

---

# First Atlantic Commerce Hosted Page Integration Guide for Developers

---

*Version 1.5, 22<sup>nd</sup> August, 2018*

*Contents*

<b>Introduction .....</b>	<b>4</b>
<b>Integration Process Overview .....</b>	<b>4</b>
<b>Testing Considerations .....</b>	<b>6</b>
<b>Hosted Pages Overview .....</b>	<b>8</b>
Transaction Steps.....	9
Hosted Payment Page Transaction Flow Diagram.....	10
<b>Creating a Payment Page (Deep Dive Example) .....</b>	<b>11</b>
Implementing the Page.....	11
Page HTML Format.....	11
Permitted Fields .....	13
Using Merchant Administration Portal to Create your Hosted Page .....	15
<b>Integrating your Hosted Payment Page to your System .....</b>	<b>26</b>
Understanding the Integration Process.....	26
Available Interfaces.....	27
Integrating with the HostedPage SOAP interface.....	27
HostedPageAuthorize Operation.....	28
HostedPageAuthorizationRequest.....	28
HostedPageAuthorizationResponse .....	31
Integrating with the HostedPage XML POST Method .....	32
HostedPagePreprocessRequest .....	32
HostedPageResults method.....	34
HostedPageResultsResponse .....	35
Letting the Cardholder Enter the Amount.....	35
Kount Fraud Control Integration .....	36
Example of the complete flow (SOAP).....	38
Code Snippets .....	40
Putting it All Together .....	40
<b>APPENDICES .....</b>	<b>41</b>
Appendix 1 – Data Field Validation.....	41
Appendix 2 – AVS Field Requirements.....	43
Appendix 3 – ISO 3166 US State Codes .....	44
Appendix 4 – Test Cards for FAC Test Environment .....	45
Appendix 5 – Response Codes .....	47
Appendix 5.1 – System Response Codes and Reason Codes.....	47
Appendix 5.2 – ISO Response Codes.....	50
Appendix 5.3 – 3D-Secure Response Codes .....	54
Appendix 5.4 – AVS Response Codes.....	55
Appendix 5.5 – CVV Response Codes .....	56
Appendix 5.6 – Fraud Control Response Codes.....	57
Appendix 6 – PHP Code Snippets.....	59
Calling the Hosted Page Service Methods .....	59
Calling HostedPageAuthorize within a Page.....	59

Calling HostedPageResults within a Page .....	61
ComputeHash function .....	63
Appendix 7 – C# Code Snippets .....	64
Calling HostedPageAuthorize (Getting an HPP Token).....	64
Calling HostedPageResults (Retrieve transaction details).....	65
ComputeHash function .....	66
Appendix 8 – Signature and URL Encoding of the Signature .....	67
What is the Signature?.....	67
URL Encoding .....	67
Validating the ComputeHash Function on C#.....	68
Appendix 9 - Glossary of Terms .....	70

### Change Log

Document Version	Description	Release Date
<b>V1.0</b>	Initial version	20 <sup>th</sup> Jan 2012
<b>V1.1</b>	Web service name updated	9 <sup>th</sup> Jan 2013
<b>V1.2</b>	Fraud Control and other Enhancements	18 <sup>th</sup> Apr 2013
<b>V1.3</b>	Updated formatting, clarified several sections, added appendices	24 <sup>th</sup> Jul 2013
<b>V1.4</b>	Updated Device Data Collector form name in code snippet to correct form name	28 <sup>th</sup> Jul 2015
<b>V1.5</b>	Updated RecurringDetails in Detailed Field Descriptions Added XML POST specifications Added SOAP message sample for HostedPageResults Request Restructured code snippets into separate Appendices (6 - PHP and 7 - C#) Added C# code snippets for: <ul style="list-style-type: none"> <li>• HostedPageAuthorize</li> <li>• HostedPageResults</li> <li>• ComputeHash</li> </ul> Added new Appendix (8) Signature and URL Encoding of the Signature Expanded information on Page Templates Removed references to deprecated “MerRespURL” Updated CKEditor/CKFinder help links	22 <sup>nd</sup> Aug 2018

## Introduction

This document will guide a developer through the integration process required to use First Atlantic Commerce's (FAC) Payment Gateway (PG) Hosted Page (HP) Service and additional operations for managing your transactions. Using a Hosted Page, a merchant never needs to never have access to or know the Card Number (PAN) used by the Cardholder. Merchants should be aware that the FAC implementation of Hosted Page is not a way of reducing the work required to integrate to the Payment Gateway. The integration using a Hosted Page is just as complex as straight Authentication via our Web Services. A payment page could be as simple or complex as a merchant would like it to be. Although FAC's servers are hosting the merchant's payment page, the creation, coding and management of the page is of the responsibility of the merchant or merchant's developer(s).

It is also important to note that to integrate a merchant's site or payment module to FAC's gateway, a developer must be able to provide client side security to be able to connect to FAC using HTTPS as to pass data via SSL. The advantages in using Hosted Pages can be great, especially when you consider the PCI Audit requirements that come into scope when you store Card Numbers on your servers.

This document will guide you through the integration process, general steps in generating/publishing a payment page to FAC's servers along with requirements for implementing your Hosted Payment Hosted Page. The Hosted Payment Page transaction specifications within the document include the following:

- Standard Authorization Only or Authorization with Capture (with or without Address Verification)
- 3D Secure Authentication with Authorization or Authorization With Capture (with or without Address Verification)
- Tokenized Authorization Transactions
- Recurring Transactions
- Fraud Control (with Kount®)

Additional web services and operations outside of the Hosted Payment Page that can be used to manage your transactions include:

- Transaction Modification (for Captures, Reversals or Refunds)
- TransactionStatus
- Notifications

## Integration Process Overview

It is important to note that every integration differs in requirements, complexity, resources allocated to the integration and the expertise of the technical person(s) working on the implementation. That being said, integrations can take anywhere from 2-3 weeks to 2-3 months.

The technical integration process is usually initiated once the merchant's acquiring bank issues a Merchant ID number or a merchant is granted bank approval. Regardless of the payment processing requirements or payment gateway services needed here is the standard set of steps for a FAC integration:

1. **Technical Integration Guide Review** – FAC will provide the merchant and/or their technical team FAC's integration guide for the technical team to review.
2. **Completion of FAC's Processing Questionnaire** – FAC's business team will have provided you with FAC's processing questionnaire to gain an initial understanding of the merchant's processing requirements.
3. **Integration Call** – Once the integration guide has been reviewed and FAC's processing questionnaire has been completed and returned for FAC, FAC will request an integration call with the technical and business teams. The purpose of the call will serve as an introduction to all the relevant parties involved in the integration project, to discuss the merchant's processing requirements, provide an overview of the integration process and provide a

forum to go over any initial questions anyone may have. The call is not always technical in nature but it ensure that both business and technical teams of FAC and the merchant are all understanding the requirements and how to proceed with the integration.

4. **Provision of Test Account** – After the integration call, FAC will set up a test account based on the processing needs of the merchant as indicated on the processing questionnaire and discussed in the integration call. Here the merchant will be able to begin testing their technical integration to FAC.
5. **Testing on FAC Test Platform** - Testing on the test environment will allow you to test your code thoroughly without performing live transactions. This platform mimics the live environment. It is highly recommended to also perform data validation on the data fields on the payment page prior to transaction submission to FAC. Testing should also include the handling of approvals, declines and failed transactions. Additionally, in this environment, cards will not be charged, as the transactions do not go out to Interchange. It is recommended that you review the following section within this document on testing considerations to help guide you through your testing phase.
6. **Provision of Production Account** – Provided FAC has been given the bank-issued Merchant ID (Merchant Number) and after a merchant has successfully tested to their satisfaction in the test environment FAC will then provide the merchant with their live account credentials.
7. **Testing on FAC Production Platform** - Production testing is restricted to FAC core business hours which are Monday through Friday 8:30 AM ADT – 5:30 PM ADT. Again, it is highly advisable to verify that the data fields on the payment page are validated prior to transaction submission. Testing should also include the handling of approvals, declines and failed transactions.
8. **FAC End-to-End Testing** - FAC will conduct a site review and an end-to-end test from your site. It is usually asked that you set up a test product with a value of \$1.00 USD for FAC to use in their site review and end-to-end test. The end-to-end test validates the full transaction cycle from payment on a merchant’s site all the way to validating that the funds of the test transactions are confirmed deposited by the bank in the merchant’s account. This process can take up to 5 days depending on the processor and bank.
9. **FAC Business Team Go-Live Approval** - Provided all prior steps have been completed, your FAC Business Development representative will arrange an approved go-live date for the new site. It is important to note that FAC has a strict no go-live policy for Fridays, weekends and Bermuda public holidays.

## Testing Considerations

When testing your payment page and payment process it is important that you consider testing for the following:

### ✓ Data Validation

---

FAC performs basic data validation on parameter values submitted for credit card payments. The information that will be provided to FAC from the Hosted Payment Page should be validated or rendered to the proper format prior to payment submission as to avoid rejected transactions. If the formatting is improper, it will be processed as is (is not modified), therefore if invalid data is supplied or does not meet FAC's specifications, the transaction will fail or be rejected outright. It is highly recommended that you implement data validation to ensure the data passed is valid, scrubbed or rendered to the proper format. Parameters and their specifications are all outlined in this integration guide and it includes a quick reference table in the Appendix.

As an example: The 'CardNumber' specifications are that it be a 16 digit numeric value. FAC will not accept spaces, dashes, alpha characters or other symbols. If submitted the transaction will result in failure. In the event a Cardholder enters an invalid card number like 4111-1111-1111-1111, how do you want to handle this? Will the Cardholder see a message on the screen asking them to check their card number and allow them to try again? Will information be provided on the screen to the cardholder to advise them of the acceptable format? Will the Cardholder be restricted from entering anything but numbers?

**NOTE:** If you are utilizing Address Verification (AVS), it very important that you follow the guidelines for data formats. Issuers only validate standard alpha and numeric values. Ensuring that you are passing the appropriate data formats will not only reduce or eliminate rejections but you will get a more accurate AVS result.

### ✓ Transaction Approvals

---

Testing approvals seems quite simple, however you may want to consider any of your processes that are initiated by an approval. As an example, is there any additional customer validations that need to take place? Does the Cardholder receive an email confirmation or a receipt? Will an internal process be initiated internally with the business administrative staff or trigger a change in inventory?

### ✓ Transaction Declines

---

When testing Issuer/Processor declines you will want to be certain that your system is handling these according to the businesses requirements. Depending on the returned 'ReasonCode' you may want to display a particular message to the Cardholder. You may want to restrict how many times a Cardholder reattempt a transaction or notify them to contact a customer service agent to assist with their payment on the website. Your team may want to receive a notification when a Cardholder reaches a threshold of attempts.

### ✓ Transaction Errors

---

Once a transaction has been processed, it can have one of three statuses. It can be approved by the Issuer, declined by the Issuer/Processor and lastly, it could have failed. In case of a transaction failure, there is a problem with processing the transaction. It could be for a number of reasons but in all cases, FAC will respond with a 'ResponseCode' of 3. In these cases, you will want to check that your payment module is handling these as you require. As an example in a live

real-time environment, you may choose to display a message to the Cardholder to try again later and initiate a notification to your team for investigation.

---

✓ **Page Errors**

---

The Hosted Payment Page has a life cycle of 5 minutes, so you may want to consider how your site will handle cases where the Cardholder takes more than 5 minutes to complete their payment details and submit a payment. Perhaps in this circumstance you may want to display a message stating that their payment session expired and to try again or maybe also bring the Cardholder back to the payment page or home page.

---

✓ **Payment Page Browser Rendering (Page Format)**

---

You can design your payment page how you wish. It can be very simple and basic or you can add a lot more complexity to it. This can have an effect on how different browsers display your page and the functionality of buttons, fields or drop-downs (if used). Assuming the merchant's customers will be using other browsers other than Internet Explorer it may be wise to test your payment page from different internet browsers to validate how they are rendered.

---

✓ **Captures, Refunds and Reversals**

---

If you are using the 'TransactionModification' method for processing capturing (in a two-pass processing method), reversing or refunding transactions you will want to test that this is functional especially within the production environment. You will want to test cases where the requests are not only approved but denied as to ensure your system is handling them as deemed necessary by the business.

As an example, should a refund be denied because the refund cutoff period on your merchant account has expired, how will your system handle this? Will notification be sent out to the technical team? Will the person that processed the refund see on their screen a message stating that the refund did not go through?

As a second example, if you have implemented or are using address verification (AVS), depending on the AVS result, you may want to reverse a transaction to cancel it or capture a transaction if you wish to proceed with the payment.

---

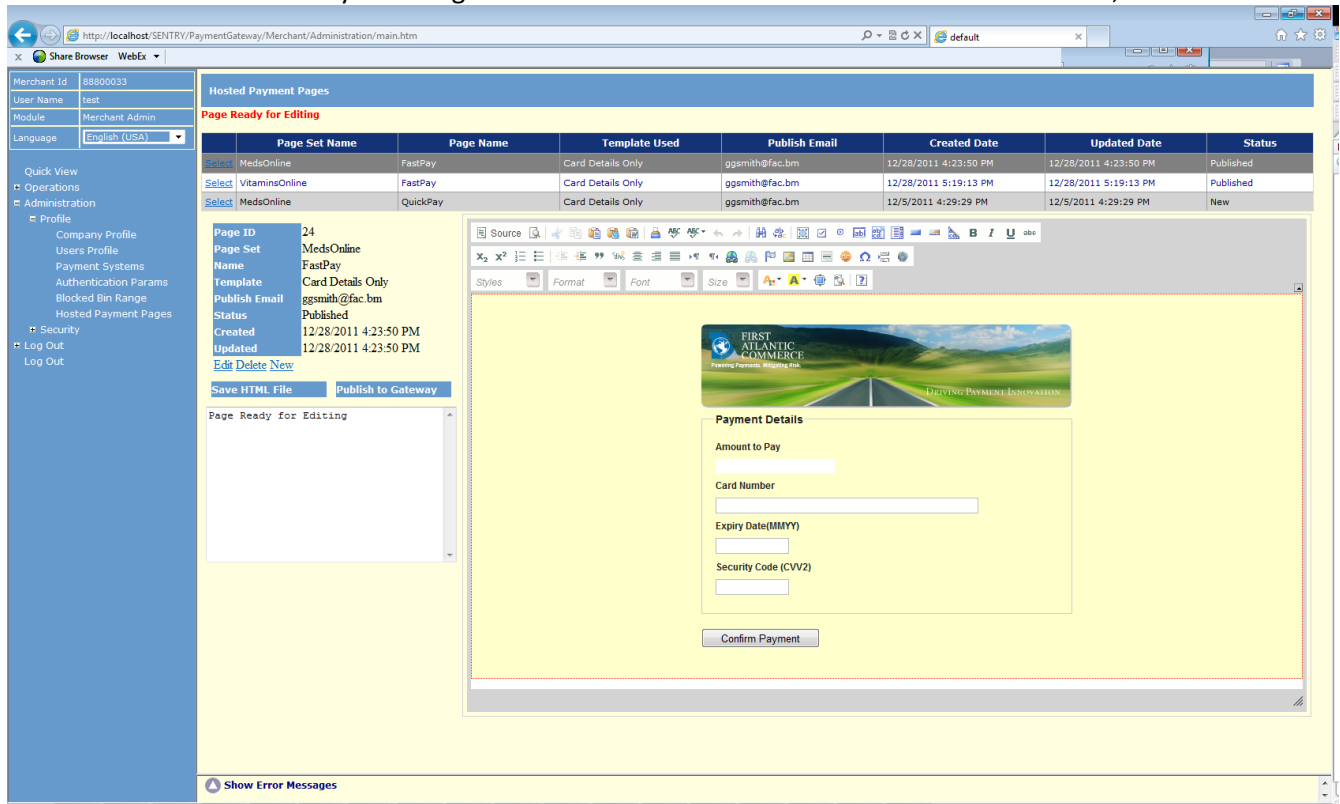
✓ **Overall Cardholder Payment Experience**

---

It is up to the merchant and the business to determine how they want to represent themselves, products, and services to their customers and how they want the overall customer experience to be when making a credit card payment on their site including your Hosted Payment Page. Although this is outside the scope of FAC we recommend that ample quality assurance be done on our site to satisfy the needs of the business and that it supports their required functions of the site/payment process.

## Hosted Pages Overview

Hosted pages are complete pages of HTML that reside on servers at FAC. They are designed and maintained by the Merchant in the “Hosted Payment Pages” section of the FAC Merchant Administration Portal, as follows:



The screenshot shows the 'Hosted Payment Pages' section of the FAC Merchant Administration Portal. The page is titled 'Page Ready for Editing'. A table lists the hosted pages:

Page Set Name	Page Name	Template Used	Publish Email	Created Date	Updated Date	Status
MedsOnline	FastPay	Card Details Only	gsmith@fac.bm	12/28/2011 4:23:50 PM	12/28/2011 4:23:50 PM	Published
VitaminsOnline	FastPay	Card Details Only	gsmith@fac.bm	12/28/2011 5:19:13 PM	12/28/2011 5:19:13 PM	Published
MedsOnline	QuickPay	Card Details Only	gsmith@fac.bm	12/5/2011 4:29:29 PM	12/5/2011 4:29:29 PM	New

Below the table, the details for the selected page (Page ID: 24) are shown:

- Page Set: MedsOnline
- Name: FastPay
- Template: Card Details Only
- Publish Email: gsmith@fac.bm
- Status: Published
- Created: 12/28/2011 4:23:50 PM
- Updated: 12/28/2011 4:23:50 PM

The main content area shows a preview of the hosted page, which includes the First Atlantic Commerce logo and a payment form with the following fields:

- Amount to Pay
- Card Number
- Expiry Date(MMY)
- Security Code (CVV2)

A 'Confirm Payment' button is located at the bottom of the form.

(Please treat all data shown here as fictitious. All screenshots are for exemplary purposes only.)

FAC will provide a “Developer” user ID for the person(s) responsible for editing and publishing the hosted payment page.

Once the page has been created and published, the merchant then needs to integrate with the FAC Payment Gateway v2.0 (FACPG2) in order to use the page in a live environment.

It is useful to note that the page can be presented to the user within an iframe on a merchant’s web page if so desired, so the hosted page then becomes minimal, with just the Card fields required and no extra formatting except for perhaps some color matching with the host site.

Before a Cardholder can use a hosted page, the merchant must obtain a HPP security token (SingleUseToken) from our server and pass this as part of the URL for the page.

The reasons behind this are security:

- 1) Sensitive information is passed with the request for a HPP security token (SingleUseToken) and therefore does not need to be included in the page. This includes the amount to pay, merchant details etc.
- 2) It stops just anyone executing a hosted page without a valid token as without the HPP security token (SingleUseToken) the URL is invalid.
- 3) The HPP security token (SingleUseToken) has a limited lifespan of 5 minutes (by default) and cannot be used twice adding another level of security.



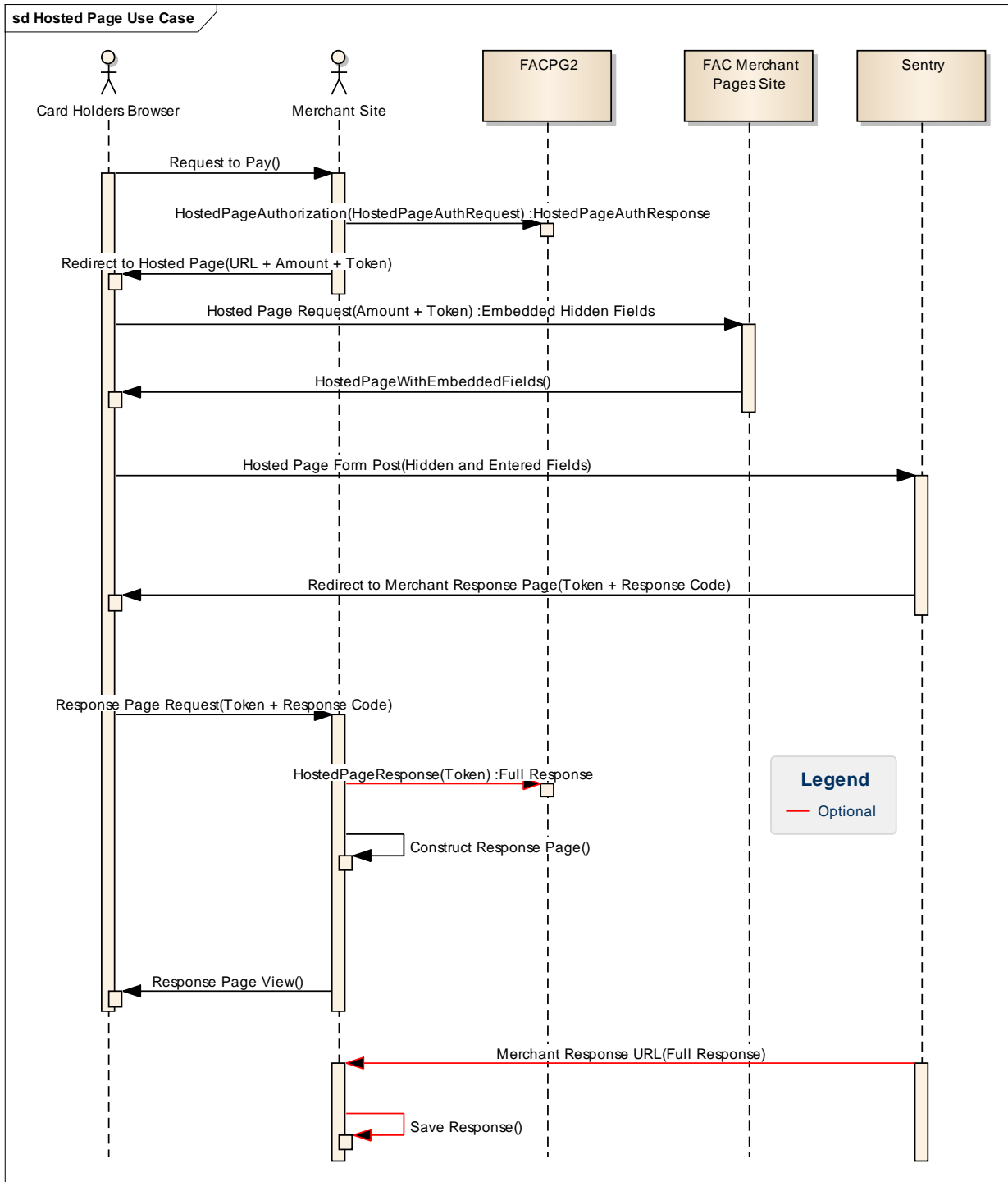
## Transaction Steps

A single transaction using a hosted page involves several steps as follows:

- The Merchant gets a request to pay from the Cardholder (they navigate to the payment page).
- To initialize the payment page, the Merchant calls a SOAP method called HostedPageAuthorize on the FACPG2 Gateway Service, passing in some of the transaction details and receiving a HPP security token (SingleUseToken).
  - **NOTE:** For XML-POST implementations, the Merchant makes a **HostedPagePreprocessRequest** instead of the HostedPageAuthorize call. [See Integrating with the HostedPage XML POST Method](#) for further details.
- The Merchant site constructs the URL to access the hosted page using the Merchant Pages Site Domain, Page Set name, the Page name and the HPP security token (SingleUseToken).
- The Cardholder is re-directed to this page or shown this page in an iframe.
- The Cardholder enters the Card Data and Posts the page. It is posted DIRECTLY to the FAC Payment Gateway and does not pass through the Merchant's Site.
- The basic response code and token are passed as parameters and sent to a "Card Holder Response URL", a page on the Merchant's site that will process the response code returned and present a suitable message to the Cardholder.

## Hosted Payment Page Transaction Flow Diagram

The following diagram shows the communication between all the entities involved.



## Creating a Payment Page (Deep Dive Example)

### Implementing the Page

#### Page HTML Format

The HTML page must adhere to a certain format as the POST to the FAC Payment Gateway expects to see a Form (and ONLY one form, with ID of "FrmCheckout") on the page. Here is what a page should look like before fields are added:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en-GB">
<head>
<title></title>
<link rel="stylesheet" href="css/blueprint/screen.css" type="text/css" media="screen, projection"/>
<meta http-equiv="Content-Type" content="application/xhtml+xml; charset=utf-8" />
</head>
<body>
<form id="FrmCheckout" method="post" action="" style="background-color: #FFFFCC" class="span-24">
  <div class="container">
    <div class="span-24">
      <p></p>
    </div>

    <div class="span-24">
      <p>
        <br />
        Content Goes Here
      <br />
    </p>
    </div>

    <div class="span-24">
      <p></p>
    </div>
  </div>
</form>
</body>
</html>
```

Note that we allow the use of the CSS library "Blueprint". This is for convenience and allows the use of div elements in a tabular format without too much CSS programming. It keeps your HTML clean of embedded STYLE elements. For more information on Blueprint, please see:

<http://blueprintcss.org/>

And, specifically, this tutorial is the most useful:

<http://net.tutsplus.com/tutorials/html-css-techniques/a-closer-look-at-the-blueprint-css-framework/>

**Page Rules:**

- The Page must start with an HTML element
- It must include a HEAD element with the links as shown above
- It must include a FORM element called “FrmCheckout”
- All the fields added must be one of the permitted fields (see the next section)
- All fields must be INPUT element fields
- There must be a button or mechanism that submits (POSTS) the Form.
- INPUT id and name attributes must be the same.

## Permitted Fields

The FAC Payment Gateway expects to see fields with specific names and descriptions. In addition, we recommend that validation be added to restrict the data entered to only the data required, in line with OWASP standards.

Some fields can be processed either by passing into the call to HostedPageAuthorize or by including on the form. If included on the form, the value of these fields take precedence over any values passed into HostedPageAuthorize. In this case, you should ensure that values on the form are valid and completed by the user before being processed.

Here is a Table of all INPUT fields of type TEXT allowed on hosted payment pages, with validation rules:

Category	Input "id"/"name"	Format	Notes
<b>Card Details</b>	Amount	N(4-10) "#0.00"	Optional. For displaying of amount to user. If added to the form will be auto-populated with Amount passed in call to HostedPageAuthorize. Will not be processed by hosted page. If edited by the user, should be used to populate the (hidden) PurchaseAmt field in Currency unit format (see below).
	CardNo	N(16 – 19)	Mandatory. Max 16 for non-Amex, 19 for Amex. <u>Numeric only</u>
	CardExpDate	N(4)	Mandatory. MMY Format
	CardCVV2	N(3 - 4)	Conditional. May be required depending on processor. Usually 3 digits.
	IssueNumber	N(2)	Required for Debit Cards Only where applicable (e.g. UK Debit cards)
	StartDate	N(4)	MMYY Format. Debit Cards only and is usually required if Issue number is not mandatory.
	PurchaseAmt	N(12) or N(4-10) Decimal "#0.00" format.	Optional. Transaction Amount in Currency units or Decimal format. Currency unit format is padded left with Zeros. E.g.: 10.00 = 0000000001000. If included in Form will override what has been passed into HostedPageAuthorize. Decimal format ("#0.00") will be converted to Currency Unit format when hosted page is posted.
	PurchaseCurrency	N(3)	Optional. ISO Numeric Currency code. E.g. 840 for US Dollars
	PurchaseCurrency Exponent	N(1)	Optional. Number of decimal places. Usually 2 for most currencies
SessionId	AN(30)	Optional. A Unique ID for Kount Fraud Control Processing. See the <a href="#">Fraud Control</a> Section for more information.	
<b>Billing Details (all optional)</b>	BillToFirstName	AN(30)	
	BillToMiddleName	AN(30)	
	BillToLastName	AN(30)	
	BillToAddress1	AN(50)	
	BillToAddress2	AN(50)	
	BillToCity	AN(30)	

	BillToState	AN(2)	State Code. Max A(2) if USA only. You could hide this field and use a drop down to set the value. Max A(3) for non US.
	BillToCounty	AN(15)	County Name
	BillToPostCode	AN(10)	Or Zip Code. Strictly Alpha-Numeric only.
	BillToCountry	N(3)	Country Code. Hide this field and use a drop down to set the value.
	BillToTelephone		
	BillToEmail		
	BillToFax	AN(30)	
	BillToMobile	AN(30)	
<b>Shipping Details (all optional)</b>	ShipToFirstName	AN(30)	
	ShipToMiddleName	AN(30)	
	ShipToLastName	AN(30)	
	ShipToAddress1	AN(50)	
	ShipToAddress2	AN(50)	
	ShipToCity	AN(30)	
	ShipToState	A(3)	State Code. Max A(2) if USA only. You could hide this field and use a drop down to set the value. Max A(3) for non US.
	ShipToCounty	AN(15)	County Name
	ShipToPostCode	AN(10)	Strictly Alpha-Numeric only.
	ShipToCountry	N(3)	Country Code. Hide this field and use a drop down to set the value.
	ShipToTelephone	AN(30)	
	ShipToEmail	AN(50)	
	ShipToFax	AN(30)	
ShipToMobile	AN(30)		

Address Text Validation Rules (for a full list, see the [Appendix](#)):

- No special characters
- No accents
- No special Symbols
- Avoid all unnecessary symbols
- Standard punctuation is OK
- Mandatory fields MUST have values

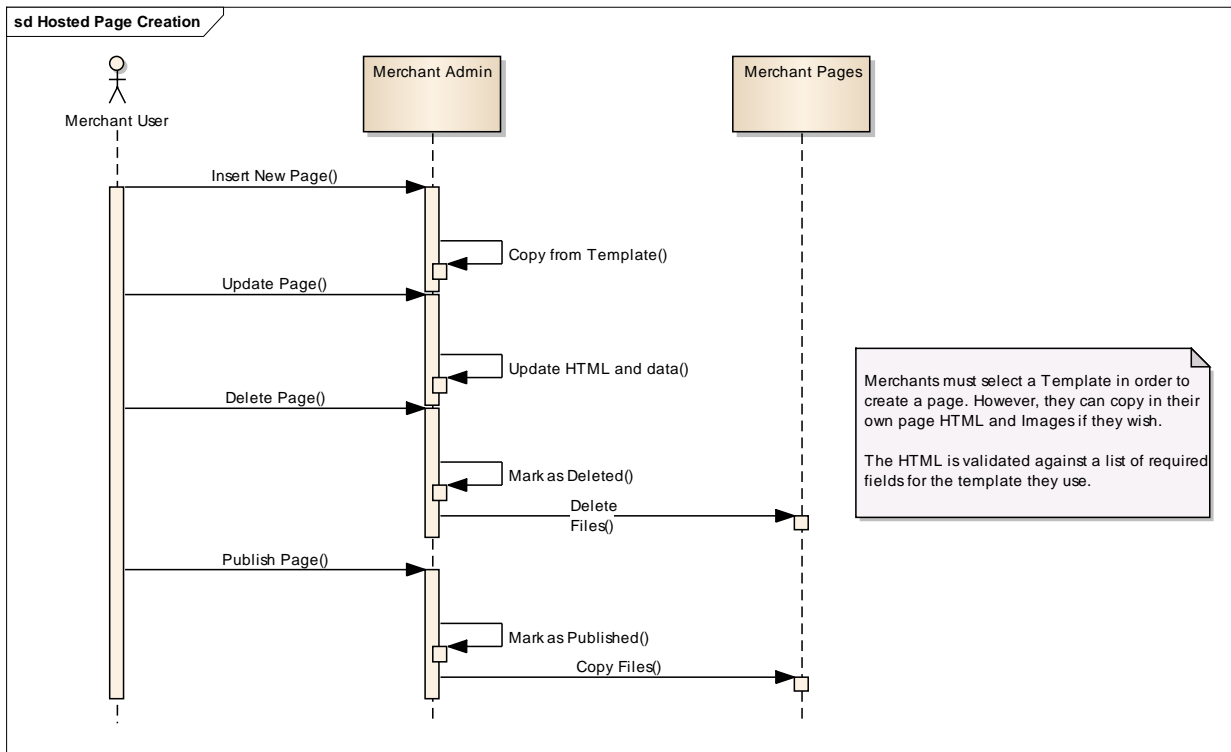
## Using Merchant Administration Portal to Create your Hosted Page

While it is possible to create a Page from scratch, it is advisable to use a template from our Merchant Hosted Page Administration App as a starting point for the Page implementation. This is accessible via FAC's Merchant Administration online portal.

You will need to use the Hosted Page Administration Application in Merchant Administration to publish the page on the FAC's Merchant Pages site.

To create a page, the developer/designer must follow some steps within the Hosted Page Administrator.

These are as follows:



FAC will provide you with access and login credentials to the Merchant Administration Online Portal. Here you will be able to create and manage your page(s).

The URL to the Merchant Administration portal:

Test environment -

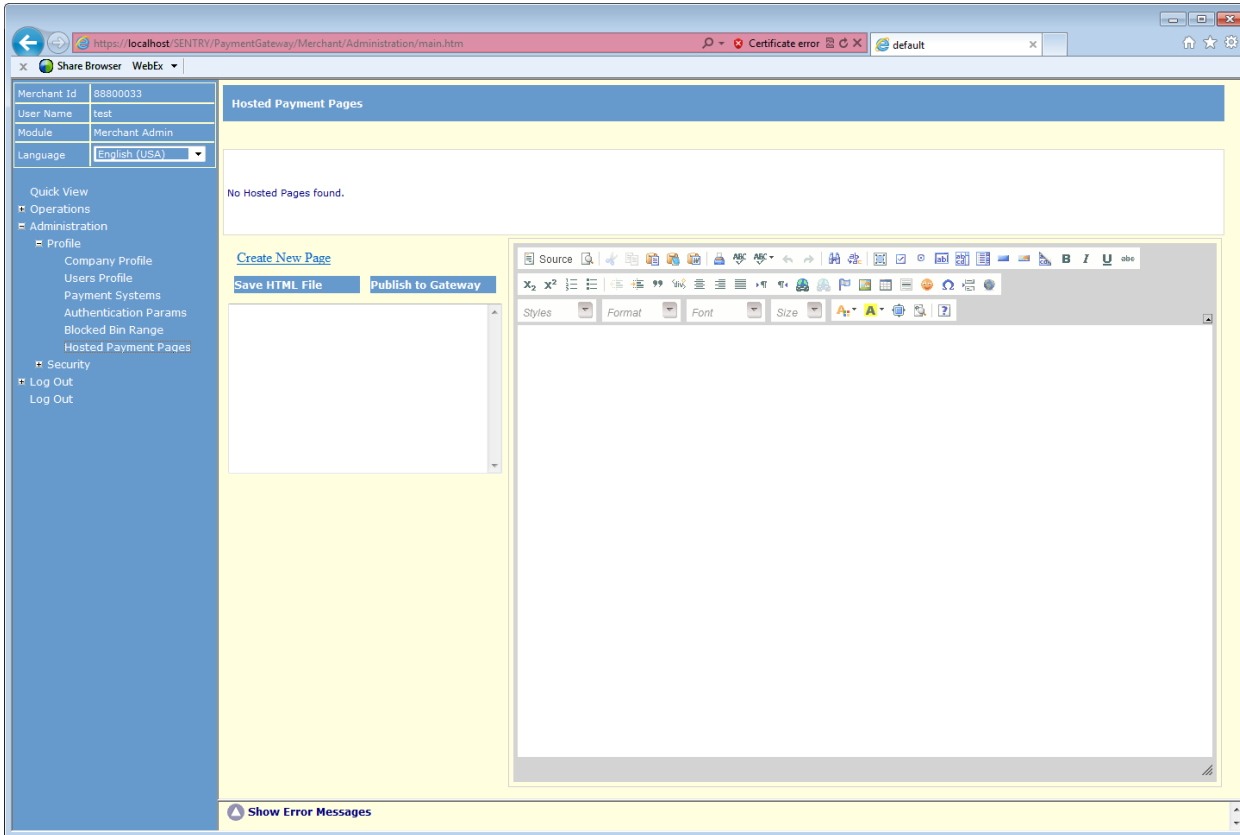
<https://ecm.firstatlanticcommerce.com/sentry/paymentgateway/merchant/administration/WFrmLogin.aspx>

Production environment -

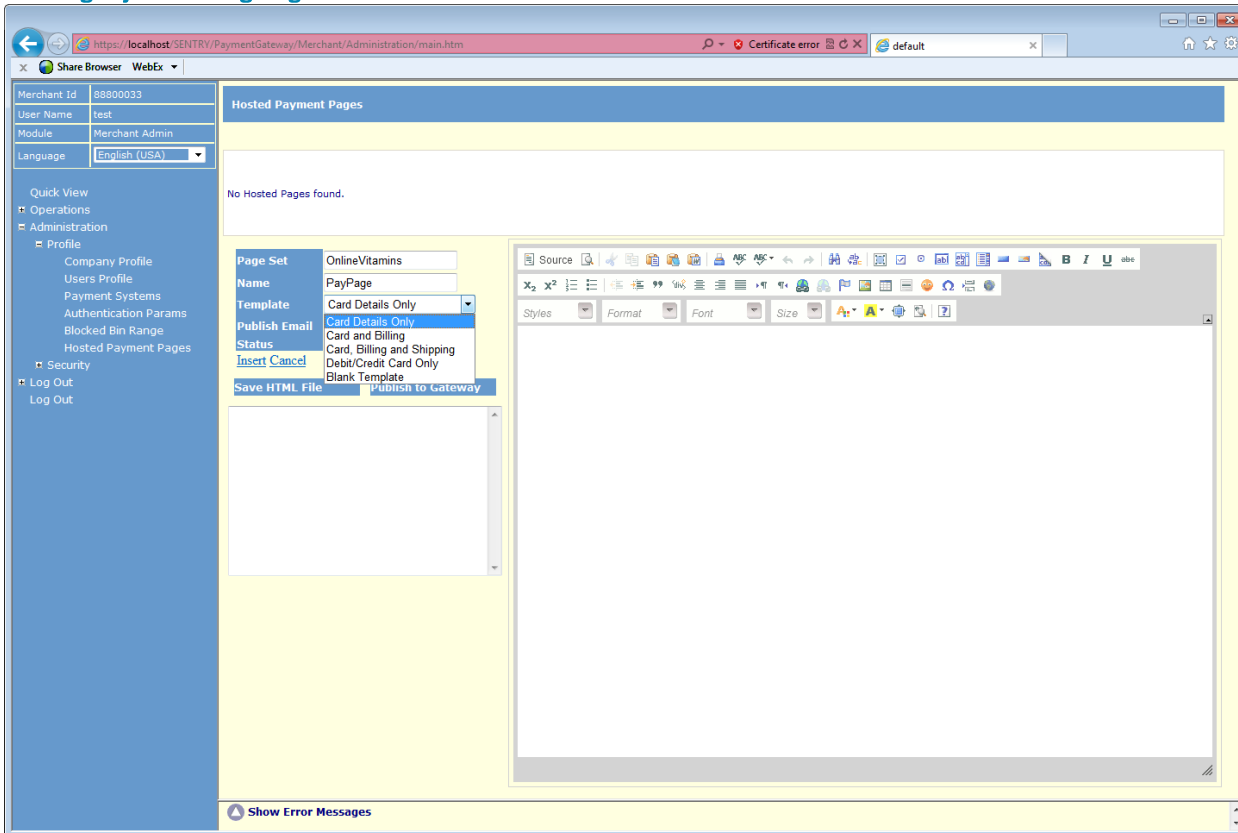
<https://marlin.firstatlanticcommerce.com/sentry/paymentgateway/merchant/administration/WFrmLogin.aspx>

Once you are logged in, you can navigate to the Hosted Pages Administration Application under the left hand side menu.

As you can see, there are no pages defined for your account. Select “Create New Page” and complete the form that displays:







The fields have the following meanings:

1. **Page Set** – The Name of a Set of Pages. This can be anything, but should be your business division name. This gives assurance to Cardholders that they are dealing with the Merchant even when paying directly to FAC.
  - Note: **Do not put any spaces in the name**, as it forms part of the URL for the Page.
2. **Page Name** – The Name of the Page. e.g. PayPage, PayNow, Payment. Can be whatever you decide.
  - Again, **do not use any spaces, as it is part of the Page URL**.
3. **Template** – Choose a page template. Choose carefully, as an inappropriate template could cause issues later. The templates and fields contained within each template are as follows:

A. Card Details Only

- **Fields:** Amount to Pay, Card Number, Expiry Date(MMY), Security Code (CVV2).
- **NOTE:** This is the most basic template containing **the minimum fields required**.
- There is no data validation built-in to the “Card Details Only” template. Merchants who use this template must add their own data validation.
- If you need a basic template with data validation already built-in, use the “**CheckoutWithValidation**” template (see **F** below).

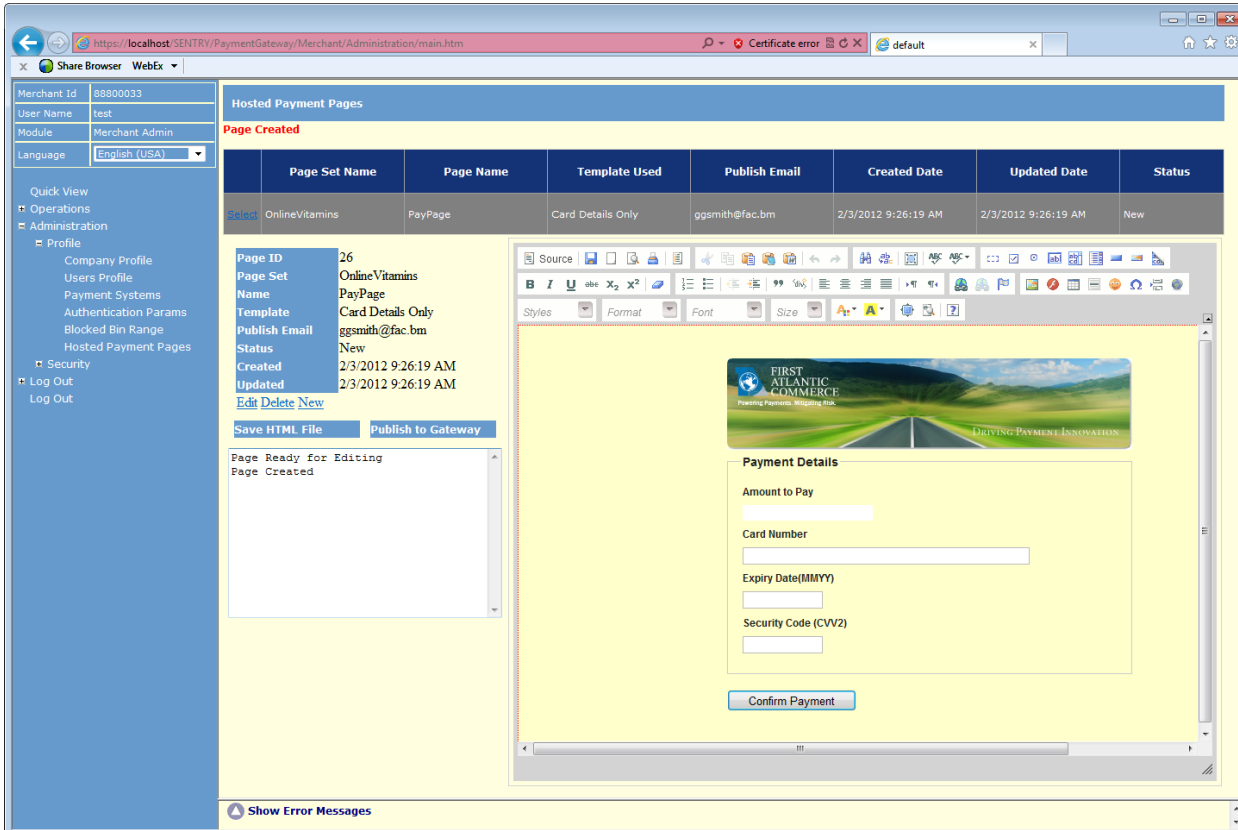
B. Card and Billing

- **Fields:** All of the above (Card Details) plus BILLING address fields:

- First Name, Last Name, Street Address, City, State/Region, Zip/Postal Code, Country Code, Telephone, Email.
  - **NOTE:** Billing address fields are OPTIONAL and not supported by all banks/processors.
  - There is no data validation built-in to the “Card and Billing” template. Merchants who use this template must add their own data validation.
- C. Card, Billing and Shipping
- **Fields:** All of the above (Card and Billing Details) plus SHIPPING address fields:
  - First Name, Last Name, Street Address, City, State/Region, Zip/Postal Code, Country Code, Telephone, Email.
  - **NOTE:** Billing and Shipping address fields are OPTIONAL and not supported by all banks/processors.
  - There is no data validation built-in to the “Card, Billing and Shipping” template. Merchants who use this template must add their own data validation.
- D. Debit/Credit Card Only
- **Fields:** Amount to Pay, Card Number, Expiry Date(MMY), Security Code (CVV2), Issue Number, Start Date(MMY)
  - **NOTE:** Issue Number, Start Date(MMY) fields are OPTIONAL and not supported by all banks/processors. It is best to avoid using this template unless instructed by FAC support.
  - There is no data validation built-in to the “Debit/Credit Card Only” template. Merchants who use this template must add their own data validation.
- E. Blank Template
- If you are starting from scratch, choose the Blank Template.
- F. CheckoutWithValidation
- **Fields:** Amount to Pay ([Currency]), Card Number, Expiry Date(MMY), Security Code (CVV2).
  - **NOTE:** The “CheckoutWithValidation” template contains **the minimum fields required, PLUS** it will enforce validation of the Card Number, Expiry Date(MMY), and Security Code.
  - For merchants’ convenience, data validation is built-in to the template for these 3 fields.
  - Merchants should replace the “Merchant Logo” image with their own logo, and replace “[Currency]” with their own merchant currency (e.g. “USD”).
  - This template also includes the ‘Visa’, ‘MasterCard’, ‘Verified by Visa’, ‘MasterCard SecureCode’ and ‘Powered by FAC’ logos. **All of these logos are required for merchants processing 3D Secure.** Merchants processing NON-3D Secure should remove the ‘Verified by Visa’ and ‘MasterCard SecureCode’ logos, as they are not applicable to non-3DS.
  - This template, when properly customized per above guidelines, meets FAC’s basic requirements that we review during **End to End testing** and **Site Review** of your website’s payment process.

4. **Publish Email** - The email address of the Author is a good candidate here. If there are any issues with the Page, FAC will contact the Author by using this email address. Note that the “Publish Email” is never visible to cardholders / end users of the payment page.
5. **Status** – a Read only field that shows the state of the page (New or Published).

Once completed, select the “Insert” link. The page will be added and the default template will appear in the editor:



The screenshot shows a web browser window displaying the Merchant Administration interface. The URL is `https://localhost:8080/ENTRY/PaymentGateway/Merchant/Administration/main.htm`. The interface includes a sidebar with navigation options like 'Quick View', 'Operations', 'Administration', and 'Profile'. The main content area is titled 'Hosted Payment Pages' and shows a table of existing pages. A 'Page Created' notification is visible. Below the table, there are details for a specific page (Page ID: 26) and a 'Payment Details' form with fields for 'Amount to Pay', 'Card Number', 'Expiry Date(MMYY)', and 'Security Code (CVV2)'. A 'Confirm Payment' button is at the bottom of the form.

Page Set Name	Page Name	Template Used	Publish Email	Created Date	Updated Date	Status
OnlineVitamins	PayPage	Card Details Only	ggsmith@fac.bm	2/3/2012 9:26:19 AM	2/3/2012 9:26:19 AM	New

Page ID: 26  
Page Set: OnlineVitamins  
Name: PayPage  
Template: Card Details Only  
Publish Email: ggsmith@fac.bm  
Status: New  
Created: 2/3/2012 9:26:19 AM  
Updated: 2/3/2012 9:26:19 AM

Payment Details

Amount to Pay:

Card Number:

Expiry Date(MMYY):

Security Code (CVV2):

Confirm Payment

This page is not set in stone. You are able to edit the page in any way you see fit. You can even use JavaScript libraries (from a CDN source) to enhance the page with UI widgets not available in plain HTML. **However, the minimum you will need to do is enhance the page to look like one of your own and to add data validation to the payment page fields to ensure the data passed is valid, scrubbed or rendered to the proper format.**

The templates use the following concepts by default:

1. A form set to the “post” method
2. A fieldset element to group “text” type input elements.
3. A label element related to each “text” input
4. Simple 3-column layout using Blueprint CSS
5. A single input with submit type for posting the form.

You are free to use any of the templates elements as they are, or change them for your own. However, the input fields posted with the form must exist in our [“Permitted Fields”](#) list.

The editor on the form is a standard HTML editor that has many features. It is called CKEditor and you can get a lot of information on this editor by looking at these links:

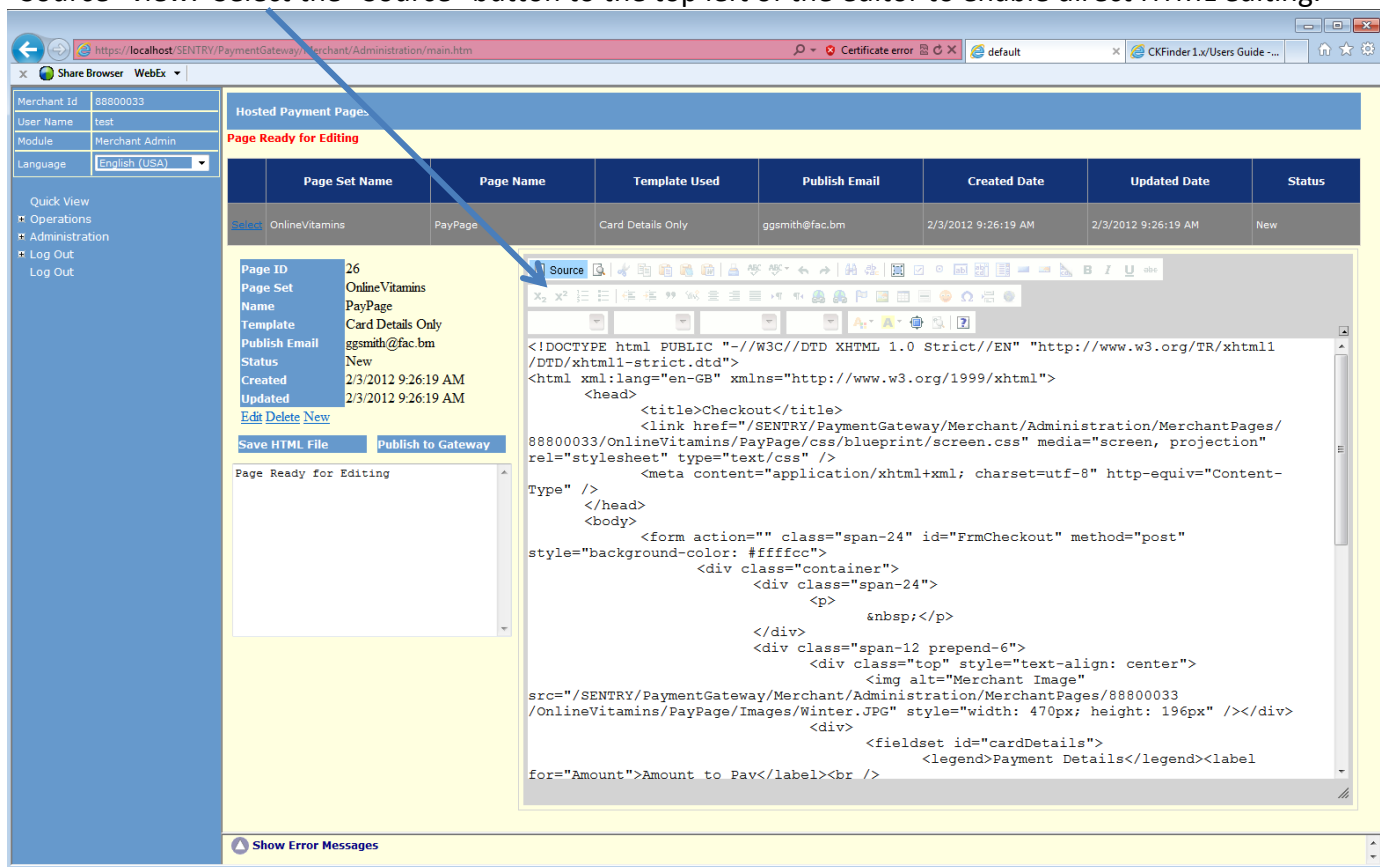
The CKEditor itself: <https://ckeditor.com/docs/ckeditor4/latest/guide/index.html>

The CKFinder image explorer and uploader: [https://docs-old.ckeditor.com/CKFinder\\_2.x/Users\\_Guide](https://docs-old.ckeditor.com/CKFinder_2.x/Users_Guide)

These are also linked to from the application via the help buttons on the editor and image uploader.

### Editing HTML directly

You can use the editor WYSIWYG facilities to edit the page or you can edit the HTML “in-situ” by using the “Source” view. Select the “Source” button to the top left of the editor to enable direct HTML editing.



Note that the Source view is not syntax highlighted, a feature we will be hoping to add in version 2.0. You may want to copy the source text and paste it into your favorite editor, and then paste it back once edited.

### Uploading Images

One thing you will definitely need to do is change the image on the Template page from the FAC default image to something related to your business. Of course, you can also delete the image. Hosted Pages allow you upload many image files for inclusion on your pages for, say, the company logo, card types accepted icons, etc. To change the image, click and select the image in the editor, then right click and select “Image Properties”

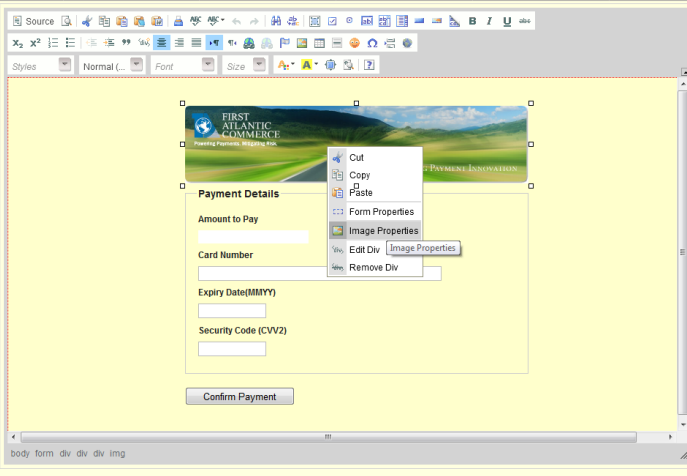
Hosted Payment Pages  
Page Ready for Editing

Page Set Name	Page Name	Template Used	Publish Email	Created Date	Updated Date	Status
OnlineVitamins	PayPage	Card Details Only	ggsmith@fac.bm	2/3/2012 9:26:19 AM	2/3/2012 9:26:19 AM	New

Page ID: 26  
Page Set: OnlineVitamins  
Name: PayPage  
Template: Card Details Only  
Publish Email: ggsmith@fac.bm  
Status: New  
Created: 2/3/2012 9:26:19 AM  
Updated: 2/3/2012 9:26:19 AM  
[Edit](#) [Delete](#) [New](#)

[Save HTML File](#) [Publish to Gateway](#)

Page Ready for Editing



You will then see an Image Properties dialog.

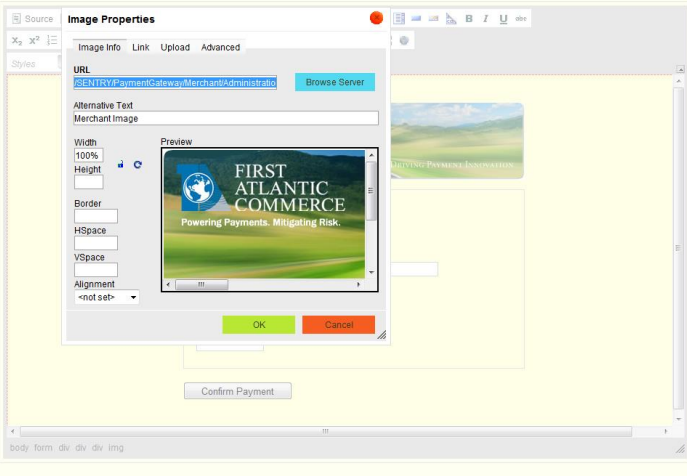
Hosted Payment Pages  
Page Ready for Editing

Page Set Name	Page Name	Template Used	Publish Email	Created Date	Updated Date	Status
OnlineVitamins	PayPage	Card Details Only	ggsmith@fac.bm	2/3/2012 9:26:19 AM	2/3/2012 9:26:19 AM	New

Page ID: 26  
Page Set: OnlineVitamins  
Name: PayPage  
Template: Card Details Only  
Publish Email: ggsmith@fac.bm  
Status: New  
Created: 2/3/2012 9:26:19 AM  
Updated: 2/3/2012 9:26:19 AM  
[Edit](#) [Delete](#) [New](#)

[Save HTML File](#) [Publish to Gateway](#)

Page Ready for Editing



You have two choices here, for a single file upload, use the “Upload” tab.

For uploading multiple files, use the Server Browser.

For a single file upload, browse to the file and then “Send to Server” as follows:

Hosted Payment Pages  
Page Ready for Editing

Page Set Name	Page Name	Template Used	Publish Email	Created Date	Updated Date	Status
OnlineVitamins	PayPage	Card Details Only	ggsmith@fac.bm	2/3/2012 9:26:19 AM	2/3/2012 9:26:19 AM	New

Page ID: 26  
Page Set: OnlineVitamins  
Name: PayPage  
Template: Card Details Only  
Publish Email: ggsmith@fac.bm  
Status: New  
Created: 2/3/2012 9:26:19 AM  
Updated: 2/3/2012 9:26:19 AM  
Edit Delete New

Save HTML File Publish to Gateway

Page Ready for Editing

**Image Properties**

Image Info Link Upload Advanced

Send it to the Server  
C:\Users\Public\Pictures\Sample Pict... [Browse...](#)

Send it to the Server

Send it to the Server

OK Cancel

It will put the Image in the “Image Properties” dialog.

Hosted Payment Pages  
Page Ready for Editing

Page Set Name	Page Name	Template Used	Publish Email	Created Date	Updated Date	Status
OnlineVitamins	PayPage	Card Details Only	ggsmith@fac.bm	2/3/2012 9:26:19 AM	2/3/2012 9:26:19 AM	New

Page ID: 26  
Page Set: OnlineVitamins  
Name: PayPage  
Template: Card Details Only  
Publish Email: ggsmith@fac.bm  
Status: New  
Created: 2/3/2012 9:26:19 AM  
Updated: 2/3/2012 9:26:19 AM  
Edit Delete New

Save HTML File Publish to Gateway

Page Ready for Editing

**Image Properties**

Image Info Link Upload Advanced

URL: /ENTRY/PaymentGateway/Merchant/Administrato... [Browse Server](#)

Alternative Text  
Merchant Image

Width: 600  
Height: 400

Border  
HSpace  
VSpace  
Alignment: <not set>

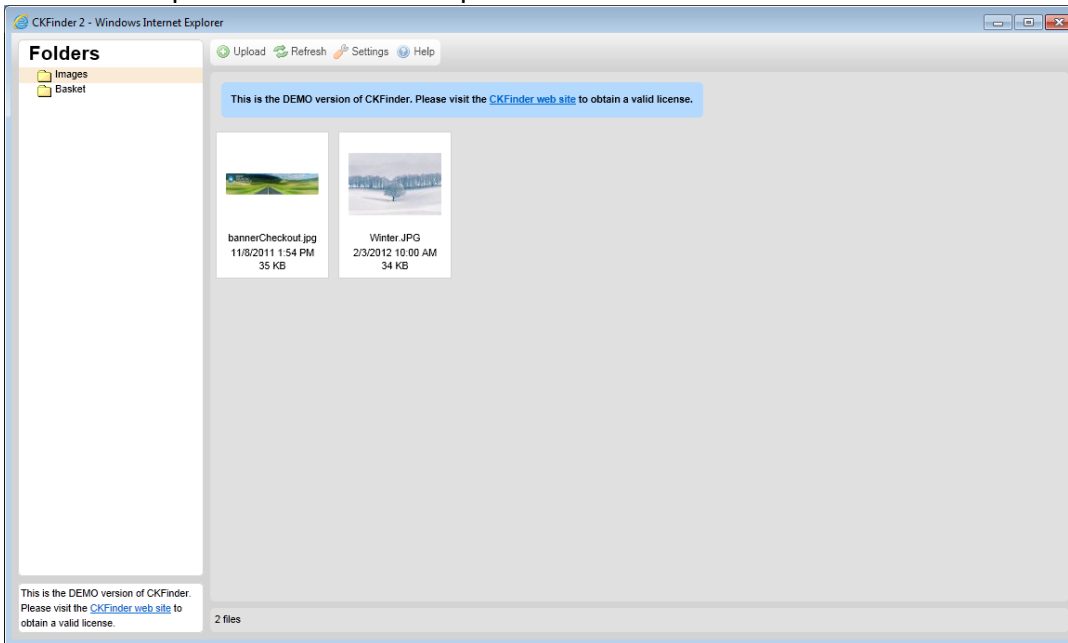
Preview

OK Cancel

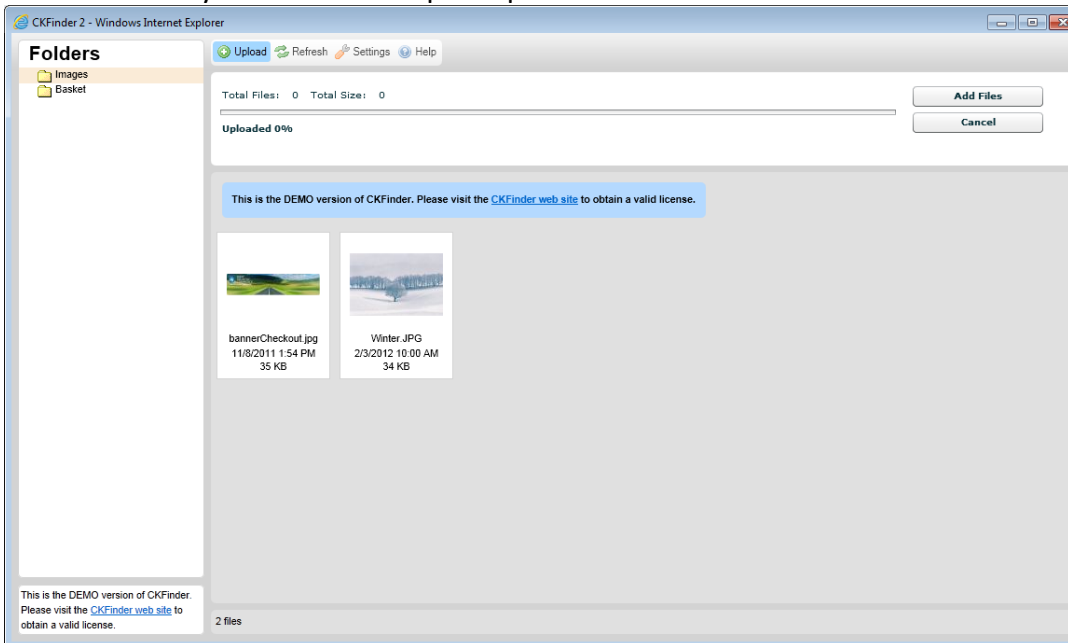
Then if you select “OK” it will replace the image in the Page.

For Multiple file uploads or purely to use a nicer browsing interface, choose “Browse Server”, this will invoke the CKFinder image explorer and uploader.

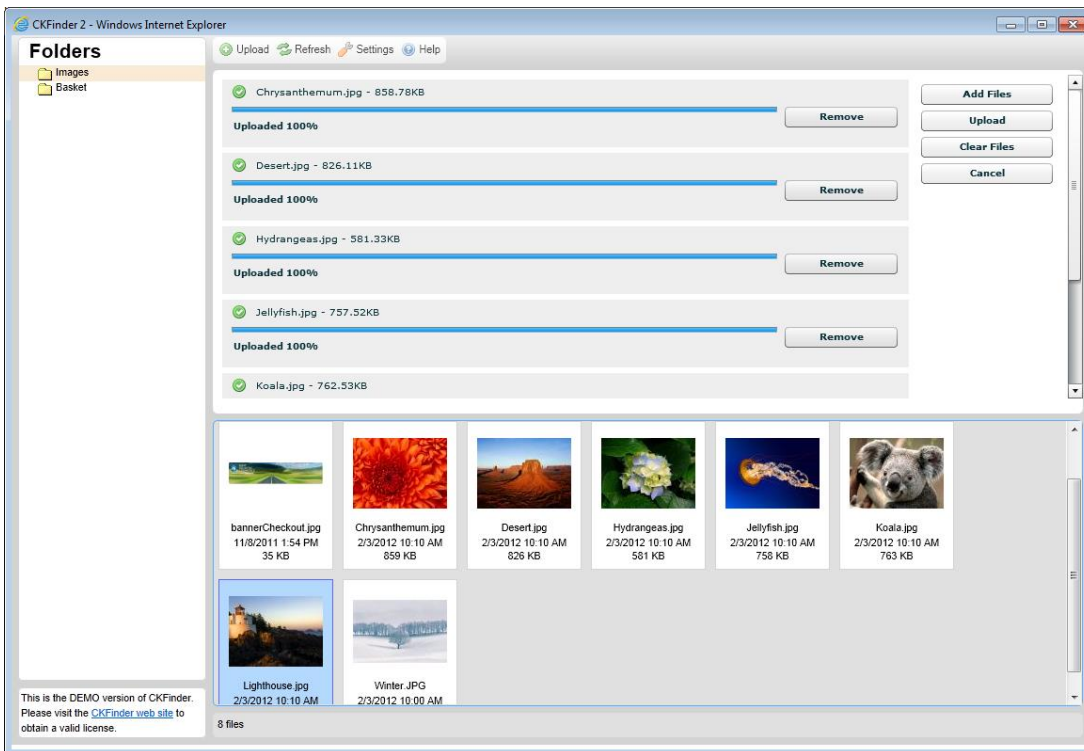
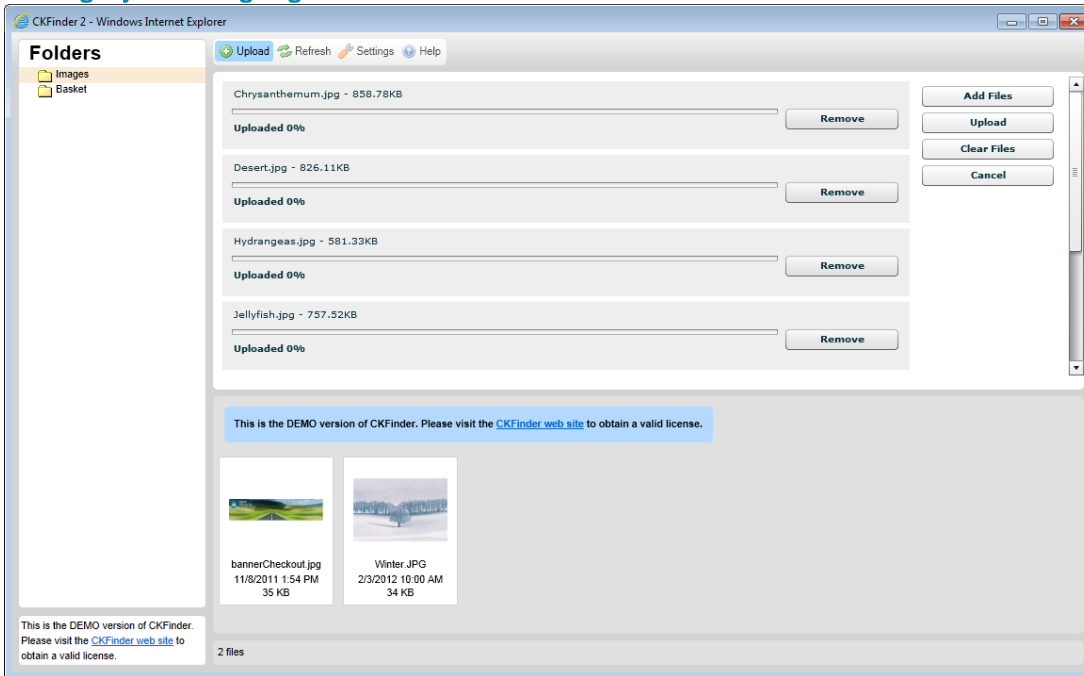
This has an upload button at the top left of the icon screen.



Select this and you will see the upload panel.



Select "Add Files" and this will launch a File dialog. Select multiple files and select "Open" and you will see the files listed in the Upload Panel. Select the "Upload" button to upload to the FAC server.



It is important to note at this stage that each Page created is “Sandboxed” and separate from any other page. All images uploaded are only available for current Page being edited. This is necessary, as it safeguards other pages when an image or page is deleted. By clicking on any of the uploaded images, the image will be displayed in the opened “Image Properties” dialog. Press “OK” on the dialog to commit the image to the Page.



Hosted Payment Pages

Page Ready for Editing

	Page Set Name	Page Name	Template Used	Publish Email	Created Date	Updated Date	Status
Select	OnlineVitamins	PayPage	Card Details Only	ggsmith@fac.bm	2/3/2012 9:26:19 AM	2/3/2012 9:26:19 AM	New

Page ID: 26

Page Set Name: OnlineVitamins

Name: PayPage

Template: Card Details Only

Publish Email: ggsmith@fac.bm

Status: New

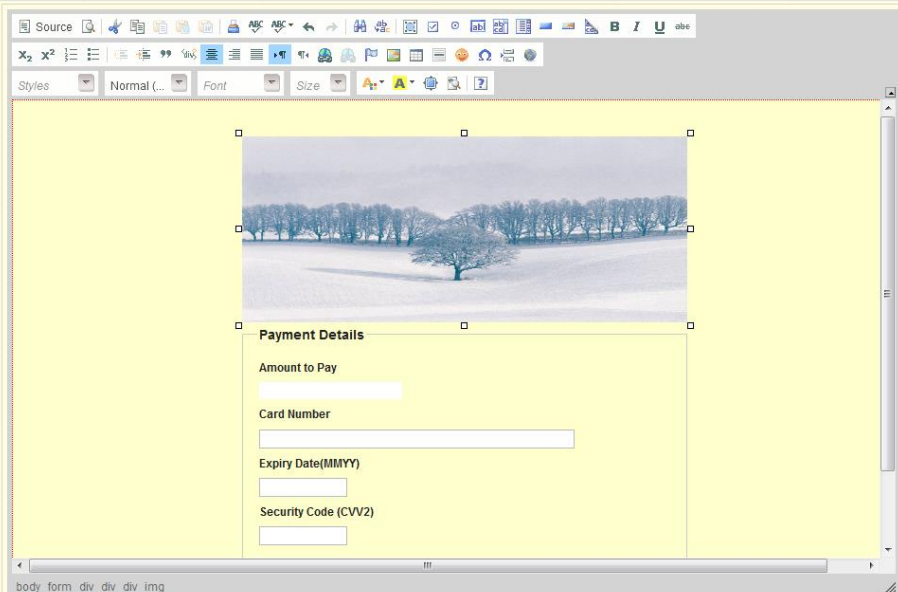
Created: 2/3/2012 9:26:19 AM

Updated: 2/3/2012 9:26:19 AM

[Edit](#) [Delete](#) [New](#)

[Save HTML File](#) [Publish to Gateway](#)

Page Ready for Editing



While the image is selected, you can re-size it as you wish.

### *Saving and Publishing the Page*

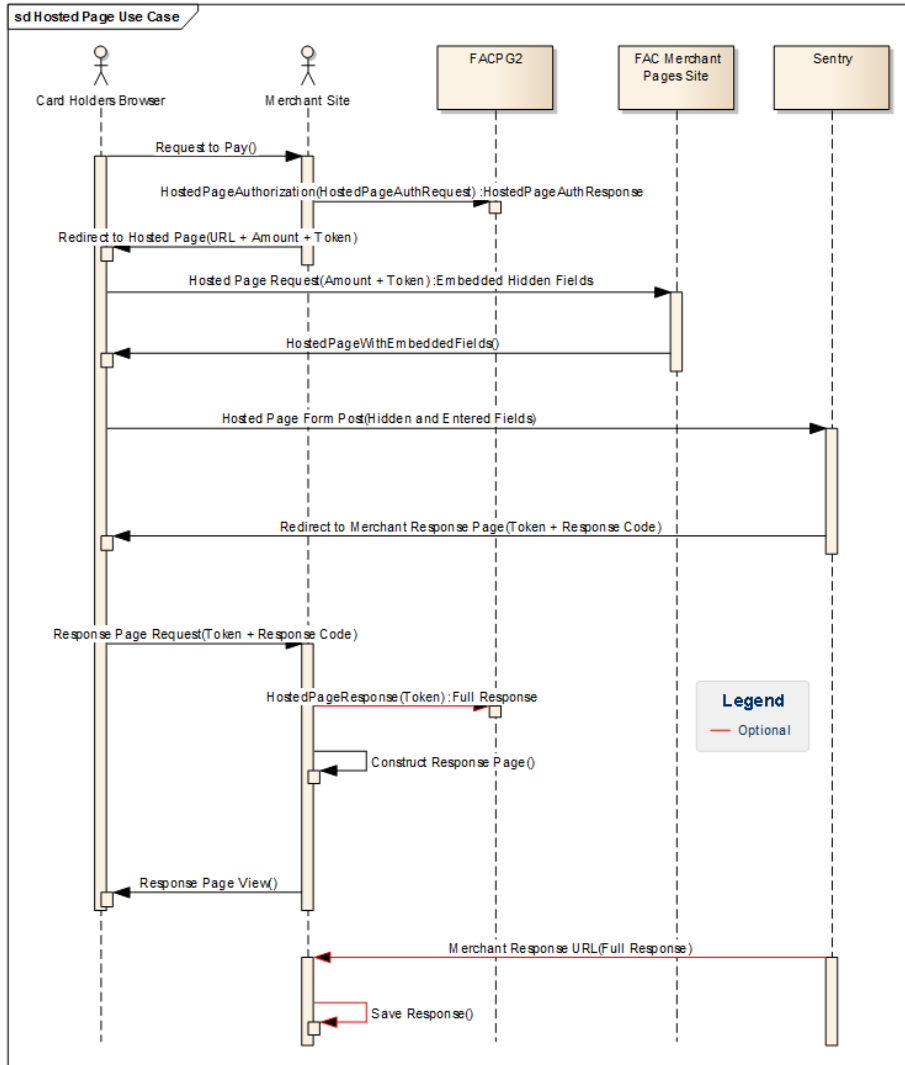
Once you are happy with your edits, it is important that you select the “Save HTML File” highlighted link to the left of the editor. If you do not do this, you will lose all your HTML edits.

Once the page is saved and you are happy to release the page, select the “Publish to Gateway” link. This will validate the page and may fail if it finds issues. This will usually be a misspelt input name or id and you may need to edit and re-publish. Once published, you are free to use this page as a Hosted Page on your e-commerce transactions. How you integrate this page is covered in the next section.

## Integrating your Hosted Payment Page to your System

### Understanding the Integration Process

As described in the [overview](#), the integration of a Hosted Page is not trivial. This is to ensure the use of the page is securely validated.



The following flow describes the interaction among all the parties during checkout

1. Customer requests checkout on Merchant's Page
2. On checkout the Merchant makes a SOAP Call to **HostedPageAuthorize** method(on FAC's payment gateway) passing transaction and merchant data as values on **HostedPageAuthorizationRequest**
3. The **HostedPageAuthorize** method returns a **HostedPageAuthorizationResponse** containing the HPP security Token that must be stored on the merchant site for later use
4. The Merchant requests the Hosted Page by calling a predetermined URL using the HPP Security Token obtained in the previous step as a parameter

- The URL would be something like:  
<https://ecm.firstatlanticcommerce.com/MerchantPages/PageSet/PageName/sSgK-E0cuEChgUF1xcPWXQ2>
  - **PageSet** and **PageName** are defined by the merchant at the time of design of the Hosted Payment Page
  - The single use token should be passed at the end of the URL as shown above: **sSgK-E0cuEChgUF1xcPWXQ2**
5. The Hosted Page is loaded and presented to the Cardholder
  6. Once the Cardholder fills in all the required payment information (credit card number, expiry, etc.), the Cardholder posts the page (by clicking the submit button) to FAC
  7. FAC sends the authorization to Interchange (card networks) and when done calls the Merchant's response URL provided in step 2) with the result of the authorization.
  8. The Merchant makes a SOAP call to **HostedPageResults** to retrieve full details about the authorization result. This is only available for the lifetime of the HPP security token (SingleUseToken) which is around 5 minutes.

## Available Interfaces

There are two interfaces that can be used to integrate with FAC: **SOAP** and **XML**, more details about these are explained below.

### Integrating with the HostedPage SOAP interface

- This section describes the interfaces to the two SOAP methods connected with Hosted Page execution, **HostedPageAuthorize** and **HostedPageResults**.
- If you do not have access to a SOAP client library or Service Proxy, we have XML handlers that can accept plain **XML posted directly to a URL**. See the section further below [Integrating with the HostedPage XML POST Method](#) for details.
- For general information on integrating with our Payment Gateway v2, please refer to our Guide "[First Atlantic Commerce Payment Gateway 2 Integration Guide for Developers](#)". This guide explains Authorization and 3DS Authentication in more detail and has definitions for Response codes and other useful information for integration.

Service Operation (SOAP)	Description	Request Data Type	Response Data Type
<b>HostedPage Authorize</b>	Used to register for using a Hosted Page for a transaction. Returns a single-use Token to be used in the URL to the hosted page	HostedPage Authorization Request	HostedPage Authorization Response
<b>HostedPage Results</b>	Allows the retrieval of the Full Authorize Response data directly from the FAC Gateway.	HostedPage Results Request	HostedPage Results Response

## HostedPageAuthorize Operation

This operation is used to register for using a Hosted Page for a transaction. The operation returns a single-use Token to be used in constructing the URL to the hosted page.

### Example:

```
HostedPageAuthorizeResponse := HostedPageAuthorize(HostedPageAuthorizeRequest)
```

## HostedPageAuthorizationRequest

- This request object needs to get passed when calling the **HostedPageAuthorize** method
- It contains among other things the **CardHolderResponseURL** which corresponds to the URL where FAC will send the response once the transaction has been processed (authorized or declined)
- This also contains three main sections **RecurringDetails**, **ThreeDSecureDetails** and **TransactionDetails**.
- Please note the Card Details, Billing Details and Shipping Details are not present in this request since they will be sent at a later point

### Detailed Field Descriptions

Section	Field	Format	Presence	Value
<b>Main Message Body</b>	CardHolderResponseURL	AN(150)	R	Required for Response to Cardholder. Should be the response Page. Cardholder is redirected here after transaction completion.
<b>TransactionDetails</b>	AcquirerId	N(11)	R	ALWAYS "464748"
	MerchantId	N(15)	R	Merchant ID provided by FAC
	OrderNumber	AN(150)	R	A unique identifier assigned by the merchant for the transaction. Must be unique.
	TransactionCode	N(4)	R	The transaction code is a numeric value that allows any combinations of the flags listed below to be included with the transaction request by summing their corresponding value. For example, to include AVS in the transaction and to tokenize the card number, assign the sum of the corresponding values 1 and 128 to the transaction code. The valid codes for an Authorization request are:  0 - Standard basic Authorization 1 - Include an AVS check in the transaction 2 - Flag as a \$0 AVS verification only transaction 4 - Transaction has been previously 3D Secure Authenticated the 3D Secure results will be included in the transaction. 8 - Flag as a single pass transaction (Authorization and Capture as a single transaction) 64 – 3DS Only (N/A) 128 – Tokenize PAN

				256 – Hosted Page Auth + 3DS
	Amount	N(12)	R	Total amount of purchase. Note: The purchase amount must be presented as a character string that is 12 characters long. (i.e. \$12.00 should be provided as “000000001200”). Can be excluded if PurchaseAmt field is on the form.
	Currency	N(3)	R	The purchase currency ISO 4217 numeric currency code (ex: USD = 840). See Appendix
	CurrencyExponent	N(1)	R	The number of digits after the decimal point in the purchase amount (i.e. \$12.00 = 2)
	SignatureMethod	AN(4)	R	ALWAYS “SHA1”
	Signature	AN(28)	R	See Appendix 8 for information on the Signature and code snippets for on creating the SHA1 signature hash
	IPAddress	AN(15)	C	Cardholder’s IP Address
	CustomData	AN(n)	O	Reserved for future use
	CustomerReference	AN(256)	O	Used with Tokenization Request to associate a Token with a Customer
	StartDate		C	Cardholder’s credit/debit card start date formatted as (MMYY format). Required for some debit cards.
<b>ThreeDSecureDetails</b>	EClIndicator	N(2)	C	This value is only needed for pre-authenticated 3D Secure transactions and the transaction code must include the value 4 in its summed value. Possible values include:  Visa: “05” - Full 3D Secure authentication “06” - Issuer and/or cardholder are not enrolled for 3D Secure “07” - 3D Secure authentication attempt failed (numerous possible reasons)  MasterCard: “01” - Issuer and/or cardholder are not enrolled for 3D Secure “02” - Full 3D Secure authentication
	AuthenticationResult	A(1)	C	This value is only needed for pre-authenticated 3D Secure transactions and the transaction code must include the value 4 in its summed value. Possible values include:  “A” = An attempt at authentication was performed (EClIndicator: V=06, MC=01) “N” = Authentication attempt not supported (EClIndicator: V=06, MC=01) “U” = Unable to authenticate (EClIndicator: V=07, MC=01) “Y” = Authentication attempted and succeeded (EClIndicator: V=05, MC=02)
	TransactionStain	AN(28)	O	A hashed version of the Transaction ID (XID). The XID is a unique tracking number assigned to the authentication request that prevents replay or resubmission of the same transaction.
	CAVV	AN(28)	O	This is a cryptographic value derived by the issuer during payment authentication that provides evidence of the results of the payment authentication process. Note that for MasterCard this field is referred to as UCAF but the field name will still be CAVV.
<b>RecurringDetails</b>	IsRecurring	A(5)	C	Set to “True” or “False” depending on whether a Recurring transaction is required.
	ExecutionDate	AN(28)	C	When to execute the first/initial authorization. Example Date format is “YYYY-MM-DD”
	Frequency	A(1)	C	Flag to determine how frequently to execute the recurring authorization.

				Possible values are: "D" – Daily "W" – Weekly "F" – Fortnightly/Every 2 weeks "M" – Monthly "E" – Bi-Monthly "Q" – Quarterly "Y" – Yearly
	NumberOfRecurrences	N(3)	C	How many times in total to execute. For example, Frequency = "D", NumberOfRecurrences = 7 will execute every day for a week.

Example:

```

<HostedPageAuthorizationRequest>
  <CardHolderResponseURL>https://merchant/response/page.php</CardHolderResponseURL>
  <RecurringDetails>
    <ExecutionDate/>
    <Frequency/>
    <IsRecurring>false</IsRecurring>
    <NumberOfRecurrences>0</NumberOfRecurrences>
  </RecurringDetails>
  <ThreeDSecureDetails>
    <AuthenticationResult/>
    <CAVV/>
    <ECIIndicator/>
    <TransactionStain/>
  </ThreeDSecureDetails>
  <TransactionDetails>
    <AcquirerId>464748</AcquirerId>
    <Amount>000000001000</Amount>
    <Currency>840</Currency>
    <CurrencyExponent>2</CurrencyExponent>
    <IPAddress>127.1.1.1</IPAddress>
    <MerchantId>99999999</MerchantId>
    <OrderNumber>MO918921929182981</OrderNumber>
    <Signature>612893689213897hg2whg1r</Signature>
    <SignatureMethod>SHA1</SignatureMethod>
    <TransactionCode>0</TransactionCode>
    <CustomerReference/>
  </TransactionDetails>
</HostedPageAuthorizationRequest>

```

## HostedPageAuthorizationResponse

- The **HostedPageAuthorizationResponse** contains FAC's validation result of the previously sent HostedPageAuthorizationRequest.
- The validation result is indicated through the **ResponseCode** field which value is described in the table below and a description
- If the validation is successful, a **SingleUseToken** will be sent.

### Detailed Field Descriptions

Section	Field	Format	Value
<b>Main Message Body</b>	ResponseCode	N(2)	0 – Preprocessing Successful 01 – Request is empty 02 –Missing transaction details 03 – Missing parameters 04 - Amount Invalid 05 – Preprocessing System Error 06 – Missing CardholderResponseURL
	ResponseCode Description	AN(150)	Example: “Authorized” when response = 0, otherwise the reason for failure.
	SingleUseToken	AN(40)	Token GUID when Response = 0, else null

#### Example:

```

<HostedPageAuthorizationResponse>
  <ResponseCode>0</ResponseCode>
  <ResponseCodeDescription>Authorized</ResponseCodeDescription>
  <SingleUseToken>8991287hggaftrwqfg55</SingleUseToken>
</HostedPageAuthorizationResponse>
  
```

## Integrating with the HostedPage XML POST Method

Please note that if you choose to integrate to FAC's Hosted Payment Page system with **XML POST**, there are some differences in the integration, versus SOAP.

Service Operation (XML)	Description	Request Data Type	Response Data Type
<b>HostedPage Preprocess</b>	This XML POST operation is used to register for using a Hosted Page for a transaction. Returns a single-use Token to be used in the URL to the hosted page	HostedPage Preprocess Request	HostedPage Authorization Response
<b>HostedPage Results</b>	Allows the retrieval of the Full Authorize Response data directly from the FAC Gateway.	HostedPage Results Request	HostedPage Results Response

Example:

The URL to call is: <https://ecm.firstatlanticcommerce.com/PGServiceXML/HostedPagePreprocess>

### HostedPagePreprocessRequest

- This request object needs to get passed when calling the **HostedPagePreprocess** method
- Similarly to the previously described HostedPageAuthorizationRequest, the **HostedPagePreprocessRequest** contains the **CardHolderResponseURL** which corresponds to the URL where FAC will send the response once the transaction has been processed (authorized or declined)
- This also contains three main sections **RecurringDetails**, **ThreeDSecureDetails** and **TransactionDetails**.
- Please note the Card Details, Billing Details and Shipping Details are not present in this request since they will be sent at a later point

```
<HostedPagePreprocessRequest xmlns="http://schemas.firstatlanticcommerce.com/gateway/data"
    xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <CardHolderResponseURL>https://merchant/response/page.php</CardHolderResponseURL>
  <RecurringDetails>
    <ExecutionDate/>
    <Frequency/>
    <IsRecurring>>false</IsRecurring>
    <NumberOfRecurrences>0</NumberOfRecurrences>
  </RecurringDetails>
```



```
<ThreeDSecureDetails>  
  <AuthenticationResult/>  
  <CAVV/>  
  <ECIIndicator/>  
  <TransactionStain/>  
</ThreeDSecureDetails>  
<TransactionDetails>  
  <AcquirerId>464748</AcquirerId>  
  <Amount>000000001000</Amount>  
  <Currency>840</Currency>  
  < CurrencyExponent>2</CurrencyExponent>  
  <IPAddress>127.1.1.1</IPAddress>  
  <MerchantId>99999999</MerchantId>  
  <OrderNumber>M0918921929182981</OrderNumber>  
  <Signature>612893689213897hg2whg1r</Signature>  
  <SignatureMethod>SHA1</SignatureMethod>  
  <TransactionCode>0</TransactionCode>  
  <CustomerReference/>  
</TransactionDetails>  
</HostedPagePreprocessRequest>
```

## HostedPageResults method

- The **HostedPageResults** operation is there for the Merchant to use to retrieve the full response data. There are two interfaces that can be used to integrate the **HostedPageResults**: SOAP and XML, more details about these are explained below.
- It is only possible to use this operation during the lifetime of the transaction as limited by the single use token expiry time (currently 5 minutes). The **HostedPageResults** operation can only be used once.
- The semantics of the operation are as follows:
  - HostedPageResultsResponse := **HostedPageResults**(token As String)
- The input parameter is the single-use Token returned from the **HostedPageAuthorize** (SOAP) or **HostedPagePreprocess** (XML POST) operation.

### Important Note:

The HostedPageResults method should not be called more than one time; otherwise, it will return an error.

### Example:

#### HostedPageResults – sample SOAP message

Here is a sample **SOAP** message for **HostedPageResults** operation:

```
<?xml version="1.0"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <HostedPageResults xmlns="http://schemas.firstatlanticcommerce.com/gateway">
      <key>_JBfLQJNiEmFBtnF3AfoeQ2</key>
    </HostedPageResults>
  </s:Body>
</s:Envelope>
```

#### HostedPageResults – sample XML message

Here is a sample **XML-POST** message for HostedPageResults operation:

```
<string xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.firstatlanticcommerce.com/gateway/data">
  _JBfLQJNiEmFBtnF3AfoeQ2
</string>
```

There should be no “soap” or “envelope” or “Body” in the XML-POST request - just one “string” tag carrying the **single-use token**. The **namespace** is important.

## HostedPageResultsResponse

- The response is made up of the **AuthorizeResponse** fields and **ThreeDSResponse** fields. These are described in full in our standard integration guide.

```

<HostedPageResultsResponse>
  <AuthResponse>
    <BillingDetails>
      <BillToAddress>123 The Street</BillToAddress>
      <BillToCity>Townsville</BillToCity>
      <BillToCountry>840</BillToCountry>
      <BillToFirstName>John</BillToFirstName>
      <BillToLastName>Doe</BillToLastName>
      <BillToState>PA</BillToState>
      <BillToCounty>Town County</BillToCounty>
    </BillingDetails>
    <CreditCardTransactionResults>
      <AVSResult>Exact Match</AVSResult>
      <AuthCode>00</AuthCode>
      <OriginalResponseCode>1</OriginalResponseCode>
      <PaddedCardNumber>XXXXXXXXXXXX7876</PaddedCardNumber>
      <ReasonCode>1</ReasonCode>
    </CreditCardTransactionResults>
    <IPGeoLocationResults>
      <City>Townsville</City>
      <CountryShort>USA</CountryShort>
    </IPGeoLocationResults>
    <MerchantId>99999999</MerchantId>
    <OrderNumber>123434566777</OrderNumber>
    <Signature>678263gghjwgs655632</Signature>
    <SignatureMethod>SHA1</SignatureMethod>
  </AuthResponse>
  <ThreeDSResponse/>
</HostedPageResultsResponse>

```

## Letting the Cardholder Enter the Amount

The standard use-case for Hosted Payment Page is to let the user enter only the card details as these are the fields Merchants do not want to save on their systems. However, it is possible to implement a Hosted Payment Page with a payment amount (named **PurchaseAmt**) on the form.

There are rules for including this on the form:

- The **PurchaseAmt** format must comply with the currency's number of decimal places and be a properly formed decimal number.

- Do not include currency symbols in **PurchaseAmt**.
- If you want to give the ability to choose a currency then you must include an input with the **PurchaseCurrency** field and the **PurchaseCurrencyExponent** field on the Hosted Page.
- The signature you create and pass to the **HostedPageAuthorize** method will not be the same. Because you are putting the **PurchaseAmt** field on the form, the signature must not include the data of the amount or currency fields.
- Note that the "CheckoutWithValidation" template is not designed for use in a payment page where the cardholder enters their own amount.

## Kount Fraud Control Integration

Fraud Control integration has been included in Hosted Page functionality but a small amount of extra code is required to enable it. However, it is a lot less work than would be required in your own pages as we have done some of the work for you. Fraud requires that you integrate the page to their Data Collector.

Here is an example of how this can be done with Hosted Payment Page:

1. First, get your FAC ID enabled for Kount Fraud Integration.
2. Then, include these script imports in the <head> element of your page.

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.8.1/jquery.min.js"></script>
```

```
<script src="//ajax.aspnetcdn.com/ajax/jquery.validate/1.9/jquery.validate.min.js"></script>
```

3. Finally, include this script towards the end of the <form> element in the Hosted Page :

```
$(document).ready(function(){
    var uuid = $("#HPPKey").val();
    var merchantId = "24XXXX";
    var frame = '<iframe width=1 height=1 frameborder=0 scrolling=no src="{0}"></iframe>';
    var input = '<input type="hidden" name="SessionID" id="SessionID" value="{0}" />';
    var logoHtml = "/MerchantPages/logo.aspx?m=" + merchantId + "&s=" + uuid;
    var logoGif = "/MerchantPages/logo_gif.aspx?m=" + merchantId + "&s=" + uuid;
    var frameHtml = $.validator.format(frame, logoHtml, logoGif);
    var inputHtml = $.validator.format(input, uuid);
    $('#FrmCheckout').append(inputHtml);
    $(document.body).append(frameHtml);
});
```

- What does this do exactly?
  - To integrate to Kount you must include an iframe in your page with references to images that are in fact scripts.

- These images redirect the cardholder to the Kount site and back again. During this process, certain data on the cardholder's browser is collected by a process on the Kount site called the "Data Collector".
- At the same time, a unique identifier is passed into Kount to create an ID for the data. This ID is then passed to FAC in the Authorization data.
- The above script does this dynamically using JavaScript and JQuery. Note that the script requires your Merchant ID. This is your Kount Merchant ID, not your FAC Merchant ID.
- When you do an Authorization with Fraud enabled, the authorization data is passed to Kount via the FAC Gateway and Kount use this and the Data Collector data to provide a fraud check score is added to the response.
- You can then use this to decide if a transaction is fraudulent or not and act accordingly (by stopping shipment of goods for example).
- In Hosted Payment page, we have already included the "logo" image files on our Merchant Pages site and integrated the Authorization data to Kount, so all you need to do is add the code above to implement the iframe "Data Collector" functionality.
- The code above does the job and can be used as a template or as-is. It all depends on your platform; you might want to use PHP or Java for example.

For more detailed description of the Kount Fraud Control Integration, our support and business development staff will be able to help.

## Example of the complete flow (SOAP)

### 1. Merchant sends HPP Token Request on HostedPageAuthorizationRequest

Web service call to <https://ecm.firstatlanticcommerce.com/PGService/HostedPage.svc>

Body:

```
<HostedPageAuthorizationRequest>
  <CardHolderResponseURL>https://merchant/response/page.php</CardHolderResponseURL>
  <RecurringDetails>
    <ExecutionDate/>
    <Frequency/>
    <IsRecurring>false</IsRecurring>
    <NumberOfRecurrences>0</NumberOfRecurrences>
  </RecurringDetails>
  <ThreeDSecureDetails>
    <AuthenticationResult/>
    <CAVV/>
    <ECIIndicator/>
    <TransactionStain/>
  </ThreeDSecureDetails>
  <TransactionDetails>
    <AcquirerId>464748</AcquirerId>
    <Amount>000000001000</Amount>
    <Currency>840</Currency>
    <CurrencyExponent>2</CurrencyExponent>
    <IPAddress>127.1.1.1</IPAddress>
    <MerchantId>99999999</MerchantId>
    <OrderNumber>MO918921929182981</OrderNumber>
    <Signature>612893689213897hg2whg1r</Signature>
    <SignatureMethod>SHA1</SignatureMethod>
    <TransactionCode>0</TransactionCode>
    <CustomerReference/>
  </TransactionDetails>
</HostedPageAuthorizationRequest>
```

### 2. FAC returns token in HostedPageAuthorizationResponse

```
<HostedPageAuthorizationResponse>
  <ResponseCode>0</ResponseCode>
  <ResponseCodeDescription>Authorized</ResponseCodeDescription>
  <SingleUseToken>8991287hggaftrwqfg55</SingleUseToken>
</HostedPageAuthorizationResponse>
```

### 3. Merchant Requests the Hosted Page

<https://ecm.firstatlanticcommerce.com/MerchantPages/PageSet/PageName/8991287hggaftrwqfg55>

- **PageSet** needs to be adjusted
- **PageName** needs to be adjusted
- The single use token needs to be appended to the end of the URL, in this case it's: **8991287hggaftrwqfg55**

### 4. Cardholder fills out and submits payment page

- Page has a 5-minute lifespan
- Merchant should validate data before submitting to FAC

### 5. FAC Redirects the Cardholder to the provided Cardholder response URL

In this example, the merchant's cardholder response URL is: <https://merchant/response/page.php>

### 6. Merchant Retrieves Hosted Page Results via SOAP call

Web service SOAP calls goes to the following URL, providing the Single Use Token

<https://ecm.firstatlanticcommerce.com/PGService/HostedPageResults.svc>

The web service call should look like this (with the corresponding token):

```
<?xml version="1.0"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <HostedPageResults
xmlns="http://schemas.firstatlanticcommerce.com/gateway">
      <key>8991287hggaftrwqfg55</key>
    </HostedPageResults>
  </s:Body>
</s:Envelope>
```

## Code Snippets

- Code snippets from our Test Merchant Site are available on Appendix 6 and 7 (PHP and C# respectively). On each one of those appendices there are sample implementations for the following operations
  - HostedPageAuthorize
  - HostedPageResults
  - ComputeHash
- Please note the code samples should not be used as-is without adding sufficient error handling and production safeguards.

## Putting it All Together

We are happy to provide help and assistance as you implement your hosted page integration. Please contact FAC integration support at [support@fac.bm](mailto:support@fac.bm).



## Appendix 1 – Data Field Validation

Category	Input “id”/”name”	Format	Notes
<b>Card Details</b>	Amount	N(4-10) “#0.00”	Optional. For displaying of amount to user. If added to the form will be auto-populated with Amount passed in call to HostedPageAuthorize. Will not be processed by hosted page. If edited by the user, should be used to populate the (hidden) PurchaseAmt field in Currency unit format (see below).
	CardNo	N(16 – 19)	Mandatory. Max 16 for non Amex, 19 for Amex. <u>Numeric only</u>
	CardExpDate	N(4)	Mandatory. MMYT Format
	CardCVV2	N(3 - 4)	Conditional. May be required depending on processor. Usually 3 digits.
	IssueNumber	N(2)	Required for Debit Cards Only where applicable (e.g. UK Debit cards)
	StartDate	N(4)	MMYT Format. Debit Cards only and is usually required if Issue number is not mandatory.
	PurchaseAmt	N(12) or N(4-10) Decimal “#0.00” format.	Optional. Transaction Amount in Currency units or Decimal format. Currency unit format is padded left with Zeros. E.g.: 10.00 = 000000001000. If included in Form will override what has been passed into HostedPageAuthorize. Decimal format (“#0.00”) will be converted to Currency Unit format when hosted page is posted.
	PurchaseCurrency	N(3)	Optional. ISO Numeric Currency code. E.g. 840 for US Dollars
	PurchaseCurrency Exponent	N(1)	Optional. Number of decimal places. Usually 2 for most currencies
	SessionId	AN(30)	Optional. A Unique ID for Kount Fraud Control Processing. See the <a href="#">Fraud Control</a> Section for more information.
<b>Billing Details (all optional)</b>	BillToFirstName	AN(30)	
	BillToMiddleName	AN(30)	
	BillToLastName	AN(30)	
	BillToAddress1	AN(50)	
	BillToAddress2	AN(50)	
	BillToCity	AN(30)	
	BillToState	AN(2)	State Code. Max A(2) if USA only. You could hide this field and use a drop down to set the value. Max A(3) for non-USA.
	BillToCounty	AN(15)	County Name
	BillToPostCode	AN(10)	Post Code or Zip Code. Strictly Alphanumeric only.

	BillToCountry	N(3)	Country Code. Hide this field and use a drop down to set the value.
	BillToTelephone		
	BillToEmail		
	BillToFax	AN(30)	
	BillToMobile	AN(30)	
<b>Shipping Details (all optional)</b>	ShipToFirstName	AN(30)	
	ShipToMiddleName	AN(30)	
	ShipToLastName	AN(30)	
	ShipToAddress1	AN(50)	
	ShipToAddress2	AN(50)	
	ShipToCity	AN(30)	
	ShipToState	A(3)	State Code. Max A(2) if USA only. You could hide this field and use a drop down to set the value. Max A(3) for non-USA.
	ShipToCounty	AN(15)	County Name
	ShipToPostCode	AN(10)	Strictly Alphanumeric only.
	ShipToCountry	N(3)	Country Code. Hide this field and use a drop down to set the value.
	ShipToTelephone	AN(30)	
	ShipToEmail	AN(50)	
	ShipToFax	AN(30)	
ShipToMobile	AN(30)		

## Appendix 2 – AVS Field Requirements

The following table lays out the field requirements for Address Verification checks. These requirements may vary somewhat by processor; however, this is the standard across the board:

Field Name	Required or Optional Field	Allowable Character Format	Character Limit	Special Considerations - Restrictions
BillToFirstName	Optional	alphanumeric (a-z, A-Z, 0-9)	up to 30 maximum	<b>No special characters, no accents, no special symbols</b> (Example: æ é à ñ * + & ; ; ) and <b>best to avoid all symbols</b> but basic punctuation is acceptable such as periods and dashes ( . and - )
BillToLastName	Optional	alphanumeric (a-z, A-Z, 0-9)	up to 30 maximum	<b>No special characters, no accents, no special symbols</b> (Example: æ é à ñ * + & ; ; ) and <b>best to avoid all symbols</b> but basic punctuation is acceptable such as periods and dashes ( . and - )
BillToAddress1	<b>Required</b>	alphanumeric (a-z, A-Z, 0-9)	up to 50 maximum	<b>No special characters, no accents, no special symbols</b> (Example: æ é à ñ * + & ; ; ) and <b>best to avoid all symbols</b> but basic punctuation is acceptable such as periods and dashes ( . and - )
BillToAddress2	Optional	alphanumeric (a-z, A-Z, 0-9)	up to 50 maximum	<b>No special characters, no accents, no special symbols</b> (Example: æ é à ñ * + & ; ; ) and <b>best to avoid all symbols</b> but basic punctuation is acceptable such as periods and dashes ( . and - )
BillToCity	Optional	alphanumeric (a-z, A-Z, 0-9)	up to 30 maximum	<b>No special characters, no accents, no symbols</b> ( Example : æ é à ñ * + & ; ; )
BillToState	Optional	alphanumeric (a-z, A-Z, 0-9)	2 minimum to 5 maximum	<b>Ideally use the 2 alpha character ISO State Code OR a 3 numeric digit ISO country code if necessary for customers where ‘State’ would not be applicable OR this field can be omitted.</b> Strictly Alpha numeric. No special characters, no accents, no spaces, no symbols
BillToPostCode	<b>Required</b>	alphanumeric (a-z, A-Z, 0-9)	up to 10 maximum	<b>Strictly Alphanumeric only - No special characters, no accents, no spaces, no dashes...etc.</b>
BillToCountry	Optional	Numeric (0-9)	must be 3 digits	This must be 3 digit country code
BillToTelephone	Optional	Numeric (0-9)	up to 20 maximum	Strictly numeric. No special characters, no accents, no spaces, no symbols
BillToEmail	Optional	Alphanumeric (a-z, A-Z, 0-9)	up to 50 maximum	Standard email format (name@domain.com). Basic punctuation is acceptable such as periods and dashes ( . and - ). <b>No special characters, no accents, no spaces</b>

### Appendix 3 – ISO 3166 US State Codes

The parameter BillToState is only valid for U.S. based addresses. The allowable values for this parameter are as follows:

Abbreviation	State Name	Abbreviation	State Name
<b>AK</b>	Alaska	<b>MS</b>	Mississippi
<b>AL</b>	Alabama	<b>MT</b>	Montana
<b>AR</b>	Arkansas	<b>NC</b>	North Carolina
<b>AS</b>	American Samoa	<b>ND</b>	North Dakota
<b>AZ</b>	Arizona	<b>NE</b>	Nebraska
<b>CA</b>	California	<b>NH</b>	New Hampshire
<b>CO</b>	Colorado	<b>NJ</b>	New Jersey
<b>CT</b>	Connecticut	<b>NM</b>	New Mexico
<b>DC</b>	District of Columbia	<b>NV</b>	Nevada
<b>DE</b>	Delaware	<b>NY</b>	New York
<b>FL</b>	Florida	<b>OH</b>	Ohio
<b>FM</b>	Federate States Of Micronesia	<b>OK</b>	Oklahoma
<b>GA</b>	Georgia	<b>OR</b>	Oregon
<b>GU</b>	Guam	<b>PA</b>	Pennsylvania
<b>HI</b>	Hawaii	<b>PR</b>	Puerto Rico
<b>IA</b>	Iowa	<b>PW</b>	Palau
<b>ID</b>	Idaho	<b>RI</b>	Rhode Island
<b>IL</b>	Illinois	<b>SC</b>	South Carolina
<b>IN</b>	Indiana	<b>SD</b>	South Dakota
<b>KS</b>	Kansas	<b>TN</b>	Tennessee
<b>KY</b>	Kentucky	<b>TX</b>	Texas
<b>LA</b>	Louisiana	<b>UT</b>	Utah
<b>MA</b>	Massachusetts	<b>VA</b>	Virginia
<b>MD</b>	Maryland	<b>VI</b>	U.S. Virgin Islands
<b>ME</b>	Maine	<b>VT</b>	Vermont
<b>MH</b>	Marshall Islands	<b>WA</b>	Washington
<b>MI</b>	Michigan	<b>WI</b>	Wisconsin
<b>MN</b>	Minnesota	<b>WV</b>	West Virginia
<b>MO</b>	Missouri	<b>WY</b>	Wyoming
<b>MP</b>	Northern Mariana Islands		

## Appendix 4 – Test Cards for FAC Test Environment

The following is a list of test cards you can use to receive specific responses for testing purposes.

It is important to note the following:

- These test cards do not apply to \$0 AVS-Only testing. For this type of transaction, use real credit cards so as to get valid AVSResult data.
- Any valid expiry date and any 3 digit CVV2 value will work for these test cards
- Note: “Normal Approval” means ResponseCode=1, ReasonCode=1 and “Normal Decline” means ResponseCode=2, ReasonCode=2 in the web responses returned for Auth only and Auth/Capture transactions.
- All card numbers not listed above are defaulted to Normal Approval.
- For every approved transaction, you will receive the same ‘dummy’ authorization ID of 123456.
- These cards are **only to be used in the test environment** (ecm.firstatlanticcommerce.com). Once you are on the production platform, **live** cards must be used.

### Visa

Card Number	Response
4111111111111111	Normal Approval, CVV2Result=M
4111111111112222	Normal Approval, CVV2Result=N
4333333333332222	Normal Approval, CVV2Result=U
4444444444442222	Normal Approval, CVV2Result=P
4555555555552222	Normal Approval, CVV2Result=S
4666666666662222	Normal Decline, OriginalResponseCode=05, CVV2Result=N
4111111111113333	Normal Decline, OriginalResponseCode=05
4111111111114444	Normal Approval, AVSResult=M
4111111111115555	Normal Approval, AVSResult=A
4111111111116666	Normal Approval, AVSResult=Z
4111111111117777	Normal Approval, AVSResult=N
4111111111118888	Normal Approval, AVSResult=G
4111111111119999	Normal Decline, OriginalResponseCode=98
4111111111110000	Normal Decline, OriginalResponseCode=91
4222222222222222	Normal Approval, CVV2Result=M, AVSResult=N

**MasterCard**

Card Number	Response
5111111111111111	Normal Approval, CVV2Result=M
5111111111112222	Normal Approval, CVV2Result=N
5333333333332222	Normal Approval, CVV2Result=U
5444444444442222	Normal Approval, CVV2Result=P
5555555555552222	Normal Approval, CVV2Result=S
5555666666662222	Normal Decline, OriginalResponseCode=05, CVV2Result=N
5111111111113333	Normal Decline, OriginalResponseCode=05
5111111111114444	Normal Approval, AVSResult=Y
5111111111115555	Normal Approval, AVSResult=A
5111111111116666	Normal Approval, CVV2Result=M, AVSResult=Z
5111111111117777	Normal Approval, CVV2Result=M, AVSResult=N
5111111111118888	Normal Approval, CVV2Result=N, AVSResult=U
5111111111119999	Normal Decline, OriginalResponseCode=98
5111111111110000	Normal Decline, OriginalResponseCode=91
5222222222222222	Normal Approval, CVV2Result=N, AVSResult=U

## Appendix 5 – Response Codes

### Appendix 5.1 – System Response Codes and Reason Codes

The ResponseCode, ReasonCode and ReasonCodeDescription fields of the AuthorizeResponse and TransactionStatusResponse messages can hold the following code combinations.

**NOTE:** If you are using Fraud Control services there will be additional potential Reason Codes and Reason Code Descriptions than described below (refer to [Fraud Response and Reason Codes](#)).

#### ResponseCode Values

Response Code	Description
1	Approved
2	Declined
3	Error

#### Reason Code for “Approved” Response Code (1)

Reason Code	Reason Text (ReasonCodeDescription)	Note
1	Transaction is approved.	Normal Approval.

#### Reason Codes for “Decline” Response Code (2)

Reason Code	Reason Text (ReasonCodeDescription)	Note
2	Transaction is declined.	Normal Decline.
3	Transaction is declined.	Referral. Call for further details on this transaction.
4	Transaction is declined.	Pick up card (if possible) or report to authorities.
35	Unable to process your request. Please try again later.	Merchant exceeds allowed limit.
38	Transaction processing terminated. Please try again later.	Transaction is not permitted to merchant.
39	Issuer or switch not available. Please try again later	Issuing bank or switch not available. Transaction has timed-out.

### Reason Codes for “Error” Response Code (3)

Reason Code	Reason Text (ReasonCodeDescription)	Note
5	Connection not secured.	Connection was not secured.
6	HTTP Method not POST.	HTTP Method not POST.
7	“Field” is missing.	Named field is missing.
8	“Field” format is invalid.	Named field format is invalid.
10	Invalid Merchant.	Not such merchant.
11	Failed Authentication (Signature computed incorrectly).	Merchant was found but computed signature does not match one included in the request.
12	Merchant is inactive.	Merchant is not enabled for processing.
14	Merchant is not allowed to process this currency.	Currency supplied is not permitted.
15	Merchant settings are not valid.	Merchant record is not correctly setup in the system.
16	Unable to process transaction.	Unable to authenticate merchant now. Try later.
36	Credit Cardholder canceled the request.	Credit Cardholder canceled the request.
37	Card Entry Retry Count exited allowed limit.	Card Entry Retry Count exited allowed limit.
40	Duplicate Order Not Allowed	Merchant order identification numbers must be unique
42	Illegal Operation by Card Holder. Check Order Status.	Cardholder Pressed the back button while the transaction was processing. Check the status of that order.
60	Duplicate Order Not Allowed.	A transaction for the same card number and same amount was processed previously and thus this transaction has been blocked (optional setting)
90	General Error during processing. Please try again later.	An unexpected error occurred in the system.
98	System is temporarily down. Try later.	System is temporarily down. Try later.
401	Cycle interrupted by the user or client/browser connection not available.	Client Browser connection not available or card holder referred in the process (Back/F5).
994	FACPGWS BeginTransactionStatus Failure	Error while attempting to run the TransactionStatus Operation Try again. If this persists, contact FAC support at support@fac.bm for assistance.



<b>995</b>	FACPGWS EndTransactionStatus Failure	Error while attempting to run TransactionStatus Operation. Try again. If this persists, contact FAC support at support@fac.bm for assistance.
<b>996</b>	Not a web-based transaction	The transaction for which you are requesting the response data is not a web-based transaction. It is a MOTO transaction and as such there is no web-based response data for this transaction.
<b>997</b>	FACPGAppWS Failure	Error while attempting to run the TransactionStatus Operation. Try again. If this persists, contact FAC support at support@fac.bm for assistance.
<b>998</b>	Missing Parameter	One of the parameters required by the TransactionStatus Operation was not supplied.
<b>999</b>	No Response	There is no response data for the Order ID provided.
<b>1001</b>	FACPGWS Invalid Protocol. Only HTTPS Allowed	The request was sent via HTTP not HTTPS.
<b>1002</b>	Missing Parameter(s)	One or more of the required parameters is missing in the web method you have called.
<b>1003</b>	Invalid Parameter Settings	Both "AVS Only" and "PreAuthenticated" flags have been included in the TransactionCode when calling the Authorize web method. This is not allowed.
<b>1004</b>	Invalid Amount. Not 12 characters in length	Amount must be exactly 12 characters in length, right-aligned, left-padded with zeros. For example, \$12.00 = 000000001200
<b>1010</b>	FACPGWS Authorize HTTP Response Not OK	Error while attempting to run the Authorize Operation. Try again. If this persists, contact FAC support at support@fac.bm for assistance.
<b>1020</b>	FACPGWS Authorize Failure	Error while attempting to run the Authorize web method. Try again. If this persists, contact FAC support at support@fac.bm for assistance.
<b>1030</b>	FACPG BeginCRRError	Error while attempting to run either the Capture, Reversal or Refund web methods. Try again. If this persists, contact FAC support at support@fac.bm for assistance.
<b>1031</b>	FACPG EndCRRError	Error while attempting to run either the Capture, Reversal or Refund web methods. Try again. If this persists, contact FAC support at support@fac.bm for assistance.

## Appendix 5.2 – ISO Response Codes

The response codes for an Authorization are returned in the OriginalResponseCode field of the Response. See also AuthorizeResponse, TransactionModificationResponse, or TransactionStatusResponse (see main FACPG2 Integration Guide). They are specific to the Card Issuer.

### VISA

Response Code & Description		Response Code & Description	
00	Approved	53	No savings account
01	Refer to issuer	54	Expired card
02	Refer to issuer (special)	55	Incorrect PIN
03	Invalid merchant	56	No card record
04	Pick-up card	57	Transaction not permitted to card
05	Do not honor	58	Transaction not permitted to card
06	Error	59	Suspected fraud
07	Pick-up card (special)	60	Card acceptor contact acquirer
08	Honor with identification	61	Exceeds withdrawal limit
09	Request in progress	62	Restricted card
10	Approved for partial amount	63	Security violation
11	VIP Approval	64	Original amount incorrect
12	Invalid transaction	65	Activity count exceeded
13	Invalid amount	66	Card acceptor call acquirer
14	Card number does not exist	67	Card pick up at ATM
15	No such issuer	68	Response received too late
16	Approved, update track 3	75	Too many wrong PIN tries
17	Customer cancellation	76	Previous message not found
18	Customer dispute	77	Data does not match original message
19	Re-enter transaction	80	Invalid date
20	Invalid response	81	Cryptographic error in PIN
21	No action taken (no match)	82	Incorrect CVV
22	Suspected malfunction	83	Unable to verify PIN
23	Unacceptable transaction fee	84	Invalid authorization life cycle
24	File update not supported by receiver	85	No reason to decline
25	Unable to locate record	86	PIN validation not possible
26	Duplicate file update record	88	Cryptographic failure
27	File update field edit error	89	Authentication failure
28	File temporarily unavailable	90	Cutoff is in process
29	File update not successful	91	Issuer or switch inoperative
30	Format error	92	No routing path
31	Issuer sign-off	93	Violation of law

32	Completed partially	94	Duplicate transmission
33	Expired card	95	Reconcile error
34	Suspected fraud	96	System malfunction
35	Card acceptor contact acquirer	97	Format Error
36	Restricted card	98	Host Unreachable
37	Card acceptor call acquirer	99	Errored Transaction
38	Allowable PIN tries exceeded	N0	Force STIP
39	No credit account	N3	Cash Service Not Available
40	Function not supported	N4	Cash request exceeds issuer limit
41	Pick-up card (lost card)	N7	Decline for CVV2 failure
42	No universal account	P2	Invalid biller information
43	Pick-up card (stolen card)	P5	PIN Change Unblock Declined
44	No investment account	P6	Unsafe PIN
51	Not sufficient funds	XA	Forward to issuer
52	No checking account	XD	Forward to issuer

### *MasterCard*

Response Code & Description		Response Code & Description	
00	Approved	44	No investment account
01	Refer to issuer	51	Not sufficient funds
02	Refer to issuer (special)	52	No checking account
03	Invalid merchant	53	No savings account
04	Pick-up card	54	Expired card
05	Do not honor	55	Incorrect PIN
06	Error	56	No card record
07	Pick-up card (special)	57	Transaction not permitted to card
08	Honor with identification	58	Transaction not permitted to card
09	Request in progress	59	Suspected fraud
10	Approved for partial amount	60	Card acceptor contact acquirer
11	VIP Approval	61	Exceeds withdrawal limit
12	Invalid transaction	62	Restricted card
13	Invalid amount	63	Security violation
14	Card number does not exist	64	Original amount incorrect
15	No such issuer	65	Activity count exceeded
16	Approved, update track 3	66	Card acceptor call acquirer
17	Customer cancellation	67	Card pick up at ATM
18	Customer dispute	68	Response received too late
19	Re-enter transaction	75	Too many wrong PIN tries
20	Invalid response	76	Previous message not found

21	No action taken (no match)	77	Data does not match original message
22	Suspected malfunction	80	Invalid date
23	Unacceptable transaction fee	81	Cryptographic error in PIN
24	File update not supported by receiver	82	Incorrect CVV
25	Unable to locate record	83	Unable to verify PIN
26	Duplicate file update record	84	Invalid authorization life cycle
27	File update field edit error	85	No reason to decline
28	File temporarily unavailable	86	PIN validation not possible
29	File update not successful	88	Cryptographic failure
30	Format error	89	Authentication failure
31	Issuer sign-off	90	Cutoff is in process
32	Completed partially	91	Issuer or switch inoperative
33	Expired card	92	No routing path
34	Suspected fraud	93	Violation of law
35	Card acceptor contact acquirer	94	Duplicate transmission
36	Restricted card	95	Reconcile error
37	Card acceptor call acquirer	96	System malfunction
38	Allowable PIN tries exceeded	97	Format Error
39	No credit account	98	Issuer Unreachable
40	Function not supported	99	Errored Transaction
41	Pick-up card (lost card)	XA	Forward to issuer
42	No universal account	XD	Forward to issuer
43	Pick-up card (stolen card)		

### AMEX

Response Code & Description	
000	Approved
001	Approved with ID
100	Deny
101	Expired Card
106	PIN tries Exceeded
107	Please Call Issuer
109	Invalid Service Establishment
110	Invalid Amount
111	Invalid Account
115	Requested Function Not Support
117	Incorrect PIN
121	Limit Exceeded

122	Invalid Manually Entered 4DBC
183	Invalid Currency Code
199	Valid PIN
200	Deny - Pick up Card
290	Refused, Retain Card
300	Successful
301	Not supported by receiver
302	Unable to locate record
303	Duplicate record
304	Field edit error
380	File update not accepted, high
400	Reversal Accepted
800	Accepted
880	File Fully Accepted
881	File Partially Accepted
882	File Fully Rejected
899	Table not found. Default used
900	Advice Accepted

**Appendix 5.3 – 3D-Secure Response Codes**

Reason Code	Reason Text (ReasonCodeDescription)	Note
13	Merchant is not allowed to process cards in this Payment system.	Merchant is blocked.
17	Unable to process transaction.	System cannot process a Card Range Request.
18	Unable to process transaction.	System cannot build a Verify Enrollment Request.
19	Unable to process transaction.	System cannot contact Visa Directory.
20	Unable to process transaction.	System cannot build a Payment Authentication.
21	Unable to process transaction.	System could not contact Issuer ACS Server
22	Unable to process transaction.	Issuer ACS responded with invalid data or returned data failed.
23	Unable to process transaction.	System cannot process a Verify Enrollment Request.
31	Authentication successful.	3-D Secure Payment Authentication successful.
32	Authentication failed.	3-D Secure Payment Authentication failed.
33	Authentication successful with attempt.	Attempt authentication was performed.
34	Authentication failed with error.	Authentication result not expected.
41	Card Holder Session Expired.	Cardholder's Session expired while performing a 3DS Transaction. Possibly because he/she closed the window, or pressed the back button in the middle of the transaction.
42	Illegal Operation by Card Holder. Check Order Status.	Cardholder Pressed the back button while the transaction was processing. Check the status of that order.
50	Verify Enrollment response unavailable.	The VERes message came back from the MPI as "U".
51	BIN Not Enrolled.	The VERes message came back from the MPI as "N"
52	Card Not Enrolled.	The VERes message came back from the MPI as "N"
53	Payer Authentication Response Unavailable	The PARes message came back from the MPI as "U".
96	Merchant URL is Missing	Merchant URL is Missing
98	System is temporarily down. Try later.	System is temporarily down. Try later.
401	Cycle interrupted by the user or client/browser connection not available.	Client Browser connection not available or cardholder referred in the process (Back/F5).
1001	FACPGWS Invalid Protocol. Only HTTPS Allowed	The request was sent via HTTP not HTTPS.
1002	Missing Parameter or Parameters	One or more of the required parameters is missing in the web method you have called.
1004	Invalid Amount. Not 12 characters in length	Amount must be exactly 12 characters in length, right-aligned, left-padded with zeros. For example, \$12.00 = 000000001200
1005	Invalid Capture Flag value provided	The CaptureFlag parameter must be set to either "M" for manual capture (authorize only) or "A" for automatic (authorize/capture)

## Appendix 5.4 – AVS Response Codes

AVS Codes are returned in the AVSResult field in the Response message of the Operation concerned; one of AuthorizeResponse, TransactionModificationResponse, or TransactionStatusResponse (see main FACPG2 Integration Guide). There are different codes depending on the card type.

### Visa

Code	Definition
<b>A</b>	Address matches, Zip code does not match.
<b>B</b>	Street addresses match for international transaction. Postal code not verified due to incompatible formats. (Acquirer sent street address and postal code.)
<b>C</b>	Street address and postal code not verified for international transaction due to incompatible formats. (Acquirer sent street address and postal code.)
<b>D</b>	Street addresses and postal codes match for international transaction.
<b>E</b>	Error response for Merchant Category Code.
<b>F</b>	Address does compare and five-digit ZIP codes does compare (UK only)
<b>G</b>	Address information is unavailable for international transaction; non-AVS participant.
<b>I</b>	Address information not verified for international transaction.
<b>M</b>	Street addresses and postal codes match for international transaction.
<b>N</b>	Address and ZIP code do not match.
<b>P</b>	Postal codes match for international transaction. Street address not verified due to incompatible formats. (Acquirer sent street address and postal code.)
<b>R</b>	Retry; system unavailable or timed out.
<b>S</b>	Service not supported by issuer.
<b>U</b>	Address information is unavailable; domestic transactions.
<b>W</b>	Nine-digit ZIP code matches, but address does not match.
<b>X</b>	Exact match, address, and nine-digit ZIP code match.
<b>Y</b>	Address and five-digit ZIP code match.
<b>Z</b>	Five-digit ZIP code matches, but address does not match.
<b>5*</b>	Invalid AVS response (from VISA).
<b>9*</b>	Address Verification Data contains EDIT ERROR.
<b>0</b>	Issuer has chosen not to perform Address Verification for an authorization that was declined.

### MasterCard

Code	Definition
<b>A</b>	Address matches, postal code does not.
<b>N</b>	Neither address nor postal code matches.
<b>R</b>	Retry, system unable to process.
<b>S</b>	AVS currently not supported
<b>U</b>	No data from issuer/Authorization System.

<b>W</b>	For U.S. addresses, nine-digit postal code matches, address does not; for address outside the U.S., postal code matches, address does not.
<b>X</b>	For U.S. addresses, nine-digit postal code and address matches; for addresses outside the U.S., postal code and address match.
<b>Y</b>	For U.S. addresses, five-digit postal code and address matches.
<b>Z</b>	For U.S. addresses, five-digit postal code matches, address does not.
<b>5*</b>	Invalid AVS response (from MasterCard)
<b>9*</b>	Address Verification Data contains EDIT ERROR.
<b>0</b>	Issuer has chosen not to perform Address Verification for an authorization that was declined.

Note: For MasterCard, if a 5 digit zip code is sent and a 9 digit zip code is on the cardholder file (and address matches) a response of 'Y' is returned.

#### Amex

Code	Definition
<b>A</b>	ADDRESS: Address correct, zip code incorrect
<b>N</b>	NO: Address and zip code are no correct.
<b>R</b>	Retry, system unavailable or timeout.
<b>S</b>	Address Verification Service not valid.
<b>U</b>	Address information is unavailable; account number is not US or Canadian.
<b>Y</b>	YES: Address and zip code are correct.
<b>Z</b>	Zip code correct; address incorrect.
<b>5*</b>	Invalid AVS response (from American Express).
<b>9*</b>	Address Verification Data contains EDIT ERROR.

\* These responses (5 & 9) for all credit card types are processor-generated responses. Response Code 9 means the record was not sent out for Address Verification. This response will also be returned when address verification has not been requested.

#### Appendix 5.5 – CVV Response Codes

After checking a CVV2/CVC2, values are returned in the CVV2Result field as follows:

Code	Definition
<b>M</b>	Match
<b>N</b>	No match.
<b>P</b>	Not Processed
<b>S</b>	Should be on card but was not provided. (Visa only)
<b>U</b>	Issuer not participating or certified.



## Appendix 5.6 – Fraud Control Response Codes

### ResponseCode

Response Code	Description
<b>1</b>	Fraud Check Successful
<b>2</b>	Decline
<b>3</b>	Error

### ReasonCode

There are only ReasonCodes for decline and errors (ResponseCode 2 and 3).

Response Code	ReasonCode	Description	Details
<b>2</b>	<b>2020</b>	FraudControl Decline	FraudControl query succeeded without error but the transaction declined, as it did not pass the fraud check rules based on Kount response code.
<b>2</b>	<b>2021</b>	BinCheck Decline	BinCheck was successful but the transaction declined as it did not pass the BinCheck rules based on BIN data.
<b>3</b>	<b>321</b>	BAD_EMAL	The email address does not meet required format or is
<b>3</b>	<b>2001</b>	Merchant Not Enabled	FraudControl is not enabled for this merchant.
<b>3</b>	<b>2002</b>	Invalid Fraud Profile	Merchant settings do not specify a valid fraud profile
<b>3</b>	<b>2003</b>	Missing MerchantId	Could not find fraud-specific MerchantId for this merchant
<b>3</b>	<b>2004</b>	Invalid Fraud Response	Response from Fraud system was invalid
<b>3</b>	<b>2005</b>	FraudCheckOnly Not Supported	FraudCheckOnly transactions are not supported with the current merchant configuration.
<b>3</b>	<b>2006</b>	Simulated Fraud Response	Fraud Response Codes and Score are simulated. For testing only.
<b>3</b>	<b>2007</b>	BinCheck System Error	BinCheck System Error
<b>3</b>	<b>2091</b>	Response Timeout	Timeout waiting for Fraud System Response or communications error
<b>3</b>	<b>2097</b>	Format Error (Various)	Various format errors. Details will be in the description.
<b>3</b>	<b>2096</b>	FraudControl System Error	FraudControl System Exception
<b>3</b>	<b>2099</b>	FraudControl System Error	FraudControl Internal Error

\* - ResponseCode 1 has no ReasonCode or ReasonCodeDesc

***FraudResponseCode (OriginalResponseCode)***

These are only if you are subscribed to FAC's fraud service which includes Kount or PayTrue services. These are the actual response codes returned by the Fraud System (third party)

Code	System	Description
<b>A</b>	Kount	Authorize
<b>D</b>	Kount	Decline
<b>R</b>	Kount	Review
<b>E</b>	Kount	Escalate
<b>[Various]</b>	PayTrue	See PayTrue Documentation
<b>B</b>	BinCheck	BinCheck decline based on merchant rules and BinCheck data
<b>91</b>	All	Timeout
<b>12</b>	All	Invalid transaction - FraudControl is not enabled for merchant (FOnly)
<b>99</b>	All	Error

## Appendix 6 – PHP Code Snippets

### Calling the Hosted Page Service Methods

To call the hosted page service methods you must use SOAP 1.1 or higher protocol. This is usually done using some kind of proxy object or library. In PHP this is called the SoapClient.

### Calling HostedPageAuthorize within a Page

This commented example is a PHP script called from another page, say your checkout page. It does the following:

- Initializes the SoapClient options array
- Initializes the SoapClient with the FACPG2 URL and options array
- Builds the HostedPageAuthorizeRequest using associative arrays in PHP
- Calls the HostedPageAuthorize service operation
- Builds the hosted page URL
- Redirects the user to the hosted page

```
<?php

// Useful for generation of test Order numbers
// You would use REAL order numbers in an Integration
function msTimeStamp()
{
    return (string)round(microtime(1) * 1000);
}

// How to sign an FAC Authorize message in PHP
function Sign($passwd, $facId, $acquirerId, $orderNumber, $amount, $currency)
{
    $stringtohash = $passwd.$facId.$acquirerId.$orderNumber.$amount.$currency;
    $hash = sha1($stringtohash, true);
    $signature = base64_encode($hash);

    return $signature;
}

// FAC Integration Domain
$domain = 'ecm.firstatlanticcommerce.com';

// Ensure you append the ?wsdl query string to the URL
$wsdlurl = 'https://' . $domain . '/PGService/HostedPage.svc?wsdl';
$soapUrl = 'https://' . $domain . '/PGService/HostedPage.svc';

// Set up client to use SOAP 1.1 and NO CACHE for WSDL. You can choose between
// exceptions or status checking. Here we use status checking. Trace is for Debug only
// Works better with MS Web Services where
// WSDL is split into several files. Will fetch all the WSDL up front.
$options = array(
    'location' => $soapUrl,
```

```
'soap_version'=>SOAP_1_1,  
'exceptions'=>0,  
'trace'=>1,  
'cache_wsdl'=>WSDL_CACHE_NONE  
);  
  
// WSDL Based calls use a proxy, so this is the best way  
// to call FAC PG Operations.  
$client = new SoapClient($wsdlurl, $options);  
  
// This should not be in your code in plain text!  
$password = '<YOUR PASSWORD HERE>';  
// Use your own FAC ID  
$facId = '<YOUR MERCHANT ID>';  
//$facId = '888XXXXX';  
// Acquirer is always this  
$acquirerId = '464748';  
// Must be Unique per order. Put your own format here. The field allows up to 150  
alphanumeric characters.  
$orderNumber = 'TEST_ORDER_ID' . msTimeStamp();  
  
// THESE next variables COME FROM THE PREVIOUS PAGE (hence $_POST) but you could drive  
these from  
// any source such as config files, server cache etc.  
  
// Passed in as a decimal but 12 chars is required  
$amount = $_POST["Amount"];  
// Page Set  
$pageset = $_POST["PageSet"];  
// Page Name  
$pagename = $_POST["PageName"];  
// TransCode  
$transCode = $_POST["TransCode"];  
  
// Formatted Amount. Must be in twelve character, no decimal place, zero padded format  
$amountFormatted = str_pad(''.($amount*100), 12, "0", STR_PAD_LEFT);  
  
// 840 = USD, put your currency code here  
$currency = '840';  
  
// Each call must have a signature with the password as the shared secret  
$signature = Sign($password, $facId, $acquirerId, $orderNumber, $amountFormatted, $curren  
cy);  
  
// You only need to initialise the message sections you need. So for a basic Auth  
// only Credit Cards and Transaction details are required.  
  
// Transaction Details.  
$TransactionDetails = array('AcquirerId' => $acquirerId,  
    'Amount' => $amountFormatted,  
    'Currency' => $currency,  
    'CurrencyExponent' => 2,  
    'IPAddress' => '',  
    'MerchantId' => $facId,  
    'OrderNumber' => $orderNumber,  
    'Signature' => $signature,
```

```
'SignatureMethod' => 'SHA1',
'TransactionCode' => $transCode);
```

```
// Where the response will end up. Should be a page your site and will get two parameters
// ID = Single Use Key passed to payment page and RespCode = normal response code for Auth
$CardHolderResponseUrl = 'https://<YOUR DOMAIN>/<YOUR RESPONSE PAGE>.php';

// The request data is named 'Request' for reasons that are not clear!
$HostedPageRequest = array('Request' => array('TransactionDetails' => $TransactionDetails
,
    'CardHolderResponseURL' => $CardHolderResponseUrl));

// Call the Authorize through the Soap Client
$result = $client->HostedPageAuthorize($HostedPageRequest);

// You should CHECK the results here!!!
// Extract Token
$token = $result->HostedPageAuthorizeResult->SingleUseToken;

// Construct the URL. This may be different for Production. Check with FAC
$PaymentPageUrl = 'https://' . $domain . '/MerchantPages/' . $pageset . '/' . $pagename .
    '/';

// Create the location header to effect a redirect. Add token required by page
$RedirectURL = 'Location: ' . $PaymentPageUrl . $token;

// Redirect user to the Payment page
header($RedirectURL);
?>
```

## Calling HostedPageResults within a Page

This code snippet shows how to call the results operation. You may do this within for example the response page passed in the call to HostedPageAuthorize (CardHolderResponseURL). The response page is passed the Token as a parameter called "ID" so is available to use.

```
<?php
// IMPORTANT: Convert URL Parameters to variables
parse_str($_SERVER['QUERY_STRING']);

$host = 'ecm.firstatlanticcommerce.com';

// Ensure you append the ?wsdl query string to the URL for WSDL URL
$wsdlurl = 'https://' . $host . '/PGService/HostedPage.svc?wsdl';
// No WSDL parameter for location URL
$loclurl = 'https://' . $host . '/PGService/HostedPage.svc';

// Set up client to use SOAP 1.1 and NO CACHE for WSDL. You can choose between
// exceptions or status checking. Here we use status checking. Trace is for Debug only
// Works better with MS Web Services where
// WSDL is split into several files. Will fetch all the WSDL up front.
$options = array(
    'location' => $loclurl,
    'soap_version'=>SOAP_1_1,
```

```

    'exceptions'=>0,
    'trace'=>1,
    'cache_wsdl'=>WSDL_CACHE_NONE
);

// WSDL Based calls use a proxy, so this is the best way
// to call FAC PG Operations as it creates the methods for you
$client = new SoapClient($wsdlurl, $options);

// Call the HostedPageResults through the Client. Note the param
// name is case sensitive, so 'Key' does not work.
$result = $client->HostedPageResults(array('key' => $ID));

// NOW: You have access to all the response fields and can evaluate as you want to
// and use them to display something to the user in an HTML page like the HTML snippet
// below. It is very simple and you have not had any exposure to the card number at all.

// While it is not necessary to make this soap call, it is advisable that you implement this
// and get the full response details to ensure the correct amount has been charged etc.
// You should also store the results in case of any chargeback issues and to check the response
// code has not been tampered with.
?>
<html>
<head>
<title>Cart</title>
    <style type="text/css">
        .label
        {
            width: 200px;
            height: 30px;
            margin-left: 10px;
        }
        .labelHeading
        {
            width: 200px;
            height: 30px;
            margin-left: 10px;
            font-family: @Arial Unicode MS;
            font-weight: bold;
        }
    </style>
</head>
<body>
<form action="">
<br/>

// EXAMPLE CODE ONLY SHOWING HOW YOU CAN ACCESS THE RESULTS WITHIN THE HTML PAGE
    <label class="labelHeading">Payment Processed</label><br />
    <br />
    <label class="label">Status: <?php echo $result->HostedPageResultsResult->AuthResponse-
    >CreditCardTransactionResults->ReasonCodeDescription; ?> </label><br/>
    <label class="label">Payment Reference: <?php echo $result->HostedPageResultsResult->AuthResponse-
    >CreditCardTransactionResults->ReferenceNumber; ?> </label><br/><br/>
    <label class="label">All Response Fields: <?php echo print_r($result); ?> </label><br/>
<br/>
</form>
</body>
</html>

```

## ComputeHash function

```
// How to sign an FAC Authorize message in PHP
function Sign($passwd, $facId, $acquirerId, $orderNumber, $amount, $currency)
{
    $stringtohash = $passwd.$facId.$acquirerId.$orderNumber.$amount.$currency;
    $hash = sha1($stringtohash, true);
    $signature = base64_encode($hash);

    return $signature;
}
```

## Appendix 7 – C# Code Snippets

### Calling HostedPageAuthorize (Getting an HPP Token)

It is required to add a Service reference to:

<https://ecm.firstatlanticcommerce.com/PGService/HostedPage.svc?wsdl>

```
[TestClass]
public class HPPTests
{
    [TestMethod]
    public void TestHPPOnStaging_WithValidInput_ShouldReturnValidHPPToken()
    {
        //Arrange
        var entryForm = new HostedPageEntryForm
        {
            SelectedTestEnvironment = { BaseUrl = "https://ecm.firstatlanticcommerce.com/" },
            TransactionDetails = new TransactionDetails
            {
                Amount = 1,
                Currency = "840",
                AllowAmountCurrencyChange = false,
                TrxnCodeValue = 0,
                OrderNumber = Right(Guid.NewGuid().ToString(), 20).Replace("-", "") + "|HPP"
            },
            MerchantDetails = new HostedPageMerchantDetails
            {
                MerchantId = "YOUR MERCHANT FAC ID NUMBER GOES HERE",
                MerchantPassword = "YOUR MERCHANT PROCESSING PASSWORD GOES HERE",
                PageName = "YOUR PAGE'S NAME GOES HERE",
                PagesetName = "YOUR PAGESET'S NAME GOES HERE"
            }
        };

        var tDetails = new HPPSvc.TransactionDetails
        {
            AcquirerId = "464748",
            MerchantId = entryForm.MerchantDetails.MerchantId ?? "",
            Amount = Common.AmountFormatted(entryForm.TransactionDetails.Amount),
            OrderNumber = entryForm.TransactionDetails.OrderNumber,
            Signature = GenerateSignature(entryForm.TransactionDetails.AllowAmountCurrencyChange,
entryForm),
            SignatureMethod = "SHA1",
            Currency = entryForm.TransactionDetails.Currency,
            CurrencyExponent = 2,
            TransactionCode = entryForm.TransactionDetails.TrxnCodeValue,
            CustomData = entryForm.TransactionDetails.CustomData,
            CustomerReference = $"FAC Web Tools HPP: {entryForm.MerchantDetails.MerchantId}"
        };

        HostedPageAuthorizationRequest PageAuthorizationRequest = new HostedPageAuthorizationRequest
        {
```



```

        CardHolderResponseURL = "https://ecm.firstatlanticcommerce.com/",
        TransactionDetails = tDetails
    };

    HostedPageAuthorizationResponse PageAuthorizationResponse;

    //Act
    using (var client = Common.GetHostedPageClient(entryForm.SelectedTestEnvironment.BaseUrl))
    {
        PageAuthorizationResponse = client.HostedPageAuthorize(PageAuthorizationRequest);
    }

    //Assert
    Assert.IsTrue(PageAuthorizationResponse.ResponseCode == "0" &&
        PageAuthorizationResponse.ResponseCodeDescription == "Authorized");

    Assert.IsTrue(!string.IsNullOrEmpty(PageAuthorizationResponse.SingleUseToken) &&
        PageAuthorizationResponse.SingleUseToken.Length > 20);
    }
}

```

### Calling HostedPageResults (Retrieve transaction details)

- A precondition for this operation is having obtained a valid HPP Token (via the HostedPageAuthorize operation)
- It is required to add a Service reference to:

<https://ecm.firstatlanticcommerce.com/PGService/HostedPage.svc?wsdl>

```

[TestClass]
public class HPPTests
{
    [TestMethod]
    public void TestHPPResultsOnStaging_WithValidToken_ShouldReturnTransactionInformation()
    {
        string hppKey = "A114Xa2IYUGnzPbj1Nn13A2";
        HostedPageResultsResponse PageResultResponse;

        using (var client = Common.GetHostedPageClient("https://ecm.firstatlanticcommerce.com/"))
        {
            PageResultResponse = client.HostedPageResults(hppKey);
        }

        Assert.IsTrue(PageResultResponse.AuthResponse.CreditCardTransactionResults.ResponseCode=="1" &&
            PageResultResponse.AuthResponse.CreditCardTransactionResults.ReasonCode=="1" &&
            PageResultResponse.AuthResponse.CreditCardTransactionResults.ReasonCodeDescription=="Transaction is approved.");
    }
}

```

## ComputeHash function

This function builds a hash with the provided with the provided Merchant and Transactional information

The following references have to be included:

- `using System.Security.Cryptography;`
- `using System.Web;`

```
private string ComputeHash(string decryptedPassword, string requestMerchantId, string acquirer,
                           string requestMerchantOrder, string requestAmount,
                           string requestCurrency)
{
    SHA1CryptoServiceProvider objSHA1 = new SHA1CryptoServiceProvider();

    string key = decryptedPassword + requestMerchantId.Trim() + acquirer +
                requestMerchantOrder.Trim() + requestAmount.Trim() + requestCurrency.Trim();

    objSHA1.ComputeHash(System.Text.Encoding.UTF8.GetBytes(key.ToCharArray()));
    byte[] buffer = objSHA1.Hash;
    string HashValue = System.Convert.ToBase64String(buffer);

    HashValue = HttpUtility.UrlEncode(HashValue);

    return HashValue;
}
```

Note: A Test class for the ComputeHash function is available on Appendix 8

## Appendix 8 – Signature and URL Encoding of the Signature

### What is the Signature?

The signature is an encoded message digest generated from the concatenation of following pieces of information:

- Processing Password
- FAC ID (Merchant ID)
- Acquirer ID
- Merchant Order ID
- Amount
- Currency

The encoded message digest is generated using the SHA1 algorithm

- SHA-1 (Secure Hash Algorithm 1) is a cryptographic hash function which takes an input and produces a 160-bit (20-byte) hash value known as a message digest - typically rendered as a hexadecimal number, 40 digits long
- More information in: <https://en.wikipedia.org/wiki/SHA-1>
- The generated message digest needs to be URL Encoded

### URL Encoding

- The Signature must be URL Encoded.
- An exception is HPP Auth+3DS (TransactionCode 256). For those, the signature cannot be URL-encoded because our routine rejects signatures longer than 28 characters. Additionally, URL-encoding increases the signature length by 2 characters per each unsafe character.
- URL encoding replaces reserved ASCII characters with a "%" followed by two hexadecimal digits.
- When a character from a reserved set (a "reserved character") has special meaning (a "reserved purpose") in a certain context, then the character must be *percent-encoded*

For example in the following signature: 9QBY9MP1gH+8bEk0d3Ycj/Het8Y=

- These reserved characters + / and = will be encoded as %2b, %2f and %3d
- The encoded signature is 9QBY9MP1gH%2b8bEk0d3Ycj%2fHet8Y%3d

More information is available in

- [https://www.w3schools.com/tags/ref\\_urlencode.asp](https://www.w3schools.com/tags/ref_urlencode.asp)
- <https://en.wikipedia.org/wiki/Percent-encoding>

## Validating the ComputeHash Function on C#

- An important point for URL encoded characters is that our signature validation routine is case sensitive.
- For example some platforms (e.g. Java) encode "=" as %3D
- Our validation routine requires "=" to be encoded as %3d and not %3D

To validate that the hash function is working as expected, the following test can be used.

- For the following input

```
MerchantId = "724589";  
TokenPAN = "411111_000021111";  
Currency = "840";  
Amount = "00000000101";  
MerchantOrder = "ORDER_NUM_1";  
decryptedPassword = "3WfCu28p";
```

- The following hash is expected

```
9QBY9MPlgH%2b8bEk0d3Ycj%2fHet8Y%3d
```

### Source code

```
using System;  
using System.Security.Cryptography;  
using System.Web;  
using Microsoft.VisualStudio.TestTools.UnitTesting;  
  
[TestClass]  
public class TestHash  
{  
    [TestMethod()]  
    public void TestComputeHash()  
    {  
        DeTokenizeRequest request = new DeTokenizeRequest();  
        request.MerchantId = "724589";  
        request.TokenPAN = "411111_000021111";  
        request.Currency = "840";  
        request.Amount = "00000000101";  
        request.MerchantOrder = "ORDER_NUM_1";  
  
        string decryptedPassword = "3WfCu28p";  
  
        string hash = ComputeHash(decryptedPassword, request.MerchantId, "464748",  
                                request.MerchantOrder, request.Amount, request.Currency);  
  
        // Verify the hash is correct  
        Assert.IsTrue(!string.IsNullOrEmpty(hash) && hash == "9QBY9MPlgH%2b8bEk0d3Ycj%2fHet8Y%3d");  
    }  
}
```

```
// Verify the Url can be decoded
Assert.IsTrue(HttpUtility.UrlDecode(hash) == "9QBY9MP1gH+8bEk0d3Ycj/Het8Y=");
}

private string ComputeHash(string decryptedPassword, string requestMerchantId, string acquirer,
                           string requestMerchantOrder, string requestAmount,
                           string requestCurrency)
{
    SHA1CryptoServiceProvider objSHA1 = new SHA1CryptoServiceProvider();

    string key = decryptedPassword + requestMerchantId.Trim() + acquirer +
                requestMerchantOrder.Trim() + requestAmount.Trim() + requestCurrency.Trim();

    objSHA1.ComputeHash(System.Text.Encoding.UTF8.GetBytes(key.ToCharArray()));
    byte[] buffer = objSHA1.Hash;
    string HashValue = System.Convert.ToBase64String(buffer);

    HashValue = HttpUtility.UrlEncode(HashValue);

    return HashValue;
}

public class DeTokenizeRequest
{
    public string TokenPAN { get; set; }
    public string MerchantId { get; set; }
    public string Currency { get; set; }
    public string Amount { get; set; }
    public string MerchantOrder { get; set; }
}
```

## Appendix 9 - Glossary of Terms

### *3D Secure*

3D Secure encompasses both Visa's *Verified by Visa* and MasterCard's *SecureCode* security solutions for online e-commerce transactions. These solutions use personal passwords to help protect cardholders' card numbers against unauthorized use.

### *Authentication*

The process of authenticating is used in 3D Secure transactions to verify that the person attempting a transaction with a given credit card number is the actual cardholder by requiring them to enter a personal password they set up when enrolling in the 3D Secure program (either *Verify by Visa* or *SecureCode*).

### *Authorization*

The process of checking that the credit card being used in a transaction contains sufficient funds to cover the amount of the transaction. Note that if sufficient funds are found, the amount is held for a given period of time, waiting to be withdrawn when settlement occurs (the period of time varies based on the issuing bank of the credit card).

### *Authorization/Capture*

An Authorize/Capture not only checks that the credit card being used in a transaction contains sufficient funds to cover the amount of the transaction, it also flags the transaction as captured meaning it is to be sent for settlement in the next settlement period.

### *AVS (Address Verification System)*

AVS is used as an extra level of security for online credit card transactions that takes the first line of the billing address and the zip/postal code of the cardholder and checks if they are valid as compared to what is stored on file for the given credit card number.

### *CID (Card Identification Digits)*

The 4-digit code found on the front of AMEX cards, the CID is used as an extra security step to help to verify that the person using the credit card is the actual cardholder.

### *CVC2 (Card Verification Code)*

The 3-digit code found on the back of MasterCard cards, the CVC2 is used as an extra security step to help to verify that the person using the credit card is the actual cardholder.

### *CVV2 (Card Verification Value)*

The 3-digit code found on the back of Visa cards, the CVV2 is used as an extra security step to help to verify that the person using the credit card is the actual cardholder.

### *Capture*

When a capture is performed (in either an Authorize/Capture or Capture only transaction), it is the process of flagging an already authorized transaction to be settled in the next settlement period.

### *Hosted Page*

A payment page hosted on the servers at FAC.

### *One-Pass Transaction*

A one-pass transaction (also called an authorize/capture transaction in this document) is a transaction that is both authorized and captured (flagged for settlement) at the same time, in a single transaction request.

### *FACPG and FACPG2*

The First Atlantic Commerce Payment Gateway Services. These services support and enable the FAC products [cGate® Secure Real-Time](#) and [cGate® Secure Verify](#).

***Refund***

A refund is the process of refunding a previously settled transaction. This will appear as a credit on the cardholder's credit card statement.

***Reversal***

A reversal is the process of reversing a previously captured, but not yet settled, transaction. It means that the transaction will never appear on the cardholder's credit card statement.

***Settle/Settled/Settlement***

The process of settling a transaction is when the money is taken from the cardholder's account and put into the merchant's account. Once a transaction is settled, it will appear as a charge on the cardholder's credit card statement.

***SHA1***

Secure Hash Algorithm 1. A message digest (hash) function defined in RFC 3174.

***Single-Use Token***

A token used to identify a transaction without revealing the details of that transaction and can only be used during the transaction time frame itself. After that, the token is unusable and meaningless.

Used in conjunction with shared secret validation it ensures a safe transaction is performed on a Hosted Page.

***Transaction***

A transaction is any e-commerce request made by you, the merchant, to FAC. This includes Authorizations (both 3D and Non-3D Secure), Authorization & Captures (both 3D and Non-3D Secure), Captures only, Reversals, Refunds, 3D Secure Authentication Only transactions and AVS Verification Only transactions.

***Two-Pass Transaction***

A two-pass transaction is a transaction that is processed in two separate transaction requests. The first transaction is the authorization only request and the second transaction (which can come seconds, minutes, hours or even days after the first transaction) captures this transaction and flags it for settlement.