

---

## *OTP PSD2 API developer's documentation*

---

Document version	1.2
Release date	28.04.2023.

## Document History

Version	Date	Author	Description
1.0	14.01.2020	Denis Baban	Initial version
1.1	14.01.2020.	Denis Baban	
1.2	28.04.2023.	Tonči Buljan	PSU-ID flow changes

## Contents

<b>Document History.....</b>	<b>2</b>
<b>1. Definitions.....</b>	<b>4</b>
<b>2. Introduction .....</b>	<b>4</b>
2.1. Standards .....	4
<b>3. Sandbox environment .....</b>	<b>5</b>
3.1. Introduction .....	5
3.2. Developer portal .....	5
3.3. Registration.....	5
3.4. Third party provider registration .....	6
3.5. Create application .....	7
3.6. Client certificate .....	10
2.6. Testing users .....	11
<b>4. XS2A interface.....</b>	<b>12</b>
4.1. Pre-Authentication.....	12
4.2. Getting OAuth token .....	16
4.3. Possible error codes with Pre-Authentication .....	18
<b>5. Strong Customer Authentication (SCA).....</b>	<b>18</b>
5.1. SCA Exemptions .....	18
5.2. Redirect approach .....	18
<b>6. Accounts endpoints .....</b>	<b>19</b>
5.1. Consent request.....	19

5.2.	Get consent request.....	22
5.3.	Get consent status request .....	23
5.4.	Delete consent .....	24
5.5.	Read account list .....	26
5.6.	Read account details .....	27
5.7.	Get balances .....	29
5.8.	Get transactions list .....	30
5.9.	Get transactions details .....	31
5.10.	Start the authorization process for a consent.....	32
5.11.	Consent authorisation using Strong Customer Authentication(SCA) .....	34
5.11.1.	Consent authorisations: redirect SCA approach .....	34
<b>4.</b>	<b>Payments endpoints .....</b>	<b>38</b>
4.1.	Payments initiation .....	38
4.2.	Get payment transaction status .....	40
4.3.	Get payment request .....	41
4.4.	Delete payment request .....	43
4.5.	Update PSU data for payment initiation.....	44
4.6.	Payment authorisation using Strong Customer Authentication(SCA) .....	46
4.6.1.	Payment authorisation: redirect SCA approach.....	46
4.6.2.	Payment authorisation: decoupled SCA approach.....	46
<b>5.</b>	<b>Miscellaneous .....</b>	<b>47</b>

## 1. Definitions

This section offers explanations to the terminology used throughout the document.

<b>TPP (Third Party Provider)</b>	Third Party Provider (TPP) is the provider of an application which the user uses and is not offered by the bank. TPP is the client/consumer of the API and acts on behalf of the user through <b>consent</b> .
<b>PSU (Payment Service User) or user</b>	The user refers to the bank customer who uses the TPP application.
<b>ASPSP (Account Servicing Payments Service Provider)</b>	This is the account servicing provider, i.e, the OTP bank.
<b>PISP (Payment Initiation Service Provider)</b>	This is a service provider who can initiate a payment transaction on behalf of the customer.
<b>Sandbox</b>	Sandbox gives access to a small set of static data and it is used as an example to illustrate what would be returned when using the production API. The Sandbox can be reached within the developer portal at <a href="https://apiportal.sandbox.otpbanka.hr/portal/">https://apiportal.sandbox.otpbanka.hr/portal/</a>
<b>SCA (Strong Customer Authentication)</b>	The process of using a strong (2-factor) identification method to identify the customer.
<b>Authentication</b>	Authentication is the process of verifying that an individual is who it claims to be. This authentication is later used to grant authorization to specific data and functions within a system.
<b>Consent</b>	Consent is the agreement given by the PSU to the TPP to share data from the bank. Consent is stored by the bank and validated by the user according to PSD2. The consent may have a duration or just be used for a single API call.

## 2. Introduction

This document describes how TPPs can connect the PSD2 Solution of OTP sandbox API.

### 2.1. Standards

OTP bank PSD2 API follows standards described in the Berlin Group standard version 1.3.

From multiple options described in BGS we have selected to implement the following:

- Pre Step OAuth authorisation mode. It requires an authentication of a PSU in a pre-step, translating this authentication into an access token. Access token is mandatory for any other API call as described in BGS (4.3 Optional Usage of OAuth2 for PSU Authentication or Authorisation).
- OTP bank offers a redirect and decoupled integration methods as main way of integration for the TPP and the PSU.

The exposure of data is done through RESTful services. For the most part API encodes data in JavaScript Object Notation (**JSON**) format. In some cases **XML** may be used.

The API request and responses must use a UTF-8 character encoding, as is the default for JSON.

### 3. Sandbox environment

#### 3.1. Introduction

The sandbox is used to document the API and offer the TPPs the possibility to view the methods.

Additionally, it is possible that TPPs can test some calls of the API and receive corresponding demo responses. Note that the APIs are not connected to the backend and therefore return generated mock data.

#### 3.2. Developer portal

For testing and mutual partnership purposes, a developer site has been created. Every market participant now can register through the developer's site registration form and create PSD2 API clients. Which later could be used for accessing Xs2a endpoints. Also, there is implemented functionality for generating TPP QWAC, QSEAL certificates for testing purposes. The developer's site can be accessed via <https://apiportal.sandbox.otpbanka.hr/portal/> link.

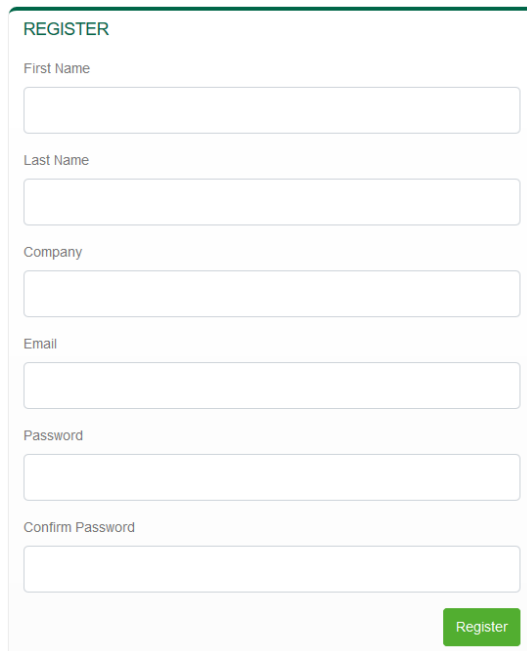
#### 3.3. Registration

User registration form can be accessed from the main page via the Register button (Figure 1).

New users must fill in the following fields:

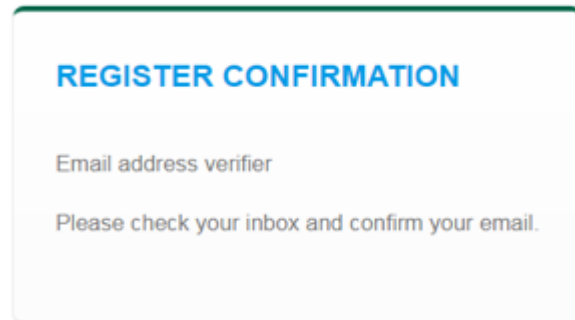
- **First name**  
Required, first name of the user
- **Last name**  
Required, last name of the user
- **Email address**  
Required, unique and valid email address which will be used as a username to access the system
- **Company name**  
Optional, Name of the company the user is working for
- **Password**  
Required, masked, must satisfy password complexity
- **Password confirmation**  
Required, must match the password field

After successful registration you will be redirected to register confirmation page with success message (Figure 2) and an account confirmation email will be sent to your mailbox shortly. Please click confirmation link inside email message to finish user registration process.



The registration form is titled "REGISTER" and contains several input fields: "First Name", "Last Name", "Company", "Email", "Password", and "Confirm Password". Each field is represented by a white rectangular box with a thin border. At the bottom right of the form is a green button labeled "Register".

Figure 1 Registration form

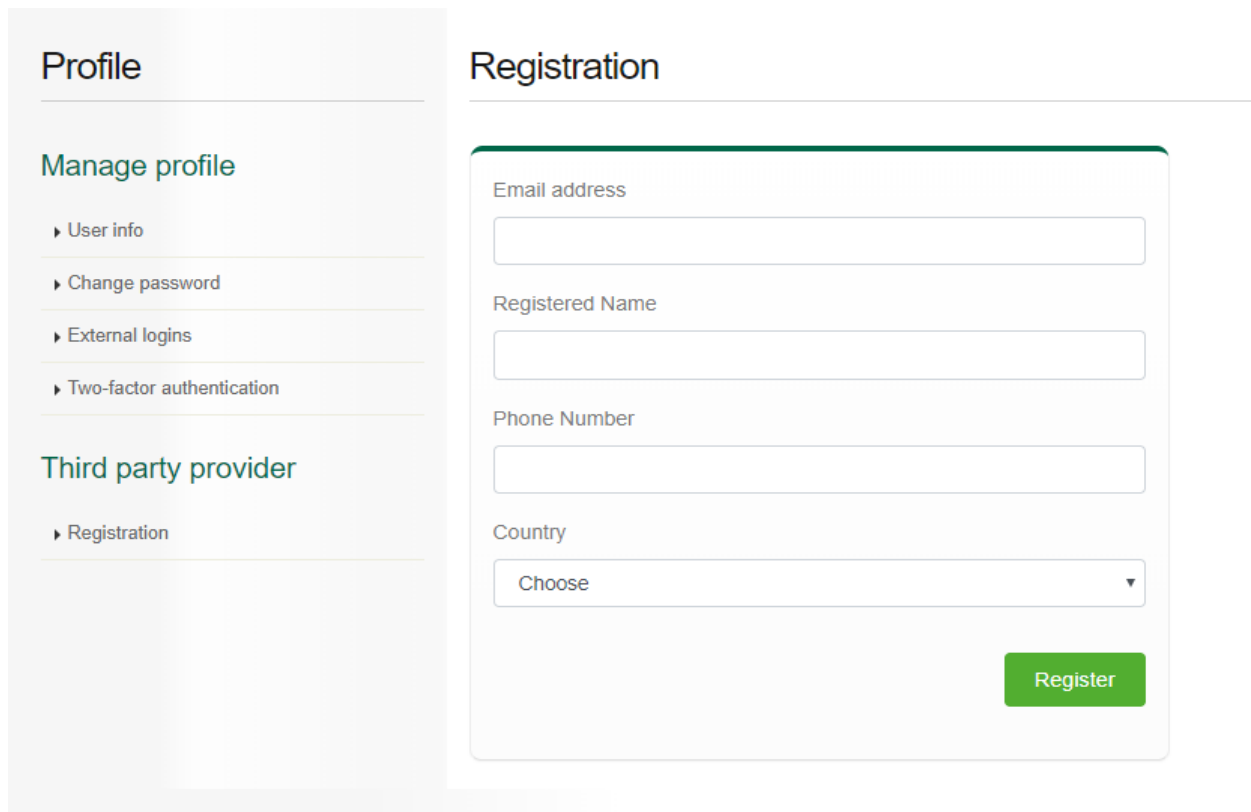


The confirmation message box has a title "REGISTER CONFIRMATION" in blue. Below the title, it says "Email address verifier" and "Please check your inbox and confirm your email." in a smaller, grey font.

Figure 2 Register confirmation message

## 3.4. Third party provider registration

After successful login user will be redirected to the main dashboard page. On the right side of the page, a Profile menu item is displayed. TPP registration form can be accessed from the main page via the Profile/Register tab (Figure 3).



**Profile**

**Manage profile**

- ▶ User info
- ▶ Change password
- ▶ External logins
- ▶ Two-factor authentication

**Third party provider**

- ▶ Registration

**Registration**

Email address

Registered Name

Phone Number

Country

Choose ▼

Register

Figure 3 Third party provider registration form

### 3.5. Create application

A TPP which wants to start working with the PSD2 API firstly has to register client application and get client credentials which later will be used with OAuth2 authorisation code grant flow during the token request process. The first one is *Application* menu item. This menu item will redirect the user to the client application add/edit form (Figure 4). In the newly opened form, a user has to fill in mandatory fields: grant type, client name, *application uri*, *OAuth redirect URL*, *OAuth post logout redirect URL*. An *OAuth redirect URL* is a URL the authorization server will redirect the user back to the application with either an authorization code in the URL. Because the redirect URL will contain sensitive information, it is critical that the service doesn't redirect the user to arbitrary locations. A user has to use the same URL with third-party OAuth client during the authorization process. Using mismatched URL will lead to OAuth error. To allow application set enabled value. A Client Id and Client secret values will be generated after form submit (Figure 5). During edit mode, the user can find out Client ID and Client secret values. The Client secret value is only displayed once after the „generate a new client secret“ button is pressed. The next time you enter the edit form it will be hidden again. So please write down this value and keep it secretly.

Application	Application
<p> <b>App Id</b>  <input type="text" value="16"/> </p> <p> <b>Grant Type</b>  <input type="text" value="Authorization code"/> </p> <p> <b>Client Name</b>  <input type="text"/> </p> <p> <b>Application Uri</b>  <input type="text"/> </p> <p> <b>Redirect Uri</b>  <input type="text"/> </p> <p> <b>Post Logout Redirect Uri</b>  <input type="text"/> </p> <p> <input type="checkbox"/> Enabled         </p> <p> <input type="button" value="Create"/> </p>	<p> <b>App Id</b>  <input type="text" value="16"/> </p> <p> <b>Grant Type</b>  <input type="text" value="Authorization code"/> </p> <p> <b>Client Name</b>  <input type="text" value="tppapp"/> </p> <p> <b>Application Uri</b>  <input type="text" value="http://myserver.com"/> </p> <p> <b>Redirect Uri</b>  <input type="text" value="http://myserver.com/callback"/> </p> <p> <b>Post Logout Redirect Uri</b>  <input type="text" value="http://myserver.com/callback"/> </p> <p> <input checked="" type="checkbox"/> Enabled         </p> <p> <input type="button" value="Edit"/> </p>

Figure 4 Application registration and edit form



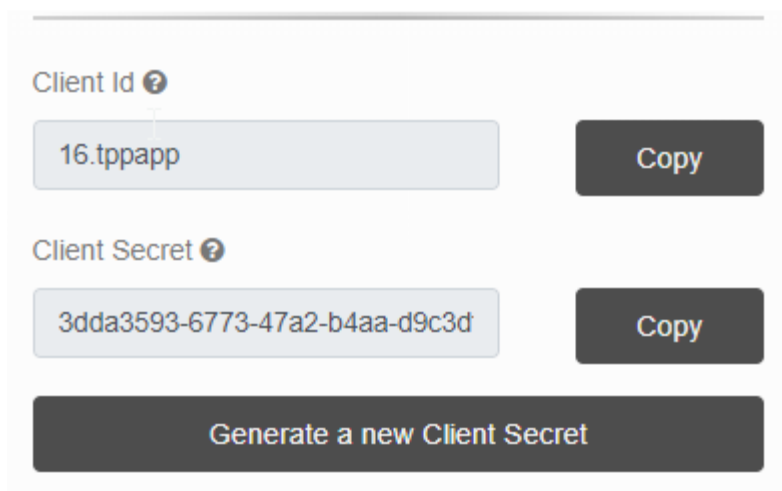


Figure 5 Client ID and client secret

The user can have many client applications with different OAuth client configurations for different testing purposes. A newly created application will always appear on application list (Figure 6).

## Applications

Manage multiple apps



Figure 6 Client application list

## 3.6. Client certificate

Developer's portal provides functionality for TPP QWAC, QSEAL self-signed certificate generation for testing purpose. This could be achieved from the TPP certificate generator menu link. Once the user enters the form it has to fill correctly mandatory fields. The last step a user has to do is to choose certificate service roles. There are four roles (AI, PI, AS, IC) user can choose to include into certificate. Every role gives a TPP client applications right to access to the certain group of endpoints. Deselecting

service role will restrict access to related endpoints. After submitting the form, certificate data will be saved for later use. You can download certificate with private key values. Generated certificate (AuthenticationCertificate.crt) and private key (AuthenticationCertificateKey.key.) should be passed in mTLS transport layer.

### Certificate

Name

TPP Name

Country

Choose

State or province name

TPP State

Locality name

TPP locally name

Organization name

TPP org name

Organizational unit name

TPP org unit name

Common name

TPP common name

Email address

TPP@email

Domain

TPP domain

NCA Id

NCA name

I

☒ Has AI Role

☒ Has PI Role

☒ Has AS Role

☒ Has IC Role

Create

AI ROLE	
	ESTABLISH ACCOUNT INFORMATION CONSENT
	GET ACCOUNT DETAILS OF THE LIST OF ACCESSIBLE ACCOUNTS
	GET BALANCES FOR A GIVEN ACCOUNT
	GET TRANSACTION INFORMATION FOR A GIVEN ACCOUNT
PI ROLE	
	INITIATION OF A SINGLE PAYMENT
AS ROLE	GET CONFIRMATION ON THE AVAILABILITY OF FUNDS
IC ROLE	CARD PAYMENT

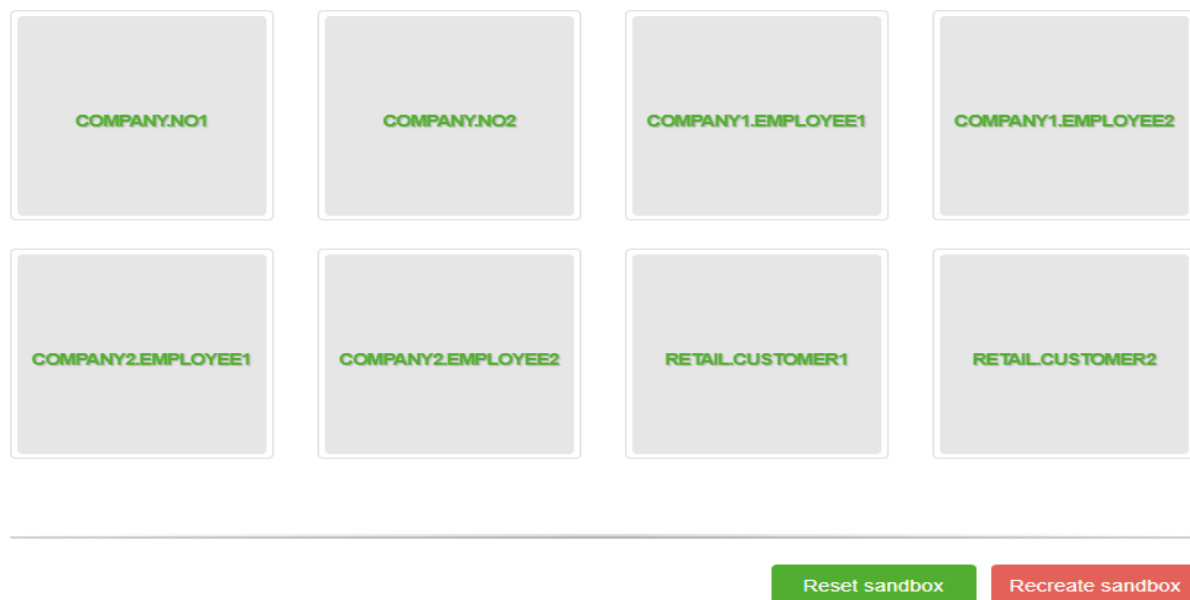
Figure 7 Endpoints related to roles

## 2.6. Testing users

Developer's portal provides access to testing data set. It could be found via *Sandbox users* link from the left side menu. There are eight PSU testing users created for test purposes (Figure 8). Each user has individual personal data sets, like name, email, username attached to PSU login (Figure 9).

### Sandbox users

Manage multiple TPP Sandbox users

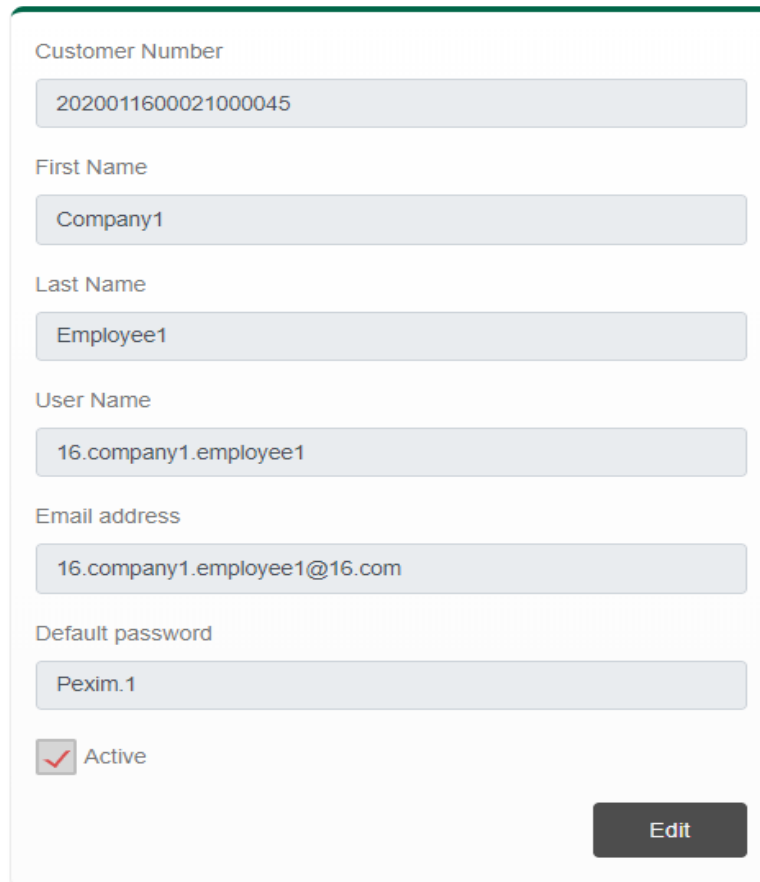


COMPANY.NO1	COMPANY.NO2	COMPANY1.EMPLOYEE1	COMPANY1.EMPLOYEE2
COMPANY2.EMPLOYEE1	COMPANY2.EMPLOYEE2	RETAIL.CUSTOMER1	RETAIL.CUSTOMER2

Reset sandbox Recreate sandbox

Figure 8 Sandbox users list

## Sandbox user



Customer Number

2020011600021000045

First Name

Company1

Last Name

Employee1

User Name

16.company1.employee1

Email address

16.company1.employee1@16.com

Default password

Pexim.1

☒ Active

Edit

Figure 9 Sandbox user detail form

## 4. XS2A interface

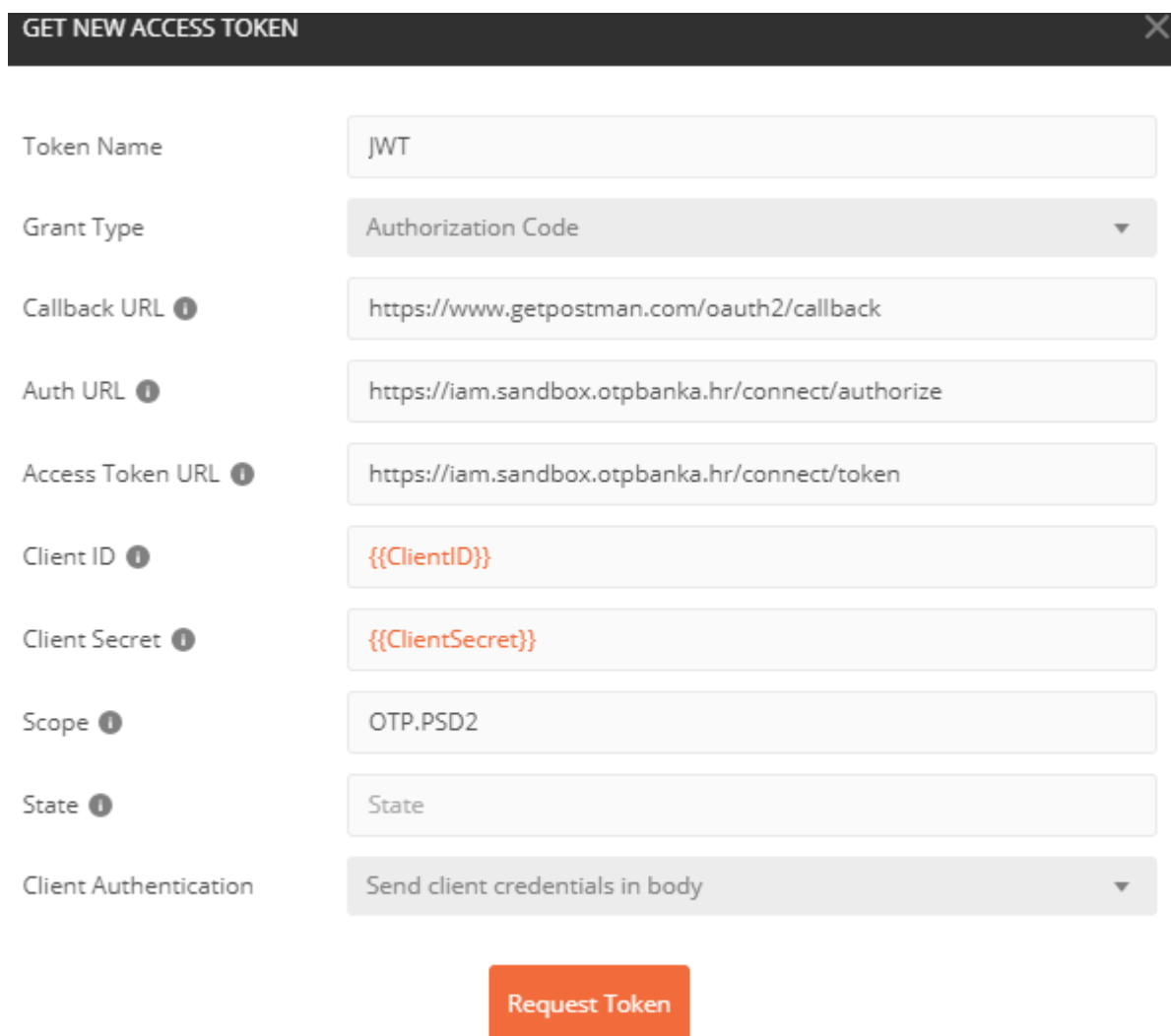
### 4.1. Pre-Authentication

We will follow the pre-authentication approach of Berlin Group. It requires an authentication of a PSU in a pre-step, translating this authentication into an access token. This corresponds to the regular behaviour of OTP online banking. Access token is mandatory for any other API call as described in BGS (4.3 Optional Usage of OAuth2 for PSU Authentication or Authorisation). Please note that supported scope for pre-step authentication is **OTP.PSD2** in a Sandbox environment and in a production environment.

After the login the TPP can request AIS, PIS or PIIS services. The information about authorization server can be accessed via <https://iam.sandbox.otpbanka.hr/.well-known/openid-configuration> link.

If client wants to get access to PSD2 API it should pass *Authorization* HTTP header parameter in every request. *Authorization* header contains bearer token issued by the Oauth server. Before accessing Oauth server client has to register client application following steps in section 2.4. After registering application

client will get *clientId* and *clientSecret* params (Figure 5). These params should be passed to the OAuth server's /authorization and /token endpoints. For example, this could be done using Postman (Figure 10).



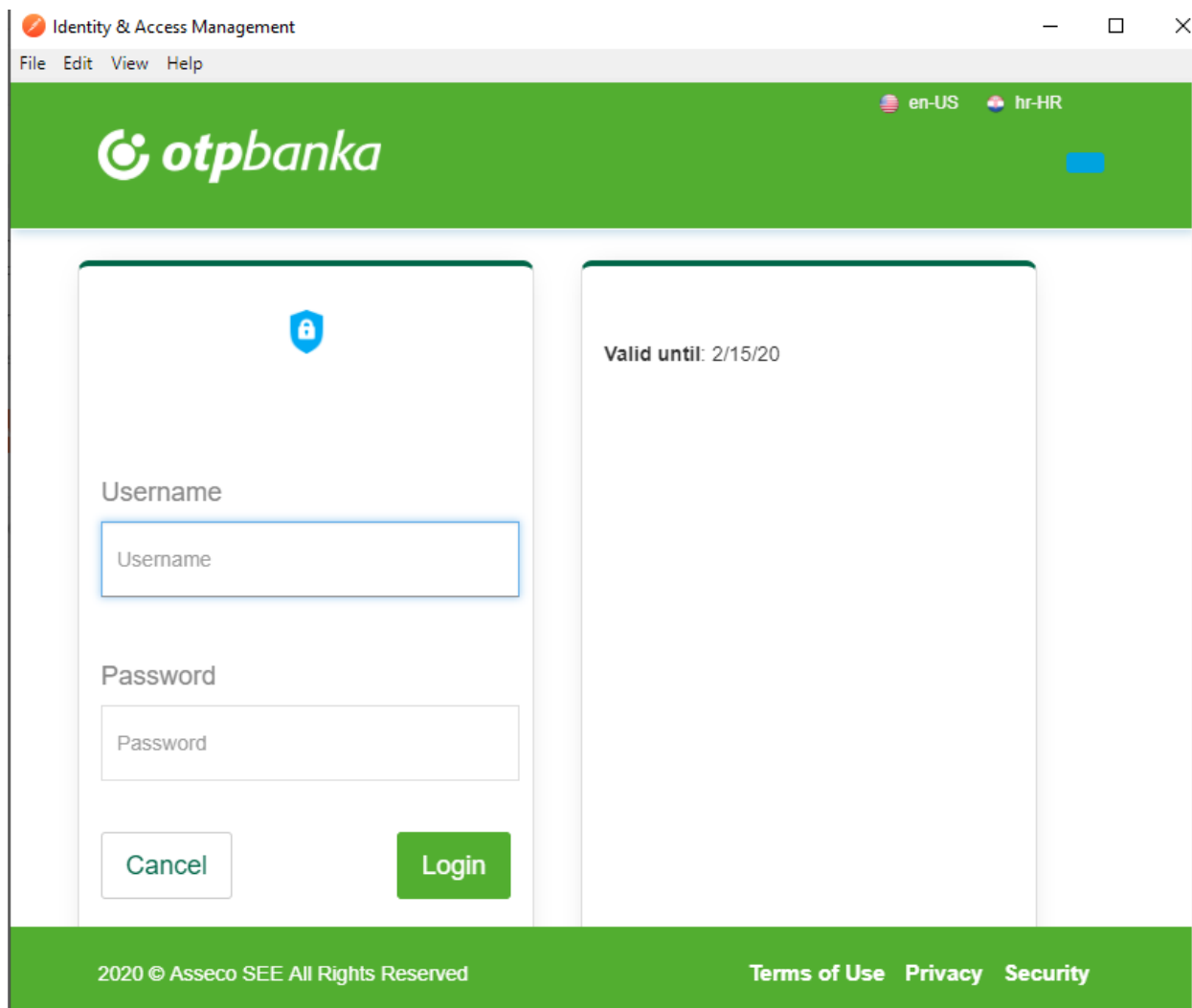
The image shows a Postman interface for an OAuth 2.0 authorization request. The title bar reads "GET NEW ACCESS TOKEN" with a close button. The form contains the following fields and values:

Field	Value
Token Name	JWT
Grant Type	Authorization Code
Callback URL ⓘ	https://www.getpostman.com/oauth2/callback
Auth URL ⓘ	https://iam.sandbox.otpbanka.hr/connect/authorize
Access Token URL ⓘ	https://iam.sandbox.otpbanka.hr/connect/token
Client ID ⓘ	{{ClientID}}
Client Secret ⓘ	{{ClientSecret}}
Scope ⓘ	OTP.PSD2
State ⓘ	State
Client Authentication	Send client credentials in body

At the bottom of the form is an orange button labeled "Request Token".

Figure 10 Postman OAuth server authorization request form

After you have filled in the parameters request the new token. The flow will take you to the IAM login page where you will have to login on behalf of one of the Sandbox users (Figure 9). You can choose any Sandbox user and input the username and password



Identity & Access Management

File Edit View Help

en-US hr-HR

otpbanka

Valid until: 2/15/20

Username

Username

Password

Password

Cancel Login

2020 © Asseco SEE All Rights Reserved

[Terms of Use](#) [Privacy](#) [Security](#)

Figure 11 IAM Login page

TPP has to pass the generated QWAC certificate in the mTLS transport layer. All information about TPP roles and authorization number are parsed from this certificate. If the information in the certificate is missing or is not valid then a TPP client will get an API error. For example, this could be done using Postman (Figure 11).

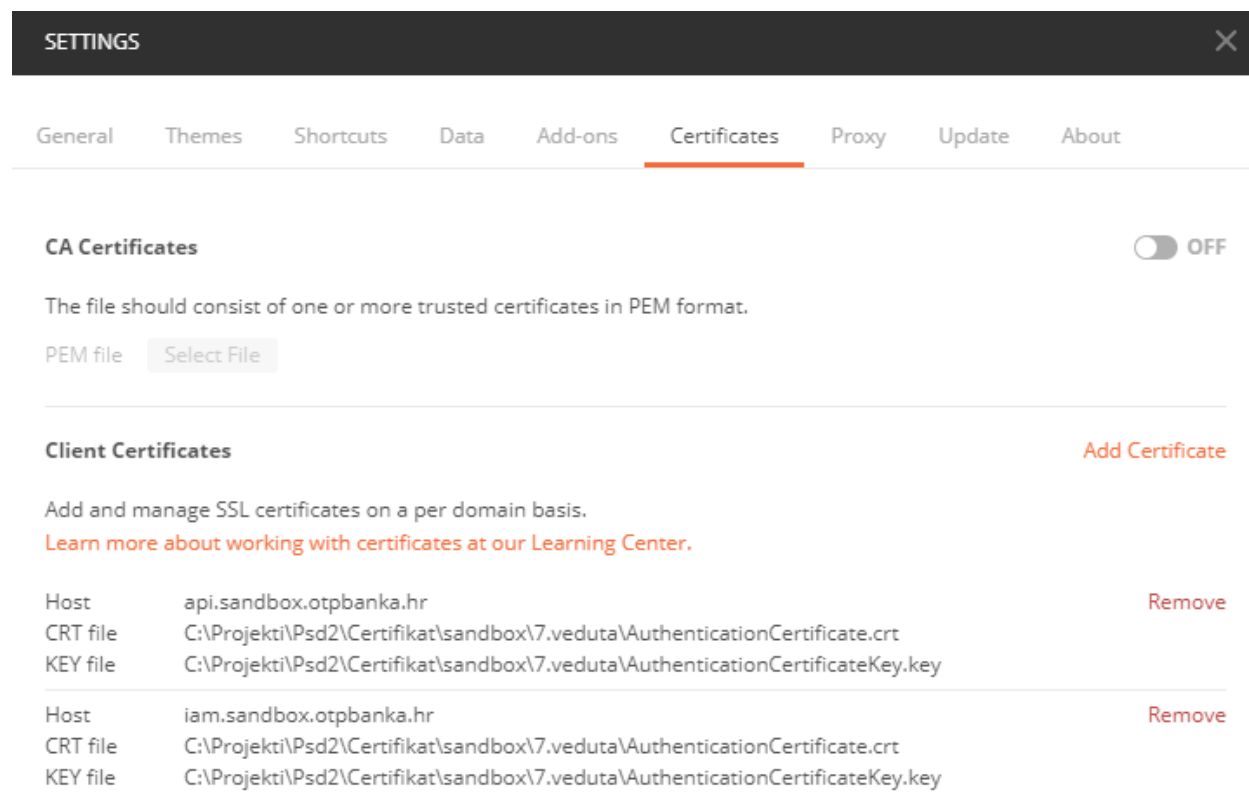


Figure 12 Postman add certificates

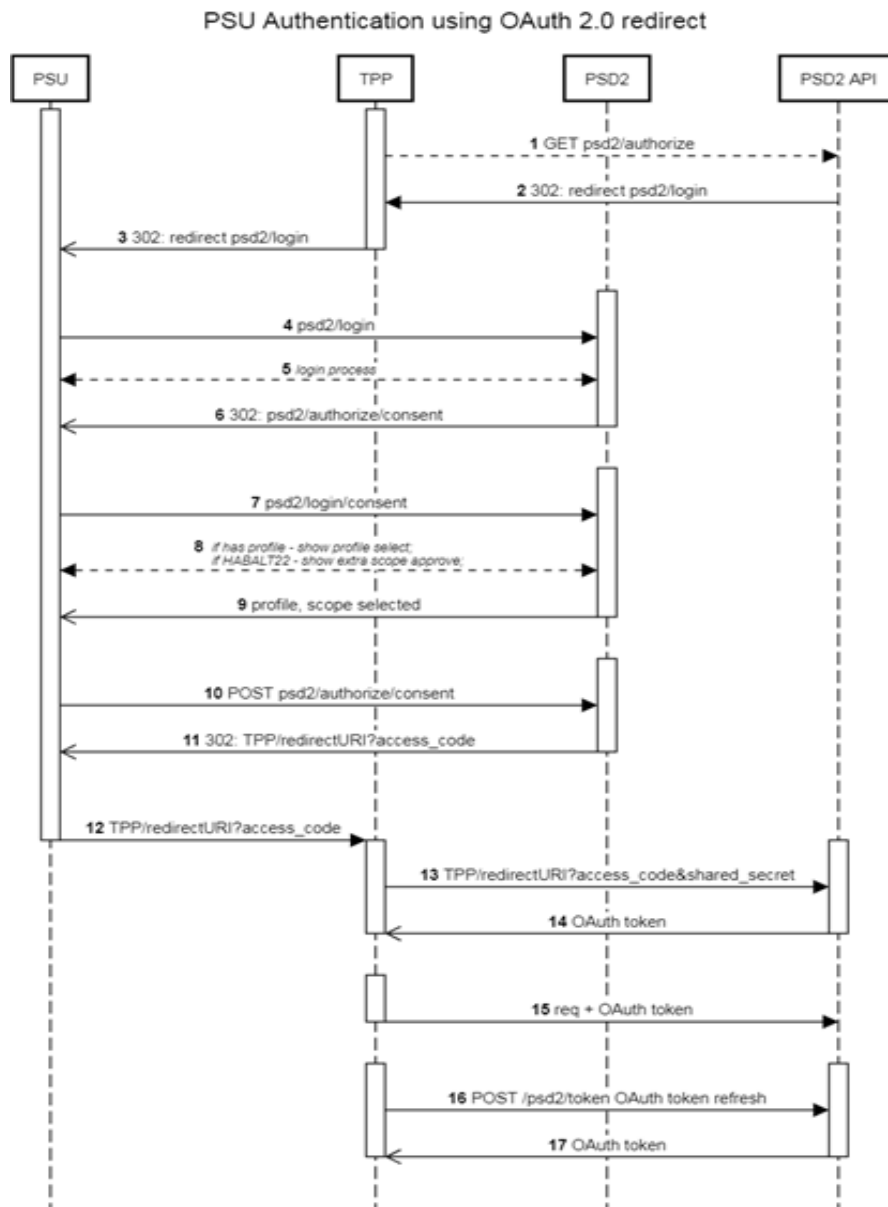


Figure 13 PSU Authentication using OAuth 2.0 redirect

## 4.2. Getting OAuth token

OAuth2 tokens are issued requesting authorization server's special endpoints. Server could be accessed via <https://iam.sandbox.otpbanka.hr> URL. Basically, there are two endpoints which participate in the OAuth2 flow process. The first one is responsible for the client authorization and the second one is responsible for the token issuing.



**Authorization endpoint GET /connect/authorize**

response_type	mandatory	„code“ is only supported as response type
grant_type	mandatory	„code“ is only supported as grant type
client_id	mandatory	Generated application clientId from developer's portal
scope	mandatory	Scope should be „OTP.PSD2“
state	mandatory	A dynamical value set by the TPP and used to prevent XSRF attacks.
redirect_uri	mandatory	the URI of the TPP where the OAuth2server is redirecting the PSU's user agent after the authorization.

**CURL authorization call:**

```
curl --location --request GET
'https://iam.sandbox.otpbanka.hr/connect/authorize?client_id=<client_id>&redirect_uri=
<redirect_uri>&response_type=code&grant_type=code&scope=openid&state=<state>'
```

Executing this call will generate 302 response with Location header, redirect the client app to the login form where the user has to authorize himself by entering “username” and “default password” from one of the sandbox users from the developer's portal (Figure 9.).

After successful user authorization client app will be redirected to the *redirect\_uri* parameter with following parameters in string format.

code	one time code that will be used for obtaining access token by TPP
state	A dynamical value set by the TPP and used to prevent XSRF attacks.
scope	scopes that were granted
session_state	this field can be omitted

After this step TPP can request token by calling token issuing endpoint.

**Token endpoint POST /connect/token**

code	mandatory	code that was received in callback
client_id	mandatory	Generated application clientId from developer's portal
client_secret	mandatory	Generated application client_secret from developer's portal
scope	mandatory	This field should be equal to the scope parameter received in callback
grant_type	mandatory	This field needs to be equal to „authorization_code“
redirect_uri	mandatory	redirect URI that was used I /connect/authorize request

**CURL token call:**

```
curl --location --request POST 'https://iam.sandbox.otpbanka.hr/connect/token' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'client_id=<client_id>' \
```

```
--data-urlencode 'client_secret=<client_secret>' \
--data-urlencode 'scope=<scope>' \
--data-urlencode 'grant_type=authorization_code' \
--data-urlencode 'code=<code>' \
--data-urlencode 'redirect_uri=<redirect_uri>'
```

### Token response example

```
{
  "access_token":
    "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQsw5c",
  "expires_in": 3600,
  "token_type": "Bearer"
}
```

## 4.3. Possible error codes with Pre-Authentication

HTTP Code	Error (Enumeration)	error_description
400	unauthorized	ClientID or ClientSecret wrong
400	unsupported_grant_type	Unsupported grant type
400	invalid_client	Given client ID does not match authenticated client

## 5. Strong Customer Authentication (SCA)

By PSD2 directive some API calls requires Strong Customer Authentication (PSU must approve request by using PIN2). SCA can be implemented either by using redirect or decoupled methods.

In redirect mode PSD2 API generates links and TPP must redirect PSU to these pages. In these pages corresponding payment/consent/... information will be displayed to PSU and PSU will authorise using security device and PIN2.

In decoupled integration mode SCA is performed without displaying bank pages. In decoupled mode PSD2 API generates SCA requests to third party identity provider, PSU gets details of requested authorisation action in his security device and approve it using PIN2.

### 5.1. SCA Exemptions

Whitelisting, low-value transaction are also not supported in the first approach of the XS2A interface.

### 5.2. Redirect approach

In redirect integration method TPP needs to provide redirects URLs in the TPP-Redirect-URI header in case when SCA is required. Redirects happen after the PSU has completed the SCA process in OTP PSD2 API pages and have to be redirected to TPP. Please note that TPP-Redirect-URI header is required for all SCA requests in redirect integration method.

An explanation of the steps:

1. If request requires PSU SCA and TPP-explicit-authorisation-preferred is omitted or false in response \_links object scaRedirect link is returned.
2. [Optional] If TPP-explicit-authorisation-preferred is set to true, explicit authorisation is started. Explicit authorisation allows more detailed control of authorisation process needed for decoupled integration method or countersigning. In such case response will have steering link in startAuthorisation parameter and TPP must request it. This will start authorisation and return scaRedirect link.
3. Redirect the PSU using scaRedirect to OTP bank environment, where PSU completes SCA flow.
4. Check SCA status until value 'finalised' or 'failed' is received. If authorisation has failed, new authorisation may be created and processed.
5. After successfully completing the SCA flow the PSU will be redirected to the URL provided earlier in the TPP-Redirect-URI with query parameter *confirmationCode*.
6. TPP continues usual flow. Please note that authorisation resource will be created automatically (unless specified otherwise over TPP-Explicit-Authorisation-Preferred header) by the bank after the submission of the endpoint POST method. Explicit authorisation should only be used when decoupled approach is selected or when countersigning of the payment is required.

## 6. Accounts endpoints

### 6.1. Consent request

In order to read account details, transactions, balances or initiate payments, TPP needs to get consent from user. An AI role is needed for accessing this endpoint. First step in doing this is creation of consent resource. For creating a consent, a SCA will always be necessary.

A consent can become invalid, if:

- the PSU, TPP or ASPSP (OTP) revokes the consent.
- the consent was created for a specific period of time (validUntil).

#### Request POST /v1/consents/

##### Request header [pre-step auth]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	mandatory	Oauth2 authorization bearer token
<b>TPP-Redirect-Preferred</b>	optional	If it equals "true", the TPP prefers a redirect over an embedded SCA approach.
<b>TPP-Redirect-URI</b>	conditional	Mandated for the Redirect SCA Approach (including OAuth2 SCA approach), specifically when TPP-Redirect-Preferred equals "true"
<b>Content-Type</b>	mandatory	Content type application/json

## Request header [PSU-ID flow retail]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Tpp-redirect-preferred</b>	optional	If it equals "true", the TPP prefers a redirect over an embedded SCA approach.
<b>TPP-Redirect-URI</b>	conditional	Mandated for the Redirect SCA Approach (including OAuth2 SCA approach), specifically when TPP-Redirect-Preferred equals "true"
<b>Content-Type</b>	mandatory	Content type application/json

## Request header [PSU-ID flow CORPORATE]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	Username for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Tpp-redirect-preferred</b>	optional	If it equals "true", the TPP prefers a redirect over an embedded SCA approach.
<b>TPP-Redirect-URI</b>	conditional	Mandated for the Redirect SCA Approach (including OAuth2 SCA approach), specifically when TPP-Redirect-Preferred equals "true"
<b>Content-Type</b>	mandatory	Content type application/json

### Sandbox user

Customer Number

2023012500161001164

First Name

Company

Last Name

No1

User Name

55.company.no1

Email address

55.company.no1@55.com

Default password

Pexim.1

☒ Active

Edit

## Request body

<b>access</b>	<b>mandatory</b>	<b>Requested access services</b>
---------------	------------------	----------------------------------

<b>recurringIndicator</b>	mandatory	true, if the consent is for recurring access to the account data. false, if the consent is for one access to the account data
<b>validUntil</b>	mandatory	This parameter is requesting a valid until date for the requested consent.
<b>frequencyPerDay</b>	mandatory	This field indicates the requested maximum frequency for an access without PSU involvement per day.
<b>combinedServiceIndicator</b>	mandatory	If true indicates that a payment initiation service will be addressed in the same "session",

## Request example

```
{
  "access": {
    "availableAccounts": "allAccounts"
  },
  "recurringIndicator": "false",
  "validUntil": "2019-12-30T10:02:29.073Z",
  "frequencyPerDay": "30",
  "combinedServiceIndicator": "false"
}
```

## Response POST /v1/consents/

### Response code

<b>201 Created</b>	The request has been fulfilled and has resulted in one or more new resources being created
--------------------	--

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Aspsp-Sca-Approach</b>	Possible values are: REDIRECT or DECOUPLED
<b>Content-Type</b>	Content type application/json

### Response example

```

{
  "consentStatus": "received",
  "consentId": "58f686ea-cd47-4501-92e1-eae26398bbee",
  "scaMethods": [],
  "chosenScaMethod": null,
  "challengeData": {
    "data": "Default challenge"
  },
  "_links": {
    "scaRedirect": {
      "href": ""
    },
    "self": {
      "href": "v1/consents/58f686ea-cd47-4501-92e1-eae26398bbee"
    },
    "status": {
      "href": "v1/consents/58f686ea-cd47-4501-92e1-eae26398bbee/status"
    },
    "scaStatus": {
      "href": "v1/consents/58f686ea-cd47-4501-92e1-eae26398bbee/authorisations/bff84163af3e43b98e6c3d8ea49b1326"
    }
  }
}

```

## 6.2. Get consent request

Returns the content of an account information consent object.

### Request GET /v1/consents/{consentId}

#### Path parameter

<b>consentId</b>	The consent identification assigned to the created resource
------------------	---

#### Request header [pre-step auth]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	Mandatory	Oauth2 authorization bearer token
<b>Content-Type</b>	mandatory	Content type application/json

#### Request header [PSU-ID flow retail]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

#### Request header [PSU-ID flow corporate]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

## Response GET /v1/consents/{consentId}

### Response code

200 Ok	The request has succeeded
--------	---------------------------

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

### Response example

```
{
  "access": {
    "accounts": [],
    "balances": [],
    "transactions": [],
    "availableAccounts": "allAccounts"
  },
  "recurringIndicator": true,
  "validUntil": "2020-01-30T10:02:29.073",
  "frequencyPerDay": 30,
  "lastActionDate": "2020-01-17T12:53:39.6005658",
  "consentStatus": "received"
}
```

## 6.3. Get consent status request

Returns the content of an account information consent object.

### Request GET /v1/consents/{consentId} /status

#### Path parameter

<b>consentId</b>	The consent identification assigned to the created resource
------------------	---

#### Request header [pre-step auth]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	Mandatory	Oauth2 authorization bearer token
<b>Content-Type</b>	mandatory	Content type application/json

## Request header [PSU-ID flow retail]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

## Request header [PSU-ID flow corporate]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

## Response GET /v1/consents/{consentId}/status

### Response code

<b>200 Ok</b>	The request has succeeded
---------------	---------------------------

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

### Response example

```
{
  "consentStatus": "received"
}
```

## 6.4. Delete consent

Delete content.



**Request DELETE /v1/consents/{consentId}****Path parameter**

<b>consentId</b>	The consent identification assigned to the created resource
------------------	---

**Request header**

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	Mandatory	Oauth2 authorization bearer token
<b>Content-Type</b>	mandatory	Content type application/json

**Request header [PSU-ID flow retail]**

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

**Request header [PSU-ID flow corporate]**

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

**Response DELETE /v1/consents/{consentId}****Response code**

<b>204 No content</b>	The request has succeeded
-----------------------	---------------------------

**Response header**

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

## 6.5. Read account list

Reads a list of bank accounts, with balances where required. It is assumed that a consent of the PSU to this access is already given and stored on the ASPSP system.

### Request GET /v1/accounts

#### Query parameter

<b>withBalance</b>	If contained, this function reads the list of accessible payment accounts including the booking balance, if granted by the PSU in the related consent and available by the ASPSP.
--------------------	---

#### Request header [pre-step auth]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Content-Type</b>	mandatory	Content type application/json
<b>Consent-ID</b>	mandatory	Identification of the consent for this access as granted by the PSU.

### Response GET /v1/accounts

#### Response code

<b>200 OK</b>	The request has succeeded
---------------	---------------------------

#### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

#### Response example

```
{
  "accounts": [
    {
      "resourceId": "2000011300",
      "iban": "HR7524070002000011300",
      "bban": "24070002000011300",
      "msisdn": "+3562596000",
      "currency": "HRK",
      "name": "My Corporate transactional account multicurrency",
      "product": "Corporate transactional account multicurrency",
      "cashAccountType": "CACC",
      "status": "enabled",
      "bic": "OTPVHR24",
      "usage": "ORGA",
      "details": "Corporate transactional account multicurrency",
      "_links": {
        "account": {
          "href": "/v1/accounts/2000011300"
        }
      }
    },
    {
      "resourceId": "0000002120",
      "iban": "HR93240700000000002120",
      "bban": "240700000000002120",
      "msisdn": "+3562514620",
      "currency": "EUR",
      "name": "ibanPaymentFailedWithScaSuccessfulMulti",
      "product": "Retail transactional account in EUR - STATELESS",
      "cashAccountType": "CACC",
      "status": "enabled",
      "bic": "OTPVHR24",
      "usage": "PRIV",
      "details": "Retail transactional account in EUR - STATELESS",
      "_links": {
        "account": {
          "href": "/v1/accounts/0000002120"
        }
      }
    }
  ]
}
```

## 6.6. Read account details

Reads details about an account, with balances where required. It is assumed that a consent of the PSU to this access is already given and stored on the ASPSP system. The addressed details of this account depends then on the stored consent addressed by consentId, respectively the OAuth2 access token.

**Request GET /v1/accounts/{account-id}**

**Path parameter**

<b>accountId</b>	The account identification assigned to the created resource
------------------	---

## Query parameter

<b>withBalance</b>	If contained, this function reads the list of accessible payment accounts including the booking balance, if granted by the PSU in the related consent and available by the ASPSP.
--------------------	---

## Request header

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Consent-ID</b>	mandatory	Identification of the consent for this access as granted by the PSU.
<b>Content-Type</b>	mandatory	Content type application/json

## Response GET /v1/accounts/{account-id}

### Response code

<b>200 Ok</b>	The request has succeeded
---------------	---------------------------

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

### Response example

```
[
  {
    "account": {
      "resourceId": "2000011300",
      "iban": "HR7524070002000011300",
      "bban": "24070002000011300",
      "msisdn": "+3562596000",
      "currency": "HRK",
      "name": "My Corporate transactional account multicurrency",
      "product": "Corporate transactional account multicurrency",
      "cashAccountType": "CACC",
      "status": "enabled",
      "bic": "OTPVHR24",
      "usage": "ORGA",
      "details": "Corporate transactional account multicurrency",
      "_links": {
        "account": {
          "href": "/v1/accounts/2000011300"
        }
      }
    }
  }
]
```

## 6.7. Get balances

Reads account data from a given account addressed by "account-id".

### Request GET /v1/accounts/{account-id} /balances

#### Path parameter

<b>accountId</b>	The account identification assigned to the created resource
------------------	---

#### Request header

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Consent-ID</b>	mandatory	Identification of the consent for this access as granted by the PSU.
<b>Content-Type</b>	mandatory	Content type application/json

### Response GET /v1/accounts/{account-id}/balances

#### Response code

<b>200 Ok</b>	The request has succeeded
---------------	---------------------------

#### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

#### Response example

```
{
  "account": {
    "resourceId": "2000011300",
    "iban": "HR7524070002000011300",
    "bban": "24070002000011300",
    "msisdn": "+3562596000",
    "currency": "HRK",
    "name": "My Corporate transactional account multicurrency",
    "product": "Corporate transactional account multicurrency",
    "cashAccountType": "CACC",
    "status": "enabled",
    "bic": "OTPVHR24",
    "usage": "ORGA",
    "details": "Corporate transactional account multicurrency",
    "_links": {
      "account": {
        "href": "/v1/accounts/2000011300"
      }
    }
  }
}
```

## 6.8. Get transactions list

Reads account data from a given account addressed by "account-id".

**Request GET /v1/accounts/{account-id} /transactions/ {query-parameters}**

### Path parameter

<b>accountId</b>	The account identification assigned to the created resource
------------------	---

### Query parameter

<b>dateFrom</b>	Starting date (inclusive the date dateFrom) of the transaction list, mandated if no delta access is required.
<b>dateTo</b>	End date (inclusive the data dateTo) of the transaction list, default is "now" if not given.
<b>bookingStatus</b>	Permitted codes are „booked“, „pending“ and „both“
<b>withBalance</b>	If contained, this function reads the list of transactions including the booking balance, if granted by the PSU in the related consent and available by the ASPSP.

### Request header

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Consent-ID</b>	mandatory	Identification of the consent for this access as granted by the PSU.
<b>Content-Type</b>	mandatory	Content type application/json

**Response GET /v1/accounts/{account-id} /transactions/ {query-parameters}**

## Response code

200 Ok	The request has succeeded
--------	---------------------------

## Response header

X-Request-ID	ID of the request, unique to the call, as determined by the initiating party
Content-Type	Content type application/json

## Response example

```
{
  "account": {
    "iban": "HR902407000xxxxxxxxxxx"
  },
  "transactions": {
    "booked": [
      {
        "transactionId": "1160555679",
        "bookingDate": "2018-12-31T00:00:00",
        "valueDate": "2018-12-31T00:00:00",
        "transactionAmount": {
          "amount": "21712.75",
          "currency": "HRK"
        },
        "exchangeRate": [],
        "remittanceInformationUnstructured": "xxxxxxxxxxxxx"
      },
      {
        "transactionId": "1161265230",
        "bookingDate": "2018-12-31T00:00:00",
        "valueDate": "2018-12-31T00:00:00",
        "transactionAmount": {
          "amount": "-22.51",
          "currency": "HRK"
        },
        "exchangeRate": [],
        "remittanceInformationUnstructured": "POS kupovina-
```

## 6.9. Get transactions details

Reads transaction details from a given transaction addressed by "transactionId" on a given account addressed by "account-id".

**Request GET /v1/accounts/{account-id}/transactions/{transactionId}**

### Path parameter

accountId	The account identification assigned to the created resource
transactionId	This identification is given by the attribute resourceId of the corresponding entry of a transaction list.

--	--

## Request header

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Consent-ID</b>	mandatory	Identification of the consent for this access as granted by the PSU.
<b>Content-Type</b>	mandatory	Content type application/json

## Response GET /v1/accounts/{account-id} /transactions/ {query-parameters}

### Response code

<b>200 Ok</b>	The request has succeeded
---------------	---------------------------

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

### Response example

```
{
  "transactionsDetails": {
    "transactionId": "1145677334",
    "bookingDate": "2018-12-19T00:00:00",
    "valueDate": "2018-12-19T00:00:00",
    "transactionAmount": {
      "amount": "-2",
      "currency": "HRK"
    },
    "exchangeRate": [],
    "remittanceInformationUnstructured": "Naknada"
  }
}
```

## 6.10. Start the authorization process for a consent

### Request POST /v1/consents/{consent-id}/authorisations

#### Path parameter

<b>consentId</b>	The consent identification assigned to the created resource
------------------	---



## Request header

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Consent-ID</b>	mandatory	Identification of the consent for this access as granted by the PSU.
<b>Content-Type</b>	mandatory	Content type application/json

## Response POST /v1/consents/{consent-id}/authorisations

### Response code

<b>201 Created</b>	The request has been fulfilled and has resulted in one or more new resources being created
--------------------	--

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Aspsp-Sca-Approach</b>	Possible values are: REDIRECT or DECOUPLED
<b>Content-Type</b>	Content type application/json

### Response example (TPP-Redirect-Preferred = false)

```
{
  "consentStatus": "received",
  "consentId": "217a5bd3-52cf-4580-b9df-cc39c3c965be",
  "scaMethods": [
    {
      "authenticationVersion": "1.00",
      "authenticationMethodId": "SCA Method 3",
      "name": "SCA Method 3"
    }
  ],
  "chosenScaMethod": {
    "authenticationVersion": "1.00",
    "authenticationMethodId": "SCA Method 3",
    "name": "SCA Method 3"
  },
  "challengeData": {
    "data": "Default challenge"
  },
  "_links": {
    "startAuthorisation": {
      "href": "v1/consents/217a5bd3-52cf-4580-b9df-cc39c3c965be/authorisations"
    },
    "self": {
      "href": "v1/consents/217a5bd3-52cf-4580-b9df-cc39c3c965be"
    },
    "status": {
      "href": "v1/consents/217a5bd3-52cf-4580-b9df-cc39c3c965be/status"
    },
    "scaStatus": {
      "href": "v1/consents/217a5bd3-52cf-4580-b9df-cc39c3c965be/authorisations/"
    }
  }
}
```

**Response example (TPP-Redirect-Preferred = true, TPP-Redirect-URI=http://....)**

```

{
  "scaStatus": "received",
  "authorizationId": "5d829ecb985c442ebc418af7c1577430",
  "scaMethods": [
    {
      "authenticationVersion": "1.00",
      "authenticationMethodId": "SCA Method 3",
      "name": "SCA Method 3"
    }
  ],
  "chosenScaMethod": null,
  "_links": {
    "scaRedirect": {
      "href": "https://iam.test.otpbanka.hr/consent/login?consentId=8a22b4d5-b651-4c52-9af5-6bece4166e7e&authorizationId=5d829ecb985c442ebc418af7c1577430&returnUrl=https://developers."
    },
    "scaStatus": {
      "href": "v1/consents/8a22b4d5-b651-4c52-9af5-6bece4166e7e/authorisations/5d829ecb985c442ebc418af7c1577430"
    },
    "confirmation": {
      "href": "v1/consents/8a22b4d5-b651-4c52-9af5-6bece4166e7e/authorisations/5d829ecb985c442ebc418af7c1577430"
    }
  }
}

```

## 6.11. Consent authorisation using Strong Customer Authentication (SCA)

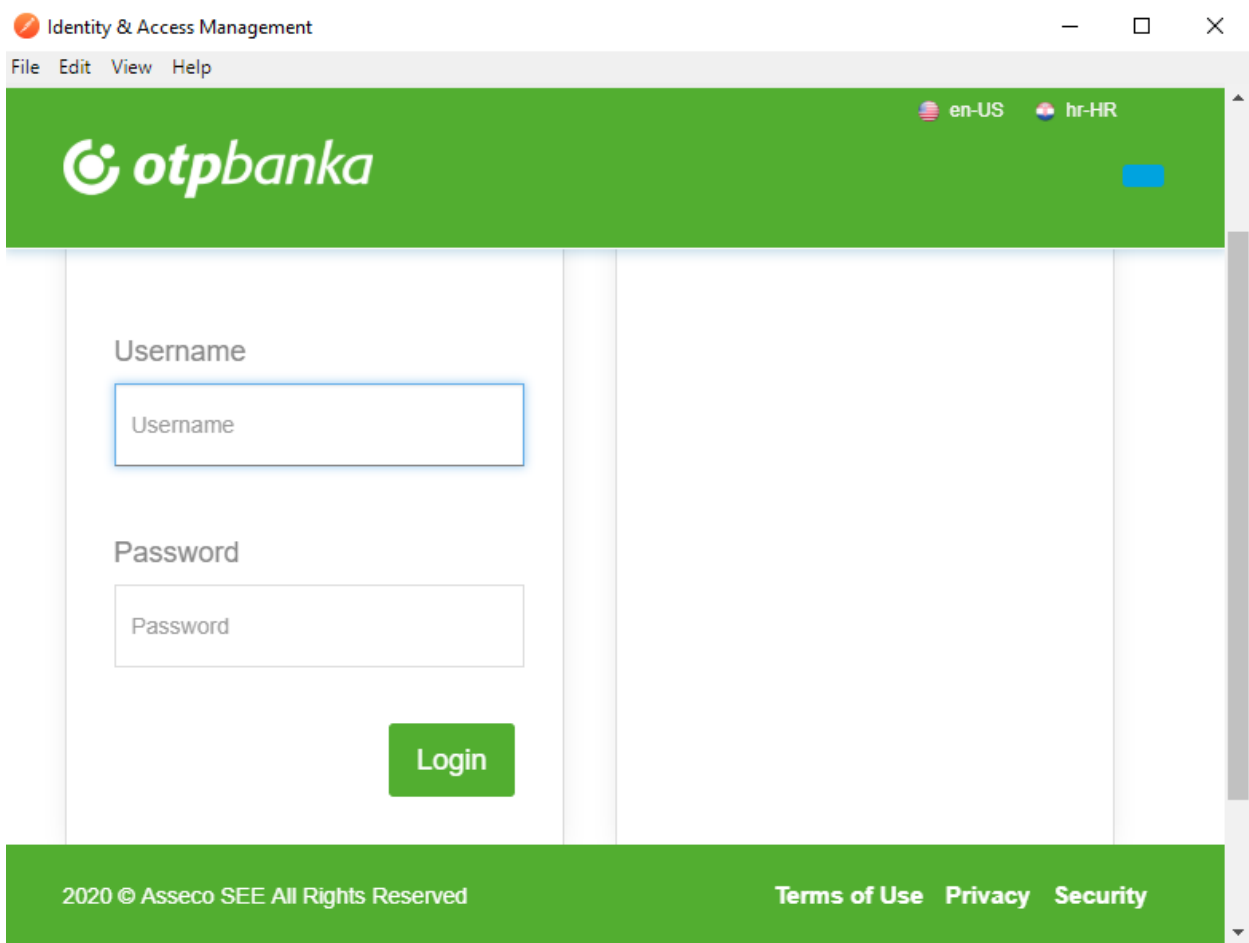
Account information can only be requested after a consent has been created. The PreAuth is not sufficient to authorize a consent.

### 6.11.1. Consent authorisations: redirect SCA approach

During this approach TPP has to send Tpp-Redirect-Preferred header set to true. This means that consent will be authorized in redirect approach. Also, there are two ways how consent authorization object will be created in redirect manner: implicit and explicit.

Implicit method will create authorization object during create consent call. No sequential calls are needed. A *scaRedirect* steering link will be added to the create consent JSON response. Following this redirect link a PSU will be redirect to the OTP bank login form (Figure 11). Please note, for successfully login enter **username** and **default password** from sandbox user form (Figure 9) After successful login consent summary and approval form will be displayed where PSU has to approval credentials. Also, Aspsp-Sca-Approach: REDIRECT header will be added to the response.

Using explicit method TPP will have to make additional call for consent authorization object creation. In such case response will have steering link in *startAuthorisation* parameter and TPP must request it. This will start the authorisation process and return *scaRedirect* steering link inside JSON response. Same as in implicit method following this redirect link will redirect PSU to the OTP bank consent summary and SCA selection, approval form. It's highly recommended to use implicit method with SCA redirect approach. Default method in OTP PSD2 API is explicit.



The screenshot shows a web browser window titled "Identity & Access Management". The browser's address bar and menu bar are visible. The page has a green header with the "otpbanka" logo on the left and language selection options "en-US" and "hr-HR" on the right. The main content area is white and contains a login form. The form has two input fields: "Username" and "Password", each with a placeholder text of the same name. Below the password field is a green "Login" button. The footer is a green bar with white text. On the left, it says "2020 © Asseco SEE All Rights Reserved". On the right, it has links for "Terms of Use", "Privacy", and "Security".

Identity & Access Management

File Edit View Help

en-US hr-HR

otpbanka

Username

Username

Password

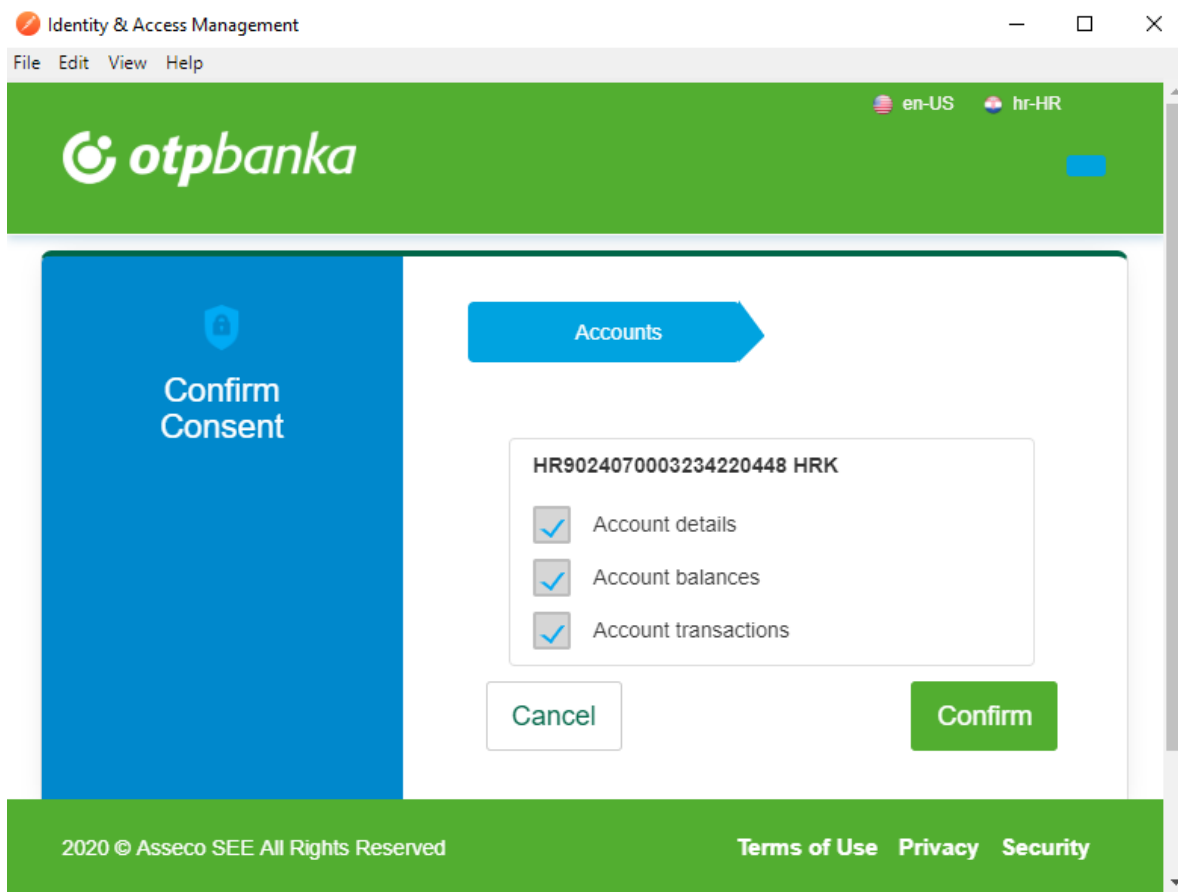
Password

Login

2020 © Asseco SEE All Rights Reserved

Terms of Use Privacy Security

Figure 14 OTP bank login form



Identity & Access Management

File Edit View Help

en-US hr-HR

**otpbanka**

**Confirm Consent**

Accounts

HR9024070003234220448 HRK

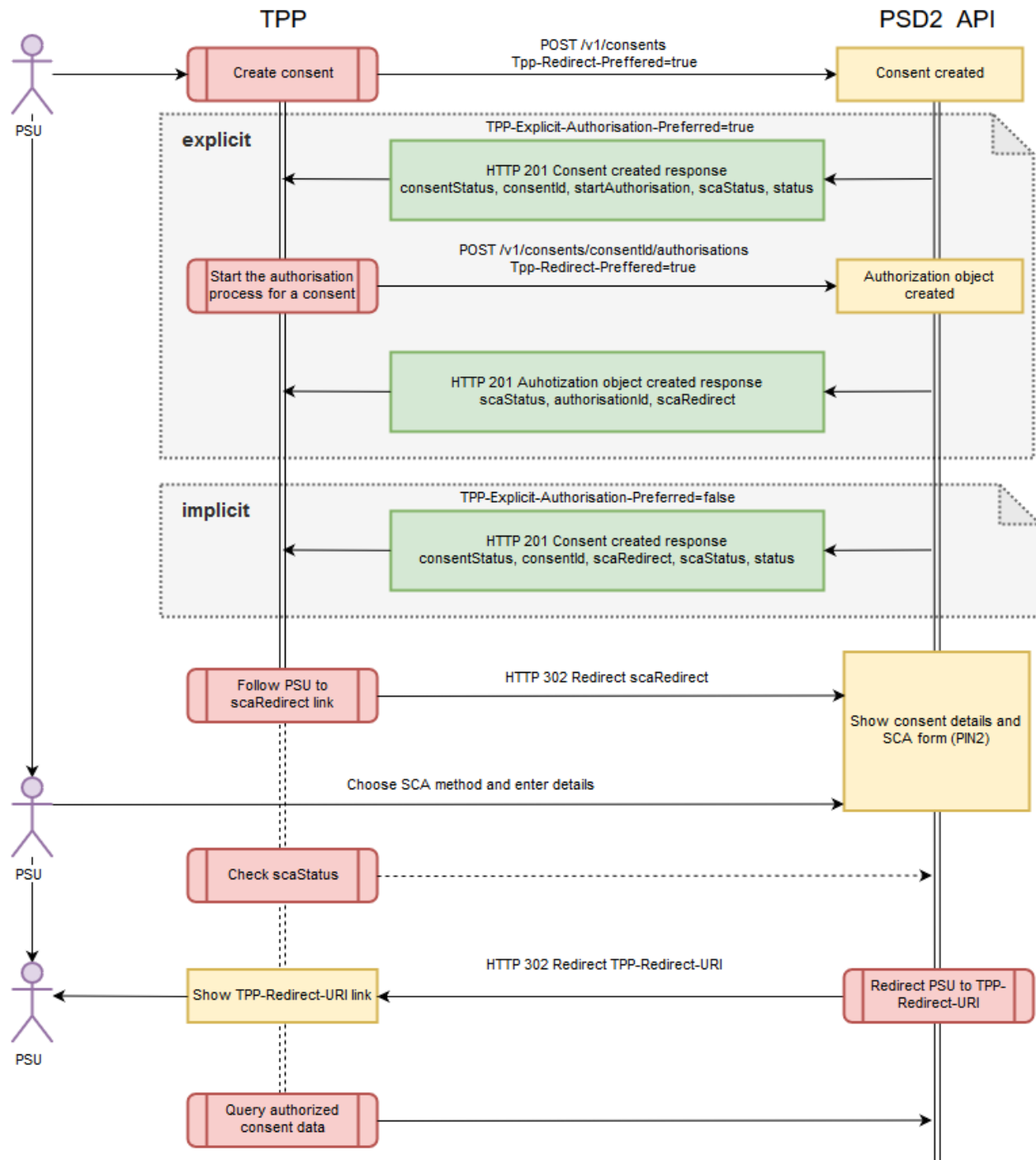
- ☒ Account details
- ☒ Account balances
- ☒ Account transactions

Cancel Confirm

2020 © Asseco SEE All Rights Reserved Terms of Use Privacy Security

Figure 15 Confirm Consent form

## Create consent redirect approach



## 7. Payments endpoints

### 7.1. Payments initiation

#### Request POST /v1/payments/{payment-product}

##### Path parameter

<b>payment-product</b>	The addressed payment product endpoint, e.g. for SEPA Credit Transfers (SCT). The supported product is: sepa-credit-transfers, domestic-payment, instant-domestic-credit-transfers, cross-border-credit-transfers
------------------------	---

##### Request header [pre-step auth]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	mandatory	Oauth2 authorization bearer token
<b>Content-Type</b>	mandatory	Content type application/json
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field between PSU and TPP. If not available, the TPP shall use the IP Address used by the TPP when submitting this request.

##### Request header [PSU-ID flow retail]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

##### Request header [PSU-ID flow CORPORATE]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	Username for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

##### Request body example for sepa-credit transfer

<b>endToEndIdentification</b>	optional	SEPA end to end reference id field
<b>debtorAccount</b>	mandatory	Debtor account object with iban and currency elements

<b>instructedAmount</b>	mandatory	Instructed payment amount has amount and currency elements.
<b>creditorAccount</b>	mandatory	Creditor account object with iban and currency elements
<b>creditorAgent</b>	optional	
<b>creditorName</b>	mandatory	Title/name of the creditor
<b>creditorAddress</b>	optional	
<b>remittanceInformationUnstructured</b>	optional	

### Request example for sepa-credit transfer

```
{
  "debtorAccount": {
    "iban": "HR7524070002000011300"
  },
  "instructedAmount": {
    "currency": "EUR",
    "amount": "0.01"
  },
  "creditorAccount": {
    "iban": "HR9024070003234220448"
  },
  "creditorName": "Test PSD2 Interface"
}
```

### Response POST /v1/payments/{payment-product}

#### Response code

<b>201 Created</b>	The request has been fulfilled and has resulted in one or more new resources being created
--------------------	--

#### Response header

<b>Location</b>	Location of the created resource (if created)
<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Aspsp-Sca-Approach</b>	Possible values are: REDIRECT or DECOUPLED
<b>Content-Type</b>	Content type application/json

#### Response example

```

{
  "transactionStatus": "RCVD",
  "paymentId": "765b125ba74440dc89a09f46f26a3a50",
  "transactionFees": {
    "amount": "0",
    "currency": "HRK"
  },
  "transactionFeeIndicator": false,
  "scaMethods": [
    {
      "authenticationVersion": "1.00",
      "authenticationMethodId": "SCA Method 3",
      "name": "SCA Method 3"
    }
  ],
  "chosenScaMethod": {
    "authenticationVersion": "1.00",
    "authenticationMethodId": "SCA Method 3",
    "name": "SCA Method 3"
  },
  "challengeData": {
    "data": "Default challenge"
  },
  "_links": {
    "startAuthorisation": {
      "href": "v1/payments/domestic-credit-transfers-hr/765b125ba74440dc89a09f46f26a3a50/authorisations"
    },
    "self": {
      "href": "v1/payments/domestic-credit-transfers-hr/765b125ba74440dc89a09f46f26a3a50"
    },
    "status": {
      "href": "v1/payments/domestic-credit-transfers-hr/765b125ba74440dc89a09f46f26a3a50/status"
    }
  }
}

```

## 7.2. Get payment transaction status

**Request POST /v1/payments/{payment-product}/{payment-id}/status**

**Path parameter**

<b>payment-product</b>	The addressed payment product endpoint, e.g. for SEPA Credit Transfers (SCT). The supported product is: sepa-credit-transfers, domestic-payment, instant-domestic-credit-transfers, cross-border-credit-transfers
<b>payment-id</b>	The consent identification assigned to the created resource

**Request header [pre-auth flow]**

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	Mandatory	Oauth2 authorization bearer token
<b>Content-Type</b>	mandatory	Content type application/json

**Request header [PSU-ID flow retail]**



<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

## Request header [PSU-ID flow CORPORATE]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	Username for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

## Response POST /v1/payments/{payment-product}/{payment-id}/status

### Response code

<b>200 Ok</b>	The request has succeeded
---------------	---------------------------

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

### Response example

```
{
  "transactionStatus": "RCVD"
}
```

## 7.3. Get payment request

### Request GET /v1/payments/{payment-product}/{payment-id}

#### Path parameter

<b>payment-product</b>	The addressed payment product endpoint, e.g. for SEPA Credit Transfers (SCT). The supported product is: sepa-credit-transfers, domestic-payment, instant-domestic-credit-transfers, cross-border-credit-transfers
<b>payment-id</b>	The consent identification assigned to the created resource

## Request header [pre-step auth]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	mandatory	Oauth2 authorization bearer token
<b>Content-Type</b>	mandatory	Content type application/json

## Request header [PSU-ID flow retail]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

## Request header [PSU-ID flow CORPORATE]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	Username for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

## Response POST /v1/payments/{payment-product}/{payment-id}

### Response code

<b>200 Ok</b>	The request has succeeded
---------------	---------------------------

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

### Response example

```
{
  "debtorAccount": {
    "iban": "HR7524070002000011300"
  },
  "instructedAmount": {
    "amount": "0.01",
    "currency": "EUR"
  },
  "creditorAccount": {
    "iban": "HR9024070003234220448"
  },
  "creditorName": "Test PSD2 Interface",
  "transactionStatus": "RCVD"
}
```

#### 7.4. Delete payment request

It initiates the cancellation of a payment. Depending on the payment-service, the payment-product and the ASPSP's implementation, this TPP call might be sufficient to cancel a payment. If an authorisation of the payment cancellation is mandated by the ASPSP, a corresponding hyperlink will be contained in the response message.

**Request DELETE /v1/payments/{payment-product}/{payment-id}**

##### Path parameter

<b>payment-product</b>	The addressed payment product endpoint
<b>payment-id</b>	The consent identification assigned to the created resource

##### Request header [pre-step auth]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	Mandatory	Oauth2 authorization bearer token
<b>Content-Type</b>	mandatory	Content type application/json

##### Request header [PSU-ID flow retail]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user

<b>Content-Type</b>	mandatory	Content type application/json
---------------------	-----------	-------------------------------

#### Request header [PSU-ID flow CORPORATE]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	Username for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

#### Response DELETE /v1/payments/{payment-product}/{payment-id}

##### Response code

<b>204 No content</b>	The request has succeeded
-----------------------	---------------------------

##### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Content-Type</b>	Content type application/json

### 7.5. Update PSU data for payment initiation

#### Request PUT /v1/ payments/{payment-product}/{payment-id}/authorisations/{authorisation-id}

##### Path parameter

<b>payment-product</b>	The addressed payment product endpoint.
<b>payment-id</b>	The consent identification assigned to the created resource
<b>authorisation-id</b>	Authorisation object ID

#### Request header [pre-step auth]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Authorization</b>	mandatory	Oauth2 authorization bearer token
<b>Content-Type</b>	mandatory	Content type application/json

#### Request header [PSU-ID flow retail]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
---------------------	-----------	--

<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	User Name for the Sandbox user

## Request header [PSU-ID flow CORPORATE]

<b>X-Request-ID</b>	mandatory	ID of the request, unique to the call, as determined by the initiating party
<b>PSU-IP-Address</b>	mandatory	The forwarded IP Address header field consists of the corresponding HTTP request IP Address field
<b>Psu-id</b>	mandatory	Username for the Sandbox user
<b>Psu-corporate-id</b>	mandatory	CustomerNumber for the Sandbox user
<b>Content-Type</b>	mandatory	Content type application/json

## Request example

```
{
  "confirmationCode": "S0pIWDVEQ01YRWn6SWkxY2NWeVZMdGEwJjFFPQ"
}
```

## Response PUT /v1/ payments/{payment-product}/{payment-id}/authorisations/{authorisation-id}

### Response code

<b>200 Ok</b>	The request has been fulfilled and has resulted in one or more new resources being created
---------------	--

### Response header

<b>X-Request-ID</b>	ID of the request, unique to the call, as determined by the initiating party
<b>Aspsp-Sca-Approach</b>	Possible values are: REDIRECT or DECOUPLED
<b>Content-Type</b>	Content type application/json

## Response example

```
{
  "ScaStatus": "finalised",
  "ChosenScaMethod": null,
  "ChallengeData": null,
  "Links": null,
  "Pain002Response": null
}
```

### 7.6. Payment authorisation using Strong Customer Authentication (SCA)

To confirm payments, a SCA takes place for each transaction. The PreAuth is not sufficient to authorize a payment. After the authorization of a payment, the API responds if the payment was accepted or declined, similar to the OTP Online Banking. It is not possible to send an automated confirmation that the payment was booked successfully because of the OTP batch-booking approach.

#### 7.6.1. Payment authorisation: redirect SCA approach

During this approach TPP has to send `Tpp-Redirect-Preferred` header set to true. This means that payment will be authorized in redirect approach. Also, there are two ways how payment authorization object will be created in redirect manner: implicit and explicit.

Implicit method will create authorization object during initiate payment call. No sequential calls are needed. A `scaRedirect` steering link will be added to the initiate payment JSON response. Following this redirect link a PSU will be redirected to the bank payment summary and approval form where PSU has to enter their PIN2 credentials. Also, `Aspsp-Scs-Approach: REDIRECT` header will be added to the response.

Using explicit method TPP will have to make additional call for consent authorization object creation. A separate call start the authorisation process for a payment will create consent authorization object and return `scaRedirect` steering link inside JSON response. Same as in implicit method following this redirect link will redirect PSU to the OTP bank payment summary and SCA selection, approval form. It's highly recommended to use implicit method with SCA redirect approach.

#### 7.6.2. Payment authorisation: decoupled SCA approach

In *decoupled* approach an explicit authorisation method only exists this means that TPP has always to make additional calls to the API after *create consent* call execution. In the first step TPP has to call *create consent* endpoint without `Tpp-Redirect-Preferred` header or setting this header value to false. In response TPP will get *startAuthorisation* steering link. In the second step TPP has to *start authorization process for a consent* using HTTP POST method. After executing this call TPP will receive a list of available SCA methods inside `scaMethods` array and *selectAuthenticationMethod* hyperlink in the JSON response. SCA methods list should be depicted in TPP environment so that PSU could select preferred SCA method (mobile signature, smart ID and etc.).

During this call ASPSP has to initialize internal SCA providers process which will push OTP challenge data to the PSU device and adds same challenge code data to the JSON response of the update PSU data for payment request. PSU has to confirm this challenge using PIN2 code. If the confirmation was successful payment transaction status will be changed from PDNG (Pending) to ACSC (AcceptedSettlementCompleted) and authorization object will be finalized. If the authorization is unsuccessful payment transaction status won't change but authorization object status will be changed to failed. In this case TPP should start authorization process from the second step: start the authorization process for a payment.

## 8. Miscellaneous