L768

Implementing Adobe Audience Manager with Customer ID sync using Adobe Launch

Lab Instructor:
Varun Kalra
Technical Support Consultant

Lab Overview 3

Objectives 3

Getting started 3

Starting and Accessing the test site 4

Prerequisites 4

Chapter 1: Setting Up Server Side Forwarding for Report Suite in Analytics 5

Task 1: Sign in to Experience Cloud using your Adobe ID 5

Task 2: Enable Server Side Forwarding (SSF) in Adobe Analytics 6

Task 3: Editing Appmeasurement Library code 8

Chapter 2: Installing AA and Experience Cloud ID extensions in Launch, by Adobe 10

Task 4: Creating web property and installing extensions 10

Task 5: Installing and setting up Adobe Analytics Extension 12

Task 6: Installing Experience Cloud ID service 14

Chapter 3: Creating Data Elements and Defining Rules 15

Task 7: Identifying Customer ID variable for Authenticated User 15

Task 8: Creating the Data Element 16

Task 9: Defining Rule for firing Adobe Analytics call (Rule 1) 18

Task 10: Defining Rule to collect Customer ID using Experience Cloud Visitor ID service (Rule 2) 22

Chapter 4: Enabling Web Property: Adding Adapters, Environments and Library 26

Task 11: Adding Adapter in Launch 26

Task 12: Creating Environments 27

Task 13: Implementing Launch Embed Codes 31

Task 14: Verify Adobe Analytics and Experience Cloud ID service call 32

For More Hands On 34

Chapter 5: Onboarding offline data to Audience Manager via FTP/S3 (Optional) 34

Task 15: Creating onboarded traits 34

Task 16: Creating inbound file for data onboarding 35

Chapter 6: Working with d_cts flag in DCS API call (Optional) 36

Task 17: Making the API call 36

Addendum 37

1: Creating Report Suite in Adobe Analytics UI 37

2: Downloading Appmeasurement and Audience Management Module Library code 38

3: Creating Data Source in Audience Manager 39

Lab Overview

The goal is of this lab is to guide you through Adobe Audience Manager implementation using Server-Side Forwarding method. We will be using Adobe Launch – Adobe's Tag Management System to implement and configure AA, AAM and Experience Cloud ID service

Objectives

- Adding Adobe Analytics (AA) extension in Launch with Audience Management Module.
- Implementing Customer ID sync using Experience Cloud ID service tool.
- Setting up extensions, adapters, environments and libraries.
- Discussing offline data ingestion flow.
- Working with DCS api with d_cts flag.

Getting started

- Find L768 user on the screen to login
- Use Google Chrome to access the sample website and Experience Cloud UI.
- Your sign in ID for Adobe Experience Cloud sandbox environment is: L768+<seat##>@adobeeventlab.com

Note: Adobe Experience Cloud credentials will be deactivated after summit.

Starting and Accessing the test site

1. Launch a terminal window. You can do this by hitting cmd + space, then typing terminal in spotlight search, and hitting enter.
2. Change the directory to Desktop:
   `cd desktop/samplesite`
3. Start the node js server by following command:
   `http-server`
4. The sample site will now start and you can access it from your browser with http://localhost:8080
5. You can edit the HTML files using Text Edit or Brackets application later.

Prerequisites

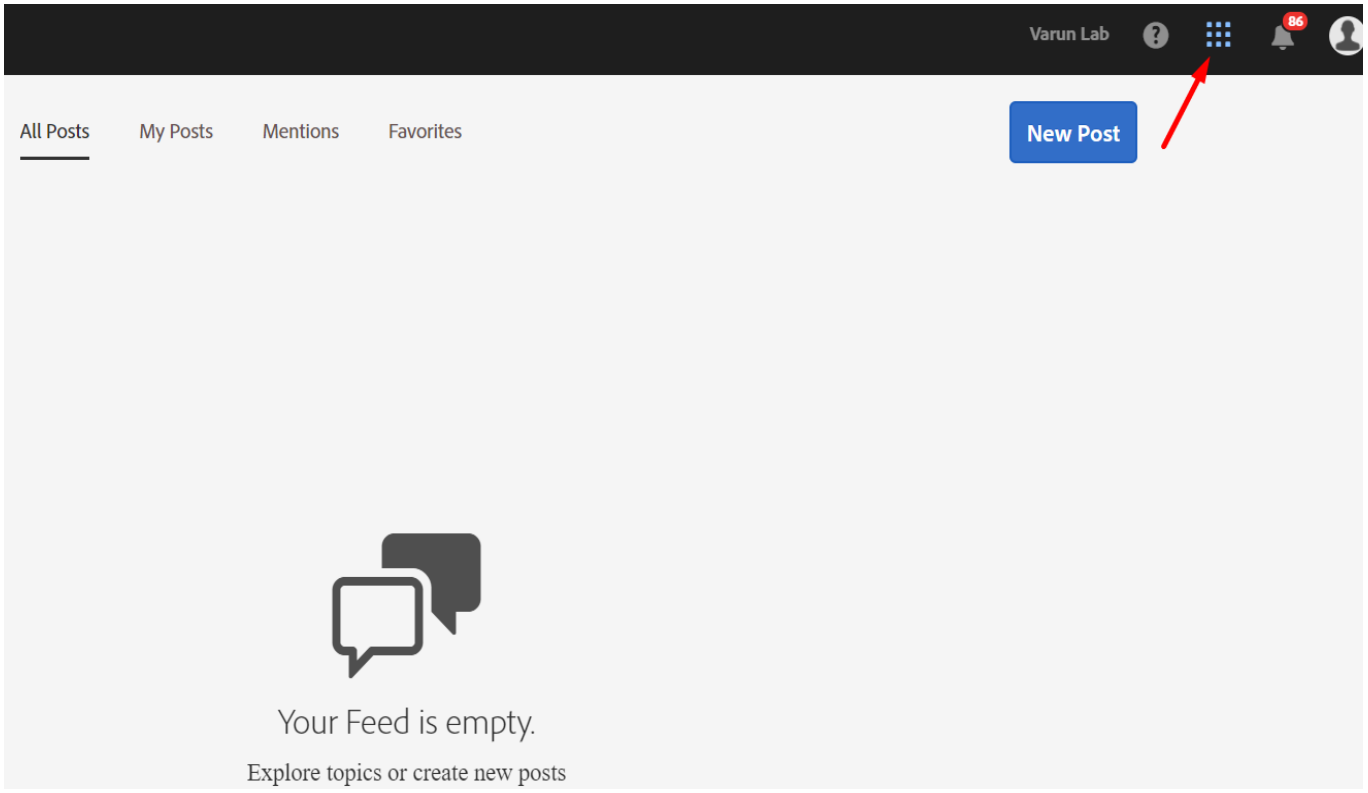
1. A website with log in functionality (CRM ID).
2. Experience Cloud Visitor ID service.
3. Server-side forwarding requires:
   Appmeasurement.js code library (version 1.5 or newer)
   AppMeasurement_Module_AudienceManagement.js
   Experience Cloud Visitor ID service
4. AA, AAM and Launch as a part of one Experience Cloud Org account.
5. Admin access for AA, AAM, and Launch via Experience Cloud.
6. A Report Suite in Adobe Analytics.
7. A Cross Device Data Source in Audience Manager.

Chapter 1: Setting Up Server Side Forwarding for Report Suite in Analytics

In this Chapter, we will turn on SSF on Report Suite level within Adobe Analytics UI. Then we will set up the AppMeasurement Library code.

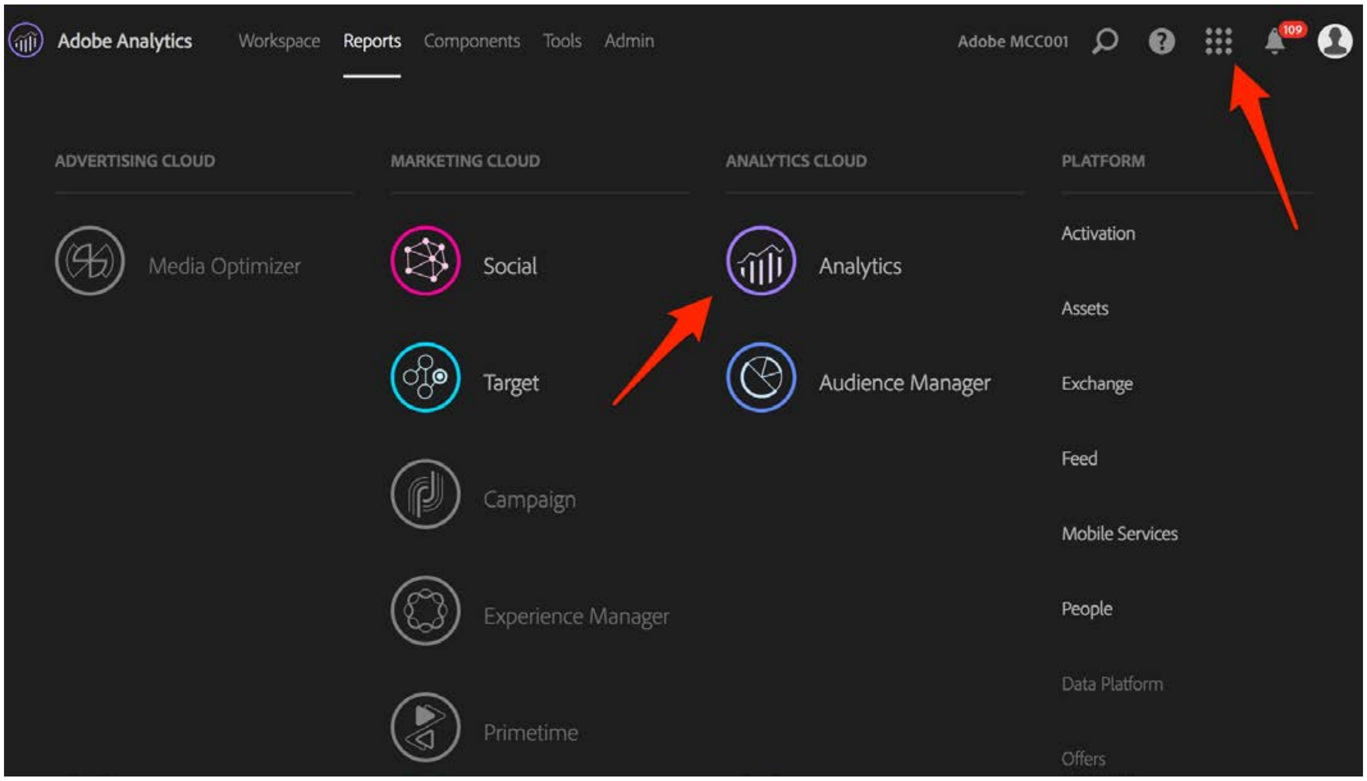
Task 1: Sign in to Experience Cloud using your Adobe ID

i. Your Adobe ID is in the format L768+<seat##>@adobeeventlab.com for example if your seat number is 5 then use: L768+05@adobeeventlab.com
ii. URL for sign in to Experience Cloud: https://aamlab.marketing.adobe.com



iii. Once you sign in, you will see feeds by default. You can click on the application switcher (9 dots) at the top right-hand side corner to access required application's UI.

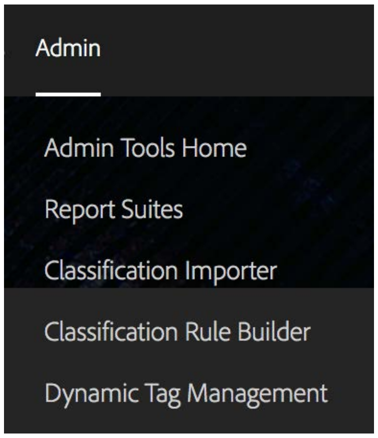
iv. Choose Analytics from the resulting navigation menu.



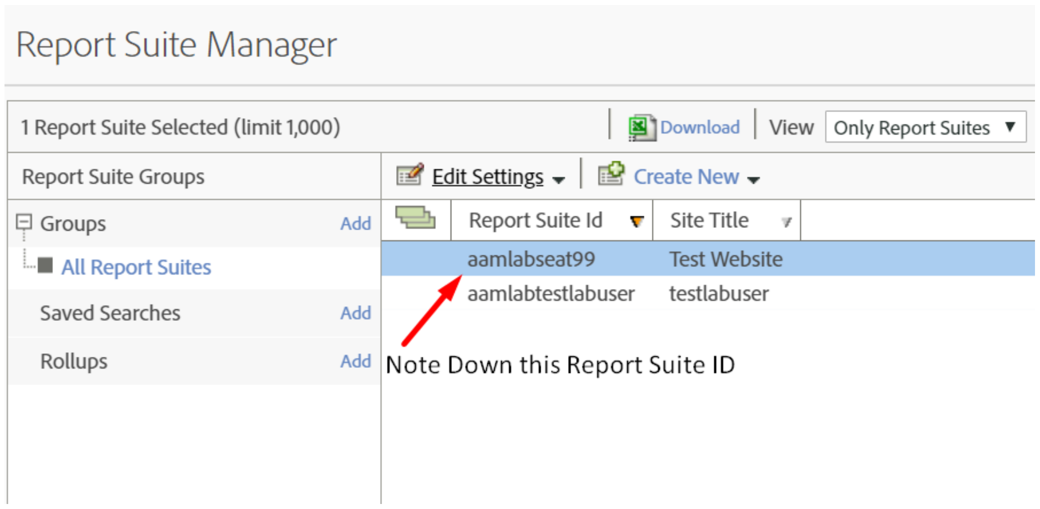
Task 2: Enable Server Side Forwarding (SSF) in Adobe Analytics

Note: The Report Suite has already been created for you. So, you can just check the report suite with the name: seat<##> and get the report suite ID and enable SSF. Example Report Suite ID: aamlabseat01
To learn how to create a Report Suite, please see Addendum 1.

i. While logged into Analytics, click on **Admin** > **Report Suites**



ii. Select your Report Suite from the list and note down your Report Suite ID for later.



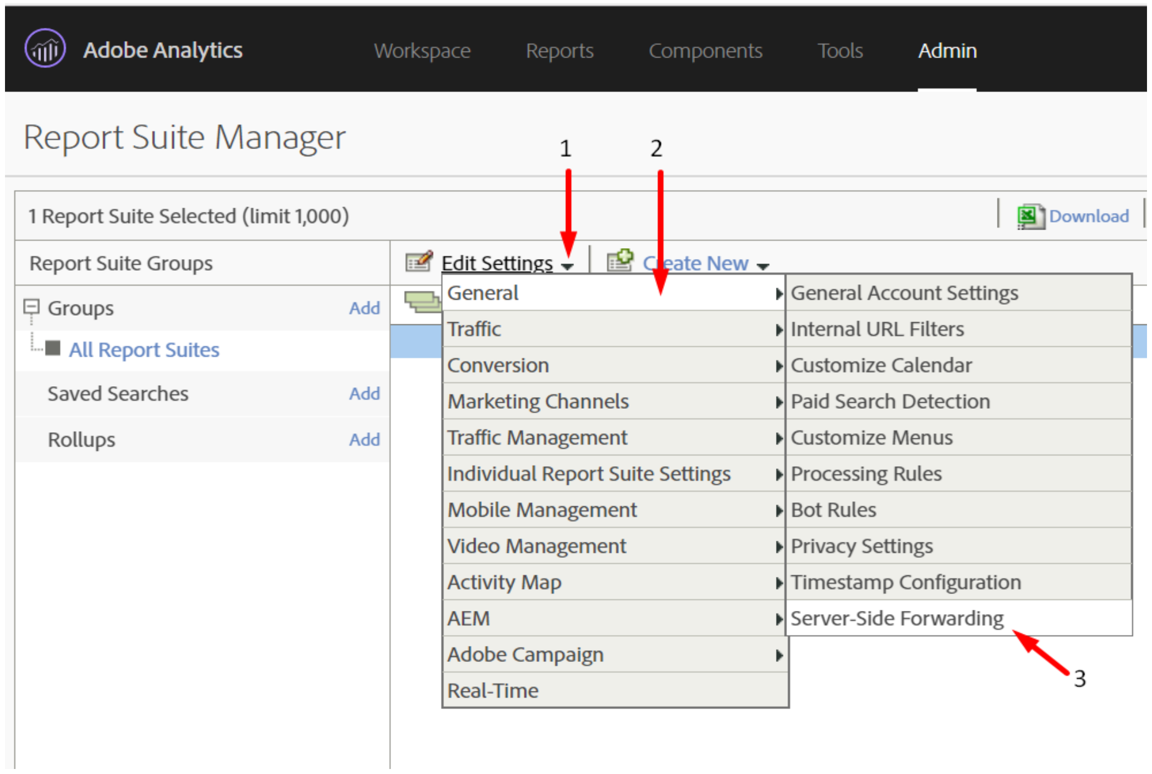
Report Suite Manager

1 Report Suite Selected (limit 1,000) | Download | View: Only Report Suites

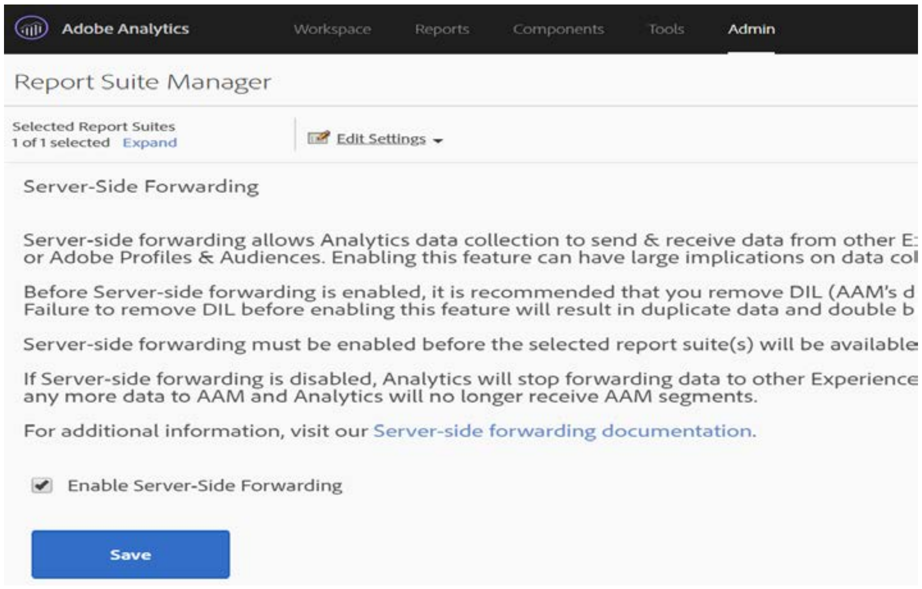
Report Suite Id	Site Title
aamlabseat99	Test Website
aamlabtestlabuser	testlabuser

Note Down this Report Suite ID

Go to **Admin** > **Report Suites** > Select your Report Suite from the list > Hover over **General** > Click on **Server Side Forwarding:**



iii. Then you will be redirected to the following page, where you will see a check box for SSF. Check **Enable Server Side Forwarding** and click on Save, and then Confirm.



Server-Side Forwarding

Server-side forwarding allows Analytics data collection to send & receive data from other E
or Adobe Profiles & Audiences. Enabling this feature can have large implications on data col

Before Server-side forwarding is enabled, it is recommended that you remove DIL (AAM's d
Failure to remove DIL before enabling this feature will result in duplicate data and double b

Server-side forwarding must be enabled before the selected report suite(s) will be available

If Server-side forwarding is disabled, Analytics will stop forwarding data to other Experience
any more data to AAM and Analytics will no longer receive AAM segments.

For additional information, visit our Server-side forwarding documentation.

✓ Enable Server-Side Forwarding

Task 3: Editing Appmeasurement Library code

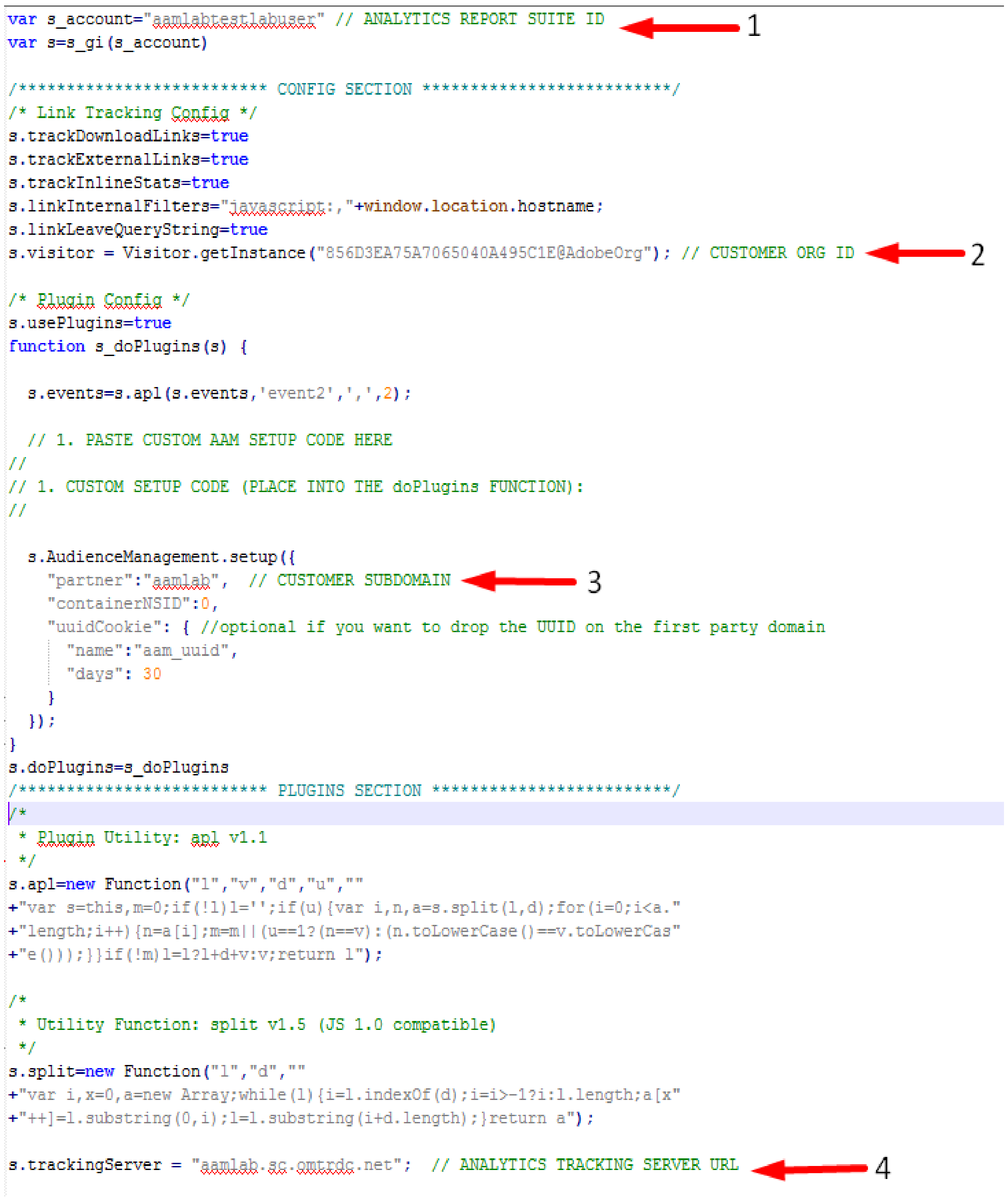
Note: The required library code is present on your Mac, so no need to download. Go to Desktop > Assets > AppMss.js. However, this code needs to be edited with your AA and AAM account settings.
To know how to download and merge code libraries, check Addendum 2.

Open the AppMss.js file in Text edit or Brackets, and edit following values (only the first value i.e. Report Suite ID needs to be added by you, rest three values are already there in the file):

i. For the variable, **s_account** enter your report suite id as its value. (Example: aamlab<seat##>)
ii. For the variable **s.visitor**, we are using Experience Cloud Org ID. The value for s.visitor is: **856D3EA75A7065040A495C1E@AdobeOrg**
iii. For **partner** under AAM set up code, we are using this value (AAM subdomain): **`aamlab`**
iv. For the variable **s.trackingServer**, we are using the value: **`aamlab.sc.omtrdc.net`**

Now that you've made the necessary changes, save the file. You'll also need to copy the contents of this file to be pasted inside Adobe Launch in a later step.

The final AppMss.js file should look like following:



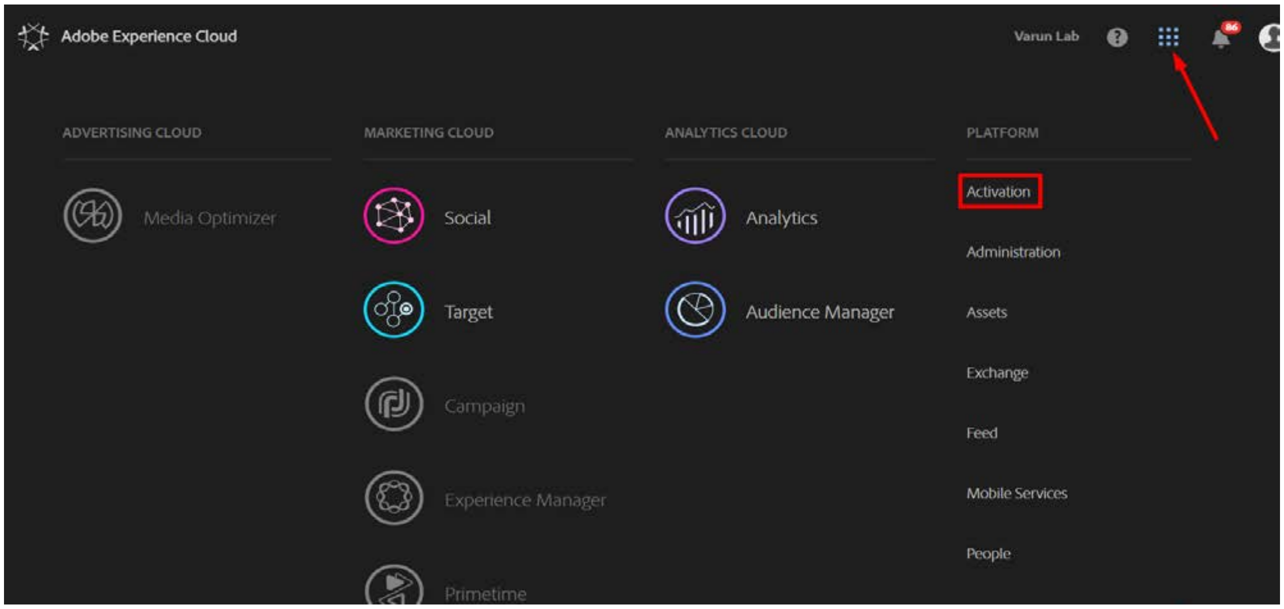
```
var s_account="aamlabtestlabuser" // ANALYTICS REPORT SUITE ID
var s=s_gi(s_account)

/************************** CONFIG SECTION **************************/
/* Link Tracking Config */
s.trackDownloadLinks=true
s.trackExternalLinks=true
s.trackInlineStats=true
s.linkInternalFilters="javascript:,"+window.location.hostname;
s.linkLeaveQueryString=true
s.visitor = Visitor.getInstance("856D3EA75A7065040A495C1E@AdobeOrg"); // CUSTOMER ORG ID

/* Plugin Config */
s.usePlugins=true
function s_doPlugins(s) {

  s.events=s.apl(s.events,'event2',',',2);

  // 1. PASTE CUSTOM AAM SETUP CODE HERE
//
// 1. CUSTOM SETUP CODE (PLACE INTO THE doPlugins FUNCTION):
//

  s.AudienceManagement.setup({
    "partner":"aamlab",  // CUSTOMER SUBDOMAIN
    "containerNSID":0,
    "uuidCookie": { //optional if you want to drop the UUID on the first party domain
      "name":"aam_uuid",
      "days": 30
    }
  });
}
s.doPlugins=s_doPlugins
/************************** PLUGINS SECTION **************************/
/*
 * Plugin Utility: apl v1.1
 */
s.apl=new Function("l","v","d","u",""
+"var s=this,m=0;if(!l)l='';if(u){var i,n,a=s.split(l,d);for(i=0;i<a."
+"length;i++){n=a[i];m=m||(u==1?(n==v):(n.toLowerCase()==v.toLowerCas"
+"e()));}}if(!m)l=l?l+d+v:v;return l");

/*
 * Utility Function: split v1.5 (JS 1.0 compatible)
 */
s.split=new Function("l","d",""
+"var i,x=0,a=new Array;while(l){i=l.indexOf(d);i=i>-1?i:l.length;a[x"
+"++]=l.substring(0,i);l=l.substring(i+d.length);}return a");

s.trackingServer = "aamlab.sc.omtrdc.net";  // ANALYTICS TRACKING SERVER URL
```

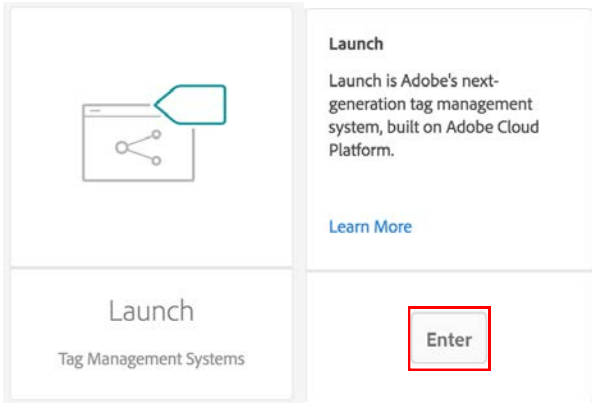
Chapter 2: Installing AA and Experience Cloud ID extensions in Launch, by Adobe

In this Chapter, we will be setting up Adobe Analytics (AA) extension in Launch with Appmeasurement Library code. Then we will set up Experience Cloud ID service.

Task 4: Creating web property and installing extensions

i. Access Launch from Experience Cloud UI. Click on the application switcher > **Activation >** Hover over the **Launch Card >** Click **Enter**
Alternatively, you can access this URL to access Launch: https://launch.adobe.com

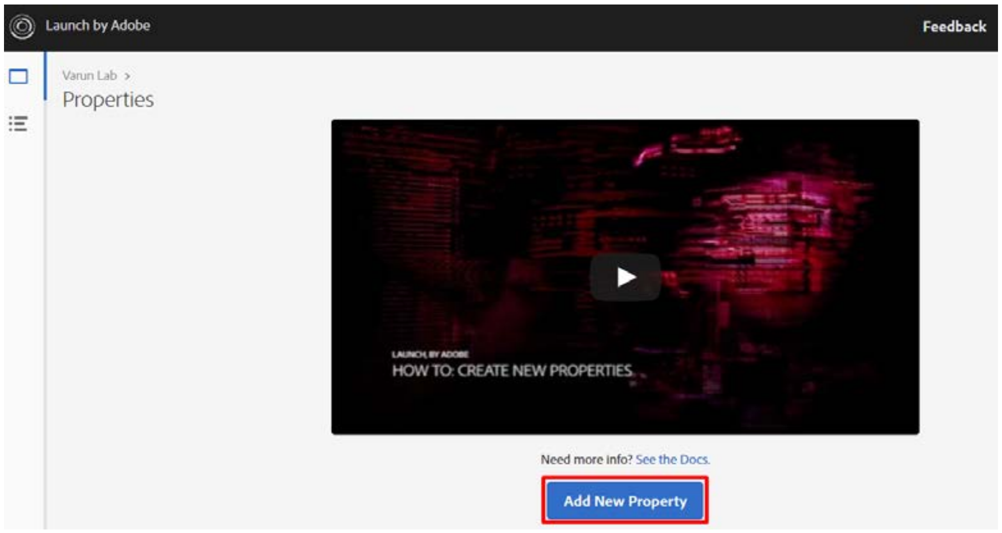




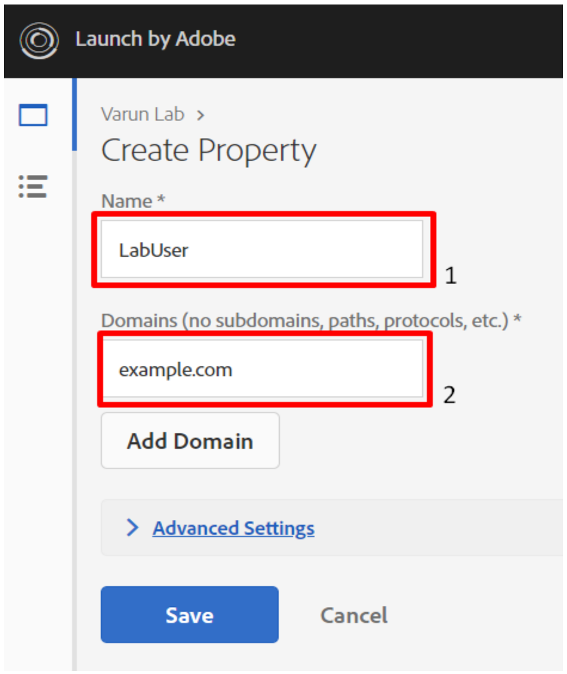
Launch

Launch is Adobe's next-generation tag management system, built on Adobe Cloud Platform.

ii. Click the **Add New Property** button:



iii. Name your property with whatever you would like. You can use same name as your report suite.



Name *: LabUser

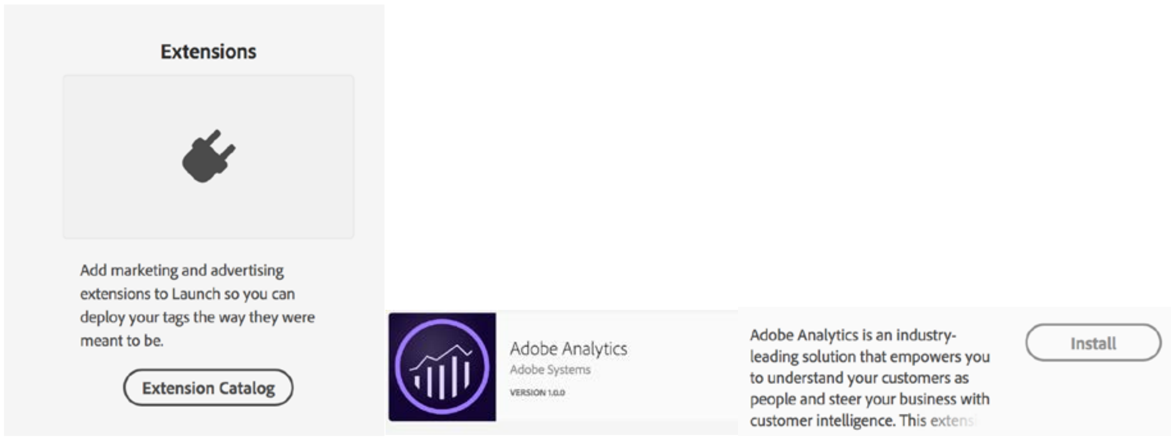
Domains (no subdomains, paths, protocols, etc.) *: example.com

iv. Use example.com as the domain name, or enter any test domain name there, since our website is hosted locally.

v. Keep Advanced Settings as default and click **Save**. Your new property should display in the list of properties.

Task 5: Installing and setting up Adobe Analytics Extension

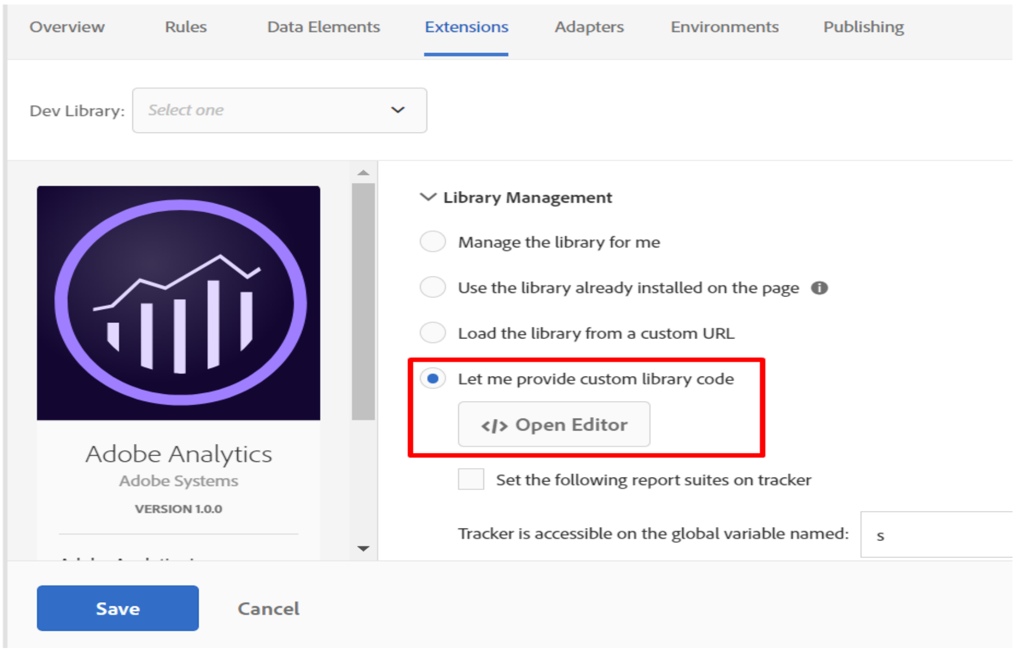
i. Click on your Web Property name from the list and go to Extensions tab → **Catalog** and install the Adobe Analytics extension.



Add marketing and advertising extensions to Launch so you can deploy your tags the way they were meant to be.

Adobe Analytics is an industry-leading solution that empowers you to understand your customers as people and steer your business with customer intelligence. This extens

ii. Under Library Management > Select the last option that says **Let me provide custom library code**.



Dev Library: Select one

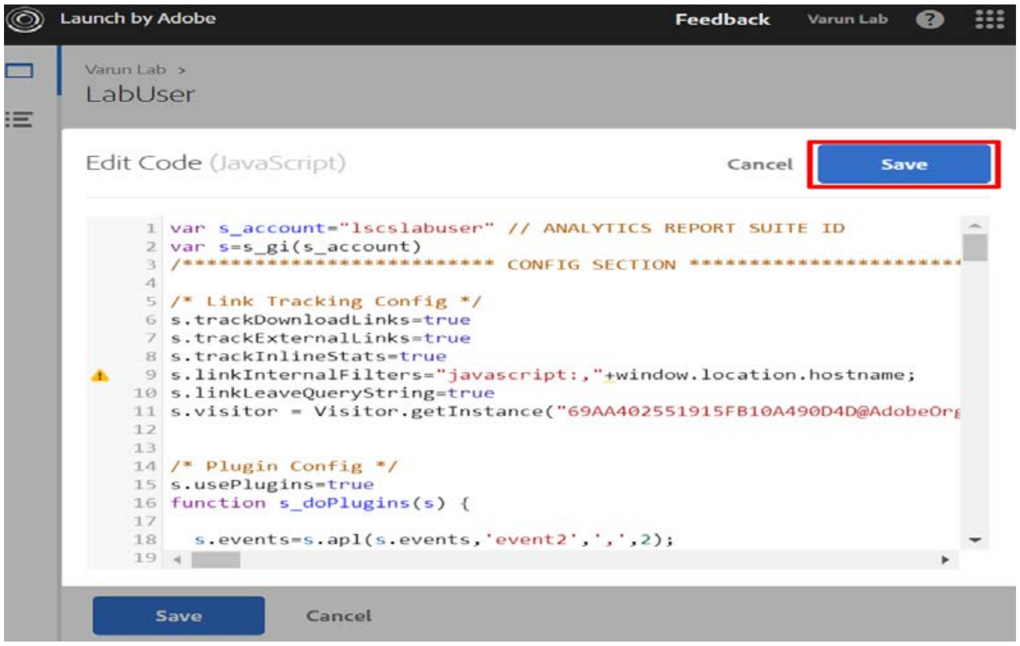
Library Management

- Manage the library for me
- Use the library already installed on the page
- Load the library from a custom URL
- ● Let me provide custom library code

Set the following report suites on tracker

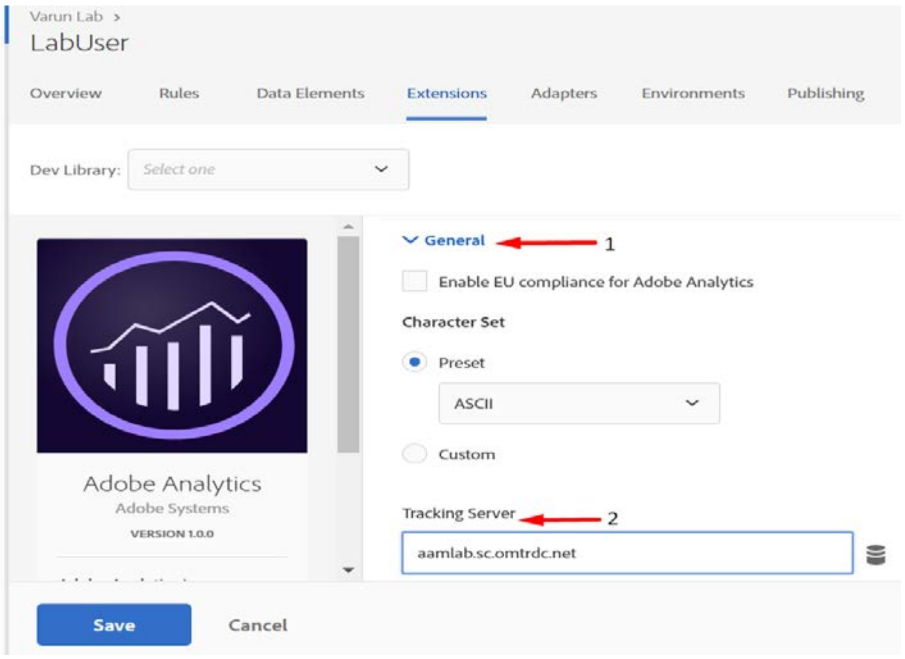
Tracker is accessible on the global variable named: s

iii. Click on **</> Open Editor** and copy the code that you set up in Chapter 1 Task 3 from AppMss.js file and paste that code in the window, and save.



```
var s_account="lscslabuser" // ANALYTICS REPORT SUITE ID
var s=s_gi(s_account)
/*************************** CONFIG SECTION ************************

/* Link Tracking Config */
s.trackDownloadLinks=true
s.trackExternalLinks=true
s.trackInlineStats=true
s.linkInternalFilters="javascript:,"+window.location.hostname;
s.linkLeaveQueryString=true
s.visitor = Visitor.getInstance("69AA402551915FB10A490D4D@AdobeOrg


/* Plugin Config */
s.usePlugins=true
function s_doPlugins(s) {

  s.events=s.apl(s.events,'event2',',',2);
```

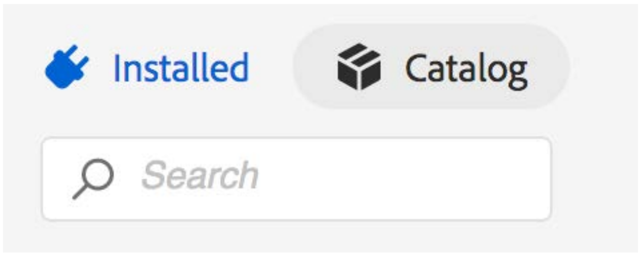
iv. Scroll down and expand **General**, then under tracking server enter your tracking server and then **Save** Extension:



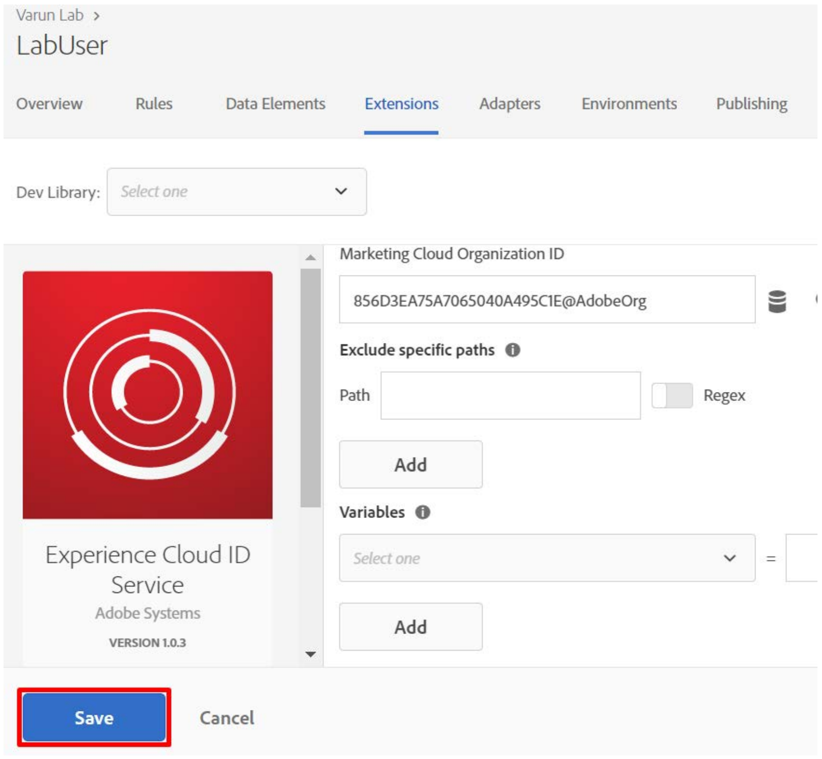
Optionally, you can expand the global variables section to add global values to variables like pagename, evars and props.

Task 6: Installing Experience Cloud ID service

i. After saving the previous extension, you'll be taken to the Installed Extensions page. We will be installing another extension so return to the catalog by clicking the Catalog button.



ii. Install **Experience Cloud ID Service** just as you did with the previous extension. You can leave the default settings and click **Save** to finish the install.



Dev Library: Select one

Marketing Cloud Organization ID: 856D3EA75A7065040A495C1E@AdobeOrg

Exclude specific paths

Path

Regex

Variables: Select one =

Next, we need to define some actions for these extensions to take and set conditions for those actions. We will do this in the next chapter with rules and data elements.

Chapter 3: Creating Data Elements and Defining Rules

In this Chapter, we need to create a Data Element that captures the Customer ID when the user signs in to the website. We will then create rules to fire Analytics Beacon and get the Customer ID in Experience Cloud Visitor ID call.

Task 7: Identifying Customer ID variable for Authenticated User

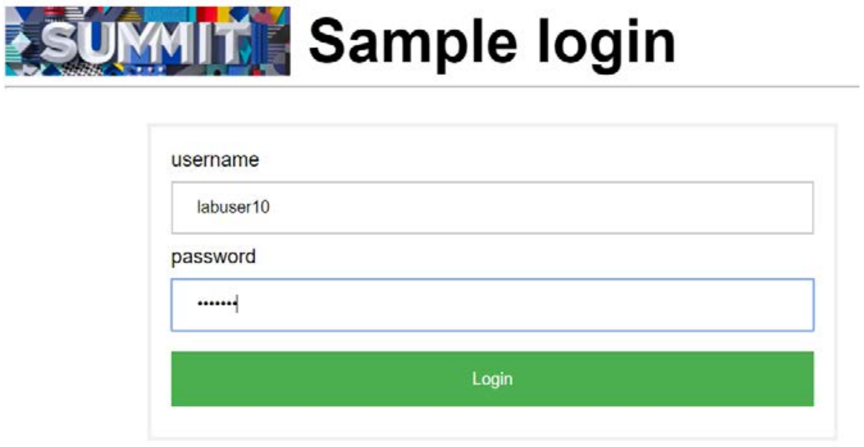
i. When a registered user signs in to the site, then they may use their username or email address. But, in that site's CMS database, that user is identified and stored as a Customer ID or a CRM ID.

ii. This Customer ID is responsible for maintaining a customer's session and their details might be keyed off this ID in your CRM. The customer ID in a browser can be stored in local storage, Cookie, JS variable or a hidden HTML element. We need to find out the variable that is keeping this Customer ID and then we can use it as a Data Element.

iii. For this lab session, we have a simple website with log in functionality, that stores the Customer ID in browser's Local Storage. The variable that is storing the customer ID is "username".

iv. You can access the website by this address in browser: http://localhost:8080 and log in using pre-created credentials:

Username: labuser# (example labuser1)
Password: same as username

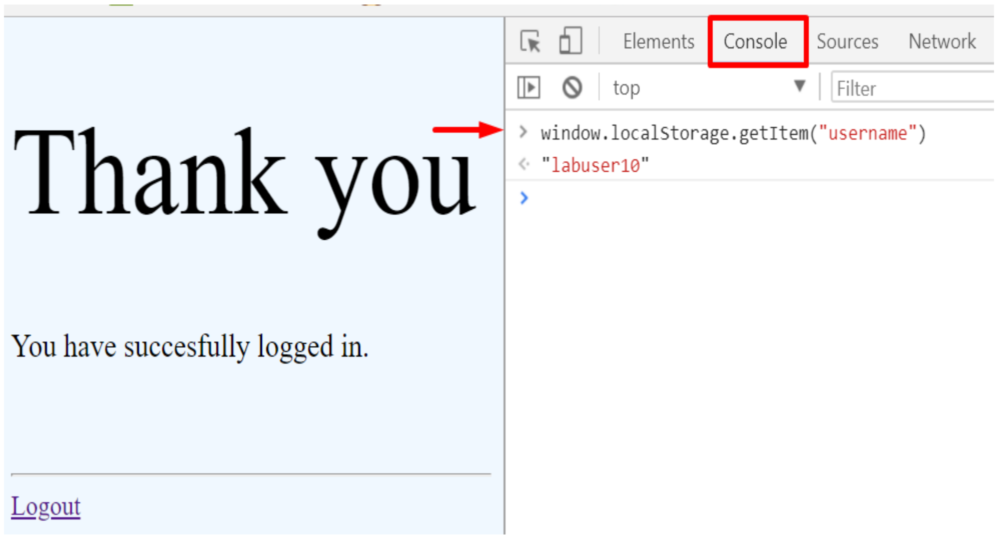


username: labuser10

v. Once you've logged in, check the variable in browser's console by opening Developer Tools. In Chrome, you can use the keyboard shortcut (command + Option + J) or choose View > Developer > Developer Tools from the top menu bar. Once the developer tools window is open, click on the Console tab. Inside the console tab type:

`window.localStorage.getItem("username")`

Doing so will return the value of this variable.

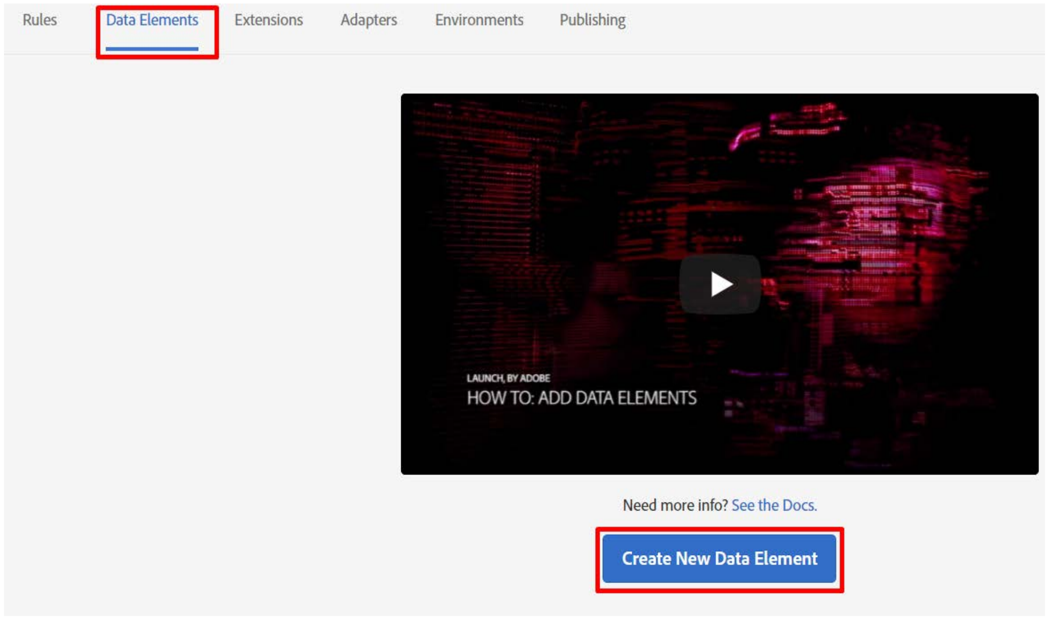


```
> window.localStorage.getItem("username")
< "labuser10"
```

i. Now we can see, **`username`** is the variable in local storage that we can use to create the Data Element.

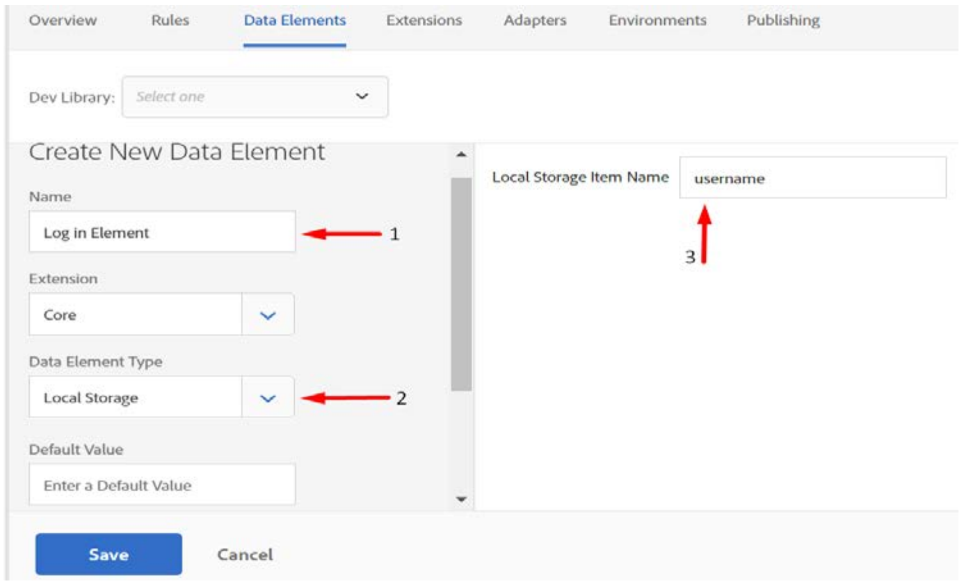
Task 8: Creating the Data Element

i. Go to **Data Element** tab, and click on **Create New Data Element:**



ii. Enter a name for Data Element. The extension should remain Core by default.

iii. Select Data Element Type as **Local Storage** from the drop-down list. (you can also type local storage for it to appear). Local Storage item name is “username” (as we found out in previous task).



Dev Library: Select one

Create New Data Element

Name: Log in Element

Extension: Core

Data Element Type: Local Storage

Default Value: Enter a Default Value

Local Storage Item Name: username

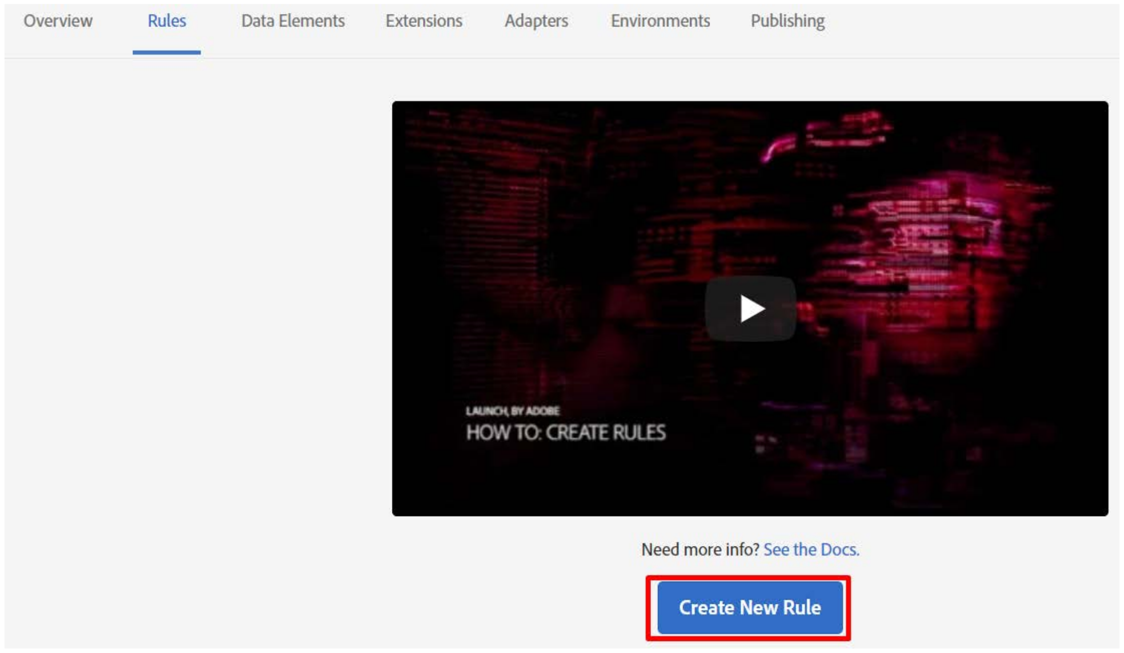
iv. Leave all other settings as default and save the data element.

Optionally, you can create another data element to capture page info in it.

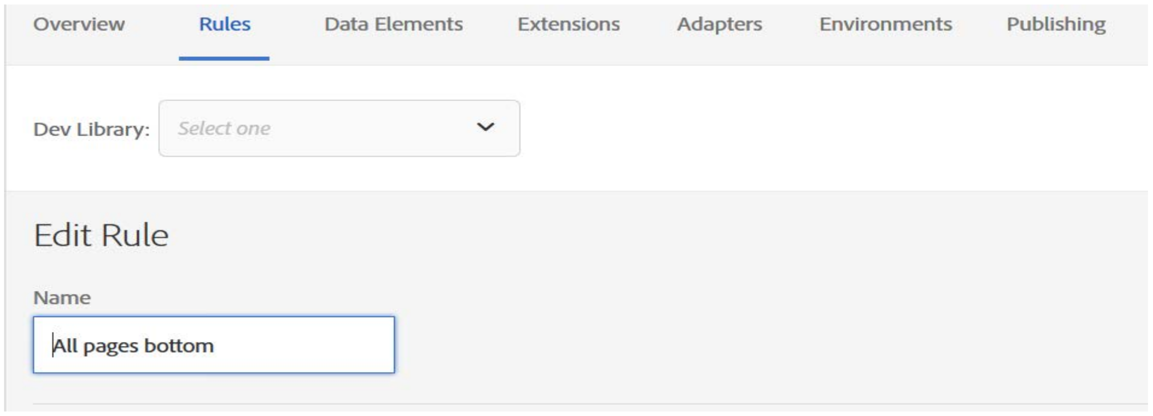
Task 9: Defining Rule for firing Adobe Analytics call (Rule 1)

IF event == Page bottom, **THEN** action → AA– set variables AND AA– Send Beacon

i. Go to **Rules** tab and click on **Create New Rule:**



ii. You can name it anything you would like. Let's name it "All Pages Bottom"**.**



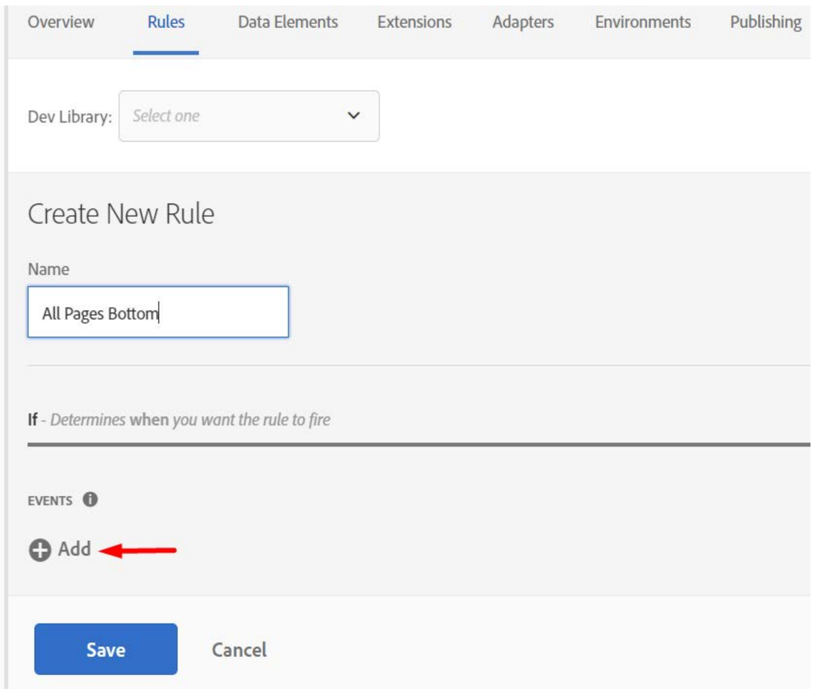
Dev Library: Select one

Edit Rule

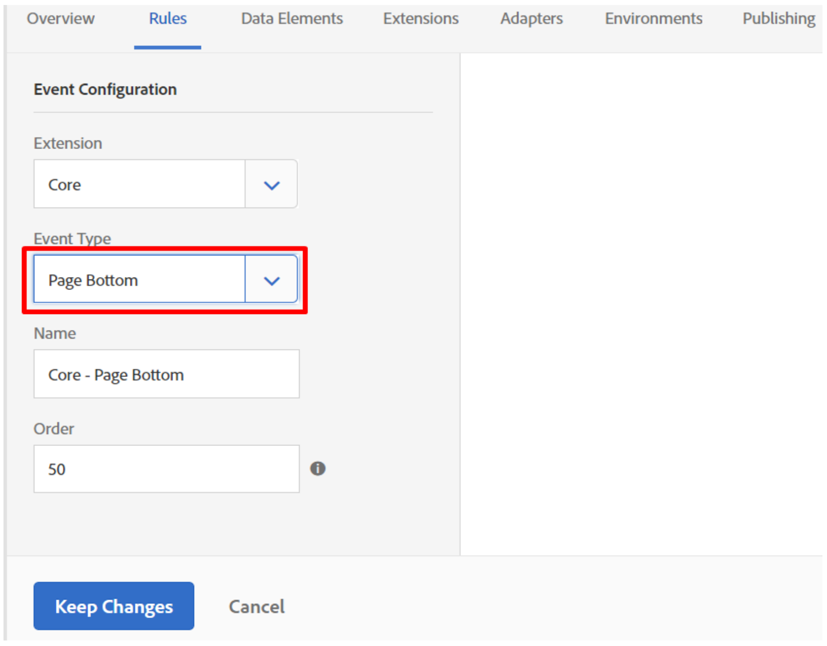
Name

All pages bottom

iii. Under **IF** condition, we will add the event when we want the rule to fire. Click on the + sign to add event:



iv. Extension should be Core, then Select the Event Type as Page Bottom from the drop-down list and click on Keep Changes.



Event Configuration

Extension: Core

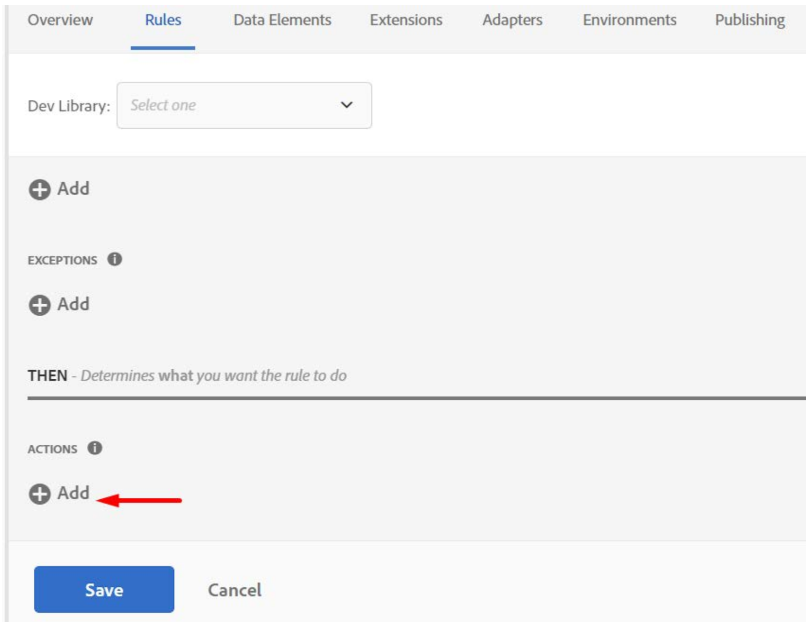
Event Type: Page Bottom

Name: Core - Page Bottom

Order: 50

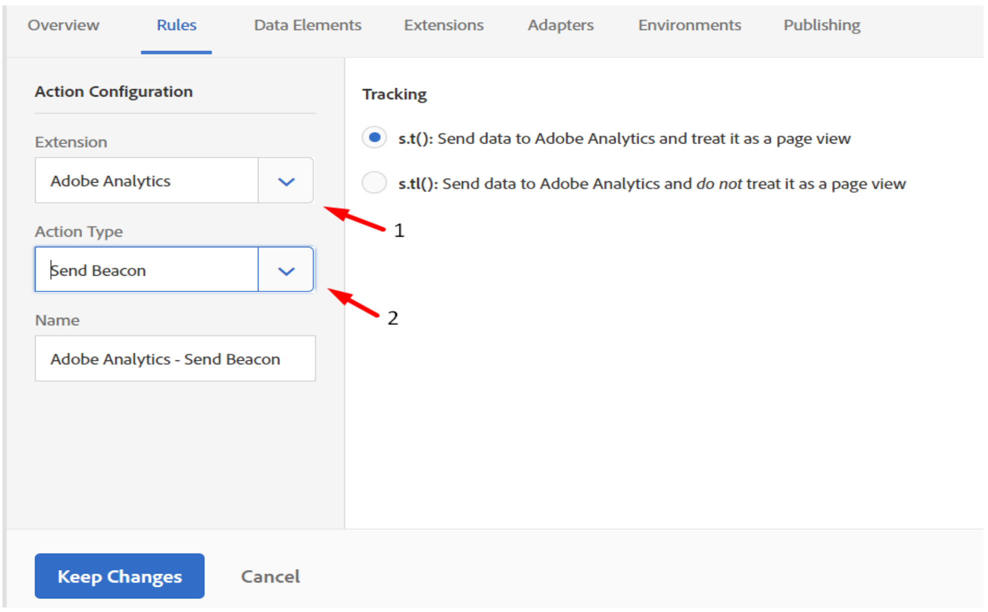
v. Then we'll be back to the rules tab under same rule. Now we need to add actions under **THEN** for this rule.

vi. Click on the + sign under **THEN**, to define the action.



vii. Under Action Configuration, select Adobe Analytics as extension, and Action type as Send Beacon. Leave all other settings as default. Then click on Keep Changes.

Optionally you can create another action for setting AA variables.



Extension: Adobe Analytics

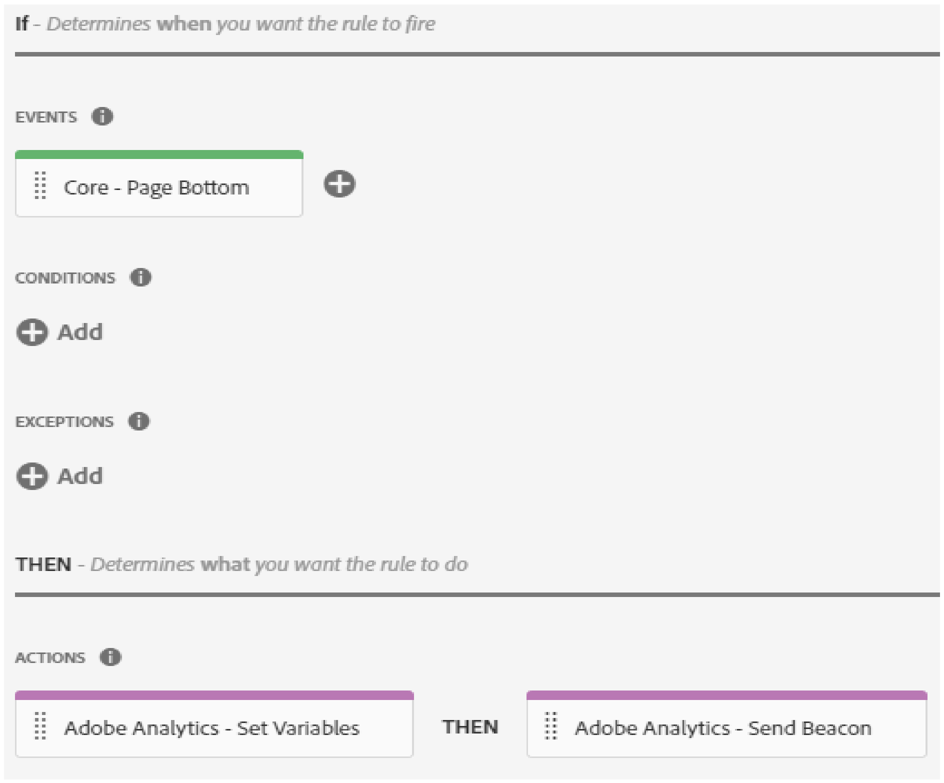
Action Type: Send Beacon

Name: Adobe Analytics - Send Beacon

[✓ s.t(): Send data to Adobe Analytics and treat it as a page view]

s.tl(): Send data to Adobe Analytics and *do not* treat it as a page view

viii. This Rule's settings should appear like following:


If - *Determines when you want the rule to fire*

EVENTS

Core - Page Bottom

CONDITIONS

Add

EXCEPTIONS

Add

THEN - *Determines what you want the rule to do*

ACTIONS

Adobe Analytics - Set Variables THEN Adobe Analytics - Send Beacon

Outcome of this task: When the page finishes loading (Page Bottom), the Analytics Beacon will fire with the variables we assigned.

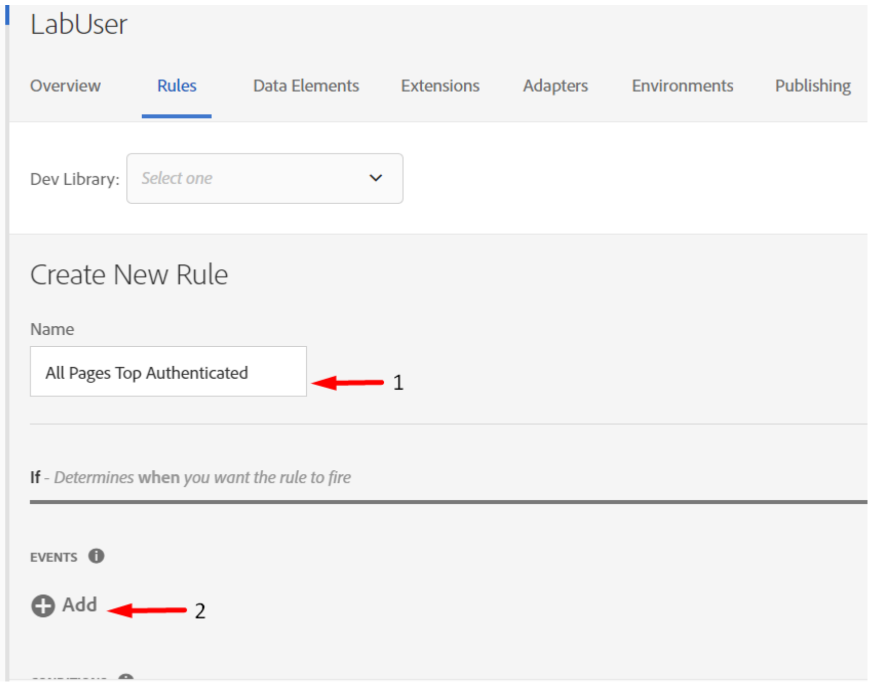
Task 10: Defining Rule to collect Customer ID using Experience Cloud Visitor ID service (Rule 2)

IF event == Page Top, **CONDITION** == Logged in **THEN** action → Experience Cloud ID Service– Set Customer ID

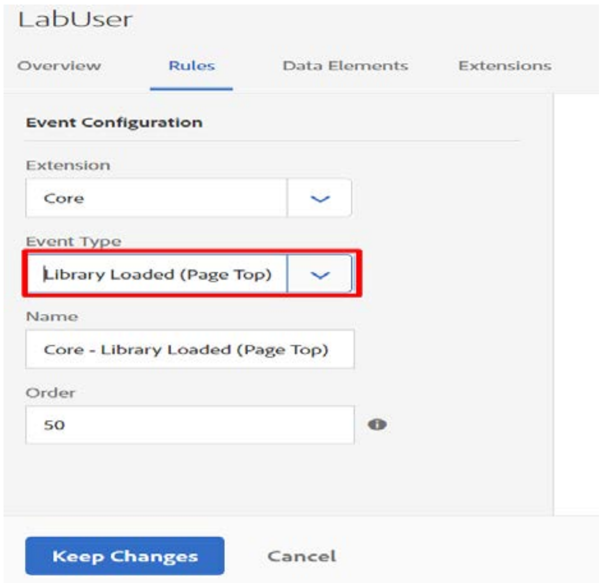
Note: The data source is already created for you. You can view the data source settings in AAM UI. Integration code for the data source is crm_id. To learn how to create a Data Source, check Addendum 3.

Once we have Data Source ready we can create a rule to fire Visitor ID service call when the user logs in and collect their Customer ID (declared ID).

i. In Launch, go to rules tab of your web property. Give this rule a name: All Pages Top Authenticated.
ii. Under **IF** condition, click on + sign to add an **Event**



iii. For the event configuration, keep the extension as **Core**, and select event type as **Library Loaded (Page Top)** from the drop-down list, and click **Keep Changes.**



Event Configuration

Extension: Core

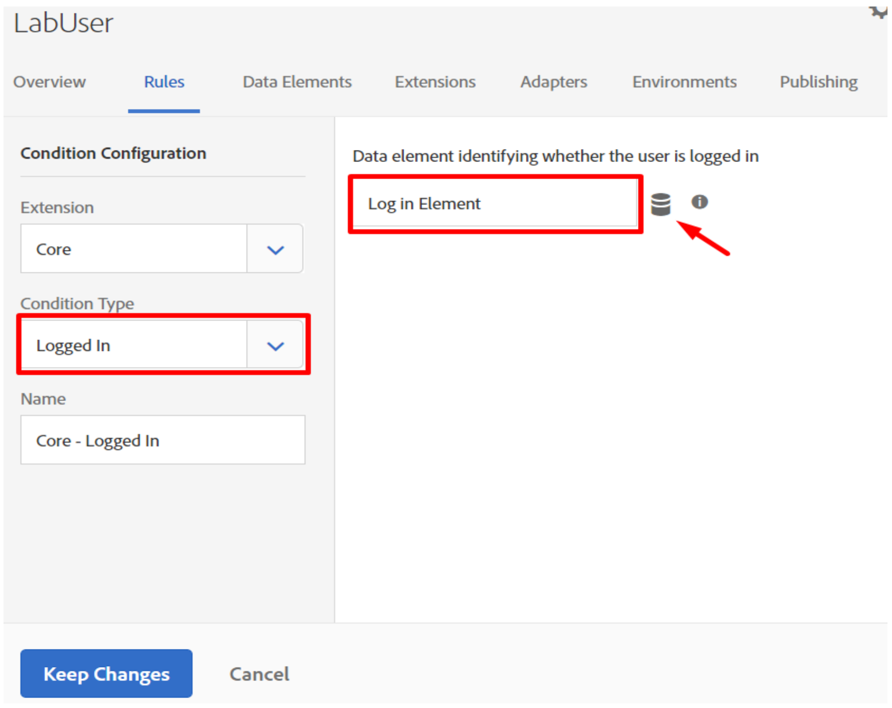
Event Type: Library Loaded (Page Top)

Name: Core - Library Loaded (Page Top)

Order: 50

iv. Now, under **CONDITIONS**, we need to add a condition to check logged in status of the site visitor. For this click on the + sign under **CONDITIONS**, then under Condition Configuration, the extension should be Core, and select Condition Type as **Logged In** from the drop down.

v. Select the data element as Log in Element that we created in previous Chapter, and then click **Keep Changes**.



Condition Configuration

Extension: Core

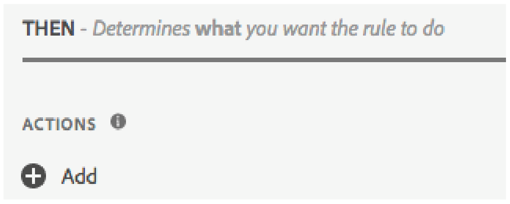
Condition Type: Logged In

Name: Core - Logged In

Data element identifying whether the user is logged in: Log in Element

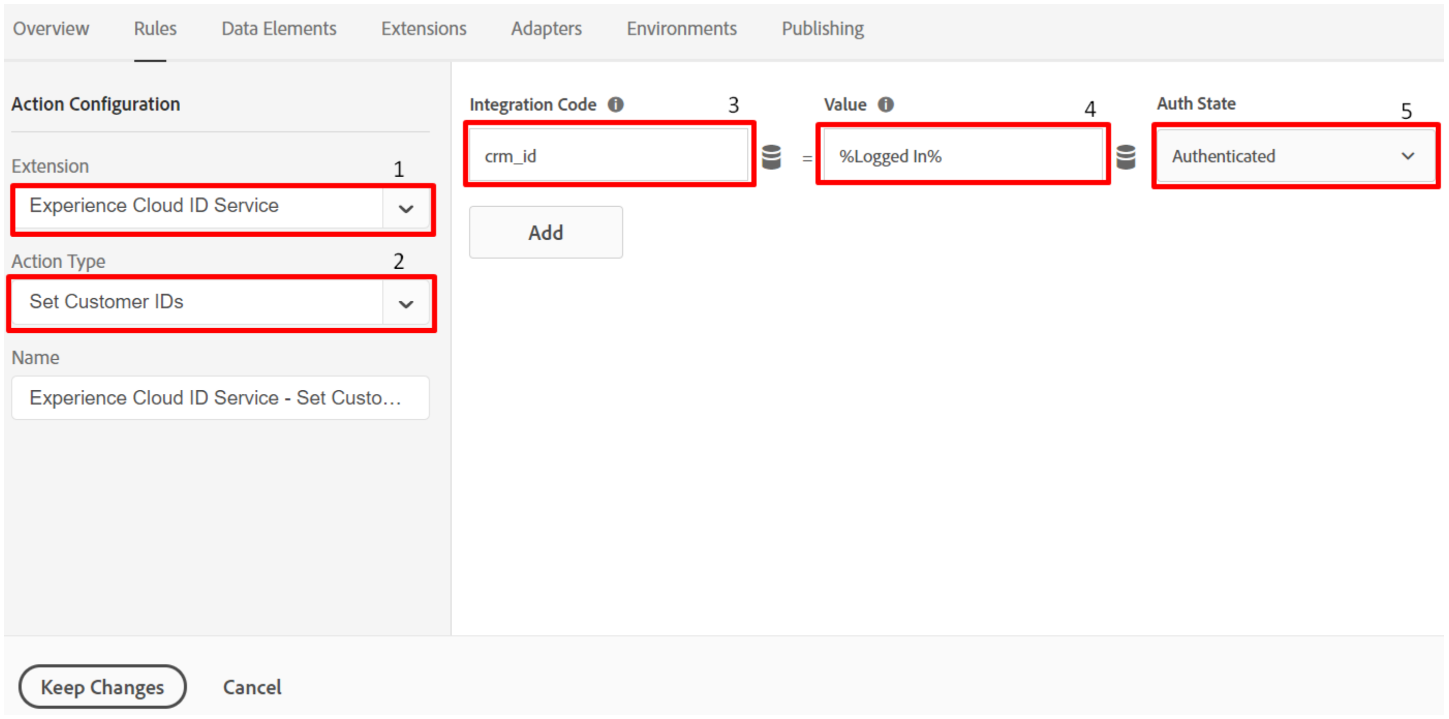
In Next step, we will be adding an action to grab Customer ID when the user signs in and collect that customer ID in our AAM Cross Device Data Source.

vi. You'll be back into your rule now, scroll down and under **THEN** click on the **+** sign to add an **Action**. Under Action Configuration, select Extension **as Experience Cloud ID service** from list.



vii. Select Action Type as **Set Customer IDs** from the drop down.
Enter the same Integration code for your Data Source that is **crm_id**

In the value field, add the data element that we created in Chapter 2 (You can check the list of all available Data Elements by clicking on the icon). For Auth State, select "Authenticated":



Extension: Experience Cloud ID Service

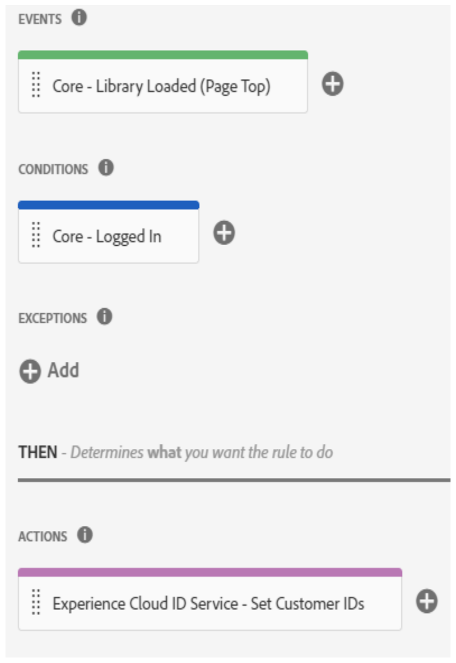
Action Type: Set Customer IDs

Name: Experience Cloud ID Service - Set Custo…

Integration Code: crm_id = Value: %Logged In%

Auth State: Authenticated

Now click on **Keep Changes** and Save the rule. This rule's exact settings should appear like this:


EVENTS

Core - Library Loaded (Page Top)

CONDITIONS

Core - Logged In

EXCEPTIONS

THEN - *Determines what you want the rule to do*

ACTIONS

Experience Cloud ID Service - Set Customer IDs

Outcome of this task: When the user logs in, the Experience Cloud ID call will fire and sync Customer ID with Device ID.

This call will have a parameter: d_cid_ic == integration code%01Customer ID
An integration code is an alternate ID you can use instead of the Data Source ID

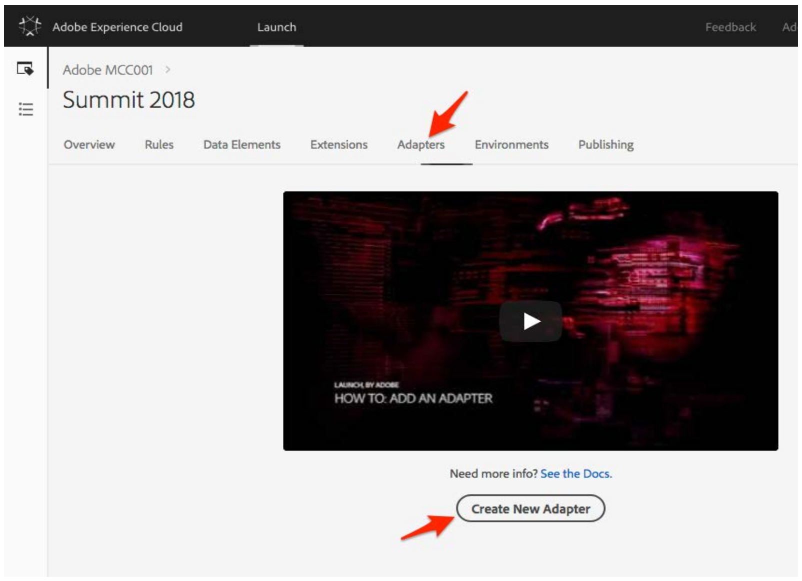
Chapter 4: Enabling Web Property: Adding Adapters, Environments and Library

In this chapter, we will add an adapter, create environments and deploy a library in Launch. Next, we will implement our library on the website by adding the embed code.

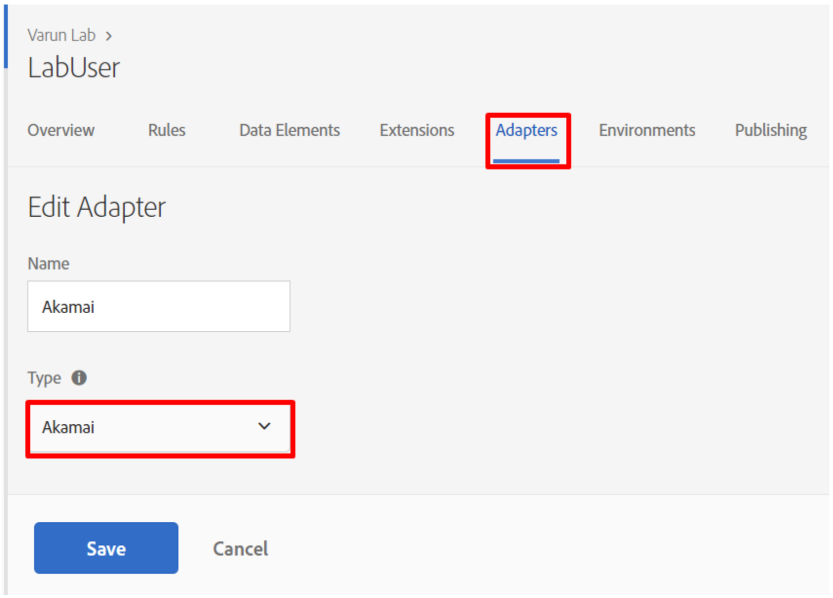
Task 11: Adding Adapter in Launch

An "Adapter" refers to the destination(s) where you want to deploy your Launch libraries—on Adobe's Akamai instance or on your own web servers. For this exercise we will create an Akamai Adapter.

i. Click **Adapters** in the Property navigation, then click **Create New Adapter:**



ii. Name the Adapter— "Akamai" and select the **Akamai** type from the dropdown and click **Save**:



Edit Adapter

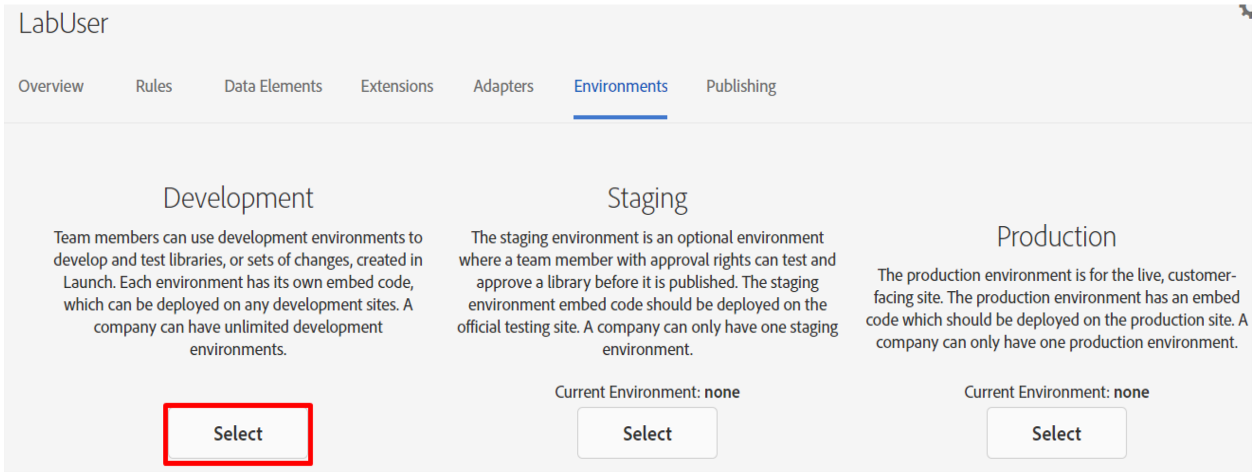
Name: Akamai

Type: Akamai

Task 12: Creating Environments

Launch introduces Development environments which allows you to create as Development builds as needed. For example, the Analytics implementation team can work in their own development environment, while the Target implementation team can work in their own development environment. They can work independently and merge their code later in the publishing process.

i. Click "**Environments**" in the top navigation and then click "**Create New Environment**" under "**Development**" click "**Select**":



Development

Team members can use development environments to develop and test libraries, or sets of changes, created in Launch. Each environment has its own embed code, which can be deployed on any development sites. A company can have unlimited development environments.

Select

Staging

The staging environment is an optional environment where a team member with approval rights can test and approve a library before it is published. The staging environment embed code should be deployed on the official testing site. A company can only have one staging environment.

Current Environment: none

Select

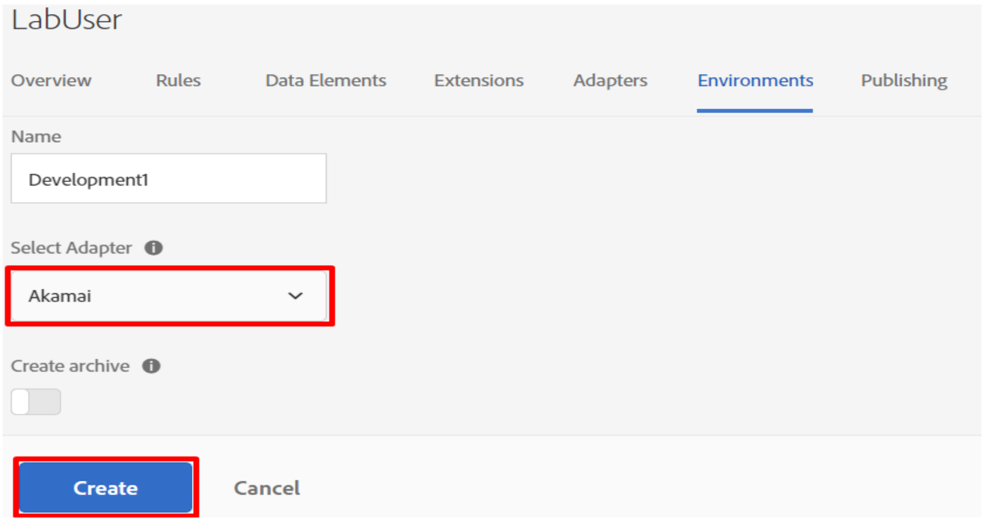
Production

The production environment is for the live, customer-facing site. The production environment has an embed code which should be deployed on the production site. A company can only have one production environment.

Current Environment: none

Select

ii. Name your environment "**Development 1**," select your "**Akamai**" adaptor, and then click "**Create**":

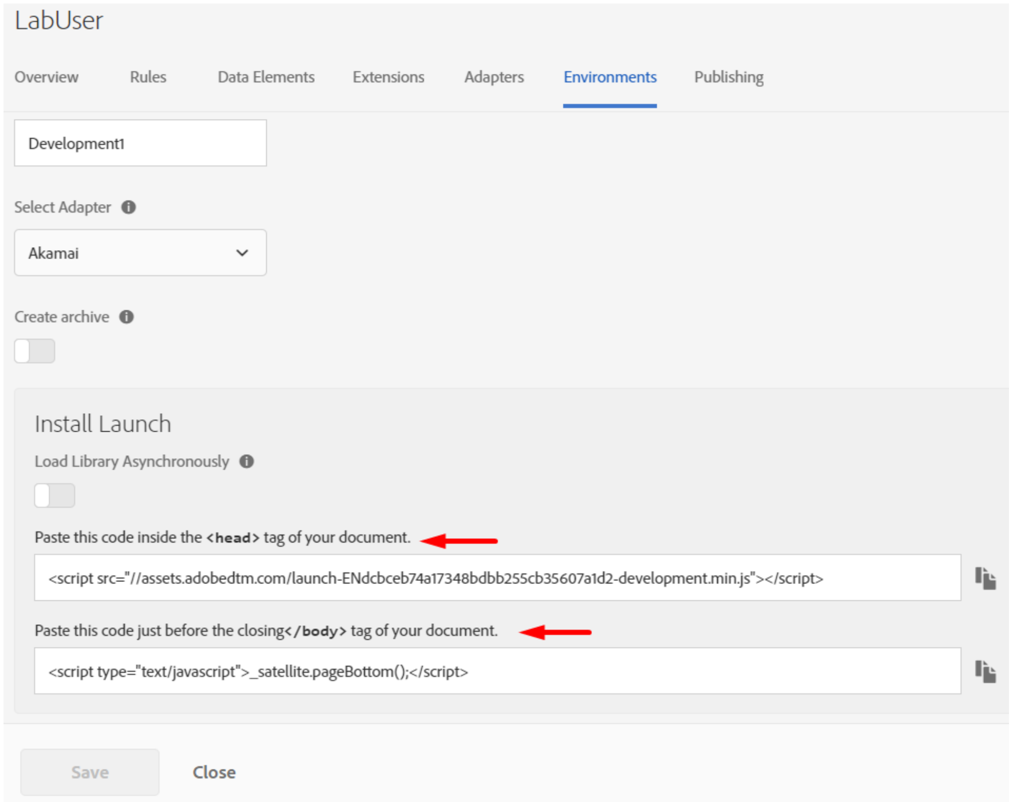


Name: Development1

Select Adapter: Akamai

Note: The Create archive option is a feature used for self-hosting, which we will not be using in this Exercise.

iii. Note that as soon as you select the Adapter and hit the create button, the embed code for that environment is created and displayed. Each environment will have its own unique embed code:


LabUser

Development1

Select Adapter: Akamai

Create archive

Install Launch

Load Library Asynchronously

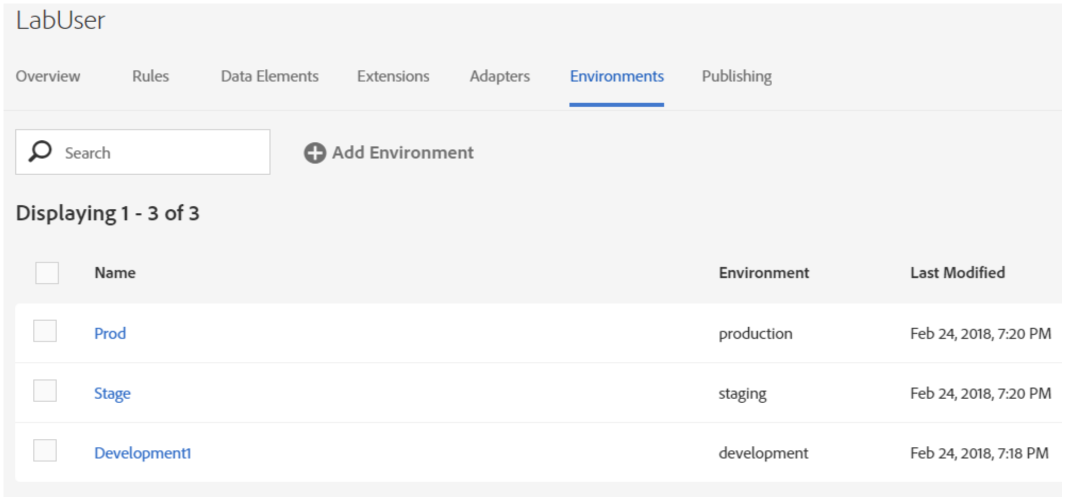
Paste this code inside the <head> tag of your document.

```
<script src="//assets.adobedtm.com/launch-ENdcbceb74a17348bdbb255cb35607a1d2-development.min.js"></script>
```

Paste this code just before the closing</body> tag of your document.

```
<script type="text/javascript">_satellite.pageBottom();</script>
```

iv. Click "Close".
v. Next, add a staging environment and a production environment (optional). However, for this exercise we only need a Development environment.
vi. Once you are done, you should have your Environments listed (staging and production environment are optional for this exercise):


LabUser

Displaying 1 - 3 of 3

Name	Environment	Last Modified
Prod	production	Feb 24, 2018, 7:20 PM
Stage	staging	Feb 24, 2018, 7:20 PM
Development1	development	Feb 24, 2018, 7:18 PM

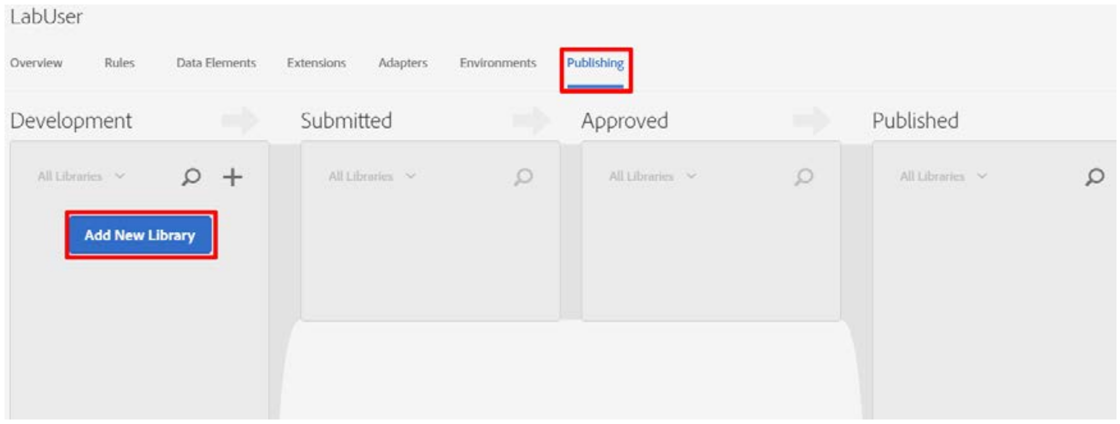
Note: You can obtain the asynchronous embed codes by using the toggle. The toggle selection is not saved in the UI. For this activity, we will not be using async code.

Task 14: Building Library

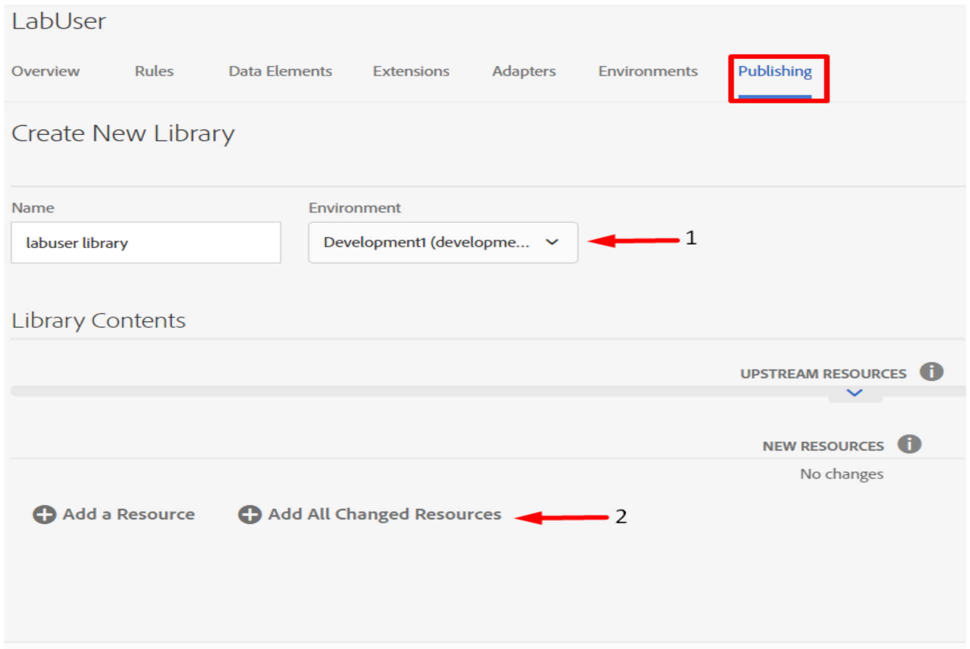
In Launch, each change must be intentionally added to a library and then that library must be "built" in order for the changes to be pushed to the environment.

These steps will guide you through adding library for Development environment:

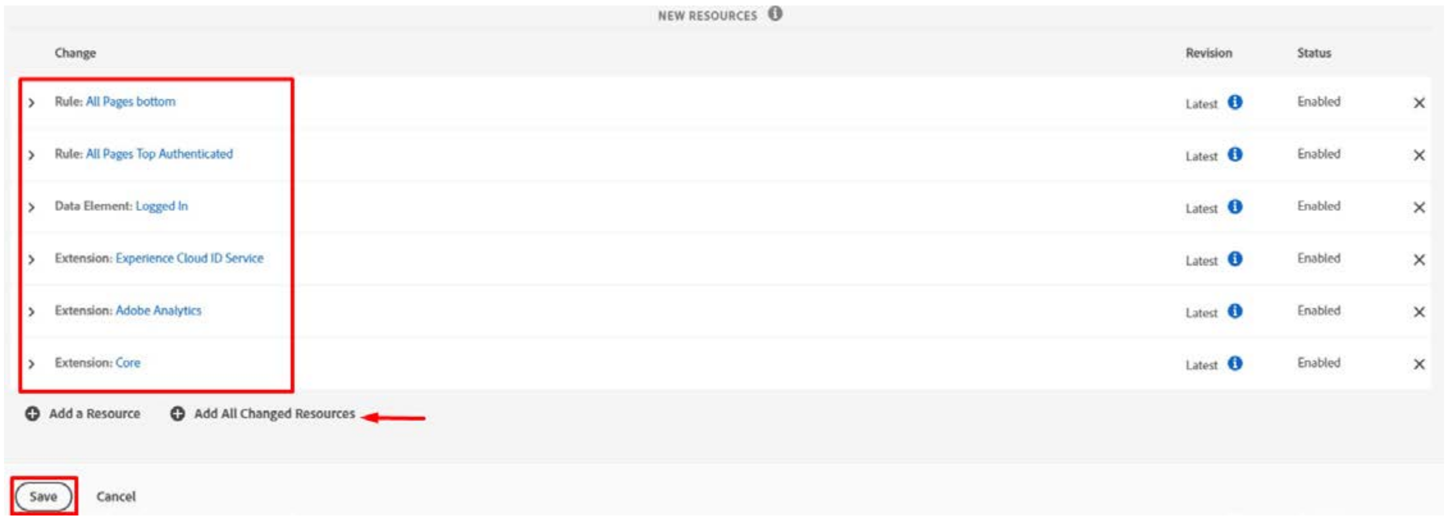
i. Go to the "**Publishing**" tab. Click the "**Add New Library**" button in the Development section



ii. Name your library (e.g. "labuser1 library"). Map it to your "Development 1" environment. Click "**Add All Changed Resources**", this will quickly add all your changes to the library.



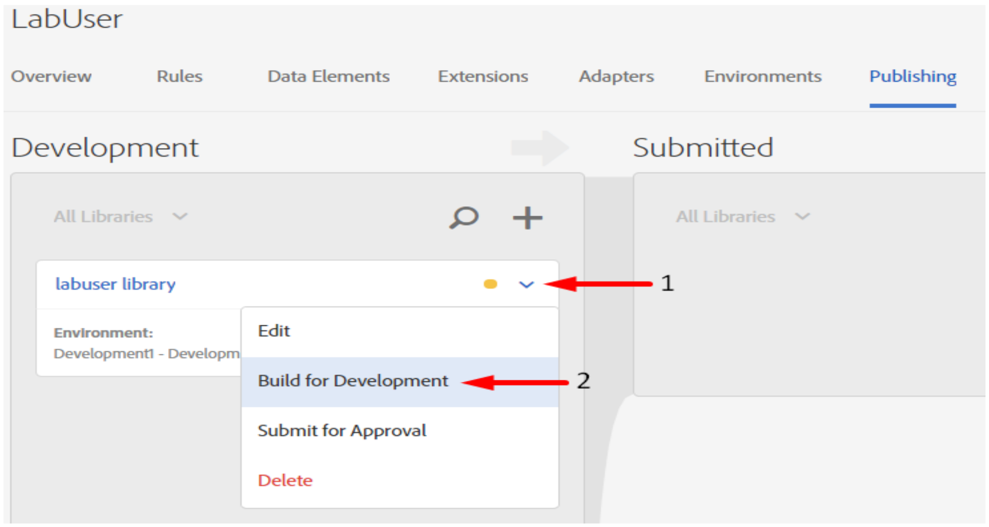
iii. In the screenshot below, note that all your Rules, Data Elements, and Extensions now appear as "Changes to be Built":



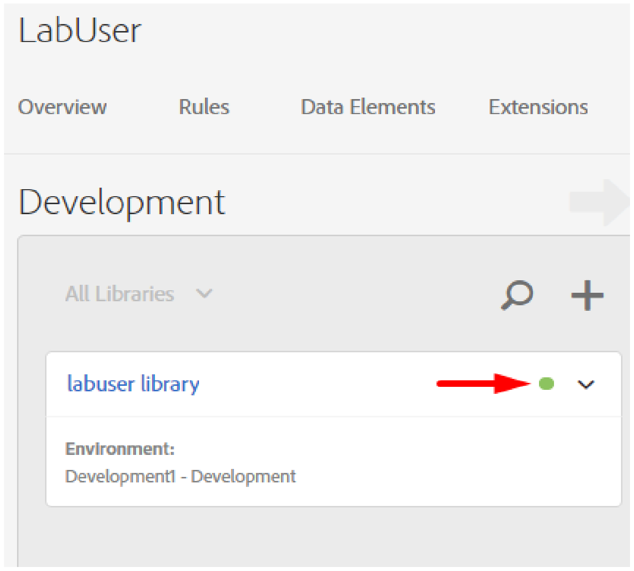
NEW RESOURCES

Change	Revision	Status
Rule: All Pages bottom	Latest	Enabled
Rule: All Pages Top Authenticated	Latest	Enabled
Data Element: Logged In	Latest	Enabled
Extension: Experience Cloud ID Service	Latest	Enabled
Extension: Adobe Analytics	Latest	Enabled
Extension: Core	Latest	Enabled

iv. Click "Save" to save the library with all the changes. The last remaining step is to build the library. The yellow dot indicates that the library hasn't been built. Select the "Build for Development" dropdown option:



v. Once the library has successfully built, you will see the green status icon. Launch will confirm that the library has been deployed to the environment before it changes the status icon to green.

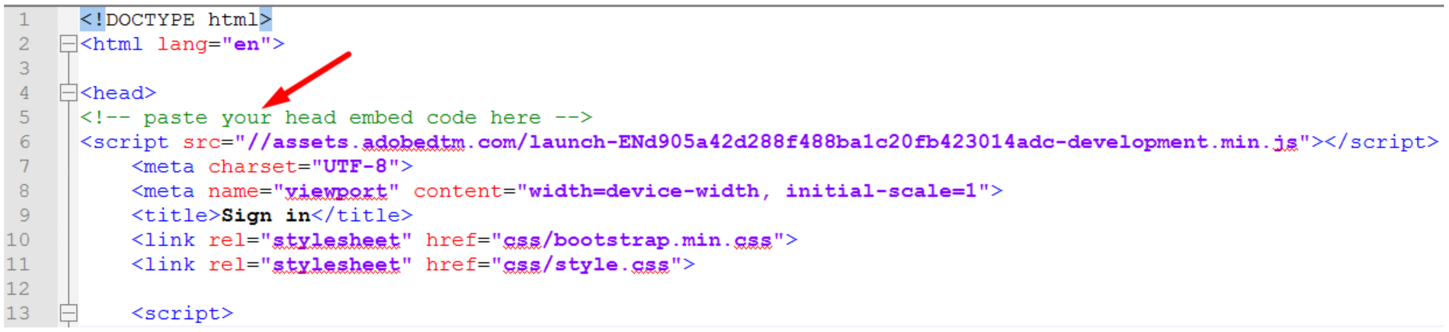


Task 13: Implementing Launch Embed Codes

Now we will be implementing the Launch Embed Codes in our sample site.
For each environment, there are two embed codes for synchronous deployments, the Header Embed Code and the Footer Embed Code. In this exercise, will be using Development environment's embed code:

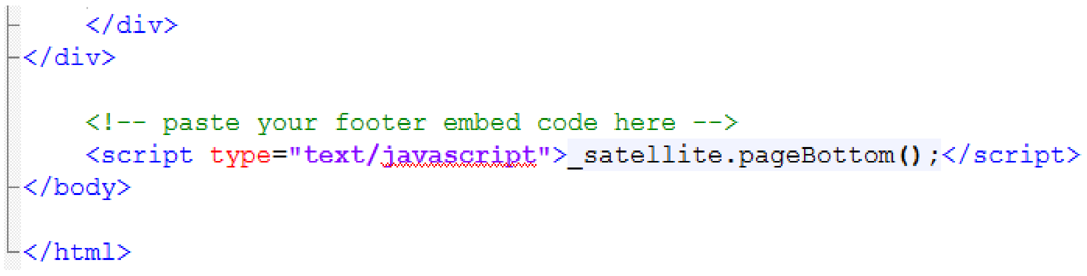
i. Header Embed Code: The Header Embed Code should be implemented in the <head> element of all HTML pages. Typically, in actual sites you have one or several template files which control the <head> globally across the site.

ii. Copy the header embed code and then go to this location on your Mac > Desktop > samplesite > open both the web pages one by one in Text Edit or Brackets, and add the head embed code just after opening of <head> element in HTML. This should be placed in both HTML files.



```
<!DOCTYPE html>
<html lang="en">

<head>
<!-- paste your head embed code here -->
<script src="//assets.adobedtm.com/launch-ENd905a42d288f488ba1c20fb423014adc-development.min.js"></script>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Sign in</title>
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <link rel="stylesheet" href="css/style.css">

    <script>
```

iii. The footer embed code remains same for all, so that is already placed for you. The footer embed code is placed just before the close of the </body> element.



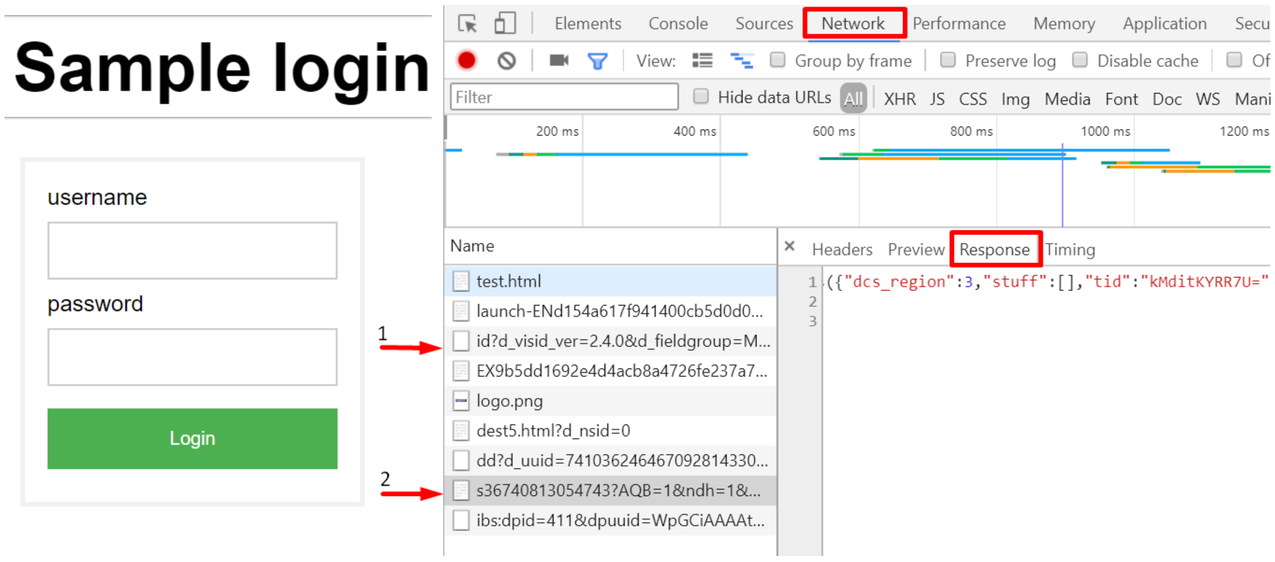
```
        </div>
</div>

    <!-- paste your footer embed code here -->
    <script type="text/javascript">_satellite.pageBottom();</script>
</body>

</html>
```

Task 14: Verify Adobe Analytics and Experience Cloud ID service call

Now, we'll test to see if our Launch property is loading and all the calls are firing as we expect. To do this, we'll open the site in Chrome and refresh. For this exercise, it is recommended that you open the web page in **New Incognito Window** of Chrome.

i. A new Incognito window can be launched by entering the keyboard shortcut cmd + shift + n key or by clicking File > New Incognito Window from the menu bar near the top of the screen. With developer tools opened in Chrome (Command + Option +J or View > Developer > Developer Tools), access the sample site: http://localhost:8080

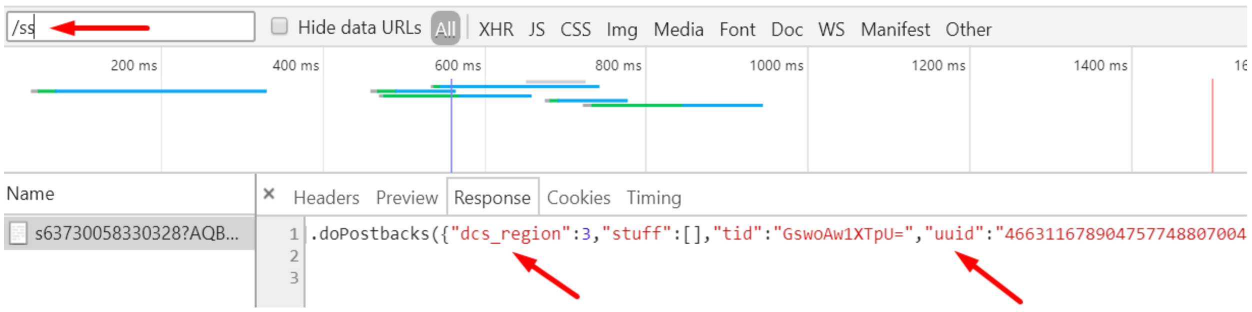
ii. Go to Network Tab in browser, to verify if network calls are being fired for Adobe Analytics and Experience cloud ID.



```
test.html
launch-ENd154a617f941400cb5d0d0...
id?d_visid_ver=2.4.0&d_fieldgroup=M...
EX9b5dd1692e4d4acb8a4726fe237a7...
logo.png
dest5.html?d_nsid=0
dd?d_uuid=741036246467092814330...
s36740813054743?AQB=1&ndh=1&...
ibs:dpid=411&dpuuid=WpGCiAAAAt...
```

```
({"dcs_region":3,"stuff":[],"tid":"kMditKYRR7U="
```

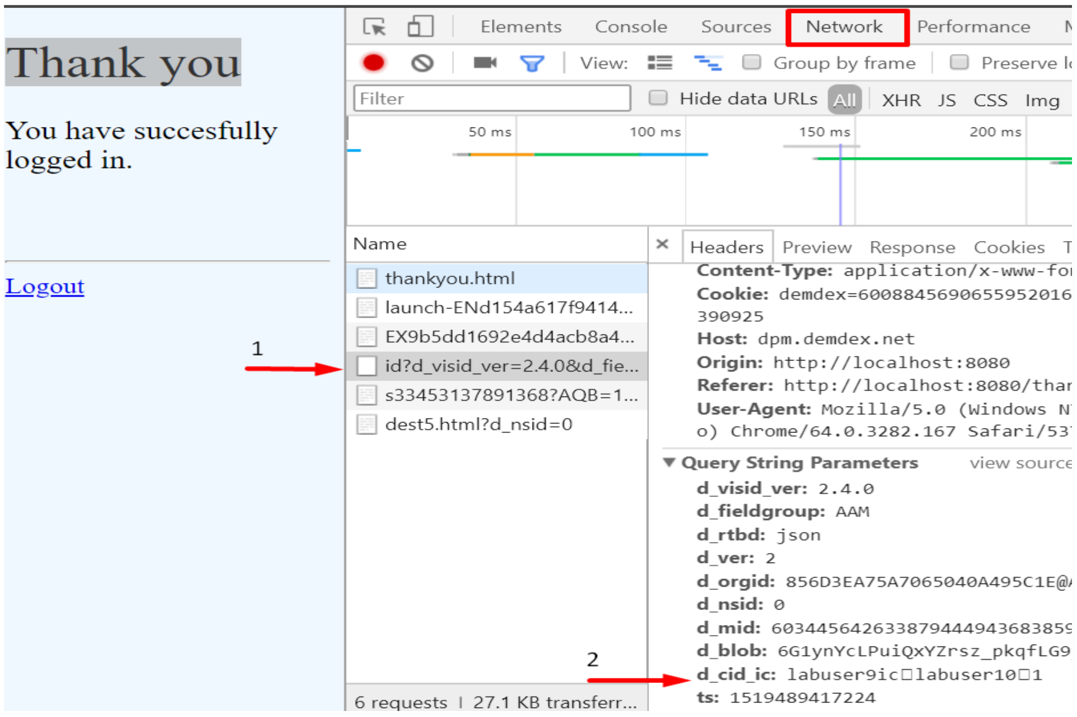
iii. Also, check the response of AA call, and see if it has a response with DCS region and UUID. That means SSF is working fine.



```
.doPostbacks({"dcs_region":3,"stuff":[],"tid":"GswoAw1XTpU=","uuid":"4663116789047577488O7004
```

iv. With developer tools still opened, sign in as authenticated user using the credentials for this site:
Username: labuser# (example: labuser1)
Password: same as username

v. After signing in, In the query parameters check the value of d_cid_ic. The value comprises a combination of the integration code and CRM ID.
This call will have a parameter: d_cid_ic == integration code%01Customer ID



```
Content-Type: application/x-www-for
Cookie: demdex=60088456906559520l6
390925
Host: dpm.demdex.net
Origin: http://localhost:8080
Referer: http://localhost:8080/thar
User-Agent: Mozilla/5.0 (Windows N
o) Chrome/64.0.3282.167 Safari/53

▼ Query String Parameters
  d_visid_ver: 2.4.0
  d_fieldgroup: AAM
  d_rtbd: json
  d_ver: 2
  d_orgid: 856D3EA75A7065040A495C1E@
  d_nsid: 0
  d_mid: 60344564263387944494368385
  d_blob: 6G1ynYcLPuiQxYZrsz_pkqfLG9
  d_cid_ic: labuser9ic□labuser10□1
  ts: 1519489417224
```

This completes our basic implementation for AA, AAM with SSF and Experience Cloud ID.

For More Hands On

Chapter 5: Onboarding offline data to Audience Manager via FTP/S3 (Optional)

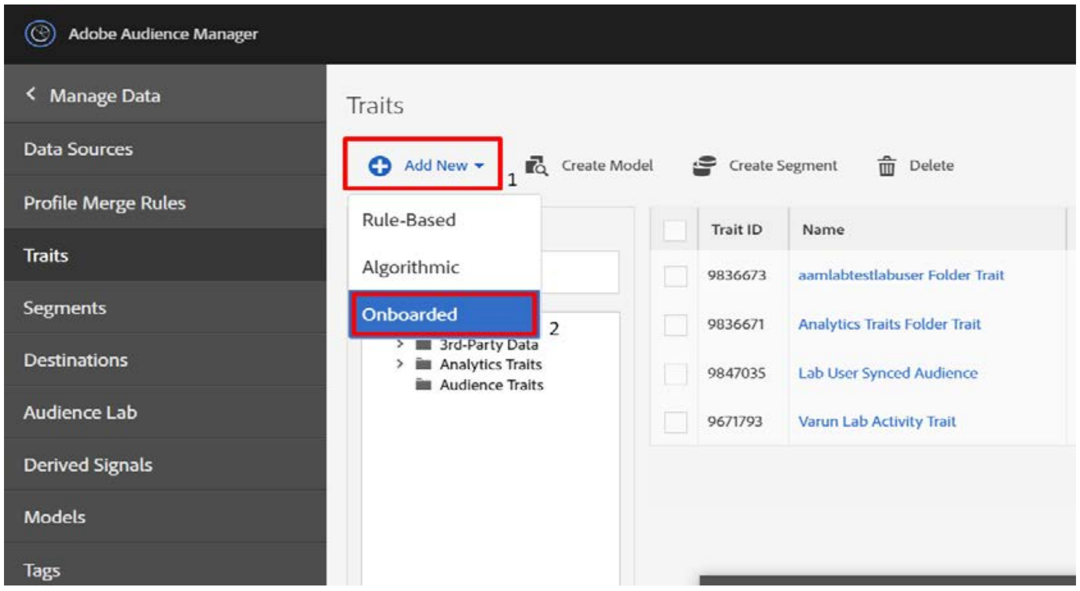
We often know even more about our customers than is available to us online. This data is most often stored "offline" in our CRM or database. Using AAM, we can now bring that data "online" in the form of traits. We then match those traits with our web data using the CRM or Customer IDs we set in the previous chapter.

Task 15: Creating onboarded traits

If we have some behavioral data available in our CRM for our customers, we can onboard this data and create some example traits.

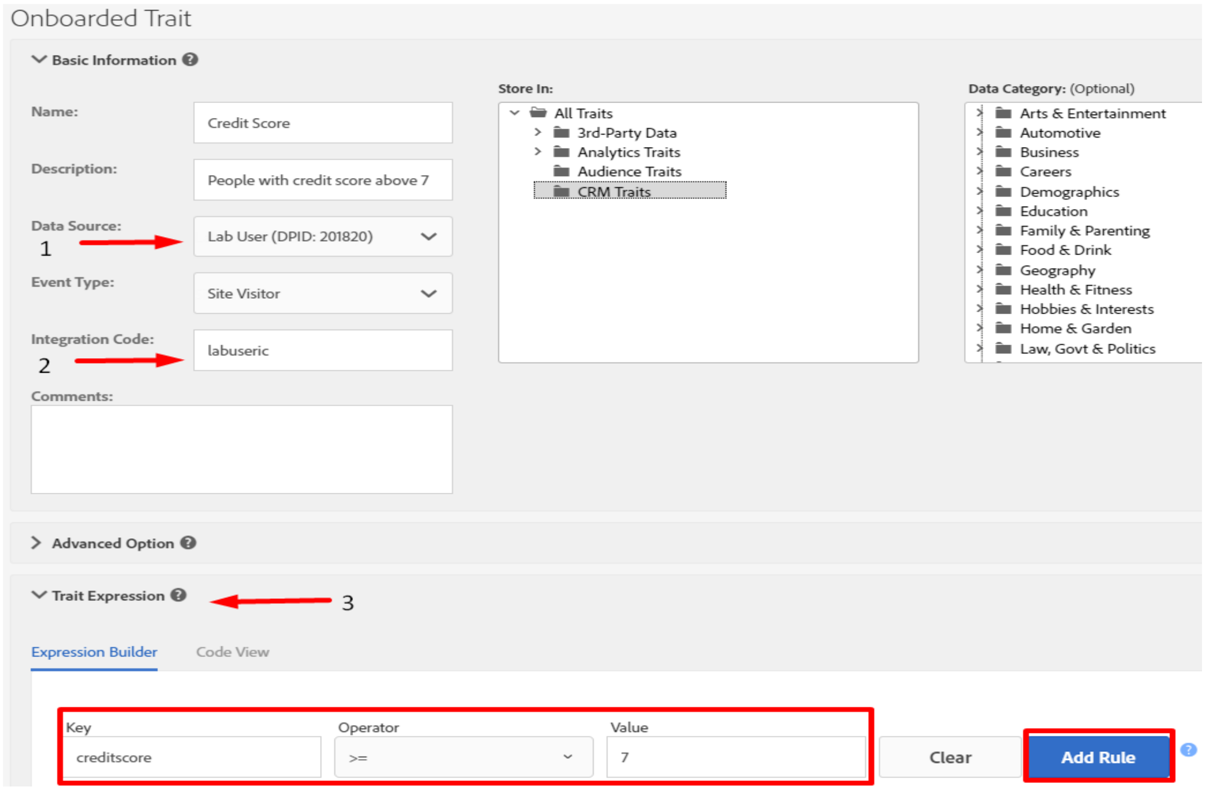
i. Go to Audience manager UI, from the application switcher at the top right corner. Then go to Manage Data → Traits.

ii. Then click on Add new → Onboarded and you will get options to define onboarded trait:



	Trait ID	Name
	9836673	aamlabtestlabuser Folder Trait
	9836671	Analytics Traits Folder Trait
	9847035	Lab User Synced Audience
	9671793	Varun Lab Activity Trait

iii. Create a trait named Credit Score, and we will qualify those users for this trait who have credit score above 7. So, the **Trait Expression** in Expression Builder is “creditscore” >= “7”. Click on **Add Rule** and Save.


Onboarded Trait

Basic Information

Name:	Credit Score
Description:	People with credit score above 7
Data Source:	Lab User (DPID: 201820)
Event Type:	Site Visitor
Integration Code:	labuseric
Comments:	

Store In: All Traits > 3rd-Party Data, Analytics Traits, Audience Traits, CRM Traits

Data Category: (Optional) Arts & Entertainment, Automotive, Business, Careers, Demographics, Education, Family & Parenting, Food & Drink, Geography, Health & Fitness, Hobbies & Interests, Home & Garden, Law, Govt & Politics

Advanced Option

Trait Expression

Expression Builder | Code View

Key	Operator	Value
creditscore	>=	7

iv. Similarly, we can create another trait, named Qualification, with trait expression: qualification contains graduation.

v. Expressions used in these traits will be used in our inbound file keyed off CRM IDs or Customer ID.

Task 16: Creating inbound file for data onboarding

i. Open Text Edit on your MAC, and enter CRM IDs (Customer ID that we grabbed to sync with device ID) that we have for this exercise (labuser1 to labuser95). You can add the same CRM ID that you have used to sign in to sample website.

ii. Add the row in this format: <CRMID><Tab>”key”=”value”,”key”=”value”
Example: labuser1 ”creditscore”=”8”,”qualification”=”graduation”

iii. Save this file on your desktop in with this file name: ftp_dpm<data source id>_<current unix time>.sync

iv. Now, this file is ready and can be uploaded to the FTP-in or s3 inbound server to be processed and onboarded. After few hours (may take up to 24 hours), we will be able to see this file’s onboarding status and these onboarded traits will have a population.

v. We can later create segments using real time and onboarded traits and use profile merge rules along with Recency and Frequency in our segments.

Chapter 6: Working with d_cts flag in DCS API call (Optional)

We will be verifying that whether CRM ID or Device ID is qualified for intended traits and segments using DCS APIs.

Task 17: Making the API call

i. We will use a pair of Data Source Integration Codes (DSIC) and the CRM ID to check whether it is qualified for the intended traits and segments.

ii. The demdex subdomain for our account is: aamlab.demdex.net, we can add following DCS parameters:

   d_cid_ic=<DSIC>%01<CRMID> → This is the data source integration code and CRM ID pair.

   d_rtbd=json → This required to get a JSON response from the DCS.

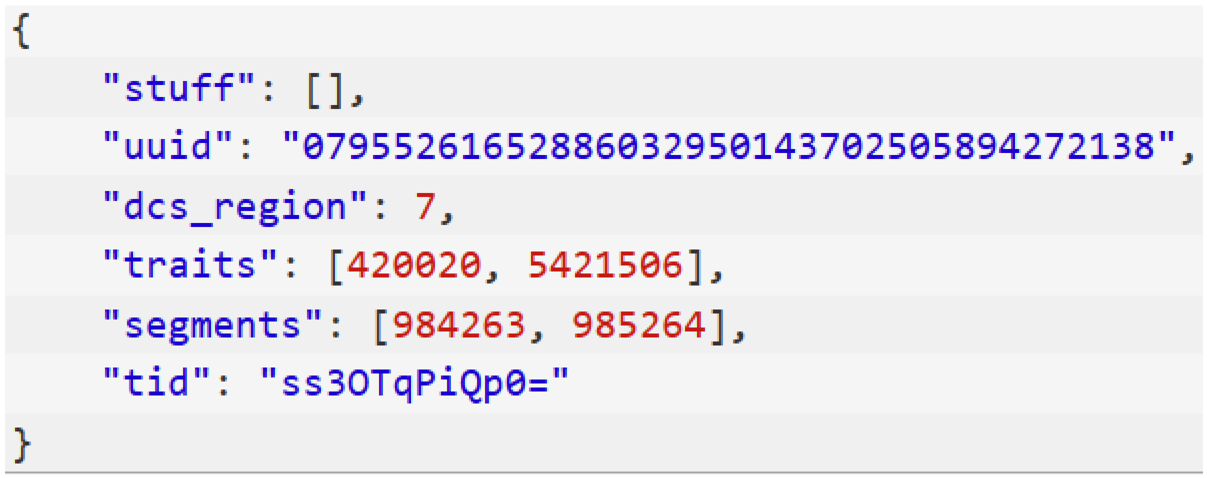
   d_cts=2 → Returns segment IDs for the segments

iii. Now our call should be like this:

   aamlab.demdex.net/event?d_cid_ic=<DSIC>%01<CRMID>& d_cts=2&d_rtbd=json

   Example DCS API call:
   http://aamlab.demdex.net/event?d_cid_ic=labuseric%01labuser10&%20d_cts=2&d_rtbd=json

iv. Now we can make this call from our browser and get the JSON response from DCS. A sample response could look like the one below:



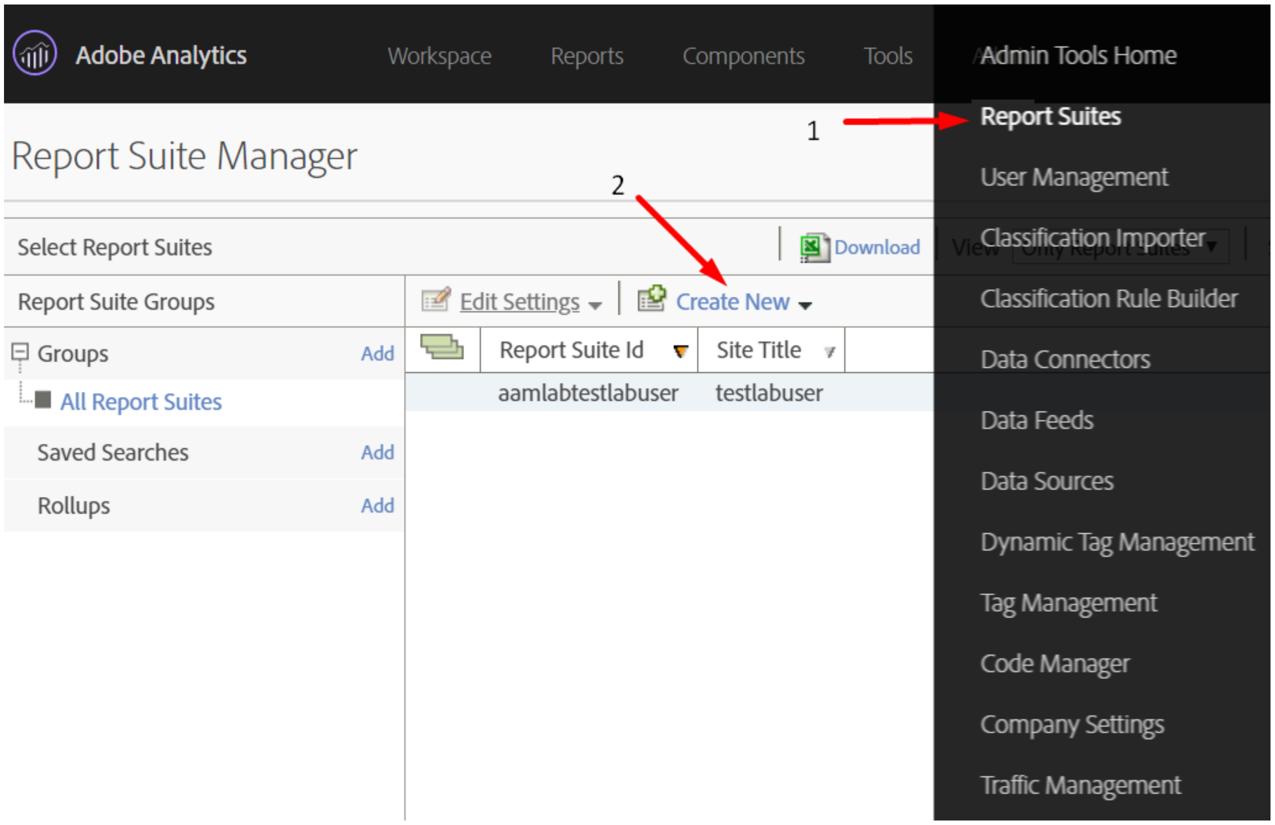
```
{
    "stuff": [],
    "uuid": "07955261652886032950143702505894272138",
    "dcs_region": 7,
    "traits": [420020, 5421506],
    "segments": [984263, 985264],
    "tid": "ss3OTqPiQp0="
}
```

This is end of your lab. Thank you very much for your attention.

Addendum

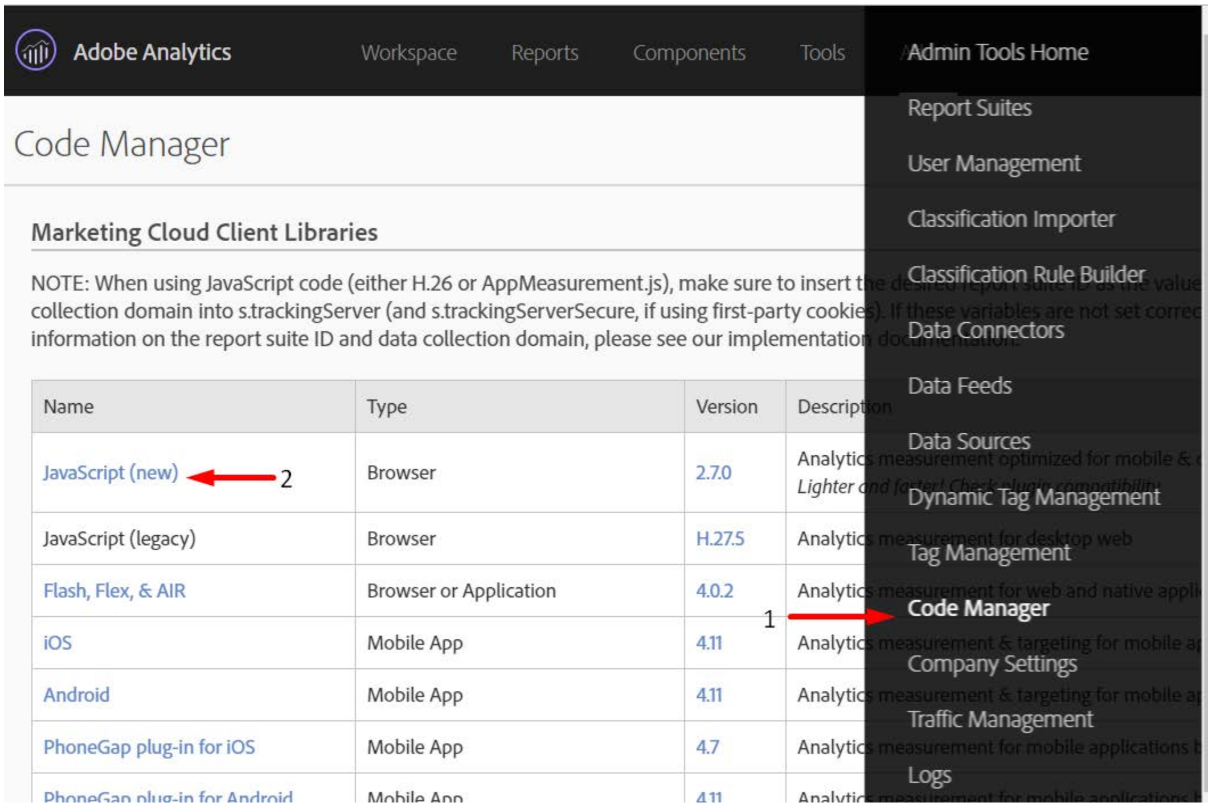
1: Creating Report Suite in Adobe Analytics UI

i. Access Adobe Analytics UI after signing into Experience Cloud.
ii. In Adobe Analytics, go to **Admin → Report Suites**. Then click on **Create New → Report Suite**.
iii. Fill in the required fields in **New Report Suite**. Click **Create Report Suite.**



2: Downloading Appmeasurement and Audience Management Module Library code

i. In Adobe Analytics UI, click on **Admin → Code Manager**.
ii. Then click on top option **Javascript (new)**



Code Manager

Marketing Cloud Client Libraries

NOTE: When using JavaScript code (either H.26 or AppMeasurement.js), make sure to insert the desired report suite ID as the value collection domain into s.trackingServer (and s.trackingServerSecure, if using first-party cookies). If these variables are not set correc information on the report suite ID and data collection domain, please see our implementation documentation.

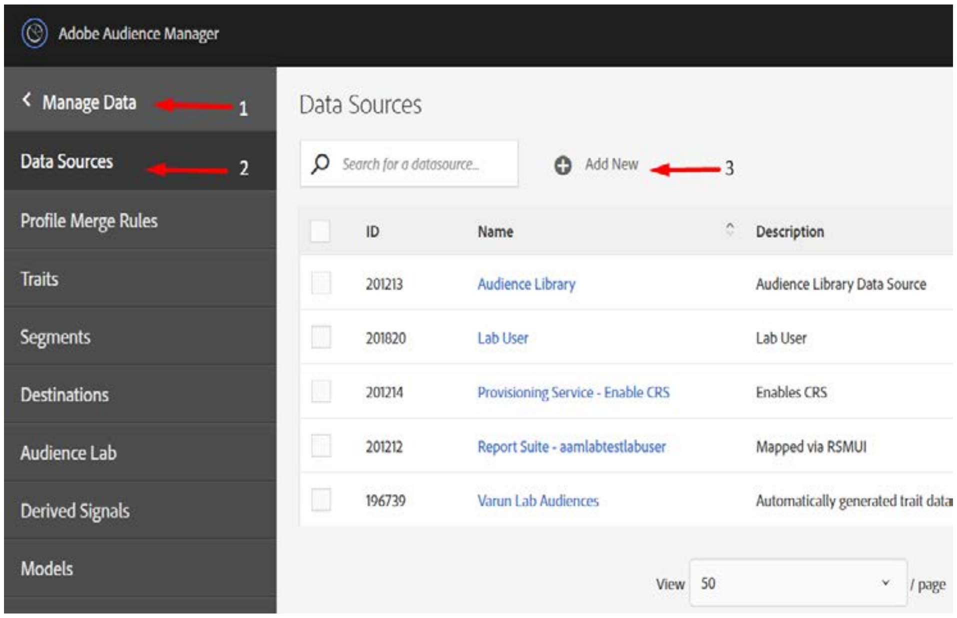
Name	Type	Version	Description
JavaScript (new)	Browser	2.7.0	Analytics measurement optimized for mobile & *Lighter and faster! Check plugin compatibility*
JavaScript (legacy)	Browser	H.27.5	Analytics measurement for desktop web
Flash, Flex, & AIR	Browser or Application	4.0.2	Analytics measurement for web and native appli
iOS	Mobile App	4.11	Analytics measurement & targeting for mobile a
Android	Mobile App	4.11	Analytics measurement & targeting for mobile a
PhoneGap plug-in for iOS	Mobile App	4.7	Analytics measurement for mobile applications b
PhoneGap plug-in for Android	Mobile App	4.11	Analytics measurement for mobile applications b

iii. This will download a zip file with the name: AppMeasurement_JavaScript-2.7.0.zip
iv. We can unzip this file and merge **AppMeasurement.js** and **AppMeasurement_Module_AudienceManagement.js** to form a single JS file.
v. For this exercise, the final Appmeasurement.js file is already present on your MAC. Check the Assets folder on your desktop.

3: Creating Data Source in Audience Manager

Before we add a rule to collect Declared ID in Visitor ID calls, we should have a Data Source defined in AAM, that should be collecting these declared IDs. Follow the steps to create a Data Source in AAM UI:

i. Click on the Experience Cloud application switcher and select Audience Manager. Alternatively, you can use this URL to sign into Audience Manager: http://bank.demdex.com
ii. Once you are logged in, click on Manage Data → Data Sources → Click on + sign to Add New Data Source:

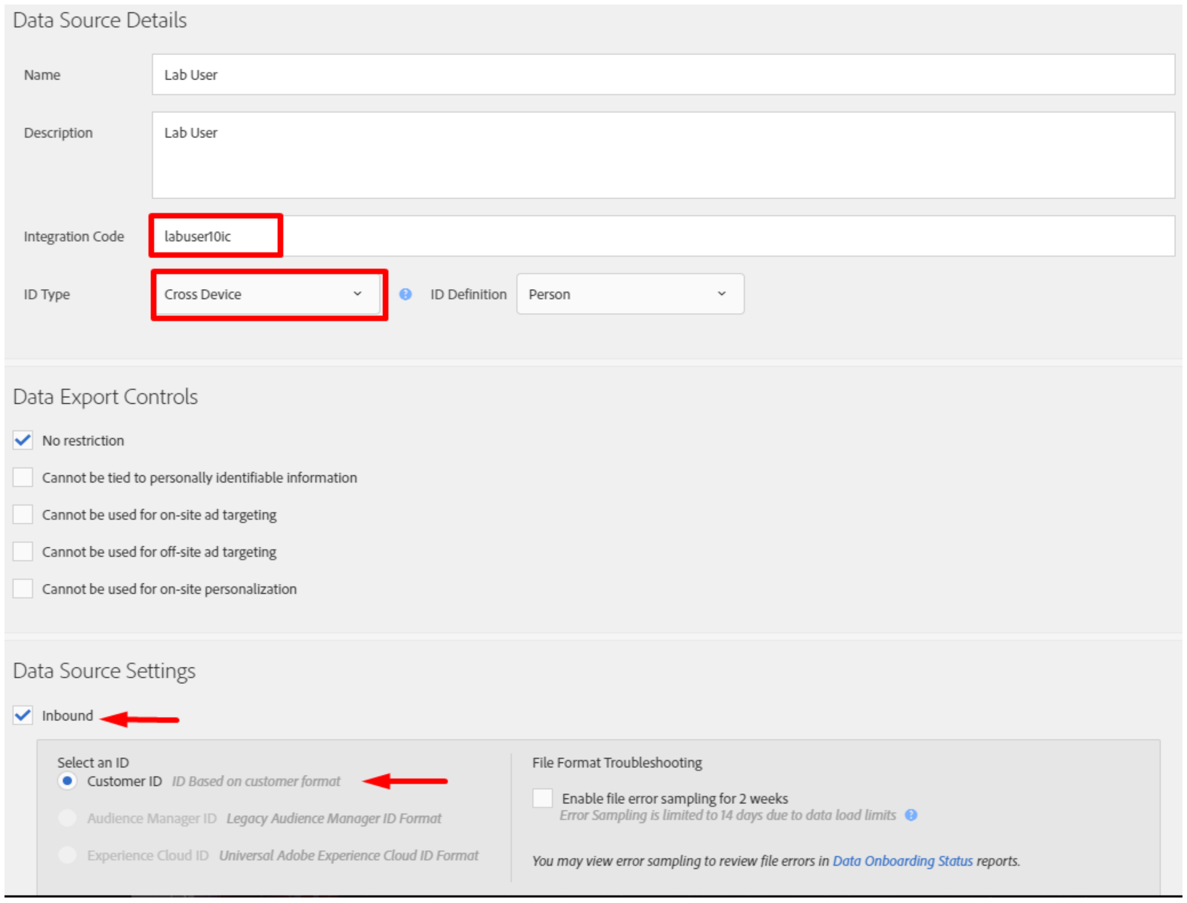


ID	Name	Description
201213	Audience Library	Audience Library Data Source
201820	Lab User	Lab User
201214	Provisioning Service - Enable CRS	Enables CRS
201212	Report Suite - aamlabtestlabuser	Mapped via RSMUI
196739	Varun Lab Audiences	Automatically generated trait data

iii. Then add the details for this Data Source: Name and Description. You can use same name as your seat ID, or Lab User#.

iv. Give an **integration code** and make a note of this, as this needs to be used in Customer ID sync implementation.
You can use labuser#ic (example: labuser10ic)

v. Select ID type from the drop-down list as **Cross Device**. Under Data Source Settings, check **Inbound** and select **Customer ID**:



Data Source Details

Name	Lab User
Description	Lab User
Integration Code	labuser10ic
ID Type	Cross Device
ID Definition	Person

Data Export Controls

- [✓] No restriction
- [ ] Cannot be tied to personally identifiable information
- [ ] Cannot be used for on-site ad targeting
- [ ] Cannot be used for off-site ad targeting
- [ ] Cannot be used for on-site personalization

Data Source Settings

- [✓] Inbound

Select an ID
- ● Customer ID *ID Based on customer format*
- ○ Audience Manager ID *Legacy Audience Manager ID Format*
- ○ Experience Cloud ID *Universal Adobe Experience Cloud ID Format*

File Format Troubleshooting
- [ ] Enable file error sampling for 2 weeks
  Error Sampling is limited to 14 days due to data load limits

You may view error sampling to review file errors in Data Onboarding Status reports.

Links and Resources:

- Adobe Audience Manager standard documentation: https://marketing.adobe.com/resources/help/en_US/aam/index.html
- To continue the conversation and ask questions, visit our forums: https://adobe.com/go/aamcommunity
- You can also use Twitter by following: @AdobeExpCare