

Version: 4.8

Published date: September 2025

Copyright © 2025 by ESW. All rights reserved. Information contained herein is subject to change without notice.

Multi Origin (MO)

INTEGRATION GUIDE



**SFCC - ESW Cartridge**

+

Contents

[1. Multi Origin 3](#_Toc201833259)

[2. Workflow 3](#_Toc201833260)

[2.1. Getting product origin 3](#_Toc201833261)

[3. BM Configurations 3](#_Toc201833262)

[3.1. Import metadata 3](#_Toc201833263)

[3.2. Import Services 4](#_Toc201833263)

[4. BM order detail page 6](#_Toc201833264)

[5. Storefront implementation 6](#_Toc201833265)

[5.1. Product listing on cart and order detail page 6](#_Toc201833266)

[5.2. Remove line item from basket 7](#_Toc201833267)

[6. Adding logic to fetch products from multiple origins 7](#_Toc201833268)

[7. Essential information to know 8](#_Toc201833269)

# **Multi Origin**

The ESW SFCC Cartridge enables retrieval of inventory information from various sources. For instance, if a shopper orders 5 products and the retailer has 3 in IE and 2 in FR, the ESW cartridge can check inventory details in real-time. Retailers simply need to implement logic in an override function to fetch products from different origins.

# **Workflow**

### **Getting product origin**

When MO feature is enabled and shopper is going to add a product in the basket, the origin will be get in the following ways:

* Check within the product which is going to add in the cart
* If no origin found, then check in master product
* If no origin found in the master product, then hit the API to get the origin
* If no origin found in the API as well, then it will not allow the shopper to add product in the basket
  1. **Line item grouping**

The cartridge will group the line items based on product ID and country origin. For example: If a shopper orders five units of a product, with two units shipped from France (FR) and three from Ireland (IE), the cartridge will create two separate line items for the product to reflect the different shipping origins.

# **BM Configurations**

### **Import metadata**

Import custom attributes (systemobjects.xml) from **link\_eshopworld\sitesdata\meta.**

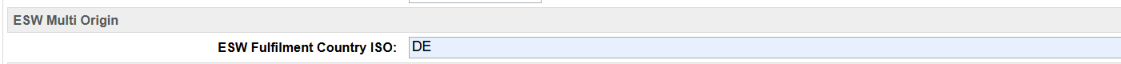
The following ESW custom attributes will be available under the 'ESW Multi Origin' group and need to be configured:

**Site preferences:**

* **ESW Multi Origin Enabled:** By default, this preference is set to NO. However, if you wish to use the multi-origin feature, you should set this preference to YES. A screenshot of a computer

  AI-generated content may be incorrect.

**Product level:**

* **ESW Fulfilment Country ISO:** At the product level, this attribute is used to store the country code where the product is available, such as "US". Only one country can be stored in this attribute. By default, it will be an empty string. 

**Product lineItem level:**

* **ESW Fulfilment Country ISO:** At the line item level, this preference will show the product origin
* **ESW Delivery Option:** At the line item level, this preference will show the delivery option coming from ESW order confirmation payload.

The above configurations will be display as below under product line item:



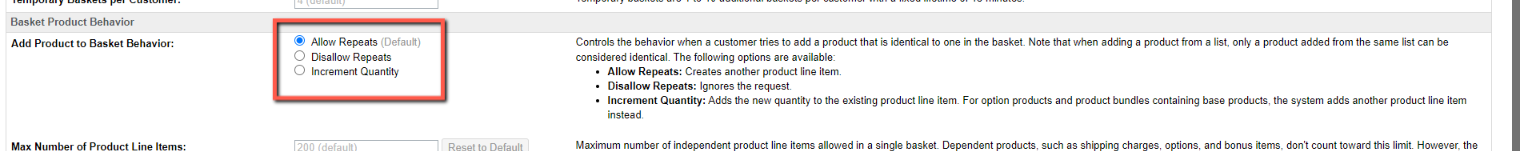
* 1. **Import services**

Import services from **link\_eshopworld\Metadata\services.xml** and configure the credentials of the following service:

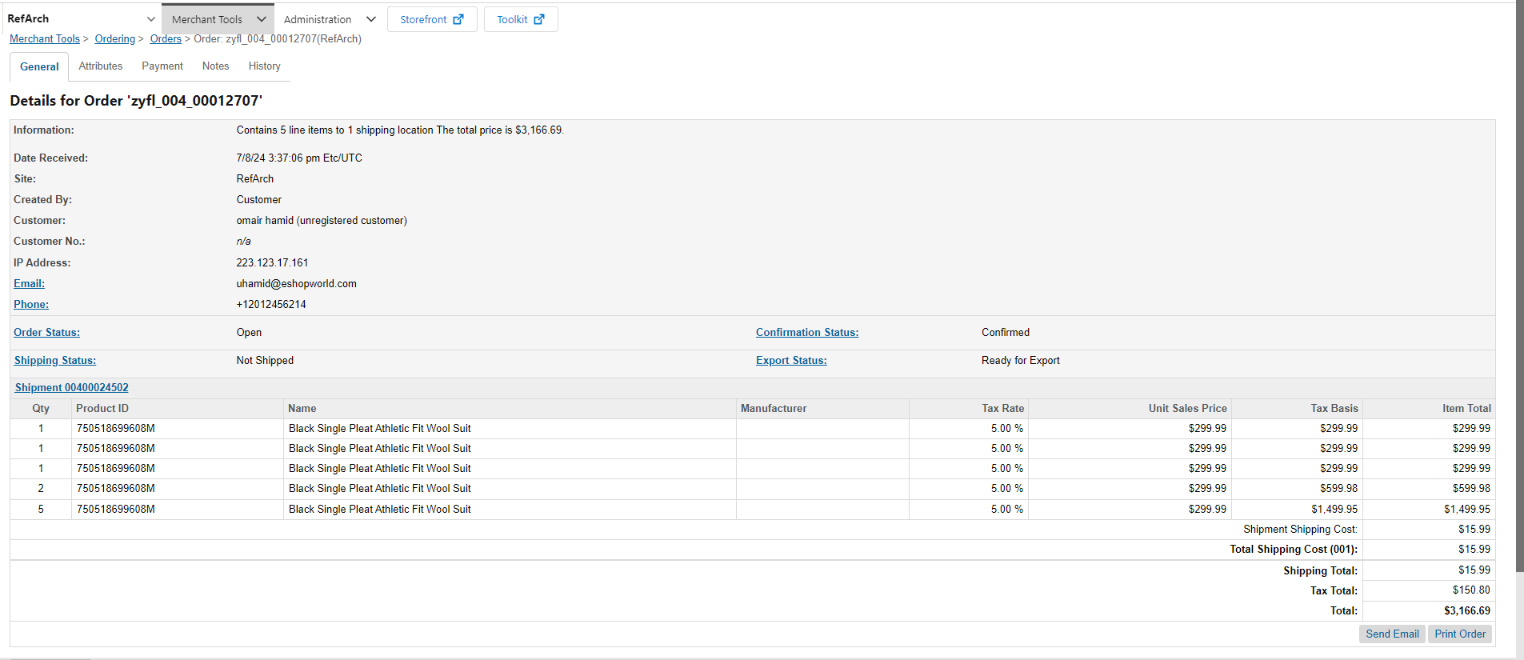
* **EswOcapiDataAuthService:**
  + This service is utilized for generating data OCAPI tokens.
  + **URL:** URL to get data API token for OCAPI , default is: https://account.demandware.com/dw/oauth2/access\_token
  + **User:** BM user who have permission to update product’s inventory
  + **Password:** Password of BM userA screenshot of a computer

    AI-generated content may be incorrect.
* **EswMoInventorySync**
  + This service use to update the inventory in SFCC from multi origin API in case there is no enough inventory in SFCC during add to cart process.
  + **URL:** OCAPI URL to update the product inventory, default is: https://{hostname}/s/-/dw/data/v23\_1/inventory\_lists/{inventory\_ID}/product\_inventory\_records/{productID}, please only change {hostname} to your instance URL and v23\_1 to your OCAPI version, and {inventory\_ID} if needed, {inventory\_ID} and {productid} will be replace automatically at code lavel
  + **User:** Should set to empty
  + **Password:** should set to emptyA white background with black text

    AI-generated content may be incorrect.
  1. **BM Configurations**

Make sure that Allow Repeat is enabled from the BM Merchant “Tools > Site Preferences > Baskets”  


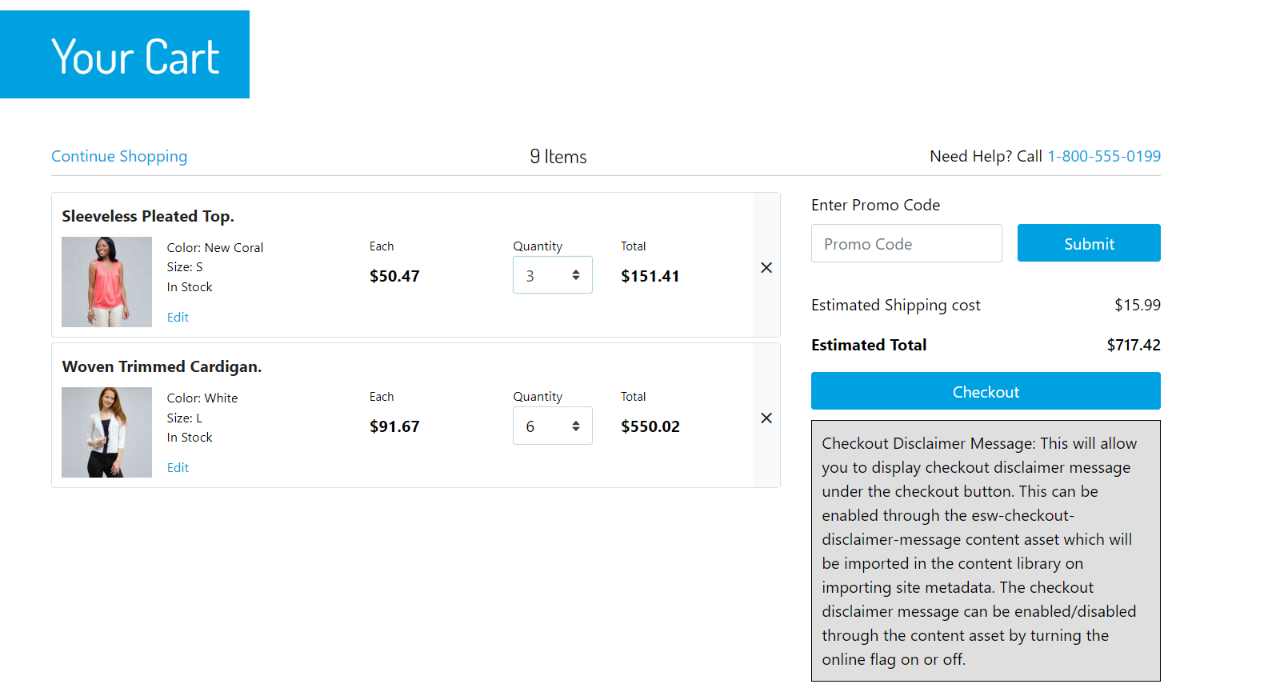
# **BM order detail page**

On the order detail page in the BM, the line items will show separately for a single product if origin is different.  


# **Storefront implementation**

### **Product listing on cart and order detail page**

On the cart and order detail pages, shoppers will see the Out of Box (OOB) behaviour of SFCC. This means that for products with the same product IDs, shoppers will see cumulative quantities and prices. This implementation only affects the view and does not alter the basket or order JSON structures. In the order and basket, separate line items will still be maintained if the product origins differ.



The above screenshot is from SFRA. SiteGenesis and PWA UI will adhere to their respective OOB UI implementations.

### **Remove line item from basket**

If a shopper removes any item from the basket on the storefront, the ESW cartridge will retrieve all line items matching that product ID and remove each one.

# **Adding logic to fetch products from multiple origins**

The ESW SFCC Cartridge provides a helper function that retailers can extend to send the product origin and quantity information. The purpose of this function is to allow the retailer to create their logic for fulfilment decisions and provide a response giving the origin of each cart item & qty.

Retailers can override a helper function getProductOriginDetails from file \link\_eshopworld\cartridges\int\_eshopworld\_core\cartridge\scripts\helper\eswMultiOriginHelper.js

The basic function definition is as follows:

/\*\*\*\*\*\*\*\*\*Function's Input\*\*\*\*\*\*\*\*\*/

// Cart Contents

var productsWithQuantities = { productId: 'pid1', quantity: 5 };

// Function's content

/\*\*

\* Return the origin and inventory information against products

\* @param {Object} productsWithQuantities - productsWithQuantities

\* @param {string} shopperCountryIso - shopperCountryIso e.g. "IE"

\* @return {Object} - productOriginDetails object

\*/

function getProductOriginDetails(productsWithQuantities, shopperCountryIso) {

// Retailer's Logic to retrieve origin country information for each product

// ...

/\*\* \*\*\*\*\*\*\*If the required quantity isn’t available at one location\*\*\*\*\*\*\*\*\*/

// In cases where the required quantity exceeds availability in one location,

// the remainder is sourced from alternative locations (see productId: pid1)

/\*\* \*\*\*\*\*\*\*If the originIso is not available for a product\*\*\*\*\*\*\*\*\*/

// If a product lacks an origin, retailer will set originIso to NULL

// ESW cartridge will check product level attribute "eswFulfilmentCountryIso"

// and Add originIso from this attribute if availble, (See productId: 'pid3')

/\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Function's Output\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// The function returns an object with product ids, origins and quantities or empty array ([])

// return [

// { productId: 'pid1', originIso: 'IE', quantity: 2 },

// { productId: 'pid1', originIso: 'FR', quantity: 3 },

// ] | [];

let multiOriginResponse = [];

// If the originIso is not available for products

let nonOriginProductIds = getNonOriginProductIds(multiOriginResponse);

for (let i = 0; i < nonOriginProductIds.length; i++) {

let nonOriginDwProduct = ProductMgr.getProduct(nonOriginProductIds[i]);

// Check, If the product has eswFulfilmentCountryIso attribute

let nonOriginProductOrigin = getOriginFromProductAttribute(nonOriginDwProduct);

if (nonOriginProductOrigin) {

multiOriginResponse = updateMultiOriginIsoInFromProductAttribute(multiOriginResponse, nonOriginDwProduct, nonOriginProductOrigin);

}

}

return multiOriginResponse;

}

The function will take one argument ({ productId: productId, quantity: totalQtyRequested }

) and retailer needs to make sure to return the proposed output, the sample output is as follows:

return [{ productId: 'pid1', originIso: 'IE', quantity: 2 },{ productId: 'pid1', originIso: 'FR', quantity: 3 }] || [];

# **Essential information to know**

* It's impossible to place an order if inventory is unavailable in SFCC, especially with SCAPI/OCAPI. OOB hooks validate inventory before any overridden hook executes. When adding to the cart, SFCC checks product inventory. If unavailable, an API call is sent to the MO service to update inventory. If the required quantity is still unavailable, the default out-of-stock behaviour is triggered.
* Retailers need to update the function **getProductOriginDetails** to ensure the output is: **[{ productId: 'pid1', originIso: 'IE', quantity: 2 }, { productId: 'pid1', originIso: 'FR', quantity: 3 }] || [].**
  + Where pid represents the product ID, originIso is the country code, and quantity denotes the available quantity in that country.