Skandiabanken
Technical documentation Open Banking

1 Introduction

Welcome to Skandiabanken Open Banking solutions!

1.1 Background

This document and our information site are created to provide the information needed to access our API's. If you have questions or suggestions, you are welcome to contact us. Contact information can be found in section 1.3 Contact us and channels for communication.

Info site: www.skandia.se/openbanking
Portal: https://developer.skandia.se/open-banking/core-bank/
Test-Portal: https://developer.test.skandia.se/open-banking/core-bank/

1.2 Status

This section will clarify what we have in place and the status of the functionality. Please read the rest of this documentation for further information and details.

Functionality	Available	Comments
SCA	Yes	Redirect and decoupled solution offered using Swedish BankID.
TPP identification (eIDAS)	Yes	We support eIDAS/QWAC. QSealC is currently not required but we are investigating it.
Relevant standards	Yes	Berlin Group NextGenPSD2 Framework version 1.3.6 OAuth2 including OpenID Connect TLS 1.2
AIS	Yes	Payment account information for private customers over 18 years. We have no business customers with payment accounts.
PIS	Yes	Domestic transfer, cross border payments, recurring payments, giro payments and signing basket.

During the timespan 04:00-04:30 each day our service provider restarts services and servers, which has impact on performance on all our channels (Online, App and Open Banking). For you to get the best experience, and in the end our common end customers, we recommend that during that time not to schedule batch calls. Response times will be much longer, and error rate will be higher. We apologize for this but we want to give you as much insight as possible to obtain the best possible service.

1.3 Contact us and channels for communication

If you have questions, feedback or general problems you are welcome to contact us via mail openbanking@skandia.se. This mailbox is mainly monitored during business hours. Please use this for technical questions or business inquires.

If you have very urgent issues outside business hours, please contact our customer and TPP support: https://www.skandia.se/kontakta-skandia/kontakta-oss/.
Mainly use phone: +46771555500
Open hours 07-23 Monday through Friday, 08-22 Saturday and Sunday, year around.

1.4 Abbreviations/Acronyms

Abbreviation/Acronym	Description
OTP	One-time password Sometimes Skandiabanken sends an OTP, one-time password, to the PSU for additional security. An OTP (6 digits, range 100000-999999) is sent to PSU's mobile phone number and should be entered by PSU and is then verified by Skandiabanken.

2 Document history

Version	Date	Description
1.42	2026-09-01	- Authorization and token URLs added for AIS authentication in test environment/sandbox. - Info about request and responses and examples moved to Swagger - Response types for decoupled authentication and signing consolidated into single chapter.
1.41	2026-07-02	- Revised information about giro cut off time
1.40	2026-05-08	- Revised information about updating certificates with different issuers
1.39	2026-03-20	- Revised documentation to align with new API versions (PIS v4, AIS v3) - Added important note on error handling about timeout responses from API-Gateway - Update PSU-IP-Address for Get Payment Status to not mandatory
1.38	2025-11-11	Added information about possible returned http status code 429
1.37	2025-06-26	Fixed query parameters in 9.2.1 Authorize, initiate authentication and get available authentication methods
1.36	2025-05-27	Removed scope information and examples with url-encoded values
1.35	2025-05	- Decoupled PSU authentication for AIS access and textual fixes
1.34	2025-04-11	- Added information about auth flow for new auth servers
1.33	2025-02-10	- Info RequestedExecutionDate: max 2 years in the future. - Info RequestedExecutionDate (cross border payment): max 6 months in the future. - New processing status: AMOUNT_LIMIT_EXCEEDED - Removed optional parameter CreditorAgentName
1.32	2024-12-19	- Signing basket operations in PIS v3: 8.4, **Error! Reference source not found.**, **Error! Reference source not found.**, and **Error! Reference source not found.**. - 10.2 Sequence diagram of decoupled SCA method.
1.31	2024-10-08	9 Decoupled SCA method

3 Test environment and sandbox

We currently support testing of the same services as we have in production.

To be able to use the test environment you need:

- Test or production eIDAS-certificate. If you use a test certificate:
  - The certificate must be issued by a publicly trusted CA. No self-signed certificates will be accepted.
  - The certificate must contain a correct Organization Identifier (2.5.4.9.7 = PSDXX-YYYY-ZZZZZZZZ)
  - The certificate must contain a QC Statement with the PSD2 roles that you intend to use.
- A test BankID. (Swedish BankID is currently the only supported SCA).
  - The BankID client must be configured as a test BankID client. More info about how this is done is available at demo.bankid.com

3.1 Onboarding of TPP in Test Environment

The process of onboarding in the Test Portal is the same as onboarding in the Production Environment but with a few exceptions.

1. You reach the test portal at https://developer.test.skandia.se/open-banking/core-bank/
   a. You will have to create separate users for the Test and Production Portals.
   b. Since there are separate users you will receive different Client-Ids in Test and Production.
2. You only need a test eIDAS-certificate and not a production certificate.
   a. You can use your production certificate, but it is not necessary.

Except for the above differences you can follow the instruction in 4 Onboarding of TPP in Production Environment when onboarding in the Test Environment.

3.2 Guidelines for the Test Environment

There is a limited number of test PSU's in the environment, please contact openbanking@skandia.se after onboarding to receive a specific test PSU to run tests on.

The account numbers used in testing must follow the logic of real accounts. You cannot test transfers to made up accounts. Notice that other TPPs also can see the test PSUs transactions history. Any account numbers used in testing can therefore be seen by other TPPs.

3.3 Authorization and Token Endpoint URLs for AIS Authentication

The general authentication flow for AIS is described in chapter 6.

AIS - Security and authentication of TPP including SCA of PSU.

URLs to use in test environment/sandbox:

Authorization endpoint (redirect SCA approach): https://csts-test.skandia.se/ct2/oauth/v2/oauth-authorize
Token endpoint (redirect and decoupled SCA approach): https://csts-test.skandia.se/ct2/oauth/v2/oauth-token

4 Onboarding of TPP in Production Environment

This section provides a process overview how a TPP can gain access to our production environment including the developer portal and API gateway.

1. Get an approval from a local NCA (Finansinspektionen in Sweden) as an AISP or PISP. If you are a bank you already have an approval.
2. Obtain a production eIDAS-certificate from a QTSP. QWAC is required.
    a. Note 1: you cannot use a CA root certificate
    b. Note 2: you cannot use a test certificate
3. Register in our portal and create apps and subscriptions. You will need the public key of your production eIDAS QWAC certificate in this process.
4. When you add a subscription to an app there will be a manual step for us to grant your app access. You will receive an e-mail when your app is granted access.
5. You can access the APIs (AIS and/or PIS) your certificate allows you to invoke

4.1 Create an account in the portal

Browse to the portal (https://developer.skandia.se/open-banking/core-bank).

To register an account:

1. Navigate to the portal. Click the “CREATE ACCOUNT” button (top right corner).
2. Enter the fields:
    a. user name (please make a note of the selected user name, it is required for login)
    b. email address (will be used for communication with your organization)
    c. phone (optional but recommended)
    d. first name
    e. last name
    f. consumer organization (defines the name of the collective under which client applications will be created)
    g. password and confirm password and captcha. Press the “Sign up” button
3. Open the email received from the portal and follow the activation link. This will activate the user account.
4. Login using the username and password selected by you

4.2 Adding a colleague to your consumer organization

It is possible to invite additional users to your consumer organization. This can be done while logged in as the consumer organization owner/creator or if you possess the administrator role of that specific consumer organization.

The new user is invited using an e-mail address. An e-mail containing a registration link is sent to the provided e-mail address. If the e-mail is registered to an existing user account, following the link will add that user to the specific consumer organization. If the e-mail is previously unused, following the link will require the user to step through the process for creating an account except that no new consumer organization needs to be created.

4.3 How to setup your apps and subscriptions

Product

The API Products view provides an overview of the products available in the portal. A product acts as a collection of APIs. To enable invocation of an API, you must subscribe with an app to a product’s plan containing the desired API.

API

An API is a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service. The API can be protected by various security mechanism, including, but not limited to, Client-Id identification, MTLS and OAuth.

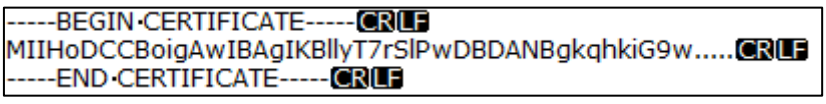
We expose a set of APIs which makes it possible for third parties to create applications using payment account information and payment initiation.

Plan
A plan represents a collection of APIs from a specific product. Subscribing to a plan allows the consuming app to invoke the APIs included in the plan given the correct security protocols are satisfied. A plan can contain terms of the usage of its APIs, including invocation rate limits and debit.

Apps
Apps are the components that are used to invoke the APIs. An app has its own set of security credentials, Client-Id and client secret.

To create an app, the following fields are required:

Field	Comment
Title	The name of the app (only visible to you and us)
QWAC Certificate	Paste the content of the public part of your eIDAS QWAC x509 Certificate in PEM format with base 64 encoding. Please provide PEM encapsulation (BEGIN CERTIFICATE, END CERTIFICATE) on separate lines with the base 64 encoded certificate on a single line between the encapsulations. This certificate will be used for both MTLS and to determine your TPP accessibility. The certificate should contain the whole certificate chain.
QSEAL Certificate (optional)	Paste the content of the public part of your eIDAS QSEAL x509 Certificate in PEM format with base 64 encoding. We currently do not validate QSEAL but will do later, by entering this field the process will be easier for you when we require QSEAL. The certificate should contain the whole certificate chain.
Application OAuth Redirect URL(s)	A list of redirect urls to be used in the OAuth Authorization code flow. These addresses will be used to receive the authorization code required when claiming the access token for the API invocation. The url: • Must follow the format (?=\w+:\/\/?*?\w+) • Cannot contain intermediate white spaces There is a timeout for how long an authorization code is valid, and that´s 60 seconds. If it takes longer than that to replace it with an access token, you will get the error message authorization code is invalid or expired.
Sign Redirect Url (optional)	Does not need to be included here since this information is not used. The redirect URL that the PISP wants to use must instead be included in the call.



***Above**: example of a correctly PEM-formatted public part of a QWAC Certificate*

After creating an app, Client-id and client secret will be provided. Take note, this is the only time the client secret will be shown for you. It is possible to reset the client secret by navigating to apps, select an app, click subscriptions, select the menu present in the top right corner of the credentials window and click reset client secret.

Step by step to create an App
1. While logged in, click the “APPS” label from the top menu
2. Click the “Create new app” button

3. Enter mandatory fields title, QWAC-certificate[1], application OAuth redirect URL(s) (minimum 1) and click the 'Submit' button
4. Take note of the API Key (Client-Id) and Secret and click the "Continue" button

Step by step to update an App
1. While logged in, click the "APPS" label from the top menu
2. Select and click the App to edit
3. Click the three vertical dots in the top right corner and select "Edit" from the drop-down menu
4. Change any of the available fields, including title, application OAuth Redirect URL(s) and Sign Redirect URL.
5. Click the 'Submit' button

Step by step to reset client secret
1. While logged in, click the "APPS" label from the top menu
2. Select and click the app to which you would like to reset client secret
3. Click the "Subscription" link located below the app name
4. Click the three vertical dots located in the top right corner of the credentials window and select "Reset Client Secret" from the drop-down menu
5. Click the "Reset" button
6. Check the 'Show' checkbox at to the top of the page
7. Take note of the new client secret

Step by step to delete an App
1. While logged in, click the "APPS" label from the top menu
2. Select and click the app to be deleted
3. Click the three vertical dots in the top right corner and select "Delete" from the drop-down menu
4. Click the "Delete" button

Subscription
A subscription defines a connection between a plan and an app. The plan defines the how and what of the subscription, including what APIs and how the APIs are accessible. The app defines the which of the subscriptions and hence details what credentials that are to be used against the gateway when attempting to invoke any of the APIs. Without a valid subscription, an app can never invoke a published API in the production environment.
To subscribe to a plan, go to the product overview page, select a product, click the plan you desire to subscribe to. Choose the App to which the subscription should be created for and confirm the subscription.

The act of requesting a new subscription will require a manual approval from us before the client app is enabled to invoke the API. It is possible to track the approval status by navigating to the app's subscriptions overview. If you know in advance that you need to get a fast approval, please contact us via mail and let us know a business day ahead.

Step by step to create a subscription
1. While logged in, click "API PRODUCTS" from the top menu
2. Select and click the product that the subscription should regard
3. Select and click the 'Subscribe' button of the desired Plan
4. Select and click the 'Select App' button of the desired app
5. Overview the subscription details and press the "Next" button
6. Click the "Done" button

Step by step to view Subscription status
1. While logged in, click the "APPS" label from the top menu
2. Select and click the app to which the subscription belongs to
3. Click the 'Subscription' link located below the app name
4. View the status of subscriptions in the subscriptions window

[1] The eIDAS certificate holds authorization/the role you have and regulates if you can access AIS and/or PIS. If you as an ASIP create an app/subscription for PIS, the subscription will be denied.

Step by step to remove a Subscription

1. While logged in, click the “APPS” label from the top menu
2. Select and click the app to which the subscription belongs to
3. Click the “Subscription” link located below the app name
4. Click the three vertical dots located to the right of the specific subscription located in the Subscriptions window
5. Select “Unsubscribe” from the drop-down menu
6. Click the “Unsubscribe” button

4.4 Update eIDAS-certificate

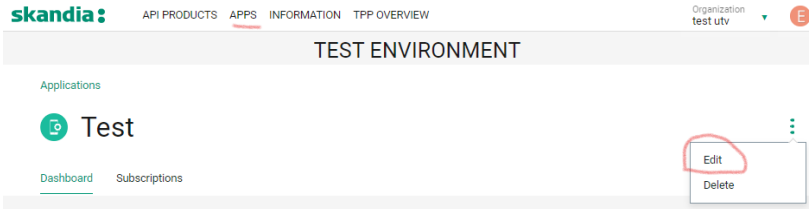
An eIDAS certificate has an expiration date and must be changed/updated before that date, otherwise the subscription will stop working. Below are instructions describing how to update an existing certificate.
An application can have multiple certificates active simultaneously. This allows you to add a new certificate before the old one expires, ensuring continuity of service.

If the new certificate is issued by a different Certificate Authority, you must contact our support before the certificate is used. In these cases, the new certificate must be manually added. Please ensure that support has completed this process before placing the certificate into production use.

Certificate updates can only be performed if the certificate roles are the same as in the existing certificate. If the roles differ, a new application must be created.

Step by step to update QWAC/Certificate

1. While logged in, click the “APPS” label from the top menu
2. Select and click the app to update
3. Click the three vertical dots
4. Select “Edit”



5. Paste the new QWAC/Certificate into the box “**New QWAC Certificate**” PEM, base-64
6. Click “Submit”

If checks are ok the new certificate will be updated and will immediately start being used going forward.
The old certificate will no longer be used and will automatically be removed when expiration date is passed.

If checks are not ok, read the message.

5 General information about API invocation

The Developer Portal provides comprehensive documentation for all APIs, including endpoint specifications, request headers, field descriptions for requests and responses, and example messages. This current document complements the OpenAPI specifications, illustrating the API flows and providing deeper understanding of the processes.

The OpenAPI specification for each API can be downloaded from the Developer Portal and viewed using any OpenAPI-compatible tool of your choice.

Developer Portal (Production Environment)
https://developer.skandia.se/open-banking/core-bank/

Developer Portal (Test Environment)
https://developer.test.skandia.se/open-banking/core-bank/

5.1 Gateway

Invoking the APIs requires three security measures:

1. Header “Client-Id” containing the client ID of your App created in the developer portal. The app holds a valid subscription containing the specific API.
2. MTLS with the eIDAS QWAC certificate provided when creating the App.
3. For AIS: An Authorization header containing a bearer access token of type reference, containing the scope demanded by the API definition.

5.2 General error messages

The APIs return errors that are common for all the available products as well as ones that are specific for the API product or method.

5.2.0 HTTP 401 Unauthorized

These are common 401 Unauthorized response messages and their implication:

- ***Invalid client id or secret:*** You are attempting to invoke the API using a Client-Id that is not valid for the specific API.
- ***Client certificates for mutual TLS in the API request doesn't match the registered certificate*:** The provided Client-Id is correct, but you are using a certificate that is not related to that Client-Id.
- ***Cannot find valid subscription for the incoming API request:*** The provided Client-Id is valid, but you are not authorized to invoke this API as that Client-Id (App) does not possess an approved subscription.
- ***Cannot pass the security checks that are required by the target API or operation, enable debug headers for more details:*** Could be one of the following three reasons.
    - Access token is not a valid reference token.
    - Access token has expired.
    - Access token does not contain the scope required by the API

5.2.1 HTTP 429 Too Many Requests

Too many requests. This error occurs when the client exceeds the allowed number of requests within a defined time window.

5.2.2 HTTP 500 Internal Server Error (Important Note on Error Handling)

When receiving an HTTP 500 Internal Server Error response from the API Gateway with the message:

```
{
   "httpCode": "500",
   "httpMessage": "URL Open error",
   "moreInformation": "Could not connect to endpoint"
}
```

This error does not indicate a server-side failure. Instead, it typically results from a timeout or network connectivity issue between the API Gateway and the downstream service.

Before retrying the operation, check the status of the payment or related process. The original request may have been completed successfully before the timeout occurred. Verifying the status helps prevent duplicate actions or inconsistencies.

5.2.3 HTTP 503 Service Unavailable

The service is unavailable. This can be caused by for example high load on the server or other unplanned issues. This error should be temporary and it is recommended to try again later.

5.3 Overview of available accounts and customers

The API:s give access to payment accounts owned by private consumers over the age of 18.

Summary of available accounts:

- “Allt i Ett-konto”. Transaction account, typically used for everyday transactions and bill payments
- “Sparkonto”. Typically, a savings account. Only domestic transfers possible, no bill payments etc.
- No fixed rate accounts are available
- The consumer must own the account
    - We have no co-owned accounts

6 AIS - Security and authentication of TPP including SCA of PSU

6.1 OAuth2

We use OAuth 2.0 for authentication and authorization. Our APIs are protected by the OAuth Authorization Code Grant type which is depicted in detail at https://oauth.net/2/.

6.2 Request Access code and ultimately Access token

To be granted access to the PSU's accounts, the PSU needs to be authenticated by us. This is performed using a redirect or decoupled SCA approach. Swedish BankID is currently the only supported SCA.

The first step is to retrieve an authorization code. The code can be retrieved either with a redirect or decoupled approach.

- In the redirect approach the PSU is redirected to Skandia's server and the code is returned in the redirect URI provided by you.
- In the decoupled approach the TPP calls Skandia Open Banking OAuth API and manages the user interface itself. The code is returned in the last call to the API.

When an authorization code has been retrieved, a token can be fetched from the token endpoint using the code.

The Swedish personal identity number for the person who authenticated can be found in the id token when scope openid is used.

6.2.0 Request for authorization, redirect

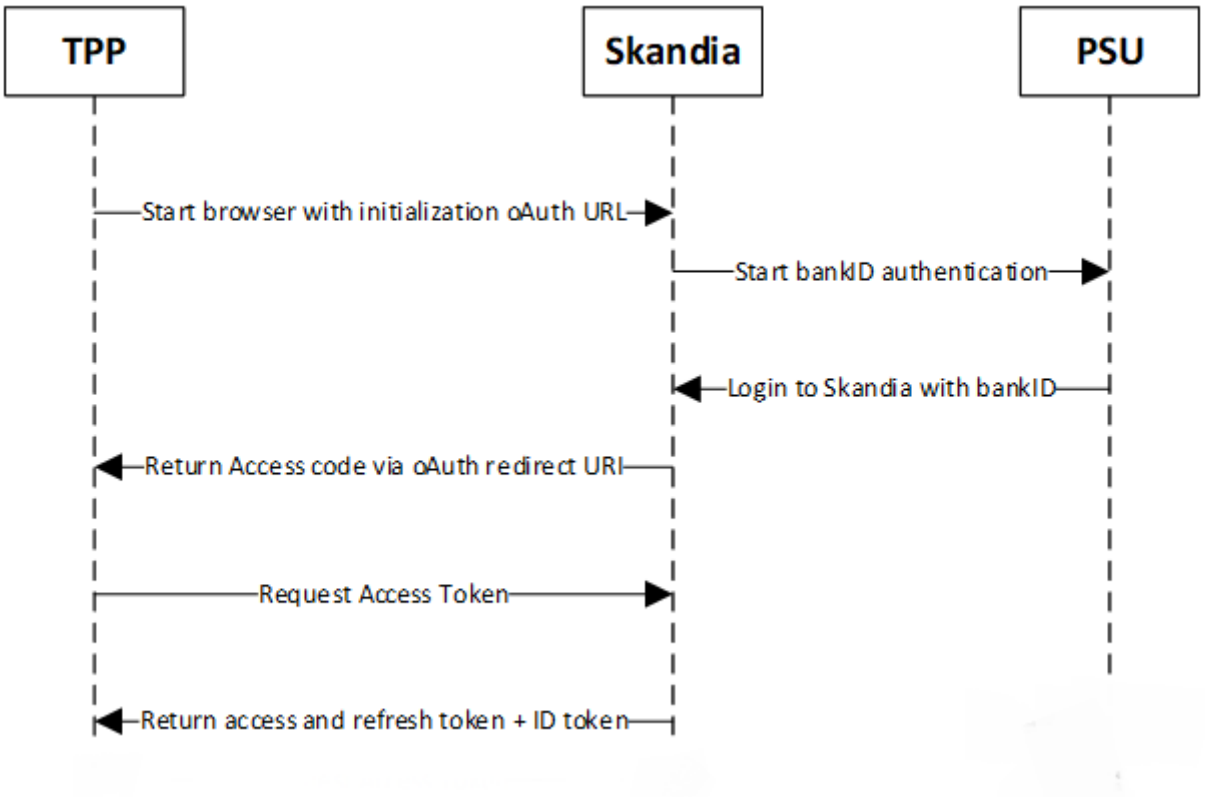


Figure 1 Overview of the authentication flow.

1. The TPP, makes a request to Skandia for the authentication of the PSU. In the request, your Client Id, a redirect URI and requested scopes must be provided.
2. The PSU identifies themselves using Swedish BankID

3. If BankID identification is successful, an access code is returned to the redirect URI provided by the TPP.
4. The TPP can now request access token using the code.

Use the client_id (denoted “Key” in Figure 2) and client_secret (denoted “Secret” in Figure 2) received when you created the app in the developer portal along with the redirect URI’s that you provided for that specific app.

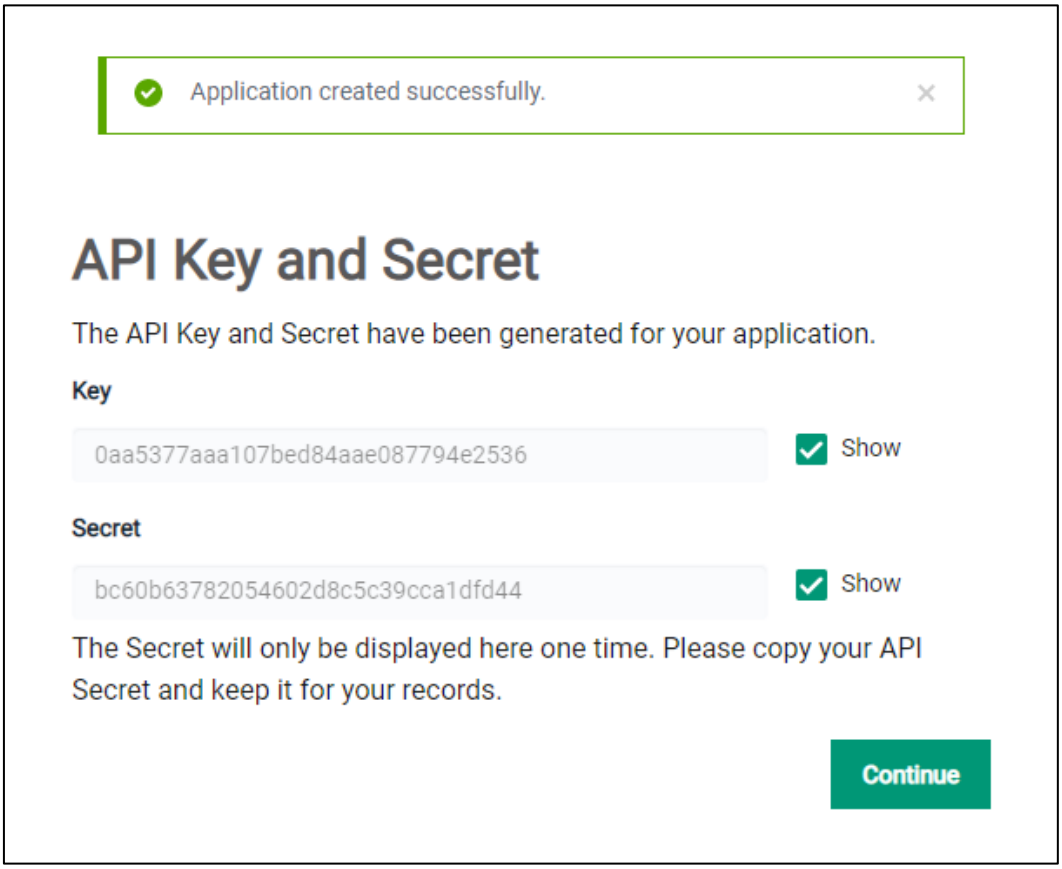


Figure 2: Example of view when an application has been created and the Key and Secret are shown.

The authorization request[2] should contain the following query parameters:

- **response_type:** code
- **client_id:** The client_id (denoted “Key” in Figure 2) of your App in the developer portal
- **redirect_uri:** The redirect URI that you have provided for the app in the portal. You cannot have a '/' as the last character in the URL, because it will be removed by the system. It must be URL encoded.
- **scope:** Should at least contain the API specific scope, i.e. psd2.aisp for AIS or psd2.pisp for PIS, and the openid scope. You must call on the format "scope=openid psd2.aisp psd2.pisp".
- **state:** A random string that should be validated to be identical in the redirect return after the user authorizes the app.
- **code_challenge:** When the App begins the authorization request, instead of immediately launching a browser, the client first creates what is known as a “*code verifier*“. This is a cryptographically random string using the characters A-Z, a-z, 0-9, and the punctuation characters -._~ (hyphen, period, underscore, and tilde), between 43 and 128 characters long.
  Once the app has generated the code verifier, it uses that to derive the *code challenge*. The code challenge is a Base64-URL-encoded string of the SHA256 hash of the code verifier.
- **code_challenge_method:** S256

[2] For production environment use: https://csts.skandia.se/prod
For test environment use: https://csts-test.skandia.se/ct2

Example Authorization Request:

```
GET https://csts.skandia.se/prod/oauth/v2/oauth-authorize?
response_type=code
&client_id=0aa5377aaa107bed84aae087794e2536
&redirect_uri=https://localhost/
&scope=openid psd2.aisp
&state=ca17f9d039024a789493641d8cdbba14
&code_challenge=N1rZDhxSTs-WZ8-jpKOSlzxaLjFT8QWoczBSXVlItgw
&code_challenge_method=S256
```

The PSU is directed to Skandia's authentication server and prompted to identify themselves with BankID. After authentication has been completed successfully, a code and state are returned to the redirect location. In this example, the PSU will be redirected to the below URL:

```
https://localhost/
?code=Im6_IkPhrd67m2gFw9upLilm9h32PmM51gUAAAAC
&state=ca17f9d039024a789493641d8cdbba14
```

6.2.1 Request for authorization, decoupled

See chapter 9, Decoupled SCA approach for AIS access.

6.2.2 Request for access token

The code received in the redirect or decoupled approach is used when requesting the access token.

Body Parameters:

- **grant_type:** authorization_code
- **code:** Access code received in the redirect from the authorization request or received in the OauthCode response when using decoupled approach.
- **redirect_uri:** Same redirect URI that was used in the authorization request.
- **client_id:** The ID of your App created in the developer portal (denoted "Key" in Figure 2).
- **client_secret:** Is received when creating the App (denoted "Secret" in Figure 2).
- **code_verifier:** The same code_verifier used to derive **code_challenge** in the authorize call

Example Access Token Request:

```
POST https://csts.skandia.se/prod/oauth/v2/oauth-token
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code
&code=661e9996602e47f7b23039a441c91061
&redirect_uri=https://localhost/
&client_id=0aa5377aaa107bed84aae087794e2536
&client_secret=bc60b63782054602d8c5c39cca1dfd44
&code_verifier= MTIzNDU2NzkwMTIzNDU2NzkwMTIzNDU2NzkwMTIzNDU2Nzkw
```

Response

```
{
    "id_token": "XXXXXX",
    "token_type": "bearer",
    "access_token": "_0XBPWQQ_dfbff42a-612e-4aa2-8a65-2730533f3eb6",
    "refresh_token": "_1XBPWQQ_0bd893af-4282-40ee-babd-13a4b024afce",
    "scope": "openid psd2.aisp",
    "expires_in": 7200
}
```

6.2.3 Token types and its uses

To access the APIs, you need to provide a valid access token in the Authorization header. The access token is a short-lived token, valid only for 2 hours. After 2 hours you will get HTTP 401 Unauthorized. When the access token has expired, you can use the refresh token to get a new access token.
The access token can be refreshed for a maximum time of 180 day from the initial time of authentication. After 180 days the PSU needs to be authenticated again as previously described.

The ID token (of type OpenID Connect) is a representation of the identity (SSN, "*personnummer*") of the authenticated PSU. This enables you as a TPP to validate that the PSU using your services corresponds to the PSU authenticated by us.

6.2.4 Refresh token

Refreshing a token is possible by using the refresh_token provided in the response to the access token request. The refresh token is only valid for one time use. When a refresh token is used to issue a new access token, the response will contain the new access token and a new refresh token.
The request should contain the following body parameters:

- **grant_type**: refresh_token
- **refresh_token:** The refresh token received from the previous access token request.
- **client_id:** The client_id of your created app in the portal.
- **client_secret:** The client_secret received when creating the app.

Example Refresh Token Request:

```
POST https://csts.skandia.se/prod/oauth/v2/oauth-token
Content-Type: application/x-www-form-urlencoded

grant_type=refresh_token
&refresh_token=661e9996602e47f7b23039a441c91061
&client_id=0aa5377aaa107bed84aae087794e2536
&client_secret=bc60b63782054602d8c5c39cca1dfd44
```

Response

```
HTTP/1.1 200 OK
{
  "token_type": "bearer",
  "access_token": "_0XBPWQQ_a50f2da8-9671-4425-82ea-216803994e04",
  "refresh_token": "_1XBPWQQ_7e675c18-2e9c-417b-883f-6e470f6b6b75",
  "scope": "openid psd2.aisp",
  "expires_in": 7200
}
```

6.2.5 AIS consent

You as a TPP are responsible of handling the consent that the customer gives you and act according to it. We do not.

After authentication by the PSU you have access to account information for 180 days without a renewed authentication. You will have access to the accounts present at the time of the SCA and any new payment accounts the PSU creates.

If the PSU is authenticated again within the 180 days, you will receive a new token valid for another 180 days. If you for some reason, for example due to some agreement with the PSU, do not want access for another 180 days, you simply throw away the new token and continue using the old one.

7 AIS v3

7.1 Introduction

The Account Information Services API is used to retrieve account information—such as transactions and balances—for customers' payment accounts.

The API contains 5 operations:

- Get a list of all payment accounts for a given PSU (identified from access token)
- Get account details for specific account
- Get balances for specific account
- Get transactions for specific account
- Get transaction detail for specific transaction

Detailed descriptions of operations, request URLs, request and response models and examples can be found in AIS API description in the developer portal.

All operations require an access token identifying the PSU, which must be sent as a Bearer token in the Authorization header.

The ResourceId found in the account list response should be used as input in the other operations to address the specific account.

Prerequisites:

1. You must have a registered user account in the portal (see section 4.1 Create an account in the portal)
2. You must have an app set up in the portal with a QWAC-type eIDAS certificate that identifies you as an AISP (see section 4.3 How to setup your apps and subscriptions)

Flow for first time access for a given customer:

1. The PSU agrees to let you use your services to retrieve account information from us
2. OAuth2/SCA – The customer is informed about sharing information with the TPP
    a. The OAuth token will contain the ID of the customer
3. AISP invokes the AIS operation of choice and receives a response with the requested information

7.2 Common Headers

All AIS requests require the following set of headers:

Name
Client-Id
X-Request-ID
Authorization

8 PIS v4

8.1 Introduction

Pre-requisites:

1. Have registered a user account in the portal
2. Have an app setup in the portal with an eIDAS-certificate of QWAC-type telling us you are an PISP

The PIS API is available for you to:

- initiate payment
- start payment authorization
- get payment status
- get payment information
- cancel payment

Detailed descriptions of operations, request URLs, request and response models and examples can be found in PIS API description in the developer portal.

We support 3 kinds of payment products:

- domestic transfer
- giro payment
- cross border payment

A domestic transfer could be one time or recurring (payment service)

The specific payment service and payment type to adress is expressed in the request URL path according to the pattern: /{payment-service}/{payment-product} where:
Payment-service is: ‘payments’ or ‘periodic-payments’ (periodic-payments allowed only for domestic transfer)
Payment-product is one of: ‘domestic-transfer’, ‘giro-payment’, ‘cross-border-payment ‘

We also support signing basket with one-time domestic transfers and giro payments.
The signing basket operations are:

- Create signing basket
- Start signing basket authorization
- Get signing basket status (status for each payment)

Both redirect and decoupled SCA approach are offered.

8.2 Common headers

All PIS requests allow the following set of headers:

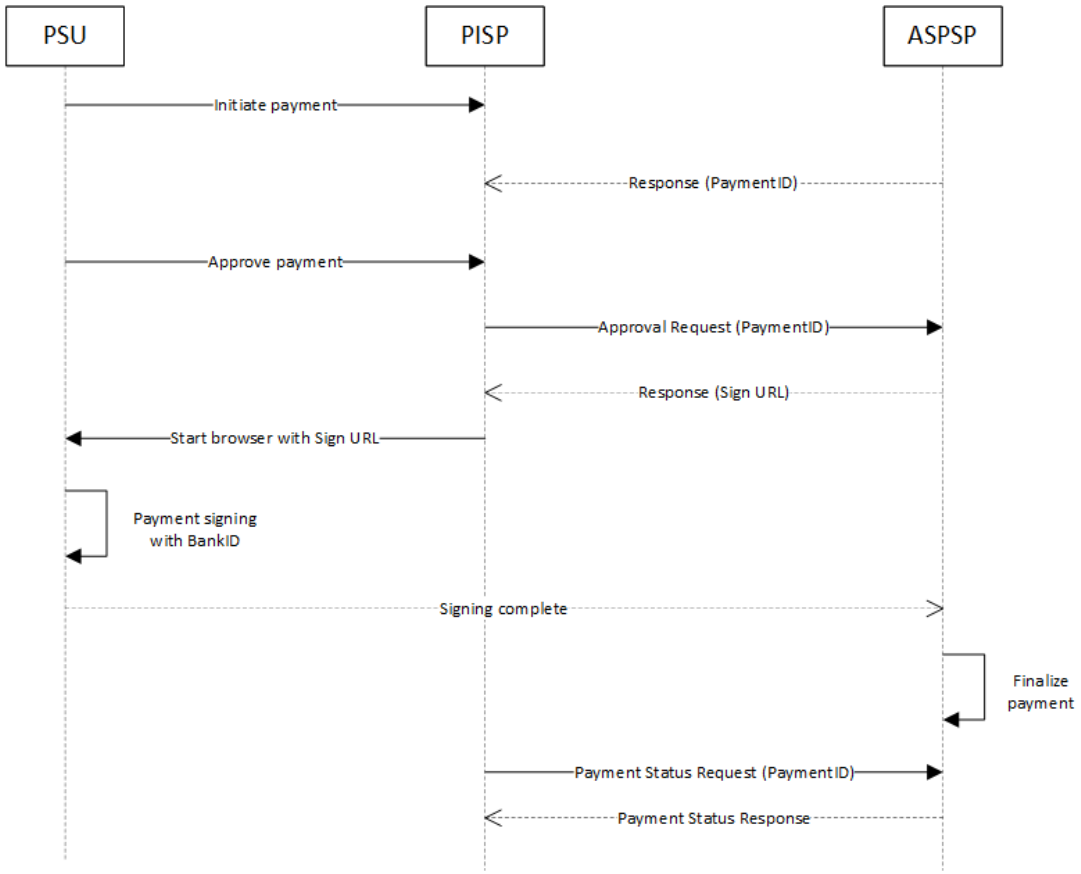
Name
Client-Id (required)
X-Request-ID (required)
PSU-IP-Address (required except for status requests)
PSU-IP-Port
PSU-Accept
PSU-Accept-Charset
PSU-Accept-Encoding
PSU-Accept-Language
PSU-Device-ID
PSU-Geo-Location
PSU-Http-Method
PSU-User-Agent

8.3 Payment Initiation flow

Skandiabanken is using a two-step Payment flow. The first step is for the TPP to Initiate the payment. The TPP can then approve and confirm the payment directly or at a later stage. (But no later than 24 hours after initiating the payment. In that case a new payment must be initiated.)

It is a single sign SCA approach, which means the customers do not need to be identified by the bank before initiating the payment. Instead, authentication occurs at the same time as the signing of the payment. Skandiabanken offers both a redirect and a decoupled SCA approach. See section 9 for a description of the decoupled SCA approach.

Payment flow (with redirect SCA approach)



8.4 Signing Basket flow

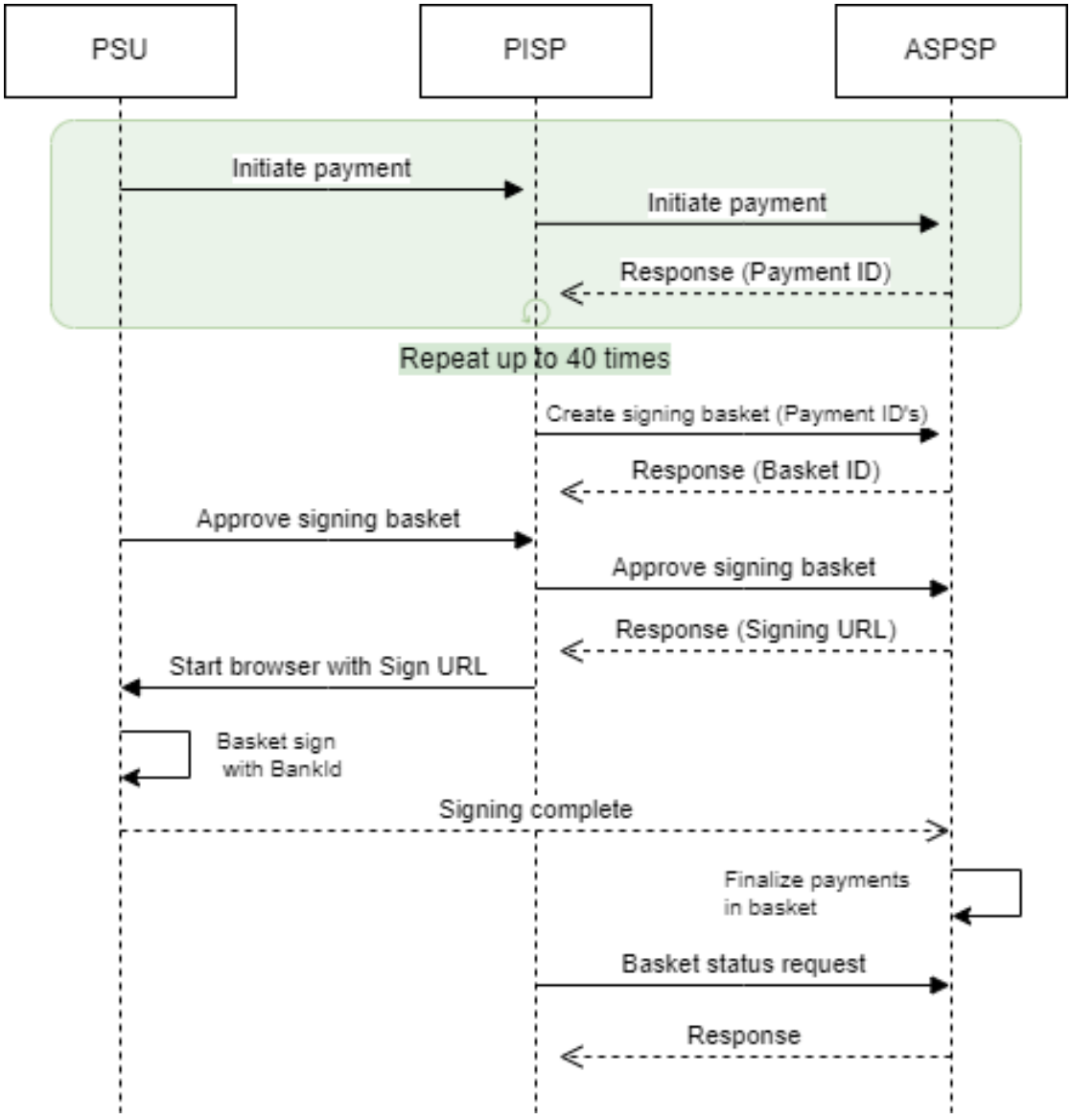
Signing baskets are a way to sign multiple payments with only one SCA. The process of a signing basket consists of first initiating payments using the ordinary payment initiation endpoints. Then the signing basket can be created with up to 40 payments.
The created basket can then be authorized with one SCA. Upon authorization, the payments must all have the status RCVD.

Only domestic transfers and giro payments can be included in a signing basket.

The payments within one signing basket must be created by the same PISP that is creating the basket. Also note that the payments within the signing basket must be created the same day that the signing basket is created.

As with the single payment flow, it is recommended that the PISP checks the status of the basket to verify the status of the included payments after the signing is completed.

If you wish to cancel the payments within a signing basket, each individual payment must be cancelled using the ordinary cancel payment method. The basket itself cannot be deleted.



8.5 Rules and different statuses for Payments

This section will contain information that is necessary for TPPs to create a good user experience around payments. Here are answers about when payments can be made and when those payments are booked and what status a payment gets in different situations.

8.5.0 Execution rules for payments

Product	Initation Time	Execution Time
RequestedExecutionDate as current day on a business day		
Domestic-Transfer	Before 13:45	Same day (These transfers can not be cancelled.)
	After 13:45	Next business day. (These transfers can not be cancelled.)
Domestic-Transfer within Skandiabanken	Same day (regardless if business day or not). (These transfers can not be cancelled.)	
Giro-Payment	You can only choose same day as RequestedExecutionDay before 08:55. After that you must choose a later date. These transactions can be cancelled by PSU in other channels.	
Cross-border-transfers	1-2 days to EU/EES in Euros, 2-3 days to EU/EES in other currencies, 3-5 for other countries.	
RequestedExecutionDate on a non-business day		
Domestic -Transfer	As current date on a non-business day	Next business day. (These transfers cannot be cancelled.)
	On a business day	First business day after RequestedExecutionDate. (Transfers can be cancelled.)
Giro-Payment	Giro cannot be initiated with a RequestedExecutionDay that is on a non-business day	
Cross-border -transfers	Same as for business days.	

* If the recipient bank is located outside the EU / EEA, the bank cannot provide any guarantees regarding execution times for payment transactions.

8.5.1 Examples of transaction status codes during different stages of PIS operation.

Operation	Day of initiation	Requested Execution Day	Status after Initiate	Status after approve & sign	Status booked transaction
Domestic -Transfer (same day)	Monday	Monday	RCVD	ACSC	ACSC
Domestic -Transfer (weekend)	Saturday	Saturday	RCVD	ACSC	ACSC. Funds transferred on Monday.
Domestic -Transfer (future)	Monday	Wednesday	RCVD	ACSP	ACSC
Domestic -Transfer (future weekend)	Monday	Saturday	RCVD	ACSP	ACSC. Funds transferred on Monday.

Giro-Payment (before 08:55 same day)	Monday	Monday	RCVD	ACSC	ACSC
Giro-Payment (weekend)	-	-	-	-	-
Giro-Payment (future)	Monday	Wednesday	RCVD	ACSP	ACSC
Payment cancellation	----------	----------	Current status	CANC	---------

Meaning of different transaction status codes

- RCVD - Received - Payment initiation has been received by the receiving agent.
- ACSP -AcceptedSettlementInProcess - All preceding checks such as technical validation and customer profile were successful and therefore the payment initiation has been accepted for execution.
- ACSC - AcceptedSettlementCompleted - Settlement on the debtor's account has been completed.
- CANC – Cancelled – Payment has been cancelled and will not be executed.
- PNDG – Pending – A future payment did not have enough funds on the account on Requested Execution Day and a new try will be made.
- RJCT – Rejected. Payment has been rejectad.

Meaning of different processing status codes

- PENDING – Payment has been initiated but authorisation or cancellation has not been completed.
- PROCESSED – Payment has been successfully authorised.
- CANCELLED – Payment has been successfully cancelled.
- UNPROCESSABLE – Something went wrong during the payment authorisation or cancellation. For example, if the SCA validation failed.
- INSUFFICIENT_FUNDS – Payment was rejected due to insufficient funds on the debtor account.
- AMOUNT_LIMIT_EXCEEDED – Payment was rejected because the daily bank limit has been exceeded.

Please be advised that if a payment is scheduled to be executed on a future date, then the transaction may still become rejected but still have the processing status “PROCESSED”. For example, if there are insufficient funds on the debtor account at the time of execution.

9 Decoupled SCA approach for AIS access

To access a PSU's accounts, the they must first be authenticated by Skandia. This chapter describes how to obtain an authorization code using the decoupled approach. The authorization code can subsequently be exchanged for an access token. For more information about AIS security, the redirect SCA approach, and the authorization code exchange flow, see chapter 6 AIS - Security and authentication of TPP including SCA of PSU.

9.1 Authentication methods

The following BankID-based authentication methods are supported:

Authentication method	API value	Intended use
BankID on file	BankIdSameDevice	For TPP applications running on a computer
Mobile BankID on the same device	MobiltBankIdSameDevice	For TPP applications running on a mobile device
Mobile BankID on another device	MobiltBankIdOtherDevicePnr	For TPP applications running on a computer or mobile device

The TPP is responsible for:

- Providing a graphical user interface (GUI) for the PSU
- Launching the BankID application when required
- Monitoring the authentication status until the process is completed successfully or is aborted

Successful authentication requires not only that the PSU authenticates using the BankID application, but also that any additional validation requirements, such as KYC checks, are successfully completed.

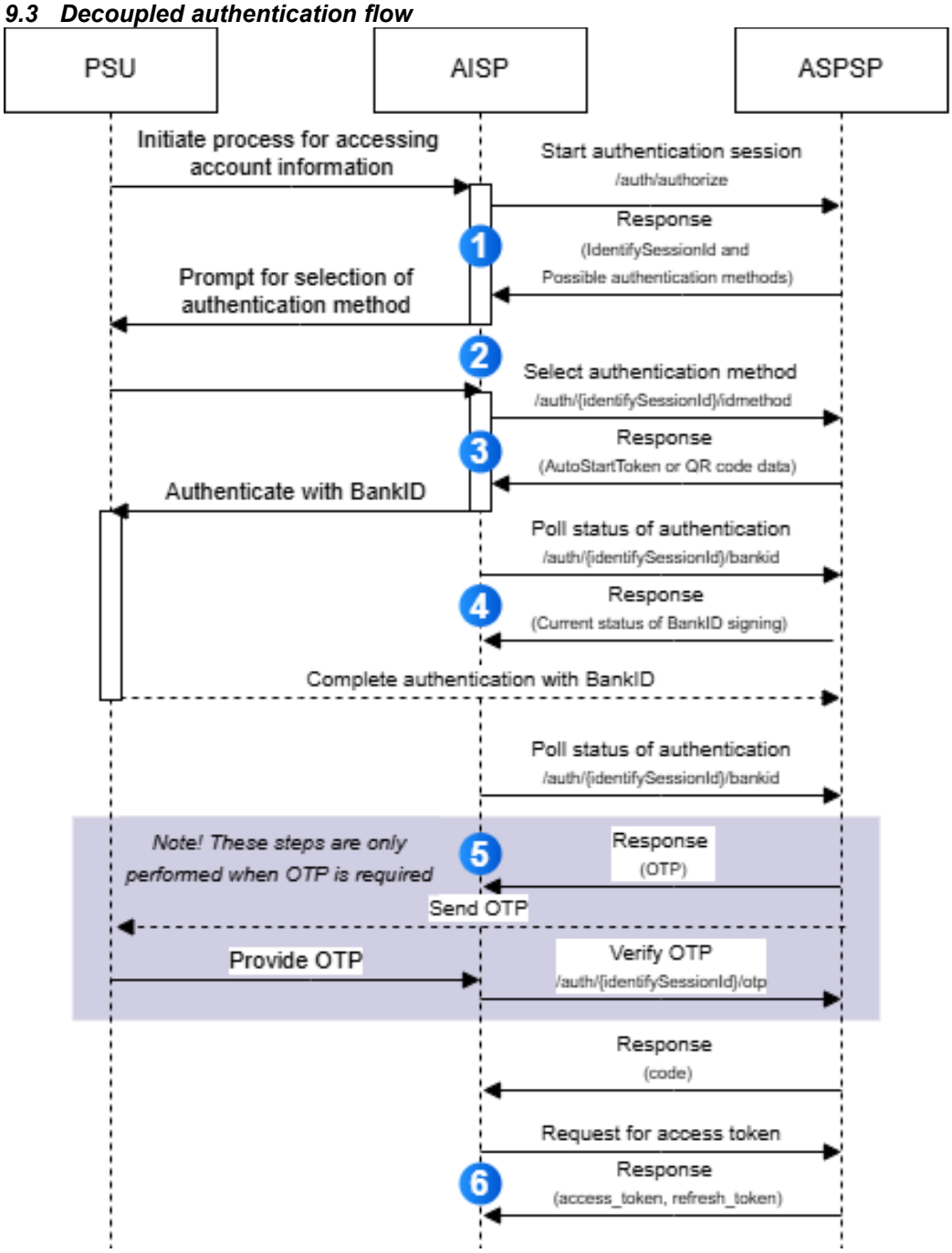
9.2 Decoupled authentication endpoints

The authentication endpoints are available through the Skandia Open Banking OAuth API. To access the API, the TPP must subscribe to the service. Request and response specifications are available in the developer portal: https://developer.skandia.se/open-banking/core-bank

The following endpoints are exposed:

- **GET /auth/authorize:** Initiates the BankID authentication and returns available authentication methods.
- **POST /auth/{identifySessionId}/idmethod:** Selection of authentication method.
- **GET /auth/{identifySessionId}/bankid:** Get authentication status.
- **POST /auth/{identifySessionId}/otp:** Verify OTP.
- **DELETE /auth/{identifySessionId}:** Cancel authentication flow.

9.3 Decoupled authentication flow



1. The TPP initiates the authentication flow by calling the **authorize** endpoint. The response contains all authentication methods available to the PSU.
2. The TPP determines whether the application is running on a desktop or mobile device and presents the applicable authentication methods in the GUI.
3. The PSU selects an authentication method.
    - When **Mobile BankID on another device** is selected, the PSU's Swedish personal identity number must be included in the call to the **idmethod** endpoint.
        a. The response contains QR code data.
        b. The TPP generates and displays the QR code in the GUI.
        c. The PSU opens the BankID application on another device and scans the QR code.
        d. The PSU enters their security code in the BankID application.
    - When **BankID on File or Mobile BankID on the Same Device** is selected.

   a. The response contains information required to start the BankID application (autoStartToken).
   b. The TPP launches the BankID application according to the BankID integration guidelines.
   c. The PSU enters their security code in the BankID application.

4. As soon as the QR code is displayed or the BankID application has been launched, the TPP should poll the authentication status endpoint once per second while the BankID order remains pending. The response to the status check may require different actions depending on the response type. Refer to section 9.5 Authentication states and response types.

5. OTP verification (if required)
   a. The TPP presents a GUI where the PSU can enter the OTP.
   b. The PSU enters the OTP received by SMS.
   c. The TPP submits the OTP for validation.
   d. The TPP processes the validation response:
      - **OTP valid –** If the OTP is valid and all additional validations, including KYC checks, have been completed successfully, the authentication flow is complete. An OAuth authorization code is returned. The TPP can proceed to exchange the authorization code for an access token and refresh token.
      - **OTP invalid, retry allowed –** The PSU may enter a new OTP and retry the verification.
      - **OTP invalid, maximum number of attempts exceeded –** The authentication flow is aborted. This is a terminal state. The TPP should inform the PSU accordingly.

6. The code is exchanged for an access token and a refresh token.

9.4 Request headers

Name
Client-Id (required)
X-Request-Id (required)
PSU-IP-Address (required)
PSU-Channel
PSU-Device-ID
PSU-User-Agent
PSU-Referring-Domain

9.5 Authentication states and response types

The authentication endpoints return the shared authentication response models. See 11 Response types for decoupled authentication and signing for each response type, its example JSON, and the next step. Note the authentication-specific items: the state property on OauthCode, and the Kyc_NotAnswered and EConditions_NotApproved abort reasons.

10 Decoupled SCA approach for signing PIS operations

A redirect and a decoupled SCA approach are offered for signing of PIS operations. In both cases the actual signing by the PSU is done using the Swedish BankID application.

10.1 Authentication methods

The following BankID-based authentication methods are supported:

Authentication method	Intended use
BankID on file	For TPP applications running on a computer
Mobile BankID on the same device	For TPP applications running on a mobile device
Mobile BankID on another device	For TPP applications running on a computer or mobile device

The TPP selects redirect or decoupled SCA approach with the request header TPP-Redirect-Preferred or TPP-Decoupled-Preferred when calling the authorization/cancellation endpoints.

The TPP is responsible for:

- Providing a graphical user interface (GUI) for the PSU
- Launching the BankID application when required
- Monitoring the authentication status until the process is completed successfully or is aborted

Throughout this chapter, “payment” also includes payment cancellation unless explicitly stated otherwise.

10.2 Decoupled authentication endpoints

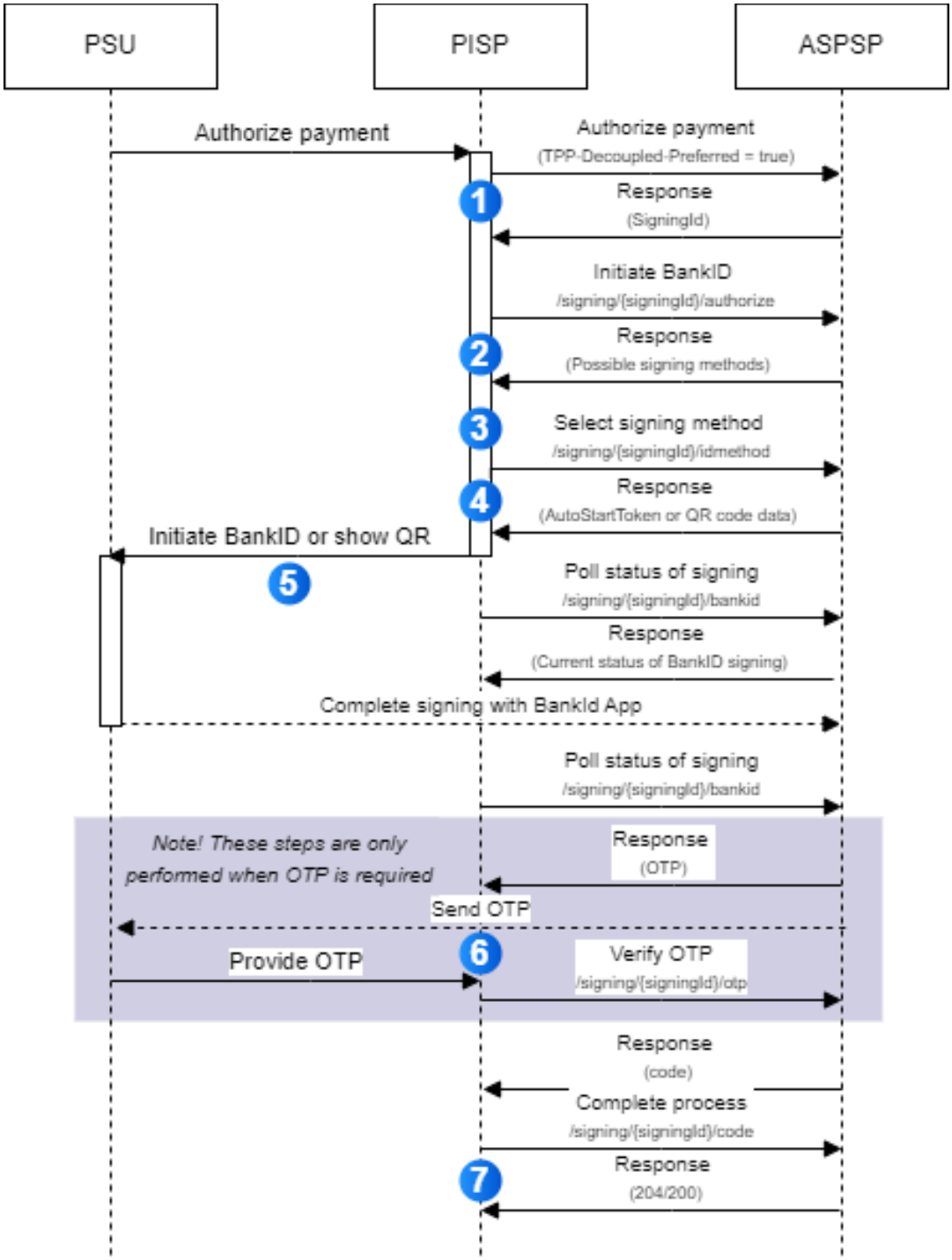
The signing service is exposed as a separate product, Skandia Open Banking API - PIS Signing Services, also requiring a subscription. Request and response details are available in the developer portal, https://developer.skandia.se/open-banking/core-bank

The following endpoints are exposed:

- **GET /signing/{signingId}/authorize**: Retrieve available signing methods
- **POST /signing/{signingId}/idmethod:** Select signing method
- **GET /signing/{signingId}/bankid:** Retrieve signing status
- **POST /signing/{signingId}/otp:** Verify OTP
- **DELETE /signing/{signingId}:** Cancel signing
- **PATCH /signing/{signingId}/code:** Complete payment after successful signing

10.3 The decoupled signing flow

The decoupled signing flow is described in general terms in the below diagram. Depending on what signing method is selected and on whether OTP is required or not, the process may differ a bit. Refer to the description of the steps for more details.



1. The TPP initiates the signing flow by calling the authorization endpoint with the TPP-Decoupled-Preferred header set to true. The response contains the path to the first decoupled endpoint and a signingId that must be included in subsequent decoupled requests.
2. The TPP calls the returned endpoint to retrieve the available signing methods.
3. The TPP determines whether the application is running on a desktop or mobile device and presents the applicable signing methods in the GUI.
4. The PSU selects a signing method.
   - When **Mobile BankID on Another Device** is selected:
     a. The response contains QR code data.
     b. The TPP generates and displays the QR code in the GUI.
     c. The PSU opens the BankID application on another device and scans the QR code.

   d. The PSU enters their security code in the BankID application.
- When **BankID on File** or **Mobile BankID on the Same Device** is selected:
   a. The response contains information required to start the BankID application (autoStartToken).
   b. The TPP launches the BankID application according to the BankID integration guidelines.
   c. The PSU enters their security code in the BankID application.

5. As soon as the QR code is displayed or the BankID application has been launched, the TPP should poll the signing status endpoint once per second while the BankID order remains pending. The response to the status check may require different actions depending on the response type. Refer to section 10.5 Signing states and response types.

6. OTP verification (if required)
   a. TPP shows GUI where PSU can enter OTP.
   b. PSU enters OTP received in SMS.
   c. The TPP submits the OTP for validation.
   d. The TPP processes the validation response
      - **OTP valid** – If the OTP is valid and all additional validations, including KYC checks, have been completed successfully, the authentication flow is complete. An OAuth authorization code is returned. The TPP can proceed to exchange the authorization code for an access token and refresh token.
      - **OTP invalid, retry allowed** – The PSU may enter a new OTP and retry the verification.
      - **OTP invalid, maximum number of attempts exceeded** – The authentication flow is aborted. This is a terminal state. The TPP should inform the PSU accordingly.

7. The TPP completes the signing flow by calling PATCH /signing/{signingId}/code with the received code. The payment is then processed and may either succeed or fail, for example due to insufficient funds.

Note! It's not possible to restart or rerun the signing flow. Neither when it is aborted by Skandia nor when cancelled by the PSU.

10.4 Request headers

Name
Client-Id (required)
X-Request-Id (required)
PSU-IP-Address (required)
PSU-Channel
PSU-Device-ID
PSU-User-Agent
PSU-Referring-Domain

10.5 Signing states and response types

The signing endpoints return the shared authentication response models. See 11 Response types for decoupled authentication and signing for each response type, its example JSON, and the next step. Note the signing-specific OauthCode next step: notify about successful signing (PATCH code) to complete the payment / payment cancellation.

11 Response types for decoupled authentication and signing

The decoupled authentication and signing endpoints return response models that share a common pattern. IdMethodResponse, BankIdResponse and OneTimePasswordResponse are all projections of a single model that is the union of every property used across all response types. For any given response, only the properties relevant to its type are populated.

The id property identifies the response type (for example IdMethods, BankId_AutoStart, BankId_QRCode, BankId_Status, Otp, OauthCode or IdentifyAborted) and determines which action the TPP should take next. Because the same response type can be returned by several endpoints — and by both the authentication and signing flows — the types are documented once here rather than per endpoint.

Throughout this appendix, status endpoint refers to the authentication status endpoint or the signing status endpoint depending on the flow in progress. Differences between the two flows are marked Authentication only or Signing only.

11.1 A note on polling

After receiving BankId_AutoStart or BankId_QRCode, the TPP should poll the status endpoint once per second until either OauthCode or IdentifyAborted is returned.

11.2 IdMethods

Returns the list of available authentication methods. Response from the authorize endpoint.

```
{
  "id": "IdMethods",
  "identifySessionId": "7478a48c-35f9-4f7f-b9c9-1b2b3b9ad581",
  "availableMethods": [
    "BankIdSameDevice",
    "MobiltBankIdSameDevice",
    "MobiltBankIdOtherDevicePnr"
  ]
}
```

Next step: Present the available methods to the PSU and submit the selected method to the idmethod endpoint. When MobiltBankIdOtherDevicePnr is selected, the PSU's Swedish personal identity number must be included in the request.

11.3 BankId_AutoStart

The BankID application should be launched using the supplied autoStartToken.

The launch mechanism depends on the platform and application type (web, native mobile application, or desktop application). The TPP is responsible for handling application launch, start failures, and cases where the BankID application is not installed. For implementation guidance, see the BankID documentation: https://developers.bankid.com/getting-started/frontend/autostart

```
{
  "id": "BankId_AutoStart",
  "autoStartToken": "00000000-aaaa-0000-aaaa-000000000001"
}
```

Next step: Launch the BankID application and begin polling the status endpoint.

11.4 BankId_QRCode

Generate a QR code from the supplied qrCodeText and display it to the PSU. For recommendations regarding QR code presentation, see: https://developers.bankid.com/getting-started/frontend/qr-code

```
{
  "id": "BankId_QRCode",
  "qrCodeText": "bankid.00000000-aaaa-0000-aaaa-
000000000002.0.09e65831d6d269e50eb5e3db2ae09a35fb7d1eb98eeada3e944e5dd407770fcc"
}
```


Next step: Generate and display the QR code and begin polling the status endpoint (after 1 second).

11.5 BankId_Status

The BankID order is still pending. The statusCode corresponds to a BankID hint code and should be used to present an appropriate status message to the PSU.
For details regarding hint codes, recommended user messages and required actions, see:

- https://developers.bankid.com/api-references/auth--sign/collect
- https://developers.bankid.com/resources/user-messages


```
{
  "id": "BankId_Status",
  "statusCode": "OutstandingTransaction"
}
```

```
{
  "id": "BankId_Status",
  "statusCode": "UserSign"
}
```


Next step: Display an appropriate status message and continue polling the status endpoint (after 1 second).

11.6 Otp

An SMS containing a one-time password has been sent to the PSU's registered mobile phone number. The TPP should provide a field where the PSU can enter the OTP (6 digits, 100000–999999).


```
{
  "id": "Otp"
}
```


Next step: Submit the entered OTP to the validation endpoint.

If the OTP has a valid format but is incorrect, and additional attempts are allowed, the OTP validation endpoint returns:


```
{
  "id": "Otp",
  "statusCode": "otp_invalid"
}
```


The PSU should be allowed to enter a new OTP and retry the verification.

11.7 OauthCode

Authentication and all required validations have completed successfully.

Authentication: required validations include KYC checks. The response includes state when it was supplied in the authorize request.


```
{
  "id": "OauthCode",
  "code": "kYmW0O0ST6w4BWQ9H2TjJqIo3JDuntfIEUkAAAAC",
  "state": "mystate"
}
```


Signing: returned once the PSU has successfully signed and all other checks (such as OTP verification) are fulfilled.


```
{
  "id": "OauthCode",
  "code": "kYmW0O0ST6w4BWQ9H2TjJqIo3JDuntfIEUkAAAAC"
}
```


Next step:

- **Authentication only:** If state was sent in the authorize request, verify it has the same value. The code is then required when requesting the access token (see 6.2.2 Request for access token).
- **Signing only:** Notify about successful signing (PATCH code). The code from the response must be included in the request. The payment / payment cancellation is then completed.

11.8 IdentifyAborted

The authentication flow has been terminated. This is a final state — calling an operation in this state results in an error. The response is 200/OK with a body containing a machine-readable reason and a Swedish user-facing message (reasonDescription) that may be displayed to the PSU.

Category	Reason	Description
BankID	`BankID_AlreadyInProgress`	A BankID order could not be created because an order for this user is already in progress. See https://developers.bankid.com/api-references/errors.
BankID	`BankID_ExpiredTransaction`, `BankID_CertificateErr`, `BankID_UserCancel`, etc.	BankID authentication failed. See the BankID documentation for an exhaustive list, descriptions and actions: https://developers.bankid.com/api-references/auth--sign/collect.
BankID	`BankID_QRTimeout`	QR code was not scanned within the time limit.
OTP	`Otp_SecureMobileNumberMissing`	No mobile number registered for OTP delivery. PSU needs to contact Skandia's customer service.
OTP	`Otp_MaxAttemptsExceeded`	Invalid OTP given too many times.
PIN	`Policy_Pin_Change`	PSU must log in to www.skandia.se and change their PIN.
KYC	`Kyc_NotAnswered`	PSU must complete KYC requirements. **(Authentication only)**
EConditions	`EConditions_NotApproved`	PSU must approve the e-service terms and conditions. **(Authentication only)**
Miscellaneous	`Cancel`	Confirmation response to a successful cancel request.
Miscellaneous	`Unknown_Reason`	Technical error. Contact Skandia's customer service if the error persists.

For BankID-specific failure reasons and recommended actions, refer to the official BankID documentation: https://developers.bankid.com/api-references/auth--sign/collect

Response examples


```
{
  "id": "IdentifyAborted",
  "reason": "BankID_AlreadyInProgress",
```

```
  "reasonDescription": "En identifiering eller underskrift för det här personnumret är redan påbörjad. Försök igen."
}
```

```
{
  "id": "IdentifyAborted",
  "reason": "BankID_CertificateErr",
  "reasonDescription": "Det BankID du försöker använda är för gammalt eller spärrat. Använd ett annat BankID eller hämta ett nytt."
}
```

```
{
  "id": "IdentifyAborted",
  "reason": "BankID_UserCancel",
  "reasonDescription": "Åtgärden avbruten."
}
```

```
{
  "id": "IdentifyAborted",
  "reason": "BankID_QRTimeout",
  "reasonDescription": "Giltighetstiden för QR-koden för att starta BankID har gått ut."
}
```

```
{
  "id": "IdentifyAborted",
  "reason": "Otp_SecureMobileNumberMissing",
  "reasonDescription": "Vi har inget mobilnummer för engångskoder till dig. Kontakta Skandias kundservice för att registrera ditt mobilnummer så att vi kan skicka SMS med engångskoder till dig."
}
```

```
{
  "id": "IdentifyAborted",
  "reason": "Policy_Pin_Change",
  "reasonDescription": "Du behöver byta din PIN-kod. Det kan du göra på https://www.skandia.se/"
}
```

```
{
  "id": "IdentifyAborted",
  "reason": "Cancel",
  "reasonDescription": "Identiferingen/signeringen avbröts."
}
```

```
{
```

```
  "id": "IdentifyAborted",
  "reason": "Unknown_Reason",
  "reasonDescription": "Ett tekniskt fel har uppstått. Kontakta Skandias
kundservice om felet kvarstår."
}
```

12 Fallback Solution

Skandia allows TPPs to make use of the fallback solution (online bank) when the special interface does not meet all the requirements or if functions are temporarily out of order or have slow response times. You do not have to wait for information from us about incidents or deficiencies, you can use the backup solution when deficiencies occur. As a TPP, you are asked to report your use of the backup solution to us. Send an e-mail to openbanking@skandia.se. The backup solution consists of the bank's customer interface via so-called screen scraping or overlay.

The fallback solution will let you continue to provide services for account information and payment initiation to customers without any interruption. You can identify yourself in the fallback solution by the usage of signed headers, using the “Signing HTTP messages”-standard that is defined in “draft-cavage-http-signatures-10” by the HTTP Networking Group (https://tools.ietf.org/html/draft-cavage-http-signatures-10).

The solution requires you to provide an extra set of http headers to the first request towards login.skandia.se. The headers include:

- **Tpp-Signature-Certificate**, your eIDAS QSEAL certificate that is to be used for signing the message. The certificate should be Base64-encoded in DER format (PEM format without the encapsulations BEGIN CERTIFICATE and END CERTIFICATE).
- **Signature**, a signature of the request as described in “draft-cavage-http-signatures-10”.
  - *keyId* is always assumed to be the certificate provided in header *Tpp-Signature-Certificate*.
  - *algorithm* is used to specify the digital signature algorithm that was used when generating the signature. Allowed algorithms include SHA-256 and SHA-512.
  - *headers* should include at least ‘(request-target)’, ‘host’ and ‘date’.
  - *signature* contains the actual signature in base64 encoding.

Example:

GET /?client_id=i_web_individual_short&redirect_uri=https://secure.skandia.se/overview/signin-Skandia&response_type=… HTTP/1.1
Host: login.skandia.se
Date: Sat, 14 Sep 2019 00:00:00 GMT
Tpp-Signature-Certificate: MIIHoDCCBoigAwIBAgIKBIlyT7rSlPwDBDANBgkqhkiG9w0BAQ….
Signature: keyId="tpp-signature-certificate",algorithm="rsa-sha256",headers="(request-target) host date", signature="Base64(RSA-SHA256(**signing_string**))"

where **signing_string** is (\n is the ASCII-value for newline):
(request-target): get
/?client_id=i_web_individual_short&redirect_uri=https://secure.skandia.se/overview/signin-Skandia&response_type=…\n
host: login.skandia.se\n
date: Sat, 14 Sep 2019 00:00:00 GMT

If you have questions about this don’t hesitate to contact us at openbanking@skandia.se.