Partner
in Payments

User Guide

SIBS API Market

PSD2 APIs

Third party payment service providers (TPP)

Version: 01.40

Date: 2024-03-21

Status: Final

Classification: Restricted

Reference: DCSIBS230100

Document Info

Reference:	DCSIBS230100
Document Title:	SIBS API Market - PSD2 APIs
Version:	01.40
Status:	Final
Classification:	Restricted
Document Type:	User Guide

Related Documents

Reference	Title	Source
N/A	N/A	N/A

Version History

Version	Date	Description	Author
01.04	2019-02-25	First version for public release	UPIDP
01.05	2019-03-12	Changes to this version: • Table in section 2.10 ASPSP's specific options updated • API endpoints' address included in section 2.2 API endpoints' address structure updated	UPIDP
01.06	2019-04-10	Changes to this version: • New chapter "message signing" included • Adjustments made on chapters 2.2 API endpoints' address structure and 2.3 Character Set.	UPIDP
01.07	2019-08-20	Changes to this version: • Character set has been updated in section 2.3 • Section 2.7 renamed and updated • New Section 2.9 • Section 2.10 updated • Clarification on Digest computation on section 4.2 • Date format of the example message amended on section 4.4	UPIDP
01.08	2019-12-12	Changes to this version: • Character set has been updated • Included information on MULTIBANCO Payment APIs • Section 2.9 and 2.10 were updated	UPIDP
01.10	2020-11-25	Changes to this version: • New sections 2.4 - reference to Message Codes, 2.11 App-to-app redirection and 2.12, Account Information API - Interpretation of Balances Fields for Card Accounts were included • New chapter 3 Developers Portal Functionalities was included • Section 2.9 was updated • Section 2.10 was updated • Section 6.2 was updated	UPIDP
01.20	2021-03-23	Changes to this version: • Section 2.9 was updated	UPIDP
01.21	2021-05-27	Changes to this version: • Section 2.9 was updated	UPIDP
01.22	2021-12-17	Changes to this version: • Section 2.9 was updated • Section 6.1 was updated • Section 2.13 was created	DGPIA
01.23	2023-03-31	Changes to this version:	DGPIA

Version	Date	Description	Author
		• Section 2 was updated, namely the subsections 2.2 and 2.10 • Section 6.1 and its subsections were updated • Section 6.2 and its subsections were updated • Section 6.3 was created • Section 7 was created	
01.30	2023-09-14	Changes to this version: • Section 6.2 was updated; • Section 7 was updated.	DGP
01.40	2024-03-21	Changes to this version: • Table 1 of section 2.7 was updated; • Table 3 of section 2.10 was updated; • Section 7 was updated.	DGP

Table of Contents

List of Figures

List of Tables

1 Introduction

PSD2 APIs provided in SIBS API Market allow account information service providers (AISP), payment initiation service providers (PISPs) and payment service providers issuing card-based payment instruments (CBPIIs), collectively known as third party payment service providers or TPPs, that are authorized by a National Competent Authority under [PSD2] scope, access payment accounts on account servicing payment service providers (ASPSPs) that have selected SIBS API Market to open their accounts to these new players.

The APIs under [PSD2] scope are:

- Account Information – can be used by AISPs;
- Payment Initiation – can be used by PISPs;
- Funds Confirmation – can be used by CBPIIs and PISPs.

With just one integration, SIBS API Market allows TPPs to reach all ASPSPs listed in section 2, through a common set of APIs, covering more than 95% of the payment accounts held by PSUs in Portugal. The list of ASPSPs can also be retrieved through the List of banks API available on SIBS API Market. The intended ASPSP is selected by the TPP, on each API call, in a parameter that is part of the path of the API endpoint.

1.1 Objective

The objective of this document is to provide information on the usage of PSD2 APIs available on SIBS API Market to TPPs.

1.2 Scope

This document covers the implementation options taken by the ASPSPs on SIBS API Market in relation to the reference specifications, and provides guidance on the sequence in which the operations of each API shall be executed.

The detailed specification of the APIs and API parameters are out of scope of this document.

1.3 References

Reference	Title
[BG-IG]	The Berlin Group Joint Initiative on a PSD2 Compliant XS2A Interface; NextGenPSD2 XS2A Framework; Implementation Guidelines; Version 1.3.12; 01 July 2022.
[eIDAS]	REGULATION (EU) No 910/2014 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC.
[ETSI-PSD2]	TECHNICAL SPECIFICATION ETSI TS 119 495 - Electronic Signatures and Infrastructures (ESI); Sector Specific Requirements; Qualified Certificate Profiles and TSP Policy Requirements under the payment services Directive (EU) 2015/2366.
[PSD2]	DIRECTIVE (EU) 2015/2366 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 25 November 2015 on payment services in the internal market.
[RTS]	COMMISSION DELEGATED REGULATION (EU) 2018/389 of 27 November 2017 supplementing Directive (EU) 2015/2366 of the European Parliament and of the Council with regard to regulatory technical standards for strong customer authentication and common and secure open standards of communication.

1.4 Definitions

For the purpose of this document the following definitions apply:

AISP	Account Information Service Provider as defined in [PSD2].
ASPSP	Account Servicing Payment Service Provider as defined in [PSD2] (e.g. Banks).
CBPII	Card-Based Payment Instrument Issuer as defined in [PSD2].
NCA	Nacional Competent Authority as defined in [PSD2].
PISP	Payment Initiation Service Provider as defined in [PSD2].
PSU	Payment Services User as defined in [PSD2].
QTSP	Qualified Trust Service Provider as defined in [eIDAS].
SCA	Strong Customer Authentication as defined in [RTS].
TPP	Third Party Payment Services Provider authorised by a National Competent Authority to provide payment services according to one or more of the following roles defined in [PSD2]: • Payment Initiation Service Provider (PISP); • Account Information Service Provider (AISP); • Card-Based Payment Instrument Issuer (CBPII).

2 Implementation options

The specification of the PSD2 APIs provided by all ASPSPs on SIBS API Market is based on the version 1.3.12. (July 2022) of NextGenPSD2 XS2A Framework of Berlin Group [BG-IG].

The NextGenPSD2 Guidelines leave to ASPSPs the decision to implement some optional features. This section identifies options taken on the implementation of SIBS API Market for the PSD2 APIs.

2.1 Qualified certificates

For the purpose of identification on access to PSD2 APIs on SIBS API Market, TPPs shall use a qualified certificate for electronic seals (QSealC) and a qualified certificate for website authentication (QWAC) with a PSD2 profile according to [ETSI-PSD2], issued by a QTSP recognised by a competent authority under [eIDAS] regulation. All request messages toward SIBS API Market API shall be signed with the private key related with the public key included in the QSeal Certificate (see details in section 2.11). The QWAC shall be used to establish the secure channel for communication between the TPP and SIBS API Market using the Transport Layer Security (TLS) protocol.

The above-mentioned certificates are not required for testing on Sandbox environment of SIBS API Market.

After applying for authorisation to a national competent authority, and while the competent authority does not grant the authorisation, TPPs may use test certificates on Test/Production environment of SIBS API Market, while in testing mode.

2.2 API endpoints' address structure

API endpoints' address follows the general structure defined in [BG-IG]:

- https://{provider}/v1/{service}{?query-parameters}

where

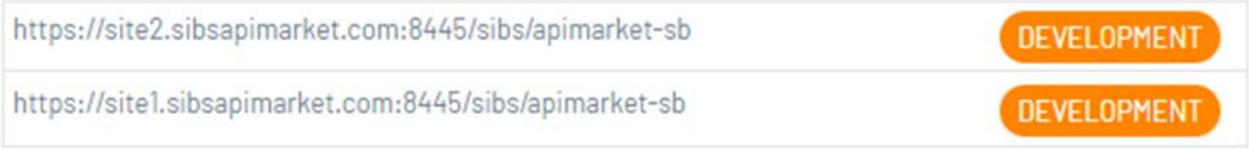
- {provider} is composed by {host}/{path}/{aspsp-cde}

SIBS API Market infrastructure is split between two redundant active/active sites. The {host}/{path} part of the API endpoints' address defines the site and environment (development or production) that will process the API call. The values to use in the {host}/{path} part of the API endpoints are available in the API documentation of each API on the Sandbox and Test & Production environments.

Sandbox environment only provides Development endpoints for site 1 and site 2.

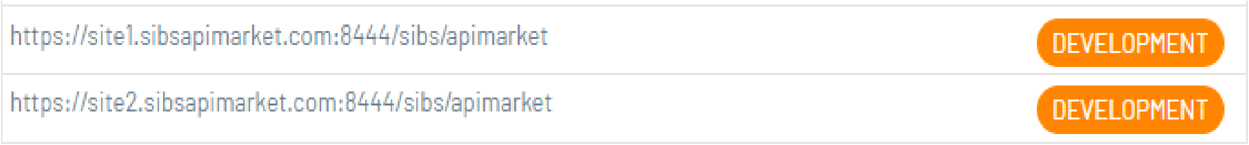
Test & Production environment provides Development and Production endpoints for site 1 and site 2.

Example of {host}/{path} part of the Payment Initiation API endpoints for tests on Sandbox environment of SIBS API Market:



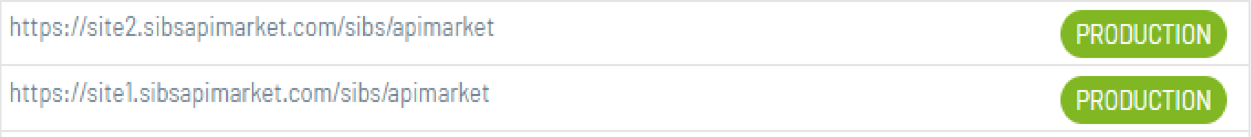
https://site2.sibsapimarket.com:8445/sibs/apimarket-sb	DEVELOPMENT
https://site1.sibsapimarket.com:8445/sibs/apimarket-sb	DEVELOPMENT

Example of {host}/{path} part of the Payment Initiation API endpoints for tests end-to-end with the ASPSPs on Test & Production environment of SIBS API Market:



https://site1.sibsapimarket.com:8444/sibs/apimarket	DEVELOPMENT
https://site2.sibsapimarket.com:8444/sibs/apimarket	DEVELOPMENT

Example of {host}/{path} part of the Payment Initiation API endpoints for production on Test & Production environment of SIBS API Market:



https://site2.sibsapimarket.com/sibs/apimarket	PRODUCTION
https://site1.sibsapimarket.com/sibs/apimarket	PRODUCTION

The {aspsp-cde} part of the API endpoints' address defines the ASPSP the TPP wants to call, for the provisioning of the service requested in the {service} part of the API endpoint. The possible values for aspsp-cde can be obtained through the List of Banks API (e.g. BBPI, BNKI, BST and CCAML).

Example to call an API on the Sandbox environment of the ASPSP CCAML via site 1:

- https://site1.sibsapimarket.com:8445/sibs/apimarket-sb/CCAML/...

2.3 Character set

SIBS API Market accepts the following character set in messages:

- a b c d e f g h i j k l m n o p q r s t u v w x y z
- A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
- 0 1 2 3 4 5 6 7 8 9
- / - ? : ( ) . , ' + < = > & % _ * #
- Space

Messages bearing characters outside this set are declined, except for the parameter TPP-Certificate where all characters are allowed.

2.4 Message codes

The list of the possible message codes that can be returned by the APIs is published in the Developers Portal, and is based on the Berlin Group implementation guidelines.

2.5 Account reference

Berlin Group's NextGen PSD2 framework allows the use of several different type of references for payment accounts / cards (e.g.: iban, pan, msisdn). SIBS API Market implementation supports "iban" and "bban" for account references, and "pan" and "mpan" for card-account references.

2.6 Options not supported

The following features included in [BG-IG] are not supported by ASPSPs on SIBS API Market:

- OAuth2 protocol for PSU authentication;
- Balances on list of accounts (GET accounts?withBalance);
- Balances on account details (GET accounts/{account-id}?withBalance);
- Balances on list of transactions (GET accounts/{account-id}/transactions ?withBalance);
- Transaction details (GET accounts/{account-id}/transactions/{transaction-id});
- Delta access on list of transactions (GET accounts/{account-id}/transactions ?transactionId and GET accounts/{account-id}/transactions?deltaList).

2.7 Supported APIs

All APIs included in [BG-IG] are, or will be, supported on Sandbox and Test & Production environments of SIBS API Market in a phased approach according to the plan defined in Table 1.

APIs supported on Sandbox environment are available on all ASPSP.

APIs supported on Test & Production environment may not be available on all ASPSP depending on their own implementation plans and support of the same operation on their direct PSU interfaces (see section 2.10).

The APIs, as well as the versions of the APIs, available on each ASPSP may be retrieved using the List of banks API, available in the Information APIs product of SIBS API Market.

The list of available APIs on each ASPSP provided by the List of banks API may differ on the environment and, for the Test and Production environment, on the Test and Production endpoints, depending on the APIs each ASPSP has implemented in each environment/endpoint.

The List of banks API on Sandbox environment returns the list of available APIs, and API versions, on each ASPSP on the Sandbox Environment for testing in closed loop with static data.

The List of banks API on Test (DEVELOPMENT) endpoints of Test & Production environment returns the list of available APIs, API versions, payment-product (e.g.: SEPA CT), on each ASPSP for testing end-to-end with the ASPSP using non-real PSU accounts/cards.

The List of banks API on Production (PRODUCTION) endpoints of Test & Production environment returns the list of available APIs, API versions, payment-product (e.g.: SEPA CT), on each ASPSP for real PSU accounts/cards access.

The payment products of the payments are also returned by the List of Banks API.

Table 1 - SIBS Market support for Berlin Group APIs

API	Operation	Sandbox	Test & Production
payments/{payment-product}	Initiation	2019Q1	2019Q1/Q2[1]
payments/{payment-product}	Cancelation	2019Q1	2019Q3[1]
periodic-payments/{payment-product}	Initiation	2019Q1	2019Q2[1]
periodic-payments/{payment-product}	Cancelation	2019Q1	2019Q3[1]
consents	All	2019Q1	2019Q1
accounts	All	2019Q1	2019Q1
funds-confirmations	For CBPII/PISP	2019Q1	2019Q1
bulk-payments/{payment-product}	All	2020Q1	2020Q1
cards accounts	All	2022Q3/Q4	2023Q4[1]
consent authorisation	All	2022Q3/Q4	2023Q4

[1] Availability of the API and payment products may differ among ASPSPs. Check above how to use the List of banks API to know what ASPSPs have implemented this API. Check section 2.10 to know what products are available in each ASPSP.

2.8 Payment product fallback on payment initiation APIs

Whenever both Debtor payment account/card account and Creditor payment account/card account are held by the same ASPSP providing the payment initiation service, the payment product used to perform the payment may be, depending on the ASPSP, switched to an internal credit transfer, and the funds immediately available to the Creditor, regardless the value of the path parameter "payment-product".

Whenever both Debtor payment account/card account and Creditor payment account/card account are held by ASPSPs belonging to the SEPA space and the currency is Euro, the payment product used to perform the payment may be, depending on the ASPSP providing the payment initiation service, switched to a SEPA CT payment product, when the path parameter "payment-product" is "cross-border-credit-transfers".

In both cases, the rules applied to the transaction (including the cost to the Debtor and/or Creditor) are the same as for the internal credit transfer and SEPA CT payments performed on the ASPSP's channels (e.g.: homebanking).

2.9 Domestic payment products APIs

Besides [BG-IG] APIs, SIBS API Market provide payment initiation APIs, in the scope of PSD2, for payment products specific to the Portuguese market.

These APIs are supported on Sandbox and Test & Production environments of SIBS API Market according to Table 2.

The multibanco-payment-type and service-payment-name path parameters are returned by the MULTIBANCO Payments Catalogue API (GET multibanco-payments/service-catalogue).

Table 2 - SIBS Market support for domestic payments APIs

API
multibanco-payments/service-catalogue
multibanco-payments/{multibanco-payment-type}/{service-payment-name}
periodic-multibanco-payments/{multibanco-payment-type}/{service-payment-name}
bulk-multibanco-payments/{multibanco-payment-type}
tsu-payments/{payment-product}
bulk-tsu-payments/{payment-product}

2.10 ASPSP's specific options

On Sandbox environment the API data is static and the implemented options are the same for all ASPSPs.

On Test and Production environments some features available on SIBS API Market are not provided by some ASPSPs, when not provided on their online interfaces for the PSUs.

The following table provides the list of supported features, on Test and Production environments, by each of the ASPSPs present on SIBS API Market for Private and Corporate payment accounts.

API List of Banks provides you real time information about the features available in each ASPSP and should always be the reference.

Table 3 - ASPSPs present on SIBS API Market - supported features on Test and Production environments

	Supported payment products[1] for single payment	Supported authentication approaches	Support for combined AI/PI access	Supported payment products[1] for future dated payments	Supported payment products[1] for periodic payments	Supported "executionRule" for periodic payments	Supported "frequency" for periodic payments	Support of "dayOfExecution"[2] for periodic payments	Supported payment products[1] for bulk payments	Support of App-to-app redirection	Support of Chargebearer as mandatory field
ActivoBank by Millennium	• SEPA CT • SCT Inst • TARGET • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Yes	• SEPA CT • SCT Inst	• SEPA CT • SCT Inst	• Preceding • Following	• Daily • Weekly • Monthly • Every two months • Quarterly • Semiannual • Annual	No	No	Yes	No
ATLANTICO BANCO ATLANTICO EUROPA	• SEPA CT • SCT Inst • TARGET • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Yes	• SEPA CT	• SEPA CT	• Following	• Weekly • Monthly • Quarterly • Semiannual • Annual	Yes	No	Yes	No

	Supported payment products[1] for single payment	Supported authentication approaches	Support for combined AI/PI access	Supported payment products[1] for future dated payments	Supported payment products[1] for periodic payments	Supported “executionRule” for periodic payments	Supported “frequency” for periodic payments	Support of “dayOfExecution”[2] for periodic payments	Supported payment products[1] for bulk payments	Support of App-to-app redirection	Support of Chargebearer as mandatory field
bankinter.	• SEPA CT • SCT Inst • TARGET • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc.	• Redirect	Not supported	• SEPA CT • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc.	• SEPA CT • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc.	• Preceding • Following	• Weekly • Every two weeks • Monthly • Every two months • Quarterly • Semiannual • Annual	Not supported	• SEPA CT	Yes	Not supported
BiG BANCO DE INVESTIMENTO GLOBAL	• SEPA CT	• Redirect	Not supported	• SEPA CT	• SEPA CT	• Following	• Daily • Weekly • Monthly • Annual	Yes	Not supported	Not supported	Not supported
BiG BANCO DE INVESTIMENTO GLOBAL Espanha	• SEPA CT	• Redirect	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
BPI	• SEPA CT • SCT Inst • TARGET • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT	• SEPA CT	• Following	• Daily • Weekly • Every two weeks • Monthly • Every two months • Quarterly • Semiannual • Annual	Yes	• SEPA CT • TARGET	Yes	Not supported
Banco Português de Gestão	• SEPA CT	• Redirect	Yes	• SEPA CT	• SEPA CT	• Following	• Daily • Weekly • Every two weeks • Monthly • Every two months • Quarterly • Semiannual • Annual	Yes	Not supported	Not supported	Not supported

	Supported payment products[1] for single payment	Supported authentication approaches	Support for combined AI/PI access	Supported payment products[1] for future dated payments	Supported payment products[1] for periodic payments	Supported “executionRule” for periodic payments	Supported “frequency” for periodic payments	Support of “dayOfExecution”[2] for periodic payments	Supported payment products[1] for bulk payments	Support of App-to-app redirection	Support of Chargebearer as mandatory field
CA Crédito Agrícola	• SEPA CT • SCT Inst • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Yes	• SEPA CT • Pag. de Serviços	• SEPA CT	• Following	• Daily • Weekly • Every two weeks • Monthly • Quarterly • Semiannual • Annual	Yes	• SEPA CT • Pag. TSU	Yes	Not supported
Caixa de Crédito de Leiria	• SEPA CT • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	Not supported	Not supported	Not supported	Not supported	• SEPA CT • Pag. Seg. Soc. • Pag. TSU	Not supported	Not supported
Caixa Agrícola Bombarral	• SEPA CT • Pag. Seg. Soc. • Pag. TSU	• Redirect	Yes	Not supported	Not supported	Not supported	Not supported	Yes	Not supported	Not supported	Not supported
Caixa de Crédito da Chamusca	• SEPA CT • Pag. Seg. Soc. • Pag TSU	• Redirect	Yes	• SEPA CT	• SEPA CT	• Preceding • Following	• Daily • Monthly • Quarterly • Semiannual • annual	Yes	Pag. TSU	Not supported	Not supported
Caixa Agrícola de Mafra	• SEPA CT • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Yes	Not supported	Not supported	Not supported	Not supported	Yes	Not supported	Not supported	Not supported
Caixa Agrícola de Torres Vedras	• SEPA CT • Pag. Seg. Soc. • Pag. TSU	• Redirect	Yes	Not supported	Not supported	Not supported	Not supported	Yes	Not supported	Not supported	Not supported

	Supported payment products[1] for single payment	Supported authentication approaches	Support for combined AI/PI access	Supported payment products[1] for future dated payments	Supported payment products[1] for periodic payments	Supported "executionRule" for periodic payments	Supported "frequency" for periodic payments	Support of "dayOfExecution"[2] for periodic payments	Supported payment products[1] for bulk payments	Support of App-to-app redirection	Support of Chargebearer as mandatory field
CEM CAIXA ECONÓMICA DA MISERICÓRDIA DE ANGRA DO HEROÍSMO	• SEPA CT • SCT Inst • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT • Pag. de Serviços • Carreg. Telemóveis • Pag. TSU	• SEPA CT • Pag. de Serviços • Carreg. Telemóveis	• Following	• Daily • Weekly • Every two weeks • Monthly • Quarterly • Semiannual • Annual	Yes	• SEPA CT	Yes	Not supported
	• SEPA CT • SCT Inst • TARGET • Cross Border[3] • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis	• SEPA CT • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis	• Preceding	• Weekly • Every two weeks • Monthly • Every two months • Quarterly • Semiannual • Annual	Yes	• SEPA CT • Cross Border • Pag. de Serviços • Carreg. Telemóveis	Yes	Mandatory
Caixa Geral de Depositos France	• SEPA CT	• Redirect	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
Cofidis De pessoas para pessoas	• SEPA CT	• Redirect	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
EuroBic	• SEPA CT • SCT Inst • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT	• SEPA CT	• Following	• Weekly • Every two weeks • Monthly • Quarterly • Semiannual • Annual	Yes	SEPA CT[4,5]	Yes	Not supported (Cross border Transaction always paid by the sender)

	Supported payment products[1] for single payment	Supported authentication approaches	Support for combined AI/PI access	Supported payment products[1] for future dated payments	Supported payment products[1] for periodic payments	Supported "executionRule" for periodic payments	Supported "frequency" for periodic payments	Support of "dayOfExecution"[2] for periodic payments	Supported payment products[1] for bulk payments	Support of App-to-app redirection	Support of Chargebearer as mandatory field
Millennium bcp	• SEPA CT • SCT Inst • TARGET • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Yes	• SEPA CT • SCT Inst • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis	• SEPA CT • SCT Inst • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis	• Preceding • Following	• Weekly • Monthly • Every two months • Quarterly • Semiannual • Annual	Not supported	• SEPA CT • SCT Inst • TARGET • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. TSU	Yes	Not supported
Banco Montepio	• SEPA CT • SCT Inst • TARGET • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT • Pag. de Serviços • Pag. ao Estado • Pag. TSU	• SEPA CT	• Following	• Daily • Weekly • Monthly • Every two months • Quarterly • Semiannual • Annual	Yes	• SEPA CT • SCT Inst	Yes	Not supported
novobanco	• SEPA CT • SCT Inst • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT	• SEPA CT	• Following	• Weekly • Every two weeks • Monthly • Every two months • Quarterly • Semiannual • Annual	Not supported	• SEPA CT	Yes	Not supported
novobanco DOS AÇORES	• SEPA CT • SCT Inst • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT	• SEPA CT	• Following	• Weekly • Every two weeks • Monthly • Every two months • Quarterly • Semiannual • Annual	Not supported	• SEPA CT	Yes	Not supported

	Supported payment products[1] for single payment	Supported authentication approaches	Support for combined AI/PI access	Supported payment products[1] for future dated payments	Supported payment products[1] for periodic payments	Supported "executionRule" for periodic payments	Supported "frequency" for periodic payments	Support of "dayOfExecution"[2] for periodic payments	Supported payment products[1] for bulk payments	Support of App-to-app redirection	Support of Chargebearer as mandatory field
Santander	• SEPA CT • SCT Inst • TARGET • Cross Border • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• Redirect	Not supported	• SEPA CT • SCT Inst • Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis • Pag. Seg. Soc. • Pag. TSU	• SEPA CT • SCT Inst	• Following	• Weekly • Monthly • Every two months • Quarterly • Semiannual • Annual	Not supported	• SEPA CT • SEPA CT Urg. • Cross Border • Pag. de Serviços • Pag. ao Estado	Yes	Cross Border Single Payment: Not mandatory. If empty, Santander assumes SHA (shared) by default. Cross Border Bulk Payment: Mandatory.
Unicre	• Pag. de Serviços • Pag. ao Estado • Carreg. Telemóveis	• Redirect	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported

Note 1: SEPA CT: sepa-credit-transfers; Cross Border: cross-border-credit-transfers; SCT Inst: instant-sepa-credit-transfers; TARGET: target-2-payments.

Note 2: A payment's date can be adjusted accordingly to the execution rule, the same rule the ASPSP uses on his own channels (e.g. homebanking).

Note 3: At CGD, for cross-border payments, it is mandatory to fill the fields for the debtor and creditor address, in accordance with ISO20022 usage guidelines.

Note 4: EuroBic has a maximum bulk payments file size equivalent to 3300 payments.

Note 5: EuroBic only accepts batchBookingPreferred='true' or <empty> for Bulk Payments.

2.11 App-to-app redirection

The ASPSP that offers an app to their users will make app-to-app redirection available to TPPs and support the authentication methods available in their own channels (e.g.: biometrics).

ASPSP present in the SIBS API Market that have already implemented app-to-app redirection, support Universal Links in their iOS apps (https://developer.apple.com/ios/universal-links/) and App Links in their Android apps (https://developer.android.com/training/app-links) to handle app-to-app redirection for PSU's authentication. See column "Support of App-to-app redirection" of the table included in section 2.10 to know who are the ASPSPs that already support app-to-app redirection.

2.11.1 Activating ASPSP's app

To activate the ASPSP's app to authenticate the PSUs, TPPs shall request the Operating System to open the link returned by the ASPSPs using Universal Links and App Links mechanisms.

TPPs currently using Web Views to open the links returned by the ASPSPs, shall change to the referred mechanisms when opening the links returned by the "app-to-app ready" ASPSPs, in order to benefit from app-to-app redirection.

Examples hereafter show how to open links returned by ASPSPs depending on the operating System (the example URLs shall be replaced by the URLs returned).

iOS example (available in https://developer.apple.com/documentation/uikit/inter-process_communication/allowing_apps_and_websites_to_link_to_your_content)

```
if let appURL = URL(string:
"https://myphotoapp.example.com/albums?albumname=vacation&index=1") {
    UIApplication.shared.open(appURL) { success in
        if success {
            print("The URL was delivered successfully.")
        } else {
            print("The URL failed to open.")
        }
    }
} else {
    print("Invalid URL specified.")
}
```

Android example

```
Intent intent = new Intent (Intent.ACTION_VIEW);
intent.setData(Uri.parse("https://myphotoapp.example.com/albums?albumname=vacation&index=1"));
startActivity (intent);
```

2.11.2 Returning to TPP's app

When the PSU is authenticated, ASPSP's app opens the callback URL sent by the TPPs in the API call (e.g. POST https://site1.sibsapimarket.com/sibs/apimarket/{aspsp-cde}/{v}/payments/sepa-credit-transfers), using the same mechanism used by the TPP to open ASPSP's app, as described in section 2.11.1.

TPPs must associate its callback URL to its app, and implement the handler in the app to receive the redirect back, as described in https://developer.apple.com/ios/universal-links/ for iOS and https://developer.android.com/training/app-links for Android. This way, by the time the PSU is authenticated the ASPSP redirects him back to the TPP's app, which is associated to the TPP's URL previously sent by the TPP.

2.12 Account information API - Interpretation of balances fields for card accounts

When receiving information of a card account through the account information API, the TPP should interpret the balances fields available as follows:

- **interimAvailable** – Available balance at the moment, calculated in the course of the account servicer's business day, at the time specified, and subject to further changes during the business day. The interim balance is calculated on the basis of booked credit and debit items during the calculation time/period specified;
- **authorized** – Credit limit, calculated by adding the expected[1] balance with the value of a pre-approved credit line the ASPSP makes permanently available to the user;
- **closingBooked** – Statement balance, the balance of the account at the end of the pre-agreed account reporting period. It is the sum of the opening booked[2] balance at the beginning of the period and all entries booked to the account during the pre-agreed account reporting period.

[1] Expected - Balance composed of booked entries and pending items known at the time of calculation, which projects the end of day balance if everything is booked on the account and no other entry is posted.

[2] Opening booked - Book balance of the account at the beginning of the account reporting period. It always equals the closing book balance from the previous report.

2.13 Account information API – Navigation fields

TPPs shall not perform any validation or processing to the content of the parameters ‘first’, ‘previous’, ‘next’ and ‘last’ returned by SIBS API Market for navigation on the transactions pages returned by the GET transactions API. Navigation links are valid during 30 minutes and shall be used without changes to access the transactions pages on further requests.

The parameters used in these links are for SIBS internal use and may be changed at any time without notice.

3 Developers Portal functionalities

3.1 Support tickets

SIBS API Market website includes a page where every TPP has the chance of reporting an issue or doubt about the available APIs. The issues are tracked through every iteration between SIBS, the TPP who opened it and the ASPSP for which the ticket was opened, until their successful closure.

Link: https://developer.sibsapimarket.com/live/support (for registered users only)

3.2 Developers Portal Forum

On this Developers Forum TPPs can check whenever an ASPSP has a scheduled downtime or discuss any relevant feature of or use for any API the platform supports. By registering to the notification feature, TPPs will be notified of any relevant information concerning the availability (e.g. scheduled maintenance) of the platform or of an individual ASPSP.

Link: https://developer.sibsapimarket.com/live/forum

4 Message signing

All messages sent towards SIBS API Market for PSD2 APIs on Test & Production environment shall be signed with the private key of the TPP related with the public key included in the QSeal eIDAS certificate issued by an eIDAS QTSP.

All messages shall include the “TPP-Certificate”, “Digest” and “Signature” parameters.

4.1 TPP-Certificate

TPP-Certificate parameter shall contain the QSeal eIDAS certificate issued by an eIDAS QTSP.

4.2 Digest

The “Digest” Header contains a Hash of the message body in the following format:

digest-algorithm=<encoded digest output>

Where:

digest-algorithm is the identifier of the algorithm used to compute the hash of the message. Possible values are SHA-256 and SHA-512.

<encoded digest output> is the base64 encoding of the result of the hash algorithm computed over the message body.

The message body shall be in linear string format for the computation of the Hash. If needed a conversion function shall be used (e.g. JSON.stringify()).

Example:

Message body	{
	"Hello": "World"
	}

Message body string	{"Hello":"World"}
Digest Header	Digest: SHA-256= ZaTEylrPxL87NOE4yAhzD2yc4UGkxriJqTReZslznXM=

4.3 Signature

Signature parameter shall contain the message signature in the following format (according to section 2 of "Signing HTTP Messages draft-cavage-http-signatures-10", https://datatracker.ietf.org/doc/draft-cavage-http-signatures/):

keyId="<key-identifier>",algorithm="<signature-algorithm>",headers="<header1> <header2> <headerN>",signature="<message-signature>"

Where:

<key-identifier> is the serial number of the TPP-Certificate sent in the TPP-Certificate parameter.

<signature-algorithm> is the identifier of the algorithm used to sign the message. Possible values are rsa-sha256 and rsa-sha512.

<header1>…<headerN> is the list of message header parameters included in the signing-string. The following message header parameters are mandatory: Digest, TPP-Transaction-ID, TPP-Request-ID and Date. The following message header parameters are mandatory if included in the message: PSU-ID or PSU-Corporate-ID.

<message-signature> is the base64 encoding of the result of the signature algorithm computed over the signing-string, using the private key that is the pair of the public key included in the TPP-Certificate sent in the TPP-Certificate parameter.

The signing-string shall be assembled as the concatenation of the parameters names and values identified in the "headers", according to the following rules:

- All HTTP header names are in lowercase;
- All HTTP header names are immediately followed by an ASCII colon ':' (no ASCII spaces (chr(20)) in between);
- A single ASCII space (chr(20)) is added between the colon ':' and the header parameter value;
- All header parameter values are trimmed (leading and trailing ASCII spaces (chr (20)) are removed);
- If the header is not the last then append an ASCII newline '\n' (one character (chr(0A))) just after the header parameter value.

Example:

```
Signature: keyId="44ba0c31580926fcda18ddee8d1ac0a9",algorithm="rsa-
sha256",headers="Digest TPP-Transaction-ID TPP-Request-ID PSU-ID
Date",signature="H+5o5vQJ1KsSlfTG5hCehXxk63warUpkcBftyAuRsZEP2KCjraZZU0yu4IRSA6txo6FhLXB
HQND3e4VfSzvvi8pxdWIxqX8/lOo4EKiVz1jZkzyO5T1hcA+eInBKNxfb5vvk6wmfJ2FxoBJ9ba8JqH1txzjXhuP
Gj3j+Bcc9PTGHtg+U5Z2BlVUhsyf+i4oD7p/gpBPQTUPBFoVxMLpEwycub3YTHdqKpF9rCpEz76WLc30DOWkCy3w
ysnoJ6iuNvH4XH9YpeukeQLrC3PdRhwf4+XrEQNGWosKNpk2Iy/QtUpPPt2gEUwws0owKd5XDKOtavG2qZBQ/+au
OK9H41A=="
```

4.4 Example of a signed message

The following data:

Private Key (in PEM format) that is the key pair of the public key included in the QSeal Certificate included in parameter TPP-Certificate	-----BEGIN PRIVATE KEY----- MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQCfpjExqo7Ep33z Z9DzOUnPAVMX3hdpHIbQZh+fVpR9FI2HKjc96GzvuhEHLB1qjas9K0szPWsR1GCH nbFVFS4tLnbSz/TEA9tlIzmeAlJHRngbprhjAx0TLq6DCMikQoWacfde/m80p4ED Ha4moiQNr+mtekLyR4upiZ1N+fYCWJmYXNgtIlOQzhmol6gfhcfhReZtfgfJswDe 7K5Zqosa84BcfCa86SE41/Gnu2adzf4eUXDwj7YhkXZ6BsUgmM9hONqFBltnzjrU C2a1pCFI4LSc61gaRLZmXFMEoFV36f+Sc2jKO6e89OqKzhMoy8zOMjYh3CVO/vYT TUkfsncvAgMBAAECggEAC4oWqjFaymoU14AcgUgVxqmg2OvLo2RVdkC7bmfyqavP owJxJb9kCFvZmTweEDKXOv+jGtwInixMoeDLDYxPXyKpogk39ucUd5X5NyuRyOcW DWGamEWEV3ubT4tV/VF1mLk+GRao8RReZdxCVzaHBpo9eLWKmzqCpMl2nkk/9+FU VuzmoLpFmyPFK03bH/Nl5w92PYQ5ttkNOdqQzCxISOD/8BqU4KbV2Ng4gi82Xvvh VDL6HffYpujspi7DejbSBZ+bOsBe1ZQEhtzNNJ/Q4Uv8XIuLm2h2cm/E+bNfEr7z A8SkG7PpB+SRjSteaSE+SoeJ5RT7t6kgWXS1C/jL6QKBgQDLtUGaQEqKZTspvbTu CbA1Vp67LEoy5T4Mga2vob1KVvGmQEfuhiI1cj5/cgPGZ1yojYHDpnwzQbfFkPxi KFFgp4wnXGlRr0EeJYMZjW5G5Ms2M/Bcl3CvwaDL2sISjWt2FhxSQZQx39kM5m6N KvEN+ICBLqwMIon6em7b91DUFQKBgQDIoZf/F/lvtReYXUrl3spkviLNYkBbZozW YCz2riQHChXNupMsa4nFlb2+Y1goELUWOTELb0Jw1lybLewx9FFdFVncRAjgHGBA P5CsoJ2/9T/vi/N2Wj0J5DiPdfEGwpwOqXwBfh1lVlkgSnNTYXlIp+ANcDHyVhQH nf/1r9IbMwKBgBP/rXsZSLghjBdi+npMFTKHWHoDtR0eCGNt78FIXa8IrhymPumZ 3Y3ls2ELrncx+pTJn623kIXvs7z/qOdyEdstV8MdfXF5hSkSgbZmficTmyetHbHZ ZES8+65HwbnUjHJBZXJl4cirs0Fi9gOB7bxzYxpLnLRsR6OBlZSeyR6pAoGARIG5 Due2yogBeItSic9bOK8b3xmPdCY+LO1GLS0lLCora8Yrft1xe8A3vAzcC0I5M09w CcGB5Fmt4Wb64cvVBH3H40hv52aJDyclVWy6sNMjc75L8bu6X+hHz+Sr2m0VMIR6 zV+s1e94G2iQnIYKDd8UyEHpLCBsUnWG8vOIQLUCgYAh7biRup4jhXmKc9TukCHI Z2h6Pe0WFaFJcR4+3o+WTEuWy8+h1pPnlkEr8FvbLhl1B3B0hJJl0TzrP6p9FLHv Y1h+Ux1J4Il7LxVVhG5wZA3ahpxhw9vQBM+2xawoGG5Zi41kpBAWERRI8Slaqnxo ZcQpdIZkwnaXQT8uVuVjcg== -----END PRIVATE KEY-----

Message Body	{
	"access": {
	"balances": [{
	"iban": "PT50000000000000000000000",
	"currency": "EUR"
	}],
	"transactions": [{
	"iban": "PT50000000000000000000000",
	"currency": "EUR"
	}],
	"accounts": [{
	"iban": "PT50000000000000000000000",
	"currency": "EUR"
	}]
	},
	"recurringIndicator": true,
	"validUntil": "2019-12-01T13:20:00",
	"frequencyPerDay": 4,
	"combinedServiceIndicator": false
	}

Digest	SHA-256=LrIQs5UqJlz0X3B+wk25SEaUEalqvRMDCbrQFFKEaRs=

Signing String	digest: SHA-256= LrIQs5UqJlzOX3B+wk25SEaUEalqvRMDCbrQFFKEaRs= tpp-transaction-id: ceae5ddb2325457bac80b43baefaf558 tpp-request-id: bcd4aad6fcc246419485a015f4cb6996 psu-id: PSU-123 date: 2019-08-19T17:44:25.918+01:00

Results in the following POST API request message headers:

```
POST https://site1.sibsapimarket.com:8444/sibs/apimarket-sb/BST/v1-0-
2/consents?tppRedirectPreferred=false&withBalance=false HTTP/1.1
Accept-Encoding: gzip,deflate
PSU-ID-Type: sdfasdf
TPP-Redirect-URI: /teste/teste
Content-Type: application/json
x-ibm-client-id: 476bd8e5-e6f9-4f83-bb38-a07f56a063f2
TPP-Request-ID: bcd4aad6fcc246419485a015f4cb6996
Digest: SHA-256=LrIQs5UqJlzOX3B+wk25SEaUEalqvRMDCbrQFFKEaRs=
Signature: keyId="44ba0c31580926fcda18ddee8d1ac0a9",algorithm="rsa-
sha256",headers="Digest TPP-Transaction-ID TPP-Request-ID PSU-ID
Date",signature="QN5IfGeEvSWX0PYMuIsxfAsjFwEuLpxR4oKClhrxL77Fiuha9rDSrNPmKjk7eS
kIVQlSQrsHwMnLuzo9uvkA9fLTPYFRVSm6seiDWeGG8Gxo2SCIyDjvGQBHXyM2k3AQ7ChQy8IKq6Uvg
xASyUKCnOJpdp11y9aLvjwbzCiIr7dbQwXMd0Jb6yptOVWwm1g7OebN9js+zkFzotgjWtSMMwpkieA4
DSNL95JYmaOQxz4K8IEhgoN96pS66pnsp0ZI+1vOi70X+LPs3EP16AC+Qh7DGlc1iyagE8EQ0FhSYwW
gYEuaFfJFv518BAVVgankwKN6Kph9sZFta+vxYMZAxA=="
PSU-ID: PSU-123
TPP-Certificate:
MIIIRTCCBi2gAwIBAgIQRLoMMVgJJvzaGN3ujRrAqTANBgkqhkiG9w0BAQsFADCBuTELMAkGA1UEBhM
CUFQxQjBABgNVBAoMOU1VTFRJQ0VSVCAtIFNlcnZpw6dvcyBkZSBDZXJ0aWZpY2HDp8OjbyBFbGVjdH
LDs25pY2EgUy5BLjEgMB4GA1UECwwXQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkxRDBCBgNVBAMMOyhDR
VJUKSBNVUxUSUNFUlQgVHJ1c3QgU2VydmljZXMgQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkgMDA0MB4X
DTE5MDQwMzE0MzYwMFoXDTE5MDcwMzE0MzYwMFowgaoxCzAJBgNVBAYTAlBUMTswOQYDVQQKDDJTSUJ
TIC0gU29jaWVkYWRlIEludGVyYmFuY8OhcmlhIGRlIFNlcnZpw6dvcywgUy5BLjEWMBQGA1UEYQwNUF
NEUFQtQlAtOTk5OTE3MDUGA1UECwwuUFNEMiBRdWFsaWZpZWQgQ2VydGlmaWNhdGUgZm9yIEVsZWN0c
m9uaWMgU2VhbDENMAsGA1UEAwwEU0lCUzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAJ+m
MTGqjsSnffNn0PM5Sc8BUxfeF2kchtBmH59WlH0UjYcqNz3obO+6EQcsHWqNqz0rSzM9axHUYIedsVU
VLi0udtLP9MQD22UjOZ4CUkdGeBumuGMDHRMuroMIyKRChZpx917+bzSngQMdriaiJA2v6a16QvJHi6
mJnU359gJYmZhc2C0iU5DOGaiXqB+Fx+FF5m1+B8mzAN7srlmqixrzgFx8JrzpITjX8ae7Zp3N/h5Rc
PCPtiGRdnoGxSCYz2E42oUGW2fOOtQLZrWkIUjgtJzrWBpEtmZcUwSgVXfp/5JzaMo7p7zO6orOEyjL
zM4yNiHcJU7+9hNNSR+ydy8CAwEAAaOCA1QwggNQMAwGA1UdEwEB/wQCMAAwHwYDVR0jBBgwFoAUpOp
A8d9T3pjwISxUIW3m7otTUtswgZwGCCsGAQUFBwEBBIGPMIGMMEoGCCsGAQUFBzAChj5odHRwczovL3
BraS50ZXN0ZS5tdWx0aWNlcnQuY29tL2NlcnQvTVVMVElDRVJUX0NBL1RTQ0FfMDA0LmNlcjA+BggrB
gEFBQcwAYYyaHR0cDovL29jc3AudGVzdGUubXVsdGljZXJ0LmNvbS9vY3NwLXNlcnZpY2VzL29jc3Aw
RwYDVR0uBEAwPjA8oDqgOIY2aHR0cDovL3BraS50ZXN0ZS5tdWx0aWNlcnQuY29tL2NybC9jcmxfdHM
wMDRfZGVsdGEuY3JsMGcGA1UdIARgMF4wCQYHBACL7EABATARBg8rBgEEAYHDbgEBAQEAAQ4wPgYNKw
YBBAGBw24BAQEABzAtMCsGCCsGAQUFBwIBFh9odHRwczovL3BraS50ZXN0ZS5tdWx0aWNlcnQuY29tM
IIBWgYIKwYBBQUHAQMEggFMMIIBSDAKBggrBgEFBQcLAjAIBgYEAI5GAQEwCwYGBACORgEDAgEHMBMG
BgQAjkYBBjAJBgcEAI5GAQYCMIGtBgYEAI5GAQUwgaIwTxZJaHR0cHM6Ly9wa2kudGVzdGUubXVsdGl
jZXJ0LmNvbS9wb2wvY3BzL01VTFRJQ0VSVF9QSi5DQTNfMjQuMV8wMDAxX2VuLnBkZhMCRU4wTxZJaH
R0cHM6Ly9wa2kudGVzdGUubXVsdGljZXJ0LmNvbS9wb2wvY3BzL01VTFRJQ0VSVF9QSi5DQTNfMjQuM
V8wMDAxX3B0LnBkZhMCUFQwXgYGBACBmCcCMFQwOTARBgcEAIGYJwEDDAZQU1BfQUkwEQYHBACBmCcB
BAwGUFNQX0lDMBEGBwQAgZgnAQIMBlBTUF9QSQwQQmFuayBvZiBQb3J0dWdhbAwFUFQtQlAwQQYDVR0
fBDowODA2oDSgMoYwaHR0cDovL3BraS50ZXN0ZS5tdWx0aWNlcnQuY29tL2NybC9jcmxfdHMwMDQuY3
JsMB0GA1UdDgQWBBQ1/ymvoiX0PE1JHfd9zeRBoApm1zAOBgNVHQ8BAf8EBAMCBkAwDQYJKoZIhvcNA
QELBQADggIBAI/X+DV/0zUH0CaFYjNBhbWh0Y6uNoZEk/Slw9eCnuBAGLBKeiyCdFwWHeol5l5XZHvg
7re5yf78cSuYij6IJNJRKRJlg/1lgSKAlsFSRlsodXOYo01kFJ33Ds0muGdX+ameD7/zsl2mT3pFn5j
nrv8TdZE1UtB3ce8mtWDwqXlLW9pQWhF9I6JgXUhqK9Bj9TxmlRBxAZIwAS22BFp6YjWTwValHqISg2
```

```
0XMVmRLVOIsyrAaeJm4PyfNY41jugREn0OGQJEpF6FMuFmckzZPAahRW+76GKdnl+3Vs7Xs6EtiozUr
u5NfsqKbbFH5WzRG8hFj4ZnKjlr01y7xky9vXaTSlQKqEdd27KP9iGLZw/rFqXMO9WsKQlkU6gbjzJ8
wRg7tUygysz1oeZI5vEd5+iDBgbDqW5PGd+l3tKvPVBCo9p08D+E60BsXTtgNuyvooUm5eaNtKeujxW
n629DZZ2p1gNwbo1wk076iQplju8vO3R3wwIakqHhorKLYCmtJ5y9xb49jTtAOaas/Zm1jck0PI9vxy
4QCczbEoYPLzBhJvSZbwJI2yrTtq+tP8lFuuJm2r/e07WealGmjndo42orCDz31voxgmIIwx4P2/+Zc
L1XrSZl3xjpIxmmsrPiv9IYLHgHAznR0wYMlOHpoYaDjgSB98EvfduVESSu+2gM
Date: 2019-08-19T17:44:25.918+01:00
TPP-Transaction-ID: ceae5ddb2325457bac80b43baefaf558
Content-Length: 575
Host: 172.23.133.52
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

5 Contingency procedures

SIBS API Market has been built to provide high availability and fault tolerant operations to TPPs and ASPSPs. SIBS API Market infrastructure is split between two redundant active/active sites.

TPPs may call API endpoints in any of the two sites any time. In case one site stops responding, TPPs shall route all API calls to the other site to continue providing services.

TPPs shall not notice any difference on the execution of the API calls, regardless the selected site. All resources created on API calls (e.g. payment and consent resources) are shared between the two sites. Resource identifiers (e.g. "consentId" or "paymentId") created in one site may be used to address the same resource in the other site.

The selection of the SIBS API Market site, that processes the required API operation, is performed through the {host} value in the API endpoint, according to section 2.2.

6 API flows

This section describes the order TPPs shall execute the operations of each API to get the intended service.

The execution flow of each API depends on the authentication approach requested by the ASPSP to authenticate the PSU. The possible authentication approaches are:

Redirect	The steps for the PSU authentication are not executed at the interface between the SIBS API Market gateway and the TPP, but directly between the PSU and the ASPSP. In this case, the TPP shall redirect the PSU's user agent (e.g.: web browser **or app**) to an authentication web interface of the ASPSP. The URL to this web interface is included in the ASPSP response to the initial API call.
Decoupled	The steps for the PSU authentication are not executed at the interface between the SIBS API Market gateway and the TPP, but directly between the PSU and ASPSP. In this case, the ASPSP asks the PSU to authenticate in a separate channel, e.g. by sending a push notification with transaction details to a dedicated mobile app or via any other application or device the ASPSP makes available to the PSU.
Embedded	When applying the embedded approach the authentication of the PSU is executed entirely as part of the transaction at the interface between the SIBS API Market gateway and the TPP. PSU's authentication elements are gathered by the TPP and sent to the ASPSP for verification through the interface between the SIBS API Market gateway and the TPP.

Each ASPSP decides the authentication approach it supports and selects on each transaction. Please check section 2.

The selection of the authentication approach is decided by the ASPSP during the initial POST API operation. In the response to the POST operation (and the following PUT operations, if requested), the ASPSP sends requests towards the TPP, that depend on the selected authentication approach, using the "_links" parameter according to the API Steering Process by Hyperlinks defined in [BG-IG].

When the ASPSP stops sending requests in the "_links" parameter of POST and PUT operations responses, the TPP shall issue the GET status API operation, to get information on the success of the API execution.

All PSD2 API requests must be addressed to one of the ASPSP available on SIBS API Market. The list of ASPSPs can be checked in section 2. The TPP selects the recipient ASPSP of each API operation call in the path parameters aspsp-cde of the operation endpoint. The aspsp-cde assigned to each ASPSP can be retrieved using the List of banks API, available in the Information APIs product. The List of banks API provides information about the ASPSPs that may be addressed on SIBS API Market. Besides the aspsp-cde this API provides information the TPPs may present to the PSU (e.g. logotype) during selection of the ASPSPs by the PSUs, and the list of APIs provided by each ASPSP. Based on the list of provided APIs TPPs may build a dynamic list of logotypes to present to the PSU whenever the selection of ASPSPs by the PSU is needed, including in the list only the ASPSPs that provide the required service (e.g. during the execution of a payment initiation using the instant-sepa-credit-transfers payment product, TPPs may filter the list of ASPSPs presented to the PSU to select his/her ASPSP, to include only the ASPSPs that show the payments/instant-sepa-credit-transfers API in the list of APIs provided by the List of banks API).

For simplicity the protocol (https://) and the {path} are omitted in the API endpoints used in the flows below.

6.1 Payment initiation

This API initiates a transfer of funds from a PSU's payment account, held by the PSU in one of the ASPSPs available in SIBS API Market, to a beneficiary's account. The list of available ASPSPs can be checked on section 2. A payment initiation request addressed to an ASPSP with an ordering IBAN pattern that does not belong to that ASPSP will be immediately rejected (e.g. 'DE89' for a Portuguese based ASPSP). Depending on the payment product, the day of the week and the time, the payment may be settled immediately, and the funds credited on the beneficiary's account, or scheduled for settlement during the next settlement routine of the ASPSP. The transaction status returned by the ASPSP provides information on the payment execution.

The payment product used for the transfer of funds (e.g. SEPA Credit Transfers, SWIFT for international credit transfers) is defined by the payment-product parameter included in the API endpoint path. The list of available payment products per ASPSP can be checked on section 2 and, preferably, through the List of banks API.

The payment initiation flow ends when the ASPSP returns a final transaction status in the GET payment status API response. The transaction status is sent in "transactionStatus" parameter.

Payment resources are deleted, and a response 404-Not found is returned:

- One month after the payment initiation reached a final status;
- 30 minutes after the ASPSP was unable to perform the PSU authentication in the redirect and decoupled approaches (e.g. redirection of the PSU's browser, or the push notification to the dedicated app, didn't work, or the PSU abandoned the authentication procedure).

While a final transaction status is not returned in the GET payment status API response, the TPP may issue the GET payments status API operation until a final status is returned. While in the redirect and embedded authentication approaches the call to the GET payment status API is performed after the PSU authentication has ended, and a final status may immediately be returned, in the decoupled authentication approach the TPP needs to poll the ASPSP in order to know when the PSU authentication has ended. To stop excessive bandwidth consumption that may put at risk the stability of the service, SIBS API Market implements throttling mechanisms. It is recommended to observe a delay of at least 5 seconds between calls to the GET payment status API during the status polling.

Once a payment initiation reaches a final status, no more changes will occur on the payment initiation status until the payment resource is deleted (e.g. if the returned status is "ACSC", all following GET status will return "ACSC", even if the payment is sent to settlement by the ASPSP during the 30 minutes before the resource is deleted.

The possible values for the "transactionStatus"/code parameter, and the definition of the final status, are included in the following table:

Table 4 – Payment Initiation - possible values for the “transactionStatus”/code parameter

“transactionStatus”	Status	Definition
RCVD	Received	The ASPSP received the payment initiation request. This is an initial status.
PDNG	Pending	The ASPSP is performing the PSU authentication. Further checks and status update will be performed. This is an intermediate status.
PATC	Partially Accepted Technical Correct	The ASPSP has successfully authenticated the PSU. The payment initiation has been accepted but the funds transfer policy for the account requires the authorization of other accountholders (typically on corporate accounts). The remaining authorizations will be gathered by the ASPSP on his client direct interfaces (e.g. home banking). Once all the required authorisations are granted the payment initiation status evolves to a final status. This is an intermediate status.
RJCT	Rejected	The PSU refused the payment or failed the authentication, or an error occurred. This is a final status.
CANC	Cancelled	The payment has been cancelled by the TPP or the PSU through the TPP. This is a final status.
ACTC	Accepted Technical Validation	The ASPSP has successfully authenticated the PSU. Availability of funds on the account has not been checked yet. The payment is scheduled to be booked on PSU’s account and sent for settlement on the next settlement routine of the ASPSP. This is a final status for the recurring/periodic and future dated payments
ACSP	Accepted Settlement in Process	The payment has been booked on the PSU’s account. All preceding checks such as technical validation and customer profile were successful and therefore the payment initiation has been accepted for execution. This is a final status.
ACSC	Accepted Settlement Completed	The payment has been booked and settled on the PSU’s account. Depending on the creditor agent, funds may have been credited on the beneficiary’s (creditor’s) account. This status will be seldom seen and may only occur if the ASPSP both sends the payment for settlement and receives a response from the settlement system just after the successful PSU authentication, and before the reception of the Get payment status API call from the TPP. This is a final status.
ACCC	Accepted Settlement Completed	Settlement on the creditor's account has been completed. This is a final status.

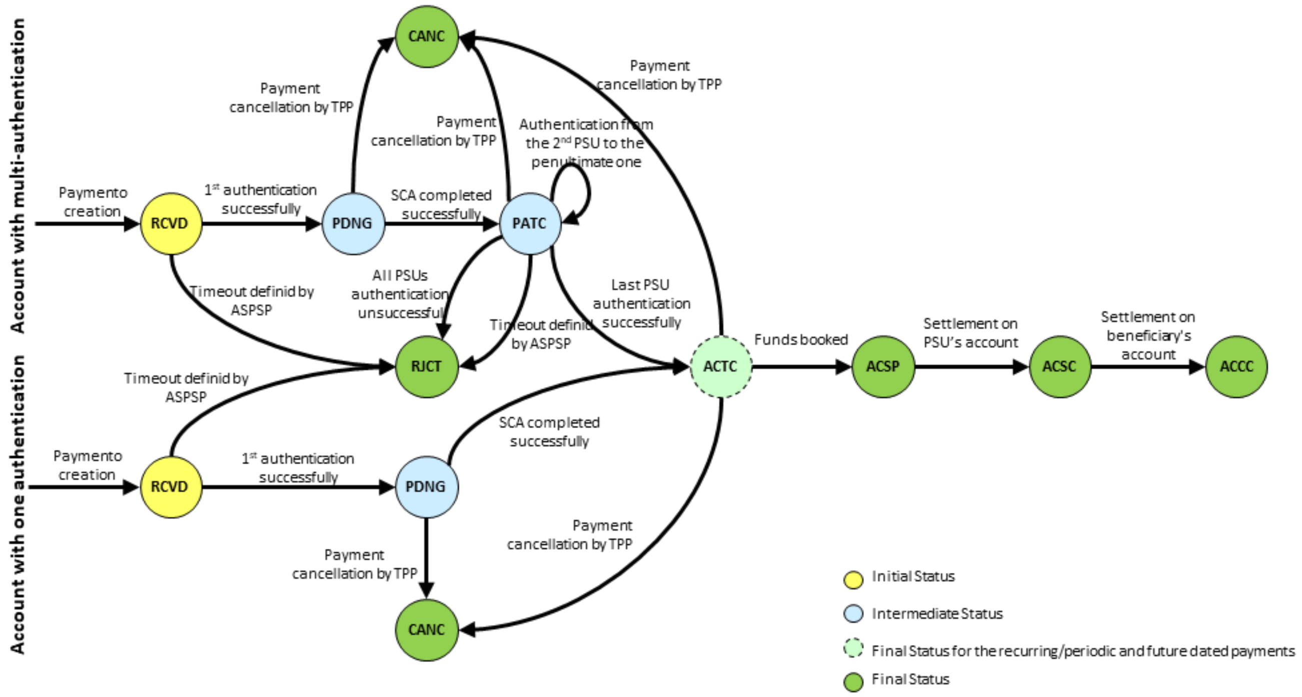

Account with multi-authentication
Account with one authentication
CANC
Payment cancellation by TPP
Payment cancellation by TPP
Payment cancellation by TPP
Authentication from the 2nd PSU to the penultimate one
Paymento creation
RCVD
1st authentication successfully
PDNG
SCA completed successfully
PATC
Timeout definid by ASPSP
All PSUs authentication unsuccessful
Timeout definid by ASPSP
Last PSU authentication successfully
RJCT
ACTC
Funds booked
ACSP
Settlement on PSU's account
ACSC
Settlement on beneficiary's account
ACCC
Timeout definid by ASPSP
Paymento creation
RCVD
1st authentication successfully
PDNG
SCA completed successfully
Payment cancellation by TPP
Payment cancellation by TPP
CANC
Initial Status
Intermediate Status
Final Status for the recurring/periodic and future dated payments
Final Status

Figure 1 – Payment initiation status diagram (with one authentication or multi-authentication)

6.1.1 Redirect flow

This is the happy path flow to execute a payment initiation with the redirect authentication approach.

The ASPSP informs the TPP that a redirect flow shall be performed by sending in the response to the POST operation the "_links" parameter containing the sub-parameter "redirect" containing a URL to the ASPSP authentication web/app interface, where the TPP shall redirect the user agent of the PSU: "links": {"redirect": "URL_of_the_authentication_web_interface"} - Implicit creation of authorization resource.

This is the default mechanism for the majority of ASPSP. Please note that specially for authorization creation, vast majority of ASPSP only require authorization by one PSU and therefore privilege the implicit creation of the authorization resource.

Once the ASPSP finishes the PSU authentication, redirects the user agent of the PSU back to a payment completion web interface of the TPP. The URL of this TPP's web interface is provided to the ASPSP in the "TPP-Redirect-URI" parameter on the initial POST payments operation. The TPP shall include in the path or query parameters of this URL, elements that allow his payment completion web interface to identify the transaction, upon redirection of the user agent of the PSU. The URL shall not include any sensitive information. The transaction identification elements should be non-reusable, randomly generated and big enough to render virtually impossible guessing a valid value.

In case of an Explicit creation of authorization resource, there is no use of an authorization ID, so the ASPSP will send the link 'start authorisation' to TPP in order to proceed. The authorization link will just appear in a second phase of the flow.

Explicit creation of the authorization resource may not be offered by some ASPSP, specially at an early stage of the implementation of release 4.

Once the user agent of the PSU reaches the payment completion web interface of the TPP, the TPP may issue the GET payment status API operation to get information about the result of the PSU authentication, and completion of the payment initiation request, via the "transactionStatus" parameter.

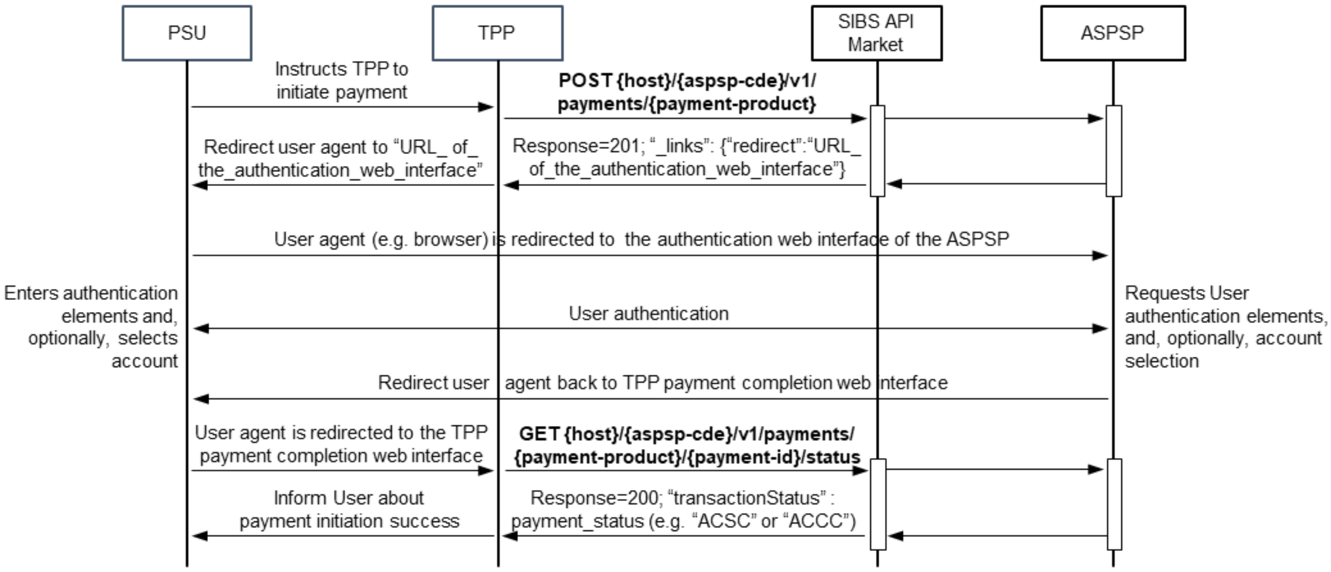

PSU
TPP
SIBS API Market
ASPSP
Instructs TPP to initiate payment
POST {host}/{aspsp-cde}/v1/ payments/{payment-product}
Redirect user agent to "URL_ of_ the_authentication_web_interface"
Response=201; "_links": {"redirect":"URL_ of_the_authentication_web_interface"}
User agent (e.g. browser) is redirected to the authentication web interface of the ASPSP
Enters authentication elements and, optionally, selects account
User authentication
Requests User authentication elements, and, optionally, account selection
Redirect user agent back to TPP payment completion web interface
User agent is redirected to the TPP payment completion web interface
GET {host}/{aspsp-cde}/v1/payments/ {payment-product}/{payment-id}/status
Inform User about payment initiation success
Response=200; "transactionStatus" : payment_status (e.g. "ACSC" or "ACCC")

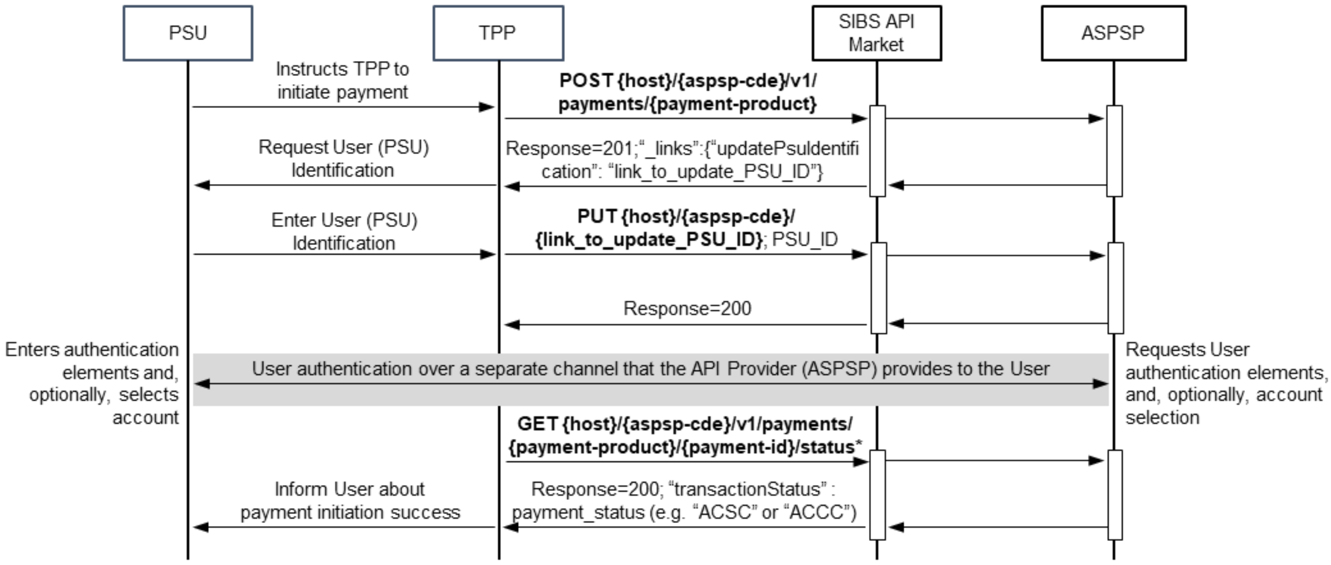
Figure 2 - Payment initiation flow for the redirect authentication approach

6.1.2 Decoupled flow

This is the happy path flow to execute a payment initiation with the decoupled authentication approach.

In the decoupled flow, the ASPSP needs to receive the User Identification used by the PSU to identify herself/himself on the ASPSP channels (e.g. home banking). For ASPSPs that have implemented only the decoupled flow, TPPs may request the User Identification to the PSU and send it on the initial POST payment initiation request. If the User Identification is not provided by the TPP in the POST operation, the ASPSP requests it by sending in the POST response the "_links" parameter containing the sub-parameter "updatePsuIdentification" containing a URL to the endpoint the TPP shall use with the PUT operation, to update the User identification. The TPP requests the PSU to enter the identifier and sends it to the ASPSP in the "PSU-ID" parameter of the PUT operation.

Whenever the parameter "psuMessage" is sent by the ASPSP in the response to the POST and PUT API operations, the TPP should present its content to the PSU, as it may provide information and guidance to the PSU (e.g. to inform the PSU about the User Identification he/she needs to enter for the "updatePsuIdentification" request, or to provide guidance to the PSU on the usage of the dedicated app for authentication).


PSU
TPP
SIBS API Market
ASPSP
Instructs TPP to initiate payment
POST {host}/{aspsp-cde}/v1/ payments/{payment-product}
Request User (PSU) Identification
Response=201;"_links":{"updatePsuIdentification": "link_to_update_PSU_ID"}
Enter User (PSU) Identification
PUT {host}/{aspsp-cde}/ {link_to_update_PSU_ID}; PSU_ID
Response=200
Enters authentication elements and, optionally, selects account
User authentication over a separate channel that the API Provider (ASPSP) provides to the User
Requests User authentication elements, and, optionally, account selection
GET {host}/{aspsp-cde}/v1/payments/ {payment-product}/{payment-id}/status*
Inform User about payment initiation success
Response=200; "transactionStatus" : payment_status (e.g. "ACSC" or "ACCC")

* Depending on the time taken by the PSU to perform the authentication, repetition of the GET payment status API call may be needed before a final status is returned. It is recommended to observe a delay of at least 5 seconds between calls to the GET payment status API.

Figure 3 - Payment initiation flow for the decoupled authentication approach

6.1.3 Embedded flow

This is the happy path flow to execute a payment initiation with the embedded authentication approach.

In the embedded flow, all the authentication elements needed by the ASPSP to authenticate the PSU are exchanged through the APIs in a chain of successive update requests sent by the ASPSP to the TPP, until the API Provider (ASPSP) receives all the elements. The ASPSP requests the authentication elements by sending in the POST and PUT responses the "_links" parameter containing sub-parameters that inform the TPP what element is required and the URL of the endpoint the TPP shall use with the PUT operation to update the authentication data in the ASPSP (e.g. "_links":{"updatePsuAuthentication":"aspsp-cde/v1/payments/payment-product/ payment-id"). Besides the "_links" parameter the ASPSP may send additional parameters to provide information to the TPP about the data that is to be requested to the PSU. Please read the [BG-IG] to check the complete list of values for the "_links" parameter and additional parameters (e.g. "chosenScaMethod" and "challengeData").

The authentication data required to authenticate the PSU depends on the ASPSP.

Whenever the parameter "psuMessage" is sent by the ASPSP in the response to the POST and PUT API operations, the TPP should present its content to the PSU, as it may provide information and guidance to the PSU (e.g. to inform the PSU about the User Identification he/she needs to enter for the "updatePsuIdentification" request).

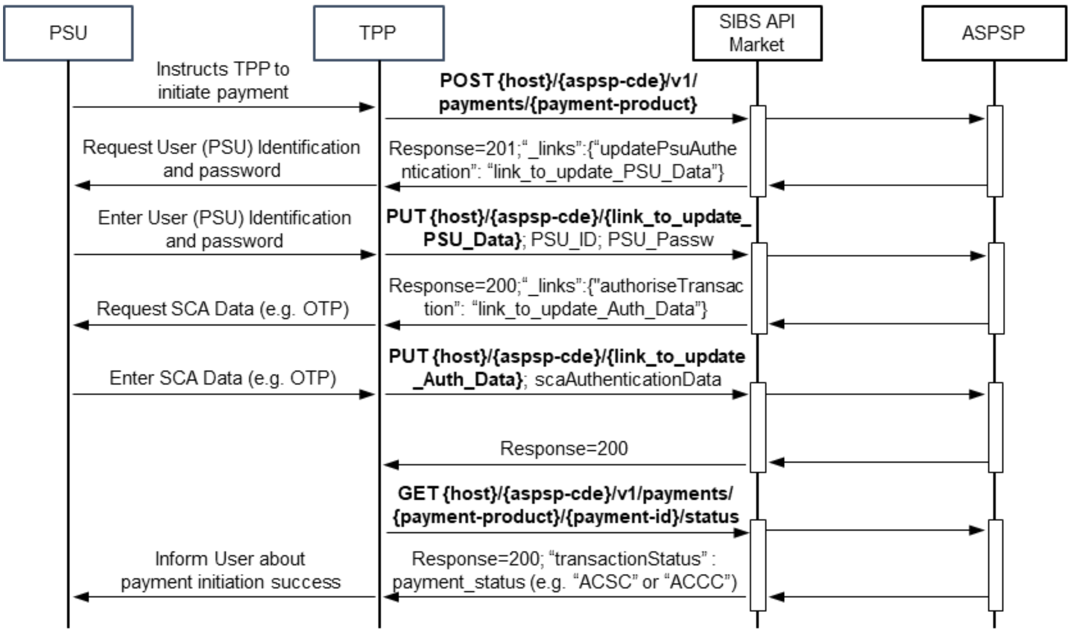

PSU
TPP
SIBS API Market
ASPSP
Instructs TPP to initiate payment
POST {host}/{aspsp-cde}/v1/ payments/{payment-product}
Request User (PSU) Identification and password
Response=201;"_links":{"updatePsuAuthentication": "link_to_update_PSU_Data"}
Enter User (PSU) Identification and password
PUT {host}/{aspsp-cde}/{link_to_update_PSU_Data}; PSU_ID; PSU_Passw
Request SCA Data (e.g. OTP)
Response=200;"_links":{"authoriseTransaction": "link_to_update_Auth_Data"}
Enter SCA Data (e.g. OTP)
PUT {host}/{aspsp-cde}/{link_to_update _Auth_Data}; scaAuthenticationData
Response=200
GET {host}/{aspsp-cde}/v1/payments/ {payment-product}/{payment-id}/status
Inform User about payment initiation success
Response=200; "transactionStatus" : payment_status (e.g. "ACSC" or "ACCC")

Figure 4 - Payment initiation flow for the embedded authentication approach

6.2 Consent, account and card account information

This APIs allows a TPP to access account or card identification details (e.g. "IBAN" or "PAN"), balances and statements of one or more accounts held by the PSU in one of the ASPSPs available in SIBS API Market. The list of available ASPSPs can be checked on section 2.

The flow shown in this section shall be repeated for each ASPSP where the PSU holds a payment account to be accessed by the TPP.

The Consent, Account and Card Account Information APIs shall be used together to receive account/card identification details (e.g.: "IBAN" or "PAN"), balances and statements.

The Consent API allows a PSU to give consent to the TPP to have access to, or to the ASPSP to provide, information of his/her payment accounts/cards. On successful completion of the Consent API the TPP will be in the possession of a Consent Identification that is used in the Account Information/card account API to get account identification details (e.g.: "IBAN" or "PAN"), balances and statements.

When requesting these statements, the TPP should notice that each transaction can have a Transaction ID. The management of the Transaction ID in the list of transactions is at the discretion of each ASPSP. Some ASPSPs do not have available a unique transaction ID for each transaction in their systems.

The Transaction ID in the first transaction of the GET transactions response is required for pagination purposes only, and it is unique only inside each pagination session (sequence of concatenated transactions messages).

Being so, TPPs should note that some ASPSPs send a Transaction ID for each transaction, but they may assign a different Transaction ID for the same transaction in different transactions pagination sessions, or the same Transaction ID for a different transaction in different transactions pagination sessions.

The TPP may request a one-shot consent ('False') or a long lasting ('True') consent via the "recurringIndicator" parameter. One-shot consents are valid during the 30 minutes following the consent creation. During the 30 minutes period there is no limitation on the number of API calls to get the information under the consent scope and the history of transaction available is similar to what is available to the user when accessing directly the ASPSP channel, and may vary among ASPSP. Long lasting consents are valid for the maximum of 180 days[3], and the validity period depends on the API provider (see next paragraph).

For the sake of compliance with Article 10 of the [RTS] for the SCA exemption on the account information, long lasting consents are created with an expiration date. One month after the expiration date the consent resources are deleted. Any attempt to access a deleted consent resource returns the http response 404 – Not Found. The TPP may include a Valid Until date in the POST consent API request, but the expiration date for the consent may be anticipated by the ASPSP according to the SCA exemption on account information policy of the ASPSP, common to all the channels provided by the ASPSP to the PSU (e.g. home banking).

The Consent API provides the parameter "access" for the definition of the consent scope. Depending on this parameter and PSU decision, the consent scope may include a list of accounts, cards, balances and transactions.

The following use cases provide some examples on the usage of the "access" parameter:

Table 5 – Use cases on the usage of the "access" parameter

ID	Use Case Description	Content of "access" parameter
#1	TPP needs to get the list of all payment accounts and cards held by the PSU at the ASPSP. The PSU gives consent to the TPP.	{"availableAccounts":"all-accounts"}
#2	TPP needs to get the list of all payment accounts and cards held by the PSU at the ASPSP, as well as the Account Holder Names of those accounts/cards. The PSU gives consent to the TPP.	{"availableAccounts":"all-accounts-with-ownerName"}
#3	TPP needs to get the list of all payment accounts and cards held by the PSU at the ASPSP, as well as the balances and transactions of those accounts/cards. The PSU gives consent to the TPP.	{"allPsd2":"all-accounts"}
#4	TPP needs to get the list of all payment accounts and cards held by the PSU at the ASPSP, as well as the balances, transactions and Account Holder Names of those accounts/cards. The PSU gives consent to the TPP.	{"allPsd2":"all-accounts-with-ownerName"}
#5	TPP needs to get the list of all payment accounts held by the PSU at the ASPSP, but the PSU doesn't give her/his consent to the TPP. In this case the TPP proposes the PSU to select the accounts on the ASPSP side, through the same interface provided by the API Provider for the PSU authentication.	{"accounts":[]"}

[3] This period changed from 90 days to 180 days upon entry into force of the revised article 10 of RTS on SCA.

#6	TPP needs to get the list of all payment accounts held by the PSU at the ASPSP, but the PSU doesn’t give her/his consent to the TPP.
	In this case the TPP proposes the PSU to enter the IBANs of the accounts she/he gives consent for the TPP.
	The PSU enters IBAN_1 and IBAN_2.
		{"accounts": [
		{ "iban": "IBAN_1" },
		{ "iban": " IBAN_2" }
		] }

#7	TPP needs to get the list of all payment accounts held by the PSU at the ASPSP, as well as the balances and transactions of those accounts, but the PSU doesn’t give her/his consent to the TPP.
	In this case the TPP proposes the PSU to give consent for the TPP to have access to balances and transactions and to select the accounts on the ASPSP side, through the same interface provided by the API Provider for the PSU authentication
		{
		"balances": [],
		"transactions": []
		}

#8	TPP needs to get the list of all payment accounts held by the PSU at the ASPSP, as well as the balances and transactions of those accounts, but the PSU doesn’t give her/his consent to the TPP. In this case the TPP gives the PSU the possibility to enter the IBANs of the accounts, and, for each IBAN, to select the type of information she/he consents the TPP to access: balances and/or transactions. The PSU enters: • IBAN_1 and only consents balances for this IBAN; • IBAN_2 and only consents transactions for this IBAN; • IBAN_3 and consents balances and transactions for this IBAN.	{ "balances": [{ "iban": "IBAN_1" }, { "iban": "IBAN_3" }], "transactions": [{ "iban": "IBAN_2" }, { "iban": "IBAN_3" }] }
#9	TPP needs to get the list of all payment cards accounts held by the PSU at the ASPSP, but the PSU doesn’t give her/his consent to the TPP. In this case the TPP proposes the PSU to select the cards on the ASPSP side, through the same interface provided by the API Provider for the PSU authentication.	{"cards-accounts":[]"}

#10	TPP needs to get the list of all payment cards accounts held by the PSU at the ASPSP, but the PSU doesn’t give her/his consent to the TPP.
	In this case the TPP proposes the PSU to enter the PANs of the cards she/he gives consent for the TPP.
	The PSU enters PAN_1 and PAN_2.
		{"cards-accounts": [
		{ "pan": "PAN_1" },
		{ "pan": "PAN_2" }
		] }

#11	TPP needs to get the list of all payment cards accounts held by the PSU at the ASPSP, as well as the balances and transactions of those accounts, but the PSU doesn’t give her/his consent to the TPP.
	In this case the TPP proposes the PSU to give consent for the TPP to have access to balances and transactions and to select the cards on the ASPSP side, through the same interface provided by the API Provider for the PSU authentication
		{
		"balances": [],
		"transactions": []
		}

#12	TPP needs to get the list of all payment cards accounts held by the PSU at the ASPSP, as well as the balances and transactions of those cards, but the PSU doesn’t give her/his consent to the TPP. In this case the TPP gives the PSU the possibility to enter the PANs of the cards, and, for each PAN, to select the type of information she/he consents the TPP to access: balances and/or transactions. The PSU enters: • PAN_1 and only consents balances for this IBAN; • PAN_2 and only consents transactions for this PAN; • PAN_3 and consents balances and transactions for this PAN.	{ "balances": [{ "pan": "PAN_1" }, { "pan": "PAN_3" }], "transactions": [{ "pan": "PAN_2" }, { "pan": "PAN_3" }] }

#13	TPP needs to get the list of all payment cards accounts held by the PSU at the ASPSP, as well as the balances and transactions of those cards, but the PSU doesn't give her/his consent to the TPP. In this case the TPP gives the PSU the possibility to enter the PANs of the cards, and, for each PAN, to select the type of information she/he consents the TPP to access: balances and/or transactions. The PSU enters: • IBAN_1 and only consents balances for this IBAN. • IBAN_3 and only consents transactions for this IBAN. • PAN_3 and consents balances and transactions for this PAN.	{ "balances": [{ "iban": "IBAN_1" }, { "pan": "PAN_3" }], "transactions": [{ "pan": "PAN_3" }, { "iban": "IBAN_3" }] }

After the creation of a consent, it's not possible to modify the accounts/cards under the scope of the consent, nor the associated information (e.g. account name). If the TPP wants to modify the list of accounts or cards accounts (e.g. to add a new account/card) a new consent shall be created with the new list of accounts/card accounts. The TPP may issue a POST consent request using the IBANs/PANs that has received for the previous consent with the required changes, or just start from scratch.

The successful creation of a long lasting consent revokes any previous long lasting consent that may exist for the same PSU, TPP and ASPSP. The unique key for verification of the uniqueness of the consent is composed by the following parameters:

- aspsp-cde;
- TPP Registration Number (retrieved from the TPP eIDAS Certificate);
- PSU-ID;
- PSU-ID-Type.

Or, for corporate accounts:

- aspsp-cde;
- TPP Registration Number (retrieved from the TPP eIDAS Certificate) ;
- PSU-Corporate-ID;
- PSU-Corporate-ID-Type.

The TPP may agree with the PSU on the number of daily accesses, without PSU involvement, for retrieving account information under the consent scope, and send this number on the POST consent request in parameter "frequencyPerDay". The ASPSP limits this number to 4 according to the [RTS]. Once the number of accesses exceeds the value sent in "frequencyPerDay" parameter (or 4 if "frequencyPerDay" is above 4) access to account information is denied (HTTP status 429 – too many requests). This limit is defined per account under the consent scope, and per data set (balances, transactions). This limit doesn't apply if the PSU is participating in real time in the information request (this is notified in the “psuInvolved” parameter).

The consent flow ends when the ASPSP returns a final transaction status in the GET consent status API response. The transaction status is sent in “transactionStatus” parameter.

Consent resources are deleted, and a response 404-Not found is returned:

- One month after the expiration date of the consent has been reached;
- 30 minutes after the response to the POST consent request, if the ASPSP was unable to perform the PSU authentication in the redirect and decoupled approaches (e.g. redirection of the PSU's browser, or the push notification to the dedicated app, didn't work, or the PSU abandoned the authentication procedure).

A deprecated consent (RJCT) can be kept in the database for a month before being deleted.

The TPP may issue the GET consent status API operation until a final status is returned. While in the redirect and embedded authentication approaches the call to the GET consent status API is performed after the PSU authentication has ended, and a final status may immediately be returned, in the decoupled authentication approach the TPP needs to poll the ASPSP in order to know when the PSU authentication has ended. To stop excessive bandwidth consumption that may put at risk the stability of the service, SIBS API Market implements throttling mechanisms. It is recommended to observe a delay of at least 5 seconds between calls to the GET consent status API during the status polling.

Once a consent reaches a final status, no more changes will occur on the consent status until the consent resource is expired/revoked/deleted (EXPD/RVKD/TERM).

The possible values for the "transactionStatus"/code parameter, and the definition of the final status, are included in the following table:

Table 6 - Consent, account and card account information - possible values for the "transactionStatus"/code parameter

"transactionStatus"	Status	Definition
RCVD	Received	The ASPSP received the consent request. This is an initial status.
RJCT	Rejected	The PSU refused the consent or failed the authentication, or an error occurred. This is a final status.
PATC	PartiallyAuthorised	The ASPSP has successfully authenticated the PSU. The consent has been accepted but the access to account information policy for the account requires the authorisation of other accountholders (typically on corporate accounts). The remaining authorisations will be gathered by the ASPSP on his client direct interfaces (e.g. home banking). Once all the required authorisations are granted the consent status evolves to a final status. This is an intermediate status.
VALD	Valid	The ASPSP has successfully authenticated the PSU. This is a final status.
RVKD	RevokedByPSU	The consent was revoked at a request of the PSU and through the ASPSP interface he was duly informed. This is a final status.

“transactionStatus”	Status	Definition
EXPD	Expired	Consent has expired. This is a final status.
TERM	TerminatedByTPP	The TPP terminated the consent by selecting the DELETE option to cancel the consent. This is a final status.

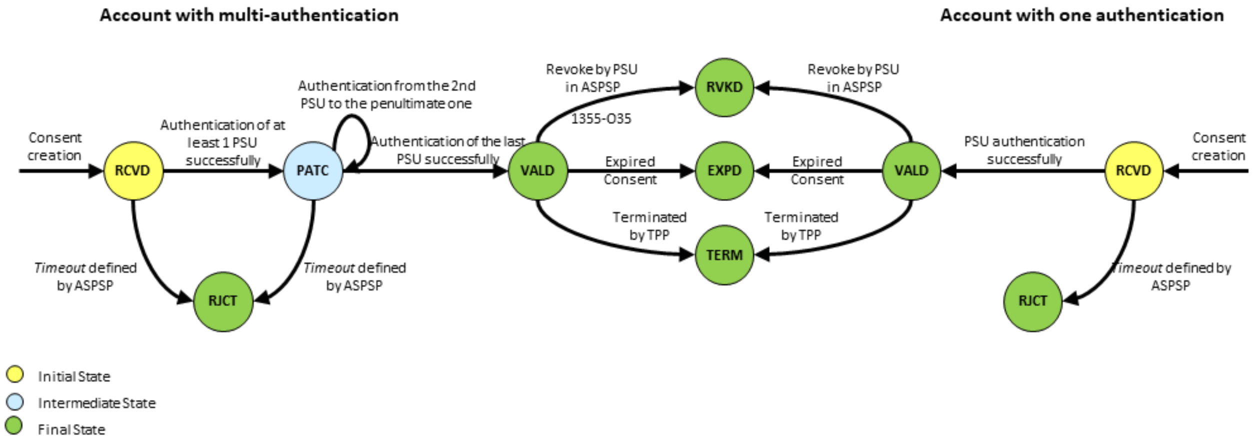


Figure 5 - Consent status diagram (with one authentication or multi-authentication)

Once the GET consent status API operation returns the “VALD” transaction status, the TPP may start requesting account information.

Account information requests will be denied if the TPP doesn’t perform the GET consent status and receives the “VALD” transaction status.

Before retrieving balances and transactions of each account/card under the consent scope, the TPP must get the links to use on the GET account details, balances and transactions operations for each account.

The GET accounts/card-accounts API operation returns the list of accounts under the consent scope, and, for each account, a.o., the following parameters:

“id”	Account identifier. This account identifiers is purely technical and randomly generated. May be used in the “account-id” path parameter of GET accounts/{account-id}, accounts/{account-id}/balances and accounts/{account-id}/transactions
"viewBalances" (in "_links" parameter)	URL to use with the GET operation to read account balances
"viewTransactions" (in "_links" parameter)	URL to use with the GET operation to read account transactions

The GET accounts/{account-id} operation, used to read account details, is denied unless the consent resource has been created with "allPsd2":"all-accounts", "accounts": [“iban”: "IBAN"] or "card-accounts": [“pan”: "PAN"] in the “access” parameter.

The GET accounts API operation returns all the accounts and the GET card-accounts returns all the cards, that are under the consent’s scope, as long as the account/card-account has been selected for at least one type of access - details, balances or transactions,

If the response to GET transactions operation is unable to include all transactions booked and/or pending between "dateFrom" and "dateTo" parameters, SIBS API Market shall include an URL for the TPP to retrieve more transactions (sub-parameter "next" within parameter "_links"). The TPP shall perform the GET operation on the received URL, without performing any modification to the URL (e.g. shall not add query parameters), until the “next” sub-parameter is no longer included in the response.

According to [RTS], ASPSPs cannot exempt from SCA the access to transactions older than 90 days. The GET transactions API operation is denied if the starting date, included in parameter “dateFrom”, is more than 90 days before the current date, unless the consent has been created, and SCA performed, within the last 30 minutes. If the consent is already older than 30 minutes, then TPPs shall create a new consent, and issue the GET transactions operation during the following 30 minutes, to retrieve transactions older than 90 days. The consent scope (list of accounts and information to retrieve) of previous consent may be reused for the new consent, to avoid requesting the PSU to define the consent scope again.

6.2.1 Redirect flow

This is the happy path flow to create a consent with the redirect authentication approach.

The ASPSP informs the TPP that a redirect flow shall be performed by sending in the response to the POST operation the “_links” parameter containing the sub-parameter “redirect” containing a URL to the ASPSP authentication web/app interface, where the TPP shall redirect the user agent of the PSU (e.g.: “_links”: {“redirect”: “URL_of_the _authentication_web_interface”}) – Implicit creation of authorization resource.

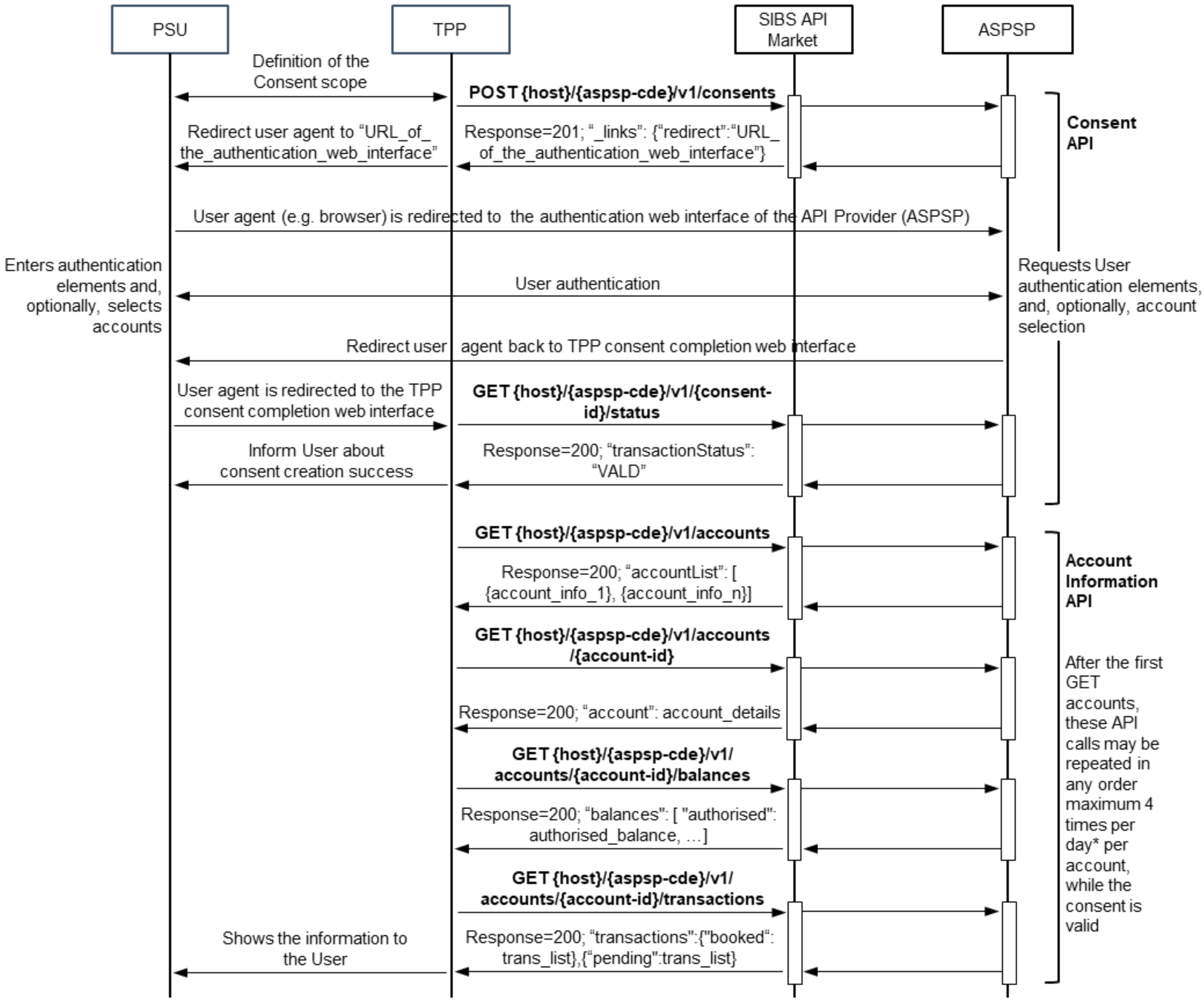
This is the default mechanism for the majority of ASPSP. Please note that specially for authorization creation, vast majority of ASPSP only require authorization by one PSU and therefore privilege the implicit creation of the authorization resource.

Once the ASPSP finishes the PSU authentication, redirects the user agent of the PSU back to a consent completion web interface of the TPP. The URL of this TPP’s web interface is provided to the ASPSP in the “TPP-Redirect-URI” parameter on the POST consents operation. The TPP shall include, in the path or query parameters of this URL, elements that allow the consent completion web interface to identify the transaction, upon redirection of the user agent of the PSU by the ASPSP. The URL shall not include any sensitive information. The transaction identification elements should be non-reusable, randomly generated, big enough to render virtually impossible guessing a valid value, and should not be valid for more than the needed time (e.g.: 30 minutes).

In case of an Explicit creation of authorization resource, there is no use of an authorization ID, so the ASPSP will send the link 'start authorisation' to TPP in order to proceed. The authorization link will just appear in a second phase of the flow.

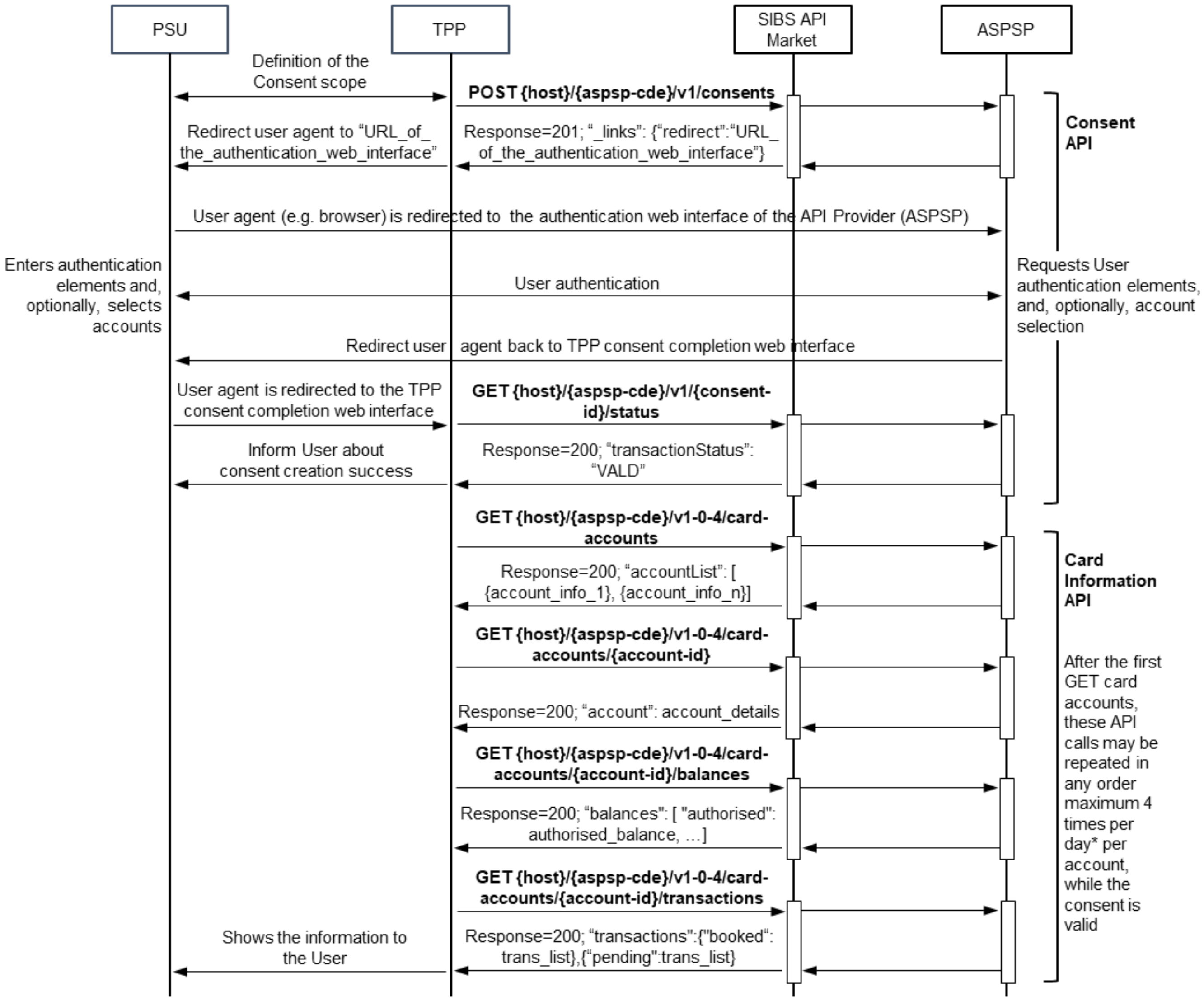
Explicit creation of the authorization resource may not be offered by some ASPSP, specially at an early stage of the implementation of release 4.

Once the user agent of the PSU reaches the consent completion web interface of the TPP, the TPP may issue the GET consent status API operation to get information about the result of the PSU authentication, and completion of the consent request, via the "transactionStatus" parameter.


PSU
TPP
SIBS API Market
ASPSP
Definition of the Consent scope
POST {host}/{aspsp-cde}/v1/consents
Consent API
Redirect user agent to "URL_of_the_authentication_web_interface"
Response=201; "_links": {"redirect":"URL_of_the_authentication_web_interface"}
User agent (e.g. browser) is redirected to the authentication web interface of the API Provider (ASPSP)
Enters authentication elements and, optionally, selects accounts
User authentication
Requests User authentication elements, and, optionally, account selection
Redirect user agent back to TPP consent completion web interface
User agent is redirected to the TPP consent completion web interface
GET {host}/{aspsp-cde}/v1/{consent-id}/status
Inform User about consent creation success
Response=200; "transactionStatus": "VALD"
GET {host}/{aspsp-cde}/v1/accounts
Account Information API
Response=200; "accountList": [{account_info_1}, {account_info_n}]
GET {host}/{aspsp-cde}/v1/accounts /{account-id}
After the first GET accounts, these API calls may be repeated in any order maximum 4 times per day* per account, while the consent is valid
Response=200; "account": account_details
GET {host}/{aspsp-cde}/v1/ accounts/{account-id}/balances
Response=200; "balances": ["authorised": authorised_balance, ...]
GET {host}/{aspsp-cde}/v1/ accounts/{account-id}/transactions
Shows the information to the User
Response=200; "transactions":{"booked": trans_list},{"pending":trans_list}

* If the PSU is not involved in the requests, in real time

Figure 6 - Consent creation and account information flow for the redirect authentication approach


PSU
TPP
SIBS API Market
ASPSP
Definition of the Consent scope
POST {host}/{aspsp-cde}/v1/consents
Consent API
Redirect user agent to "URL_of_the_authentication_web_interface"
Response=201; "_links": {"redirect":"URL_of_the_authentication_web_interface"}
User agent (e.g. browser) is redirected to the authentication web interface of the API Provider (ASPSP)
Enters authentication elements and, optionally, selects accounts
User authentication
Requests User authentication elements, and, optionally, account selection
Redirect user agent back to TPP consent completion web interface
User agent is redirected to the TPP consent completion web interface
GET {host}/{aspsp-cde}/v1/{consent-id}/status
Inform User about consent creation success
Response=200; "transactionStatus": "VALD"
GET {host}/{aspsp-cde}/v1-0-4/card-accounts
Card Information API
Response=200; "accountList": [{account_info_1}, {account_info_n}]
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}
After the first GET card accounts, these API calls may be repeated in any order maximum 4 times per day* per account, while the consent is valid
Response=200; "account": account_details
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}/balances
Response=200; "balances": ["authorised": authorised_balance, …]
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}/transactions
Shows the information to the User
Response=200; "transactions":{"booked": trans_list},{"pending":trans_list}

* If the PSU is not involved in the requests, in real time

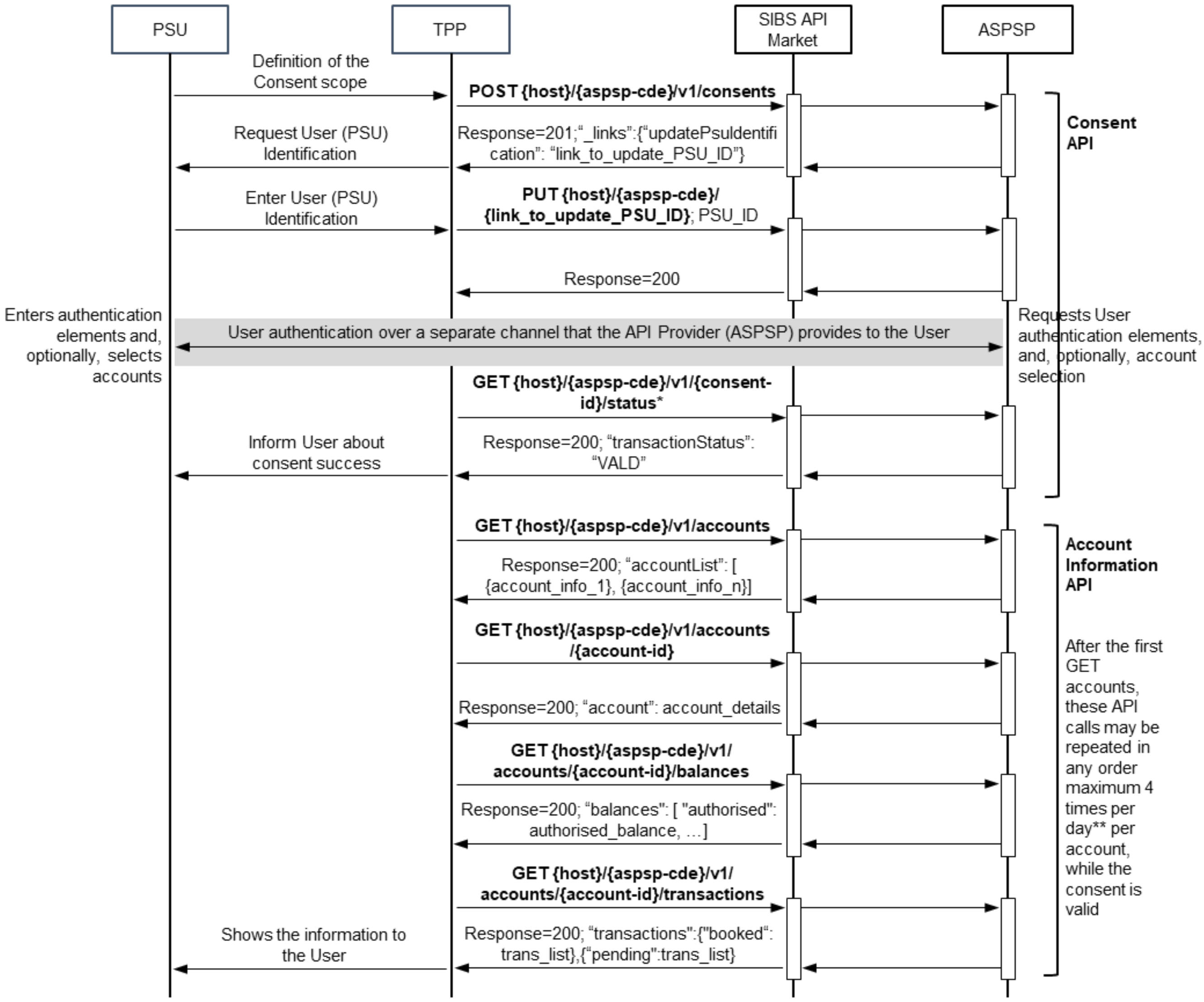
Figure 7 - Consent creation and card-account information flow for the redirect authentication approach

6.2.2 Decoupled flow

This is the happy path flow to create a consent with the decoupled authentication approach.

In the decoupled flow, the ASPSP needs to receive the User Identification used by the PSU to identify herself/himself on the ASPSP channels (e.g. home banking). For ASPSPs that have implemented only the decoupled flow, TPPs may request the User Identification to the PSU and send it on the initial POST consent request. If the User Identification is not provided by the TPP in the POST operation, the ASPSP requests it by sending in the POST response the "_links" parameter containing the sub-parameter "updatePsuIdentification" holding a URL to the endpoint the TPP shall use with the PUT operation, to update the User identification. The TPP requests the PSU to enter the identifier and sends it to the ASPSP in the "PSU-ID" parameter of the PUT operation.

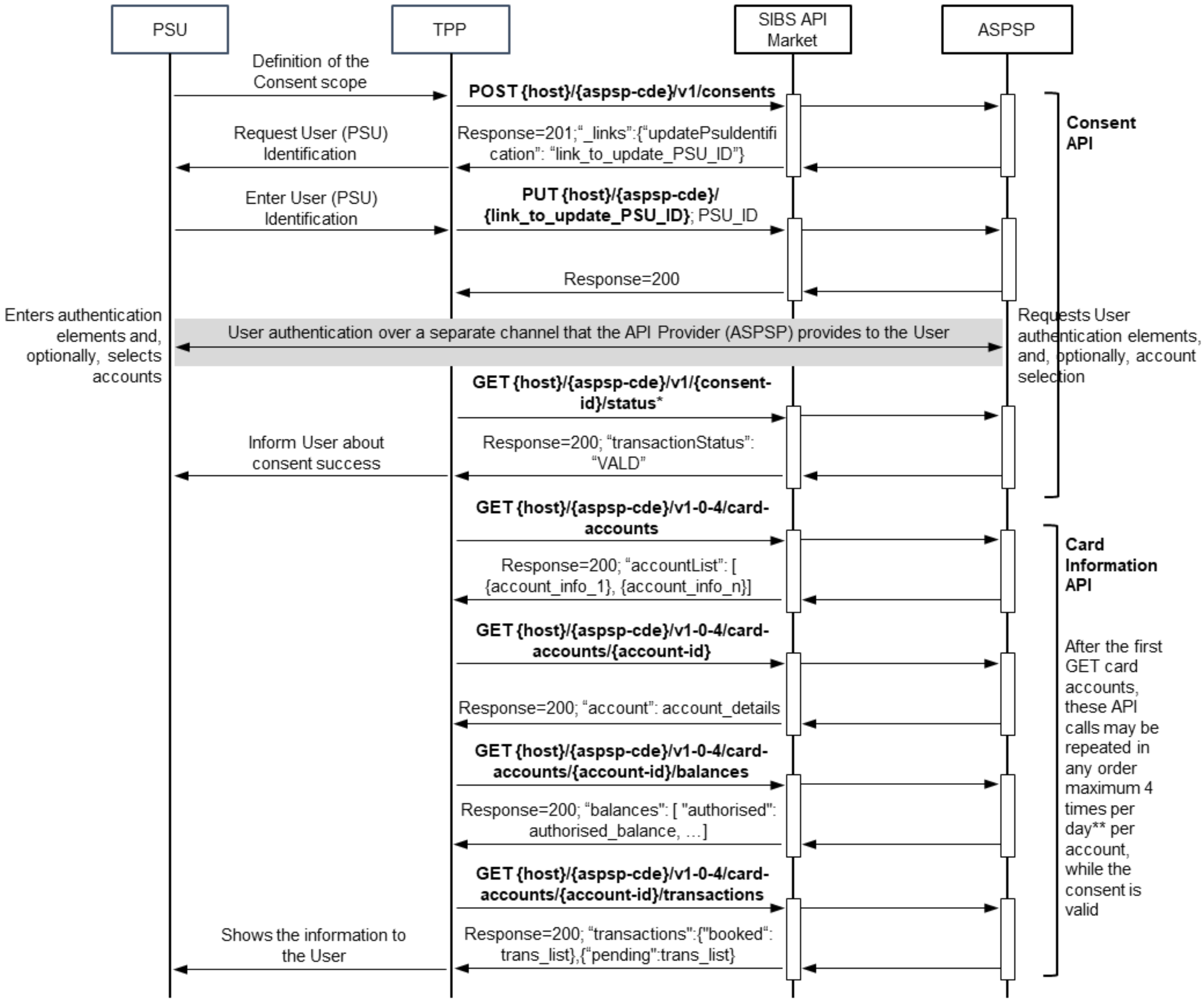
Whenever the parameter “psuMessage” is sent by the ASPSP in the response to the POST and PUT API operations, the TPP should present its content to the PSU, as it may provide information and guidance to the PSU (e.g. to inform the PSU about the User Identification he/she needs to enter for the “updatePsuIdentification” request, or to provide guidance to the PSU on the usage of the dedicated app for authentication).


PSU
TPP
SIBS API Market
ASPSP
Definition of the Consent scope
POST {host}/{aspsp-cde}/v1/consents
Consent API
Request User (PSU) Identification
Response=201;“_links”:{“updatePsuIdentification”: “link_to_update_PSU_ID”}
Enter User (PSU) Identification
PUT {host}/{aspsp-cde}/ {link_to_update_PSU_ID}; PSU_ID
Response=200
Enters authentication elements and, optionally, selects accounts
User authentication over a separate channel that the API Provider (ASPSP) provides to the User
Requests User authentication elements, and, optionally, account selection
GET {host}/{aspsp-cde}/v1/{consent-id}/status*
Inform User about consent success
Response=200; “transactionStatus”: “VALD”
GET {host}/{aspsp-cde}/v1/accounts
Account Information API
Response=200; “accountList”: [{account_info_1}, {account_info_n}]
GET {host}/{aspsp-cde}/v1/accounts /{account-id}
After the first GET accounts, these API calls may be repeated in any order maximum 4 times per day** per account, while the consent is valid
Response=200; “account”: account_details
GET {host}/{aspsp-cde}/v1/ accounts/{account-id}/balances
Response=200; “balances”: ["authorised": authorised_balance, …]
GET {host}/{aspsp-cde}/v1/ accounts/{account-id}/transactions
Shows the information to the User
Response=200; “transactions”:{"booked“: trans_list},{“pending”:trans_list}

* Depending on the time taken by the PSU to perform the authentication, repetition of the GET payment status API call may be needed before a final status is returned. It is recommended to observe a delay of at least 5 seconds between calls to the GET payment status API.

** If the PSU is not involved in the requests, in real time

Figure 8 - Consent creation and account information flow for the decoupled authentication approach


PSU
TPP
SIBS API Market
ASPSP
Definition of the Consent scope
POST {host}/{aspsp-cde}/v1/consents
Consent API
Request User (PSU) Identification
Response=201;"_links":{"updatePsuIdentification": "link_to_update_PSU_ID"}
Enter User (PSU) Identification
PUT {host}/{aspsp-cde}/ {link_to_update_PSU_ID}; PSU_ID
Response=200
Enters authentication elements and, optionally, selects accounts
User authentication over a separate channel that the API Provider (ASPSP) provides to the User
Requests User authentication elements, and, optionally, account selection
GET {host}/{aspsp-cde}/v1/{consent-id}/status*
Inform User about consent success
Response=200; "transactionStatus": "VALD"
GET {host}/{aspsp-cde}/v1-0-4/card-accounts
Card Information API
Response=200; "accountList": [{account_info_1}, {account_info_n}]
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}
After the first GET card accounts, these API calls may be repeated in any order maximum 4 times per day** per account, while the consent is valid
Response=200; "account": account_details
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}/balances
Response=200; "balances": ["authorised": authorised_balance, …]
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}/transactions
Shows the information to the User
Response=200; "transactions":{"booked": trans_list},{"pending":trans_list}

* Depending on the time taken by the PSU to perform the authentication, repetition of the GET payment status API call may be needed before a final status is returned. It is recommended to observe a delay of at least 5 seconds between calls to the GET payment status API.

** If the PSU is not involved in the requests, in real time

Figure 9 - Consent creation and card-account information flow for the decoupled authentication approach

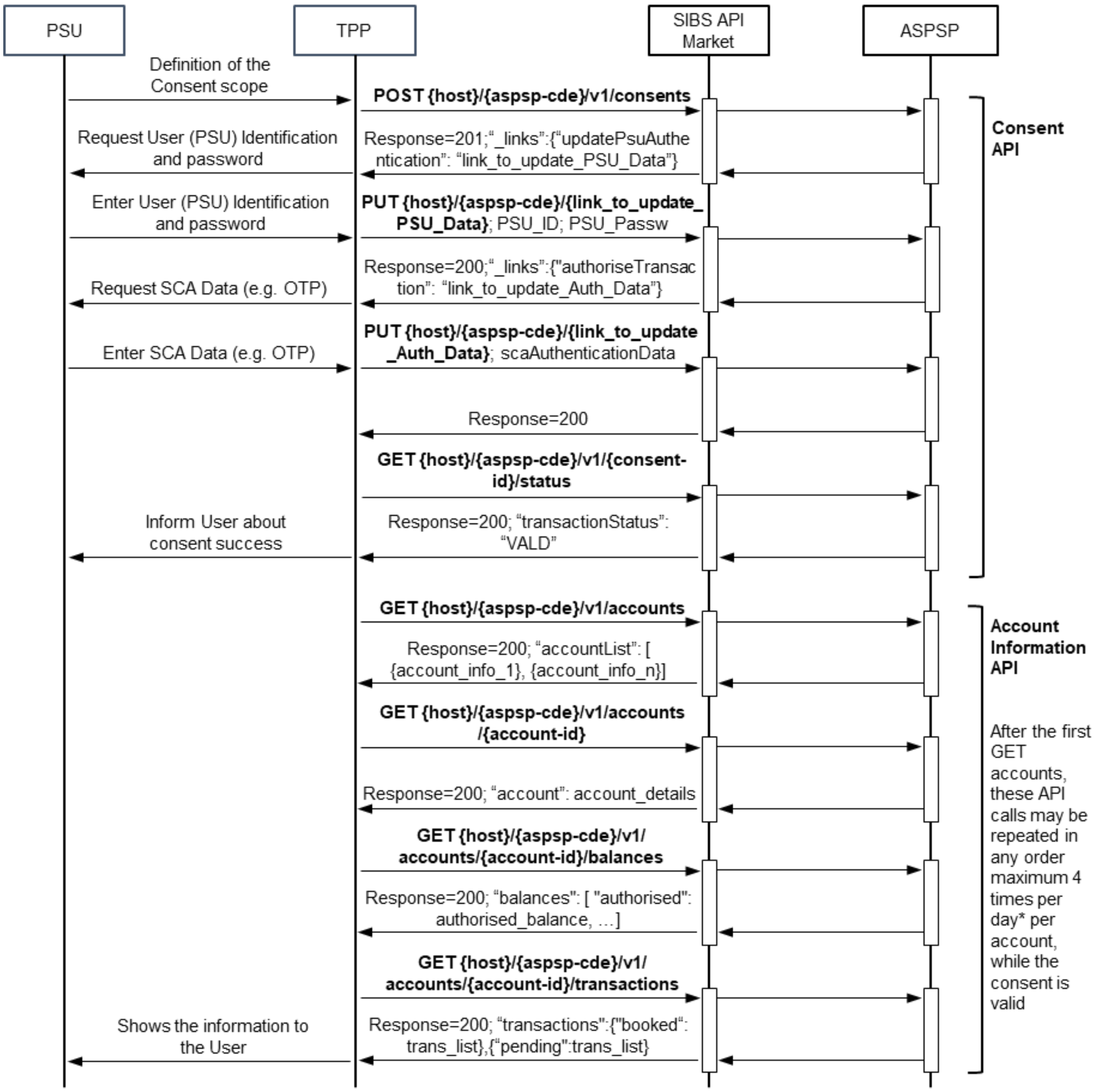
6.2.3 Embedded flow

This is the happy path flow to create a consent with the embedded authentication approach.

In the embedded flow, all the authentication elements needed by the ASPSP to authenticate the PSU are exchanged through the APIs in a chain of successive requests sent by the ASPSP to the TPP, until the API Provider (ASPSP) receives all the authentication elements. The ASPSP requests the authentication elements by sending in the POST and PUT responses the "_links" parameter containing sub-parameters that inform the TPP what element is required and the URL of the endpoint the TPP shall use with the PUT operation to update the authentication data in the ASPSP (e.g. "_links":{"updatePsuAuthentication":"aspsp-cde/v1/consents/consent-id"}). Besides the "_links" parameter the API Provider (ASPSP) may send additional parameters to provide information to the TPP about the data that is to be requested to the PSU. Please read the [BG-IG] to check the complete list of values for the "_links" parameter and additional parameters (e.g.: "chosenScaMethod" and "challengeData").

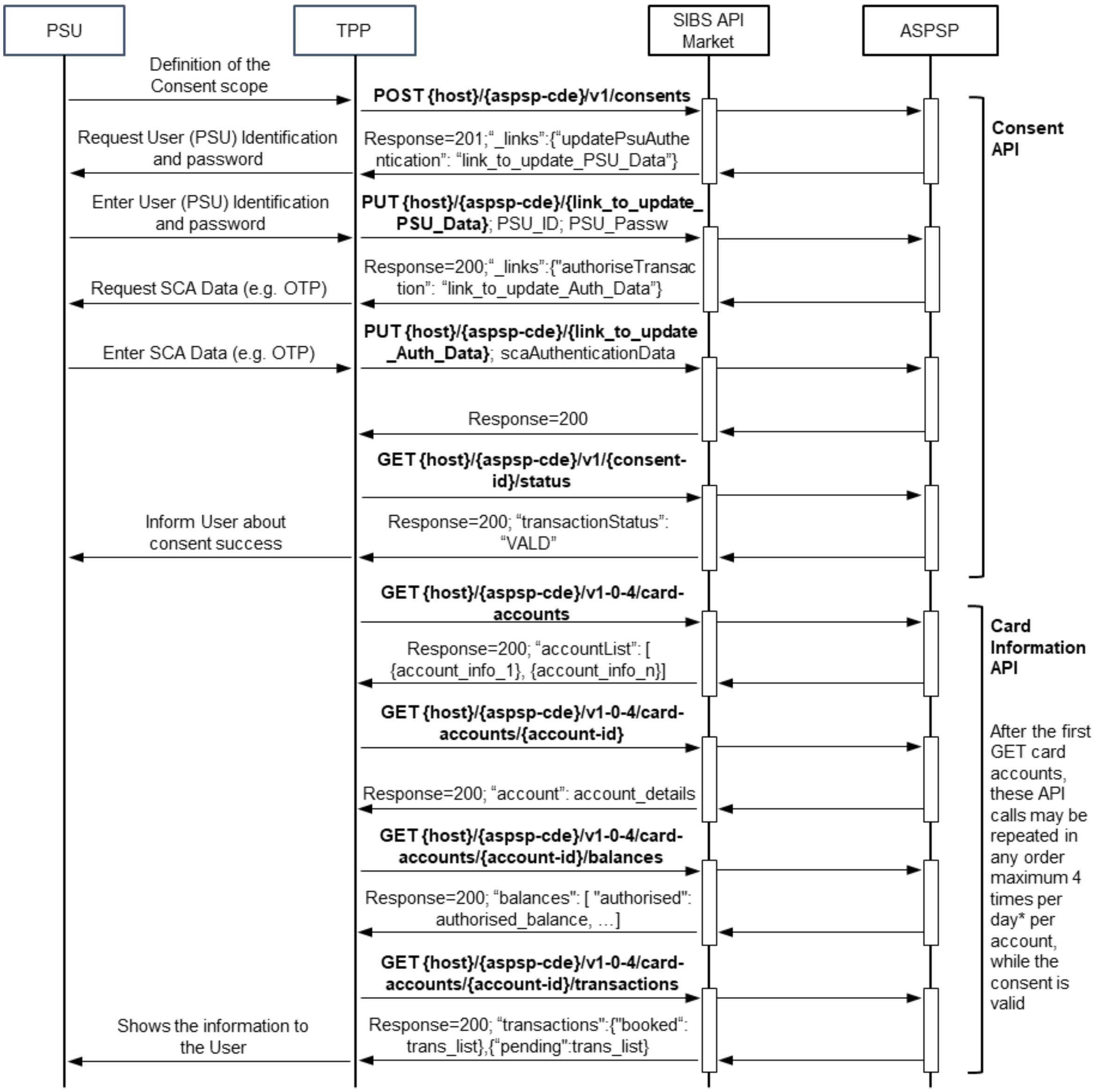
The authentication data required to authenticate the PSU depends on the ASPSP.

Whenever the parameter "psuMessage" is sent by the ASPSP in the response to the POST and PUT API operations, the TPP should present its content to the PSU, as it may provide information and guidance to the PSU (e.g.: to inform the PSU about the User Identification he/she needs to enter for the "updatePsuIdentification" request).


PSU
TPP
SIBS API Market
ASPSP
Definition of the Consent scope
POST {host}/{aspsp-cde}/v1/consents
Consent API
Request User (PSU) Identification and password
Response=201;"_links":{"updatePsuAuthentication": "link_to_update_PSU_Data"}
Enter User (PSU) Identification and password
PUT {host}/{aspsp-cde}/{link_to_update_PSU_Data}; PSU_ID; PSU_Passw
Request SCA Data (e.g. OTP)
Response=200;"_links":{"authoriseTransaction": "link_to_update_Auth_Data"}
Enter SCA Data (e.g. OTP)
PUT {host}/{aspsp-cde}/{link_to_update_Auth_Data}; scaAuthenticationData
Response=200
GET {host}/{aspsp-cde}/v1/{consent-id}/status
Inform User about consent success
Response=200; "transactionStatus": "VALD"
GET {host}/{aspsp-cde}/v1/accounts
Account Information API
Response=200; "accountList": [{account_info_1}, {account_info_n}]
GET {host}/{aspsp-cde}/v1/accounts /{account-id}
After the first GET accounts, these API calls may be repeated in any order maximum 4 times per day* per account, while the consent is valid
Response=200; "account": account_details
GET {host}/{aspsp-cde}/v1/ accounts/{account-id}/balances
Response=200; "balances": ["authorised": authorised_balance, ...]
GET {host}/{aspsp-cde}/v1/ accounts/{account-id}/transactions
Shows the information to the User
Response=200; "transactions":{"booked": trans_list},{"pending":trans_list}

* If the PSU is not involved in the requests, in real time

Figure 10 - Consent creation and account information flow for the embedded authentication approach


PSU
TPP
SIBS API Market
ASPSP
Definition of the Consent scope
POST {host}/{aspsp-cde}/v1/consents
Request User (PSU) Identification and password
Response=201;"_links":{"updatePsuAuthentication": "link_to_update_PSU_Data"}
Consent API
Enter User (PSU) Identification and password
PUT {host}/{aspsp-cde}/{link_to_update_PSU_Data}; PSU_ID; PSU_Passw
Request SCA Data (e.g. OTP)
Response=200;"_links":{"authoriseTransaction": "link_to_update_Auth_Data"}
Enter SCA Data (e.g. OTP)
PUT {host}/{aspsp-cde}/{link_to_update_Auth_Data}; scaAuthenticationData
Response=200
GET {host}/{aspsp-cde}/v1/{consent-id}/status
Inform User about consent success
Response=200; "transactionStatus": "VALD"
GET {host}/{aspsp-cde}/v1-0-4/card-accounts
Response=200; "accountList": [{account_info_1}, {account_info_n}]
Card Information API
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}
Response=200; "account": account_details
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}/balances
Response=200; "balances": ["authorised": authorised_balance, ...]
GET {host}/{aspsp-cde}/v1-0-4/card-accounts/{account-id}/transactions
Shows the information to the User
Response=200; "transactions":{"booked": trans_list},{"pending":trans_list}
After the first GET card accounts, these API calls may be repeated in any order maximum 4 times per day* per account, while the consent is valid

* If the PSU is not involved in the requests, in real time

Figure 11 - Consent creation and card-account information flow for the embedded authentication approach

6.3 Authorisations

The new version of Berlin Group's NextGen PSD2 framework foresees the possibility of carrying out transactions that depend on the authorization of several account holders, as is often the case for companies, exclusively through APIs. To allow the authorization of a transaction by several holders, and respective authentications, a new authorization management model was introduced, based on Authorization Resources. For each payment initiation or consent resource, one or more authorization resources are created. The entire authorization and authentication process is managed through these resources.

Implicit and explicit creation of Authorization Resources

The BG specification provides for the possibility of creating authorization resources implicitly in the POST for creating the payment initiation and consent.

- Implicit creation of the authorization resource

Whenever the TPP does not send the TPP-Explicit-Authorisation-Preferred flag, or sends it equal to “false”, an implicit creation of the authorization resource is performed, regardless of whether only one or multiple authentications are required. The PSU, the only one in case only one authorization is required, or the first of several in case several authorizations are required, must carry out the authentication immediately after the response to the POST for creation of payment initiation and consent and the DELETE for cancellation payment where authentication is required.

This is the default mechanism for the majority of ASPSP. Please note that specially for consent creation, vast majority of ASPSP only require authorization by one PSU and therefore privilege the implicit creation of the authorization resource.

- Explicit creation of the authorization resource

Whenever the TPP sends the Authorization-Preferred = true flag, the authorization resource must be explicitly created by the TPP, regardless of whether it requires only one or several authentications.

When the TPP invokes an API to query the Bank's Payment/Consent Status, the status of the associated authorizations is not returned, i.e., SIBS may have consents and payments in the final status, however, the respective authorizations (or some of them) may be in non-final status.

These authorization statuses are updated when the TPP invokes the API to query authorization status or when the bank updates the respective status.

Explicit creation of the authorization resource may not be offered by some ASPSP, specially at an early stage of the implementation of release 4.

The possible values for the “transactionStatus”/code parameter, and the definition of the final status, are included in the following table:

Table 7 – Authorisations - possible values for the “transactionStatus”/code parameter

“transactionStatus”	Status	Definition
RCVD	Received	An authorisation or cancellation-authorisation resource has been created successfully. This is an initial status.
SCAM	SCAMethodSelected	The PSU/TPP has selected the related SCA routine. If the SCA method is chosen implicitly since only one SCA method is available, then this is the first status to be reported instead of "Received". This is an initial or intermediate status.
PSAT	PSUAuthenticated	The PSU related to the authorisation or cancellation-authorisation resource has been identified and authenticated e.g. by a password or by an access token. This is an intermediate status.
STRT	Started	The addressed SCA routine has been started. This is an intermediate status.
FNLS	Finalised	The SCA routine has been finalised successfully (including a potential confirmation command). This is a final status.
FALD	Failed	The SCA routine failed. This is a final status.
EXMP	Exempted	SCA was exempted for the related transaction, the related authorisation is successful. This is a final status.

7 FAQs

1. **How many days of transactions can be retrieved directly after the consent has been created or by a separate authorization from the PSU?**

   There is no limit of transactions history request, vary among ASPSP, according with the transaction history they provide in their digital channels. When all the transaction history is requests, ASPSP will provide all the information they have, provided user is authenticated.

2. **How many days of transactions can be retrieved via recurring transactions calls with a consent token?**

   The transaction history of the last 90 days is provided without PSU involved.

3. **If a transaction ID is provided by your API, is the provided ID consistent within a session (access token), a consent or forever?**

   The transaction ID is consistent within a session.

4. **What is the maximum number of days for which an approved consent token can be used before it expires and has to be refreshed or renewed?**

   Since July 25th 2023 the maximum number of days of a consent is 180 days. Before, the maximum number of days was 90 (according to the revised article 10 of EBA RTS).

5. **Is a recurring access to the list of accounts via a consent token supported?**

   Yes.

6. **Is a recurring access to the additional account details via a consent token supported?**

   Yes.

7. **Is a recurring access to the list of transactions via a consent token supported?**

   Yes.

8. **Is a recurring access to the balance via a consent token supported?**

   Yes.

9. **Is it possible for the PSU to revoke a consent via their online banking portal?**

   Yes (status ‘RevokedByPSU’).

10. **How is it possible to terminate a consent via API?**

    The 3 ways to terminate a consent are: “Revoked by PSU” (via ASPSP direct channel), “Terminated by TPP” or upon reaching the expiration date “Expired”.

11. **Can the lifetime of a consent be configured by the TPP?**

    Yes, considering the maximum number of days of a consent is 180 days.

12. Can the lifetime of a consent be changed by the PSU?

Yes, considering the maximum number of days of a consent is 180 days.

13. Can the scope of a consent be configured by the TPP?

Yes.

14. Is the PSU able to change the scope of a consent (e.g.: prior to the final confirmation)?

Yes. However, after the consent creation, any scope change will trigger a new consent request that will replace the previous consent (consequently having a new consent ID assigned).

15. Can the TPP configure accounts for a consent?

Yes, there are two options for consent account request: the TPP can request access all accounts, or let the PSU selects the accounts he wants to give access. This may depend on ASPSP implementation.

16. Can the PSU reconfigure accounts for a consent?

No, just before consent confirmation and in case the TPP lets the PSU select the accounts they want to give access to.

17. What is the process to deploy new releases?

The standard procedure for implementing changes to the technical specifications or deploying releases of the SIBS API Market's interface involves notifying TPPs. This notification is accompanied by the relevant version of the Manual and is provided at least three months prior to the scheduled implementation of the change (according to the point 4 of article 30 of EBA Regulatory Technical Standards (RTS) on strong customer authentication and secure communication under PSD2.

Additionally, whenever a new release is communicated to the TPP, assurance is provided that the developments are already deployed in a Quality environment and the developer’s portal is providing all the API specifications.

Except in emergency situations, where expedited deployment may be necessary to address critical issues. In such cases, TPPs will be promptly notified of the emergency release and any relevant information pertaining to the deployment process.

18. How is the certificate renewal process carried out?

The process for renewing elDAS certificates (QSEAL and QWAC) for TPPs on the SIBS API Market platform in the QLY and PRD environments follows the same steps as the initial registration process. TPPs seeking to renew their certificates should follow these steps:

1. Access the following link: TPPs Registration - SIBS Pay;
2. Choose the option "I want to consume (Required)" and select "PSD2 APIs";
3. Upload the PKCS#10 for both certificates (QSEAL and QWAC), ensuring they adhere to the specified file types: zip, with a maximum file size of 2 MB;
4. Fill in the requested fields, paying special attention to the mandatory fields such as "Name", "Email" and "Phone Number".

No unnecessary TPP information is required for proceeding with TPP registration or certification renewal.

After completing the above process, you will receive notification from SIBS's support team confirming that the renewal process has been successfully completed

19. Is it possible to use different API versions?

The standard procedure on the SIBS API Market platform is to perform request/response operations using the same API version for the same ASPSP. Any deviation from this standard process would depend on the ASPSP's (Account Servicing Payment Service Provider) specifications. However, it's important to note that using different API versions may not accommodate all information according to the specific API version or ASPSP's specifications. Therefore, it's generally recommended to adhere to the standard procedure of using the same API version for request/response operations on the SIBS API Market platform.

It is possible to TPPs to use different API versions for different ASPSPs, especially if these ASPSPs are at different stages of updating or adopting the latest versions of the APIs.

For example, one ASPSP may have migrated to a new version of the API, while another ASPSP is still using the previous version. In this case, TPPs interacting with these ASPSPs will need to be aware of the different API versions and adapt their integrations according to each ASPSP's specifications, accordingly with API “List of Banks”.

20. What is the process to enter in Production environment with my application?

To transition your application to the production environment after completing the onboarding process and thorough testing, follow these steps:

1. Submit Request on Developer Portal: Access the Developer Portal and submit a request to transition your application to the production environment. Provide all necessary details and documentation required for the production deployment;
2. Open Ticket: Additionally, open a ticket with the relevant support team to formalize and communicate your request for production environment access. Ensure that the ticket includes essential information about your application and the transition process.

By following these steps and submitting the necessary requests, you can trigger the promotion of your application(s) into production environment. This ensures a smooth transition and proper communication with the platform's support team.

21. What are the differences on displaying balance between a debit and credit accounts?

The display of balances between debit and credit accounts may differ based on their respective natures and functionalities. Here are some key differentiations:

1. Debit Account (Checking Account):
    - Available Balance: Typically displays the actual amount of funds available for withdrawal or spending.

- Account Balance: Shows the total amount of funds in the account, including pending transactions and any overdraft protection.
- Transaction History: Provides details of both incoming and outgoing transactions, reflecting the actual movement of funds in and out of the account.
- Overdraft Limit: Indicates the maximum negative balance allowed before incurring overdraft fees or penalties.

2. Credit Account (Credit Card Account):
    - Available Credit: Shows the amount of credit available for spending, representing the difference between the credit limit and the current balance.
    - Current Balance: Reflects the total outstanding balance owed on the credit card, including all purchases, cash advances, fees, and interest charges.
    - Minimum Payment Due: Indicates the minimum amount required to be paid by the due date to avoid late fees and penalties.
    - Transaction History: Displays details of all transactions made using the credit card, including purchases, payments, cash advances, and fees.

Overall, while debit account balances represent available funds that can be accessed immediately, credit account balances reflect outstanding debts owed to the issuer, subject to repayment according to the terms of the credit agreement.

22. What type of Cards does ASPSP make available on the API Account Information?

The majority of ASPSPs only display Credit Cards in the API Account Information, as the API Accounts already provide similar information on transactions and balances of the Debit Cards.

23. What type of Cards does ASPSP make available on the API Payment Initiation?

Depending on the services offered by ASPSPs on their digital channels (home banking/mobile app), the API Payment Initiation can offer payment initiation for Debit Cards, Credit Cards, or both, and the related subset of payment instruments available in each.