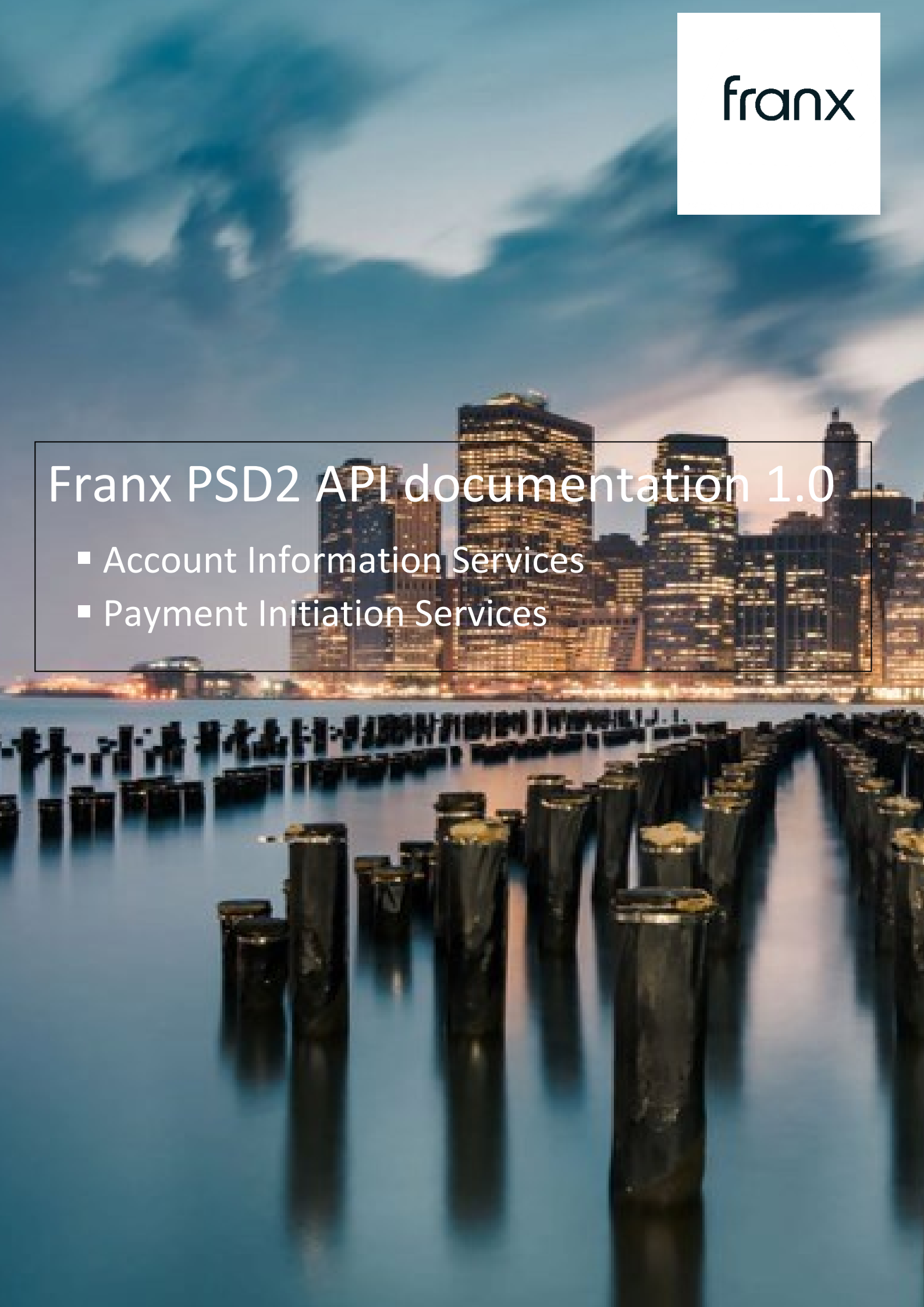

Franx PSD2 API documentation 1.0

- Account Information Services
- Payment Initiation Services

Table of Content

1. Introduction

The Revised Payment Services Directive (PSD2) is European regulation designed to make managing cash-flows and making payments online more convenient for consumers and business. It aims to promote innovation by opening up to third parties – including companies that are not banks, often referred to as TPPs – and increase competition. PSD2 puts people and businesses in control of who can access their bank account information, gives them more options on how they manage their money and make payments online.

We welcome PSD2 and have been working closely with our development partners and TPP's to open up our banking platform. We look forward to offering our customers more enhanced products and better way of managing their money. We invite developers to partner with us to create innovative solutions using our Application Programming Interfaces (APIs).

This new environment completes the Franx product offering perfectly and will enable our corporates clients to achieve their business goals.

2. Access

2.2 Strong Customer Authentication Method

Franx will make use of authorization code Oauth2 flow for the authentication of its client. Exact information on authorization code flow can be found at:
https://tools.ietf.org/html/rfc6749#section-1.3.1

2.1 TPP User Access (Third party provider access)

Once a TPP has authenticated, using hybrid or implicit flow and given permission to accounts, it is able to create consents request.

The consent screen will display all available accounts the PSU has access to and allows him to choose which accounts he want to give the TPP access to. By doing this the TPP can have an access to all or a subset of the accounts the PSU have access to, not more. Then when the PSD2 API is called, the token and TPP permissions are verified to ensure only information exposed which the client provided a consent for in the interface.

3. Test data

To be completed

4. Overview of Endpoints & Sequence diagrams

The Franx API's are developed based on the Berling Group standard, NextGenPSD2 version 1.3.

Supported endpoints by Franx:

Endpoint Name	Account Services	Payment initiation services	Authorization
Accounts	List of available accounts		
accounts/{account-id}	Account details		
accounts/{account-id}/balances	Account details		
accounts/{account-id}/transactions	Overview of account movements		
consents			
consents/{consentId}			
onsents/{consentId}/status			
consents/{consentId}/authorisations			
consents/{consentId}/authorisations			
payments/{payment-product}		Initiate a payment	
payments/{payment-product}/{paymentId}		Retrieve details of initiated payment	
payments/{payment-product}/{paymentId}/status		Retrieve payment status	

For the sequence diagram please see document :

- Sequence diagram PSD2 Franx_1.0

5. PSD2 API's

The PSD2 API's are RESTful services with swagger documentation available at:

5.1 Scopes

Scopes are used to give granular access to certain endpoints.

Resource	Scope	Description
Account	rAccounts	Read only Access to summary account information
Transaction	rTransactions	Read only Access to transactions
Balance	rBalances	Read access to balances related information
Transfer	wTransfer	Transfer of money between accounts
Consent	rConsents	Read access to consent related information.
Consent	wConsents	Write access to consent related information.
Consent	dConsents	Delete access to consent related information.

5.2 Security

The endpoints are secured using a **JWT** issued by the Franx **Identity Server**. The access token must be provided in the header of every request to the API as a bearer token.

Example:

Calling an API endpoint

GET https:// psd2-public-t.franx.link:50115/account/api/v1/accounts
Authorization: Bearer [ACCESS_TOKEN]
Consent-ID: [consentID obtained]
X-Request-ID: 87b46377-604a-4e63-b103-ffe11e8f7858

5.4 Error Handling

API-ID's are returned whenever an error on that API occurs. The api-id will be combined with an error code. Clients are able to identify the failing API by reading the api-id which is returned in an HTTP header.
Resource endpoints may respond with an error. In those cases, an appropriate HTTP status is returned in conjunction with an error message. HTTP status codes are well defined but do not always indicate the exact cause for an error. Resource endpoints will also include an error message but these have to be parsed by clients to extract the information about the error cause.

The following are the HTTP response codes for the different HTTP methods – across all Account Info API endpoints

Situation	HTTP Status	Description	POST	GET	DELETE
Query completed successfully	200 OK	PUT, GET Response Codes This return code is permitted if a request was repeated due to a time-out. The response in that might be either a 200 or 201 code depending on the ASPSP implementation. The POST for a Funds request will also return 200 since it does not create a new resource. DELETE Response Code where a payment resource has been cancelled successfully and no further cancellation authorization is required.	✗	✓	✗
Normal execution where a resource is created. The request has succeeded.	201 Created	POST response code where Payment Initiation or Consent Request was correctly performed.	✓	✗	✗
Delete operation accepted but requires further authorization.	202 Accepted	DELETE response code, where a payment resource can be cancelled in general, but where a cancellation authorization is needed in addition.	✗	✗	✓
Delete operation completed successfully	204 No Content	DELETE response code where a consent resource was successfully deleted. The code indicates that the request was performed, but no content was returned.	✗	✗	✓
Request has malformed, missing or non-compliant JSON body or URL parameters	400 Bad Request	Validation error occurred. This code will cover malformed syntax in request or incorrect data in payload. The operation was refused access.	✓	✗	✗
Authorization header missing or invalid token	401 Unauthorized	The TPP or the PSU is not correctly authorized to perform the request. Retry the request with correct authentication information.	✓	✓	✓

		The operation was refused access.			
Token invalid, has incorrect scope or a security policy was violated	403 Forbidden	Returned if the resource that was referenced in the path exists but cannot be accessed by the TPP or the PSU. This code should only be used for non-sensitive id references as it will reveal that the resource exists even though it cannot be accessed.	✅	✅	✅
Token invalid, has incorrect scope, a security policy was violated, resource does not exist or resource cannot be referenced	404 Not found	Returned if the resource or endpoint that was referenced in the path does not exist or cannot be referenced by the TPP or the PSU. When in doubt if a specific id in the path is sensitive or not, use the HTTP response code 404 instead of the HTTP response code 403.	✅	✅	✅
PUT/POST/DELETE/GET is not supported by this endpoint	405 Method Not Allowed	This code is only sent when the HTTP method (PUT, POST, DELETE, GET etc.) is not supported on a specific endpoint. It has nothing to do with the consent, payment or account information data model.	✅	✅	✅
	406 Not Acceptable	The ASPSP cannot generate the content that the TPP specified in the Accept header.	✅	✅	✅
A request is taking too long	408 Request Timeout	The server is still working correctly, but an individual request has timed out.	✅	✅	✅
XML is requested where only JSON is supported	415 Unsupported Media Type	The TPP has supplied a media type which the ASPSP does not support.	✅	✅	✅
		Throttling is a NFR.			
The operation was refused as too many requests have been made within a certain time frame	429 Too Many Requests	The TPP has exceeded the number of requests allowed by the consent or by the RTS.	✅	✅	✅
Something went wrong on the API gateway or micro-service	500 Internal Server Error	The operation failed.	✅	✅	✅

Service is temporarily unavailable	503 Service Unavailable	The ASPSP server is currently unavailable. Generally, this is a temporary state.	✓	✓	✓

Example:

Error Response

```
HTTP/1.1 401 Unauthorized
Transfer-Encoding: chunked
Content-Type: application/problem+json
WWW-Authenticate: Bearer
X-Request-ID: d8e59ee3-9495-44aa-9d6d-ce3f554d4e91
X-Rate-Limit-Limit: 10s
X-Rate-Limit-Remaining: 4997
X-Rate-Limit-Reset: 2019-03-07T21:09:20.9416799Z
WWW-Authenticate: NTLM
X-Powered-By: ASP.NET
Date: Thu, 07 Mar 2019 21:09:11 GMT

{"error": {
  "type": "https://berlingroup.com/error-codes/TOKEN_INVALID",
  "title": "invalid_request",
  "detail": "The token is missing or it has been provided more than once",
  "code": "TOKEN_INVALID"
}}
```

6. Access to API

Redirect to authorisation - obtaining access token.

https://psd2-public-t.franx.link:50115/connect/authorize?client_id=[CLIENT_ID]&response_type=code&scope=PSD2%20rAccounts%20rTransactions&redirect_uri=[REDIRECT_URI]&response_mode=form_post

For more information please below link on to obtain the access token.

http://docs.identityserver.io/en/latest/endpoints/authorize.html

7.1 Account Information Service (AIS)

GET /accounts

Accounts Endpoint

```
GET https:// psd2-public-t.franx.link:50115/account/api/v1/accounts HTTP/1.1
Accept-Encoding: gzip,deflate
Authorization: Bearer [ACCESS_TOKEN]
Consent-ID: [ consent_ID ]
X-Request-ID: 87b46377-604a-4e63-b103-ffe11e8f7858
```

GET /accounts/{account-id}

GET https:// psd2-public-t.franx.link:50115/account/api/v1/accounts/[IBAN]
Authorization: Bearer [ACCESS_TOKEN]
Consent-ID: [Consent_ID]
X-Request-ID: 47e3444b-2646-48d6-b98e-e3a361dfe21a

GET /accounts/{account-id}/balances

GET https:// psd2-public-t.franx.link:50115/account/api/v1/accounts/[IBAN] /balances
Authorization: Bearer [ACCESS_TOKEN]
Consent-ID: [Consent ID]
X-Request-ID: f2e28322-b0cb-4713-a3eb-b8e64cee6ad3

GET /accounts/{account-id}/transactions

GET https:// psd2-public-t.franx.link:50115/account/api/v1/accounts/[IBAN]/transactions
Authorization: Bearer [ACCESS_TOKEN]
Consent-ID: [consent ID]
X-Request-ID: fc3bf47e-68c6-45a4-a9d8-0c6bb7d85d80

POST /consents

POST https:// psd2-public-t.franx.link:50115/account/api/v1/consents
Content-Type: application/json
Authorization: Bearer [ACCESS_TOKEN]
X-Request-ID: 7f898b24-77ab-4e0d-869c-f620ab0ac3e3
Content-Length: 185

```
{
  "access": {
    "allPsd2": "allAccounts"
  },
  "recurringIndicator": true,
  "validUntil": "2019-04-07T20:56:24.821Z",
  "frequencyPerDay": 4,
  "combinedServiceIndicator": false
}
```

GET /consents/{consentId}

GET https: psd2-public-t.franx.link:50115/account/api/v1/consents/123
Authorization: Bearer [ACCESS_TOKEN]
X-Request-ID: 496a31dd-dff2-4708-a19a-54d81f99f747

DELETE /consents/{consentId}

DELETE https:// psd2-public-t.franx.link:50115/account/api/v1/consents/123 HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: application/json
Authorization: Bearer [ACCESS_TOKEN]
X-Request-ID: 1d78f5a7-de57-4fb9-97f3-1b20485ceb34

Content-Length: 0
Host: localhost:55550
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

GET /consents/{consentId}/status

GET https:// psd2-public-t.franx.link:50115/account/api/v1/consents/123/status
Authorization: Bearer [ACCESS_TOKEN]
X-Request-ID: 618bdd40-ed9a-4c90-bc30-ec8d8bd4792b

POST /consents/{consentId}/authorizations

POST https:// psd2-public-t.franx.link:50115/account/api/v1/consents/123/authorisations
Content-Type: application/json
Authorization: Bearer [ACCESS_TOKEN]
X-Request-ID: 77df78db-f789-48fd-8214-fcdd53e4f537
{

7.2 Payment Initiation Service (PIS)

POST /payments/{payment-product}

POST https:// psd2-public-t.franx.link:50115/payment/api/v1/payments/sepa-credit-transfers
X-Request-ID: 5d4074ad-2f3c-4ef0-a343-3eda515dcb4c
PSU-IP-Address: 1
Content-Type: application/json
Authorization: Bearer [ACCESS_TOKEN]

```
{
  "endToEndIdentification": "123",
  "debtorAccount": {
    "currency": "EUR",
    "accountReference": {
      "bban": "5000005603"
    }
  },
  "instructedAmount": {
    "currency": "EUR",
    "amount": "20"
  },
  "creditorAccount": {
    "currency": "EUR",
    "accountReference": {
      "iban": "NL34RABO0323018074"
    }
  },
  "creditorAgent": "AAAADEBBXXX",
  "creditorName": "Name of creditor",
  "creditorAddress": {
    "street": "Creditor street",
    "buildingNumber": "12",
    "city": "City",
    "postalCode": "23000",
    "country": "NL"
  },
```

```
  "remittanceInformationUnstructured": "some comment"
}
```

GET /payments/{payment-product}/{paymentId}

GET https:// psd2-public-t.franx.link:50115/payment/api/v1/payments/sepa-credit-transfers/10713
X-Request-ID: 6ba7e49c-cd3c-458a-b1aa-29398b7826a4
Authorization: Bearer [ACCESS_TOKEN]

GET /payments/{payment-product}/{paymentId}/status

GET https:// psd2-public-t.franx.link:50115/payment/api/v1/payments/sepa-credit-transfers/10713/status
X-Request-ID: 522af217-06d9-453f-b8f8-ea9a0f717e3c
Authorization: Bearer [ACCESS_TOKEN]