Version

2.23

SOUTH EASTERN EUROPE

NestPay®

Merchant Integration API Manual

www.asseco-see.com.tr

Document Information

Project/Product Name	
Project Manager	
Document Version No	

Document Code			
Prepared By		**Preparation Date**	21/10/2015
Reviewed By		**Review Date**	21/10/2015

Distribution List

From	Date	Phone/Fax

To	Action*	Due Date	Phone/Fax

* Action Types: Approve, Review, Inform, File, Action Required, Attend Meeting, Other (please specify)

Version History

Ver. No.	Ver. Date	Revised By	Description
2.6	10/21/2015	Okan Gürbüz	File Template has been changed.
2.7	12/5/2012	Nildem Demir	TelFax
2.8	12/31/2012	Nildem Demir	Grace period
2.9	10/06/2013	Selcuk Yılmaz	Port information has been Updated
2.10	12/11/2013	Nihal Müstecaplıoğlu	Recurring Order Modification
2.11	20/01/2014	Nildem Üçok	ProcReturnCode
2.12	10/02/2014	Sinan Özışık	Wording
2.13	18/03/2014	Selcuk Yılmaz	IP parameter is edited.
2.14	2/7/2014	Nildem Üçok	MailOrder
2.15	25/8/2014	Nildem Üçok	AAV Response Code
2.16	02/09/2014	Yiğit İPÇİOĞLU	Mode parameter has been removed
2.17	04/11/2014	Erdem BEĞENİLMİŞ	Card Brand
2.18	03/02/2015	Erdem BEĞENİLMİŞ	Adding 3D Params to Order Status & History Queries
2.19	05/03/2015	Yiğit İPÇİOĞLU	Information about "data" added
2.20	28/04/2015	Nihal Müstecaplıoğlu	InvoiceNumber extra added
2.21	26/06/2015	Ünsal Körlü	Terminal id and merchant id extras added to transaction response and order history response.
2.22	27/07/2015	Emre Çağlar	Note added to PostAuthorization about TransId

Ver. No.	Ver. Date	Revised By	Description
2.23	31/07/2015	Erdem Beğenilmiş	Order Status Query – Recurring Transaction Support

Proprietary Notice

Contents

1. Virtual POS API

Virtual POS API is provided for the merchants so that they can make:

- Primary transactions (PreAuthorization, Auto-PostAuthorization)
- Secondary transactions (PostAuthorization, Void, Refund)
- Order status query
- Order history query

The main request is an XML document in CC5AS XML format. Different API's are provided for different programming languages and platforms to create the proper request and send it to the API server. The following **API's** are provided:

- XML API
- Java API (Jpay)
- DLL API
- .NET API

1.1 Transaction Types

1.1.1 Preauthorization

Preauthorization is obtained from the authorizer for the amount of the purchase. For credit card transactions, an approved preauthorization places a hold on the account holder's "open-to-buy" amount.

Before a merchant can start the process to collect the payment, the purchase must be "completed" by a corresponding PostAuthorization transaction.

1.1.2 Sale

Authorization and PostAuthorization is performed in a single step. For credit card transactions, an approved Auth places a hold on the account holder's "open-to-buy" balance and the purchase is immediately made ready to be settled if approved without any further action.

1.1.3 Postauthorization

A PostAuthorization request confirms that the purchase has been completed (ordered goods have been shipped) and is ready to be settled. The amount that is deposited must be less than or equal to the authorized amount.

1.1.4 Refund (Credit)

A refund transfers money from the merchant’s account to an account holder’s account. This kind of transaction is used to refund an account holder’s money for an order that was settled. Multiple partial refunds are supported. The total refund amount must not exceed the sum of the deposited and settled transactions associated with the order minus the amounts already refunded.

1.1.5 Void

The transaction is cancelled. Transactions of type PreAuthorization, PostAuthorization, sale and refund can be voided.

1.2 Order Query Services

If a network problem occurs while Payment Gateway is trying to send the response to the merchant, the response can’t be sent to the merchant. The response couldn't be recorded to the merchant databases, therefore there is nothing to be saved.
The problem is that there is a transaction in Payment Gateway and bank systems, but there is an unsuccessful transaction or even no transaction in the merchant systems.
The possible causes of the problems that may occur are listed below:
A late response from one of the network components may cause the other components to switch to time-out status, which causes the merchant response to be timed out.

- A network problem
- A problem with the merchant server
- Although the response is sent successfully from NestPay and received by the merchant system, the merchant system can’t read the message as it supposed to. Software updates, version updates or table updates may cause this problem.
- Some problem on the merchant servers and systems other than the merchant web server.
- General problems on country networks
- The maintenance works on the country networks.

To remedy such synchronization problems without letting them cause more serious risks, payment gateway offers some API services as back-ups.
These additional services for the merchants are:

• **Order Status Service:** The merchant can query the status of an order to see if it is completed successfully or not.

• **Order History Service:** The merchant can query the status history of an order; e.g. about whether there are any refunds related with this order, or whether the order or refund is successful.

2.XML API

2.1 CC5AS XML Format

2.1.1 How to Start XML Request

The name of the parameter has to be "DATA" when posting XML request and the value of it will be XML message.

DATA=<?xml version="1.0" encoding="UTF-8"?>

For the other programming languages, you may find this information in the sample codes.

2.1.2 Transaction Request

The complete CC5AS Transaction Request consists of the following **XML** elements within the root element **CC5Request**:

```
<CC5Request>
          <Name>User name</Name>
          <Password>User password</Password>
          <ClientId>Merchant number</ClientId>
          <Type>{Auth, PreAuth, PostAuth, Void, Credit}</Type>
          <IPAddress>IP address of the customer</IPAddress>
          <Email>Email of the customer</Email>
          <OrderId>Order Id</OrderId>
          <GroupId>Group Id</GroupId>
          <TransId>Transaction Id</TransId>
          <Total>Total Amount</Total>
          <Currency>Currency code</Currency>
          <Number>Card number</Number>
          <Expires>Card expiry<Expires>
          <Cvv2Val>CVV2 value of card</Cvv2Val>
          <Instalment>Installment count</Instalment>
          <PayerSecurityLevel>ECI</PayerSecurityLevel>
          <PayerTxnId>Internet transaction Id</PayerTxnId>
        <PayerAuthenticationCode>CAVV</PayerAuthenticationCode>
          <BillTo>
              <Name>BillTo customer name</Name>
              <Company>BillTo company name</Company>
              <Street1>BillTo address line 1</Street1>
```

```
                <Street2>BillTo address line 2</Street2>
                <Street3>BillTo address line 3</Street3>
                <City>BillTo city</City>
                <StateProv>BillTo state</StateProv>
                <PostalCode>BillTo postal code</PostalCode>
                <Country>BillTo country code</Country>
                <TelVoice>BillTo phone number</TelVoice>
                <TelFax>BillTo fax number</TelFax>
          </BillTo>
          <ShipTo>
                <Name> ShipTo customer name</Name>
                <Company> ShipTo company name</Company>
                <Street1> ShipTo address line 1</Street1>
                <Street2> ShipTo address line 2</Street2>
                <Street3> ShipTo address line 3</Street3>
                <City> ShipTo city</City>
                <StateProv> ShipTo state</StateProv>
                <PostalCode> ShipTo postal code</PostalCode>
                <Country> ShipTo country code</Country>
                <TelVoice> ShipTo phone number</TelVoice>
                <TelFax> ShipTo phone number</TelFax>
          </ShipTo>
          <OrderItemList>
               <OrderItem>
                      <ItemNumber>Item number</ItemNumber>
                      <ProductCode>Product code</ProductCode>
                      <Qty>Quantity</Qty>
                      <Desc>Description</Desc>
                      <Id>Item Id</Id>
                    <Price>Item unit price</Price>
                      <Total>Total price</Total>
                 </OrderItem>
          </OrderItemList>
   </CC5Request>
```

2.1.3 Transaction Request Tags

Tag	Definition	Format	Mandatory
Name	Username*	Alphanumeric, max 255 chars	YES

	credential		
Password	Password* credential	Alphanumeric, max 255 chars	YES
ClientId	Merchant Id	Alphanumeric, max 15 chars	YES
Type	Transaction type	Alphanumeric, accepted values {Auth, PreAuth, PostAuth, Void, Credit}	YES
IPAddress	IP address of the customer	Max 39 chars	NO
OrderId	Order Id	Alphanumeric, max 64 chars	
GroupId	Group Id	Alphanumeric, max 64 chars	NO
TransId	Transaction Id	Alphanumeric, max 64 chars	
Total	Total amount	Number, Use decimal separator "," or "." No grouping character	
Currency	ISO currency code	Numeric, 3 digits (949 for TR)	
UserId	User Id, for reporting	Numeric, max 64 digits	NO
Number	Card number	Alphanumeric + symbol	
Cvv2Val	CVV2 value	Numeric, 3 digits	
Expires	Card expiry	MM/YYYY	
Instalment	Instalment count	Numeric	NO
IPAddress	Cardholders IP address	Alphanumeric + symbol	NO
PayerSecurityLevel	ECI	Numeric, 2 digits	
PayerTxnId	Internet transaction Id	Alphanumeric + symbol, 28 characters, base64-encoded	
PayerAuthenticationCode	CAVV	Alphanumeric + symbol, 28 characters, base64-encoded	
BillTo.Name	BillTo customer name	Maximum 255 characters	NO
BillTo.Company	BillTo company name	Maximum 255 characters	NO

BillTo.Street1	BillTo address line 1	Maximum 255 characters	NO
BillTo.Street2	BillTo address line 2	Maximum 255 characters	NO
BillTo.Street3	BillTo address line 3	Maximum 255 characters	NO
BillTo.City	BillTo city	Maximum 64 characters	NO
BillTo.StateProv	BillTo state	Maximum 32 characters	NO
BillTo.PostalCode	BillTo postal code	Maximum 32 characters	NO
BillTo.Country	BillTo country code	Maximum 3 characters	NO
BillTo.TelVoice	BillTo phone number	Maximum 32 characters	NO
BillTo.TelFax	BillTo fax number	Maximum 32 characters	NO
ShipTo.Name	ShipTo customer name	Maximum 255 characters	NO
ShipTo.Company	ShipTo company name	Maximum 255 characters	NO
ShipTo.Street1	ShipTo address line 1	Maximum 255 characters	NO
ShipTo.Street2	ShipTo address line 2	Maximum 255 characters	NO
ShipTo.Street3	ShipTo address line 3	Maximum 255 characters	NO
ShipTo.City	ShipTo city	Maximum 64 characters	NO
ShipTo.StateProv	ShipTo state	Maximum 32 characters	NO
ShipTo.PostalCode	ShipTo postal code	Maximum 32 characters	NO
ShipTo.Country	BillTo country code	Maximum 3 characters	NO
ShipTo.TelVoice	BillTo phone number	Maximum 32 characters	NO
ShipTo.TelFax	BillTo fax number	Maximum 32 characters	NO
OrderItem.id	Id of item	Maximum 128 characters	NO
OrderItem.itemnumber	Item number	Maximum 128 characters	NO
OrderItem.productcode	Product code	Maximum 64 characters	NO
OrderItem.qty	Quantity	Maximum 32 characters	NO
OrderItem.desc	Description	Maximum 128 characters	NO

OrderItem.price	Price	Maximum 32 characters	NO
Extra.InvoiceNumber	Invoice Number	Maximum 15 characters	NO

The tags marked with **mandatory=YES** should be set for each transaction request.
The tags marked with **mandatory=NO** are optional and used for information only.

Other tags must be set depending on the transaction type. Please see transaction type specific request examples.
NOTE: The password of an API user does not expire. Passwords of other users expire every 3 months and can be changed by User Administration panel. The requests with an expired user will return an error.

2.1.4 Transaction Response

CC5AS Transaction Response for consists of the following XML elements within the root element CC5Response:

```
< CC5Response>
        <OrderId>Order Id</OrderId>
        <GroupId>Group Id</GroupId>
        <Response>{Approved, Declined, Error}</Response>
        <AuthCode>Preauthorization code</AuthCode>
        <HostRefNum>Host reference number</HostRefNum>
        <ProcReturnCode>Transaction status code</ProcReturnCode>
        <TransId>Transaction Id</TransId>
        <ErrMsg>Error message</ErrMsg>
        <Extra>
                <SETTLEID>Settlement Id</SETTLEID>
                <TRXDATE>Transaction date</TRXDATE>
                <ERRORCODE>Error Code</ERRORCODE>
                <HOSTMSG>Host message</HOSTMSG>
                <NUMCODE>End error code</NUMCODE>
                <TERMINALID>Terminal Id</TERMINALID>
                <MERCHANTID>Merchant Id</MERCHANTID>
        </Extra>
</ CC5Response>
```

2.1.5 Transaction Response Tags

Tag	Definition	Format
OrderId	Order Id	Alphanumeric, max 64 chars
GroupId	Group Id, ignore	Alphanumeric, max 64 chars
Response	Transaction Response	Possible values: "Approved" for successful transactions, "Declined" for declined transactions "Error" for gateway errors
AuthCode	Host preauthorization code	Alphanumeric, 6 chars
HostRefNum	Host reference number	Alphanumeric, 12 chars
ProcReturnCode	Transaction status code	Alphanumeric, 2 chars, "00" for authorized transactions, "99" for gateway errors, others for ISO-8583 error codes
TransId	Transaction Id	Alphanumeric, max 64 chars
ErrMsg	Error message (if any)	Alphanumeric, max 255 chars
Extra.SETTLEID	Settlement Id	Numeric, 3 digits
Extra.TRXDATE	Transaction date	Formatted as "yyyyMMdd HH:mm:ss"
Extra.ERRORCODE	Error code (if any)	Alphanumeric, max 16 chars
Extra.HOSTMSG	Card number	Alphanumeric, max 255 chars
Extra.NUMCODE	End Error code, generated by adding numeric portion of Extra.ERRORCODE to ProcReturnCode	Numeric, max 20 digits
Extra. AAVRESPONSECODE	AAV response code for Amex transaction	Alphanumeric, 1 char
Extra.CARDBRAND	Card Brand of Card	Alphanumeric, max 16 chars
Extra.TERMINALID	Terminal id	Alphanumeric, max 15 chars
Extra.MERCHANTID	Merchant id	Alphanumeric, max 15 chars
Extra.CARDISSUER	Issuer bank name of the card	Alphanumeric

2.2 Transaction Types

2.2.1 Preauthorization

To make a preauthorization request, set the *Type* field to "**PreAuth**". If *OrderId* element does not exist, the system will generate a unique OrderId and send the id back in the response message.

Example XML Request:

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <Type>PreAuth</Type>
        <Total>10.15</Total>
        <Currency>949</Currency>
        <Number>4242424242424242</Number>
        <Expires>10/2028</Expires>
        <Cvv2Val>123</Cvv2Val>
</CC5Request>
```

NOTE: For 3D-secure authorizations, *PayerAuthenticationCode*, *PayerTxnId*, and *PayerSecurityLevel* tags should be set instead of *Expires* and *Cvv2Val* tags. Please refer to the related payment model guide about how to set these tags.

2.2.2 Sale

To make a Sale transaction, set the *Type* field to "**Auth**". If *OrderId* element does not exist, the NestPay system will generate a unique OrderId and send it back in the response message.

Example XML Request :

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <Type>Auth</Type>
        <Total>10.15</Total>
        <Currency>949</Currency>
        <Number>4242424242424242</Number>
        <Expires>10/2028</Expires>
        <Cvv2Val>123</Cvv2Val>
</CC5Request>
```

Note: For 3D-secure Sale transactions, *PayerAuthenticationCode*, *PayerTxnId*, and *PayerSecurityLevel* tags should be set instead of *Expires* and *Cvv2Val* tags. Please refer to the related payment model guide about how to set these tags.

2.2.3 Sale with Instalment

To perform a sale transaction with instalment, the *Type* field should be set as "**Auth**" and the *Instalment* field should be set as the instalment count. To send the grace period, the extra field "*GRACEPERIOD*" can optionally be set.

Example XML Request :

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <Type>Auth</Type>
        <Total>10.15</Total>
        <Currency>949</Currency>
        <Number>4242424242424242</Number>
        <Expires>10/2028</Expires>
        <Cvv2Val>123</Cvv2Val>
        <Instalment>4</Instalment>
        <Extra><GRACEPERIOD>2</GRACEPERIOD></Extra>
</CC5Request>
```

2.2.4 Mailorder sale

To perform a sale transaction with MAILORDER flag, the *Type* field should be set as "**Auth**" and the mailorder indicator <MAILORDER> should be set in Extra field.

Example XML Request :

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <Type>Auth</Type>
        <Total>10.15</Total>
        <Currency>949</Currency>
        <Number>4242424242424242</Number>
        <Expires>10/2028</Expires>
        <Cvv2Val>123</Cvv2Val>
        <Extra><MAILORDER>MAILORDER</MAILORDER></Extra>
</CC5Request>
```

2.2.5 Postauthorization

To make a PostAuthorization request, set the *Type* field to “**PostAuth**”. OrderId needs to be set to indicate the order PostAuthorization is for. The PostAuthorization amount can be lower than the PreAuthorization amount.

Note: TransId is also can be used for PostAuthorization.

Example XML Request :

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <Type>PostAuth</Type>
        <OrderId>ORDER12345</OrderId>
</CC5Request>
```

To make a partial postauthorization set the **Total** tag to the partial refund amount.

2.2.6 Refund

To make a refund request, set the *Type* field to “**Credit**”. OrderId needs to be set to indicate the order Refund is for. Multiple partial refunds are supported as long as original sale amount is not exceeded.

Example XML Request :

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <Type>Credit</Type>
        <OrderId>ORDER12345</OrderId>
</CC5Request>
```

To make a partial refund set the Total tag to the partial refund amount

2.2.7 Void

To make a void request, set the Type field to “**Void**”. Either *OrderId* or *TransId* needs to be set to indicate the order the Void is for. If *TransId* is set, the transaction with the *TransId* will be voided. If *OrderId* is set, the successful transaction of the order will be searched for and voided. If there are multiple successful transactions for the order such as multiple refunds, the system will return an error.

Example XML Request :

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <Type>Void</Type>
        <OrderId>ORDER12345</OrderId>
</CC5Request>
```

2.3 Order Status Query

OrderStatus queries can be made for the transaction types *Auth*, *Void*, *Credit*, *PreAuth*, and *PostAuth*. No modifications are made on the gateway or the bank's system. Status of the order is returned. There may be more than one transaction for an order, in which case the response for the OrderStatus query returns the last successful transaction. If there is no successful transaction for the corresponding order, it returns the last unsuccessful transaction record.

2.3.1 Order Status Query For Non-Recurring Order

The order status query type is set within the **Extra.ORDERSATUS** tag:

```
<Extra>
    <ORDERSTATUS>QUERY</ORDERSTATUS>
</Extra>
```

Example XML Request :

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <OrderId>ORDER12345</OrderId>
        <Extra>
                 <ORDERSTATUS>QUERY</ORDERSTATUS>
        </Extra>
</CC5Request>
```

2.3.1.1 Order Status Query Response

Order Status Query Response consists of the following XML elements within the root element **CC5Response**. Order query response values can be returned in 2 ways:

- Within the Extra.ORDERSATUS tag of the response document as tab-separated name:value pairs
- Separate tags within the Extra tag

```
<CC5Response>
    <ErrMsg>Error message</ErrMsg>
    <ProcReturnCode>Transaction status code</ProcReturnCode>
    <Response>{Approved, Error}</Response>
    <OrderId>Order Id</OrderId>
    <TransId>Transaction Id</TransId>
    <Extra>
        <AUTH_DTTM>Preauthorization time</AUTH_DTTM>
        <HOSTDATE>Host date</HOSTDATE>
        <TRANS_STAT>Transaction status</TRANS_STAT>
        <ORDERSTATUS>ORD_ID:OrderId CHARGE_TYPE_CD:TransactionTtype
         ORIG_TRANS_AMT:FirstAmount CAPTURE_AMT:TransactionAmount
         TRANS_STAT:TransactionStatus AUTH_DTTM:AuthorizationTime
         CAPTURE_DTTM:DepositTime AUTH_CODE:118889
         TRANS_ID:TransactionId
      </ORDERSTATUS>
        <ORIG_TRANS_AMT>First amount</ORIG_TRANS_AMT>
        <PROC_RET_CD>Host return code</PROC_RET_CD>
        <CAPTURE_AMT>Transaction amount</CAPTURE_AMT>
        <HOST_REF_NUM>Host reference number</HOST_REF_NUM>
        <SETTLEID>Settlement Id</SETTLEID>
        <TRANS_ID>Transaction Id</TRANS_ID>
        <ORD_ID>Order Id</ORD_ID>
        <CHARGE_TYPE_CD>Transaction type</CHARGE_TYPE_CD>
        <AUTH_CODE>Host preauthorization code</AUTH_CODE>
        <NUMCODE>Number code</NUMCODE>
        <CAPTURE_DTTM>Postauthorization time</CAPTURE_DTTM>
        <XID_3D>Xid</XID_3D >
        <CAVV_3D>Cavv </CAVV_3D>
        <ECI_3D> Eci </ECI_3D>
        <MDSTATUS>Md Status </MDSTATUS>
    </Extra>
</CC5Response>
```

2.3.2 Order Status Query For Recurring Order

The order status query type is set within the **Extra.ORDERSATUS** tag. However, if the request is for a recurring operation, **Extra.RECURRINGID** is also needed.

```
<Extra>
    <RECURRINGID>15210MWwD180004</RECURRINGID>
    <ORDERSTATUS>QUERY</ORDERSTATUS>
</Extra>
```

Example XML Request :

```
<CC5Request>
    <Name>Erdem</Name>
    <Password>***</Password>
    <ClientId>700655008993</ClientId>
    <Extra>
        <RECURRINGID>15210MWwD180004</RECURRINGID>
        <ORDERSTATUS>QUERY</ORDERSTATUS>
    </Extra>
</CC5Request>
```

Example XML Response :

Response for Recurring Operations is separated in two ways :

- If planned process date of the recurring order has come and order is processed.
- If the recurring order is not processed due to an error, cancellation or future planned process date

Furthermore, since recurring operations might have more than one order, tags in the response is illustrated with an underscore and recurringNumber of order. For example, if an xml tag in the response is illustrated as **"<TRANS_STAT_2>PN</TRANS_STAT_2>",** this means Transaction Status for the second order in the recurring operation.
In the below example response, there are two orders belong to the Recurring operation. First Recurring Order is a Pending Order and the second order is processed and successfully completed order.

```
<?xml version="1.0" encoding="ISO-8859-9"?>
<CC5Response>
  <ErrMsg>Record(s) found for 15210MWwD180004</ErrMsg>
  <Extra>

    <RECURRINGCOUNT>2</RECURRINGCOUNT>
    <RECURRINGID>15210MWwD180004</RECURRINGID>



    <ORIG_TRANS_AMT_1>1001</ORIG_TRANS_AMT_1>
    <CHARGE_TYPE_CD_1>S</CHARGE_TYPE_CD_1>
    <ORDERSTATUS_1>ORD_ID:ORDER-15210MWwD180003       CHARGE_TYPE_CD:S
     ORIG_TRANS_AMT:1001     TRANS_STAT:PN      PLANNED_START_DTTM:2016-03-27
05:00:00.0</ORDERSTATUS_1>
    <ORD_ID_1>ORDER-15210MWwD180003</ORD_ID_1>
    <TRANS_STAT_1>PN</TRANS_STAT_1>
    <PAN_1>4242 42** **** 4242</PAN_1>
    <PLANNED_START_DTTM_1>2016-03-27 05:00:00.0</PLANNED_START_DTTM_1>




    <CAPTURE_AMT_2>1001</CAPTURE_AMT_2>
    <CAPTURE_DTTM_2>2015-07-29 15:31:00.78</CAPTURE_DTTM_2>
    <AUTH_DTTM_2>2015-07-29 15:31:00.78</AUTH_DTTM_2>
    <ORIG_TRANS_AMT_2>1001</ORIG_TRANS_AMT_2>
    <MDSTATUS_2></MDSTATUS_2>
    <TRANS_ID_2>15210MfAA180146</TRANS_ID_2>
    <PROC_RET_CD_2>00</PROC_RET_CD_2>
    <ECI_3D_2></ECI_3D_2>
    <HOST_REF_NUM_2>521000000043</HOST_REF_NUM_2>
    <CHARGE_TYPE_CD_2>S</CHARGE_TYPE_CD_2>
    <ORDERSTATUS_2>ORD_ID:ORDER-15210MWwD180003-2     CHARGE_TYPE_CD:S
     ORIG_TRANS_AMT:1001     CAPTURE_AMT:1001 TRANS_STAT:C
     AUTH_DTTM:2015-07-29 15:31:00.78     CAPTURE_DTTM:2015-07-29 15:31:00.78
     AUTH_CODE:P53293TRANS_ID:15210MfAA180146</ORDERSTATUS_2>
    <PAN_2>4242 42** **** 4242</PAN_2>
    <TRANS_STAT_2>C</TRANS_STAT_2>
    <AUTH_CODE_2>P53293</AUTH_CODE_2>
    <CAVV_3D_2></CAVV_3D_2>
    <SETTLEID_2></SETTLEID_2>
    <XID_3D_2></XID_3D_2>
    <ORD_ID_2>ORDER-15210MWwD180003-2</ORD_ID_2>
    <HOSTDATE_2>0729-123100</HOSTDATE_2>

    <NUMCODE>0</NUMCODE>
  </Extra>
</CC5Response>
```

Order Status Tag	**Definition**	**Format**
ORD_ID	Order Id	Alphanumeric, max 64 chars
CHARGE_TYPE_CD	Transaction Type	S: Auth/PreAuth/PostAuth C: Refund
ORIG_TRANS_AMT	Preauthorization Amount	Without decimal separator, precision is based on the smallest unit of money
CAPTURE_AMT	Postauthorization Amount	Without decimal separator, precision is based on the smallest unit of money
TRANS_STAT	Transaction Status	D : NOT Successful A : Preauthorization, not settled C : Capture, not Settled S : Deposited R : Reversal Required V : Voided PN: Pending NW: First Commit (the transaction is still processing, transaction not finalized)
AUTH_DTTM	Preauthorization date-time	Formatted as "yyyy-MM-dd HH:mm:ss.S"
CAPTURE_DTTM	Postauthorization date-	Formatted as "yyyy-MM-dd

	time	HH:mm:ss.S”
AUTH_CODE	Host preauthorization code	Alphanumeric, 6 chars
HOST_REF_NUM	Host reference number	Alphanumeric, 12 chars
PROC_RET_CD	Transaction status code	Alphanumeric, 2 chars, “00” for authorized transactions, “99” for gateway errors, others for ISO-8583 error codes
TRANS_ID	Transaction Id	Alphanumeric, max 64 chars
SETTLEID	Settlement Id	Numeric
XID_3D	3D Xid Value	Alphanumeric, max 64 chars
CAVV_3D	3D Cavv Value	Alphanumeric, max 64 chars
ECI_3D	3D Eci Value	Numeric, 2 digits
MDSTATUS	3D Md Status	Numeric, 2 digits
RECURRINGCOUNT	Order Count Belongs to the Recurring Operation	Numeric
RECURRINGID	Recurring Id	Alphanumeric, max 64 chars
PLANNED_START_DTTM	Planned Start date of an order which belongs to the Recuriing Operation	Formatted as “yyyy-MM-dd HH:mm:ss.S”

If there is no error for the order status query the ErrMsg will contain the text in the following format:

```
<ErrMsg>Record(s) found for (OrderId or RecurringId)</ErrMsg>
```

2.3.3 Transaction Status Table

If transaction status is a **transient** state, it is recommended to run the order status query every 5 minutes until status changes. The query interval can be adjusted according to business needs. For transactions with ‘NW’ status (transient state), the transaction is still processing and has to be queried to get the final status. For transactions with status ‘R’, it is recommended to query until end of day. For transactions with ‘PN’ status this duration can be longer (72 hours). Please see **Appendix A** transaction status transition diagram.

TRANS_STAT	CHARGE_TYPE_CD	Status	Description	Shipment possible?	Transient State?
D	S	Declined	Unsuccessful transaction	NO	NO
A	S	Approved	Successful transaction (preauthorization)	NO	NO
C	S	Approved	Successful transaction	YES	NO
S	S	Deposited	Settled transaction	YES	NO
PN	C/S	Pending	Transaction pending to be confirmed by the bank	NO	**YES**
V	S	Voided	Cancelled transaction	NO	NO
C or S	C	Credited	Payment refunded	NO	NO
R	S	Reversal	Reversal Required	NO	**YES**
ERR	C/S	Error	Errorred Recurring Order	NO	**NO**
CNCL	C/S	Cancelled	Calcelled Recurring Order	NO	**NO**
NW	C/S	FirstCommit	Transaction was initiated (the transaction is still processing, transaction not finalized)	NO	**YES**

2.4 Order History Query

The merchant can query the status history of an order to investigate, for instance, whether there are any refunds related with this order, or whether the order or refund is successful or not.

The order history query type is set within the **Extra. ORDERHISTORY** tag:

```
<Extra>
    <ORDERHISTORY>QUERY</ORDERHISTORY>
</Extra>
```

Example XML Request :

```
<CC5Request>
        <Name>testuser</Name>
        <Password>TEST1234</Password>
        <ClientId>990000001</ClientId>
        <OrderId>ORDER12345</OrderId>
        <Extra>
                <ORDERHISTORY>QUERY</ORDERHISTORY>
         </Extra>
</CC5Request>
```

2.4.1 Order History Query Response

Order History Query Response consists of the following XML elements within the root element **CC5Response**. Order history transactions are returned within the Extra tag. There will be a corresponding TRX*n* tag for the *n*-th transaction of the order.

```
<CC5Response>
       <ErrMsg>Error message</ErrMsg>
       <ProcReturnCode>Transaction status code</ProcReturnCode>
       <Response>{Approved, Error}</Response>
       <OrderId>Order Id</OrderId>
       <Extra>
          <TRX1>tab-separeted transaction line of first trx</TRX1>
          <TRXCOUNT>Transaction count</TRXCOUNT>
          <TRX2>tab-separeted transaction line of second trx</TRX2>
          <TRXn>tab-separeted transaction line of n-th trx</TRX2>
          <NUMCODE>0</NUMCODE>
          <TERMINALID>Terminal Id</TERMINALID>
```

```
            <MERCHANTID>Merchant Id</MERCHANTID>
        </Extra>
</CC5Response>
```

The details of the transaction in the TRX*n* tag are tab-separated and have the following format. Please refer to order status query tags for the definition of these fields.

```
CHARGE_TYPE_CD + TRANS_STAT  + ORIG_TRANS_AMT  + CAPTURE_AMT+AUTH_DTTM +
    CAPTURE_DTTM VOID_DTTM + HOST_REF_NUM + AUTH_CODE + PROC_RET_CD +
    TRANS_ID + SETTLEID + XID + CAVV + ECI + MD STATUS
```

3.Java Jpay API

JPAY is the cross-patform JAVA API which provides the virtual POS functionality.

3.1 JPAY API Installation

JDK version 1.3 or higher must be installed on your operating system. The submitted "jpay.jar" file should be copied and pasted into a directory which is in your class-path.

3.1.1 JPAY API Usage

1. Set necessary fields by calling setters.
2. Call ***processTransaction*** function.
3. Get the result with using getters.

Example Usage: Make a sale only with mandatory fields:

*** Port value must be written as **443**. *myjpay.processTransaction("host",* ***port****,"/fim/api")*

```
jpay myjpay = new jpay();
        myjpay.setName("apiuser");
        myjpay.setPassword("apipassword");
        myjpay.setClientId("990000000000001");
        myjpay.setOrderId("ORDER123");
        myjpay.setType("Auth");
        myjpay.setTotal("10.5");
        myjpay.setCurrency("949");
        myjpay.setNumber("4242424242424242");
        myjpay.setCvv2Val("000");
        myjpay.setExpires("10/2028");
        if (myjpay.processTransaction("host", 443,"/fim/api") > 0){
    // Transaction successful
        } else {
  System.out.println(myjpay.getErrMsg());
                                  }
```

3.1.2 Jpay API Request Getters

Jpay setter	Definition	Format	Mandatory
setName	Username* credential	Alphanumeric, max 255 chars	YES
setPassword	Password* credential	Alphanumeric, max 255 chars	YES
setClientId	Merchant Id	Alphanumeric, max 15 chars	YES
setType	Transaction type	Alphanumeric, accepted values {Auth, PreAuth, PostAuth, Void, Credit}	YES
setIPAddress	IP address of the card holder	Max 39 chars	NO
setOrderId	Order Id	Alphanumeric, max 64 chars	
setGroupId	Group Id	Alphanumeric, max 64 chars	NO
setTransId	Transaction Id	Alphanumeric, max 64 chars	
setTotal	Total amount	Number, Use decimal separator "," or "." No grouping character	
setCurrency	ISO currency code	Numeric, 3 digits (949 for TR)	
setUserId	User Id, for	Numeric, max 64 digits	NO

	reporting		
setNumber	Card number	Alphanumeric + symbol	
setCvv2Val	CVV2 value	Numeric, 3 digits	
setExpires	Card expiry	MM/YYYY	
setTaksit	Instalment count	Numeric	NO
setPayerSecurityLevel	ECI	Numeric, 2 digits	
setPayerTxnId	Internet transaction Id	Alphanumeric + symbol, 28 characters, base64-encoded	
setPayerAuthenticationCode	CAVV	Alphanumeric + symbol, 28 characters, base64-encoded	
setBName	BillTo customer name	Maximum 255 characters	NO
setBCompany	BillTo company name	Maximum 255 characters	NO
setBStreet1	BillTo address line 1	Maximum 255 characters	NO
setBStreet2	BillTo address line 2	Maximum 255 characters	NO
setBStreet3	BillTo address line 3	Maximum 255 characters	NO
setBCity	BillTo city	Maximum 64 characters	NO
setBStateProv	BillTo state	Maximum 32 characters	NO
setBPostalCode	BillTo postal code	Maximum 32 characters	NO
setBCountry	BillTo country code	Maximum 3 characters	NO
setBTelVoice	BillTo phone number	Maximum 32 characters	NO
setSName	ShipTo customer name	Maximum 255 characters	NO

setSCompany	ShipTo company name	Maximum 255 characters	NO
setSStreet1	ShipTo address line 1	Maximum 255 characters	NO
setSStreet2	ShipTo address line 2	Maximum 255 characters	NO
setSStreet3	ShipTo address line 3	Maximum 255 characters	NO
setSCity	ShipTo city	Maximum 64 characters	NO
setSStateProv	ShipTo state	Maximum 32 characters	NO
setSPostalCode	ShipTo postal code	Maximum 32 characters	NO
setSCountry	BillTo country code	Maximum 3 characters	NO
setSTelVoice	BillTo phone number	Maximum 32 characters	NO
SetOrderItem(ItemNumber, ProductCode, Qty,Desc,Id,Price,Total)	Order item		NO

The tags marked with **mandatory=YES** needs to set for each transaction request.

The tags marked with **mandatory=NO** are optional and used for information only.
Other tags must be set depending on the transaction type. Please see transaction type specific request examples.
NOTE: The password of an API user does not expire. Passwords of other users expire every 3 months and can be changed by user administration panel. **The requests with an expired user will return an error.**

3.1.3 Jpay API Response Getters

Jpay getter	**Definition**	**Format**
getOrderId	Order Id	Alphanumeric, max 64 chars
getGroupId	Group Id, ignore	Alphanumeric, max 64 chars
getResponse	Transaction Response	Possible values: “Approved” for successful transactions,

		"Declined" for declined transactions "Error" for gateway errors
getAuthCode	Host preauthorization code	Alphanumeric, 6 chars
getHostRefNum	Host reference number	Alphanumeric, 12 chars
getProcReturnCode	Transaction status code	Alphanumeric, 2 chars, "00" for authorized transactions, "99" for gateway errors, others for ISO-8583 error codes
getTransId	Transaction Id	Alphanumeric, max 64 chars
getErrMsg	Error message (if any)	Alphanumeric, max 255 chars
getExtra("SETTLEID")	Settlement Id	Numeric, 3 digits
getExtra("TRXDATE")	Transaction date	Formatted as "yyyyMMdd HH:mm:ss"
getExtra("ERRORCODE")	Error code (if any)	Alphanumeric, max 16 chars
getExtra("HOSTMSG")	Card number	Alphanumeric, max 255 chars
getExtra("NUMCODE")	End Error code, generated by adding numeric portion of Extra.ERRORCODE to ProcReturnCode	Numeric, max 20 digits

3.1.4 Order Status Query Response

Jpay getter	**Definition**	**Format**
getExtra("ORD_ID")	Order Id	Alphanumeric, max 64 chars
getExtra("CHARGE_TYPE_CD")	Transaction Type	S: Auth/PreAuth/PostAuth C: Refund
getExtra("ORIG_TRANS_AMT")	Preauthorization Amount	Without decimal separator, precision is based on the smallest unit of money
getExtra("CAPTURE_AMT")	Postauthorization Amount	Without decimal separator, precision is based on the smallest unit of money

getExtra("TRANS_STAT")	Transaction Status	D : NOT Successful A : Preauthorization, not settled C : Capture, not Settled S : Deposited R : Reversal Required V : Voided
getExtra("AUTH_DTTM")	Preauthorization date-time	Formatted as "yyyy-MM-dd HH:mm:ss.S"
getExtra("CAPTURE_DTTM")	Postauthorization date-time	Formatted as "yyyy-MM-dd HH:mm:ss.S"
getExtra("AUTH_CODE")	Host preauthorization code	Alphanumeric, 6 chars
getExtra("HOST_REF_NUM")	Host reference number	Alphanumeric, 12 chars
getExtra("PROC_RET_CD")	Transaction status code	Alphanumeric, 2 chars, "00" for authorized transactions, "99" for gateway errors, others for ISO-8583 error codes
getExtra("TRANS_ID")	Transaction Id	Alphanumeric, max 64 chars
getExtra("SETTLEID")	Settlement Id	Numeric

4.DLL API

The epayapi.dll is a self-registering Windows DLL which provides an API to the virtual POS functionality in Windows Operating System. With **"epayapi.dll "**, it is possible to call the API functions from any language and from scripts such as Visual Basic Script.

4.1 DLL API Installation

- Internet Information Services (ISS) should be stopped before Epayapi.dll defined
- Once IIS is stopped then, Start>>Run and type " regsvr32 epayapi.dll " and press OK.
- Epayapi.dll must be saved where it would not be deleted, such as in ***C:\WINDOWS\system32***.
- Definition screen pops up with successful completion. Press OK.

- Once definition process is complete then rerstart IIS

4.2 DLL API Usage

Once epayapi.dll is set up, it can be retrieved via CreateObject function. For example;

```
Set pay= CreateObject ("epayapi.payment")
```

After creating the object in ASP, Visual Basic, or C++, transaction parameters should be set. After ProcessOrder() method is called, the result determines whether the amount is refunded or not. If the result is "1", it means that the connection with the bank was established properly and the transaction was successful. If the result is "0", it usually means that the connection with the bank could not be established properly.

Example Usage: Create a "Sale" transaction using only mandatory fields:

```
set myPay = Server.createObject("epayapi.payment")
        myPay.host = "host.com.pl"
        myPay.name = "apiuser"
        myPay.password = "TEST1234"
        myPay.clientid = "990000001"
        myPay.oid = "ORDER-123"
        myPay.orderresult = 0
        myPay.chargetype = "Auth"
        myPay.currency = "949"
        myPay.cardnumber = "4242424242424242"
        myPay.expmonth = "12"
        myPay.expyear = "12"
        myPay.cv2 = "000"
        myPay.subtotal = "10"
        result = myPay.processorder
```

4.2.1 DLL API Request Fields

Field	Definition	Format	Mandatory
name	Username* credential	Alphanumeric, max 255 chars	YES
password	Password* credential	Alphanumeric, max 255 chars	YES

clientid	Merchant Id	Alphanumeric, max 15 chars	YES
chargetype	Transaction type	Alphanumeric, accepted values {Auth, PreAuth, PostAuth, Void, Credit}	YES
ip	IP address of the customer	Max 39 chars	NO
oid	Order Id	Alphanumeric, max 64 chars	
groupid	Group Id	Alphanumeric, max 64 chars	
subtotal	Total amount	Number, Use decimal separator "," or "." No grouping character	
currency	ISO currency code	Numeric, 3 digits (949 for TR)	
setUserId	User Id, for reporting	Numeric, max 64 digits	NO
cardnumber	Card number	Alphanumeric + symbol	
cv2	CVV2 value	Numeric, 3 digits	
expmonth	Card expiry month	MM	
expyear	Card expiry year	YY	
taksit	Instalment count	Numeric	NO
payersecuritylevel	ECI	Numeric, 2 digits	
payertxnid	Internet transaction Id	Alphanumeric + symbol, 28 characters, base64-encoded	
payerauthenticationcode	CAVV	Alphanumeric + symbol, 28 characters, base64-	

		encoded	
bname	BillTo customer name	Maximum 255 characters	NO
baddr1	BillTo address line 1	Maximum 255 characters	NO
baddr2	BillTo address line 2	Maximum 255 characters	NO
baddr3	BillTo address line 3	Maximum 255 characters	NO
bcity	BillTo city	Maximum 64 characters	NO
bstate	BillTo state	Maximum 32 characters	NO
bzip	BillTo postal code	Maximum 32 characters	NO
bcountry	BillTo country code	Maximum 3 characters	NO
phone	BillTo phone number	Maximum 32 characters	NO
sname	ShipTo customer name	Maximum 255 characters	NO
saddr1	ShipTo address line 1	Maximum 255 characters	NO
saddr2	ShipTo address line 2	Maximum 255 characters	NO
saddr3	ShipTo address line 3	Maximum 255 characters	NO
scity	ShipTo city	Maximum 64 characters	NO
sstate	ShipTo state	Maximum 32 characters	NO
szip	ShipTo postal code	Maximum 32 characters	NO
scountry	BillTo country code	Maximum 3 characters	NO
id	Id of item	Maximum 128 characters	NO
itemNumber	Item number	Maximum 128 characters	NO

productCode	Product code	Maximum 64 characters	NO
quantity	Quantity	Maximum 32 characters	NO
desc	Description	Maximum 128 characters	NO
price	Price	Maximum 32 characters	NO

The tags marked with **mandatory=YES** should be set for each transaction request.
The tags marked with **mandatory=NO** are optional and used for information only.
Other tags must be set depending on the transaction type. Please see transaction type specific request examples.
NOTE: The password of an API user does not expire. Passwords of other users expire every 3 months and can be changed by user administration panel. **The requests with an expired user will return an error.**

4.2.2 DLL API Response Fields

Field	**Definition**	**Format**
oid	Order Id	Alphanumeric, max 64 chars
groupid	Group Id, ignore	Alphanumeric, max 64 chars
appr	Transaction Response	Possible values: “Approved” for successful transactions, “Declined” for declined transactions “Error” for gateway errors
code	Host preauthorization code	Alphanumeric, 6 chars
refno	Host reference number	Alphanumeric, 12 chars
err	Transaction status code	Alphanumeric, 2 chars, “00” for authorized transactions, “99” for gateway errors, others for ISO-8583 error codes
transid	Transaction Id	Alphanumeric, max 64 chars
errmsg	Error message (if any)	Alphanumeric, max 255 chars
extra(“SETTLEID”)	Settlement Id	Numeric, 3 digits

extra("TRXDATE")	Transaction date	Formatted as "yyyyMMdd HH:mm:ss"
extra("ERRORCODE")	Error code (if any)	Alphanumeric, max 16 chars
extra("HOSTMSG")	Card number	Alphanumeric, max 255 chars
extra("NUMCODE")	End Error code, generated by adding numeric portion of Extra.ERRORCODE to ProcReturnCode	Numeric, max 20 digits

5.NET API

.NET API provides the virtual POS functionality for the .NET Framework.

5.1 .NET API Installation

Create bin directory under the .NET, and copy the epayment.dll into that directory.

5.2 .NET API Usage

The epayment object named ***dim mycc5pay*** should be created by calling the method ***new ePayment.cc5payment()***.

- The fields should be filled using the set functions.
- The payment should be sent for processing by calling ***mycc5pay.processorder()*** function.
- The result can be retrieved using get functions.

NOTE: Port value must be given as "**443**".
e.g. mycc5pay.port = "**443**"

```
<script language="VB" runat="server">
Sub Page_Load(Sender As Object, E As EventArgs)
dim mycc5pay as new ePayment.cc5payment()
    mycc5pay.host="https://host/fim/api"
    mycc5pay.name="apiuser"
    mycc5pay.password="TEST1234"
    mycc5pay.clientid="990000001"
    mycc5pay.orderresult="0"
    mycc5pay.oid="ORDER-123"
    mycc5pay.cardnumber = "4242424242424242"
    mycc5pay.expmonth = "12"
    mycc5pay.expyear = "12"
    mycc5pay.cv2 = "000"
    mycc5pay.subtotal = 10
    mycc5pay.currency = 949
    mycc5pay.chargetype = "Auth"
      mycc5pay.port = "443"
        Result1.Text= mycc5pay.processorder()
        Procreturncode.Text = mycc5pay.procreturncode
        ErrMsg.Text = mycc5pay.errmsg
        Oid1.Text = mycc5pay.oid
        appr1.Text = mycc5pay.appr
End Sub
</script>
```

5.2.1 .NET API Request Fields

Field	**Definition**	**Format**	**Mandatory**
name	Username* credential	Alphanumeric, max 255 chars	YES
password	Password* credential	Alphanumeric, max 255 chars	YES
clientid	Merchant Id	Alphanumeric, max 15 chars	YES
chargetype	Transaction type	Alphanumeric, accepted values {Auth, PreAuth, PostAuth, Void, Credit}	YES
ip	IP address of the customer	Max 39 chars	NO
oid	Order Id	Alphanumeric, max 64 chars	

groupid	Group Id	Alphanumeric, max 64 chars	
subtotal	Total amount	Number, Use decimal separator "," or "." No grouping character	
currency	ISO currency code	Numeric, 3 digits (949 for TR)	
setUserId	User Id, for reporting	Numeric, max 64 digits	NO
cardnumber	Card number	Alphanumeric + symbol	
cv2	CVV2 value	Numeric, 3 digits	
expmonth	Card expiry month	MM	
expyear	Card expiry year	MM	
taksit	Instalment count	Numeric	NO
payersecuritylevel	ECI	Numeric, 2 digits	
payertxnid	Internet transaction Id	Alphanumeric + symbol, 28 characters, base64-encoded	
payerauthenticationcode	CAVV	Alphanumeric + symbol, 28 characters, base64-encoded	
bname	BillTo customer name	Maximum 255 characters	NO
baddr1	BillTo address line 1	Maximum 255 characters	NO
baddr2	BillTo address line 2	Maximum 255 characters	NO
baddr3	BillTo address line 3	Maximum 255 characters	NO
bcity	BillTo city	Maximum 64	NO

		characters	
bstate	BillTo state	Maximum 32 characters	NO
bzip	BillTo postal code	Maximum 32 characters	NO
bcountry	BillTo country code	Maximum 3 characters	NO
phone	BillTo phone number	Maximum 32 characters	NO
sname	ShipTo customer name	Maximum 255 characters	NO
saddr1	ShipTo address line 1	Maximum 255 characters	NO
saddr2	ShipTo address line 2	Maximum 255 characters	NO
saddr3	ShipTo address line 3	Maximum 255 characters	NO
scity	ShipTo city	Maximum 64 characters	NO
sstate	ShipTo state	Maximum 32 characters	NO
szip	ShipTo postal code	Maximum 32 characters	NO
scountry	BillTo country code	Maximum 3 characters	NO
additem(ItemNumber, ProductCode, Qty, Desc, Id, Price, Total)	Order item		NO

The tags marked with **mandatory=YES** should be set for each transaction request.

The tags marked with **mandatory=NO** are optional and used for information only.

Other tags must be set depending on the transaction type. Please see transaction type specific request examples.

NOTE: The password of an API user does not expire. Passwords of other users expire every 3 months and can be changed by user administration panel. **The requests with an expired user will return an error.**

5.2.2 .NET API Response Fields

Field	**Definition**	**Format**
oid	Order Id	Alphanumeric, max 64 chars
groupid	Group Id, ignore	Alphanumeric, max 64 chars
appr	Transaction Response	Possible values: “Approved” for successful transactions, “Declined” for declined transactions “Error” for gateway errors
code	Host preauthorization code	Alphanumeric, 6 chars
refno	Host reference number	Alphanumeric, 12 chars
err	Transaction status code	Alphanumeric, 2 chars, “00” for authorized transactions, “99” for gateway errors, others for ISO-8583 error codes
transid	Transaction Id	Alphanumeric, max 64 chars
errmsg	Error message (if any)	Alphanumeric, max 255 chars
extra(“SETTLEID”)	Settlement Id	Numeric, 3 digits
extra(“TRXDATE”)	Transaction date	Formatted as “yyyyMMdd HH:mm:ss”
extra(“ERRORCODE”)	Error code (if any)	Alphanumeric, max 16 chars
extra("HOSTMSG")	Card number	Alphanumeric, max 255 chars
extra(“NUMCODE”)	End Error code, generated by adding numeric portion of Extra.ERRORCODE to ProcReturnCode	Numeric, max 20 digits

6.Recurring Payment

Recurring payments can be defined with Authorization requests. Each recurring payment contains information about first initial payment, recurring interval, order frequency and number of instalments.

Recurring payments is useful when implementing subscription-based payments.

6.1 Usage

Recurring payments can be defined in XML requests with <PbOrder> tags.

Sample XML request:

```
<CC5Request>
        <Name>FINTESTAPI</Name>
        <Password>***</Password>
        <ClientId>600100000</ClientId>
        <IPAddress>1.1.1.1</IPAddress>
        <OrderId></OrderId>
        <Type>Auth</Type>
        <Number>424242***4242</Number>
        <Expires>***</Expires>
        <Cvv2Val>***</Cvv2Val>
        <Total>180</Total>
        <Currency>949</Currency>
        <PbOrder>
            <OrderType>0</OrderType>
            <TotalNumberPayments>3</TotalNumberPayments>
            <OrderFrequencyCycle>M</OrderFrequencyCycle>
            <OrderFrequencyInterval>1</OrderFrequencyInterval>
        </PbOrder>
        <VersionInfo>EPAYAPI-1.2.0.32</VersionInfo>
        <BillTo>
            <Name></Name>
        </BillTo>
        <ShipTo>
            <Name></Name>
        </ShipTo>
        <Extra></Extra>
</CC5Request>
```

Parameter Name	Explanation	Values
OrderType	Defines if there's instalment in recurring payments.	0: Defult, no-instalment 1: instalment exists
TotalNumberPayments	Defines instalment count. Valid if OrderType=1	Number
OrderFrequencyCycle	Defines unit type of OrderFrequencyInterval parameter	D: Days W: Weeks M: Months
OrderFrequencyInterval	Defines interval value	Number

6.2 Code Samples for Recurring

Sample declaration for a total of 3 payments which occur monthly, with no-instalment in different API's:

6.2.1 XML API Sample

```
<PbOrder>
     <OrderType>0</OrderType>
     <TotalNumberPayments>3</TotalNumberPayments>
     <OrderFrequencyCycle>M</OrderFrequencyCycle>
     <OrderFrequencyInterval>1</OrderFrequencyInterval>
</PbOrder>
```

6.2.2 JAVA API Sample

NOTE: Port value must be given as “**443”**

e.g. *myjpay.processTransaction("host",* ***port****,"/fim/api")*

```
jpay myjpay = new jpay();
          myjpay.setName(“apiuser”);
           myjpay.setPassword(“apipassword”);
           myjpay.setClientId(“990000000000001”);
           myjpay.setOrderId(“ORDER123
           myjpay.setType(“Auth”);
           myjpay.setTotal("10.5");
           myjpay.setCurrency("949");
           myjpay.setNumber(“4242424242424242”);
```

```
            myjpay.setCvv2Val("000");
            myjpay.setExpires("10/2028");
            myjpay.setPbOrder("0", "5", "M", "17");

   if (myjpay.processTransaction("host", 443,"/fim/api") > 0){
  // Transaction successful
    } else {
System.out.println(myjpay.getErrMsg());
          }
```

6.2.3 DLL API Sample

```
ePayment.cc5payment paymentObject = new cc5payment();
         paymentObject.host = txtHost.Text;
         paymentObject.clientid = txtClientId.Text;
         paymentObject.name = txtName.Text;
         paymentObject.password = txtPassword.Text;
         paymentObject.ip = String.Empty;
         paymentObject.cardnumber = txtCardNumber.Text;
         paymentObject.expmonth = txtExpMonth.Text;
         paymentObject.expyear = txtExpYear.Text;
         paymentObject.cv2 = txtCV2.Text;
         paymentObject.subtotal = txtAmount.Text;
         paymentObject.currency = txtCurrencyCode.Text;
         paymentObject.oid = txtOrderId.Text;
         paymentObject.addpborder("0", "5", "M", "1");
         paymentObject.taksit = txtTaksit.Text;
         result = paymentObject.processorder();
```

6.3 Recurring Error Codes

Code	Error Message	Description
CORE-1007	Invalid value for 'OrderType' for 'PbOrder'. Please check API manuals.	OrderType value is not entered or a value other than "0" is entered.
CORE-2020	The recurring period unit is missing or empty.	OrderFrequencyCycle value is not entered.

CORE-2021	The recurring period unit is not valid.	OrderFrequencyCycle value other these "M", "D" or "Y" is entered.
CORE-2022	The recurring period is not valid.	OrderFrequencyInterval value is not entered or a value is bigger then "99".
CORE-2023	The recurring duration is not valid.	TotalNumberPayments value is not entered.
CORE-2024	Only Sale (Auth) transaction have recurring payment.	Only Sale (Auth) transaction have recurring payment.
CORE-2029	Recurring or futurerequest is not allowed to plan for long term.	TotalNumberPayments value is bigger then "121".
CORE-2034	Recurring payment cannot be instalment sale.	Recurring payment cannot be instalment sale.

7.Modifying Recurring Orders & Future Requests

This modification process supports three functionalities:

1) Order Cancellation
2) Modification of Order Amount
3) Modification of Order Planned Start Date

7.1.1 API Example

Order Cancel :

```
<?xml version="1.0" encoding="UTF-8"?>
<CC5Request>
   <Name>User</Name>
   <Password>****</Password>
   <ClientId>700656522091</ClientId>
   <Extra>
        <RECURRINGOPERATION>Cancel</RECURRINGOPERATION>
        <RECORDTYPE>Recurring / Order </RECORDTYPE>
        <RECORDID>13039MDLA10003 / ORDER-13037PI3A10002 </RECORDID>
        <RECORDID>13039MDLA10004 / ORDER-13037PI3A10002 </RECORDID>
   </Extra>
</CC5Request>
```

As seen above, **Cancel** API request can be called for two types of records.

1) For Recurring Orders : API cancels all orders belong to the same recurring order by

using **Record Type** as *Recurring*, and **Record Id** as *Recurring ID*

2) For Orders: API cancels only the orders whose order id is given in **Record ID** section if Record Type is selected as **Order**.

Furthermore, API also supports modification of more than one record which have the same Record Type by querying more than one Record Id as shown above.

Modification Of Order Amount :

```
<?xml version="1.0" encoding="UTF-8"?>
<CC5Request>
   <Name>User</Name>
   <Password>****</Password>
   <ClientId>700656522091</ClientId>
   <Currency>949</Currency>
   <Extra>
        <RECURRINGOPERATION>Update</RECURRINGOPERATION>
        <RECORDTYPE>Recurring / Order </RECORDTYPE>
        <RECORDID>13039MDLA10003 / ORDER-13037Pl3A10002 </RECORDID>
        <RECORDID>13039MDLA10004 / ORDER-13037Pl3A10002 </RECORDID>
        <AMOUNT>1000.00</AMOUNT>
    </Extra>
</CC5Request>
```

Amount Modification of any recurring order or future request can be done by calling the API Request as seen above.

For changing the amount of orders, API can be called for two types of records.

1) For Recurring Orders : API modifies amounts of all orders belong to the same recurring order by using **Record Type** as *Recurring*, and **Record Id** as *Recurring ID.*
2) For Orders: API modifies amounts of the orders whose order id is given in **Record ID** section if Record Type is selected as **Order**.

Furthermore, API also supports modification of more than one record which have the same Record Type by querying more than one Record Id as shown above.

NOTE: It is mandatory to specify Currency Code in Amount Modification Requests.

Modification of Order Planned Start Date :

Modification of planned process dates of any recurring order or future request can be done by calling the API Request below.

```
<?xml version="1.0" encoding="UTF-8"?>
<CC5Request>
   <Name>User</Name>
   <Password>****</Password>
   <ClientId>700656522091</ClientId>
   <Extra>
        <RECURRINGOPERATION>Update</RECURRINGOPERATION>
        <RECORDTYPE>Recurring / Order </RECORDTYPE>
        <RECORDID>13039MDLA10003 / ORDER-13037Pl3A10002 </RECORDID>
         <RECORDID>13039MDLA10004 / ORDER-13037Pl3A10002 </RECORDID>
        <STARTDATE>2013-10-04</STARTDATE>
    </Extra>
</CC5Request>
```

As seen above, for changing start date of orders, API can be called for two types of records.

1) For Recurring Orders : API modifies planned start dates of all orders belong to same recurring order by using **Record Type** as *Recurring*, and **Record Id** as *Recurring ID*
2) For Orders: API modifies planned start dates of the orders whose order id is given in **Record ID** section if Record Type is selected as **Order**.

Furthermore, API also supports modification of more than one record which have the same Record Type by querying more than one Record Id as shown above.

In addition, it is also possible to modify both amount and planned start date of recurring orders and future requests as below.

```
<?xml version="1.0" encoding="UTF-8"?>
<CC5Request>
   <Name>User</Name>
   <Password>****</Password>
   <ClientId>700656522091</ClientId>
   <Currency>949</Currency>
   <Extra>
        <RECURRINGOPERATION>Update</RECURRINGOPERATION>
        <RECORDTYPE>Recurring / Order </RECORDTYPE>
        <RECORDID>13039MDLA10003 / ORDER-13037Pl3A10002 </RECORDID>
        <RECORDID>13039MDLA10004 / ORDER-13037Pl3A10002 </RECORDID>
        <AMOUNT>1000.00</AMOUNT>
         <STARTDATE>2013-10-04</STARTDATE>
    </Extra>
</CC5Request>
```

7.2 Recurring & Future Request Modification Error Codes

Error Code	Error Message
CORE-5102	In order to change recurring orders, obligatory parameters in the query must not be empty.
CORE-5103	Query, which is sent to modify Recurring Orders, has wrong or missing parameters.
CORE-5104	There is no parameter for update process.
CORE-5105	User Permission is needed to modify recurring orders.
CORE-5106	General Error in modification of Recurring Records.

8.Error Codes Descriptions and Solutions

Error codes and possible solutions are described below.

Error codes:

- Core Errors
- MPI Errors
- 3D Gateway Errors (Applicable only for internet integration)
- BM Errors
- ISO8583 Errors

8.1 CORE Errors

Core errors are initiated by NestPay. The following list includes core possible error descriptions and solutions.

There are two types of Core errors consist of 2 parts:

- Common Errors
- Specific Errors (Insufficient Permissions)

8.1.1 Common Errors

Code	Description	Solution
1001	General initialization error	For detailed information please contact your administrator.
1002	System error. First commit phase general exception.	For detailed information please contact your administrator.
1003	It is caused by Acquirer System	For detailed information please contact your administrator.
1004	System error. Response parameters general	For detailed information please contact

	exception.	your administrator.
1005	System error. Last commit phase general exception.	For detailed information please contact your administrator.
1007	Invalid value for 'OrderType' for 'PbOrder'. Please check API manuals.	Please check the length of PbOrder field and send again.
1010	'Currency' is unparsable value.	Please check the Currency field and try again.
1011	'User Name' can not be null or empty.	Please check the username field and make sure that it is not null and try again.
1012	'User Name' field size is out of limit.	Please check the username field and make sure that it has less than 255 characters.
1013	'Merchant Id' can not be null or empty.	You should give your clientId. If you don't know your ID, Please call your administrator.
1014	'Merchant Id' field size is out of limit.	Your merchantID can be maximum 15 characters. Please check and try again
1015	'Order Id' field size is out of limit.	OrderId is alphanumeric and it can be maximum 64 characters. Please check and try again.
1016	'Criteria' field size is out of limit.	Criteria field can be maximum 64 characters. Please check and try again.
1017	'Transaction Id' field size is out of limit.	TransactionId can be maximum 64 characters. Please check and try again.
1018	Total Amount is out of length	Total amount should include less than 18 digits. Please check and try again.
1019	Currency Code is wrong	Please check your currency code and try again. (eg. PLN = 985)
1020	'Api Version' field size is out of limit.	ApiVersion can be maximum 32 characters. Please check and try again.
1021	'Description' field size is out of limit.	Description can be maximum 255 characters. Please check and try again.

1022	'Consumer IP' field size is out of limit.	ConsumerIP can be maximum 39 characters. Please check and try again.
1023	'Installments' field size is out of limit.	Installment can be maximum 3 characters. Please check and try again.
1024	'Amount' can not be negative.	Please check that total amount is not less than 0.
1025	'Points' can not be negative.	Please check that points field is not less than 0.
1026	'Instalment' can not be negative.	Please check that installment is not less than 0.
1027	'Amount' must be equal to or greater than 'Points'.	The amount must be more than points. Please check and try again.
1028	'Transaction Id' must be used only for void request.	Do not transactionId except void transaction. For detailed information please call your administrator.
1029	Instalment should be sent for this query.	You can not send installment field null. Please check and try again.
2001	This is an invalid transaction type. Auth, PreAuth, PostAuth, Credit, Void are valid.	Check the transaction type and make sure you send the correct type.
2008	There is no transaction available to cancel	Make sure that the transaction is convenient to cancel
2009	Zero (0) is not valid amount for sale and preauth transaction.	You should give more than 0 amount for sale or preauth transaction. Please check your amount.
2010	Card Expiry Date is wrong	Check the expiry date and make sure that it is valid
2011	Card Expiry Date is not in valid format	Card expiry date should be in MM/YY format
2012	PAN Number is invalid.	1 - The card number should include at least 13 digits 2 – It should be validated by bank. A randomly produced card number can not pass the validation even it has effective length.

2013	Invalid settlement request detected.	Request xml is incorrect. Please, check the request xml and try again.
2014	Invalid query request detected.	Request xml is incorrect. Please, check the request xml and try again.
2015	The credit card number is missing or empty.	The credit card number is empty. Please enter your credit card number and try again.
2020	The recurring period unit is missing or empty.	Recurring period unit should be given. It can not be null. Please check and try again.
2021	The recurring period unit is not valid.	Recurring period unit should be in correct format. Please check recurring period unit and try again.
2022	The recurring period is not valid.	Recurring period can not be less than 0. Please check recurring period and try again.
2023	The recurring duration is not valid.	Recurring duration can not be null and it must be a natural number. Please check and try again.
2024	Only sale orders can have recurring.	Recurring transactions have to be sale. Please check that your transaction is sale and try again.
2201	The user is not authenticated.	Check your user id and password and make sure that you are really authenticated
2202	User has not permission to do this operation	The operation you are trying to make need a permission. So please make sure that you have this permission.
2254	IP restriction	Probably your IP settings are not defined. Please contact your administrator.
2506	Cannot post auth on zero net amount	The transaction amount is sent zero. Please give the amount more than 0
2507	The order-number is duplicated.	The order number you sent is already

		used before. Please give another one.

8.1.2 Specific Errors

8.1.2.1 Insufficient Permissions

Code	**Description**	**Solution**
2201	User is not authenticated to perform this process	The user is not authenticated. Please authenticate and try again.
2202	User does not have the permission	User does not have the permission to do this process. Please call your administrator.
2203	Simultaneous use of system user id and login name	Please use user id or login name. It is not possible to use both of them at the same time.

8.2 MPI Errors

There are two options for comprehending MPI errors.

- The administrator should check the "**returnform**" part of the log which contains the response from MPI. On *returnform* part, "**mdErrorMsg**" input contains the error description.
- The administrator can reach MPI errors from Control Center – Administrator MPI.

Best way to get experience and specialize in MPI errors is to create a variety of test cases that return different errors and compare these errors from the log to check if error logs are consistent with test cases. Common errors and their solution are described below.

8.2.1 Database Error

Administrators should fix the database problem and restart MPI module.

8.2.2 Merchant Not Participating

Sometimes it takes longer than expected to activate the merchant on Visa/ Mastercard site. Either consult the Visa/Mastercard if the merchant definitions are properly loaded or resend Visa/Mastercard excel files.

```
<form name="returnform" action="https://merchant.com/modules/estpay/ok.php"
   method="POST">
    <input type="hidden" name="PAResSyntaxOK" value="false">
    <input type="hidden" name="PAResVerified" value="false">
    <input type="hidden" name="version" value="2.0">
    <input type="hidden" name="merchantID" value="900000175">
    <input type="hidden" name="xid" value="RUlSOTM4NDIyOVhYMjY0ODg2NDg=">
    <input type="hidden" name="mdStatus" value="6">
    <input type="hidden" name="mdErrorMsg" value="Error with Directory Server
    (https://ds.visa3dsecure.com,https://dsw.visa3dsecure.com) response: ERROR
    message:Merchant not participating">
    <input type="hidden" name="txstatus" value="U">
    <input type="hidden" name="iReqCode" value="">
    <input type="hidden" name="iReqDetail" value="">
    <input type="hidden" name="vendorCode" value="">
    <input type="hidden" name="eci" value="">
    <input type="hidden" name="cavv" value="">
    <input type="hidden" name="cavvAlgorithm" value="">
    <input type="hidden" name="md"
   value="402275:DC7544C55C252F19B0744B246BB671B7895EE02052A5BFDBBC0C4FE423
   27CD52:3803
     :#">
    <input type="hidden" name="digest" value="dZBgADB2YmDiA6eo7G/n1W2Qz+E=">
    <input type="hidden" name="sID" value="1">
```

8.2.3 Merchant Password Error

To solve this problem, double checking merchant visa password or resending Visa excel file will be useful.

```
<form name="returnform" action="https://195.95.188.70/test/ok.php" method="POST">
       <input type="hidden" name="PAResSyntaxOK" value="false">
       <input type="hidden" name="PAResVerified" value="false">
       <input type="hidden" name="version" value="2.0">
       <input type="hidden" name="merchantID" value="720000000000000">
       <input type="hidden" name="xid" value="RUlSOTM4NDIyOVhYMjY0ODg2NDg=">
       <input type="hidden" name="mdStatus" value="6">
       <input type="hidden" name="mdErrorMsg" value="Error with Directory Server
       (https://ds.visa3dsecure.com,https://dsw.visa3dsecure.com) response:
        ERROR message:Format of one or more elements is invalid according to
        the specification">
       <input type="hidden" name="txstatus" value="U">
       <input type="hidden" name="iReqCode" value="">
       <input type="hidden" name="iReqDetail" value="Merchant.password">
       <input type="hidden" name="vendorCode" value="">
       <input type="hidden" name="eci" value="">
       <input type="hidden" name="cavv" value="">
       <input type="hidden" name="cavvAlgorithm" value="">
       <input type="hidden" name="md"
   value="424242:0D67A4F3A74304CA9A393B31B47CA482FA6FC9A74848E03D51211B4D75
   28F634:3
        711:#">
       <input type="hidden" name="digest" value="dZBgADB2YmDiA6eo7G/n1W2Qz+E=">
       <input type="hidden" name="sID" value="1">
```

8.2.4 Invalid Credit Card

Cardholder has entered wrong credit card number (PAN).

```
<form name="returnform" action="https://testsanalpos.est.com.tr/servlet/est3dteststore"
   method="POST">
       <input type="hidden" name="PAResSyntaxOK" value="false">
       <input type="hidden" name="PAResVerified" value="false">
       <input type="hidden" name="version" value="2.0">
       <input type="hidden" name="merchantID" value="500000150">
       <input type="hidden" name="xid" value="F3+kkPe50Ra64nbNAtRGcUqxmr4=">
       <input type="hidden" name="mdStatus" value="7">
       <input type="hidden" name="mdErrorMsg" value="Invalid input data: Invalid
        input data. Field PAN is not valid : Length &lt; 13 or Length &gt; 19">
       <input type="hidden" name="txstatus" value="U">
       <input type="hidden" name="iReqCode" value="">
       <input type="hidden" name="iReqDetail" value="">
       <input type="hidden" name="vendorCode" value="">
       <input type="hidden" name="eci" value="">
       <input type="hidden" name="cavv" value="">
       <input type="hidden" name="cavvAlgorithm" value="">
       <input type="hidden" name="md"
   value="424242:A6334E1359D14E6AE82BA325E56F3BB2F010F9D69F8B47FF5F144998A69
   10A01:385
```

```
        3:#">
        <input type="hidden" name="digest" value="">
        <input type="hidden" name="sID" value="1">
```

8.2.5 Not authenticated Credit Card (MD Status = 0)

Cardholder cannot be authenticated, and cardholder should check card information.

```
<form name="returnform" action="https://195.95.188.70/test/ok.php" method="POST">
  <input type="hidden" name="PAResSyntaxOK" value="true">
 <input type="hidden" name="PAResVerified" value="true">
 <input type="hidden" name="version" value="2.0">
 <input type="hidden" name="merchantID" value="130000011">
 <input type="hidden" name="xid" value="AFCKbhneUg4+x2JzVL+LRs4xhF8=">
 <input type="hidden" name="mdStatus" value="0">
 <input type="hidden" name="mdErrorMsg" value="Not authenticated">
 <input type="hidden" name="txstatus" value="N">
 <input type="hidden" name="iReqCode" value="">
 <input type="hidden" name="iReqDetail" value="">
 <input type="hidden" name="vendorCode" value="">
 <input type="hidden" name="eci" value="">
 <input type="hidden" name="cavv" value="">
 <input type="hidden" name="cavvAlgorithm" value="">
 <input type="hidden" name="md"
     value="554960:290FF5AD74D03F19F4B7C63F2745145F4F9445B4FF0CB81DAEB676
    F0A7DD4F58:4022:#">
 <input type="hidden" name="digest" value="1ZGCEWB/IcfZ+OsdZxlgO+87tWo=">
 <input type="hidden" name="sID" value="2">
```

8.2.6 Authentication Unavailable (MD Status = 5)

3D Authentication is unavailable on Directory Server.

```
<input type="hidden" name="mdStatus" value="5">
        ...
<input type="hidden" name="mdErrorMsg" value="U-status from Directory Server:
    https://ds.visa3dsecure.com,https://dsw.visa3dsecure.com">
```

8.2.7 Error with Directory Server (MD Status = 6)

MPI definition error. Merchant's MPI definitions may be wrong. To solve this problem, the administrator should check and correct the merchant's MPI definition on MPI.

```
<input type="hidden" name="mdStatus" value="6">
        ...
<input type="hidden" name="mdErrorMsg" value="Error with Directory Server
    &#40;https://test2003.est.com.tr:9601/mdpayacs/vereq&#41; response: ERROR
    message: Required element missing. detail=&#40;merID&#41; vendorCode=">
```

8.2.8 3D-Secure Model Description Error (MD Status: 7)

This is a merchant integration error. The "storeType" parameter send by the merchant must match the configuration of the merchant. In the example below, the merchant needs to be informed to send the storetype parameters as "3d_pay_hosting"

```
DEBUG [21:05:30.314] Est3DGate: doPost: doPost started...
DEBUG [21:05:30.315] Est3DGate: doPost : dimUid =  130000011
DEBUG [21:05:30.315] Est3DGate: doPost : Processing for client 130000011 started
DEBUG [21:05:30.316] Est3DGate: doPost : store type = 3d_hosting
DEBUG [21:05:30.320] Base3DGate: getStoreType : DimStatus = OK
DEBUG [21:05:30.320] Base3DGate: getStoreType : store3d = 3D_PAY_HOSTING
DEBUG [21:05:30.320] Base3DGate: isValidStoreType: database = 3d_pay_hosting ,web =
    3d_hosting
DEBUG [21:05:30.320] Est3DGate: doPost : store type is not valid
DEBUG [21:05:30.321] Est3DGate: doPost : Storetype do not match db for 130000011
DEBUG [21:05:30.326] Base3DGate: getTemplate : acq of 130000011 is 13
DEBUG [21:05:30.329] Base3DGate: getTemplate : templatebase =
    /data/tomcatcore/webapps/fim/WEB-INF/est3dtemplates/13/
DEBUG [21:05:30.329] Est3DGate: getTemplate : template is =
/data/tomcatcore/webapps/fim/WEB-INF/est3dtemplates/13/gateerr.htm returning
DEBUG [21:05:30.329] Base3DGate: printErrorMessage: errTemplate =
    /data/tomcatcore/webapps/fim/WEB-INF/est3dtemplates/13/gateerr.htm
```

8.2.9 MPI Fallback (MD Status = 7)

The merchant cannot be identified by MPI. MPI definition of the merchant should be checked.

```
<input type="hidden" name="mdStatus" value="7">
                        ...
```

```
<input type="hidden" name="mdErrorMsg" value="Invalid merchant ID: 130000011">
```

8.3 3D Gateway Errors

3D gateway errors applicable only for internet integration.

Code	**Description**
3D-1001	Card does not support 3D
3D-1002	Wrong card data - Card no or CVV2
3D-1003	Wrong card data - Expiry
3D-1004	Wrong security code
3D-1005	System error
3D-1006	Wrong currency
3D-1007	Merchant does not support this payment model
3D-1008	Amount parameter is missing or in invalid format
3D-1009	Merchant payment model is not 3D or PAY
3D-1010	BIN is restricted to 3D
3D-1011	PAN is missing or not merchant payment page
3D-1012	Cards except Visa and MC do not support 3D
3D-1013	DinersClub does not support 3D
3D-1014	Payment Host is unavailable
3D-1015	This order has been paid already
3D-1016	Payment cannot be completed
3D-1017	Choose a currency for the merchant
3D-1018	Errors in merchant data
3D-1019	Unable to locate merchant URL for return, issuer ACS responds with an invalid PARes or system error
3D-1020	Missing parameter: clientid

8.4 BM Errors

Bank communication errors can occur if there is a problem on the communication between Nestpay and the bank.

Types of BM errors are:

- HOST based messaging problem
- Issuer or switch inoperative

8.4.1 HOST Based Messaging Problem

Code	Description
BM-1001	Invalid length TAG data in fields parsing.
BM-1002	Invalid reversal message.
BM-1006	Message parse error. Merchant id mismatch.
BM-1007	Message parse error. Terminal id mismatch.
BM-1008	Create acquirer request general error.

8.4.2 Issuer or Switch Inoperative

Code	Description
BM-9100	Host name is unknown.
BM-9101	Unable to connect to host, port closed.
BM-9102	Host communications IO error.
BM-9103	General connection failure to host.

8.5 ISO8583 Errors

ISO8583 error is caused by bank. ISO8583 errors descriptions and solutions are listed below.

Code	Description	Solution
ISO8583-9999	ISO8583 Non-Numeric Error.	Undefined error, contact your acquirer help desk
ISO8583-9998	ISO8583 Unknown Error from Issuer.	Undefined error, contact your acquirer help desk
ISO8583-01	Referral - call bank for manual approval.	Card owner can contact his/her issuer for detailed information
ISO8583-02	Fake Approval, but should not be used in a VPOS system, check with your bank.	Card owner can contact his/her issuer for detailed information
ISO8583-03	Invalid merchant or service provider.	Virtual POS might be deactivated. Contact your acquirer

ISO8583-04	Pick-up card.	Card owner can contact his/her issuer for detailed information
ISO8583-05	Do not honour	Card owner can contact his/her issuer for detailed information
ISO8583-06	Error (found only in file update responses).	Card owner can contact his/her issuer for detailed information
ISO8583-07	Pick up card, special condition.	Card owner can contact his/her issuer for detailed information
ISO8583-08	Fake Approval, but should not be used in a VPOS system, check with your bank.	Card owner can contact his/her issuer for detailed information
ISO8583-11	Fake Approved (VIP), but should not be used in a VPOS system, check with your bank.	Card owner can contact his/her issuer for detailed information
ISO8583-12	Transaction is not valid.	Contact your acquirer about the transaction
ISO8583-13	Invalid amount.	Amount is not in valid format
ISO8583-14	Invalid account number.	Terminal number or merchant number is wrong. Contact your acquirer
ISO8583-15	No such issuer.	Issuer is not defined
ISO8583-19	Reenter, try again.	Contact your acquirer help desk
ISO8583-20	Invalid amount.	Contact your acquirer help desk
ISO8583-21	Unable to back out transaction.	Contact your acquirer help desk
ISO8583-25	Unable to locate record on file.	Contact your acquirer help desk
ISO8583-28	Original is denied.	Contact your acquirer help desk
ISO8583-29	Original not found.	Contact your acquirer help desk
ISO8583-30	Format error (switch generated).	Contact your acquirer help desk
ISO8583-32	Referral (General).	Contact your acquirer help desk
ISO8583-33	Expired card, pick-up.	Card is expired, acquirer reject the transaction
ISO8583-34	Suspected fraud, pick-up.	Suspected fraud, acquirer reject the transaction
ISO8583-36	Restricted card, pick-up.	Card owner can contact his/her issuer for detailed information
ISO8583-37	Pick up card. Issuer wants card returned.	Card is stolen, card owner must return the card to his acquirer
ISO8583-38	Allowable PIN tries exceeded, pick-up.	Contact your acquirer help desk
ISO8583-41	Lost card, Pick-up.	Card is reported as lost, card

		owner cannot use this card
ISO8583-43	Stolen card, pick-up.	Card is reported as stolen, card owner cannot use this card
ISO8583-51	Insufficient funds.	Card limit is not sufficient
ISO8583-52	No checking account.	Contact your acquirer help desk
ISO8583-53	No savings account.	Contact your acquirer help desk
ISO8583-54	Expired card.	Card is expired, card owner cannot use this card
ISO8583-55	Incorrect PIN.	Contact your acquirer help desk
ISO8583-56	No card record.	Contact your acquirer help desk
ISO8583-57	Transaction not permitted to cardholder.	Contact your acquirer help desk
ISO8583-58	Transaction not permitted to terminal.	Contact your acquirer help desk
ISO8583-61	Exceeds withdrawal amount limit.	Contact your acquirer help desk
ISO8583-62	Restricted card.	Contact your acquirer help desk
ISO8583-63	Security violation	Contact your acquirer help desk
ISO8583-65	Activity limit exceeded.	Contact your acquirer help desk
ISO8583-75	Allowable number of PIN tries exceeded.	Contact your acquirer help desk
ISO8583-76	Key synchronization error.	Contact your acquirer help desk
ISO8583-77	Inconsistent data.	Contact your acquirer help desk
ISO8583-80	Date is not valid.	Card owner should check card details
ISO8583-81	Encryption Error.	Contact your acquirer help desk
ISO8583-82	CVV Failure or CVV Value supplied is not valid.	Card owner should check card details
ISO8583-83	Cannot verify PIN.	Contact your acquirer help desk
ISO8583-85	Declined (General).	Contact your acquirer help desk
ISO8583-91	Issuer or switch is inoperative.	Cannot communicate with the host, Contact your acquirer help desk
ISO8583-92	Timeout, reversal is trying.	Cannot communicate with the host, Contact your acquirer help desk
ISO8583-93	Violation, cannot complete (installment, loyalty).	Contact your acquirer help desk
ISO8583-96	System malfunction.	Contact your acquirer help desk
ISO8583-98	Duplicate Reversal.	Contact your acquirer help desk
ISO8583-YK	Card in black list.	Contact your acquirer help desk

9.Sample Scenarios

Question: What should be done if a client is making 12 month recurring payments and the amount changes after 3rd month?
Answer: Recurring payment amounts can't be changed afterwards. Remaining payments can be cancelled via control center or API calls and new recurring payment should be issued.

Question: What will happen if the client's card is cancelled or the client wants to change card number during recurring payments?
Answer: Recurring payment transactions will go on regardless of previous attempts results. Remaining payments can be cancelled via control center or API calls and new recurring payment should be issued.

Question: Can some recurring payments be skipped (cancelled) and others continue to exist?
Answer: It can only be done using the control center interface. For example 4th payment of a 10 month payment transaction can be cancelled. While using API calls, all remaining payments will be cancelled.