Server Protocol and Integration Guideline

Document Index

Welcome to the Sage Pay Server Integration Method

The Sage Pay Payment system provides a secure, simple means of authorising credit and debit card transactions from your web site.

The Sage Pay system provides a straightforward payment interface for the customer, and takes complete responsibility for the online transaction, including the collection and encrypted storage of credit and debit card details, eliminating the security implications of holding such sensitive information on your own servers.

The Sage Pay Server Integration method is our flagship, and original system. Unlike many online Payment Service Providers who receive transaction details through hidden fields sent via the customer's browser (which could be tampered with or spoofed), the Server integration talks directly to your web server over a 128-bit encrypted secure channel, exchanging digitally signed messages to register the transaction and notify you directly of the authorisation results. No sensitive information is sent via the customer's browser, and because the customer is redirected to Sage Pay, no card details need to be taken or stored on your site (removing the need for you to maintain highly secure encrypted databases, obtain digital certificates or undergo auditing against the Visa and MasterCard PCI-DSS security standard).

This document explains how your Web servers communicate with Sage Pay Server, goes on to explain how to integrate with our testing and live environments, and contains the complete Sage Pay Server Payment Protocol in the Appendix.

PLEASE NOTE: Originally the Sage Pay Form integration was called the "Verified Payment System" and was referred to by the acronym VPS. When the product expanded to support a variety of interfaces, the name was changed to reflect the type of interface used. You may see references made to either VSP or VPS throughout the document, (for example, VPSTxId or VPSProtocol); these are not transposition errors. These field names date back to the original system and have remained unchanged because the protocols have been *extended* rather than replaced. With the rebrand change from Protx to Sage Pay the VSP prefix was dropped from the solution names, but VPS still features in some parts of the Protocol.

Overview of how Server integrated Payments work

The final “Pay Now” button on your website is your link to the Sage Pay System. Once the customer has selected their purchases, entered delivery details, billing address and so forth, all on your own site, and pressed the final pay or proceed button, a HTTPS post is sent from your servers to Sage Pay, registering the transaction. In response we return a registration Status, further transaction identifiers (which you store in your database) and a URL to which your site should redirect the customer.

The redirected customer arrives on the Sage Pay Server payment page where they enter their credit/debit card details, security codes and address (if you have not already captured it). The Sage Pay Server main page carries your logo, and a description (sent by your site) of the goods the customer is paying for, so they can remain confident they are buying from you. You can even customise those payment pages to carry the look and feel of your site at no additional cost.

Once the customer has selected their payment method and confirmed they really wish to complete the payment, our Server requests authentication from the card issuing bank (where appropriate), then requests authorisation from your acquiring bank. Once the bank has authorised the payment (and assuming the address and card value checks have passed any rules you may have set up), we send a HTTP or HTTPS POST directly to your web servers, informing you of the outcome. Anti-tampering mechanisms are attached to the POST, so that you can confirm the server messages have not been modified in any way.

Having received this POST, your site confirms the transaction status against your own records and replies to us with a final redirection URL. The Server integration method then redirects your customer back to your website for confirmation of their order and any other completion pages you wish to display.

Sage Pay provides Integration Kits, which are simple worked examples in various different scripting languages that perform all the tasks described above. You simply customise these to work with your particular environment. So whether you are running .NET, ASP, PHP, Coldfusion or PERL, and whether your servers are Linux Apache, Win32 IIS or Domino, we’ve already done half of the work for you.

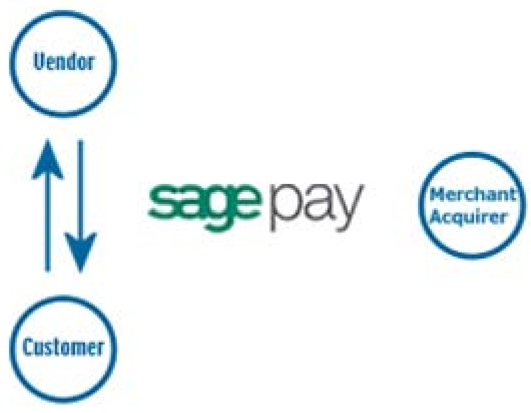
The following sections explain the integration process in more detail. The complete Server Payment integration protocol is attached in the appendix, providing a detailed breakdown of the contents of the HTTPS messages sent between your servers and ours during a transaction.

A companion document, “Server and Direct Shared Protocols”, gives details of how to perform other transaction related POSTs, such as Refunds, Repeat payments, additional Authorisations and the Release/Abort mechanisms for Deferred transactions.

The Server Payment integration Process in Detail

This section details the messages exchanged between your Web servers and the Sage Pay's Server system.

Step 1: The customer orders from your site.



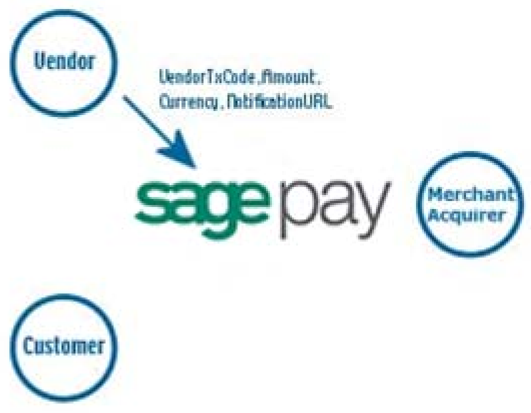
A payment begins with the customer ordering goods or services from your site. This process can be as simple as selecting an item from a drop down list, or can involve a large shopping basket containing multiple items with discounts and delivery charges. Your interaction with your customer is entirely up to you and the Server integrated system only requires you to collect a few compulsory pieces of information, which are detailed in the latter part of this guide.

It is generally a good idea to identify the customer by name, e-mail address, delivery and billing address and telephone number. It is also helpful to have your server record the IP Address from which the user is accessing your system. You should store these details in your database alongside details of the customer's basket contents or other ordered goods.

YOU DO NOT NEED TO COLLECT CREDIT OR DEBIT CARD DETAILS. All your site needs to do is calculate the total cost of the order in whatever currency your site operates and present the user with a confirmation page, summarising their order. On this page there will be a Proceed or Continue button which, when clicked, will initiate the payment process outlined in the following sections.

Step 2: Your server registers the payment with Sage Pay.

Once the user has clicked Continue, a script on your web server will construct a payment registration message (see Appendix A1) and POST it via HTTPS to the Sage Pay Server transaction registration URL.



This POST contains your **Vendor Name** (chosen by you on the Sage Pay online application form, or assigned to you by Sage Pay when your account is created) and your own unique reference to this payment (in a field called **VendorTxCode**, which you must ensure is a completely unique value for each transaction).

The message also contains the total value and currency of the payment, and billing address details for the customer. You must specify a brief description of the goods bought to appear on the payment screen and provide a URL for the Sage Pay servers to call back to once the payment process is complete (this is called the **NotificationURL**).

Because this message is POSTed directly from your servers to ours across a 128-bit encrypted session, no sensitive information is passed via the customer's browser, and anyone who attempted to intercept the message would not be able to read it. Using the Server integration method, you can be assured that the information you send to us cannot be tampered with or understood by anyone other than us. Your script sends the payment registration message in the Request object of the HTTPS POST and the response from our server (see step 3 below) is in real time in the Response object of the same POST.

The integration kits we provide contain scripts in a variety of languages that illustrate how you compose and send this message from your server to ours. Please visit the developer's area of the website at www.sagepay.com/developers.html to obtain your kit.

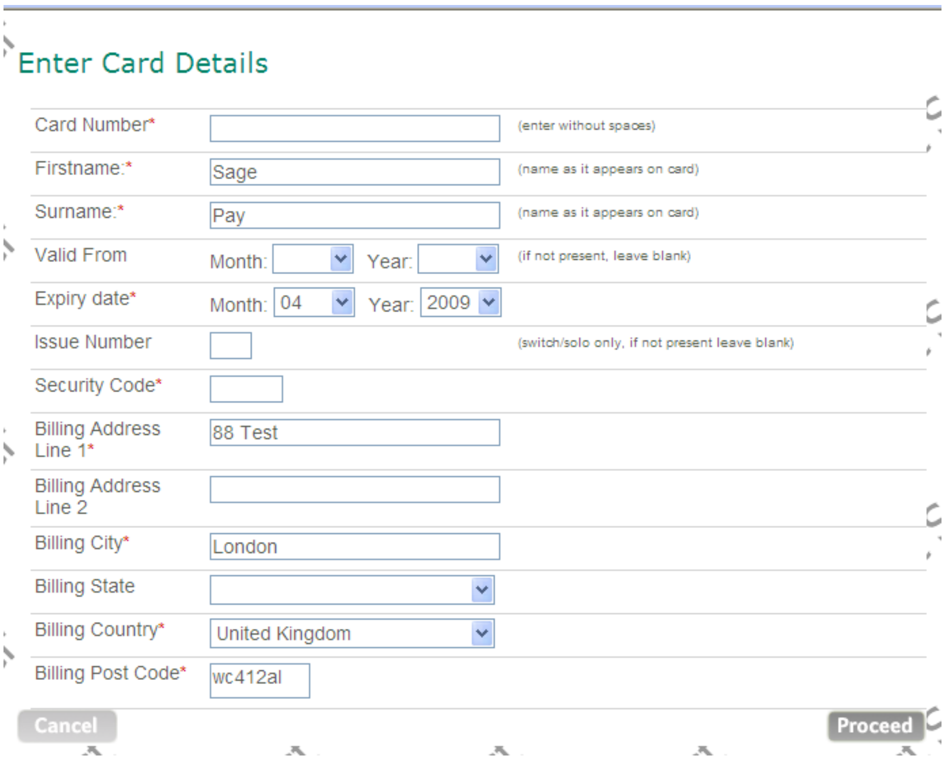
LOW PROFILE Payment Pages

With the Server integration, you now have the option of using **LOW PROFILE** payment pages, enabling you to select the minimal payment page screens instead of the normal default set.

Low Profile templates are designed to run inside Iframes and present simple HTML pages with no pop-ups, limited formatting and minimal graphics. Whilst all transaction information passed between the merchant site and the Sage Pay Systems is encrypted using our high-security SSL certificates, from a customer's perspective, the secure padlock will not display in the main browser window. You must therefore ensure you obtain your own SSL certificate for these pages.

Please note that you will NOT be able to accept PayPal transactions with Low Profile templates enabled.

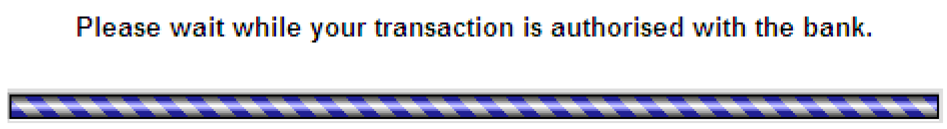
The Low Profile option simply displays a Card Details page to the shopper (rather than the initial 'card selection' screen), asking for the Card details and Billing Address details.



Enter Card Details

Card Number*		(enter without spaces)
Firstname:*	Sage	(name as it appears on card)
Surname:*	Pay	(name as it appears on card)
Valid From	Month: Year:	(if not present, leave blank)
Expiry date*	Month: 04 Year: 2009	
Issue Number		(switch/solo only, if not present leave blank)
Security Code*		
Billing Address Line 1*	88 Test	
Billing Address Line 2		
Billing City*	London	
Billing State		
Billing Country*	United Kingdom	
Billing Post Code*	wc412al	

If 3D Secure is active on your account, the customer is redirected to the card issuing bank's 3D Secure page, (presented as a full screen HTML page). Once the shopper completes 3D Authentication, (or if 3D Secure is disabled on your account), the shopper is presented with the page below whilst Sage Pay contact the bank for authorisation.



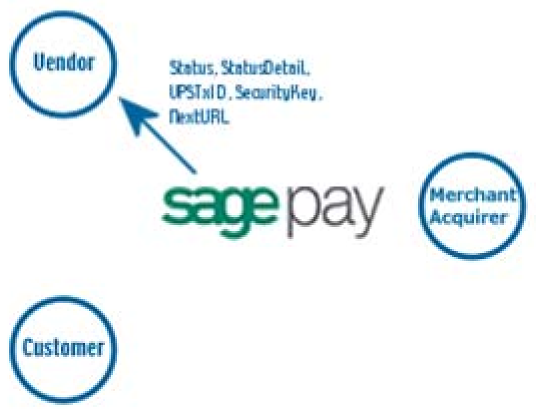
Please wait while your transaction is authorised with the bank.

If you would like to use the Low profile payment pages, you will need to pass a value of “LOW” in the ‘Profile’ field included in the Transaction Registration Post (see Appendix A1).

You have the option of customising the Low Profile pages, (including the ‘ReadOnly’ and ‘NoAddress’ options) so that the look and feel of the payment pages is similar to your own site. For further information about how you can customise the LOW PROFILE payment pages, please refer to the Sage Pay Custom Templates Kit, which can be obtained from the developer’s area on our website:

www.sagepay.com/developers.html

Step 3: Sage Pay reply to the payment registration POST.



On receipt of your POST, our Server begins by validating its contents.

It first checks to ensure all the required fields are present, and that their format is correct. If any are not present a reply with a **Status** of **MALFORMED** is generated, with the **StatusDetail** field containing a human readable error message. This normally only happens in the development stage.

If all fields are present, the information in those fields is then validated. The Vendor name is checked against a pre-registered set of IP addresses, so our Server can ensure the POST came from a recognised source. The currency of the transaction is validated against those accepted by your merchant accounts. The VendorTxCode is checked to ensure it has not been used before. The amount field is validated. Flag fields are checked, in fact, every field is checked to ensure you have passed the correct values. If any of the information does not check out, a reply with a **Status** of **INVALID** is returned, again with a human readable error message in **StatusDetail**.

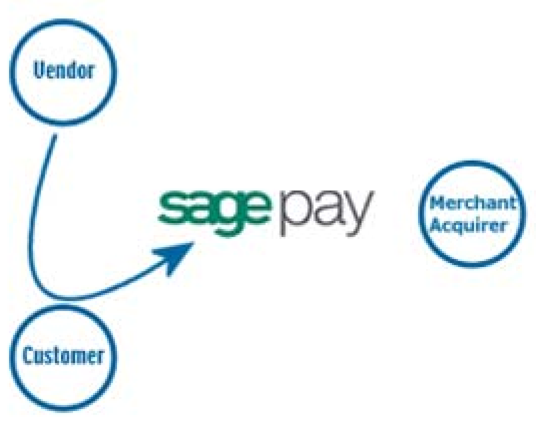
If you receive either a MALFORMED or INVALID message you should use the detailed response in the StatusDetail error message to help debug your scripts. If you receive these messages on your live environment, you should inform your customer that there has been a problem registering their transaction, then flag an error in your back-office systems to help you debug. You can e-mail the Sage Pay Support team (support@sagepay.com) for help with your debugging issues.

If everything in the original POST checks out, the transaction is registered with the Sage Pay Server system and a new transaction code is generated that is unique across ALL vendors using our payment systems, not just unique to you. This code, the **VPSTxId**, is our unique reference to the transaction, and is sent back to you in the reply along with a **Status** of **OK** and a blank StatusDetail field.

An OK message also contains a **SecurityKey** field. This is a ten character long alphanumeric string used as a key for confirming the MD5 hash signature in the notification post (see Step 10 below).

You should store the **VPSTxId** and **SecurityKey**, along with your own **VendorTxCode**, in your database alongside the customer and order details for this transaction.

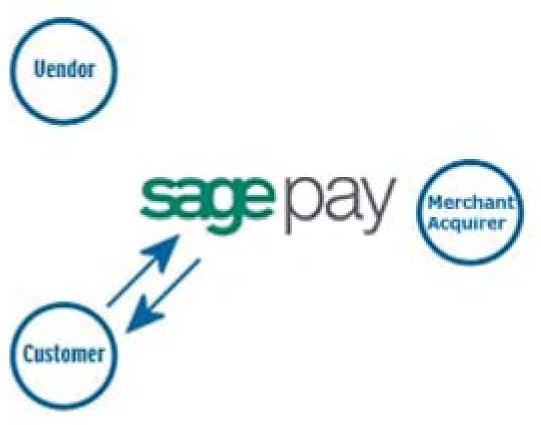
The final component of the reply is a field called **NextURL**, which is the page to which you should redirect the customer to allow them to continue with their purchase.



If the Status is OK, your script should send a redirect request containing this URL to your customer's browser.

This is the first stage at which anything noticeable has happened at the customer end. The HTTPS POST and response described above are completely invisible to the customer. As far as the customer is concerned they clicked the "continue" button and now find themselves on Sage Pay's payment page.

Step 4: Customer enters card details on Sage Pay's Server.



The customer is presented with a page requesting their credit/debit card details. This page will contain your company logo and the description of goods passed in Step 2 above. As explained above, you can elect to use the LOW PROFILE payment pages, and you can customise these pages further by producing your own custom templates (please contact templates@sagepay.com if you require more information about custom templates).

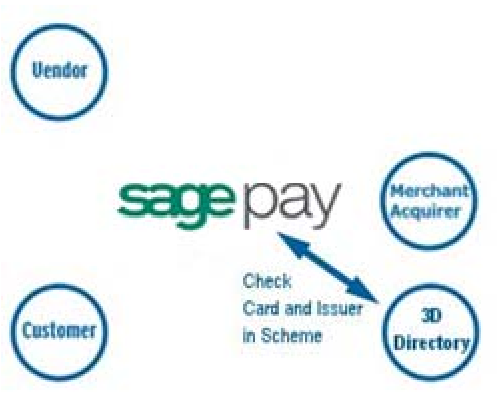
Once the customer has entered their details, our Server system verifies that information prior to communicating with the bank, to ensure the card number is valid, the card type matches the card number, the expiry date is not in the past and, where appropriate, the issue number and start date are in the correct format.

If valid card details have been entered, the customer is presented with a confirmation screen where they have one last chance to change their mind and cancel the transaction. If the customer decides to cancel, you will be sent a cancellation message at the notification stage and nothing is communicated to the banks (jump to Step 9).

If the customer wishes to continue, the Sage Pay Servers checks with either the Visa or MasterCard 3D-Secure schemes to determine if the customer's card can be authenticated (see step 5 below), or if you have not enrolled for 3D-Secure processing, initiates the process of obtaining an authorisation code from the merchant acquirer (jump to step 8).

Step 5: The Sage Pay MPI checks 3D-Secure enrolment.

The Sage Pay's Server sends the card details provided by your customer to the Sage Pay 3D-Secure Merchant Plug-In (MPI). This formats a request called a VEReq, which is sent to the 3D-Secure directory servers to query whether the card and card issuer are part of the 3D-Secure scheme.



The 3D Secure directory servers send a VERes response back to the MPI where it is decoded and our server informed of the inclusion or exclusion of the card.

If the card or the issuer are not part of the scheme, or if an MPI error occurs, our server will check your 3D-Secure rule base to determine if authorisation should occur. By default you will not have a rule base established and transactions that are not, or cannot, be 3D-authenticated will still be forwarded to your acquiring bank for authorisation.

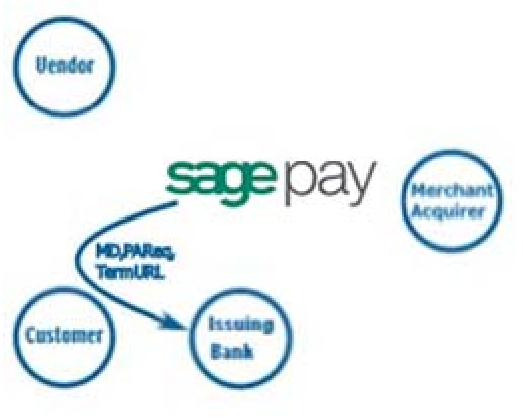
If you do have a rule base, the value of the transaction and the AllowCardNotInScheme, AllowIssuerNotInScheme and AllowMPIErrors flags will determine if authorisation should be attempted.

If your rule base rejects the transaction due to your criteria not being reached, the Sage Pay Server replies with **Status** of **REJECTED** and **StatusDetail** indicating the reason for the rejection in the Notification POST (see step 9 below). The **3DSecureStatus** field will contain the results of the 3D-Secure lookup. REJECTED transactions will never be authorised and the customer's card never charged, so your code should redirect your customer to an order failure page, explaining why the transaction was aborted.

If your rule base DOES allow authorisation to occur for non-3D-authenticated transactions, the Sage Pay Server continues with the authorisation process (jump ahead to step 8).

If the card and the card issuer are both part of the scheme, our server will pass the customer across to their card issuing bank for authentication.

Step 6: Sage Pay Server redirects your customer to their Card Issuing Bank.

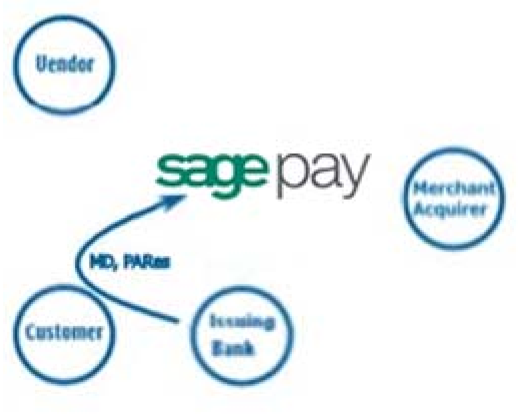


The customer's browser is redirected to their Card Issuing Bank's 3D-Secure authentication pages. These vary from bank to bank, but their purpose is to require the user to authenticate themselves as the valid card holder.

3D-Secure is much like an online version of Chip and PIN. The customer must answer questions at their card issuer site (these might be a simple password, characters from a password, or a full challenge-response mechanism using a two-factor authentication device, depending on the level of security emplyed by the bank) and in so doing, the bank is validating the customer's right to use the card for the transaction on your site.

If they determine that the person attempting the transaction IS the real card holder, they assume the liability for fraudulent use of that card and you are protected from what are known as Chargebacks if the cardholder claims in future that they did not use the card. **This level of protection for you is ONLY afforded by 3D-Secure, which is why it is a good idea to enable it on your merchant account through Sage Pay.**

Step 7: The Issuing Bank returns the customer back to the Sage Pay Server.

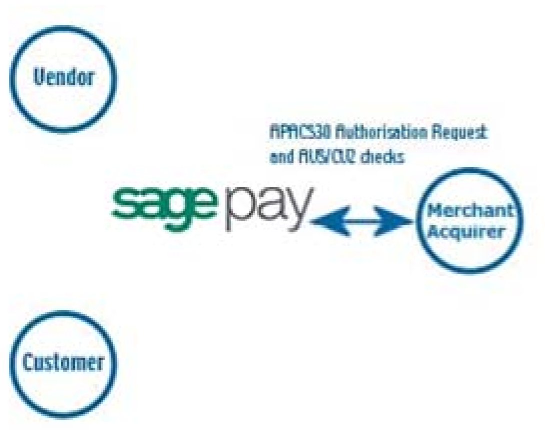


If the customer successfully completes 3D-Authentication with their bank, they are redirected to Sage Pay along with a unique authentication value (called CAVV for cards issued by Visa, and UCAF for MasterCard issued cards). This is passed to your acquiring bank during authorisation (see step 8 below) to secure the liability shift for the transaction.

If the customer does not successfully 3D-Authenticate with their issuing bank, they are passed back to the Sage Pay Server anyway, but without the CAVV/UCAF value. At this stage the Sage Pay Server system consults your 3D-Secure rule base to see if authentication should be attempted. By default 3D-Authentication failures are NOT sent for authorisation, but all other message types are. Refer to the Sage Pay Rule base guide for more information about using 3D-Secure and AVS/CV2 rules.

If authorisation is not possible, our server will POST a **REJECTED** message to your NotificationURL (see step 9 below), otherwise an authorisation will be gained from your acquiring bank (see step 8)

Step 8: Sage Pay Server requests card authorisation.



The Sage Pay servers format a bank specific authorisation message (including any 3D-Secure authentication values where appropriate) and pass it to your merchant acquirer over the private banking network.

The request is normally answered within a second or so with either an authorisation code, or a failure message.

Whilst communicating with the merchant acquirer, the customer is shown a page containing the text, "Please wait while your transaction is authorised with the bank"

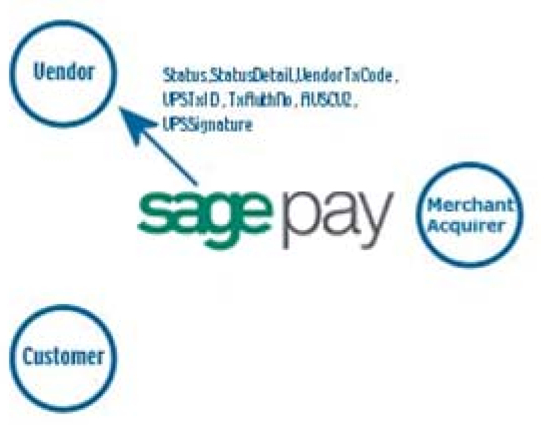
The Sage Pay Server system handles all authorisation failures in the same way, replying to your site with a **NOTAUTHED** message and a blank authorisation code.

If the acquirer does return an Authorisation code, our server prepares an **OK** response to send back to you (next step).

If AVS/CV2 fraud checks are being performed, the results are compared to any rule bases you have set up (see the Fraud Screening companion documentation for more information). If the bank has authorised the transaction but the card has failed the fraud screening rules you have established, the Sage Pay Server immediately reverses the authorisation with the bank, requesting the shadow on the card for this transaction to be cleared, and prepares a **REJECTED** response for your web site.

Please note: Some card issuing banks may decline the online reversal which can leave an authorisation shadow on the card for up to 10 working days. The transaction will never be settled by Sage Pay and will appear as a failed transaction in *My Sage Pay.*

Step 9: Sage Pay Server contacts your NotificationURL



The Sage Pay Server sends a HTTP or HTTPS POST to the NotificationURL script on your server to indicate the outcome of the transaction.

This POST contains a **Status** field that holds either **OK**, if the transaction was authorised at step 8, **NOTAUTHED** if the authorisation was failed by the bank, **ABORT** if the user decided to cancel the transaction whilst on our payment pages, **REJECTED** if your fraud screening rules were not met, or **ERROR** if an error has occurred at Sage Pay (these are very infrequent, but your site should handle them anyway. They normally indicate a problem with bank connectivity).

The **StatusDetail** field of the POST contains further human readable details about the Status field.

The URL to which the completion message is POSTed is the **NotificationURL** sent in the original transaction registration (see Step 2 above), or the pre-registered address held in the Sage Pay Server database (which can be specified when you set up your account). Most people prefer to specify the NotificationURL on a per transaction basis because this gives the greatest degree of flexibility.

The transaction authorisation results are ALWAYS POSTed to your Notification URL, so whether the Status is OK, NOTAUTHED, REJECTED, ABORT or ERROR, your Notification script must decide how to process each message type and redirect the user accordingly. The integration kits have example pages that show how to process the Notification POST.

The Notification POST can be over HTTPS if you have an SSL certificate securing your web site. If you do not then the POST will just be HTTP, which means it will be plain text and not encrypted. The problem with plain text POSTs is that a clever hacker could intercept the packets of information and modify the response before sending it on to you (although we must stress this is a very complex and difficult process). They could, for example, change a NOTAUTHED message to an OK message. To counteract this, the Notification POST has a **VPSSignature** field attached to sign the POST (which is an MD5 hash of the contents of the message. See below for more information)

Your Notification script should read the **VendorTxCode** and **VPSTxId** from the POST and retrieve the relevant information about the order from your database, including, most importantly, the **SecurityKey** for the transaction (that was sent back to your servers in step 2)

Using the SecurityKey and the contents of the notification POST, your script can reconstruct that message and run it through a MD5 Hash algorithm. Hash algorithms are one-way functions (that is, if you pass the same data through the same algorithm you'll get the same signature value every time you run it. There is, however, no way to regenerate the original data from the signature data, even if you know the algorithm used and the key). Hashing is often used to digitally sign messages in this manner.

Your script can then compare the value it has generated to the **VPSSignature** value in the POST. If they match, the message has not been tampered with. If they do not, then the message may well have been altered and you can act accordingly by declining the transaction and notifying us immediately!

If the Hash values match, you should store the **TxAuthNo** field from the notification POST in your database alongside the VendorTxCode, VPSTxId and SecurityKey. The TxAuthNo field DOES NOT contain the actual Bank Authorisation Code because it is not unique (although we do store this in our system for you), but contains instead a unique reference number to that authorisation that we call the **VPSAuthCode**. This is the transaction ID sent to the bank during settlement (we cannot use your VendorTxCode because it is too long and might contain invalid characters) so the bank will use this value to refer to your transaction if they need to contact you about it.

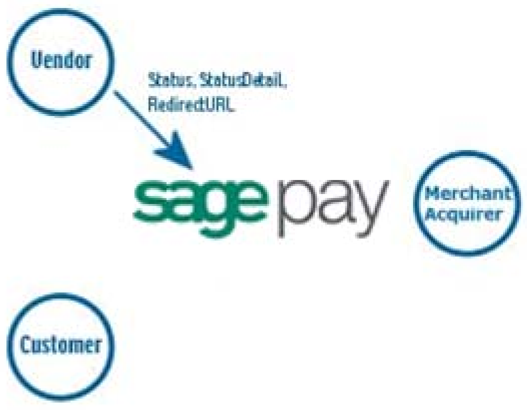
As mentioned above, your Notification script must reply to the Notification POST in all circumstances, irrespective of the Status of the message, otherwise the Sage Pay transaction monitor will cancel the transaction and keep trying to notify you about it (see “Transaction Monitor” later in this document).

If the Sage Pay Server system cannot contact your Notification URL on the first attempt, it should try to notify you a further 9 times, at approximately 1 second intervals in case your server is busy. If your Notification URL still cannot be contacted after 10 seconds (i.e. after the 10th attempt), the transaction is timed out by the Transaction Monitor (see The Transaction Monitor section later in this document) and never settled, so your customer is not charged*. After the 10th unsuccessful attempt to notify you of the status of the transaction, the Sage Pay system continues to attempt to send Notification Posts to your Notification URL with a Status of ABORT to inform you of the cancelled transaction.

***Important note for PayPal transactions**: Non-PayPal transactions are timed out by the Transaction Monitor and never settled if our Server cannot contact your Notification URL, however as all PayPal transactions are settled instantly (once the shopper has returned to the Sage Pay Order Confirmation Page), if there is a problem with Sage Pay notifying you of the transaction, it is possible that your PayPal Admin area will display a transaction as successful, but the *My Sage Pay* Admin area will state the transaction has failed. We strongly recommend you to log into your PayPal Admin area regularly, and cancel any transactions which are displayed as failed in the *My Sage Pay* Admin area (so that your PayPal Admin area and *My Sage Pay* Admin area coincide).

Step 10: You reply to the Notification POST

Your notification script should reply to the Sage Pay Server POST with three fields:



Status, which indicates if you wish to accept the transaction notification, **StatusDetail** to hold human readable reasons for accepting the transaction or otherwise, and **RedirectURL**, which is the completion page on your own site to which the user should be redirected by the Sage Pay Server.

You can reply with a status of either **OK**, **INVALID** or **ERROR**.

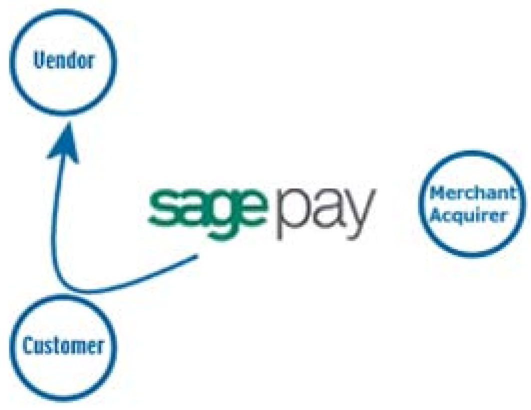
ERROR should be used very rarely, and should ONLY be sent if something unforeseen has happened on your server or database (if you receive a notification POST for a transaction you cannot find, for instance). A Status of **INVALID** should be sent if you are not happy with the contents of the POST, either because the MD5 hash signatures did not match or you do not wish to proceed with the order. **OK** should be sent if you are happy with the notification and wish to proceed to charge the customer.

Regardless of status, the **RedirectURL** must be sent that contains a valid Fully Qualified URL to the final completion page on your site (i.e. an address starting http:// or https://). When the Status is OK, this is normally a page saying "Thank you for your order, reference 123456, please visit us again." In the case of INVALID or ERROR, the RedirectURL will normally point to an error page, often with a support telephone number.

If the Status field you send back to our Server is anything other than OK then the transaction is never settled with the bank (see Step 12) and the customer is NOT charged for the goods or services*. In these circumstances you should not send goods out to the customer.

***Important Note for PayPal transactions**: Non-PayPal transactions are never settled if the Status field you send back to our Server is anything other than OK, however as all PayPal transactions are settled instantly (once the shopper has returned to the Sage Pay Order Confirmation Page), if there is a problem with Sage Pay notifying you of the transaction, it is possible that your PayPal Admin area will display a transaction as successful, but the *My Sage Pay* Admin area will state the transaction has failed. If Sage Pay are unable to contact your Notification URL, or we do not receive a response from you to the Notification Post, you will need to log into your PayPal Admin area to cancel the transaction (so that your PayPal Admin area coincides with your *My Sage Pay* Admin area).

Step 11: Sage Pay Server Redirects the Customer to your site.

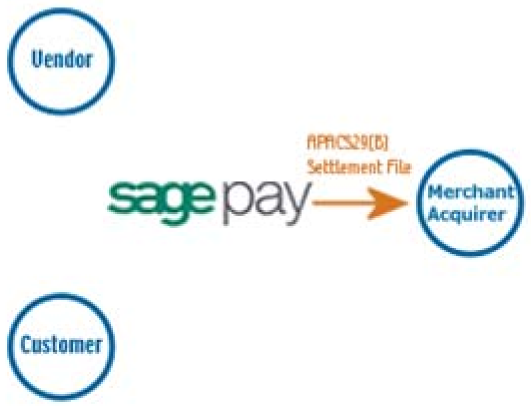


The Sage Pay Server sends a simple HTML page to the customer's browser that redirects them to the page on your server pointed to by the **RedirectURL** field sent in step 10 above.

As before, the customer is unaware of the background POST and response process in the previous two steps. From their perspective they simply clicked "Proceed", got a message saying "Authorising please wait…" and then found themselves back on your website on a completion page of some description.

The real time processing of the transaction by Sage Pay is now complete

Step 12: Sage Pay sends a daily Batch File to confirm payments.



Once per day, from 2:00am, the Sage Pay system batches all authorised transactions for each acquirer and creates a bank specific settlement file.

Transactions for ALL merchants who use the same merchant acquirer are included in this file. Every transaction (excluding PayPal transactions*) from 00:00:00am until 11:59:59pm on the previous day is included in the files.

They are uploaded directly to the acquiring banks on a private secure connection. This process requires no feedback or input from you or your site.

If the file does not transmit correctly, the system tries a further nine times at 10-minute intervals. If all 10 attempts fail the transactions for that bank are rescheduled for inclusion in the following day's batch instead. Sage Pay monitor this process each day to ensure the files have been sent, and if not, the support department correct the problem during the day to ensure the file is sent correctly that evening (or normally resubmit the file manually the same day to ensure funds are available to all vendors more expediently).

The acquirers send summary information back to Sage Pay to confirm receipt of the file, then more detailed information later about rejections or errors. If transactions are rejected, we correct any errors and resubmit them for you. Your bank will contact you directly if there are any non-formatting related problems with the transactions.

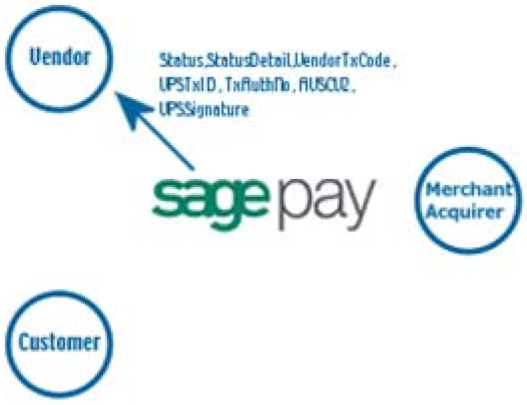
***Important Note for PayPal transactions**: PayPal transactions are settled by PayPal. The funds from your customers' PayPal payments are deposited into your PayPal Business account immediately. You can then withdraw or transfer the funds electronically into your specified bank account. Although PayPal transactions will now be included in the Settlement Reports displayed within *My Sage Pay*, as PayPal transactions are <u>not</u> settled by Sage Pay, we recommend you to log into your PayPal Admin area to obtain a report of your PayPal transactions.

The Transaction Monitor

If the Sage Pay Server system is unable to inform your web site of the success or failure (see step 9 above), even after multiple attempts, then the transaction is placed in suspension.

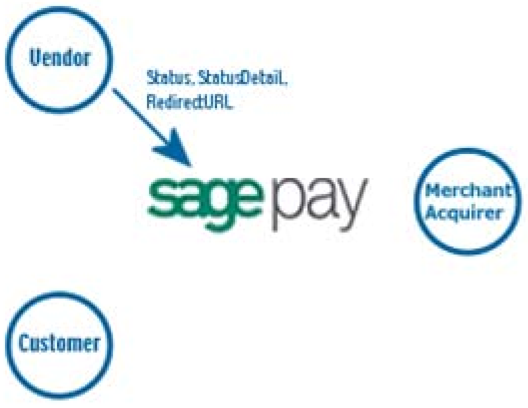
Likewise, if a customer reaches the Sage Pay payment pages, changes their mind but does not click Cancel, choosing instead to simply close their browser, or go elsewhere, then the transaction is stuck in limbo.

Sage Pay guarantee to inform you about the success or failure of every transaction you send to us, so transactions such as those mentioned above have to be dealt with.


Vendor
Status,StatusDetail,VendorTxCode, VPSTxID, TxAuthNo, AVSCV2, VPSSignature
sagepay
Merchant Acquirer
Customer

The Sage Pay transaction monitor is a service that runs within our secure private network, monitoring the database, looking for unfinished transactions that are over 15 minutes old. When it finds one, it changes the status to CANCELLED and sends a POST to your Notification URL (in exactly the same manner as in Step 9 above) with a Status of **ABORT**.

Because the process is identical to a normal Notification POST, your script should reply as it would to any ABORT notification POST (see step 10), with a **Status** and a **RedirectURL**. Because the user is no longer online, no redirection message will be sent to the client browser, but our server operates on the principle that if it receives a reply from your server, then your site must be aware that the transaction has been cancelled, so goods will not be shipped and the user will not be charged.



If the Sage Pay Server system cannot contact your Notification URL on the first attempt, it should try to notify you a further 9 times, at approximately 1 second intervals in case your server is busy. If your Notification URL still cannot be contacted after 10 seconds (i.e. after the 10th attempt), the transaction is timed out by the Transaction Monitor and never settled, so your customer is not charged. After the 10th unsuccessful attempt to notify you of the status of the transaction, the Sage Pay system continues to attempt to send Notification Posts to your Notification URL with a Status of ABORT to inform you of the cancelled transaction.

If your site does not reply to the ABORT Post, the service continues to try and notify you at the following intervals:

5 attempts at 5 minutes intervals

15 attempts at 15 minute intervals

13 attempts at 1 hour intervals

1 attempt per day for the next 30 days

During this period, the transaction is still classed as 'active', and therefore will not appear within the *My Sage Pay* tables (where archived transactions are listed). If your Notification URL still cannot be contacted after 30 days, the monitor stops trying, and the transaction will be archived and listed within the failed transactions tables displayed in *My Sage Pay*.

Integrating with the Server payment method

Linking your Web site to Sage Pay using the Server integration method involves creating two scripts (or modifying the examples provided in the integration kits), one to register the transaction with our servers, process the response we send back and redirect the customer across to us; and the other to handle the notification call-back from our servers, process the message and respond with a status and completion page URL.

Stage 1
The Sage Pay Simulator system is the starting point for your integration. This user-friendly expert system on our test environment analyses the messages that your site sends to us, reports any errors therein, and simulates all possible responses from the real Sage Pay live Server.

The Sage Pay Simulator can be configured on the following URL:
https://test.sagepay.com/Simulator

Payment transactions should be sent from your scripts to the following URL:
https://test.sagepay.com/Simulator/VSPServerGateway.asp?Service=VendorRegisterTx

Stage 2
Once your site is able to talk to the Sage Pay Simulator and process all possible outcomes, you will be able to move over to the Sage Pay Test Server. This is an exact copy of the live site but without the banks attached and with a simulated 3D-Secure environment. Authorisations on the test server are only simulated, but the user experience is identical to Live, and a version of the *My Sage Page* pages also runs here so you can familiarise yourself with the features available to you.

The *My Sage Pay* admin system for viewing your Test transactions is at:
https://test.sagepay.com/mysagepay

Transactions from your scripts should be sent to the Sage Pay Test Server at:
https://test.sagepay.com/gateway/service/vspserver-register.vsp

Stage3
Once you are happily processing end-to-end transactions on the test server and we can see test payments and refunds going through your account, AND you've completed the online Direct Debit signup, your account on the Live Server is activated for you to start using. You will need to redirect your scripts to send transactions to the live service, send through a Payment using your own credit card, then VOID it through the *My Sage Pay* Admin service so you don't charge yourself. If this works successfully, then you are ready to trade online.

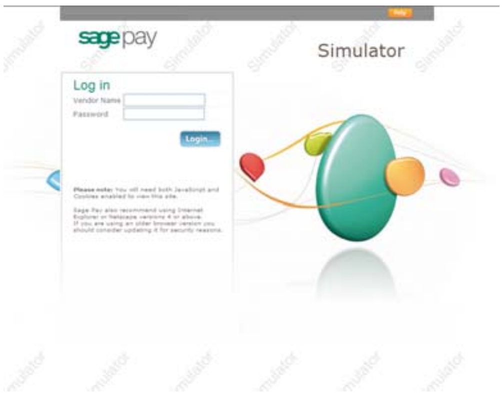
The Live *My Sage Pay* Admin screens are at:
https://live.sagepay.com/mysagepay

Transactions from your scripts should be sent to the Sage Pay Live Server at:
https://live.sagepay.com/gateway/service/vspserver-register.vsp

Stage 1: Integrating with the Sage Pay Simulator

The Sage Pay Simulator is an expert system that emulates the Sage Pay Server system and allows you to develop your site to correctly send and process the messages exchanged between your site and ours. The Simulator will provide more detailed feedback of any errors or issues than the real Sage Pay Server, allowing you to debug and enhance your code.

Log into the Sage Pay Simulator at https://test.sagepay.com/simulator and enter your Vendor Name (as you selected on the Online Registration forms) and the password (also the same as that used on those forms. You can change it in the Simulator if you wish).



Please note: You will need both JavaScript and Cookies enabled to view this site.

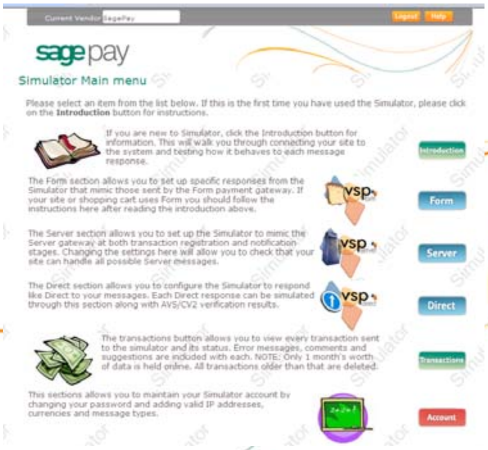
Sage Pay also recommend using Internet Explorer or Netscape versions 4 or above. If you are using an older browser version you should consider updating it for security reasons.

If you wish to test your integration with Sage Pay before you have obtained a Merchant Account, you can do so free of charge with the Sage Pay Simulator. To register for a Simulator account, please visit our website:

https://support/apply/requestsimaccount.aspx

If you already have a Test/Live account and would like a Simulator account, contact our Support Team at support@sagepay.com, stating the Vendor Name of your account.

When you log in to the Sage Pay Simulator you will be presented with the main menu screen. Extensive help is provided in the Simulator (click the context sensitive Help button on each screen for more details) so this document will not cover everything in too much detail, but outlined in subsequent sections are the important steps you should take to get your site talking to the Simulator.



Please select an item from the list below. If this is the first time you have used the Simulator, please click on the **Introduction** button for instructions.

If you are new to Simulator, click the Introduction button for information. This will walk you through connecting your site to the system and testing how it behaves to each message response.

The Form section allows you to set up specific responses from the Simulator that mimic those sent by the Form payment gateway. If your site or shopping cart uses Form you should follow the instructions here after reading the introduction above.

The Server section allows you to set up the Simulator to mimic the Server gateway at both transaction registration and notification stages. Changing the settings here will allow you to check that your site can handle all possible Server messages.

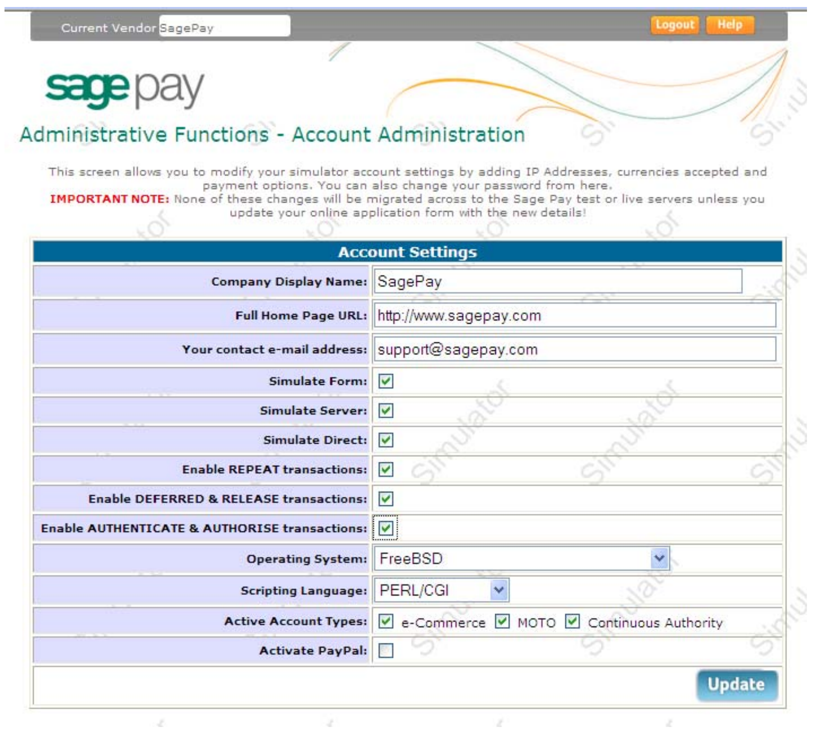
The Direct section allows you to configure the Simulator to respond like Direct to your messages. Each Direct response can be simulated through this section along with AVS/CV2 verification results.

The transactions button allows you to view every transaction sent to the simulator and its status. Error messages, comments and suggestions are included with each. NOTE: Only 1 month's worth of data is held online. All transactions older than that are deleted.

This sections allows you to maintain your Simulator account by changing your password and adding valid IP addresses, currencies and message types.

1: Sage Pay Simulator Account Set up

Click the Account button in the main menu to open the following screen:



Current Vendor SagePay

Administrative Functions - Account Administration

This screen allows you to modify your simulator account settings by adding IP Addresses, currencies accepted and payment options. You can also change your password from here.
IMPORTANT NOTE: None of these changes will be migrated across to the Sage Pay test or live servers unless you update your online application form with the new details!

Account Settings	
Company Display Name:	SagePay
Full Home Page URL:	http://www.sagepay.com
Your contact e-mail address:	support@sagepay.com
Simulate Form:	✓
Simulate Server:	✓
Simulate Direct:	✓
Enable REPEAT transactions:	✓
Enable DEFERRED & RELEASE transactions:	✓
Enable AUTHENTICATE & AUTHORISE transactions:	✓
Operating System:	FreeBSD
Scripting Language:	PERL/CGI
Active Account Types:	✓ e-Commerce ✓ MOTO ✓ Continuous Authority
Activate PayPal:	☐

Update

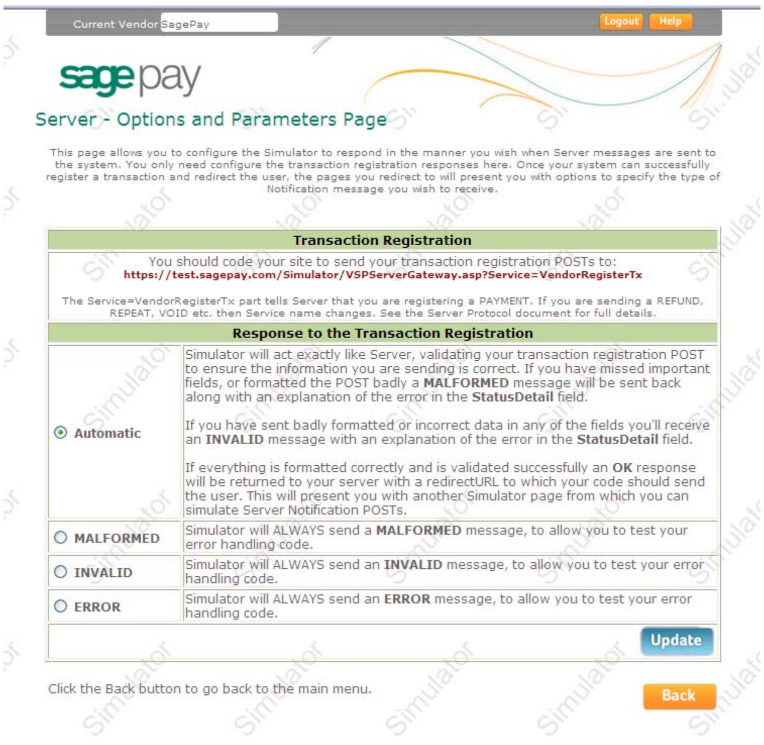
You should ensure that:

- all company details are correct.
- all technical details about web server and platform are correct.
- the “Simulate Server” box is checked.
- all relevant payment types have been set up.
- you have at least one payment currency set up (usually GBP unless your site accepts multi-currency transactions).
- the IP addresses of your servers are listed.

Add and/or correct any entries and click the Update button to save any changes. Back takes you back to the main menu.

2: Server Integration method Set up

Click the Server button in the main menu to open the Server options page.



Server - Options and Parameters Page

This page allows you to configure the Simulator to respond in the manner you wish when Server messages are sent to the system. You only need configure the transaction registration responses here. Once your system can successfully register a transaction and redirect the user, the pages you redirect to will present you with options to specify the type of Notification message you wish to receive.

Transaction Registration

You should code your site to send your transaction registration POSTs to:
https://test.sagepay.com/Simulator/VSPServerGateway.asp?Service=VendorRegisterTx

The Service=VendorRegisterTx part tells Server that you are registering a PAYMENT. If you are sending a REFUND, REPEAT, VOID etc. then Service name changes. See the Server Protocol document for full details.

Response to the Transaction Registration

[✓ **Automatic**]	Simulator will act exactly like Server, validating your transaction registration POST to ensure the information you are sending is correct. If you have missed important fields, or formatted the POST badly a **MALFORMED** message will be sent back along with an explanation of the error in the **StatusDetail** field. If you have sent badly formatted or incorrect data in any of the fields you'll receive an **INVALID** message with an explanation of the error in the **StatusDetail** field. If everything is formatted correctly and is validated successfully an **OK** response will be returned to your server with a redirectURL to which your code should send the user. This will present you with another Simulator page from which you can simulate Server Notification POSTs.
MALFORMED	Simulator will ALWAYS send a **MALFORMED** message, to allow you to test your error handling code.
INVALID	Simulator will ALWAYS send an **INVALID** message, to allow you to test your error handling code.
ERROR	Simulator will ALWAYS send an **ERROR** message, to allow you to test your error handling code.

Click the Back button to go back to the main menu.

This page allows you to define the behaviour of the Sage Pay Simulator when it responds to your initial transaction registrations (Steps 2 and 3 of the payment process described above). By default the system will verify your POST to ensure the contents are correctly formatted and if they are, return a Status of OK, a VPSTxId, a SecurityKey and a NextURL. If your POST is incorrectly formatted or contains bad data, it will respond with a Status of MALFORMED or INVALID and explain what was wrong in the StatusDetail field.

You can use this page to force errors even if your data is okay. This is useful when testing upgrades to your scripts and proofing your error handling routines.

For now you should leave the default setting of Automatic (clicking Update if necessary) then log out of the Simulator.

3: Registering a Payment

If you don't plan to implement the protocol entirely on your own, you should install the most appropriate integration kit or worked example for your platform. These can be obtained from the development area on the Sage Pay website www.sagepay.com/developers.html.

The kits will not quite run out of the box because you have to provide some specific details about your site in the configuration files before a transaction can occur, but they will provide end to end examples of registering the transactions and handling the notification POSTs. Ensure you've completed all configuration in the includes file as detailed in the kit instructions, then locate the Transaction Registration script (normally called transactionRegistration).

This script provides a worked example of how to construct the Transaction Registration POST (see Appendix A section A1 in the attached protocol) and how to read the response that comes back (section A2).

Check that this script is sending transactions to the Sage Pay Simulator (rather than the test or live sites), then execute this page, passing it some dummy transaction data, to send a payment registration to the Simulator. You may wish to modify the script at this stage to echo the results of the POST to the screen, or a file, so you can examine the Status and StatusDetail reply fields to check for errors.

Once your script can successfully register a Payment and you receive a Status of OK, you should ensure your code stores the VPSTxId and SecurityKey alongside your uniquely generated VendorTxCode and the order details in your own database before redirecting the browser to the URL sent by us in the NextURL field.

At this stage it is wise to log in to Sage Pay Simulator in a separate browser window and change the response type to each of the error messages in turn so you can write code in your registration page to handle each error appropriately (by logging the error, informing the user that a problem has occurred, perhaps giving them a phone number to call instead and alerting support staff as appropriate).

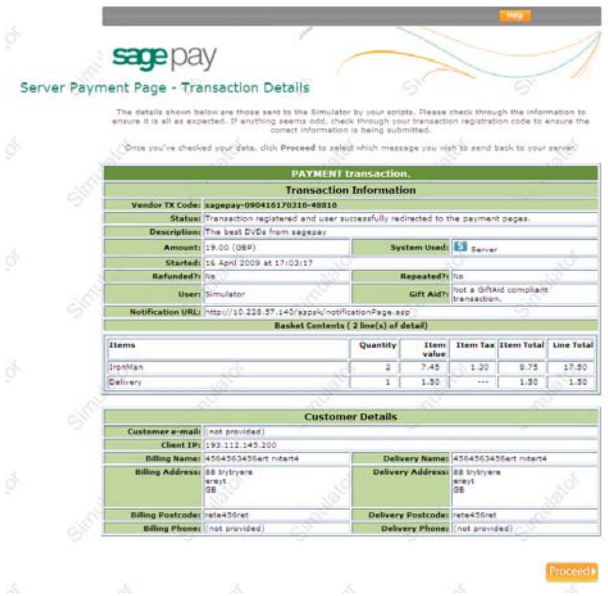
When you are happy that your script can handle all errors, set the Server option setting back to Automatic so your script can redirect the user to the Simulator payment pages.

4: Handling the Server payment Callback

After your site has passed the customer across to the Sage Pay payment pages, they enter their card details and the bank authorise their transaction (an OK response) or fail it (a NOTAUTHED response), or Sage Pay may reverse an authorisation if your fraud screening rules are not met (a REJECTED response). The customer may also change their mind and click Cancel on one of the payment pages (an ABORT response).

Irrespective of the type of feedback the Sage Pay Server needs to send you, the message is always sent to the same script on your server. We refer to this script as the Notification Script and it is pointed to by the contents of the NotificationURL field you sent to us in step 2 of the process above (see the protocol section A1). In your kits this script is normally called notificationPage.

The Simulator will show you the contents of your registration POST, including the NotificationURL, in the screens following the redirection step. The final page will allow you to choose which type of message you wish to send back to your Notification script from our servers.



Server Payment Page - Transaction Details

The details shown below are those sent to the Simulator by your scripts. Please check through the information to ensure it is all as expected. If anything seems odd, check through your transaction registration code to ensure the correct information is being submitted.

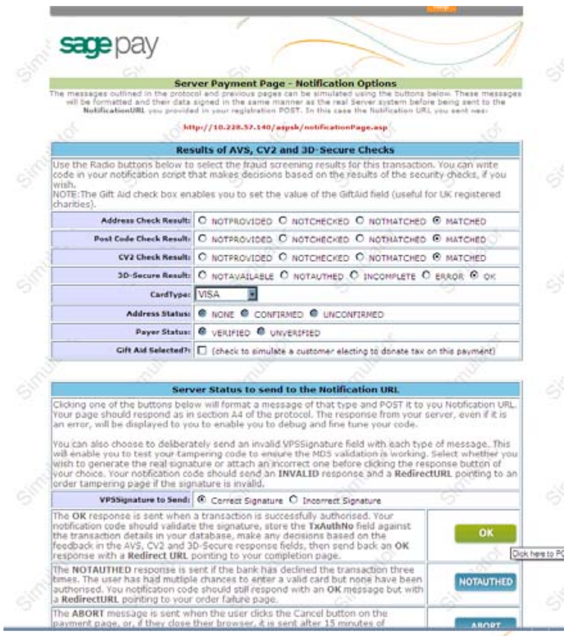
Once you've checked your data, click **Proceed** to select which message you wish to send back to your server.

PAYMENT transaction.			
Transaction Information			
Vendor TX Code:	sagepay-090416170316-48816		
Status:	Transaction registered and user successfully redirected to the payment pages.		
Description:	The best DVDs from sagepay		
Amount:	19.00 (GBP)	System Used:	Server
Started:	16 April 2009 at 17:03:17		
Refunded?:	No	Repeated?:	No
User:	Simulator	Gift Aid?:	Not a GiftAid compliant transaction.
Notification URL:	http://10.228.57.140/aspsk/notificationPage.asp		

Basket Contents (2 line(s) of detail)

Items	Quantity	Item value	Item Tax	Item Total	Line Total
IronMan	2	7.45	1.30	8.75	17.50
Delivery	1	1.50	---	1.50	1.50

Customer Details			
Customer e-mail:	(not provided)		
Client IP:	193.112.145.200		
Billing Name:	4564563456ert ritert4	Delivery Name:	4564563456ert ritert4
Billing Address:	88 trytryere ereyt GB	Delivery Address:	88 trytryere ereyt GB
Billing Postcode:	rete456ret	Delivery Postcode:	rete456ret
Billing Phone:	(not provided)	Delivery Phone:	(not provided)



Server Payment Page - Notification Options

The messages outlined in the protocol and previous pages can be simulated using the buttons below. These messages will be formatted and their data signed in the same manner as the real Server system before being sent to the **NotificationURL** you provided in your registration POST. In this case the Notification URL you sent was:

http://10.228.57.140/aspsk/notificationPage.asp

Results of AVS, CV2 and 3D-Secure Checks	
Use the Radio buttons below to select the fraud screening results for this transaction. You can write code in your notification script that makes decisions based on the results of the security checks, if you wish. NOTE:The Gift Aid check box enables you to set the value of the GiftAid field (useful for UK registered charities).	
Address Check Result:	NOTPROVIDED NOTCHECKED NOTMATCHED [✓ MATCHED]
Post Code Check Result:	NOTPROVIDED NOTCHECKED NOTMATCHED [✓ MATCHED]
CV2 Check Result:	NOTPROVIDED NOTCHECKED NOTMATCHED [✓ MATCHED]
3D-Secure Result:	NOTAVAILABLE NOTAUTHED INCOMPLETE ERROR [✓ OK]
CardType:	VISA
Address Status:	NONE CONFIRMED UNCONFIRMED
Payer Status:	VERIFIED UNVERIFIED
Gift Aid Selected?:	☐ (check to simulate a customer electing to donate tax on this payment)

Server Status to send to the Notification URL	
Clicking one of the buttons below will format a message of that type and POST it to you Notification URL. Your page should respond as in section A4 of the protocol. The response from your server, even if it is an error, will be displayed to you to enable you to debug and fine tune your code. You can also choose to deliberately send an invalid VPSSignature field with each type of message. This will enable you to test your tampering code to ensure the MD5 validation is working. Select whether you wish to generate the real signature or attach an incorrect one before clicking the response button of your choice. Your notification code should send an **INVALID** response and a **RedirectURL** pointing to an order tampering page if the signature is invalid.	
VPSSignature to Send:	[✓ Correct Signature] Incorrect Signature
The **OK** response is sent when a transaction is successfully authorised. Your notification code should validate the signature, store the **TxAuthNo** field against the transaction details in your database, make any decisions based on the feedback in the AVS, CV2 and 3D-Secure response fields, then send back an **OK** response with a **Redirect URL** pointing to your completion page.	OK
The **NOTAUTHED** response is sent if the bank has declined the transaction three times. The user has had mutiple chances to enter a valid card but none have been authorised. You notification code should still respond with an **OK** message but with a **RedirectURL** pointing to your order failure page.	NOTAUTHED
The **ABORT** message is sent when the user clicks the Cancel button on the payment page, or, if they close their browser, it is sent after 15 minutes of	ABORT

You can choose not only the type of message sent back, but also the results of the additional fraud checks. For now, leave everything set as default (MATCHED and OK) and click the OK button to send back a positive response (mimicking a successfully authorised transaction).

This message (see Steps 9 and 10 in the payment process above, and section A3 in the Appendix) is POSTed to your Notification script, which should process it and reply with a Status and a RedirectURL (see Appendix A4).

Processing the Notification POST is slightly more complex because you need to validate the MD5 digital signature that is attached to the message to ensure it has not been tampered with and genuinely comes from Sage Pay. The example scripts in the Integration Kits show you how to do this, but the steps are:

1. Split the fields out of the POST to obtain the authorisation result, transaction ids and VPSSignature value.
2. Use the transaction ids to look up the order in your database and retrieve the SecurityKey passed to you during transaction registration.
3. Rebuild the Notification POST using the contents of your database and the POST itself in the order specified in the protocol (see A3).
4. Pass that data through a MD5 hashing algorithm (provided either as part of your scripting language or as part of our kits) to generate a hash value.
5. Compare that hash value to the contents of the VPSSignature field. If they match, the data has not been tampered with. If they do not, either the data has been modified or there is a mismatch between your data and ours, and the transaction should be cancelled.

If the signatures match, your Notification Script should respond with a Status of OK and a RedirectURL pointing to either an order completion page (if the Status was OK) or an appropriate order failure page (if the Status was NOTAUTHED or ERROR). You may wish ABORT messages to redirect the customer to a page providing them with alternative methods of payment, or asking them why they chose to cancel.

If the signatures do not match, you should check that your code is rebuilding the message correctly, and if you are sure that it is, all such messages should be responded to with an INVALID and a RedirectURL pointing the user to a failure page.

If you cannot find the transaction we are notifying you about, you should return an ERROR Status and a RedirectURL pointing to an error page.

The Sage Pay Simulator will show you the response returned by your notification script. If the page throws an error, this error will be displayed to enable you to debug it. If it responds correctly, with a Status and a RedirectURL, you will be shown a Redirect button that will send the browser to your completion page.

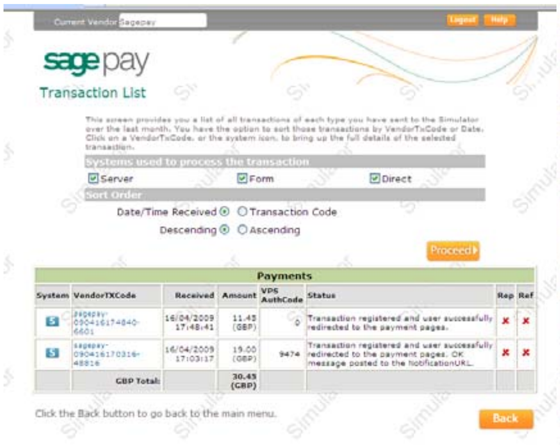
Important Note: Your Notification URL should ONLY respond with a Status field, a RedirectURL field and optionally a StatusDetail field. No other HTML, headers, comments or text should be included either before or after these fields. The Sage Pay Server will treat all such text as an error and fail the transaction!

For OK responses, you should store the TxAuthNo field against the other fields in your database for this transaction. This reference number uniquely identifies the transaction with your acquiring bank and they are likely to quote you this value if there are issues with it.

You should use the Sage Pay Simulator to send each type of message (OK, ABORT, NOTAUTHED, REJECTED and ERROR) to your notification page to check that all message types are handled correctly. You may also wish to add code that stores the 3DSecureStatus and CAVV fields, if you plan to use Visa and Mastercard's extended fraud checking systems (Verified by Visa, VbV, and Mastercard Secure Code, MSC), and specific code that stores or reacts to the AVS and CV2 results (additional card security checks). See our Fraud Screening document for more information about these systems.

5: Examining your transactions

The Sage Pay Simulator keeps the last month's worth of simulated transactions online for you to examine at your leisure. Using the Transactions button you can view everything you've sent us to ensure the data is as you expected.



Current Vendor Sagepay

Transaction List

This screen provides you a list of all transactions of each type you have sent to the Simulator over the last month. You have the option to sort those transactions by VendorTxCode or Date. Click on a VendorTxCode, or the system icon, to bring up the full details of the selected transaction.

Systems used to process the transaction

[✓ Server] [✓ Form] [✓ Direct]

Sort Order

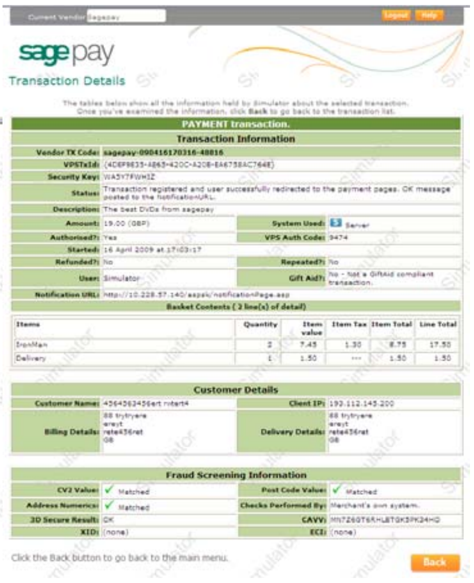
Date/Time Received [✓ ●] ○ Transaction Code

Descending [✓ ●] ○ Ascending

Payments

System	VendorTXCode	Received	Amount	VPS AuthCode	Status	Rep	Ref
S	sagepay-090416174840-6601	16/04/2009 17:48:41	11.45 (GBP)	0	Transaction registered and user successfully redirected to the payment pages.	✗	✗
S	sagepay-090416170316-48816	16/04/2009 17:03:17	19.00 (GBP)	9474	Transaction registered and user successfully redirected to the payment pages. OK message posted to the NotificationURL.	✗	✗
	GBP Total:		30.45 (GBP)				

Click the Back button to go back to the main menu.



Current Vendor sagepay

Transaction Details

The tables below show all the information held by Simulator about the selected transaction. Once you've examined the information, click **Back** to go back to the transaction list.

PAYMENT transaction.

Transaction Information

Vendor TX Code:	sagepay-090416170316-48816		
VPSTxId:	{4D8F9835-A863-420C-A208-EA6758AC7648}		
Security Key:	WA5Y7FWHIZ		
Status:	Transaction registered and user successfully redirected to the payment pages. OK message posted to the NotificationURL.		
Description:	The best DVDs from sagepay		
Amount:	19.00 (GBP)	System Used:	S Server
Authorised?:	Yes	VPS Auth Code:	9474
Started:	16 April 2009 at 17:03:17		
Refunded?:	No	Repeated?:	No
User:	Simulator	Gift Aid?:	No - Not a GiftAid compliant transaction.
Notification URL:	http://10.228.57.140/aspski/notificationPage.asp		

Basket Contents (2 line(s) of detail)

Items	Quantity	Item value	Item Tax	Item Total	Line Total
IronMan	2	7.45	1.30	8.75	17.50
Delivery	1	1.50	---	1.50	1.50

Customer Details

Customer Name:	4364363456ert rvtert4	Client IP:	193.112.145.200
Billing Details:	88 trytryere ereyt rete436ret GB	Delivery Details:	88 trytryere ereyt rete436ret GB

Fraud Screening Information

CV2 Value:	✓ Matched	Post Code Value:	✓ Matched
Address Numerics:	✓ Matched	Checks Performed By:	Merchant's own system.
3D Secure Result:	OK	CAVV:	MN7Z60T6RHLBTGK3PK24HO
XID:	(none)	ECI:	(none)

Click the Back button to go back to the main menu.

You can also see from this screen which transactions have been subsequently refunded or used as the basis for repeat payments.

Once your site can initiate transactions AND handle the callbacks, then you've completed the basic Sage Pay Server integration and can move on to testing your site against the real Sage Pay Servers, firstly on the Test Server (see the next main section). If, however, you wish to link in additional processes, such as Refunds or Repeats, or the ability to Release or Abort Deferred transactions, you should continue with step 6 below.

6: Additional Transaction Types

Sage Pay supports a number of methods of registering a transaction and completing the payment.

DEFERRED transactions

By default a PAYMENT transaction type is used in your scripts to gain an authorisation from the bank, then settle that transaction early the following morning, committing the funds to be taken from your customer's card.

In some cases you may not wish to take the funds from the card immediately, but merely place a "shadow" on their card to ensure they cannot subsequently spend those funds elsewhere, and then only take the money when you are ready to ship the goods. This type of transaction is called a **DEFERRED** transaction and is registered in exactly the same way as a normal PAYMENT. You just need to change your script to send a TxType of DEFERRED when you register the transaction (protocol A1) instead of PAYMENT.

DEFERRED transactions are NOT sent to the bank for completion the following morning. In fact, they are not sent at all until you RELEASE them, either by sending a **RELEASE** message to our servers from yours (see the "Server and Direct Shared Protocols" document for details on how to send this message) or by logging into the *My Sage Pay* Admin interface, finding the transaction and clicking the Release button.

You can release ONLY ONCE and ONLY an amount up to and including the amount of the original DEFERRED transaction.

If you are unable to fulfil the order, you can also **ABORT** deferred transactions in a similar manner and the customer will never be charged.

DEFERRED transactions work well in situations where it is only a matter of days between the customer ordering and you being ready to ship. Ideally all DEFERRED transaction should be released within 6 days (according to card scheme rules). After that the shadow may disappear from the card before you settle the transaction, and you will have no guarantee that you'll receive the funds if the user has maxed out their card in the mean time. If you regularly require longer than 6 days to fulfil orders, you should consider using AUTHENTICATE and AUTHORISE instead of DEFERRED payments (see below).

DEFERRED transactions remain available for RELEASE for up to 30 days. After that time they are automatically ABORTed by the Sage Pay systems.

Using Deferred/Release with PayPal transactions

You may find that you cannot RELEASE certain DEFERRED PayPal transactions taken through Sage Pay.

As part of our DEFERRED transaction type, we send an ORDER message to PayPal and the message returned is successful. When you attempt to RELEASE a DEFERRED transaction via the Sage Pay system, Sage Pay send a CAPTURE request.

Please note: Unlike a normal Sage Pay DEFERRED transaction, no shadow is placed on the customer's card for a PAYPAL DEFERRED transaction.

A successful authorisation for a DEFERRED transaction only confirms the availability of funds and does not place any funds on hold.

Sage Pay recommend that you do NOT ship the goods until the RELEASE/CAPTURE has been successful.

Whilst PayPal applies best efforts to capture funds there is a possibility that funds will not be available at that time.

REPEAT payments

If you have already successfully authorised a customer's card using a PAYMENT, a released DEFERRED or an AUTHORISE (see below) you can charge an additional amount to that card using the REPEAT transaction type, without the need to store the card details yourself.

If you wish to regularly REPEAT payments, for example for monthly subscriptions, you should ensure you have a "Continuous Authority" merchant number from your bank (please contact your acquiring bank for further details), but ad-hoc REPEATs do not require them. **REPEAT payments cannot be 3D-Secured, or have CV2 checks performed on them** (since Sage Pay are not allowed to store these values) so you are better to make use of Authenticate and Authorise if you need to vary the transaction amount on a regular basis.

At present, you **cannot** **REPEAT** a PayPal transaction.

AUTHENTICATE and AUTHORISE

The AUTHENTICATE and AUTHORISE methods are specifically for use by merchants who are either (i) unable to fulfil the majority of orders in less than 6 days (or sometimes need to fulfil them after 30 days) or (ii) do not know the exact amount of the transaction at the time the order is placed (for example, items shipped priced by weight, or items affected by foreign exchange rates).

Unlike normal PAYMENT or DEFERRED transactions, AUTHENTICATE transactions do not obtain an authorisation at the time the order is placed. Instead the card and card holder are validated using the 3D-Secure mechanism provided by the card-schemes and card issuing banks.

Your site will register your transaction with a TxType of AUTHENTICATE. The Sage Pay Server will contact the 3D-Secure directories to check if the card is part of the scheme. If it is not, then the card details are simply held safely at Sage Pay and our servers will reply with a Status of **REGISTERED** (this also happens if you do not have 3D-Secure active on your account or have used the Apply3DSecure flag to turn it off for that transaction).

NB: For PayPal transactions, you can use the Authenticate and Authorise Payment Type but the transaction will only ever be **REGISTERED** (because the transaction will never be 3D Secured). Similarly to Releasing a Deferred transaction, we recommend you to Authorise the transaction via the *My Sage Pay* Admin area when you are ready to ship the goods and take the funds.

If, however, the card *is* part of the 3D-Secure scheme, the Sage Pay server will redirect the customer to their card issuing bank for authentication (just like a normal 3D-Secure payment, see steps 5 and 6 in the Payment Process above).

If the customer has not passed authentication, your rule base is consulted to check if they can proceed for authorisation anyway. If not the Sage Pay Servers replies with a Status of **REJECTED**. If the customer failed authentication but can proceed, the Sage Pay Server replies with a **REGISTERED** Status. If the user passed authentication with their bank and a CAVV/UCAF value is returned, the Sage Pay Server sends a Status of **AUTENTICATED** and a **CAVV** value for you to store if you wish.

In all cases, the customer's card is never authorised. There are no shadows placed on their account and your acquiring bank is not contacted. The customer's card details and their associated authentication status are simply held at Sage Pay for up to 90 days (a limit set by the card schemes, 30 days for International Maestro cards) awaiting an **AUTHORISE** or **CANCEL** request from your site (see the "Server and Direct Shared Protocols" document for details of these messages).

To charge the customer when you are ready to fulfil the order, your site will need to send an **AUTHORISE** request. You can Authorise any amount up to 115% of the value of the original Authentication, and use any number of Authorise requests against an original Authentication so long as the total value of those authorisations does not exceed the 115% limit, and the requests are inside the 90 days limit. This is the stage at which your acquiring bank is contacted for an auth code. AVS/CV2 checks are performed at this stage and rules applied as normal. This allows you greater flexibility for partial shipments or variable purchase values. If the AUTHENTICATE transaction was AUTHENTICATED (as opposed to simply REGISTERED) all authorisations will be fully 3D-Secured, so you will still receive the fraud liability shift.

When you have completed all your Authorisations, or if you do not wish to take any, you can send a **CANCEL** message to our Server to archive away the Authentication and prevent any further Authorisations being made against the card. This happens automatically after 90 days.

Both AUTHORISE and CANCEL operations can also be performed within the *My Sage Pay* Admin area.

REFUNDs and VOIDs

Once a PAYMENT, AUTHORISE or REPEAT transaction has been authorised, or a DEFERRED transaction has been RELEASEd, it will be settled with the acquiring bank early the next morning and the funds will be moved from the customer's card account, across to your merchant account. The bank will charge you for this process, the exact amount depending on the type of card and the details of your merchant agreement.

If you wish to cancel that payment before it is settled with the bank the following morning, you can send a **VOID** message to our servers to prevent the transaction ever being settled (see the "Server and Direct Shared Protocols" document for more detail), thus saving you your transaction charges and the customer from ever being charged. You can also VOID transactions through the *My Sage Pay* Admin interface. VOIDed transactions can NEVER be reactivated though, so use this functionality carefully.

Once a transaction has been settled, however, you can no longer VOID it. If you wish to return funds to the customer you need to send a **REFUND** message to our servers, or use the *My Sage Pay* Admin screens to do the same.

You can REFUND any amount up to the value of the original transaction. You can even send multiple refunds for the same transaction so long as the total value of those refunds does not exceed the value of the original transaction. Again, the REFUND protocol can be found in the "Server and Direct Shared Protocols" document.

You **cannot** **VOID** a PayPal transaction, but you are able to **REFUND** a PayPal transaction.

The Sage Pay Simulator and Additional Transaction Types

The Sage Pay Simulator can handle all the additional transaction types discussed above. It will accept PAYMENT, AUTHENTICATE and DEFERRED transactions at the registration stage, plus it has services that emulate those of the real server when you send REFUND, RELEASE, ABORT, REPEAT, AUTHORISE, CANCEL and VOID messages to it.

The additional transaction types, however, **do not** have a user configurable interface associated with them. By default they are all set to Automatic mode, so they will respond with an OK unless the data you send would generate a MALFORMED or INVALID response.

For information regarding registering additional transaction types using HTTPS POSTS, please refer to the Server and Direct Shared Protocols Guide, which can be obtained from the developer's area on our website:

www.sagepay.com/developers.html

Stage 2: Testing on the Test Server

If your site works correctly against the Sage Pay Simulator then this is normally a very quick step. The Test Server is an exact copy of the Live System but without the banks attached. This means you get a true user experience but without the fear of any money being taken from your cards during testing.

In order to test on the Test Server, however, you need a Test Server account to be set up for you by the Sage Pay Support team. These accounts can **only** be set up once you have completed all sections of the Online Registration forms (https://support.sagepay.com/apply/) including the Merchant Account section. Often when applying to trade online it takes a while for the Merchant Account to be assigned by your acquirer, so you may wish to ensure that you set those wheels in motion before you begin your integration with Sage Pay, to ensure things don't bottleneck at this stage.

The Support Team will set up an account for you on the Test Server under the same Vendor Name as your online application form and Simulator account. You will, however, be issued with different passwords for security purposes. The Support Team will let you know how to retrieve those passwords and from there how to use the *My Sage Pay* Admin screens to look at your transactions.

To link your site to the Test Server, you need only to change your transaction registration script to send the message to the Test Server URL for the Server integrated payment method rather than the Simulator. In many kits this is done simply by change this Test Server flag in the configuration scripts to 1. If you've been developing your own scripts, then the Test Site URL for payment registration is:

https://test.sagepay.com/gateway/service/vspserver-register.vsp
(for other transaction types, the final server-register.vsp section would be changed to refund.vsp, release.vsp, void.vsp etc.)

When your site redirects the customer you will find yourself on the real Sage Pay payment pages rather than the Simulator.

You will always receive an OK message and an Authorisation Code from the test server if you are using one of the test cards listed below. All other valid card numbers will be declined, allowing you to test your failure pages. If you do not use the correct Address, Post Code and CV2 digits, the transaction will still authorise, but you will receive NOTMATCHED messages in the AVS/CV2 checks, allowing you to test your rule-bases and fraud specific code.

Any cardholder name and start/expiry dates will be accepted for these cards so long as the dates are valid and the card not expired.

Card Type	Card Number	Issue	CV2	Address	PostCode
Visa Credit	4929000000006		123	88	412
MasterCard Credit	5404000000000001		123	88	412
Visa Debit / Delta	4462000000000003		123	88	412
Solo	6334900000000005	1	123	88	412
UK Maestro	5641820000000005	01	123	88	412
American Express	374200000000004		123	88	412
Visa Electron	4917300000000008		123	88	412
JCB	3569990000000009		123	88	412
Diner's Club	36000000000008		123	88	412
Laser (LASER)	6304990000000000044		123	88	412

If you have 3D-Secure set up on your test account, you can use the *My Sage Pay* Admin interface to switch on the checks at this stage to test 3D-Secure.

This simulation is more advanced than the Sage Pay Simulator process because it creates real 3D-secure messages. It does not talk to the Visa and MasterCard systems though, so no live authentications can occur.

At the Simulated Authentication screens, to successfully authenticate the transaction, enter "password" (without the quotes) into the password box. Any other phrase will fail the authentication, allowing you to test your rules and 3D-Secure response handling. We'll be extending this to allow you to simulate all 3D-Secure responses.

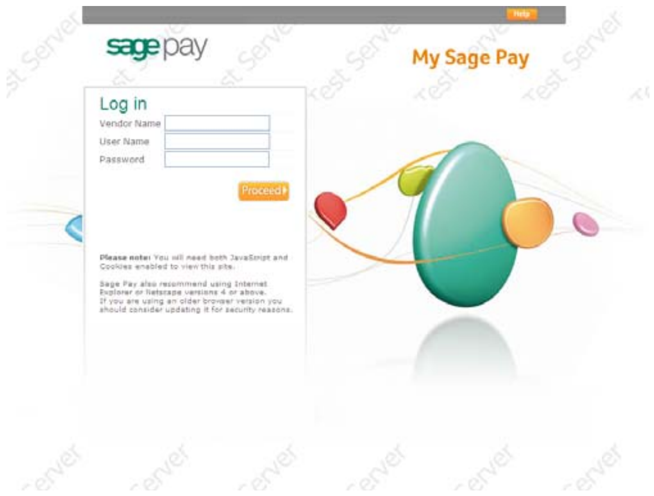
The process will then continue as per the Live Servers. Only the authorisation stage is simulated.

Once you've checked you can process an end-to-end transaction and tested any additional transaction types you have set up (such a Refunds and Releases) then you are almost ready to go live. Before doing so, however, you should log in to the *My Sage Pay* Admin system on the test servers to view your transactions and familiarise yourself with the interface.

The Test Server *My Sage Pay* interface

A Test Server version of the *My Sage Pay* Admin system is available to you whilst using your test account to view your transactions, refund payments, release deferred payments, void transactions etc. You should familiarise yourself with this system on the Test Server before you go live so you know how to use the system on the Live Servers.

The Test Server *My Sage Pay* can be found at:
https://test.sagepay.com/mysagepay

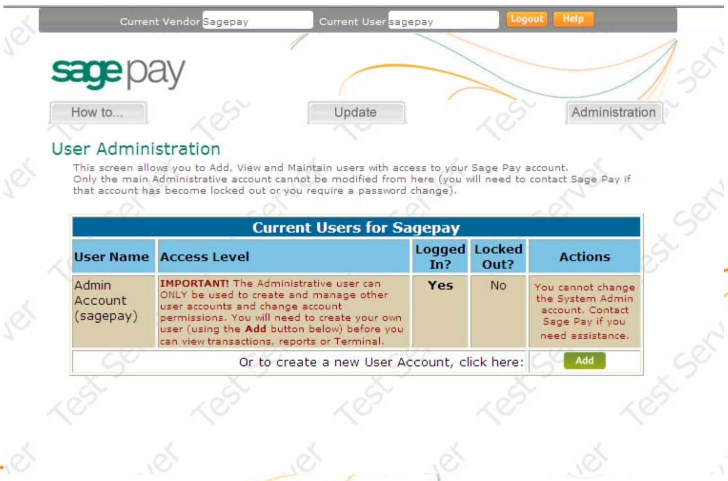


Please note: You will need both JavaScript and Cookies enabled to view this site.

Sage Pay also recommend using Internet Explorer or Netscape versions 4 or above.
If you are using an older browser version you should consider updating it for security reasons.

When you log in to the *My Sage Pay* Admin screens you will be asked for a **Vendor Name**, a **User Name** and a **Password**. The first time you log in you will need to do so as your system Administrator:

- In the **Vendor Name** box, enter your Vendor Name, as selected in your Online Registration screens and used throughout the development as your unique merchant identifier.
- In the **User Name** box, enter the Vendor Name again.
- In the **Password** box, enter the *My Sage Pay* Admin password as supplied to you by Sage Pay when your test account was set up.
- Click **Login**.



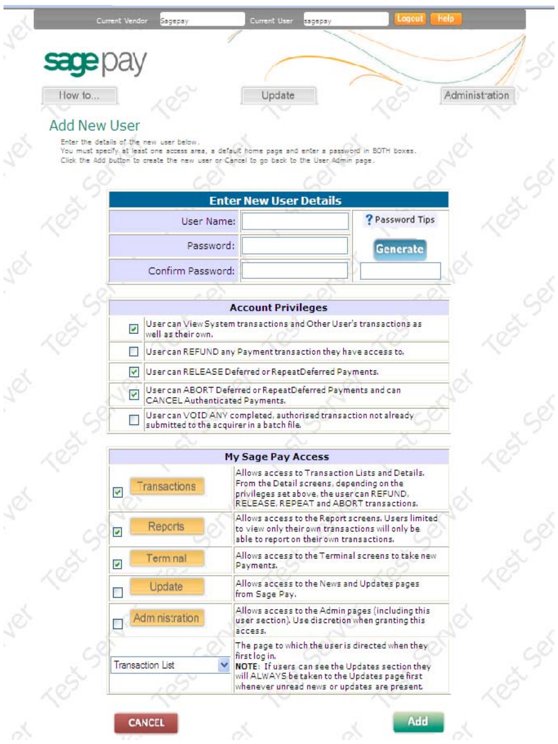
User Administration

This screen allows you to Add, View and Maintain users with access to your Sage Pay account.
Only the main Administrative account cannot be modified from here (you will need to contact Sage Pay if that account has become locked out or you require a password change).

Current Users for Sagepay

User Name	Access Level	Logged In?	Locked Out?	Actions
Admin Account (sagepay)	**IMPORTANT!** The Administrative user can ONLY be used to create and manage other user accounts and change account permissions. You will need to create your own user (using the **Add** button below) before you can view transactions, reports or Terminal.	Yes	No	You cannot change the System Admin account. Contact Sage Pay if you need assistance.
			Or to create a new User Account, click here:	Add

The administrator can ONLY create user accounts, unlock other accounts and change account parameters. You cannot, whilst logged in as administrator, view your transactions or take payments through the online terminal.



Add New User

Enter the details of the new user below.
You must specify at least one access area, a default home page and enter a password in BOTH boxes.
Click the Add button to create the new user or Cancel to go back to the User Admin page.

Enter New User Details

User Name:		? Password Tips
Password:		Generate
Confirm Password:		

Account Privileges

✓	User can View System transactions and Other User's transactions as well as their own.
☐	User can REFUND any Payment transaction they have access to.
✓	User can RELEASE Deferred or RepeatDeferred Payments.
✓	User can ABORT Deferred or RepeatDeferred Payments and can CANCEL Authenticated Payments.
☐	User can VOID ANY completed, authorised transaction not already submitted to the acquirer in a batch file.

My Sage Pay Access

✓ Transactions	Allows access to Transaction Lists and Details. From the Detail screens, depending on the privileges set above, the user can REFUND, RELEASE, REPEAT and ABORT transactions.
✓ Reports	Allows access to the Report screens. Users limited to view only their own transactions will only be able to report on their own transactions.
✓ Terminal	Allows access to the Terminal screens to take new Payments.
☐ Update	Allows access to the News and Updates pages from Sage Pay.
☐ Administration	Allows access to the Admin pages (including this user section). Use discretion when granting this access.
Transaction List	The page to which the user is directed when they first log in. **NOTE**: If users can see the Updates section they will ALWAYS be taken to the Updates page first whenever unread news or updates are present.

To use those functions, and to protect the administrator account, you need to create new users for yourself and others. Click the Add button to add a new user.

Enter a username for yourself and a password you'll remember, then ensure all the check boxes are enabled for your account. Click the Add button and your new account will appear in the list.

Now click the Logout button and click to Log back in, this time entering:

- Your Vendor name in the **Vendor Name** box.
- The User Name of the account you just created in the **User Name** box.
- The password for the account you just created in the **Password** box.

...and click **Login**.

You are now logged in using your own account and can view your test transactions and use all additional functions. You need only log in as Administrator again if you wish to create additional users, or if you lock yourself out of your own account, you can use the Administrator account to unlock yourself. If you happen to lock out the Administrator account, you will need to contact Sage Pay to unlock it for you.

Detailed context sensitive help is available on every Admin page by clicking the Help button, so a description of the functions will not be presented here. Play with the system until you are comfortable with it though, you cannot inadvertently charge anyone or damage anything whilst on the test server.

Stage 3: Going Live

Once Sage Pay receives your application your account will be created and details will be sent to the bank for confirmation. The bank will be expected to confirm your merchant details within 3 to 5 working days. Once both the Direct Debit (filled out during application) and the confirmation of your merchant details reach Sage Pay, your account will become Live immediately.

This does not mean you will immediately be able to use your Live account

You must ensure you have completed testing of your account before you are granted access to your Live account. Details can be found below:

http://www.sagepay.com/developers/integration_manual/processes_go%20live.html

NB – Without confirmation from the bank and without Direct Debit submission, Sage Pay will not be able to activate your account Live. Sage Pay will only charge you when your account has a valid Direct Debit and confirmation of your merchant details from the bank.

Once your Live account is active, you should point your web site transaction registration scripts at the following URL:

https://live.sagepay.com/gateway/service/vspserver-register.vsp

(for other transaction types, the server-register.vsp section would be changed to refund.vsp, void.vsp, release.vsp etc.)

You should then run an end-to-end transaction through your site, ordering something relatively inexpensive from your site and paying using your own valid credit or debit card. If you receive an authorisation code, then everything is working correctly.

You should then log into the Live Server *My Sage Pay* Admin screens at https://live.sagepay.com/mysagepay and in a similar manner to the test server, first log in as the Administrator, then create a Live System user for yourself, log in as that user, locate your test transaction and VOID it, so you are not charged for the transaction. At this stage the process is complete.

It is worth noting here that none of the users you set up on the *My Sage Pay* system on the Test Server are migrated across to Live. This is because many companies use third party web designers to help design the site and create users for them during test that they would not necessarily like them to have in a live environment. You will need to recreate any valid users on the Live system when you first log in.

Congratulations, you are now Live with the Sage Pay Server Integration method.

Well done. Hopefully the process of getting here was as painless and hassle free as possible. You'll be pleased to know that now you are live we don't cut the strings and run away. You should contact us with any transaction queries that arise or for any help you need with the *My Sage Pay* Admin system.

Here are the best ways to reach us and the best people to reach:

- If you require any information on additional services, e-mail sales@sagepay.com

- If you have a query regarding a Sage Pay invoice, e-mail finance@sagepay.com

- If you have a question about a transaction, have issues with your settlement files, are having problems with your payment pages or *My Sage Pay* Admin screens, or have a general question about online payments or fraud, e-mail support@sagepay.com with your Vendor Name included in the mail.

- If you have any suggestions for future enhancements to the system, or additional functionality you'd like to see added, please e-mail feedback@sagepay.com with your comments. We do take all comments on board when designing upgrades, although we may not be able to answer every mail we get.

- You can call us as well on 0845-111-4455, although our primary method of contact is via e-mail, especially for the Support team, who work on ticketed systems to ensure queries are answered in strict rotation. Lines into Support are limited so where possible it is better to e-mail.

We will also keep you updated about major system changes, new reports and other enhancements via the Updates section in your *My Sage Pay* Admin, plus your e-mail address will be added to our group mail list used to alert you to upgrades and other pending events.

You can also always check our system availability and current issues on the system Monitor page at http://www.sagepay.com/system_monitor.asp

Thanks again for choosing Sage Pay, and we wish you every success in your e-commerce venture.

Appendix A - The Server Integration Protocol v2.23

This section details the contents of the POSTs sent to and from Sage Pay using the Server integration method, quantifying the returned values where appropriate.

A1: Transaction registration

This is performed via a **HTTPS POST** request, sent to the initial Sage Pay Payment URL service **server-register.vsp**. The details should be URL encoded Name=Value fields separated by '&' characters.

Request format (continued overleaf)

Name	Format	Values	Comments
VPSProtocol	Alphanumeric. Fixed 4 characters.	"**2.23**" in this release	Default or incorrect value is taken to be 2.23.
TxType	Alphanumeric Max 15 characters.	"**PAYMENT**", "**DEFERRED" or "AUTHENTICATE"**	See companion document "Server and Direct Shared Protocols" other transaction types (such as Refund, Releases, Aborts and Repeats).
Vendor	Alphanumeric Max 15 characters.	Vendor Login Name	Used to authenticate your site. This should contain the Vendor Name supplied by Sage Pay when your account was created.
VendorTxCode	Alphanumeric Max 40 characters	Vendor Transaction Code	This should be your own reference code to the transaction. Your servers should provide a completely unique VendorTxCode for each transaction.
Amount	Numeric. 1.00 to 100,000.00	Amount for the Transaction containing minor digits formatted to 2 decimal places where appropriate.	Must be positive and numeric, and may include a decimal place where appropriate. Minor digits should be formatted to two decimal places. e.g. 5.10, or 3.29. Values such as 3.235 will be rejected.
Currency	Alphanumeric 3 characters	Three-letter currency code to ISO 4217 Examples: "**GBP**", "**EUR**" and "**USD**"	The currency must be supported by one of your merchant accounts or the transaction will be rejected.
Description	Alphanumeric Max 100 characters	Free text description of goods or services being purchased	The description of good purchased is displayed on the Sage Pay Server payment page.
NotificationURL	Alphanumeric Max 255 characters	Full qualified URL (including http:// or https:// header).	Callback URL to which Notification POSTs are sent (see step A3).
BillingSurname	Alphanumeric Max 20 characters	Customer's surname	In Protocol 2.23, unlike previous protocols, the Billingxxxxx columns are compulsory.
BillingFirstnames	Alphanumeric Max 20 characters	Customer's first names	
BillingAddress1	Alphanumeric Max 100 characters	First line of billing address	
Optional: BillingAddress2	Alphanumeric Max 100 characters	Second line of billing address	
BillingCity	Alphanumeric Max 40 characters	City component of the address	In Protocol 2.23, unlike previous protocols, the Billingxxxxx columns are compulsory.
BillingPostCode	Alphanumeric Max 10 characters	The Post/Zip code of the Card Holder's Billing	
BillingCountry	Alphanumeric Max 2 characters	ISO 3166-1 country code of the cardholder's billing address	

Optional*: BillingState	Alphanumeric Max 2 characters	State code for US customers only*	
Optional: BillingPhone	Alphanumeric Max 20 characters	Phone number at billing address	
DeliverySurname	Alphanumeric Max 20 characters	Customer's surname	In Protocol 2.23, unlike previous protocols, the Deliveryxxxx columns are compulsory.
DeliveryFirstnames	Alphanumeric Max 20 characters	Customer's first names	
DeliveryAddress1	Alphanumeric Max 100 characters	First line of delivery address	
Optional: DeliveryAddress2	Alphanumeric Max 100 characters	Second line of delivery address	
DeliveryCity	Alphanumeric Max 40 characters	City component of the address	
DeliveryPostCode	Alphanumeric Max 10 characters	The Post/Zip code of the Card Holder's delivery address	
DeliveryCountry	Alphanumeric Max 2 characters	ISO 3166-1 country code of the cardholder's delivery address	
Optional*: DeliveryState	Alphanumeric Max 2 characters	State code for US customers only*	
Optional: DeliveryPhone	Alphanumeric Max 20 characters	Phone number at delivery address	
Optional: CustomerEMail	Alphanumeric Max 255 characters	The customer's e-mail address	The current version of the Server integration method does not send confirmation e-mails to the customer. This field is provided for your records only.
Optional: Basket	Alphanumeric Max 7500 characters	See the next page for the Format of the Basket field	You can use this field to supply details of the customer's order. This information will be displayed to you in the *My Sage Pay* Admin screens.
Optional: AllowGiftAid	Flag	**0** = No Gift Aid Box displayed (default) **1** = Display Gift Aid Box on payment screen.	This flag allows the gift aid acceptance box to appear for this transaction on the payment page. This only appears if your vendor account is Gift Aid enabled.
Optional: ApplyAVSCV2	Flag	**0** = If AVS/CV2 enabled then check them. If rules apply, use rules. (default) **1** = Force AVS/CV2 checks even if not enabled for the account. If rules apply, use rules. **2** = Force NO AVS/CV2 checks even if enabled on account. **3** = Force AVS/CV2 checks even if not enabled for the account but DON'T apply any rules.	Using this flag you can fine tune the AVS/CV2 checks and rule set you've defined at a transaction level. This is useful in circumstances where direct and trusted customer contact has been established and you wish to override the default security checks. **This field is ignored for PAYPAL transactions**
Optional: Apply3DSecure	Flag	**0** = If 3D-Secure checks are possible and rules allow, perform the checks and apply the authorisation rules. (default) **1** = Force 3D-Secure checks for this transaction if possible and apply rules for authorisation. **2** = Do not perform 3D-Secure checks for this transaction and always authorise. **3** = Force 3D-Secure checks for this transaction if possible but ALWAYS obtain an auth code, irrespective of rule base.	Using this flag you can fine tune the 3D-Secure checks and rule set you've defined at a transaction level. This is useful in circumstances where direct and trusted customer contact has been established and you wish to override the default security checks. **This field is ignored for PAYPAL transactions**

Optional: Profile	Alphanumeric Max 10 characters	**"NORMAL"** (DEFAULT) or **"LOW"**	A profile of **LOW** returns the new simpler VSP Server payment pages which have only one step and minimal formatting. Designed to run in i-Frames. Omitting this field or sending **NORMAL** renders the normal card selection screen.

Basket Contents

The shopping basket contents can be passed in a single, colon-delimited field, in the following format:

```
Number of lines of detail in the basket field:
Item 1 Description:
Quantity of item 1:
Unit cost item 1 without tax:
Tax applied to item 1:
Cost of Item 1 including tax:
Total cost of item 1 (Quantity x cost including tax):
Item 2 Description:
Quantity of item 2:
....
Cost of Item n including tax:
Total cost of item n
```

IMPORTANT NOTES:

(i) The line breaks above are included for readability only. No line breaks should be included; the only separators should be the colons.

(ii) The first value "The number of lines of detail in the basket" is **NOT** the total number of items ordered, but the total number of rows of basket information. In the example below there are 6 items ordered, (1 DVD player and 5 DVDs) but the number of lines of detail is 4 (the DVD player, two lines of DVDs and one line for delivery)

So, for example, the following shopping cart...

Items	**Quantity**	**Item value**	**Item Tax**	**Item Total**	**Line Total**
Pioneer NSDV99 DVD-Surround Sound System	1	£424.68	£74.32	£499.00	£499.00
Donnie Darko Director's Cut	3	£11.91	£2.08	£13.99	£41.97
Finding Nemo	2	£11.05	£1.94	£12.99	£25.98
Delivery	---	---	---	---	£4.99

Would be represented thus:

```
4:Pioneer NSDV99 DVD-Surround Sound System:1:£424.68:£74.32:£499.00:
£499.00:Donnie Darko Director's Cut:3:£11.91:£2.08:£13.99:£41.97:
Finding Nemo:2:£11.05:£1.94:£12.99:£25.98: Delivery:---:---:---:
---:£4.99
```

If you wish to leave a field empty, you must still include the colon. e.g.

```
DVD Player:1:£199.99:::£199.99
```

A2: Server Response to the Transaction Registration POST

This is the plain text response part of the POST originated by your servers in A1. Encoding will be as Name=Value fields separated by carriage return and linefeeds (CRLF).

Response format:

Name	Format	Values	Comments
VPSProtocol	Alphanumeric. Fixed 4 characters.	Version number of the protocol of the system. This release will return "**2.23**"	This will match the protocol version supplied in A1.
Status	Alphanumeric Max 15 characters.	**"OK"** – Process executed without error **"MALFORMED"** – Input message was missing fields or badly formatted – normally will only occur during development. **"INVALID"** – Transaction was not registered because although the POST format was valid, some information supplied was invalid. E.g. incorrect vendor name or currency. "**ERROR**" – A problem occurred at Sage Pay which prevented transaction registration.	If the VendorTxCode passed in A1 has been used before, but that transaction is still active, then details of that transaction are passed back in this POST and the suffix "REPEATED" is appended to the Status. Your system must be able to handle repeated messages from the VSP. If the status is not OK, the **StatusDetail** field will give more information about the problem. Please notify Sage Pay if a Status report of **ERROR** is seen, together with your VendorTxCode and the StatusDetail text.
StatusDetail	Alphanumeric Max 255 characters	Human-readable text providing extra detail for the Status message.	Always check StatusDetail if the Status is not **OK**
VPSTxId	Alphanumeric 38 characters	Sage Pay's ID to uniquely identify the Transaction on our system.	Only present if Status is **OK** or **OK REPEATED**.
SecurityKey	Alphanumeric 10 characters	A Security key which VSP uses to generate a MD5 Hash for to sign the Notification message (A3 below). The signature is called VPSSignature.	This value is used to allow detection of tampering with notifications from VSP Server. It must be kept secret from the Customer and held in your database. Only present if Status is **OK**.
NextURL	Alphanumeric Full Qualified URL Max 255 characters	URL to which the Vendor must redirect the Customer to continue the Transaction	Only present if Status is **OK**. Note that the full URL must be used for the redirect, including any appended parameters.

A3: Notification of Results for Transactions

The Sage Pay Server will send notification in the request part of a POST to the Notification URL provided in A1. The request will be URL encoded, with Name=Value fields separated by '&' characters.

Request format (continued overleaf)

Name	Format	Values	Comments
VPSProtocol	Alphanumeric 4 characters	"**2.23**" in this release	Protocol version used by the system. Same as sent in Step C1
TxType	Alphanumeric Max 20 characters	"**PAYMENT**", "**DEFERRED" or "AUTHENTICATE"**	As supplied by your site in C1.
VendorTxCode	Alphanumeric Max 40 characters	Your unique Vendor Transaction Code	Same as sent by your servers in Step A1.
VPSTxId	Alphanumeric 38 characters	Sage Pay's ID to uniquely identify the Transaction on our system.	Same value as returned in the response in A2.
Status	Alphanumeric Max 20 characters	"**OK**" – Transaction completed successfully with authorisation. "**NOTAUTHED**" – The Sage Pay system could not authorise the transaction because the details provided by the Customer were incorrect, or not authenticated by the acquiring bank. "**ABORT**" – The Transaction could not be completed because the user clicked the CANCEL button on the payment pages, or went inactive for 15 minutes or longer. **"REJECTED"** – The Sage Pay System rejected the transaction because of the rules you have set on your account. "**AUTHENTICATED"** – The 3D-Secure checks were performed successfully and the card details secured at Sage Pay. "**REGISTERED"** – 3D-Secure checks failed or were not performed, but the card details are still secured at Sage Pay. "**ERROR**" – An error occurred at Sage Pay which meant the transaction could not be completed successfully.	In the case of **NOTAUTHED**, the Transaction **has** completed through the VSP System, but it has not been authorised by the bank. A status of **REJECTED** means the bank may have authorised the transaction but your own rule bases for AVS/CV2 or 3D-Secure caused the transaction to be rejected. In the cases of **ABORT** and **ERROR** (see below) the Transaction **has not** completed through the Server and can be retried. Please notify Sage Pay if a Status report of **ERROR** is seen, together with your VendorTxCode and the StatusDetail text. **AUTHENTICATED** and **REGISTERED** statuses are only returned if the TxType is **AUTHENTICATE**.
StatusDetail	Alphanumeric Max 255 characters	Human-readable text providing extra detail for the Status message	You should always check this value if the Status is not **OK**.
TxAuthNo	Long Integer	Sage Pay unique Authorisation Code for a successfully authorised transaction aka **VPSAuthCode**.	Only present if the transaction was successfully authorised (Status **OK**).

Request format (continued overleaf)

AVSCV2	Alphanumeric Max 50 characters	Response from AVS and CV2 checks. Will be one of the following: "**ALL MATCH", "SECURITY CODE MATCH ONLY", "ADDRESS MATCH ONLY", "NO DATA MATCHES" or "DATA NOT CHECKED".**	Provided for Vendor info and backward compatibility with the banks. Rules set up at the Sage Pay server will accept or reject the transaction based on these values. More detailed results are split out in the next three fields. Not present if the Status is **AUTHENTICATED** or **REGISTERED.**
AddressResult	Alphanumeric Max 20 characters	**"NOTPROVIDED", "NOTCHECKED", "MATCHED", "NOTMATCHED"**	The specific result of the checks on the cardholder's address numeric from the AVS/CV2 checks. Not present if the Status is **AUTHENTICATED** or **REGISTERED**
PostCodeResult	Alphanumeric Max 20 characters	**"NOTPROVIDED", "NOTCHECKED", "MATCHED", "NOTMATCHED"**	The specific result of the checks on the cardholder's Post Code from the AVS/CV2 checks. Not present if the Status is **AUTHENTICATED** or **REGISTERED**
CV2Result	Alphanumeric Max 20 characters	**"NOTPROVIDED", "NOTCHECKED", "MATCHED", "NOTMATCHED"**	The specific result of the checks on the cardholder's CV2 code from the AVS/CV2 checks. Not present if the Status is **AUTHENTICATED** or **REGISTERED**
GiftAid	Flag	**0** = The Gift Aid box was not checked this transaction. **1** = The user checked the Gift Aid box on the payment page	This field is always present even if GiftAid is not active on your account.
3DSecureStatus	Alphanumeric Max 50 characters	**"OK"** - 3D Secure checks carried out and user authenticated correctly. **"NOTCHECKED"** – 3D-Secure checks were not performed. **"NOTAVAILABLE"** – The card used was either not part of the 3D Secure Scheme, or the authorisation was not possible. **"NOTAUTHED"** – 3D-Secure authentication checked, but the user failed the authentication. **"INCOMPLETE"** – 3D-Secure authentication was unable to complete. No authentication occurred. **"ERROR"** - Authentication could not be attempted due to data errors or service unavailability in one of the parties involved in the check.	This field details the results of the 3D-Secure checks (where appropriate) **NOTCHECKED** indicates that 3D-Secure was either switched off at an account level, or disabled at transaction registration with a setting like Apply3DSecure=2
CAVV	Alphanumeric Max 32 characters	The encoded result code from the 3D-Secure checks (CAVV or UCAF).	Only present if the **3DSecureStatus** field is "**OK**"

Request format

AddressStatus	Alphanumeric Max 20 characters	Either "**NONE", "CONFIRMED"** or **"UNCONFIRMED"**	**PayPal Transactions Only**. If AddressStatus is confirmed and PayerStatus is verified, the transaction may be eligible for PayPal Seller Protection. To learn more about PayPal Seller Protection, please contact PayPal directly or visit: https://www.paypal.com/uk/cgi-bin/webscr?cmd=p/gen/ua/policy_spp-outside#spp-policy for further information.
PayerStatus	Alphanumeric Max 20 characters	Either "**VERIFIED"** or **"UNVERIFIED"**	
CardType	Alphanumeric Max 15 characters	**"VISA", "MC", "DELTA", "SOLO", "MAESTRO", "UKE", "AMEX", "DC", "JCB", "LASER", "PAYPAL"**	**MC** is MasterCard, **UKE** is Visa Electron. **MAESTRO** should be used for both UK and International Maestro. **AMEX** and **DC** (DINERS) can only be accepted if you have additional merchant accounts with those acquirers.
Last4Digits	Numeric Max 4 characters	The last 4 digits of the card number used in this transaction. PayPal transactions have 0000	This field is supplied to allow merchants using wallet systems to identify the card to their customers
VPSSignature		MD5 signature of the concatenation of the values of: **VPSTxId+VendorTxCode+Status+TxAuthNo+VendorName+AVSCV2+SecurityKey+AddressResult+PostCodeResult+CV2Result+GiftAid+3DSecureStatus+CAVV+AddressStatus+PayerStatus+CardType+Last4Digits**. **NOTE: MD5 value is returned in UPPER CASE**	To detect any possible tampering with messages, your site should compute the same MD5 signature (which incorporates the Security key provided at Transaction registration) and check it against VPSSignature. You can then decide what to do with transactions that appear to have been tampered with.

A4: Vendor acknowledges receipt of the Notification POST

This is the plain text response part of the POST originated by the Server in the step above. Encoding must be as Name=Value fields separated by carriage-return-linefeeds (CRLF).

Important note for PayPal transactions: Non-PayPal transactions are timed out by the Transaction Monitor and never settled if our Server cannot contact your Notification URL, however as all PayPal transactions are settled instantly (once the shopper has returned to the Sage Pay Order Confirmation Page), if there is a problem with Sage Pay notifying you of the transaction, it is possible that your PayPal Admin area will display a transaction as successful, but your *My Sage Pay* Admin area will state the transaction has failed. We strongly recommend you to log into your PayPal Admin area regularly, and cancel any transactions which are displayed as failed in your *My Sage Pay* Admin area (so that your PayPal Admin area and *My Sage Pay* Admin area coincide).

Response format:

Name	Format	Values	Comments
Status	Alphanumeric Max 20 characters	"**OK**" – Send this if you successfully received the notification Post in C3. Send this unless an error occurs during notification. "**INVALID**" – send INVALID if the details you received in the A3 post were consistent with expectations for this Transaction. The RedirectURL must still be provided, and the VSP will still redirect the Customer back to your site, but the transaction will NOT be settled with the bank. Only send this result if you want to cancel the transaction. "**ERROR**" – An error has occurred during your Notification processing. The Sage Pay system will check for a RedirectURL, and if one is provided the Customer will be redirected to your site, but the transaction will NOT be settled with the bank. Only send this result if you want to cancel the transaction and report an ERROR to Sage Pay.	"**OK**" statuses will allow the transaction to settle and money to move into the Vendor account. **"INVALID"** or **"ERROR"** responses will prevent the transaction from settling, so the customer will not be charged. You should send **OK** in all circumstances where no errors occur in validating the Notification POST, so even if Sage Pay send you a status of ABORT or NOTAUTHED in A3 above, you should reply with an **OK** and a **RedirectURL** that points to a page informing the customer that the transaction did not complete.
RedirectURL	Alphanumeric Max 255 characters	Fully qualified URL to which you'd like the customer redirected on completion of the transaction. This should include the http:// or https:// header.	If you wish to pass parameters back to your own site (such as the session id or transaction code), these should be included in RedirectURL.
StatusDetail	Alphanumeric Max 255 characters	Human-readable text providing extra detail for the Status message	If Status is not **OK**, state what is wrong with the Transaction and why your are rejecting it.

IMPORTANT NOTE: Before writing the three fields above to the Response object of the POST, please ensure you clear your response buffer to remove any header code, comments or HTML. The Sage Pay Server is expecting "Status=" to be the first characters in the response. If it does not see these, it treats the response as though it is an error and fails the transaction! Sage Pay Simulator will warn you about this when you are testing.

A5: Server Integration Full URL Summary

The table below shows the complete web addresses to which you send the messages detailed above.

Transaction Registration (PAYMENT, DEFERRED, AUTHENTICATE)	
Simulator:	https://test.sagepay.com/Simulator/VSPServerGateway.asp?Service=VendorRegisterTx
TEST System:	https://test.sagepay.com/gateway/service/vspserver-register.vsp
Live System:	https://live.sagepay.com/gateway/service/vspserver-register.vsp

Please ensure that you use allow Ports 80 and 443 in order to communicate with our servers (on Simulator/Test/Live).