

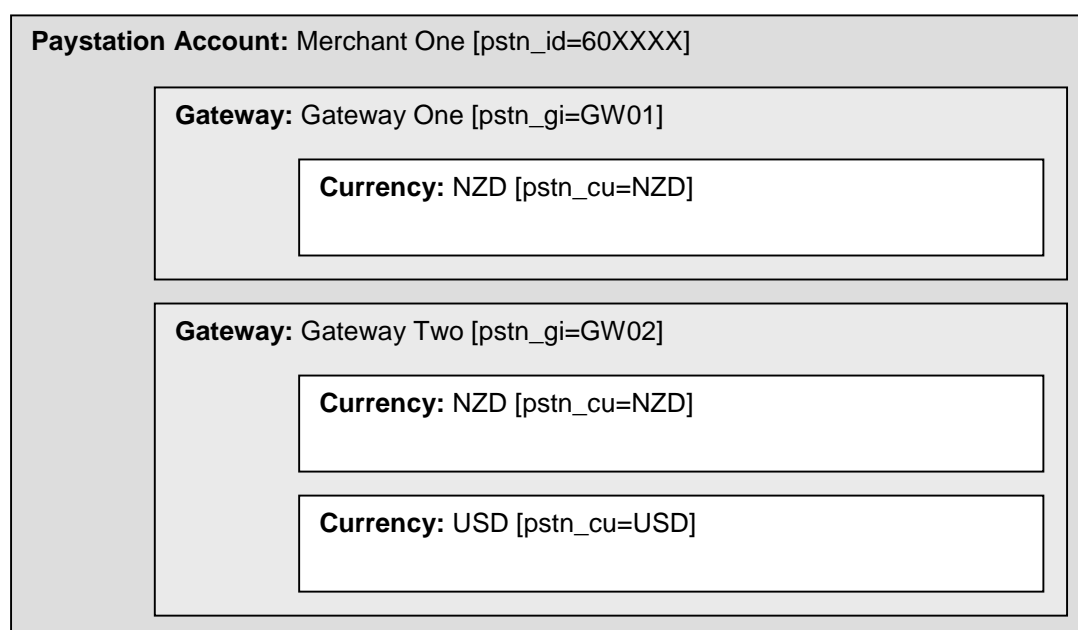
Paystation

Future Payment Hosted Three Party Interface API

A Paystation account has three transactional components: the account itself and one or more gateways, each with one or more currencies.

When you initiate a transaction you need to specify which gateway and can optionally specify a currency to use. If you leave the gateway out the transaction will fail. If you specify a gateway that isn't loaded against your account or a currency that isn't loaded against the gateway the transaction will fail.

Below is a diagram to help you visualise how the pieces fit together.



It is very important to understand that our gateways are in no way directly related to your Merchant Facilities at the Bank – you might have two gateways set up within Paystation that send transactions to a single Bank Merchant Facility (a good reason to do this would be if you were running two websites over a single Bank Merchant Facility and wanted different return URLs for the different sites) – or you might have one gateway that goes to a Bank Merchant Facility and one that goes to a pago account.

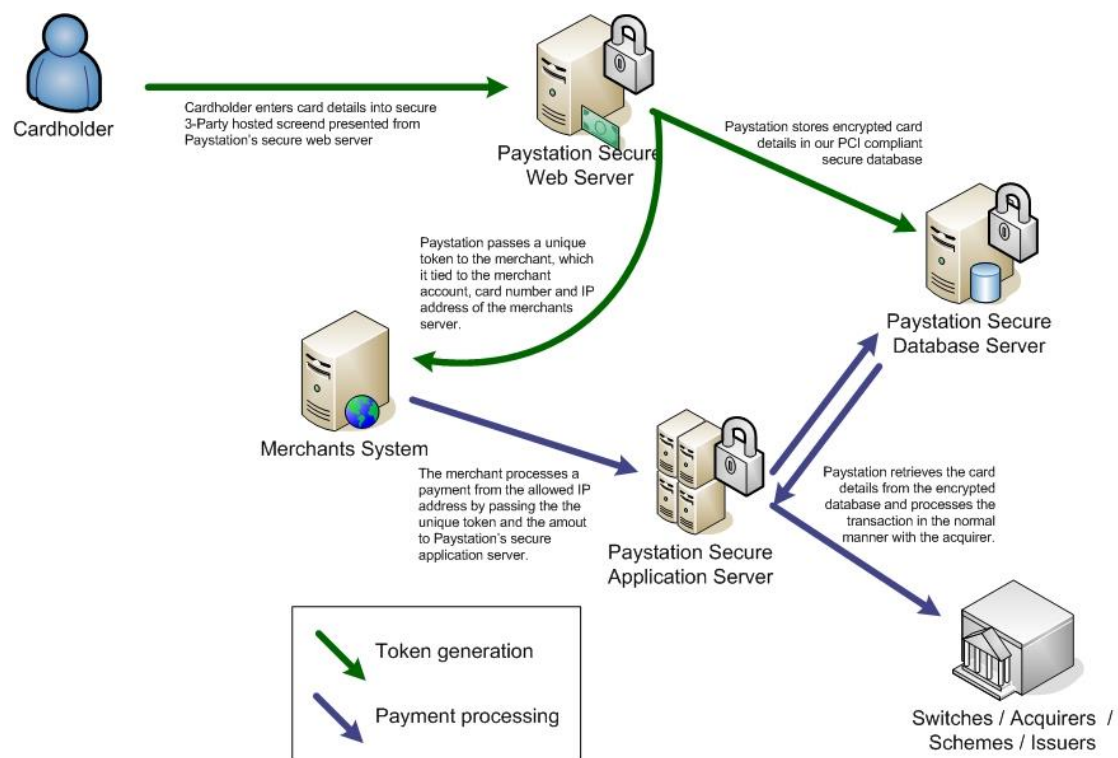
Please note that your gateway probably won't be called GW01 or GW02 – your gateways will have more meaningful names, like CARDPAY and PAGO. We will confirm the name of your gateway(s) when your account is established. You can ask for specific gateway names if that is useful for you, like SHOP01 and SHOP02 or SHOP and DONATIONS.

Transaction process

There are five things you can do with our Future Payment system. They are called Future Pay Save Only, Future Pay Initiator, Future Pay Bill transactions and Future Pay Update.

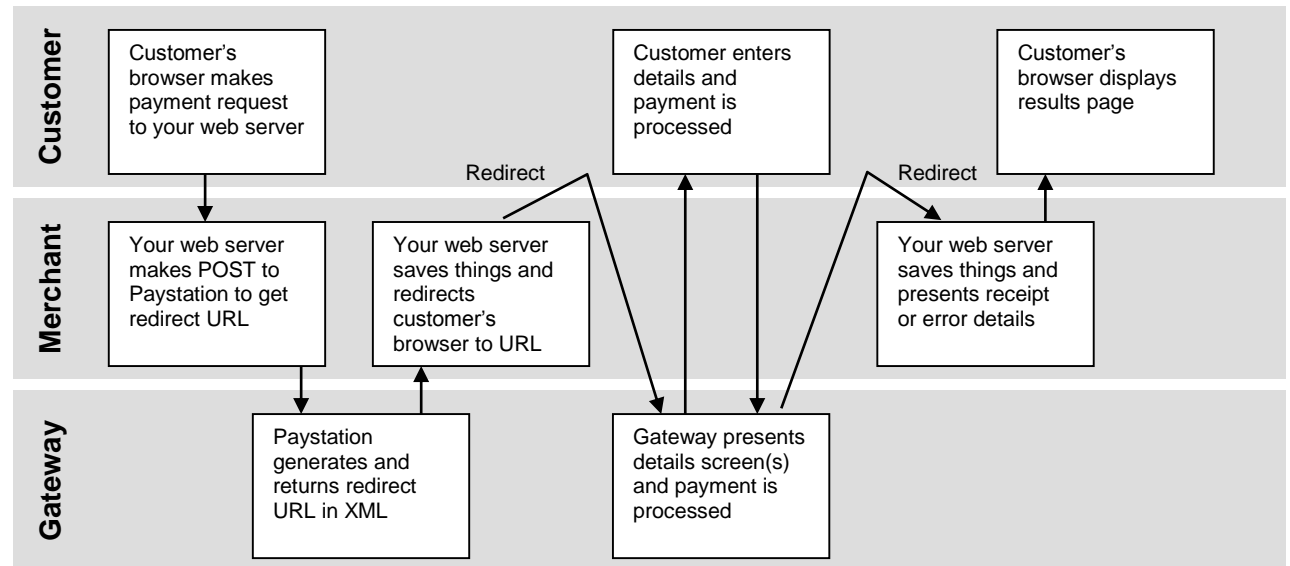
1. Future Payment Save Only (Create a token for future billing only)
2. Future Payment Initiator (Create a token and process a transaction at the same time)
3. Future Payment Bill (Make a payment given only a token previously stored)
4. Future Payment Update (Update a previously stored token)
5. Future Payment Delete (Delete a previously stored token)

The initial transaction (either a Future Pay Save Only or Future Pay Initiator transaction) will be a 3-Party (hosted) transaction, where the card holder is redirected to our servers to enter the card details. Once the card details are saved within our systems you will be returned a token. Subsequent transactions (Future Pay Bill transactions, Future Pay Update and Future Pay Delete) against that card will be processed by sending us the token in a 2-Party (non-hosted) manner. Throughout this process your systems will never have visibility of the card details.



The steps to perform when making a payment using the 3-Party payment method are:

1. Initiate the payment.
2. Store everything locally.
3. Redirect the card holder's browser to the payment page.
4. When the card holder's browser returns to your site interpret the result and action according to your business rules (issue receipt, send emails, etc).
5. Perform a Quick Lookup to verify the payment details, and store these locally.
6. If they don't come back perform a Quick Lookup on the transaction to see if it was actually processed.



The Future Payment Bill and Future Payment Update processes are different to the other two processes in that it is used in a 2-Party manner (there are no redirects to other websites). The trigger for the processing of the transaction will vary depending on your business processes – it could be cardholder initiated in the case of future web shop purchases, or it could be system generated in the case of subscriptions, bill payments, or memberships.

To perform a transaction using the 2-Party method POST your variables directly to Paystation and you will receive the results as XML in the response.

Payment Initiation Request

Payments are initiated via a POST to the Paystation server. Paystation responds with an XML formatted response which (in the case of Future Pay Save Only and Future Pay Initiator transactions) includes the URL you should redirect the card holder's browser to. In the case of Future Pay Bill transactions, you will receive an immediate success or failure to your request in the XML response (ie it functions as a 2-Party transaction).

You do a POST to <https://www.paystation.co.nz/direct/paystation.dll> with the relevant variables, which are documented later on.

After a request is sent Paystation will respond with an XML formatted string. The XML response will be in the same format regardless of the transaction result. You will be able to extract the response values using the standard tools available within your development environment.

If it fails you will get something like this back:

```
<?xml version="1.0" standalone="yes"?>
<response>
<ec>10</ec>
<em>The amount specified is too high or low and exceeds the limits set by this
merchant</em>
<ti>0000000482-01</ti>
<ct/>
<merchant_ref>auto_test</merchant_ref>
<tm>T</tm>
<MerchantSession>cf3tPlBA</MerchantSession>
<UsedAcquirerMerchantID>TEST850047</UsedAcquirerMerchantID>
<TransactionID>0000000482-01</TransactionID>
<PurchaseAmount>10</PurchaseAmount>
<Locale>en</Locale>
<ReturnReceiptNumber/>
<ShoppingTransactionNumber/>
<AcqResponseCode/>
<QSIResponseCode/>
<CSCResultCode/>
<AVSResultCode/>
<TransactionTime>2008-10-02 17:10:38</TransactionTime>
<PaystationErrorCode>10</PaystationErrorCode>
<PaystationErrorMessage>The amount specified is too high or low and exceeds the limits
set by this merchant</PaystationErrorMessage>
<MerchantReference>auto_test</MerchantReference>
<TransactionMode>T</TransactionMode>
<BatchNumber/>
<AuthorizeID/>
<Cardtype/>
<Username>605002</Username>
<RequestIP>203.118.134.77</RequestIP>
<RequestUserAgent/>
<RequestHttpReferrer/>
<PaymentRequestTime>2008-10-02 17:10:38</PaymentRequestTime>
<DigitalOrder/>
<DigitalOrderTime/>
<DigitalReceiptTime/>
<PaystationTransactionID/>
```

</response>

When it all works you will get something like the following (for a Future Pay Save Only and Future Pay Initiator).

```
<?xml version="1.0" standalone="yes"?>
<InitiationRequestResponse>
  <Username>605002</Username>
  <RequestIP>203.118.134.77</RequestIP>
  <RequestUserAgent/>
  <RequestHttpReferrer/>
  <PaymentRequestTime>2008-10-02 15:59:29</PaymentRequestTime>
  <DigitalOrder>https://payments.paystation.co.nz/hosted/?hk=KBzTIBy5oYEI</DigitalOrder>
  <DigitalOrderTime>2008-10-02 15:59:30</DigitalOrderTime>
  <DigitalReceiptTime/>
  <PaystationTransactionID>0000000466-01</PaystationTransactionID>
</InitiationRequestResponse>
```

When you receive this response you would ideally store it with the order details in your order database.

Then send the card holder's browser a redirect to the value returned in DigitalOrder using the appropriate call within your development environment, preferably as a header redirection, remembering that these often have to be the first output from the script / application in order to work.

After the transaction is processed the response will be returned to your Paystation return URL. Your return URLs are stored against your Paystation account. You will have to inform us of your initial return URLs and email us at info@paystation.co.nz if you wish to change them.

The response is sent as a browser redirect in the form of a standard HTTP GET query. You will be able to extract the response values using the standard tools available within your development environment.

We strongly encourage you to implement the Quick Lookup process after the payment is completed to verify that the payment was processed. Please refer to the Quick Lookup API for details on this process – a copy can be downloaded from our website (<http://www.paystation.co.nz>) or requested from us at info@paystation.co.nz.

When you are doing a Future Pay Bill transaction, you will receive a response similar to the following:

```
<?xml version="1.0" standalone="yes" ?>
<PaystationFuturePaymentResponse>
  <ec>0</ec>
  <em>Transaction successful</em>
  <ti>0000922872-01</ti>
  <ct>mastercard</ct>
  <merchant_ref />
  <tm>T</tm>
  <MerchantSession>my_unique_reference</MerchantSession>
  <UsedAcquirerMerchantID>TEST850047</UsedAcquirerMerchantID>
  <TransactionID>0000922872-01</TransactionID>
  <PurchaseAmount>15300</PurchaseAmount>
  <Locale>en</Locale>
  <ReturnReceiptNumber>000000001181</ReturnReceiptNumber>
  <ShoppingTransactionNumber>23334</ShoppingTransactionNumber>
  <AcqResponseCode>00</AcqResponseCode>
```

```

<QSIResponseCode>0</QSIResponseCode>
<CSCResultCode />
<AVSResultCode />
<TransactionTime>2007-06-01 11:26:27</TransactionTime>
<PaystationErrorCode>0</PaystationErrorCode>
<PaystationErrorMessage>Transaction successful</PaystationErrorMessage>
<MerchantReference />
<TransactionMode>T</TransactionMode>
<BatchNumber>20070601</BatchNumber>
<AuthorizeID>001181</AuthorizeID>
<Cardtype>MC</Cardtype>
<Username>pstn_testing</Username>
<RequestIP>203.118.134.90</RequestIP>
<RequestUserAgent>Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0; .NET CLR 1.1.4322;
.NET CLR 2.0.50727)</RequestUserAgent>
<RequestHttpReferrer />
<PaymentRequestTime>2007-06-01 11:26:24</PaymentRequestTime>
<DigitalOrderTime>2007-06-01 11:26:24</DigitalOrderTime>
<DigitalReceiptTime>2007-06-01 11:26:27</DigitalReceiptTime>
<FuturePaymentToken>Im a token</FuturePaymentToken>
<IssuerName>unknown</IssuerName>
<IssuerCountry>unknown</IssuerCountry>
</PaystationFuturePaymentResponse>

```

When you are doing a Future Pay Update, you will receive a response similar to the following:

```

<?xml version="1.0" standalone="yes"?>
<PaystationFuturePaymentResponse>
<ec>34</ec>
<em>Future Payment Saved Ok</em>
<ti>0027575407-01</ti>
<ct>visa</ct>
<merchant_ref/>
<tm>P</tm>
<MerchantSession>karlsuniquemERCHANTSESSIONID060320141447pm</MerchantSession>
<UsedAcquirerMerchantID/>
<TransactionID>0027575407-01</TransactionID>
<PurchaseAmount>0</PurchaseAmount>
<Locale>en</Locale>
<ReturnReceiptNumber/>
<ShoppingTransactionNumber/>
<AcqResponseCode/>
<QSIResponseCode/>
<CSCResultCode/>
<AVSResultCode/>
<TransactionTime>2014-03-06 14:48:20</TransactionTime>
<PaystationErrorCode>34</PaystationErrorCode>
<PaystationErrorMessage>Future Payment Saved Ok</PaystationErrorMessage>
<MerchantReference/>
<TransactionMode>P</TransactionMode>
<BatchNumber/>
<AuthorizeID/>
<Cardtype>VC</Cardtype>
<Username>610691</Username>
<RequestIP>210.4.215.14</RequestIP>

```

```

<RequestUserAgent/>
<RequestHttpReferrer/>
<PaymentRequestTime>2014-03-06 14:48:20</PaymentRequestTime>
<DigitalOrderTime/>
<DigitalReceiptTime>2014-03-06 14:48:20</DigitalReceiptTime>
<PaystationTransactionID>0027575407-01</PaystationTransactionID>
<FuturePaymentToken>paystationtoken</FuturePaymentToken>
<IssuerName>unknown</IssuerName>
<IssuerCountry>unknown</IssuerCountry>
</PaystationFuturePaymentResponse>

```

Future Payment Types

The Future Payments process is IP limited – please tell Paystation what IP numbers you will be running your queries from so we can adjust the security settings on the account. If you are currently running any other type of 2-Party transactions through our system, then the IP numbers you already have loaded will automatically be included in your allowed list for Future Payments.

There are five different phases for Future Payments, these are:

1. Future Payment Initiator (Create a token and process a transaction at the same time)
2. Future Payment Bill (Make a payment given only a token previously stored)
3. Future Payment Save Only (Create a token for future billing only)
4. Future Payment Update (Update a previously stored token)
5. Future Payment Delete (Delete a previously stored token)

The Future Payment Bill, Update and Delete process is different to the other two processes in that it is used in a 2-Party manner (there are no redirects to other websites).

The following is an example URL which is from a Future Payment *Initiator* phase.

[https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_ft=Im%20a%20token&pstn_am=153.00&pstn_af=dollars.cents&pstn_ms=my unique reference&pstn_pi=pstn_testing&pstn_gi=gatewayid&pstn_fp=t&pstn_tm=t&pstn_nr=t](https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_ft=Im%20a%20token&pstn_am=153.00&pstn_af=dollars.cents&pstn_ms=my%20unique%20reference&pstn_pi=pstn_testing&pstn_gi=gatewayid&pstn_fp=t&pstn_tm=t&pstn_nr=t)

The following is an example URL which is from a Future Payment *Bill* phase.

[https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_ft=Im%20a%20token&pstn_am=153.00&pstn_af=dollars.cents&pstn_ms=my unique reference&pstn_pi=pstn_testing&pstn_gi=t&pstn_fp=t&pstn_tm=t&pstn_2p=t&pstn_nr=t](https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_ft=Im%20a%20token&pstn_am=153.00&pstn_af=dollars.cents&pstn_ms=my%20unique%20reference&pstn_pi=pstn_testing&pstn_gi=t&pstn_fp=t&pstn_tm=t&pstn_2p=t&pstn_nr=t)

The following is an example URL which is from a Future Payment *Save Only* phase.

[https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_ft=Im%20a%20token&pstn_ms=my unique reference&pstn_pi=pstn_testing&pstn_gi=t&pstn_fp=t&pstn_fs=t&pstn_tm=t&pstn_nr=t](https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_ft=Im%20a%20token&pstn_ms=my%20unique%20reference&pstn_pi=pstn_testing&pstn_gi=t&pstn_fp=t&pstn_fs=t&pstn_tm=t&pstn_nr=t)

The following is an example URL which is from a Future Payment *Update* phase.

[https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_fp=t&pstn_ft=paystationtoken&pstn_pi=610691&pstn_gi=BUYPOINT&pstn_ms=karlsunique merchantsessionid060320141447pm&pstn_2p=t&pstn_nr=t&pstn_fu=t&pstn_cn=4987654321098769&pstn_ex=1803](https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_fp=t&pstn_ft=paystationtoken&pstn_pi=610691&pstn_gi=BUYPOINT&pstn_ms=karlsunique%20merchant%20sessionid060320141447pm&pstn_2p=t&pstn_nr=t&pstn_fu=t&pstn_cn=4987654321098769&pstn_ex=1803)

The following is an example URL which is from a Future Payment *Delete* phase.

[https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_pi=paystationID&pstn_gi=gatewayID&pstn_ms=unique reference&pstn_fp=t&pstn_ft=savedtoken&pstn_2p=t&pstn_nr=t&pstn_mr=payment reference&pstn_fx=t](https://www.paystation.co.nz/direct/paystation.dll?paystation=empty&pstn_pi=paystationID&pstn_gi=gatewayID&pstn_ms=unique%20reference&pstn_fp=t&pstn_ft=savedtoken&pstn_2p=t&pstn_nr=t&pstn_mr=payment%20reference&pstn_fx=t)

Transaction initiation variables

Variable	Value	Description
paystation REQUIRED	String value	This is an initiator flag for the payment engine and can be nothing, or if your environment requires to assign a value please send ‘_empty’
pstn_pi REQUIRED	String value	The Paystation ID for the account that the payments will be made against
pstn_gi REQUIRED	String value	The Gateway ID that the payments will be made against
pstn_ms REQUIRED	String value (64 chars, 50 chars for PAGO)	Merchant Session – a unique identification code for each financial transaction request . Used to identify the transaction when tracing transactions. Must be unique for each attempt at every transaction.
pstn_am REQUIRED	Integer only For example \$1.00 would be 100, and \$12.45 would be 1245.	Amount – the amount of the transaction, in <i>cents</i> .
pstn_fp REQUIRED	Single character ‘t’ or ‘T’	Future Payment flag – tells Paystation this transaction should be treated as a future payment
pstn_nr REQUIRED	Single character ‘t’ or ‘T’	This is a mandatory for every transaction.
pstn_ft <i>optional</i>	String value (up to 64 chars)	Future Payment Token [optional] – the token to either load or bill a token. If a token is not specified on the <i>initiation phase</i> , Paystation will generate a 64 char token and hand back with the XML packet. Passing this parameter through on a Future Pay Bill transaction is mandatory
pstn_fs <i>optional</i>	Single character ‘t’ or ‘T’	Future Payment Save Only flag [optional] – tells Paystation not to bill the card straight away and to save it for future payments. If this is set and the card details are saved ok, an error 34 will be returned
pstn_cu <i>optional</i>	String value For example New Zealand dollars would be NZD.	Currency [optional] – the three letter currency identifier. If not sent the default currency for the gateway is used.
pstn_tm <i>optional</i>	Single character ‘t’ or ‘T’	Test Mode [optional] - sets the Paystation server into Test Mode (for the single transaction only). It uses the merchants TEST account on the VPS server, and marks the transaction as a Test in the Paystation server. This allows the merchant to run test transactions without incurring any costs or running live card transactions.
pstn_mr <i>optional</i>	String value (64 chars)	Merchant Reference Code [optional] - a non-unique reference code which is

		stored against the transaction. This is recommended because it can be used to tie the transaction to a merchants customers account, or to tie groups of transactions to a particular ledger at the merchant. This will be seen from Paystation Admin. pstn_mr can be empty or omitted.
pstn_ct <i>optional</i>	String Requires EPS. Can contain ONE of the following. mastercard visa amex (if enabled) dinersclub (if enabled) bankcard (if enabled) Please note that pago is not a card type, but a gateway.	Card Type [optional] - the type of card used. When used, the card selection screen is skipped and the first screen displayed from the bank systems is the card details entry screen. Your merchant account must be enabled for External Payment Selection (EPS), you may have to ask your bank to enable this - check with us if you have problems. CT cannot be empty, but may be omitted.
pstn_af <i>optional</i>	String Value "dollars.cents" or "cents"	Amount Format [optional] - Tells Paystation what format the Amount is in. If omitted, it will be assumed the amount is in cents
pstn_mc <i>optional</i>	String Value (255 Chars)	Customer Details [optional] – Stores information for a transaction to be displayed in Paystation Admin
pstn_mo <i>optional</i>	String Value (255 Chars)	Order Details [optional] - Stores information for a transaction to be displayed in Paystation Admin
pstn_2p <i>optional</i>	Single character 't' or 'T'	2-Party [optional] – This is mandatory for a Future Pay Bill transaction and Future Pay Update.
pstn_cn <i>optional</i>	Integer only (no spaces). All 16 (for visa/mastercard) digits of the card number. For example 5123456789012346	Credit card number [optional] – This is mandatory for Future Pay Update if the credit card number is being updated
pstn_ex <i>optional</i>	Integer only (no spaces).	Card Expiry [optional] – Date Format YYMM. This is mandatory for Future Pay Update if the expiry date is being updated
Pstn_fu <i>optional</i>	Single character 't' or 'T'	Update flag [optional] - This is mandatory for Future Pay Update
pstn_fx <i>optional</i>	Single character 't' or 'T'	Delete flag [optional] – This will delete a previously stored token. An error 34 will be returned

Transaction response variables

After the transaction is processed the response will be returned to your Paystation return URL. Your return URLs are stored against your Paystation account. You will have to inform us of your initial return URLs and email us at info@paystation.co.nz if you wish to change them.

The response (on initiator and save only transactions) is sent as a browser redirect in the form of a standard HTTP GET query. You will be able to extract the response values using the standard tools available within your development environment.

Variable	Value	Description
ti	String value	Transaction ID - a string containing the unique transaction ID assigned to the transaction attempt by the Paystation server.
ec	Integer value	Error Code - an integer indicating the success or failure of the transaction. If the transaction was not successful the EC value will help to work out why the payment failed.
em	URL encoded string value	Error Message - a string containing a description of the Error Code.
ms	String value	Merchant Session – a string containing a copy of the value sent to Paystation in the pstn_ms initiation variable.
am	Integer value	Amount – the amount of the transaction, in <i>cents</i> .
futurepaytoken	String value (up to 64 chars)	The token associated with this card

Plus any other items that were originally submitted to Paystation (excluding pstn_ms, pstn_am and pstn_mr which are returned as ms, am, and merchant_ref respectively).

The pstn_pi (pi), pstn_gi and pstn_cu values do not get returned.

POST Back Response

The POST back response XML will look similar to below and you should be able to extract the response values using the standard tools available within your development environment:

```
<?xml version="1.0" standalone="yes"?>
<PaystationPaymentVerification>
<ec>0</ec>
<em>Transaction successful</em>
<ti>0013218363-01</ti>
<ct>visa</ct>
<merchant_ref>hugh@face.co.nz</merchant_ref>
<tm>T</tm>
<MerchantSession>210.4.210.2-1332299131-31</MerchantSession>
<UsedAcquirerMerchantID>TEST850047</UsedAcquirerMerchantID>
<TransactionID>0013218363-01</TransactionID>
<PurchaseAmount>10500</PurchaseAmount>
<Locale>0</Locale>
<ReturnReceiptNumber>000000088493</ReturnReceiptNumber>
<ShoppingTransactionNumber>124992</ShoppingTransactionNumber>
<AcqResponseCode>00</AcqResponseCode>
```

<QSIResponseCode>0</QSIResponseCode>
<CSCResultCode>M</CSCResultCode>
<AVSResultCode>U</AVSResultCode>
<TransactionTime>2012-03-21 16:05:32</TransactionTime>
<PaystationErrorCode>0</PaystationErrorCode>
<PaystationErrorMessage>Transaction successful</PaystationErrorMessage>
<MerchantReference>hugh@face.co.nz</MerchantReference>
<TransactionMode>T</TransactionMode>
<Authentication>
 <auth_Type>3DS</auth_Type>
 <auth_Status>E</auth_Status>
 <auth_SecurityLevel>06</auth_SecurityLevel>
 <auth_HashToken/>
 <auth_3DSID>FOeo5zsKIIvQrM8FCIpN/ZfZP1k=</auth_3DSID>
 <auth_3DSElectronicCommerceIndicator/>
 <auth_3DSEnrolled>N</auth_3DSEnrolled>
 <auth_3DSStatus/>
</Authentication>
<BatchNumber>20120321</BatchNumber>
<AuthorizeID>088493</AuthorizeID>
<Cardtype>VC</Cardtype>
<Username>608878</Username>
<RequestIP>210.4.210.100</RequestIP>
<RequestUserAgent/>
<RequestHttpReferrer/>
<PaymentRequestTime>2012-03-21 16:05:32</PaymentRequestTime>
<DigitalOrderTime>2012-03-21 16:05:32</DigitalOrderTime>
<DigitalReceiptTime>2012-03-21 16:06:02</DigitalReceiptTime>
<PaystationTransactionID>0013218363-01</PaystationTransactionID>
</PaystationPaymentVerification>

Error Code (EC) and Error Message (EM) values.

EC	EM
0	No error - transaction successful
1	Transaction Declined - Bank Error
2	Bank declined transaction
3	Transaction Declined - No Reply from Bank
4	Transaction Declined - Expired Card
5	Transaction Declined - Insufficient funds
6	Transaction Declined - Error Communicating with Bank
7	Payment Server Processing Error - Typically caused by invalid input data such as an invalid credit card number. Processing errors can also occur
8	Transaction Declined - Transaction Type Not Supported
9	Bank Declined Transaction (Do not contact Bank)
10	Purchase amount less or greater than merchant values
11	Paystation couldn't create order based on inputs
12	Paystation couldn't find merchant based on merchant ID
13	Transaction already in progress
22	Merchant_Ref contains invalid characters
23	Merchant Session (ms) contains invalid characters
25	URL encoding error
26	Invalid Amount
34	Future Payment Saved Ok
101	Merchant has been disabled
102	Browser type not supported
104	No return URL for merchant

Codes 0-9 relate to the response code from the ETSL, bank and pago payment systems. Codes 10-26 and 101, 102 and 104 would be identified during data validation, and are Paystation errors.

Extended Error Code Set

If the merchant account is set-up to accept Verify by Visa and SecureCode authenticated transactions, you can ask Paystation to have the extended error codes enabled for your account. These codes are in *addition* to the error result codes above and give information about specifically authentication errors.

EC	EM
A	Transaction Aborted
C	Transaction Cancelled
D	Deferred Transaction
E	Card Issuer Institution Returned a Referral Response - e.g. Contact details to call bank
F	3D Secure Authentication Failed
I	Card Security Code Failed
L	Shopping Transaction Locked (This indicates that there is another transaction taking place using the same shopping transaction number)
N	Cardholder is not enrolled in 3D Secure (Authentication Only)
P	Transaction is Pending
R	Retry Limits Exceeded, Transaction Not Processed
S	Duplicate OrderInfo used. (This is only relevant for Payment Servers that enforce the uniqueness of this field)
T	Address Verification Failed
U	Card Security Code Failed
V	Address Verification and Card Security Code Failed
?	Response Unknown