Digital Payments
for a Trusted World

WORLDLINE

Open Banking API v3 for iDEAL – Implementation Guide

Status	Published
Author:	Worldline
Document date:	19 February 2024
Classification:	Confidential
Version:	1.5

Version history

Version no.	Version date	Status	Edited by	Most important edit(s)
1.0	13-4-2022	Published	Worldline	Initial Version
1.1	30-06-2022	Published	Worldline	Moved Fast Checkout description to Chapter 2 Process reviewed comments
1.2	14-12-2022	Published	Worldline	Rearranged Chapters Processed additional comments Amended pictures with better resolution
1.3	15-03-2023	Published	Worldline	Extended Security Chapter Updated Requests and Responses Updated Sequence Diagrams
1.4	21-04-2023	Published	Worldline	Updated Requests and Responses
1.5	19-2-2024	Published	Worldline	Chapters are organised by Flow type Added tables covering fields relevant for iDEAL 2.0 API requests and responses

Table of contents

1 Introduction

This document describes the components from the Open Banking API version 3 which are used to initiate iDEAL 2.0 payments. Some sections are based on the iDEAL 2.0 implementation guide from Currence, the iDEAL scheme owner.

This document is meant to be used in conjunction with the Open Banking API Swagger Files as well as the iDEAL 2.0 Getting Started Guide.

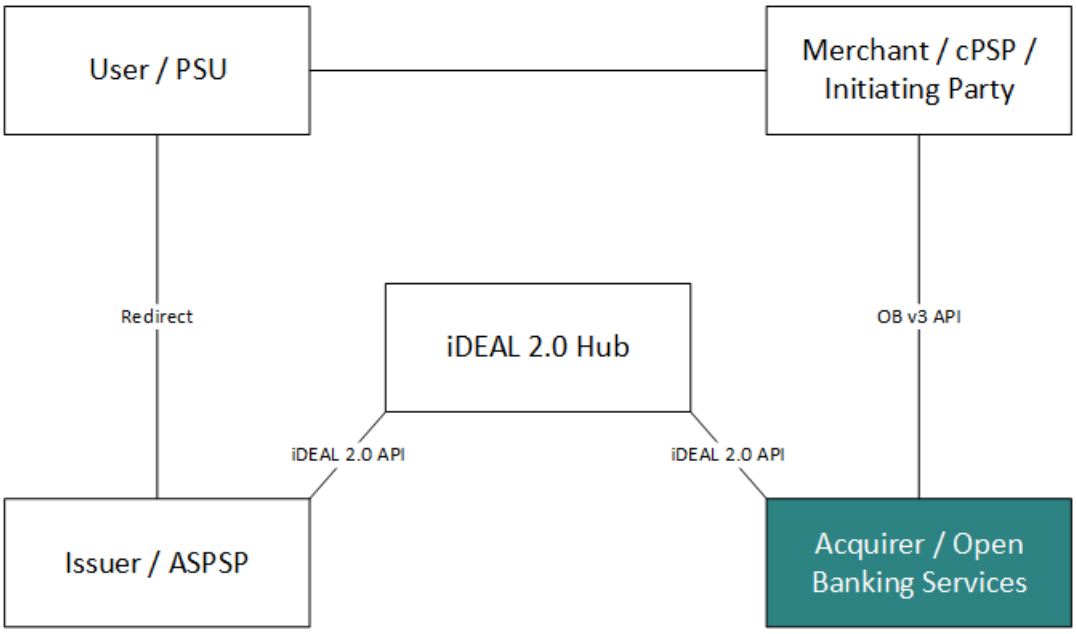
1.1 Terminology

The terminology used in this document is based on the iDEAL documentation of Currence. The Open Banking API Swagger files use some different terms based on the Payment Service Directive 2.

Term	Equivalent term in Open Banking API swagger	Description
Open Banking Service	-	Open Banking Service is an intermediary between multiple Merchants and Issuers and it provides an interface which is used by the Merchant. The 'Open Banking Service' refers to the Worldline provided software (described in this document), which handles the routing of the iDEAL 2.0 payments.
iDEAL HUB	-	The iDEAL hub is a solution owned by Currence which provides a unified iDEAL experience. It is connected to the Issuers which provide the iDEAL 2.0 product.
User	PSU	The individual or Business using the iDEAL Scheme to make an iDEAL Payment through the Banking Environment of its Issuer, also commonly referred to as the debtor. In the Payment Service Directive 2 this role is know as the 'Payment Service User' (PSU). Which is identified with a PsuId.
User Token/ Debtor token	Debtor token	The token as provided by Currence to a Merchant to identify an iDEAL Profile, which is used to exchange the Users preferences (i.e. preferred IBAN) for the iDEAL Transactions with the Merchant. The Open Banking Service links the Debtor token to a PsuId. The PsuId is the identifier for the User which should be used in the Open Banking Service
Merchant / cPSP	Initiating Party	The Merchant contracts the Open Banking Service for the iDEAL 2.0, and sends an iDEAL payment request to the Open Banking Service on behalf of a User.
Issuer	ASPSP	The Issuer holds the account of the User. In the Payment Service Directive 2 this role is know as the 'Account Servicing Payment Service Provider (ASPSP)
MSP Portal	-	The Multi Service Platform is the GUI used by Merchants and Back office Admin from the Acquirer to manage the Merchant Subscriptions and view the Transaction Details. Here, the Merchants can configure their iDEAL Subscriptions and relevant fields.

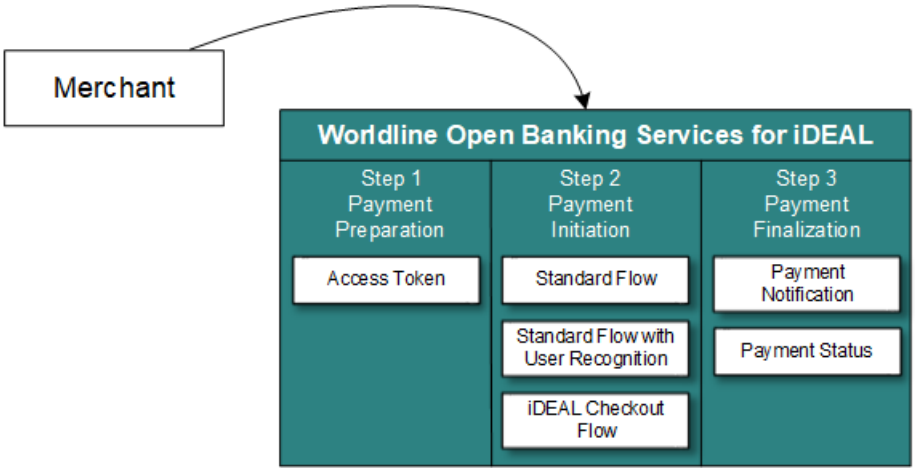
1.2 Ecosystem overview

The Open Banking Service, marked in green, is provided by Worldline.



1.3 Steps to Create a successful iDEAL Payment

In order to create a successful iDEAL Payment, the steps below must be followed. Worldline prepared a set of functions to support these steps (the white boxes within the green part in the diagram below). Each of those functions have a chapter with more explanation. Sequence diagrams with more details on the 3 iDEAL payment flows can be found on the Payment Initiation chapters.



1.3.1 Step 1 : Payment preparation

The first step is the preparation of the payment. Here the Merchant should take care of the Security Requirements and ensure they retrieve an access token from the Authorization API endpoint: POST /token. This token will be used to authenticate subsequent requests. The token will be valid for 60 minutes from the time that it was generated.

1.3.2 Step 2 : Payment Initiation

There are 3 different flows offered in iDEAL 2.0 to make payments :

- **Standard flow**
  - This is a standard flow via the POST /payments API. The User is redirected to the iDEAL Hub where the Issuer can be selected.
  - The iDEAL Hub will try to streamline the Issuer selection by using cookies to recognize a Users iDEAL profile.

Or

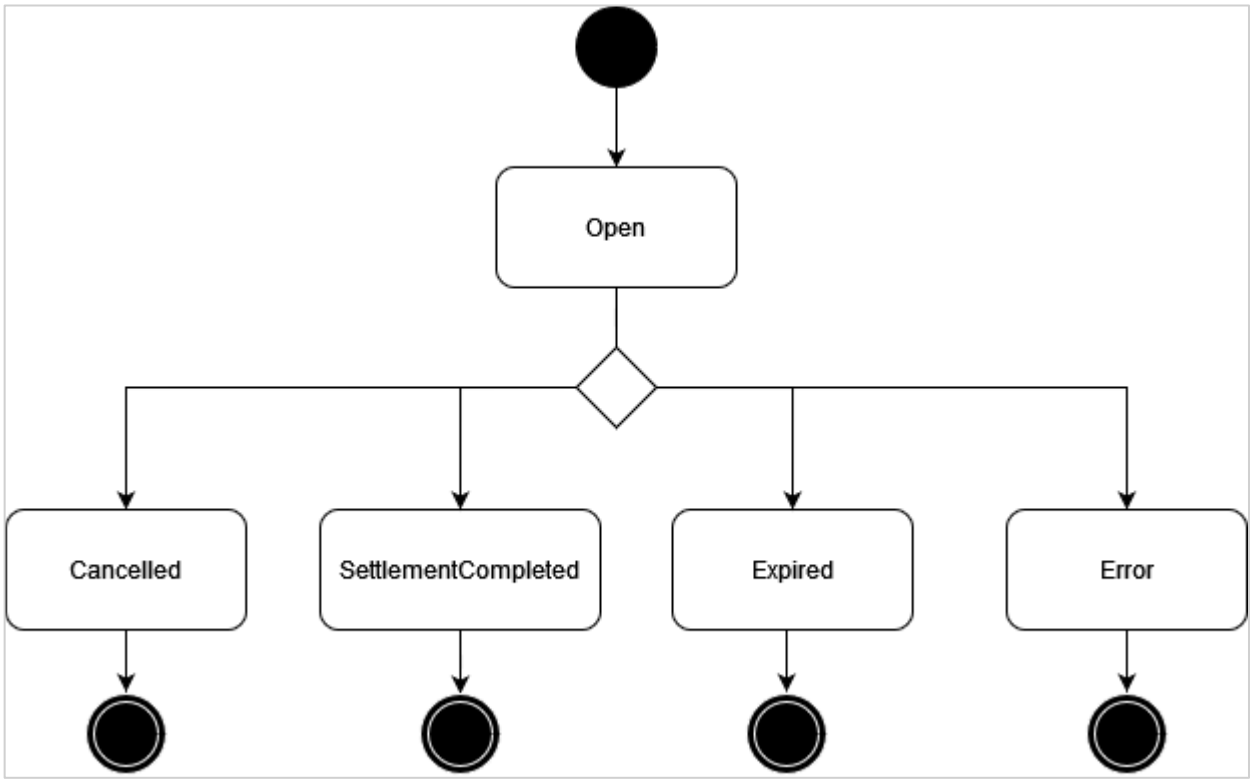
- **Standard flow with User recognition**
  - This flow allows the Merchant to show a pre selected Issuer and masked IBAN on the checkout page for a returning User.
  - To use this flow, additional fields (such as PsuId, DebtorTokenCallbackURL, UseDebtorToken) are to be sent in in the POST /payments API.
  - User preferences may be requested using the GET /preferences API.

Or

- **iDEAL Checkout Flow**
  - This flow allows the User to share it's address and contact details stored in it's iDEAL profile with the Merchant. This means that Users, for example, do not have to enter their address details each time they shop at a different Merchant
  - To use this flow additional fields (Flowtype, ExpectedCheckoutData) are to be sent in the POST /payments API

1.3.3 Step 3 : Payment Finalization

Payment finalization is the final step to receive the status of the payment. Please refer to the figure below for all possible statuses.



The status can be retrieved in one of the two ways :

- **(Push) Receive a status notification by receiving a request from POST /status API**
  This is the preferred method of payment finalization as per the Scheme Owner, Currence

And / Or

- **(Pull) Request for payment status via GET /payments/{paymentId}/status**

2 Common Elements for Open Banking v3

2.1 Timezones and Time Formats

The ISODateTime standard is used in the Open Banking v3 API.

- Datetime fields sent towards the Open Banking Service should have a timezone (for example: YYYY-MM-DDThh:mm:ssZ or YYYY-MM-DDThh:mm:ss+02)
- Date fields sent towards the Open Banking Service are always interpreted as being in UTC.

2.2 API Compatibility Policy

All APIs follow the robustness principle: "be conservative in what you do, be liberal in what you accept from others". This means the following cases MUST NOT cause an error on client or server side:

- unrecognized query parameters
- unrecognized headers
- unrecognized fields in body

2.3 Links Section

In the responses to API calls, usually a section called *Links* is included. This section contains endpoints that might be or even must be called by the Merchant for the next steps.

In order to inform the Merchant of the next request required for an authorization, the links section provides the corresponding links. If a redirection of the User is required, the corresponding URL for that redirection is given in the link called *RedirectUrl*.

The URLs given in the links section are provided as complete ones including the domain name and query parameters if required. An example for links with query parameters are paginated responses where the query parameter allows to address the first, next, previous or last page.

2.4 Error Codes

If a request cannot be executed successfully an error response is given. The structure of error responses is the same for all APIs. Additional to the standard HTTP Status a body is provided giving more detailed information on the error. The link is provided in the error response for future use and is currently not filled for any error response.

Body fields	Mult.	Type	Description
Code	1..1	String	The Error code. The yaml file has a list of the codes that can be received
Message	1..1	String	The description of the error. In a human readable form.
Details	0..1	String	Details of the error which could be useful for a developer
Link	0..1	±	
+ Href	1..1	String	URL

2.5 Retries

- **Transaction initiation:** In case of receipt of a non-deterministic failure (timeout or 5xx), parties SHOULD retry the POST /payments API once.
    - When after this retry a second non-deterministic failure (timeout or 5xx) is received, this SHOULD be considered and communicated as a deterministic failure.
- **Transaction callback:** The Open Banking Service will attempt to perform retries of a callback up to 25 minutes if it does not receive a response within 8 seconds and/or receives a different response code than 200 OK, 202 ACCEPTED or 204 NO CONTENT to the initial callback.

3 Security Requirements

There are 3 main types of security mechanisms used in the Open Banking Service.

1. Transport Level Security
2. Access Tokens
3. Digital Signatures

3.1 Transport Level Security

On the transport level, all requests to the Open Banking Service, as well as all requests sent by the Open Banking Service must be encrypted using TLS and be made over HTTPS. TLS 1.3 SHOULD be used; TLS 1.2 may be used. Anything below TLS 1.2 must not be used and will be refused by the Open Banking Service. The TLS authentication method used is one-way, that means in requests to Open Banking Service, the server authenticates itself with its certificate and in case of requests from Open Banking Service to the Merchant, the latter authenticates itself with its certificate. Any connection without TLS encryption, such as plain http will be refused.

3.2 Access Tokens

There are two types of access tokens used for the purpose of Security.

- **Oauth tokens via Authorization API request**
    - These tokens are requested by the Merchant as the preliminary step before making any other API calls towards the Open Banking Service. The Open Banking service validates the Merchant's identity through the Initiating Party ID, Acquirer and Certificate which is uploaded in the MSP Portal.
- **Notification Bearer Tokens**
    - This token is set by the merchant in the MSP Portal under the iDEAL Subscription. This is received by the Merchant in the authorization header of the notification request sent by the Open Banking Service.

Both of these will be explored in further detail below.

3.2.1 Authorization Requests

The first step a Merchant will take to interact with the Open Banking Service will be to retrieve an access token. This is done by a POST /token call via the Authorization API. Doing so facilitates a secure connection between the Merchant and the Open Banking Service.

> Before making the Authorization request, the Merchant will first need to upload their public certificate in their iDEAL subscription of the MSP Portal.Self-signed certificates are allowed. If previously using a certificate for signing iDEAL 1.0 requests, the same certificate can be used also for iDEAL 2.0, which means there is no need to upload another certificate.

3.2.1.1 Retrieve an access token via the Authorization API

Endpoint : POST **{Base Domain}**/xs2a/routingservice/services/authorize/token
Full endpoint in test environment : https://digitalroutingservice.awltest.de/xs2a/routing/services/authorize/token
Production Environment Base Domain: Consult the 'Getting Started Guide as the base domain may differ depending on the Acquirer

To request an access token, the merchant must digitally sign the relevant request headers using the private key associated with the certificate that they have uploaded to the iDEAL MSP Portal.

Upon a successful request, the Open Banking Service provides the merchant with an access token, which must be used in all future API interactions, such as POST /payments or GET /preferences, etc.
This token should be included in the Authorization header of subsequent requests as follows:

```
Authorization:Bearer 4944daeae6c9115a10dafecbfad4a9c
```

This access token allows the Open Banking Service to validate and authorize each API request from the merchant.

3.2.1.2 Signing the authorization request

Steps on signing the Authorization requests are available within the section Security Requirements - Digital Signatures. The Merchant should follow the guidelines in order to properly sign the Authorization requests.

3.2.1.3 Important Fields for creating API call for Authorization API

3.2.1.3.1 Request

Query parameter	Mult.	Type	Description
FormData	1..1	String	To be set to 'client_credentials'

Header fields	Mult.	Type	Description
Content-Type	1..1	String	To identify the media type of the source. If not provided, the Merchant can expect to come across the error "415, Unsupported Media Type"

Authorization	1..1	String	The signature. It contains the header attributes 'app', 'client', 'id' and 'date' signed with the private key of the client.
			The signature will be used to sign the authorization request with the private key which corresponds to the certificate provided for the onboarding.
			Structure
			`Signature keyId="<thumbprint of certificate>", algorithm="SHA256withRSA", headers="app client id date", signature="<signature>"`
			Example
			`Signature keyId="58AF4EC5ADD4C4A3F28D3AEFF60656B2F2xxxxxx", algorithm="SHA256withRSA", headers="app client id date", signature="Abczym2rZF…r5qcvgmA=="`
			Refer to the chapter Security - Digital Signatures for more information

App	1..1	String	The name of the service. Only `IDEAL` is allowed.
Client	1..1	String	The name of the client is an abbreviation of the name of the Acquirer. Please refer to the Getting Started Guide to retrieve this field.
Id	1..1	String	The combination of Initiating Party ID and sub Id. For example if Initiating Party ID is 433 and the sub ID is 5 the ID will be 433:5. This ID is found in the Multiservice Platform GUI. It is referred to as "Dashboard ID", "BC number" or "Platform Merchant ID", depending on the Acquirer.

			Information on how to find this ID is stated in the Getting Started Guide. Please note, in the test environment the Merchant should use the test Merchants which are also provided in the Getting Started Guide. Other Initiating Party IDs will not work. IP=433, subId=5 -> 433:5 IP=434, no subId -> 434
Date	1..1	Date	Should be filled with the current date. The following date formats are supported: 1. EEE MMM dd HH:mm:ss zzz yyyy 2. ISO DATE: for example 2011-12-03T10:15:30+01:00 3. RFC 1123: for example Tue, 3 Jun 2008 11:05:30 GMT

3.2.1.3.2 Response

Body fields	Mult.	Type	Description
access_token	1..1	String	Token to be used in further API calls
token_type	1..1	String	Type of the token: Bearer
expires_in	1..1	Integer	Expiration time in seconds.

3.2.1.4 Example: Authorization

3.2.1.4.1 Request

```
    Address:
https://digitalroutingservice.awltest.de/xs2a/routingservice/services/authorize/token?grant_type=client_c
redentials
    HttpMethod: POST
    Headers: {App=IDEAL, Accept=application/json, Date=2024-02-08T18:31:37.548Z, Authorization=Signature
KeyID="39d8e82bb33e7e2a09cbcb3ef3eab351ee1c5e8f", algorithm="SHA256withRSA", headers="app client id
date", signature=
"r0EqeVMfc4au9fq7ZVhOwjfXbMHkf2GShAH/WIQtjMyjl2swlJss10Y0Fjpqt47aLdeBXpG28mVxzPLjwqNHPvnkNPslepJmHzB3W2A6
YgXcoiUqFFv+pxzIBmohVoqVEGb5QC4rxTBpAleE178hSzIJrxxbhlCQv/dSAcoI5V1P1M3pU9VqM7vmql8XlB/yXOjmayeXVtB0T93Bz
qutHokHBwDjxN8ocKlsX40GcFoFjxov9cKoVjlyMZKyfHRkqZ5u1dUkcN+RFeGgJWCcCcaBydXSicK/elbpq55rAgLCTQF4xFldSDPjcr
RsswMIYMZmGz0tCycjDV225pGPGg==", content-type=application/x-www-form-urlencoded, Id=000081,
Client=Worldline}
```

3.2.1.4.2 Response

```
    ResponseCode: 200
    Headers: {X-Request-ID=2aa0dc88-21dd-4034-a027-1d98123596f1, MessageCreateDateTime=2024-02-
08T18:31:38.385Z, Date=Thu, 08 Feb 2024 18:31:38 GMT, Content-Type=application/json;charset=UTF-8}
    Payload: {
    "access_token": "bb00ffd85e0b619a7a9da34df24346f8b309dae54383e5e5481bada22aaa000e",
    "token_type": "Bearer",
    "expires_in": 3600
```

3.2.1.5 Validity of access tokens

As a default, tokens are valid for 60 minutes (3600 seconds). After 60 minutes have elapsed, the Merchant will need to request for a new token.
There is an overlap period in the last 10 minutes (between 50th minute - 60th minute). This means if the Merchant were to request for a new token between 50th minute to 60th minute, they would receive a new access token. However, for those 10 minutes, the previous token would also still be valid.

3.2.2 Notification Bearer Tokens

Notification Bearer Tokens are access tokens received by the Merchant from the Open Banking Service for Notification requests. Two types of Notification requests :

- Status Notification requests
- Debtor token Callback requests

The Notification Bearer Token can be configured in the MSP Portal under the iDEAL Subscription. This is a static token and does not have a set validity period.

The Open Banking Service will send the notification bearer token in the Authorization header of the Notification request. When receiving the notification request, the Merchant can confirm if the value of the Notification Bearer Token matches what was set in the iDEAL Subscription. This would validate the request received.

3.3 Digital Signatures

In this section we cover the different ways digital signatures are used for Security in the Open Banking API. There are three main uses :

- **Sending signed the authorization requests towards the Open Banking Service**
  - This applies to all Merchants sending requests to the Open Banking Service.
- **Sending other signed requests and responses to the Open Banking Service**
  - This only applies if the Acquirer of the Merchant requires messages to be signed. Please check the Getting Started Guide Acquirer Profile if this applies. If not applicable, this section can be skipped.
- **Receiving signed requests and responses from the Open Banking Service**
  - This refers to the Notification requests that the Merchant receives from the Open Banking Service.
  - This only applies if the Acquirer of the Merchant requires messages to be signed. Please check the Getting Started Guide Acquirer Profile if this applies. If not applicable, this section can be skipped.

3.3.1 Sending signed the authorization requests towards Open Banking Service

On the application level, the Merchant is authenticated and authorized by sending a digitally signed request to the Authorization API endpoint (see Security Requirements - Access Tokens). The signature validation allows to check the authenticity and integrity of the request. This is achieved by applying the "Authorization" scheme as described in https://datatracker.ietf.org/doc/html/draft-cavage-http-signatures-12 and further detailed in below.

In order to generate the string that is signed with a key, the Merchant must use the values of each HTTP header field in the `headers` Signature Parameter, in the order they appear in the `headers` Signature Parameter. The `headers` Signature Parameter is fixed:

```
headers="app client id date"
```

The order of the headers **must be exactly as displayed above**. The header field string is created by concatenating the lowercased header field name followed with an ASCII colon `:`, an ASCII space ` `, and the header field value. Leading and trailing optional whitespace (OWS) in the header field value MUST be omitted. If the value is not the last value, then an ASCII newline `\n` is appended.

Example: For the request

```
POST /authorize/token HTTP/1.1
App: IDEAL
Date: Fri, 25 Mar 2022 20:51:35 GMT
Client: idealClient
Id: 434
```

The concatenated **"String to sign"** would be:

```
app: IDEAL
client: idealClient
id: 434
date: Fri, 25 Mar 2022 20:51:35 GMT
```

The signature algorithm is fixed:

algorithm=”SHA256withRSA”

The ‘keyId’ Signature Parameter is the thumbprint of the public certificate, viewed with the SHA1 algorithm. The private key associated with `keyId` is used to generate a digital signature on the concatenated signature string applying the SHA256withRSA algorithm. The complete Authorization header looks like this:

```
Authorization: Signature keyId="DCAC7209573D506FC56095B8B23E8555A8F38B29", algorithm="SHA256withRSA",
headers="app client id date",
signature="guoLSHgl/zGRujqkDnmaWCL8kgCVnDazqkKu7nWU/uAHrS+M9eQsI8ueB4uWgxyPOnZps3vpNgkW1f4aBsdFYLS0jYeup4
yhCMN6vis2zfMKxUhZFkjELslQkit9Gwc9pqvcyH0IxUnDLbCQwkiYjf6nGbP1YNfoxVXQpfq6i6CbIXCotLfwH2kbkrnSWwAS5skZY77
+znmLDjtP3et2K94C36yPo0EEGqGkQ5xkD7owA7YxzA30xzsvkDvU3hzDzTK5wZmsgVsoyjRvMrokG0HrszUpNTwUtxflukcgs0pH7GuT
+JrIpQ55f1dpzULqxeBggnCvD9DRSuKeTakqlw=="
```

The Merchant must upload the used public certificate in the Open Banking Service GUI so that Open Banking Service is able to validate the signature. If the signature can be validated and the sender has a valid iDEAL subscription, a response containing an access token is returned. This access token can be used in all follow-up iDEAL requests until it expires. After expiration a new access token must be requested via the Authorization API.

3.3.2 Sending signed other requests and responses to the Open Banking Service

Signing requests and responses could be enforced depending on the Acquirer configuration. Please check the Getting Started Guide to see of signing requests and responses is enforced for you. If this does not apply to you, please proceed to the next section on Payment Preparation.

Signing would impact the following cases:

- POST /payments API requests
- GET /payments/{paymentId}/status API requests
- GET /preferences/{id}/ API requests
- Notification API requests received by the Merchant
- Responses received by the Merchant

In this section, we will be looking into signing POST /payments requests and the responses from the Open Banking Service.

If signatures are enforced by the Merchant's Acquirer, the following headers will be mandatory in the API requests sent by the Merchant to the Open Banking Service:

- Signature
- Digest

3.3.2.1 The Signature header

The digital signing should be done by the Merchant by applying the “Signature” scheme as described in https://datatracker.ietf.org/doc/html/draft-cavage-http-signatures-12. This is equivalent to the “Authorization” scheme and the same procedure is followed to generate the signature and header parts but it uses the Signature header instead of the Authorization header.

In order to generate the signature string that is signed with a key, the Merchant must use the values of following HTTP header fields:

- Digest
- X-Request-ID
- MessageCreateDateTime

The headers used to generate the signature string also have to appear in the `headers` parameter of the Signature header, as shown below:

```
headers=”digest x-request-id messagecreatedatetime (request-target)”
```

To include the HTTP request target in the signature calculation, use the special `(request-target)` header field name. You can generate the header field value by concatenating the lowercased :method, an ASCII space, and the :path.

The (request-target) header field value to be used for the different APIs is:

API	(request-target) header field value
POST /payments	post /xs2a/routingservice/services/ob/pis/v3/payments
GET /payments/{paymentId}/status	get /xs2a/routingservice/services/ob/pis/v3/payments/{paymentId}/status*
GET /preferences/{Psuid}	get /xs2a/routingservice/services/ob/pis/v3/preferences/{psuId}*

*The placeholders {paymentId} and {psuId} must be substituted with the values as obtained within the respective flows.

The order of the headers present in the headers parameter must be followed to create the signature string. The signature string is created by concatenating the lowercased header field names followed with an ASCII colon `:`, an ASCII space ` `, and the header field value. Leading and trailing optional whitespace (OWS) in the header field value MUST be omitted. If the value is not the last value, then an ASCII newline `\n` is appended.

Example: For the payment request with the following required headers

```
POST /payments HTTP/1.1
digest: SHA-256=B/O1sG0L8+bEAqWF3aMZn3I0rx5YVi8r5cM6JHlTW7Q=
x-request-id: 1aad5e0f-02d7-aefb-61e3-6f4d3322cf71
messagecreatedatetime: 2023-03-15T10:07:26.264Z
```

The concatenated **"Signature String"** would be:

```
digest: SHA-256=B/O1sG0L8+bEAqWF3aMZn3I0rx5YVi8r5cM6JHlTW7Q=
x-request-id: 1aad5e0f-02d7-aefb-61e3-6f4d3322cf71
messagecreatedatetime: 2023-03-15T10:07:26.264Z
(request-target): post /xs2a/routingservice/services/ob/pis/v3/payments
```

The resulting signature parameter of the Signature header would be:

```
signature="N7kFLMMi/2R5hCd1gdO+GYhS70DOLMl+n8hborf42nFuu0HFjreoqU70gvxFWzgTPaWjdmNYY/7sOAUAQWudsM61Vc536X
maOGrrSxOINlH9l9QBk31xZMlJBf/+1+GtPb1BR26PYBjxKDMbN9W7PEVZLCDoObSnVLkvKbkLRWl0U8a39mDkUBu70Jw8yWusDU0g1OV
N+5YRfENPNtC2ZnVD80gxih4JoFV6f4WCcX4HXVl229veFNO5joNQyUc7qOkXUGN2g0omgN4iJxVGnzEJ9BCrNe+vK9T25LC0fwSp/W6A
9dDfuHQzMZgDJZZKpaX0Gg34i68etmi5oLrM3A=="
```

The algorithm parameter of the Signature header is fixed:

algorithm=”SHA256withRSA”

The ‘keyId’ parameter of the Signature header is the thumbprint of the used certificate, viewed with the SHA1 algorithm. The private key associated with `keyId` is used to generate a digital signature on the concatenated signature string applying the SHA256withRSA algorithm. The complete Signature header looks then like this:

```
Signature: keyId="DCAC7209573D506FC56095B8B23E8555A8F38B29", algorithm="SHA256withRSA", headers="digest
x-request-id messagecreatedatetime (request-target)",
signature="N7kFLMMi/2R5hCd1gdO+GYhS70DOLMl+n8hborf42nFuu0HFjreoqU70gvxFWzgTPaWjdmNYY/7sOAUAQWudsM61Vc536X
maOGrrSxOINlH9l9QBk31xZMlJBf/+1+GtPb1BR26PYBjxKDMbN9W7PEVZLCDoObSnVLkvKbkLRWl0U8a39mDkUBu70Jw8yWusDU0g1OV
N+5YRfENPNtC2ZnVD80gxih4JoFV6f4WCcX4HXVl229veFNO5joNQyUc7qOkXUGN2g0omgN4iJxVGnzEJ9BCrNe+vK9T25LC0fwSp/W6A
9dDfuHQzMZgDJZZKpaX0Gg34i68etmi5oLrM3A=="
```

In order for the Open Banking Service to validate the signature, the Merchant must upload the public certificate in the MSP Portal. Once the signature is confirmed and the iDEAL Subscription is authenticated, the Merchant will receive a successful response.

3.3.2.2 The Digest header

Calculate the Digest header as follows:

- **Step1:** Create a SHA-256 hash of the Payload

  - Note-1: if the output is hex-encoded, please make sure to convert it to binary data (convert the hex-encoded string to a byte array)
  - Note-2: payload formatting is important. If the Digest is generated by using an unformatted JSON payload, then please make sure that it matches with an unformatted request body used in the API request.

- **Step 2:** Base64-encode the SHA-256 hash
- **Step 3:** Prepend 'SHA-256=' to the resulting base64-encoded value

Example payload :

```
{"PaymentProduct":["IDEAL"],"CommonPaymentData":{"Amount":{"Type":"Fixed","Amount":"10.00","Currency":"EU
R"},"RemittanceInformation":"Cookie","RemittanceInformationStructured":{"Reference":"iDEALpurchase21"}},"
IDEALPayments":{"UseDebtorToken":false,"FlowType":"Standard"}}
```

Step 1: The SHA-256 hash of this request body is: 0d426d36fca1659980b9e371b25e2f17281bb285a634290d3da04233249b56ca. (Note: this is a hex-encoded representation)

Step 2: The base64-encoded hash (Note: hex to base64 encoding): DUJtNvyhZZmAueNxsl4vFygbsoWmNCkNPaBCMySbVso=

Step 3: The Digest header value: SHA-256=DUJtNvyhZZmAueNxsl4vFygbsoWmNCkNPaBCMySbVso=

3.3.3 Receiving signed requests and responses from the Open Banking Service

Receiving signed requests and responses could be enforced depending on the Acquirer configuration. Please check the Getting Started Guide Acquirer Profile to check it if applies for you. If this does not apply to you, please proceed to the next section on Payment Preparation.

In the event the Acquirer enforces Signatures, the following headers will be present in the API requests and responses received by the Merchant:

- Signature
- Digest

The digital signing performed by the Open Banking Service is done by applying the same “Signature” scheme as described above.

A notification request or response from Open Banking Service to the Merchant may contain the following headers (Signature and Digest included in the example):

```
Headers: {Authorization=Bearer iDEAL2.0testnotificationtoken, X-Request-ID=7e04be55-f710-4660-8254-
a48d0246d56b, MessageCreateDateTime=2024-01-30T17:03:52.111+01:00, Digest=SHA-
256=9CfdR8v5UlVl8YHNnpbO4v6uB/1B0EtWGLtnP7t2iVs=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-
sha256",headers="messagecreatedatetime x-request-id
digest",signature="v+IzPw8RKwGD3GWgLyuy/4RbA25PVwJxpvzs8QbqfAGLUSvOLhEL9dpQwvZi05DDbC80Z+1H7Kdyh3DumXRday
Y7XYnunA05tcirszq1fOmESP5S6iw0It9XoV5u/L8EPTgMvOXYECuDT+zVKDsB0PXRIyfT1p+kS1iKc7kckPvDycVGRYMyfXHATmcrlHY
6lSjMuw7WMlBOUo9Ac+dU8AQeqWzpzFjMa2Nd5XZkhd1vyKeVqh5cmWapJ2tZDk4/FwDZnpH3Po9PWKXwX/s+UolR/vlIUcRw+avIhU7L
6Qme7JDQDpZlcAgJfj/OpF8ZDlb6yfW32yFFzYnkMyGdYQ==",   Content-Type=application/json}
```

In order to ensure that the contents of the sent messages are correct and have not been altered during transmission or storage, the Merchant can validate the received signature. For the validation of the signature, the Merchant can use the public certificate of the Open Banking Service which can be downloaded from the MSP Portal.

The high-level steps that a Merchant needs to take in order to validate the Signatures are:

1. Digest calculation (as Digest is a part of the Signature)
2. Signature validation by re-creating the signature string

The above steps (1) and (2) are detailed below to describe the process behind signature validation. These steps are by default followed if standard signature validation libraries are used by the Merchant.

3.3.3.1 Digest header calculation for Notification requests and API responses

To verify the digest header, calculate the value and ensure you receive the same value as what is in the header of the Request/Response from the Open Banking Service. This is demonstrated below using a Notification request body.

```
{"PaymentProductUsed":"IDEAL","CommonPaymentData":{"PaymentStatus":"Expired","PaymentId":"141110","AspspP
aymentId":"0001115682120510","AspspId":"10002","DebtorInformation":{"Name":"Edsger Wybe Dijkstra -
Callback","Agent":"ABNANL2AXXX","Account":{"SchemeName":"IBAN","Identification":"NL44RABO0123456789"}}}}
```

Step 1 : The Open Banking Service always sends unformatted JSON payloads in the Requests/Reponses towards the Merchant. However, in case the Merchant sees a pretty-printed JSON object (e.g. by viewing logs via a tool), the payload should be converted to an unformatted JSON. All blank spaces should be removed unless they are part of the field value (Example: the debtor name has spaces which can remain).

Step 2 : Create a SHA 256 hash of this
payload: b1219370189b7c7d67f64fd6f72168187343d639f3aeada8ea3a2b36e0fac297
Note that this string is hex encoded.

Step 3 : Convert this hex encoding to a base64 encoding :
sSGTcBibfH1n9k/W9yFoGHND1jnzrq2o6jorNuD6wpc=

Step 4 : Finally prepend "SHA-256=" to the value. The final value is : SHA-256=sSGTcBibfH1n9k/W9yFoGHND1jnzrq2o6jorNuD6wpc=

This value should match the Digest as sent by the Open Banking Service.

3.3.3.2 Signature validation in Notification requests and API responses

The digital signing is done by the Open Banking Service by applying the “Signature” scheme as described in https://datatracker.ietf.org/doc/html/draft-cavage-http-signatures-12.

In order to verify the signature header, the Merchant should recreate the signature with the steps outlined below:

3.3.3.2.1 Re-create signature string

In order to generate the signature string that is signed with a key, the Merchant must use the values of HTTP header fields:

- MessageCreateDateTime
- X-Request-ID
- Digest

The headers used to generate the signature string also have to appear in the `headers` parameter of the Signature header:

```
headers=”messagecreatedatetime x-request-id digest”
```

The order of the headers present in the headers parameter must be followed to create the signature string. The signature string is created by concatenating the lowercased HTTP header field names followed with an ASCII colon `:`, an ASCII space ` `, and the HTTP header field value. Leading and trailing optional whitespace (OWS) in the HTTP header field value MUST be omitted. If the value is not the last value, then an ASCII newline `\n` is appended.

Example: For the notification request with the following required headers

```
POST /notification/status HTTP/1.1
Digest: SHA-256=9CfdR8v5UlVl8YHNnpbO4v6uB/1B0EtWGLtnP7t2iVs=
X-Request-ID: 7e04be55-f710-4660-8254-a48d0246d56b
MessageCreateDateTime: 2024-01-30T17:03:52.111+01:00
```

The concatenated **"Signature String"** would be:

```
messagecreatedatetime: 2024-01-30T17:03:52.111+01:00
x-request-id: 7e04be55-f710-4660-8254-a48d0246d56b
digest: SHA-256=9CfdR8v5UlVl8YHNnpbO4v6uB/1B0EtWGLtnP7t2iVs=
```

The string to verify is now defined and it can be validated using the public key of the Open Banking Service which can be downloaded from the Open Banking Service GUI.

Validate signature

The signature parameter of the Signature header sent by the Open Banking Service was:

```
signature="v+IzPw8RKwGD3GWgLyuy/4RbA25PVwJxpvzs8QbqfAGLUSvOLhEL9dpQwvZi05DDbC80Z+1H7Kdyh3DumXRdayY7XYnunA
05tcirszq1fOmESP5S6iw0It9XoV5u/L8EPTgMvOXYECuDT+zVKDsB0PXRIyfT1p+kS1iKc7kckPvDycVGRYMyfXHATmcrlHY6lSjMuw7
WMlBOUo9Ac+dU8AQeqWzpzFjMa2Nd5XZkhd1vyKeVqh5cmWapJ2tZDk4/FwDZnpH3Po9PWKXwX/s+UolR/vlIUcRw+avIhU7L6Qme7JDQ
DpZlcAgJfj/OpF8ZDlb6yfW32yFFzYnkMyGdYQ=="
```

The signature algorithm used is also contained in the Signature header: algorithm=”SHA256withRSA”

In order to verify a signature, a server MUST:

1. Re-create the signature string (see step above)
2. Hash the signature string with SHA256
3. Verify the signature with the public key of the Open Banking Service.

4 Payment Preparation

This section outlines the necessary preliminary steps a Merchant must take to send iDEAL payment requests.

4.1 Authorization Requests

In order to make payment requests, the Merchant will firstly be required to make an Authorization request. This is done by a POST /token call via the Authorization API. Doing so facilitates a secure connection between the Merchant and the Open Banking Service. Refer to the section Security Requirements - Access Tokens to find more information on how to make the request.

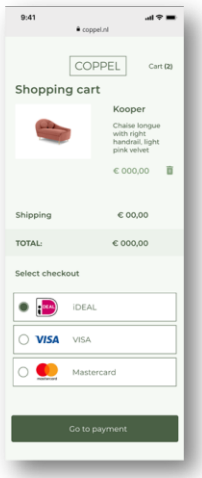
5 Payment Initiation - Standard Flow

A standard iDEAL payment process begins with a User choosing iDEAL at the Merchant's point of sale, which can vary between mobile apps, desktop browsers, or other platforms. The specific steps a User follows depend on the starting point of the transaction, the device used for payment authorization with their Issuer, the User's registration status with iDEAL, and whether they are recognized by a browser cookie.

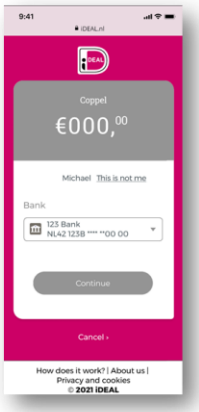
The iDEAL payment process can be distilled into these essential steps:

1. **Shopping Process:** The User places an order with the Merchant and selects iDEAL for payment.
2. **Payment Initiation:** The Merchant initiates the transaction via the Open Banking Service's Post /payments API, which then provides a redirect URL to the iDEAL Hub pay page or the Issuer's page.
3. **Issuing Bank Selection or User Recognition**: The User is taken to the iDEAL Hub pay page to select their Issuer. If a User has an iDEAL profile AND is recognized by a browser cookie; the iDEAL Hub pay page will show the Users stored payment preferences. The flow can continue by a redirect to the Issuer or a QR code scan or by the User receiving a push notification for a decoupled authorization process.
4. **Authorization at Issuer:** The User reviews and authorizes the payment details at their Issuer.
5. **Confirmation and Return to Merchant:** Post-authorization, the Issuer sends a confirmation to the iDEAL Hub. The User may be prompted to set up an iDEAL profile if they haven't already. The iDEAL Hub updates the Open Banking Service on the payment status, which then informs the Merchant. If a Merchant uses the Post /status API, the payment status is communicated directly by the Open Banking Service. In a redirect scenario, the User is then taken back to the Merchant's platform.

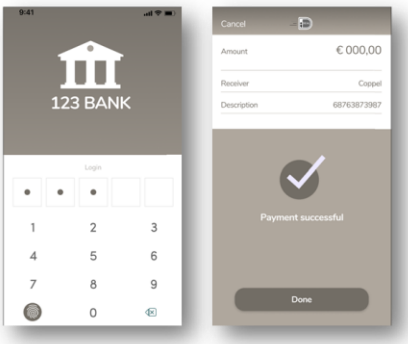
Below are two example screen flows. The first one is using a mobile device throughout. Notice that the iDEAL Hub recognized the Users iDEAL profile based on a browser cookie, which allows it to show the Users payment preferences.



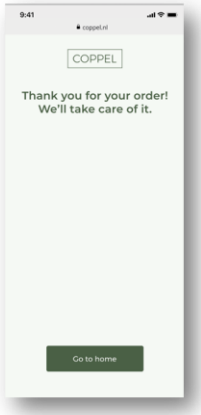
Shopping Process & Transaction Initiation



Customer Recognition / Issuer Selection

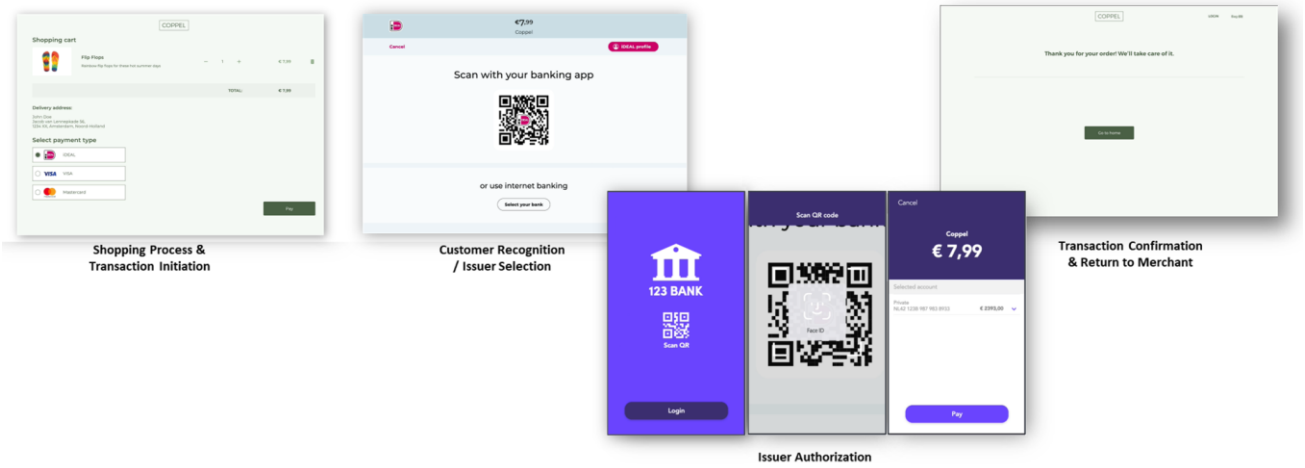


Issuer Authorization



Transaction Confirmation & Return to Merchant

The flow below starts on a non-mobile device and switches to a mobile device for the Issuer Authorization, in this example the User is not recognized by the iDEAL Hub and QR code is used to switch to the mobile device.



5.1 Sequence diagram

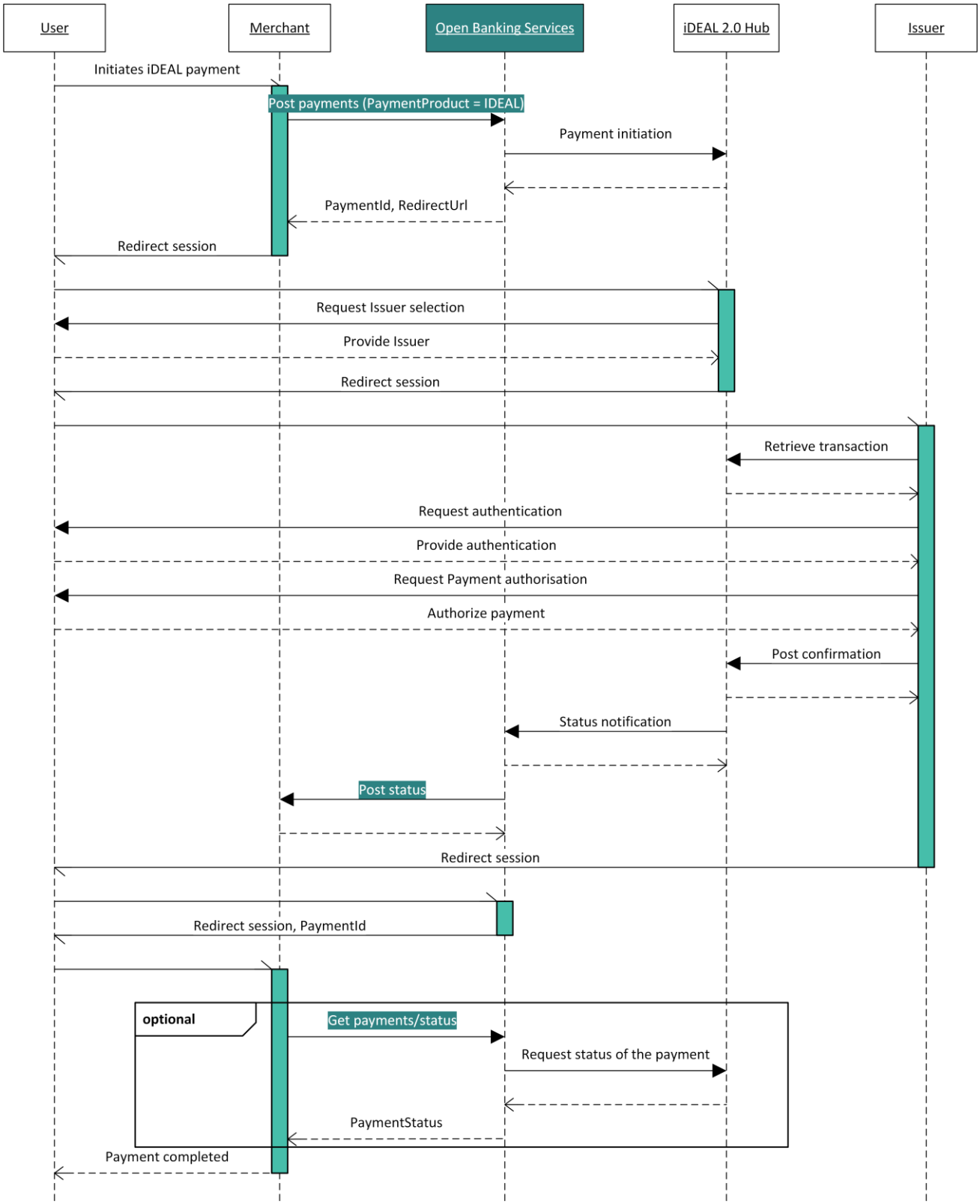
The sequence diagram here is an example of the possible flow for standard iDEAL payments without profile recognition. This does not encompass all the possible flows. The vertical green bars indicate which party is responsible for the session of the User. If a party has the session a screen can be displayed.

Notice that in the iDEAL 2.0 flow, the Merchant can receive a notification when the authorization of the payment is finished on the Issuer side. In order to receive this notification, the Merchant should implement the POST /status API, so this can be called by the Open Banking Service. The Merchant also has the option to request the status by calling the GET /payments/status API of the Open Banking Service.

5.1.1 Redirect and no User Recognition

In the Redirect Approach the browser session of the User is redirected from the Merchant to the iDEAL Hub pay page. The iDEAL Hub will ask the User which Issuer he would like to use. The browser session is then redirected to the chosen Issuer. The Issuer provides all the pages required for authentication of the User and will then ask for authorization of the payment. After that the User is redirected back to the Merchant.

Below an example of an iDEAL payment where the User is not recognized by the iDEAL Hub.


User
Merchant
Open Banking Services
iDEAL 2.0 Hub
Issuer
Initiates iDEAL payment
Post payments (PaymentProduct = IDEAL)
Payment initiation
PaymentId, RedirectUrl
Redirect session
Request Issuer selection
Provide Issuer
Redirect session
Retrieve transaction
Request authentication
Provide authentication
Request Payment authorisation
Authorize payment
Post confirmation
Status notification
Post status
Redirect session
Redirect session, PaymentId
optional
Get payments/status
Request status of the payment
PaymentStatus
Payment completed

5.2 POST /payments API

This endpoint is to be used to make an iDEAL 2.0 payment request towards the Open Banking Service.

Endpoint: POST **{Base Domain}**/xs2a/routingservice/services/ob/pis/v3/payments
Full URL of test environment :
https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments
Full URL for Production : Please refer to the Getting Started Guide because the base domain may defer depending on the Acquirer.

More information on the applicable fields is available in the tables below.

The examples available help to illustrate how you may formulate your requests.

5.2.1 Relevant Fields from POST /payments for Standard Payment Flow

These tables include specific details about iDEAL Payments and are to be used as supporting material to the Swagger Files. This is because the Swagger Files support more services other than iDEAL. For this reason, there will be discrepancies between Mandatory and Optional fields mentioned in the Swagger and mentioned here for iDEAL specifically. This applies to all of the remaining tables found in the Guide.

5.2.1.1 Request

Headers fields	Mult.	Type	Description
Authorization	1..1	String	Valid access token as retrieved from the Authorization API. Prepend the value with "Bearer" and a space. Example : Bearer b5f13c2a9fc24a3b31a000f03022f08b28039572b96179abdd6d28dfd4769604
X-Request-ID	1..1	String	UUID for unique request identification. This is generated by the Merchant so that the request towards the Open Banking Service can be uniquely identified.
Content-Type	1..1	String	To identify the media type of the source provided by the Merchant. If not provided, the Merchant can expect to come across the error "415, Unsupported Media Type"
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.
InitiatingPartyNotificationURL	0..1	String	The URL where Merchant expects to receive a notification of the payment status. This is strongly recommended to be used. Merchants have to specify this in the MSP Portal along with a Notification bearer token. Additionally, they may override the value in the MSP Portal by sending it here in the API. API value will take precedence over the value in the MSP Portal.
NotificationVersion	0..1	String	This is always v3 for iDEAL payments. This field does not need to be used, because the URL can simply be provided in the InitiatingPartyNotificationURL by ending the URL with a version number. For example : https://example.com/path1/path2/webbook/v3 As an alternative, Merchants may add the NotificationVersion to this field instead and provide a full URL in the InitiatingPartyNotificationURL field, ending with /notification. For example: Notificationversion : v3 InitiatingPartyNotificationURL : https://example.com/path1/path2/webbook/notification
InitiatingPartyReturnURL	0..1	String	This is equivalent to the Merchant Return URL from iDEAL 1.0. This is the URL where the User is returned to once the payment has been completed. There is an option to provide this in the MSP Portal, or as an alternative to be provided in the API request. The value provided in the API request will take precedence to the value in the MSP Portal.
Signature	0..1	String	May be mandatory depending on Acquirer. Information on whether it is mandatory is available in the Getting Started Guide.
Digest	0..1	String	Is mandatory if and only if the Signature element is contained in the header of the request.

Body fields	Mult.	Type	Description
PaymentProduct	1..1	Enum	This value should be IDEAL for iDEAL Payments. Note : The IDEAL payment product cannot be mixed with the other PSD2-xxxx payment products, because it requires a separate subscription and therefore uses a different authorization token.
CommonPaymentData	1..1	±	
+ TransactionType	0..1	Enum	• ONLINE - Used particularly for Online transactions, e.g. a webshop. If not provided, this is the default value. • QR - Used for transactions from a QR. eg. Invoice. • INSTORE - Used for instore transactions for eg.- a POS device • P2P - Used for peer-to-peer (customer-to-customer) transactions, e.g. a Transaction Request/Betaalverzoek
+ ExpirationPeriod	0..1	Integer	If not provided a default value will be used, which is as follows : Online : 1200 Instore : 120 This is required for QR type transactions
+ RemittanceInformation	1..1	String	Corresponds to iDEAL 1.0 field 'transaction.description' This field indicates what the iDEAL transaction is for. It will be shown on the Users bank statement. This is also shown to the User when doing the iDEAL transaction. Maximum Length is 35 characters for iDEAL transactions.
+ Amount	1..1	±	
++ Type	0..1	Enum	• Fixed - The amount defined by the merchant can not be changed by the user. This is taken as the default. • Change - The amount defined by the merchant can be changed by the user within the defined bounds. • Define - The amount needs to be defined by the user.
++ Amount	1..1	String	Amount of the payment. The decimal separator is a dot.
++ MaximumAmount	0..1	String	Used for amount type Change or Define. In case of "Change", value should be greater than or equal to the amount value. In case of Fixed, this value will be ignored. *pattern: ^\d{1,13}\.\d{1,5}$* *example: 1000.00*
++ MinimumAmount	0..1	String	Used for amount type Change or Define. In case of "Change", value should be less than or equal to the amount value. In case of Fixed, this value will be ignored. *pattern: ^\d{1,13}\.\d{1,5}$* *example: 5.00*
++ Currency	0..1	String	Currency of the payment. Only EUR is allowed for iDEAL transactions. EUR is also the default.
+ DebtorInformation	0..1	±	

++ Agent	0..1	String	BIC of the Issuer. Adding this field will skip the iDEAL Hub Paypage and redirect the User to the Issuer.
			• Corresponds to the iDEAL 1.0 field "issuerId"
			• Only applicable for FlowType = Standard.
			• This functionality will only be supported for the following cases:
			  ○ Merchants who know the preferred bank of the User.
			  ○ Merchants that store the preferred bank of the User must offer the option to pay with another bank than the preferred bank, which can only be done via the iDEAL Payment page.
			  ○ Merchants that can store a preferred bank must be able to support all (new) iDEAL Issuers as preferred bank.
			  ○ If using iDEAL QR with iDEAL 2.0, it is mandatory to provide the issuerId. Not doing so will lead to a breaking flow.

+ CreditorInformation	0..1	±	
++ InitiatingPartySubId	0..1	String	ID of the subMerchant. Use this field if the access token was generated only for the parent Merchant and not a specific subMerchant.
+ RemittanceInformationStructured	1..1	±	
++ Reference	1..1	String	Corresponds to iDEAL 1.0 field 'transaction.purchaseId'. This should contain an ID used by the Merchant to reference the transaction to its own order. This is shown on User's statement. As this field is used to reconcile within SEPA, the character set is restricted.
IDEALPayments	0..1	±	
+ FlowType	0..1	Enum	Transaction flow which the Merchant/CPSP desires to initiate. Standard will be used as the default if not provided in the request • Standard - Used for normal iDEAL transactions.

			• FastCheckout - Used for initiating the iDEAL checkout flow, see chapter 'Payment Initiation - iDEAL Checkout Flow' for more details.

5.2.1.2 Response

Headers fields	Mult.	Type	Description
X-Request-ID	1..1	String	UUID for unique request identification.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.
Signature	0..1	String	May be present depending on Acquirer. Information on whether it is mandatory is available in the Getting Started Guide.
Digest	0..1	String	Is contained if and only if the Signature element is contained in the header of the request.

Body fields	Mult.	Type	Description
CommonPaymentData	1..1	±	
+ExpiryDateTimestamp	0..1	String	The timestamp from which the transaction operation will expire, expressed in UTC time format(YYYY-MM-DDThh:mm:ss.sssZ)
+GuaranteedAmount	0..1	String	This is the amount guaranteed by the Issuer to the Merchant. This will be provided if the minimum and maximum amounts were defined in the request.
+ PaymentStatus	1..1	Enum	The following statuses apply for iDEAL : • SettlementCompleted • Cancelled • Expired • Open • Error
+ PaymentId	1..1	String	Id generated by the Open Banking Service. This should be used to refer to this payment in subsequent api calls. Max length : 35
+ AspspPaymentId	1..1	String	iDEAL Hub payment Id used to refer to the payment Max length : 35
Links	1..1	±	
+RedirectUrl	0..1	String	
++ Href	1..1	String	Contains the link to the iDEAL hub or to the Issuer.
+GetPaymentStatus	0..1	String	
++ Href	1..1	String	Endpoint to be called for retrieving the payment status.
UseWaitingScreen	0..1	Boolean	Will always be false for the standard flow.

5.2.2 Example: Standard iDEAL payment without Debtor token

This is an example of a standard iDEAL payment request with a minimum set of mandatory fields in the body.

Signature-related fields "Digest" and "Signature" may be mandatory depending on the Acquirer. Please check the Acquirer Profile in the Getting Started Guide for more information:

5.2.2.1 Request

```
Address: https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments
    HttpMethod: POST
    Content-Type: application/json; charset=UTF-8
    ExchangeId: b02d7597-3109-4291-86c3-a1d7faaee11c
    Headers: {Accept=application/json, Digest=SHA-256=0P2tSst5wZ/1rBS+ZF5Pvp79klNUek7tetk0j9w+mNs=, X-
Request-ID=fd3a9dae-5323-ea5a-5460-48857a264b9d, Authorization=Bearer
55aefa2dbd728d28dce18ef04e12e7014cde1e665901bc55ae0be044f114001e, MessageCreateDateTime=2023-12-
29T16:38:45.770Z,
Signature=keyId="39d8e82bb33e7e2a09cbcb3ef3eab351ee1c5e8f",algorithm="SHA256withRSA",headers="x-request-
id (request-target) digest
messagecreatedatetime",signature="a94S2WkCTBMdzx3nZrDcA61YkAsesjR9BjKkwvGLEmsCPyEUqk55zYHHE0rLVHwVD7YNchM
ZnEWFR3Vf63egGSTMPTD0xfmHZJua7Wz/VZVpmIkckCjfYiaQTilVImTcKeBIjhnkAd7MP2qQkr6bDtyQQ1zy1DGwDQgTlNVxOHbv+WcM
KJxoQcFn5qLTvaq6fnat3a2Ka/jAUHRytAJ+5xWRT0tKq85LvjB10fho8gMgHQ+4q/0H2gmPhh9QYDwCawbBdgdDWb5HtX0qjhftO3iAb
4G5HOlrqSBdTprYVDJmtqwh3YEekfy8KyHifiMMWiKVMrKohAcdvazlSMpKXA=="}
    Payload: {
    "PaymentProduct": ["IDEAL"],
    "CommonPaymentData": {
        "Amount": {
            "Amount": "1.00"
        },
        "RemittanceInformation": "iDEAL Standard Flow Minimum fields",
        "RemittanceInformationStructured": {
            "Reference": "iDEALStandardFlow"
        }
    }
}
```

5.2.2.2 Response

```
ResponseCode: 201

Headers: X-Request-ID=d04820e1-d2da-4def-8def-13062b57d3a6, MessageCreateDateTime=2023-12-
29T16:38:45.949Z, Digest=SHA-256=kyULomhnKJVz5IfrMDOWTs5ZpmBTKwjFrF6HO/LQG6s=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-
sha256",headers="messagecreatedatetime x-request-id
digest",signature="m//b5oj7D4Vuw5leONeJnlV6P2YvQEPs/fNq9WxE+HsS68iEICh3uMPPwhqv5JLh5UMDudZG0fY/YtvYx3GMUr
5DqGaTxplUWW5+l8dV6JM94POvgOveJOPayCMAW0pWd55sbt/IvBJmc19+z6UKMJ5liNwrvYboxYQp3gb9iCeVyqkUPjjDZmo3EvQXV/6
gcZAJCNwPy7vAKmGqiI57f/8EJQEZnwopTVMgn77JdpKht/dMw1oulCot4xaxmgcCwwWVxS951yqsNLxKhOnga4zHCe6anjr1F0qL0qZr
85z1eDwVfwwAOiv7tGEzGaXszZnOAnsmwMNjktaVvFBPUQ==", Date=Fri, 29 Dec 2023 16:38:45 GMT, Content-
Type=application/json;charset=UTF-8}
    Payload: {
    "CommonPaymentData": {
        "ExpiryDateTimestamp": "2023-12-29T20:38:45.925Z",
        "PaymentStatus": "Open",
        "PaymentId": "142641",
        "AspspPaymentId": "0001143558019460"
    },
    "Links": {
        "RedirectUrl": {
            "Href": "https://worldline.com"
        },
        "GetPaymentStatus": {
            "Href":
"https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments/142641/status"
        }
    },
    "UseWaitingScreen": false
}
```

5.3 Browser Redirect to Merchant Web shop

Once the User has interacted with the iDEAL Hub Paypage, the session will be redirected back to the Merchant's web shop. Here, the Open Banking Service appends the Merchant Return URL with the Service Name and the payment ID so that the session can used to recognise the transaction. This also enables the Merchant to display the correct screen to the customer.

An example of a return URL is as follows : https/www.example.org?scope=SURFQUw6MTcwNjAw

The Merchant would need to base64 decode this string : SURFQUw6MTcwNjAw.

The example above results in: {ServiceName}:{paymentId}, i.e. IDEAL:170600

5.4 Requirements from Currence

This section describes requirements that Currence sets for Merchants to follow for Standard payments.

5.4.1 Presentation of Standard iDEAL payment on Merchant environment

There are some rules regarding the presentation of iDEAL on the Merchants environment. The main purpose of these is to create a uniform user experience for Users whenever and wherever they pay with iDEAL.

- The iDEAL payment option must be presented in the list of payment options in such a way that it receives at least the same amount of attention as other payment options.
- **A Bank selection list MUST NOT be offered to Users on the Merchant webpage anymore for iDEAL 2.0.**
- The rules on presentation the iDEAL Payment button and the iDEAL logo can be found here: http://www.ideal.nl/en/payee/logos-banners/ .
    - In cases where the iDEAL logo cannot be displayed at the required size on a mobile device the mandatory free space of 15px around the logo may be reduced to accommodate the requirements of the mobile device.

5.4.2 Redirecting of Users

As a response to an iDEAL transaction initiation, the Merchant will receive a RedirectUrl, which is either an Issuer URL (directing the User directly to the Issuer domain) or a Payment Page URL (directing the User to the iDEAL Payment Page).

- If redirected from a browser, the redirect to the Issuer URL or iDEAL Payment page MUST be performed from the browser window where the User selected iDEAL as payment method. The complete page of the Merchant shall be replaced by the complete iDEAL Payment page.
    - Merchants MUST NOT open the redirect to the iDEAL Payment page or Issuer URL in a new browser window.
    - Merchants MUST NOT present the iDEAL Payment page or Issuer URL embedded within its own page, as this disallows recognition of a cookie.
- If redirected from a Merchant app, the redirect MUST take place outside the app in the default browser of the User.

Merchants MUST NOT redirect Users in a custom made in-app webview browser. Doing so will disallow for Users to be redirected to their mobile banking apps and will seriously breach privacy regulations.

Exceptions to the above are the use of SafariViewController for Apple iOS and Chrome Custom Tabs for Android. However be aware that these may not correspond to the Users default browsers, so that User recognition based on cookie might not work.

- During the migration period to the updated iDEAL specifications, the Merchant is still allowed to share a preferred Issuer (for example when the Merchant also stores this preference). This feature will be discontinued after the migration period ends.

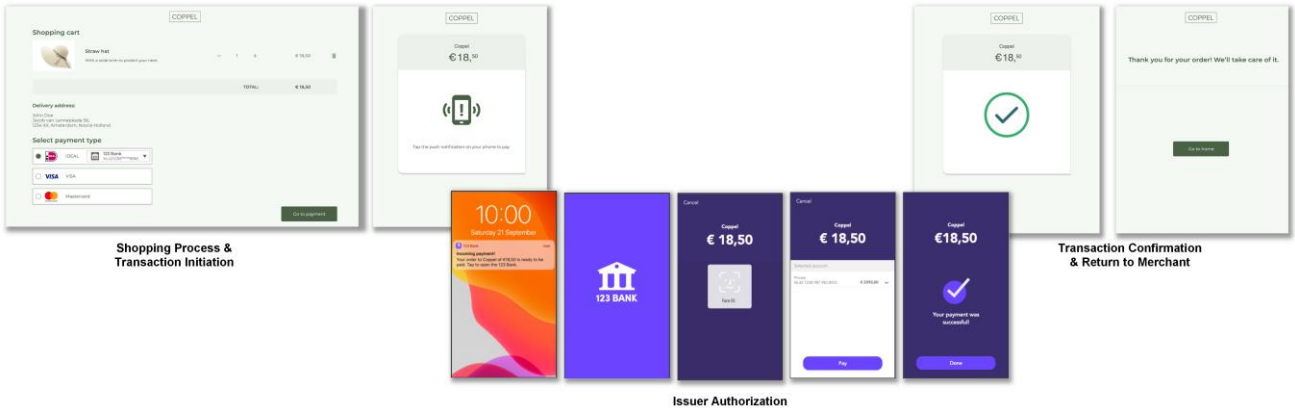
6 Payment Initiation - Standard Flow with Debtor Token

As an enhancement on the simple single iDEAL Payment, a Merchant can make use of Debtor Tokens for User recognition by iDEAL. By using Debtor tokens, Merchants are able to enhance the iDEAL user flow by cutting out the browser redirect to the Issuer or payment page in some cases. This is possible for transactions where the following conditions apply:

- User has an iDEAL profile and has consented to provide a user token to Merchant in a previous transaction
- User works with iDEAL push notifications in its Issuer's Mobile banking app
- User shops at the Merchant on a desktop browser or application

If one or more of the above conditions do not apply, a redirect is needed for the User.

Below an example screen flow where an existing User Token is used (notice the preferred bank and account are already visible next to the iDEAL logo on the first screen) to complete an iDEAL transaction. The sequence diagram for this flow can be seen under the 'Decoupled with existing Debtor token' section.



6.1 User Recognition identifier

- Debtor tokens are used by the iDEAL hub to identify a specific User
- PsuId is used by the Open Banking Service to identify a specific User. The Open Banking Solution internally links the Debtor token to a PsuId and maintains this coupling (an existing PsuId might be coupled to a renewed Debtor token)

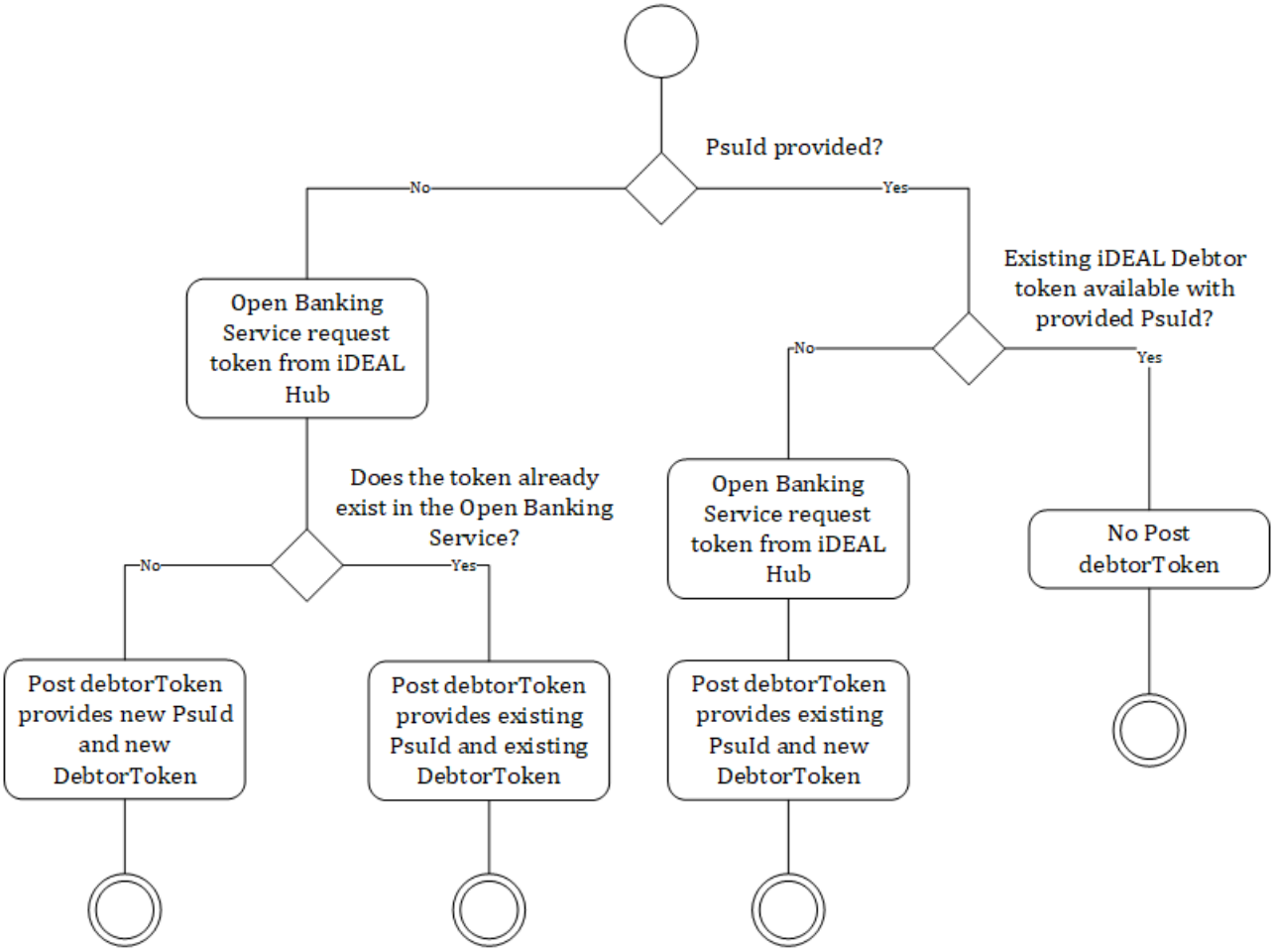
Towards the Open banking Service it's the **PsuId** should be used to identify the User.

The PsuId can either be provided by the Merchant, or can be generated by the Open Banking Service in which case the PsuId is provided in the Post /debtorToken notification.

If the PsuId is provided by the Merchant, it should be a unique identifier.

The Debtor token is also provided in the Post /debtorToken notification, for scenarios where the Merchant connects directly to the iDEAL Hub. This scenario is out of scope for this guide.

The Activity diagram below describes if and how a Post /debtorToken notification is sent towards the Merchant when the UseDebtorToken boolean is set to true in the POST /payments request.



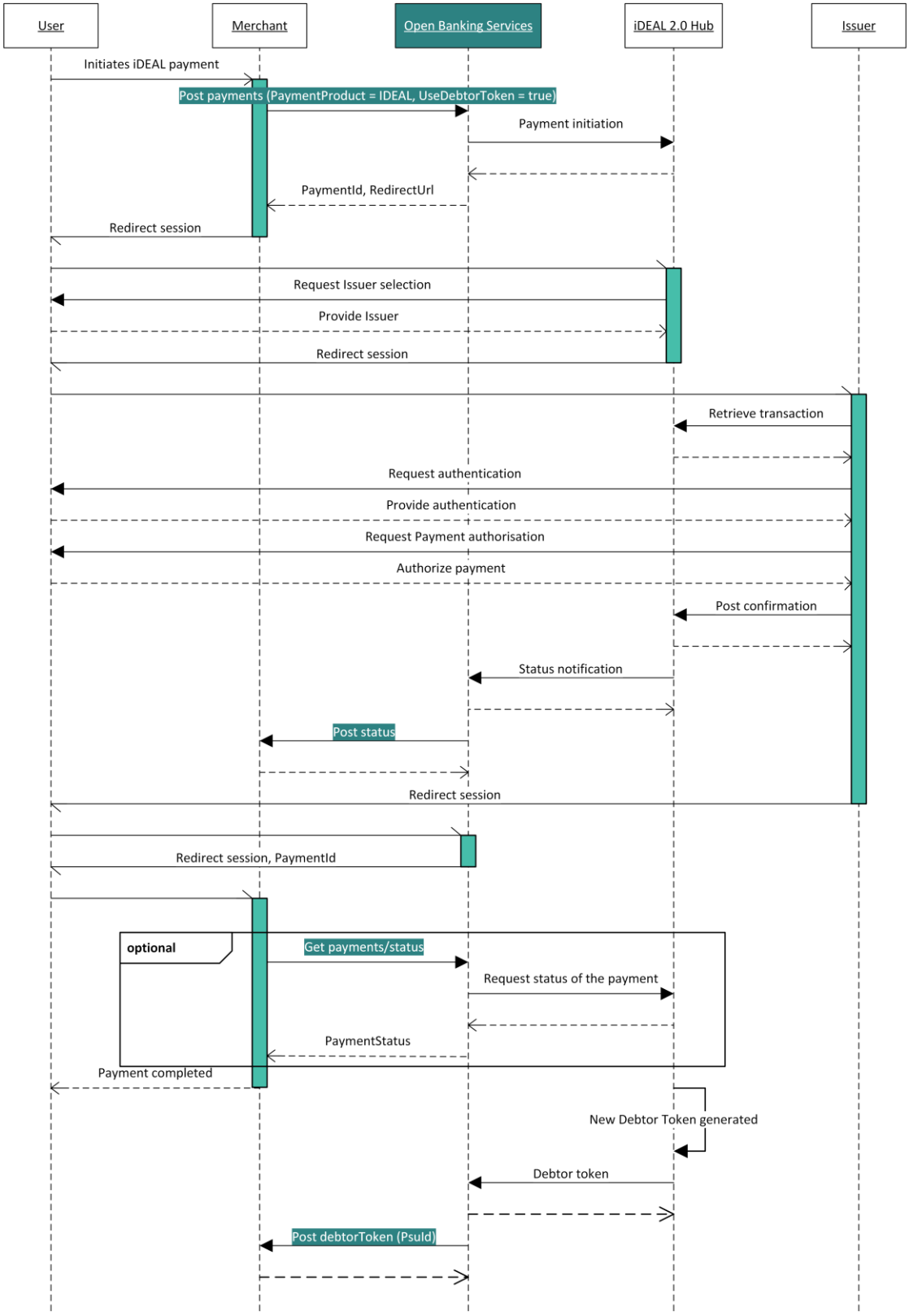
6.2 Sequence Diagrams

Below you can find 3 example flows of an iDEAL transaction with user recognition. In order to use this functionality, the Merchant should set the field '`UseDebtorToken = true`' in a Post /payments request towards the Open Banking Service.

6.2.1 Redirect with newly generated Debtor token

In this flow, user recognition is requested by the Merchant, but a Debtor token does not exist yet. A PsuId is required to identify the user; it can be provided in the Post /payments request, or (if not provided) it will be created by the Open Banking Service.

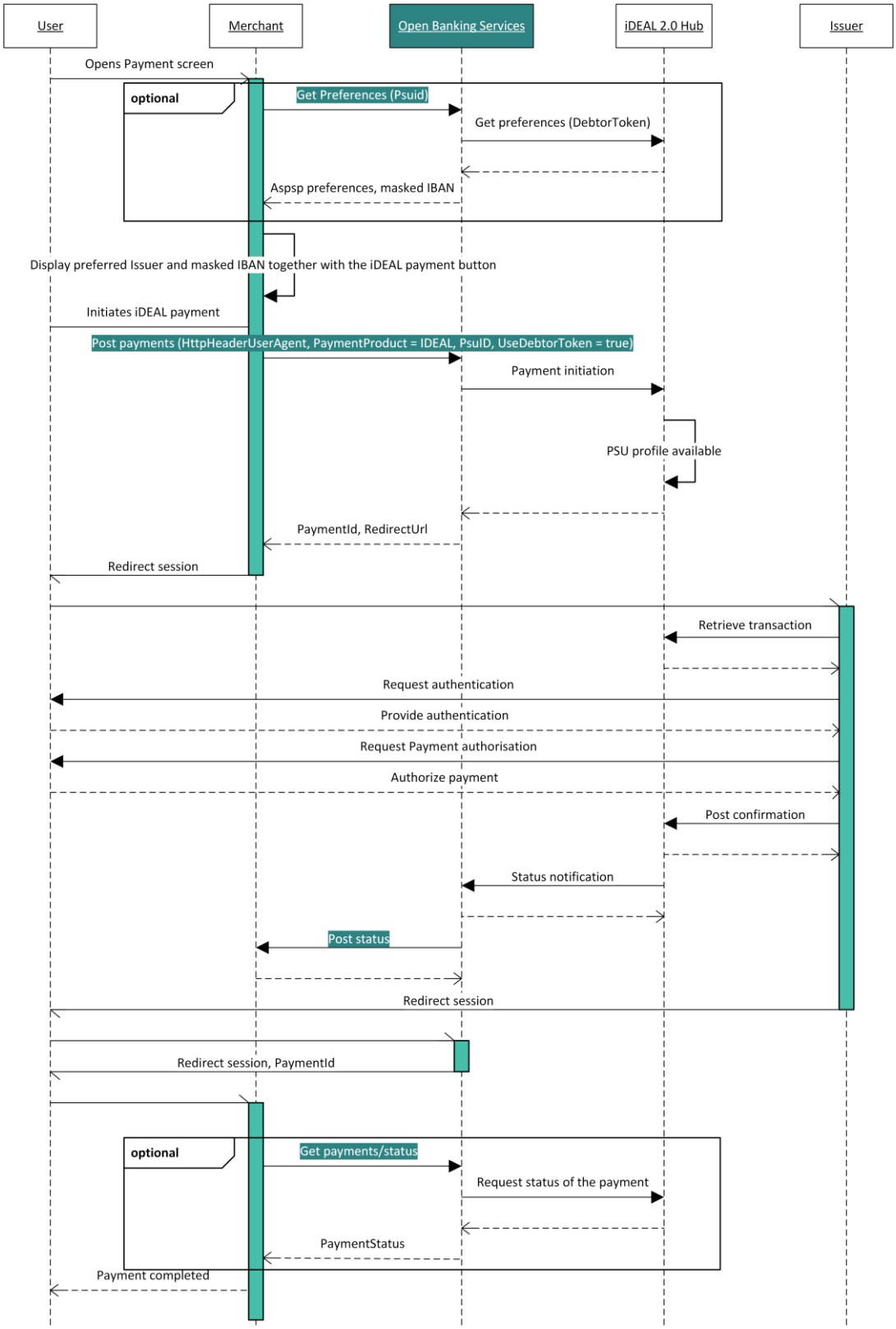
The PsuId will be provided in the Post /debtorToken API. This `PsuID` can then be used in subsequent iDEAL payments request for the same user (again with '`UseDebtorToken`' set to true).



6.2.2 Redirect with existing Debtor token

The Open Banking Service also offers a separate API which can be used to retrieve the preferences of the User, the Get /preferences API.

In the sequence diagram below the Get /preferences API is the optional step in the beginning. This flow is using the Debtor token to reduce the number of steps for the User to complete the payment (no Issuer selection required in the iDEAL Hub).


User
Merchant
Open Banking Services
iDEAL 2.0 Hub
Issuer
Opens Payment screen
optional
Get Preferences (Psuid)
Get preferences (DebtorToken)
Aspsp preferences, masked IBAN
Display preferred Issuer and masked IBAN together with the iDEAL payment button
Initiates iDEAL payment
Post payments (HttpHeaderUserAgent, PaymentProduct = IDEAL, PsuID, UseDebtorToken = true)
Payment initiation
PSU profile available
PaymentId, RedirectUrl
Redirect session
Retrieve transaction
Request authentication
Provide authentication
Request Payment authorisation
Authorize payment
Post confirmation
Status notification
Post status
Redirect session
Redirect session, PaymentId
optional
Get payments/status
Request status of the payment
PaymentStatus
Payment completed

6.2.3 Decoupled with existing Debtor token

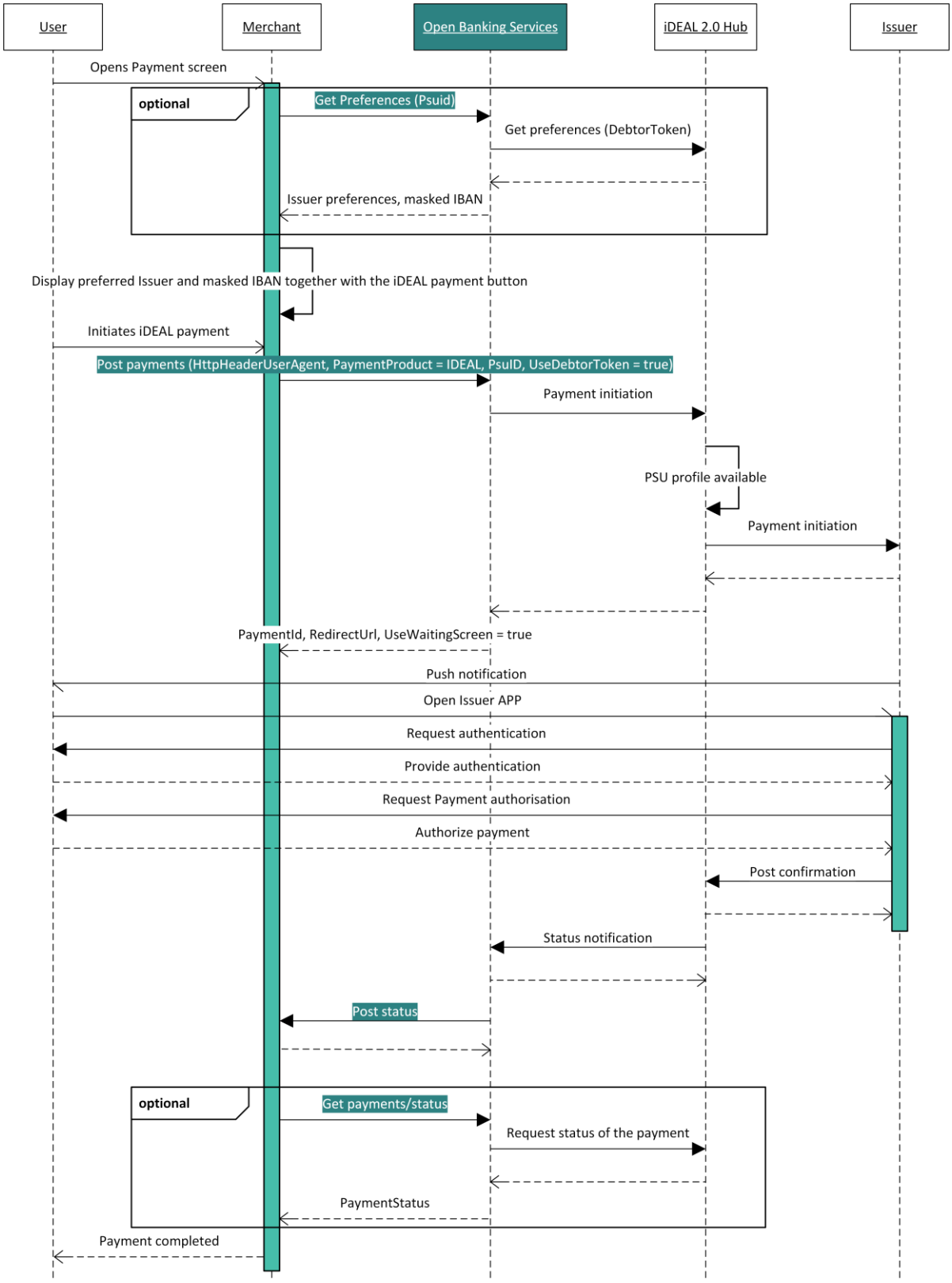
This sequence diagram shows a possible decoupled flow, but is otherwise the same as the 'Redirect with existing debtor token' flow.

In a decoupled flow another device is used to authenticate and authorize the payment, this can be seen in the sequence diagram below by the fact that 2 sessions (green vertical bars) are active at the same time. The User doesn't leave the session with the Merchant, in the meantime a session is started on another device (for example a smartphone) which handles the interaction between the User and the Issuer. The decoupled flow can potentially speed up the authentication process when bio-metrics are used in the Issuer's app on the smartphone.

If the following fields are provided in the Post /payments request a decoupled flow can be initiated by the Issuer:

- `HttpHeaderUserAgent`
- `PsuId`
- `UseDebtorToken = true`

In case of a decoupled flow the Post /payments response will have '`UseWaitingScreen = true`' field.


User
Merchant
Open Banking Services
iDEAL 2.0 Hub
Issuer
Opens Payment screen
optional
Get Preferences (Psuid)
Get preferences (DebtorToken)
Issuer preferences, masked IBAN
Display preferred Issuer and masked IBAN together with the iDEAL payment button
Initiates iDEAL payment
Post payments (HttpHeaderUserAgent, PaymentProduct = IDEAL, PsuID, UseDebtorToken = true)
Payment initiation
PSU profile available
Payment initiation
PaymentId, RedirectUrl, UseWaitingScreen = true
Push notification
Open Issuer APP
Request authentication
Provide authentication
Request Payment authorisation
Authorize payment
Post confirmation
Status notification
Post status
optional
Get payments/status
Request status of the payment
PaymentStatus
Payment completed

6.3 GET /Preferences

Endpoint: Get **{Base Domain}**/xs2a/routingservice/services/ob/pis/v3/preferences

This API can be used to retrieve information stored in the iDEAL Hub about an existing Debtor token connected to the User. The response can be used in the payment page of the Merchant to display that the Issuer and Account number are already selected; signaling the User that a quick payment flow can be expected.

6.3.1 Request

Path parameter	Mult.	Type	Description
Psuid	1..1	String	The Id of the User registered at the Open Banking Service

Header fields	Mult.	Type	Description
Authorization	1..1	String	Valid access token as retrieved from the Authorization API. Append the value with "Bearer" and a space. Example : Bearer b5f13c2a9fc24a3b31a000f03022f08b28039572b96179abdd6d28dfd4769604
X-Request-ID	1..1	String	UUID for unique request identification. This is generated by the Merchant so that the request towards the Open Banking Service can be uniquely identified.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.
Signature	0..1	String	May be mandatory depending on Acquirer. Information on whether it is mandatory is available in the Getting Started Guide.
Digest	0..1	String	Is mandatory if and only if the Signature element is contained in the header of the request.
AspspId	0..1	String	ID of the ASPSP as per PSD2. iDEAL Hub is the ASPSP for iDEAL payments. For iDEAL, these are the default values: Sandbox : 10002 Production : 10000 This field can be left blank by the Merchant.
InitiatingPartySubId	0..1	String	Sub ID of the Merchant. If retrieving the debtor preferences for a subMerchant's User, this field is mandatory.

6.3.2 Response

Header fields	Mult.	Type	Description
X-Request-ID	1..1	String	UUID for unique request identification. This is generated by the Merchant so that the request towards the Open Banking Service can be uniquely identified.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.
Signature	0..1	String	May be mandatory depending on Acquirer. Information on whether it is mandatory is available in the Getting Started Guide.
Digest	0..1	String	Is mandatory if and only if the Signature element is contained in the header of the request.

Body fields	Mult.	Type	Description
AspspName	0..1	String	The name of the ASPSP / Issuer
AspspLogoUrl	0..1	String	The logo of the ASPSP / Issuer
DebtorAccountIdentification	0..1	String	The masked IBAN or any other identifier recognized by the User and Issuer

6.3.3 Example : Get preferences

Signature-related fields "Digest" and "Signature" may be mandatory depending on the Acquirer. Please check the Acquirer Profile in the Getting Started Guide for more information:

6.3.3.1 Request

```
Address:
https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/preferences/9bfe1c8d-
e60a-4b45-b7fc-4d4c71f33a07
    HttpMethod: GET
    Headers: {Accept=application/json, Digest=SHA-256=47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=, X-
Request-ID=60070e69-2a1c-d720-5b1e-1184057054e6, Authorization=Bearer
12d0cec2d9e30041da49b7e1f0e48d6c4a815cb47cfd83becc99c49669d2ee78, MessageCreateDateTime=2024-02-
07T07:21:55.826Z,
Signature=keyId="39D8E82BB33E7E2A09CBCB3EF3EAB351EE1C5E8F",algorithm="SHA256withRSA",headers="x-request-
id (request-target) digest
messagecreatedatetime",signature="izRxEOduvf88kybUfVlyvuksuxoicPZEafSuRyrzFEruQ+4iCKK1G5NPZ5bp10UL2grmeQ7
HAYwHFVn/AcQ8LcVi1U7i/fIY5V90NL0SWw3A3rNzb4wSoiwsQT7Kz/pNwdN6/p+MeyYoqXpFvR8ct5IJw9Na8pj+L4VH+wFmhSOZdoj2
xcbXHKrEP6scb8ExahK3H+j2Z16zv0DgFX/UZEE0EBkTbV6FWDXc1JgYLMDtYFRI4t6ZCdbcdQ/LFzanwAXp9l3KpAV3qmTyBE/JzBdsI
zAMn5dK4zgUrOe3AFAtZGJ7jg83ulxL9wU5ggRtjOz/ACu2/R61Sf66LfgjgQ==", content-type=application/json}
```

6.3.3.2 Response

```
    ResponseCode: 200
    Headers: {X-Request-ID=62b999a6-704d-4b74-a588-71544a8e06b2, MessageCreateDateTime=2024-02-
07T07:21:56.25Z, Digest=SHA-256=bd7zfJ14uHY5YwvcXpqr78Df5jUpb0Z64styUET3afI=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-
sha256",headers="messagecreatedatetime x-request-id
digest",signature="wWRp5yyxbH1PRz6qH1NM2LZsW32oHagRr23rYmjCL0R1iUXRmBpuooNon5GKNRtstL/XxzbkVA95yvWmv8/8KU
qNj1BmZdz/O1KXQks0yrf5oJ0CzMTHSSUXBTp5I9Ygmv6/aSmVlRYRtSQB13yScCtBtkx2L3HhB4bJtYuxvTV7716M/qKEy5dYeGHsLoM
8ZmvGcx6xnX3DtL6+fPNQ4yr0FqD7RtZ5rq3HEsUpuia8Tt8Y7AgY0UGQiCVVgxQWhy2w3ErWu5RLmqKKMliQDnZZNUUdVyog80RVqMyF
LizzJ4wRw6N2tmohUlcmooAQ0bPFawq86ZuEp4KbdBgACw==", Date=Wed, 07 Feb 2024 07:21:56 GMT, Content-
Type=application/json;charset=UTF-8}
    Payload: {
    "AspspName": "IssuerName",
    "AspspLogoUrl": "https://checkout.company.com/login",
    "DebtorAccountIdentification": "NL44RABO******6789"
}
```

6.4 POST /payments (additional fields for Debtor Tokens)

Endpoint: POST **{Base Domain}**/xs2a/routingservice/services/ob/pis/v3/payments

The table below outlines **additional fields** which are relevant on top of the fields mentioned in the Standard Flow. The fields mentioned here are either added or have a change in the Multiplicity or Description.

6.4.1 Request

Header fields	Mult.	Type	Description
PsuId	0..1	String	Unique identification of the User at the (sub)Merchant. The Debtor token from iDEAL will be connected to the provided PsuId or, if the PsuId is not provided, the Open Banking Services

			will create one. The Open Banking Service will handle the Debtor token; if it is deleted at the iDEAL hub and a new one was is generated, it will be connected to the existing PsuId.
HttpHeaderUserAgent	1..1	String	Information about the User session. This field is filled with the Agent header field of the HTTP request between User and the Merchant.

Body fields	Mult.	Type	Description
IDEALPayments	1..1	±	
+ UseDebtorToken	1..1	Boolean	To be marked as true for using Standard Flow with Debtor tokens.
+ DebtorTokenCallbackUrl	0..1	String	The URL in which the notification for the POST /debtorToken API should be sent. This URL should end with /v3. The Open Banking service will then append this value with /notification/debtorToken when sending the request out. Because this value can be sent through the iDEAL Subscription in the MSP Portal, it is optional to send in the API. Any value sent in the API will override the value from the iDEAL Subscription. Note : the Notification Bearer Token in the iDEAL Subscription in the MSP Portal **MUST** be set in order to receive a POST /debtorToken call

6.4.2 Response

The POST /payments response for Standard Flow with User Recognition is almost identical to the POST /payments response for Payment Initiation - Standard Flow. Only the UseWaitingScreen field has a variation. Please refer to the Standard Flow payment response for reference for the other fields.

Body fields	Mult.	Type	Description

UseWaitingScreen	0..1	Boolean	If set to true it means the Merchant should display a waiting screen while the Issuer is starting a decoupled flow with the User. The RedirectUrl is provided as backup. Currence set the following requirements for the waiting screen:
			• There is a <Back> button, which leads the User back to the previous page of the Merchant;
			  ○ The Merchant MUST make a call to retrieve the status of iDEAL payment;
			  ○ When there is no final status available, the Merchant MUST inform the User that the payment is not finalised;
			  ○ When a final status is available, the Merchant MUST inform the User about that status (payment successful, payment failed, etc.) and follow up with the applicable action;
			• There is a <Didn't receive notification> button, which leads the User to the iDEAL Payment Page RedirectUrl to finalise the payment;
			• A message is displayed which informs the User about the action that is required, for example: 'Tap the notification on your phone to pay with the following bank';
			  ○ The Issuer name and logo are displayed;
			This page can be in the look and feel of the Merchant.

6.4.3 Example : POST /payments with Debtor Token

Signature-related fields "Digest" and "Signature" may be mandatory depending on the Acquirer. Please check the Acquirer Profile in the Getting Started Guide for more information:

6.4.3.1 Request

```
Address: https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments
    HttpMethod: POST
    Headers: {Accept=application/json, HttpHeaderUserAgent=Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36, Digest=SHA-256=71yw7TrlBmkiEf1iuuvOFQ16I58DcpKjPmH34E3JbXs=, X-Request-ID=8ac69eeb-
e0ad-b750-004a-84420c532b24, Authorization=Bearer
12d0cec2d9e30041da49b7e1f0e48d6c4a815cb47cfd83becc99c49669d2ee78, MessageCreateDateTime=2024-02-
07T07:12:09.144Z,
Signature=keyId="39D8E82BB33E7E2A09CBCB3EF3EAB351EE1C5E8F",algorithm="SHA256withRSA",headers="x-request-
id (request-target) digest
messagecreatedatetime",signature="d3Z8LwGupiPKmgTmhDyDV8ZoTGZy68i+7Dk7Icl2QuJQWs/hk1cfaY5rV4Hpwhf4HlKoWPP
lzfqCaEpxAb0ApwJpQpqUs1pTV4PR4lEhQwrwnR3+SZkiUwuzjKG0pOqH+asje1R+gCtcbu1EeIqP/HRnORBUaP5lt+hAP7t6gZ6ToJ5b
s1LXcFoZFTcQbkzKMrANUFHsR/K8Iq3U1bi+vkqUiAancdVvRPuJJwJBNBLKBtBirGYqWD6E1WUN73//q9CbyaT83HazkiCmNbvSY9i7s
GIMRBCKklp03DGzBJy4cCPPSJ4Xt6LTeuDmF4kVwh6Fl4+PhblCeTG7ZzY7NA==", content-type=application/json}
    Payload: {
    "PaymentProduct": ["IDEAL"],
    "CommonPaymentData": {
        "Amount": {
            "Amount": "8.00"
        },
        "RemittanceInformation": "iDEAL2.0 Transaction Standard Flow",
        "RemittanceInformationStructured": {
            "Reference": "iDEALDebtortoken"
        }
    },
    "IDEALPayments": {
        "FlowType": "Standard",
        "UseDebtorToken": true,
        "DebtorTokenCallbackUrl": "https://merchantdomain.com/notification/v3"
    }
}
```

6.4.3.2 Response

```
    ResponseCode: 201
    Headers: {X-Request-ID=770a78de-a69a-42a4-91eb-09865869a0e2, MessageCreateDateTime=2024-02-
07T07:12:09.543Z, Digest=SHA-256=I8J5GEV4aAPnMpVmwa9SghIsgW+krmLDZzc5ihG/clo=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-
sha256",headers="messagecreatedatetime x-request-id
digest",signature="c4GXuERD2VGpbTSVxeGqiBU1tNkGf5aWW6OKaoLFUK7wk2roN4nkPRih6nzw5YMBnuN0l7nJ5jmDTdzrG9RofD
Dg+qDBvB4NB42JtxDaxfPsvA4NTAuZbnSuNcE+96/NgWUNUueDQamV29+ET6XqxBCn7dWiEnhyZXRIhHLh/WpJeEyzbkL4sX2op4Zw3lB
kxIdLsE3H2nDM5iTwJmlzfkGkGZOGc+b+9bChWOonOh/GuW9/m8WSyawzSdkpb0fG3EhJ42xKB+E0mhW6W0umhhNHo9veDfkgMKFWBaxc
5J1pZnL17bODLTaqm1FL4+g2s8X1XvDJhsJZlL2eifK0aw==", Date=Wed, 07 Feb 2024 07:12:09 GMT, Content-
Type=application/json;charset=UTF-8}
    Payload: {
    "CommonPaymentData": {
        "ExpiryDateTimestamp": "2024-02-07T11:12:09.522Z",
        "PaymentStatus": "Open",
        "PaymentId": "171203",
        "AspspPaymentId": "0001090703303174"
    },
    "Links": {
        "RedirectUrl": {
            "Href": "https://worldline.com"
        },
        "GetPaymentStatus": {
            "Href":
"https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments/171203/status"
        }
    },
    "UseWaitingScreen": false
}
```

6.5 POST /debtorToken

The Merchant will receive the PsuId, which is required to pull debtor preferences for future payments via this API request. If the Merchant is providing the PsuId themselves, this request is a form of notification that the User has given permission to add Debtortoken for the Merchant, and that the Debtortoken flow was successful.

> If the Debtortoken Callback URL and Notification BearerToken are not filled out in the iDEAL Subscription of the MSP Portal, no POST /debtorToken notification will be sent out by the Open Banking Service.

Endpoint: POST **{Debtortoken Callback URL*}**/notification/debtorToken

6.5.1 Request

Header fields	Mult.	Type	Description
Authorization	1..1	String	The value of the Notification Bearer token set in the iDEAL Subscription of the MSP Portal. The value is prepended with "Bearer" and a space. Example: Bearer iDEAL2.0testnotificationtoken
X-Request-ID	1..1	String	UUID for unique request identification. This is generated by the Merchant so that the request towards the Open Banking Service can be uniquely identified.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.

Signature	0..1	String	May be mandatory depending on Acquirer. Information on whether it is mandatory is available in the Getting Started Guide.
Digest	0..1	String	Is mandatory if and only if the Signature element is contained in the header of the request.

Body fields	Mult.	Type	Description
PsuId	0..1	String	Unique identification of the User at the Merchant. The PsuId is connected to the Debtor token provided by the iDEAL Hub.
PaymentId	0..1	String	Id generated by the Open Banking Service. This should be used to refer to this payment in subsequent API calls.
DebtorToken	0..1	String	The Debtor token generated by the the IDEAL Hub. This field is provided as information only and does not have to be used in any of the flows described in this Guide.

6.5.2 Response

Header fields	Mult.	Type	Description
X-Request-ID	1..1	String	UUID for unique request identification. This is generated by the Merchant so that the request towards the Open Banking Service can be uniquely identified.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.

Body : HTML 204 (No content)

6.5.3 Example : POST /debtorToken

As visible in the example, the URL will be appended with /notification/debtorToken if the debtortoken callback URL was provided ending with /v3.

Signature-related fields "Digest" and "Signature" may be mandatory depending on the Acquirer. Please check the Acquirer Profile in the Getting Started Guide for more information:

6.5.3.1 Request

```
  Address: https://merchantdomain.com/notification/v3/notification/debtorToken
  HttpMethod: POST
  Headers: {Authorization=Bearer iDEAL2.0testnotificationtoken, X-Request-ID=88646b72-ac86-48d9-b2b7-
dbc7acdf6223, MessageCreateDateTime=2024-02-07T08:12:22.638+01:00, Digest=SHA-
256=r/zBvnHJNLMGs3oIsx5TH3z7XFXc1XILxqMOA+YksiY=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-
sha256",headers="messagecreatedatetime x-request-id
digest",signature="LXEqAe7rDNJkDobyS8wc3/QJddjSB4Y57zV3ND4U5qpKhzXspcHMKkGJOMNEoL3clKgZ4m9pJyAeAAEZpVcERa
io2vnyZ7snEtPARCraL2hHEUHkOkbUeIoT8a5ERMaFBL46FYCEfQmGGZcAOlS4IvtMrQok0/BZhsFwawHuw4wb2vpxslPCZvNP7CXu4XB
OFptNAPerCeVChDjLd2rX6Xuws1tE86ioY9HVmtJT2xKm7ojfmkkpFGefA6g6VXNIHjqWXGTgZGOEqqY9W+SbpnyFaj1Qyq5wStzeLJq6
Z+uu3AfhvY61m5DpWgzckYuN2ekrFKb0qX22akmss85F8Q==", Content-Type=application/json}
  Payload: {
  "PsuId" : "9bfe1c8d-e60a-4b45-b7fc-4d4c71f33a07",
  "PaymentId" : "171203",
  "DebtorToken" : "0b8a10ae-ad7c-4675-94d7-c0c94f48ee07"
}
```

6.5.3.2 Response

```
ResponseCode: 204
```

6.6 Requirements from Currence

6.6.1 Profile recognition via Debtor Tokens

If a Merchant wishes to make use of Debtor Tokens for its Users, the following applies:

- The User MUST hold an account with the Merchant;
- To retrieve the preferred Issuer and (masked) IBAN of the User, the Merchant MUST provide a PsuId that uniquely identifies the User at the Merchant;
- The Merchant MUST be able to recognize the User on a return visit;
- The Merchant MUST retrieve and display the preferred Issuer and masked IBAN of the User together with the iDEAL payment button;

Please note that a Debtor Token can only be linked to one iDEAL profile, and therefore only one Debtor Token (one PsuId in the Open Banking Service) can be registered per Merchant account per User.

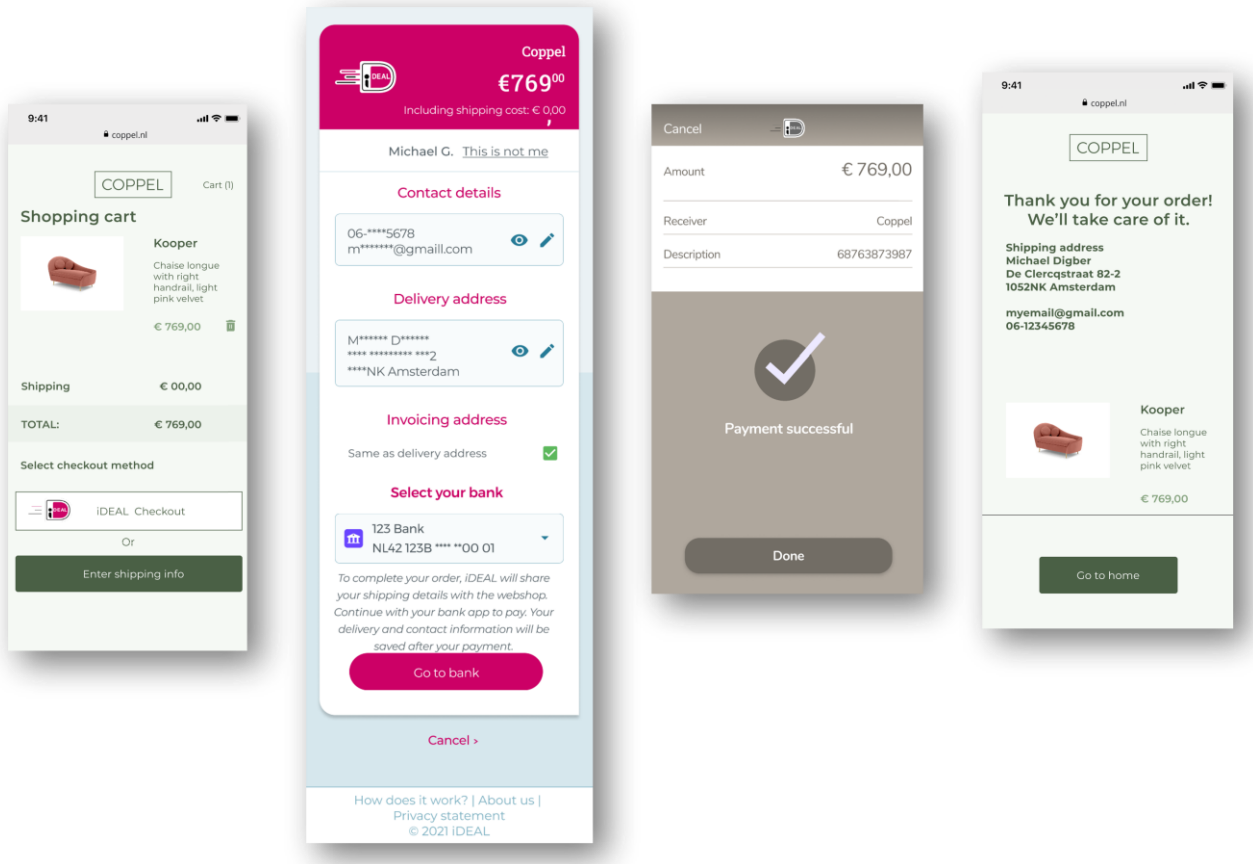
7 Payment Initiation - iDEAL Checkout Flow

An iDEAL payment with iDEAL Checkout allows a User to centrally manage his shipping, invoice and contact details in his iDEAL profile and provide these to a Merchant as part of the iDEAL payment.

In an iDEAL payment with iDEAL Checkout, a User is redirected to the iDEAL Payment Page. If the User is recognized by a cookie and an iDEAL profile is detected, the shipping address preferences are shown to him on the iDEAL payment page. The User then confirms his preferred contact details and shipping address on the iDEAL payment page, after which he proceeds with the transaction through either a redirect to the Issuer, a QR code scan, or a push notification to its mobile banking app. After the User has authorized the payment at the Issuer, he is redirected back to the Merchant. Upon authorization, the Merchant receives confirmation of the payment (similar to a regular iDEAL transaction), which will additionally include the chosen contact details and shipping address of the User.

Currently only Dutch shipping addresses are allowed for Users

Below an example screen flow of a iDEAL Checkout Payment.



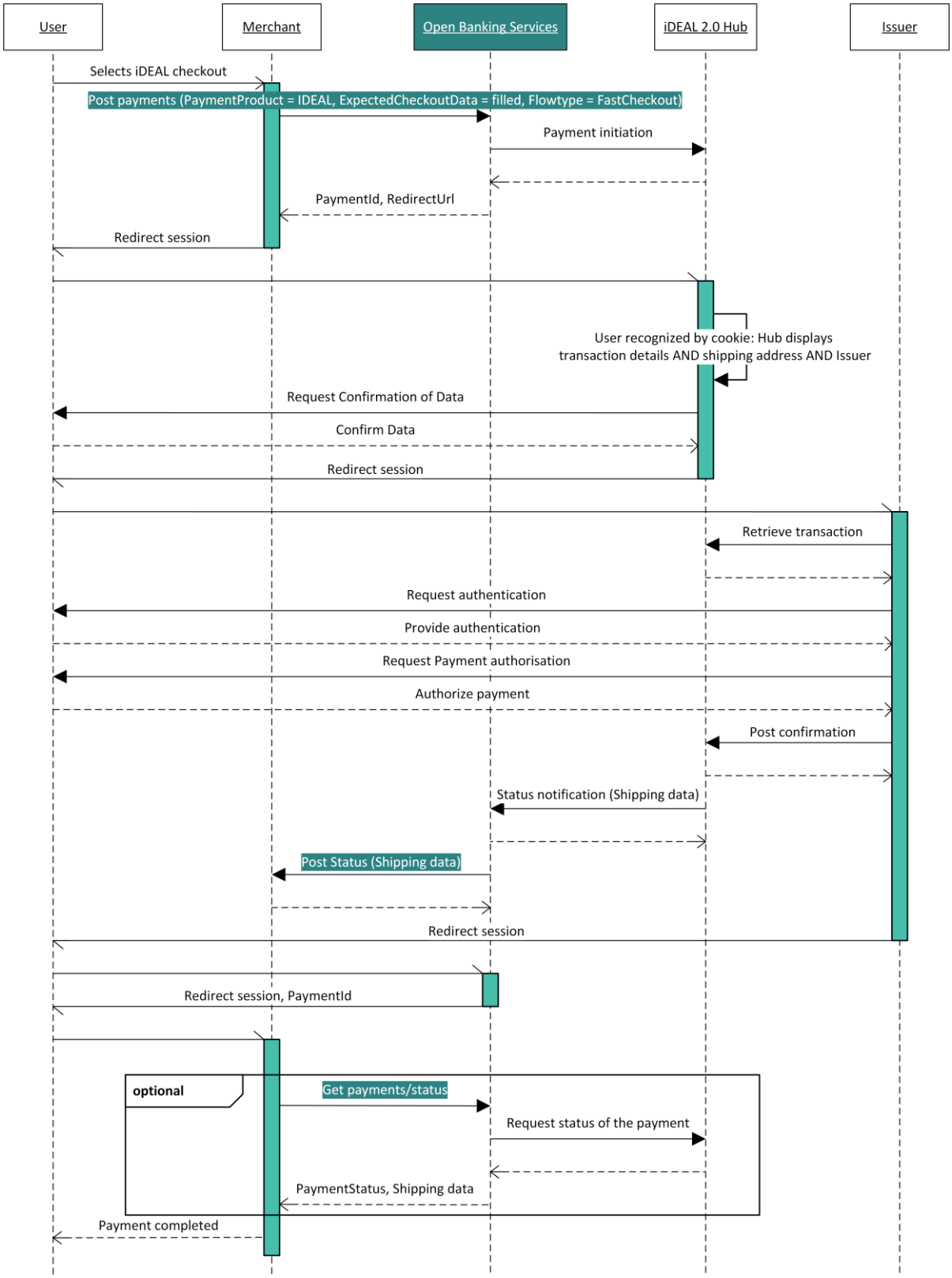
7.1 Sequence diagram

7.1.1 iDEAL checkout and User recognized

In the sequence diagram below the User's iDEAL profile is recognized and the requested shipping information is part of the Users profile. This flow will simplify the checkout screen of the Merchant; shipping, invoice and contact details are all handled by the iDEAL Hub. The shipping data is provided in the Post /status request or the Get /payments/status response after the payment was finalized at the Issuer.

In case the User cannot or does not want to provide his iDEAL Checkout data, the transaction is canceled. This means that a successful iDEAL Checkout transaction should always include the iDEAL Checkout data.

Additional scenarios are described in the 'Requirements from Currence' at the end of this chapter.


User
Merchant
Open Banking Services
iDEAL 2.0 Hub
Issuer
Selects iDEAL checkout
Post payments (PaymentProduct = IDEAL, ExpectedCheckoutData = filled, Flowtype = FastCheckout)
Payment initiation
PaymentId, RedirectUrl
Redirect session
User recognized by cookie: Hub displays
transaction details AND shipping address AND Issuer
Request Confirmation of Data
Confirm Data
Redirect session
Retrieve transaction
Request authentication
Provide authentication
Request Payment authorisation
Authorize payment
Post confirmation
Status notification (Shipping data)
Post Status (Shipping data)
Redirect session
Redirect session, PaymentId
optional
Get payments/status
Request status of the payment
PaymentStatus, Shipping data
Payment completed

7.2 POST /payments (additional fields for iDEAL Checkout)

Endpoint: POST **{Base Domain}**/xs2a/routingservice/services/ob/pis/v3/payments

The table below outlines **additional fields** which are relevant on top of the fields mentioned in the Standard Flow. The field mentioned here are either added or have a change in the Multiplicity or Description

7.2.1 Request

Request Body fields	Mult.	Type	Description
CommonPaymentData	1..1	±	
+ Amount	1..1	±	The addition of OrderAmount and ShippingCost must be equal to Amount.
++ AmountBreakdown	1..1	±	
+++ OrderAmount	1..1	String	Required for iDEAL Checkout Flow
+++ ShippingCost	1..1	String	Required for iDEAL Checkout Flow. Shipping Cost can also be 0.00€
IDEALPayments	0..1	±	
+ FlowType	1..1	Enum	Transaction flow which the Merchant/CPSP desires to initiate. This must state "**FastCheckout**" in order to use iDEAL Checkout Flow.
+ ExpectedCheckoutData	0..1	±	For iDEAL Checkout Flow, least one of the three must be set to true: • Shipping Address, • Billing Address • DebtorContactDetails Only fields set to true will be returned in the response. The Merchant MUST comply with the General Data Protection Regulation (GDPR).
++ ShippingAddress	0..1	Boolean	Only addresses in the Netherlands are allowed to be entered by the Consumer.
++ BillingAddress	0..1	Boolean	Only addresses in the Netherlands are allowed to be entered by the Consumer.
++ DebtorContactDetails	0..1	±	Only fields set to true will be returned in the response.
+++ FirstName	0..1	Boolean	
+++ LastName	0..1	Boolean	
+++ PhoneNumber	0..1	Boolean	
+++ Email	0..1	Boolean	

7.2.2 Response

The POST /payments response for iDEAL Checkout is identical to the POST /payments response for Payment Initiation - Standard Flow. Please refer to the Standard Flow payment response for reference.

The fields requested via CommonPaymentdata.IDEALPayments.ExpectedCheckoutData will be present in the Status Notification request of the payment or in the GET Status response.

7.2.3 Example: iDEAL Checkout

Signature-related fields "Digest" and "Signature" may be mandatory depending on the Acquirer. Please check the Acquirer Profile in the Getting Started Guide for more information:

7.2.3.1 Request

```
    Address: https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments
    HttpMethod: POST
    Headers: {Accept=application/json, Digest=SHA-256=OCbJurWSWF7ltmMlP/kM4oT9C2p5ZEjh4Oh0hcGrzek=, X-
Request-ID=8221d7a3-f41b-514f-bbc8-c39a744d145d, Authorization=Bearer
175a6d1c801969021f1071168661d617a17e3f8ba3824b7401491968644d2704, MessageCreateDateTime=2024-02-
14T17:33:58.301Z,
Signature=keyId="39d8e82bb33e7e2a09cbcb3ef3eab351ee1c5e8f",algorithm="SHA256withRSA",headers="x-request-
id (request-target) digest
messagecreatedatetime",signature="CAK3vtjrbVlYjtoi5nmor+hGvOTMHGRT4p+hXHU8MnLv5mJvtQRmQVbzujHENclCxTIyQjo
01wy/ckBuYdFj1hyJfWW/JPJ0h1qCZX8/XBrI+IA89ZDqNvcd62wUHOcYeFMtIhcySk03Ln7YRjObIJ1g/ky/jB4/aA0UStAWOstc72YU
8S2A+5JLy+lN7MzQoV2KNo3QzOyOzPZ9Ga6/2xdMnq7bG+uSBP9n07JljQPoqPr6onsX7UFSjYtind0kX+6wfH2XjQI2pFEhF1q8brL3p
onnmQ053JJpp+dX68HFIuDSXn+O9CHL7HLY51xQIbDilPaocA1dPlPLeSsU+Q==", content-type=application/json;
charset=UTF-8}
    Payload: {
    "PaymentProduct": ["IDEAL"],
    "CommonPaymentData": {
        "Amount": {
            "Type": "Fixed",
            "Amount": "24.50",
            "AmountBreakdown": {
                "OrderAmount": "20.50",
                "ShippingCost": "4.00"
            }
        },
        "RemittanceInformation": "iDEAL2.0 FastCheckout Flow",
        "RemittanceInformationStructured": {
            "Reference": "iDEALpurchase21"
        }
    },
    "IDEALPayments": {
        "FlowType": "FastCheckout",
        "ExpectedCheckoutData": {
            "DebtorContactDetails": {
                "FirstName": true,
                "LastName": true,
                "PhoneNumber": true,
                "Email": true
            },
            "ShippingAddress": true,
            "BillingAddress": true
        }
    }
}
```

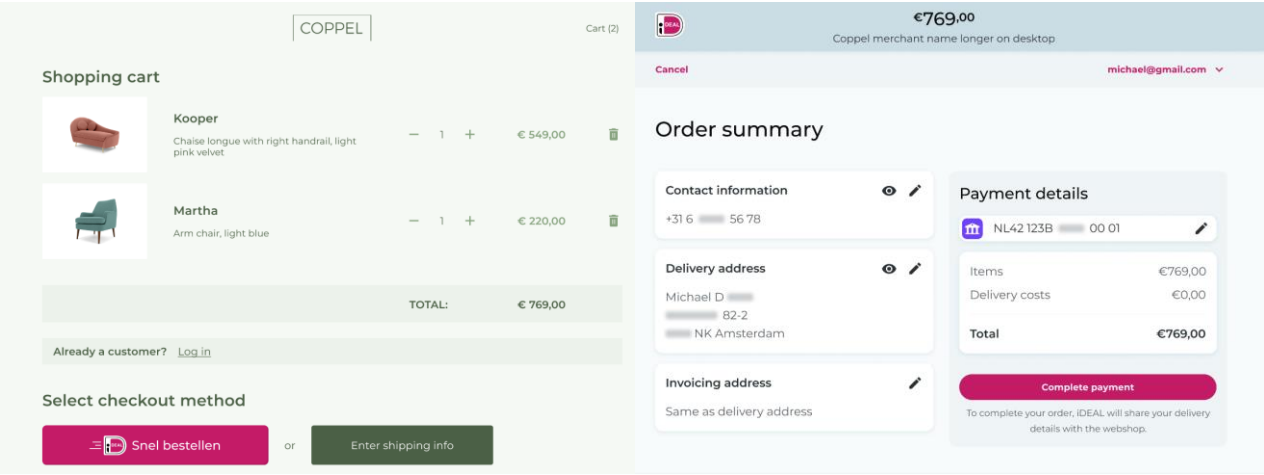
7.2.3.2 Response

```
    ResponseCode: 201
    Headers: {X-Request-ID=e3727e35-e3fd-4bb9-9c6c-1725da231527, MessageCreateDateTime=2024-02-
14T17:33:58.939Z, Digest=SHA-256=16eS+f5lOwGtN05ukpGcKvDrKt2DR5QCNNsDQfLBe5c=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-
sha256",headers="messagecreatedatetime x-request-id
digest",signature="RcDMNd5jy/HiJ9s2QxmzwfKuXMTzT4Pk5lp8z0NhOG9fS1VrPVsEcq0u8Hm3C+6UwdPjGDekSC1kytvOm/gHSo
ZQTIEvpbh99w7yeBCzQNbIQNWr9Fnl4QyXgyrUfVgR889yEGntw+0esl0vu+LpFqi+0CI4rdu5vbzaqLz/oG00D9NC8W0PfEmj469XBev
p3hDcnb6ieINmOVo4ZYu2ByaD3p6TC1vN+YYRqw6t/n+qZVjYG3DIhceTtiqsPvoucU7wYVRplqta7GyAK9RGevXzZfY6guiZrsDHl6mW
hToKiQoS2YjGE6COF89oOeCfC5NXy4n+/fwKAhqmWBdPsQ==", Date=Wed, 14 Feb 2024 17:33:58 GMT, Content-
Type=application/json;charset=UTF-8}
    Payload: {
    "CommonPaymentData": {
        "ExpiryDateTimestamp": "2024-02-14T21:33:58.919Z",
        "PaymentStatus": "Open",
        "PaymentId": "174142",
        "AspspPaymentId": "0001077680035886"
    },
    "Links": {
        "RedirectUrl": {
            "Href": "https://worldline.com"
        },
        "GetPaymentStatus": {
            "Href":
"https://routingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments/174142/status"
        }
    },
    "UseWaitingScreen": false
}
```

7.3 Requirements from Currence

7.3.1 Presentation of iDEAL Checkout payment on Merchant environment

- To indicate to Users that iDEAL Checkout is offered, the Merchant MUST show the iDEAL Checkout logo. Logo's can be found here: https://www.ideal.nl/bedrijven/logos/ . This logo may be accompanied by the term 'Snel Bestellen' to provide additional clarity on the flow the User will enter after selecting this option;
- Before the User selects iDEAL Checkout, the applicable shipping costs MUST be communicated to the User, to prevent the User being confronted by a higher amount in the iDEAL screens than expected;
- The Merchant MUST confirm the datafields, received from Currence, to the User, preferably on the order confirmation screen.



7.3.2 Selection/registration of iDEAL Checkout data by User

After initiating an iDEAL Checkout, the User is redirected to the iDEAL Payment Page. Here, the amount breakdown in order amount and shipping amount is shown, next to the regular payment details.

- If the User is recognized by a browser cookie, the known shipping address preferences from his iDEAL profile are shown, in addition to the regular profile preferences. Note that only the data fields indicated in ExpectedCheckoutData will be shown to the User (this applies to all iDEAL Checkout flows).
- Profile information presented will be partly masked. This can be unmasked by the User if needed by logging in via his Issuer.
- If a registered User does not have any shipping address registered with its profile yet, he will first be asked to provide an address before continuing with the iDEAL Checkout transaction.
- If the User is not recognized, but does have an iDEAL profile, he can login at his Issuer to load his shipping data from his profile.
- At any time during the iDEAL Checkout flow, the User may choose to change his preferred address for the transaction or add a new address. This is done on the iDEAL payment page.
- If the User is not recognized and is a new User, the User is asked to register an iDEAL profile, including shipping address.

7.3.3 Shipping data in Status Notification Request

- After confirming the selected delivery details and preferred IBAN, the User is redirected to the Issuer for authenticating and authorizing the payment. This is done at the Issuer, similar to a regular transaction.
- After the User has confirmed the transaction payment at the Issuer, the User's iDEAL Checkout data (like shipping address) is added to the Post /status callback alongside the other payment details.
- The iDEAL Checkout data can be found in the DebtorInformation group. Note that only data fields that were indicated in the ExpectedCheckoutData will be included in the Status Notification request.

8 Payment Finalisation - Payment Notification

The notification API is to be implemented on the Merchant's side in order to receive the payment notification to the URL provided by the Merchant.
The Open Banking Service will post status notifications to these endpoints.

Please note that in this section we do not cover the POST /debtorToken requests. Those are covered in the Payment Initiation - Standard Flow with Debtor Token.

8.1 Prerequisites

In order to receive a notification request from the Open Banking Service, you will need to fill in two fields in the MSP Portal iDEAL Subscriptions page.

- Status Notification URL
- Notification BearerToken

The Notification BearerToken is a static token and does not have an expiration period. In the test environment, the value of this field is "iDEAL2.0testnotificationtoken"

The purpose of this token is to allow Merchants to authenticate requests received. This will be sent in the Authorization header of the Notification request.

> If the Status Notification URL and Notification BearerToken are not filled out in the iDEAL Subscription of the MSP Portal, no notifications will be sent out by the Open Banking Service.

8.2 POST Status

Endpoint: POST **{Status Notification URL*}**/notification/status

More information on the applicable fields is available in the tables below.

8.2.1 Relevant fields in for POST /status

8.2.1.1 Request

Headers fields	Mult.	Type	Description
Authorization	1..1	String	The value of the Notification Bearer token set in the iDEAL Subscription of the MSP Portal. The value is prepended with "Bearer" and a space. Example: Bearer iDEAL2.0testnotificationtoken
X-Request-ID	1..1	String	UUID for unique request identification.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.
Signature	0..1	String	May be present depending on Acquirer. Information on whether it is mandatory is available in the Getting Started Guide.
Digest	0..1	String	Is mandatory if and only if the Signature element is contained in the header of the request.

Body fields	Mult.	Type	Description
PaymentDetailedInformation		±	
+ PaymentProductUsed	0..1	Enum	IDEAL
+ UseWaitingScreen	0..1	Boolean	If true the Merchant should show a waiting screen even if the the RedirectUrl is provided. On the waiting screen a redirection button should be placed. Click on the button should redirect the PSU by the link from RedirectUrl.
+ CommonPaymentData	0..1	±	
++ ExpiryDateTimestamp	0..1	String	The timestamp from which the transaction operation will expire in ISODateTime standard, expressed in UTC time format(YYYY-MM-DDThh:mm:ss.sssZ)
++ GuaranteedAmount	0..1	String	The amount guaranteed by the Issuer to the Merchant. This is provided when minimum and maximum values are defined in the POST /payments request
++ PaymentStatus	1..1	Enum	The following statuses apply for iDEAL : • SettlementCompleted : corresponds to Success in iDEAL 1.0 • Cancelled • Expired • Open • Error : Corresponds Failure in iDEAL 1.0
++ PaymentId	1..1	String	ID of payment requested for status request. Max length : 35
+ AspspPaymentId	1..1	String	iDEAL Hub payment Id used to refer to the payment Max length : 35
+ AspspId	0..1	String	ID of the ASPSP as per PSD2. For iDEAL, these are default values: Sandbox : 10002 Production : 10000 Merchants can ignore this field.
+ DebtorInformation	0..1	±	Only received for payments with Status : SettlementCompleted
++ Name	0..1	String	Name of the Debtor
++ Agent	0..1	String	BIC of the financial institution servicing an account for the debtor.
++ Account	0..1	±	
+++ SchemeName	0..1	Enum	Type of Account Number
+++ Identification	0..1	String	Account identification number
++ ContactDetails	0..1	±	
+++ FirstName	0..1	String	The first name of the User
+++ LastName	0..1	String	The last name of the User
+++ PhoneNumber	0..1		Phone number of the User in E.164 format
+++ Email	0..1	String	The e-mail address of the User.
++ ShippingAddress	0..1	±	
+++ FirstName	0..1	String	The first name of the User.
+++ LastName	0..1	String	The last name of the User.
+++ CompanyName	0..1	String	The company name of the User.
+++ Postcode	0..1	String	Postal code without spaces.
+++ BuildingNumber	0..1	String	House Number
+++ StreetName	0..1	String	Street Name
+++ TownName	0..1	String	City. Example : Rotterdam
+++ Country	0..1	String	The code of the country regarding ISO 3166 standard. For the IDEAL payments the country name will be provided.
++ BillingAddress	0..1	±	
+++ FirstName	0..1	String	The first name of the User.
+++ LastName	0..1	String	The last name of the User.
+++ CompanyName	0..1	String	The company name of the User.
+++ Postcode	0..1	String	Postal code without spaces.
+++ BuildingNumber	0..1	String	House Number
+++ StreetName	0..1	String	Street Name
+++ TownName	0..1	String	City. Example : Rotterdam
+++ Country	0..1	String	The code of the country regarding ISO 3166 standard. For the IDEAL payments the country name will be provided

8.2.1.2 Response

As a response, the Open Banking service expects a HTML 204 with no content.

Header fields	Mult.	Type	Description
X-Request-ID	1..1	String	UUID for unique request identification. This is generated by the Merchant so that the request towards the Open Banking Service can be uniquely identified.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.

Body : HTML 204 (No Content)

8.2.2 Example: Notification for Standard iDEAL payment

As visible in the example, the URL will be appended with /notification/status if the Status Notification URL was provided ending with /3 in the API or in the iDEAL Subscription of the MSP Portal.

Signature-related fields "Digest" and "Signature" may be mandatory depending on the Acquirer. Please check the Acquirer Profile in the Getting Started Guide for more information:

8.2.2.1 Request

```
Address: https://checkout.company.com/transaction/webhook/91FA6EEC30844FAAB5/v3/notification/status
  HttpMethod: POST
  Headers: {Authorization=Bearer 123456789, X-Request-ID=c1452392-6c3f-4365-93f8-40558f61ac36,
MessageCreateDateTime=2023-03-15T11:51:24.185+01:00, Digest=SHA-
256=0hq1mKzxB1yyc6+hut2bEX7ps+nWyWb2pgQb6AhfhfM=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-sha256",headers="digest x-
request-id messagecreatedatetime (request-
target)",signature="uYgovoK+ibAE7+MzJEKrApDUAgWfUv7RQK22zAxWHCdKCuG4d0HgqpDSqcGlKmP2IMFsC787zDU3oqKeeIIVX
R72uZBiOnm0/84UL9e7LVDHDLQsRbfDnmvgX/4xQvdwROmyqh8kkcXTf/48zY0wo2n9iDspCbgTn1DEqAqtAlwunIpea8eYA3FQc+pV2p
x77wVP7l+9mTxexzLSmum61wWbqE4ESJn0K37gXY54229ZtCnNSlu9rsvjQ5xmDf1e6MvMLBOblXHIReN2t8IH85VGK7mpi8T7JeKb8rI
G8qDbQ5TD3BmIS1+RspI95FldLCKLH91/KNrxsgPsrC2QgQ==", Content-Type=application/json}
  Payload: {
  "PaymentProductUsed" : "IDEAL",
  "CommonPaymentData" : {
    "GuaranteedAmount" : "10.00",
    "PaymentStatus" : "SettlementCompleted",
    "PaymentId" : "19928",
    "AspspPaymentId" : "0001070883053837",
    "AspspId" : "10002",
    "DebtorInformation" : {
      "Name" : "Edsger Wybe Dijkstra - Callback",
      "Agent" : "ABNANL2AXXX",
      "Account" : {
        "SchemeName" : "IBAN",
        "Identification" : "NL44RABO0123456789",
        "Currency" : "EUR"
      },
  "UseWaitingScreen" : false
}
```

8.2.2.2 Response:

```
ResponseCode: 204
```

8.2.3 Example: Notification for iDEAL Payment with Fast Checkout

Signature-related fields "Digest" and "Signature" may be mandatory depending on the Acquirer. Please check the Acquirer Profile in the Getting Started Guide for more information:

8.2.3.1 Request

```
Address: https://checkout.company.com/transaction/webhook/91FA6EEC30844FAAB5/v3/notification/status
  HttpMethod: POST
  Headers: {Authorization=Bearer 123456789, X-Request-ID=c1452392-6c3f-4365-93f8-40558f61ac36,
MessageCreateDateTime=2023-03-15T11:51:24.185+01:00, Digest=SHA-
256=0hq1mKzxB1yyc6+hut2bEX7ps+nWyWb2pgQb6AhfhfM=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-sha256",headers="digest x-
request-id messagecreatedatetime (request-
target)",signature="uYgovoK+ibAE7+MzJEKrApDUAgWfUv7RQK22zAxWHCdKCuG4d0HgqpDSqcGlKmP2IMFsC787zDU3oqKeeIIVX
R72uZBiOnm0/84UL9e7LVDHDLQsRbfDnmvgX/4xQvdwROmyqh8kkcXTf/48zY0wo2n9iDspCbgTn1DEqAqtAlwunIpea8eYA3FQc+pV2p
x77wVP7l+9mTxexzLSmum61wWbqE4ESJn0K37gXY54229ZtCnNSlu9rsvjQ5xmDf1e6MvMLBOblXHIReN2t8IH85VGK7mpi8T7JeKb8rI
G8qDbQ5TD3BmIS1+RspI95FldLCKLH91/KNrxsgPsrC2QgQ==", Content-Type=application/json}
  Payload: {
  "PaymentProductUsed" : "IDEAL",
  "CommonPaymentData" : {
    "GuaranteedAmount" : "10.00",
    "PaymentStatus" : "SettlementCompleted",
    "PaymentId" : "19928",
    "AspspPaymentId" : "0001070883053837",
    "AspspId" : "RABONL2UXXX",
    "DebtorInformation" : {
      "Name" : "Edsger Wybe Dijkstra - Callback",
      "Agent" : "ABNANL2AXXX",
      "Account" : {
        "SchemeName" : "IBAN",
        "Identification" : "NL44RABO0123456789",
        "Currency" : "EUR"
      },
      "ContactDetails" : {
        "FirstName" : "Edsger",
        "LastName" : "Dijkstra",
        "PhoneNumber" : "+31612345678",
        "Email" : "edsger@domain.nl"
      },
    "ShippingAddress" : {
        "FirstName" : "Edsger",
        "LastName" : "Dijkstra",
        "PostCode" : "52066",
        "Country" : "NL"
        },
    "BillingAddress" : {
        "FirstName" : "Edsger",
        "LastName" : "Dijkstra",
        "PostCode" : "52066",
        "Country" : "NL"
        }
    }
  },
  "UseWaitingScreen" : false
}
```

8.2.3.2 Response:

```
ResponseCode: 204
```

9 Payment Finalisation - Payment Status

9.1 GET Payment Status

This endpoint is used to retrieve the status of a payment.

Endpoint: GET **{Base Domain}**/xs2a/routingservice/services/ob/pis/v3/payments/{paymentId}/status
Full URL for test environment
: https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments/{paymentId}/status
Full URL for Production environment : Please refer to the Getting Started Guide because the Base Domain may defer depending on the Acquirer.

More information on the applicable fields is available in the tables below. These tables should be used as supporting material to the Swagger Files.

9.1.1 Request

Path parameter	Mult.	Type	Description
paymentId	1..1	String	Transaction ID which is used by the Open Banking Service to refer to the payment.

Headers fields	Mult.	Type	Description
Authorization	1..1	String	Valid access token as retrieved from the Authorization API. This is mandatory in order to use the other end points of the Open Banking Service.
X-Request-ID	1..1	String	UUID for unique request identification.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.
Signature	0..1	String	May be mandatory depending on Acquirer. Information on whether it is mandatory is available in the Getting Started Guide. The signature in the request should include a keyId with the value of fingerprint of the certificate.
Digest	0..1	String	Is contained if and only if the Signature element is contained in the header of the request.

9.1.2 Response

Headers fields	Mult.	Type	Description
X-Request-ID	1..1	String	UUID for unique request identification.
MessageCreateDateTime	1..1	String	The message create date time. ISO 8601 DateTime.
Signature	0..1	String	May be present depending on Acquirer. Information on whether it is mandatory is available in the Getting Started Guide. The signature in the request should include a keyId with the value of fingerprint of the certificate.
Digest	0..1	String	Is contained if and only if the Signature element is contained in the header of the request.

Body fields	Mult.	Type	Description
PaymentProductUsed	0..1	Enum	IDEAL will be returned
CommonPaymentData	1..1	±	
+ ExpiryDateTimeStamp	0..1	string	The timestamp from which the transaction operation will expire, expressed in UTC time format(YYYY-MM-DDThh:mm:ss.sssZ)
+ GuaranteedAmount	0..1	string	The amount guaranteed by the Issuer to the Merchant. This is provided if the minimum and maximum amount values are defined in the POST /payments request
+ PaymentStatus	1..1	Enum	The following statuses apply for iDEAL : • SettlementCompleted : Success in iDEAL 1.0 • Cancelled • Expired • Open • Error : Failure in iDEAL 1.0
+ PaymentId	1..1	String	ID of payment requested for status request. Max length : 35
+ AspspPaymentId	1..1	String	iDEAL Hub payment Id used to refer to the payment Max length : 35
+ AspspId	0..1	String	ID of the ASPSP as per PSD2. For iDEAL, these are default values: Sandbox/Test Environment : 10002 Production : 10000 Merchants can ignore this field.
+ DebtorInformation	0..1	±	
++ Name	0..1	String	Name of the Debtor
++ Agent	0..1	String	BIC of the financial institution servicing an account for the debtor.
++ Account	0..1	±	
+++ SchemeName	0..1	Enum	Type of Account Number
+++ Identification	0..1	String	Account identification number
++ ContactDetails		±	
+++ FirstName	0..1	String	The first name of the User
+++ LastName	0..1	String	The last name of the User
+++ PhoneNumber	0..1		Phone number of the User in E.164 format
+++ Email	0..1	String	The e-mail address of the User
++ ShippingAddress		±	
+++ FirstName	0..1	String	The first name of the User
+++ LastName	0..1	String	The last name of the User
+++ CompanyName	0..1	String	The company name of the User
+++ Postcode	0..1	String	Postal code without spaces.
+++ BuildingNumber	0..1	String	House Number
+++ StreetName	0..1	String	StreetName
+++ TownName	0..1	String	City. Example : Rotterdam
+++ Country	0..1	String	The code of the country regarding ISO 3166 standard. For the IDEAL payments the country name will be provided
++ BillingAddress	0..1	±	
+++ FirstName	0..1	String	The first name of the User
+++ LastName	0..1	String	The last name of the User
+++ CompanyName	0..1	String	The company name of the User
+++ Postcode	0..1	String	Postal code without spaces.
+++ BuildingNumber	0..1	String	House Number
+++ StreetName	0..1	String	StreetName
+++ TownName	0..1	String	City. Example : Rotterdam
+++ Country	0..1	String	The code of the country regarding ISO 3166 standard. For the IDEAL payments the country name will be provided

9.2 Example: iDEAL payment status

Signature-related fields "Digest" and "Signature" may be mandatory depending on the Acquirer. Please check the Acquirer Profile in the Getting Started Guide for more information:

9.2.1 Request

```
    Address:
https://digitalroutingservice.awltest.de/xs2a/routingservice/services/ob/pis/v3/payments/143374/status
    HttpMethod: GET
    Headers: {Accept=application/json, Digest=SHA-256=47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=, X-
Request-ID=aa9ec3e7-dc34-f625-2e97-e16644ea1e77, Authorization=Bearer
59519322f1f56db6bd1d23ac0e22cf5015088f085970fb9b461ff2684bc2e96a, MessageCreateDateTime=2024-01-
08T12:55:34.630Z,
Signature=keyId="39d8e82bb33e7e2a09cbcb3ef3eab351ee1c5e8f",algorithm="SHA256withRSA",headers="x-request-
id (request-target) digest
messagecreatedatetime",signature="W/6ViHZJ/x0/KM61H7mwCZQp2/3WPax5seqLpCby0BXwgpVoFrSmCu5oHF2pLS2AdHuTiE6
qFsBnMkfeGMEDwtqCdY0GZVQGwZ6j9+6zjJHwm21xZ+BmXXH4puRgqxt7DSnOou3SkstzCblf4gSL8MqIyYu7n+eLQlWvjl57t7dNgX0s
NTwJE/1GgU/ukDyUoJl4aO1n9ID8EnLDUpXunbZ+1pyCPGbfz5/pDMrJsKZ9l58tLxj7IneU+Mnai8UJEqrQ441ONgYCHF7KoDxu15XA+
IY7DnwkA5u/+QN7IY5KpHG3zP19m7jiOimkI6lrqLAc2Ld4iehjNMGve7Ci7w=="}
```

9.2.2 Response

```
    ResponseCode: 200

    Headers: {X-Request-ID=3a3df5d3-fa9e-4fee-b9fd-067c23dc91fa, MessageCreateDateTime=2024-01-
08T12:55:35.032Z, Digest=SHA-256=pV9HQh/XGpLmawhfj9d/hxYucKkOQCKV9BJ978PeW5k=,
Signature=keyId="3EBEF6033C00730D9C6DA05165A3CAA1F31036FB",algorithm="rsa-
sha256",headers="messagecreatedatetime x-request-id
digest",signature="DMvxRTvhRG1kFAy0JPbBzdoRrXq5C9OUJHl6GMQkdNriG4SeuHGaFpdaDA6NgFw2+Ky/RBYiruL18ul+1Zvmdh
lyhMaRqR0J7Jg/m9JMyu3wFunBVINysQ+2yj0ucXRMck/CoBW1mYwjDCGCN7xOofw2mZLV6PHjP9xHaDA855iPYE7HUaukB/qveA/S6B2
qsjFxDs5AJh6VSWEoknA1+NJZCEX2VC23bnkMgy92qN6bZch5bL3462yFA/0uA75k9XH3O7+Jesl86tBU3kKATNk5n2//vun+Bg4NlSQj
U21m6B+nOZT1M+TUkVzG0KWdByf6yFv/x0rp7hifnFZvjA==", Date=Mon, 08 Jan 2024 12:55:35 GMT, Content-
Type=application/json;charset=UTF-8}
    Payload: {
    "PaymentProductUsed": "IDEAL",
    "CommonPaymentData": {
        "PaymentStatus": "SettlementCompleted",
        "PaymentId": "143374",
        "AspspPaymentId": "0001092688873027",
        "AspspId": "10002",
        "DebtorInformation": {
            "Name": "Edsger Wybe Dijkstra - Callback",
            "Agent": "ABNANL2AXXX",
            "Account": {
                "SchemeName": "IBAN",
                "Identification": "NL44RABO0123456789"
            }
        }
    }
}
```

9.3 Currence Requirements

9.3.1 Retrieving the Status of the transaction

- If the Merchant has not yet received a confirmation of the payment status when the User opens the Return URL, the Merchant MUST retrieve the transaction status by calling the status API of the Open Banking Service.
- The Merchant MUST inform the User of a success status of the transaction, so that the User is certain that the payment status was correctly received by the Merchant.