

Payment Gateway Connector - Help Guide

About

Payment Gateway Connector (referred as PG Connector) lets you integrate third-party payment gateways that are not directly available with Zoho Commerce.

TABLE OF CONTENTS

- DRAFT
- [What is a PG Connector?](#)
 - [Lifecycle of a PG Connector](#)
 - [Working model of a PG Connector](#)
 - [Creating PG Connector](#)
 - [Creating a transform face file](#)
 - [Upload](#)
 - [Configure PG Connector](#)
 - [Private sharing](#)
 - [Publishing in Zoho marketplace](#)
 - [Updating PG Connector](#)
 - [Example](#)
 - [Endpoint configuration in Payment Gateways](#)

• What is a PG Connector?

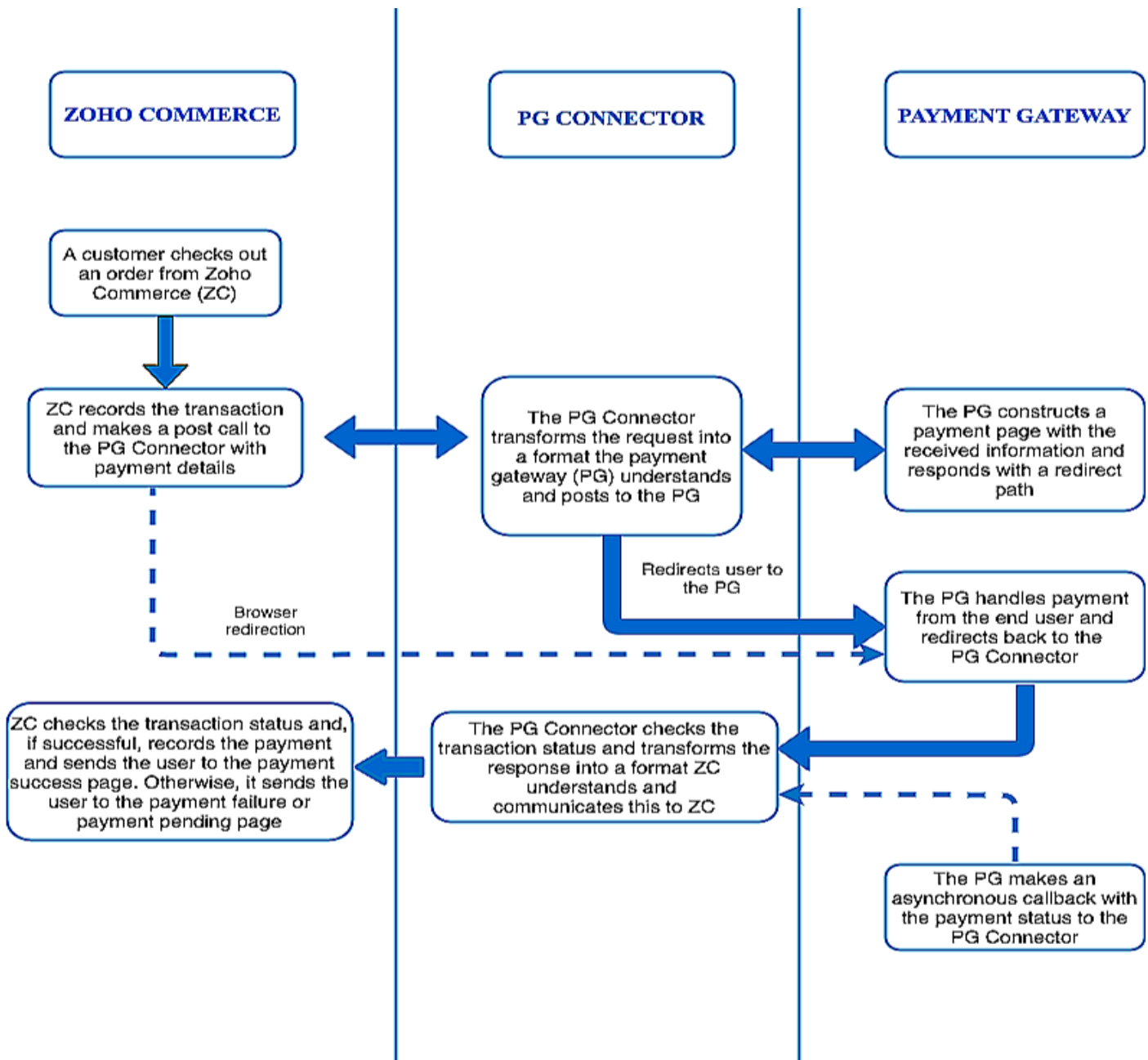
A PG Connector acts as an intermediary between Zoho Commerce and payment gateways. They allow store owners (hereafter referred to as merchants) to receive payments from their customers via payment gateways that are not directly integrated with Zoho Commerce.

A PG Connector processes an incoming payment request from Zoho Commerce (with the help of transform instructions provided to it) into a format that is understandable for a payment gateway. The Payment Gateway then processes the request and sends an asynchronous payment status notification to the PG Connector. The PG Connector processes the notification from the Payment Gateway into a format understandable for Zoho Commerce using the transform instructions.

• Lifecycle of a PG Connector



• Working model of a PG Connector

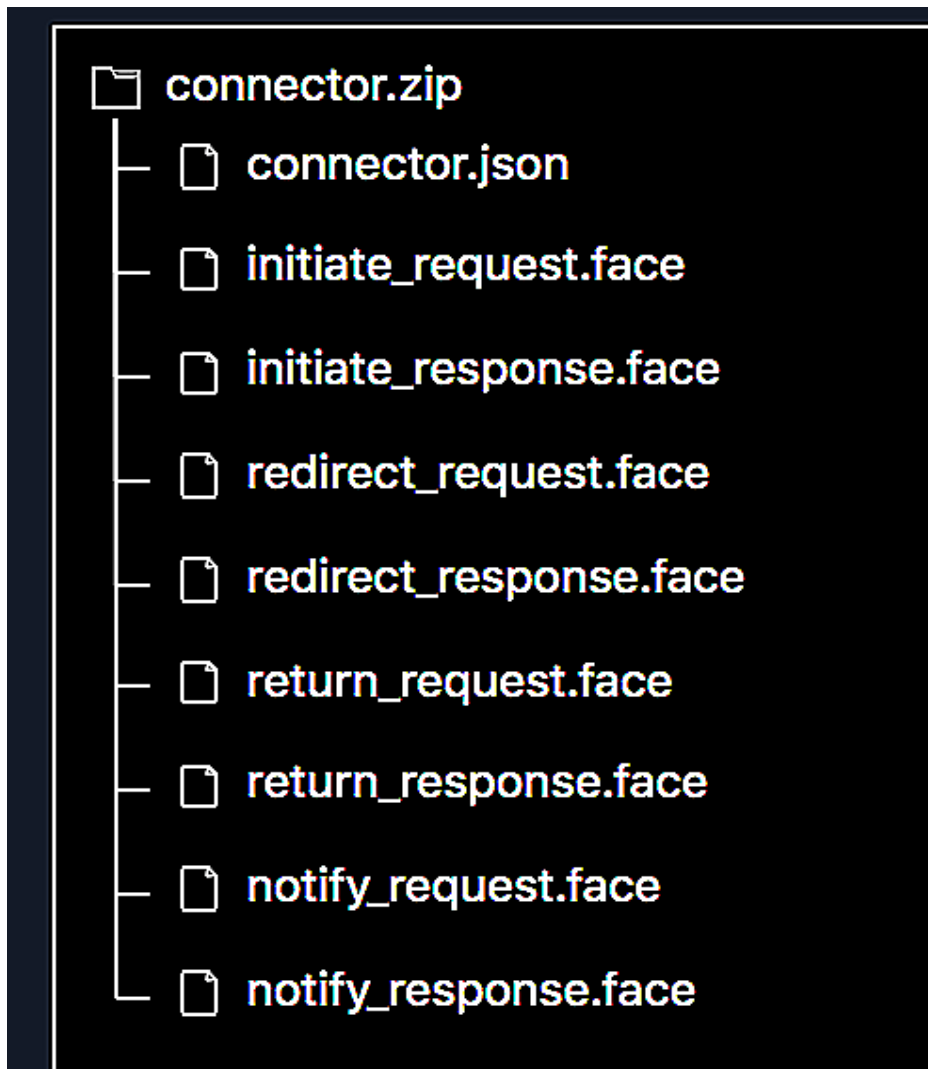


• Creating PG Connector

To create a PG connector, you need to upload a connector ZIP containing all the required details to create a PG connector. A connector zip should contain the following

- **Connector json** (json file having connector properties/configurations)
- **Transform files** (one or more files containing transform instructions)

This is the file structure that needs to be maintained in the connector ZIP:



With all the PG Connector properties and transform files (which have been discussed as a separate step later in this doc), you will be able to create a PG Connector in Zoho Commerce.

➤ **Connector JSON**

This Connector json holds the payment gateway's display properties like gateway name, description , endpoints etc. Lets look at them in detail.

A sample connector json (written for PayGate - a leading payment gateway for South Africa)

would be like :

```
1 {
2   "gateway_name" : "PayGate",
3   "description" : "This is a test gateway - PayGate",
4   "type":"server_redirection",
5   "endpoints" : {
6     "/notify":{
7       "request_transform":"notify_request",
8       "response_transform":"notify_response",
9       "type":"notify"
10    },
11   "/initiate":{
12     "request_transform":"initiate_request",
13     "response_transform":"initiate_response",
14     "type":"initiate_payment"
15   },
16   "/redirect":{
17     "request_transform":"redirect_request",
18     "response_transform":"redirect_response",
19     "type":"redirect"
20   },
21   "/return":{
22     "request_transform":"return_request",
23     "response_transform":"return_response",
24     "type":"return"
25   },
26   "/failure":{
27     "request_transform":"failure_request",
28     "response_transform":"failure_response",
29     "type":"failure"
30   }
31 },
32 "instance_fields":[
33   {
34     "field_label":"Enter your Paygate ID",
35     "field_name":"paygateID",
36     "field_type":"number",
37     "field_max_length":"25"
38   },
39   {
```

```

40  "field_label":"Enter your password",
41  "field_name":"password",
42  "field_type":"alphanumeric",
43  "field_min_length":"5",
44  "field_max_length":"25"
45  }
46 ],
47 "shared_config":{
48  "supported_currencies" : ["ZAR","USD","AED"],
49  "supported_payment_modes" : ["credit_card", "debit_card"]
50 },
51 "error_codes" : {
52  "101":"106",
53  "102":"107",
54 },
55 }
56

```

A summary of all the required properties and corresponding keys are as follows:

Key	Description	Allowed Values
gateway_name*	Name of the payment gateway	String
description*	Description of payment gateway	String/Paragraph
type*	Type with which "initiate_payment" endpoint has to be communicated	server_redirection, browser_redirection
endpoints*	A json consisting of various endpoint urls and their corresponding properties	json object
instance_fields*	A json array consisting of various field names	json array
shared_config*	A json array consisting of shared configurations of connector	json object
error_codes*	Mapping of error codes between Payment gateways and Zoho Commerce	json object, can also be an empty json

endpoints

Endpoints are simply an unique URI, which acts as one end of communication to the server. You can define the endpoint URI as you like while adhering to the pre-defined set of rules. It can only have alpha-numeric characters, hyphen(-) and underscore(_) with a "/" in the prefix. An endpoint can be defined as a key-value pair json with URI as its key and corresponding properties object as its value. Every endpoint will have a request and a response transform to be processed. Either the request or the response transform can be empty for an endpoint but both cannot be empty at the same time.

To decide the nature of the endpoint, we have certain pre-defined fields which help us identify its type and properties. Let's have a look at them:

A sample endpoint json would be:

```
1 "endpoints" : {
2   "/notify":{
3     "request_transform":"notify_request",
4     "response_transform":"notify_response",
5     "type":"notify"
6   }
7 }
```

A summary of all the endpoint properties and the allowed values are listed in the below table:

Key	Type	Allowed Values
type	String	initiate_payment, redirection, return, failure, notify
request_transform	String	alpha numeric, underscore or _, hyphen or -
response_transform	String	alpha numeric, underscore or _, hyphen or -

- **type**

This field is used to know the type of endpoint that has been defined. The endpoint types have

been broadly classified as following:

- **initiate_payment :**

This type of endpoint acts as a communication point between Zoho Commerce and a PG Connector during payment initiation. Whenever a buyer checks out an order from Zoho Commerce, it will be processed, and the payment details will be passed to the PG Connector using this type endpoint type. Upon receiving the payment details, the PG Connector will start transforming the request from Zoho Commerce into a Payment Gateway-understandable format using the request transform instructions provided to PG Connector. The transaction will be recorded in this stage and details will be provided to you (to transforms) in further stages of transaction. The communication will happen with Payment Gateway as instructed and response can be stored for further processing. The control will then pass on to the next endpoint as specified in response transform.

- **redirect :**

In this endpoint, buyers will be redirected to Payment Gateway's UI with the help of the transform instructions provided to PG Connector. The redirection can happen by any means. For eg., the redirection can be a 302 redirect in server or a form submit with necessary details from client etc., You need to make use of the request and response transforms accordingly to achieve the requirement which will then be processed by PG Connector.

- **return :**

This type of endpoint acts as a communication point between the Payment Gateway and the PG Connector, denoting an asynchronous return from the Payment Gateway to the PG Connector. In this, the Payment Gateway will make an async callback to the PG Connector without actually communicating the payment status back to the PG Connector. In turn, the PG Connector will transform this request into a format understandable for Zoho Commerce with the help of transform instructions, and the buyer gets redirected to the payment pending page.

- **notify :**

Most of the payment gateways notify the payment status of the transaction through a separate payment webhook. This type of endpoint will act as a communication point to Payment Gateways for sending webhooks. These urls can be configured in payment gateways to enable communication between the Payment Gateway and PG Connector. The PG Connector receives this request, transforms it, and posts it to Zoho Commerce. If the status is a "success", an order will be created. Otherwise the transaction will be recorded as a failed payment.

- **failure :**

This type of endpoint will act as a communication point to denote the payment failure at any of the stages involved in this process. The PG Connector will communicate the payment failure on encountering failures at any of the stages with Zoho Commerce.

- **request_transform**

This field will contain the name of the transform file which is associated with the corresponding endpoint. The transform file will contain all the transform instructions for request that need to be processed when the server encounters this endpoint. Writing transform instructions for an endpoint will be discussed in a later section of this document.

- **response_transform**

This field will contain the name of the transform file which is associated with the corresponding endpoint. The transform file will contain all the transform instructions for response that need to be processed when the server encounters this endpoint. Writing transform instructions for an endpoint will be discussed in a later section of this document.

instance_fields

These field names are specific for a particular merchant who installs and uses the connector. These include configuration fields like user name, password, api-key, etc. Having all these, Zoho Commerce can prompt the fields to the merchants and get corresponding values from them during the Payment Gateway installation in their store. Along with the `field_name`, its properties like `field_label`, `field_type` etc., can be given as an input which will be helpful in validation during configuration. A sample instance field json array would be:

```
1 {
2   "field_label":"Enter your password",
3   "field_name":"password",
4   "field_type":"alphanumeric",
5   "field_min_length":"5",
6   "field_max_length":"25"
7 }
```

shared_config

A PG Connector, once written and published in the market place, will be made available for other merchants to install and use. This shared config denotes any specific configuration of the connector that is common for all the merchants who install this PG Connector. For example, configurations like currencies and payment modes supported by a PG Connector are common across

all installations.

```
1 "shared_config":{
2   "supported_currencies" : ["ZAR","USD","AED"],
3   "supported_payment_modes" : ["credit_card", "debit_card"]
4 },
```

error_codes

Error codes are a pre-defined set of failure codes that will be given along with the response when a request has been made to a platform. To enable the communication of the error codes between Zoho Commerce and Payment Gateways, a json containing the mapping of error codes between Zoho Commerce and Payment Gateways should be given in the configuration json while creating a PG Connector. An example of json can be:

```
1 {
2   "101" : "106"
3 }
```

Here, the key should be the error code of the Payment Gateway, and the value should be the error code of Zoho Commerce. Both error codes in the above case convey the meaning **"The card is not valid/invalid"** with respect to their platforms.

Below are the error codes used in Zoho Commerce for various payment failure reasons.

Payment Failure Error Codes

Code	Error message displayed to buyer
100	Error occurred while processing transaction at the gateway
101	An error occurred while processing your card. Try again after some time

102	Gateway credentials are not valid. Please provide valid credentials
103	The reference ID used in this transaction is invalid
105	The card number provided is invalid
106	The card is invalid
107	The expiry date provided for the card is invalid
108	The card provided has expired
111	The payment has been declined
112	The transaction has been declined
113	The address for the card is invalid
114	The CVV provided for the card is invalid
115	The address or CVV provided for the card is invalid
116	The card type provided is not supported
118	The currency used for this transaction is not supported
119	Duplicate transaction. An identical transaction is being processed
121	No funding source to complete the transaction
124	This transaction has been identified as a fraud and has been declined
125	The API information provided is invalid
128	Invalid merchant
129	Invalid payment card details
130	Invalid values are passed in the transaction

132	The amount to be charged is less than the minimum amount supported by the payment gateway
135	Invalid card details
136	Card has been declined
141	Due to too many failed login attempts, your user account has been temporarily disabled. Please try again after some time
142	There was a connection error while validating your gateway credentials. Please try again after some time
143	Please ensure that the field values are valid and non-empty

► Transform files DRAFT

These are face files (with the extension .face) (Refer: [Face Language](#)) that contain the transform instructions to convert the request/response from one format to another. A transform file must have a name that is the same as the one mentioned in the corresponding endpoint json, which is restricted to contain only alpha numeric, hyphens, and underscore characters.

A transform file will generally have a request, a response mapping, or any combination of the two that facilitates or eases the communication between two different platforms. (This is discussed in detail in next section).

For example, let's say Zoho Commerce gives the total order value in the node **"amount"**. But for the same thing, the value can be expected in the node **"price"** by the Payment Gateway. This conversion of the node in the request will happen if the nodes have been mapped properly in the transform file. A PG Connector does this transformation by acting as an intermediary between Zoho Commerce and the Payment Gateway with the help of transform instructions provided by you.

NOTE: While writing transform files, make sure that the control is directly passed from the PG Connector to the Payment Gateway—and vice versa—and not to any intermittent servers. If this requirement is not met, your connector will not be allowed to publish in the marketplace.

• Creating a transform face file

Let's see how a sample transform file can be created. Lets create a sample transform file to register a transaction in the Payment Gateway. Before that, let's see how the payment request will look and how a payment notification response should be with respect to Zoho Commerce.

A sample payment request's parameter values from Zoho Commerce are listed below:

Parameters	Values
shipping_charges	50.000
customer_shipping_state	Germany
reference_id	4rXu2L0hsiaJkDO50evba_20_0
excluded_payment_types	
signature	37849a71d7797hisc5d3d860b49f7d6a9b98e49c4eba3d70bd8dc5e672314d4
cc_phone_number	
notification_url	https://zohosecurepay.com/checkout/n/paymentipn/100003?order_id=4rXu2L0hsiaJkDO50evba_20_0&encryptedHPID=ow1ei9c-ya567bjyz41a
discount_amt	0.0
customer_shipping_phone	09876543210
currency_code	EUR
customer_billing_zip	
customer_shipping_address1	777,STORY RD

callback_url	https://zohosecurepay.com/checkout/paymentrequest/redirect/100003/4rXu2L0hsiaJkDO50evba_20_0?encryptedHPID=ow1ei9c-ya567bjyz41a
customer_shipping_address2	
item_details	[{"quantity":1,"picture_url":"","id":87996111111104370,"title":"Pencil","unit_price":20,"currency_id":"EUR"}]
customer_billing_address1	
customer_billing_address2	
customer_billing_phone	
customer_billing_country_code	
first_name	Ann
email	aishwaryalakshmi.ss+4@zohotest.com
amount	70.00
payment_mode	
organization_language_code	en
entity_number	2-343084ab02cef468c4d65118f6172462309e2c2139316657997f71f7432322f6e89a6257752db480636280e0f950de09
customer_billing_city	
last_name	Bay

organization_name	
entity_id	87996111111104370
payment_complete_url	https://zohosecurepay.com/checkout/paymentrequest/redirect/100003/4rXu2L0hsiaJkDO50evba_20_0?encryptedHPID=ow1ei9c-ya567bjyz41a
customer_billing_state	
customer_shipping_zip	95122-2628
entity_type	20
account_id	66135125
customer_shipping_country_code	DE
organization_id	66135125
phone_number	09876543210
adjustment	0.0
customer_shipping_city	Berlin
cancel_url	https://zohosecurepay.com/checkout/paymentrequest/redirect/cancel/100003/4rXu2L0hsiaJkDO50evba_20_0

A payment notification response to Zoho Commerce should be in the following format:

Parameters	Example
amount	40000
gateway_fee (optional)	40

gateway_reference_id	916754771
currency_code	EUR
payment_mode	card
transaction_status	1 (Success) or 0(failure) or -1(In progress)
gateway_errorcode	5012 (Gateway Transaction Error code in case of payment failure)
zcm_error_code	105 (Zoho Commerce Error Code in case of payment failure)

To form a proper HTTP request to PG, you can use the following nodes available in Zoho Commerce:

Parameters	Example
url	Request URL
method	Request method. Can be GET, POST, PUT, DELETE
headers	Request headers
body	body data of the request
query_string	query string for the request

The request parameters should be sorted in alphabetical order, signed using the HMAC-SHA256 signing mechanism, and sent as "signature" in the request parameters. To sort and sign the request parameters, the PG Connector will provide a built-in method that can be used in the transform files.

There are additional built-in methods in PG Connector, using which you can additionally store / retrieve properties and transaction data too. A summary of the available built-in methods are as follows:

Functions	Response format	Usages
getConfigData(String key)	Object	can be used to retrieve saved connector configurations
store(key, value)	-	used to store run time data which might be useful in other stages of transaction

retrieve(key)	Object	can be used to retrieve stored run time transaction data
sortAndSign(algorithm, delimiter, paramJson, signKey)	String	sorts the request params in alphabetical order and generates the signature using the specified algorithm
getUrl(endpoint_uri)	String	provides fully qualified endpoint url with the given endpoint URI
getUri(endpoint_uri)	String	provides full PG Connector URI for the endpoint without storefront domain.
paymentSuccessStatus()	String	returns payment success status
paymentFailureStatus()	String	returns payment failure status
paymentPendingStatus()	String	returns payment pending status
queryStringToJson(queryString)	JSONObject	converts the given query string to a json object
transactionDetails()	JSONObject	provides current transaction details like amount, currency, reference id, etc.,
getErrorCode(errorCode)	Integer	retrieves corresponding Zoho Commerce error code for the given error code from PG using the provided error code map.
md5(value)	String	provides md5 hash for the given value
getISO3CountryCode(lang_code, ISO2CountryCode)	String	provides ISO3 (3 - letter country code) for the given language code and 2 letter country code.
paymentCompleteUrl()	String	provides payment complete url for the current transaction
paymentNotificationUrl()	String	provides payment notification (IPN) url for the current

		transaction
checkSignature(algorithm, delimiter, paramJson, signKey, signature)	Boolean	validates the signature for the given params and returns the status
extractQueryParams(urlString)	JSONObject	extracts the query params from the given url string if any.
getTimeInMillis()	Object	provides current instant epoch second
getCurrentTime(format, timezone)	Object	provides current date as string in the given format for the given time zone

With the help of these details, let's see how a transform for the payment page can be created:

1. Check the request parameters provided by Zoho Commerce and find the appropriate equivalent node facilitated in the Payment Gateway request.
2. Write the request format including url, method, etc. as guided by your Payment Gateway and map these parameters from Zoho Commerce, which will be available to you as a key named "form_data" in the object named "request".
3. Use any built-in methods, if needed, for pre-stored data or merchant-specific configurations.
4. Construct the asynchronous return url with the help of the built-in method if your payment gateway has a facility to redirect asynchronously to PG Connector after handling payments.

A sample request transform will look as follows:

```

1 {
2   "url" :
   "https://secure.paygate.co.za/payweb3/initiate.trans",
3   "method" : "POST",
4   "headers" : {
5     "content-type" : "application/x-www-form-urlencoded",
6   }
7
8   {% assign params = {}%}
9   {% assign password = connector.getConfigData("password") %}

```

```

10
11     {%
12         assign reference_id = request.form_data.reference_id,
13
14         params.PAYGATE_ID =
15         connector.getConfigData("paygateID"),
16         params.REFERENCE = reference_id,
17         params.CURRENCY = "ZAR",
18         params.AMOUNT = request.form_data.amount | multiply(100)
19         | splitFirst("."),
20         params.RETURN_URL = connector.getUrl("/return"),
21         params.TRANSACTION_DATE =
22         connector.getCurrentTime("yyyy-MM-dd HH:mm:ss","UTC"),
23         params.LOCALE =
24         request.form_data.organization_language_code | append("-") |
25         append("ZA" | lower),
26         params.COUNTRY =
27         connector.getISO3CountryCode(request.form_data.organization_lang
28         uage_code, "ZA") | upper,
29         params.EMAIL = request.form_data.email,
30         params.NOTIFY_URL = connector.getUrl("/notify")
31     %}
32     {%
33         assign value_to_checksum = params.PAYGATE_ID |
34         append(params.REFERENCE) |
35         append(params.AMOUNT) |
36         append(params.CURRENCY) |
37         append(params.RETURN_URL) |
38         append(params.TRANSACTION_DATE) |
39         append(params.LOCALE) |
40         append(params.COUNTRY) |
41         append(params.EMAIL) |
42         append(params.NOTIFY_URL) |
43         append(password),
44         params.CHECKSUM = connector.md5(value_to_checksum)
45     %}
46     {% capture post_data %}PAYGATE_ID={{ params.PAYGATE_ID
47     }}&REFERENCE={{ params.REFERENCE }}&AMOUNT={{ params.AMOUNT

```

```

    }}&CURRENCY={{ params.CURRENCY }}&RETURN_URL={{
    params.RETURN_URL}}&TRANSACTION_DATE={{ params.TRANSACTION_DATE
    }}&LOCALE={{ params.LOCALE }}&COUNTRY={{ params.COUNTRY
    }}&EMAIL={{ params.EMAIL }}&NOTIFY_URL={{
    params.NOTIFY_URL}}&CHECKSUM={{ params.CHECKSUM }}{% endcapture
    %}
42
43     "body" : "{{post_data | strip}}"
44 }

```

To send a proper HTTP response using transforms, you can use the following nodes in Zoho Commerce:

Parameters	Example
headers	Response headers
status_code	Status code of response
response_text	response string
response	response json
error_code	error code if any
error_text	error text if any

A sample response transform would be as follows:

```

1 {
2     "headers" : {
3         "content-type" : "text/html"
4     },
5     {% assign pay_request_id = connector.retrieve("pay_request_id"),
6         checksum = connector.retrieve("checksum")
7     %}
8
9     {% capture form %}
10 <!DOCTYPE html><head></head><body
    onload='document.redirect.submit()'><form name='redirect'
    action='https://secure.paygate.co.za/payweb3/process.trans'
    method='POST' ><input type='hidden' name='PAY_REQUEST_ID' value={{
    pay_request_id }}><input type='hidden' name='CHECKSUM' value={{
    checksum }}></form></body></html>

```

```

11     {% endcapture %}
12
13     "response_text" : "{{form | strip}}"
14 }

```

• Upload

Below are the steps to upload a PG Connector in Zoho Commerce:

➤ Create new extension

To upload a PG Connector to Zoho Commerce you need to create an extension in Zoho Sigma as a first step. Visit [Zoho Sigma](#). Create a new extension with the required details as shown in the screenshot below. Please select Zoho Commerce (bag icon). If you are not seeing Zoho Commerce, it needs to be enabled. To do this, please email pg-connector@zohocommerce.com.

isions
Extensions Gallery Help

New Extension

Name *

Description *

Meet the payments platform built for any business and every customer journey. A single payments platform to accept payments anywhere, on any device.

Service *

NO Upload Existing File

I agree to the terms and conditions

Save as Draft
Cancel

Help

What is Extension?

Extension is a software add-on that extends the capabilities of your service. Build your extension using any language, such as, HTML, Javascript, CSS etc. Sigma provides functions for handling server side operations. Integrate your service with any third party application and extend its scope using extensions.

Private extension

Extensions that are built for personal or specific organization purpose. These extensions are installed using hashed private link.

Public extension

Extensions that are hosted through Zoho Marketplace, a platform where extensions of all Zoho services are listed. All Zoho users can install and use the extension.

Once an extension is created you need to add the following details to the plugin-manifest.json file

Page 21

Key	Value
connector_type	commerce
is_pg_connector	true
scope	store id in which you're going to create the PG connector
location (under modules json, in widgets array. Please check the below screenshot once)	zoho.commerce.paymentgateways

DRAFT

Your plugin-manifest.json will look like this.

```

1- {
2-   "defaultLocale": "en",
3-   "whitelistedDomains": [],
4-   "service": "COMMERCE",
5-   "storage": false,
6-   "locale": [
7-     "en"
8-   ],
9-   "config": [],
10-  "cspDomains": {
11-    "connect-src": []
12-  },
13-  "connector_type": "commerce",
14-  "scope": "67139147",
15-  "is_pg_connector": true,
16-  "modules": {
17-    "widgets": [{
18-      "name": "Commerce_Extensions",
19-      "location": "zoho.commerce.paymentgateways",
20-      "url": "/app/home.html"
21-    }]
22-  }
23- }

```

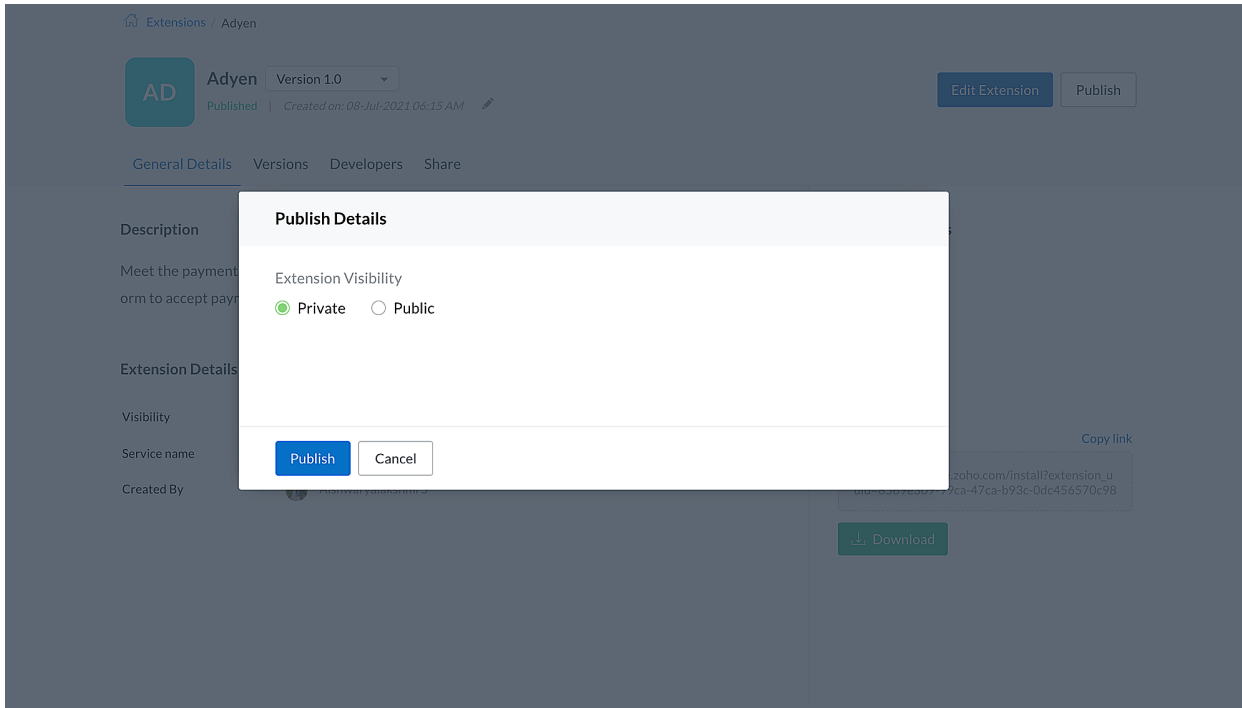
➤ **Publish extension**

You need to publish your extension to make it work in stores. Publish can be of two visibility.

- ★ Private
- ★ Public

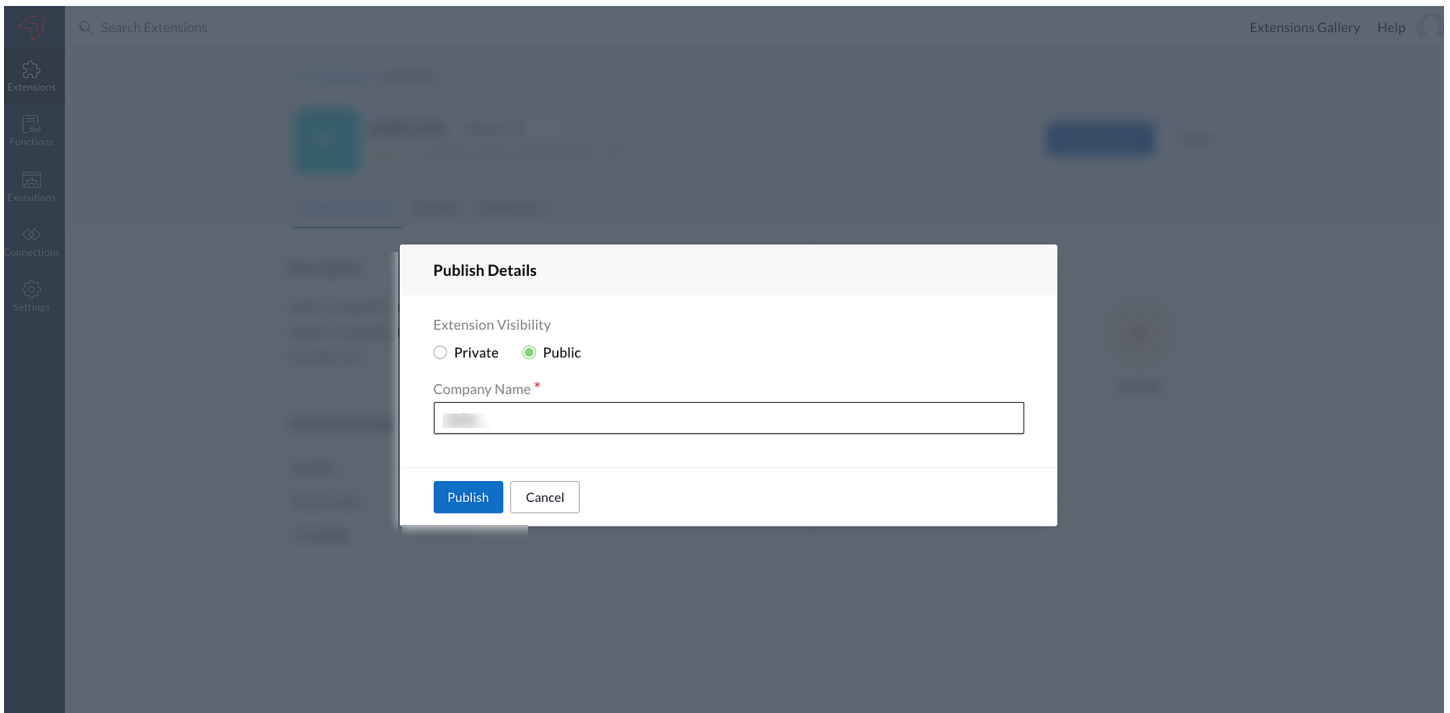
Private

If the Extension Visibility is chosen as Private, only you and invited merchants will be able to install in stores.



Public

If the Extension Visibility is chosen as Public, anyone can install this app in their stores and start receiving payments.



Note : Once an app is made private, it can't be made public and vice versa. You need to create a new extension to get the desired Extension visibility. To publish the gateway app in the marketplace, you need to get an approval from the Zoho Commerce team, for which you can write to us at pg-connector@zohocommerce.com

Search Extensions Extensions Gallery Help

[Extensions](#) / PayGate

PA

PayGate Version 1.0

Published | Created on: 22-Oct-2021 04:37 AM

Edit Extension Publish

[General Details](#) [Version](#) [Developers](#)

Description Last Modified on: 22-Oct-2021 04:37 AM

Testing PayGate which is a south african based payment gateway that supports ZAR currency

Extension Details

Visibility Private

Service name Zoho Commerce

Created By

Installation Stats

0

USERS

Install URL

[Copy link](#)

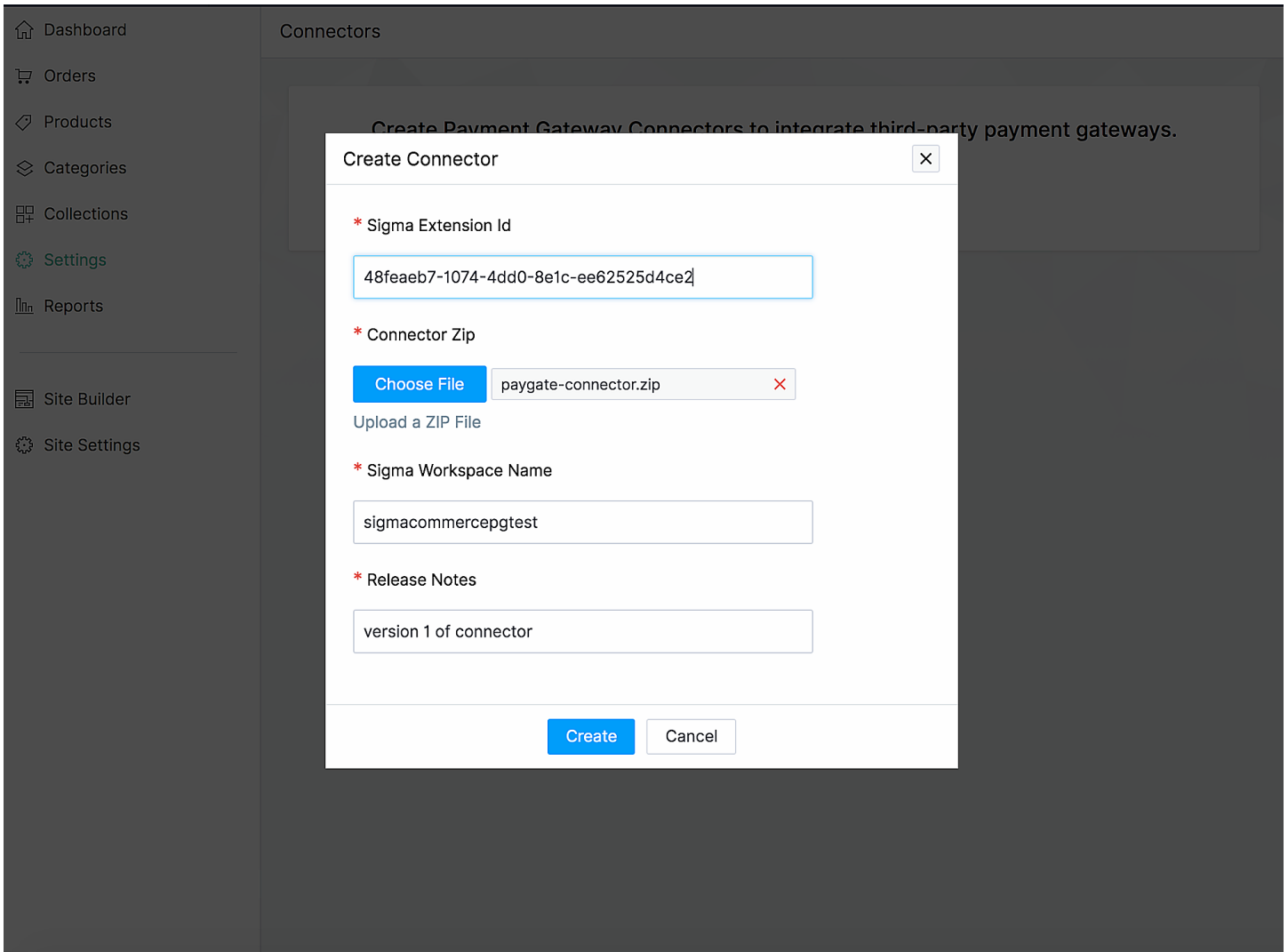
https://commerce.zoho.com/install?extension_uuid=48fe9eb7-1074-4dd0-8e1c-ee62525d4ce2&extension_version=1.0&hash=89

Download

Page 24

► Upload PG Connector

Once app is published in Sigma, you need to upload the same in Zoho Commerce. In your Zoho Commerce store, under **Settings** you can find **Connectors**. Upload connector as shown in the screenshot with the required details. If you don't find connectors under settings, then please email us at pg-connector@zohocommerce.com to enable the same.



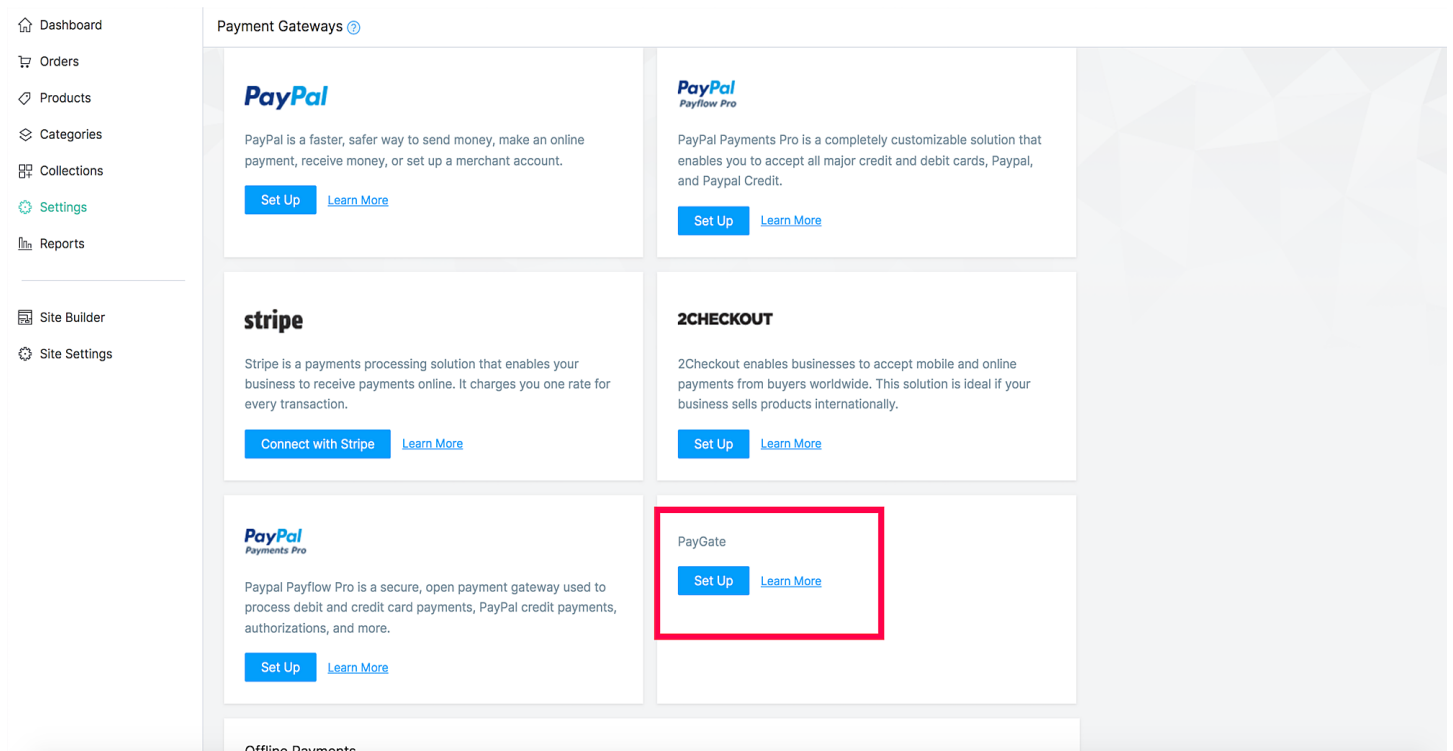
On clicking Upload in the above screen, your PG Connector will get uploaded to your store.

● Configure PG Connector

After uploading PG Connector in your store, it will now list as one of the payment gateways under **Settings --> Payment Gateways** in Zoho Commerce (as shown in the figure).

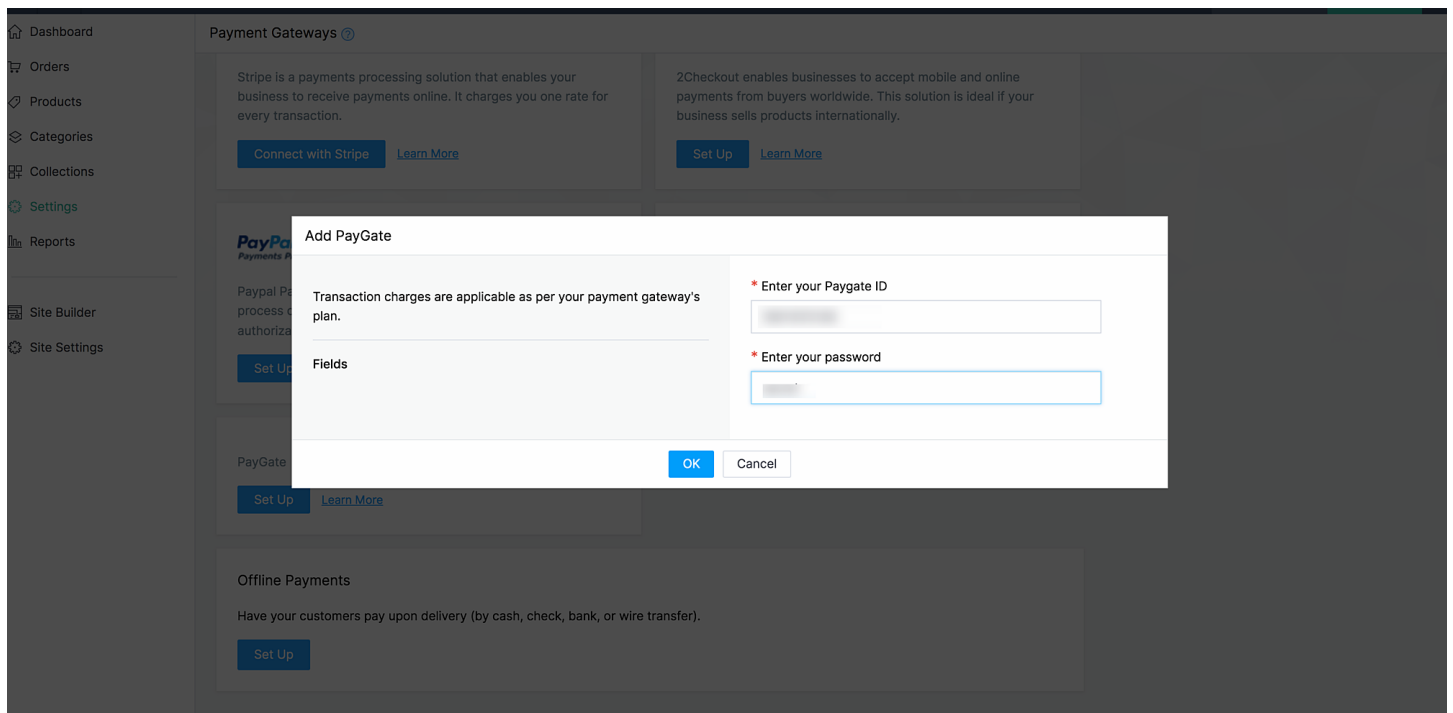
You can configure the PG Connector and cross-check the payment integration in your storefront

checkout page.

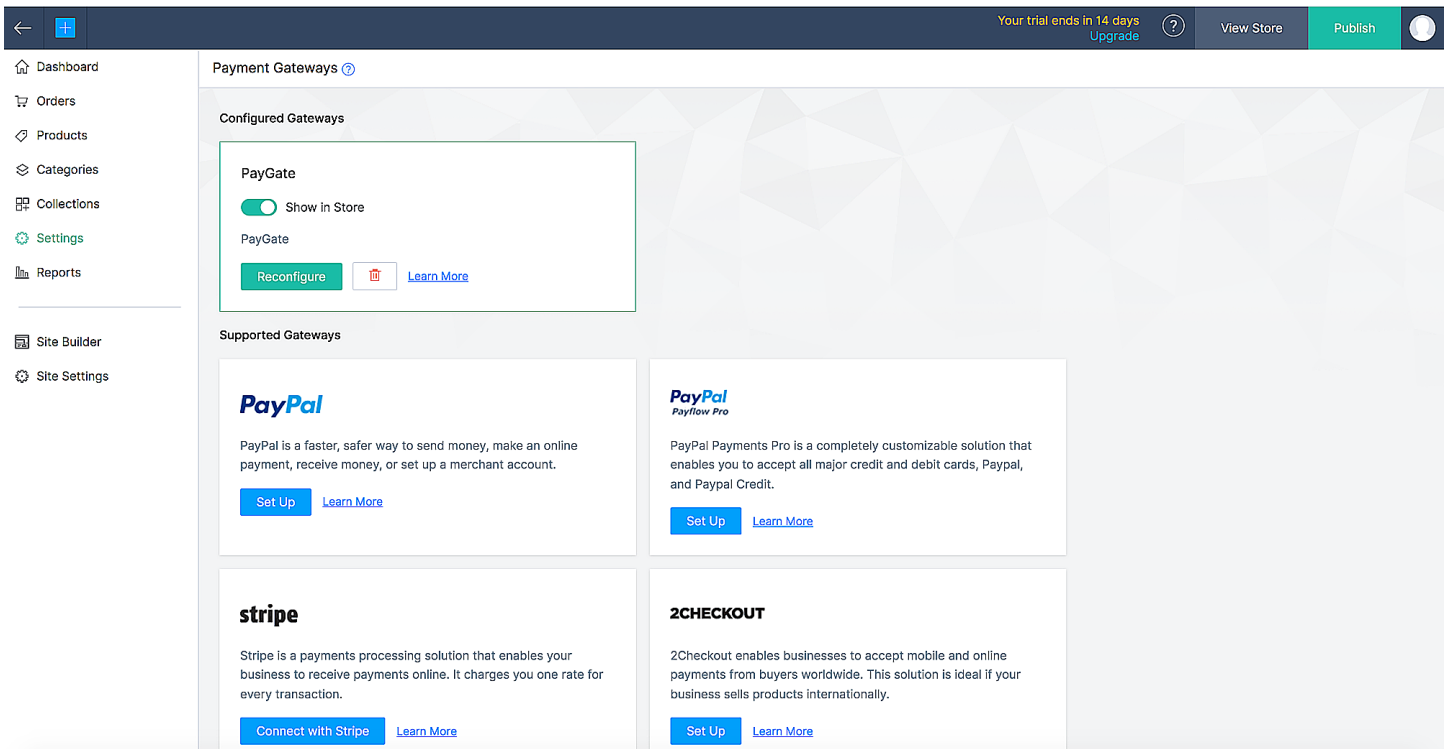


DRAFT

Configure :



Upon configuring your PG connector will look like below,



• Private sharing

Private sharing will work only if your extension is "private" published. You will be able to share the sigma app with other merchants privately.

Note: Before privately sharing the connector, you need to get a private review from the Zoho Commerce team. For this, you can write to us at pg-connector@zohocommerce.com. Once we review your connector and transform files, you can share it privately to another user's store by following the below steps:

Invite

- Visit [Zoho Sigma](#) and view the extension details of the extension you wish to share as shown in the screenshot
- Share the extension by giving store id as portal name and email address of the admin of that store as shown in the screenshot.

Extensions / PayGate



PayGate

Version 1.0

Published

Created on: 18-Nov-2021 07:16 PM

Edit Extension

Share Extension

General Details Version Developers Share

Share Extension

Portal Name *

Store Id

Admin Email Address *

mail id

Share

Cancel

Install

DRAFT

- Invited/Shared members will receive mail notification. They need to follow on screen steps to install the same in their store.

Accept access to install 'PayGate' extension in 'COMMERCE'



sigma@localinfo.zohosigma.com

TUE NOV 30 6:05 PM



Hello,

' is trying to share the extension 'PayGate' with you. On confirmation, you will be able to install this extension for all users in your portal for the service 'COMMERCE'. Would you like to accept this extension?

ACCEPT

REJECT

This email is from Zoho Sigma. If you think this is spam, please report it to abuse@zohocorp.com for immediate action.

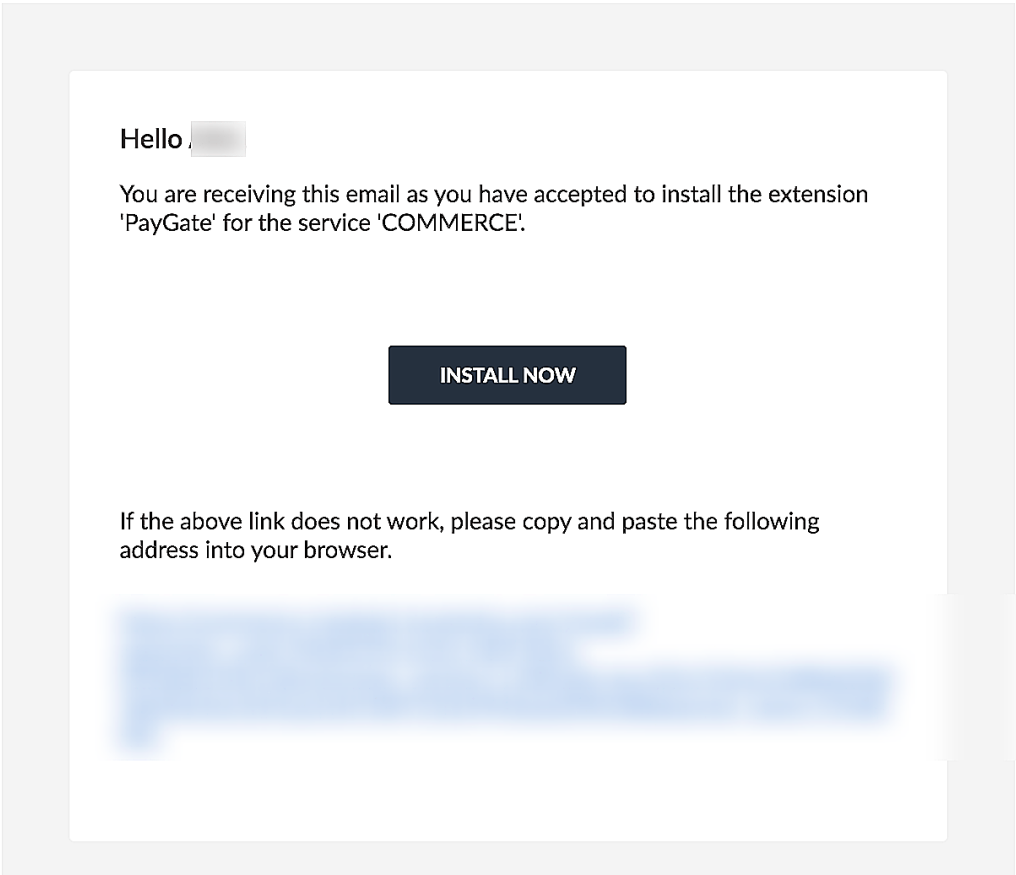
Zoho Corporation, 4141 Hacienda Drive, Pleasanton, CA 94588, USA
www.zoho.com

■ Upon accepting it, they will get an install url through mail using which they will be able to install in their store to which it has been shared. A sample install mail from sigma will look like the one in the below screenshot

SI sigma@localinfo.zohosigma.com 🔍
▶ TUE OCT 19 2:05 PM • 🔒

🕒 ⏪ ⏩ ⏴ ⏵

🔒   



- Click "Install Now" and follow the screens
- Payment Gateway extension will get successfully installed in their stores.

● Publishing in Zoho Marketplace

If you wish to publish your PG Connector publicly to all users, you need to do the following

1. Create a new extension in Zoho Sigma.(as steps mentioned already)
2. Create a new connector in one of your store in Zoho Commerce for the new extension id with the latest connector zip. (as steps mentioned already)
3. Get the latest version of the PG Connector approved by the Zoho Commerce team. To do this,

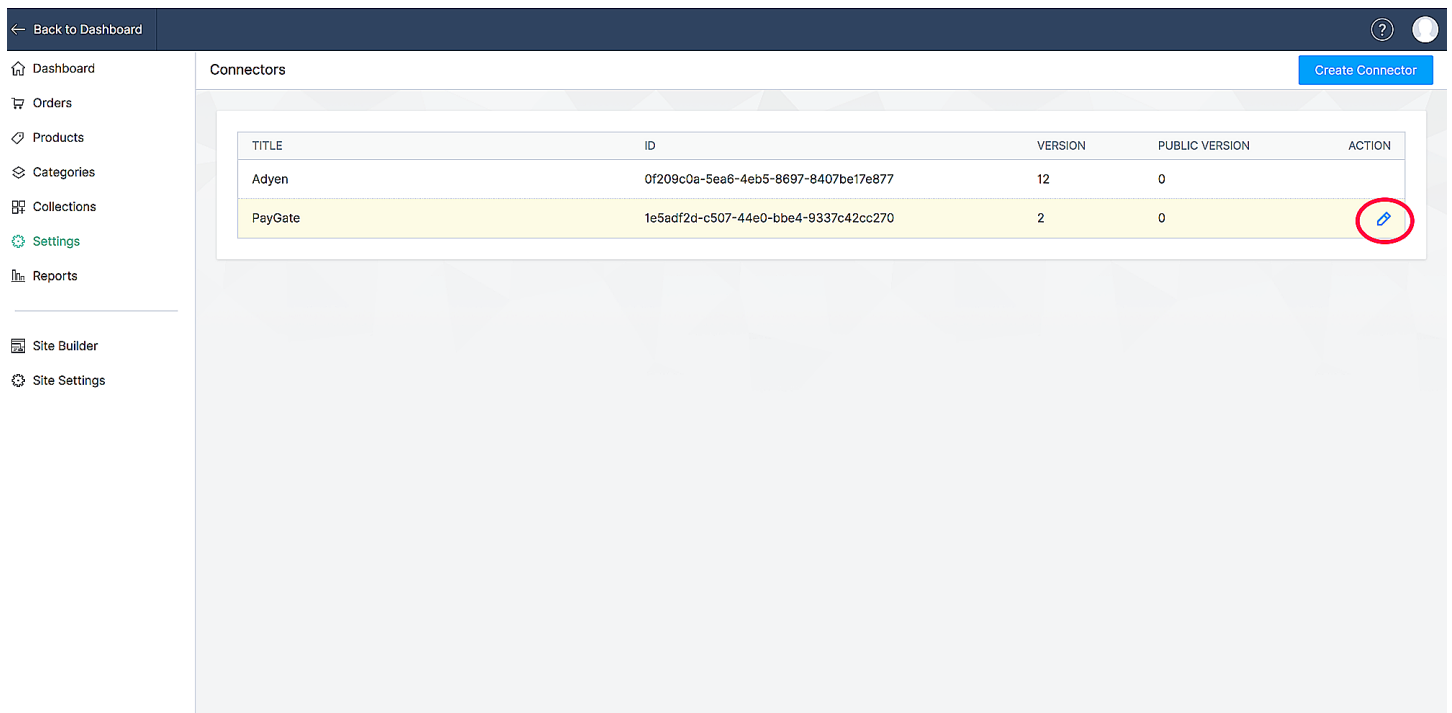
please write to us at pg-connector@zohocommerce.com. We will verify all your changes, and, if valid, it will be marked approved to be published in the marketplace.

4. Publish the extension in sigma with visibility public and submit the app for publishing in Zoho Marketplace.


P.S : For every new version of the connector, you will have to get it reviewed by Zoho Commerce Team before submitting it for publishing in the marketplace.

• Updating PG Connector

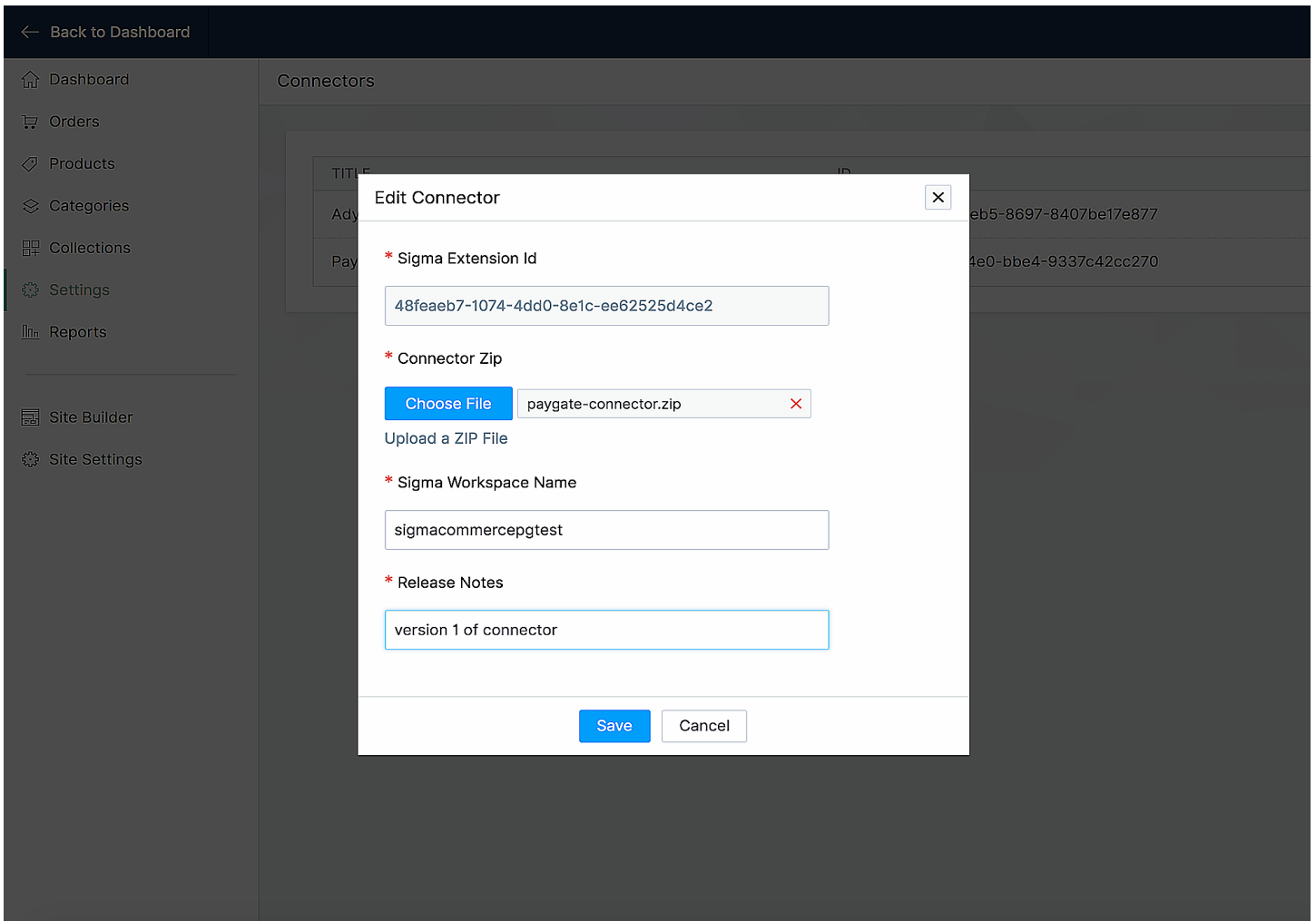
You can update your PG Connector too. From the list of PG Connectors available in your store, choose the one you would like to edit as shown below.



The screenshot displays the 'Connectors' management page in the Zoho Commerce dashboard. On the left is a navigation menu with options like Dashboard, Orders, Products, Categories, Collections, Settings, Reports, Site Builder, and Site Settings. The main content area shows a table of connectors. The 'PayGate' connector is highlighted in yellow, and its 'Action' column contains a blue pencil icon circled in red, indicating it is selected for editing.

TITLE	ID	VERSION	PUBLIC VERSION	ACTION
Adyen	0f209c0a-5ea6-4eb5-8697-8407be17e877	12	0	
PayGate	1e5adf2d-c507-44e0-bbe4-9337c42cc270	2	0	

Upload the modified connector json zip as shown in the figure and click on save to update the connector.



● Example

An example PG Connector has been developed for the Payment Gateway PayGate. The connector ZIP can be downloaded [here](#).

● Endpoint configuration in Payment Gateways

This depends on the Payment Gateway which you are trying to integrate. Some Payment Gateway expect this configuration in there dashboard/settings itself. You need to configure endpoints like payment notification urls or call back urls in Payment Gateways dashboard/settings to enable them to communicate back to your PG Connector. Given below is the general format to configure such urls in Payment Gateways.

```
{{published_store_url}}/pgc/{{extension_id}}/{{endpoint_uri}}
```


Example: <https://zylkershop.zohocommerce.com/pgc/66c74926-0c04-498c-777a-5d420d961234/paymentstatus>

In case of any violation found, Zoho Commerce Team has the right to revoke any PG Connector at any time. For any queries, kindly email us at pg-connector@zohocommerce.com.

DRAFT