**PIXLEE**

***Version 19.1.0***

****

Table of Contents

[1 Summary 1-3](#_Toc10022223)

[2 Component Overview 2-3](#_Toc10022224)

[2.1 Functional Overview 2-3](#_Toc10022225)

[2.1.1 Overview of Components 2-3](#_Toc10022226)

[2.1.2 Use Cases 2-4](#_Toc10022227)

[2.2 Compatibility 2-5](#_Toc10022228)

[2.3 Privacy, Payment 2-5](#_Toc10022229)

[3 Implementation Guide 3-6](#_Toc10022230)

[3.1 Before you begin 3-6](#_Toc10022231)

[3.1.1 Pixlee account setup 3-6](#_Toc10022232)

[3.1.2 Getting your Pixlee Account ID and API keys 3-6](#_Toc10022233)

[3.1.3 Configuring your widgets 3-6](#_Toc10022234)

[3.2 Deploy Pixlee cartridge to your SFCC environment 3-7](#_Toc10022235)

[3.2.1 Getting the cartridge 3-7](#_Toc10022236)

[3.2.2 Updating the metadata 3-7](#_Toc10022237)

[3.3 Building and deploying the code 3-9](#_Toc10022238)

[3.3.1 SFRA 3-9](#_Toc10022239)

[3.3.2 Site Genesis 3-9](#_Toc10022240)

[3.4 Configuring SFCC environment 3-11](#_Toc10022241)

[3.4.1 Updating cartridge paths 3-11](#_Toc10022242)

[3.4.2 Configuring Site Preferences 3-12](#_Toc10022243)

[3.4.3 Organization preferences 3-14](#_Toc10022244)

[3.4.4 Configuring Jobs 3-14](#_Toc10022245)

[3.5 Updating your SFCC application 3-17](#_Toc10022246)

[3.5.1 SFRA 3-17](#_Toc10022247)

[3.5.2 Site Genesis 3-17](#_Toc10022248)

[4 Release History 4-22](#_Toc10022249)

# Summary

This document explains how to install Pixlee extension module (cartridge) on web stores built on top of Salesforce Commerce Cloud (SFCC) B2C platform, including both SFRA and Site Genesis.

For the most up-to-date version please check the following article:

<https://developers.pixlee.com/docs/salesforce-commerce-cloud-demandware>

# Component Overview

Pixlee LINK cartridge is comprised of the following components:

* A service definition to allow SFCC applications to communicate with Pixlee web services;
* A job component allowing product and category details to be passed from SFCC to Pixlee;
* ISML templates for rendering Pixlee widgets on product details and category landing pages (PDP and CLP respectively);
* Components for enabling Pixlee analytics, including triggering add-to-cart and conversion events.

## Functional Overview

The owners of SFCC web stores who have integrated Pixlee cartridge can benefit from extending users experience by allowing their customers to share and view photos of their products associated with a specific product or category.

For that to happen, the first step would be to make details of the products and categories known to Pixlee, for which purpose an export job component is provided by the cartridge.

Once products are loaded into Pixlee, photos from social media can be pulled and associated with them by merchandisers in Pixlee admin tool. Merchandiser can also define the widgets appearance and behavior there.

Having configured PDP and CLP widgets in Pixlee and entered their IDs in corresponding site preferences in SFCC, those widgets will start showing on the site, allowing customers to add their own photos from social media or upload. The uploaded photos are moderated.

Pixlee integration also allows monitoring the usa of the widgets and tracking conversion. This functionality can be enabled or disabled depending on customers giving their consent to be tracked.

### Overview of Components

The following three cartridges are provided:

* **int\_pixlee\_core** – this is a core cartridge containing components that could be used by both SFRA and SiteGenesis based applications, including service and job component definitions, isml templates and modules, resources etc.
* **int\_pixlee\_sfra** – this cartridge contains SFRA specific components, including controllers, isml templates, client-side scripts;
* **int\_pixlee** – this cartridge contains Site Genesis specific components, including controllers, isml templates, client-side scripts.

### Use Cases

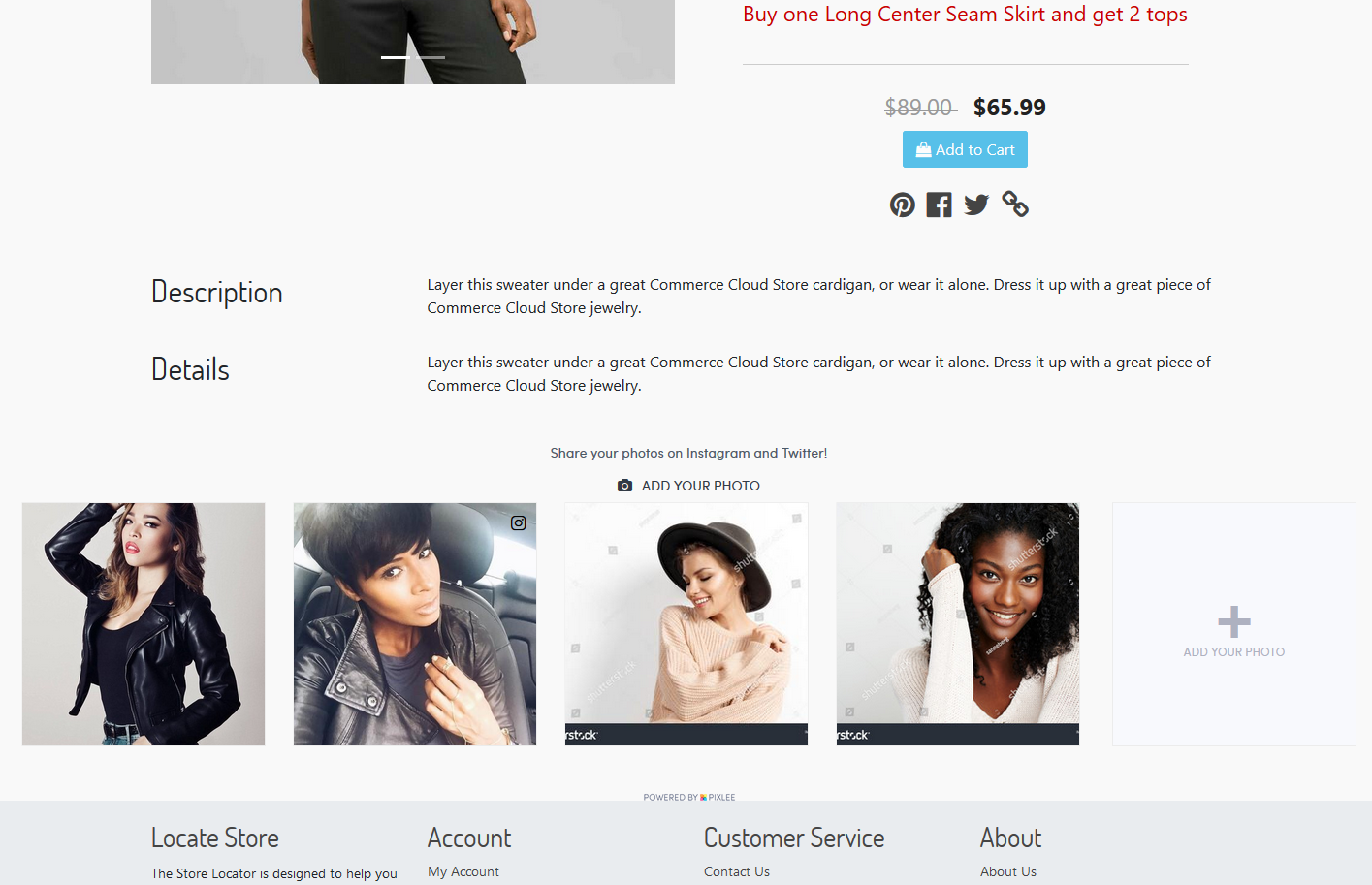
#### Business Manager

* *Export Products*
  + *Administrator decided to export the current products in the system to Pixlee. Administrator clicked on ‘Administrator’ followed by ‘Operations’ and then ‘Pixlee Export Products’. Administrator schedules the time to export the products or clicks ‘Run’ to execute the job immediately. Products will be exported to Pixlee accordingly.*

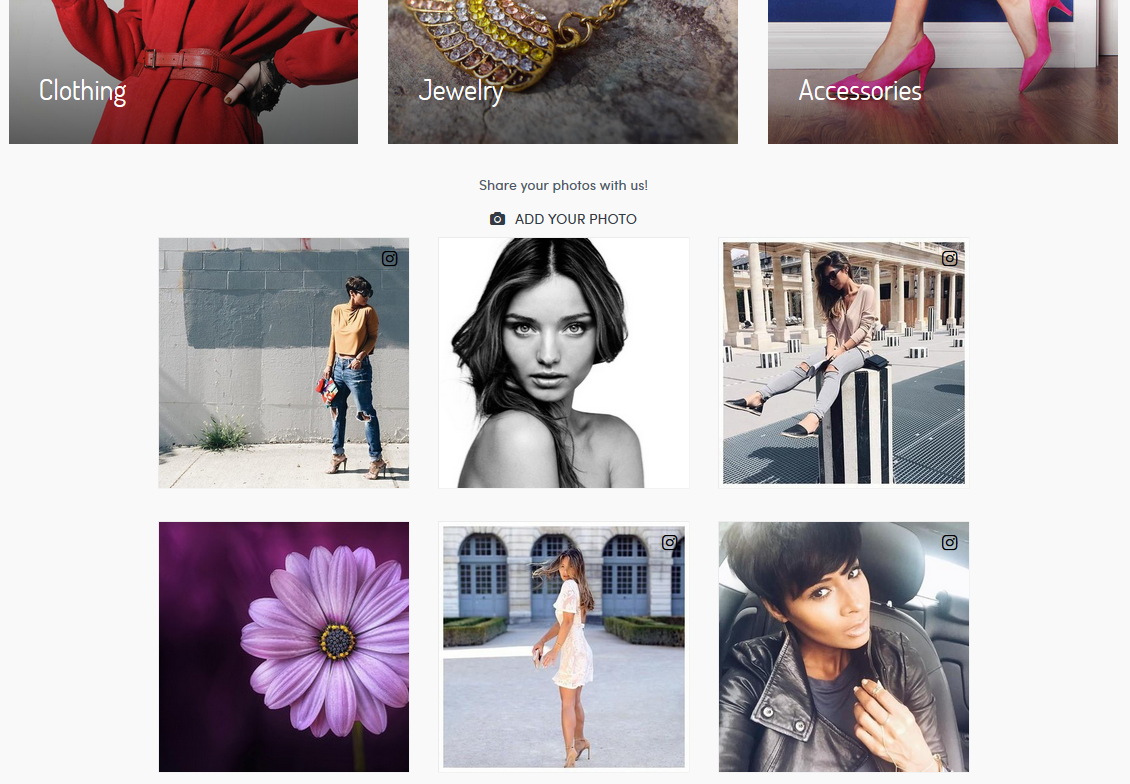
#### Storefront

##### Widgets

* *PDP Widget*
  + *User has decided to view the product and click on the product link. Product page displays the PDP widget as illustrated on the following screenshot:*

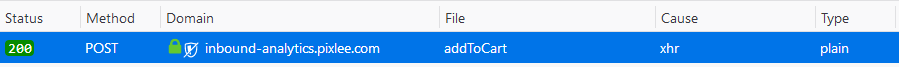


* *CLP Widget*
  + *User has decided to view the category landing and click on the category link. Category landing page displays the CLP widget as illustrated on the following screenshot:*

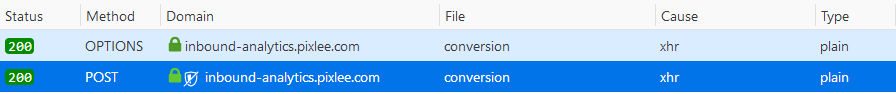


##### Events

* *Add to Cart*
  + *User has identified an item to buy and clicked on ‘Add to Cart’. Item was successfully added into the cart and analytics data was sent to Pixlee. A successful POST HTTP request to* <https://inbound-analytics.pixlee.com/events/addToCart> should be observed in browser’s network monitor.



* *Checkout*
  + *User has decided to checkout the current items in the cart and clicked on the cart. Cart with the current list of items was loaded and analytics data was sent to Pixlee. A successful POST HTTP request to* <https://inbound-analytics.pixlee.com/events/conversion> should be observed in browser’s network monitor:



## Compatibility

*Tested and compatible with* ***SFRA version 4.1*** *and* ***Site Genesis 104.1.3****.*

## Privacy, Payment

*Cart actions such as add to cart and checkout are being collected for analytics purpose. Product information will also be collected upon request on Demandware Business Manager.*

*Pixlee does not collect credit card data.*

# Implementation Guide

## Before you begin

### Pixlee account setup

To take advantage of services provided by Pixlee, you first need to sign up for Pixlee account. You can start with [requesting a demo](https://www.pixlee.com/request-demo) or contacting Pixlee directly.

### Getting your Pixlee Account ID and API keys

Once you have an account, you will be able to look up your Account ID and API keys (Account API Key and the Account Secret Key) required for connecting your store to Pixlee services.  
Please refer to [Getting your API Keys](https://developers.pixlee.com/docs/getting-your-api-keys) for more details as to how to obtain those keys.

### Configuring your widgets

The cartridge allows Pixlee widgets (a.k.a. displays) to appear on storefront pages like product details and category landing. Those widgets allow moderated content collected from social media or manually uploaded, related to your products or navigation categories, to be presented to your web site visitors. The visitors can also share or upload their own content straight from the site.

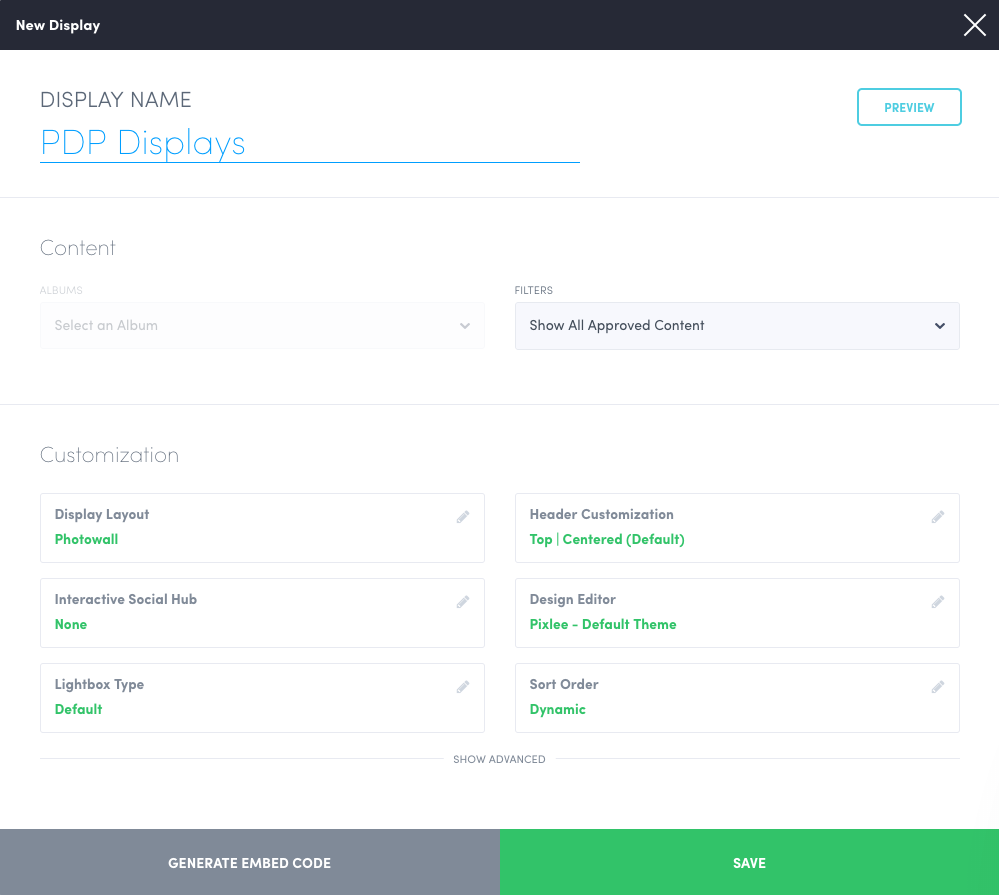
Pixlee widgets can be created and configured in Pixlee Admin, the configuration allowing to control their appearance and behavior.

As per your needs, configure two such widgets – one for product detail page and another one for category landing page (the widgets will automatically pick the relevant content to show for each product or category).

To configure your widgets:

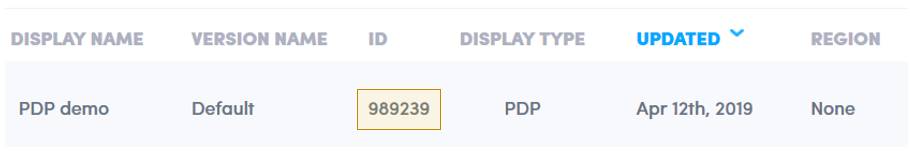
1) Go to <https://www.pixlee.com>, log in, and navigate to the **Publish** tab. Alternatively, point your browser to <https://app.pixlee.com/app#publish> while logged in.

2) Click the **Install Product Displays** and it should present you with a lightbox that looks like following.

[](https://files.readme.io/c1b6be2-01_99_new_pdp.png)

3) Customize the widget as you wish and click **Save**.

4) Take note of the widget ID. This number we be need in one of the later steps.

[](https://files.readme.io/e69b33f-widget_id.png)

Repeat the steps 1-4 and generate your second widget for category landing pages.

## Deploy Pixlee cartridge to your SFCC environment

### Getting the cartridge

You can obtain the cartridge by cloning Pixlee’s LINK marketplace [GitHub repository](https://github.com/SalesforceCommerceCloud/link_pixlee). Alternatively, you can download [Pixlee\_Demandware.zip](https://assets.pixlee.com/demandware/Pixlee_Demandware.zip)

The repository (or zip file) has three top-level folders:

* cartridges
* documentation
* metadata

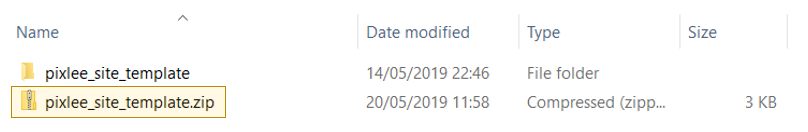
### Updating the metadata

The first thing we need to do when deploying the cartridge is to update the metadata of your SFCC environment. This metadata update includes:

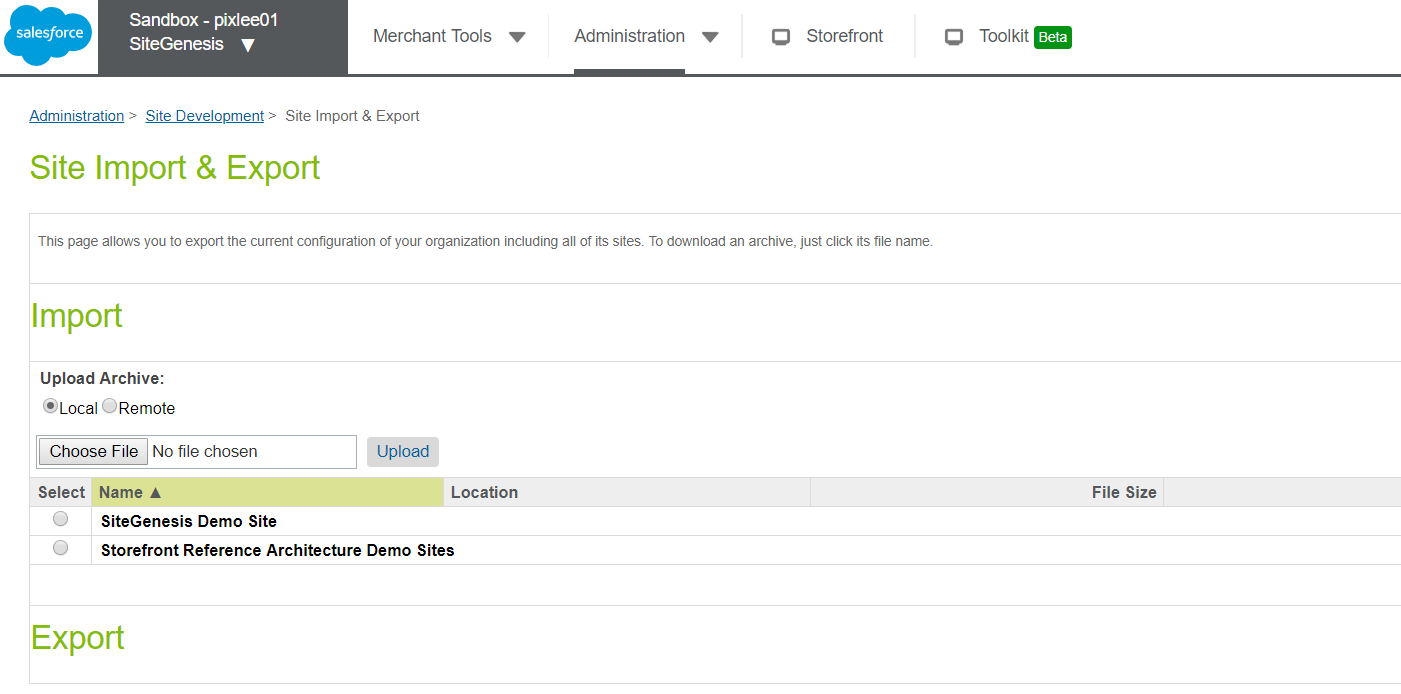
* Extensions to SFCC systems objects (site and organization preferences)
* Service definition
* Two sample jobs

The easiest way to update the metadata of your environment with the above is to do a site import. For that purpose:

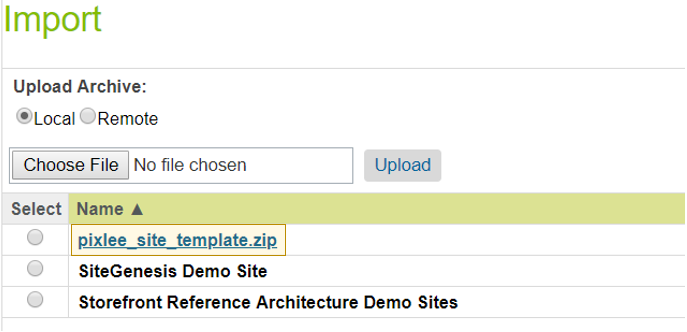
1) Open the metadata folder and create a zip of its **pixlee\_site\_template** subfolder:  
Please make sure the zip file has the same name as that folder:

[](https://files.readme.io/eed40b4-site_import_zip.png)

2) In your SFCC Business Manager, navigate to Administration > Site Development > Site Import & Export:

[](https://files.readme.io/c7a1065-site_import_export_landing.png)

3) Click **Choose File**, select the zip you created in the previous step, and then click **Upload**. Once upload completes, you should be able to see the zip file in the list of available imports:

[](https://files.readme.io/3c4c890-site_import_export_uploaded.png)

**NOTE**: The label of **Choose File** button can be different depending on your browser

4) Select the zip file from the list of available site imports and click **Import**, and then click **OK** to confirm.

5) Wait for the import job to finish, it should take less than a minute.

In case the import was successful, you should be able to see the following:

* Pixlee site preferences in Merchant Tools > Site Preferences > Custom Preferences.
* Pixlee Organization preferences in Administration > Global Preferences > Global Custom Preferences.
* Pixlee service definition (**pixlee.http.service**) in Administration > Operations > Services.
* Two sample jobs (**Pixlee Product Export – SFRA** and **Pixlee Product Export – SiteGenesis**) in Administration > Operations > Jobs.

## Building and deploying the code

To make your store application take advantage of Pixlee services, we need to make sure that the cartridge code is built and deployed (uploaded) to your SFCC environment. This process is different depending on the reference application your store is built upon, so a separate set of instructions will be given for SFRA and Site Genesis.

### SFRA

The SFRA version of Pixlee cartridge is designed around the same build and deployment concepts as the SFRA core cartridges and plugins provided by Salesforce.

1) In the root folder of your cloned repository, run npm install to install all of the local dependencies (node version 8.x or current LTS release recommended)

2) If necessary, update the path to your base SFRA installation in package.json file from the same root folder.

Normally you would have a top-level project folder, into which the repositories of SFRA base cartridge and all required plugins, libraries and any other LINK cartridges will be cloned. In case you have cloned the Pixlee cartridge into that folder as well, the below change will not be required. Otherwise, update paths.base property in the package.json to contain a relative path to the local directory containing the Storefront Reference Architecture repository. Its default value will be as follows:

"paths": {

"base": "../storefront-reference-architecture/cartridges/app\_storefront\_base/"

}

3) Once you are certain the correct path to SFRA cartidges is configured, run npm run compile:js command from the root folder of Pixlee repository.

4) Finally, having downloaded all local resources and having compiled the script files, we can upload the code to our SFCC environment (sandbox). That can be done from the command line as well, for which purpose a file called dw.json needs to be created in the root folder of Pixlee repository. It should have the folliwing content:

{

"hostname": "{{your-sandbox-hostname}}.demandware.net",

"username": "{{your\_login}}",

"password": "{{your\_pwd}}",

"code-version": "{{version\_to\_upload\_to}}"

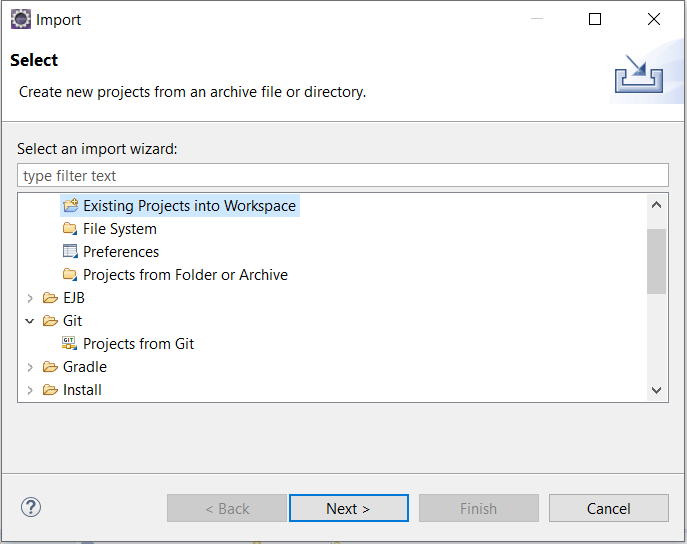
}

Once you have the file in place, run npm run uploadCartridge command. Make sure it uploads the two cartridges (**int\_pixlee\_sfra** and **int\_pixlee\_core**) required for SFRA based applications (even though uploadCartridge command sounds like a single cartridge will be uploaded, it should actually upload those two and report so).

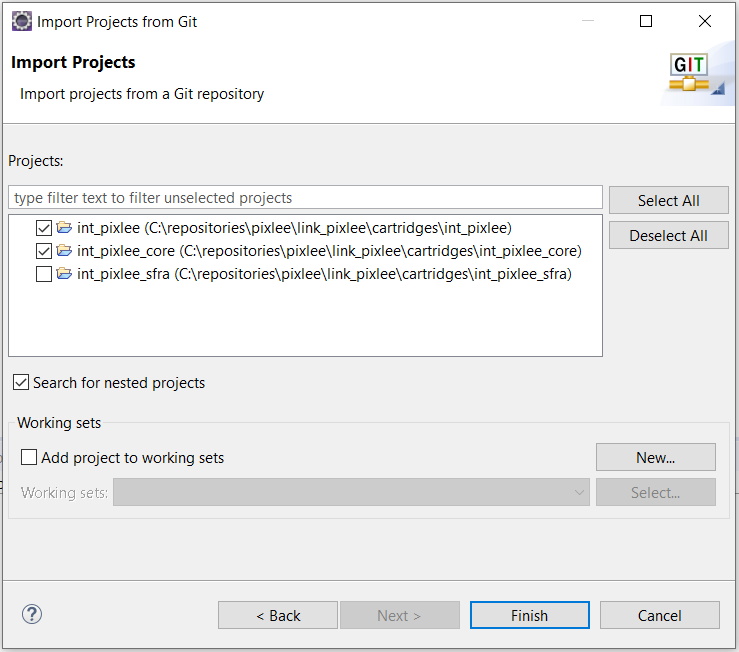
### Site Genesis

For Site Genesis based application we will need to upload **int\_pixlee** and **int\_pixlee\_core** cartridges as they are, no preliminary building or installing local resources will be required.

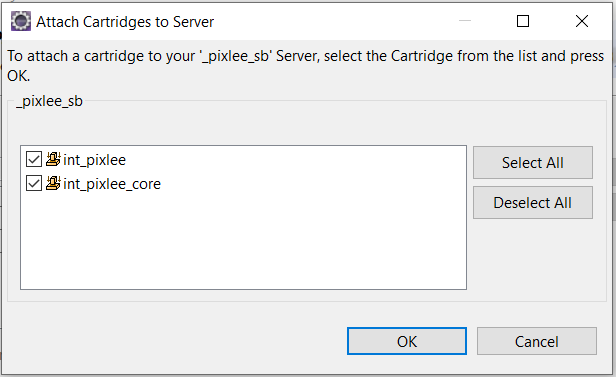
In case you are using Eclipse, make sure to import both those to your project workspace (by using File > Import > General > Existing Projects into Workspace or File > Import > Git > Projects from Git):

[](https://files.readme.io/618fdc0-eclipse_import_projects.png)

Make sure to select only the cartridges needed for Site Genesis but not **int\_pixlee\_sfra** (though it should not affect the application if uploaded):

[](https://files.readme.io/a523fb7-eclipse_import_projects_sitegen.png)

Finally, make sure that both cartridges are attached to you SFCC environment:

[](https://files.readme.io/2880c86-eclipse_import_projects_attach.png)

If you are using an alternative editor or IDE, you can still use the process described for SFRA to upload the cartridges (withouth running the npm run compile:js), having first updated scripts.uploadCartridge property in package.json to include **int\_pixlee** instead of **int\_pixlee\_sfra**:

"scripts": {

. . .

"uploadCartridge": "sgmf-scripts --uploadCartridge int\_pixlee\_core && sgmf-scripts --uploadCartridge int\_pixlee",

. . .

},

Having uploaded the cartridges relevant for your application in any of the ways described above, you may confirm they are present in your active code version (that should be the one you specified in dw.json file) by navigating to Administration > Site Development > Code Deployment in Business Manager and clicking on that version.

## Configuring SFCC environment

Now that we have the metadata updated and code uploaded, it's time to configure our SFCC instance (sandbox).

### Updating cartridge paths

We will need to add **int\_pixlee\_core** and one of the other cartridges to your cartridge path, depending on the type of reference application your store us built upon.

#### SFRA

For SFRA based applications, add the following to your cartridge path:

int\_pixlee\_sfra:int\_pixlee\_core

Note that with SFRA based applications custom cartridges should be added to the left of the base SFRA cartridge but to the right of the custom cartridge containing the customization for the specific project or client. For example, your effective cartridge path may look as follows:

app\_client:int\_pixlee\_sfra:int\_pixlee\_core:app\_storefront\_base:modules

#### Site Genesis

For SiteGenesis based stores, add the following cartridges:

int\_pixlee:int\_pixlee\_core

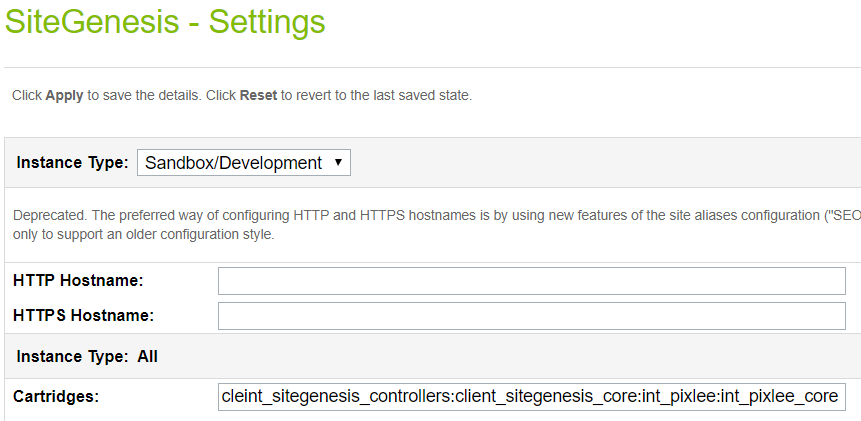
Unlike SFRA, for SiteGenesis based applications project or client specific customization should have been made to the SiteGenesis cartridges themselves, so Pixlee cartridges should be to the right of those. For example, your cartridge path may look as follows:

cleint\_sitegenesis\_controllers:client\_sitegenesis\_core:int\_pixlee:int\_pixlee\_core

To update cartridge path:

1) Navigate to Administration > Sites > Manage Sites, and click on the name of the site cartridge path of which you want to update.

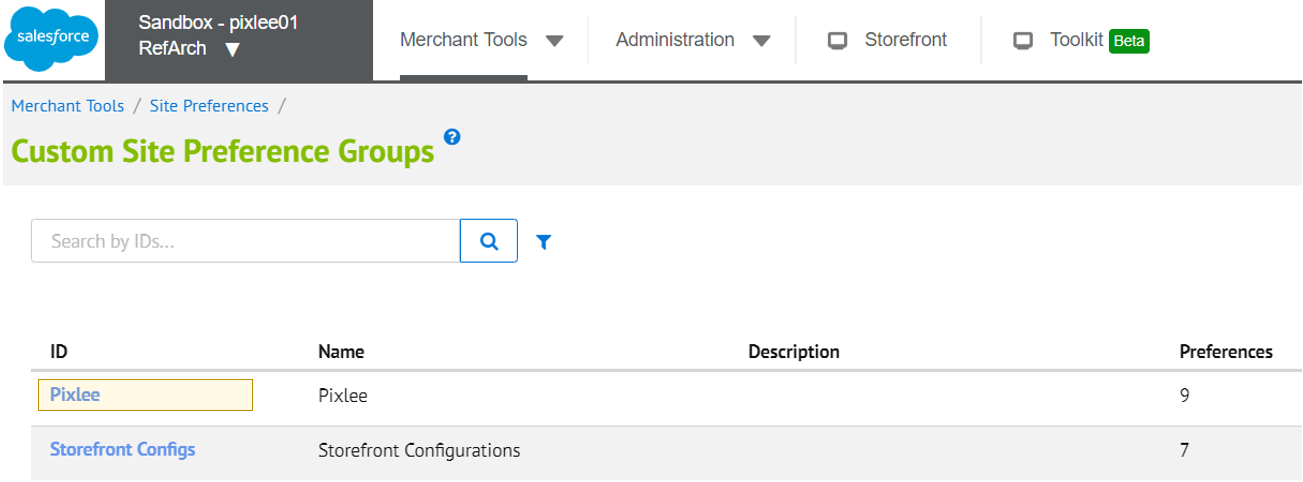
2) Click on **Settings** tab and make the required changes to the **Cartridges** field, and then click **Apply**:

[](https://files.readme.io/7f30220-cartridge_path.png)

**NOTE**: Please make sure all cartridge names are properly delimited by colons (:)

### Configuring Site Preferences

To configure the site preferences for your site, select that site in Business Manager, navigate to Merchant Tools > Site Preferences > Custom Preferences and select **Pixlee**:

[](https://files.readme.io/9963e49-pixlee_site_preferences.png)

The following table lists the site preferences that we need to configure:

|  |  |
| --- | --- |
| **Site Preference name** | **Type** |
| Enable Pixlee Plugin | Boolean [Dropdown] |
| Account ID | Number |
| Pixlee API Key | String |
| Pixlee Secret Key | String |
| SKU Reference | String [Dropdown] |
| Custom Product Host | String |
| PDP Widget ID | Number |
| CLP Widget ID | Number |
| Tracking option | String [Dropdown] |

**Enable Pixlee Plugin** is there to control whether the functionality provided by the cartridge is available on the storefront for a site.

**Account ID**, **Pixlee API Key** and **Pixlee Secret Key** should be populated with the corresponding details from Profile Settings > API Credentials, as described in the earlier section of this article. These credentials can be accessed directly by following this URL while logged in to Pixlee Admin: <https://app.pixlee.com/app#settings/pixlee_api>

**SKU Reference** is a setting to control which SFCC product attribute should be used to retrieve product IDs to be reported to Pixlee. The two options are:

* **Product ID** - this is the default option, in which case ID system attribute of Product system object will be used.
* **Manufacturer SKU** - an alternative options, allowing the value of manufacturerSKU system attribute of Product system object to be used.

**Custom Product Host** allows the hostnames of PDP URLs to be replaced when products are exported to Pixlee by the means of **Pixlee Product Export** job. If left empty, the hostname will not be replaced. If populated, the entered hostname will replace the one configured in SFCC. For example, if **www.example.com** is set, the following PDP URL

http://pixlee-test-sandbox-dw.demandware.net/s/SiteGenesis/demandware-mens-yarmouth-gloves/TG786.html?lang=en\_US

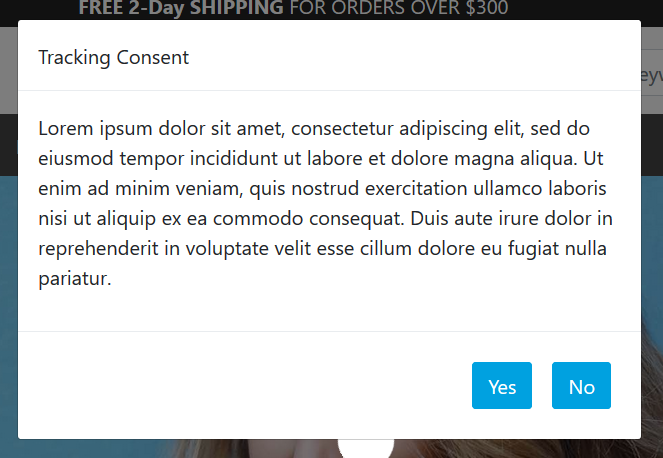
will be changed to

http://www.example.com/s/SiteGenesis/demandware-mens-yarmouth-gloves/TG786.html?lang=en\_US

**PDP Widget ID** and **CLP Widget ID** should be populated with the IDs of the widgets you want to show on product details and category landing pages respectively, steps on how to retrieve these values were covered earlier in this guide in the **Configuring your widgets** section.

**Tracking option** allows controlling the tracking behavior of the Pixlee cartridge. The cartridge contains some logic to fire events to Pixlee every time a product is added to cart, and when an order is finally placed. In addition to that, the scripts loaded from Pixlee and the widgets provided by Pixlee to be embedded on product details and category landing pages come with their own tracking functionality.

Both SFRA and Site Genesis reference applications include some logic to obtain consent from the customer to be tracked each time a new session is initiated, for which purpose a popup like this is shown:

[](https://files.readme.io/59f0639-sfra_tracking_consent.png)

The **Tracking option** setting allows the behaviour of both the cartridge, Pixlee scripts and widgets tracking to be controlled and aligned with the consent given by the customer. The available options are:

* **Always track** (TRACK\_ALWAYS) - in this case tracking will always be enabled, no matter if customers have given consent or not. This is the default option, replicating the behaviour of the previous version of the cartridge.
* **Track except if customer explicitly opted out** (TRACK\_IF\_NOT\_OPTED\_OUT) - in this case the customers will be tracked until they explicitly indicate they do not want to be tracked (like by clicking the No button on the above popup). For example, this option allows customers to be tracked on the first page they open, on which the tracking consent popup is shown.
* **Track only if customer explicitly opted in** (TRACK\_IF\_OPTED\_IN) - in this case customers will not be tracked until the explicitly give their consent (like by clicking **Yes** button on the above popup). For example, with this option customers will surely not be tracked on the first page they visit, on which the tracking consent popup is shown.
* **Disable tracking** (TRACK\_NEVER) - this is the most restrictive option, with which customers will never be tracked no matter of their consent. It is intended to be used for test purposes only, on non-production environments.

### Organization preferences

Even though there are two organization preferences defined by Pixlee cartridge, there is no need to set any of them, both will be automatically populated by the cartridge setup or job execution.

### Configuring Jobs

The first thing you would do in order to start using Pixlee widgets on your store is to let Pixlee know of the products you are selling. The product information is to be communicated offline, by the means of a job.

Pixlee SFCC cartridge provides a job step called **custom.PixleeExportProducts** allowing products from a single or multiple sites to be exported at once. The following table lists the configuration parameters:

|  |  |  |
| --- | --- | --- |
| **Parameter name** | **Type** | **Mandatory** |
| Products Source | String [Dropdown] | Yes |
| Images View Type | String | Yes |
| Main site ID | String | Yes |
| Break After | String [Dropdown] | Yes |
| Test Product ID | String | No |

**Products Source** allows the source of the product to be exported to Pixlee. It has the following options:

* **CATALOG\_API** - this is the default option, which replicates the behaviour of the previous version of Pixlee cartridge. If selected, dw.catalog.ProductMgr.queryAllSiteProducts() API will be used to retrieve the products.
* **SEARCH\_INDEX** - this option is considered to provide better performance and guarantees that only products showing on the site will be exported. It will use ProductSearchModel related APIs.

**Images View Type** allows the view type of the product images, URLs for which will be included in the export to Pixlee, to be specified. Its default value is **large**. It should have one of the values defined in Image Settings of the master catalog.

**Main site ID** should contain the ID of the site for which full product details should be exported. It could be useful in situations where you have multiple sites in SFCC sharing the same products. For example, let's consider you have a US site in English, a Canadian site in English and French, and a Mexican site in Spanish, all using products from the same master catalog. In this case you will want to nominate one of those sites as a main one, let's assume that would be the US site. With that configuration, having the job step set to run for all 3 sites, it will export full product details (including main PDP and image URLs, price in USD) and US English regional details for the US site, and **only regional details** (including localized PDP and image URLs) in Canadian French and English for Canada site and Spanish for Mexico site.

On the other hand, you may have multiple different brand sites each using a different set of products running on the same SFCC environment. In that case you would not want to nominate any of those sites as main, so in order to properly export the products to Pixlee a separate jobs using the same job step component should be configured.

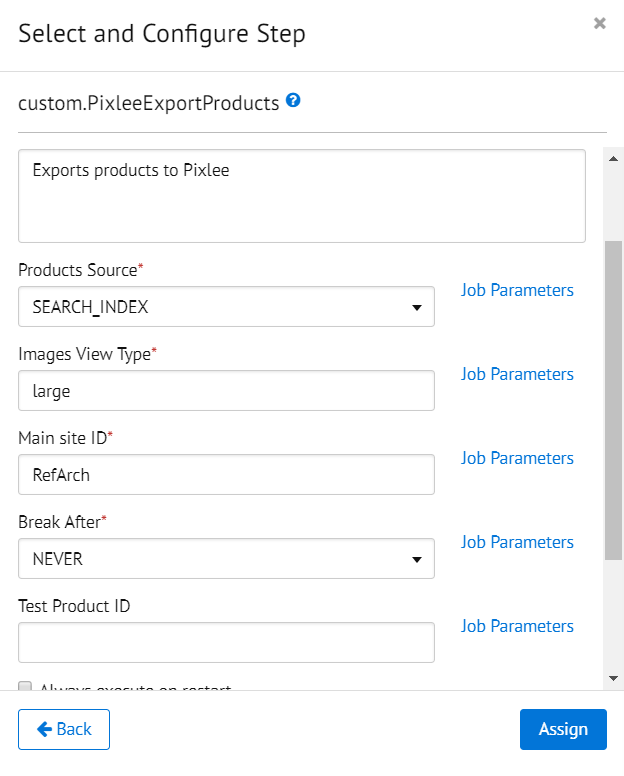
The cartridge comes with two sample jobs like that, one for SFRA (RefArch) and another one for Site Genesis. You may adjust them to the way that suits your needs, or remove and set up your own job.

**Break After** is a setting that can be used for testing and debugging the job with a large number of products. In general, if an export for a product fails, the failure will be logged, but the job will continue and try to export the next product (though it will finally end with an error). If you have a large number of products in the catalog though (say 100,000) and for some reason exporting each product results in an error, you may prefer to stop the job after a number of consecutive fails. The default value is **NEVER**, which means that the job will never stop even if the export of each product fails, and that is the one to be used in Production environments.

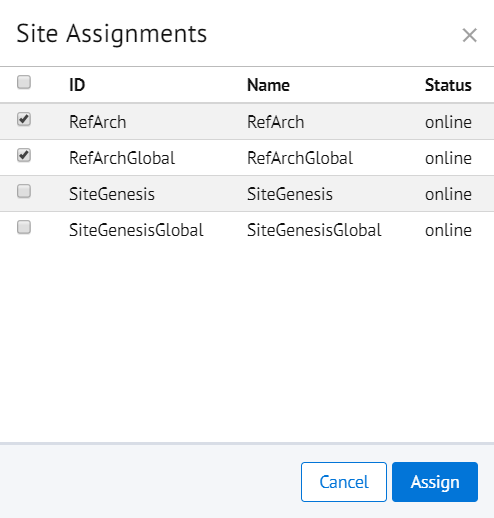
**Test Product ID** is another attribute that should be used only during testing or debugging the job. If set, it will look up the product for the ID specified and only export that product. If no such product exists, nothing will be exported. If left bank, which should be the default value and the value for Production environment, all products will be exported.

To configure your jobs, navigate to Administration > Operations > Jobs.

The following screenshot shows the example job step configuration for SFRA sites (RefArch and RefArchGlobal):

[](https://files.readme.io/12be114-job_step_configuration.png)

Site assignments of that step are illustrated on the following screenshot:

[](https://files.readme.io/20f1e05-job_step_sites_assigned.png)

## Updating your SFCC application

Depending on your application, some changes to its modules and components might be required to make Pixlee services function properly.

### SFRA

In an ideal scenario where you only have SFRA base cartridge and Pixlee cartridge installed, there is no need to make any changes to your application, the functionality provided by the cartridge will be available immediately after the cartridge is build and deployed as described above.

In real world there be lots of other 3rd party cartridges and plugins which may override the same components as the ones in Pixlee SFRA cartridge (int\_pixlee\_sfra). In that case you will need to merge changes from all those cartridges and plugins following the guidelines from this example repository provided by Salesforce:  
<https://github.com/SalesforceCommerceCloud/plugin_cartridge_merge>

Also, in case you override any of the templates that int\_pixlee\_sfra cartridge overrides (that would be the ones outside of pixlee folder), please make sure you port the changes from Pixlee cartridge into your client cartidge. All such changes should be wrapped in comments as follows:

<iscomment>Pixlee changes BEGIN</iscomment>

<isinclude template="pixlee/widgets/pdp" />

<iscomment>Pixlee changes END</iscomment>

### Site Genesis

As mentioned earlier, with Site Genesis based applications your client cartridges will very likely appear to the left of Pixlee cartridge in the cartridge path. Because of that changes will have to be made to some of the components in your client code. These will include changes to isml templates, and some changes to models or pipelines depending on whether you are using JavaScript controllers or pipelines.

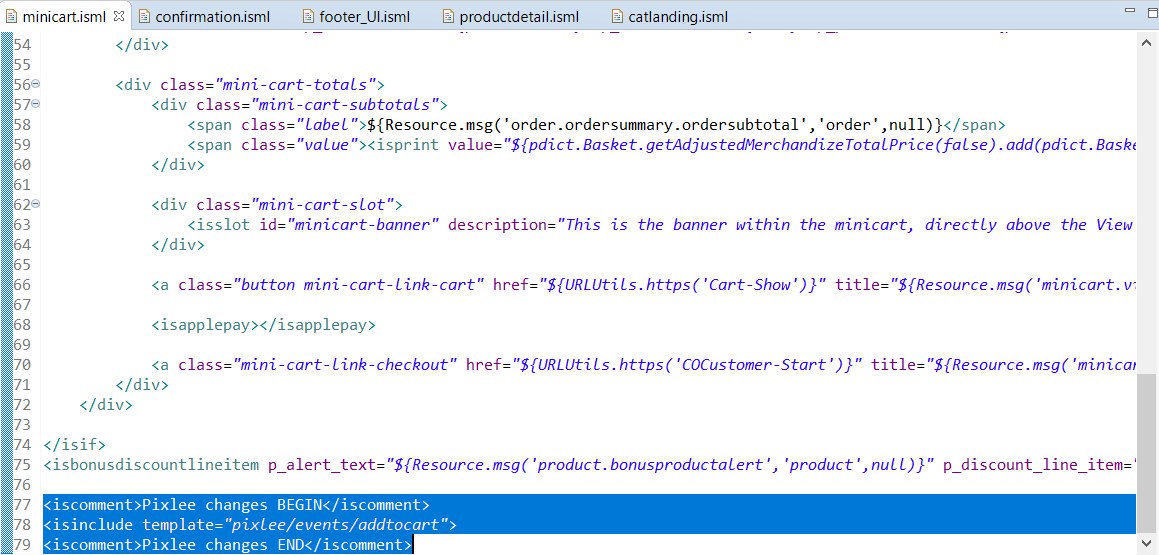
#### Changes to templates

1) Add the following lines to the very end of your checkout/cart/minicart.isml template:

<iscomment>Pixlee changes BEGIN</iscomment>

<isinclude template="pixlee/events/addtocart">

<iscomment>Pixlee changes END</iscomment>

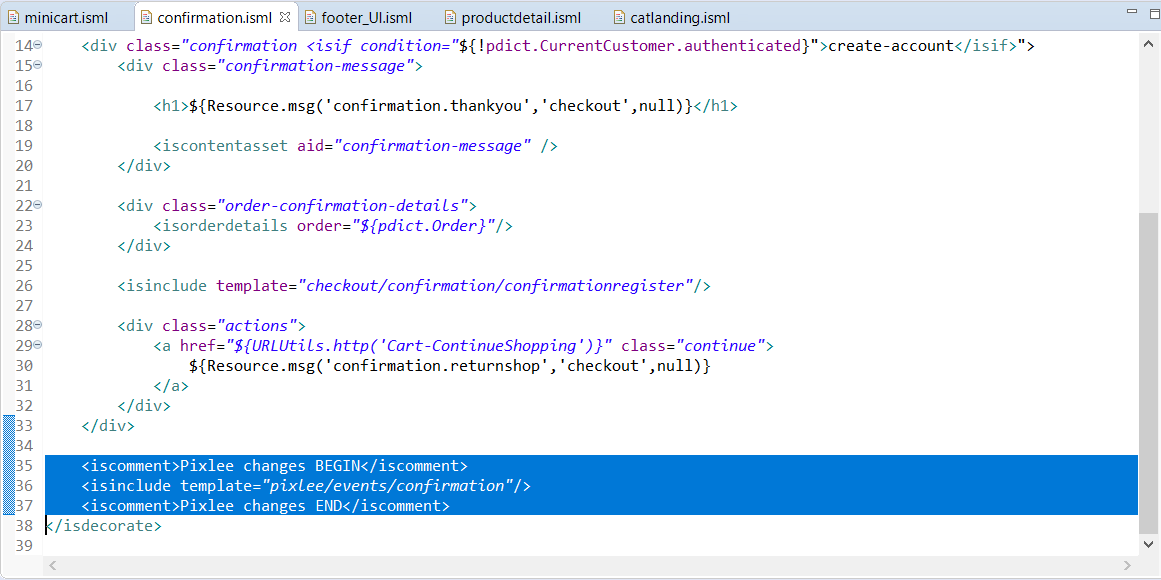
[](https://files.readme.io/4b53bfd-sg_minicart_template_changes.png)

2) Add the following lines to the end of your checkout/confirmation/confirmation.isml template, just above </isdecorate>:

<iscomment>Pixlee changes BEGIN</iscomment>

<isinclude template="pixlee/events/confirmation"/>

<iscomment>Pixlee changes END</iscomment>

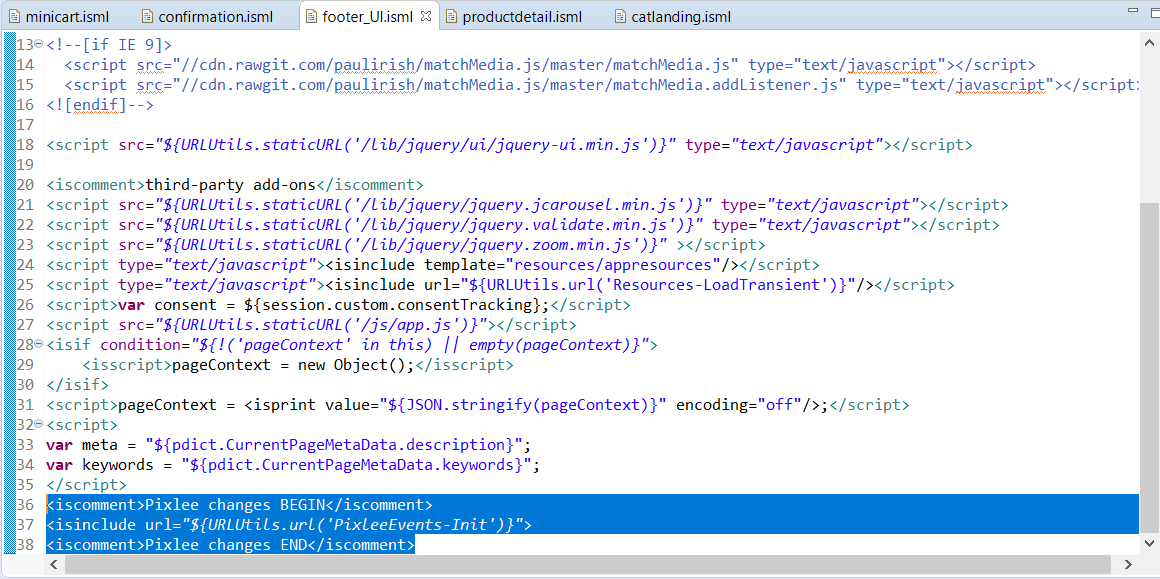
[](https://files.readme.io/2f9da91-sg_confirmation_template_changes.png)

3) Add the following lines to the very end of your components/footer/footer\_UI.isml template

<iscomment>Pixlee changes BEGIN</iscomment>

<isinclude url="${URLUtils.url('PixleeEvents-Init')}">

<iscomment>Pixlee changes END</iscomment>

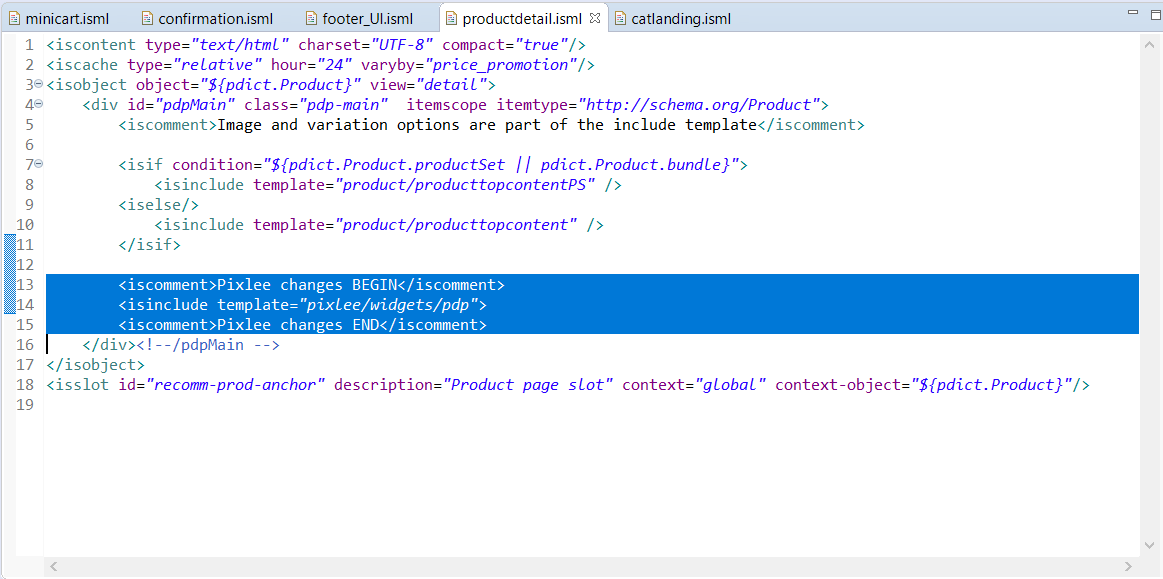
[](https://files.readme.io/6ddedd3-sg_footerui_template_changes.png)

4) Add the following lines to the end of your product/productdetail.isml template, just above the closing div tag for pdpMain container:

<iscomment>Pixlee changes BEGIN</iscomment>

<isinclude template="pixlee/widgets/pdp">

<iscomment>Pixlee changes END</iscomment>

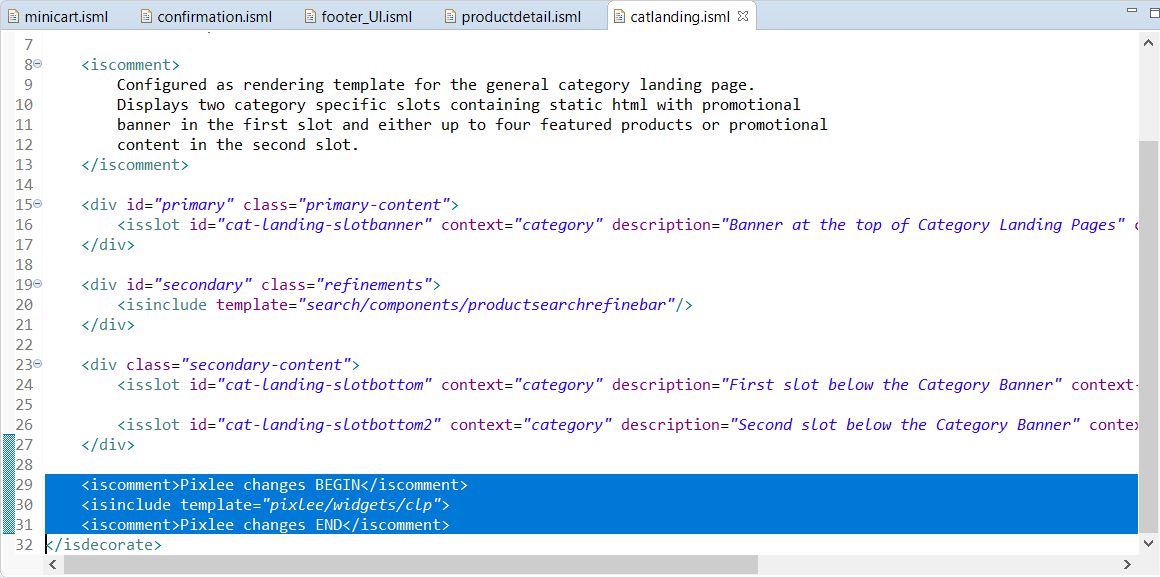
[](https://files.readme.io/b1bc4f8-sg_productdetail_template_changes.png)

5) Add the following lines to the end of your rendering/category/catlanding.isml template

<iscomment>Pixlee changes BEGIN</iscomment>

<isinclude template="pixlee/widgets/clp">

<iscomment>Pixlee changes END</iscomment>

[](https://files.readme.io/acc9fe3-sg_catlanding_template_changes.png)

#### Changes to JavaScript controllers based Site Genesis applications

**NOTE** This change is required to allow add to cart events to be triggered from wishlist page. In case wishlists are not in use on your store, there will be no need to apply it.

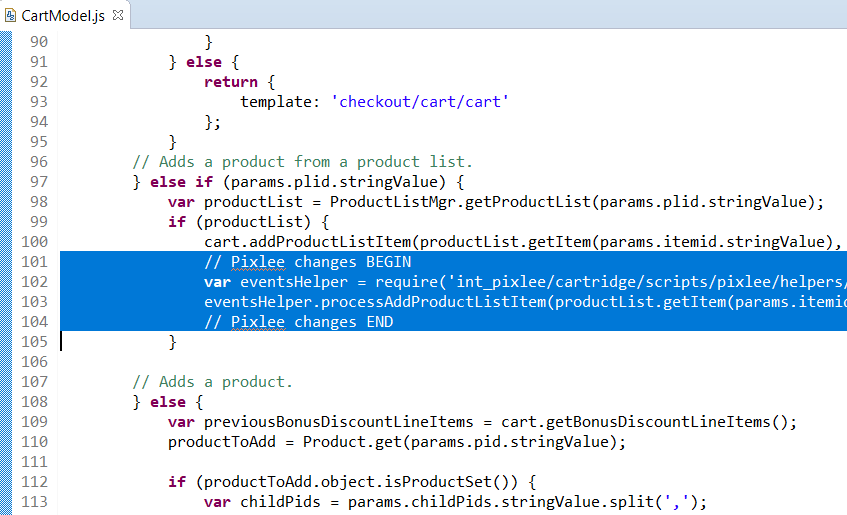
In you scripts/models/CartModel.js, add the following lines to addProductToCart function, in the condition handling the case for product addition from list (around line #100):

// Pixlee changes BEGIN

var eventsHelper = require('int\_pixlee/cartridge/scripts/pixlee/helpers/eventsHelper');

eventsHelper.processAddProductListItem(productList.getItem(params.itemid.stringValue), params.Quantity.stringValue);

// Pixlee changes END

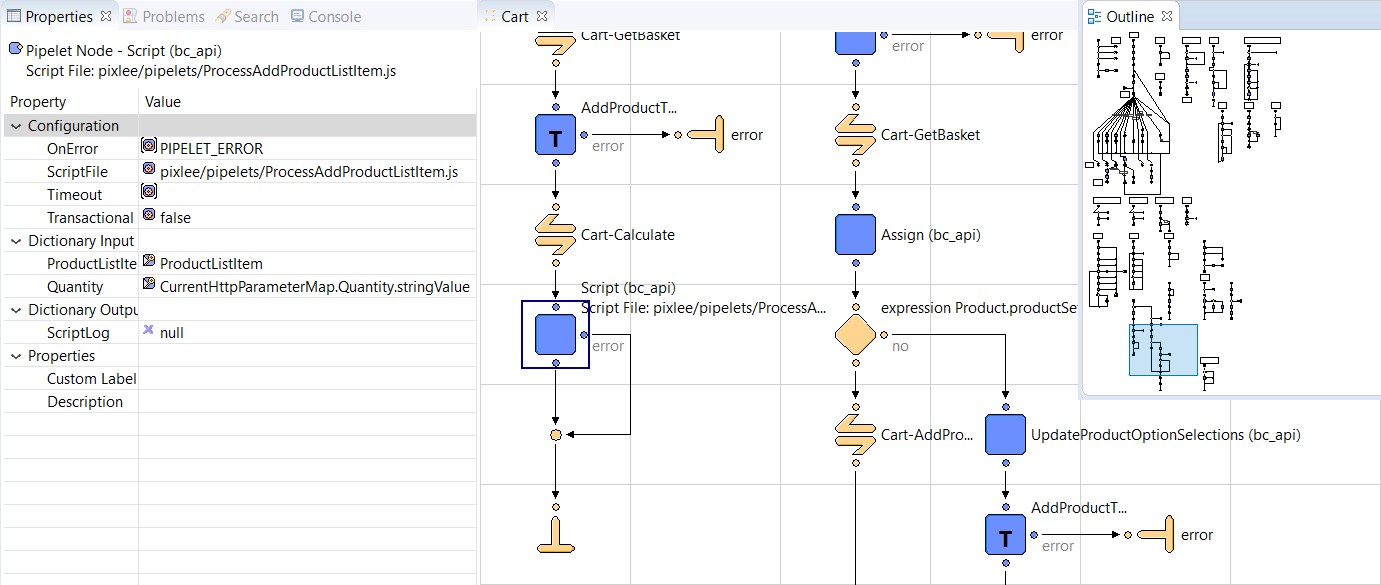
[](https://files.readme.io/bc72a31-sg_controllers_cartmodel_changes.png)

#### Changes to pipelines based Site Genesis applications

**NOTE**: This change is required to allow add to cart events to be triggered from wishlist page. In case wishlists are not in use on your store, there will be no need to apply it.

In your Cart pipeline update Cart-AddItem subpipeline, the branch that takes care of adding products to cart from a product list, you must add pixlee/pipelets/ProcessAddProductListItem.js Script File below Cart-Calculate Call Node and handle the error exit as per the screenshot below. 2 Dictionary Input parameters should be configured in the Script File:

* ProductListItem – set value to ProductListItem
* Quantity – set value to CurrentHttpParameterMap.Quantity.stringValue



# Release History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 15.1.0 | 04/19/2015 | Initial release |
| 15.2.0 | 06/23/2015 | Added PDP Widget, Export Product Photo URL |
| 15.2.1 | 10/07/2015 | Bug fixes |
| 16.1.0 | 12/08/2016 | Remove “Remove from Cart” |
| 17.1.0 | 03/06/2017 | Updated PDP widget instructions |
| 19.1.0 | 29/05/2019 | Refactored, added support for SFRA |