Paymont API PSD2

Third party manual and documentation (TPP)

Paymont UAB

Author Paymont UAB
Version 1.0

Obsah

1. Introduction

Solution of API PSD2 in PAYMONT UAB is based on the Open Banking Standard (OBS) version 2 and following versions (referred to as **OBS/ OBS v2**) issued by the Czech Banking Association (CBA). The specifications below describe the individual interfaces, with possible deviations from the standard. If some attributes or interfaces are not described, then they are not supported by the bank.

1.1 Technical details

1.1.1 Architecture

The Representative State Transfer (**REST**) transport protocol is used for API communication.
JSON (JavaScript Object Notation) is used for the format of writing query and response data via the API.
The OAuth 2.0 authentication protocol is used to authorize requests.

1.1.2 API

The third party will send its requests to the issued endpoints.

Making minor changes to a backward-compatible API will not upgrade the API version. We reserve the right to make such changes without notice, and therefore a third party application should be prepared for these situations. These are in particular:

- Extending the response with new attributes without changing the structure of current attributes
- Editing error messages and exceptions, including their codes
- Extension of optional parameters (header, URL parameter, body) of the request

1.1.3 Sorting

All APIs return records by default in the order in which they are presented to users via other electronic channels of the bank (InternetBanking). If you use optional parameters specifying OBS sorting when calling API services, they will be ignored by the bank interface.

1.1.4 Pagination

For specific APIs that return collections (Transaction Summary), it is possible to request a paged list. The query parameters page and size are used for this query. Every resource that allows you to request a paged list has this property explicitly specified.

Query parameters per page

page - The required page number. Pages are numbered from 0. If the parameter is not specified, the API returns the first (zero) page.
size - The required number of records per page. If the parameter is not specified, the API returns the entire collection.

Paged response parameters

pageNumber - The number of the current page. The first page has the number 0.

pageCount - The total number of pages.

pageSize - The number of records per page. This parameter can match the required size value from the query, except when it is the last page, or when the required page range has exceeded the maximum limit defined for a particular resource API.

Filtering

API does not support filtering except for "Transaction Summary" report

Filtering rules

fromDate – date from.
It can be a maximum of 2 years in history. If not filled in, the current date is used.
Type: date (optional, format YYYY-MM-DD), example: 2018-04-28
toDate - date to.
If not filled in, the current date will be used.
Type: date (optional, format YYYY-MM-DD), example: 2018-05-16
fromDate cannot be larger than toDate.

If filtering parameters not listed here are used in the request, they will be ignored by the bank interface.

1.2 Terminology

ASPSP

Account Servicing Payment Service Provider – a payment service provider, in this case a bank.

OBS

Abbreviation for Open Banking Standard.

TPP

Third Party Provider – a third party, entity, payment service provider.

A Third Party Provider (TPP) can be an institution that has been granted a licence by the CNB or Another official authority within the EU countries. The bank through whose API the TPP wants to send PSD2 requests can check this licence in the list available on the CNB website.

TPPs are divided into the following Types, whereby a TPP may be licensed to operate a combination of the services listed below:

- AISP (Account Information Service Provider)
- PISP (Payment Initiation Service Provider)
- CISP (Card issuing Service Provider)

Consent

Client's consent to the provision of services through intermediaries - TPP (services such as AIS, PIS, CIS).

By granting consent, the client allows the TPP application access to its accounts (the consent specifies which accounts it grants access to for TPP and which services (AIS, PIS, CIS) it allows).

AISP

Account Information Service – payment account information service provider - based on the client's consent, the TPP provides information about the payment account and transactions that are executed on the client's account with the bank. For example, if the client has accounts with multiple banks, through a third party, he/she can see the transaction history or balances on all these accounts simultaneously in one place (via the TPP application or portal).

PISP

Payment Initation Service - A third party may:

- initiate payments on behalf of the customer
- send the initiated payment for processing (if the client has previously authorised the payment)
- inquire about the status of the processed payment
- enquire about the sufficiency of payment funds in the account from which the payment is to be made

CISP
Card-based Payment Instrument Issuer - a payment service provider issuing a payment instrument (payment card). There are TPPs that can issue a payment instrument that will be linked to a payment account at a bank. The TPP will then be able to verify whether the customer has sufficient funds in the bank account to which the TPP has issued the card to complete the card transaction. The bank will respond to the TPP's query with a YES / NO answer.

API Gateway
Provides TPP access to bank services.

Payment services
Execution of a payment (transaction) on the basis of an authorized request.

Authentication server (eCobra)
Authentication server, operated by Paymont.

IB
Abbrevation for electronic banking, operated by Paymont.

PSD2
Payment Services Directive 2 – EU banking directive from 2015, which deals with payment services in general. The directive was created, to unify the provision of payment services in the EU**.**

EV
Extended Validation certificate.

2. Safety

Interface request authorization is based on the token-secured OAuth2 authorization concept. The client provides a token (access_token) each time the API requests, as proof that it can access the requested data. The bank's interface verifies the used token in relation to the used interface, and only after the successful verification of the token and the rights resulting from it, the required operation is executed. The token must be specified in the request header, eg:

Authorization: Bearer aT6oKuCt6i0plw26nxI7r32Lpi89bt

Parameters:

Parameter	Value
Validity period of Refresh token	90 days
Validity period of Access token	3600 seconds (1 hour)
Validity period of the generated Code, generated by Request / Authorization completion	10 minutes
Client_secret validity period	Unlimited
The validity period of the signID authorization request	5 minutes (Redirection of the user to the Federated authorization of the bank is possible only for the period of validity of the signID . After expiration, it is necessary to request the generation of a new signId using the method - Generation of authorization ID)

2.1 Basic security model

The basic security model for API access is based on a combination of the security elements listed below (in order for the TPP to send requests via the API, all of the following security elements must be met).

- Encrypted communication between TPP and the bank (use of a valid certificate on the part of TPP and the bank)
- Registered verified valid TPP record in Internet Banking
- **Registered TPP application** in Internet Banking (with unique client_id and client_secret)
- Existence of a valid consent to the defined access **of the TPP application** to the accounts of the dispositor
- **Valid access token (linked to a specific consent created by the dispositor) specified in the header of the sent request via API**

2.2 TPP-Bank encrypted communication

Communication between the client system and the bank assumes security using the SSL protocol with at least 128-bit encryption. On the bank and TPP side, a qualified certificate for web server authentication **according to eIDAS** must be used to create a secure channel . The certificate used must be issued in accordance with ETSI TS 119 495 (Qualified certificates and TSP policy requirements according to the Payment Services Directive (EU) 2015/2366). TLS 1.2+ is required to secure the communication layer.

2.3 Registration of TPP in E-Banking

A record of each TPP wishing to send requests via the PSD2 API issued by the bank must exist in the IB database.

The creation of a new TPP record and the updating of TPP records already existing in the e-Banking database (verification and registration of TPPs) is performed by the E-Banking administrator via the GUI of the IB intranet part.

The e-Banking administrator will only enter TPPs into the E-Banking database that contact the bank - upon receipt of the certificate, the IB administrator will verify the TPP licence number and establish the TPP database in IB. When registering a verified TPP record, the administrator will add the full licence number including the prefix specified in the TPP certificate to the TPP record.

If a valid TPP record exists in the IB database, the TPP must perform a registration flow in the bank via a specific endpoint of the exposed PSD2 API.
During the registration flow, the TPP registers its **application/multibank portal with the bank** (a TPP may operate multiple applications - note: if a TPP offers multiple applications to its clients, it must register each of its PSD2 applications with the bank). The TPP receives technical identifiers (client_id, client_secret) for each registered application from the bank.

2.4 Consent to TPP access to the accounts of the disponent

Another condition that must be met in order for TPP to send requests via the API is the existing valid consent of the **TPP application** to access the user's accounts. The consent is created by an access request that the dispositor creates on the bank side and authorises with his authentication device (the IB administrator does not interfere in this process).

The stored consent includes the following items:

- Selected TPP application
- List of service permissions (AISP, PISP, CISP) enabled by the dispatcher for the TPP application.
- List of accounts to which the user has granted access (the request offers only current accounts to which the user has set up active access in the bank and if the user has PSD2 service enabled in the bank in relation to the client who is the account holder)
- The "Validity TO" date of the consent issued (the validity of the consent is implicitly unlimited)

2.5 Solution description

2.5.1 Main API features

- ***API supports:***
  - all mandated services required under the OBS
  - non-mandatory services used for automated application registration TPP
- ***API:***
  - TPP will use the banking API, which is designed as a web service (WS)
  - The transport protocol is used for the API communication interface **REST** (Representational State Transfer).
  - The format for writing query and response data through the API is **JSON** (JavaScript Object Notation).
- ***Registration of TPP applications****:* each TPP record can be linked to 1...n TPP applications; the application registers the TPP in the bank during the registration flow.
- ***API:*** E-Banking performs TPP verification on every request received via API. A request sent from the TPP via API to the bank will receive the requested response only if all of the following conditions are met:
  - on the basis of the certificate used by the TPP during communication, the TPP record is traced in the TPP table (the TPP is traced on the basis of the TPP licence number specified in the certificate (licence number including prefix) - the identical licence number must also be specified in the TPP record (in the IdentifierInCertificate item) in the IB database)
  - traceable TPP record is valid
  - the Type of method used corresponds to the service (AISP, PISP, CISP) that is enabled in the tracked TPP record in the TPP table
  - access code used in the request is valid
  - based on the access code used (OAUTH protocol) specified in the request, a valid consent is traced in the Disponent-Application TPP bindings
  - the account specified in the request is contained in the consent that was traced from the access code (specified in the request header)
  - the TPP application has a service (AISP, PISP, CISP) enabled from the user in the tracked consent that matches the method used in the received request

2.5.2 Description of basic procedures

2.5.2.1 TPP registration at the bank

In order for a TPP to communicate via the bank's PSD2 API, it must obtain the technical security elements (client_id, client_secret) required for its application from the bank to subsequently obtain the token used in OAuth 2.0. The TPP can only obtain these elements after registering its application with the bank.

1. the TPP registers its application with the bank via an API issued by the bank using specific methods - see chapter (2.8.1).

2. At the moment of receiving the request for registration of the TPP application via the PSD2 API issued by the bank, the TPP will be verified on the bank's side in the e-banking system. Verification is performed on the basis of the license ID issued by the national regulator and the certificate of the entity concerned. In order for the request to be executed, the following must be met:
   - The license ID specified in the certificate used by the TPP when communicating via the PSD2 API issued by the bank must be traced in the TPP record (in the IdentifierInCertificate item in the IB database).
   - A TPP record that has been traced to a license ID must be valid.

3. In case the license ID contained in the certificate used for TPP communication via API is not contained in any TPP record in the database, the procedure may be as follows:
   - The TPP contacts a bank employee who performs a manual verification (the TPP gives the bank its certificate (without the secret part) with the necessary documents, on the basis of which the bank

employee verifies the TPP. Verification of the TPP record will be based on **the TPP entity's name, the license ID issued by the national regulator and the entity's certificate**. After manual verification, the bank employee will create a new TPP record via the IB Intranet interface and add the licence ID (contained in the TPP certificate) to the TPP record in the IB database.

- After the bank employee creates a new TPP record in the IB database, the TPP makes Another attempt to register its application via the API.

4. When registering the TPP application for communication via the bank's API, the following technical security elements required for authentication flow using OAuth 2.0 are generated in electronic banking:
    - The identifier (client_id) that the TPP application will use when communicating via the API
    - secret code (client_secret), (TPP will never use secret_code alone, it must always be used in combination with client_id - the combination of client_id and client_secret is included in the request when replacing a one-time authorization code with a refresh and access token (Get token resource).

5. The generated technical security elements are passed to the TPP (the TPP receives these technical security elements when registering via the API in response to the registration request)

6. From the moment the technical security features are generated, the name of the registered TPP application will be offered to the bank's customers when creating the consents for TPP access to accounts.

2.5.2.2 Registration of the bank's client in the TPP application - setting up TPP access to the dispositor's accounts

In order for the TPP to send queries to the dispositor's accounts or to create a payment on behalf of the dispositor and subsequently authorise it, the dispositor must consent to this. The following are the steps that the user must take.

1. The Dispatcher can create consent for third party access to their current accounts via the Central Authorization page to which they are redirected when activating PSD2 access via the third party application.

2. If the third-party application is already registered with the bank, the bank's client's dispositor can log in to IB in the standard way and create a request in the PSD2 section to create a consent for the specific TPP application to access their accounts:
    - enable the required AIS / PIS / CIS services for the TPP application
    - enable access to their accounts (only current accounts to which the dispatcher has **active** access in the dispatcher-client relationship and at the same time has PSD2 service enabled in the service package for the client).
    - The Disponder authorises the request for consent with its authorisation device.

3. The bank's client installs the TPP application or accesses the TPP portal.

4. The bank client (application user) selects his bank in the application / portal and starts the registration workflow.

5. After requesting registration in the TPP application, the client is redirected to the CITFIN **authentication frontend (central authentication page)** using the OAuth 2.0 protocol with a request to authorize access to services.

6. The client is authenticated on the central page in a standard way (authenticated with his identifier, password and code generated on his authentication device).

7. After authenticating the client, IB checks whether a valid consent exists for the application and the currently logged-in dispatcher in the **Disponent - TPP application link** (consent created on the basis of a request from IB or from the Central Authorization Page).

- **Option - valid consent not found:** a page is displayed in which the client's dispatcher agrees to grant the TPP application access to the services (AISP, PISP, CISP) and to the current accounts to which he/she has set active access (if he/she is a dispatcher for more than one client, when creating the consent, he/she will be offered the accounts of all clients to which he/she **has set active access via a specific client and at the same time has PSD2 service enabled for that client in the service package**).
- **Option - valid consent is traced:** in the OAuth response, the TPP application receives a one-time authorization code, which it then sends to the TPP server. The TPP server then contacts the /**token endpoint** issued on the bank's frontend to exchange this one-time authorization code for a **pair of access and refresh tokens**.

8. The TPP application then uses the Access token to communicate with the PSD2 API issued by the bank. The API Gateway will request the validation of the token and obtain the appropriate scope (AIS/PIS/CIS) and the corresponding user identity to which the token belongs.

2.5.2.3 Client payment authorization

1. When a payment is initiated via the TPP application/portal, the TPP receives in response via the PSD2 API number under which the payment was stored on the bank side (orderId) and the Token Identifier generated for the specific transaction authorization process (signId). The validity of the signId is limited (5 minutes). This payment must then be authorized by the bank's customer directly on the bank's side.
2. The application starts the initialization of the payment authorization (POST /api/payments/{paymentId}/sign/{signId}). Starting the authorization process is only allowed if the signId is valid. After calling this method containing the CODE Type corresponding to the federated authorization (USERAGENT-REDIRECT), the response is a URL and parameters for redirection to the federated authorization page (central authentication page).
3. The third party uses these parameters to redirect the client to the bank's central authentication page. This redirection includes the payment number (orderId) to be authorized by the client.
4. The logic of the central page verifies whether the application from which the bank's referrer was redirected is registered in IB (based on the client_id specified in the URL). If the client_id used in the request is not traceable in the IB database in any TPP application, the request is rejected (an error is returned in the response).
5. If the TPP application is registered and the redirect_uri used in the request is valid (the URI must be specified in the record of the registered TPP application in IB), the client goes through the authentication process provided by the bank in the principle of SCA authentication (as it is known to the client from the IB environment) after redirection.
6. After authentication of the user, the user is shown the detail of the payment based on the orderID. By authorizing the payment, the user agrees to the transaction.

2.6 Services supported in API PSD2

The PSD2 solution allows third parties to use PSD2 services via exposed APIs - see the following tables:

2.6.1 AIS area services

AIS	Description
Account balance (JSON)	through this service, a third party authorized by the client receives an overview of the balances of the client's bank account held with the bank in question
Transaction overview (JSON)	through this service, the client's authorized third party receives an overview of transactions
List of client payment accounts (JSON)	upon request, the service returns a list of accounts for which the client consents to a specific TPP (not a list of all client accounts) without balances

2.6.2 PIS area services

PIS	Description
Enquiry about sufficient resources (JSON)	Verification of sufficient balance in the payer's account
New Payment (Payment Initiation) (JSON)	through this service, a third party authorized by the client initiates (creates) one non-eCommerce payment order from the client's bank account in JSON format The TPP then applies the payment authorization initiation method to the given order - the client is redirected to the central authentication page of the bank and here he authorizes the given order with his authentication device (TOKEN).
Status of established / initiated payment (JSON)	checking the status of a payment order
Delete an unauthorized payment (JSON)	A service that allows you to cancel a payment that has not yet been authorized and that has been created via PISP New Payment (payment initiation)
Payment authorization (JSON)	Through this service, a third party can run a payment authorization workflow initiated through the third party application
Generating an authorization ID (JSON)	Through this method, a third party can request the generation of a new SignId. Used in cases where: › The original authorization request (e.g. generated as output from a Payment Initiation method call) has expired. Authorization requests are limited to 5 minutes from the time they are created. › Authorization of the request by the user was not done at the will of the user - the instruction to authorize was refused - and he is interested in a repeat if the authorization. › For technical reasons on the TPP application side *Note: by establishing a new authorization, the original and unused authorization requirements remain in effect.*

2.6.3 CIS area services (Verification of sufficient resources)

Service defined by PSD2 as resource adequacy information provided to CISP providers.

CIS	Description
Enquiry about sufficient funds (JSON)	Verification of sufficient balance in the payer's account

2.7 Additional services - automatic TPP registration via banking API

The solution includes the implementation of the following methods, which are optional within OBS. These TPP methods enable automatic application registration / application registration changes via the banking API.

Assignment of TPP technical safety features (Enroll)	Description
TPP application registration (JSON)	through this service, a TPP with a valid certificate and license number automatically registers its application with the bank and in response receives technical security elements (client_id and client_secret) for the registered application
Information about application registration data (JSON)	Through this method, the TPP can request an overview of the registration data for a specific application
Change application registration (JSON)	through this service the TPP will be able to change the registration data
Deleting an application registration (JSON)	through this service, the TPP will be able to deregister the application
Request to generate a new client_secret	through this service the TPP will be able to request the generation of a new client_secret

2.8 Description of methods used for service providers (TPP)

2.8.1 Registration resource (Enrollment)

The following sections describe the methods by which a TPP applies to register its application with the bank, or can make changes or unregister its application.

2.8.1.1 Automatic generation of technical identifiers

To call a resource you need:

- **use valid certificate**.

The output is the **client_id** and **client_secret** parameters, which TPP needs to obtain the **access_token** and **refresh_token** token pairs.

Endpoint: POST https://api.paymont.eu/api/**oauth2/register**

Request			
Attribute	**Mandatory**	**Type / allowed values**	**Description**
application_Type	Yes	string	The Type of application that will use the client_id. (only Web, native Types are allowed).
redirect_uris	Yes	Array of strings e.g. URL [Max 3x 2047 B]	A list of URIs where the authentication flow is redirected at the end. The authorization request must contain just one of these registered in the exact format.
client_name	Yes	String [Max 255 B]	Name of the TPP application
client_name#en-US	No	String [Max 1024 B]	The name of the TPP application in the appropriate language/encoding.
logo_uri	No	URI [Max 2047 B]	URI of the application logo (or the place from where it can be downloaded during registration)
contact	No	string [Max 320 B]	Email as a contact to the responsible person on the client side of the application.
scopes	No	Array of strings [Max 10x 255 B]	An array of applications of the required scopes. When registering, the scopes are validated against the contents of the certificate used and against the scopes listed in the TPP record, which must already exist in the IB database at that time.

Response			
Attribute	**Mandatory**	**Type**	**Description**
client_id	Yes	String	The **client_id** assigned to the application. This ID is used when the authentication process is started and during the communication process (replacing a one-time code with a pair of access_token and refresh_token tokens and refreshing a token).
client_secret	Yes	String	**Client_secret -** password / token issued by the bank (ASPSP) for the TPP application **(client_id)**
client_secret_expires_at	No	DateTime	The default value is 0 (client_secret never expires). Otherwise, the value is in seconds from 1970-01-01T0:0:0Z
api_key	No	String	The API key that the application uses when communicating with the bank's API. **The API key is not supported by the bank in this solution** (the response states "NOT_PROVIDED")
application_Type	Yes	String	The Type of application that will use the client_id. (only Web, native Types are allowed).
redirect_uris	Yes	Array of strings e.g. URL [Max 3x 2047 B]	A list of URLs where the authentication flow is redirected at the end.
client_name	Yes	String [Max 255 B]	Name of the TPP application
client_name#en-US	No	String [Max 1024 B]	The name of the TPP application in the appropriate language/encoding.
logo_uri	No	URI [Max 2047 B]	Application logo URI
contact	No	string [Max 320 B]	Email as a contact to the responsible person on the client side of the application.
scopes	No	Array of strings [Max 10x 255 B]	Array of required scopes.

Error codes		
HTTP Status	Error code	Description
400	**invalid_request**	The request is missing a required field or is in an inappropriate / non-valid format.
400	**invalid_scope**	Invalid scope in the request.
400	**invalid_redirect_uri**	The value of one or more redirect uri is not valid.
401	**unauthorized_client**	TPP is not authorized to make this inquiry.
401	**access_denied**	The authorization server denied access.
403	**insufficient_scope**	E.g. insufficient authorization to use the required scope.
500, 503	**server_error**	Authorization server error.

For example use, see source [6] chapter 1.4.1.1.

2.8.1.2 Information about the application registration data

By calling this resource, the TPP can request an overview of the registration data for a specific application.
To call the resource, you need:

- **Use a valid certificate**
- **Use the client_id, which is issued to this TPP**.

The output is an overview of registration data.

Endpoint: GET https://api.paymont.eu/api**/oauth2/register/{client_id}**

Response			
Attribute	**Mandatory**	**Type**	**Description**
client_id	Yes	String	Unique **client_id** identifier assigned to the TPP application by the bank.
client_secret	Yes	String	**Client_secret -** password / token issued by the bank (ASPSP) for the TPP application **(client_id)**
client_secret_expires_at	No	DateTime	The default value is 0 (client_id never expires). Otherwise, the value is in seconds from 1970-01-01T0:0:0Z
api_key	No	String	The API key that the application uses when communicating with the bank's API. **The API key is not supported by the bank in this solution** (the response states "NOT_PROVIDED")
application_Type	Yes	String	Type of application that uses client_id (web, native values are allowed)
redirect_uris	Yes	Array of strings e.g. URL	A list of URLs where the authentication flow is redirected at the end. The authorization request must contain just one of these registered URIs in the exact format.
client_name	Yes	String	Name of TPP application
client_name#en-US	No	String	The name of the TPP application in the appropriate language/encoding.
logo_uri	No	URI	Application logo URI
contact	No	string	List of E-mail addresses, contacts to the responsible person on the TPP side of the application.
scopes	No	Array of strings [Max 10x 255 B]	Array of required scopes.

Error codes		
HTTP Status	**Error code**	**Description**
400	**invalid_request**	A non-valid request. The request is missing a required field or is in an inappropriate / non-valid format.
401	**invalid_client**	Invalid client_id.
401	**unauthorized_client**	TPP is not authorized to make this inquiry.
401	**access_denied**	The authorization server denied access.
403	**insufficient_scope**	E.g. insufficient authorization to use the required scope.
500, 503	**server_error**	Authorization server error.

Example of use, see source [6] 1.4.1.2.

2.8.1.3 Change of registration data

By calling this resource, the TPP can request to change the registration details for a specific application.
To call the resource, you need:

- **use a valid certificate**
- **use the client_id that is issued for this TPP**.

The output is a summary of the changed data.

Endpoint: PUT https://api.paymont.eu/api/**oauth2/register/{client_id}**

Request			
Attribute	**Mandatory**	**Type**	**Description**
application_Type	Yes	string	The Type of application that will use client_id (web, native values are allowed)
redirect_uris	Yes	Array of strings e.g. URL [Max 3x 2047 B]	A list of URLs where the authentication flow is redirected at the end. The authorization request must contain just one of these registered URIs in the exact format.
client_name	Yes	String [Max 255 B]	Name of TPP application
client_name#en-US	No	String [Max 1024 B]	Name of the TPP application in the appropriate language/encoding.
client_Type	Yes	String	OAuth defines two Types of clients (Confidential / Public). ASPSP (bank) supports only the **Confidental Type.**
logo_uri	No	URI [Max 2047 B]	URI of the application logo (or the place from where it can be downloaded during registration)
contact	No	string [Max 320 B]	Email as a contact to the responsible person on the client side of the application.
scopes	No	Array of strings [Max 10x 255 B]	An array of applications of the required scopes. When registering, the scopes are validated against the content of the certificate used and against the scopes listed in the TPP record, which must already exist in the IB database at that time (the TPP record is created when updating data from the NBS).

Response			
Attribute	**Mandatory**	**Type**	**Description**
client_id	Yes	String	Unique client_id identifier assigned to the TPP application by the bank.
application_Type	Yes	String	The Type of application that will use client_id
redirect_uris	Yes	Array of strings e.g. URL	List of URLs where the authentication flow is redirected at the end.
client_name	Yes	String	Name of TPP application
client_name#en-US	No	String	The name of the TPP application in the appropriate language/encoding.
logo_uri	No	URI	Application logo URI
contact	No	string	List of E-mail addresses, contacts to the responsible person on the TPP side of the application.
scopes	No	Array of strings	Array of required scopes.

Error codes		
HTTP Status	**Error code**	**Description**
400	**invalid_request**	A non-valid request. Required field is missing or in an inappropriate / non-valid format.
400	**invalid_scope**	Invalid scope in the request.
400	**invalid_redirect_uri**	The value of one or more redirect uri is not valid.
401	**invalid_client**	Invalid client_id.
401	**unauthorized_client**	TPP is not authorized to make this inquiry.
401	**access_denied**	The authorization server denied access.
403	**insufficient_scope**	E.g. insufficient authorization to use the required scope.
500, 503	**server_error**	Authorization server error.

Example of use, see source [6] 1.4.1.3.

2.8.1.4 Deleting an application

By calling this resource, the TPP can request to delete data and access a specific application. To call the resource, you need:

- **use a valid certificate**
- **use a valid client_id that is issued to this TPP**.

The output is a confirmation of deletion.

Endpoint: DELETE https://api.paymont.eu/api/**oauth2/register/{client_id}**

If the application is deleted, an HTTP 204 response is returned as a successful delete response záznamu aplikace s konkrétním client_id).

Error codes		
HTTP Status	Error code	Description
400	**invalid_request**	A non-valid request. The request is missing a required field or is in an inappropriate / non-valid format.
401	**invalid_client**	Invalid client_id.
401	**unauthorized_client**	TPP is not authorized to make this inquiry.
401	**access_denied**	The authorization server denied access.
500, 503	**server_error**	Authorization server error.

Example of use, see source [6] chapter 1.4.1.4.

2.8.1.5 Request for a new client_secret

By calling this resource, the TPP can request the issuance of a new client_secret. To call the resource, you need to use:

- **valid certificate**
- **valid client_id issued to this TPP**.

The original client_secret will be cancelled by this request.

Endpoint: POST https://api.paymont.eu/api/oauth2/register/{client_id}/renewSecret

Response			
Attribute	Mandatory	Type	Description
client_id	Yes	String	The client_id assigned by the application.
client_secret	Yes	String	**New Client_secret** - password / token issued by the bank (ASPSP) for the TPP application **(client_id)**
client_secret_expires_at	No	DateTime	The default value is 0 (client_secret never expires). Otherwise, the value is in seconds from 1970-01-01T0:0:0Z

Error codes		
HTTP Status	Error code	Description
400	**invalid_request**	A non-valid request. The request is missing a required field or is in an inappropriate / non-valid format.
401	**invalid_client**	Invalid client_id.
401	**unauthorized_client**	TPP is not authorized to make this inquiry.
401	**access_denied**	The authorization server denied access.
500, 503	**server_error**	Authorization server error.

Example of use, see source [6] chapter 1.4.1.5.

2.8.2 Request Authentication and Authorization (OAuth2)

Request authorization is based on the OAuth2 authorization flow concept secured by a token - the application only checks the validity of the token used in the request header, which the TPP provides for each call as proof that it can access the requested data.
In the context of these APIs, the authorization token is considered a short-lived and stateless element that must be used in every API call that requests request authorization.

The basis of the solution is to use the OAuth2 open protocol for issuing authorization tokens - only the Authorization code grant framework is supported. The **Client Credentials Grant flow** variant is not supported in the implemented solution.

2.8.2.1 OAuth2 Authorization Code Grant

Within the OAuth2 protocol, the Authorization code grant framework is a way to issue both a refresh token and an access token to a partner application as a result of user identification and authentication.
The short-term access token is used by the partner application to communicate with the bank's API, and when it expires, the partner application can use the refresh token to request a new access token.

2.8.2.1.1 Basic properties

- access token is issued as a short-term (3600 s)
- the access token is **issued for a specific application and a specific user (it is bound to the consent created by the user - dispenser)**, it cannot be successfully used for Another application
- refresh token cannot be used directly for communication with API, it has long validity (90 days in case of PSD2)
- the bank and the application (TPP) share a common "secret" (client secret)
- the result of user identification and authentication is a one-time code, which is exchanged by a third-party application using a client secret for a refresh token and an access token
- one-time code alone cannot be used without knowing the client secret

2.8.2.1.2 Description of Code grant flow

Condition of use flow:

- the TPP application has its own unique **client_id** assigned by the bank and knows the client secret for that **client_id**
- when **client_id** and **client_secret are issued**, the **bank** gets information about the **redirect uri** - i.e. the URL where to redirect the user after successful authentication

Steps in the code grant flow:

2. The TPP application calls the bank's /auth resource and then the user (bank client) is redirected to the central authentication page to perform identification and authentication of the user (bank client)
3. Customer identification and authentication is in progress - these steps are fully under the bank's control
4. After successful authentication, the bank generates a code and redirects the user to the URI that was part of the /auth request (redirect_uri)
5. TPP uses resource /token to get the refresh_token and access_token. When calling this resource, the TPP passes the client_id and client_secret pair and the code value it received in the response of the previous /auth request to the bank in the request.
6. The TPP application uses the access_token obtained in the request header when communicating to the bank's API, when necessary.
7. The bank internally verifies the access_token. During this verification, it obtains the identity of the user on the basis of whose authentication the access token was issued.

2.8.2.1.3 Authentication resource issued by the bank

If there is no valid token pair (access_token and refresh_token), the TPP must create an Authorization Request, on the basis of which the bank's client is redirected from the application to the bank's central authentication page, where the request is subsequently authorized (see 2.5.2.2). The request is of the Oauth 2.0 **Authorization Code Grant Type**.

Endpoint: GET https://api.paymont.eu/oauth2/auth

Request			
Attribute	**Mandatory**	**Type**	**Description**
response_Type	Yes	**code**	The value of this parameter determines what Type of authentication flow is required. In this case, if it is a **code grant**. For the authentication process, this means that the result of this request will be a one-time **auth_code**, **which the TPP will then exchange for a pair of access_token and refresh_token tokens using Another request (token method)**
client_id	Yes	String	Unique identifier generated by the bank for the TPP application
redirect_uri	Yes	URL	URL where the authentication flow is redirected at the end. This URL is already set when the client_id is issued, and as part of the authentication process, this parameter is validated against the URL introduced to the client_id in the application record registered with the bank. The value must match one of the values specified in the registered application record.
Scope	Yes	String	This is an array of applications that require scope (permissions). In the case of PSD2, these can be AISP, PISP, CISP roles. For example, if a TPP holds multiple permissions, he can request only one or more of them here for his application. If multiple scope Types are used, they are separated by a space.
state	Yes	Any string [min 128 bits]	This parameter increases the security of communication during redirection. It protects against CSRF attacks and forwards information from the application through the authentication flow.

Response			
Attribute	**Mandatory**	**Type**	**Description**
Code	Yes	String	One-time Authorization Code
State	Yes	String	Attribute value passed from the TPP request

Error codes

› Error codes are defined according to [1] RFC 6749, section 4.1.2.1

Example URL for authentication:
https://api.paymont.eu/oauth2/auth?state=profil&redirect_uri=https://www.mypfm.cz/start&client_id=MyPFM&response_Type=code_grand&scope=aisp

2.8.2.1.4 Get token resource

If the TPP receives an authorization code (**code**) based on a previous request (see chapter 2.8.2.1.3) and the string specified in the **state** entry is valid (the state value in the response is the same as the state value specified in the request), the TPP can request access tokens from the ASPSP using the authorization code. The TPP shall send the **client_id and client_secret** together with this authorisation code (which shall be provided in the request body)**.**

Endpoint: POST https://api.paymont.eu/oauth2**/token**

Request			
Attribute	**Mandatory**	**Type**	**Description**
code	Yes	string	Authorization code returned from the authentication flow (code grant)
client_id	Yes	String	Id of the TPP application
client_secret	Yes	String	Security code issued by the bank for the TPP application (client_id)
redirect_uri	Yes	URL	Redirect URL matching the URL passed in the authentication request
grant_Type	Yes	**authorization_code**	According to the current OAuth2 definition/custom, this value will be the authorization_code when the code is exchanged for the token pair **access_token** and refresh_token.

Response			
Attribute	Mandatory	Type	Description
access_token	Yes	string	A short-lived (in some cases one-time) token (token validity is 3600s) that can be re-generated using the refresh_token. This token is used to authorize a request to the API.
expires_in	Yes	number	Remaining time until the access_token expires - in seconds.
refresh_token	Yes	String	Long-term token (valid for 90 days) issued in exchange for an one-time code.
token_Type	Yes	String	Type of token “Bearer”
acr	No	Number	Level of verification. Takes values from 0 to 4. Default 3. The value '0' corresponds to nonSCA.
scope	No	String	A space-separated list of Scope for which the token is issued (an extra item compared to OBS).

Error codes

- Error codes are defined according to [1] RFC 6749, Chapter 5.2

Error codes are defined according to [6] chapter 1.4.4. 2a Get token resource

2.8.2.1.5 Access token recovery

The TPP can request a new access_token after the access_token expires by refreshing the token. To do this, the "Get Token" resouce can be used with the parameters below.

Endpoint: POST https://api.paymont.eu/oauth2**/token**

Request			
Attribute	Mandatory	Type	Description
client_id	No	string	TPP application ID
grant_Type	Yes	**refresh_token**	By current OAuth2 definition/custom, this value will be **refresh_token** if the access_token is refreshed based on refresh_token.
refresh_token	Yes	String	Valid **refresh_token for which the access_token is replaced**

Response			
Attribute	Mandatory	Type	Description
access_token	Yes	string	A short-lived (in some cases one-time) token (token validity is 3600s) that can be re-generated using the refresh_token. This token is used to authorize a request to the API.
token_Type	Yes	String	Type of token "Bearer"
expires_in	Yes	number	Remaining time until the access_token expires - in seconds.
Acr	No	number	Level of verification. Takes values from 0 to 4. Default 3 Nobo 4. The value "0" automatically corresponds to nonSCA. Values 1 to 4 correspond to values defined by ISO 29115.

Error codes

› Error codes are defined according to [1] RFC 6749, chapter 5.2

For an example of use see source [6] chapter 5.2.4.

2.9 Description of methods available to service providers (TPPs) via the PSD2 API

2.9.1 Services for AISP (Account enquiries, transaction overview)

The chapter defines the list of methods provided for the AISP.

2.9.1.1 Prerequisites for using API methods for AISP

a/ the use of the TPP certificate is required - the TPP is traced in the IB database in the TPP table on the basis of the licence number (including the prefix used) specified in the certificate used by the TPP in communication - the identical licence number must be specified in the TPP record in the IdentifierInCertificate item in the IB database

b/ the traced TPP record is valid,

c/ the TPP has AISP enabled in the IB database record

d/ the registered TPP application has the AISP service enabled

e/ the certificate used by the TPP for communication has the AISP service enabled

f/ the TPP has used an access_token in the request header (defined in the context of "OAuth2 Authorization Code Grant"), based on which a valid consent created by DispoNont is traced on the bank side in the DispoNont-application links of the TPP.

g/ client authorization is required (the TPP application has AISP service enabled from DispoNont in the traced consent)

2.9.1.2 List of methods used for the AISP service

Endpoint	Method	Description
/api/v1/**accounts/{id}/balance**	GET	Account balance - through this service, the third party authorised by the dispoNont gets an overview of the balances of the dispoNont's bank account held with the bank in question
/api/v1/**accounts/{id}/transactions**	GET	Transaction overview - through this service, an authorised third party receives an overview of transactions
/api/v2/accounts	GET	List of client's payment accounts - on request, the service returns a list of accounts that dispoNont has specified in the consent to use with a specific TPP (not a list of all dispoNont accounts) without balances

2.9.1.3 Header definition

Request header

Attribute	Mandatory	Type	Description
Content-Type	Yes	String	Defines the Media Type of the MIME resource. For example, application/json Nobo application/x-www-form-urlencoded (OAuth2/auth resources)
Authorization	Yes	String	Authorization type defined according to RFC 6750 - The OAuth 2.0 Authorization Framework: Bearer Token Usage
TPP-Name	Yes	Text	Name of the provider (third party) that created the request
TPP-Identification	No	Text	Identification (license number) of the provider (third party) that created the request

Response header

Attribute	Mandatory	Type	Description
Content-Type	Yes	String	application/json Nobo application/xml

2.9.1.4 AISP operation: account balance

The balance of a specific client account according to the reference account id.

Endpoint: GET https://api.paymont.eu/api/**v1/accounts/{id}/balance**

Query request parameters

Method: accounts/{id}/balance			
Attribute name	Format	Mandatory	Note
Id	Text	Yes	API Payment account identifier from the response to the account overview query. The account must be included in the tracked consent from the dispositor

Response

Method: accounts/{id}/balance			
Attribute name	Format	Mandatory	Note
balances	Type:ArrayOfAccountsInformationResponseBalance	Yes	Array of balances

Error codes		
HTTP Status	Error code	Description
400	parameter_missing	A mandatory parameter is missing.
400	parameter_invalid	Invalid value of the input parameter.
401	UNAUTHORISED	Missing access token = user is not authenticated Missing certificate = provider is not authenticated
403	FORBIDDEN	Authentication with an invalid certificate or expired access token, or a call that does not match a third-party license.
404	ID_NOT_FOUND	Invalid or Unknown Account ID
500, 503	server_error	Authorization server error.
Use of other http status codes and error codes according to [1] RFC 6749, chapter 5.2		

2.9.1.4.1 Definition of type ArrayOfAccountsInformationResponseBalance

	Method: accounts/{id}/balance - Type: ArrayOfAccountsInformationResponseBalance				
Attribute name (each column represents one level in the JSON structure)			Format	Mandatory	Note
amount	value		decimal (2 decimal points)	Yes	Balance value
	currency		String (3)	Yes	ISO 4217 balance currency code - 3 capital letters
creditDebitIndicator			enum	Yes	Indicator abbreviation Credit / Debit CRDT (Credit) DBIT (Debit)
date	dateTime		dateTime	Yes	Balance update date
Type	codeOrProprietary	code	enum	Yes	Type of balance CLAV (Available Balance Available) CLBD (current balance)

2.9.1.5 AISP operations: overview of transactions

Through this service, the third party authorised by the user receives an overview of the transactions made on the customer's bank account within the specified timeframe. The transaction history only includes transactions that affect the balance (bookings, posted transactions). Transactions are sorted from most recent to oldest.

Endpoint: POST https://api.paymont.eu/api/**v1/accounts/{id}/transactions{?fromDate,toDate,page,size}**

Query request parametres

Method: /accounts/{id}/transactions{?fromDate,toDate,page,size}			
Attribute name	Format	Mandatory	Note
id	String	Yes	API Payment account identifier from the response to the account overview query The account must be included in the tracked consent from the user
fromDate	dateTime	No	The start date of the period for the transaction history. The default value is the current day
toDate	dateTime	No	The end date of the transaction history period. The bank allows you to view transactions up to 2 years old The default value is the current day.
page	integer	No	The page sequence number with respect to the page size for the recordset. The default value is 0 (first page).
pageSize	integer	No	The number of records included on a single page for display. The default value is 50 records. The maximum value allowed is **100** records per page.

Response

Method: /accounts/{id}/transactions			
Attribute name	Format	Mandatory	Note
pageCount	integer	No	Total number of pages
Transactions	Type: ArrayOfAccountsTransactionsResponseTransaction	Yes	Array of transactions

Error codes		
HTTP Status	Error code	Description
400	**parameter_missing**	A mandatory parameter is missing.
400	**parameter_invalid**	Invalid value of the input parameter.
400	**DT01**	Invalid date
401	**UNAUTHORISED**	Missing access token = user is not authenticated Missing certificate = provider is not authenticated
403	**FORBIDDEN**	Authentication with an invalid certificate or expired access token, or a call that does not match a third-party license.
404	**ID_NOT_FOUND**	Invalid or unknown account ID
404	**PAGE_NOT_FOUND**	Query a non-existent page
500, 503	**server_error**	Authorization server error.

For an example of use see source [6] chapter 5.2.6.

2.9.1.5.1 Definition of type ArrayOfAccountsTransactionResponseTransaction

Method: accounts/transactions - Type: ArrayOfAccountsTransactionsResponseTransaction					
Attribute name (each column represents one level in the JSON structure)			Format	Mandatory	Note
entryReference			String (35)	No	unique transaction identifier assigned by the bank
amount	**value**		decimal (2 decimal points)	Yes	The value of the transaction amount.
	currency		String (3)	Yes	Currency of the transaction amount according to ISO 4217 - 3 capital letters
creditDebitIndicator			enum	Yes	Indicator abbreviation Credit / Debit CRDT (Credit) DBIT (Debit)
reversalIndicator			boolean	No	The flag indicates whether this is a reverse transaction (cancellation) true: This is a reversal false: It is not a reversal
status			enum	Yes	The status of the item (debited Nobo credited payments) in the account from the bank's perspective. In the statement only: - posted items (BOOK), - blocked items, (PDNG).
bookingDate	**date**		dateTime	Yes	Date of processing/charging of the payment by the bank in ISODate format, or ISODateTime, i.e. YYYYMM-DD, or YYYY-MM-DDThh:mm:ss.sTZD.
valueDate	**date**		dateTime	Yes	Due date/currency of payment in ISODate format, respectively. ISODateTime, i.e. YYYYMM-DD, or YYYY-MM-DDThh:mm:ss.sTZD.
bankTransactionCode	proprietary	**code**	String	Yes	Transaction Type category code from the SBA code list.
		issuer		Yes	Identification of the issuer of the bank transaction codebook, which takes the value of CBA (CBA = Czech Banking Association)

entryDetails	transactionDetails		Type:AccountsTransactionsResponseTransactionDetail	Yes	Transaction detail items (turnover detail). Each turnover detail starts with this pairing:
					"entryDetails": {
					"transactionDetails": {

2.9.1.5.2 Definition of type AccountsTransactionsResponseTransactionDetail

Method: accounts/transactions - Type: ArrayOfAccountsTransactionsResponseTransaction							
Attribute name (each column represents one level in the JSON structure)					Format	Mandatory	Note
additionalTransactionInformation					Text (500)	No	Bank Transaction Description
amountDetails	instructedAmount	**amount**	**value**		Decimal (2 decimal points)	Yes, when used „InstructedAmounts“	The value of the transaction amount.
			currency		String (3)		Currency of the transaction amount according to ISO 4217 - 3 capital letters
	counterValueAmount	**amount**	**value**		Decimal (2 decimal points)	Yes, when used „CounterValueAmount“	The final amount and currency of the payment that has been requested by the client to be transferred.
			currency		String (3)		
		currencyExchange	**sourceCurrency**		String (3)	Yes, when used „currencyExchange“	The currency of the client's account (original currency).
			targetCurrency		String (3)	No	Currency of payment (target currency).
			exchangeRate		Decimal (2 decimal points)	Yes, when used „currencyExchange“	The exchange rate used to convert from the instructed currency to the currency of the target account.
references	accountServicerReference				String (35)	No	Unique transaction ID generated by the bank (e.g. it is also the number under which the payment was stored in IB).
	clearingSystemReference				String (35)	No	A bank-defined code value identifying the Payment Type or the name of the Payment Type used. For card transactions - identification of the card association.
	chequeNumber				String (35)	No	Used for card transactions Card number format **** **** **** 1111

	endToEndIdentification				String (35)	No	A unique identification entered by the client initiating the payment, which is used to make it unmistakable (e.g. a variable symbol can be filled in here).
	paymentInformationIdentification				String (35)	No	Additional/other bank reference assigned to the payment by the bank, in the case of payments from payment cards, the sequence number of the payment card may be added, or a specific symbol may be filled in.
	mandateIdentification				String (35)	No	Identification of the payment entered by the third party, or a constant symbol may be entered here (for SEPA direct debits, the Unique Mandate Reference for the SEPA direct debit is specified as a mandatory field)
	messageIdentification				String (35)	No	Payment ID (the assumed identification of the payment entered by the client when initiating the payment)
relatedAgents	creditorAgent	financialInstitutionIdentification	**Bic**		String (11)	Yes, when used „creditorAgent“ and not used „memberIdentification“	BIC / SWIFT code of the beneficiary's bank
			clearingSystemMemberIdentification	**memberIdentification**	String (35)	Yes, when used „creditorAgent“ “ and not used „Bic“	code of the beneficiary's bank according to the CNB bank codebook
	debtorAgent	financialInstitutionIdentification	**Bic**		String (11)	Yes, when used „debtorAgent“ and not	BIC / SWIFT code of the payer's bank

						used „memberIdentification“	
			clearingSystemMemberIdentification	**memberIdentification**	String (35)	Yes, when used „debtorAgent“ and not used „Bic“	code of the payer's bank according to the CNB bank codebook
relatedDates	acceptanceDateTime				Date	No	The date the transaction was entered (the date the transaction was received by the bank).
relatedParties	creditor	name			String (70)	No	Name of recirver.
	creditorAccount	identification	**iban**		IBAN2007Identifier	Yes, when not entered „Other“	IBAN of reciever.
			other	**identification**	String (34)	Yes, when not entered „iban“	account number of the recipient in national (CZ) format
	debtor	**Name**			String	No	Name of the payer
	debtorAccount	Identification	**iban**		IBAN2007Identifier	Yes, when not entered „Other“	Unique identification of the payer's account (IBAN).
			other	**identification**	String (34)	Yes, when not entered „iban“	account number in national (CZ) format
	tradingParty	Identification			String (35)	No	Unique third party identification. For card transactions, the merchant ID is provided here.
		merchantCode			String (4)	No	Trader Code Code (MCC) coordinated by MasterCard and Visa.
		name			String	No	Name of the third party. For card transactions, the name of the merchant is given here.
remittanceInformation	unstructured				Text(140)	No	Text for transaction recipient

	structured	creditorReferenceInformation	**reference**		String (35)	No	Collection of payment symbols. The first 2 characters of each value in the field define the Symbol Type. The colon is followed by the symbol value (max. 10 numbers). E.g. "reference": ["VS:123"] means variable symbol=123 If no variable, specific or constant symbol was filled in the payment, then the whole structure remains Structure is empty.

2.9.1.6 AISP operations: Account List

Endpoint: GET https://api.paymont.eu/api/**v1/accounts**

Request

The request body does not contain any Attributes.

Response (unless an error occurs during the processing of the request)

Method: accounts			
Attribute name	Format	Mandatory	Note
accounts	Type: ArrayOfAccountInfo	Yes	Array of balances

Error codes		
HTTP Status	Error code	Description
400	**parameter_invalid**	Non-valid value of the input parameter.
401	**unauthorised**	Missing access token = user is not authenticated Missing certificate = provider is not authenticated
403	**forbidden**	Authentication with an invalid certificate or expired access token, or a call that does not match a third-party license.
500, 503	**server_error**	Authorization server error.
Use of other http status codes and error codes according to [1] RFC 6749, chapter 5.2		

For an example of use see source [6] chapter 3.1.3.

2.9.1.6.1 Definition of type ArrayOfAccountsInfo

Method: accounts - Type: ArrayOfAccountInfo				
Attribute name (each column represents one level in the JSON structure)		Format	Mandatory	Note
id		Text	Yes	Unique API User Account Identifier
identification	Iban	String (34)	No	IBAN of the dispositor's account (deviation from the OBS standard, the IBAN item is not given if it is the account number of FT (financial markets) clients)
	Other	String	No	account number in national (CZ) format
currency		String (3)	Yes	Account currency (ISO 4217 currency code - 3 capital letters)
nameI18N		String	No	account name or user account name, if available
productI18N		String	No	Product name
servicer	bankCode	Text	No	Bank code
	countryCode	String (2)	No	ISO 3166 bank country (2 characters)
	Bic	String (35)	No	Bank BIC code (ASPSP)

2.9.2 Services for PISP (Payment Creation, Payment Status Detection, Payment Authorization)

The chapter defines the list of methods provided for PISP.

Note: through the described API, the TPP application can ONLY serve instructions that have been entered by the application itself.

2.9.2.1 Prerequisites for using API methods for PISP

a/ the use of the TPP certificate is required - the TPP is traced in the IB database in the TPP table on the basis of the licence number (including the prefix used) specified in the certificate used by the TPP in communication - the identical licence number must be specified in the TPP record in the IdentifierInCertificate item in the IB database
b/ the traced TPP record is valid,
c/ the TPP has PISP enabled in the IB database record
d/ the registered TPP application has PISP service enabled
e/ the certificate used by the TPP for communication has the PISP service enabled
f/ the TPP has used an access_token in the request header (generated in the context of the "OAuth2 Authorization Code Grant"), based on which a valid consent created by the dispatcher is traced on the bank side in the TPP's Disponent-application bindings.
g/ client authorization is required (the TPP application has PISP service enabled from the dispoPoNont in the tracked consent)

2.9.2.2 List of methods used for the PISP service

Endpoint	Method	Description
/api/v1/**accounts/balanceCheck**	POST	**Sufficient funds inquiry** - through this method, the TPP can verify whether the client has sufficient funds in the bank account to which the TPP has issued the card to complete the card transaction
/api/v1/**payments**	POST	**New** payment (payment initiation) - through this method, a third party authorized by the user initiates (creates) one non-eCommerce payment order from the user's bank account. The TPP must then initiate the payment authorization (see section 2.5.2.3 for workflow)
/api/v1/**payments/{paymentId}**	GET	**Pledged/Initiated Payment Info** - Provide details of a payment initiated via the "Payment Initiation" interface
/api/v1/**payments/{paymentId}/status**	GET	**Inquiry on the status of established/initiated payment -** through this service the TPP is enabled to find out the status of the payment order
/api/v1/**payments/{paymentId}**	DELETE	**Deleting an established unauthorised payment** - this service enables cancellation of a payment that has not yet been authorised and that has been created via the PISP New Payment service (payment initiation)
/api//v1/**payments/{paymentId}/sign**	POST	**Authorization ID geofencing -** Searches for a new SignId**.**
/api//v1/**payments/{paymentId}/sign/{signId}**	POST	**Initiate Payment Authorization -** by calling this method, the third party will receive a URL in the response to perform a Federated Authorization of the selected Payment Order. Only Federated Payment Authorization is supported within the solution - redirecting the client to the bank's central authentication page, where the client then authorizes the payment with their authorization device after logging in.

2.9.2.3 Header definition

Request header

Attribute	Mandatory	Type	Description
Content-Type	Yes	String	Defines the Media Type of the MIME resource. For example, application/json or application/x-www-form-urlencoded (OAuth2/auth resources)
Authorization	Yes	String	Authorization type defined according to RFC 6750 - The OAuth 2.0 Authorization Framework: Bearer Token Usage
TPP-Name	Yes	Text	Name of the provider (third party) that created the request
TPP-Identification	No	Text	Identification (license number) of the provider (third party) that created the request

Response header

Attribute	Mandatory	Type	Description
Content-Type	Yes	String	application/json or application/xml

2.9.2.4 PISP Operation: enquiry for sufficient resources

Through this method, the TPP can verify whether the client has sufficient funds in the bank account to which the TPP has issued the card to complete the card transaction

Endpoint: POST https://api.paymont.eu/api/v1/payments/balanceCheck

Request

	Method: accounts/balanceCheck					
Attribute name				Format	Mandatory	Note
exchangeIdentification				String (18)	Yes	Clear query identification
Card	cardHolderName				No	cardholder's name
	maskedPAN				Yes, if „Card“ is used	masked card number
debtorAccount	identification	iban		IBAN2007Identifier	No	IBAN of the account - the item is optional, as IBAN cannot be used for FT Type client accounts The account must be included in the tracked consent from the user
		other	Identification	String (34)	Yes, if „iban“ is not used	Deviation from OBS - new entry
	currency			String (3)	No	Payer's account currency according to ISO 4217
authenticationMethod				Enum	No	Client authentication method Allowed enum values - see chapter 2.9.3.6
Merchant	identification			String (35)	Yes, if „merchant“ level 1 is used	Identification of the trader
	Type				No	Type of trader
	shortName			String (35)	Yes, if „merchant“ level 1 is used	Name of trader
	commonName			String (70)	Yes, if „merchant“ level 1 is used	Trader's name as it appears on the payment receipt
	address			Text (140)	No	Trader's address
	countryCode			String (2)	No	Two-character trader country code according to ISO3166.
	merchantCategoryCode			String (4)	Yes, if „merchant“ level 1 is used	Trader code in relation to Type of trade according to ISO 18245
transactionDetails	totalAmoun			Decimal (2 decimal points)	Yes	The value of the transaction amount.
	currency			String (3)	Yes	Currency of the transaction amount according to ISO 4217 - 3 capital letters

Response (unless an error occurs during the processing of the request)

Method: accounts/balanceCheck			
Attribute name	Format	Mandatory	Note
responseIdentification	Integer	Yes	Clear identification of the answer to the sufficiency of resources query (by the ASPSP)
exchangeIdentification	String (18)	Yes	Re-identification of a sufficiency query by the card issuer
response	Enum	Yes	The result of the call. It can take the following values APPR (sufficient funds in the account) DECL (Insufficient funds in the account)

Error codes		
HTTP Status	Error code	Description
400	**field_missing**	A mandatory parameter is missing.
400	**field_invalid**	Invalid value of the input parameter.
400	**AC02**	[InvalidDebtorAccountNumber] - Invalid account identifier in the request content.
400	**AC09**	[InvalidAccountCurrency] - the currency of the requested account is not valid.
401	**Unauthorised**	Missing certificate
403	**Forbidden**	Calling a method that does not match the license or an invalid certificate.
403	**AG01**	[TransactionForbidden] - non-existent consent to access the given Transaction Type.
400	**AM12**	[InvalidAmount] - incorrectly entered amount.
400, 50x	**NARR**	Narrative - general reason for refusal of payment, with the addition of information about the error.
500, 503	**server_error**	Authorization server error.

2.9.2.5 **PISP Operation: new payment (payment initialization)**

The operation allows initialization of one payment in JSON structure.
PISP sends a request via API containing a payment based on JSON structure.
By sending this request, a payment order is created on the bank side, which is related to a commercial transaction between the PSU and the provider (TPP Type PISP).

Endpoint: POST https://api.paymont.eu/api**/v1/payments**

Domestic Payment Request - Request

		Method: payments				
Attribute name				Format	Mandatory	Note
paymentIdentification	instructionIdentification			String (35)	Yes	identifying an instruction in a third-party application
paymentTypeInformation	instructionPriority			string	No	Priority of instruction. If not filled in, the Normal instruction priority is used. (NORM = Normal, HIGH = Express)
amount	instructedAmount	value		Decimal (2 decimal points)	Yes	Amount in instruction
		currency		String (3)	Yes	Currency of transfer
requestedExecutionDate				ISODate	Yes	Requested execution date. Compared to OBS, this is a mandatory item
debtorAccount	identification	iban		IBAN2007Identifier	No	Payer's account number Deviation from OBS - the item is optional as IBAN cannot be used for FT Type client accounts
		other	identification	String (34)	Yes, if **iban** is not used	Payer's account number in local BBAN format
	currency			String (3) CurrencyCode ISO 4217	No	Currency of the payer's account
creditorAccount	identification	iban		IBAN2007Identifier	No	Recipient's account number Deviation from OBS - this item is optional
		other	identification	String (34)	Yes, if **iban** is not used	Beneficiary's account number in local BBAN format
	currency			String (3) CurrencyCode ISO 4217	No	Currency of the beneficiary 's account
remittanceInformation	unstructured			Text (140)	No	Unstructured report for recipients
	structured	creditorReferenceInformation	reference	String	No	Collection of payment symbols. The first 2 characters of each value in the field define the Symbol Type (VS, KS, SS). The colon is followed by the symbol value (max. 10 numbers). E.g. "reference": ["VS:123"] means variable symbol=123

SEPA payment order – Request

		Method: payments				
Attribute name				Format	Mandatory	Note
paymentIdentification	instructionIdentification			String (35)	Yes	identifying an instruction in a third-party application
	endToEndIdentification			String (35)	Yes	identification agreed between payer and payee
paymentTypeInformation	instructionPriority			string	No	Priority of instruction. If not filled in, the Normal instruction priority is used. (NORM = Normal)
amount	instructedAmount	value		Decimal (2 decimal points)	Yes	Amount in instruction
		currency		String (3)	Yes	Currency of transfer (must be EUR)
requestedExecutionDate				ISODate	Yes	Requested date of execution. Compared to OBS, this is a mandatory item
debtorAccount	identification	iban		IBAN2007Identifier	No	IBAN of the payer's account; deviation from OBS - the item is optional as IBAN cannot be used for FT Type clients
		other	identification	String (34)	Yes, if **iban** is not used	Payer's account number in local BBAN format
	currency			String (3) CurrencyCode ISO 4217	No	Currency of the payer's account
creditorAccount	identification	iban		IBAN2007Identifier	Yes	IBAN of the beneficiary‘s account
creditorAgent	financialInstitutionIdentification	bic		String (11)	Yes	BIC / SWIFT code of the beneficiary 's bank
creditor	name			String (70)	Yes	Name of the beneficiary
	postalAddress	streetName		String (70)	No	Mailing address (street)
		buildingNumber		String (16)	No	Mailing address (building number)
		townName		String (35)	No	Mailing address (town)
		postCode		String (16)	No	Mailing address (postal code)
		country		String (2)	No	Mailing address (ISO3166 country code)
remittanceInformation	unstructured			Text (140)	No	Unstructured report for beneficiaries

Foregin payment - Request

		Method: payments				
Attribute name				Format	Mandatory	Note
paymentIdentification	instructionIdentification			String (35)	Yes	identifying an instruction in a third-party application
paymentTypeInformation	instructionPriority			string	No	Priority of instruction. If not filled in, the Normal instruction priority is used. (NORM = Normal)
amount	instructedAmount	value		Decimal (2 decilam points)	Yes	Amount in instruction
		currency		String (3)	Yes	Currency of transfer (must be EUR)
requestedExecutionDate				ISODate	Yes	Requested date of execution. Compared to OBS, this is a mandatory item
chargeBearer				Enum	No	Fee payer
debtorAccount	identification	iban		IBAN2007Identifier	No	IBAN of the payer's account Deviation from OBS - this item is optional
		other	identification	String (34)	Yes, if **iban** is not used	Payer's account number in local format
	currency			String (3) CurrencyCode ISO 4217	No	Currency of the payer's account
creditorAccount	identification	iban		IBAN2007Identifier	Yes	IBAN of the beneficiary‘s account
		other	identification	String (34)	Yes	Other account number format
creditorAgent	financialInstitutionIdentification	bic		String (11)	Yes	BIC / SWIFT code of the beneficiary 's bank
creditor	name			String (70)	Yes	Name of the beneficiary
	postalAddress	streetName		String (70)	No	Mailing address (street)
		buildingNumber		String (16)	No	Mailing address (building number)
		townName		String (35)	No	Mailing address (town)
		postCode		String (16)	No	Mailing address (postal code)
		country		String (2)	No	Mailing address (ISO3166 country code)
remittanceInformation	unstructured			Text (140)	No	Unstructured message for beneficiaries

Response for all of the above Payment Types (unless an error occurs while processing the request)

The output structure is the same as the input, plus the following values are returned:

			Method: payments/standard/iso		
Attribute name			Format	Mandatory	Note
paymentIdentification	transactionIdentification		String	Yes	Number of the order, created in the E-Banking database, in other queries it is used as **{paymentId}** in the input
paymentTypeInformation	serviceLevel	code	String	Yes	Type of payment entered DMCT = Domestic Credit Transfer Order ESCT = SEPA payment XBCT = Foreign payment EXCT = EEA foreign payment NXCT = Foreign payment outside the EEA
signInfo	signId		String	No	The identifier of the authorization process for a specific transaction.
	state		Enum	Yes	Command status The status can take the following values: ACTC - Authentication and syntactical and semantical validation are successful (in IB the created command will have the status "To be signed")

Error codes		
HTTP Status	**Error code**	**Description**
400	**field_missing**	A mandatory parameter is missing.
400	**filed_invalid**	Invalid value of the input parameter.
400	**AC02**	[InvalidDebtorAccountNumber] - invalid account identifier in the request content.
400	**AC03**	[InvalidCreditorAccountNumber] - the beneficiary's account number given in an invalid format (note: validated only for in-house payments).
400	**AC10**	[InvalidDebtorAccountCurrency] - the specified currency of the payer's account does not match the currency of the customer's account for the given account number held at the bank (the account currency is optional).
400	**AM11**	[InvalidTransactionCurrency] - an untraded/unsupported currency is specified in the request.
400	**AM12**	[InvalidAmount] - incorrectly entered amount.
400	**FF01**	[Invalid File Format] - invalid JSON format or other technical problem with query processing.
400	**BE19**	[InvalidChargeBearerCode] - an invalid fee type for that transaction type.
400	**DT01**	[InvalidDate] - Invalid due date.
400, 50x	**NARR**	Narrative - general reason for refusal of payment, with the addition of information about the error.
400	**RC07**	[InvalidCreditorBICIdentifier] - invalid SWIFT / BIC code of the beneficiary's bank.
400	**RC10**	[InvalidCreditorClearingSystemMemberIdentifier] - invalid identification of the beneficiary's bank code.
403		[TransactionForbidden] - non-existent consent to access the PISP operation.
500, 503	**server_error**	Authorization server error.

2.9.2.6 PISP operations: status of pledged/initiated payments

The operation provides information about the processing status of the received payment transaction based on the {paymentId} parameter.

Endpoint: GET https://api.paymont.eu/api/**v1/payments/{paymentId}/status**

Input URI parameters

- paymentId - payment identifier in e-Banking, Type: string (mandatory)

Request

The request body does not contain any attributes.

Response

Method: payments/{paymentId}/status			
Attribute name	Format	Mandatory	Note
instructionStatus	Enum	Yes	Command status The status can take the following values: - RJCT (Rejected) - PDNG (Authorized) - ACTC (WaitingForSignatures) - ACSP (InProgress, Exported) - ACSC (Accepted by the banking system) - ACCR (Payment cancelled by client) - OTHR (Reserve)

Error codes		
HTTP Status	Error code	Description
400	**parameter_missing**	Mandatory parameter missing.
400	**parameter_invalid**	Invalid value of the input parameter.
500, 503	**server_error**	Authorization server error.

2.9.2.7 PISP operation: info about established/initiated payment

Resource for displaying the information of a received payment transaction based on the **{paymentId}** parameter. This is a payment that has been received for authorization but has not yet been authorized by the client. The Resource only works with transactions established through a specific provider.

Endpoint: GET https://api.paymont.eu/api/**v1/payments/{paymentId}**

Input URI parameters

- paymentId - payment identifier in e-Banking, Type: string (mandatory)

Request

The request body does not contain any attributes.

Response

The output of the message is information about a payment that has been established or initiated. Therefore, the list of elements corresponds to the elements from the resource (request+response) New payment. See chapter 2.9.2.5.

Error codes		
HTTP Status	Error code	Description
401	UNAUTHORISED	Invalid/missing certificate = provider is not authenticated.
501	NOT_IMPLEMENTED	Non-implemented method
404	TRANSACTION_MISSING	Calling a method that does not match the license or an invalid certificate.
500, 503	server_error	Authorization server error.

2.9.2.8 PISP operation: deletion of an established unauthorized payment

The operation allows you to cancel a payment that has been initiated through an identical PISP (third party) using the "New Payment (Payment Initiation)" service. The payment can be cancelled until the payment is authorised by the client.

Endpoint: DELETE https://api.paymont.eu/api/**v1/payments/{paymentId}**

Input URI parameters

- paymentId - payment identifier in e-Banking, Type: string (mandatory)

Request

The request body does not contain any attributes.

Response (if no error occurs while processing the request)

Response (HTTP 204) - entered payment deleted

Error codes		
HTTP Status	Error code	Description
401	UNAUTHORIZED	Invalid/missing certificate = provider is not authenticated.
501	NOT_IMPLEMENTED	Non-implemented method
404	TRANSACTION_MISSING	Calling a method that does not match the license or an invalid certificate.
500, 503	server_error	Authorization server error.

2.9.2.9 PISP Authorization ID Generation

Generates a new request to generate a signId for payment authorization. It is used in cases where:

- The original authorization request (e.g. originating as output from an IF-200 call) has expired. Authorization requests are limited to 5 minutes from the time they are created.
- Authorization of the request by the user was not performed at will - the user refused the authorization instruction - and is interested in a retry of the authorization
- For technical reasons on the TPP side of the application

Note: by establishing a new authorization, the original and unused authorization requirements remain in effect.

Endpoint: POST https://api.paymont.eu/api/**v1/payments/{paymentId}/sign**

Input URI parameters

- paymentId - payment identifier in e-Banking, Type: string (mandatory)

Request

The request body does not contain any attributes.

Response

	Method: payments/{paymentId}/status			
Attribute name		Format	Mandatory	Note
scenarios		Array of supported authorization methods	Yes	collection of possible payment authorization methods, in this case, only one type of payment authorization is returned - Federated Authorization, or redirecting the user to the bank's page where the authorization will take place [USERAGENT-REDIRECT]
signInfo	state	string	Yes	Command Authorization Status The status in this case can only have a value: › OPEN = Newly created authorization request (the command has internal status to sign and the signId is valid)
	signId	String	Yes	Unique identifier of the token generated to authorize the transaction (valid for 5 minutes from generation)

Error codes		
HTTP Status	Error code	Description
401	**UNAUTHORIZED**	Invalid/missing certificate = provider is not authenticated.
403	**FORBIDDEN**	Invalid/missing certificate = provider is not authenticated.
501	**NOT_IMPLEMENTED**	Non-implemented method
404	**TRANSACTION_MISSING**	Calling a method that does not match the license or an invalid certificate.
500, 503	**server_error**	Authorization server error.

2.9.2.10 PISP Operation: initiate payment authorization

This resource is used to start the authorization method from the selected scenario.
Only Federated Authorization is supported in the Citfin PSD2 solution.
The input is a JSON object containing the desired Type of the authorization method - CODE and all elements specific to this step.
The output of this resource is a list of values needed to complete the authorization.
The response for the CODE corresponding to the federated authorization will be the response URL and parameters to redirect to the federated authorization page.

Endpoint: POST https://api.paymont.eu/api**/v1/payments/{paymentId}/sign/{signId}**

Input URI parameters

- paymentId - payment identifier in e-Banking, Type: string (mandatory)
- signId - identifier of the authorization request, Type: string (mandatory)

Request

Method: payments/{paymentId}/sign/{signId}			
Attribute name	Format	Mandatory	Note
authorizationType	String	Yes	Selected payment authorization method supported by the bank Only the following method is supported: USERAGENT-REDIRECT = Federated authorization, i.e. redirecting the user to the bank's website where the authorization will take place.

Response (unless an error occurs during the processing of the request)

	Method: payments/{paymentId}/sign/{signId}			
Attribute name		Format	Mandatory	Note
authorizationType		String	Yes	The selected payment authorization method. The next fields of the answer describe this selected authorization method USERAGENT-REDIRECT = Federated authorization, i.e. redirecting the user to the bank's page where the authorization will take place
href	id	String	No	Not supported
	url	string	Yes	URI address for authorizing a specific instruction. Note: the complete URL for authorizing the request is constructed by concatenating the URI specific to the issued banking API and the following returned URI
method		string	Yes	HTTP method to redirect to Federated Authorization (GET)
signInfo	State	String	Yes	Command Authorization Status The status in this case can only have a value: **OPEN** = Newly created authorization request (the command has an internal status to sign and the signId is valid) **DONO** = Authorization successful (the command has changed status) **EXPIRED** = The order is available for signature, but the user did not authorize the order during the authorization request validity period. To repeat the authorization, a new request must be created to generate a new SignID and repeat the authorization.
	SignId	String	Yes	Unique identifier of the current transaction authorization

Error codes		
HTTP Status	Error code	Description
400	**AUTH_LIMIT_EXCEE DED**	SignId expired at the time the bank accepted the authorization initiation request.
401	**UNAUTHORISED**	Invalid/missing certificate = provider is not authenticated.
403	**FORBIDDEN**	Invalid/missing certificate = provider is not authenticated.
404	**ID_NOT_FOUND**	Required id does not exist.
500, 503	**server_error**	Authorization server error.

2.9.3 CISP (Confirmation of Sufficient Funds on Account)

The chapter defines a list of methods provided for CISP.

2.9.3.1 Prerequisites for using API methods for CISP

a/ the use of the TPP certificate is required - the TPP is traced in the IB database in the TPP table on the basis of the licence number (including the prefix used) specified in the certificate used by the TPP for communication - the identical licence number must be specified in the TPP record in the IdentifierInCertificate item in the IB database
b/ the traced TPP record is valid,
c/ the TPP has CISP service enabled in the IB database record
d/ the registered TPP application has the CISP service enabled
e/ the certificate used by the TPP for communication has the CISP service enabled
f/ the TPP has used an access_token in the request header (generated in the context of the "OAuth2 Authorization Code Grant"), based on which a valid consent created by the User is traced on the bank side in the Disponent-application links of the TPP.
g/ client authorization is required (the TPP application has CISP service enabled from the dispositor in the tracked consent)

2.9.3.2 List of methods used for the CISP service

Endpoint	Method	Description
/api/v1/**accounts/balanceCheck**	POST	Sufficient funds inquiry - through this method, the TPP can verify whether the client has sufficient funds in the bank account to which the TPP has issued the card to complete the card transaction

2.9.3.3 Token for CISP operation

For the CISP operation the access_token obtained on the basis of the Authorization Code Grant resource described in chapter 2.8.2.1.3 or alternatively see [1], chapter 4.1.

Generating access_token based on Client Credentials Grant flow **is not supported in the solution.**

2.9.3.4 Header definition

Request header

Attribute	Mandatory	Type	Description
Content-Type	Yes	String	Defines the Media Type of the MIME resource. For example, application/json or application/x-www-form-urlencoded (OAuth2/auth resources)
Authorization	Yes	String	Authorization type defined according to RFC 6750 - The OAuth 2.0 Authorization Framework: Bearer Token Usage
TPP-Name	Yes	Text	Name of the provider (third party) that created the request
TPP-Identification	No	Text	Identification (license number) of the provider (third party) that created the request

Response header

Attribute	Mandatory	Type	Description
Content-Type	Yes	String	application/json or application/xml

2.9.3.5 CISP Operation: query for sufficient resources

Through this method, the TPP can verify whether the client has sufficient funds in the bank account to which the TPP has issued the card to complete the card transaction.

The sufficient funds query is performed against the available account balance (debtorAccount)

Endpoint: POST https://api.paymont.eu/api/v1/payments/balanceCheck

Request

	Method: accounts/balanceCheck					
Attribute name				Format	Mandatory	Note
exchangeIdentification				String (18)	Yes	Unique query identification
card	cardHolderName				No	cardholder's name
	maskedPAN				Yes, if „Card“ is used	masked card number
debtorAccount	identification	iban		IBAN2007Identifier	No	unambiguous identification of the payer's account on which sufficient funds are ascertained IBAN of the account - the item is optional, as IBAN cannot be used for FT Type client accounts The account must be included in the tracked consent from the dispositor
		other	Identification	String (34)	Yes, if „iban“ is not used	Deviation from OBS - new entry
	currency			String (3)	No	Payer's account currency according to ISO 4217
authenticationMethod				Enum	No	Client authentication method Allowed enum values - see chapter 2.9.3.6
merchant	identification			String (35)	Yes, if level 1 "merchant" is used	Trader identification
	Type				No	Type of trader
	shortName			String (35)	Yes, if level 1 "merchant" is used	Name of trader
	commonName			String (70)	Yes, if level 1 "merchant" is used	Trader's name as it appears on the payment receipt
	address			Text (140)	No	Trader's address
	countryCode			String (2)	No	Two-character trader country code according to ISO3166.
	merchantCategory Code			String (4)	Yes, if level 1 "merchant" is used	Trader code in relation to Type of trade according to ISO 18245

transactionDetails	totalAmoun			Decimal (2 decimal points)	Yes	The value of the transaction amount.
	currency			String (3)	Yes	Currency of the transaction amount according to ISO 4217 - 3 capital letters

Response (unless an error occurs during the processing of the request)

Metoda: accounts/balanceCheck			
Attribute name	Format	Mandatory	Note
responseIdentification	Integer	Yes	Clear identification of the answer to the sufficiency of resources query (by the ASPSP)
exchangeIdentification	String (18)	Yes	Re-identification of a sufficiency query by the card issuer
Response	Enum	Yes	The result of the call. It can take the following values APPR (sufficient funds in the account) DECL (Insufficient funds in the account)

Error codes		
HTTP Status	Error code	Description
400	**field_missing**	A mandatory parameter is missing.
400	**field_invalid**	Invalid value of the input parameter.
400	**AC02**	[InvalidDebtorAccountNumber] – non-valid account identifier in the content of the request.
400	**AC09**	[InvalidAccountCurrency] - the currency of the requested account is not valid.
401	**Unauthorised**	Missing certificate
403	**Forbidden**	Calling a method that does not match the license or an invalid certificate.
403	**AG01**	[TransactionForbidden] - non-existent consent to access the given transaction type.
400	**AM12**	[InvalidAmount] – wrong amount entered.
400, 50x	**NARR**	Narrative - general reason for refusal of payment, with the addition of information about the error.
500, 503	**server_error**	Authorization server error.

2.9.3.6 Enum used in the AuthenticationMethod

Abbreviation	Meaning
NPIN	On-liNo PIN authentication (PersonalIdentification Number)
PPSG	Handwritten paper signature.
PSWD	Authentication by a password
SCRT	Electronic commerce transaction secured with the X.509 certificate of a customer
SCNL	ChanNol-encrypted transaction
SNCT	Secure electronic transaction without cardholder certificate
CPSG	Electronic signature capture (handwritten signature);
ADDB	Cardholder billing address verification
BIOM	Biometric authentication of the cardholder
CDHI	Cardholder data provided for verification, for instance social security number, driver license number, passport number
CRYP	Verification of a cryptogram geNorated by a chip card or Another device, for instance ARQC (Authorisation Request Cryptogram).
CSCV	Verification of Card Security Code
PSVE	Authentication based on statistical cardholder behaviour
CSEC	Authentication performed during a secure electronic commerce transaction
ADDS	Cardholder shipping address verification
TOKP	Verification or authentication related to the use of a payment token, for instance the validation of the authorized use of a token

3. Resources

1. *RFC 6749 - The OAuth 2.0 Authorization Framework*, [onliNo]. The InterNot EngiNoering Task Force, October 2012. WWW: https://tools.ietf.org/html/rfc6749

2. *RFC 6750 - The OAuth 2.0 Authorization Framework: Bearer Token Usage,* [onliNo]. The InterNot EngiNoering Task Force, October 2012. WWW: https://tools.ietf.org/html/rfc6750

3. *RFC 7636 - Proof Key for Code Exchange by OAuth Public Clients,* [onliNo]. The InterNot EngiNoering Task Force, September 2015. WWW: https://tools.ietf.org/html/rfc7636

4. *RFC 7519 - JSON Web Token (JWT)*, [onliNo]. The InterNot EngiNoering Task Force, May 2015. WWW: https://tools.ietf.org/html/rfc7519

5. *RFC 7515 - JSON Web Signature (JWS)*, [onliNo]. The InterNot EngiNoering Task Force, May 2015. WWW: https://tools.ietf.org/html/rfc7515

6. *ISO 20022 Financial Services - Universal financial industry message scheme,* [onliNo]. International Organization for Standardization. WWW: https://www.iso20022.org/

7. *OBS / Czech Open Banking Standard*, dokument. WWW: https://cbaonline.cz/upload/644-cobs-rulebook-verze-2-fin-en.pdf