INTEGRATING NEW PAYMENT GATEWAY WITH HOSTING CONTROLLER

Version 1.0

www.hostingcontroller.com

Proprietary Notice

Contents

Proprietary Notice 1
Target Audience 3
Introduction 3
Accepting Payments Online 3
Supported Gateways 4
Third-party Gateways: 4
Merchant Gateways: 4
Part 1 (on HC portal): 4
Part 2 (Backend Hook): 5
1. IMerchantGateway 6
2. IThirdPartyGateway 8
Contact Us 12

Target Audience

This guide is solely intended for those users who want to integrate their own payment Gateway with Hosting Controller for processing online payments. Integrating a Gateway means that the intended Gateway will be visible at the front end and your clients will be able to pay their bills online through that Gateway. It is assumed that the integrator has sufficient working knowledge of Microsoft Visual Studio and C#.net and knows how to program in these tools.

Introduction

Hosting Controller is a hybrid cloud control panel, designed to automate all aspects of your business including service provisioning, billing, metering, customer management and on-boarding of new services. It offers a fully integrated billing module, capable of accepting one-time and recurring online payments through tight integration of payment Gateways. If for any reason you don't want to utilize any of the already available payment Gateways in the panel, you may choose to integrate your own Gateway. The objective of the guide is to elaborate the process of integrating your own Gateway in the panel.

Accepting Payments Online

For E-commerce merchants it's particularly important to accept payments online. Accepting payments online entails online credit card processing. A payment Gateway is the main link between your bank and the customer's credit card provider. HC supports various payment Gateways as part of its billing module. Gateways can be enabled in the "Configure Billing" menu.

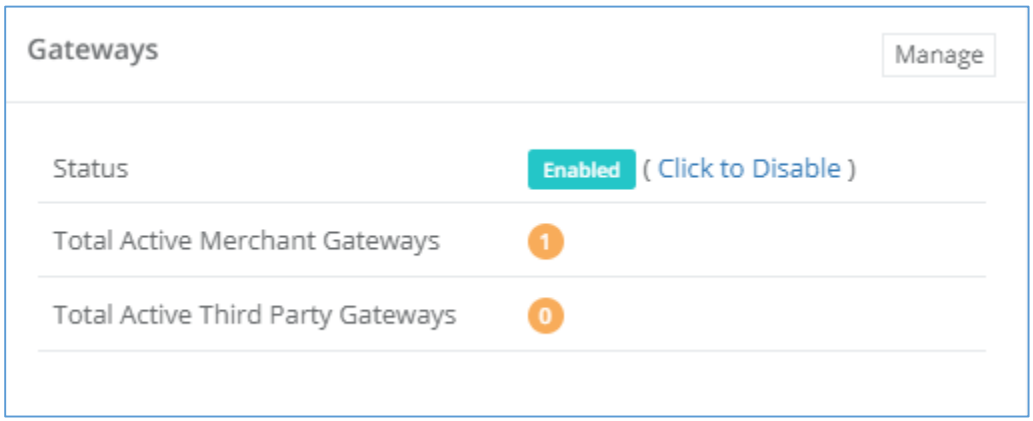

Figure 1: Third-party Gateways and Merchant Gateways as part of HC billing module.

For users who don't want to accept payments with already available Gateways in the panel, HC sanctions easy integration of other payment Gateways of choice. This guide demonstrates the integration process of such Gateways of choice.

Supported Gateways

Hosting Controller supports two types of payment Gateways.

Third-party Gateways:

These Gateways direct the customers away from the HC portal, taking them to the payment Gateway's website. Here customers fill in their payment details, and after paying, are redirected back to HC portal to complete the checkout process. A popular example of a Third-party Gateway is PayPal.

Merchant Gateways:

These Gateways process the payments remaining within HC portal. They use the securely stored credit card information from HC database and process payment within HC. An example of a Merchant Gateway is Authorize.Net.

Hosting Controller allows easy integration of new payment Gateways. There are two parts to these integrations. First where you add a new Gateway to HC with its detail and second where you need to add its backend code to send request to the Gateway and parse response to decide the payment status.

Part 1 (on HC portal):

The first step in integrating a new Gateway is determining its type. Depending on the type navigate to the appropriate section and click "Integrate New Gateway".

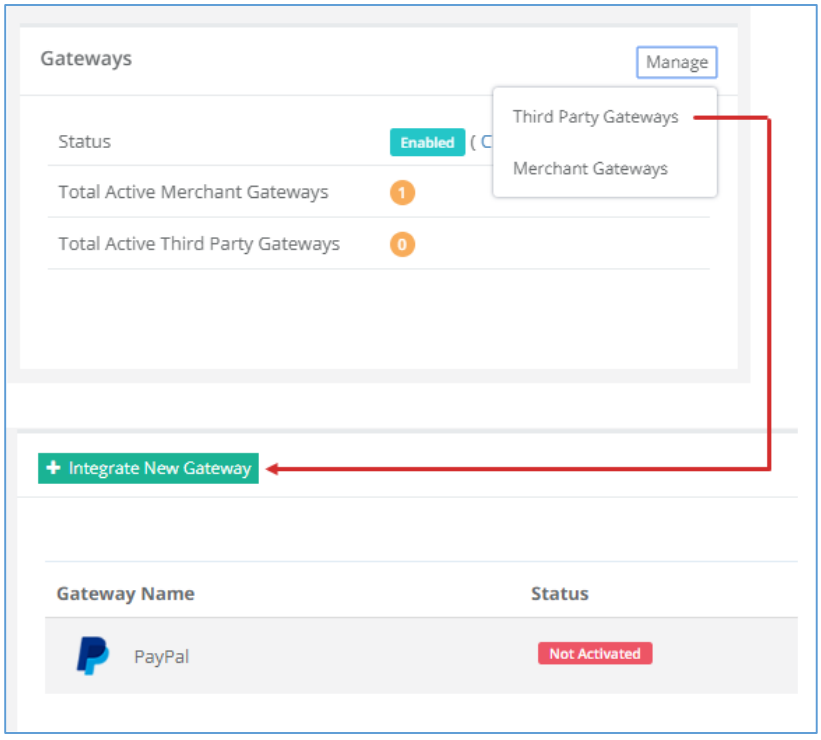

Figure 2: Integrating a new payment Gateway

Fill in the required fields. Enter Gateway's basic details. You may also add some additional fields (referred to as "custom fields") which will be displayed on actual payment form. Custom fields are used for asking additional information from the user while making payment.

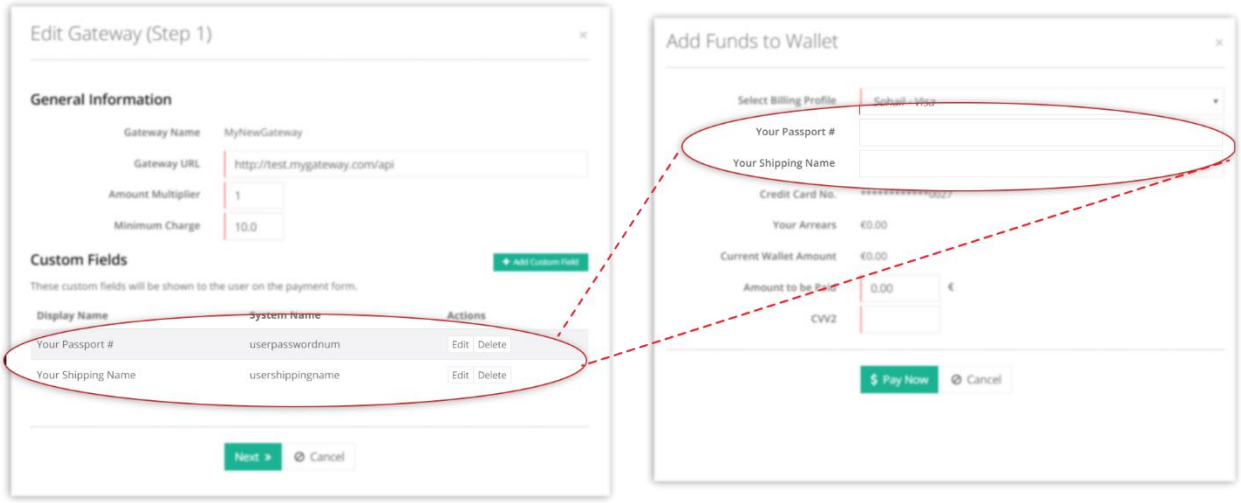

Figure 3: First step in integrating a new Gateway. Enter Gateway's basic details and additional details.

Once done, you can proceed to the next screen where you need to add Gateway configuration fields. These fields will be shown on Gateway configuration page mainly for Gateway credentials.

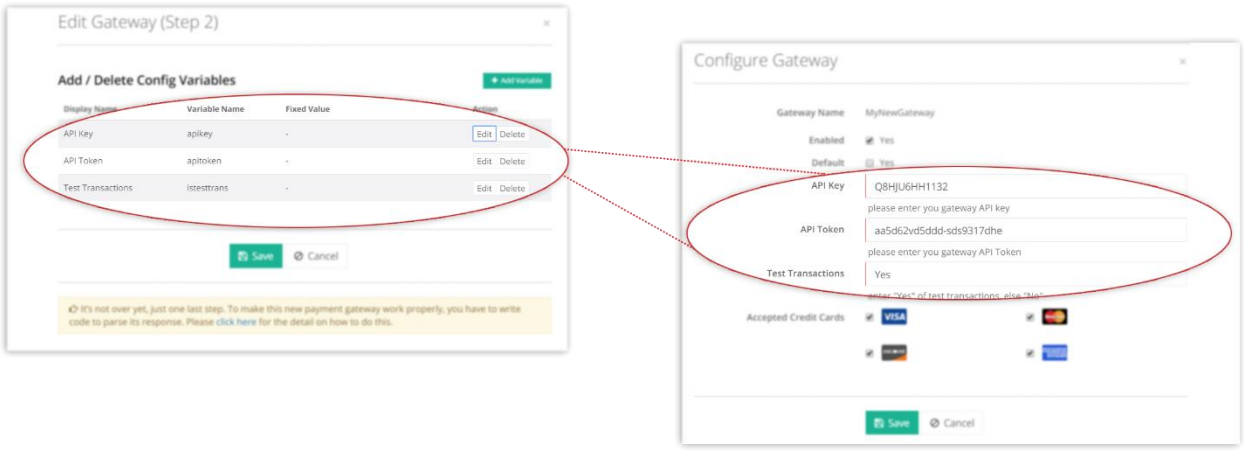

Figure 4: Second step in integrating a new Gateway. Add Gateway configuration fields.

Part 2 (Backend Hook):

The second step is to add the backend code of the Gateway to process the payment requests. For backend-code you need to implement methods of an interface from an assembly "HC.Common.Hooks.Interfaces.dll" which is shipped with HC installation, and can be found in the "Components" directory. This assembly contains two interfaces related to payment Gateways; one each for the two types (Third-party, Merchant).

Download the skeleton project as per the type of your Gateway:

- ThirdPartyGateway.zip
- MerchantGateway.zip

Then perform the following steps to get the project ready:

a. Open the downloaded project in Visual Studio 2015. It's a class library type project.

> NOTE
>
> Do not update the versions of any NuGet package already added in the skeleton project.

b. Add reference to the assembly "*HC.Common.Hooks.Interfaces.dll*" from "*Components*" folder in HC Installation location and rebuild the project.

c. To give a name to your implementation of the provided interface, open "*Module.cs*" file and update the component name (encircled string) with your new Gateway name.

   i. Name must be exactly the same as entered on the "Integrate New Gateway" form in HC.

Figure 5: Name your implementation of the provided interface.

d. Open "MyNewGateway.cs" file which implements an interface (either IMerchantGateway or IThirdPartyGateway) and add your code as per requirements.

   i. You can change file or class name as per your requirement.

Following is the detail of the interfaces:

1. IMerchantGateway

This interface contains only a single method and is for Merchant Gateways. It receives user billing profile, its credit card detail, Gateway credentials and other details as JSON input parameter.

Methods		

PayNow	**Input**	{
		"UserProfile":{
		"UserId":0,
		"UserName":" string ",
		"FirstName":" string ",
		"LastName":" string ",

		"EmailAddress":" string ",
		"City":" string ",
		"State":" string ",
		"StateCode":" string ",
		"Country":" string ",
		"CountryCode":" string ",
		"AddressLine1":" string ",
		"AddressLine2":" string ",
		"ZipCode":" string ",
		"FaxNo":" string ",
		"PhoneNo":" string ",
		"Language":" string ",
		
		"CreditCardNo":" string ",
		"Cvv2":" string ",
		"ExpiryMonth":0,
		"ExpiryYear":0,
		},
		"Transaction":{
		"TransactionId":" guid ",
		"AmountToPay":0.00,
		"GatewayAmount":0.00,
		"CurrencyCode":" string "
		},
		"Gateway":{
		"Name":" string ",
		"GatewayUrl":" string ",
		"ConfVariables":[
		{
		"VariableName":" string ",
		"Value":" string "
		},
		{
		"VariableName":" string ",
		"Value":" string "
		},
		.
		.
		.
		],
		"CustomFields":[
		{
		"FieldName":" string ",
		"Value":" string "
		},
		{
		"FieldName":" string ",
		"Value":" string "
		},
		.
		.
		.
		]

		} }
	Output	{ **"Status"**:"*Approved	Failed*", **"StatusDetail"**:" string ", **"GatewayResponse"**:" optional" }

Following image shows the payment process of Merchant Gateways and role of Hook in the process.

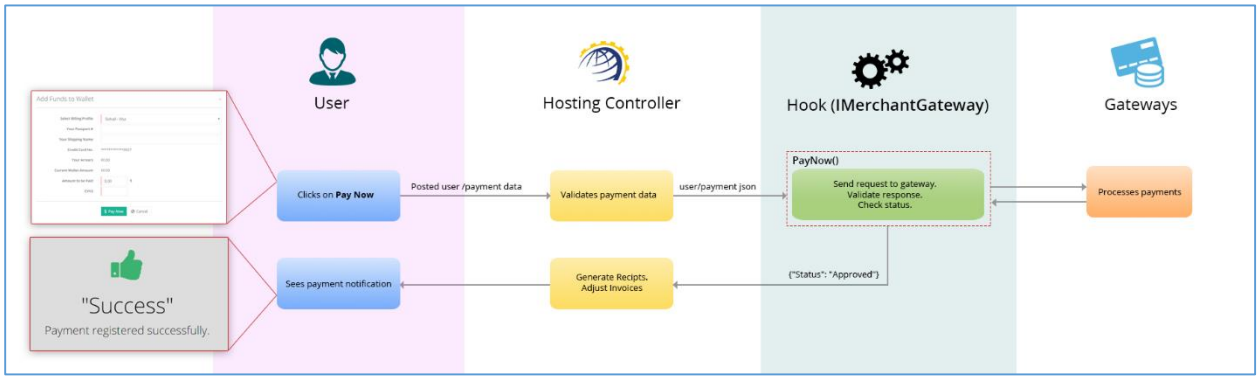

Figure 6: How HC, your implementation (Hook) and Merchant Gateway works.

2. IThirdPartyGateway

This interface has two methods; one that returns a URL which HC uses to redirect user to the Gateway's website to make payment and, second, which receives the payment response from the Gateway after completion of payment and decides the payment status.

Methods		

ConstructRequestUrl	**Input**	{
		"UserProfile":{
		"UserId":0,
		"UserName":" string ",
		"FirstName":" string ",
		"LastName":" string ",
		"EmailAddress":" string ",
		"City":" string ",
		"State":" string ",
		"StateCode":" string ",
		"Country":" string ",
		"CountryCode":" string ",
		"AddressLine1":" string ",
		"AddressLine2":" string ",
		"ZipCode":" string ",
		"FaxNo":" string ",

<table>
<tr>
<td></td>
<td></td>
<td>
<pre>
    "PhoneNo":" string ",
    "Language":" string "
  },
  "Transaction":{
    "TransactionId":" guid ",
    "AmountToPay":0.00,
    "GatewayAmount":0.00,
    "CurrencyCode":" string "
  },
  "Gateway":{
    "Name":" string ",
    "GatewayUrl":" string ",
    "NotifyUrl":" string ",
    "ReturnUrl":" string ",
    "ConfVariables":[
      {
        "VariableName":" string ",
        "Value":" string "
      },
      {
        "VariableName":" string ",
        "Value":" string "
      },
      .
      .
      .
    ],
    "CustomFields":[
      {
        "FieldName":" string ",
        "Value":" string "
      },
      {
        "FieldName":" string ",
        "Value":" string "
      },
      .
      .
      .
    ]
  }
}   .
    ]
  }
}
</pre>
<hr>
Note: “NotifyUrl and “ReturnUrl” properties contain two URL values which have to be used in the constructed “RedirectionUrl” in the appropriate URL variables (as mentioned by Gateway)
e.g for PayPal
…&return={ReturnUrl}&notify_url={NotifyUrl}….
</td>
</tr>
</table>

	Output	{
		"RedirectionUrl":" string "
		}

ParseResponse	Input	Param1 Serialized string of data posted by the Gateway to our notify URL e.g. payer_id=43EDT6MZJM7L4&payer_status=VERIFIED&first_name=mark&last_name=jhon&address_name=.....

		Param2
		{
		"UserProfile":{
		"UserId":0,
		"UserName":" string ",
		"FirstName":" string ",
		"LastName":" string ",
		"EmailAddress":" string ",
		"City":" string ",
		"State":" string ",
		"StateCode":" string ",
		"Country":" string ",
		"CountryCode":" string ",
		"AddressLine1":" string ",
		"AddressLine2":" string ",
		"ZipCode":" string ",
		"FaxNo":" string ",
		"PhoneNo":" string ",
		"Language":" string "
		},
		"Transaction":{
		"TransactionId":" guid ",
		"AmountToPay":0.00,
		"GatewayAmount":0.00,
		"CurrencyCode":" string "
		},
		"Gateway":{
		"Name":" string ",
		"GatewayUrl":" string ",
		"ConfVariables":[
		{
		"VariableName":" string ",
		"Value":" string "
		},
		{
		"VariableName":" string ",
		"Value":" string "
		},
		.

		.
		.
		],
		"CustomFields":[
		{
		"FieldName":" string ",
		"Value":" string "
		},
		{
		"FieldName":" string ",
		"Value":" string "
		},
		.
		.
		.
		]
		}
		}

	Output	{
		"Status":"*Approved | Failed*",
		"StatusDetail":" string ",
		"GatewayResponse":" optional"
		}

Following image shows the payment process of Third-party Gateways and role of Hook in the process.

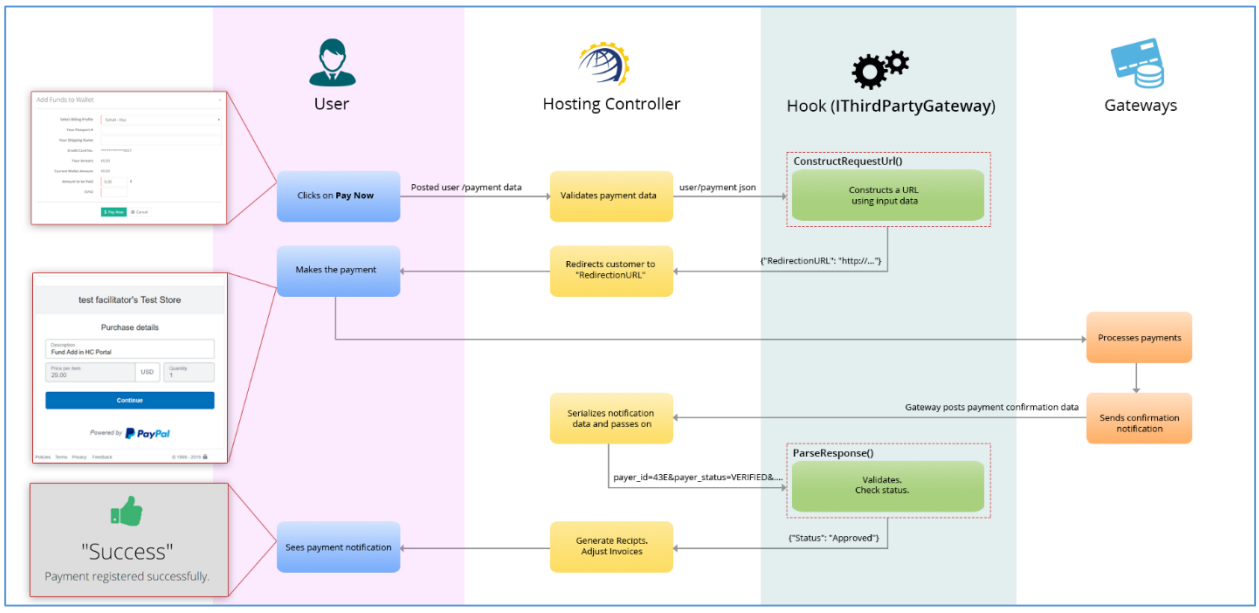

Figure 7: How HC, your implementation (Hook) and Third-party Gateway works.

Once done, just build the project and place the output assembly in the *"Components"* directory of HC10. That's it, restart HC API app pool and you are good to test your new integration.

Your assembly name must start with "Hook" e.g HookMyNewGateway.dll.

Contact Us

In case of any ambiguity/query regarding integration of payment Gateways, please feel free to contact us at support@hostingcontroller.com.