Zakeke 3d Configurator

Sitegenesis extension

*link\_zakeke v: 21.1.3*



****

Table of Contents

[Summary 4](#_Toc31181455)

[1. Component Overview 5](#_Toc31181456)

[Functional Overview 5](#_Toc31181457)

[Use Cases 5](#_Toc31181458)

[Limitations, Constraints 5](#_Toc31181459)

[Compatibility 5](#_Toc31181460)

[Privacy, Payment 5](#_Toc31181461)

[2. Implementation Guide 6](#_Toc31181462)

[Setup 6](#_Toc31181463)

[Deploying cartridge to a sandbox 6](#_Toc31181464)

[Sandbox setup 6](#_Toc31181465)

[Configuration 6](#_Toc31181466)

[Commerce Cloud: register your application API Client using Account manager 6](#_Toc31181467)

[Zakeke platform: configuring the account 7](#_Toc31181468)

[Commerce Cloud: configuring the site preferences 9](#_Toc31181469)

[Commerce Cloud: configuring the credential 9](#_Toc31181470)

[Commerce Cloud: configuring the services 10](#_Toc31181471)

[Commerce Cloud: configuring the OCAPI setting 11](#_Toc31181472)

[Commerce Cloud: configuring the custom job ZakekeNotification 12](#_Toc31181473)

[Commerce Cloud: configuring log settings 13](#_Toc31181474)

[Custom Code 14](#_Toc31181475)

[External Interfaces 24](#_Toc31181476)

[Firewall Requirements 25](#_Toc31181477)

[3. Testing 26](#_Toc31181478)

[4. Operations, Maintenance 27](#_Toc31181479)

[Data Storage 27](#_Toc31181480)

[Availability 27](#_Toc31181481)

[Support 28](#_Toc31181482)

[5. User Guide 29](#_Toc31181483)

[Roles, Responsibilities 29](#_Toc31181484)

[Business Manager 29](#_Toc31181485)

[Storefront Functionality 29](#_Toc31181486)

[6. Known Issues 34](#_Toc31181487)

[7. Release History 34](#_Toc31181488)

# Summary

This document describes how to implement Zakeke cartridge in SiteGenesis platform.

This APP includes an extension for “Commerce Cloud” to plugs a cloud-based component into your web store and allows you to create configurable products and set personalization rules to offer your customers an immersive and interactive "personalize-it" experience.

Zakeke offers two types of personalized shopping experience:

“Zakeke - 3D Product Configurator” is a product configurator in real-time 3D to enable customers to compose store products variations according to their needs and tastes.

“Zakeke - 3D Product Customizer” is a product customizer that enables customers to add text customizations, custom images, variations, according to their needs and tastes and let them see the result in a 3d environment.

This APP requires a Business Manager configuration of the cartridges and the use of Zakeke platform to prepare and configure the products.

The integration encompasses three cartridges: a business cartridge (int\_zakeke\_core), a storefront cartridge (int\_zakeke) and a cartridge for Business Manager (bm\_zakeke). Furthermore, the integration includes an external software module which calls OCAPI Web API to have information about the items of the store.

# **Component Overview**

## Functional Overview

The present cartridges let you get both a cloud-based product configurator or a product customizer that you can use into your web store. These plugins enable final customers to compose/customize chosen product according to their needs and tastes.

## Use Cases

The Zakeke customization process can be divided in the following functional phases:

* Configure products for Zakeke customization using external Zakeke platform.
* Select product to customize in the storefront PDP (product detail page) through a “customize” button.
* An additional web page that includes an external proprietary plugin gives customers the tools for customize/configure the selected product using a 3d model. The plugin interacts with Zakeke platform using Zakeke Web API.
* The client adds the product in the cart. The extension saves the customization details in the storefront DB.
* At the end of the order, a custom job, configured in the business manager, notifies Zakeke platform for positively closed orders.

## Limitations, Constraints

At the moment the products that can be customized are:

* standard product
* variants of a master product

It’s not possible to customize bundle, set and options product.

## Compatibility

Cartridge is designed and developed for:

* Salesforce platform version 21.2
* Sitegenesis version 105.1.0
* Compatibility mode 21.2

This APP supports the following locales: en, es, de, it, fr.

## Privacy, Payment

Zakeke process doesn’t access customer profile data and no credit cards are stored and processed during customization process.

For Zakeke’s privacy agreement go to <https://www.zakeke.com/privacy>

# Implementation Guide

## Setup

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

### Deploying cartridge to a sandbox

1. Upload int\_zakeke, int\_zakeke\_core and bm\_zakeke to your production or development sandbox. Apply standard flow to upload cartridges to your environment.

### Sandbox setup

1. Go to Business Manager -> Site -> Manage Sites. Select correct “Sitegenesis” site, then select Settings tab. In cartridge path at the end write the following:

*int\_zakeke:int\_zakeke\_core*

1. Go to Business Manager -> Site -> Manage Sites. Click on “Business manager” site, then select Settings tab. In cartridge path at the end write the following:

*bm\_zakeke:int\_zakeke\_core*

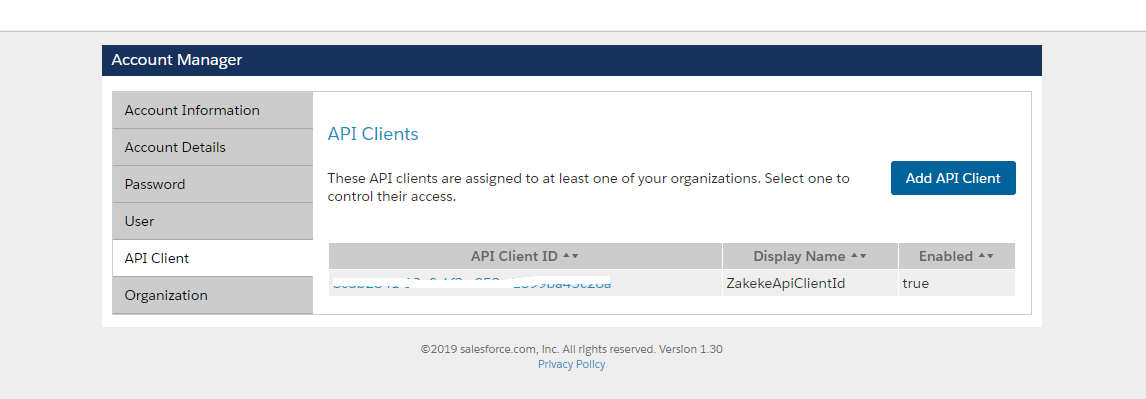
1. In the “metadata” folder of Zakeke LINK cartridge, downloads the zak.zip file.
2. Import zip file to Site Development > Site Import & Export
3. Verify that the Site Preferences group was created with the “Zakeke Configs” ID and “Zakeke Configs”name.

## Configuration

Section describes steps that should be completed to configure the extension both in the Business Manager or in the Zakeke platform front-end.

### Commerce Cloud: register your application API Client using Account manager

1. Go to account manager to [https://account.demandware.com](https://account.demandware.com/) and log in with administrator credential. Click the link API client.



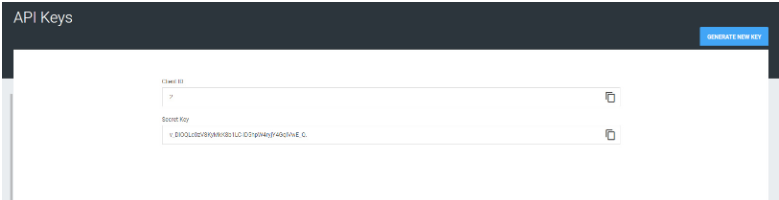
1. Click “Add API Client” button.
2. Insert *ZakekeApiClientId* as Display name, select *client\_secret\_basic* as Token Endpoint Auth Method.
3. Save.
4. A new **API client ID** will be created. The **API secret key** correspond to the password of the logged account. It’s advisable to change the password and choose a different password for every client ID.
5. Take a copy of the preceding values.

### Zakeke platform: configuring the account

1. Go to <https://portal.zakeke.com/en-US/Composer/Login> and log in using your merchant credential (ask Zakeke for a merchant account with credentials).
2. The “customizer” dashboard page opens. If you would like to use the configurator instead, switch to “3d product configurator” clicking the link at the bottom of the page.

For “**Customizer**”:

1. Go to “Api keys” page. If “Api ClientId” is not present click the “Generate new key” button. A new Api clientID and Api secret key will be generated by the system.



1. Copy "Client ID" and "Secret Key" fields. They are the "**Zakeke Client Id**" and the “**Zakeke Client Secret**”.
2. Go to Advanced -> E-commerce. Choose Salesforce tab.
3. Fill the form as below:

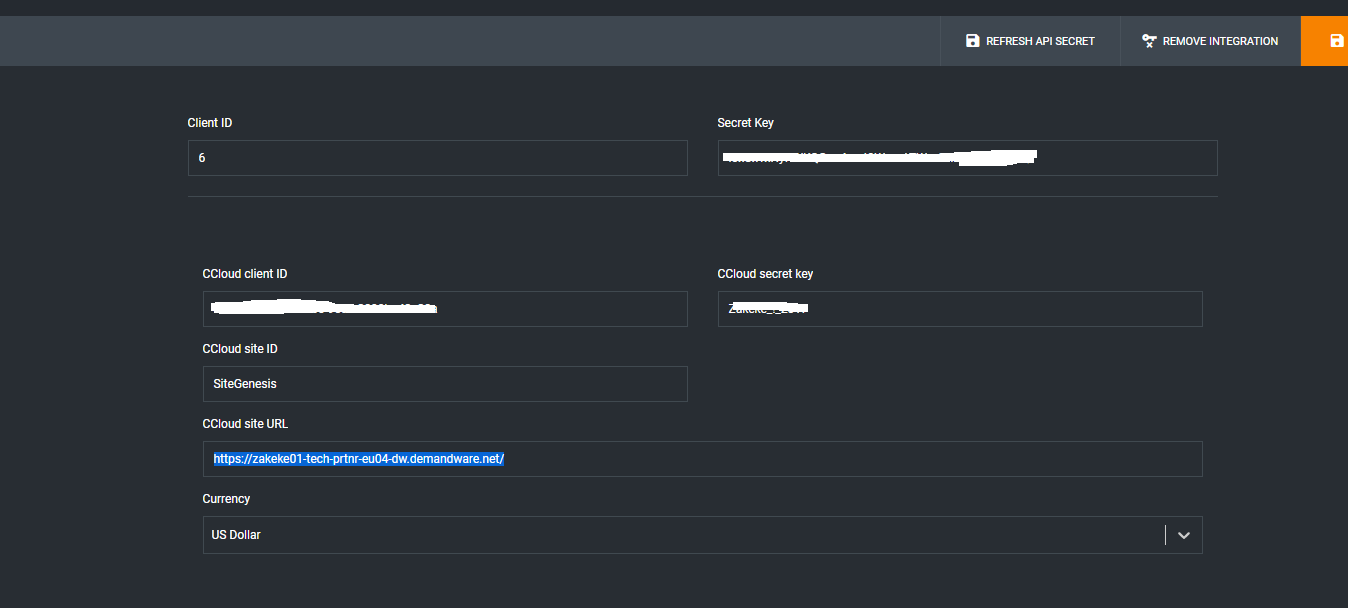
* **CCloud client id**: insert the "Api client ID" previously generated into account manager.
* **CCloud secret key**: insert the “Api secret key” previously generated into account manager.
* **CCloud site id:**insert the “SiteID” of the web site linked to Zakeke platform (es. sitegenesis)
* **CCloud site URL:** insert the base URL of Commerce Cloud store (es. https://domain01-dw.demandware.net/)
* **Currency:**choose the base currency of your store

For “**Configurator**”:

1. Navigate to “Integration” link and select “Commerce cloud”.
2. Fill the form as below:

* **CCloud client id**: insert the "Api client ID" previously generated into account manager.
* **CCloud secret key**: insert the “Api secret key” previously generated into account manager.
* **CCloud site id**: insert the “SiteID” of the web site linked to Zakeke platform (es. sitegenesis)
* **CCloud site URL**: insert the base URL of Commerce Cloud store (es. https://domain01-dw.demandware.net/)
* **Currency**: choose the base currency of your store

1. Copy "Client ID" and "Secret Key" fields at the top of the form. They are the "**Zakeke Client Id**" and the “**Zakeke Client Secret**”.
2. Click on “Save integration”.

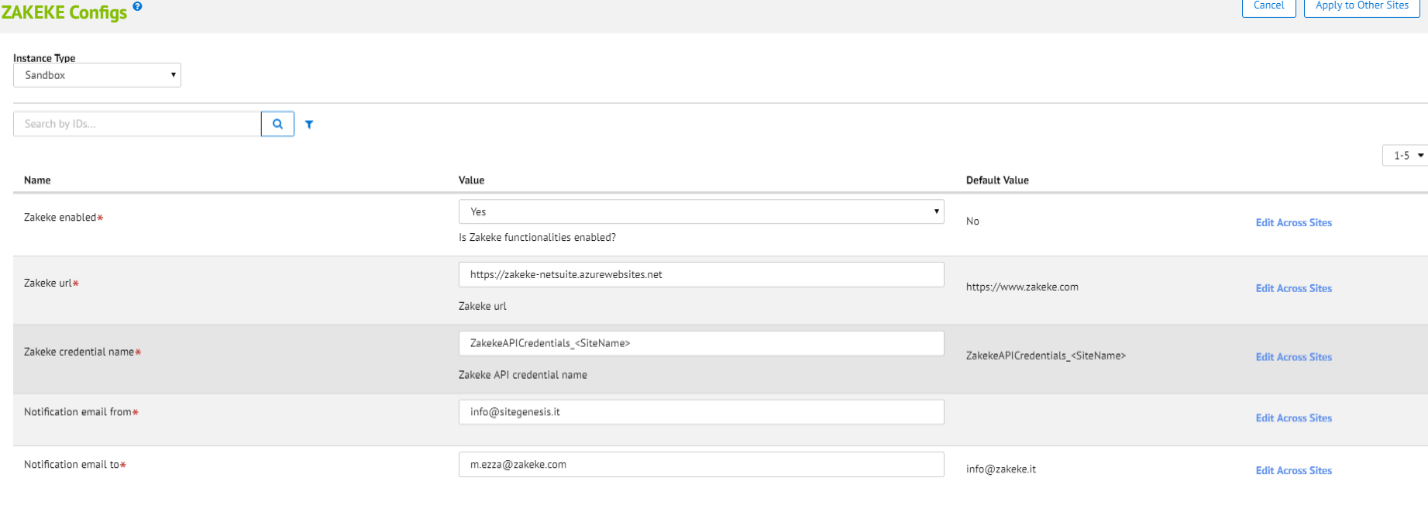


### Commerce Cloud: configuring the site preferences

1. Choose your site. Go to Business Manager-> Merchant tools -> Custom preferences.
2. Click to “ZAKEKE Configs”.
3. Fill the form as below:

* **Zakeke enabled**: choose “Yes” to enable the Zakeke extension.
* **Zakeke url**: insert the base url of Zakeke platform (https://portal.zakeke.com).
* **Zakeke credential name**: insert the string *ZakekeAPICredentials\_<SiteName>* putting the store siteID in place of <SiteName> (es. ZakekeApiCredentials\_Sitegenesis).
* **Notification email from**: insert the email address to receive the notification from.
* **Notification email to**: insert the emails to send the notifications to.

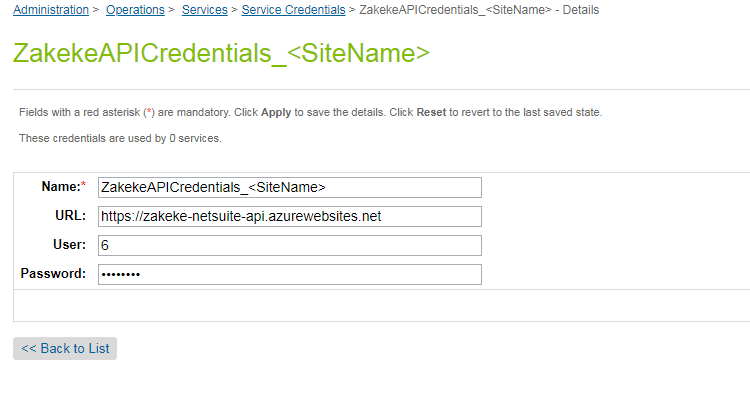
1. Save the configurations.



### Commerce Cloud: configuring the credential

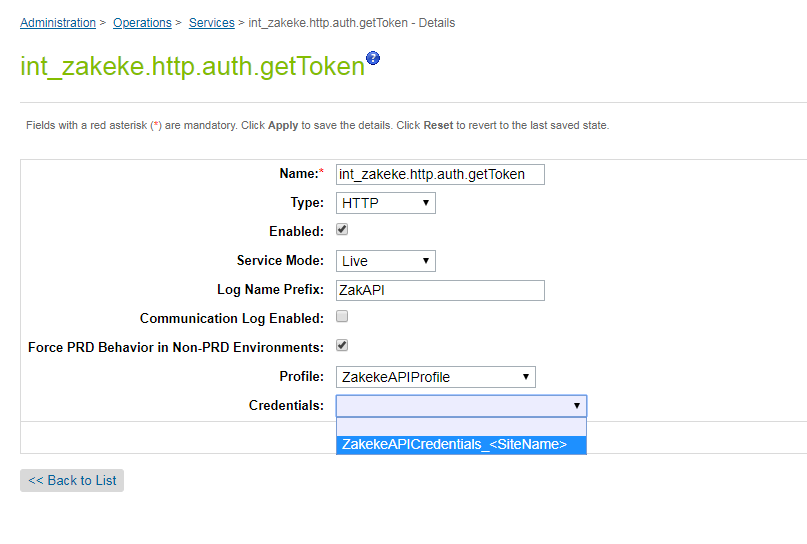
1. Go to Administrator -> Operations -> Services.
2. Go to Credentials and click to [ZakekeAPICredentials\_<SiteName>](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ServiceCredential-DisplayDetails?ServiceCredentialUUID=2c549deb94b0eac38bed5f97e1&csrf_token=AyxOUOe8SlUPyXUGhcuJQM90FCe7rak6zkxqk8p83vacE5OcmM4L6Yin9aKKU--0BtI_09C5wdKYSapGE3rchRsAamHcltleT2dsv1TzlJxSpmqZ1HUyirILYC77bj7HNCiD5NVP3sArVMzDFOnqgD36GTVukEpfOBYKxNsIfXqKA4UuZVc=)
3. Fill the form as below:

* **Name**: update the string *ZakekeAPICredentials\_<SiteName>* putting the store *siteID* in place of <SiteName>. This value has to be equal to “Zakeke credential name” inserted above in the custom site preferences.
* **URL**: insert the base url of Zakeke platform api (https://api.zakeke.com).
* **User**: insert the “Api client ID” previously generated in Zakeke configuration phase (**Zakeke Client Id)**.
* **Password**: insert the “Api client secret” previously generated in Zakeke configuration phase (**Zakeke Client Secret)**.



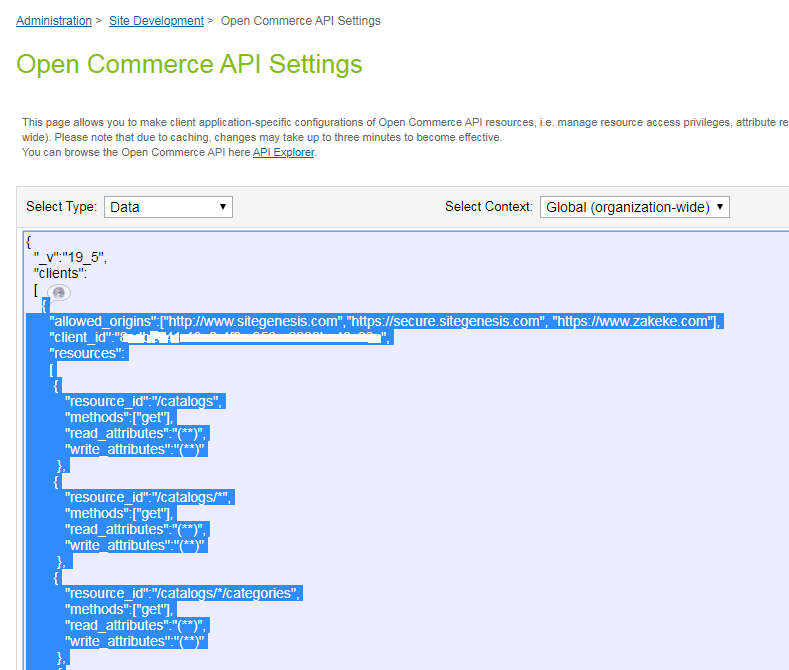
### Commerce Cloud: configuring the services

1. Go to Administrator -> Operations -> Services.
2. Go to Services and **for each services** in the list repeat the step 3
3. In the list credentials, select the credential defined in the previous step. See the image below.



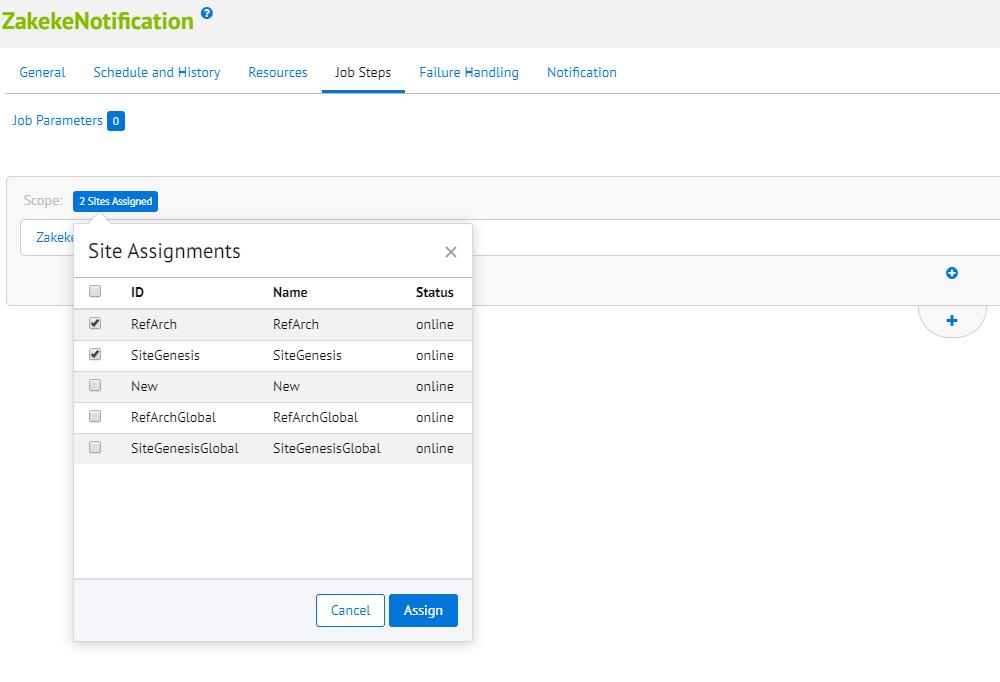
### Commerce Cloud: configuring the OCAPI setting

1. Go to Administrator -> Site development -> Open Commerce API Settings.
2. Select “Data” as Type and “Global (organization-wide)” as context.
3. In the text field, add a new JSON object in the array named “clients”. The object is specified in the **Ocapi.txt** file that is contained in the documentation directory of the project.
4. Remember to insert into “**client id**” field the Commerce Cloud “*Api client Id*” generated into account manager. See the image below.

**

### Commerce Cloud: configuring the custom job ZakekeNotification

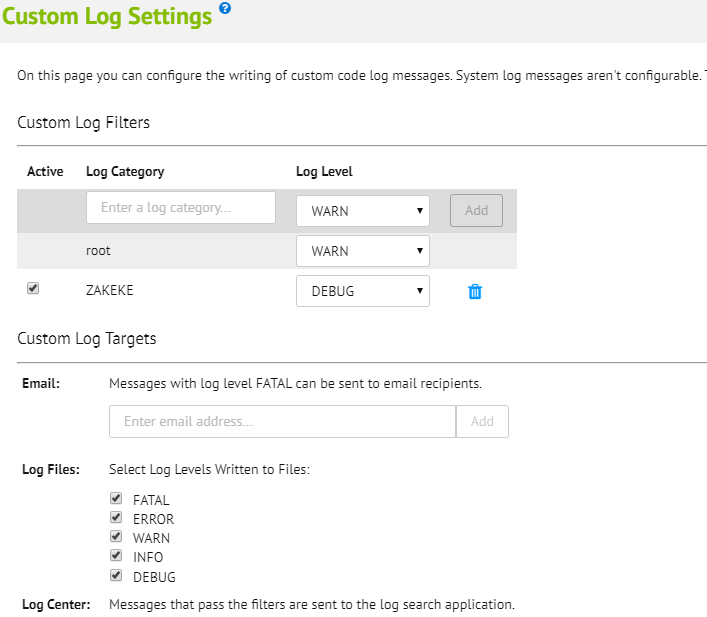
1. Go to Administrator -> Operations -> Jobs
2. Click on job ZakekeNotification
3. In the “Schedule and History” page enable the job, clicking the “Enabled” check box
4. Go to Job steps page
5. Assign the siteID of your store as context of job step. See the image below



### Commerce Cloud: configuring log settings

1. Go to Administrator -> Operations -> Custom Log settings.
2. Add a new *Log Category* for Zakeke called ZAKEKE

See the image below.



## Custom Code

Zakeke application needs some updates on Sitegenesis storefront files. The files to be modified are the following:

**storefront\_controllers**:

* storefront\_controllers/cartridge/controllers/COSummary.js

**storefront\_core**:

* storefront\_core /cartridge/templates/default/checkout/cart/cart.isml
* storefront\_core /cartridge/templates/default/checkout/components/minilineitems.isml
* storefront\_core /cartridge/templates/default/checkout/summary/summary.isml
* storefront\_core /cartridge/templates/default/product/components/displayliproduct.isml
* storefront\_core /cartridge/templates/default/product/productcontent.isml
* storefront\_core/cartridge/scripts/cart/calculate.js

You can find the changed version of this files in the documentation directory of the project. Let’s see each of them in detail.

**storefront\_controllers/cartridge/controllers/COSummary.js**

Substitute the function start() with the code below:

function start(context) {

var cart = Cart.get();

// Checks whether all payment methods are still applicable. Recalculates all existing non-gift certificate payment

// instrument totals according to redeemed gift certificates or additional discounts granted through coupon

// redemptions on this page.

var COBilling = app.getController('COBilling');

if (!COBilling.ValidatePayment(cart)) {

COBilling.Start();

return;

} else {

Transaction.wrap(function () {

cart.calculate();

});

Transaction.wrap(function () {

if (!cart.calculatePaymentTransactionTotal()) {

COBilling.Start();

}

});

var pageMeta = require('~/cartridge/scripts/meta');

//???????????????????????????????????????????

//Zakeke customize begin---------------------

/\*var viewContext = require('storefront\_core/cartridge/scripts/common/extend').immutable(context, {

Basket: cart.object

});\*/

var viewContext = require('\*/cartridge/scripts/common/extend').immutable(context, {

Basket: cart.object

});

//Zakeke customize end-----------------------

//???????????????????????????????????????????

pageMeta.update({pageTitle: Resource.msg('summary.meta.pagetitle', 'checkout', 'SiteGenesis Checkout')});

app.getView(viewContext).render('checkout/summary/summary');

}

}

Substitute the function submit() with the code below:

**function** submit() {

// Calls the COPlaceOrder controller that does the place order action and any payment authorization.

// COPlaceOrder returns a JSON object with an order\_created key and a boolean value if the order was ..

// If the order creation failed, it returns a JSON object with an error key and a boolean value.

**var** placeOrderResult = app.getController('COPlaceOrder').Start();

**if** (placeOrderResult.error) {

start({

PlaceOrderError: placeOrderResult.PlaceOrderError

});

} **else** **if** (placeOrderResult.order\_created) {

showConfirmation(placeOrderResult.Order);

//???????????????????????????????????????????

//Zakeke customize begin---------------------

dw.system.HookMgr.callHook('zak.orderClosed', 'orderClosed', placeOrderResult.Order);

//Zakeke customize end-----------------------

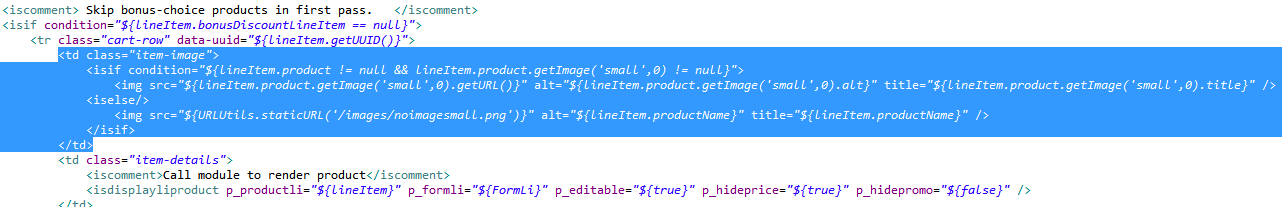
//???????????????????????????????????????????

}

}

**storefront\_core/cartridge/templates/default/checkout/cart/cart.isml**

Substitute the selected code:



With:

<!--Zakeke customize begin---------------------- -->

<isif condition="${!empty(lineItem.getCustom().ZAKEKE\_CompositionID)}">

<td class="item-image">

<img src="${lineItem.getCustom().ZAKEKE\_CompositionPreview}" alt="${lineItem.productName}" title="${lineItem.productName}" width="65px" height="65px"/>

</td>

<iselse/>

<td class="item-image">

<isif condition="${lineItem.product != null && lineItem.product.getImage('small',0) != null}">

<img src="${lineItem.product.getImage('small',0).getURL()}" alt="${lineItem.product.getImage('small',0).alt}" title="${lineItem.product.getImage('small',0).title}" />

<iselse/>

<img src="${URLUtils.staticURL('/images/noimagesmall.png')}" alt="${lineItem.productName}" title="${lineItem.productName}" />

</isif>

</td>

</isif>

<!--Zakeke customize end---------------------- -->

Substitute the selected code:



With:

<!--Zakeke customize begin---------------------- -->

<isif condition="${!empty(lineItem.getCustom().ZAKEKE\_CompositionID)}">

<isdisplayliproduct p\_productli="${lineItem}" p\_formli="${FormLi}" p\_editable="${false}" p\_hideprice="${true}" p\_hidepromo="${false}" />

<iselse/>

<isdisplayliproduct p\_productli="${lineItem}" p\_formli="${FormLi}" p\_editable="${true}" p\_hideprice="${true}" p\_hidepromo="${false}" />

</isif>

<!--Zakeke customize end---------------------- -->

Substitute the selected code:



With:

<div class="item-user-actions">

<isif condition="${!lineItem.bonusProductLineItem || lineItem.getBonusDiscountLineItem() != null}">

<button class="button-text" type="submit" value="${Resource.msg('global.remove','locale',null)}" name="${FormLi.deleteProduct.htmlName}"><span>${Resource.msg('global.remove','locale',null)}</span></button>

</isif>

<!--Zakeke customize begin---------------------- -->

<isif condition="${!!empty(lineItem.getCustom().ZAKEKE\_CompositionID)}">

<isif condition="${empty(pdict.ProductAddedToWishlist) || pdict.ProductAddedToWishlist != lineItem.productID}">

<isif condition="${!isInWishList && !empty(lineItem.product)}">

<a class="add-to-wishlist" href="${URLUtils.https('Cart-AddToWishlist', 'pid', lineItem.productID)}" name="${FormLi.addToWishList.htmlName}" title="${Resource.msg('global.addthisitemtowishlist.title','locale',null)}">

${Resource.msg('global.addtowishlist','locale',null)}

</a>

</isif>

<iselse/>

<div class="in-wishlist">

${Resource.msg('global.iteminwishlist','locale',null)}

</div>

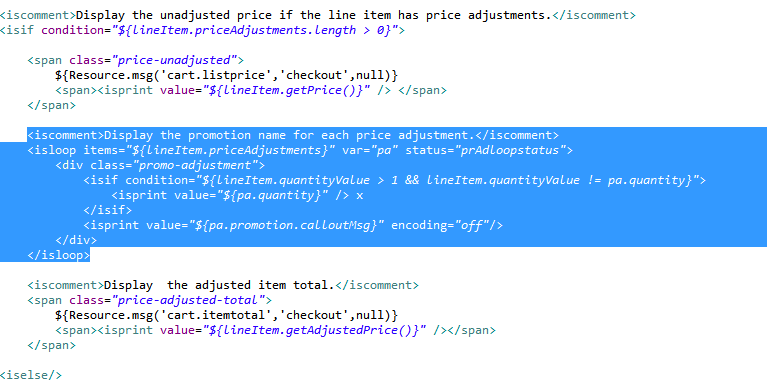
</isif>

</isif>

<!--Zakeke customize begin---------------------- -->

</div>

Substitute the selected code:



With:

<iscomment>Display the promotion name for each price adjustment.</iscomment>

<isloop items="${lineItem.priceAdjustments}" var="pa" status="prAdloopstatus">

<div class="promo-adjustment">

<isif condition="${lineItem.quantityValue > 1 && lineItem.quantityValue != pa.quantity}">

<!--Zakeke customize begin---------------------- -->

<!-- isprint value="${pa.quantity}" /> x -->

<!--Zakeke customize begin---------------------- -->

</isif>

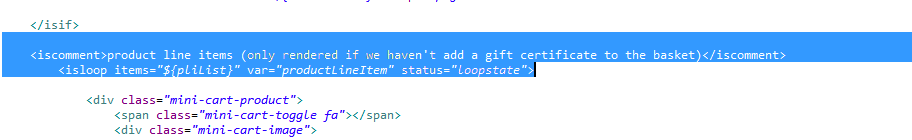
<isprint value="${pa.promotion.calloutMsg}" encoding="off"/>

</div>

</isloop>

**storefront\_ core /cartridge/templates/default/checkout/components/minilineitems.isml**

After the selected code:

****

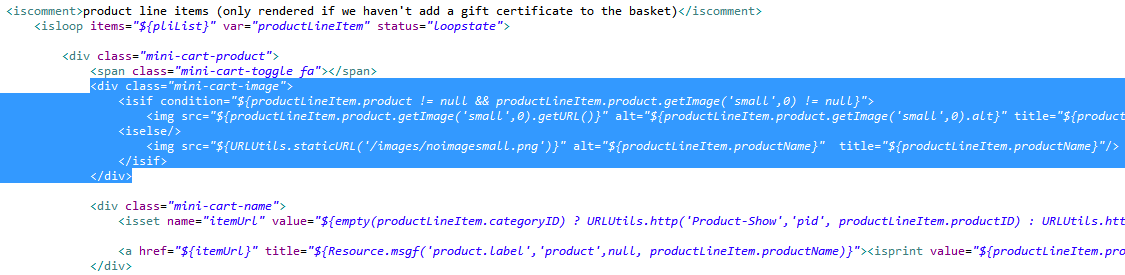
Insert the code:

<!--Zakeke customize begin---------------------- -->

<isset name="IsZakekeProduct" value="${!empty(productLineItem.getCustom().ZAKEKE\_CompositionID)}" scope="page" />

<!--Zakeke customize end---------------------- -->

Substitute the selected code:



With the code:

<div class="mini-cart-image">

<!--Zakeke customize begin---------------------- -->

<isif condition="${IsZakekeProduct}">

<img src="${productLineItem.getCustom().ZAKEKE\_CompositionPreview}" alt="${productLineItem.productName}" title="${productLineItem.productName}" width="65px" height="65px"/>

<iselse/>

<isif condition="${productLineItem.product != null && productLineItem.product.getImage('small',0) != null}">

<img src="${productLineItem.product.getImage('small',0).getURL()}" alt="${productLineItem.product.getImage('small',0).alt}" title="${productLineItem.product.getImage('small',0).title}"/>

<iselse/>

<img src="${URLUtils.staticURL('/images/noimagesmall.png')}" alt="${productLineItem.productName}" title="${productLineItem.productName}"/>

</isif>

</isif>

<!--Zakeke customize end---------------------- -->

</div>

**storefront\_ core/cartridge/templates/default/checkout/summary/summary.isml**

Substitute the selected code:



With the code:

<!--Zakeke customize begin---------------------- -->

<isif condition="${!empty(productLineItem.getCustom().ZAKEKE\_CompositionID)}">

<td class="item-image">

<img src="${productLineItem.getCustom().ZAKEKE\_CompositionPreview}" alt="${productLineItem.productName}" title="${productLineItem.productName}" width="65px" height="65px"/>

</td>

<iselse/>

<td class="item-image">

<isif condition="${productLineItem.product != null && productLineItem.product.getImage('small',0) != null}">

<img src="${productLineItem.product.getImage('small',0).getURL()}" alt="${productLineItem.product.getImage('small',0).alt}" title="${productLineItem.product.getImage('small',0).title}"/>

<iselse/>

<img src="${URLUtils.staticURL('/images/noimagesmall.png')}" alt="${productLineItem.productName}" title="${productLineItem.productName}"/>

</isif>

<isif condition="${productLineItem.bonusProductLineItem}">

<div class="bonus-item">

<isset name="bonusProductPrice" value="${productLineItem.getAdjustedPrice()}" scope="page"/>

<isinclude template="checkout/components/displaybonusproductprice" />

<isprint value="${bonusProductPriceValue}" />

</div>

</isif>

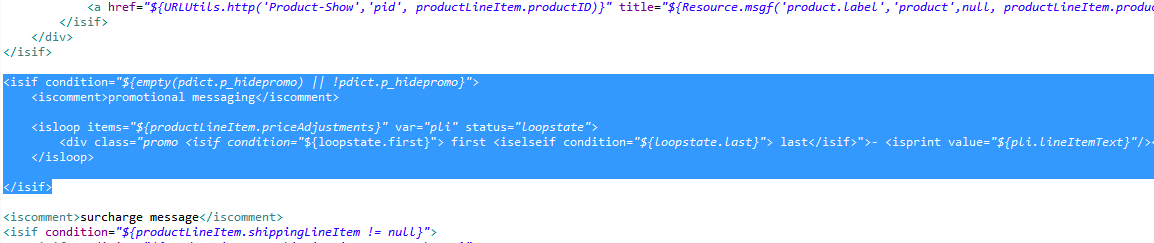
</td>

</isif>

<!--Zakeke customize end---------------------- -->

**storefront\_ core /cartridge/templates/default/product/components/displayliproduct.isml**

Substitute the selected code:



With:

<isif condition="${empty(pdict.p\_hidepromo) || !pdict.p\_hidepromo}">

<iscomment>promotional messaging</iscomment>

<!--Zakeke customize begin---------------------- -->

<isloop items="${productLineItem.priceAdjustments}" var="pli" status="loopstate">

<isif condition="${pli.promotionID!=productLineItem.getCustom().ZAKEKE\_CompositionID}">

<div class="promo <isif condition="${loopstate.first}"> first <iselseif condition="${loopstate.last}"> last</isif>">- <isprint value="${pli.lineItemText}"/></div>

</isif>

</isloop>

<!--Zakeke customize end---------------------- -->

</isif>

Insert after the selected code (before the </div>):



The code:

<!--Zakeke customize begin---------------------- -->

<isif condition="${!empty(productLineItem.getCustom().ZAKEKE\_Composition)}">

<br>

${Resource.msg('displayliproduct.productPriceAdjustment','zakeke',null)}: <b>${productLineItem.getCustom().ZAKEKE\_CompositionID.slice(0,20)} ..</b><br>

${Resource.msg('displayliproduct.customAttributes','zakeke',null)}:

<b>

<isloop items="${JSON.parse(productLineItem.getCustom().ZAKEKE\_Composition)}" var = "zakOption">

[${zakOption.optionID}: ${zakOption.valueID}]

</isloop>

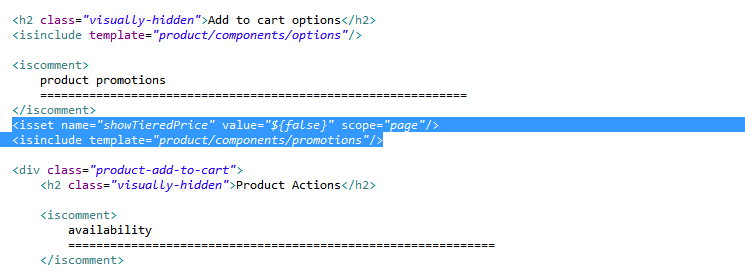
</b>

</isif>

<!--Zakeke customize end---------------------- -->

**storefront\_ core/cartridge/templates/default/product/productcontent.isml**

After the selected code:



Insert the code (before <div class=”product-add-to-cart”>):

<!--Zakeke customize begin---------------------- -->

<isset

name="enabled"

value="${(pdict.isProductAvailable && !pdict.Product.bundle && !pdict.Product.optionProduct && !pdict.Product.master && (pdict.Product.product || pdict.Product.variant))}"

scope="page"

/>

<isif condition="${enabled}" >

<isinclude url="${URLUtils.url('ZakButtonCustomize-Show', 'pid', productNumber, 'enabled', enabled )}" />

</isif>

<!--Zakeke customize end------------------------ -->

**storefront\_core/cartridge/scripts/cart/calculate.js**

The module calculate.js contains the function calculate. Add the following codes in the calculate function between PromotionMgr. applyDiscounts(basket) and calculateProductPrices(basket) in this way:

PromotionMgr.applyDiscounts(basket);

//----------------------------------------------

//Zakeke customize begin---------------------

calculateProductPricesAdjustmentZakeke(basket);

//Zakeke customize end-----------------------

//----------------------------------------------

calculateProductPrices(basket);

See image below for greater clarity:



After the “calculate” function insert these two new functions “calculateProductPricesAdjustmentZakeke” and “isToCallCustomizationModel” whose code is reported below:

/\*\*

\* @function calculateProductPricesAdjustmentZakeke

\*

\* Add product price adjustment only for customized product

\* in the product line items. This function returns nothing

\*

\* @param {object} basket The basket containing the elements to be computed

\*/

function calculateProductPricesAdjustmentZakeke(basket){

var Resource = require('dw/web/Resource');

var zakGlobalSite = require('\*/cartridge/scripts/zakGlobalSite');

var ZakCustomizationModel=require('\*/cartridge/scripts/models/ZakCustomizationModel');

var productLineItems = basket.getAllProductLineItems().iterator();

while (productLineItems.hasNext()) {

var productLineItem = productLineItems.next();

var fixedAmount=null;

if (!empty(productLineItem.getCustom().ZAKEKE\_CompositionID)) {

var compositionId=productLineItem.getCustom().ZAKEKE\_CompositionID;

var quantity=productLineItem.quantity.value;

var productID=productLineItem.productID;

var customizationType=productLineItem.getCustom().ZAKEKE\_CustomizationType;

var platformType=zakGlobalSite.PLATFORM\_USED;//SITEGENESIS

var priceCustomiz=productLineItem.getCustom().ZAKEKE\_CompositionPrice;

var priceCustomizNoTaxes=productLineItem.getCustom().ZAKEKE\_CompositionPriceNoTaxes;

//ZAKEKE\_Control: this variable is used to choose if customizationAPI call has to be done or not

//It saves the preceding "<compositionId>\_\_\_\_<quantity>" for which the call has already been done

var controlCustomizationCall=productLineItem.getCustom().ZAKEKE\_Control;

var controlObjectReturn= {control: ''};

var toCall=isToCallCustomizationModel(controlCustomizationCall, compositionId, quantity, controlObjectReturn);

if(toCall){

var modelCustomization=ZakCustomizationModel.get(productID, compositionId, quantity, customizationType, platformType);

var obj=modelCustomization.customization(); //obj={compositionPreview: xxxx, compositionUnitPrice: xxxx, compositionUnitPriceNoTaxes: xxxxx}

priceCustomiz=obj.compositionUnitPrice;

priceCustomizNoTaxes=obj.compositionUnitPriceNoTaxes;

productLineItem.getCustom().ZAKEKE\_CompositionPreview = obj.compositionPreview;

productLineItem.getCustom().ZAKEKE\_CompositionPrice = priceCustomiz;//Unit composition price in session currency (WITH TAXES)

productLineItem.getCustom().ZAKEKE\_CompositionPriceNoTaxes = priceCustomizNoTaxes;//Unit composition price in session currency (NO TAXES)

productLineItem.getCustom().ZAKEKE\_Control = controlObjectReturn.control;

}

if (priceCustomiz>0){

var adj=productLineItem.getPriceAdjustmentByPromotionID(compositionId);

if (empty(adj)){

fixedAmount = productLineItem.createPriceAdjustment(compositionId);

}

else {

fixedAmount=adj;

}

fixedAmount.setManual(false);

fixedAmount.setPriceValue(priceCustomiz\*quantity);

fixedAmount.setLineItemText(Resource.msg('displayliproduct.productPriceAdjustment','zakeke',null)+': '+compositionId);

}

}

}

}

function isToCallCustomizationModel(control, compositionId, quantity, controlReturn){

var toCall=false;

var chrSpli='\_\_\_\_';//4chr

var controlArray=null;

if(empty(control)){

toCall=true;

controlReturn.control = compositionId+chrSpli+quantity;

}

else{

controlArray=control.split(chrSpli);

var compId=controlArray[0];

var quant=controlArray[1];

if(compId!=compositionId || quant!=quantity){

toCall=true;

controlReturn.control = compositionId+chrSpli+quantity;

}

}

return toCall;

}

## External Interfaces

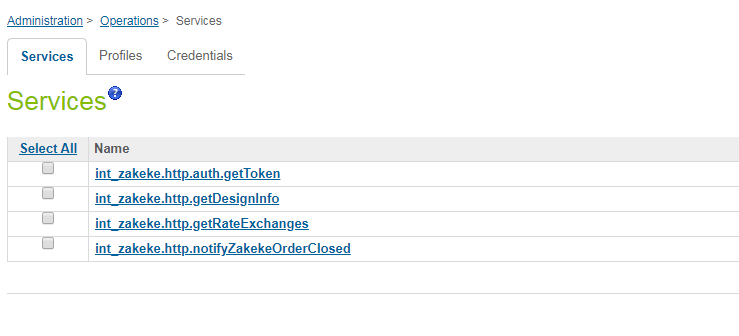
Zakeke platform use OCAPI Data API to get information about sites, catalog and products of the storefront.

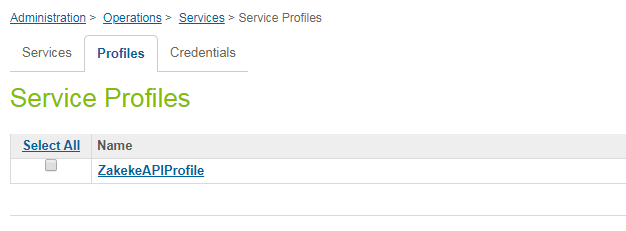
They are mainly “read” operation (GET). The unique “write” operation (PATCH) is on the product document. When the merchant, using Zakeke platform, has completed the configuration of the product, an OCAPI PATCH request has made to update the custom product fields ‘’[**ZAKEKE\_is\_customizable**](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=d53034a5cb9e03cce877d4b4c6&AttributeDefinitionUUID=fec75094cd6274e35ac0262a5a&csrf_token=0qJ5sb-QnvzrIyPJSC04PC3nsIyJvKKIqZch8krWnnrKHIoev6jLfJylCalvtzUqnrosdHqzDgWFdWXsmcQCXk6nELW5gLfPP8H9nxtiGXeykMb7_CP_NtfmU06frzifZv7V-CoFWxyFP4rW-rWDpCvyV9U55Ac6O5a6umIAjBjnegsm6ZE=)” and “**ZAKEKE\_type”**.

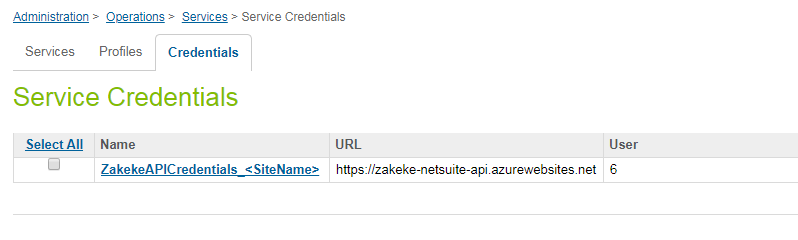
The meaning of the first field is to define which products can be customized into the storefront by the final customers. The second explains which plugin to use: “customizer” or “configurator”.

In the Salesforce Commerce Cloud Platform each API call to Zakeke platform is wrapped into the Salesforce Service framework that manages the monitoring and logging functionality.

The Zakeke extension includes the following services, profiles and credentials:







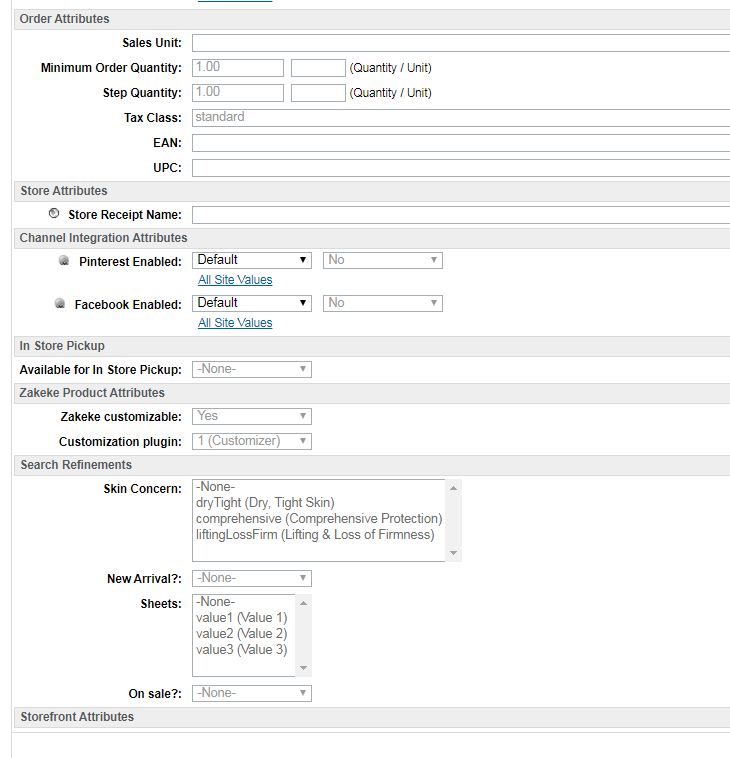
## Firewall Requirements

No specific Firewall requirements.

# Testing

After installation and configuration of CommerceCloud environment and Zakeke platform, the client can test the extension this way.

1. Import a new product using Zakeke platform. You can find instructions at the following url: <https://zakeke.zendesk.com/hc/en-us?customizer>
2. Go to business manager. Go to Merchant tools->Product and search for the just configured product.
3. If the installation is correct the product should have the field “Zakeke customizable = TRUE” and the field “Customization plugin=<chosen plugin>”as you can see in the image below.



1. If you go to the web store and search the selected product, in the PDP (product detail page), a new button “Customize” should appear after the selection of variation options.

# Operations, Maintenance

## Data Storage

Zakeke cartridge extends Salesforce Commerce Cloud system objects to store related Zakeke data.

Following system objects are extended:

* Product
  + [ZAKEKE\_is\_customizable](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=d53034a5cb9e03cce877d4b4c6&AttributeDefinitionUUID=fec75094cd6274e35ac0262a5a&csrf_token=uVqpBOEngbgCLNE4cVm5qJc4sKbq1ZN6axpgDZrwuNhKYuTMj47EPKlnvBlliN7yJ5X_9Er_oJJBAS__SsHTrhT2t_pMYvU6FygAOucuOHMWSgWeSyHw9kHt-xx0XiyERU-xqgGuZ4f7AL1lPu0gE16EqZDa6lWSYwmtuhugFJHecT3BtI8=)
  + ZAKEKE\_type
* SitePreferences
  + [ZAKEKE\_ApiCredential](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=28084049df074ef6092558724f&AttributeDefinitionUUID=5f3b1287e660b28bdc2d4df3d9&csrf_token=w-kI9kdSiqTXkCgcivpZVGV2Qqtl_KccHmDOM-yeumuMBcumcS_x8ZD5rBPsfNbAnNew8rH9bqO2GJAXeTSpsoRrjpYBYZR25-tAw1JZscZ_7ZK6dOHG4zSO-50Jz7-oD9bvqoCl6QNtdAXHD1gJhuquBmvDf_ffUAIDaedbn94fThiG9tQ=)
  + [ZAKEKE\_Email\_From](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=28084049df074ef6092558724f&AttributeDefinitionUUID=da2d0d3fa7480c2b493522df8f&csrf_token=w-kI9kdSiqTXkCgcivpZVGV2Qqtl_KccHmDOM-yeumuMBcumcS_x8ZD5rBPsfNbAnNew8rH9bqO2GJAXeTSpsoRrjpYBYZR25-tAw1JZscZ_7ZK6dOHG4zSO-50Jz7-oD9bvqoCl6QNtdAXHD1gJhuquBmvDf_ffUAIDaedbn94fThiG9tQ=)
  + [ZAKEKE\_Email\_To](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=28084049df074ef6092558724f&AttributeDefinitionUUID=0bea75ac1b794f9b986f580d5d&csrf_token=w-kI9kdSiqTXkCgcivpZVGV2Qqtl_KccHmDOM-yeumuMBcumcS_x8ZD5rBPsfNbAnNew8rH9bqO2GJAXeTSpsoRrjpYBYZR25-tAw1JZscZ_7ZK6dOHG4zSO-50Jz7-oD9bvqoCl6QNtdAXHD1gJhuquBmvDf_ffUAIDaedbn94fThiG9tQ=)
  + [ZAKEKE\_Enabled](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=28084049df074ef6092558724f&AttributeDefinitionUUID=4c789628a6f9394cfa2228cf01&csrf_token=w-kI9kdSiqTXkCgcivpZVGV2Qqtl_KccHmDOM-yeumuMBcumcS_x8ZD5rBPsfNbAnNew8rH9bqO2GJAXeTSpsoRrjpYBYZR25-tAw1JZscZ_7ZK6dOHG4zSO-50Jz7-oD9bvqoCl6QNtdAXHD1gJhuquBmvDf_ffUAIDaedbn94fThiG9tQ=)
  + [ZAKEKE\_Url](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=28084049df074ef6092558724f&AttributeDefinitionUUID=d1505c2aafb17f514da2dcb892&csrf_token=w-kI9kdSiqTXkCgcivpZVGV2Qqtl_KccHmDOM-yeumuMBcumcS_x8ZD5rBPsfNbAnNew8rH9bqO2GJAXeTSpsoRrjpYBYZR25-tAw1JZscZ_7ZK6dOHG4zSO-50Jz7-oD9bvqoCl6QNtdAXHD1gJhuquBmvDf_ffUAIDaedbn94fThiG9tQ=)
* ProductLineItem
  + [ZAKEKE\_Composition](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=e17ef98e9a9d7e96e6f6ca774e&AttributeDefinitionUUID=58b6298fcddd1e678b2130ea0b&csrf_token=8YEFio0fdtQXIpVqrr8rbOb4LTmGmGW1N5cNu5t34M-ojYy1D8C8smsCh-rdgihcW-qTQ8YSr5NxDnSxw3LlIRbv9YbBKy7a99o_04e9l_GFcXBvBOSXGEIr9bgJYmERT5aR1k3O-K2VtsY6uLejCrNLwYc2naLaWb5gu2glc-mG4KjXHVo=)
  + [ZAKEKE\_CompositionID](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=e17ef98e9a9d7e96e6f6ca774e&AttributeDefinitionUUID=5b69cd9fa1abaf114aa487008c&csrf_token=8YEFio0fdtQXIpVqrr8rbOb4LTmGmGW1N5cNu5t34M-ojYy1D8C8smsCh-rdgihcW-qTQ8YSr5NxDnSxw3LlIRbv9YbBKy7a99o_04e9l_GFcXBvBOSXGEIr9bgJYmERT5aR1k3O-K2VtsY6uLejCrNLwYc2naLaWb5gu2glc-mG4KjXHVo=)
  + [ZAKEKE\_CompositionPreview](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=e17ef98e9a9d7e96e6f6ca774e&AttributeDefinitionUUID=20d6c95db025cf68a705efe76b&csrf_token=8YEFio0fdtQXIpVqrr8rbOb4LTmGmGW1N5cNu5t34M-ojYy1D8C8smsCh-rdgihcW-qTQ8YSr5NxDnSxw3LlIRbv9YbBKy7a99o_04e9l_GFcXBvBOSXGEIr9bgJYmERT5aR1k3O-K2VtsY6uLejCrNLwYc2naLaWb5gu2glc-mG4KjXHVo=)
  + [ZAKEKE\_CompositionPrice](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=e17ef98e9a9d7e96e6f6ca774e&AttributeDefinitionUUID=fef9979c88e28eb10a17389e2b&csrf_token=8YEFio0fdtQXIpVqrr8rbOb4LTmGmGW1N5cNu5t34M-ojYy1D8C8smsCh-rdgihcW-qTQ8YSr5NxDnSxw3LlIRbv9YbBKy7a99o_04e9l_GFcXBvBOSXGEIr9bgJYmERT5aR1k3O-K2VtsY6uLejCrNLwYc2naLaWb5gu2glc-mG4KjXHVo=)
  + [ZAKEKE\_CompositionPriceNoTaxes](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=e17ef98e9a9d7e96e6f6ca774e&AttributeDefinitionUUID=dce9339db09b341070675aef12&csrf_token=edUdBlogFC9Y_4Xtrn6pUYsRnqW0Px_OzljrcO5QxGWCPboLzBxlWMRUODJpsaHAgLaQM8PrP0l5vYJ9nTTADTGAB0MVyOmk7ziREa4i1ofaER0RXqK34nt9wSBCVCBZs4G0kx_CXOOw68HJF2N7eOtbsOQJToiElcgRIVHAGw86kYI1i_w=)
  + [ZAKEKE\_Control](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=e17ef98e9a9d7e96e6f6ca774e&AttributeDefinitionUUID=a203fd58043f0411e08fa25931&csrf_token=edUdBlogFC9Y_4Xtrn6pUYsRnqW0Px_OzljrcO5QxGWCPboLzBxlWMRUODJpsaHAgLaQM8PrP0l5vYJ9nTTADTGAB0MVyOmk7ziREa4i1ofaER0RXqK34nt9wSBCVCBZs4G0kx_CXOOw68HJF2N7eOtbsOQJToiElcgRIVHAGw86kYI1i_w=)
  + [ZAKEKE\_CustomizationType](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=e17ef98e9a9d7e96e6f6ca774e&AttributeDefinitionUUID=00bdb74a815e452b42eb4bf112&csrf_token=edUdBlogFC9Y_4Xtrn6pUYsRnqW0Px_OzljrcO5QxGWCPboLzBxlWMRUODJpsaHAgLaQM8PrP0l5vYJ9nTTADTGAB0MVyOmk7ziREa4i1ofaER0RXqK34nt9wSBCVCBZs4G0kx_CXOOw68HJF2N7eOtbsOQJToiElcgRIVHAGw86kYI1i_w=)

The following object is a new custom object

* [zakOrderNotifications](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewCustomObjectType-Edit?ObjectTypeDefinitionUUID=afe4721d552d6c3d056e073cfe&csrf_token=6Eyek9VqkAGPzMXfM7ASG-CY-WnlTe2MAKInmRkYxYmZqQE81GnmsbwQVz_JDJpa0W0yBlzTKmCZP0UwqQk-y5gD0RsqpylOAHU5hXFUb-Ix0n2IcMdPpPCqBSDoTbI7A8Hi9oUoNphvskMeqBhFkEcPtFHq9-Rmxsnp66--SHTL3UDWWE4=)
  + [OrderID](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=afe4721d552d6c3d056e073cfe&AttributeDefinitionUUID=629126efb44433e5300344c7fa&csrf_token=0jByhRy7nswbifCeIyzUwq6uzI_Et84rLI_lIaHGB3yWyeylx4u4ON9tgTvpqexGvcXym0WsAZeh1NbPNrZfnntHE-SXT-HN6hlrqNq61fGQXbgNmHe6m7ZSS2kkkBFpTKvSou_dWcJyH21Ju2h3ChDAwHi7YZJGbq40l0JtA53D_kKPENI=)
  + [Message](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=afe4721d552d6c3d056e073cfe&AttributeDefinitionUUID=d1a65bb151b1eaabce99822d33&csrf_token=0jByhRy7nswbifCeIyzUwq6uzI_Et84rLI_lIaHGB3yWyeylx4u4ON9tgTvpqexGvcXym0WsAZeh1NbPNrZfnntHE-SXT-HN6hlrqNq61fGQXbgNmHe6m7ZSS2kkkBFpTKvSou_dWcJyH21Ju2h3ChDAwHi7YZJGbq40l0JtA53D_kKPENI=)
  + [Status](https://zakeke01-tech-prtnr-eu04-dw.demandware.net/on/demandware.store/Sites-Site/default/ViewObjectTypeAttribute-Start?ObjectTypeDefinitionUUID=afe4721d552d6c3d056e073cfe&AttributeDefinitionUUID=5f1ca560e95466974ac5162514&csrf_token=0jByhRy7nswbifCeIyzUwq6uzI_Et84rLI_lIaHGB3yWyeylx4u4ON9tgTvpqexGvcXym0WsAZeh1NbPNrZfnntHE-SXT-HN6hlrqNq61fGQXbgNmHe6m7ZSS2kkkBFpTKvSou_dWcJyH21Ju2h3ChDAwHi7YZJGbq40l0JtA53D_kKPENI=)

The zakOrderNotifications is created during import of metadata with a **retention of 120 days**. After this period the rows of the zakOrderNotifications table are deleted by the system. For this reason, it’s necessary for the merchant to assure that each notification record is successfully closed (OK state) before the end of the retention period.

## Availability

The Zakeke extension guarantees an uptime of almost 100%. However, in case the system does not respond, customers will not be able to use Zakeke 3d configurator. In this case ask Zakeke tech support for help by email.

Customers are able to exit the plugin page at all times coming back to the PDP page. In the cart page the client can delete a customized line item if he decides not to go through with the order.

## Support

Email: help@zakeke.com

Business Hours (Time Zone): GMT+1

# User Guide

## Roles, Responsibilities

Zakeke introduce no roles with this integration.

Zakeke will provide an access account to be used in the Zakeke platform.

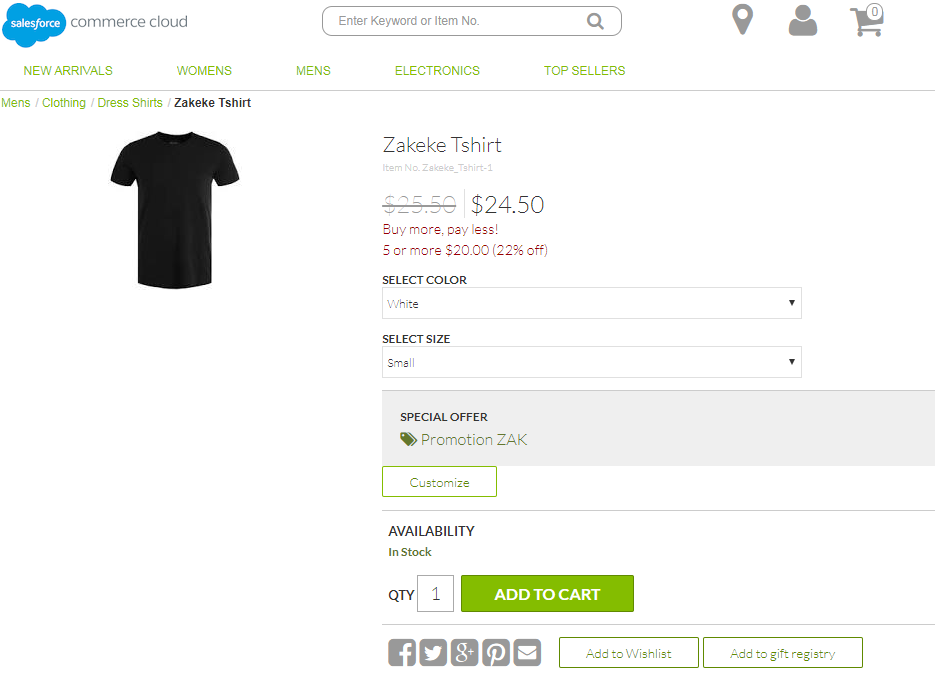
## Business Manager

Zakeke extension involves a module for Business Manager that manages the notification process through a custom job. For the detail of the process see the storefront functionality.

No business manager module is present for the frontend.

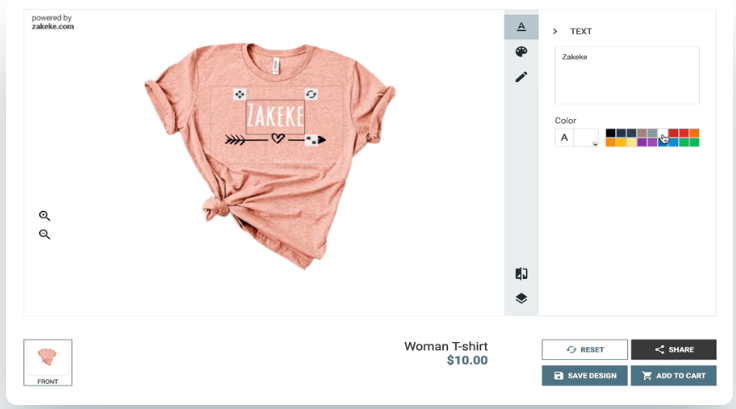
## Storefront Functionality

1. Go to the web store.
2. Search a customizable product (standard or master product) and go to its detail page.
3. Choose the options to define the variation of the item
4. A new button “Customize” should appear after the options variation choices. See the image below.

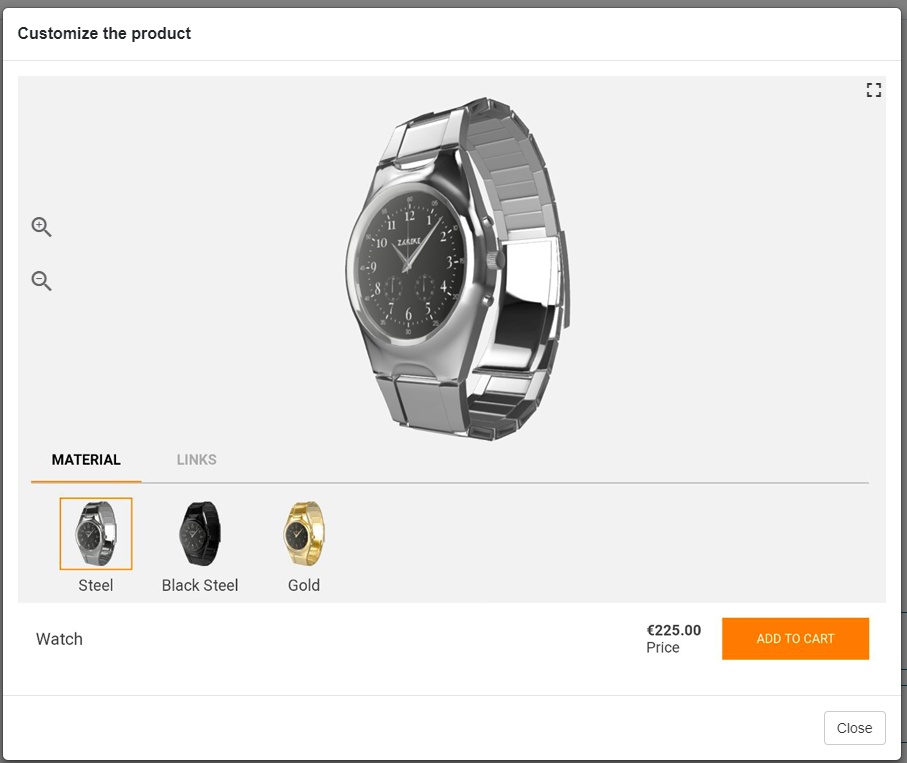


1. Clicking the “Customize button”, if you used the “**Customizer**” as type of customization for the selected product, you are redirected to a custom Zakeke page in which you can use Zakeke <iframe> to customize the product in a 3d environment.

See the image below.

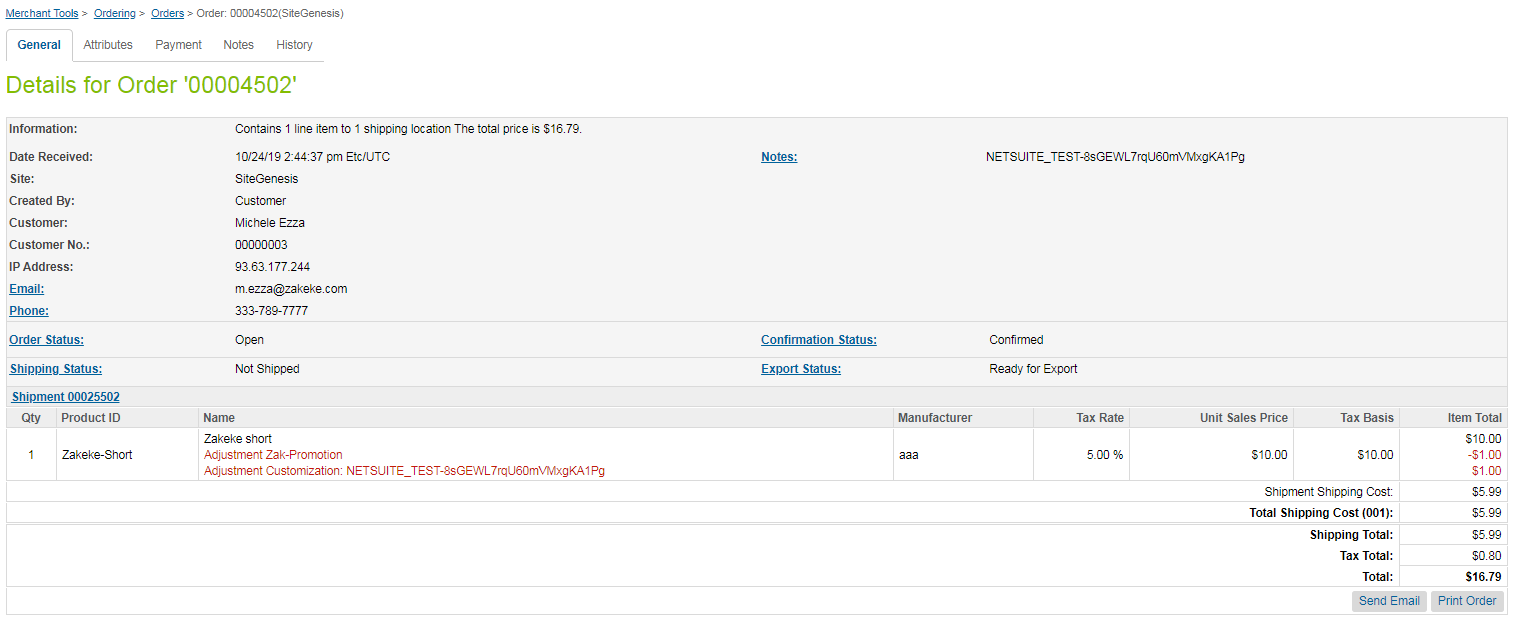


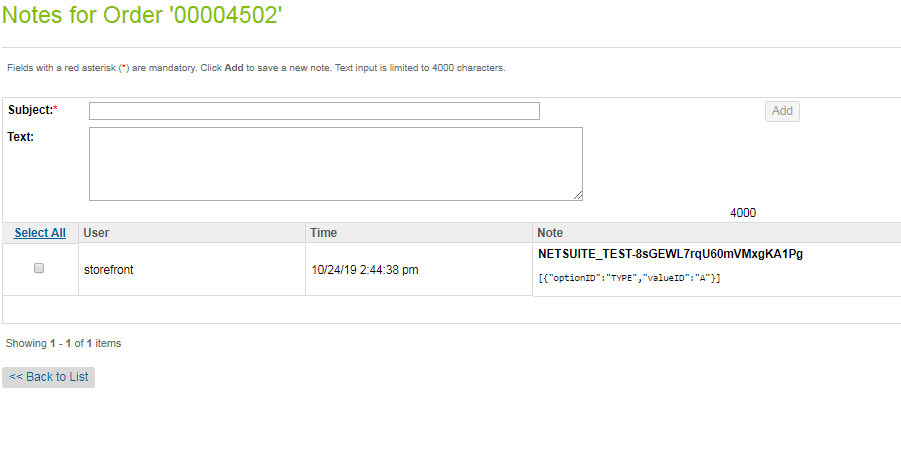
1. Clicking the “Customize button”, if you used the “**Configurator“** as type of customization for the selected product, the configurator plugin page appears. See the image below.



1. At the end, you can either close the page or accept the customization clicking the “Add to cart” button.
2. The “Add to cart” action adds the selected item to the cart redirecting the customer to the cart page.
3. The workflow and the steps of the purchase process remain unchanged for both checkout and payment phase
4. At the end of the order if you go to Business manager -> Merchant tools-> Orders, you can see the new order record. There are some additional fields that are added by Zakeke extension: **customizationID**, **price** **adjustment** (if present) and **customization detail** in the “Notes” section of the order.

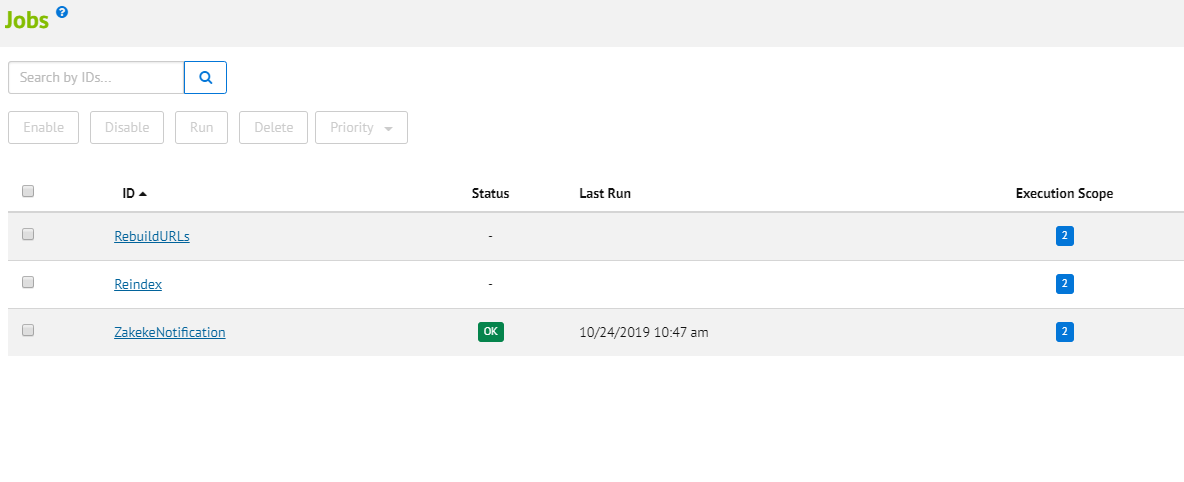
See the images below.





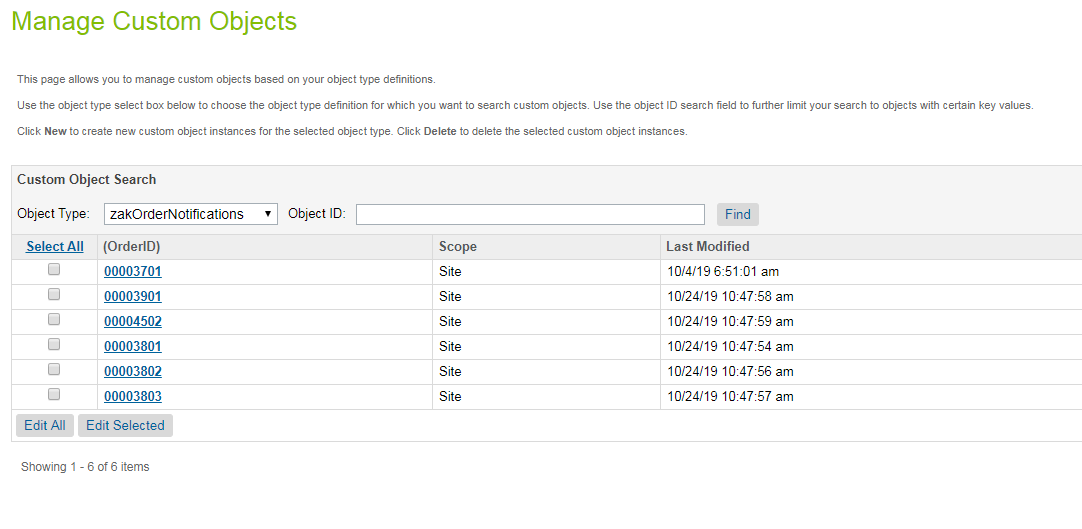
1. The last phase of the customization process is the notification of the order to Zakeke platform. A custom Zakeke job called *ZakekeNotification* is present in the Administrator-> Operations -> jobs application.

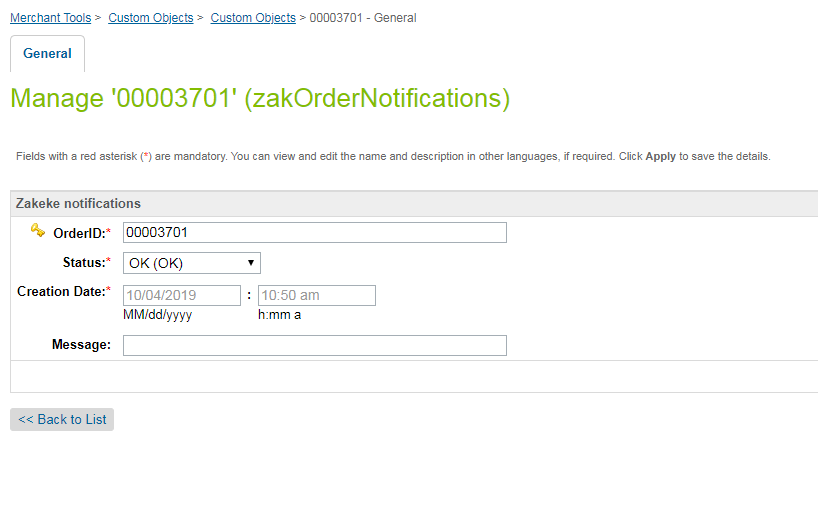
See the image below.



The job is configured to run every 30 min in a recurring manner. The results of the job is saved in the custom object *zakOrderNotification.*

To access the custom table go to business manager -> Merchant Tools -> Custom Objects -> Custom Object Editor. In the page “Manage custom object” choose *zakOrderNotification as* Object type and click Find button. To see the detail of the row you can click the orderID link. See the images below.





The record can have different status. Status can have three different values:

* **ADDED** = the row has just been added by the web process but not worked yet by the job
* **OK** = the row has been successfully worked by the job
* **ERROR** = the row has been worked by the job but with error. The message field contains the detail of the error

# Known Issues

No issues known at the moment.

# Release History

Release history of the component. This document is part of the component and does not have its own release/version number.

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 20.1.0 | 2020-01-31 | First release |
| 20.1.1 | 2020-02-04 | Parameters CustomerId/visitorCode change |
| 20.1.2 | 2020-10-23 | Bug fix |
|  |  |  |