



Gateway APIv4 Integration Documentation

updated: 23.08.2023

Table of Contents

Test environment endpoints	4
Production environment endpoints	4
Test Cards	5
Payment Integration Overview	6
Payment Process Overview	6
Payment Integration Variants	7
Payment Types	7
One off Payment	7
Merchant Initiated Transaction (MIT) Payments	8
Customer Initiated Transaction(CIT) Payments	8
Payment API Overview	9
Changes compared to v3	9
NEW PAYMENT STATUSES	9
2-STEP PAYMENT PROCESS	9
OPEN BANKING REFUNDS	10
API Endpoints	10
The API endpoints for initiating payments	10
The API endpoints for payment management	11
The API endpoints for tokens	12
Other API endpoints	12
Payment Initiation Response	12
Important Factors	13
Token agreement	13
Nonce	13
Order Reference	13
Date and Time formats	13
Callback Notification	14
Request Parameter Types	15
Payment Statuses	15
API Details	16
Supported Formats	16
Security	17
Authentication	17
HTTP Response Codes	17
HMAC calculation	17
Payment Endpoints	18
POST /payments/oneoff	18
POST /payments/mit	24
POST /payments/cit	26
GET /payments/:payment_reference	30
POST /payments/void	34

POST /payments/capture	35
POST /payments/refund	37
POST /payments/charge	38
POST /refunds/xml_payment_file	42
POST /refunds/revert	42
GET /shops	44
GET /shops/:id	45
GET /processing_accounts/:account_name	47
POST /mobile_payments/card_details	49
Token Endpoints	51
POST /tokens/delete	51
GET /tokens/get_card_art	52
GET /tokens/status	53
Version history	54

Test environment endpoints

Application	endpoint URL	usage
Gateway API	https://igw-demo.every-pay.com/api/v4	For JSON API
Merchant Portal	https://mwt-seb-demo.every-pay.com	Access API username and secret and track payment data. Note: API username and secret are different in Production and Test environment.

Production environment endpoints

Application	endpoint URL	usage
Gateway API	https://payment.ecommerce.sebgroup.com/api/v4	For JSON API
Merchant Portal	https://portal.ecommerce.sebgroup.com/merchant_settings/general	Access API username and secret and track payment data. Note: API username and secret are different in Production and Test environment.

Test Cards

Please note that only test cards must be used for testing. The following test cards can be used to perform successful test payments:

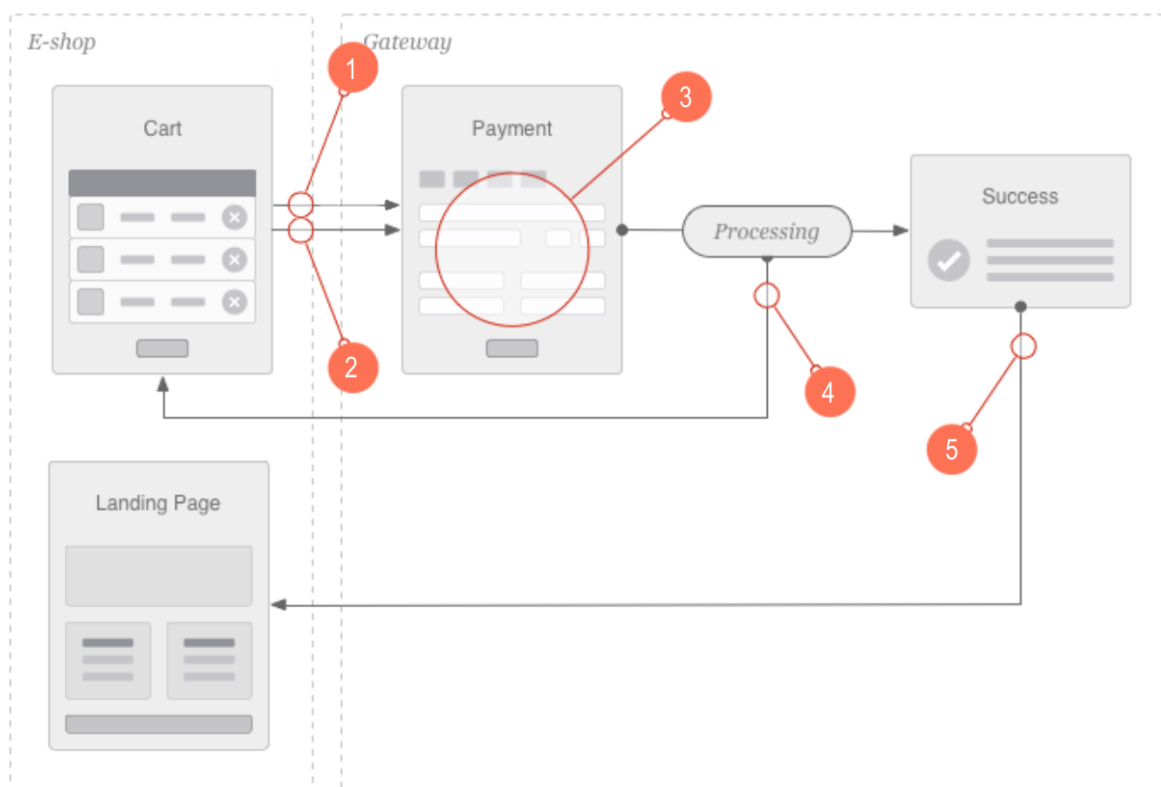
Card type	Card number	Expiration date	CVC code	Cardholder name	3DS Password
Mastercard	2223000010021381	12/25	656	(any name)	Secret!33
Visa	4051700003921116	12/22	137	(any name)	Secret!33

The following test cards can be used to perform negative scenarios:

Card type	Card number	Expiration date	CVC code	Cardholder name	3DS Password
Mastercard	5413330089020029	any	any	(any name)	Secret!33
Mastercard	5413330089010020	09/17	111	(any name)	Secret!33

1. Payment Integration Overview

1.1. Payment Process Overview



On a high level the steps in the payment collection are as follows:

1. Payment Initiation request by the Merchant e-shop to Payment Gateway (this is Backend JSON API request). Response to this request contains, among other things, the Payment Link.
2. E-shop redirects customers to (or opens in an iframe) the Payment Link.
3. Payment Gateway displays a selection of available payment methods. The customer chooses one and initiates the payment flow. Depending on the payment method, the flow can consist of one or more screens as well as redirects to third-party authentication services.
4. When payment processing has been completed, notification is sent to merchant and merchant queries payment status.
5. Iframe is not notified in APIv4 so that customer is always redirected to customer_url (even inside iframe). For that reason, **if a customer is returned to customer_url before the notification is sent, the Merchant should trigger the same status update request if notification is received.**

Note: If merchants treat customer return as trigger for payment status check, it will prevent problems in case of not getting the notification (e.g due to network outages)

1.2. Payment Integration Variants

To integrate payments to e-shop, there are two possible variants:

1. Redirect Integration – the Customer is redirected away from the e-shop and payment UI is displayed full screen.
2. Iframe integration – the Customer does not leave e-shop, but payment UI is integrated inside an iframe. Payment UI will detect automatically if it is rendered inside an iframe and adjust accordingly. Iframe solution is based on redirect solution, so merchants should first implement redirect flow and then switch to iframe. Please also note that some banks do not allow the use of iFrame, so open banking payments might break out of iFrame.

The payment UI would remain mostly the same regardless of the integration option. In case of iframe integration, certain parts of the UI are omitted, such as Merchant's logo at the top and payment gateway details at the bottom of the page.

1.3. Payment Types

1.3.1. One off Payment

When the Payment Service User (PSU) completes the purchase and proceeds with the payment, a merchant initiates payment with(`/payments/oneoff`) to us. As a response to this request, payment link with payment methods are provided. PSU is redirected to payment link to complete the payment.

When merchant is PCI DSS compliant then merchant initiates payment with `/payments/oneoff` and uses `/payments/charge` to complete the payment.

Customer selects one of the available payment methods (card payment, open banking, etc.). Based on the selected payment method, PSU needs to perform a different number of operations. For example, while in card payments these operations include entering card details and 3DS operations, in open banking, PSU is supposed to complete initial authentication, selection of IBAN and authorization of payment. After completion of payment, merchant gets the notification and merchant queries then update payment information with the payment status (`/payments/:payment_reference`). Finally, PSU is redirected back to the merchant's website to see the payment information.

This endpoint also can be used for tokenization (**currently only for card payments**) by setting the required request parameters (explained in payment endpoints). Additionally, without any purchase, tokenization for further payments can be made by setting the amount as 0.

Note: The redirection of PSU is explicitly visible only in redirect integration. In iframe integration the parent page will stay in place and redirection happens inside the iframe.

Get Payment Methods List

For one-off payment, it uses the same endpoint as above but differs from regular one-off payment with initial payment methods list call (`/processing_accounts/:account_name`). Merchant gets the payment methods list belonging to a specific processing account. Afterwards, operations which are used for regular one-off payment are performed. It is **highly recommended** to use (`/processing_accounts/:account_name`) endpoint to get the payment methods list instead of (`/payments/oneoff`) endpoint. With this way, there won't be new payment initiation when the user goes back and forth in the checkout page and payments which are in initial status will reflect actually initiated payments.

1.3.2. Merchant Initiated Transaction (MIT) Payments

MIT's are token-based payments governed by an agreement between the cardholder and merchant that, once set up, allows the merchant to initiate subsequent payments from the card without any direct involvement of the cardholder. As the cardholder is not present when an MIT is performed, the cardholder authentication is not performed. However, MIT always relates to a previous customer present transaction (even if it is a zero-value transaction) that was performed to establish the initial agreement with the cardholder - this initial payment must be always performed by using 3DS - strong customer authentication (regulated by Payments Service Directive 2 (PSD2)).

MIT payment is initiated by the merchant by using the token which has been received with one-off payment when storing the credentials. Steps for payment method selection and processing payments are eliminated. There is no PSU involvement and 3DS is not available in this payment type. Simply, the merchant initiates the payment (`/payments/mit`) and uses `/payments/charge` to complete the payment. Notification is also sent to the merchant for any case. (If the merchant cannot get the results (e.g. connection issues), they will be able to get the final payment status with the payment reference in the notification). MIT payment can be performed as *recurring* payment (fixed amount and interval) or as *unscheduled* credential-on-file transaction (fixed or variable amount, fixed or variable frequency) as established in an [agreement](#) between the Merchant and Customer.

1.3.3. Customer Initiated Transaction(CIT) Payments

As in MIT payments, previously stored credentials are used for this type of payment. The main difference with MIT payments is that the customer actively participates in the transaction (like one-click-payments). Also as a response, a payment link is provided to complete the payment which means PSU involvement might be required as well as 3DS authentication. Simply, the merchant initiates the payment (`/payments/cit`) and uses `/payments/charge` to complete the payment. When the payment is processed, notification is sent to the merchant and the merchant queries GET the payment status (`/payments/:payment_reference`).

2. Payment API Overview

2.1. Changes compared to v3

APIv4 is built on top of v3 with few critical changes to make the API work better for different use cases.

NEW PAYMENT STATUSES

With the addition of Open Banking and other APM payment methods, the statuses for payments are being refactored to present the true state more correctly.

The payment statuses can be divided into two categories:

- *Major statuses* – these statuses affect Merchant's use-cases and Merchants are expected to know how to handle all of them. NOTE: it is possible for a payment to move from one final state to another (e.g. when a payment is voided or refunded). Changes to major statuses are considered breaking and are always published as new API versions.
- *Minor statuses* – used internally by the gateway to process payments. Mostly merchants should not even see these statuses, unless they happen to fetch payment status or list payments in Portal during processing. Merchants are expected to ignore any minor statuses and gateway could add new minor statuses between two API releases.

2-STEP PAYMENT PROCESS

Now all payment flows are 2-step processes, just like One-Off payments and 3DS-challenged CIT payments have been in previous version – this ensures that the Merchant always has the `payment_reference` to request the payment status. The new payment flow will be:

1. Initiate payment using `/oneoff`, `/mit` or `/cit` endpoints.
2. Use the `/payments/charge` endpoint to complete the payment
 - a. Note: for one-off payments, this charge endpoint is only available for PCI DSS Compliant merchants. Other merchants need to redirect customers to the Payment Page.

CHARGE ENDPOINT

The `charge` endpoint to complete One-off, CIT or MIT payment.

Also via `charge` endpoint it is possible to share a token. Token Sharing is the process of two Merchants acting together to allow Customers to use the saved Token of one Merchant to

pay for goods or services of the other. In this process, one merchant, the Token Owner (TO) authorises the use of the saved token by the other merchant – the Token User (TU).

Token Owner must gather the following information and forward it to Token User:

- token – the credit card Token to be shared to Token User.
- token_owner – the api_username of Token Owner (as given in the Merchant portal).
- token_timestamp – UNIX timestamp of the token sharing authorisation moment in UTC. This timestamp cannot be older than 5 minutes by the time the request reaches Gateway.
- token_hmac – HMAC that proves to the Gateway that Token Owner does allow the sharing. See chapter HMAC calculation.

OPEN BANKING REFUNDS

Now merchants are able to make open banking refunds. For that use API endpoint `payments/refund`. After successful API request payment status changes from “Settled” to “Refunded”. After the refund is made, we send a callback with `event_name=refunded`. API endpoint `payment/refund` response includes `transaction_reference` that is required to get XML pain 001 file with `refunds/xml_payment_file` endpoint. To get the XML pain 001 file transaction references can be separated by a comma and the maximum amount of allowed transaction reference is 1500.

If it later turns out that refunding the payment was unsuccessful in the SEB system it is possible to revert refunded payment via API endpoint `refunds/revert`. After the refund is reverted and the standing amount becomes equal to the initial amount payment status becomes “Settled”.

2.2. API Endpoints

Payment Gateway is separated into two parts:

- **JSON API** - interfaced by Merchant’s e-shop directly
- **HTML UI** - interfaced by Customer using a browser, normally by being redirected there from the Merchant’s e-shop.

The JSON API is protected by TLS and authenticated using HTTP Basic Auth with the `api_username` and `api_secret` that can be found in the Merchant portal.

The API endpoints for initiating payments

- `/payments/oneoff` – this endpoint can be used for the following use cases:
 - For regular one-off payment; when the value of the `request_token` parameter is false, card details will not be stored and regular one-off payment is made. PSU can make card, open banking or PayPal payment.
 - To save card details for MIT and CIT; when `request_token = 'true'` and `token_agreement` is specified as `'unscheduled'`, card details are saved

and payment is made. The stored card can be used in future MIT/CIT payments.

- To save card details for recurring payments and subscriptions; when `request_token = 'true'` and `token_agreement` is specified as `'recurring'`, card details are saved and payment is made.
- To save card details without any purchase; when the amount is set as 0, `request_token = 'true'` and `token_agreement` is selected as either `'unscheduled'` or `'recurring'` card details are saved for future MIT/CIT or subscription without any purchase.

Note: `Token_agreement` must be specified when `request_token='true'`.

- `/payments/mit` - Merchant Initiated Transaction payment, the token agreement can be `'unscheduled'` or `'recurring'`. To finalize the payment merchant needs to use `/payments/charge` endpoint. This is only used for card transactions.
- `/payments/cit` - Customer Initiated Transaction payment, the token agreement can only be `'unscheduled'`. To finalize the payment merchant needs to use `/payments/charge` endpoint. This is only used for card transactions.
- `/payments/charge` - .endpoint to complete the payment for the one-off, cit or mit payment. Note: for one-off payments, this charge endpoint is only available for PCI DSS Compliant merchants. Other merchants need to redirect customers to the Payment Page. Also with charge endpoint it is possible to share a token between merchants.

The API endpoints for payment management

- `/payments/void` – Void transaction reverses an authorised payment that has not been set to be captured. This is only used for card transactions.
- `/payments/capture` – Capturing an authorised payment will complete (finalise) the payment. This is only used for card transactions.
- `/payments/refund` – It is possible to refund card and open banking payments, payment can be refunded in full or partial amount.
 - Refunding a settled card payment will return the funds to the customer's bank.
 - Open banking flow supports only marking the payment status as refunded, no real transactions are made. Refund response includes `transaction_reference` that is required to get XML pain 001 file with `refunds/xml_payment_file` endpoint.
- `refunds/xml_payment_file` - to get XML pain 001 file that can be later uploaded to the SEB system. Transaction references can be separated by a comma and the maximum amount of allowed transaction reference is 1500.
- `refunds/revert` - If it later turns out that refunding the payment was unsuccessful in the SEB system it is possible to revert refunded payment. IT is recommended to use to align transaction status with real payment status. After the refund is reverted and the standing amount becomes equal to the initial amount payment status becomes "Settled".

- `/mobile_payments/card_details` – It can be used by SDKs to set card details.
- `/payments/charge` this is only used for card transactions, to collect card details from the PCI DSS Compliant merchants.

The API endpoints for tokens

- `/tokens/delete` – It is used to delete ACTIVE or SUSPENDED tokens.
- `/tokens/get_card_art` – It is used to get card art of tokens.
- `/tokens/status` – It returns the status of tokens.

Other API endpoints

- `/payments/:payment_reference` - Returns the current state of the payment (similar response to payment initiation, with statuses updated to reflect changes).
- `/processing_accounts/:account_name` – Returns details of the Processing Account, most importantly the available Payment Methods list.
- `/shops` – Return a list of shops with primary configuration parameters for the shops.
- `/shops/:id` – Return info about a specific shop, including configured processing accounts.

See below for a detailed explanation of the Payment API.

2.3. Payment Initiation Response

The response to the above mentioned payment initiation requests consists of the following details:

- *Payment Reference* – reference to identify this payment in the Payment Gateway and Portal
- *Payment State* – current state of the payment
- *Payment Link* – URL to redirect the customer to. Returned when payment needs Customer interaction.
- *Payment Methods* – a list of payment methods (Card payments, Open Banking payments, PayPal) available for this payment. The list contains direct links to the methods, links to logos and display names for the method.
- *Payment Details* – rest of the details for the payment, such as the purchase amount, order reference, customer details etc, depending on the state of the payment.

The exact content of the response depends on the state of the payment – if the payment was finalised upon creation (e.g. MIT payment), then the response would not contain Payment Link or the Payment Methods list, but instead details regarding the used payment method (e.g. credit card details).

After initiating the payment and receiving the response, the response should always be examined especially for CIT payment, as the results can vary depending on the 3DS

requirements by Issuing Bank – it could be waiting for 3DS authentication in which case Payment Link is provided in the response.

After payment is processed by us, **notification** is sent to the merchant (e-shop) which contains payment reference and order reference. With this payment reference and order reference, the merchant gets the updated payment status as well as other payment information.

2.4. Important Factors

2.4.1. Token agreement

The Payment Gateway supports the following token agreements:

- 'Unscheduled' - type of the agreement where both the amount and time period between payments can vary and cannot be defined at the time of agreement. Payment is usually triggered based on usage. This type of agreement is also used for future CIT payments.
- 'Recurring' - these are payments for the delivery of ongoing goods or services. They have a fixed amount and interval for each payment, as established in the merchant customer agreement.

The token agreement indicators are only used with payment methods that support/require them (e.g. card payments) and ignored for others (e.g. SEPA payment). If not specified, then payments default to regular one-off payment.

2.4.2. Nonce

All message requests contain the 'nonce' field that can be used to verify the uniqueness of the request messages. This approach helps to prevent possible message replay attacks.

2.4.3. Order Reference

The main purpose of the `order_reference` parameter is to match the payment in the system to the correct corresponding order in the merchant's e-shop. By default, the `order_reference` uniqueness validation is enabled for the e-shop meaning that multiple payment attempts are allowed for one order reference until a successful payment is performed. As an additional benefit, matching the `order_reference` and validating its uniqueness in the merchant's e-shop provides an extra layer of security against tampering attacks. However, if needed the `order_reference` uniqueness validation can be turned off. When disabled, multiple successful payments are allowed for the same order reference.

2.4.4. Date and Time formats

The `timestamp` field represents the time of the request. The request will be rejected if the provided timestamp is outside of an allowed time-window.

The `timestamp` field, as well as all other datetime fields, will use ISO 8601 standard formatting, for example: `2019-05-31T09:14:58+03:00`.

2.5. Callback Notification

Callback notifications are used to inform merchants to get the updated status of the payments. When the payment is finalized either by a customer or automatically after the payment initiation, notification is sent to the merchant's `callback_url`.

Merchants must set their `callback_url` in the merchant portal under e-shop settings.

`callback_url` includes `payment_reference`, `order_reference` (deprecated) and `event_name` merchants can use this reference to get the payment information by using `GET/payments/:payment_reference` endpoint. `Event_name` is sent for payment status, fraud and dispute updates as well as token updates.

NB! Please do not use `order_reference` when you integrate, we are planning to remove it in the next API version from the callbacks.

Possible values for `event_names`:

`status_updated` - payment status is updated. To check the status of that payment, please use `GET /payments/:payment_reference`

`voided` - payment is voided, cancellation of authorization.

`refunded` - payment is refunded, reimbursement of payment.

`refund_failed` - open banking payment refund fails. In this case payment status might change, please use `GET /payments/:payment_reference` to check the final status of that payment.

`chargebacked` - payment is charged backed, the cardholder has disputed payment and the issuer bank has initiated a chargeback process.

`marked_for_capture` - payment is marked for capture.

`abandoned` - payment is abandoned, final status and means that payment is failed.

`issuer_reported_fraudulent` - Payment has been marked as fraudulent payment and reported by the issuer.

`merchant_reported_fraudulent` - Payment has been marked as fraudulent payment and reported by merchant.

`dispute_opened` - Dispute is opened.

`dispute_updated` - Dispute is updated.

`dispute_reversed` - Dispute is reversed

`dispute_charged_back` - Dispute is charged back.

`dispute_reopened` - Dispute is reopened.

Possible values for instrument token updates:

`card_art_updated` - Card art is received. Merchants can get the card art after this notification.

`token_updated` - Token status is updated by the issuer. Merchants can check the status of the token after getting this notification.

`status_update` - When payment acquiring is completed and funds received to the merchant's account. Available only if the acquiring bank supports this functionality.

Note: If initial callback notification does not reach the merchant's server, we try to resend the callback several times until it succeeds or fails permanently. Maximum 6 retries will be done according to the schedule below.

Delivery Attempt	Interval
1	1 second
2	5 minutes
3	1 hour
4	24 hours
5	48 hours
6	72 hours
.freeze	the callback fails permanently

2.6. Request Parameter Types

M - field is mandatory

O - field is optional

OF - field is optional, but including it improves fraud detection (therefore it's highly recommended to provide this information whenever possible)

C - field is conditional, which means this parameter needs to be sent in case of the existence of another parameter. (e.g. token_agreement) or depends on a payment method.

2.7. Payment Statuses

Initial: Payment is initiated by a Merchant. Payment method is not selected yet. It is a common state for all types of payments (card, open banking, alternative payments). The customer is supposed to select the payment method and continue the payment flow.

Waiting for SCA (strong customer authentication): For open banking in this state, the IBAN is selected, payment initiation is sent to the bank and the customer is waited to complete SCA. For card payment the 3DS authentication window was prompted.

Sent for processing: This state is used for a case in which payment is confirmed by the user but final confirmation by the bank has not arrived yet. In case of this status, a customer can be returned to the e-shop depending on the merchant's preference. So, customer return should be treated similarly as a trigger as if callback notification was received and merchants need to check payment status. Until the callback notification for final status is received, merchants should show a proper message to the customer which says payment is in progress. It is possible to test sent for processing status with any open banking payment method in the demo environment. When initiating payment, set the amount to 33 EUR and go through the payment flow like normally. After confirming the payment, it will stay in sent for processing status for 30 seconds and then it will go to final status, normally it will be settled.

Abandoned: This state refers to the case in which a customer does not complete the payment confirmation and abandon the payment. For card payments every customer has 15 minutes to perform 3DS authentication. If the time is exceeded and 3DS authentication is not finalised the payment status is changed from 'Waiting for 3DS' to 'Abandoned'. It is Final status and means failed payment.

Failed: payment that failed for technical reasons (either our or processors, i.e 399, 4999 failures) or authorization was declined by a card issuer. This is the final status of the payment.

Settled: For card payments it is the settlement of the transaction, the acquirer bank has transferred the funds to the merchant's bank account. For open banking payments this state is obtained after the customer completes SCA and payment passes all the checks on the bank side. This is the final state. For Paypal payments, it means settlement is completed to the merchant's PayPal account.

Authorised: Initialization of the payment. A process whereby a card Issuer approves or declines the use of the card for a particular purchase transaction at a merchant. If the authorization is successful the purchase amount will be reserved on the cardholder's account. In the case of 3DS payments, the authorization process also involves cardholder authentication*.

Voided: Cancellation of authorization. Void blocks funds transfer for an authorized payment. This is the final status of the payment.

Refunded: payment is partially or fully refunded, reimbursement of the payment.

Charged Back: The cardholder has disputed payment and Issuer bank has initiated a chargeback process.

3DS confirmed: Intermediate status when 3DS flow is completed but payment is not processed further due to technical errors.

* *Authentication:* A process whereby the card Issuer assures that the presenter of the card is a valid card owner.

3. API Details

All below API endpoints are accessible under the top level APIv4 endpoint: `/api/v4`

The API is implemented in the REST architectural style:

- payments and transactions are exposed as resources
- operations on resources are performed using standard HTTP methods (GET, POST, etc)
- each request must specify a media type for the resource presentation format
- error conditions on operations are expressed as HTTP response codes

3.1. Supported Formats

Payment gateway supports communication in JSON format.

API format must be specified in the request HTTP header as follows:

- `Content-Type: application/json`
- `Accept: application/json`

3.2. Security

All connections to Payments Gateway are carried out over TLS connection. Merchant API client **MUST** validate the certificate of the Payment Gateway to avoid Man-in-the-Middle attacks on payment data.

3.3. Authentication

Payment Gateway will authenticate Merchant's API client using HTTP Basic Auth, using the `api_username` and `api_secret` as the username and password.

3.4. HTTP Response Codes

The API will use the following HTTP response codes to indicate the outcome of the request. These response codes are for the API request only, the response can include additional response codes/error codes, as appropriate for the response (e.g. processing error codes for payment initiation).

Code	Status	Description
200	Success	Standard response for successful queries.
400	Bad Request	Returned if HTTP operation was not understood or was incorrectly formatted.
401	Unauthorised	Returned if processing the request is refused because of failed authentication, nonce or timestamp.
403	Forbidden	Returned if processing the request is refused.
422	Unprocessable Entity	Returned if processing the request was not successful for any reason, including processing errors such as validation (e.g. invalid CC details) , fraud check or issuer declines.
503	Internal Server Error	Returned if the request cannot be processed because of the technical errors in the server.

3.5. HMAC calculation

The HMAC is calculated by Token Owner as follows:

1. Create `hmac_string`:
`"cc_token=<Token>&token_owner=<api_username>&token_timestamp=<timestamp>"`

- a. This is the `api_username` of Token Owner.
 - b. The keys in the string are in alphabetical order.
2. Calculate HMAC: `hmac_sha1(hmac_string, api_secret)`
 - a. This is the `api_secret` of Token Owner
 - b. The HMAC must be lowercase hex

Example Ruby code to calculate `token_hmac`

```
timestamp = Time.now.utc.to_i
hmac_string =
"cc_token=#{cc_token}&token_owner=#{api_username}&token_timestamp=#{timestamp}"
token_hmac = OpenSSL::HMAC.hexdigest("sha1", "api_secret", hmac_string)
```

3.6. Payment Endpoints

3.6.1. POST /payments/oneoff

This endpoint is used to initiate payment, i.e. a payment that will ask for payment details (e.g. card number) and is not related to any previous payment. Also is used as a first step to initiate CIT or MIT payments

ENDPOINT: `/payments/oneoff`

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
<code>api_username</code>	M	The <code>api_username</code> of the Merchant sending the request. Must match with username in the Authorization HTTP header.
<code>account_name</code>	M	Processing account used for the payment. Most importantly, this will determine available payment methods and currency of the payment.
<code>amount</code>	M	Transaction amount, use decimal number with 2 digit precision, e.g. 10.55. The currency is taken from the specified processing account. Can be also set as 0 for card verification (saving card for further token payments).
<code>customer_url</code>	M	URL where the Customer should be redirected after completing the payment. <code>payment_reference</code> and <code>order_reference</code> parameters are added when a customer is redirected to <code>customer_url</code> . Customer URL has to be a fully qualified domain

		name, it is not possible to use an IP address or localhost.
token_agreement	C	It must be sent when request_token=true. It is the type of the agreement. Valid values: `unscheduled` or `recurring`. See the above Payment Types. It is a conditional parameter.
mobile_payment	O	Payment is initiated via mobile apps like SDK. Valid values: true, false. The default value is false.
order_reference	M	Merchant's order reference. Uniqueness requirements can be configured in Portal. Maximum length is 255 character for card payments and 120 character for open banking payments, unless otherwise defined by Acquirer bank/host. Allowed characters: [a-zA-Z0-9/-?:(),'+]
nonce	M	Unique request (see below for details).
structured_reference	O	Structured reference, different formats in different countries.
payment_description	O	When this parameter is provided, it will be used with open banking payment methods instead of the standard generated description value. Allowed characters: [a-zA-Z0-9/-?:(),'+] With possible truncation due to payment method configuration and transliteration requirements we recommend to use 65 chars when sending the payment description field.
email	OF	Customer's email. Used for Fraud Prevention.
customer_ip	OF	Customer's IP address. Used for Fraud Prevention. Do not set this to some fixed value, e.g Merchant's server, as this will start generating false positives in Fraud Check.
preferred_country	O	Default country for a payment selection page. Possible values <ul style="list-style-type: none"> ● 'EE', for Estonia ● 'LT', for Lithuania ● 'LV' for Latvia. When the payment selection page is opened, the preferred country's alternative payment methods will be listed first.
billing_city	O	Cardholder Billing address city.

billing_country	O	Cardholder Billing address country. Format: ISO alpha-2.
billing_line1	O	Cardholder Billing address line 1.
billing_line2	O	Cardholder Billing address line 2.
billing_line3	O	Cardholder Billing address line 3.
billing_postcode	O	Cardholder Billing address postal code.
billing_state	O	Cardholder Billing address state.Format: ISO 3166-2.
shipping_city	O	Cardholder shipping address city. It will be sent if available.
shipping_country	O	Cardholder shipping address country. It will be sent if available. Format: ISO alpha-2.
shipping_line1	O	Cardholder shipping address line 1. It will be sent if available.
shipping_line2	O	Cardholder shipping address line 2. It will be sent if available.
shipping_line3	O	Cardholder shipping address line 3. It will be sent if available.
shipping_code	O	Cardholder shipping address postal code. It will be sent if available.
shipping_state	O	Cardholder shipping address state. It will be sent if available.
locale	O	<p>A locale that should be used as the default for payment UI. Customers can change it via menu. Defaults to 'en'.</p> <p>Accepted values are:</p> <ul style="list-style-type: none"> ● 'en' - English ● 'et' - Estonian ● 'fi' - Finnish ● 'de' - German ● 'lv' - Latvian ● 'lt' - Lithuanian ● 'ru' - Russian ● 'es' - Spanish ● 'sv' - Swedish ● 'da' - Danish ● 'pl' - Polish ● 'it' - Italian ● 'fr' - French

		<ul style="list-style-type: none"> • 'nl' - Dutch • 'pt' - Portuguese • 'no' - Norwegian • 'hu' - Hungarian • 'sk' - Slovak • 'cz' - Czech <p>Note: User's previous preference overrides the locale requested by the merchant.</p>
request_token	O	Boolean to indicate that payment should return a token in the response - for future token payment usage. Valid values: true, false.
token_consent_agreed	O	Boolean to indicate that merchant has token consent in Terms and Conditions to avoid the save card details checkbox in card details form
timestamp	M	A timestamp of request's creation time (see below for details).
integration_details	O	Details of integration. Following fields are subfields of this.
integration_details.software	O	The name of the software.
integration_details.version	O	The version number of the integration software.
integration_details.integration	O	Type of integration. (Plugin name or 'custom').

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
account_name	Processing Account ID that was used to process the transaction.
initial_amount	Amount used for the transaction.
standing_amount	Payment standing amount.
order_reference	Merchant's order ID.
email	Customer's email.
customer_ip	Customer's IP address.
customer_url	URL where the Customer should be redirected after completing

	<p>the <code>payment</code>, <code>payment_reference</code> and <code>order_reference</code> parameters are added when the customer is redirected to <code>customer_url</code>.</p> <p>Customer URL has to be a fully qualified domain name, it is not possible to use an IP address or localhost.</p>
<code>mobile_access_token</code>	Token to use POST/ <code>mobile_payments/card_details</code> endpoint. It will return if and only <code>mobile_payment</code> is sent as true.
<code>payment_created_at</code>	A time when the payment was initiated at Payment Gateway.
<code>payment_reference</code>	Reference ID of the payment.
<code>payment_link</code>	Link to complete payment. Used when payment needs user action (e.g. to fill card details or complete 3DS, etc).
<code>payment_state</code>	Current payment status.
<code>payment_methods</code>	List of available payment methods. Merchants can use these instead of <code>payment_link</code> to offer custom UI inside e-shop. These methods are returned together with <code>payment_link</code>
<code>payment_methods[#].source</code>	Source of payment method. See Payment Methods above for a list of possible values. Example: 'card'.
<code>payment_methods[#].display_name</code>	A display name for the Payment Method. Ex: 'Card Payment'.
<code>payment_methods[#].logo_url</code>	URL to fetch logo for this Payment Method.
<code>payment_methods[#].country_code</code>	Country code for this Payment Method, to be used for better UX (e.g. group Open Banking links by country). Omitted if not relevant for this payment method.
<code>payment_methods[#].payment_link</code>	Link to complete the payment with this specific payment method. If a customer is directly redirected to this link, he will skip payment method selection and continue to pay.

Example request

```
{
  "api_username": "abc12345",
  "account_name": "EUR3D1",
  "amount": 10.00,
  "order_reference": "912987",
  "token_agreement": "unscheduled",
  "nonce": "a9b7f7e794367c2c85d73154a01b9902",
  "timestamp": "2019-06-05T13:14:15+03:00",
  "email": "user@example.com",
  "customer_ip": "1.2.3.4",
  "customer_url": "https://shop.example.com/cart",
  "preferred_country": "EE",
  "billing_city": "Tartu",
}
```

```

"billing_country": "EE",
"billing_line1": "Main street 1",
"billing_line2": "Building 3",
"billing_line3": "Room 11",
"billing_postcode": "51009",
"billing_state": "EE",
"locale": "en",
"request_token": true,
"integration_details":
{
  "software": "magento",
  "version": "1.6.4",
  "integration": "plugin"
}
}

```

Example response

```

{
  "api_username": "abc12345"
  "account_name": "EUR3D1",
  "initial_amount": 10.00,
  "standing_amount": 10.00,
  "order_reference": "feiwph28qy8ks7i12i63",
  "email": "user@example.com",
  "customer_ip": "1.2.3.4",
  "customer_url": "www.abc.com/callback",
  "payment_created_at": "2019-06-05T13:14:15+03:00",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "payment_link": "https://igw-demo.every-pay.com/lp/aedf32/ed4dod",
  "payment_state": "initial"
  "payment_methods": [
    {
      "source": "card",
      "display_name": "VISA/Mastercard",
      "logo_url": "https://igw-demo.every-pay.com/assets/card_logo.png",
      "payment_link": "https://igw-demo.every-pay.com/lp/aedf32/ed4dod?method_source=card"
    },
    {
      "source": "ob_mybank_ee",
      "display_name": "MyBank Eesti",
      "country_code": "EE",
      "logo_url": "https://igw-demo.every-pay.com/assets/mybank_logo.png",
      "payment_link": "https://igw-demo.every-pay.com/lp/aedf32/ed4dod?method_source=ob_mybank_ee"
    }
  ]
}

```

3.6.2. POST /payments/mit

This endpoint is used to initiate MIT payment, i.e. a payment that will not ask for payment details (e.g. card number) it is related to previous One-Off payment.

ENDPOINT: /payments/mit

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
account_name	M	Processing account used for the payment. Most importantly, this will determine available payment methods and currency of the payment.
amount	M	Transaction amount, use decimal number with 2 digit precision, e.g. 10.55. The currency is taken from the specified processing account.
token_agreement	M	Type of the agreement. Valid values: `unscheduled` or `recurring` .
merchant_ip	M	The IP of the Merchant server (as Customer is not involved in the payment process).
order_reference	M	Merchant's order reference. Uniqueness requirements can be configured in Portal. Maximum length is 255 characters, unless otherwise defined by Acquirer bank/host. Allowed characters: [a-zA-Z0-9/-?:()'+]
nonce	M	Unique request identifier (see below for details).
email	OF	Customer's email. Used for Fraud Prevention.
timestamp	M	Timestamp of request's creation time (see below for details).
integration_details	O	Details of integration. Following fields are subfields of this.
integration_details.software	O	The name of the software.
integration_details.version	O	The version number of the integration software.
integration_details.integration	O	Type of integration.

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
account_name	Processing Account name that was used to process the transaction.
initial_amount	The initial payment amount.
standing_amount	Payment standing amount. (It might be different than the initial amount in the case that payment is refunded).
order_reference	Merchant's order ID.
email	Customer's email.
stan	Payment STAN number - a unique ID to identify payments on acquiring bank payment reports.
fraud_score	Payment fraud score.
warnings	Payment processing warnings in JSON format.
payment_created_at	A time when the payment was initiated at Payment Gateway.
payment_reference	Reference ID of the payments.
payment_state	Current status of the payment.

Example request

```
{
  "api_username": "abc12345",
  "account_name": "EUR3D1",
  "amount": 10.00,
  "order_reference": "912987",
  "token_agreement": "unscheduled",
  "nonce": "a9b7f7e794367c2c85d73154a01b9902",
  "timestamp": "2019-06-05T13:14:15+03:00",
  "merchant_ip": "5.6.7.8",
  "email": "user@example.com",
  "integration_details": {
    "software": "magento",
    "version": "1.6.4",
    "integration": "plugin"
  }
}
```

Example response

```
{
  "api_username": "abc12345",
  "account_name": "EUR3D1",
  "initial_amount": 10.00,
  "standing_amount": 10.00,
}
```

```

"order_reference": "feiwHP28qy8ks7i12i63",
"stan": null,
"email": "user@example.com",
"fraud_score": null,
"warnings": {},
"payment_created_at": "2019-06-05T13:14:15+03:00",
"payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
"payment_state": "initial"
}

```

3.6.3. POST /payments/cit

This endpoint is used to initiate CIT payments, i.e. a payment may ask for some payment details (e.g. CVC) related to previously completed One-Off payment.

ENDPOINT: /payments/cit

METHOD: POST

REQUEST PARAMS

Parameter	Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
account_name	M	Processing account used for the payment. Most importantly, this will determine available payment methods and currency of the payment.
amount	M	Transaction amount, use decimal number with 2 digit precision, e.g. 10.55. The currency is taken from the specified processing account.
token_agreement	M	Type of the agreement. Valid values: `unscheduled` .
order_reference	M	Merchant's order reference. Uniqueness requirements can be configured in Portal. Maximum length is 255 characters, unless otherwise defined by Acquirer bank/host. Allowed characters: [a-zA-Z0-9/-?:()., '+]
nonce	M	Unique request identifier (see below for details).
email	OF	Customer's email. Used for Fraud Prevention.
customer_ip	OF	Customer's IP address. Used for Fraud Prevention. Do not set this to some fixed value, e.g Merchant's server, as this will start generating false positives in Fraud Check.

customer_url	M	<p>URL where the Customer should be redirected after completing the payment. (If there is a customer involvement in the payment such as 3DS). <code>payment_reference</code> and <code>order_reference</code> parameters are added when a customer is redirected to <code>customer_url</code>.</p> <p>Customer URL has to be a fully qualified domain name, it is not possible to use an IP address or localhost.</p>
timestamp	M	Timestamp of request's creation time (see below for details).
locale	O	<p>A locale that should be used as the default for payment UI. Default is 'en'.</p> <p>Accepted values are:</p> <ul style="list-style-type: none"> • 'en' - English • 'et' - Estonian • 'fi' - Finnish • 'de' - German • 'lv' - Latvian • 'lt' - Lithuanian • 'ru' - Russian • 'es' - Spanish • 'sv' - Swedish • 'da' - Danish • 'pl' - Polish • 'it' - Italian • 'fr' - French • 'nl' - Dutch • 'pt' - Portuguese • 'no' - Norwegian • 'hu' - Hungarian • 'sk' - Slovak • 'cz' - Czech
billing_city	O	Cardholder Billing address city.
billing_country	O	Cardholder Billing address country. Format: ISO alpha-2.
billing_line1	O	Cardholder Billing address line 1.
billing_line2	O	Cardholder Billing address line 2.
billing_line3	O	Cardholder Billing address line 3.
billing_postcode	O	Cardholder Billing address postal code.
billing_state	O	Cardholder Billing address state. Format: ISO 3166-2.
shipping_city	O	Cardholder shipping address city. It will be sent if

		available.
shipping_country	O	Cardholder shipping address country. It will be sent if available. Format: ISO alpha-2.
shipping_line1	O	Cardholder shipping address line 1. It will be sent if available.
shipping_line2	O	Cardholder shipping address line 2. It will be sent if available.
shipping_line3	O	Cardholder shipping address line 3. It will be sent if available.
shipping_code	O	Cardholder shipping address postal code. It will be sent if available.
shipping_state	O	Cardholder shipping address state. It will be sent if available.
integration_details	O	Details of integration. Following fields are subfields of this.
integration_details. software	O	The name of the software.
integration_details. version	O	Version number of the integration software.
integration_details. integration	O	Type of integration.

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
account_name	Processing Account name that was used to process the transaction.
initial_amount	Initial amount used for the transaction.
standing_amount	Payment standing amount.
order_reference	Merchant's order ID.
email	Customer's email.
payment_method	What payment method was used. See above Payment Methods for valid values here (only card at the moment). Depending on

	the payment method, other fields in the response would be present or omitted. Returned when payment is completed.
customer_ip	Customer's IP address.
customer_url	URL where the Customer should be redirected after completing the payment, <code>payment_reference</code> and <code>order_reference</code> parameters are added when customer is redirected to <code>customer_url</code> . Customer URL has to be a fully qualified domain name, it is not possible to use an IP address or localhost.
payment_created_at	Time when the payment was initiated at Payment Gateway.
payment_reference	Reference ID of the payments.
payment_link	Link to complete payment. Used when payment needs user action (e.g. to fill card details or complete 3DS, etc).
payment_state	Current status of the payment.
fraud_score	Payment fraud score.
warnings	Payment processing warnings in JSON format.

Example request

```
{
  "api_username": "abc12345",
  "account_name": "EUR3D1",
  "amount": 10.00,
  "order_reference": "912987",
  "token_agreement": "unscheduled",
  "nonce": "a9b7f7e794367c2c85d73154a01b9902",
  "timestamp": "2019-06-05T13:14:15+03:00",
  "email": "user@example.com",
  "customer_ip": "1.2.3.4",
  "customer_url": "https://shop.example.com/cart",
  "billing_city": "Tartu",
  "billing_country": "EE",
  "billing_line1": "Main street 1",
  "billing_line2": "Building 3",
  "billing_line3": "Room 11",
  "billing_postcode": "51009",
  "billing_state": "EE",
  "integration_details": {
    "software": "magento",
    "version": "1.6.4",
    "integration": "plugin"
  }
}
```

Example response

```

{
  "api_username": "abc12345",
  "account_name": "EUR3D1",
  "initial_amount": 10.00,
  "standing_amount": 10.00,
  "order_reference": "feiwHP28qy8ks7i12i63",
  "stan": null,
  "email": "user@example.com",
  "customer_ip": "1.2.3.4",
  "customer_url": "https://shop.example.com/cart",
  "fraud_score": null,
  "warnings": {},
  "payment_created_at": "2019-06-05T13:14:15+03:00",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "payment_state": "initial",
  "payment_method": null
}

```

3.6.4. GET /payments/:payment_reference

ENDPOINT: /payments/:payment_reference

METHOD: GET

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
payment_reference	M	Reference ID of the payment.
detailed	O	Boolean to indicate that payment result should return fraudcheck details

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
account_name	Processing Account name that was used to process the transaction.
initial_amount	Initial amount used for the transaction.
standing_amount	Payment standing amount.

order_reference	Merchant's order ID.
email	Customer's email.
payment_method	What payment method was used. See above Payment Methods for valid values here. Depending on the payment method, other fields in the response would be present or omitted. Returned when payment is completed.
stan	Payment STAN number - a unique ID to identify payments on acquiring bank payment reports.
fraud_score	Payment fraud score.
warnings	Payment processing warnings in JSON format.
customer_ip	Customer's IP address.
customer_url	URL where the Customer should be redirected after completing the payment, <code>payment_reference</code> and <code>order_reference</code> parameters are added when customer is redirected to <code>customer_url</code> . Customer URL has to be a fully qualified domain name, it is not possible to use an IP address or localhost.
transaction_time	Time of the transaction.
payment_created_at	Time when the payment was initiated at Payment Gateway.
payment_reference	Reference ID of the payment.
payment_state	Current status of the payment. Possible values: "initial", "waiting_for_sca", "sent_for_processing", "waiting_for_3ds_response", "settled", "failed", "abandoned", "voided", "refunded", "chargebacked"
acquiring_completed_at	DateTime when payment acquiring was completed and funds received to the merchant's account. Available only if the acquiring bank supports this functionality. Only for OB payments.
sca_exemption	There are several transaction types out of the scope of strong customer authentication (SCA) or not requiring SCA with card payments. We call them SCA exemptions. We currently use the following exemptions - mit, recurring, one-leg
processing_error	Details for processing error
code	Processing error code
message	Processing error
cc_details	Details of card payment. Following fields are subfields of

	this
cc_details.token	Token referencing a bank card that can later be used to initiate recurring payments. It is returned only if the token was requested with <code>request_token</code> .
cc_details.last_four_digits	Last four digits of the card number.
cc_details.month	Card expiration month (mm format - 1-2 digits).
cc_details.year	Card expiration year (YYYY format - 4 digits).
cc_details.holder_name	Name on the card.
cc_details.type	Card type. Possible values are 'visa' or 'master_card'.
cc_details.issuer_country	Card issuer country. ISO 3166 two-letter (alpha-2) format (e.g. EE).
cc_details.issuer	Card issuing organization.
cc_details.cobrand	Name of the cobrand
cc_details.funding_source	Funding source of the card. (debit or credit)
cc_details.product	Product type of the card.
cc_details.state_3ds	3DS state of the transaction
cc_details.authorisation_code	Authorisation code of the transaction.
ob_details	Details for Open Banking payment. Following fields are subfields of this.
ob_details.debtor_iban	Customer IBAN which money is taken from (in case of SEPA payment).
ob_details.creditor_iban	Merchant IBAN which money is sent to.
ob_details.ob_payment_reference	Reference of the payment in the bank.
ob_details.ob_payment_state	State of the payment in the bank.
detailed_fraud_check_results	When parameter "detailed" is set "true" it gives list of all enabled fraud rules, their settings and scores regardless whether the rule was triggered or not (equivalent of "Fraud Check Results" in Portal payment details page)

Example request

```
/payments/db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a?api_username=abc12345
```

Example response

```
{
  "api_username": "abc12345",
  "account_name": "EUR1",
  "initial_amount": 10.00,
  "standing_amount": 10.00,
  "order_reference": "feiwHP28qy8ks7i12i63",
  "stan": "1234",
  "email": "user@example.com",
  "payment_method": "card",
  "cc_details": {
    "token": "d841bcc672b0f76523a7fa13",
    "last_four_digits": "1234",
    "month": "1",
    "year": "2017",
    "holder_name": "Tom Smith",
    "type": "master_card",
    "issuer_country": "EE",
    "issuer": "LHV Bank",
    "cobrand": "Partner deebet",
    "funding_source": "Debit",
    "product": "DEBIT STANDARD",
    "state_3ds": "no3ds",
    "authorization_code": "00590A"
  },
  "processing_error": {
    "code": null,
    "last_four_digits": null
  },
  "fraud_score": "500",
  "warnings": {
    "country_match": [
      "Card issuer country (Estonia) does not match the buyer country ()."
    ]
  },
  "customer_ip": "1.2.3.4",
  "customer_url": "https://customerurl.com",
  "transaction_time": "2019-06-05T13:15:20+03:00",
  "payment_created_at": "2019-06-05T13:14:15+03:00",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "payment_state": "settled"
}
```

3.6.5. POST /payments/void

ENDPOINT: /payments/void

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
payment_reference	M	Reference ID of the payment.
nonce	M	Unique request identifier (see below for details).
timestamp	M	Timestamp of request's creation time (see below for details).
reason	O	Reason to void the payment.

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
transaction_time	Time of the transaction.
payment_reference	Reference ID of the payment created by the completed transaction.
payment_state	Current state of the payment.

Example request

```
{
  "api_username": "abc12345",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "nonce": "a9b7f7e794367c2c85d73154a01b9902",
  "timestamp": "2019-06-05T13:14:15+03:00",
  "reason": "fraud suspicion"
}
```

Example response

```
{
  "api_username": "abc12345",
  "transaction_time": "2015-04-02T07:53:07Z",
}
```

```

    "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
    "payment_state": "voided"
}

```

3.6.6. POST /payments/capture

ENDPOINT: /payments/capture

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
amount	M	Amount to be captured, use a decimal number with 2 digit precision, e.g. 10.55.
payment_reference	M	Reference ID of the payment.
nonce	M	Unique request identifier (see below for details).
timestamp	M	Timestamp of request's creation time (see below for details).

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
initial_amount	Initial payment amount.
standing_amount	Standing amount after capture transaction.
transaction_time	Time of the transaction.
payment_reference	Reference ID of the payment created by the completed transaction.
payment_state	Current state of the payment.

Example request

```
{
  "api_username": "abc12345",
  "amount": 10.00,
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "nonce": "a9b7f7e794367c2c85d73154a01b9902",
  "timestamp": "2019-06-05T13:14:15+03:00"
}
```

Example response

```
{
  "api_username": "abc12345",
  "initial_amount": 10.00,
  "standing_amount": 10.00,
  "transaction_time": "2015-04-02T07:53:07Z",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "payment_state": "settled"
}
```

3.6.7. POST /payments/refund

This endpoint is used to refund card or open banking payments.

ENDPOINT: /payments/refund

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
amount	M	Amount to be refunded, use a decimal number with 2 digit precision, e.g. 10.55.
payment_reference	M	Reference ID of the payment.
nonce	M	Unique request identifier (see below for details).
timestamp	M	Timestamp of request's creation time (see below for details).
customer_name	C	The name of the customer to whom the money will be credited. Used only with open banking payments.

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
initial_amount	Initial payment amount.
standing_amount	Standing amount after refund transaction.
transaction_reference	Reference ID of the transaction.
transaction_time	Time of the transaction.
payment_reference	Reference ID of the payment created by the completed transaction.
payment_state	Current state of the payment.

Example request

```
{
  "api_username": "abc12345",
  "amount": "2.50",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "nonce": "a9b7f7e794367c2c85d73154a01b9902",
  "timestamp": "2019-06-05T13:14:15+03:00"
}
```

Example response

```
{
  "api_username": "abc12345",
  "initial_amount": "2.50",
  "standing_amount": "1.50",
  "transaction_time": "2015-04-02T07:53:07Z",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "transaction_reference": "c4ad248765023341e35377dc6c8176331089def474c9da7a2586177d45daffab",
  "payment_state": "refunded"
}
```

3.6.8. POST /payments/charge

ENDPOINT: /payments/charge

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
-----------	------------------------	-------------

api_username	M	The api username of the Merchant sending the request. Must match with username in Authorization HTTP header.
payment_reference	M	Reference ID of the payment.
cc_details	O	Details of card. Following bold fields are subfields of this. <u>It is mandatory to send for oneoff payment type, that is available for PCI DSS Compliant merchants</u>
cc_details.cc_number	M	Number on the card. Numeric string, 16 digits number.
cc_details.month	M	Card expiration month (mm format - 1-2 digits) Numeric string, values from 1-12.
cc_details.year	M	Card expiration year (YYYY format - 4 digits). Numeric string, values from 20-40.
cc_details.holder_name	M	Name on card. String, only letters.
cc_details.cvc	M	CVC number of the card. Numeric string, 3 or 4 digits.
token_details	O	It is mandatory to send token_details.token for CIT/MIT payments. Optional fields are needed when a token is shared between merchants.
token_details.token	M	The credit card Token to be shared to Token User.
token_details.token_owner	O	the api_username of Token Owner (as given in merchant portal)
token_details.token_timestamp	O	UNIX timestamp of the token sharing authorisation moment in UTC. This timestamp cannot be older than 5 minutes by the time the request reaches Gateway.
token_details.token_hmac	O	HMAC that proves to the Gateway that Token Owner does allow the sharing.
token_consented	O	It shows if the user consents to save the card details. Boolean: true, false.

nonce	M	Unique request identifier.
timestamp	M	A timestamp of request's creation time (see below for details)

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
account_name	Processing Account name that was used to process the transaction.
initial_amount	Initial amount used for the transaction.
standing_amount	Payment standing amount.
order_reference	Merchant's order ID.
email	Customer's email.
payment_method	What payment method was used. See above Payment Methods for valid values here (only card at the moment). Depending on the payment method, other fields in the response would be present or omitted. Returned when payment is completed.
customer_ip	Customer's IP address.
customer_url	URL where the Customer should be redirected after completing the payment, <code>payment_reference</code> and <code>order_reference</code> parameters are added when customer is redirected to <code>customer_url</code> . Customer URL has to be a fully qualified domain name, it is not possible to use an IP address or localhost.
payment_created_at	Time when the payment was initiated at Payment Gateway.
payment_reference	Reference ID of the payments.
payment_link	Link to complete payment. Used when payment needs user action (e.g. to fill card details or complete 3DS, etc).
payment_state	Current status of the payment.
fraud_score	Payment fraud score.
warnings	Payment processing warnings in JSON format.

cc_details	Details of card payment. Following fields are subfields of this.
cc_details.token	Token which is used in the payment.
cc_details.last_four_digits	Last four digits of the card number.
cc_details.month	Card expiration month (mm format - 1-2 digits).
cc_details.year	Card expiration year (YYYY format - 4 digits).
cc_details.holder_name	Name on card.
cc_details.type	Card type. Possible values are 'visa' or 'master_card'.
cc_details.issuer_country	Card issuer country. ISO 3166 two-letter (alpha-2) format (e.g. EE).
cc_details.issuer	Card issuing organization.
cc_details.cobrand	Name of the cobrand
cc_details.funding_source	Funding source of the card. (debit or credit)
cc_details.product	Product type of the card.
cc_details.state_3ds	3DS state of the transaction
cc_details.authorisation_code	Authorisation code of the transaction.

Example request

```
{
  "api_username": "abc12345",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "token_details": {
    "token": "1234567812345678"
  },
  "token_consented": true,
  "nonce": "xcncndkjsfkjdhuyr843y8579837458798327592hsfdkj",
  "timestamp": "2019-06-05T13:14:15+03:00"
}
```

Example response

```
{
  "api_username": "abc12345",
  "account_name": "EUR1",
  "initial_amount": 10.00,
  "standing_amount": 10.00,
  "order_reference": "feiwph28qy8ks7i12i63",
  "email": "user@example.com",
  "payment_method": "card",
  "payment_link": "https://igw-demo.every-pay.com/lp/aedf32/ed4dod",
  "payment_state": "waiting_for_3ds_response",
  "fraud_score": 325,
}
```



```

"warnings": {
  "country_match": [
    "Card issuer country (Estonia) does not match the buyer country ()."
  ]
},

"cc_details": {
  "token": "d841bcc672b0f76523a7fa13",
  "last_four_digits": "1234",
  "month": "1",
  "year": "2017",
  "holder_name": "Tom Smith",
  "type": "master_card",
  "issuer_country": "EE",
  "issuer": "LHV Bank",
  "cobrand": "Partner deebet",
  "funding_source": "Debit",
  "product": "DEBIT STANDARD",
  "state_3ds": "3ds",
  "authorization_code": "00590A"
},
"customer_ip": "1.2.3.4",
"customer_url": "https://shop.example.com/cart",
"payment_created_at": "2019-06-05T13:14:15+03:00",
"payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a"
}

```

3.6.9. POST /refunds/xml_payment_file

ENDPOINT: /refunds/xml_payment_file

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
Transaction_references	M	Reference ID of the refund transaction. Transaction references can be separated by a comma and the maximum amount of allowed transaction reference is 1500.

RESPONSE: XML pain 001 file.

3.6.10. POST /refunds/revert

ENDPOINT: /refunds/revert

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
payment_reference	M	Reference ID of the payment.
transaction_references	M	Reference ID of the refund transaction. Transaction references can be separated by a comma and the maximum amount of allowed transaction reference is 1500.
nonce	M	Unique request identifier.
timestamp	M	A timestamp of request's creation time (see below for details)

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
payment_reference	Reference ID of the payment.
initial_amount	Initial amount used for the transaction.
standing_amount	Payment standing amount.

payment_state	Current status of the payment. Possible values: "settled", "refunded"
---------------	---

Example request

```
{
  "api_username": "abc12345",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "transaction_reference": "abc8561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88abc1",
  "nonce": "xcncndkjsfkjdhuyr843y8579837458798327592hsfdkj",
  "timestamp": "2019-06-05T13:14:15+03:00"
}
```

Example response

```
{
  "api_username": "abc12345",
  "payment_reference": "db98561ec7a380d2e0872a34ffccdd0c4d2f2fd237b6d0ac22f88f52a",
  "initial_amount": 10.00,
  "standing_amount": 10.00,
  "payment_state": "settled"
}
```

3.6.11. GET /shops

ENDPOINT: /shops

METHOD: GET

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
shops	List of shops for a merchant that makes the request.
shops[#].id	Unique ID for the shop.

shops[#].url	Web address of the shop.
shops[#].descriptor	Descriptor of the shop.
shops[#].mcc	Merchant category code of the shop.

Example request

```
/shops/api_username=abc12345
```

Example response

```
{
  "api_username": "abc12345",
  "shops": [
    {
      "id": "1",
      "url": "https://electronicshop.com/",
      "descriptor": "Eesti Electronic Shop",
      "mcc": "1234"
    },
    {
      "id": "2",
      "url": "https://myshoes.com/",
      "descriptor": "My Shoes AS",
      "mcc": "1235"
    },
    {
      "id": "3",
      "url": "https://rentacar.com/",
      "descriptor": "My Car Rental",
      "mcc": "2222"
    }
  ]
}
```

3.6.12. GET /shops/:id

ENDPOINT: /shops/:id

METHOD: GET

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
id	M	Unique ID for the shop.

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
id	Unique ID for the shop.
url	Web address of the shop.
descriptor	Descriptor of the shop.
mcc	Merchant category code of the shop.
processing_accounts	List of processing accounts belong to the shop.
processing_accounts[#].account_name	Processing Account name that was used to process the transaction.
processing_accounts[#].pre_authorisation	Type of authorization. Pre or Final Authorization.False means it is not pre_authorisation.
processing_accounts[#].currency	Currency of processing account.
processing_accounts[#].support_3ds	It shows if the processing account supports 3ds or not.
processing_accounts[#].capture_delay_days	It shows the delay days for automatic capture of the transaction.
processing_accounts[#].bav_required	It shows if bank account verification is necessary or not.

Example request

```
/shops/id=8?api_username=abc12345
```

Example response

```
{  
  "api_username": "abc12345",  
  "id": "8",
```

```

"url": "https://electronicshop.com/",
"descriptor": "Eesti Electronic Shop",
"mcc": "1234",
"processing_accounts": [
  {
    "account_name": "EUR3D1",
    "pre_authorisation": "true",
    "currency": "EUR",
    "support_3ds": "true",
    "capture_delay_days": "0",
    "bav_required": "false",
  },
  {
    "account_name": "EUR1",
    "pre_authorisation": "false",
    "currency": "EUR",
    "support_3ds": "false",
    "capture_delay_days": "3",
    "bav_required": "false",
  },
  {
    "account_name": "USD3D1",
    "pre_authorisation": "false",
    "currency": "USD",
    "support_3ds": "true",
    "capture_delay_days": "0",
    "bav_required": "false",
  },
  {
    "account_name": "AUD1",
    "pre_authorisation": "false",
    "currency": "AUD",
    "support_3ds": "false",
    "capture_delay_days": "0",
    "bav_required": "false",
  }
}
]]

```

3.6.13. GET /processing_accounts/:account_name

ENDPOINT: /processing_accounts/:account_name

METHOD: GET

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
account_name	M	Processing Account name that was used to process the transaction.

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
account_name	Processing Account name that was used to process the transaction.
pre_auth	Type of authorization. Pre or Final Authorization.
currency	Currency of processing account.
support_3ds	It shows if the processing account supports 3ds or not.
capture_delay_days	It shows the delay days for automatic capture of the transaction.
bav_required	It shows if bank account verification is necessary or not.
payment_methods	List of available payment methods for the processing account.
payment_methods[#].source	Source of payment method. See Payment Methods above for a list of possible values. Example: 'card'.
payment_methods[#].display_name	Display name for the Payment Method. Ex: 'Card Payment'.
payment_methods[#].logo_url	URL to fetch logo for this Payment Method.
payment_methods[#].country_code	Country code for this Payment Method, to be used for better UX (e.g. group Open Banking links by country). Omitted if not relevant for this payment method.
payment_methods[#].card_acceptor_id	Only for card payments. It is the terminal name.

payment_methods[#].terminal_id	Only for card payments.
--------------------------------	-------------------------

Example request

```
/processing_accounts/EUR3D2?api_username=abc12345
```

Example response

```
{
  "api_username": "abc12345",
  "account_name": "EUR3D2",
  "pre_auth": "final",
  "currency": "EUR",
  "support_3ds": "true",
  "capture_delay_days": "0",
  "bav_required": "false",
  "payment_methods": [
    {
      "source": "card",
      "display_name": "VISA/Mastercard",
      "logo_url": "https://igw-demo.every-pay.com/assets/card_logo.png",
      "card_acceptor_id": "10234957 100",
      "terminal_id": "EVR20797"
    },
    {
      "source": "ob_mybank_ee",
      "display_name": "MyBank Eesti",
      "country_code": "EE",
      "logo_url": "https://igw-demo.every-pay.com/assets/mybank_logo.png"
    }
  ]
}
```

3.6.14. POST /mobile_payments/card_details

NB! Merchant should validate information about card details before sending it to the gateway. Validation information can be found under description.

ENDPOINT: /mobile_payments/card_details

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/Optional	Description
-----------	--------------------	-------------

api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
mobile_access_token	M	This is a header parameter. Token to use this endpoint. It will be taken in payments/oneoff response. Format: 'Bearer mobile_access_token' Ex: Authorization: 'Bearer bb760db84c9b53dc4bbcc7cbe799fd0f1237d56d'
cc_details		Details of card. Following bold fields are subfields of this.
cc_details.cc_number	M	Number on the card. Numeric string, 16 digits number.
cc_details.month	M	Card expiration month (mm format - 1-2 digits) Numeric string, values from 1-12.
cc_details.year	M	Card expiration year (YYYY format - 4 digits). Numeric string, values from 20-40.
cc_details.holder_name	M	Name on card. String, only letters.
cc_details.cvc	M	CVC number of the card. Numeric string, 3 or 4 digits.
token_consented	M	It shows if the user consents to save the card details. Boolean: true, false.
timestamp	M	Timestamp of request's creation time (see below for details).

RESPONSE PARAMS

Parameter	Description
payment_state	The status of the payment. Possible values; "initial", "settled", "failed", "waiting_for_3ds_response", "waiting_for_sca" and "confirmed_3ds".
processing_errors	Related error codes and messages if there is any.

Example request

Headers:

Authorization: 'Bearer bb760db84c9b53dc4bbcc7cbe799fd0f1237d56d'

Content-Type: 'application/json'

```
{
  "api_username": "abc12345",
  "mobile_access_token": "Bearer bb760db84c9b53dc4bbcc7cbe799fd0f1237d56d",
  "payment_reference": "abc12345sdjkhfhuweyruirkdj",
  "cc_details": {
    "cc_number": "1234567812345678",
    "month": "11",
    "year": "2019",
    "holder_name": "Tom Smith",
    "cvc": "100"
  },
  "token_consented": true,
  "timestamp": "2019-06-05T13:14:15+03:00"
}
```

Example response

```
{
  "payment_state": "waiting_for_3ds_response",
  "processing_errors": []
}
```

3.7. Token Endpoints

3.7.1. POST /tokens/delete

It is used to delete existing ACTIVE/SUSPENDED tokens.

ENDPOINT: /tokens/delete

METHOD: POST

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
token	M	Instrument token which is got from <code>payments/:payment reference</code> endpoint.

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
token	Instrument token which is sent in the request.
status	The status of instrument token
warning	Processing warnings.

Example request

```
{
  "api_username": "abc12345",
  "token": "d841bcc672b0f76523a7fa13",
}
```

Example response

```
{
  "api_username": "abc12345",
  "token": "d841bcc672b0f76523a7fa13",
  "status": "DEACTIVATED"
}
```

3.7.2. GET /tokens/get_card_art

It is used to get card art for the existing tokens.

ENDPOINT: /tokens/get_card_art

METHOD: GET

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.
token	M	Instrument token which is got from <code>payments/:payment reference endpoint</code> .

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
token	Instrument token which is sent in the request.
media_contents	Media content of instrument token.
media_contents.data	Data to receive media content.
media_contents.type	Type of media content.

Example request

```
{
  "api_username": "abc12345",
  "token": "d841bcc672b0f76523a7fa13",
}
```

Example response

```
{
  "api_username": "abc12345",
  "token": "d841bcc672b0f76523a7fa13",
  "media_contents": {
    "data": "https://s3-eu-west-1.amazonaws.com/production-ups-logos/images/12/large/4d32c959-094b-47ab-8a4e-44862fce351c.png?1606925690",
    "type": "image/png"
  }
}
```

3.7.3. GET /tokens/status

It is used to get the status of the tokens.

ENDPOINT: /tokens/status

METHOD: GET

REQUEST PARAMS

Parameter	Mandatory/ Optional	Description
api_username	M	The api_username of the Merchant sending the request. Must match with username in Authorization HTTP header.

token	M	Instrument token which is got from payment/:payment reference endpoint.
-------	---	---

RESPONSE PARAMS

Parameter	Description
api_username	The api_username of the Merchant sending the request.
token	Instrument token which is sent in the request.
status	The status of instrument token.
card_art	Media content of instrument token.
last_four_digits	Last four digits of the tokenized card number.
month	Tokenized card's expiry month.
year	Tokenized card's expiry year.
holder_name	The name of the tokenized card owner.

Example request

```
{
  "api_username": "abc12345",
  "token": "d841bcc672b0f76523a7fa13",
}
```

Example response

```
{
  "api_username": "abc12345",
  "token": "d841bcc672b0f76523a7fa13",
  "status": "ACTIVE"
  "card_art": "null"
  "last_four_digits": "1234"
  "month": "12"
  "year": "2020"
  "holder_name": "John Doe"
}
```

Version history

Date	Change
23.08.2023	Updated allowed characters for order_reference and payment_description: [a-zA-Z0-9/-?:(),'+]
09.12.2022	Allowed characters for order_reference and payment_description: [a-zA-Z0-9/-?:(),'+]
09.03.2022	<code>/tokens/status</code> endpoint release. <code>/tokens/suspend</code> and <code>/tokens/unsuspend</code> endpoints are deleted. <code>/tokens</code> endpoints update: account_name parameter is removed from the requests and responses.
14.12.2021	New optional parameter payment_description for <code>payments/oneoff</code> endpoint.
09.12.2021	Locale parameter to CIT payments
01.12.2021	order_reference should not be used when callback is sent. It will be removed in the next API version from callbacks.
22.10.2021	Some small wording improvements
14.10.2021	HMAC calculation details
27.09.2021	Description and new endpoints that are related to open banking refunds.
24.08.2021	Initial document of the APIv4.