[Introduction](#h.nkbawgdgexrr)

[Purpose](#h.5yqivti7ay1k)

[Scope](#h.99yberltmu6s)

[Glossary](#h.nf7yj1hace0q)

[Reference materials](#h.hlop0jigtsiq)

[High-Level Design](#h.ezxewtbnqto9)

[Description](#h.l6yzlblepyv5)

[Implementation](#h.29se5grn7pty)

[Cartridges](#h.juods8vweg3m)

[BM extension](#h.y10pk2aolz40)

[Javascript snippet](#h.6u1f9uqbau4)

[Forter Send Order](#h.ab3rogw5gbak)

[Forter Decision](#h.d3w8ly8q6r85)

[Authorize.net](#h.6b3cxa42ittz)

[Schedules](#h.i63bun99bb17)

[Logging](#h.lr8ofsne4fu6)

[Meta-data](#h.hky7456mw50)

[UX / User Interface](#h.gijy4twpdqbj)

[Configuration Guide](#h.7q2c1iq7out5)

[Localization](#h.mevw547oy755)

# Introduction

## Purpose

Implement Demandware Forter cartridge which will enable easy integration with Demandware ecommerce stores.

## Scope

The following features will be implemented:

|  |  |
| --- | --- |
| Description | Details |
| Business Manager Module | OSF will create a Business Manager Module to support Forter cartridge configuration. |
| Site Preferences | * + Note: Forter will provide a configured test account.   + OSF will create the admin pipelines and templates to display the necessary information   + OSF will apply the style to the new template   + Create logic to switch between register and post registration |
| Forter Register Configuration Page | This will contain the following elements:   * + Forter Logo   + Site ID field   + Secret Key field   + Link Button   + Store Details:     - Website Name     - Website ID     - Created     - Merchant Admin Email     - Payment Gateways List and Settings (name, auth settings.)   OSF will create the event for asynchronous requests with details of the merchant store |
| Forter Post Registration Page | This will include the functionality to change the API credentials and contain the following elements:   * + Forter Logo   + Extension Version Static text   + Enabled (True/False)   + Communication     - Show default decline page     - Multiline textbox   + Auto Invoice when transaction is approved checkbox   + Cancel order when transaction is declined checkbox   + Change API credentials link     - Site ID   + Secret Key |
| Orders View | OSF will create a screen to view new orders, a new node and logic to query and sort orders; orders will be able to be sorted by the Forter decision column.  OSF will create the template to render the query, with pagination. |
| JavaScript Snippet | JavaScript Snipped will be injected into the footer section for all webpages on the platform; the configured Site ID will be included in the JavaScript snipped.  OSF will create the html hook to add the javascript snippet, create helper library for the requests and create the headers for request. |
| Sending Order | Order data will be sent to Forter API Endpoint during the checkout process.  Order data to be sent:   * Payment gateway result * Order details * Customer information data   OSF will create the method to send order data, create the logic based on response from send order request and set status (Forter decision) and add custom attributes to the request |
| Forter Decision | OSF will implement the logic of receiving the JSON format response from Forter platform. |
| Status Update Notification | When order status changes, the new status will be sent to Forter API endpoint via HTTPS;  The format will be JSON.  OSF will create a job to set the Forter status for timed out requests |
| Integrate Authorize.net payment gateway | OSF will integrate the Forter cartridge with Authorize.net payment gateway  Note: Forter will provide a configured Authorize.net account  BM will be configured to use the Authorize.net processor and OSF will create the pipeline to authorize payment, and the script to authorize payment with Authorize.net and pass the value to Forter request |
| Customer Account Updates | OSF will create a method to send the account data |
| Log | OSF will create logging for order data answer; logs will be available only for merchants.  OSF will also clean-up the logs and check against deprecated API |
| Error Handling Mechanism | OSF will create payload and request for Forter related error logging. |

## Glossary

## Reference materials

* Forter DW Cartridge Vision and scope document 0.4.docx
* Forter-ext-v1.pdf
* <https://projects.invisionapp.com/share/G25ARVCUS#/screens/111403485/comments>

# High-Level Design

## Description

The Forter cartridge implementation will have the following major functionalities:

1. **BM extension module**
2. **Forter integration**
3. **Authorize.net integration**
4. **Schedules**

# Implementation

## Cartridges

The following cartridges will be presented:

* **int\_forter** - this cartridge will contain all the Forter integration code, needed for the API calls, logging, jobs execution, etc.
* **bm\_forter** - this cartridge will contain the Forter business manager extension functionality, needed for building the pre- and post-registration pages and orders view page
* **int\_authorizenet** - this cartridge will hold the Authorize.net integration code, needed for providing the payment processing - authorize and capture payment
* **app\_forter** - this cartridge will contain only the changed SG templates, thus integrators and developers can easily find what is changed in base SG functionality during Forter integration process
* **app\_storefront** - this cartridge is the latest Demandware SG cartridge at the time of the start of the development which won’t be modified

## BM extension

Forter business manager extension will consists of two templates:

* **forterregistration.isml** - responsible for the pre- and post-registration process
* **fortersearchorder.isml** - responsible for the searching and sorting forter orders

A **Forter.xml** pipeline will be created, handling the registration process and order search functionality.

The configuration will be saved into site preferences / custom object.

All the needed Forter logos and images will be located into: **cartridge/static/default/icons.**

Translations will be located into **cartridge/templates/resources/forter\_bm.properties.**

## Javascript snippet

Forter requires the usage of a provided javascript snippet. It will be added into a **fortersnippet.isml** template. After that this template will be included into the SG footer, thus being loaded on all site pages. The snippet code will be the following:

<script type="text/javascript">

(function() {

var siteId = [SITEID]

, url = '//cdn4.forter.com/script.js?sn=' + siteId,

timeout = 10;

setTimeout(function() {

var f = document.createElement('script')

f.type = 'text/javascript'; f.id = 'ftr\_\_script'; f.async = true;f.src =

('https:' == document.location.protocol ? 'https:' : 'http:') + url;var s =

document.getElementsByTagName('script')[0];s.parentNode.insertBefore(f, s);

}, timeout);

window.ftr\_\_startScriptLoad = 1\*new Date();

})();

</script>

## Forter Send Order

During the checkout process after a customer has clicked the checkout button all the order data, customer info and payment gateway details will be sent to Forter. This call will be implemented using the Demandware service framework. The following code structure will be used:

* **cartridge/scripts/init/forterServiceInit.ds** - base service configuration
* **cartridge/scripts/lib/forter/services/ForterSendOrderService.ds** - handles the request

## Forter Decision

The response of sending the order to Forter will be handled as updating the order status accordingly, logging the response data and showing the appropriate page based on the Forter configuration. The logic for these actions will be in **COPlaceOrder.xml.** Appropriate pipelets will be created for handling the described logic.

## Authorize.net

A **int\_authorizenet** cartridge will be created. It will be responsible for handling the payment with credit cards. In order to make this integration working, the following payment processor will be created: **AUTHORIZE\_NET**. There should be a pipeline created with the same name, containing **Handle** and **Authorize** start nodes. These nodes will be responsible for the payment handling in Demandware.

## Schedules

Three jobs will be created:

* **Order status update job** - Orders will be extended with a new attribute that will contain the last known status. The job will run every x hour and will pick up the orders changed since last run time. If status of order is different than last known status, it will do the request to Forter and update the attribute value.
* **Forter status update job** - a job to set Forter status for timed out requests. It will check the orders with no Forter status and will do a retry request.
* **Order void job** – a job to void authorize.net payments for orders being in the system more than 24h and having the capture payment declined.

# Logging

A Forter specific logging script will be created. It will be responsible for logging all the needed and erroneous communication between Forter and Demandware. It will be used in all the request-response communications, jobs, order updates, etc. The following file will be created: **int\_forter/cartridge/scripts/lib/forter/ForterLogger.ds**

# Meta-data

All the metadata will be stored into a separate directory - **metadata**. Only additional to the default SG properties and configurations will be listed there.

# UX / User Interface

All mockups for the Forter business manager extension are provided on the following site:

<https://projects.invisionapp.com/share/G25ARVCUS#/screens/111403485/comments>

# Configuration Guide

A separate document prior to Demandware certification will be created which will holds all the configuration and screenshots. The document will be created after the Forter cartridge code completion.

# Localization

There will be localization properties files related to each cartridge. Initially only their English version will be created.

# 