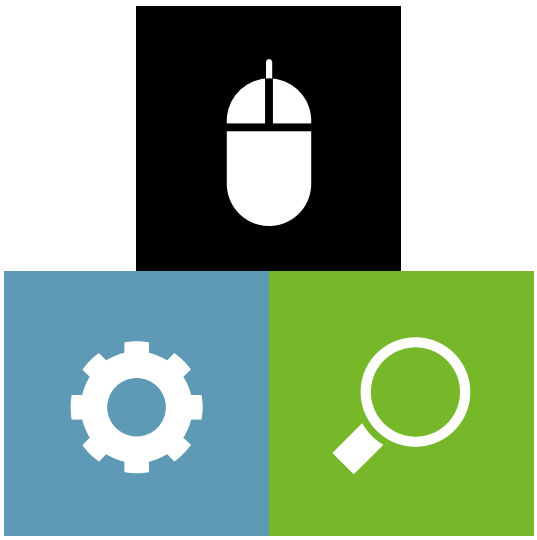
COMMERCIAL BANKING



LLOYDS BANK ONLINE PAYMENTS CONNECT INTEGRATION GUIDE

Version 2023

LLOYDS BANK

Contents

Getting Support

There are different manuals available for Lloyds Bank Online Payments (LBOP) eCommerce solutions. This Integration Guide will be the most helpful for integrating hosted payment forms or a Direct Post.

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

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

1. Introduction

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

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

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

When making decisions on your way of integration, please consider that we do not recommend to use the hosted payment forms inside an iFrame since some Internet browsers do not allow cookies to be sent to the 3rd party hosts, moreover some features (e.g.: 3-D Secure authentications) and some Alternative Payment methods that involve redirections to the 3rd party services (e.g. PayPal) do not allow displaying their screens within an iFrame. However, if you still plan to embed our hosted payment pages inside an iFrame you must use the 'parentUri' parameter to specify an URL of a page, where the hosted payment page will be embedded.

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 several 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.

2.1 Hosted Payment Page

If you want to fully outsource the payment process in order not to have any sensitive cardholder data on your systems, you can use our ready-made hosted pages for your customers to enter their payment information.

The most important aspect around the usage of hosted payment page 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 LBOP.

For a standard hosted payment page integration, you should use the checkout option 'combinedpage' that 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.).

The hosted page is localised in many languages and can be easily customised with your merchant's logo, colors, and font types to make it fit to the look and feel of your shop environment (refer to the User Guide Virtual Terminal to learn more). It also shows your merchant's name (i.e.: legal name) and allows you to display a summary of the purchased items to your customer in the 'Your Order' box.

If you do not want to let your customer select the payment method on our hosted page but want to handle that part upfront within your shop environment, you should submit a value for the parameter 'paymentMethod' in your request to the gateway. In addition, if you do not want to distinguish between different card brands (but just card vs. alternative payment methods), you can send a valid card brand value for the parameter 'paymentMethod' and your customer will see a hosted page for the card details entry with no card brand logo shown. Please contact your local support team if you want to enable this feature. This will be managed with a specific setting performed on your account (service configuration) ('hideCardBrandLogoInCombinedPage').

If you do not submit a value for the parameter 'paymentMethod', the gateway will take your customer to a hosted page to choose from the payment methods activated for your store.

If you do not include in your request the fields like e.g.: the card number or the expiry date for a card payment, the gateway will take your customer to a hosted page to collect this information as being mandatory for a transaction processing.

When e.g.: you plan to integrate a specific local alternative payment method i.e.: Local Wallets India, PayLater by ICICI Bank and RuPay, or you require the gateway to collect a full set of billing and/or shipping information, or your consumers use an old operating system with outdated browser versions, please contact your local support team to discuss an alternative hosted payment page integration while using the legacy checkout option 'classic'.

2.2 Direct Post

In the scenarios where you prefer not to use a hosted payment page, you can submit the required customer data directly from your own form to LBOP, 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).

You create the payment form and display it within your website or app. When your customer has entered the card details and presses the "continue button", the customer's device sends the payment information directly to the gateway.

If you choose the Direct Post option and create your own forms, there are additional fields that must be included in your transaction request to the gateway, which are listed in the chapter on using your own forms to capture the data.

3. Getting Started

This section provides a simple example on how to integrate your website using the "combinedpage" checkout option. 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 Lloyds Bank Cardnet.
For example: 10123456789

- Shared Secret

This is the shared secret provided to you by Lloyds Bank Cardnet.
This is used when constructing the hash value (see more below).

3.2 ASP Example

The following ASP example demonstrates a simple page that will communicate with the payment gateway.

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

```
<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="HMACSHA256"/>
   <input type="hidden" name="hashExtended" value="<% call createExtendedHash("13.00","978") %>"/>
   <input type="hidden" name="storename" value="10123456789" />
   <input type="hidden" name="checkoutoption" value="combinedpage"/>
   <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 hash as is required by LBOP. 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 Lloyds Bank Cardnet 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 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.

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

```
<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="HMACSHA256"/>
<input type="hidden" name="hashExtended" value="<?php echo createExtendedHash("13.00","978") ?>"/>
<input type="hidden" name="storename" value="10123456789"/>
```

```
<input type="hidden" name="checkoutoption" value="combinedpage"/>
<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 Lloyds Bank Cardnet 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 hash as is required by LBOP. 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 other product options.

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)

Field Name	Description, possible values and format	Sale transaction	PreAuth*	PostAuth*	Void	PayerAuth**
timezone	Time zone 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: HMACSHA256 HMACSHA384 HMACSHA512 Only one algorithm value should be used.	X	X	X	X	X
hashExtended	The extended hash needs to be calculated using all non-empty gateway specified request parameters in ascending order of the parameter names, where the upper-case characters come before the lower case (based on ASCII value) and the shared secret must be used as the secret key for calculating the hash value. When you are using Direct Post, there is also an option where you do not need to know the card details (PAN, CVV and Expiry Date) for the hash calculation. This will be managed with a specific setting performed on your store. Please contact your local support team if you want to enable this feature. An example of how to generate a hash is given in Appendix I.	X	X	X	X	X
storename	This is the ID of the store provided by LBOP.	X	X	X	X	X
chargetotal	This is the total amount of the transaction using a dot or comma as decimal separator, e.g. 12.34 for an amount of 12 Euro and 34 Cent. Group separators like1,000.01 / 1.000,01 are not allowed.	X	X	X	X	X
checkoutoption	Set the value for this parameter to 'combinedpage' for a standard hosted payment page integration.	X	X			X
currency	The numeric ISO code of the transaction currency, e.g. 978 for Euro (see examples in Appendix IV)	X	X	X	X	X
oid	The order ID of the initial action a PostAuth shall be initiated for.			X	X	

Field Name	Description, possible values and format	Sale transaction	PreAuth*	PostAuth*	Void	PayerAuth**
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.

** The transaction type 'payer_auth' is only required if you want to split the 3-D Secure authentication process from the payment transaction (authorisation) process. See more information in the 3-D 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.
comments	Place any comments here about the transaction.
customerid	This field allows you to transmit any value, e.g. your ID for the customer.
dynamicMerchantName	The name of the merchant to be displayed on the cardholder's statement. The length of this field should not exceed 25 characters. If you want to use this field, please contact your local support team to verify if this feature is supported in your country.
hideOrderDetails	Set this parameter to 'true' when you want to hide (remove) the 'Your Order' box from our hosted payment page.
highRiskPurchaseIndicator	This optional parameter needs to be set to 'true', for transactions handling a cryptocurrency and initiated from a MCC 6051 (Quasi Cash—Merchant) store; or for transactions handling high risk securities initiated from the store with MCC 6211 (Securities—Brokers/ Dealers).
invoicenumber	This field allows you to transmit any value, e.g. an invoice number or class of goods. Please note that the maximum length for this parameter is 48 characters.
item1 up to item999	Line items are regular Connect integration key-value parameters (URL-encoded), where: ▪ the name is a combination of the keyword item and a number, where the number indicates the list position e.g.: item1 ▪ the value is represented by a semicolon-separated list of values, where the position indicates the meaning of the list item property e.g.: <1>;<2>;<3>;<4>;<5>;<6>;<7> The 'item1' to 'item999' parameters allow you to send basket information in the following format: `id;description;quantity;item_total_price;sub_total;vat_tax;shipping;local_tax;category;detailed_category`

<table>
<tr><th>Field Name</th><th colspan="2">Description, possible values and format</th></tr>
<tr><td>language</td><td colspan="2">This parameter can be used to override the default payment page language configured for your merchant store.
The following values are currently possible:</td></tr>
<tr><td></td><td>Language</td><td>Value</td></tr>
<tr><td></td><td>Chinese (simplified)</td><td>zh_CN</td></tr>
<tr><td></td><td>Chinese (traditional)</td><td>zh_TW</td></tr>
<tr><td></td><td>Czech</td><td>cs_CZ</td></tr>
<tr><td></td><td>Danish</td><td>da_DK</td></tr>
<tr><td></td><td>Dutch</td><td>nl_NL</td></tr>
<tr><td></td><td>English (USA)</td><td>en_US</td></tr>
<tr><td></td><td>English (UK)</td><td>en_GB</td></tr>
<tr><td></td><td>Finnish</td><td>fi_FI</td></tr>
<tr><td></td><td>French</td><td>fr_FR</td></tr>
<tr><td></td><td>German</td><td>de_DE</td></tr>
<tr><td></td><td>Greek</td><td>el_GR</td></tr>
<tr><td></td><td>Hungarian</td><td>hu_HU</td></tr>
<tr><td></td><td>Italian</td><td>it_IT</td></tr>
<tr><td></td><td>Japanese</td><td>ja_JP</td></tr>
<tr><td></td><td>Norwegian (Bokmål)</td><td>nb_NO</td></tr>
<tr><td></td><td>Polish</td><td>pl_PL</td></tr>
<tr><td></td><td>Portuguese (Brazil)</td><td>pt_BR</td></tr>
<tr><td></td><td>Serbian (Serbia)</td><td>sr_RS</td></tr>
<tr><td></td><td>Slovak</td><td>sk_SK</td></tr>
<tr><td></td><td>Slovenian</td><td>sl_SI</td></tr>
<tr><td></td><td>Spanish (Spain)</td><td>es_ES</td></tr>
<tr><td></td><td>Spanish (Mexico)</td><td>es_MX</td></tr>
<tr><td></td><td>Swedish</td><td>sv_SE</td></tr>
<tr><td>merchantTransactionId</td><td colspan="2">Allows you to assign a unique ID for the transaction. This ID can be used to reference to this transaction in a PostAuth or Void request (referencedMerchantTransactionId).</td></tr>
<tr><td>mobileMode</td><td colspan="2">The legacy checkout option specific parameter: If your customer uses a mobile device for shopping at your online store you can submit this parameter with the value ‘true’, when using the ‘classic’ checkout option. This will lead your customer to a payment page flow that has been specifically designed for mobile devices.</td></tr>
<tr><td>mode</td><td colspan="2">The legacy checkout option specific parameter: If you are building a payment request for the Sale, PreAuth or PayerAuth transaction, when using the ‘classic’ checkout option, your request needs to include a value for one of the three different modes to define the range of data that shall be captured by the gateway:
▪ ‘payonly’ – shows a hosted page to collect the minimum set of information for the transaction (e. g. cardholder name, card number, expiry date and card code for a credit card transaction),
▪ ‘payplus’ – in addition to the above, the 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.</td></tr>
</table>

Field Name	Description, possible values and format
oid	This field allows you to assign a unique ID for your order. If you choose not to assign an order ID, the LBOP system will automatically generate one for you. Please note, that only the following characters are allowed, if you are generating oid yourselves: A-Z, a-z, 0-9, "-"
parentUri	If you plan to embed our hosted payment pages inside an iFrame you must use this parameter, with the maximum length of 2048 characters, to specify an URL of a page, where the hosted payment page will be embedded. However, note that we do not recommend using the hosted payment forms inside an iFrame since some Internet browsers do not allow cookies to be sent to the 3rd party hosts, moreover some features (e.g.: 3-D Secure authentications) and some Alternative Payment methods that involve redirections to the 3rd party services (e.g. PayPal) do not allow displaying their screens within an iFrame.
partialApproval	The partial approval feature is particularly useful when the transaction amount exceeds the available funds on the customer's card. This feature will allow an approval of the available amount to pay for a portion of the transaction, then the remainder can be paid using another payment method. If you are eligible to use this feature, then you can use this parameter to indicate whether to allow partial approval or not. Valid values: ▪ true ▪ false (default)
paymentMethod	If you let the customer select the payment method (e.g. Mastercard, Visa) 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.
referencedSchemeTransactionId	Credentials on file (COF) specific parameter. This field allows you to include in your request 'schemeTransactionId' that has been returned in the response of the initial transaction to provide a reference to the original transaction, which stored the credentials for the first time.
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.
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. Possible values are: ▪ 'MAIL' (for transactions where the payment details are captured manually and provided in written form the Card Code entry is not allowed), ▪ 'PHONE' (for transactions where you have received the order over the phone and enter the payment details yourself the Card Code entry is required), ▪ 'ECI' (for standard usage in an eCommerce environment where your customer enters the payment details).
unscheduledCredentialOnFileType	Credentials on file (COF) specific parameter. This field allows you to flag transactions as unscheduled credential on file type. Currently the valid values are: FIRST, CARDHOLDER_INITIATED or MERCHANT_INITIATED to advise the scenario if the credential is stored on your side.
vattax	This field allows you to submit an amount for Value Added Tax or other taxes, e.g.: GST in Australia. Please ensure the sub total amount plus shipping plus tax equals the charge total.

6. Using your own forms to capture the data

If you decide to create your own forms, i.e.: Direct Post (not to use the ones provided and hosted by LBOP), there are additional mandatory fields that you need to include. These fields are listed in the following sections.

Using Direct Post allows you to have full control over the look and feel of the form where your customers enter their card details for payment while simultaneously avoiding the need to have sensitive card data within your systems.

It is also important that you check if JavaScript is activated in your customer's browser. If necessary, inform your customer that JavaScript needs to be activated for the payment process.

6.1 Capture payment details

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, your form needs to contain the following fields (part of them can be hidden).

For Credit/Debit Card fields

Field Name	Description, possible values and format	Credit Card (+ Visa Debit/ Electron/Delta)	Maestro
cardnumber	Your customer's card number. 12-24 digits.	**X**	**X**
expmonth	The expiry month of the card (2 digits)	**X**	**X**
expyear	The expiry year of the card (4 digits)	**X**	**X**
cvm	The card code, in most cases on the backside of the card (3 to 4 digits)	**X**	**X** as an optional field "if on card"

6.2 Capture billing information

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 limited to 96	Customer's Company
bname	Alphanumeric characters, spaces, and dashes limited to 96	Customer's Name
baddr1	Limit of 96 characters, including spaces	Customer's Billing Address 1

Field Name	Possible Values	Description
baddr2	Limit of 96 characters, including spaces	Customer's 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	Customer's Phone Number
fax	Limit of 32 Characters	Customer's Fax Number
email	Limit of 254 Characters	Customer's Email Address

6.3 Capture shipping information

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 limited to 96	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

Field Name	Possible Values	Description
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
sphnumber	Limit of 32 Characters	Customer's Phone Number
semail	Limit of 254 Characters	Customer's Email Address

6.4 Validity checks

Prior to the authorisation request for a transaction, the payment gateway performs the payment methods' specific validation checks:

For Credit/Debit Card the following checks are performed:

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

If the submitted data should not be valid, the 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.

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.

Field Name	Description, possible values and format
customParam_key	If you want to use this feature, please send the custom fields in the format customParam_ key=value. The maximum length of a custom parameter is 100 characters. Example:<input type="hidden" name="customParam _ color" value="green"/>

To remain compliant the custom fields are not to be used to submit credit card detail or sensitive card holder information, please use the designated fields defined by the Gateway for this purpose.

8. 3-D Secure

The Connect solution includes the ability to authenticate transactions using Verified by Visa, Mastercard SecureCode, American Express SafeKey and Diners ProtectBuy to provide an additional security layer for online card transactions.

If your store is enabled for 3-D Secure, all Sale or preAuth transactions that you initiate by posting an HTML form will by default go through the 3-D Secure process without the need for you to do anything, i.e.: cardholders with an enrolled card will see a page from the card issuer to enter the password unless the card issuer decides not to check it.

The generic fields to be considered:

Field Name	Description, possible values and format

authenticateTransaction	Optional parameter to be set either to 'true' or 'false' to enable or disable 3-D Secure authentication on a Transaction-by-Transaction basis.
	Example for a transaction with 3-D Secure:
	`<input type="hidden" name="authenticateTransaction" value="true"/>`
	Example for a transaction without 3-D Secure:
	`<input type="hidden" name="authenticateTransaction" value="false"/>`

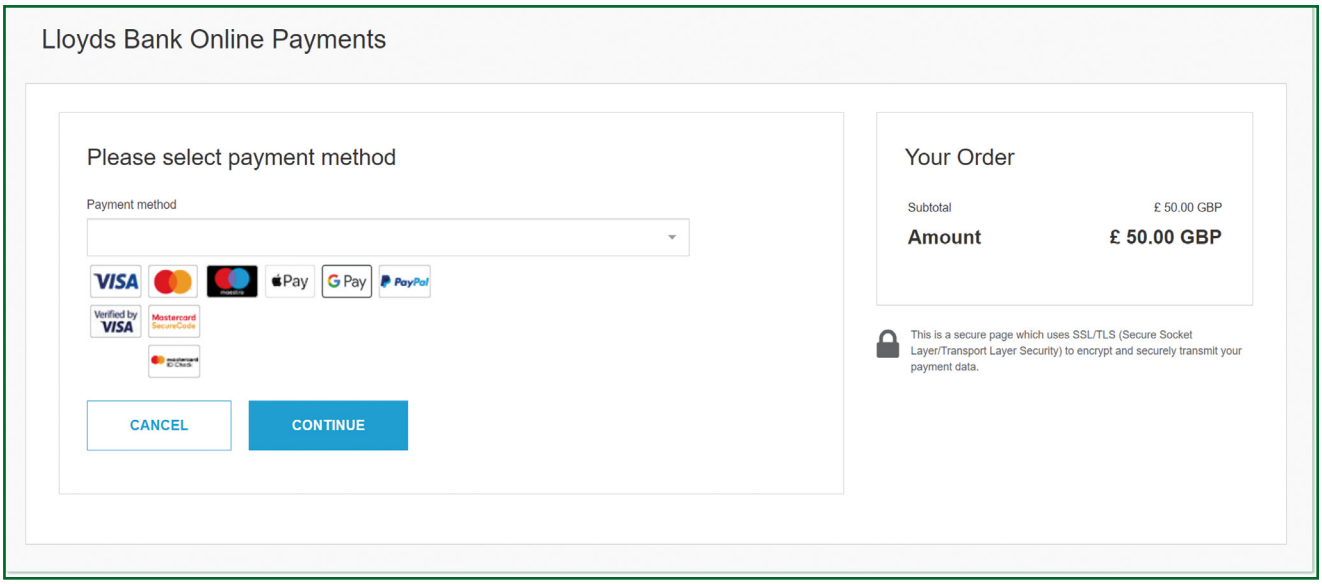
threeDSRequestorChallengeIndicator	Optional parameter for EMV 3-D Secure (2.0) to be set tone of the values from the list below in order to indicate the preferred type of authentication. If no specific value is present in the transaction request, default value "01" is used.: 01 – no preference (set as default value) 02 – no challenge requested 03 – challenge requested 3-DS requestor preference 04 – challenge requested mandate 05 – No challenge requested (Transaction Risk Analysis is already performed) 06 – No challenge requested (Data Share Only) 07 – No challenge requested (SCA is already performed) 08 – No challenge requested (Utilise approved list exemption if no challenge required) 09 – Challenge requested ('Allow' prompt requested if challenge required)
threeDSTransType	The parameter for EMV 3-D Secure (2.0) represents the type of purchased item, mandatory for Visa and Brazilian market, otherwise optional. If no specific value is present in the transaction request, default value is used. 01 – Goods/ Service Purchase (default value) 03 – Check Acceptance 10 – Account Funding 11 – Quasi-Cash Transaction 28 – Prepaid Activation and Load

scaExemptionIndicator1	Optional parameter to request an exemption from Strong Customer Authentication (SCA) without the need to perform 3-D Secure authentication. Currently available values: ▪ Low Value Exemption ▪ TRA Exemption ▪ Trusted Merchant Exemption ▪ SCP Exemption ▪ Delegated Authentication ▪ Authentication Outage Exception Note this parameter is relevant only for the European merchants impacted by the PSD2 requirements.
skipTRA	This optional parameter allows you to use 3-D Secure even if the transaction has been evaluated as low risk and would be eligible for an exemption. Currently available values: ▪ true ▪ false When your store has been set up with Transaction Risk Analysis (TRA) service, but you do want to force 3-D Secure authentication for a certain transaction, set 'skipTRA' to 'true'. Note this parameter is relevant only for the European merchants impacted by the PSD2 requirements.
oid	Use this optional parameter to assign an identifier for your order; in case you plan to authenticate the transaction using EMV 3-DS protocol (aka 3-DS 2.1 or 2.2) only the following characters are allowed: A-Z, a-z, 0-9, "-"
deviceChannel	Use this optional parameter to request a 3DS Requestor Initiated flow to be performed. If no value is submitted in the request a default value "02" is automatically submitted by the Gateway. Currently available values: ▪ 02 – Browser Flow ▪ 03 – 3RI Flow
threeRIInd	In cases you require a 3RI flow to be performed, you must indicate the character of your request. Currently available values: ▪ 01 – Recurring transaction ▪ 02 – Instalment transaction ▪ 03 – Add card ▪ 04 – Maintain card information ▪ 05 – Account verification ▪ 06 – Split / delayed shipment ▪ 07 – Top-up ▪ 08 – Mail Order ▪ 09 – Telephone Order ▪ 10 – 'Allow list' status check ▪ 11 – Other payment
recurringExpiry	This field represents a date after which no further recurring transactions are performed, The field needs to be submitted in cases, where the first recurring or installment transaction is to be performed through 3-D Secure. If no specific value is present in the transaction request, the Gateway calculates the value based on populated recurring parameters.
recurringFrequency	Indicates the minimum number of days between authorisations for a recurring transaction and should include a numeric value between 1 and 9999. If no specific value is present in the transaction request, the Gateway calculates the value based on populated recurring parameters.

In principle, it may occur that 3-D 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 3-D Secure hold in a pending status while cardholders search for their password or need to activate their card for 3-D 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 3-Dsecure". If the session expires before the cardholder returns from the 3-D 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 3-D Secure transactions differs in some points compared to a normal transaction flow. If you already have an existing shop integration and plan to activate 3-D Secure subsequently, we recommend performing some test transactions on our test environment.



8.1 3-D Secure Split Authentication

If your business or technical processes require the cardholder authentication to be separated from the payment transaction (authorisation), 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="HMACSHA256"/>
    <input type="hidden" name="hashExtended" value="<% call createExtendedHash( "13.00","978" ) %>"/>
    <input type="hidden" name="storename" value="10123456789" />
```

```
    <input type="hidden" name="checkoutoption" value="combinedpage"/>
    <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=17/04/20 17:17:32,
ccbin=542606,
timezone=Europe/Berlin,
oid=C-2101f68a-45e9-4f3c-a6da-1337d5574717,
cccountry=N/A,
expmonth=12,
hash_algorithm=HMACSHA256
currency=978,
chargetotal=13.00,
approval_code=Y:ECI2/5:Authenticated,
hiddenSharedsecret=sharedsecret,
hiddenTxndatetime=2020:04:17-17:32:41,
expyear=2024,
response_hash=LarWYFSNgEToq13HlvyslX6hywi2T/nMn8jMY+1kxkI=,
response_code_3-Dsecure=1,
hiddenStorename=10123456789,
transactionNotificationURL=https://test.ipg-online.com/webshop/transactionNotification,
tdate=1491824253,
ignore_refreshTime=on,
ccbrand=Mastercard,
txntype=payer_auth,
paymentMethod=M,
txndatetime=2020:04:17-17:32:41,
cardnumber=(Mastercard) ... 4979,
ipgTransactionId=84120276797,
status=APPROVED}
```

In a second step, you need to submit a payment transaction ('sale' or 'preauth') via the LBOP 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 chapter Split Authentication, 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, possible values and format
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	Postcode

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

10. Data Vault

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

If you have ordered this feature, 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 account holder name 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 LBOP'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. There is no option to check, which existing 'hosteddataid' is holding duplicate cardholder data.

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

11. Recurring Payments

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

Field Name	Possible Values	Description
recurringInstallmentCount	Number between 1 and 999	Number of 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

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

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

12. Transaction Response

12.1 Response to your Success/Failure URLs

Upon completion, the transaction details will be sent back to the defined 'responseSuccessURL' or 'responseFailURL' as hidden fields. You can define these URLs in your transaction request. Alternatively, you can define them once in the Customisation section of our Virtual Terminal.

Field Name	Description, possible values and format
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

Field Name	Description, possible values and format
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 more below)
extended_response_hash	Hash-Value to protect the communication, where all response parameters are included in the hash calculation (see more 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 successful 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 MAESTRO Filled with "N/A" for any payment method which is not a credit card or debit card
schemeTransactionId	Credentials on file (COF) specific parameter. Returned in the response by a scheme for stored credentials transactions to be used in subsequent transaction request for future reference.

For 3-D Secure transactions only:

response_code_3-Dsecure	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 by Gateway) **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 3-D Secure (VISA ECI 06) **8** – Invalid 3-D Secure values received, most likely by the credit card issuing bank's Access Control Server (ACS) **9** – Challenge requested (Allow list prompt requested if challenge required) Please see note about blocking ECI 7 transactions in the 3-D Secure section of this document.

For Partial Approval:

partiallyApprovedAmount	Available balance as a partial amount approved.
status	Transaction status: 'PARTIALLY APPROVED'. This unique status allows you to identify this transaction and subtract the partially approved amount from the total transaction amount, and request another form of payment, using split-tender functionality.

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.

12.2 How to generate a hash for a response

Make sure to use the parameter 'response_hash' to recheck if the received transaction response has really been sent by LBOP to protect you from fraudulent manipulations. The value is created with a HMAC Hash using the following parameter string:

```
approval_code|chargetotal|currency|txndatetime|storename
```

Shared secret ('`sharedsecret`') will be used as a key in HMAC to calculate the hash with the above hash string. The hash algorithm is the same as the one that you have set in the transaction request.

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

You can also use the parameter 'extended_response_hash' to include all response parameters in the hash calculation. Please contact your local support team if you want to enable this feature. This will be managed with a specific setting performed on your account (service configuration 'extendedResponseHashSupported').

12.3 Creating the extended response hash

Step 1: Retrieve all non-empty Gateway specified response parameters and then remove the parameter 'extended_response_hash' from your list, so that it will not get included in the hash calculation. Consider also that shared secret will be used as a key in HMAC to calculate the hash and the hash algorithm must be the same as the one that you have set in the transaction request.

Step 2: Sort the response parameters in ascending order of the parameter names, where the upper-case characters come before the lower case (based on ASCII value). Join the parameters' values to one string with pipe separator (use only parameters' values and not the parameters' names).

Step 3: Pass the created string to the HMAC algorithm while using shared secret ('sharedsecret') as a key for calculating the hash value.

Step 4: Encode the result of HMAC with Base64 to generate the extended response hash.

Only HMAC algorithm (i.e.: HMACSHA256, HMACSHA384 or HMACSHA512) is supported for generating the extended response hash.

12.4 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 on 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|currency|txndatetime|storename|approval_code
```

Shared secret ('sharedsecret') will be used as a key in HMAC to calculate the hash with the above hash string.

Such notifications can also be set up for the recurring payments that get automatically triggered by the gateway. Please contact your local support team to get a shared secret ('rcpSharedSecret') agreed for these notifications. You can configure your Recurring Transaction Notification URL ('rcpTransactionNotificationURL') in the Customisation section of the Virtual Terminal.

In case of the recurring transactions the response hash parameter 'notification_hash' is calculated differently as follows:

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

The shared secret ('rcpSharedSecret') is part of the string (it is not used as a key in HMAC to calculate the hash with the hash string). Moreover, the response hash parameter for the recurring transaction notifications is calculated with the SHA256-value (as the default value).

Appendix I – How to generate a hash for a request

If you are using an HTML form to initiate a transaction, your request needs to include a security hash for verification of the message integrity.

The hash (parameter 'hashExtended') needs to be calculated using all non-empty gateway specified request parameters in ascending order of the parameter names, where the shared secret (parameter 'sharedsecret') must be used as the secret key for calculating the hash value. The gateway sorts the request parameters in the "natural order". For strings this means the "Lexicographic Order", thus the upper-case characters come before the lower case (based on ASCII value).

The request parameters that are not specified in our solution can still be submitted in your request to the gateway, but they must be excluded from the hash calculation. They will be ignored during processing and returned in the response.

When you are using Direct Post, there is also an option where you do not need to know the card details (PAN, CVV and Expiry Date) for the hash calculation. This will be managed with a specific setting performed on your store. Please contact your local support team if you want to enable this feature.

Creating the hash with all parameters

Transaction request values used for the hash calculation can be considered as a set of mandatory as well as optional gateway specified request parameters depending on the way you decide to build your request. See an example below:

- chargetotal= 13.00
- checkoutoption = combinedpage
- currency= 978
- hash_algorithm=HMACSHA256
- paymentMethod=M
- **responseFailURL=https://localhost:8643/webshop/response_failure.jsp**
- **responseSuccessURL=https://localhost:8643/webshop/response_success.jsp**
- storename=10123456789
- timezone= Europe/Berlin
- transactionNotificationURL=https://localhost:8643/webshop/transactionNotification
- txndatetime= 2021:09:06-16:43:04
- txntype=sale
- sharedsecret=sharedsecret (to be used as the secret key for calculating the hash value)

The steps below provide the guidelines on how to calculate a hash, while using the values from our example.

Step1. Extended hash needs to be calculated using all non-empty gateway specified request parameters in ascending order of the parameter names, where the upper-case characters come before the lower case (based on ASCII value). Join the parameters' values to one string with pipe separator (use only parameters' values and not the parameters' names).

stringToExtendedHash = 13.00|combinedpage|978|HMACSHA256|M|https://localhost:8643/webshop/response_failure.jsp|https://localhost:8643/webshop/response_success.jsp|10123456789|Europe/Berlin|https://localhost:8643/webshop/transactionNotification|2021:09:06-16:43:04|sale

Corresponding hash string does not include 'sharedsecret', which has to be used as the secret key for the HMAC instead.

Step2. Pass the created string to the HMACSHA256 algorithm and using shared secret as a key for calculating the hash value.

HmacSHA256(stringToExtendedHash, sharedsecret)

Step3. Step 3. Encode the result of HMACSHA256 with Base64 and pass it to the gateway as part of your request.

Base64:

EapafBqqOF6N/kch8USkHPGh+fwSko24h6FpQnQHfQ8=

```
<input type="hidden" name="hashExtended" value="
EapafBqqOF6N/kch8USkHPGh+fwSko24h6FpQnQHfQ8="/>
```

Only HMAC algorithm (i.e.: HMACSHA256, HMACSHA384 or HMACSHA512) is supported for generating the extended request hash.

Appendix II – ipg-util.asp

```
<!-- google CryptoJS for HMAC -->
<script LANGUAGE=JScript RUNAT=Server src="script/cryptoJS/crypto-js.min.js"></script>
<script LANGUAGE=JScript RUNAT=Server src="script/cryptoJS/enc-base64.min.js"></script>
<script LANGUAGE=JScript RUNAT=Server>
var today = new Date();
var txndatetime = today.formatDate("Y:m:d-H:i:s");
/*
Function that calculates the hash of the following parameters as an example:
- chargetotal
- checkoutoption
- currency
- hash_algorithm
- paymentMethod
- responseFailURL
- responseSuccessURL
- storename
- timezone
- transactionNotificationURL
- txndatetime
- txntype
- and sharedsecret as the secret key for calculating the hash value
*/
function createExtendedHash(chargetotal, currency) {
// Please change the storename to your individual Store Name
var storename = "10123456789";
// NOTE: Please DO NOT hardcode the secret in that script. For example read it from a database.
var stringToExtendedHash = chargetotal|checkoutoption|currency|hash_algorithm|paymentMethod|response
FailURL|responseSuccessURL|storename|timezone|transactionNotificationURL|txndatetime|txntype;
var hashHMACSHA256 = CryptoJS.HmacSHA256(stringToExtendedHash, sharedSecret);
var extendedhash = CryptoJS.enc.Base64.stringify(hashHMACSHA256);
Response.Write(extendedhash);
}
function getDateTime() {
Response.Write(txndatetime);
}
</script>
```

Appendix III – ipg-util.php

```
<!DOCTYPE HTML>
<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">
<fieldset>
<legend>IPG Connect Request Details</legend>
<p>
<label for="storename">Store ID:</label>
<input type="text" name="storename" value="10123456789" readonly="readonly" />
</p>
<p>
<label for="timezone">Timezone:</label>
<input type="text" name="timezone" value="Europe/London" readonly="readonly"/>
</p>
<p>
<label for="chargetotal">Transaction Type:</label>
<input type="text" name="txntype" value="sale" readonly="readonly" />
</p>
<p>
<label for="chargetotal">Transaction Amount:</label>
<input type="text" name="chargetotal" value="13.00" readonly="readonly" />
</p>
<p>
<label for="currency">Currency (see ISO4217):</label>
<input type="text" name="currency" value="978" readonly="readonly" />
</p>
<p>
<label for="txndatetime">Transaction DateTime:</label>
<input type="text" name="txndatetime" value="<?php echo getDateTime(); ?>"/>
</p>
<p>
<label for="hashExtended">Hash Extended:</label>
<input type="text" name="hashExtended" value="<?php echo createExtendedHash('13.00', '978'); ?>"
readonly="readonly" />
</p>
<p>
```

```
<label for="hashExtended">Hash Algorithm :</label>
<input type="text" name="hash_algorithm" value="HMACSHA256" readonly="readonly" />
</p>
<p>
<label for="hashExtended">Checkout option :</label>
<input type="text" name="checkoutoption" value="combinedpage" readonly="readonly" />
</p>
<p>
<input type="submit" id="submit" value="Submit" />
</p>
</fieldset>
</form>
<?php
function getDateTime() {
return date("Y:m:d-H:i:s");
}
function createExtendedHash($chargetotal, $currency) {
// Please change the store Id to your individual Store ID
// NOTE: Please DO NOT hardcode the secret in that script. For example read it from a database.
$sharedSecret = "sharedsecret";
$separator = "|";
$storeId= "10123456789";
$timezone= "Europe/London";
$txntype= "sale";
$checkoutoption = "combinedpage";
$stringToHash = $chargetotal . $separator . $checkoutoption . $separator . $currency . $separator
. "HMACSHA256" . $separator . $storeId . $separator . $timezone. $separator . date("Y:m:d-H:i:s") .
$separator . $txntype;
$hash = base64_encode(hash_hmac('sha256', $stringToHash, $sharedSecret, true));
return $hash;
}
?>
</body>
</html>
```

The above is the working PHP example, to run it you can copy the above and paste it on **https://www.w3schools.com/php/phptryit.asp?filename=tryphp_function1**

Appendix IV – Currency Code List

Currency name	Currency code	Currency number
CFA Franc BCEAO	XOF	952
Afghan Afghani	AFN	971
Albanian	ALL	008
Algerian Dinar	DZD	012
Argentine Peso	ARS	032
Armenian Dram	AMD	051
Aruban Florin	AWG	533
Australian Dollar	AUD	036
Azerbaijanian Manat	AZN	944
Bahamian Dollar	BSD	044
Bahrain Dinar	BHD	048
Bangladeshi Taka	BDT	050
Barbados Dollar	BBD	052
Belarussian Ruble	BYN	933
Belize Dollar	BZD	084
Bermudian Dollar	BMD	060
Bolívar Soberano	VES	928
Bolivian Boliviano	BOB	068
Bosnian Convertible	BAM	977
Botswana Pula	BWP	072
Brazilian Real	BRL	986
British Pound	GBP	826
Bruneian Dollar	BND	096
Bulgarian Lev	BGN	975
Burundi Franc	BIF	108
Cambodian Riel	KHR	116
Canadian Dollar	CAD	124
Cape Verdean (Cabo Verde Escudo)	CVE	132
Cayman Islands Dollar	KYD	136
Central African CFA	XAF	950
CFP	XPF	953
Chilean Peso	CLP	152
Chinese Renminbi	CNY	156
Colombian Peso	COP	170
Comorian Franc	KMF	174

Currency name	Currency code	Currency number
Congolese Franc	CDF	976
Costa Rican Colon	CRC	188
Croatian Kuna	HRK	191
Cuban Peso	CUP	192
Czech Koruna	CZK	203
Danish Krone	DKK	208
Djiboutian Franc	DJF	262
Dobra	STN	930
Dominican Peso	DOP	214
East Caribbean Dollar	XCD	951
Egyptian Pound	EGP	818
Ethiopian Birr	ETB	230
Euro	EUR	978
Falkland Islands Pound	FKP	238
Fijian Dollar	FJD	242
Gambian Dalasi	GMD	270
Georgian Lari	GEL	981
Gibraltar Pound	GIP	292
Gourde	HTG	332
Guatemalan Quetzal	GTQ	320
Guinea Franc	GNF	324
Guyanese Dollar	GYD	328
Honduran Lempira	HNL	340
Hong Kong Dollar	HKD	344
Hungarian Forint	HUF	348
Iceland Krona	ISK	352
Indian Rupee	INR	356
Indonesian Rupiah	IDR	360
Iranian Rial	IRR	364
Iraqi Dinar	IQD	368
Israeli New Shekel	ILS	376
Jamaican Dollar	JMD	388
Japanese Yen	JPY	392
Jordanian Dinar	JOD	400
Kazakhstani Tenge	KZT	398
Kenyan Shilling	KES	404
Kuwaiti Dinar	KWD	414
Kwanza	AOA	973

Currency name	Currency code	Currency number
Laotian Kip	LAK	418
Lebanese Pound	LBP	422
Liberian Dollar	LRD	430
Libyan Dinar	LYD	434
Lilangeni	SZL	748
Loti	LSL	426
Macau Pataca	MOP	446
Macedonian Denar	MKD	807
Malagasy Ariary	MGA	969
Malawian Kwacha	MWK	454
Malaysian Ringgit	MYR	458
Maldivian Rufiyaa	MVR	462
Mauritian Rupee	MUR	480
Mexican Peso	MXN	484
Moldovan Leu	MDL	498
Mongolian Tugrik	MNT	496
Moroccan Dirham	MAD	504
Mozambique Metical	MZN	943
Mvdol	BOV	984
Myanmar Kyat	MMK	104
Nakfa	ERN	232
Namibia Dollar	NAD	516
Nepalese Rupee	NPR	524
Netherlands Antillean Guilder	ANG	532
New Zealand Dollar	NZD	554
Ngultrum	BTN	064
Nicaraguan Cordoba Oro	NIO	558
Nigerian Naira	NGN	566
Norwegian Krone	NOK	578
Omani Rial	OMR	512
Ouguiya	MRU	929
Pakistani Rupee	PKR	586
Panamanian Balboa	PAB	590
Papua New Guinean Kina	PGK	598
Paraguayan Guarani	PYG	600
Peruvian Nuevo Sol	PEN	604
Philippine Peso	PHP	608
Polish Zloty	PLN	985
Qatari Rial	QAR	634

Currency name	Currency code	Currency number
Romanian New Leu	RON	946
Russian Ruble	RUB	643
Rwandan Franc	RWF	646
Saint Helena Pound	SHP	654
Salvador Colon	SVC	222
Samoan Tala	WST	882
Saudi Rihal	SAR	682
Serbian Dinar	RSD	941
Seychelles Rupee	SCR	690
Sierra Leonean	SLL	694
Singapore Dollar	SGD	702
Solomon Islands Dollar	SBD	090
Som	KGS	417
Somali Shilling	SOS	706
South African Rand	ZAR	710
South Korean Won	KRW	410
South Sudanese Pound	SSP	728
Sri Lanka Rupee	LKR	144
Sudanese Pound	SDG	938
Surinamese Dollar	SRD	968
Swedish Krona	SEK	752
Swiss Franc	CHF	756
Syrian Pound	SYP	760
Taiwan Dollar	TWD	901
Tajikistani Somoni	TJS	972
Tanzanian Shilling	TZS	834
Thai Baht	THB	764
Tongan Pa'anga	TOP	776
Trinidad and Tobago Dollar	TTD	780
Tunisian Dinar	TND	788
Turkish Lira	TRY	949
Turkmenistan New Manat	TMT	934
Uganda Shilling	UGX	800
Ukrainian Hryvnia	UAH	980
UAE Dirham	AED	784
US Dollar	USD	840
Uruguayan Peso	UYU	858
Uzbekistan Sum	UZS	860
Vanuatu Vatu	VUV	548

Currency name	Currency code	Currency number
Vietnamese Dong	VND	704
Yemeni Rial	YER	886
Zambian Kwacha	ZMW	967
Zimbabwe Dollar	ZWL	932

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.

Payment Method	Value
American Express	A
Apple Pay on the web	applePay
Diners	C
Google Pay on the web	googlePay
Maestro	MA
Maestro UK	maestroUK
Mastercard	M
PayPal	paypal
Visa (Credit/Debit/Electron/Delta)	V

Appendix VI – PayPal

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

Connect Transaction Type (txntype)	PayPal operation
Sale	SetExpressCheckoutPayment (sets PaymentAction to Authorisation in SetExpressCheckout and DoExpressCheckoutPayment requests)
Preauth	GetExpressCheckoutDetails
sale – with additional parameters for installing a Recurring Payment	DoExpressCheckoutPayment*
Postauth	DoCapture (,DoReauthorisation)
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.

Recurring Payment Transaction

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

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

Appendix VII – Digital Wallets

Refer to the following information only when you are integrating Google Pay or/and Apple Pay on the web as a payment method.

Google Pay on the web

Google Pay is a digital wallet solution provided by participating banks and supported by Google. It allows users to store cards from participating banks. To learn more about Google Pay, please visit **https://pay.google.com/about/**

Initiating a transaction (Checkout Process)

The checkout process for Google Pay can be initiated with a "Google Pay" 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 transaction request, with the required parameters including the payment method parameter. This will take your customers to a hosted page from where they can be redirected to the Google Pay payment screen, with list of cards added to customer Google Pay wallet. Selecting the card by customers from the list and clicking the 'Pay' button would complete the payment.

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

Apple Pay on the web

Apple Pay on the web allows making purchases on the web in Safari on your iPhone, iPad, or Mac, you can use Apple Pay without having to create an account or fill out lengthy forms. Moreover, with Touch ID on MacBook Air and MacBook Pro, paying takes just a touch and is quicker, easier, and more secure than ever before. To learn more about Apple Pay on the web, please visit **https://developer.apple.com/documentation/apple_pay_on_the_web**

Initiating a transaction (Checkout Process)

The checkout process for Apple Pay on web can be initiated in Safari browser with "Apple Pay" 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 transaction request, with the required parameters including the payment method parameter. This will take your customers directly to the Apple Pay payment screen, with list of cards added to customers' Apple Pay wallet. Selecting the card by customers from the list and authenticate using Touch id/Face id on Apple device would complete the payment.

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

Apple Pay on the web transaction can only be initiated with Apple's Safari browser and authorisation from an iOS device like iPhone, Apple Watch or MacBook.

The generic fields to be considered:

Field Name	M/O	Description, possible values and format
checkoutoption	M	Set the value for this parameter to 'combinedpage'
paymentMethod	O	Set the value for this parameter to 'googlePay' or 'applePay' If you do not submit this parameter, gateway will display a page to your consumer to choose from the payment methods activated for your store.

Find out more

Go to lloydsbank.com/business

Please contact us if you'd like this information in an alternative format such as Braille, large print or audio.

Important information

We accept calls made via Text Relay. We may monitor or record phone calls with you in case we need to check we have carried out your instructions correctly and to help improve the quality of our service. Lloyds Bank is a trading name of Lloyds Bank plc and Bank of Scotland plc. Lloyds Bank plc. Registered Office: 25 Gresham Street, London EC2V 7HN. Registered in England and Wales No. 2065. Bank of Scotland plc. Registered Office: The Mound, Edinburgh EH1 1YZ. Registered in Scotland No. SC327000. Authorised by the Prudential Regulation Authority and regulated by the Financial Conduct Authority and the Prudential Regulation Authority.

We aim to provide the highest level of customer service possible. If you do experience a problem, we will always seek to resolve this as quickly and efficiently as possible.

If you would like a copy of our complaint procedures, please contact your relationship manager or any of our offices. You can also find details at lloydsbankcommercial.com/contactus

LLOYDS BANK

COMM0035 (07/23)