LINK\_mercaux SiteGenesis integration

Version <20.1.0>

****

Table of Contents

[**Summary**](#_heading=h.2bn6wsx) **4**

[**Component Overview**](#_heading=h.qsh70q) **5**

[*Functional Overview*](#_heading=h.3as4poj) *5*

[*Limitations, Constraints*](#_heading=h.1pxezwc) *5*

[*Compatibility*](#_heading=h.49x2ik5) *5*

[*Privacy, Payment*](#_heading=h.2p2csry) *5*

[**Implementation Guide**](#_heading=h.28pe86kart2g) **6**

[*Setup of Business Manager*](#_heading=h.2s8eyo1) *6*

[Download cartridge files from B2C Commerce marketplace/repository](#_heading=h.ms87kgx3anzq) 6

[Download required 3rd party libraries](#_heading=h.48j72yok6h6) 6

[Add cartridges to your development sandbox](#_heading=h.91xkiqtvia76) 6

[Import metadata](#_heading=h.9ymju6jzo76w) 7

[Check and edit files to be imported](#_heading=h.1v1yuxt) 7

[*Configuration*](#_heading=h.2u6wntf) *8*

[Setup Service to work with Mercaux API](#_heading=h.19c6y18) 8

[Update Searchable Attributes](#_heading=h.3tbugp1) 8

[Add Search Refinement to the Storefront Catalog](#_heading=h.28h4qwu) 8

[Run Job to fetch Looks from Mercaux API](#_heading=h.nmf14n) 9

[Enable/Disable Content Asset to show Looks Carousel on the PDP](#_heading=h.111kx3o) 10

[Enable/Disable Content Slot to show/hide Looks Carousel on the Homepage](#_heading=h.3l18frh) 10

[Looks Gallery page settings](#_heading=h.206ipza) 10

[Mercaux Custom Preferences](#_heading=h.4k668n3) 10

[Configure Recurring Interval of Analytic Job](#_heading=h.2zbgiuw) 11

[SFTP configuration to upload Analytic files (Optional)](#_heading=h.1egqt2p) 11

[*Custom Code*](#_heading=h.3ygebqi) *12*

[StoreFront code](#_heading=h.2dlolyb) 12

[Define Global File Assets and link 3rd party libraries in current base cartridge that the site uses](#_heading=h.sqyw64) 12

[Define Looks Carousel at the Product Detail Page](#_heading=h.3cqmetx) 13

[Define Look Modal at PLP search template](#_heading=h.1rvwp1q) 13

[Define Looks Carousel at the Homepage](#_heading=h.4bvk7pj) 13

[StoreFront Controllers code](#_heading=h.wp1m98evq7ls) 14

[*External Interfaces*](#_heading=h.3q5sasy) *16*

[**Testing**](#_heading=h.25b2l0r) **16**

[**Operations, Maintenance**](#_heading=h.kgcv8k) **17**

[*Data Storage*](#_heading=h.34g0dwd) *17*

[*Availability*](#_heading=h.1jlao46) *17*

[*Failover/Recovery Process*](#_heading=h.43ky6rz) *17*

[*Support*](#_heading=h.2iq8gzs) *17*

[**User Guide**](#_heading=h.xvir7l) **18**

[*Roles, Responsibilities*](#_heading=h.3hv69ve) *18*

[*Business Manager*](#_heading=h.1x0gk37) *18*

[Enable the cartridge](#_heading=h.4h042r0) 18

[Enable the cartridge globally](#_heading=h.2w5ecyt) 18

[Enable Look gallery](#_heading=h.nf0py872m4iz) 19

[Enable the cartridge globally](#_heading=h.av3iojstjh8f) 22

[Enable Look carousel in Homepage](#_heading=h.7gausx4j7ndd) 25

[*Storefront Functionality*](#_heading=h.pkwqa1) *28*

[Looks in carousels](#_heading=h.39kk8xu) 28

[Product page](#_heading=h.1opuj5n) 28

[Homepage carousel](#_heading=h.48pi1tg) 28

[Look Gallery](#_heading=h.2nusc19) 29

[Look details](#_heading=h.1302m92) 30

[Look pop-up](#_heading=h.3mzq4wv) 30

[Look Product detail pop-up](#_heading=h.2250f4o) 31

[**Known Issues**](#_heading=h.haapch) **31**

[**Release History**](#_heading=h.319y80a) **31**

# Summary

One of the numerous Mercaux Applications functionalities is a looks creation tool. These looks, or outfits, are created in HQ or in-store by store associates through Mercaux Applications.

The component we are looking at here will allow to display the Looks created in Mercaux Applications on three pages of your website:

1. Product pages
2. Look Gallery
3. Homepage

In order to achieve that, an access to Mercaux Apps and particularly Mercaux Public API is required.

The integration described here consists of a cartridge to retrieve and index Looks as products sets in Salesforce Commerce Cloud backend, along with Storefront new page and slot contents.

# Component Overview

## Functional Overview

The looks created in Mercaux Apps will be available in B2C Commerce and displayed in 3 different pages of the website:

1. Product Pages  
   In a product page, all looks that contain the product of the product page are displayed in a carousel
   * Looks details can be displayed in a pop-up to show products that are part of the look
   * Product size can be selected and product added to cart
2. Look Gallery  
   A new page link can be added to the menu to redirect to the look gallery
   * All Mercaux Looks can be browsed and filtered in the Look Gallery page
   * Look details can be displayed in a pop-up to show products that are part of the look
   * Product size can be selected and product added to cart
3. Homepage  
   From the Homepage, Looks can be displayed in a carousel
   * A list of specific Looks or random Looks can be displayed in a carousel
   * Look details can be displayed in a pop-up to show products that are part of the look
   * Product size can be selected and product added to cart

## Limitations, Constraints

In order to display the looks in the Look Gallery, Product Sets need to be searchable which makes them visible in the search result as well.

## Compatibility

The Mercaux cartridge is designed for Salesforce Commerce Cloud platform version 20.2 with controller based SiteGenesis architecture and new job framework.

Compatibility Mode: 19.10.   
Tested against SiteGenesis version 104.1.3

The cartridge has been tested on default en\_US locale however, it can support all the locales.

## Privacy, Payment

Customer profile data is not being accessed with Mercaux Cartridge and it doesn’t process or store credit card data.

# Implementation Guide

## Setup of Business Manager

#### Download cartridge files from B2C Commerce marketplace/repository

For SiteGenesis architecture, the following cartridges are required:  
 int\_mercaux\_sgjc -- storefront cartridge  
 int\_mercaux\_sgjc\_controllers -- storefront cartridge controllers  
 bm\_mercaux -- business manager cartridge

#### Download required 3rd party libraries

OWL Carousel

* Download and unpack OWL Carousel v. 2.3.4 distributive. [Link to download](https://owlcarousel2.github.io/OwlCarousel2/)
* Copy file owl.carousel.min.js from OWL Carousel dist/ to your static folder somewhere under js directory

Example:  
static/default/js/lib/owl.carousel.min.js

* Copy files owl.carousel.min.css and owl.theme.default.min.css from OWL Carousel dist/assets folder to your static folder somewhere under css directory

Example:  
static/default/css/lib/owl.carousel.min.css  
static/default/css/lib/owl.carousel.min.css

* You also need to include references to those files into default SiteGenesis templates.  
  See Custom Code section for details and make sure to follow all the instructions before checking carousel functionality.

#### Add cartridges to your development sandbox

Upload following cartridges to your CI development sandbox

int\_mercaux\_sgjc  
int\_mercaux\_sgjc\_controllers  
bm\_mercaux

Add cartridges to site cartridge path

* Go to *Administration > Sites > Manage Sites > ${Your site} > Settings* tab
* Add int\_mercaux\_sgjc\_controllers:int\_mercaux\_sgjc to the Cartridge field value at the beginning of the line.

Example:  
int\_mercaux\_sgjc\_controllers:int\_mercaux\_sgjc:app\_storefront\_controllers:app\_storefront\_core

* Click *Apply* button to save changes

Add cartridges to business manager cartridge path

* Go to *Administration > Sites > Manage Sites > Business Manager > Settings* tab
* Add bm\_mercaux to the Cartridge field value at the end of the line.

Example:  
bm\_app\_storefront\_base:bm\_custom\_plugin:bm\_mercaux

* Click *Apply* button to save changes

#### Import metadata

All required metadata for SiteGenesis is located under the metadata/SiteGenesis folder.  
Default configuration uses SiteGenesis site and storefront-catalog-m-en catalogue.  
Please replace them with your actual values before importing as described below.

#### Check and edit files to be imported

* metadata/SiteGenesis/sites/SiteGenesis/library/library.xml

Import Content Assets configurations to enable/disable the ability to show Looks Slider on the Product Detail Page.  
Make sure to replace SiteGenesisSharedLibrary with your library-id actual value. By default, the value is library-id="SiteGenesisSharedLibrary"

* metadata/SiteGenesis/sites/SiteGenesis/slots.xml

Import Content Slot configurations to have abilities to enable/disable Looks Slider on the home page.

You would have warnings during the import slots as you should define them in the codebase before import. Therefore define slot tags under your homepage template file as described in **Custom Code** section

* metadata/SiteGenesis/sites/SiteGenesis/ folder

Please rename this folder with your site id like this

metadata/SiteGenesis/sites/YOUR\_SITE\_ID/

* metadata/SiteGenesis/meta/system-objecttype-extensions.xml

The file contains attributes definition to extend following System Object Types: Product, SitePeferences

* metadata/SiteGenesis/meta/custom-objecttype-extensions.xml

The file contains Custom-Object Type definitions to have the proper configuration for Analytics.

* metadata/SiteGenesis/catalogs/storefront-catalog-m-en/catalog.xml

Import Catalog file to create Category “Looks” that Job uses to import products and assign them to the imported category.

Catalog storefront-catalog-m-en is used by default. Please replace this value with actual storefront catalog ID that is assigned to your site as follows.

<?xml version="1.0" encoding="UTF-8"?>

<catalog xmlns="http://www.demandware.com/xml/impex/catalog/2006-10-31" catalog-id="YOUR\_STOREFRONT\_CATALOG\_ID">

* metadata/SiteGenesis/jobs.xml

The file contains Jobs configurations for Import Looks and Analytic logic.

By default, job steps are configured to run with SiteGenesis site scope.  
Replace this value with your current site id and catalog ids like this:

<context site-id="YOUR\_SITE\_ID"/>

* metadata/SiteGenesis/services.xml

The file contains Service configurations to work with Mercaux API and this service is used in Jobs flow.

Create archive with metadata directory and import it

* Archive metadata/SiteGenesis folder.
* Go to *Administration > Site Development > Site Import & Export*
* Upload the archive
* Select archive in file list and push *Import* button

Alternatively, you can import each file separately. See this [cheatsheet](https://documentation.b2c.commercecloud.salesforce.com/DOC1/index.jsp?topic=%2Fcom.demandware.dochelp%2Fcontent%2Fb2c_commerce%2Ftopics%2Fimport_export%2Fb2c_import_export_object_cheatsheet.html&resultof=%22%69%6d%70%6f%72%74%22%20%22%65%78%70%6f%72%74%22%20) for the details.

## Configuration

#### Setup Service to work with Mercaux API

Go to the Services *Administration > Operations > Services* and find the recently imported Mercaux service named int\_mercaux\_api\_service.

You can go through the settings of service int\_mercaux\_api\_service to set up additional settings.

Setup credentials by going to*Credentials:* Select int\_mercaux\_api\_service\_creds credential item and put the relevant information of API to use within the cartridge, such as *URL, Password* endpoint of API.

By default, the service uses https://api.mercaux.com/1.0/api URL API endpoint.

You may get an error while Job is executing. If there is any error with provided credentials, make sure to check the Log directory to identify the error message.

Make sure that the service is enabled and credentials settings have the correct values.

#### Update Searchable Attributes

Go to *Merchant Tools > Search > Searchable Attributes* and add one more attribute as manufacturerSKU if that hasn’t been added before.

Mercaux's Job uses this attribute to find products by SKU.

You will get an error under the Logs section if your manufacturerSKU wasn't defined.

Example of error:

ErrorMessage: Products with the SKUs from Response weren't found or manufacturerSKU search refinement attribute wasn't defined.

#### Add Search Refinement to the Storefront Catalog

Go to your current storefront catalog for the considered site and define one more Search Refinement value if it hasn’t been done yet.

Under the *Search Refinement Definitions* tab, you should add new search refinement definition:

* Display name: SKU
* AttributeID: manufacturerSKU

After these steps your manufacturerSKU refinement attribute will be added to the storefront catalog.

Rebuild all indexes on *Merchant Tools > Search > Search Indexes*page.

#### Run Job to fetch Looks from Mercaux API

Go to the *Administration > Operations > Jobs* and select mercauxLooks Job.

You can find extra configuration in the different tabs of the Job, but the main settings you can find in the *Job Steps* tab.

The job has following steps:

**createXMLFile**

Contains functionality and configuration to generate XML files and collect them at the Impex folder.

Config fields such as:

* CatalogID - ID of your current Master catalog
* StorefrontCatalogID - ID of your current Storefront catalog. Best practice of Catalog design is using both Master and Storefront catalogs, but you also could set your Master catalog ID to this field.
* CatalogImageBasePath - Image Base path value of your current Master catalog. See Base path value on *Image Settings* tab of your master catalogue settings page.
* CategoryID -  Category ID where generated Looks will be assigned. Default value is "Looks".
* RootMercauxFolder - Root folder path where generated XML catalog with the looks are placed.

**Import**

Contains functionality to import generated catalog files and import them. There are configuration fields that you can edit based on your own site requirements.

**CleanArchiveFolder**

Contains functionality to keep the archive folder clean and has only a fixed amount of .zip files with the catalog files that have been imported.

**Reindex**

Before clicking *Run Now* make sure that you have successfully setup proper Scope per each Job step. There are values that should be per each Job Step:

* createXMLfile: Scope is Current Storefront Site
* Import, CleanArchiveFolder: Scope is Organization
* Reindex: Scope is Current Storefront Site

After the success of the Import step, Job runs Reindex step to update Search Indexes based on catalog import.

#### Enable/Disable Content Asset to show Looks Carousel on the PDP

Content Assets are used to show/hide Looks Carousel slider on the PDP.

After successful metadata import, you will find content asset mercaux-pdp-slider-looks-content.

This content asset responsible for global configuration shows/hides Looks Carousel on the PDP.

To globally enable or disable this content asset you need to

* Edit content csset
* Change *Online*option to Yes or No
* Click *Apply*

#### Enable/Disable Content Slot to show/hide Looks Carousel on the Homepage

Content Slots are used to show/hide Looks Carousel on the homepage.

After a successful import, go to *Merchant Tools > Online Marketing > Content Slots* and you will be able to find Mercaux's imported Content Slot homepage-mercaux-looks-slider

This content slot has two configurations:

* homepage-looks-slider-category-block

This slot configuration is responsible for the logic that is displaying looks from the selected category.  
By default - ***Looks*** category is selected.

You can set up the number of Looks that are displayed in the Slider from Category on the settings page *Merchant Tools > Custom Preferences > Mercaux Cartridge Settings* using the Number of Products in Slider Slot Category field. By default, the value is 6.

* homepage-looks-slider-product-ids-block

This slot configuration provides settings to show Looks Carousel based on selected Product Look IDs.

You have the flexibility to choose between what type of configuration you need to show on the homepage by changing the statuses to Active/Inactive.

Meanwhile, you are able to add configuration to the existing Content Slot that you have selected, just repeat the configuration of homepage-mercaux-looks-slider slot to the existing one.

#### Looks Gallery page settings

* Go to Merchant Tools > Products and Catalogs > Catalogs
* Select storefront catalog that current site is assigned to
* Notice Looks category and click Edit link to see the settings.

Under these settings you are able to configure Looks category based on your own configurations.

#### Mercaux Custom Preferences

* Go to Merchant Tools > Site Preferences > Custom Preferences
* Go under Mercaux Cartridge Settings settings page

Mercaux Cartridge provides Custom Preferences definition.

This definition will appear after the XML file of system object types is imported.

***This page has a few fields:***

* Default Look Gallery page Category ID

This value is important because all logic in the Controllers use this Category ID to check and append template value to render Mercaux’s templates only where necessary. By default, the value ID is Looks.

* Number of Products in Slider Slot Category

The number of products to display in the Looks Carousel Slider for Content Slot with Category logic to show.

* Enable Cartridge Globally

Turn on/off cartridge globally for all site and all places. By default, the value is No.

* Page size used while searching result in slot

The number of products received in one search result in the Content Slot templates to show sliders. By default, the value is 64.

#### Configure Recurring Interval of Analytic Job

After Job files are imported, Analytic job is configured to run every hour for generating Analytic files.

You can change the settings of Recurring config in MercauxAnalytic job settings.

* Go to your Analytic Job configuration *Administration > Operations > Jobs > MercauxAnalytic > Schedule and History* tab
* Under *Active* and *Run Time* you could see initial settings of run execution of job. You can configure by yourself or leave as is.

You can read more about this in the link to the [DOC](https://documentation.b2c.commercecloud.salesforce.com/DOC2/index.jsp?topic=%2Fcom.demandware.dochelp%2Fcontent%2Fb2c_commerce%2Ftopics%2Fjobs%2Fb2c_creating_jobs.html)

You can also add using SFTP Uploader step (See section #10 below) or just click *Run Now* button to start Analytic Job.

#### SFTP configuration to upload Analytic files (Optional)

If you would like to upload your generated analytic .XML files, you could setup one more job step based on 3rd party LINK cartridge Job Components

**Download and Install Job Components**

Download following LINK cartridge from the marketplace repository and install cartridge based on cartridge documentation.

Repository link: <https://github.com/SalesforceCommerceCloud/job-components>

**Setup your SFTP service configuration**

If you have already configured SFTP service you can skip this guide step.

* Create your SFTP service profile (Doc link: [create web service profile](https://documentation.b2c.commercecloud.salesforce.com/DOC2/topic/com.demandware.dochelp/content/b2c_commerce/topics/web_services/b2c_create_web_service_profile.html))
* Create your SFTP service credential (Doc link: [create web service credential](https://documentation.b2c.commercecloud.salesforce.com/DOC2/topic/com.demandware.dochelp/content/b2c_commerce/topics/web_services/b2c_create_web_service_credential.html))
* Create your SFTP service configuration (Doc link: [create web service configuration](https://documentation.b2c.commercecloud.salesforce.com/DOC2/topic/com.demandware.dochelp/content/b2c_commerce/topics/web_services/b2c_create_web_service_config.html))

In addition, one of the important steps of this setup is that you should select Service Type and set as SFTP ([Read more about Service Types](https://documentation.b2c.commercecloud.salesforce.com/DOC2/topic/com.demandware.dochelp/content/b2c_commerce/topics/web_services/b2c_service_types.html)).

Type and select profile and credential that you've created based on previous steps.

**Add SFTP job step**

* Go to *Administration > Operations > Jobs > MercauxAnalytic > Job Steps* tab
* Add one more sequential flow by clicking on + icon and click *Configure a step* to add one more Job step.
* If you successfully configured Job Components LINK cartridge type in search field custom.CSComponents.FtpUpload, you will find Job step that will be used for SFTP upload.
* Fill required fields based on configuration values. Short descriptions of fields:
  + ID - Add your Job step ID
  + ServiceID - Put Service configuration ID of your existing SFTP service configuration or which you created based on **Setup your SFTP service configuration** step
  + FilePattern - Set regex pattern of files that should be uploaded.  
    In our case value: .+\.xml
  + SourceFolder - Set Source folder where analytic files are placed. In our case value: src/mercaux/Analytics
  + TargetFolder - Set your target folder on the SFTP server where files will be uploaded
  + ArchiveFolder - Set Local folder where files will be archived after they will be uploaded on SFTP server, relative to IMPEX/. For example: src/mercaux/Analytics/archive

You could also configure additional configuration of Job step and when you're done click *Assign* button to save your step configuration.

After these steps, your Analytic job will use SFTP step to upload your analytic files to the SFTP server based on configuration.

## Custom Code

#### StoreFront code

**Define Global File Assets and link 3rd party libraries in current base cartridge that the site uses**

Modifications:

* cartridge/templates/default/components/footer/footer\_UI.isml

Replace {YOUR\_LIBS\_FOLDER} with the path you have placed in OWL Carousel js files and add following code on line #30.

<isif condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('mercaux\_GlobalEnableStatus')}">

<!-- Mercaux JS Assets Include -->

<span class="js-mercaux-analytics-tag" data-mercaux-analytics-url="${URLUtils.url('Analytics-Add')}"></span>

<script defer src="${URLUtils.staticURL('/js/{YOUR\_LIBS\_FOLDER}/owl.carousel.min.js')}"></script>

<script defer src="${URLUtils.staticURL('/js/mercauxApp.js')}"></script>

</isif>

* cartridge/templates/default/components/header/htmlhead.isml

Replace {YOUR\_LIBS\_FOLDER} with the path you have placed in OWL Carousel css files and add following code on line #83.

<isif condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('mercaux\_GlobalEnableStatus')}">

<!-- Mercaux CSS Assets Include -->

<link rel="stylesheet" href="${URLUtils.staticURL('/css/{YOUR\_LIBS\_FOLDER}/owl.carousel.min.css')}" />

<link rel="stylesheet" href="${URLUtils.staticURL('/css/{YOUR\_LIBS\_FOLDER}/owl.theme.default.min.css')}" />

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

</isif>

**Define Looks Carousel at the Product Detail Page**

We replaced the template of the Product Detail page with adding some part of code to have the ability to show Looks Carousel somewhere in the page and keep that file in the templates of the cartridge as well.

You can replace your own Product Detail template based on your current site template and define code below:

Modifications:

* cartridge/templates/default/product/producttopcontent.isml

Add the following code to define Mercaux Slider content slot:

<isinclude url="${URLUtils.url('Mercaux-IncludeSlider', 'cid', 'mercaux-pdp-slider-looks-content', 'pid', pdict.Product.ID)}" />

* cartridge/templates/default/product/producttopcontent.isml

Add the following code to define Mercaux Shop the Look Button:

<isif condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('mercaux\_GlobalEnableStatus')}">

<div class="shop-the-look-btn js-insert-shop-the-look-btn" data-slider-id="SliderLooksCarousel">

<a href="#SliderLooksCarousel" class="btn-primary">SHOP THE LOOK</a>

</div>

</isif>

**Define Look Modal at PLP search template**

To enable showing Looks Popup modal at the search page (PLP), you should add isinclude tag to include Modal Popup template.

Modifications:

* cartridge/templates/default/search/pt\_productsearchresult.isml

Put the following code before clothing </body> tag:

<isinclude template="modal/baseModal"/>

**Define Looks Carousel at the Homepage**

To enable showing Looks Carousel on the home page, you should define the place of the content slot definition in the Homepage template.

You can put the content slot that was imported along with XML import step:  
<isslot id="homepage-mercaux-looks-slider" description="" context="global" />

For example add following code to homepage.isml file.

* app\_storefront\_core/cartridge/templates/default/content/home/homepage.isml

<div class="homepage-mercaux-slot">

<!-- Mercaux Slot START -->

<isslot id="homepage-mercaux-looks-slider" description="" context="global" />

<!-- Mercaux Slot END -->

</div>

If you have already defined the content slot, just ensure that proper configurations of Mercaux Content Slot are in place:

* homepage-looks-slider-category-block - Show slider with Category logic
* homepage-looks-slider-product-ids-block - Show slider with predefined Product IDs logic

#### StoreFront Controllers code

**Define Hooks into Controllers of the current base cartridge that site uses to have the logic on PLP page for Looks Gallery**

Modifications:

* cartridge/controllers/Product.js method hitTile()

Add the following code before render method call:

// Mercaux Controller Logic ---- Start

var HookMgr = require('dw/system/HookMgr');

var isHookExecuted = HookMgr.callHook('mercaux.plp.looks.tile', 'renderProductTile', { params: params, productView: productView });

if (isHookExecuted) return;

// Mercaux Controller Logic ---- End

Final code should look like this:

function hitTile() {

// ..... rest code above .......

productView.product = product.object;

// Mercaux Controller Logic ---- Start

var HookMgr = require('dw/system/HookMgr');

var isHookExecuted = HookMgr.callHook('mercaux.plp.looks.tile', 'renderProductTile', { params: params, productView: productView });

if (isHookExecuted) return;

// Mercaux Controller Logic ---- End

productView.render(product.getTemplate() || 'product/producttile');

}

}

* cartridge/controllers/Product.js method variation()

Add the following code before render method call:

// Mercaux Controller Logic ---- Start

var HookMgr = require('dw/system/HookMgr');

var objArgs = {

params: params,

app: app,

product: product,

resetAttributes: resetAttributes,

currentVariationModel: currentVariationModel

}

var isHookExecuted = HookMgr.callHook('mercaux.modal.variation.filters', 'renderVariationInModal', objArgs);

if (isHookExecuted) return;

// Mercaux Controller Logic ---- End

Final code should look like this:

function variation() {

// ..... rest code above .......

if (product.isVisible()) {

// Mercaux Controller Logic ---- Start

var HookMgr = require('dw/system/HookMgr');

var objArgs = {

params: params,

app: app,

product: product,

resetAttributes: resetAttributes,

currentVariationModel: currentVariationModel

}

var isHookExecuted = HookMgr.callHook('mercaux.modal.variation.filters', 'renderVariationInModal', objArgs);

if (isHookExecuted) return;

// Mercaux Controller Logic ---- End

if (params.source.stringValue === 'bonus') {

const Cart = app.getModel('Cart');

const bonusDiscountLineItems = Cart.get().getBonusDiscountLineItems();

let bonusDiscountLineItem = null;

// ..... rest code below .......

}

* cartridge/controllers/Product.js method getSetItem()

Add the following code before render method call:

// Mercaux Controller Logic ---- Start

var HookMgr = require('dw/system/HookMgr');

var objArgs = {

product: product,

CurrentVariationModel: currentVariationModel,

isSet: true,

params: params,

app: app

}

var isHookExecuted = HookMgr.callHook('mercaux.modal.variation.tile.parentModal', 'renderVariationInParentModal', objArgs);

if (isHookExecuted) return;

// Mercaux Controller Logic ---- End

Final code should look like this:

function getSetItem() {

// ..... rest code above .......

if (product.isVisible()) {

// Mercaux Controller Logic ---- Start

var HookMgr = require('dw/system/HookMgr');

var objArgs = {

product: product,

CurrentVariationModel: currentVariationModel,

isSet: true,

params: params,

app: app

}

var isHookExecuted = HookMgr.callHook('mercaux.modal.variation.tile.parentModal', 'renderVariationInParentModal', objArgs);

if (isHookExecuted) return;

// Mercaux Controller Logic ---- End

app.getView('Product', {

product: product,

CurrentVariationModel: currentVariationModel,

isSet: true

}).render('product/components/productsetproduct');

} else {

// ..... rest code below .......

}

## External Interfaces

To import looks into storefront the cartridge sends HTTP requests to Mercaux public API https://api.mercaux.com/1.0/api/look endpoint.  
See the [documentation](https://mercaux.atlassian.net/wiki/pages/viewpage.action?pageId=2000322687) for complete reference of Mercaux public API.

# Testing

Please refer to the Test Case document included in the documentation folder.

# Operations, Maintenance

## Data Storage

The cartridge stores the following objects within Commerce Cloud:

* **Looks**  
  Looks are being stored as product sets using Product system object type.
* **Analytics Info**  
  Analytics is stored in custom object type.   
  Default retention time is 2 days.  
  Cartridge stores following information for the looks and the products that are part of the looks were shown
  + From which device (at least mobile vs. desktop)
  + From what page the look was shown
  + When it was shown
  + Session\_id (Mercaux developers must be able to disable this data from all events)
  + Customer\_id if logged-in (Mercaux developers must be able to disable this data from all events)
  + Products added to basket

Cartridge doesn’t store any data outside commerce cloud.

## Availability

The cartridge relies on access to Mercaux service only during the time mercauxLooks job is running.  
Any issues with Mercaux service availability would only affect the job ability to add new looks to storefront catalogue.  
All other functionality of the cartridge should be available. Eg. all the looks imported previously will be displayed as usual.

## Failover/Recovery Process

If the cartridge can't connect to Mercaux server, you’ll see the mercauxLooks job failing with related errors in logs.

Possible reasons could be:

1. Mercaux API service is not configured properly. Please go to *Administration > Operations > Services* and check that service is configured with proper URL and valid API Key  
   See Configuration section of this document for more information.
2. Firewall is configured to block requests to Mercaux API. Please contact your system administrator for help
3. Mercaux service is down Please reach out to Mercaux contact in your team

## Support

If you have any questions about using this cartridge please contact Mercaux manager supporting your company.  
If you’re not a Mercaux customer yet please contact us at [mercaux.com](https://mercaux.com/)

# User Guide

## Roles, Responsibilities

There are no recurring tasks required by the merchant. Once configurations and job schedules are set up, the functionality runs on demand.

## Business Manager

### Enable the cartridge

By default, the cartridge is disabled. In order to ensure installation has been performed correctly, you need to enable the Cartridge and the different Use Case you want to activate.

#### Enable the cartridge globally

In Business Manager, select the site for which you want to enable the cartridge and navigate to Merchant Tools > Site Preference > Custom Preferences.

A screenshot of a cell phone

Description automatically generated

Select Mercaux Cartridge Settings

A screenshot of a cell phone

Description automatically generated

Change the value of Enable Cartridge Globally to Yes and Save your modifications

A screenshot of a computer

Description automatically generated

Now the Cartridge is enabled we can go through basic test cases to ensure everything that has been installed works.

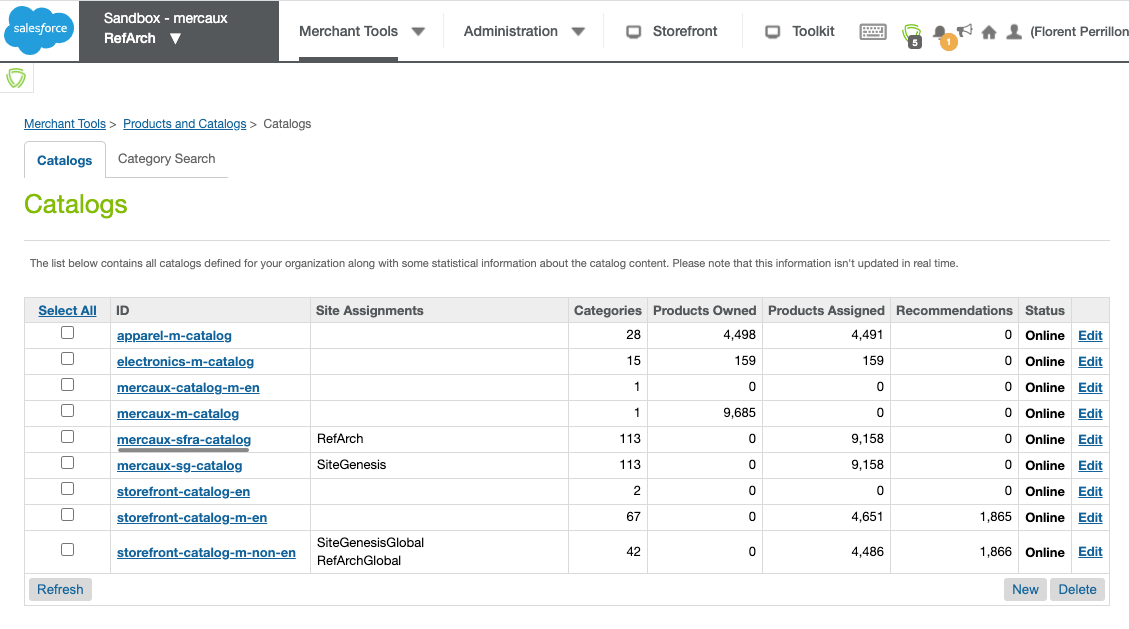
#### Enable Look gallery

You can choose to enable or disable Look Gallery from Business Manager. In order to change your preference, navigate to Merchant Tools > Product and Catalogs > Catalogs

A screenshot of a social media post

Description automatically generated

Select the catalog for the site you are considering



You should have a Look related category to select

A screenshot of a cell phone

Description automatically generated

In the Look category block, select Edit

A screenshot of a cell phone

Description automatically generated

Finally, check or uncheck the Online tick-box and Apply your modifications

A screenshot of a cell phone

Description automatically generated

#### 

#### Enable the cartridge globally

You can choose to enable or disable Look carousel in product pages from Business Manager. In order to change your preference, from Business Manager, navigate to Merchant Tools > Content > Content Assets

A screenshot of a cell phone

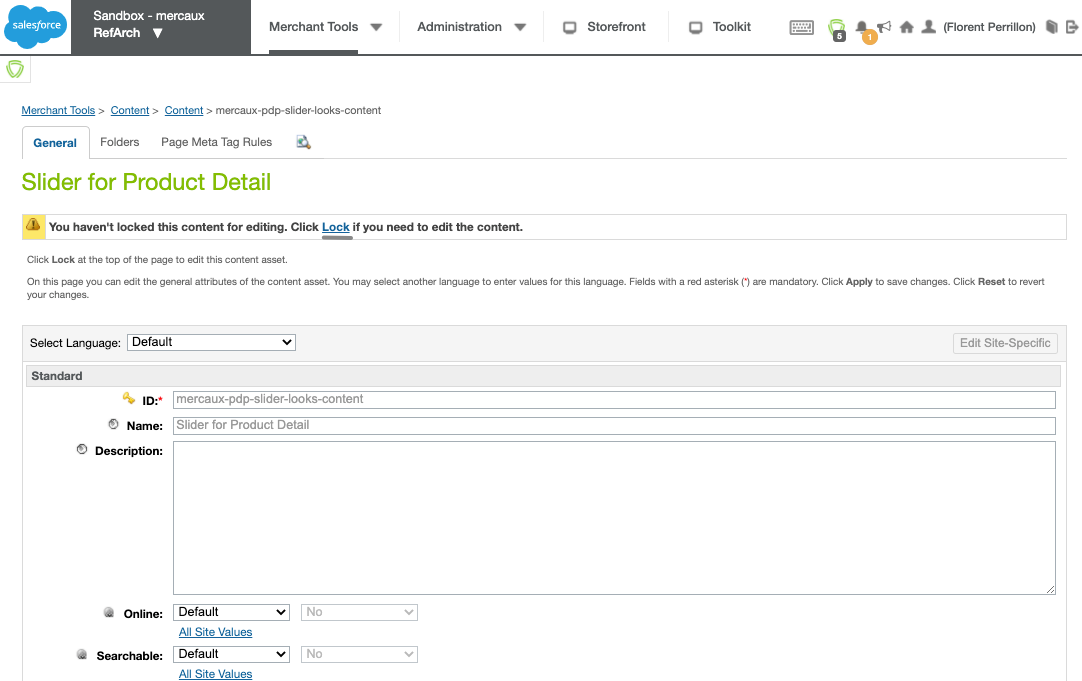
Description automatically generated

In the opened page, search for "mercaux-pdp-slider-looks-content" and click on the found “mercaux-pdp-slider-looks-content” element.

A screenshot of a cell phone

Description automatically generated

In the content page, enable edits by clicking on Lock button.



Near Online field, into Standard block, select Yes if you want to enable the carousel in product pages or No is you want to disable it. Don’t forget to Apply your modifications and Unlock the content to block the item from editing.

A screenshot of a cell phone

Description automatically generated

#### Enable Look carousel in Homepage

You can choose to enable or disable Look carousel in Homepage from Business Manager.  
In order to change your preference, from Business Manager, navigate to Merchant Tools > Online Marketing > Content Slots

A screenshot of a cell phone

Description automatically generated

In the opened page, search for "homepage-mercaux-looks-slider" and click on the found “homepage-mercaux-looks-slider” element.

A screenshot of a social media post

Description automatically generated

In the Slot configurations page, you will find two different slots:

* homepage-looks-slider-category-block which will display random Looks from a selected Category
* homepage-looks-slider-product-ids-block which will display selected Product IDs

In both cases, you can activate one of them by selecting the slot you want to enable, list the Products or Category you want to take into account and Apply your modification

A screenshot of a social media post

Description automatically generated

## Storefront Functionality

### Looks in carousels

#### Product page

A picture containing different, box, display, several

Description automatically generated

#### Homepage carousel

A picture containing photo, looking, different, table

Description automatically generated

### Look Gallery

A screenshot of a cell phone

Description automatically generated

### Look details

#### Look pop-up

A picture containing photo, different, room, white

Description automatically generated

#### Look Product detail pop-up

A screenshot of a cell phone

Description automatically generated

# Known Issues

None known.

# Release History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 20.1.0 | 12 June 2020 | Initial release |