monri
PAYMENTS

WebPay

Form integration manual

Test environment

CONTENT

Requirements

Integration must be done within our test environment first. When this process is finished and approved by our staff, you may go live and start processing with real money.

To start integrating with WebPay Form service you will need the following:

- test merchant account
- HTTP client library if you need to programmatically manage approved transactions through API

If you don't have a test merchant account, please contact us at podrska@monri.com and we will open one for you.
Then you can login into your account at https://ipgtest.monri.com/en/login with login and password provided.

Payment form

WebPay Form is a simple web service; merchant should collect data consisted of buyer's profile and order info at his site and submit that data to https://ipgtest.monri.com/v2/form using HTTP POST method.

IMPORTANT Parametrize https://ipgtest.monri.com URL, in production mode the subdomain will be different.

After a valid request is made to WebPay service, a payment form is presented to the buyer. Buyer now fills in the card details and submits the form.
Service issues a request for authorization to acquirer bank and redisplays appropriate message if authorization is declined.
If authorization is approved, a redirect is made back to a merchant site.

NOTICE Buyer stays at our side if transaction is declined, form is redisplayed with appropriate message. There is no means to communicate the failure of transaction to merchant at this point.

Optionally, email messages are sent to merchant and/or buyer to notify both sides of successful purchase. This setting is available under merchant account together with custom email templates for outgoing email message

Transactions types

Authorization

Authorization is preferred transaction type for e-commerce. Merchant must capture these transactions within 28 days (or this capture period depends on the bank), in order to transfer the money from buyer's account to his own. This transaction can also be voided if a buyer cancels the order. Refund can be done after original authorization is captured.

Purchase

Purchase doesn't need to be approved; funds are transferred in next settlement between issuer and acquirer banks, usually within one business day. These transactions can be refunded within 180 days.

Capture

Approved authorization must be captured for the funds to be transferred form buyer's card to merchant account. This action can be done through merchant account interface or programmatically through an API call. A lesser amount than an original authorization amount can be captured as well, ie. a merchant can make partial delivery of goods and/or services. Capture must be done within 28 days or it will be automatically voided.

Refund

Approved purchases and captures can be refunded within 180 days. This action can be done through merchant account interface or programmatically through an API call. This is required if a merchant cancels the order after transaction is settled, or a buyer is not satisfied with delivered goods and/or services. Refunds can be done with a lesser amount than an original authorization amount.

Void

Approved authorization must be voided within 28 days if merchant cancels the order. This action can be done through merchant account interface or programmatically through an API call.

API settings

API settings can be found under your merchant account. This information is necessary for service to work properly. API settings values reflect a configuration at the merchant site:

- **Authenticity token** is unique merchant identifier, can be found under merchant settings
- **Key** is a shared secret used to calculate digest value
- **Success URL** is an URL at the merchant site where a buyer is redirected after approved authorization.
- **Cancel URL** is an URL at the merchant site where a buyer is redirected if cancel link is clicked at payment form
- **Redirect to success URL** should be set to true if merchant parses the data carried in redirect to success URL, ie. a listener will wait for this request and update order status.
- **Callback URL** should be set under merchant profile if the merchant wants us to send POST request message for each approved authorization

IMPORTANT Parametrize these values for API settings, we strongly advice a merchant to have both, testing and production environments.

Return values

Redirect to success URL is made after transaction is approved and if redirect flag is set in API settings. Data for this transaction is returned back to merchant site in this step. Following variables are set in redirect GET request to success URL:

- acquirer
- amount
- approval_code
- authentication
- cc_type
- ch_full_name
- currency
- custom_params
- enrollment
- language
- masked_pan
- number_of_installments
- order_number
- response_code
- digest (returned digest)

This request may look like this:

```
https://ipgtest.monri.com/shop/success?acquirer=integration_acq&amount=100&approval_cod
e=629762&authentication=Y&cc_type=visa&ch_full_name=John+Doe&currency=USD&custom_params
=%7Ba%3Ab%2C+c%3Ad%7D&enrollment=Y&language=en&masked_pan=434179-xxx-xxx-
0044&number_of_installments=&order_number=02beded6e6106a0&response_code=0000&digest=575
c64b2f5a0701997c8f9cfe4293706e88203cd911695ab747ce45830e4e3cbf71577c401e476988e4a4e1b0b
5f3ecbc56277394df07fa51fbe05869d1b067a
```

Returned digest is calculated as `SHA512(key+success_url without digest)`. You must check for this value at merchant application before updating status of transaction because malicious user can forge this URL.

Returned digest is calculated using following formula:

```
digest = SHA512(key + succesURL(without DIGEST))
```

- key: 2345klj

- success URL(without Digest):

```
https://ipgtest.monri.com/shop/success?acquirer=integration_acq&amount=100&approval_cod
e=629762&authentication=Y&cc_type=visa&ch_full_name=John+Doe&currency=USD&custom_params
=%7Ba%3Ab%2C+c%3Ad%7D&enrollment=Y&language=en&masked_pan=434179-xxx-xxx-
0044&number_of_installments=&order_number=02beded6e6106a0&response_code=0000
```

```
 digest
```

```
=
```

```
SHA512(2345kljhttps://ipgtest.monri.com/shop/success?acquirer=integration_acq&amount=10
0&approval_code=629762&authentication=Y&cc_type=visa&ch_full_name=John+Doe&currency=USD
&custom_params=%7Ba%3Ab%2C+c%3Ad%7D&enrollment=Y&language=en&masked_pan=434179-xxx-xxx-
0044&number_of_installments=&order_number=02beded6e6106a0&response_code=0000)
```

```
digest=575C64B2F5A0701997C8F9CFE4293706E88203CD911695AB747CE45830E4E3CBF71577C401E47698
8E4A4E1B0B5F3ECBC56277394DF07FA51FBE05869D1B067A
```

IMPORTANT Success URL should expire after some sensible amount of time or protected using session.

Callback

You can set callback URL under your merchant profile data if you want us to send a POST request with all the transaction parameters for each approved transaction.
POST request is sent to your endpoint in JSON format.
We expect `HTTP 200 OK` status response code to terminate the job, otherwise we'll send the data periodically until we receive 200.

Here is a list of parameters included in callback request:

Callback Example:

```
{"id":186562,"acquirer":"integration_acq","order_number":"a6b62d07cc89aa0","amount":100
,"currency":"EUR","ch_full_name":"John
Doe","outgoing_amount":100,"outgoing_currency":"EUR","approval_code":"914783","response
_code":"0000","response_message":"approved","reference_number":"000002902038","systan":
"186561","eci":"05","xid":"fake authenticated xid +=","acsv":"fake authenticated cavv
+=","cc_type":"visa","status":"approved","created_at":"2019-09-
06T14:24:44.906+02:00","transaction_type":"purchase","enrollment":"Y","authentication":
"Y","pan_token":null,"masked_pan":"434179-xxx-xxx-0044","issuer":"zaba-
hr","number_of_installments":null,"custom_params":"{a:b, c:d}"}
```

Email notifications

Service can notify merchant and buyer upon successful purchase.
Merchant can use this message to track pending transactions and buyer can keep them as reference. Both can be customized under your merchant account.

The following variables are interpolated into body of each email template:

- FULL_NAME
- ADDRESS
- CITY
- ZIP
- COUNTRY
- PHONE
- EMAIL
- AMOUNT
- ORDER_NUMBER
- ORDER_INFO
- CC_TYPE
- DATE
- SUCCESS_URL
- DOMAIN

Email template to notify the buyer may look like this:

```
Thank you for your business!


Credit Card Owner: ----------------
-- name: FULL_NAME address:
ADDRESS, CITY, ZIP, COUNTRY phone:
PHONE email: EMAIL


Order Details: -------------
----- order amount: AMOUNT
order number: ORDER_NUMBER
order description:
ORDER_INFO payment type:
CC_TYPE date: DATE order
url: SUCCESS_URL


Please come again!
Store adress: DOMAIN
```

NOTICE `SUCCESS_URL` has the same value as URL generated for redirect to success URL after transaction is approved. It should expire after some sensible amount of time or protected using session.

Variables - names, lengths and formats

Here are the definitions of variables which are submitted to WebPay form:

Buyer's profile

name	length	format	additional info
ch_full_name	3-30	alphanumeric	buyer's full name
ch_address	3-100	alphanumeric	buyer's address
ch_city	3-30	alphanumeric	buyer's city
ch_zip	3-9	alphanumeric	buyer's zip
ch_country	3-30	alphanumeric	buyer's country
ch_phone	3-30	alphanumeric	buyer's phone
ch_email	3-100	alphanumeric	buyer's email

Order details

name	length	format	additional info
order_info	3-100	alphanumeric	short description of order being processed
order_number	1-40	alphanumeric	unique identifier
amount	3-11	integer	amount is in minor units, ie. 10.24 USD is send as 1024
currency	predefined	alpha	possible values are **USD, EUR, BAM, HRK, AUD, CHF, RSD, CAD,GBP,MDL,NOK,SEK**

Processing data

Name	lenght	format	additional info
Language	predefined	alpha	used for form localization, possible values are **en**, **es**, **ba**, **me**, **sr**, **slo** or **hr**
transaction_type	predefined	alpha	possible values are **authorize** or **purchase**
authenticity_token	40	alphanumeric	autogenerated value for merchant account, can be found under merchant settings
Digest	128	alphanumeric	SHA512 hash generated from concatenation of key, order_number, amount and currency as strings; key can be found under merchant settings
Moto	predefined	alpha	possible value is **true** or **false**; missing variable is equivalent to **false**
tokenize_pan	-	boolean,string	**true** to enable card tokenization when the client enters card details
tokenize_pan_offered	-	boolean,string	**true** to offer the client to tokenize his PAN
whitelisted_pan_tokens	predefined	alphanumeric	Token representing previously saved customer card

Additional info:

tokenize_pan_offered – if true and merchant has secure vault active (tokenization enabled) then *save card for future payments* will be shown to customer.
If customer decided to save the card and transaction is approved we'll provide pan_token which you can store on your side.

whitelisted_pan_token provide this value if customer decides to pay with previously saved card(s). If this value is provided and valid (card not expired, token valid etc) then only cvv input will be shown on the payment form. All other information (masked pan, expiry date etc) will be prefilled. Multiple tokens can be sent (separated by comma). In that case, the user will have an option which card to use.

tokenize_pan tokenize pan without prompting the user

Custom variables

Merchant can also send custom variables if needed. They will be passed back in redirect after approved authorization. Just pack all the data you wish to send in JSON format and submit that under `custom_params` variable.

Calculating digest

Digest is calculated using following formula:

```
digest = SHA512(key + order_number + amount +currency)
```

With the following example data

- key: `2345klj`
- order_number: `abcdef`
- amount: `54321`
- currency: `EUR`

the digest formula gives a result as follows:

```
digest - SHA512.hexdigest(2345kljabcdef 54321EUR) =>
fc26113bfc9e7ea03b0db032c8da8fcded7f71fdc8b5aed47f469a01827c7fe4126b5ba2e80121a3472986
bb7342cbddb03f6e42fecf95272b231fab2daaf67c
```

Transaction management through API

WebPay API for capture/refund/void is a REST web service and you will need a HTTP client library (like cURL - http://curl.haxx.se). All API calls are XML documents POST-ed over HTTPS to our test server at https://ipgtest.monri.com

IMPORTANT Parametrize https://ipgtest.monri.com URL, in production mode the subdomain will be different.

An example of such call using cURL from command line may look like this:

```
curl -H "Content-Type: application/xml" -H "Accept: application/xml" -k -i -d

request_xml https://ipgtest.monri.com/action
```

where `request_xml` is a XML document that contains data necessary for transaction processing and https://ipgtest.monri.com/action is an URL that responds to certain API call.

IMPORTANT Accept and Content-Type headers must be set to `application/xml` for all message types.

Capture

After an authorization is successfully made, a capture call must be done to settle that authorization.

Capture XML document is POST-ed to

`https://ipgtest.monri.com/transactions/:order_number/capture.xml`, where

`:order_number` has a value from original authorization.

Document example for capture message may look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<transaction>
  <amount>54321</amount>
  <currency>EUR</currency>
  <digest>e64d4cd99367f0254ed5296d38fad6ce87d3acab</digest>
  <authenticity-token>7db11ea5d4a1af32421b564c79b946d1ead3daf0</authenticity-token>
  <order-number>11qqaazz</order-number>
</transaction>
```

Digest is calculated using following formula: `digest =`

```
SHA1(key + order_number + amount + currency)
```

With the following example data

- key: `qwert123`
- order_number: `abcdef`
- amount: `54321`
- currency: `EUR`

the digest formula gives a result as follows: `digest = SHA1("qwert123abcdef54321EUR") =`

```
"16e943d2b84546ce4271de51679abc3bf1eb163b"
```

NOTICE Node names are dasherized version of corresponding variable names, ie. underscores are replaced with dashes.

If all values pass validations at our side, transaction is send to the bank and response is returned. That response may look like this:

- **HTTP status code:** 201 - Created

- **HTTP headers:** `{:connection=>"close", :date=>"Tue, 25 Oct 2011 01:18:37 GMT", :location=>"https://ipgtest.monri.com/transactions/845", :content_type=>"application/xml; charset=utf-8", :cache_control=>"no-cache", :x_ua_compatible=>"IE=Edge", :x_runtime=>"1.475305", :transfer_encoding=>"chunked"`

- **HTTP body:**

```
<?xml version="1.0" encoding="UTF-8"?>

<transaction>

  <id type="integer">845</id>

  <acquirer>rogach bank</acquirer>

  <order-number>abcdef</order-number>

  <amount type="integer">54321</amount>

  <response-code>000</response-code>

  <approval-code>38860</approval-code>

  <response-message>authorization OK</response-message>

  <reference-number>898951263</reference-number>

  <systan>83704</systan>

  <cc-type>visa</cc-type>

  <status>approved</status>

  <transaction-type>authorize</transaction-type>

  <created-at type="datetime">2011-10-25T03:18:38+02:00</created-at>

</transaction>
```

New transaction is generated - `201 Created HTTP status code`, and it's location is set in appropriate HTTP header. A client then must parse a body from HTTP response and extract all values from that XML document. Transaction is approved only and if only status is set to `approved`. All other fields are standard data carried over payment networks. If issuer declines a transaction, status flag is set to `decline`. In a case of an error, the flag will be set to `invalid`.

IMPORTANT Do not rely on any output variable except status to determine success of autorization.

NOTICE We highly recommend to our merchants to keep a whole response string and to save all parsed values for easier troubleshooting during the integration phase and production later on.

The quality of our support depends on availability of these information.

In case of invalid request, service will also return a response with

`406 Not Acceptable HTTP status code` and XML document in its body. Each offended variable will be printed out along with brief explanation what went wrong. That response may look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<errors>
  <error>Digest is invalid</error>
</errors>
```

Refund

Purchase and capture messages can be refunded within 180 days. Request XML for this `transaction_type` is identical to capture example, but now the document is POST-ed to `https://ipgtest.monri.com/transactions/:order_number/refund.xml`, where `:order_number` has a value from original purchase or capture.

Response has identical structure as capture response and all response fields should be treated in the same way.

Void

Void messages are POST-ed to

`https://ipgtest.monri.com/transactions/:order_number/void.xml`, where

`:order_number` has a value from original message (authorization, capture, purchase or refund). They have identical structure as capture or refund messages.

Response has identical structure as capture response and all response fields should be treated in the same way.

Transactions with installments

Form will automatically recognize cards which are eligible for payments in installments. Installment range setup will be defined by the merchant-acquirer agreement.

3-D Secure

WebPay handles 3-D secure processing for you, this kind of integration doesn't require any additional programming.

Demo client (Test client)

For easier integration we provided a demo client. Turn on the debug mode in API settings under your merchant account to activate the client. A link to client is available upon activation.

Append /success to that link and set as success URL if you want to see how those page could look like.

Look & feel

Custom headers, footer and custom CSS support are currently disabled, due to introduction of new form templates.

List of response codes

Here is the list of response codes and their description:

- 0000 - Transaction Approved
- 1001 - Card Expired
- 1002 - Card Suspicious
- 1003 - Card Suspended
- 1004 - Card Stolen
- 1005 - Card Lost
- 1011 - Card Not Found
- 1012 - Cardholder Not Found
- 1014 - Account Not Found
- 1015 - Invalid Request
- 1016 - Not Sufficient Funds
- 1017 - Previously Reversed
- 1018 - Previously Reversed
- 1019 - Further activity prevents reversal
- 1020 - Further activity prevents void
- 1021 - Original transaction has been voided
- 1022 - Preauthorization is not allowed for this card
- 1023 - Only full 3D authentication is allowed for this card
- 1024 - Installments are not allowed for this card
- 1025 - Transaction with installments can not be send as preauthorization
- 1026 - Installments are not allowed for non ZABA cards
- 1050 - Transaction declined
- 1802 - Missing fields
- 1803 - Extra fields exist
- 1804 - Invalid card number
- 1806 - Card not active
- 1808 - Card not configured

- 1810 - Invalid amount
- 1811 - System Error, Database
- 1812 - System Error, Transaction
- 1813 - Cardholder not active
- 1814 - Cardholder not configured
- 1815 - Cardholder expired
- 1816 - Original not found
- 1817 - Usage Limit Reached
- 1818 - Configuration error
- 1819 - Invalid terminal
- 1820 - Inactive terminal
- 1821 - Invalid merchant
- 1822 - Duplicate entity
- 1823 - Invalid Acquirer
- 2000 - Internal error - host down
- 2001 - Internal error - host timeout
- 2002 - Internal error - invalid message
- 2003 - Internal error - message format error
- 2013 - 3D Secure error - invalid request
- 3000 - Time expired
- 3100 - Function not supported
- 3200 - Timeout
- 3201 - Authorization host not active
- 3202 - System not ready
- 4001 - 3D Secure error - ECI 7
- 4002 - 3D Secure error - not 3D Secure, store policy
- 4003 - 3D secure error - not authenticated
- 5018 - RISK: Minimum amount per transaction
- 5019 - RISK: Maximum amount per transaction
- 5001 - RISK: Number of repeats per PAN
- 5020 - RISK: Number of approved transactions per PAN
- 5003 - RISK: Number of repeats per BIN
- 5016 - RISK: Total sum on amount

- 5021 - RISK: Sum on amount of approved transactions per PAN
- 5022 - RISK: Sum on amount of approved transactions per BIN
- 5005 - RISK: Percentage of declined transactions
- 5009 - RISK: Number of chargebacks
- 5010 - RISK: Sum on amount of chargebacks
- 5006 - RISK: Number of refunded transactions
- 5007 - RISK: Percentage increment of sum on amount of refunded transactions
- 5023 - RISK: Number of approved transactions per PAN and MCC on amount
- 5011 - RISK: Number of retrieval requests
- 5012 - RISK: Sum on amount of retrieval requests
- 5013 - RISK: Average amount per transaction
- 5014 - RISK: Percentage increment of average amount per transaction
- 5015 - RISK: Percentage increment of number of transactions
- 5017 - RISK: Percentage increment of total sum on amount
- 5050 - RISK: Number of repeats per IP
- 5051 - RISK: Number of repeats per cardholder name
- 5052 - RISK: Number of repeats per cardholder e-mail
- 6000 - Internal error - systan mismatch