

Appnit Technologies

MERCHANT INTEGRATION DOCUMENT





Revision History:

Version Number	Modification Made	Modified By	Date
4.0.0	-----	Santosh Kumar	15-Feb-16
4.0.1	Integration of android pgSDK for eclipse	Ajeet Maurya	26-Feb-16
4.0.1	Reponse validation	Sandeep Prajaati	8-Apr-16



Last Modified Date: 20-Oct-16



Table of Contents

Introduction

Merchant Integration Process

Request Parameters

Encryption/Decryption Logic

Posting URLs for staging and Live environment

Response Parameters

Response Validation

Parameters Explained

Error Codes

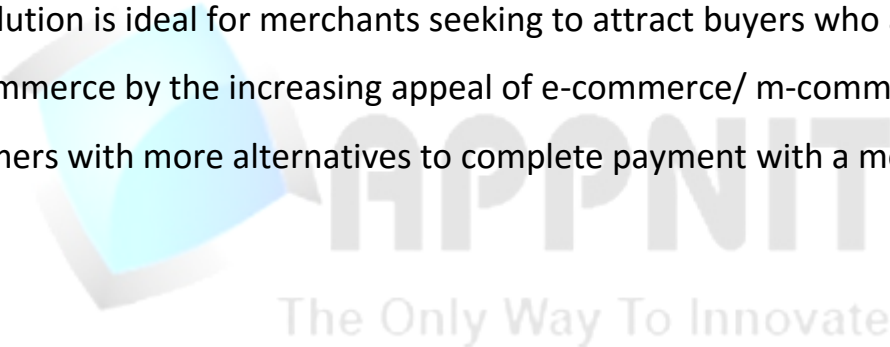
Important Instructions



Introduction

PayPlutus™ is a unique payment gateway service from Appnit Technologies in the burgeoning e-commerce space of India. It offers merchants with cost-effective solutions, wide range of payment acceptance modes and a secure technology platform. It also offers merchants an option of accepting payment using Credit & Debit cards, Internet banking and mobile payment options.

This payment solution is ideal for merchants seeking to attract buyers who are keen about banking on e-commerce by the increasing appeal of e-commerce/ m-commerce and providing customers with more alternatives to complete payment with a mere click.





PayPlutus™ - Merchant Integration Process

PayPlutus™ system maintains two environments viz.:

1. Staging (Testing) Environment
2. Live (Production) Environment

After signing of agreement between PayPlutus™ and merchant, we release staging environment details for the merchant to integrate PayPlutus™ Payment Gateway on their website / mobile site / mobile app.

Merchant integrates their system with PayPlutus™ staging environment, primarily checking for the following:

Whether parameters are getting posted on to PayPlutus™ system or not (customer landing on bank login page)

Whether merchant is getting response of either a successful or failure transaction back on its site (success/failure response page of the merchant)

After testing the integration successfully on PayPlutus™ staging environment, the merchant should contact PayPlutus™ representative for releasing live environment credentials. Only after the merchant successfully integrates its website on PayPlutus™ live environment, can they start accepting payments using PayPlutus™ Payment gateway on its website.



For integration, PayPlutus™ requires certain set of parameters to be passed on its posting URL, the details of which are explained below.

While passing transaction response back onto the merchants' site (for either success or failure transactions), PayPlutus™ system passes set of parameters back to merchant system with PayPlutus™ Reference ID mapped to them.

PayPlutus™ integration allows a merchant to either pre-populate customer's details on PayPlutus™ Billing and Shipping page or a customer has to fill-up these details on PayPlutus™ website and select a pay mode (bank/credit card/debit card gateway) to process the transaction.

For transaction related queries, the merchant is required to quote PayPlutus™ Reference ID (ATRN) in their correspondence with PayPlutus.





Payment Gateway integration for Web:

Request Parameters

The script for encryption / decryption logic of transaction parameters are mentioned below:

Here is the list of transaction parameters, which you should pass in an encoded format to PayPlutus™ for payment processing:

S.No	Parameters	Data Sample	Data Type
1	Merchant ID (MID)	12345678901234	Number 14
2	Application Name	XYZ	Varchar 20
3	Country	IND	Varchar 3
4	Currency	INR	Varchar 3
5	Amount	100.00	Number 5,2
6	Merchant Order no.	XXXXXX	Varchar 20
7	Callbackurl	XXXXXX	Varchar 200
8	Other Details	XXXXXX	Varchar 100
9	Application User	XYZ	Varchar 20

Request parameter and Billing-Shipping Details:

requestparameter = MID=%MID%+"|" + Appname=% Application Name

%+"|" +Country=%Country%+"|" +Currency=%Currency%+"|" +Amount=%Amount%+"|" +Tra



nsID=% Merchant Order no%+"|" + CallbackURL=% CallbackURL%+"|" + Appuser=% Appuser
%+"|" + Other=%Other Details%

Below is the list of parameters, which a merchant can post as Billing and Shipping details:

No	Parameter Name	Parameter description and its value	Max Char length
1	custName	Customer Name	50
2	custAddress	Customer Address	200
3	custCity	Customer City	50
4	custState	Customer State	50
5	custPinCode	Customer Postal Code	6
6	custCountry	Customer Country (2 Character Iso Country Code)	2
7	custPhoneNo1	Country Code for The Customer Phone Number	3
8	custPhoneNo2	Area Code for The Customer Phone Number	8
9	custPhoneNo3	Customer Phone Number	9
10	custMobileNo	Customer Mobile Number	10
11	custEmailId	Customer Email Id	100
12	otherNotes	Other details	200
13	deliveryName	Name of The Person to Whom Delivery / Shipping Is Being Made	50
14	deliveryAddress	Delivery / Shipping Address	200
15	deliveryCity	Delivery / Shipping City	50
16	deliveryState	Delivery / Shipping State	50
17	deliveryPinCode	Delivery / Shipping Postal Code	6



18	deliveryCountry	Delivery / Shipping Country (2 Character Iso Country Code)	2
19	deliveryPhNo1	Country Code for The Phone Number of the Person to Whom Delivery / Shipping Is Being Made	3
20	deliveryPhNo2	Area Code for The Phone Number of the Person to Whom Delivery / Shipping Is Being Made	8
21	deliveryPhNo3	Phone Number of the Person to Whom Delivery / Shipping Is Being Made	9
22	deliveryMobileNo	Mobile Number of the Person to Whom Delivery / Shipping Is Being Made	10

billingDtls=custName+"|"+custAddress+"|"+custCity+"|"+custState+"|"+custPinCode+"|"+custCountry+"|"+custPhoneNo1+"|"+custPhoneNo2+"|"+custPhoneNo3+"|"+custMobileNo+"|"+custEmailId+"|"+otherNotes;

shippingDtls=deliveryName+"|"+deliveryAddress+"|"+deliveryCity+"|"+deliveryState+"|"+deliveryPinCode+"|"+deliveryCountry+"|"+deliveryPhNo1+"|"+deliveryPhNo2+"|"+deliveryPhNo3+"|"+deliveryMobileNo;



PayPlutus™ will provide encryption kit with separate key for staging and production environment.

We request you to please mail @ pgsupport@appnittech.com and take the encryption key for staging and production environment

Encryption/Decryption Logic

Include API appnit_aes<jdk version>.jar provided by PayPlutus™ with your classpath.

Import package net.pg.appnit.utility;

Windows DLL Library

Include DLL library provided by PayPlutus™

Call the below mentioned script to encrypt requestparameter, billingDtls and shippingDtls

requestparameter = AES128Bit.encrypt(requestparameter, Key);

billingDtls = AES128Bit.encrypt(billingDtls, Key);

shippingDtls = AES128Bit.encrypt(shippingDtls, Key);

requestparameter = requestparameter.replaceAll("\n", "");

billingDtls = billingDtls.replaceAll("\n", "");

shippingDtls = shippingDtls.replaceAll("\n", "");

Sample:

*requestparameter =MID=12345678901234|Appname=abcFURNITURE|Country
=IND|Currency=INR|Amount=10|TransID=test-order|callBackURL=
www.url.com/callbackurl.jsp|Others=others|*



BillingDtls

*=TestUser/Mumbai/Mumbai/Maharashtra/400001/IN/91/022/28000000/9820000000/
testuser@gmail.com /test transaction for PayPlutus;*

ShippingDtls

=TestUser/Mumbai/Mumbai/Maharashtra/400234/IN/91/022/28000000/9920000000;

Insert the following code within the <body> tag of your Checkout / Pay Now page

```
<form name="ecom" method="post"
action="Posting URL for Staging or LIVE Environment">
<input type="hidden" name="requestparameter" value="<%= requestparameter %>">
<input type="hidden" name="billingDtls" value="<%= billingDtls %>">
<input type="hidden" name="shippingDtls" value="<%= shippingDtls %>">
<input type="hidden" name="MID" value="200904281000001"/>
<input type="submit" name="submit" value="Submit"> </form>
```

Encryption/Decryption in PHP

Use the below function for Encryption/Decryption in php

iv-: fedcba9876543210

key: -Key using for Encryption/Decryption

Encryption:

function encryptString(\$unencryptedText, \$key, \$iv)

Decryption:

function decryptString(\$encryptedText, \$key, \$iv)

Call the below mentioned script to encrypt requestparameter, billingDtls and shippingDtls.



```
<?php
$iv = 'fedcba9876543210';
// encryptString() Function use for encrypt the String
// unencryptedText :- Un encrypt String
// key : Encryption Key provided by Appnit technologies
function encryptString($unencryptedText, $key, $iv) {
    $passphrase = base64_decode($key);
    $unencryptedText=pkcs5_pad($unencryptedText);
    $enc = mcrypt_encrypt(MCRYPT_RIJNDAEL_128, $passphrase, $unencryptedText,
    MCRYPT_MODE_CBC, $iv);
    return base64_encode($enc);
}
// decryptString() Function use for decrypt the encrypt string
// unencryptedText :- encrypt String
// key : Encryption Key provided by Appnit technologies
function decryptString($encryptedText, $key, $iv) {
    $passphrase = base64_decode($key);
    $dec = mcrypt_decrypt(MCRYPT_RIJNDAEL_128, $passphrase,
    base64_decode($encryptedText), MCRYPT_MODE_CBC, $iv);
    return $dec;
}
function pkcs5_pad ($text) {
    $blocksize = 16;
    $pad = $blocksize - (strlen($text) % $blocksize);
    return $text . str_repeat(chr($pad), $pad);
}
?>
```



Posting URLs for staging and Live environment

The posting URLs for integration are as follows:

For PayPlutus™ staging (testing) environment:

<http://testpg.payplutus.in:8080/AppnitPG/WebPaymentRequest.action>

For PayPlutus™ live (production) environment:

<https://secure.payplutus.in/AppnitPG/WebPaymentRequest.action>

Response Parameters

S.No	Parameters	Data Sample	Data Type
1	Response Code	3001	Number 4
2	Status	Success/Failure	Varchar 100
3	Plutus reference ID	18823022222	Number max 20
4	Merchant Order ID	XYZ12121	Varchar 14
5	Amount	100.00	Number 5,2
6	Application name	XYZ	Varchar 20

Amount, merchant order number and other details remain same as given by the merchant at the time of posting them to PayPlutus™ system.

responseparams= SUCCESS=%status%|responseCode= %Responsecode%|plutusTxnId
=%PayPlutusreferenceid%|appTransId =%merchantorderno%|appUserId
=%applicationUserID%|amount =%amount%|appName %Applicationname%



Values will be passed back to the merchant's site in a two variable 'responseparams' as pipe separated values and 'hash' of 'responseparams' with secure key.

Sample transaction:

```
status=SUCCESS|responseCode=01|plutusTxnId=11111|appTransId=xyz123|appUserId=am  
it|amount=100.00|appName=amitPG
```

Response Validation

The Secure Hash is a hexadecimal encoded SHA-256 HMAC of a concatenation of Secure key.

Sample hash Calculation:

Make String concatenate with same Secure key share for encryption.

Example:

```
status=SUCCESS|responseCode=01|plutusTxnId=11111|appTransId=xyz123|appUserId=am  
it|amount=100.00|appName=amitPG|<Secure key share for encryption>
```

The purpose of the SHA2signaturefield is to ensure the integrity of the data posted back to your server. You should always compare the SHA2signaturefield's value posted by PayPlutus servers with the one you calculated.

To calculate the SHA2sig, you need to take the values of the fields listed above exactly as they were posted back to you, concatenate them and perform a SHA2 calculation on this string.



Payment Gateway integration for Android:

Integration (For Android Studio):

1. Import plutuspg-release.aar as a module.
2. Make sure that app has a module dependency on plutuspg-release.
3. Require minSdkVersion 11.
4. Auto enable to read OTP and approve on PG page.

Integration (For Eclipse):

1. Add pgsdk.jar in your libs folder.
2. Add SDK manifest attributes in your manifest.
3. Add PlutusInt.getInstance(Activity.this, obj) ;.
4. Handle onActivityResult method.

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.RECEIVE_SMS" />
<uses-permission android:name="android.permission.READ_SMS" />
<activity
    android:name="com.appnit.plutus.pg.screen.PGScreen"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="adjustResize" />
<receiver
    android:name="com.appnit.plutus.pg.util.SmsReceiver"
    android:enabled="false"
    android:exported="true" >
    <intent-filter android:priority="500" >
        <action android:name="android.provider.Telephony.SMS_RECEIVED" />
    </intent-filter>
```



</receiver>

Request Generation:

1. Get all the required parameters from PayPlutus™.
2. Create an object of PayReqParams, and to create this object you need three more object.
 - i. Object of MerchantParams and its some attributes is requiring to initiate Payment Gateway.
 - ii. Object of ShippingDtls is mandatory.
 - iii. Object of BillingDtls is mandatory.

Put all the obtained parameters and object in its object of PayReqParams.

Example:

```
PayReqParams obj = new PayReqParams();
MerchantParams params = new MerchantParams();
BillingDtls billingDtls = new BillingDtls();
ShippingDtls shippingDtls = new ShippingDtls();
params.setMerAmount("1");    //Mandatory
params.setMerMID("13");      // Mandatory(As per Credential provided)
params.setMerAppName("hd");// Mandatory(As per Credential provided)
params.setMerAppUser("test");//Mandatory(As per Credential provided)
params.setMerCBaseUrl("android");//Fixed 'android'
params.setMerCountry("IND"); //Fixed 'IND'
params.setMerCurrency("INR");//Fixed 'INR'
params.setMerOthersDtls("other");//Fixed 'other' or (As per Credential provided)
params.setMerKey("YVICSrOq+nZ5tdQoPUBMMo="); //Mandatory(As per Credential provided)
params.setMerTxnId(String.valueOf(System.currentTimeMillis())); //Mandatory(running variable)
params.setActionBarTitle("PaymentScreen"); //Mandatory( Action Bar Title)
//params.setIsProductionEnv(true);           //open for production .
obj.setMerchantParams(params);
```




```
billingDtls.setCustName("xyz");//optional
billingDtls.setCustAddress("XYZ");//optional
billingDtls.setCustCity("City");//optional
billingDtls.setCustCountry("IN");//optional
billingDtls.setCustEmailId("xyz@gmail.com");//mandatory
billingDtls.setCustMobileNo("9999999999");//mandatory
billingDtls.setCustPhoneNo1("12");//optional
billingDtls.setCustPhoneNo2("12");//optional
billingDtls.setCustPhoneNo3("12");//optional
billingDtls.setCustState("Uttar Pradesh");//optional
billingDtls.setCustPinCode("201301");//optional
billingDtls.setOtherNotes("nfdddd");//optional

shippingDtls.setDeliveryAddress("XYZ");//optional
shippingDtls.setDeliveryCity("City");//optional
shippingDtls.setDeliveryCountry("IN");//optional
shippingDtls.setDeliveryMobileNo("999999999999");//optional
shippingDtls.setDeliveryName("xyz");//optional
shippingDtls.setDeliveryPhNo1("22");//optional
shippingDtls.setDeliveryPhNo2("22");//optional
shippingDtls.setDeliveryPhNo3("22");//optional
shippingDtls.setDeliveryPinCode("201301");//optional
shippingDtls.setDeliveryState("Uttar Pradesh");//optional
```

```
params.setHaveBillingDtls(true);
obj.setBillingDtls(billingDtls);
params.setHaveShippingDtls(true);
obj.setShippingDtls(shippingDtls);
PlutusInt.getInstance(MainActivity.this, obj);
```

Note → ** Optional parameters can be left blank as “null”.

Note → ** throw null pointer exception if your given value match to any of them “null”, null, “ ” in Mandatory field.



- You don't need to set `setShippingDtls()` and `setBillingDtls()` in case you are not using them .and if you need then first set value of `haveBillingDtls` and `haveShippingDtls` true in object of `MerchantParams`.
- Start the PayPlutus activity by using below code on button or any click event and let PayPlutus SDK take care of the rest.

`PlutusInt.getInstance(activity, obj);`

- Plutus SDK internally start new activity using our activity reference and object of `PayReqParams`.
- After get response from Plutus server, SDK return value on Override `onActivityResult` method in your reference activity.

Response:

1. Once callback has been made by PayPlutus server, you get response Override `onActivityResult` method in your current activity .Add this line in your code.

@Override

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode== AppConstants.PG_REQ_CODE) {  
        Object result = null;  
        if (data != null && data.getExtras() != null && data.getSerializableExtra(AppConstants.PG_RESULT) != null) {  
            result = data.getSerializableExtra(AppConstants.PG_RESULT);  
            if (result != null && result instanceof PlutusResponse) {  
                PlutusResponse res = (PlutusResponse) result  
  
                /** Add Here Own Code for handle PG Response .Response code if other than 01 means some problem on PG, see  
status*/  
                } else if (result != null && result instanceof String) {  
                    String res = (String) result;  
                    /** Add Here Own Code for handle User Cancel PG from back or cancel txn.*/  
                }  
            }  
        }  
    }
```



```
super.onActivityResult(requestCode, resultCode, data);  
}
```

2. You get object of PlutusResponse response or String in response as mention above (WEB).

Parameters Explained:

MID: Merchant ID is generated by PayPlutus™ system. It is a unique identifier for PayPlutus™ system to identify the merchant and their transactions. The merchant should use the fixed MID: 200904281001 for integrating on the PayPlutus™ staging (testing) environment. After the merchant successfully tests the integration on PayPlutus™ staging (testing) environment, should get in touch with PayPlutus™ representative for release of live (production) MID.

Country: It is a mandatory field and merchant should only use IND as its value. This field is case sensitive.

Currency: It is a mandatory field and merchant should only use INR as its value. This field is case sensitive

Amount: It is a mandatory field and merchant can pass 7 digit amount values (excluding decimal) in 5.2 format.

Merchant Order No: The merchant is required to pass its unique order number for each request in this field. It is a mandatory field.



Other Details: It is a mandatory field and merchant can pass any desired value in this field. Merchant could use any customer narration in this field. If not, then “NULL” as a text value, should be passed.

CallbackURL: The protocol ‘http://’ or ‘https://’ is mandatory and should be prefixed in the URL. For example, https://www.merchantsuccesspage.com. It is a mandatory field.

PayPlutus Reference ID: This is a unique transaction reference number generated by PayPlutus™ system to identify transaction and it gets mapped to the order number and other values generated by the merchant. This value is passed on to the merchant system in response parameters.

Flag: Depending upon the response received by PayPlutus™ system from the bank, we pass on the transaction status response back to merchant site in response parameters

Response Codes:

01=User has successfully completed a transaction.

1301=Txn Failed.

1206=Merchant key not found/valid.

1207=Duplicate Transaction Id.

1001=Missing Application Name.



1002=Application Name is invalid.

1003=Missing Application User Name.

4001=Invalid/Null Payment Object.

4002=Amount should be greater than 0.

4003=plutusTxnIdentifier is invalid.

4006=Amount is invalid.

4008=Currency is invalid.

4009=Transaction Identifier is invalid.



Important Instructions:

The merchant might require installing www.payplutus.in public key certificate on their web server for SSL mode of data transfer.

Special characters are NOT allowed while posting transaction parameters on PayPlutus™ system. The indicative list is as follows:

~ (tilde)



" (double quote)

' (apostrophe)

& (ampersand)

() (parenthesis)

(hash)

% (percent)

