SessionM Integration

Version 19

1 Contents

1. SUMMARY

This document provides technical overview and implementation details for each SessionM service integrated within SFCC platform with SiteGenesis Controller version. The SessionM cartridges (int_ int_sessionm_core, int_sessionm_controllers) extends the functionality of Commerce Cloud Storefront, enabling synchronous and asynchronous access to SessionM Loyalty Customers with services listed below.

Customer Profile SyncUp

- SessionM New Accout Creation
- SessionM Customer profile update
- SessionM Customer Profile Sync
- SessionM Customer Profile Search

SessionM Reward Redemption

- SessionM Reward Store Integration
- SessionM Reward Point Redemption

SessionM Export Feeds

- Order Export
- Master Catalog Export
- Navigation Catalog Export
- Inventory Export
- Realtime Syncup Failed Customers Export

SessionM Import Feeds

- Campaign and Promotion Import
- Coupon Import Import

SessionM Activity Timeline

- Show Activity timeline of order purchased, order confirmed, offer used, tier changed

SessionM Reward Store

- Get available and active offers from SessionM
- Click to Purchase Offers from SessionM
- Use offers to purchase products

SessionM Cleanup Tiers

- Clean up the tier stages that if user on the top tier don't show unnecessary details.

2. COMPONENT OVERVIEW

2.1 FUNCTIONAL OVERVIEW

2.1.1 SessionM Profile Actions

This section provides an overview for Customer Profile Data SyncUP between Salesforce commerce cloud and SessionM using SessionM customer API's.
SessionM provides customer Data profile sync up for the following use case scenarios

2.1.1.1 SessionM New Account Creation

SFCC create account feature on storefront to be leveraged to integrate with Push new account create data to SM for user creation. Cartridge will be making SM POST "Users" service API call to POST the customer profile data and create account with SM. After user successfully submit the create account form on storefront, first SFCC account will be created and on successful account creation SM "users" service POST API call will be made to create profile with SM. SM service call will happen in background and in case of success/failure no user message will be displayed. In case of service call failure, cartridge will make retry to a defined number of times and then after it will send a notification email to a merchant defined email ID with a raw message in email body.

2.1.1.2 SessionM Customer profile update

After successful Sign In/Sign Up to SFCC account and moving to account section, customer is able to see their pre populated details on account form. This feature enables the customer to update their existing details by providing the details on account from with new customer data.

2.1.1.3 SessionM Customer Profile Sync

Once the customer is Sign In to SFCC account, this feature enables the customer to retrieve an existing customer profile with all the profile's associated attribute

2.1.1.4 SessionM Customer Profile Search

Search API service to be called during SFCC sign in process, where SM ID is not available into SFCC customer profile. After successful authentication invoke the search call for external ID and sync up the records.

2.1.2 Scheduler Jobs

The SessionM cartridge provides support to a batch jobs to multiple scenarios as given in section 2.1.1. These jobs are defined for specific time interval and are configurable through Business Manager as per merchant need.

2.2 USE CASE

S.No	Module	HLS Description
3.	Create Profile	Verify that the guest user can Create customer at SessionM when user registers at SFCC storefront from Header/My Account section
4.	Create Profile	Verify that the guest user can Create customer at SessionM when user is in Checkout page
5.	Create Profile	Verify that the Anonymous user completes Checkout flow and opted for Account Creation in Thank You page then can Create customer at SessionM when user registers at SFCC storefront.
6.	Create Profile	Verify the Auth token was fetched from SM and Stored in SFCC when Customer was created
7.	Create Profile	Verify that If real time create user fails, re-try should be attempted and failure to be logged and a flag should be marked against the record into SFCC and email notification to be sent to SessionM for same failure
8.	Create Profile	When 'SessionM SessionM Loyalty Opt in' feature is enabled verify that if user selects to Opt in/Opt out of loyalty system of SessionM it is reflected in the sessionM. By default user is opted into loyalty system.
9.	Sign In	Verify the Auth token was fetched from SM and Stored in SFCC when Customer Logs in
10.	Update Profile	Verify that when customer updates Profile data at SFCC, same should be updated for customer profile data at SessionM.
11.	Update Profile	Verify that when customer updates/edit address data at SFCC, same should be updated for customer address data at SessionM.
12.	Update Profile	Verify that when customer creates multiple address data at SFCC, same should be updated for customer address data at SessionM.
13.	Update Profile	Verify that when customer deletes address data at SFCC, same should be updated for customer address data at SessionM.
14.	Update Profile	Verify the preferred address, if customer makes an address as preferred
15.	Update Profile	Verify that If real time update user fails, re-try should be attempted and failure to be logged and an email notification to be sent to SessionM for same failure
16.	Update Profile	Verify the customer data from SessionM for already synced-up customer during login at SFCC storefront and update the customer data at SFCC and customer reward points
17.	Update Profile	Verify the customer data from SessionM for already synced-up customer during login on Checkout process at SFCC storefront and update the customer data at SFCC and customer reward points
18.	Update Profile	Verify the Flag "Is SM Profile Sync Failed" is marked as true if SM is down (for already synced-up customer during login on Checkout process at SFCC storefront)

19.	Update Profile	When 'SessionM SessionM Loyalty Opt in' feature is enabled verify that if user selects to Opt in/Opt out of loyalty system of SessionM it is reflected in the SessionM.
20.	Search Profile	Verify the SFCC already registered customer whose making first time sync-up call with SessionM system by loging in. SessionM will search for user in SessionM system-based om email id.
21.	Search Profile	Verify the SFCC already registered customer whose making first time sync-up call with SessionM system by login in. SessionM will search for user in SessionM system-based om email id. If the searched email Id does not exist, then update a Flag User not Synced in SFCC
22.	Sync Profile	Verify the Batch Job to Pull data from SessionM for realtime failure records only (Registered and synced Customer data in both the system)
23.	Batch Job for Record Sync	Verify Batch Job Success/Failure notification to Batch User
24.	Rewards Store	Verify the "Rewards Store" link is present in "My Account" section (left panel)
25.	Rewards Store	Verify the "Rewards Store" link is present in "My Account" header section
26.	Rewards Store	Verify the Rewards page is opening after clicking on Rewards link
27.	Rewards Store	Verify Rewards page is redirected to error page after cliking on Rewards link, if Auth token is not generated OR due to any other error
28.	Rewards Store	Verify the BM page has an option to turn on/off SM Rewards Page
29.	Inventory Feed	Verify the Inventory Job by running or scheduling
30.	Inventory Feed	Verify that email is sent to the configured email address with Job status
31.	Catalog Feed	Verify the Catalog Job by running or scheduling
32.	Catalog Feed	Verify that email is sent to the configured email address with Job status
33.	Promotion Coupon	Verify that the SessionM Coupon/Promotion/Coupon feed is imported from the SFTP location to BM
34.	Promotion Coupon	Verify the Promotion/Campaign/Coupon in BM.
35.	Promotion Coupon	Run the batch job with no feed file in the FTP folder.
36.	Promotion Coupon	Run the batch job with a feed file which is not in the agreed format (format to be defined)
37.	Promotion Coupon	Run the batch job with incorrect username and password in the Transfer from FTP component from BM.

38.	Promotion Coupon	Verify that email is sent to the configured email address with Job status
39.	Promotion Coupon	Verify that SM Imported Promotion/Campaign is applied during Checkout / Order placement journey
40.	Order Export	Verify the configuration for Order Export - via API Call OR via Feed Job
41.	Order Export	Verify that SFCC Orders to SessionM via API Call
42.	Order Export	Every time when the status of the order changes we send the offer to SessionM that the Status of the order is changed
43.	Order Export	Check if the Order is sent to SessionM and has offer attached to it then notify that Offer with details to SessionM to make the Offer as redeemed.
44.	Order Export	Verify that email is sent to the configured email address with Job status
45.	Order Export	Verify that SFCC Orders to SessionM via SFTP
46.	Order Export	Verify that email is sent to the configured email address with Job status
47.	Order Export	Verify the configuration for Payment mode is masked OR full encrypted
48.	Order Export	Verify the Order XML for Payment mode
49.	Order Export	Verify the Order XML for SM provided Coupon
50.	Order Export	Verify the Order XML for SM Sales Associate ID
51.	Redemption	Verify registered user is able to redeem the rewards points during checkout
52.	Redemption	Verify on billing page the rewards points section is displayed
53.	Redemption	Verify the redeemable points value cannot be greater than %age of Order total (%age is defined in BM)
54.	Redemption	Verify the redemption information is shared with SM and the customer point balance is updated
55.	Redemption	Verify if Redemption order API service cal fails, make retry if still fails then mark that order for JOb Processing. User will be see thank you page. even Order API call failed, we need to update the balance into SFCC.
56.	Redemption	Verify the Job should pick the failed orders and make order API call for these orders and update the points balance into SFCC and order custom attributes with order API response data
57.	Redemption	Verify the Custom Preference "Redemption limit in percentage" in BM

58.	Redemption	Verify the Custom Preference "Value of 1 SessionM point" in BM
59.	Retry Create Profiles	Verify the job is running if and as scheduled
60.	Tier	Verify the Tier status of the customer and Tier Display
61.	Tier	Verify Tier should be updated dynamically on the basis of order placement using Redemption points
62.	Activity Timeline	Verify the User and Show last 500 Activities User has done.
63.	Activity Timeline	Set pagination and number of records in the preferences to see data in the Activity table page.
64.	Sales Associate ID	Verify that Sales Associate ID field is present In Business Manager for each order created.
65.	Sales Associate ID	Verify that if added on storefront, when a user makes an order, an option to add Sales Associate ID is present on checkout page.
66.	Sales Associate ID	Verify that in Business Manager Custom Preferences > SessionM, Sales Associate Field Name can be modified.
67.	Reward Store	Verify and get all the available offers related to that user which are active and are available from SessionM
68.	Reward Store	Get user functionality to purchase the available offers from SessionM using their available reward points

68.1 LIMITATIONS, CONSTRAINTS

This section details the limitations, constraints and the best practice for merchants planning to integrate the SessionM cartridge with Demandware Storefront.

S.No	Description
1	Demandware does not have capability to validate customer sign up email belong with him/her. SessionM need to apply validation checks at their end to prevent any possible frauds as if a customer creates an SFCC account with an email which is not owned by him/her
2	There is a possibility of the rewards point in SFCC not be in sync with SessionM, if profile sync during login fails or if the order redemption service fails during checkout. To minimize the misuse of the redemption points we should run the Order Redemption feed Job on very frequent manner as low as possible. But still there will a timeframe between Redemptions Service failure at checkout and Order Redemption Job Ran, during the period points will not be in Sync
3	Merchant need to whitelist the IP address into SessionM for the integration and

	Production environments

68.2 ASSUMPTIONS

This section details the assumptions while integration SessionM cartridge.

S.No	Description
1.	Existing customer Data Sync up between SFCC and SM has to be taken care offline by Merchants
2.	SFCC will be sending only the customer provided user Profile /Address data to SessionM
3.	No Batch Job to Push customer updates on Real time failure
4.	SFCC provided out of the box Customer Profile fields only to be considered for sync up
5.	SFCC Customer no. will be used to as external ID into SM, SM need to use the same during legacy data migration
6.	In case of multi-channel update (SFCC, Kiosk etc), SFCC customer ID is considered as external_id and other channel customer ids are considered as proxy ids at SessionM side.
7.	The order rewards redemption job would be executed to a very low frequently as possible, so that the possibility of reward points in SFCC not being in sync with SessionM is minimal to avoid misuse of points
8.	If there is a case where a customer gets a discount through SessionM rewards redemption that he is not eligible for due to SFCC rewards points not being in sync with SessionM. Then the case needs to be handled by the merchant manually.
9.	In COPlaceOrder.JS or COPlaceOrder.XML - Order Redemption API call need to be made before Placing the Order. Because in case Order Redemption API throws error, then system will not allow to Fail Order, Because Order with created state only can be Failed.
10.	Look and feel of Rewards Store page will be managed by Merchants/SessionM.
11.	Merchants/ SessionM will work together to match the Reward store style guide with storefront
12.	UI changes and form validation has to be handled by Merchant only, cartridge only support basic SFCC out of the box form validations
13.	Auth Token's expiry is managed by SM and it is greater or equal to SFCC storefront session

68.3 COMPATIBILITY

This cartridge is integrated and tested with site genesis code base 17.4 and compatibility mode 16.2 of Demandware.

69. IMPLEMENTATION GUIDE

69.1 SETUP

This section describes the Controller structure and name of files in SessionM cartridges.

3.1.1 Cartridge Structure

int_sessionm_core:

- package.json – initialize hook or service

cartridge/scripts:

- adaptor/ CustomerProfile.js
- adaptor / Rewards.js
- common/ CommonConstants.js
- facade/ CustomerProfileFacade.js
- facade / RewardsFacade.js
- helper / SMHelper.js
- jobs / DownloadFilesFromSFTP.js
- jobs / OrderExportFeed.js
- jobs / RedemptionOrderExport.js
- jobs / RetrieveMultipleProfilesJob.ds
- jobs / SMFetchTierDataJob.js
- jobs / UploadFilesToSFTP.js
- jobs/RetryCreateProfiles.js
- pipelets/ AppendTimeStamp.ds
- pipelets / GetCatalogExportDetail.ds
- pipelets / GetInventoryExportDetail.ds
- pipelets/ RedeemRewards.ds
- pipelets/ RegisterCustomer.ds
- pipelets/ RemoveRewardPoints.ds
- pipelets/ RemoveRewardsRedemption.ds
- pipelets/ RetrieveCustomer.ds
- pipelets/ RewardsPoints.ds
- pipelets/ UpdateCustomer.ds
- services/init/ ServiceInit.js
- static/default/css/ sessionm.css

Templates:

- default/ account / tierdetails.isml
- default/checkout/ redeemablepoints.isml
- default/mail/ notification.isml

- default/rewards/ rewards.isml

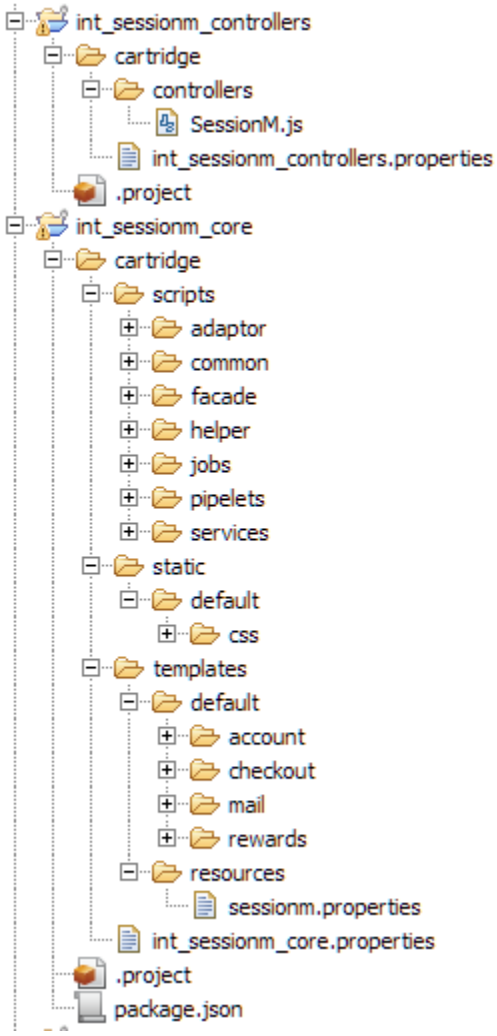
Properties:

- resources/ sessionm.properties

int_sessionm_controllers :

- SessionM .js

Please refer the screen shot below:



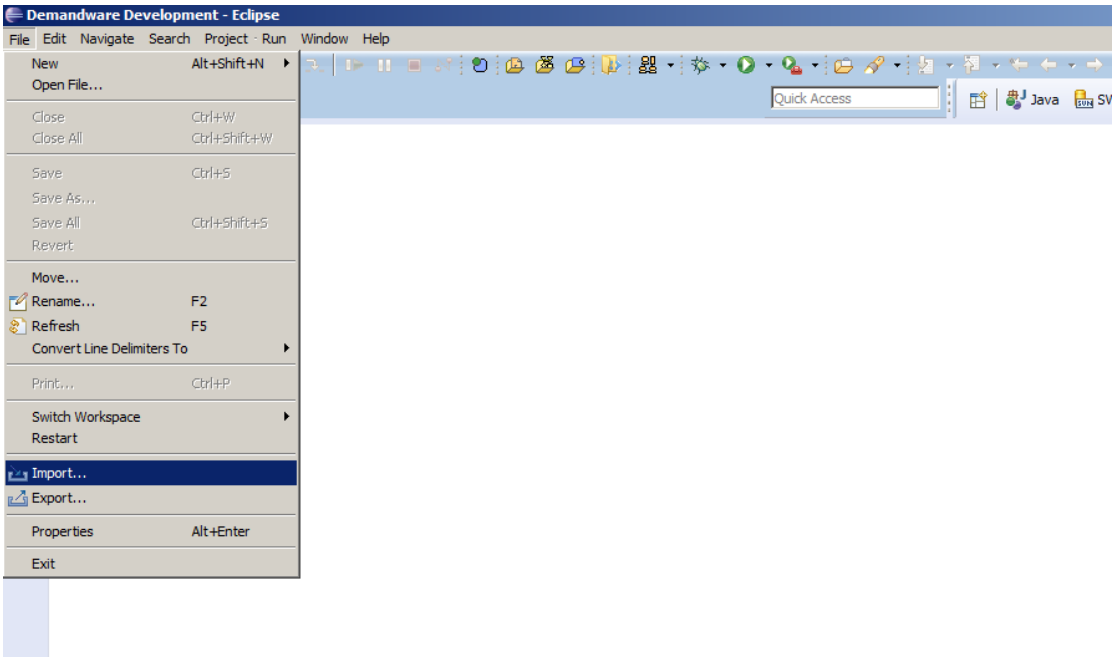
69.2 INSTALLATION

3.2.1 Adding the Cartridges in Demandware Studio

To upload the cartridges into the SFCC server you first need to add the cartridges into Demandware studio. In order to do this, follow these instructions:

1. In Demandware UXStudio select in the menu *File → Import*.

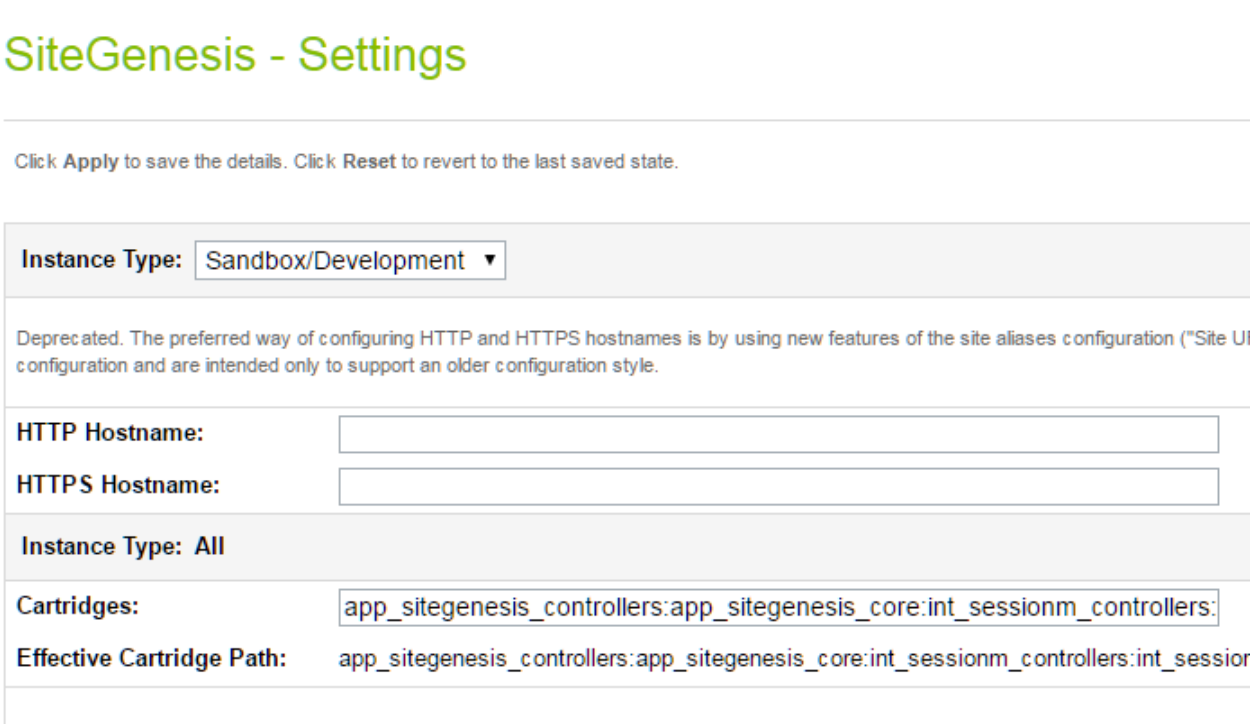
2. In the import dialog select *General → Existing projects in the workspace* and click next
3. Ensure *Select archive file* is selected and select the compressed cartridge file by clicking on the *Browse* button.
4. Click *Finish* to import the cartridge.
5. Studio will now ask you if you want to link the cartridge to your active Demandware server connection. Click on yes or manually link the cartridge to your server by checking the project under project references in the server connection properties



3.2.2 Activating the Cartridge in Business Manager

Before the SessionM functionality can become available to SiteGenesis, the cartridges have to be added to the cartridge path of the Site in question. In order to do this, follow the following instructions:

1. Log into Business Manager
2. Navigate to *Administration → Sites → Manage Sites*.
3. Click on the site name and on the next page go to the *Settings* tab.
4. In the textbox *Cartridges* append to the "**:int_sessionm_controllers:int_sessionm_core**" before cartridge inclusion
5. Click *Apply*.
6. To activate the cartridge for the Sandbox/Development/Production instances repeat steps **4** and **5** After selecting the appropriate instance from the *Instance Type* dropdown menu.
7. Repeat steps **3** to **6** for each site that is to use SessionM.

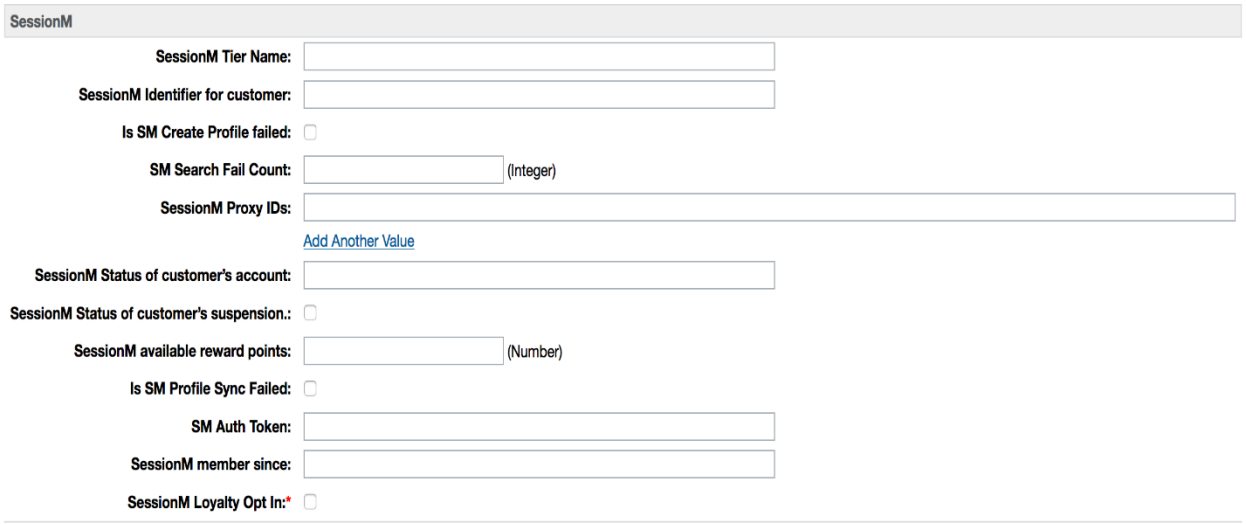


69.3 CONFIGURATION

This chapter will guide the user to configure the cartridge in Business Manager

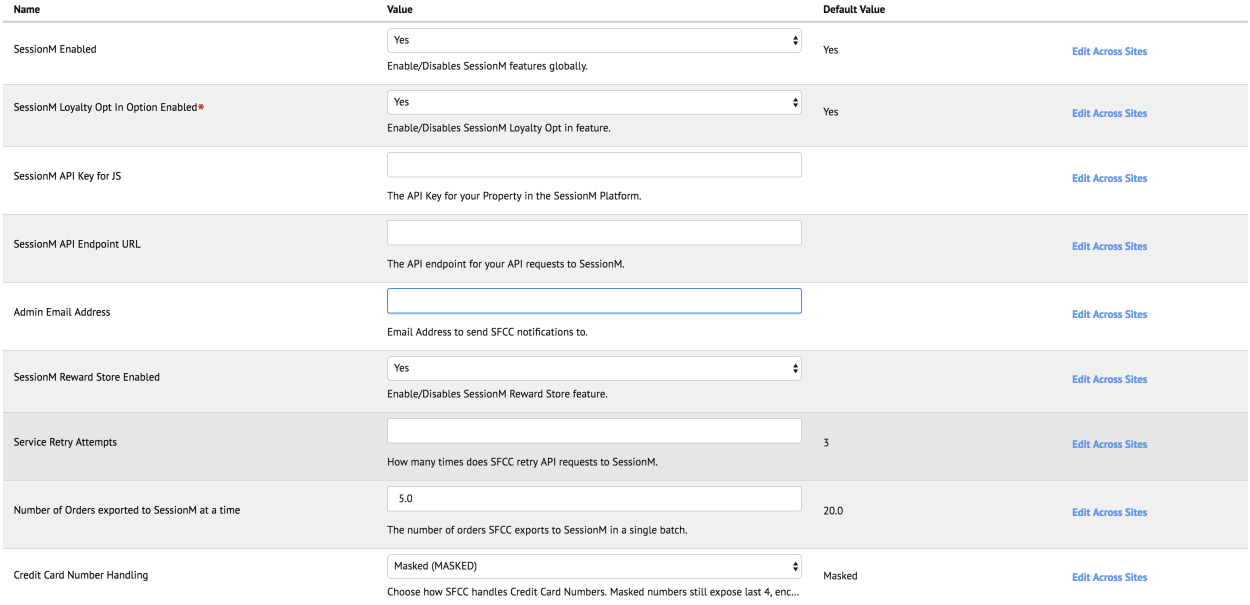
3.3.1 Importing Meta data

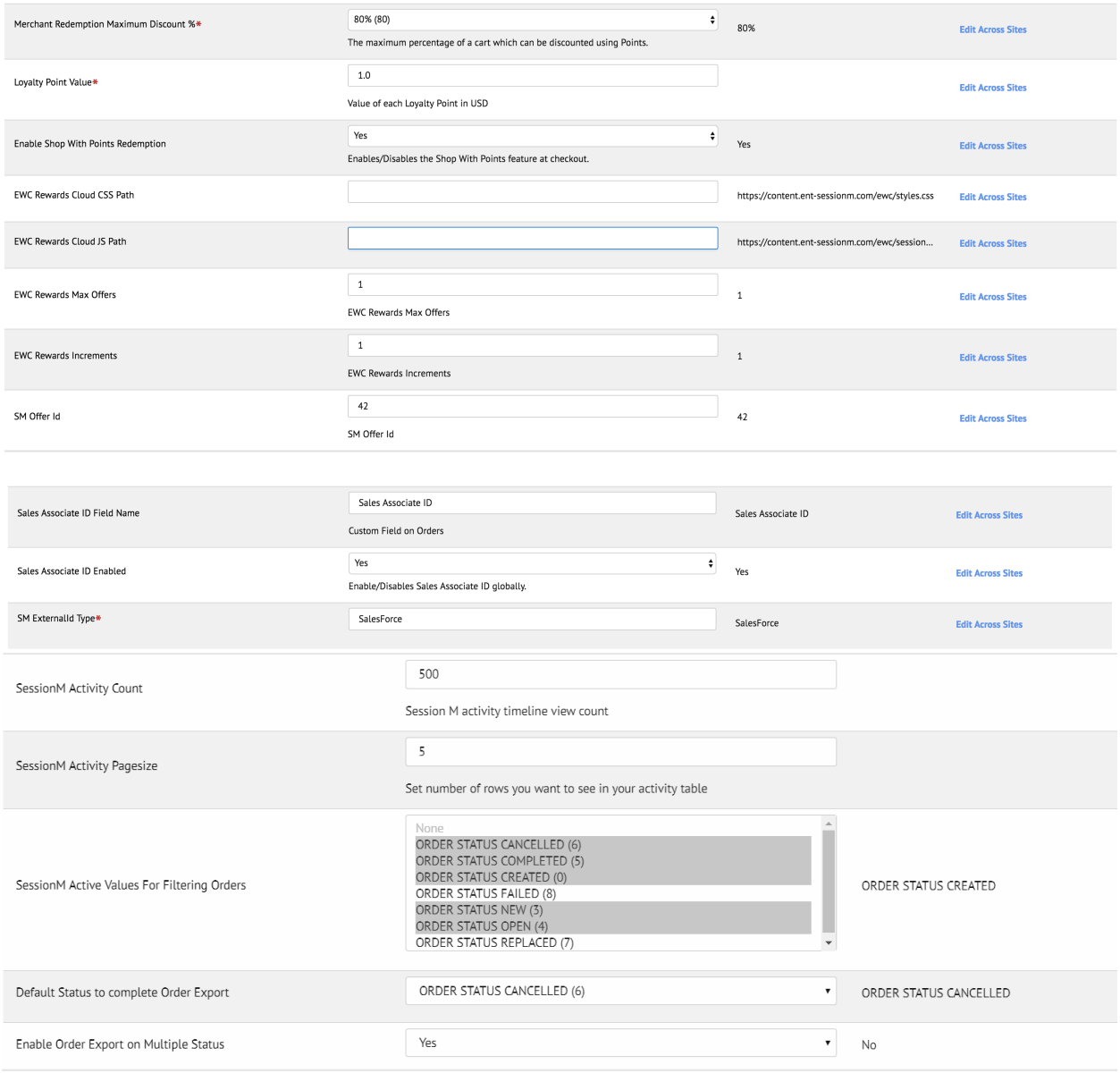
All import files can be found in the import folder within cartridge installation pack. To import all necessary SessionM settings, log in to the Business Manager and navigate to *Administration* → *Site Development* → *Import & Export*. Now upload the *sessionm_metadata.xml* file using the upload button and, finally go back and use the import button to import the file. After a successful import, you will be able to configure the custom site preferences according to your SessionM account data. Also, verify the Customer Profile Attributes in BM (Site -> Customers -> customers) open any customer and navigate to the Attributes tab. It should be as below:



3.3.2 Setting SessionM Custom Site Preference

In Business Manager, navigate to the *SiteGenesis Site* → *Site Preferences* → *Custom Preferences*. A custom site preference group with the ID SessionM is available. Please select it and edit the attributes according to your SessionM account data and the data shown in figure. In addition to the ones supplied by SessionM fill out the following properties with the values listed against them:





Sr. No.	Site Preference ID	Description	Default Value
1	IsSMEnabled	Enable/Disables SessionM features globally.	YES
2	smLoyaltyOptIn	Enable/Disables SessionM Loyalty Opt in feature.	YES
3	SMApiKey	The API Key for your Property in the SessionM Platform.	
4	SMJsUrl	The API endpoint for your	

		API requests to SessionM.	
5	smAdminEmail	Email Address to send SFCC notifications to	
6	IsSMRewardsPageEnabled customer	Enable/Disables SessionM Reward Store feature	Yes
7	SMRetryAttempt	How many times does SFCC retry API requests to SessionM.	3
8	smNoOfOrderExported	The number of orders SFCC exports to SessionM in a single batch	10
9	smCcNoExportType	Choose how SFCC handles Credit Card Numbers. Masked numbers still expose last 4, encrypted are encrypted with the public key exposed below.	Encrypted
10	smOrderRedemetionLimit	The maximum percentage of a cart which can be discounted using Points.	80%
11	smPointValue	Value of each Loyalty Point in USD	Yes
12	smRewardsRedemptionEnabled	Enables/Disables the Shop With Points feature at checkout.	Yes
13	smRewardPageCSS	EWC Rewards Cloud CSS Path	
14	smRewardPageJS	EWC Rewards Cloud JS Path	
15	SMMaxOffers	EWC Rewards Max Offers	1
16	SMIncrements	EWC Rewards Increments	1
17	smOfferId	SM Offer Id to be send in Order API	42
18	smOrderSalesAssociateIDFieldName	Sales Associate ID field which customer can fill	Sales Associate ID
19	smExternalIdType	External Id type	SalesForce
20	SM_Default_Status_to_complete_Order_Export	Default Status to complete Order Export	5
21	SM_ISOrder_Export_on_Multiple_Status	Enable Order Export on Multiple Status	1
22	SessionM_Active_Order_Filter_Values	SessionM Active Values For Filtering Orders	0

23	SessionM_Activity_Count	SessionM Activity Count	5
24	SessionM_Activity_Pagesize	SessionM Activity Pagesize	10

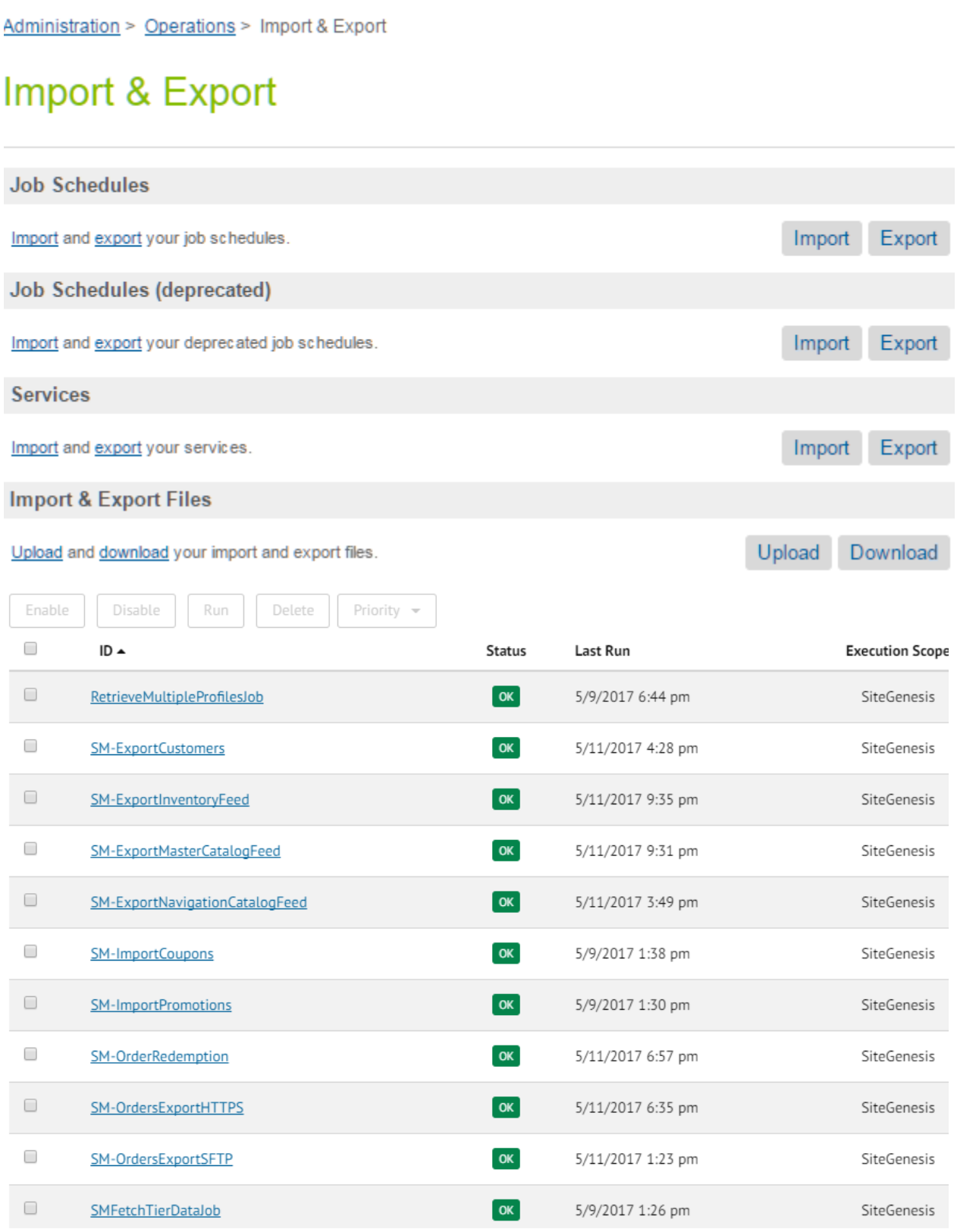
3.3.3 Importing Jobs

Prerequisite: Make sure to append "int_sessionm_pipelines" in Business Manager under Manage Sites > Settings for your site

In the integration package, a job definition is provided in *sessionm_jobs.xml* file. All import files can be found in the import folder within the cartridge installation pack. To import this file follow the instructions below:

1. In Business Manager navigate to *Administration* → *Operations* → *Import & Export*.
2. Click on the upload button and in the next screen select the file by clicking the browse button. Notice that the file is in the import subdirectory. After you have selected the file click on the upload button.
3. After the file has been uploaded click on the back button. In the Job Schedules area, click on the Import button.
4. Select the file you just uploaded and click on the Next button.
5. After the file is successfully validated click on the Next button.
6. Make sure MERGE is selected and click on the Import button:

Please find below screen shots for Import Jobs and list of Jobs :

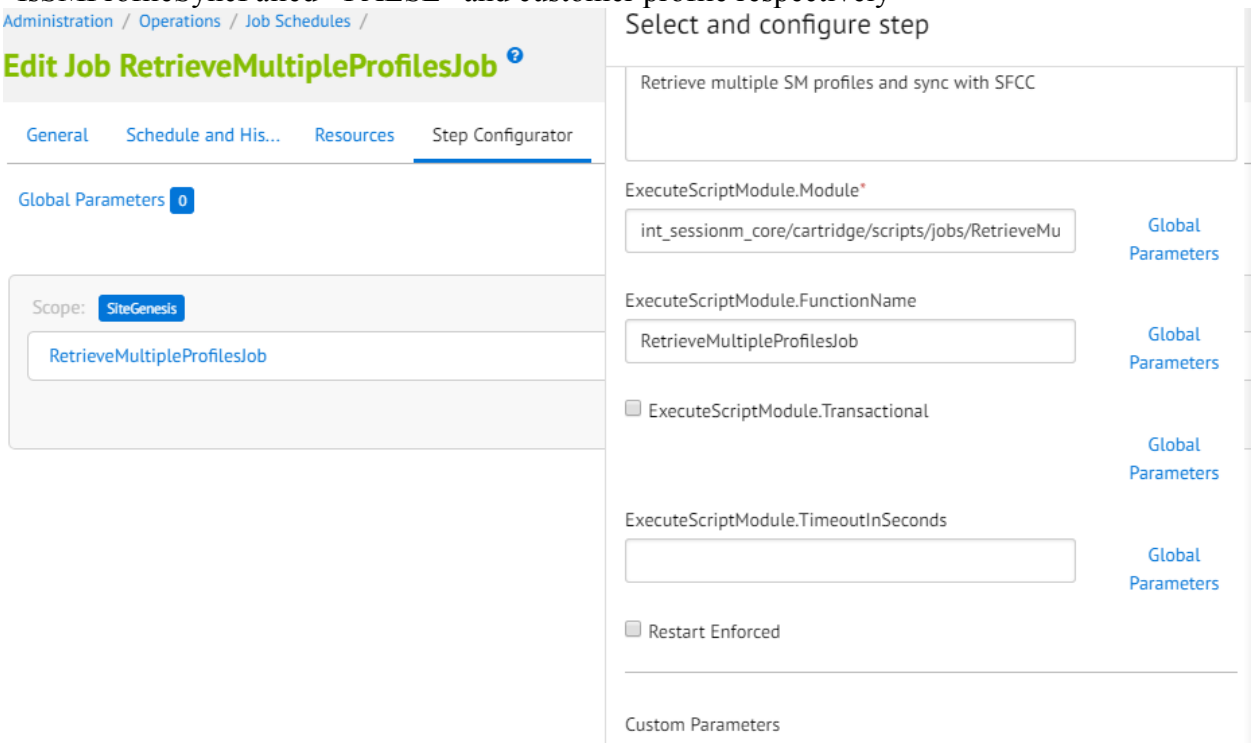

Administration > Operations > Import & Export
Import & Export
Job Schedules
Import and export your job schedules.
Import
Export
Job Schedules (deprecated)
Import and export your deprecated job schedules.
Import
Export
Services
Import and export your services.
Import
Export
Import & Export Files
Upload and download your import and export files.
Upload
Download
Enable
Disable
Run
Delete
Priority
ID
Status
Last Run
Execution Scope
RetrieveMultipleProfilesJob OK 5/9/2017 6:44 pm SiteGenesis
SM-ExportCustomers OK 5/11/2017 4:28 pm SiteGenesis
SM-ExportInventoryFeed OK 5/11/2017 9:35 pm SiteGenesis
SM-ExportMasterCatalogFeed OK 5/11/2017 9:31 pm SiteGenesis
SM-ExportNavigationCatalogFeed OK 5/11/2017 3:49 pm SiteGenesis
SM-ImportCoupons OK 5/9/2017 1:38 pm SiteGenesis
SM-ImportPromotions OK 5/9/2017 1:30 pm SiteGenesis
SM-OrderRedemption OK 5/11/2017 6:57 pm SiteGenesis
SM-OrdersExportHTTPS OK 5/11/2017 6:35 pm SiteGenesis
SM-OrdersExportSFTP OK 5/11/2017 1:23 pm SiteGenesis
SMFetchTierDataJob OK 5/9/2017 1:26 pm SiteGenesis

☐	RetryCreateProfiles	OK	7/20/2018 11:32 am	SiteGenesis

Enable the jobs that need to be activated, apply the configurations setting of scheduling into **General** and **Sites** sections one by one

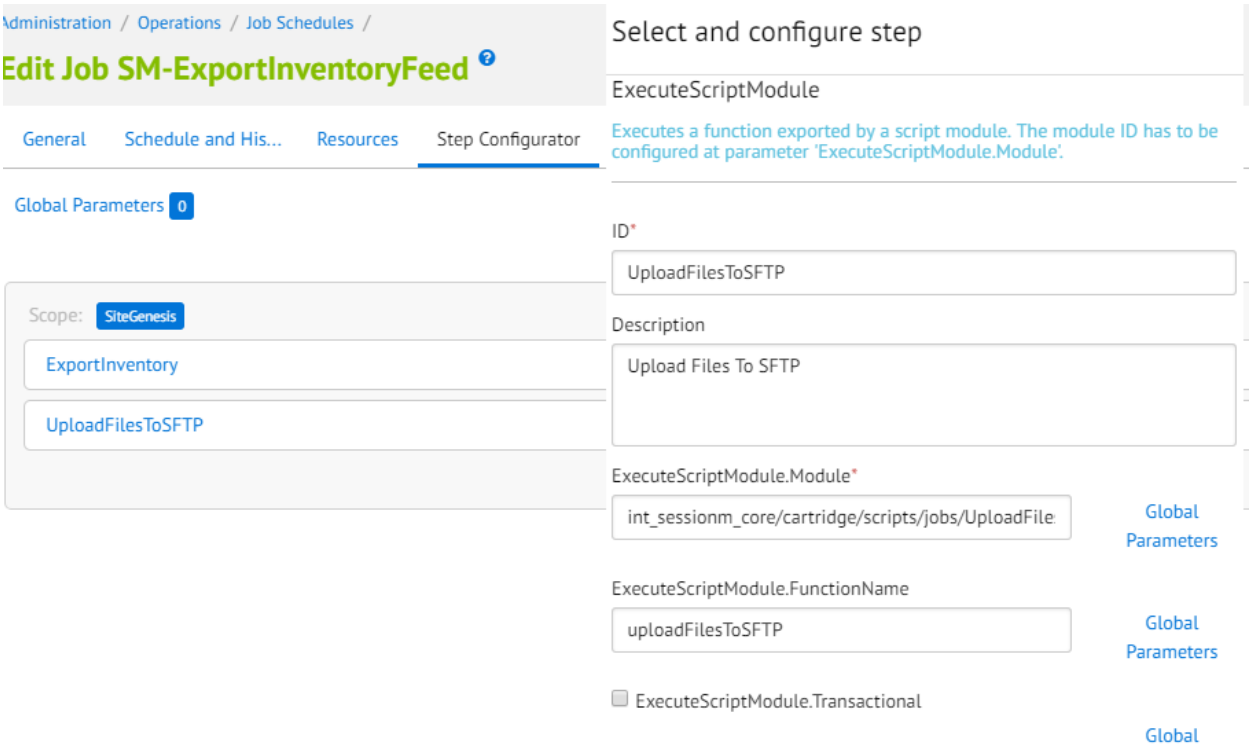
3.3.3.1 SM Retrieve Multiple Profile Job

This Job will pick the list of customers which are failed to synced on SFCC and with flag value "IsSMProfileSyncFailed= TRUE" and "IsSMProfileCreationFailed" = FALSE which is being set at the time of Sign In to SFCC account and when a new customer is registered in SessionM. SFCC read the response containing SM ID, external_id and email and update sync status as "IsSMProfileSyncFailed =FALSE" and customer profile respectively



3.3.3.2 SM Export Inventory feed Job

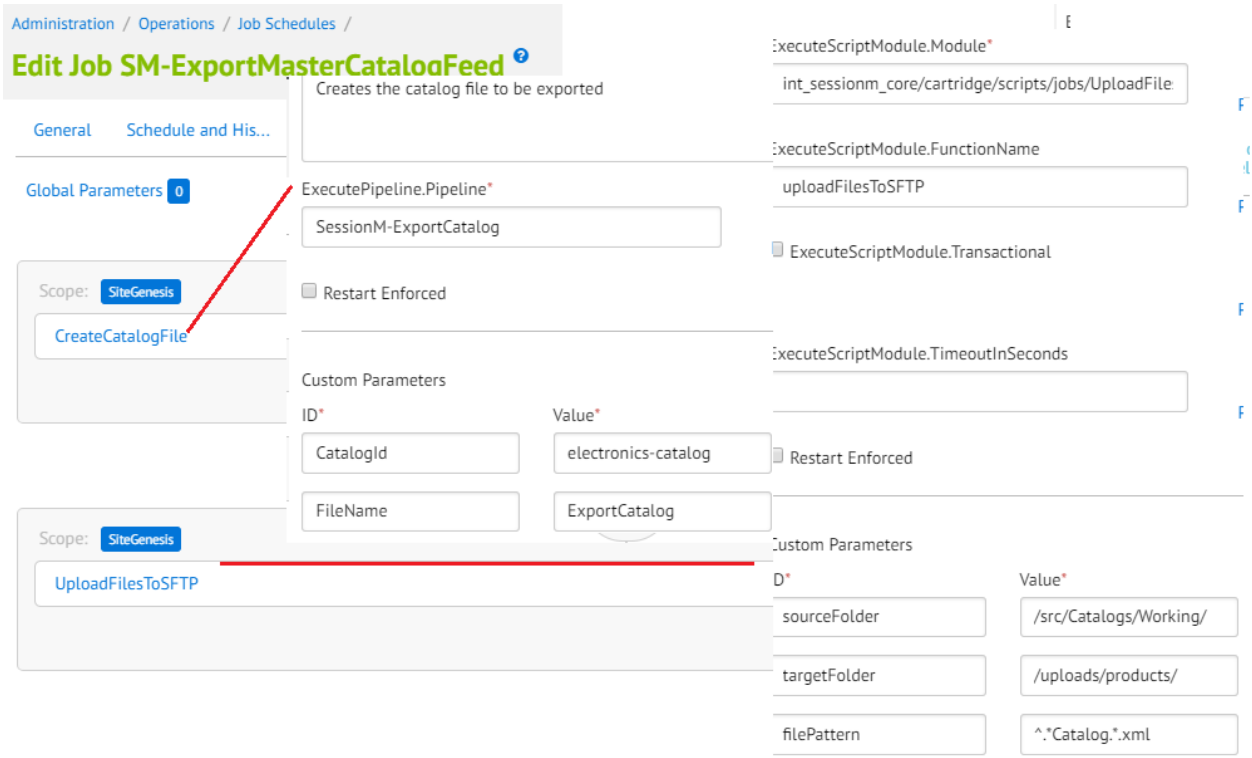
The SFCC job's framework would be leveraged to create the Inventory feed export file. The file will be in the OOTB SFCC format and would send to an SM SFTP location.



3.3.3.3 SM Product Export Feed

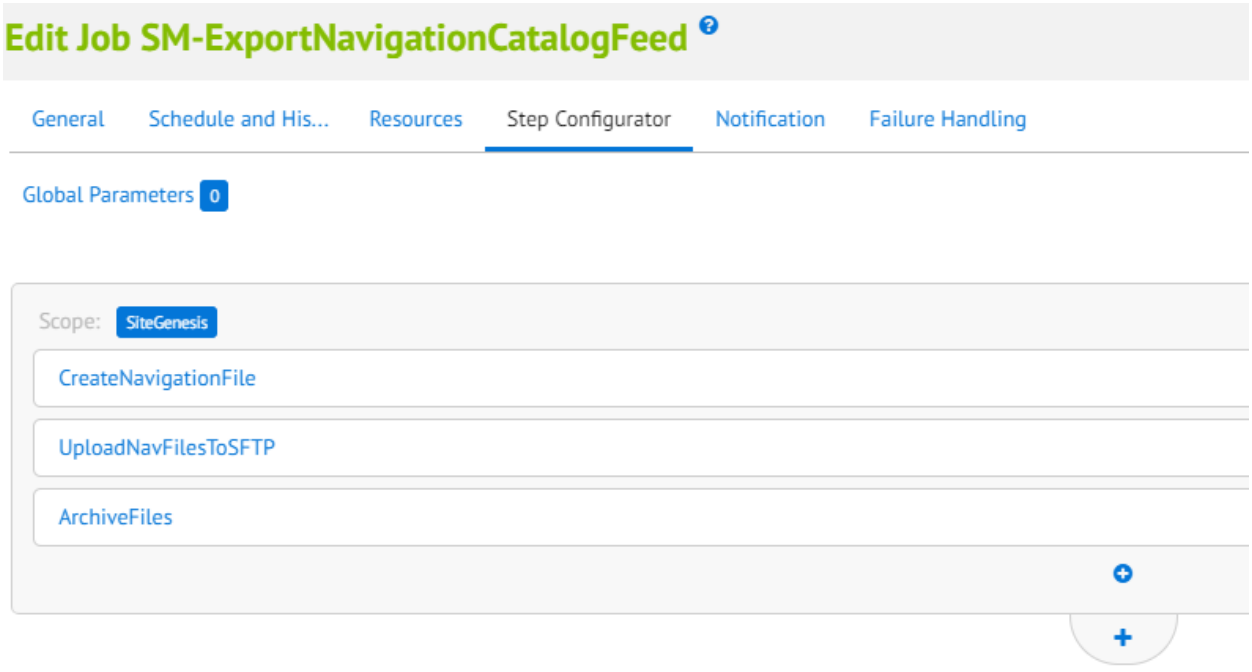
The export feed would export all the catalogs present in SFCC i.e. the Master catalogs and all the subsequent Navigation catalogs assigned to the sites in SFCC.

1. The xml file to be exported would be created in the SFCC IMPEX folder.
2. After the file has been created it would be exported to the SessionM SFTP location.
3. After the successful export the file would be archived at a predetermined location in IMPEX.
   If SFTP transfer fails then the job finishes with error and email notification is sent through email



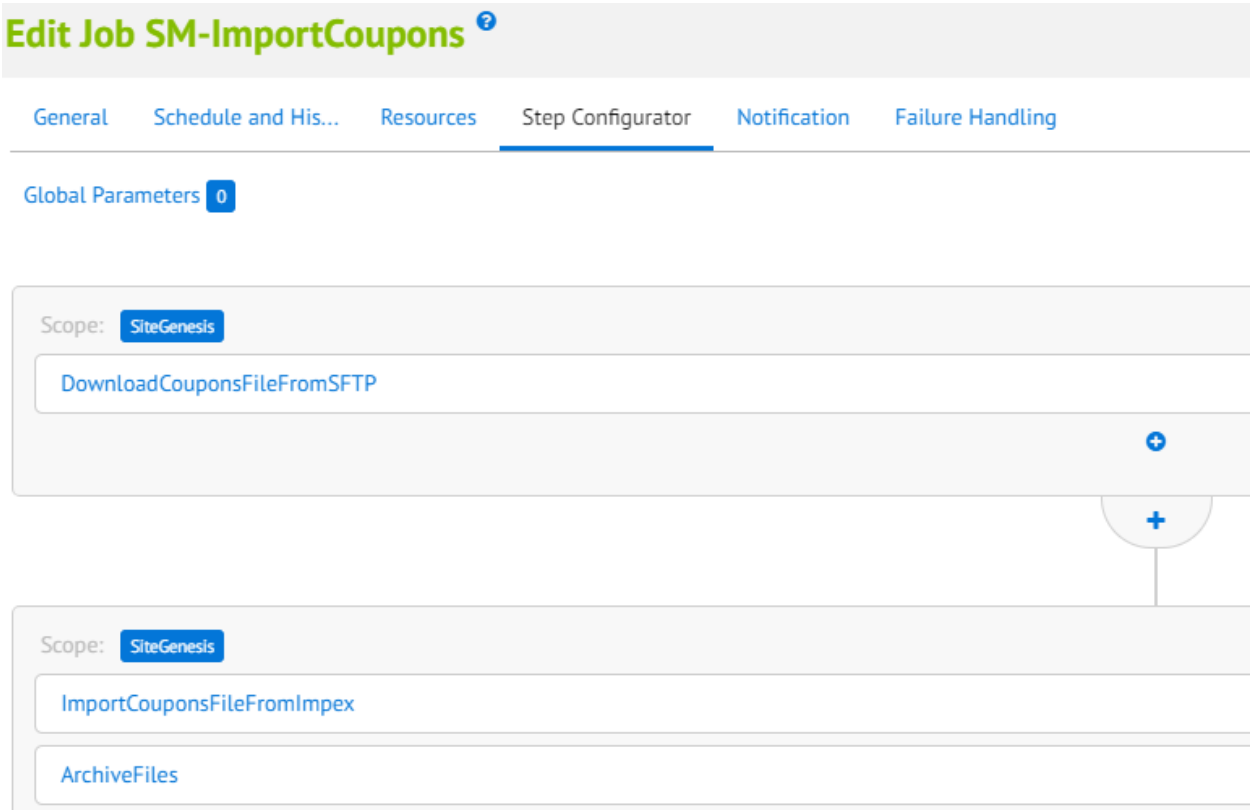
3.3.3.4 SM Export Navigation Catalog Feed Job

SFCC navigation catalog feed job is used to export navigation catalog to SessionM is SFCC OOTB format.



3.3.3.5 SM Import Coupons Job

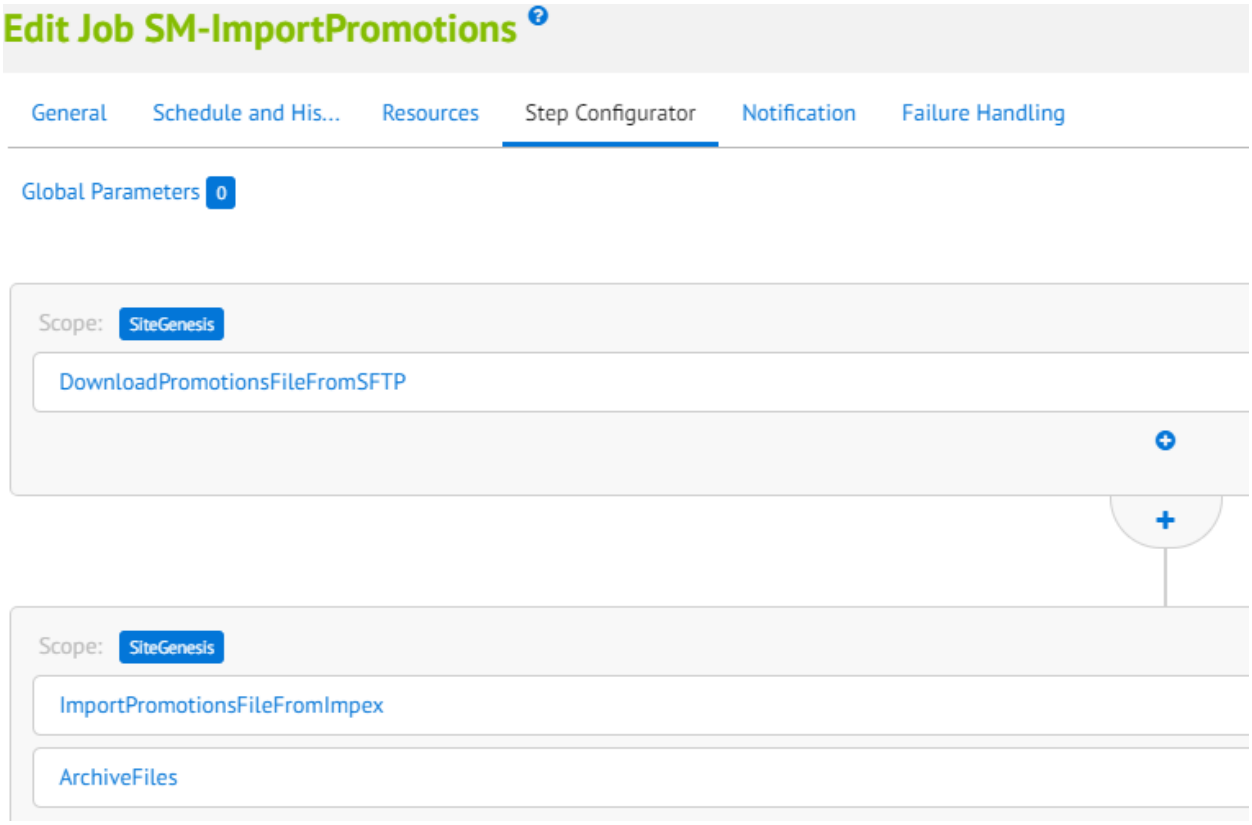
1. Coupon feed xml document would be in the SFCC format.
2. Site for which the file will be imported will be set in the job by the merchant in SFCC. As part of SessionM cartridge job setup is done for SFCC Site Genesis site.
3. The xml file to be imported to SFCC IMPEX first.
4. After file has been successfully transferred to SFCC IMPEX, it would be imported in SFCC.
5. After the successful import the file would be archived at a predetermined location in IMPEX.
6. If file is not available on SFTP transfer then the job finishes with error and email notification is sent through email.



3.3.3.6 SM Import Promotions Job

1. Campaign & Promotion feed xml document would be in the SFCC format.
2. Site for which the file will be imported will be set in the job by the merchant in SFCC. As part of SessionM cartridge, job setup is done for SFCC Site Genesis site.
3. The xml file to be imported to SFCC IMPEX first.
4. After file has been successfully transferred to SFCC IMPEX, it would be imported in SFCC.
5. After the successful import the file would be archived at a predetermined location in IMPEX.

6. If file is not available on SFTP transfer then the job finishes with error and email notification is sent through email.

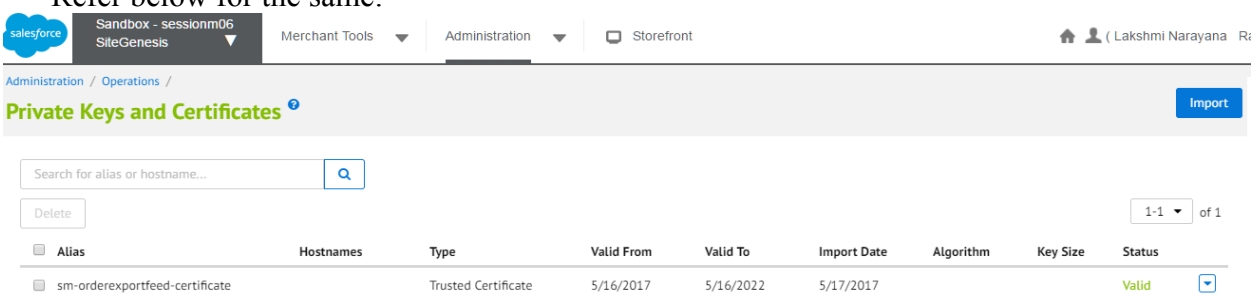


3.3.3.7 SM Order Export HTTPS Job

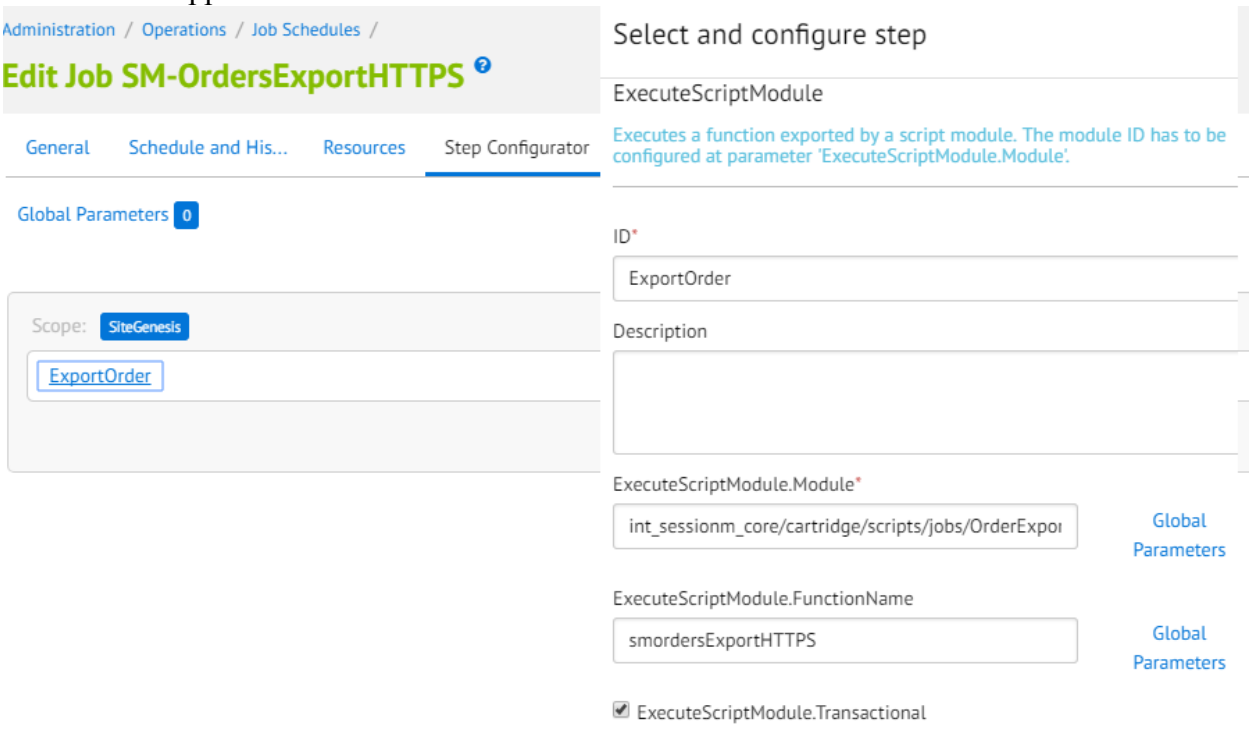
Order export feed would be leveraging the latest SFCC Job framework. All the orders would be exported to SessionM through the Job feed.
In order to run Order Export job, merchant should first import certificate using below steps:

1. Merchant should create a valid Certificate on their end.
2. Import Certificate in Business Manager under *Administration* → *Operations*→ *Private Keys and Certificates*
3. While Importing set Alias as '**sm-orderexportfeed-certificate**'.

Refer below for the same:



The orders would be exported through service calls to SessionM and the service call body would be in SFCC supported XML format

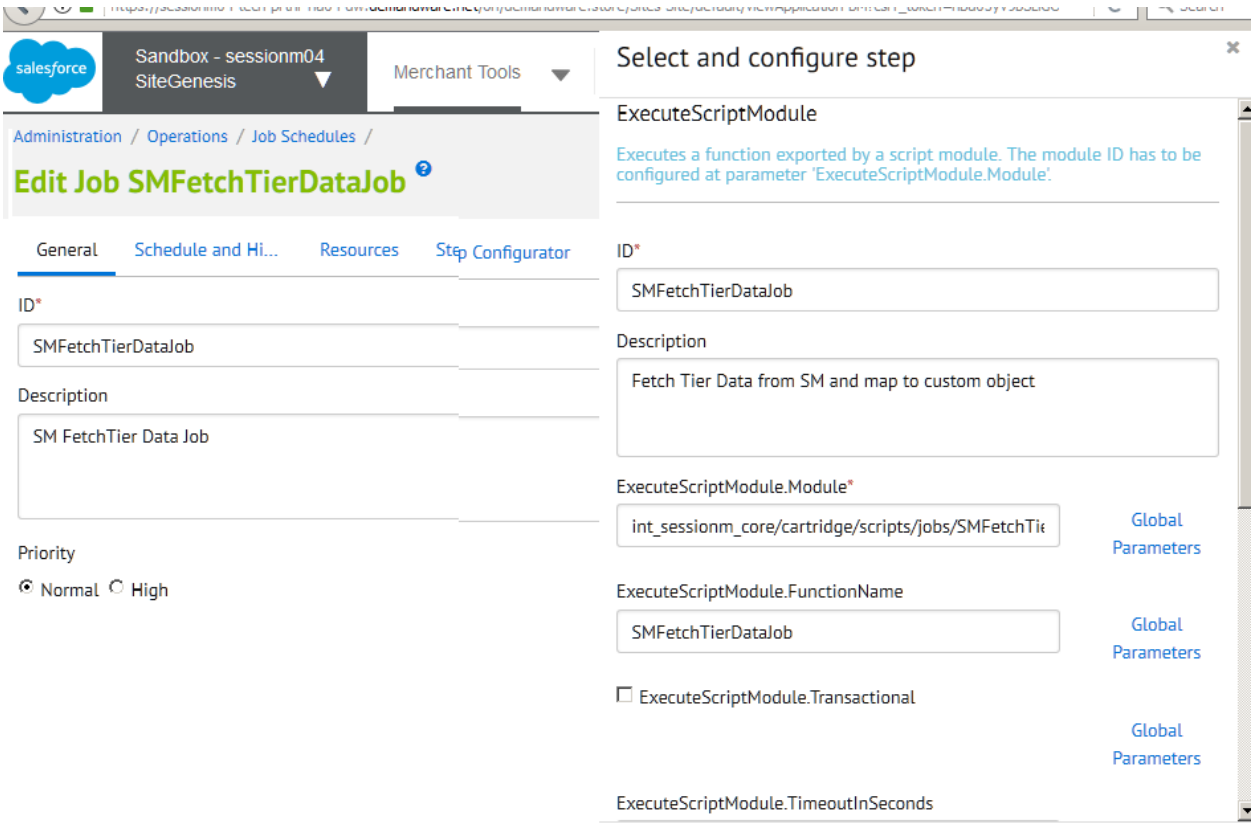


3.3.3.8 SM – SFCC Tier Status Display

SFCC tier data batch job will fetch tier details from SessionM and stored the detail in custom objects in SFCC in order to display tier details on customer account page as well as on billing page.

1. A Batch Job configured in SFCC Business Manager shall run (as per the set schedule), which internally call SessionM tier data API to fetch tier details.
2. Based on response received from SessionM, SFCC create custom object for each tier in order to display tier status on account and billing section.

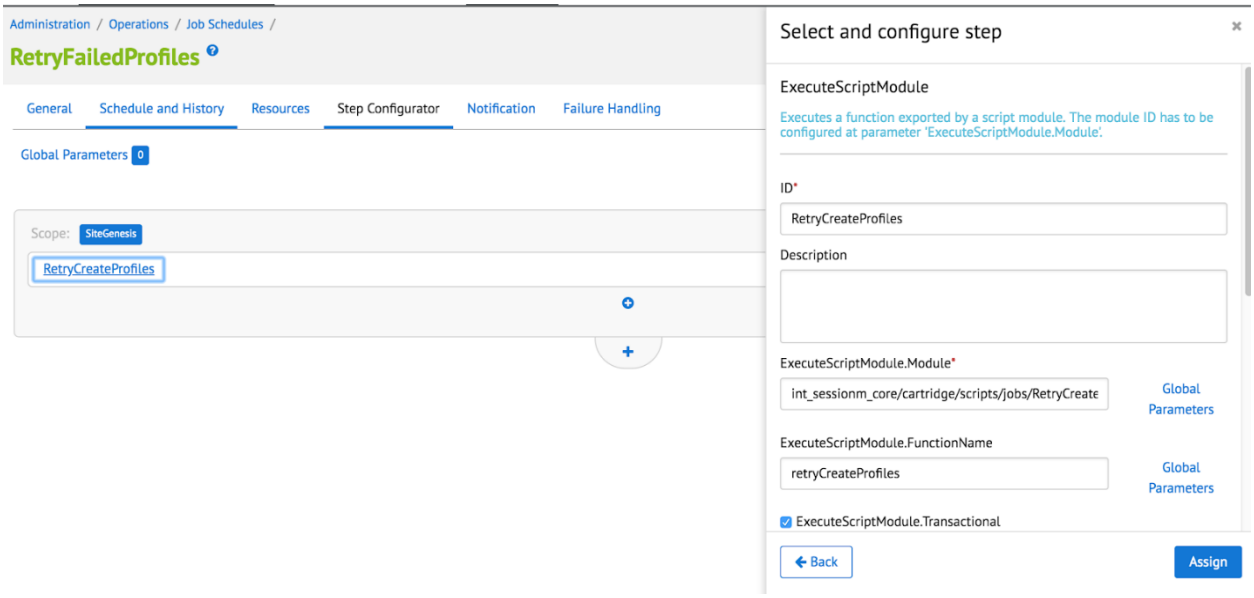
   The Tier information help customer to know their current tier status and next possible tier.



3.3.3.9 Retry Create Profiles

This Job will pick the list of customers from SFCC which have attribute value “smUUID” = null which is set when a new customer is registered in SessionM.

It’ll check if the customer exists on SessionM Platform. If the customer exists, it is synced on SFCC and smUUID is set with user id from SessionM. If the customer does not exist on SessionM, customer is created on SessionM Platform and synced with SFCC.



3.3.4 Importing Services

To import the service file from integration package, follow the instructions below:

1. In Business Manager navigate to Administration → Operations→ Import & Export.
2. Click on the upload button and in the next screen select the file session_services.xml by clicking the browse button. Notice that the file is in the import subdirectory. After you have selected the file click on the upload button.
3. After the file has been uploaded click on the back button. In the Import/export Service area, click on the Import button.
4. Select the file you just uploaded and click on the Next button.
5. After the file is successfully validated click on the Next button.
6. Make sure MERGE is selected and click on the Import button.

Below is the list of Services consumed in SessionM cartridge:

Services

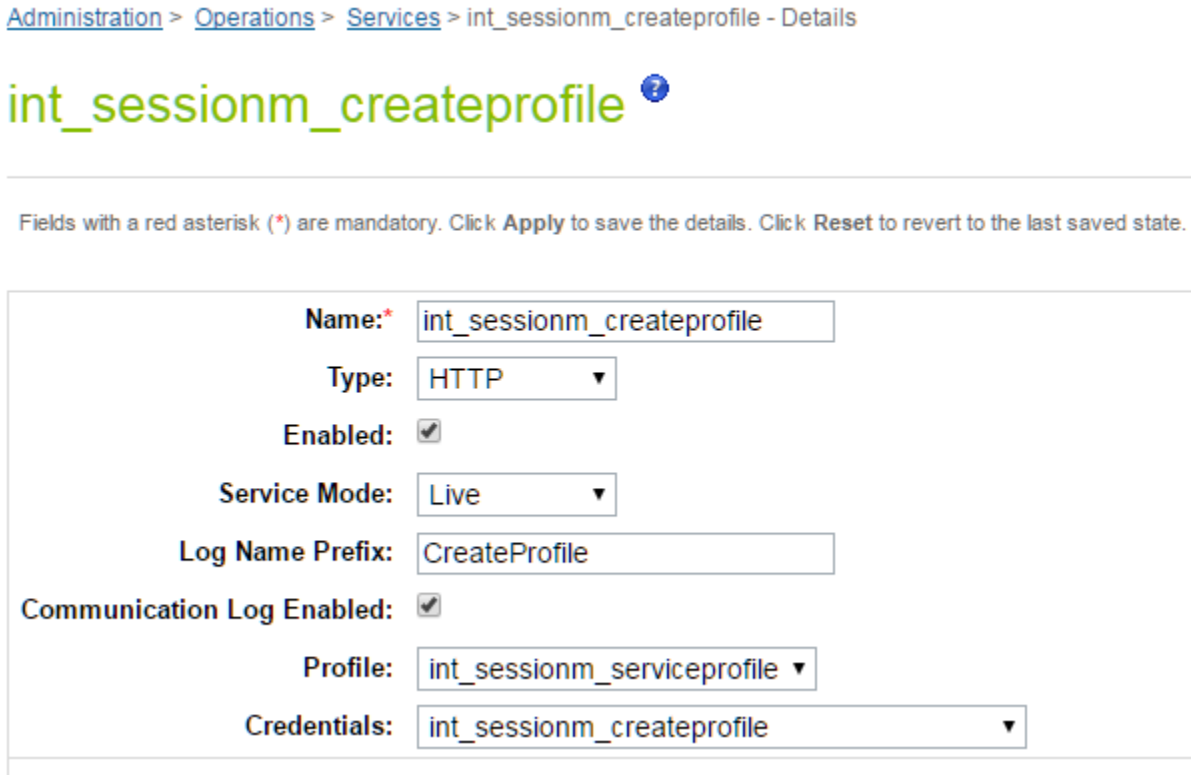
Select All	Name	Type	Profile	Credentials	Status
	int_sessionm_createprofile	HTTP	int_sessionm_serviceprofile	int_sessionm_createprofile	Live
	int_sessionm_exportorder	HTTP	int_sessionm_serviceprofile	int_sessionm_exportorder_servicecredential	Live
	int_sessionm_order	HTTP	int_sessionm_serviceprofile	int_sessionm_order_credential	Live
	int_sessionm_retrievemultipleprofile	HTTP	int_sessionm_serviceprofile	int_sessionm_retrievemultipleprofile	Live
	int_sessionm_retrieveprofile	HTTP	int_sessionm_serviceprofile	int_sessionm_retrieveprofile	Live
	int_sessionm_searchprofile	HTTP	int_sessionm_serviceprofile	int_sessionm_searchprofile	Live
	int_sessionm_sftpservice	SFTP	int_sessionm_serviceprofile	int_sessionm_sftpservice	Live
	int_sessionm_tierdetail	HTTP	int_sessionm_serviceprofile	int_sessionm_tierdetail	Live
	int_sessionm_updateprofile	HTTP	int_sessionm_serviceprofile	int_sessionm_updateprofile	Live

New Delete

3.3.4.1 SessionM New Accout Creation Service

SFCC Create Profile service builds a new standard profile for a customer. Provides the primary operation for adding a standard profile to the SFCC and specifying that customer's characteristics. Below are the service Configurations:

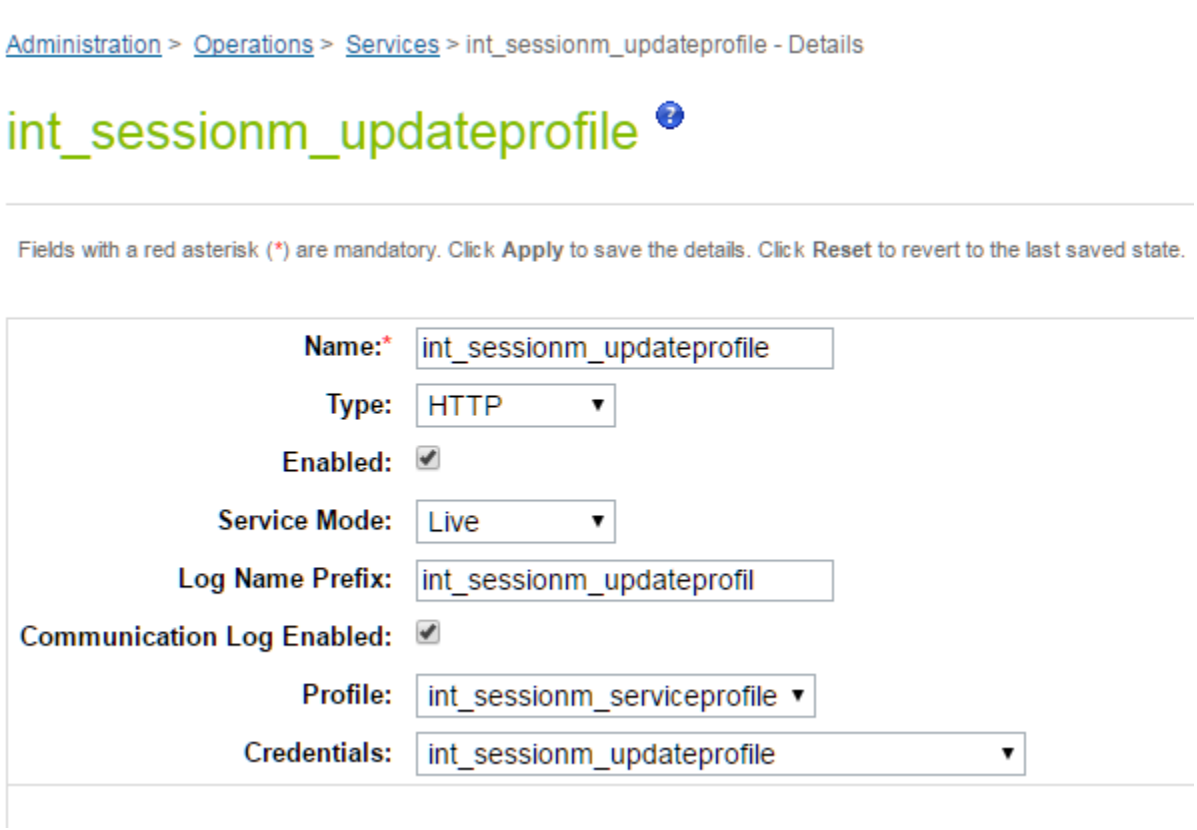
Service Configuration Field	Value
Service Name/ID	int_sessionm_createprofile
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_createprofile



3.3.4.2 SessionM Customer Profile Update

SFCC update Profile service updates a standard profile for a customer with new data. Below are the service Configurations for customer profile Update:

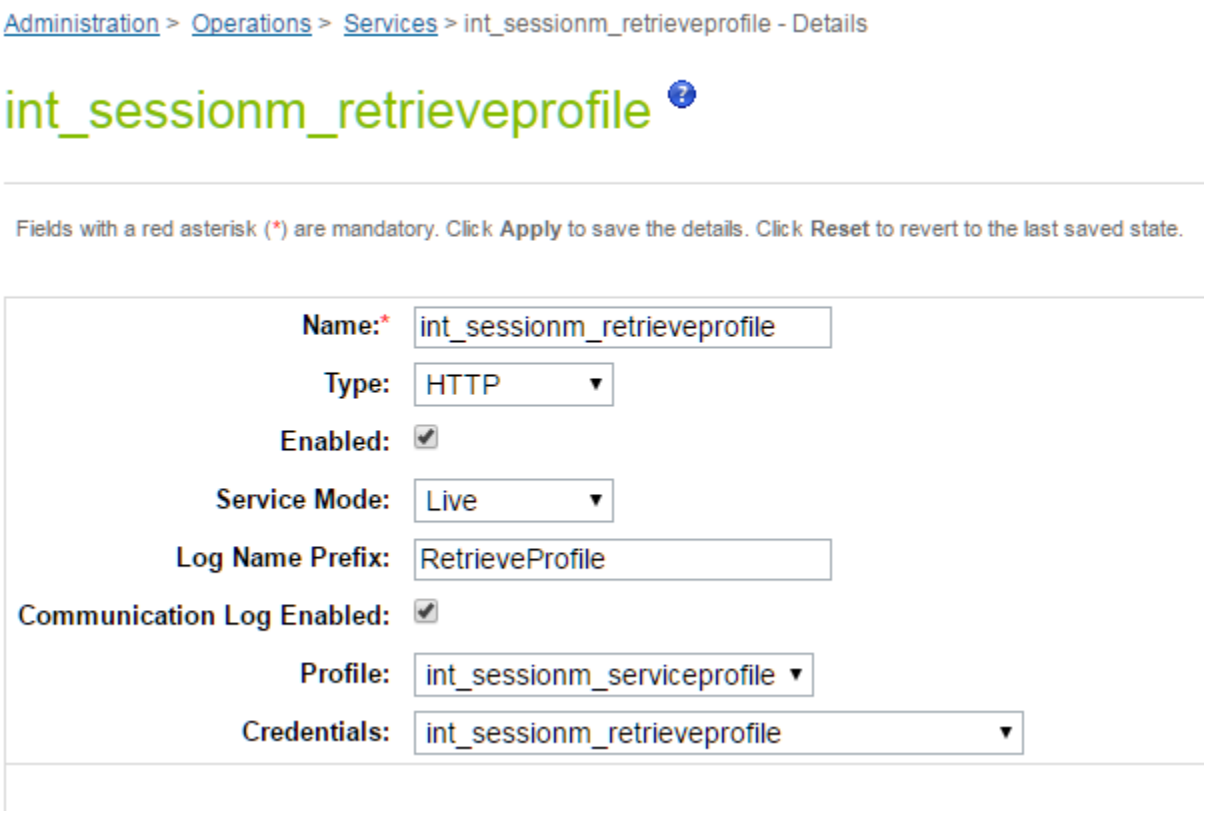
Service Configuration Field	Value
Service Name/ID	int_sessionm_updateprofile
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_updateprofile



3.3.4.3 SessionM Customer Profile Sync

SFCC Retrieve Profile Service retrieves an existing standard profile for a customer with all of the profile's associated attributes. Below are the service Configurations for customer profile sync:

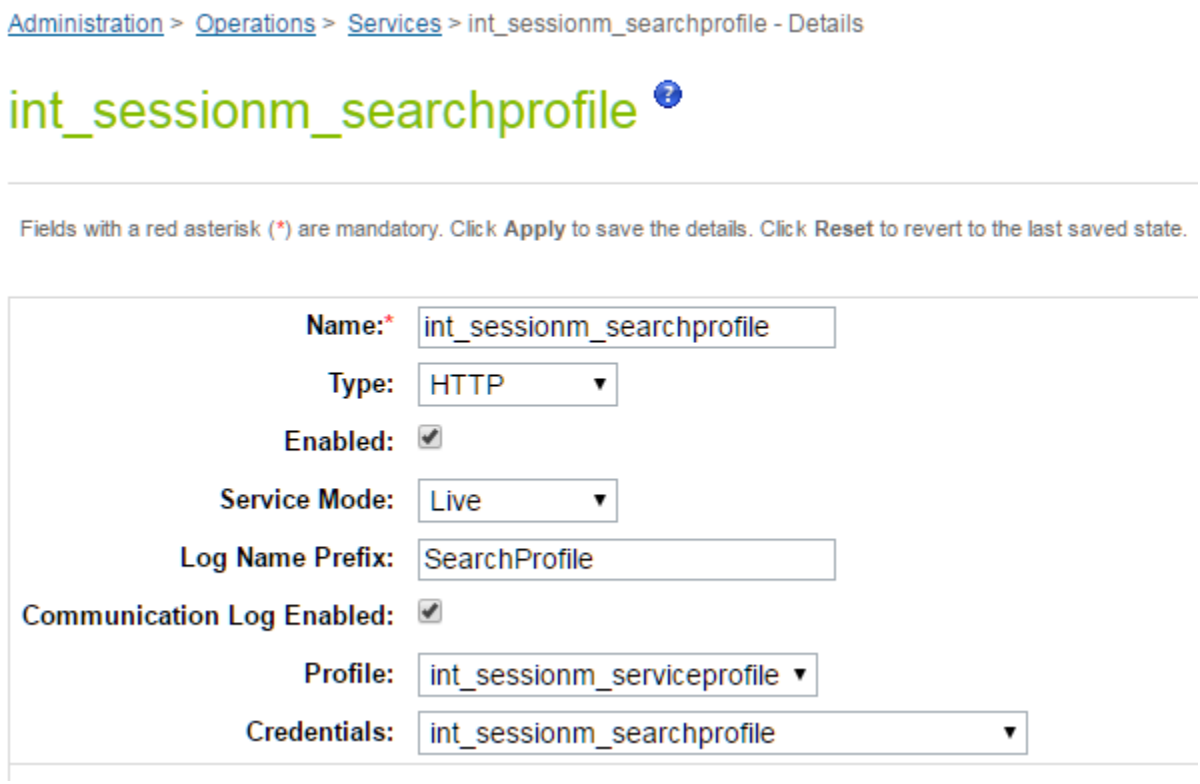
Service Configuration Field	Value
Service Name/ID	int_sessionm_retrieveprofile
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_retrieveprofile



3.3.4.4 SessionM Customer Profile Search

SFCC search profile service searches for an existing standard customer profile. Returns the specified customer profile with all of its associated attributes.
Below are the service Configurations for customer profile Search:

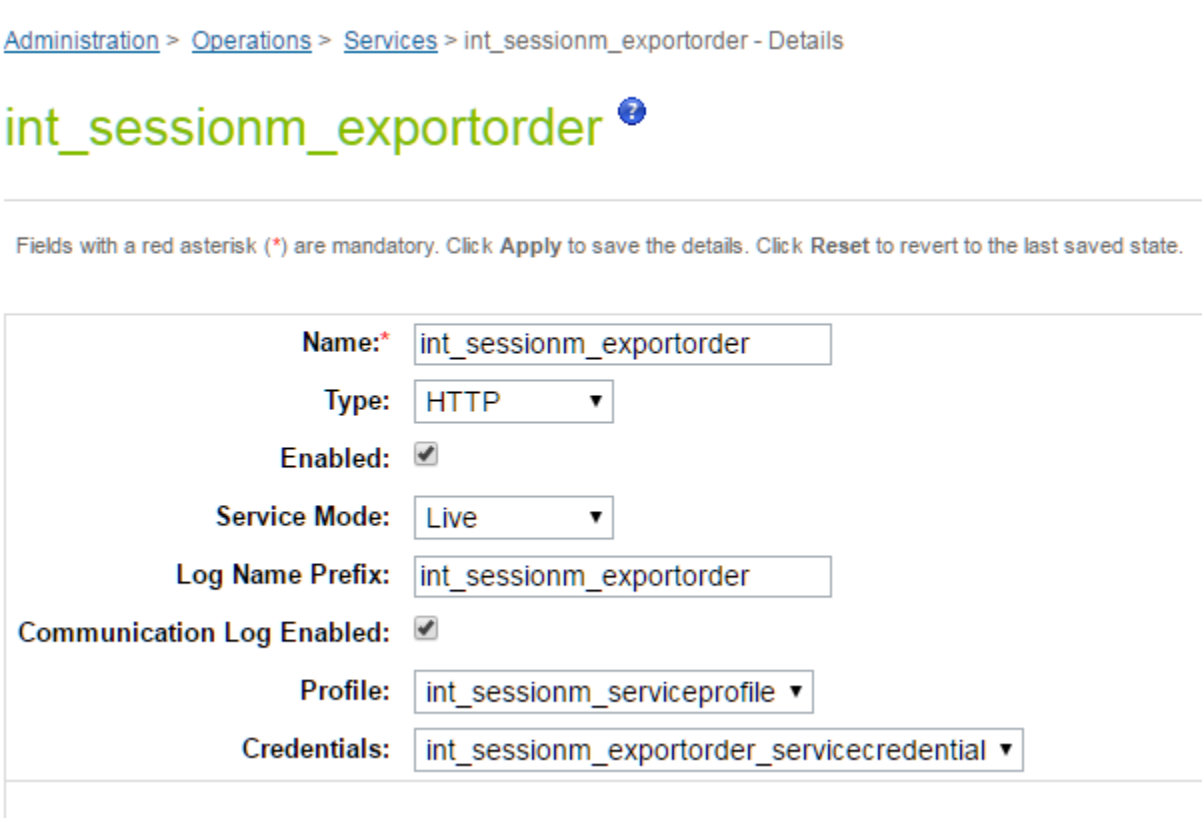
Service Configuration Field	Value
Service Name/ID	int_sessionm_searchprofile
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_searchprofile



3.3.4.5 SessionM Export Order Service

SFCC Export Order service via HTTPS is used to export orders placed by logged in customer to SessionM. The orders are exported in default SFCC format.
Below are the service Configurations for SM Order Export:

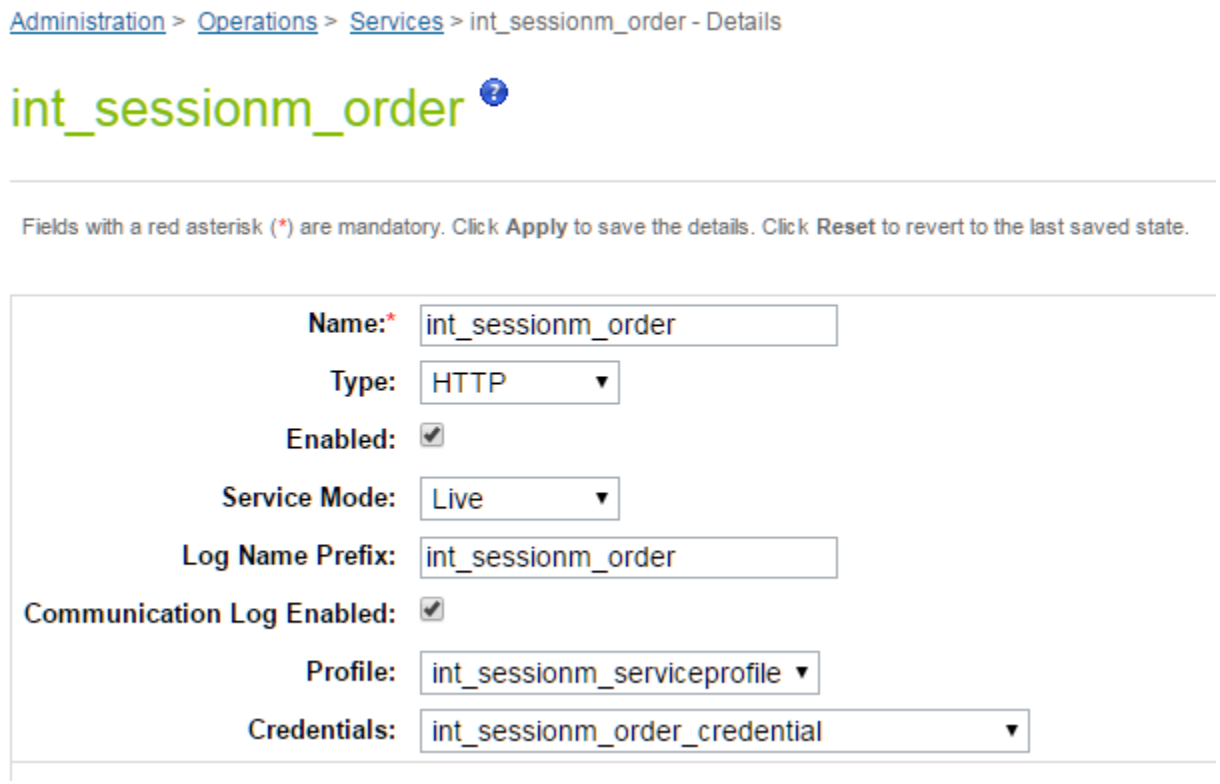
Service Configuration Field	Value
Service Name/ID	int_sessionm_exportorder
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_exportorder_servicecredential



3.3.4.6 SessionM Order Service

SFCC Order service is used to export orders to SFTP for redemption of reward points.
Below are the service Configurations for SM Order Service:

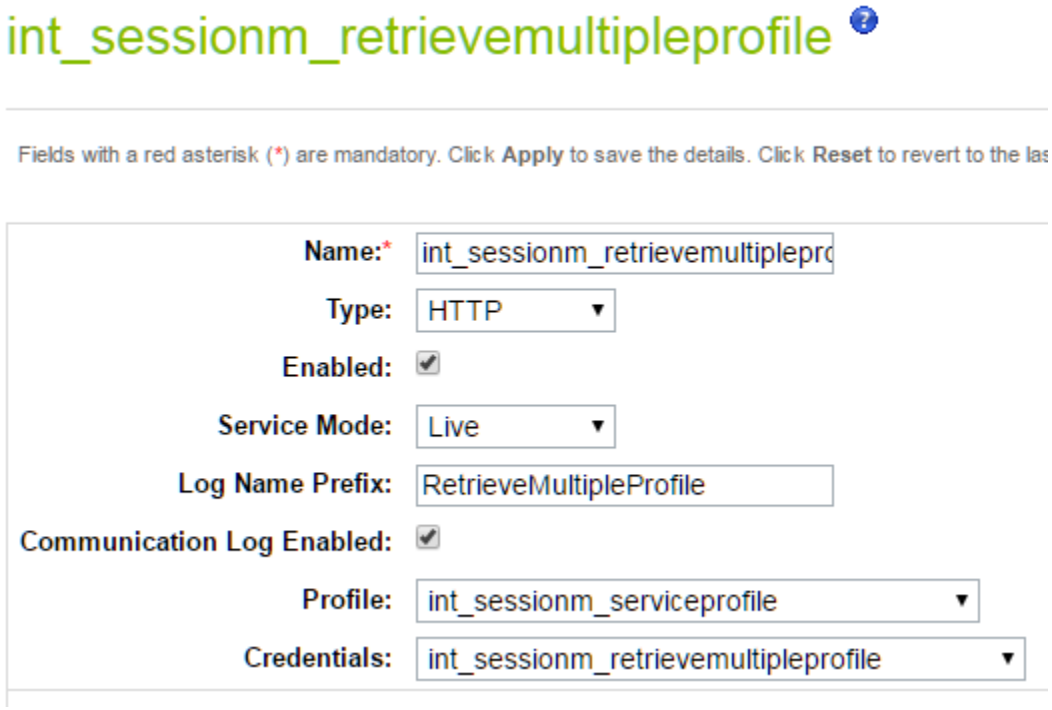
Service Configuration Field	Value
Service Name/ID	int_sessionm_order
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_order_credential



3.3.4.7 SessionM Retrieve Multiple Profile Service

SessionM Retrieve Multiple Profile service retrieves multiple standard profiles for customers with all of their associated attributes.
Below is the service Configurations for SM Retrieve multiple Customer Service:

Service Configuration Field	Value
Service Name/ID	int_sessionm_retrievemultipleprofile
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_retrievemultipleprofile

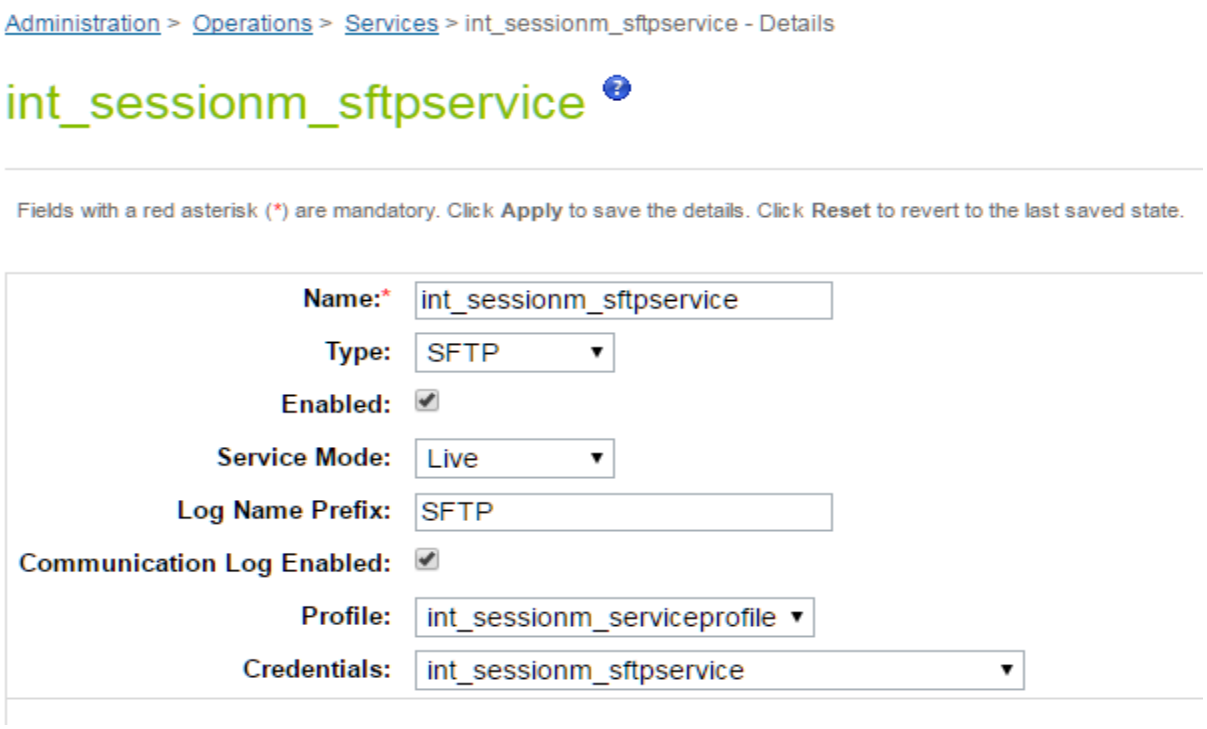


3.3.4.8 SessionM SFTP Service

SFCC SFTP service is used to export/import files between SFCC IMPEX and SessionM SFTP server.
Below are the service Configurations for SM Order Service:

Service Configuration Field	Value
Service Name/ID	int_sessionm_sftpservice
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_sftpservice

Service Profile Credentials	Description
Name	int_sessionm_sftpservice
URL	sftp.ent-sessionm.com
User	Sapient
Password	-

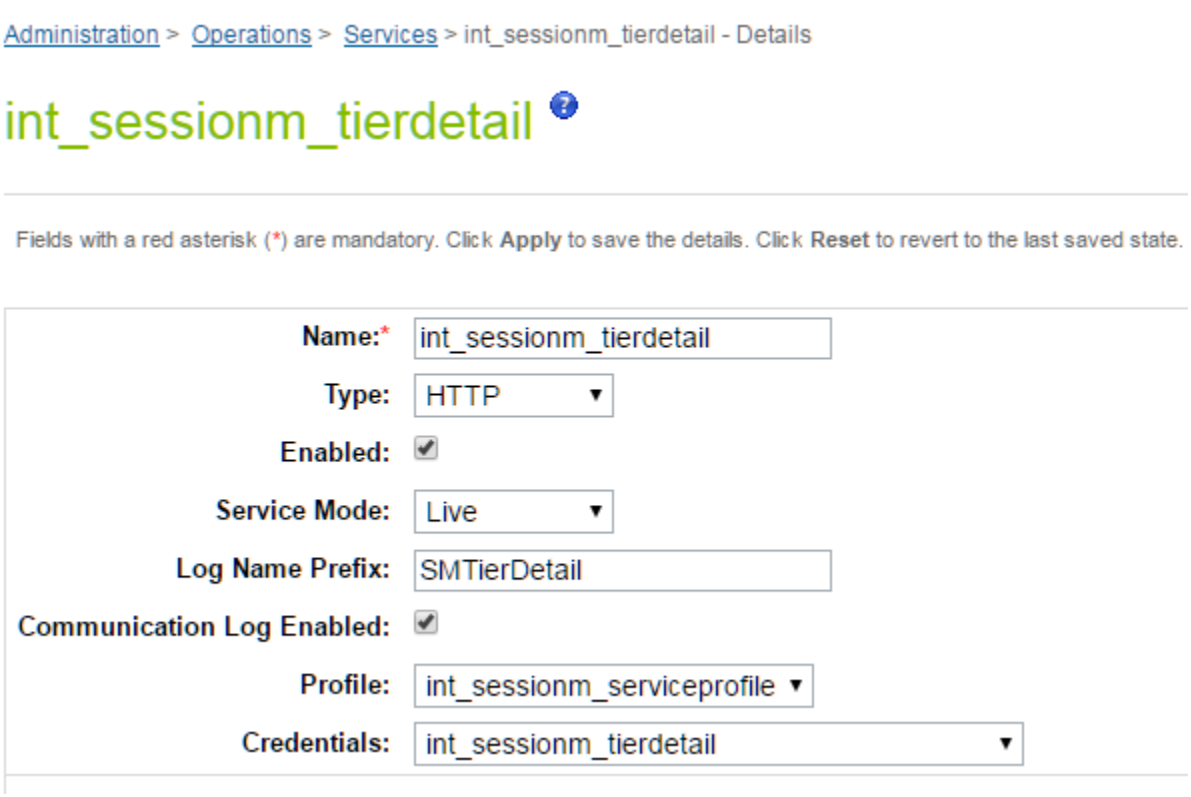


3.3.4.9 SessionM Tier Detail Service

SFCC tier data batch job will fetch tier details from SessionM and stored the detail in custom objects in SFCC in order to display tier details on customer account page as well as on billing page.

Below are the service Configurations for SM Order Service:

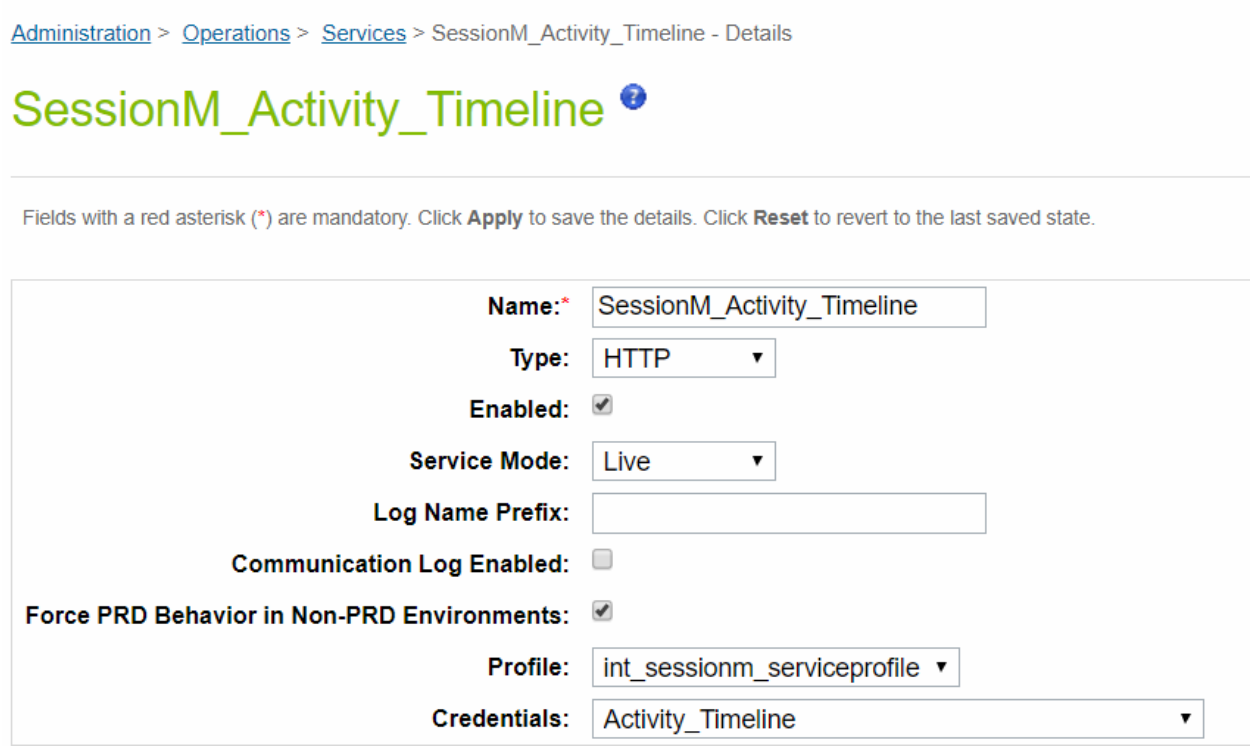
Service Configuration Field	Value
Service Name/ID	int_sessionm_tierdetail
Profile	int_sessionm_serviceprofile
Credentials	int_sessionm_tierdetail



3.3.4.10 SessionM Activity Timeline Service

SFCC Activity Timeline page incase if you want your users to see how many points are there in there wallet and what are their recent transactions look like. Which will appear in Tier Progress tab under Account Detail section.
Below are the service Configurations for SM Order Service:

Service Configuration Field	Value
Service Name/ID	SessionM_Activity_Timeline
Profile	int_sessionm_serviceprofile
Credentials	Activity_Timeline



3.4 CUSTOM CODE SECTION

Pre-Requisite: Make sure the controller cartridges of site site-genesis is (say, e.g. app_sitegenesis_controllers and "int_sessionm_core, int_sessionm_controllers" are specified in Site Settings path under Manage Sites > Merchant Site as per current site.

This section covers generic code changes for controller.

[Note]: All Sites related constants are defined in **CommonConstants.js** under int**_sessionm_core** cartridge. Please refer the same file to update the value for future reference.

```
/*SessionM attribute mapping Arrays*/
var addressArr = ['ID','address1','address2','city','companyName','countryCode','firstNa
commonConstants.ADDRESSARRAY = addressArr;
var profileArr = ['companyName','fax','firstName','lastName','gender','phoneBusiness','p
commonConstants.PROFILEARRAY = profileArr;
var smCustomAttrMap = [ {key:'smUUID' , value:'id'}, {key:'smProxy_ids',value:'proxy_ids
commonConstants.SFCC_CUSTOM_ATTRIBUTESMAP = smCustomAttrMap;
var smTierCustomAttrMap = [ {key:'smId' , value:'id'}, {key:'smIdentifier',value:'identi
commonConstants.SFCC_TIER_MAP = smTierCustomAttrMap;
//Cartridge Names
commonConstants.SG_CORE = 'app_sitegenesis_core';
commonConstants.SG_PIPELINE = 'app_sitegenesis_pipelines';
commonConstants.SG_CONTROLLER = 'app_sitegenesis_controllers';
//SM Cartridge Names
commonConstants.SM_CONTROLLER = 'int_sessionm_controllers';
commonConstants.SM_CORE = 'int_sessionm_core';
commonConstants.SM_PIPELINE = 'int_sessionm_pipelines';

//Folder Paths
commonConstants.PATH_PIPELET = commonConstants.SM_CORE+'/cartridge/scripts/pipelets/';
commonConstants.PATH_ADAPTOR = commonConstants.SM_CORE+'/cartridge/scripts/adaptor/';
commonConstants.PATH_COMMON = commonConstants.SM_CORE+'/cartridge/scripts/common/';
commonConstants.PATH_FACADE = commonConstants.SM_CORE+'/cartridge/scripts/facade/';
commonConstants.PATH_JOBS = commonConstants.SM_CORE+'/cartridge/scripts/jobs/';
commonConstants.PATH_HELPER = commonConstants.SM_CORE+'/cartridge/scripts/helper/';
```

3.4.1 Generic Section

Core - billing.js (compiled to app.js)

Update “exports.init” Function

Add below code to add reward redemption div.

```
var selectedPaymentMethod = $selectPaymentMethod.find(':checked').val();
    var $redeemRewards  = $('#redeempoints');
    var $removeRewards  = $('#removepoints');
```

Add new redeem Rewards click function just after creditCardList on change function to display Reward page.

```
// select credit card from list
    $('#creditCardList').on('change', function () {
        var cardUUID = $(this).val();
        if (!cardUUID) {return;}
        populateCreditCardForm(cardUUID);

        // remove server side error
        $('.required.error').removeClass('error');
        $('.error-message').remove();
    });

    $redeemRewards.on('click',function(e){
       e.preventDefault();
       var redeemableAmount=  $("#rewardAmount").val();
```

```
      var url = util.appendParamsToUrl(Urls.rewardsPoints, {redeemableAmount:
redeemableAmount, format: 'ajax'});
        $.getJSON(url, function (data) {
              var fail = false;
              var msg = '';
              if (!data.success) {
                  msg = Resources.BAD_RESPONSE;
                  fail = true;
                  $('#rewardAmount-error').html('Unbale to redeem given amount,
please try again with other amount');
              } else {
                  window.location.assign(Urls.billing);
              }
          });
    });
       $removeRewards.on('click',function(e){
      e.preventDefault();
      var redeemableAmount=  $("#rewardAmount").val();
      var url = util.appendParamsToUrl(Urls.removeRewardsPromo, {format: 'ajax'});
        $.getJSON(url, function (data) {
              var fail = false;
              var msg = '';
              if (!data.success) {
                  msg = Resources.BAD_RESPONSE;
                  fail = true;
              } else {
                  window.location.assign(Urls.billing);
              }
          });
    });
   // SM redemption
    $('#rewardAmount').on('change',function(e){
      var amount = $("#rewardAmount").val();
      if(amount%1!==0){
            var value = $(this).val();
            amount =parseFloat(value).toFixed(2);
            $(this).val(amount);
      }
      var dollerValue = SitePreferences.SM_AMOUNT_VALUE;
      var total = Math.round((amount/dollerValue) * 100) / 100;
      $('#pointslabel').html(total);
});
```

Update “exports.init” Function

Add below code to add offer purchase and offer redemption and cancel applied coupon div.

```
$('body').on('click', 'button.offer-redeem', function (e) {
        e.preventDefault();
        var offerId = $(this).attr('id');
        var name = $(this).attr('name');
        var offer = { 'reference_id': offerId, 'name': name };
        var url = util.appendParamsToUrl(Urls.redeemOffer, { offerId: offerId,
```

```
format: 'ajax' });
        $.getJSON(url, function (data) {
            var fail = false;
            var msg = '';
            if (!data.success) {
                msg = Resources.BAD_RESPONSE;
                fail = true;
                alert('error');
            } else {
             window.location.assign(Urls.billing);
            }
        });

    });

    $('body').on('click', 'button.purchase-offer', function (e) {
        e.preventDefault();
        var offerId = $(this).attr('offerId');
        var retailerId = $(this).attr('retailerId');
        console.log('offerId: ' + offerId);
        var name = $(this).attr('name');
        var offer = { 'reference_id': offerId, 'name': name };
        var url = util.appendParamsToUrl(Urls.purchaseAvailableOffer, { offerId:
offerId, retailerId: retailerId, format: 'ajax' });
        $.ajax({
            dataType: 'json',
            url: url,
            async: true
        })
        .done(function (response) {
             if(response.success) {
                    window.location.assign(Urls.billing);
             } else {
                    alert("Error: " + response.result.error);
             }
        });
    });

    $('body').on('click', 'button.cancel-redeem', function (e) {
        e.preventDefault();
        var offerId = $(this).attr('id');
        var url = util.appendParamsToUrl(Urls.cancelRedeem, { offerId: offerId,
format: 'ajax' });
        $.getJSON(url, function (data) {
            var fail = false;
            var msg = '';
            if (!data.success) {
                msg = Resources.BAD_RESPONSE;
                fail = true;
                alert('error');
            } else {
                window.location.assign(Urls.billing);
            }
```

```
        });
        window.location.assign(Urls.billing);
    });
```

Core - Resource.ds

Update "ResourceHelper.getUrls" Function

Add below URL for Reward redemption.

```
csrffailed                  : URLUtils.url('CSRF-Failed').toString(),
        rewardsPoints                       : URLUtils.url('SessionM-
RedeemRewards').toString(),
        removeRewardsPromo              : URLUtils.url('SessionM-
RemoveRewardsPromo').toString(),
purchaseAvailableOffer          : URLUtils.url('SessionM-
PurchaseAvailableOffer').toString(),
```

Update "ResourceHelper.getPreferences" Function

Add below code to get amount configured through BM.

```
CHECK_TLS: Site.getCurrent().getCustomPreferenceValue('checkTLS'),
        SM_AMOUNT_VALUE :  Site.getCurrent().getCustomPreferenceValue('smPointValue')
```

Core – Template

Update "billing.xml"

Add below custom field to use Sales Associate ID field.

```
<group formid="custom">
            <field formid="smOrderSalesAssociateID" type="string"
binding="smOrderSalesAssociateID" mandatory="false" default-value="default"/>
      </group>
```

Update "accountnavigation.isml"

Add below condition to show reward points on UI just after 'account-nav-registered' content assert.

```
<iscontentasset aid="account-nav-registered"/>

      <isif condition="${!empty(customer.profile.custom.smAuthToken) &&
dw.system.Site.current.preferences.custom.IsSMEnabled &&
dw.system.Site.current.preferences.custom.IsSMRewardsPageEnabled &&
pdict.CurrentCustomer.profile.custom.smLoyaltyOptIn}" >
            <span class="toggle">Rewards </span>
            <ul>
                  <li><a title="Rewards" href="${URLUtils.https('SessionM-
ShowRewards')}">Rewards</a></li>
            </ul>
      </isif>
```

Update "accountoverview.isml"

Include below line to display tier details on account overview section just after closing of </h1> tag.

```
<isif condition="${pdict.CurrentCustomer.profile.custom.smLoyaltyOptIn}">
    <isinclude template="account/tierdetails.isml"/>
</isif>
<iscontentasset aid="account-landing"/>
```

Update "paymentmethod.isml"

Include below code 3 times in file to display redeemable points on payment section inside credit card, Bill me later and custom processor block.

Credit Card:-

```
<div class="payment-method <isif condition="${empty(pdict.selectedPaymentID) ||
pdict.selectedPaymentID=='CREDIT_CARD'}">payment-method-expanded</isif>" data-
method="CREDIT_CARD">

                    <iscomment>Template included for SessionM points redemnption on
checkout </iscomment>
                    <isif condition="${customer.authenticated &&
!customer.profile.custom.IsSMProfileSyncFailed &&
customer.profile.custom['smAvailable_points'] > 0 &&
dw.system.Site.getCurrent().getCustomPreferenceValue('IsSMEnabled') &&
dw.system.Site.getCurrent().getCustomPreferenceValue('smRewardsRedemptionEnabled') &&
pdict.CurrentCustomer.profile.custom.smLoyaltyOptIn}">
                        <isinclude template="checkout/redeemablepoints"/>
                    </isif>

                    <iscomment>display select box with stored credit cards if customer
is authenticated</iscomment>
                    <isif condition="${pdict.CurrentCustomer.authenticated &&
!empty(pdict.ApplicableCreditCards)}">
```

PayPal:-

```
<div class="payment-method <isif condition="${!empty(pdict.selectedPaymentID) &&
pdict.selectedPaymentID=='PayPal'}">payment-method-expanded</isif>" data-
method="Custom">
                    <iscomment>Template included for SessionM points redemption on
checkout </iscomment>
                    <isif condition="${customer.authenticated &&
!customer.profile.custom.IsSMProfileSyncFailed &&
customer.profile.custom['smAvailable_points'] > 0 &&
dw.system.Site.getCurrent().getCustomPreferenceValue('IsSMEnabled') &&
dw.system.Site.getCurrent().getCustomPreferenceValue('smRewardsRedemptionEnabled') &&
pdict.CurrentCustomer.profile.custom.smLoyaltyOptIn}">
                        <isinclude template="checkout/redeemablepoints"/>
                    </isif>
                    ${Resource.msg('billing.custompaymentmethod','checkout',null)}
```

BML:-

```
<div class="payment-method <isif condition="${!empty(pdict.selectedPaymentID) &&
pdict.selectedPaymentID=='BML'}">payment-method-expanded</isif>" data-method="BML">
                    <iscomment>Template included for SessionM points redemnption on
checkout </iscomment>
                    <isif condition="${customer.authenticated &&
!customer.profile.custom.IsSMProfileSyncFailed &&
customer.profile.custom['smAvailable_points'] > 0 &&
dw.system.Site.getCurrent().getCustomPreferenceValue('IsSMEnabled') &&
dw.system.Site.getCurrent().getCustomPreferenceValue('smRewardsRedemptionEnabled') &&
pdict.CurrentCustomer.profile.custom.smLoyaltyOptIn}">
                          <isinclude template="checkout/redeemablepoints"/>
                    </isif>

                    <p class="form-
caption">${Resource.msg('billing.bmlhelp','checkout',null)}</p>
```

Include below code 3 times in file to display Sales Associate ID field on payment section:
Credit Card:-

```
<div class="payment-method <isif condition="${empty(pdict.selectedPaymentID) ||
pdict.selectedPaymentID=='CREDIT_CARD'}">payment-method-expanded</isif>" data-
method="CREDIT_CARD">

                    <iscomment>Template included for SessionM points redemnption on
checkout </iscomment>
              <div class="form-row ">
                    <label
for="smOrderSalesAssociateID"><span>${dw.system.Site.getCurrent().getCustomPreference
Value('smOrderSalesAssociateIDFieldName')}</span></label>
                    <div class="field-wrapper">
                          <input class="input-text valid" type="text"
id="dwfrm_billing_custom_smOrderSalesAssociateID"
                          name="dwfrm_billing_custom_smOrderSalesAssociateID"
value="default" aria-invalid="false">
                    </div>
              <div class="form-caption"></div>
              </div>

                    <iscomment>display select box with stored credit cards if
customer is authenticated</iscomment>
                    <isif condition="${pdict.CurrentCustomer.authenticated &&
!empty(pdict.ApplicableCreditCards)}">
```

PayPal:-

```
<div class="payment-method <isif condition="${!empty(pdict.selectedPaymentID) &&
pdict.selectedPaymentID=='PayPal'}">payment-method-expanded</isif>" data-
method="Custom">
                    <div class="form-row ">
                    <label
for="smOrderSalesAssociateID"><span>${dw.system.Site.getCurrent().getCustomPreference
Value('smOrderSalesAssociateIDFieldName')}</span></label>
                    <div class="field-wrapper">
                          <input class="input-text valid" type="text"
id="dwfrm_billing_custom_smOrderSalesAssociateID"
```

```
                        name="dwfrm_billing_custom_smOrderSalesAssociateID"
value="default" aria-invalid="false">
                    </div>
                <div class="form-caption"></div>
                </div>
                    ${Resource.msg('billing.custompaymentmethod','checkout',null)}
```

BML:-

```
<div class="payment-method <isif condition="${!empty(pdict.selectedPaymentID) &&
pdict.selectedPaymentID=='BML'}">payment-method-expanded</isif>" data-method="BML">
                    <iscomment>Template included for SessionM points redemnption on
checkout </iscomment>
                    <div class="form-row ">
                    <label
for="smOrderSalesAssociateID"><span>${dw.system.Site.getCurrent().getCustomPreference
Value('smOrderSalesAssociateIDFieldName')}</span></label>
                    <div class="field-wrapper">
                        <input class="input-text valid" type="text"
id="dwfrm_billing_custom_smOrderSalesAssociateID"
                        name="dwfrm_billing_custom_smOrderSalesAssociateID"
value="default" aria-invalid="false">
                    </div>
                <div class="form-caption"></div>
                </div>

                    <p class="form-
caption">${Resource.msg('billing.bmlhelp','checkout',null)}</p>
```

Update "headercustomerinfo.isml"

Add below code inside script for reward display.

```
      // User has session and is validated
            } else if (customer.registered) {
                    title = Resource.msgf('global.user.name', 'locale', null,
customer.profile.firstName, customer.profile.lastName);

                    var isSmRewardEnabled = false;
                    var logoutLink = {
                        href: URLUtils.https('Login-Logout'),
                        title: Resource.msg('global.logout', 'locale', null),
                        class: 'user-logout buttonstyle'
                    };
                    if(dw.system.Site.current.preferences.custom.IsSMEnabled &&
dw.system.Site.current.preferences.custom.IsSMRewardsPageEnabled){
                      isSmRewardEnabled = true;
                    }
```

Remove below Login-logout href code from link section.

```
                }, {
                        href: URLUtils.https('GiftRegistry-Start'),
                        title: Resource.msg('global.header.registrylink',
'locale', null)
```

```
                    }, {
                            href: URLUtils.https('Login-Logout'),
                            title: Resource.msg('global.logout', 'locale', null),
                            class: 'user-logout buttonstyle'
                    }];
            // user has no session
            }
```

Add smReward If Condtion after link section closing.

```
}];

                    if(isSmRewardEnabled &&
!empty(customer.profile.custom.smAuthToken &&
customer.profile.custom.smLoyaltyOptIn)){

                            var customLink = {
                                  href: URLUtils.https('SessionM-ShowRewards'),
                                  title: Resource.msg('global.header.showrewards',
'sessionm', null)
                             };
                            links.push(customLink);
                            links.push(logoutLink);
                     }else{
                            links.push(logoutLink);
                     }



            // user has no session
```

Update "ordertotals.isml"

Apply reward redemption discount by adding below code.

```
<iscomment>calculate order level discounts</iscomment>
                    <isscript>
                            var merchTotalExclOrderDiscounts : dw.value.Money =
LineItemCtnr.getAdjustedMerchandizeTotalPrice(false);
                            var merchTotalInclOrderDiscounts : dw.value.Money =
LineItemCtnr.getAdjustedMerchandizeTotalPrice(true);
                            var orderDiscount : dw.value.Money =
merchTotalExclOrderDiscounts.subtract( merchTotalInclOrderDiscounts );
                            var comconst =
require('int_sessionm_core/cartridge/scripts/common/CommonConstants').CommonConstants
;


                    </isscript>

                    <isset name="rewardDiscount"
value="${LineItemCtnr.getPriceAdjustmentByPromotionID(comconst.REWARDS_POINTS)}"
scope="page" />
                    <isif condition="${!empty(rewardDiscount)}">
                            <tr class="order-discount discount">
```

```
        <td>${Resource.msg('sessionm.ordertotoal.label','sessionm',null)}</td>
                                <td> <isprint value="${rewardDiscount.basePrice}"/></td>
                                </tr>
                                <isif
condition="${orderDiscount.add(rewardDiscount.basePrice) > 0.0}">
                                        <tr class="order-discount discount">

        <td>${Resource.msg('order.ordersummary.orderdiscount','order',null)}</td>
                                                <td>- <isprint
value="${orderDiscount.add(rewardDiscount.basePrice)}"/></td>
                                        </tr>
                                </isif>
                        <iselseif condition="${!empty(orderDiscount) &&
orderDiscount.value > 0.0}">
                                <tr class="order-discount discount">

        <td>${Resource.msg('order.ordersummary.orderdiscount','order',null)}</td>
                                        <td>- <isprint value="${orderDiscount}"/></td>
                                </tr>
                        </isif>



                        <iscomment>render each single shipment or shipping
total</iscomment>
```

Update "summary.isml"

Handle SessionM error messages by applying below If condition PlaceOrderError If condition.

```
<isif condition="${pdict.PlaceOrderError != null}">
                <div class="error-
form">${Resource.msg(pdict.PlaceOrderError.code,'checkout',null)}</div>
        </isif>

        <isif condition="${pdict.SmPLaceOrderError != null}">
                <div class="error-
form">${Resource.msg(pdict.SmPLaceOrderError.code,'sessionm',null)}</div>
        </isif>
```

Update "htmlhead.isml"

Apply Reward and session css on Header page if SM flag is enable form business manager.

```
        <link rel="stylesheet" href="${URLUtils.staticURL('/css/style.css')}" />

<isif condition="${'IsSMEnabled' in dw.system.Site.current.preferences.custom &&
dw.system.Site.current.preferences.custom.IsSMEnabled}">
        <link rel="stylesheet"
href="${dw.system.Site.current.preferences.custom.smRewardPageCSS}" />
        <link rel="stylesheet" href="${URLUtils.staticURL('/css/sessionm.css')}" />
</isif>
```

3.4.2 Custom Code for Controller

Account.js

Update "editForm" Function

Add below code inside confirm function 'isProfileUpdateValid' condition to update customer profile in SessionM.

```
            if(!dw.system.Site.current.preferences.custom.smLoyaltyOptIn){
                app.getForm('profile.custom.smLoyaltyOptIn').object.value = true;
            }

             if (isProfileUpdateValid) {
                hasEditSucceeded =
Customer.editAccount(app.getForm('profile.customer.email').value(),
app.getForm('profile.login.password').value(),
app.getForm('profile.login.password').value(), app.getForm('profile'));
                if (!hasEditSucceeded) {
                    app.getForm('profile.login.password').invalidate();
                    isProfileUpdateValid = false;
                }else{

require('int_sessionm_controllers/cartridge/controllers/SessionM').UpdateCustomer();
                }
             }
```

Add below code inside changepassword function 'isProfileUpdateValid' condition to update customer profile in SessionM.

```
            if(!dw.system.Site.current.preferences.custom.smLoyaltyOptIn){
                app.getForm('profile.custom.smLoyaltyOptIn').object.value = true;
            }

             if (isProfileUpdateValid) {
                hasEditSucceeded =
Customer.editAccount(app.getForm('profile.customer.email').value(),
app.getForm('profile.login.newpassword').value(),
app.getForm('profile.login.currentpassword').value(), app.getForm('profile'));
                if (!hasEditSucceeded) {
                    app.getForm('profile.login.currentpassword').invalidate();
                }else{

             }
        }
```

Update "registrationForm" Function

Register customer in SessionM after successful Login to SFCC by adding below code inside 'profileValidation' TRUE condition.

```
if (profileValidation) {
                 profileValidation = Customer.createAccount(email, password,
```

```
app.getForm('profile'));

                if (orderNo) {
                    var orders = OrderMgr.searchOrders('orderNo={0} AND status!={1}',
'creationDate desc', orderNo,
                            dw.order.Order.ORDER_STATUS_REPLACED);
                    if (orders) {
                        var foundOrder = orders.next();
                        Transaction.wrap(function(){
                            foundOrder.customer = profileValidation;
                        })
                        session.custom.TargetLocation = URLUtils.https('Account-
Show','Registration','true');
                    }
                }
                // Integration of SM Customer Profile creation

require('int_sessionm_controllers/cartridge/controllers/SessionM').RegisterCustomer()
;
            }
```

Address.js

Update "handleForm" Function

Update Customer details as Address in SessionM by adding below code at the end of function.

```
if (request.httpParameterMap.format.stringValue === 'ajax') {
        let r = require('~/cartridge/scripts/util/Response');

require('int_sessionm_controllers/cartridge/controllers/SessionM').UpdateCustomer();
        r.renderJSON({
            success: success,
            message: message
        });
        return;
    }
```

Update "setDefault" Function

Update Customer details as Address in SessionM by adding below code at the end of function.

```
    response.redirect(URLUtils.https('Address-List'));

require('int_sessionm_controllers/cartridge/controllers/SessionM').UpdateCustomer();
}
```

Update "Delete" Function

Update Customer details as Address in SessionM by adding below code after defining 'r' variable.

```
    let r = require('~/cartridge/scripts/util/Response');
```

```
require('int_sessionm_controllers/cartridge/controllers/SessionM').UpdateCustomer();

    r.renderJSON({
        status: deleteAddressResult ? 'OK' :
CustomerStatusCodes.CUSTOMER_ADDRESS_REFERENCED_BY_PRODUCT_LIST,
        message: deleteAddressResult ? '' : Resource.msg('addressdetails.' +
CustomerStatusCodes.CUSTOMER_ADDRESS_REFERENCED_BY_PRODUCT_LIST, 'account', null)
    });
}
```

Cart.js

Update "submitForm" Function

Update Reward point details after removing item from Cart by adding below code inside 'deleteProduct' function.

```
'deleteProduct': function (formgroup) {
            Transaction.wrap(function () {
                cart.removeProductLineItem(formgroup.getTriggeredAction().object);

require('int_sessionm_core/cartridge/scripts/adaptor/Rewards').UpdateCartRewardPoints
(cart);
            });
```

Update Reward point details after editing cart items by adding below code inside 'editLineItem' function.

```
'editLineItem': function (formgroup) {
            var product, productOptionModel;
            product =
app.getModel('Product').get(request.httpParameterMap.pid.stringValue).object;
            productOptionModel =
product.updateOptionSelection(request.httpParameterMap);

            Transaction.wrap(function () {
                cart.updateLineItem(formgroup.getTriggeredAction().object, product,
request.httpParameterMap.Quantity.doubleValue, productOptionModel);

require('int_sessionm_core/cartridge/scripts/adaptor/Rewards').UpdateCartRewardPoints
(cart);
                cart.calculate();
            });

            ISML.renderTemplate('checkout/cart/refreshcart');
            return null;
        },
```

Update Reward point details after updating item from Cart by adding below code inside 'updateCart' function.

```
'updateCart': function () {
```

```
            Transaction.wrap(function () {
                var shipmentItem, item;

                // remove zero quantity line items
                for (var i = 0; i < session.forms.cart.shipments.childCount; i++) {
                    shipmentItem = session.forms.cart.shipments[i];

                    for (var j = 0; j < shipmentItem.items.childCount; j++) {
                        item = shipmentItem.items[j];

                        if (item.quantity.value === 0) {
                            cart.removeProductLineItem(item.object);
                        }
                    }
                }

require('int_sessionm_core/cartridge/scripts/adaptor/Rewards').UpdateCartRewardPoints
(cart);
                session.forms.cart.shipments.accept();
                cart.checkInStoreProducts();
            });

            return {
                cart: cart,
                EnableCheckout: true
            };
        },
```

COBilling.js

Update "publicStart" Function

Update Customer Reward points if customer is authenticated and reward redemption flag is TRUE in BM by add below code above couponcode and giftcert clear function.

```
// Update customer reward points if SessionM reward redemption is enabled.
        if( customer.authenticated &&
dw.system.Site.getCurrent().getCustomPreferenceValue('IsSMEnabled') &&
dw.system.Site.getCurrent().getCustomPreferenceValue('smRewardsRedemptionEnabled')){

      require('int_sessionm_controllers/cartridge/controllers/SessionM').UpdateRewar
dPoints();
        }

        app.getForm('billing.couponCode').clear();
        app.getForm('billing.giftCertCode').clear();

        start(cart, {ApplicableCreditCards: creditCardList.ApplicableCreditCards});
    } else {
        app.getController('Cart').Show();
    }
```

Update "billing" Function

Update Customer details in SessionM after editing address on billing page by adding below code under save function and inside customer.authenticated codition.

```
if (customer.authenticated &&
app.getForm('billing').object.billingAddress.addToAddressBook.value) {

app.getModel('Profile').get(customer.profile).addAddressToAddressBook(cart.getBilling
Address());
                    // Integration of SM Customer Profile updation

require('int_sessionm_controllers/cartridge/controllers/SessionM').UpdateCustomer();

                }
```

COPlaceOrder.js

Update "start" Function

Update Reward point on Summary page and handle error by adding below lines after create order and before the handle.

```
//update customer reward points
    var ComConst =
require('int_sessionm_core/cartridge/scripts/common/CommonConstants').CommonConstants
;
    var adjustment = order.getPriceAdjustmentByPromotionID(ComConst.REWARDS_POINTS);
if( !empty(adjustment) && customer.authenticated &&
dw.system.Site.getCurrent().getCustomPreferenceValue('IsSMEnabled') &&
dw.system.Site.getCurrent().getCustomPreferenceValue('smRewardsRedemptionEnabled')){
      var result =
require('int_sessionm_controllers/cartridge/controllers/SessionM').UpdateRewardPoints
();
      if(empty(result) || empty(result.responseObj)){
            OrderMgr.failOrder(order);
           return {
               error: true,
               smError : true,
               PlaceOrderError: new Status(Status.ERROR,
'confirm.error.sessionm.pointsync')
           };
       }
    }
```

To sends to redemption details to SessionM add below code after handle payment and before Place order

```
    // Calls the SessionM rewards redemption service
    if(!empty(adjustment) && customer.authenticated &&
dw.system.Site.getCurrent().getCustomPreferenceValue('IsSMEnabled') &&
dw.system.Site.getCurrent().getCustomPreferenceValue('smRewardsRedemptionEnabled')){
       var result =
require('int_sessionm_controllers/cartridge/controllers/SessionM').RedeemOrder(order)
```

```
;
        if(result.error == true){
            return {
                error: true,
                smError : true,
                PlaceOrderError: new Status(Status.ERROR,
'confirm.error.sessionm.redemption')
            };
        }
    }

    var orderPlacementStatus = Order.submit(order);
    if (!orderPlacementStatus.error) {
        clearForms();
    }
```

To save Sales Associate ID field on storefront while user places order, edit start() function and add following code below `var order = cart.createOrder();`

```
Transaction.wrap(function () {
             order.custom.smOrderSalesAssociateID =
app.getForm('billing.custom.smOrderSalesAssociateID').value();
    });
```

COShipping.js

Update "singleShipping" Function

Update Customer details in SessionM after updating address on shipping page by adding below code after 'handleShippingSettings' function and inside customer.authenticated condition.

```
handleShippingSettings(cart);

            // Attempts to save the used shipping address in the customer address
book.
            if (customer.authenticated &&
session.forms.singleshipping.shippingAddress.addToAddressBook.value) {

app.getModel('Profile').get(customer.profile).addAddressToAddressBook(cart.getDefault
Shipment().getShippingAddress());
                // Integration of SM Customer Profile updation

require('int_sessionm_controllers/cartridge/controllers/SessionM').UpdateCustomer();

            }
```

COSummary.js

Update "submit" Function

Handle SessionM error by adding below code insidePlaceOrder.error IF condition

```
var placeOrderResult = app.getController('COPlaceOrder').Start();
```

```
    if (placeOrderResult.error) {
       if(placeOrderResult.smError){
              start({
                    SmPLaceOrderError: placeOrderResult.PlaceOrderError
               });
       }else{
               start({
                    PlaceOrderError: placeOrderResult.PlaceOrderError
               });
       }
    } else if (placeOrderResult.order_created) {
         showConfirmation(placeOrderResult.Order);
    }
```

Login.js

Update "handleLoginForm" Function

Retrieve Customer details from SessionM after successful Login by adding below code.

```
} else {

       require('int_sessionm_controllers/cartridge/controllers/SessionM').RetrieveCus
tomer();
                loginForm.clear();
            }
```

CustomerModel.js

2.1.2.1.1 Update "editAccount" Function

Commit the smLoyaltyOptIn option value from form fields into the data base

```
    if (setCustomerPassword.error || !CustomerModel.setLogin(customer, email, password) ||
!Form.get('profile.customer').copyTo(customer.profile)) {
        Transaction.rollback();
        return false;
    }

    customer.profile.custom['smLoyaltyOptIn'] =
Form.get('profile.custom.smLoyaltyOptIn').value();
```

4 OPERATION, MAINTENANCE

4.1 DATA STORAGE

4.1.1 Customer Profile level attribute

Some additional attributes are defined and stored in Profile object. These are the custom attributes and will be sent as a part of Profile request object, so that the same can be used by SessionM for further processing.

Sr. No.	Additional Custom Fields	Attribute Id	Description
1	SM available Reward Points	smAvailablePoints	This field captures the available Reward points for Customer
2	SessionM Status of customer's account	smAccountStatus	Account status returned by SM
3	SM suspend state	smSuspended	True or False
4	SessionM Tier Name	smTierName	SM available Tier for Customer
5	SessionM Identifier for customer	smUUID	SessionM Identifier for customer
6	Is SM Create Profile failed	IsSMProfileCreationFailed	Set to "FALSE" if profile is not created in SM
7	SM Search Fail Count	smSearchFailCount	Set To TRUE if customer not found in SM
8	SessionM Proxy IDs	smProxy_ids	Profile created other than SFCC
9	Is SM Profile Sync Failed	IsSMProfileSyncFailed	If profile is not synced with SFCC
10	SM Auth Token	smAuthToken	SM authentication token
11	SessionM member since	smMember_Since	Set foe new SM Customer profile
12	SessionM required points for next tier level	smNextRequiredPoints	Points required to reach next tier level
13	SessionM Loyalty Opt In	smLoyaltyOptIn	Set to TRUE if the customer is opted in
14	SessionM Offers Available To Purchase	smOffersAvailableToPurchase	Captures the offers available in SessionM

4.1.2 Order level attribute

Some additional attributes are defined and stored in Order object. These are the custom attributes and will be sent as a part of Order request object, so that the same can be used by SessionM for further processing.

Sr. No.	Additional Custom Fields	Attribute Id	Type
1	Order has been exported to SessionM	smExported	Boolean
2	SM Redeemed Points	smOrderCustom1	String

3	SM Order Transaction ID	smOrderCustom2	String
4	SM Order Offer Id	smOrderCustom3	String
5	SM Order custom attribute	smOrderCustom4	String
6	SM Order custom attribute	smOrderCustom5	String
7	Order has to be redeemed by SessionM	smOrderTobeRedeemed	Boolean
8	Sales Associate ID	smOrderSalesAssociateID	String
9	SessionM Offers	SessionM_Offers	String
10	SessionM TrasactionID	SessionM_TrasactionID	String

4.1.2 Basket level attribute

Some additional attributes are defined and stored in Basket object. These are the custom attributes and will be sent as a part of Basket request object, so that the same can be used by SessionM for further processing.

Sr. No.	Additional Custom Fields	Attribute Id	Type
1	SessionM Offers	SessionM_Offers	String
2	SessionM TrasactionID	SessionM_TrasactionID	String

4.2 AVAILIBILITY

<EXPECTED AVAILABILITY /UPTIME OF ANY EXTERNAL SERVICE, INTERFACES>

<FALLBACK SOLUTION, BEHAVIOR IF EXTERNAL SERVICES ARE NOT AVAILABLE, IMPACT ON CUSTOMER STOREFRONT>

<ANY EXISTING UTILITIES THAT HELP TO DETECT AVAILABILITY/UPTIME OF EXTERNAL SERVICE, E.G. WEBSERVICE CALL, GOMEZ PING>

<ESTIMATED PERFORMANCE METRICS FOR PEAK BUSINESS HOURS IF AVAILABLE>

<NOTIFICATION PROCESS IF EXTERNAL SERVICES, INTERFACES ARE NOT RESPONDING, E.G. HOTLINE /SUPPORT PHONE NUMBER>

4.3 SUPPORT

Name	Email	Support Type

5 USER GUIDE

5.1 ROLES, RESPONSIBILITES

Typically most of the integration works is done by the backend developer. We expect that the person doing this integration is familiar with the web service, xml processing and has hands on experience with the Demandware platform.

5.2 BUSINESS MANAGER

NA [As changes required for storefront has been already covered under setup section]

5.3 STOREFRONT FUNCTIONALITY

NA [As changes required for storefront has been already covered under setup section]

6 TESTING

6.1 TEST CASES

Please refer the use case section for the high level cases to be covered.

7 RELEASE HISTORY

Version	Date	Changes
17.0	19-05-2017	Initial Release – SiteGenesis compatible
18.1	15-08-2018	Bug fix with loyalty details

APPENDIX A: Error Codes & Error Messages

Code	Description
200	Indicates successful call; returned string can be either ok or error
401	Indicates unsuccessful call; not authorized.

404	Indicates unsuccessful call; not found by ID
500	Indicates internal error associated with SessionM