PayU
Integration document

**Vatika Towers, Ground Floor, Block A,
DLF Golf Course Road, Sector 54
Gurgaon, 122002
India
T: 0124-6749078
F: 0124-6749101**

Overview

This note describes the how to do the technical integration between PayU Payment Gateway and your website in respect of powering online transactions.

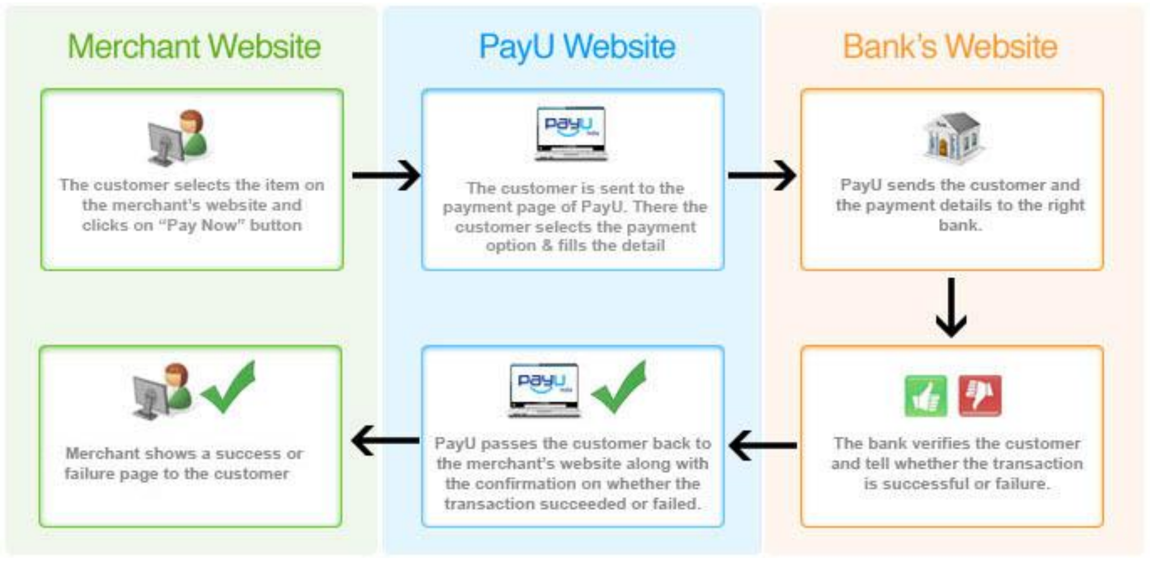
PayU Payment Gateway

PayU offers electronic payment service to your website through its various partnerships with banks and payment instrument companies. Through PayU, your clients would be able to make electronic payments through credit card, debit card and online net banking account

PayU also offers an online interface where the merchant can view transaction details, settlement reports, analytic reports etc. and also take action like capture, cancel, refund etc. This online interface can be accessed through https://www.PayU.in by using the username and password provided to you.

Payment Process Flow

The following diagram explains how the customer makes the payment and how the process flows:


Merchant Website
PayU Website
Bank's Website
The customer selects the item on the merchant's website and clicks on "Pay Now" button
The customer is sent to the payment page of PayU. There the customer selects the payment option & fills the detail
PayU sends the customer and the payment details to the right bank.
The bank verifies the customer and tell whether the transaction is successful or failure.
PayU passes the customer back to the merchant's website along with the confirmation on whether the transaction succeeded or failed.
Merchant shows a success or failure page to the customer

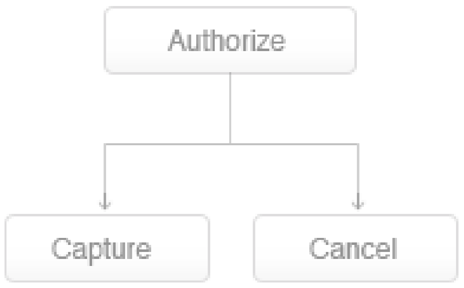
Step 1: The consumer selects the product on your website and clicks on "Pay Now" button.
Step 2: The consumer is then taken from your website to the transaction page of PayU.in where in all the payment related details are entered by the consumer.
Step 3: PayU.in redirects the consumer to Visa, Mastercard or the relevant bank for the next level of authorization.
Step 4: The Bank/Visa/Mastercard authorizes and confirms the transaction
Step 5: The consumer is sent back to PayU.
Step 6: PayU sends the consumer back to your website along with the transaction status.

Status of a Transaction

A transaction can have several different status as have been explained below.

Authorize

This is the step in which the customer (who is using credit/debit card for payment) is authorized by the bank and the money on the card is blocked for a maximum period of 3 days. Lets say, the customer has a credit card limit of Rs 50,000 and the transaction is for Rs 2000. When a transaction is authorized, Rs 2000, is blocked on the card for a period of 8 days. The customer, for the next 3 days, can only use the remaining limit i.e. (Rs 50,000 – Rs 2000 = Rs 48,000). This Rs 2000, which has been blocked, are now ready to be Captured by the merchant.



Capture

In this step, the money that has been blocked in the Authorize step (explained above), is captured i.e. transferred from the bank to the merchants account. The merchant has a period of upto 3 days to capture a transaction.

Cancel

An authorized transaction can be cancelled by the merchant. In this case the money that was blocked on the customer's card is freed up.

Refund

If for some reason, you have captured the amount and later want to refund it, then this is needed for that. By using the refund option, you can retrun the entire or part of the money of the user.

Chargeback

This is a scenario, where the customer disputes the transaction and wants the money back. The reason for dispute can be two:

1. Fraud transaction i.e. the customer did not make the transaction. In India, this risk is low since a password is required to make any transaction.
2. Non delivery of promised goods and service. If the merchant does not deliver the promised good or service then the user can raise a dispute and demand money back.

Settlement process

Settlement is the process by which the money gets transferred from the customer to the bank account of the merchant. PayU follows a T+2 settlement scheme where T is the date on which the transaction is captured.

There is a reconciliation process at PayU. On the next day, after you have captured the transactions, PayU will reconcile the online transactions with the credits received based on batch files received from the banks. After reconciling, we will generate a report and payment will be made for all the transactions for which payment has been received from the bank. All the details will be visible to you in the online interface.

Technical Integration

In the payment process flow, to move the consumer from Step 1 to Step 2, a POST request needs to be generated by merchant to the following URL

Production server:

POST URL: https://secure.payu.in/_payment.php

Test server:

POSL URL: https://test.payu.in/_payment.php

Key notes and terms

1. **Key (MerchantID)** : This ID is generated at the time of activation of your site and helps to uniquely identify you to PayU

2. **TxnID**: A Unique alphanumeric Transaction ID generated by you to uniquely identify a transaction. The TxnID should be unique since it would allow you to identify the transaction easily.

3. **Amount**: Amount is the total amount of the transaction(greater than 0) in INR, without a currency symbol or other non-numeric character. Only a decimal allowed.

4. **MIHPayID**: Unique ID generated for a transaction by PayU.in

5. **Hash (Checksum)**: This refers to a random numeric string generated using a mathematical algorithm to ensure that data is not tampered along the way. Lets say a message has to be sent from location X to Y. X and Y both mutually agree on a Secret Key called “Salt” that only both of them possess. A checksum is generated by a mathematical function using the message and the Salt as input. This checksum is then sent along with the message to Y. Y then recalculates this checksum using the Salt and the same algorithm. If the checksum that Y calculates is different from the checksum that X passed then the data was tampered along the way and is thus rejected.

6. The **Checksum algorithm** used is SHA2 which is globally well known alogirthm. To need help with implementation, feel free to call us, mail us or use Google to find the desired function library for your implementation. Some example codes are also mentioned at the end of this document

The parameters to post are described below

S.No.	Variable	Description
1	key (MerchantID) (compulsory)	Merchant ID provided by PayU

2	txnid (compulsory)	Transaction number that merchant uses to track order *(Must be unique every time if already successful, other wise you got an error of duplicate transaction.)*
3	amount (compulsory)	Payment amount
4	productinfo (compulsory)	Product Description
5	firstname (compulsory)	Self-Explanatory *(only alphabets a-z are allowed)*
6	Lastname	Self-Explanatory *(only alphabets a-z are allowed)*
7	address1	Self-Explanatory *(Length of address1 and address2 must not more than 100 characters each and the allowed characters are only) A TO Z, a to z, 0 to 9, @, - (Minus), _ (Underscore), / (Backslash), (Space), . (Dot)*
8	address2	self-explanatory *(allowed characters are same as for address1)*
9	City	self-explanatory *(allowed characters are same as in address)*
10	State	self-explanatory *(allowed characters are same as in address)*
11	Country	self-explanatory *(allowed characters are same as in address)*
12	Zipcode	self-explanatory *(numeric value only)*
13	email (compulsory)	self-explanatory
14	phone (compulsory)	mobile number or landline number *(numeric value only)*
15	udf1	user defined field 1
16	udf2	user defined field 2
17	udf3	user defined field 3
18	udf4	user defined field 4
19	udf5	user defined field 5
20	surl (compulsory)	Success URL where PayU.in will redirect after successful payment.
21	Furl (compulsory)	Failure URL where PayU.in will redirect after failed payment.
22	curl	Cancel URL where PayU.in will redirect when user cancel the transaction.
23	Hash (Checksum) (compulsory)	Hash or Checksum (*Formula for checksum =sha512(key|txnid|amount|productinfo|firstname|email|udf1|udf2|udf3|udf4|udf5|udf6|udf7|udf8|udf9|udf10|<SALT>) SALT will be provided by PayU.in)*
24	Pg	Which tab you want to be open default on PayU. (E.g.: NB = Net Banking & CC=Credit Card); Credit Card is recommended

Important Things to remember

- **Allowed characters** for address1, address2, city, state, country, productinfo, email, and phone are:
  1. Characters: A to Z, a to z, 0 to 9
  2. - (Minus)
  3. _ (Underscore)
  4. @ (At the Rate)
  5. / (Slash)

6. (Space)
7. . (Dot)

If the merchant sends any other special characters then they will be automatically removed. The address will consider only first 100 characters.

- **Formula for checksum before transaction**

  sha512 (key|txnid|amount|productinfo|firstname|email|udf1|udf2|udf3|udf4|udf5|udf6|udf7|udf8|udf9|udf10|<SALT>)

  SALT will be provided by PayU. The algorithm used is SHA2 which is globally well known alogirthm. To need help with implementation, feel free to call us, mail us or use Google to find the desired function library.

- **Formula for checksum after transaction**

  This time the variables are in reverse order and status variable added between salt and udf1

  sha512 (<SALT>|status|udf10|udf9|udf8|udf7|udf6|udf5|udf4|udf3|udf2|udf1|email|firstname|productinfo|amount|txnid|key)

  It is strongly recommended that the *hash* (or checksum) is computed again after the transaction and is compared with what we post in the return parameters below.

Return Parameters

Sno	**Variable**	**Description**
1	mihpayid:	Unique id from PayU.in
2	mode:	'CC' for credit-card / 'NB' for net-banking / 'CD' for cheque or DD / 'CO' for Cash Pickup
3	status:	SUCCESS/PENDING/FAILURE
4	key:	Merchant key provided by PAYU
5	txnid:	transaction id sent by merchant.
6	amount:	original amount send by merchant
7	discount:	This is discount given to user - based on promotion set by merchants.
8	Offer	description offer for what PayU given the offer to user - based on promotion set by merchants.
9	productinfo:	<Self Explanatory>
10	firstname:	<Self Explanatory>
11	lastname:	<Self Explanatory>
12	address1:	<Self Explanatory>
13	address2:	<Self Explanatory>
14	city:	<Self Explanatory>

15	state:	<Self Explanatory>
16	country:	<Self Explanatory>
17	zipcode:	<Self Explanatory>
18	email:	<Self Explanatory>
19	phone:	<Self Explanatory>
20	udf1:	<Self Explanatory>
21	udf2:	<Self Explanatory>
22	udf3:	<Self Explanatory>
23	udf4:	<Self Explanatory>
24	udf5:	<Self Explanatory>
25	hash:	Hash must be verified before confirmation of transaction
26	Error:	If transaction failed, then reason of failure

Web Services

Web services are available to help you with reconciliation of your transactions in real time. The web service is also a post request with the following format:

Web Service URL: https:/info.payu.in/merchant/postservice.php

Mandatory parameters:

Parameter	**Description**	**Sample Value**
Key	Merchant key provided by PayU.	ibibo
Command	Name of the command (or function) as described below. Currently two functions are available – Verify_payment and check_payment	Verify_payment
Hash	hash value generated as follows: sha512(key|command|var1|salt)	ajh84ba8abvav
var1	This value changes according to command. (read command explanations for that)	Abc
Var2, Var3 ... upto var15	This value changes according to command. (read command explanations for that)	Abc

Commands

There are currently two commands or functions that are available for the purpose of reconcialiation:

1. verify_payment: This command is used to reconcile the transaction with PayU. The return parameter are MIHPayID, Amount, Discount, Mode and Status of transaction.

Additional Variables description**:**

Parameter	**Description**	**Sample Value**

var1	Pipe separated transaction ids(txnid)	100123|100124|100125|100126

2. check_payment: This command is used to check payment after transaction. It return all the parameters for a given transaction.

Additional Variables description:

Parameter	**Description**	**Sample Value**
var1	Payu id (id) of transaction	8000123

The above web services can also be accesses through a form by calling the following Form URL
Form URL: https://info.payu.in/merchant/postservice.php?form=1

The Form URL can also be used to check the kind of output a web service returns so that the return string can be easily interpreted.

Checksum Algorithm Example codes

The **Checksum algorithm** used is SHA2 which is globally well known alogirthm. To need help with implementation, feel free to call us, mail us or use Google to find the desired function library for your implementation. Some example codes are also mentioned below:

For PHP

Example code:

```
 $output = hash("sha512", $text);
```

For .NET

Link: http://msdn.microsoft.com/en-us/library/system.security.cryptography.sha512.aspx
Example code:

```
byte[] data = new byte[DATA_SIZE];
byte[] result;
SHA512 shaM = new SHA512Managed();
result = shaM.ComputeHash(data);
```

For JSP

Example code:

```
import java.io.FileInputStream;
import java.security.MessageDigest;

public class SHACheckSumExample
{
    public static void main(String[] args)throws Exception
    {
        MessageDigest md = MessageDigest.getInstance("SHA-512");
        FileInputStream fis = new FileInputStream("c:\\loging.log");

        byte[] dataBytes = new byte[1024];

        int nread = 0;
        while ((nread = fis.read(dataBytes)) != -1) {
          md.update(dataBytes, 0, nread);
        };
        byte[] mdbytes = md.digest();

        //convert the byte to hex format method 1
        StringBuffer sb = new StringBuffer();
        for (int i = 0; i < mdbytes.length; i++) {
          sb.append(Integer.toString((mdbytes[i] & 0xff) + 0x100, 16).substring(1));
        }

        System.out.println("Hex format : " + sb.toString());

       //convert the byte to hex format method 2
        StringBuffer hexString = new StringBuffer();
                  for (int i=0;i<mdbytes.length;i++) {
                    hexString.append(Integer.toHexString(0xFF & mdbytes[i]));
                  }

                  System.out.println("Hex format : " + hexString.toString());
    }
}
```

Technical Contact Person in case of any queries

Gajinder Singh
[M] 99100 61896
[T] 0124-6749000
[Ext] 378
[E] Gajinder.Singh@PayU.in

Vatika Towers, Ground Floor, Block A,
DLF Golf Course Road, Sector 54
Gurgaon, 122002