MCP Payment
Integration Guide

Disclaimer
Every effort has been made to ensure the accuracy of information published in this guide. However MCP cannot accept any responsibility for any errors, inaccuracies, or omissions that may or may not be
published in the guide. To the extent permitted by law, MCP is not liable for loss, damage, or
liability arising from errors, omissions, inaccuracies, or any misleading or out-of-date information whether
published in this guide or from any link in this guide. Realex Payments reserves the right to change this guide and the included software without prior notice or consent.

Table of Content

Document Change History

Version	Revision	Description	Revision Date
3	0	Remove Merchant-Hosted model	5 Dec 2017
2	29	Add frequency param Add paymentmethods param Add Notes for Gateway-Hosted model implementation	27 Nov 2017
2	28	Add TokenizeCard() method Add ValidateCard() method	5 June 2017
2	27	Remove obsolete statusurl parameter from Merchant Hosted Direct Model methods. Add masked cardno and cardco to all Merchant Hosted Direct Model methods response.	10 Apr 2017
2	26	Make stan not mandatory and accept both YYYY and YY expiry date format	31 Mar 2017
2	25	Add custom params 1..3 and is3dssecured to Sale and Authorize	30 Dec 2016
2	24	Update UAT url	28 Dec 2016
2	23	Add 3DS additional info to Sale and Authorize (xid, cavv, eci)	28 Dec 2016
2	22	Rename istoken param on tokenize Remove cardtoken from gateway hosted sale request Provide additional cardtoken usage info	22 Dec 2016
2	21	Add recurring for sale	23 Nov 2016
2	20	Make authcode and stan not mandatory for Capture	22 Nov 2016
2	19	Make CVV optional	4 Nov 2016
2	18	Add LCP Loyalty Cash Points to Appendix A	16 May 2016
2	17	Add mid and reference to Check3DS response.	13 May 2016
2	16	Void, Capture and Authorize ref param was changed on reference	24 Apr 2016
2	15	Reverse method has been removed. Use Void instead. Added Query method.	18 Apr 2016
2	14	Added Check3DS for Merchant Hosted Model. Obsolete 3DS Check section has been removed. GetResponseFgKey() function code has been updated.	15 Apr 2016
2	13	Added authorize, capture and void. PreAuth removed. Added description for every operation.	29 Mar 2016
2	12	Add tokenization	21 Aug 2015
2	11	Add more description for product info	26 Jun 2015
2	10	Use ref in fg key generation string	10 Mar 2015

2	9	Add reverse, refund, pre auth Add xml request for purchase	16 Feb 2015
2	8	New Live and Test URL Change parameter name ref to reference	5 Feb 2015
2	7	Support stand-alone Check3DS service	8 Sep 2014
2	6	Support Merchant-Hosted Response (Direct API model, non-3DS)	4 Jul 2014
2	5	Support Merchant Hosted payment flow (Redirect model, support 3DS)	12 Mar 2014
2	4	Set buyer, tel, email, lang fields as non-mandatory	8 Feb 2014
2	3	Fix a missing '&' typo in MD5 example	30 Jan 2014
2	2	Add Authorization details to the parameter tables	12 Sep 2013
2	1	Change mandatory parameter indication from ‘O’ to ‘M’ to avoid confusion.	19 Apr 2013
2	0	New webpay and webpaytest URL	1 Apr 2013
1	3	Add fgkey parameter	19 May 2008
1	2	Change Live URL	2 Apr 2008
1	1	Add 3Dsecure parameter	25 Mar 2008
1	1	Initial Document	28 Nov 2007

Introduction

Thank you for using Mobile Credit Payment (MCP) as your payment partner.

MCP is a e-commerce payment solution that is simple to integrate and get started with online payment. Both Gateway-Hosted and Merchant-Hosted payment model can be supported with Payment integration.

This document describes the integration of credit card transactions with MCP. It is intended to be read by technical personnel responsible for implementing MCP Payment integration with their respective website or shopping cart. It is assumed that the reader has working knowledge of the programming languages required to integrate.

We hope this document will help you integrate your website to our payment gateway as quickly as possible.

Steps for integration with MCP

- Apply as a merchant of MCP.
- Audit merchant website by MCP.
- Decide/select integration model
- Implement test payment with test URL.
- Request Live MID to MCP.
- Open service.
- Merchant shall be provided ID and PW with URL for retrieving transaction details.

Comments, feedback and Support

Please email to ecom.support.team@mcpayment.com for technical support.

Integration Model

This document describes gateway-hosted integration model.

In the gateway-hosted payment model, the consumer is directed to the gateway service provider (MCP) to complete the payment transaction. (Figure 1)

The advantage of this method is that merchant do not need to worry about handling and transmitting consumer card details. This helps to reduce risk on merchant.
Merchant also do not need to implement payment flow and confirmation which will be handled by webPay.

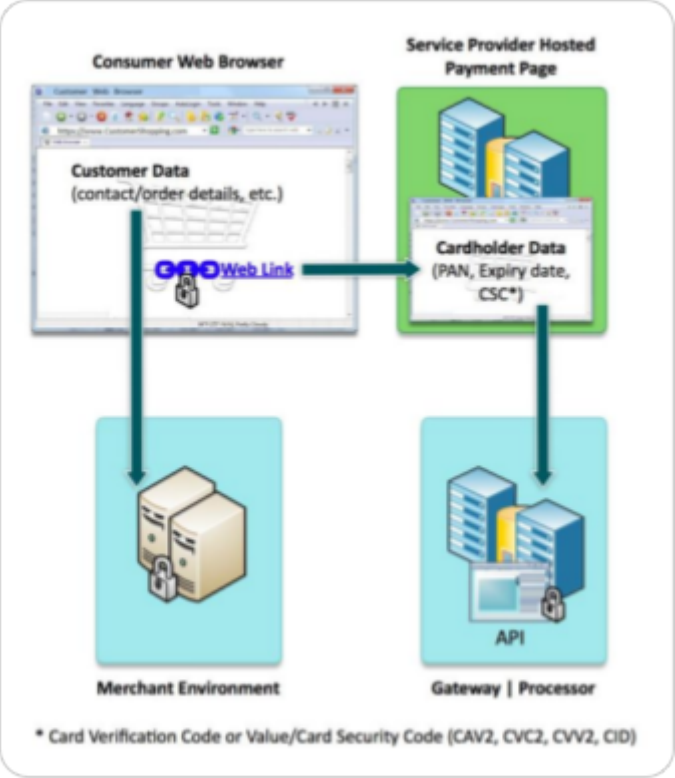

Figure 1

PCI-DSS

PCI-DSS is the Payment Card Industry - Data Security Standard which provides the standard to enhance payment card data security. All merchants and service providers are required to adhere to the respective scope applicable.

Merchant should refer to Self-Assessment Questionnaire (SAQ) to assess readiness according to PCI-DSS.
Merchant using gateway-hosted model should refer to SAQ A.
Merchant using merchant-hosted model should refer to SAQ D - Merchant.
Partners and gateways should have their own PCI certification.

Terms & Glossary

Term	Description
3DS	3-D Secure
PCI-DSS	Payment Card Industry - Data Security Standard
SAQ	Self-Assessment Questionnaire

Technical Integration

How to send the request

Send the request data via https with “POST”. This can be achieved, for example, by using HTML form or code.

Request URL

- Test URL : https://map.uat.mcpayment.net/payment/dopayment
- Live URL : https://map.mcpayment.net/payment/dopayment

Request Parameters

Parameter[1,2]	Type	Length	Mandatory [3]	Explanation
mid	AN	10	M	Merchant ID assigned by MCP
txntype	A	30	M	SALE or AUTH
reference	AN	30	M	A unique transaction reference generated by merchant. This number cannot be recycled even though the previous transaction could be unsuccessful. (e.g. orderID)
cur	A	3	M	e.g. USD, SGD, KRW…(Refer to Appendix A.)
amt	N	10, 2	M	Payment total amount for the transaction, e.g. 10.50, 1000.15. Do not use “,”.
shop	AN	255		Shop name (if the displaying merchant name on payment window is different with registered merchant name in MCP)
buyer	AN	64		Buyer name
tel	AN	32		Buyer tel.
email	AN	32		Buyer email address. The buyer will receive approval email from MCP.
product	AN	255	M	Product details. - Single line item: “product name” (note: no line breaks before, after or within) - Multi-line items: “Name <|> ID <|> Description <|> quantity <|> unit price <|> taxable"

lang	A	2		The language will be shown on the payment window. (Refer to Appendix B.)
returnurl	AN	255	M	The merchant page where the user will be redirected to upon end of transaction. This merchant page will display “Success” or “Failed” based on returned **rescode**. This page may not be called if user simply closed the browser.
statusurl	AN	255	M	Where transaction status and other information will be sent to via the backend. This merchant page is guaranteed be called by WebPay Gateway. So you can implement database update logic in this page.
param1	AN	255		This value will be sent to merchant without any changes.
param2	AN	255		This value will be sent to merchant without any changes.
param3	AN	255		This value will be sent to merchant without any changes.
charset	A	32		Default is “UTF-8”. If a merchant has another charset, set to this parameter.
fgkey	AN	32	M	Refer to 3.2 MD5 Hashing
tokenize	AN	1		If the merchant wants to do tokenization, then set value to Y. System will return **cardtoken.** Default N. In case direct sale without tokenization: pass N, empty value or exclude param from request. **cardtoken** can be used to create further Sale or Authorize using Merchant-Hosted integration model
paymentmethods	AN	255		Merchant can dynamically configure what payment sections will be shown to payer by specifing **paymentmethods** = "card,mwr,ib,fpx" subset. If empty shows all sections supported by configured terminals. Empty by default. Use ‘,’ as delimiter. Any subset of supported payment methods can

				be used. For example: “card”, “card,mwr”, “ib,mwr”, “ib”, etc.
frequency	A	32		In order to make a payment recurring, merchant can specify frequency param with possible values: MONTHLY, QUARTERLY, YEARLY. All further transactions will be triggered as copy of original transaction at same day time with specified frequency. Payer will be charged until recurring will be manually canceled in WebPay merchant backend.

Note. 1 – Parameter names have to be written lower case alphabet.
Note. 2 – If parameter values include special characters, please use “URLEncoder.encode” and “URLDecoder.decode”.
Note. 3 – M – Mandatory.

Response Parameters

Response parameters will be returned to returnurl and statusurl callback handlers as HTTP Form Post data.

The following is a sample of how XML response looks like:

Parameter	Type	Length	Mandatory	Explanation
mid	AN	10	M	Merchant ID assigned by MCP
txntype	A	30	M	SALE
reference	AN	30	M	A unique transaction reference generated by merchant. This number cannot be recycled even though the previous transaction could be unsuccessful. (e.g. orderID)
cur	A	3	M	e.g. USD, SGD, KRW…(Refer to Appendix A.)
amt	N	10, 2	M	Payment total amount for the transaction. (e.g. 10.50, 1000.15)
shop	AN	255		Shop name (if the displaying merchant name on payment window is different with registered merchant name in MCP)
buyer	AN	64		Buyer name
tel	AN	32		Buyer tel.
email	AN	32		Buyer email address. The buyer will receive approval

				email from MCP.
product	AN	255		Product name (note : no line breaks before, after or within)
lang	A	2		The language will be shown on the payment window. (Refer to Appendix B.)
param1	AN	255		Same with request parameter value
param2	AN	255		
param3	AN	255		
transid	AN	24	M	The unique transaction ID of MCP
rescode	AN	6	M	The response code. If a transaction is successful, it will be “0000”. Else will be other error codes.
resmsg	AN	700	M	The response message: “Approved” or other reason messages.
authcode	AN	15	M	The bank approval code for successful transactions.
stan	N	6	OM	System Trace Audit Number.
cardno	N	20	M	Masked credit card number.
cardco	AN	32	M	The credit card type of customer (VISA, MASTER, AMEX, DINERS)
resdt	N	14	M	yyyy-MM-dd HH:mm:ss
secure3D	A	1	M	‘Y’ or ‘N’
fgkey	AN	32	M	Refer to 4.2 MD5 Hashing (if only rescode=0000, fgkey’s value is valid.)
cardtoken	AN	50	C	If the merchant set tokenize “Y”, MCP will response back the cardtoken value for future use
frequency	AN	30		Same with request parameter value
isrecurring	A	3		“Yes” or “No” Indicates whether payment has been added as recurring to the system with specified **frequency**

Implementation notes

statusurl is main callback handler. It triggers first and postback full data package about the payment. System sends it in background. No page will be opened for it. Please implement database update

logic in this callback handler only.

returnurl is auxiliary callback handler. After end of the payment process system redirects payer to this url. Use data provided to show basic info about the payment to payer. Please do not use this url for database update. Otherwise, if payer will close the page, your database will be not updated and you will lose the transaction.

We strictly recommend to implement both callback url handlers: **resulturl** and **statusurl.** Otherwise we do not guarantee postback data (payment result) delivery.

For both callback handlers read POST data ($_POST array in php) only. GET url data is deprecated and will be removed.

Analyze **rescode** in the callback handlers. Payment could be considered successful if rescode=**0000**. In all other cases error occurred.

Validate response fgkey if rescode == “0000” only. FgKey generation rules
Purpose of fgkey validation is to ensure if response data wasn’t changed by fraud.

Request samples

Please refer to attached sample source files. (Sample are provided for reference purpose only and should not be used directly in production system as is.)

Dummy card numbers for test

- Card No. : 4111 1111 1111 1111 or 5555 5555 5555 4444
- Expiry Date : any expiry
- CVV : 123

In case of approval test using test URL, above Card information should be used for approval.

Query

Query Transaction Information.

Merchant System calls the Query function with either the Merchant’s transaction number (Reference) or Payment Platform’s transaction number of the sale order to return the transaction details.

Use either **reference** or **transid** to query transaction information.
If both parameters are specified, then **transid** will be used.
If there are more than one transaction with the **reference,** then error will be returned.

Possible scenario for using **transid** or **reference**:

Scenario 1. If Purchase/Sale/Authorize/Void/Refund transaction failed and no response received, then Merchant will not receive the **transid.** So you can call Query with **reference** only.

Scenario 2. If transaction succeed and Merchant received response with **transid,** then **transid** can be

used for Query.

Request URL

- Test URL : https://map.uat.mcpayment.net/api/PaymentAPI/Query
- Live URL : https://map.mcpayment.net/api/PaymentAPI/Query

Request Parameters

Parameter[1,2]	Type	Length	Mandatory [3]	Description
mid	AN	10	M	Merchant ID assigned by MCP
txntype	A	30	M	QUERY
reference	AN	20	C	A unique transaction reference generated by merchant. This number cannot be recycled even though the previous transaction could be unsuccessful. (e.g. orderID)
transid	AN	24	C	The unique transaction ID of MCP.
cur	A	3	M	e.g. USD, SGD, KRW…(Refer to Appendix A.)
amt	N	10, 2	M	Payment total amount for the transaction, e.g. 10.50, 1000.15. Do not use “,”.
lang	A	2		The language will be shown on the payment window. (Refer to Appendix B.)
charset	A	32		Default is “UTF-8”. If a merchant has another charset, set to this parameter.
fgkey	AN	32	M	Refer to 4.2 MD5 Hashing

Note. 1 – Parameter names have to be written lower case alphabet.
Note. 2 – If parameter values include special characters, please use “URLEncoder.encode” and “URLDecoder.decode”.
Note. 3 – M – Mandatory, C - Conditional.

Request JSON

```
{
   "mid":"3112040001",
   "txntype":"QUERY",
```

```
   "reference":"12132s",
   "cur":"SGD",
   "amt":"1.00",
   "lang":"EN",
   "charset":"",
   "fgkey":"c5ce6c40b3141ca394cbf684c0bfb044"
}
```

Response Parameters

Parameter	Type	Length	Mandatory	Explanation
mid	AN	10	M	Merchant ID assigned by MCP
txntype	A	30	M	QUERY
reference	AN	30	M	A unique transaction reference generated by merchant. This number cannot be recycled even though the previous transaction could be unsuccessful. (e.g. orderID)
cur	A	3	M	e.g. USD, SGD, KRW…(Refer to Appendix A.)
amt	N	10, 2	M	Payment total amount for the transaction. (e.g. 10.50, 1000.15)
transid	AN	24	M	The unique transaction ID of MCP.
authcode	AN	8	M	The bank approval code for successful transactions.
stan	N	6	M	System Trace Audit Number.
cardno	N	20	M	Masked credit card number.
cardco	AN	32	M	The credit card type of customer (VISA, MASTER, AMEX, DINERS)
cardtoken	AN	50	O	Credit card token.
status	AN	10	M	Transaction status code. (Refer to Appendix D.)
txndt	N	14	M	Transaction date time. yyyy-MM-dd HH:mm:ss
txnlastupdatedt	N	14	M	Transaction last update date time. yyyy-MM-dd HH:mm:ss

txnrescode	AN	6	M	Queried transaction response code. If a transaction is successful, it will be “0000”. Else will be other error codes.
txnresmsg	AN	700	M	Queried transaction response message: “Approved” or other reason messages.
rescode	AN	6	M	The response code. If a transaction is successful, it will be “0000”. Else will be other error codes.
resmsg	AN	700	M	The response message: “Approved” or other reason messages.
resdt	N	14	M	yyyy-MM-dd HH:mm:ss
fgkey	AN	32	M	Refer to 4.2 MD5 Hashing (if only rescode=0000, fgkey’s value is valid.)

Response JSON


```
{
   "mid":"3112040001",
   "txntype":"QUERY",
   "reference":"12132s",
   "cur":"SGD",
   "amt":"1.00",
   "transid":"1001604137i4rnnm7",
   "authcode":"",
   "stan":"",
   "cardno":"411111XXXXXX1111",
   "cardco":"VISA",
   "status":"REJ",
   "txnrescode":"999",
   "txnresmsg":"System error. There is an error in XML document",
   "txndt":"2016-04-13 15:48:03",
   "txnlastupdatedt":"2016-04-13 15:51:26",
   "resdt":"2016-04-19 15:30:29",
   "fgkey":"85c5d89c2d79babdc432f8b7314718f5",
   "cardtoken":null,
   "rescode":"0000",
   "resmsg":"Success"
}
```


MD5 Hashing

MD5 hashing is an encryption method that ensures that URLs have not been manipulated or changed. This allows the integrity of the link in a data exchange between two parties to be checked.

MD5 is an abbreviation for “Message Digest Algorithm 5” and is a widely used cryptographic hash function.
This length of the hash value produced by this hash function is 128 bits. The hash value is returned as a 32-digit hexadecimal number, which look like this: **34048ce4cd069b624f6e021ba63ecde5**

How it works

MD5 should always be used when you want to request every link. The MD5 verifies the integrity of the URL of the MCP link entered by comparing the new MD5 sum with a previously established sum. In this way, it can be determined whether the URL has been changed. The primary purpose of this is to prevent the manipulation of data.

Setting up the MD5 hashings

MD5 requires a secret key which you approve. This key will be saved by your account support manager in the service area. You must then use this key to form a hash and build it into the MCP link.

Hashing via the merchant

In order to access the MCP link so-called “fgkey”, which contains the MD5 hash and is assigned as a parameter in MCP link, is required. The fgkey is generated by the “dynamic secret key(dynkey)” and MCP link inclusive “mid”, “reference” or “ref”, “cur”, “amt” parameters. In case of response, inclusive “mid”, “reference” or “ref”, “cur”, “amt”, “rescode”, “transid” parameters. (**The order of parameters is important**.)

Example:

- dynkey
  F6DCE41DA82064F478B934663FD2A07E (secret key)
- MCP link
  - Request
    ?mid=18A726ECD3&ref=abcd1234567890&cur=USD&amt=1.0
  - Response
    ?mid=18A726ECD3&ref=abcd1234567890&cur=USD&amt=1.0&rescode=0000&transid=20150212000008
- MD5
  (**F6DCE41DA82064F478B934663FD2A07E?**mid=18A726ECD3&ref=abcd1234567890&cur=USD&amt=1.0)
- The result is the hash
  5b3795539ac23dee247682094b74e1d1 (This is value of the fgkey parameter.)

Sample Code (C#)

```
        private static string GetFgKey(string dynkey, string mid, string reference,
string currency, decimal amount)
        {
            string fgKey = string.Empty;
```

```
            try
            {
                string source = string.Format("{0}?mid={1}&ref={2}&cur={3}&amt={4}",
                    dynkey, mid, reference, currency,
amount.ToString(CultureInfo.InvariantCulture));

                using (MD5 md5Hash = MD5.Create())
                {
                    fgKey = GetMd5Hash(md5Hash, source);
                }

                return fgKey;
            }
            catch (Exception ex)
            {
                LogManager.LogException(ex);
                return string.Empty;
            }
        }

        private static string GetResponseFgKey(string dynkey, string mid, string
reference, string currency, decimal amount, string rescode, string transid)
        {
            string fgKey = string.Empty;

            try
            {
                string source =
string.Format("{0}?mid={1}&ref={2}&cur={3}&amt={4}&rescode={5}&transid={6}",
                    dynkey, mid, reference, currency,
amount.ToString(CultureInfo.InvariantCulture), rescode, transid);

                using (MD5 md5Hash = MD5.Create())
                {
                    fgKey = GetMd5Hash(md5Hash, source);
                }

                return fgKey;
            }
            catch (Exception ex)
            {
                return string.Empty;
            }
        }

        private static string GetTokenizeFgKey(string dynkey, string mid, string cardno,
string expmonth, string expyear, string cardholder)
        {
            string fgKey = String.Empty;

            try
            {
                string source =
String.Format("{0}?mid={1}&cardno={2}&expmonth={3}&expyear={4}&cardholder={5}",
                    dynkey, mid, cardno, expmonth, expyear, cardholder);

                using (MD5 md5Hash = MD5.Create())
                {
                    fgKey = GetMd5Hash(md5Hash, source);
                }

                return fgKey;
            }
            catch (Exception ex)
            {
```

```
                LogManager.LogException(ex);
                return String.Empty;
            }
        }

        private static string GetTokenizeResponseFgKey(string dynkey, string mid, string
cardtoken, string rescode)
        {
            string fgKey = String.Empty;

            try
            {
                string source = String.Format("{0}?mid={1}&cardtoken={2}&rescode={3}",
                    dynkey, mid, cardtoken, rescode);

                using (MD5 md5Hash = MD5.Create())
                {
                    fgKey = GetMd5Hash(md5Hash, source);
                }

                return fgKey;
            }
            catch (Exception ex)
            {
                return String.Empty;
            }
        }

        static string GetMd5Hash(MD5 md5Hash, string input)
        {
            // Convert the input string to a byte array and compute the hash.
            byte[] data = md5Hash.ComputeHash(Encoding.UTF8.GetBytes(input));

            // Create a new Stringbuilder to collect the bytes
            // and create a string.
            var sBuilder = new StringBuilder();

            // Loop through each byte of the hashed data
            // and format each one as a hexadecimal string.
            for (int i = 0; i < data.Length; i++)
            {
                sBuilder.Append(data[i].ToString("x2"));
            }

            // Return the hexadecimal string.
            return sBuilder.ToString();
        }
```

Shopping Cart reference integration

Reference integration to these shopping carts are available for merchant to adapt according to their requirement. At this time, we are unable to provide customisation for individual merchant requirement. Please contact us for details.

- Magento
- ZenCart
- OpenCart
- nopCommerce

Going Live

This section describes the steps to go live with the integration.

Preliminary checks:

1. Integration and testing in the test environment is completed successfully.
2. Commercial paperworks and payment are all cleared.
3. Any necessary approval from banks had been received.

Once the above are completed:

1. Inform our support team: support.ecom.team@mcpayment.com
2. You will receive the Go Live Form. Fill up and send back.
3. You account will be setup in live environment.
4. You should receive necessary live account details in 1-2 days.

Appendix A Currency List

Currency Code	Currency Name
SGD	Singapore Dollar
HKD	Hong Kong Dollar
MYR	Malaysia Ringgit
BRD	Brunei Dollar
KRW	Korea Won
USD	US Dollar
AUD	Australian Dollar
EUR	Euro
GBP	Pounds Sterling
JPY	Japan Yen
LCP	MyCash Points or Loyalty Cash Points

Further code not listed can be found in ISO 4217 Currency Code.

You must send only currency code that you are approved for during the sign up.

Appendix B Languages

Language Code	Language Name
EN	English
CN	Chinese
KR	Korean
JP	Japanese

Appendix C Response codes and messages

Code	Message
200/0000	Approved
300	Send Data Error
350	Path Info Not Valid
400	Encrypted Hash mismatch
450	Transaction Failed
500	Server Error
3031	Refer to Card Issuer
3032	Refer to Issuer's Special Conditions
3033	Invalid Merchant
3034	Pick Up Card
3035	Do Not Honour
3036	Error
3037	Pick Up Card, Special Conditions
3039	Request in Progress
3130	Partial Amount Approved
3132	Invalid Transaction
3133	Invalid Amount
3134	Invalid Card Number
3135	No Such Issuer
3137	Customer Cancellation
3138	Customer Dispute
3139	Re-enter Transaction
3230	Invalid Response
3231	No Action Taken
3232	Suspected Malfunction
3233	Unacceptable Transaction Fee
3234	File Update not Supported by Receiver
3235	Unable to Locate Record on File
3236	Duplicate File Update Record
3237	File Update Field Edit Error
3238	File Update File Locked Out
3239	File Update not Successful
3330	Format Error
3331	Bank not Supported by Switch
3332	Completed Partially
3333	Expired Card—Pick Up
3334	Suspected Fraud—Pick Up
3335	Contact Acquirer—Pick Up
3336	Restricted Card—Pick Up
3337	Call Acquirer Security—Pick Up
3338	Allowable PIN Tries Exceeded
3339	No CREDIT Account
3430	Requested Function not Supported

3431	Lost Card—Pick Up
3432	No Universal Amount
3433	Stolen Card—Pick Up
3434	No Investment Account
3531	Insufficient Funds
3532	No Cheque Account
3533	No Savings Account
3534	Expired Card
3535	Incorrect PIN
3536	No Card Record
3537	Trans. not Permitted to Cardholder
3538	Transaction not Permitted to Terminal
3539	Suspected Fraud
3630	Card Acceptor Contact Acquirer
3631	Exceeds Withdrawal Amount Limits
3632	Restricted Card
3633	Security Violation
3634	Original Amount Incorrect
3635	Exceeds Withdrawal Frequency Limit
3636	Card Acceptor Call Acquirer Security
3637	Hard Capture—Pick Up Card at ATM
3638	Response Received Too Late
3735	Allowable PIN Tries Exceeded
3836	ATM Malfunction
3837	No Envelope Inserted
3838	Unable to Dispense
3839	Administration Error
3930	Cut-off in Progress
3931	Issuer or Switch is Inoperative
3932	Financial Institution not Found
3933	Trans Cannot be Completed
3934	Duplicate Transmission
3935	Reconcile Error
3936	System Malfunction
3937	Reconciliation Totals Reset
3938	MAC Error
3939	Reserved for National Use
999	System Error
998	Mandatory field(s) is(are) empty
997	Invalid field(s)'s length
996	Invalid Merchant
995	Invalid currency
994	Invalid transaction
993	Invalid card number

992	Invalid refund amount

Appendix D Transaction status list

Code	Status
PRO	Processing
APP	Approved
REJ	Rejected
DEC	Declined
VOI	Voided
REF	Refunded
REV	Reversed
ARV	Auto Reversed
SUB	Submitted
SET	Settled
CAP	Captured
PEN	Pending