Bank of Baroda

(Payment Gateway)

Merchant Implementation Guide

Version 3.1.0

Executive Summary

This document is prepared to provide information about integrating merchant portal with Payment Gateway.

About the Document

The Payment Gateway follows industry standards and norms as prescribed by MasterCard and Visa International as well in conformity with Payment Card Industry – Data Security Standards commonly referred to as PCI – DSS.

In order that the merchants are integrated in a secure and mandated manner, this reference document is being shared. The expectation being that the merchant's system integrator or auditor is able to refer to a document while performing integration as well as post integration. It contains the technical integration details including message formats to be used in communicating to the Payment Gateway irrespective of the merchant platform being used. The document also shares the best practices and recommendations the merchant should follow during the integration with Payment Gateway.

Contents

1. Merchant Prerequisites

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

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

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, Linux, 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)

Logging: - In order to generate logs please follow below steps:-

JAVA Integration:-

1. Include a server startup parameter "PGPLUGIN_LOGPATH" with value as the system path where Logs generated are to be stored. Example :- -DPGPLUGIN_LOGPATH="D:\BOBPGPluginLogs\ipaypipetrace.log"
2. Include the jar file "log4j-1.2.14.jar" in Application Library.
3. Include below configurations in case there is an existing log4j.xml file in Merchant's application:-

```
<appender name="Trace" class="org.apache.log4j.RollingFileAppender">
```

```
                    <param name="File" value="${PGPLUGIN_LOGPATH}"/>

                <param name="MaxFileSize" value="2048KB"/>
                <param name="MaxBackupIndex" value="10"/>

                    <param name="DatePattern" value="'.'dd-MM-yyyy"/>
                    <layout class="org.apache.log4j.PatternLayout">
                            <param        name="ConversionPattern"        value="%d{MM-dd-yyyy
HH:mm:ss} [%-5p] [%C{1} - %M : %L] - %m%n" />
                    </layout>
                    <filter class="org.apache.log4j.varia.LevelRangeFilter">
  <param name="LevelMin" value="INFO" />
   <param name="LevelMax" value="INFO" />
  </filter>
          </appender>

          <appender name="Exception" class="org.apache.log4j.RollingFileAppender">

                <param name="File" value="${PGPLUGIN_LOGPATH}"/>

                <param name="MaxFileSize" value="2048KB"/>
                <param name="MaxBackupIndex" value="10"/>
                <param name="DatePattern" value="'.'dd-MM-yyyy"/>
                    <layout class="org.apache.log4j.PatternLayout">
                            <param        name="ConversionPattern"        value="%d{MM-dd-yyyy
HH:mm:ss} [%-5p] [%C{1} - %M : %L] - %m%n" />
                    </layout>
                    <filter class="org.apache.log4j.varia.LevelRangeFilter">
  <param name="LevelMin" value="ERROR" />
   <param name="LevelMax" value="ERROR" />
  </filter>
          </appender>
          <logger name="PluginLogs" additivity="false">
                    <level value="ALL"/>
                    <appender-ref ref="Trace"/>
                    <appender-ref ref="Exception"/>
          </logger>
```

.NET Integration :-

To enable the Jar level logging in the application :

1. Add the below snippet under Appsettings tag in Web.config or App.config File.

```
<appSettings>
<add key="ikvm:my.log" value="D:/"/>
</appSettings>
```

Key => Specifies the variable declared in the given Jar to enable log4j properties in the application.

Value => Specifies the Path to write the logs.

2. Merchant Integration Process

Integration Process covers the process to be followed by Merchants.

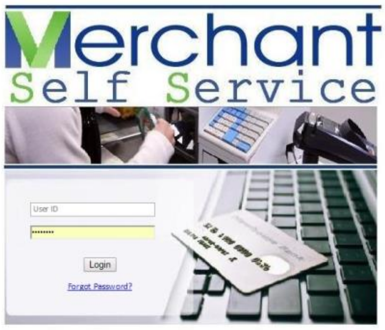
Steps to be followed by Merchants:

1) Merchant should login through payment gateway URL using their Merchant ID, User ID and Password
   URL: Will be provided separately from the respective environment (TEST / Production)
2) Merchant should download the plug-in to connect Payment Gateway using the Merchant Self Service portal login.
3) Merchant should download the Resource file & Keystore file which is generated for the specific Merchant and Terminal by the Bank
4) Merchant should copy to a folder location eg. /usr/local/payment gateway/
5) Integrate the plug-in and resource file with merchant web page.
6) Construct the Request message as expected by Payment Gateway
7) Process the response message receiving the response from Payment Gateway.

2.1 To login into Merchant Self Service application follow the below steps

Type the application URL in the browser and click Enter. Payment Gateway user login page is displayed as shown below (URL: Will be provided separately from the respective environment (TEST / Production) :

Figure 1: Merchant Self Service Login Screen



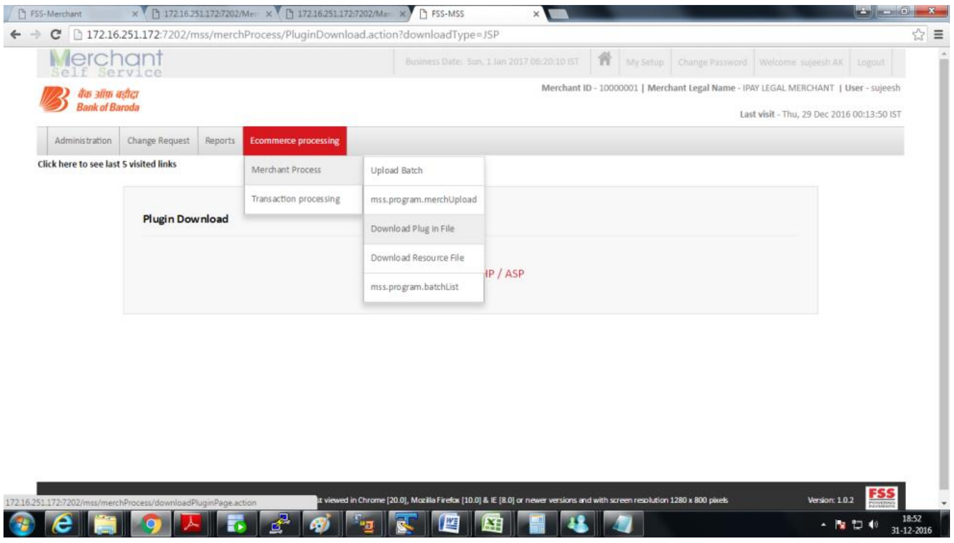
Please enter the information communicated to you through email (**Merchant ID**, **User ID**, **Password**).

Click **Submit**. The application displays the “Home Page” screen with menus.

2.2 To download Plugin to your system, follow the steps

🗁✍ Click Ecommerce Process → Merchant Process →Download Plugin File. the application displays the following screen:

Figure 3: Plugin Download Screen

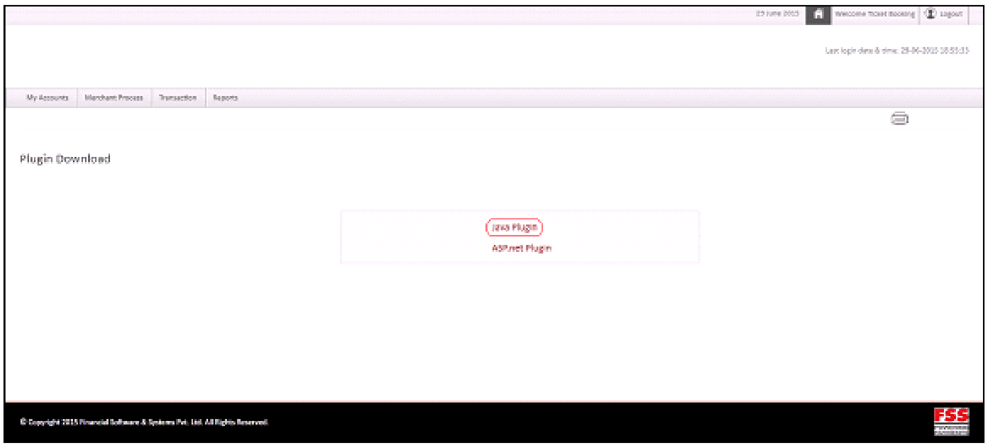

172.16.251.172:7202/mss/merchProcess/PluginDownload.action?downloadType=JSP

Business Date: Sun, 1 Jan 2017 06:20:10 IST

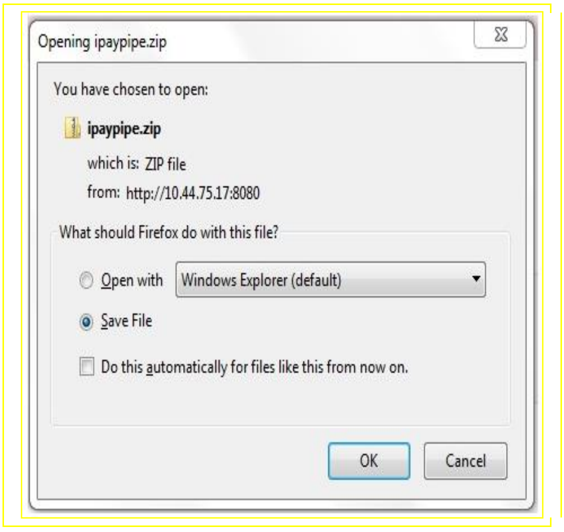
Merchant ID - 10000001 | **Merchant Legal Name** - IPAY LEGAL MERCHANT | **User** - sujeesh

Last visit - Thu, 29 Dec 2016 00:13:50 IST

a) To download Java Plugin, click **Java Plugin** link; the application displays the 'File Download' dialog box:



Plugin Download



Opening ipaypipe.zip

You have chosen to open:

ipaypipe.zip

which is: ZIP file

from: http://10.44.75.17:8080

What should Firefox do with this file?

( ) Open with Windows Explorer (default)

(•) Save File

☐ Do this automatically for files like this from now on.

Figure 4: File Download Dialog Box

Note:

Merchant have to extract the zip file and can get the ipaypipe.jar and the other three supported jar files and have to do the following steps.

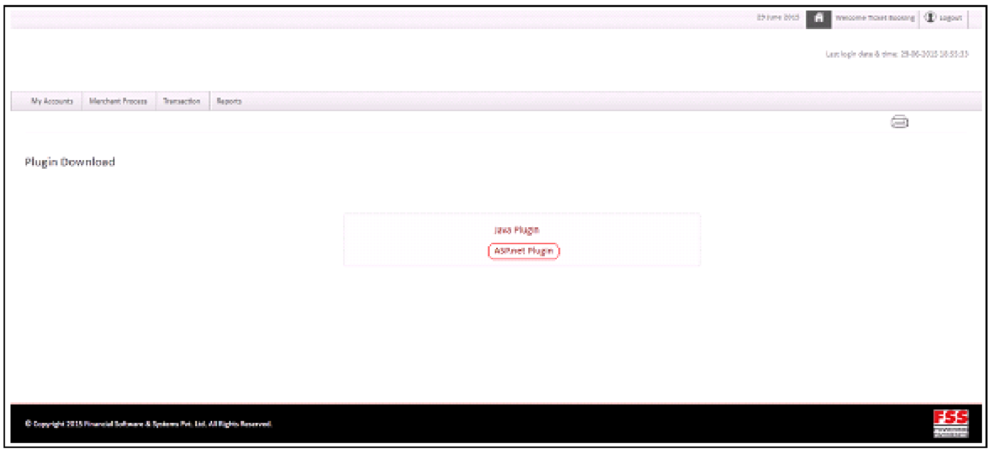
Step 1: Copy all the jar files and paste it in merchant application lib folder.

Step 2: Copy bcprov-jdk15-145.jar from extracted zip file and paste it in Java\jdk1.6.0_26\jre\lib\ext

Step 3: Add the below entry in the java. Security file (Java\jdk1.6.0_26\jre\lib\security)

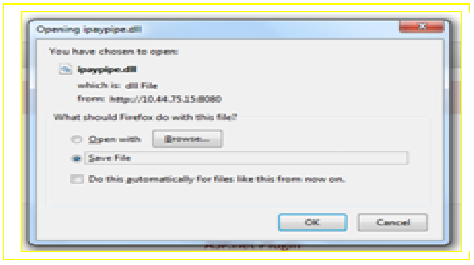
security.provider.10=org.bouncycastle.jce.provider.BouncyCastleProvider

b) To download ASP.net Plugin, click **ASP.net Plugin** link; the application displays the 'File Download' dialog box:



Plugin Download

© Copyright 2015 Financial Software & Systems Pvt. Ltd. All Rights Reserved.



Opening ipaypipe.dll

You have chosen to open:

ipaypipe.dll

which is: dll File

from: http://10.44.75.15:8080

What should Firefox do with this file?

( ) Open with [Browse...]

(●) Save File

[ ] Do this automatically for files like this from now on.

[OK] [Cancel]

Figure 5: File Download Dialog Box

Save the Plugin file in your system by clicking **Save**.

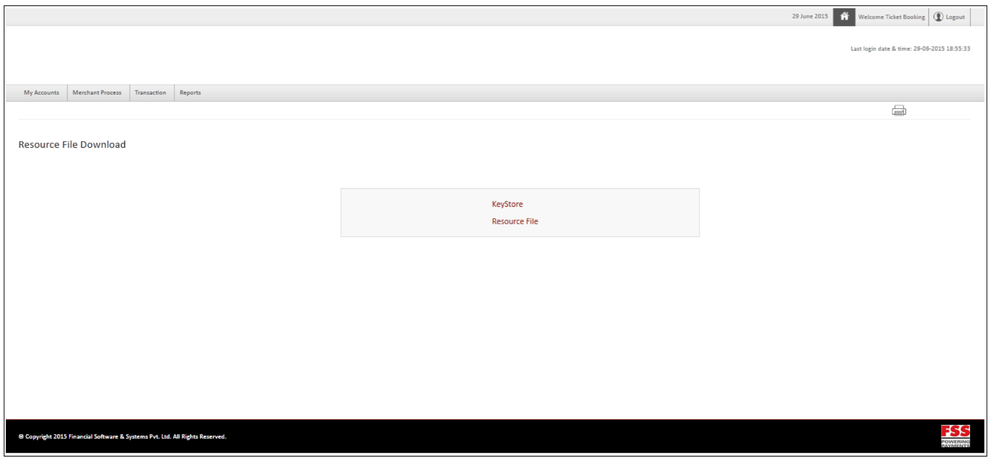
This activity completes the task of downloading plugin to your system.

2.3 Downloading Resource and Keystore Files

To download Resource Files to your system, follow the steps below:

1. Click Merchant Process Resource File Download; the application displays the following screen:

Figure 6: Resource File Download Screen



29 June 2015 | Welcome Ticket Booking | Logout

Last login date & time: 29-06-2015 18:55:33

My Accounts | Merchant Process | Transaction | Reports

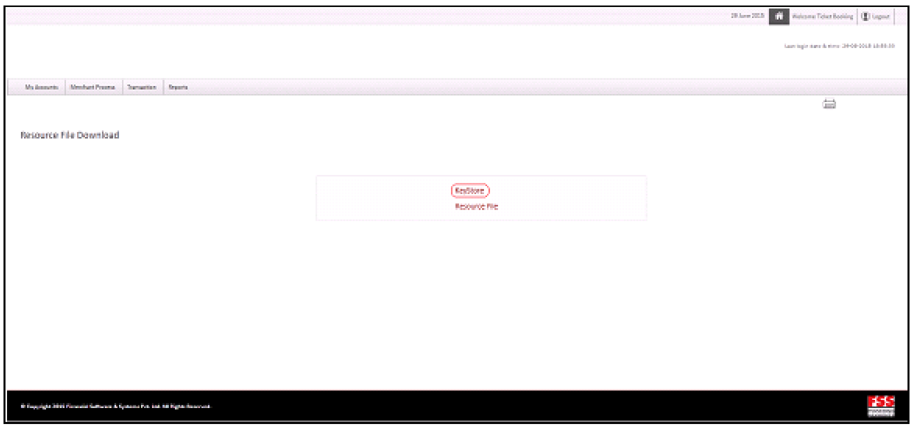
Resource File Download

KeyStore
Resource File

© Copyright 2015 Financial Software & Systems Pvt. Ltd. All Rights Reserved.

a) Downloading KeyStore File

I. Click **KeyStore** link; the application displays the 'File Download' dialog box:



29 June 2015 | Welcome Ticket Booking | Logout

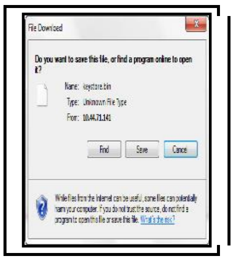
Last login date & time: 29-06-2015 18:55:33

My Accounts | Merchant Process | Transaction | Reports

Resource File Download

KeyStore
Resource File

© Copyright 2015 Financial Software & Systems Pvt. Ltd. All Rights Reserved.



File Download

Do you want to save this file, or find a program online to open it?

Name: keystore.bin
Type: Unknown File Type
From: 10.44.71.141

Find | Save | Cancel

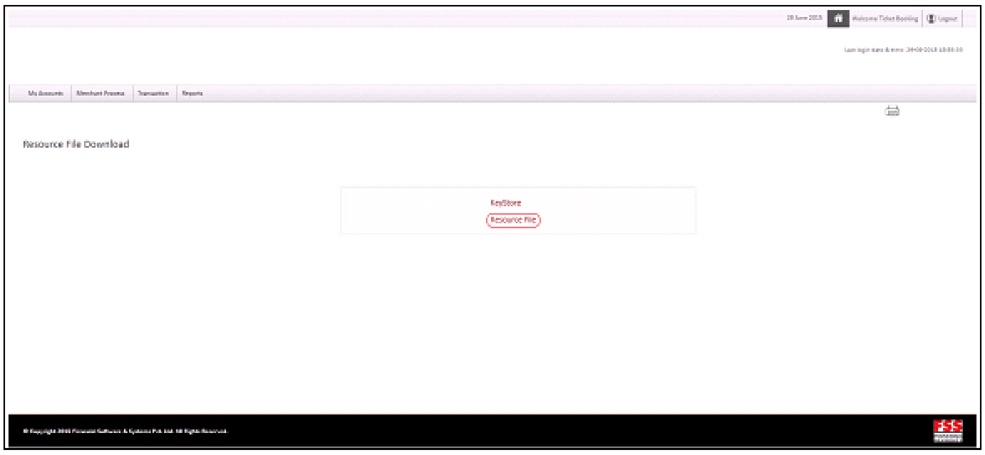
While files from the Internet can be useful, some files can potentially harm your computer. If you do not trust the source, do not find a program to open this file or save this file. What's the risk?

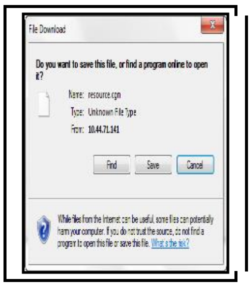
Figure 7: File Download Dialog Box

II. Save the KeyStore file in your system by clicking **Save**.

b) Downloading Resource File

Click **Resource File** link; the application displays the 'File Download' dialog box:





File Download

Do you want to save this file, or find a program online to open it?

Name: resource.cgn
Type: Unknown File Type
From: 10.44.71.141

While files from the Internet can be useful, some files can potentially harm your computer. If you do not trust the source, do not find a program to open this file or save this file. What's the risk?

Figure 8: File Download Dialog Box

Save the Resource File in your system by clicking **Save**.

This activity completes the task of downloading Resource Files to your system.

2.4 Copy Resource / Keystore / Plugin to a folder location

Move the downloaded resource & Keystore to a specific folder.

Ex. C:\\resourcepath

2.5 Integrate the plug-in and resource file with merchant web page

<u>**Note:** In case of Sun JDK environment. Project library should be added with following jar files:</u>

1. ibmpkcs.jar
2. ibmjceprovider.jar

2.5.1 Code snippet for JSP Integration

2.5.1.1 Hosted Payment Integration (Single Step Integration)

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
        iPayPipe pipe = new iPayPipe();
```

//Initialization

```
String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
String recieptURL= "http://www.demomerchant.com/result.jsp";
String errorURL= "http://www.demomerchant.com/error.jsp";
// 1 – Purchase, 4 – Auth, 17 – IMPS Transaction
String action="1";
//Terminal Alias Name
String aliasName = "aliasName";
//Transaction Currency
String currency = "currency"; (ex: "356")
String language = "language"; (ex: "USA")
//Transaction Amount
String amount = "1000.00";
//Merchant Track ID
String trackid = "109088888";
//User Defined Fields
String Udf1= "Udf1";
String Udf2= "Udf2";
String Udf3= "Udf3";
String Udf4= "Udf4";
String Udf5= "Udf5";
//Faster Checkout Related information
String custid = "custid";

// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the UAT and production.
String Udf6 = "Udf6";
String Udf7 = "Udf7";
```

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
```

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);// For Faster Checkout , it should be given as “FC”
pipe.setUdf5(Udf5);
pipe.setCustid(custid); //For Faster Checkout, this field should be given.

// For Setting UDF 6 to UDF 32
pipe.setUdf6(Udf6);
pipe.setUdf7(Udf7);

// For **Hosted Payment Integration(Single Step integration)**, the method to be called is

pipe.performPaymentInitializationHTTP();

//To redirect the web address.

response.sendRedirect(**pipe.getWebAddress**);

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A3469434850677825704053 3E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F

//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,

iPayPipe pipe = new iPayPipe();

//Initialization

String resourcePath = “c:\\resourcepath”;

String keystorePath = “c:\\ resourcepath”;

//Terminal Alias Name

String aliasName = “aliasName”;

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
```

The method to be called is,

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.

Ex:

```
// To get result,
        pipe.getResult();  // Ex: CAPTURED or SUCCESS or APPROVED or VOIDED
// To get payment ID
        pipe.getPaymentId();
//To get Transaction ID
        pipe.getTransId();
// To get Amount
        pipe.getAmt();
```

Case2: 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();
```

Case3: If request.getParameter("**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 trackid

                        request.getParameter("trackid");

                 //To get Transaction ID

                        request.getParameter("tranid");

                //To get Payment ID

                        request.getParameter("paymentid");

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

2.5.1.2 Hosted Payment Integration (Two Step Integration)

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
        iPayPipe pipe = new iPayPipe();
```

//Initialization

```
String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
String recieptURL= "http://www.demomerchant.com/result.jsp";
String errorURL= "http://www.demomerchant.com/error.jsp";
// 1 – Purchase, 4 – Auth ,17 -IMPS
String action="1";

//Terminal Alias Name
String aliasName = "aliasName";

//Transaction Currency
String currency = "currency"; (ex: "356")
String language = "language"; (ex: "USA")
//Transaction Amount
String amount = "1000.00";
//Merchant Track ID
String trackid = "109088888";

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

String Udf5= “Udf5”;


// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the UAT and production.

String Udf6 = “Udf6”;

String Udf7 = “Udf7”;


//Faster Checkout  Related information

 String custid = “custid”;
```

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
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);//For  Faster Checkout  ,it should be given as “FC”
pipe.setUdf5(Udf5);
pipe.setCustid(custid); //For Faster Checkout ,this field should be given.


// For Setting UDF 6 to UDF 32

pipe.setUdf6(Udf6);

pipe.setUdf7(Udf7);
```

// For **Hosted Payment Integration (Two Step integration)**, the method to be called is

// Step 1

pipe.performPaymentInitialization();

// Then they have to redirect the paymentpage url with payment ID which is sent by Payment Gateway.

//Step 2

 response.sendRedirect (**pipe.getPaymentPage ()**+”?PaymentID”+ **pipe.getPaymentId ()**);

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/

Here the above mentioned response parameters are sent as opened text.

Response to Merchant:

```
paymentid=201518226389866&result=SUCCESS&ref=518210000087&tranid=201518226408041&custid=&postd
ate=0701&trackid=1263556640&udf1=0&udf2=8681F6DEAB81434EFE33F9F591CF09F9&udf3=8681F6DEAB814
34E2B8E695FC76E4DA8&udf4=FC&udf5=Tidal+5&amt=10000.0
```

Merchant should follow the steps to get the response fields.

```
    //To get Result

        request.getParameter(“result”);

    //To get Error

        request.getParameter(“ErrorText”);

   //To get Transaction ID

        request.getParameter("tranid");

   //To get Payment ID

        request.getParameter("paymentid");

   //To get Transaction Amount

        request.getParameter("amt");

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

2.5.1.3 Tranportal Transaction – With Encryption

[Merchant saved card option use this integration method]

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
        iPayPipe pipe = new iPayPipe();
```

//Initialization

```
String resourcePath = “c:\\resourcepath”;
String keystorePath = “c:\\ resourcepath”;
String recieptURL= “http://www.demomerchant.com/result.jsp”;
String errorURL= “http://www.demomerchant.com/error.jsp”;
// 1 –Purchase //4-Auth
String action=”1”;

//Terminal Alias Name
String aliasName = “aliasName”;
//Merchant Track ID
String trackid = “109088888”;

//Transaction Currency
String currency = “currency”; (ex: “356”)
String language = “language”; (ex: “USA”)
//Transaction Amount
String amount = “1000.00”;
// Transaction  ID
String transId = “2015789645632”;
//Card Type
String cardType = “C”;   // C – Credit  //D - Debit
//Card Number
String pan =”4000000000000002”;
//Cvv
String cvv = ”123”;
//Expiry Month
String expmm = ”12”;
// Expiry Year
String expyy = ”2015”;
//Member Name
String name = ”test”;
```

```
//User Defined Fields
String Udf1= “Udf1”;
String Udf2= “Udf2”;
String Udf3= “Udf3”;
String Udf4= “Udf4”;
String Udf5= “Udf5”;

// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the UAT and production.
String Udf6 = “Udf6”;
String Udf7 = “Udf7”;

//Saved card Flag
String savedCardFlg= “Y”;
```

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
pipe.setAction( action );
pipe.setCurrency(currency);
pipe.setLanguage(language);
pipe.setResponseURL( receiptURL );
pipe.setErrorURL(errorURL);
pipe.setAmt(amount);
pipe.setTransId(transId);
pipe.setType(cardType);
pipe.setTrackId(trackid);
pipe.setCard(pan);
pipe.setUdf1 (Udf1);
pipe.setUdf2(Udf2);
pipe.setUdf3(Udf3);
pipe.setUdf4(Udf4);
 pipe.setUdf5(Udf5);
pipe.setCvv2(cvv);

pipe.setExpMonth(expmm);

pipe.setExpYear(expyy);

pipe.setMember(name);

pipe.setSavedCard(savedCardFlg); // For Merchant saved card option

                                        this field should be given.( value-“Y”)
```

```
// For Setting UDF 6 to UDF 32
pipe.setUdf6(Udf6);
pipe.setUdf7(Udf7);

//The method to be called is
```

pipe.performVbVTransaction();

// Then the Merchant have to redirect the Payment Gateway url.

response.sendRedirect (**pipe.getWebAddress()**);

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A3469434850677825704053 3E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F

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,

iPayPipe pipe = new iPayPipe();

//Initialization

String resourcePath = “c:\\resourcepath”;

String keystorePath = “c:\\ resourcepath”;

//Terminal Alias Name

String aliasName = “aliasName”;

//Set Values

pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);

The method to be called is,

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.

Ex:

```
// To get result,
        pipe.getResult();  // Ex: CAPTURED or SUCCESS or VOIDED or APPROVED
// To get payment ID
        pipe.getPaymentId();
//To get Transaction ID
        pipe.getTransId();
// To get 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”);
// To get Transaction ID
        pipe.getTransId();
// To get Payment ID
        pipe.getPaymentId();
```

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 **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**/
```

2.5.1.4 Tranportal Refund Transaction (Credit) – With Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

iPayPipe pipe = new iPayPipe();

//Initialization

String resourcePath = "c:\\resourcepath";

String keystorePath = "c:\\ resourcepath";

String recieptURL= "http://www.demomerchant.com/result.jsp";

String errorURL= "http://www.demomerchant.com/error.jsp";

// 2 –Credit

String action="2";

//Terminal Alias Name

String aliasName = "aliasName";

//Merchant Track ID

String trackid = "109088888";

//Transaction Currency

String currency = "currency"; (ex: "356")

String language = "language"; (ex: "USA")

//Transaction Amount

String amount = "1000.00";

// Transaction ID

String transId = "2015789645632";

//Card Type

String cardType = "C";   // C - Credit

//User Defined Fields

String Udf1= "Udf1";

String Udf2= "Udf2";

String Udf3= "Udf3";

String Udf4= "Udf4";

String Udf5= "PaymentID" (If merchant is sending payment id in transId field) or
"TrackID" (If merchant is sending merchant track id in transId field) or
"TRANID " (If merchant is sending transaction id in transId field).

```
// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the UAT and production.

String Udf6 = “Udf6”;

String Udf7 = “Udf7”;

//Set Values

pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
pipe.setAction( action );
pipe.setCurrency(currency);
pipe.setLanguage(language);
pipe.setResponseURL( receiptURL );
pipe.setErrorURL(errorURL);
pipe.setAmt(amount);
pipe.setTransId(transId);
pipe.setType(cardType);

pipe.setTrackId(trackid);


pipe.setUdf1 (Udf1);

pipe.setUdf2(Udf2);

pipe.setUdf3(Udf3);

pipe.setUdf4(Udf4);

 pipe.setUdf5(Udf5);


// For Setting UDF 6 to UDF 32
pipe.setUdf6(Udf6);
pipe.setUdf7(Udf7);

//The method to be called is

pipe.performTransactionHTTP();

// Then the Merchant have to redirect the Payment Gateway url.

response.sendRedirect (pipe.getWebAddress());

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/
```

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA9375313637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589B8F8566FC9F

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,

```
iPayPipe pipe = new iPayPipe();
```

//Initialization

```
String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
//Terminal Alias Name
String aliasName = "aliasName";
```

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
```

The method to be called is,

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.

Ex:

```
// To get result,
        pipe.getResult();  // Ex: CAPTURED or SUCCESS or VOIDED
// To get payment ID
        pipe.getPaymentId();
//To get Transaction ID
        pipe.getTransId();
// To get 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");
// To get Transaction ID
        pipe.getTransId();
// To get Payment ID
```

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

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 **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**/
```

2.5.1.5 Tranportal Refund Transaction (Credit) – Without Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
        iPayPipe pipe = new iPayPipe();
```

//Initialization

```
String resourcePath = “c:\\resourcepath”;
String keystorePath = “c:\\ resourcepath”;
String recieptURL= “http://www.demomerchant.com/result.jsp”;
String errorURL= “http://www.demomerchant.com/error.jsp”;
// 2 –Credit
String action=”2”;

//Terminal Alias Name
String aliasName = “aliasName”;
//Merchant Track ID
String trackid = “109088888”;

//Transaction Currency
String currency = “currency”; (ex: “356”)
String language = “language”; (ex: “USA”)
//Transaction Amount
String amount = “1000.00”;
// Transaction  ID
String transId = “2015789645632”;
//Card Type
String cardType = “C”;   // C - Credit

//User Defined Fields
String Udf1= “Udf1”;
String Udf2= “Udf2”;
String Udf3= “Udf3”;
String Udf4= “Udf4”;
String Udf5= “PaymentID” (If merchant is sending payment id in transId field )  or
           “TrackID” (If merchant is sending merchant track  id in transId field) or
           “TRANID “ (If merchant is sending transaction  id in transId field).
```

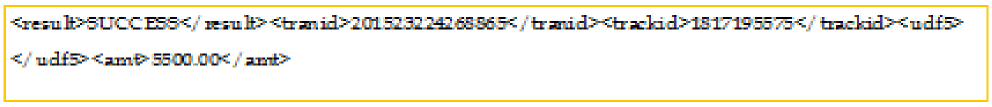
```
// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the
UAT and production.
String Udf6 = “Udf6”;
String Udf7 = “Udf7”;
```

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
pipe.setAction( action );
pipe.setCurrency(currency);
pipe.setLanguage(language);
pipe.setResponseURL( receiptURL );
pipe.setErrorURL(errorURL);
pipe.setAmt(amount);
pipe.setTransId(transId);
pipe.setType(cardType);
pipe.setTrackId(trackid);


pipe.setUdf1 (Udf1);
pipe.setUdf2(Udf2);
pipe.setUdf3(Udf3);
pipe.setUdf4(Udf4);
 pipe.setUdf5(Udf5);

// For Setting UDF 6 to UDF 32
pipe.setUdf6(Udf6);
pipe.setUdf7(Udf7);

//The method to be called is
```

pipe.performTransaction();

```
/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant in TCPIP **/
```



```
<result>SUCCESS</result><tranid>201523224268865</tranid><trackid>1817195575</trackid><udf5>
</udf5><amt>5500.00</amt>
```

Merchant should follow the steps to get the response fields.

```
        // To get result,

                pipe.getResult();  // Ex: CAPTURED or SUCCESS or VOIDED

        // To get payment ID

                pipe.getPaymentId();

        //To get Transaction ID

                pipe.getTransId();
        // To get Amount
                pipe.getAmt();
        //To get  Error(If error occured)
                pipe.getError();
/** End of Response Processing**/
```

2.5.1.6 Tranportal Inquiry Transaction – With Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
        iPayPipe pipe = new iPayPipe();
```

//Initialization

```
String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
String recieptURL= "http://www.demomerchant.com/result.jsp";
String errorURL= "http://www.demomerchant.com/error.jsp";
// 8 –Inquiry
String action="8";

//Terminal Alias Name
String aliasName = "aliasName";
//Merchant Track ID
String trackid = "109088888";

//Transaction Currency
String currency = "currency"; (ex: "356")
String language = "language"; (ex: "USA")
//Transaction Amount
String amount = "1000.00";
// Transaction  ID
String transId = "2015789645632";
//Card Type
String cardType = "C";   // C - Credit

//User Defined Fields
String Udf1= "Udf1";
String Udf2= "Udf2";
```

String Udf3= "Udf3";

String Udf4= "Udf4";

String Udf5= "PaymentID" (If merchant is sending payment id in transId field) or

"TrackID" (If merchant is sending merchant track id in transId field) or

"TRANID " (If merchant is sending transaction id in transId field).

// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the UAT and production.

String Udf6 = "Udf6";

String Udf7 = "Udf7";

//Set Values

pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
pipe.setAction(action);
pipe.setCurrency(currency);
pipe.setLanguage(language);
pipe.setResponseURL(receiptURL);
pipe.setErrorURL(errorURL);
pipe.setAmt(amount);
pipe.setTransId(transId);
pipe.setType(cardType);

pipe.setTrackId(trackid);
pipe.setUdf1 (Udf1);

pipe.setUdf2(Udf2);

pipe.setUdf3(Udf3);

pipe.setUdf4(Udf4);

pipe.setUdf5(Udf5);

// For Setting UDF 6 to UDF 32
pipe.setUdf6(Udf6);
pipe.setUdf7(Udf7);

//The method to be called is

pipe.performTransactionHTTP();

// Then the Merchant have to redirect the Payment Gateway url.

response.sendRedirect (**pipe.getWebAddress()**);

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A3469434850677825704053 3E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F

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,

```
iPayPipe pipe = new iPayPipe();
```

//Initialization

```
String resourcePath = “c:\\resourcepath”;
String keystorePath = “c:\\ resourcepath”;
//Terminal Alias Name
String aliasName = “aliasName”;
```

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
```

The method to be called is,

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.

Ex:

```
    // To get result,
            pipe.getResult();  // Ex: CAPTURED or SUCCESS
    // To get payment ID
            pipe.getPaymentId();
    //To get Transaction ID
            pipe.getTransId();
    // To get 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”);
// To get Transaction ID
      pipe.getTransId();
// To get Payment ID
      pipe.getPaymentId();
```

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 **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**/

2.5.1.7 Tranportal Inquiry Transaction – Without Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

iPayPipe pipe = new iPayPipe();

//Initialization

```
String resourcePath = “c:\\resourcepath”;
String keystorePath = “c:\\ resourcepath”;
String recieptURL= “http://www.demomerchant.com/result.jsp”;
String errorURL= “http://www.demomerchant.com/error.jsp”;
// 8 –Inquiry
String action=”8”;

//Terminal Alias Name
String aliasName = “aliasName”;
//Merchant Track ID
String trackid = “109088888”;

//Transaction Currency
String currency = “currency”; (ex: “356”)
String language = “language”; (ex: “USA”)
//Transaction Amount
String amount = “1000.00”;
// Transaction  ID
String transId = “2015789645632”;
//Card Type
String cardType = “C”;   // C - Credit


//User Defined Fields
String Udf1= “Udf1”;
String Udf2= “Udf2”;
String Udf3= “Udf3”;
String Udf4= “Udf4”;
String Udf5= “PaymentID” (If merchant is sending payment id in transId field )  or
          “TrackID” (If merchant is sending merchant track  id in transId field) or
          “TRANID “ (If merchant is sending transaction  id in transId field).
```

// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the UAT and production.

String Udf6 = “Udf6”;

String Udf7 = “Udf7”;

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
pipe.setAction( action );
pipe.setCurrency(currency);
pipe.setLanguage(language);
pipe.setResponseURL( receiptURL );
pipe.setErrorURL(errorURL);
pipe.setAmt(amount);
pipe.setTransId(transId);
pipe.setType(cardType);
pipe.setTrackId(trackid);
pipe.setUdf1 (Udf1);
pipe.setUdf2(Udf2);
pipe.setUdf3(Udf3);
pipe.setUdf4(Udf4);
 pipe.setUdf5(Udf5);
```

// For Setting UDF 6 to UDF 32

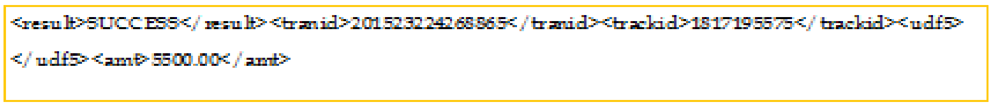
```
pipe.setUdf6(Udf6);
pipe.setUdf7(Udf7);
```

//The method to be called is

pipe.performTransaction();

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant in TCPIP **/



```
<result>SUCCESS</result><tranid>201523224268865</tranid><trackid>1817195575</trackid><udf5>
</udf5><amt>5500.00</amt>
```

Merchant should follow the steps to get the response fields.

```
        // To get result,
                pipe.getResult();  // Ex: CAPTURED or SUCCESS
        // To get payment ID
                pipe.getPaymentId();
        //To get Transaction ID
```

```
                    pipe.getTransId();
          // To get Amount
                    pipe.getAmt();
          //To get  Error(If error occured)
                    pipe.getError();

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

2.5.2 Code snippet for ASP.NET Integration

2.5.2.1 Hosted Payment Integration (Single Step Integration)

/ Request Processing**/**

Merchant can connect iPay Plugin using below steps:

i) Add the java package into the Dotnet application by adding the ,

using com.fss.plugin.bob; namespace into the corresponding pages.

Note: It should be added after the Plugin DLL refered into the application. (Please refer to the DLL integration steps for development Registering ASP. NET Plug-in the Merchant Server)

ii) create a instance for an ipaypipe class like below:

iPayPipe pipe = new iPayPipe();

For Ex: Refer **"HostedPaymentBuy.aspx.cs"** in the sample application.

//Initialization

```
String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
String recieptURL= "http://www.demomerchant.com/result.aspx";
String errorURL= "http://www.demomerchant.com/error.aspx";
// 1 – Purchase, 4 - Auth
String action="1";

//Terminal Alias Name
String aliasName = "aliasName"; //To be used by referring in MSS portal application, credentials provided by Bank.

//Transaction Currency
String currency = "currency"; (ex: "356")
String language = "language"; (ex: "USA")
//Transaction Amount
String amount = "1000.00";
//Merchant Track ID – to be unique
String trackid = "109088888";
```

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

// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the
UAT and production.
String Udf6 = "Udf6";
String Udf7 = "Udf7";

//Faster Checkout  Related information
 String custid = "custid";
```

//Set Values

```
pipe.setResourcePath(resourcePath);
pipe.setKeystorePath(resourcePath);
pipe.setAlias(aliasName);
pipe.setAction( action );
pipe.setCurrency(currency);
pipe.setLanguage(language);
pipe.setResponseURL( receiptURL );
pipe.setErrorURL(errorURL);
pipe.setAmt(amount);
pipe.setTransId(transId);
pipe.setType(cardType);
pipe.setTrackId(trackid);
pipe.setUdf1 (Udf1);
pipe.setUdf2(Udf2);
pipe.setUdf3(Udf3);
pipe.setUdf4(Udf4);
 pipe.setUdf5(Udf5);
```

// For **Hosted Payment Integration (Single Step integration)**, the method to be called is

pipe. PerformPaymentInitializationHTTP ();

//To redirect the web address.

Response.Redirect (**pipe.getWebAddress()**);

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F

//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,

iPayPipe pipe = new iPayPipe();

//Initialization

String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
//Terminal Alias Name
String aliasName = "aliasName"; //To be used by referring in MSS portal application, credentials provided by Bank.
String trandata = Request.Form["trandata"].ToString(); //which is getting from payment gateway response.
If trandata is null then
trandata= Request.QueryString["trandata"];

//Set Values

pipe.setResourcePath(resourcePath);

pipe.setKeystorePath(resourcePath);

pipe.setAlias(aliasName);

pipe.setTranData= trandata;

The method to be called is,
pipe. ParseEncryptedRequest ();

Note: Pass the trandata value into the above method as parameter

For Ex: Refer **"HostedPaymentResult.aspx.cs"** in the sample application.

Ex:

```
            // To get result,
                        pipe.getResult(); // Ex: CAPTURED or SUCCESS or VOIDED or APPROVED
            // To get payment ID
pipe.getPaymentID;
            //To get Transaction ID
 pipe.getTranID;
            // To get Amount
 pipe.getAmt;
            //To get ErrorText
 pipe.getErrorText;
            /** End of Response Processing**/
```

Note: Access privilege for the folder will be required where the resourcePath and keystorePath is provided.

2.5.2.2 Hosted Payment Integration (Two Step Integration)

/ Request Processing**/**

Merchant can connect iPay Plugin using below steps:

i) Add the java package into the Dotnet application by adding the ,

using com.fss.plugin.bob; namespace into the corresponding pages.

Note: It should be added after the Plugin DLL refered into the

application.

ii) create a instance for an ipaypipe class like below:

iPayPipe pipe = new iPayPipe();

For Ex: Refer **"HostedPaymentBuyTCP.aspx.cs"** in the sample application.

//Initialization

```
String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
String recieptURL= "http://www.demomerchant.com/result.aspx";
String errorURL= "http://www.demomerchant.com/error.aspx";
// 1 – Purchase, 4 - Auth
String action="1";

//Terminal Alias Name
String aliasName = "aliasName"; //To be used by referring in MSS portal application, credentials provided by Bank.

//Transaction Currency
String currency = "currency"; (ex: "356")
String language = "language"; (ex: "USA")
//Transaction Amount
String amount = "1000.00";
//Merchant Track ID
String trackid = "109088888";
//User Defined Fields
String Udf1= "Udf1";
String Udf2= "Udf2";
String Udf3= "Udf3";
String Udf4= "Udf4";
String Udf5= "Udf5";
```

```
// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the UAT and production.
String Udf6 = “Udf6”;
String Udf7 = “Udf7”;

// Faster Checkout  Related information
 String custid = “custid”;
//Set Values

        pipe.setResourcePath(resourcePath);
        pipe.setKeystorePath(resourcePath);
        pipe.setAlias(aliasName);
        pipe.setAction( action );
        pipe.setCurrency(currency);
        pipe.setLanguage(language);
        pipe.setResponseURL( receiptURL );
        pipe.setErrorURL(errorURL);
        pipe.setAmt(amount);
        pipe.setTransId(transId);
        pipe.setType(cardType);
        pipe.setTrackId(trackid);
        pipe.setUdf1 (Udf1);
        pipe.setUdf2(Udf2);
        pipe.setUdf3(Udf3);
        pipe.setUdf4(Udf4);
        pipe.setUdf5(Udf5);

        pipe.setcustid = custid ; //For Faster Checkout, this field should be given.

// For Setting UDF 6 to UDF 32
       pipe.setUdf6 = Udf6;
       pipe.setUdf7 = Udf7;


 // For Hosted Payment Integration (Two Step integration), the method to be called is

  // Step 1

  pipe.PerformPaymentInitialization();

// Then they have to redirect the paymentpage url with payment  ID which is sent by Payment Gateway.
```

To get payment page and payment ID ,

use,

pipe.getpaymentid and **pipe.getPaymentpage.**

//Step 2

Response.Redirect(**pipe.getPaymentPage** + "?PaymentID=" + **pipe.getPaymentID**);

For Ex: Refer **"HostedPaymentBuyTCP.aspx.cs"** in the sample application.

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/

Here the above mentioned response parameters are sent as opened text.
Response to Merchant:

```
paymentid=201518226389866&result=SUCCESS&ref=518210000087&tranid=201518226408041&custid=&postd
ate=0701&trackid=1263556640&udf1=0&udf2=8681F6DEAB81434EFE33F9F591CF09F9&udf3=8681F6DEAB814
34E2B8E695FC76E4DA8&udf4=FC&udf5=Tidal+5&amt=10000.0
```

Merchant should follow the steps to get the response fields by either in Form or in Querystring from Payment Gateway,

For Ex: Refer **"HostedPaymentResultTCP.aspx.cs"** in the sample application.

```
//To get Result

Request.Form["result"].ToString()  or Request.QueryString["result"]

//To get Error

Request.Form["ErrorText"].ToString();   or Request.QueryString["ErrorText"]

//To get Transaction ID

Request.Form["tranid"].ToString();   or Request.QueryString["tranid"]

//To get Payment ID

Request.Form["paymentid"].ToString(); or Request.QueryString["paymentid"]

//To get Transaction Amount

Request.Form["amt"].ToString();  or Request.QueryString["amt"]
```

/** End of Response Processing**/

2.5.2.3 Tranportal Transaction – With Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below steps:

i) Add the java package into the Dotnet application by adding the ,

using com.fss.plugin.bob; namespace into the corresponding pages.

Note: It should be added after the Plugin DLL refered into the application. (Please refer to the DLL integration steps for development Registering ASP. NET Plug-in the Merchant Server)

ii) create a instance for an ipaypipe class like below:

iPayPipe pipe = new iPayPipe();

For Ex: Refer **"TranportalBuy.aspx.cs"** in the sample application.

//Initialization

```
String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
String recieptURL= "http://www.demomerchant.com/result.aspx";
String errorURL= "http://www.demomerchant.com/error.aspx";
// 1 – Purchase, 4 - Auth
String action="1";

//Terminal Alias Name
String aliasName = "aliasName"; //To be used by referring in MSS portal application, credentials provided by Bank.
String pan ="4000000000000002"; //Card Number
String cvv = "123"; //Cvv
String expmm = "12"; //Expiry Month
String expyy = "2015"; // Expiry Year
//Card Type
String cardType = "C";   // C – Credit  //D – Debit
String name = "test"; //Card holder Name
//Transaction Currency
String currency = "currency"; (ex: "356")
String language = "language"; (ex: "USA")
//Transaction Amount
String amount = "1000.00";
//Merchant Track ID
String trackid = "109088888";
```

```
//User Defined Fields
String Udf1= “Udf1”;
String Udf2= “Udf2”;
String Udf3= “Udf3”;
String Udf4= “Udf4”;
String Udf5= “Udf5”;

// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the
UAT and production.
String Udf6 = “Udf6”;
String Udf7 = “Udf7”;

//Faster Checkout  Related information
 String custid = “custid”;
```

//Set Values

```
pipe.setResourcePath(resourcePath);

pipe.setKeystorePath(resourcePath);

pipe.setAlias(aliasName);

pipe.setAction( action );
pipe.setCard(pan);

pipe.setCurrency(currency);

pipe.setLanguage(language);

pipe.setResponseURL( receiptURL );

pipe.setErrorURL(errorURL);

pipe.setAmt(amount);

pipe.setTransId(transId);

pipe.setType(cardType);
pipe.setCvv2(cvv);
pipe.setExpMonth(expmm);
pipe.setExpYear(expyy);
pipe.setMember(name);
pipe.setTrackId(trackid);

pipe.setUdf1 (Udf1);
pipe.setUdf2(Udf2);
pipe.setUdf3(Udf3);
pipe.setUdf4(Udf4);
 pipe.setUdf5(Udf5);
```

// For **Tranportal Integration (with Encryption)**, the method to be called is

pipe. performVbVTransaction();

//To redirect the web address.

Response.Redirect(**pipe.getWebAddress()**);

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F

//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,

iPayPipe pipe = new iPayPipe();

//Initialization

String resourcePath = “c:\\resourcepath”;
String keystorePath = “c:\\ resourcepath”;
//Terminal Alias Name
String aliasName = “aliasName”;
String trandata = Request.Form["trandata"].ToString(); //which is getting from payment gateway response.
If trandata is null then
trandata= Request.QueryString["trandata"];

//Set Values

pipe.setResourcePath(resourcePath);

pipe.setKeystorePath(resourcePath);

pipe.setAlias(aliasName);

pipe.setTranData= trandata;

The method to be called is,
pipe. parseEncryptedResult (trandata);

Note: Pass the trandata value into the above method as parameter

For Ex: Refer **“TranportalResult.aspx.cs“** in the sample application.

Ex:

```
            // To get result,

                        pipe.getResult(); // Ex: CAPTURED or SUCCESS or VOIDED or APPROVED

            // To get payment ID

pipe.getPaymentID;

            //To get Transaction ID

 pipe.getTranID;

            // To get Amount

 pipe.getAmt;

            //To get ErrorText

 pipe.getErrorText;

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

Note: Access privilege for the folder will be required where the resourcePath and keystorePath is provided.

2.5.2.4 Tranportal Refund Transaction (Credit) – With Encryption

Merchant can connect iPay Plugin using below steps:

i) Add the java package into the Dotnet application by adding the,

using com.fss.plugin.bob; namespace into the corresponding pages.

Note: It should be added after the Plugin DLL referred into the

application

ii) create a instance for an ipaypipe class like below:

iPayPipe pipe = new iPayPipe();

For Ex: Refer **"TranportalBuy.aspx.cs"** in the sample application.

//Initialization

```
String resourcePath = "c:\\resourcepath";
String keystorePath = "c:\\ resourcepath";
String recieptURL= "http://www.demomerchant.com/result.aspx";
String errorURL= "http://www.demomerchant.com/error.aspx";
// 2 –Credit

String action="2";

//Terminal Alias Name
String aliasName = "aliasName";

//Transaction Currency
String currency = "currency"; (ex: "356")
String language = "language"; (ex: "USA")
//Transaction Amount
String amount = "1000.00";
//Merchant Track ID
String trackid = "109088888";

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

```
// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided properly during the
UAT and production.
String Udf6 = “Udf6”;
String Udf7 = “Udf7”;

// Transaction  ID
String transId = “2015789645632”;
//Card Type
String cardType = “C”;   // C - Credit
//Card Number
String cardNumber  =”4000000000000002”;
//Cvv
String cvv = ”123”;
//Expiry Month
String expmm = ”12”;
// Expiry Year
String expyy = ”2015”;
//Member Name
String name = ”test”;
```

//Set Values

```
        pipe.setResourcePath(resourcePath);
        pipe.setKeystorePath(resourcePath);
        pipe.setAlias(aliasName);
        pipe.setAction( action );
        pipe.setCurrency(currency);
        pipe.setLanguage(language);
        pipe.setResponseURL( receiptURL );
        pipe.setErrorURL(errorURL);
        pipe.setAmt(amount);
        pipe.setTransId(transId);
        pipe.setType(cardType);
        pipe.setTrackId(trackid);
        pipe.setUdf1 (Udf1);
        pipe.setUdf2(Udf2);
        pipe.setUdf3(Udf3);
        pipe.setUdf4(Udf4);
        pipe.setUdf5(Udf5);
```

// For Setting UDF 6 to UDF 32

pipe.setUdf6 = Udf6;

pipe.setUdf7 = Udf7;

// The method to be called is

pipe.PerformTransactionHTTP();

Then the merchant have to redirect the url.

Response.Redirect(**pipe.getWebAddress**, false);

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C 78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753 13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69 FC577300F9AAF38E044F0A3469434850677825704O533E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589 B8F8566FC9F

//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,

iPayPipe pipe = new iPayPipe();

//Initialization

String resourcePath = “c:\\resourcepath”;
String keystorePath = “c:\\ resourcepath”;
//Terminal Alias Name
String aliasName = “aliasName”;
String trandata = Request.Form["trandata "].ToString(); //which is getting from payment gateway response.
If trandata is null then
trandata= Request.QueryString["trandata"];

//Set Values

pipe.setAlias = aliasName;
pipe.setResourcePath = resourcePath;
pipe.setKeystorePath = resourcePath;
pipe.setTranData = trandata;

The method to be called is,

//pipe. ParseEncryptedRequest ();

pipe.parseEncryptedResult();

For Ex: Refer **"TranportalResult.aspx.cs"** in the sample application.

Ex:

```
// To get result

    pipe.getResult; // Ex: CAPTURED or SUCCESS or VOIDED

//To get Transaction ID

    pipe.getTranID;

// To get Amount

    pipe.getAmt;

//To get ErrorText

    pipe.getErrorText;
```

/** End of Response Processing**/

2.5.2.5 Tranportal Refund Transaction (Credit) – Without Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below steps:

i) Add the java package into the Dotnet application by adding the ,

using com.fss.plugin.bob; namespace into the corresponding pages.

Note: It should be added after the Plugin DLL refered into the application.

ii) create a instance for an ipaypipe class like below:

iPayPipe pipe = new iPayPipe();

For Ex: Refer **"TranportalBuyTCP.aspx.cs"** in the sample application.

//Initialization

String resourcePath = "c:\\resourcepath";

String keystorePath = "c:\\ resourcepath";

String recieptURL= "http://www.demomerchant.com/result.aspx";

String errorURL= "http://www.demomerchant.com/error.aspx";

// 2 –Credit

String action="2";

//Terminal Alias Name

String aliasName = "aliasName";

//Transaction Currency

String currency = "currency"; (ex: "356")

String language = "language"; (ex: "USA")

//Transaction Amount

String amount = "1000.00";

//Merchant Track ID

String trackid = "109088888";

//User Defined Fields

String Udf1= "Udf1";

String Udf2= "Udf2";

String Udf3= "Udf3";

String Udf4= "Udf4";

String Udf5= "Udf5";

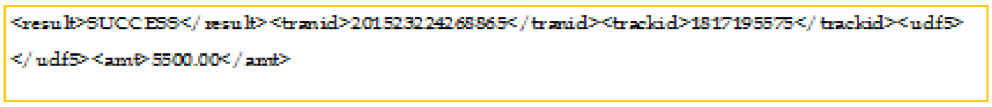
```
// For Setting UDF 6 to UDF 32, values provided are sample, actual value should be provided propery during the UAT and production.
String Udf6 = “Udf6”;
String Udf7 = “Udf7”;


// Transaction  ID
String transId = “2015789645632”;
//Card Type
String cardType = “C”;   // C - Credit
//Card Number
String cardNumber  =”4000000000000002”;
//Cvv
String cvv = ”123”;
//Expiry Month
String expmm = ”12”;
// Expiry Year
String expyy = ”2015”;
//Member Name
String name = ”test”;
```

//Set Values

```
        pipe.setResourcePath(resourcePath);
        pipe.setKeystorePath(resourcePath);
        pipe.setAlias(aliasName);
        pipe.setAction( action );
        pipe.setCurrency(currency);
        pipe.setLanguage(language);
        pipe.setResponseURL( receiptURL );
        pipe.setErrorURL(errorURL);
        pipe.setAmt(amount);
        pipe.setTransId(transId);
        pipe.setType(cardType);
        pipe.setTrackId(trackid);
        pipe.setUdf1 (Udf1);
        pipe.setUdf2(Udf2);
        pipe.setUdf3(Udf3);
        pipe.setUdf4(Udf4);
        pipe.setUdf5.(Udf5);
```

```
// For Setting UDF 6 to UDF 32
pipe.setUdf6 =Udf6;
pipe.setUdf7 = Udf7;

    // The method to be called is

    pipe.PerformTransaction();

/** End of Request Processing**/

/** Response received from Payment Gateway to Merchant **/
```

Response to Merchant:



```
<result>SUCCESS</result><tranid>201523224268865</tranid><trackid>1817195575</trackid><udf5>
</udf5><amt>5500.00</amt>
```

Merchant should follow the steps to get the response fields.

```
                              //To get Result

                                             Pipe.getResult;

                                //To get Error

Pipe.getError;

  //To get Transaction ID

Pipe.getTranID;

  //To get Transaction Amount

Pipe.getAmt;
```

2.5.2.6 Registering ASP. NET Plug-in the Merchant Server

1. Kindly Place the provided DLL's into some Physical Path of the system.

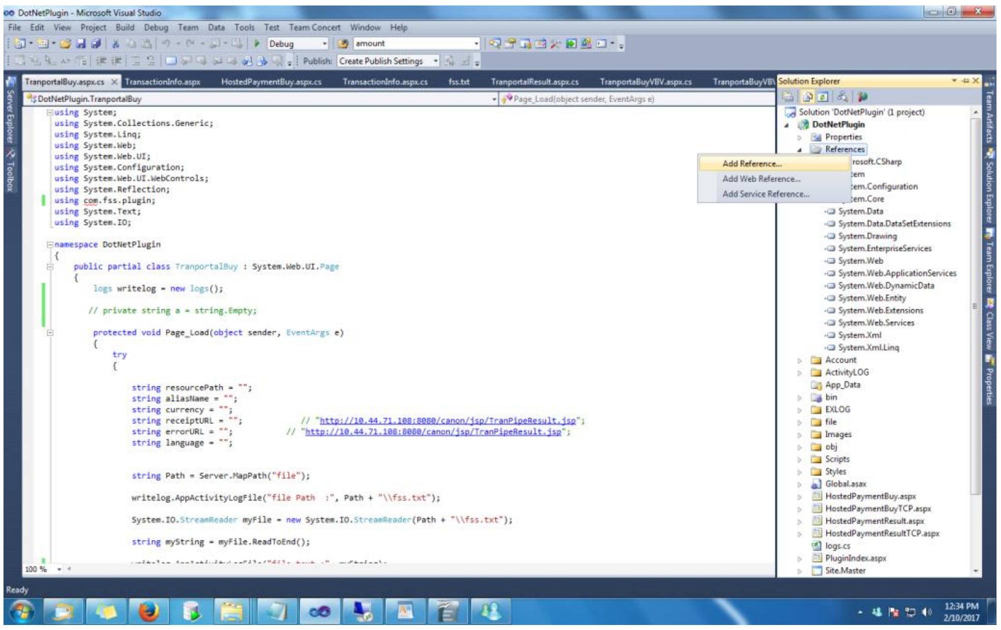
2. And Refer the DLL's into the application by following the below steps

   DLL's to be referred:

   - iPayPipe.dll
   - IKVM.OpenJDK.Core.dll
   - IKVM.Runtime.dll

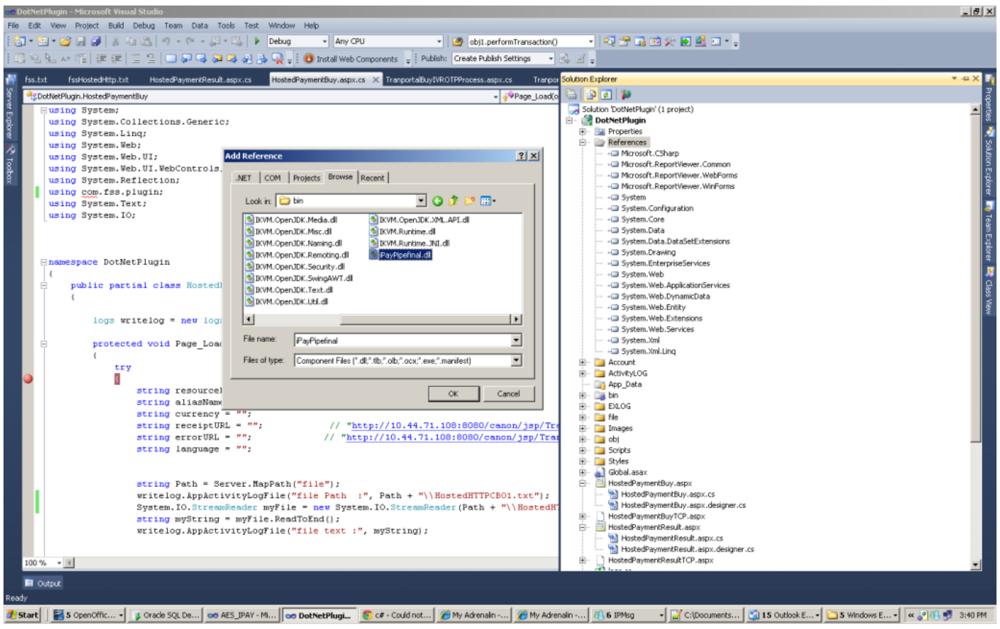
3. How to add reference:

Right Click the **Reference tab** of the Solution Explorer in the Visual studio IDE and click **Add Reference**



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Configuration;
using System.Web.UI.WebControls;
using System.Reflection;
using com.fss.plugin;
using System.Text;
using System.IO;

namespace DotNetPlugin
{
    public partial class TranportalBuy : System.Web.UI.Page
    {
        logs writelog = new logs();

       // private string a = string.Empty;

        protected void Page_Load(object sender, EventArgs e)
        {
            try
            {

                string resourcePath = "";
                string aliasName = "";
                string currency = "";
                string receiptURL = "";            // "http://10.44.71.108:8080/canon/jsp/TranPipeResult.jsp";
                string errorURL = "";            // "http://10.44.71.108:8080/canon/jsp/TranPipeResult.jsp";
                string language = "";


                string Path = Server.MapPath("file");

                writelog.AppActivityLogFile("file Path  :", Path + "\\fss.txt");

                System.IO.StreamReader myFile = new System.IO.StreamReader(Path + "\\fss.txt");

                string myString = myFile.ReadToEnd();
```

4. In the Add Reference Dialog box , browse the DLL's Physical path and select the DLL and click OK.


DotNetPlugin - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools Test Window Help
Debug
Any CPU
obj1.performTransaction()
Install Web Components
Publish: Create Publish Settings
fss.txt
fssHostedHttp.txt
HostedPaymentResult.aspx.cs
HostedPaymentBuy.aspx.cs
TranportalBuyIVROTPProcess.aspx.cs
DotNetPlugin.HostedPaymentBuy
Page_Load(o
Solution Explorer
Output
Ready

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Reflection;
using com.fss.plugin;
using System.Text;
using System.IO;


namespace DotNetPlugin
{
    public partial class HostedI
    {

        logs writelog = new logs

        protected void Page_Loa
        {
            try
            {
                string resourceI
                string aliasName
                string currency = "";
                string receiptURL = "";          // "http://10.44.71.108:8080/canon/jsp/Tr
                string errorURL = "";           // "http://10.44.71.108:8080/canon/jsp/Tra
                string language = "";


                string Path = Server.MapPath("file");
                writelog.AppActivityLogFile("file Path  :", Path + "\\HostedHTTPCB01.txt");
                System.IO.StreamReader myFile = new System.IO.StreamReader(Path + "\\HostedH
                string myString = myFile.ReadToEnd();
                writelog.AppActivityLogFile("file text :", myString);
```

Add Reference

.NET | COM | Projects | Browse | Recent

Look in: bin

IKVM.OpenJDK.Media.dll
IKVM.OpenJDK.Misc.dll
IKVM.OpenJDK.Naming.dll
IKVM.OpenJDK.Remoting.dll
IKVM.OpenJDK.Security.dll
IKVM.OpenJDK.SwingAWT.dll
IKVM.OpenJDK.Text.dll
IKVM.OpenJDK.Util.dll
IKVM.OpenJDK.XML.API.dll
IKVM.Runtime.dll
IKVM.Runtime.JNI.dll
iPayPipefinal.dll

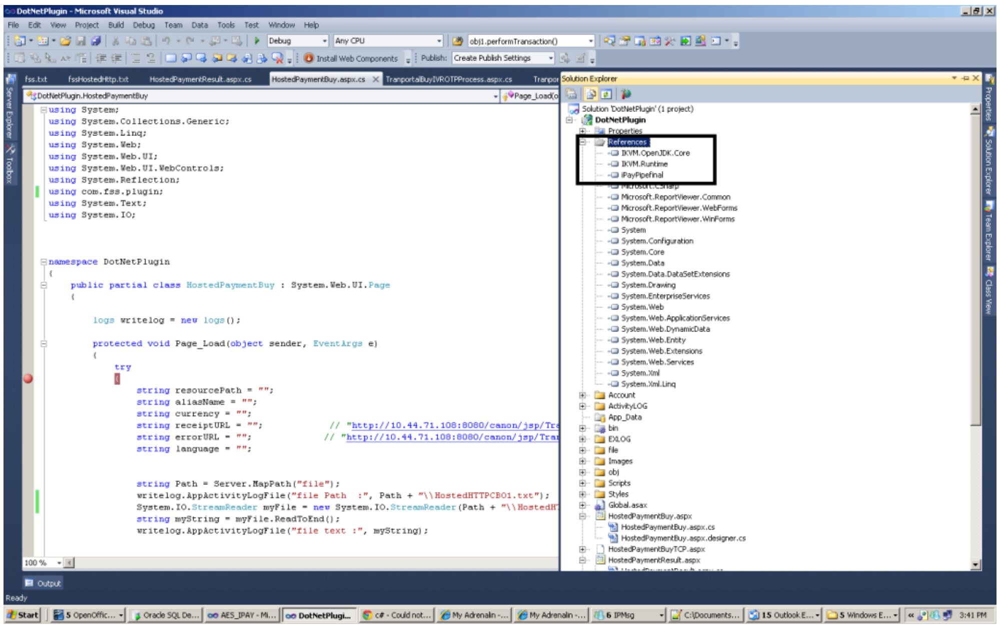
File name: iPayPipefinal

Files of type: Component Files (*.dll;*.tlb;*.olb;*.ocx;*.exe;*.manifest)

OK | Cancel

Solution 'DotNetPlugin' (1 project)
- DotNetPlugin
  - Properties
  - References
    - Microsoft.CSharp
    - Microsoft.ReportViewer.Common
    - Microsoft.ReportViewer.WebForms
    - Microsoft.ReportViewer.WinForms
    - System
    - System.Configuration
    - System.Core
    - System.Data
    - System.Data.DataSetExtensions
    - System.Drawing
    - System.EnterpriseServices
    - System.Web
    - System.Web.ApplicationServices
    - System.Web.DynamicData
    - System.Web.Entity
    - System.Web.Extensions
    - System.Web.Services
    - System.Xml
    - System.Xml.Linq
  - Account
  - ActivityLOG
  - App_Data
  - bin
  - EXLOG
  - file
  - Images
  - obj
  - Scripts
  - Styles
  - Global.asax
  - HostedPaymentBuy.aspx
    - HostedPaymentBuy.aspx.cs
    - HostedPaymentBuy.aspx.designer.cs
  - HostedPaymentBuyTCP.aspx
  - HostedPaymentResult.aspx
    - HostedPaymentResult.aspx.cs
    - HostedPaymentResult.aspx.designer.cs
  - HostedPaymentResultTCP.aspx

5. After adding, you could able to see the added DLL into the list of References


DotNetPlugin - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools Test Window Help
Debug
Any CPU
obj1.performTransaction()
Install Web Components
Publish: Create Publish Settings
fss.txt
fssHostedHttp.txt
HostedPaymentResult.aspx.cs
HostedPaymentBuy.aspx.cs
TranportalBuyIVROTPProcess.aspx.cs
DotNetPlugin.HostedPaymentBuy
Page_Load(o
Solution Explorer
Output
Ready

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Reflection;
using com.fss.plugin;
using System.Text;
using System.IO;


namespace DotNetPlugin
{
    public partial class HostedPaymentBuy : System.Web.UI.Page
    {

        logs writelog = new logs();

        protected void Page_Load(object sender, EventArgs e)
        {
            try
            {
                string resourcePath = "";
                string aliasName = "";
                string currency = "";
                string receiptURL = "";          // "http://10.44.71.108:8080/canon/jsp/Tr
                string errorURL = "";           // "http://10.44.71.108:8080/canon/jsp/Tra
                string language = "";


                string Path = Server.MapPath("file");
                writelog.AppActivityLogFile("file Path  :", Path + "\\HostedHTTPCB01.txt");
                System.IO.StreamReader myFile = new System.IO.StreamReader(Path + "\\HostedH
                string myString = myFile.ReadToEnd();
                writelog.AppActivityLogFile("file text :", myString);
```

Solution 'DotNetPlugin' (1 project)
- DotNetPlugin
  - Properties
  - References
    - IKVM.OpenJDK.Core
    - IKVM.Runtime
    - iPayPipefinal
    - Microsoft.CSharp
    - Microsoft.ReportViewer.Common
    - Microsoft.ReportViewer.WebForms
    - Microsoft.ReportViewer.WinForms
    - System
    - System.Configuration
    - System.Core
    - System.Data
    - System.Data.DataSetExtensions
    - System.Drawing
    - System.EnterpriseServices
    - System.Web
    - System.Web.ApplicationServices
    - System.Web.DynamicData
    - System.Web.Entity
    - System.Web.Extensions
    - System.Web.Services
    - System.Xml
    - System.Xml.Linq
  - Account
  - ActivityLOG
  - App_Data
  - bin
  - EXLOG
  - file
  - Images
  - obj
  - Scripts
  - Styles
  - Global.asax
  - HostedPaymentBuy.aspx
    - HostedPaymentBuy.aspx.cs
    - HostedPaymentBuy.aspx.designer.cs
  - HostedPaymentBuyTCP.aspx
  - HostedPaymentResult.aspx

likewise add the remaining DLL's into the appplication .

- In order to use the **iPayPipe** into the application, Add the java package into the Dotnet application by adding the, **using com.fss.plugin.bob;** namespace into the corresponding ASP pages.

Note:

Save or Store all the IKVM DLL's into the Particular path, after that refer the IKVM.OpenJDK.Core.dll and IKVM.Runtime.dll as mentioned in Step 2, into the application in order to avoid referring all those supporting DLL's into the application.
Once process are done correctly you could able to see all the DLL's (referred & Supported) into the BIN folder of the Application.

2.5.3 Code snippet for PHP Integration

Download the PHP kit and copy the lib files in the project location and respective path to be provided as mentioned below.

2.5.3.1 Hosted Payment Integration (Single Step Integration)

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

//Include or Require

```
require('../libfiles/iPay24Pipe.php');
```

//Initialization

```
        $myObj = new iPay24Pipe();

$myObj->setResourcePath(“c:\\resourcepath”);        //Resource File Path, provide the actual resource path
$myObj->setAlias(“aliasName”);    //Terminal Alias Name
// 1 – Purchase
$myObj->setAction(”1”);   //Transaction Action Code
//Transaction Currency
$myObj->setCurrency(“356”);        //Currency Code
$myObj->setLanguage(“USA”);        //Language
//Success URL
$myObj->setResponseURL(“http://www.demomerchant.com/result.jsp”);
//Error URL
$myObj->setErrorURL(“http://www.demomerchant.com/error.jsp”);
//Transaction Amount
$myObj->setAmt(“1500.00”);
//Merchant Track ID. Unique number generated by the Merchant
$myObj->setTrackId(“123456789”);
//Faster Checkout Related information
$myObj->setCustid("201502889575739");//Customer Id for faster checkout
//User Defined Fields
//Actual value should be provided properly during the UAT and production. UDF1 to UDF5 are conditional.
$myObj->setUdf1(“udf1”);
$myObj->setUdf2(“udf2”);
$myObj->setUdf3(“udf3”);
```

```
$myObj->setUdf4(”udf4”);      //For  Faster Checkout  ,it should be given as “FC”
$myObj->setUdf5(“udf5”);


//Merchant can use the UDF6 to UDF32 fields for passing merchant defined values like mobile number of the
customer, email ID etc.. These are non mandatory values.
$myObj->setUdf6(“udf6”);
$myObj->setUdf7(“udf7”);
$myObj->setUdf8(“udf8”);
$myObj->setUdf9(“udf9”);
$myObj->setUdf10(“udf10”);
$myObj->setUdf11(“udf11”);
$myObj->setUdf12(“udf12”);
$myObj->setUdf13(“udf13”);
$myObj->setUdf14(“udf14”);
$myObj->setUdf15(“udf15”);
$myObj->setUdf16(“udf16”);
$myObj->setUdf17(“udf17”);
$myObj->setUdf18(“udf18”);
$myObj->setUdf19(“udf19”);
$myObj->setUdf20(“udf20”);
$myObj->setUdf21(“udf21”);
$myObj->setUdf22(“udf22”);
$myObj->setUdf23(“udf23”);
$myObj->setUdf24(“udf24”);
$myObj->setUdf25(“udf25”);
$myObj->setUdf26(“udf26”);
$myObj->setUdf27(“udf27”);
$myObj->setUdf28(“udf28”);
$myObj->setUdf29(“udf29”);
$myObj->setUdf30(“udf30”);
$myObj->setUdf31(“udf31”);
$myObj->setUdf32(“udf32”);
```

// For **Hosted Payment Integration(Single Step integration)**, the method to be called is $myObj-

```
>performPaymentInitializationHTTP()

//To redirect the web address.

header("location:".$myObj->getwebAddress());

/** End of Request Processing**/
```

/ Response received from Payment Gateway to Merchant **/**

```
trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F
```

//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,

//Include or Require

require('../libfiles/iPay24Pipe.php');

//Initialization

$myObj = new iPay24Pipe();

//Set Values

$myObj->setResourcePath("c:\\resourcepath"); //Resource File Path
$myObj->setAlias("aliasName"); //Terminal Alias Name
The method to be called is,

$trandata = isset($_GET["trandata"]) ? $_GET["trandata"] : isset($_POST["trandata"]) ? $_POST["trandata"] : "";

$myObj->parseEncryptedRequest(trim($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.

Ex:

```
// To get result,
        $myObj->getResult(); // Ex: CAPTURED or SUCCESS or APPROVED or VOIDED
// 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"]
        // To get Transaction ID
                $myObj->getTransId();
        // To get Payment ID
                $myObj->getPaymentId();
```

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**/**

2.5.3.2 Hosted Payment Integration (Two Step Integration)

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
//Include or Require

require('../libfiles/iPay24Pipe.php');
```

//Initialization

```
$myObj = new iPay24Pipe();

$myObj->setResourcePath("c:\\resourcepath");        //Resource File Path
$myObj->setAlias("aliasName");     //Terminal Alias Name
// 1 – Purchase, 4 – Auth, 17 – IMPS Transaction
$myObj->setAction("1");   //Transaction Action Code
//Transaction Currency
$myObj->setCurrency("356");        //Currency Code
$myObj->setLanguage("USA");      //Language
//Success URL
$myObj->setResponseURL("http://www.demomerchant.com/result.jsp");
//Error URL
$myObj->setErrorURL("http://www.demomerchant.com/error.jsp");
//Transaction Amount
$myObj->setAmt("1500.00");
//Merchant Track ID
$myObj->setTrackId("123456789");
//Faster Checkout Related information
$myObj->setCustid("201502889575739");//Customer Id for faster checkout
//User Defined Fields
//Actual value should be provided properly during the UAT and production. UDF1 to UDF5 are conditional.
$myObj->setUdf1("udf1");
$myObj->setUdf2("udf2");
$myObj->setUdf3("udf3");
$myObj->setUdf4("udf4");     //For  Faster Checkout  ,it should be given as "FC"
$myObj->setUdf5("udf5");

//Merchant can use the UDF6 to UDF32 fields for passing merchant defined values like mobile number of the customer, email ID etc.. These are non mandatory values.

$myObj->setUdf6("udf6");
$myObj->setUdf7("udf7");
```

```
$myObj->setUdf8(“udf8”);
$myObj->setUdf9(“udf9”);
$myObj->setUdf10(“udf10”);
$myObj->setUdf11(“udf11”);
$myObj->setUdf12(“udf12”);
$myObj->setUdf13(“udf13”);
$myObj->setUdf14(“udf14”);
$myObj->setUdf15(“udf15”);
$myObj->setUdf16(“udf16”);
$myObj->setUdf17(“udf17”);
$myObj->setUdf18(“udf18”);
$myObj->setUdf19(“udf19”);
$myObj->setUdf20(“udf20”);
$myObj->setUdf21(“udf21”);
$myObj->setUdf22(“udf22”);
$myObj->setUdf23(“udf23”);
$myObj->setUdf24(“udf24”);
$myObj->setUdf25(“udf25”);
$myObj->setUdf26(“udf26”);
$myObj->setUdf27(“udf27”);
$myObj->setUdf28(“udf28”);
$myObj->setUdf29(“udf29”);
$myObj->setUdf30(“udf30”);
$myObj->setUdf31(“udf31”);
$myObj->setUdf32(“udf32”);
```

// For **Hosted Payment Integration (Two Step integration)**, the method to be called is

// Step 1

$myObj->performPaymentInitialization();

// Then they have to redirect the paymentpage url with payment ID which is sent by Payment Gateway.

//Step 2

response.sendRedirect (**pipe.getPaymentPage ()**+”?PaymentID”+ **pipe.getPaymentId ()**);

$trandata = $myObj->getPaymentPage()+”?PaymentID”+ $myObj->getPaymentId ()

header("location:".$trandata);

/** End of Request Processing**/

/ Response received from Payment Gateway to Merchant **/**

Here the above mentioned response parameters are sent as opened text.

Response to Merchant:

```
paymentid=201518226389866&result=SUCCESS&ref=518210000087&tranid=201518226408041&custid=&postd
ate=0701&trackid=1263556640&udf1=0&udf2=8681F6DEAB81434EFE33F9F591CF09F9&udf3=8681F6DEAB814
34E2B8E695FC76E4DA8&udf4=FC&udf5=Tidal+5&amt=10000.0
```

Merchant should follow the steps to get the response fields.

```
        //To get Result

                $_GET["result"]

        //To get Error

                $_GET["ErrorText"]

         //To get Transaction ID

                $_GET["tranid"]

        //To get Payment ID

                $_GET["paymentid"]

        //To get Transaction Amount

                $_GET["amt"]

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

2.5.3.3 Tranportal Transaction – With Encryption

[Merchant saved card option use this integration method]

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
//Include or Require
require('../libfiles/iPay24Pipe.php');
//Initialization
$myObj = new iPay24Pipe();
$myObj->setResourcePath("c:\\resourcepath");          //Resource File Path
$myObj->setAlias("aliasName");     //Terminal Alias Name
// 1 – Purchase, 4 – Auth, 17 – IMPS Transaction
$myObj->setAction("1");   //Transaction Action Code
//Transaction Currency
$myObj->setCurrency("356");          //Currency Code
$myObj->setLanguage("USA");       //Language
//Success URL
$myObj->setResponseURL("http://www.demomerchant.com/result.jsp");
//Error URL
$myObj->setErrorURL("http://www.demomerchant.com/error.jsp");
//Transaction Amount
$myObj->setAmt("1500.00");
//Merchant Track ID
$myObj->setTrackId("123456789");
//Faster Checkout Related information
$myObj->setCustid("201502889575739");//Customer Id for faster checkout
//User Defined Fields
//Actual value should be provided properly during the UAT and production. UDF1 to UDF5 are conditional and not mandatoty.
$myObj->setUdf1("udf1");
$myObj->setUdf2("udf2");
$myObj->setUdf3("udf3");
$myObj->setUdf4("udf4");      //For  Faster Checkout  ,it should be given as "FC"
$myObj->setUdf5("udf5");

//Merchant can use the UDF6 to UDF32 fields for passing merchant defined values like mobile number of the customer, email ID etc.. These are non mandatory values.

$myObj->setUdf6("udf6");
$myObj->setUdf7("udf7");
$myObj->setUdf8("udf8");
```

```
$myObj->setUdf9(“udf9”);
$myObj->setUdf10(“udf10”);
$myObj->setUdf11(“udf11”);
$myObj->setUdf12(“udf12”);
$myObj->setUdf13(“udf13”);
$myObj->setUdf14(“udf14”);
$myObj->setUdf15(“udf15”);
$myObj->setUdf16(“udf16”);
$myObj->setUdf17(“udf17”);
$myObj->setUdf18(“udf18”);
$myObj->setUdf19(“udf19”);
$myObj->setUdf20(“udf20”);
$myObj->setUdf21(“udf21”);
$myObj->setUdf22(“udf22”);
$myObj->setUdf23(“udf23”);
$myObj->setUdf24(“udf24”);
$myObj->setUdf25(“udf25”);
$myObj->setUdf26(“udf26”);
$myObj->setUdf27(“udf27”);
$myObj->setUdf28(“udf28”);
$myObj->setUdf29(“udf29”);
$myObj->setUdf30(“udf30”);
$myObj->setUdf31(“udf31”);
$myObj->setUdf32(“udf32”);

//Saved card Flag
$myObj->setSavedCard(“Y”); // For Merchant saved card option this field should be given.( value-“Y”)

//For Tranportal Transaction with 3-D Secure transaction, the method to be called is
```

$myObj->performVbVTransaction();

```
// Then the Merchant have to redirect the Payment Gateway url.

header("location:".$myObj->getwebAddress());

/** End of Request Processing**/
```

/ Response received from Payment Gateway to Merchant **/**

trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A34694348506778257040533E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F

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

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

//Include or Require
require('../libfiles/iPay24Pipe.php');
//Initialization
$myObj = new iPay24Pipe();
//Set Values
$myObj->setResourcePath("c:\\resourcepath"); //Resource File Path
$myObj->setAlias("aliasName"); //Terminal Alias Name
The method to be called is,
$trandata = isset($_GET["trandata"]) ? $_GET["trandata"] : isset($_POST["trandata"]) ? $_POST["trandata"] : "";
$myObj->parseEncryptedRequest(trim($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.

Ex:

```
// To get result,
        $myObj->getResult(); // Ex: CAPTURED or SUCCESS or APPROVED or VOIDED
// 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"]
```

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

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"]
```

2.5.3.4 Tranportal Refund Transaction (Credit) – With Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
//Include or Require
require('../libfiles/iPay24Pipe.php');

//Initialization

$myObj = new iPay24Pipe();
$myObj->setResourcePath("c:\\resourcepath");          //Resource File Path
$myObj->setAlias("aliasName");     //Terminal Alias Name

// 2 – Credit
$myObj->setAction("2");   //Transaction Action Code

// Original Transaction Instrument Type
$myObj->setType("D");  // D (Debit) or C (Credit)

//Transaction Currency
$myObj->setCurrency("356");          //Currency Code

$myObj->setLanguage("USA");       //Language

//Success URL
$myObj->setResponseURL("http://www.demomerchant.com/result.jsp");

//Error URL
$myObj->setErrorURL("http://www.demomerchant.com/error.jsp");

//Transaction Amount
$myObj->setAmt("1500.00");

//Merchant Track ID
$myObj->setTrackId("123456789");

//Faster Checkout Related information
$myObj->setCustid("201502889575739");//Customer Id for faster checkout

//Original Transaction ID
$myObj->setTransId("201555248554515");

//User Defined Fields
//Actual value should be provided properly during the UAT and production.
$myObj->setUdf1("udf1");
$myObj->setUdf2("udf2");
$myObj->setUdf3("udf3");
$myObj->setUdf4("udf4");      //For  Faster Checkout  ,it should be given as "FC"

//Values to set in UDF5 based on the Original Transaction ID set in TransId as follows:

//If TransID contains Original Payment ID → UDF5 -   PaymentID

//If TransID contains Original Track ID → UDF5 -   TrackID

//If TransID contains Original Tran ID → UDF5 -  TRANID

$myObj->setUdf5("udf5");   // PaymentID or  TrackID or  TRANID


//Merchant can use the UDF6 to UDF32 fields for passing merchant defined values like mobile number of the
customer, email ID etc.. These are non mandatory values.
```

```
$myObj->setUdf6(“udf6”);
$myObj->setUdf7(“udf7”);
$myObj->setUdf8(“udf8”);
$myObj->setUdf9(“udf9”);
$myObj->setUdf10(“udf10”);
$myObj->setUdf11(“udf11”);
$myObj->setUdf12(“udf12”);
$myObj->setUdf13(“udf13”);
$myObj->setUdf14(“udf14”);
$myObj->setUdf15(“udf15”);
$myObj->setUdf16(“udf16”);
$myObj->setUdf17(“udf17”);
$myObj->setUdf18(“udf18”);
$myObj->setUdf19(“udf19”);
$myObj->setUdf20(“udf20”);
$myObj->setUdf21(“udf21”);
$myObj->setUdf22(“udf22”);
$myObj->setUdf23(“udf23”);
$myObj->setUdf24(“udf24”);
$myObj->setUdf25(“udf25”);
$myObj->setUdf26(“udf26”);
$myObj->setUdf27(“udf27”);
$myObj->setUdf28(“udf28”);
$myObj->setUdf29(“udf29”);
$myObj->setUdf30(“udf30”);
$myObj->setUdf31(“udf31”);
$myObj->setUdf32(“udf32”);

//The method to be called is
```

$myObj->performTransactionHTTP();

```
// Then the Merchant have to redirect the Payment Gateway url.

header("location:".$myObj->getwebAddress());

/** End of Request Processing**/
```

/ Response received from Payment Gateway to Merchant **/**

```
trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A346943485067782570405 33E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F
```

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

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

//Include or Require

```
require('../libfiles/iPay24Pipe.php');
```

//Initialization

```
$myObj = new iPay24Pipe();
```

//Set Values

```
$myObj->setResourcePath(“c:\\resourcepath”);          //Resource File Path
$myObj->setAlias(“aliasName”);      //Terminal Alias Name
```

The method to be called is,

```
$trandata =  isset($_GET["trandata"]) ? $_GET["trandata"] : isset($_POST["trandata"]) ? $_POST["trandata"] : "";
```
$myObj->parseEncryptedRequest(trim($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.

Ex:

```
// To get result,
        $myObj->getResult(); // Ex: CAPTURED or SUCCESS or APPROVED or VOIDED
// 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"]
// To get Transaction ID
        $myObj->getTransId();
// To get Payment ID
        $myObj->getPaymentId();
```

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**/
```

2.5.3.5 Tranportal Refund Transaction (Credit) – Without Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

//Include or Require

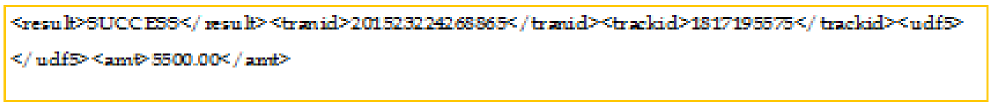
```
require('../libfiles/iPay24Pipe.php');
```

//Initialization

```
$myObj = new iPay24Pipe();
$myObj->setResourcePath("c:\\resourcepath");         //Resource File Path
$myObj->setAlias("aliasName");     //Terminal Alias Name

// 2 – Credit
$myObj->setAction("2");   //Transaction Action Code

// Original Transaction Instrument Type
$myObj->setType("D");  // D (Debit) or C (Credit)

//Transaction Currency
$myObj->setCurrency("356");          //Currency Code

$myObj->setLanguage("USA");       //Language

//Success URL
$myObj->setResponseURL("http://www.demomerchant.com/result.jsp");

//Error URL
$myObj->setErrorURL("http://www.demomerchant.com/error.jsp");

//Transaction Amount
$myObj->setAmt("1500.00");
```

```
//Merchant Track ID
$myObj->setTrackId("123456789");

//Faster Checkout Related information
$myObj->setCustid("201502889575739");//Customer Id for faster checkout

//Original Transaction ID
$myObj->setTransId("201555248554515");

//User Defined Fields
//Actual value should be provided properly during the UAT and production.
$myObj->setUdf1("udf1");
$myObj->setUdf2("udf2");
$myObj->setUdf3("udf3");
$myObj->setUdf4("udf4");      //For  Faster Checkout  ,it should be given as "FC"

//Values to set in UDF5 based on the Original Transaction ID set in TransId as follows:

//If TransID contains Original Payment ID → UDF5 -   PaymentID

//If TransID contains Original Track ID → UDF5 -   TrackID

//If TransID contains Original Tran ID → UDF5 -  TRANID

$myObj->setUdf5("udf5");   // PaymentID or  TrackID or  TRANID

//Merchant can use the UDF6 to UDF32 fields for passing merchant defined values like mobile number of the customer, email ID etc.. These are non mandatory values.


$myObj->setUdf6("udf6");
$myObj->setUdf7("udf7");
$myObj->setUdf8("udf8");
$myObj->setUdf9("udf9");
$myObj->setUdf10("udf10");
$myObj->setUdf11("udf11");
$myObj->setUdf12("udf12");
$myObj->setUdf13("udf13");
$myObj->setUdf14("udf14");
$myObj->setUdf15("udf15");
$myObj->setUdf16("udf16");
$myObj->setUdf17("udf17");
$myObj->setUdf18("udf18");
$myObj->setUdf19("udf19");
$myObj->setUdf20("udf20");
$myObj->setUdf21("udf21");
$myObj->setUdf22("udf22");
$myObj->setUdf23("udf23");
$myObj->setUdf24("udf24");
$myObj->setUdf25("udf25");
$myObj->setUdf26("udf26");
$myObj->setUdf27("udf27");
$myObj->setUdf28("udf28");
$myObj->setUdf29("udf29");
$myObj->setUdf30("udf30");
$myObj->setUdf31("udf31");
$myObj->setUdf32("udf32");


//The method to be called is

$myObj->performTransaction();

/** End of Request Processing**/
```

/ Response received from Payment Gateway to Merchant in TCPIP **/**



```
<result>SUCCESS</result><tranid>201523224268865</tranid><trackid>1817195575</trackid><udf5>
</udf5><amt>5500.00</amt>
```

Merchant should follow the steps to get the response fields.

```
            // To get result,
$myObj->getResult() // Ex: CAPTURED or SUCCESS  or VOIDED


            // To get payment ID
$myObj->getPaymentId()
            //To get Transaction ID
                        $myObj->getTransId()
            // To get Amount
$myObj->getAmt()
            //To get  Error(If error occured)
                         $myObj->getError()


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

2.5.3.6 Tranportal Inquiry Transaction – With Encryption

/ Request Processing**/**

Merchant can connect iPay Plugin using below step

```
//Include or Require
require('../libfiles/iPay24Pipe.php');

//Initialization

$myObj = new iPay24Pipe();
$myObj->setResourcePath("c:\\resourcepath");          //Resource File Path
$myObj->setAlias("aliasName");     //Terminal Alias Name

// 8 – Inquiry
$myObj->setAction("8");   //Transaction Action Code

// Original Transaction Instrument Type
$myObj->setType("D");  // D (Debit) or C (Credit)

//Transaction Currency
$myObj->setCurrency("356");          //Currency Code

$myObj->setLanguage("USA");       //Language

//Success URL
$myObj->setResponseURL("http://www.demomerchant.com/result.jsp");

//Error URL
$myObj->setErrorURL("http://www.demomerchant.com/error.jsp");

//Transaction Amount
$myObj->setAmt("1500.00");

//Merchant Track ID
$myObj->setTrackId("123456789");

//Faster Checkout Related information
$myObj->setCustid("201502889575739");//Customer Id for faster checkout

//Original Transaction ID
$myObj->setTransId("201555248554515");

//User Defined Fields
//Actual value should be provided properly during the UAT and production.
$myObj->setUdf1("udf1");
$myObj->setUdf2("udf2");
$myObj->setUdf3("udf3");
$myObj->setUdf4("udf4");      //For  Faster Checkout  ,it should be given as "FC"

//Values to set in UDF5 based on the Original Transaction ID set in TransId as follows:

//If TransID contains Original Payment ID → UDF5 -   PaymentID

//If TransID contains Original Track ID → UDF5 -   TrackID

//If TransID contains Original Tran ID → UDF5 -  TRANID

$myObj->setUdf5("udf5");   // PaymentID or  TrackID or  TRANID


//Merchant can use the UDF6 to UDF32 fields for passing merchant defined values like mobile number of the
customer, email ID etc.. These are non mandatory values.
```

```
$myObj->setUdf6(“udf6”);
$myObj->setUdf7(“udf7”);
$myObj->setUdf8(“udf8”);
$myObj->setUdf9(“udf9”);
$myObj->setUdf10(“udf10”);
$myObj->setUdf11(“udf11”);
$myObj->setUdf12(“udf12”);
$myObj->setUdf13(“udf13”);
$myObj->setUdf14(“udf14”);
$myObj->setUdf15(“udf15”);
$myObj->setUdf16(“udf16”);
$myObj->setUdf17(“udf17”);
$myObj->setUdf18(“udf18”);
$myObj->setUdf19(“udf19”);
$myObj->setUdf20(“udf20”);
$myObj->setUdf21(“udf21”);
$myObj->setUdf22(“udf22”);
$myObj->setUdf23(“udf23”);
$myObj->setUdf24(“udf24”);
$myObj->setUdf25(“udf25”);
$myObj->setUdf26(“udf26”);
$myObj->setUdf27(“udf27”);
$myObj->setUdf28(“udf28”);
$myObj->setUdf29(“udf29”);
$myObj->setUdf30(“udf30”);
$myObj->setUdf31(“udf31”);
$myObj->setUdf32(“udf32”);
```

//The method to be called is

$myObj->performTransactionHTTP();

// Then the Merchant have to redirect the Payment Gateway url.

header("location:".$myObj->getwebAddress());

/** End of Request Processing**/

/ Response received from Payment Gateway to Merchant **/**

```
trandata=1C1A967D16C877543E0A1DD90D2933853F05BB0AA2E6B47BB77D27EEAF32EF02CF6A0A5643F31C
78340913929D90879615DFA9CDCCBB03761B9AE87CD76FE8633E0A1DD90D29338545DA582B0F3500BA93753
13637690531C6D7F8F7489C7CD8B73F9EC7E1C622DDD06B0809A709C2CB2A6EFD72F36FEB044B73810204E69
FC577300F9AAF38E044F0A3469434850677825704O533E4FD48A9C61DD83E906AB93110CE0E57A1C548FA589
B8F8566FC9F
```

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

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

//Include or Require

require('../libfiles/iPay24Pipe.php');

//Initialization

$myObj = new iPay24Pipe();

//Set Values

```
$myObj->setResourcePath("c:\\resourcepath");          //Resource File Path
$myObj->setAlias("aliasName");     //Terminal Alias Name
```

The method to be called is,

```
$trandata =  isset($_GET["trandata"]) ? $_GET["trandata"] : isset($_POST["trandata"]) ? $_POST["trandata"] : "";
$myObj->parseEncryptedRequest(trim($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.

Ex:

```
// To get result,
        $myObj->getResult(); // Ex: CAPTURED or SUCCESS or APPROVED or VOIDED
// 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"]
// To get Transaction ID
        $myObj->getTransId();
// To get Payment ID
        $myObj->getPaymentId();
```

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**/**

Appendix 1

The below are the details Merchant web application should send while connecting Payment Gateway

S. No	Merchant Request Data	Type	Max	Mandatory
1	Tranportal ID	Alpha Numeric	10	Yes
2	Transaction Action Type	Number	10	Yes
3	Response URL	Text	255	Yes
4	Error URL	Text	255	Yes
5	Amount	Number with decimals		Yes
6	Currency	Number	3	Yes
7	Track ID [Text Field]	Numeric or Alpha Numeric	255	Yes
8	UDF 1 [Text Field - Optional]	Alpha Numeric	255	No
9	UDF 2 [Text Field - Optional]	Alpha Numeric	255	No
10	UDF 3 [Text Field - Optional]	Alpha Numeric	255	No
11	UDF 4 [Text Field - Optional]	Alpha Numeric	255	No, Yes(For Faster Checkout)
12	UDF 5 [Text Field - Optional]	Alpha Numeric	255	No
13	Language	Text	50	No
14	Customer ID(Specific for Faster Checkout)	Number	25	Yes
15	Mobile Number(In case of IMPS)	Number	15	Yes
16	MMID(In case of IMPS)	Number	7	Yes
17	OTP(In case of IMPS)	Number	6	Yes

Merchant Request Parameters & Description:

1. **Tranportal ID**: Unique ID assigned for a Terminal under a merchant. (ex: 101001)

2. **Transaction Action Type**: Describes the received transaction type

(ex: 1 – Purchase, 2 – Authorization,17- IMPS etc)

3. **Response URL:** Merchant response URL

(ex: http://10.44.71.46/demomerchant/successresult.jsp); if authentication and authorization successful at the Payment Gateway then response would be sent to the Response URL.

4. **Error URL:** Merchant Error URL

(ex: http:// 10.44.71.46/demomerchant/errorresult.jsp); if authentication and authorization failed at the Payment Gateway then response would be sent to the error URL.

5. **Amount:** Transaction amount to be charged to the Customer for the product purchased (ex: Rs. 100.00).

6. **Currency:** ISO Currency code (ex: 356 [for INR]) to be set by the merchant for initiating the

7. **Track ID [Text Field]:** Allows you to map unique track ID to every transaction originated by the merchant (ex: 112312312313)

8. **UDF 1 [Text Field - Optional]:** Allows Merchant to send any additional information

9. **UDF 2 [Text Field - Optional]:** Allows Merchant to send any additional information

10. **UDF 3 [Text Field - Optional]:** Allows Merchant to send any additional information

11. **UDF 4 [Text Field - Optional]:** Allows Merchant to send any additional information .In case of Faster Checkout ,this field should be sent by Merchant as “FC”.

12. **UDF 5 [Text Field - Optional]:** Allows Merchant to send any additional information

13. **Language:** Describes the consumer language.(ex: USA).

14. **Customer ID:** Unique ID which is specific for Merchant.

15. **Mobile Number:** Customer’s Mobile number.

16. **MMID:** Mobile Money Identifier which is given by bank to the customer.

17. **OTP:** One Time Password.

Appendix 2

The below are the details Payment Gateway shall send to Merchant web application. In **Single Step Integration**, these fields are encrypted and sent in **trandata** field to the Merchant.

S. No	Payment Gateway Response Parameters	Type
1	Payment ID	Number
2	Result	Text
3	Reference ID	Number
4	Transaction ID	Number
5	Customer ID(In case of Standing Instruction)	Number
6	Post Date	Number
7	UDF 1 [Text Field - Optional]	Text
8	UDF 2 [Text Field - Optional]	Text
9	UDF 3 [Text Field - Optional]	Text
10	UDF 4 [Text Field - Optional]	Text
11	UDF 5 [Text Field - Optional]	Text
12	Amount	Number with decimals
13	Error	Text
14	Error Text	Text

Response Parameters Description:

1. **Payment ID:** Unique ID which is generated if the request from the merchant received and successfully validated at the payment gateway.
2. **Result:** Describes the result of the transaction (ex: **CAPTURED**).
3. **Reference ID :** RRN –generated by Base-24
4. **Transaction ID:** Unique ID generated by Payment Gateway for the transaction.
5. **Customer ID:** Unique ID generated by Payment Gateway for Standing Instruction.
6. **Post Date:** Transaction post date.
7. **UDF 1 [Text Field - Optional]:** Additional information
8. **UDF 2 [Text Field - Optional]:** Additional information
9. **UDF 3 [Text Field - Optional]:** Additional information
10. **UDF 4 [Text Field - Optional]:** Additional information
11. **UDF 5 [Text Field - Optional]:** Additional information
12. **Amount:** Transaction Amount

In case of failure transaction, Payment Gateway shall send the below mentioned parameters with Payment ID, Result, UDF fields, Amount.

1. **Error:** Describes the error with Error code.
2. **Error Text:** Describes the error with Error code.

Appendix 3

Intermediate Response Parameters

S. No	Response Parameters	Type
1	Payment ID	Number
2	Payment Page URL	Text

1. Payment ID: Unique ID which is generated if the request from the merchant received and successfully validated at the payment gateway.

2. Payment Page URL: URL of the payment page on which customer has to enter the card credentials.

Appendix 4

Final Response and Description

S. No	Final Response	Description
1	CAPTURED or APPROVED or VOIDED or SUCCESS	Result of successful transaction.
2	Error or ErrorText	Result of failure transaction.

Appendix 5

Transaction Relevant Data to be passed as transaction request to Payment Gateway.

Please set the relevant customer data as below into UDF fields by Using setUdf<> methods.

Data	**UDF Field Name**	**Sample Values**
Merchant name	UDF 6	Merchant Name
Customer Name	UDF 7	Firstname Lastname
Email Id	UDF 8	sample@abc.com
Mobile Number	UDF 9	9999999999
Address	UDF 10	Customer Address
Original Txn Amount	UDF 11	Original Amount
Txn Ref.	UDF 12	Any Txn details as required by Merchant
Merchant TransactionID	UDF 13	Merchant txn id (passed from merchant side)

Error Code Details:

IPAY0100056 - Instrument not allowed in Terminal and Brand
IPAY0100203 - Problem occured while doing perform transaction
IPAY0100290 - Problem occured while validating original transaction
IPAY0100136 - Transaction denied due to previous capture check failure (Validate Original Transaction)
IPAY0100199 - Transaction denied due to previous credit check failure (Validate Original Transaction)
IPAY0100140 - Transaction denied due to previous void check failure (Validate Original Transaction)
IPAY0100137 - Transaction denied due to credit amount greater than debit amount 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)
IPAY0100141 - Transaction denied due to authorization already captured (Validate Original Transaction)
IPAY0200079 - Chargeback transaction not allowed.
IPAY0100163 - Problem occured during transaction.
IPAY0100180 - Authentication not available.
IPAY0100205 - Problem occurred while getting PARES details.
IPAY0200001 - Problem occured while getting terminal.
IPAY0200017 - Problem occurred while getting payment instrument list
IPAY0200016 - Problem occured while getting payment instrument.
IPAY0100206 - Problem occurred while getting currency minor digits
IPAY0200022 - Problem occured while getting currency.
IPAY0200041 - Problem occured while getting institution configuration.
IPAY0200004 - Problem occured while getting password security rules.
IPAY0200042 - Problem occured while getting brand.
IPAY0200008 - Problem occured while verifying payment details.
IPAY0200043 - Problem occured while getting bin range details.
IPAY0200044 - Problem occured while adding transaction log details.
IPAY0100207 - Bin range not enabled.
IPAY0100208 - Action not enabled.
IPAY0100125 - Payment instrument not enabled.
IPAY0100186 - Encryption enabled.
IPAY0100034 - Currency code not enabled.
IPAY0100209 - Institution config not enabled.

IPAY0100016 - Password security not enabled.
IPAY0100042 - Transaction time limit exceeds.
IPAY0100041 - Payment details missing.
IPAY0100037 - Payment id missing.
IPAY0100039 - Invalid payment id .
IPAY0100142 - Problem occurred while validating original transaction
IPAY0100053 - Problem occured while processing direct debit.
IPAY0100038 - Unable to process the request.
IPAY0100160 - Unable to process the transaction.
IPAY0100100 - Problem occured while authorize
IPAY0100210 - Problem occured during veres process.
IPAY0100211 - Problem occured during pareq process.
IPAY0100212 - Problem occured while getting veres.
IPAY0200023 - Problem occured while determining payment instrument.
IPAY0100213 - Problem occured while processing the hosted transaction request.
IPAY0100014 - Terminal Authentication requested with invalid tranportal id data.
IPAY0100015 - Invalid tranportal password.
IPAY0100019 - Invalid login attempt.
IPAY0100017 - Inactive terminal.
IPAY0100018 - Terminal password expired.
IPAY0200007 - Problem occured while validating payment details
IPAY0100214 - Problem occurred while verifying tranportal id.
IPAY0200006 - Problem occurred while verifying tranportal password.
IPAY0100215 - Invalid tranportal id.
IPAY0100020 - Invalid action type.
IPAY0100021 - Missing currency.
IPAY0100022 - Invalid currency.
IPAY0100216 - Invalid data received.
IPAY0100217 - Invalid payment detail.
IPAY0100023 - Missing amount.
IPAY0100024 - Invalid amount.
IPAY0100218 - Invalid brand id.
IPAY0100105 - Action type not supported by maestro brand.
IPAY0100219 - Missing card number.
IPAY0100220 - Invalid card number.

IPAY0100221 - Missing card holder name.
IPAY0100222 - Invalid card holder name.
IPAY0100069 - Missing payment instrument.
IPAY0100106 - Invalid payment instrument.
IPAY0100107 - Instrument not enabled.
IPAY0100223 - Missing cvv.
IPAY0100224 - Invalid cvv.
IPAY0100162 - Merchant is not allowed for encryption process.
IPAY0100011 - Merchant has not enabled for encryption process.
IPAY0100010 - Institution has not enabled for the encryption process.
IPAY0100225 - Missing card expiry year.
IPAY0100226 - Invalid card expiry year.
IPAY0100227 - Missing card expiry month.
IPAY0100228 - Invalid card expiry month.
IPAY0100291 - Transaction denied due to invalid PIN
IPAY0100292 - Transaction denied due to missing PIN
IPAY0100229 - Invalid card expiry day.
IPAY0100230 - Card expired.
IPAY0100027 - Invalid track id.
IPAY0100231 - Invalid user defined field.
IPAY0100028 - Invalid user defined field1.
IPAY0100029 - Invalid user defined field2.
IPAY0100030 - Invalid user defined field3.
IPAY0100031 - Invalid user defined field4.
IPAY0100032 - Invalid user defined field5.
IPAY0100026 - Invalid language id
IPAY0100025 - Invalid amount or currency.
IPAY0100232 - Missing original transaction id
IPAY0100233 - Invalid original transaction id
IPAY0100001 - Missing error url.
IPAY0100002 - Invalid error url.
IPAY0100003 - Missing response url.
IPAY0100004 - Invalid response url.
IPAY0100005 - Missing tranportal id.
IPAY0100007 - Missing transaction data.

IPAY0100006 - Invalid tranportal id.
IPAY0100013 - Invalid transaction data.
IPAY0100095 - Terminal inactive.
IPAY0200018 - Problem occurred while getting transaction details
IPAY0200009 - Problem occurred while getting payment details.
IPAY0100044 - Problem occured while loading payment page.
IPAY0200025 - Problem occurred while getting terminal details.
IPAY0200002 - Problem occurred while getting institution details.
IPAY0200003 - Problem occurred while getting merchant details.
IPAY0200056 - Problem occurred while getting brand details.
IPAY0200038 - Problem occurred while getting vpas merchant details.
IPAY0200024 - Problem occurred while getting brand rules details.
IPAY0200057 - Problem occurred while getting external connection details.
IPAY0200058 - Problem occured while updating message log 2fa details.
IPAY0200059 - Problem occured while updating vpas details.
IPAY0200060 - Problem occured while adding vpas details.
IPAY0200033 - Problem occured while getting vpas log details.
IPAY0200061 - Problem occured during batch 2fa process.
IPAY0200062 - Problem occured while getting brand rules details.
IPAY0200029 - Problem occured while getting external connection details.
IPAY0200012 - Problem occured while updating payment log ip details.
IPAY0200063 - Problem occured while updating payment log process code details.
IPAY0200064 - Problem occured while updating payment log process code and ip details.
IPAY0200065 - Problem occured while updating payment log description details.
IPAY0200066 - Problem occured while updating payment log instrument details.
IPAY0200067 - Problem occured while updating payment log udf Fields.
IPAY0200069 - Problem occured while updating payment log card details.
IPAY0200011 - Problem occured while getting ipblock details.
IPAY0200070 - Problem occured while updating ipblock details.
IPAY0100178 - Merchant encryption enabled.
IPAY0100254 - Merchant not enabled for performing transaction.
IPAY0100255 - External connection not enabled.
IPAY0100126 - Brand not enabled.
IPAY0100257 - Brand rules not enabled.
IPAY0100182 - Vpas merchant not enabled.
IPAY0100008 - Terminal not enabled.
IPAY0200015 - Problem occured while getting terminal details.
IPAY0100009 - Institution not enabled.
IPAY0100046 - Payment option not enabled.
IPAY0100033 - Terminal action not enabled.

IPAY0100260 - Payment option(s) not enabled
IPAY0100054 - Payment details not available.
IPAY0200072- Payment log details not available.
IPAY0100263 - Transaction details not available.
IPAY0200026 - Problem occured while getting transaction log details.
IPAY0100243 - NOT SUPPORTED
IPAY0100242 - RC_UNAVAILABLE
IPAY0100036 - UDF MISMATCHED
IPAY0100045 - DENIED BY RISK
IPAY0100266 - Brand directory unavailable.
IPAY0100268 - 3d secure not enabled for the brand
IPAY0100269 - Invalid card check digit
IPAY0100185 - Problem occured while authentication
IPAY0100270 - pares not successfull
IPAY0100267 - PARES status not sucessfull.
IPAY0100265 - PARES validation failed.
IPAY0100264 - Signature validation failed.
IPAY0100258 - Certification verification failed.
IPAY0100262 - Problem occured during VEREQ process.
IPAY0200071 - Probelm occured during authentication.
IPAY0100181 - Card encryption failed.
IPAY0100051 - Missing terminal key.
IPAY0100050 - Invalid terminal key.
IPAY0100176 - Decrypting transaction data failed.
IPAY0100256 - Payment encryption failed.
IPAY0100111 - Card decryption failed.
IPAY0100261 - Payment hashing failed.
IPAY0100178 - Invalid input data received.
IPAY0200014 - Problem occured during merchant response.
IPAY0100052 - Problem occured during merchant response encryption.
IPAY0100035 - Problem occured during merchant hashing process.
IPAY0100259 - Problem occured during merchant hashing process.
IPAY0100253 - Problem occured while cancelling the transaction.
IPAY0100252 - Missing veres.
IPAY0100251 - Invalid payment data.
IPAY0100204 - Missing payment details.
IPAY0100250 - Payment details verification failed.
IPAY0100249 - Merchant response url is down.
IPAY0100088 - Empty mobile number.
IPAY0100089 - Invalid mobile number.

IPAY0100090 - Empty MMID.
IPAY0100091 - Invalid MMID.
IPAY0100092 - Empty OTP number.
IPAY0100093 - Invalid OTP number.
IPAY0100272 - Problem occured while validating xml message format.
IPAY0100273 - Problem occured while validation VERES message format
IPAY0100274 - VERES message format is invalid
IPAY0100248 - Problem occured while validating PARES message format
IPAY0100247 - PARES message format is invalid
IPAY0100283 - Problem occured in determine payment instrument
IPAY0100109 - Invalid subsequent transaction, payment id is null or empty.
IPAY0100110 - Invalid subsequent transaction, Tran Ref id is null or empty.
IPAY0100284 - Invalid subsequent transaction, track id is null or empty.
IPAY0100114 - Duplicate Record
IPAY0100057 - Transaction denied due to invalid processing option action code
IPAY0100115 - Transaction denied due to missing original transaction id.
IPAY0100116 - Transaction denied due to invalid original transaction id.
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.
IPAY0100117 - Transaction denied due to missing card number.
IPAY0100118 - Transaction denied due to card number length error
IPAY0100119 - Transaction denied due to invalid card number
IPAY0100071 - Transaction denied due to missing CVD2.
IPAY0100072 - Transaction denied due to invalid CVD2.
IPAY0100086 - Transaction denied due to missing CVV.
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.
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.
IPAY0100063 - Transaction denied due to invalid trackID

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.
IPAY0100082 - Card address is not present
IPAY0100083 - Card postal code is not present
IPAY0100084 - AVS Check : Fail
IPAY0100087 - Card pin number is not present
IPAY0100085 - Electronic Commerce Indicator is invalid
IPAY0100080 - Transaction denied due to invalid expiration date.
IPAY0100081 - Card holder name is not present
IPAY0200027 - Missing encrypted card number.
IPAY0100112 - Problem occurred in method loading original transaction data(card number, exp month / year) for orig_tran_id
IPAY0100124 - Problem occured while validating transaction data
IPAY0100094 - Sorry, this instrument is not handled
IPAY0100285 - Transaction denied due to invalid original transaction
IPAY0100127 - Problem occured 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
IPAY0200028 - Problem occurred while loading default institution configuration (Validate Original Transaction)
IPAY0100108 - Perform risk check : Failed
IPAY0100101 - Denied by risk : Risk Profile does not exist
IPAY0200019 - Problem occurred while getting risk profile details
IPAY0100200 - Denied by risk : Negative BIN check - Fail
IPAY0100191 - Denied by risk : Negative Card check - Fail
IPAY0100201 - Denied by risk : Declined Card check - Fail
IPAY0100102 - Denied by risk : Maximum Floor Limit Check - Fail
IPAY0100198 - Transaction denied due to Risk : Transaction count limit exceeded for the IP
IPAY0100246 - Problem occurred while doing perform ip risk check
IPAY0200040 - Problem occurred while performing card risk check
IPAY0200021 - Problem occurred while performing risk check
IPAY0200020 - Problem occurred while performing transaction risk check

IPAY0100103 - Transaction denied due to Risk : Maximum transaction count
IPAY0100197 - Transaction denied due to Risk : Maximum debit amount
IPAY0100190 - Transaction denied due to Risk : Maximum floor limit transaction count
IPAY0100289 - Transaction denied due to Risk : Maximum credit amount
IPAY0100104 - Transaction denied due to Risk : Maximum processing amount
IPAY0100196 - Transaction denied due to Risk : Maximum processing amount
IPAY0100195 - Transaction denied due to Risk : Maximum credit processing amount
IPAY0100194 - Transaction denied due to Risk : Minimum Transaction Amount processing.
IPAY0100144 - ISO MSG is null. See log for more details!
IPAY0100245 - Problem occurred while sending/receivinig ISO message
IPAY0200034 - Problem occurred while getting details from VPASLOG table for payment id : null
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
IPAY0100043 - IP address is blocked already
IPAY0200068 -Problem occured while validating IP address blocking
IPAY0200051 - Problem occurred while updating risk in declined card table
IPAY0200052 - Problem occurred while updating risk factor
IPAY0100143 - Transaction action is null
IPAY0100286 - Unknown IMPS Tran Action Code encountered
IPAY0100097 - IMPS for Terminal Not Active for Transaction request, Terminal
IPAY0100287 - Terminal Action not enabled for Transaction request, Terminal
IPAY0100288 - Terminal Payment Instrument not enabled for Transaction request, Terminal
IPAY0100096 - IMPS for Institution Not Active for Transaction request, Institution
IPAY0100164 - Transaction Not Processed due to Invalid ECI value
IPAY0100165 - Transaction Not Processed due to Empty ECI value
IPAY0100167 - Transaction Not Processed due to Invalid Authentication Status
IPAY0100166 - Transaction Not Processed due to Empty Authentication 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
IPAY0100168 - Transaction Not Processed due to Empty Enrollment Status
IPAY0100187 - Customer ID is missing for Faster Checkout
IPAY0100188 - Transaction Mode(FC) is missing for Faster Checkout
IPAY0200039 - Problem occured while getting Faster Checkout details
IPAY0100192 - Transaction Not Processed due to Empty Xid

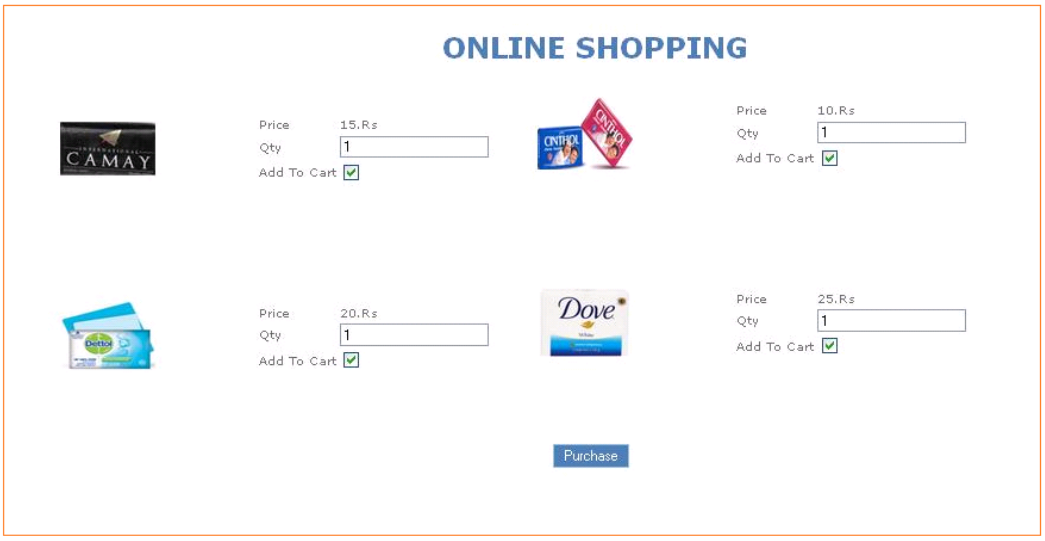
IPAY0100193 - Transaction Not Processed due to Invalid Xid
IPAY0100189 - Transaction denied due to brand directory unavailable
IPAY0100048 - CANCELLED
IPAY0100049 - Transaction Declined Due To Exceeding OTP Attempts
IPAY0200030 - No external connection details for extr conn id :
IPAY0200031 - Alternate external connection details not found for the alt extr conn id :
IPAY0200032 - Problem occurred while getting external connection details for extr conn id :
IPAY0100145 - Problem occurred while loading default messages in ISO Formatter
IPAY0100146 - Problem occurred while encrypting PIN
IPAY0100147 - Problem occurred while formatting purchase request in B24 ISO Message Formatter
IPAY0100148 - Problem occurred while hashing ecom pin.
IPAY0100149 - Invalid PIN Type
IPAY0100150 - Problem occurred while formatting Reverse purchase request in B24 ISO Message Formatter
IPAY0100151 - Problem occurred while formatting Credit 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
IPAY0100154 - Problem occurred while formatting Reverse Credit 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
IPAY0200037 - Error Occured while getting Merchant ID
IPAY0100183 - Error Occured Due to bytePAReq is null
IPAY0100184 - Error Occured while Parsing PAReq
IPAY0100158 - Host timeout
IPAY0100159 - External message system error
IPAY0100241 - Problem occurred while formatting purchase request in VISA ISO Message Formatter
IPAY0100240 - Problem occurred while formatting Credit request in VISA ISO Message Formatter
IPAY0100239 - Problem occurred while formatting authorization request in VISA ISO Message Formatter
IPAY0100238 - Problem occurred while formatting Capture request in VISA ISO Message Formatter
IPAY0100237 - Problem occurred while formatting Reverse purchase request in VISA ISO Message Formatter
IPAY0100236 - Problem occurred while formatting Reverse Credit request in VISA ISO Message Formatter

IPAY0100235 - Problem occurred while formatting reverse authorization request in VISA ISO Message Formatter
IPAY0100234 - Problem occurred while formatting Reverse Capture request in VISA ISO Message Formatter
IPAY0100271 - Problem occurred while formatting purchase request in MASTER ISO Message Formatter
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
IPAY0200053 - Problem occured while updating payment log currency details.
IPAY0200054 - Problem occured while inserting currency conversion currency details.
IPAY0200055 - Problem occured while updating currency conversion currency details.
IPAY0100281 - Transaction Denied due to missing Master Brand
IPAY0100282 - Transaction Denied due to missing Visa Brand
IPAY0100293 - Transaction denied due to duplicate Merchant trackid
IPAY0100294 - Transaction denied due to missing Merchant trackid
IPAY0200073 - Country Code not available for the Card.
IPAY0200074 - Restricted Country Code for the Transaction.
IPAY0200075 - Problem occured while getting Original transaction log details.
IPAY0100211 - Problem occured during EnStage process.
IPAY0100267 - Enstage Response status not sucessfull.
IPAY0100265 - Enstage Response validation failed.
IPAY0100072 - Transaction denied due to invalid CVD2 for rupay card.
IPAY0100265 - enstage response validation failed.
IPAY0100205 - Problem occurred while getting enstage response details.

Sample Demo Page Navigation

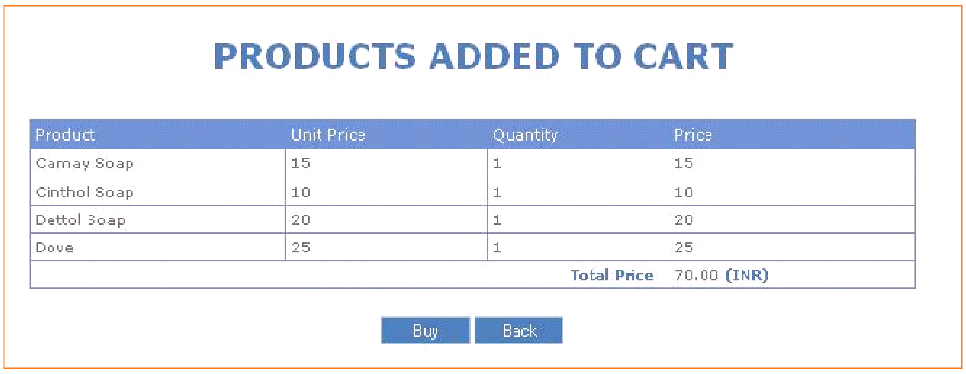
Step 1: Merchant's Product page

When the Customer log on to the Merchant website, Merchant will show the products page for online shopping.



Step2: Products selected by the customer for purchase:

If the customer clicks "Purchase", then the products added to cart will be shown.



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
		Total Price	70.00 (INR)

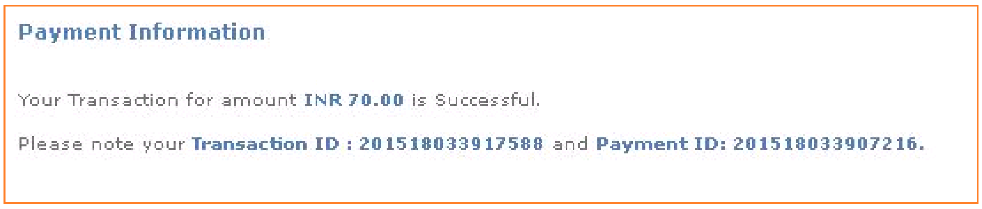
Step3: Payment Page:

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

//Payment page for Credit, Debit, Net Banking details

Step4: Merchant Response Page:

After processing transaction, Payment Gateway shall send 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.**

Best Practices:

a) The Merchant should mandatory maintain logs for each transaction as mentioned below

a. The parameters before setting the values in the respective variable.

b. Request from the merchant server to Payment Gateway

c. Response that is received from the Payment Gateway in the Merchant Response URL

b) 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)

c) 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 singed 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 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 consent is Mandatory.

f) 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 have to 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.

h) 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 in order to trace transaction history in case of a dispute raised by a customer or even internal audit purposes. These logs should ideally include the customer IP address as well apart from the other transaction details.