

Adobe Analytics Mobile App Technical Specification Document for RBG & BB App– Android & iOS

Vietnam Technological and Commercial Joint Stock Bank

A picture containing light

Description automatically generated

**Executive summary**

This tech spec contains basic setup and tracking best practices for implementing the Adobe Experience Platform (AEP) SDK on mobile apps (Android & iOS) using Adobe Launch. As a companion to this guide, a reference implementation using these best practices can be found at: <https://developer.adobe.com/client-sdks/documentation/>

This document is divided up into different sections, each of which describes a different component of the overall Analytics solution. Each component is designed to cover specific business need based on what was captured in the Business Requirement Document (BRD).

Eventhough most of the components outlined in this document can be used to track apps in other devices (like Cars - Android Auto, Watch - Android Wear), this specification is related to the typical mobile app which is meant to be installed on a mobile phone. Please refer device specific technical specifications for more details.

You can also use the below list of Adobe Analytics reference materials (Not necessarily related AEP SDK). Throughout this spec, certain other additional documents and sections will be called out for further information.

* [Adobe Analytics Implementation Guide](https://experienceleague.adobe.com/docs/analytics/implementation/home.html?lang=en)
* [Adobe Analytics Common Metric Definition List](https://experienceleague.adobe.com/docs/analytics/components/metrics/overview.html?lang=en)
* [Adobe Analytics Variable Definitions and Limitations](https://experienceleague.adobe.com/docs/analytics/implementation/vars/overview.html?lang=en)
* [AEP SDK Github Page](https://github.com/Adobe-Marketing-Cloud/aep-sdks-documentation)
* [Overview on Tags by Adobe](https://experienceleague.adobe.com/docs/platform-learn/data-collection/tags/create-a-property.html?lang=en)

Document control

|  |  |
| --- | --- |
| Document title | **Adobe Analytics Mobile App Technical Specification Document for RBG & BB App– Android & iOS**  Vietnam Technological and Commercial Joint Stock Bank |
| Company | Vietnam Technological and Commercial Joint Stock Bank |
| Authors | Sandipan Roy |

Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| **1.0** | 24.04.2023 | Sandipan Roy | Initial version for internal XDM review. |
| **1.1** | 02.05.2023 | Sandipan Roy | Initial version for TCB review |
| **1.2** | 03.05.2023 | Sandipan Roy | DataStream update after alignment with RTCDP team |
| **1.3** | 15.05.2023 | Sandipan Roy | Added new variable for calculator fields & results.  Added Timed action tracking. |
| **1.4** | 29.05.2023 | Sandipan Roy | Added description for extensions in section 3.4.4. |
| **1.5** | 07.06.2023 | Sandipan Roy | Added description about Business App property and Datastreams in section 2.2, 3.1, 3.2. |
| **1.6** | 27.06.2023 | Sandipan Roy | Added instruction for registration of extensions as per new Adobe Edge SDK release in section 4.1.  Added new dimension “*Device general name*” to capture mobile device name in all screen loads & all clickevent tracking. Please check XDM code from the section 4.3 onwards. |
| **1.7** | 19.07.2023 | Sandipan Roy | Deleted country variable since it is not required as per current scope. |
| **1.8** | 10.8.2023 | Sandipan Roy | Updated links for documentation, added Lifecycle tracking rule snapshot |
| **1.9** | 30.08.2023 | Sandipan Roy | Updated xdm path for variables “previousPageName” and “customPageView” event. Please refer section 4.3.2 |
| **2.0** | 12.09.2023 | Sandipan Roy | Updated notes related to data types and expected values for RTCDP. Please refer section 4.2 |

Table of contents

[1 Mobile App Setup Introduction [Done: Adobe Team] 8](#_Toc145453754)

[2 Datastreams Configuration [Done: Adobe Team] 9](#_Toc145453755)

[2.1 Provisioning [Done: Adobe Team] 9](#_Toc145453756)

[2.2 Creation of Datastreams [Done: Adobe Team] 9](#_Toc145453757)

[3 Tags setup [Done: Adobe Team] 9](#_Toc145453758)

[3.1 Creating a mobile property [Done: Adobe Team] 9](#_Toc145453759)

[3.2 Adobe Data Collection - Setup of environments [Done: Adobe Team] 10](#_Toc145453760)

[3.3 Review pre-installed extensions. [Done: Adobe Team] 11](#_Toc145453761)

[3.4 Add Experience Cloud Solution Extensions [Done: Adobe Team] 12](#_Toc145453762)

[3.4.1 Adobe Experience Platform Edge Network [Done: Adobe Team] 12](#_Toc145453763)

[3.4.2 Identity for Edge Network [Done: Adobe Team] 12](#_Toc145453764)

[3.4.3 Consent for Edge Network [Done: Adobe Team] 12](#_Toc145453765)

[3.4.4 Other Extensions [Done: Adobe Team] 12](#_Toc145453766)

[3.5 Create and publish a Tags library. [Done: Adobe Team] 13](#_Toc145453767)

[3.6 Install the AEP SDK in your app (Android) – Step 1 (Add Dependencies) [To do: TCB IT Team] 13](#_Toc145453768)

[3.7 Install the AEP SDK in your app (Android) – Step 2 (Update code in the app) [To do: TCB IT Team] 16](#_Toc145453769)

[3.8 Install the AEP SDK in your app (iOS)– Step 1 (Add Dependencies) 17](#_Toc145453770)

[3.9 Install the AEP SDK in your app (iOS) – Step 2 (Update code in the app) 18](#_Toc145453771)

[3.10 Verify the AEP SDK imports. [To do: TCB IT Team] 20](#_Toc145453772)

[4 Solution Components (RGB & BB Apps) 20](#_Toc145453773)

[4.1 Lifecycle Tracking [To do: TCB IT Team] 20](#_Toc145453774)

[4.2 General guidelines for screen and action tracking 23](#_Toc145453775)

[4.2.1 Screen Tracking (Equivalent to Page view tracking in web) 24](#_Toc145453776)

[4.2.2 Action Tracking (Equivalent to trackLink/Success event tracking in web) 25](#_Toc145453777)

[4.3 Global - Screens 27](#_Toc145453778)

[4.3.1 Variables in this Section 27](#_Toc145453779)

[4.3.2 Deployment Instructions [To do: TCB IT Team] 27](#_Toc145453780)

[4.3.3 Create Processing Rules [To do: Adobe Team] 32](#_Toc145453781)

[4.4 Global - Custom Link Tracking (Link/Button) 33](#_Toc145453782)

[4.4.1 Variables in this Section 33](#_Toc145453783)

[4.4.2 Deployment Instructions [To do: TCB IT Team] 33](#_Toc145453784)

[4.4.3 Create Processing Rules [To do: Adobe Team] 35](#_Toc145453785)

[4.5 Global – Login/Logout 35](#_Toc145453786)

[4.5.1 Variables in this Section 35](#_Toc145453787)

[4.5.2 Deployment Instructions [To do: TCB IT Team] 36](#_Toc145453788)

[4.5.3 Create Processing Rules [To do: Adobe Team] 38](#_Toc145453789)

[4.6 Global – Banner 38](#_Toc145453790)

[4.6.1 Variables in this Section 38](#_Toc145453791)

[4.6.2 Deployment Instructions [To do: TCB IT Team] 39](#_Toc145453792)

[4.6.3 Create Processing Rules [To do: Adobe Team] 41](#_Toc145453793)

[4.7 Global – Carousel 41](#_Toc145453794)

[4.7.1 Variables in this Section 41](#_Toc145453795)

[4.7.2 Deployment Instructions [To do: TCB IT Team] 41](#_Toc145453796)

[4.7.3 Create Processing Rules [To do: Adobe Team] 43](#_Toc145453797)

[4.8 Global – Calculator 44](#_Toc145453798)

[4.8.1 Variables in this Section 44](#_Toc145453799)

[4.8.2 Deployment Instructions [To do: TCB IT Team] 44](#_Toc145453800)

[4.8.3 Create Processing Rules [To do: Adobe Team] 46](#_Toc145453801)

[4.9 Global – Downloads 46](#_Toc145453802)

[4.9.1 Variables in this Section 46](#_Toc145453803)

[4.9.2 Deployment Instructions [To do: TCB IT Team] 47](#_Toc145453804)

[4.9.3 Create Processing Rules [To do: Adobe Team] 49](#_Toc145453805)

[4.10 Global – FAQs 49](#_Toc145453806)

[4.10.1 Variables in this Section 49](#_Toc145453807)

[4.10.2 Deployment Instructions [To do: TCB IT Team] 49](#_Toc145453808)

[4.10.3 Create Processing Rules [To do: Adobe Team] 51](#_Toc145453809)

[4.11 Global – Navigation clicks 52](#_Toc145453810)

[4.11.1 Variables in this Section 52](#_Toc145453811)

[4.11.2 Deployment Instructions [To do: TCB IT Team] 52](#_Toc145453812)

[4.11.3 Create Processing Rules [To do: Adobe Team] 54](#_Toc145453813)

[4.12 Global – Inbox notifications. 54](#_Toc145453814)

[4.12.1 Variables in this Section 54](#_Toc145453815)

[4.12.2 Deployment Instructions [To do: TCB IT Team] 54](#_Toc145453816)

[4.12.3 Create Processing Rules [To do: Adobe Team] 57](#_Toc145453817)

[4.13 Global – Tab. 57](#_Toc145453818)

[4.13.1 Variables in this Section 57](#_Toc145453819)

[4.13.2 Deployment Instructions [To do: TCB IT Team] 57](#_Toc145453820)

[4.13.3 Create Processing Rules [To do: Adobe Team] 59](#_Toc145453821)

[4.14 Global – Error tracking. 60](#_Toc145453822)

[4.14.1 Variables in this Section 60](#_Toc145453823)

[4.14.2 Deployment Instructions [To do: TCB IT Team] 60](#_Toc145453824)

[4.14.3 Create Processing Rules [To do: Adobe Team] 63](#_Toc145453825)

[4.15 Global – Time out. 63](#_Toc145453826)

[4.15.1 Variables in this Section 63](#_Toc145453827)

[4.15.2 Deployment Instructions [To do: TCB IT Team] 63](#_Toc145453828)

[4.15.3 Create Processing Rules [To do: Adobe Team] 65](#_Toc145453829)

[4.16 eKYC – Journey 66](#_Toc145453830)

[4.16.1 Variables in this Section 66](#_Toc145453831)

[4.16.2 Deployment Instructions [To do: TCB IT Team] 66](#_Toc145453832)

[4.16.3 Create Processing Rules [To do: Adobe Team] 67](#_Toc145453833)

[4.17 eKYC video – Journey 67](#_Toc145453834)

[4.17.1 Variables in this Section 67](#_Toc145453835)

[4.17.2 Deployment Instructions [To do: TCB IT Team] 68](#_Toc145453836)

[4.17.3 Create Processing Rules [To do: Adobe Team] 69](#_Toc145453837)

[4.18 eKYC – Video call start/end. 69](#_Toc145453838)

[4.18.1 Variables in this Section 69](#_Toc145453839)

[4.18.2 Deployment Instructions [To do: TCB IT Team] 69](#_Toc145453840)

[4.18.3 Create Processing Rules [To do: Adobe Team] 71](#_Toc145453841)

[4.19 All Products – Journey Start/Complete/Fail 71](#_Toc145453842)

[4.19.1 Variables in this Section [To do: TCB IT Team] 72](#_Toc145453843)

[4.19.2 Deployment Instructions 72](#_Toc145453844)

[4.19.3 Create Processing Rules [To do: Adobe Team] 73](#_Toc145453845)

[4.20 All Product pages CTA clicks. 73](#_Toc145453846)

[4.20.1 Variables in this Section 73](#_Toc145453847)

[4.20.2 Deployment Instructions [To do: TCB IT Team] 73](#_Toc145453848)

[4.20.3 Create Processing Rules [To do: Adobe Team] 76](#_Toc145453849)

[4.21 Auto Loan 76](#_Toc145453850)

[4.21.1 Variables in this Section 77](#_Toc145453851)

[4.21.2 Deployment Instructions [To do: TCB IT Team] 77](#_Toc145453852)

[4.21.3 Create Processing Rules [To do: Adobe Team] 77](#_Toc145453853)

[4.22 Flexible Term Deposit . 78](#_Toc145453854)

[4.22.1 Variables in this Section 78](#_Toc145453855)

[4.22.2 Deployment Instructions [To do: TCB IT Team] 78](#_Toc145453856)

[4.22.3 Create Processing Rules [To do: Adobe Team] 80](#_Toc145453857)

[4.23 Goal Saving 81](#_Toc145453858)

[4.23.1 Variables in this Section 81](#_Toc145453859)

[4.23.2 Deployment Instructions [To do: TCB IT Team] 81](#_Toc145453860)

[4.23.3 Create Processing Rules [To do: Adobe Team] 83](#_Toc145453861)

[4.24 Manage Profile – Change Mobile Number 83](#_Toc145453862)

[4.24.1 Variables in this Section 83](#_Toc145453863)

[4.24.2 Deployment Instructions [To do: TCB IT Team] 84](#_Toc145453864)

[4.24.3 Create Processing Rules [To do: Adobe Team] 86](#_Toc145453865)

[4.25 Notification 86](#_Toc145453866)

[4.25.1 Variables in this Section 87](#_Toc145453867)

[4.25.2 Deployment Instructions [To do: TCB IT Team] 87](#_Toc145453868)

[4.25.3 Create Processing Rules [To do: Adobe Team] 87](#_Toc145453869)

[4.26 Unsecured Lending 88](#_Toc145453870)

[4.26.1 Variables in this Section 88](#_Toc145453871)

[4.26.2 Deployment Instructions [To do: TCB IT Team] 88](#_Toc145453872)

[4.26.3 Create Processing Rules [To do: Adobe Team] 89](#_Toc145453873)

[4.27 Transfer 89](#_Toc145453874)

[4.27.1 Variables in this Section 89](#_Toc145453875)

[4.27.2 Deployment Instructions [To do: TCB IT Team] 89](#_Toc145453876)

[4.27.3 Create Processing Rules [To do: Adobe Team] 90](#_Toc145453877)

[4.28 Overdraft 90](#_Toc145453878)

[4.28.1 Variables in this Section 90](#_Toc145453879)

[4.28.2 Deployment Instructions [To do: TCB IT Team] 90](#_Toc145453880)

[4.28.3 Create Processing Rules [To do: Adobe Team] 92](#_Toc145453881)

[4.29 Certificate of Deposit 92](#_Toc145453882)

[4.29.1 Variables in this Section 92](#_Toc145453883)

[4.29.2 Deployment Instructions [To do: TCB IT Team] 92](#_Toc145453884)

[4.29.3 Create Processing Rules [To do: Adobe Team] 93](#_Toc145453885)

[4.30 Rewards 93](#_Toc145453886)

[4.30.1 Variables in this section: 93](#_Toc145453887)

[4.30.2 Deployment Instructions [To do: TCB IT Team] 93](#_Toc145453888)

[4.30.3 Create Processing Rules [To do: Adobe Team] 95](#_Toc145453889)

[4.31 Term Deposit 95](#_Toc145453890)

[4.31.1 Variables in this section: 96](#_Toc145453891)

[4.31.2 Deployment Instructions [To do: TCB IT Team] 96](#_Toc145453892)

[4.31.3 Create Processing Rules [To do: Adobe Team] 98](#_Toc145453893)

[4.32 Credit card 99](#_Toc145453894)

[4.32.1 Variables in this section: 99](#_Toc145453895)

[4.32.2 Deployment Instructions [To do: TCB IT Team] 99](#_Toc145453896)

[4.32.3 Create Processing Rules [To do: Adobe Team] 101](#_Toc145453897)

[4.33 Time between two Action. 101](#_Toc145453898)

[4.33.1 Variables in this section: 102](#_Toc145453899)

[4.33.2 Deployment Instructions [To do: TCB IT Team] 102](#_Toc145453900)

[4.33.3 Create Processing Rules [To do: Adobe Team] 103](#_Toc145453901)

[5 App View to Web View Tracking 103](#_Toc145453902)

[5.1 Implementation Instructions: [To do: TCB IT Team] 103](#_Toc145453903)

[6 Steps to Capture ECID. [To do: TCB IT Team] 105](#_Toc145453904)

[7 Validation [To do: Adobe Team] & [TCB IT Team] 106](#_Toc145453905)

[7.1 Project Griffon (Adobe Experience Platform Assurance) 106](#_Toc145453906)

[7.1.1 Implement Assurance on the App 106](#_Toc145453907)

[7.1.2 Using Project Griffon for Validation 107](#_Toc145453908)

[7.1.3 Understanding Analytics calls in the Griffon Session 110](#_Toc145453909)

[7.2 How to use Charles for Validation [To do: TCB IT Team] 113](#_Toc145453910)

[7.2.1 Android build settings for Charles SSL Proxying 113](#_Toc145453911)

[7.2.2 Trust Charles certificate on Device 113](#_Toc145453912)

[7.2.3 Enable SSL Proxying in Charles 114](#_Toc145453913)

[7.2.4 Configure Charles for testing with your mobile device 114](#_Toc145453914)

[7.2.5 Use Charles with VPN connection 115](#_Toc145453915)

[7.2.6 Understanding Analytics calls in the Charles Session 115](#_Toc145453916)

[7.2.7 Verifying Lifecycle Calls 116](#_Toc145453917)

[7.2.8 Verifying Page View calls (previously trackState server calls) 116](#_Toc145453918)

[7.2.9 Verifying Link click calls (previously trackAction server calls) 117](#_Toc145453919)

[7.2.10 Verifying analytics Products variable (previously &&products) 117](#_Toc145453920)

[7.2.11 Verifying event serialization (previously &&events) 118](#_Toc145453921)

# Mobile App Setup Introduction [Done: Adobe Team]

AEP Mobile SDK can be deployed using AEP Data Collection - Tags. Tags is the next generation tag management solution from Adobe. Adobe Tags enables an enterprise to manage all their digital properties (like website/SPA/Mobile Apps) all in one single place. Same workflows can be used for managing and publishing changes using Tags.

**NOTE:** Tags is not a mobile tag management system. Only some configurations are managed by Tags. Any change in the tracking/extensions, still requires an update to the app.

# Datastreams Configuration [Done: Adobe Team]

A Datastream is required to route data from the AEP Edge Network to one of the end systems – Adobe Analytics, Adobe Experience Platform etc. Learn more about Datastreams [here](https://experienceleague.adobe.com/docs/experience-platform/edge/fundamentals/datastreams.html?lang=en).

## Provisioning [Done: Adobe Team]

Ensure the client’s account has been provisioned for Mobile Edge SDK. Contact your CSM to enable this.

## Creation of Datastreams [Done: Adobe Team]

Login into Adobe Tags.

Navigate to Datastreams on Left rail.

Click on New Datastream.

Enter a Friendly Name and Description (optional)

Click on Save.

In the resulting screen click on the Datastream you just created, click on New Environment

Turn on Adobe Analytics and select the report suite.

Click on Save to save the configuration.

# Tags setup [Done: Adobe Team]

## Creating a mobile property [Done: Adobe Team]

This is the first step in implementing the AEP SDK. We will be using Tags for implementation of the AEP Mobile SDK in Android apps. A property needs to be created and below are the steps:

1. Click the “Add New Property” or “New Property” button:
2. Name your property (to the name of the app) and select “Mobile” as the platform.
3. Don’t change any of the Advanced Settings.
4. Click “Save”.
5. Your new property should display on Property list page.
6. Click on the name of your property to open the Overview screen and then click on the Extensions tab. Note that the “Mobile Core” and “Profile” extensions are automatically added when you created the Property.

|  |
| --- |
| NOTE |
| * To complete these steps, you must have relevant permissions in AEP Data Collection - Tags. These are permissions to Develop, Approve, Publish, Manage Extensions, and Manage Environments * Contact your Experience Cloud administrator to request access |

|  |
| --- |
| ***Best Practices for creating properties in case you have different apps across multiple app stores*** |
| * If your apps send data to the same Analytics report suites, use the same extensions, rules, data elements, and so on, we recommend that you group all the mobile apps into the same Tags property. * If your apps send data to different Analytics report suites, or user different extensions per app, and so on, we recommend that you create separate mobile properties. |

## Adobe Data Collection - Setup of environments [Done: Adobe Team]

Following environments will be setup under the above created property (ex. Retail Banking App, Business Banking App) to test the code at different development lifecycle phases. Upon successful validation of code in the environment, code will be pushed to next environment via publishing flow.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Order environments (lower to higher)** | **Mobile app environment**  **RBG** | **Mobile app environment**  **BB** | **Adobe Data Collection environment** | **Report suite** |
| 1 | Development | NA | Development | TCB Dev (techcombank-vn-dev) |
| 2 | SIT | SIT | Development | TCB Dev (techcombank-vn-dev) |
| 3 | UAT | UAT | Development | TCB Dev (techcombank-vn-dev) |
| 4 | Staging | Staging | Staging | TCB Dev (techcombank-vn-dev) |
| 5 | Production | Production | Production | TCB Prod (techcombank-vn-prod) |

Below are the details of the Adobe Tag Property created for the Mobile Edge SDK configuration:

|  |  |
| --- | --- |
| **Tag Property Name** | **Details** |
| Retail Banking App | This Tag property will cover RBG Pre & Post Login journeys. |
| Business Banking App | This Tag property will cover BB journeys. |

Following are the Datastream IDs for the respective Analytics Dev & Prod Report Suites:

|  |  |  |  |
| --- | --- | --- | --- |
| **Datastream Name** | **Datastream IDs** | **Datastream Instance** | **Report Suites Tagged** |
| Techcombank RBG App | 17f52581-5fc5-401f-a595-4ee989d552fb | Development | TCB Dev (techcombank-vn-dev) |
| Techcombank RBG App | TBD | Production | TCB Prod (techcombank-vn-prod) |
| Techcombank BB App | 20295d76-38f9-4c4f-a8e5-541fec35ffeb | Development | TCB Dev (techcombank-vn-dev) |
| Techcombank BB App | TBD | Production | TCB Prod (techcombank-vn-prod) |

*\*TBD : dependency on RTCDP Prod environment setup.*

Following are the Schema details for the Dev & the Prod environments:

|  |  |
| --- | --- |
| **Schema Name** | **Schema Instance** |
| Techcombank Online Inlet- Web/App | Development |
| Techcombank Online Inlet- Web/App | Production |

**Environment File IDs for RBG App:**

|  |  |  |
| --- | --- | --- |
| **Sl.** | **Environments** | **Environment File ID** |
| 1 | Development | 04734335a754/7ab520637377/launch-8090ea75c962-development |
| 2 | SIT | 04734335a754/7ab520637377/launch-bb52b55279df-development |
| 3 | UAT | 04734335a754/7ab520637377/launch-eb64716a078f-development |
| 4 | Staging | 04734335a754/7ab520637377/launch-8fec53d3a62a-staging |
| 5 | Production | 04734335a754/7ab520637377/launch-84594e646f07 |

**Environment File IDs for BB App:**

|  |  |  |
| --- | --- | --- |
| **Sl.** | **Environments** | **Environment File ID** |
| 1 | Development | 04734335a754/5456be1f9bc6/launch-cfc739eb2a10-development |
| 2 | SIT | 04734335a754/5456be1f9bc6/launch-14abb96280b9-development |
| 3 | UAT | 04734335a754/5456be1f9bc6/launch-62ebd81e99b4-development |
| 4 | Staging | 04734335a754/5456be1f9bc6/launch-72f07149224c-staging |
| 5 | Production | 04734335a754/5456be1f9bc6/launch-7200406dfa4b |

## Review pre-installed extensions. [Done: Adobe Team]

As discussed in the above section, the two extensions “Mobile Core” and “Profile” are pre-installed in your mobile web property. You should review the “Mobile Core” extension for some pre-populated settings. “Profile” extension does not need any setting/validation.

1. Click the Extensions tab to go to the extensions page
2. Click the Configure button on the Core extension to examine its settings
3. Mobile Core contains common set of functionality and frameworks such as a Experience Cloud Identity services, data event hub, rules engine etc. Check the Experience Cloud Org ID which is pre-populated.
4. “Experience Cloud Server” field allows you to specify a custom endpoint for Visitor ID Service requests. Use the default setting for now (leave it blank)
5. “Session Timeout” field allows you to specify when an app’s Lifecycle session should timeout after an app is pushed to the background. By default, it is setup as 300 seconds.
6. Click “Save” to confirm your settings for Mobile Core.

## Add Experience Cloud Solution Extensions [Done: Adobe Team]

### Adobe Experience Platform Edge Network [Done: Adobe Team]

1. Click the “Extensions” tab to go to the extensions page
2. Click on the “Catalog” tab to see the uninstalled extensions
3. Find the “Adobe Experience Platform Edge Network” extension and click “Install”.
4. Select the appropriate Datastreams configured as per earlier sections.
5. Specify the domain name to be used for edge server calls.
6. Click “Save”
7. Follow the publishing process to update SDK configuration.

### Identity for Edge Network [Done: Adobe Team]

The Adobe Experience Platform Edge Network extension (described above) requires the Identity for Edge Network extension to operate. As a first step install and configure the [Identity for Edge Network](https://developer.adobe.com/client-sdks/documentation/identity-for-edge-network/) extension.

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

### Consent for Edge Network [Done: Adobe Team]

The Adobe Experience Platform Consent mobile extension enables consent preferences collection from your mobile app when using the [Adobe Experience Platform Mobile SDK](file:///C:/docs/foundation-extensions/mobile-core) and the [Edge Network extension](file:///C:/docs/foundation-extensions/experience-platform-extension).

1. In Data Collection UI, in your mobile property, select the Extensions tab.
2. On the Catalog tab, locate or search for the Consent extension, and select Install.
3. Set your desired default consent level.
4. Select Save.
5. Follow the publishing process to update SDK configuration.

### Other Extensions [Done: Adobe Team]

The other extensions required for Analytics implementation are as follows:

1. **AEP Assurance** - It easily inspects, validates, and debugs app data collection and experiences.
2. **Mobile Core** (Present by default) - It allows configuration of the Mobile SDK and provides access to default lifecycle events and conditions..
3. **Profile** (Present by default) - It allows the SDK to store data into a client-side profile.

## Create and publish a Tags library. [Done: Adobe Team]

A library packages together all the desired Tags settings/recent changes and generates the specific implementation instructions required to install the library in your mobile app. After making all the changes, you need to create a library and publish it to either of the environments [Dev/Staging/Production (Publish)]. Below are the steps:

1. Go to the “Publishing” tab
2. Click “Add New Library” button
3. Name the Library “Initial Setup”
4. Select Environment > Development
5. Click “Add All Changed Resources”
6. Adobe Launch will include all the resources that was added or changed
7. Click “Save & Build for Development”.

Environments will be set up under the above created Mobile property to test the code at different development lifecycle phases. Upon successful validation of code in the environment, code will be pushed to the next environment via publising flow.

## Install the AEP SDK in your app (Android) – Step 1 (Add Dependencies) [To do: TCB IT Team]

To incorporate the SDK into the app, the initial action is to obtain the Installation Instructions. The Installation Instructions for mobile Tags properties are a collection of code snippets that you need to add to specific locations in your mobile app. Below are the steps:

1. Click on the “Environments” tab in the top navigation to go to the environments page.
2. Development, Staging, and Production environments have been pre-created. These correspond to the typical environments in the code development and release process.
3. Select the corresponding environment in which you want to install the AEP SDK and click on the box icon to open the code/instructions.

Graphical user interface, text, application

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with medium confidence

1. The Adobe Mobile SDK for Android uses Gradle to manage dependencies between its various components. Next, we need to add the Adobe Mobile SDK dependencies to the app's build.gradle file. Below images illustrate the steps using Android Studio IDE.
2. Copy the dependencies to your clipboard, by clicking the https://docs.adobe.com/content/dam/help/experience-cloud.en/help/mobile-android-implementation/images/mobile-launch-copyIcon.png icon and paste in gradle file.
3. Click the "Sync Now" link to sync the project.

## Install the AEP SDK in your app (Android) – Step 2 (Update code in the app) [To do: TCB IT Team]

In this step, we need to make code changes in the App & initialize the SDK. These changes include the import and registration statements for Identity, Lifecycle, and Signal extensions, as well as the initialization of the Lifecyle metrics for Mobile Core extension. Below are the steps:

1. Open the Main Application file of your app, which is typically “mainActivity.java”
2. In the Adobe Launch interface, scroll to the “Add Initialization Code” section. Here you will find three https://docs.adobe.com/content/dam/help/experience-cloud.en/help/mobile-android-implementation/images/mobile-launch-copyIcon.pngicons (use the scroll bars if you don’t see it) to copy 3 different sets of code.
3. Copy the first set of import statements, click the first https://docs.adobe.com/content/dam/help/experience-cloud.en/help/mobile-android-implementation/images/mobile-launch-copyIcon.png icon in the “Add Initialization Code” section.
4. In Android Studio, paste these import statements before the existing imports in the main file.
5. In the Tags interface, copy the two lines related to the Core extension, by clicking the second https://docs.adobe.com/content/dam/help/experience-cloud.en/help/mobile-android-implementation/images/mobile-launch-copyIcon.png icon in the “Add Initialization Code” section. The second line turns on console logging statements (available options are "DEBUG", "VERBOSE", "WARNING", and "ERROR").
6. In Android Studio, paste these Core statements into the main file right after “super.onCreate()”.
7. If there are comments in the “try” “catch” lines, then these should be removed.
8. In the Launch interface, copy the extension statements, by clicking the third https://docs.adobe.com/content/dam/help/experience-cloud.en/help/mobile-android-implementation/images/mobile-launch-copyIcon.png icon in the “Add Initialization Code” section.
9. In Android Studio, paste these extension statements into the try section.

|  |
| --- |
| ***Important Note:*** |
| * The 3rd code section contains an identifier like this: MobileCore.configureWithAppID("<ENVIRONMENT FILE ID>"); * This is the identifier of the Tags environment, and the above code will import all the libraries that are in ‘Development’ environment. * Once development and testing are complete, the changes (through libraries) should be pushed to Production (Publish) and this code needs to be updated with the production identifier and be added in the public facing apps. * Also, in production apps, the debug logging can be removed. * Please note, the latest versions of the extensions are accessible through the provided link for both iOS and Android platforms. We strongly recommend utilizing the specific version numbers in your App project instead of indicating the '+' sign for automatic updates. You can find the link here:   <https://developer.adobe.com/client-sdks/documentation/current-sdk-versions/> |

## Install the AEP SDK in your app (iOS)– Step 1 (Add Dependencies)

The first step in the process is to get the SDK into the app is to get the Installation Instructions. The Installation Instructions for mobile Tags properties are a collection of code snippets that you need to add to specific locations in your mobile app. Below are the steps:

1. Click on the “Environments” tab in the top navigation to go to the environments page.
2. Development, Staging, and Production environments have been pre-created. These correspond to the typical environments in the code development and release process. We have also added UAT and SIT Environments as per requirement.
3. Select the corresponding environment in which you want to install the AEP SDK and click on the box icon to open the code/instructions.

Graphical user interface, text, application

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with low confidence

1. The Adobe Mobile SDK for iOS uses Podfile to manage dependencies between its various components. Next, we need to add the Adobe Mobile SDK dependencies to the app's Podfile.
2. Copy the dependencies to your clipboard, by clicking the https://docs.adobe.com/content/dam/help/experience-cloud.en/help/mobile-android-implementation/images/mobile-launch-copyIcon.png icon and paste in podfile.

## Install the AEP SDK in your app (iOS) – Step 2 (Update code in the app)

In this step, we need to make code changes in the App & initialize the SDK. These changes include the import and registration statements for Identity, Lifecycle, and Signal extensions, as well as the initialization of the Lifecyle metrics for Mobile Core extension. Below are the steps:

1. Add the dependencies to your Pod file by clicking on the copy icon as shown

A screenshot of a computer

Description automatically generated with low confidence

1. Add initialization code in App project-

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

|  |
| --- |
| ***Important Note:*** |
| * The 3rd code section contains an identifier like this: MobileCore.configureWithAppID("<ENVIRONMENT FILE ID>"); * This is the identifier of the Tags environment and the above code will import all the libraries that are in ‘Development’ environment * Once development and testing are complete, the changes (through libraries) should be pushed to Production (Publish) and this code needs to be updated with the production identifier and be added in the public facing apps * Also, in production apps, the debug logging can be removed * Please note, all the updated extension versions are available in the below link for both iOS and Android. It is recommended to use the exact version number in your App project rather than mentioning ‘+’ sign for automatic updates. Link - <https://developer.adobe.com/client-sdks/documentation/current-sdk-versions/> |

## Verify the AEP SDK imports. [To do: TCB IT Team]

1. Save your Android Studio project.
2. Run the app in the Emulator (running Android 4.1 or later) or side-load it in the connected Android device (physical, developer phone, connected to your laptop).
3. Confirm the calls are being made to the Adobe servers in Android Studio Logcat.
4. Below are some of the key calls:
   1. Calls to retrieve the Launch configuration (filter Logcat to adobedtm.com)
   2. Request to the Identity Service (filter Logcat to IdentityExtension)
   3. Request to experience Edge (filter Logcat for AEP Request Event)
5. These calls confirms that AEP SDK has been successfully added to your app/project.
6. Now the tracking calls/solution components can be deployed to the app.

# Solution Components (RGB & BB Apps)

## Lifecycle Tracking [To do: TCB IT Team]

Lifecycle metrics/dimensions are valuable, out-of-the-box information about your app user. “Lifecycle metrics” are environment-based metrics and dimensions that can be easily enabled in an app using the AEP SDK. These metrics contain information on the app user's lifecycle such as device information, install or upgrade information, session start and pause times, etc. If you would have followed to above steps of configuring the SDK/Extensions in Adobe Launch and importing the SDK, then you must complete the below step to enable lifecycle tracking:

**For Android:**

1. Add the below Lifecycle code to the main onResume() function in the app, in order to trigger the Lifecycle functions

@Override

public void onResume() {

MobileCore.setApplication(getApplication());

MobileCore.lifecycleStart(null);

}

2. Use the onPause function to pause the lifecycle data collection.

@Override

public void onPause() {

MobileCore.lifecyclePause();

}

3. To ensure accurate session and crash reporting, these calls must be added to every activity.

|  |
| --- |
| ***Best Practices for adding lifecycle tracking*** |
| * It is recommended that lifecycle tracking is implemented on all the screens/activities, to ensure an accurate reporting. * Setting the application [MobileCore.setApplication(getApplication());] is only necessary on activities that are entry points for your application. However, setting the application on each Activity has no negative impact and ensures that the SDK always has the necessary reference to your application. We recommend that you call “setApplication” in each of your activities. * You can also use global lifecycle callbacks & and do not need to implement the code for each of the Activity (refer – <https://github.com/Adobe-Marketing-Cloud/aep-sdks-documentation/blob/master/foundation-extensions/mobile-core/lifecycle/lifecycle-extension-in-android.md> ) * You can also set additional data with lifecycle tracking using XDM data variables. (added only in the mainActivity and any other Activity with which your app may be launched) |

**For iOS:**

1.Start lifecyle data collection by calling lifecycleStart: in your app’s -

application:didFinishLaunchingWithOptions:delegate method

**Note:** If your iOS application supports background capabilities, your didFinishLaunchingWithOptions method might be called when iOS launches your app in the background. If you do not want background launches to count towards your lifecycle metrics, then lifecycleStart: should only be called when the application state is not equal to UIApplicationStateBackground.

func application(\_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplicationLaunchOptionsKey: Any]?) -> Bool {

let appState = application.applicationState;

MobileCore.registerExtensions(extensions, {

if appState != .background {

// only start lifecycle if the application is not in the background

MobileCore.lifecycleStart(additionalContextData: [:])

}

}

2. When launched, if your app is resuming from a backgrounded state, iOS might call your applicationWillEnterForeground: delegate method. You also need to call lifecycleStart:, but this time you do not need all of the supporting code that you used in application:didFinishLaunchingWithOptions:

func applicationWillEnterForeground(\_ application: UIApplication) {

MobileCore.lifecycleStart(additionalContextData: [:])

}

3.When the app enters the background, pause Lifecycle data collection from your app's applicationDidEnterBackground: delegate method

func applicationDidEnterBackground(\_ application: UIApplication) {

MobileCore.lifecyclePause()

}

|  |
| --- |
| ***Best Practices for adding lifecycle tracking*** |
| * It is recommended that lifecycle tracking is implemented on all the screens/activities, to ensure an accurate reporting. * Setting the application [MobileCore.setApplication(getApplication());] is only necessary on activities that are entry points for your application. However, setting the application on each Activity has no negative impact and ensures that the SDK always has the necessary reference to your application. We recommend that you call “setApplication” in each of your activities. * You can also use global lifecycle callbacks & and do not need to implement the code for each of the Activity (refer - <https://github.com/Adobe-Marketing-Cloud/aep-sdks-documentation/blob/master/foundation-extensions/mobile-core/lifecycle/lifecycle-extension-in-android.md> ) * You can also set additional data with lifecycle tracking using context data variables. (added only in the mainActivity and any other Activity with which your app may be launched) |

**Tags rules for Lifecycle tracking**

A screenshot of a computer

Description automatically generated

## General guidelines for screen and action tracking

|  |
| --- |
| IMPORTANT NOTE |
| * Generally, when mapping your XDM data in your code it needs to be a 1:1 mapping to your Schema definition. If the Serialized event has multi-level objects, you will need a multi-level map to represent that. * Any custom XDM data variables should be added against a new node in the HashMap – this node should always use the format “\_<Tenant ID>”. The tenant ID can be retrieved from the Experience Cloud Landing Page URL for your organization.   **https://experience.adobe.com/#/@techcombank/home**   * For string variables –   1. Enclose any valid values in double quotes e.g. “product page”   2. Null values to be mentioned as empty string “”. Do not send null as the value for string variables. * For integer/number variables –   1. Do not enclose any number in double quotes   2. In case of null values, omit sending any value. Do not send null as the value for number variables. * For boolean variables –   1. Do not enclose any boolean in double quotes. Do not send null as the value for boolean variables.   2. Pass only true or false * For object variables –   1. Correctly map the XDM path   Please refer to this [link](https://techcombank.sharepoint.com/:x:/r/sites/MS1StrategicInitiative/_layouts/15/Doc.aspx?sourcedoc=%7BAEFCAB5E-D039-4579-924A-141F71EC974A%7D&file=RTCDP_3.2_Data-Mapping_Interim_Approach_v1.0.xlsx&action=default&mobileredirect=true&wdOrigin=TEAMS-ELECTRON.p2p.bim&wdExp=TEAMS-CONTROL&wdhostclicktime=1694616312686&web=1) for RTCDP mapping and data types expected for all variables. |

### Screen Tracking (Equivalent to Page view tracking in web)

To implement page level tracking, we need to use “Edge.sendEvent()” call with event type as “web.webPageDetails.pageViews” to pass all page level custom data. Each time a new screen is displayed in your application (for example, when a user navigates from the home page to the products page), you should send a “Edge.sendEvent()” call with event type as ”web.webPageDetails.pageViews”.

For other custom variables, you need to set XDM objects and provide the key to map the XDM objects via Processing Rules in the Adobe Analytics User Interface.

**Syntax Android**

HashMap < String, Object > customXdmData = new HashMap < String, Object > ();

//Update customXdmData map with custom nodes as per your requirement

Map <String, Object> xdmData = new HashMap <String, Object>();

xdmData.put("<\_tenantID>", customXdmData);

//required if there are any custom XDM data to be sent

xdmData.put("web", new HashMap <String, Object>() {

    {

        put("webPageDetails", new HashMap <String, Object> () {

            {

                put("name", "<page name>");

                put("pageViews", new HashMap <String, Object> () {

                    {

                        put("value", 1);

                    }

                });

            }

        });

    }

});

xdmData.put("eventType", "web.webPageDetails.pageViews");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

    .setXdmSchema(xdmData)

    .build();

Edge.sendEvent(experienceEvent, null);

**Syntax iOS**

var xdmData: [String: Any] = [: ]

xdmData["<\_tenantID>"] = []

//required if there are any custom context data to be sent

xdmData["web"] = [

    "webPageDetails": [

        "pageViews": [

            "value": "1"

        ],

        "name": "[Page Name]",

    ]

]

xdmData["eventType"] = "web.webpagedetails.pageViews"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

|  |
| --- |
| NOTE |
| * The “webPageDetails” value in the “ Edge.sendEvent()” call represents the “pageName” value. If you want to set and eVar or a prop with the page/screen name, then this can be done using the custom XDM data variable or using processing rules. |

### Action Tracking (Equivalent to trackLink/Success event tracking in web)

You should use the “Edge.sendEvent()” call with event type equal to “web.webInteraction.linkClicks” in order to measure your user’s engagement inside your app’s screens. Action represents any micro/macro conversion event that you want to track. Ex: Registrations Complete & Cart Additions.

To implement action tracking, we need to use “Edge.sendEvent()” call with event type as ”web.webInteraction.linkClicks” upon clicking a link or button in your application.

There are three required variables:

* web.webInteraction.name
* web.webInteraction.type
* web.webInteraction.linkClicks.value

The link type can be one of three values:

* other**:** A custom link.
* download**:** A download link.
* exit**:** An exit link.

Further details in the link below:

[**https://experienceleague.adobe.com/docs/experience-platform/edge/data-collection/track-links.html?lang=en**](https://experienceleague.adobe.com/docs/experience-platform/edge/data-collection/track-links.html?lang=en)

**Syntax - Android**

HashMap < String, Object > customXdmData = new HashMap < String, Object > ();

//Update customXdmData map with custom nodes as per your requirement

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("<\_tenantID>", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

    {

        put("webInteraction", new HashMap < String, Object > () {

            {

                put("name", "<link/button name>");

                put("type", "other");

                put("linkClicks", new HashMap < String, Object > () {

                    {

                        put("value", 1);

                    }

                });

            }

        });

    }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

    .setXdmSchema(xdmData)

    .build();

Edge.sendEvent(experienceEvent, null);

**Syntax – iOS**

var xdmData: [String: Any] = [: ]

xdmData["<\_tenantID>"] = [ ]

//if any custom data required

        xdmData["web"] = [

            "webInteraction": [

                "linkClicks": [

                    "value": "1"

                ],

                "name": "[Link Name]",

                "type": "other"

            ]

        ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

**NOTE for developers:**

1. **All XDM values should be in English.**
2. **All XDM values should be in lower case.**
3. **In case of no values, please do not populate the variable.**
4. **The character limit for Analytics variable eVars is 255 bytes. So please let Adobe team know in case you notice any data which is beyond this limit to avoid truncation.**

## Global - Screens

This solution allows the business unit to evaluate the effectiveness of the app’s screens/pages, including their influence on conversion & landing screens. To do that requires that every screen of the app be tagged with an effective screen (page) name and data to represent each level of the content hierarchy.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.pageName, s.eVar11 | Pages/Screens | web.webPageDetails.name |
| s.channel, s.eVar12 | Site Section | web.webPageDetails.siteSection |
| s.eVar6, s.prop3 | Language | \_techcombank.customerOnlineTracking.language |
| s.prop6, s.eVar9 | Business Unit | \_techcombank.customerOnlineTracking.businessUnit |
| s.prop7, s.eVar10 | App Platform & version | \_techcombank.customerOnlineTracking.platform |
| s.prop17, s.eVar57 | User Info status | \_techcombank.loginTracking.loggedInStatus |
| s.eVar2 | User Info Customer ID | \_techcombank.Identities.customerID |
| s.prop1, s.eVar1 | ECID | endUserIDs.\_experience.mcid.id |
| s.prop13 | Screen load time | \_techcombank.pageTracking.pageLoadTime |
| s.eVar0 | Marketing tracking code | marketing.trackingCode |
| s.eVar4 | GAID | \_techcombank.webAppIdentities.GAID |
| s.eVar3 | IDFA | \_techcombank.webAppIdentities.IDFA |
| s.eVar64 | Internal Entry Point | \_techcombank.clickTracking.internalEntryPoint |
| s.prop14 | Percentage page viewed | \_techcombank.pageTracking.percentagePageViewed |
| s.prop10 | Timestamp | \_techcombank.pageTracking.eventTimestamp |
| s.prop16, s.eVar15 | New Repeat Visitor | \_techcombank.customerOnlineTracking.newRepeat |
| s.prop15, s.eVar14 | Previous page name | \_techcombank.pageTracking.previousPagename |
| s.prop18, s.eVar92 | Device General Name | \_techcombank.customerOnlineTracking.deviceGeneralName |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables on every screen and trigger an 'Edge.sendEvent' call (typically on load of every screen):

|  |  |
| --- | --- |
| XDM Data | Notes |
| web.webPageDetails.name | Set a clear, descriptive page/screen name. Consider including the type or category of screen to start the screen name. Continue the screen name with each level of the site hierarchy until the final descriptive portion of the screen. Use colon ":" as the delimiter between each portion of the page. |
| web.webPageDetails.siteSection | Set this variable to the top-level category |
| \_techcombank.customerOnlineTracking.businessUnit | Set this variable to business unit e.g., “RGB”, “BB”, etc. |
| \_ techcombank.customerOnlineTracking.platform | Set this variable to “App-Android [version]” or “App-iOS [version]”. |
| \_ techcombank.customerOnlineTracking.deviceGeneralName | Set this variable to device general name |
| \_techcombank.loginTracking.loggedInStatus | Set this variable to the logged in status of user “logged-in”, “not-logged-in” |
| \_techcombank.Identities.customerID | Set this variable to Capture customer ID (generated by ROC backend) at login success (Hash it only if required) |
| endUserIDs.\_experience.mcid.id | Set this variable to Adobe Experience Cloud ID. |
| \_techcombank.pageTracking.pageLoadTime | Set this variable to time taken to load a screen. |
| marketing.trackingCode | Set this variable to Marketing tracking code. |
| \_techcombank.customerOnlineTracking.language | Set this variable to Language. |
| \_techcombank.customerOnlineTracking.environment | Set this variable to Environment (eg. dev, stg, prod, etc) |
| \_techcombank.pageTracking.percentagePageViewed | Set this to percentage page viewed |
| \_techcombank.webAppIdentities.GAID | Set this to GAID |
| \_techcombank.webAppIdentities.IDFA | Set this to IDFA |
| \_techcombank.clickTracking.internalEntryPoint | Set this to organic- Push notification, in-app message, nudge, dashboard banner, saving landing, savings list + sign. product finding method, promotions, "Discover Product" from the dashboard menu |
| \_techcombank.pageTracking.eventTimestamp | Set this to Timestamp |
| \_techcombank.pageTracking.previousPagename | Set this to previous page name |
| \_techcombank.customerOnlineTracking.newRepeat | Set this to New or Repeat visitor |

|  |
| --- |
| ***Best Practices for Screen (Page) Names*** |
| * **Context -** Include the directory structure or content hierarchy in the screen name to help users understand where the screen "lives" within the site. This also allows for simplified report filtering. * **Clarity -** Ensure the screen name is clear and easily identifiable for infrequent users. This will promote faster adoption of Adobe Analytics by business users and allow for easier, more efficient reporting. * **Conciseness -** Keep the screen name as short as possible to maximize limited character space. This is also important for clean, simple reports but should not take precedence over context and clarity. * Suggested page naming convention:   **<business Unit>:<platform>:<Country>:<Language>:<Page Title>**  **Example- rbg:android:vn:en:home**  **bb:ios:vn:en:trade** |

**Android XDM Data Format**

HashMap < String, Object > customXdmData = new HashMap < String, Object > ();

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

  put("language", "[LANGUAGE]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("newRepeat", "[NEW/REPEAT VISITOR]");

});

customXdmData.put("endUserIDs", new HashMap < String, Object > () {

  {

    put("\_experience", new HashMap < String, Object > () {

      {

        put("mcid", new HashMap < String, Object > () {

          {

            put("id", "EXPERIENCE CLOUD ID"); //Steps to capture mentioned in sec 6.

          }

        });

      }

    });

  }

});

customXdmData.put("webAppIdentities", new HashMap < String, Object > () {

  put("IDFA", "[IDFA]");

  put("GAID", "[GAID]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("pageLoadTime", "[SCREEN LOAD TIME]");

  put("percentagePageViewed", "[PERCENTAGE PAGE VIEWED]");

put("customPageview", 1);

put("previousPageName", "[PREVIOUS PAGE NAME]");

// pass it on next page load

put("eventTimestamp", "[TIME STAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

customXdmData.put("loginTracking", new HashMap < String, Object > () {

  put("loggedInStatus", "[LOGGED IN STATUS]");

});

customXdmData.put("Identities", new HashMap < String, Object > () {

  put("customerID", "[CUSTOMER ID]");

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

// if marketing code exists

xdmData.put("marketing", new HashMap < String, Object > () {

  {

    put("trackingCode", "[SET MARKETING TRACKING CODE]"); // structure of the code source:medium:campaign:content(need to confirm)

  }

});

// Add this section if it is a product page

customXdmData.put("productsViewOnline", new HashMap < String, Object > () {

  put("productName", "[PRODUCT NAME]");

  put("productViews", 1);

});

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("internalEntryPoint", "[INTERNAL ENTRY POINT]");

  // example - organic- Push notification, in-app message, nudge, dashboard banner, saving landing, savings list + sign. product finding method, promotions, "Discover Product" from the dashboard menu

});

// End of Product section

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webPageDetails", new HashMap < String, Object > () {

      {

        put("name", "[APP SCREEN NAME]");

        put("siteSection", "[APP SECTION NAME]");

        put("server", "[SERVER NAME]");

        put("pageViews", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webPageDetails.pageViews");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [:]

xdmData["\_techcombank"] = ["customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

    "language": "[LANGUAGE]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "newRepeat": "[NEW/REPEAT VISITOR]"

    ]]

xdmData["endUserIDs"] = ["\_experience": ["mcid": ["id": "[EXPERIENCE CLOUD ID]"]]] //Steps to capture mentioned in sec 6.

xdmData["\_techcombank"] = ["pageTracking": [

"pageLoadTime": "[SCREEN LOAD TIME]",

"previousPageName": "[PREVIOUS PAGE NAME]",

"customPageview": 1,

"percentagePageViewed": "[PERCENTAGE PAGE VIEWED]"

// pass it on next page load

"eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["\_techcombank"] = ["loginTracking": ["loggedInStatus": "[LOGGED IN STATUS]"]    ]

xdmData["\_techcombank"] = ["Identities": ["customerID": "[CUSTOMER ID]"]]

xdmData["\_techcombank"] = ["webAppIdentities": [

    "IDFA": "[IDFA]",

    "GAID": "[GAID]"

    ]]

xdmData["\_techcombank"] = ["clickTracking": [

    " internalEntryPoint ": "[ INTERNAL ENTRY POINT]",

// example - organic- Push notification, in-app message, nudge, dashboard banner, saving landing, savings list + sign. product finding method, promotions, "Discover Product" from the dashboard menu

  ]]

xdmData["web"] = [

    "webPageDetails": [

        "name": "[SCREEN NAME]",

        "siteSection": "[APP SECTION NAME]",

        "server": "[SERVER NAME]",

        "pageViews": ["value": 1]

    ]

]

xdmData["eventType"] = "web.webpagedetails.pageViews"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar11 | Screen Name | a.x.\_ web.webPageDetails.name |
| eVar12/s.channel | Site Section | a.x.\_ web.webPageDetails.siteSection |
| eVar9/prop6 | Business Unit | a.x.\_techcombank.customerOnlineTracking.businessUnit |
| eVar1/prop1 | Platform | a.x.\_techcombank.customerOnlineTracking.platform |
| prop10 | Day of week | a.x.\_techcombank.weekdayWeekend |
| eVar57/prop17 | User Info status | a.x.\_techcombank.loginTracking.loggedInStatus |
| eVar2 | User Info Customer Id | a.x.\_techcombank.Identities.customerID |
| eVar1/prop1 | ECID | a.x.\_endUserIDs.\_experience.mcid.id |
| prop13 | Screen Load time | a.x.\_techcombank.pageTracking.pageLoadTime |
| eVar0 | Marketing tracking code | a.x.\_marketing.trackingCode |
| eVar6/prop3 | Language | a.x.\_customerOnlineTracking.language |
| eVar8/prop5 | Environment | a.x.\_techcombank.customerOnlineTracking.environment |
| s.eVar4 | GAID | a.x.\_techcombank.webAppIdentities.GAID |
| s.eVar3 | IDFA | a.x.\_techcombank.webAppIdentities.IDFA |
| s.eVar64 | Internal entry point | a.x.\_techcombank.clickTracking.internalEntryPoint |
| s.prop14 | Percentage page viewed | a.x.\_techcombank.pageTracking.percentagePageViewed |
| s.event1 | Custom pageview | a.x.\_techcombank.pageTracking.customPageview |
| eVar15, prop16 | New Repeat visitor | a.x.\_techcombank.customerOnlineTracking.newRepeat |
| eVar14, prop15 | Previous page name | a.x.\_techcombank.pageTracking.previousPagename |
| eVar92, prop18 | Device general name | a.x.\_techcombank.customerOnlineTracking.deviceGeneralName |

## Global - Custom Link Tracking (Link/Button)

Set the following on the interaction of the header/footer clicks, internal (ex. Tools and Utilities)/outbound links (ex. Social Network in Footer), name of the link and location on the page.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar20 | Link Name | \_techcombank.clickTracking.linkName |
| eVar6/prop3 | Language | a.x.\_customerOnlineTracking.language |
| eVar8/prop5 | Environment | a.x.\_techcombank.customerOnlineTracking.environment |
| eVar9/prop6 | Business Unit | a.x.\_techcombank.customerOnlineTracking.businessUnit |
| eVar92, prop18 | Device general name | a.x.\_techcombank.customerOnlineTracking.deviceGeneralName |
| s.prop18, s.eVar92 | Device General Name | \_techcombank.customerOnlineTracking.deviceGeneralName |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.clickTracking.linkName | Set this to the name of button/link/Call-to-Action interacted with |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the button/link/Call-to-Action interacted with is present |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("linkName", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

  put("language", "[LANGUAGE]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

put("eventTimestamp", "[TIME STAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = "clickTracking": [

    "linkName": ["NAME OF BUTTON/LINK/CTA INTERACTED WITH"]

      ]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

"customPagename": "[SCREEN NAME]",

"eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar20 | Link Name | a.x.\_ techcombank..clickTracking.linkName |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| event7 | Link Clicks | a.x.\_web.webinteraction.linkclicks.value |
| s.prop18, s.eVar92 | Device General Name | a.x.\_techcombank.customerOnlineTracking.deviceGeneralName |

## Global – Login/Logout

This solution allows the business to capture user's login / logout actions.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| Event42 | Logout action | \_techcombank.loginTracking.signOutEvent |
| Event39 | Login action | \_techcombank.loginTracking.loginComplete |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.loginTracking.signOutEvent | Set this to 1 |
| \_techcombank.loginTracking.loginComplete | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("linkName", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

});

customXdmData.put("loginTracking", new HashMap < String, Object > () {

  put("loginComplete", 1); // if login success

  put("signOutEvent", 1); // if logout success

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = "clickTracking": [

    "linkName": ["NAME OF BUTTON/LINK/CTA INTERACTED WITH"]

      ]

xdmData["\_techcombank"] = ["loginTracking": [

    "loginComplete": 1, // if login success

    "signOutEvent": 1 // if logout success

]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

"customPagename": "[SCREEN NAME]",

"eventTimestamp": "[TIMESTAMP]",

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| Event42 | Logout action | a.x.\_techcombank.loginTracking.signOutEvent |
| Event39 | Login action | a.x.\_techcombank.loginTracking.loginComplete |

## Global – Banner

This solution allows the business to capture interactions on the hero, promo or offer banner.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar19 | Banner Name | \_techcombank.clickTracking.bannerName |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.clickTracking.bannerName | Set this to the name of banner interacted with. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.clickTracking.bannerClicks | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("bannerName", "[NAME OF BANNER INTERACTED WITH]");

  put("bannerClicks", 1);

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

put("eventTimestamp", "[TIME STAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

     xdmData["\_techcombank"] = ["clickTracking": [

        "bannerName": "[NAME OF BANNER INTERACTED WITH]",

        "bannerClicks": 1

    ]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

"customPagename": "[SCREEN NAME]",

"eventTimestamp": "[TIMESTAMP]",

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar19 | Banner Name | a.x.\_techcombank.clickTracking.bannerName |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| event6 | Banner clicked | a.x\_techcombank.clickTracking.bannerClicks |

## Global – Carousel

This solution allows the business to capture interactions with Carousel.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar23 | Carousel Name | \_techcombank.carouselActivity.carouselName |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM data | Notes |
| \_techcombank.carouselActivity.carouselName | Set this to the name of carousel along with position, separated by a pipe delimiter. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.carouselActivity.carouselClicks | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("carouselActivity", new HashMap < String, Object > () {

  put("carouselName", "[NAME OF CAROUSEL & POSITION]");

  put("carouselClicks", 1);

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("eventTimestamp", "[TIME STAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [:]

xdmData["\_techcombank"] = ["carouselActivity": [

    "carouselName": "[NAME OF CAROUSEL & POSITION]",

    "carouselClicks": 1

    ]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

"customPagename": "[SCREEN NAME]",

"eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar19 | Carousel Name | a.x.\_ techcombank.carouselActivity.carouselName |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| event10 | Carousel clicked | a.x.\_techcombank.carouselActivity.carouselClicks |

## Global – Calculator

This solution allows the business to capture interactions with Calculators.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar24 | Calculator Name | \_techcombank.calculatorActivity.calculatorName |
| s.eVar88 | Calculator Fields & Results | \_techcombank.calculatorActivity.calculatorFields |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.calculatorActivity.calculatorName | Set this to the name of calculator along with details like field values & results, separated by a pipe delimiter. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.calculatorActivity.calculatorInteractions | Set this to 1. |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("calculatorActivity", new HashMap < String, Object > () {

  put("calculatorName", "[NAME OF CALCULATOR]");

  put("calculatorInteractions", 1);

  put("calculatorFields", "[FIELDS VALUES & RESULT]"); //separated by pipe delimiter

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [:]

xdmData["\_techcombank"] = ["calculatorActivity": [

    "calculatorName": "[NAME OF CALCULATOR]",

    "calculatorFields": "[FIELDS VALUES & RESULT]", //separated by pipe delimiter

    "calculatorInteractions": 1]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": ["customPagename": "[SCREEN NAME]"]]

xdmData["web"] = [

    "webInteraction": [

    "name": "[SCREEN NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar24 | Calculator Name | a.x.\_techcombank.calculatorActivity.calculatorName |
| s.eVar88 | Calculator Fields & Results | a.x.\_techcombank.calculatorActivity.calculatorFields |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| event11 | Calculator clicked | a.x.\_techcombank.calculatorActivity.calculatorInteractions |

## Global – Downloads

This solution allows the business to capture File downloads.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar21 | File Name | \_techcombank.fileDownload.downloadedFileName |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.fileDownload.downloadedFileName | Set this to the name of the file downloaded. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.fileDownload.fileDownloads | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("fileDownload", new HashMap < String, Object > () {

  put("downloadedFileName", "[NAME OF DOWNLOADED FILE]");

  put("fileDownloads", 1);

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("eventTimestamp", "[TIME STAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "download");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = ["\_fileDownload": ["downloadedFileName": "[NAME OF DOWNLOADED FILE]", "fileDownloads": 1]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

"customPagename": "[SCREEN NAME]",

"eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar21 | File Name | a.x.\_techcombank.fileDownload.downloadedFileName |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| Event8 | File Downloaded | a.x.\_techcombank.fileDownload.fileDownloads |

## Global – FAQs

This solution allows the business to capture FAQs interaction.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar18 | FAQ Name | \_techcombank.clickTracking.FAQTitle |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.clickTracking.FAQTitle | Set this to the name of the FAQ clicked or FAQ accordion is opened. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.clickTracking.FAQClicks | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("FAQTitle", "[NAME OF FAQ]");

  put("FAQClicks", 1);

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [:]

xdmData["\_techcombank"] = ["clickTracking": ["FAQTitle": "[NAME OF FAQ]", "FAQClicks": 1]]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

   "customPagename": "[SCREEN NAME]",

   "eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar18 | FAQ Name | a.x.\_techcombank.clickTracking.FAQTitle |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| event5 | FAQ clicked | a.x.\_techcombank.clickTracking.FAQClicks |

## Global – Navigation clicks

This solution allows the business to capture Navigation click interactions.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| eVar17 | Global Nav Menu Name | \_techcombank.pageTracking.GlobalNavigationMenu |
| eVar11 | Screen Name | \_techcombank.pageTracking.customPagename |
| event4 | Global Nav Menu clicked | \_techcombank.clickTracking.globalNavClicks |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.pageTracking.GlobalNavigationMenu | Set this to the name of the menu clicked on global Navigation. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.clickTracking.globalNavClicks | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("linkName", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("GlobalNavigationMenu", "[MENU NAME]");

  put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = "clickTracking": [

    "linkName": ["NAME OF BUTTON/LINK/CTA INTERACTED WITH"]

      ]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

    "customPagename": "[SCREEN NAME]",

    "GlobalNavigationMenu": "[MENU NAME]",

"eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

    ]

    ]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": "1"] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar17 | Global Nav Menu Name | a.x.\_techcombank.pageTracking.GlobalNavigationMenu |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| event4 | Global Nav Menu clicked | a.x\_techcombank.clickTracking.globalNavClicks |

## Global – Inbox notifications.

This solution allows the business to capture Inbox notification click interactions.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar61 | Notification Title | \_techcombank.AppNotification.appNotificationMsg |
| s.eVar38 | Notification Type | \_techcombank.AppNotification.appNotificationType |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.AppNotification.appNotificationMsg | Set this to the title of the notification clicked. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.AppNotification.appNotificationType | Set this to the Notification type |
| \_techcombank.clickTracking.appNotificationClick | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("AppNotification", new HashMap < String, Object > () {

  put("appNotificationMsg", "[TITLE OF NOTIFICATION]");

  put("appNotificationType", "[NOTIFICATION TYPE]");

});

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("appNotificationClick", 1);

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = ["AppNotification": ["appNotificationMsg": "[TITLE OF NOTIFICATION]", "appNotificationType": "[NOTIFICATION TYPE]"]]

xdmData["\_techcombank"] = ["clickTracking": ["appNotificationClick": 1]]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

   "customPagename": "[SCREEN NAME]",

   "eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar61 | Notification Title | a.x.\_techcombank.AppNotification.appNotificationMsg |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| eVar38 | Notification Type | a.x.\_techcombank.AppNotification.appNotificationType |
| event71 | Notification clicked | a.x\_techcombank.clickTracking.appNotificationClick |

## Global – Tab.

This solution allows the business to capture Tab interactions.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar20 | Tab name | \_techcombank.clickTracking.linkName |

### Deployment Instructions [To do: TCB IT Team]

Set the following variables XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.clickTracking.linkName | Set this to the name of the Tab clicked. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.clickTracking.linkClicks | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("linkName", "[NAME OF TAB]"); //ex- TAB-<TAB NAME>

  put("linkClicks", 1);

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = ["clickTracking": [

    "linkName": ["NAME OF BUTTON/LINK/CTA INTERACTED WITH"]  //ex- TAB-<TAB NAME>

      ]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

   "customPagename": "[SCREEN NAME]",

   "eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar20 | Tab name | a.x.\_techcombank.clickTracking.linkName |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| event7 | Tab clicked | a.x.\_techcombank.clickTracking.linkClicks |

## Global – Error tracking.

This solution allows the business to capture all errors occurred in any Product/Journey/Global pages.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar16 | Error Type | \_techcombank.error.errorType |
| s.eVar28 | Error Code with Message | \_techcombank.error.errorMessageWithCode |
| s.eVar40 | Journey Name | \_techcombank.journeyTracking.journeyName |
| s.eVar53 | Journey Step Name | \_techcombank.journeyTracking.journeyStepName |
| s.eVar52 | Product Name | \_techcombank.productViewsOnline.productName |
| Event43 | Error | \_techcombank.error.error |

### Deployment Instructions [To do: TCB IT Team]

Set the following variables XDM variable :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.error.errorType | Set this to Error type |
| \_techcombank.error.errorMessageWithCode | Set this to Error code and error message separated by a pipe delimiter (|) |
| \_techcombank.journeyTracking.journeyName | Set this to the name of the Journey where this error occurred. |
| \_techcombank.journeyTracking.journeyStepName | Set this to the step name of the Journey where this error occurred. |
| \_techcombank.productViewsOnline.productName | Set this to the product name where this error occurred. |
| \_techcombank.error.error | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("error", new HashMap < String, Object > () {

  put("errorType", "[TYPE OF ERROR]");

  put("errorMessageWithCode", "[ERROR MESSAGE WITH CODE]");

put("error", 1);

});

//if applicable

customXdmData.put("journeyTracking", new HashMap < String, Object > () {

  put("journeyName", "[JOURNEY NAME]");

  put("journeyStepName", "[JOURNEY STEP NAME]");

});

//if applicable

customXdmData.put("productsViewOnline", new HashMap < String, Object > () {

  put("productName", "[PRODUCT NAME]");

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = ["error":[

    "errorType": "[TYPE OF ERROR]",

    "errorMessageWithCode": "[ERROR MESSAGE WITH CODE]",

"error": 1

]

]

//if applicable

xdmData["\_techcombank"] = ["journeyTracking": ["journeyName": "[JOURNEY NAME]", "journeyStepName": "[JOURNEY STEP NAME]"]

]

//if applicable

xdmData["\_techcombank"] = ["productsViewOnline": ["productName": "[PRODUCT NAME]"]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

   "customPagename": "[SCREEN NAME]",

   "eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| s.eVar11 | Page Name/Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| s.eVar16 | Error Type | a.x.\_techcombank.error.errorType |
| s.eVar28 | Error Code with Message | a.x.\_techcombank.error.errorMessageWithCode |
| s.eVar40 | Journey Name | a.x.\_techcombank.journeyTracking.journeyName |
| s.eVar53 | Journey Step Name | a.x.\_techcombank.journeyTracking.journeyStepName |
| s.eVar52 | Product Name | a.x.\_techcombank.productViewsOnline.productName |

## Global – Time out.

This solution allows the business to capture login timeouts.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| event41 | Time out | \_techcombank.loginTracking.loginTimeout |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.loginTracking.loginTimeout | Set this to 1 when timeouts occur. |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("loginTracking", new HashMap < String, Object > () {

  put("loginTimeout", 1);

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = "loginTracking": ["loginTimeout": 1]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

   "customPagename": "[SCREEN NAME]",

   "eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| event41 | Login Time Out | a.x.\_techcombank.loginTracking.loginTimeout |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |

## eKYC – Journey

This solution allows the business to capture eKYC specific journey details.

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM data variables should be additionally added along with the global XDM data variables for all screens.

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar7 | eKYC Journey Name | \_techcombank.eKYC.eKYCJourney |
| s.eVar70 | eKYC Journey Step Name | \_techcombank.eKYC.eKYCStepname |
| s.eVar71 | eKYC hashed email | \_techcombank.eKYC.eKYCemail |
| s.list1 | eKYC Demographic info | \_techcombank.eKYC.eKYCDemographicInfo |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM data variables | Notes |
| \_techcombank.eKYC.eKYCJourney | Set this variable to “eKYC Journey” |
| \_techcombank.eKYC.eKYCStepname | Set this variable to the step name of eKYC Journey |
| \_techcombank.eKYC.eKYCemail | Set this variable to hashed email id used during eKYC journey |
| \_techcombank.eKYC.eKYCDemographicInfo | Set this variable to capture demographic information like gender, occupation, position, salary separated by a pipe delimiter(|). |
| \_techcombank.eKYC.eKYCJourneyStart | Set this to 1 when eKYC journey starts. |
| \_techcombank.eKYC.eKYCjourneyComplete | Set this to 1 when eKYC journey completes. |
| \_techcombank.eKYC.eKYCjourneyFail | Set this to 1 when eKYC journey fails. |

**Android XDM Data Format**

// Add the below custom xdm data along with the screen load custom xdm data as in sec 4.3

customXdmData.put("eKYC", new HashMap < String, Object > () {

  put("eKYCJourney", "eKYC Journey");

  put("eKYCStepName", "[eKYC JOURNEY STEP NAME]");

  put("eKYCDemographicInfo", "[eKYC DEMOGRAPHIC INFO]");

  put("eKYCJourneyStart", 1); //set when eKYC Journey starts

  put("eKYCjourneyComplete", 1); //set when eKYC Journey completes

  put("eKYCjourneyFail", 1); //set when eKYC Journey fails (need to confirm)

});

**iOS - XDM DATA Format**

// Add the below custom xdm data along with the screen load custom xdm data as in sec 4.3

xdmData["\_techcombank"] = [

    "eKYC": [

        "eKYCJourney": "[eKYC Journey]",

        "eKYCStepName": "[eKYC JOURNEY STEP NAME]",

        "eKYCDemographicInfo": "[eKYC DEMOGRAPHIC INFO]",

        "eKYCJourneyStart": 1, //set when eKYC Journey starts

        "eKYCjourneyComplete": 1, //set when eKYC Journey completes

        "eKYCjourneyFail": 1 //set when eKYC Journey fails (need to confirm)

    ]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar7 | eKYC Journey Name | a.x.\_techcombank.eKYC.eKYCJourney |
| eVar70 | Journey step name | a.x.\_techcombank.eKYC.eKYCStepname |
| eVar71 | eKYC hashed email id | a.x.\_techcombank.eKYC.eKYCemail |
| List1 | eKYC Demographic Info | a.x.\_techcombank.eKYC.eKYCDemographicInfo |
| Event50 | eKYC Journey Start | a.x.\_techcombank.eKYC.eKYCJourneyStart |
| Event51 | eKYC Journey Complete | a.x.\_techcombank.eKYC.eKYCjourneyComplete |
| Event52 | eKYC Journey Fail | a.x.\_techcombank.eKYC.eKYCjourneyFail |

## eKYC video – Journey

This solution allows the business to capture video eKYC specific journey details.

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM data variables should be additionally added along with the global XDM data variables for all screens.

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar7 | eKYC Video Journey Name | \_techcombank.eKYC.eKYCJourney |
| s.eVar70 | eKYC Video Journey Step Name | \_techcombank.eKYC.eKYCStepname |
| s.eVar72 | eKYC Video Journey Call Duration | \_techcombank.eKYC.eKYCVideocallduration |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.eKYC.eKYCJourney | Set this variable to “eKYC Video Journey” |
| \_techcombank.eKYC.eKYCStepname | Set this variable to the step name of eKYC Video Journey |
| \_techcombank.eKYC.eKYCVideocallduration | Set this variable to eKYC video journey call duration |

**Android XDM Data Format**

//Add the below custom xdm data along with the page load custom xdm data as mentioned in sec 4.3

customXdmData.put("eKYC", new HashMap < String, Object > () {

  put("eKYCJourney", "eKYC Video Journey");

  put("eKYCStepName", "[eKYC VIDEO JOURNEY STEP NAME]");

  put("eKYCvideocallduration", "[eKYC VIDEO CALL DURATION]");

  put("videoeKYCStart", 1); //set when video eKYC starts

  put("videoeKYCComplete", 1); ////set when video eKYC completes

  put("videoeKYCFail", 1); ////set when video eKYC fails

});

**iOS - XDM DATA Format**

//Add the below custom xdm data along with the page load custom xdm data as mentioned in sec 4.3

xdmData["\_techcombank"] = ["eKYC": [

    "eKYCJourney": "eKYC Video Journey",

    "eKYCStepName": "[eKYC VIDEO JOURNEY STEP NAME]",

    "eKYCvideocallduration": "[eKYC VIDEO CALL DURATION]",

    "videoeKYCStart": 1, //set when video eKYC starts

    "videoeKYCComplete": 1, //set when video eKYC completes

    "videoeKYCFail": 1, //set when video eKYC fails

]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar7 | eKYC Journey Name | a.x.\_techcombank.eKYC.eKYCJourney |
| eVar70 | Journey step name | a.x\_techcombank.eKYC.eKYCStepname |
| eVar72 | eKYC video call duration | a.x.\_techcombank.eKYC.eKYCVideocallduration |
| Event53 | Video eKYC Journey Start | a.x\_techcombank.eKYC.videoeKYCStart |
| Event54 | Video eKYC Journey Complete | a.x\_techcombank.eKYC.videoeKYCComplete |
| Event55 | Video eKYC Journey Fail | a.x\_techcombank.eKYC.videoeKYCFail |

## eKYC – Video call start/end.

This solution allows the business to capture the number of times video call started and ended in Video eKYC journey.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| event56 | eKYC video call start | \_techcombank.eKYC.eKYCVideoCallStart |
| event57 | eKYC video call end | \_techcombank.eKYC.eKYCVideoCallEnd |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.eKYC.eKYCVideoCallStart | Set this to 1 when video call starts |
| \_techcombank.eKYC.eKYCVideoCallEnd | Set this to 1 when video call ends |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("eKYC", new HashMap < String, Object > () {

  put("eKYCVideoCallStart", 1); // when the video call starts

  put("eKYCVideoCallEnd", 1); // when the video call ends

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = ["eKYC": [

    "eKYCVideoCallStart": 1, // when the video call starts

    "eKYCVideoCallEnd": 1 // when the video call ends

]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

   "customPagename": "[SCREEN NAME]",

   "eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| event56 | eKYC Video Call Start | a.x.\_ techcombank.eKYC.eKYCVideoCallStart |
| event57 | eKYC Video Call End | a.x.\_ techcombank.eKYC.eKYCVideoCallEnd |

## All Products – Journey Start/Complete/Fail

This solution allows the business to capture all product journey details (ex – Auto Loan, Banca, etc.).

### Variables in this Section [To do: TCB IT Team]

The below table consists of variables used for this solution. The below mentioned XDM data variables should be additionally added along with the global XDM data variables for all screens.

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar40 | Journey Name | \_techcombank.journeyTracking.journeyName |
| s.eVar53 | Journey Step Name | \_techcombank.journeyTracking.journeyStepName |
| s.eVar52 | Product Name | \_techcombank.productViewsOnline.productName |
| event19 | Journey start | \_techcombank.journeyTracking.journeyStart |
| event20 | Journey complete | \_techcombank.journeyTracking.journeyComplete |
| event21 | Journey fail | \_techcombank.journeyTracking.journeyFail |

### Deployment Instructions

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.journeyTracking.journeyName | Set this variable to Journey Name |
| \_techcombank.journeyTracking.journeyStepName | Set this variable to the step name of the Journey |
| \_techcombank.productViewsOnline.productName | Set this variable to Product name related to the journey. |
| \_techcombank.journeyTracking.journeyStart | Set this variable to 1 when the journey starts. |
| \_techcombank.journeyTracking.journeyComplete | Set this variable to 1 when the journey completes. |
| \_techcombank.journeyTracking.journeyFail | Set this variable to 1 when the journey fails. |

**Android XDM Data Format**

// Add the following custom XDM data along with global screen variables as mentioned in sec 4.3

customXdmData.put("journeyTracking", new HashMap < String, Object > () {

  put("journeyName", "[JOURNEY NAME]");

  put("journeyStepName", "[JOURNEY STEP NAME]");

  put("journeyStart", 1); //if Journey starts

  put("journeyComplete", 1); //if Journey completes

  put("journeyFail", 1); //if Journey fails

});

**iOS - XDM DATA Format**

// Add the following custom XDM data along with global screen variables as mentioned in sec 4.3

xdmData["\_techcombank"] = ["journeyTracking": [

  "journeyName": "[JOURNEY NAME]",

  "journeyStepName": "[JOURNEY STEP NAME]",

  "journeyStart": 1, //if Journey starts

  "journeyComplete": 1, //if Journey completes

  "journeyFail": 1 //if Journey fails

]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| s.eVar40 | Journey Name | a.x.\_techcombank.journeyTracking.journeyName |
| s.eVar53 | Journey Step Name | a.x.\_techcombank.journeyTracking.journeyStepName |
| s.eVar52 | Product Name | a.x.\_techcombank.productViewsOnline.productName |
| event19 | Journey start | a.x.\_techcombank.journeyTracking.journeyStart |
| event20 | Journey complete | a.x.\_techcombank.journeyTracking.journeyComplete |
| event21 | Journey fail | a.x.\_techcombank.journeyTracking.journeyFail |

## All Product pages CTA clicks.

This solution allows the business to capture all CTA clicks present on Products pages only.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar40 | Journey Name | \_techcombank.journeyTracking.journeyName |
| s.eVar52 | Product Name | \_techcombank.productViewsOnline.productName |
| s.eVar53 | Journey Step name | \_techcombank.journeyTracking.journeyStepName |
| s.eVar55 | Product link/CTA name | \_techcombank.productViewsOnline.productLinkButtonName |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the interacted banner is present |
| \_techcombank.journeyTracking.journeyName | Set this to the name of the Journey where the CTA button was clicked |
| \_techcombank.productViewsOnline.productName | Set this to the name of the product where the CTA button was clicked. |
| \_techcombank.journeyTracking.journeyStepName | Set this to the Journey Step name where the CTA button was clicked. |
| \_techcombank.productViewsOnline.productLinkButtonName | Set this to the name of the CTA button that was clicked. |
| \_techcombank.clickTracking.productPageClicks | Set this to 1. |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("productsViewOnline", new HashMap < String, Object > () {

  put("productName", "[PRODUCT NAME]");

  put("productLinkButtonName", "[PRODUCT LINK/BUTTON NAME]"); //eg. ProductName|Link Name

});

customXdmData.put("clickTracking", new HashMap < String, Object > () {

    put("productPageClicks", 1);

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("language", "[LANGUAGE]");

});

customXdmData.put("journeyTracking", new HashMap < String, Object > () {

  put("journeyName", "[JOURNEY NAME]");

  put("journeyStepName", "[JOURNEY STEP NAME]"); //eg. ProductName|Journey Step Name

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = ["productsViewOnline": [

    "productName": "[PRODUCT NAME]",

    "productLinkButtonName": "[PRODUCT LINK/BUTTON NAME]" //eg. ProductName|Link Name

]

]

xdmData["\_techcombank"] = ["clickTracking": [ "productPageClicks": 1]

]

xdmData["\_techcombank"] = ["journeyTracking": [

    "journeyName": "[JOURNEY NAME]",

    "journeyStepName": "[JOURNEY STEP NAME]" //eg. ProductName|Journey Step Name]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

    "language": "[LANGUAGE]"

      ]

xdmData["\_techcombank"] = ["pageTracking": [

   "customPagename": "[SCREEN NAME]",

   "eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| s.eVar40 | Journey Name | a.x\_techcombank.journeyTracking.journeyName |
| s.eVar52 | Product Name | a.x\_techcombank.productViewsOnline.productName |
| s.eVar53 | Journey Step name | a.x\_techcombank.journeyTracking.journeyStepName |
| s.eVar55 | Product link/CTA name | a.x\_techcombank.productViewsOnline.productLinkButtonName |
| event34 | Product pages CTA clicks | a.x\_techcombank.clickTracking.productPageClicks |

## Auto Loan

This solution allows the business to capture the loan amount, term and interest rate at the end of the journey.

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM variables should be additionally added along with the global XDM data variables for all Product Journey flows [Sec 4.19].

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| Event46 | Loan Amount | \_techcombank.Loan.loanAmount |
| eVar59 | Term | \_techcombank.Loan.loanTerm |
| eVar60 | Interest rate | \_techcombank.Loan.loanInterestRate |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.Loan.loanAmount | Set this variable to loan amount. |
| \_techcombank.Loan.loanTerm | Set this variable to loan term. |
| \_techcombank.Loan.loanInterestRate | Set this variable to loan interest rate. |

**Android XDM Data Format**

// Add this custom XDM data along with the existing XDM data in sec 4.19 on confirmation page

customXdmData.put("Loan", new HashMap < String, Object > () {

  put("loanAmount", "[LOAN AMOUNT]");

  put("loanTerm", "[LOAN TERM]");

  put("loanInterestRate", "[LOAN INTEREST RATE]");

});

**iOS - XDM DATA Format**

// Add this custom XDM data along with the existing XDM data in sec 4.19 on confirmation page

xdmData["\_techcombank"] = ["loan": [

    "loanAmount": "[LOAN AMOUNT]",

    "loanTerm": "[LOAN TERM]",

    "loanInterestRate": "[LOAN INTEREST RATE]"

    ]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| Event46 | Loan amount | \_techcombank.Loan.loanAmount |
| eVar59 | Loan Term | \_techcombank.Loan.loanTerm |
| eVar60 | Loan interest rate | \_techcombank.Loan.loanInterestRate |

## Flexible Term Deposit .

This solution allows the business to capture -

* 1. amount, term and interest rate and maturity method at the journey completion
  2. when withdraw happens after fulfilling deposit period and amount
  3. what point of time (=how many months to go) user is terminating and termination amount
  4. if user makes changes in Saving Plan (ex. Auto Roll etc.)

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM variables should be additionally added along with the global XDM data variables for all Product Journey flows [Sec 4.19].

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| eVar75 | Flexible Term Deposit term | \_techcombank.flexibleTermDeposit.flexibleTermDepositTerm |
| eVar76 | Flexible Term Deposit interest rate | \_techcombank.flexibleTermDeposit.flexibleTermDepositInterestRate |
| eVar77 | Flexible Term Deposit maturity method | \_techcombank.flexibleTermDeposit.flexibleTermDepositMaturityMethod |
| Event47 | Flexible Term Deposit period | \_techcombank.flexibleTermDeposit.flexibleTermDeposit |
| eVar78 | Flexible Term Deposit period | \_techcombank.flexibleTermDeposit.flexibleTermDepositperiod |
| Event59 | Flexible Term Deposit Withdraw Amount | \_techcombank.flexibleTermDeposit.flexibleTermDepositWithdrawAmount |
| Event60 | Flexible Term Deposit Withdraw | \_techcombank.flexibleTermDeposit.flexibleTermDepositWithdraw |
| eVar79 | Flexible Term Deposit termination term | \_techcombank.flexibleTermDeposit.flexibleTermDepositTerminationTerm |
| Event61 | Flexible Term Deposit Termination Amt | \_techcombank.flexibleTermDeposit.flexibleTermDepositTerminationAmt |
| eVar52 | Product Name | \_techcombank.productViewsOnline.productName |
| Event35 | Saving Plan Change | \_techcombank.transactionWebApp.savingPlanChange |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.flexibleTermDeposit.flexibleTermDepositTerm | Set this variable to Term |
| \_techcombank.flexibleTermDeposit.flexibleTermDepositInterestRate | Set this variable to interest rate |
| \_techcombank.flexibleTermDeposit.flexibleTermDepositMaturityMethod | Set this variable to Maturity method |
| \_techcombank.flexibleTermDeposit.flexibleTermDeposit | Set this to 1 |
| \_techcombank.flexibleTermDeposit.flexibleTermDepositperiod | Set this to term deposit period |
| \_techcombank.flexibleTermDeposit.flexibleTermDepositWithdrawAmount | Set this to withdraw amount |
| \_techcombank.flexibleTermDeposit.flexibleTermDepositWithdraw | Set this to 1 |
| \_techcombank.flexibleTermDeposit.flexibleTermDepositTerminationTerm | Set this to termination term |
| \_techcombank.flexibleTermDeposit.flexibleTermDepositTerminationAmt | Set this to termination amount |
| \_techcombank.productViewsOnline.productName | Set this to product name |
| \_techcombank.transactionWebApp.savingPlanChange | Set this to 1 |

**Android XDM Data Format**

//Savings

customXdmData.put("flexibleTermDeposit", new HashMap < String, Object > () {

  put("flexibleTermDepositTerm", "[FLEXIBLE DEPOSIT TERM]");

  put("flexibleTermDepositperiod", "[FLEXIBLE TERM DEPOSIT PERIOD]");

  put("flexibleTermDepositMaturityMethod", "[MATURITY METHOD]");

  put("flexibleTermDepositInterestRate", "[INTEREST RATE]");

  put("flexibleTermDeposit", 1);

});

// Withdraws

customXdmData.put("flexibleTermDeposit", new HashMap < String, Object > () {

  put("flexibleTermDepositWithdrawAmount", "[WITHDRAW AMOUNT]");

  put("flexibleTermDepositWithdraw", 1);

});

// Termination

customXdmData.put("flexibleTermDeposit", new HashMap < String, Object > () {

  put("flexibleTermDepositTerminationTerm", "[TERMINATION TERM]");

  put("flexibleTermDepositTerminationAmt", "[TERMINATION AMOUNT]");

});

// Change of Saving plan

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("savingPlanChange", 1);

});

**iOS - XDM DATA Format**

//Savings

xdmData["\_techcombank"] = ["flexibleTermDeposit": [

    "flexibleTermDepositTerm": "[FLEXIBLE DEPOSIT TERM]",

    "flexibleTermDepositperiod": "[FLEXIBLE TERM DEPOSIT PERIOD]",

    "flexibleTermDepositMaturityMethod": "[MATURITY METHOD]",

    "flexibleTermDepositInterestRate": "[INTEREST RATE]",

    "flexibleTermDeposit": 1]

]

// Withdraws

xdmData["\_techcombank"] = ["flexibleTermDeposit": ["flexibleTermDepositWithdrawAmount": "[WITHDRAW AMOUNT]","flexibleTermDepositWithdraw": 1]

]

// Termination

xdmData["\_techcombank"] = ["flexibleTermDeposit": ["flexibleTermDepositTerminationTerm": "[TERMINATION TERM]", "flexibleTermDepositTerminationAmt": "[TERMINATION AMOUNT]"]

]

// Change of Saving plan

xdmData["\_techcombank"] = ["transactionWebApp": ["savingPlanChange": 1]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar75 | Flexible Term Deposit term | a.x\_techcombank.flexibleTermDeposit.flexibleTermDepositTerm |
| eVar76 | Flexible Term Deposit interest rate | a.x\_techcombank.flexibleTermDeposit.flexibleTermDepositInterestRate |
| eVar77 | Flexible Term Deposit maturity method | a.x\_techcombank.flexibleTermDeposit.flexibleTermDepositMaturityMethod |
| Event47 | Flexible Term Deposit period | a.x\_techcombank.flexibleTermDeposit.flexibleTermDeposit |
| eVar78 | Flexible Term Deposit period | a.x\_techcombank.flexibleTermDeposit.flexibleTermDepositperiod |
| Event59 | Flexible Term Deposit Withdraw Amount | a.x\_techcombank.flexibleTermDeposit.flexibleTermDepositWithdrawAmount |
| Event60 | Flexible Term Deposit Withdraw | a.x\_techcombank.flexibleTermDeposit.flexibleTermDepositWithdraw |
| eVar79 | Flexible Term Deposit termination term | a.x\_techcombank.flexibleTermDeposit.flexibleTermDepositTerminationTerm |
| Event61 | Flexible Term Deposit Termination Amt | a.x\_techcombank.flexibleTermDeposit.flexibleTermDepositTerminationAmt |
| eVar52 | Product Name | a.x\_techcombank.productViewsOnline.productName |
| Event35 | Saving Plan Change | a.x\_techcombank.transactionWebApp.savingPlanChange |

## Goal Saving

This solution allows the business to capture -

* 1. amount, term and interest rate at the journey completion
  2. when withdraw happens after reaching the goal and amount
  3. what point of time (=how much savings to go) user is terminating and termination amount

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM variables should be additionally added along with the global XDM data variables for all Product Journey flows [Sec 4.19].

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| eVar59 | Loan Term | \_techcombank.Loan.loanTerm |
| eVar60 | Loan Interest Rate | \_techcombank.Loan.loanInterestRate |
| Event46 | Loan amount | \_techcombank.Loan.loanAmount |
| Event36 | Goal Saving Withdraw Event | \_techcombank.transactionWebApp.goalSavingWithdrawEvent |
| Event37 | Goal Saving Withdraw Amount | \_techcombank.transactionWebApp.goalSavingWithdrawAmount |
| eVar62 | Goals savings Termination period | \_techcombank.transactionWebApp.goalsSavingsTerminationPeriod |
| Event62 | Goal Savings Termination Amount | \_techcombank.transactionWebApp.goalSavingsTerminationAmount |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.Loan.loanTerm | Set this to savings term |
| \_techcombank.Loan.loanInterestRate | Set this to Rate |
| \_techcombank.Loan.loanAmount | Set this to Savings amount |
| \_techcombank.transactionWebApp.goalSavingWithdrawEvent | Set this to 1 |
| \_techcombank.transactionWebApp.goalSavingWithdrawAmount | Set this to Savings withdraw amount |
| \_techcombank.transactionWebApp.goalsSavingsTerminationPeriod | Set this to Savings Termination period |
| \_techcombank.transactionWebApp.goalSavingsTerminationAmount | Set this to Savings Termination amount |

**Android XDM Data Format**

//Savings

customXdmData.put("loan", new HashMap < String, Object > () {

  put("loanTerm", "[SAVING TERM]");

  put("loanInterestRate", "[SAVING INTEREST RATE]");

  put("loanAmount", "[SAVING AMOUNT]");

});

//Goal Saving Withdraw

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("goalSavingWithdrawEvent", 1);

  put("goalSavingWithdrawAmount", "[WITHDRAW AMOUNT]");

});

//Goal Saving Termination

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("goalsSavingsTerminationPeriod", "[TERMINATION PERIOD]");

  put("goalSavingsTerminationAmount", "[TERMINATION AMOUNT]");

});

**iOS - XDM DATA Format**

//Savings

xdmData["\_techcombank"] = ["Loan": ["loanAmount": "[SAVINGS AMOUNT]",  "loanTerm": "[SAVINGS TERM]", "loanInterestRate": "[SAVINGS INTEREST RATE]"]

]

//Goal Saving Withdraw

xdmData["\_techcombank"] = ["transactionWebApp": [ "goalSavingWithdrawAmount": "[WITHDRAW AMOUNT]", "goalSavingWithdrawEvent": 1]

]

//Goal Saving Termination

xdmData["\_techcombank"] = ["transactionWebApp": [ "goalsSavingsTerminationPeriod": "[TERMINATION PERIOD]",  "goalSavingsTerminationAmount": "[TERMINATION AMOUNT]"]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar59 | Loan Term | a.x\_techcombank.Loan.loanTerm |
| eVar60 | Loan Interest Rate | a.x\_techcombank.Loan.loanInterestRate |
| Event46 | Loan amount | a.x\_techcombank.Loan.loanAmount |
| Event36 | Goal Saving Withdraw Event | a.x\_techcombank.transactionWebApp.goalSavingWithdrawEvent |
| Event37 | Goal Saving Withdraw Amount | a.x\_techcombank.transactionWebApp.goalSavingWithdrawAmount |
| eVar62 | Goals savings Termination period | a.x\_techcombank.transactionWebApp.goalsSavingsTerminationPeriod |
| Event62 | Goal Savings Termination Amount | a.x\_techcombank.transactionWebApp.goalSavingsTerminationAmount |

## Manage Profile – Change Mobile Number

This solution will help to know when user is changing the mobile number under Manage profile section.

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar20 | Link name | \_techcombank.clickTracking.linkName |
| s.eVar40 | Journey Name | \_techcombank.journeyTracking.journeyName |
| s.eVar52 | Product Name | \_techcombank.productViewsOnline.productName |
| Event63 | Mobile number changed | \_techcombank.customerOnlineTracking.mobileNumberChanged |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM variables :

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.clickTracking.linkName | Set this to the name of button/link/Call-to-Action interacted with |
| \_techcombank.pageTracking.customPagename | Set this to the name of the screen where the button/link/Call-to-Action interacted with is present |
| \_techcombank.journeyTracking.journeyName | Set this to journey name |
| \_techcombank.productViewsOnline.productName | Set this to Product name |
| \_techcombank.customerOnlineTracking.mobileNumberChanged | Set this to 1 |

**Android XDM Data Format**

Map < String, Object > customXdmData = new HashMap < > ();

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("linkName", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

});

customXdmData.put("productsViewOnline", new HashMap < String, Object > () {

  put("productName", "[PRODUCT NAME]");

});

customXdmData.put("journeyTracking", new HashMap < String, Object > () {

  put("journeyName", "[JOURNEY NAME]");

  put("journeyStepName", "[JOURNEY STEP NAME]");

});

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("businessUnit", "[BUSINESS UNIT]");

  put("platform", "[PLATFORM]");

  put("environment", "[ENVIRONMENT]");

  put("language", "[LANGUAGE]");

put("deviceGeneralName", "[DEVICE GENERAL NAME]");

  put("mobileNumberChanged", 1);

});

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("customPagename", "[SCREEN NAME]");

  put("eventTimestamp", "[TIMESTAMP]");

  // pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

  {

    put("webInteraction", new HashMap < String, Object > () {

      {

        put("name", "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]");

        put("type", "other");

        put("linkClicks", new HashMap < String, Object > () {

          {

            put("value", 1);

          }

        });

      }

    });

  }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

  .setXdmSchema(xdmData)

  .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [:]

xdmData["\_techcombank"] = ["clickTracking": ["linkName": "[NAME OF BUTTON/LINK/CTA INTERACTED WITH]"]

xdmData["\_techcombank"] = ["productsViewOnline": ["productName": "[PRODUCT NAME]"]

]

xdmData["\_techcombank"] = ["journeyTracking": ["journeyName": "[JOURNEY NAME]", "journeyStepName": "[JOURNEY STEP NAME]"]

]

xdmData["\_techcombank"] = "customerOnlineTracking": [

    "businessUnit": "[BUSINESS UNIT]",

    "platform": "[PLATFORM]",

    "environment": "[ENVIRONMENT]",

"deviceGeneralName": "[DEVICE GENERAL NAME]",

    "language": "[LANGUAGE]",

    "mobileNumberChanged": 1

      ]

xdmData["\_techcombank"] = ["pageTracking": [

   "customPagename": "[SCREEN NAME]",

   "eventTimestamp": "[TIMESTAMP]"

// pass timestamp here in UTC format – YYYY-MM-DD and then the timestamp- HH:MM:SS , example - "2023-05-02T00:00:00Z"

]

]

xdmData["web"] = [

    "webInteraction": [

    "name": "[LINK NAME]",

    "type":"other",

    "linkClicks": ["value": 1] ]

    ]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar20 | Link Name | a.x.\_ techcombank..clickTracking.linkName |
| eVar11 | Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| s.eVar40 | Journey Name | a.x\_techcombank.journeyTracking.journeyName |
| s.eVar52 | Product Name | a.x\_techcombank.productViewsOnline.productName |
| Event63 | Mobile number changed | a.x\_techcombank.customerOnlineTracking.mobileNumberChanged |

## Notification

This solution will help to track notification type that user clicked to open the app (ex. Account balance change, Credit card transaction notification, Open/close TD notification, Loan reminder).

### Variables in this Section

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar11 | Page Name/Screen Name | \_techcombank.pageTracking.customPagename |
| s.eVar38 | Notification Type | \_techcombank.AppNotification.appNotificationType |
| Event71 | Notification clicked | \_techcombank.clickTracking.appNotificationClick |

### Deployment Instructions [To do: TCB IT Team]

The below mentioned XDM variables should be additionally added along with the global XDM data variables for all Global screens [Sec 4.3].

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.pageTracking.customPagename | Set this variable to Screen name |
| \_techcombank.AppNotification.appNotificationType | Se this variable to App notification type |
| \_techcombank.clickTracking.appNotificationClick | Set this variable to 1 |

**Android XDM Data Format**

//Add the below Custom XDM data along with the existing global screen XDM data

customXdmData.put("AppNotification", new HashMap < String, Object > () {

  put("appNotificationType", "[APP NOTIFICATION TYPE]");

});

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("appNotificationClick", 1);

});

**iOS - XDM DATA Format**

//Add the below Custom XDM data along with the existing global screen XDM data sec 4.13

xdmData["\_techcombank"] = ["AppNotification":["appNotificationType": "[APP NOTIFICATION TYPE]"]

]

xdmData["\_techcombank"] = ["clickTracking":["appNotificationClick": 1]

]

### Create Processing Rules [To do: Adobe Team]

Create the following processing rules to configure tracking:

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| s.eVar11 | Page Name/Screen Name | a.x.\_techcombank.pageTracking.customPagename |
| s.eVar38 | Notification Type | a.x\_techcombank.AppNotification.appNotificationType |
| Event71 | Notification clicked | a.x\_techcombank.clickTracking.appNotificationClick |

## Unsecured Lending

This solution allows the business to capture the loan amount, term and interest rate at the end of the journey.

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM variables should be additionally added along with the global XDM data variables for all Product Journey flows [Sec 4.19].

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| Event46 | Loan Amount | \_techcombank.Loan.loanAmount |
| eVar59 | Term | \_techcombank.Loan.loanTerm |
| eVar60 | Interest rate | \_techcombank.Loan.loanInterestRate |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.Loan.loanAmount | Set this variable to loan amount. |
| \_techcombank.Loan.loanTerm | Set this variable to loan term. |
| \_techcombank.Loan.loanInterestRate | Set this variable to loan interest rate. |

**Android XDM Data Format**

// Add this custom XDM data along with the existing XDM data in sec 4.19

customXdmData.put("Loan", new HashMap < String, Object > () {

  put("loanAmount", "[LOAN AMOUNT]");

  put("loanTerm", "[LOAN TERM]");

  put("loanInterestRate", "[LOAN INTEREST RATE]");

});

**iOS - XDM DATA Format**

// Add this custom XDM data along with the existing XDM data in sec 4.19

xdmData["\_techcombank"] = [ "loan": [

    "loanAmount": "[LOAN AMOUNT]",

    "loanTerm": "[LOAN TERM]",

    "loanInterestRate": "[LOAN INTEREST RATE]"

    ]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| Event46 | Loan amount | \_techcombank.Loan.loanAmount |
| eVar59 | Loan Term | \_techcombank.Loan.loanTerm |
| eVar60 | Loan interest rate | \_techcombank.Loan.loanInterestRate |

## Transfer

This solution allows the business to track number of transactions for (between my account, to other account, bill & top-up) and amount.

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM variables should be additionally added along with the global XDM data variables for all Product Journey flows [Sec 4.19].

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| eVar80 | Transfer Type | \_techcombank.transactionWebApp.transferType |
| eVar81 | Transfer Transaction Id | \_techcombank.transactionWebApp.transferTransactionID |
| Event48 | Transfer Amount | \_techcombank.transactionWebApp.transferAmount |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.transactionWebApp.transferType | Set this variable to transfer type |
| \_techcombank.transactionWebApp.transferTransactionID | Set this variable to transaction ID |
| \_techcombank.transactionWebApp.transferAmount | Set this variable to Transfer Amount |

**Android XDM Data Format**

// Add this custom XDM data along with the existing XDM data in sec 4.19

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("transferTransactionID", "[TRANSACTION ID]");

  put("transferType", "[TRANSFER TYPE]");

  put("transferAmount", "[TRANSFER AMOUNT]");

});

**iOS - XDM DATA Format**

// Add this custom XDM data along with the existing XDM data in sec 4.19

xdmData["\_techcombank"] = ["transactionWebApp": [

    "transferTransactionID": "[TRANSACTION ID]",

    "transferType": "[TRANSFER TYPE]",

    "transferAmount": "[TRANSFER AMOUNT]"]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar80 | Transfer Type | a.x.\_techcombank.transactionWebApp.transferType |
| eVar81 | Transfer Transaction Id | a.x.\_techcombank.transactionWebApp.transferTransactionID |
| Event48 | Transfer Amount | a.x.\_techcombank.transactionWebApp.transferAmount |

## Overdraft

This solution allows the business to track selection of additional information during application journey, track credit limit change event under Loan preference, and track downloads of the contract.

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM variables should be additionally added along with the global XDM data variables for all Product Journey flows [Sec 4.19].

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| List2 | Online Application Selections | \_techcombank.clickTracking.onlineApplicationSelections |
| Event32 | Credit Limit Change | \_techcombank.creditCard.creditLimitChange |
| Event33 | Contract Downloads | \_techcombank.fileDownload.contractDownloads |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.clickTracking.onlineApplicationSelections | Set to capture all additional information in application journey. |
| \_techcombank.creditCard.creditLimitChange | Set this to 1 |
| \_techcombank.fileDownload.contractDownloads | Set this to 1 |

**Android XDM Data Format**

// Track selection of additional information during application journey

customXdmData.put("clickTracking", new HashMap < String, Object > () {

  put("onlineApplicationSelections", "[ADDITIONAL INFO DURING APPLICATION JOURNEY]");

});

//Track credit limit change event under Loan preference

customXdmData.put("creditCard", new HashMap < String, Object > () {

  put("creditLimitChange", 1); //if credit limit changed successfully

});

//Track downloads of the contract

customXdmData.put("fileDownload", new HashMap < String, Object > () {

  put("contractDownloads", 1); //if contracts downloaded

});

**iOS - XDM DATA Format**

// Track selection of additional information during application journey

xdmData["\_techcombank"] = ["clickTracking":["onlineApplicationSelections": "[ADDITIONAL INFO DURING APPLICATION JOURNEY]"]

]

//Track credit limit change event under Loan preference

xdmData["\_techcombank"] = ["creditCard":["creditLimitChange": 1]

]

//Track downloads of the contract

xdmData["\_techcombank"] = ["fileDownload": ["contractDownloads": 1]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| List2 | Online Application Selections | a.x.\_techcombank.clickTracking.onlineApplicationSelections |
| Event32 | Credit Limit Change | a.x.\_techcombank.creditCard.creditLimitChange |
| Event33 | Contract Downloads | a.x.\_techcombank.fileDownload.contractDownloads |

## Certificate of Deposit

This solution allows business to track when user withdraws together with amount and term that user has been holding the product.

### Variables in this Section

The below table consists of variables used for this solution. The below mentioned XDM variables should be additionally added along with the global XDM data variables for all Product Journey flows [Sec 4.19].

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| eVar63 | COD Term | \_techcombank.transactionWebApp.CODTerm |
| Event49 | COD Withdraw Amount | \_techcombank.transactionWebApp.CODwithdrawAmount |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.transactionWebApp.CODTerm | Set to COD Term. |
| \_techcombank.transactionWebApp.CODwithdrawAmount | Set this to COD Amount. |

**Android XDM Data Format**

//Track when user withdraws together with amount and term that user has been holding the product

//send this custom XDM data along with the other custom XDM data in the confirmation page sec 4.19

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("CODTerm", "[COD TERM]");

  put("CODwithdrawAmount", "[COD WITHDRAW AMOUNT]");

});

**iOS - XDM DATA Format**

//Track when user withdraws together with amount and term that user has been holding the product

//send this custom XDM data along with the other custom XDM data in the confirmation page sec 4.19

xdmData["\_techcombank"] = ["transactionWebApp": [

    "CODTerm": "[COD TERM]",

    "CODwithdrawAmount": "[COD WITHDRAW AMOUNT]"

    ]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar63 | COD Term | a.x.\_techcombank.transactionWebApp.CODTerm |
| Event49 | COD Withdraw Amount | a.x.\_techcombank.transactionWebApp.CODwithdrawAmount |

## Rewards

This solution allows business to track sub type of the Rewards (ex. campaigns, Last chance, Special deal for you, Vouchers, Deals, Gifts), Capture clicks and the name of the category on offer listing page, Capture voucher details (name of the merchant, point) upon saving to My Voucher.

The below table consists of variables used for this solution.

### Variables in this section:

The below mentioned XDM variables should be additionally added along with the global XDM data variables for all pages [Sec 4.3].

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| s.eVar68 | Reward subtype | \_techcombank.rewardTracking.rewardsubtype |
| s.eVar67 | Reward category | \_techcombank.rewardTracking.rewardcategory |
| s.eVar54 | Merchant Details | \_techcombank.transactionWebApp.merchantDetails |
| s.eVar69 | Voucher points | \_techcombank.voucherTracking.voucherPoints |
| Event70 | Rewards Clicked | \_techcombank.rewardTracking.rewardsClicked |
| Event69 | Voucher Saved | \_techcombank.voucherTracking.voucherSaved |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.rewardTracking.rewardsubtype | Set to reward sub type |
| \_techcombank.rewardTracking.rewardcategory | Set this to reward category |
| \_techcombank.transactionWebApp.merchantDetails | Set this to Merchanr details |
| \_techcombank.voucherTracking.voucherPoints | Set this to voucher points |
| \_techcombank.rewardTracking.rewardsClicked | Set this to rewards clicked |
| \_techcombank.voucherTracking.voucherSaved | Set this to voucher saved |

**Android XDM Data Format**

//Capture sub type of the Rewards (ex. campaigns, Last chance, Special deal for you, Vouchers, Deals, Gifts)

//On load of rewards page

customXdmData.put("rewardTracking", new HashMap < String, Object > () {

  put("rewardsubtype", "[REWARD SUBTYPE]"); //ex. campaigns, Last chance, Special deal for you, Vouchers, Deals, Gifts

});

//Capture clicks and the name of the category on offer listing page

customXdmData.put("rewardTracking", new HashMap < String, Object > () {

  put("rewardcategory", "[REWARD CATEGORY]"); //

  put("rewardsClicked", 1);

});

//Capture voucher details (name of the merchant, point) upon saving to My Voucher

// Add this custom XDM data along with the other page level variables on confirmation page

customXdmData.put("voucherTracking", new HashMap < String, Object > () {

  put("voucherPoints", "[VOUCHER POINTS]"); //

  put("voucherSaved", 1);

});

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("merchantDetails", "[MERCHANT DETAILS]"); //

});

**iOS - XDM DATA Format**

//Capture sub type of the Rewards (ex. campaigns, Last chance, Special deal for you, Vouchers, Deals, Gifts)

//On load of rewards page

xdmData["\_techcombank"] = ["rewardTracking":["rewardsubtype": "[REWARD SUBTYPE]"]

]

//ex. campaigns, Last chance, Special deal for you, Vouchers, Deals, Gifts

//Capture clicks and the name of the category on offer listing page

xdmData["\_techcombank"] = ["rewardTracking":["rewardcategory": "[REWARD CATEGORY]",  "rewardsClicked": 1]

]

//Capture voucher details (name of the merchant, point) upon saving to My Voucher

// Add this custom XDM data along with the other page level variables on confirmation page

xdmData["\_techcombank"] = ["voucherTracking":["voucherPoints": "[VOUCHER POINTS]", "voucherSaved": 1]

]

xdmData["\_techcombank"] = ["transactionWebApp":["merchantDetails": "[MERCHANT DETAILS]"]

]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| s.eVar68 | Reward subtype | a.x.\_techcombank.rewardTracking.rewardsubtype |
| s.eVar67 | Reward category | a.x.\_techcombank.rewardTracking.rewardcategory |
| s.eVar54 | Merchant Details | a.x.\_techcombank.transactionWebApp.merchantDetails |
| s.eVar69 | Voucher points | a.x.\_techcombank.voucherTracking.voucherPoints |
| Event70 | Rewards Clicked | a.x.\_techcombank.rewardTracking.rewardsClicked |
| Event69 | Voucher Saved | a.x.\_techcombank.voucherTracking.voucherSaved |

## Term Deposit

This solution allows business to capture –

1. amount, term and interest rate and maturity method at the journey completion
2. when withdraw happens after fulfilling deposit period and amount
3. what point of time (=how many months to go) user is terminating and termination amount
4. if user makes changes in Saving Plan (ex. Auto Roll etc.)

The below table consists of variables used for this solution.

### Variables in this section:

The below mentioned XDM variables should be additionally added along with the global XDM data variables for all pages [Sec 4.3] as applicable.

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| eVar82 | Term Deposit term | \_techcombank.termDeposit.termDepositTerm |
| eVar83 | Term Deposit interest rate | \_techcombank.termDeposit.termDepositInterestRate |
| eVar84 | Term Deposit maturity method | \_techcombank.termDeposit.termDepositMaturityMethod |
| eVar85 | Term Deposit withdraw period | \_techcombank.termDeposit.termDepositWithdrawPeriod |
| eVar86 | Term Deposit termination term | \_techcombank.termDeposit.termDepositTerminationTerm |
| eVar52 | Product Name | \_techcombank.productViewsOnline.productName |
| Event64 | Terms deposit Savings Amount | \_techcombank.termDeposit.termsdepositSavingsAmount |
| Event65 | Term Deposit Withdraw Amount | \_techcombank.termDeposit.termDepositWithdrawAmount |
| Event66 | Term Deposit Withdraw | \_techcombank.termDeposit.termDepositWithdraw |
| Event67 | Term Deposit Termination Amount | \_techcombank.termDeposit.termDepositTerminationAmount |
| Event68 | Term Deposit Termination | \_techcombank.termDeposit.termDepositTermination |
| Event35 | Saving Plan Change | \_techcombank.transactionWebApp.savingPlanChange |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.termDeposit.termDepositTerm | Set this to Deposit term |
| \_techcombank.termDeposit.termDepositInterestRate | Set this to Deposit interest rate |
| \_techcombank.termDeposit.termDepositMaturityMethod | Set this to Deposit Maturity method |
| \_techcombank.termDeposit.termDepositWithdrawPeriod | Set this to Deposit withdraw period |
| \_techcombank.termDeposit.termDepositTerminationTerm | Set this to Deposit Termination Term |
| \_techcombank.productViewsOnline.productName | Set this to product name |
| \_techcombank.termDeposit.termsdepositSavingsAmount | Set this to Deposit savings amount |
| \_techcombank.termDeposit.termDepositWithdrawAmount | Set this to Deposit withdraw amount |
| \_techcombank.termDeposit.termDepositWithdraw | Set this to 1 when deposit completes |
| \_techcombank.termDeposit.termDepositTerminationAmount | Set this to Deposit termination amount |
| \_techcombank.termDeposit.termDepositTermination | Set this to 1 when deposit termination completes |
| \_techcombank.transactionWebApp.savingPlanChange | Set this to 1 when saving plan change completes |

**Android XDM Data Format**

//Capture amount, term and interest rate and maturity method at the journey completion

// add this custom data variables along with existing custom xdm data for Journey as in sec 4.19

customXdmData.put("termDeposit", new HashMap < String, Object > () {

  put("termDepositTerm", "[TERM DEPOSIT TERM]");

  put("termDepositInterestRate", "[TERM DEPOSIT INTEREST RATE]");

  put("termDepositMaturityMethod", "[TERM DEPOSIT MATURITY METHOD]");

  put("termsdepositSavingsAmount", "[TERM DEPOSIT SAVINGS AMOUNT]");

});

//Track when withdraw happens after fulfilling deposit period and amount

customXdmData.put("termDeposit", new HashMap < String, Object > () {

  put("termDepositWithdrawPeriod", "[TERM DEPOSIT WITHDRAW PERIOD]");

  put("termDepositWithdrawAmount", "[TERM DEPOSIT TERMINATION AMOUNT]");

  put("termDepositWithdraw", 1);

});

//Track what point of time (=how many months to go) user is terminating and termination amount

customXdmData.put("termDeposit", new HashMap < String, Object > () {

  put("termDepositTerminationTerm", "[TERM DEPOSIT TERMINATION TERM]");

  put("termDepositTerminationAmount", "[TERM DEPOSIT TERMINATION AMOUNT]");

  put("termDepositWithdraw", 1);

});

//Capture if user makes changes in Saving Plan (ex. Auto Roll etc.)

customXdmData.put("productViewsOnline", new HashMap < String, Object > () {

  put("productName", "[PRODUCT NAME]");

  });

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("savingPlanChange", 1);

  });

**iOS - XDM DATA Format**

//Capture amount, term and interest rate and maturity method at the journey completion

// add this custom data variables along with existing custom xdm data for Journey as in sec 4.19

xdmData["\_techcombank"] = ["termDeposit": [

    "termDepositTerm": "[TERM DEPOSIT TERM]",

    "termDepositInterestRate": "[TERM DEPOSIT INTEREST RATE]",

    "termDepositMaturityMethod": "[TERM DEPOSIT MATURITY METHOD]",

    "termDepositSavingsAmount": "[TERM DEPOSIT SAVINGS AMOUNT]"

    ]

]

//Track when withdraw happens after fulfilling deposit period and amount

xdmData["\_techcombank"] = ["termDeposit": [

    "termDepositWithdrawPeriod": "[TERM DEPOSIT WITHDRAW PERIOD]",

    "termDepositWithdrawAmount": "[TERM DEPOSIT TERMINATION AMOUNT]",

    "termDepositWithdraw": 1

    ]

]

//Track what point of time (=how many months to go) user is terminating and termination amount

xdmData["\_techcombank"] = ["termDeposit":[

    "termDepositTerminationTerm": "[TERM DEPOSIT TERMINATION TERM]",

    "termDepositTerminationAmount": "[TERM DEPOSIT TERMINATION AMOUNT]",

    "termDepositWithdraw": 1

    ]

]

//Capture if user makes changes in Saving Plan (ex. Auto Roll etc.)

xdmData["\_techcombank"] = "productViewsOnline": ["productName": "[PRODUCT NAME]"]

xdmData["\_techcombank"] = "transactionWebApp": ["savingPlanChange": 1]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| s.eVar68 | Reward subtype | a.x.\_techcombank.rewardTracking.rewardsubtype |
| s.eVar67 | Reward category | a.x.\_techcombank.rewardTracking.rewardcategory |
| s.eVar54 | Merchant Details | a.x.\_techcombank.transactionWebApp.merchantDetails |
| s.eVar69 | Voucher points | a.x.\_techcombank.voucherTracking.voucherPoints |
| Event70 | Rewards Clicked | a.x.\_techcombank.rewardTracking.rewardsClicked |
| Event69 | Voucher Saved | a.x.\_techcombank.voucherTracking.voucherSaved |

## Credit card

This solution allows business to capture –

1. merchant's name and amount that user selects for instalment
2. instalment period that user selects
3. total amount that user is applying for instalment
4. change of the credit limit and amount
5. customer segment (pre-qualified vs. organic)

The below table consists of variables used for this solution.

### Variables in this section:

The below mentioned XDM variables should be additionally added along with the global XDM data variables for all pages [Sec 4.3] and 4.19 as applicable.

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| eVar54 | Merchant Details | \_techcombank.transactionWebApp.merchantDetails |
| eVar87 | Credit Card Instalment Period | \_techcombank.creditCard.creditCardInstalmentPeriod |
| eVar73 | Credit Card Limit Edit Amount | \_techcombank.creditCard.creditCardLimitEditAmount |
| eVar74 | Customer segment | \_techcombank.customerOnlineTracking.customersegment |
| Event45 | Instalment Amount | \_techcombank.transactionWebApp.installmentAmount |
| Event58 | Credit Card Limit Edit | \_techcombank.creditCard.creditCardLimitEdit |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.transactionWebApp.merchantDetails | Set to Merchant details |
| \_techcombank.creditCard.creditCardInstalmentPeriod | Set to credit card instalment period |
| \_techcombank.creditCard.creditCardLimitEditAmount | Set to credit card limit edit amount |
| \_techcombank.customerOnlineTracking.customersegment | Set to customer segment |
| \_techcombank.transactionWebApp.installmentAmount | Set to instalment amount |
| \_techcombank.creditCard.creditCardLimitEdit | Set to 1 when credit card limit edit success. |

**Android XDM Data Format**

//Track merchant's name and amount that user selects for instalment

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("merchantDetails", "[MERCHANT DETAILS]");

  put("installmentAmount", "[INSTALMENT AMOUNT]");

  });

//Track instalment period that user selects

customXdmData.put("creditCard", new HashMap < String, Object > () {

  put("creditCardInstalmentPeriod", "[INSTALMENT PERIOD]");

  });

//Track total amount that user is applying for instalment

customXdmData.put("transactionWebApp", new HashMap < String, Object > () {

  put("installmentAmount", "[INSTALMENT AMOUNT]");

  });

//Track change of the credit limit and amount

//add this custom XDM data on amount change along with other variables (TBD - onload/click)

customXdmData.put("creditCard", new HashMap < String, Object > () {

  put("creditCardLimitEditAmount", "[CC CARD LIMIT EDIT AMOUNT]");

  put("creditCardLimitEdit", 1);

  });

//Track customer segment (pre-qualified vs. organic)

// add this custom XDM data when the credit card journey starts along with sec 4.19

customXdmData.put("customerOnlineTracking", new HashMap < String, Object > () {

  put("customersegment", "[CUSTOMER SEGMENT]"); //eg. pre-qualified or organic

  });

**iOS - XDM DATA Format**

//Track merchant's name and amount that user selects for instalment

xdmData["\_techcombank"] = ["transactionWebApp": [ "merchantDetails": "[MERCHANT DETAILS]", "installmentAmount": "[INSTALMENT AMOUNT]"]]

//Track instalment period that user selects

xdmData["\_techcombank"] = ["\_creditCard": ["creditCardInstalmentPeriod": "[INSTALMENT PERIOD]"]]

//Track total amount that user is applying for instalment

xdmData["\_techcombank"] = ["transactionWebApp": ["installmentAmount": "[INSTALMENT AMOUNT]"]]

//Track change of the credit limit and amount

//add this custom XDM data on amount change along with other variables (TBD - onload/click)

xdmData["\_techcombank"] = ["\_creditCard": [ "creditCardLimitEditAmount": "[CC CARD LIMIT EDIT AMOUNT]", "creditCardLimitEdit": 1]

  ]

//Track customer segment (pre-qualified vs. organic)

// add this custom XDM data when the credit card journey starts along with sec 4.19

xdmData["\_techcombank"] = ["\_customerOnlineTracking": ["customersegment": "[CUSTOMER SEGMENT]"]]

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar54 | Merchant Details | a.x.\_techcombank.transactionWebApp.merchantDetails |
| eVar87 | Credit Card Instalment Period | a.x.\_techcombank.creditCard.creditCardInstalmentPeriod |
| eVar73 | Credit Card Limit Edit Amount | a.x.\_techcombank.creditCard.creditCardLimitEditAmount |
| eVar74 | Customer segment | a.x.\_techcombank.customerOnlineTracking.customersegment |
| Event45 | Instalment Amount | a.x.\_techcombank.transactionWebApp.installmentAmount |
| Event58 | Credit Card Limit Edit | a.x.\_techcombank.creditCard.creditCardLimitEdit |

## Time between two Action.

This solution allows business to capture time between the start and the end of an action.

**Implementation Instructions -** A variable should be added through the app code to capture the timestamp at that point where Action Start is required and has to be stored until Action End.

Trigger Edge.sendEvent() call to Adobe. Pass the time duration in seconds between 2 events (Action End & Action Start).

The below table consists of variables used for this solution.

### Variables in this section:

The below table consists of variables used for this solution:

|  |  |  |
| --- | --- | --- |
| Analytics Variable | Used For | XDM Data |
| eVar91 | Timed Action Name | \_techcombank.pageTracking.timedActionName |
| Event72 | Timed Action | \_techcombank.pageTracking.timedAction |

### Deployment Instructions [To do: TCB IT Team]

Set the following XDM data variables along with the global XDM data variables:

|  |  |
| --- | --- |
| XDM Data | Notes |
| \_techcombank.pageTracking.timedActionName | Set this variable to Timed Action name |
| \_techcombank.pageTracking.timedAction | Set this variable to total seconds consumed for the action |

**Android XDM Data Format**

HashMap <String, Object> customXdmData = new HashMap <String, Object> ();

customXdmData.put("pageTracking", new HashMap < String, Object > () {

  put("timedActionName", "[TIMED ACTION NAME]");

  put("timedAction", "[TIME DURATION BETWEEN EVENTS]"); //pass value in seconds

});

Map < String, Object > xdmData = new HashMap < String, Object > ();

xdmData.put("\_techcombank", customXdmData);

xdmData.put("web", new HashMap < String, Object > () {

    {

        put("webInteraction", new HashMap < String, Object > () {

            {

                put("name", "[As defined by Timed Action End]");

                put("type", "other");

                put("linkClicks", new HashMap < String, Object > () {

                    {

                        put("value", "1");

                    }

               });

            }

        });

    }

});

xdmData.put("eventType", "web.webInteraction.linkClicks");

ExperienceEvent experienceEvent = new ExperienceEvent.Builder()

    .setXdmSchema(xdmData)

    .build();

Edge.sendEvent(experienceEvent, null);

**iOS - XDM DATA Format**

var xdmData: [String: Any] = [: ]

xdmData["\_techcombank"] = ["pageTracking": [

"timedAction": "[TIME DURATION BETWEEN EVENTS]",

"timedActionName": "[TIMED ACTION NAME]"

]

]

xdmData["web"] = [

    "webInteraction": [

        "linkClicks": [

            "value": "1"

        ],

        "name": "[As defined by Timed Action End]",

        "type": "other"

    ]

]

xdmData["eventType"] = "web.webInteraction.linkClicks"

let experienceEvent = ExperienceEvent(xdm: xdmData)

Edge.sendEvent(experienceEvent: experienceEvent)

### Create Processing Rules [To do: Adobe Team]

Processing rules help to copy the values from XDM data variables and set eVars/events/props. Because of the processing rules, developers no longer need to learn the different variables needed for implementation. Marketers have more flexibility and less dependency on app dev teams to get the data into eVars/props. Once deployment is complete, the processing rules should be setup in order to see the data in the reports.

|  |  |  |
| --- | --- | --- |
| Variable | Name | XDM Data |
| eVar91 | Timed Action Name | a.x.\_techcombank.pageTracking.timedActionName |
| Event72 | Timed Action | a.x.\_techcombank.pageTracking.timedAction |

# App View to Web View Tracking

Visitor Tracking Between an App and a Web View.

This solution allows the business unit to evaluate the usage of App to Web Journey as part of the requirement.

## Implementation Instructions: [To do: TCB IT Team]

**Android:**

Identity.appendVisitorInfoForURL("<WEB URL HERE> ", new AdobeCallback<String>() {

    @Override

    public void call(String urlWithAdobeVisitorInfo) {

        //handle the new URL here

        //For example, open the URL on the device browser

        Intent i = new Intent(Intent.ACTION\_VIEW);

        i.setData(Uri.parse(urlWithAdobeVisitorInfo));

        startActivity(i);

    }

});

This API appends Adobe visitor information to the query component of the specified URL.

If the provided URL is null or empty, it is returned as is. Otherwise, the following information is added to the query component of the specified URL and is returned in the [AdobeCallback](https://experienceleague.adobe.com/docs/platform-learn/implement-mobile-sdk/app-implementation/web-views.html?lang=en) instance:

* The adobe\_mc attribute is a URL encoded list that contains:
  + MCMID - Experience Cloud ID (ECID)
  + MCORGID - Experience Cloud Org ID
  + MCAID - Analytics Tracking ID (AID), if available from the Analytics extension
  + TS - A timestamp taken when this request was made.
* The optional adobe\_aa\_vid attribute is the URL-encoded Analytics Custom Visitor ID (VID), if previously set in the Analytics extension.

The ID service code on the destination domain extracts the ECID from the URL instead of sending a request to Adobe for a new ID. The ID service code on the destination page uses this ECID to track the visitor. On hits from the mobile web content, verify that the *mid* parameter exists on each hit, and that this value matches the *mid* value that is being sent by the app code.

iOS:

Identity.appendTo(url: URL(string: "<WEB URL HERE>")) { (appendedURL, error) in

    if let appendedURL = appendedURL {

      DispatchQueue.main.async {

        uiView.load(URLRequest(url: appendedURL))

      }

    }

   };

**Reference Documents:**

**Analytics verifications:** To ensure that the IDs that are appended to the URL that is being opened, verify that the adobe\_mc query parameter appears in the URL.

**Reference Link:** <https://experienceleague.adobe.com/docs/platform-learn/implement-mobile-sdk/app-implementation/web-views.html?lang=en>

**Sample value of URL parameter adobe\_mc:**

ts%3d1544353810%7cmcmid%3d92018745312349688606805986756705962996%7cmcorgid%3d1e4734fa53dac2360a495566%40adobeorg

**Additional Notes:**

Please include an additional “source” URL parameter with value “mobile\_app” in the link URL after the adobe\_mc query parameter. This is to tag the link with the correct channel attribution source.

**Sample tagged URL with adobe\_mc and source query parameters:**

https://techcombank.com/en/personal/invest**?adobe\_mc=<value generated from app>&source=mobile\_app**

# Steps to Capture ECID. [To do: TCB IT Team]

**For Android :**

Syntax: public static void getExperienceCloudId(final AdobeCallback<String> callback);

// callback is invoked after the ECID is available. The callback may be invoked on a different thread.

Example –

Identity.getExperienceCloudId(new AdobeCallback<String>() {

@Override

public void call(String id) {

// Handle the ID returned here

}

});

**For iOS:**

Syntax: static func getExperienceCloudId(completion: @escaping (String?, Error?) -> Void)

// completion is invoked after the ECID is available. The default timeout is 1000ms.

Example –

Identity.getExperienceCloudId { (ecid, error) in

if let error = error {

// Handle the error here

} else {

// Handle the retrieved ID here

}

}

Link to refer - <https://developer.adobe.com/client-sdks/documentation/identity-for-edge-network/api-reference/>

# Validation [To do: Adobe Team] & [TCB IT Team]

## Project Griffon (Adobe Experience Platform Assurance)

### Implement Assurance on the App

This is a Launch extension that enables capabilities for [Project Griffon](https://aep-sdks.gitbook.io/docs/foundation-extensions/adobe-experience-platform-assurance). To get started, follow the below steps:

* Install the AEP Assurance extension in Experience Platform Launch
* Add AEP Assurance SDK extension library to your App.
* Import AEP Assurance into your App.
* Register and implement extension APIs.

1. **Install the AEP Assurance extension in Experience Platform Launch**

In Experience Platform Launch, click the **Extensions** tab.

On the **Catalog** tab, locate the **AEP Assurance** extension, and click **Install**.

Follow the publishing process to update SDK configuration.

Graphical user interface, text, application, chat or text message

Description automatically generated

1. **Add AEP Assurance SDK extension library to your app**

Verify if the following dependencies are added in your project's build.gradle file:

implementation 'com.adobe.marketing.mobile:core:1+'

1. **Import AEP Assurance into your app**

Verify if the following libraries are imported:

import com.adobe.marketing.mobile.Assurance;

import com.adobe.marketing.mobile.MobileCore;

1. **Register AEPAssurance with Mobile Core**

Registering the extension with Core, sends Experience Platform SDK events to an active Project Griffon session. To start using the extension library, you must first register the extension with the [Mobile Core](https://github.com/adobe/aepsdk-core-android) extension. Verify if the extensions are registered as per below format:

public class MobileApp extends Application {

@Override

public void onCreate() {

super.onCreate();

MobileCore.setApplication(this);

MobileCore.configureWithAppId("<LAUNCH ENVIRONMENT ID>");

try {

Assurance.registerExtension();

MobileCore.start(null);

} catch (Exception e) {

// Log the exception

}

}

}

**The Assurance registration method should be conditionally blocked from being executed in Production build**

### Using Project Griffon for Validation

#### Logging in to Griffon

1. Visit <https://experience.adobe.com/griffon>
2. Log in using your Adobe ID credentials for the Experience Cloud. If you do not know your Adobe ID credentials, contact your Adobe administrator or see [how to log in](https://experienceleague.adobe.com/docs/analytics/analyze/landing.html?lang=en).
3. In the beta access page, click **Register**.

#### Creating sessions

1. Click **Create Session** in the top right.
2. In the **Create New Session** dialog, review instructions, and proceed by selecting **Start**

Graphical user interface, text, application

Description automatically generated

1. Enter a name to identify the session, then provide a **Base URL** (deep linking URL for your app) Click **Next**.
2. In the **Base URL** field, type your app's base deep link definition (As provided by your App dev team)

Graphical user interface, text, application, chat or text message

Description automatically generated

#### Connecting to a session

After you've created a session, you can begin connecting to it by following these steps:

1. Ensure that you see a the Create New Session dialog now shows you a link, a QR code, and a PIN.

Graphical user interface, text, application, chat or text message

Description automatically generated Qr code

Description automatically generated Graphical user interface, application

Description automatically generated

1. Complete one of the following tasks:

* Use your device camera app to scan the QR code and to open your app OR
* Copy the link and open in your app.

1. When your app launches, you should see the PIN entry screen overlaid.

A picture containing shape

Description automatically generated

1. Type in the PIN from the previous step and press Connect.
2. Verify that your app is connected to Project Griffon when Adobe Experience Platform icon is displayed on your app. (In Griffon UI, it will show as a green dot with number of devices connected to the session)

Icon

Description automatically generated Graphical user interface, text, application, chat or text message

Description automatically generated

(In app icon) (Griffon UI connection status)

**Note – You can also connect to an existing session by clicking on the Session Details of the previous session and then using the QR code/Link with the PIN.**

#### Exporting a session

To export a Project Griffon session, on your app’s sessions details page, click **Export to JSON** in a session:

A picture containing screenshot, text

Description automatically generated

### Understanding Analytics calls in the Griffon Session

1. Go to the Events section of the navigation panel and look for **analytics.hit** and **analytics.mapping** event type – Ensure both the events are firing

A screenshot of a computer

Description automatically generated with medium confidence

Note – In case you do not notice these events firing, also try to disconnect and reconnect the app with Project Griffon (to prevent any connection issues).

1. Click on **analytics.mapping** event type to verify the contextData values under **mappedQueryParams**

Graphical user interface, text, application, email

Description automatically generated

#### Verifying Lifecycle Calls

Go to the Events section of the navigation panel and look for **Lifecycle** event type – Ensure lifecycle events are firing based on the app lifecycle actions.

Graphical user interface, application, Teams

Description automatically generated

#### Verifying Page View calls (previously trackState server calls)

Click on **analytics.mapping** event type to verify the contextData values under **mappedQueryParams.** Check for below parameter values:

* eventtype= web.webPageDetails.pageViews
* web.webpagedetails.pageviews.value=1

Graphical user interface, application

Description automatically generated

#### Verifying Link click calls (previously trackAction server calls)

Click on **analytics.mapping** event type to verify the contextData values under **mappedQueryParams.** Check for below parameter values:

* eventtype=web.webInteraction.linkClicks
* web.webinteraction.linkclicks.value=1
* pe=lnk\_o

Graphical user interface, application

Description automatically generated

#### Verifying analytics Products variable (previously &&products)

Old contextData - &&products

New contextData – productlistitems[i].name, ‘**pl’** parameter

Graphical user interface, application

Description automatically generated

#### Verifying event serialization (previously &&events)

In the Griffon event **analytics.mapping** where serialized event is expected, look for **events** parameter along with the expected serialized value for the particular event, sample below:

Graphical user interface, application, table

Description automatically generated

## How to use Charles for Validation [To do: TCB IT Team]

### Android build settings for Charles SSL Proxying

If you are using android 8.0 or above, you need to add configuration to your app in order to have it trust the SSL certificates generated by Charles SSL Proxying. This means that you can only use SSL Proxying with apps that you control. These steps should be performed for development builds or QA buildings only.

Note: we recommend to NOT perform below steps for production version of the app.

**Step-1:** In order to configure your app to trust Charles, you need to add a [Network Security Configuration File](https://developer.android.com/training/articles/security-config.html) to your app.

Include *<?*xml version="1.0" encoding="utf-8"*?>*<network-security-config>  
    <debug-overrides>  
        <trust-anchors>  
            <certificates src="user" />  
        </trust-anchors>  
    </debug-overrides>  
</network-security-config>

**Step-2:** Then add a reference to this file in your app's manifest, as follows:

<?xml version="1.0" encoding="utf-8"?>  
<manifest ... >  
 <application android:networkSecurityConfig="@xml/network\_security\_config" ... >  
 ...  
 </application>  
</manifest>

### Trust Charles certificate on Device

1. Ensure Charles Proxy is installed and trusted. Settings > General > Profile, open the Charles Proxy item and choose "Install"
2. Ensure under Settings > General > About > Certificate Trust Settings, and find the Charles Proxy certificate, and trust it

### Enable SSL Proxying in Charles

* **Install the Charles Certificate**
  + Select menu item Help-> SSL Proxying...-> Install Charles Root Certificate
  + Be sure to set trust for the certificate to Trust Always (Mac).

Graphical user interface, text, table

Description automatically generated

On Windows Machine, follow the instruction to install the certificate.

* **Configure Charles to see your ssl traffic**
  + Select menu item Proxy-> SSL Proxying Settings...
  + In settings dialog check the Enable SSL Proxying checkbox
  + Click on the Add button
  + In Host: field enter "\*.\*".  This will decrypt all https traffic. To limit this to Adobe server specific calls enter the host domain - **edge.adobedc.net:443**, [**<analytics-tracking-server-domain>**](http://dbs.sc.omtrdc.net/), [**dpm.demdex.net**](http://dpm.demdex.net/)
  + In the Port: field enter "443"
  + Click the OK button, then again click OK to dismiss dialog.

### Configure Charles for testing with your mobile device

Charles can be configured to serve as a proxy server for your mobile device.  To do this, both the Charles host machine (your laptop) and the mobile device **must be on the same wireless network**.

1. Launch Settings
2. Select WiFI
3. Long tap on connected network's name (e.g. on 'NEDBANK')
4. Select Modify network
5. Select Advanced Options
6. Tap on Proxy and select Manual
7. Enter your computer's IP address in Proxy Hostname
8. Enter "8888" for Proxy Port then click Save
9. Launch Chrome on your Android device
10. On your host computer (the one with Charles) click the Allow button in the Charles warning dialog (see image above)
11. In Chrome go to http://charlesproxy.com/getssl
12. In the dialog that pops up provide a name for the certificate ("Charles") and click OK
13. You should now be able to see your mobile devices http/https traffic in Charles

### Use Charles with VPN connection

#### **Desktop/Laptop**

1. Log into VPN
2. Launch Charles Proxy App
3. Launch whatever app you are testing

#### **Device**

1. Log into VPN on host machine (laptop)
2. Launch Charles on host machine
3. Set Proxy on Device to wireless IP of host machine with port 8888  
   \* Do NOT use VPN on your device.  You will be able to reach Internal sites though the proxy server (Charles) on your host machine.

#### **Within a VM**

1. Log into VPN on host machine (laptop)
2. Launch VM
3. Launch Charles within the VM

### Understanding Analytics calls in the Charles Session

* Tracking server for Analytics for Edge SDK:
  + <analytics tracking server> for lifecycle calls
  + edge.adobedc.net > for custom tracking > which in turn sends data to <analytics tracking server>

1. In Charles look for a folder named – “[edge.adobedc.net](http://edge.adobedc.net/)**”**
2. Within the folder check for ‘**interact’** server calls (shown in image below)
3. Navigate to the JSON in the view of the server call and check the folders within **events** to verify the XDM Data (within \_<tenantID> folder)

Note: the **eventType** within xdm folder refers to the type of analytics server call – whether it’s a page view (web.webPageDetails.pageViews) or a link click (web.webInteraction.linkClicks)

Graphical user interface

Description automatically generated

### Verifying Lifecycle Calls

Go to <analytics tracking server>folder and open the network calls within the intended report suite and look for **Lifecycle** value against **internalaction** key – Ensure lifecycle events are firing based on the intended app lifecycle actions eg InstallEvent, LaunchEvent, CrashEvent etc.

Graphical user interface, text, application

Description automatically generated

### Verifying Page View calls (previously trackState server calls)

* For a page view following info would be present in the JSON :
  + **eventType =** web.webPageDetails.pageViews
  + web.webpagedetails.pageviews.value=1
  + web.webpagedetails.name = <pageName> value

Graphical user interface, application

Description automatically generated

### Verifying Link click calls (previously trackAction server calls)

* For a page view following info would be present in the JSON :
  + eventtype=web.webInteraction.linkClicks
  + web.webinteraction.linkclicks.value=1
  + web.webinteraction.name = <button name> value
  + web.webinteraction.type=other OR download OR exit

Graphical user interface, application

Description automatically generated

### Verifying analytics Products variable (previously &&products)

* Old contextData - &&products
* New contextData to verify – productlistitems[i].name

Graphical user interface, application, Word

Description automatically generated

### Verifying event serialization (previously &&events)

* Old contextData - &&events
* New contextData to verify –
  + \_experience > analytics > event1to100 (or similar bucket) > event1 (eventnumber) > id (serialized value)
  + \_experience > analytics > event1to100 (or similar bucket) > event1 (eventnumber) > value (event value)

A picture containing chart

Description automatically generated