.com

Paypay89 Payment System

Integration Guide

Version - 1.1.0

Confident - for professional use only

02 May 2023

Table of Contents

1. System Overview
    1.1. Security Requirements
2. Integration
    2.1. SOAP
        2.1.1.SOAP, Parameter Definitions
        2.1.2.SOAP, Credit Card, Peer-to-Peer(P2P)
            2.1.2.1.General information
            2.1.2.2.Verification
            2.1.2.3.Secure3D Integration
            2.1.2.4.Pre-Authentications (PreAuths)
            2.1.2.5.Settlements (Capture)
            2.1.2.6.Single transactions (Sale)
            2.1.2.7.Responses
        2.1.3.SOAP, ACH
            2.1.3.1.General Information
            2.1.3.2.Single Transactions
            2.1.3.3.ACH Account Types
            2.1.3.4.ACH Classes
            2.1.3.5.Responses
        2.1.4.SOAP, Card Load
            2.1.4.1.General Information
            2.1.4.2.Interface Location
            2.1.4.3.Load Transactions
            2.1.4.4.Responses
            2.1.4.5.Process Flow
        2.1.5.SOAP, Refunds
            2.1.5.1.General Information
            2.1.5.2.Refunding Transactions
            2.1.5.3.Responses
    2.2. REST
        2.2.1.Payout
        2.2.2.Deposit
    2.3. TxHandler
        2.3.1.General Information
            2.3.1.1.Interface Location
            2.3.1.2.Will I need to make changes to my integration if I am using Secure3D with TxHandler?
        2.3.2.Single Transactions
        2.3.3.Single Transactions - CUP/WeChat/AliPay
        2.3.4.Parameter Definitions
        2.3.5.Responses
            2.3.5.1.Decryption
        2.3.6.Single Transactions-P2P
        2.3.7.Flow diagram-P2P
        2.3.8.Parameter Definitions-P2P
        2.3.9.Responses-P2P
            2.3.9.1.Decryption
    2.4. Secure Hosted Payment
        2.4.1.General Information
            2.4.1.1.Interface Location
            2.4.1.2.Testing
            2.4.1.3.Will I need to make changes to my integration if I am using Secure Hosted Payment?
        2.4.2.Single Transactions
        2.4.3.Single Transactions - CUP/WeChat/AliPay/P2P
        2.4.4.Parameter Definitions
        2.4.5.Responses
            2.4.5.1.Email
            2.4.5.2.Postback URL
            2.4.5.3.Success URL
            2.4.5.4.Failure URL
    2.5. Postbacks
        2.5.1.General
        2.5.2.Usage
        2.5.3.Parameters
        2.5.4.Signature
        2.5.5.Usages of Parameter: **response**
    2.6. Admin Interfaces
        2.6.1.General Information
            2.6.1.1.Interface Location

2.6.2.Get Transactions By TID
2.6.3.Get Transactions By RID
2.6.4.Get Transactions By SID
2.6.5.Get Transaction Details
2.6.6.Parameter Definitions
2.6.7.Responses
3. Code Examples
3.1. SOAP Credit Card
3.1.1.Single Transaction [Non-Secure3D OR Secure3D] (php + nusoap)
3.1.2.Single Transaction [Non-Secure3D OR Secure3D] (PHP soap)
3.2. SOAP Peer-to-Peer(P2P)
3.2.1.Single Transaction-P2P(php + nusoap)
3.2.2.Single Transaction-P2P (PHP soap)
3.2.3.Payout Transaction-P2P (php + nusoap)
3.2.4.Single Transaction-P2P (PHP soap)
3.3. SOAP ACH
3.3.1.Single Transaction (php + nusoap)
3.4. SOAP Card Load
3.4.1.Single Card Load (php + nusoap)
3.5. SOAP Refunds
3.5.1.Refund (php)
3.6. Secure Hosted Payment
3.6.1.Single Transaction-P2P(HTML)
3.6.2.P2P Deposit - Server To Server
3.6.2.1.Environments
3.6.2.2.Request Details and Flow
3.6.3.P2P Crypto Currency Deposit - Server To Server
3.6.3.1.Environments
3.6.3.2.Request Details and Flow
3.6.4.P2P Payout - Server To Server
3.6.4.1.Environments
3.6.4.2.Request Details and Flow
3.6.5.Single Transaction (HTML)
3.7. TxHandler
3.7.1.Single Transaction (HTML)
3.7.2.Single Transaction-P2P (HTML)
3.8. Postbacks
3.8.1.PAYMENT
3.8.2.REFUND
3.8.3.CHARGEBACK
3.8.4.LOAD
4. Appendix
4.1. Paypay89 Generic Error Types
4.2. Transaction Action Types
4.3. Transaction Status Types
4.4. Transaction Specific Error Codes
4.5. Bank Codes
4.6. CVV(2) Response Codes
4.7. AVS Response Codes (Visa)
4.8. AVS Response Codes (Mastercard)
4.9. Country ISO Codes
4.10. State ISO Codes
4.11. Industry Standard Credit Card Response Codes
4.12. Payout Bank Codes
4.13. Payment/Deposit Bank Codes

1. System Overview

Paypay89 is a payment system that allows you to do quick, efficient and secure online transaction processing through a number of third party and bank payments processing systems. Paypay89 uses real time processing with customer specified payment gateways to verify credit card details. Paypay89 stores the purchase and verification details in an internal database.

With the Paypay89 Administration interface you are able to overview user transactions and marketing figures. It also allows you to cancel existing accounts and to view and analyse declined payments.

1.1. Security Requirements

Important: To comply with the PCI Security Standards Council policy, PCI-DSS, all integrations are required to be TLS 1.2 or higher.

SSL and TLS 1.1 has been removed as an example of strong cryptography in the PCI DSS, and can no longer be used as a security control.

2. Integration

2.1. SOAP

2.1.1.SOAP, Parameter Definitions

Important: All parameters must be sent with the SOAP request, but only required parameters must have a value. All other values can be sent as an empty String.

Name	Data Type	Required	Description
sid	Integer	✓	Site id (unique identifier for the web site)
rcode	String	✓	Retailer authentication code
amount	Decimal	✗	Transaction amount
card_types	String	✗	CSV of card types
description	String	✗	What a recurring billing is for. e.g. "Monthly Subscription"
filter_search	String	✗	Filtration value for the filtration type
filter_type	String	✗	Filtration type for the filtration value
from_timestamp	Integer	✗	Timestamp as search from date
limit	Integer	✗	Maximum number of results to return
parent_txid	String	✗	Registration TXID for a particular transaction
postbackurl	String	✗	Postback URL
reason	String	✗	Reason for refund, etc...
responses	String	✗	Responses
sendNotification	String	✗	Whether to send an email notification of transaction
to_timestamp	Integer	✗	Timestamp as search to date
tx_actions	String	✗	List of tx_actions
txid	String	✗	A txid
udetails	Array	✓	Contains user details
username	String	✗	Legacy field - not used
password	String	✗	Legacy field - not used
firstname	String	✓	First name of the customer
lastname	String	✓	Last name of the customer
email	String	✓	Customer email address
phone	String	✓	Customer phone number
mobile	String	✗	Customer mobile number
address	String	✓	Customer address
suburb_city	String	✓	Customer suburb or city
state	String	✓	Customer state, 2 or 3 letter code for US/Canada/Australia Appendix - State ISO Codes
postcode	String	✓	Customer postcode/zipcode
country	String	✓	Customer country, ISO 3166 2 letter code Appendix - Country ISO Codes
ship_firstname	String	✗	Shipping first name
ship_lastname	String	✗	Shipping last name
ship_address	String	✗	Shipping address
ship_suburb_city	String	✗	Shipping suburb or city
ship_state	String	✗	Shipping state, 2 or 3 letter code for US/Canada/Australia Appendix - State ISO Codes
ship_postcode	String	✗	Shipping postcode/zipcode
ship_country	String	✗	Shipping country, ISO 3166 2 letter code Appendix - Country ISO Codes
ship_address	String	✗	Shipping address
bank_name	String	✗	Card issuer name
bank_phone	String	✗	Card issuer phone number
ssn	String	✗	Social security number
dl	String	✗	Drivers license number
dob	String	✗	Date of birth - Format: YYYY-MM-DD, *(required on some integrations)
uip	String	✓	IP address of the purchasing person. If taking phone orders please set uip to 127.0.0.1. This is only valid for phone orders.
paydetails	Array	✓	Contains payment details
payby	String	✓	Card Type (eg. visa, mastercard, ach, etc...). You will be notified if you have a special requirement here.
card_name	String	✗	Card holder name
card_no	String (16)	✗	No spaces or dashes allowed

card_ccv	String (3)	✗	Card security code (CVV2) 3 or 4 digits
card_exp_month	String (2)	✗	Card expiry month - 2 digits
card_exp_year	String (4)	✗	Card expiry year - 4 digits
md	String	✗	Session ID - Required when performing Secure3D transactions
redirecturl	String	✗	Where to redirect users after Secure3D password entry - Required when performing Secure3D transactions
useragent	String	✗	Browser user agent string - Required when performing Secure3D transactions
browseragent	String	✗	Browser agent string - Required when performing Secure3D transactions
routing_no	String	✗	ACH routing number. Only required for ACH/Check21 transactions.
account_no	String	✗	ACH account number. Only required for ACH/Check21 transactions.
type	String	✗	ACH account type. Valid values are (1 = Checking / 2: Savings). Only required for ACH/Check21 transactions.
regulation_e	String	✗	ACH regulation e acceptance. Only required for ACH/Check21 transactions. - Set to 1 if unsure.
class	String	✗	ACH origination. Please set to "WEB" if online payment, or "TEL" if order taken by phone. Only required for ACH/Check21 transactions.
receive_name	String	✗	ACH receivers name. Only required for ACH/Check21 transactions.
schedule_id	String	✗	Recurring billing schedule identifier.
payoutdetails	Array	✗	Contains recipient details for payout
payby	String	✓	`p2punion_pay`
amount	String	✓	Payout amount
currency	String	✓	Currency code - used for validation only
card_name	String	✗	Card holder name **union_pay only*
card_no	String	✗	No spaces or dashes allowed **union_pay only*
account_name	String	✗	Customer's bank account name **p2p only*
account_number	String	✗	Customer's bank account number **p2p only*
bank_province	String	✗	The province of account opening branch **p2p only(Bank city it is mandatory for P2P SEA Payout MYR , THB, VND and IDR)*
bank_city	String	✗	The city of account opening branch **p2p only(Bank city it is mandatory for P2P SEA Payout MYR , THB, VND and IDR)*
bank_branch	String	✗	Address of the account opening branch **p2p only(Bank city it is mandatory for P2P SEA Payout MYR , THB, VND and IDR)*
bank_code	String	✗	Bank code. Appendix - Payout Bank Codes **p2p only(Bank city it is mandatory for P2P SEA Payout MYR , THB, VND and IDR)*
cart	Array	✓	Information about the purchase items
summary	Array	✓	Contains a summary of cart contents
quantity	Integer	✓	Total quantity of cart items
amount_purchase	String	✓	Total purchase amount (without commas and only 2 decimals places)
amount_shipping	String	✓	Total shipping amount (without commas and only 2 decimals places)
currency_code	String	✓	3 digit currency code (Only "USD" supported at this time)
items	Array	✓	Contains the cart items
(numerical index)	Array	✓	Contains information about one item in the cart **(Repeat until all cart items are listed)**
name	String	✓	Category, can be used freely
quantity	Integer	✓	Quantity of the item
amount_unit	String	✓	Unit price amount (without commas and only 2 decimals places)
item_no	String	✓	Article number
item_desc	String	✓	Item description (**NOT TO USE GENERIC ITEM DESCRIPTION**)
txparams	Array	✓	Additional parameters
wid	String	✗	Webmaster ID
tid	String	✓	Tracking ID, its value must be unique and a maximum of 40 alphanumeric characters are allowed in this field. And unique tid value can be generated through concatenation of marchant name or curreny or any string, sid and currenttimestamp with micro seconds.
ref1	String	✗	Used for your own reference
ref2	String	✗	Used for your own reference
ref3	String	✗	Used for your own reference
ref4	String	✗	Used for your own reference
addinfo	String	✗	Additional information, in most cases blank

cmd	String	✗	Reserved
postbackurl	String	✗	This is where our system will send a server-server postback notification.
successurl	String	✗	Used in secure payments as a redirection URL for successful payments
failureurl	String	✗	Used in secure payments as a redirection URL for failed payments

2.1.2.SOAP, Credit Card, Peer-to-Peer(P2P)

2.1.2.1.General information

2.1.2.1.0. Interface location

The Paypay89 payment system uses SOAP to communicate with customer sales systems. It is important that the required parameters passed to the server are formatted with the correct data type, (see Error! Reference source not found.) otherwise the transaction will not be accepted.

The SOAP service is located at: https://admin.paypay89.com/soap/tx3.php?wsdl

2.1.2.1.1. Testing

To test your integration, please contact support and you will be provided with a testing SID. Any transaction processed on the testing SID will only be checked syntactically and semantically by Paypay89. This way you can test if you have sent sufficient and correct data.

When performing testing with your testing SID, the following card details can be used.

Field	Value
Card Number	4111111111111111 or 5555555555554444
Card Type	visa or mastercard
CCV	123
Expiry Date	Any future month/year
Card name	Tester

The following CVV value can be used to test different transaction results with your testing SID.

CVV	Description
222	Decline
333	Gateway Error
411	Secure3D Approved Result (Secure3D tests only)
422	Secure3D Declined Result (Secure3D tests only)
All other values	Approved transaction

You will be able to view the results of your testing transactions in Paypay89 by searching for them on the Browse Transactions page with your testing SID as a search item.

2.1.2.2.Verification

2.1.2.2.0. What is Verification?

Verification allows you to verify customer submitted details, like email or mobile phone, by asking the user to submit extra confirmation details that may be sent to them.

Verification integration will be required only for sites that have verification options enabled in the fraud check settings.

2.1.2.2.1. How is Verification required?

A transaction that requires verification will return a **VER** response. If this repose is recieved you will also get the extra question required for verification as encoded html. Here some example code:

PHP

```
<?php
if($response["status"]=="VER"){
        echo '<form action="yoursubmitpage.php" method="POST">';
        echo urldecode($response["required"]["embedhtml"]);
        echo '<br/>';
        echo '<input type="submit"/> </form>';
}
?>
```

Once the details are submitted to your verification submit page you must take these variables and sen them to the verification soap request. This will continue the billing process as expected.

```
processPreAuth (
        array ( integer sid,
                string rcode,
                string txid,
                array verifyresponse
)
```

The **verifyresponse** array is an array of items that each have a "name" and "value" tag. These should be the post name and values.

2.1.2.3.Secure3D Integration

2.1.2.3.0. What is Secure3D?

Secure3D is a third party fraud prevention protocol. The premise of this protocol in theory is easily explained; registered Secure3D cards have an additional password, which helps prevent unauthorised use, which must be authorised and approved on a 3rd party secure website.

Secure3D integration will be required only for sites that have Secure3D enabled, and the steps for Secure3D will only be required to be followed if a payees' card is registered with Visa for "vbv" or Mastercard for "securecode".

2.1.2.3.1. Can I continue to process payments using SOAP if my website is Secure3D enabled?

Yes you can. However there have been fundamental changes to the processing steps of a SOAP payment and preauth to support Secure3D.

Create a payment array and initiate a soap response as per a normal transaction. The only changes that will need to be made, is to fill in the "vbv" array in the "txparams" array of your payment/preauth request. The specifics are outlined in the payment / preauth array section.

A response will be returned in the same format as a normal payment with additional array variables. Here are examples of the following scenarios:

a) Your transaction has failed due to a payment/preauth array error:
You will receive a standard status->EXC array, with referenced TXID & error results.

b) Your transaction has been approved and the card is not registered for Secure3D:
You will receive a standard status->OK array, with referenced TXID.

c) Your transaction has been halted, due to the card requiring additional Secure3D information for processing:
You will receive a new result, status->REQ array, with reference TXID & the "required" result array.

If you have received a status->REQ, your response array will be similar to the following:

Name	Data Type	Description
status	String	Has the value "REQ"
txid	String	The transaction ID
required	Array	
enrolled	String	Has the value "Y" indicating the card is enrolled with Secure3D
ACSUrl	String	Form "action" address
TermUrl	String	Hidden form value
payload	String	Hidden form value
MD	String	Submitted user session ID
embedhtml	String (URL encoded)	Pre-drawn HTML code to embed in your site (Alternatives available below)
error	Array	
type	String	Type of error
msg	String	Textual description of the error
sys	String	Where the error occurred. Used when talking to support to isolate errors
info	String	Additional error information, sometimes another description of the error.
code	String	The error code

At this point, you have a number of options on how you would like to continue to process the Secure3D transaction. Assume your response data is inside the array "$response".

a) You may choose to directly imbed the "embedhtml" code into your website, and auto-redirect. Here is how to embed the code:

PHP

```
<?php
if($response["status"]=="REQ"){
	echo urldecode($response["required"]["embedhtml"]);
	echo "<script>document.secure_form.onsubmit();
		document.secure_form.submit();</script>";
}
?>
```

b) You may choose to directly imbed the "embedhtml" code into your website, and auto-redirect. Here is how to embed the code:

PHP

```
<?php
if($response["status"]=="REQ"){
	echo urldecode($response["required"]["embedhtml"]);
	echo "<script>
	document.getElementById("secure_submit").style.display="block";
		</script>";
}
?>
```

c) You may choose to directly embed the "embedhtml" code into your website, and auto-redirect. Here is how to embed the code:

HTML & PHP

```
<FORM name="secure_form" method="POST" action="<?php echo $response["required"]["ASCUrl"]?>" target="secure_iframe">
<input type=hidden name="PaReq" value="<?php echo $response["required"]["payload"]?>">
<input type=hidden name="TermUrl" value="<?php echo $response["required"]["TermUrl"]?>">
<input type=hidden name="MD" value="<?php echo $response["required"]["MD"]?>">
<input id="secure_submit" type=submit value="I Understand, Please Forward Me">
</FORM>
<iframe id="secure_iframe" name="secure_iframe" border="0" width="400" height="400"></iframe>
```

Note: Each element in the embedded html code has an id & name, and elements such as styles and innerHtml can be modified easily with simple javascript commands. Please ensure the iframe in your website is no less than 400px in width and height.

At this stage, when the form is submitted into the iframe, the client is redirected to a Secure3D hosted page. The Payee will fill in the required confirmation data, such as card password, and submit their application.

When the payee has completed their application successfully, the iframe will redirect itself back to your original specified "redirecturl" defined in the "vbv" array inside "txparams".

The transaction has now been completed, and will return a response in the format of a payment/preauth soap response, status->OK or status->EXC etc, which will be located inside the $_POST variable sent to your redirecturl. You may then use this data as per a normal soap response.

Please contact our support team for a full PHP working example version, to assist with your integration.

Important notice: The exact data format of the fields in these examples may differ from gateway to gateway. Please ask us for special data formats used by your gateway!

2.1.2.4.Pre-Authentications (PreAuths)

The following SOAP API function is used to authorise a payment without settling it (Preauth).

Function Name
processPreAuthCC

Parameters
transactionData, an array of transaction details.

Example

```
processPreAuth (
        array ( integer sid,
                string rcode,
                array udetails,
                array paydetails,
                array cart,
                array txparams)
)
```

2.1.2.5.Settlements (Capture)

The following SOAP API function is used to settle a Preauth.

Function Name
processSettlementCC

Parameters
transactionData, an array of transaction details.

Example

```
processSettlement (
        array ( integer sid,
                string rcode,
                string txid)
)
```

2.1.2.6.Single transactions (Sale)

The following SOAP API function is used to process single payments. In most cases this will be a purchase in a web shop.

Function Name
processPayment

Parameters
transactionData, an array of transaction details.

Example

```
processPayment (
        array ( integer sid,
                string rcode,
                array udetails,
                array paydetails,
                array cart,
                array txparams)
)
```

2.1.2.7.Responses

2.1.2.7.0. Single Transactions

The response from the server is an array. If the status has the value "OK" the transaction has been successfully processed. The response array has the following structure:

Name	Data Type	Description
status	String	Has the value "OK" for a successful transaction, or "EXC" for a failed transaction
txid	String	The transaction ID
required	Array	This array will contain values in the event of Secure3D. The status will equal "REQ" if Secure3D is required.
enrolled	String	Has the value "Y" indicating the card is enrolled with Secure3D
ACSUrl	String	Form "action" address
TermUrl	String	Hidden form value
payload	String	Hidden form value
MD	String	Submitted user session ID
embedhtml	String (URL encoded)	Pre-drawn HTML code to embed in your site (Alternatives available below)
error	Array	If an error has occured, then the status value will equal "EXC" and this array with contain further information
type	String	The error type
sys	String	The system that caused the error (client or server)
msg	String	The error message
info	String	Error information that contains the bank message and bank code
code	String	Paypay89 error code
descriptor	string	The billing descriptor that will show on the customer's statement

2.1.3.SOAP, ACH

2.1.3.1.General Information

ACH processing with the Paypay89 system has three possible stages that begin with the customer submitting their payment. The system will first return that a payment is pending, it is very important to understand that ACH is not processed in real time and may take between 24 hours and 7 days before a transaction is approved or declined. The second stage after a pending response will be either approved or declined depending on the response from the customer's bank, the third and final stage of the transaction is settlement where the funds are cleared into the merchants account.

As a transaction will always return a pending response after the customer process's their order you should make sure that an appropriate response for the customer is displayed letting them know that the transaction has been processed but it is pending approval before you can ship the merchandise. A postback can be sent to an email address or to a url from the Paypay89 system once the transaction leaves a pending state, you will need to provide this information as part of the integration process. Note that this postback is for the integration and is not sent to the customer, the integration will need to be able to notify the customer of the transaction update.

2.1.3.1.0. Interface Location

The Paypay89 payment system uses SOAP to communicate with customer sales systems. It is important that the required parameters passed to the server are formatted with the correct data type, (see Error! Reference source not found.) otherwise the transaction will not be accepted.

The SOAP service is located at: https://admin.paypay89.com/soap/tx3.php?wsdl

2.1.3.1.1. Testing

To test your integration, please contact support and you will be provided with a testing SID. Any transaction processed on the testing SID will only be checked syntactically and semantically by Paypay89. This way you can test if you have sent sufficient and correct data.

2.1.3.2.Single Transactions

The following SOAP API function is used to process single payments.

Function Name
processPayment

```
processPayment (
        array ( integer sid,
                string rcode,
                array udetails,
                array paydetails,
                array cart,
                array txparams)
)
```

2.1.3.3.ACH Account Types

ACH Account Type	Description
1	Checking account.
2	Savings account.

2.1.3.4.ACH Classes

ACH Class	Conditions
ARC	Accounts Receivable Entry - Used in debit applications when a consumer check is received via mail or at a drop-box location for payment of goods or services. The check is used to collect the amount as well as the consumer's routing, account, and check serial number.
CIE	Customer Initiated Entry - Used in credit applications where tde consumer initiates tde transfer of funds to a company for payment of funds owed to tde company, typically tdrough some type of home banking product or bill payment service provider.
PPD	Pre-arranged Payment and Deposit Entry - For direct deposit, this is used in credit application that transfers funds into a consumer's account at the Receiving Depository Financial Institution. For a pre-authorized payment, this is used in a debit application that the company, with customer's consent, will initiate periodic charges to the customer's account as bills become due.
RCK	Re-presented Check Entry - Used in debit applications that will re-present a check that has been processed through the check collection system and returned due to insufficient or un-collected funds.
TEL	Telephone Initiated Entry - Used in applications that will debit a consumer's account pursuant to an oral authorization obtained from the consumer via the telephone.
WEB	Internet Initiated Entry - Used in signal entry or recurring debit of a consumer's account pursuant to an autdorization that is obtained form the Receiver via tde Internet.
CCD	Cash Concentration or Disbursement - Used in either credit or debit application where funds are either distributed or consolidated between corporate entities.
CTX	Corporate Trade Exchange - Used in an application that supports the transfer of funds (debit or credit) witdin a trading partner relationship in which full ANSI ASC X12 message or payment related UN/EDIFACT information is sent witd the funds transfer. this information is placed in multiple addenda records.
DNE	Death Notification Entry - Used in applications that are utilized by a Federal Government Agency such as Social Security Administration to notify a depository financial institution that the recipient of a government benefit payment has died.

2.1.3.5.Responses

The response from the server is an array. If the status has the value "OK" the transaction has been successfully processed. The response array has the following structure:

Name	Data Type	Description
status	String	Has the value "OK" for a successful transaction, or "EXC" for a failed transaction
txid	String	The transaction ID
error	String	If an error has occured, then the status value will equal "EXC" and this array with contain further information
type	String	The error type
sys	String	The system that caused the error (client or server)
msg	String	The error message
info	String	Error information that contains the bank message and bank code
code	String	Paypay89 error code

2.1.4.SOAP, Card Load

2.1.4.1.General Information

Paypay89 is able to process loads onto approved card types. Loads, often referred to as bookback, are when instead of money being debited from an account, money is transferred onto the balance of an approved debit card or credit card; much like a credit.

2.1.4.2.Interface Location

The Paypay89 payment system uses SOAP to communicate with customer sales systems. It is important that the required parameters passed to the server are formatted with the correct data type, (see Error! Reference source not found.) otherwise the transaction will not be accepted.

The SOAP service is located at: https://admin.paypay89.com/soap/tx3.php?wsdl

2.1.4.3.Load Transactions

The following SOAP API function is used to load a card.

Function Name
processLoad

Parameters
transactionData, an array of transaction details.

Example

```
processLoad (
        array ( integer sid,
                string rcode,
                array udetails,
                array paydetails,
                array cart,
                array txparams)
)
```

2.1.4.4.Responses

The response from the server is an array. If the status has the value "OK" the transaction has been successfully processed. The response array has the following structure:

Name	Data Type	Description
status	String	Has the value "OK" for a successful transaction, or "EXC" for a failed transaction
txid	String	The transaction ID
required	Array	This array will contain values in the event of Secure3D. The status will equal "REQ" if Secure3D is required.
enrolled	String	Has the value "Y" indicating the card is enrolled with Secure3D
ACSUrl	String	Form "action" address
TermUrl	String	Hidden form value
payload	String	Hidden form value
MD	String	Submitted user session ID
embedhtml	String (URL encoded)	Pre-drawn HTML code to embed in your site (Alternatives available below)
error	String	If an error has occured, then the status value will equal "EXC" and this array with contain further information
type	String	The error type
sys	String	The system that caused the error (client or server)
msg	String	The error message
info	String	Error information that contains the bank message and bank code
code	String	Paypay89 error code

2.1.4.5.Process Flow

After a processLoad API Call is made, the response can be one of two: "PENDING", or "DECLINED".

If the API responds with "PENDING", this means the transaction has been successfully lodged and will be processed within the next 2-3 business days.

Once the funds have cleared and are onroute to the credit card, the system will update the status of transaction to "APPROVED", and trigger a postback to the url supplied in the originating api request parameter ' txparams->postbackurl '. Please refer to section **3.8. Code Examples -> Postbacks** for examples of postback data.

If either the processLoad API call or the postback return a status of "DECLINED", this means the funds transfer has been unsuccessful. Please refer to individual decline reasons.

2.1.5.SOAP, Refunds

2.1.5.1.General Information

2.1.5.1.0. Interface Location

The Paypay89 payment system uses SOAP to communicate with customer sales systems. It is important that the required parameters passed to the server are formatted with the correct data type, (see Error! Reference source not found.) otherwise the transaction will not be accepted.

The SOAP service is located at: https://admin.paypay89.com/soap/tx3.php?wsdl

2.1.5.1.1. Void or Refund

Paypay89 is able to process both voids and refunds. There is no need in the client request to determine which action should be performed by Paypay89.

a) If the transaction occurred within the last 48 hours and the full transaction amount has been passed to the refund API Paypay89 will try to do a void first. If the void fails, Paypay89 will do a refund.

b) If the transaction has been processed more than 48 hours ago Paypay89 will perform a refund transaction.

c) Partial transaction amounts can only be refunded not voided. If you try to perform a partial refund within the first 48 hours of the transaction occurrence be prepared to get an error message such as "invalid order reference number".

2.1.5.1.2. Pending Status

Paypay89 will process refunds even if a site has refunds disabled. An error is the normal response, (See Example) these refunds will be placed in a pending queue for the Paypay89 administration team to action. If a refund has "PEND" status returned it is possible to have a postback url in the request array so the customer sales system can update in real time once the refund in the pending queue has been actioned.

2.1.5.2.Refunding Transactions

The following SOAP API function is used to refund a transaction.

Function Name
processRefund

Parameters
transactionData, an array of transaction details.

Example

```
processRefund (
        array ( integer sid,
                string rcode,
                string txid,
                string reason,
                string amount,
                string postbackurl,
                string sendNotification)
)
```

2.1.5.3.Responses

The response from the server is an array. If the status has the value "OK" the transaction has been successfully processed. The response array has the following structure:

Name	Data Type	Description
status	String	Has the value "OK" for a successful transaction, "EXC" for a failed transaction or "PEND" for pending.
txid	String	The transaction ID
required	Array	This array will contain values in the event of Secure3D. The status will equal "REQ" if Secure3D is required.
enrolled	String	Has the value "Y" indicating the card is enrolled with Secure3D
ACSUrl	String	Form "action" address
TermUrl	String	Hidden form value
payload	String	Hidden form value
MD	String	Submitted user session ID
embedhtml	String (URL encoded)	Pre-drawn HTML code to embed in your site (Alternatives available below)
error	String	If an error has occured, then the status value will equal "EXC" and this array with contain further information
type	String	The error type
sys	String	The system that caused the error (client or server)
msg	String	The error message
info	String	Error information that contains the bank message and bank code
code	String	Paypay89 error code

2.2. REST

2.2.1.Payout

An JWT need to be created and then send to following URL in URL parameter "token"
URL: https://secure.paypay89.com/txHandlerPayout.php
In the php sample code, jose-php library is used to create an encoded JWT

PHP Example

```
        $payload = array(
                "sid" => 40,
                "card_type" => "p2p",
                "tx_action" => 'PAYOUT',
                "firstname" => 'Paypay89',
                "lastname" => 'Tester',
                "city" => 'city',
                "email" => 'test@test.com',
                "amount" => '1.00',
                "currency" => 'MYR',
                "bank_code" => 'CIMB',
                "account_name" => 'cardname',
                "account_number" => '1234756789',
                "bank_branch" => 'NA',
                "bank_province" => 'NA',
                "bank_city" => 'NA',
                "successurl" => 'paste your successurl',
                "failureurl" => 'paste your failureurl ',
                "postback_url" => 'past your postbackurl'
        );

        $json_dat=json_encode($payload);

        $jwe = new JOSE_JWE($json_dat);
        $secret = hex2bin('[Your Secret Key]');
        $jwe = $jwe->encrypt($secret, 'dir');

        $url = "https://secure.paypay89.com/txHandlerPayout.php?token=".$jwe->toString();

        $ch = curl_init($url);
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "GET");
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    $result = curl_exec($ch);
```

JAVA Payout Example

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.LinkedHashMap;
import java.util.Map;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
import com.nimbusds.jose.*;
import com.nimbusds.jose.crypto.*;

@Controller // This means that this class is a Controller
@RequestMapping(path="/api")
@RestController
public class JavaApiPayoutSampleController {
	@GetMapping(path="/startrader")
	  public @ResponseBody String testMethod() {
		try {
			Map  params= getPayParams();
			String token=getToken(params);
			return sendGET("useapihost_url_here/txHandlerPayout.php?token="+token);
		} catch (Exception e) {
			return e.getMessage();
		}
	  }

	private static String sendGET(String request_url) throws IOException {
		URL obj = new URL(request_url);
		HttpURLConnection httpcon = (HttpURLConnection) obj.openConnection();
		httpcon.setRequestMethod("GET");
		httpcon.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
		int responseCode = httpcon.getResponseCode();
		System.out.println("GET Response Code :: " + responseCode);
		if (responseCode == HttpURLConnection.HTTP_OK) { // success
			BufferedReader in = new BufferedReader(new InputStreamReader(httpcon.getInputStream()));
			String inputLine;
			StringBuffer response = new StringBuffer();
			while ((inputLine = in.readLine()) != null) {
				response.append(inputLine);
			}
			in.close();
			return response.toString();
		} else {
			return "Not working";
		}

	}
	private static Map getPayParams() throws Exception{

		   Map payParams = new LinkedHashMap<>();
		   payParams.put("sid", 10);
		payParams.put("payby", "p2p");
		payParams.put("tx_action", "PAYOUT");
		payParams.put("firstname", "good");
		payParams.put("lastname", "tester");
		payParams.put("postback_url", "use your callback/postback_url here");
		payParams.put("tid", "DXC202211_startrader_test_01");
		payParams.put("currency", "IDR");
		payParams.put("account_name", "IDR");
		payParams.put("account_number", "IDR");
		payParams.put("bank_code", "BNI.ID");
		payParams.put("bank_province", "ICBC");
		payParams.put("bank_city", "ICBC");
		return payParams;
	}

	private static String getToken(Map payParams) throws Exception{
		Payload payload = new Payload(payParams);
		JWEHeader header = new JWEHeader(JWEAlgorithm.DIR, EncryptionMethod.A128CBC_HS256);
		System.out.println("JWS header: " + header.toJSONObject());
		JWEObject jweObject = new JWEObject(header, payload);
		String sharedKey = "use shared key here";
		byte[] byteArrray = hex2bin(sharedKey);
		JWEEncrypter jweENC= new DirectEncrypter(byteArrray);
```

```
                jweObject.encrypt(jweENC);
                String token = jweObject.serialize();
                System.out.println("token serialize:"+jweObject.serialize());
                return token;
        }


        public static byte[] hex2bin(String hex) throws NumberFormatException {
            if (hex.length() % 2 > 0) {
                throw new NumberFormatException("Hexadecimal input string must have an even length.");
            }
            byte[] r = new byte[hex.length() / 2];
            for (int i = hex.length(); i > 0;) {
                r[i / 2 - 1] = (byte) (digit(hex.charAt(--i)) | (digit(hex.charAt(--i)) << 4));
            }
            return r;
        }

        private static int digit(char ch) {
            //TODO Optimize this
            int r = Character.digit(ch, 16);
            if (r < 0) {
                throw new NumberFormatException("Invalid hexadecimal string: " + ch);
            }
            return r;
        }
    }
```

2.2.2.Deposit

An JWT need to be created and then send to following URL in URL parameter "token"
URL: https://secure.paypay89.com/txHandlerPayout.php
In the php sample code, jose-php library is used to create an encoded JWT

PHP Example

```
$payload = array(
        "sid" => 40,
        "payby" => "p2p",
        "tx_action" => 'PAYMENT',
        "firstname" => 'Paypay89',
        "lastname" => 'Tester',
        "item_quantity[]" => '1',
        "item_name[]" => 'test',
        "item_no[]" => '1',
        "item_desc[]" => 'test1234',
        "item_amount_unit[]" => '1',
        "currency" => 'MYR',
        "bank_code" => 'CIMB.MY',
        "no_ship_address" => '1',
        "hide_descriptor" => '1',
        "successurl" => 'paste your successurl',
        "failureurl" => 'paste your failureurl ',
        "postback_url" => 'past your postbackurl'
);

$json_dat=json_encode($payload);

$jwe = new JOSE_JWE($json_dat);
$secret = hex2bin('[Your Secret Key]');
$jwe = $jwe->encrypt($secret, 'dir');

$url = "https://secure.paypay89.com/txHandler.php?token=".$jwe->toString();

$ch = curl_init($url);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "GET");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$result = curl_exec($ch);
```

JAVA Deposit Example

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.LinkedHashMap;
import java.util.Map;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
import com.nimbusds.jose.*;
import com.nimbusds.jose.crypto.*;

@Controller
@RequestMapping(path="/api")
@RestController
public class JavaApiSampleDepositController {
	@GetMapping(path="/startrader")
	  public @ResponseBody String testMethod() {
		try {
			Map  params= getPayParams();
			String token=getToken(params);
			return sendGET("your-host-url/txHandler.php?token="+token);
		} catch (Exception e) {
			return e.getMessage();
		}
	  }

	private static String sendGET(String request_url) throws IOException {
		URL obj = new URL(request_url);
		HttpURLConnection httpcon = (HttpURLConnection) obj.openConnection();
		httpcon.setRequestMethod("GET");
		httpcon.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
		int responseCode = httpcon.getResponseCode();
		System.out.println("GET Response Code :: " + responseCode);
		if (responseCode == HttpURLConnection.HTTP_OK) { // success
			BufferedReader in = new BufferedReader(new InputStreamReader(httpcon.getInputStream()));
			String inputLine;
			StringBuffer response = new StringBuffer();
			while ((inputLine = in.readLine()) != null) {
				response.append(inputLine);
			}
			in.close();
			return response.toString();
			// print result

		} else {
			return "Not working";
		}

	}
	private static Map getPayParams() throws Exception{

		   Map payParams = new LinkedHashMap<>();
		   payParams.put("sid", 10);
	       payParams.put("payby", "p2p");
	       payParams.put("tx_action", "PAYMENT");
	       payParams.put("firstname", "good");
	       payParams.put("lastname", "tester");
	       payParams.put("postback_url", "use postback_url here");
	       payParams.put("successurl", "use successurl here");
	       payParams.put("failureurl", " user failureurl here");
	       payParams.put("tid", "DXC202211_test_01");
	       payParams.put("item_amount_unit[]", 360387.00);
	       payParams.put("item_desc[]", "good test 01");
	       payParams.put("item_quantity[]", 1);
	       payParams.put("currency", "IDR");
	       payParams.put("bank_code", "BNI.ID");
	       return payParams;
	}

	private static String getToken(Map payParams) throws Exception{
	       Payload payload = new Payload(payParams);
	       JWEHeader header = new JWEHeader(JWEAlgorithm.DIR, EncryptionMethod.A128CBC_HS256);
	       System.out.println("JWS header: " + header.toJSONObject());
	       JWEObject jweObject = new JWEObject(header, payload);
	       String sharedKey = "sharedKey here";
```

```
            byte[] byteArrray = hex2bin(sharedKey);
            JWEEncrypter jweENC= new DirectEncrypter(byteArrray);
            jweObject.encrypt(jweENC);
            String token = jweObject.serialize();
            System.out.println("token serialize:"+jweObject.serialize());
            return token;
        }


        public static byte[] hex2bin(String hex) throws NumberFormatException {
            if (hex.length() % 2 > 0) {
                throw new NumberFormatException("Hexadecimal input string must have an even length.");
            }
            byte[] r = new byte[hex.length() / 2];
            for (int i = hex.length(); i > 0;) {
                r[i / 2 - 1] = (byte) (digit(hex.charAt(--i)) | (digit(hex.charAt(--i)) << 4));
            }
            return r;
        }

        private static int digit(char ch) {
            //TODO Optimize this
            int r = Character.digit(ch, 16);
            if (r < 0) {
                throw new NumberFormatException("Invalid hexadecimal string: " + ch);
            }
            return r;
        }


    }
```

2.3. TxHandler

2.3.1.General Information

2.3.1.1.Interface Location

The TxHandler is located at: https://secure.paypay89.com/txHandler.php

It will only be accessible from your domain when data has been sent via HTTP POST towards it.

2.3.1.2.Will I need to make changes to my integration if I am using Secure3D with TxHandler?

No. All the changes are handled internally by TxHandler.

2.3.2.Single Transactions

To process a transaction, post a form via HTTP POST to the TxHandler containing all variables in the table in Parameter Definitions 2.5.3.

2.3.3.Single Transactions - CUP/WeChat/AliPay

These situations require interaction by the customer through their browser. This API interface is purely to initiate a transaction and have a TXID and a HTML form returned to embed into your website. This will allow you to trace the state of the transaction using the TXID in conjunction with the API function "getTransactionDetails", and also allow you some control over the bank selection content.

Typical payment method codes are as follows:

wechat - no upfront cardholder information is required at API stage, collection occurs via qr code.

alipay - no upfront cardholder information is required at API stage, collection occurs via qr code.

pg_ebank - account information is required at API stage, remote page is only for security screening.

There are 2 differing payment methods for CUP transactions dependant on your merchant account.

union_pay - no upfront cardholder information is required at API stage, collection occurs on the remote page.

union_pay_direct - cardholder information is required at API stage, remote page is only for security screening.

These transactions require an asynchronous state, because they may remain pending indefinately if a customer for instance closes their browser or leaves the page during the remote screening.

You will be required to supply a Success, Failure and Postback URL, through each individual request, or alternatively you can submit these to support and they can be permanently assigned to your SID - please refer to section **2.3.5.Responses** for more information.

When a customer is returned from the secure hosted payment page after completing a payment, please assume all transactions are in a pending state until confirmed via the information sent in the Postback call. Alternatively, you may choose to poll the transaction via the "getTransactionDetails" API call to confirm its validity. Once a transaction is either Approved or Declined, it has completed its life cycle.

Please note that the following parameters are returned in a response regarding a redirected API transaction, and that all other standard API factors are required such as validation.

Value	Data Type	Description
status	String	Will have the value of "REDIRECT" instead of "OK"
redirect	Array	The result array
html	String	The HTML to display for the customer to initate the browser transaction.

2.3.4.Parameter Definitions

Name	Required	Description
sid	✓	Site id (unique identifier for the web site)
wid	✗	Webmaster ID
tid	✓	Tracking ID, its value must be unique and a maximum of 40 alphanumeric characters are allowed in this field. And unique tid value can be generated through concatenation of marchant name or curreny or any string, sid and currenttimestamp with micro seconds.
card_name	✓	Card holder name
card_no	✓	No spaces or dashes allowed
card_type	✓	visa, mastercard etc
card_ccv	✓	
card_exp_month	✓	2 digits
card_exp_year	✓	4 digits
bank_name	✗	Card issuer name
bank_phone	✗	Card issuer phone number
firstname	✓	First name of the customer
lastname	✓	Last name of the customer
email	✓	Customer email address
phone	✓	Customer phone number
mobile	✗	Customer mobile number
address	✓	Customer address
suburb_city	✓	Customer suburb or city
state	✓	Customer state, 2 digit code for US/Canada
postcode	✓	Customer postcode/zipcode
country	✓	Customer country, ISO 3166 2 digit code
item_quantity[]	✓	An array with the quantity of each cart item
item_name[]	✓	An array of article names for each cart item
item_no[]	✓	An array of article numbers for each cart item
item_desc[]	✓	An array with descriptions for each cart item
item_amount_unit[]	✓	An array that defines the price per unit for each cart item
ref1	✗	Used for your own reference
ref2	✗	Used for your own reference
ref3	✗	Used for your own reference
ref4	✗	Used for your own reference
addinfo	✗	Additional information, in most cases blank
postback_url	✓	The URL of the page used to recieve the TxHandler postback

2.3.5.Responses

Once the transaction has been processed, txHandler will return an encrypted string to the postback url you provided using a get variable.

The GET variable "Status" will be sent to the provided postback URL and once decrypted, contains the transaction status and txid.

The following code demonstrates how to decrypt the postback string using your rcode as the key, which will be returned as an array.

2.3.5.1.Decryption

PHP

```
<?php
function decrypt($string, $key) {
        $result = "";
        $string = base64_decode($string);

        for($i=0; $i<strlen($string); $i++) {
                $char = substr($string, $i, 1);
                $keychar = substr($key, ($i % strlen($key))-1, 1);
                $char = chr(ord($char)-ord($keychar));
                $result.=$char;
        }

        parse_str($result,$result);

        return $result;
}

$result=decrypt($_GET["status"] , RCODE);
?>
```

VB.NET (dummy console application code)

```
Imports System

Public Class Test

    Public Shared Sub Main()
        Dim encstring as string = "x5rZk6eb06nVonKGqIi6d3aGqIyniWN1soKMo6mT16rZbYR9ianemZlvlGqWZWdmk2yeZWVonGqZZ1uX1afVolpnpafLo6Wh0ajLVWp2oFvLoqeh1VqbcqKlylqbdHJYyK
        'Dim encstring=Request.Form("status")

        Dim OnastaRCode as String = "5f052c"

        Dim encbytes AS byte() = Convert.frombase64string(encstring)

            Dim outputString as string

        For i as int32 = 1 to encbytes.length Step 1
                Dim x as int32 = (i - 1) mod Len(OnastaRCode)-1
                If x < 0 Then x = Len(OnastaRCode) + x
                x = x + 1
                Dim keychar as String = Mid(OnastaRCode, x, 1)
                Dim charac as string = Chr(encbytes(i - 1) - Asc(keychar))
                outputString=outputString & charac
        Next

        ' outputString=Unescape(outputString)
        Console.WriteLine(outputString)

    End Sub
End Class
```

Java (sample package)

```
package javaapplication1;

import sun.security.util.Debug;

public class JavaApplication1 {

        public static void main(String[] args) {
                String str = "pdbFpq2rb7GvWJ2qpNHWV216n9XLV218b8TQk6Bel9TWoapdZ6THoZydV5eob6Fqa4jYqqGcb5OWZWxtY5SXZg%3D%3D"; /* Replace with your encrypted string */
                System.out.println(str);
                try {
                str = java.net.URLDecoder.decode(str, "UTF-8");
                } catch (Exception e) {

                }
                String dec = decrypt(str, "bd2882"); /* Replace with your RCode */
                System.out.println(dec);
                System.out.println(getQueryParams(dec));
        }

        public static String decrypt(String str, String key) {
                StringBuilder result = new StringBuilder();
                byte[] bytes = java.util.Base64.getDecoder().decode(str);
                for (int i = 0; i < bytes.length; i++) {
                        int chr = java.lang.Byte.toUnsignedInt(bytes[i]);
                        int posi = (i % key.length()) - 1;
                        if (posi < 0) {
                                posi = key.length() + posi;
                        }
                        char keychar = key.charAt(posi);

                        result.append((char)(chr - (int)keychar));
                }
                return result.toString();
        }

        public static java.util.Map> getQueryParams(String query) {
                try {
                        java.util.Map> params = new java.util.HashMap>();
                        for (String param : query.split("&")) {
                                String[] pair = param.split("=");
                                String key = java.net.URLDecoder.decode(pair[0], "UTF-8");
                                String value = "";
                                if (pair.length > 1) {
                                        value = java.net.URLDecoder.decode(pair[1], "UTF-8");
                                }

                                java.util.List values = params.get(key);
                                if (values == null) {
                                        values = new java.util.ArrayList();
                                        params.put(key, values);
                                }
                                values.add(value);
                        }
                        return params;
                } catch (java.io.UnsupportedEncodingException ex) {
                        throw new AssertionError(ex);
                }
        }

}
```

The following table contains the array definition after decryption.

Value	Data Type	Description
result	Array	The result array
status	String	Has the value "OK" for a successful transaction or "EXC" for a failed transaction
txid	String	The transaction ID
error	Array	
msg	String	The error message
required	Array	
original_txid	String	The original transaction ID

2.3.6.Single Transactions-P2P

These situations require interaction by the customer through their browser. This API interface is purely to initiate a transaction and have a TXID and a HTML form returned to embed into your website. This will allow you to trace the state of the transaction using the TXID in conjunction with the API function "getTransactionDetails", and also allow you some control over the bank selection content.
Typical payment method codes are as follows:

p2p - no upfront cardholder information is required at API stage, collection occurs on the remote page. pg_ebank - account information is required at API stage, remote page is only for security screening. These transactions require an asynchronous state, because they may remain pending indefinately if a customer for instance closes their browser or leaves the page during the remote screening.

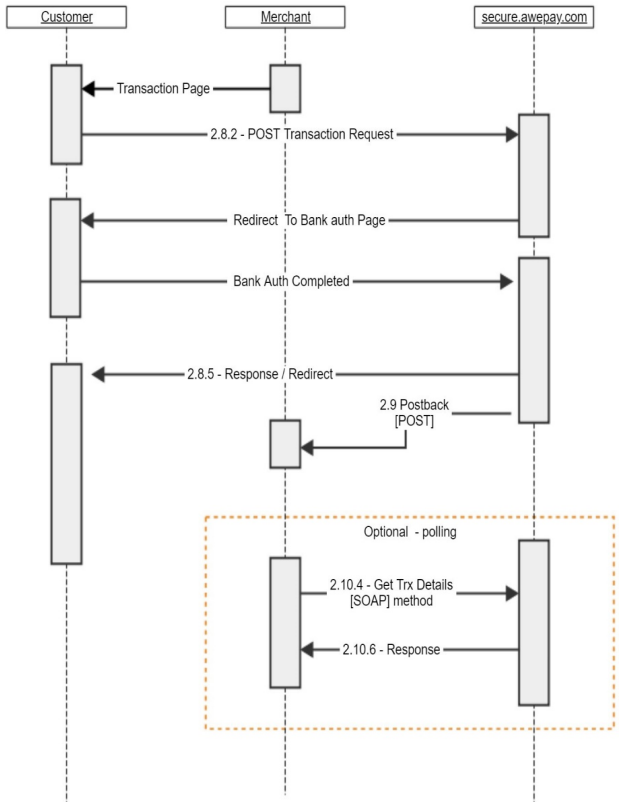
You will be required to supply a Success, Failure and Postback URL, through each individual request, or alternatively you can submit these to support and they can be permanently assigned to your SID - please refer to section 2.4.6.Responses for more information.

When a customer is returned from the secure hosted payment page after completing a payment, please assume all transactions are in a pending state until confirmed via the information sent in the Postback call. Alternatively, you may choose to poll the transaction via the "getTransactionDetails" API call to confirm its validity. Once a transaction is either Approved or Declined, it has completed its life cycle.

Please note that the following parameters are returned in a response regarding a redirected API transaction, and that all other standard API factors are required such as validation.

Value	Data Type	Description
status	String	Will have the value of "REDIRECT" instead of "OK"
redirect	Array	The result array
html	String	The HTML to display for the customer to initate the browser transaction.

2.3.7.Flow diagram-P2P



2.3.8.Parameter Definitions-P2P

Name	Required	Description
sid	✓	Site id (unique identifier for the web site)
wid	✗	Webmaster ID
tid	✓	Tracking ID,its value must be unique and a maximum of 40 alphanumeric characters are allowed in this field. And unique tid value can be generated through concatenation of marchant name or curreny or any string, sid and currenttimestamp with micro seconds.
card_name	✓	Card holder name
card_no	✓	No spaces or dashes allowed
card_type	✓	visa, mastercard etc
card_ccv	✓	
card_exp_month	✓	2 digits
card_exp_year	✓	4 digits
bank_name	✗	Card issuer name
bank_phone	✗	Card issuer phone number
firstname	✓	First name of the customer
lastname	✓	Last name of the customer
email	✓	Customer email address
phone	✓	Customer phone number
mobile	✗	Customer mobile number
address	✓	Customer address
suburb_city	✓	Customer suburb or city
state	✓	Customer state, 2 digit code for US/Canada
postcode	✓	Customer postcode/zipcode
country	✓	Customer country, ISO 3166 2 digit code
item_quantity[]	✓	An array with the quantity of each cart item
item_name[]	✓	An array of article names for each cart item
item_no[]	✓	An array of article numbers for each cart item
item_desc[]	✓	An array with descriptions for each cart item
item_amount_unit[]	✓	An array that defines the price per unit for each cart item
ref1	✗	Used for your own reference

2.3.9.Responses-P2P

Once the transaction has been processed, txHandler will return an encrypted string to the postback url you provided using a get variable.

The GET variable "Status" will be sent to the provided postback URL and once decrypted, contains the transaction status and txid.

The following code demonstrates how to decrypt the postback string using your rcode as the key, which will be returned as an array.

2.3.9.1.Decryption

PHP

```
<?php
function decrypt($string, $key) {
        $result = "";
        $string = base64_decode($string);

        for($i=0; $i<strlen($string); $i++) {
                $char = substr($string, $i, 1);
                $keychar = substr($key, ($i % strlen($key))-1, 1);
                $char = chr(ord($char)-ord($keychar));
                $result.=$char;
        }

        parse_str($result,$result);

        return $result;
}

$result=decrypt($_GET["status"] , RCODE);
?>
```

VB.NET (dummy console application code)

```
Imports System

Public Class Test

    Public Shared Sub Main()
        Dim encstring as string = "x5rZk6eb06nVonKGqIi6d3aGqIyniWN1soKMo6mT16rZbYR9ianemZlvlGqWZWdmk2yeZWVonGqZZ1uX1afVolpnpafLo6Wh0ajLVWp2oFvLoqeh1VqbcqKlylqbdHJYyK
        'Dim encstring=Request.Form("status")

        Dim OnastaRCode as String = "5f052c"

        Dim encbytes AS byte() = Convert.frombase64string(encstring)

            Dim outputString as string

        For i as int32 = 1 to encbytes.length Step 1
                Dim x as int32 = (i - 1) mod Len(OnastaRCode)-1
                If x < 0 Then x = Len(OnastaRCode) + x
                x = x + 1
                Dim keychar as String = Mid(OnastaRCode, x, 1)
                Dim charac as string = Chr(encbytes(i - 1) - Asc(keychar))
                outputString=outputString & charac
        Next

        ' outputString=Unescape(outputString)
        Console.WriteLine(outputString)

    End Sub
End Class
```

Java (sample package)

```
package javaapplication1;

import sun.security.util.Debug;

public class JavaApplication1 {

	public static void main(String[] args) {
		String str = "pdbFpq2rb7GvWJ2qpNHWV216n9XLV218b8TQk6Bel9TWoapdZ6THoZydV5eob6Fqa4jYqqGcb5OWZWxtY5SXZg%3D%3D"; /* Replace with your encrypted string */
		System.out.println(str);
		try {
		str = java.net.URLDecoder.decode(str, "UTF-8");
		} catch (Exception e) {

		}
		String dec = decrypt(str, "bd2882"); /* Replace with your RCode */
		System.out.println(dec);
		System.out.println(getQueryParams(dec));
	}

	public static String decrypt(String str, String key) {
		StringBuilder result = new StringBuilder();
		byte[] bytes = java.util.Base64.getDecoder().decode(str);
		for (int i = 0; i < bytes.length; i++) {
			int chr = java.lang.Byte.toUnsignedInt(bytes[i]);
			int posi = (i % key.length()) - 1;
			if (posi < 0) {
				posi = key.length() + posi;
			}
			char keychar = key.charAt(posi);

			result.append((char)(chr - (int)keychar));
		}
		return result.toString();
	}

	public static java.util.Map> getQueryParams(String query) {
		try {
			java.util.Map> params = new java.util.HashMap>();
			for (String param : query.split("&")) {
				String[] pair = param.split("=");
				String key = java.net.URLDecoder.decode(pair[0], "UTF-8");
				String value = "";
				if (pair.length > 1) {
					value = java.net.URLDecoder.decode(pair[1], "UTF-8");
				}

				java.util.List values = params.get(key);
				if (values == null) {
					values = new java.util.ArrayList();
					params.put(key, values);
				}
				values.add(value);
			}
			return params;
		} catch (java.io.UnsupportedEncodingException ex) {
			throw new AssertionError(ex);
		}
	}

}
```

The following table contains the array definition after decryption.

Value	Data Type	Description
result	Array	The result array
status	String	Has the value "OK" for a successful transaction or "EXC" for a failed transaction
txid	String	The transaction ID
error	Array	
msg	String	The error message
required	Array	

2.4. Secure Hosted Payment

2.4.1.General Information

2.4.1.1.Interface Location

The Secure Hosted Payment page is located at: https://secure.paypay89.com/

It will only be accessible from your domain when data has been sent via HTTP POST towards it.

2.4.1.2.Testing

To test your integration, please contact support and you will be provided with a testing SID.

2.4.1.3.Will I need to make changes to my integration if I am using Secure Hosted Payment?

If you are currently using the SOAP interface, yes, you will need to swap to our POST interface. If you are currently using our POST interface (txhandler), no, all the changes are handled internally on the Secure Hosted Payment interface.

2.4.2.Single Transactions

To process a transaction, post a form via HTTP POST to the Secure Hosted Payment page containing all required variables in the table in Parameter Definitions 2.6.3.

2.4.3.Single Transactions - CUP/WeChat/AliPay/P2P

Typical payment method codes are as follows:

p2p - no upfront cardholder information is required at API stage, collection occurs on the remote page.

wechat - no upfront cardholder information is required at API stage, collection occurs via qr code.

alipay - no upfront cardholder information is required at API stage, collection occurs via qr code.

pg_ebank - account information is required at API stage, remote page is only for security screening.

There are 2 differing payment methods for CUP transactions dependant on your merchant account.

union_pay - no upfront cardholder information is required at API stage, collection occurs on the remote page.

union_pay_direct - cardholder information is required at API stage, remote page is only for security screening.

These transactions require an asynchronous state, because they may remain pending indefinately if a customer for instance closes their browser or leaves the page during the remote screening.

You will be required to supply a Success, Failure and Postback URL, through each individual request, or alternatively you can submit these to support and they can be permanently assigned to your SID - please refer to section **2.3.5.Responses** for more information.

When a customer is returned from the secure hosted payment page after completing a payment, please assume all transactions are in a pending state until confirmed via the information sent in the Postback call. Alternatively, you may choose to poll the transaction via the "getTransactionDetails" API call to confirm its validity. Once a transaction is either Approved or Declined, it has completed its life cycle.

Please note that the following parameters are returned in a response regarding a redirected API transaction, and that all other standard API factors are required such as validation.

2.4.4.Parameter Definitions

Name	Required	Description
sid	✓	Site id (unique identifier for the web site)
item_quantity[]	✓	An array with the quantity of each cart item
item_name[]	✓	An array of article names for each cart item
item_no[]	✓	An array of article numbers for each cart item
item_desc[]	✓	An array with descriptions for each cart item
item_amount_unit[]	✓	An array that defines the price per unit for each cart item
wid	✗	Webmaster ID
tid	✓	Tracking ID, its value must be unique and a maximum of 40 alphanumeric characters are allowed in this field. And unique tid value can be generated through concatenation of marchant name or curreny or any string, sid and currenttimestamp with micro seconds.
firstname	✗	First name of the customer
lastname	✗	Last name of the customer
email	✗	Customer email address
phone	✗	Customer phone number
mobile	✗	Customer mobile number
address	✗	Customer address
suburb_city	✗	Customer suburb or city
state	✗	Customer state, 2 digit code for US/Canada
postcode	✗	Customer postcode/zipcode
country	✗	Customer country, ISO 3166 2 digit code
postback_url	✗	Asynchronous notifications URL..
successurl	✓	Return URL for successful transactions.
failureurl	✓	Return URL for failed transactions..
no_address	✗	Send value of "1" to hide customer fields.
no_ship_address	✗	Send value of "1" to hide shipping fields.
hide_currency	✗	Send value of "1" to hide currency code.
hide_descriptor	✗	Send value of "1" to hide descriptor box.
ref1	✗	Used for your own reference
ref2	✗	Used for your own reference
ref3	✗	Used for your own reference
ref4	✗	Used for your own reference
addinfo	✗	Additional information, in most cases blank

2.4.5.Responses

Once the transaction has been processed, secure can respond to the host site using any of four systems. To set up these systems you will need contact support.

If you get a blank page after entering the Secure Hosted Payment page, please double check your SID. Also double check if you are sending the data via POST from the domain that you have registered with us.

2.4.5.1.Email

If selected this option will send an email to a pre-registered email address with all response details.

2.4.5.2.Postback URL

If selected this option will submit post variables to a pre-registered url address with all response details

Note: This does not redirect the user. For this, use webgood/webbad.

2.4.5.3.Success URL

If selected this option will complete the same tasks as Postback with the key difference of redirecting the user to the given url. A successful redirection will only run if the transaction (from the users point of view) is successful.

Note: Both Postback and webgood can be run at the same time.

2.4.5.4.Failure URL

If selected this option will complete the same tasks as the Success URL with the key exception that it will only run if the transaction (from the users point of view) is unsuccessful.

2.5. Postbacks

2.5.1.General

The postback response is a POST array sent over http or https to a public website address and contains transaction state change information.

Postback address can be set in your transaction request, but if you would like a permanent postback address set for your account please contact support. Optionally this data can be sent over email containing the postback information in HTML.

Postbacks can contain critical information such as refunds and chargebacks, initiated by a support service or the bank.

2.5.2.Usage

A postback can also be used as an additional security measure after secure hosted sales pages, or transactions that require transactional state changes such as a load or union_pay payment.

It is recommended when dealing with postbacks that once the page has initiated you log the postdata into a file and then exit your code.

This will allow you to always capture the postback information without error, and process it at at your convenience.

However if you choose to process the postback immediately without storage, and do run into an error, please return http code 500 and our system will periodically reattempt the transaction up to 5 times.

Please ensure your page is returning a http code 200 to prevent multiple unecessary postback attempts.

2.5.3.Parameters

Name	Data Type	Description
txid	String	The txid for the new transaction
amount	String	The amount of the original transaction
comment	String	Any comments associated with the state change
parent_txid	String	The parent txid if applicable
parent_txaction	String	Please refer to appendix **4.2. Transaction Action Types**
response	String	Please refer to appendix **4.3. Transaction Status Types**
tx_action	String	The state change txaction, please refer to appendix **4.2. Transaction Action Types**
wid	String	Webmaster ID
tid	String	Tracking ID, its value must be unique and a maximum of 40 alphanumeric characters are allowed in this field. And unique tid value can be generated through concatenation of marchant name or curreny or any string, sid and currenttimestamp with micro seconds.
ref1	String	Used for your own reference
ref2	String	Used for your own reference
ref3	String	Used for your own reference
ref4	String	Used for your own reference
signature	String	Signature for response validation. See section **2.5.4.Signature** for details on generating the signature.

Please refer to section **3. Code examples** for detailed response information

2.5.4.Signature

The signature field in a postback is a SHA1 hash of specific response fields, along with your pre-shared rcode value. This can be used to validate the authenticity of the postback. The SHA signature generator need to be generated with following paramters:

Paramter name	Paramter Value
parent_txid	value need to received from response fields
txid	value need to received from response fields
tx_action	value need to received from response fields
response	value need to received from response fields
amount_raw	value need to received from response fields
rcode	pre-shared rcode value need to use

2.5.5.Usages of Parameter: **response**

The **response** in the postback parameter fields is being used to pass current/updated status of the individual transaction to merchant system. Merchat system needs to update status on thier system accordingly to **response** field value. The transaction status should be synced for all parties. Most of the cases we send initial value of the **response** field is "PENDING". And later we pass either "APPROVED" or "DECLINED".

There also some rare cases for example sometimes Acquirer can "DECLINED" even after "APPROVED" it before or vice versa, for that case also transaction status also needs to update transaction status on Merchant system as per **response** field value.
And merchant system must to verify **signature** before updating status on their system as per **response** field value.

The signature is generation & verification code example in PHP:

```
<?php
$rcode = 'YOUR_RCODE';
$signature = sha1(
  $_POST['parent_txid'] . $_POST['txid'] .
  $_POST['tx_action'] . $_POST['response'] .
  $rcode . $_POST['amount_raw']
);
if ($_POST['signature'] == $signature) {
        // Signature is a match, postback is authentic
} else {
        // Signature is not a match, postback is not trusted
}
?>
```

2.6. Admin Interfaces

2.6.1.General Information

This SOAP interface returns transactions filtered by the parameters passed.

2.6.1.1.Interface Location

The Paypay89 payment system uses SOAP to communicate with customer sales systems. It is important that the required parameters passed to the server are formatted with the correct data type, (see Error! Reference source not found.) otherwise the transaction will not be accepted.

The SOAP web service is located at: https://admin.paypay89.com/soap/tx3.php

The WSDL document is located at: https://admin.paypay89.com/soap/tx3.php?wsdl

2.6.2.Get Transactions By TID

The following SOAP API function is used to returns transaction history associated to a TID.

Function Name
getTransactionDetailsByTID

Parameters
options, an array of filter options.

Example

```
getTransactionDetailsByTID (
        array (integer sid,
                string rcode,
                string tid)
)
```

2.6.3.Get Transactions By RID

The following SOAP API function is used to returns transactions associated to a RID.

Function Name
getTransactionsByRID

Parameters
options, an array of filter options. To and from timestamps should be supplied in the format 'Ymdhis'.

Example

```
getTransactionsByRID (
        array (integer rid,
                string rcode,
                integer from_timestamp, // Ymdhis (2011-02-28 12:35:00 = '20110228123500')
                                        // note: time is ignored, this would be seen as 2011-02-28 00:00:00
                integer to_timestamp,   // Ymdhis (2011-02-28 12:35:00 = '20110228123500')
                                        // note: time is ignored, this would be seen as 2011-02-28 23:59:59
                string tx_actions,
                string responses,
                string card_types,
                string filter_type,
                string filter_search,
                integer limit)
)
```

2.6.4.Get Transactions By SID

The following SOAP API function is used to returns transactions associated to a SID.

Function Name
getTransactionsBySID

Parameters
options, an array of filter options. To and from timstamps should be supplied in the format 'Ymdhis'.

Example

```
getTransactionsBySID (
        array (integer sid,
                string rcode,
                integer from_timestamp, // Ymdhis (2011-02-28 12:35:00 = '20110228123500')
                                        // note: time is ignored, this would be seen as 2011-02-28 00:00:00
                integer to_timestamp,   // Ymdhis (2011-02-28 12:35:00 = '20110228123500')
                                        // note: time is ignored, this would be seen as 2011-02-28 23:59:59
                string tx_actions,
                string responses,
                string card_types,
                string filter_type,
                string filter_search,
                integer limit)
)
```

2.6.5.Get Transaction Details

The following SOAP API function is used to returns transaction associated to a TXID.

Function Name

getTransactionDetails

Example

```
getTransactionDetails (
        array (integer sid,
                string rcode,
                string txid)
)
```

2.6.6.Parameter Definitions

Important: All parameters must be sent with the SOAP request, but only required parameters must have a value. All other values can be sent as an empty String.

Name	Data Type	Required	Description
sid	Integer	✓	Site id (unique identifier for the web site)
rid	Integer	✓	Retailer id (unique identifier for the retailer)
rcode	String	✓	Retailer authentication code
txid	String	✓	Transaction ID (TXID)
tid	String	✓	Merchant Transaction ID (TID) *TID search only, its value must be unique and a maximum of 40 alphanumeric characters are allowed in this field. And unique tid value can be generated through concatenation of marchant name or curreny or any string, sid and currenttimestamp with micro seconds.
from_timestamp	Integer	✓	Start date as timestamp
to_timestamp	Integer	✓	End date as timestamp
tx_actions	String	✓	The action that has been performed by the transaction. Valid values:'PAYMENT', 'PREAUTH', 'SETTLEMENT', 'VOID', 'REFUND', 'CHARGEBACK','RETRIEVAL' (Separate Multiple searches by a comma or leave blank for all)
responses	String	✓	The response of the payment gateway. Valid values: 'APPROVED', 'DECLINED', 'PAYG_ERROR', 'PENDING' (Separate Multiple searches by a comma or leave blank for all)
card_types	String	✓	The payment type that has been used in the transaction. Valid values: 'VISA', 'MASTERCARD', 'ACH', 'V8pay', 'cardOnePlus' (Separate Multiple searches by a comma or leave blank for all)
filter_type	String	✗	The filter_search parameter will search inside the following fields (other requests will be ignored): 'TXID', 'EMAIL', 'LASTNAME', 'IP', 'CARD_NO', 'TID', 'WID', 'REF1', 'REF2', 'REF3', 'REF4', 'ADD_INFO'
filter_search	String	✗	This will search inside the identified filter_type. Filter_type is required if this value is not null.
limit	String	✓	The maximum number of transactions returned. Cannot be higher than 500 and will default to this value if a higher value is passed.

2.6.7.Responses

The response from the server is an array. If the status has the value "OK" the transaction has been successfully processed. The response array has the following structure:

Name	Data Type	Description
status	String	Has the value "OK" for a successful transaction, or "EXC" for a failed transaction
details	Array	The main transaction data *TID search only
txid	String	Transaction ID
sid	Integer	Site id (unique identifier for the web site)
tid	String	Internal Transaction ID
wid	Integer	Internal Operator ID
card_type	String	Payment Type
amount_purchase	Decimal	Total amount of purchase
amount_shipping	Decimal	Shipping cost
currency_code	String	Currency code
card_name	String	Card name
card_no	String	Hashed card number. e.g. "411111******1111"
email	String	Email address
firstname	String	Customers first name
lastname	String	Customers last name
uip	String	Customer IP address
ref1	String	Reference field 1
ref2	String	Reference field 2
ref3	String	Reference field 3
ref4	String	Reference field 4
addinfo	String	Additional Information
site_name	String	Site Name
history	Array	A numerically indexed array of related transactions *TID search only
txid	String	Related Transaction ID
amount	Decimal	Transaction amount
tx_action	String	Transaction action
timestamp	Integer	Timestamp
response	String	Response
comment	String	Comment for the transaction
txs	Array	The transaction ID
txid	String	Transaction ID
sid	Integer	Site id (unique identifier for the web site)
tid	String	Internal Transaction ID
wid	Integer	Internal Operator ID
card_type	String	Payment Type
amount_purchase	Decimal	Total amount of purchase
amount_shipping	Decimal	Shipping cost
currency_code	String	Currency code
card_name	String	Card name
card_no	String	Hashed card number. e.g. "411111******1111"
email	String	Email address
firstname	String	Customers first name
lastname	String	Customers last name
uip	String	Customer IP address
ref1	String	Reference field 1
ref2	String	Reference field 2
ref3	String	Reference field 3
ref4	String	Reference field 4
addinfo	String	Additional Information
site_name	String	Site Name
txhistory	Array	A numerically indexed array of related transactions
txid	String	Related Transaction ID
amount	Decimal	Transaction amount
tx_action	String	Transaction action
timestamp	Integer	Timestamp
response	String	Response
comment	String	Comment for the transaction
txitems	Array	A numerically indexed array of cart items

item_no	Integer	Item number
name	String	Item name
item_desc	String	Description of the item
quantity	Integer	Quantity in transaction
amount_unit	Decimal	Amount per single unit
amount_total	Decimal	Total amount
error	String	If an error has occured, then the status value will equal "EXC" and this array with contain further information
type	String	The error type
sys	String	The system that caused the error (client or server)
msg	String	The error message
info	String	Error information that contains the bank message and bank code
code	String	Paypay89 error code

3. Code Examples

3.1. SOAP Credit Card

3.1.1.Single Transaction [Non-Secure3D OR Secure3D] (php + nusoap)

PHP

```php
<?php
    require_once("nusoap.php");
    $client = new nusoap_client("https://admin.paypay89.com/soap//tx3.php?wsdl", true);
    $sid = 1;
    $rcode= "i7u5J9";
    $udetails = array(
        "username" => "",
        "password" => "",
        "firstname" => "John",
        "lastname" => "Smith",
        "email" => "john@email.com",
        "phone" => "0777777770",
        "mobile" => "0410827364",
        "address" => "13/45 Sydney St",
        "suburb_city" => "some suburb",
        "state" => "QLD",
        "postcode" => "40000",
        "country" => "AU",
        "ship_firstname" => "James",
        "ship_lastname" => "Collins",
        "ship_address" => "45/13 Melbourne St",
        "ship_suburb_city"=> "another suburb",
        "ship_state" => "NSW",
        "ship_postcode" => "50000",
        "ship_country" => "AU",
        "bank_name" => "",
        "bank_phone" => "",
        "ssn" => "",
        "dl" => "",
        "dob" => "",
        "uip" => "123.123.123.123"
    );

    $paydetails = array(
        "payby" => "visa",
        "card_name" => "John Smith",
        "card_no" => "4242424242424242",
        "card_ccv" => "987",
        "card_exp_month" => "10",
        "card_exp_year" => "2006",
        "md" => $_SESSION["id"],
        "redirecturl" => "http://www.mywebsite.com/responseiframe.php",
        "useragent" => $_SERVER["HTTP_USER_AGENT"],
        "browseragent" => $_SERVER["HTTP_ACCEPT"],
        "routing_no" => "",
        "account_no" => "",
        "type" => "",
        "regulation_e" => "",
        "class" => "",
        "receive_name" => ""
    );

    $txparams = array(
        "wid" => "HGTRHJIGS585v536",
        "tid" => "o3tf3toct9ctb9ctr9",
        "ref1" => "ref value",
        "ref2" => "ref value",
        "ref3" => "ref value",
        "ref4" => "ref value",
        "addinfo" => "addinfo value",
        "cmd" => "",
        "postbackurl" => "https://postback.payment.com/postback.php"
    );

    $cart = array(
        "items" => array(
            array(
                "name"=>"printer",
                "quantity"=> 2,
                "amount_unit"=>"150.00",
                "item_no"=>"XY2344",
                "item_desc"=>"XY Color Printer C123"
            ),
            array(
                "name"=>"mouse",
                "quantity"=> 1,
                "amount_unit"=>"14.50",
                "item_no"=>"XY9876",
```

```
                "item_desc"=>"XY optical mouse M12"
            )
        ),
        "summary" => array(
                    "quantity"=> 3,
                    "amount_purchase"=>"314.50",
                    "amount_shipping"=>"10.00",
                    "currency_code" => "USD"
                )
    );
    $param = array(
        "sid" => $sid,
        "rcode" => $rcode,
        "udetails" => $udetails,
        "paydetails" => $paydetails,
        "cart" => $cart,
        "txparams" => $txparams
    );
    $response = $client->call("processPayment", $param);
?>
```

3.1.2.Single Transaction [Non-Secure3D OR Secure3D] (PHP soap)

PHP

```php
<?php
    $client = new SoapClient("https://admin.paypay89.com/soap//tx3.php?wsdl");
    $sid = 1;
    $rcode= "i7u5J9";
    $udetails = array(
        "username" => "",
        "password" => "",
        "firstname" => "John",
        "lastname" => "Smith",
        "email" => "john@email.com",
        "phone" => "0777777770",
        "mobile" => "0410827364",
        "address" => "13/45 Sydney St",
        "suburb_city" => "some suburb",
        "state" => "QLD",
        "postcode" => "40000",
        "country" => "AU",
        "ship_firstname" => "James",
        "ship_lastname" => "Collins",
        "ship_address" => "45/13 Melbourne St",
        "ship_suburb_city"=> "another suburb",
        "ship_state" => "NSW",
        "ship_postcode" => "50000",
        "ship_country" => "AU",
        "bank_name" => "",
        "bank_phone" => "",
        "ssn" => "",
        "dl" => "",
        "dob" => "",
        "uip" => "123.123.123.123"
    );

    $paydetails = array(
        "payby" => "visa",
        "card_name" => "John Smith",
        "card_no" => "4242424242424242",
        "card_ccv" => "987",
        "card_exp_month" => "10",
        "card_exp_year" => "2006",
        "md" => $_SESSION["id"],
        "redirecturl" => "http://www.mywebsite.com/responseiframe.php",
        "useragent" => $_SERVER["HTTP_USER_AGENT"],
        "browseragent" => $_SERVER["HTTP_ACCEPT"],
        "routing_no" => "",
        "account_no" => "",
        "type" => "",
        "regulation_e" => "",
        "class" => "",
        "receive_name" => ""
    );

    $txparams = array(
        "wid" = "HGTRHJIGS585v536",
        "tid" = "o3tf3toct9ctb9ctr9",
        "ref1" => "ref value",
        "ref2" => "ref value",
        "ref3" => "ref value",
        "ref4" => "ref value",
        "addinfo" => "addinfo value",
        "cmd" => "",
        "postbackurl" => "https://postback.payment.com/postback.php"
    );

    $cart = array(
        "items" => array(
            array(
                "name"=>"printer",
                "quantity"=> 2,
                "amount_unit"=>"150.00",
                "item_no"=>"XY2344",
                "item_desc"=>"XY Color Printer C123"
            ),
            array(
                "name"=>"mouse",
                "quantity"=> 1,
                "amount_unit"=>"14.50",
                "item_no"=>"XY9876",
                "item_desc"=>"XY optical mouse M12"
```

```
            )
        ),
        "summary" => array(
                    "quantity"=> 3,
                    "amount_purchase"=>"314.50",
                    "amount_shipping"=>"10.00",
                    "currency_code" => "USD"
                )
    );
    $param = array(
        "sid" => $sid,
        "rcode" => $rcode,
        "udetails" => $udetails,
        "paydetails" => $paydetails,
        "cart" => $cart,
        "txparams" => $txparams
    );

    $response = $client->__soapCall("processPayment", $param);
?>
```

3.2. SOAP Peer-to-Peer(P2P)

3.2.1.Single Transaction-P2P(php + nusoap)

PHP

```
<?php
require_once("nusoap.php");
$client = new nusoap_client("https://admin.paypay89.com/soap//tx3.php?wsdl", true);
$sid = 1;
$rcode= "i7u5J9";
$udetails = array(
"email" => "john@email.com",
"uip" => "123.123.123.123"
);


$paydetails = array( "payby" => "p2p",
);


$txparams = array(
"wid" => "HGTRHJIGS585v536",
"tid" => "o3tf3toct9ctb9ctr9", "ref1" => "ref value",
"ref2" => "ref value", "ref3" => "ref value", "ref4" => "ref value",
"addinfo" => "addinfo value", "cmd" => "",
"postbackurl" => "https://postback.payment.com/postback.php",
"successurl" => "https://postback.payment.com/success.php",
"failureurl" => "https://postback.payment.com/failure.php"

);


$cart = array( "items" => array(
array(
"name"=>"printer", "quantity"=> 2, "amount_unit"=>"150.00", "item_no"=>"XY2344",
"item_desc"=>"XY Color Printer C123"
),
array(
"name"=>"mouse", "quantity"=> 1, "amount_unit"=>"14.50", "item_no"=>"XY9876",
"item_desc"=>"XY optical mouse M12"
)

),
"summary" => array(
"quantity"=> 3, "amount_purchase"=>"314.50", "amount_shipping"=>"10.00", "currency_code" => "USD"
)
);
$param = array( "sid" => $sid,
"rcode" => $rcode, "udetails" => $udetails, "paydetails" => $paydetails, "cart" => $cart,
"txparams" => $txparams
);
$response = $client->call("processPayment", $param); ?>
```

3.2.2.Single Transaction-P2P (PHP soap)

PHP

```
<?php
$client = new SoapClient("https://admin.paypay89.com/soap//tx3.php?wsdl");
$sid = 1;
$rcode= "i7u5J9";
$udetails = array(
"email" => "john@email.com",
"uip" => "123.123.123.123"
);


$paydetails = array( "payby" => "p2p",
);


$txparams = array(
"wid" = "HGTRHJIGS585v536",
"tid" = "o3tf3toct9ctb9ctr9", "ref1" => "ref value",
"ref2" => "ref value", "ref3" => "ref value", "ref4" => "ref value",
"addinfo" => "addinfo value", "cmd" => "",
"postbackurl" => "https://postback.payment.com/postback.php",
"successurl" => "https://postback.payment.com/success.php",
"failureurl" => "https://postback.payment.com/failure.php"
);


$cart = array( "items" => array(
array(
"name"=>"printer", "quantity"=> 2, "amount_unit"=>"150.00", "item_no"=>"XY2344",
"item_desc"=>"XY Color Printer C123"
),
array(
"name"=>"mouse", "quantity"=> 1, "amount_unit"=>"14.50", "item_no"=>"XY9876",
"item_desc"=>"XY optical mouse M12"
)
),
"summary" => array(
"quantity"=> 3, "amount_purchase"=>"314.50", "amount_shipping"=>"10.00", "currency_code" => "USD"
)
);
$param = array( "sid" => $sid,
"rcode" => $rcode, "udetails" => $udetails, "paydetails" => $paydetails, "cart" => $cart,
"txparams" => $txparams
);


$response = $client-> soapCall("processPayment", $param);
?>
```

3.2.3.Payout Transaction-P2P (php + nusoap)

PHP

```php
<?php
require_once("nusoap.php");
$client = new nusoap_client("https://admin.paypay89.com/soap//tx3.php?wsdl", true);
$sid = 1;
$rcode= "i7u5J9";
$udetails = array(
"email" => "john@email.com",
"uip" => "123.123.123.123"
);


$payoutdetails = array( "payby" => "p2p",
);


$txparams = array(
"wid" => "HGTRHJIGS585v536",
"tid" => "o3tf3toct9ctb9ctr9", "ref1" => "ref value",
"ref2" => "ref value", "ref3" => "ref value", "ref4" => "ref value",
"addinfo" => "addinfo value", "cmd" => "",
"postbackurl" => "https://postback.payment.com/postback.php",
"successurl" => "https://postback.payment.com/success.php",
"failureurl" => "https://postback.payment.com/failure.php"

);


$cart = array( "items" => array(
array(
"name"=>"printer", "quantity"=> 2, "amount_unit"=>"150.00", "item_no"=>"XY2344",
"item_desc"=>"XY Color Printer C123"
),
array(
"name"=>"mouse", "quantity"=> 1, "amount_unit"=>"14.50", "item_no"=>"XY9876",
"item_desc"=>"XY optical mouse M12"
)
),
"summary" => array(
"quantity"=> 3,
"amount_purchase"=>"314.50",
"amount_shipping"=>"10.00",
"currency_code" => "USD"
)
);
$param = array( "sid" => $sid,
"rcode" => $rcode,
"udetails" => $udetails,
"payoutdetails" => $payoutdetails,
"cart" => $cart,
"txparams" => $txparams
);
$response = $client->call("processPayout", $param);
?>
```

3.2.4.Single Transaction-P2P (PHP soap)

PHP

```php
<?php
$client = new SoapClient("https://admin.paypay89.com/soap//tx3.php?wsdl");
$sid = 1;
$rcode= "i7u5J9";
$udetails = array(
"email" => "john@email.com",
"uip" => "123.123.123.123"
);


$payoutdetails = array( "payby" => "p2p",
);


$txparams = array(
"wid" = "HGTRHJIGS585v536",
"tid" = "o3tf3toct9ctb9ctr9", "ref1" => "ref value",
"ref2" => "ref value", "ref3" => "ref value", "ref4" => "ref value",
"addinfo" => "addinfo value", "cmd" => "",
"postbackurl" => "https://postback.payment.com/postback.php",
"successurl" => "https://postback.payment.com/success.php",
"failureurl" => "https://postback.payment.com/failure.php"
);


$cart = array( "items" => array(
array(
"name"=>"printer", "quantity"=> 2, "amount_unit"=>"150.00", "item_no"=>"XY2344",
"item_desc"=>"XY Color Printer C123"
),
array(
"name"=>"mouse", "quantity"=> 1, "amount_unit"=>"14.50", "item_no"=>"XY9876",
"item_desc"=>"XY optical mouse M12"
)
),
"summary" => array(
"quantity"=> 3,
"amount_purchase"=>"314.50",
"amount_shipping"=>"10.00",
"currency_code" => "USD"
)
);
$param = array( "sid" => $sid,
"rcode" => $rcode,
"udetails" => $udetails,
"payoutdetails" => $payoutdetails,
"cart" => $cart,
"txparams" => $txparams
);


$response = $client-> soapCall("processPayout", $param);
?>
```

3.3. SOAP ACH

3.3.1.Single Transaction (php + nusoap)

PHP

Please note that we recommend use of the latest CVS version of nusoap.

```php
<?php
    require_once("nusoap.php");
    $client = new soapclient("https://admin.paypay89.com/soap//tx3.php?wsdl", true);
    $sid = 1;
    $rcode= "i7u5J9";
    $udetails = array(
        "username" => "",
        "password" => "",
        "firstname" => "John",
        "lastname" => "Smith",
        "email" => "john@email.com",
        "phone" => "0777777770",
        "mobile" => "0410827364",
        "address" => "13/45 Sydney St",
        "suburb_city" => "some suburb",
        "state" => "QLD",
        "postcode" => "40000",
        "country" => "AU",
        "ship_firstname" => "James",
        "ship_lastname" => "Collins",
        "ship_address" => "45/13 Melbourne St",
        "ship_suburb_city"=> "another suburb",
        "ship_state" => "NSW",
        "ship_postcode" => "50000",
        "ship_country" => "AU",
        "bank_name" => "",
        "bank_phone" => "",
        "ssn" => "",
        "dl" => "",
        "dob" => "",
        "uip" => "123.123.123.123"
    );

    $paydetails = array(
        "payby" => "ach",
        "card_name" => "",
        "card_no" => "",
        "card_ccv" => "",
        "card_exp_month" => "",
        "card_exp_year" => "",
        "md" => "",
        "redirecturl" => "",
        "useragent" => "",
        "browseragent" => "",
        "routing_no" => "123456789",
        "account_no" => "12345678",
        "type" => "1",
        "regulation_e" => "1",
        "class" => "WEB",
        "receive_name" => "Jane Doe"
    );

    $txparams = array(
        "wid" = "HGTRHJIGS585v536",
        "tid" = "o3tf3toct9ctb9ctr9",
        "ref1" => "ref value",
        "ref2" => "ref value",
        "ref3" => "My Description",
        "ref4" => "ref value",
        "addinfo" => "addinfo value",
        "cmd" => "",
        "postbackurl" => "https://postback.payment.com/postback.php"
    );

    $cart = array(
    "items" => array(
        array(
            "name"=>"printer",
            "quantity"=> 2,
            "amount_unit"=>"150.00",
            "item_no"=>"XY2344",
            "item_desc"=>"XY Color Printer C123"
        ),
        array(
            "name"=>"mouse",
            "quantity"=> 1,
            "amount_unit"=>"14.50",
            "item_no"=>"XY9876",
```

```
            "item_desc"=>"XY optical mouse M12" ) ),
            "summary" => array(
            "quantity"=> 3,
            "amount_purchase"=>"314.50",
            "amount_shipping"=>"10.00",
            "currency_code" => "USD"
        )
    );

    $param = array(
        "sid" => $sid,
        "rcode" => $rcode,
        "udetails" => $udetails,
        "paydetails" => $paydetails,
        "cart" => $cart,
        "txparams" => $txparams
    );

    $response = $client->call("processPayment", $param);
?>
```

3.4. SOAP Card Load

3.4.1.Single Card Load (php + nusoap)

PHP

```php
<?php
    require_once("nusoap.php");
    $client = new soapclient("https://admin.paypay89.com/soap//tx3.php?wsdl", true);
    $sid = 1;
    $rcode= "i7u5J9";
    $udetails = array(
        "username" => "",
        "password" => "",
        "firstname" => "John",
        "lastname" => "Smith",
        "email" => "john@email.com",
        "phone" => "0777777770",
        "mobile" => "0410827364",
        "address" => "13/45 Sydney St",
        "suburb_city" => "some suburb",
        "state" => "QLD",
        "postcode" => "40000",
        "country" => "AU",
        "ship_firstname" => "James",
        "ship_lastname" => "Collins",
        "ship_address" => "45/13 Melbourne St",
        "ship_suburb_city"=> "another suburb",
        "ship_state" => "NSW",
        "ship_postcode" => "50000",
        "ship_country" => "AU",
        "bank_name" => "",
        "bank_phone" => "",
        "ssn" => "",
        "dl" => "",
        "dob" => "",
        "uip" => "123.123.123.123"
    );

    $paydetails = array(
        "payby" => "visa",
        "card_name" => "John Smith",
        "card_no" => "4242424242424242",
        "card_ccv" => "987",
        "card_exp_month" => "10",
        "card_exp_year" => "2006",
        "md" => $_SESSION["id"],
        "redirecturl" => "http://www.mywebsite.com/responseiframe.php",
        "useragent" => $_SERVER["HTTP_USER_AGENT"],
        "browseragent" => $_SERVER["HTTP_ACCEPT"],
        "routing_no" => "",
        "account_no" => "",
        "type" => "",
        "regulation_e" => "",
        "class" => "",
        "receive_name" => ""
    );

    $txparams = array(
        "wid" = "HGTRHJIGS585v536",
        "tid" = "o3tf3toct9ctb9ctr9",
        "ref1" => "ref value",
        "ref2" => "ref value",
        "ref3" => "ref value",
        "ref4" => "ref value",
        "addinfo" => "addinfo value",
        "cmd" => "",
        "postbackurl" => "https://postback.payment.com/postback.php"
    );

    $cart = array(
        "summary" => array(
            "amount_purchase"=>"314.50",
            "currency_code" => "USD"
        )
    );
    $param = array(
        "sid" => $sid,
        "rcode" => $rcode,
        "udetails" => $udetails,
        "paydetails" => $paydetails,
        "cart" => $cart,
        "txparams" => $txparams
    );
```

```
    $response = $client->call("processLoad", $param);
?>
```

3.5. SOAP Refunds

3.5.1.Refund (php)

PHP

```php
<?php
    require_once("nusoap.php");
    $client = new soapclient("https://admin.paypay89.com/soap//tx3.php?wsdl", true);
        $param = array(
        "sid" => "1",
        "rcode" => "i7u5J9",
        "txid" => "1148437566168394",
        "reason" => "customer request",
        "amount" => "23.00",
        "postbackurl" => "https://admin.mydomain.com/postback.php"
        "sendNotification" => "1"
    );
    $response = $client->call("processRefund", $param);
?>
```

3.6. Secure Hosted Payment

3.6.1.Single Transaction-P2P(HTML)

HTML & PHP

```
<form action="https://secure.paypay89.com/" method="post">
sid: <input type="text" name="sid" value="19"><BR>
wid: <input type="text" name="wid" value="1234"><BR> tid: <input type="text" name="tid" value="5678"><BR>
email: <input type="text" name="email" value="jack@testing.com"><BR>
card_type: <input type=”text” name=”card_type” value=”p2p”><BR>
successurl: <input type="text" name="successurl" value="http://example.com/successurl"><BR>
failureurl: <input type="text" name="failureurl" value="http://example.com/failureurl"><BR>
<table>

<tr>
<td>
item_quantity: <input type="text" name="item_quantity[]" value="2">
item_name: <input type="text" name="item_name[]" value="apple">
item_no: <input type="text" name="item_no[]" value="a234">
item_desc: <input type="text" name="item_desc[]" value="juicy green apple">
item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.59">
</td>


<td>
item_quantity: <input type="text" name="item_quantity[]" value="2">
item_name: <input type="text" name="item_name[]" value="apple">
item_no: <input type="text" name="item_no[]" value="a234">
item_desc: <input type="text" name="item_desc[]" value="juicy green apple">
item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.59">
</td>

<td>
item_quantity: <input type="text" name="item_quantity[]" value="1">
item_name: <input type="text" name="item_name[]" value="pear">
item_no: <input type="text" name="item_no[]" value="p567">
item_desc: <input type="text" name="item_desc[]" value="fresh green pear">
 item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.68">
</td>
</tr>
</table>
<input type="submit" value=" post to https://secure.paypay89.com">
</form>
```

3.6.2.P2P Deposit - Server To Server

3.6.2.1.Environments

QA (Sandboxed) for testing : https://secure.paypay89.com/txHandler.php

Test transactions only
Request a SID for this environment
Whitelisting of caller IP required. Please send your IPS to Paypay89 for this call.

Production - URL: https://secure.paypay89.com/txHandler.php
Live accounts
Whitelisting of caller IP required. Please send your IPS to Paypay89 for this call.

3.6.2.2.Request Details and Flow

Request Details
Request Payload:

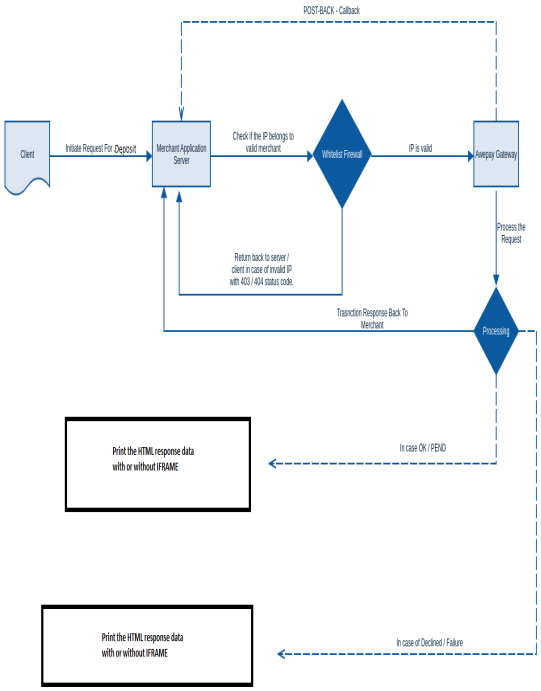
```
{
                sid: '40',
                firstname: 'Paypay89',
                lastname:  'TESTER',
                tid: 'AWPMYR16045602355521', // merchant side generated aplanumeric string , it should be unique for each transaction
                payby: 'p2p',
                tx_action: 'PAYMENT',
                currency: 'MYR',
                bank_code: 'CIMB.MY', // pass bank code if needs to redirect bank login page
                no_ship_address: '1',
                hide_descriptor: '1',
                successurl: 'paste your successurl',
                failureurl: 'paste your failureurl ',
                postbackurl: 'past your postbackurl',
                item_quantity[]: '1',
                item_name[]:'test',
                item_no[]:'1',
                item_desc[]:'test1234',
                item_amount_unit[]:'1'
}
```

Request Method: **POST**

Headers:

```
{ 'Content-Type' : 'application/x-www-form-urlencoded' }
```

Request Flow



PHP Server Example

```
$url = "Paypay89 Gateway URL for Deposit";
$postdata = $_POST; // data posted in the request

curl_setopt_array($curl, array(
  CURLOPT_URL => $url,
  CURLOPT_RETURNTRANSFER => true,
  CURLOPT_ENCODING => "",
  CURLOPT_TIMEOUT => 60,
  CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
  CURLOPT_CUSTOMREQUEST => "POST",
  CURLOPT_POSTFIELDS => http_build_query($postdata),
  CURLOPT_HTTPHEADER => array(
    "content-type: application/x-www-form-urlencoded"
  ),
));
$response = curl_exec($curl);
$err = curl_error($curl);

curl_close($curl);

if ($err) {
  echo "cURL Error #:" . $err;
} else {
  echo $response;
}
```

3.6.3.P2P Crypto Currency Deposit - Server To Server

3.6.3.1.Environments

QA (Sandboxed) for testing : https://secure.paypay89.com/txHandler.php

Test transactions only
Request a SID for this environment
Whitelisting of caller IP required. Please send your IPS to Paypay89 for this call.

Production - URL: https://secure.paypay89.com/txHandler.php
Live accounts
Whitelisting of caller IP required. Please send your IPS to Paypay89 for this call.

3.6.3.2.Request Details and Flow

Request Details
Request Payload:

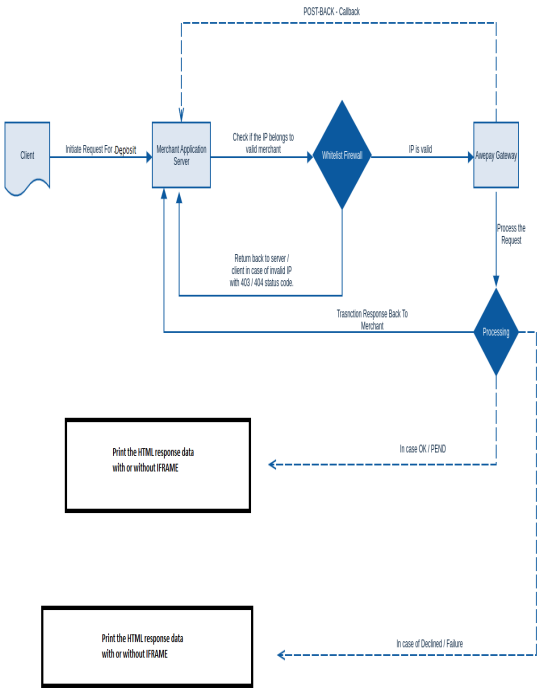
```
{
        sid: '3665',
        firstname: 'Mahabubur',
        lastname:  'Tester',
        email:  'mahabubur@email.com',
        tid: 'AWPMYR16045602355521', // merchant side generated aplanumeric string , it should be unique for each transaction
        payby: 'p2p',
        tx_action: 'PAYMENT',
        currency: 'USD',
        phone: '0111234567',
        address: '123 address',
        suburb_city: 'city',
        country: 'CA',
        postcode: '41010',
        no_ship_address: '1',
        no_address: '1',
        hide_descriptor: '1',
        successurl: 'paste your successurl',
        failureurl: 'paste your failureurl ',
        postbackurl: 'past your postbackurl',
        item_quantity[]: '1',
        item_name[]:'test',
        item_no[]:'1',
        item_amount_unit[]:'1.00'
}
```

Request Method: **POST**

Headers:

```
{ 'Content-Type' : 'application/x-www-form-urlencoded' }
```

Request Flow



PHP Server Example

```
$url = "Paypay89 Gateway URL for Deposit";
$postdata = $_POST; // data posted in the request

curl_setopt_array($curl, array(
  CURLOPT_URL => $url,
  CURLOPT_RETURNTRANSFER => true,
  CURLOPT_ENCODING => "",
  CURLOPT_TIMEOUT => 60,
  CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
  CURLOPT_CUSTOMREQUEST => "POST",
  CURLOPT_POSTFIELDS => http_build_query($postdata),
  CURLOPT_HTTPHEADER => array(
    "content-type: application/x-www-form-urlencoded"
  ),
));
$response = curl_exec($curl);
$err = curl_error($curl);

curl_close($curl);

if ($err) {
  echo "cURL Error #:" . $err;
} else {
  echo $response;
}
```

3.6.4.P2P Payout - Server To Server

3.6.4.1.Environments

QA (Sandboxed) for testing : https://secure.paypay89.com/txHandlerPayout.php

Test transactions only
Request a SID for this environment
Whitelisting of caller IP required. Please send your IPS to Paypay89 for this call.

Production - URL: https://secure.paypay89.com/txHandlerPayout.php
Live accounts
Whitelisting of caller IP required. Please send your IPS to Paypay89 for this call.

3.6.4.2.Request Details and Flow

Request Details
Request Payload:

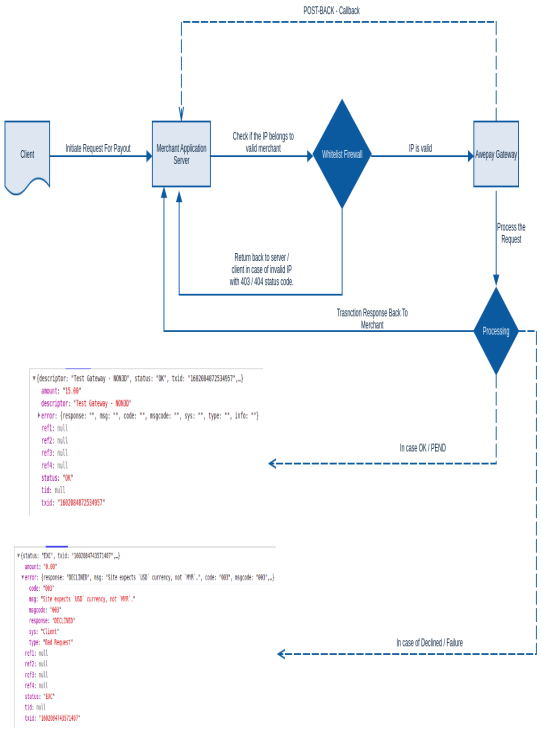
```
{
        sid: '31',
        card_type: 'p2p',
        tid: 'DPMYR1604560234444',  // merchant side generated aplanumeric string , it should be unique for each transaction
        tx_action: 'PAYOUT',
        firstname: 'Payout',
        lastname: 'TESTER',
        city: 'city',
        email: 'test@test.com',
        amount: '15.00',
        currency: 'MYR',
        bank_code: 'CIMB',
        account_name: 'cardname',
        account_number: '1234756789',
        bank_province: 'ICBC',
        bank_city: 'ICBC',
        postback_url: 'https://yourpostbackURL.com'
}
```

Request Method: **POST**

Headers:

```
{ 'Content-Type' : 'application/x-www-form-urlencoded' }
```

Request Flow



PHP Server Example

```php
$url = "Paypay89 Gateway URL for PAYOUTs";
$data = $_POST; // data posted in the request


$curl = curl_init($url);
curl_setopt($curl, CURLOPT_POST, true);
curl_setopt($curl, CURLOPT_POSTFIELDS, http_build_query($_POST));
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
curl_setopt($curl, CURLOPT_HTTPHEADER, array('Content-Type: application/x-www-form-urlencoded'));
$response = curl_exec($curl); // getting the response
$http_status = curl_getinfo($curl, CURLINFO_HTTP_CODE); // getting the RESPONSE-HTTP-STATUS
echo $response;
// decode the json response using json_decode($response); function and use it in your app logic.
curl_close($curl);
```

Java Server Example

```java
OkHttpClient client = new OkHttpClient().newBuilder().build();
MediaType mediaType = MediaType.parse("application/x-www-form-urlencoded");
RequestBody body = RequestBody.create(mediaType, "sid=31&postback_url=https://qa.admin.paypay89.com/insights/postback.php&card_type=p
Request request = new Request.Builder()
  .url(https://secure.paypay89.com/txHandlerPayout.php)
  .method("POST", body)
  .addHeader("Content-Type", "application/x-www-form-urlencoded")
  .addHeader("Cookie", "PHPSESSID=27r542c0ra810ud0njl4n8tt3d")
  .build();
Response response = client.newCall(request).execute();
```

NodeJs Example

```nodejs
var axios = require('axios');
var qs = require('qs');
var data = qs.stringify({
 'sid': '31',
'postback_url': 'https://qa.admin.paypay89.com/insights/postback.php',
'card_type': 'p2p',
'firstname': 'Paypay89',
'lastname': 'tester',
'tx_action': 'PAYOUT',
'amount': '1.00',
'currency': 'MYR',
'account_name': 'Paypay89',
'account_number': '23453534534',
'bank_code': 'CIMB',
'bank_province': 'ICBC',
'bank_city': 'ICBC',
'city': 'KL'
});
var config = {
  method: 'post',
  url: https://secure.paypay89.com/txHandlerPayout.php,
  headers: {
    'Content-Type': 'application/x-www-form-urlencoded',
  },
  data : data
};

axios(config)
.then(function (response) {
  console.log(JSON.stringify(response.data));
})
.catch(function (error) {
  console.log(error);
});
```

C# Example

```
var client = new RestClient(https://secure.paypay89.com/txHandlerPayout.php);
client.Timeout = -1;
var request = new RestRequest(Method.POST);
request.AddHeader("Content-Type", "application/x-www-form-urlencoded");
request.AddParameter("sid", "31");
request.AddParameter("postback_url", "https://qa.admin.paypay89.com/insights/postback.php");
request.AddParameter("card_type", "p2p");
request.AddParameter("firstname", "Paypay89");
request.AddParameter("lastname", "tester");
request.AddParameter("tx_action", "PAYOUT");
request.AddParameter("amount", "1.00");
request.AddParameter("currency", "MYR");
request.AddParameter("account_name", "Paypay89");
request.AddParameter("account_number", "23453534534");
request.AddParameter("bank_code", "CIMB");
request.AddParameter("bank_province", "ICBC");
request.AddParameter("bank_city", "ICBC");
request.AddParameter("city", "KL");
IRestResponse response = client.Execute(request);
Console.WriteLine(response.Content);
```

Possible Responses

Success / Pending
Response Http Status Code: **201**
Response Body:

```
{"descriptor":"Test Gateway - NON3D","status":"OK",
                        "txid":"1602084872534957",
                        "error":{"response":"","msg":"","code":"","msgcode":"","sys":"","type":"","info":""},
                    "amount":"15.00","tid":null,"ref1":null,"ref2":null,"ref3":null,"ref4":null
                        }
```

Declined / Failure
Response Http Status Code: **400**
Response Body:

```
                        {"status":"EXC","txid":"1602086024466192",
                        "error":{"response":"DECLINED","msg":"Site expects `USD` currency, not `MYR`.",
                        "code":"003","msgcode":"003","sys":"Client","type":"Bad Request"},
                        "amount":"0.00","tid":null,"ref1":null,"ref2":null,"ref3":null,"ref4":null
                        }
```

Server Failures
Response Http Status Code: **500**

IP Not whitelisted
Response Http Status Code: **403**

3.6.5.Single Transaction (HTML)

HTML & PHP

```
<form action="https://secure.paypay89.com/index.php" method="post">
        sid: <input type="text" name="sid" value="19"><BR>
        wid: <input type="text" name="wid" value="1234"><BR>
        tid: <input type="text" name="tid" value="5678"><BR>
        firstname: <input type="text" name="firstname" value="Jack"><BR>
        lastname: <input type="text" name="lastname" value="Tester"><BR>
        email: <input type="text" name="email" value="jack@testing.com"><BR>
        phone: <input type="text" name="phone" value="1234567"><BR>
        address: <input type="text" name="address" value="12 Test Lane"><BR>
        suburb_city: <input type="text" name="suburb_city" value="TestCity"><BR>
        state: <input type="text" name="state" value="CA-ON"><BR>
        postcode: <input type="text" name="postcode" value="41010"><BR>
        country: <input type="text" name="country" value="CA"><BR>
        ship_firstname: <input type="text" name="ship_firstname" value="Julie"><BR>
        ship_lastname: <input type="text" name="ship_lastname" value="Tester"><BR>
        ship_address: <input type="text" name="ship_address" value="13 Test Lane"><BR>
        ship_suburb_city: <input type="text" name="ship_suburb_city" value="TestCity"><BR>
        ship_state: <input type="text" name="ship_state" value="CA-ON"><BR>
        ship_postcode: <input type="text" name="ship_postcode" value="41010"><BR>
        ship_country: <input type="text" name="ship_country" value="CA"><BR>
        card_name: <input type="text" name="card_name" value="_test_"><BR>
        card_no: <input type="text" name="card_no" value="4111111111111111"><BR>
        card_ccv: <input type="text" name="card_ccv" value="123"><BR>
        card_exp_month: <input type="text" name="card_exp_month" value="10"><BR>
        card_exp_year: <input type="text" name="card_exp_year" value="2007"><BR><BR>
        shipping: <input type="text" name="shipping" value="1"><BR>
        amount_shipping: <input type="text" name="amount_shipping" value="0.30"><BR>
        successurl: <input type="text" name="successurl" value="http://example.com/successurl"><BR>
        failureurl: <input type="text" name="failureurl" value="http://example.com/failureurl"><BR>
        <table>
                <tr>
                        <td>
                        item_quantity: <input type="text" name="item_quantity[]" value="2">
                        item_name: <input type="text" name="item_name[]" value="apple">
                        item_no: <input type="text" name="item_no[]" value="a234">
                        item_desc: <input type="text" name="item_desc[]" value="juicy green apple">
                        item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.59">
                        </td>
                </tr>
                <tr>
                        <td>
                        item_quantity: <input type="text" name="item_quantity[]" value="1">
                        item_name: <input type="text" name="item_name[]" value="pear">
                        item_no: <input type="text" name="item_no[]" value="p567">
                        item_desc: <input type="text" name="item_desc[]" value="fresh green pear">
                        item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.68">
                        </td>
                </tr>
        </table>
        <input type="submit" value=" post to https://secure.paypay89.com ">
</form>
```

3.7. TxHandler

3.7.1.Single Transaction (HTML)

HTML

```
<form action="https://secure.paypay89.com/txhandler.php" method="post">
        sid: <input type="text" name="sid" value="19"><BR>
        wid: <input type="text" name="wid" value="1234"><BR>
        tid: <input type="text" name="tid" value="5678"><BR>
        postback url: <input type="text" name="postback_url" value="http://mysite.com/postback.php"><BR>
        card type: <input type="text" name="card_type" value="visa"><BR>
        firstname: <input type="text" name="firstname" value="Jack"><BR>
        lastname: <input type="text" name="lastname" value="Tester"><BR>
        email: <input type="text" name="email" value="jack@testing.com"><BR>
        phone: <input type="text" name="phone" value="1234567"><BR>
        address: <input type="text" name="address" value="12 Test Lane"><BR>
        suburb_city: <input type="text" name="suburb_city" value="TestCity"><BR>
        state: <input type="text" name="state" value="CA-ON"><BR>
        postcode: <input type="text" name="postcode" value="41010"><BR>
        country: <input type="text" name="country" value="CA"><BR>
        ship_firstname: <input type="text" name="ship_firstname" value="Julie"><BR>
        ship_lastname: <input type="text" name="ship_lastname" value="Tester"><BR>
        ship_address: <input type="text" name="ship_address" value="13 Test Lane"><BR>
        ship_suburb_city: <input type="text" name="ship_suburb_city" value="TestCity"><BR>
        ship_state: <input type="text" name="ship_state" value="CA-ON"><BR>
        ship_postcode: <input type="text" name="ship_postcode" value="41010"><BR>
        ship_country: <input type="text" name="ship_country" value="CA"><BR>
        card_name: <input type="text" name="card_name" value="_test_"><BR>
        card_no: <input type="text" name="card_no" value="4111111111111111"><BR>
        card_ccv: <input type="text" name="card_ccv" value="123"><BR>
        card_exp_month: <input type="text" name="card_exp_month" value="10"><BR>
        card_exp_year: <input type="text" name="card_exp_year" value="2007"><BR><BR>
        shipping: <input type="text" name="shipping" value="1"><BR>
        amount_shipping: <input type="text" name="amount_shipping" value="0.30"><BR>
        successurl: <input type="text" name="successurl" value="http://example.com/successurl"><BR>
        failureurl: <input type="text" name="failureurl" value="http://example.com/failureurl"><BR>
        <table>
                <tr>
                        <td>
                        item_quantity: <input type="text" name="item_quantity[]" value="2">
                        item_name: <input type="text" name="item_name[]" value="apple">
                        item_no: <input type="text" name="item_no[]" value="a234">
                        item_desc: <input type="text" name="item_desc[]" value="juicy green apple">
                        item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.59">
                        </td>
                </tr>
                <tr>
                        <td>
                        item_quantity: <input type="text" name="item_quantity[]" value="1">
                        item_name: <input type="text" name="item_name[]" value="pear">
                        item_no: <input type="text" name="item_no[]" value="p567">
                        item_desc: <input type="text" name="item_desc[]" value="fresh green pear">
                        item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.68">
                        </td>
                </tr>
        </table>
        <input type="submit" value=" post to https://secure.paypay89.com/txhandler ">
</form>
```

3.7.2.Single Transaction-P2P (HTML)

HTML

```
<form action="https://secure.paypay89.com/txHandler.php" method="post"> sid: <input type="text" name="sid" value="19"><BR>
wid: <input type="text" name="wid" value="1234"><BR> tid: <input type="text" name="tid" value="5678"><BR>
postback url: <input type="text" name="postback_url" value="http://mysite.com/postback.php"><BR>
 card type: <input type="text" name="card_type" value="p2p"><BR>
email: <input type="text" name="email" value="jack@testing.com"><BR>
successurl: <input type="text" name="successurl" value="http://example.com/successurl"><BR>
 failureurl: <input type="text" name="failureurl" value="http://example.com/failureurl"><BR>
<table>
<tr>
<td>
item_quantity: <input type="text" name="item_quantity[]" value="2">
item_name: <input type="text" name="item_name[]" value="apple"> item_no: <input type="text" name="item_no[]" value="a234">
item_desc: <input type="text" name="item_desc[]" value="juicy green apple">
item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.59">
</td>



<td>
item_quantity: <input type="text" name="item_quantity[]" value="1">
item_name: <input type="text" name="item_name[]" value="pear">
item_no: <input type="text" name="item_no[]" value="p567">
item_desc: <input type="text" name="item_desc[]" value="fresh green pear">
 item_amount_unit: <input type="text" name="item_amount_unit[]" value="0.68">
</td>
</tr>
</table>
</form>
```

3.8. Postbacks

3.8.1.PAYMENT

PHP

```
<?php
var_dump($_POST);
?>

//Example Output
array (
  'txid' => '12345678901234',
  'response' => 'APPROVED',
  'tx_action' => 'PAYMENT',
)
```

3.8.2.REFUND

PHP

```
<?php
var_dump($_POST);
?>

//Example Output
array (
  'txid' => '12345678901234',
  'parent_txid' => '12345678901233',
  'parent_txaction' => 'PAYMENT',
  'response' => 'PENDING',
  'tx_action' => 'REFUND',
  'comment' => 'Merchandise not received',
)
```

3.8.3.CHARGEBACK

PHP

```
<?php
var_dump($_POST);
?>

//Example Output
array (
  'txid' => '12345678901234',
  'parent_txid' => '12345678901233',
  'parent_txaction' => 'PAYMENT',
  'response' => 'APPROVED',
  'tx_action' => 'CHARGEBACK',
  'comment' => 'Unauthorized charge',
)
```

3.8.4.LOAD

PHP

```
<?php
var_dump($_POST);
?>

//Example Output
array (
  'txid' => '12345678901234',
  'response' => 'APPROVED',
  'tx_action' => 'LOAD',
)
```

4. Appendix

4.1. Paypay89 Generic Error Types

Code	Description
001	Declined
002	Paygate Error
003	Bad Request
004	DB Error
005	Internal Error
100	Warning

4.2. Transaction Action Types

Code	Description
PAYMENT	Debit funds from a card or payment method
PAYOUT	Withdrawn money from Merchant site to customers account
PREAUTH	Pre-authorization to debit funds from a card or payment method
SETTLEMENT	Process and debit funds of a successful preauth
LOAD	Credit funds to a card or payment method
VOID	Cancel a pending transaction or preauth
REFUND	Reverse the funds withdrawn for a transaction
FRAUDCHECK	A transactional state where antifraud measures are taken
CHARGEBACK	A customer has requested their bank to return their funds
CHALLENGE	Documentation has been received and submitted for chargeback challenge
REPRESENTMENT	A merchant uses to dispute a chargeback
ARBITRATION	A thirdparty settles the dispute between chargeback and challenge
RETRIEVAL	Customer's issuing bank has formally requested more information on the transaction from the acquiring bank.
REVERSAL	Customer and/or Customer's issuing bank has reversed the chargeback after investigation.

4.3. Transaction Status Types

Code	Description
APPROVED	The transaction was processed successfully
PENDING	The transaction is currently pending further action
DECLINED	The transaction attempt has been declined
PAYG_ERROR	An error has occured in the communication with the bank

4.4. Transaction Specific Error Codes

Code	Description
PAYMENT	
100	Cannot attempt payment against previously processed transaction.
101	Payment is not available for this transaction.
403	The request is blocked.
PAYOUT	
100	Cannot attempt payment against previously processed transaction.
101	Payment is not available for this transaction.
403	The request is blocked.
SETTLEMENT	
400	Cannot settle previously settled transaction.
401	Settlement is not available for this transaction.
402	Cannot settle a voided preauthorization.
REFUND	
500	Cannot refund more than the available non-refunded balance.
501	There is a refund attempt still queued for this transaction.
502	Refund has been queued for processing.
503	Partial refunds are not accepted on this solution.
504	Refund is not available for this transaction.
505	Cannot refund payments/settlements over 180 days old.
506	Cannot refund charged back transaction.
507	Cannot process a refund for $0.00 or a negative amount.
508	Bank only allows 1 refund per transaction
509	Limit of refunds per day exceeded
VOID	
600	Cannot void payments/settlements over 5 days old.
601	Cannot void previously refunded transaction.
602	Cannot void previously voided transaction.
603	Partial voids are not accepted by this solution.
604	Cannot void charged back transaction.
CHARGEBACK	
700	Cannot chargeback previously charged back transaction.
RETRIEVAL	
800	Cannot retrieve previously retrieved transaction.
OTHER	
G00	The gateway is currently unavailable while system maintenance and required updates are performed. This transaction has not been recorded or submitted for processing. Please try again shortly.
G01	%field_description% ("%field_name%") is required by the active solution.
G02	The active solution requires Secure3D to be enabled. Configuration Error. Contact technical support.
3DA	The 3D Secure authentication failed or was not completed.

4.5. Bank Codes

Code	Description
00	successful approval (corresponds to 200 response)
01	refer to issuer
02	refer to issuer's special conditions
03	invalid merchant
04	pickup card
05	do not honour
06	error
07	pickup card, special conditions
08	honour with ID (signature)(corresponds to 200 response)
09	request in progress
10	approved for partial amount
11	approved VIP
12	invalid transaction
13	invalid amount
14	invalid card number
15	no such issuer
16	approved, update track 3
17	customer cancellation
18	customer dispute
19	re-enter transaction
20	invalid response
21	no action taken
22	suspected malfunction
23	unacceptable transaction fee
24	file date not supported
25	unable to locate record on file
26	duplicate file update record, old record replaced
27	file update field error
28	file update file locked out
29	file update not successful, contact acquirer
30	format error
31	bank not supported by switch
32	completed partially
33	expired card
34	suspected fraud
35	contact acquirer
36	restricted card
37	contact acquirer security
38	allowable PIN retries exceeded
39	no credit account
40	request function not supported
41	lost card
42	no universal account
43	stolen card
44	no investment account
45	reserved, will not be returned
46	reserved, will not be returned
47	reserved, will not be returned
48	reserved, will not be returned
49	reserved, will not be returned
50	reserved, will not be returned
51	insufficient funds
52	no cheque account
53	no savings account
54	expired card
55	incorrect PIN
56	no card record
57	transaction not permitted to cardholder

58	transaction not permitted to terminal
59	suspected fraud
60	contact acquirer
61	exceeds withdrawal amount limit
62	restricted card
63	security violation
64	original amount incorrect
65	exceeds withdrawal frequency limit
66	contact acquirer security
67	hard capture
68	response received too late
69	reserved, will not be returned
70	reserved, will not be returned
71	reserved, will not be returned
72	reserved, will not be returned
73	reserved, will not be returned
74	reserved, will not be returned
75	allowable number of PIN retries exceeded
76	reserved, will not be returned
77	reserved, will not be returned
78	reserved, will not be returned
79	reserved, will not be returned
80	reserved, will not be returned
81	reserved, will not be returned
82	reserved, will not be returned
83	reserved, will not be returned
84	reserved, will not be returned
85	reserved, will not be returned
86	reserved, will not be returned
87	reserved, will not be returned
88	reserved, will not be returned
89	reserved, will not be returned
90	cutoff in progress
91	issuer inoperative
92	financial institution cannot be found
93	transaction cannot be completed, violation of law
94	duplicate transmission
95	reconcile error
96	system malfunction
97	reconciliation totals have been reset
98	MAC error
99	reserved, will not be returned

4.6. CVV(2) Response Codes

Code	Description
M	Match
N	No Match
P	Not Processed
X	Cannot Verify (also used as a test response by some processors)
U	Unable To Verify
S	Unavailable For Verification

4.7. AVS Response Codes (Visa)

Code	Description
A	Address matches, ZIP code does not.
B	Street address match for international transaction; postal code not verified.
C	Street & postal code not verified for international transaction.
D	Street & Postal codes match for international transaction. Both the five-digit postal zip code as well as the first five numerical characters contained in the address match for the international transaction.
E	Transaction is ineligible for address verification.
F	Street address & postal codes match for international transaction. (UK Only)
G	AVS not performed because the international issuer does not support AVS.
I	Address information not verified for international transaction.
M	Street address & postal codes match for international transaction.
N	Neither the ZIP nor the address matches.
P	Postal codes match for international transaction; street address not verified.
S	AVS not supported at this time.
R	Issuer's authorization system is unavailable, try again later.
U	Unable to perform address verification because either address information is unavailable or the Issuer does not support AVS.
W	Nine-digit zip match, address does not. The nine-digit postal zip code matches that stored at the VIC or card issuer's center. However, the first five numerical characters contained in the address do not match.
X	Exact match (nine-digit zip and address). Both the nine-digit postal zip code as well as the first five numerical characters contained in the address match.
Y	Address & 5-digit or 9-digit ZIP match.
Z	Either 5-digit or 9-digit ZIP matches, address does not.
0 (zero)	Service Not Allowed. Generally associated with credit cards that are either not allowed to be used for any online transactions or are not allowed to be used for a specific classification of company.

4.8. AVS Response Codes (Mastercard)

Code	Description
A	Address matches, ZIP code does not.
B	Street address match for international transaction; postal code not verified.
C	Street & postal code not verified for international transaction.
D	Street & Postal codes match for international transaction. Both the five-digit postal zip code as well as the first five numerical characters contained in the address match for the international transaction.
E	Ineligible transaction. Address verification not allowed for card type.
F	Street address & postal codes match for international transaction. (UK Only)
G	International Address information unavailable. The international address information was not available at the VIC or issuer's center.
I	Address information not verified for international transaction.
M	Street address & postal codes match for international transaction.
N	Neither the ZIP nor the address matches.
P	Postal codes match for international transaction; street address not verified.
S	AVS not supported at this time.
R	Retry, system unable to process.
U	No data from issuer/BankNet switch.
W	9-digit ZIP code matches, but address does not.
X	Exact, all digits match, 9-digit ZIP code.
Y	Exact, all digits match, 5-digit ZIP code.
Z	5-digit ZIP code matches, but address does not.

4.9. Country ISO Codes

Code	Country
AF	Afghanistan
AX	Aland Islands
AL	Albania
DZ	Algeria
AS	American Samoa
AD	Andorra
AO	Angola
AI	Anguilla
AQ	Antarctica
AG	Antigua and Barbuda
AR	Argentina
AM	Armenia
AW	Aruba
AU	Australia
AT	Austria
AZ	Azerbaijan
BS	Bahamas
BH	Bahrain
BD	Bangladesh
BB	Barbados
BY	Belarus
BE	Belgium
BZ	Belize
BJ	Benin
BM	Bermuda
BT	Bhutan
BO	Bolivia
BA	Bosnia and Herzegovina
BW	Botswana
BV	Bouvet Island
BR	Brazil
VG	British Virgin Islands
IO	British Indian Ocean Territory
BN	Brunei Darussalam
BG	Bulgaria
BF	Burkina Faso
BI	Burundi
KH	Cambodia
CM	Cameroon
CA	Canada
CV	Cape Verde
KY	Cayman Islands
CF	Central African Republic
TD	Chad
CL	Chile
CN	China
HK	Hong Kong, Special Administrative Region of China
MO	Macao, Special Administrative Region of China
CX	Christmas Island
CC	Cocos (Keeling) Islands
CO	Colombia
KM	Comoros
CG	Congo (Brazzaville)
CD	Congo, Democratic Republic of the
CK	Cook Islands
CR	Costa Rica
CI	Côte d'Ivoire
HR	Croatia

CU	Cuba
CY	Cyprus
CZ	Czech Republic
DK	Denmark
DJ	Djibouti
DM	Dominica
DO	Dominican Republic
EC	Ecuador
EG	Egypt
SV	El Salvador
GQ	Equatorial Guinea
ER	Eritrea
EE	Estonia
ET	Ethiopia
FK	Falkland Islands (Malvinas)
FO	Faroe Islands
FJ	Fiji
FI	Finland
FR	France
GF	French Guiana
PF	French Polynesia
TF	French Southern Territories
GA	Gabon
GM	Gambia
GE	Georgia
DE	Germany
GH	Ghana
GI	Gibraltar
GR	Greece
GL	Greenland
GD	Grenada
GP	Guadeloupe
GU	Guam
GT	Guatemala
GG	Guernsey
GN	Guinea
GW	Guinea-Bissau
GY	Guyana
HT	Haiti
HM	Heard Island and Mcdonald Islands
VA	Holy See (Vatican City State)
HN	Honduras
HU	Hungary
IS	Iceland
IN	India
ID	Indonesia
IR	Iran, Islamic Republic of
IQ	Iraq
IE	Ireland
IM	Isle of Man
IL	Israel
IT	Italy
JM	Jamaica
JP	Japan
JE	Jersey
JO	Jordan
KZ	Kazakhstan
KE	Kenya

KI	Kiribati
KP	Korea, Democratic People's Republic of
KR	Korea, Republic of
KW	Kuwait
KG	Kyrgyzstan
LA	Lao PDR
LV	Latvia
LB	Lebanon
LS	Lesotho
LR	Liberia
LY	Libyan Arab Jamahiriya
LI	Liechtenstein
LT	Lithuania
LU	Luxembourg
MK	Macedonia, Republic of
MG	Madagascar
MW	Malawi
MY	Malaysia
MV	Maldives
ML	Mali
MT	Malta
MH	Marshall Islands
MQ	Martinique
MR	Mauritania
MU	Mauritius
YT	Mayotte
MX	Mexico
FM	Micronesia, Federated States of
MD	Moldova
MC	Monaco
MN	Mongolia
ME	Montenegro
MS	Montserrat
MA	Morocco
MZ	Mozambique
MM	Myanmar
NA	Namibia
NR	Nauru
NP	Nepal
NL	Netherlands
AN	Netherlands Antilles
NC	New Caledonia
NZ	New Zealand
NI	Nicaragua
NE	Niger
NG	Nigeria
NU	Niue
NF	Norfolk Island
MP	Northern Mariana Islands
NO	Norway
OM	Oman
PK	Pakistan
PW	Palau
PS	Palestinian Territory, Occupied
PA	Panama
PG	Papua New Guinea
PY	Paraguay
PE	Peru
PH	Philippines

PN	Pitcairn
PL	Poland
PT	Portugal
PR	Puerto Rico
QA	Qatar
RE	Réunion
RO	Romania
RU	Russian Federation
RW	Rwanda
BL	Saint-Barthélemy
SH	Saint Helena
KN	Saint Kitts and Nevis
LC	Saint Lucia
MF	Saint-Martin (French part)
PM	Saint Pierre and Miquelon
VC	Saint Vincent and Grenadines
WS	Samoa
SM	San Marino
ST	Sao Tome and Principe
SA	Saudi Arabia
SN	Senegal
RS	Serbia
SC	Seychelles
SL	Sierra Leone
SG	Singapore
SK	Slovakia
SI	Slovenia
SB	Solomon Islands
SO	Somalia
ZA	South Africa
GS	South Georgia and the South Sandwich Islands
ES	Spain
LK	Sri Lanka
SD	Sudan
SR	Suriname *
SJ	Svalbard and Jan Mayen Islands
SZ	Swaziland
SE	Sweden
CH	Switzerland
SY	Syrian Arab Republic
TW	Taiwan, Republic of China
TJ	Tajikistan
TZ	Tanzania *, United Republic of
TH	Thailand
TL	Timor-Leste
TG	Togo
TK	Tokelau
TO	Tonga
TT	Trinidad and Tobago
TN	Tunisia
TR	Turkey
TM	Turkmenistan
TC	Turks and Caicos Islands
TV	Tuvalu
UG	Uganda
UA	Ukraine
AE	United Arab Emirates
GB	United Kingdom

US	United States of America
UM	United States Minor Outlying Islands
UY	Uruguay
UZ	Uzbekistan
VU	Vanuatu
VE	Venezuela (Bolivarian Republic of)
VN	Viet Nam
VI	Virgin Islands, US
WF	Wallis and Futuna Islands
EH	Western Sahara
YE	Yemen
ZM	Zambia
ZW	Zimbabwe

4.10. State ISO Codes

Country	Code	State
US		
	AL	Alabama
	AK	Alaska
	AB	Alberta
	AS	American Samoa
	AZ	Arizona
	AR	Arkansas
	AA	Armed Forces Ame
	AE	Armed Forces Eur
	AP	Armed Forces Pac
	BC	British Columbia
	CA	California
	CO	Colorado
	CT	Connecticut
	DE	Delaware
	DC	District of Columbia
	FM	Fed St Micronesia
	FL	Florida
	GA	Georgia
	GU	Guam
	HI	Hawaii
	ID	Idaho
	IL	Illinois
	IN	Indiana
	IA	Iowa
	KS	Kansas
	KY	Kentucky
	LA	Louisiana
	ME	Maine
	MB	Manitoba
	MH	Marshall Islands
	MD	Maryland
	MA	Massachusetts
	MI	Michigan
	MN	Minnesota
	MS	Mississippi
	MO	Missouri
	MT	Montana
	NE	Nebraska
	NV	Nevada
	NB	New Brunswich
	NH	New Hampshire
	NJ	New Jersey
	NM	New Mexico
	NY	New York
	NF	Newfoundland
	NC	North Carolina
	ND	North Dakota
	MP	Northern Mariana Is
	NT	Northwest Ter
	NS	Nova Scotia
	OH	Ohio
	OK	Oklahoma
	ON	Ontario
	OR	Oregon
	PW	Palau
	PA	Pennsylvania
	PE	Prince Edward Is

	PR	Puerto Rico
	QC	Quebec
	RI	Rhode Island
	SK	Saskatchewan
	SC	South Carolina
	SD	South Dakota
	TN	Tennessee
	TX	Texas
	UT	Utah
	VT	Vermont
	VI	Virgin Islands
	VA	Virginia
	WA	Washington
	WV	West Virginia
	WI	Wisconsin
	WY	Wyoming
	YT	Yukon
CA		
	AB	Alberta
	BC	British Columbia
	MB	Manitoba
	NB	New Brunswick
	NL	Newfoundland and Labrador
	NS	Nova Scotia
	NT	Northwest Territories
	NU	Nunavut
	ON	Ontario
	PE	Prince Edward Island
	QC	Quebec
	SK	Saskatchewan
	YT	Yukon
AU		
	ACT	Australian Capital Territory
	NSW	New South Wales
	NT	Northern Territory
	QLD	Queensland
	SA	South Australia
	VIC	Victoria
	WA	Western Australia
Outside US/CA/AU		
	ZZ	No value required

4.11. Industry Standard Credit Card Response Codes

Code	Message
00	Approved
01	Refer to issuer
02	Refer to issuer
03	Invalid merchant
04	Pick up card
05	Do not honor
06	Error
07	Pick up card
09	Request in process
12	Invalid transaction
13	Invalid amount
14	Invalid card number
15	Invalid issuer
19	Re-enter transaction
21	No action taken
22	Suspected malfunction
23	Unacceptable transaction fee
25	Unable to locate record on file
30	Format error
31	Bank not supported by switch
33	Expired card, capture
34	Suspected fraud, retain card
35	Card acceptor, contact acquirer, retain card
36	Restricted card, retain card
37	Contact acquirer security department, retain card
38	PIN tries exceeded, capture
39	No credit account
40	Invalid function
41	Lost card
42	No universal account
43	Stolen card
44	No investment account
51	Insufficient funds
52	No checking account
53	No savings account
54	Invalid expiration date
55	Invalid PIN
56	No card record
57	Transaction not permitted
58	Transaction not permitted
59	Suspected fraud
60	Acceptor contact acquirer
61	Amount limit exceeded
62	Restricted card
63	Security violation
64	Original amount incorrect
65	Count limit exceeded
66	Acceptor contact acquirer, security
67	Capture card
68	Response received late
75	PIN tries exceeded
77	PIN entry required
78	Stop payment order
79	Revocation of authorization order
80	Invalid date
81	Message flow error
82	Incorrect CVV
89	Invalid route service

90	Cutoff in progress
91	Issuer inoperative
92	Routing failure
93	Law violation
94	Duplicate transaction
95	Reconcile error
96	System error
97	Reconciliation totals have been reset
98	MAC error
99	Timeout

4.12. Payout Bank Codes

Currency	Code	Bank
VND		
	TCB.VN	Techcombank
	SCM.VN	Sacombank
	VCB.VN	Vietcombank
	ACB.VN	Asia Commercial Bank
	DAB.VN	DongA Bank
	VTB.VN	Vietinbank
	BIDV.VN	BIDV Bank
	EXIM.VN	Eximbank
	VBARD.VN	Agribank
THB		
	KTB.TH	Krung Thai Bank
	SCB.TH	Siam Commercial Bank
	BBL.TH	Bangkok Bank
	KBANK.TH	Kasikorn Bank
	TMB.TH	Thai Military Bank
	BAY.TH	Krungsri Bank
IDR		
	BRI.ID	Bank Rakyat Indonesia
	BNI.ID	Bank Negara Indonesia
	BCA.ID	Bank Central Asia
	MDR.ID	Bank Mandiri
	BBKP.ID	Bank Bukopin
	BCW.ID	Bank Commonwealth
	BDNM.ID	Bank Danamon
	BM.ID	Bank Mega
	BMP.ID	Bank Maspion
	BMTK.ID	Bank Mestika
	BPNN.ID	Bank Panin
	BSMT.ID	Bank Sumut
	BSNM.ID	Bank Sinar Mas
	BTPN.ID	Bank Btpn
	CIMBN.ID	CIMB Niaga
	HSBC.ID	HSBC Indonesia
	MBBI.ID	Maybank /bii
	OCBC.ID	OCBC Indonesia
	BBKP.ID	Bank Bukopin
	PMTB.ID	Bank Permata
	UOB.ID	UOB Indonesia
MYR		
	AFB.MY	AFFIN BANK BERHAD / AFFIN ISLAMIC BANK
	ALB.MY	ALLIANCE BANK
	ARB.MY	AMBANK BERHAD
	BIMB.MY	BANK ISLAM MALAYSIA
	BKR.MY	BANK RAKYAT MALAYSIA BERHAD
	BSN.MY	BANK SIMPANAN NASIONAL BERHAD
	CIMB.MY	CIMB Bank
	HLB.MY	Hong Leong Bank
	MBB.MY	Maybank Berhad
	PBB.MY	Public Bank
	RHB.MY	RHB
	SCB.MY	STANDARD CHARTERED BANK
	UOB.MY	UNITED OVERSEAS BANK BERHAD
KRW		
	경남은행 Kyungnam Bank 039	Kyungnam Bank
	광주은행 Kwangju Bank 034	Kwangju Bank
	KB국민은행 Kookmin Bank 004	Kookmin Bank

	IBK기업은행 IBK 003	IBK
	NH농협은행 Nonghyup Bank 011	Nonghyup Bank
	대구은행 Daegu Bank 031	Daegu Bank
	부산은행 Busan Bank 032	Busan Bank
	산업은행 KDB 002	KDB
	상호저축 dmsgod Mutual Savings Bank 050	dmsgod Mutual Savings Bank
	카카오뱅크 KakaoBank 090	KakaoBank
	새마을금고 Saemaeul Bank 045	Saemaeul Bank
	수협 Suhyup Bank 007	Suhyup Bank
	SC제일은행 SC First Bank 023	SC First Bank
	신한은행 Shinhan Bank 088	Shinhan Bank
	신용협동조합 Credit Union 048	Credit Union
	우리은행 Woori Bank 020	Woori Bank
	우체국은행 Post Bank 071	Post Bank
	전북은행 Jeonbuk Bank 037	Jeonbuk Bank
	KEB하나은행 KEB Hana Bank 081	KEB Hana Bank
	제주은행 Jeju Bank 035	Jeju Bank
	한국씨티은행 Citibank Korea 027	Citibank Korea
	케이뱅크 K Bank 089	K Bank
CNY		
	安徽省农村信用社	安徽省农村信用社
	鞍山商业银行	鞍山商业银行
	包头市商业银行	包头市商业银行
	北京农村商业银行	北京农村商业银行
	北京顺义银座村镇银行	北京顺义银座村镇银行
	北京银行	北京银行
	渤海银行	渤海银行
	沧州银行	沧州银行
	常熟农村商业银行	常熟农村商业银行
	成都农村商业银行	成都农村商业银行
	成都银行	成都银行
	承德银行	承德银行
	大连银行	大连银行
	德阳银行	德阳银行
	德州银行	德州银行
	东莞农村商业银行	东莞农村商业银行
	东莞银行	东莞银行
	东亚银行	东亚银行
	东营莱商村镇银行	东营莱商村镇银行
	东营市商业银行	东营市商业银行
	鄂尔多斯银行	鄂尔多斯银行
	佛山顺德农村商业银行	佛山顺德农村商业银行
	福建省农村信用社	福建省农村信用社
	阜新银行	阜新银行
	富邦华一银行	富邦华一银行
	富滇银行	富滇银行
	赣州银行	赣州银行
	光大银行	光大银行
	广东华兴银行	广东华兴银行
	广东南粤银行	广东南粤银行
	广东省农村信用社	广东省农村信用社
	广发银行	广发银行
	广西北部湾银行	广西北部湾银行
	广西壮族自治区农村信用社	广西壮族自治区农村信用社

	广州农村商业银行	广州农村商业银行
	广州银行	广州银行
	贵阳银行	贵阳银行
	贵州省农村信用社	贵州省农村信用社
	桂林银行	桂林银行
	哈尔滨银行	哈尔滨银行
	海口联合农村商业银行	海口联合农村商业银行
	海南省农村信用社	海南省农村信用社
	邯郸商业银行	邯郸商业银行
	韩亚银行	韩亚银行
	汉口银行	汉口银行
	杭州银行	杭州银行
	河北银行	河北银行
	恒丰银行	恒丰银行
	衡水银行	衡水银行
	葫芦岛银行	葫芦岛银行
	湖北省农村信用社	湖北省农村信用社
	湖北银行	湖北银行
	湖南省农村信用社	湖南省农村信用社
	湖州银行	湖州银行
	华融湘江银行	华融湘江银行
	华夏银行	华夏银行
	徽商银行	徽商银行
	吉林省农村信用社	吉林省农村信用社
	吉林银行	吉林银行
	济宁银行	济宁银行
	嘉兴银行	嘉兴银行
	江苏农村信用社	江苏农村信用社
	江苏银行	江苏银行
	江苏长江商业银行	江苏长江商业银行
	江西赣州银座村镇银行	江西赣州银座村镇银行
	江西农村信用社	江西农村信用社
	江西银行	江西银行
	江阴农村商业银行	江阴农村商业银行
	交通银行	交通银行
	焦作中旅银行	焦作中旅银行
	金华银行	金华银行
	锦州银行	锦州银行
	晋城银行	晋城银行
	晋商银行	晋商银行
	晋中银行	晋中银行
	九江银行	九江银行
	昆仑银行	昆仑银行
	昆山农村商业银行	昆山农村商业银行
	莱商银行	莱商银行
	兰州银行	兰州银行
	廊坊银行	廊坊银行
	临汾市尧都区农村信用社	临汾市尧都区农村信用社
	临商银行	临商银行
	柳州银行	柳州银行
	龙江银行	龙江银行
	洛阳银行	洛阳银行
	绵阳商业银行	绵阳商业银行
	南昌银行	南昌银行

	南充商业银行	南充商业银行
	南京银行	南京银行
	内蒙古银行	内蒙古银行
	宁波通商银行	宁波通商银行
	宁波银行	宁波银行
	宁夏黄河农村商业银行	宁夏黄河农村商业银行
	宁夏银行	宁夏银行
	攀枝花商业银行	攀枝花商业银行
	平安银行	平安银行
	平顶山银行	平顶山银行
	浦东发展银行	浦东发展银行
	齐鲁银行	齐鲁银行
	齐商银行	齐商银行
	企业银行	企业银行
	青岛银行	青岛银行
	青海银行	青海银行
	曲靖商业银行	曲靖商业银行
	泉州银行	泉州银行
	日照银行	日照银行
	厦门国际银行	厦门国际银行
	厦门银行	厦门银行
	山东省农村信用社	山东省农村信用社
	陕西省农村信用社	陕西省农村信用社
	上海农村商业银行	上海农村商业银行
	上海银行	上海银行
	上饶银行	上饶银行
	绍兴银行	绍兴银行
	深圳福田银座村镇银行	深圳福田银座村镇银行
	深圳农村商业银行	深圳农村商业银行
	深圳前海微众银行	深圳前海微众银行
	盛京银行	盛京银行
	四川省农村信用社	四川省农村信用社
	苏州银行	苏州银行
	台州市商业银行	台州市商业银行
	太仓农村商业银行	太仓农村商业银行
	泰安商业银行	泰安商业银行
	天津滨海农村商业银行	天津滨海农村商业银行
	天津农村商业银行	天津农村商业银行
	天津银行	天津银行
	威海市商业银行	威海市商业银行
	潍坊银行	潍坊银行
	温州银行	温州银行
	乌海银行	乌海银行
	乌鲁木齐市商业银行	乌鲁木齐市商业银行
	无锡农村商业银行	无锡农村商业银行
	吴江农商行	吴江农商行
	武汉农村商业银行	武汉农村商业银行
	西安银行	西安银行
	新韩银行	新韩银行
	新网银行	新网银行
	邢台银行	邢台银行
	兴业银行	兴业银行
	烟台银行	烟台银行
	宜昌市商业银行	宜昌市商业银行

	鄞州农村合作银行	鄞州农村合作银行
	营口银行	营口银行
	玉溪商业银行	玉溪商业银行
	云南省农村信用社	云南省农村信用社
	枣庄银行	枣庄银行
	张家港农村商业银行	张家港农村商业银行
	张家口银行	张家口银行
	长安银行	长安银行
	长沙银行	长沙银行
	招商银行	招商银行
	浙江稠州商业银行	浙江稠州商业银行
	浙江景宁银座村镇银行	浙江景宁银座村镇银行
	浙江民泰商业银行	浙江民泰商业银行
	浙江三门银座村镇银行	浙江三门银座村镇银行
	浙江省农村信用社	浙江省农村信用社
	浙江泰隆商业银行	浙江泰隆商业银行
	浙江网商银行	浙江网商银行
	浙商银行	浙商银行
	郑州银行	郑州银行
	中国工商银行	中国工商银行
	中国建设银行	中国建设银行
	中国民生银行	中国民生银行
	中国农业银行	中国农业银行
	中国银行	中国银行
	中国邮政储蓄银行	中国邮政储蓄银行
	中信银行	中信银行
	中原银行	中原银行
	重庆农村商业银行	重庆农村商业银行
	重庆黔江银座村镇银行	重庆黔江银座村镇银行
	重庆三峡银行	重庆三峡银行
	重庆银行	重庆银行
	重庆渝北银座村镇银行	重庆渝北银座村镇银行
	珠海华润银行	珠海华润银行
	自贡市商业银行	自贡市商业银行

4.13. Payment/Deposit Bank Codes

Currency	Code	Bank
VND		
	TCB.VN	Techcombank
	SCM.VN	Sacombank
	VCB.VN	Vietcombank
	ACB.VN	Asia Commercial Bank
	DAB.VN	DongA Bank
	VTB.VN	Vietinbank
	BIDV.VN	BIDV Bank
	EXIM.VN	Eximbank
	ACB.QR.VN	Asia Commercial Bank (QRservice)
	BIDV.QR.VN	BIDV Bank (QRservice)
	VCB.QR.VN	Vietcombank (QR service)
	VTB.QR.VN	Vietinbank (QR service)
	MB.QR.VN	Military Commercial Bank (QR Service)
	VPB.QR.VN	Vietnam Prosperity Joint-Stock Commercial Bank (QR Service)
	TCB.QR.VN	Techcom Bank (QR Service)
	MM.QR.VN	Momo QR Payment (QR service)
THB		
	KTB.TH	Krung Thai Bank
	SCB.TH	Siam Commercial Bank
	BBL.TH	Bangkok Bank
	KBANK.TH	Kasikorn Bank
	TMB.TH	Thai Military Bank
	BAY.TH	Krungsri Bank
	QR.TH	Thai QR Payment
IDR		
	BCA.ID	Bank Central Asia
	BRI.ID	Bank Rakyat Indonesia
	BNI.ID	Bank Negara Indonesia
	MDR.ID	Bank Mandiri
	BNI.VA.ID	Bank Negara Indonesia (VA)
	MDR.VA.ID	Bank Mandiri (VA)
	BRI.VA.ID	Bank Rakyat Indonesia (VA)
	MBBI.VA.ID	Maybank Indonesia (VA)
	BSS.VA.ID	Bank Sahabat Sampoerna (VA)
	BSM.VA.ID	Bank Sinarmas (VA)
	CIMN.VA.ID	CIMB Niaga (VA)
	PTB.VA.ID	Bank Permata (VA)
	BAG.VA.ID	Bank Artha Graha (VA)
MYR		
	CIMB.MY	CIMB Bank
	HLB.MY	Hong Leong Bank
	MBB.MY	Maybank Berhad
	PBB.MY	Public Bank
	RHB.MY	RHB
	QR.MY	QR Pay
	BST.QR.MY	Boost EWallet
	CIMB.FPX.MY	CIMB Bank (FPX)
	HLB.FPX.MY	Hong Leong Bank (FPX)
	MBB.FPX.MY	Maybank (FPX)
	PBB.FPX.MY	Public Bank (FPX)
	RHB.FPX.MY	RHB Bank (FPX)
	HSBC.FPX.MY	HSBC Bank Malaysia (FPX)
	UOB.FPX.MY	UOB Bank (FPX)
	SCB.FPX.MY	Standard Chartered Bank (FPX)
	OCBC.FPX.MY	OCBC Bank (Malaysia) (FPX)
	ALB.FPX.MY	Alliance Bank (FPX)
	ARB.FPX.MY	Ambank (FPX)

	BSN.FPX.MY	Bank Simpanan Nasional (FPX)
	AFB.FPX.MY	Affin Bank Berhad (FPX)
	BIMB.FPX.MY	Bank Islam Malaysia (FPX)
	BKR.FPX.MY	Bank Rakyat (FPX)

Disclaimer

© 2023 paypay89. All rights reserved.

Paypay89 and the authors assume no liability for errors or omissions, or for damages, resulting from the use of this guide or the information contained in this guide.