



# **FSS MERCHANT IMPLEMENTATION GUIDE**

---

## Version 1.0

---

This document provides information about integrating merchant portal with Payment Gateway.

Publishing Date: December, 2020

**Merchant Integration Document**



# Copyright Statement

---

## Copyright Notice

---

This document is confidential and proprietary to Financial Software and Systems (P) Ltd. and no part of this material shall be reproduced, published in any form by any means, electronic or mechanical including photocopy or any information storage or retrieval system nor shall the material be disclosed to third parties without the express written authorization of FSS.

Copyright © 2020 FSS (P) Ltd. All rights reserved.

## Restrictions and Disclaimer

---

The information contained in this document is confidential. This document shall not be disclosed, used or duplicated, in whole or in part, for any purpose other than to internally disseminate this information within FSS. The screens are captured in Demo version and are expected to change.



# Table of Contents

<b>1</b>	<b>ABOUT THIS DOCUMENT .....</b>	<b>6</b>
1.1	PURPOSE OF THE DOCUMENT .....	7
1.2	TARGET AUDIENCE .....	7
1.3	CONVENTIONS USED .....	7
<b>2</b>	<b>GETTING STARTED .....</b>	<b>8</b>
2.1	MERCHANT PREREQUISITES .....	9
2.2	HARDWARE PREREQUISITES .....	9
2.3	SOFTWARE PREREQUISITES .....	9
<b>3</b>	<b>TRANSACTION FLOW .....</b>	<b>10</b>
3.1	BANK HOSTED TRANSACTION FLOW .....	11
3.2	MERCHANT HOSTED TRANSACTION FLOW .....	11
3.3	OMANNET DEBIT CARD TRANSACTION FLOW .....	12
3.4	TOKENIZATION TRANSACTION FLOW .....	12
<b>4</b>	<b>MERCHANT INTEGRATION PROCESS .....</b>	<b>13</b>
4.1	INTEGRATION PROCESS .....	14
4.1.1	<i>Login into Payment Gateway .....</i>	<i>14</i>
4.1.2	<i>Download Plug-in.....</i>	<i>15</i>
4.1.3	<i>Downloading Resource and KeyStore Files .....</i>	<i>17</i>
4.1.4	<i>Copy Resource / Keystore / Plug-in to a folder location.....</i>	<i>19</i>
4.1.5	<i>Copy Terminal Alias Name .....</i>	<i>19</i>
4.2	INTEGRATE PLUG-IN/RESOURCE FILE WITH MERCHANT PAGE .....	19
4.2.1	<i>Code snippet for JSP Integration.....</i>	<i>19</i>
4.2.1.1	Bank Hosted Payment Integration .....	19
4.2.1.2	Merchant Hosted Payment Integration (Purchase) .....	22
4.2.1.3	Merchant Hosted Integration supported transactions (Refund, Void, Inquiry) .....	26
4.2.2	<i>Code snippet for PHP Integration .....</i>	<i>30</i>
4.2.2.1	Bank Hosted Payment Integration (Purchase) .....	30
4.2.2.2	Merchant Hosted Payment Integration (Purchase) .....	33
4.2.2.3	Merchant Hosted Integration supported transactions (Refund, Void, Inquiry) .....	36
4.2.3	<i>Code snippet for ASP Integration .....</i>	<i>41</i>
4.2.3.1	Bank Hosted Payment Integration .....	41
4.2.3.2	Merchant Hosted Integration(HTTP) (Purchase) .....	43
4.2.3.3	Merchant Hosted Integration(HTTP) (Refund, Void, Inquiry) .....	45

<b>5</b>	<b>APPENDIX I .....</b>	<b>51</b>
5.1	BANK HOSTED TRANSACTION .....	52
5.1.1	<i>Request from Merchant to PG .....</i>	<i>52</i>
5.1.2	<i>Response from PG to Merchant .....</i>	<i>54</i>
5.2	MERCHANT HOSTED TRANSACTION .....	55
5.2.1	<i>Request from Merchant to PG .....</i>	<i>55</i>
5.2.2	<i>Response from PG to Merchant .....</i>	<i>57</i>
5.2.3	<i>Final Response and Description .....</i>	<i>58</i>
5.3	RESULT .....	59
5.4	ERROR CODES .....	60
5.5	SAMPLE DEMO PAGE NAVIGATION .....	73
5.5.1	<i>Hosted Payment Integration .....</i>	<i>73</i>
5.6	BEST PRACTICES .....	74
5.6.1	<i>Important Notes and Information .....</i>	<i>76</i>
5.6.2	<i>Essentials .....</i>	<i>77</i>
5.6.3	<i>Frequently Asked Queries (General) .....</i>	<i>78</i>
<b>INDEX</b>	<b>80</b>	

# 1 ABOUT THIS DOCUMENT

---

*This document is prepared to provide information about integrating merchant portal with Payment Gateway using REST API.*

---

1.1	PURPOSE OF THE DOCUMENT .....	7
1.2	TARGET AUDIENCE .....	7
1.3	CONVENTIONS USED .....	7

## 1.1 Purpose of the Document

---

This is a technical document that explains the various methods of integration that is supported by Payment Gateway along with sample code snippets to help the merchant to integrate the merchant portal to Payment Gateway and process payments.

## 1.2 Target Audience

---


The target audiences for this document are as:

- Any developer/integrator working on behalf / for any merchant who has signed up with bank to process international Debit, domestic/international Credit card e-commerce transactions with 3D Secure Authentication on Payment Gateway.
- Any administrator or acquirer institution user who requires appropriate information of Payment Gateway integration process.

## 1.3 Conventions Used

---

The following table lists the conventions that are used in this book:

Convention	Indicates
Bold text	User interface (UI) elements and names of APIs
 Note:	Additional useful information that emphasizes or supplements important points. A note provides information that may apply only in special cases.
Code	Sample code

## 2 GETTING STARTED

---

*This section introduces the Hardware prerequisites, software prerequisites, Bank Hosted Transaction Flow and Merchant Hosted Transaction Flow.*

---

2.1	MERCHANT PREREQUISITES .....	9
2.2	HARDWARE PREREQUISITES .....	9
2.3	SOFTWARE PREREQUISITES .....	9



## 2.1 Merchant Prerequisites

---

Readers of this user guide should be familiar with basic either of the languages JSP, ASP .Net, PHP and Java.

## 2.2 Hardware Prerequisites

---

Merchants can use their existing hardware for transaction processing via Payment Gateway. Merchants may have a variety of arrangements for hosting their websites and thus have relevant security mandates for internet access controls and checks. This may include utilization of a Proxy Server which presents informed challenges. It is recommended that the merchant use a Public IP during the integration testing for transaction processing to the Payment Gateway. The merchant should ensure the Payment Gateway Domain and IP address is enabled at the firewall for both incoming and outgoing request/response.

## 2.3 Software Prerequisites

---

The merchant should have the requisite software for connecting to the Payment Gateway depending on the merchant application environment. The merchant may use combinations of OS/Web Server/ Application server whilst setting up and operating the website. Standard Software options are listed below, this list is for reference use only

Operating Systems - Windows 2000/ 2003 / 2008 Server, Sun Solaris, IBM AIX  
Web/Application Servers - Web/Application Server that support JSP & ASP .Net. The current version with all required patches is recommended to ensure success. Software Installation – Basic software that are required for Web/Application server should be installed at the merchant site. (Java/JDK for JSP integration is essential; similarly, .NET frame work is essential for ASP.NET integration).

# 3 TRANSACTION FLOW

---

*This section introduces the Bank Hosted Transaction Flow and Merchant Hosted Transaction Flow.*

---

3.1	BANK HOSTED TRANSACTION FLOW .....	11
3.2	MERCHANT HOSTED TRANSACTION FLOW .....	11
3.3	OMANNET DEBIT CARD TRANSACTION FLOW .....	12
3.4	TOKENIZATION TRANSACTION FLOW .....	12

## 3.1 Bank Hosted Transaction Flow

---

- Customer visits online through Merchant Website and select payment option as Payment Gateway
- Pre-requisite- Merchants website needs to be integrated with OAB PG through the secure plug-in
- Transaction request from merchant will reach to the application server
- OAB PG internally redirects the customer to the hosted payment page wherein customer enters his card credentials if Credit Card payment is selected then Credit card no, CVV and Expiry Date shall be accepted on the screen.
- OAB PG will send VEREQ and payment data to PG-web server
- A service in PG-Web Server will connect VISA/Master card directory (SSL Handshake) and send the VEREQ obtained from OAB PG
- PG-Web Server will receive VERES from VISA/Master card directory for the respective VEREQ from VISA/MasterCard directory
- After receiving PG-Web server will send VERES and payment data will be to OAB PG application server
- If Customer is enrolled for 3 D secure service
- PAREQ shall be initiated from OAB PG to issuer ACS. Customer shall be redirected to the banks issuer ACS server see the 2FA password entry screen on the respective issuer ACS.
- Once ACS verifies and provides response to OAB PG
- OAB PG shall build the ISO message and forwards to the Open/2 Transaction switch for authorization
- Once payment response arrived from Transaction switch to OAB PG then final response shall be communicated back to the Merchant Website.
- Merchant shall customize the response and shows it appropriately to their customer

## 3.2 Merchant Hosted Transaction Flow

---

- Customer shops online through Merchant Website and select payment option as Payment Gateway
- Pre-requisite- Merchants website needs to be integrated with OAB PG through the secure plug-in
- Merchant gets all the card credentials on their website and forwards the payment request to OAB PG
- Transaction request from merchant will reach to the PG application server
- OAB PG will send VEREQ and payment data to PG-web server
- Web service in PG-Web Server will connect VISA/Master card directory (SSL Handshake) and send the VEREQ obtained from OAB PG
- PG-Web Server will receive VERES from VISA/Master card directory for the respective VEREQ from VISA/Master card directory

- After receiving PG-Web server will send VERES and payment data to OAB PG once enrollment process is succeeding then
- If Customer is enrolled for 3 D secure service
- PAREQ shall be initiated from OAB PG to issuer ACS. Customer shall be redirected to the banks issuer ACS server see the 2FA password entry screen on the respective ACS.
- Once ACS verifies and provides response to OAB PG then OAB PG shall build the ISO message and forwards to the OpenN/2 Transaction switch for authorization
- Once payment response arrived from Transaction switch to OAB PG then final response shall be communicated back to the Merchant Website.
- Merchant shall customize the response and shows it appropriately to their customer

### 3.3 OmanNet Debit Card Transaction Flow

---

- OAB Payment Gateway receives the request from the merchant hosted or the bank hosted model.
- OAB Payment Gateway identifies the OmanNet Debit Cards based on BIN range.
- The request will be forwarded to CBO Payment Gateway for authentication and authorization.
- Inquiry request will be initiated to CBO Payment Gateway after OAB PG receives response.
- Once payment response arrived from the CBO Payment Gateway to the OAB PG then the final response shall be communicated back to the Merchant Website. OAB Payment Gateway receives the request from the merchant hosted or bank hosted model.

### 3.4 Tokenization Transaction Flow

---

- Tokenization Transaction can be performed only through Merchant Hosted Transaction. Therefore Merchant has to register to perform Tokenization Transaction.
- To register Tokenization, Merchant has to send the TokenFlag as "1" through Merchant Hosted Transaction.
- If the Tokenization registration is successful, then a unique Token Id will be generated by PG and PG will send this in response to the merchant along with other response parameters.
- If the merchant wants to perform subsequent Tokenization Transaction then Merchant should send the TokenFlag as "2" along with the TokenID and there is no need for card number and expiry.

## 4 MERCHANT INTEGRATION PROCESS

*This chapter covers the Hardware and Software pre-requisites for the merchant.*

4.1	INTEGRATION PROCESS .....	14
4.1.1	Login into Payment Gateway .....	14
4.1.2	Download Plug-in.....	15
4.1.3	Downloading Resource and KeyStore Files .....	17
4.1.4	Copy Resource / Keystore / Plug-in to a folder location.....	19
4.1.5	Copy Terminal Alias Name .....	19
4.2	INTEGRATE PLUG-IN/RESOURCE FILE WITH MERCHANT PAGE .....	19
4.2.1	Code snippet for JSP Integration.....	19
4.2.1.1	Bank Hosted Payment Integration .....	19
4.2.1.2	Merchant Hosted Payment Integration (Purchase) .....	22
4.2.1.3	Merchant Hosted Integration supported transactions (Refund, Void, Inquiry) 26	
4.2.2	Code snippet for PHP Integration .....	30
4.2.2.1	Bank Hosted Payment Integration (Purchase) .....	30
4.2.2.2	Merchant Hosted Payment Integration (Purchase) .....	33
4.2.2.3	Merchant Hosted Integration supported transactions (Refund, Void, Inquiry) 36	
4.2.3	Code snippet for ASP Integration .....	41
4.2.3.1	Bank Hosted Payment Integration .....	41
4.2.3.2	Merchant Hosted Integration(HTTP) (Purchase) .....	43
4.2.3.3	Merchant Hosted Integration(HTTP) (Refund, Void, Inquiry) .....	45

## 4.1 Integration Process

Steps to be followed by Merchant Integration Team:

1. Merchant should login through payment gateway URL using their Merchant ID, User ID and Password  
URL: <https://certpayments.oabipay.com/mrch/merchantLogin.htm>
2. Merchant should download the respective plug-in (based on the platform (Java/ASP/PHP etc.) to connect Payment Gateway from the menu.  
**"Merchant Process -> Plugin Download"**
3. Merchant should download the Resource file & Keystore file from the menu  
**"Merchant Process-> Resource File Download"**.
4. Merchant should copy to a folder in the server location.  
For Example: **/usr/local/paymentgateway/**
5. Integrate the plug-in and resource file with the merchant web page.
6. Construct the Request message as expected by the Payment Gateway.
7. Process the response message receiving from the Payment Gateway.

### 4.1.1 Login into Payment Gateway

To login into Payment Gateway application follow the below steps:

1. Merchant should login through payment gateway URL using their Merchant ID, User ID and Password:  
URL: <https://certpayments.oabipay.com/mrch/merchantLogin.htm>  
Payment Gateway user login page is displayed.

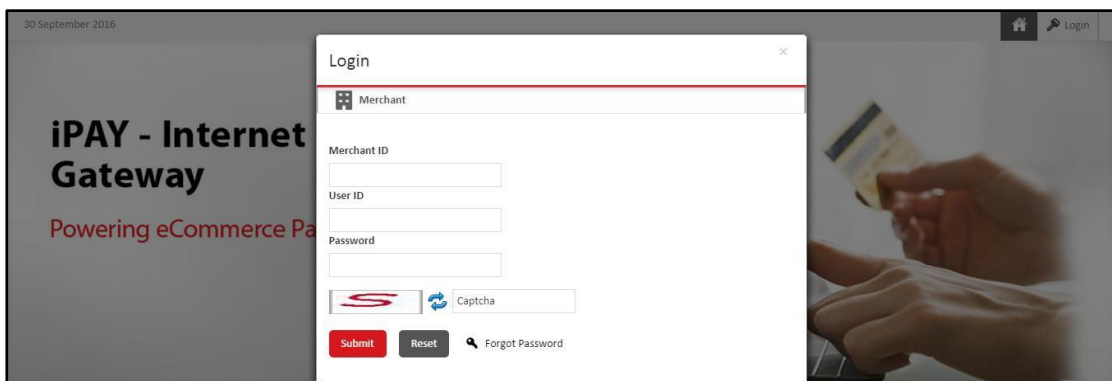


Figure 1: Merchant Login Screen

2. Click **Submit**. The application displays the following "Home Page" screen with menus.

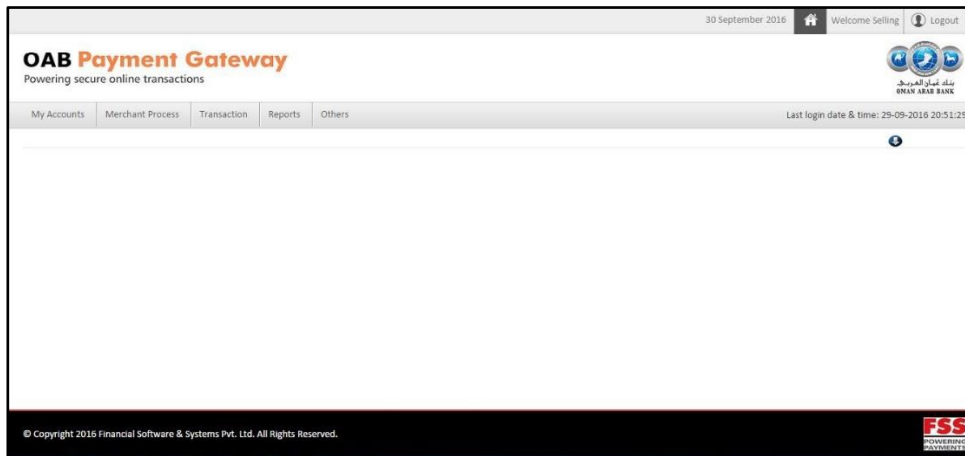


Figure 2: Home Page Screen

## 4.1.2 Download Plug-in

### a. JSP Plug-in

To download JSP Plug-in, click **JSP** link in the path "**Merchant Process->Plugin Download**".

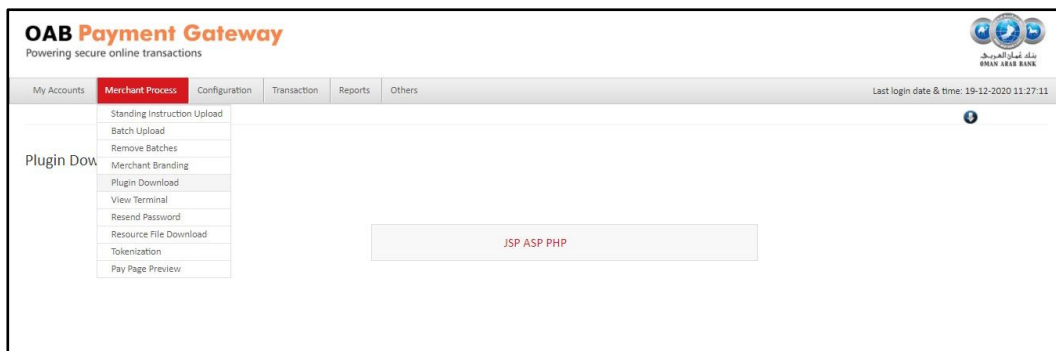



Figure 1: Plugin Download Screen

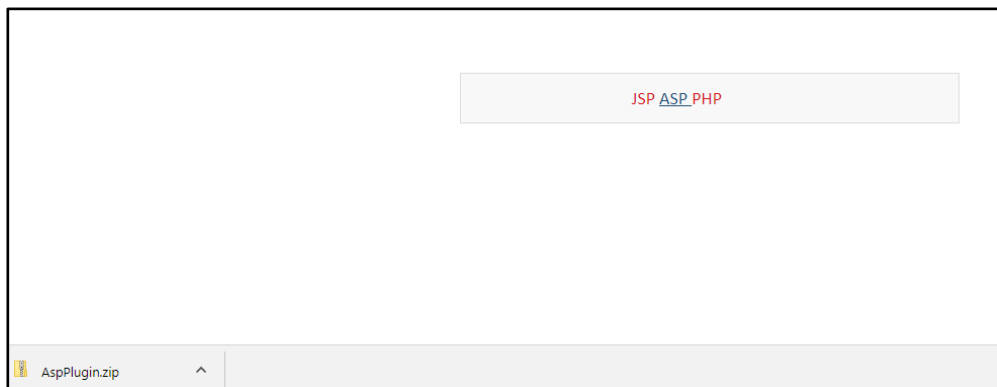
- Click the **JSP** link. The Plug in jar file gets downloaded in the system. Merchant needs to add this Jar file with their Merchant Application Libraries.

### b. ASP Plug-in

- To download **ASP.net Plug-in**, click **ASP** link in the path "**Merchant Process->Plugin Download**". The Plug gets downloaded in the system on clicking the link.

**Figure 2: Plugin Download screen**

 **Note:** We can download all plugins under same menu

**Figure 3: Plugin.zip folder Download screen**

### c. **PHP Plug-in**

- To download **PHP** Plug-in, click **PHP** Plug-in link in the path "**Merchant Process->Plugin Download**".
- Click the **PHP** link. The Plug gets downloaded in the system.

**Figure 4: Plugin Download screen**


 **Note:** We can download all plugins under same menu





Figure 5: Plugin .exe Download screen

- Click **Save** to download the Plug-in file in your system.

### 4.1.3 Downloading Resource and KeyStore Files

#### a. Downloading JSP/ASP KeyStore File

- Click **Java/ASP.NET KeyStore** link in the path "**Merchant Process-> Resource File Download**".



Figure 6: Resource File Download screen

- Click the **JAVA/ASP.NET KeyStore** file link to download the KeyStore file in the system.

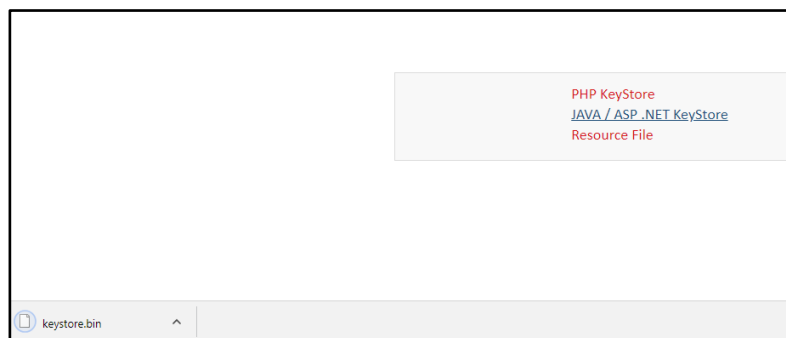


Figure 7: KeyStore File Download screen

- Click **Save** to download the KeyStore file in your system.

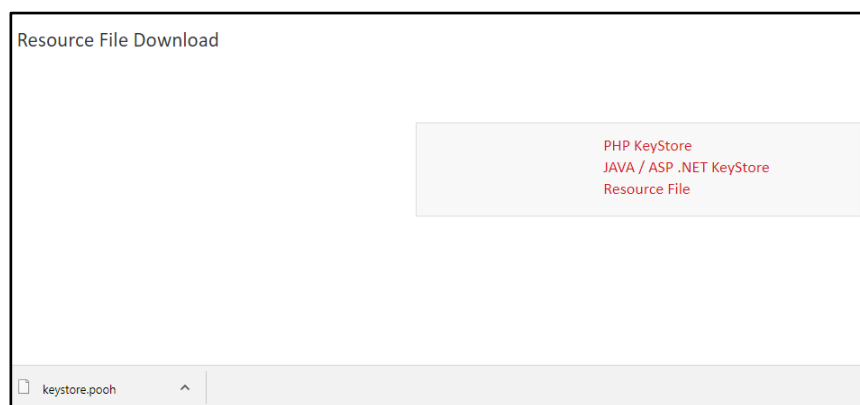
#### b. Downloading PHP KeyStore File

- Click **PHP KeyStore** link in the path  
**"Merchant Process-> Resource File Download"**.



**Figure 8: Resource File Download screen**

- Click the **PHP KeyStore** File link to download the KeyStore file in the system.

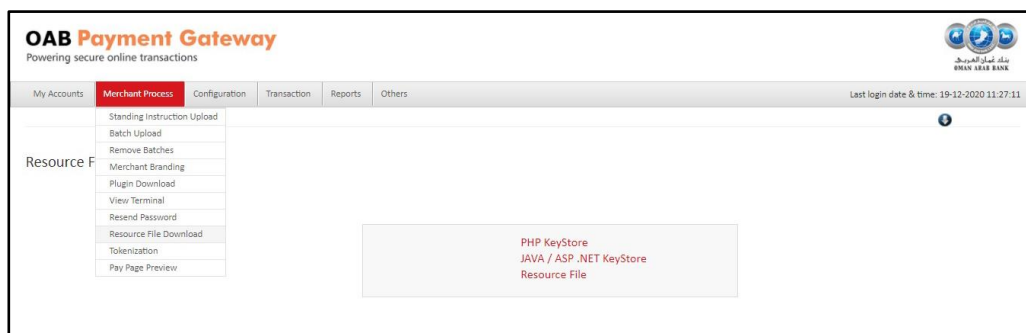


**Figure 9: KeyStore File Download screen**

- Click **Save** to download the KeyStore file in your system.

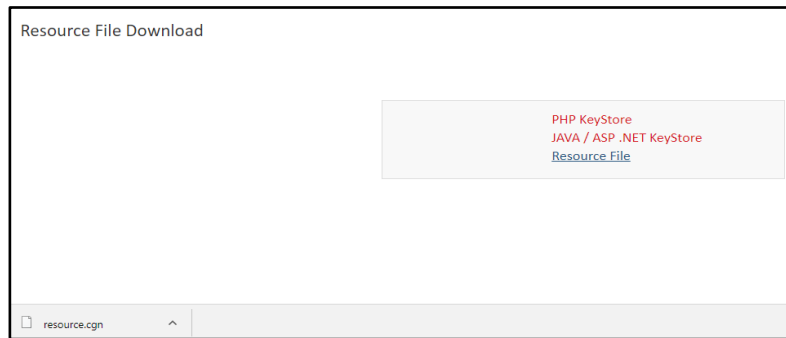
### c. Downloading Resource File

- Click **Resource file** link in the path  
**"Merchant Process-> Resource File Download"**



**Figure 10: Resource File Download screen**

- Click the **Resource File** link to download the resource file in the system.



**Figure 11: Resource File Download screen**

- Move the downloaded resource & Key store to a specific folder.  
For Example:- **C:\\resourcepath**

#### 4.1.4 Copy Resource / Keystore / Plug-in to a folder location

- Move the downloaded Resource and KeyStore files to a specific folder.  
For Example: **C:\\resourcepath**

#### 4.1.5 Copy Terminal Alias Name

1. For Accessing the Terminal Details, go to **Merchant Process -> View Terminal**.
2. Click "**View**" link in the record corresponding to the terminal to open the Terminal Details.
3. Copy **Terminal Alias name** under the **Plugin** tab.

### 4.2 Integrate Plug-in/Resource File with Merchant page

#### 4.2.1 Code snippet for JSP Integration

##### 4.2.1.1 Bank Hosted Payment Integration

##### Purchase Transaction

Merchant can connect Payment gateway using iPay Plugin with the below steps:

```
iPayOabPipe pipe = new iPayOabPipe();
String resourcePath = "c:\\resourcepath"; // Mandatory
String keystorePath = "c:\\ resourcepath"; // Mandatory
```

```

String receiptURL= "http://www.demomerchant.com/result.jsp"; // Mandatory
String errorURL= "http://www.demomerchant.com/error.jsp";// Mandatory
// 1 - Purchase
String action="1"; // Mandatory
//Terminal Alias Name. Mandatory
String aliasName = "Terminal aliasName";
//Transaction Currency. Mandatory
String currency = "currency"; (ex: "512")
//Optional
String language = "language"; (ex: "USA")
//Transaction Amount. Mandatory
String amount= "10.35";
//Merchant Track ID. Mandatory, Merchant should ensure the uniqueness.
String trackid = "109088888";
//User Defined Fields.
String Udf1= "Udf1"; // Optional

String Udf2= "Udf2"; // Optional

String Udf3= "Udf3"; // Optional

String Udf4= "Udf4"; // Optional
String Udf5= "Udf5"; // Optional

//Set Values
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(keystorePath);
pipe.setAlias(aliasName);
pipe.setAction( action );
pipe.setCurrency(currency);
pipe.setLanguage(language);
pipe.setResponseURL(receiptURL );
pipe.setErrorURL(errorURL);
pipe.setAmt(amount);
pipe.setTrackId(trackid);
pipe.setUdf1 (Udf1);
pipe.setUdf2 (Udf2);
pipe.setUdf3 (Udf3);
pipe.setUdf4 (Udf4);
pipe.setUdf5 (Udf5);
//Tokenization snippet
pipe.setTokenFlag(tokenFlag);//1- Registration, 2-Payment
pipe.setTokenNumber(tokenNumber);//Set token number for subsequent
transaction

int result = pipe.performPaymentInitializationHttp();
if(result!=0)
System.out.println("Error Occured!See Console for more details");
else{
//Merchant can get trandata and tranportal id from ipaypipe inorder to set
in post body.

```

```
String trandata = pipe.getTranPostData();
String tranportalID = pipe.getId();
String webAddress = pipe.getWebAddress()
Merchant Error Url and Merchant Response Url can be set by merchant in post
request body.
/** End of Request Processing**/
return type :Integer
returns value -1 on exception being caught otherwise returns 0 .
```

### Response received from Payment Gateway to Merchant

```
trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32
EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE863
3E0A1DD90D29338545DA582B0F3500BA9375313637690531C6D7F8F7489C7CD8B73F
9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69FC577300F9A
AF38E044F0A34694348506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F
```

//To decrypt the above response, Merchant should follow the below step:

Merchant must set certain fields in iPay plugin to process the response.

Create the plugin object as below,

```
iPayOabPipe pipe = new iPayOabPipe();
//Initialization
String resourcePath = c:\\resourcepath"; //Mandatory
String keystorePath = "c:\\ resourcepath"; //Mandatory
//Terminal Alias Name. Mandatory

String aliasName = "Terminal aliasName";

//Set Values
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(keystorePath);
pipe.setAlias(aliasName);
```

//The method to be called for decrypting the response send by Payment Gateway

```
int result = pipe.parseEncryptedRequest(request.getParameter("trandata"));
```

**Case1:** If the return value from this method is "0" and request. getParameter ("ErrorText") is null, then Merchant can get the decrypted data of the response fields.

### For Example:

```
//To get result,
pipe.getResult(); // Ex: CAPTURED OR NOT CAPTURED
//To get payment ID
pipe.getPaymentId();
//To get Transaction ID
pipe.getTransId();
//To get Amount pipe.getAmt();
```

```
//To get Transaction Reference ID
pipe.getRef()
//To get Mrch Track ID
pipe.getTrackId()
//To get token number if registered for tokenization
pipe.getTokenCustomerId();
```

**Case2:** If request.getParameter("ErrorText") is not null, then do the following to get response fields.

```
//To get error
request.getParameter("ErrorText");
```

**Case3:** If the return value from this method is not "0", then Merchant will get the error from below mentioned steps.

```
//To get error pipe.getError();
//To get error text
pipe.getError_text();
```

```
// To get Transaction ID
pipe.getTransId();
```

```
// To get Payment ID
pipe.getPaymentId();
```

**Case4:** If trandata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
request.getParameter("ErrorText");
//To get Transaction ID
request.getParameter("tranid");
//To get Payment ID
request.getParameter("paymentid");
/** End of Response Processing**/
```

## 4.2.1.2 Merchant Hosted Payment Integration (Purchase)

### Purchase Transaction

```
/** Request Processing**/
```

Merchant can connect payment gateway using iPay Plugin with the below steps:

```
iPayOabPipe pipe = new iPayOabPipe();
//Initialization
String resourcePath = "c:\\resourcepath\\";
String keystorePath = "c:\\ keystorePath\\";
String receiptURL= "http://www.demomerchant.com/result.jsp";
String errorURL= "http://www.demomerchant.com/error.jsp";
String action="1";
```

```
//Terminal Alias Name in the merchant portal for the terminal
String aliasName = "aliasName";
String cvv="123";
//Value "C" for Credit and "D" for Debit
String Type="C";

//Transaction Currency
String currency = "currency"; //(ex: "512")
String language = "language"; //(ex: "USA")
//Transaction Amount
String amount = "10.000";
//Merchant Track ID
String trackid = "109088888";
String cardNumber = "4000000000000002";
String expmm = "12";
String expyy = "2016";
String name = "Suresh";

//User Defined Fields String Udf1= "Udf1"; String Udf2= "Udf2"; String Udf3=
"Udf3"; String Udf4= "Udf4"; String Udf5= "Udf5";

//Set Values pipe.setTrackId(trackid);
pipe.setAlias(aliasName);
pipe.setResourcePath(resourcePath);
pipe.setAction(action);
pipe.setAmt(amount);
pipe.setCurrency(currency);
pipe.setCard(cardNumber);
pipe.setCvv2(cvv);
pipe.setExpMonth(expmm);
pipe.setExpYear(expyy);
pipe.setMember(name);
pipe.setType(Type);
pipe.setLanguage(language);
//If supporting reversal transaction, transaction id has to be given
pipe.setTransId("");
pipe.setUdf1(Udf1); pipe.setUdf2(Udf2);
pipe.setUdf3(InetAddress.getLocalHost().getHostAddress());
pipe.setUdf4(Udf4);
pipe.setUdf5(Udf5);
pipe.setKeystorePath(keystorePath);
pipe.setResponseURL(receiptURL);
pipe.setErrorURL(errorURL);
//Tokenization snippet
pipe.setTokenFlag(tokenFlag);//1- Registration, 2-Payment
pipe.setTokenNumber(tokenNumber);
//The method to be called for Merchant Hosted normal transaction
int result = pipe.performTransactionHTTP();
//The method to be called for Merchant Hosted 3D secure/VPAS transaction
int result = pipe.performVbVTransaction();
if(result!=0)
```

```

System.out.println("Error Occured!See Console for more details");
else{
//Merchant can get trandata and tranportal id from ipaypipe inorder to set
in post body.
//Set token number for subsequent transaction
(pipe.performTransactionHTTP()!=0)
System.out.println("Error Occured!
See Console for more details");
// Step 1
//The method to be called for Merchant Hosted normal transaction
int result = pipe.performTransactionHTTP();
//The method to be called for Merchant Hosted 3D secure/VPAS transaction
int result = pipe.performVbVTransaction();
if(result!=0)
System.out.println("Error Occured!See Console for more details");
else{
//Merchant can get trandata and tranportal id from ipaypipe inorder to set
in post body.String trandata = pipe.getTranPostData();
String tranportalID = pipe.getId();
String webAddress = pipe.getWebAddress();
Merchant Error Url and Merchant Response Url can be set by merchant in post
request body.
Once the request received from Merchant, PG will verify the merchant and
terminal and proceed the transaction and provides the response
/** End of Request Processing**/
//Step 2
/** Response received from Payment Gateway to Merchant **/
Here the above mentioned response parameters are sent as opened text.

```

### Response sent to Merchant

```

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEA
F32EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87CD7
6FE8633E0A1DD90D29338545DA582B0F3500BA9375313637690531C6D7F8F7489C
7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F

```

```

//To decrypt the above response, Merchant should follow the below step:
Merchant must set certain fields in plugin as in request processing.
Create the plugin object as,
iPayOabPipe pipe = new iPayOabPipe();
//Initialization
String resourcePath = "c:\\resourcePath";
String keystorePath = "c:\\ keystorePath";
//Terminal Alias Name in the merchant portal for the terminal
String aliasName = "aliasName";
//Set Values
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(keystorePath);

```



```
pipe.setAlias(aliasName);
```

//The method to be called to decrypt the response sent by Payment Gateway

```
int result =
```

```
pipe.parseEncryptedResult(request.getParameter("trandata"));
```

**Case1:** If the return value from this method is "0" and request.getParameter("ErrorText") is null, then Merchant can get the decrypted data of the response fields.

String tokenNumber = pipe.getTokenCustomerId();// contains token number if registered for tokenization.

**For Example:**

//Value "CAPTURED" is success and other values must be treated as failure.

```
String Result = pipe.getResult(); // gives the value in the tag.
```

```
String PostDate = pipe.getDate(); // contains post date
```

```
String refNum = pipe.getRef(); // contains RRN no generated by PG
```

```
String trackId = pipe.getTrackId(); // contains merchant track ID
```

```
String tranId = pipe.getTransId(); // contains PG Transaction ID
```

```
String amt = pipe.getAmt(); // contains transaction amount
```

```
String paymentId = pipe.getPaymentId(); // contains payment ID
```

```
String auth = pipe.getAuth(); // contains Auth code
```

```
String errorText = pipe.getError_text(); // contains get Error Text
```

```
String error = pipe.getError(); // contains get Error Text
```

**Case2:** If request.getParameter("ErrorText") is not null, then do the following to get response fields.

```
//To get error
```

```
request.getParameter(" ErrorText");
```

**Case3:** If the return value from this method is not "0", then Merchant will get the error from below mentioned steps.

```
//To get error pipe.getError();
```

```
//To get error text pipe.getError_text();
```

```
// To get Transaction ID
```

```
pipe.getTransId();
```

```
// To get Payment ID
```

```
pipe.getPaymentId();
```

**Case4:** If trandata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
```

```
request.getParameter("ErrorText");
```

```
//To get Transaction ID
```

```
request.getParameter("tranid");
```

```
//To get Payment ID
```

```
request.getParameter("paymentid");
```

```
/** End of Response Processing**/
```

### 4.2.1.3 Merchant Hosted Integration supported transactions (Refund, Void, Inquiry)

Merchant can perform the supporting transaction either using HTTP or TCP model.

#### Supported transactions using HTTP model

Merchant can connect iPay Plugin using below step :

```
iPayOabPipe pipe = new iPayOabPipe();
//Initialization//Mandatory
String resourcePath = "c:\\resourcepath";
//Mandatory
String keystorePath = "c:\\ resourcepath";
//Mandatory
String receiptURL= "http://www.demomerchant.com/result.jsp";
//Mandatory
String errorURL= "http://www.demomerchant.com/error.jsp";
//2-Refund, 3-Void, 8-Inquiry
String action="2";
//Terminal Alias Name in the merchant portal for the terminal
String aliasName = "aliasName";
//Merchant Track ID
String trackid = "109088888";

//Transaction Currency
String currency = "currency"; // (ex: "512")
String language = "language"; //(ex: "USA")
//Transaction Amount
String amount = "10.000";
// Transaction ID of already completed transaction
String transId = "2015789645632";
//User Defined Fields
String Udf1= "Udf1";
String Udf2= "Udf2";
String Udf3= "Udf3";
String Udf4= "Udf4";
String Udf5= "Udf5";
//Set Values
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(keystorePath);
pipe.setAlias(aliasName);
pipe.setAction( action );
pipe.setCurrency(currency);
pipe.setLanguage(language);
pipe.setResponseURL( receiptURL );
pipe.setErrorURL(errorURL);
pipe.setAmt(amount);
```

```
//Transaction Id value of the completed transaction
pipe.setTransId(transId);
//Track ID should be unique for all transactions
pipe.setTrackId(trackid);
pipe.setUdf1 (Udf1);
pipe.setUdf2 (Udf2);
pipe.setUdf3 (Udf3);
pipe.setUdf4 (Udf4);
pipe.setUdf5 (Udf5);
//The method to be called for HTTP process is
int result = pipe.performTransactionHTTP();
if(result!=0)
System.out.println("Error Occured!See Console for more details");
else{
//Merchant can get trandata and tranportal id from ipaypipe inorder to set
in post body.
String trandata = pipe.getTranPostData();
String tranportalID = pipe.getId();
String webAddress = pipe.getWebAddress();
Merchant Error Url and Merchant Response Url can be set by merchant in post
request body.
Once the request received from Merchant, PG will verify the merchant and
terminal and proceed the transaction and provides the response
When performing inquiry or void purchase transaction, Mandatory fields are
Action Type, Transaction Amount,UDF5(Merchant Track id/Payment
id/Transaction id/Transaction sequence number),Transaction id(setTransid
method).
Support transaction can be done using Payment ID or Merchant Track ID or
Transaction ID.
Payment ID: Set text "PaymentID" in the UDF5 using pipe.setUdf5("PaymentID")
and Set payment ID value using pipe.setTransId.
Merchant Track ID: Set "TrackID" in the UDF5 field using
pipe.Udf5("TrackID") and set the track Id value using pipe.setTransId.
Sequence Number: Enter text "SeqNum" in the UDF5 field using
pipe.Udf5("SeqNum") and set the sequence number value using pipe.setTransId.
Transaction ID:Set Transaction ID value using pipe.setTransId.
/** End of Request Processing**/
//Step 2
/** Response received from Payment Gateway to Merchant **/
Here the above mentioned response parameters are sent as opened text.
```

### Response sent to Merchant

```
trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF0
2CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1D
D90D29338545DA582B0F3500BA9375313637690531C6D7F8F7489C7CD8B73F9EC7E1C62
2DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69FC577300F9AAF38E044F0A3
4694348506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F
```

//To decrypt the above response, Merchant should follow the below

step: Merchant should set certain fields in plugin for processing the response.

Create the plugin object as below,

```
iPayOabPipe pipe = new iPayOabPipe();
//Initialization
String resourcePath = c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
//Terminal Alias Name

String aliasName = "Terminal aliasName";

//Set Values
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);

// The method to be called is to decrypt the Payment Gateway response

int result =
pipe.parseEncryptedResultHttp(request.getParameter("trandata");
```

**Case1:** If the return value from this method is "0"and request.getParameter("ErrorText") is null, then Merchant can get the decrypted data of the response fields.

**For Example:**

```
// To get result,
pipe.getResult();
// To get payment ID
pipe.getPaymentId();
//To get Transaction ID
pipe.getTransId();
// To get Amount pipe.getAmt();
//To get Transaction Reference ID
pipe.getRef()
// To get Mrch Track ID
pipe.getTrackId()
// To get transaction amount pipe.getAmt()
```

**Case2:** If request.getParameter("ErrorText") is not null, then do the following to get response fields.

```
//To get error
request.getParameter("ErrorText");
```

**Case3:** If the return value from this method is not "0", then Merchant will get the error from below mentioned steps.

```
//To get error pipe.getError();
//To get error text pipe.getError_text();

// To get Transaction ID
pipe.getTransId();

// To get Payment ID
```

```
pipe.getPaymentId();
```

**Case4:** If trandata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
request.getParameter("ErrorText");
//To get Transaction ID
request.getParameter("tranid");
//To get Payment ID
request.getParameter("paymentid");
/** End of Response Processing**/
```

## Supported transactions using TCP model

Merchant can connect iPay Plugin using below step :

```
iPayOabPipe pipe = new iPayOabPipe();
//Initialization
String resourcePath = "c:\\resourcepath\\";
String keystorePath = "c:\\ keystorePath\\";
// 2-Refund, 3- Void Purchase, 8 - Inquiry String action="8";
//Terminal Alias Name in the merchant portal for the terminal
String aliasName = "aliasName";
//Merchant Track ID
String trackid = "109088888";

//Transaction Currency
String currency = "currency"; // (ex: "512")
String language = "language"; // (ex: "USA")
//Transaction Amount
String amount = "10.000";
// Transaction ID of already completed transaction
String transId = "2015789645632";

//User Defined Fields String Udf1= "Udf1"; String Udf2= "Udf2"; String Udf3=
"Udf3"; String Udf4= "Udf4"; String Udf5= "Udf5";

//Set Values pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(keystorePath); pipe.setAlias(aliasName);
pipe.setAction( action ); pipe.setCurrency(currency);
pipe.setLanguage(language); pipe.setAmt(amount);
// Transaction Id value of the completed transaction
pipe.setTransId(transId);
pipe.setTrackId(trackid);
//Track ID should be unique for all transactions
pipe.setUdf1 (Udf1); pipe.setUdf2(Udf2); pipe.setUdf3(Udf3);
pipe.setUdf4(Udf4); pipe.setUdf5(Udf5); if(pipe.performTransaction!=0)
System.out.println("Error Occured! See Console for more details");

//The method to be called is
```

```
int i = pipe.performTransaction();
```

Once the request received from Merchant, PG will verify the merchant and terminal and proceed the transaction and provides the response

When performing inquiry or void purchase transaction,

Mandatory fields are Action Type, Transaction Amount, UDF5 (Merchant Track id/Payment id/Transaction id/Transaction sequence number), Transaction id (setTransid method).

Payment ID: Set text "PaymentID" in the UDF5 using pipe.setUdf5("PaymentID") and Set payment ID value using pipe.setTransId.

Merchant Track ID: Set "TrackID" in the UDF5 field using pipe.Udf5("TrackID") and set the track Id value using pipe.setTransId.

Sequence Number: Enter text "SeqNum" in the UDF5 field using

pipe.Udf5("SeqNum") and set the sequence number value using pipe.setTransId.

Transaction ID: Set Transaction ID value using pipe.setTransId.

/\*\* End of Request Processing\*\*/

## 4.2.2 Code snippet for PHP Integration

### 4.2.2.1 Bank Hosted Payment Integration (Purchase)

#### Purchase Transaction

/\*\* Request Processing\*\*/

Merchant can connect Payment gateway using iPay Plugin with the below step

```
//Initialization
$resourcePath = "c:\\resourcepath"; // Mandatory
$keystorePath = "c:\\ resourcepath"; // Mandatory
$recieptURL= "http://www.demomerchant.com/result.jsp"; // Mandatory
$errorURL= "http://www.demomerchant.com/error.jsp"; // Mandatory
// 1 - Purchase
$action="1"; // Mandatory
//Terminal Alias Name. Mandatory
$aliasName = "Terminal aliasName";
//Transaction Currency. Mandatory
$currency = "currency"; (ex: "512")
//Optional
$language = "language"; (ex: "USA","EN")
//Transaction Amount. Mandatory
$amount= "10.35";
//Merchant Track ID. Mandatory

$trackid = "109088888";
//User Defined Fields.
$Udf1= "Udf1"; // Optional
```

```

$Udf2= "Udf2"; // Optional

$Udf3= "Udf3"; // Optional

$Udf4= "Udf4"; // Optional
$Udf5= "Udf5"; // Optional

$myObj =new iPayOabPipe();

$myObj->setResourcePath($resourcePath);
$myObj->setKeystorePath($keystorePath);
$myObj->setAlias($aliasName);
$myObj->setAction( $action );
$myObj->setCurrency($currency);
$myObj->setLanguage($language);
$myObj->setResponseURL( $ receiptURL );
$myObj->setErrorURL($errorURL);
//Tokenization snippet
$myObj->settokenFlg($tokenFlg); //1- Registration, 2-Payment
$myObj->settokenNumber($tokenNumber); //Set token number for subsequent
transaction
$myObj->setAmt($amount);
$myObj->setTrackId($trackid);
$myObj->setUdf1 ($Udf1);
$myObj->setUdf2 ($Udf2);
$myObj->setUdf3 ($Udf3);
$myObj->setUdf4 ($Udf4);
$myObj->setUdf5 ($Udf5);
/** For Bank Hosted Payment Integration, the method to be called is **/
$result = $myObj->performPaymentInitializationHTTP();

//To post the request
$url=$myObj->getWebAddress();
$strandata=$myObj->getTranPostData();
$stranportalId=$myObj->getId();
die();
/** End of Request Processing**/

```

### Response sent from Payment Gateway

```

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEA
F32EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87CD7
6FE8633E0A1DD90D29338545DA582B0F3500BA9375313637690531C6D7F8F7489C
7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F

```

//To decrypt the above response, Merchant should follow the below step:

Merchant should set certain fields in iPay plugin to process the response.

Create the plugin object as,

```
//Initialization
$resourcePath = c:\\resourcepath"; //Mandatory
$keystorePath = "c:\\ resourcepath"; //Mandatory
//Terminal Alias Name. Mandatory
$myObj =new iPayOabPipe();
$aliasName = "Terminal aliasName";
$myObj->setAlias(trim($aliasName));
$myObj->setResourcePath(trim($resourcePath));
$myObj->setKeystorePath(trim($keystorePath));
```

// For Hosted Payment Integration, the method to be called is

```
$returnValue = $myObj->parseEncryptedRequest($_GET['trandata']);
```

**Case1:** If the return value from this method is "0" and \$\_GET["ErrorText"] is null, then Merchant can get the decrypted data of the response fields.

**For Example:**

// To get result. Value "CAPTURED" is success and other values must be treated as failure.

```
// To get result,
$myObj->getResult();
// To get payment ID
$myObj->getPaymentId();
//To get Transaction ID
$myObj->getTransId();
// To get Amount
$myObj->getAmt();
//To get token number if registered for tokenization
$myObj->gettokenCustomerId();
```

**Case2:** If \$\_GET["ErrorText"] is not null, then do the following to get response fields.

```
//To get error
$_GET['ErrorText']
```

**Case3:** If the return value from this method is not "0", then Merchant will get the error from below mentioned steps.

```
//To get error
$myObj->getError();
// To get Transaction ID
$myObj->getTransId();
// To get Payment ID
$myObj->getPaymentId();
```

**Case4:** If trandata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
$_GET['ErrorText'];
//To get Transaction ID
$_GET['tranid'];
```



```
//To get Payment ID
$_GET['paymentid'];
/** End of Response Processing**/
```

## 4.2.2.2 Merchant Hosted Payment Integration (Purchase)

### Purchase Transaction

#### **/\*\* Request Processing\*\*/**

Merchant can connect Payment gateway using iPay Plugin with the below step:

```
//Initialization
$resourcePath = "c:\\resourcepath"; //Mandatory
$keystorePath = "c:\\ resourcepath"; //Mandatory
$recieptURL= "http://www.demomerchant.com/result.jsp"; //Mandatory
$errorURL= "http://www.demomerchant.com/error.jsp"; //Mandatory
// 1 - Purchase
$action="1"; //Mandatory
//Terminal Alias Name. Mandatory
$aliasName = "Terminal aliasName";
//Transaction Currency
$ currency = "currency"; (ex: "512")
$ language = "language"; (ex: "USA")
//Transaction Amount
$ amount = "10.000";
//Merchant Track ID
$ trackid = "109088888";
$ cardNumber = "40000000000000002";
$ expmm = "12";
$ expyy = "2015";
$ name = "Mohammed";
//For Credit "C" and For Debit "D"
$type = "C";

//User Defined Fields
$ Udf1= "Udf1";
$ Udf2= "Udf2";
$ Udf3= "10.22.22.2";
$ Udf4= "Udf4";
$ Udf5= "Udf5";

$myObj =new iPayOabPipe();
//Set Values
$myObj->setTrackId($trackId);
$myObj->setAlias($aliasName);
$myObj->setResourcePath($resourcePath);
$myObj->setAction($action);
```

```

$myObj->setAmt($amount);
$myObj->setCurrency($currency);
$myObj->setCard($cardNumber);
$myObj->setCvv2($cvv);
$myObj->setExpMonth($expmm);
$myObj->setExpYear($expyy);
$myObj->setMember(name);
$myObj->setType(Type);
$myObj->setLanguage(language);
$myObj->setUdf1(udf1);
$myObj->setUdf2(udf2);
$myObj->setUdf3(Udf3);
$myObj->setUdf4(udf4);
$myObj->setUdf5(udf5);
$myObj->setKeystorePath(keystorePath);
$myObj->setResponseURL(receiptURL);
$myObj->setErrorURL(errorURL);
//Tokenization snippet
$myObj->settokenFlg($tokenFlg); //1- Registration, 2-Payment
$myObj->settokenNumber($tokenNumber); //Set token number for subsequent
transaction
//The method to be called for connecting Payment Gateway
$result = $myObj->performVbVTtransaction(); //3D Secure
$result = $myObj->performTransactionHttp(); //Non3D
Merchant can get trandata and tranportal id from ipaypipe inorder to set in
post body.
String trandata = pipe.getTranPostData();
String tranportalID = pipe.getId();
String webAddress = pipe.getWebAddress();
Merchant Error Url and Merchant Response Url can be set by merchant in post
request body.
} else
{echo $myObj->getError(); // Problem in connecting Payment Gateway
}
Once the transaction request received from Merchant, PG will verify the merchant and
terminal Details.

/** End of Request Processing**/

```

## Response to Merchant

```

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB
77D27EEAF32EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCB
B03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2
CB2A6EFD72F36FEB044B73810204E69FC577300F9AAF38E044F0A346943
48506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F

```

//To decrypt the above response, Merchant should follow the below step:  
Create the plugin object as,

Merchant must set certain fields in plugin as in request processing. Create the plugin object as,

```
//Initialization
$ resourcePath = "c:\\resourcepath\\";
$ keystorePath = "c:\\ keystorePath\\";
//Terminal Alias Name in the merchant portal for the terminal
$ aliasName = "aliasName";
$myObj =new iPayOabPipe();
$myObj->setAlias(trim($aliasName));
$myObj->setResourcePath(trim($resourcePath));
$myObj->setKeystorePath(trim($keystorePath));
$returnValue = $myObj->parseEncryptedResult($_GET['trandata']);
//The method to be called to decrypt the response sent by Payment Gateway
$result = $myObj->parseEncryptedResult($_GET["trandata"]);
```

**Case1:** If the return value from this method is "0" and \$\_GET["ErrorText"] is null, then Merchant can get the decrypted data of the response fields.

**For Example:**

//Value "CAPTURED" is success and other values must be treated as failure.

```
// To get result,
$myObj->getResult();
// To get payment ID
$myObj->getPaymentId();
//To get Transaction ID
$myObj->getTransId();
// To get Amount
$myObj->getAmt();
//To get token number if registered for tokenization
$myObj->gettokenCustomerId();
```

**Case2:** If \$\_GET["ErrorText"] is not null, then do the following to get response fields.

```
//To get error
$myObj->getError();
```

**Case3:** If the return value from this method is not "0", then Merchant will get the error from below mentioned steps.

```
//To get error
$myObj->getError();
// To get Transaction ID
$myObj->getTransId();
```

```
// To get Payment ID
$myObj->getPaymentId();
```

**Case4:** If trandata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
$_GET['ErrorText'];
```

```
//To get Transaction ID
$_GET['tranid'];
//To get Payment ID
$_GET['paymentid'];
/** End of Response Processing**/
```

### 4.2.2.3 Merchant Hosted Integration supported transactions (Refund, Void, Inquiry)

Merchant can perform the supporting transaction either using HTTP or TCP model.

#### Supported transactions using HTTP model

Merchant can connect iPay Plugin using below step :

```
//Initialization
$resourcePath = "c:\\resourcepath";
//Mandatory
$keystorePath = "c:\\ resourcepath";
//Mandatory
$recieptURL= "http://www.demomerchant.com/result.jsp";
//Mandatory
$errorURL= "http://www.demomerchant.com/error.jsp";
// 2 - Credit/Refund, 3 - Void Purchase, 8 - Inquiry
//Mandatory
$action= "2";
$aliasName = "aliasName"; //Terminal Alias Name. Mandatory
//Transaction Currency. Mandatory for all the transactions
$currency = "currency"; (ex: "512")
//Optional.
$language = "language"; (ex: "USA","EN")
//Transaction Amount. Mandatory for all the transactions
$amount = "10.000 ";
//Merchant Track ID. Mandatory for all the transactions
$trackid = "2015789645632";
//User Defined Fields
$ Udf1= "Udf1";
$ Udf2= "Udf2";
$ Udf3= "Udf3";
$ Udf4= "Udf4";
$ Udf5= "Udf5";

$myObj =new iPayOabPipe();

//Set Values
$myObj->setResourcePath(resourcePath);
$myObj->setKeystorePath(keystorePath);
$myObj->setAlias(aliasName);
```

```

$myObj->setAction( action );
$myObj->setCurrency(currency);
$myObj->setLanguage(language);
$myObj->setResponseURL( recieptURL );
$myObj->setErrorURL(errorURL);
$myObj->setAmt(amount);
$myObj->setTransId(transId);
$myObj->setTrackId(trackid);
$myObj->setUdf1 (Udf1);
$myObj->setUdf2 (Udf2);
$myObj->setUdf3 (Udf3);
$myObj->setUdf4 (Udf4);
$myObj->setUdf5 (Udf5);
$myObj ->performTransactionHTTP();
/** End of Request Processing**/

```

## Response received from Payment Gateway to Merchant

```

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEA
F32EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87CD7
6FE8633E0A1DD90D29338545DA582B0F3500BA9375313637690531C6D7F8F7489C
7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F

```

//To decrypt the above response, Merchant should follow the below step:

Support transaction can be done using Payment ID or Merchant Track ID or Transaction ID

Payment ID: Set text "PaymentID" in the UDF5 using pipe.setUdf5("PaymentID") and Set payment ID value using pipe.setTransId

Merchant Track ID: Set "TrackID" in the UDF5 field using pipe.Udf5("TrackID") and set the track Id value using pipe.setTransId

Sequence Number: Enter text "SeqNum" in the UDF5 field using pipe.Udf5("SeqNum") and set the sequence number value using pipe.setTransId

Transaction ID: set Transaction ID value using pipe.setTransIdCreate the plugin object as,

//Initialization

```
$ resourcePath = "c:\\resourcepath\\";
```

```
$ keystorePath = "c:\\ keystorePath\\";
```

//Terminal Alias Name in the merchant portal for the terminal

```
$ aliasName = "aliasName";
```

```
$myObj =new iPayOabPipe();
```

```
$myObj->setAlias(trim($aliasName));
```

```
$myObj->setResourcePath(trim($resourcePath));
```

```
$myObj->setKeystorePath(trim($keystorePath));
```

```
$returnValue = $myObj->parseEncryptedResult($_GET['trandata']);
```

**Case1:** If the return value from this method is "0"and \$\_GET["ErrorText"] is null, then Merchant can get the decrypted data of the response fields.

### For Example:

```
// To get result,
```

```
$myObj->getResult(); //
// To get payment ID
$myObj->getPaymentId();
//To get Transaction ID
$myObj->getTransId();
// To get Amount
$myObj->getAmt();
```

**Case2:** If \$\_GET["ErrorText"] is not null, then do the following to get response fields.

```
//To get error
$_GET['ErrorText']
```

**Case3:** If the return value from this method is not "0", then Merchant will get the error from below mentioned steps.

```
//To get error
$myObj->getError();
// To get Transaction ID
$myObj->getTransId();
```

```
// To get Payment ID
$myObj->getPaymentId();
```

**Case4:** If tranndata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
$_GET['ErrorText'];
//To get Transaction ID
$_GET['tranid'];
//To get Payment ID
$_GET['paymentid'];
/** End of Response Processing**/
```

### Supported transactions using TCP model

Merchant can connect iPay Plugin using below step :

```
//Initialization
$ resourcePath = "c:\\resourcepath\\";
$ keystorePath = "c:\\ resourcepath\\";
2-Refund, 3- Void Purchase, 8 - Inquiry
$ action="8";
//Terminal Alias Name in the merchant portal for the terminal
$ aliasName = "aliasName";
//Merchant Track ID
$ trackid = "109088888";
//Transaction Currency
$ currency = "currency"; //(ex: "512")
$ language = "language"; //(ex: "USA")
//Transaction Amount
$ amount = "10.000";
```

```
// Transaction ID
$ transId = "2015789645632";
//User Defined Fields
$ Udf1= "Udf1";
$ Udf2= "Udf2";
$ Udf3= "Udf3";
$ Udf4= "Udf4";
$ Udf5= "Udf5";

$myObj=new iPayOabPipe();

//Set Values
$myObj->setResourcePath(resourcePath);
$myObj->setKeystorePath(keystorePath);
$myObj->setAlias(aliasName);
$myObj->setAction( action );
$myObj->setCurrency(currency);
$myObj->setLanguage(language);
$myObj->setAmt(amount);
$myObj->setTransId(transId);
$myObj->setTrackId(trackid);
$myObj->setCard(pan) ;
$myObj->setUdf1 (Udf1) ;
$myObj->setUdf2 (Udf2) ;
$myObj->setUdf3 (Udf3) ;
$myObj->setUdf4 (Udf4) ;
$myObj->setUdf5 (Udf5) ;

//The method to be called is
$myObj ->performTransaction() ;
/** End of Request Processing**/
```

Merchant have to set certain fields in plugin as in request processing.

When performing inquiry or void purchase transaction,Mandatory fields are Action Type,Transaction Amount,UDF5(Merchant Track id/Payment id/Transaction id/Transaction sequence number),Transaction id(setTransid method).

Payment ID: Set text "PaymentID" in the UDF5 using pipe.setUdf5("PaymentID") and Set payment ID value using pipe.setTransId.

Merchant Track ID: Set "TrackID" in the UDF5 field using pipe.Udf5("TrackID") and set the track Id value using pipe.setTransId

Sequence Number: Enter text "SeqNum" in the UDF5 field using pipe.Udf5("SeqNum") and set the sequence number value using pipe.setTransId

Transaction ID: Set Transaction ID value using pipe.setTransId.

## Response from Payment Gateway:

Inquiry transaction results can be extracted from the iPayOabPipe object after calling the method performTransaction().

//Value "CAPTURED" / "SUCCESS" should be considered as success and other values must be treated as failure.

```
if ( $myObj ->performTransaction() != 0) {  
$myObj ->getError_text(); // contains get Error Text  
$myObj ->getError(); // contains get Error Text;
```



```

}
else {

$myObj ->getResult(); // gives the result value
$myObj ->getDate(); // contains post date
$myObj ->getRef(); // contains RRN no generated by PG
$myObj ->getTrackId(); // contains merchant track ID
$myObj ->getTransId(); // contains PG Transaction ID
$myObj ->getAmt(); // contains transaction amount
$myObj ->getPaymentId(); // contains payment ID
$myObj ->getAuth(); // contains Auth code
/** End of Response Processing**/

```

## 4.2.3 Code snippet for ASP Integration

### 4.2.3.1 Bank Hosted Payment Integration

```

iPayOabPipe pipe1= new iPayOabPipe();
//Initialization
pipe1.trackId = '1234567890'; /// Track Id from merchant
pipe1.Alias = 'aliasName'; // Merchant Alias name in the merchant portal
for each terminal pipe1.ResourcePath = 'c://resource//';
pipe1.Action = "1"; // Action code pipe1.Amt = '100';
pipe1.CurrencyCode = '512';// Currency code pipe1.Udf1 = "User Defined
Field";
pipe1.Udf2 = "User Defined Field"; pipe1.Udf3 = "User Defined Field";
pipe1.Udf4 = "User Defined Field"; pipe1.Udf5 = "User Defined Field";
pipe1.KeystorePath = 'c://keystore//';
pipe1.ResponseURL = "http://www.demomergchant/HostedPaymentResult.aspx"; //
Response Url pipe1.ErrorURL =
"http://www.demomergchant/HostedPaymentResult.aspx"; // errorURL
//Tokenization snippet
pipe1.setTokenFlag //1- Registration, 2-Payment
pipe1.setTokenNumber //Set token number for subsequent transaction
short s1 = pipe1.PerformPaymentInitializationHTTP();

```

Merchant can get trandata and tranportal id from ipaypipe inorder to set in post body.

```

String trandata = pipe1.getTranPostData();
String tranportalID = pipe1.getId();
String webAddress = pipe1.getWebAddress();
Merchant Error Url and Merchant Response Url can be set by merchant in post
request body.
/** End of Request Processing**/

```

## Response received from Payment Gateway to Merchant

```
trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EE
AF32EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87C
D76FE8633E0A1DD90D29338545DA582B0F3500BA9375313637690531C6D7F8F7
489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B738102
04E69FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F
```

```
//To decrypt the above response, Merchant should follow the below step:
Merchant have to set certain fields in plugin as in request processing.
Create the plugin object as,
iPayOabPipe pipe1= new iPayOabPipe();
//Initialization
pipe1.Alias = 'aliasName'; // Merchant Alias name in the merchant portal
for each terminal
pipe1.ResourcePath = 'c://respource//';
pipe1.KeystorePath = 'c://keystore//';
```

The method to be called is,

```
pipe1.parseEncryptedRequest(Page.Request.Form["trandata"]);
```

**Case1:** If the return value from this method is "0" and Request.Form("ErrorText") is null, then Merchant can get the decrypted data of the response fields.

### For Example:

```
// To get result,
pipe1.getResult(); //
// To get payment ID
pipe1.getPaymentId();
//To get Transaction ID
pipe1.getTransId();
// To get Amount pipe1.getAmt();
//To get token number if registered for tokenization
pipe1.getTokenCustomerId();
```

**Case2:** If Request.Form("ErrorText") is not null, then do the following to get response fields.

```
//To get error
Page.Request.Form["ErrorText"]
```

**Case3:** If the return value from this method is not "0", then Merchant will get the error from below mentioned steps.

```
//To get error
pipe.getError();
//To get Transaction ID
pipe.getTransId();
```

```
//To get Payment ID
pipe.getPaymentId();
```

**Case4:** If tranndata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
Page.Request.Form["ErrorText"]
//To get Transaction ID
Page.Request.Form["tranid"];
//To get Payment ID
Page.Request.Form["paymentid"];
/** End of Response Processing**/
```

### 4.2.3.2 Merchant Hosted Integration(HTTP) (Purchase)

#### **/\*\* Request Processing\*\*/**

Merchant can connect iPay Plugin using below step

```
iPayOabPipe pipe1= new iPayOabPipe();
//Initialization
pipe1.trackId = '1234567890'; /// Track Id from merchant
pipe1.Alias = 'aliasName'; // Merchant Alias name in the merchant portal
for each terminal pipe1.ResourcePath = 'c://responce//';
pipe1.Action = "1"; // Action code
pipe1.Amt = '100';
pipe1.CurrencyCode = '512';// Currency code
pipe1.Udf1 = "User Defined Field"; pipe1.Udf2 = "User Defined Field";
pipe1.Udf3 = "User Defined Field"; pipe1.Udf4 = "User Defined Field";
pipe1.Udf5 = "User Defined Field"; pipe1.KeystorePath = 'c://keystore//';
pipe1.ResponseURL = "http://www.demomergchant/HostedPaymentResult.aspx"; //
Response Url
pipe1.ErrorURL = "http://www.demomergchant/HostedPaymentResult.aspx"; //
errorURL

pipe1.Card = '821937812371283123'; // card number pipe1.CVV2 = '123'; //
CVV
pipe1.ExpMonth = '12'; // card expire month pipe1.ExpYear = '2019'; // card
expire Year pipe1.Member = 'card holder name'; pipe1.Type = "C";
String url = "", strvall = "";
//Tokenization snippet
pipe1.setTokenFlag //1- Registration, 2-Payment
pipe1.setTokenNumber //Set token number for subsequent transaction
// For normal transaction below method is used to connect Payment Gateway
short s1 = pipe.PerformTransactionHTTP();
```

```
// For 3D secure/VPAS transaction below method is used to connect Payment
Gateway short s1 = pipe.performVbVTransaction();
Merchant can get trandata and tranportal id from ipaypipe inorder to set in
post body.
String trandata = pipe1.getTranPostData();
String tranportalID = pipe1.getId();
String webAddress = pipe1.getWebAddress();
Merchant Error Url and Merchant Response Url can be set by merchant in post
request body.
Once the request received from Merchant, PG will verify the merchant and
terminal then PG proceeds the transaction
//Step 2
```

### Response received from Payment Gateway to Merchant

```
trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EE
AF32EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87C
D76FE8633E0A1DD90D29338545DA582B0F3500BA9375313637690531C6D7F8F7
489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B738102
04E69FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F
```

```
//To decrypt the above response, Merchant should follow the below step:
Merchant have to set certain fields in plugin as in request processing.
Create the plugin object as,
iPayOabPipe pipe1= new iPayOabPipe();
//Initialization
pipe1.Alias = 'aliasName';
// Merchant Alias name in the merchant portal for each terminal
pipe1.ResourcePath = 'c://respource//';
pipe1.KeystorePath = 'c://keystore//';
The method to be called is,
pipe.parseEncryptedResult (Request.Form("trandata"));
Case1: If the return value from this method is "0"and Request.Form("ErrorText")
is null, then Merchant can get the decrypted data of the response fields.
```

### For Example:

```
// To get result,
pipe1.getResult(); //
// To get payment ID
pipe1.getPaymentId();
//To get Transaction ID
pipe1.getTransId();
// To get Amount
pipe1.getAmt();
//To get token number if registered for tokenization
```

```
pipe1.getTokenCustomerId();
```

**Case2:** If Request.Form("ErrorText") is not null, then do the following to get response fields.

```
//To get error
```

```
Page.Request.Form["ErrorText"]
```

**Case3:** If the return value from this method is not "0", then

Merchant will get the error from below mentioned steps. //To get error

```
//To get error pipe1.getError();
```

```
// To get Transaction ID
```

```
pipe1.getTransId();
```

```
// To get Payment ID
```

```
pipe1.getPaymentId();
```

**Case4:** If trandata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
```

```
Page.Request.Form["ErrorText"]
```

```
//To get Transaction ID
```

```
Page.Request.Form["tranid"];
```

```
//To get Payment ID
```

```
Page.Request.Form["paymentid"];
```

```
/** End of Response Processing**/
```

### 4.2.3.3 Merchant Hosted Integration(HTTP) (Refund, Void, Inquiry)

#### **/\*\* Request Processing\*\*/**

Merchant can connect iPay Plugin using below step

```
iPayOabPipe pipe1= new iPayOabPipe();
```

```
//Initialization
```

```
pipe1.trackId = '1234567890'; /// Track Id from merchant
```

```
transId = "2015789645632"; //Original transaction id
```

```
pipe1.TransId(transId);
```

```
pipe1.Alias = 'aliasName'; // Merchant Alias name in the merchant portal  
for each terminal pipe1.ResourcePath = 'c://responce//';
```

```
pipe1.Action = "2"; // Action code
```

```
pipe1.Amt = '100';
```

```
pipe1.CurrencyCode = '512'; // Currency code
```

```

pipe1.Udf1 = "User Defined Field"; pipe1.Udf2 = "User Defined Field";
pipe1.Udf3 = "User Defined Field"; pipe1.Udf4 = "User Defined Field";
pipe1.Udf5 = "User Defined Field"; pipe1.KeystorePath = 'c://keystore//';
pipe1.ResponseURL = "http://www.demomergchant/HostedPaymentResult.aspx"; //
Response Url
pipe1.ErrorURL = "http://www.demomergchant/HostedPaymentResult.aspx"; //
errorURL

```

```

pipe1.Type = "C";
String url = "", strvall = "";
// For normal transaction below method is used to connect Payment Gateway
short s1 = pipe.PerformTransactionHTTP();
// For 3D secure/VPAS transaction below method is used to connect Payment
Gateway short s1 = pipe.performVbVTransaction();
Merchant can get trandata and tranportal id from ipaypipe inorder to set in
post body.

```

```

String trandata = pipe1.getTranPostData();
String tranportalID = pipe1.getId();
String webAddress = pipe1.getWebAddress();
Merchant Error Url and Merchant Response Url can be set by merchant in post
request body.

```

Once the request received from Merchant, PG will verify the merchant and terminal then PG proceeds the transaction

When performing refund, inquiry or void purchase transaction, Mandatory fields are Action Type, Transaction Amount, UDF5 (Merchant Track id/Payment id/Transaction id/Transaction sequence number), Transaction id (setTransid method).

Payment ID: Set text "PaymentID" in the UDF5 using pipe.setUdf5("PaymentID") and Set payment ID value using pipe.setTransId.

Merchant Track ID: Set "TrackID" in the UDF5 field using pipe.Udf5("TrackID") and set the track Id value using pipe.setTransId.

Sequence Number: Enter text "SeqNum" in the UDF5 field using pipe.Udf5("SeqNum") and set the sequence number value using pipe.setTransId.

Transaction ID: Set Transaction ID value using pipe.setTransId

## Response received from Payment Gateway to Merchant

```

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EE
AF32EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87C
D76FE8633E0A1DD90D29338545DA582B0F3500BA9375313637690531C6D7F8F7
489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B738102
04E69FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C
61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F

```

//To decrypt the above response, Merchant should follow the below step:  
Merchant have to set certain fields in plugin as in request processing.

Create the plugin object as,

```
iPayOabPipe pipe1= new iPayOabPipe();
//Initialization
pipe1.Alias = 'aliasName';
// Merchant Alias name in the merchant portal for each terminal
pipe1.ResourcePath = 'c://responce//';
pipe1.KeystorePath = 'c://keystore//';
The method to be called is,
```

```
pipe.parseEncryptedRequest(Request.Form("trandata"));
```

**Case1:** If the return value from this method is "0"and Request.Form("ErrorText") is null, then Merchant can get the decrypted data of the response fields.

#### For Example:

```
// To get result,
pipe1.getResult(); //
// To get payment ID
pipe1.getPaymentId();
//To get Transaction ID
pipe1.getTransId();
// To get Amount
pipe1.getAmt();
```

**Case2:** If Request.Form("ErrorText") is not null, then do the following to get response fields.

```
//To get error
Page.Request.Form["ErrorText"]
```

**Case3:**If the return value from this method is not "0", then

Merchant will get the error from below mentioned steps. //To get error

```
//To get error pipe1.getError();
// To get Transaction ID
pipe1.getTransId();
```

```
// To get Payment ID
pipe1.getPaymentId();
```

**Case4:** If trandata is null, then merchant need to follow below step to get response fields from Payment Gateway.

```
//To get Error
Page.Request.Form["ErrorText"]
//To get Transaction ID
Page.Request.Form["tranid"];
```

```
//To get Payment ID
Page.Request.Form["paymentid"];
/** End of Response Processing**/
Supported transactions using TCP model
```

```
/** Request Processing**/
```

Merchant can connect iPay Plugin using below step :

```
iPayOabPipe pipe1= new iPayOabPipe();
//Initialization
pipe1.trackId = '1234567890'; /// Track Id from merchant
pipe1.Alias = 'aliasName'; // Merchant Alias name in the merchant portal
for each terminal pipe1.ResourcePath = 'c://respource//';
pipe1.Action = "8"; // Action code pipe1.Amt = '100';
pipe1.CurrencyCode = '512';// Currency code pipe1.setTransId("123456"); //
Already completed transaction ID pipe1.Udf1 = "User Defined Field";
pipe1.Udf2 = "User Defined Field";
pipe1.Udf3 = "User Defined Field"; pipe1.Udf4 = "User Defined Field";
pipe1.Udf5 = "User Defined Field"; pipe1.KeystorePath = 'c://keystore//';
pipe1.TransId = '123456'; // Transaction ID pipe1.Type = "C";
String url = "", strvall = "";
// For normal transaction below method is used to connect Payment Gateway
short s1 = pipe1.PerformTransaction();
```

When performing inquiry or void purchase transaction, Mandatory fields are Action Type, Transaction Amount, UDF5 (Merchant Track id/Payment id/Transaction id/Transaction sequence number), Transaction id (setTransid method).

Payment ID: Set text "PaymentID" in the UDF5 using

pipe.setUdf5("PaymentID") and Set payment ID value using pipe.setTransId

Merchant Track ID: Set "TrackID" in the UDF5 field using

pipe.Udf5("TrackID") and set the track Id value using pipe.setTransId

Sequence Number: Enter text "SeqNum" in the UDF5 field using

pipe.Udf5("SeqNum") and set the sequence number value using pipe.setTransId

Transaction ID: Set Transaction ID value using pipe.setTransId

## Response received from Payment Gateway to Merchant

Response from Payment Gateway:

Inquiry transaction results can be extracted from the iPayOabPipe object after calling the method performTransaction().

```
//Value "CAPTURED" / "SUCCESS" should be considered as success and other
values must be treated
as failure.
```

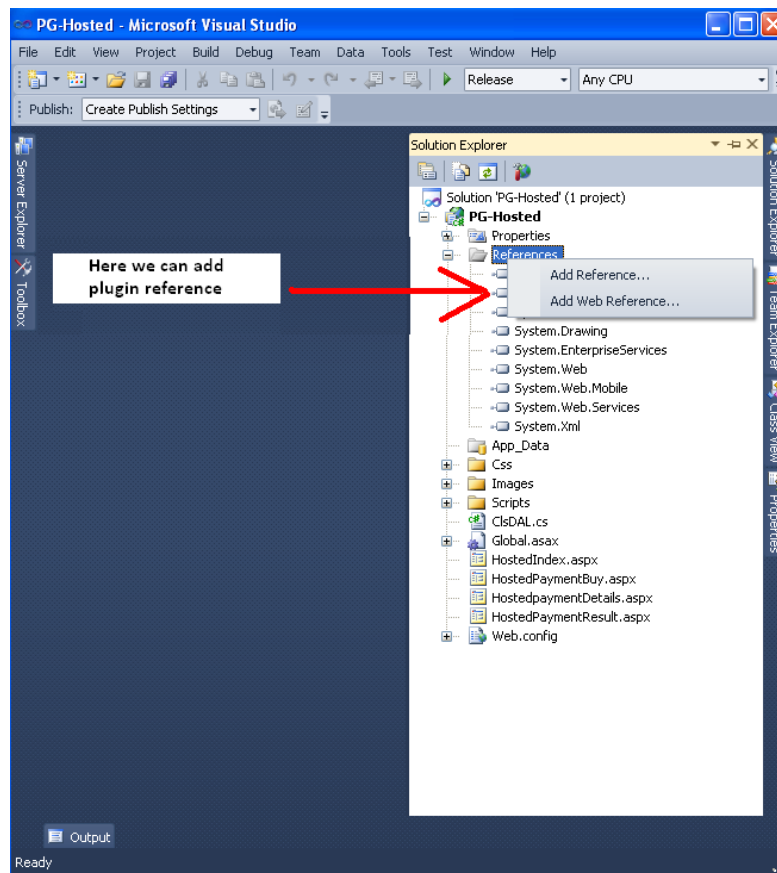


```
short s1= pipel. performTransaction(); // To connect Payment Gateway
if (s1 != 0) {
    pipel.getError_text(); // contains get Error Text
    pipel.getError(); // contains get Error Text;
}
else {
    pipel.getResult(); // gives the result value pipel.getDate(); // contains
    post date
    pipel. getRef(); // contains RRN no generated by PG
    pipel. getTrackId(); // contains merchant track ID
    pipel.getTransId(); // contains PG Transaction ID
    pipel.getAmt(); // contains transaction amount pipel.getPaymentId(); //
    contains payment ID
    pipel.getAuth(); // contains Auth code
}
/** End of Response Processing**/
```

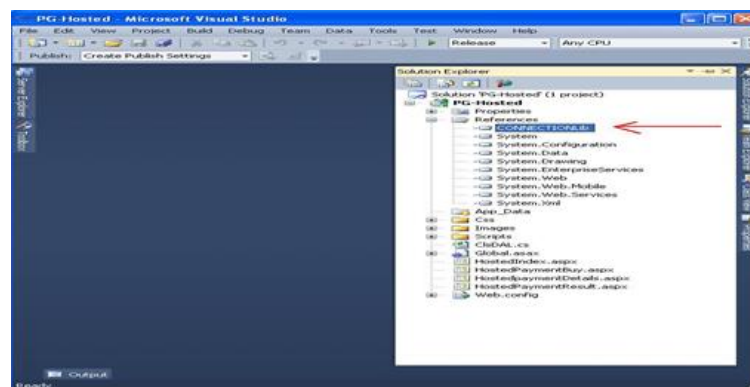
## Registering ASP. NET Plug-in the Merchant Server

Below steps explain the Configuration for 'Plug-in' to integrate with the merchant application.

1. Extract the **iPayOabPipe.zip** folder.
2. Rename **ikvm.txt** and **ikvmstub.txt** to **ikvm.exe** and **ikvmstub.exe**.
3. Add the (IKVM.OpenJDK.Core.dll , IKVM.Runtime.dll and iPayOabPipe.dll) dll's in the reference from extracted folder
4. Add the plug-in reference with in merchant application
5. Right click on the **Reference folder**-> Select **Add Reference**

**Figure 12: Add Reference screen**

The remaining reference(zip folder dll's) will automatically add in the merchant application(bin folder from solution explorer).  
This activity completes the task of integrating plug-in with your merchant application.

**Figure 13: DLL Reference screen**

This activity completes the task of integrating plug-in with your merchant application.

## 5 APPENDIX I

---

*The below are the details of the data fields that are exchanged between the Merchant web application and FSS Payment Gateway.*

---

5.1	BANK HOSTED TRANSACTION .....	52
5.1.1	<i>Request from Merchant to PG</i> .....	52
5.1.2	<i>Response from PG to Merchant</i> .....	54
5.2	MERCHANT HOSTED TRANSACTION .....	55
5.2.1	<i>Request from Merchant to PG</i> .....	55
5.2.2	<i>Response from PG to Merchant</i> .....	57
5.2.3	<i>Final Response and Description</i> .....	58
5.3	RESULT .....	59
5.4	ERROR CODES .....	60
5.5	SAMPLE DEMO PAGE NAVIGATION .....	73
5.5.1	<i>Hosted Payment Integration</i> .....	73
5.6	BEST PRACTICES .....	74
5.6.1	<i>Important Notes and Information</i> .....	76
5.6.2	<i>Essentials</i> .....	77
5.6.3	<i>Frequently Asked Queries (General)</i> .....	78

## 5.1 Bank Hosted Transaction

### 5.1.1 Request from Merchant to PG

No	Field	Type	Max	Mandatory	Description
1	Tran portal ID	AN	16	Yes	Unique ID assigned for a Terminal under a merchant. (ex: 101001).
2	Transaction Action Type	N	2	Yes	Ex: 1 – Purchase, 2 – Refund/Credit.
3	Response URL	Text	255	Yes	Ex: <a href="http://www.merchantdemo.com/response">http://www.merchantdemo.com/response</a> ; if authentication and authorization successful at the Payment Gateway then response would be sent to this Response URL.
4	Error URL	Text	255	Yes	Ex: <a href="http://www.merchantdemo.com/error">http://www.merchantdemo.com/error</a> ; if authentication and authorization failed at the Payment Gateway then response would be sent to this error URL.
5	Amount	N	10	Yes	Transaction amount to be charged to the Customer for the product purchased (ex: 10.35).
6	Currency	N	3	Yes	Transaction Currency can be any Three Digit Numeric Code with the Currency allowing a maximum of 3 Decimal digits. (Ex. Currency Code : 512)
7	Track ID	AN	40	Yes	This must be a unique track ID generated by merchant for every transaction, this must be at least of 12 length (ex: 112312312313)
8	UDF 1	AN	255	No	Allows Merchant to send any additional information
9	UDF 2	AN	255	No	Allows Merchant to send any additional information
10	UDF 3	AN	255	No	Allows Merchant to send any additional information
11	UDF 4	AN	255	No	Allows Merchant to send any additional information

No	Field	Type	Max	Mandatory	Description
12	UDF 5	AN	255	No	Allows Merchant to send any additional information
13	Language	Text	3	No	English – USA , EN

## 5.1.2 Response from PG to Merchant

No	Field	Type	Mandatory	Description
1	Payment ID	N	No	Unique ID which is generated if the request from the merchant received and successfully validated at the payment gateway.
2	Result	Text	Yes	Please refer to the <a href="#">Result</a> Section.
3	Reference ID	N	Conditional	This is retrieval reference Number (RRN) generated by PG/Interchange for all transactions
4	Transaction ID	N	Yes	Unique ID generated by Payment Gateway for the transaction.
5	Customer ID	N	No	Unique ID generated by Payment Gateway In case of recurring Payment enabled and opted by card holder
6	Post Date	Date	No	Transaction postdate.
7	UDF 1 [Text Field-Optional]	Text	Conditional	Additional information
8	UDF 2 [Text Field-Optional]	Text	Conditional	Additional information
9	UDF 3 [Text Field-Optional]	Text	Conditional	Additional information
10	UDF 4 [Text Field-Optional]	Text	Conditional	Additional information
11	UDF 5 [Text Field-Optional]	Text	Conditional	Additional information
12	Amount	N	Yes	Transaction Amount with three decimal values

No	Field	Type	Mandatory	Description
13	Error	Text	No	In case of transaction not being successful, Payment Gateway shall send the Error code corresponding to the failure.
14	Error Text	Text	No	In case of transaction not being successful, Payment Gateway shall send the Error description corresponding to the failure.
15	Track ID	Text	Yes	Merchant Reference ID sent in the transaction request.

## 5.2 Merchant Hosted Transaction

### 5.2.1 Request from Merchant to PG

No	Field	Type	Max	Mandatory	Description
1	Tran portal ID	AN	16	Yes	Unique ID assigned for a Terminal under a merchant. (ex: 101001).
2	Transaction Action Type	N	2	Yes	Ex: 1 – Purchase, 2 – Refund/Credit., 8 – Inquiry
3	Response URL	Text	255	Yes	Ex: <a href="http://www.merchantdemo.com/response">http://www.merchantdemo.com/response</a> ; if authentication and authorization successful at the Payment Gateway then response would be sent to this Response URL.
4	Error URL	Text	255	Yes	Ex: <a href="http://www.merchantdemo.com/error">http://www.merchantdemo.com/error</a> ; if authentication and authorization failed at the Payment Gateway then response would be sent to this error URL.
5	Card Number	N	19	Yes	Card Number using which the Transaction needs to be carried out.
6	CVV2	N	3	Yes	CVV value of the Card Number
7	Expiry Month	N	2	Yes	Expiry Month Value of the Card Number
8	Expiry Year	N	4	Yes	Expiry Year value of the Card Number

No	Field	Type	Max	Mandatory	Description
9	Type	Text	2	Yes	Type of the Card – C - Credit D – Debit – this need not be shared, better remove it
10	Member	Text	255	Yes	Card Holder Name
11	Password	AN	15	Yes	Password for the Transaction Terminal, this will be fetched from the resource file
12	Amount	N	10	No	Transaction amount to be charged to the Customer for the product purchased (ex: 100.35).
13	Currency	N	3	Yes	Transaction Currency can be any Three Digit Numeric Code with the Currency allowing a maximum of 3 Decimal digits. (Ex. Currency Code : 512).
14	Track ID	AN	40	Yes	This must be a unique track ID generated by merchant for every transaction, this must be at least of 12 length (ex: 112312312313). Merchant should ensure the uniqueness.
15	UDF 1	AN	255	No	If recurring Payment is enabled this field is mandatory.
16	UDF 2	AN	255	No	Allows Merchant to send any additional information.
17	UDF 3	AN	255	No	Allows Merchant to send any additional information.
18	UDF 4	AN	255	No	Allows Merchant to send any additional information.
19	UDF 5	AN	255	No	Allows Merchant to send any additional information.
20	Language	Text	3	No	USA (English), EN (English).
21	Token Flag	Numeric	2	No	TokenFlag will be used to register/deregister and perform the tokenization transaction
22	Token Number	Numeric	24	No	Token number is used to perform token deregistration and the tokenization transaction



## 5.2.2 Response from PG to Merchant

No	Field	Type	Mandatory	Description
1	Payment ID	N	No	Unique ID which is generated if the request from the merchant received and successfully validated at the payment gateway.
2	Result	Text	Yes	Please refer to the <a href="#">Result</a> Section.
3	Reference ID	N	Conditional	This is retrieval reference Number (RRN) generated by PG/Interchange for all transactions
4	Transaction ID	N	Yes	Unique ID generated by Payment Gateway for the transaction.
5	Customer ID(In case of Standing Instruction)	N	No	Unique ID generated by Payment Gateway for Standing Instruction.
6	Post Date	Date	No	Transaction postdate.
7	UDF 1 [Text Field-Optional]	Text	Conditional	Additional information
8	UDF 2 [Text Field-Optional]	Text	Conditional	Additional information
9	UDF 3 [Text Field-Optional]	Text	Conditional	Additional information
10	UDF 4 [Text Field-Optional]	Text	Conditional	Additional information
11	UDF 5 [Text Field-Optional]	Text	Conditional	Additional information
12	Amount	N	Yes	Transaction Amount with three decimal values
13	Error	Text	No	In case of transaction not being successful, Payment Gateway shall send the Error code with Payment ID, Result, UDF fields and Amount corresponding to the failure.
14	Error Text	Text	No	In case of transaction not being successful, Payment Gateway shall send the Error

No	Field	Type	Mandatory	Description
				description with Payment ID, Result, UDF fields and Amount corresponding to the failure.

### 5.2.3 Final Response and Description

No	Final Response	Description
1	SUCCESS	Result of successful transaction. If the transaction processed successfully, then the Payment Gateway will send the result as "SUCCESS".
2	Result	Result of failure transaction. If any error occurred during transaction process, then the Payment Gateway will send the result as "FAILURE".

## 5.3 Result

The result parameter in the OAB Payment Gateway transaction response helps the merchant to determine the transaction status. The merchant should first check for any error received; if no error received, check for the result parameter and basis the same, determine whether the transaction is successful or failed. Mentioned below are the values that could be sent in the response by the Payment Gateway in the result parameter to the merchant response URL.

Result Code	Description
CAPTURED	Transaction was captured (successful) (For Action Code "1 - Purchase", "2 - Refund")
NOT CAPTURED	Transaction was not captured (failed) (For Action Code "1 - Purchase" For Action Code "2 - Refund")
SUCCESS	Transaction was Available and approved (successful) (For Action Code "8 - Inquiry")
DENIED BY RISK	OAB PG will return this when the transaction is denied due to violation of any Risk profile parameter for below action codes "1- Purchase" "2 - Refund" "3 - Void Purchase"
HOST TIMEOUT	The Issuer Bank did not respond within the transaction Time Limit
FAILURE(SUSPECT)	The Transaction was failed For Action Code "8 - Inquiry"
AUTH ERROR	OAB PG will return this when the transaction is failed due to any internal validations.

## 5.4 Error Codes

If an error occurs during the processing of the transaction, then the response format will contain a single string indicating that an error occurred. All Response Error Messages begins with a! ERROR! Identifier. Therefore, it is important for the developer to first look at the actual response message string to determine if an error occurred. The error codes are listed below.

The following table contains the known error codes and their descriptions:

Error Code	Error Description
IPAY0100001	Missing error URL.
IPAY0100002	Invalid error URL.
IPAY0100003	Missing response URL.
IPAY0100004	Invalid response URL.
IPAY0100005	Missing tranportal id.
IPAY0100006	Invalid tranportal ID.
IPAY0100007	Missing transaction data.
IPAY0100008	Terminal not enabled.
IPAY0200001	Problem occurred while getting terminal.
IPAY0100009	Institution not enabled.
IPAY0200002	Problem occurred while getting institution details.
IPAY0100010	Institution has not enabled for the encryption process.
IPAY0200003	Problem occurred while getting merchant details.
IPAY0100011	Merchant has not enabled for encryption process.
IPAY0100012	Empty terminal key.
IPAY0100013	Invalid transaction data.
IPAY0100014	Terminal Authentication requested with invalid tranportal ID data.
IPAY0100015	Invalid tranportal password.
IPAY0100016	Password security not enabled.
IPAY0200004	Problem occurred while getting password security rules.
IPAY0200005	Problem occurred while updating terminal details.
IPAY0100018	Terminal password expired.
IPAY0200006	Problem occurred while verifying tranportal password.

IPAY0100020	Invalid action type.
IPAY0100022	Invalid currency.
IPAY0100023	Missing amount.
IPAY0100025	Invalid amount or currency.
IPAY0100026	Invalid language id
IPAY0100028	Invalid user defined field1.
IPAY0100029	Invalid user defined field2.
IPAY0100031	Invalid user defined field4.
IPAY0100032	Invalid user defined field5.
IPAY0100033	Terminal action not enabled.
IPAY0100034	Currency code not enabled.
IPAY0100035	Problem occurred during merchant hashing process.
IPAY0100036	UDF Mismatched'
IPAY0100038	Unable to process the request.
IPAY0100039	Invalid payment id.
IPAY0200009	Problem occurred while getting payment details.
IPAY0100041	Payment details missing.
IPAY0100042	Transaction time limit exceeds.
IPAY0200011	Problem occurred while getting IP block details.
IPAY0100043	IP address is blocked already
IPAY0100044	Problem occurred while loading payment page.
IPAY0100045	Denied by Risk
IPAY0200013	Problem occurred while updating description details in payment log.
IPAY0100047	Payment Page validation failed due to invalid Order Status:
IPAY0100049	Transaction declined due to exceeding OTP attempts
IPAY0200015	Problem occurred while getting terminal details.
IPAY0100050	Invalid terminal key.
IPAY0100051	Missing terminal key.
IPAY0100053	Problem occurred while processing direct debit.
IPAY0100054	Payment details not available
IPAY0100056	Instrument not allowed in Terminal and Brand
IPAY0200016	Problem occurred while getting payment instrument.
IPAY0200018	Problem occurred while getting transaction details

IPAY0100057	Transaction denied due to invalid processing option action code
IPAY0100058	Transaction denied due to invalid instrument
IPAY0100059	Transaction denied due to invalid currency code.
IPAY0100060	Transaction denied due to missing amount.
IPAY0100061	Transaction denied due to invalid amount.
IPAY0100062	Transaction denied due to invalid Amount/Currency.
IPAY0100063	Transaction denied due to invalid track ID');
IPAY0100064	Transaction denied due to invalid UDF1
IPAY0100065	Transaction denied due to invalid UDF2
IPAY0100066	Transaction denied due to invalid UDF3
IPAY0100067	Transaction denied due to invalid UDF4
IPAY0100068	Transaction denied due to invalid UDF5
IPAY0100069	Missing payment instrument.
IPAY0100070	Transaction denied due to failed card check digit calculation.
IPAY0100071	Transaction denied due to missing CVD2.
IPAY0100072	Transaction denied due to invalid CVD2.
IPAY0100073	Transaction denied due to invalid CVV.
IPAY0100074	Transaction denied due to missing expiry year.
IPAY0100075	Transaction denied due to invalid expiry year.
IPAY0100076	Transaction denied due to missing expiry month.
IPAY0100077	Transaction denied due to invalid expiry month.
IPAY0100078	Transaction denied due to missing expiry day.
IPAY0100079	Transaction denied due to invalid expiry day.
IPAY0100080	Transaction denied due to expiration date.
IPAY0100081	Card holder name is not present
IPAY0100082	Card address is not present
IPAY0100083	Card postal code is not present
IPAY0100084	AVS Check : Fail
IPAY0100085	Electronic Commerce Indicator is invalid
IPAY0100086	Transaction denied due to missing CVV.
IPAY0100087	Card pin number is not present
IPAY0100088	Empty mobile number.
IPAY0100089	Invalid mobile number.

IPAY0100091	Invalid MMID.
IPAY0100092	Empty OTP number.
IPAY0100093	Invalid OTP number.
IPAY0100095	Terminal inactive.
IPAY0100096	IMPS for Institution Not Active for Transaction request, Institution :
IPAY0100098	Terminal Action not enabled for Transaction request, Terminal "termid" ,Tran Action : "action",-'opted action was not supported by that terminal
IPAY0100099	Terminal Payment Instrument not enabled for Transaction request, Terminal "termid " , Tran Instrument : "PAYMENT_INSTRUMENT"
IPAY0100100	Problem occurred while authorize
IPAY0200019	Problem occurred while getting risk profile details
IPAY0100102	Denied by risk : Maximum Floor Limit Check - Fail
IPAY0100103	Transaction denied due to Risk : Maximum transaction count
IPAY0100104	Transaction denied due to Risk : Maximum processing amount
IPAY0200022	Problem occurred while getting currency.
IPAY0100105	Action type not supported by maestro brand.
IPAY0100106	Invalid payment instrument
IPAY0200024	Problem occurred while getting brand rules details.
IPAY0100107	Instrument not enabled.
IPAY0200025	Problem occurred while getting terminal details.
IPAY0100109	Invalid subsequent transaction, payment id is null or empty.
IPAY0200026	Problem occurred while getting transaction log details.
IPAY0200027	Missing encrypted card number.
IPAY0100111	Card decryption failed.
IPAY0100113	"transaction id" is a subsequent transaction, but original transaction id is invalid :
IPAY0100114	Duplicate Record
IPAY0100115	Transaction denied due to missing original transaction id.
IPAY0100116	Transaction denied due to invalid original transaction id.
IPAY0100118	Transaction denied due to card number length error
IPAY0100119	Transaction denied due to invalid card number
IPAY0100120	Transaction denied due to invalid payment instrument for brand data.

IPAY0100121	Transaction denied due to invalid card holder name.
IPAY0100122	Transaction denied due to invalid address.
IPAY0100123	Transaction denied due to invalid postal code.
IPAY0100124	Problem occurred while validating transaction data
IPAY0100125	Payment instrument not enabled.
IPAY0100126	Brand not enabled.
IPAY0100127	Problem occurred while doing validate original transaction
IPAY0100128	Transaction denied due to Institution ID mismatch
IPAY0100129	Transaction denied due to Merchant ID mismatch
IPAY0100130	Transaction denied due to Terminal ID mismatch
IPAY0100131	Transaction denied due to Payment Instrument mismatch
IPAY0100132	Transaction denied due to Currency Code mismatch
IPAY0100133	Transaction denied due to Card Number mismatch
IPAY0100134	Transaction denied due to invalid Result Code
IPAY0100135	Problem occurred while doing perform action code reference id (Validate Original Transaction)
IPAY0200028	Problem occurred while loading default institution configuration (Validate Original Transaction)
IPAY0100136	Transaction denied due to previous capture check failure ( Validate Original Transaction )
IPAY0100138	Transaction denied due to capture amount versus auth amount check failure ( Validate Original Transaction )
IPAY0100139	Transaction denied due to void amount versus original amount check failure (Validate Original Transaction )
IPAY0100140	Transaction denied due to previous void check failure (Validate Original Transaction )
IPAY0100141	Transaction denied due to authorization already captured (Validate Original Transaction)
IPAY0100142	Problem occurred while validating original transaction
IPAY0200030	No external connection details for Extr Conn id :
IPAY0200031	Alternate external connection details not found for the alt Extr Conn id :
IPAY0100143	Transaction action is null
IPAY0200033	Problem occurred while getting vpas log details.
IPAY0200034	Problem occurred while getting details from VPASLOG table



IPAY0100144	ISO MSG is null. See log for more details!
IPAY0100145	Problem occurred while loading default messages in ISO Formatter
IPAY0100147	Problem occurred while formatting purchase request in B24 ISO Message Formatter
IPAY0100148	Problem occurred while hashing E-com PIN
IPAY0100150	Problem occurred while formatting Reverse purchase request in B24 ISO Message Formatter
IPAY0100152	Problem occurred while formatting authorization request in B24 ISO Message Formatter
IPAY0100153	Problem occurred while formatting Capture request in B24 ISO Message Formatter
IPAY0100155	Problem occurred while formatting reverse authorization request in B24 ISO Message Formatter
IPAY0100156	Problem occurred while formatting Reverse Capture request in B24 ISO Message Formatter
IPAY0100157	Problem occurred while formatting vpas capture request in B24 ISO Message Formatter
IPAY0100159	External message system error
IPAY0100160	Unable to process the transaction.
IPAY0100162	Problem occurred while validating IMPS
IPAY0100163	Problem occurred during transaction.
IPAY0100164	Transaction Not Processed due to Wrong ECI value
IPAY0100166	Transaction Not Processed due to Empty Authentication Status
IPAY0100167	Transaction Not Processed due to Invalid Authentication Status
IPAY0100168	Transaction Not Processed due to Empty Enrollment Status
IPAY0100169	Transaction Not Processed due to Invalid Enrollment Status
IPAY0100170	Transaction Not Processed due to invalid CAVV
IPAY0100171	Transaction Not Processed due to Empty CAVV
IPAY0200035	Amex Payment log details not available.
IPAY0200036	Problem occurred while getting amex payment log details.
IPAY0100172	Problem occurred while converting amount.
IPAY0100173	Problem occurred while building refund request.
IPAY0100174	Problem occurred while calling amex web service.
IPAY0100175	Problem occurred in refund process.
IPAY0100017	Inactive terminal.

IPAY0100019	Invalid log in attempt.
IPAY0100021	Missing currency.
IPAY0100024	Invalid amount.
IPAY0100027	Invalid track id.
IPAY0100030	Invalid user defined field3.
IPAY0200007	Problem occurred while validating payment details
IPAY0200008	Problem occurred while verifying payment details.
IPAY0100037	Payment id missing.
IPAY0100040	Transaction in progress in another tab/window.
IPAY0200010	Problem occurred while updating payment details.
IPAY0200012	Problem occurred while updating payment log IP details.
IPAY0100046	Payment option not enabled.
IPAY0100048	Cancelled
IPAY0200014	Problem occurred during merchant response.
IPAY0100052	Problem occurred during merchant response encryption.
IPAY0100055	Invalid Payment Status
IPAY0200017	Problem occurred while getting payment instrument list
IPAY0100090	Empty MMID.
IPAY0100094	Sorry, this instrument is not handled
IPAY0100097	IMPS for Terminal Not Active for Transaction request, Terminal :
IPAY0100101	Denied by risk: Risk Profile does not exist
IPAY0200020	Problem occurred while performing transaction risk check
IPAY0200021	Problem occurred while performing risk check
IPAY0200023	Problem occurred while determining payment instrument
IPAY0100108	Perform risk check: Failed
IPAY0100110	Invalid subsequent transaction, Tran Ref id is null or empty.
IPAY0100112	Problem occurred in method loading original transaction data (card number, exp month / year) for orig_tran_id
IPAY0100117	Transaction denied due to missing card number.
IPAY0100137	Transaction denied due to credit amount greater than auth amount check failure (Validate Original Transaction )
IPAY0200029	Problem occurred while getting external connection details.
IPAY0200032	Problem occurred while getting external connection details for Extr Conn id :

IPAY0100146	Problem occurred while encrypting PIN
IPAY0100149	Invalid PIN Type
IPAY0100151	Problem occurred while formatting Credit request in B24 ISO Message Formatter
IPAY0100154	Problem occurred while formatting Reverse Credit request in B24 ISO Message Formatter
IPAY0100158	Host timeout
IPAY0100161	Merchant is not allowed for encryption process.
IPAY0100165	Transaction Not Processed due to Empty ECI value
IPAY0100176	Decrypting transaction data failed.
IPAY0100177	Invalid input data received.
IPAY0100178	Merchant encryption enabled.
IPAY0100179	IVR not enabled.
IPAY0100180	Authentication not available.
IPAY0100181	Card encryption failed.
IPAY0200037	Error occurred while getting Merchant ID
IPAY0100182	Vpas merchant not enabled.
IPAY0200038	Problem occurred while getting vpas merchant details.
IPAY0100183	Error occurred Due to byte PAREq is null
IPAY0100184	Error occurred while Parsing PAREq
IPAY0100185	Problem occurred while authentication
IPAY0100186	Encryption enabled.
IPAY0100187	Customer ID is missing for Faster Checkout.
IPAY0100188	Transaction Mode(FC) is missing for Faster Checkout.
IPAY0100189	Transaction denied due to brand directory unavailable
IPAY0200039	Problem occurred while getting Faster Checkout details.
IPAY0100190	Transaction denied due to Risk : Maximum transaction count
IPAY0100191	Denied by risk : Negative Card check - Fail
IPAY0100192	Transaction Not Processed due to Empty XID
IPAY0100193	Transaction Not Processed due to invalid XID
IPAY0100202	Error occurred in Determine Payment Instrument
IPAY0100194	Transaction denied due to Risk : Minimum Transaction Amount processing
IPAY0100195	Transaction denied due to Risk : Maximum credit processing amount

IPAY0100196	Transaction denied due to Risk : Maximum processing amount
IPAY0100197	Transaction denied due to Risk : Maximum debit amount
IPAY0100198	Transaction denied due to Risk : Transaction count limit exceeded for the IP
IPAY0100199	Transaction denied due to previous credit check failure ( Validate Original Transaction )
IPAY0100200	Denied by risk : Negative BIN check - Fail
IPAY0100201	Denied by risk: Declined Card check – Fail
IPAY0100203	Problem occurred while doing perform transaction
IPAY0100204	Missing payment details
IPAY0100205	Problem occurred while getting PARES details
IPAY0100206	Problem occurred while getting currency minor digits
IPAY0100207	Bin range not enabled
IPAY0100208	Action not enabled
IPAY0100209	Institution config not enabled
IPAY0100210	Problem occurred during veres process
IPAY0100211	Problem occurred during pareq process
IPAY0100212	Problem occurred while getting veres
IPAY0100213	Problem occurred while processing the hosted transaction request
IPAY0100214	Problem occurred while verifying tranportal id
IPAY0100215	Invalid tranportal id
IPAY0100216	Invalid data received
IPAY0100217	Invalid payment detail
IPAY0100218	Invalid brand id
IPAY0100219	Missing card number
IPAY0100220	Invalid card number
IPAY0100221	Missing card holder name
IPAY0100222	Invalid card holder name
IPAY0100223	Missing cvv
IPAY0100224	Invalid cvv
IPAY0100225	Missing card expiry year
IPAY0100226	Invalid card expiry year
IPAY0100227	Missing card expiry month

IPAY0100228	Invalid card expiry month
IPAY0100229	Invalid card expiry day
IPAY0100230	Card expired
IPAY0100231	Invalid user defined field
IPAY0100232	Missing original transaction id
IPAY0100233	Invalid original transaction id
IPAY0100234	Problem occurred while formatting Reverse Capture request in VISA ISO Message Formatter
IPAY0100235	Problem occurred while formatting Reverse Credit request in VISA ISO Message Formatter
IPAY0100236	Problem occurred while formatting Reverse Credit request in VISA ISO Message Formatter
IPAY0100237	Problem occurred while formatting Reverse purchase request in VISA ISO Message Formatter
IPAY0100238	Problem occurred while formatting Capture request in VISA ISO Message Formatter
IPAY0100239	Problem occurred while formatting authorization request in VISA ISO Message Formatter
IPAY0100240	Problem occurred while formatting Credit request in VISA ISO Message Formatter
IPAY0100241	Problem occurred while formatting purchase request in VISA ISO Message Formatter
IPAY0100242	RC_UNAVAILABLE
IPAY0100243	NOT SUPPORTED
IPAY0100244	Payment Instrument Not Configured
IPAY0100245	Problem occurred while sending/receiving ISO message
IPAY0100246	Problem occurred while doing perform ip risk check
IPAY0100247	PARES message format is invalid
IPAY0100248	Problem occurred while validating PARES message format
IPAY0100249	Merchant response url is down
IPAY0100250	Payment details verification failed
IPAY0100251	Invalid payment data
IPAY0100252	Missing veres
IPAY0100253	Problem occurred while cancelling the transaction
IPAY0100254	Merchant not enabled

IPAY0100255	External connection not enabled
IPAY0100256	Payment encryption failed
IPAY0100257	Brand rules not enabled
IPAY0100258	Certification verification failed
IPAY0100259	Problem occurred during merchant hashing process
IPAY0100260	Payment option(s) not enabled
IPAY0100261	Payment hashing failed
IPAY0100262	Problem occurred during VEREQ process
IPAY0100263	Transaction details not available
IPAY0100264	Signature validation failed
IPAY0100265	PARES validation failed
IPAY0100266	Brand directory unavailable
IPAY0100267	PARES status not successful
IPAY0100268	3d secure not enabled for the brand
IPAY0100269	Invalid card check digit
IPAY0100270	pares not successful
IPAY0100271	Problem occurred while formatting purchase request in MASTER ISO Message Formatter
IPAY0100272	Problem occurred while validating xml message format
IPAY0100273	Problem occurred while validation VERES message format
IPAY0100274	VERES message format is invalid
IPAY0100275	Problem occurred while formatting Credit request in MASTER ISO Message Formatter
IPAY0100276	Problem occurred while formatting Reverse purchase request in MASTER ISO Message Formatter
IPAY0100277	Problem occurred while formatting Reverse Credit request in MASTER ISO Message Formatter
IPAY0100278	Problem occurred while formatting reverse authorization request in MASTER ISO Message Formatter
IPAY0100279	Problem occurred while formatting Reverse Capture request in MASTER ISO Message Formatter
IPAY0100280	Problem occurred while formatting Capture request in MASTER ISO Message Formatter
IPAY0100281	Transaction Denied due to missing Master Brand
IPAY0100282	Transaction Denied due to missing Visa Brand

IPAY0200040	Problem occurred while performing card risk check
IPAY0200041	Problem occurred while getting institution configuration
IPAY0200042	Problem occurred while getting brand
IPAY0200043	Problem occurred while getting bin range details
IPAY0200044	Problem occurred while adding transaction log details
IPAY0200045	Problem occurred while updating VPASLOG table
IPAY0200046	Unable to update VPASLOG table, payment id is null
IPAY0200047	Problem occurred while getting details from VPASLOG table for payment id
IPAY0200048	Problem occurred while getting details from VPASLOG table
IPAY0200049	Card number is null. Unable to update risk factors in negative card table & declined card table
IPAY0200050	Problem occurred while updating risk in negative card details
IPAY0200051	Problem occurred while updating risk in declined card table
IPAY0200052	Problem occurred while updating risk factor
IPAY0200053	Problem occurred while updating payment log currency details
IPAY0200054	Problem occurred while inserting currency conversion currency details
IPAY0200055	Problem occurred while updating currency conversion currency details
IPAY0200068	Problem occurred while validating IP address blocking
IPAY0200069	Problem occurred while updating payment log card details
IPAY0200070	Problem occurred while updating ipblock details
IPAY0200071	Problem occurred during authentication
IPAY0200072	Payment log details not available
IPAY0100284	Invalid subsequent transaction, track id is null or empty
IPAY0100285	Transaction denied due to invalid original transaction
IPAY0100286	Unknown IMPS Tran Action Code encountered
IPAY0100287	Terminal Action not enabled for Transaction request, Terminal
IPAY0200056	Problem occurred while getting brand details
IPAY0200057	Problem occurred while getting external connection details
IPAY0200058	Problem occurred while updating message log 2fa details
IPAY0200059	Problem occurred while updating vpas details
IPAY0200060	Problem occurred while adding vpas details

IPAY0200061	Problem occurred during batch 2fa process
IPAY0200062	Problem occurred while getting brand rules details
IPAY0200063	Problem occurred while updating payment log process code details
IPAY0200064	Problem occurred while updating payment log process code and ip details
IPAY0200065	Problem occurred while updating payment log description details
IPAY0200066	Problem occurred while updating payment log instrument details
IPAY0200067	Problem occurred while updating payment log udf Fields
IPAY0100288	Terminal Payment Instrument not enabled for Transaction request, Terminal: termId, Tran Instrument: instrument name
IPAY0100289	Transaction denied due to Risk: Maximum credit amount
IPAY0100283	Problem occurred in determine payment instrument







## 5.5 Sample Demo Page Navigation

### 5.5.1 Hosted Payment Integration

#### Merchant's Product page

- If Customer visits the merchant site, the merchandise in the Merchant page is displayed for online shopping.

**ONLINE SHOPPING**

 Price: 15.Rs Qty: <input type="text" value="1"/> Add To Cart <input checked="" type="checkbox"/>	 Price: 10.Rs Qty: <input type="text" value="1"/> Add To Cart <input checked="" type="checkbox"/>
 Price: 20.Rs Qty: <input type="text" value="1"/> Add To Cart <input checked="" type="checkbox"/>	 Price: 25.Rs Qty: <input type="text" value="1"/> Add To Cart <input checked="" type="checkbox"/>

[Purchase](#)

**Figure 14: Online Shopping Merchant Page**

- ☑ **Note:** Amount will be in OMR this is a sample screen only

#### Products selected by the customer for purchase:

- If the customer clicks "**Purchase**", the products added to cart will be displayed.

**PRODUCTS ADDED TO CART**

Product	Unit Price	Quantity	Price
Camay Soap	15	1	15
Cinthol Soap	10	1	10
Dettol Soap	20	1	20
Dove	25	1	25
<b>Total Price</b>			<b>70.00 (INR)</b>

[Buy](#)
[Back](#)

**Figure 15: Products in the Cart screen**

## Payment Page for Debit card instruments

- Once the customer clicks "buy, this page is shown by Payment Gateway. Here the customers have to give their card credentials.

## Merchant Response Page:

After processing transaction, Payment Gateway sends response to the Merchant.

**Payment Information**

Your Transaction for amount **INR 70.00** is Successful.

Please note your **Transaction ID : 201518033917588** and **Payment ID: 201518033907216**.

Figure 16: Payment Information screen

- ☑ **Note:** Amount will be in OMR with 3 decimals, the above is a sample screen only

**OMAN ARAB BANK** بنك عمان العربي

Institution/ Merchant	OABTEST2	Website	https://securepayments.oabipay.com
Amount - Omani Riyals	OMR 46.000	Track ID	1318831737

**Pay by**


**Card Details**

Card Number

Expiry Date  MM /  YYYY

Cardholder Name

Card Secure Code (CVV/CVC)



3-digit Card Verification Number

\* Please do not click Back button/ refresh the page/ close the window while the transaction is processing  
This is a secure payment gateway using 128-bit SSL encryption.












Figure 17: Transaction processing Screen

## 5.6 Best Practices


- The Merchant should mandatory maintain logs for each transaction as mentioned below.
- The parameters before setting the values in the respective variable.

- Request from the merchant server to Payment Gateway
- Response that is received from the Payment Gateway in the Merchant Response URL
- The Merchant should maintain "OWASP" (Open Web Application Security Project) Top 10 recommendation in their web application. (These recommendations are available on [www.owasp.org](http://www.owasp.org))
- The Merchant should have the latest SSL security certificate in the payment request and receive webpage, if any. Always ensure that the SSL certificate is valid and has not expired. Such certificates should be as per the approved list of the Acquiring Bank. Self-signed certificates are not supported by Payment Gateway in Test and Production Environment.
- d) The Merchant should mandatory complete the UAT and ensure all results are in line with the recommended response prior to going LIVE.
- e) Any changes in the pages would need to be tested in UAT before moving to Production after proper communication to the Bank personnel and receipt of approval. If the pages have a change in logic or transaction flow particularly, the Acquiring Bank's and OAB's consent is Mandatory.
- The transaction request and Response Handling: For ease in integration, "Sample/Demo pages" provided in the integration document are essentially for representation purposes only. The actual pages must be necessarily developed and implemented by the Merchant's development team and used in both the Test and Production environment. The Sample demo pages are provided for the logical understanding and transaction flow only. An ideal logical flow for the merchant to process the customer input data is to collect the shopping details of the customer such as transaction amount, merchant track id and other parameters and stored in a secure storage location and validated immediately against the details of shopping cart module.
- Maintenance of Transaction Logs: It is essential for the transaction logs to be maintained in a secure storage location within the environment. This is crucial to trace transaction history in case of a dispute raised by a customer or even internal audit purposes. These logs should include the customer IP address as well apart from the other transaction details.

## 5.6.1 Important Notes and Information

To commence the test implementation, the merchant should ensure the information captured below is available

- **Merchant Console:** - The Payment Gateway provides a web based Merchant Console for merchants from where a merchant can download the resource file, ID, password, and transaction reports. Bank provides the Merchant Console login credentials to the merchant to start integration
- **Resource.cgn** :- This is a Confidential Key File which is required by merchants to facilitate integration with OAB Payment Gateway. This resource.cgn file is downloadable from OAB Payment Gateway merchant console. (Note – For security reasons this file can be downloaded ONLY once via the Payment Gateway Merchant Console, Hence, merchants should take utmost care while downloading the resource.cgn file. If the merchant, for any reason requires to re-download the resource.cgn file, the merchant has to contact the Acquiring Bank team for the same.)

 **Note:** The resource.cgn file is ONLY required for the integration if the merchant is using Payment Gateway Plugin for connecting to Payment Gateway and sending request via Payment Gateway Plugin supplied by Bank. The resource.cgn file is not required if merchant is posting the data on the Payment Gateway URL as mentioned in the Communication Protocol Sections

- **Test Integration and UAT Environment** :- The merchant should enable port 443 for communicating with the Payment Gateway. The request send by merchant to Payment Gateway is via port 443 and hence the same communication. For the OAB Payment Gateway Test Environment IP Addresses merchant should contact bank team. The merchant should ensure that the firewall is configured for both the IP Addresses. The domain for the test environment is <https://securepgtest.C.co.in>.
- **Response URL:** Response URL is URL where Payment Gateway sends the transaction response to merchant, the response URL is merchant page located in merchant environment and Payment Gateway sends response on this page. All transaction response validations like parameter validation, response handling validation, dual verification and transaction payment status update in backend (like database) MUST be implemented on Response URL ONLY. Response URL is merchant web page or servlet, and merchant SHOULD ensure that security measures are implemented to ensure transaction security is intact at merchant web environment. Merchant should use best security policies to ensure the data is not editable by end user or third party within the once the response is received in the merchant environment. Merchant can use firewall / proxy server rules where the Response URL and Error URL are only accessible to Payment Gateway IP addresses or domain, this ensures that the response is always received from Payment Gateway and not for third party. Once the merchant has updated the records in the database and is ready to redirect the user on the result / success or failure page, merchant should ensure limited or required

information is passed from response URL to result / success / failure pages. Post receiving response from payment gateway, successful validations and update of records in database merchants can use their own logics / methods for passing the value from response URL to result / success / failure page.

## 5.6.2 Essentials

- The merchant should ensure that any sensitive transaction information like transaction amount, merchant track id should not be passed through the customer browser to the merchant system/database as these can be intercepted and modified by the customer or even by third party to fraudulently alter the transaction data.
- Any transaction information displayed to a customer, such as amount, merchant transaction id (track id) should be passed only for display and information purposes to the customer on browser. It is imperative that the actual transactional data should be retrieved EXCLUSIVELY from the database/server and at the final stage of processing the transaction to Payment Gateway. In case where merchant is providing multiple services or products to customer in the same shopping cart or session and allows customer to make one single payment in such cases merchant should ensure the total price of all product / services are done at server side and the total amount sent to Payment Gateway is either the total done on the server side or from merchant database.
- ALL transaction data input from the customer should ONLY be accepted once from a browser at the point of input, and then inserted and stored in a way that does not allow vulnerability or access to others to modify details (e.g. database, server session, etc.)
- Merchant should ensure that parameters posted to the Payment Gateway are solely from secure source and not from the customer's browser.
- The merchant should ensure that the merchant code/page that forwards the transaction request to the Payment Gateway is secure and should not be vulnerable for alteration/modification.
- The merchant should ensure that each parameter is validated before sending request to Payment Gateway (example amount should be numeric, junk characters should not be passed in any of the input parameters).
- Characters like [ !#\$%^&\*()+[]\';,{}|\"':<>?~` ] are treated as hack characters and should not be passed in any parameters by the merchant. Merchant should also ensure that "return carriage" is not passed to Payment Gateway in any of the input parameter specially the UDF values. The "return carriage" leads to improper formatting of the reports in Payment Gateway which may lead to reconciliation issues at merchant level at later stage.
- The merchant MUST validate the Track Id and Amount received in final response from Payment Gateway against the request sent. It is very IMPORTANT that merchant uses Payment Gateway Transaction ID / TrackID / PaymentID / SeqNum for Dual Verification; this to ensure that payment transaction security remains intact. We

recommend and highly suggest merchant uses combination of Payment Id, Track Id and Amount for dual verification for exact transaction result

- The request sent from the merchant to the Payment Gateway can be a simple merchant servlet also that manages request/response handling and does not interact directly with the customer browser.
- THE SAMPLE PAGE PROVIDED WITH THIS GUIDE IS FOR DEMONSTRATIVE PURPOSES ONLY AND CANNOT BE USED FOR THE PRODUCTION SETUP AS THE MERCHANT IS REQUIRED TO BUILD IN THE REQUISITE CHECKS AND BALANCES AS MENTIONED IN THE STEPS ABOVE IN THE PAGES DEPLOYED IN PRODUCTION.
- Sensitive details of merchant like ID, Password, should be encrypted and the same be stored it in the database and retrieve each time, decrypted, and used

### 5.6.3 Frequently Asked Queries (General)

1. Not able to connect to payment gateway

The following options can help to solve the problem-FSS / QMS / Merchant Integration document / V 1.2

**Initiate a nslookup command from the merchant server**

Run the "nslookup" command from merchant server and check for successful connection establishment. The correct command is: nslookup securepgtest.OAB.co.om

The expected result of nslookup is the Payment Gateway IP Address of the Web Server

The merchant logs into the site, this will help to identify whether there is any restrictions in the network to open the page

<https://securepgtest.oab.com/pgway/gateway/Merchant>

Confirm that port 80 and 443 are open for the Payment Gateway messaging

Please collect Payment Gateway Test and Production IP Address from the Bank Team

2. Sometime Merchant gets "Denied By Risk" as the response

The merchant gets this message in response when any transaction on the merchant terminal cross the threshold transaction amount limit defined by Bank for the merchant/terminal. The merchant needs to contact bank on this error

3. The ID and Password differs from terminal to terminal, a merchant can have single or multiple terminal with the same Bank.

Merchant should ensure proper ID and password is used for respective terminal transactions.

4. In case the merchant is redirecting the response from the Payment Gateway to ports other than 80/443, the merchant should inform the Payment Gateway helpdesk team so that they can open the respective port for response handling
5. What are the browsers supported by Payment Gateway in Hosted Page environment?

Payment Gateway Bank Hosted Page is best viewed in Internet Explorer 5.5 and

above. Payment Gateway Bank Hosted page supports below versions of different browsers

- Internet Explorer 5.5 and above
- Mozilla Firefox 1.6 and above

The other browsers are supported but limited to difference in look and feel could appear

6. What needs to be done by merchant developer for integrating?

- i. Merchant developer takes all required parameters mentioned above and formulates a string from all the parameters as per sample provided below.

Sample String -

```
id=1234567890&password=merchantdemo123&action=1&langid=USA  
&currencycode=512&amt=1.000&responseURL=http://www.merchant.c  
om/responseservlet&errorURL=http://www.merchant.com/errorURL  
&trackid=TRC99673502&udf1=customer@emailid.com&udf2=address1  
&udf3=address3&udf4=city&udf5=OMAN
```

- ii. Merchant now POST the above string on the Payment Gateway URL using

HTTPS port (443 port) TCP IP  
Payment Gateway Test Environment URL -

<https://certpayments.oabipay.com/mrch/merchantLogin.htm>

# INDEX

---

## A

About this Document..... 6

## B

Bank Hosted Transaction.....51

## C

Conventions..... 7

## D

Download Plug-in.....15

Downloading Resource and KeyStore Files ...17

## E

Error Codes .....59

## G

Getting Started ..... 8

## I

Integrate Plug-in/Resource File with Merchant  
page .....19

Integration Process .....14

## M

Merchant Hosted Transaction.....54

Merchant Integration Process.....13

Merchant Prerequisites ..... See

## P

Purpose of the Document..... 7

## R

Result.....14-49

## T

Target Audience ..... 7

## U

Users..... 7