

SFRA (Storefront Reference Architecture)

Version 20.1.2

SFRA 5.0.1

PEPPERJAM-SALESFORCE COMMERCE CLOUD INTEGRATION

08.10.21

Presented by

pepperjam[®]

Contents

COMPONENT OVERVIEW	3
FUNCTIONAL OVERVIEW	3
Limitations, Constraints	3
Compatibility	3
PRIVACY, PAYMENT	4
PRE-WORK	4
Implementation Guide	4
GENERAL IMPLEMENTATION: INSTALLING THE CARTRIDGE ON THE SANDBOX	4
SET UP BUSINESS MANAGER	6
MANAGE SITES SETTINGS	6
IMPORT AND EXPORT SETTINGS	7
System Object Type Extensions	7
Configure Jobs.xml for Product Feed and Order Correction Feed	8
CONFIGURE ORDER CORRECTION FEED (OPTIONAL)	9
CONFIGURE CUSTOM SITE PREFERENCES FOR PEPPERJAM	10
INTEGRATE PEPPERJAM SETTINGS INTO YOUR STOREFRONT	11
PEPPERJAM DYNAMIC ADVERTISER PIXEL	11
PEPPERJAM GATEWAY ADVERTISER PIXEL	12
PEPPERJAM TAG CONTAINER	13
REFERENCE: FEED JOBS RUN	13
TESTING	14
Availability	14
Support	14

Summary

This document provides technical instructions for using the Pepperjam Salesforce Commerce Cloud Cartridge to integrate the Pepperjam Dynamic Advertiser Pixel, Gateway Advertiser and Pixel Tag Container into a Salesforce Commerce Cloud storefront. It also provides access to Advanced Product Feed and Sales Correction Feed for the Pepperjam service.

Glossary

Term	Description
SFCC	Salesforce Commerce Cloud
Business Manager (BM)	The primary tool used to configure the SFCC platform and customer storefront
SFRA	Storefront Reference Architecture

Version

Version	Last Updated	Notes
1.0	8/13/2018	Initial version
20.1.0	10/12/2020	SFRA 5.0.1 adaptation
20.1.1	07/27/2021	Added Custom Tracking Domain
20.1.2	08/10/2021	Updated Integration Guide

1 Component Overview

1.1 FUNCTIONAL OVERVIEW

The Pepperjam cartridge enables a merchant to set the Pepperjam Dynamic Advertiser Pixel, Gateway Advertiser and Pixel Tag Container in SFCC and manage each by selecting enable or disable. The cartridge also generates a product feed. Additionally, an optional order correction feed is generated that shows orders with corrections (like returns).

1.2 LIMITATIONS, CONSTRAINTS

pepperjam

The cartridge is designed for the US locale, but can work with other locales.

1.3 COMPATIBILITY

The cartridge is designed for Salesforce Commerce Cloud API version 20.9 with controllers. Compatibility Mode: 19.10. The cartridge was developed against SFRA version 5.0.1.

The cartridge utilizes the Pepperjam Dynamic Pixel.

1.4 PRIVACY, PAYMENT

The Pepperjam cartridge does not collect and process any user profile information or billing information.

2 Pre-work

Use of the Pepperjam SFCC Cartridge requires credentials from Pepperjam. Please contact your Pepperjam account team or launch team representative to receive:

- Pepperjam Program ID
- Pepperjam Tag ID

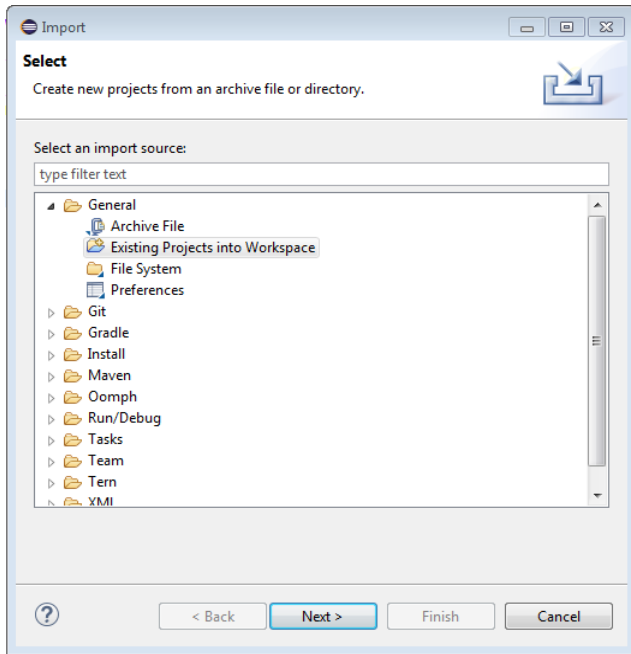
From your internal team:

- To use Custom Tracking Domain functionality, you will need to complete the setup wizard in Ascend as well as make changes with your domain registrar for the specific subdomain. More information about configuring Custom Tracking Domains is available [here](#).
- Optional: If you are setting up the order correction feed, you'll need an FTP site that Pepperjam can pick the feed up from. Therefore, gather these FTP credentials prior to starting.

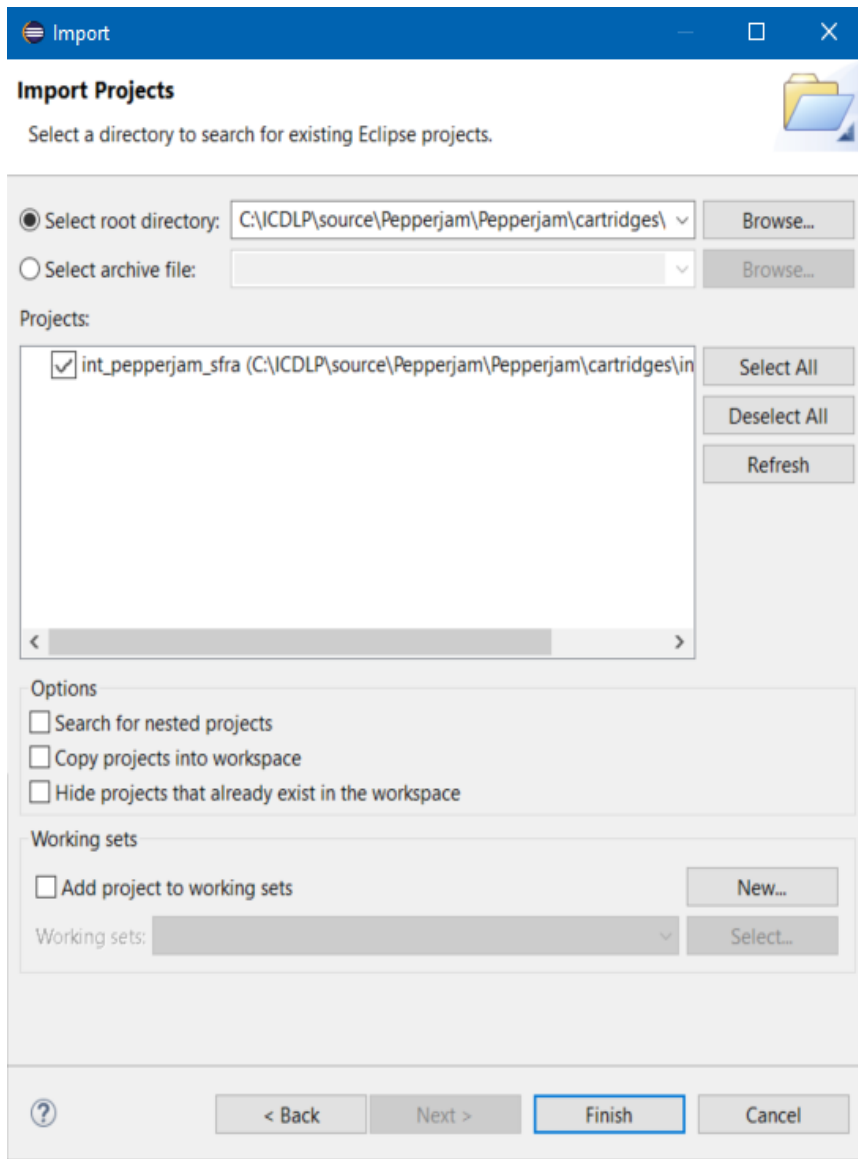
3 Implementation Guide

3.1 GENERAL IMPLEMENTATION: INSTALLING THE CARTRIDGE ON THE SANDBOX

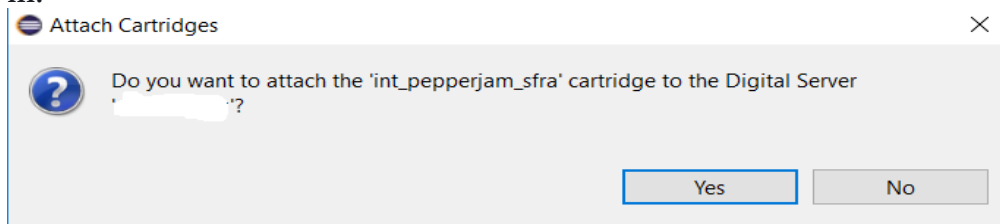
1. Open UX Studio.
2. Import the downloaded cartridge. On the import screen, select: Existing Projects into Workspace



3. Link the cartridge to the sandbox



4. Select sandbox connection, then Properties. Select Project Reference and check in.



3.2 SET UP BUSINESS MANAGER

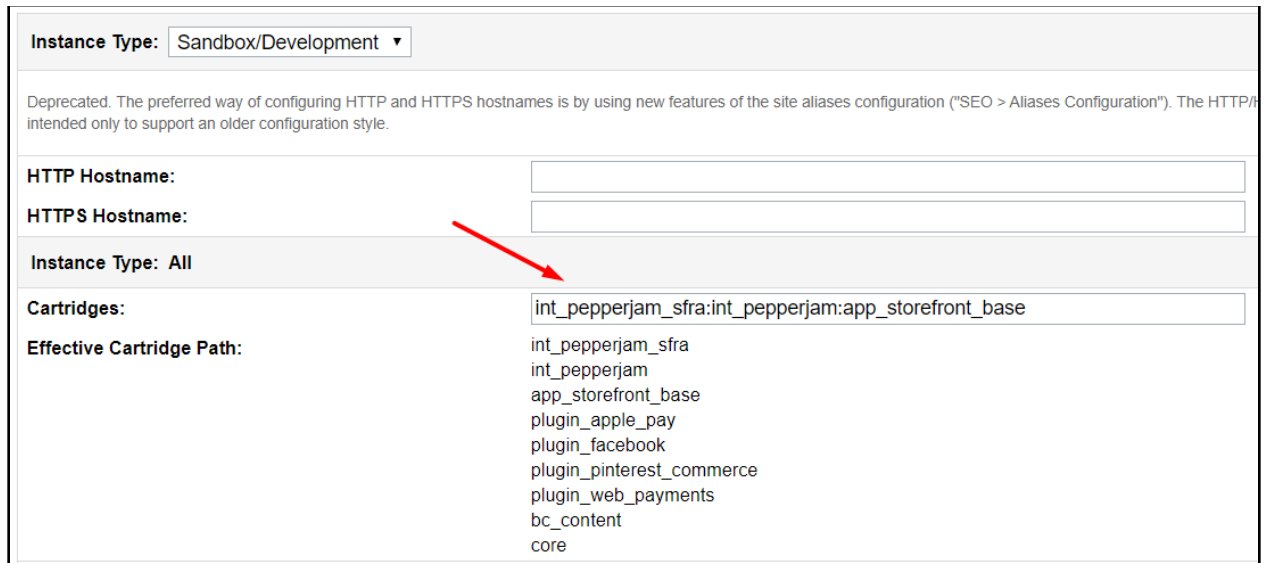
3.2.1 Manage Sites Settings

Go to Business Manager -> Site -> Manage Sites. Select the correct site, then

pepperjam

select **Settings** tab. In the cartridge path add the following line:

```
int_pepperjam_sfra:app_storefront_base:int_pepperjam
```

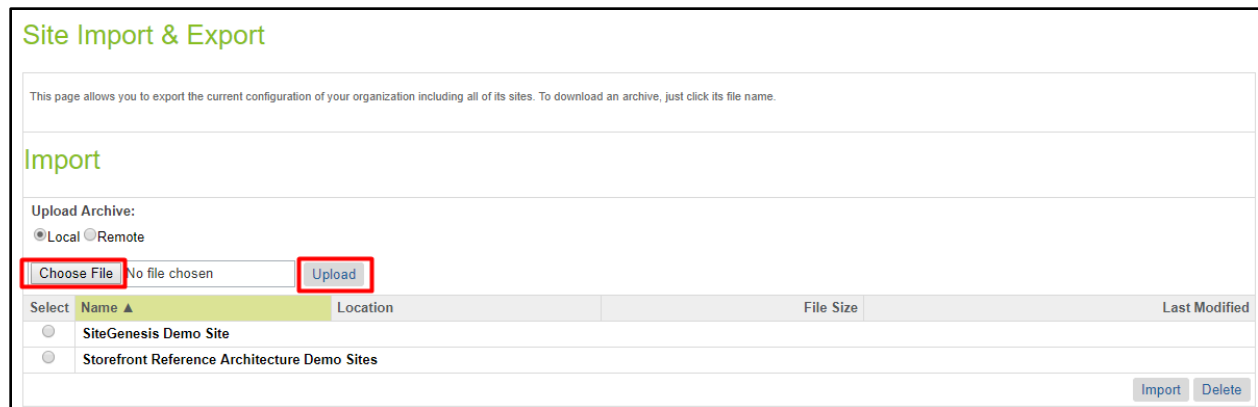


The screenshot shows a configuration page for an instance. At the top, the Instance Type is set to 'Sandbox/Development'. Below this, there is a deprecation notice about HTTP and HTTPS hostnames. The HTTP and HTTPS Hostname fields are empty. The Instance Type is set to 'All'. The Cartridges field contains the path 'int_pepperjam_sfra:int_pepperjam:app_storefront_base', which is highlighted with a red arrow. The Effective Cartridge Path lists several cartridges: int_pepperjam_sfra, int_pepperjam, app_storefront_base, plugin_apple_pay, plugin_facebook, plugin_pinterest_commerce, plugin_web_payments, bc_content, and core.

3.2.2 Import and Export Settings

3.2.2.1 System Object Type Extensions

Go to Administration -> Site Development -> Site Import & Export. Upload system-objecttype-extensions.xml Pepperjam.zip from the metadata folder.



The screenshot shows the 'Site Import & Export' page. It has a heading 'Import' and a sub-heading 'Upload Archive:'. There are two radio buttons: 'Local' (selected) and 'Remote'. Below this is a file selection area with a 'Choose File' button (highlighted with a red box) and a text field 'No file chosen'. To the right of the text field is an 'Upload' button (also highlighted with a red box). Below the file selection area is a table with columns: 'Select', 'Name', 'Location', 'File Size', and 'Last Modified'. The table contains two rows: 'SiteGenesis Demo Site' and 'Storefront Reference Architecture Demo Sites'. At the bottom right of the table are 'Import' and 'Delete' buttons.

Click on the import button to follow steps to import uploaded settings system-objecttype-extensions.xmlPepperjam.zip

Select	Name ▲	Location	File Size	Last Modified
<input checked="" type="radio"/>	Pepperjam.zip	local	2.46 KB	8/30/19 4:59:35 am
<input type="radio"/>	SiteGenesis Demo Site			
<input type="radio"/>	Storefront Reference Architecture Demo Sites			

3.2.2.2 Configure Jobs.xml for Product Feed and Order Correction Feed

Reference: To get the link for the HTTP GET access to the product feed, add "Pepperjam-GetProductsExport" to your main site address as shown below

The screenshot shows a browser address bar with a URL: `...alliance-prtnr...dw.demandware.net/on/demandware.store/Sites- SITE-ID -Site/default/Pepperjam-GetProductsExport`. The text 'Pepperjam-GetProductsExport' is highlighted with a red rectangular box.

In the metadata folder in the file jobs.xml you need to define your siteID for the Product Feed and Order Corrections Feed jobs as shown below (NOTE: The Order Corrections Feed is Optional).

Product Feed:

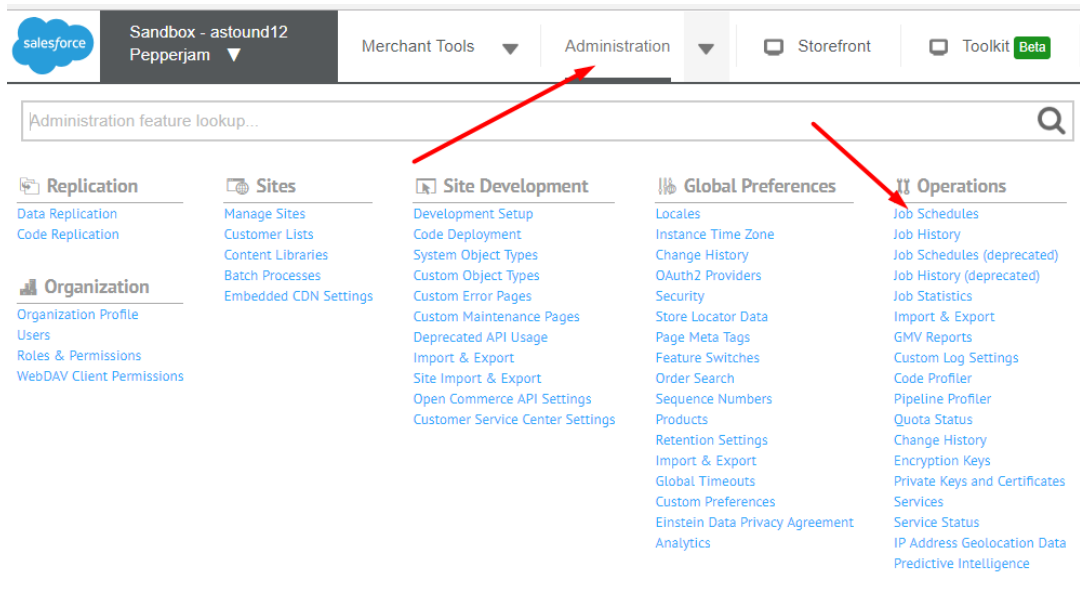
```
<?xml version="1.0" encoding="ISO-8859-1"?>
- <jobs xmlns="http://www.demandware.com/xml/impex/jobs/2015-07-01">
  - <job priority="0" job-id="Pepperjam-product_feed">
    <description>creates csv file with all site products</description>
    <parameters/>
    - <flow>
      <context site-id="Pepperjam"/>
      - <step enforce-restart="false" type="ExecuteScriptModule" step-id="Pepperjam_product_feed">
        <description/>
        - <parameters>
          <parameter name="ExecuteScriptModule.Module">int_pepperjam/cartridge/scripts/job/product_feed.js</parameter>
          <parameter name="ExecuteScriptModule.FunctionName">execute</parameter>
          <parameter name="ExecuteScriptModule.Transactionnal">>false</parameter>
        </parameters>
      </step>
    </flow>
    <rules/>
  - <triggers>
    - <run-once enabled="false">
      <date>2018-05-18Z</date>
      <time>10:19:40.000Z</time>
    </run-once>
  </triggers>
</job>
```

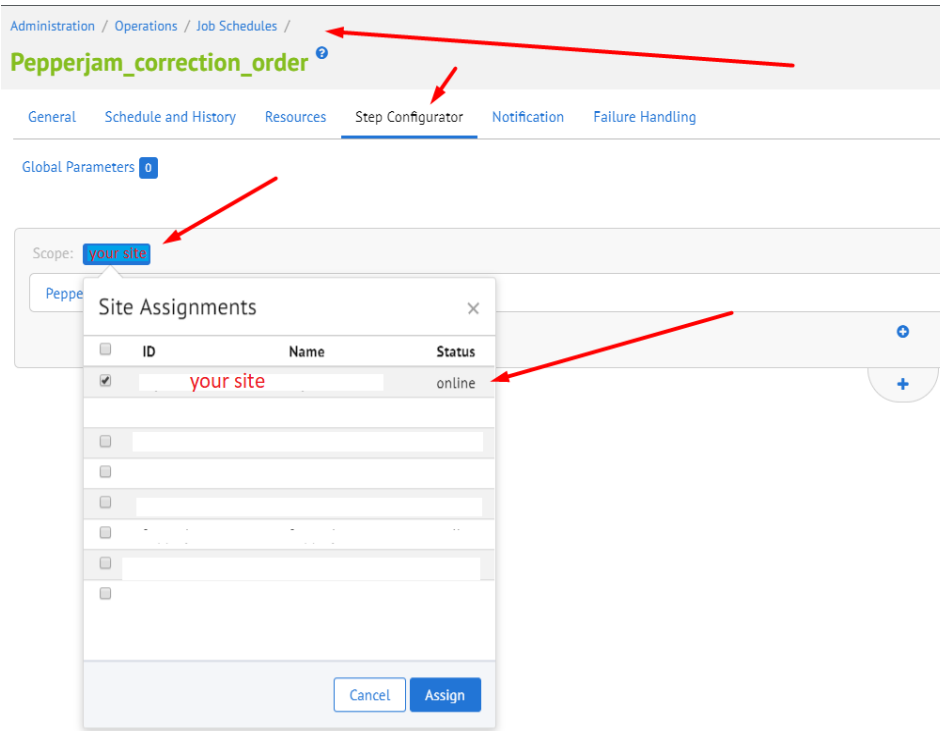
Order Feed (Optional – see section 3.2.3):

pepperjam

```
- <job priority="0" job-id="Pepperjam_correction_order">
  <description>creates csv file with order info, that will be available through ftp</description>
  <parameters/>
  - <flow>
    <context site-id="Pepperjam"/>
    - <step enforce-restart="false" type="ExecuteScriptModule" step-id="Pepperjam_correction_order">
      <description/>
      - <parameters>
        - <parameter name="ExecuteScriptModule.Module">int_pepperjam/cartridge/scripts/job/order_corrections.js</parameter>
        - <parameter name="ExecuteScriptModule.FunctionName">execute</parameter>
        - <parameter name="ExecuteScriptModule.Transactional">false</parameter>
      </parameters>
    </step>
  </flow>
  <rules/>
  - <triggers>
    - <run-once enabled="false">
      <date>2018-05-24Z</date>
      <time>08:54:44.000Z</time>
    </run-once>
  </triggers>
</job>
</jobs>
```

Or you can manually configure these settings in the Job Schedules section within BM after the import of jobs.xml as shown below:





3.2.3 Configure Order Correction Feed (OPTIONAL)

Pepperjam can receive order corrections such as full returns/cancellations to automatic reflect in the network within the transaction locking period to ensure advertisers do not pay commissions on orders that customers returned or were cancelled. The Pepperjam-Salesforce Commerce Cloud cartridge is able to process full order corrections, but not partial order corrections.

NOTE: If you use the Salesforce Commerce Cloud’s Order Management System (called SFCC OMS), then follow the steps below to configure the order correction feed for Pepperjam. For other order management systems, contact your Pepperjam account strategist or Pepperjam launch team representative for more information.

Before beginning this section, make sure you have your FTP credentials ready. In Business Manager go to Administration —> Operations —>Services then choose tab credentials:

Service Credentials

Select All	Name	URL	User
<input type="checkbox"/>			
<input type="checkbox"/>	pepperjamFTP		
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			

Choose 'pepperjamFTP' and enter your credentials. Press save.

pepperjamFTP

Fields with a red asterisk (*) are mandatory. Click Apply to save the details. Click Reset to revert to the last saved state. These credentials are used by 1 service.

Name: pepperjamFTP
 URL:
 User:
 Password: *****

Apply Reset

<< Back to List

3.2.4 Configure Custom Site Preferences for Pepperjam

Go to Business Manager: Sites -> Your Site -> Site Preferences -> Custom Site Preferences -> Pepperjam.

Name	Description
Enable Pepperjam	Turns on\off the Pepperjam integration. Set this value to Yes.
Pepperjam ID	Pepperjam Program ID (PID). You can get this from your Pepperjam account representative or Pepperjam launch team member.
Pepperjam Tag ID	Tag ID for the Pepperjam Pixel Tag Container which is provided by Pepperjam. You can get this from your Pepperjam account representative or Pepperjam launch team member.
Enable Custom Tracking Domains	Turns on\off the use of a Custom Tracking Domain. If you have completed the Custom Tracking Domain wizard in Ascend, set this value to Yes.
Pepperjam Custom Domain URL	Domain name for Pepperjam Tag and Conversion Pixel. This is the URL that was provisioned as part of completing the Custom Tracking domain wizard. Example: pepperjam.my-domain.com.

Enable Pepperjam Tag	Turns on\off the Pepperjam Pixel Tag Container. This should always be set to Yes
Enable Pepperjam Gateway Advertiser	Turns on\off the Pepperjam Gateway Advertiser. This should always be set to Yes.

Below is an example of how to complete the fields in the administration area. Press save when you are done.

The screenshot shows the Pepperjam administration interface. At the top, there are navigation links: Merchant Tools / Site Preferences / Custom Site Preference Groups. The Pepperjam logo is on the left, and there are buttons for Cancel, Apply to Other Sites, and Save on the right. Below the navigation is a dropdown for Instance Type (Sandbox). A search bar is present with the text 'Search by ID'. A table lists configuration items with columns for Name, Value, and Default Value. The items are: Enable Pepperjam (Yes), Pepperjam ID (0000), Pepperjam Tag ID (00000000), Enable Custom Tracking Domains (Yes), Pepperjam Custom Domain URL (pepperjam.my-domain.com), Enable Pepperjam Tag (No), and Enable Pepperjam Gateway Advertiser (No). Each item has an 'Edit Across Sites' link.

Name	Value	Default Value
Enable Pepperjam (Pepperjam_Enable) Enable/disable all Pepperjam functionality	Yes	No
Pepperjam ID (Pepperjam_ProgramId) (String) Authorization key for Pepperjam	0000	
Pepperjam Tag ID (Pepperjam_TagId) (String) Tag Container Identifier	00000000	
Enable Custom Tracking Domains (Pepperjam_CustomTracking) Enable/disable Custom Tracking Domains	Yes	No
Pepperjam Custom Domain URL (Pepperjam_CustomUrl) (String) Custom Tracking Domains URL	pepperjam.my-domain.com	
Enable Pepperjam Tag (Pepperjam_EnableTag) Allows Pepperjam Tag container to be rendered	No	No
Enable Pepperjam Gateway Advertiser (Pepperjam_EnableGatewayAdvertiser) Allows Gateway Advertiser Logic (using cookies) to be used	No	No

3.3 INTEGRATE PEPPERJAM SETTINGS INTO YOUR STOREFRONT

To integrate Pepperjam into your storefront you need to work with a developer on your team to insert code into specific templates in the core cartridge for:

- Pepperjam Dynamic Advertiser Pixel
- Pepperjam Gateway Advertiser Pixel
- Pepperjam Tag Container

3.3.1 Pepperjam Dynamic Advertiser Pixel

For the Dynamic Advertiser Pixel, insert the code below into the checkout/confirmation/confirmation.isml template

```
<include url="{URLUtils.url('Pepperjam-RenderPixelOrder', 'order',
```

pepperjam

pdict.Order.getCurrentOrderNo()"/>

```
it displays the order related information, such as the order number,
creation date, payment information, order totals and shipments of
the order.
</iscomment>

<div class="confirmation <isif condition="${!pdict.CurrentCustomer.authenticated}">create-account</isif">
  <div class="confirmation-message">

    <h1>${Resource.msg('confirmation.thankyou', 'checkout', null)}</h1>

    <iscontentasset aid="confirmation-message" />
  </div>

  <div class="order-confirmation-details">
    <isorderdetails order="${pdict.Order}"/>
  </div>
<isinclude template="checkout/confirmation/confirmationregister"/>
<isinclude url="${URLUtils.url('Pepperjam-RenderPixelOrder', 'order', pdict.Order.getCurrentOrderNo())}"/>

  <div class="actions">
    <a href="${URLUtils.http('Cart-ContinueShopping')}" class="continue">
      ${Resource.msg('confirmation.returnshop', 'checkout', null)}
    </a>
  </div>
</div>
```

3.3.2 Pepperjam Gateway Advertiser Pixel

For the Gateway Advertiser Pixel, insert the code below into the components/footer/pageFooter.isml template

```
<isinclude url="${URLUtils.url('Pepperjam-SetCookie', 'source',
(pdict.CurrentHttpParameterMap.source.stringValue||')), 'clickId',
(pdict.CurrentHttpParameterMap.clickId.stringValue||'))}" />
```

```
<iscontent type="text/html" charset="UTF-8" compact="true"/>
<iscomment>
  This is the footer for all pages. Be careful caching it if it contains
  user dependent information. Cache its elements instead if necessary (do not forget
  to change the isinclude into a pipeline include for that).
</iscomment>
<isinclude template="util/modules"/>
<isinclude url="${URLUtils.url('Pepperjam-RenderTag')}" />
<isinclude url="${URLUtils.url('Pepperjam-SetCookie', 'source', (pdict.CurrentHttpParameterMap.source.stringValue||'))}" />
</footer>
<div class="footer-container">
  <div class="footer-item">
    <isslot id="footer-column" description="Content in column 1 of the Footer" context="global" />
  </div>
  <div class="footer-item">
    <iscontentasset aid="footer-account"/>
  </div>
  <div class="footer-item">
    <iscontentasset aid="footer-support"/>
  </div>
  <div class="footer-item">
    <iscontentasset aid="footer-about"/>
  </div>
</div>
</footer>
```

pepperjam

```
<footer>
<isinclude url="{URLUtils.url('Pepperjam-RenderTag')}}" />
<isinclude url="{URLUtils.url('Pepperjam-SetCookie', 'source',
(pdict.CurrentHttpParameterMap.source.stringValue||')), 'clickId',
(pdict.CurrentHttpParameterMap.clickId.stringValue||'))}" />
  <div class="container">
    <div class="footer-container row">
      <div class="footer-item col-sm-3 store">
        <iscontentasset aid="footer-locate-store" />
      </div>
      <div class="footer-item col-sm-3 collapsible-xs">
        <iscontentasset aid="footer-account" />
      </div>
      <div class="footer-item col-sm-3 collapsible-xs">
        <iscontentasset aid="footer-support" />
      </div>
    </div>
  </div>
</footer>
```

3.3.3 Pepperjam Tag Container

For the Tag Container, insert the code below into the components/footer/pageFooter.isml template above the Gateway Advertiser pixel

```
<isinclude url="{URLUtils.url('Pepperjam-RenderTag')}}" />
```

```
<iscontent type="text/html" charset="UTF-8" compact="true"/>
<iscomment>
  This is the footer for all pages. Be careful caching it if it contains
  user dependent information. Cache its elements instead if necessary (do not forget
  to change the isinclude into a pipeline include for that).
</iscomment>
<isinclude template="util/modules"/>
<isinclude url="{URLUtils.url('Pepperjam-RenderTag')}}" />
<isinclude url="{URLUtils.url('Pepperjam-SetCookie', 'source', (pdict.CurrentHttpParameterMap.source.stringValue||'))}" />
<footer>
  <div class="footer-container">
    <div class="footer-item">
      <isslot id="footer-column" description="Content in column 1 of the Footer" context="global" />
    </div>
    <div class="footer-item">
      <iscontentasset aid="footer-account"/>
    </div>
    <div class="footer-item">
      <iscontentasset aid="footer-support"/>
    </div>
    <div class="footer-item">
      <iscontentasset aid="footer-about"/>
    </div>
  </div>
</footer>
```

```
<footer>
<isinclude url="{URLUtils.url('Pepperjam-RenderTag')}}" />
<isinclude url="{URLUtils.url('Pepperjam-SetCookie', 'source',
(pdict.CurrentHttpParameterMap.source.stringValue||')), 'clickId',
(pdict.CurrentHttpParameterMap.clickId.stringValue||'))}" />
  <div class="container">
    <div class="footer-container row">
      <div class="footer-item col-sm-3 store">
        <iscontentasset aid="footer-locate-store" />
      </div>
    </div>
  </div>
</footer>
```

4 Reference: Feed Jobs Run

After all imports of metadata are complete and credentials are configured in site preferences, you can view the jobs run in Business Manager. In Business Manager go to Administration —> Operations —> Job Schedules. Select the job which you want to run:

ID	Status	Last Run	Execution Scope	Resources	Priority	Enabled	Delete
Pepperjam-product_feed	OK	5/30/2018 10:26 am		-		<input type="checkbox"/>	
Pepperjam_correction_order	OK	6/4/2018 7:05 am		-		<input type="checkbox"/>	

Then click run manually or schedule it as you need in the Schedule and History tab.

Pepperjam-product_feed

General | **Schedule and History** | Resources | Step Configurator | Notification | Failure Handling

ID*
Pepperjam-product_feed

Description
creates csv file with all site products

Priority
 Normal High

Run Now

5 Testing

Once you are done in your sandbox environment and have tested it, replicate the changes to your production environment following SFCC best practices on replication: https://documentation.b2c.commercecloud.salesforce.com/DOC1/index.jsp?topic=%2Fcom.demandware.dochelp%2Fcontent%2Fb2c_commerce%2Ftopics%2Freplication%2Fb2c_replication.html

Test cases for checking cartridge performance are described in the “PepperJam Test cases.xlsx” document.

Then, let Pepperjam know the steps above have been completed. They will begin the testing process and verification of the pixel.

6 Availability

FTP server should always be available. In the event that server is not available, Pepperjam cannot pick the feed up from.

7 Support

Along the way, if you need assistance with your implementation, please contact your

pepperjam

Pepperjam account strategist or launch team representative.

In cases when service API doesn't respond, Pepperjam iframe html will still be rendered on a page, error will be displayed in the console, though rest of website functionality (including client side scripts) won't be disrupted.