

# Nexi XPay Integration Guide For salesforce commerce cloud

## Table of Contents

<b>1</b>	Prerequisites .....	3
<b>2</b>	Install the cartridge.....	3
<b>3</b>	Setup the cartridge.....	3
3.1	Set up the cartridge path.....	3
3.2	Set up the Business Manager path.....	3
<b>4</b>	Setup Roles and Permissions.....	4
4.1	Create a role.....	4
4.2	Assign users.....	4
4.3	Grant permission to back office users.....	4
4.4	Grant permission to admin users .....	5
<b>5</b>	Import metadata.....	5
<b>6</b>	Setup payment methods .....	6
6.1	Create payment processor.....	6
6.2	Import payment method.....	6
<b>7</b>	Services.....	6
7.1	Import service metadata.....	6
7.2	Setup service settings .....	6
<b>8</b>	Log Settings .....	7
<b>9</b>	Setup site preferences .....	7
<b>10</b>	Update Merchant Profile.....	8
<b>11</b>	SiteGenesis code changes .....	9
11.1	template changes.....	9
11.2	html elements.....	11
11.3	Automatic form submission.....	11
11.4	javascript events .....	12
<b>12</b>	Customize stylings.....	12
12.1	Data model changes .....	13
12.2	Template changes.....	14
12.3	Setup business manager extension .....	15
12.4	Grant permission to back office users.....	17
<b>13</b>	SiteGenesis code changes .....	18
13.1	Remove default payment methods.....	18
13.2	XPay Build integration .....	19
13.2.1	Template changes .....	19
13.2.2	Html elements .....	21
13.2.3	Automatic form submission .....	21
13.2.4	Javascript events .....	22
13.3	Easy Payment integration.....	23
13.3.1	Template changes .....	23
13.4	OneClick Payment integration.....	24
13.4.1	Template changes .....	24
<b>14</b>	Customize stylings.....	27

## 1 Prerequisites

- ☐ Salesforce B2C Commerce v19.10
- ☐ An implementation of a Site Genesis site

## 2 Install the cartridge

Nexi provides a LINK cartridge to integrate with Salesforce Commerce Cloud (SFCC). This cartridge enables the Nexi XPay payment service for store-fronts using the SiteGenesis JS-Controllers architecture (SGJC).

1. Open the provided NexiXPay.zip package
2. Extract the LINK-NexiXPay folder which contains two folders with the following cartridges:
  - int\_nexipay
  - bm\_nexipay
3. Upload int\_nexipay and bm\_nexipay cartridges using Commerce Cloud UX-studio.

## 3 Setup the cartridge

### 3.1 Set up the cartridge path

1. Select *Administration > Sites > Manage Sites*
2. Open the site where you like to install the cartridge and select Settings tab
3. Add int\_nexipay:cc\_customization followed by a colon to the Cartridges textbox.
4. Click *Apply*.

[Administration](#) > [Sites](#) > [Manage Sites](#) > NEXI SiteGenesis - Settings

General **Settings** Cache Site Status Page Meta Tag Rules

### NEXI SiteGenesis - Settings

Click Apply to save the details. Click Reset to revert to the last saved state.

Instance Type:

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 hostname values set in this section will be u

HTTP Hostname:

HTTPS Hostname:

Instance Type: All

Cartridges:

Effective Cartridge Path:

[<< Back to List](#)

Picture 3.1

### 3.2 Set up the Business Manager path

1. Select *Administration > Sites > Manage Sites*
2. Click on *Manage*
3. In the *Business Manager Site* section, click on *Manage the Business Manager site*.
4. Add bm\_nexipay followed by a colon to the cartridges textbox.
5. Click *Apply*.

[Administration](#) > [Sites](#) > [Manage Sites](#) > Business Manager - Settings

Settings **Cache** Hostnames

### Business Manager - Settings

Click Apply to save the details. Click Reset to revert to the last saved state.

Instance Type:

Deprecated. Up to two instance specific hostname aliases for Business Manager can be configured here.

HTTP Hostname:

HTTPS Hostname:

Instance Type: All

Cartridges:

Effective Cartridge Path:

[<< Back to List](#)

Picture 3.2

## 4 Setup Roles and Permissions

The Nexi XPay cartridge is provided with various Business Manager modules to implement back-office functionalities. In order to grant permissions to a restricted set of users, you need to create a proper role and assign it to your back-office administrators.

### 4.1 Create a role

1. Select *Administration > Organization > Roles*
2. Click New to create a new role
3. For ID, enter a role name for your back-office users.
4. Click *Apply*.

Administration > Organization > Roles > New Access Role

General Users Business Manager Modules Functional Permissions WebDAV Permissions Locale Permissions Price Adjustment Limits Customer Service Center Permissions

### New Role

This page allows you to create a new access role. Please type in an ID that uniquely identifies the access role. An error is displayed if no ID is provided or if an access role with such an ID already exists. Click Apply to create the role.

ID:\* XPay.Merchant

Description:

<< Back to List

Figure 4.1

### 4.2 Assign users

1. Select *Administration > Organization > Roles*
2. Click on the role just created in the previous step.
3. Select the Users tab and click Assign.
4. Check your back-office users.
5. Click *Assign*.

Administration > Organization > Roles > XPay.Merchant - Users

General Users Business Manager Modules Functional Permissions WebDAV Permissions Locale Permissions Price Adjustment Limits Customer Service Center Permissions

### XPay.Merchant - Assigned Users

The list shows all users that are assigned to this role. Click Assign to add a user. Use the checkboxes and the Unassign button to remove users from this role.

Select All	Last Name	First Name	Login
<input checked="" type="checkbox"/>	Admin2		admin2
<input type="checkbox"/>	Admin	Admin	admin

Assign Unassign

<< Back to List

Figure 4.2

### 4.3 Grant permission to back office users

1. Select *Administration > Organization > Roles*
2. Click on the role just created in the previous step.
3. Select the **Business Manager Modules** tab and click Assign.
4. Click Select Context, select your site and click Apply.

Administration > Organization > Roles > XPay.Merchant - Users

General Users Business Manager Modules Functional Permissions WebDAV Permissions Locale Permissions Price Adjustment Limits Customer Service Center Permissions

### Business Manager Modules

Selected Context: None  
[Select Context](#)

Business Manager Module	Module Description
<a href="#">Select Context</a>	

Reset Update

<< Back to List

Figure 4.3

5. Check the following Nexi XPay modules:

under **Ordering** section:

- Nexi XPay Reimbursement
- Nexi XPay Capture
- Nexi XPay Reports
- NEXI PayMail Reports

under **Ordering** section:

- Reimbursement
- Capture
- Reports
- PayMail Reports

under Site Preferences section:

- Nexi XPay Merchant Profile

 Nexi XPay Reimbursement	Click here to start a reimbursement	<input checked="" type="checkbox"/>
 Nexi XPay Capture	Click here to start a capture (only available if deferred accounting is enabled)	<input checked="" type="checkbox"/>
 Nexi XPay Reports	Click here to search for backoffice reports	<input checked="" type="checkbox"/>
 Nexi PayMail Reports	Click here to search for backoffice reports	<input checked="" type="checkbox"/>

Figure 4.4







 Nexi XPay		<input checked="" type="checkbox"/>
 Reimbursement	Click here to start a reimbursement	<input checked="" type="checkbox"/>
 Capture	Click here to start a capture (only available if deferred accounting is enabled)	<input checked="" type="checkbox"/>
 Reports	Click here to search for backoffice reports	<input checked="" type="checkbox"/>
 PayMail Reports	Click here to search for backoffice payment reports	<input checked="" type="checkbox"/>
 Recurring Pay Tester	Click here to test recurring payments	<input checked="" type="checkbox"/>


Figure 4.5

6. Click *Update*.

#### 4.4 Grant permission to admin users

The Nexi XPay modules are not granted by default to the administrators of the Commerce Cloud instance. To grant access to the administrators, you need to manually edit the administrator role and apply the same changes applied to back-office users. To do that just repeat the previous steps, selecting the Administrator role instead of your back-office role.

#### Roles

The page shows all roles in this organization.  
Roles marked with  have permission to view or manage users or access roles or both. They are therefore security-sensitive roles. Please be careful when changing these roles so as not to unintentionally give access privileges to certain users.


Select All	ID	Description
<input type="checkbox"/>	 <a href="#">Administrator</a>	The administrator has the rights to perform tasks related to the overall administration of the merchant organization and its users and roles. This access role is not site-specific and will grant the user access to the entire organization.

Figure 4.6

## 5 Setting Permissions for Customer Service Center

To allow your back-office users to create new orders using Customer Service Center, they must have the necessary permissions.

To grant proper permissions to Customer Service users follow these steps:

1. Select *Administration > Organization > Roles*
2. Click on the role which your back-office users are assigned.
3. Select the **Business Manager Modules** tab and click Assign.
4. Click Select Context, select your site and click *Apply*.

Administration > Organization > Roles > Merchant - Business Manager Modules

General Users **Business Manager Modules** Functional Permissions WebDAV Permissions Locale Permissions Price Adjustment Limits Customer Service Center Permissions

### Merchant - Business Manager Modules

This list shows all Business Manager modules available in the system for which permissions can be granted. Click **Select Context** to select the context for which you would like to modify permissions. Available context options are the organization, one site, or multiple sites.  
Select the checkboxes and use the **Update** button at the bottom of the page to grant permissions to certain Business Manager modules. Deselect the checkboxes and use the **Update** button to revoke permissions to specific Business Manager modules.  
Read access may be granted to an increasing number of Business Manager modules. Write access includes read access. When removing read access, make sure you haven't also granted write access for the respective feature.  
When multiple contexts are selected with different permissions for a feature or module, detailed information can be viewed by clicking in the "Details" column.

Selected Context: None <a href="#">Select Context</a>	
Business Manager Module	Module Description
<a href="#">Select Context</a>	
<input type="button" value="Reset"/> <input type="button" value="Update"/>	

Figure 5.1

5. Enable the following permissions:
  - Orders
  - Customer Service Center
6. Click the *Functional Permissions* tab.
7. Select your site.
8. Enable the following permissions:
  - Adjust\_Item\_Price
  - Adjust\_Shipping\_Price
  - Adjust\_Order\_Price
  - Delete\_Order\_Note
  - Create\_Order\_On\_Behalf\_Of
  - Search\_Orders
9. Click the *Locale Permissions* tab.
10. Check locales to enable them, uncheck to disable them for this role.
11. Click *Apply*.

## 6 Import metadata

1. Select *Administration > Site Development > Import & Export*
2. In the *Import & Export Files* section, click *Upload* and select the provided file:  
Custom\_Attributes.xml
3. In *Meta Data* section, click *Import* and select the file uploaded in the previous step

## 7 Setup payment methods

### 7.1 Create payment processor

1. Select *Merchant Tools > Ordering > Payment Processors*
2. Click *New* to create a new payment processor with name: NEXI\_XPAY
3. Click *Apply*.

NEXI\_XPAY

Click *Back to List* to display the list again.

ID:*	NEXI_XPAY
Description:	<div></div>
<input type="button" value="Apply"/> <input type="button" value="Reset"/> <input type="button" value="Delete"/>	

Picture 7.1

### 7.2 Import payment method

1. Select *Merchant Tools > Ordering > Import & Export*
2. In the *Import & Export Files* section, click *Upload* and select the provided file:  
payment-methods.xml
3. In *Payment Methods* section, click *Import* and select the file uploaded in the previous step.
4. Click *Next* to complete step 1 and step 2 of the import process.
5. On step 3, select *MERGE* for import mode.
6. Click *Import*.

## 8 Web Services

### 8.1 Import service metadata

1. Select *Administration > Operations > Import & Export*
2. In the *Import & Export Files* section, click *Upload* and select the provided file:  
NexiXPay\_Service.xml
3. In *Services* section, click *Import* and select the file uploaded in the previous step.
4. Click *Next* to complete step 1 and step 2 of the import process.
5. On step 3, select *MERGE* for import mode.
6. Click *Import*.

## 8.2 Setup service settings

Once you have successfully imported the services metadata, you need to configure endpoint and credentials about the **Nexi.XPay.restservice**.

1. Select *Administration > Operations > Services*
2. Click **Nexi.XPay.cred** to edit the credentials about XPay services.
3. For **URL**, enter the end point of the api services provided by Nexi.
4. For **User** and **Password**, enter the credentials assigned to the merchant by Nexi.
5. Click *Apply*.

### Nexi.XPay.cred

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:*	Nexi.XPay.cred
URL:	
User:	
Password:	

[Apply](#)

Picture 7.1

If needed, you can also edit the **Nexi.XPay.profile** to adjust the timeout time and rate limit parameters. The default connection timeout is set to 30ms.

### Nexi.XPay.profile

Fields with a red asterisk (\*) are mandatory. Click Apply to save the details. Click Reset to revert to the last saved state.  
This profile is used by 1 service.

Name:*	Nexi.XPay.profile
Connection Timeout (ms):	30,000
Enable Circuit Breaker:	<input checked="" type="checkbox"/>
Max Circuit Breaker Calls:	100
Circuit Breaker Interval (ms):	60,000
Enable Rate Limit:	<input type="checkbox"/>
Max Rate Limit Calls:	0
Rate Limit Interval (ms):	0

[Apply](#)

[<< Back to List](#)

Picture 7.2

## 9 Log Settings

1. Select *Administration > Operations > Custom Log Settings*
2. Create the following Custom Log Filters:

nexi.xpay.bm  
nexi.xpay.int

3. Set a debug level (WARN for Production and DEBUG for Development/Testing)

## 10 Setup site preferences

The functionalities provided by XPay integration cartridge can be customized by several custom site preferences.

Preference name	Values	Description
XPAY_DEF_LANGUAGE	<ul style="list-style-type: none"> <li>• ITA</li> <li>• ENG</li> <li>• SPA</li> <li>• FRA</li> <li>• GER</li> <li>• JPN</li> <li>• CHI</li> <li>• ARA</li> <li>• RUS</li> <li>• POR</li> </ul>	Language code as defined in this language Id table

XPAY_CURRENCY	EUR	Currency code (978 is Euro currency)
XPAY_ENVIRONMENT	<ul style="list-style-type: none"> <li>for dev environment: INTEG</li> <li>for production environment: PROD</li> </ul>	Implementation environment
XPAY_TCONTAB	<ul style="list-style-type: none"> <li>C</li> <li>D</li> </ul>	<p>The field identifies the collection method that the merchant wants to apply to the single transaction, if valued with:</p> <ul style="list-style-type: none"> <li>C (immediate) the transaction if authorized is also collected without further intervention by the operator and without considering the default profile set on the terminal.</li> <li>D (deferred) or the field is not entered the transaction if authorized is managed according to what is defined by the terminal profile.</li> </ul> <p>See TCONTAB parameter at Nexi documentation site:  <a href="https://ecommerce.nexi.it/specifiche-tecniche/build/pagamento.html">https://ecommerce.nexi.it/specifiche-tecniche/build/pagamento.html</a></p>
XPAY_API_KEY		Merchant profile identification code (fixed value communicated by Nexi during the activation phase).
XPAY_BUILD_JS_URL_PREFIX	<ul style="list-style-type: none"> <li>for dev environment: <a href="https://int-ecommerce.nexi.it/ecom/XPayBuild/js?alias=">https://int-ecommerce.nexi.it/ecom/XPayBuild/js?alias=</a></li> <li>for production environment: <a href="https://ecommerce.nexi.it/ecom/XPayBuild/js?alias=">https://ecommerce.nexi.it/ecom/XPayBuild/js?alias=</a></li> </ul>	Url base address of Nexi services
XPAY_PAYMODE	<ul style="list-style-type: none"> <li>XPayBuild</li> <li>EasyPayment</li> <li>OneClick</li> </ul>	Current payment method selected.
XPAY_TERM_WEB_ALIAS	Ex. ALIAS_WEB_00018210	Alias Provided by Nexi - Information for first terminal
XPAY_TERM_WEB_SECRETKEY	Ex. LOCH7SWIRMO1A5L3VI13W...	Key for Mac Provided by Nexi - Information for first terminal
XPAY_TERM_WEB_GROUP	Ex. GRP_33912	Group Provided by Nexi - Information for first terminal
XPAY_TERM_RICO_ALIAS	Ex. ALIAS_RICO_00018212	Alias Provided by Nexi - Information for second terminal
XPAY_TERM_RICO_SECRETKEY	Ex. LOCH7SWIRMO2A5L3VI13W...	Key for Mac Provided by Nexi - Information for second terminal
XPAY_TERM_RICO_GROUP	Ex. GRP_33912	Group Provided by Nexi - Information for second terminal
XPAY_S2S_ENABLED	<ul style="list-style-type: none"> <li>Yes</li> <li>No</li> </ul>	Enable selected for server to server mode of payment confirmation

The parameters that should be provided by Nexi are highlighted in yellow.

To modify the preferences of your site follow these steps:

1. Select *Merchant Tools > Site Preferences*
2. Open the Nexi.XPay preference group.
3. Edit the preferences.
4. Click *Save*.



Name	Value
XPAY_DEF_LANGUAGE	<div>ITA</div> <div>XPay Default Language</div>
XPAY_CURRENCY	<div></div> <div>XPay Currency</div>
XPAY_ENVIRONMENT	<div>INTEG (INTEG)</div> <div>Implementation environment</div>
XPAY_TCONTAB	<div>deferred (D)</div> <div>The field identifies the collection method that the merchant wants to apply ...</div>
XPAY_API_KEY	<div>ALIAS_WEB_00019302</div> <div>Merchant profile identification code (fixed value communicated by Nexi duri...</div>
XPAY_BUILD_JS_URL_PREFIX	<div>https://int-ecommerce.nexi.it/ecom/XPayBuild/js?alias=</div>
XPAY_PAYMODE	<div>OneClick (OneClick)</div> <div>Payment Mode</div>
XPAY_TERM_WEB_ALIAS	<div>ALIAS_WEB_00019302</div> <div>Merchant profile identification code (fixed value communicated by Nexi duri...</div>
XPAY_TERM_WEB_SECRETKEY	<div>79PXRL1UXJ0CX5I7TFDFHMYRB4ZCPSEM</div> <div>Secret key for calculating MACs (fixed value communicated by Nexi during t...</div>
XPAY_TERM_WEB_GROUP	<div>GRP_36472</div> <div>Group code assigned by Nexi during activation</div>
XPAY_TERM_RICO_ALIAS	<div>ALIAS_RICO_00019303</div> <div>Merchant profile identification code (fixed value communicated by Nexi duri...</div>
XPAY_TERM_RICO_SECRETKEY	<div>79TNVIGFBSWMTWIOZ79RW28CUSZDLCW</div> <div>Secret key for calculating MACs (fixed value communicated by Nexi during t...</div>
XPAY_TERM_RICO_GROUP	<div>GRP_36472</div> <div>Group code assigned by Nexi during activation.</div>
XPAY_S2S_ENABLED	<div>Yes</div> <div>Enable/disable the server to server notification to handle the payments proc...</div>

Picture 10.1

### 11 Update Merchant Profile

Some features of the cartridge rely on data stored in the *Merchant Profile Info custom* preference. The profile is specific for any merchant and is provided by an api provided by Nexi.  
The first time you install the Nexi cartridge, there is no profile available, so you must manually updated it.  
In order to refresh the Merchant Profile just follow these steps:

1.

Select *Merchant Tools > Site Preferences > Nexi XPay Merchant Profile*
2.

Click *Update*.




Update			
Url	Code	Description	Image
https://ecommerce.nexi.it/ecom/payment/img/maestro.svg	MAESTRO	Maestro	
https://ecommerce.nexi.it/ecom/payment/img/mastercard.svg	MASTERCARD	Mastercard	
https://ecommerce.nexi.it/ecom/payment/img/visa.svg	VISA	Visa	

Figure 11.1

## 12 Recurring Payment integration

Commerce Cloud does not support recurring payment / subscription solutions built-in as standard feature.

For this reason, it's not possible to offer a ready-to-go implementation as any merchant can implement a different use case of recurring payments. In order to provide the widest range of custom integrations, the Nexi XPay cartridge expose an API to execute recurring payments compliant with Nexi XPay specifications.

Nevertheless, the XPay package is provided with a demo business manager extension to simulate recurring payments.

In the following steps we describe how to setup a possible site customization using the recurring payment api.

### 12.1 Data model changes

1. Select *Administration > Site Development > System Object Types*
2. Open *Order* and click on *Attribute Definitions* tab
3. Create a new custom attribute as follows:  
*ID:* xpay\_recurContractNum  
*Display Name:* [Nexi] Recurring Contract Number  
*Value Type:* String
4. Click *Apply*.

[Administration](#) > [Site Development](#) > [System Object Types](#) > [Order - Attribute Definitions](#) > Attribute Definition Details

### Object Type 'Order' - Attribute Definition Details

On this page you can manage details of your attribute definition. Different options are available depending on the value type of your attribute. Click **Apply** to create a new attribute definition.

Select Language: Default Apply

ID\*

☒ Display Name:

☒ Help Text:

Value Type\* String

Apply Cancel

### 12.2 Template changes

#### cc\_customization/cartridge/templates/default/checkout/summary/xpay\_recurpay.isml

The xpay\_recurpay.isml template contains the main code required to integrate the recurring payment option to Easy Payment and XPay Build payment methods into your checkout page.

Extract the following file from the provided package:

LINK-SiteGenesis-NexiXPay/cc\_customization/cartridge/templates/default/checkout/summary/xpay\_recurpay.isml  
to your site implementation folder:

<yoursite>/cartridge/templates/default/checkout/summary/xpay\_recurpay.isml

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

#### Recurring Easy Payment

To add recurring option to Easy Payment method, add the highlighted lines of code to the summary.isml file as follows:

```
<!--comment-->
  Start XPay EASY PAYMENT
</!--comment-->
<iselseif condition="${(XPayUtils.XPAY_PAYMODE === XPayUtils.PM_EASY_PAYMENT) ||
  (XPayUtils.XPAY_PAYMODE === XPayUtils.PM_ONECLICK && empty(customer.profile))}" />
  <div class="order-summary-footer">
    <div class="place-order-totals">
      <isordertotals p_lineitemctnr="${pdict.Basket}" p_showshipmentinfo="${false}" p_shipmenteditable="${false}" p_totallabel="${Resource.msg('summary
ordertotal','checkout',null)}"/>
    </div>
    <form action="${URLUtils.https('XPayOrderMng-CreateOrder')}" method="post" id="form-xpay-easypay" 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>
          <!--comment-->XPay Recurring Payment</!--comment-->
          <isinclude template="/checkout/summary/xpay_recurpay" />
        </div>
      </fieldset>
    </form>
  </div>
```

```

<isinclude template="/checkout/summary/xpay_easypay" />
</div>
</fieldset>
</form>
</div>

```

### Recurring XPay Build payment

To add recurring option to XPay Build method, add the highlighted lines of code to the summary.isml file as follows:

```

<iscomment>
    Start XPay BUILD
</iscomment>
<isif condition="${XPayUtils.XPAY_PAYMODE === XPayUtils.PM_XPAY_BUILD}">

    <div class="order-summary-footer">
        <div class="place-order-totals">
            <isordertotals p_lineitemctnr="${pdict.Basket}" p_showshipmentinfo="${false}" p_shipmenteditable="${false}" p_
totallabel="${Resource.msg('summary.ordertotal','checkout',null)}"/>
        </div>
        <form action="${URLUtils.https('COSummary-Submit')}" method="post" class="submit-order">
            <fieldset>

                <iscomment>XPay Recurring Payment</iscomment>
                <isinclude template="/checkout/summary/xpay_recurpay" />

                <isinclude template="/checkout/billing/xpay_build" />
                <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 id="xpay-build-orderbtn"
                        class="button-fancy-large" disabled="disabled"
                        type="submit" name="submitBtn"
                        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.ge-
nerateToken()}" />
            </fieldset>
        </form>
    </div>

```

### 12.3 Setup business manager extension

1. Select *Administration > Sites > Manage Sites*
2. Click on *Manage*
3. In the *Business Manager Site* section, click on *Manage the Business Manager site*.
4. Add `cc_customization` followed by a colon to the cartridges textbox.
5. Click *Apply*.

General	Settings	Cache	Site Status	Page Meta Tag Rules
<h2>NEXI SiteGenesis - Settings</h2> <p>Click <b>Apply</b> to save the details. Click <b>Reset</b> to revert to the last saved state.</p>				
<p><b>Instance Type:</b> <span>Sandbox/Development</span></p>				
<p>Deprecated. The preferred way of configuring HTTP and HTTPS hostnames is by using new features of the site aliases configuration ("S" and "H" aliases) and are intended only to support an older configuration style.</p>				
<p><b>HTTP Hostname:</b></p> <input type="text"/>		<input type="text"/>		
<p><b>HTTPS Hostname:</b></p> <input type="text"/>		<input type="text"/>		
<p><b>Instance Type:</b> All</p>				
<p><b>Cartridges:</b> <span>bm_nexipay:cc_customization:bm_custom_plugin</span></p>				

**cc\_customization/cartridge/bm\_extensions.xml**

Extract the following file from the provided package:  
 LINK-SiteGenesis-NexiXPay/cc\_customization/cartridge/bm\_extensions.xml  
 to your site implementation folder:  
 <yoursite>/cartridge/bm\_extensions.xml

This business manager extension enables the “Recurring Pay Tester” page where the back-office users can search for recurring orders and manually perform subsequent recurring payment after a first payment. This page is provided to simulate and test subsequent payments and to demonstrate the use of the api to execute a recurring payment.

Please note that this demo page is not intended to be used in production environments.

Recurring orders can be created by customers from the front-end site by checking the Recurring Order checkbox at the billing page.

**Nexi XPay**

[Storno](#)  
[Incasso](#)  
[Reportistica](#)  
[Reportistica PayMail](#)  
[Recurring Pay Tester](#)

**Nexi XPay - Recurring Pay Tester**

Search recurring orders				
Order No:	<input type="text"/>	<a href="#">Ricerca</a>		
Ordine	Contract Num.	Stato	Importo	Actions
<a href="#">00004801</a>	ablrR9vQgaYL9C7XNpB2AmMkDQ	New	124,43 EUR	<a href="#">Pay</a>
<a href="#">00004502</a>	2800187334d579281b73544d6e	New	33,60 EUR	<a href="#">Pay</a>
<a href="#">00004401</a>	64402e1bd33dd59814177a4073	Aperto	34,84 EUR	<a href="#">Pay</a>

**DEMO PAGE:** This page is for testing purposes only. Do not use it in production environments.

The Nexi XPay cartridge expose the following apis to perform recurring payments:

- XPayRecurPay.execRecurPay(order : dw.order.Order) : Object  
Helper method to execute a subsequent payment on an recurring order.
- RecurPaySvc.subseqPay(jsonRequest : Object) : Result  
Low level api to call the Nexi XPay web service “*ecomm/api/recurring/pagamentoRicorrente*” to perform a subsequent payment after a first recurring payment; see the complete specs at:  
<https://ecommerce.nexi.it/specifiche-tecniche/pagamentoricorrente/pagamentisuccessivi.html>

**Example:**

```
try {
  var XPayRecurPay = require("~/../int_nexipay/cartridge/scripts/util/XPayRecurPay");
  var order = OrderMgr.getOrder(request.httpParameterMap.orderNum.value);

  if(empty(order.custom.xpay_recurContractNum)) throw {
    errorMessage : 'Missing recurring contract number'
  }

  var result = XPayRecurPay.execRecurPay(order);

  var jsonReq = result.jsonRequest;
  var jsonRes = result.jsonResponse;
}
catch(ex) {
  logger.error('Exception: '+ex);
}
```

**12.4 Grant permission to back office users**

1. Select *Administration > Organization > Roles*
2. Click on the role which your back-office users are assigned.
3. Select the **Business Manager Modules** tab and click Assign.
4. Click Select Context, select your site and click Apply.

[Administration](#) > [Organization](#) > [Roles](#) > Merchant - Business Manager Modules

General Users **Business Manager Modules** Functional Permissions WebDAV Permissions Locale Permissions Price Adjustment Limits Customer Service Center Permissions

## Merchant - Business Manager Modules

This list shows all Business Manager modules available in the system for which permissions can be granted. Click **Select Context** to select the context for which you would like to modify permissions. Available context options are the organization, one site, or multiple sites.

Select the checkboxes and use the **Update** button at the bottom of the page to grant permissions to certain Business Manager modules. Deselect the checkboxes and use the **Update** button to revoke permissions to specific Business Manager modules.

Read access may be granted to an increasing number of Business Manager modules. Write access includes read access. When removing read access, make sure you haven't also granted write access for the respective feature.

When multiple contexts are selected with different permissions for a feature or module, detailed information can be viewed by clicking in the "Details" column.

**Selected Context: None**  
[Select Context](#)

Business Manager Module	Module Description
<a href="#">Select Context</a>	
<input type="button" value="Reset"/> <input type="button" value="Update"/>	

Figure 12.1

### 5. Under Nexi XPAY section, check "Recurring Pay Tester"

XPAY	Nexi XPay		
XPAY	Reimbursement	Click here to start a reimbursement	<input checked="" type="checkbox"/>
XPAY	Capture	Click here to start a capture (only available if deferred accounting is enabled)	<input checked="" type="checkbox"/>
XPAY	Reports	Click here to search for backoffice reports	<input checked="" type="checkbox"/>
XPAY	PayMail Reports	Click here to search for backoffice paymail reports	<input checked="" type="checkbox"/>
XPAY	Recurring Pay Tester	Click here to test recurring payments	<input checked="" type="checkbox"/>

Figure 12.2

### 6. Click *Update*.

## 13 SiteGenesis code changes

A sample of a site customization is provided in the following folder of the provided package:

LINK-SiteGenesis-NexiXPay\cc\_customization

This sample shows a working integration assuming Nexi XPay as the only payment processor available.

The Nexi XPay cartridge provide integrations with the following payment methods:

- ☐ XPay Build
- ☐ Easy Payment
- ☐ One Click

The XPAY\_PAYMODE preference must be set to the payment method that you want to integrate in your site.

Then, to integrate XPay features to your Site Genesis implementation, you need to add custom code to change the default behaviour of the standard storefront core code. Once you have set the payment mode preference, apply the custom code modifications according to the payment selected.

### 13.1 Remove default payment methods

*templates/default/checkout/billing/paymentmethods.isml*

Extract the following file from the provided package:

LINK-SiteGenesis-NexiXPay\cc\_customization\cartridge\templates/default/checkout/billing/paymentmethods.isml

to your site implementation folder:

<yoursite>\cartridge\templates/default/checkout/billing/paymentmethods.isml

Remove the default payment options included between the <fieldset> and leave only the commented code as shown here:

```
<isif condition="{${pdict.OrderTotal > 0}}">
  <fieldset>
    <iscomment>---- BEGIN CUSTOM CODE ----</iscomment>
    <iscomment>
      No payment options configured for this demo.
    </iscomment>
    <iscomment>---- END CUSTOM CODE ----</iscomment>
  </fieldset>
```

## 13.2 XPay Build integration

### 13.2.1 Template changes

#### **templates/default/checkout/billing/xpay\_build.isml**

The `xpay_build.isml` template contains the main code required to integrate the XPay Build payment method into your checkout page.

Extract the following file from the provided package:

`LINK-SiteGenesis-NexiXPay/cc_customization/cartridge/templates/default/checkout/billing/xpay_build.isml`

to your site implementation folder:

`<yoursite>/cartridge/templates/default/checkout/billing/xpay_build.isml`

#### **templates/default/checkout/summary/summary.isml**

After this standard block of code:

```
<isslot id="placeorder-slot" description="Slot next to Order Totals in the footer of the Place Order page." context="global"/>
```

modify the `summary.isml` adding a custom block of code as follows:

```
<isslot id="placeorder-slot" description="Slot next to Order Totals in the footer of the Place Order page." context="global"/>

<iscomment>---- BEGIN CUSTOM CODE ----</iscomment>

<isscript>
  var XPayUtils = require("**/cartridge/scripts/util/XPayUtils");
</isscript>

<iscomment>---- BEGIN XPay BUILD ----</iscomment>
<isif condition="${XPayUtils.XPAY_PAYMODE === XPayUtils.PM_XPAY_BUILD}">

  <div class="order-summary-footer">
    <div class="place-order-totals">
      <isordertotals p_lineitemctnr="${pdict.Basket}" p_showshipmentinfo="${false}" p_shipmenteditable="${false}" p_totallabel="${Resource.msg('summary.ordertotal','checkout',null)}"/>
    </div>
    <form action="${URLUtils.https('COSummary-Submit')}" method="post" class="submit-order">
      <fieldset>
        <iscomment>XPay Recurring Payment</iscomment>
        <include template="/checkout/summary/xpay_recurpay" />

        <include template="/checkout/billing/xpay_build" />
        <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 id="xpay-build-orderbtn"
            class="button-fancy-large" disabled="disabled"
            type="submit" name="submitBtn"
            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>
<iscomment>---- END XPay BUILD ----</iscomment>

<iscomment>---- BEGIN Default Payment of Site Genesis ----</iscomment>
<elseif>
  <div class="order-summary-footer">
    <div class="place-order-totals">
      <isordertotals p_lineitemctnr="${pdict.Basket}" p_showshipmentinfo="${false}" p_shipmenteditable="${false}" p_totallabel="${Resource.msg('summary.ordertotal','checkout',null)}"/>
    </div>
```

```

<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>
<iscomment>---- END Default Payment of Site Genesis ----</iscomment>

<iscomment>---- END CUSTOM CODE ----</iscomment>

</isdecorate>

```

### **/templates/default/billing.xml**

Extract the following file from the provided package:

*LINK-SiteGenesis-NexiXPay/cc\_customization/cartridge/templates/default/billing.xml*

to your site implementation folder:

*<yoursite>/cartridge/templates/default/billing.xml*

Edit *billing.xml* and set Nexi XPay as the default payment method:

```

<group formid="paymentMethods">
  <!-- set NEXI_XPAY as default-value -->
  <field formid="selectedPaymentMethodID" type="string" default-value="NEXI_XPAY">
    <options optionid-binding="ID" value-binding="ID" label-binding="name"/>
  </field>

```

### **13.2.2 Html elements**

As documented by Nexi in the XPay Build developer page, the XPay html elements are hosted into some specific `<div>` containers.

Figure 13.1

```

<div id="xpay-card-panel">
  <iscomment>an empty div with a unique id within the form;
    allow the SDK to create an item hosted on XPay,
    for the secure collection of card data</iscomment>
  <div id="xpay-card"></div>
</div>

```

The `xpay-card-panel` div is an outer container to host the main card html area.

The `xpay-card` div is the main container where the iframe provided by Nexi shows the form for entering the credit card data.

```
<div id="xpay-card-errors"></div>
```

The `xpay-card-errors` div is a container that is populated with error messages when some card fields are entered incorrectly.

```
<div id="xpay-buttons"></div>
```

The `xpay-buttons` div is a container for buttons related to additional payment methods provided by Nexi. The current version of the cartridge supports only XPay Build, so this container should be considered empty.

### **13.2.3 Automatic form submission**

By default, when returning from the card verification triggered by the Send Order button, if the card is successfully validated, the form is automatically submitted to finalize the payment order.

To disable the auto-submit feature, remove or comment the lines of code in `xpay-build.isml` as shown here:

```
if (response.esito && response.esito === "OK") {
    document.getElementsByName("${session.forms.billing.paymentMethods.xpay.xpayNonce.htmlName}")[0].setAttribute("value", response.xpayNonce);
    document.getElementsByName("${session.forms.billing.paymentMethods.xpay.xpayIdOperazione.htmlName}")[0].setAttribute("value", response.idOperazione);
    document.getElementsByName("${session.forms.billing.paymentMethods.xpay.xpayTimeStamp.htmlName}")[0].setAttribute("value", response.timeStamp);
    document.getElementsByName("${session.forms.billing.paymentMethods.xpay.xpayEsito.htmlName}")[0].setAttribute("value", response.esito);
    document.getElementsByName("${session.forms.billing.paymentMethods.xpay.xpayMac.htmlName}")[0].setAttribute("value", response.mac);
    document.getElementsByName("${session.forms.billing.paymentMethods.xpay.xpayCodiceTransazione.htmlName}")[0].setAttribute("value", response.codiceTransazione);
    document.getElementsByName("${session.forms.billing.paymentMethods.xpay.xpayImporto.htmlName}")[0].setAttribute("value", response.importo);
    XPay.$messages.removeClass("xpay-hidden");
    $("#xpay-card-panel").hide();

    // BEGIN - remove this to disable auto-submit
    // Auto-submit form
    $(".submit-order").submit(); // auto submit
    // END - remove this to disable auto-submit
} else {
```

### 13.2.4 Javascript events

The XPay Build integration can also be controlled by handling the following javascript events:

- **load**  
This is the standard page load event used to initialize the XPay object after the Checkout page load has completed.

Example:

```
window.addEventListener('load', function () {
    // ...
    XPay.init();
    // ...
});
```

- **XPay\_Ready**  
Event triggered when the XPay iframe loading has completed.

Example:

```
window.addEventListener("XPay_Card_Error", function (event) {
    //handle this
});
```

- **XPay\_Card\_Error**  
Event triggered when a validation error occurs when entering data in the credit card form.

Example:

```
window.addEventListener("XPay_Ready", function (event) {
    if(event.detail === 'card') {
        //handle this
    }
});
```

## 13.3 Easy Payment integration

### 13.3.1 Template changes

**`templates/default/checkout/summary/xpay_easypay.isml`**

The `xpay_easypay.isml` template contains the main code required to integrate the XPay Easy payment method into your checkout page.

Extract the following file from the provided package:

`LINK-SiteGenesis-NexiXPay/cc_customization/cartridge/templates/default/checkout/summary/xpay_easypay.isml`  
to your site implementation folder:

`<yoursite>/cartridge/templates/default/checkout/summary/xpay_easypay.isml`

**`templates/default/checkout/summary/summary.isml`**

After the XPay Build custom code, modify the `summary.isml` adding the Ease Payment custom code, as follows:



```

<isslot id="placeorder-slot" description="Slot next to Order Totals in the footer of the Place Order page." context="global"/>

<iscomment>---- BEGIN CUSTOM CODE ----</iscomment>

<isscript>
  var XPayUtils = require("**/cartridge/scripts/util/XPayUtils");
</isscript>

<iscomment>---- BEGIN XPay BUILD ----</iscomment>
<isif condition="${XPayUtils.XPAY_PAYMODE === XPayUtils.PM_XPAY_BUILD}">

  <div class="order-summary-footer">
    <div class="place-order-totals">
      <isordertotals p_lineitemctnr="${pdict.Basket}" p_showshipmentinfo="${false}" p_shipmenteditable="${false}" p_totallabel="${Resource.msg('summary.ordertotal','checkout',null)}"/>
    </div>
    <form action="${URLUtils.https('COSummary-Submit')}" method="post" class="submit-order">
      <fieldset>
        <iscomment>XPay Recurring Payment</iscomment>
        <include template="/checkout/summary/xpay_recurpay" />

        <include template="/checkout/billing/xpay_build" />
        <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 id="xpay-build-orderbtn"
            class="button-fancy-large" disabled="disabled"
            type="submit" name="submitBtn"
            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>
</iscomment>---- END XPay BUILD ----</iscomment>

<iscomment>---- BEGIN EASY PAYMENT ----</iscomment>
<iselseif condition="${(XPayUtils.XPAY_PAYMODE === XPayUtils.PM_EASY_PAYMENT) ||
  (XPayUtils.XPAY_PAYMODE === XPayUtils.PM_ONECLICK && empty(customer.profile))}" />
  <div class="order-summary-footer">
    <div class="place-order-totals">
      <isordertotals p_lineitemctnr="${pdict.Basket}" p_showshipmentinfo="${false}" p_shipmenteditable="${false}" p_totallabel="${Resource.msg('summary.ordertotal','checkout',null)}"/>
    </div>
    <form action="${URLUtils.https('XPayOrderMng-CreateOrder')}" method="post" id="form-xpay-easypay" 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>
          <iscomment>XPay Recurring Payment</iscomment>
          <include template="/checkout/summary/xpay_recurpay" />

          <include template="/checkout/summary/xpay_easypay" />
        </div>
      </fieldset>
    </form>
  </div>
</iscomment>---- END EASY PAYMENT ----</iscomment>

<iscomment>---- BEGIN Default Payment of Site Genesis ----</iscomment>
<iselse/>
  <div class="order-summary-footer">
    <div class="place-order-totals">
      <isordertotals p_lineitemctnr="${pdict.Basket}" p_showshipmentinfo="${false}" p_shipmenteditable="${false}" p_totallabel="${Resource.msg('summary.

```

```

ordertotal','checkout',null))"/>
</div>
<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>
<iscomment>---- END Default Payment of Site Genesis ----</iscomment>

<iscomment>---- END CUSTOM CODE ----</iscomment>

</isdecorate>

```

The EasyPay payment method will also be available in OneClick mode when a not registered customer tries to pay an order as a guest user.

## 13.4 OneClick Payment integration

### 13.4.1 Template changes

#### **templates/default/checkout/summary/xpay\_oneclick.isml**

The `xpay_oneclick.isml` template contains the main code required to integrate the XPay OneClick payment method into your checkout page.

Extract the following file from the provided package:

**LINK**-SiteGenesis-NexiXPay/cc\_customization/cartridge/templates/default/checkout/summary/xpay\_oneclick.isml  
to your site implementation folder:

`<yoursite>/cartridge/templates/default/checkout/summary/xpay_oneclick.isml`

#### **templates/default/checkout/summary/summary.isml**

After the Easy Payment custom code, modify the `summary.isml` adding the OneClick custom code, as follows:

```

<isslot id="placeorder-slot" description="Slot next to Order Totals in the footer of the Place Order page." context="global"/>

<iscomment>---- BEGIN CUSTOM CODE ----</iscomment>

<isscript>
  var XPayUtils = require("**/cartridge/scripts/util/XPayUtils");
</isscript>

<iscomment>---- BEGIN XPay BUILD ----</iscomment>
<isif condition="${XPayUtils.XPAY_PAYMODE === XPayUtils.PM_XPAY_BUILD}">

  <div class="order-summary-footer">
    <div class="place-order-totals">
      <isordertotals p_lineitemctnr="${pdict.Basket}" p_showshipmentinfo="${false}" p_shipmenteditable="${false}" p_totallabel="${Resource.msg('summary.ordertotal','checkout',null)}"/>
    </div>
    <form action="${URLUtils.https('COSummary-Submit')}}" method="post" class="submit-order">
      <fieldset>
        <iscomment>XPay Recurring Payment</iscomment>
        <include template="/checkout/summary/xpay_recurpay" />

        <include template="/checkout/billing/xpay_build" />
        <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 id="xpay-build-orderbtn"
        class="button-fancy-large" disabled="disabled"
        type="submit" name="submitBtn"
        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>
<iscomment>----- END XPay BUILD -----</iscomment>

<iscomment>----- BEGIN EASY PAYMENT -----</iscomment>
<elseif condition="{(XPayUtils.XPAY_PAYMODE === XPayUtils.PM_EASY_PAYMENT) ||
    (XPayUtils.XPAY_PAYMODE === XPayUtils.PM_ONECLICK && empty(customer.profile))}" />
<div class="order-summary-footer">
    <div class="place-order-totals">
        <isordertotals p_lineitemctnr="{pdict.Basket}" p_showshipmentinfo="{false}" p_shipmenteditable="{false}" p_totallabel="{Resource.msg('summary.ordertotal','checkout',null)}" />
    </div>
    <form action="{URLUtils.https('XPayOrderMng-CreateOrder')}" method="post" id="form-xpay-easypay" 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>
                <iscomment>XPay Recurring Payment</iscomment>
                <isinclude template="/checkout/summary/xpay_recurpay" />

                <isinclude template="/checkout/summary/xpay_easypay" />
            </div>
        </fieldset>
    </form>
</div>
<iscomment>----- END EASY PAYMENT -----</iscomment>

<iscomment>----- BEGIN XPay ONECLICK -----</iscomment>
<elseif condition="{(XPayUtils.XPAY_PAYMODE === XPayUtils.PM_ONECLICK && !empty(customer.profile))}" />
<div class="order-summary-footer">
    <div class="place-order-totals">
        <isordertotals p_lineitemctnr="{pdict.Basket}" p_showshipmentinfo="{false}" p_shipmenteditable="{false}" p_totallabel="{Resource.msg('summary.ordertotal','checkout',null)}" />
    </div>
    <form action="{URLUtils.https('XPayOrderMng-CreateOrder')}" method="post" id="form-xpay-oneclick" 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>
                <isinclude template="/checkout/summary/xpay_oneclick" />
            </div>
        </fieldset>
    </form>
</div>
<iscomment>----- END XPay ONECLICK -----</iscomment>

<iscomment>----- BEGIN Default Payment of Site Genesis -----</iscomment>
<elseif />
<div class="order-summary-footer">
    <div class="place-order-totals">
        <isordertotals p_lineitemctnr="{pdict.Basket}" p_showshipmentinfo="{false}" p_shipmenteditable="{false}" p_totallabel="{Resource.msg('summary.ordertotal','checkout',null)}" />
    </div>
    <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>
<iscomment>---- END Default Payment of Site Genesis ----</iscomment>

<iscomment>---- END CUSTOM CODE ----</iscomment>

</isdecorate>

```

## 14 Customize stylings

In order to apply your custom css stylings to the XPay html components, you need to create a template with the `pt_` prefix that define the look and feel of your customized checkout page (for example, fonts, colors and display properties).  
To know more about naming convention and page structure of Site Genesis pages, see the Commerce Cloud documentation at SiteGenesis and CSS.

A sample of a site customization is provided in the following folder of the provided package:

LINK-SiteGenesis-NexiXPay\cc\_customization

### ***pt\_checkout\_UI.isml***

This file defines resources that are automatically included in the header of your checkout page.

Extract the following file from the provided package:

LINK-SiteGenesis-NexiXPay\cc\_customization\cartridge\templates\default\checkout\pt\_checkout\_UI.isml

to your site implementation folder:

<yoursite>\cartridge\templates\default\checkout\pt\_checkout\_UI.isml

### ***xpay.css***

This is a style sheet that customize the look and feel of XPay html elements. It is automatically injected into your checkout page.

Extract the following file from the provided package:

LINK-SiteGenesis-NexiXPay\cc\_customization\cartridge\static\default\css\xpay.css

to your site implementation folder:

<yoursite>\cartridge\static\default\css\xpay.css