First Data.

Connect
Integration Guide

Version 2018-2 (IPG)

Integration Guide Connect

Version 2018-2 (IPG)

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 usage with our distribution channels in Europe, Asia, Australia, Latin America and Africa.

For information about settings, customization, 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 webshop.

When making decisions on your way of integration, please consider that some Alternative Payment methods (e.g. iDEAL) do not allow displaying their screens within an iFrame.

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 want to get collected by one of the gateway's hosted forms or what you want to collect yourself within your webshop environment.

You can e.g. let customers select their preferred payment method within your webshop 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 that shall be captured by the payment 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 payment gateway collects a full set of billing information on an additional page
- fullpay: in addition to the above, the payment gateway displays a third page to also collect shipping information

The most important aspect around the usage of 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 customized with your own logo, colors, and font types in order to make them fit to the look and feel of your webshop. Please refer to the User Guide Virtual Terminal to learn about how to make such customizations.

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 optimized 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, PayPal, iDEAL, SEPA Direct Debit, SOFORT Banking, giropay, MasterPass and Klarna as well as payment methods covered by the First Data Local Payments product option.
- 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) as well as the SEPA Direct Debit payments (the parameter ‘debitDE’).

It works with HTML 5 enabled browsers like e.g.: Microsoft Internet Explorer 9 onwards, Google Chrome and Mozilla Firefox, while the minimum size of the iFrame is: 900 px (height) and 460 px (width).

The JSP example below demonstrates 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>
```

In case of Javascript, you must register a method e.g. 'receiveMessage' with the listener like window.addEventListener("message", receiveMessage, false). You can see the sample implementation for the method 'receiveMessage' shown below:

```
function receiveMessage(event){
       if (event.origin != "https://test.ipg-online.com")// Need to replace
       this URI with production one return;
        var connectForm = event.source.document.forms[0];
        var newForm = document.createElement("form");
        newForm.setAttribute('method',"post");
        newForm.setAttribute('action',event.data);
        newForm.setAttribute('id',"newForm");
        newForm.setAttribute('name',"newForm");
        document.body.appendChild(newForm);
```

```
            for(j=0 ; j<connectForm.elements.length; j++){
                    var element = connectForm.elements[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();
    }
```

The parameter 'hostURI' 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.

The parameter 'hexColorCode' allows you to easily align the color of the buttons in an iFrame to the look and feel of your website.

3. Getting Started

This section 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. This section assumes that the developer has a basic understanding of his chosen scripting language.

3.1 Checklist

In order to integrate with the payment 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 payment 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/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="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="978"/>
        <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, the included file, ipg-util.asp uses a server side JavaScript file to build the SHA-256 hash. This file can be provided on request. 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 payment 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","978" )
?>"/>
  <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="978"/>

  <input type="submit" value="Submit">
  </form>
  </body>
</html>
```

Note that 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 EUR or 13.99 EUR) 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 EUR 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 payment gateway (X = mandatory field). Please refer to this Integration Guide's Appendixes for implementation details in relation to alternative payment methods and the First Data Fraud Detect product.

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) 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 (sale)	X (preauth)	X (postauth)	X (void)	X (payer_auth)
timezone	Timezone of the transaction in Area/Location format, e.g. Africa/Johannesburg America/New_York America/Sao_Paulo Asia/Calcutta Australia/Sydney Europe/Amsterdam Europe/Berlin Europe/Dublin Europe/London Europe/Rome	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
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'	X	X			

	(the chosen mode for the transaction when using the 'classic' checkout option)					
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 below)	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, PayPal and Klarna.
** 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.

Please see a list of currencies and their ISO codes in Appendix IV.

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 pages, • '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 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. Please note that for: • Direct Debit transactions, the Customer ID can be submitted to the bank, depending on the length of the Order ID. The maximum amount of characters that can be submitted to the bank is 78. • iDEAL transactions, the Customer ID can be submitted in your request filled in with any relevant data which can be populated in a field in the iDEAL TransactionRequest to be displayed on your consumers' bank account statements.
dccInquiryId	Inquiry ID for a Dynamic Pricing request. Used to send the Inquiry ID you have obtained via a Web Service API call (RequestMerchantRateForDynamicPricing). This value will be used to retrieve the currency conversion information (exchange rate, converted amount) for this transaction.
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

<table>
<tr><td></td><td>this field, please contact your local support team to verify if this feature is supported in your country.</td></tr>
<tr><td>idealIssuerID</td><td>This parameter can be used to submit the iDEAL issuing bank in case you let your customers select the issuer within your shop environment. If you do not pass this value for an iDEAL transaction, a hosted selection form will be displayed to your customer.
<table>
<tr><th>iDEAL issuer</th><th>Value</th></tr>
<tr><td>ABN AMRO</td><td>ABNANL2A</td></tr>
<tr><td>ING</td><td>INGBNL2A</td></tr>
<tr><td>SNS Bank</td><td>SNSBNL2A</td></tr>
<tr><td>van Lanschot</td><td>FVLBNL22</td></tr>
<tr><td>Triodos Bank</td><td>TRIONL2U</td></tr>
<tr><td>Knab</td><td>KNABNL2H</td></tr>
<tr><td>Rabobank</td><td>RABONL2U</td></tr>
<tr><td>RegioBank</td><td>RBRBNL21</td></tr>
<tr><td>ASN Bank</td><td>ASNBNL21</td></tr>
<tr><td>Bunq</td><td>BUNQNL2A</td></tr>
</table></td></tr>
<tr><td>invoicenumber</td><td>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.</td></tr>
<tr><td>hashExtended</td><td>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.</td></tr>
<tr><td>hexColorCode</td><td>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 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.</td></tr>
<tr><td>hostURI</td><td>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.</td></tr>
<tr><td>item1 up to item999</td><td>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 items. 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</td></tr>
<tr><td>language</td><td>This parameter can be used to override the default payment page language configured for your merchant store.
The following values are currently possible:
<table>
<tr><th>Language</th><th>Value</th></tr>
<tr><td>Chinese (simplified)</td><td>zh_CN</td></tr>
<tr><td>Chinese (traditional)</td><td>zh_TW</td></tr>
</table></td></tr>
</table>

	Czech	cs_CZ
	Dutch	nl_NL
	English (USA)	en_US
	English (UK)	en_GB
	Finnish	fi_FI
	French	fr_FR
	German	de_DE
	Greek	el_GR
	Italian	it_IT
	Polish	pl_PL
	Portuguese (Brazil)	pt_BR
	Serbian	sr_RS
	Slovak	sk_SK
	Spanish	es_ES
mandateDate	This field allows you to reference to the date of the original mandate when performing recurring Direct Debit transactions. The date needs to be submitted in format YYYYMMDD. **Please note that this is a mandatory field for recurring Direct Debit transactions.**	
mandateReference	This field allows you to transmit a Mandate Reference for Direct Debit payments. Please note the regulatory requisite to keep the Mandate Reference unambiguous.	
mandateType	This field allows you to process Direct Debit transactions that are based on mandates for recurring collections. The mandate type can be set to 'single' for single (one-off) debit collections, to 'firstCollection' when submitting the initial transaction related to a mandate for recurring Direct Debit collections, to 'recurringCollection' for subsequent recurring transactions or to 'finalCollection' for the last direct debit in a series of recurring direct debits. Transactions where this parameter is not submitted by the merchant will be flagged as a single debit collection. **Please note that it is mandatory to submit a mandateReference in case of recurring collections.**	
mandateUrl	When your store is enabled for SEPA Direct Debit as part of the Local Payments offering, this field allows you to transmit a valid URL of SEPA Direct Debit mandate to enable the Risk and Compliance department to access the details. **Please note that it is mandatory to submit a mandateReference and a mandateDate together with a mandateUrl in case you manage SEPA Direct Debit mandates on your side in the combination with the Local Payments offering.**	
merchantTransactionId	Allows you to assign a unique ID for the transaction. This ID can be used to reference to this transactions in a PostAuth or Void request (referencedMerchantTransactionId).	
mobileMode	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.	
numberOfInstallments	This parameter allows you to set the number of instalments for a Sale transaction if your customer pays the amount in several parts.	
installmentsInterest	This parameter allows you to choose, if instalment interest should be applied or not, the values "true" or "false" are currently possible.	
installmentDelayMonths	This parameter allows you to delay the first instalment payment for several months, values 2-99 are currently possible.	
oid	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. Please note that for Direct Debit transactions, a maximum of 78 characters can be submitted to the bank.	
paymentMethod	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 payment gateway will display a drop-down menu to the customer to choose from the payment methods available for your shop.	

	For valid payment method values please refer to Appendix V.
ponumber	This field allows you to submit a Purchase Order Number with up to 50 characters.
refer	This field describes who referred the customer to your store.
referencedMerchantTransactionID	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.
responseFailURL	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.
responseSuccessURL	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.
reviewOrder	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.
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 transaction after having clicked on "Finish shopping" within the Wallet.
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).
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)**	**SEPA Direct Debit**	**Maestro**	**PayPal, SOFORT, giropay, IDEAL, Klarna, Netbanking, MasterPass**	**Bancontact**

cardnumber	Your customer's card number. 12-24 digits.	X		X		X
expmonth	The expiry month of the card (2 digits)	X		X		X
expyear	The expiry year of the card (4 digits)	X		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)		X			
bic	Your customer's BIC – Business Identifier Code (8 or 11 digits)		(X) mandatory if foreign IBAN			
bname	Name of the bank account owner that will be debited		X			

6.2 payplus Mode

Using payplus mode, it is possible to additionally transfer billing information to the payment 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 payment 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 payment 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)
- An IBAN must contain 22 digits
- A BIC needs to contain 8 or 11 digits

If the submitted data should not be valid, the payment 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 (RequestMerchantRateForDynamicPricing).

Note, if you implement the payment method Klarna in (Full-)ByPass mode, you will need to follow certain rules for allowing item handling in your request. For more information please refer to Appendix VII.

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 payment 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,American Express SafeKey and JCB J/Secure. 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 payment 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","978" ) %>"/>
                <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="978"/>
                <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=978,
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 Mandate in 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.

If you are a UK 6012 merchant use the following parameters for your transaction request:

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

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 or IBAN and BIC 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. Solvency Information from Bürgel

The Connect solution is integrated with Bürgel Wirtschaftsinformationen, a leading company in the field of business information.

This integration allows you to select the payment methods you offer to an individual customer based on Bürgel's information on the non-payment risk. Please see information on setting options in the User Guide Virtual Terminal..

If you have a contract with Bürgel and have ordered this product option, use the following parameters for your transaction requests:

Field Name	Description	Mandatory
valueaddedservices	Buergel	Please submit this parameter for all transactions where you want to use this feature
bfirstname, blastname, bname	Customer name	Yes, bfirstname and blastname or bname
baddr1	Customer address	Yes, format must be street and house number
bzip	Customer ZIP or Postal Code	Yes
bcity	Customer city	Yes
bcountry	Customer country	Yes, in the ISO alpha code format, e.g. DE
bbirthday	Customer birthday	Not mandatory. Format: DD.MM.YYYY

If any of the mandatory address information is missing, the transaction request will be declined.

12. 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 installments 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 installments
recurringComments	Limit of 100 characters, including spaces	Any comments about the recurring transaction

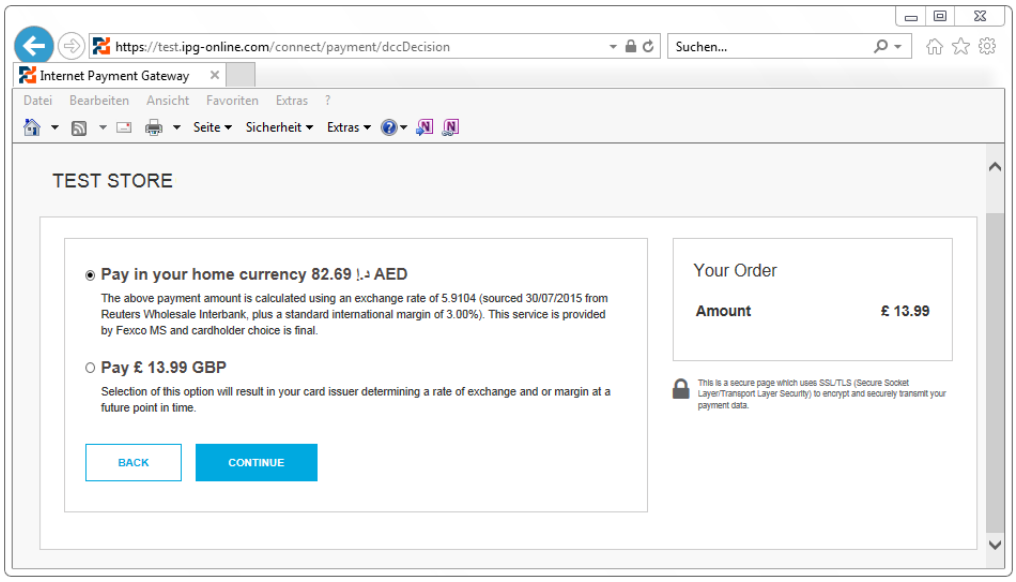
Note that the start date of the recurring payments will be the current date and will be automatically calculated by the system.

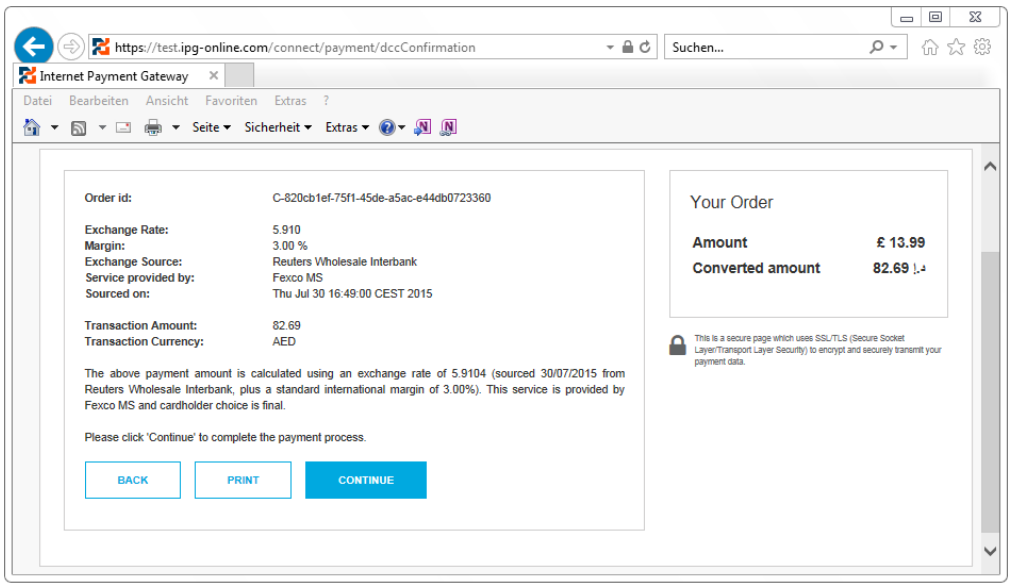
The recurring payments installed using Connect can be modified or cancelled using the Virtual Terminal or Web Service API.

13. Global Choice™ and Dynamic Pricing

With First Data's Global Choice™, 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.





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.

14. Purchasing Cards

Purchasing Cards offer businesses the ability to allow their employees to purchase items with a credit card while providing additional information on sales tax, customer code etc. When providing specific details on the payment being made with a Purchasing card favourable addendum interchange rates are applied.

There are three levels of details required for Purchasing Cards:

- Level I — The first level is the standard transaction data; no enhanced data is required at this level.
- Level II — The second level requires that data such as tax amount and customer code be supplied in addition to the standard transaction date. (Visa only have a level II option)
- Level III — The third level allows a merchant to pass a detailed accounting of goods and services purchased to the buyer. All the data for Level I and Level II must also be passed to participate in Level III. (Visa and MasterCard).

You can submit Level II and Level III data in your transaction request using the following parameters:

Field Name	Description
pcCustomerReferenceID	Merchant-defined reference for the customer that will appear on the customer's statement.
pcSupplierInvoiceNumber	Merchant-defined reference for the invoice, e.g. invoice number.
pcSupplierVATRegistrationNumber	The Identification number assigned by the taxing authorities to the merchant.
pcTotalDiscountAmount	The total discount amount applied to a transaction (i.e. total transaction percentage discounts, fixed transaction amount reductions or summarisation of line item discounts).
pcTotalDiscountRate	The rate of the discount for the whole transaction.
pcVatShippingRate	The total freight/shipping amount applied to the transaction. Merchants can choose to deliver the contents of a single transaction in multiple shipments and this field reflects the total cost of those deliveries.
pcVatShippingAmount	The total freight/shipping amount applied to the transaction. Merchants can choose to deliver the contents of a single transaction in multiple shipments and this field reflects the total cost of those deliveries.
pcLineItemsJson	Line Item Details in JSON format. See table below for more information.

Purchasing Cards Line Item Details in JSON format:

Field Name	Description
CommodityCode	A reference to a commodity code used to classify purchased item.
ProductCode	A reference to a merchant product identifier, the Universal Product Code (UPC) of purchased item.
Description	Represents a description of purchased item.
Quantity	Represents a quantity of purchased items.

UnitOfMeasure	Represents a unit of measure of purchased items.
UnitPrice	Represents mandatory data for Level III transactions.
VATAmountAndRate	Represents a rate of the VAT amount, e.g. 0.09 (means 9%).
DiscountAmountAndRate	Represents a rate of the discount amount, e.g. 0.09 (means 9%).
LineItemTotal	This field is a calculation of the unit cost multiplied by the quantity and less the discount per line item. The calculation is reflected as: [Unit Cost * Quantity] - Discount per Line Item = Line Item Total.

15. Transaction Response

15.1 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),'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, the backend for giropay transactions for example returns the response code '4000' for succesful transactions.
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 DINERSCLUB JCB CHINA_UNION_PAY CABAL MAESTRO RUPAY BCMC SOROCRED Filled with "N/A" for any payment method which is not a credit card or debit card

For 3D Secure transactions only:

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 Global Choice™ transactions only:

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 iDEAL transactions only:

accountOwnerName	Name of the owner of the bank account that has been used for the iDEAL transaction
iban	IBAN of the bank account that has been used for the iDEAL transaction
bic	BIC of the bank account that has been used for the iDEAL transaction

For Klarna transactions only:

klarnaFirstname	First name of the customer that has been used for the Klarna transaction
klarnaLastname	Last name of the customer that has been used for the Klarna transaction
klarnaStreetName	Name of the street that has been used for the Klarna transaction
klarnaHouseNumber	Number of the house that has been used for the Klarna transaction
klarnaHouseNumberExtension	Extension of the house number that has been used for the Klarna transaction
klarnaCellPhoneNumber	Number of the cell phone that has been used for the Klarna transaction
klarnaPClassID	The pclasses ID. For Invoice: -1. For Part Pay please refer to the table Klarna cost calculation values (pclasses) on the *Klarna Setup* page in the Virtual Terminal's Customisation section

For MasterPass transactions only:

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

For Fraud Detect transactions only:

fraudScore	Score returned based on Fraud Detect check

When your store is enabled for SEPA Direct Debit as part of the Local Payments offering:

mandateReference	Mandate reference as returned for the first direct debit transaction
mandateDate	Date of the initial direct debit transaction as returned for the first 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 if you want to use this feature, you have to store the 'txndatetime' that you have submitted with the transaction request in order to be able to validate the response hash.

15.2 Server-to-Server Notification

In addition to the response you receive in hidden fields to your 'responseSuccessURL' or 'responseFailURL', the payment 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 a SHA-256 Hash or Extended Hash

Example of Hash

- 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.

393837363534333231303132303133 3a30373a31362d30393a35373a3038312e30303832365 46f70536563726574

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

SHA256(393837363534333231303132303133 3a30373a31362d30393a35373a3038312e303 0383236546f70536563726574)

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="hash"
value="3d7e75aa0b4e0e1d4a7ac87e451e64692cced46f4358ef35a69d967213412
43c
"/>
```

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.

393837363534333231303132303133 3a30373a31362d30393a35373a3038312e303038323 6546f70536563726574

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

SHA256(393837363534333231303132303133 3a30373a31362d30393a35373a3038312e303 0383236546f70536563726574)

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
<?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
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	ILS	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 – Payment Method List

If you let your consumer select the payment method in your website or want to define the payment method yourself, submit the parameter 'paymentMethod' in your transaction request. If you do not submit this parameter, the gateway will display a hosted page to the consumer to choose from the payment methods that are enabled for your store and supported for the combination of the consumer's country and the transaction currency.

For the Local Payments method specific (mandatory/optional) fields please refer to Appendix X.

Payment Method	**Value**
Alipay	aliPay
American Express	A
AstroPay Card	astropayCard
AstroPay Direct (dLocal)	astropayDirect
Bancontact	BCMC
Boleto Bancário	boleto
Cabal	CA
Diners	C
Entercash	enterCash
Equated Monthly Installments (EMI)	emi
Finnish Online Banking Transfer (Verkkopankki)	finlandOnlineBanking
giropay	giropay
iDEAL	ideal
JCB	J
Klarna	klarna
Local Wallets India	indiawallet
Maestro	MA
Maestro UK	maestroUK
MasterCard	M
MasterPass	masterpass
MyBank	mybank
Netbanking (India)	netbanking
PayPal	paypal
Paysafecard	paySafeCard
QIWI	qiwi
RuPay	RU
SafetyPay	safetypay
Santander (Online Banking)	santander
Santander Cash (Cash Payment)	santanderCash
SEPA Direct Debit	debitDE
SEPA Direct Debit (as part of the Local Payments offering)	direct_debit-apm
SOFORT Banking (SOFORT Überweisung)	sofort
Sorocred	SO
TrustPay	trustPay
Visa (Credit/Debit/Electron/Delta)	V

Appendix VI – 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 payment 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.

Appendix VII – Klarna Payment Methods

Refer to the following information when integrating Klarna Invoice and Part Pay as payment methods.

Prepare your website

You can choose to integrate your website in such a way that your customers will be redirected to hosted payment forms for the Klarna checkout process. Using this integration method, the Connect solution is providing all required input forms for you.

For Klarna, all of Connect's payment modes (payonly, payplus, fullpay) behave in the same way. Since Klarna does not allow the shipping address to be different from the billing address, no separate entry form for a shipping address will be displayed to your customer. Instead, Klarna's specific billing form (customer details form) will be shown to your customers in order to capture their details.

If you prefer your customers never to leave your website, you can create your own specific payment forms for Klarna by modifying your website accordingly to the guidelines presented on Klarna's website (http://developers.klarna.com/en). In that case you will be able to collect all data required for the transaction on your side and send it over to the payment gateway as a part of the transaction request. You could also decide to send a subset of the data needed for the transaction. In that case the Connect solution will only display selected hosted forms to your customers in order to collect the mandatory data that has not been submitted by you.

Activate Klarna for your test store

- Obtain test credentials from Klarna via https://developers.klarna.com/en/se+php/kpm/apply-for-test-account
- Make sure your payment gateway test Store ID has been enabled for Klarna
- Activate Klarna as a payment method in your test store, via the *Klarna Setup* page in the Virtual Terminal's Customisation section.

Order Process with Klarna Payment Methods

The process begins with the customer selecting the goods in your web shop and placing the order. To allow your customers to pay by Klarna you have to submit a PreAuth transaction, which is used to create the order. If it is more suitable for your processes, you can alternatively use the Sale transaction type which will then be automatically translated into a PreAuth transaction. Klarna PreAuth transaction requires a number of mandatory and additional parameters which are described in more detail below.

When the order is submitted, Klarna will run fraud and credit checks on the consumer and tell in return if the purchase is approved. Once the purchase is approved, you should start preparing the goods for shipment.

When the goods are ready for shipment, a Completion needs to be submitted by you to the gateway since it activates the order on Klarna side and allows you to provide the customer with the invoice.

Required parameters for Klarna PreAuth transactions

Basket information (Line items) - Klarna requires the list of items in order to approve the purchase. Therefore it is mandatory to send line items with all PreAuth transactions. The basket information has to be sent in request parameters: 'item1', 'item2' up to 'item999', in the following format:
id;description;quantity;item_total_price;sub_total;vat_tax;shipping.
Transactions without line items will be declined.

Customer information details - Depending on the country selection (please refer to http://developers.klarna.com/en), Klarna requires different consumer information details in order to approve the purchase such as e.g.:

klarnaPersonalNumber	Required for Denmark, Finland, Norway and Sweden
klarnaBirthDate	Required for Austria, Netherlands and Germany. Possible values: yyyy-MM-dd.
klarnaClientGender	Required for Austria, Netherlands and Germany. Possible values: MALE, FEMALE.
klarnaFirstname	Required for all countries. Possible values: alphanumeric characters, spaces and dashes. Maximum length: 48 characters.
klarnaLastname	Required for all countries. Possible values: alphanumeric characters, spaces and dashes. Maximum length: 48 characters.
klarnaStreetName	Required for all countries. Possible values: alphanumeric characters, spaces and dashes. Maximum length: 84 characters.
klarnaHouseNumber	Possible values: alphanumeric characters, spaces and dashes. Maximum length: 6 characters.
klarnaHouseNumberExtension	Possible values: alphanumeric characters, spaces and dashes. Maximum length: 6 characters.
klarnaCellPhoneNumber	Possible values: alphanumeric characters, spaces and dashes. Maximum length: 32 characters
klarnaPClassID	Possible values: whole numbers. The pclasses ID for Invoice: -1. For Part Pay please refer to the table Klarna cost calculation values (pclasses) on the Klarna Setup page in the Virtual Terminal's Customisation section.
klarnaCity	Required for all countries. Possible values: alphanumeric characters, spaces and dashes. Maximum length: 96 characters
klarnaCountry	ISO 3166-1 Alpha 3 format (e.g. DEU, SWE).
klarnaZip	Required for all countries. Possible values: alphanumeric characters, spaces and dashes. Maximum length: 24 characters
klarnaPhone	Possible values: alphanumeric characters, spaces and dashes. Maximum length: 32 characters (same as API).
klarnaEmail	Required for all countries. Possible values: alphanumeric characters, spaces and dashes. Maximum length: 64 characters

Note that Klarna requires that at least one phone will be filled in. The payment gateway does not validate nor explicitly support any special characters like ä,ö,ü ect. or the format of the personal number, it simply passes the data through to Klarna. For more information please refer to http://developers.klarna.com/en.

Taking into account the country selection as well as the chosen integration mode, you can send all customer information details in your transaction request or only a subset of it.

If you send billing information (general parameters starting with "b" e.g.: bname, bcity, bcountry, etc.), the Klarna customer details form, displayed to the customer, will already be prefilled with the details based on the following fields:

- bcity -> klarnaCity
- bcountry -> klarnaCountry
- bzip -> klarnaZip
- phone -> klarnaPhone
- email -> klarnaEmail

Currency – The currency of a PreAuth transaction for Klarna has to correspond to the currency of the customer's country (the buyer's country). You should submit currency in the parameter 'currency' as a numeric ISO code. See examples below.

Country name	Currency name	Currency code	Currency number
Austria Germany Netherlands Norway	Euro	EUR	978
Denmark	Danish Krone	DKK	208
Norway	Norwegian Krone	NOK	578
Sweden	Swedish Krona	SEK	752

Additional data - As part of the order, you may provide additional data such as:

- Shipping fee. This data is not mandatory for Klarna. There are several options to specify it:

  1. The preferred option is to include the shipping fee to the parameter 'shipping', in the same format as 'chargetotal'. 'subtotal' and 'vaxtax' have to be submitted as well in this case. Note that the 'chargetotal' has to be equal to 'subtotal' plus 'shipping' plus 'vattax'. At the same time, the sum of the 'item_total_price' has to be equal to 'chargetotal' without 'shipping'. Do not forget to regard the 'quantity' when calculating the values. See examples below:

     chargetotal=89.00
     subtotal=58.00
     shipping=10.00
     vattax=21.00
     A;Product A;1;5.0;3.0;2.0;0
     B;Product B;5;10.0;7.0;3.0;0
     C;Product C;2;12.0;10.0;2.0;0

  2. The second option is to include the shipping fee as a separate line item with a specific id: 'IPG_SHIPPING'. Note that the sum of the 'item_total_price' has to be equal to the 'chargetotal'. Do not forget to regard the 'quantity' when calculating the values. See examples below:

     chargetotal=85.00
     subtotal=63.00
     shipping=0.0
     vattax=22.00
     A;Product A;1;5.0;3.0;2.0;0
     B;Product B;5;10.0;7.0;3.0;0
     C;Product C;2;12.0;10.0;2.0;0
     IPG_SHIPPING;Shipping costs;1;6.0;5.0;1.0;0

- Discounts. This data is not mandatory for Klarna.
  You can define the discount by submitting the item with a negative amount. See examples below:

  chargetotal=84.00
  subtotal=62.10
  shipping=0.0
  vattax=21.90
  A;Product A;1;5.0;3.0;2.0;0
  B;Product B;5;10.0;7.0;3.0;0
  C;Product C;2;12.0;10.0;2.0;0
  D;Basic clipboard;1;-1;-0.9;-0.1;0
  IPG_SHIPPING;Shipping costs;1;6.0;5.0;1.0;0

- Handling fee (Charge fee=Invoice fee).
  This data is required only for Klarna Invoice and is set to '0' by default it. You can however decide to set the invoice fee to a different extend, via the *Klarna Setup* page in the Virtual Terminal's Customisation section.

  Note that when a customer will select Invoice as a payment method in the normal Connect flow, the handling fee is automatically added to the order as an additional item with the specific id: 'IPG_HANDLING'.

  In case you use (Full)ByPass mode you have to follow certain rules to allow the handling item in your request by submitting:

  - an item with id: 'IPG_HANDLING'

Example: IPG_HANDLING;Transaction fee;1;6.0;6.0;0;0

- klarnaPClassID = -1
- klarnaCountry filled in

All other transactions containing 'IPG_HANDLING' item will be declined.

Transaction types mapping

Connect Transaction Type (txntype)	**Klarna operation**
sale	Not applicable / will be translated into preauth
preauth	reserveAmount
postauth *(full and partial)*	Activate
void	Cancel_Reservation Credit_Invoice

The basket information is required for all PreAuth transactions. PreAuth transactions without items will be declined. The full PostAuth transaction can be submitted without items but the order amount has to be the same as the remaining value of the original transaction.
Void can be used for PreAuth and PostAuth transactions with the restriction that PostAuth transactions can only be voided during the same day.

Note that a PreAuth transaction for Klarna could return the transaction result: WAITING. This status indicates that the transaction is being reviewed by Klarna and will be updated at a later point.

Appendix VIII – 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 payment 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 finalize 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 payment gateway test Store ID has been enabled for MasterPass.

Appendix IX – Fraud Detect

Refer to the following information when you are signed up to First Data's Fraud Detect product to have card transactions reviewed for a fraud scoring.

You can submit a payment transaction to the gateway, which routes it to the appropriate authorization front-end. The gateway receives the authorization response. If an approval is received, the gateway submits the transaction to Fraud Detect including authorization response details (e.g. AVS/Card Code match). Fraud Detect returns a fraud score (between 1-1000) to the gateway and depending on how the risk tolerance level is set for your store (default 500) the transaction is either approved or voided and declined. Refer to the Virtual Terminal Guide to learn more about the way how to set the risk tolerance level for your store.

In case you use the Fraud Detect product and want to pass mobile device details for the scoring, you need to pass these with the following parameter naming:

- customParam_deviceRiskId
- customParam_deviceRiskAPIKey
- customParam_deviceRiskHost

Example:

```
<input type="hidden" name="customParam_deviceRiskId" value="*****"/>
```

These fields are handled in the same way as other optional request parameters. The gateway stores these parameters and passes them on to Fraud Detect. These parameters have no impact on the transaction processing flow.

Appendix X – Local Payments™

Refer to the following information when your store is enabled for the Local Payments offering.

The Local Payments solution offers a unique combination of global coverage, a single contracting and integration experience, and a broad and expanding portfolio of local payment methods.

Local Payments, also often referred to as Alternative Payment Methods, are defined as payment transactions where neither credit/debit cards or paper currencies are used as the form of payment. These payment methods are primarily used in eCommerce and mCommerce transactions, although some solutions are making a push for adoption at point of sale locations. In many markets, they are more commonly used than credit/debit cards.

Local Payments differ from card/association processing in a number ways. They are generally designed to meet local needs and used in one or a limited number of markets. Unlike traditional credit/debit card processing, pricing across these payment methods is not uniform and retail pricing depend on local costs and merchant industries (e.g. high-risk vs. low-risk). Local Payments offerings and user experiences also vary greatly, though most are quite different from debit/credit user experiences.

There are over 300 local payment methods in use across the globe. Most of these fall into various categories: Online Banking, Direct Debits, Direct Carrier Billing, Cash on Delivery, Invoice/Installments, eWallets /mWallets, Cash/Voucher Payments and Payouts.

Consumer demand and preference are driving the growth in new methods of payment across the globe. In fact, local payment methods are growing more rapidly than major card schemes, and merchant demand for non-card (credit/debit) methods of payment is on the rise. These new payment methods deliver many benefits to both merchants and consumers.

Local Payments help you reach and securely process payments from a broader base of consumers in each local market, reduce shopping cart abandonment/improve conversion and improve customer experiences. They enable more consumers to easily and confidently shop online (i.e. provide easy access to secure payment methods for those that are unbanked and/or without credit or debit cards), expand their ability to access international merchants and enable them to 'pay their way,' all of which improve their shopping experiences and overall satisfaction.

Initiating a Sale transaction

A Sale transaction for most Local Payments requires a direct interaction with the consumer who needs to be redirected to the payment method's screens (e.g. the login page of the consumer's bank or a wallet provider) and back to your website after all required steps are completed.

As we handle all the required redirections to the various stakeholders for you, all you need to do is to post a form to a URL with the parameters and values required for the transaction.

URL for Test Transactions

```
https://test.ipg-online.com/connect/gateway/processing
```

You will get the production URL with your production account credentials.

When building a Request, independently of the payment method, there are some mandatory fields that need to be included in every request for a Sale transaction.
Example of a form with the minimum number of fields:

```
<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="America/New_York"/>
<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","840" )
%>"/>
<input type="hidden" name="storename" value="541234567" />
<input type="text" name="chargetotal" value="13.00" />
<input type="hidden" name="currency" value="978"/>
<input type="submit" value="Submit">
</form>
```

Other generic fields to be considered:

Field Name	Comment
paymentMethod	You can submit the parameter 'paymentMethod' in your transaction request relevant for a selected local payment method, as defined in Appendix V. If you do not submit this parameter, gateway will display a page to your consumer to choose from the payment methods that are supported for the combination of the consumer's country and the transaction currency.
bname	The consumer's name, e.g. Albert Einstein. This is required for all Local Payments transactions. If you do not submit this field, a hosted page will be displayed to the consumer to capture the name.
bcountry	The consumer's country in 2 Letter Country Code format, e.g. US for the United States or DE for Germany. The country is required for many Local Payments methods so we recommend to include it in every Sale transaction request. If you do not submit this field and the payment method requires it, a hosted page will be displayed with the country that we have identified based on IP address and the option to change the country if not appropriate.

The payment method specific fields to be considered:
(M)=Mandatory (O)=Optional

Field Name	Relevant for	Comment
nationalId	Boleto Bancário (M) Santander (M) Santander Cash (M)	Consumer's National ID (up to 30 characters)
customerid	AstroPay Card (M) AstroPay Direct (M) Boleto Bancário (M) Entercash (M) Santander (M) Santander Cash (M)	Unique reference to identify the consumer
email	Boleto Bancário (M) SEPA Direct Debit (M) Santander (M) Santander Cash (M)	Consumer's email address
bbirthday	Boleto Bancário (O) Santander (O) Santander Cash (O)	Consumer's birthdate, format: DD.MM.YYYY
bic	Entercash (O) giropay (O) SOFORT Banking (O)	Consumer's BIC – Business Identifier Code (8 or 11 digits)
iban	Entercash (O) SEPA Direct Debit (M)	Consumer's IBAN - International Bank Account Number (22 digits)
mandateDate	SEPA Direct Debit (M)	
mandateReference	SEPA Direct Debit (M)	
mandateType	SEPA Direct Debit (M)	
mandateUrl	SEPA Direct Debit (M)	
phone	QIWI (M)	Consumer's phone number

Initiating a Return transaction

When Return is supported for a selected local payment, you can initiate a Return transaction with a reference to the Transaction ID of the original Sale transaction to the API Web Service. Please see details in the Integration Guide for the Web Service API, chapter Generic Transaction Type for Voids and Returns.

There is the limit for the amount of Return transaction to a maximum of 100 000 either EUR or USD, which are the only currencies that are applicable for these limit for the moment. Returns using other currencies will not be limited.

Options for SEPA Direct Debit

When you manage SEPA Direct Debit mandates on your side you can use these in combination with the Local Payments offering by submitting the reference and date of the mandate as well as a link to the mandate itself. This is especially useful in cases where you have a large number of mandates on file from previously used solutions and want to continue to use these mandates.

Single payment or recurring payment

Field Name	Description
email	Consumer's email address
iban	Consumer's IBAN - International Bank Account Number (22 digits)
mandateType	Sequence type of Direct Debit, defaults to 'single' Values: single - Direct Debit is executed once firstCollection - First Direct Debit in a series of recurring recurringCollection – Follow-up Direct Debit in a series of recurring finalCollection – Last Direct Debit in a series of recurring
mandateReference	To be populated with the mandate reference
mandateDate	To be populated with the initial mandate signature date
mandateUrl	To be populated with the valid URL of the SEPA mandate to enable the Risk and Compliance department to access the details

When you do not want to manage the SEPA Direct Debit mandates on your side, you can instead use the *out-of-box* solution offered by First Data.

Single payment or First payment in recurring series

Field Name	Description
email	Consumer's email address
iban	Consumer's IBAN - International Bank Account Number (22 digits)
mandateType	Sequence type of Direct Debit, defaults to 'single' Values: single - Direct Debit is executed once firstCollection - First Direct Debit in a series of recurring

Follow-up payments in recurring series

Field Name	Description
email	Consumer's email address
iban	Consumer's IBAN - International Bank Account Number (22 digits)
mandateType	Sequence type of Direct Debit Values: recurringCollection – Follow-up Direct Debit in a series of recurring finalCollection – Last Direct Debit in a series of recurring
mandateReference	To be populated with the mandate reference from the response
mandateDate	To be populated with the initial mandate signature date from the response