

**Affirm - SFRA version**

***Version 18.******3.1***



**Table of Contents**

[**Summary**](#_3znysh7) **4**

[**Component Overview**](#_2et92p0) **5**

[Functional Overview](#_tyjcwt) 5

[Use Cases](#_3dy6vkm) 5

[Limitations, Constraints](#_4d34og8) 5

[Compatibility](#_2s8eyo1) 5

[Privacy, Payment](#_17dp8vu) 5

[**Implementation Guide**](#_3rdcrjn) **6**

[Setup](#_26in1rg) 6

[Deploying cartridge to a sandbox](#_lnxbz9) 6

[Sandbox setup](#_35nkun2) 7

[Configuration](#_1ksv4uv) 11

[Site Preferences configuration](#_g8fu1wew601r) 11

[Site Preferences configuration](#_44sinio) 11

[Affirm Services configuration](#_2jxsxqh) 14

[Affirm Job configuration](#_z337ya) 14

[3.2.4 Other config](#_6dlybrrzuzze) 16

[Custom Code (SFRA)](#_3j2qqm3) 16

[External Interfaces](#_1y810tw) 24

[Testing](#_4i7ojhp) 25

[**Operations, Maintenance**](#_2xcytpi) **26**

[Data Storage](#_1ci93xb) 26

[Support](#_3whwml4) 26

[**User Guide**](#_qsh70q) **27**

[Roles, Responsibilities](#_3as4poj) 27

[Business Manager](#_49x2ik5) 27

[Storefront Functionality](#_147n2zr) 27

[**Known Issues**](#_3o7alnk) **28**

[**Release History**](#_23ckvvd) **28**

# Summary

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

# Component Overview

## Functional Overview

“Buy with Affirm” is a closed-end installment loan product offered by [Cross River Bank](http://www.crossriverbank.com/) through Affirm’s technology platform. “Buy with Affirm” allows you to buy goods or services offered by online merchants with whom Affirm partners (“Merchants”). If you agree to use “Buy with Affirm,” Cross River Bank will pay the Merchant on your behalf in exchange for your promise to repay the same amount plus a finance charge as determined by your creditworthiness.

## Use Cases

Customers can use Affirm payment method to pay their purchases.

On the PDP, PLP and Cart Page, Customers can see Affirm promotions beneath the price displayed monthly payment.

## Limitations, Constraints

Affirm cartridge has the following limitation for purchasing products. List of goods that is excluded from cartridge support for buying:

Gift Certificates

## Compatibility

Cartridge is designed and developed for: Salesforce platform version 19.1 (Compatibility Mode 18.1).

## Privacy, Payment

Affirm stores and processes your information maintaining physical, electronic and procedural safeguards. We maintain physical security measures to guard against unauthorized access to systems and use safeguards such as firewalls and data encryption. We enforce physical access controls to our buildings, and we authorize access to personal information only for those employees who require it to fulfill the responsibilities of their jobs.

# Implementation Guide

## Setup

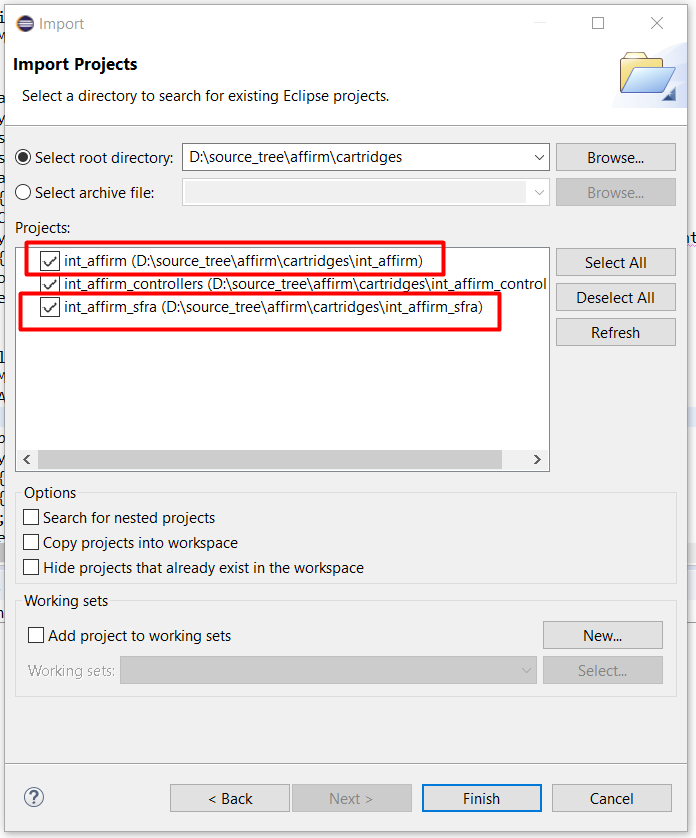
This section describes steps that should be completed before cartridge configuration in Business Manager.

### Deploying cartridge to a sandbox

1. Download cartridge source code.
2. Import the Affirm cartridge to a workspace in Salesforce UX Studio.



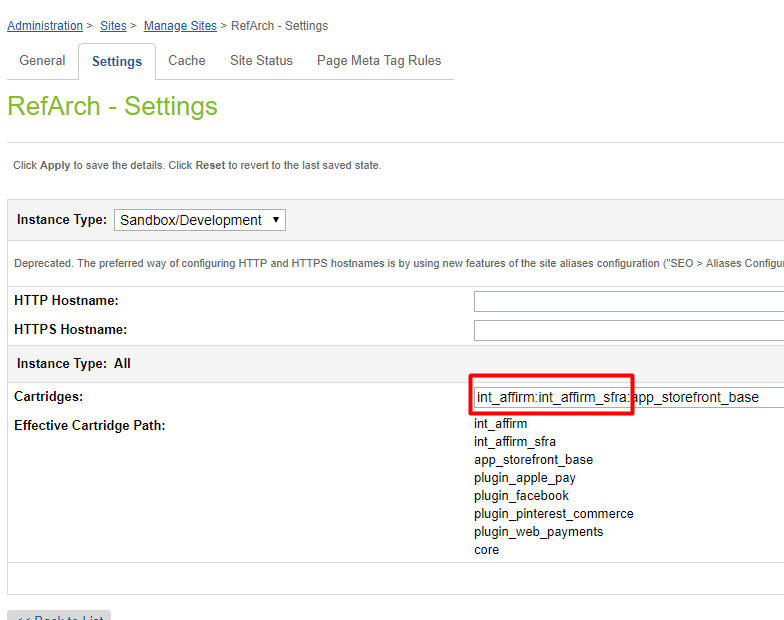
1. Add Affirm cartridges to Project Reference of Server Connection.



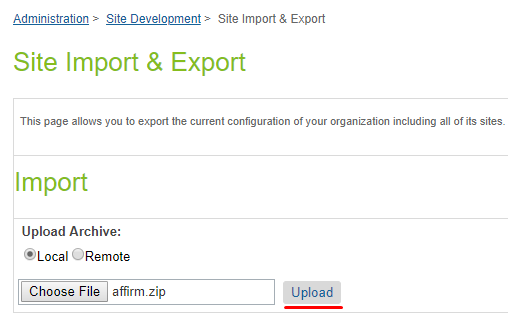
1. Wait until Studio completes workspace build and uploading source codes to a sandbox.

### Sandbox setup

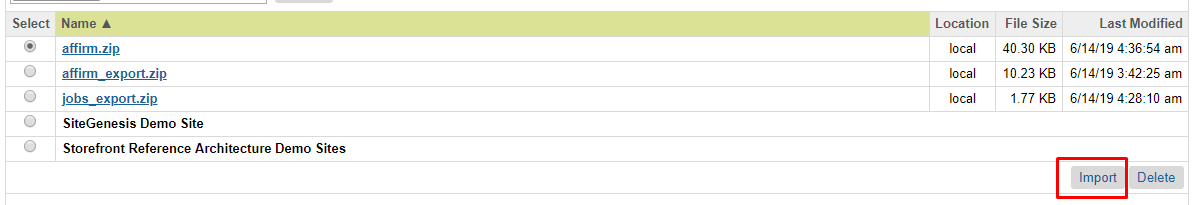
1. Go to Business Manager -> Site -> Manage Sites. Select the correct site, then select Settings tab. In the cartridge path at the beginning write the following: int\_affirm\_sfra:int\_affirm:



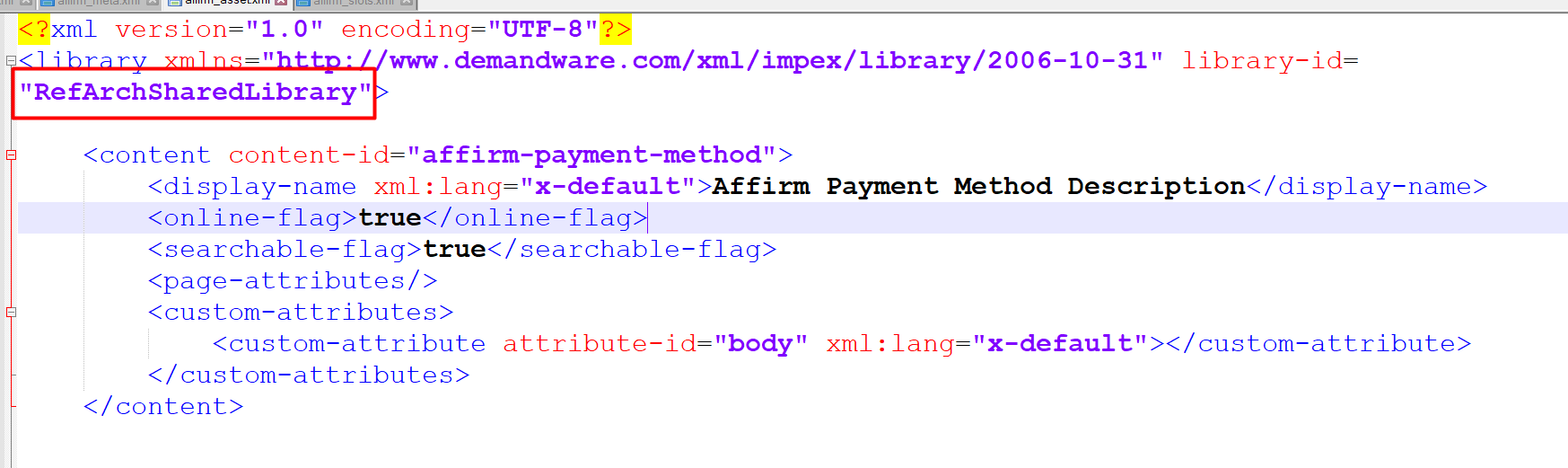
1. Download cartridge source code.
2. Find affirm.zip in metadata folder.
3. Go to Administration > Site Development > Site Import & Export. Click the Upload button and select affirm.zip.



1. Click the Import button and select affirm.zip.



**Note!** You should configure your library ID in library.xml file in metadata folder. Currently there is default ‘**RefArchSharedLibrary’**



1. Verify that Site Preferences group was created with ID – AFFIRM\_PAYMENT and name – Affirm.
2. Verify that Payment Method was created with ID – Affirm and name – Affirm. Enable the Affirm payment method.
3. You can add image or text for your payment method by updating content asset “affirm-payment-method”.

## Configuration

This section describes configuration of the sandbox.

### Site Preferences configuration

1. Go to Merchant Tools > Site Preferences > Custom Preferences> Affirm
2. Enable attribute – Affirm Online Status. This attribute defines status (enable/disable) of the Affirm integration.
3. Select a value from the dropdown Affirm Mode. This attribute defines in which mode the cartridge will work. Allowable values are “Sandbox” and “Production”.
4. Add site preference attribute – Affirm Public Key with provided public key from Affirm.
5. Add site preference attribute – Affirm Private Key with provided private key from Affirm.
6. Select a value from the Affirm Payment Action dropdown. This attribute defines which Affirm payment actions will be used in the checkout process.
7. Enable attribute – Cart Promo Message. This attribute defines status (enable/disable) of Promotional messages on Cart Page.
8. Enable attribute – Product Promo Message. This attribute defines status (enable/disable) of Promotional messages on Product Detail Page.
9. Enable attribute – PLP Promo Message. This attribute defines status (enable/disable) of Promotional messages on Product Listing Page.
10. Add site preference value attribute – Default financing program. This attribute defines which financing program will be used by default (if any other financing programs have not been applied).
11. Add site preference value – Affirm Financing programs mapping. It allows mapping financing programs to promo-id.

Please use the following format: {financing program name}|{promo-id}

Example:



1. Add site preference value – Financing programs customer group. This allows detection of financing programs based on customer group.
2. Please use the following format: {customerGroup}|{financingProgram},
3. Example: Registered|registeredFinProgram



1. Add site preference value – Financing programs cart total range. This allows detection of financing programs based on cart total range.
2. Please use the following format: {minValue}|{maxValue}|{financingProgram}.
3. Example: 100|200|CartRange



1. Add site preference value – Financing programs date range. This allows detection of financing programs based on date range.
2. Please use the following format: {startDate}|{endDate}|{financingProgram}, where startDate and endDate format is “yyyy-MM-dd”.
3. Example: 2017-01-01|2017-02-28|myWinter



1. Enable attribute – Affirm Virtual Card Integration.
2. Add site preference attribute – VCN Payment Instrument with last part of hook name (see External Interfaces section of this document).
3. Add site preference value – Affirm Min Total. It allows setting minimal price value for displaying promotional messages. Products with price lower than this value will not have a promotional message.
4. Add site preference attribute – Payment Min Total with minimal cart total when Affirm payment method is available.
5. Add site preference attribute – Payment Max Total with maximal cart total when Affirm payment method is available.
6. Add site preference value – Enable Affirm Analytics. It allows to put analytics script into order confirmation page

### Affirm Job configuration

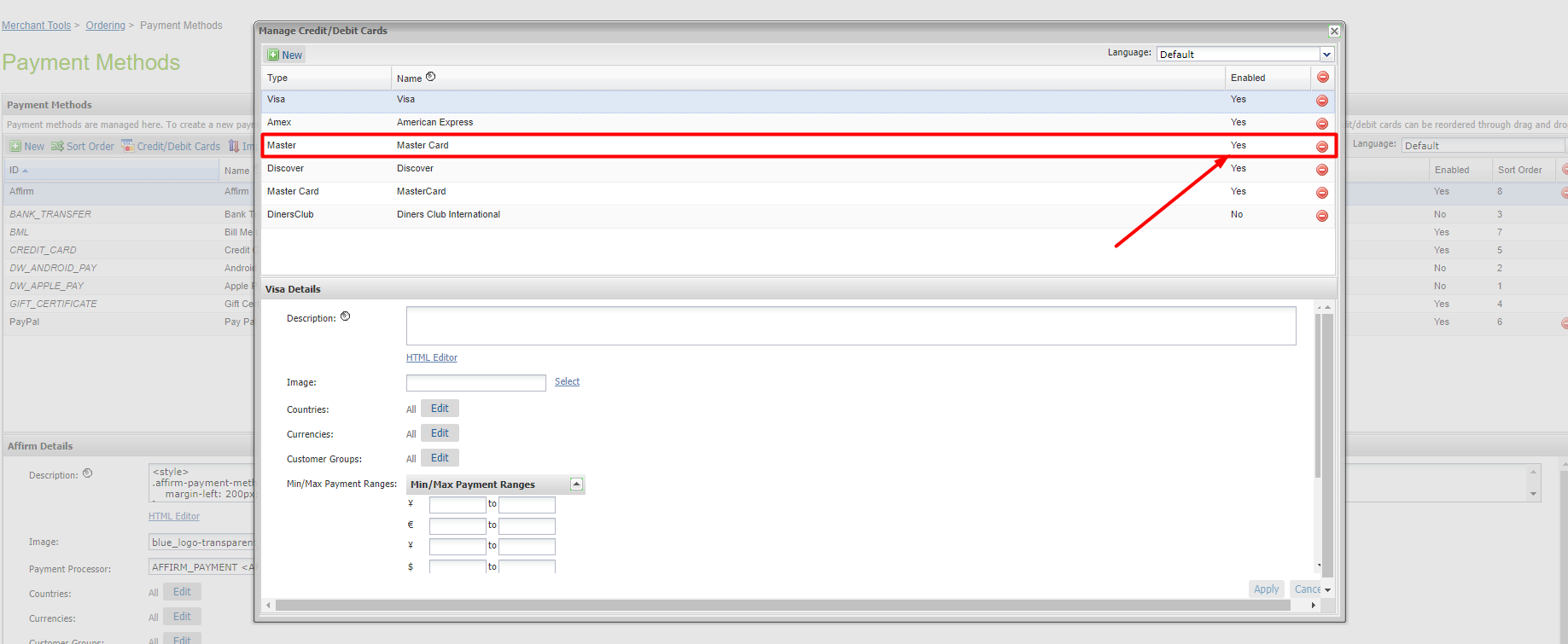
1. Go to Administration > Operations > Job Schedules
2. Verify that each imported job was created.



1. Change execution scope of every job to your site.

### Other configuration

Credit Card type of Master Card should be enabled.



## Custom Code (SFRA)

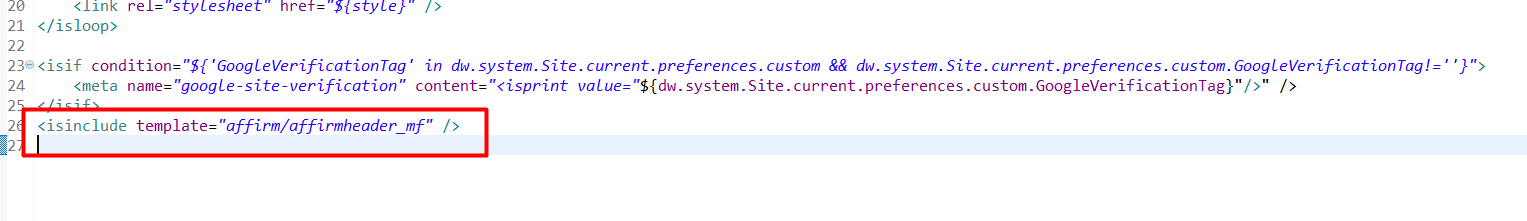
This section describes changes that should be made to a merchant’s custom cartridge. All of this functionality is implemented in int\_affirm\_sfra cartridge for default Storefront Reference Architecture so you can find exact code there.

1. Find and open the below template **/ommon/htmlHead.isml**. Find the following code.



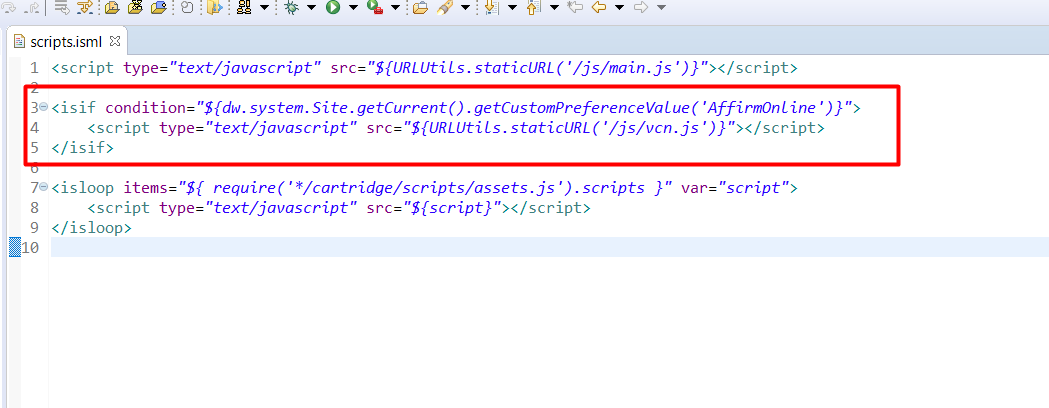
Paste code after:

<isinclude template=*”affirm/affirmheader\_mf”* />



1. Find and open template –**/common/scripts.isml**. Paste code at the top of the file:

<isif condition=”${dw.system.Site.getCurrent().getCustomPreferenceValue(‘AffirmOnline’)}”>   
 <script type=”text/javascript” src=”${URLUtils.staticURL(‘/js/vcn.js’)}”></script>  
</isif>

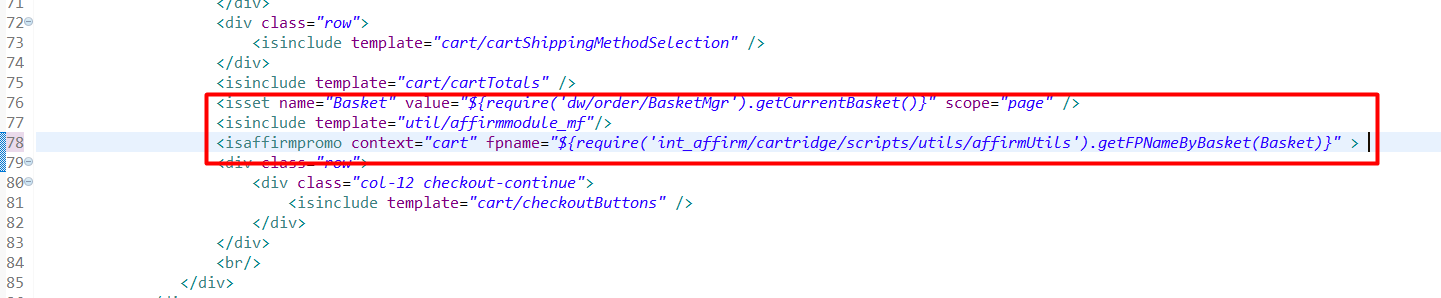


1. Find and open template –**/cart/cart.isml**. Paste the following code in the section where you want to see Affirm promo message. If it should be below the cart then you need to paste the code after **<isinclude template=”cart/cartTotals” />** line:

<isset name=”Basket” value=”${require(‘dw/order/BasketMgr’).getCurrentBasket()}” scope=”page” />

<isinclude template=”util/affirmmodule\_mf”/>

<isaffirmpromo context=”cart” fpname=”${require(‘int\_affirm/cartridge/scripts/utils/affirmUtils’).getFPNameByBasket(Basket)}” >



1. Paste the following code where the Affirm promo message should be shown on product details page. You need to modify setDetails.isml, productDetails.isml, bundleDetails.isml in the product folder:

<isinclude template=”util/affirmmodule\_mf”/>

<isaffirmpromo context=”pdp” fpname=”${require(‘int\_affirm\_mobile\_first/cartridge/scripts/utils/affirmUtils’).getFPNameForPDP(product)}” >

**sfra/cartridge/templates/default/product/setDetails.isml**



**sfra/cartridge/templates/default/product/setDetails.isml**



**sfra/cartridge/templates/default/product/productDetails.isml**

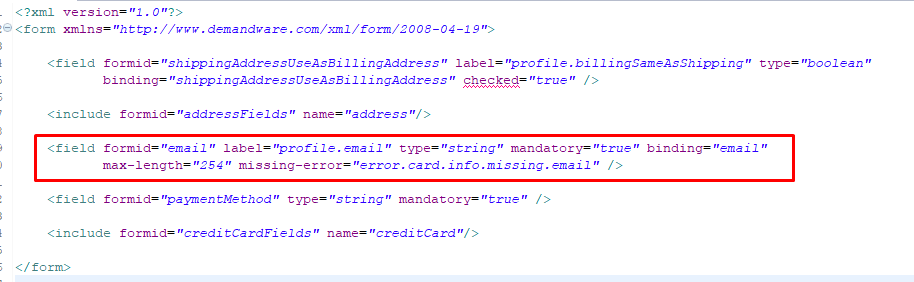


1. Paste the following code where the Affirm promo message should be shown on the product listing page. You need to modify **product/productTile.isml**

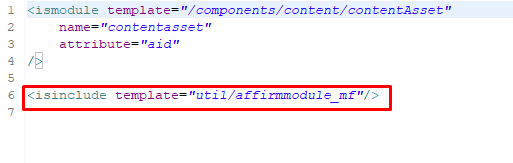
<isinclude template=”util/affirmmodule\_mf”/>

<isaffirmpromo context=”plp” fpname=”${require(‘int\_affirm\_mobile\_first/cartridge/scripts/utils/affirmUtils’).getFPNameForPLP(product)}” price=”${price}”/>



1. Since credit card is hardcoded in Storefront Reference Architecture you need to manually modify templates to enable the Affirm payment:  
   Forms folder:  
   **billing.xml** – added email field in billing level  
   
2. Find and open template: **components/modules.isml.**

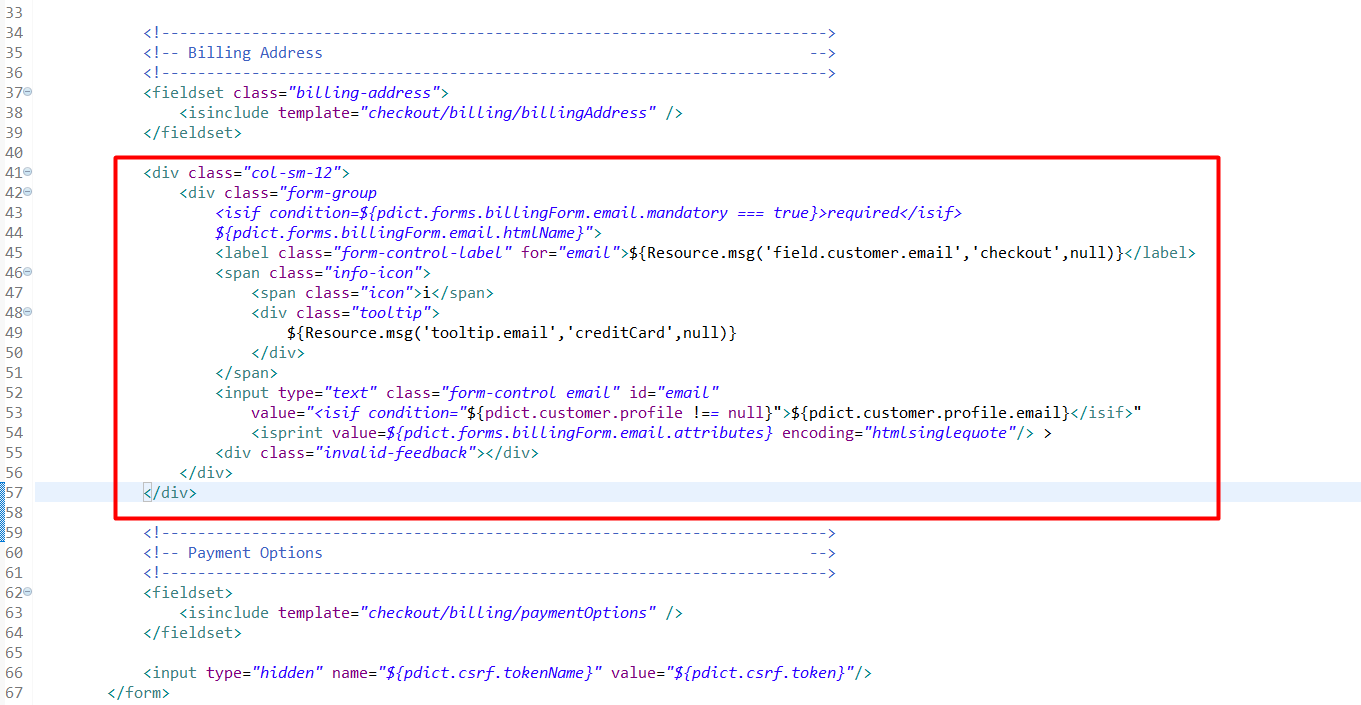
Paste code after the last line:

<isinclude template=”util/affirmmodule\_mf”/>  


1. **creditCard.xml** - removed email field



Templates folder:

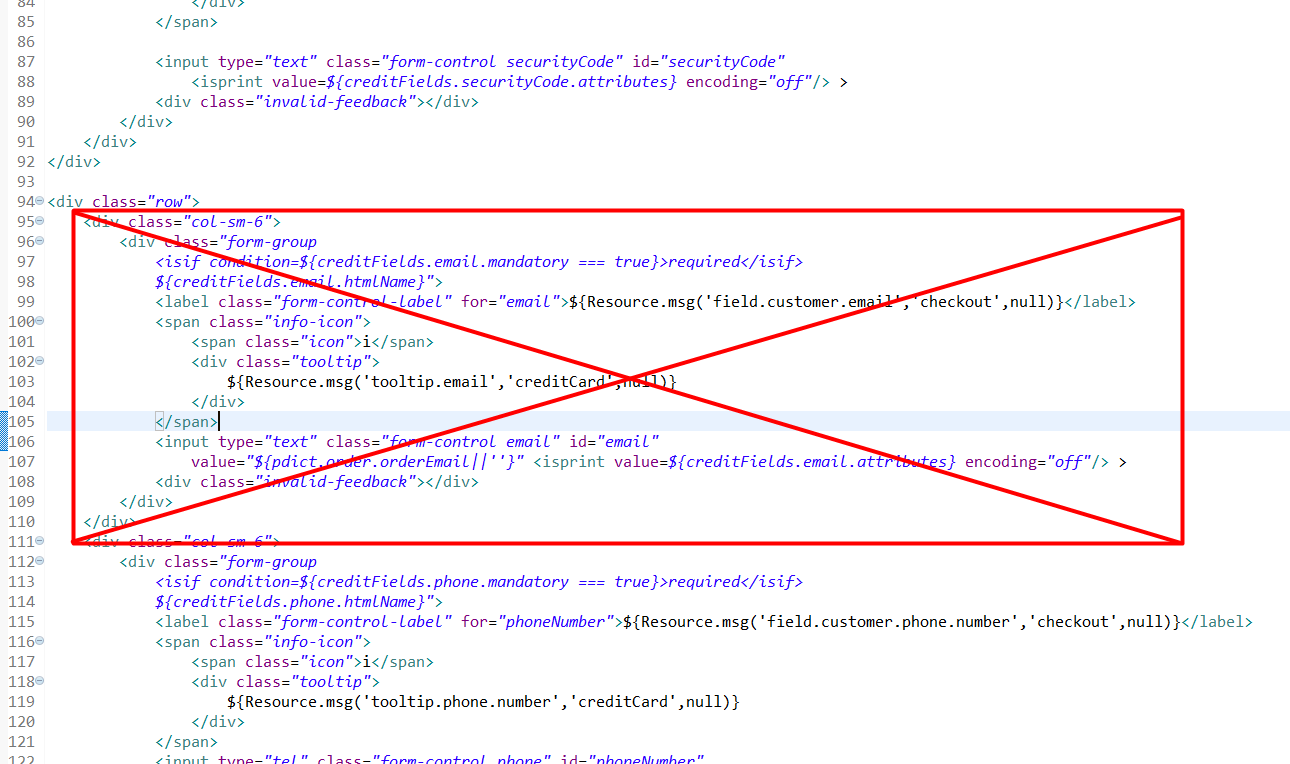
**checkout/billing/billing.isml** – added email field HTML  


**checkout/billing/billingSummary.isml** – added div with vcn data:

<isset name=”CurrentBasket” value=”${require(‘dw/order/BasketMgr’).getCurrentBasket()}” scope=”page” />  
 <isif condition=”${CurrentBasket}”>   
 <isset name=”basketCheck” value=”${CurrentBasket.getAllProductLineItems().isEmpty()}” scope=”page” />  
 <isif condition=”${!basketCheck}”>  
 <div  
 data-vcndata=’<isprint value=”${affirm.basket.getCheckout(CurrentBasket, 1)}” encoding=”htmlsinglequote” />’  
 data-enabled=’<isprint value=”${affirm.data.getAffirmVCNStatus() == ‘on’}” encoding=”htmlsinglequote” />’  
 data-affirmselected=’<isprint value=”${true}” encoding=”on” />’  
 data-errormessages=’<isprint value=”${affirm.data.getErrorMessages()}” encoding=”htmlsinglequote” />’  
 data-updateurl=’<isprint value=”${dw.web.URLUtils.https(“Affirm-Update”)}” encoding=”htmlsinglequote” />’  
 data-errorurl=’<isprint value=”${dw.web.URLUtils.https(“Checkout-Begin”)}” encoding=”htmlsinglequote” />’  
 data-csrfname=’<isprint value=”${dw.web.CSRFProtection.getTokenName()}” encoding=”htmlsinglequote” />’  
 data-csrftoken=’<isprint value=”${dw.web.CSRFProtection.generateToken()}” encoding=”htmlsinglequote” />’  
 id=”vcn-data”></div>  
 </isif>  
 </isif>

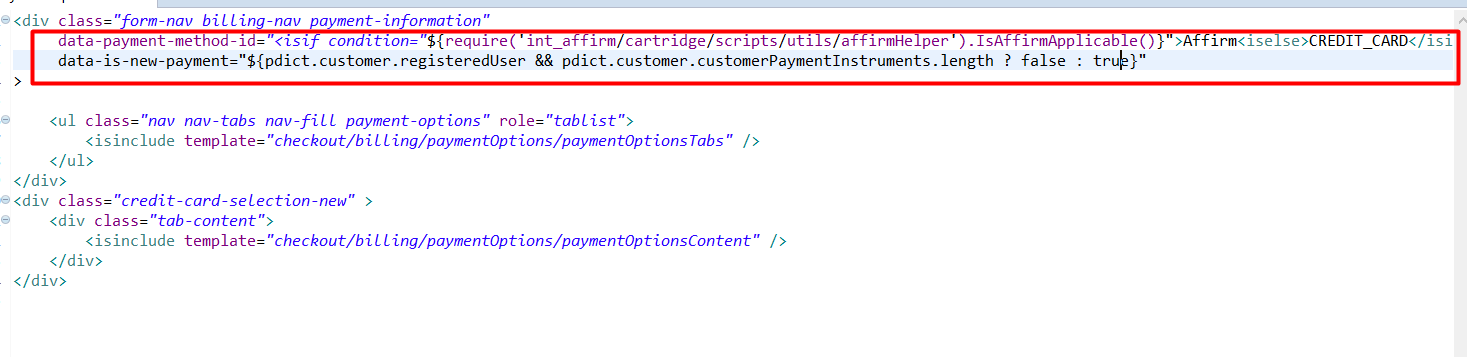


**checkout/billing/creditCardForm.isml** – removed email field from credit card HTML



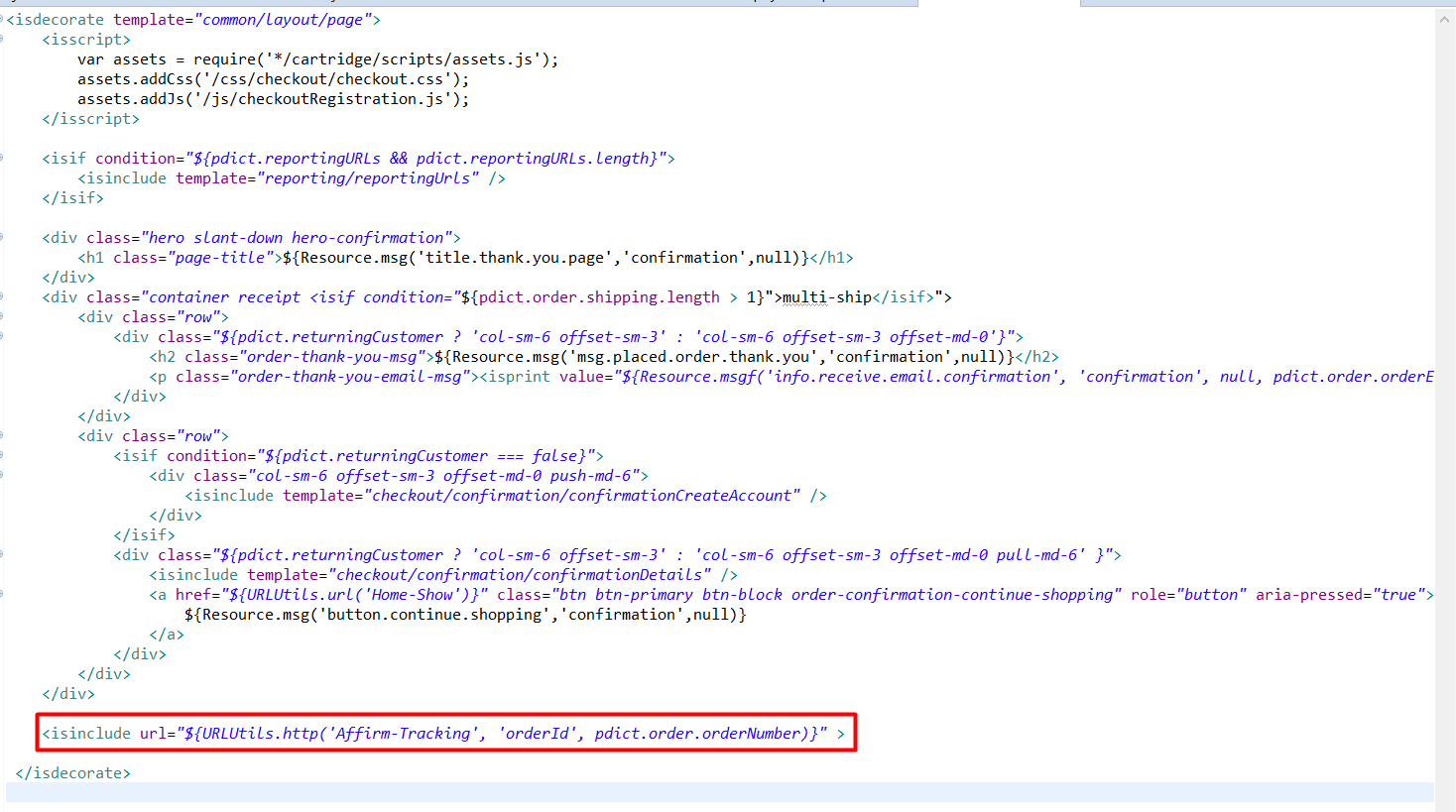
**checkout/billing/paymentOption.isml** – added condition for switching payment methods if Affirm is not applicable.

<div class=”form-nav billing-nav payment-information”  
 data-payment-method-id=”<isif condition=”${require(‘int\_affirm/cartridge/scripts/utils/affirmHelper’).IsAffirmApplicable()}”>Affirm<iselse>CREDIT\_CARD</isif>”  
 data-is-new-payment=”${pdict.customer.registeredUser && pdict.customer.customerPaymentInstruments.length ? false : true}”  
>



**checkout/confirmation/confirmation.isml** before closing </isdecoratre> tag

<isinclude url=”${URLUtils.http(‘Affirm-Tracking’, ‘orderId’, pdict.order.orderNumber)}” >



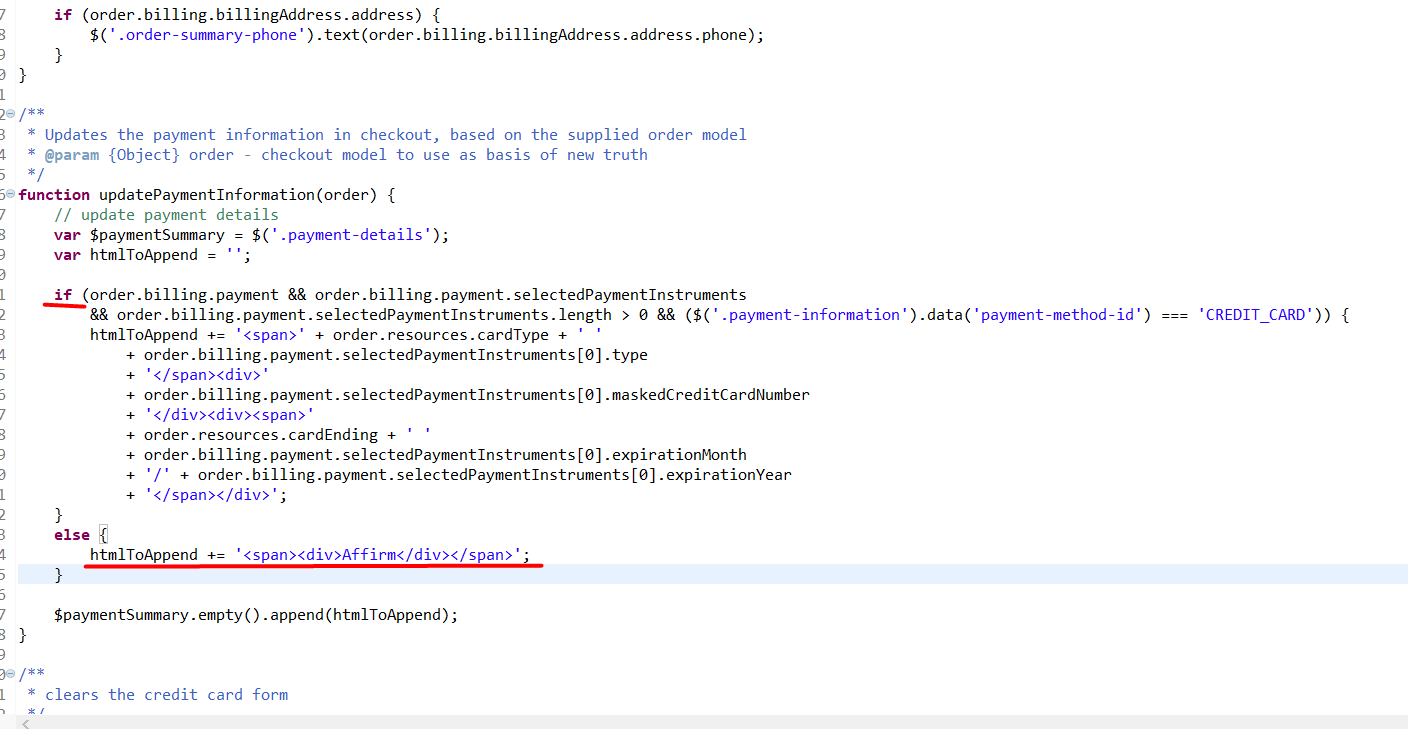
**Controllers:**

CheckoutServices.js was overridden for handling both Affirm and Credit card payments.

Client JS code:

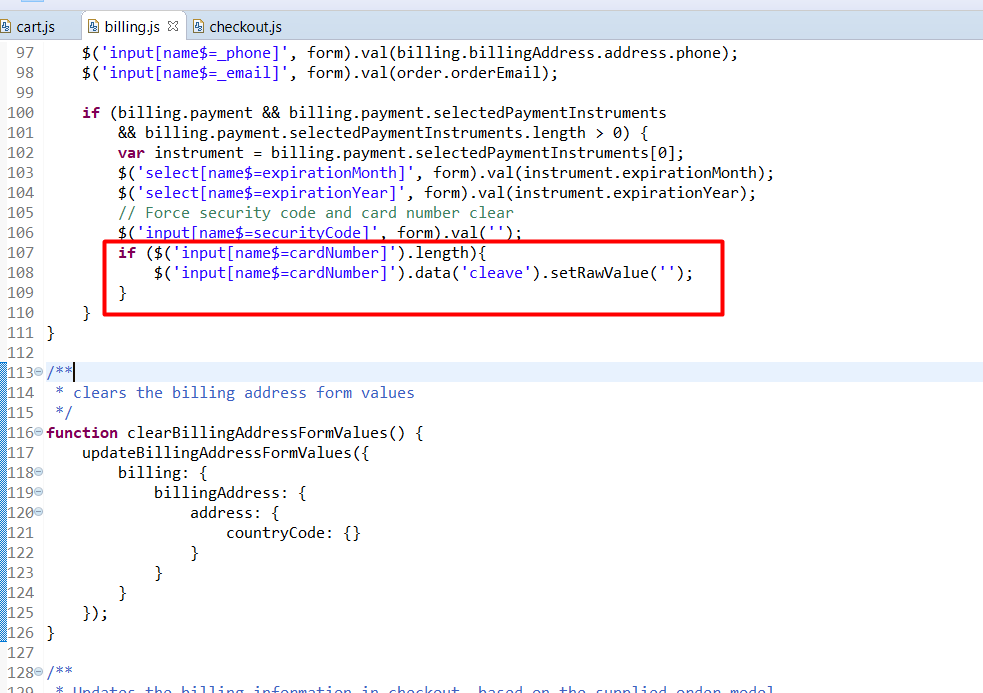
**js/checkout/billing.js** – added holder for updating payment information.

If (order.billing.payment && order.billing.payment.selectedPaymentInstruments  
 && order.billing.payment.selectedPaymentInstruments.length > 0 && ($(‘.payment-information’).data(‘payment-method-id’) === ‘CREDIT\_CARD’)) {  
 htmlToAppend += ‘<span>’ + order.resources.cardType + ‘ ‘  
 + order.billing.payment.selectedPaymentInstruments[0].type  
 + ‘</span><div>’  
 + order.billing.payment.selectedPaymentInstruments[0].maskedCreditCardNumber  
 + ‘</div><div><span>’  
 + order.resources.cardEnding + ‘ ‘  
 + order.billing.payment.selectedPaymentInstruments[0].expirationMonth  
 + ‘/’ + order.billing.payment.selectedPaymentInstruments[0].expirationYear  
 + ‘</span></div>’;  
 }  
 else {  
 htmlToAppend += ‘<span><div>Affirm</div></span>’;  
 }



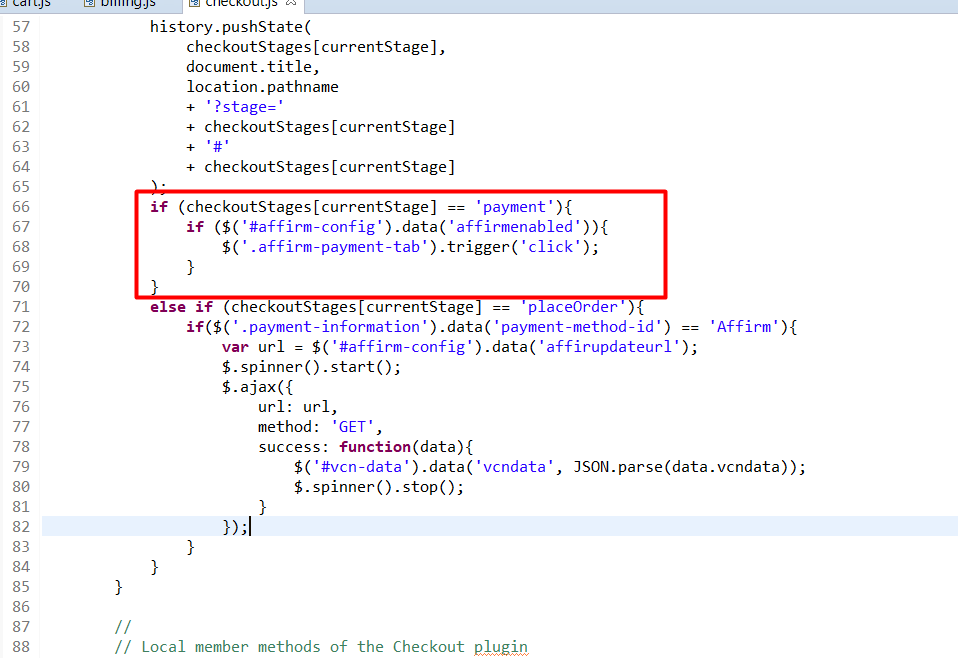
Added checks if credit card fields are absent.

If ($(‘input[name$=cardNumber]’).length){  
 $(‘input[name$=cardNumber]’).data(‘cleave’).setRawValue(‘’);  
 }



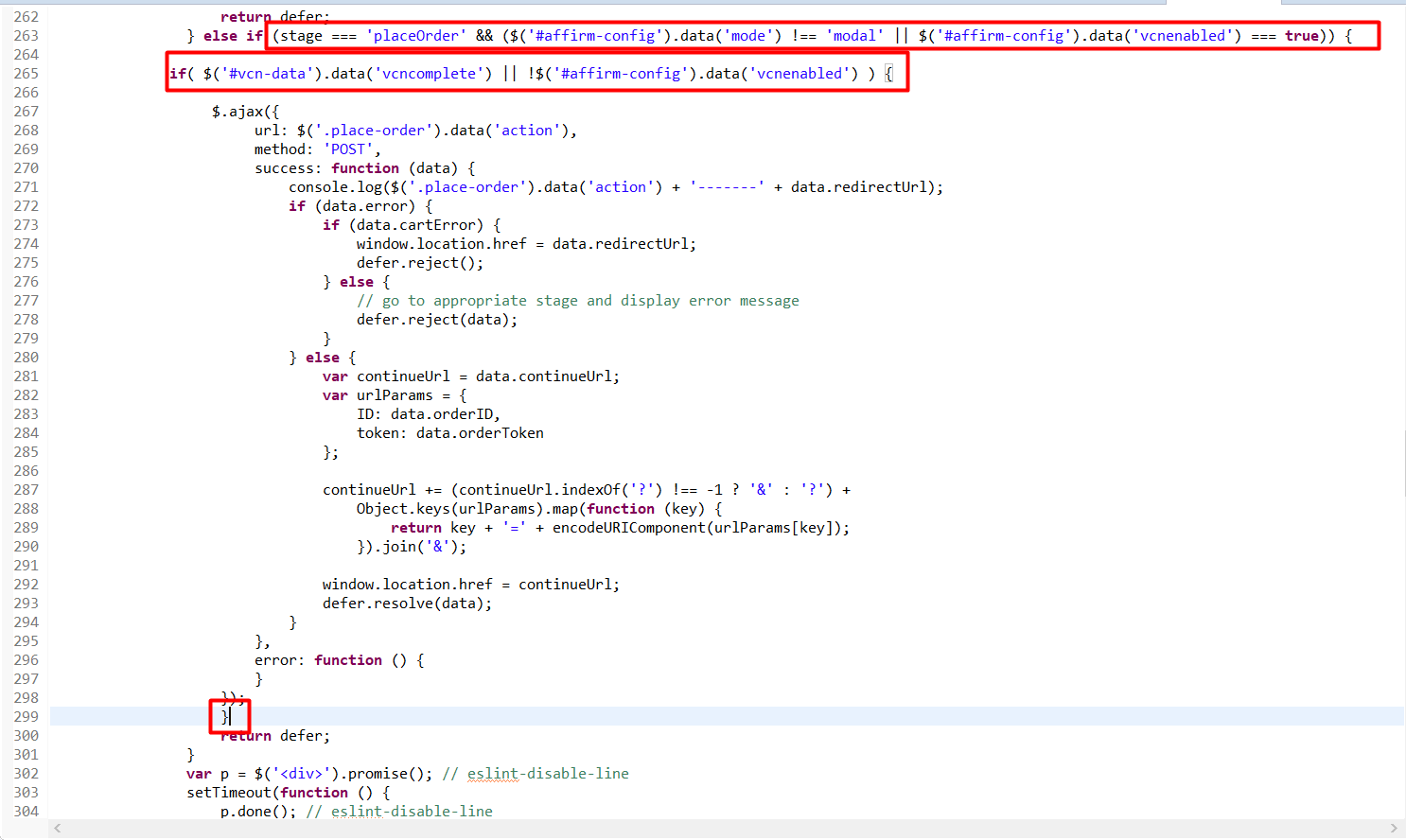
**js/checkout/checkout.js**  
added checking to ensure affirm object is formed properly.

If (checkoutStages[currentStage] == ‘payment’){  
 if ($(‘#affirm-config’).data(‘affirmenabled’)){ $(‘.affirm-payment-tab’).trigger(‘click’);  
 }  
 }



Added insertion of Modal mode to checkout steps as a reminder to close the curly bracket.

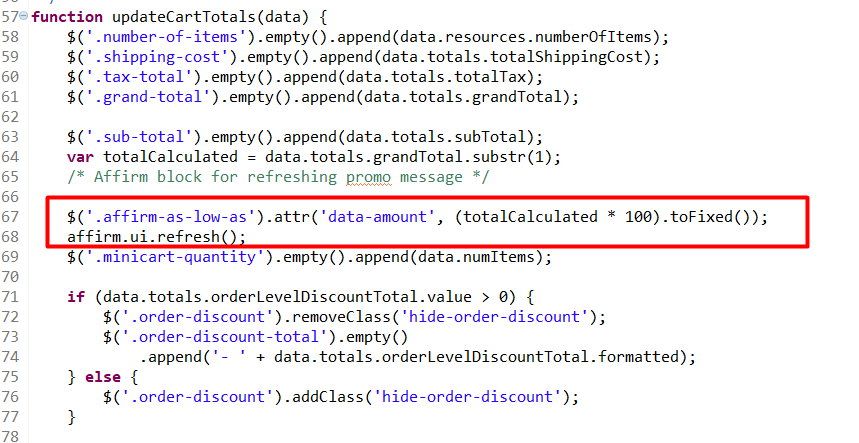
(stage === ‘placeOrder’ && ($(‘#affirm-config’).data(‘mode’) !== ‘modal’ || $(‘#affirm-config’).data(‘vcnenabled’) === true)) {  
   
 if( $(‘#vcn-data’).data(‘vcncomplete’) || !$(‘#affirm-config’).data(‘vcnenabled’) ) {



**js/cart/cart.js**

Added recalculation for the Affirm Modal Promo price.

$(‘.affirm-as-low-as’).attr(‘data-amount’, (totalCalculated \* 100).toFixed());  
 affirm.ui.refresh();



**Don’t forget to rebuild your app.js after changing JS-files**

## External Interfaces

Commutation between Affirm service and Salesforce Commerce Cloud Platform is based on Affirm 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 Affirm is wrapped into Service that is handling monitoring and logging functionality.

Affirm integration documentation –<https://docs.affirm.com/Integrate_Affirm/Direct_API>

For as low as – <https://docs.affirm.com/Integrate_Affirm/Promotional_Messaging>

Virtual card integration assumes processing credit card using customer’s payment processors. Initially, it is BASIC\_CREDIT processor but if you need to integrate Affirm to site with other payment processors, do the next steps:

* Set site preference attribute “VCN Payment Instrument” with the last part of the hook name, e.g. “CYBERSOURCE\_CREDIT\_CARD” (“BASIC\_CREDIT” by default);
* Register new hook handler for extension point “dw.int\_affirm.payment\_instrument.%PI%”, where %PI% is payment instrument value you’ve set in previous step. For example “dw.int\_affirm.payment\_instrument.CYBERSOURCE\_CREDIT\_CARD”. By default it is “dw.int\_affirm.payment\_instrument.BASIC\_CREDIT”, as defined in the file “int\_affirm\_sfra/cartridge/scripts/hooks.json”;
* Implement a method “add” of this hook handler. It will use basket as the parameter. The method should return a newly created payment instrument. Credit card info can be extracted from HttpParametersMap. An example of the implementation can be found in “int\_affirm\_sfra/cartridge/scripts/payment/instrument/BASIC\_CREDIT.js”.

## Testing

Test scenarios and test cases are included in the document Affirm Test Cases.docx

# Operations, Maintenance

## Data Storage

The Affirm cartridge extends Salesforce Commerce Cloud system objects to store related Affirm data form request. The following objects were extended: Order, Product, Category and SitePreference.

## Support

In case of availability issues, Affirm support can be reached via email at [merchanthelp@affirm.com](mailto:merchanthelp@affirm.com).

# User Guide

## Roles, Responsibilities

There are no recurring tasks that need to be fulfilled to set up the Affirm cartridge. As long as the configuration details are correct in your Site Preferences based on instructions in section 3.1 3.2 and 3.3 that is all that needs to be done.

## Business Manager

The Int\_affirm cartridge should be placed into the BM cartridge path to avoid error in logs while using BM.

## Storefront Functionality

The Affirm cartridge is an implementation of the Affirm payment method. Affirm is represented as a default payment method in Salesforce Commerce Cloud platform. Such implementation provides native support of Payment methods.

# Known Issues

|  |  |
| --- | --- |
| **#** | **Description** |

# Release History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 15.1.0 | 07/03/2015 | Initial release |
| 17.1.0 | 02/01/2017 | * Controllers support * Enhancements and bug fixes * Added VCN support * Promotional Messaging support |
| 17.1.1 | 04/12/2017 | * Bug fixes for certification |
| 17.1.2 | 08/11/2017 | * Fix for Inconsistent Payment display at checkout after editing payment method |
| 17.1.3 | 09/18/2017 | * Ability to disable Affirm as a payment method |
| 18.1.0 | 05/27/2018 | * SFRA support |
| 18.2.0 | 12/13/2018 | * Added attributes to promo messages * Added tracking script on order confirmation page * Cartridge restructure |
| 18.3.1 | 14/06/2019 | * Metadata restructure * Changed view of Affirm promo |