

# Klarna Payments

---

*Version 19.1.6*



## Table of Contents

1.	Summary.....	1-3
2.	Component Overview.....	2-4
2.1	Functional Overview.....	2-4
2.2	Use Cases.....	2-5
2.3	Limitations, Constraints.....	2-23
2.4	Compatibility.....	2-24
2.5	Privacy, Payment Data.....	2-24
3.	Implementation Guide.....	3-29
3.1	Setup.....	3-29
3.1.1	int_klarna_payments.....	3-29
3.1.2	int_klarna_payments_pipelines.....	3-30
3.1.3	int_klarna_payments_controllers.....	3-30
3.2	Configuration.....	3-31
3.2.1	Metadata Import.....	3-31
3.2.2	Cartridge Path.....	3-32
3.2.3	Klarna Payments Configurations.....	3-33
3.2.4	Klarna Payments Logo and Payment Option Name.....	3-35
3.2.5	Services Configuration.....	3-36
3.2.6	Custom attributes.....	3-37
3.3	Custom Code.....	3-38
3.3.1	Integration efforts.....	3-38
3.3.2	Templates modifications.....	3-38
3.3.3	Pipeline modifications.....	3-44
3.3.3.1	COBilling pipeline.....	3-44
3.3.3.2	COSummary pipeline.....	3-46
3.3.3.3	COPlaceOrder pipeline.....	3-47
3.3.4	Controller modifications.....	3-48
3.3.4.1	COBilling controller.....	3-48
3.3.4.2	COSummary controller.....	3-50
3.3.4.3	OrderModel.js.....	3-51
3.3.5	VCN Decryption.....	3-52
4.	Release History.....	4-53

## 1. Summary

This cartridge enables a Commerce Cloud store to use the Klarna Payments service.

The developer has to install the cartridge and integrate it into the online store following the instructions from this document. Merchant teams are required also to configure the cartridge with the valid merchant credentials and site configurations in Commerce Cloud Business Manager to enable Klarna payments methods in the checkout. It is a requirement that Merchant sign a contract for integration support and production go-live with Klarna.

Klarna offers a test environment, so the integration can be tested before switching to the Klarna production environment. Based on the contract, Klarna shall provide assistance with integration and testing prior to sign-off for go-live.

The integration consists of an archive, which contains the following contents:

- Cartridge called 'int\_klarna\_payments', and optionally either "int\_klarna\_payments\_pipelines" or "int\_klarna\_payments\_controllers" depending on which architecture is to be employed.
- A site-template archive containing new attributes and settings
- This document
- The integration is based on the Site Genesis demo store provided by Commerce Cloud.

## 2. Component Overview

### 2.1 Functional Overview

---

Klarna payment can be integrated via the Klarna Payment REST API and the JavaScript SDK on the storefront. Klarna Payment enables consumers to choose from the different payment method products offered by Klarna. Multiple Klarna products are available within the categories Pay Now, Pay Later and Pay Over Time. The cartridge integration displays payment options via a widget (iframe) added inline on the billing page, referred to as Klarna widget or just “the widget”. The widget with information about the payment method is displayed to the customer when the individual clicks on the Klarna payment method.

Customers can authorize the payment after reviewing the payment method terms and clicking Place Order button. The Klarna order is confirmed once order is placed and customer re-directed to the confirmation page.

Klarna Payment integration sent customer browser to the location returned in `redirect_url` as Klarna needs to interact with customer’s browser as first party before redirecting to the confirmation page on the store. This ensures a faster checkout experience for returning customers, as the customers device is recognized.

Orders successfully placed with Klarna return a Fraud Status: APPROVED and displayed in Business manager (BM). Klarna orders with Fraud Status PENDING, updates are sent to the `notification_url` on the merchant Commerce Cloud site within 4-24 hours. The updated Fraud status depends on the fraud screening, (e.g. `FRAUD_RISK_ACCEPTED`, `FRAUD_RISK_REJECTED`, `FRAUD_RISK_STOPPED`) returns a Fraud Status and displayed in BM. The push notification is repeatedly sent until (up-to 24 hours, every 10 mins) the POST request is acknowledged with a 200 response.

In case order is ACCEPTED, order creation in SCC proceeds as usual, in case of REJECTED status, order is failed and in case of PENDING status the 'two phase ordering' approach is used, where the order is first created but only 'placed' after notification for ACCEPTED status is asynchronously received. If neither ACCEPTED, nor REJECTED status is received the order will stay in status 'created' until an action is taken by the merchant. Klarna payment status is saved in a custom attribute with id `kpFraudStatus` in the PaymentTransaction system object, and can be seen in BM on the order details Payment tab as below:

Merchant Tools >
Ordering >
Orders >
Order: 00005402(SiteGenesis)

General

Attributes

Payment

Notes

History

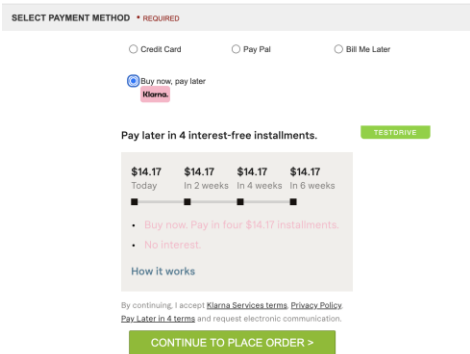
Payment Information for Order '00005402'

Order Total:	\$265.63
Amount Paid:	\$0.00
Balance Due:	\$265.63
Invoice Number:	00023003
Payment Status:	Not Paid
Payment Method:	<div> <div>Klarna</div> <div>Processor: KLARNA_PAYMENTS</div> <div>Transaction: f909b950-faa5-5b8e-b8f7-b6857aba0a59</div> <div>Amount: \$265.63</div> </div> <div>Fraud Status: ACCEPTED</div>

2.2    Use Cases

1. Create an order in SCC using Klarna Payments as a payment method though Klarna Payments widget on the billing page. All the SCC OOTB checkout functionality remains in place, such as but not limited to: cart updates during checkout, checkout with applied coupon(s) code(s), checkout with applied product level promotion, checkout with applied order level promotion, checkout with applied shipping level promotion, checkout with applied order level promotion with bonus product.

Select 'Klarna' as the payment method on Billing page of checkout process and click the 'CONTINUE TO PLACE ORDER' button:



SELECT PAYMENT METHOD \* REQUIRED

☐ Credit Card
 ☐ Pay Pal
 ☐ Bill Me Later

☒ Buy now, pay later  
**Klarna**

Slice it  
 Pay a little every month

TEST DRIVE **Klarna**

12 months	from \$17.50/mo.
18 months	from \$11.67/mo.

- Minimum total: \$209.95
- APRs from 0%-19.99%\*
- No impact on credit score to check eligibility

\* Promotional offer. Subject to a potential \$2 monthly minimum interest charge and [credit terms](#). Accounts are issued by WebBank.




You will see the Order Confirmation page with payment method Klarna and can press the 'PLACE ORDER' button:

salesforce commerce cloud

Enter Keyword or Item No.

[NEW ARRIVALS](#)
[WOMENS](#)
[MENS](#)
[ELECTRONICS](#)
[TOP SELLERS](#)

STEP 1: Shipping > STEP 2: Billing > **STEP 3: Place Order**

PRODUCT	QTY	TOTAL
 <b>Quilted Jacket</b> Item No.: 701642853695 Color royal Size L	1 In Stock	\$110.99
 <b>Pull On Pant</b> Item No.: 701642867098 Color Grey Heather Size L	1 In Stock	\$69.00
 <b>Zerrick</b> Item No.: 740357357531 Color Black Size 6.5 Width M	1 In Stock	\$99.00

Help? 800-555-0199

**ORDER SUMMARY** [Edit](#)

Subtotal	\$278.99
<a href="#">Edit</a> Shipping Ground	\$9.99
Sales Tax	\$14.45
<b>Order Total:</b>	<b>\$303.43</b>

**SHIPPING ADDRESS** [Edit](#)

Alexander Gaydardzhiev  
Sofia  
Washington, DC 20230  
United States  
Method: Ground

**BILLING ADDRESS** [Edit](#)

Alexander Gaydardzhiev  
Sofia  
Washington, DC 20230  
United States

**PAYMENT METHOD** [Edit](#)

Klarna  
Amount: \$303.43

[« Edit Cart](#)
[PLACE ORDER](#)

The customer's browser is sent to the redirect\_url and immediately thereafter shown the Commerce cloud Order Confirmation page.

[NEW ARRIVALS](#)
[WOMENS](#)
[MENS](#)
[ELECTRONICS](#)
[TOP SELLERS](#)

Thank you for your order.

If you have questions about your order, we're happy to take your call (800-555-0199) Monday - Friday, 8AM - 8PM

---

**Order Number: 00000907**

Order Placed: **Apr 5, 2017**

**PAYMENT METHOD**

Klarna  
Amount: \$303.43

**BILLING ADDRESS**  
Alexander Gaydardzhiev  
SoRa  
Washington, DC 20030  
United States  
Phone: 3333333333

**SHIPMENT NO. 1**

**SHIPPING STATUS:**  
Not Shipped

**METHOD:**  
Ground

**SHIPPING TO**  
Alexander Gaydardzhiev  
SoRa  
Washington, DC 20030  
United States  
3333333333

ITEM	QTY	PRICE
<p><b>Quilted Jacket</b> ITEM NO: 701A-2553675 COLOR: royal SIZE: L</p>	1	\$110.99
<p><b>Pull On Pant</b> ITEM NO: 701A-2567098 COLOR: Grey Heather SIZE: L</p>	1	\$69.00
<p><b>Zerrick</b> ITEM NO: 740357357531 COLOR: Black SIZE: 4-5 WIDTH: 1-1</p>	1	\$99.00

**PAYMENT TOTAL**

Subtotal: \$278.99

Shipping Ground: \$7.99

Sales Tax: \$14.45

**Order Total: \$303.43**

The newly created order can be inspected in BM

Sandbox - klarna01

SiteGenesis

[Merchant Tools](#)
[Administration](#)
[Storefront](#)

[Merchant Tools](#)
[Orders](#)
[Orders](#)

### Orders

You are using your new Search service.  
This page allows you to search for orders by order number. Select Advanced to use more search options. Select By Number to search by providing a list of order numbers. Order numbers can be separated by either ";" or "," or space or newline. Extended text is treated as case-sensitive, substring matching is not supported.

Order Search

Order Number:

Number	Order Date	Site	Created By	Registration Status	Customer	Email	Total	Status
00000907	4/5/17 12:59:31 pm Etc/UTC	SiteGenesis	Customer	Unregistered	Alexander Gaydardzhiev	alexander.gaydardzhiev@zautre.com	\$303.43	New
00000906	4/5/17 7:39:01 am Etc/UTC	SiteGenesis	Customer	Registered	Yalamova	denitsa.yalamova@zautre.com	\$1,749.26	Open
00000905	4/5/17 7:20:49 am Etc/UTC	SiteGenesis	Customer	Registered	Yalamova	denitsa.yalamova@zautre.com	\$56.69	New

Klarna Payments order id can be inspected in the Attributes tab of the order

Sandbox - klarna01

SiteGenesis

[Merchant Tools](#)
[Administration](#)
[Storefront](#)

[Merchant Tools](#)
[Orders](#)
[Orders](#)
[Order: 00000907\(SiteGenesis\)](#)

[General](#)
[Attributes](#)
[Payment](#)
[Notes](#)
[History](#)

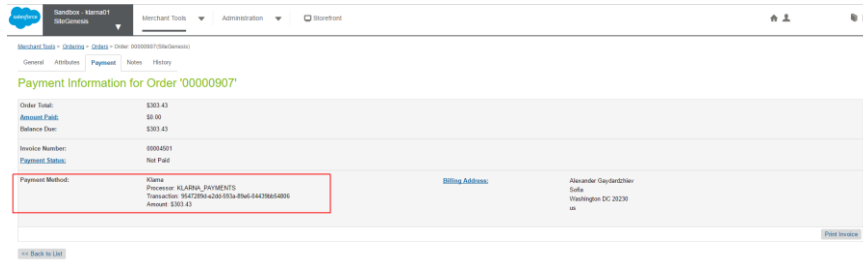
### Attributes for Order '00000907'

On this page you can edit the attributes of the order. Fields with a red asterisk (\*) are mandatory. Click Apply to save changes. Click Reset to revert your changes.

Klarna Payments Order ID:

[<< Back to List](#)

Payment method details can be inspected on the Payment tab of the order, and it should be Klarna



Order can be further inspected in Klarna Merchant Portal:

EU: eu.portal.klarna.com, US: us.portal.klarna.com, OC: us.portal.klarna.com

**Klarna. Merchant Portal**

[Overview](#) > Order #FWZ0CSLP

## #FWZ0CSLP

Merchant reference 1  
**00067625**

[Edit](#)

Created  
**22 Jun 2020, 11:27**

Expires  
**20 Jul 2020, 02:00**

Merchant ID  
**K500670**

**Customer**

**Shipping address** ^

**UK approved**  
34 Cavendish Square, Marylebone  
33 Cavendish Square  
London  
W1G 0PW  
GB  
Tel  
**+447911123456**  
Email  
**tester-uk-2@klarna.com**  
[Edit shipping address](#)

**Billing address** v

**Additional Info** v

**Order lines (2)**

[Refund](#)

Qty	Item	Reference	Unit price	Discount	Tax	Amount
1	Копринена Tie	793775370033	19.19	0.00	20% 3.20	19.19
1	Free Shipping	GBPFree	0.00	0.00	0% 0.00	0.00

**PAYMENT DETAILS**

Initial Payment Method

Statement

[Resend statement](#)

ORDER TOTAL		£19.19
Captured		£19.19
Refunded		£0.00
Not Captured		£0.00
<b>CUSTOMER BILLED</b>		<b>£19.19</b>

**Activity Log**

- 26 Jun 2020  
a few seconds ago  
13:58  
**Order customer details updated**  
By vivek.poulose via Merchant Portal
- 22 Jun 2020  
4 days ago  
11:27  
**Captured: £19.19**  
Via API
- 11:27  
**Order placed: £19.19**

## 2. Refuse Klarna Payments as a payment method

Upon selecting 'Klarna' as the payment method on Billing page of checkout process and based on the information provided Klarna Payments can be refused as a payment method.



SELECT PAYMENT METHOD

REQUIRED

☐ Credit Card
☐ Pay Pal
☐ Bill Me Later

☒ Buy now, pay later
☐ Buy now, pay later

Klarna.

Klarna.

Option not available

TESTDRIVE

Unfortunately this option is not available. Please choose a different payment method.

CONTINUE TO PLACE ORDER >

3. Klarna Payments not available – if Klarna API is not available, Klarna is not presented as a payment option

4. Order accepted after review - Instead of immediately accepting the order, it may happen that Klarna flags this transaction for additional review and accept it later on. The order is then created in the SCC system:

SiteGenesis

Merchant Tools

Administration

Storefront

Home

Users

Merchant Tools

Orders

Orders

Orders

You are viewing our new Search results.

This page allows you to search for orders by order number. Select Advanced to use more search options. Select By Number to search by providing a list of order numbers. Order numbers can be separated by either "-" or "+" or space or newline. Entered text is treated as case-sensitive, substring matching is not supported.

Order Search

Order Number:

Find

Number	Order Date	Site	Created By	Registration Status	Customer	Email	Total	Status
0000099	4/5/17 1:41:58 pm Etc/UTC	SiteGenesis	Customer	Registered	Gaydarzhiev	ted.smith+pend+accept-02@example.com	\$602.67	Created
0000098	4/5/17 1:31:20 pm Etc/UTC	SiteGenesis	Customer	Registered	Gaydarzhiev	ted.smith+pend+accept-02@example.com	\$110.24	Open
0000097	4/5/17 12:59:31 pm Etc/UTC	SiteGenesis	Customer	Unregistered	Alexander Gaydarzhiev	alexander.gaydarzhiev@zaurle.com	\$303.43	Open

This order is marked with EXPORT\_STATUS\_NOTEXPORTED, confirmation status NOTCONFIRMED and NOTPAID.

Merchant Tools

Administration

Storefront

Merchant Tools

Orders

Order - 0000099 (SiteGenesis)

General

Attributes

Payment

Notes

History

Details for Order '0000099'

Information:

Contains 3 line items to 1 shipping location The total price is \$602.67

Date Received:

4/5/17 1:41:58 pm Etc/UTC

Site:

SiteGenesis

Created By:

Customer

Customer:

Gaydarzhiev

Customer No.:

00000001

IP Address:

31.13.219.164

Email:

ted.smith+pend+accept-02@example.com

Phone:

3333333333

Order Status:

Created

Shipping Status:

Not Shipped

Confirmation Status:

Not Confirmed

Export Status:

Not Exported

Shipment

Qty	Product ID	Name	Manufacturer	Tax Rate	Unit Sales Price	Tax Totals	Item Total
2	701642967005	Catball Jacket		6.00 %	\$110.00	\$2.21 06	\$221.06
2	701642967008	Pull On Pant		6.00 %	\$90.00	\$1.08 00	\$180.00
2	740307307531	Zanica		6.00 %	\$95.00	\$1.14 00	\$185.99
Shipment Shipping Cost							\$15.99
Total Shipping Cost (00%)							\$15.99
Shipping Total:							\$15.99
Tax Totals							\$28.79
Totals:							\$602.67

Send Email

Print Order

Merchant Tools Administration Dashboard

Merchant Data > Orders > Order #00000909 (View Order)

General Attributes Payment Notes History

Details for Order '00000909'

Information: Contains 3 new items to 1 shipping location The total price is \$662.67

Date Received: 4/5/17 1:41:08 pm EST/UTC

Site: SiteCommerce

Created By: Customer

Customer: Gayden@Slev

Customer No.: 00000001

IP Address: 31.13.219.164

Email: ted.smith+pend-accept-02@example.com

Phone: 3333333333

Order Status: Open Confirmation Status: Confirmed

Shipping Status: Not Shipped Export Status: Ready for Export

Qty	Product ID	Name	Manufacturer	Tax Rate	Unit Sales Price	Tax Basis	Item Total
2	701642853055	Quilted Jacket		5.00 %	\$110.99	\$221.98	\$221.98
2	701642867098	Pull On Pant		5.00 %	\$69.00	\$138.00	\$138.00
2	746157357331	Zenith		5.00 %	\$59.99	\$119.98	\$119.98
						Shipment Shipping Cost	\$15.99
						Total Shipping Cost (001)	\$15.99
						Shipping Total:	\$15.99
						Tax Total:	\$26.70
						Total:	\$662.67

Save Order Print Order

In Klarna's Merchant Portal (Orders) the order is marked as accepted after review:

**Klarna. Merchant Portal**

[Overview](#) > Order #DR7XBFL5

**#DR7XBFL5** Uncaptured **US\$37.09**

Merchant reference 1: **00045902** [Edit](#)

Merchant reference 2: **a2c79d78ca1b673d64f4163557** [Edit](#)

Created: **26 Jun 2020, 14:18**

Expires: **24 Jul 2020, 02:00**

Merchant ID: **N100141**

Customer

Shipping address

US-norm Person

629 N High St

Columbus

43215 OH

US

Tel: **+491607891234**

Email: **tester+pend-accept-05@klarna.com**

[Edit shipping address](#)

Billing address

Additional Info

Order lines (4)

[Capture](#) [Edit order lines](#) [Cancel order](#)

Qty	Item	Reference	Unit price	Discount	Tax	Amount
1	\$50 Fixed Products Amount Above 100	\$50FixedProductsAmountAbove100	-475.00	0.00	0% 0.00	-475.00
1	Woven Trimmed Cardigan.	70164439173 7M	500.00	0.00	0% 0.00	500.00
1	2-Day Express	002	9.99	0.00	0% 0.00	9.99
1	Sales Tax	Sales Tax	2.10	0.00	0% 0.00	2.10

**PAYMENT DETAILS**

Initial Payment Method

Pay later in parts

VISA \*\*\*\*\*1111

**ORDER TOTAL** **US\$37.09**

Captured: **US\$0.00**

Refunded: **US\$0.00**

Not Captured: **US\$37.09**

**CUSTOMER BILLED** **US\$0.00**

**Activity Log**

26 Jun 2020 6 minutes ago

14:21 Order accepted after review By Klarna

14:18 Order placed: US\$37.09 By Klarna

5. Order rejected after review - Instead of immediately accepting the order, it may happen that Klarna flags this transaction for additional review and reject it later on. The order is then created in the SCC system:



Klarna. Merchant Portal

[Overview](#) > Order #19FTMKQV

**#19FTMKQV** Uncaptured US\$329.65

Merchant reference 1 00045903 <a href="#">Edit</a>	Merchant reference 2 ff35a31ea398fddcfd857ef0ba <a href="#">Edit</a>	Created 26 Jun 2020, 14:27	Expires 24 Jul 2020, 02:00	Merchant ID N100141
--	--	-------------------------------	-------------------------------	------------------------

Rejected after review

**Customer**  
**Shipping address**  
 US-norm Approved  
 629 N High St  
 Columbus  
 43215 OH  
 US  
 Tel  
 +491607898065  
 Email  
 tester+pend-reject-05@klarna.com  
[Edit shipping address](#)

**Order lines (3)**  

☐ Qty
 ☐ Item
 ☐ Reference
 ☐ Unit price
 ☐ Discount
 ☐ Tax
 ☐ Amount

<input type="checkbox"/> 1	Casual Spring Easy Jacket	8833605242 52M	295.00	0.00	0% 0.00	295.00
<input type="checkbox"/> 1	2-Day Express	002	15.99	0.00	0% 0.00	15.99
<input type="checkbox"/> 1	Sales Tax	Sales Tax	18.66	0.00	0% 0.00	18.66

**PAYMENT DETAILS**  
 Initial Payment Method  
 Pay later in parts  
 VISA \*\*\*\*\*1111

**ORDER TOTAL** US\$329.65  
 Captured US\$0.00  
 Refunded US\$0.00  
 Not Captured US\$329.65  
**CUSTOMER BILLED** US\$0.00

**Activity Log**  

26 Jun 2020

14 minutes ago

14:30

Order rejected after review

By Klarna

14:27

Order placed: US\$329.65

By Klarna

6. Klarna Payment option not available for the current purchase – the customer is presented with an appropriate message when attempt to choose Klarna is made.

**SELECT PAYMENT METHOD** • REQUIRED

☐ Credit Card
 ☐ Pay Pal
 ☐ Bill Me Later

☒ Buy now, pay later  

Klarna.

☐ Buy now, pay later  

Klarna.

**Option not available**

TESTDRIVE

Unfortunately this option is not available. Please choose a different payment method.

CONTINUE TO PLACE ORDER >

## 7. Gift Certificate / Klarna Payments split payment

Use Klarna Payments in combination with OOTB Site Genesis Gift Certificate functionality. When a gift certificate(s) is(are) redeemed in the checkout flow and remaining order amount is paid with Klarna Payments an order line item(s) 'Gift Certificate' is added to the Klarna Order reducing the amount owned with the gift certificate Payment Instrument(s).

## 8. Virtual Card settlements as an alternative to standard order management

The virtual card solution can be enabled through the site preference as shown below:

The screenshot shows the 'Site Preferences' page in the Klarna Merchant Tools interface. The page has a header with the Salesforce logo, 'Sandbox - klarna01 SiteGenesisGlobal', and navigation tabs for 'Merchant Tools', 'Administration', and 'Storefront'. A table lists various site preferences. The 'Virtual Card Network Enabled' preference is highlighted with a red border. It is currently set to 'No'.

Name	Value	Default Value	
Secondary Text Color Preference	CSS hex color to be used in Klarna Payments iFrame	#333333	<a href="#">Edit Across Sites</a>
Border Radius Preference	30 Size (in pixels) of the border radius to be used in Klarna Payments iFrame	5px	<a href="#">Edit Across Sites</a>
Attachments	Yes Flag to switch on/off the using of attachments when creating a session	No	<a href="#">Edit Across Sites</a>
Not available message on billing page	<pre>{   "GB": "Klarna Payment not available",   "FR": "String in French",   "AT": "Klarna Rechnung nicht verfügbar",   "default": "Klarna Payment not available" }</pre> The Klarna Payments not available message on billing page. JSON string holding co...		<a href="#">Edit Across Sites</a>
Virtual Card Network Enabled	Yes If set to true SFCC will create Virtual Card Network settlement from every Klarna or...	No	<a href="#">Edit Across Sites</a>

This option is disabled by default. However, if standard order management is not a reasonable option for a Klarna integration, instead Klarna's virtual card solution (MCS), can be enabled. When a customer places an order, the order is first booked in SFCC. If the order has a `fraud_status` of `PENDING`, action is not taken on the order until receiving Klarna's push notification that the `fraud_status` has changed to `FRAUD_RISK_ACCEPTED`. Once an order has been accepted by Klarna, the merchant platform creates a virtual card settlement, per the merchant card services (MCS) API.

Once a settlement has been created, the merchant platform can authorize the virtual card until the Klarna order is valid. Then, once the order has been fulfilled, the card funds should be captured. (For delays in capture, or other special use cases, please speak with the Klarna key account manager in advance) While Klarna is the original payment method of the order, the order will be settled with a credit card instead of direct bank account transfer. You can find more information [here](#) around other use cases. To utilize virtual card integration option the merchant should:

- Enable VCN option in Site Preferences as shown above
- Enter the VCN Public Key ID. Unique UUIDv4 value, which is different for playground testing and Production (live site)

Name	Value	Default Value
	The Klarna Payments not available message on billing page. JSON string holdin...	
Virtual Card Network Enabled	Yes If set to true SFCC will create Virtual Card Network settlement from every Klarn...	No <a href="#">Edit Across Sites</a>
VCN Public Key ID	6c5b99c5-b0de-4689-b569-1ae12ec898eb Unique identifier for the public key used for encryption of the card data	<a href="#">Edit Across Sites</a>
VCN Private Key	MIUuKQjBAACagEAtirQO1705I0J5GLAgvncKeyK3V3eXZMxjH+gO5X5SAG K51a Y6mpACVNsLjYRrh4K2QV1C5n8vm1Dz56bUjNKfYyuWSgEusn1ewYKp20 audwj yJYWBWkQc50aHTYYAleKUC5wAh5ix5c+o25X6wx88M3Rv57mQra4sWM Geu1z9Z bpbUOTK1738Is3/6ccC09usU2Z9FC4r2RIUGD8kyV7TVvmbDSIC8JelpKbZl 3w OB1FDw08uNIE+TJYUUKXpCnw8y5cR+Xc3gPkL8y6M7RQOKL1L1S18LKRE IKCK LMk3rbaNTbPN+1N5LqxrDjrmh2HnaDFkKbInHqZn93Y57ZfZZaHSF02O6 Your 4096 bit RSA Private Key	<a href="#">Edit Across Sites</a>
	MIICjANBgqhkiG9wOBAQEFAAOCAg8AMIICGKCAgEAtirQO1705I0J5GLA...	

- Generate a 4096-bit RSA key pair. Set the custom preference 'vcnPublicKey' with the value of the public key without the header and footer lines (begin and end public key) and the custom preference 'vcnPrivateKey' with the value of the private key without the header and footer lines (begin and end private key). As shown below:

Name	Value	Default Value
	The Klarna Payments not available message on billing page. JSON string holding country code and cor...	
Virtual Card Network Enabled	Yes If set to true SFCC will create Virtual Card Network settlement from every Klarna order. Default is false	No <a href="#">Edit Across Sites</a>
VCN Private Key	V5evexzo3d5AqBAGU3qyPojz1+9D15a8UW2CABtq4z9g84ABZ0z0xX3q7W x1Pm2yQ5Q5R0g1B13p29Z0R77ygnQjg1Y0Bjc+nglQWpuafK3RIBQ0aC+val 9dnTLJMGghnhtunKaby1crovHtdqPpKvnoNAjYUqG4822xhmmkayEjAanIPa4 stDrVPEWPTLx5xcOCdc13khpK5nkgRvalEFpwKX7W7hK/205FayTnmrzXYBQ7 cdD/9d30a4nLb/mu+Tq67519s53Qq/EGeTfqpRvPQKyhUnKXCGWBMZLTwyIG 59eTFDKaloc5TgipW7bPua93wZ8eEbRbP4QEcgEaNNhQBeEI0acdBVMtdrEI crDaa8XOW1aj15d04NYCrajKcMAF/QdghQhQm+UC5Y64En/Q+DAVWhGwYpXO Rhc3aep5Fh3ayk2m7AD0XG+RnBRW4VVR9R9R4ajp7+Rcub0cdePERXGKX9 c/B3mZ8R+valF8yFV02HwermNK0FuDfcoak4MajcFF4eWQp9f2BajPT+ xAFP4QD1UqicpmuF5SwAwXo6LXUV7NTSD0KMLuTLtLTLtMLnO9+9jmNagWRP Trak71u4Kf0u18T4EmY31871n6EDN5a753uVv4Bn7h111u1C1B4bVuuu473uW87u+F4B Your 4096 bit RSA Private Key	<a href="#">Edit Across Sites</a>
VCN Public Key	MIICjANBgqhkiG9wOBAQEFAAOCAg8AMIICGKCAgEAnNYG712G8nZa+22oBYZk rY228iw3UE9W04oxfentkEdHn4k5SUL8KQ7H9NvdeKCN7TgYvMu/GNca18 nu2V/IGYDfK85h5V5EjOum+5A1uFEX61BX7nMDHc1KyWm9p2kbg88mF1X63 KV94Q0NEKcNaRDPYR-qz55+fadQZQ1stVNSdrocZD1+Lx8y9V+Bdm8K1E RLkN1u3kYmE4FkY+20b7TQkWW/SET120GQZsHozjgHjHjNvvebWHP xal4m1Z3/9yVsp7YANQbT5C1bRcGaapPn30CNaqUqapf5g8VY3E5CoXH Dd04Lx1qcuYIDhaDzey6W+b8m755xLi+rqQyM4PWL0Dm3FVd8+4YKILex 3AKBFclqCHMSOGaEeyXKTjAghr9R8PInQR/L440cHqzw2XODvpj5WcmUJ 0W4wUq5RN5sobrnVmoV6f1267Q/LP+H5+eowdahR5etIqIO+2PN0K415Vds /Pkz3FBwWk3mp2ONT2440+NilyRPFa1x7JAgcv2ZnFamY4QAgXLCpgEM Your 4096 bit RSA Public Key	<a href="#">Edit Across Sites</a>

## How to generate a 4096-bit RSA key pair:

In order to generate an RSA keypair with a 4096-bit private key you can use the following openssl command:

***openssl genpkey -algorithm RSA -out private\_key.pem -pkeyopt rsa\_keygen\_bits:4096***

In order to extract the public key from an RSA keypair, you can use the following openssl command:

***openssl rsa -pubout -in private\_key.pem -out public\_key.pem***

In the folder where you have executed the above commands two new files will be created - public\_key.pem and private\_key.pem.

The contents of the files should look something like:

#### **public\_key.pem**

```
-----BEGIN PUBLIC KEY-----
MIICljANBgkqhkiG9w0BAQEFAAOCAg8AMIICGgKCAgEAOiNYG7l2G8nZa+22oBYZk
tV228lw3UE9WO4oxfknJtKEdHn84x55ULT8KQTh9NVtdeKC8nTfTgyvMt/GNca18
xuZV/IGYDftKt85hbV5EjOum+StAlufEXvIBX7nMOMc1KyWm9kp2kbqd88mFIX63
KV94OoNEXcNatRDFYR+qz53+ifadDQtQ1sIVNStdrcZDJ1+LxtBy9V+BdmsBK1E
RLsKh/JLXyWE24FJKV+z00s7TQkdWW/5ET12OGQYZsWo1yqgi9HplNvrise8vWP
xaL4m8iZ3l/9yYdg7yANQbTxSJcbbRCgaaagPo30CNxeqU6qafY5g8vY3E52CoXH
DdO4UsIX1qcuYIDhqaDzey6W+b8m755xLi+rqQyM4PBWL0J0dM3FVid8+4YKILex
3AKBFciqRCMHsOGaEeyrXKTjIAsghr9RS8PifvQRrL440cHzqw2vX0DvpjSWcmUJ
tW4wUq5RNSsobrxnVmoV6fj1z67Q/1P+I5le+oowdahR5ztVqJlO+2PNoX4I5VDs
/Pkz3f8wWVc3Mp2oNT244o+/NliYRfPFaJjx7JAgrcvZt2nFAMy4QApXLFJCpgEM
wYucE4AH4gJKsh3KZbxRERrrO72bL2rxvWqBp/0h7DcMsV9sQs4BvxxlI6CF506F
ThzmclakLBAyd5LALiXiPfkCAwEAAQ==
-----END PUBLIC KEY-----
```

#### **private\_key.pem**

```
-----BEGIN PRIVATE KEY-----
MIIEJQIBADANBgkqhkiG9w0BAQEFAASCCSswggKnAgEAAoICAQCg1gbuXYbydlr7
bagFhmS1XbbyXDdQT1Y7ijF+Scm0oR0efzjHnlQu3wpBOH01W114oLydN9ODK8y3
8Y0JrXzG5IX+UZgN+0q3zmFtXkSM66b5K0Ai58Re+UFFucw4xzUrJab2SnaRup3z
yYUhfrcpX3g6g0Rdw1q1EMVhH6rPnf6J9p0NC1DWyVU1K12ugJkMnX4vG0HL1X4F
2awErUREuwqH8ktfJYtbgUkpX7PTSztNCR1Zb/kRPXY4ZBhmxajXKqCL0emU2+uK
y97y9Y/FovibyJncj/3Jh2DvIA1BtPFilxttEKBppqA+jfQl3F6pTqpp9jmDy9jc
TnYKhccN07hSyVfWpy5ggOGpoPN7Lpb5vybvnnEuL6upDIzg8FYvQnR0zcVWJ3z7
hgogt7HcAoEVYKpElwdl4ZoR7KtcpOOUCyCGv1FLw+J+9BGsvjjRwfOrDa9fQO+m
NJZyZQm1bjBSrIE1KyhuvGdWahXp+PXPrtd/U/6Xkh76ijB1qFHnO1WomU77Y82h
fgjIUoz8+TPd/zBVZvcynag1Pbjij780iLJF88VoknHskCCty9m3acUCZjhAClcs
UkKmAQzBi5wTgAfiAkqyHcplvFERGus7vZsvavG9aoGn/SHsNwyxX2xCzgG/HEiX
oiXnToVOHOZyVoosEDJ3ksAuJel9+QIDAQABAoICACRkaUsUNI22RB3yEPu3DiCP
pO6v+QAeA4gTW+GUdQR9dCZLaSCZ7bhxVVOuoX4qPzslO6hjUmOyzG6upFgVPk+P
HNQfyEUZoC148Eib9OziAXUN2URMpv1KbwVm+BO814X8zguai7uru0PHTG1oy677
4Ct1OknxAxxHQDlaxT6XJFo5SA4EinUfNz2Bo3/xry/QjxW/mCK0GwDd4PNp9TGM
FPTv2SgdSDOWzGQlOH5N3owuzMpl8NV6z74wv+i5Ptv41Dzu8WhyXpiYSsk00SRK
HPC68j2bAzTPghp5aSZ9976SGm2SPonJXyboXdiHbl/osdyqDxeIT3iB9GmrHX/i
kHPGJCh7fRZvqj39Hc+lxYjabwW3rDeDIPB7ab9z1KLF4z1D6AZOKCPyTaDRdQ1Q
eDi7LwDmk7NHEPrmF/nlcguQdqblbmFO2zEs0TOe6y4uBMndRsbQprTNSMUDBkrA
lNaYVSTQ1Z0Y/8DZDpGcyS1OnJv74F15uDjKN6/ov991mZ1JrZ+V2sdS3EDUlmvP
6thQKwI7Ln6h+ApHtWUG1NmvQe5gJE0qAeJ9b45clUzIRUwhVmEp8NoIjh0kAjaN
d4lk7xy9ZRDUY5yekPeYrJPSHjsHAYeoktJlJRufI2UUq3uxNjjiCoQcOVGFNDIS
YTTPwpu1pmC0C+rh2fgBAoIBAQRultRARvtc2JKhVOUyZk88zd9kvrI6fNiyKmi
HgiWf7qkTPD9xhOQWDw3iwrFQAD+YkgV5MCBO8wp8oO8GESOCI+XZWEExOcPT0Vfj
PZHiQrTFnlfG/+fAO14xLf3j3ED4YQXdHOKI3xoLknQx/EydLoctxgkpgWLRsA7
DwdSag1/0sBvaHY27ogAfdimHdaKZ5OAE4a9k1qP3xVZBuOe8Sd65unBavUJLDuv
ikeNmkSVgW1sm55/729Jlr63USHF76lt+vE1cdZ+vKg5vYotsQgPzvNBmUO/E8Gj
zMXQRfQfVdINEX0rCupTkW1G6AGTwQc/NPzyr/LTpLe6UBAoIBAQDEUjTiG11V
hf7WjdG3gctRlr+mYapQHgXdVLx2QSaQUYid+0QXK11YfJlsRB6nwa+OED83RfPO
-----END PRIVATE KEY-----
```

```

lIFqxpzudSLPmoDuIBT7DI5c/aleyKs/siUusP8QVDXk6OAR84XSytC35sIRV7pE
VMuBL91jfkQ0Lf/PreslK/kl6Yvwwp4qrHK6/f9TgciHclYtf+/oti4ky6GJgfmP
fmuCqjxmUKbXXFPd5RbL2THGOowilb8zDLjf3RljlQFqogAk6H9hp2V0VZLiJHp
UWM3z3zxDWeDaqJ08sHuk/rA9QpsVTu8IGTQsxdj8JwluN1Q+YZiOuPiSENBqPzT
V3exexzo3sD5AoIBAGU3qEyPoJz1+9D1SaI8LW2CABzlq4z9g84ABAZOsIx5q7W
x1PinZyDSQSRXg1B13jt29ZdIR79ygnQlg1YOBjcvtgVQHPuafk3RIBQbbCh+val
9dn/tUxMGqhnhunKaby1rovJHfdqnPpKwzNAjYUqaGkJ822xhmmke/fEyAanIPa4
stDRviPEWPTLx5xcOCdx13khpKSnkgRvaLEfpwkVX7Vr7hK/2OSFaYTNmrzXYBQ7
c6D/9d3Oo4nLb/mu+Tq67S19t53Qg/GEgTfkpuRoVPi0KyhUnKKCGWIBMZLTWylG
S9eTFDKoJ0cSTGipjW7bPua93wZ8eEbRABpf4QECggEANNhQBeEJ0aCdBVHtdrEI
crDaa8X0W1aJi5dol4hYCRajaKsfHAF/QfdgMQVxHwUC5YG4En/Q+DAVWhGWYpXD
RhC3zeFy5FVszyk0sx/fAOIKGvRn5BRW4YRR9GMRzbjsT+RcruBnckdE9ERXGpX9
c/JB3rxZBlit+oliFM8yfWktMwsrmNktFuDftvJeok4KejycFF4eWDqsf828xjPT+
xA/FP4CQD1UqkcpmuFSIwAwXo6LXVY7NTS0nKMiUnTLkLITIhtLnO9+9jmNapWRP
Tc+hZUuHklpI8DHFmX2j87LgkFD05eD5lynY4RgZtU1W1C1RdVYwoA72WB7knEaB
uQKCAQAH9s67P/7fFX9dfEans3PHU4nGjD8dJ8eoNQ6DhBMydZpGWI5ZUeEBZDRk
OcBOeRs5BOcS43Em9kETpzawyCwxmnwzl+CzoPzMqCtw9tXomF9HG6RJ9XBdJfGA
ALAwCd4bASxmFM6guSP5GKnZ9aY3tR3tWWDfr7f9z8wOewzzpPclwRh009fPe4TC
NXoEm1MELJVeUieDSLKZgigCw8WHGqLItONpAO/fwSM2glcxETVV7qx3aPuJzCVh
LQZoBLQk3UMKsWDdpzeBdiERe66NAGVk92Xe7SY9EY2vymaq761i1x1vlpT27qp
240LDJawqM0IraKmdCvWjofWSaOU
-----END PRIVATE KEY-----

```

- Finally, you need to send the generated key\_id/public key in JWK format to Klarna prior to testing/go-live. It will be used to encrypt the virtual card pan and csc on Klarna side when settlement request is made. After confirmation from Klarna that the key has been successfully added to your merchant profile you would be able to use virtual card-based settlement option for Klarna payment methods.

**Note:**

The recommend RSA keypair size of 4096 bits. This key pair must be associated with a unique key\_id (UUIDv4). The public key must be shared in JWK format with Klarna contact. Note that for production and playground, the Key id and corresponding keypair combination shared are different and must be configured prior to testing/go-live with the agreed virtual card product

- Authorize the virtual credit card

The virtual credit card details are stored as order level custom attributes, and can be accessed through the Order object as follows:

- Order.custom.kpVCNBrand – Virtual credit card brand;
- Order.custom.kpVCNHolder - Virtual credit card holder;
- Order.custom.kpVCNCardID- Virtual credit card ID (unique UUIDv4 value);



- Order.custom.kpVCNPCIData – Virtual credit card encrypted data;
- Order.custom.kpVCNIV – Virtual credit card initialization vector;
- Order.custom.kpVCNAESKey – Virtual credit card AES key;
- Order.custom.kpIsVCN – Boolean property indicating whether VCN is used.

Note: Please review the details in the order export XML prior to go-live. This is to ensure that card details are fetched from the correct custom attribute fields in your Orders XML file for post purchase order processing.

These attributes can be inspected in the BM order details in the attributes tab. Out of the box only the card ID is visible as shown below.

[Merchant Tools](#) > [Ordering](#) > [Orders](#) > Order: 00049309(RefArch)

General **Attributes** Payment Notes History

### Attributes for Order '00049309'

On this page you can edit the attributes of the order. Fields with a red asterisk (\*) are mandatory. Click **Apply** to save changes. Click **Reset** to revert your changes.

Klarna Payments	
<b>Klarna Payments Order ID:</b>	<input type="text" value="72bf2c96-6523-2add-8c50-f2af87712019"/>
<b>Is VCN Used:</b>	<input checked="" type="checkbox"/>
<b>VCN Card ID:</b>	<input type="text" value="befbb0e9-5e98-4e39-9c00-75aba0c3372b"/>


[<< Back to List](#)



If required, the additional virtual card details can be assigned to this group in Administration > Site Development > System Object Types > select “Order”. In the Attribute Grouping tab select Klarna\_Payments and click on “edit”. Assign the new attributes and save the data.

## Object Type 'Order' - Attribute Definition Assignments

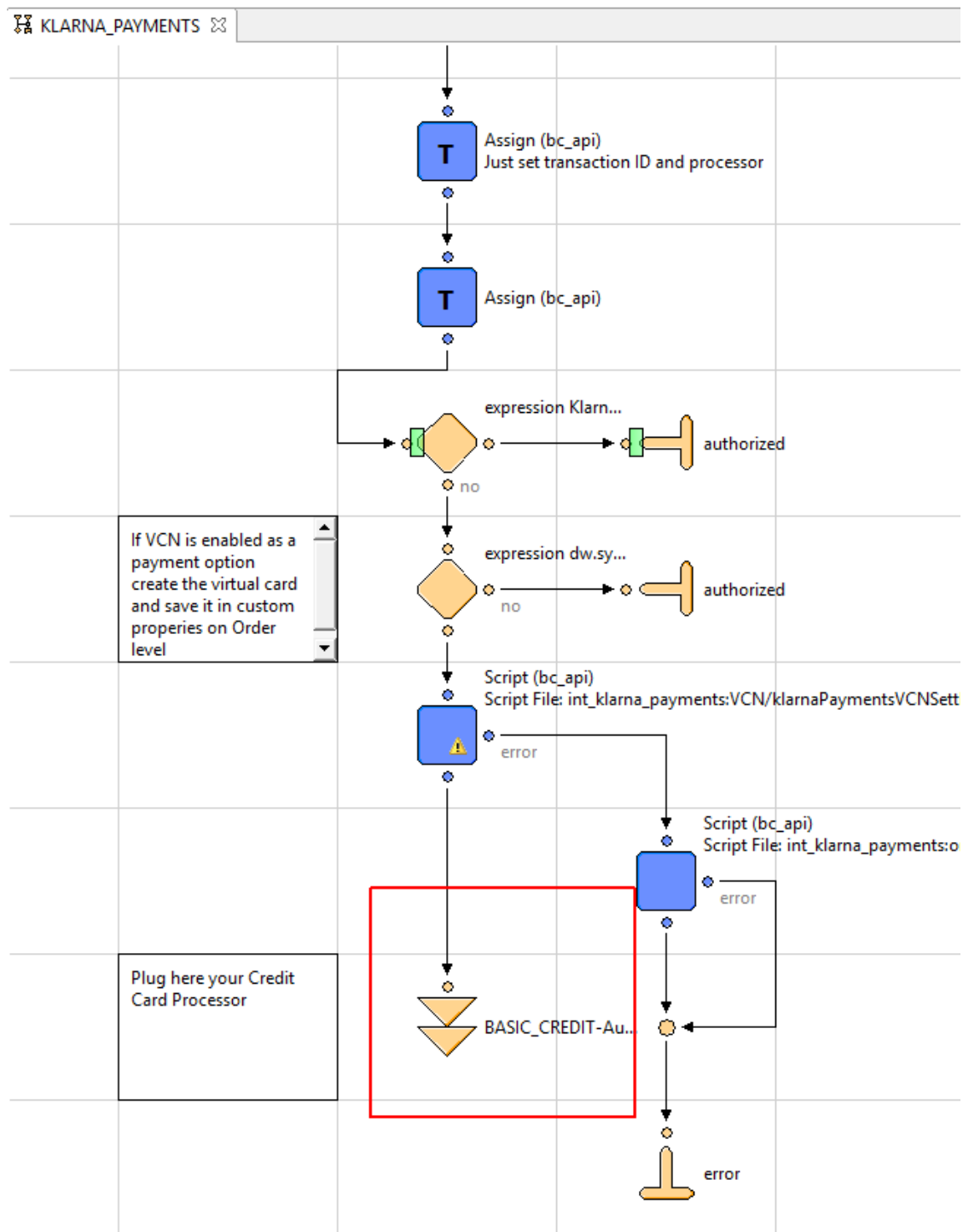
On this page you can assign existing attribute definitions to your attribute group.

Assign Attribute Definition

ID: \*   ...

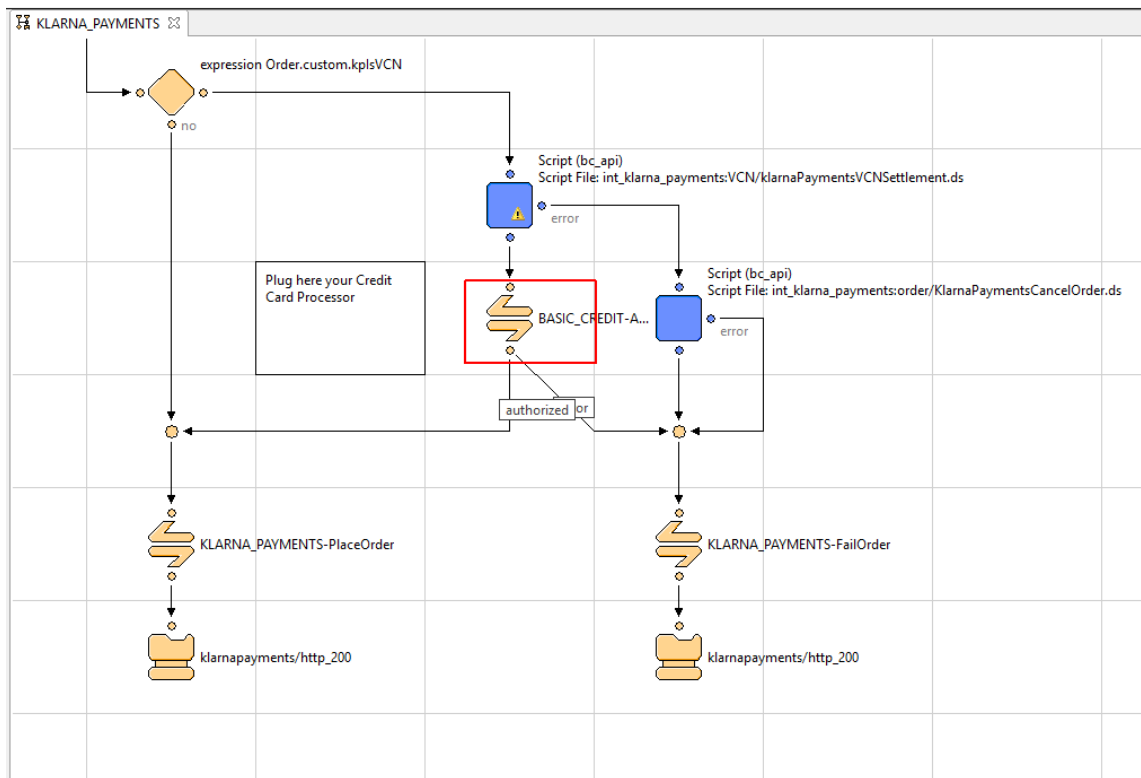
<a href="#">Select All</a>	ID	Name	Type	Attribute Settings	Sorting
<input type="checkbox"/>	<b>kpOrderID</b>	Klarna Payments Order ID	String		
<input type="checkbox"/>	<b>kpIsVCN</b>	Is VCN Used	Boolean		
<input type="checkbox"/>	<b>kpVCNCardID</b>	VCN Card ID	String		
<input type="checkbox"/>	<b>kpVCNHolder</b>	VCN Holder	String		
<input type="checkbox"/>	<b>kpVCNBrand</b>	VCN Brand	String		
<input type="checkbox"/>	<b>kpVCNPCIData</b>	VCN PCI Data	String		
<input type="checkbox"/>	<b>kpVCNIV</b>	VCN Initialization Vector	String		
<input type="checkbox"/>	<b>kpVCNAESKey</b>	VCN AES Key	Text		

It is up to the merchant to use the credit card details as described above to authorize the virtual credit card. As an example, authorization is done with calling BASIC\_CREDIT-Authorize as below (pipeline version):



You need to add your own logic in KLARNA\_PAYMENTS-Authorize in the place of BASIC\_CREDIT-Authorize call node.

You would also need to add your own card authorization login in the place of BASIC\_CREDIT-Authorize call node in the KLARNA\_PAYMENTS-Notification pipeline as shown below:



## 9. Configure On-Site Messaging

Klarna On-Site Messaging (OSM) is configured per-site per-locale via the **KlarnaCountries** custom object. To configure the OSM settings for a locale, you must visit “*Merchant Tools – Custom Object Editor*” and search for **KlarnaCountries** custom object. Provide a valid locale for the OSM tag based on the country being configured. The OSM Data Client ID and Data keys required are available in Klarna Merchant Portal – On-site Messaging App

Merchant Tools > Custom Objects > Custom Objects > US - General

General

### Manage 'US' (KlarnaCountries)

Fields with a red asterisk (\*) are mandatory. You can view and edit the name and description in other languages, if required. Click [Apply](#) to save the details.

custom	
Country Code*	US
On-site Messaging Data Locale:	en-US
Service Credential ID:	klarna.http.uscredentials
On-site messaging Data Client ID:	60a22a39-c2fd-5d09-bfe1-771459318a4d
Cart Placement Tag Enabled:	<input checked="" type="checkbox"/>
Cart Placement Data Key:	credit-promotion-standard
PDP Placement Tag Enabled:	<input checked="" type="checkbox"/>
PDP Placement Data Key:	credit-promotion-standard
Library URL:	https://na-library.playground.klarnaservices.com/lib.js

[<< Back to List](#)

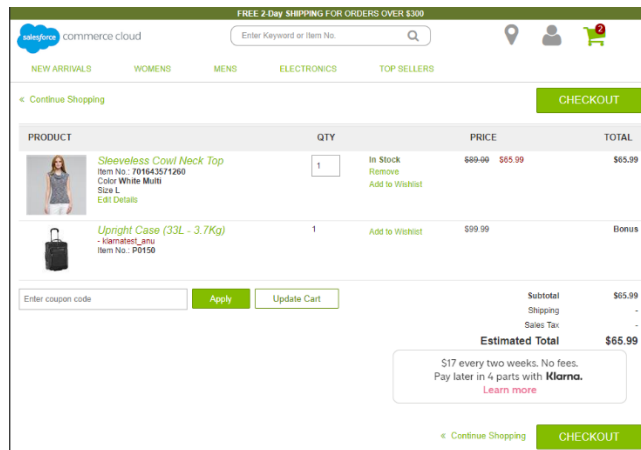
To enable Placement tag for the Cart Page, the “*Cart Placement Data Key*” must be filled with Data Key value and the “*Cart Placement Tag Enabled*” must be checked

To enable Placement tag for the PDP Page, the “*PDP Placement Data Key*” must be filled with Data Key value and the “*PDP Placement Tag Enabled*” must be checked

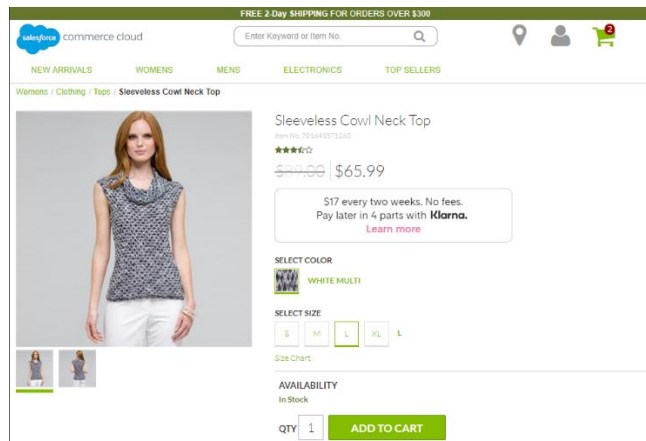
In Library URL, please input the full URL to the On-Site Messaging JavaScript Library.

In On-site messaging Data Client ID, please input the On-Site Messaging Data client ID value

On-site messaging Data locale, please input the valid On-Site Messaging Data locale



Klarna On-Site Messaging Enabled on Cart Page



Klarna On-Site Messaging Enabled on PDP Page

For more information regarding customizations and best practices, please refer to the Klarna Developer: <https://developers.klarna.com/documentation/on-site-messaging/customization/>

## 10.Configure Auto-capture

When the preference is enabled (disabled by default), a full amount capture is attempted prior to acknowledging the order with Klarna. If the capture is successful, the SFCC order's payment transaction is marked as Paid, and viewable in the Business Manager.

[Merchant Tools](#) > [Ordering](#) > [Orders](#) > Order: 00029204(RefArchGlobal)

[General](#) [Attributes](#) [Payment](#) [Notes](#) [History](#)

### Payment Information for Order '00029204'

Order Total:	£199.98
<a href="#">Amount Paid:</a>	£0.00
Balance Due:	£199.98
Invoice Number:	00119002
<a href="#">Payment Status:</a>	Paid
Payment Method:	KLARNA_PAYMENTS Processor: KLARNA_PAYMENTS Transaction: 91ab359d-ada0-71e0-834e-6eb59b83bf9f Amount: £199.98  Klarna Payment Category ID: pay_later Klarna Payment Category Name: Pay later. Fraud Status: ACCEPTED

The order will be marked as “*Captured*” in the Klarna’s Merchant Dashboard.

Merchant reference 1  
**00029205**  
[Edit](#)

Klarna reference  
**BJJGH9P1**

Created  
Jun 25, 2019, 5:00 PM

Expires  
Jul 23, 2019, 3:00 AM

Merchant ID  
**K500726**

**Customer**  
Shipping address  
Angela Gill  
4939 Wyatt Street  
West Palm Beach  
SW42 4RG Florida  
GB  
Tel  
01222 555 555  
Email  
ivan.zanev@tryzens.com  
[Edit shipping address](#)  
Billing address  
Additional Info

**Order lines (2)**  
[Refund](#)

Qty	Item	Reference	Unit price	Discount	Tax	Amount
1	Black Flat Front Wool Suit	7505187030...	191.99	0.00	5% 9.14	191.99
1	Наземен транспорт	GBPO01	7.99	0.00	5% 0.38	7.99

**PAYMENT DETAILS**  
Initial Payment Method  
Statement  
[Resend statement](#)

Currency: GBP  
**ORDER TOTAL**  
Captured  
Refunded  
Not Captured  
**CUSTOMER BILLED**

**£199.98**  
**£199.98**  
**£0.00**  
**£0.00**  
**£199.98**

**Activity Log**

Jun 25, 2019  
9 minutes ago

5:00 PM  
Order acknowledged  
Via API

5:00 PM  
Captured: £199.98  
Via API

5:00 PM  
Order placed: £199.98

If the capture is unsuccessful, an error will be logged in the custom error log.

## 2.3 Limitations, Constraints

---

The merchant will need a configured Klarna Payments account/MID for playground and production. Note: The later requires a contract with Klarna

The merchant can style the Klarna Payments widget (skin), to make it matching the look and feel of their store. The list with the graphic elements that can be customized, through site preferences is as follows:

- "color\_details" (site preference kpColorDetails): "#COFFEE"
- "color\_button" (site preference kpColorButton): "#COFFEE"
- "color\_button\_text" (site preference kpColorButtonText): "#COFFEE"
- "color\_checkbox" (site preference kpColorCheckbox): "#COFFEE"
- "color\_checkbox\_checkmark" (site preference kpCheckboxCheckmark): "#COFFEE"
- "color\_header"(site preference kpColorHeader): "#COFFEE"
- "color\_link"(site preference kpColorLink): "#COFFEE"
- "color\_border"(site preference kpColorBorder): "#COFFEE"
- "color\_border\_selected"(site preference kpBorderSelected): "#COFFEE"
- "color\_text"(site preference kpColorText): "#COFFEE"
- "color\_text\_secondary"(site preference kpColorTextSecondary): "#COFFEE"
- "radius\_border"(site preference kpRadiusBorder): "0px"

Successful payment authorizations (Klarna Payments status APPROVED) lead to a 'Paid' order payment status and 'Ready for Export' export status

Refused and pending payment authorizations (Klarna Payments statuses REJECTED and PENDING) lead to a 'Not Paid' order payment status, and 'Not Exported' export status

Cancels and refunds lead to a 'Not Paid' order payment status (even if the status was 'Paid' before), and 'Not Exported' export status

The cartridge makes use of the Klarna Payments JSON REST API and a JavaScript SDK.

## 2.4 Compatibility

---

Compatible with Commerce Cloud 18.1.0 and SiteGenesis 105.0.0

## 2.5 Privacy, Payment Data

---

1. The cartridge is compliant with GDPR recommendation and follows the best practice mentioned [here](#) and implementation follows practice mentioned [here](#)
2. EMD (Extra Merchant Data): The cartridge supports sending additional information on the customer's past purchase history when turned on in custom preferences: Attachments(kpAttachments). The type of data that can be send as an attachment is mentioned here. EMD is required for certain types of merchant order and inclusion of EMD is (e.g.customer\_account\_info: past purchase with merchant store) generally beneficial to improve acceptance rates. EMD is included as part of Authorization step in Commerce Cloud checkout. The data send to Klarna is customizable through a hook. An example hook contents are provided in int\_klarna\_payments/scripts/EMD/KlarnaPaymentsBuildEMD\_hooks.js. The merchant can customize the contents of the BuildEMD function. Current Basket or Order object can be accessed thorough args. LineltemCtnr. This script should return a JSON string to be used as a value for the body sub-field of the attachment field as [described here](#) . If the example KlarnaPaymentsBuildEMD\_hooks.js file is used unchanged the data send to Klarna is by the following schema:

```
{
  "$schema": "http://json-schema.org/draft-03/schema#",
  "id": "http://klarna.com/v2/emd#",
  "description": "Extended Merchant Data Payload Schema",
  "type": "object",
  "properties": {
    "customer_account_info": {
      "type": "array",
      "items": {
```



```

        "type": "object",
        "properties": {
            "unique_account_identifier": {
                "type": "string",
                "maxLength": 24
            },
            "account_registration_date": {
                "description": "ISO 8601 e.g. 2012-11-24T15:00",
                "type": "string",
                "format": "date-time",
                "pattern": "^[0-9][0-9][0-9][0-9]-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9](:[0-5][0-9]){0,1}Z{0,1}$"
            },
            "account_last_modified": {
                "description": "ISO 8601 e.g. 2012-11-24T15:00",
                "type": "string",
                "format": "date-time",
                "pattern": "^[0-9][0-9][0-9][0-9]-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9](:[0-5][0-9]){0,1}Z{0,1}$"
            }
        }
    },
    "payment_history_full": {
        "type": "array",
        "items": {
            "type": "object",

```

```

"additionalProperties": false,
"properties": {
  "unique_account_identifier": {
    "type": "string"
  },
  "payment_option": {
    "type": "string",
    "enum": ["card", "direct banking", "non klarna credit",
"sms", "other"]
  },
  "number_paid_purchases": {
    "type": "integer"
  },
  "total_amount_paid_purchases": {
    "type": "number"
  },
  "date_of_last_paid_purchase": {
    "description": "ISO 8601 e.g. 2012-11-24T15:00",
    "type": "string",
    "format": "date-time",
    "pattern": "^([0-9][0-9][0-9][0-9]-[0-1][0-9]-[0-3][0-9]T[0-2][0-
9]:[0-5][0-9])(:[0-5][0-9]){0,1}Z{0,1}$"
  },
  "date_of_first_paid_purchase": {
    "description": "ISO 8601 e.g. 2012-11-24T15:00",
    "type": "string",
    "format": "date-time",

```



### 3. PCI-DSS compliance

**Important Note: DO NOT SAVE DECRYPTED PCI DATA ON THE SERVER!**

The virtual card (MCSv3) solution enables settlements using individual virtual card issued against a Klarna order. To be compliant with PCI-DSS requirements, merchant must ensure the data is securely maintained and transmitted as part of their operation in their live shop environment. This maybe be done in consultation with your payment service provider/acquirer and completed prior to go-live. Please review in advance the order export details required for virtual card-based Klarna orders. Any historical decrypted pci data should also be expunged, regardless of their validity date

### 3. Implementation Guide

To setup and implement Klarna Payments Salesforce Commerce Cloud Cartridge, the following steps must be followed:

#### 3.1 Setup

---

Klarna Payments integration has three cartridges:

- `int_klarna_payments` which implements the core storefront functionality;
- `int_klarna_payments_pipelines` which implements the storefront functionality with pipeline based approach;
- `int_klarna_payments_controllers` which implements the storefront functionality with controllers based approach;

##### 3.1.1 *int\_klarna\_payments*

---

The `int_klarna_payments` cartridge has the following components:

Cartridge name

- `int_klarna_payments`

Scripts

- `/checkout`
  - `SetPendingOrderStatus.ds`
- `/common`
  - `KlarnaPaymentsApiContext.js`
  - `KlarnaPaymentsHttpService.ds`
- `/EMD`
  - `KlarnaPaymentsBuildEMD_hooks.js`
- `/locale`
  - `KlarnaPaymentsGetCountryFromLocale.ds`
  - `KlarnaPaymentsGetLocale.ds`
- `/order`
  - `KlarnaPaymentsAcknowledgeOrder.ds`
  - `KlarnaPaymentsCancelOrder.ds`
  - `KlarnaPaymentsCreateOrder.ds`
  - `KlarnaPaymentsOrderModel.js`
  - `KlarnaPaymentsOrderRequestBuilder.js`
- `/session`
  - `KlarnaPaymentsCreateSession.ds`
  - `KlarnaPaymentsSessionModel.js`
  - `KlarnaPaymentsSessionRequestBuilder.js`
  - `klarnaPaymentsUpdateSession.ds`
- `/util`
  - `Builder.js`
  - `KlarnaPaymentsConstants.js`
- `/VCN`
  - `klarnaPaymentsVCNSettlement.ds`
- `createKlarnaPaymentInstrument.ds`
- `hooks.json`
- `klarnaRemovePreviousPI.ds`

#### *Static*

- */default*
  - *Js*
    - *klarna-payments.js*

#### *Templates*

- */default*
  - *klarnapayments*
    - *http\_200.isml*
    - *modules.isml*
    - *paymentmethodshelper.isml*
    - *shippingaddresshelper.isml*
    - *klarnapaymentblock.isml*
    - *klarnapaymentscategories.isml*
- */resources*
  - *klarnapaymentsresources.isml*

### 3.1.2 **int\_klarna\_payments\_pipelines**

---

The `int_klarna_payments_pipelines` cartridge has the following components:

#### *Cartridge name*

- `int_klarna_payments_pipelines`

#### *Pipelines*

- `KLARNA_PAYMENTS.xml`

### 3.1.3 **int\_klarna\_payments\_controllers**

---

The `int_klarna_payments_controllers` cartridge has the following components:

#### *Cartridge name*

- `int_klarna_payments_controllers`

#### *Controllers*

- `KLARNA_PAYMENTS.js`

#### *Scripts*

- `hooks.json`

#### *Other*

- `package.json`

If the base cartridges are named differently the variables shown above need to be renamed to the correct paths based on the individual integration.

## 3.2 Configuration

### 3.2.1 Metadata Import

Go to 'metadata' folder, review and edit, if needed, the site-template contents. (Site template is prepared to setup SiteGenesis and RefArch sites - you may want to change that to your actual sites and delete the ones that are not needed). Zip the directory and you'll have 'site-template.zip' installation package. Import it through BM Administration > Site Development > Site Import & Export section.

The following configuration items will be added after you import 'site-template.zip' archive:

The screenshot shows the 'Klarna Payments' configuration page in the SiteGenesis administration interface. The page is titled 'Klarna Payments' and has a breadcrumb trail: 'Merchant Tools / Site Preferences / Custom Site Preference Groups'. There are buttons for 'Cancel', 'Apply to Other Sites', and 'Save'. The 'Instance Type' is set to 'Sandbox'. Below this is a search bar 'Search by IDs...'. The main content area is a table with columns 'Name', 'Value', and 'Default Value'. The table contains three rows: 1. 'Send product\_url and image\_url' with a value of 'None' and a description: 'If set to true product\_url and image\_url fields in Klarna Payments create session call will be included in API call, otherwise product\_url and image\_url fields will not be populated'. 2. 'merchant\_reference2 mapping' with a value of 'The field from SCC order (basket) object that is mapped to merchant\_reference2 field from klarna API request'. 3. 'Pre-Assessment is on for:' with a value of 'Pre-assessment flag. Comma separated string values with country codes, which should have the pre-assessment flag ON. For example the following flag "UK" "CN"'. Each row has an 'Edit Across Sites' link.

Name	Value	Default Value
Send product_url and image_url	None	
merchant_reference2 mapping	The field from SCC order (basket) object that is mapped to merchant_reference2 field from klarna API request	
Pre-Assessment is on for:	Pre-assessment flag. Comma separated string values with country codes, which should have the pre-assessment flag ON. For example the following flag "UK" "CN"	

See full list below.

Besides configuration parameters, the following order attributes will be added:

The screenshot shows the 'Custom Object Definition' page in the SiteGenesis administration interface. The page is titled 'Object Type 'Order'' and has a breadcrumb trail: 'Administration / Site Development / Custom Object Types / Order - Attribute Definitions'. There are buttons for 'General', 'Attribute Definitions', and 'Attribute Grouping'. Below this is a search bar 'Search Attribute Definitions'. The main content area is a table with columns 'ID or Name', 'ID', 'Name', 'Type', 'Attribute Settings', 'Values', and 'Edit'. The table contains one row: 'klarna\_order\_id' with a value of 'Klarna Payments Order ID'. Below the table is a 'Show 1 - 1 of 1 items' link and a 'Back to List' button.

ID or Name	ID	Name	Type	Attribute Settings	Values	Edit
klarna_order_id		Klarna Payments Order ID	String			

Custom Object Definition:

After importing metadata there will be a site specific custom object to store Klarna countries:

## Object Type 'KlarnaCountries'

This page lists the attribute definitions of your object type. Use the search to find attribute definitions by ID and name.  
Click New to create new attribute definitions. Click Delete to delete existing attribute definitions.

Search Attribute Definitions						
ID or Name:		<input type="text"/>	<input type="button" value="Find"/>			
Select All	ID	Name	Type	Attribute Settings	Values	
<input type="checkbox"/>	<a href="#">UUID</a>	UUID	String	*	0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">country</a>	Country Code	String	*	0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">creationDate</a>	Creation Date	Date+Time	*	0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">credentialID</a>	Service Credential ID	String		0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">klarnaLocale</a>	Klarna Locale	String		0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">lastModified</a>	Last Modified	Date+Time	*	0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">osmCartEnabled</a>	Cart Placement Tag Enabled	Boolean		0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">osmCartTagId</a>	Cart Placement Tag ID	String		0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">osmLibraryUrl</a>	Library URL	String		0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">osmPDPEnabled</a>	PDP Placement Tag Enabled	Boolean		0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">osmPDPTagId</a>	PDP Placement Tag ID	String		0	<a href="#">Edit</a>
<input type="checkbox"/>	<a href="#">osmUCI</a>	On-site messaging UCI	String		0	<a href="#">Edit</a>

Note: The same custom object is used by Klarna Checkout

The cartridge is selecting the object based on the request locale. E.g. site with locale “de\_DE” will use the “DE” custom object. Even if you have locales that are not supported by Klarna Payments we recommend to make a corresponding entry in the custom object for that locale (you can use one of the other endpoints). Thus, on the billing page of the unsupported locale you will have the Klarna Payments widget showing an appropriate message

### 3.2.2 Cartridge Path

If using the controller based integration, cartridge path should be setup as such under Sites / Manage Sites /

app\_storefront\_controllers:app\_storefront\_core:int\_klarna\_payments\_controllers:int\_klarna\_payments

Instance Type: Sandbox/Development

Deprecated: The preferred way of configuring HTTP and HTTPS hostnames is by using new features of the site aliases configuration ("Site URLs/Aliases Configuration"). The HTTP/HTTPS hostnames values set in this section will be used if no hostnames are defined by

HTTP Hostname:

HTTPS Hostname:

Instance Type: All

Cartridges:

Effective Cartridge Path:

If using pipelines, the cartridge path should be similar to:

app\_storefront\_pipelines:app\_storefront\_core:int\_klarna\_payments\_pipelines:int\_klarna\_payments

Instance Type: Sandbox/Development

Deprecated: The preferred way of configuring HTTP and HTTPS hostnames is by using new features of the site aliases configuration ("Site URLs/Aliases Configuration"). The HTTP/HTTPS hostnames values set in this section will be used if no hostnames are defined by

HTTP Hostname:

HTTPS Hostname:

Instance Type: All

Cartridges:

Effective Cartridge Path:



### 3.2.3 Klarna Payments Configurations

Also, there is a page in Business Manager where you have to configure Klarna Payments cartridge, you can find it in Site Preferences > Custom Preferences > Klarna\_Payments:

The screenshot shows the 'Klarna Payments' configuration page. At the top, there's a navigation bar with 'Merchant Tools', 'Administration', and 'Storefront'. Below that, the page title is 'Klarna Payments'. There are buttons for 'Cancel', 'Apply to Other Sites', and 'Save'. The main content area has a table with columns 'Name', 'Value', and 'Default Value'. The table lists several configuration items: 'Send product\_url and image\_url', 'merchant\_reference2 mapping', and 'Pre-Assessment is on for'. Each item has a text input field in the 'Value' column and a 'Default Value' column. There are also 'Edit Across Sites' links for each row.

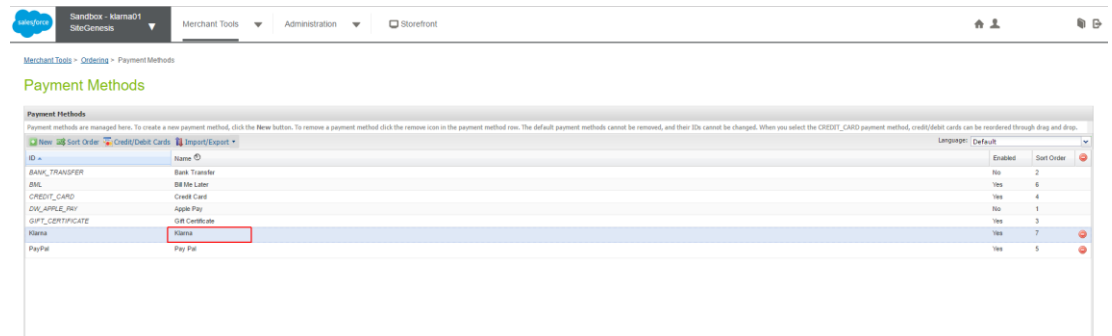
Parameter Name	Attribute ID	Description
Auto-capture	kpAutoCapture	If enabled, a full capture will be attempted automatically prior to order acknowledge. If disabled, the merchant needs to manually capture the order amount from merchant OMS or via Klarna Merchant Portal. Auto-capture is possible for orders when virtual card solution is not enabled.
Send product_url and image_url	sendProductAndImageURLs	If set to true product_url and image_url fields in Klarna Payments create session call will be included in API call, otherwise product_url and image_url fields will not be populated.
Merchant Reference 2 Mapping	merchant_reference2_mapping	The field from SCC order (basket) object that is mapped to merchant_reference2 field from klarna API request. Has to be one of the class attributes of SCC LineltemCtnr. Note that for complex data structures result may not always be as expected.

Border Color Preference	kpColorBorder	CSS hex color to be used in Klarna Payments iFrame
Border Selected Color Preference	kpColorBorderSelected	CSS hex color to be used in Klarna Payments iFrame
Button Color Preference	kpColorButton	CSS hex color to be used in Klarna Payments iFrame
Button Text Color Preference	kpColorButtonText	CSS hex color to be used in Klarna Payments iFrame
Checkbox Color Preference	kpColorCheckbox	CSS hex color to be used in Klarna Payments iFrame
Checkbox Checkmark Color Preference	kpColorCheckboxCheckmark	CSS hex color to be used in Klarna Payments iFrame
Details Color Preference	kpColorDetails	CSS hex color to be used in Klarna Payments iFrame
Header Color Preference	kpColorHeader	CSS hex color to be used in Klarna Payments iFrame
Link Color Preference	kpColorLink	CSS hex color to be used in Klarna Payments iFrame
Text Color Preference	kpColorText	CSS hex color to be used in Klarna Payments iFrame
Secondary Text Color Preference	kpColorTextSecondary	CSS hex color to be used in Klarna Payments iFrame
Border Radius Preference	kpRadiusBorder	Size (in pixels) of the border radius to be used in Klarna Payments iFrame
Attachments	kpAttachments	Flag to switch on/off the using of attachments when creating a session. Default is OFF.
Not available message on billing page	kpNotAvailableMessage	The Klarna Payment not available message on billing page. JSON string holding country code and corresponding message string. For example: <pre>{   "GB": "Klarna Payment not available",   "default": "Klarna Payment not available" }</pre>
Virtual Card Number Enabled	kpVCNEnabled	If set to true SFCC will create Virtual Card Number settlement from every Klarna order. Default is false
VCN Public Key ID	kpVCNkeyId	Unique identifier for the public key used for encryption of the card data
VCN Private Key	vcnPrivateKey	Your VCN 4096 bit RSA Private Key
VCN Public Key	vcnPublicKey	Your VCN 4096 bit RSA Public Key

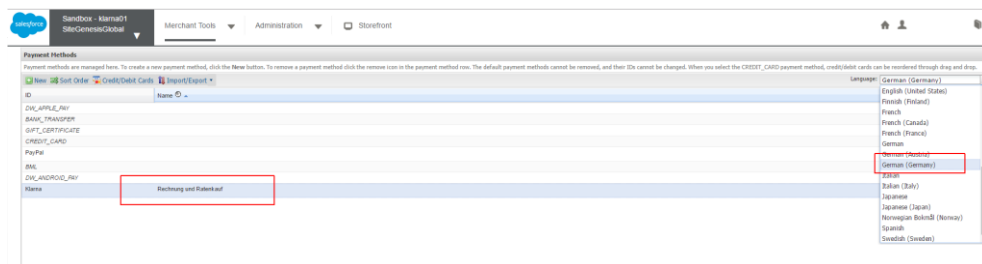
Here is the description of the Site Preferences fields which have to be configured:

### 3.2.4 Klarna Payments Logo and Payment Option Name

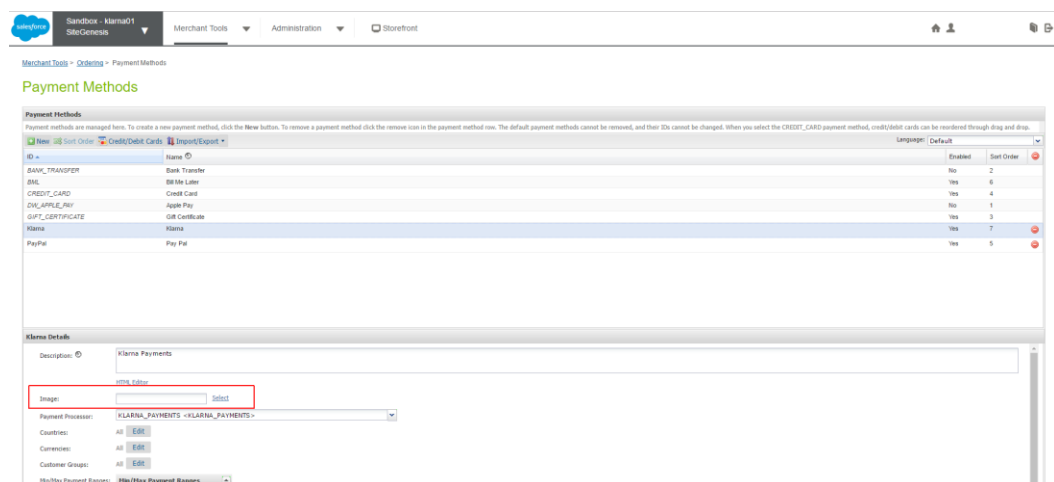
Payment option name can be changed in BM Merchant Tools > Ordering > Payment Methods as shown below.



Payment option name can be localized for the specific country and language through the language dropdown menu on the right, as shown below:

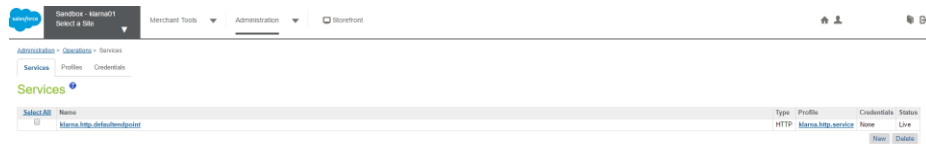


Logo can be changed from BM Merchant Tools > Ordering > Payment Methods. Enter the file name with the logo in the image field on payment method details view (see screenshot below)

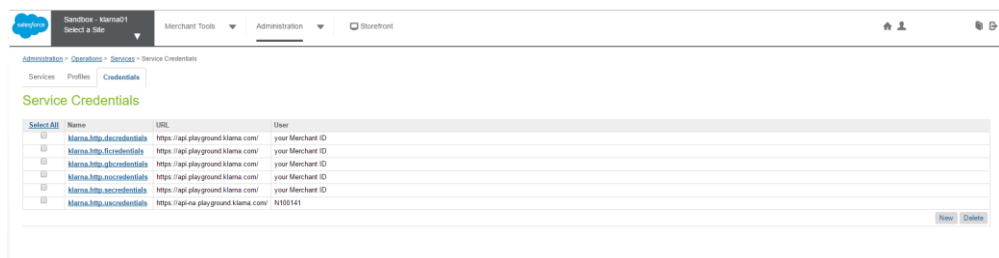


### 3.2.5 Services Configuration

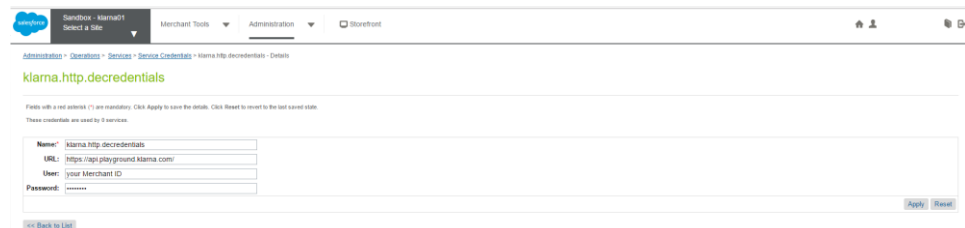
After the metadata import you should have Klarna service ready in Administration > Operations > Services like so:



You need to configure Klarna Payments services credentials. Go to Credentials tab as shown below:



You need to edit the existing credentials or create new ones using the credentials provided to you by Klarna and the API URL for the specific location, see for example below:



The Klarna Payments API is accessible through a few different URLs. There are different URLs for testing and for making live purchases as well as different URLs for depending on if you are based in Europe or in the U.S.

#### Live environment

The API for the European production environment can be found at

- <https://api.klarna.com/>

The API for the U.S. production environment can be found at

- <https://api-na.klarna.com/>

The API for the Oceania production environment can be found at

- <https://api-oc.klarna.com/>

### Testing environment

The API for the European Playground/testing environment can be found at

- <https://api.playground.klarna.com/>

The API for the U.S. Playground/testing environment can be found at

- <https://api-na.playground.klarna.com/>

The API for the Oceania Playground/testing environment can be found at

- <https://api-oc.playground.klarna.com/>

### 3.2.6 Custom attributes

---

ORDER system object has been extended with the following custom attributes:

Parameter Name	Attribute ID	Description
Klarna Payments Order ID	kpOrderID	Holding the Klarna Payments Order ID, if Klarna Payments is used for the specific order
VCN Brand	kpVCNBrand	Holding the Klarna Payments Virtual Card Number Card Brand
VCN Holder	kpVCNHolder	Holding the Klarna Payments Virtual Card Number card holder name
VCN Card ID	kpVCNCardID	Holding the Klarna Payments Virtual Card Number Card ID
VCN PCI Data	kpVCNPCIData	Holding the Klarna Payments Virtual Card Number Encrypted Card Data
VCN Initialization Vector	kpVCNIV	Holding the Klarna Payments Virtual Card Number Initialization Vector
VCN AES Key	kpVCNAESKey	Holding the Klarna Payments Virtual Card Number AES Key
Is VCN Used	kpIsVCN	True if VCN is used for payment of the order, otherwise false

PAYMENT TRANSACTION system object has been extended with the following custom attributes:

Parameter Name	Attribute ID	Description
Fraud Status	kpFraudStatus	Klarna Payments order fraud status

### 3.3 Custom Code

---

Integration may vary based on the customers' storefront. Site Genesis version 2 is used to demonstrate Klarna Payments integration.

#### 3.3.1 *Integration efforts*

---

The following steps are needed to complete the integration:

- Install the cartridges
- Change and import 'site-template.zip' (system objects and custom objects)
- Do the required code changes
- Configure your merchant's Klarna Payments Account in the Klarna back office
- Configure Klarna Payments parameters in Commerce Cloud
- Test
- Create styling for Klarna Widget (optional)

These steps should not require more than a few hours.

Note that the name of your storefront site ID should match the name in the 'site-template.zip', otherwise you will get import errors.

Also, note that the payment methods 'Klarna' and the payment processors 'KLARNA\_PAYMENTS' will be created or updated from the 'site-template.zip'.

#### 3.3.2 *Templates modifications*

---

The following changes should be made regardless of whether a controller or a pipeline integration approach are being used.

To integrate Klarna Payments the following storefront cartridge templates should be updated:

- default/checkout/billing/billing.isml
- default/checkout/summary/summary.isml
- default/checkout/billing/paymentmethods.isml
- default/checkout/shipping/minishipments.isml
- default/components/footer/footer\_UI.isml
- default/product/producttopcontentPS.isml
- default/product/productcontent.isml
- default/checkout/cart/cart.isml

#### default/checkout/summary/summary.isml

Add Code:

```
<isif condition="${session.privacy.KlarnaPaymentsFinalizeRequired}">
    <script><isinclude template="/resources/klarnapaymentsresources.isml"/></script>
    <script type="text/javascript" src="${URLUtils.staticURL('/js/klarna-
payments-finalize.js')}"></script>
    <script src="https://x.klarnacdn.net/kp/lib/v1/api.js"
async></script>
</isif>
```

Before `</isdecorate>` closing tag as shown below:

```
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
<form action="${URLUtils.https('cosummary-submit')}" method="post" class="submit-order">
  <fieldset>
    <div class="form-row">
      <a class="back-to-cart" href="${URLUtils.url('Cart-Show')}">
        <isprint value="${Resource.msg('summary.editcart','checkout',null)}" encoding="off" />
      </a>
      <button class="button-fancy-large" type="submit" name="submit" value="${Resource.msg('global.submitorder','locale',null)}">
        ${Resource.msg('global.submitorder','locale',null)}
      </button>
    </div>
    <input type="hidden" name="${dw.web.CSRFProtection.getTokenName()}" value="${dw.web.CSRFProtection.generateToken()}" />
  </fieldset>
</form>
</div>
<isif condition="${session.privacy.KlarnaPaymentsFinalizeRequired}">
  <script><isinclude template="/resources/klarnapaymentsresources.isml"/></script>
  <script type="text/javascript" src="${URLUtils.staticURL('/js/klarna-payments-finalize.js')}"></script>
  <script src="https://x.klarnacdn.net/kp/lib/v1/api.js" async></script>
</isif>
</isdecorate>
```

#### default/checkout/billing/billing.isml

Add Code:

```
<script><include template="/resources/klarnapaymentsresources.isml"/></script>
<script type="text/javascript" src="${URLUtils.staticURL('/js/klarna-
payments.js')}"></script>
<script src="https://x.klarnacdn.net/kp/lib/v1/api.js" async></script>
```

Before `</isdecorate>` closing tag as shown below:

```
184
185
186         <isif condition="${pdic.Basket.totalGrossPrice!=null && pdic.Basket.totalGrossPrice.value-pdic.gpITotal<0}">
187             <islet name="OrderTotal" value="${pdic.Basket.totalGrossPrice.value-pdic.gpITotal}" scope="pdic"/>
188             </isif>
189
190         </div>
191     </isif>
192
193     </isif>
194
195     </div>
196
197 </fieldset>
198
199 <iscomment> *****
200 payment methods
201 *****</iscomment>
202
203
204 <iscomment>payment method area</iscomment>
205 <include template="checkout/billing/paymentmethods"/>
206 <isbonusdiscountlineitem p_alert_text="${Resource.msg('billing.bonusproductalert','checkout',null)}" p_discount_line_item="${pdic.BonusDiscountLineItem}"/>
207
208
209 <div class="form-row form-row-button">
210     <button class="button-fancy-large" type="submit" name="${pdic.CurrentForms.billing.save.htmlName}" value="${Resource.msg('global.continueplaceorder','')}
211 </div>
212 <input type="hidden" name="${dw.web.CSRFPProtection.getTokenName()}" value="${dw.web.CSRFPProtection.generateToken()}" />
213
214 </form>
215 <script>
216     importScript("util/ViewHelpers.de");
217     var addressForm = pdic.CurrentForms.billing.billingAddress.addressFields;
218     var countries = ViewHelpers.getCountriesAndRegions(addressForm);
219     var json = JSON.stringify(countries);
220 </script>
221 <script><iscomment>window.Countries = <isprint value="2isoml" encoding="off"/></script>
222 <script><iscomment><include template="/resources/klarnapaymentsresources.isml"/></script>
223 <script type="text/javascript" src="${URLUtils.staticURL('/js/klarna-payments.js')}"></script>
224 <script src="https://x.klarnacdn.net/kp/lib/v1/api.js" async></script>
225 </isdecorate>
```

### default/checkout/billing/paymentmethods.isml

Add code:

```
<isif condition="${paymentMethodType.value === 'Klarna'}">hide</isif>
```

After `<div class="form-row label-inline">` close to line 18 as shown below:

```
1 <iscomment type="text/html" charset="UTF-8" compact="true"/>
2 <iscomment> TEMPLATE: paymentmethods.isml </iscomment>
3 <isinclude template="util/modules"/>
4 <isif condition="${pdic.OrderTotal > 0}">
5     <fieldset>
6
7         <legend>
8             ${Resource.msg('billing.paymentheader','checkout',null)}
9         </div class="dialog-required"> <span class="required-indicator">#48226; <em>${Resource.msg('global.requiredfield','locale',null)}</em></span></div>
10        </legend>
11
12        <div class="payment-method-options form-indent">
13            <isloop items="${pdic.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.options}" var="paymentMethodType">
14
15                <iscomment>Ignore GIFT CERTIFICATE method, GCs are handled separately before other payment methods.</iscomment>
16                <isif condition="${paymentMethodType.value.equals(dw.order.PaymentInstrument.METHOD_GIFT_CERTIFICATE)}"><iscontinue/></isif>
17
18                <div class="form-row label-inline">
19                    <isif condition="${paymentMethodType.value === 'Klarna'}">hide</isif>
20                    <islet name="radioID" value="${paymentMethodType.value}" scope="page">
21                        <div class="field-wrapper">
22                            <input id="is-${radioID}" type="radio" class="input-radio" name="${pdic.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlName}" v
23                        </div>
24                        <label for="is-${radioID}"><isprint value="${Resource.msg(paymentMethodType.label,'forms',null)}"/><isif condition="${!empty(dw.order.PaymentMpr.getI
25
26                </isloop>
```

Add Code:



<isinclude template="klarnapayments/klarnapaymentcategories.isml"/>

After </isloop> Close to line 28 as shown below:

```
paymentmethods.isml
6
7
8 <legend>
9   ${Resource.msg('billing.paymentheader','checkout',null)}
10 </legend>
11
12 <div class="payment-method-options form-indent">
13   <isloop items="${pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.options}" var="paymentMethodType">
14     <iscomment>Ignore GIFT_CERTIFICATE method, GCs are handled separately before other payment methods.</iscomment>
15     <isif condition="${paymentMethodType.value.equals(dw.order.PaymentInstrument.METHOD_GIFT_CERTIFICATE)}"><iscontinue/></isif>
16
17     <div class="form-row label-inline <isif condition="${paymentMethodType.value === 'Klarna'}">hide</isif>">
18       <isset name="radioID" value="${paymentMethodType.value}" scope="page"/>
19       <div class="field-wrapper">
20         <input id="is-${radioID}" type="radio" class="input-radio" name="${pdict.CurrentForms.billing.paymentMethods.selecte
21       </div>
22       <label for="is-${radioID}"><isprint value="${Resource.msg(paymentMethodType.label,'forms',null)}"/></isif condition="${!e
23     </div>
24   </isloop>
25
26   <isinclude template="klarnapayments/klarnapaymentcategories.isml"/>
27 </div>
28
29
30 <div class="form-row form-row-button">
```

Add Code:

<iscomment>Klarna Payments

</iscomment>

<isinclude template="klarnapayments/klarnapaymentblock.isml"/>

Before </fieldset> closing tag, close to line 150 as shown below:

```
paymentmethods.isml
144 <inputfield formfield="${pdict.CurrentForms.billing.paymentMethods.bml.year}" type="select" rowclass="year"/>
145 <inputfield formfield="${pdict.CurrentForms.billing.paymentMethods.bml.month}" type="select" rowclass="month"/>
146 <inputfield formfield="${pdict.CurrentForms.billing.paymentMethods.bml.day}" type="select" rowclass="day"/>
147
148 <inputfield formfield="${pdict.CurrentForms.billing.paymentMethods.bml.ssn}" type="input"/>
149
150 <div class="bml-terms-and-conditions form-caption">
151   <iscontentasset aid="bml-tc"/>
152 </div>
153
154 <div class="form-row form-caption">
155   <inputfield formfield="${pdict.CurrentForms.billing.paymentMethods.bml.termsandconditions}" type="checkbox"/>
156 </div>
157
158 </div>
159
160 <iscomment>
161   Custom processor
162   -----
163 </iscomment>
164
165 <div class="payment-method <isif condition="${!empty(pdict.selectedPaymentID) && pdict.selectedPaymentID='PayPal'}">paym
166   <!-- Your custom payment method implementation goes here. -->
167   ${Resource.msg('billing.custompaymentmethod','checkout',null)}
168 </div>
169
170 <iscomment>
171   Klarna Payments
172   -----
173 </iscomment>
174
175 <isinclude template="klarnapayments/klarnapaymentblock.isml"/>
176
177 </fieldset>
178
179 <iselse/>
180 <div class="gift-cert-used form-indent">
181   <isif condition="${pdict.goPITotal>0}">${Resource.msg('billing.giftcertnomethod','checkout',null)}</isif></iselse>${Resource.msg
182   <input type="hidden" name="${pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.htmlName}" value="${dw.or
```

default/checkout/shipping/minishipments.isml

Add the following code:

```
<isinclude template="klarnapayments/modules.isml"/>
```

In the end beginning of the file as shown below:

```
minishipments.isml
1 <iscontent type="text/html" charset="UTF-8" compact="true"/>
2 <isinclude template="util/modules.isml"/>
3 <isinclude template="klarnapayments/modules.isml"/>
4
5 <iscomment>
6     This template renders a summary of all shipments of the basket which is
7     used below the order summary at the right hand side in the checkout
8     process.
9 </iscomment>
10 <isset name="Shipments" value="${pdict.Basket.shipments}" scope="page"/>
11
12 <iscomment>the url to edit shipping addresses depends on the checkout scenario</iscomment>
13 <isset name="editUrl" value="${URLUtils.https('COShipping-Start')}" scope="page"/>
14 <isif condition="${pdict.CurrentForms.multishipping.entered.value}">
15     <isset name="editUrl" value="${URLUtils.https('COShippingMultiple-Start')}" scope="page"/>
16 </isif>
17
```

Add the following code:

```
<iskpaddresshelper p_address="${shipment.shippingAddress}"/>
```

After `<isminicheckout_address p_address="${shipment.shippingAddress}"/>` as shown below:

```
minishipments.isml
29
30
31 <isif condition="${Shipments.size() > 1 && pdict.Basket.productLineItems.size() > 0}"><div class="name">${Resource.msg(
32
33     <h3 class="section-header">
34         <isif condition="${shipment.giftCertificateLineItems.size() > 0}">
35             ${Resource.msg('minishipments.shipping', 'checkout', null)} <span>${Resource.msg('minishipments.giftcertdeliver',
36         <iselseif condition="${shipment.custom.shipmentType == 'instore'}">
37             <isset name="editUrl" value="${URLUtils.https('Cart-Show')}" scope="page"/>
38             <a href="${editUrl}" class="section-header-note">${Resource.msg('global.edit', 'locale', null)}</a>
39             <isset name="editUrl" value="${URLUtils.https('Cart-Show')}" scope="page"/>
40             <a href="${editUrl}" class="section-header-note">${Resource.msg('global.edit', 'locale', null)}</a>
41             <isset name="editUrl" value="${URLUtils.https('Cart-Show')}" scope="page"/>
42             <a href="${editUrl}" class="section-header-note">${Resource.msg('global.edit', 'locale', null)}</a>
43         </isif>
44     </h3>
45
46     <div class="details">
47         <iscomment>
48             render the detail section depending on whether this is a physical shipment with products
49             (shipped to an address) or if this is a gift certificate (send via email)
50         </iscomment>
51         <isif condition="${shipment.giftCertificateLineItems.size() > 0}">
52             <isloop items="${shipment.giftCertificateLineItems}" var="giftCertLI">
53                 <div><isprint value="${giftCertLI.recipientName}"></div>
54                 <div><isprint value="${giftCertLI.recipientEmail}"></div>
55             </isloop>
56         <iselseif condition="${shipment.shippingAddress != null && pdict.Basket.productLineItems.size() > 0}">
57             <iskpaddresshelper p_address="${shipment.shippingAddress}"/>
58             <isif condition="${!empty(shipment.shippingMethod)}">
59                 <div class="minishipments-method">
60                     <span>${Resource.msg('order.orderdetails.shippingmethod', 'order', null)}</span>
61                     <span><isprint value="${shipment.shippingMethod.displayName}"></span>
62                 </div>
63             </isif>
64         </isif>
65     </div>
66 </isif>
67 </isloop>
68 </isif>
69
```

**default/components/footer/footer\_UI.isml**

Add Code:

```
<isinclude template="klarnapayments/scripts.isml"/>
```

Before `<script src="{URLUtils.staticURL('/js/app.js')}"></script>` script tag as shown below:

```
27
28 <include template="klarnapayments/scripts.isml"/>
29
30 <script src="{URLUtils.staticURL('/js/app.js')}"></script>
```

### default/product/producttopcontentPS.isml

Add the following code:

```
<include template="klarnapayments/modules.isml"/>
```

```
<iskosmpdp p_product="{psProduct}" />
```

, right under the `pricing` template include as shown below:

```
165 <label>${Resource.msg('product.setpricelabel','product',null)}</label>
166 <include template="product/components/pricing"/>
167
168 <include template="klarnapayments/modules.isml"/>
169 <iskosmpdp p_product="{psProduct}" />
170
```

### default/product/productcontent.isml

Add the following code:

```
<isif condition="{!isQuickView}">

    <include template="klarnapayments/modules.isml"/>

    <iskosmpdp p_product="{pdict.Product}" />

</isif>
```

, right under the `pricing` template include as shown below:

```
77 <include template="product/components/pricing"/>
78
79 <isif condition="{!isQuickView}">
80 <include template="klarnapayments/modules.isml"/>
81 <iskosmpdp p_product="{pdict.Product}" />
82 </isif>
83
84 <isset name="pam" value="{pdict.Product.getAttributeModel()}" scope="page"/>
```

### js/pages/product/variant.js

Add the following code:

```
window.KlarnaOnsiteService = window.KlarnaOnsiteService || [];
window.KlarnaOnsiteService.push({eventName: 'refresh-placements'});
```

in the variation update callback, on line 35

```

29 |         callback: function () {
30 |             if (SitePreferences.STORE_PICKUP) {
31 |                 productStoreInventory.init();
32 |             }
33 |             image.replaceImages();
34 |             tooltip.init();
35 |             window.KlarnaOnsiteService = window.KlarnaOnsiteService || [];
36 |             window.KlarnaOnsiteService.push({eventName: 'refresh-placements'});
37 |         }

```

### default/checkout/cart/cart.isml

Add the following code:

```
<include template="klarnapayments/modules.isml"/>
```

```
<iskosmcart p_lineitemctnr="${pdict.Basket}" />
```

, right after the `<isordertotals>`:

```

820 |         <div class="cart-order-totals">
821 |             <isordertotals p_lineitemctnr="${pdict.Basket}" p_totallabel="${Resource.msg('global.estimatedtotal', 'locale', null)}"
822 |                 <include template="klarnapayments/modules.isml"/>
823 |                 <iskosmcart p_lineitemctnr="${pdict.Basket}" />
824 |             </div>
825 |

```

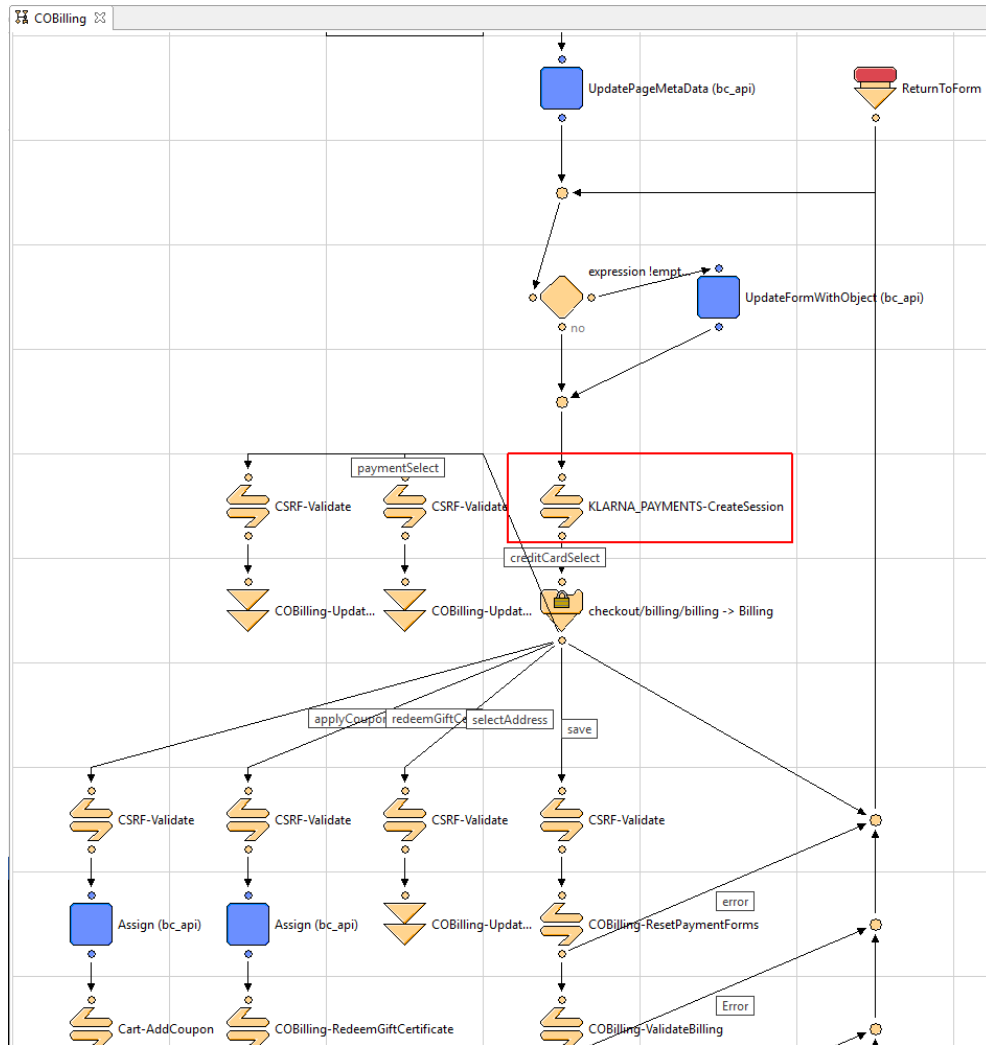
### 3.3.3 Pipeline modifications

If using a pipeline based SiteGenesis integration, additionally follow the instructions in this chapter. If integrating via the controller based model see next chapter.

To use Klarna Payments cartridge you will need to make the following changes in pipelines.

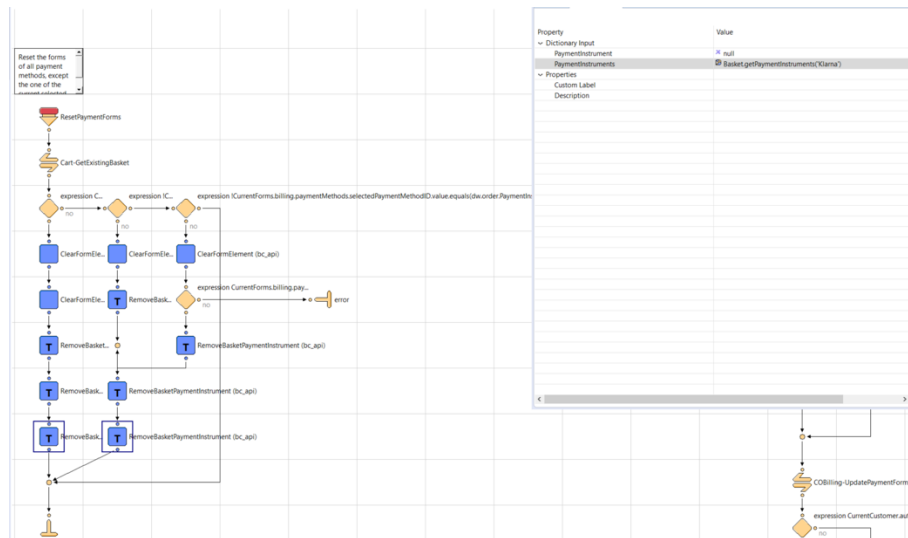
#### 3.3.3.1 COBilling pipeline

Go to COBilling pipeline, Start node and add KLARNA\_PAYMENTS-CreateSession call node entry point before checkout/billing/billing -> Billing interaction continue node (see screen shot below)



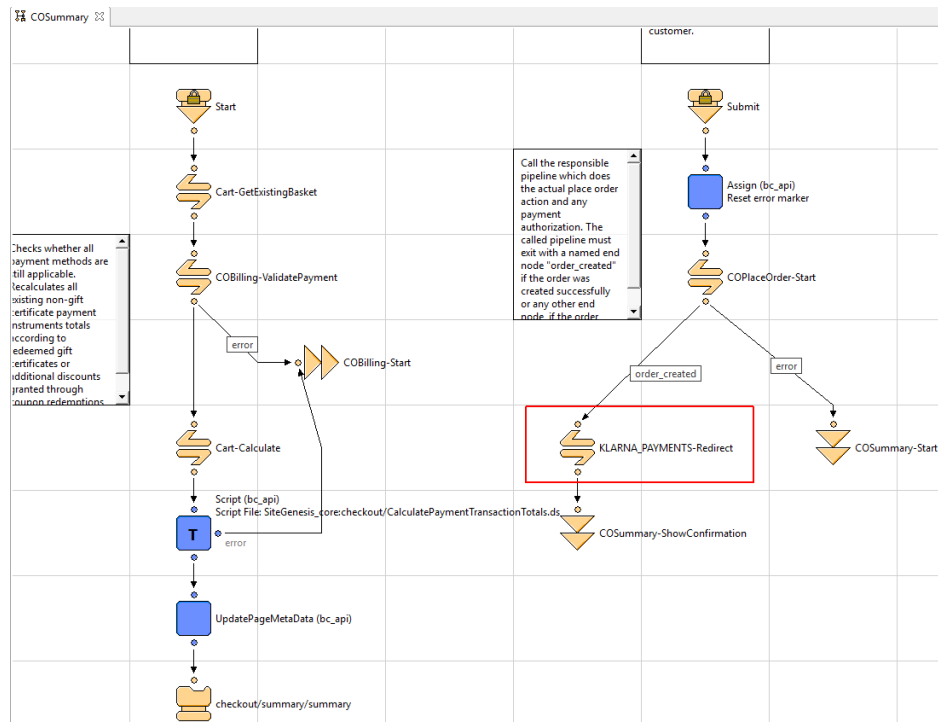
In pipeline COBilling-ResetPaymentForms, add two pipelets 'com.demandware.pipelet.basket.RemoveBasketPaymentInstrument' at the end of the pipeline.

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



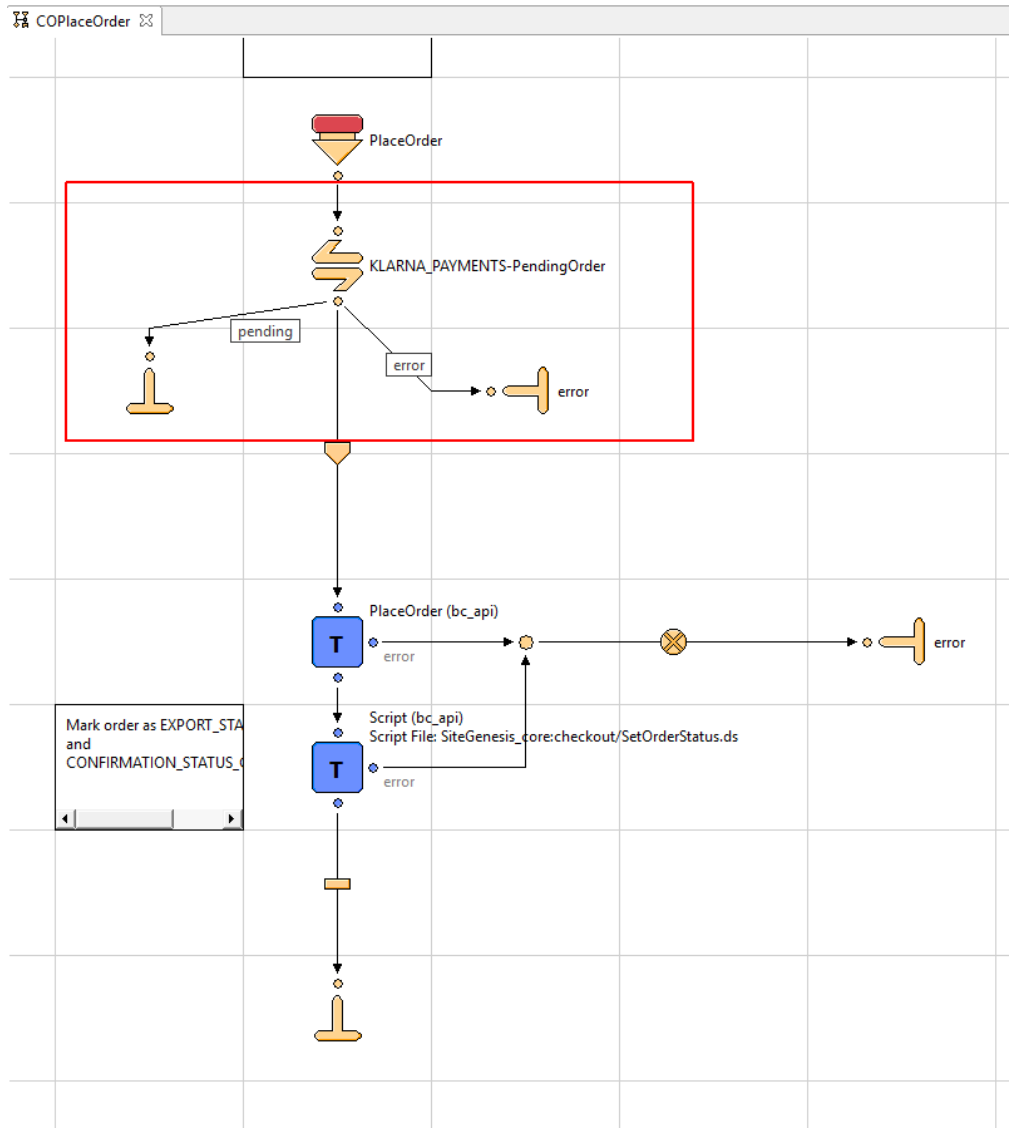
### 3.3.3.2 COSummary pipeline.

Go to COSummary pipeline, Submit node and add KLARNA\_PAYMENTS-Redirect call node entry point before COSummary-ShowConfirmation jump node (see screen shot below)



### 3.3.3.3 **COPlaceOrder pipeline.**

Go to COPlaceOrder pipeline, PlaceOrder node and add KLARNA\_PAYMENTS-PendingOrder call node entry point before PlaceOrder script node node (see screen shot below). Create 'pending' and 'error' transitions and end nodes as shown on the screenshot



### 3.3.4 Controller modifications

If using a controller based SiteGenesis integration, additionally follow the instructions in this chapter. If integrating via the pipeline method please see Pipeline modifications chapter above.

#### 3.3.4.1 COBilling controller.

Go to COBilling controller, returnToForm method and add the following code block:

```
try
{
```



```

        require('int_klarna_payments_controllers/cartridge/controllers/KLARNA_PAYMENTS.js')
        .CreateSession();
    } catch( e )
    {
        require( 'dw/system/Logger' ).getLogger( 'COBilling.js' ).error( 'Klarna Create Session
Error: {0}', e );
    }

```

*after pageMeta.update (see screen shot below)*

```

79= function returnToForm(cart, params) {
80     var pageMeta = require('~/cartridge/scripts/meta');
81
82     // if the payment method is set to gift certificate get the gift certificate code from the form
83     if (!empty(cart.getPaymentInstrument()) && cart.getPaymentInstrument().getPaymentMethod() === PaymentInstrument.METHOD_GIFT_CERTIFICATE) {
84         app.getForm('billing').copyFrom({
85             giftCertCode: cart.getPaymentInstrument().getGiftCertificateCode()
86         });
87     }
88
89     pageMeta.update({
90         pageTitle: Resource.msg('billing.meta.pagetitle', 'checkout', 'SiteGenesis Checkout')
91     });
92
93     try
94     {
95         require('int_klarna_payments_controllers/cartridge/controllers/KLARNA_PAYMENTS.js').CreateSession();
96     } catch( e )
97     {
98         require( 'dw/system/Logger' ).getLogger( 'COBilling.js' ).error( 'Klarna Create Session Error: {0}', e );
99     }
100
101     if (params) {
102         app.getView(require('~/cartridge/scripts/object').extend(params, {
103             Basket: cart.object,
104             ContinueURL: URLUtils.https('COBilling-Billing')
105         })).render('checkout/billing/billing');
106     } else {
107         app.getView({
108             Basket: cart.object,
109             ContinueURL: URLUtils.https('COBilling-Billing')
110         }).render('checkout/billing/billing');
111     }
112 }

```

In the method `resetPaymentForms()`, add the command for the three conditions:  
`cart.removePaymentInstruments(cart.getPaymentInstruments('Klarna'));`

**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 Klarna PaymentInstrument`

```

        cart.removePaymentInstruments(cart.getPaymentInstruments('Klarna'));
    } else if
    (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(Pa
ymentInstrument.METHOD_CREDIT_CARD)) {
        app.getForm('billing').object.paymentMethods.bml.clearFormElement();

    cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHO
D_BML));
        cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));
        // remove Klarna PaymentInstrument
        cart.removePaymentInstruments(cart.getPaymentInstruments('Klarna'));
    } else if
    (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(Pa
ymentInstrument.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.METHO
D_CREDIT_CARD));
        cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));
        // remove Klarna PaymentInstrument
        cart.removePaymentInstruments(cart.getPaymentInstruments('Klarna'));
    }
    return true;
});

return status;
}

```

#### 3.3.4.2 COSummary controller.

Go to COSummary controller, submit() method and add the following code block:

```

try
{
    require('int_klarna_payments_controllers/cartridge/controllers/KLARNA_PAYMENTS.js').Red
irect();
} catch( e )
{
    require( 'dw/system/Logger' ).getLogger( 'COSummary.js' ).error( 'Klarna Redirect Error: {0}',
e );
}
before showConfirmation(placeOrderResult.Order) (see screen shot below)

```

```

57@ /**
58 * This function is called when the "Place Order" action is triggered by the
59 * customer.
60 */
61@function submit() {
62 // Calls the COPlaceOrder controller that does the place order action and any payment authorization.
63 // COPlaceOrder returns a JSON object with an order_created key and a boolean value if the order was created successfully.
64 // If the order creation failed, it returns a JSON object with an error key and a boolean value.
65 var placeOrderResult = app.getController('COPlaceOrder').Start();
66 if (placeOrderResult.error) {
67     start({
68         PlaceOrderError: placeOrderResult.PlaceOrderError
69     });
70 } else if (placeOrderResult.order_created) {
71
72     try
73     {
74         require('int_klarna_payments_controllers/cartridge/controllers/KLARNA_PAYMENTS.js').Redirect();
75     } catch( e )
76     {
77         require( 'dw/system/Logger' ).getLogger( 'COSummary.js' ).error( 'Klarna Redirect Error: {0}', e );
78     }
79
80     showConfirmation(placeOrderResult.Order);
81 }
82 }
83

```

### 3.3.4.3 OrderModel.js

Go to OrderModel.js, placeOrder method and add the following code block:

```

if ( session.privacy.KlarnaPaymentsFraudStatus === 'PENDING' )
{
    try
    {
        require('int_klarna_payments_controllers/cartridge/controllers/KLARNA_PAYMENTS
.js').PendingOrder(order);
    } catch( e )
    {
        require( 'dw/system/Logger' ).getLogger( 'OrderModel.js' ).error( 'Klarna Payments
Pending Order Error: {0}', e );
    }
    return;
}

```

**before** var placeOrderStatus = OrderMgr.placeOrder(order); (see screen shot below)

```

1  'use strict';
2
3@ /**
4 * Module for ordering functionality.
5 * @module models/OrderModel
6 */
7
8 /* API Includes */
9 var AbstractModel = require('./AbstractModel');
10 var Order = require('dw/order/Order');
11 var OrderMgr = require('dw/order/OrderMgr');
12 var Resource = require('dw/web/Resource');
13 var Status = require('dw/system/Status');
14 var Transaction = require('dw/system/Transaction');
15
16@ /**
17 * Place an order using OrderMgr. If order is placed successfully,
18 * its status will be set as confirmed, and export status set to ready.
19 * @param {dw.order.Order} order
20 */
21@function placeOrder(order) {
22
23     if( session.privacy.KlarnaPaymentsFraudStatus === 'PENDING' )
24     {
25         try
26         {
27             require('int_klarna_payments_controllers/cartridge/controllers/KLARNA_PAYMENTS.js').PendingOrder( order );
28         } catch( e )
29         {
30             require( 'dw/system/Logger' ).getLogger( 'OrderModel.js' ).error( 'Klarna Payments Pending Order Error: {0}', e );
31         }
32         return;
33     }
34
35     var placeOrderStatus = OrderMgr.placeOrder(order);
36     if (placeOrderStatus === Status.ERROR) {
37         OrderMgr.failOrder(order);
38         throw new Error('Failed to place order.');
```

### 3.3.5 VCN Decryption

---

In order to decrypt the virtual card details stored on order level and authorize the credit card processor you can use the following code snippet. You can find more information about the decryption process [here](#).

```
var OrderMgr = require( 'dw/order/OrderMgr' );
var Cipher = require( 'dw/crypto/Cipher' );
var Encoding = require( 'dw/crypto/Encoding' );
var Site = require( 'dw/system/Site' );

var Order = OrderMgr.getOrder( "order_id" );
var VCNPrivateKey = Site.getCurrent().getCustomPreferenceValue(
    'vcnPrivateKey' );
var cipher = new Cipher();

var keyEncryptedBase64 = Order.custom.kpVCNAESKey;
var keyEncryptedBytes = Encoding.fromBase64( keyEncryptedBase64 );
var keyDecrypted = cipher.decryptBytes( keyEncryptedBytes, VCNPrivateKey,
    "RSA/ECB/PKCS1PADDING", null, 0 );
var keyDecryptedBase64 = Encoding.toBase64( keyDecrypted );
var cardDataEncryptedBase64 = Order.custom.kpVCNPCIData;
var cardDataEncryptedBytes = Encoding.fromBase64( cardDataEncryptedBase64
    );
var cardDecrypted = cipher.decryptBytes( cardDataEncryptedBytes,
    keyDecryptedBase64, "AES/CTR/NoPadding", Order.custom.kpVCNIV, 0 );

var cardDecryptedUtf8 = decodeURIComponent( cardDecrypted );
var cardObj = JSON.parse( cardDecryptedUtf8 );
var expiryDateArr = cardObj.expiry_date.split( "/" );

// Retrieve ecnrypted card details
var cardPAN = cardObj.pan, cardCVV = cardObj.cvv,
    cardExpiryMonth = expiryDateArr[0], cardExpiryYear =
    expiryDateArr[1];
```

#### 4. Release History

Version	Date	Changes
17.1.0		Initial release
18.1.0		Implement payments endpoint
19.1.0		Added SFRA version.
19.1.1		Updated VCN to use the newest API version
19.1.2		Fix auto capture for the pipeline's cartridge
19.1.3		On-site Messaging updates OC endpoint added
19.1.4		New country locales added. Minor bug fixes. Cartridge templates and forms updated for latest SFRA.
19.1.5		Added additional verification for all notifications. Minor fixes around the configuration objects. Added Canadian support. Documentation updates.
19.1.6		New country locales added. Updated VCN to store encrypted card details