Ingenico Cartridge

Integration documentation

version 20.9.0

Table of Contents

1. Summary

The Ingenico LINK cartridge facilitates integration of Salesforce Commerce Cloud with the Ingenico Connect Platform.

This integration guide describes how to integrate the Ingenico payment processor cartridge into the Salesforce Commerce Cloud Storefront Reference Architecture. To this end, this guide shows how to setup and configure the **int_ingenico_sfra** and **bm_ingenico** cartridges, and how to customize your Storefront to make use of the cartridges.

2. Component Overview

2.1 Functional Overview

The cartridge supports creditcard, creditcard via hosted payment page, iDEAL, PayPal and Trustly via the Ingenico Connect APIs.

2.2 Use Cases

The cartridge supports the following use cases:

- Consumers can use creditcard, iDEAL, PayPal or Trustly to pay for their purchase
- Credit card, iDEAL, PayPal and Trustly payments can be done with or without an approval.
- Approve payments can be integrated with your Order Manager.
- Cancellations of transactions can be integrated with your Order Manager.

2.3 Compatibility

This cartridge has been available since Salesforce Commerce Cloud version 20.9 (compatibility version 18.10).

It was designed and developed for Storefront Reference Architecture version 4.2.1.

2.4 Privacy, Payment

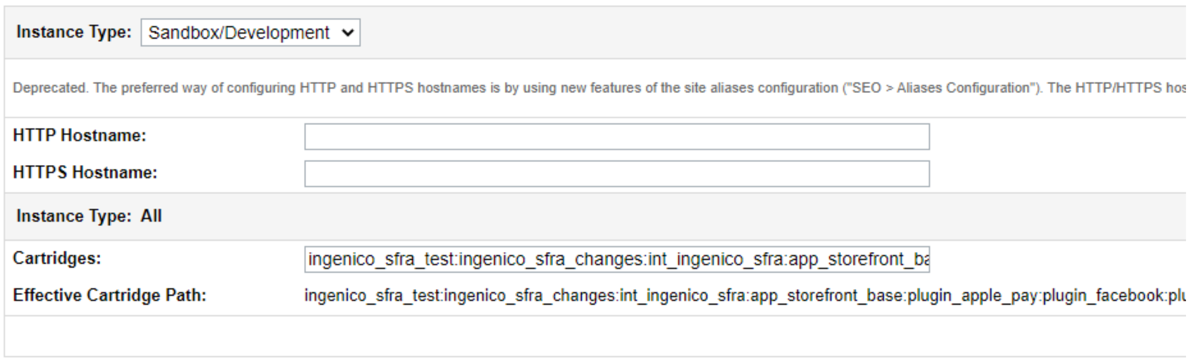
This cartridge does not process credit card data. As part of a payment, the cartridge does process the customer's billing, shipping and email address as well as the contents of the shopping basket.

3. Implementation Guide

3.1 Setup

To use the Ingenico Payment integration you need to install the int_ingenico_sfra and bm_ingenico cartridges. The ingenico_sfra_changes cartridge contains an example of how your custom code could work with the Ingenico integration.

To install these cartridges, go to Administration -> Sites -> Manage Sites. Click on the site where you want to install them and continue on the tab "Settings". Here you can alter the cartridges path, include the int_ingenico_sfra and ingenico_sfra_changes before the app_storefront_base but after your own overlay cartridges. The path should look similar to this: "your_site_overlay:ingenico_sfra_changes:int_ingenico_sfra:app_storefront_base" .



Instance Type:	Sandbox/Development

Deprecated. The preferred way of configuring HTTP and HTTPS hostnames is by using new features of the site aliases configuration ("SEO > Aliases Configuration"). The HTTP/HTTPS hos

HTTP Hostname:	
HTTPS Hostname:	
Instance Type: All	
Cartridges:	ingenico_sfra_test:ingenico_sfra_changes:int_ingenico_sfra:app_storefront_ba
Effective Cartridge Path:	ingenico_sfra_test:ingenico_sfra_changes:int_ingenico_sfra:app_storefront_base:plugin_apple_pay:plugin_facebook:plu

To install the business manager cartridge, return to Administration -> Sites -> Manage Sites and click on the link under Business Manager Site. Alter the cartridge path as before, make sure to include bm_ingenico and int_ingenico_sfra before bm_custom_plugin and append app_storefront_base cartridge to the end of the cartridge path like this: "bm_ingenico:int_ingenico_sfra:bm_custom_plugin: app_storefront_base".

After installing the cartridges, the site import needs to be prepared. Rename the metadata/site_import/sites/siteId directory to match the siteId you want to install the cartridge for and create a zip file of the site_import directory.

When the zip file is created go to Administration -> Site Development -> Site Import & Export. Here you can upload the zip file. To do so, select local and click browse to find the zip file, when selected, click upload. When the upload is done select the file from the list and select import. You will get an email when the import job is complete.

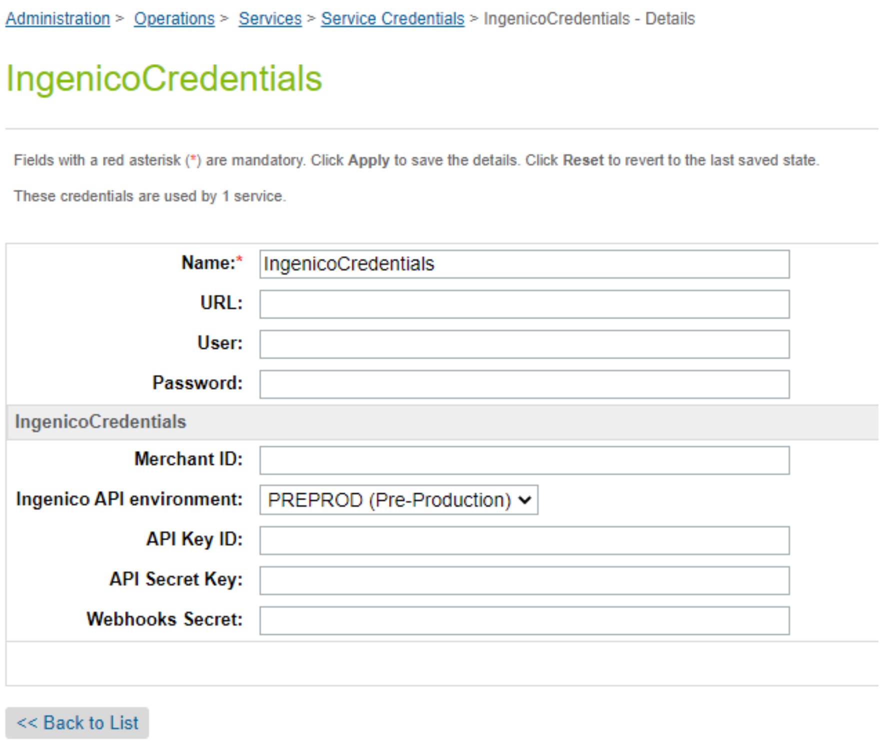
3.2 Configuration

There are several locations to configure this cartridge, the site import already creates the payment processor and payment methods for your site.

3.2.1 Services

Found under Administration -> Operations -> Services. Here you can find (or create new) services that can be linked on the Payment Processors page. A default IngenicoPayment service is already created via the site import process.

The Service itself has no specific Ingenico configuration but it is linked to a Service credential that does.


Administration > Operations > Services > Service Credentials > IngenicoCredentials - Details

IngenicoCredentials

Fields with a red asterisk (*) are mandatory. Click **Apply** to save the details. Click **Reset** to revert to the last saved state.

These credentials are used by 1 service.

Name:*	IngenicoCredentials
URL:	
User:	
Password:	
IngenicoCredentials	
Merchant ID:	
Ingenico API environment:	PREPROD (Pre-Production)
API Key ID:	
API Secret Key:	
Webhooks Secret:	

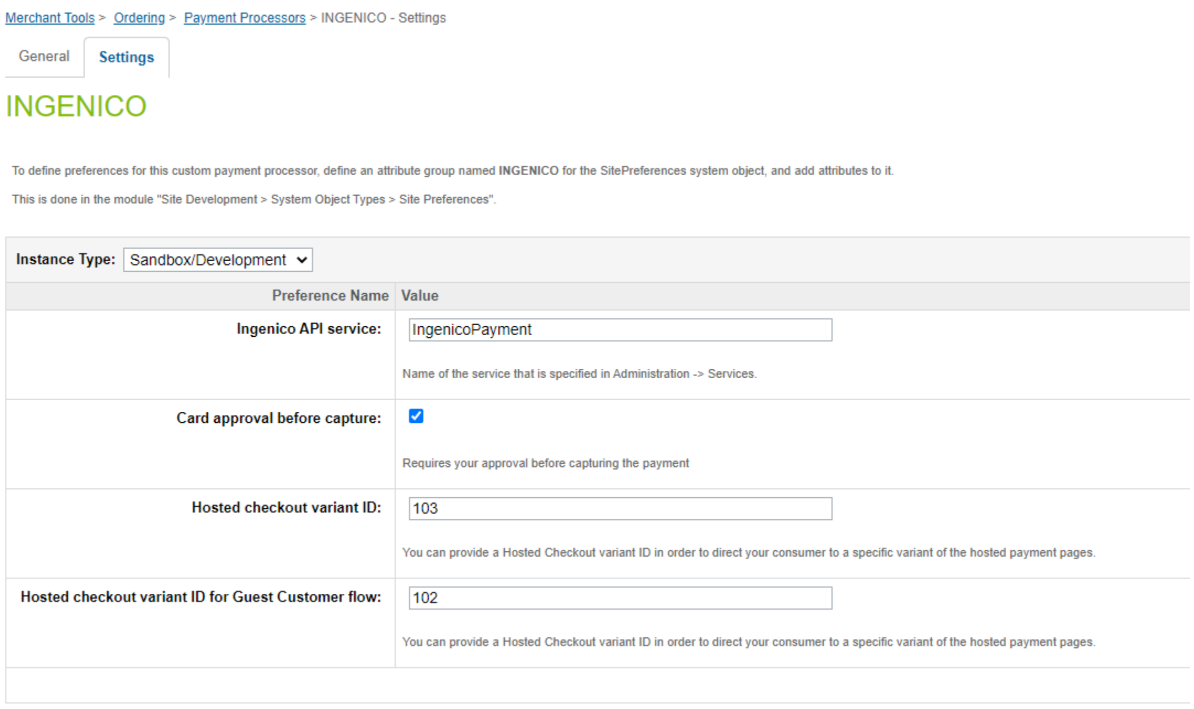
Here you can enter the information needed to access the Ingenico API, the data for these are supplied in the Configuration Center when logging in using your Ingenico account.

Note that the webhooks URL is site-specific: *<your host>/on/demandware.store/Sites-<your site>-Site/default/Ingenico-Notify*

3.2.2 Payment Processors

By default, every site uses the same service, if you need sites to use different services you can configure them to have their own.

Found under Merchant Tools -> Ordering -> Payment Processors. Here you can find the INGENICO payment processor, click on it to find its configuration under the settings tab.


Merchant Tools > Ordering > Payment Processors > INGENICO - Settings

General | **Settings**

INGENICO

To define preferences for this custom payment processor, define an attribute group named **INGENICO** for the SitePreferences system object, and add attributes to it.

This is done in the module "Site Development > System Object Types > Site Preferences".

Instance Type: Sandbox/Development

Preference Name	Value
Ingenico API service:	IngenicoPayment Name of the service that is specified in Administration -> Services.
Card approval before capture:	✓ Requires your approval before capturing the payment
Hosted checkout variant ID:	103 You can provide a Hosted Checkout variant ID in order to direct your consumer to a specific variant of the hosted payment pages.
Hosted checkout variant ID for Guest Customer flow:	102 You can provide a Hosted Checkout variant ID in order to direct your consumer to a specific variant of the hosted payment pages.

Here you can place the name of the service to use which we will configure next. By default, IngenicoPayment is selected. This configuration is done here so that different sites can use different service credentials.

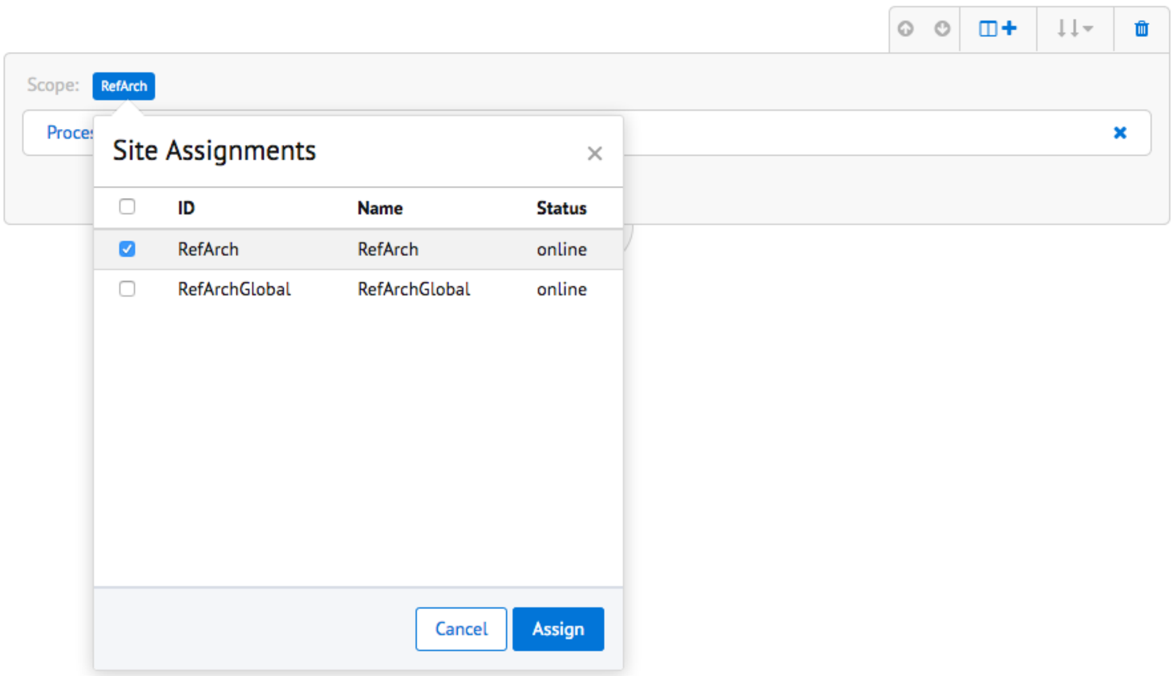
3.2.3 Credit Cards and redirect payment methods

In the same location as chapter 3.2.2 you can alter “Card approval before capture” if you prefer payments to be approved before being captured. When approval is needed before capture, you will need to capture the payment yourself, this will be explained in 3.3.5. Finally, you can also provide a Hosted Checkout variant ID in order to direct your consumer to a specific variant of the hosted payment pages.

3.2.4 Job

Found under Administration -> Operations -> Jobs. Two jobs, ProcessIngenicoWebhooks and ProcessExpiredOrders, are already created for you, but you do need to assign your site contexts to the job step in order for it to work.

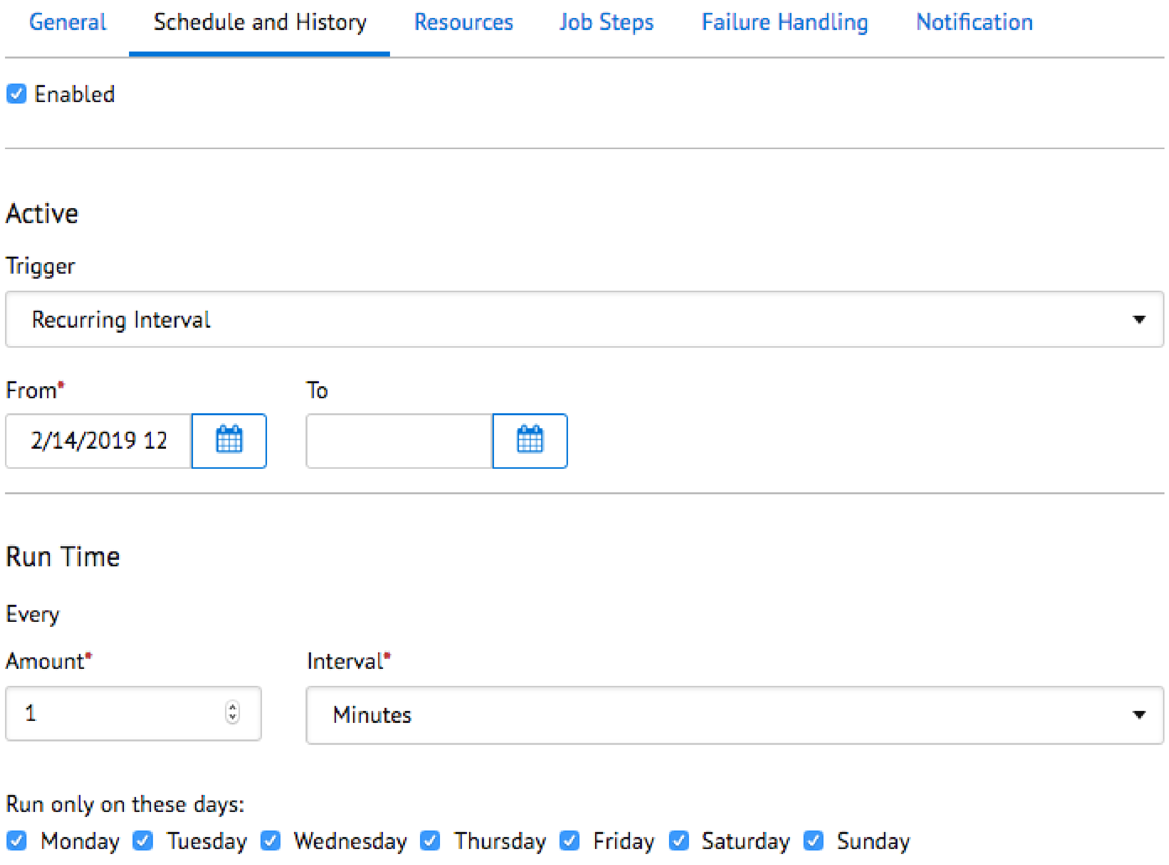
Click on the scope to assign your relevant sites, by default only RefArch is selected you will need to do this for the Ingenico Job.


Scope: RefArch

Site Assignments

	ID	Name	Status
✓	RefArch	RefArch	online
	RefArchGlobal	RefArchGlobal	online

On the "Schedule and History" tab a recurring interval should already be set.



✓ Enabled

Active

Trigger: Recurring Interval

From*: 2/14/2019 12

To:

Run Time

Every

Amount*: 1

Interval*: Minutes

Run only on these days:
✓ Monday ✓ Tuesday ✓ Wednesday ✓ Thursday ✓ Friday ✓ Saturday ✓ Sunday

3.3 Custom Code

An example implementation is supplied in the *ingenico_sfra_changes* cartridge. You can use this in one of two ways:

- Add it to your cartridge path and overwrite where needed
- copying over only what you need from it into your own code.

In this chapter the different alterations in the *ingenico_sfra_changes* cartridge are explained as if you are making these changes in your overlay cartridge.

3.3.1 Display

templates/default/checkout/checkout.isml

Find the comment saying "Checkout Forms: Shipping, Payment, Coupons, Billing, etc", below that there is an alert box, replace it with the alert box found in the *ingenico_sfra_changes*.

Copy over the following files:
templates/default/checkout/billing/paymentOptions/paymentOptionsContent.isml

templates/default/checkout/billing/paymentOptions/paymentOptionsSummary.isml

templates/default/checkout/billing/paymentOptions/paymentOptionsTabs.isml

templates/default/checkout/billing/paymentOptions/creditCardContent.isml

templates/default/checkout/billing/paymentOptions/creditCardSummary.isml

templates/default/checkout/billing/paymentOptions/creditCardTab.isml

templates/default/checkout/billing/paymentOptions/hostedCheckoutContent.isml

templates/default/checkout/billing/paymentOptions/hostedCheckoutSummary.isml

templates/default/checkout/billing/paymentOptions/hostedCheckoutTab.isml

templates/default/checkout/billing/paymentOptions/idealContent.isml

templates/default/checkout/billing/paymentOptions/idealSummary.isml

templates/default/checkout/billing/paymentOptions/idealTab.isml

templates/default/checkout/billing/paymentOptions/trustlyContent.isml

templates/default/checkout/billing/paymentOptions/trustlySummary.isml

templates/default/checkout/billing/paymentOptions/trustlyTab.isml

templates/default/checkout/billing/paymentOptions/paypalContent.isml

templates/default/checkout/billing/paymentOptions/paypalSummary.isml

templates/default/checkout/billing/paymentOptions/paypalTab.isml

templates/default/checkout/billing/paymentOptions.isml

templates/default/checkout/billing/creditCardForm.isml

templates/default/checkout/billing/storedPaymentInstruments.isml

templates/default/account/payment/paymentForm.isml

static/default/images/paymentmethod-card.png

static/default/images/ideal.png

static/default/images/paypal.png

static/default/images/trustly.png

For the files where you already have your own overwrites try to compare the files and find out what needs to be copied over.

Copy over the .properties files from **templates/resources** or the separate properties inside.

Copy over **controllers/Checkout.js** and **controllers/CheckoutServices.js**, these files append the "Begin" and "SubmitPayment" endpoints respectively to set extra viewData.

Copy over **controllers/PaymentInstrument.js**, this file append the "AddPayment", "DeletePayment" and "SavePayment" endpoints. An endpoint "GetClientSession" has been added for obtaining an Ingenico client session that is required for interacting with the Connect SDK. The Connect SDK has been included in the client directory.

Copy over **models/order.js**, this file adds a resource text to the orderModel.

Copy over the files found in the **client** directory.

After adding the overlay cartridge files make sure to run the command "npm run compile:js" in the root of the main cartridge. The static files created from this command can be found in *ingenico_sfra_changes*'s **static** directory, if you are applying the changes to a custom overlay you will need to recompile the files in that cartridge.

3.3.2 Forms

Copy over the following files:

forms/default/address.xml.

forms/default/ideal.xml.

forms/default/creditCard.xml

forms/default/billing.xml

Add the idealFields include tag as shown in the *ingenico_sfra_changes*.

scripts/hooks/checkout/checkoutHelper.js

This file contains the processForm method that set additional fields to viewData.

3.3.3 Webhooks

Ingenico webhooks enter in a *int_ingenico_sfra* controller and are then processed in a job. When this job processes a webhook it will call a hook where a merchant can add business logic. You can currently add logic to the following hooks:

ingenico.transactionUpdate.payment

hooks.json in the *ingenico_sfra_changes* cartridge shows how you can do this.

scripts/hooks/payment/transactionUpdate/ingenico.js

This is the file that is called in the hooks. The different functions show the recommended way to handle the cases that can occur when receiving the Ingenico webhooks. Please note that this file should not be used as is, it should be extended to meet your specific needs and the expectations of your Order Manager.

scripts/hooks/checkout/confirmationEmailHelper.js

Normally the confirmation email is sent during a request by the user which has a different context than when done during a job. Since we are sending it in a job's context some changes were needed, which is done in this file.

4. Operations, Maintenance

4.1 Data Storage

The Ingenico LINK cartridge extends several System Objects with custom properties for saving data related to Ingenico transactions and creates Custom Objects for processing webhooks.

4.1.1 OrderPaymentInstrument custom properties

ingenicoEncryptedCustomerInput: client-side encrypted customer input

ingenicoIssuerId: selected the issuer

saveCard: Indicates whether a card must be saved to the registered customer's account.

storedPaymentUUID: the UUID of the selected payment.

4.1.2 PaymentTransaction custom properties

ingenicoTransactionId: Unique transaction ID created by the Ingenico API.

ingenicoCheckoutId: Id of the created hosted checkout.

ingenicoLastProcessedNotificationSortKey: SortKey for the last succesfully processed Notification.

ingenicoRedirect: URL including the query string to which the customer will be directed in case of a payment for which the customer needs to be redirected to a bank.

ingenicoResult: The result code returned by the Ingenico API. Examples: PAID, CAPTURE_REQUESTED, PENDING_APPROVAL, etc.

ingenicoMerchantReference: A reference to identify the order for which the transaction belongs to.

ingenicoTransactionAmount: The amount of the payment.

ingenicoIsCancellable: Indicates whether the transaction is cancellable or not.

ingenicoIsRefundable: Indicates whether the transaction is refundable or not.

ingenicoRETURNMAC: a RETURNMAC returned by the Connect API in case of a payment with redirection.

4.1.3 ServiceCredential custom properties

ingenicoApiClientId: Your Ingenico API Client ID.

ingenicoApiClientSecret: Your Ingenico API Client Secret.

ingenicoApiEnvironment: The API environment. Examples: PROD, SANDBOX or PREPROD.

ingenicoMerchantId: Your Ingenico Merchant ID.

ingenicoWebhooksSecret: Your Ingenico Webhooks Secret.

4.1.4 SitePreferences custom properties

ingenicoApiService: The name of the service to use to connect to the Ingenico API.

ingenicoRequiresApproval: Whether approval is required before capturing the payment.

ingenicoVariantId: You can provide a Hosted Checkout variant ID in order to direct your registered consumer to a specific variant of the hosted payment pages.

ingenicoVariantIdGuest: You can provide a Hosted Checkout variant ID in order to direct your guest consumer to a specific variant of the hosted payment pages.

4.1.5 ingenicoNotification Custom Object

Webhooks received from the Ingenico API will be temporarily saved in this custom object for further processing.

createTime: UTC date time (ISO-8601) when the event was registered on the webhooks system.

orderNumber: Order number.

invoiceNumber: Invoice number.

merchantId: The ID of a merchant.

payload: Event content retrieved from the webhook. This is a string representation of the JSON object.

processed: Indicates if the Notification has been processed by a job.

sortKey: Contains the createTime without the last "Z". It can be used for sorting on the createTime.

transactionId: The transaction ID

type: Type of the event in the format: [object].[event]. For example: "payment.create".

4.2 Logs

The Ingenico LINK cartridge logs data to the custom logs using the "Ingenico" Log Category. The following Log Levels are used:

debug: Contains detailed information regarding requests to the Ingenico API and received Ingenico webhooks. For example, issues regarding the webhook checksum or JSON Web Token are logged in this log file.

info: These logs will contain information like the result for a processed webhook.

warn: Warnings will be logged here.

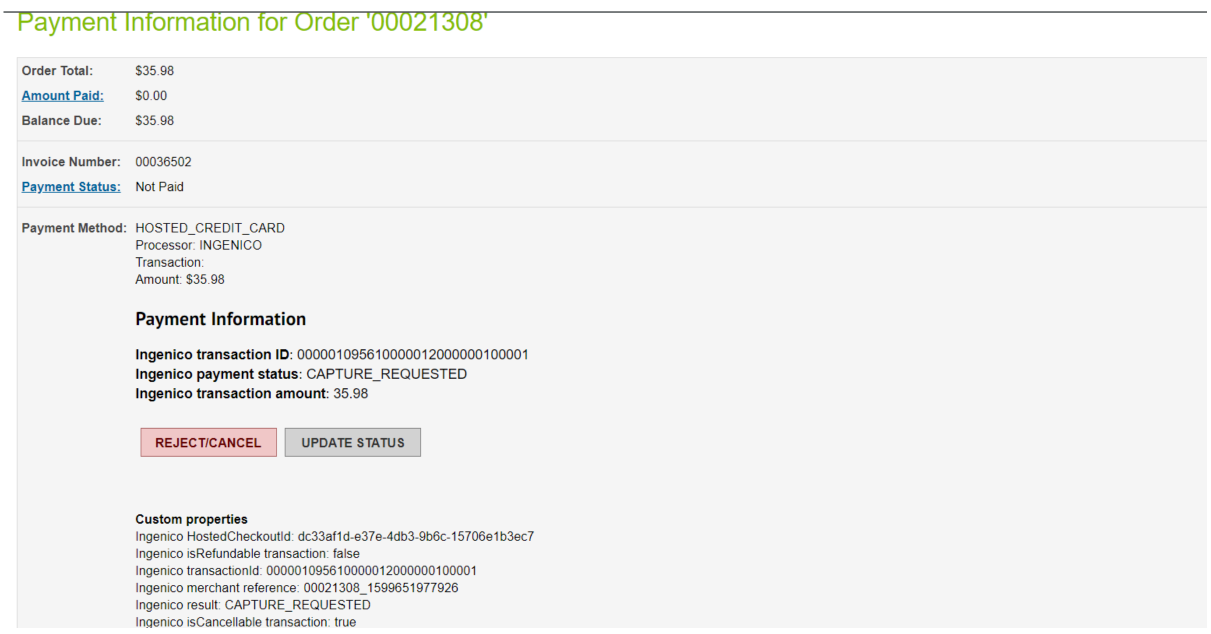
error: Errors will be logged here.

5. User Guide

5.1 Business Manager

The Ingenico cartridge does extend the functionality of the Business Manager. The bm_ingenico cartridge does contain an example of how to render relevant Ingenico-specific payment details in the payment tab of an order.

This is an example of the payment tab of the order with extended functionality:



Payment Information for Order '00021308'

Order Total:	$35.98
Amount Paid:	$0.00
Balance Due:	$35.98
Invoice Number:	00036502
Payment Status:	Not Paid
Payment Method:	HOSTED_CREDIT_CARD Processor: INGENICO Transaction: Amount: $35.98

Payment Information

Ingenico transaction ID: 000001095610000012000000100001
Ingenico payment status: CAPTURE_REQUESTED
Ingenico transaction amount: 35.98

Custom properties
Ingenico HostedCheckoutId: dc33af1d-e37e-4db3-9b6c-15706e1b3ec7
Ingenico isRefundable transaction: false
Ingenico transactionId: 000001095610000012000000100001
Ingenico merchant reference: 00021308_1599651977926
Ingenico result: CAPTURE_REQUESTED
Ingenico isCancellable transaction: true