**Emarsys - Demandware Link Cartridge**

**19.10**

Integration Documentation

**Table of Contents**

[1.](#_heading=h.30j0zll) Summary 6

[2.](#_heading=h.1fob9te) Component overview 6

[2.1.](#_heading=h.3znysh7) Functional overview 6

[2.2.](#_heading=h.tyjcwt) Use cases 8

[2.2.1](#_heading=h.3dy6vkm) Global footer newsletter subscription 8

[2.2.2](#_heading=h.1t3h5sf) Newsletter subscription during the checkout process 9

[2.2.3](#_heading=h.4d34og8) Newsletter subscription from *My account* section 11

[2.2.4](#_heading=h.2s8eyo1) Transactional emails 11

[2.2.5](#_heading=h.17dp8vu) Emarsys Predict JavaScript Recommendations 12

[2.2.6](#_heading=h.3rdcrjn) Initial database load 15

[2.3.](#_heading=h.26in1rg) Compatibility 16

[2.4.](#_heading=h.lnxbz9) Limitations, constraints 16

[2.5.](#_heading=h.35nkun2) Privacy, Payment 16

[3.](#_heading=h.1ksv4uv) Implementation guide 16

[3.1](#_heading=h.44sinio) Integration components 16

[3.2](#_heading=h.2jxsxqh) Setup 23

[3.3](#_heading=h.4i7ojhp) Configuration 23

[3.3.1](#_heading=h.2xcytpi) Business Manager Configuration 23

[3.3.2](#_heading=h.1ci93xb) Import site\_template.zip 26

[3.3.3](#_heading=h.3whwml4) Business Manager Modules 27

[3.3.4](#_heading=h.2bn6wsx) Site preferences configuration 28

[3.3.5](#_heading=h.qsh70q) Emarsys Predict Export Attributes Configuration 37

[3.3.6](#_heading=h.3as4poj) Emarsys SmartInsight Export Attributes Configuration 39

[3.3.7](#_heading=h.2p2csry) Emarsys Catalog and SmartInsight Site Preferences 41

[3.3.8](#_heading=h.147n2zr) Emarsys Services 43

[3.4](#_heading=h.3o7alnk) Job Schedules 44

[Emarsys-Setup](#_heading=h.23ckvvd) 44

[Emarsys-ExportOrders](#_heading=h.ihv636) 47

[Emarsys-Init-Database](#_heading=h.32hioqz) 50

[Emarsys-ExportCatalog](#_heading=h.41mghml) 53

[Emarsys - OrderStatusСhangeNotification](#_heading=h.vx1227) 56

[3.5](#_heading=h.4f1mdlm) Custom code 58

[3.5.1](#_heading=h.2u6wntf) Global footer subscription integration 58

[3.5.2](#_heading=h.19c6y18) Custom code inside core templates 58

[3.5.3](#_heading=h.1mrcu09) Account subscription Business Manager configuration 59

[3.5.4](#_heading=h.46r0co2) Controller integration configuration 60

[3.5.5](#_heading=h.sqyw64) Shipping confirmation emails integration 64

[3.5.6](#_heading=h.1rvwp1q) Initial database load integration 64

[3.5.7](#_heading=h.4bvk7pj) Emarsys Predict 64

[3.5.8](#_heading=h.2r0uhxc) Emarsys Loyalty Wallet 75

[3.6](#_heading=h.1664s55) Testing 76

[4.](#_heading=h.3q5sasy) Operations, maintenance 76

[4.1.](#_heading=h.25b2l0r) Data storage 76

[4.2.](#_heading=h.kgcv8k) Availability 78

[4.3.](#_heading=h.34g0dwd) Support 78

[5.](#_heading=h.1jlao46) User guide 78

[5.1.](#_heading=h.43ky6rz) Business manager 79

[5.1.1.](#_heading=h.2iq8gzs) Newsletter subscription box – global footer 79

[5.1.2.](#_heading=h.xvir7l) Newsletter subscription – checkout process 80

[5.1.3.](#_heading=h.3hv69ve) Newsletter subscription – My account page 81

[5.1.4.](#_heading=h.1x0gk37) Transactional emails – order confirmation 82

[5.1.5.](#_heading=h.4h042r0) Transactional emails – shipping confirmation 83

[5.1.6.](#_heading=h.2w5ecyt) Initial Database Load – configuration screen 85

[5.1.7.](#_heading=h.1baon6m) Send Order Confirmation Email – send email 88

[5.1.8.](#_heading=h.khdul5v4bvy9) External events page 88

[5.2.](#_heading=h.3vac5uf) Storefront functionality 91

[5.2.1](#_heading=h.2afmg28) Newsletter subscription box – global footer 91

[5.2.2](#_heading=h.pkwqa1) Newsletter subscription during checkout process 92

[5.2.3](#_heading=h.39kk8xu) Newsletter subscription/unsubscription – My Account 93

[5.2.4](#_heading=h.1opuj5n) Demandware landing pages 94

[5.3.](#_heading=h.48pi1tg) Emarsys smartinsight 96

[5.3.1](#_heading=h.2nusc19) Introduction 96

[5.3.2](#_heading=h.1302m92) Purpose 96

[5.3.3](#_heading=h.3mzq4wv) BM Settings for Emarsys SmartInsight attribute configuration for export 97

[5.3.3.1](#_heading=h.2250f4o) Emarsys SmartInsight Extension Configuration 97

[5.3.3.2](#_heading=h.haapch) Job for SmartInsight that exports all orders in CSV format and upload to FTPS server 99

[5.4.](#_heading=h.319y80a) Emarsys ExportCatalog 102

[5.4.1](#_heading=h.1gf8i83) Introduction 102

[5.4.2](#_heading=h.40ew0vw) Purpose 102

[5.4.3](#_heading=h.2fk6b3p) BM settings for Emarsys Catalog 103

[5.4.3.1.](#_heading=h.upglbi) BM Settings for Emarsys Catalog Configuration 103

[5.4.3.2.](#_heading=h.3ep43zb) BM Settings for Emarsys Catalog custom object 108

[5.4.3.3.](#_heading=h.1tuee74) Job for Emarsys ExportCatalog for export of the product catalog in CSV format and upload to FTP location 108

[5.5.](#_heading=h.4du1wux) Java script tracking code and java script recommendations 111

[5.5.1.](#_heading=h.2szc72q) Introduction 111

[5.5.2.](#_heading=h.184mhaj) JavaScript recommendation code 111

[5.5.3.](#_heading=h.3s49zyc) Settings for JavaScript recommendation code 111

[5.5.4.](#_heading=h.279ka65) Settings for JavaScript tracking code 114

[6.](#_heading=h.meukdy) Known issues 116

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  | 25.12.2019 | Phase2 installation guide | 1.0 |
|  | 24.03.2020 | Update guide jobs and custom code. | 2.0 |
|  |  |  |  |

# Summary

Founded in 2000, Emarsys platform integrates customer intelligence, personalization, predictive recommendations, and automation across channels. Emarsys serves more than 1,200 clients in 140 countries. Among its clients are eBay, ToysRUs, Yahoo 7, Sky, and Volvo.

Retailers need to maximize customer engagement and conversion rates by providing highly customized customer experience and recommendations. By implementing the Emarsys solutions to their eCommerce platform, merchants will be able to collect data from the end user interaction with the platform, gain insights into their behavior and reach the customers through email campaigns and tailored recommendations.

The cartridge will offer merchants the ability to use the powerful Emarsys capabilities on the Demandware storefronts in order to increase conversion rates.

# Component overview

## Functional overview

The integration cartridges use default locale – en. The Emarsys integration cartridges will allow the merchant to:

* Use the Emarsys Business Manager module extension to make the necessary configurations for the following:
  + Newsletter subscriptions methods (global footer subscription, checkout process subscription and my account subscription):
    - Define the opt-in strategy for each newsletter subscription method (please see below details regarding the opt-in strategies)
    - Configure external events to be triggered once the end user has subscribed
  + Transactional emails (order and shipping confirmation emails)
    - Define and map the fields that will be added to the transactional emails
    - Configure the external event that will trigger the transactional email
  + Catalog configuration (Product data feed)
    - Define and map the fields that will be added to the product export feed
    - Choose the type of catalog product export
      * Export only main product
      * Export product variations
  + SmartInsight configuration (Order export feed)
    - Define and map the fields that will be added to the order export feed
  + Initial database load
    - Define the mapping between Demandware and Emarsys fields – this mapping will be used to create the auto-import profile (calling an Emarsys endpoint) and to create the CSV files that will contain the customer information. These files are uploaded the Emarsys WebDAV so that the Emarsys platform can process them and import the contacts into the system.
* Allow end users to subscribe to newsletters from the website storefront using the subscription methods:
  + Global footer newsletter subscription
  + Newsletter subscription during the checkout process
  + Newsletter subscription from *My account* section for registered users
* Allow registered end users to unsubscribe from newsletters

As stated above, the merchant has the option to set the desired opt-in strategy for each one of the three storefront subscription methods. The available values for the opt-in strategy are the following:

* + Single opt-in – the customer records gets the system field opt-in in Emarsys Suite set to true immediately.
  + Double opt-in – the customer (end user) needs to click a link in a confirmation email in order to get the system field opt-in in Emarsys Suite set to true.

External events – an external event is an Emarsys instance. Every external event has ID that can be used to run some programs or send emails using Emarsys API. External events used not only to subscribe customers for newsletters. There are also some other cases where external events are triggered:

* + On **Contact Us** page after form is submitted
  + In **Emarsys-OrderStatusСhangeNotification** job to notify Emarsys about shipped and cancelled orders
  + On **My Account Login** page when customer sends email to reset password
  + On **Create Account** page when new customer account is created (registration form submitted)

JavaScript tracking code was embedded in Phase2 of the Emarsys code. The analysis generates product recommendations and maps personal interests of users based on browsing information collected from website (product views, searches, conversions). The information is collected by a set of JavaScript snippets that were embedded into website page templates.

These JavaScript commands are pushed to Emarsys Predict:

* cart: on the page where the cart is shown
* category: on pages which lists products in a category
* purchase: on the order confirmation page
* setCustomerId: on every page as soon as the customerid is known
* setEmail: on every page as soon as the email address is known
* view: on every product detail page
* searchTerm: on the search results page

## Use cases

Please find listed below the use cases for the Emarsys cartridge integration.

### Global footer newsletter subscription

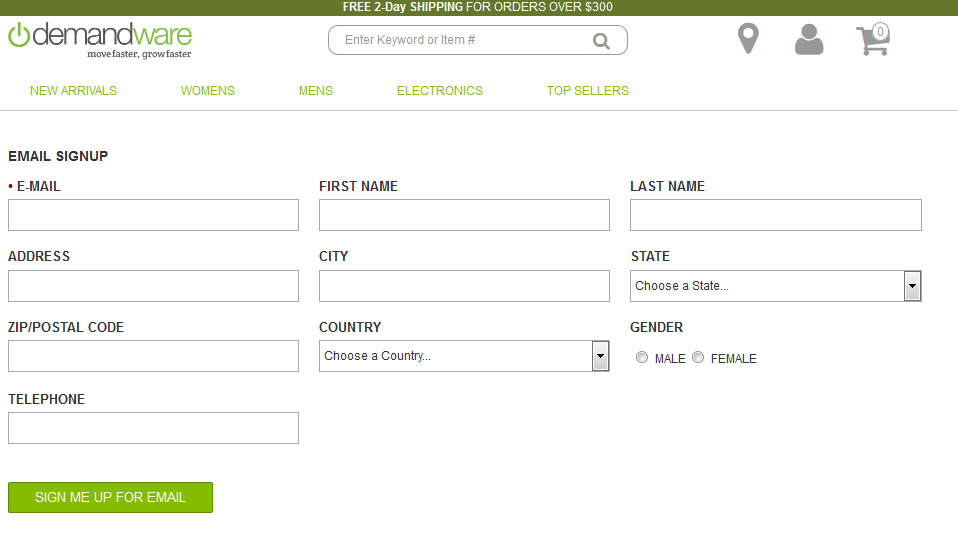
The end user can enter his email address directly into the newsletter registration field in standard SiteGenesis.



If the end user enters his e-mail address and clicks the button, based on the opt-in strategy set by the merchant for this subscription method, one of the following scenarios will be followed:

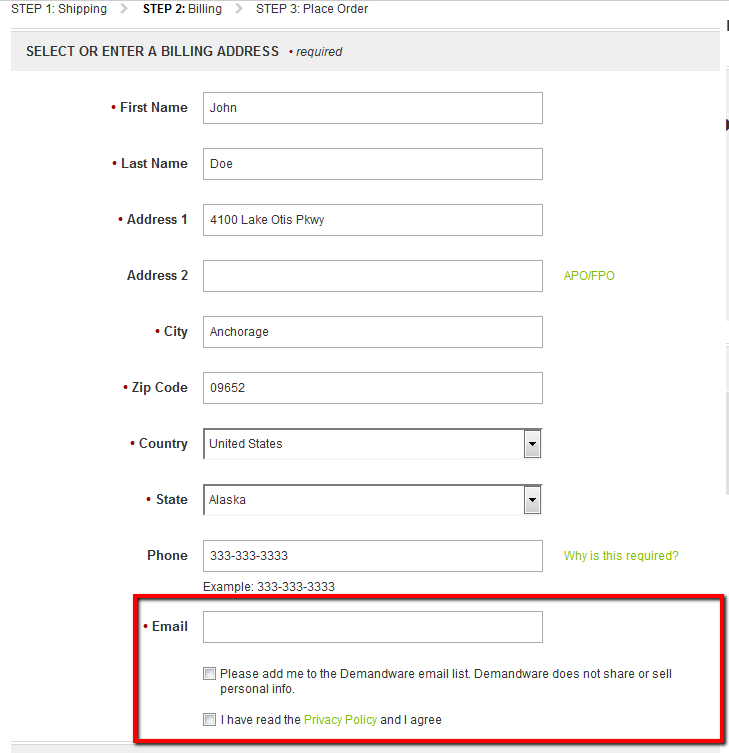
* + - merchant has decided to use the Single Opt-in strategy, the end user is lead to the thank you page.
    - merchant has decided to use the Double-Opt-in strategy, the end user will be directed to a Data Submitted Page.

If the end user clicks the submit button without entering the email address, he will be redirected to the sign-up page.



### Newsletter subscription during the checkout process

Using this method, the end user can subscribe for newsletters during the checkout process, directly from the billing page.

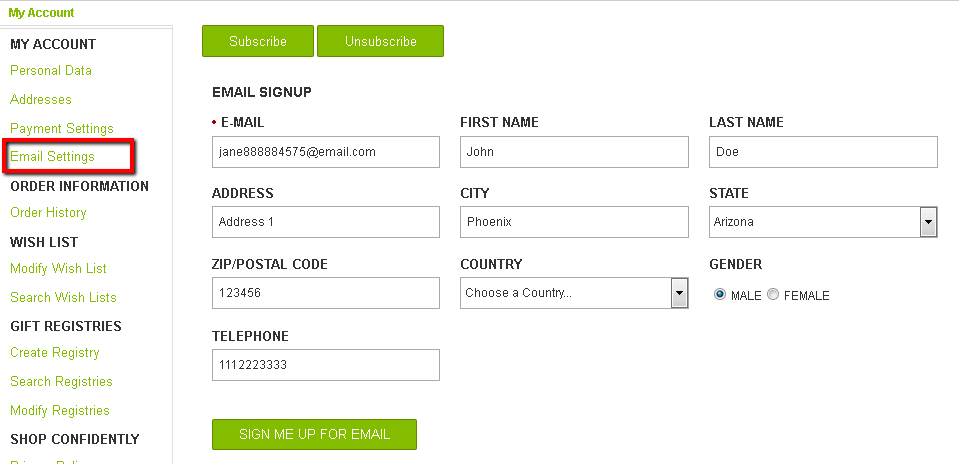


The email field is a required field on the billing page in standard SiteGenesis functionality. If the user enters his emails address and selects the *Please add me to the Demandware email* *list* checkbox, based on the opt-in strategy set by the merchant in Business Manager for this subscription method, the user record will be created on Emarsys side and the opt-in field will be set directly to true (Single Opt-in strategy) or after the customer clicks a link in a confirmation email (Double Opt-in strategy).

If the user does not opt-in in adding him in the E-Mail list all customer data are transferred to Emarsys, so that Emarsys can create this user in his database. Even though the user is created in Emarsys the status for E-Mail sign up (Opt-in field) is still inactive.

### Newsletter subscription from *My account* section

There is also the option for a registered customer to subscribe for newsletter from the *My account* section in Demandware storefront. Just like for the previous two subscription methods, the my account subscription process will consider the configuration for the Opt-in strategy (Single Opt-in / Double Opt-in). Also, the registered user has the option of unsubscribing from newsletter by clicking a button inside his account.



### Transactional emails

There are two types of transactional emails:

* + - Order confirmation email
    - Shipping Confirmation email

Order confirmation emails – as soon as an end user finishes a transaction, the order data is transmitted to Emarsys Suite and the corresponding external event is triggered so that the confirmation email to be sent. When triggering the external event, transactional information is transferred with the API call. The transactional information sent to Emarsys is based on the configuration made by the merchant for order emails.

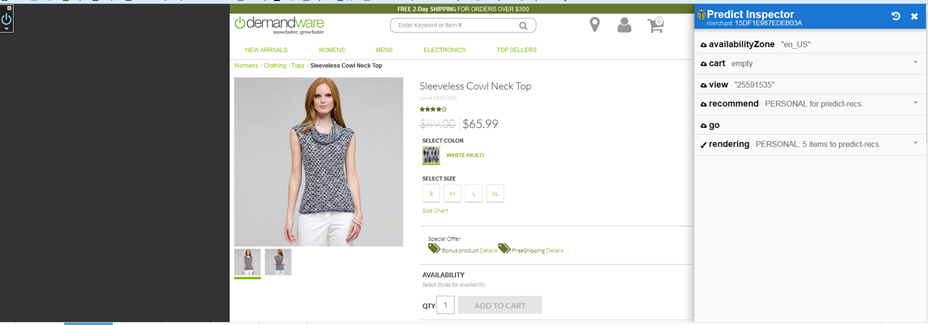
Shipping confirmation email – as soon as a shipment to an end user is finished, a shipping confirmation email is sent. The approach is similar to the one used for order confirmation emails.

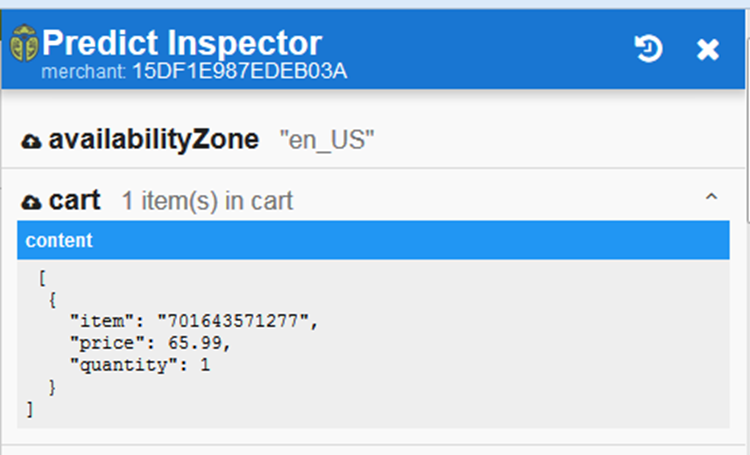
### Emarsys Predict JavaScript Recommendations

Emarsys Predict is being used to recommend products based on a self-learning algorithm. Recommended products can be rendered within shop pages and within emails. Emarsys has own system ScarabResearch that renders the recommendations.

The user browsing information is pushed into Emarsys on the following pages:

* Cart page: list of items in the visitor's shopping cart
* Category page: category currently browsed by visitor
* Order Confirmation Page: list of items purchased by the customer
* Search Page: Push search term entered by visitor
* Product Page: Push the unique ID of the product being browsed

****



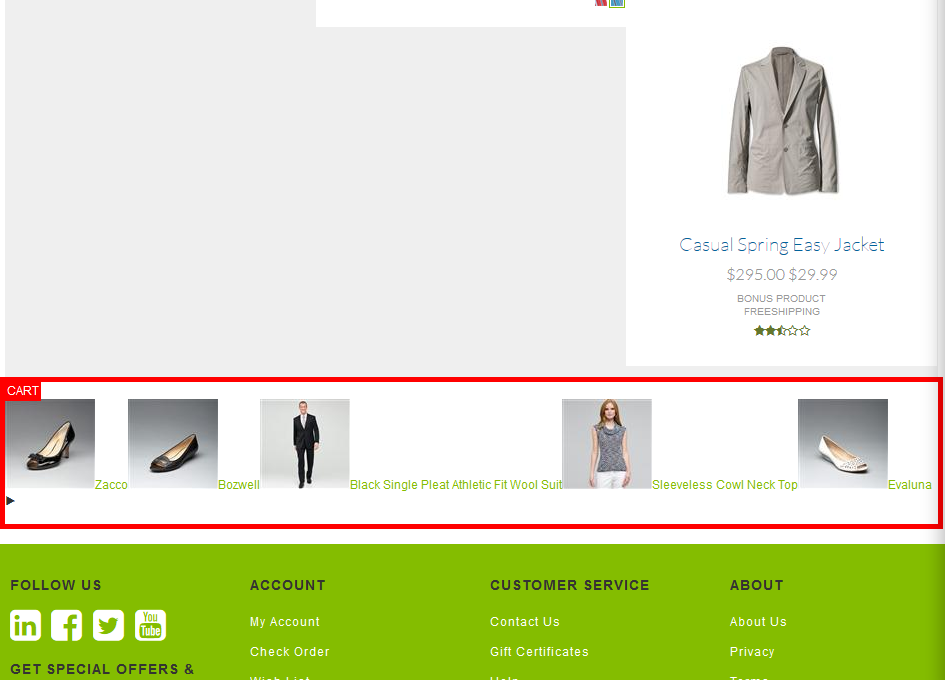
While browsing the site the user should be able to see the recommendations at the following pages:

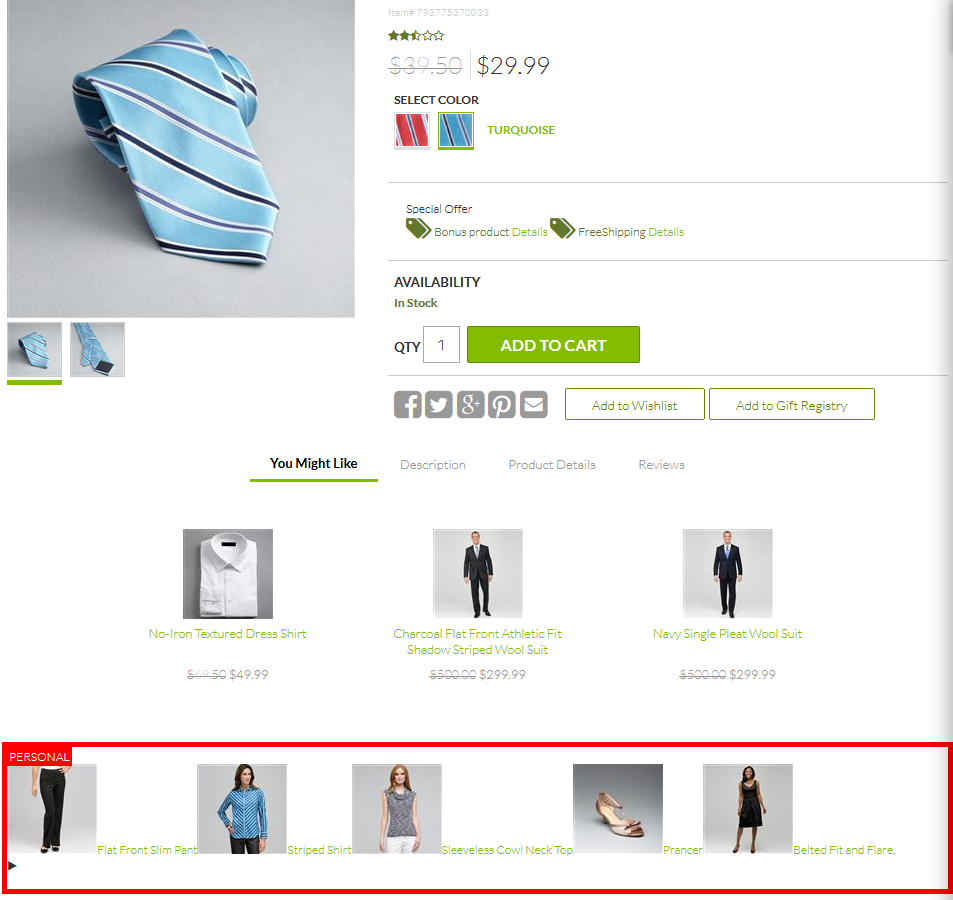
* Product detail page
* Home page
* Cart page
* Search results page
* Category page
* Order confirmation page

Recommendations are displayed based on previous user actions from the recommendation catalog exported into Emarsys.

**Use Case:**

1. User enters to the site
2. Customer visits product detail page of some product
3. Customer adds product to the basket
4. Customer goes to the cart page
5. The list of recommendations recommended by Emarsys Scarabresearch in displayed





### Initial database load

Other than newsletter subscription and transactional emails functionality, the cartridge offers the merchant the option of loading the existing Demandware customers into Emarsys Suite. Using the *Database load configuration* extension, he defines the customer data that will be sent. Once the configuration is saved, the merchant needs to run the *Initialize Emarsys Database* Job schedule to create the CSV file containing customer information, create an auto-import profile on Emarsys side and finally upload the generated file on Emarsys WebDAV so it can be further processed by the system.

## Compatibility

The integration is available since Demandware 19.10.

## Limitations, constraints

* In order to avoid Demandware quota violations, the initial database load feature will load the existing customers into Emarsys Suite in bulks of configurable sizes. By default, the size of bulk is set to 10000 customers. However, this setting can be overwritten by navigating to Administration > Operations > Jobs > Emarsys-InitDatabase > ExportCustomerInfo and setting the value for the Profiles export threshold field. Please note that setting a value higher than 20000 will cause a script failure. Also, trying to export a high number of profiles at a time might affect the performance, so be very careful when updating this custom field.
* Managing multiple currencies and localization is out of scope for the current version of the cartridge
* DW doesn’t has build-in OMS so we suggest including returned items in a feed generated by the external OMS.

## Privacy, Payment

The cartridge uses customer profile and transfers it to the Emarsys Suite. All data is transferred over a secure HTTPS connection. The integration does not process any customer credit card data.

For the phase2 customer order and product information are passed to the Emarsys Suite via SFTP server connection.

# Implementation guide

## Integration components

The Emarsys integration consists of two Demandware cartridges, one used for communicating with the Emarsys platform and the other one used to extend the standard Business Manager functionality with a custom module – this module will allow the merchant to customize the newsletter subscriptions behavior and to configure the Demandware fields that will be added to the transactional emails. Also, the Business Manager module extension will allow the merchant to set the fields that will be added to the CSV file that is used for the initial DB load feature. Please find listed below the cartridge components.

Demandware cartridges

1. ***Integration Cartridge***

* *int\_emarsys*

***Forms***

* *default/emarsyssignup.xml*
* *default/newsletter\_unsubscribe.xml*

***Models***

* *loyaltyCustomerData.js*

***Scripts***

* *js/emarsys.js*
* *js/emarsysAnalytics.js*
* *js/emarsysSubscription.js*
* *js/util.js*
* *scripts/hooks/emails.js*
* *scripts/service/emarsysService.js*
* *scripts/service/emarsysFTPService.js*
* *scripts/util/StepUtil.js*
* *scripts/helpers/emarsysAnalyticsHelper.js*
* *scripts/helpers/emarsysEventsHelper.js*
* *scripts/helpers/emarsysHelper.js*
* *scripts/helpers/emarsysResource.js*
* *scripts/helpers/ftpClientHelper.js*
* *scripts/helpers/scarabQueueHelper.js*
* *scripts/helpers/jobHelper.js*
* *scripts/helpers/emarsysEventsHelper.js*
* *scripts/helpers/triggerEventsHelper.js*
* ***JOBS***
  + *scripts/jobsteps/CreateAutoImportProfile.js*
  + *scripts/jobsteps/CreateExternalEvents.js*
  + *scripts/jobsteps/CreateSingleChoiceValueMapping.js*
  + *scripts/jobsteps/ExportCatalog.js*
  + *scripts/jobsteps/ExportCustomerInfo.js*
  + *scripts/jobsteps/ExportOrders.js*
  + *scripts/jobsteps/FtpUpload.js*
  + *scripts/jobsteps/GetExternalEventsJob.js*
  + *scripts/jobsteps/GetProfileFieldsJob.js*
  + *scripts/jobsteps/GetSourceID.js*
  + *scripts/jobsteps/OrderStatusСhangeNotification.js*
  + *scripts/jobsteps/UploadProfilesCSVToWebDAV.js*

***SCSS***

* *scss/\_mixins.scss*
* *scss/\_newsletter.scss*
* *scss/stylesheet.scss*

***Static***

* *static/default/css/stylesheet.css*
* *static/default/js/emarsys.js*
* *static/default/js/emarsysAnalytics.js*
* *static/default/js/emarsysSubscription.js*
* *static/default/js/util.js*

***Templates***

* *default/email/dbload\_notification.isml*
* *default/resources/emarsysresources.isml*
* *default/subscription/double\_optin\_thank\_you\_page.isml*
* *default/subscription/emarsys\_alreadyregistered.isml*
* *default/subscription/emarsys\_datasubmitted.isml*
* *default/subscription/emarsys\_disabled.isml*
* *default/subscription/emarsys\_emailsettings.isml*
* *default/subscription/emarsys\_error.isml*
* *default/subscription/emarsys\_thankyou.isml*
* *default/subscription/emarsyssignup.isml*
* *default/unsubscribe/account\_unsubscribe.isml*
* *default/unsubscribe/landing\_unsubscribe.isml*
* *default/loyaltyWallet.isml*
* *default/successjson.isml*

***Resources***

* *resources/dbload\_notification.properties*
* *resources/emarsys.properties*
* *resources/emarsysinfo.properties*
* *resources/forms.properties*
* *resources/locale.properties*
* *resources/newsletter\_unsubscribe.properties*

1. ***Controller cartridges***

* *int\_emarsys\_controllers*

***Controllers***

* *controllers/Account.js*
* *controllers/CustomerService.js*
* *controllers/EmarsysNewsletter.js*
* *controllers/Predict.js*

***Scripts***

* *models/* *AbstractModel.js*
* *models/* *EmarsysNewsletterModel.js*
* *models/* *FormModel.js*
* *util/Browsing.js*
* *util/Class.js*
* *util/Response.js*
* *views/View.js*
* *app.js*
* *guard.js*
* *meta.js*
* *object.js*
* *view.js*

***Templates***

* *default/subscription/userdataform.isml*

***Resources***

* *resources/emarsys.properties*

1. ***Business Manager Cartridge***

* *bm\_emarsys*

***Controllers***

* *EmarsysAdmin.js*
* *ExternalEvents.js*
* *NewsletterSubscription.js*

***Forms***

* *default/newsletterSub.xml*
* *default/sendOrderConfEmail.xml*

***Scripts***

* *js/components/showStatus.js*
* *js/components/dialogPopup.js*
* *js/dynamic.js*
* *js/externalEvents.js*
* *js/forms.js*
* *js/main.js*
* *js/tabs.js*
* *js/utils.js*
* *scripts/helpers/BMEmarsysHelper.js*

***SCSS***

* *scss/dialogPopup.scss*
* *scss/externalEvents.scss*
* *scss/global.scss*
* *scss/main.scss*
* *scss/notification.scss*
* *scss/tabs.scss*

***Static***

* *default/scss/dialogPopup.scss*
* *default/scss/externalEvents.scss*
* *default/scss/global.scss*
* *default/scss/main.scss*
* *default/scss/notification.scss*
* *default/scss/tabs.scss*
* *default/js/dynamic.js*
* *default/js/externalEvents.js*
* *default/js/forms.js*
* *default/js/main.js*
* *default/js/tabs.js*
* *default/js/util.js*

***Templates***

* *default/application/inputfield.isml*
* *default/application/modules.isml*
* *default/components/dialogPopup.isml*
* *default/components/dynamic.isml*
* *default/components/errorPage.isml*
* *default/components/externalEventsTable.isml*
* *default/components/modules.isml*
* *default/components/newsletterConfiguration.isml*
* *default/components/pageHeader.isml*
* *default/components/sendOrderConfEmail.isml*
* *default/components/tabs.isml*
* *default/externalEvents.isml*
* *default/mainPage.isml*
* *default/pageWrapper.isml*

***Resources***

* *resources/ dialogPopup.properties*
* *resources/dynamicTemplate.properties*
* *resources/errorMessages.properties*
* *resources/externalEvents.properties*
* *resources/forms.properties*
* *resources/newsletter.properties*
* *resources/pageDescription.properties*
* *resources/pageTitle.properties*
* *resources/* *tabs.properties*

***Extension file***

* *bm\_extensions.xml*

## Setup

#### Emarsys cartridge installation

To install the Emarsys integration for the first time on your instance you need to follow the next steps:

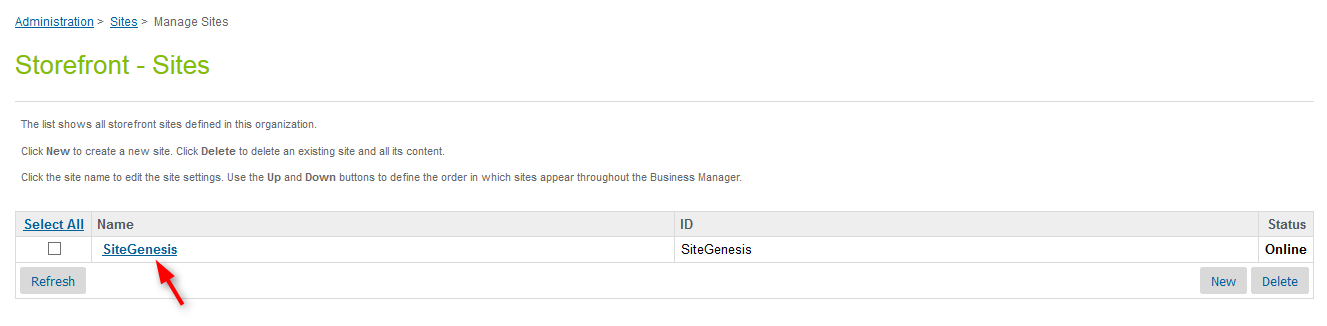
* 1. Download and extract Emarsys integration archive to your local file system – e.g. the cartridge folder of your project
  2. Inside site\_template folder, modify the SiteGenesis folder name and SiteGenesis references inside the xml files to match your site name.
  3. Import the cartridge into your workspace and link it to the Demandware Server Connection.
  4. Import site\_template.zip into your instance (please see more details in the following sections of the document).
  5. Assign the *int\_emarsys**, int\_emarsys\_controllers* and *bm\_emarsys* cartridges to sites that you want to integrate with Emarsys – make sure that *int\_emarsys and* *int\_emarsys\_controllers* cartridges are placed before the storefront cartridge (detailed below).
  6. Assign the *int\_emarsys**, int\_emarsys\_controllers* and *bm\_emarsys* cartridges to the Business Manager site (detailed below).

## Configuration

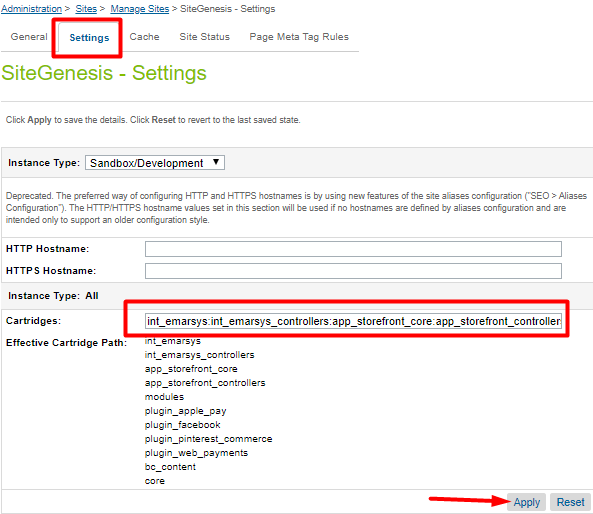
Once you have the cartridges installed, you will need to make the necessary configurations in order for the integration to work.

### Business Manager Configuration

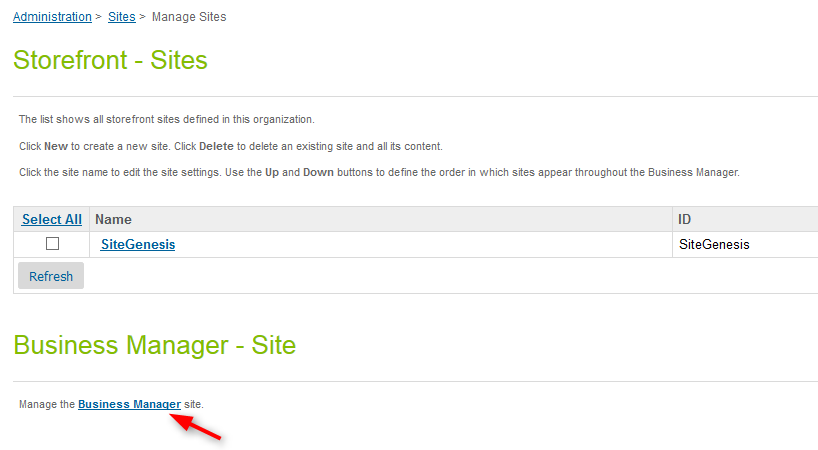
1. Login to **Business Manager**.
2. Go to **Administration > Sites > Manage Sites**
3. Select the site that you would like to integrate with Emarsys



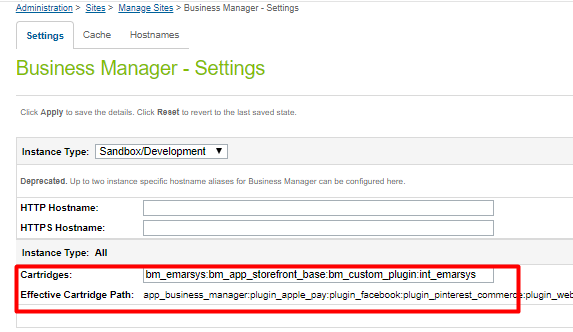
1. Go to the **Settings** tab
2. Add the cartridges (*int\_emarsys, int\_emarsys\_controllers* and *bm\_emarsys*) into the Cartridges field. Make sure that *int\_emarsys and int\_emarsys\_controllers* cartridges appear before the storefront cartridge.
3. Click on the **Apply** button.



1. Navigate back to **Manage sites** (using breadcrumbs or by clicking the **Back to list** button).
2. Click on **Business Manager** link.



1. Add the *int\_emarsys*and*bm\_emarsys*cartridges to the cartridges path.



1. Click the **Apply** button.

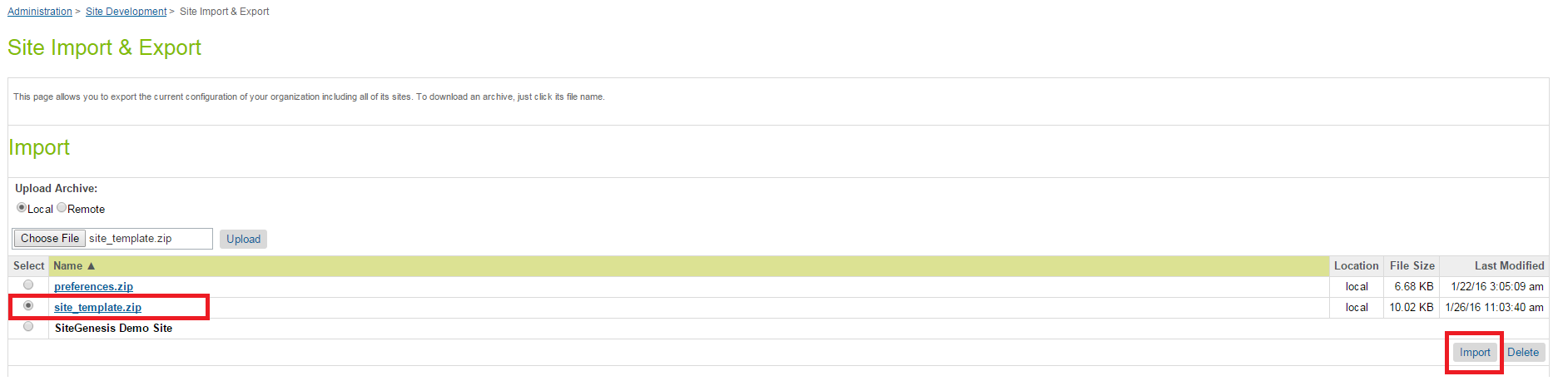
### Import site\_template.zip

Inside site\_template folder, modify the SiteGenesis folder name and SiteGenesis references inside the xml files to match your site name. Afterwards you’ll need to make a zip archive of the folder in order to be able to import it into Business Manager.

1. Login to Business Manager and navigate to **Administration > Site Development > Site Import & Export**;
2. From **Import > Upload archive** section upload the zip archive you’ve just created: *site\_template.zip*.



1. Upload the file.
2. Select the uploaded archive file and import it.

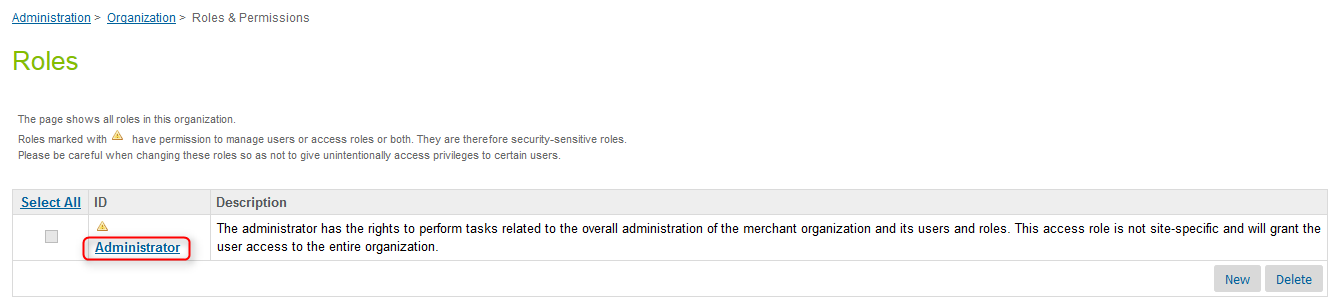


1. Once the archive has been uploaded, select it from the list of available files and hit the Import button from the bottom right corner.
2. Confirm that you want to import it, so that the import process can start.
3. Wait until the import job has successfully finished.

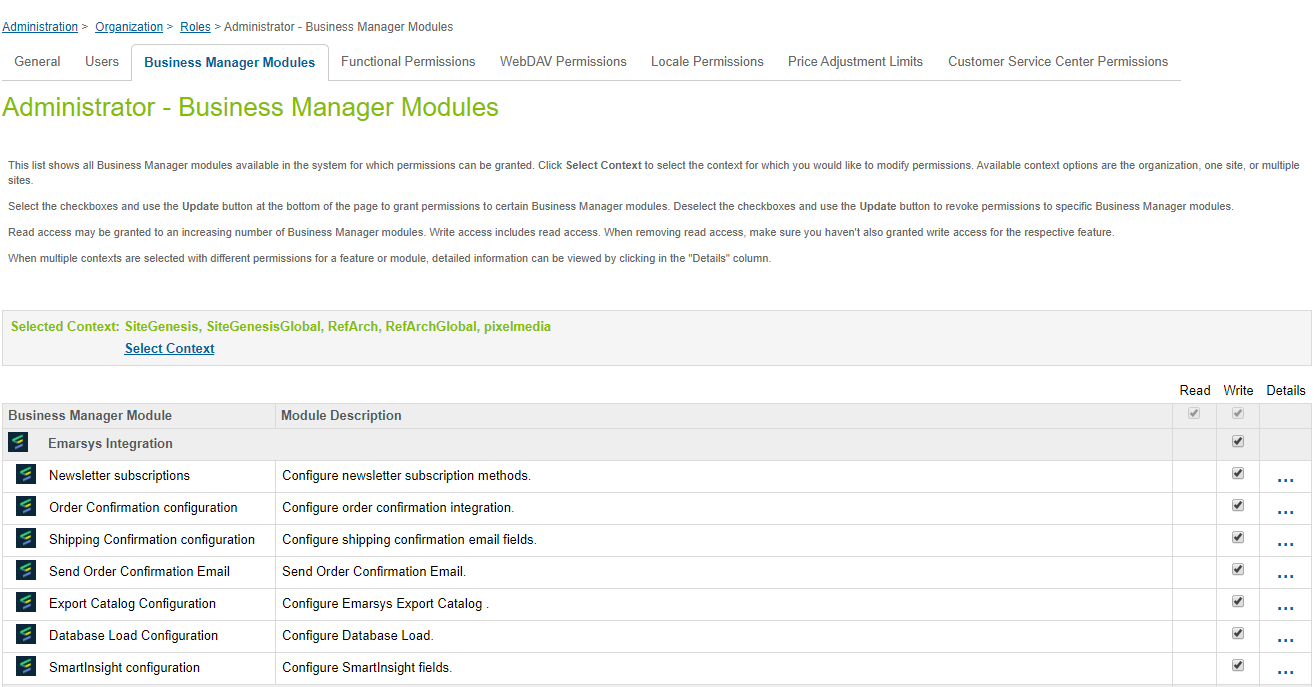
### Business Manager Modules

After the cartridge has been installed, you’ll have to activate the Emarsys Business Manager modules that will allow you to make the configuration for the integration. Please see below the steps that you need to follow in order to activate the modules.

1. Login to **Business Manager.**
2. Go to **Administration > Organization > Roles & Permissions**.
3. Select the **Administrator** role.

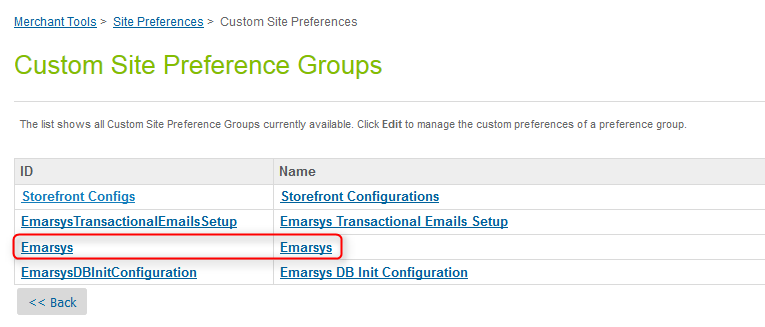


1. Switch to **Business Manager Modules** tab.
2. Open **Context** drop-down. Select your site from the context dropdown list.
3. Click the **Apply** button.
4. Enable needed Emarsys modules with checkboxes.
5. Click the **Update** button.



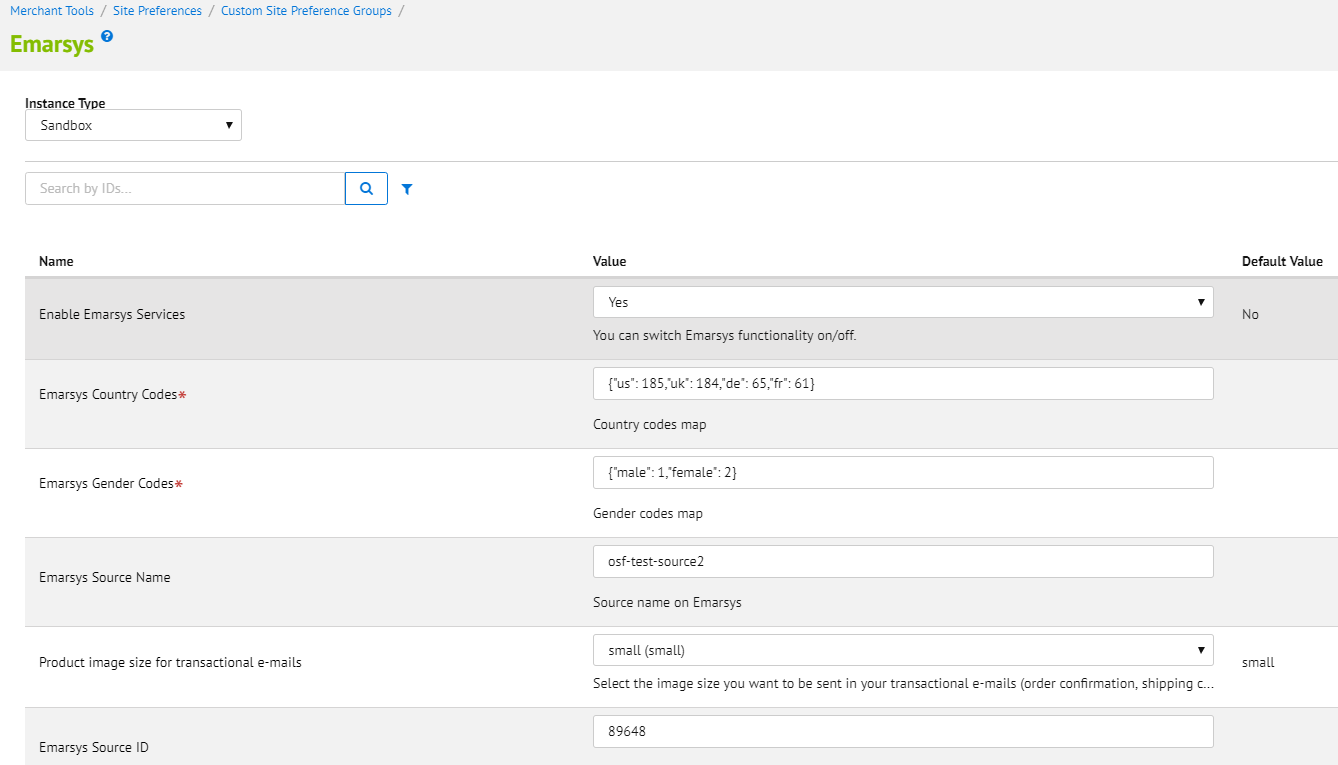
### Site preferences configuration

1. Login to **Business Manager.**
2. Go to **Merchant Tools > Site Preferences > Custom Site Preferences.**
3. Select **Emarsys.**



1. Set the following attributes:

* **Enable Emarsys Services** – this is switch Emarsys functionality on/off.
* **Emarsys Environment** – this should be set to the Emarsys suite that you will connect with. Make sure to omit the *https://* part (please check the screenshot below)
* **Emarsys Source Name** – this is the name of a source that was previously created in Emarsys.
* **Emarsys Country Codes –** JSON used to map the country IDs on Demandware side with the IDs on Emarsys side. If you need to add another country you’ll have to manually update the JSON with the correct key-value pair, where the *key* is the DW country ID and the *value* is the Emarsys country ID. Please check this link for the complete list of Emarsys country IDs: <http://documentation.emarsys.com/resource/b2c-cloud/contacts/fields/single-choice-fields/>
* **Emarsys Gender Codes** – automatically populated after metadata import





* **Emarsys Contact Fields Map –** JSON used to map account information and shipping address on Demandware side with the IDs on Emarsys side. If you need to add another fields to map you’ll have to manually update the JSON with correct key-value pair, where the *key* is ID on the Emarsys side and the *value* is the Profile/Address System Object attribute.

Example of the JSON for this field is:

{

"1": "firstName",

"2": "lastName",

"3": "emailAddress",

"4": "birthday",

"5": "gender",

"10": "address1",

"11": "city",

"12": "stateCode",

"13": "postalCode",

"14": "countryCode",

"15": "phone",

"17": "jobTitle",

"18": "companyName",

"46": "salutation"

}

This configuration will map first saved by customer address or shipping or billing address if any is available during checkout.

* **Emarsys Address Fields Map –** JSON used to map the rest of saved for every customer addresses on Demandware side with the IDs on Emarsys side. If you need to add another fields to map or map more addresses you’ll have to manually update the JSON with correct key-value pair, where the *key* is ID on the Emarsys side and the *value* is the Address System Object attribute. Example of the JSON for this field is:

{

"1": {

"1\_1": "firstName",

"2\_1": "lastName",

"3\_1": "emailAddress",

"10\_1": "address1",

"11\_1": "city",

"12\_1": "stateCode",

"13\_1": "postalCode",

"14\_1": "countryCode",

"15\_1": "phone",

"18\_1": "companyName"

},

"2": {

"1\_2": "firstName",

"2\_2": "lastName",

"3\_2": "emailAddress",

"10\_2": "address1",

"11\_2": "city",

"12\_2": "stateCode",

"13\_2": "postalCode",

"14\_2": "countryCode",

"15\_2": "phone",

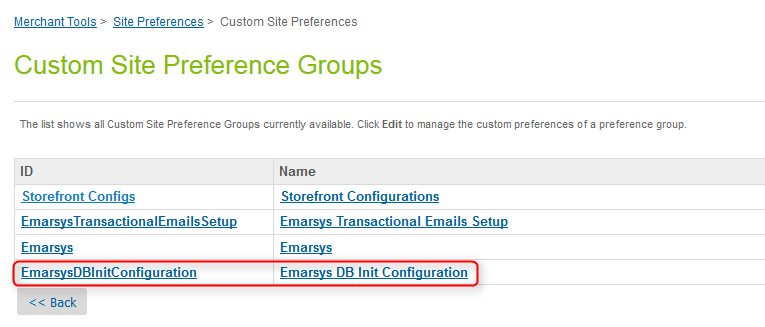
"18\_2": "companyName",

}

}

This configuration will map second and third saved by customer address if any. All fields like “1\_1”, “2\_1”, “2\_1” must be configured on the Emarsys side at first.

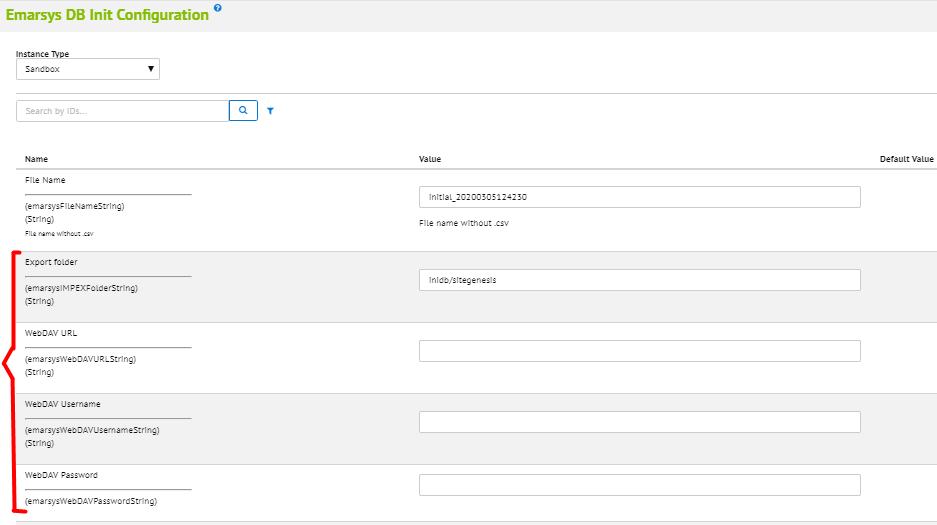
1. Click the **Apply** button.
2. Click the **Back** button.
3. Open **EmarsysDBInitConfiguration**.



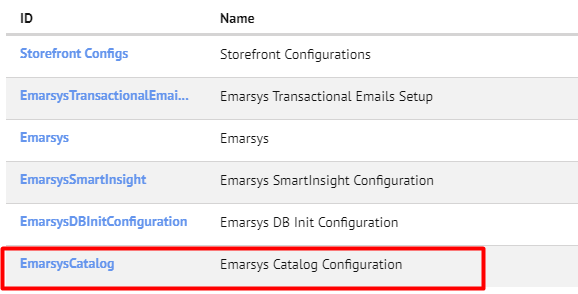
1. Set the following attributes:

* **Export folder** - represents the folder where the CSV file will be uploaded on Emarsys WebDAV
* **WebDAV URL**
* **WebDAV Password**
* **WebDAV Username**

1. Click on **Apply** button.

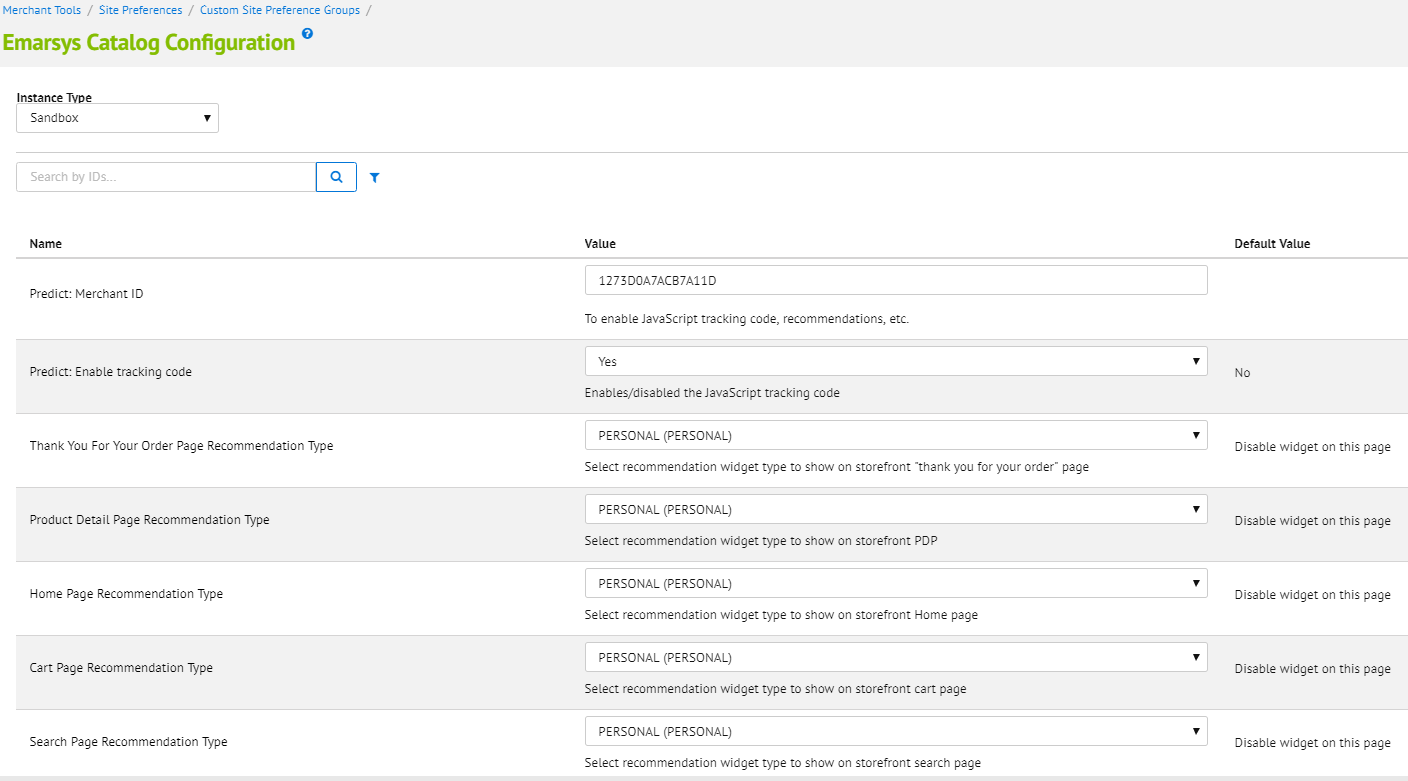


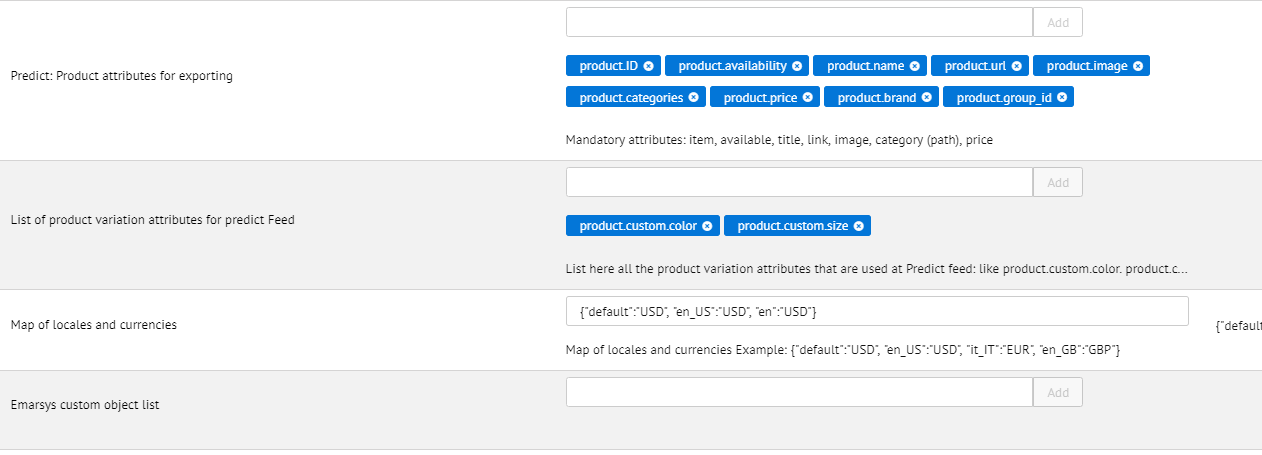
1. Open **EmarsysCatalog** configurations



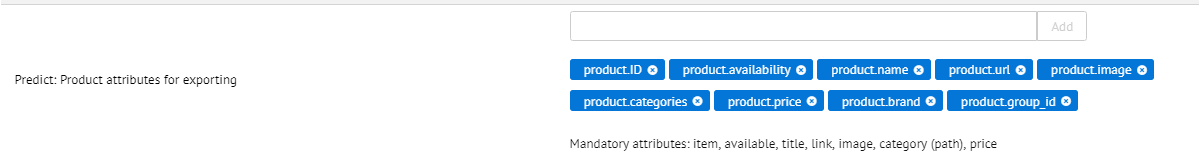
1. Set the following attributes

* **Predict Merchant ID** - represents the ID of the Predict merchant user. Should be taken from Emarsys dashboard.
* **Predict: Enable tracking code -** enable or disable js tracking codeat your site





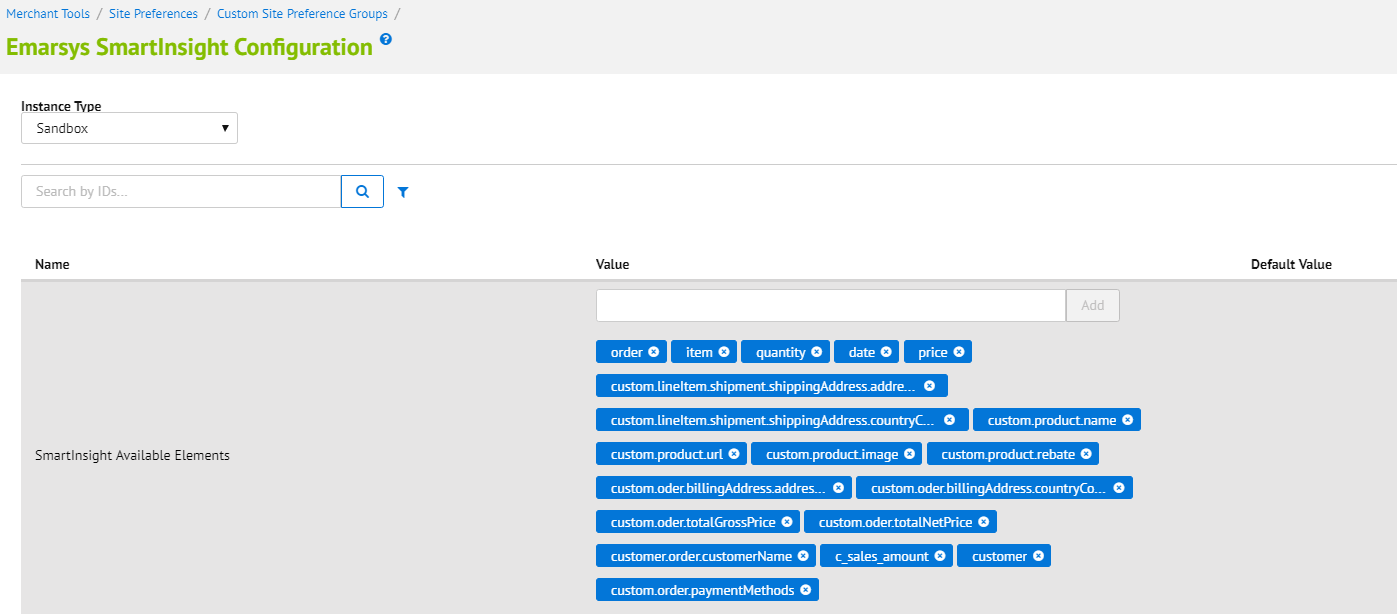
* **List of Product Variation attributes for predict feed –** list of product variation attributes used for the catalog job. All the product variation attributes should be putted here e.g:
  + product.custom.color
  + product.custom.size
* **Predict:** Product attributes for exporting – list of product attributes for export feed. Mandatory attributes:
  + Merchant will be able to add any Demandware product standard and custom attributes, e.g product.ID, product.availability, product.name, product.url, product.image, product.categories, product.price, product.onlineFlag, product.skinConcern, product.custom.color, product.custom.size, product.shortDescription, product.longDescription, product.pageTitle, product.taxClassID, product.brand,
  + Some mandatory attributes with specific syntax: url, image, price, availability, categories.
  + Please see the section **3.3.5 Emarsys Predict Export Attributes Configuration** for the detailed description

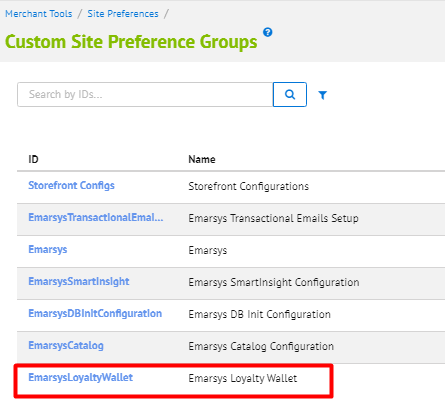


1. Open **EmarsysSmartInsight** configurations
2. Set the following attributes

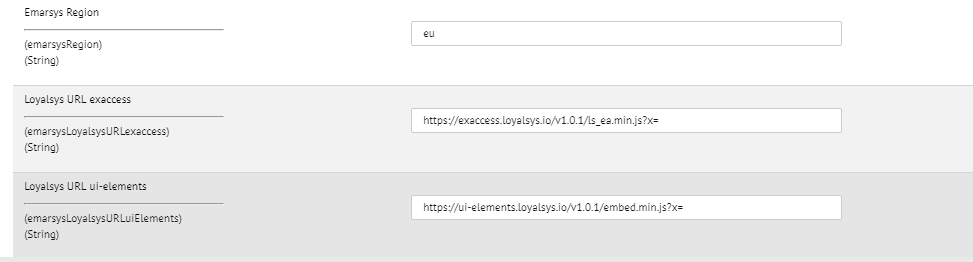
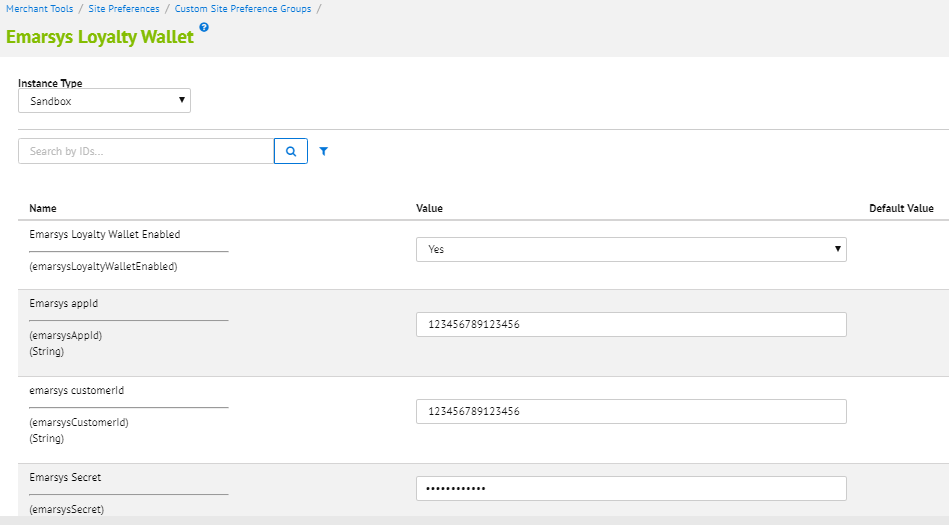
* **SmartInsight Available Elements –** List of available attributes for SmartInsight order export csv
  + The following attributes are required in csv
    - order– order number
    - date – date of the order (YYYY-MM-DD)
    - customer – unique customer id (this ID must be available in Emarsys Suite as well)
    - item – ProductId of the sold item (this Id must be in the products CSV as well!)
    - quantity – quantity of the ordered item
    - c\_sales\_amount – Price of the sold items (this value must be negative if an order was cancelled!)

Please see the section **3.3.6 Emarsys SmartInsight Export Attributes Configuration** for the detailed description



1. Open **EmarsysLoyaltyWallet** configurations  
   
2. Set the following attributes

* **Emarsys Loyalty Wallet Enabled**– this is switch Emarsys Loyalty Wallet functionality on/off.
* **Emarsys appId**
* **emarsys customerId**
* **Emarsys Secret –** server side secret from Loyalsys
* **Emarsys Region**
* **Loyalsys URL exaccess.** *Example:* https://exaccess.loyalsys.io/v1.0.1/ls\_ea.min.js?x=
* **Loyalsys URL ui-elements.** *Example:* https://ui-elements.loyalsys.io/v1.0.1/embed.min.js?x=



### Emarsys Predict Export Attributes Configuration

Emarsys Catalog is being used to recommend products based on a self-learning algorithm. Recommended products can be rendered within shop pages and within emails. To get this working Catalog requires a CSV product data feed which will be fetched from a remote location (HTTP / FTP / SFTP).

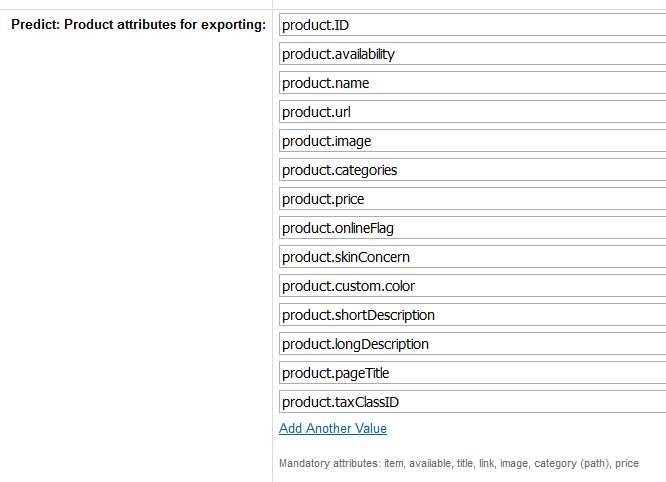
Emarsys Predict Attributes will be configurable under **emarsysPredictProductAttributes** site preference.

The few attributes are mandatory for the feed :

* product.ID
* product.availability
* product.name
* product url
* product.images
* product.categories
* product.price

If you need to add another attributes the following attributes are available:

* Standard product attributes
* Product custom attributes



### Emarsys SmartInsight Export Attributes Configuration

Emarsys SmartInsgiht attributes will be configurable under **emarsysSmartInsightAvailableElements** site preference. Merchant will be able to add product custom attributes like **custom.product.color**.

The column name of these additional columns must start with “c\_”. For example, adding the attribute color to the sales items the column name could be named as c\_color.

The following attributes are mandatory in the feed:

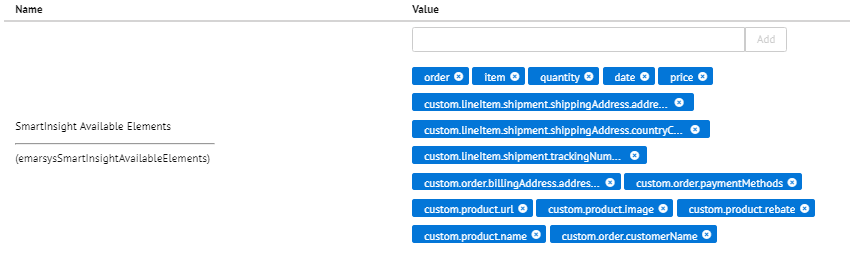
* + - order– order number
    - date – date of the order (YYYY-MM-DD)
    - customer – unique customer id (this ID must be available in Emarsys Suite as well)
    - item – ProductId of the sold item (this Id must be in the products CSV as well!)
    - quantity – quantity of the ordered item
    - c\_sales\_amount – Price of the sold items (this value must be negative if an order was cancelled!)

The following kind of attributes is possible:

* **Billing address:** The available element should start with 'billingAddress' and it should contain real attributes. In this way we get needed values from billingAddress object. Examples: billingAddress.address1, billingAddress.postalCode, billingAddress.countryCode.displayValue.
* **Shipping address:** The available element should start with 'shippingAddress' and it should contain real attributes. In this way we get needed values from order.shipments[0].shippingAddress object. Examples: shippingAddress.address1, shippingAddress.postalCode, shippingAddress.countryCode.displayValue, etc.
* **General order attributes:** The available element should start with 'order' and it should contain real attributes in this way we get needed values from order object.Examples: order.orderNo, order.creationDate, etc.
* **Delivery method.** Separate case for 'deliveryMethod.display' element only.

It reads shipping method name and description from order.shipments[0].shippingMethod object

* **Payment method:** Separate case for 'paymentMethod.display' element only. It reads 1st payment method from order object
* **Order rebate:** Separate case for 'orderRebate' element only.
* **Shipping costs:** Separate case for 'shippingCosts.display' element only, it reads shipping total price from order.shipments[0] object
* **customerNo:** Separate case for customer No
* **masterid, variantid:** include those attributes to send the id of master and variation products for purchased product
* **Custom attributes:** you have 3 types of custom attributes available for output: custom.order, custom.product and custom.lineItem. Each type represents Order, Product and ProductLineItem/GiftCertificateLineItem object attributes respectively, please visit https://documentation.demandware.com/DOC2/index.jsp to see the full list of available attributes for these objects.



* **Specific custom attributes:** Tracking number, shipment company, date of arrival, tracking link, should have the following element definition: custom.shipmentTrackingNumber, custom.shippingCompany, custom.arrivalDate, custom.trackingLink. The available element should be started with 'custom' it reads custom attributes values from order.shipments[0] object.

### Emarsys Catalog and SmartInsight Site Preferences

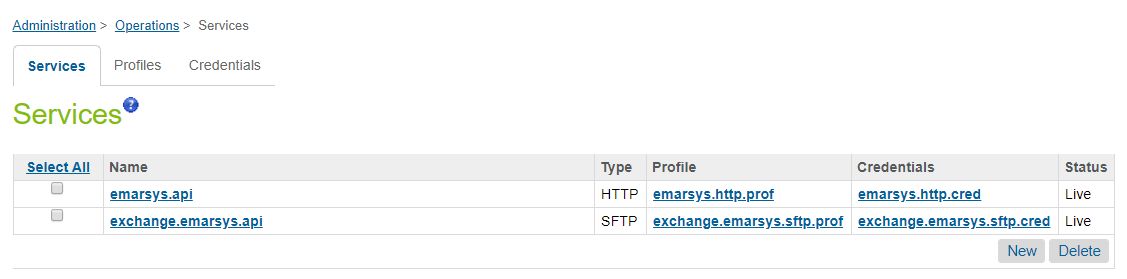
The table below contains the list of Site Preferences for Emarsys Catalog and SmartInsight.

|  |  |  |
| --- | --- | --- |
| Custom attribute | Type | Description |
| Predict: Merchant ID | String | Used to enable JavaScript tracking code, recommendations, etc. |
| Predict: Enable tracking code | Boolean | Enables/disables the JavaScript tracking code |
| Thank You For Your Order Page Recommendation Type | Enum Of Strings | Select recommendation widget type to show on storefront "thank you for your order" page |
| Product Detail Page Recommendation Type | Enum Of Strings | Select recommendation widget type to show on storefront PDP |
| Home Page Recommendation Type | Enum Of Strings | Select recommendation widget type to show on storefront Home page |
| Cart Page Recommendation Type | Enum Of Strings | Select recommendation widget type to show on storefront cart page |
| Search Page Recommendation Type | Enum of Strings | Select recommendation widget type to show on storefront search page |
| Category Page Recommendation Type | Enum of Strings | Select recommendation widget type to show on storefront category page |
| List of product variation attributes for predict Feed | Set of String | List of product variation attributes for Predict Feed |
| Predict: Product attributes for exporting | Set Of String | Mandatory attributes: item, available, title, link, image, category (path), price for catalog export |
| Export Folder | String | Emarsys Predict export folder relative to IMPEX/src/ |
| Map of locales and currencies | String | Map of configured locales and their currency codes  { "en\_US":"USD", "it\_IT":"EUR", "en\_GB":"GBP"} |
| SmartInsight Available Elements | Set Of String | SmartInsight Available Elements |

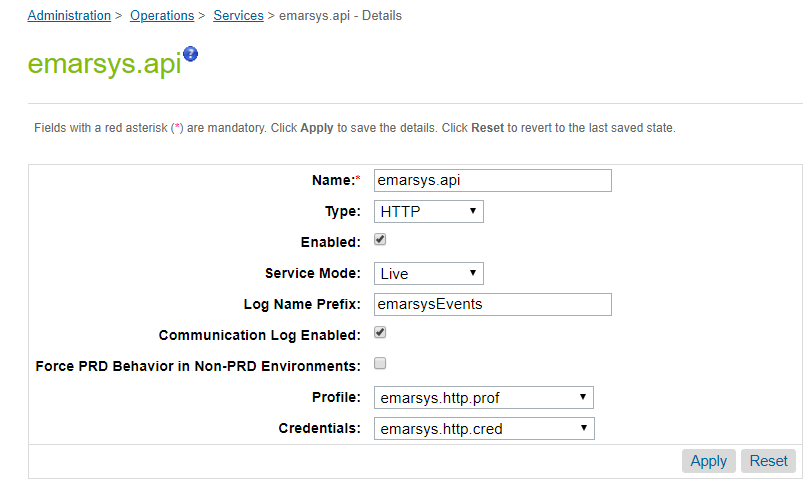
### Emarsys Services

Services are used to interact with API Emarsys.

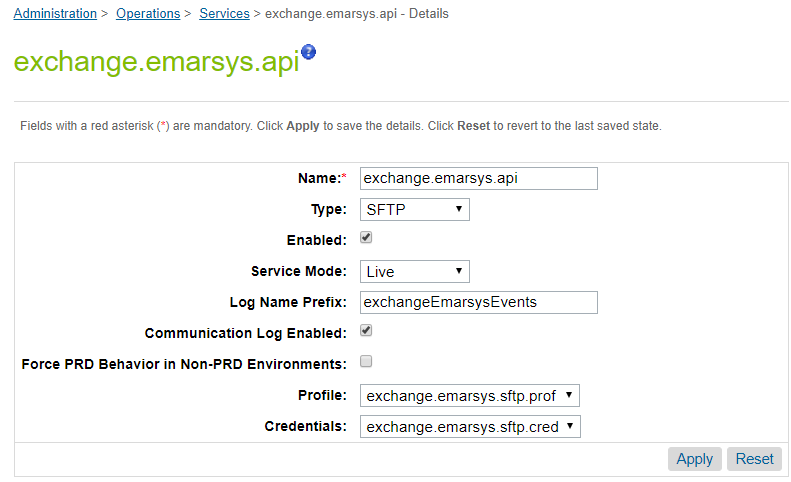
* Navigate to **Administration >  Operations >  Services**



* emarsys.api - service communicates with emarsys events.



* exchange.emarsys.api - used to load the directory on emarsys



## Job Schedules

### Emarsys-Setup

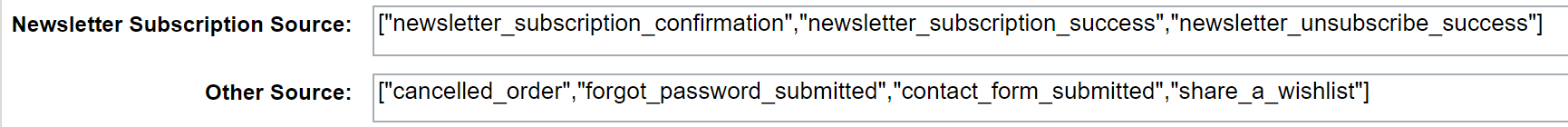
Once the cartridges are installed and the credentials have been configured inside Business Manager, a job schedule needs to be run in order to get from the Emarsys platform all the data that is necessary for the integration to work:

* Emarsys profile fields (profile fields are stored in a custom object on Salesforce Commerce Cloud side: *EmarsysProfileFields*)
* External events configuration (whitelisted SFCC side events and their mapping with Emarsys side events: *EmarsysExternalEvents*)
* The source id defined on Emarsys (will be saved on Salesforce Commerce Cloud side in a custom preference attribute: *Merchant Tools > Site Preferences > Custom Site Preferences > Emarsys > Emarsys Source ID*)
* All available values for Emarsys single choice fields (will be saved on Salesforce Commerce Cloud side in a custom preference attribute: *Merchant Tools > Site Preferences > Custom Site Preferences > Emarsys DB Init Configuration > Emarsys Single Choice Value Mapping*)

All this data will be automatically stored on Salesforce Commerce Cloud when the job is successfully executed.

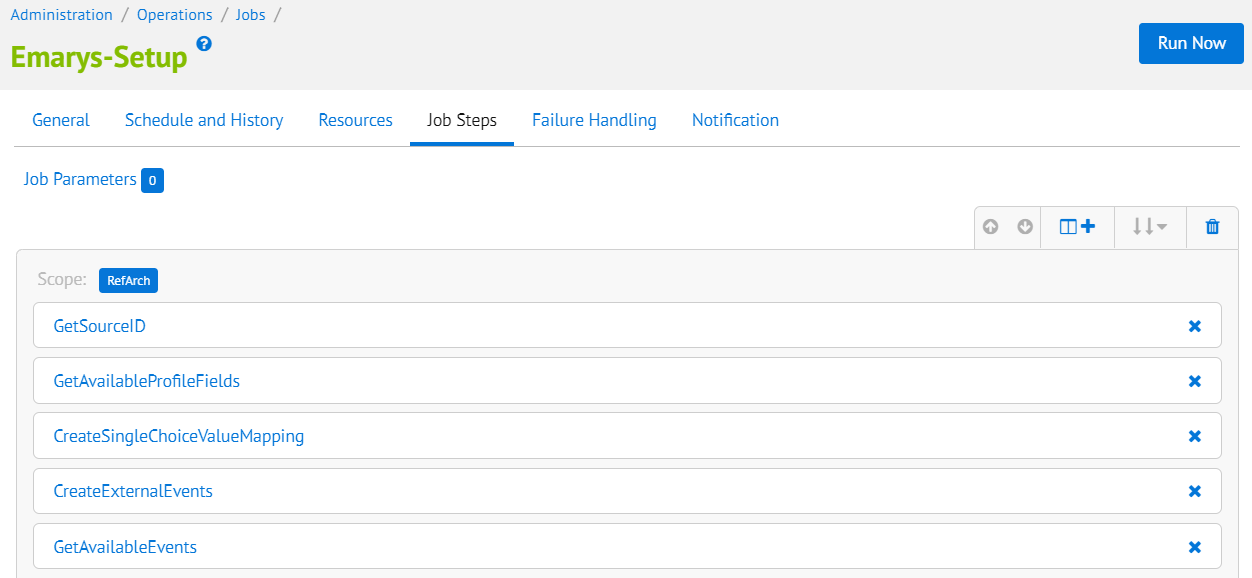
Before running the job, you should check some Business Manager configuration.

1. Login in **Business Manager.**
2. Select your site in the left top corner of the page
3. Navigate to **Merchant Tools > Custom Objects > Custom Object Editor**
4. Select object type **EmarsysExternalEvent** and click **Find** button
5. Click **StoredEvents** link in the column with **(ExternalEvents)** title
6. Fields **Newsletter Subscription Source** and **Other Source** should not be empty. These fields should contain lists of SFCC events, for which appropriate Emarsys external events should be created



After you checked the configuration you are all set to run the job

1. Login in **Business Manager.**
2. Go to **Administration > Operations** **> Job schedules**
3. Select **Emarsys-Setup** job and run it



This job includes 5 steps:

* **GetSourceID** - reads contact source name from **emarsysSourceName** site custom preference. Then tries to get it’s ID from Emarsys. If there are no contact sources with this name, the job step creates it. The main purpose of the step is to store ID of the contact source in **emarsysSourceID** site custom preference
* **GetAvailableProfileFields** - gets Emarsys system and custom fields descriptions and stores them to **EmarsysProfileFields** custom object. The language of the fields is taken from **emarsysGetProfileFieldsLanguage** site custom preference
* **CreateSingleChoiceValueMapping** - collects options descriptions for Emarsys fields of singlechoice type. These fields hold only the index of the selected option but not it’s value. The job step stores options descriptions for such fields in **emarsysSingleChoiceValueMapping** site custom preference
* **CreateExternalEvents** - reads SFCC events names from **newsletterSubscriptionSource** and **otherSource** fields of **EmarsysExternalEvents** custom object. For every SFCC event creates an appropriate event on Emarsys side. Store initial mapping between SFCC events and Emarsys events. Mapping objects are written in **newsletterSubscriptionResult** or **otherResult** field depending on the type of the event.
* **GetAvailableEvents** - gets all Emarsys external events descriptions and stores them in **result** field of **EmarsysExternalEvents** custom object

**Steps Configuration Description**

1. **custom.EmarsysComponents.CreateExternalEvents**
   1. **customObjectKey** – specifies key of **EmarsysExternalEvents** custom object. In this object stored external events configuration (default value: **StoredEvents**)
2. **custom.EmarsysComponents.GetAvailableEvents**
   1. **isDisabled** – provides convenient way to deactivate this job step

### Emarsys-ExportOrders

**EmarsysExportOrders** will export order data into csv file through new Job Schedule. Every field in the CSV file will be mapped to a corresponding field defined in the Emarsys documentation.

File structure:

* Name: **sales\_items\_<YYYYMMDDHHiiss>\_<shopinfo>.csv**
* Encoding: UTF-8
* Columns separator: “,”

At least these columns are required in the CSV:

1) order – order number

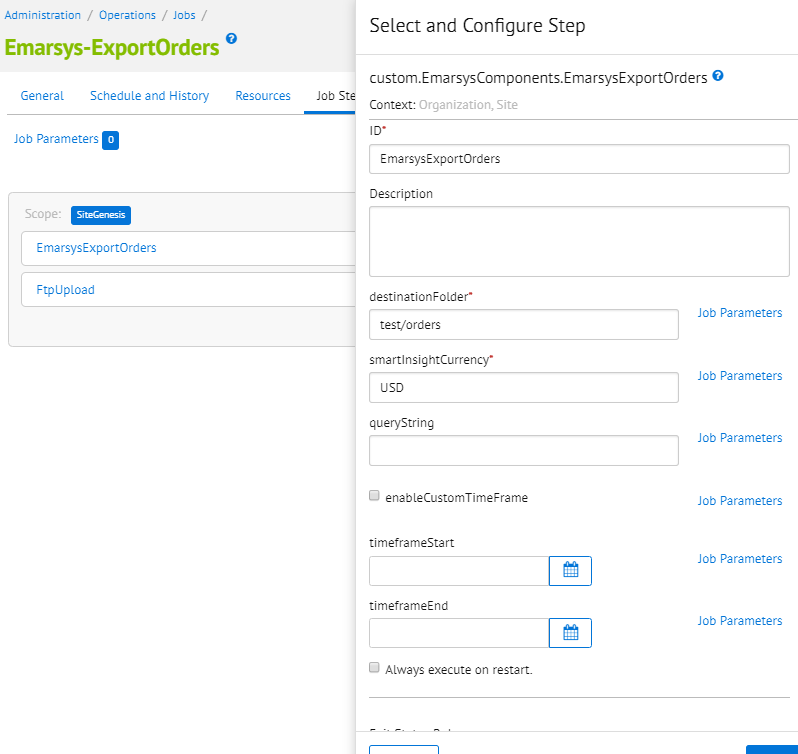
2) date – date of the order (YYYY-MM-DD)

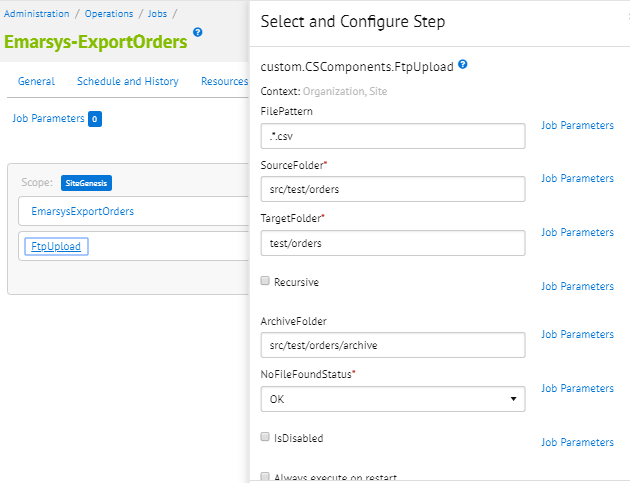
3) customer – unique customer id (this ID must be available in Emarsys Suite as well)

4) item – ProductId of the sold item (this Id must be in the products CSV as well!)

5) quantity – quantity of the ordered item

* Login in **Business Manager.**
* Go to **Administration > Operations** **> Jobs Schedules.**
* Open **Emarsys-ExportOrders** job.
* Check if the job is scheduled daily.





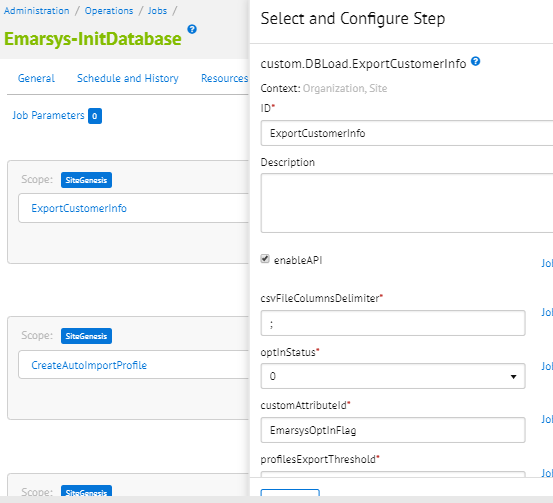
**Description Configure Steps**

1. **custom.EmarsysComponents.EmarsysExportOrders**
   1. **destinationFolder** – target folder path for save CSV file.
   2. **smartInsightCurrency** – mnemonic currency code of the currency that is configured for the.Smart.Insight.
   3. **queryString** – Add a query addendum for searchOrders. Start with AND.
   4. **enableCustomTimeFrame** – enable timeframe for exported orders.
   5. **timeframeStart** - timeframe start date for the Emarsys historical export job.
   6. **timeframeEnd** - timeframe end date for the Emarsys historical export job.
2. **custom.CSComponents.FtpUpload**
   1. **ServiceID** – service ID used in Job. Default value: exchange.emarsys.api.
   2. **FilePattern** – File pattern where used regular expression.
   3. **SourceFolder** – Local folder with files, relative to IMPEX/.
   4. **TargetFolder** - Remote folder on Server, relative to home directory.
   5. **ArchiveFolder** – Local folder where to archive files, relative to IMPEX/.
   6. **NoFileFoundStatus** – Exit code in case no files were found.
   7. **IsDisabled** – Mark the step as disabled. This will skip the step and returns a OK status.

### Emarsys-Init-Database

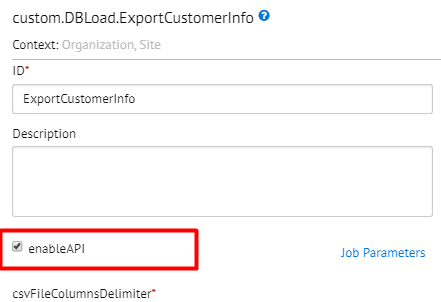
This job will initialize the Emarsys Database in 3 steps:

* **ExportCustomerInfo -** Used write profiles data into CSV files for initial DB load.
* **CreateAutoImportProfile -** Used to generate the auto-import profile on Emarsys side, based on the configuration saved through Initial DB load BM extension.
* **UploadCustomerInfoToWebDAV -** Used to upload the generated CSV file to Emarsys WebDAV.

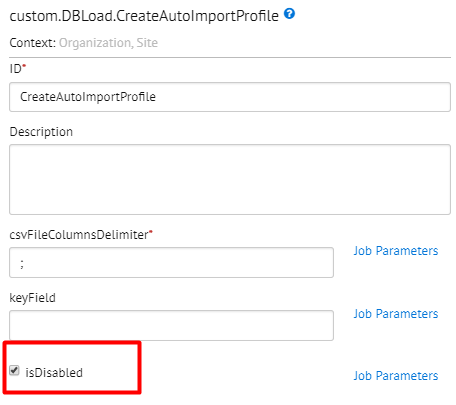
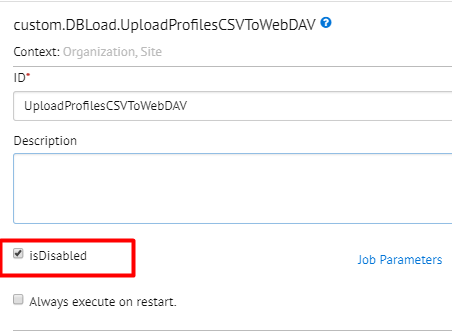


**Job behavior #1(using API call on Emarsys) -** When you need to send data directly on Emarsys.

1. Parameter **enabledAPI** must be enabled in *ExportCustomerInfo* step.

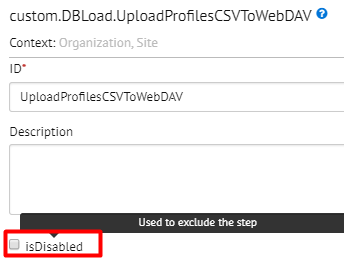
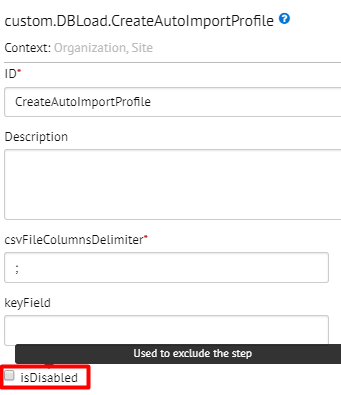
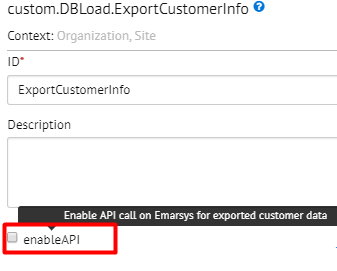


1. You must also enable parameter **isDisabled** in *CreateAutoImportProfile* step and *UploadProfilesCSVToWebDA*V step.



**Job behavior #2(using upload file to Emarsys WebDAV) -** When you need to upload the generated CSV file to Emarsys WebDAV.

1. Parameter **enabledAPI** must be exclude in *ExportCustomerInfo* step. You must also exclude parameter **isDisabled** in *CreateAutoImportProfile* step and *UploadProfilesCSVToWebDAV* step.



**Description Configure Steps**

1. **custom.DBLoad.ExportCustomerInfo**
   1. **enableAPI** – Enable API call on Emarsys for exported customer data
   2. **csvFileColumnsDelimiter** - character used to separate columns in the csv file.
   3. **optInStatus -** 0 - All users empty; 1 - All users true; 2 - Depending on attribute.
   4. **customAttributeId -** opt-In Status custom attribute ID.
   5. **profilesExportThreshold -** how many profiles should be exported at once?
   6. **queryString** – Add a query addendum for searchOrders. Start with AND.
   7. **enableCustomTimeFrame** – Enable timeframe for exported orders.
   8. **timeframeStart** – Timeframe start date for the Emarsys historical export job.
   9. **timeframeEnd** – Timeframe end date for the Emarsys historical export job
   10. **fromEmail** – email address for forwarding notifications.
   11. **mailTo** – notifications destination e-mail address.
   12. **mailSubject** – mail subject.
2. **custom.DBLoad.CreateAutoImportProfile**
   1. **csvFileColumnsDelimiter** – character used to separate columns in the csv file.
   2. **keyField** – Key that will be used to identify the customer
   3. **isDisabled** – Used to exclude the step
3. **custom.DBLoad.UploadProfilesCSVToWebDAV**
   1. **isDisabled** – Used to exclude the step

### Emarsys-ExportCatalog

A new job schedule **Emarsys-ExportCatalog** will export catalog data into csv. Every field in the CSV file will be mapped to a corresponding field defined in the Emarsys documentation.

File structure:

* Name: **products\_<nameSite>.csv**
* Encoding: UTF-8
* Columns separator: “,”

At least these columns are required in the CSV:

1) Item – unique product id

2) available – Is the product available (and can be recommended)?: true or false

3) title\_multilang – Product title

4) link\_multilang – Deep link to the product

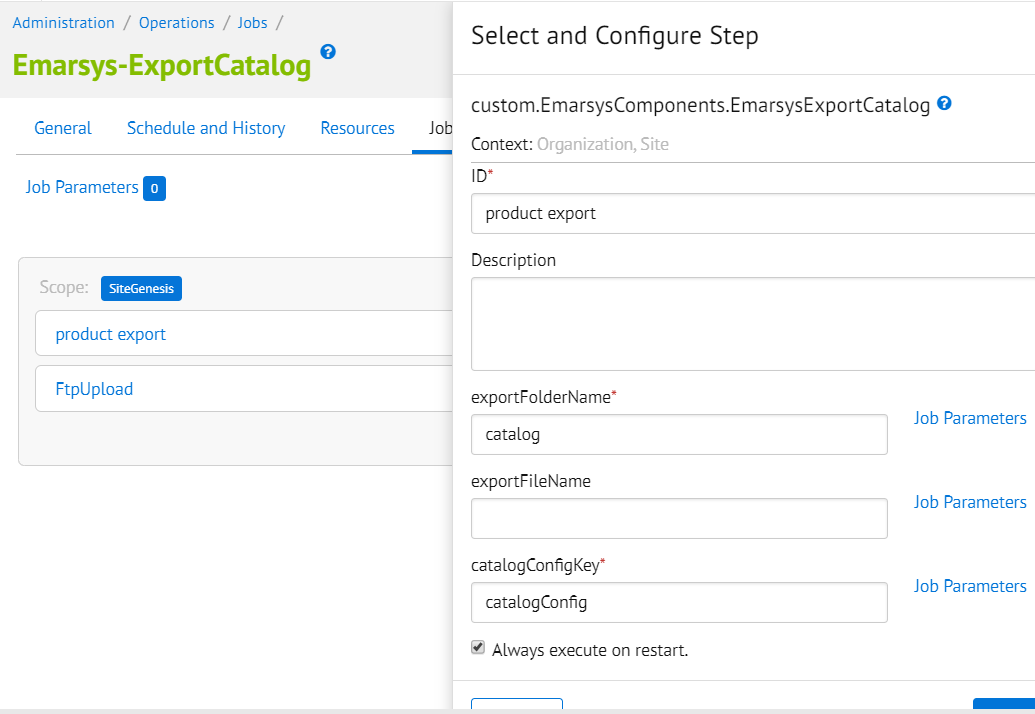
5) image – URL of the product image

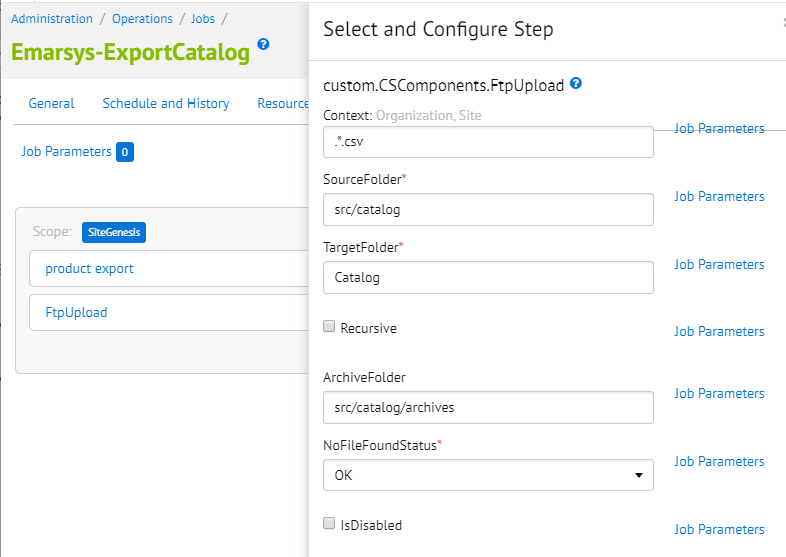
6) category\_multilang – Category path to the product separated by “ > “, e.g. books > scifi > startrek

7) price\_multilang – Product price (float value 1234.99)

8) group\_id – master product id

* Login in **Business Manager.**
* Go to **Administration > Operations** **> Job schedules**
* Check if **Emarsys-ExportCatalog** job is scheduled daily
* Run the job



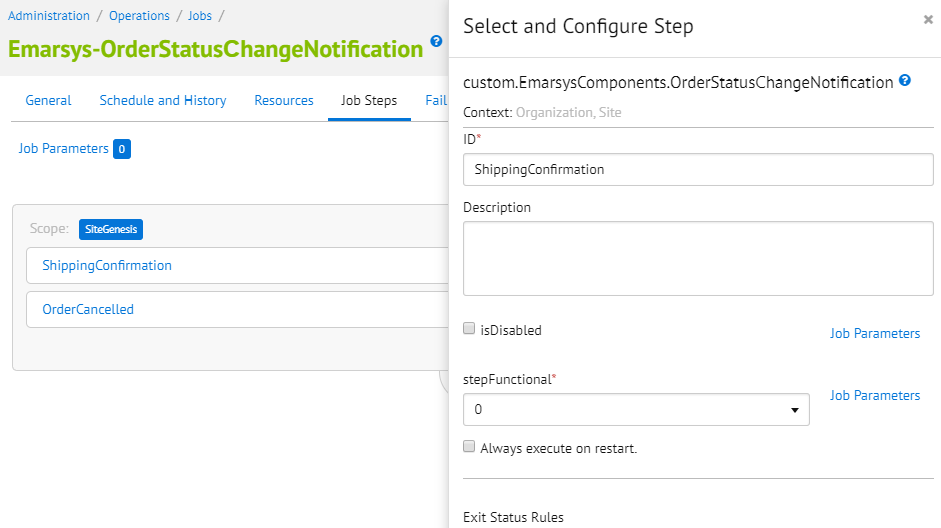


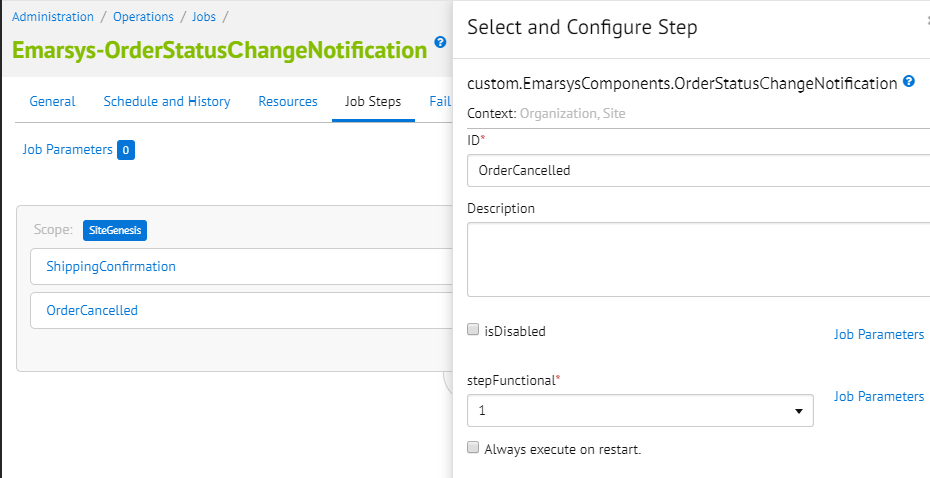
**Description Configure Steps**

1. **custom.EmarsysComponents.EmarsysExportCatalog**
   1. **exportFolderName –** Target folder path
   2. **exportFileName –** Export file name
   3. **catalogConfigKey** – Add the unique key value for custom object
2. **custom.CSComponents.FtpUpload**
   1. **ServiceID** – service ID used in Job. Default value: exchange.emarsys.api.
   2. **FilePattern** – File pattern where used regular expression.
   3. **SourceFolder** – Local folder with files, relative to IMPEX/.
   4. **TargetFolder** - Remote folder on Server, relative to home directory.
   5. **ArchiveFolder** – Local folder where to archive files, relative to IMPEX/.
   6. **NoFileFoundStatus** – Exit code in case no files were found.
   7. **IsDisabled** – Mark the step as disabled. This will skip the step and returns a OK status.

### Emarsys - OrderStatusСhangeNotification

This job initialized send email about confirmation or cancellation order through campaign on Emarsys





**Description Configure Steps**

1. **custom.EmarsysComponents.OrderStatusСhangeNotification**
   1. **isDisabled –** Mark the step as disabled. This will skip the step and returns a OK status.
   2. **stepFunctional** – 0 - send shipping data for Emarsys; 1 - send cancelled orders for Emarsys.

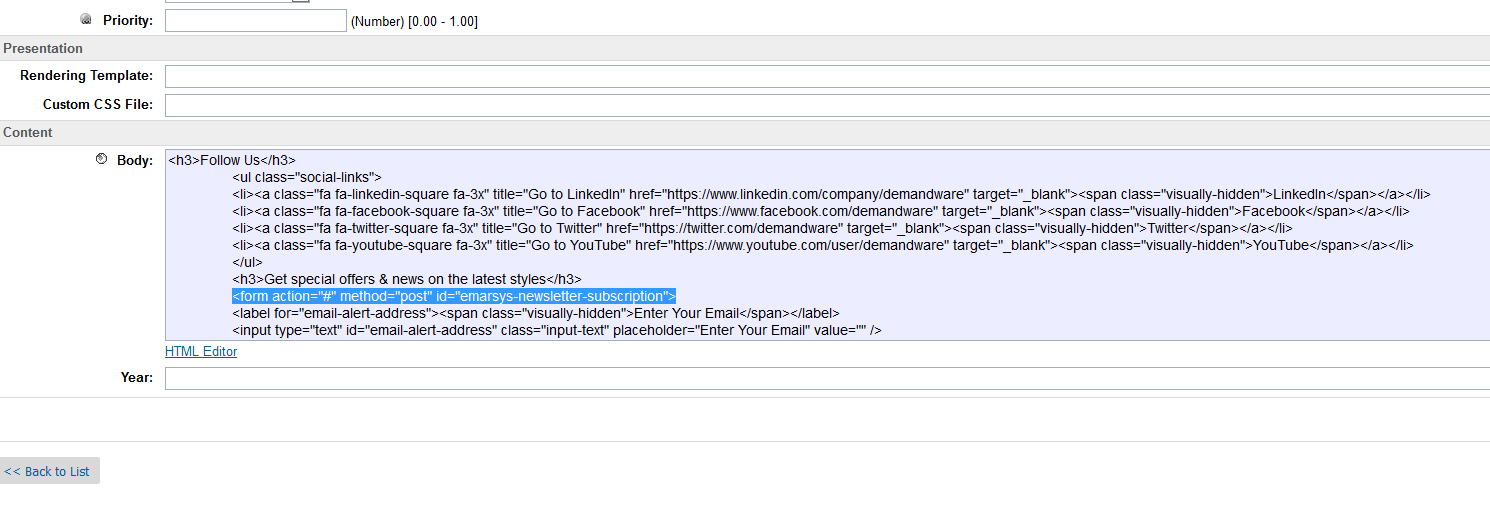
## Custom code

In order for the email subscriptions to work, a few updates need to be made on the storefront cartridge and inside Business Manager.

### Global footer subscription integration

For the global footer subscription to work, an update is required in the body of the *footer-social-email* content asset.

1. Navigate to **Business Manager > Merchant Tools > Content > Content Assets**
2. Search for the *footer-social-email* content asset
3. Lock the content asset in order to be able to edit it.
4. Go to the body attribute and update the id of the form to: *emarsys-newsletter-subscription*.



### Custom code inside core templates

1. In the storefront cartridge find **footer\_UI.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\components\footer\footer\_UI.isml

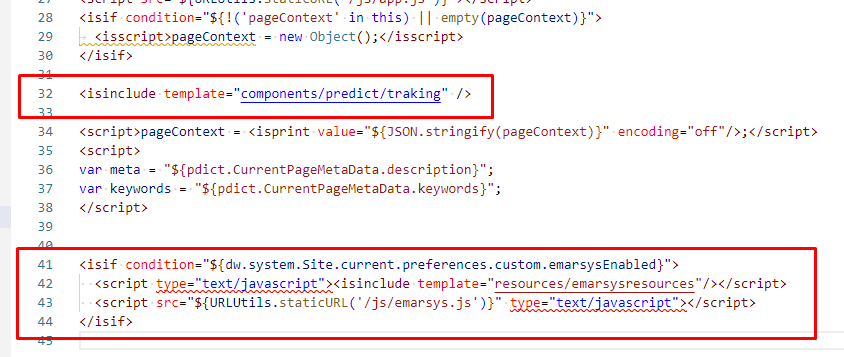
Add the following lines of code at the bottom of the ***footer\_UI.isml*** template:

<isif condition="${dw.system.Site.current.preferences.custom.emarsysEnabled}">

  <script type="text/javascript"><isinclude template="resources/emarsysresources"/></script>

  <script src="${URLUtils.staticURL('/js/emarsys.js')}" type="text/javascript"></script>

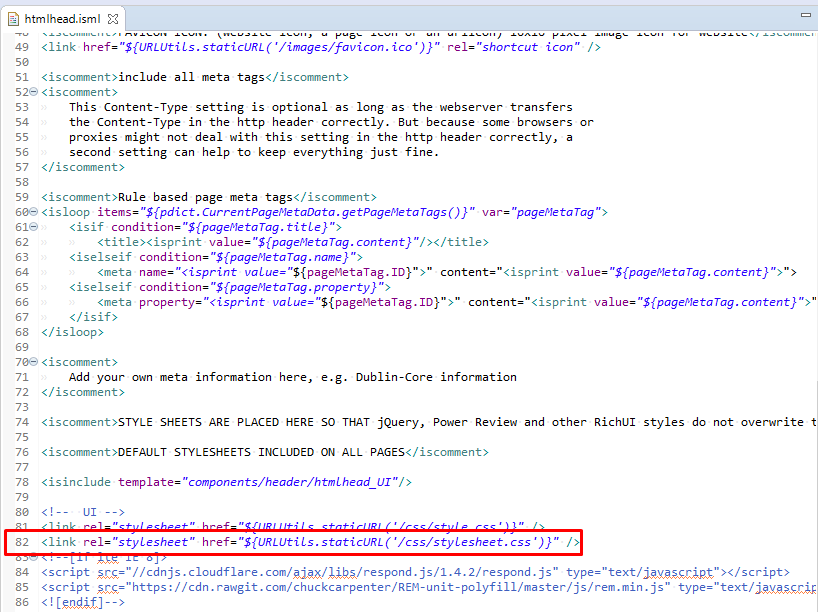
</isif>



Path: app\_storefront\_core\cartridge\templates\default\components\header\htmlhead.isml

Include the css file into the ***htmlhead.isml*** template :

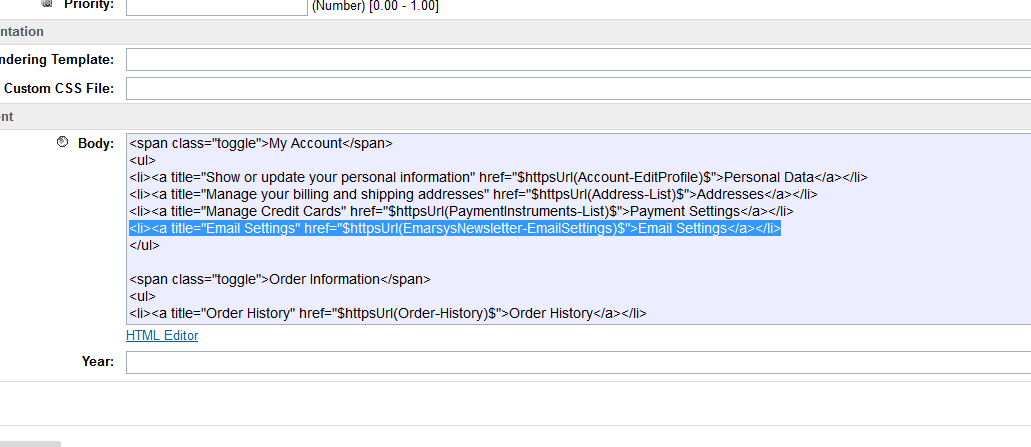
<link rel="stylesheet" href="${URLUtils.staticURL('/css/stylesheet.css')}" />



### Account subscription Business Manager configuration

1. Navigate to **Business Manager > Merchant Tools > Content > Content Assets**
2. Search for the *account-nav-registered* content asset.
3. Lock the content asset in order to be able to edit it.
4. Go to the body attribute and add the following line to the list of My Account section (please also check the screenshot below):

*<li><a title="Email Settings" href="$httpsUrl(EmarsysNewsletter-EmailSettings)$">Email Settings</a></li>*



### Controller integration configuration

#### Account subscription

For the subscription from the account profile to work, an update to the Account needs to be made. Also, there is a small change required in the *account-nav-registered* content asset.

* In your storefront controllers cartridge find *CustomerModel.js* and add in the createAccount function and replace the return with*:*

// Logs the customer in.

var result = Transaction.wrap(function () {

 return CustomerMgr.loginCustomer(email, password, rememberMe);

});

// Emarsys subscription

require('int\_emarsys\_controllers/cartridge/controllers/EmarsysNewsletter').AccountSubscriptionPipe();

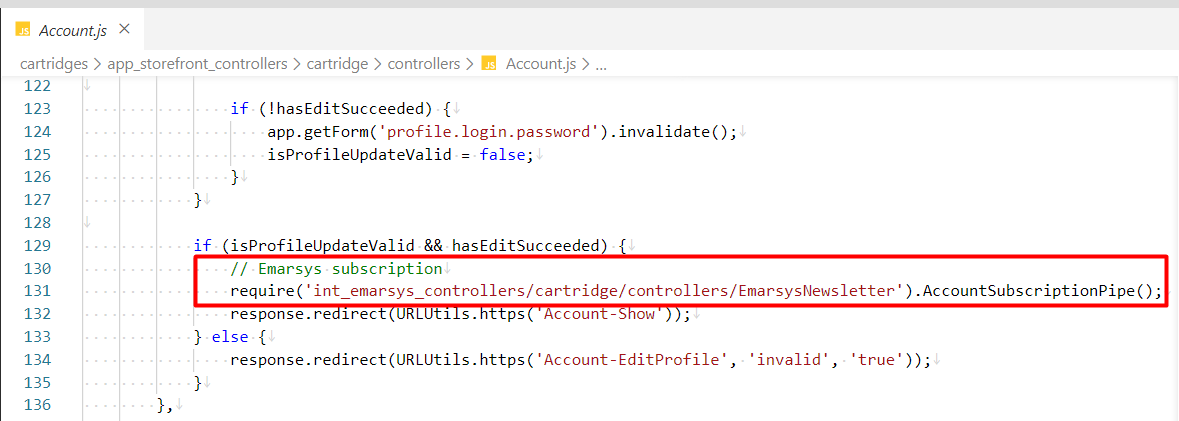
return result;



In the storefront controllers cartridge find *Account.js* controller and add in the confirm function after *line 129*:

// Emarsys subscription

require('int\_emarsys\_controllers/cartridge/controllers/EmarsysNewsletter').AccountSubscriptionPipe();

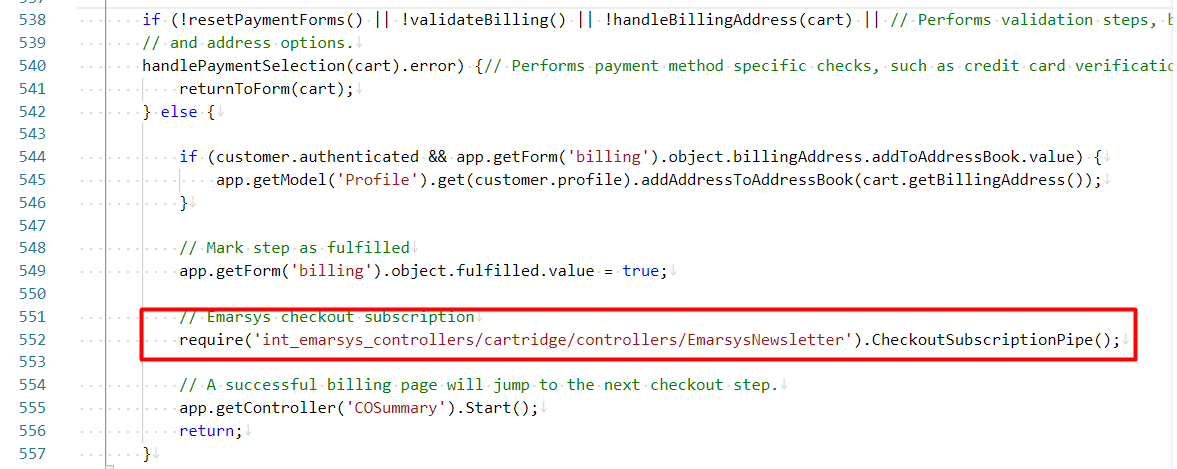


#### Checkout subscription

In order for the end user to be able to subscribe for newsletters during the checkout process, an update to the *COBilling.js*  controller file from the controllers cartridge needs to be made. At *line 549*, in the *save* function and after the step is marked as fulfilled, add a call to *EmarsysNewsletter-CheckoutSubscription* method:

// Emarsys checkout subscription

require('int\_emarsys\_controllers/cartridge/controllers/EmarsysNewsletter').CheckoutSubscriptionPipe();



#### Order confirmation emails

In the storefront controllers cartridge find *COPlaceOrder.js*  and in the *start()* function:

before payments handling add

    if(require('dw/system/HookMgr').hasHook('payment.beforeHandle')) {

        require('dw/system/HookMgr').callHook('payment.beforeHandle', 'beforeHandlePayment', {Order: order});

 }

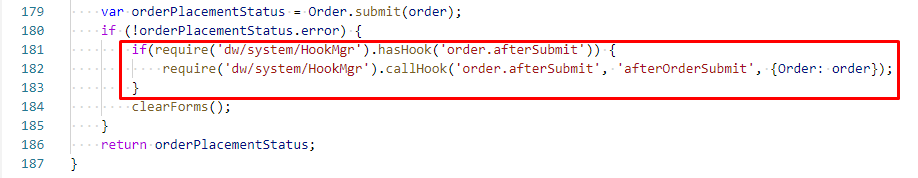


after the order is created and submitted, and execution in the successful execution or the order placement at *line 175*:

    if(require('dw/system/HookMgr').hasHook('order.afterSubmit')) {

            require('dw/system/HookMgr').callHook('order.afterSubmit', 'afterOrderSubmit', {Order: order});

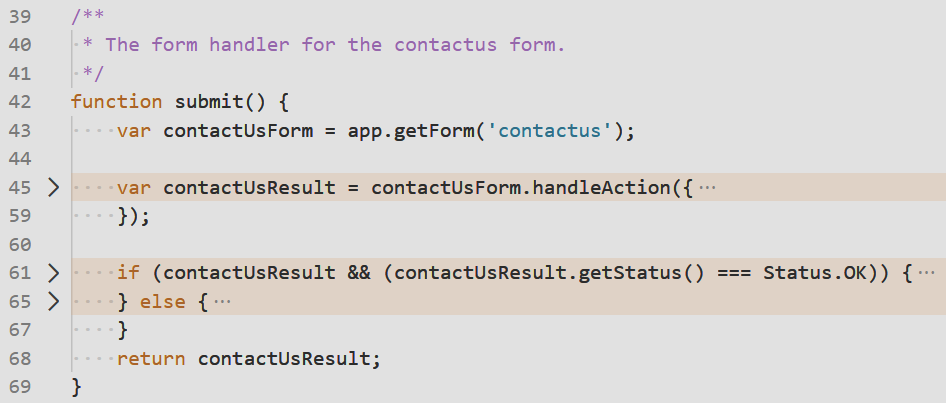
        }



#### External events triggering functionality

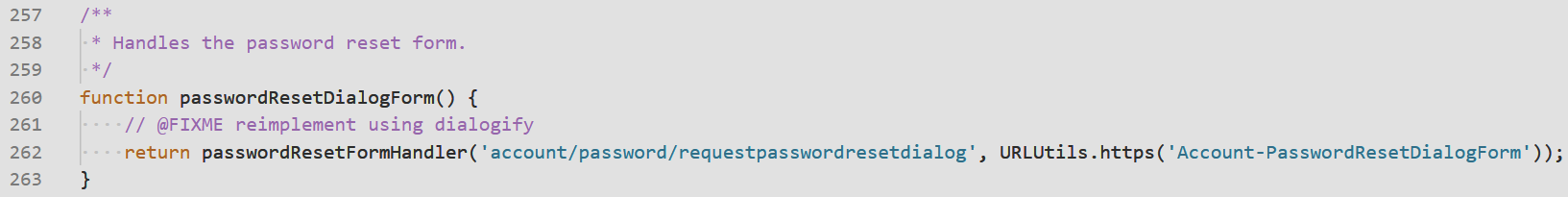
#### Contact Form Submitted

In the storefront controllers cartridge find the controller *CustomerService.js.* Function *submit()* should return value of the *contactUsResult* variable (str. 68):

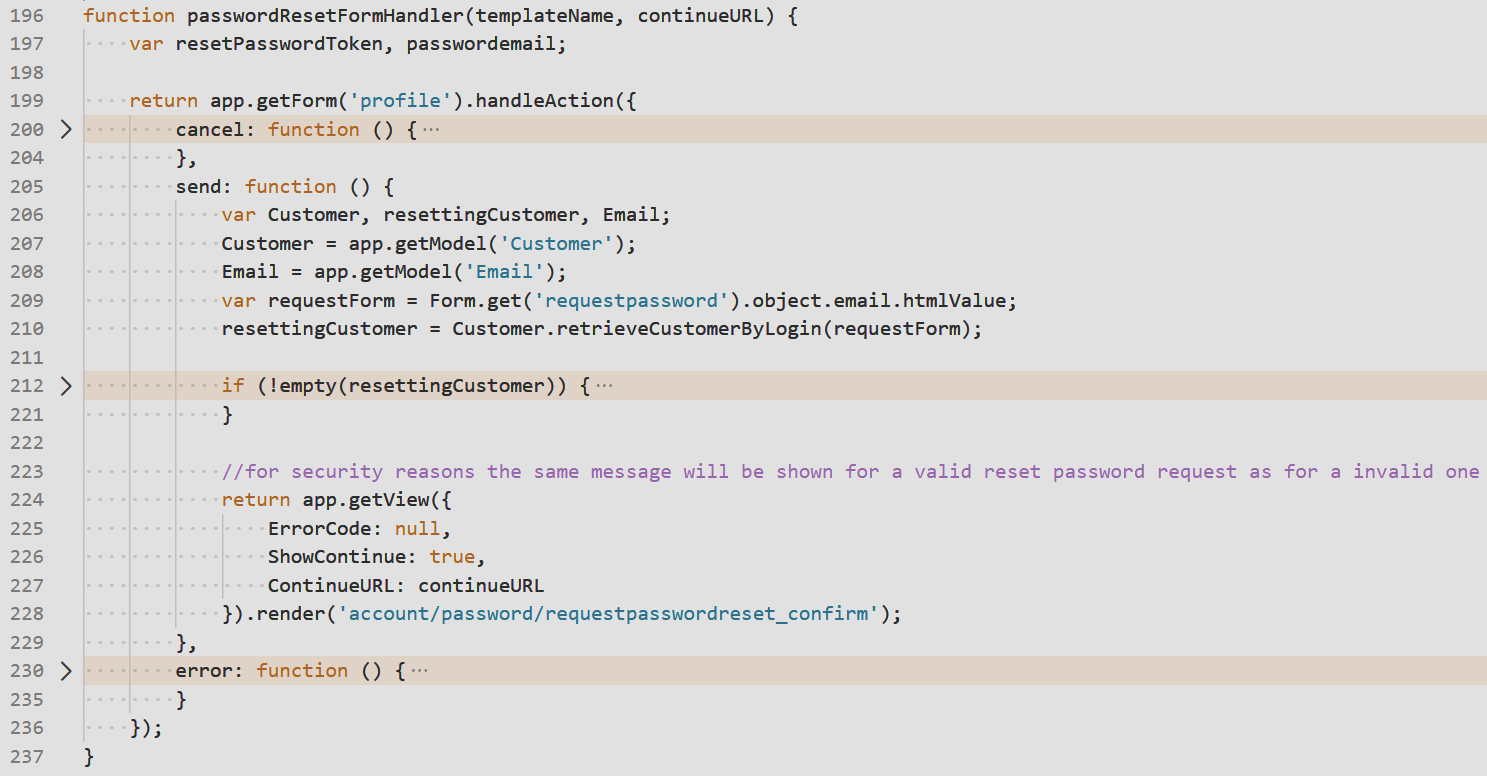


#### Forgot password submitted

In the storefront controllers cartridge find the controller *Account.js.* Add *return* statement in the *passwordResetDialogForm()* function (str. 262):



*passwordResetDialogForm()* function also need some additional *return* statements (str. 199 and str. 224):



### Shipping confirmation emails integration

It doesn’t require any additional modifications to the core cartridge.

### Initial database load integration

It doesn’t require any additional modifications to the core cartridge.

### Emarsys Predict

1. In the storefront cartridge find **pt\_productdetail.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\product\pt\_productdetails.isml

Include the following code update after line 8 :

    <isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/emarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        var pageContext = {

            title: (pdict.Product.name || 'Product Detail'),

            type: 'product',

            ns:'product',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

    </isscript>



1. In the storefront cartridge find **pt\_cart.isml** template.
   1. Path : app\_storefront\_core\cartridge\templates\default\checkout\cart\pt\_cart.isml
   2. Include the following code before the closing **isif** tag near line 891.

    <isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/emarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        var pageContext = {

            title: 'Cart',

            type: 'Cart',

            ns: 'cart',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

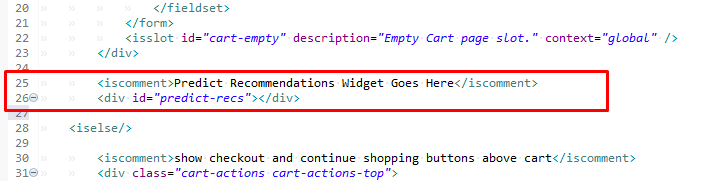
    </isscript>



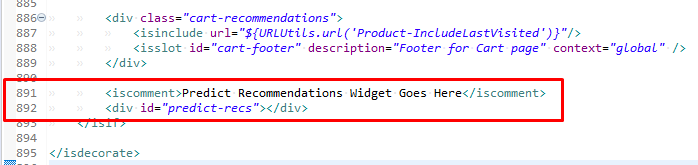
1. In the storefront cartridge find **cart.isml** template.
   1. Path : app\_storefront\_core\cartridge\templates\default\checkout\cart\cart.isml
   2. Include the following code before the **iselse** tag under <isif condition="${pdict.Basket == null || (empty(pdict.Basket.productLineItems) && empty(pdict.Basket.giftCertificateLineItems))}" condition near line 24 .

*<iscomment>Predict Recommendations Widget Goes Here</iscomment>*

*<div id="predict-recs"></div>*



After 890 line:



1. In the storefront cartridge find **confirmation.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\checkout\confirmation\confirmation.isml

Include the following code before the closing **isdecorate** tag near line 34.

*<iscomment>Predict Recommendations Widget Goes Here</iscomment>*

*<div id="predict-recs"></div>*



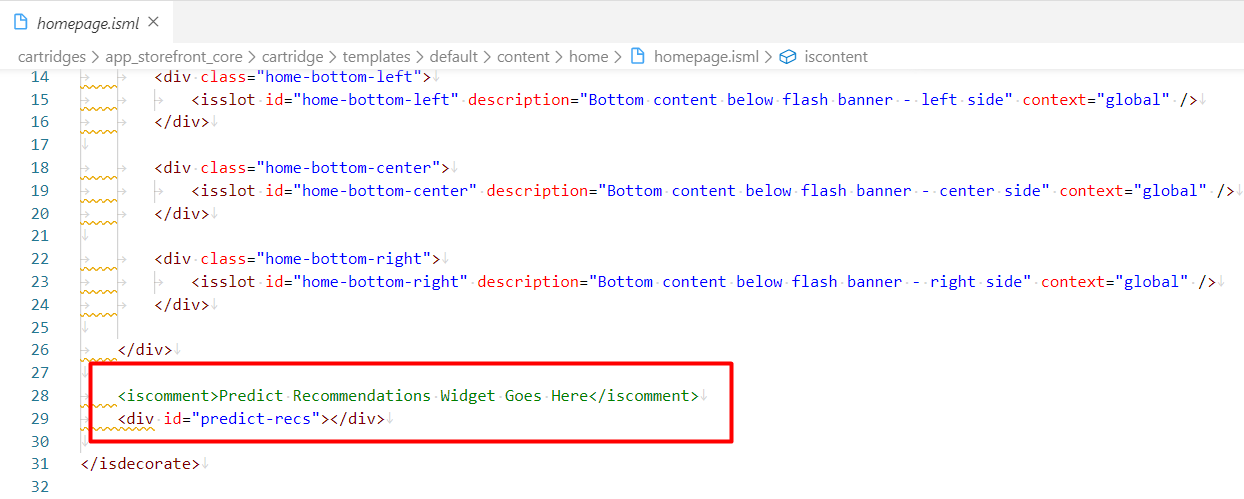
1. In the storefront cartridge find **homepage.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\content\home\homepage.isml

Include the following code before the closing **isdecorate** tag near line 29.

*<iscomment>Predict Recommendations Widget Goes Here</iscomment>*

*<div id="predict-recs"></div>*



1. In the storefront cartridge find **producttopcontent.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\product\producttopcontent.isml

Include the following code before the closing **isif** tag near line 92.

*<iscomment>Predict Recommendations Widget Goes Here</iscomment>*

*<div id="predict-recs"></div>*



1. In the storefront cartridge find **catlanding.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\rendering\category\catlanding.isml

Include the following code before the closing **decorate** tag near line 30.

*<iscomment>Predict Recommendations Widget Goes Here</iscomment>*

*<div id="predict-recs"></div>*



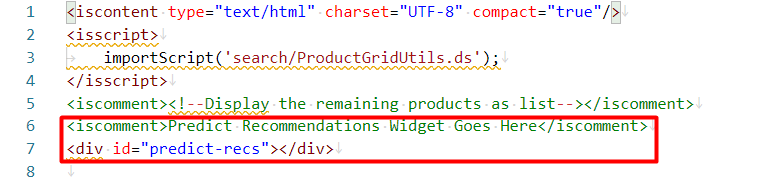
1. In the storefront cartridge find **productgrid.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\search\productgrid.isml

Include the following code before the **<ul id="search-result-items"** tag near line 9.

*<iscomment>Predict Recommendations Widget Goes Here</iscomment>*

*<div id="predict-recs"></div>*



1. In the storefront cartridge find **pt\_account.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\account\pt\_account.isml

Replace the following code after line 8.  
<isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/emarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        var pageContext = {

            title: 'My Account',

            type: 'MyAccount',

            ns: 'account',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

    </isscript>



1. In the storefront cartridge find **pt\_checkout.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\checkout\pt\_checkout.isml

Replace the following code after line 8.

    <isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/emarsysAnalyticsHelper '))()}" scope="page"/>

    <isscript>

        var pageContext = {

            title: 'Checkout',

            type: 'checkout',

            ns: 'checkout',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

    </isscript>



1. In the storefront cartridge find **pt\_orderconfirmation.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\checkout\pt\_orderconfirmation.isml

Replace the following code after line 8.

    <isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/emarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        var pageContext = {

            title: 'Order Confirmation',

            type: 'orderconfirmation',

            ns: 'orderconfirmation',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

    </isscript>



1. In the storefront cartridge find **pt\_storefront.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\content\home\pt\_storefront.isml

Replace the following code after line 8.

<isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/emarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        pdict.showCountrySelector = true;

        var pageContext = {

            title: 'Storefront',

            type: 'storefront',

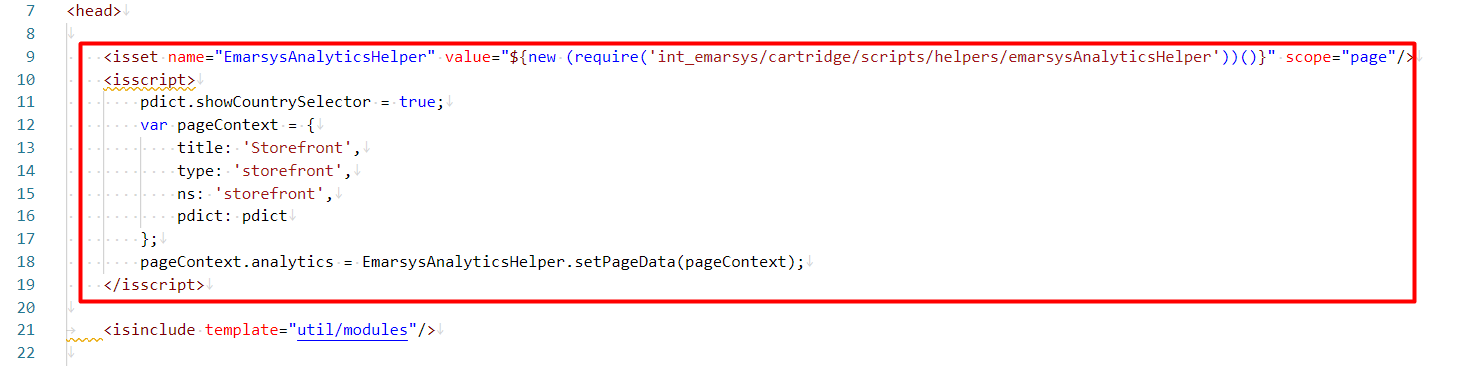
            ns: 'storefront',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

    </isscript>



1. In the storefront cartridge find **pt\_categorylanding.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\search\pt\_categorylanding.isml

Replace the following code after line 8.

    <isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/EmarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        var pageContext = {

            title: 'Product Search Results',

            type:'search',

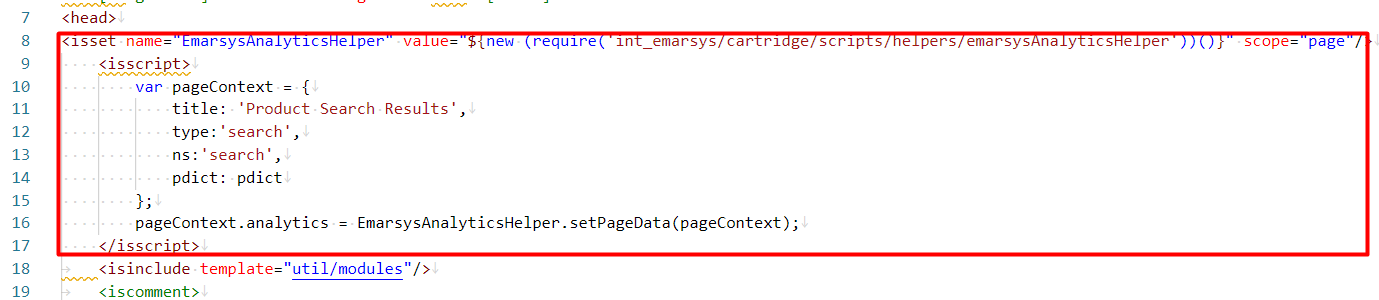
            ns:'search',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

  </isscript>



1. In the storefront cartridge find **pt\_contentsearchresult.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\search\pt\_contentsearchresult.isml

Replace the following code after line 8.

    <isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/EmarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        var pageContext={

            title: 'Content Search Results',

            type: 'search',

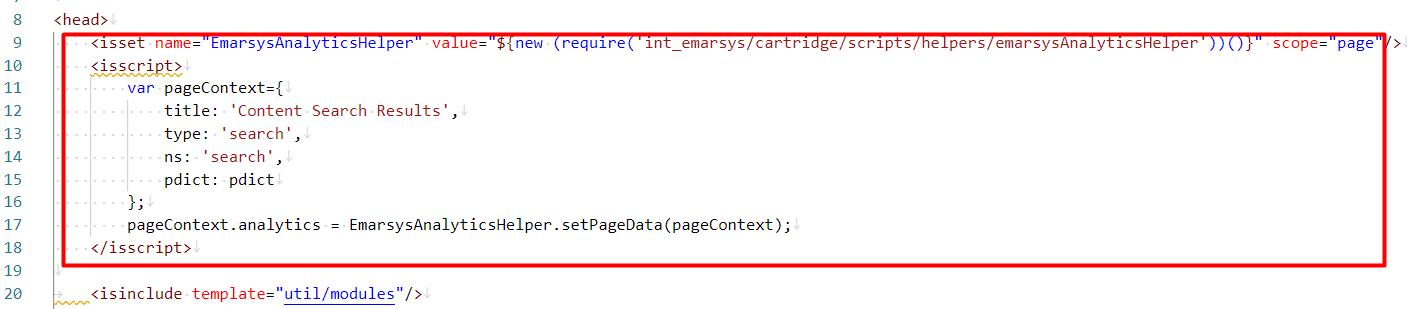
            ns: 'search',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

    </isscript>



1. In the storefront cartridge find **pt\_contentsearchresult\_nohits.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\search\pt\_productsearchresult\_nohits.isml

Replace the following code after line 8.

    <isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/EmarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        var pageContext = {

            title: 'Product Search Results No Hits',

            type:'search',

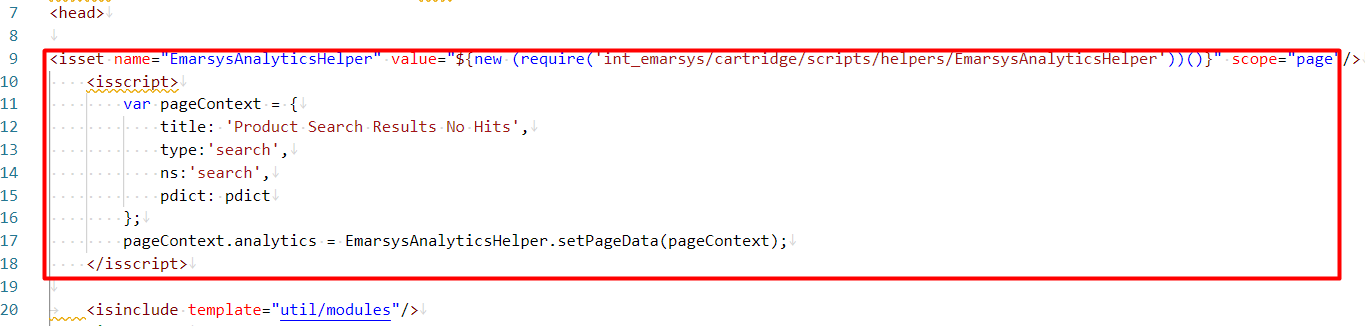
            ns:'search',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

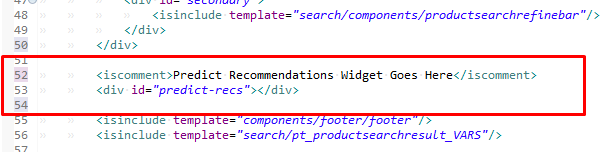
    </isscript>



Include the following code after line 52.

<iscomment>Predict Recommendations Widget Goes Here</iscomment>

        <div id="predict-recs"></div>



1. In the storefront cartridge find **pt\_productsearchresult.isml** template.

Path : app\_storefront\_core\cartridge\templates\default\search\pt\_productsearchresult.isml

Replace the following code after line 8.

    isset name="EmarsysAnalyticsHelper" value="${new (require('int\_emarsys/cartridge/scripts/helpers/emarsysAnalyticsHelper'))()}" scope="page"/>

    <isscript>

        var pageContext = {

            title: 'Product Search Results',

            type:'search',

            ns:'search',

            pdict: pdict

        };

        pageContext.analytics = EmarsysAnalyticsHelper.setPageData(pageContext);

    </isscript>



1. In order to handle cart data sending to Emarsys right after user clicks on the ‘Add to cart’ button next modifications are required: in the storefront cartridge find **addToCart.js** script. Add next code in the **addToCart** function  
   Path : app\_storefront\_core\cartridge\js\pages\product\addToCart.js

*$.ajax({*

*url: EmarsysUrls.emarsysAddToCartAjax*

*}).done(function (data) {*

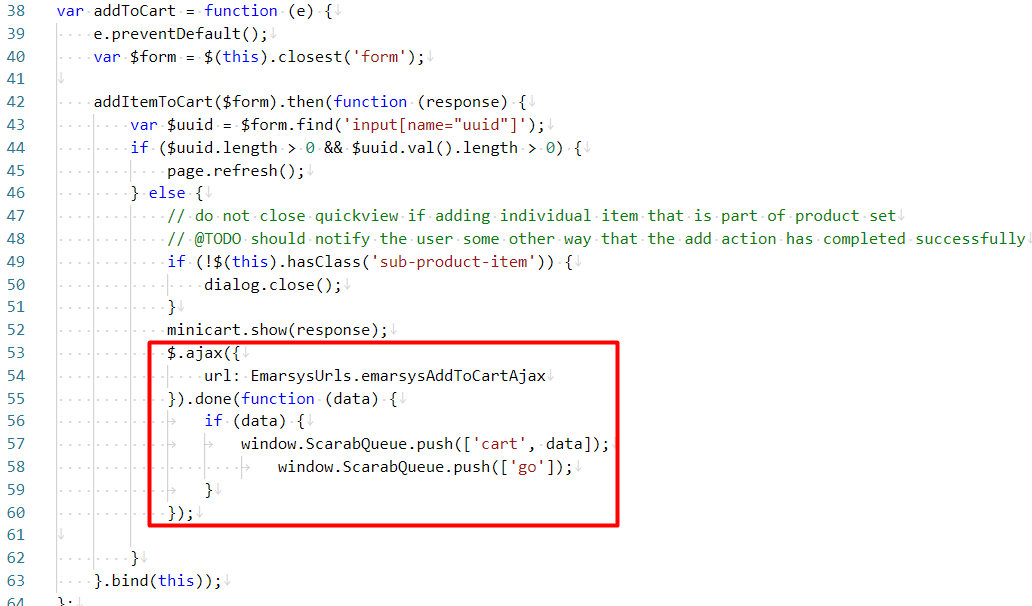
*if (data) {*

*window.ScarabQueue.push(['cart', data]);*

*window.ScarabQueue.push(['go']);*

*}*

*});*



### Emarsys Loyalty Wallet

To connect your Loyalty wallet should be placed in the footer

(path: Path : app\_storefront\_core\cartridge\templates\default\components\footer\footer\_UI.isml) after 38 line:

<isinclude template="/loyaltyWallet"/>



Also, to be customized Emarsys Loyalty Wallet:

1. Navigate to **Business Manager > Merchant Tools > Site Preferences > Custom Preferences.**
2. Search for the *EmarsysLoyaltyWallet* Custom Site Preference Groups
3. *Emarsys Loyalty Wallet Enabled* must be enabled.
4. Other fields must also be filled. Please see the section **3.4.4 Site preferences configuration** for the detailed description

## Testing

For a complete list of test cases, please refer the test cases document.

# Operations, maintenance

## Data storage

The integration uses several custom objects to store the saved configurations for newsletter subscriptions, external events defined on the Emarsys side, transactional emails configuration, initial DB load and Emarsys profile fields. Please find below details about each of these custom objects:

* **EmarsysExternalEvents** – used to store Emarsys external events configuration. Source fields store whitelisted SFCC events while result fields store mapping between SFCC side events and Emarsys side events. Job Emarsys-Setup creates all necessary events on Emarsys side and sets initial events mapping but needs source fields to be filled.
* **EmarsysNewsletterSubscription** – used to store the configuration for each newsletter subscription type (global footer subscription, checkout subscription and my account subscription). Instances of this custom object are created on the fly, when the merchant makes the configurations for newsletter subscriptions using the Emarsys Business Manager extension.
* **EmarsysTransactionalEmailsConfig** – used to store the configurations for Emarsys transactional emails. There will be two instances of this custom object, which will be created automatically when the merchant saves the configurations for order and shipping confirmation emails through Business Manager extension.
* **EmarsysDBLoadConfig** – used to store the mapping between Demandware and Emarsys fields for the initial DB load. There will be only one instance of the custom object, which will be automatically created when the merchant saves the initial DB load mapping through Business Manager extension.
* **EmarsysProfileFields** – used to store the Emarsys profile fields. It’s automatically populated after the Emarsys-Setup job schedule has finished running.

There are also several custom site preferences attributes used to define the Demandware fields that appear in the Emarsys Business Manager extension – these custom attributes are used to define the values that will appear in the dropdowns from the configuration screens. Please find below details about each of these attributes:

* **emarsysOrderConfirmationElements** – JSON used to define the Demandware available fields that can be added to the order confirmation emails.
* **emarsysShippingInformationElements** – JSON used to define the Demandware available fields that can be added to the shipping confirmation emails.
* **emarsysDBLoadAttributes** – JSON used to define the Demandware available fields that can be added to the CSV file that is generated for the initial DB load.
* **emarsysSingleChoiceValueMapping** – JSON used to store all Emarsys single choice fields and the available values for each field. This attribute is used when creating the Emarsys auto-import profile for the initial DB load, and it is automatically populated after the Emarsys-Setup job schedule has finished running.
* **emarsysContactFieldsMap** – JSON used to define the mapping between several Demandware fields and the corresponding IDs for these fields on Emarsys side.

**Emarsys Catalog** uses **EmarsysCatalogConfig** custom object type to store the list of field mapping for catalog configuration. This custom object is updated automatically when configs are set up in Catalog BM extension (see User Guide section).

**Example of generated mapped fields**

[{"field":"product.ID","placeholder":"item"},{"field":"product.availability","placeholder":"available"},{"field":"product.name","placeholder":"title"},{"field":"product.url","placeholder":"link"},{"field":"product.image","placeholder":"image"},{"field":"product.categories","placeholder":"category"},{"field":"product.price","placeholder":"price"}]

**Emarsys SmartInsight** uses **EmarsysSmartInsightConfiguration** custom object type to store the list of field mapping for smartinsight configuration. This custom object is updated automatically when configs are set up in SmartInsight BM extension (see User Guide section).

**Example of generated mapped fields**

[{"field":"order.orderNo","placeholder":"order"},{"field":"product.ID","placeholder":"item"},{"field":"product.quantity","placeholder":"quantity"},{"field":"order.customerEmail","placeholder":"c\_email"},{"field":"order.totalGrossPrice","placeholder":"c\_sales\_amount"},{"field":"order.creationDate","placeholder":"date"}]

## Availability

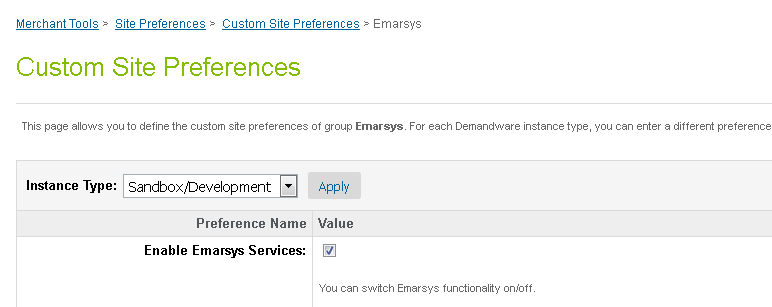
The Emarsys platform is expected to be available 24/7. However, for the scenario when the platform is not available a custom error message is logged in Demandware.

## Support

Please contact a OSF representative if you need support for this integration.

# User guide

The Emarsys integration can be enabled/disabled from Business Manager using the toggle on/off feature. In order to turn on/off the integration, navigate to **Merchant Tools > Site Preferences > Custom Preferences > Emarsys > Enable Emarsys Services.**

****

When the integration is disabled and the end user tries to subscribe for newsletter using the global footer or my account subscription method, he will be redirected to a custom page which will show the message that says the subscriptions are not active.

Note: even if the integration is disabled, as long as the cartridge is still installed, the end user has the option to unsubscribe from newsletters, from his Demandware account (only registered users) or by clicking the un-subscription link present in the emails received through Emarsys platform.

## Business manager

A new Business Manager extension has been created in order to allow the merchant to configure the behavior for newsletter subscriptions and to define the Demandware fields that will be added to the transactional emails (order and shipping confirmation emails) that will be send through the Emarsys platform. The extension will also allow the merchant to map Demandware with Emarsys fields for the initial DB load feature.

### Newsletter subscription box – global footer

Follow the next steps to configure the Newsletter subscription into Business Manager:

1. Login in **Business Manager**.
2. Go to **Merchant Tools > Emarsys Integration**.
3. Select the **Newsletter subscriptions** item.



1. The page opens with the **Global footer** tab selected by default.
2. The merchant needs to select the **Opt-in** strategy. There are two options available:

* *Single Opt-In* (recipient gets opt-in system field set to true immediately in the Emarsys Suite). Optionally the merchant can select an external event that gets fired after the user record was transmitted to Emarsys.
* *Double Opt-In* (recipient needs to click a confirmation link in an email in order to get the system field opt-in in Emarsys Suite set to true). If double opt-in is selected, then the merchant must select the external event that should be triggered in Emarsys Suite to send out the confirmation email. Optionally the merchant can set a second event that will be triggered as soon an end user clicks the confirmation link (e.g. to send a welcome email).

1. Hit the **Apply** button to save the configuration.

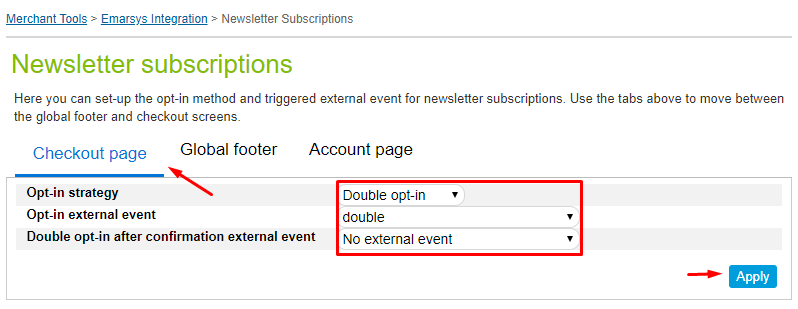


### Newsletter subscription – checkout process

1. Login in **Business Manager**.
2. Go to **Merchant Tools > Emarsys Integration**.
3. Select the **Newsletter subscriptions** item.
4. Click the **Checkout page** tab.
5. The merchant needs to select the **Opt-in** strategy. There are two options available:

* *Single Opt-In* (recipient gets opt-in system field set to true immediately in the Emarsys Suite). Optionally the merchant can select an external event that gets fired after the user record was transmitted to Emarsys.
* *Double Opt-In* (recipient needs to click a confirmation link in an email in order to get the system field opt-in in Emarsys Suite set to true). If double opt-in is selected, then the merchant must select the external event that should be triggered in Emarsys Suite to send out the confirmation email. Optionally the merchant can set a second event that will be triggered as soon an end user clicks the confirmation link (e.g. to send a welcome email).

1. Hit the **Apply** button to save the configuration.

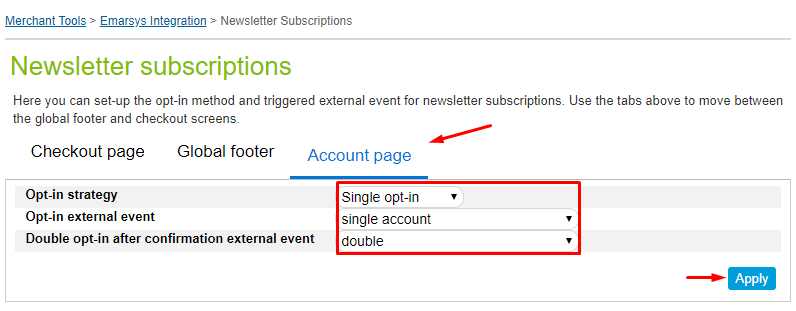


### Newsletter subscription – My account page

1. Login in **Business Manager**.
2. Go to **Merchant Tools > Emarsys Integration**.
3. Select the **Newsletter subscriptions** item.
4. Click the **Account page** tab.
5. The merchant needs to select the **Opt-in** strategy. There are two options available:

* *Single Opt-In* (recipient gets opt-in system field set to true immediately in the Emarsys Suite). Optionally the merchant can select an external event that gets fired after the user record was transmitted to Emarsys.
* *Double Opt-In* (recipient needs to click a confirmation link in an email in order to get the system field opt-in in Emarsys Suite set to true). If double opt-in is selected, then the merchant must select the external event that should be triggered in Emarsys Suite to send out the confirmation email. Optionally the merchant can set a second event that will be triggered as soon an end user clicks the confirmation link (e.g. to send a welcome email).

1. Hit the **Apply** button to save the configuration.



### Transactional emails – order confirmation

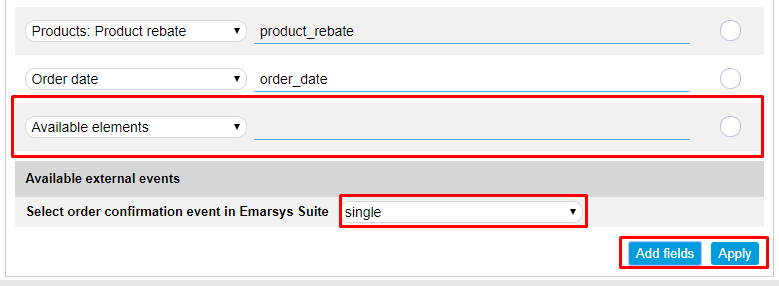
As soon as an end user finishes a transaction, the order data should be transmitted to Emarsys Suite and the corresponding external event must be triggered. The external event will trigger an order confirmation email. Demandware will check in the Emarsys Suite if the user record already exists and based on the result it will update the record or create a new one. Also, the merchant has the option to choose what fields will be included in the order confirmation email.

To configure and map the Demandware fields, that will be added to the order confirmation email, with the Emarsys defined placeholders, a custom configuration screen has been created. Please find below the steps you need to follow to make the necessary configurations for order confirmation emails:

1. Navigate to **Business Manager > Merchant Tools > Emarsys Integration > Order Confirmation configuration**



1. Click the **Add fields** button.
2. From the dropdown, select the Demandware available field and then define the **placeholder** name in the right column.
3. Select the external event that will be triggered after the order is placed and the record is created/updated in the Emarsys Suite.



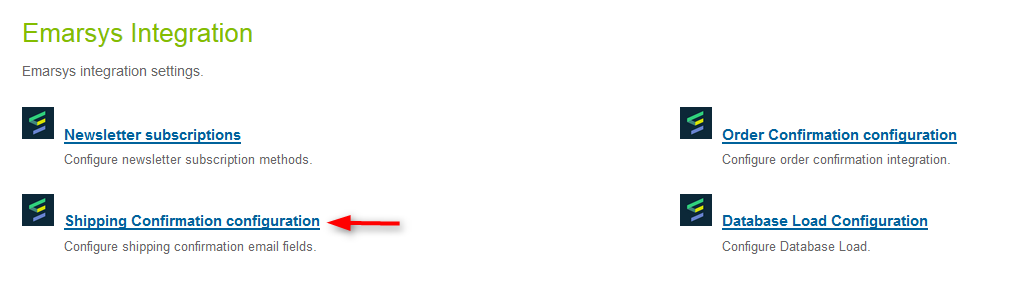
1. To remove some of the defined Demandware field/Emarsys placeholder name pairs, select the corresponding *Remove* checkbox and hit the **Apply** button.

### Transactional emails – shipping confirmation

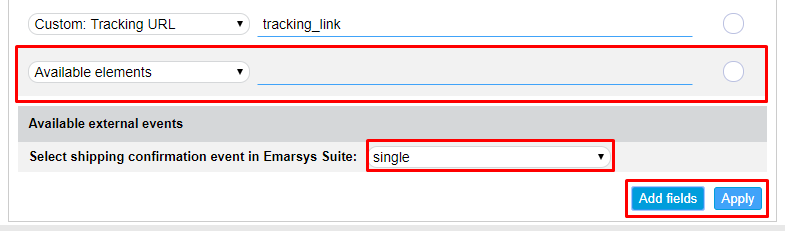
As soon as a shipment was finished an email should be sent to the end user and the corresponding external event must be triggered. The external event will trigger a shipping confirmation email.

To configure and map the Demandware fields, that will be added to the shipping confirmation email, with the Emarsys defined placeholders, a custom configuration screen has been created. Please find below the steps you need to follow to make the necessary configurations for shipping confirmation emails:

1. Navigate to **Business Manager > Merchant Tools > Emarsys Integration > Shipping Confirmation configuration**

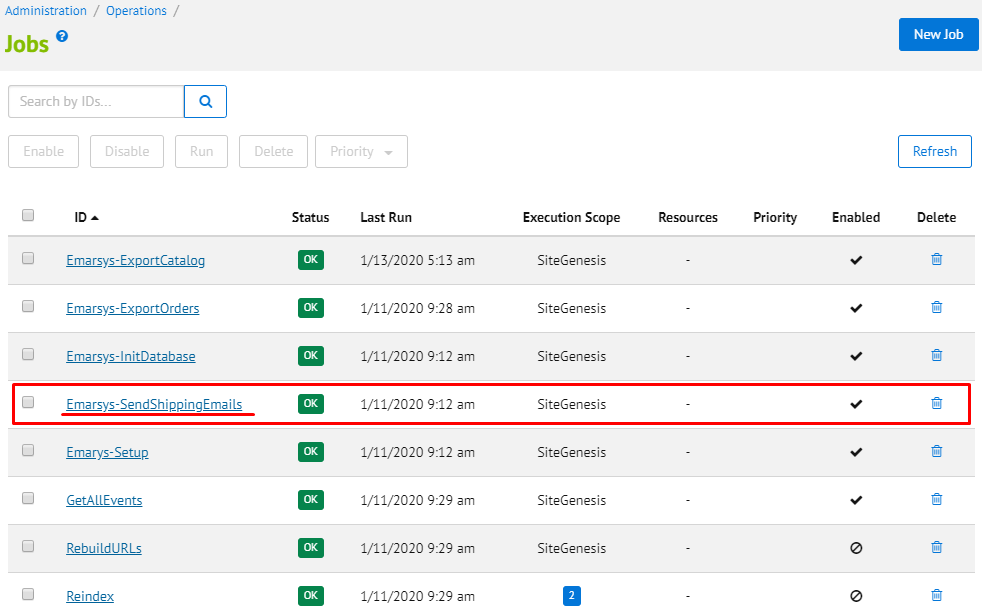


1. Click the **Add fields** button.
2. From the dropdown, select the Demandware available field and then define the **placeholder** name in the right column.
3. Select the external event that will be triggered after the shipment is finished.



1. To remove some of the defined Demandware field/Emarsys placeholder name pairs, select the corresponding *Remove* checkbox and hit the **Apply** button.

The shipping confirmation emails are driven by a job schedule (*Emarsys-SendShippingEmails*), which can be configured to run at specific intervals of time and will trigger the emails for orders with shipping status set to shipped, that are marked for Emarsys shipping confirmation email. This email is sent once per order.



Note: please keep in mind that the shipping confirmation emails are sent only for orders placed while the Emarsys integration was enabled. Also, no legacy orders will be considered for shipping confirmation emails.

Once the recurrence has been configured, the job can be start.

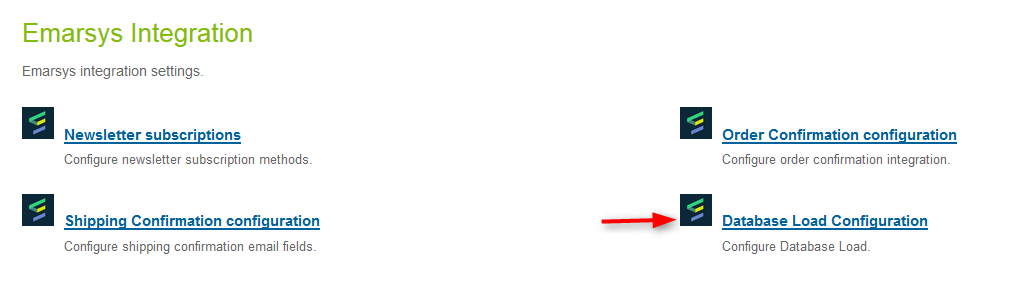
### Initial Database Load – configuration screen

The initial database load feature will upload the existing Demandware customers to the Emarsys Suite. Please find below the steps required to accomplish this :

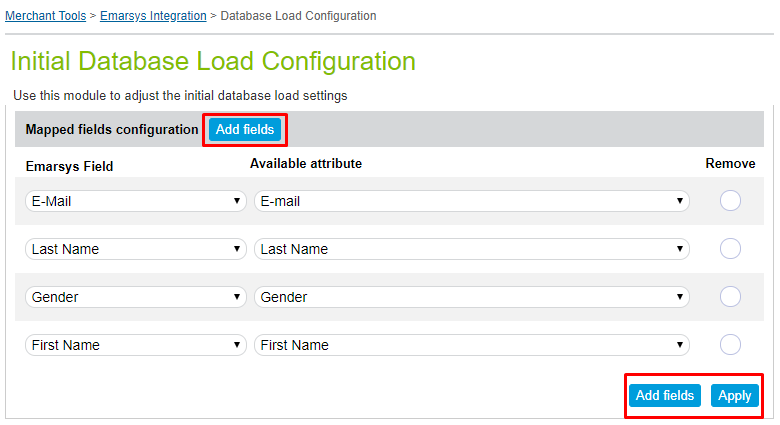
* + - Create the mapping between Demandware and Emarsys fields using the initial database load configuration screen. The mapping will be used to generate an auto-import profile on Emarsys side and to create a CSV file with the Demandware customers
    - Based on the mapping, generate the CSV file containing the Demandware customers
    - Upload the CSV file to the Emarsys WebDAV

In order to generate the mapping, a new configuration screen has been created.

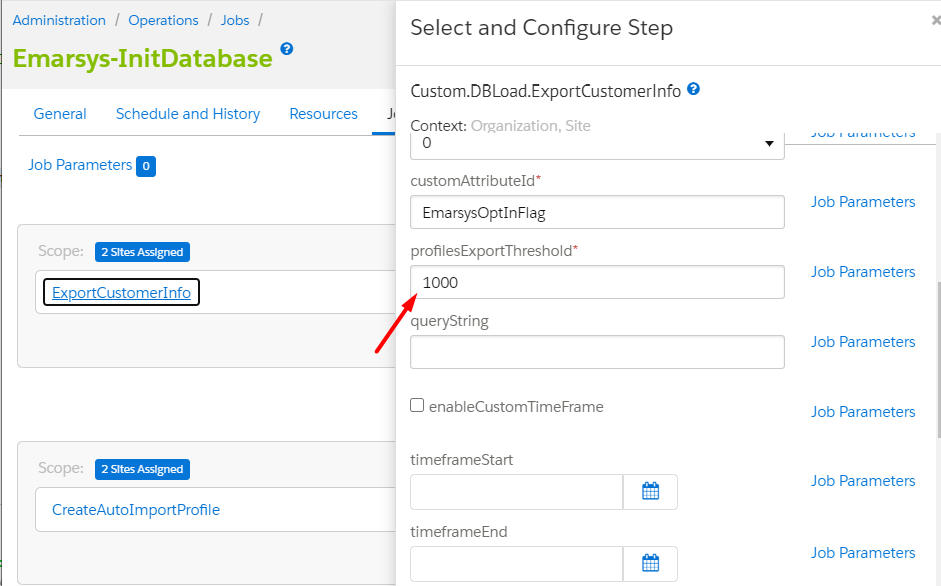
1. Go to **Merchant Tools > Emarsys Integration > Database Load Configuration**



1. Click on **Add fields**
2. Use the dropdowns to map Emarsys fields with Demandware available attributes
3. Click on the **Apply** button to save the configuration



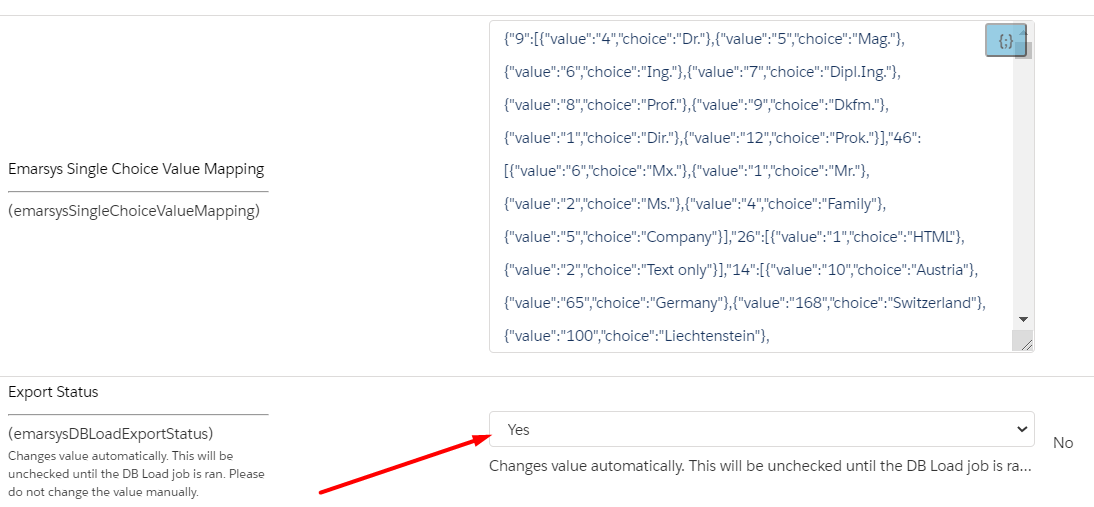
Once the settings for the initial database load feature has been made, the *Initialize Emarsys Database* job schedule needs to be configured and run. The customers will be exported to Emarsys Suite in bulks of configurable sizes. The default size of a bulk is set to 10000 customers, but the merchant has the option of updating this value inside Business Manager, by navigating to **Administration > Operations > Jobs > Emarsys-InitDatabase > ExportCustomerInfo**  and setting the value for the **Profiles export threshold** field.



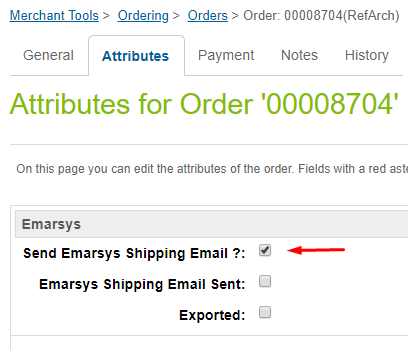
Important notes:

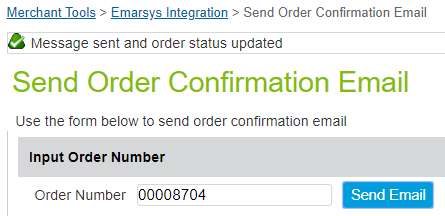
* + Please keep in mind that setting a high value for the bulk size might affect the performance. Also, the bulk size should never be set to a value higher than 20000 customers in order to avoid Demandware quota violations.
  + The recommendation is to schedule the job to run during low traffic periods.
  + Set the reccurence according to the number of exported customers per iteration.

When the job schedule is finished, the merchant will have to manually disable it. The **Export Status** field is used to prevent customers export after the initial database load was done, so that even if the job is not disabled, the export will be skipped. The value of **Export Status** field is automatically set once all the customers from Demandware were loaded into Emarsys Suite.



### Send Order Confirmation Email – send email

You can manually send order confirmation email to the customer at the order number via Emarsys service.

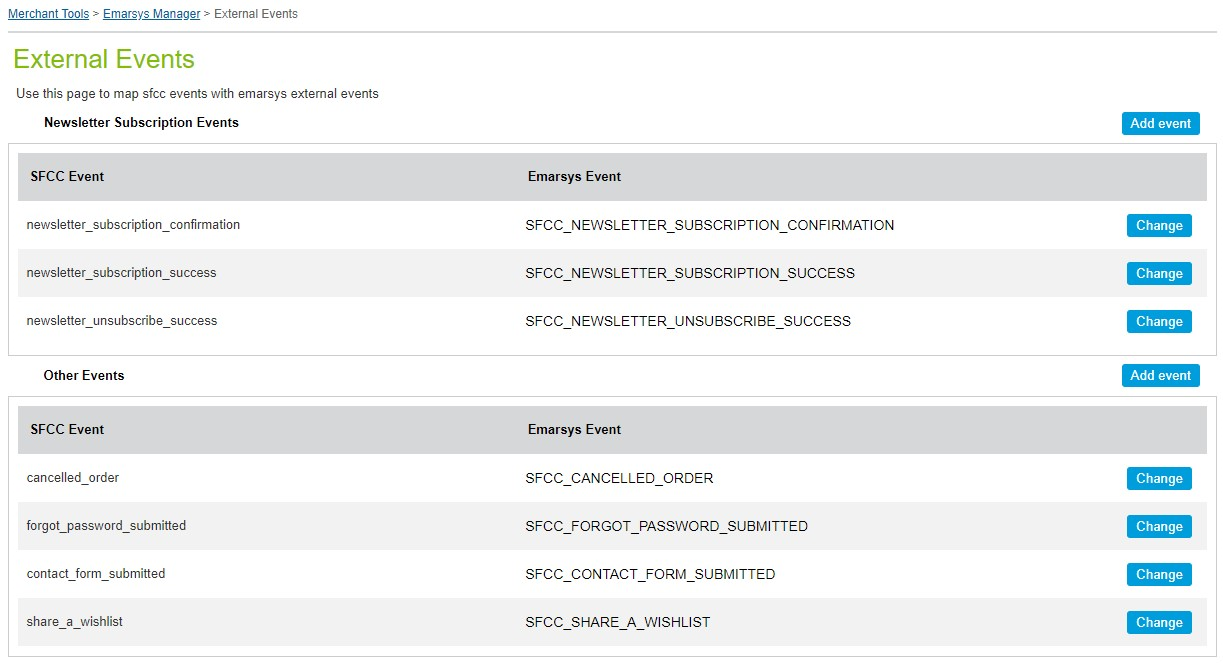


After sending the message, the order changes the status of (sendEmarsysShippingEmail).

### 5.1.8 External events page

After successful execution of Emarsys-Setup job all initial events configuration is already set. External events page is developed to change events configuration easily. In order to get there, you should:

1. Login in **Business Manager**.
2. Select your site in the left top corner of the page
3. Go to **Merchant Tools > Emarsys Integration > External Events**

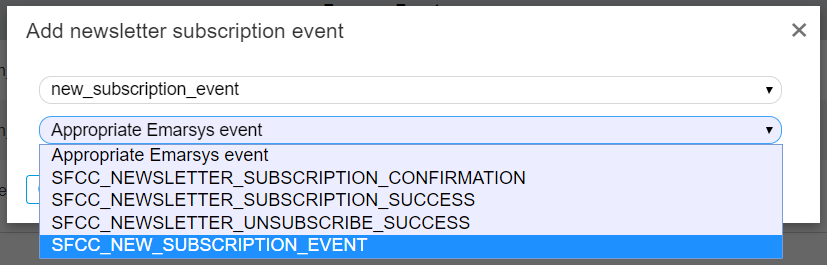


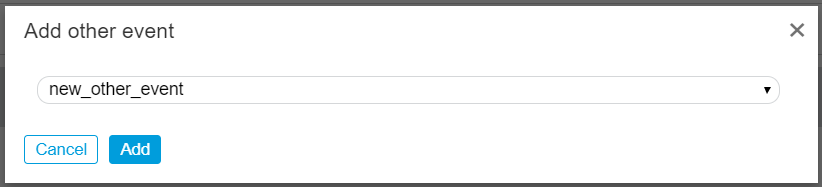
All external events are separated by its type into two tables: **Newsletter Subscription Events** and **Other Events**. Events mapping are taken from appropriate fields of **EmarsysExternalEvents** custom object.

Each table has **Add event** button over the top right corner. You should use this button to map new SFCC events. Depending on the type of the event, new sfcc event should be listed in one of **EmarsysExternalEvents** source fields (**newsletterSubscriptionSource** or **otherSource**) but should not be presented on the page (should not yet be mapped). If you want to change configuration of mapped SFCC event you should click **Change** button in appropriate row.

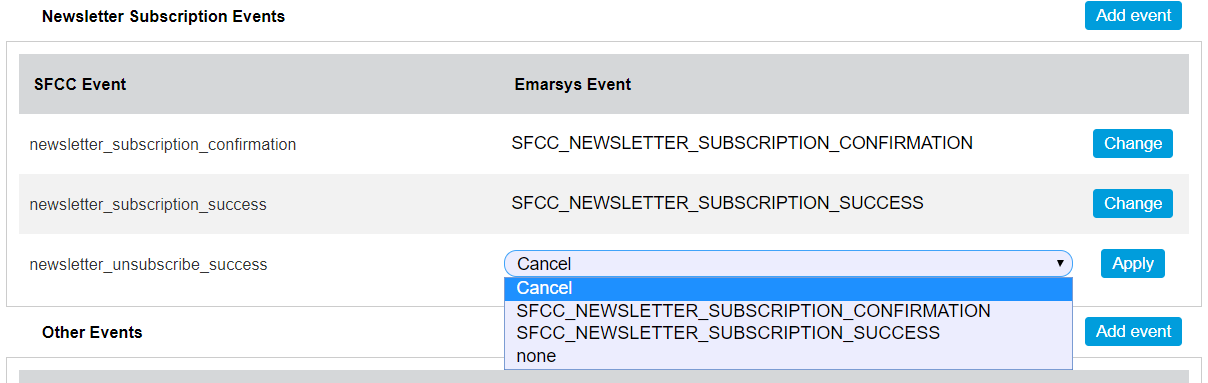
After **Add event** button was clicked, create event dialog appears. Use upper select to choose SFCC event you want to map. Events from **newsletterSubscriptionSource** field you can map with any allowed Emarsys event (of the same type), while events from **otherSource** field are allowed to be mapped only with appropriate Emarsys event. Click **Add** button to apply mapping for selected SFCC event.

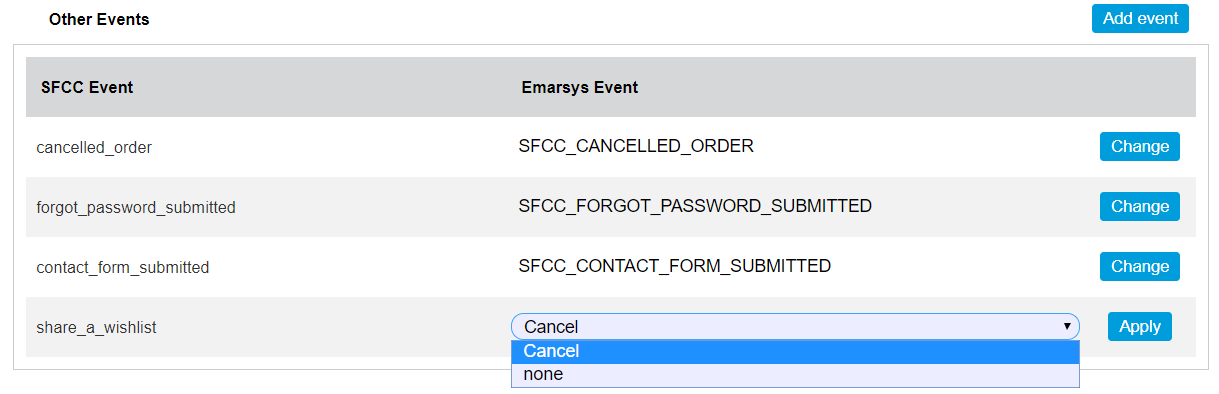
If appropriate Emarsys event is chosen for mapping, this Event will be created using Emarsys API. If there are any issues during request, error notification will be shown at the top of the events page.





Interface of the page allows to map SFCC **Newsletter Subscription Events** with any of Emarsys **Newsletter Subscription Events**. Events of other type may be mapped only with appropriate Emarsys events or to be disabled. In order to disable any SFCC event choose **none** option of mapping select. To apply mapping changes click **Apply** button.





Important notes:

* + Do not run Emarsys-Setup job after making some custom events mapping configuration because the job will rewrite the mapping. It means that all sfcc events will be mapped to appropriate Emarsys events even if they were not mapped or disabled before running

## Storefront functionality

By installing the Emarsys cartridge, the retailers will provide their end user with the option of subscribing to newsletters from the website storefront. There are 3 ways for an end user to accomplish this:

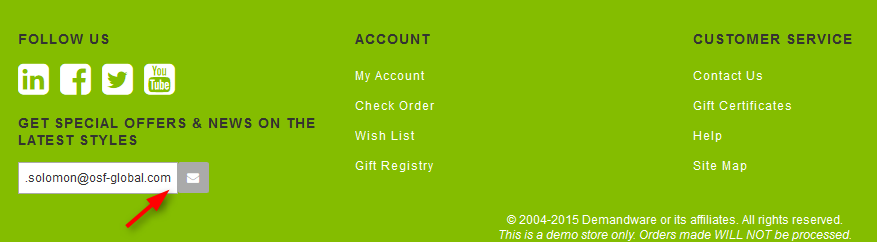
* + using the newsletter subscription box displayed on the global footer
  + during the checkout process, by selecting the *Please add me to the Demandware email list. Demandware does not share or sell personal info* checkbox on the billing page
  + from Demandware *My Account* page

All of these options are detailed in the lines below.

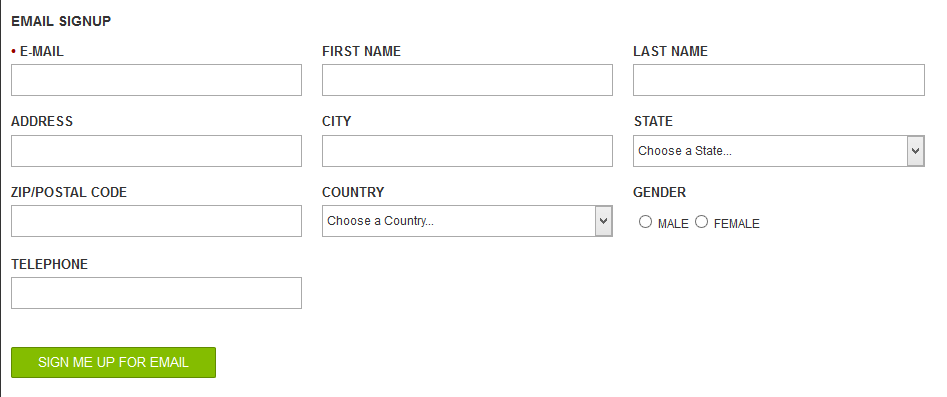
### Newsletter subscription box – global footer

The end user can subscribe for newsletters from footer on every page of the storefront. There are 2 ways to do that:

* + Subscribe directly by typing the email address in the subscription box and clicking the subscribe button.



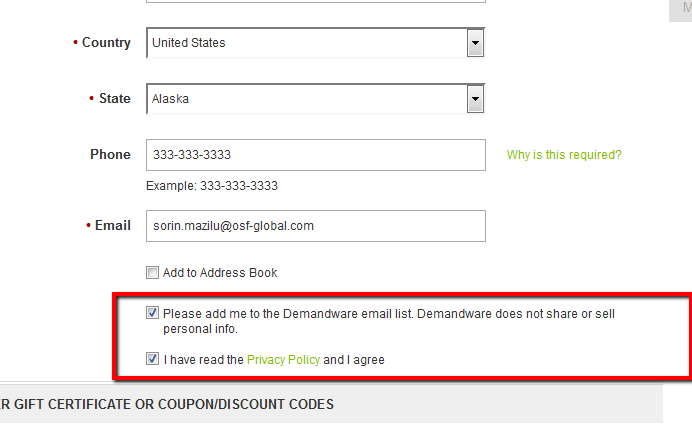
* + Clicking on subscribe button without entering any data into field. This will redirect user to **Sign up form.** Completing all the necessary fields and clicking on the **Sign me up for email** button will subscribe the user. The data is sent to Emarsys.



### Newsletter subscription during checkout process

The end user also has the option to subscribe for a newsletter during the checkout process, as described below.

1. As a guest/registered user **add a product to cart.**
2. Proceed to checkout
3. On the billing step **type in your email address** in the e-mail field.
4. Select *Please add me to the Demandware email list. Demandware does not share or sell personal info* and *I have read the Privacy Policy and I agree* checkboxes.

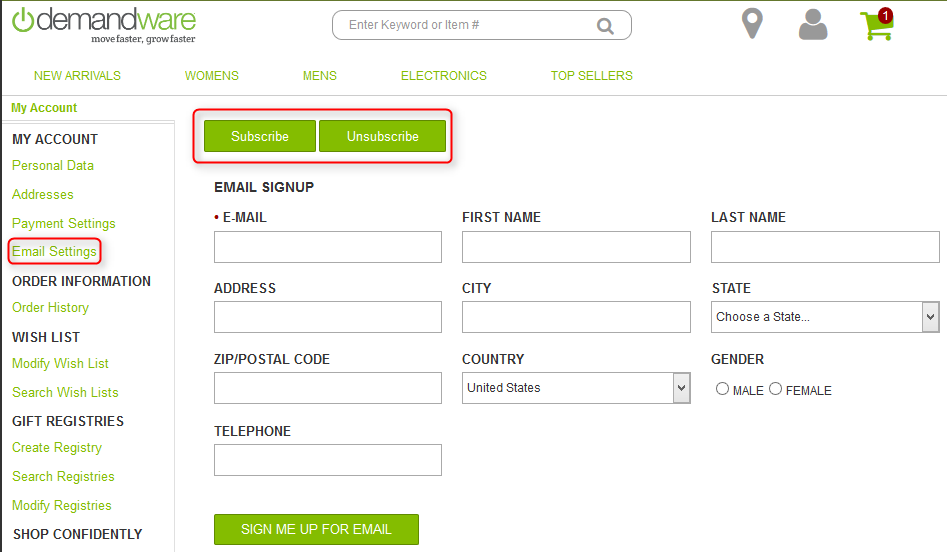


1. Complete all the mandatory fields and **proceed to next step**, so thatthe end user data to be send to Emarsys Suite.

### Newsletter subscription/unsubscription – My Account

Registered user has the option to subscribe/unsubscribe from newsletter from the Demandware *My Account* section.

1. Open the storefront and **log in.**
2. In the left navigation bar click on the **Email Settings** link.
3. Clicking the **Subscribe** button will open an email sign up form. Complete the fields and hit the **Sign me up for email** button to subscribe for newsletters.

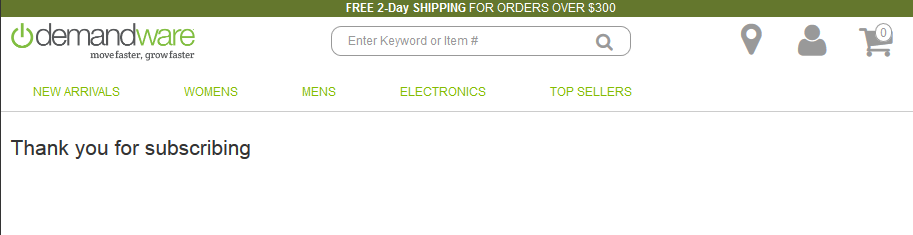


1. Clicking the **Unsubscribe** button will unsubscribe the end user from newsletters.

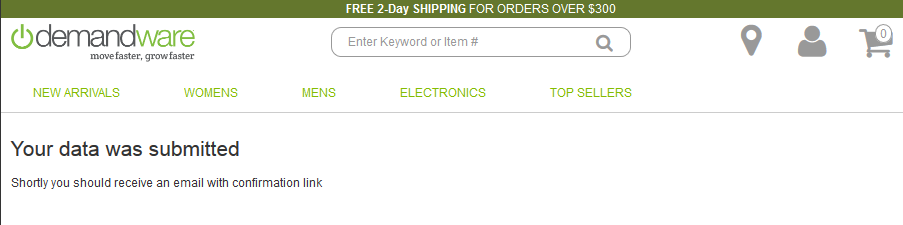
### Demandware landing pages

Depending on which opt-in strategy has been configured for the subscriptions methods and depending on the unsubscription strategy (direct/indirect), the end user can be redirected to several landing pages, as following:

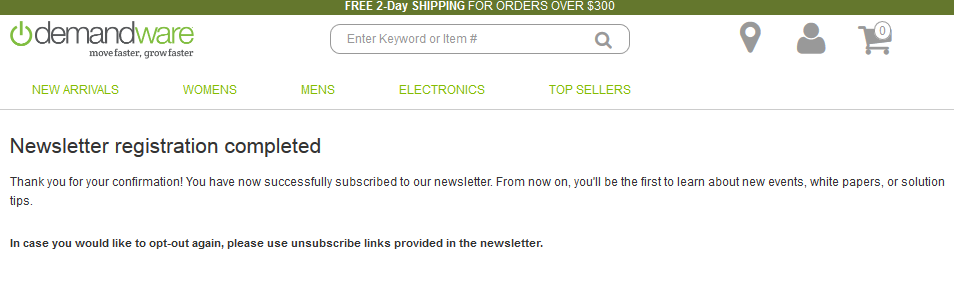
* + Thank you page – Single Opt-in strategy



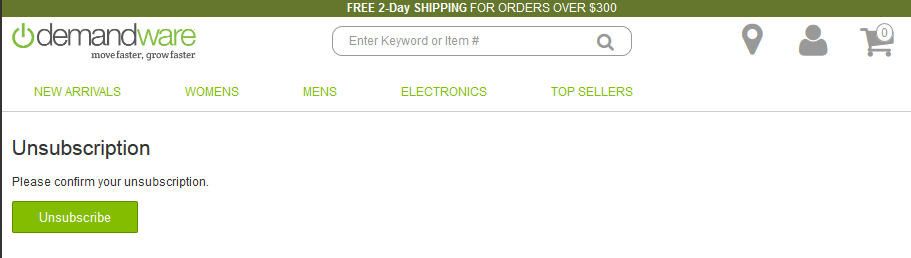
* + Data submitted page – Double Opt-in strategy



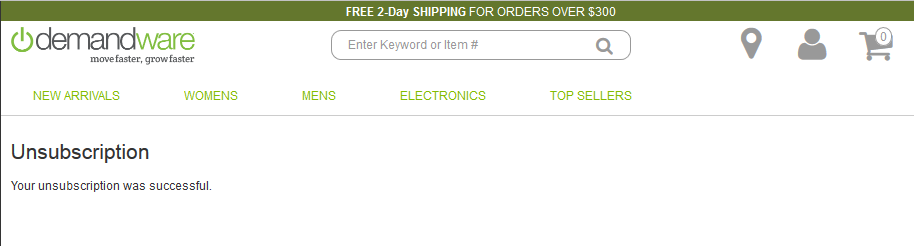
* + Thank you page – Double Opt-in strategy



* + Action page – Indirect unsubscribe



* + Thank you page – Unsubscribe



## Emarsys smartinsight

### Introduction

SmartInsight is an integrated solution for marketers that utilizes customer data from various touch points in order to provide actionable intelligence focused on customer engagement. This data can be uploaded from external sources as well as collected by the Web Extend scripts directly from your website.

### Purpose

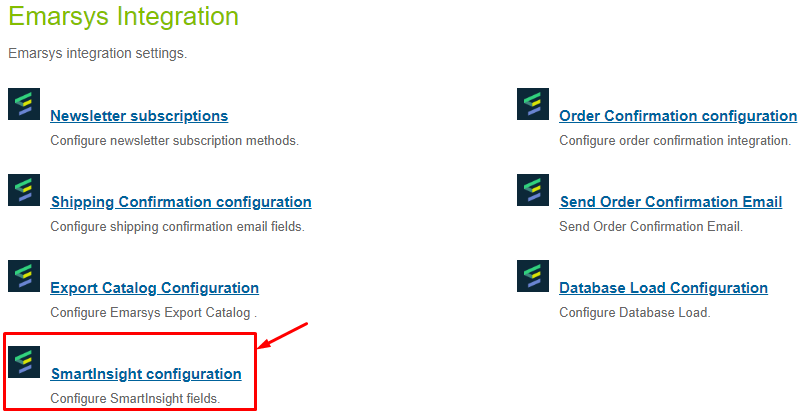
For the Emarsys needs a daily salesitems CSV feed with new orders (new orderitems) of the last 24 hours. This feed must be delivered to a FTPS server on a daily basis. For this it is required to set logon credentials for this server in the Demandware backend (hostname, username, password, destination folder).

### BM Settings for Emarsys SmartInsight attribute configuration for export

Log in to BM where Emarsys integration is enabled.

### Emarsys SmartInsight Extension Configuration

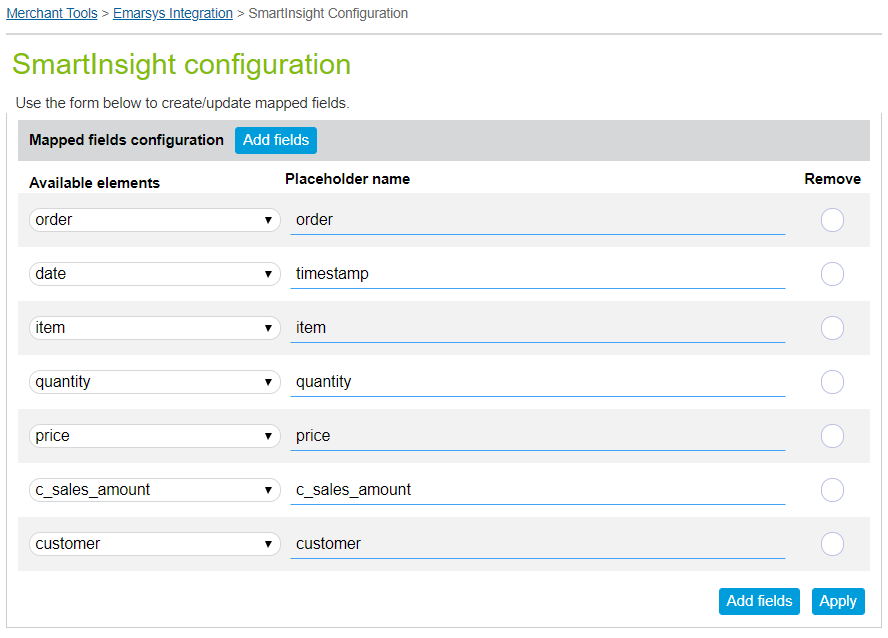
Navigate to Merchant Tools > Emarsys Integration -> You should see SmartInsight Configuration link:



In SmartInsight configuration page user can add, remove mapped fields. These fields will be used to create the CSV file with this data. These fields will be displayed as columns in exported files.

The salesitems CSV has to contain at least these columns:

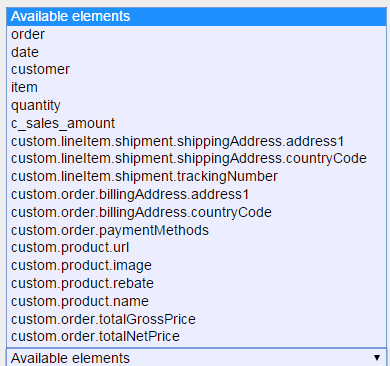
1. order– order number
2. date – date of the order (YYYY-MM-DD)
3. customer – unique customer id (this ID must be available in Emarsys Suite as well)
4. item – ProductId of the sold item (this Id must be in the products CSV as well!)
5. quantity – quantity of the ordered item
6. c\_sales\_amount – Price of the sold items (this value must be negative if an order was cancelled!)



Customer is able to add additional order attributes to the data feed. For this he should click on ‘Add fields’ link and fill Attribute name and name that will we displayed in the file. The column name of these additional columns must start with “c\_”.



Elements that were added in Site Preferences > Custom Site Preferences > Emarsys SmartInsight Configuration page are displayed in the ‘Available elements’ drop-down list.



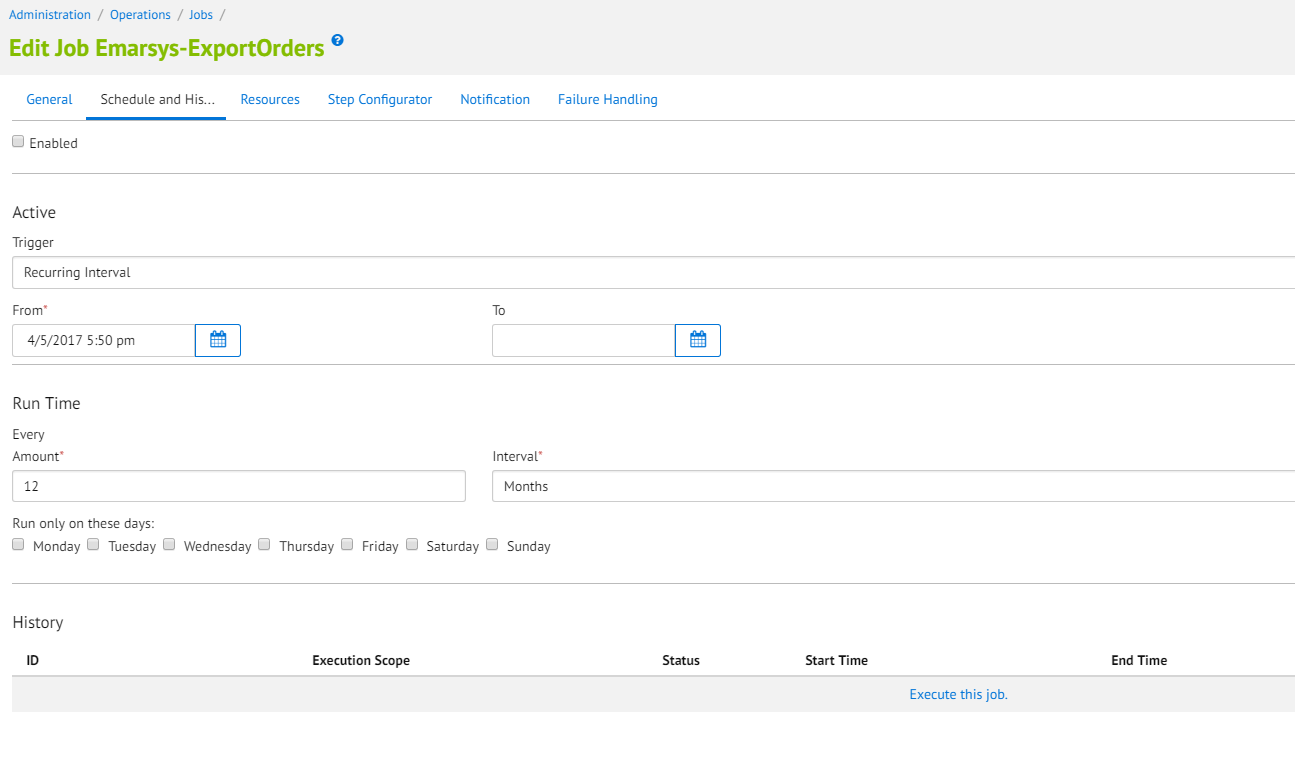
### Job for SmartInsight that exports all orders in CSV format and upload to FTPS server

Navigate to Administration > Job Schedules and click on edit for ‘Emarsys Export Orders’ job.



On this page user can set up ‘Emarsys Export Order Job’ configurations:

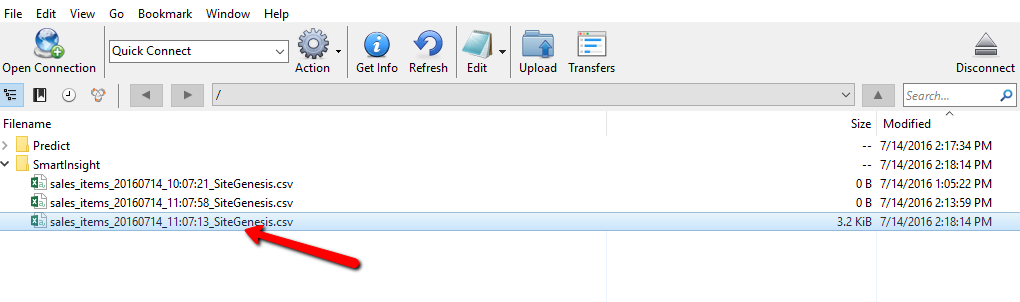
* Enabled – Checkbox which enabled or disabled ‘Emarsys Export Orders’ job on the site.
* Name – Job’s Name.
* Description – Description of this Job.
* Runtime – User can set up time and period when job will be run automatically.



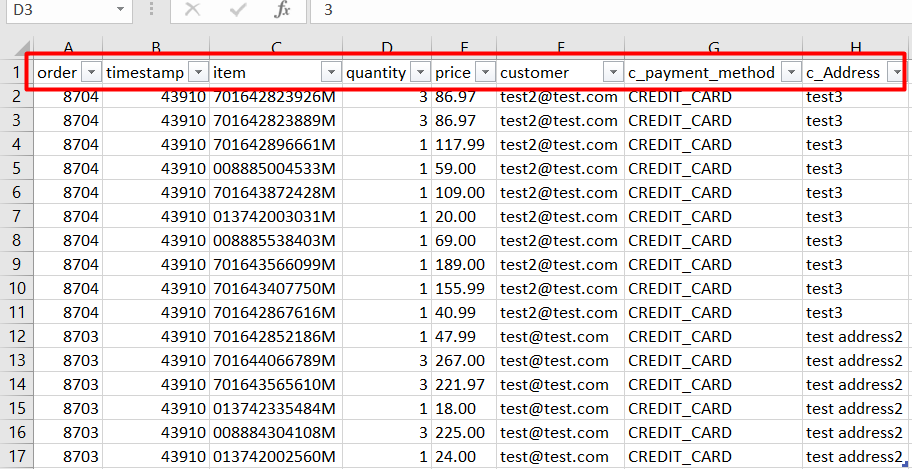
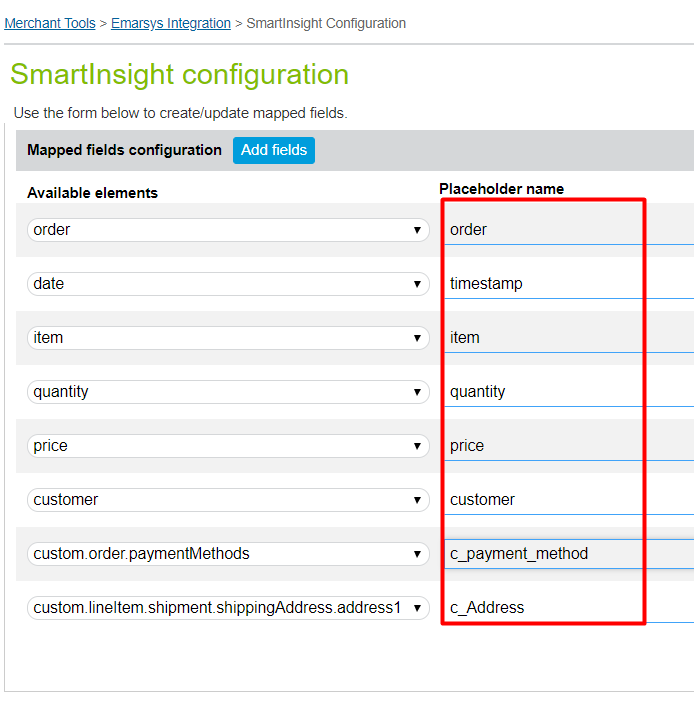
To run the job user should click on ‘Run’ button.

After finishing the successful ‘Job’ go to SFTP server and check folder with that was set up in Emarsys SmartInsight Configuration page. In this folder should be new exported file with orders.

File’s name should be in such format: sales\_items\_<YYYYMMDD\_HHiiss>\_<shopinfo>.csv



Open the new exported file with orders. All the order that fit to set up timeframe are displayed in the file with certain columns that were added into Merchant Tools > Emarsys Integration -> SmartInsight Configuration page:



## Emarsys ExportCatalog

### Introduction

Emarsys Catalog is a predictive recommendation engine which uses machine learning algorithms to offer your customers the best products wherever they interact with your business. The data used by Catalog to make these recommendations is collected by the Web Extend scripts.

### Purpose

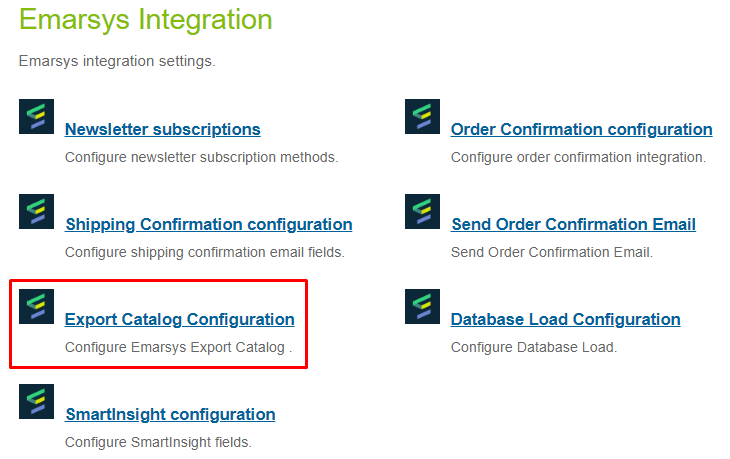
Emarsys Catalog is being used to recommend products based on a self-learning algorithm. Recommended products can be rendered within shop pages and within emails. To get this working Catalog requires a CSV product data feed which will be fetched from a remote location (HTTP / FTP / SFTP).

### BM settings for Emarsys Catalog

Log in to BM where Emarsys integration is enabled

### BM Settings for Emarsys Catalog Configuration

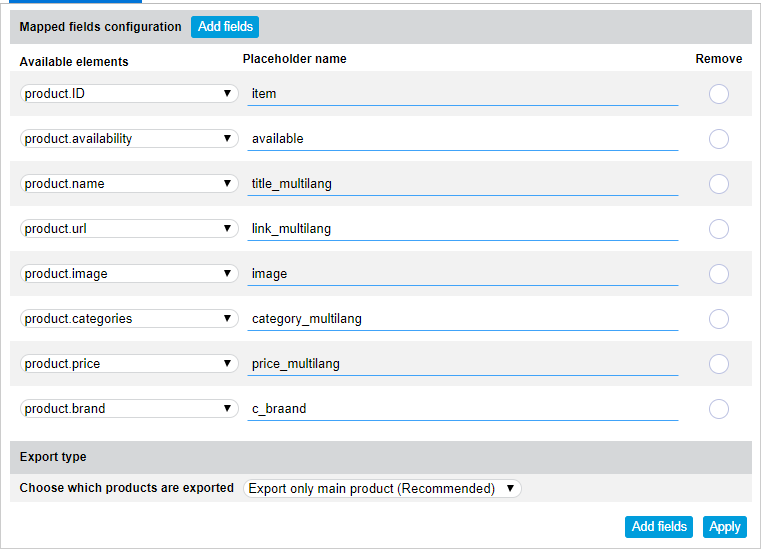
Navigate to Merchant Tools > Emarsys Integration > Export Catalog Configuration:



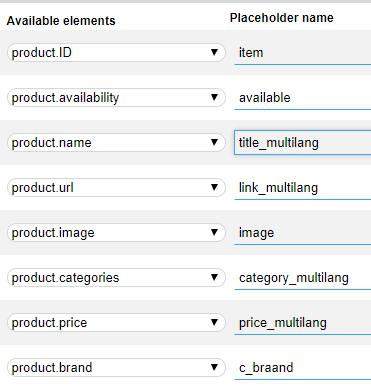
In SmartInsight Predict configuration page user can add, remove mapped fields. These fields will be used to create the CSV file with this data. These fields will be displayed as columns in exported files.

The salesitems CSV has to contain at least these columns:

1. Item – unique product id
2. Available – Is the product available (and can be recommended)?: true or false
3. Title – Product title
4. Link – Deep link to the product
5. Image – URL of the product image
6. Category – Category path to the product separated by “ > “, e.g. books > scifi > startrek
7. Price – Product price (float value 1234.99)



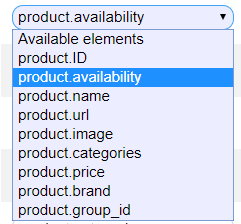
In order to send multi language values, in case if there are several locales configured, a value from the ‘Placeholder’ field must be adjusted with a ‘\_multilang’ prefix.



Customer is able to add additional order attributes to the data feed. For this he should click on ‘Add fields’ link and fill Attribute name and name that will we displayed in the file. The column name of these additional columns must start with “c\_”.

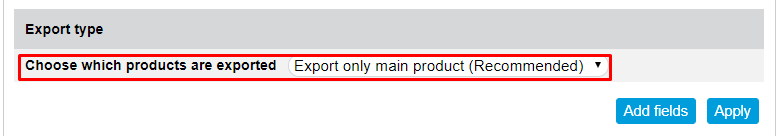


Elements that were added in Site Preferences > Custom Site Preferences > Emarsys Catalog Configuration page are displayed in the ‘Available elements’ drop-down list.

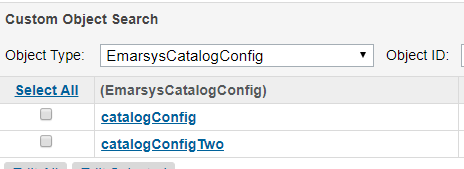


There is ability to choose Export type.

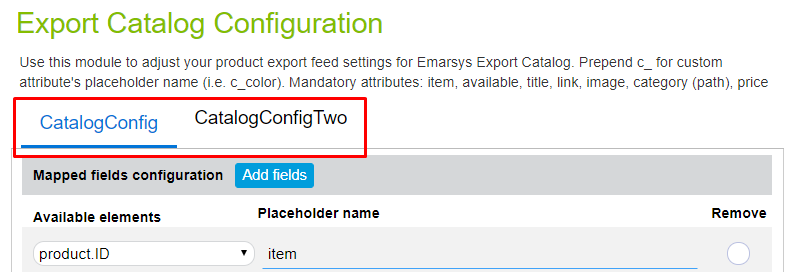
1. Export only the main product (no product version) – (default setting)
2. Export each product version / each combination of product versions



You can create new instance of the custom object in BM for individual configurations.

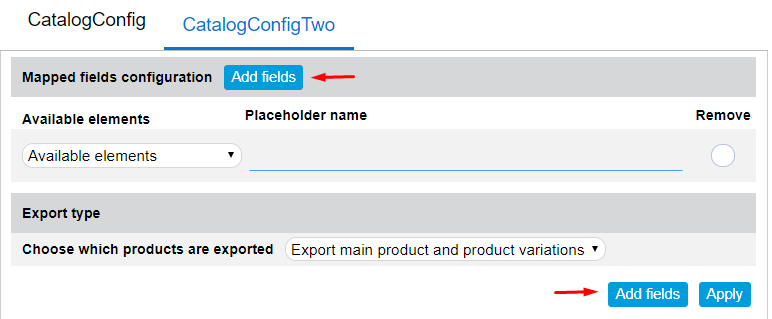


You can choose them as individual configurations.





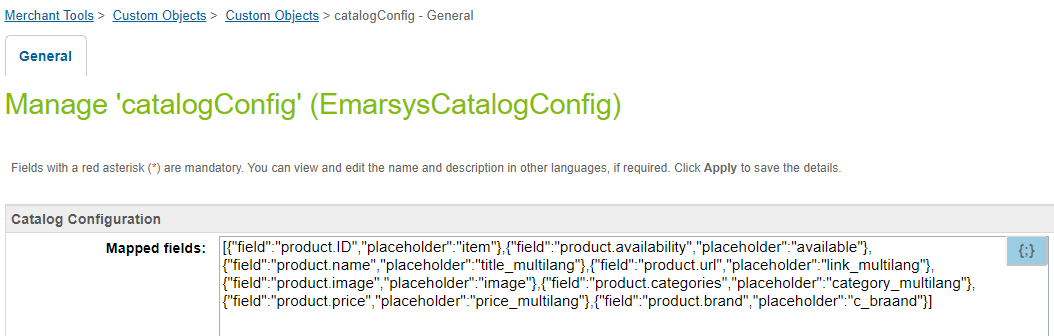
You should click on ‘Add fields’ for add new field.



Setting up the step for job in Section 3.4 EmarsysExportCatalog.

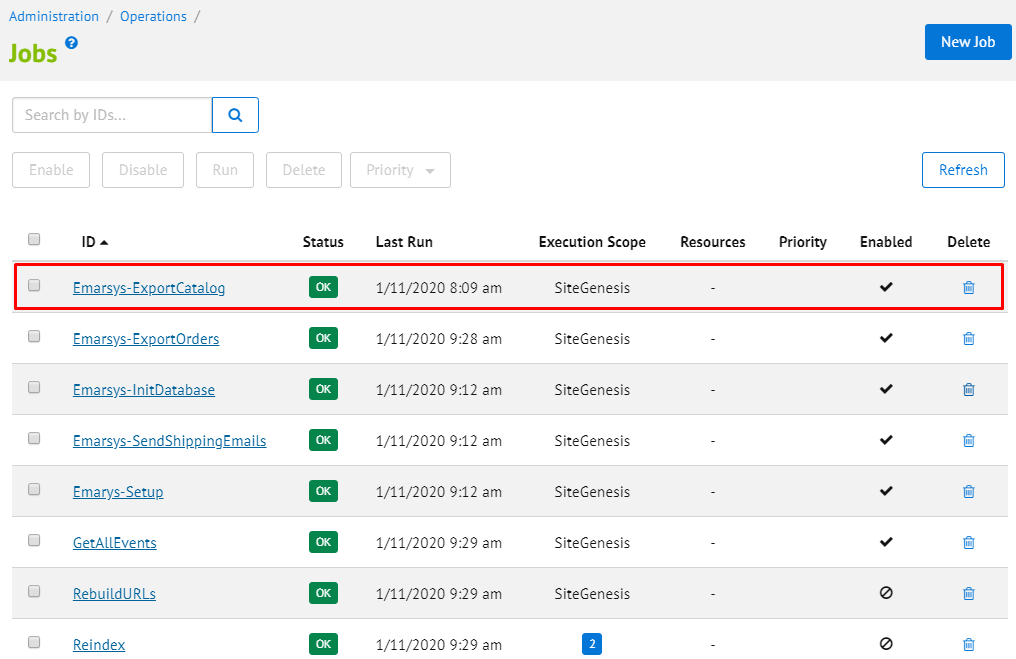
### BM Settings for Emarsys Catalog custom object

Navigate to Merchant Tools > Custom Objects > Custom Objects > catalogConfig – General and verify Mapped fields text box. In this field displayed all selected fields from Merchant Tools > Emarsys Integration > Catalog Configuration.



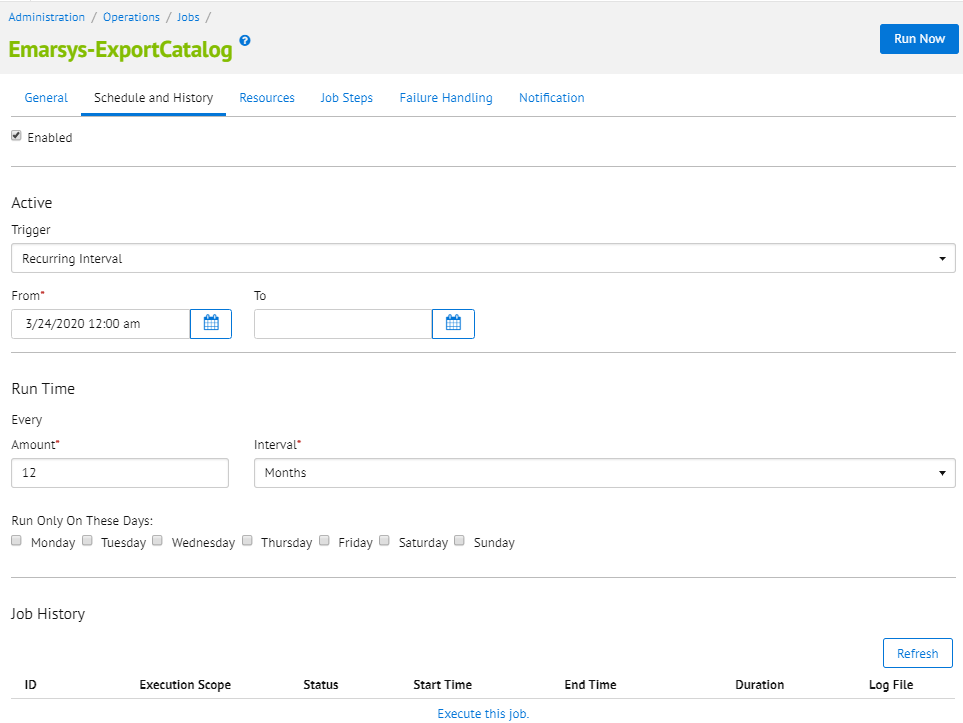
### Job for Emarsys ExportCatalog that exports all orders in CSV format and upload to FTP location

Navigate to Administration > Job Schedules and click on edit for ‘Emarsys ExportCatalog` job.



On this page user can set up ‘Emarsys ExportCatalog` Job configurations:

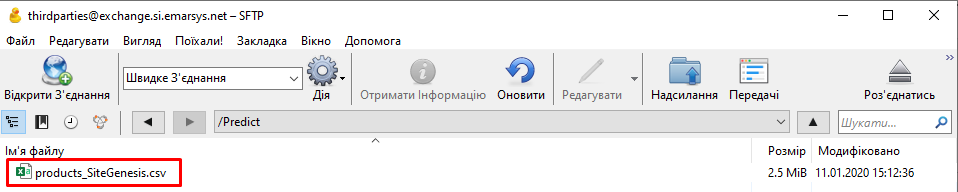
* Enabled – Checkbox which enabled or disabled ‘Emarsys Export Orders’ job on the site.
* Name – Job’s Name.
* Description – Description of this Job.
* Runtime – User can set up time and period when job will be run automatically.



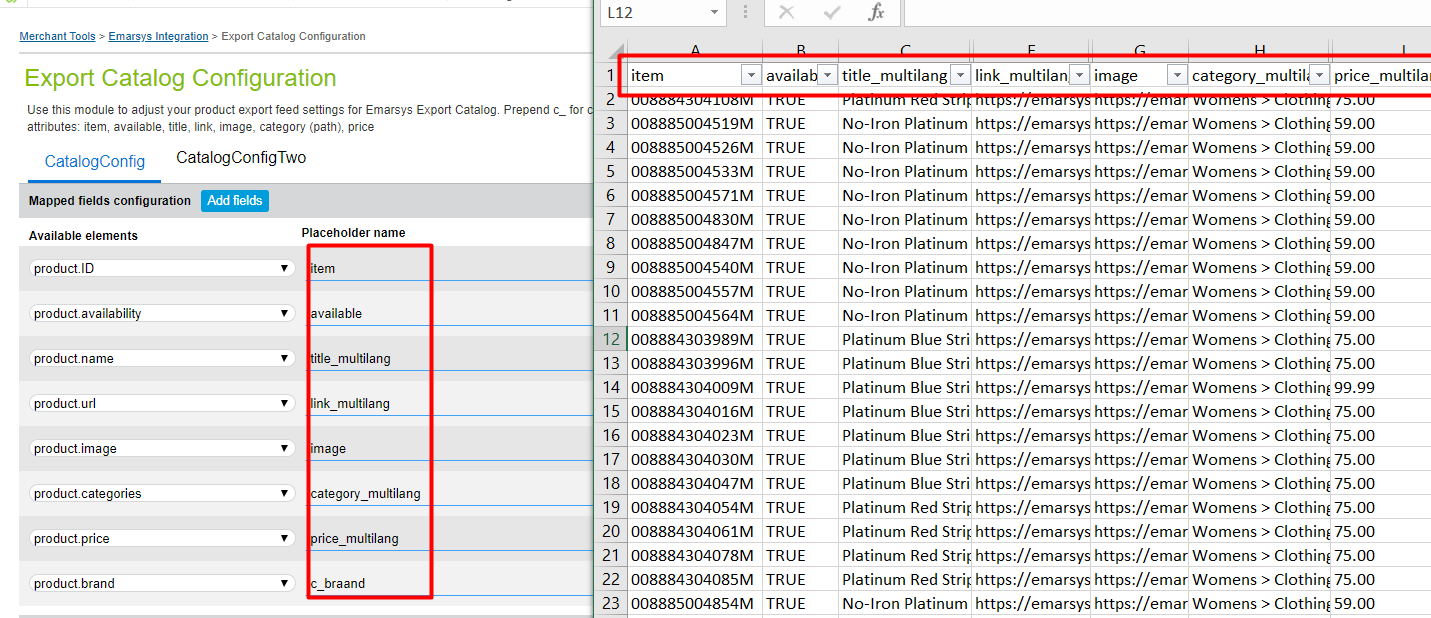
To run the job user should click on ‘Run’ button.

After finishing the successful ‘Job’ go to SFTP server and check folder with that was set up in Merchant Tools > Site Preferences > Custom Site Preferences > Emarsys Catalog Configuration page. In this folder should be new exported file with products.

File’s name should be in such format: products\_<siteName>.csv



Open the new exported file with orders. All the order that fit to set up timeframe are displayed in the file with certain columns that were added into Merchant Tools > Emarsys Integration > Catalog Configuration:



## Java script tracking code and java script recommendations

### Introduction

The Web Extend JavaScript API is a data collection system that captures the activity and behavior of website visitors. This information is used to enrich your Emarsys contact database with Web Extend fields, and to generate personalized product recommendations for the Catalog solutions.

### JavaScript recommendation code

Log in to BM where Emarsys integration is enabled

### Settings for JavaScript recommendation code

Navigate to Merchant Tools > Site Preferences > Custom Site Preferences > Emarsys Catalog Configuration link:

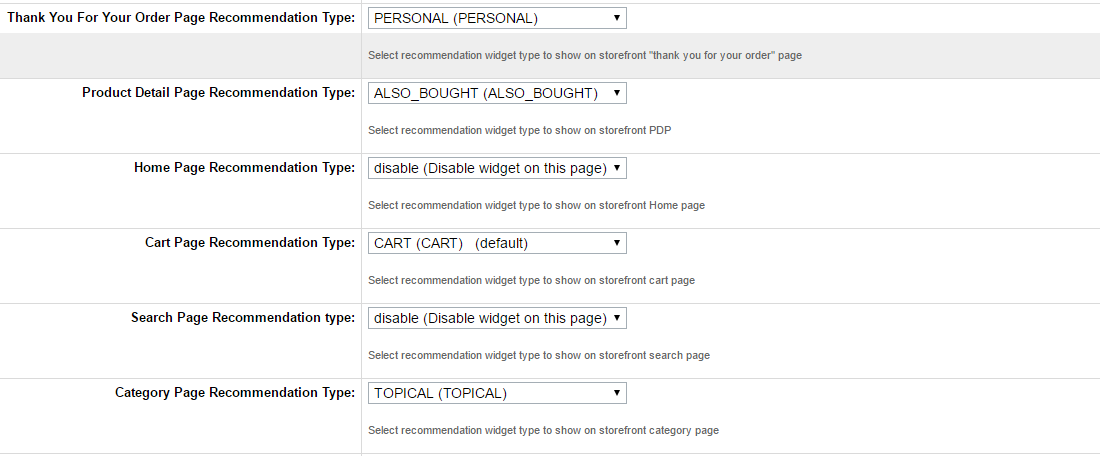


On this page user can set up Emarsys JS Configuration

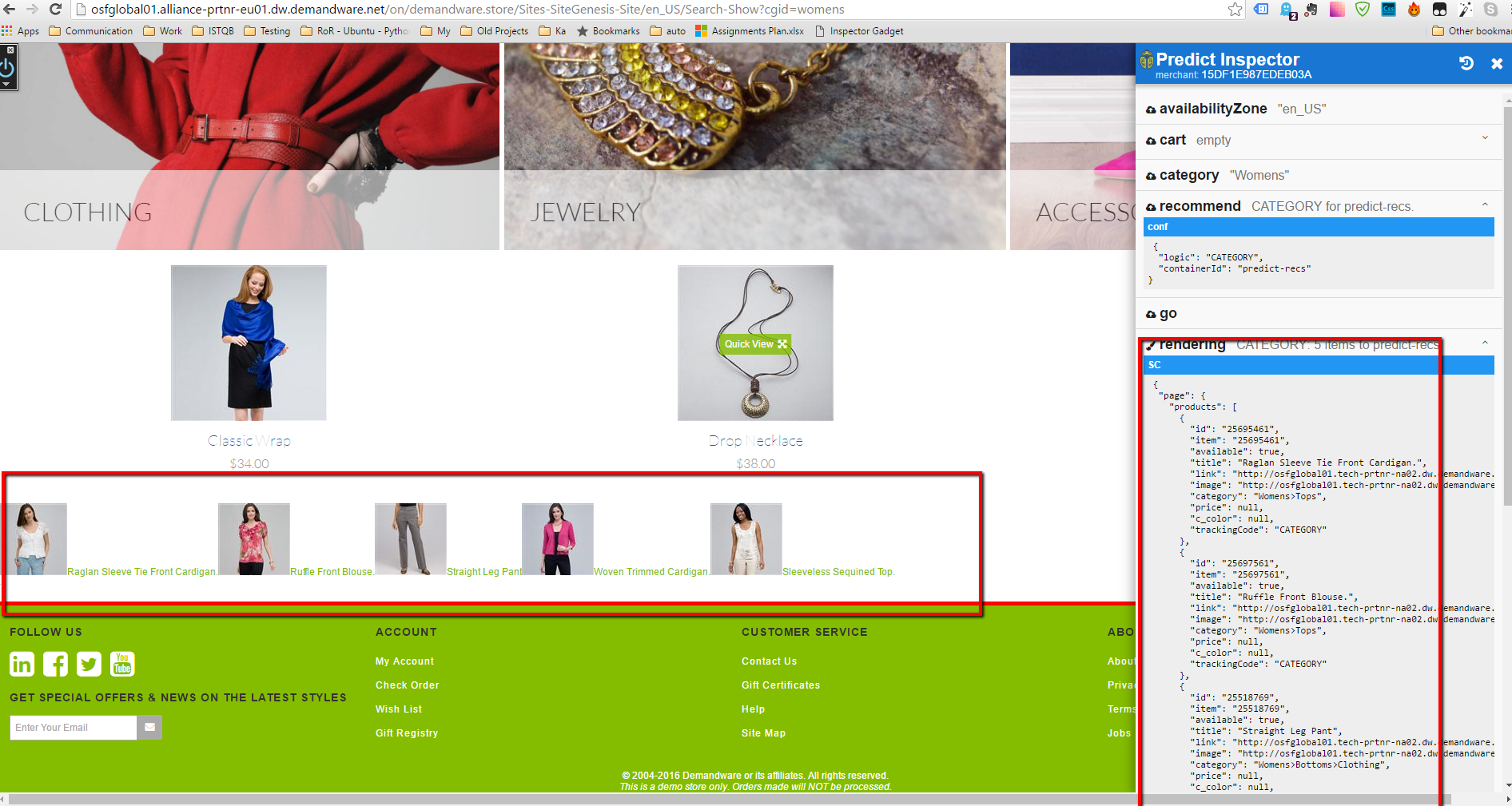
* Predict: Enable tracking code – Checkbox which enabled or disabled JavaScript tracking code on the site.



* User can select widget type to show on storefront for different pages:
* Thank You For Your Order Page
* Product Detail Page
* Home Page
* Cart Page
* Search Page
* Category Page
* Widget types:
* Personal - the Personal widget offers targeted recommendations which match a visitor’s known characteristics and interests, based on their own browse history and the wisdom of the crowd. It helps them to find products they are interested in before they even know they want them.
* Topical - this is a special widget. Each visiting customer will see a different selection of products based on what they are most likely to be interested in and displayed in sections designed to catch their interest and optimize conversion.
* Related - the Related widget suggests alternative products to the items currently being viewed. It does this by looking at the browse behavior for the whole web shop and identifying product-to-product relationships.
* Also Bought - the classic ‘people who bought x, also bought y‘ widget, Also Bought offers recommendations based on items that have frequently been bought with the item currently being viewed. This is a more targeted version of the Related widget and can be used well in combination with it.
* Cart - similar to Also\_Bought, the Cart widget offers recommendations based on the current selection of items in the customer’s shopping cart. It looks for relationships between these items and then makes additional cross- and up-sell recommendations (e.g. a spare power cord for cars, an extra SD card, etc.).
* Category - similar to the Personal widget, the Category widget directs customers to the right products, but within a particular category, depending on the specific category page they are on.



Open Storefront and visit pages where recommendation widgets must be shown.



### Settings for JavaScript tracking code

Navigate to Merchant Tools > Site Preferences > Custom Site Preferences > Emarsys Catalog Configuration link:

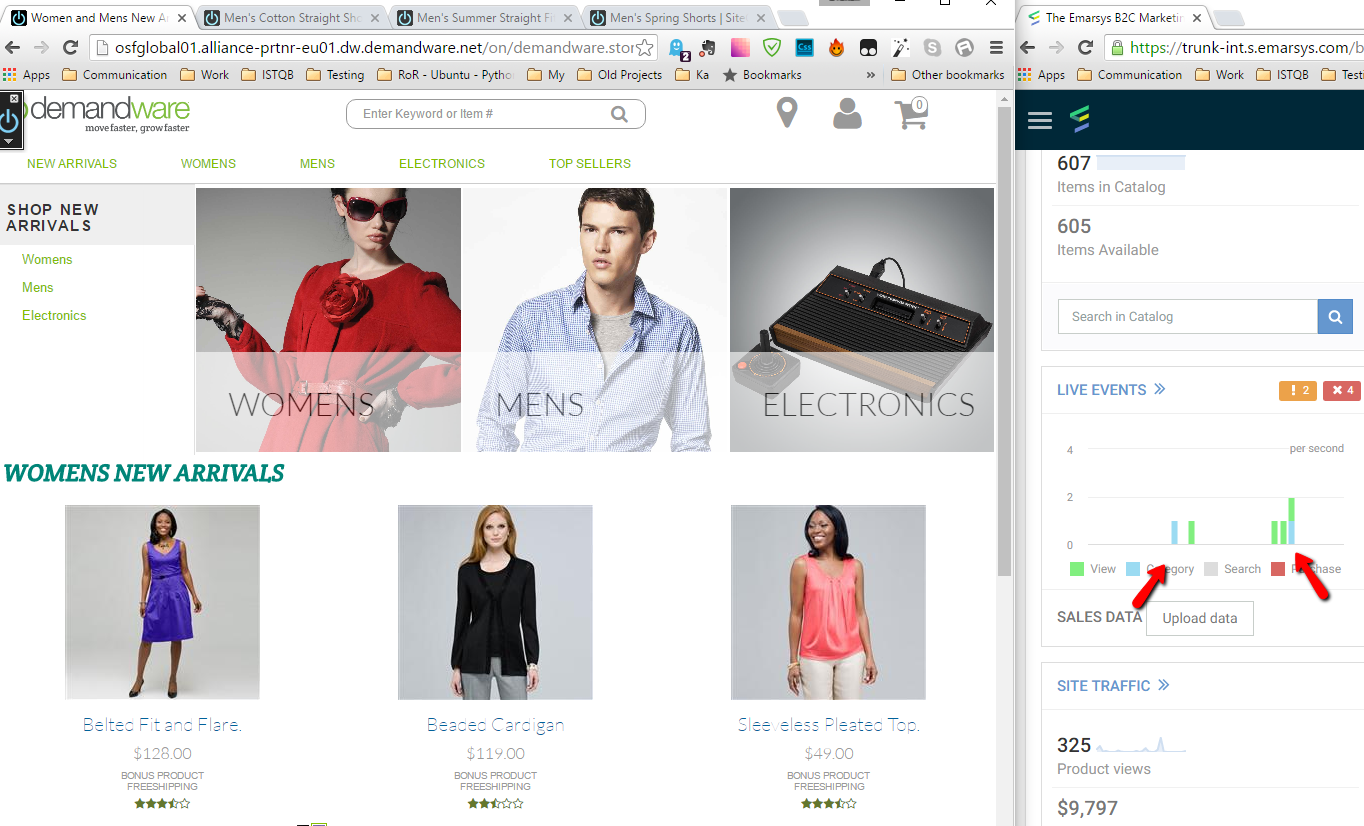


On this page user can set up Emarsys JS Configuration

* Predict: Enable tracking code – Checkbox which enabled or disabled JavaScript tracking code on the site.



Navigate to the storefront and visit pages where tracking code supposed to execute (category page, search page, pdp, etc...). Check on Emarsys predict dashboard you should see a real-time chart that shows what action you've done.



# Known issues

There are no known issues.