***Version 18.1.0***

****

**int\_sezzle**

**Table of Contents**

1. Summary 3

2. Component Overview 3

2.1 Functional Overview 3

2.2 Use Cases 3

2.3 Limitations, Constraints 3

2.4 Compatibility 3

2.5 Privacy, Payment 3

3. Implementation Guide 4

3.1 Setup 4

3.1.1 Deploying cartridge to a sandbox 4

3.1.2 Sandbox setup

3.2 Configuration 5

3.2.1 Site Preferences configuration 5

3.2.2 Sezzle Services configuration 5

3.2.3 Sezzle Job configuration

3.3 Custom Code (Controllers) 6

3.5 External Interfaces 10

3.6 Firewall Requirements 11

3.7 Testing 11

4. Operations, Maintenance 12

4.1 Data Storage 12

4.2 Support 12

5. User Guide 13

5.1 Roles, Responsibilities 13

5.2 Business Manager 13

5.3 Storefront Functionality 13

6. Known Issues 13

7. Release History 13

# Summary

This document describes how to implement Sezzle cartridge in SiteGenesis site. This cartridge can be configured in the Business Manager and contains all elements necessary to perform a successful best practices implementation of Sezzle.

# Component Overview

## Functional Overview

Sezzle is an alternative payment platform that increases sales and basket sizes by enabling interest-free installment plans at online stores. Consumers pay over time, but our merchant partners are paid upfront, eliminating risk of fraud or non-payment.

When you pay with Sezzle, your purchase is split into four interest-free installments automatically scheduled over the next six weeks. It's a financially responsible way to pay over time and build credit.

## Use Cases

Customers can use Sezzle payment method to pay their purchases.

## Limitations, Constraints

The Sezzle cartridge does not have support for partial refunds.  These can be placed in the Sezzle Merchant Dashboard. Sezzle’s product does not allow for greater than the amount of the original purchase.

## Compatibility

Cartridge is designed and developed for: Salesforce platform version 17.1, SiteGenesis 2.0 version 104.0.1 and Compatibility mode 16.2

## Privacy

Sezzle’s privacy agreement can be found on our legal website at https://legal.sezzle.com/privacy

# Implementation Guide

## Setup

Section describes steps that should be completed before cartridge configuration in Business Manager.

### Deploying cartridge to a sandbox

1. Download cartridge source code.
2. Import Sezzle cartridge to a workspace in Salesforce UX Studio.
3. Add Sezzle cartridge to Project Reference of Server Connection.
4. Wait until Studio completes workspace built and uploading source codes to a sandbox.

### Sandbox setup

1. Go to Business Manager -> Site -> Manage Sites. Select correct site, then select Settings tab. In cartridge path at the end write the following:

**:int\_sezzle**

1. Download cartridge source code.
2. Upload and import metadata from the **site\_import** folder. To do so, go to **Business Manager > Administration > Site Development > Site Import & Export**. Then apply the standard procedure for uploading metadata into the Commerce Cloud site. You can compress the **site\_import,** after renaming the site name inside it to your site, to a .zip archive or upload it via XML files and import it separately.

## Configuration

This section describes configuration of the sandbox.

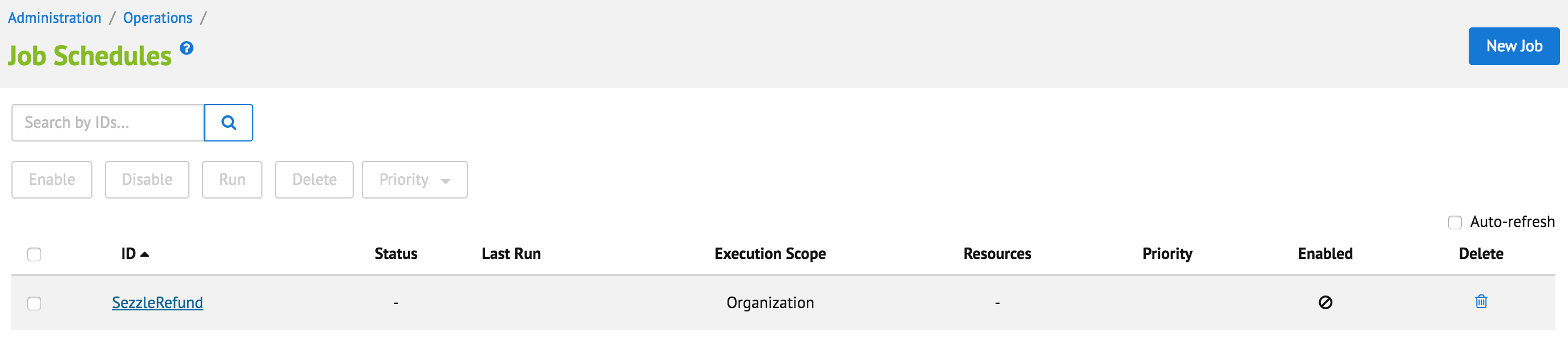
### Site Preferences configuration

1. Go to Merchant Tools > Site Preferences > Custom Site Preferences > Sezzle
2. Enable attribute - Sezzle Online Status. This attribute defines status (enable/disable) of Sezzle integration.
3. Select a value from dropdown Sezzle Mode. This attribute defines in which mode cartridge will work. Allowable values are “Sandbox” and “Production”.
4. Add site preference attribute - Sezzle Public Key with provided public key from Sezzle.
5. Add site preference attribute - Sezzle Private Key with provided private key from Sezzle.

### Sezzle Services configuration

1. Go to Administration > Operations > Import & Export.
2. Click Upload button and select sezzle\_services.xml
3. Click Import button and select sezzle\_services.xml.

### Sezzle Job configuration

1. Go to Administration > Operations > Import & Export.
2. Click Upload button and select sezzle\_jobs.xml.
3. Click Import button and run import of sezzle\_jobs.xml.
4. Go to Administration > Operations > Job Schedules
5. Verify that each imported job was created.
6. Change execution scope of every job to your site.

## Custom Code (Controllers)

This section describes changes that should be made to a merchant storefront cartridge.

1. Find and open the below template:

storefront/cartridge/templates/default/components/footer/footer.isml.

Paste the below code at the bottom the file:

<isinclude template=*"sezzle/sezzlefooter"* />

1. Find and open template – storefront/default/checkout/billing/paymentmethods.isml. Find the following code:



Replace it with the next code.

<isif condition=*"${paymentMethodType.value.equals('Sezzle')}"*>

<isinclude template=*"sezzle/paymentmethodinput"* />

<iselse/>

<div class=*"form-row label-inline"*>

<isset name=*"radioID"* value=*"${paymentMethodType.value}"* scope=*"page"*/>

<div class=*"field-wrapper"*>

<input id=*"is-${radioID}"* type=*"radio"* class=*"input-radio"* name=*"${pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlName}"* value=*"${paymentMethodType.htmlValue}"* <isif condition=*"${paymentMethodType.value == pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlValue}"*>checked="checked"</isif> />

</div>

<label for=*"is-${radioID}"*><isprint value=*"${Resource.msg(paymentMethodType.label,'forms',null)}"*/></label>

</div>

</isif>

Find the following code:

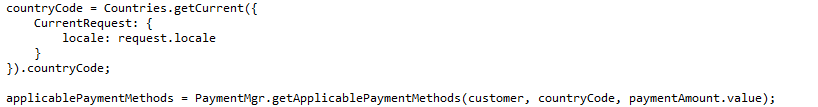
image73.png

Paste code after:

<isinclude template=*"sezzle/sezzlepaymentmethod"* />

1. Find and open controller:

storefront/cartridge/controllers/COBilling.js. Find the following code (lines 139-145):



Paste code after:

applicablePaymentMethods = require('int\_sezzle/cartridge/controllers/Sezzle').Init(cart, applicablePaymentMethods);

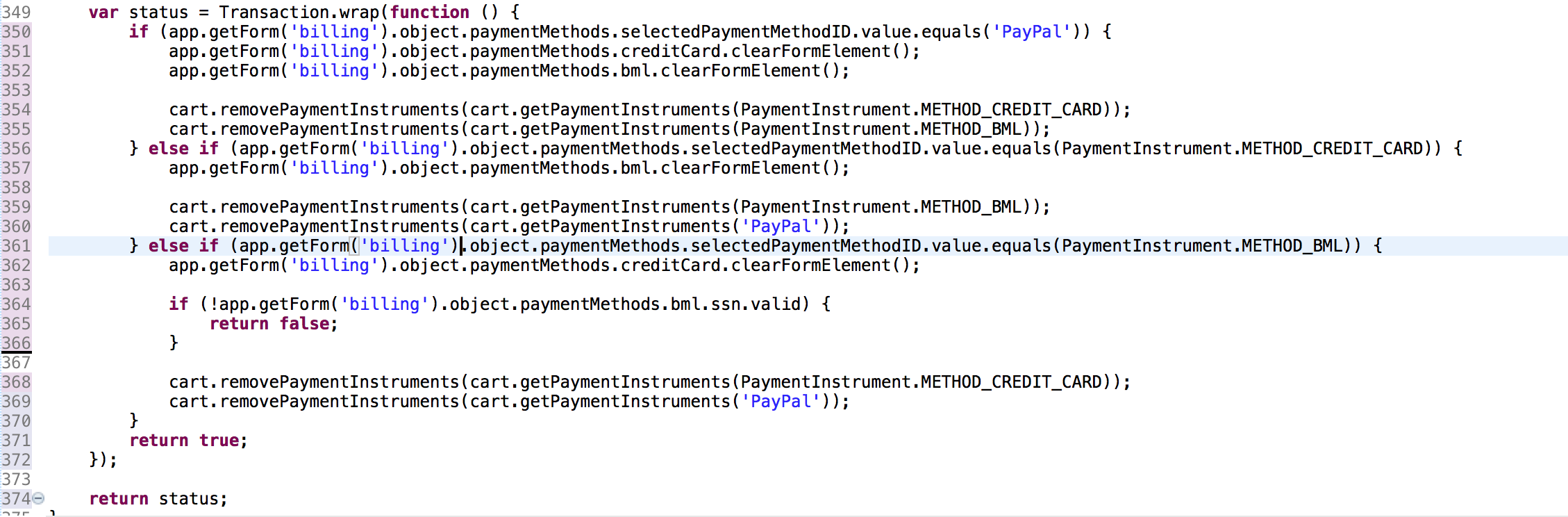
Find the following code (lines 174-179):



Replace the last line with the following code:

var applicablePaymentMethods = require('int\_sezzle/cartridge/controllers/Sezzle').Init(cart, creditCardList.ApplicablePaymentMethods);

Find the following code (lines 351-372):



Replace the above code with the following code:

**var** status = Transaction.wrap(**function** () {

**if** (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals('PayPal')) {

app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();

app.getForm('billing').object.paymentMethods.bml.clearFormElement();

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_CREDIT\_CARD));

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_BML));

cart.removePaymentInstruments(cart.getPaymentInstruments('Sezzle'));

} **else** **if**

(app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(PaymentInstrument.METHOD\_CREDIT\_CARD)) {

app.getForm('billing').object.paymentMethods.bml.clearFormElement();

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_BML));

cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

cart.removePaymentInstruments(cart.getPaymentInstruments('Sezzle'));

}

**else** **if** (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(PaymentInstrument.METHOD\_BML)) {

app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();

**if** (!app.getForm('billing').object.paymentMethods.bml.ssn.valid) {

**return** **false**;

}

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_CREDIT\_CARD));

cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

cart.removePaymentInstruments(cart.getPaymentInstruments('Sezzle'));

}

**else** **if** (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals('Sezzle')) {

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_CREDIT\_CARD));

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_BML));

cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

}

**return** **true**;

});

**return** status;

1. Find and open controller – storefront/cartridge/controllers/COSummary.js. Find “submit” function. Add the below code at the beginning of the function:

**var** redirected = require('int\_sezzle/cartridge/controllers/Sezzle').Redirect();

**if** (redirected) {

**return**;

}

1. Find and open property file – int\_sezzle/cartridge/templates/resources/sezzle.properties. Find property “sezzle.controllers.cartridge”. Set it as name of your controllers cartridge (with script files app.js, guard.js), for example:



## External Interfaces

Commutation between Sezzle service and Salesforce Commerce Cloud Platform is based on Sezzle REST API. All outside traffic from Salesforce Commerce Cloud instance is handled by HTTPS protocol.

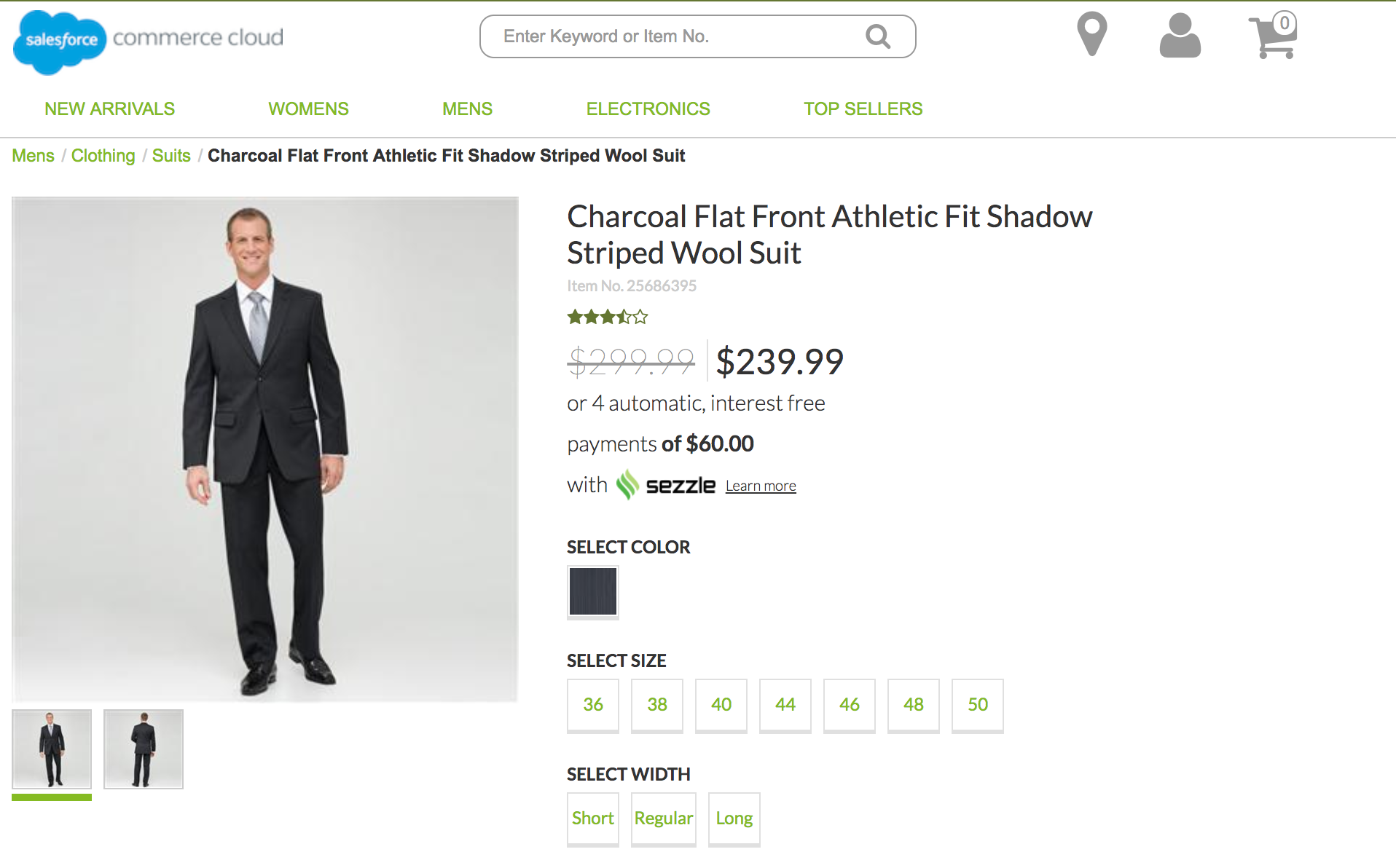
In the Salesforce Commerce Cloud Platform each API call to Sezzle is wrapped into Service that is handling monitoring and logging functionality.

Sezzle integration documentation - <https://docs.sezzle.com/sezzle-pay/>

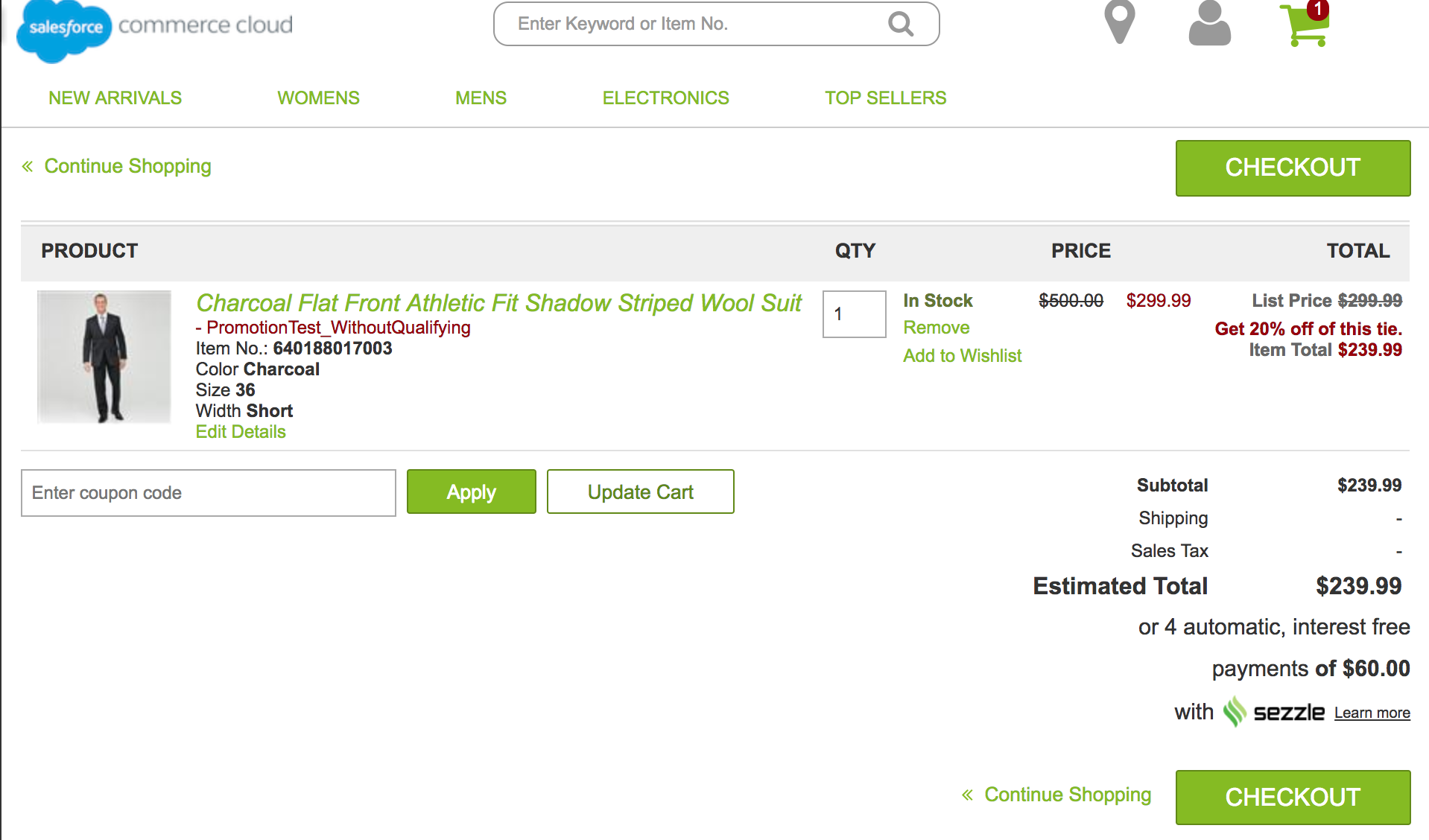
### Sezzle Widget Integration

Merchants can integrate Sezzle’s widgets to display Sezzle’s service on their product and cart pages.

Link for integrating widgets - https://docs.sezzle.com/sezzle-pay/#sezzlejs  
Here are some snapshots after integrating Sezzle’s widgets.

Product Page  


Cart Page



## Firewall Requirements

No specific Firewall requirements.

## Testing

Sezzle has a sandbox that can be used for testing. In Business Manager, navigate to the SiteGenesis Site -> Site Preferences->Custom Preferences. A custom site preference group with the ID SEZZLE\_PAYMENT is available. Please select it and locate ‘Sezzle Mode’. Select ‘Sandbox’ as the mode for testing and ‘Save’ it. Please use “123123”, when prompted for OTPs while using Sandbox mode.

# Operations, Maintenance

## Data Storage

Sezzle cartridge is extending Salesforce Commerce Cloud system objects to store related Sezzle data for request. Following objects that were extended: Order, Product, Category, SitePreference.

## Availability

The Sezzle’s payment gateway guarantees an uptime of almost 100%. However, in case the system does not respond, customers will not be able to use Sezzle to checkout and will have to use a different payment method.

Customers are able to logout at all times from Sezzle’s checkout method, thus enabling a flexible checkout process.

## Support

In case of any availability issues, Sezzle support can be reached via email at merchantsupport@sezzle.com.

# User Guide

## Roles, Responsibilities

Integration of this cartridge will typically be done by a SFCC(Demandware) developer. Sezzle will provide access keys for be used with the API.

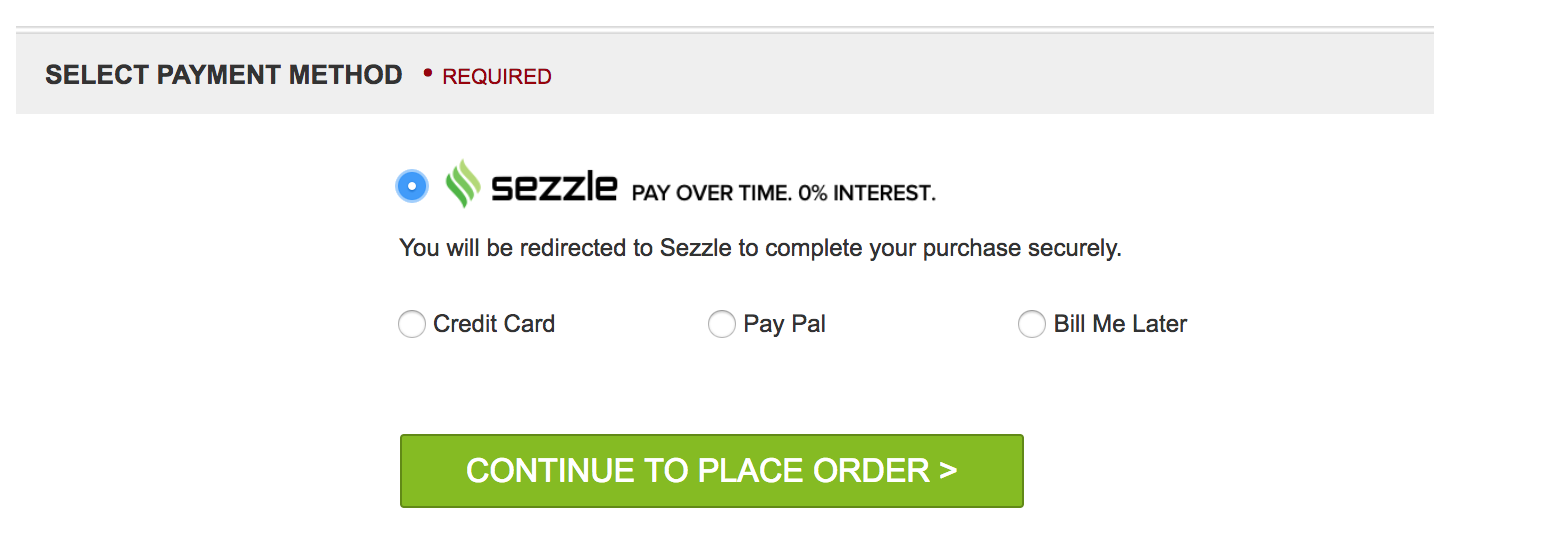
## Business Manager

The cartridge doesn’t extend Business Manager.

## Storefront Functionality

Once enabled, the Sezzle Link integration will add a new functionality to your Salesforce Commerce Cloud store.

The Sezzle payment method will show as an option on the billing page.



# Known Issues

No known Issues.

# Release History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 18.1.0 | 25/09/2018 | Initial release |