTENSION,

com.adobe.marketing.mobile.Identity.EXTENSION,

Assurance.EXTENSION,

Lifecycle.EXTENSION,

Signal.EXTENSION,

UserProfile.EXTENSION

);

MobileCore.registerExtensions(extensions, new AdobeCallback () {

@Override

public void call(Object o) {

MobileCore.configureWithAppID("a3d08833fce3/13c5c32a5683/launch-175c67b044ae-development"); **/\*set/update respective environment fileID based on Development / Production environments\*/**

}

});

**3. Ensure app permissions (Android only):** For Android, the SDK requires standard [network connection](https://developer.android.com/training/basics/network-ops/connecting) permissions in your manifest to send data, collect cellular provider, and record offline tracking calls.

To enable these permissions, add the following lines to your AndroidManifest.xml file, located in your app's application project directory:

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

## iOS

**1. Add the dependency to your project:** We'll use [CocoaPods](https://cocoapods.org/)  to manage dependencies.

Create a Podfile if you do not already have one:

$ pod init

**2. Add the dependencies to your Podfile for each extension:**

use\_frameworks!

pod 'AEPEdgeIdentity', '~> 5.0'

pod 'AEPAssurance', '~> 5.0'

pod 'AEPCore', '~> 5.0'

pod 'AEPIdentity', '~> 5.0'

pod 'AEPSignal', '~>5.0'

pod 'AEPLifecycle', '~>5.0'

pod 'AEPUserProfile', '~> 5.0'

**3. If CocoaPods could not find the dependencies, you may need to run this command:**

$ pod repo update

**4. Save the Podfile and run the install:**

$ pod install

**5. Add initialization code:**

Next you'll need to import SDK libraries into your project and register them for initialization. Extensions are registered with Mobile Core so that they can dispatch and listen for events.

The following code snippets demonstrate how you can import and register the Mobile Core and Profile extensions. You can also see, for reference, how Identity, Lifecycle, Signal, Profile, and other extensions are imported and registered.

**Swift:**

import AEPCore

import AEPEdgeIdentity

import AEPAssurance

import AEPIdentity

import AEPLifecycle

import AEPSignal

import AEPServices

import AEPUserProfile  
...

@UIApplicationMain

class AppDelegate: UIResponder, UIApplicationDelegate {

func application(\_application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool{

MobileCore.setLogLevel(.debug)

let appState = application.applicationState

let extensions = [

AEPEdgeIdentity.Identity.self,

AEPIdentity.Identity.self,

Assurance.self,

Lifecycle.self,

Signal.self,

UserProfile.self

]

MobileCore.registerExtensions(extensions, {

MobileCore.configureWith(appId: ("a3d08833fce3/13c5c32a5683/launch-175c67b044ae-development"); **/\*set/update respective environment fileID based on Development / Production environments\*/**  
 if appState != .background {

MobileCore.lifecycleStart(additionalContextData: ["contextDataKey": "contextDataVal"])

}

})

return true

}

}

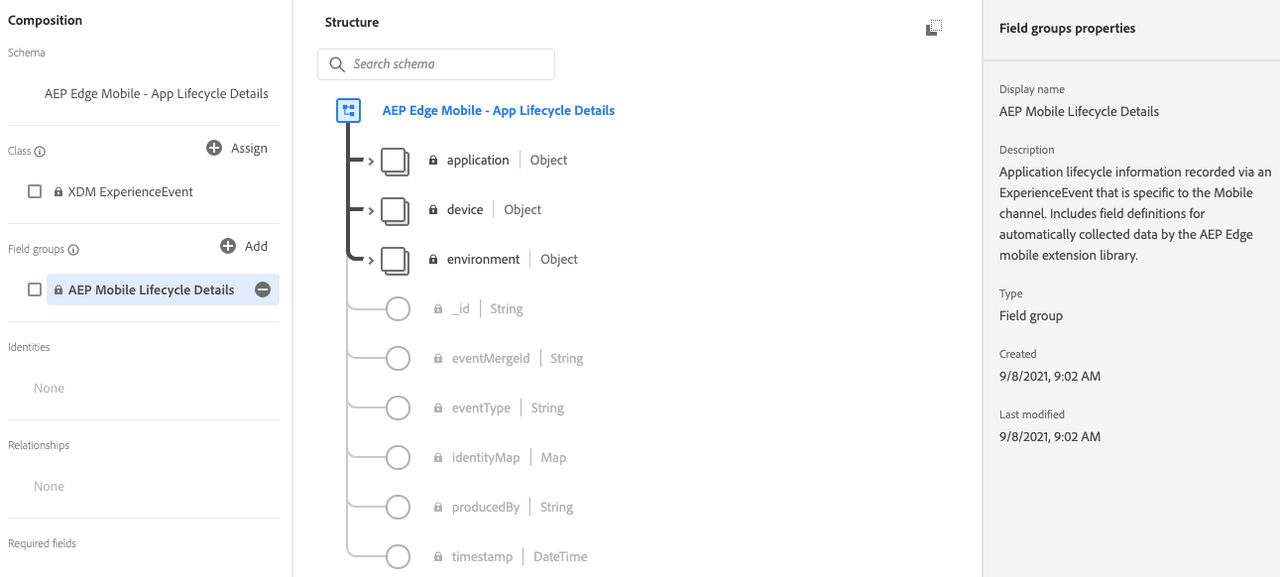
# Configure and Install Lifecycle for Edge Network

The Adobe Experience Platform Mobile SDK Lifecycle extension enables application lifecycle data collection from your mobile app when using the Adobe Experience Platform Mobile SDK and the [Edge Network extension](https://developer.adobe.com/client-sdks/edge/edge-network/). Use the following steps to set up an application to forward Lifecycle extension data to the Adobe Experience Platform.

Include the "AEP Mobile Lifecycle Details" field group to the schema defined in your Edge configuration datastream. This field group defines all the data auto-collected by the Lifecycle for Edge Network extension.

Creating an XDM schema is optional if your workflow does not require one. An XDM schema is required for some services, such as Adobe Experience Platform and Data Prep for mapping XDM fields on the server. However, if the datastream only enables the Analytics service, for example, then an XDM schema is not strictly required.

1. In Adobe Experience Platform, log in to your organization.
2. Under **Schemas** section, select the **Browse** tab and search for the schema used in the application's Edge configuration.
3. In the **Field groups** section, select **Add**.
4. Search for **AEP Mobile Lifecycle Details**. Select it, then select **Add field groups**.
5. Select **Save**.



The Lifecycle for Edge Network extension is part of the Mobile Core extension. There is no separate card for the Lifecycle for Edge Network extension in the Extensions Catalog.

The configuration setting "Session Timeout" in the Mobile Core extension configuration is used when collecting Lifecycle session data for Analytics and is not used for the Lifecycle for Edge Network extension.

1. In the Data Collection UI, from your mobile property, select the **Extensions** tab.
2. On the **Catalog** tab, locate or search for the **Mobile Core** extension, and select **Install**.
3. There are no configuration settings for **Lifecycle for Edge Network**.
4. Select **Save**.
5. Follow the publishing process to update SDK configuration.

In addition, use the following steps to [Configure the Edge Network extension](https://developer.adobe.com/client-sdks/edge/edge-network/#configure-edge-network-extension).

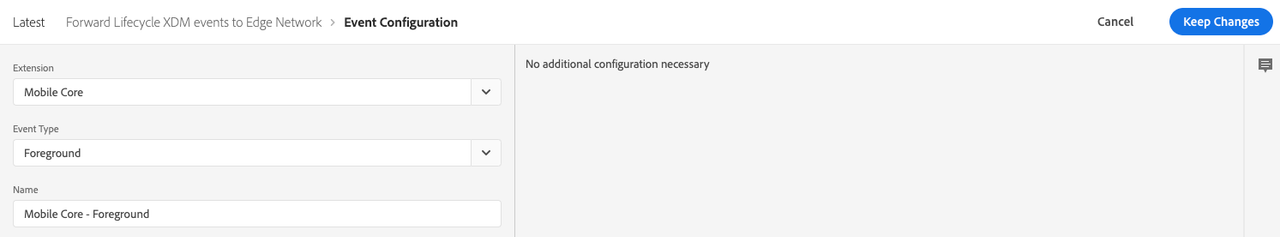
The Lifecycle for Edge Network extension dispatches application launch (foreground) and application close (background) events to the Mobile SDK. Create a rule to forward these events to the Adobe Experience Platform Edge Network.

1. On the **Rules** tab, select **Create New Rule**.
2. Give your rule an easily recognizable name in your list of rules.

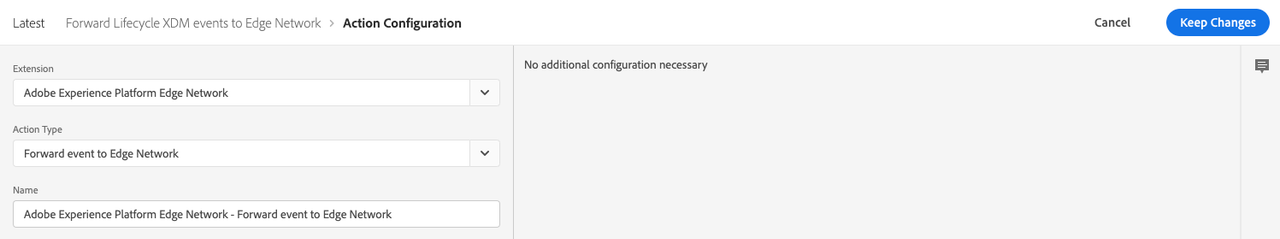
In this example, the rule is named "Forward Lifecycle XDM events to Edge Network".

If you do not have existing rules for this property, the **Create New Rule** button will be in the middle of the screen. If your property has rules, the button will be in the top right of the screen.

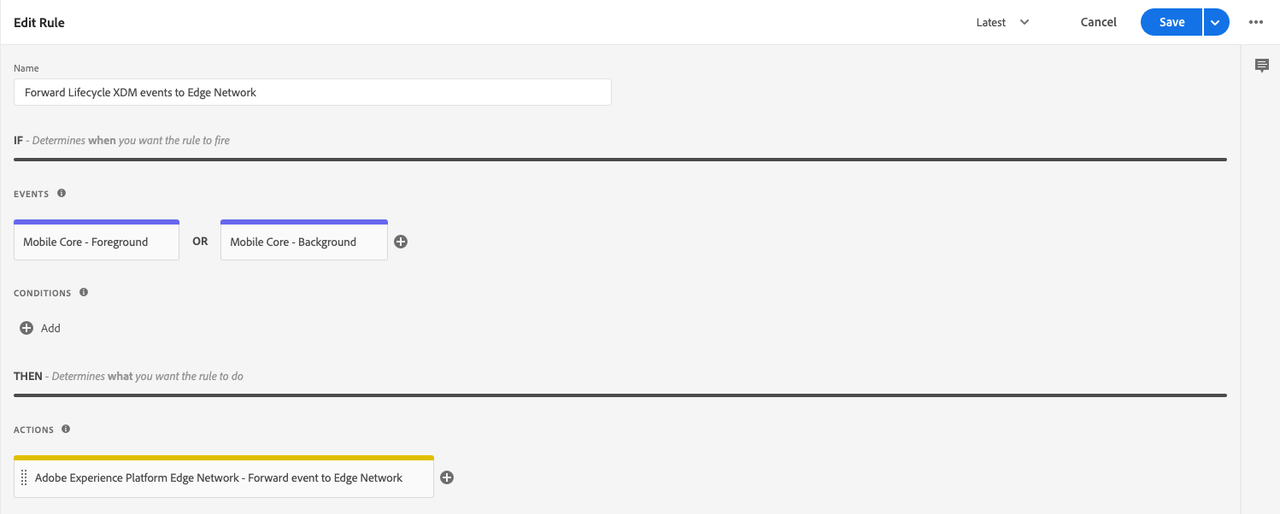
1. Under the **Events** section, select **Add**.
2. From the **Extension** dropdown list, select **Mobile Core**.
3. From the **Event Type** dropdown list, select **Foreground**.
4. Select **Keep Changes**.
5. Under the **Events** section again, select the plus icon to add another Event.
6. From the **Extension** dropdown list, select **Mobile Core**.
7. From the **Event Type** dropdown list, select **Background**.
8. Select **Keep Changes**.



1. Under the **Actions** section, select **Add**.
2. From the **Extension** dropdown list, select **Adobe Experience Platform Edge Network**.
3. From the **Action Type** dropdown list, select **Forward event to Edge Network**.
4. Select **Keep Changes**.



After you complete your configuration, verify that your rule looks like the following:



1. Select **Save**.
2. Rebuild your mobile property and deploy it to the correct environment.

# Track Events (Send events using Edge Network extension): This step requires knowledge of Experience Data Model (XDM) in Adobe Experience Platform. For more information about XDM, please read the [XDM documentation](https://experienceleague.adobe.com/docs/experience-platform/xdm/home.html).

|  |  |  |  |
| --- | --- | --- | --- |
| **Click Tracking Type** | **Screenshot** | **React Native - Android/iOS Tracking Code** | **Expected Values** |
| **Example value for Screen Name:** <Sign in to your account> | A screenshot of a login form  Description automatically generated | import com.adobe.marketing.mobile.MobileCore;  import com.adobe.marketing.mobile.Identity;  import com.adobe.marketing.mobile.AdobeCallback;  public void trackLoginSuccess() {  Map<String, String> additionalContextData = new HashMap<>();  additionalContextData.put("interactionName", "signin");  /\* Example: interactionName as a “sign in” \*/  // Retrieve ECID and add it to context data  Identity.getExperienceCloudId(new AdobeCallback<String>() {  @Override  public void call(String ecid) {  additionalContextData.put("ecid", ecid);  additionalContextData.put("userId", <user-id>); // set the masked/encrypted user id value  additionalContextData.put("pageName", "sign in to your account"); // Example context data of page name  MobileCore.trackAction("signinSuccess", additionalContextData);  }  });  } |  |