**HiPay Integration**

Version 20.1.1

****

Table of Contents

[1. Summary 1-4](#_Toc40272892)

[2. Component Overview 2-5](#_Toc40272893)

[2.1 Functional Overview 2-5](#_Toc40272894)

[2.1.1 Frontend and Business Manager Functionalities 2-5](#_Toc40272895)

[2.1.2 Security Features and Functionalities 2-6](#_Toc40272896)

[2.1.3 Payment Configurations 2-6](#_Toc40272897)

[2.2 Use Cases 2-7](#_Toc40272898)

[2.3 Limitations, Constraints 2-12](#_Toc40272899)

[2.4 Compatibility 2-13](#_Toc40272900)

[2.5 Privacy, Payment 2-13](#_Toc40272901)

[3. Implementation Guide 3-14](#_Toc40272902)

[3.1 Integration Efforts 3-14](#_Toc40272903)

[3.2 Setup 3-14](#_Toc40272904)

[3.2.1 Installation 3-14](#_Toc40272905)

[3.2.2 Metadata Import 3-14](#_Toc40272906)

[3.3 Configuration 3-15](#_Toc40272907)

[3.4 Custom Code (20.1 version) 3-19](#_Toc40272908)

[3.4.1 Controller Changes 3-19](#_Toc40272909)

[3.4.2 Template Changes 3-19](#_Toc40272910)

[3.4.3 Model Changes 3-19](#_Toc40272911)

[3.4.4 Script Changes 3-19](#_Toc40272912)

[3.4.5 Client-Side Java Script Changes 3-20](#_Toc40272913)

[3.4.6 Form Changes 3-21](#_Toc40272914)

[3.5 Custom Code (19.1 and previous versions) 3-21](#_Toc40272915)

[3.5.1 Controller Changes 3-21](#_Toc40272916)

[3.5.2 Template Changes 3-25](#_Toc40272917)

[3.5.3 Model Changes 3-29](#_Toc40272918)

[3.5.4 Script Changes 3-29](#_Toc40272919)

[3.5.5 Client-Side Java Script Changes 3-30](#_Toc40272920)

[3.5.6 Form Changes 3-34](#_Toc40272921)

[3.6 External Interfaces 3-35](#_Toc40272922)

[3.7 Testing 3-35](#_Toc40272923)

[4. Operations, Maintenance 3-36](#_Toc40272924)

[4.1 Data Storage 3-36](#_Toc40272925)

[4.1.1 Orders 3-36](#_Toc40272926)

[4.1.2 ClearHungOrders job 3-37](#_Toc40272927)

[4.2 Availability 3-38](#_Toc40272928)

[4.3 Support 3-38](#_Toc40272929)

[5. User Guide 5-39](#_Toc40272930)

[5.1 Roles, Responsibilities 5-39](#_Toc40272931)

[5.2 Business Manager 5-39](#_Toc40272932)

[5.2.1 HiPay business module 5-39](#_Toc40272933)

[5.2.2 HiPay Site Preferences 5-43](#_Toc40272934)

[5.2.3 Services 5-46](#_Toc40272935)

[5.2.4 Schedules 5-48](#_Toc40272936)

[5.2.5 Payment Processors 5-48](#_Toc40272937)

[5.2.6 Payment Methods 5-49](#_Toc40272938)

[5.2.7 Logs 5-50](#_Toc40272939)

[5.2.8 Notification url 5-50](#_Toc40272940)

[5.2.9 Custom CSS configuration 5-51](#_Toc40272941)

[5.2.10 Products configuration 5-52](#_Toc40272942)

[5.3 Storefront Functionality 5-52](#_Toc40272943)

[5.3.1 Merchant Payment API - Credit Cards 5-52](#_Toc40272944)

[5.3.2 Payment API - Merchant Link Credit Cards (with 3DSecure) 5-53](#_Toc40272945)

[5.3.3 Hosted Merchant Link 5-54](#_Toc40272946)

[5.3.4 Hosted Merchant Link Credit Cards 5-54](#_Toc40272947)

[5.3.5 Hosted Merchant iFrame 5-55](#_Toc40272948)

[6. Known Issues 6-57](#_Toc40272949)

[7. Release History 7-57](#_Toc40272950)

# Summary

HiPay Fullservice offers tailored payment solutions for retailers, with functionalities adapted to today’s e-commerce landscape. HiPay provides a single integration solution for the most relevant domestic and international payment methods in each market.

The HiPay cartridge enables merchants to use HiPay full service as a payment option on their Salesforce Commerce Cloud storefronts. The cartridge provides several options for entering payment information (credit card or debit card information) during the checkout process: through a hosted page that is, external to the Salesforce Commerce Cloud platform or directly on the merchant’s Salesforce Commerce Cloud storefront, through API module calls. Beside this, a full range of other payment methods can be supported.

This document fully describes how to install the cartridge and integrate it into the online store. It is required for the merchant to first contact HiPay and request an account for the integration to work properly. HiPay shall provide a testing and production accounts for the merchant.

There are also configurations to be made in both Salesforce Commerce Cloud Business Manager and HiPay back office.

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

* cartridge called ‘int\_hipay\_sfra’ – core integration cartridge
* cartridge called ‘bm\_hipay\_controllers’ - extension for the cartridge within Business Manager
* ‘site\_template’ folder with metadata

The integration is based on the RefArchGlobal demo store provided by Salesforce Commerce Cloud.

The HiPay cartridge supports next payment methods:

|  |  |
| --- | --- |
| * Credit Card * Belfius Direct Net * Giropay * PayPal * iDEAL * ING HomePay * Klarna * MasterCard * Przelewy24 * QIWI Wallet * Sisal * Sofort Überweisung * WebMoney Transfer * Yandex Money * Oney Facily Pay (3xcb) * Oney Facily Pay (3xcb-no-fees) * Oney Facily Pay (4xcb) * Oney Facily Pay (4xcb-no-fees) | * Hosted Credit Cards * Hosted Belfius Direct Net * Hosted Giropay * Hosted iDEAL * Hosted ING HomePay * Hosted Klarna * Hosted Przelewy24 * Hosted QIWI Wallet * Hosted Sisal * Hosted Sofort Überweisung * Hosted WebMoney Transfer * Hosted Yandex Money * Hosted Oney Facily Pay (3xcb) * Hosted Oney Facily Pay (3xcb-no-fees) * Hosted Oney Facily Pay (4xcb) * Hosted Oney Facily Pay (4xcb-no-fees) |

# Component Overview

## Functional Overview

This section describes the functionalities that HiPay cartridge offers for the merchant.

### Frontend and Business Manager Functionalities

**Business Manager Extension for HiPay**

Merchants can configure the HiPay cartridge features from the Salesforce Commerce Cloud BM. On the Business Manager page each merchant can configure the connection to HiPay. The merchant will have to enter his credentials to create a connection to HiPay.

**Hosted Pages**

The cartridge supports the hosted payment page approach of entering payment information directly on pages that are hosted and provided by HiPay. During the checkout process, when the customer is asked to enter the payment details, he/she will be redirected to the HiPay hosted payment pages where he/she can select the preferred payment method. Each merchant can choose what payment methods shall be offered to the customer. The hosted payment pages can be customized in order to match the merchant storefront style.

**iFrame Integration**

The cartridge enables the merchant to use the HiPay payment solution via an iFrame that is integrated into his own Salesforce Commerce Cloud implementation.

**Multiple Payment Methods**

The cartridge allows users to complete the checkout process by using a wide variety of payment methods.

**Direct Payment API**

The cartridge offers to customers the possibility of filling in the payment information directly on the merchant’s website; in this case, the merchant will have to be PCI compliant. The transaction will be validated through the module called HiPay Fullservice API.

The cartridge can provide several options for entering payment details (credit card or debit card information) during the checkout process: through a hosted page, external to Salesforce Commerce Cloud platform or directly on the merchants’ Salesforce Commerce Cloud storefront, through API module calls. Additionally, a full range of other payment methods can be supported.

### Security Features and Functionalities

**3D Secure**

The cartridge supports 3D secure rules and activation where it is available on the card used. The 3D secure authentication process redirects the customer to a special page that is hosted by the card company that issued the customers’ credit or debit card. The cartridge enables the redirection to that page. After the authentication is completed, the cartridge has to ensure that the result of the authentication process is transferred back to Salesforce Commerce Cloud, and after the process is completed, the right message will be displayed on the frontend and the status of the order is set accordingly.

When using 3D Secure the merchant can set specific rules. These rules can be created, edited or deleted in the configuration parameter named “rules 3d secure”.

**PSD2 and Strong Customer Authentication – 3D Secure 2 compliance and guidance**

In effect since the beginning of 2018, the second Payment Services Directive (PSD2) redefines security standards for online payments. Given the strong growth of e-commerce in Europe, it aims to increase security during payment processing, while fighting more actively against fraud attempts.

For more information about these evolutions, please read our dedicated support page: <https://support.hipay.com/hc/en-us/articles/360005968459-PSD2-and-Strong-Customer-Authentication-3-D-Secure-v2-compliance-and-guidance>

**Device Fingerprint**

The cartridge supports device fingerprint information by sourcing the dynamically generated JavaScript from HiPay TPP. The JavaScript determines the available information and creates or generates a so called “black box”. For the usage of this functionality custom configurations are needed which require individual implementation for each merchant, as they are not part of the standard development.

**Signature Verification**

A signature verification mechanism is available for verifying the content of requests and redirections between the merchant site and the HiPay pages. The first step is for the merchant to set a secret password or phrase in the HiPay backend. The secret password or phrase will be used to generate a unique string that will be hashed with a SHA algorithm.

**Automatic Cancel**

Pending orders are automatically cancelled if they are not completed within 30 minutes.

### Payment Configurations

These cartridge supports both authorization mode and direct capture mode.

**Direct Capture**

In this case the capture is requested automatically after the authorization. If the capture is not successful, the customer (the buyer) is redirected to an error page. If the capture is successful, the customer is redirected to a success page and completes the checkout process.

**Authorization**

In this case the customer (the buyer) is not charged automatically or direct after the order is placed. The merchant has a defined amount of time to capture the amount money from the customer. This can be configured manually from Business Manager > HiPay Integration > Order capture; partial or full capture action can be done; Order Cleanup Time can be configured from **Hung Order Cleanup Time** configuration option.

If the transaction fails, the customer is redirected to an error page. If the transaction is successful, the customer is redirected to a success page. The authorization can be used within an interval of 7 days otherwise it will be cancelled.

## Use Cases

**Short description of the HiPay payment system**

HiPay generally can receive payments in 2 ways:

* via the HiPay Direct Payment API. A result is returned right away. This can only be used for Credit Cards.
* via the 'redirect model' where a form containing certain fields is posted to HiPay and the shopper is redirected to HiPay's Hosted Payment Page (HPP). After completing the payment, the shopper is returned to the shop's resultURL. This resultURL can be configured and contains the result of the payment (success or fail). The redirect model can also be implemented in an iFrame solution where the merchants’ customer is redirected to HiPay pages, but these pages render within a frame that is inserted inside the merchant’s page. This approach gives the merchant’s customer the feeling that he has not left the merchant’s site.

Compared to the API implementation, the redirect model supports more payment methods as well as some other functionality.

Payment results should be sent asynchronously to a 'notification URL' at the Merchant (SFCC) site. Payments can change status over the time (Authorized, Refunded, Cancelled and others) on the HiPay side, and these statuses are updated automatically on the Salesforce Commerce Cloud side through the sending update notifications. HiPay Fullservice is sending notifications to Salesforce Commerce Cloud for each event occurred/ transaction changed. In order to handle all notifications, notes are added to the order.

In both cases, the 'two phase ordering' approach is used. The order is first created (but only 'placed', status is “Created”) and after the payment has been authorized (which is now standard behavior in Site Genesis) the order can be captured, and its status will change to „Paid”. In cases where payment cannot be authorized, the order will fail and its status will change to “Failed”.

Registered and unregistered shoppers will be able to follow the standard Site Genesis flow with the following changes:

HiPay redirect (hosted payment pages, iFrame integration and multiple payment methods) and the HiPay Direct Payment API will be the only payment option.

**If HiPay redirect is enabled**

* when the user selects this payment method, no credit card details will be entered on the Billing page (this is done on HiPay hosted payment pages)
* the possibility to choose one of the predefined payment methods on the Billing page with further redirection to HiPay hosted payment pages, if Directory Lookup is enabled and the user selected HiPay payment method
* a redirect to HiPay hosted payment pages after clicking the ‘SUBMIT ORDER’ button on the Order Confirmation page
* a return to the SFCC Order Summary page after successful payment authorization
* a return to the SFCC Order Confirmation page, if the user cancelled the payment on HiPay hosted payment pages
* a return to the SFCC Order Confirmation page with an error message displayed, if the payment was refused

**If HiPay API payments are enabled**

* credit card details are entered and stored in your SFCC shop
* for the other payment methods, you can still redirect to the HiPay hosted payment pages, but can disable credit cards via the configuration of your skin

**Sample of testing scenarios**

* All test scenarios involve successful and declined payments
* Test scenarios should be done with every available payment method
* The used currency should be EUR, USD has only credit cards enabled
* If for a payment method on a hosted page there is an emulation page, all options (e.g. cancel, exception, decline) should be used as an outcome to test how the system will behave
* When making a payment a Salesforce Commerce Cloud Order object changes its status as follows
  + CREATED - the IFrame or Hosted page has been shown – the Basket has been cleared – the order may stay/hang in this status if the process did not complete properly (bug) or the user abandons the Iframe or Hosted page and does not complete the payment
  + NEW – the Iframe or Hosted page has been closed with an accept – the Order changed it’s status
  + OPEN – the Order has been viewed in SFCC Business Manager after it was in status NEW
  + FAILED – all cases of cancel and decline

| **UC - 1** | **HiPay Hosted Page** |
| --- | --- |
| This use case describes the high-level steps in which a registered/guest customer successfully creates an order and the order with HiPay Hosted Page. | Preconditions   * Go to Merchant Tools > Site Preferences > Custom Preferences > HiPay Settings * For HiPay Operation Mode, select hosted (Hosted Page) and click on Apply  1. A registered customer navigates to the site, an item, add it to the cart and proceeds to the cart page. 2. The customer clicks on the **Checkout** button and fills in the shipping form requirements. 3. The customer clicks on the **Continue** button and proceeds to fill in the billing form requirements and for payment method, select “**HiPay Hosted**”. 4. The customer clicks on the **Continue** button, proceeds to the **Payment** page and clicks on the **Place order** button. 5. A HiPay hosted page is opened and the customer enters the card details. 6. After successful payment the customer gets redirected to the Summary page and the thank you message is successfully loaded. 7. The customer can verify Order Summary information 8. Merchant can verify Order status in SFCC BM 9. Merchant can verify information in HiPay Fullservice account   **Note that a similar flow can be done for guest checkout** |

| **UC - 2** | **HiPay Hosted Page with 3-D Secure** |
| --- | --- |
| This use case describes the high-level steps in which a registered/guest customer successfully creates an order and the order with HiPay Hosted Page with 3-D Secure. | Preconditions   * Go to Merchant Tools > Site Preferences > Custom Preferences > HiPay Settings * For HiPay Operation Mode, select hosted (Hosted Page) and click on Apply * For 3-D Secure, select “1 (3-D Secure authentication if available)” and click on “Apply”  1. A registered customer navigates to the site, an item, add it to the cart and proceeds to the cart page. 2. The customer clicks on the **Checkout** button and fills in the shipping form requirements. 3. The customer clicks on the **Continue** button and proceeds to fill in the billing form requirements and for payment method, select “**HiPay Hosted**”. 4. The customer clicks on the **Continue** button, proceeds to the **Payment** page and clicks on the **Place order** button. 5. A HiPay hosted page is opened and the customer enters the 3-D Secure card details. 6. The customer is redirected to the card vendor’s site and it enters the 3-D Secure password. 7. After successful identification the customer clicks on “Back to Payment”. 8. After successful payment the customer gets redirected to the Summary page and the thank you page is successfully loaded. 9. The customer can verify Order Summary information 10. Merchant can verify Order status in SFCC BM 11. Merchant can verify information in HiPay Fullservice account   **Note that a similar flow can be done for guest checkout** |

| **UC - 3** | **HiPay API Integration** |
| --- | --- |
| This use case describes the high-level steps in which a registered/guest customer successfully creates an order and the order with HiPay API Integration. | Preconditions   * Go to Merchant Tools > Site Preferences > Custom Preferences > HiPay Settings * For HiPay Operation Mode, select api (API) and click on Apply  1. A registered customer navigates to the site, an item, add it to the cart and proceeds to the cart page. 2. The customer clicks on the **Checkout** button and fills in the shipping form requirements. 3. The customer clicks on the **Continue** button and proceeds to fill in the billing form requirements and for payment method, select “iDEAL” 4. The customer selects the desired issuing bank and clicks on the **Place order** button. 5. The customer is redirected to the iDEAL vendor site. 6. The customer enters the payment details. 7. After successful payment the customer gets redirected to the Summary page and the thank you message is successfully loaded. 8. The customer can verify Order Summary information 9. Merchant can verify Order status in SFCC BM – in Order screen, open Payment tab to verify the payment information. 10. Merchant can verify information in HiPay Fullservice account   **Note that a similar flow can be done for guest checkout** |

## Limitations, Constraints

* The merchant needs a configured HiPay account
* The merchant can style the hosted pages to make it match the look and feel of their store.
* The following features are out of scope:
  + Cancelled Order support
  + Refund support
  + One-Click Payment (credit card only)

## Compatibility

The HiPay integration cartridge is compatible with the latest Salesforce Commerce Cloud API version, currently 19.10 and the Storefront Reference Architecture version 4.4.1.

## Privacy, Payment

When the redirect method is used, all payment data is entered into the HiPay hosted pages by the customer and no Credit Card data is stored in Salesforce Commerce Cloud (other than the brand of the used card).

Examples:

HiPay Product Name: hosted

HiPay Payment Product List: bcmc,cb,maestro,mastercard,visa,american-express

HiPay Payment Category List: credit-card,debit-card

HiPay Product Name: dexia-directnet

HiPay Payment Product List: dexia-directnet

HiPay Payment Category List: realtime-banking

# Implementation Guide

## Integration Efforts

The following steps are needed to complete the integration:

* Install the cartridge
* Change and import ‘site-template.zip’ (system objects)
* Do the required code changes
* Configure your merchant's HiPay Account in the HiPay back office
* Configure HiPay parameters in Salesforce Commerce Cloud BM
* Test

## Setup

### Installation

Install the “int\_hipay\_sfra”, “bm\_hipay\_controllers” cartridges from the distributive zip-archive in the standard way using the Salesforce Commerce Cloud UX-studio.

### Metadata Import

Open the folder ‘metadata’ in the installation package. Please review the ‘site\_template\_sfra’ folder, do the necessary modifications if required:

1. **Rename the “RefArchGlobal” folder (under the site\_template\_sfra/sites folder) to the ID of your site.**
2. **In the file “jobs.xml”, change the “RefArchGlobal” in the markups** <context site-id="RefArchGlobal"/> **to the ID of your site.**
3. **In the file “services.xml”, change the “RefArchGlobal” in the markups** <service service-id="hipay.rest.\*.RefArchGlobal"> **to the ID of your site.**
4. **If you don’t need** ‘pin-price-lists.xml’ with pricebook, you may just remove this file and the ‘pricebooks’ folder.

For multiple sites integration, you will also need to repeat these steps for each site.

Then, archive the ‘site\_template\_sfra’ folder into a file named site\_template\_sfra.zip.

Before importing the ‘site-template\_sfra.zip’, check and save the cartridge paths of your site and Business Manager, because these will be modified. By saving them elsewhere you will be able to configure them manually if something goes wrong. Also, check the fields which will be updated after importing the file ‘site-template\_sfra.zip’ and be sure that there will be no conflict with any existing fields.

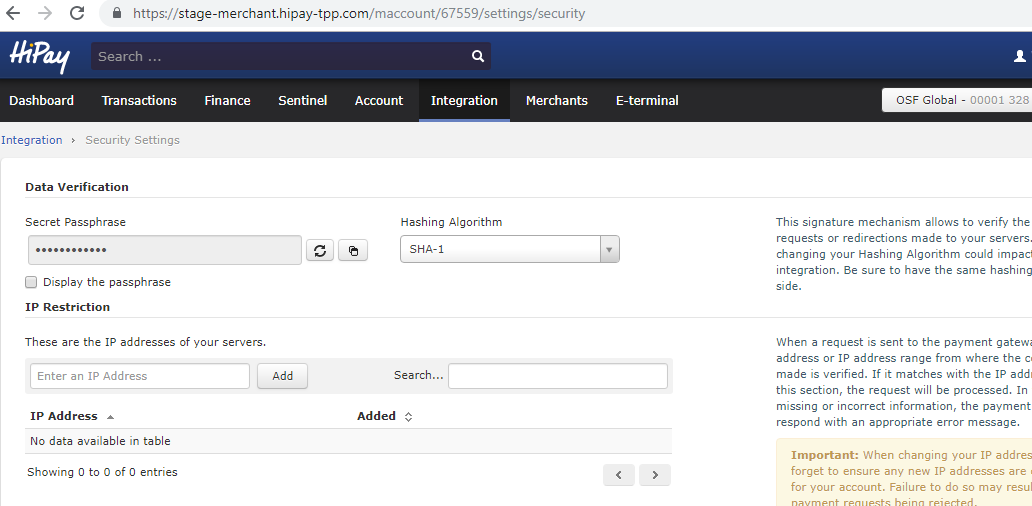
Finally, import it through BM Administration > Site Development > Site Import & Export section: Browse your local file and click “Upload”, then select it and click “Import” and confirm the import by clicking “OK”. The import lasts a few minutes only.

## Configuration

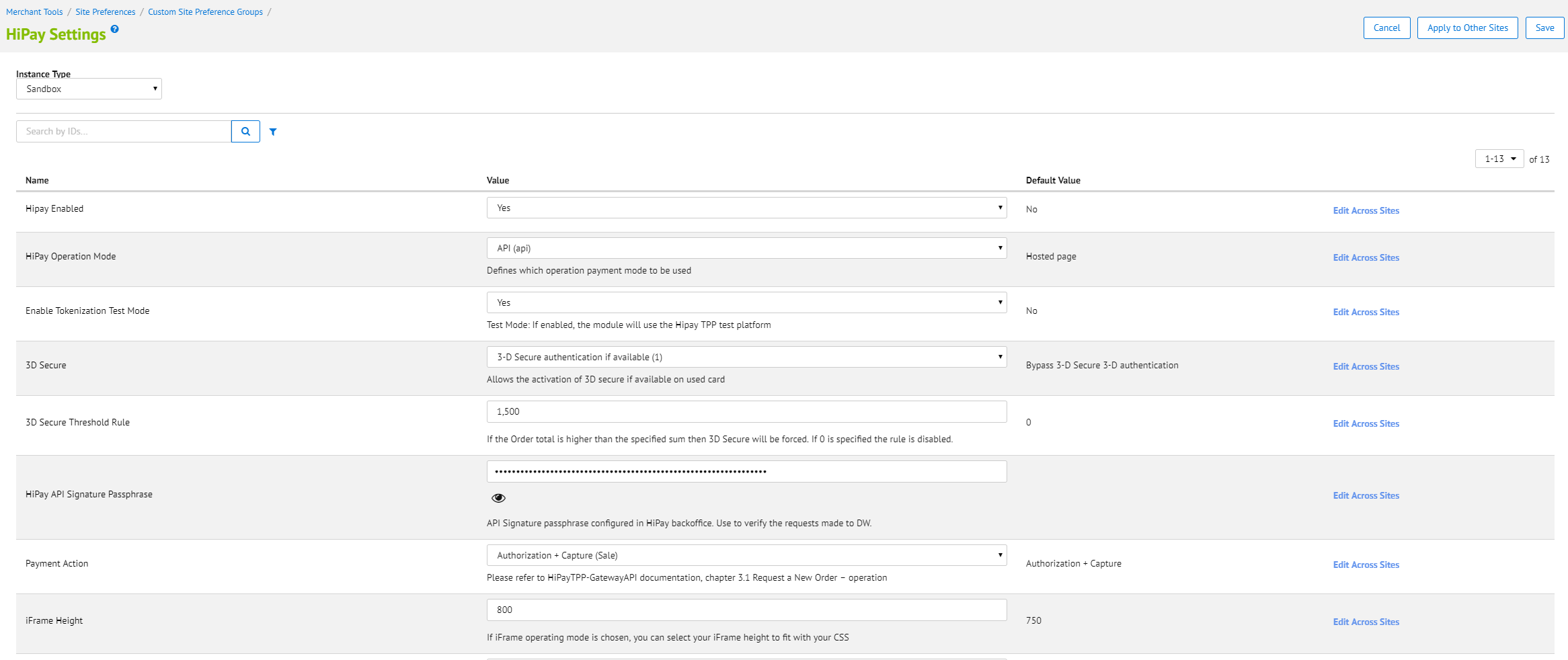
* Add **int\_hipay\_sfra** to the cartridge site path:  
  Go to BM > Administration > Sites > Manage Sites, under Storefront Sites click your site name, then click the Settings tab and add “int\_hipay\_sfra:” at the beginning of the “Cartridges” field.
* Add **bm\_hipay\_controllers** to the cartridge BM path:  
  Go to BM > Administration > Sites > Manage Sites, under Business Manager Sites click “Business Manager” link, while in the Settings tab add “bm\_hipay\_controllers:” at the beginning of the “Cartridges” field.
* Custom site preferences (HiPay preferences):

**Merchant Tools** > **Site Preferences** > **Custom Preferences** (or Custom Site Preference Groups) > HiPay (HiPay Settings):

* + **Hipay Enabled** – Set to Yes.  
    *Enable/disable HiPay.*
  + **Enable One-Click payments** – Set to Yes.  
    *Enable/disable One-Click payments.*
  + **HiPay Operation Mode**  
    *Defines which operation payment mode to be used – these are HOSTED/IFRAME/API.*  
    *When selecting the API mode then the HIPAY\_HOSTED payment method will be hidden from Billing page. If HOSTED or IFRAME is selected, then all API payment methods are hidden from the Billing page.*
  + **Enable Tokenization Test Mode** – Set to Yes.  
    *Test Mode: If enabled, the module will use the HiPay TPP test platform.*
  + **3D Secure** -Set to ‘Bypass 3-D Secure 3-D authentication (0)’.  
    *Allows the activation of 3D secure if available on used card.*
  + **3D Secure Threshold Rule** – Set to 100.  
    *If the Order total is higher than the specified sum, then 3D Secure will be forced.*  
    *If 0 is specified, the rule is disabled.*
  + **HiPay API Signature Passphrase**  
    *API Signature passphrase configured in HiPay backoffice. Use to verify the requests made to SFCC.* *Can be find under the Integration -> Security Settings -> Secret Passphrase section.*

****

* + **Payment Action** -Select ‘Authorization + Capture (Sale)’.  
    *Please refer to documentation on HiPay portal, Payments section (*[*https://developer.hipay.com/doc-api/enterprise/gateway/*](https://developer.hipay.com/doc-api/enterprise/gateway/)*).*
  + **iFrame Height** - Set 750.  
    *If iFrame operating mode is chosen, you can select your iFrame height to fit with your CSS.*
  + **iFrame Width** - Set 900.  
    *If iFrame operating mode is chosen, you can select your iFrame width to fit with your CSS.*
  + **HiPay CSS content** - May be set to empty.  
    *CSS applied to either Hosted or IFrame page, not enclosed in style tags.*
  + **Display card selector** – Set to Yes.  
    *Enable/disable the payment methods selector on iFrame and Hosted page.*
  + **Hung Order Cleanup Time** - Set to 30.  
    *The time in minutes after which all Orders hung in status CREATED that are left by the payment processing are cleaned.*
  + **List of shipping methods**  
    A comma-separated list of available shipping methods ids. Required for the Oney payments configuration BM module.



**Do not forget to click “Save” button at the top right of the page, in order to save your preferences!**

* **Review System Object Types configuration:**  
  Go to **BM > Administration > Site Development > System Object Types.**
  + In Profile, click “Attribute Definitions”. Check that you see a new attribute: **datePasswordLastChange**.  
    It is used for 3-D Secure 2.0 (PSD2), in order to detect if the customer changed its password recently.
  + In Product, click “Attribute Definitions”. Check that you see a new attribute: **productDematerialized**.  
    It is used for 3-D Secure 2.0 (PSD2), in order to process extra security checks if the product could be sent immediately to the customer.
* **Review Custom Object Types configuration:**  
  Go to **BM > Site Development > Custom Object Types**. In the list, check that you see a new object type: **SaveOneclick**.  
  It is used for 3-D Secure 2.0 (PSD2), in order to count the credit cards added by a customer during the last 24 hours.
* **Review Jobs configuration:**  
  You will need to assign and schedule our two jobs (ClearHungOrder and HiPayClear) to your site(s).  
  Please read the section 4.2.4 Schedules for this configuration.
* **Review Service configuration:**  
  You will need to update the services credentials values User and Password. You can take these values from the Hipay Dashboard, under **Integration > Security Settings > Api credentials**. Please note that two kind of values exists: “Private” Accessibility and “Public” Accessibility.  
    
  Please read the section 4.2.3 Services before applying the configuration below.
  + For Private Accessibility - Update fields User and Password for the following services:

**Administration > Operations > Services > Tab “Credentials” > hipay.hosted.cred**  
**Administration > Operations > Services > Tab “Credentials” > hipay.maintenance.cred**  
**Administration > Operations > Services > Tab “Credentials” > hipay.order.cred**

* + For Public Accessibility - Update fields User and Password for the following services:

**Administration > Operations > Services > Tab “Credentials” > hipay.token.cred**

## Custom Code (20.1 version)

### Controller Changes

* ***Account.js***  
  PATH: /cartridge/controllers/Account.js  
  File has been added for save the last date of modification/creation of password.
* ***CheckoutServices.js***  
  PATH: /cartridge/controllers/CheckoutServices.js  
  File has been extended for save the last date of modification/creation of password.

### Template Changes

* ***HiPayContent.isml***  
  PATH: \cartridge\templates\default\checkout\billing\paymentOptions\hiPayContent.isml  
  Fix minor XML issue(s).
* ***CreditCardForm.isml***  
  PATH: \cartridge\templates\default\checkout\billing\creditCardForm.isml  
  Fix minor XML issue(s).
* ***PaymentOptions.isml***  
  PATH: \cartridge\templates\default\checkout\billing\paymentOptions.isml  
  Add component SDKJS.
* ***DeviceFingerprint.isml***  
  PATH: \cartridge\templates\default\checkout\components\deviceFingerprint.isml  
  Fix minor XML issue(s).
* ***sdkjs.isml***  
  PATH: \cartridge\templates\default\checkout\components\sdkjs.isml  
  Add template SDKJS.

### Model Changes

No code changes

### Script Changes

* ***CheckoutHelpers.js***  
  PATH: /cartridge/scripts/checkout/checkoutHelpers.js  
  File has been extended to add a method: write To Custom Object.
* ***HiPayServiceInit.js***  
  PATH: /cartridge/scripts/init/hiPayServiceInit.js  
  For the services modify ‘Content-Type’ in json.
* ***HipayCheckoutModule.js***  
  PATH: /cartridge/scripts/lib/hipay/modules/hipayCheckoutModule.js  
  hiPayOrderRequest Function has been modified: params.eci to ‘String’.
* ***HiPayNotificationModule.js***  
  PATH: /cartridge/scripts/lib/hipay/modules/hipayNotificationModule.js  
  Remove timestamp for orderId.
* ***HiPayOrderModule.js***  
  PATH: /cartridge/scripts/lib/hipay/modules/hipayOrderModule.js  
  Remove timestamp for orderId.
* ***HiPayHostedService.js***  
  PATH: /cartridge/scripts/lib/hipay/services/hipayHostedService.js  
  Call service by json in parameter (not by string).
* ***HipayMaintenanceServicejs***  
  PATH: \cartridge\scripts\lib\hipay\services\hipayMaintenanceService.js  
  Call service by json in parameter (not by string).
* ***HipayOrderService.js***  
  PATH: \cartridge\scripts\lib\hipay\services\hipayOrderService.js  
  Call service by json in parameter (not by string).
* ***HipayTokenService.js***  
  PATH: \cartridge\scripts\lib\hipay\services\hipayTokenService.js  
  Call service by json in parameter (not by string).
* ***HipayHelperjs***  
  PATH: \cartridge\scripts\lib\hipay\hipayHelper.js  
  extended to prepare the DSP2 object to be sent to HiPay.
* ***HipayUtils.js***  
  PATH: \cartridge\scripts\lib\hipay\hipayUtils.js  
  HipayUtils.js to factorization of Hipay methods.

### Client-Side Java Script Changes

* ***HipayCheckout.js***  
  PATH: \cartridge\static\default\js\hipayCheckout.js  
  Config and call getBrowserInfo from SDKJS.

### Form Changes

* ***Billing.xml***  
  PATH: /cartridge/forms/default/billing.xml  
  File has been extended to add a field that manages “browserInfo”
* ***Billing.xml***  
  PATH: /cartridge/forms/fr\_FR/billing.xml  
  File has been extended to add a field that manages “browserInfo”

## Custom Code (19.1 and previous versions)

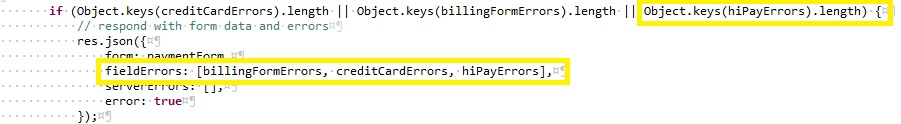
### Controller Changes

#### CheckoutServices.js

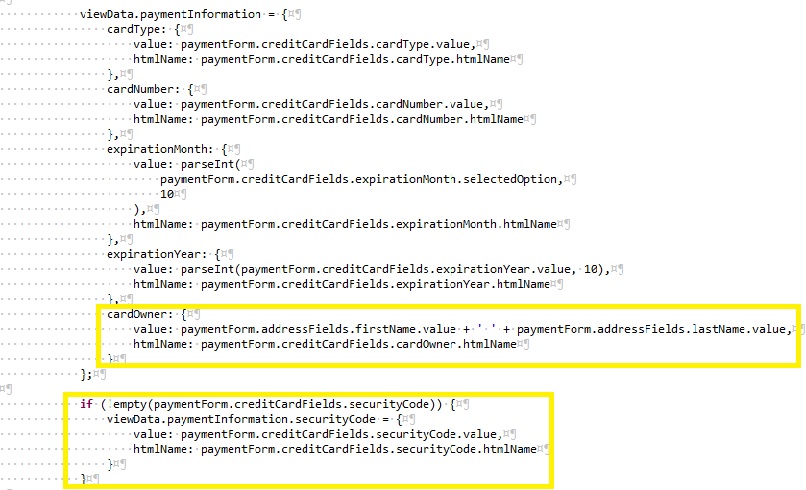
PATH: cartridge/controllers/CheckoutServices.js

* SubmitPayment function has been replaced with following code changes
  + Function has been extended to validate HiPay payment methods.





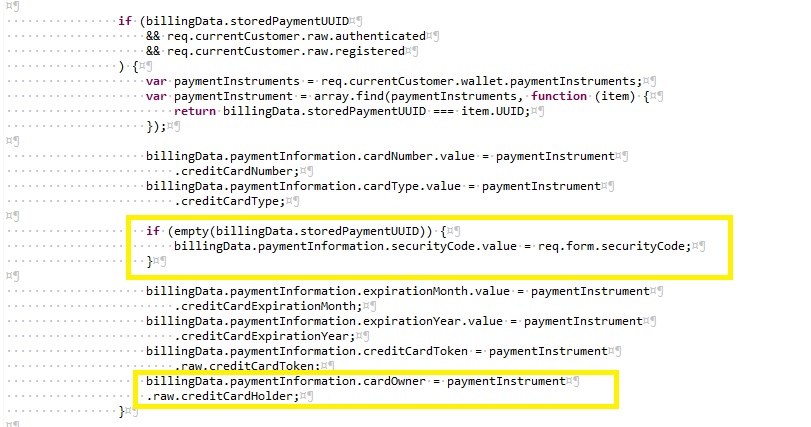
* + Card Owner has been added to the payment information view data.
  + Since the security code is not mandatory for some credit cards, it has been prevented to send null value if the security code is empty.



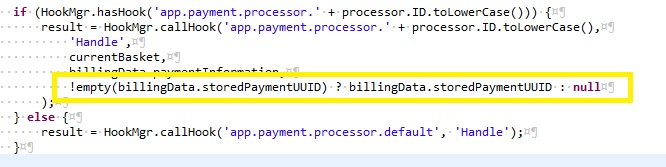
* + Since the email is mandatory for all requests to HiPay, email address field has been moved to address fields. Location of the email field can be customized according to client’s requirements.



* + Security code should not be sent if the credit card is already stored. Credit card holder is written to the billing data to use in the hipay hooks.



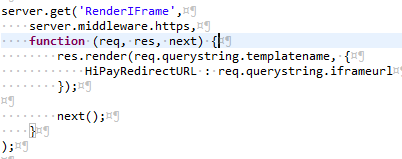
* + If the payment instrument is already stored, UUID is sent to the hook to use.



* + UUID has been sent to handlePayments function of ChekoutHelpers to determine if the payment is recurring or not.
  + errorMessage parameter has been updated to display the error message from Hipay.
  + HiPay snippet has been added to handle returning response from HiPay servers. If redirection URL has been sent from HiPay, user is redirected to the hosted page, if the operation mode is IFrame, user is redirected to the IFrame.



* + RenderIframe get method has been added to the CheckoutServices file to redirect the user to the Iframe page.



#### COPlaceOrder.js

PATH: cartridge/controllers/COPlaceOrder.js

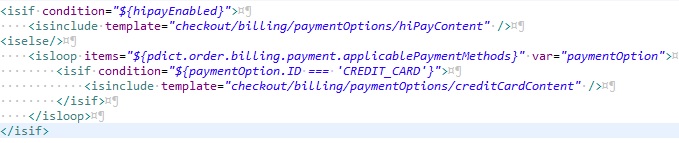
COPlaceOrder class has been added to the cartridge to handle hosted payments.

### Template Changes

#### paymentOptionsContent.isml

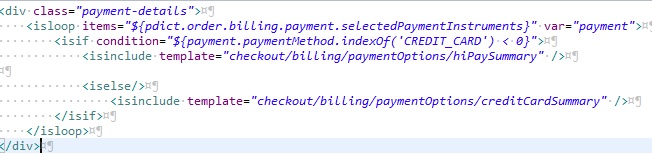
PATH: cartridge/templates/default/checkout/billing/paymentOptions/paymentOptionsContent.isml

* + If HiPay is enabled, hiPayContent template should be rendered.



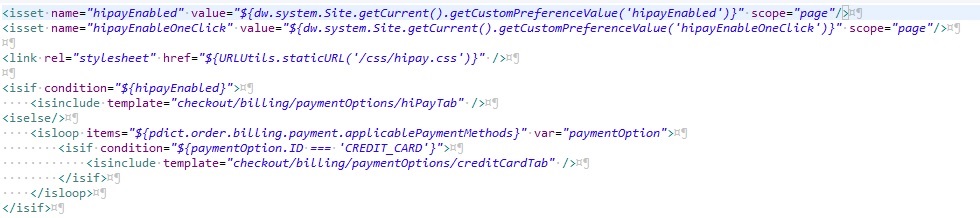
#### paymentOptionsSummary.isml

* + If condition has been added to paymentOptionsSummary template to display summary. If the payment method is credit card, credit card information display as it is on base cartridge. If the payment method is one of the HiPay specific methods, payment method name is displayed. Payment summary can be extended, based on business requirements.



#### paymentOptionsTabs.isml

* + If condition has been implemented to handle displaying of the payment method sections. If the HiPay is enabled hipayTab.isml is rendered which includes all payment methods by HiPay.



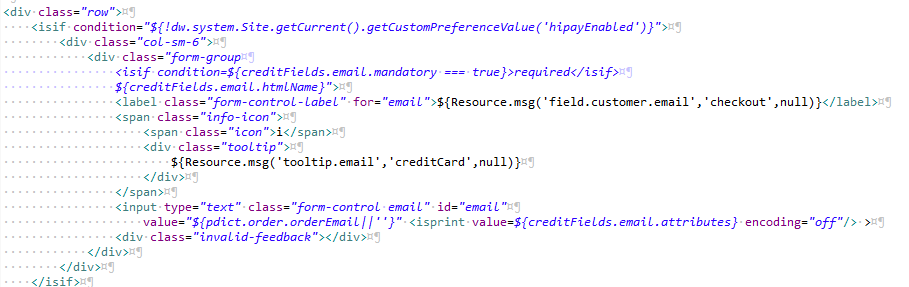
#### billingAddress.isml

* + Since email address field has been added to the billingAddress.isml. As default email is asked on billing address section however, it can be customized based on business requirements.



#### creditCardForm.isml

* + Since we ask the email information on billing address section, there is no need to ask for it on credit card form. This behavior can be extended by the integrator based on the business requirements. However, it should not be forgotten to send email as a parameter to HiPay server.



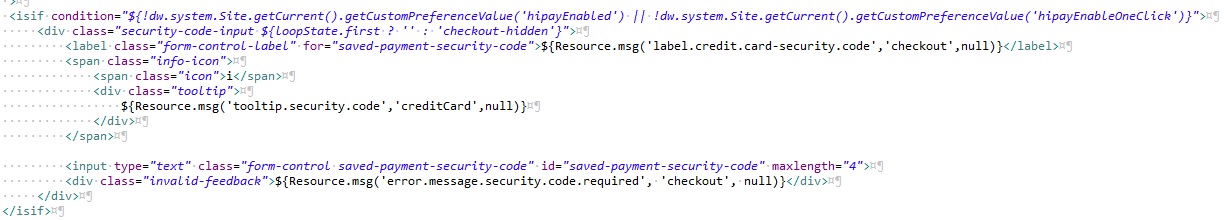
#### paymentOptions.isml

* + Script section has been added to the paymentOptions.isml to set HiPay specific variables which are used for template rendering. Also devicefingerprint.isml template has been included in the template for security purposes.



#### storedPaymentInstruments.isml

* + If statement has been added to the template, to not display stored payment instruments when the HiPay One Click is not enabled.



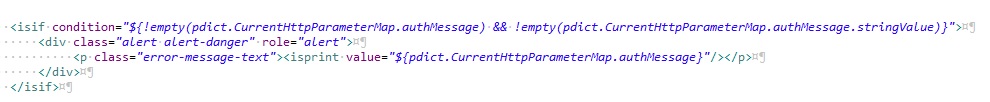
#### confirmationEmail.isml

* + If statement has been added to display selected payment method name, when the selected payment method is not credit card. This behavior can be extended based on business requirements.



#### checkout.isml

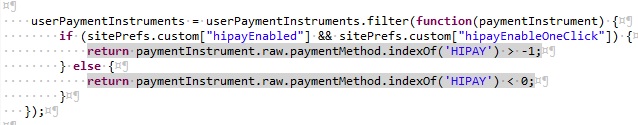
* + Section has been added to checkout.isml to display the error messages in the response from HiPay servers.



### Model Changes

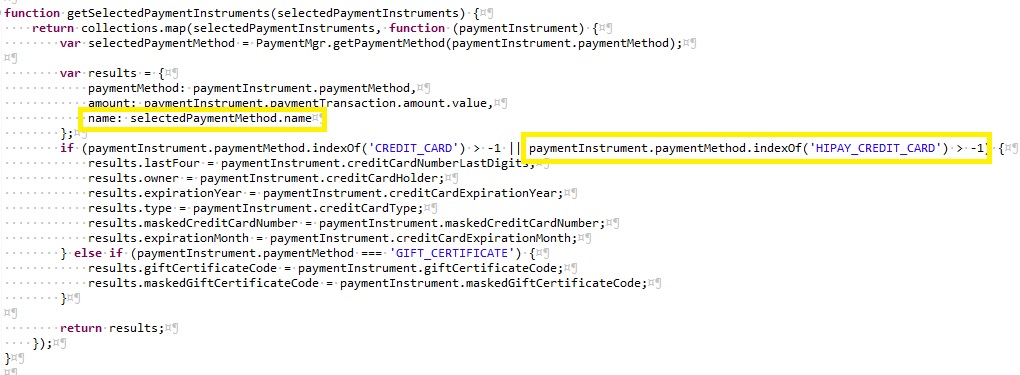
#### account.js

* + userPaymentInstruments has been filtered. If HiPay and HiPay One click is enabled for the site, HiPay payment methods are displayed as stored payment instruments.



#### payment.js

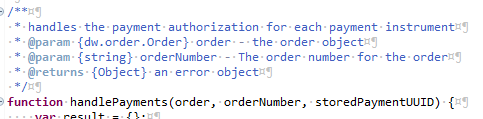
* + getSelectedPaymentInstruments function has been customized to set name node with selected payment method name and get the detailed information of Hipay credit cards.



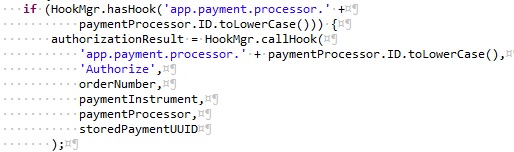
### Script Changes

#### checkoutHelpers.js

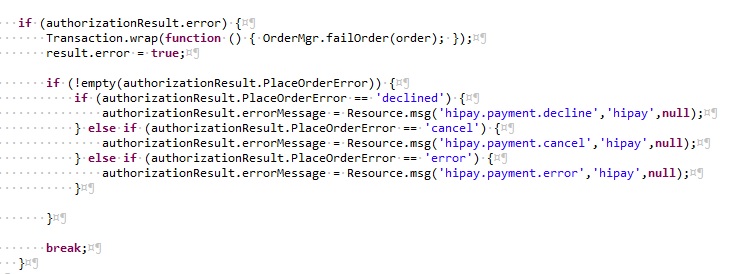
* + storedPaymentUUID has been added as a parameter for handlePayments function.



* + storedPaymentUUID has been added as parameter on calling Authorize hook to understand if the payment instrument is recurring or not.



* + Authorization error statement has been extended for displaying error messages which return from HiPay server.



### Client-Side Java Script Changes

#### billing.js

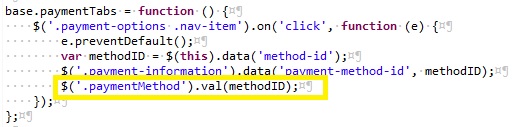
* + Since the email is mandatory on HiPay side, email field has been moved to billing address section and it is set with orderEmail by default. This behavior can be customized based on business requirements.



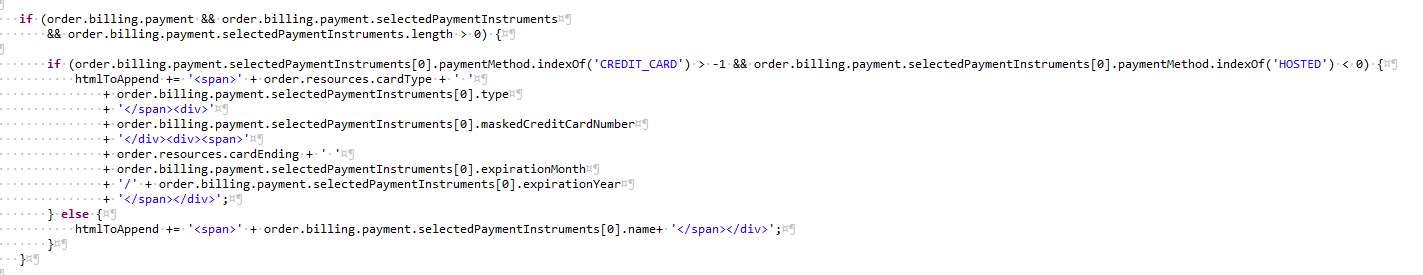
* + Cleave should not be called if the payment method is not credit card. If statements have been added for the purpose.



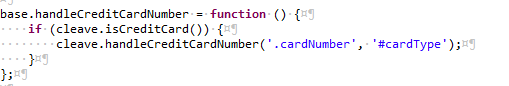
* + paymentTabs function has been extended to change payment method sections on click action of payment tabs.



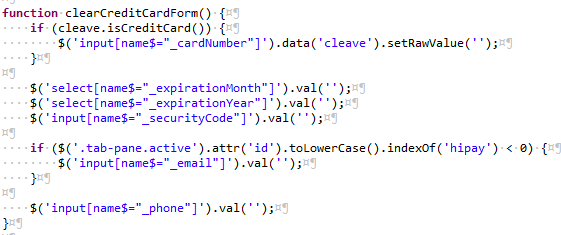
* + Customization has been done to write only the payment method name, if the selected method is not credit card. This behavior can be extended by the integrator.



* + Credit card check has been added into handleCreditCardNumber function.



* + The same check has been added for clearCreditCardForm. Also, if statement has been added to check if the hipay is enabled. If the HiPay is enabled, email field is not removed.

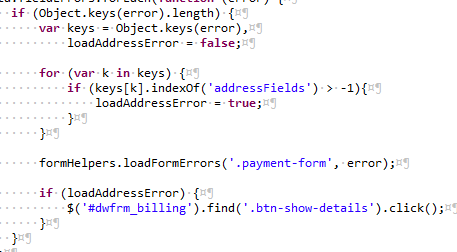


#### checkout.js

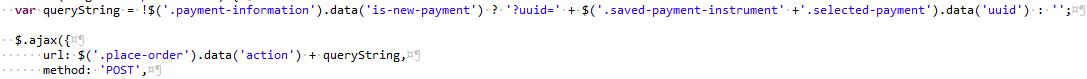
* + Since security code is not mandatory for all payment methods, if statements have been added to check if the security code is enabled or not.



* + Functionality has been added to expand billing address section if there is a missing field.

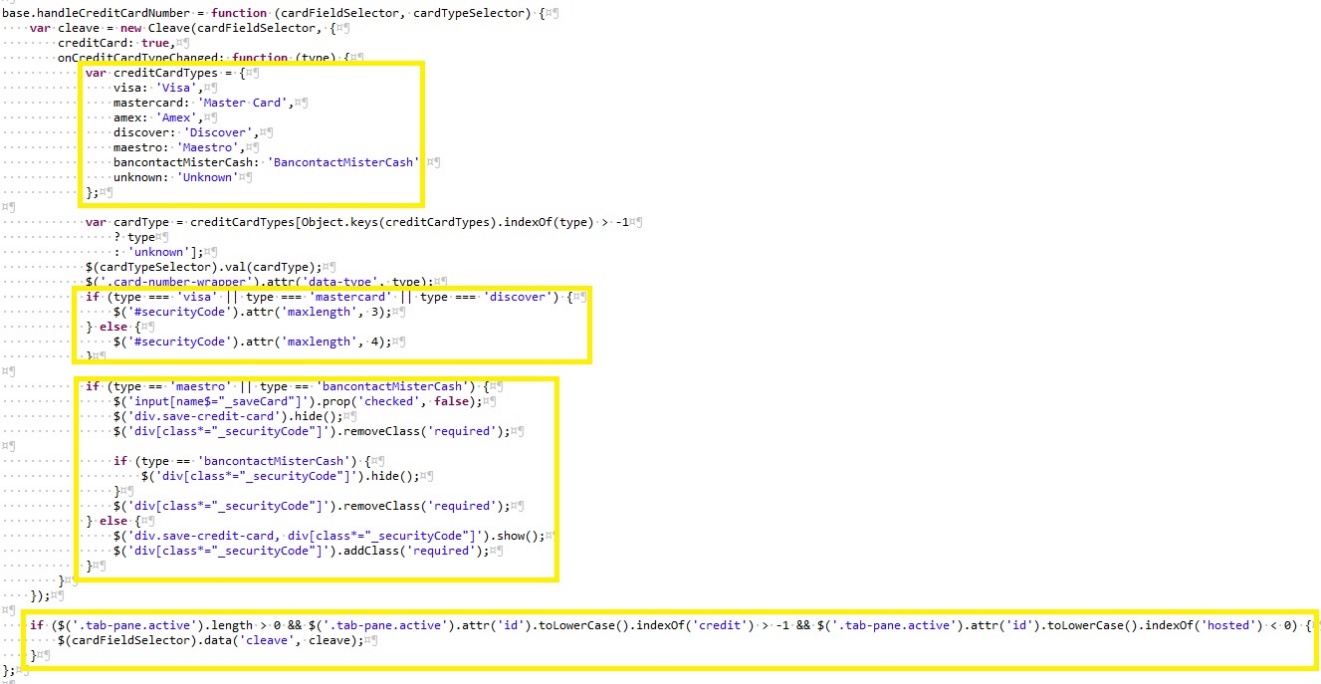


* + queryString is sent via AJAX request for getting the uuid of stored payment instrument. UUID is used in the HiPay hooks.



#### cleave.js

* + Cleave.js has been customized to recognize new credit card types and handling cvv and one click payment specifications.

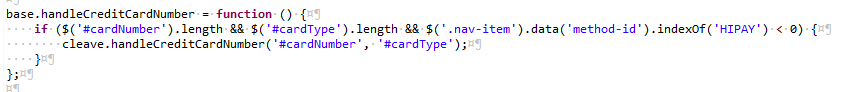


#### libCleave.js

* + Since npm cleave library does not have all credit card types which is included in HiPay, we prefer to use libCleave.js file which is customized based on HiPay needs.

#### paymentinstruments.js

* + Prevented calling handleCreditCardNumber method of cleave if the selected payment method is not credit card.



### Form Changes

Sample address.xml, billing.xml, creditCard.xml forms have been created. Forms should be modified by the integrator according to business requirements and specific forms should be created by the integrator i.e. Russian address form for the Russian payment methods.

#### address.xml

* Email field has been moved to address.xml. This field can be moved to any other forms; however, it should not be forgotten email information is mandatory for HiPay requests.

#### billing.xml

* deviceFingerprint and hipayMethodsFields forms have been included in the billing.xml

#### creditCard.xml

* Max lengths of card type and card owner fields have been increased according to HiPay need. Email field has been moved to the address.xml.

#### hipayMethods.xml

* Since the birthday and the house number is not asked by default on SFRA, those fields have been added in Klarna group. Since the form structures are totally depended on integrator, the fields can be moved to i.e. German address form.

## External Interfaces

None.

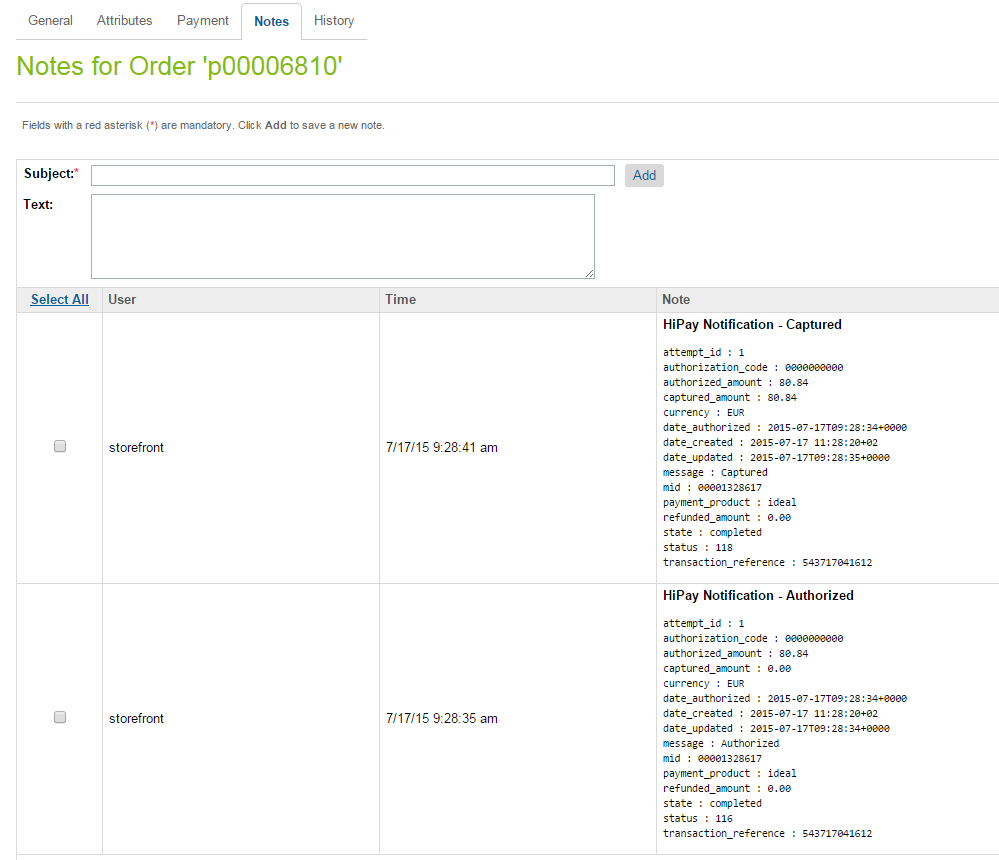
## Testing

Please refer to the use cases in section 2.2. If you have any issues, our technical team will work with you to ensure that everything is thoroughly tested and meets all requirements.

# Operations, Maintenance

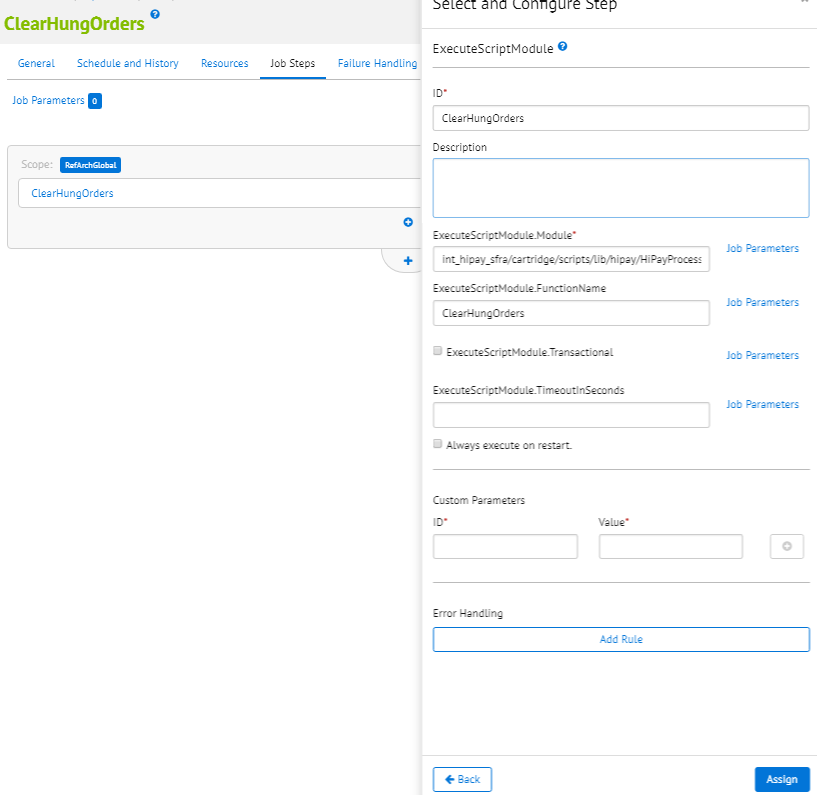
## Data Storage

### Orders

HiPay Fullservice sends notifications for each event that occurs. In order to handle all notifications, notes are added to the order. There should be one event logged for each notification.

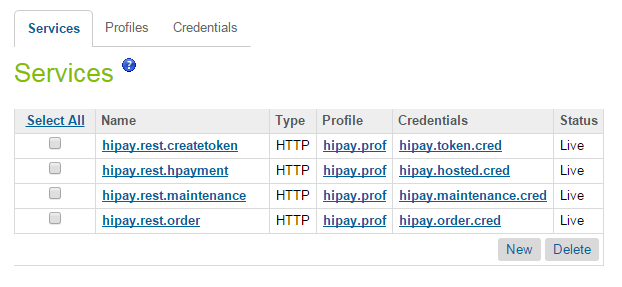
### ClearHungOrders job

When using the HiPay hosted page or an iFrame, the actual payment can be interrupted due to, for example, the user closing out of the browser. In this situation the orders are left in created state and no further progress is made. In order to handle this these orders, a ClearHungOrders job has been created. This can be configured to change the order status to “Failed” and it runs on the configured timeframe (for example, at 30-minutes intervals).

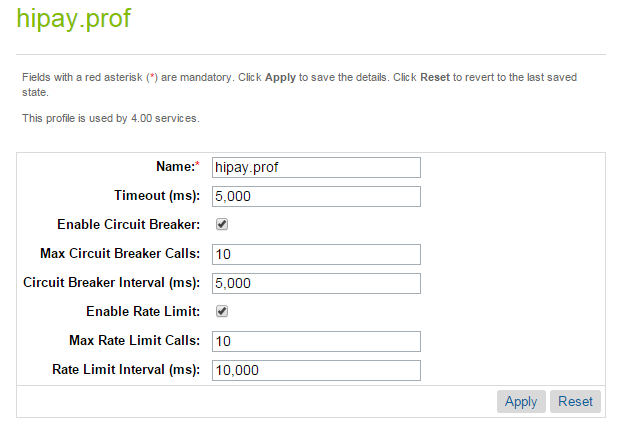


## Availability

HiPay integration cartridge uses the below services.



All services are using the below Profile configuration.



The circuit breaker suspends platform calls to a web service if a certain number of calls fail within a specified time interval. It can be easily changed for different merchants. A different/ a separate Profile can be added for each service.

## Support

In the event of problems with the integration, missing features, etc. please contact the HiPay’s Business IT Services at <https://support.hipay.com> or your HiPay’s account manager.

# User Guide

## Roles, Responsibilities

The store administrator should check the correct configuration of the HiPay Merchant account in the HiPay back office and should check the receipt of HiPay notification messages on a regular basis.

Admins should regularly, check all the payments received to ensure they contain the expected data (currencyCode, amount, shopperEmail, etc).

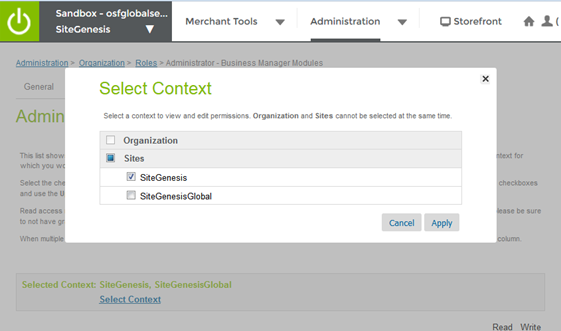
## Business Manager

### HiPay business module

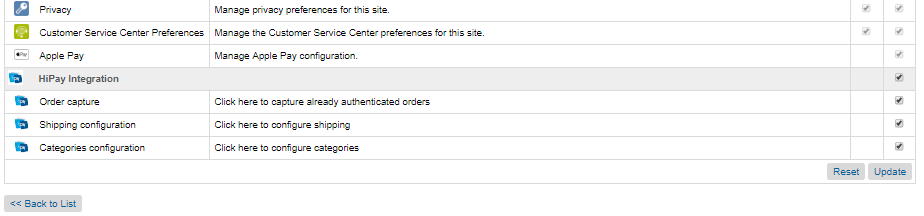
HiPay Integration business modules have been created. They are used to capture amounts for already authorized orders and configure shipping methods and product categories for Oney payment method.

**Configuration**

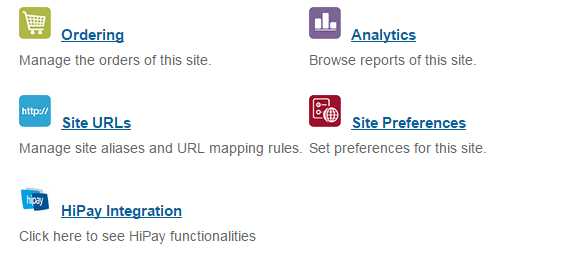
1. Go to: Administration > Organization > Roles > Administrator - Business Manager Modules
2. Select your site context and press **Apply.**

****

1. Check HiPay Integration module and press Update.

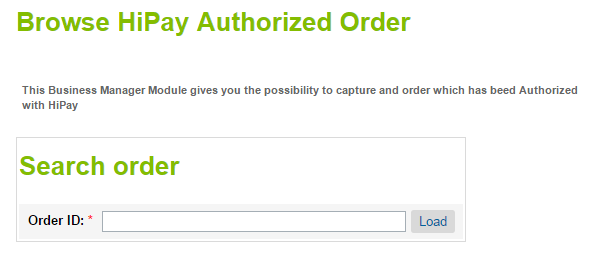
****

1. Go to the Business Manager> Merchant Tools > HiPay module should be displayed.

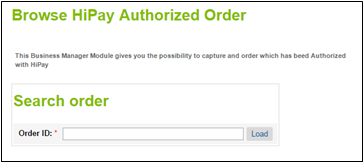


Usage

1. Go to: **HiPay Integration > Order capture**
2. Enter an order ID and press **Load**

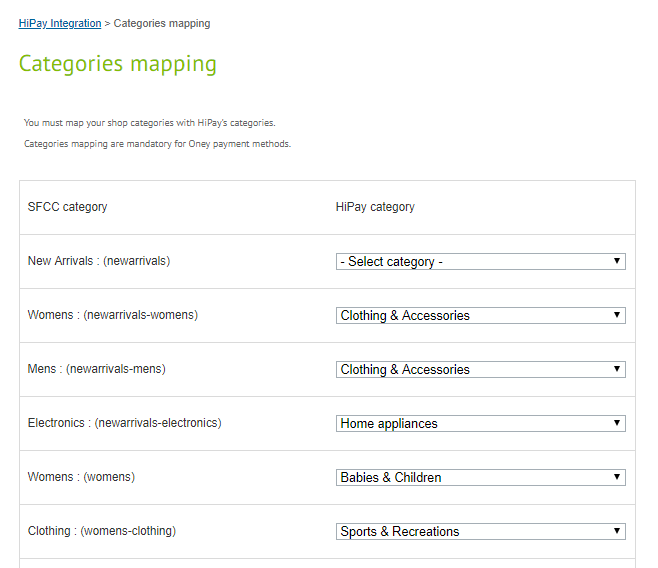


1. Partial or full capture can be requested. If the requested amount is captured, this will send a HiPay notification to Salesforce Commerce Cloud and the **Capture amount** will be updated.



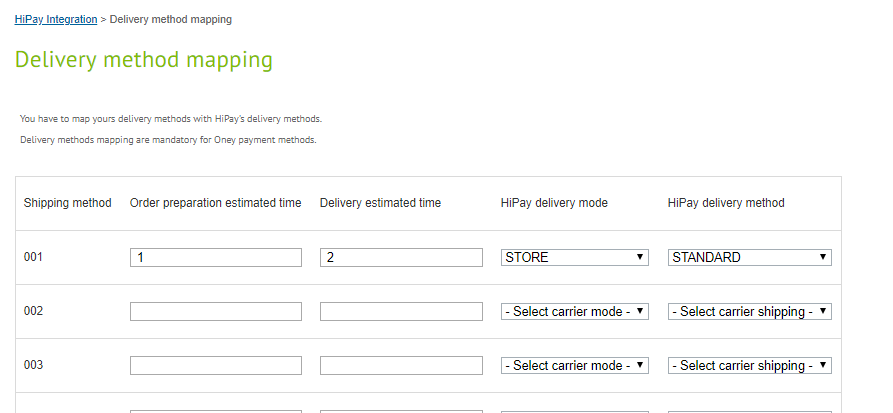
1. For the Oney payment method configuration go to: **HiPay Integration > Categories Configuration.**

This page allows to map SFCC product categories with HiPay’s categories. Once categories mapped, save configuration. In case if a SFCC category is not mapped – the Oney payment method will not be rendered if basket contains at least one product with not configured category.



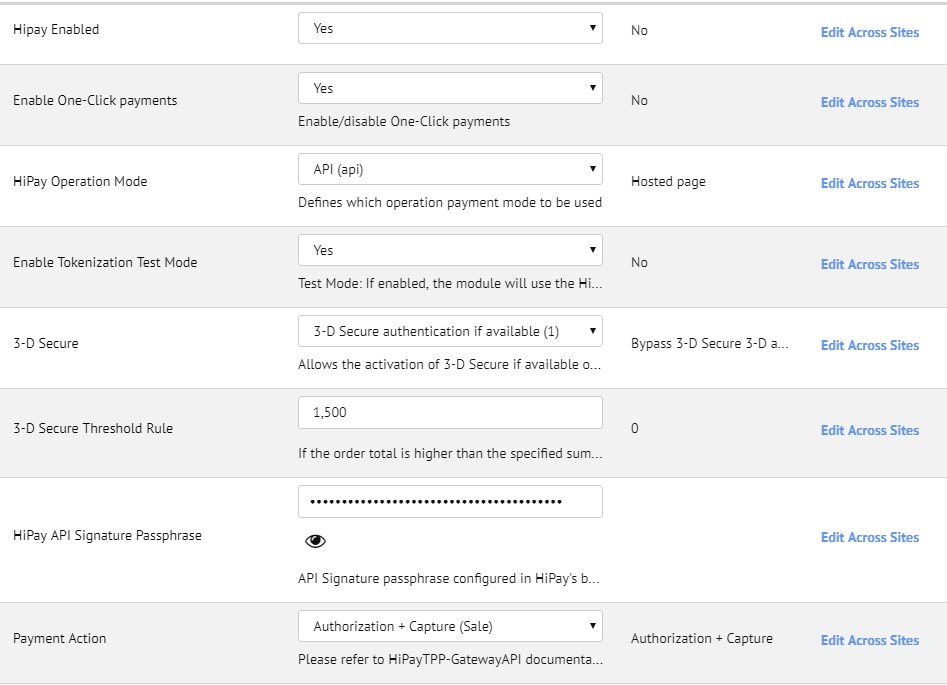
1. For the Oney payment method configuration go to: **HiPay Integration > Shipping Configuration.**

This page allows to configure SFCC shipping methods with HiPay’s delivery methods. The ‘Order preparation estimated time’ and ‘Delivery estimated time’ fileds use Integer, meaning 1 is one day, 2 is two days etc. Once shipping methods, save configuration. In case if a SFCC shipping method is not configured – the Oney payment method will not be rendered if the not-configured shipping method selected on the shipping page.



### HiPay Site Preferences

Go to: Merchant Tools > Site Preferences > Custom Site Preferences > HiPay Settings



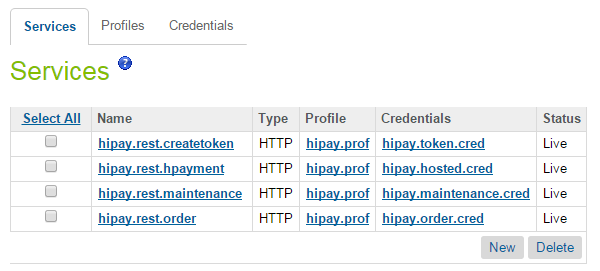


Here you will find the HiPay configuration options.

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Possible values | Default Value | Description |
| Hipay Enabled | true / false | false | Enable/disable HiPay functionality |
| Enable One-Click payments | true / false | false | Enable/disable One-Click payments |
| HiPay Operation Mode | hosted (Hosted page)  iframe (IFrame)  api (API) | hosted (Hosted page) | Defines which operation payment mode to be used |
| Enable Tokenization Test Mode | true / false | false | Test Mode: If enabled, the module will use the Hipay TPP test platform |
| 3D Secure | 0 *(Bypass 3-D Secure 3-D authentication)*  1 (3-D Secure authentication if available)  2 (3-D Secure authentication mandatory) | 0 *(Bypass 3-D Secure 3-D authentication)* | Allows the activation of 3D secure if available on used card |
| 3D Secure Threshold Rule | Integer | 0 | If the Order total is higher than the specified sum, then 3D Secure will be forced. If 0 is specified, the rule is disabled. |
| HiPay API Signature Passphrase | String |  | API Signature passphrase configured in HiPay backoffice. Use to verify the requests made to SFCC. |
| Payment Action | Sale *(Authorization + Capture)*  Authorization (Authorization Only) |  | Please refer to HiPayTPP-GatewayAPI documentation, chapter 3.1 Request a New Order – operation |
| iFrame Height | Integer | 750 | If iFrame operating mode is chosen, you can select your iFrame height to fit with your CSS |
| iFrame Width | Integer | 950 | If iFrame operating mode is chosen, you can select your iFrame width to fit with your CSS |
| HiPay CSS content | Text |  | CSS applied to either Hosted or iFrame page |
| Display card selector | true / false | false | Enable/disable the payment methods selector on iFrame and Hosted page |
| Hung Order Cleanup Time | Integer | 30 | The time in minutes after which all Orders hung in status CREATED that are left by the payment processing are cleaned. |
| List of shipping methods | String |  | A comma-separated list of the shipping methods that should be in use for the Oney payment method. |

### Services

Go to: Administration > Operations > Services



There are four services implemented. Each one corresponds to a different HiPay service, as   
follows:

* **hipay.rest.createtoken** – handles the credit card tokenization
* **hipay.rest.hpayment –** handles the hosted payment calls
* **hipay.rest.maintenance** – handles the order updates after being placed
* **hipay.rest.order** – handles placing order

#### HiPay Multi-account

In order to configure sandbox for multiple HiPay accounts you need duplicate previous services with new names:

* **hipay.rest.createtoken.{siteID}**
* **hipay.rest.hpayment.{siteID}**
* **hipay.rest.maintenance.{siteID}**
* **hipay.rest.order.{siteID}**

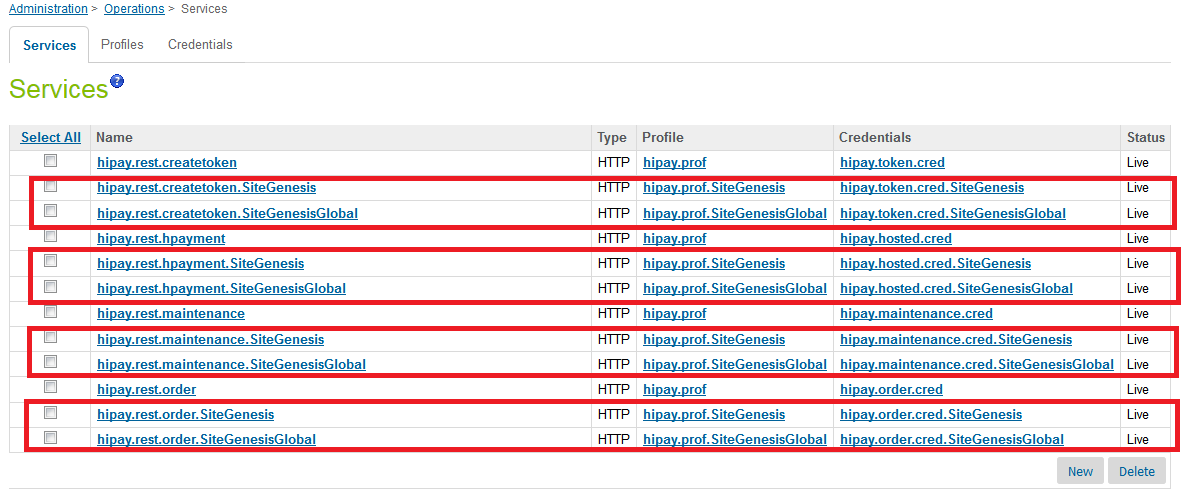
Where siteID is the ID of the site for which you wish to add a new HiPay account. Services without a siteID in the service name should be deleted as, they will not become default services, in case a specific service cannot be found in BM. You can use the **hipay.prof** profile for those services or you can create a new one in order to make customizations in future for specific site. Also, you will need duplicate default credentials:

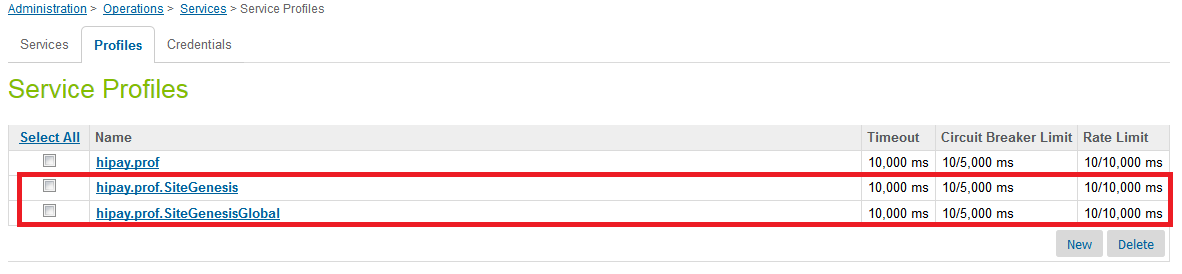
* **hipay.hosted.cred**
* **hipay.maintence.cred**
* **hipay.order.cred**
* **hipay.token.cred**

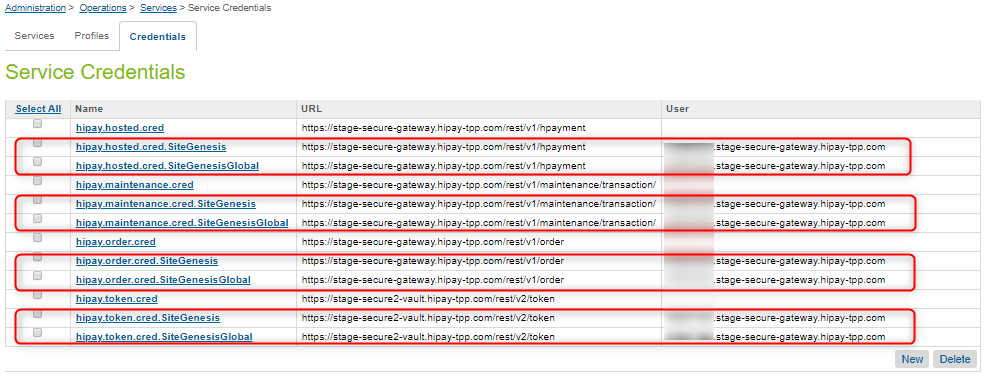
Update them with a new name, HiPay account username and credentials. This is an example mapping for each service which credential should be assigned:

* **hipay.rest.hpayment.SiteGenesis** *service* **- hipay.hosted.cred.SiteGenesis** *credential*
* **hipay.rest.createtoken.SiteGenesis** *service* **- hipay.token.cred.SiteGenesis** *credential*
* **hipay.rest.order.SiteGenesis** *service* **- hipay.order.cred.SiteGenesis** *credential*
* **hipay.rest.maintenance.SiteGenesis** *service* **- hipay.maintenance.cred.SiteGenesis** *credential*

Below you may find an example with Services configured for different sites and two HiPay accounts (please refer to the marked lines).



****

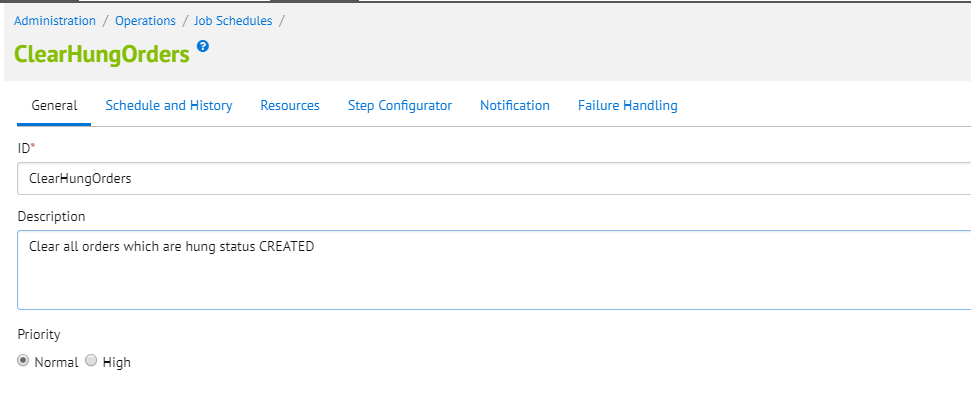
****

### Schedules

Two new job processes are implemented for HiPay:

A **ClearHungOrder** job process is implemented, which is used to clear all orders that have been started with a HiPay hosted solution but have not been completed within the configured time period.

Go to: **Administration > Operations > Jobs (previously Job Schedules) > ClearHungOrders**

****

Under the “Job Steps” tab (previously “Step Configurator”), change the scope in order to assign the job to your site. Under the “Schedule and History” tab, you can also check that the job is active and review the “Run Time” settings for its automatic execution.

A **HiPayClear** job process is also implemented, which is used to clear custom objects that are useless after a time period.

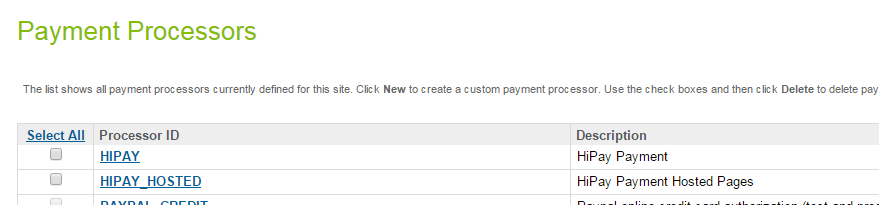
Again, Go to: **Administration > Operations > Jobs (previously Job Schedules) > HiPayClear.**

Under the “Job Steps” tab (previously “Step Configurator”), change the scope in order to assign the job to your site. Under the “Schedule and History” tab, you can also check that the job is active and review the “Run Time” settings for its automatic execution.

### Payment Processors

There are two payment processors added for HiPay – **HIPAY\_HOSTED** and **HIPAY.** The hosted is used only for the hosted and iframe solution. HIPAY processor is used for the HiPay API integration. It is used for all different types of payments like credit cards, IDEAL, ING HomePay, Giropay, KLARNA, etc.

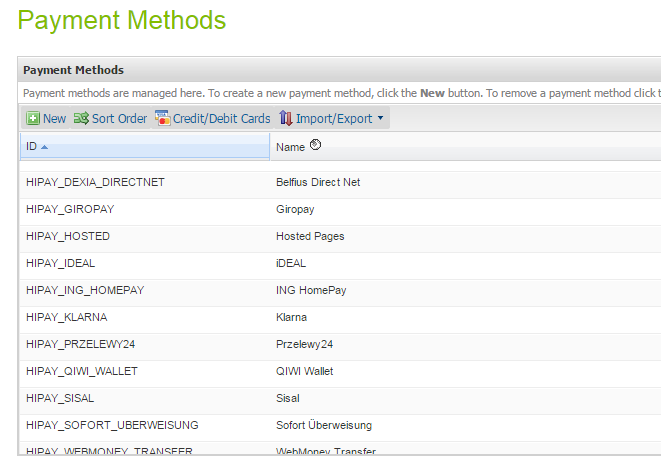
Go to: **Merchant Tools > Ordering > Payment Processors**

****

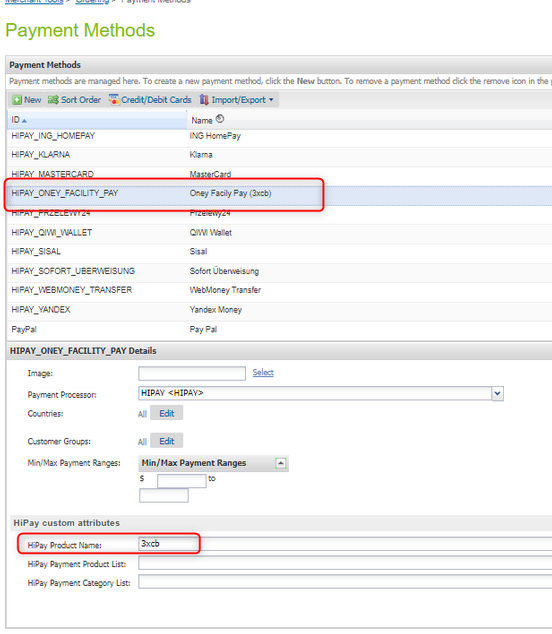
### Payment Methods

All possible HiPay payment methods have been added to the BM. In order to function in Salesforce Commerce Cloud, they must also be activated in the HiPay Fullservice application.

Go to: **Merchant Tools > Ordering > Payment Methods**

****

In addition, every payment method has a custom filed ‘HiPay Product Name’ which is sent with every order request to HiPay service.



For example, there are 8 payment methods available for Oney Facility Pay. 4 of them (with HOSTED prefix in payment method name) are in use in case if a custom site preference ‘HiPay Operation Mode’ is set to Hosted or Iframe. The payment-product depends on the contract the merchant has with Oney or with any other payment method. It will be the merchant responsibility to configure the right one. Every payment method must be configured and enable on the HiPay side first.

### Logs

The following logs can be found related to the HiPay implementation:

* service-HIPAY-REST-\*service\*-blade1-3-appserver-\*date\*.log – logs HiPay services specific information
* error-blade1-3-appserver-\*date\*.log – logs Salesforce Commerce Cloud errors
* customerror-blade1-3-appserver-\*date\*.log – logs Salesforce Commerce Cloud custom errors

All of the logs can be found in: Administration > Site Development > Development Setup

Select logs and enter you BM credentials if needed.

### Notification url

Go to HiPay Fullservice -> Integration -> Notifications

In the Notification URL, the following url should be added:

http://\*domain.name\*/on/demandware.store/Sites-SiteGenesis-Site/default/HiPayNotification-Notify

This url will be used to handle all HiPay notifications.

### Custom CSS configuration

In order to use custom CSS for HiPay a custom site preference must be configured with the desired CSS.

Go to: **Merchant Tools > Site Preferences > Custom Site Preferences,** select the HiPay custom preferences group. In the **HiPay CSS content** field enter the CSS that is to be utilized on the payment page and click **Apply.** After completing this configuration, the applied CSS should be visible. The content of this preference is printed via a controller called HiPayResource-Style and its URL (e.g. http://\*domain.name\*/on/demandware.store/Sites-\*site.name\*-Site/default/HiPayResource-Style ) is sent when calling the HiPay service, which in turn adds this CSS in the header section of the iFrame or hosted page so it can be applied.

The CSS used must be plain (@include syntax is not supported) and any <style> tags must be omitted.

### Products configuration

For the PSD2 security standards, orders with dematerialized products (immediate delivery) needs to be identified during the payment. The module adds a custom attribute “**Product Dematerialized**” in a new section called “Dematerialization Attributes”, dedicated to this requirement. All your dematerialized products must have this custom attribute value set to “Yes”.

To find this new attribute, go to: **Merchant Tools > Products and Catalogs > Products**. Open the product of your choice and scroll to the bottom of the page.



The attribute has 3 possible values:

* “**-None-**”: The default value, when it is not defined yet. Acts as “No” value.
* “**No**”: The product is not dematerialized. It needs to be physically delivered to the customer, generally in days.
* “**Yes**”: The product is dematerialized. It could be electronically delivered to the customer, generally in minutes.

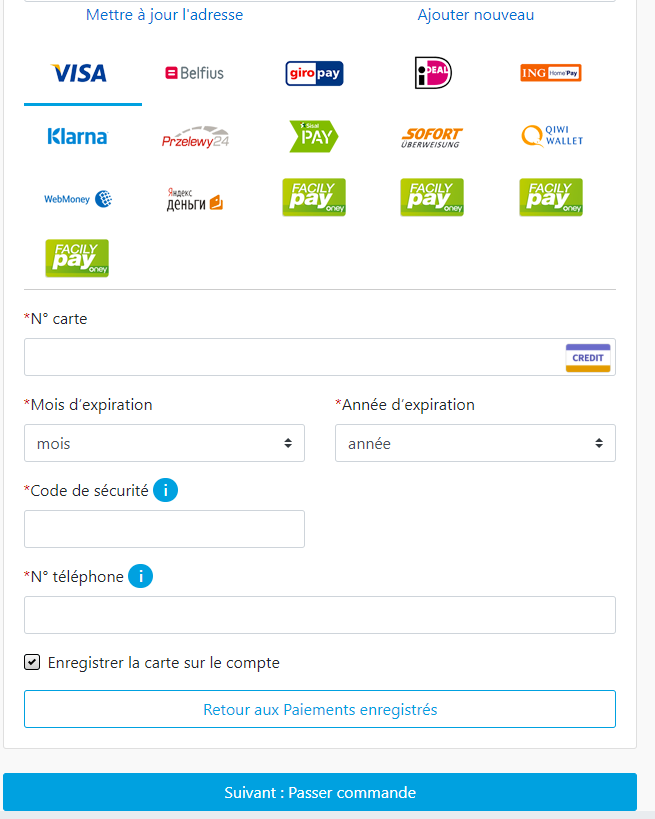
In order to set the value, Lock your product for edition. Then, at the right of “**Product Dematerialized**”, click the select list and choose the correct value. Then, click “Apply” button at the bottom of the screen. Do not forget to Unlock your product for other BM users.

## Storefront Functionality

Storefront functionality is fully described in section Use Cases.

### Merchant Payment API - Credit Cards

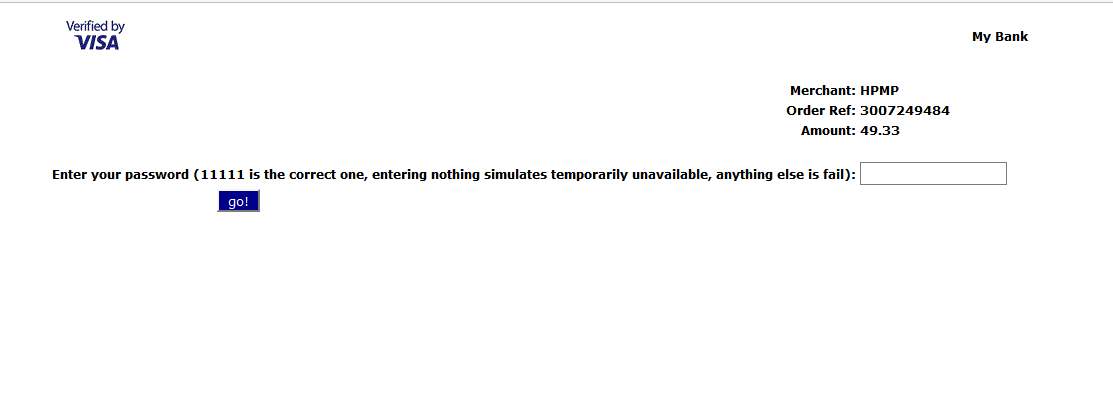
When using this method, the storefront functionality will be the same as the standard one for credit cards. The card is authorized inline during checkout directly on the merchant's website.



### Payment API - Merchant Link Credit Cards (with 3DSecure)

When using this method, the customer still selects what type of card they wish to use on the Billing page.

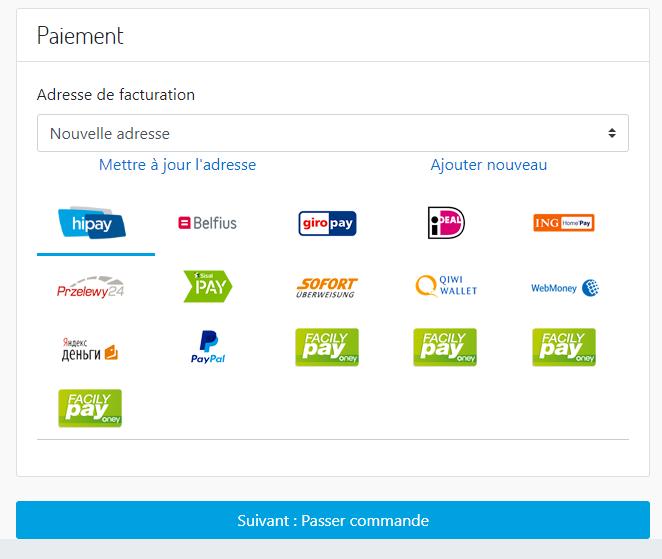
After Submitting the order from the summary page, if card is applicable by the customer’s bank to 3D Secure, they will be directed to the 3DSecure page.



When payment is complete, the customer is redirected to the confirmation page as per the normal flow.

### Hosted Merchant Link

When using these methods, if the currency is applicable with one of these methods, the bank payment option will be shown (each one of them work only in specific currencies).

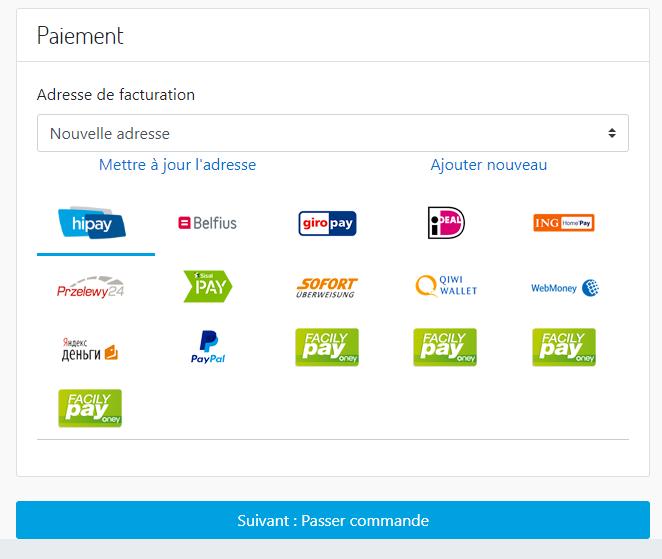


After Submitting the order from the summary page, the customer is redirected to the 3rd party website.

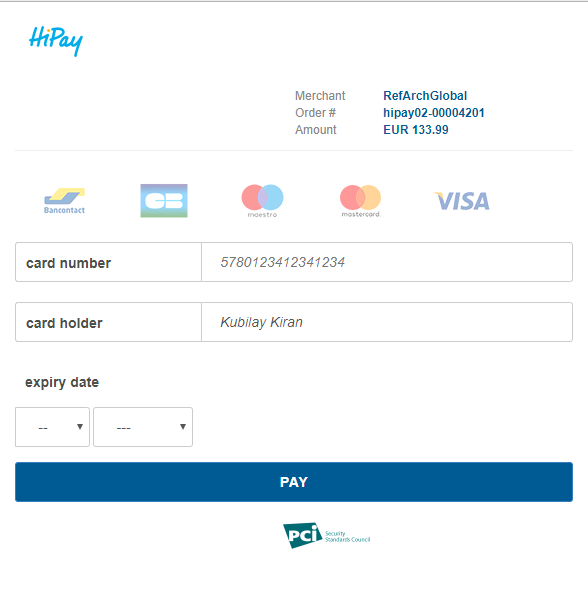
After completing payment with the 3rd party, the customer is redirected back to the storefront. If the payment is successful or unknown, the customer is taken to the confirmation page. If the payment was unsuccessful, the customer is given an error message on the summary page and asked to select a different payment method.

### Hosted Merchant Link Credit Cards

On the billing page, the customer selects Hosted Credit Cards.



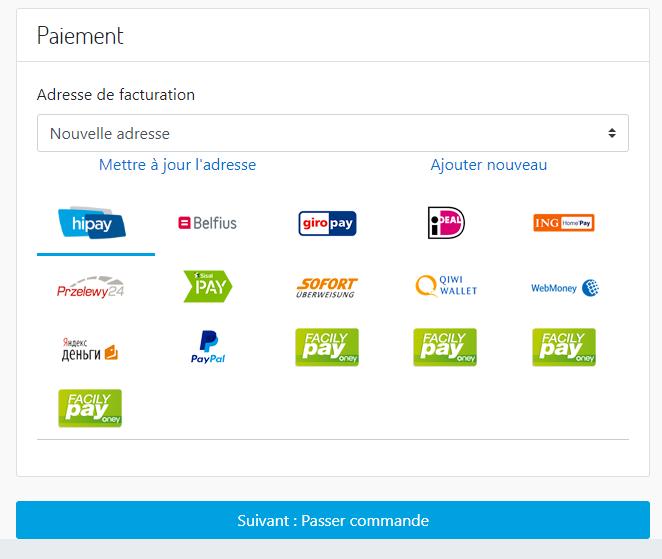
After the customer submits the order is redirected to a Payment page hosted and provided by HiPay, where the credit card form appears where the end-customer can enter the payment details:



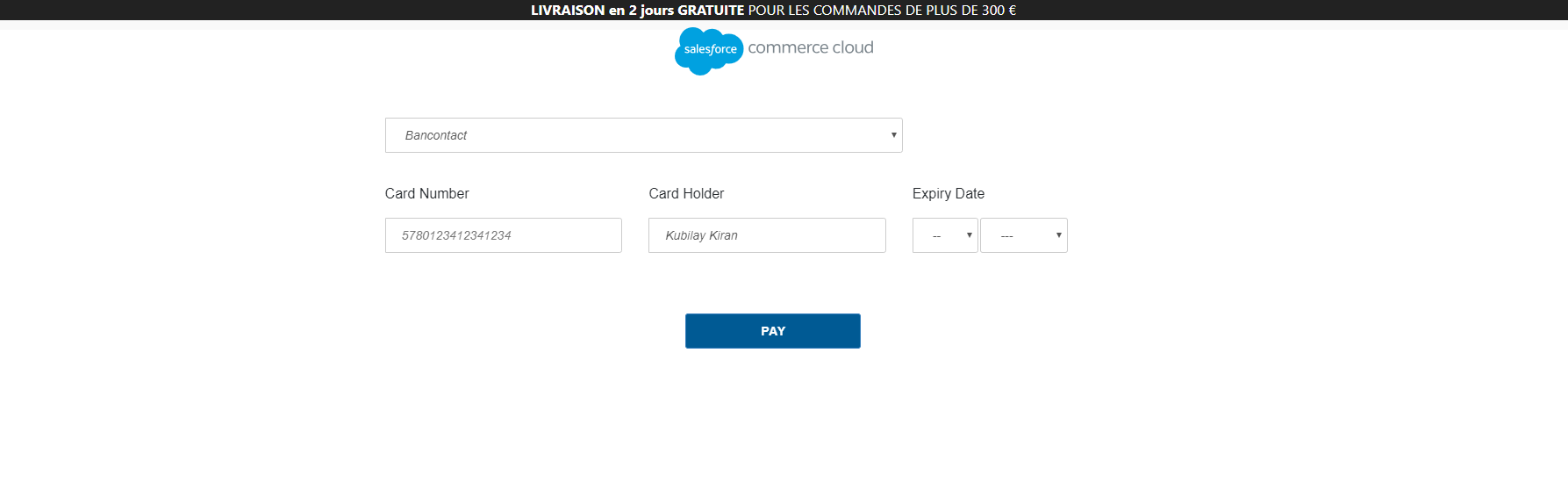
After completing payment with the HiPay, the customer is redirected back to the storefront. If the payment is successful or unknown, the customer is taken to the confirmation page.

### Hosted Merchant iFrame

On the billing page, the customer selects Hosted Credit Cards or other payment method.



After Submitting the order from the summary page, the customer is then taken to a Payment page, where the HiPay credit card form appears inside an iFrame:



When payment is complete, the customer is directed to the confirmation page as usual. If the payment is successful or unknown, the customer is taken to the confirmation page. If the payment was unsuccessful, the customer is given an error message on the summary page and asked to select a different payment method.

# Known Issues

The cartridge and its documentation will be in English only.

The HiPay integration cartridge is compatible with the latest SFCC API version (currently 19.10) and SFRA version (currently 4.4.1). It might require updates and reviews for future versions and releases of the Storefront Reference Architecture.

# Release History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 1.0.0 | 07/01/2019 | Initial release. |
| 20.1.1 | 05/13/2020 | Add compliance with PSD2 and Strong Customer Authentication – 3D Secure 2.  Add Multibanco and MBway payment methods |