Afterpay Integration

Version 20.0.0

Contents

1. Summary

This implementation guide describes how to integrate Afterpay Payment Processor (int_afterpay) in to SFCC(Demandware) reference application SiteGenesis v2.

This cartridge provides the ability to use Afterpay as a Payment Method for during checkout of the SFCC storefront.

To integrate you will need to modify SiteGenesis storefront cartridge.

2. Component Overview

2.1 Functional Overview

Afterpay provides a payment processing gateway. Afterpay's payment gateway connects your storefront to process transactions that allow customers to purchase products.

2.2 Use Cases

Allows checkout using Afterpay payment method.

2.3 Limitations, Constraints

The following Afterpay APIs are supported,

1. Direct Capture Payment
2. Authorize Payment
3. Get Payments
4. Create Refund
5. Create Checkout
6. Get Checkout
7. Get Configuration

The following Afterpay APIs are **not** currently supported,

1. Void Authorisation
2. Capture Payment
3. Update Shipping Courier

2.4 Compatibility

Available since SFCC 16.4

SiteGenesis reference application integration examples and references version v2

2.5 Privacy, Payment

This integration requires access to the following customer data elements:

- Billing Address, Shipping Address, Order Details and Customer Profile.

No credit card details are stored within SFCC using this integration.

3. Integration Guide

Once downloaded, please import the cartridge into your UX studio. Please refer to the following article on how to install UX studio:

- *Getting started >> Installing or updating UX studio in Commerce Cloud's documentation portal.*

Please refer to the following articles if you need assistance with how to connect UX studio with your sandbox and registering (installing) cartridge to sandbox:

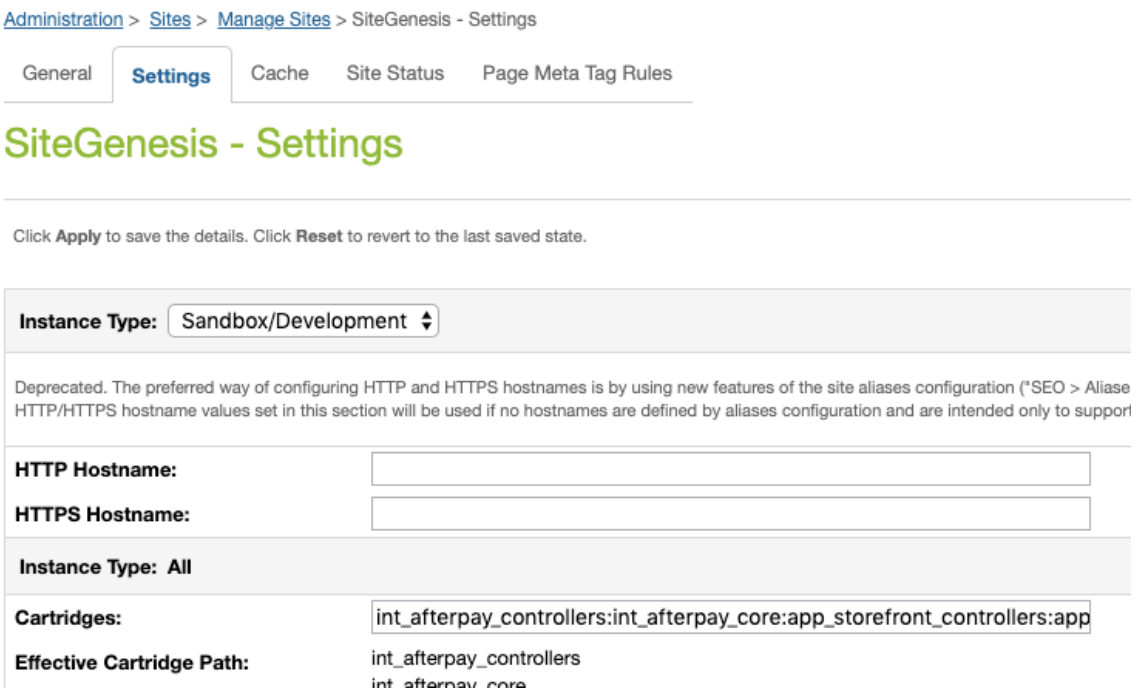
- *Getting started >> connection to a server >> in Commerce Cloud's documentation portal.*
- *Getting started >> Registering your cartridge >> in Commerce Cloud's documentation portal.*

3.1 Activating the Cartridges in Business Manager

Before the Afterpay functionality can become available to SFRA, the cartridge names have to be added to the cartridge path of the Site's settings. In order to do this, follow the below instructions:

1. Log into Business Manager
2. Navigate to Administration > Sites > Manage Sites.
3. Click on the target site name and Click on the Settings tab.

4. In the textbox Cartridges add **“int_afterpay_controllers:int_afterpay_pipelines:int_afterpay_core”** in front of sitegenesis cartridge.



5. Click Apply.

6. To activate the cartridge for the Sandbox/Development/Production instances repeat steps 4 and 5 after selecting the appropriate instance from the Instance Type dropdown menu.

7. Repeat steps 3 to 6 for each site that is to use Afterpay.

8. Go to “Manage the Business Manager site””

9. In the textbox Cartridges add “:bm_afterpay”.

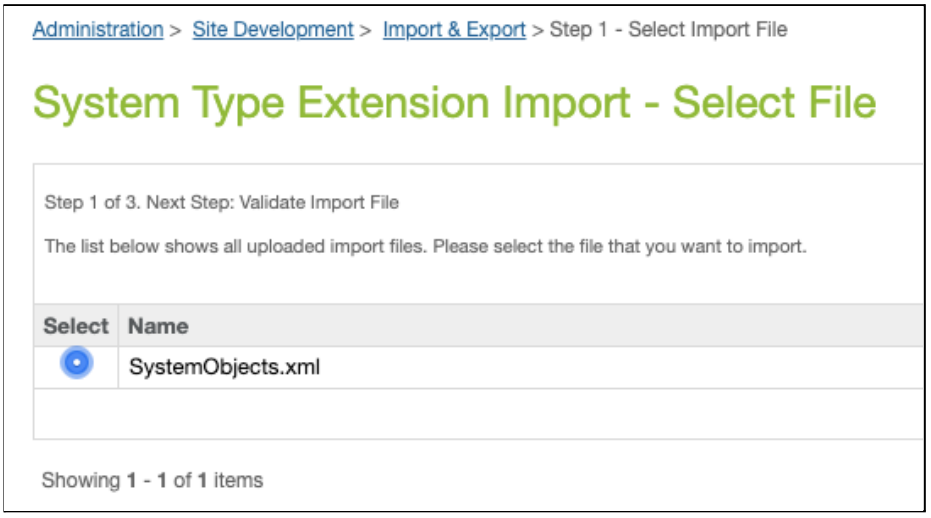
3.3 Configuration

This chapter will describe how to configure the cartridge in Business Manager for SiteGenesis.

3.2.1 Importing Metadata

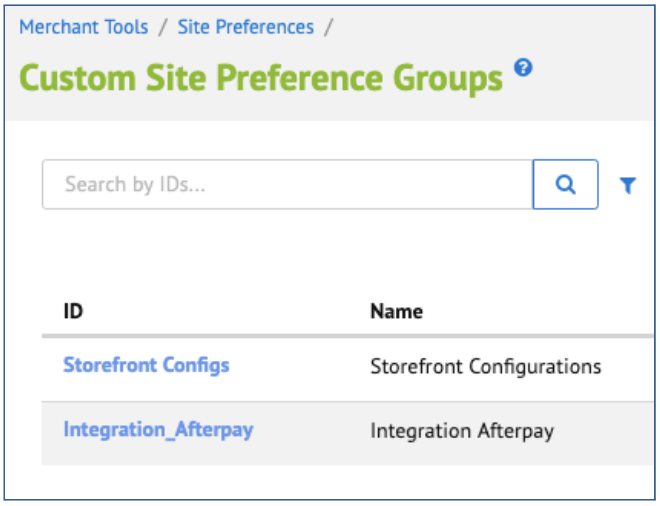
All import files can be found in the metadata folder within cartridge installation.

To import all necessary Afterpay settings, log in to the Business Manager and navigate to Administration > Site Development > Import & Export. Now upload the **SystemObjects.xml** file using the upload button and, finally go back and use the import button to import the file. After a successful import, you will be able to see the Afterpay custom attributes.



3.3.2 Setting Afterpay Custom Site Preferences

In Business Manager, navigate to the SiteGenesis Site > Site Preferences > **Custom Preferences**. A custom site preference group with the ID Integration_Afterpay is available. Please select it and edit the attributes according to your Afterpay account data and the data shown below.



Site Preference	Description	Default Values
apDelayRetry	The field is required to set delay time for the unavailable service.	
CapturePayment	The field is used to set timeout for the Direct Timeout Capture payments.	
UserAgent	The field is used to send the UserAgent details to Afterpay.	
Enable Afterpay	The field determines if the Afterpay is enabled / disabled in the cartridge.	False
Afterpay Display Product Details Page Information	The field determines if the Afterpay messaging is displayed on the Product Details Page.	true
Afterpay Display Product List Page Information	The field determines if the Afterpay messaging is displayed on the Product List Page.	False
Afterpay Learn More URL	The URL that links to more information about Afterpay that users can click to view.	https://www.afterpay.com/purchase-payment-agreement
Afterpay Payment Mode	This field determines if the payment should be Authorised only or if a Direct Capture is to be used.	Merchant Specific – Select the ones to be supported.
Afterpay Service Name	This field is used to identify the service with Afterpay. This only needs to be changed if you are using multiple sites where those sites are using additional Afterpay accounts as the accounts are stored in the service	afterpay.http.defaultendpoint (for AUS site) afterpay.http.defaultendpoint.NZ (for NZsite) afterpay.http.ddefaultendpoint.US (for US site)
Afterpay JavaScript URL	The endpoint where the javascript library is obtained from – Use Sandbox - https://portal.sandbox.afterpay.com/afterpay.js Production - https://portal.afterpay.com/afterpay.js	

Default Controller Cartridge Name	Name of default storefront controller cartridge	app_storefront_controllers
Default Core Cartridge	Name of default storefront core cartridge	
Afterpay Minimum Threshold Amount	Afterpay minimum threshold amount	35
Afterpay Maximum Threshold Amount	Afterpay maximum threshold amount	1000

3.2.3 Setting Afterpay Payment Methods :

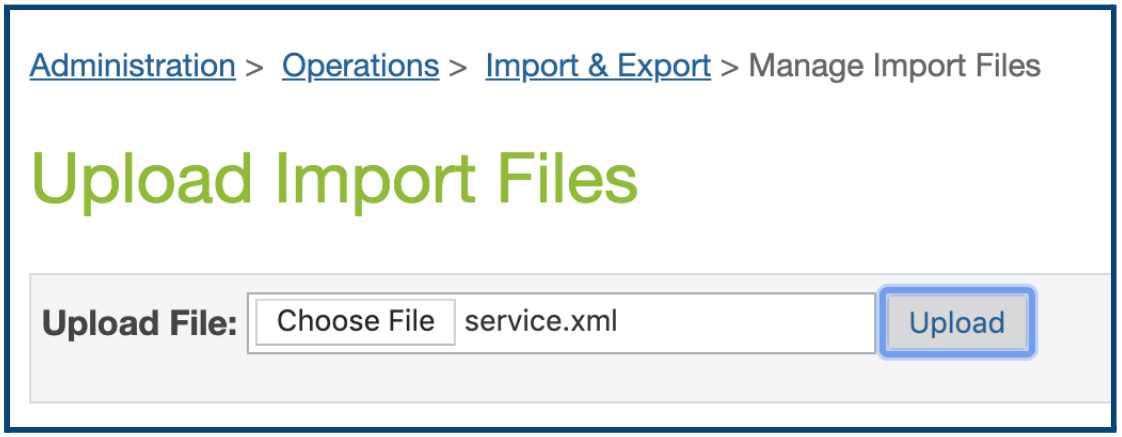
Merchant Tools > Ordering > Import & Export > Upload the "payment-methods.xml" from the metadata folder > Import the xml. A payment method with the name **AFTERPAY_PBI** is available that was imported.

3.2.4 Setting Afterpay Payment Processors:

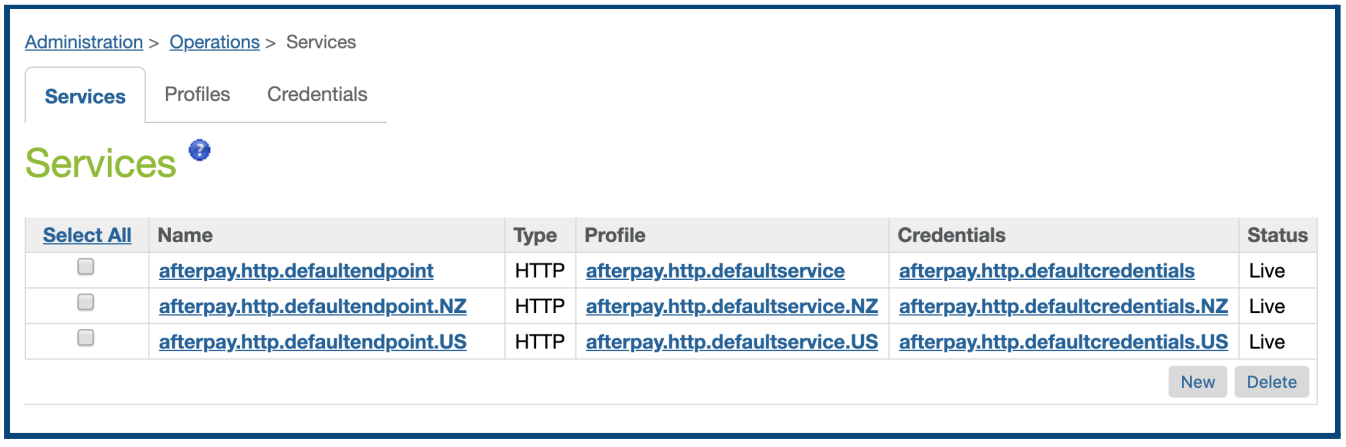
Merchant Tools > Ordering > Payment Processors > Click New
Give a unique ID for the processor: "*AFTERPAY_CREDIT*"
Description: **"***Pay By Installment" and click Apply*

3.3.3 Setting SFCC(Demandware) Service

Administration > Operations > Import & Export > Upload the "service.xml" from the metadata folder > Import the xml .



Three services for US, AUS and NZ locale are available.
Administration > Operations > Services > Credentials > afterpay.http.defaultcredentials > Details > Credentials tab and select required service and enter the account details



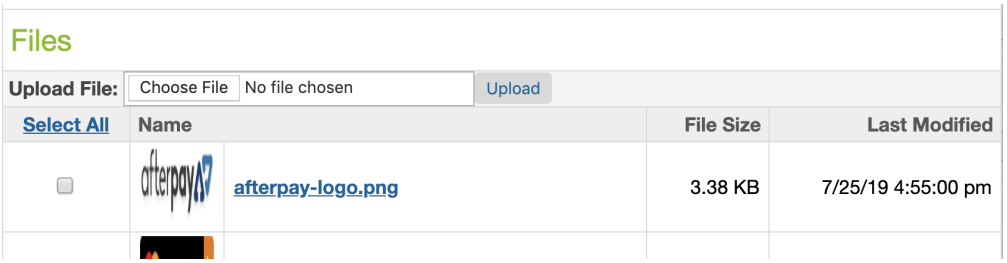
Attribute	Description

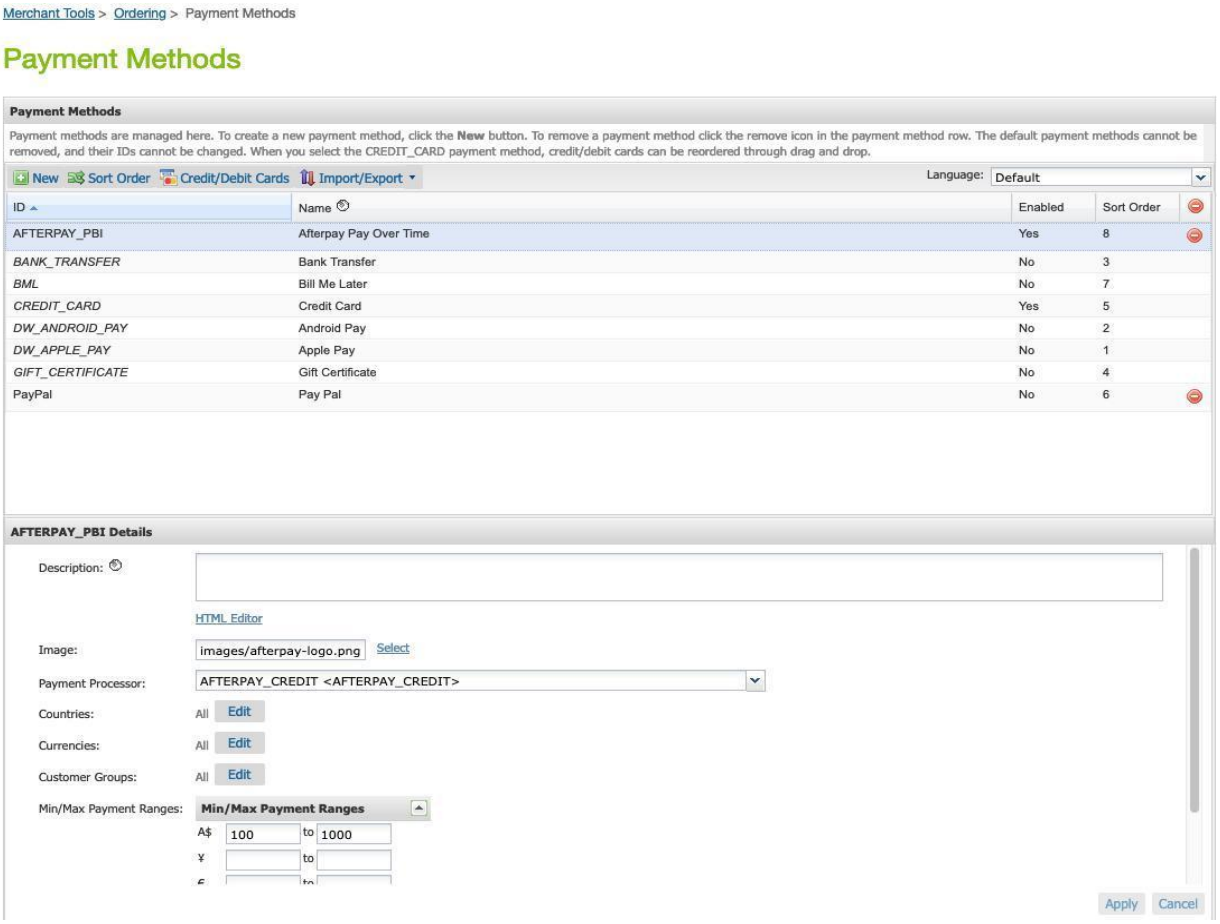
URL	United States : Sandbox https://api.us-sandbox.afterpay.com/v2/ Production https://api.us.afterpay.com/v2/ Australia & New Zealand : Sandbox https://api-sandbox.afterpay.com/v2/ Production https://api.afterpay.com/v2/
User	Enter the Merchant ID provided by Afterpay
Password	Enter the Secret Key provided by Afterpay

3.2.6 Adding Afterpay image to Payment method

In order to display the Afterpay image on the site the image needs to be added to your site.

- Merchant Tools > Ordering > Payment Methods
- Select the payment method with ID AFTERPAY_PBI and locate the image attribute and upload the image included in the cartridge (int_afterpay_sfra/cartridge/static/default/images/**afterpay-logo.png**).




Merchant Tools > Ordering > Payment Methods
Payment Methods
Payment Methods
Payment methods are managed here. To create a new payment method, click the New button. To remove a payment method click the remove icon in the payment method row. The default payment methods cannot be removed, and their IDs cannot be changed. When you select the CREDIT_CARD payment method, credit/debit cards can be reordered through drag and drop.
New
Sort Order
Credit/Debit Cards
Import/Export
Language:
Default
ID
Name
Enabled
Sort Order
AFTERPAY_PBI
Afterpay Pay Over Time
Yes
8
BANK_TRANSFER
Bank Transfer
No
3
BML
Bill Me Later
No
7
CREDIT_CARD
Credit Card
Yes
5
DW_ANDROID_PAY
Android Pay
No
2
DW_APPLE_PAY
Apple Pay
No
1
GIFT_CERTIFICATE
Gift Certificate
No
4
PayPal
Pay Pal
No
6
AFTERPAY_PBI Details
Description:
HTML Editor
Image:
images/afterpay-logo.png
Select
Payment Processor:
AFTERPAY_CREDIT <AFTERPAY_CREDIT>
Countries:
All
Edit
Currencies:
All
Edit
Customer Groups:
All
Edit
Min/Max Payment Ranges:
Min/Max Payment Ranges
A$
100
to
1000
¥
to
Apply
Cancel

3.2.7 Modifying Content

There is a content asset (**afterpay-checkout-pbi**) and content slot (**afterpay-checkout-pbi**) those are used in the checkout and are displayed when selecting Afterpay payment method. The content asset (**afterpay-checkout-pbi**) can be modified as required to suit your site.

To add the content asset into your site :

Merchant Tools > Content > Import & Export > Upload the contents assets xml located in (cartridge/metadata/ContentAssets.xml) > Import the xml

To add the content slot into your site :

Merchant Tools > Online Marketing > Import & Export > Upload the contents slots xml located in (cartridge/metadata/slots.xml) > Import the xml

There are a few items added to the resource file in this cartridge.

Property	**Description**
afterpay.pbi.text	The text used on product and tiles to show Afterpay payments information (if enabled)
afterpay.api.pending	The message displayed if payment is pending (currently not supported by Afterpay)
afterpay.api.error	The message displayed if there is an error

4. Custom Code

In order to make this Afterpay cartridge work, a few modifications to SiteGenesis are required

In *app_storefront_controllers/cartridge/controllers/**COBilling.js*** ,add the below mentioned code snippet:

In the method **resetPaymentForms()**, add the command for the three conditions:

```
cart.removePaymentInstruments(cart.getPaymentInstruments('AFTERPAY_PBI'));
```

```
function resetPaymentForms() {
 var cart = app.getModel('Cart').get();

 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)
     );
     // remove Afterpay PaymentInstrument

     cart.removePaymentInstruments(
       cart.getPaymentInstruments('AFTERPAY_P BI')
     );
   } 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')); // remove
Afterpay PaymentInstrument

    cart.removePaymentInstruments(
      cart.getPaymentInstruments('AFTERPAY_P BI')
    );
  } 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')); // remove
Afterpay PaymentInstrument

    cart.removePaymentInstruments(
      cart.getPaymentInstruments('AFTERPAY_P BI')
    );
  }

  return true;
 });

 return status;
}
```

In *app_storefront_controllers/cartridge/controllers/**COSummary.js***, add the below mentioned code snippet:

Replace the code block

```
app.getView(viewContext).render('checkout/summary/summary');
```

With

```
require("int_afterpay_core/cartridge/scripts/util/AfterpayUtilities.js").disable
SummaryForAfterpay(cart, viewContext);
```

In *app_storefront_controllers/cartridge/controllers/**COPlaceOrder.js*** ,add the below mentioned code snippet:

Replace the code block :

```
if (authorizationResult.not_supported || authorizationResult.error) {
 return {
   error: true,
 };
}
```

With :

```
if (authorizationResult.not_supported || authorizationResult.error) {
 return {
   authorizeError: authorizationResult.authorizationResponse,
   error: true,
 };
}
```

Replace the code block :

```
if (handlePaymentsResult.error) {
 return Transaction.wrap(function() {
   OrderMgr.failOrder(order);
   return {
     error: true,
     PlaceOrderError: new Status(Status.ERROR, 'confirm.error.technical'),
   };
 });
}
```

With:

```
if (handlePaymentsResult.error) {
 return Transaction.wrap(function() {
   OrderMgr.failOrder(order);
   return {
     error: true,
     afterpayOrderAuthorizeError: handlePaymentsResult.authorizeError,
     PlaceOrderError: new Status(Status.ERROR, 'confirm.error.technical'),
   };
 });
}
```

In ***app_storefront_controllers/cartridge/controllers/Product.js*** ,add the below mentioned code snippet:

Add the below mentioned code snippet in the method **show()** , inside the condition -> if (product.isVisible()) { :

```
var sitePreferences =
require('int_afterpay_core/cartridge/scripts/util/AfterpayUtilities.js').getSitePreferen
cesUtilities();
var afterpayEnable = sitePreferences.isAfterpayEnabled();

if (afterpayEnable) {

require('int_afterpay_core/cartridge/scripts/util/AfterpayCallThreshold.js').SetThreshol
d();
}
```

In ***app_storefront_controllers/cartridge/scripts/views/CartView.js*** :

Add the below mentioned code snippet in the method **CartView()** ,after the transaction wrap for cart.calculate :

```
var sitePreferences =
require('int_afterpay_core/cartridge/scripts/util/AfterpayUtilities.js').getSitePreferen
cesUtilities();
var afterpayEnable = sitePreferences.isAfterpayEnabled();

if (afterpayEnable) {

require('int_afterpay_core/cartridge/scripts/util/AfterpayCallThreshold.js').SetThreshol
d();
}
```

In *app_storefront_controllers/cartridge/controllers/**Search.js*** :

Add the below mentioned code snippet in the method **Show()** , **inside the elseif**

(productSearchModel.count > 0) { :

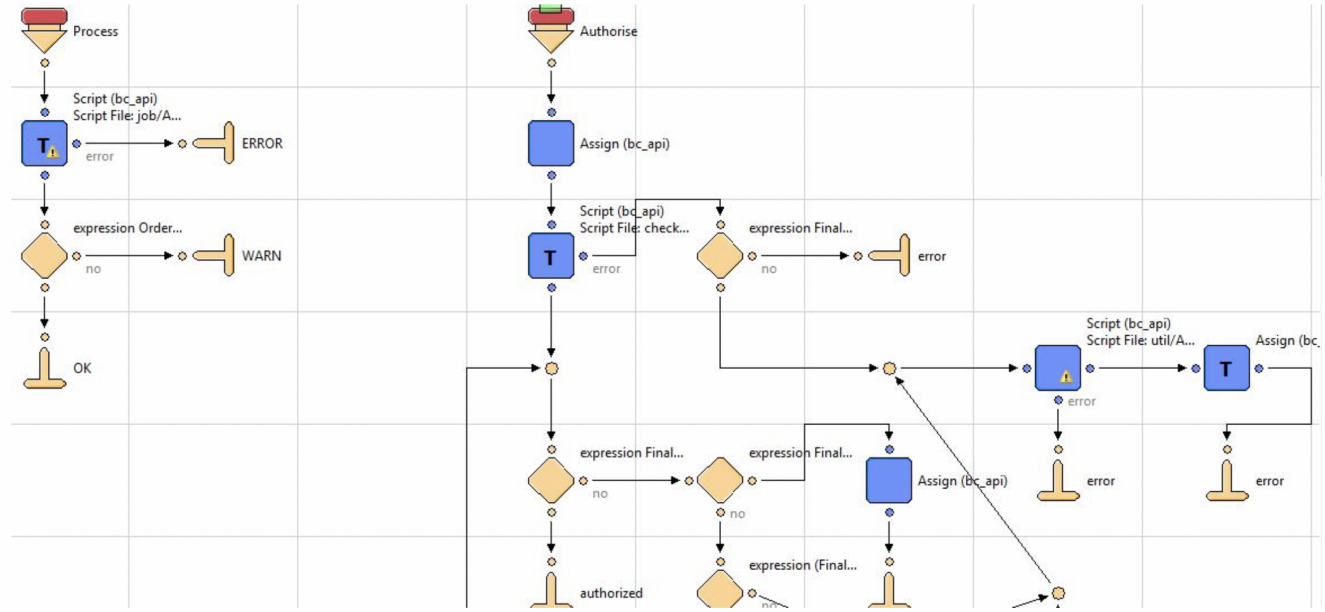
```
var sitePreferences =
require('int_afterpay_core/cartridge/scripts/util/AfterpayUtilities.js').getSitePreferen
cesUtilities();
var afterpayEnable = sitePreferences.isAfterpayEnabled();

if (afterpayEnable) {

require('int_afterpay_core/cartridge/scripts/util/AfterpayCallThreshold.js').SetThreshol
d();
}
```

Pipeline Changes

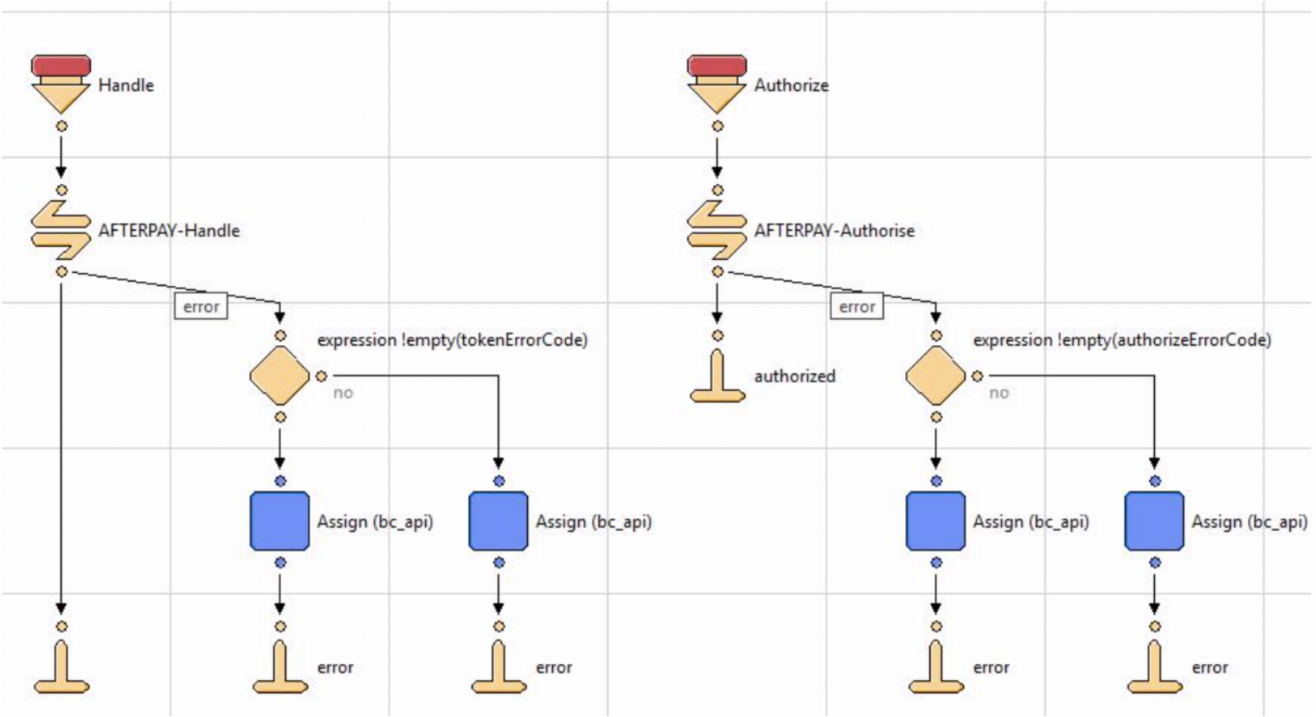
CARTRIDGE STRUCTURE

Visualizations of the pipelines:

Visualization of the pipeline AFTERPAY_CREDIT



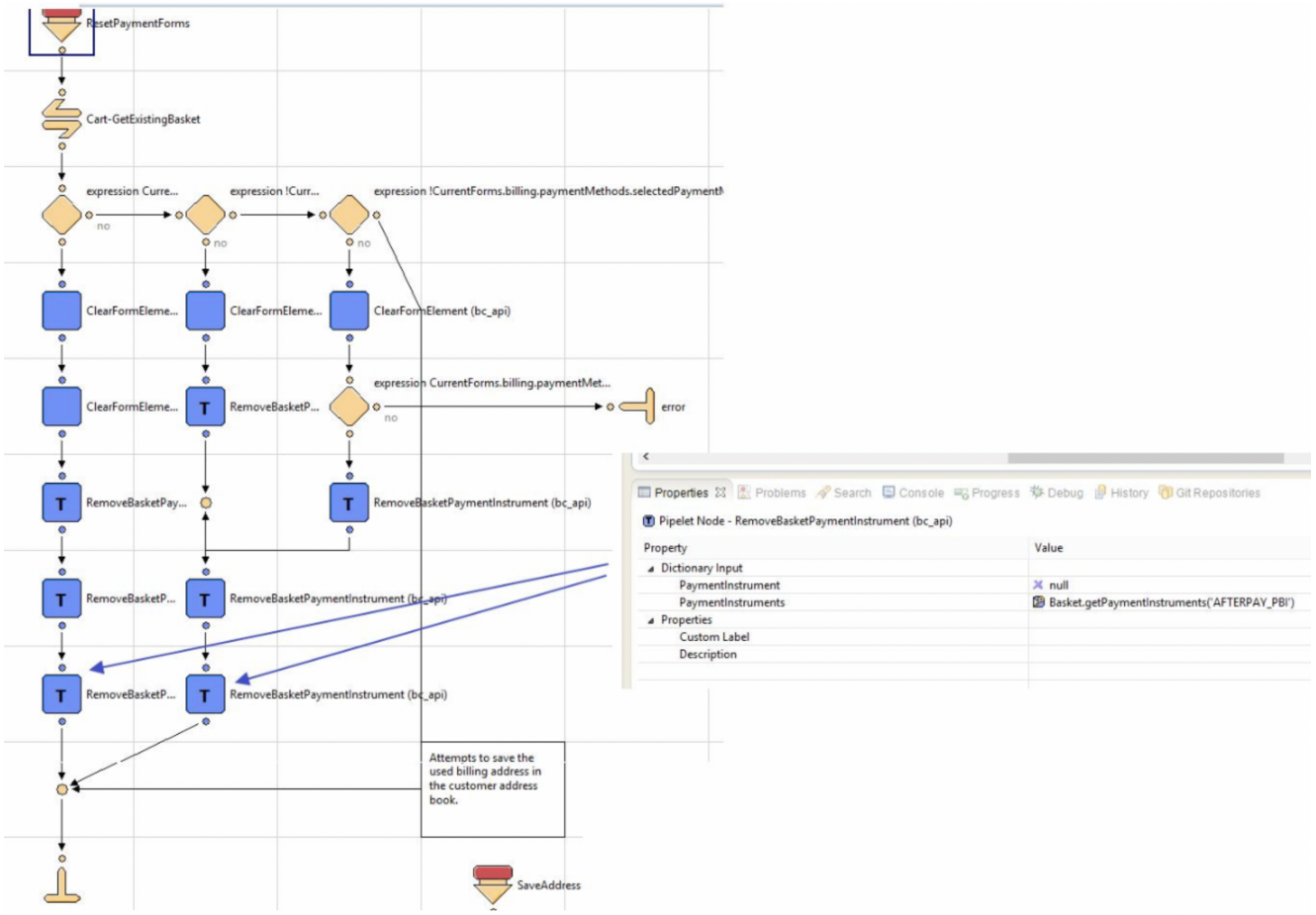
Visualization of the pipeline AFTERPAY



In pipeline **COBilling-ResetPaymentForms**

Add two pipelets '**com.demandware.pipelet.basket.RemoveBasketPaymentInstrument**' at the end of the pipeline:

In Dictionary Input 'PaymentInstruments', set it with value:
Basket.getPaymentInstruments('AFTERPAY_PBI')



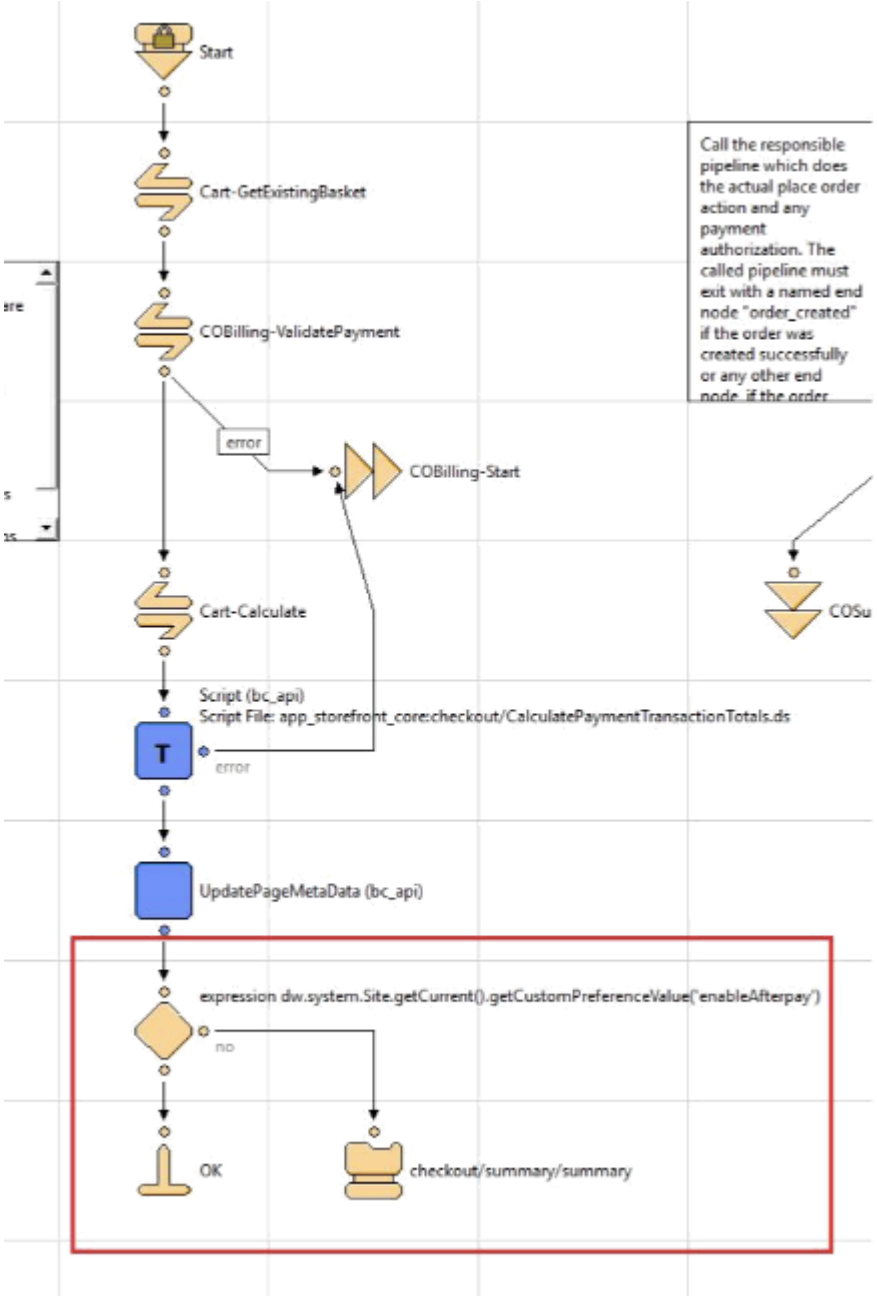
In pipeline **COSummary-Start,**

add condition to skip summary page based on the payment method

Set dictionary inputs as :

Conditional node :

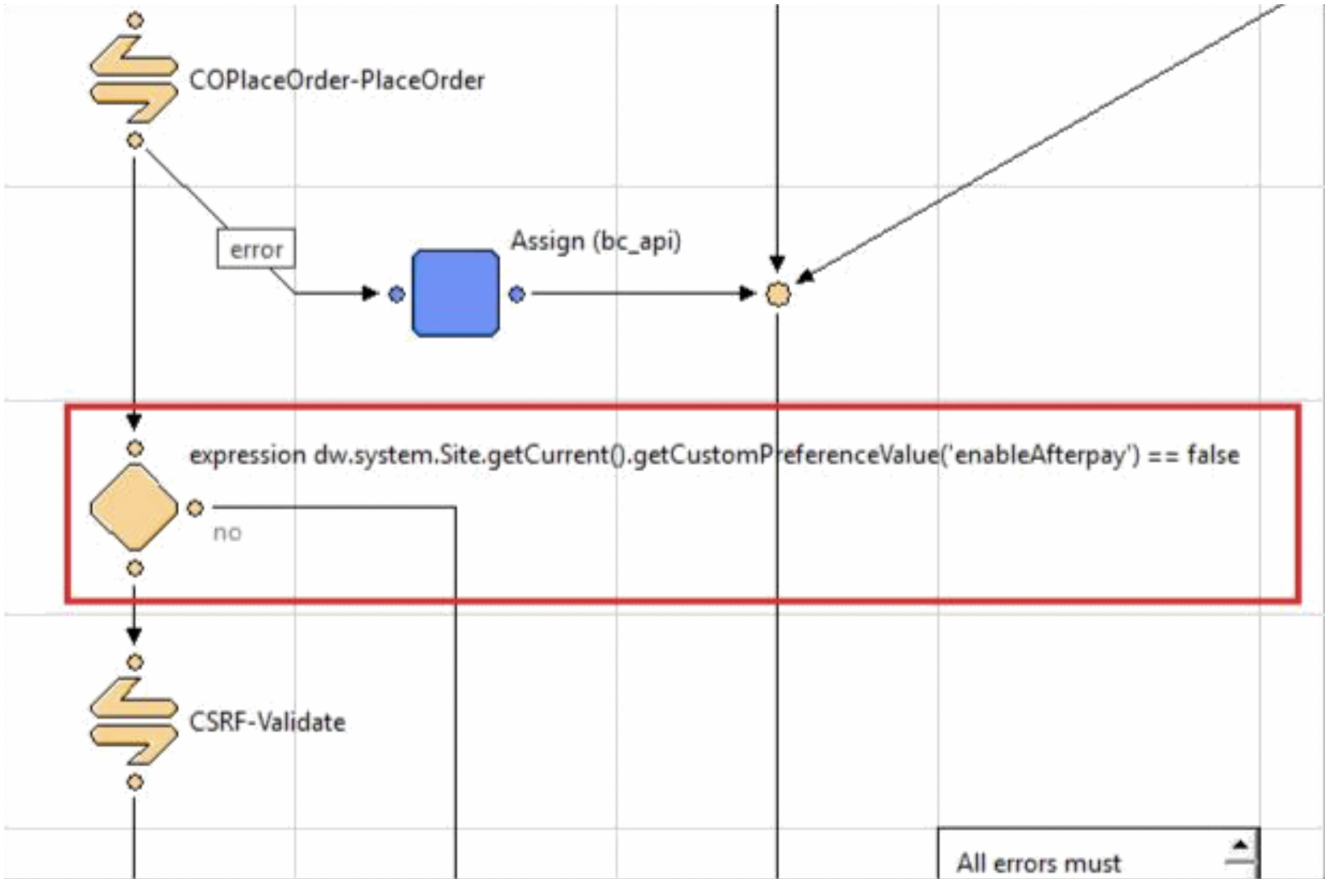
dw.system.Site.getCurrent().getCustomPreferenceValue('enableAfterpay')



In COPlaceOrder-Start, add the following node:

Conditional node :

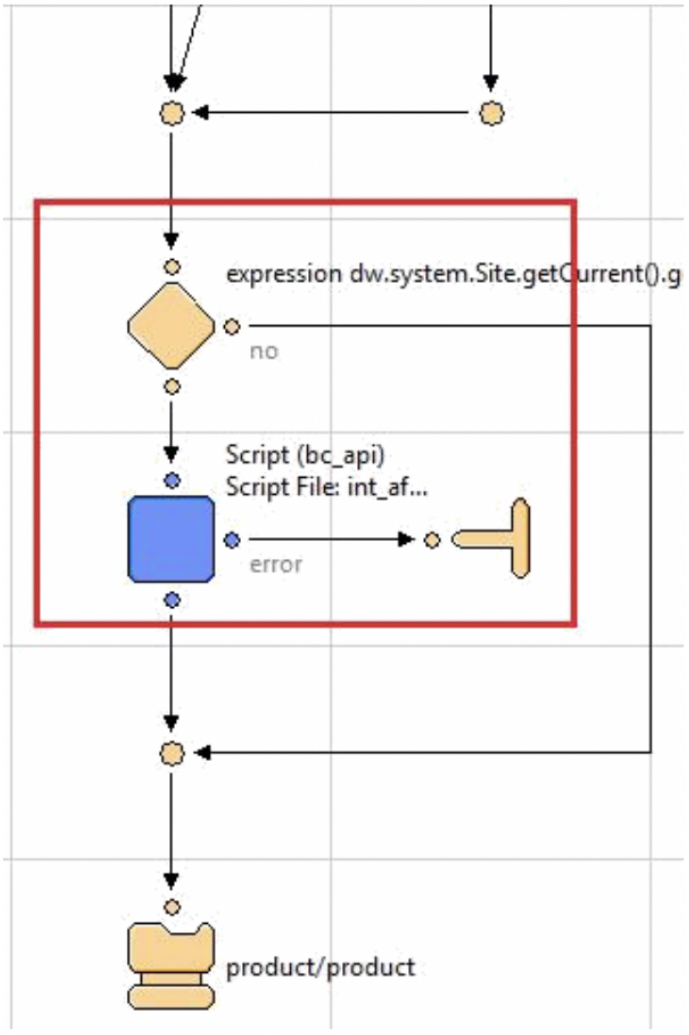
dw.system.Site.getCurrent().getCustomPreferenceValue('enableAfterpay') == false



In Product-Show :

add the ScriptFile "*int_afterpay_core:util/**AfterpayCallThreshold.js***" when afterpay is enabled in the configuration

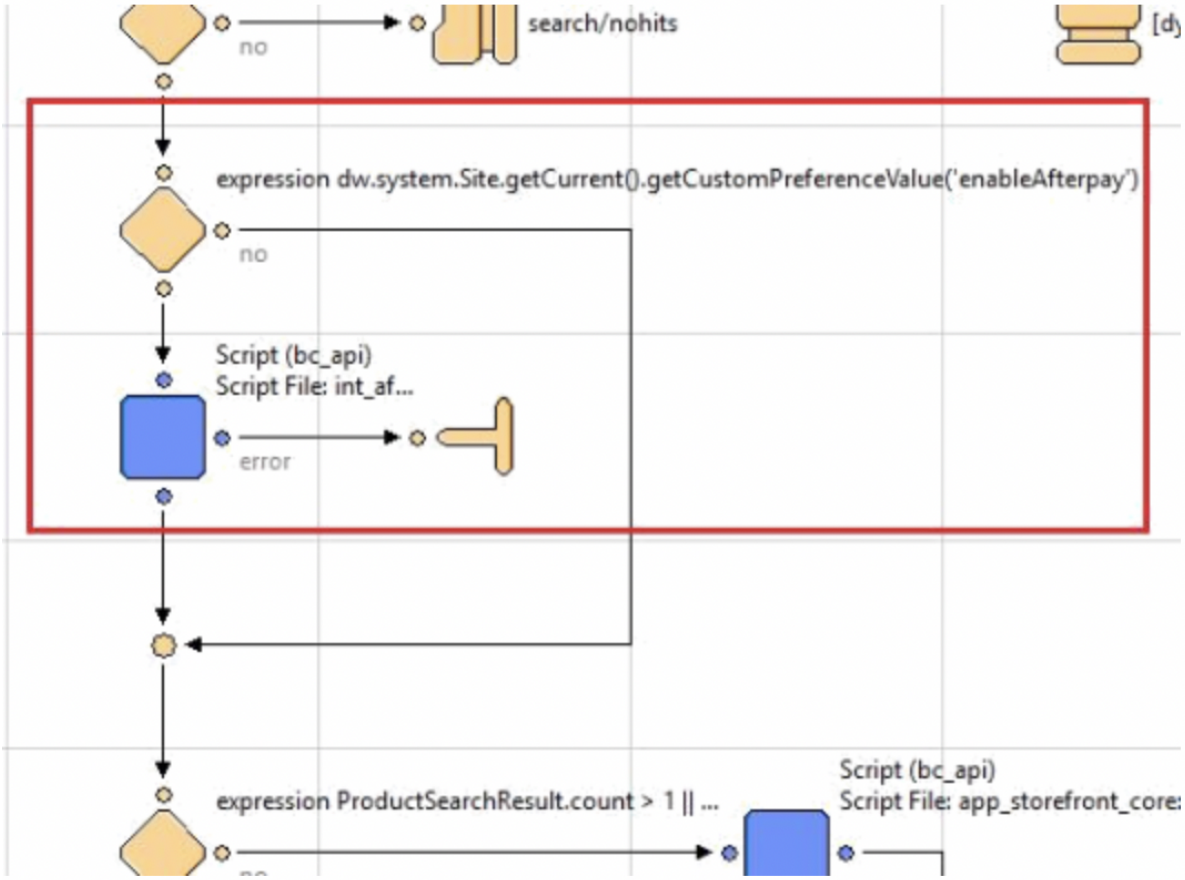
Conditional node :
dw.system.Site.getCurrent().getCustomPreferenceValue('enableAfterpay')



In Search-Show :

add the ScriptFile "*int_afterpay_core:util/**AfterpayCallThreshold.js***" when afterpay is enabled in the configuration

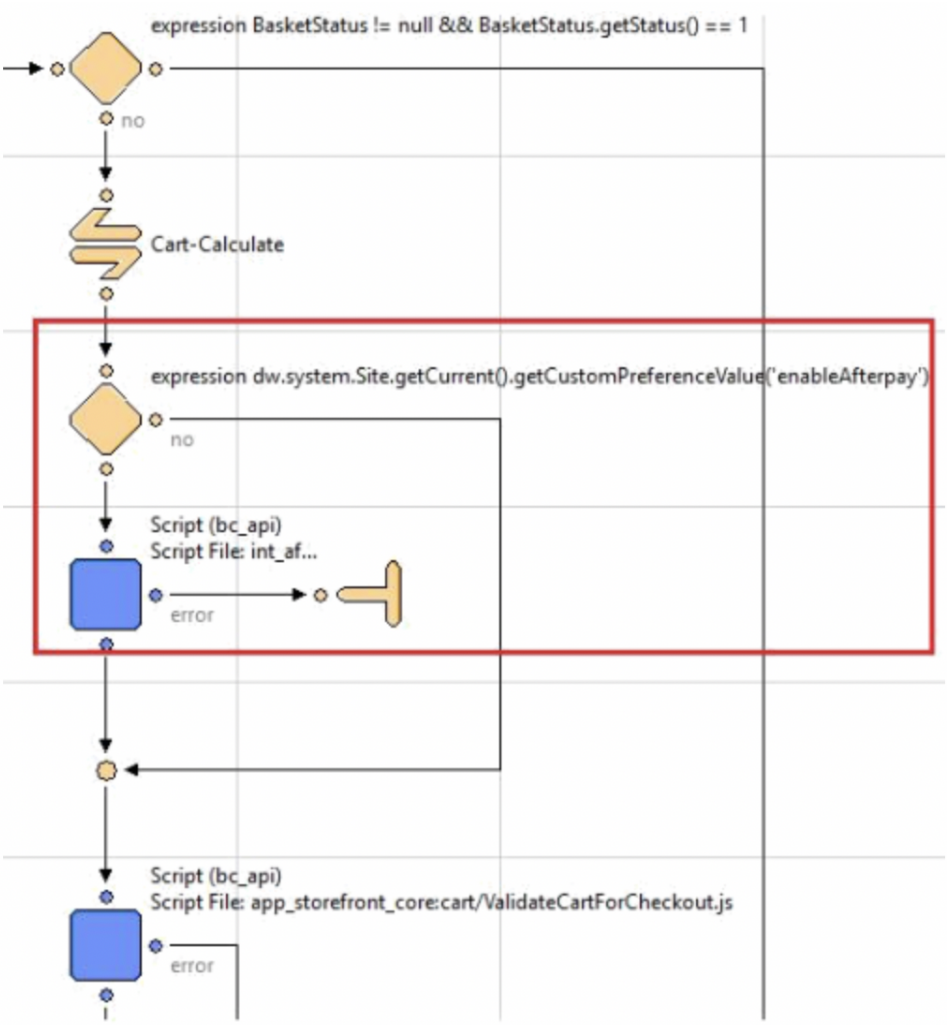
Conditional node :
dw.system.Site.getCurrent().getCustomPreferenceValue('enableAfterpay')



In Cart – PrepareView

add the ScriptFile "*int_afterpay_core:util/**AfterpayCallThreshold.js***" when afterpay is enabled in the configuration

Conditional node :
dw.system.Site.getCurrent().getCustomPreferenceValue('enableAfterpay')



3.4.1 Templates

Checkout/billing/billing.isml :

Locate the <iscomment> ++++++++++++++++++++billing address</iscomment> and add the below mentioned code after the comment:

```
<isset name="afterpayError"
   value="${!empty(pdict.AfterpayApiError) ? pdict.AfterpayApiError :
(!empty(request.httpParameterMap.get('afterpay'))
          .stringValue ? request.httpParameterMap.get('afterpay').stringValue : null)}"
   scope="page" />

<isif condition="${!empty(afterpayError)}">
   <div class="error-form">
       <isprint value="${afterpayError}" encoding="off" />
   </div>
</isif>
```

Checkout/billing/paymentmethods.isml

After the line : <isinclude template="util/modules"/> , add the below mentioned code :

```
<isscript>

var sitePreferences =
require("int_afterpay_core/cartridge/scripts/util/AfterpayUtilities.js").getSitePreferen
cesUtilities();
   var afterpayEnable = sitePreferences.isAfterpayEnabled();
   var basketObject = dw.order.BasketMgr.getCurrentBasket();
   var orderGrandTotal = basketObject.totalGrossPrice;
   var apMessageService =
require("int_afterpay_core/cartridge/scripts/product/AfterpayDisplayProductMessage");
   var thresholdResponse = apMessageService.getThresholdRange(orderGrandTotal);
   var disableAfterpayPaymentMethod = false;

   if(thresholdResponse && (thresholdResponse.belowThreshold ||
thresholdResponse.aboveThreshold)){
   disableAfterpayPaymentMethod = true;
   }
</isscript>
```

After the line :
<isif condition="${paymentMethodType.value.equals(dw.order.PaymentInstrument.METHOD_GIFT_CERTIFICATE)}"><iscontinue/></isif>, add the below mentioned code :

```
<iscomment>Ignore the payment method AFTERPAY_PBI if Afterpay payment method is disabled in configuration</iscomment>

<isif
condition="${paymentMethodType.value.equals('AFTERPAY_PBI') && afterpayEnable ==
false}"><iscontinue/></isif>

<isif condition="${paymentMethodType.value.equals('AFTERPAY_PBI') &&
disableAfterpayPaymentMethod}">
<isset name="disableAfterpay" value="${true}" scope="page" />
       <iselse>
<isset name="disableAfterpay" value="${false}" scope="page" />
</isif>
```

Locate the following div:

```
<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="${paymentMethodType.value ==
pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlValue}">checked="c
hecked"</isif> />
         </div>
<label for="is-${radioID}"><isprint
value="${Resource.msg(paymentMethodType.label,'forms',null)}"/></label>
</div>
```

And replace with:

```
<isif condition="${!disableAfterpay}">
 <div class="form-row label-inline">
      <isset name="radioID" value="${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="c
hecked"</isif> />
      </div>
      <label for="is-${radioID}">
      <isset name="paymentObject" value="${paymentMethodType.object}" scope="page"/>
      <isif condition="${empty(paymentObject.image)}">
         <isprint value="${Resource.msg(paymentMethodType.label,'forms',null)}"/>
      <iselse>
        <img src="${paymentObject.image.URL}" class="payment-image
${radioID.toLowerCase()}-image" alt="${Resource.msg('billing.logo','afterpay',null)}"/>
      </isif>
</label>
</div>
</isif>
```

Locate the following on line 138

```
 <iscomment>
 Custom processor
 ---------------------------------------------------------------
</iscomment>
```

Add this after:

```
<iscomment>
 Afterpay Pay Over Time
 ---------------------------------------------------------------
</iscomment>
```

```
<isif condition="${afterpayEnable == true}">
 <div class="payment-method <isif condition="${!empty(pdict.selectedPaymentID) &&
pdict.selectedPaymentID=='AFTERPAY_PBI'}">payment-method-expanded</isif>"
data-method="AFTERPAY_PBI">
 <isslot id="afterpay-checkout-pbi" description="Afterpay checkout PBI content"
 context="global" />
 </div>
</isif>
```

product/components/pricing.isml

Locate this code

```
<isinclude template="product/components/standardprice"/>
```

And add the below mentioned code snippet after this line

```
<isinclude template="util/modules">
```

Locate this code on line 160 at the very bottom of file:

```
 <isif condition="${!empty(pdict.OrgProduct)}">
 <iscomment>Restore current product instance</iscomment>

 <isset name="Product" value="${pdict.OrgProduct}" scope="pdict"/>
 <isset name="OrgProduct" value="${null}" scope="pdict"/>
</isif>
```

Add this before:

```
 <isscript>
 var sitePreferences =

require("int_afterpay_core/cartridge/scripts/util/AfterpayUtilities.js").getSitePreferen
cesUtiliti
 es();
 var afterpayEnable = sitePreferences.isAfterpayEnabled();

</isscript>
<isif condition="${afterpayEnable == true}">
 <isafterpaythreshold totalprice="${(SalesPrice ? SalesPrice : StandardPrice)}"
 classname="pdp-afterpay-message" />
</isif>
```

product/producttile.ismI

Locate this code at the top of file:

```
<iscontent type="text/html" charset="UTF-8" compact="true"/>
 <isif condition="${(pdict.cache != null) ? pdict.cache : true}">
 <iscache type="relative" hour="24" varyby="price_promotion"/>
</isif>
```

Add this after:

```
<isinclude template="util/modules"></isinclude>
```

Locate this code on line 239:

```
prices.push(price);
if (extraPrice) {prices.push(extraPrice);}
```

And replace with:

```
if (price && price.value) {prices.push(price);}
if (extraPrice && extraPrice.value) {prices.push(extraPrice);}
```

Locate this comment on line 248:

```
<iscomment>Promotion</iscomment>
```

Add this before:

```
<iscomment>Afterpay message</iscomment>
<isscript>
 var sitePreferences =
require("int_afterpay_core/cartridge/scripts/util/AfterpayUtilities.js").getSitePreferen
cesUtilities();
 var afterpayEnable = sitePreferences.isAfterpayEnabled();
</isscript>
<isif condition="${afterpayEnable == true}">
```

```
<isafterpaythreshold totalprice="${(prices[prices.length-1].value)}"
classname="plp-afterpay-message" />
</isif>
```

checkout/cart/cart.isml

Locate the div:

```
<div class="cart-actions"></div>
```

And add the below mentioned code before the div:

```
<isscript>
 var sitePreferences =

require("int_afterpay_core/cartridge/scripts/util/AfterpayUtilities.js").getSite
PreferencesUtiliti
 es();
 var afterpayEnable = sitePreferences.isAfterpayEnabled(); </isscript>
 <isif condition="${afterpayEnable == true}">
   <div class="learn-more">
   <isif condition="${pdict.Basket.totalGrossPrice.available}"> <isset
name="orderTotalValue" value="${pdict.Basket.totalGrossPrice}" scope="page"/>
<iselse/>
   <isset name="orderTotalValue"
value="${pdict.Basket.getAdjustedMerchandizeTotalPrice(true).add(pdict.Basket.gi
ftCertific ateTotalPrice)}" scope="page"/>
 </isif>
 <iscomment>Afterpay message</iscomment>
 <isafterpaythreshold totalprice="${orderTotalValue}"
 classname="cart-afterpay-message" />
 </div>
</isif>
```

util/modules.isml

Add this:

```
<iscomment>
 Render the Afterpay modules
</iscomment>
<isinclude template="util/afterpay/modules" />
```

app_storefront_core/cartridge/scripts/util/Resource.ds

Find the line:

```
ResourceHelper.getPreferences = function(pageContext) {
```

Add this code inside the method :

```
AFTERPAY_PORTAL : Site.getCurrent().getCustomPreferenceValue('apJavascriptURL')
```

3.4.2 Javascript

app_storefront_core/cartridge/js/app.js

Add the below mentioned code snippet after the method:

```
$('.user-account').on('click', function (e) {
 $(document).on('click', '.afterpay-link', function (e) { e.preventDefault();

 dialog.open({
   url: $(e.target).attr('href')
 });
});
```

app_storefront_core/cartridge/js/ bonus-products-view.js

```
function createOrder() {
 var url = util.ajaxUrl(Urls.createAfterpayOrder);

 ajax.getJson({
   url: url,
   callback: function (data) {
   if (data) {
     checkAuthorize(data);
   }
   }
 });
}

function checkAuthorize(data) {
 if (window.AfterPay) {
   AfterPay.init();
   AfterPay.display({token: data.token});
 } else {
   setTimeout(checkAuthorize.bind(null, data), 3000);
 }
}
 function initScript() {
 var script = document.createElement('script');

 script.src = SitePreferences.AFTERPAY_PORTAL;
 script.async = true;

 document.body.appendChild(script);
}
 function initSubmitOrderEvent() {
 var $form = $('.checkout-billing');

 var $paymentMethods = $form.find('input[name$="_selectedPaymentMethodID"]'); var
$btnContinue = $form.find('button[name$="_billing_save"]');

 $btnContinue.on('click', function(e) {

   if ($paymentMethods.filter(':checked').val() === 'AFTERPAY_PBI') {
e.preventDefault();
```

```
    createOrder();
    return false;
  }
});
var $declined = $('.ap-declined'),

$form = $('.submit-order');

if ($declined.length > 0) {
  $form.on('submit', function (e) {
    e.preventDefault();
    window.location = window.Urls.billing;
  });
} else {
  $form.off('submit').on('submit', function (e) {
    return true;
  });
}
}
exports.init = function () {
 initScript();
  initEvent();
};
module.exports.initSubmitOrderEvent = function () {

 initSubmitOrderEvent();
};
```

3.4.3 CSS

app_storefront_core/cartridge/scss/default/style.scss

Add the code:

```
@import "afterpay";
```

Note: Copy the file '**_afterpay.scss** from the reference folder in the root folder of the cartridge to the path: app_storefront_core/cartridge/scss/default/_afterpay.scss

3.5 Testing

Afterpay 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 Integration_Afterpay is available. Please select it and locate the Afterpay Javascript URL preference and edit the attributes to use the sandbox URL (https://portal.sandbox.afterpay.com/afterpay.js). The service credentials also need to be updated with an Account ID and Key provided by Afterpay. The Service URL needs to be set to the relevant region and environment. See 3.3.3

4. Operations, Maintenance

4.1 Data Storage

Some additional custom attributes are stored against system objects. The following table lists these attributes and their uses, other than Site Preferences (which are listed under section 3.3.2).

Object	Attribute	Type	Description
OrganizationPreferences	flowIPAddresses (flowIPAddresses)	Text	IP addresses
PaymentTransaction	Afterpay Refund ID (apRefundID)	String	Refund ID
Order	Afterpay Order ID (apPaymentD)	Boolean	Flags the order as an Afterpay order
PaymentTransaction	Afterpay Token (apToken)	String	The token used to place order in Afterpay
PaymentTransaction	Afterpay Initial Status (apInitialStatus)	String	The pre-approval status first returned
PaymentTransaction	Afterpay Payment Mode (apPaymentMode)	String	The payment mode used
PaymentTransaction	Afterpay Direct Payment (apDirectPayment Status)	String	The status of Direct Payment, if this payment mode is used
PaymentTransaction	Afterpay Authorise Status (apAuthoriseStatus)	String	The status of Authorise, if this payment mode is used
PaymentTransaction	Afterpay Order ID (apPaymentD)	String	The Payment ID generated by Afterpay
PaymentTransaction	Auth Expiry (apAuthExpiry)	String	The time the Afterpay Order Auth Expires
PaymentTransaction	Payment Event ID (apPaymentEventID)	String	The unique ID for the Afterpay Auth event

4.2 Availability

If the Afterpay service is unavailable the user will not be able to checkout using this payment method.
The service availability can be tracked in SFCC(Demandware) using the Service Status.

4.3 Support

For support please contact your Afterpay account contact

5. User Guide

5.1 Roles, Responsibilities

Integration of this cartridge will typically be done by a SFCC(Demandware) developer.

Afterpay will provide access keys for be used with the API.

5.2 Business Manager - Optional

he following can to be configured in the Business manager.

1. Upload the cartridge bm_afterpay into server.

2 Go to Demandware > Administration > Manage Sites > Business Manager > Settings

> Cartridges input field add cartridge name “bm_afterpay“.

3 Administration > Organization > Roles > Administrator > Business manager modules

> Select context(select your site) > enable the Afterpay transactions management .

4 Merchant tools > Afterpay > transactions Management > Click on the “APPROVED” order ID for which you want to refund the transaction.

Merchant Tools > After Pay > Transactions Management

Transactions Management

Order Search

Order ID: Find

Order ID	Order Date	Email	Total	Status
00009501	2019-01-02 01:34	tara.shipton@touchcorp.com	AUD 164.84	REFUNDED
00009401	2018-10-02 07:17	steven.g@touchcorp.com	AUD 146.97	REFUNDED
00009303	2018-09-20 07:50	steven.g@touchcorp.com	AUD 146.97	APPROVED
00009301	2018-09-20 07:47	steven.g@touchcorp.com	AUD 251.99	APPROVED
00009201	2018-09-07 04:15	skasi@ideatarmac.com	AUD 314.99	APPROVED
00002502	2018-07-13 08:02	tara.shipton@touchcorp.com	AUD 251.95	REFUNDED
00002402	2018-07-09 07:06	rebecca.ferrington@afterpaytouch.com	AUD 318.14	REFUNDED
00002308	2018-07-06 07:39	REBECCA.FERRINGTON@afterpaytouch.com	AUD 223.64	APPROVED
00002307	2018-07-06 06:04	REBECCA.FERRINGTON@afterpaytouch.com	AUD 528.14	REFUNDED
00002306	2018-07-06 05:48	REBECCA.FERRINGTON@afterpaytouch.com	AUD 161.68	REFUNDED

Merchant Tools > After Pay > Transactions Management > Order Details

Details for Order 00009303

Information:	Contains 1 line item.The total price is 146.97
Date Received:	Thu Sep 20 07:50:52 GMT 2018
Site:	SiteGenesis
Created By:	Customer
Customer:	Steven G
Customer No.:	anonymousefJbt36cjCoKarwssfrOdTMsti
Email:	steven.g@touchcorp.com
Transaction Status	APPROVED

Qty	Product ID	Name	Manufacture	Tax Rate	Unit Sale Price	Tax Basis	Item Total
2	701643571284	Sleeveless Cowl Neck Top		5%	AUD 131.98	AUD 131.98	AUD 131.98
						Shipment Shipping Cost:	AUD 7.99
						Total Shipping cost:	AUD 7.99
						Shipping Total:	AUD 7.99
						Tax Total:	AUD 7.00
						Total:	AUD 146.97

Payment Information for Order 00009303

Order Total:	AUD 146.97
Invoice Number:	00006502
Payment Status:	APPROVED
Billing Address:	Steven G 406 Collins St Melbourne VIC 3000 au

Payment Actions

Current status: APPROVED

Refund

5.3 Storefront Functionality

This cartridge provides an additional payment method during checkout

6. Known Issues

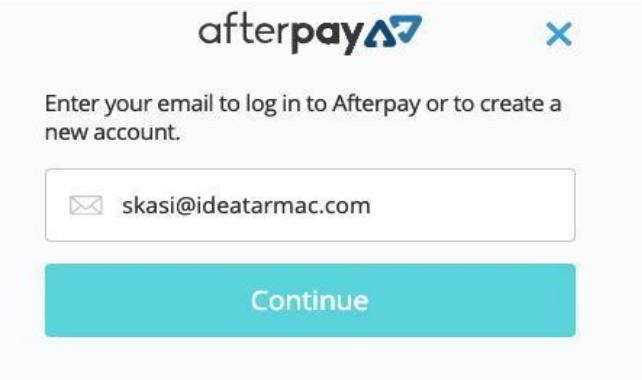
No known issues at the time

7. Testing

Checkout Flow :

Once the cartridge is installed and integrated based on instructions, please try to place an order on your sandbox to test the functionality.

- After you click on Place order then you should be redirected to Afterpay Payment Page. If you are not registered then you will be asked to create an account.



Sandbox Customer Accounts

You can create pre-existing test customer accounts in the Sandbox environment at https://portal.sandbox.afterpay.com/us or within your checkout redirect flow. Each Sandbox customer account requires a unique email address and unique phone number.

Account Data	**Unique**	**Description**
First Name	No	A first name. Do not use 'null'
Last Name	No	A last name. Do not use 'null'
Email Address	Yes	Your email address
Phone Number	Yes	unique 10 digit US phone number
Billing Address	No	a US address. It does not need to be unique
SMS Verification Code	No	**111111**
Credit Card Number Expiry CCV	No	4111 1111 1111 1111 01/22 000

8. Release History

Version	Date	Changes
18.3	3 September 2016	Initial release
18.3	07 May 2018	Pipeline and controller version
19.1.0	20 March 2019	No Changes
20.0.0	09 August 2019	Changed to Afterpay API v2