Your APIs for us

We can send following information to your API endpoints:

- 3DS OTP code, so you can handle delivery to the user yourself via SMS, Push or other channel.
- Notification about outcome of KYC process.
- Transaction request, so you can keep users balance on your end and accept or reject transactions their transactions (we call this "External Balance").
- Simple notification about transactions.

To make this work, you need to expose an API according to relevant section of this documentation.

> To use external balance you must have Banking License or Payment Institution License. Additionally if you don't directly own BIN range, total sum of transactions of your users will be limited by deposit called "Master Balance".

Security

To set secured server-server connection between our services Verestro requires a similar connection as in the case of client to Verestro communication based on the x509 certificate, more about this model here.
In the first step, Verestro will send to the client a CSR for the dev and production environments, more details here.
The next step is for the client to sign the CSR and send the certificate back to Verestro along with the base URL for the methods listed below. Verestro will authorize itself with each request with a certificate, which should be checked on the client side.

Additional data encryption & integration

Some requests and responses contain sensitive data, to additionally secure the connection we require JSON Web Encryption (JWE).

	normal	encrypted
Example of request with sensitive data	`{ "cardNo" : "5555444455554444" }`	`{ "payload" : "very long JWE token" }`

	normal	encrypted
Example ofresponse with sensitive data	`{ "id" : 1125, "type" : "1125" , "cvv" : "123" , "cardNo" : "5555444455554444" , "exp" : "2026-01-31" }`	`{ "payload" : "very long JWE token" }`

Idempotency Key

With some requests additional header X-Idempotency-Key could be send. This header contain unique random id allowing to identify single request.
If client send this header, operation should be triggered only once and for any further request with this key, response should be identical - in most cases, returned from cache.

example headers:

```
X-Idempotency-Key: 20e87975-dbfb-4c95-b239-169516c0b707
```

JWE configuration

connection we need from you enc and alg from JWE parameters. Acceptable values are:

- Algorithm used by Verestro to encipher content of message (enc) - A256GCM,
- Algorithm used by Verestro to encipher encryption key (alg) - RSA-OAEP-256,
- Algorithm needed from you to encipher content of message (enc) - A256GCM,
- Allowed algorithms for key encryption (alg) - RSA-OAEP-256 or RSA-OAEP.

Recommended JWE libraries for various programming languages:

- PHP,
- JAVA.

Request:

To process encrypted message you need to perform a few additional steps on top of standard message processing:

- Verestro add headlines:

    - Public-Key through which you can encrypt response to us (if needed),
    - Encrypted-Request headline confirming message encryption; value true or false,
- Expose endpoint with your Public Key - you will find a similar endpoint in our technical documentation, GET /secure/public_key
- Use Verestro Public Key to create JWE and transfer data in payload,

Response:

When the response contains sensitive data that requires encryption, use Verestro public key encryption available here GET /secure/public_key

Additional information:

- In case of errors (i.e. validation errors) you will receive unencrypted response,
    - ENCRYPTION_REQUIRED,
    - INVALID_PUBLIC_KEY,
    - INVALID_PAYLOAD,
    - CANT_DECRYPT_PAYLOAD.

Example request:

Correct request	Sent request (incorrect)	Received by CMS Antaca (after decipher action with private key)

`{"card_no" : 1337}`	{"payload" : "eyJhbGciOiJSU0EtT0FFUC0yNTYiLCJlbmMiOiJBMjU2R0NNIn0.rdUrW12XCZQgLFDJ-2zAHWYYnaAanctceE1-Y6yJUplX0B2dLu-bvYOEJ83KxxUs-ZjA41R4PmAVilx1cTF4pv-7CZR0_ki85XRATBYF2-MvZdcC81fHy2QPU_ZsAEWAW00a1wKJmuEsgPB2m1aLZ7oK4fC1hciep4PyAtuWQRYHjhNb-UDT41_gDKTbnSGTwheL7S0mAJ_HsKfnZFHYUrM77UcxQGZKnH7Mzqvndf9THiMo0-3MWliYFDAm1bqN2_KTIoBNCprYjFnyIXPCjib73bjWX_P2ip5Ul84cngbQmFVzc7o91JrpJvYou1INS7zL4XKLFcADN4nZ_9ePWsm5_kX5SOMyUyEhOC9gusrLNAJ0MHaIFHni8WqnMAWM3_MC4OQDYetKax5bnHK6x42_5eFaf6ZmzmioKny5aGm-4Vo8TEu691FmPxglhyenWlMhvBvf6ZeVsy58Ofr0mi3TXjwYbAyas7m6sncxZu1FhEJ4da6gtNjmjuKdikOOntu8V71QQ07nczNqfGlUv0RcUc9uKJq5je4b9BEbK9WuQcroxmALqC4HTt1xhICHrVUA0d_t3fglhS2n7wNaKKCFq70ZWIrpdTaBd35kdVQOEjZgCavSjbZOzgOzcEqS6P2Blm7bZ7ZZBmnfk8y8M4m0xWoQNTmLC6nqz9bSbME.UEryKNClDxQZpyWu.6Lw_5CcZ9HiVxHfi_XTAFw.pYbQ6tdmQYe1kiPonm1GhA"}	`{"card_no" : 1337}`

Example response:

Correct response	Sent response by CMS Antaca	Received by you

`{"card_no" : 1338}`	{"payload" : "eyJhbGciOiJSU0EtT0FFUC0yNTYiLCJlbmMiOiJBMjU2R0NNIn0.iPmvEKtMAMrrEiR89vlwsL77ZfqxXrcMiy-bx3z6_7HAo__aQzBpMVDtLyj3kTHYWxen8bhPuVyebXyaIHL20sekFzcIFFzvaGoyQYU6zOK8tPv81tgixQe8SDnEr5v9VWBfiHxtPvqlpQIig2is5ynBkyqjdpQWEagR3MpqpATGl7f-omG82Jq0OwZByWI8I6P89hczwgK37F-MUnQDxcRUM3RagbHKNeIcfmPdJpNeqFZHe45y4wUkTWN0uzW72qydkN_4uM9fy0nrUpgsJNbtJGAVIUVmDz4pIZkiI1zyGbfZX-PT7Wh9UNM06gEUf4i2goZY-m4wPB0n2zXvxzcEdfTH27iPp-aKiJjfJpYb_ZnHyklk__gZlAy9r7W0594dY-eBJ_iUa5aeDsFS2TIfsfjMJsL8NRWY2noiTw5lsneD8dwvr6N_rYcWoFXDyWXHoRitSSd2iYrB80gbeSOBW0wfKtPxNIZrR0uDhkE8FouS5Pk7QBw412kd43GtrEpAijqn3ne7MNUpCtuNfJ8e_NdGDLTR7CSHhC0jfFlchpIvklF42o216NO-OnyJsjdv1w4_w1ugs61fTHDl8lgBalOjOxauKwIvJJOyFdWmpjlXuzJhrray7ov25uh2ibvFv3Gfd2iuGUnLIZzYBOTT8ftGWTCGXTDvVOvzGbs.c3qMNb2Bne-7g0Wz.PInghFM6Q8Gn0p4Tlebig32s-ZrpLqTMqQDlpXLLYx0iq-StrKco_HrjdN4MxondP4CicCgseIjcV8JR29jKYX-nqKdchEYq_vVIzFHcNI_Mx7y1el192QbMyx6b0Gbj5L79wpuB7qCUqTBNhJZ2c07PuyPsewcNwglvnc-OrA-2vL6lJnBi5ZGH8gBH1cZCgmbrMpZGNFPG3oFpOn9JPzmnvQxe9tvSFFj5989A8d_XMHP-ZQ.dJZxnBRxJeMKswDsCA3cXA"}	Check yourself by using Private Key included in the response.

Public Key

This method is used to share your public key for encryption.

```
GET https://server-domain.com/public-key
```

Headers:

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

response:

```
200 OK
{
  "publicKey": "QSBwdWJsaWMga2V5IHNob3VsZCBiZSBoZXJlIGhvd2V2ZXIgaXQgd2FzIHRvbyBsb25nIDoo"
}
```

3DS External OTP Notifier

This document describes API for external OTP notifier handling. Clients that are interested into having OTP notifier on their side must have implement this API to allow communication with Antaca to provide one time password about the transaction to client own users.

API 3DS External OTP Notifier

Below you will find a list of endpoints that you should implement on your server side. Please pay special attention to the appropriate security of our connection, the syntax of requests that you can expect from the Verestro side, idempotency and the exact way in which you should respond to each request.

> These notifications support sending **Idempotency Key**

Notification OTP

This method is used to transfer a one-time password generated for transactions without a card present in the 3DS standard.

```
POST https://server-domain.com/notifications/otp
```

Headers:

```
Content-Type: application/json
X-Idempotency-Key: 20e87975-dbfb-4c95-b239-169516c0b707
```

request body:

```
{
    "storageCustomerId" => "1337",
    "storageCardId" => "1337",
    "balanceId" => "b334b384-328c-11ed-a261-0242ac120002",
    "amount" => "1000",
    "currency" => "PLN",
    "merchantName" => "merchant test",
    "otp" => "1111"
}
```

Parameters:

Parameter	Required	Description	Type
storageCustomerId	TRUE	Customer identifier	integer value
storageCardId	TRUE	Card identifier	integer value
balanceId	TRUE	User balance identifier	uuid v4
amount	TRUE	Transaction value in gross (minor value)	integer value
currency	TRUE	Currency 3-letters code in ISO 4217 https://www.iban.com/currency-codes	ISO 4217 3-letter code
merchantName	TRUE	Merchant name	string value
otp	TRUE	One time password	string value

success response:

```
204 No Content
```

error responses:

```
Code 422
{
    "detail": "some specific details provided by server"
}
```

External Verification Notifier

This document describes API for processed KYC verification notifier handling. Clients that are interested into having information about status KYC verification on their side must have implement this API to allow communication with Antaca.

> Notifier provide notifications only with internal KYC status processes

> These notifications support sending **Idempotency Key**

Notification verification In-progress

This method is used to transfer information about changed KYC verification status to 'IN_PROGRESS'.

```
POST https://server-domain.com/notifications/verificationInProgress
```

Headers:

```
Content-Type: application/json
X-Idempotency-Key: 20e87975-dbfb-4c95-b239-169516c0b707
```

request body:

```
{
  "verificationId": "6faaa45a-41f6-4922-95fe-16e316ba7e91",
   "userId": "1337",
"email": "leonbakiewicz@gmail.com",
   "firstName": "Leon",
   "lastName": "Bakiewicz",
   "status": "IN_PROGRESS",
  "reason": null,
}
```

response:

```
204 No Content
```

Notification verification accepted

This method is used to transfer information about changed KYC verification status to 'ACCEPTED'.

```
POST https://server-domain.com/notifications/verificationAccepted
```

Headers:

```
Content-Type: application/json
X-Idempotency-Key: 20e87975-dbfb-4c95-b239-169516c0b707
```

request body:

```
{
  "verificationId": "6faaa45a-41f6-4922-95fe-16e316ba7e91",
    "userId": "1337",
 "email": "leonbakiewicz@gmail.com",
    "firstName": "Leon",
    "lastName": "Bakiewicz",
    "status": "ACCEPTED",
  "reason": null,
}
```

response:

```
204 No Content
```

Notification verification rejected

This method is used to transfer information about changed KYC verification status to 'REJECTED'.

```
POST https://server-domain.com/notifications/verificationRejected
```

Headers:

```
Content-Type: application/json
X-Idempotency-Key: 20e87975-dbfb-4c95-b239-169516c0b707
```

request body:

```
{
  "verificationId": "6faaa45a-41f6-4922-95fe-16e316ba7e91",
    "userId": "1337",
 "email": "leonbakiewicz@gmail.com",
    "firstName": "Leon",
    "lastName": "Bakiewicz",
    "status": "REJECTED",
  "reason": 'INVALID_CUSTOMER_DATA',
}
```

response:

```
204 No Content
```

Parameters:

Parameter	Required	Description	Type
verificationId	TRUE	Verification identifier	uuid v4
userId	TRUE	User identifier	integer value
email	TRUE	User's email address	string value
firstName	TRUE	User first name	string value
lastName	TRUE	User last name	string value
status	TRUE	Verification status. Possible values: • REJECTED • IN_PROGRESS • ACCEPTED	string value

Parameter	Required	Description	Type
reason	TRUE	Verification status reason **ACCEPTED**: null **IN_PROGRESS**: null **REJECTED**: • INVALID_CUSTOMER_DATA • BLURRED_DOCUMENT_PHOTO • INVALID_DOCUMENT_PHOTO • BLURRED_SELFIE • INVALID_SELFIE	null/string value

Sensitive data:

This method is used to share your public key for encryption.

```
GET https://server-domain.com/public-key
```

response:

```
200 OK
{
  "publicKey": "QSBwdWJsaWMga2V5IHNob3VsZCBiZSBoZXJlIGhvd2V2ZXIgaXQgd2FzIHRvbyBsb25nIDoo"
}
```

External Balance

External Balance API is used, when client wishes to keep end users’ balances on their side. Thanks to this API, a client who maintains his clients accounts or has his own business logic affecting transaction authorizations has the opportunity to expand his offer with various payment instruments offered by Verestro, including payment cards, bank transfers, transfers to a telephone number and others. Workflow is reversed when using External Balance API - Antaca is sending request to server on client’s side.

Features

• linking (connecting) balance with customer in Antaca,

• getting list of balances,
• deleting balance link (connection),
• handling transactions,
• updating transaction status.

Purpose and scope

This guide provides an instruction and case study for using Extrnal Balance API. Document covers following topics:
• how to use External Balance API,
• transaction flow,
• how clearings are handled,
• use cases study.

Terminology

User - The end user for whom a balance is maintained along with the associated payment instruments.
Server - API exposed by Antaca’s client.
Client - company using Antaca services.

External Balance API

Below you will find a list of endpoints that you should implement on your server side. Please pay special attention to the appropriate security of our connection, the syntax of requests that you can expect from the Verestro side, idempotency and the exact way in which you should respond to each request.

Process of linking balances

After establishing secure connection, the client should create balance aliases for their users on the Antaca side. For identifiers created in this way, Antaca will be able to generate payment cards and process transactions. When the client creates a user on the Verestro side with confirmed KYC status and then orders the creation of a balance for him, this API will forward the request to link the balance to the user. The linking process is presented in the diagram below.

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
participant "Client server" as cs
participant Antaca as a
participant Lifecycle as lc
cs->lc: 1. create user by POST /wallet (firstName, lastName, phone, email)
lc-->cs: 2. userId
alt With own KYC process
cs->lc: 3. update KYC status by PUT /user (KYC)
else With Verestro KYC process
cs->a: 4. send KYC data by /register (user data with documents and selfie)
a->a: 5. process KYC
a->lc: 6. update KYC status
end
alt With automatic balance creation
lc-->a: 7. event about the new user with KYC
a->a: 8. create a balance in the default currency
else Every time with create a balance
cs->a: 9. create balance by POST /balance (userId, currency)
end
a->cs: 10. link balance (userId, balanceId, currency)
cs-->a: 11. 204 (OK)
@enduml
```

Once balances are linked, Antaca can:
1. send a GET requests to retrieve information about specific balance. Using user ID and/or balance ID, Antaca can obtain information about balance currency and money amount.
2. send a GET requests to retrieve information about all user’s balances. Using only user ID Antaca can retrieve list of users balances.
3. delete link between balances
When user’s balance is equal to 0, it can be unlinked and deleted on Antaca side.

> **Remember to avoid billing problems**
> Deleting a balance is only possible when its status is 0. This also applies to situations in which a user with at least one balance is deleted.

Transaction processing

When a balance is created and linked for a given user with verified KYC, from that moment the client-side API should be ready to accept transactions related to it. Depending on the payment instrument used, the data transferred in the transaction object may be different, but will always refer to a specific balance.

> **Remember to avoid communication errors**
> Verestro servers attach an X-Idempotency-Key to each request in the header. This header contains a unique ID for each request to ensure idempotence. Each request with a unique identifier in this header should be processed only once on the client side and the response to it should be identical - in most cases, returned from cache

example:

```
curl POST "https://server-domain.com/transactions/debit"
--header "X-Idempotency-Key: 21aa0c2a-5554-4071-bd48-b9c64a0b6270"
```

Transaction object

Each transaction request contains following data:

```
{
    "id": "b4f534ef-77c2-4f16-ab4d-496806a76fb6",
    "balanceId": "b334b384-328c-11ed-a261-0242ac120002",
    "resourceId": "9d673932-3291-11ed-a261-0242ac120002",
    "resource": "card",
    "transactionId": "ab3d89e4-3291-11ed-a261-0242ac120002",
    "referenceTransactionId": "b759931c-3291-11ed-a261-0242ac120002",
    "type": "POS",
    "amount": 10000,
    "currency": "PLN",
    "originalAmount": 10000,
    "originalCurrency": "PLN",
    "status": "AUTHORIZED",
    "description": "transaction description",
    "date": "2020-08-17T18:43:42+00:00",
    "transactionData": {
        "mcc": "5942",
        "merchantIdentifier": "003060300000005",
        "merchantName": "Book store",
        "captureMode": "NFC",
        "lastFourDigits": "4560",
        "acquirerCountry": "PL",
        "mdesDigitizedWalletId": "Google Pay",
        "cashbackPosCurrencyCode": "PLN",
        "cashbackPosAmount": 10000,
        "paymentTokenLastFourDigits": "7890"
    }
}
```

Parameters:

Parameter	Required	Description	Allowed values
id	TRUE	Unique identifier of the transaction in UUID format	any value in UUID v4 format, eg. ddb55ff9-11ca-4621-9129-81f939e66011
balanceId	TRUE	Unique user balance identifier	any string value (recommended uuid v4)
resourceId	TRUE	Unique resource identifier	any string value (recommended uuid v4)
resource	TRUE	Name of a resource.	balance, card
transactionId	TRUE	Transaction identifier obtained from card netwok or generated on client side using the method to generate transaction in Antaca. **IMPORTANT: this id may not be unique - it is generated by different systems**	any string value (recommended uuid v4)
referenceTransactionId	FALSE	Id of previous transaction to witch current request relates	any string value (recommended uuid v4)
type	TRUE	Type of transaction	cashback, loan, payment, topup, commision, fee, funding, interest, withdrawal, POS, ATM, Cashback at POS
amount	TRUE	Transaction value in gross (minor value)	integer value
currency	TRUE	Currency 3-letters code in ISO 4217	ISO 4217 3-letter code

originalAmount	FALSE	Original transaction value in gross (minor value)	integer value
originalCurrency	FALSE	Original currency 3-letters code in ISO 4217	ISO 4217 3-letter code
status	TRUE	Transaction status	AUTHORIZED, CLEARED, REVERSED
description	TRUE	Transaction description	any string value
date	TRUE	Date of transaction in UTC	date in UTC
transactionData	FALSE	Additional transaction data. This object presents detailed data depending on the transaction type	This object is described below

TransactionData data object:

Name	Required	Description	Allowed values
mcc	FALSE	Merchant category code	any mcc value, eg. can be found here: https://global.alipay.com/docs/ac/files/mcclist
merchantIdentifier	FALSE	The merchant identifier for the transaction	
merchantName	FALSE	Name of merchant	
captureMode	FALSE	Capture mode	magstripe, manual, emv, on behalf (EMV), nfc, ecommerce, adj
lastFourDigits	FALSE	Last 4 digits of card	
acquirerCountry	FALSE	Country of acquirer	
mdesDigitizedWalletId	FALSE	The Wallet ID (Wallet Reference) used to digitize the card.	m4m, google pay, samsung pay, apple pay
cashbackPosCurrencyCode	FALSE	Represents the currency code of the cashback amount	ISO 4217 3-letter code
cashbackPosAmount	FALSE	Displays the actual cashback amount	integer value in gross

Name	Required	Description	Allowed values
paymentTokenLastFourDigits	FALSE	Last 4 digits of Device Primary Account Number (tokenized PAN)	

Integration with External Balance

By using external balance API, you take an active part in processes affecting settlements, therefore the API issued by you will be subject to approval before production starts. To make this process easier for you, below you will find a postman collection with test cases.

API External balance

Below you will find a list of endpoints that you should implement on your server side. Please pay special attention to the appropriate security of our connection, the syntax of requests that you can expect from the Verestro side, idempotency and the exact way in which you should respond to each request. If you decide to implement external balance to be able to keep the balance on your side and authorize transactions, remember that the implementation of all the methods below is required to ensure the API works.

Link balance

This method is used to link customer balance between client and server. Requested **balanceId** will be used for communication between client and Verestro side.

```
POST https://server-domain.com/users/:id/balances
```

path parameters:

id - user identifier

Headers:

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

request body:

```
{
    "balanceId":"2e520dc2-329d-11ed-a261-0242ac120002",
    "currency": "PLN"
}
```

parameters:

```
balanceId - unique identifier of balance. This id will be used in communication between client and server.
currency - required, should be 3 letters code in ISO 4217 https://www.iban.com/currency-codes
```

response:

```
204 NoContent
```

error codes:
404 - should be returned if no user has been matched by requested path parameter.

```
Code 404
{
    "title": "USER_NOT_FOUND",
    "detail": "some specific details provided by server"
}
```

Get single user balance

Method used to obtain single user balance information.

```
GET https://server-domain.com/users/:id/balances/:balanceId
```

path parameters:

```
id - user identifier
balanceId - unique balance identifier
```

headers:

```
Accept: application/json
```

response:

```
200 OK
{
    "currency": "PLN",
    "amount": 250
}
```

response parameters:

```
currency  - three letter iso 4217 code
amount - actual balance amount in minor (penny), numeric value
```

error codes:
404 - should be returned if no balance found by requested balanceId.

```
Code 404
{
   "title": "BALANCE_NOT_FOUND",
   "detail": "some specific details provided by server"
}
```

403 - if requested balance does not belong to user.

```
Code 403
{
   "title": "FORBIDDEN",
   "detail": "some specific details provided by server"
}
```

Get balance collection

This method should return collection of customer balances.

```
GET https://server-domain.com/users/:id/balances
```

path parameters:

```
id - user identifier
```

headers:

```
Accept: application/json
```

response:

```
200 OK
[
   {
        "id": "a072bd0e-328c-11ed-a261-0242ac120001",
        "currency": "PLN",
        "amount": 250
    },
    {
        "id": "b334a5e2-328c-11ed-a261-0242ac120002",
        "currency": "USD",
        "amount": 460
    }
]
```

If user has not created any balance yet, there should be returned empty collection.

```
200 OK
[]
```

response parameters:

```
id - unique identifier of user balance
currency  - three letter iso 4217 code
amount - actual balance amount in minor (penny), numeric value
```

Delete balance

This method is used to unattached balance from user. From legal point of view, balance should be deleted only if there is no money on it.

```
DELETE https://server-domain.com/users/:id/balances/:balanceId
```

path parameters:

```
id - user identifier
balanceId - unique balance identifier
```

response:

```
204 No Content
```

error responses:

404 - if requested balance has not been found.

```
Code 404
{
   "title": "BALANCE_NOT_FOUND",
   "detail": "some specific details provided by server"
}
```

403 - if requested balance does not belong to user.

```
Code 403
{
   "title": "FORBIDDEN",
   "detail": "some specific details provided by server"
}
```

Debit transaction

This kind of transaction is used to authorize transaction. In debit transactions Antaca asks 'if user has money?'.

```
POST https://server-domain.com/transactions/debit
```

headers:

```
Content-Type: application/json
X-Idempotency-Key: uuidV4
```

request body:

Description of the contents of the transaction object can be found above.

success response:

```
204 No Content
```

error responses:

```
422
{
   "title": "INSUFFICIENT_FUNDS",
   "detail": "some specific details provided by server"
}
```

```
422
{
   "title": "LIMITS_EXCEEDED",
   "detail": "some specific details provided by server"
}
```

```
422
{
   "title": "FRAUDS_DETECTED",
   "detail": "some specific details provided by server"
}
```

```
404
{
   "title": "BALANCE_NOT_FOUND",
   "detail": "some specific details provided by server"
}
```

Force debit transaction

This kind of transaction is used to inform server side that transaction has occurred.
For this request, actual transaction already happen so server can not reject this request. This behavior can occur for offline transactions fe: in plane, subway, for refunds and referring to previously authorized transactions.

```
POST https://server-domain.com/transactions/force-debit
```

headers:

```
Content-Type: application/json
X-Idempotency-Key: uuidV4
```

request body:

Description of the contents of the transaction object can be found above.

response:

```
204 No Content
```

error response:

As mention in description section, **we do not accept transaction rejection**.

Credit transaction

Method is used to credit user balance. In credit transactions Antaca asks 'can user get money?'.

```
POST https://server-domain.com/transactions/credit
```

headers:

```
Content-Type: application/json
X-Idempotency-Key: uuidV4
```

request body:

Description of the contents of the transaction object can be found above.

success response:

```
204 No Content
```

error responses:

```
404
{
    "title": "BALANCE_NOT_FOUND",
    "detail": "cannot find requested balance"
}
```

```
422
{
    "title": "FRAUDS_DETECTED",
    "detail": "some specific details provided by server"
}
```

Force credit

Method is used to credit user balance. This kind of transaction is used to inform server side that transaction has occurred. For this request, actual transaction already happen so server can not reject this request.

```
POST https://server-domain.com/transactions/force-credit
```

headers:

```
Content-Type: application/json
X-Idempotency-Key: uuidV4
```

request body:

Description of the contents of the transaction object can be found above.

success response:

```
204 No Content
```

error response:

As mention in description section, **we do not accept transaction rejection**.

Reversal

Method is used to revert any changes for previous transaction. Request body will be identical to transaction witch client try to revert. If server cannot find referenced transaction then no action is required.

```
POST https://server-domain.com/transactions/reversal
```

headers:

```
Content-Type: application/json
X-Idempotency-Key: uuidV4
```

request body:

Description of the contents of the transaction object can be found above.

success response:

```
204 No Content
```

error responses:

IMPORTANT: for this method, we do not accept any error. Only satisfying behavior is to revert referenced transaction and no action if cannot find transaction.

Update transaction status

From time to time, client will inform about clearings triggered by acquirer side. If client mark transaction as cleared it means that transaction will not be corrected by any other transaction request and requested amount is final.

```
PUT https://server-domain.com/transactions/:id
```

path parameters:

id - transaction identifier

request body:

Description of the contents of the transaction object can be found above.

success response:

```
204 No Content
```

error responses:

```
404
{
    "title": "TRANSACTION_NOT_FOUND",
    "detail": "cannot find requested transaction"
}
```

Transactions notifier

This document describes API for external transaction event notifications. Client who is interested in receiving notification about any transaction that occur in system, must implement below API.

These notifications support sending**Idempotency Key**

Security

Security for this endpoint is described in **The Security** section in the beginning of this page.

Api

For obtaining transaction event notification the Antaca is using single endpoint.

```
POST https://server-domain.com/notifications/transaction
```

Header

```
Content-Type: application/json
X-Idempotency-Key: 51ec546d-049a-4b8f-a05e-933938656eb2
```

Request body

```
{
  "status": "SUCCESS",
  "date": "2023-11-17T11:32:16+00:00",
  "description": "APPROVED",
  "transaction": {
    "id": "b4f534ef-77c2-4f16-ab4d-496806a76fb6",
    "balanceId": "60036f20-3b2c-470e-b9de-3c6cfbe8a5ff",
    "resourceId": "b3de5060-2ae2-4f3c-9b94-9c27a90dc6fe",
    "resource": "card",
    "cardId": "357970",
    "externalTransactionId": "d275ecb8-138e-4d0e-b5bf-c4158b4ce516",
    "referenceExternalTransactionId": null,
    "type": "POS",
    "category": "DEBIT",
    "amount": 10000,
    "currency": "PLN",
    "originalAmount": 12300,
    "originalCurrency": "PLN",
    "status": "AUTHORIZED",
    "description": "FrogShop Lublin POL",
    "date": "2023-11-17T11:32:16+00:00",
    "referenceExternalTransactionDate": null,
    "transactionData": {
      "mcc": "5122",
      "merchantIdentifier": "12345",
      "captureMode": "NFC",
      "lastFourDigits": "0911",
      "acquirerCountry": "PL"
    }
  }
}
```

Parameters:

Name	Required	Description	Allowed values
status	TRUE	Status of the transaction processing. **SUCCESS** - indicates that the transaction has been successfully processed in Antaca. **DECLINED** - indicates that the transaction has been declined. **INVALID -** indicates that the transaction can't be processed.	SUCCESS, DECLINED, INVALID
date	TRUE	Date time of request generation in ISO 8601 date	ISO 8601 date, eg. 2023-11-16T13:41:40+00:00
description	TRUE	Describes more details for returned status	describes in **description section details**
transaction	TRUE	The transaction properties	transaction object described in **the transaction object** section

Description details:

Value	Description
APPROVED	Indicates a successful transaction. Antaca processed the transaction with no errors.
EXCEEDS_AMOUNT_LIMIT	Occurs when the transaction amount exceeds card limits.

INSUFFICIENT_FUNDS	There is not enough money on balance.
CARD_NOT_FOUND	Antaca cannot find card for which the transaction was invoked.
BALANCE_NOT_FOUND	Antaca cannot find balance for which the transaction was invoked.
INVALID_AMOUNT	Amount of the transaction was passed as <= 0.
INSUFFICIENT_FUNDS_ON_DEPOSIT_BALANCE	Deposit balance has not enough funds to process the transaction.
DEPOSIT_BALANCE_NOT_FOUND	Occurs when client has not configured deposit balance in currency used for rejected the transaction.
AML_EXCEPTION	AML regulations does not allow to process the transaction.
AMBIGUOUS_REFERENCED_TRANSACTION	Antaca cannot determine for which a transaction refer current the transaction request. System has found more then one transaction matched by transaction parameters.
REFERENCED_TRANSACTION_NOT_FOUND	System cannot find any transaction for which the request refer.
CURRENCY_MISMATCH	The transaction currency is different than balance currency.
CUSTOMER_NOT_FOUND	System cannot find customer who is involved in the transaction.
BUDGET_EXCEEDED	The budget limitation for an card or an customer has been exceeded.
LIMIT_EXCEED	General limitation for the transaction has been exceeded.
COLLATERAL_BALANCE_NOT_FOUND	System cannot find an collateral balance in proper currency configured for client instance. Those balances could be eg. deposit, credit, technical etc.
INSUFFICIENT_FUNDS_ON_COLLATERAL_BALANCE	An collateral balance has not enough funds to process transaction request.
UNKNOWN_TRANSACTION_TYPE	System cannot determine kind of the transaction and reject it for security reason.
UNKNOWN_ERROR	General error. System cannot match any of concrete description.

Transaction object:

Name	Required	Description	Allowed values
id	TRUE	Unique identifier of the transaction in UUID format.	any value in UUID v4 format, eg. ddb55ff9-11ca-4621-9129-81f939e66011
balanceId	TRUE	The balance identifier in UUID format. This could refere to a customer or any of collateral balance.	any value in uuid v4 format, eg. 6bb3745f-1ddf-4579-855f-913c3f272d19
resourceId	TRUE	Identifier of resource used to process transaction. This is always in uuid format.	any value in uuid v4 format, eg. 846edf0f-9a96-4f1d-bc38-9c963605b9e8
resource	TRUE	Name of resource used to process transaction. This could be eg. card, balance, creditBalance, depositBalance etc. This list could change in future so please do not hardcode this value.	card, balance, creditBalance, debitBalance
cardId	TRUE	The card identifier in string format. This value could be used to communicate with the Antaca services.	any string value. Mostly it should be eg. "1234" but it can change in the future and become UUID format.
externalTransactionId	TRUE	This is transaction identifier obtained from the transaction processor. This value is not unique and can be duplicated over time. The Antaca is not responsible for this value.	any string value
referenceExternalTransactionId	FALSE	This is similar like externalTransactionId except it refers to previously obtained a transaction. This value is not unique and can be duplicated over time. Antaca is not responsible for this value.	any string value

type	TRUE	Type of transaction.	This list could evolve over time so please check this documentation from time to time. POS, ATM, Cashback, AFT, Balance Inquiry, Payment, commission, fee, funding, interest, withdrawal, collateralDebit, companyDebit, ibanTechnicalDebit, cashback, creditIbanTransfer, loan, payment, topUp, collateralCredit, companyCredit, ibanTechnicalCredit
category	TRUE	Category of the transaction used for identification of funds movement.	CREDIT, DEBIT
amount	TRUE	Amount of the transaction. This is always integer in minor value.	any integer value greater than 0.
currency	FALSE	Currency of transaction in ISO 4217 3-letter code.	any ISO 4217 3 letter code eg. PLN, USD, EUR
originalAmount	FALSE	Amount of the original transaction in integer minor value.	any integer value greater than 0. Also this field could has null value
originalCurrency	FALSE	Currency of the original transaction in ISO 4217 3-letter code.	any ISO 4217 3 letter code eg. PLN, USD, EUR. Also this field could has null value
status	TRUE	Current status of the transaction (after Antaca service process).	AUTHORIZED, CLEARED,
description	TRUE	Detailed description of the transaction.	any string value
date	TRUE	Date of the transaction in ISO 8601 date.	any data specified by iso 8601. Eg. 2023-11-17T11:32:18+00:00
referenceExternalTransactionDate	FALSE	Date of the transaction for which this transaction is refer to. Date in ISO 8601 date.	any data specified by ISO 8601. Eg. 2023-11-17T11:32:18+00:00

transactionData	TRUE	The transaction data object described in **the transaction data section.**	Keep in mind that this object is always passed but it can be empty.

Transaction data object

Name	**Required**	**Description**	**Allowed values**
mcc	FALSE	Merchant category code.	any mcc value, eg. can be found here: https://global.alipay.com/docs/ac/files/mcclist
merchantIdentifier	FALSE	The merchant identifier for the transaction.	
merchantName	FALSE	Name of the merchant.	
captureMode	FALSE	Capture mode.	magstripe, manual, emv, on behalf (EMV), nfc, ecommerce, adj
lastFourDigits	FALSE	last 4 digits of a card.	
acquirerCountry	FALSE	Country of acquirer.	
mdesDigitizedWalletId	FALSE	The Wallet ID (Wallet Reference) used to digitize the card.	m4m, google pay, samsung pay, apple pay
cashbackPosCurrencyCode	FALSE	Represents the currency code of the cashback amount.	ISO 4217 3-letter code
cashbackPosAmount	FALSE	Displays the actual cashback amount.	integer value in gross

Response

Only responses with http code 200 & 204 are allowed.

```
200 OK


204 NoContent
```

In case of any other response code, Antaca will try to send a request once again (up to 5 times). Every time a request will be identical with the same **X-idempotency-key.** Keep in mind that if your service has answered properly, network errors can arise either way. If Antaca resends the request with the same X-Idempotency-Key, the response should be retrieved from the cache.

Revision #61
Created 19 September 2023 07:00:51 by Tadeusz Krysa
Updated 18 June 2024 07:43:35 by Barbara Tudruj