First Data.

First Data Gateway

Connect® Integration Guide

Version 3.7

First Data is a trading name of First Data Europe Limited, a private limited company incorporated in England (company number 02012925) with a registered address at Janus House, Endeavour Drive, Basildon, Essex, SS14 3WF. First Data Europe Limited is authorised and regulated by the UK Financial Conduct Authority (FCA register No. 582703; CCA No. 739230).

First Data Europe Limited has appointed FDR Limited as payment and collection agent for the services provided under your Merchant Agreement. FDR Limited is a company incorporated in the State of Delaware, United States, under registration number 22692 35, registered in England as a branch of an overseas company with limited liability (company number FC015955) and branch number BR001147, whose registered office in the United Kingdom is at Janus House, Endeavour Drive, Basildon, Essex, SS14 3WF.

Contents

Getting Support

There are different manuals available for First Data's eCommerce solutions. This Integration Guide will be the most helpful for integrating the Connect solution.

For information about settings, customisation, reports and how to process transactions manually (by keying in the information) please refer to the User Guide Virtual Terminal.

If you have read the documentation and cannot find the answer to your question, please contact your local support team.

1 Introduction

The Connect solution provides a quick and easy way to add payment capabilities to your website.

Connect manages the customer redirections that are required in the checkout process of many payment methods or authentication mechanisms and gives you the option to use secure hosted payment pages which can reduce the burden of compliance with the Data Security Standard of the Payment Card Industry (PCI DSS)

This document describes how to integrate your website using Connect and provides step by step instructions on how to quickly start accepting payments from your web shop.

Depending on your business processes, it can also make sense to additionally integrate our Web Service API solution (see Web Service API Integration Guide).

2 Payment process options

The Connect solution provides a number of different options for the payment process to support integrations where you handle most of the customer interactions on your own website up to integrations where you use ready-made form pages for the entire payment process.

In the scenarios where you prefer not to use a hosted form, you can submit the required customer data directly from your own form to First Data but please be aware that if you store or process sensitive cardholder data within your own application, you must ensure that your system components are compliant with the Data Security Standard of the Payment Card Industry (PCI DSS).

2.1 Checkout option ‘Classic’

The checkout option ‘classic’ splits the payment process into multiple pages where you can easily decide what kind of information you would like collected by one of the Gateway’s hosted forms or what you want to collect yourself within your web shop environment.

As an example you can let customers select their preferred payment method within your web shop and submit that payment method in your request to Connect – or if you should prefer not to send the payment method, the Connect solution will automatically show a payment method selection page to your customer where they can choose from all payment methods that are activated for your store.

With three different modes, you can define the range of data captured by the Gateway:

- payonly: shows a hosted page to collect the minimum set of information for the transaction (e.g. cardholder name, card number, expiry date and card code for a credit card transaction).
- payplus: in addition to the above, the Gateway collects a full set of billing information on an additional page.
- fullpay: in addition to the above, the Gateway displays a third page to also collect shipping information.

The most important aspect when using the hosted payment pages is the security of sensitive cardholder data. When you decide to let your customers enter their credit card details on the page that we provide and host on our servers for this purpose, it facilitates your compliance with the Data Security Standard of the Payment Card Industry (PCI DSS) as the payment processing is completely hosted by First Data.

The hosted pages can be customised with your own logo, colours, and font types in order to make them fit to the look and feel of your web shop. Please refer to the Virtual Terminal User Guide to learn about how to make such customisations.

2.2 Checkout option ’combinedpage’

The checkout option ‘combinedpage’ consolidates the payment method choice and the typical next step (e.g. entry of card details or selection of bank) in a single page which gets automatically optimised for different kinds of user devices, e.g. PC, smartphone, tablet, etc.

This hosted page also shows your merchant name at the top and allows you to display a summary of the purchased items to your customer.

Please note that this checkout option has some functional limitations in comparison to the ‘classic’ option:

- Supported payment methods are currently limited to: credit cards, Maestro, MasterPass and PayPal. There are no customisation options (logo, colours, etc.)
- It makes use of technical mechanisms that may not work with out-dated browser versions

2.3 Checkout option ’simpleform’

The checkout option ‘simpleform’ offers you a minimalistic form version to just capture the sensitive data (e.g. card number or/and CVC code) and to be displayed within an iFrame embedded in the context of your website.

The simplified hosted payment form with the checkout option ‘simpleform’ can be used for the credit/debit card processing incl.: 3D Secure and Dynamic Currency Conversion (DCC).

It works with HTML 5 enabled browsers like e.g.: Microsoft Internet Explorer 9 onwards, Google Chrome and Mozilla Firefox, the minimum size of the iFrame is: 900px (h) and 460px (w). You can also send a hex color code to easily align the background color of the buttons in an iFrame to the look and feel of your website.

When using an iFrame, you need to have a mechanism in your webshop to receive the response from the iFrame after the processing is complete by the gateway. In order to do so, you need to register a callback method e.g.: receiveMessage(), with a Window EventListener, listening on the Javascript “message“ event. The callback method will have the “event” object. This object will contain all the information needed to process the response from the iFrame.

The code examples of how to integrate the simplified hosted payment form with the checkout option ‘simpleform’ are given in Appendix VII.

3 Getting Started

This section assumes that the developer has a basic understanding of the chosen scripting language and provides a simple example on how to integrate your website using the “classic” checkout option in payonly Mode. Examples are provided using ASP and PHP.

3.1 Checklist

In order to integrate with the First Data Gateway, you must have the following items:

- Store Name
  This is the ID of the store that was given to you by First Data.
  For example: 10123456789

- Shared Secret
  This is the shared secret provided to you by First Data.
  This is used when constructing the hash value (see below).

3.2 ASP Example

The following ASP example demonstrates a simple page that will communicate with the Gateway in payonly mode.

When the cardholder clicks Submit, they are redirected to the First Data secure page to enter the card details. After payment has been completed, the user will be redirected to the merchants receipt page. The location of the receipt page can be configured.

```
<!-- #include file="ipg-util.asp"-->

<html>
<head><title>IPG Connect Sample for ASP</title></head>
<body>

<p><h1>Order Form</h1></p>

<form method="post" action="https://test.ipg-online.com/connect/gateway/processing">
<input type="hidden" name="txntype" value="sale"/>
<input type="hidden" name="timezone" value="Europe/London"/>
<input type="hidden" name="txndatetime" value="<%getDateTime() %>"/>
<input type="hidden" name="hash_algorithm" value="SHA256"/>
<input type="hidden" name="hash" value="<% call createHash("13.00","826") %>"/>
<input type="hidden" name="storename" value="10123456789"/>
<input type="hidden" name="mode" value="payonly"/>
<input type="hidden" name="paymentMethod" value="M"/>
<input type="text"   name="chargetotal" value="13.00" />
<input type="hidden" name="currency" value="826"/>

<input type="submit" value="Submit">

</form>
</body>
</html>
```

The code presented in “Appendix II” represents the included file ipg-util.asp. It includes code for generating a SHA-256 hash as is required by First Data. The provision of a hash in the example ensures that this merchant is the only merchant that can send in transactions for this store.

Note: the POST URL used is for integration testing only. When you are ready to go into production, please contact First Data and you will be provided with the live production URL.

Note: To prevent fraudulent transactions, it is recommended that the 'hash' is calculated within your server and JavaScript is not used like shown in the samples mentioned.

3.3 PHP Example

The following PHP example demonstrates a simple page that will communicate with the Gateway in payonly mode.

When the cardholder clicks Submit, they are redirected to the First Data secure page to enter the card details. After payment has been completed, the user will be redirected to the merchants receipt page. The location of the receipt page can be configured.

```
<? include("ipg-util.php"); ?>

<html>
<head><title>IPG Connect Sample for PHP</title></head>
<body>

<p><h1>Order Form</h1>

<form method="post" action="https://test.ipg-
online.com/connect/gateway/processing">
<input type="hidden" name="txntype" value="sale"/>
<input type="hidden" name="timezone" value="Europe/Berlin"/>
<input type="hidden" name="txndatetime" value="<?php echo getDateTime() ?>"/>
<input type="hidden" name="hash_algorithm" value="SHA256"/>
<input type="hidden" name="hash" value="<?php echo createHash("13.00","826") ?>"/>
<input type="hidden" name="storename" value="10123456789"/>
<input type="hidden" name="mode" value="payonly"/>
<input type="hidden" name="paymentMethod" value="M"/>
<input type="text"   name="chargetotal" value="13.00"/>
<input type="hidden" name="currency" value="826"/>

<input type="submit" value="Submit">

</form>
</body>
</html>
```

Note: The POST URL used in this example is for integration testing only. When you are ready to go into production, please contact First Data and you will be provided with the live production URL.

The code presented in “Appendix III” represents the included file ipg-util.php. It includes code for generating a SHA-256 hash as is required by First Data. The provision of a hash in the example ensures that this merchant is the only merchant that can send in transactions for this store.

3.4 Amounts for test transactions

When using our test system for integration, odd amounts (e.g. 13.01 GBP or 13.99 GBP) can cause the transaction to decline as these amounts are sometimes used to simulate unsuccessful authorisations.

We therefore recommend using even amounts for testing purpose, e.g. 13.00 GBP like in the example above.

4 Mandatory Fields

Depending on the transaction type, the following form fields must be present in the form being submitted to the Gateway (X = mandatory field). Please refer to this Integration Guide’s Appendixes for implementation details in relation to alternative payment methods.

Field Name	Description, possible values and format	„Sale“ transaction	PreAuth*	PostAuth*	Void	PayerAuth**
txntype	'sale’, 'preauth’, 'postauth’, 'void’ or ‘payer auth’ (the transaction type – please note the descriptions of transaction types in the User Guide Virtual Terminal & Manager) The possibility to send a ‘void’ using the Connect interface is restricted. Please contact your local support team if you want to enable this feature.	X	X	X	X	X
timezone	Time zone of the transaction in Area/Location format, e.g. Europe/London	X	X	X	X	X
txndatetime	YYYY:MM:DD-hh:mm:ss (exact time of the transaction)	X	X	X	X	X
hash_algorithm	This is to indicate the algorithm that you use for hash calculation. The possible values are SHA256 and SHA512.	X	X	X	X	X

Field name	Description, possible values and format	„Sale“ transaction	PreAuth*	PostAuth*	Void	PayerAuth**
hash	This is a SHA hash of the following fields: storename + txndatetime + chargetotal + currency + sharedsecret. Note, that it is important to have the hash generated in this exact order. An example of how to generate a SHA-256 hash is given in Appendix I. Either hash SHA256 or hash SHA512 should be used, not both parameters.	X	X	X	X	X
storename	This is the ID of the store provided by First Data.	X	X	X	X	X
mode	‘fullpay’, ‘payonly’ or ‘payplus’ (the chosen mode for the transaction when using the ‘classic’ checkout option)	X	X			
chargetotal	This is the total amount of the transaction using a dot or comma as decimal separator, e. g. 12.34 for an amount of 12 Euro and 34 Cent. Group separators like1,000.01 / 1.000,01 are not allowed.	X	X	X	X	X
currency	The numeric ISO code of the transaction currency, e. g. 978 for Euro (see examples in Appendix IV)	X	X	X		X
oid	The order ID of the initial action a PostAuth shall be initiated for.			X		
ipgTransactionId or merchantTransactionId	Exact identification of a transaction that shall be voided. You receive this value as result parameter, ‘ipgTransactionId’ of the corresponding transaction. Alternatively ‘merchantTransactionId’ can be used for the Void in case the merchant has assigned one.				X	

* The transaction types ‘preauth’ and ‘postauth’ only apply to the payment methods credit card and PayPal

** The transaction type ‘payer_auth’ is only required if you want to split the 3D Secure authentication process from the payment transaction (authorization) process. See more information in the 3D Secure section of this guide.

5 Optional Form Fields

Field Name	Description, possible values and format
cardFunction	This field allows you to indicate the card function in case of combo cards which provide credit and debit functionality on the same card. It can be set to ‘credit’ or ‘debit’. The field can also be used to validate the card type in a way that transactions where the submitted card function does not match the card’s capabilities will be declined. If you e.g. submit “cardFunction=debit” and the card is a credit card, the transaction will be declined.
checkoutoption	This field allows you to set the checkout option to: • ‘classic’ for a payment process that is split into multiple. • ‘combinedpage’ for a payment process where the payment method choice and the typical next step (e.g. entry of card details or selection of bank) in a consolidated in a single page. • ‘simpleform’ to just capture the sensitive data by using the simplified hosted payment form displayed within an iFrame embedded in the context of your website.
comments	Place any comments here about the transaction.
customerid	This field allows you to transmit any value, e.g. your ID for the customer.
dccInquiryId	Inquiry ID for a Dynamic Pricing request. Used to send the Inquiry ID you have obtained via a Web Service API call (Request Merchant Rate For Dynamic Pricing). This value will be used to retrieve the currency conversion information (exchange rate, converted amount) for this transaction.
dccSkipOffer	If the cardholder declines the currency conversion offer within your environment, the request parameter ‘dccSkipOffer’ can be set to ‘true’ so that the hosted consumer dialogue will automatically be skipped.
dynamicMerchantName	The name of the merchant to be displayed on the cardholder’s statement. The length of this field should not exceed 25 characters. If you want to use this field, please contact your local support team to verify if this feature is supported in your country.
invoicenumber	This field allows you to transmit any value, e.g. an invoice number or class of goods. Please note that the maximum length for this parameter is 48 characters.
hashExtended	The extended hash is an optional security feature that allows you to include all parameters of the transaction request. It needs to be calculated using all request parameters in ascending order of the parameter names.

Field Name	Description, possible values and format
hexColorCode	You can to use this parameter as an optional parameter when you use the simplified iFrame integration with the checkout option 'simpleform'. The 'hexColorCode' parameter allows you to easily align the background color of the buttons in an iFrame to the look and feel of your website. You can send e.g.: '#9c22ce' then the color of buttons will be 'violet' but when you send 'hexColorCode' set with non-existing hex color code then there won't be any impact.
hostURI	You have to use this parameter as a mandatory parameter when you use the simplified iFrame integration with the checkout option 'simpleform'. The 'hostURI' parameter allows you to send the URI of the page where an iFrame is hosted in order to dissolve an iFrame after the sensitive data has been submitted i.e. the response parameters go to the parent window that way.
item1 *up to* item999	Line items are regular Connect integration key-value parameters (URL-encoded), where: • the name is a combination of the keyword item and a number, where the number indicates the list position e.g.: item1 • the value is represented by a semicolon-separated list of values, where the position indicates the meaning of the list item property e.g.: <1>;<2>;<3>;<4>;<5>;<6>;<7> The 'item1' to 'item999' parameters allow you to send basket information in the following format: id;description;quantity;item_total_price;sub_total;vat_tax;shipping 'shipping' always has to be set to '0' for single line item. If you want to include a shipping fee for an order, please use the predefined id IPG_SHIPPING. For other fees that you may want to add to the total order, you can use the predefined id IPG_HANDLING. When you want to apply a discount, you should include an item with a negative amount and change accordingly the total amount of the order. Do not forget to regard the 'quantity' when calculating the values e.g.: subtotal and VAT since they are fixed by items. Examples: A;Product A;1;5;3;2;0 B;Product B;5;10;7;3;0 C;Product C;2;12;10;2;0 D;Product D;1;-1.0;-0.9;-0.1;0 IPG_SHIPPING;Shipping costs;1;6;5;1;0 IPG_HANDLING;Transaction fee;1;6.0;6.0;0;0

<table>
<tr><th>Field Name</th><th colspan="2">Description, possible values and format</th></tr>
<tr><td rowspan="18">language</td><td colspan="2">This parameter can be used to override the default payment page language configured for your merchant store. The following values are currently possible:</td></tr>
<tr><td><b>Language</b></td><td><b>Value</b></td></tr>
<tr><td>Chinese (simplified)</td><td>zh_CN</td></tr>
<tr><td>Chinese (traditional)</td><td>zh_TW</td></tr>
<tr><td>Czech</td><td>cs_CZ</td></tr>
<tr><td>Dutch</td><td>nl_NL</td></tr>
<tr><td>English (USA)</td><td>en_US</td></tr>
<tr><td>English (UK)</td><td>en_GB</td></tr>
<tr><td>Finnish</td><td>fi_FI</td></tr>
<tr><td>French</td><td>fr_FR</td></tr>
<tr><td>German</td><td>de_DE</td></tr>
<tr><td>Greek</td><td>el_GR</td></tr>
<tr><td>Italian</td><td>it_IT</td></tr>
<tr><td>Polish</td><td>pl_PL</td></tr>
<tr><td>Portuguese (Brazil)</td><td>pt_BR</td></tr>
<tr><td>Serbian</td><td>sr_RS</td></tr>
<tr><td>Slovak</td><td>sk_SK</td></tr>
<tr><td>Spanish</td><td>es_ES</td></tr>
<tr><td>merchantTransactionId</td><td colspan="2">Allows you to assign a unique ID for the transaction. The ID can be used to reference the transactions in a PostAuth or Void request (referencedMerchantTransactionId).</td></tr>
<tr><td>mobileMode</td><td colspan="2">If your customer uses a mobile device for shopping at your online store you can submit this parameter with the value ‘true’. This will lead your customer to a payment page flow that has been specifically designed for mobile devices.</td></tr>
<tr><td>numberOfInstallments</td><td colspan="2">This parameter allows you to set the number of instalments for a Sale if your customer pays the amount in several parts.</td></tr>
<tr><td>oid</td><td colspan="2">This field allows you to assign a unique ID for your order. If you choose not to assign an order ID, the First Data system will automatically generate one for you.</td></tr>
</table>

<table>
<tr><th>Field Name</th><th colspan="2">Description, possible values and format</th></tr>
<tr><td rowspan="10">paymentMethod</td><td colspan="2">If you let the customer select the payment method (e.g. MasterCard, Visa, Direct Debit) in your shop environment or want to define the payment type yourself, transmit the parameter ‘paymentMethod’ along with your Sale or PreAuth transaction.
If you do not submit this parameter, the Gateway will display a drop-down menu to the customer to choose from the payment methods available for your shop.</td></tr>
<tr><td><b>Payment Method</b></td><td><b>Value</b></td></tr>
<tr><td>MasterCard</td><td>M</td></tr>
<tr><td>Visa (Credit/Debit/Electron/Delta)</td><td>V</td></tr>
<tr><td>American Express</td><td>A</td></tr>
<tr><td>Diners</td><td>C</td></tr>
<tr><td>Maestro</td><td>MA</td></tr>
<tr><td>Maestro UK</td><td>maestroUK</td></tr>
<tr><td>MasterPass</td><td>Masterpass</td></tr>
<tr><td>PayPal</td><td>Paypal</td></tr>
<tr><td>ponumber</td><td colspan="2">This field allows you to submit a Purchase Order Number with up to 50 characters.</td></tr>
<tr><td>refer</td><td colspan="2">This field describes who referred the customer to your store.</td></tr>
<tr><td>referencedMerchantTransactionID</td><td colspan="2">This field allows to reference to a merchantTransactionId of a transaction when performing a Void. This can be used as an alternative to ipgTransactionId if you assigned a merchantTransactionId in the original transaction request.</td></tr>
<tr><td>responseFailURL</td><td colspan="2">The URL where you wish to direct customers after a declined or unsuccessful transaction (your Sorry URL) – only needed if not setup in Virtual Terminal / Customisation.</td></tr>
<tr><td>responseSuccessURL</td><td colspan="2">The URL where you wish to direct customers after a successful transaction (your Thank You URL) – only needed if not setup in Virtual Terminal / Customisation.</td></tr>
<tr><td>reviewOrder</td><td colspan="2">MasterPass-specific parameter for scenarios where the final amount needs to be confirmed by the customer after returning from the Wallet. Set the value for this parameter to ‘true’ in order to indicate that the final transaction amount needs to be reviewed by the cardholder.</td></tr>
</table>

Field Name	Description, possible values and format
reviewURL	MasterPass-specific parameter for scenarios where the final amount needs to be confirmed by the customer after returning from the MasterPass environment. Use this parameter to indicate where the customer shall be redirected to in order to review and complete the
shipping	This parameter can be used to submit the shipping fee, in the same format as ‘chargetotal’. If you submit ‘shipping’, the parameters ‘subtotal’ and ‘vattax’ have to be submitted as well. Note that the ‘chargetotal’ has to be equal to ‘subtotal’ plus ‘shipping’ plus ‘vattax’.
trxOrigin	This parameter allows you to use the secure and hosted payment form capabilities within your own application for Mail/Telephone Order (MOTO) payments. Possible values are ‘MOTO‘ (for transactions where you have received the order over the phone or by mail and enter the payment details yourself) and ‘ECI‘ (for standard usage in an eCommerce environment where your customer enters the payment details).
unscheduledCredentialOnFileType	Credentials on file (COF) specific parameter. This field allows you to flag transactions as unscheduled credential on file type. Currently the valid values are: FIRST, CARDHOLDER_INITIATED or MERCHANT_INITIATED to advise the scenario if the credential is stored on your side.
vattax	This field allows you to submit an amount for Value Added Tax or other taxes, e.g. GST in Australia. Please ensure the sub total amount plus shipping plus tax equals the charge total.

6 Using your own forms to capture the data

If you decide to create your own forms i. e. not to use the ones provided and hosted by First Data, there are additional mandatory fields that you need to include. These fields are listed in the following sections, depending on the mode you choose.

In addition, you should check if JavaScript is activated in your customer's browser and if necessary, inform your customer that JavaScript needs to be activated for the payment process.

6.1 payonly Mode

After your customer has decided how to pay, you present a corresponding HTML-page with a form to enter the payment data as well as hidden parameters with additional transaction information.

In addition to the mandatory fields listed above, your form needs to contain the following fields (part of them can be hidden):

Field name	Description, possible values and format	Credit Card (+ Visa Debit / Electron / Delta)	Maestro
cardnumber	Your customer's card number. 12-24 digits.	X	X
expmonth	The expiry month of the card (2 digits)	X	X
expyear	The expiry year of the card (4 digits)	X	X
cvm	The card code, in most cases on the backside of the card (3 to 4 digits)	X	X as an optional field "if on card"
iban	Your customer's IBAN - International Bank Account Number (22 digits)		
bic	Your customer's BIC – Business Identifier Code (8 or 11 digits)		

6.2 payplus Mode

Using payplus mode, it is possible to additionally transfer billing information to the Gateway. The following table describes the format of these additional fields:

Field Name	Possible Values	Description
bcompany	Alphanumeric characters, spaces, and dashes	Customers Company
bname	Alphanumeric characters, spaces, and dashes	Customers Name
baddr1	Limit of 96 characters, including spaces	Customers Billing Address 1
baddr2	Limit of 96 characters, including spaces	Customers Billing Address 2
bcity	Limit of 96 characters, including spaces	Billing City
bstate	Limit of 96 characters, including spaces	State, Province or Territory
bcountry	2 Letter Country Code	Country of Billing Address
bzip	Limit of 24 characters, including spaces	Zip or Postal Code
phone	Limit of 32 Characters	Customers Phone Number
fax	Limit of 32 Characters	Customers Fax Number
email	Limit of 254 Characters	Customers Email Address

6.3 fullpay Mode

Using fullpay mode, it is possible to additionally transfer shipping information to the Gateway. The billing information is as specified above. The following table describes the format of the shipping fields:

Field Name	Possible Values	Description
sname	Alphanumeric characters, spaces, and dashes	Ship-to Name
saddr1	Limit of 96 characters, including spaces	Shipping Address Line 1
saddr2	Limit of 96 characters, including spaces	Shipping Address Line 2
scity	Limit of 96 characters, including spaces	Shipping City
sstate	Limit of 96 characters, including spaces	State, Province or Territory
scountry	2 letter country code	Country of Shipping Address
szip	Limit of 24 characters, including spaces	Zip or Postal Code

6.4 Validity checks

Prior to the authorisation request for a transaction, the Gateway performs the following validation checks:

- The expiry date of cards needs to be in the future
- The Card Security Code field must contain 3 or 4 digits
- The structure of a card number must be correct (LUHN check)

If the submitted data should not be valid, the Gateway presents a corresponding data entry page to the customer.

To avoid this hosted page when using your own input forms for the payment process, you can transmit the following additional parameter along with the transaction data:

```
full_bypass=true
```

In that case you get the result of the validity check back in the transaction response and can display your own error page based on this.

Please note, if the transaction is eligible for DCC (your store is configured for DCC and the customer is paying by credit card capable of DCC), your customer will be presented the DCC page despite having full_bypass set to true. This is due to regulatory reasons. You can avoid displaying of DCC choice pages by doing the DCC Inquiry yourself via our Web Service API (Request Merchant Rate For Dynamic Pricing).

7 Additional Custom Fields

You may want to use further fields to gather additional customer data geared toward your business specialty, or to gather additional customer demographic data which you can then store in your own database for future analysis. You can send as many custom fields to the Gateway as you wish and they will get returned along with all other fields to the response URL.

Up to ten custom fields can be submitted in a way that they will be stored within the Gateway so that they appear in the Virtual Terminal's Order Detail View as well as in the response to Inquiry Actions that you send through our Web Service API . If you want to use this feature, please send the custom fields in the format customParam_key=value.

Example:

```
      <input type="hidden" name="customParam_color" value="green"/>
```

8 3D Secure

The Connect solution includes the ability to authenticate transactions using Verified by Visa, MasterCard SecureCode. If your credit card agreement includes 3D Secure and your Merchant ID has been activated to use this service, you do not need to modify your payment page.

If you are enabled to submit 3D Secure transactions but for any reason want to submit specific transactions without using the 3D Secure protocol, you can use the additional parameter authenticateTransaction and set it to either “true” or “false”.

Example for a transaction without 3D Secure:

```
      <input type="hidden" name="authenticateTransaction" value="false"/>
```

In principle, it may occur that 3D Secure authentications cannot be processed successfully for technical reasons. If one of the systems involved in the authentication process is temporarily not responding, the payment transaction will be processed as a “regular” eCommerce transaction (ECI 7). **A liability shift to the card issuer for possible chargebacks is not warranted in this case**. If you prefer that such transactions shall not be processed at all, our technical support team can block them for your Store on request.

Credit card transactions with 3D Secure hold in a pending status while cardholders search for their password or need to activate their card for 3D Secure during their shopping experience. During this time when the final transaction result of the transaction is not yet determined, the Gateway sets the Approval Code to „?:waiting 3dsecure“. If the session expires before the cardholder returns from the 3D Secure dialogue with his bank, the transaction will be shown as “N:-5103:Cardholder did not return from ACS”.

Please note that the technical process of 3D Secure transactions differs in some points compared to a normal transaction flow. If you already have an existing shop integration and plan to activate 3D Secure subsequently, we recommend performing some test transactions on our test environment.

8.1 3DSecure Split Authentication

If your business or technical processes require the cardholder authentication to be separated from the payment transaction (authorization), you can use the transaction type ‘payer_auth’. This transaction type only performs the authentication (and stores the authentication results).

Example of a ‘payer_auth’ request:

```
<!-- #include file="ipg-util.asp"-->

<html>
<head><title>IPG Connect Sample for ASP</title></head>
<body>
<p><h1>Order Form</h1></p>

<form method="post" action=" https://test.ipg-
online.com/connect/gateway/processing ">
    <input type="hidden" name="txntype" value="payer_auth">
                <input type="hidden" name="timezone"
value="Europe/Berlin"/>
                <input type="hidden" name="txndatetime" value="<%
getDateTime() %>"/>
                <input type=”hidden” name=”hash_algorithm”
value=”SHA256”/>
                <input type="hidden" name="hash" value="<% call
createHash( "13.00","826" ) %>"/>
                <input type="hidden" name="storename"
value="10123456789" />
    <input type="hidden" name="mode" value="payonly"/>
                <input type="hidden" name="paymentMethod"
value="M"/>
    <input type="text" name="chargetotal" value="13.00" />
    <input type="hidden" name="currency" value="826"/>
                <input type="hidden"
name="authenticateTransaction" value="true"/>
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Example of a ‘payer_auth’ response:

```
{txndate_processed=10/04/17 13:37:33,
ccbin=542606,
timezone=CET,
oid=C-2101f68a-45e9-4f3c-a6da-1337d5574717,
cccountry=N/A,
expmonth=12,
currency=826,
chargetotal=13.99,
approval_code=Y:ECI2/5:Authenticated,
hiddenSharedsecret=sharedsecret,
hiddenTxndatetime=2017:04:10-13:37:08,
expyear=2024,
response_hash=927d3c3108d596c593f74fd79184ece7c33103fe,
response_code_3dsecure=1,
hiddenStorename=120995000,
transactionNotificationURL=https://test.ipg-
online.com/webshop/transactionNotification,
tdate=1491824253,
ignore_refreshTime=on,
ccbrand=MASTERCARD,
txntype=payer_auth,
paymentMethod=M,
txndatetime=2017:04:10-13:37:08,
cardnumber=(MASTERCARD) ... 4979,
ipgTransactionId=84120276797,
status=APPROVED}
```

In a second step, you need to submit a payment transaction (‘sale’ or ‘preauth’) via the IPG Web Service API and reference it to the prior authentication. To review an example of a ‘sale’ transaction that refers to a previous ‘payer_auth’ transaction, please review the 3DSecure Split Authentication section, in the Web Service API integration guide.

9 MCC 6012, 6051 & 7299 Mandate (UK)

For UK-based Financial Institutions with Merchant Category Code 6012, Visa and Mastercard have mandated additional information of the primary recipient of the loan to be included in the authorisation message.

Field Name	Description
mcc6012BirthDay	Date of birth in format dd.mm.yyyy
mcc6012AccountFirst6	First 6 digits of recipient PAN (where the primary recipient account is a card)
mcc6012AccountLast4	Last 4 digits of recipient PAN (where the primary recipient account is a card)
mcc6012AccountNumber	Recipient account number (where the primary recipient account is not a card)
mcc6012Surname	Surname
mcc6012Zip	Post Code

If you are a UK 6051 and 7299 merchant, you can reuse the MCC 6012 parameters to send the optional data to be included in the authorization message. However, please note that you have to either populate all the parameters or none otherwise the transaction will be declined.

10 Data Vault

With the Data Vault product option you can store sensitive cardholder data in an encrypted database in First Data's data centre to use it for subsequent transactions without the need to store this data within your own systems.

If you have ordered this product, the Connect solution offers you the following functions:

Store or update payment information when performing a transaction

- Additionally send the parameter 'hosteddataid' together with the transaction data as a unique identification for the payment information in this transaction. Depending on the payment type, credit card number and expiry date it will be stored under this ID if the transaction has been successful. In cases where the submitted 'hosteddataid' already exists for your store, the stored payment information will be updated.

  If you want to assign multiple IDs to the same payment information record, you can submit the parameter 'hosteddataid' several times with different values in the same transaction.

  If you prefer not to assign a token yourself but want to let the Gateway do this for you, send the parameter 'assignToken' and set it to 'true'. The Gateway will then assign a token and include it in the transaction response as 'hosteddataid'.

  If you have use cases where you need some of the tokens for single transactions only (e.g. for consumers that check out as a "guest", use the additional parameter 'tokenType' with the values 'ONETIME' (card details will only be stored for a short period of time) or 'MULTIPAY' (card details will be stored for use in future transactions).

Initiate payment transactions using stored data

- If you stored cardholder information using the Data Vault option, you can perform transactions using the 'hosteddataid' without the need to pass the credit card or bank account data again.
  Please note that it is not allowed to store the card code (in most cases on the back of the card) so that for credit card transactions, the cardholder still needs to enter this value. If you use First Data's hosted payment forms, the cardholder will see the last four digits of the stored credit card number, the expiry date and a field to enter the card code.

  When using multiple Store IDs, it is possible to access stored card data records of a different Store ID then the one that has been used when storing the record. In that way you can for example use a shared data pool for different distributive channels. To use this feature, submit the Store ID that has been used when storing the record as the additional parameter 'hosteddatastoreid'.

Avoid duplicate cardholder data for multiple records

- To avoid customers using the same cardholder data for multiple user accounts, the additional parameter 'declineHostedDataDuplicates' can be sent along with the request. The valid values for this parameter are 'true'/'false'. If the value for this parameter is set to 'true' and the cardholder data in the request is already found to be associated with another 'hosteddataid', the transaction will be declined.

  See further possibilities with the Data Vault product in the Integration Guide for the Web Service API.

11 Recurring Payments

For credit card and PayPal transactions, it is possible to install recurring payments using Connect. To use this feature, the following additional parameters will have to be submitted in the request:

Field Name	Possible Values	Description
recurringInstallmentCount	Number between 1 and 999	Number of instalments to be made including the initial transaction submitted
recurringInstallmentPeriod	day week month year	The periodicity of the recurring payment
recurringInstallmentFrequency	Number between 1 and 99	The time period between instalments
recurringComments	Limit of 100 characters, including spaces	Any comments about the recurring transaction

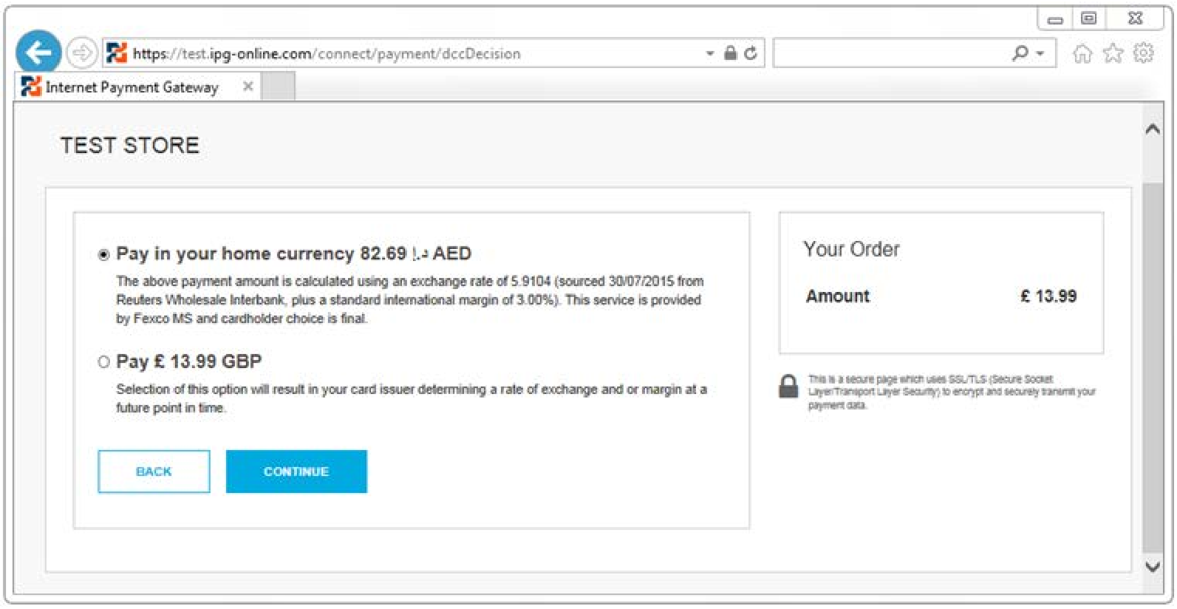
Note that the start date of the recurring payments will be the current date and will be automatically calculated by the system.

The recurring payments installed using Connect can be modified or cancelled using the Virtual Terminal or Web Service API.

12 Dynamic Currency Conversion and Dynamic Pricing

With First Data's Dynamic Currency Conversion, foreign customers have the choice to pay for goods and services purchased online in their home currency when using their Visa or MasterCard credit card for the payment. The currency conversion is quick and eliminates the need for customers to mentally calculate the estimated cost of the purchase in their home currency. International Visa and MasterCard eCommerce customers can make informed decisions about their online purchases and eradicate any unexpected pricing or foreign exchange conversions on receipt of their monthly statements.

If your Store has been activated for this product option, the Connect solution automatically offers a currency choice to your customers if the card they use has been issued in a country with a currency that is different to your default currency.



TEST STORE

● Pay in your home currency 82.69 د.إ AED

The above payment amount is calculated using an exchange rate of 5.9104 (sourced 30/07/2015 from Reuters Wholesale Interbank, plus a standard international margin of 3.00%). This service is provided by Fexco MS and cardholder choice is final.

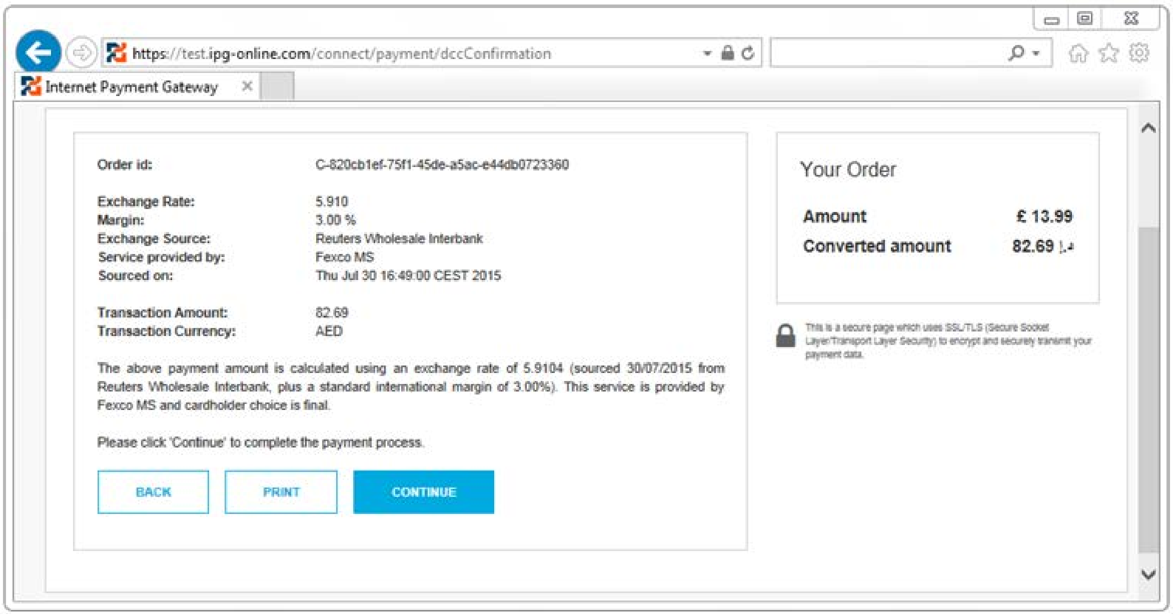
○ Pay £ 13.99 GBP

Selection of this option will result in your card issuer determining a rate of exchange and or margin at a future point in time.

Your Order

Amount	£ 13.99

This is a secure page which uses SSL/TLS (Secure Socket Layer/Transport Layer Security) to encrypt and securely transmit your payment data.



Order id:	C-820cb1ef-75f1-45de-a5ac-e44db0723360
Exchange Rate:	5.910
Margin:	3.00 %
Exchange Source:	Reuters Wholesale Interbank
Service provided by:	Fexco MS
Sourced on:	Thu Jul 30 16:49:00 CEST 2015
Transaction Amount:	82.69
Transaction Currency:	AED

The above payment amount is calculated using an exchange rate of 5.9104 (sourced 30/07/2015 from Reuters Wholesale Interbank, plus a standard international margin of 3.00%). This service is provided by Fexco MS and cardholder choice is final.

Please click 'Continue' to complete the payment process.

Your Order

Amount	£ 13.99
Converted amount	82.69 د.إ

This is a secure page which uses SSL/TLS (Secure Socket Layer/Transport Layer Security) to encrypt and securely transmit your payment data.

Please note that for compliance reasons First Data’s Global Choice can only be offered on transactions that take place in full at that time (e.g. Sale, Refund) and not on any delayed settlement (e.g. pre/post auth, recurring) due to the fluctuation of the rate of exchange.

Another option for your foreign customers is to display all pricing within your online store in their home currency using our Dynamic Pricing solution. This solution removes the need for your company to set pricing in any other currency other than your home currency.
Please see the Integration Guide for our Web Service API for details on how to request the exchange rates.

If your Store has been activated for this product option and you want to submit the payment transaction via our Connect solution, you need to send the DCC Inquiry ID that you have received along with the exchange rate request in the parameter ‘dccInquiryId’.

You can also use the ‘dccInquiryId’ for cases where Global Choice is being offered and handled on your side (e.g. within a mobile app). If the cardholder declines the currency conversion offer within your environment, the request parameter dccSkipOffer can be set to ‘true’ so that the hosted consumer dialogue will automatically be skipped.

13 Transaction Response

13.1 Response to your Success/Failure URLs

Upon completion, the transaction details will be sent back to the defined 'responseSuccessURL' or 'responseFailURL' as hidden fields:

Field Name	Description
approval_code	Approval code for the transaction. The first character of this parameter is the most helpful indicator for verification of the transaction result. 'Y' indicates that the transaction has been successful 'N' indicates that the transaction has not been successful "?" indicates that the transaction has been successfully initialised, but a final result is not yet available since the transaction is now in a waiting status. The transaction status will be updated at a later stage.
oid	Order ID
refnumber	Reference number
status	Transaction status, e.g. 'APPROVED', 'DECLINED' (by authorisation endpoint or due to fraud prevention settings) or 'FAILED' (wrong transaction message content/parameters, etc.) or 'WAITING' (asynchronous Alternative Payment Methods)
txndate_processed	Time of transaction processing
ipgTransactionId	Transaction identifier assigned by the Gateway, e.g. to be used for a Void
tdate	Identification for the specific transaction
fail_reason	Reason the transaction failed
response_hash	Hash-Value to protect the communication (see note below)
processor_response_code	The response code provided by the backend system. Please note that response codes can be different depending on the used payment type and backend system. While for credit card payments, the response code '00' is the most common response for an approval.
fail_rc	Internal processing code for failed transactions
terminal_id	Terminal ID used for transaction processing
ccbin	6 digit identifier of the card issuing bank
cccountry	3 letter alphanumeric ISO code of the cardholder's country (e.g. USA, DEU, ITA, etc.) Filled with "N/A" if the cardholder's country cannot be determined or the payment type is not credit card
ccbrand	Brand of the credit or debit card: MASTERCARD VISA AMEX DINERS CLUB/DISCOVER MAESTRO Filled with "N/A" for any payment method which is not a credit card or debit card

For 3D Secure transactions only:

Field Name	Description
response_code_3dsecure	Return code indicating the classification of the transaction: 1 – Successful authentication (VISA ECI 05, MasterCard ECI 02) 2 – Successful authentication without AVV (VISA ECI 05, MasterCard ECI 02) 3 – Authentication failed / incorrect password (transaction declined) 4 – Authentication attempt (VISA ECI 06, MasterCard ECI 01) 5 – Unable to authenticate / Directory Server not responding (VISA ECI 07) 6 – Unable to authenticate / Access Control Server not responding (VISA ECI 07) 7 – Cardholder not enrolled for 3D Secure (VISA ECI 06) 8 – Invalid 3D Secure values received, most likely by the credit card issuing bank's Access Control Server (ACS) *Please see note about blocking ECI 7 transactions in the 3D Secure section of this document.*

For Dynamic Currency Conversion transactions only:

Field Name	Description
dcc_foreign_amount	Converted amount in cardholder home currency. Decimal number with dot (.) as a decimal separator.
dcc_foreign_currency	ISO numeric code of the cardholder home currency. This transaction is performed in this currency. String.
dcc_margin_rate_percentage	Percent of margin applied to the original amount. Decimal number with dot (.) as a decimal separator.
dcc_rate_source	Name of the exchange rate source (e.g. Reuters Wholesale Inter Bank). String.
dcc_rate	Exchange rate. Decimal number with dot (.) as a decimal separator.
dcc_rate_source_timestamp	Exchange rate origin time. Integer - Unix timestamp (seconds since 1.1.1970).
dcc_accepted	Indicates if the card holder has accepted the conversion offer (response value 'true') or declined the offer (response value 'false')

For MasterPass transactions only:

redirectURL	When reviewOrder has been set to 'true', the response contains the URL that you need to finalize the transaction

Additionally when using your own error page for negative validity checks (full_bypass=true):

fail_reason_details	Comma separated list of missing or invalid variables. Note that 'fail_reason_details' will not be supported in case of payplus and fullpay mode.
invalid_cardholder_data	**true** – if validation of card holder data was negative **false** – if validation of card holder data was positive but transaction has been declined due to other reasons

In addition, your custom fields and billing/shipping fields will also be sent back to the specific URL.

Please consider when integrating that new response parameters may be added from time to time in relation to product enhancements or new functionality.

The parameter ‘response_hash’ allows you to recheck if the received transaction response has really been sent by First Data and can therefore protect you from fraudulent manipulations. The value is created with a SHA Hash using the following parameter string:

```
sharedsecret + approval_code  + chargetotal + currency + txndatetime
+ storename
```

The hash algorithm is the same as the one that you have set in the transaction request.

Please note that you have to implement the response hash validation, when doing so remember to store the ‘txndatetime’ that you have submitted with the transaction request in order to be able to validate the response hash. Furthermore, you must always use the https-connection (instead of http) to prevent eavesdropping of transaction details.

13.2 Server-to-Server Notification

In addition to the response you receive in hidden fields to your ‘responseSuccessURL’ or ‘responseFailURL’, the Gateway can send server-to-server notifications with the above result parameters to a defined URL. This is especially useful to keep your systems in synch with the status of a transaction. To use this notification method, you can specify an URL in the Customisation section of the Virtual Terminal or submit the URL in the following additional transaction parameter ‘transactionNotificationURL’.

Please note that:

- The Transaction URL is sent as received therefore please don’t add additional escaping (e.g. using %2f for a Slash (/).
- No SSL handshake, verification of SSL certificates will be done in this process.
- The Notification URL needs to listen either on port 80 (http) or port 443 (https) – other ports are not supported.
- The response hash parameter for validation (using the same algorithm that you have set in the transaction request) ‘notification_hash’ is calculated as follows:

```
chargetotal + sharedsecret + currency + txndatetime + storename
+ approval_code.
```

Such notifications can also be set up for Recurring Payments that get automatically triggered by the Gateway. Please contact your local support team to get a Shared Secret agreed for these notifications. You can configure your Recurring Transaction Notification URL in the Virtual Terminal (Customisation/ Store URL Settings). In case of the recurring transactions the hash parameter is calculated as follows:

```
chargetotal  +  rcpSharedSecret+  currency  +  txndatetime  +
storename + approval_code
```

Appendix I – How to generate SHA-256 Hash or Extended Hash

Example

- storename = 98765432101
- txndatetime = 2013:07:16-09:57:08
- chargetotal = 1.00
- currency = 826
- sharedsecret = TopSecret

Step 1. Collect selected parameters: storename, txndatetime, chargetotal, currency and sharedsecret and join the parameters' values to one string (use only parameters' values and not the parameters' names).

98765432101 2013:07:16-09:57:08 1.00 826 TopSecret

Step 2. Convert the created string to its ascii hexadecimal representation.

3938373635343332313031323031333a30373a31362d30393a35373a3038312e3030383236546f70536563726574

Step 3. Pass the ascii hexadecimal representation of the concatenated string to the SHA-256 algorithm

SHA256(3938373635343332313031323031333a30373a31362d30393a35373a3038312e3030383236546f70536563726574)

Step 4. Use the value returned by the SHA-256 algorithm and submit it to our Gateway in the given form.

3d7e75aa0b4e0e1d4a7ac87e451e64692cced46f4358ef35a69d96721341243c

```
	<input type="hidden" name="hash"  value="
3d7e75aa0b4e0e1d4a7ac87e451e64692cced46f4358ef35a69d96721341243c
"/>
```

Example of Extended Hash

- P1 = abc
- P2 = xyz
- P3 = ccc
- sharedsecret = TopSecret
- t1=zzz
- t2=yyy

Step 1. Extended hash needs to be calculated using all request parameters in ascending order of the parameter names, adding sharedsecret at last. Join the parameters' values to one string (use only parameters' values and not the parameters' names).

abcxyzccczzzyyyTopSecret

Step 2. Convert the created string to its ascii hexadecimal representation.

39383736353433323130313230313333a30373a31362d30393a35373a3038312e3030383236546f70536563726574

Step 3. Pass the ascii hexadecimal representation of the created string to the SHA-256 algorithm.

SHA256(39383736353433323130313230313333a30373a31362d30393a35373a3038312e3030383236546f70536563726574)

Step 4. Use the value returned by the SHA-256 algorithm and submit it to our payment gateway in the given form.

3d7e75aa0b4e0e1d4a7ac87e451e64692cced46f4358ef35a69d96721341243c

```
<input type="hidden" name="hashExtended"
value="3d7e75aa0b4e0e1d4a7ac87e451e64692cced46f4358ef35a69d96721341243c"/
>
```

Appendix II – ipg-util.asp

```
<!-- sha1.js contains also helper functions (dateFormatter, charToByte,
byteToHex, ...) -->
<script LANGUAGE=JScript RUNAT=Server src="sha1.js">
</script>
<!-- google CryptoJS for SHA256 -->
<script LANGUAGE=JScript RUNAT=Server src="sha256.js">
</script>
<script LANGUAGE=JScript RUNAT=Server>
    var today = new Date();
    var formattedDate = today.formatDate("Y:m:d-H:i:s");

    /*
        Function that calculates the hash of the following parameters:
        - Store Id
        - Date/Time(see $dateTime above)
        - chargetotal
        - currency (numeric ISO value)
        - shared secret
    */
    function createHash(chargetotal, currency) {
        // Please change the store Id to your individual Store ID
        var storeId = "10123456789";
        // NOTE: Please DO NOT hardcode the secret in that script. For example
read it from a database.
        var sharedSecret = "sharedsecret";

        var stringToHash = storeId + formattedDate + chargetotal + currency +
sharedSecret;

        var ascii = getHexFromChars(stringToHash);

        var hash = CryptoJS.SHA256(ascii);

        Response.Write(hash);
    }

    function getHexFromChars(value) {
        var char_str = value;
        var hex_str = "";
        var i, n;
        for(i=0; i < char_str.length; i++) {
            n = charToByte(char_str.charAt(i));
            if(n != 0) {
                hex_str += byteToHex(n);
            }
        }
        return hex_str.toLowerCase();
    }

    function getDateTime() {
        Response.Write(formattedDate);
    }
</script>
```

Appendix III – ipg-util.php

```
<?php
    // Timezeone needs to be set
    date_default_timezone_set('Europe/Berlin');
    $dateTime = date("Y:m:d-H:i:s");

    function getDateTime() {
        global $dateTime;
        return $dateTime;
    }

    /*
        Function that calculates the hash of the following parameters:
        - Store Id
        - Date/Time(see $dateTime above)
        - chargetotal
        - currency (numeric ISO value)
        - shared secret
    */
    function createHash($chargetotal, $currency) {
        // Please change the store Id to your individual Store ID
        $storeId = "10123456789";
        // NOTE: Please DO NOT hardcode the secret in that script. For example read
it from a database.
        $sharedSecret = "sharedsecret";

        $stringToHash = $storeId . getDateTime() . $chargetotal . $currency .
$sharedSecret;

        $ascii = bin2hex($stringToHash);

        return hash("sha256", $ascii);
    }
```

Appendix IV – Currency Code List

Currency name	Currency code	Currency number
Aruban Florin	AWG	533
Australian Dollar	AUD	036
Bahamian Dollar	BSD	044
Bahrain Dinar	BHD	048
Barbados Dollar	BBD	052
Belarusian Ruble	BYR	933
Belize Dollar	BZD	084
Bolívar Soberano	VES	928
Brazilian Real	BRL	986
Burundi Franc	BIF	108
Canadian Dollar	CAD	124
Cayman Islands Dollar	KYD	136
Chinese Renmibi	CNY	156
Croatian Kuna	HRK	191
Czech Koruna	CZK	203
Danish Krone	DKK	208
Dominican Peso	DOP	214
East Caribbean Dollar	XCD	951
Euro	EUR	978
Guyanese Dollar	GYD	328
Hong Kong Dollar	HKD	344
Hungarian Forint	HUF	348
Indian Rupee	INR	356
Israeli New Shekel	ISL	376
Jamaican Dollar	JMD	388
Japanese Yen	JPY	392
Kuwaiti Dinar	KWD	414
Lithuanian Litas	LTL	440
Malaysian Ringgit	MYR	458
Mexican Peso	MXN	484
Netherlands Antillean Guilder	ANG	532
New Zealand Dollar	NZD	554
Norwegian Krone	NOK	578
Omani Rial	OMR	512
Polish Zloty	PLN	985
Pound Sterling	GBP	826
Romanian New Leu	RON	946
Russian Ruble	RUB	643
Saudi Rihal	SAR	682
Serbian Dinar	RSD	941
Singapore Dollar	SGD	702
South African Rand	ZAR	710
South Korean Won	KRW	410
Surinamese Dollar	SRD	968
Swedish Krona	SEK	752
Swiss Franc	CHF	756
Taiwan Dollar	TWD	901
Trinidad and Tobago Dollar	TTD	780
Turkish Lira	TRY	949
UAE Dirham	AED	784
US Dollar	USD	840

Appendix V – PayPal

Refer to the following information when integrating PayPal as a payment method.

Transaction types mapping

Connect Transaction Type (txntype)	PayPal operation
sale	SetExpressCheckoutPayment *(sets PaymentAction to Authorization in SetExpressCheckout and DoExpressCheckoutPayment requests)*
preauth	GetExpressCheckoutDetails
sale – with additional parameters for installing a Recurring Payment	DoExpressCheckoutPayment*
postauth	DoCapture (,DoReauthorization)
void	DoVoid

Address handling

If you pass a complete set of address values within your request to Connect (name, address1, zip, city and country within billing and/or shipping address), these values will be forwarded to PayPal, setting the PayPal parameter ‘addressOverride’ to ‘1’.

Please note that it is an eligibility requirement for PayPal’s Seller Protection that the shipping address will be submitted to PayPal.

If you submit no or incomplete address data within the Connect request, no address data will be forwarded to PayPal and the PayPal parameter ‘addressOverride’ will not be set.

Regardless of that logic, the Gateway will always store the shipTo address fields received from PayPal in the GetDetails request in the ShippingAddress fields, possibly overwriting values passed in the request to Connect (such overwriting depends on the above logic).

* If you want to use PayPal’s Reference Transactions feature for recurring payments, please contact PayPal upfront to verify if your PayPal account meets their requirements for this feature.

Recurring Payment Transaction

You have to submit a SALE transaction request with the corresponding parameters to install the recurring payments. The first transaction is always conducted immediately along with the request.

The subsequent transactions are executed by the Gateway’s scheduler, via the API Web Service, as defined during the initial SALE transaction with the installation.

Appendix VI – MasterPass

Refer to the following information when integrating MasterPass as a payment method.

MasterPass is a digital wallet solution provided by participating banks and supported by MasterCard. When purchasing online, customers log in to their MasterPass account and select a stored card for the payment. MasterPass allows users to store MasterCard, Maestro, VISA, American Express and Diners cards. Please note that your customers will however only be able to select the card brands that your Store has been set up for in general.

To learn more about MasterPass, please visit www.masterpass.com.

Checkout Process with MasterPass

The checkout process with MasterPass can be initiated with a "BUY WITH MasterPass" button that you place on your website either as a specifically alternative checkout option or next to other payment methods that you offer.

When consumers click this button, you construct a 'sale' or 'preauth' request with the parameter 'paymentMethod' set to 'masterpass'.

This will take your customer to the MasterPass login screen, from there to the subsequent pages of the digital wallet and finally back to your web shop (responseSuccessURL, responseFailURL or reviewURL).

Alternatively you can let your customers select the payment method on the Gateway's hosted payment method selection page. If you prefer that option, simply do not submit the parameter 'paymentMethod'.

Good to know prior the integration

- The **Billing Address** for a MasterPass transaction is associated with the card stored inside the wallet thus even if you should use the Gateway's 'payplus' or 'fullpay' mode, there will be no additional entry form for the Billing Address when a customer uses MasterPass. The Billing Address stored in the wallet will also automatically override any billing address data you may send within your transaction request to the Gateway. You will always receive the Billing Address from the wallet in the transaction response - even in 'payonly' mode, which is different compared to other payment methods.

- If you use the Gateway's 'fullpay' mode, the **Shipping Address** can be selected by the customer inside the wallet (no additional page for that from the Gateway). If you use the 'payonly' or 'payplus' mode, the Shipping Address selection in the wallet gets omitted as a non-required step. Thus you can send the Shipping Address with your request and your customers will not have to select/provide it again inside the wallet (it reduces the number of steps in the transaction flow when purchasing e.g. software products available as downloads where no shipping address is really required).

- For the cases where the shipping address and thus **Shipping Fee** is not clear yet when your customer enters the wallet process by clicking the 'BUY WITH Masterpass' button, you can send additional parameters in your transaction request which allow you to present a final confirmation page with the final amount to your customers when they return from the wallet. The parameter 'reviewOrder' needs to be set to 'true' in order to indicate that the final transaction amount needs to be reviewed by your customer before completion. In addition you will need to provide the URL for your confirmation page in the parameter ‘reviewURL’. When your customer confirms the final amount on this page, you will need to send a request to finalise the transaction to the ‘redirectURL’ that you received in your response from the Gateway. This final request needs to include: oid, ipgTransactionId, subtotal, shipping, vattax, chargetotal, currency and hashExtended.

Note that you can also set a static ‘reviewURL’ via the Virtual Terminal (in Customisation/Online store integration/Define the URLs for the integration with your online store section).

- When your Store is activated for **3D Secure**, these settings will also apply to your MasterPass transactions. In the specific case of MasterPass, the authentication process will however be handled by MasterCard inside the wallet (MasterPass Advanced Checkout), where supported programmes are limited to MasterCard SecureCode and Verified by Visa (no American Express SafeKey). However, the parameter ‘authenticateTransaction’ can also be used to dynamically steer the behaviour for MasterPass e.g. depending on the purchase amount. If you submit the parameter 'authenticateTransaction’ and set it to ‘false', the MasterPass transaction will be initiated using the MasterPass Basic Checkout which doesn't include 3D Secure authentication.

Note that merchants requesting liability shift for MasterPass transactions should use the MasterPass Advanced Checkout/3D Secure and must enable 3D Secure service such that it is invoked within the MasterPass wallet.

- The **Card Code** (CVV2/CVC2/4DBC) is not required for MasterPass transactions unless otherwise required in network rules. At the time when a customer adds a card to the wallet, the Card Code gets entered and checked once. No further Card Code entry is required from your customers. Requesting a CVC2/CVV/4DBC is allowed when required by network rules.

- **Address Verification Service** (AVS) is handled for MasterPass transactions in the same way as for any other card transaction, however as the billing address is associated with a card and stored inside the wallet, the AVS result is based on the address stored inside the wallet and not the billing address provided by your customer in your web shop.

- MasterPass is not available for **Betting/Casino Gambling** merchants (MCC 7995).

Activate MasterPass for your Test Store

- Obtain the credentials for the sandbox consumer accounts listed in the online documentation provided by MasterCard.

- Make sure your Gateway test Store ID has been enabled for MasterPass.

Appendix VII – Code Examples

Refer to the following information when integrating the simplified hosted payment form with the checkout option ‘simpleform’.

When building a request with the checkout option ‘simpleform’, a part from the mandatory fields required for every payment request, you will also need to include some specific fields in your transaction request.

The specific fields to be considered:

Field Name	Description, possible values and format
checkoutoption	Set the value for this parameter to ‘simpleform’
hostURI	Set the value for this parameter to the URI (with the upper case ”i”) of the page where an iFrame is hosted.
hexColorCode	Set the value for this parameter when you want to align the background color of the buttons in an iFrame to the look and feel of your website.

Example of a form with the minimum number of fields:

```
<form id=”checkoutForm” target=”myFrame” method="post" action="https://test.ipg-online.com/connect/gateway/processing">
<input type="hidden" name="checkoutoption“ value=“simpleform“>
<input type="hidden" name="hostURI“ value=“https://www.merchant.com/.../.../“>
<input type="hidden" name="txntype" value="preauth">
<input type="hidden" name="timezone" value="Europe/Berlin"/>
<input type="hidden" name="txndatetime" value="<% getDateTime() %>"/>
<input type="hidden” name="hash_algorithm" value="SHA256"/>
<input type="hidden" name="hash" value="<% call createHash( "13.00","978" ) %>"/>
<input type="hidden" name="storename" value="12123456789" />
<input type="hidden" name="chargetotal" value="13.00" />
<input type="hidden" name="currency" value="978"/>
<input type="submit" value="Submit">
</form>
```

JSP example showing how the simplified hosted payment form is hosted in an iFrame:

```
<div id="embeddableConnect">
 <table border="0" cellpadding="0" cellspacing="0" width="100%">
  <tbody>
   <tr>
    <iframe name="myFrame" id="myFrame" src="#" width="460px" height="900px"
style="border: none;">
                      Your browser does not support inline frames.
    </iframe>
   </tr>
  </tbody>
 </table>
</div>
```

JavaSript example showing the submission of the request to the gateway via the simplified hosted payment form and the need to add the event listeners:

```
function submitForm() {
       ----
       ----

        var obj1 = document.getElementById('checkoutoption_simpleform');
        if(obj1.checked){
                document.myForm.target = "myFrame";
                obj2.style.visibility = 'visible';
                obj2.style.display = 'block';
                window.addEventListener("message", receiveMessage, false);
                document.myForm.submitFormBtn.disabled = true;
        }

        -----
        -----

}
function receiveMessage(event){
        var hostName = getHostName();
        if (event.origin != "https://"+getHostName())
                return;
        forwardForm(event.data);
}

function forwardForm(responseObj) {
        var newForm = document.createElement("form");
        newForm.setAttribute('method',"post");
        newForm.setAttribute('action',responseObj.redirectURL);
        newForm.setAttribute('id',"newForm");
        newForm.setAttribute('name',"newForm");
        document.body.appendChild(newForm);
        var elementArr = responseObj.elementArr;
        for(j=0 ; j<elementArr.length; j++){
                var element = elementArr[j];
                var input = document.createElement("input");
                input.setAttribute("type", "hidden");
                input.setAttribute("name", element.name);
                input.setAttribute("value", element.value);
                document.newForm.appendChild(input);
        }
        document.newForm.submit();
}
```

Note that you need to register a method e.g. 'receiveMessage' with the listener. To learn more please review the links regarding Javascript MessageEvent and the ways of adding the event listeners:

- https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener
- https://developer.mozilla.org/en-US/docs/Web/API/MessageEvent

Once the transaction is completed, the gateway uses window.postMessage() method to send the sensitive transaction data back to the parent window. Then receiveMessage() method is called asynchronously.

The receiveMessage() creates a new form with all the sensitive transaction data and submits it to the preconfigured successful/failure URL while dissolving the iFrame.

The window.postMessage() method safely enables cross-origin communication between the parent and the child window objects i.e. between the merchant’s webpage and the child iFrame embedded within it. Generally, scripts on different pages are allowed to access each other if and only if the pages they originate from share the same protocol, port number, and host. window.postMessage() provides a controlled mechanism to securely circumvent this restriction.

First Data™