My Data Account Specification

Notice

Implementation or use of certain elements of this document may require licenses under third party intellectual property rights, including without limitation, patent rights. The Participants of and any other contributors to the Specification are not and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights. This Specification is provided "AS IS," and no Participant makes any warranty of any kind, expressed or implied, including any implied warranties of merchantability, non-infringement of third party intellectual property rights, and fitness for a particular purpose.

MyData Architecture defines the operations and APIs between the Operational Roles (Operator, Source, Sink etc.). Any descriptions or figures of the role's internal structure or operations are for illustrative purposes only.

1 Introduction

This document specifies MyData Account Management.

This document is part of the MyData architecture release 1.2.1. The reader is assumed to be familiar with the 'MyData Architecture - Consent Based Approach for Personal Data Management' document and with the parallel technical specification documents available at https://hiit.github.io/mydata-stack/.
Known deficiencies in this release: account export/import functionality.

1.1 Definitions

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2 Terminology

Key terminology used in this specification is defined in the Glossary of 'MyData Architecture - Consent Based Approach for Personal Data Management' release 1.2 available at https://hiit.github.io/mydata-stack/.

Account Owner is the individual who created and is using the account to link new services (see *Service Linking*) and authorise data flow (see *Authorisation*). Usually Account Owner is the Data Subject as defined in Data Protection legislation.

Authorisation [interaction] Account Owner's act of granting permission for 1) a service to process data or 2) data transfer from a specific Source to a specific Sink. 1) results in a Consent Record and 2) results in a pair of Consent Records (one each for the Source and the Sink) documenting the granted permission.

Consent Record (CR) documents the permission the Account Owner has granted to a specific service. For authorising data processing within a service, the Account Owner creates a single Consent Record for the related service. For authorising data transfer from a specific Source to a specific Sink, the Account Owner creates a pair of Consent Records (one for the Source and one for the Sink). The Source's CR defines, what data can be provisioned to the specified Sink, and the Sink's CR defines, how the data can be accessed. The Sink's CR can also include the permissions for data processing. A Consent Record is a manifestation of legally valid Consent and makes it technically feasible to change or withdraw the consent dynamically. Consent Records are stored in the MyData Account.

Consent Status Record (CSR) is a record MyData Operator sends to a service when status of a consent changes. Service MUST store these records for future use.

MyData Account is a human centric concept in MyData architecture. MyData Account contains Account Owner's digital identity or identities, linked services and service authorisations. MyData Account can include additional data about Account Owner to help in providing improved services.

Service Link Record (SLR) is the outcome of a successful Service Linking. It documents in machine readable form the terms and scope of the agreement between the Account Owner and a single Source or Sink. Service Link Records are stored in the MyData Account.

Service Link Status Record (SSR) is a record MyData Operator sends to a service when status of a Service Link changes. Service MUST store these records for future use.

Service Linking [interaction] Account Owner's act of linking a service (Source or Sink) to their MyData Account. As the result the Service Linking status and parameters are documented within a digital machine-readable record, called a Service Link Record.

Surrogate ID is a *pseudonym* that associates Account Owner's MyData Account to her account at the service being linked (see *Service Linking*). This ID is meaningful only to Operator and to the service that generated it. It is used in communication between these two parties whenever they need to unambiguously refer to a specific Account Owner's MyData Account (messages from service to Operator), or to a specific user account at the service (messages from Operator to service).

1.3 Formats

In MyData Architecture, all data records and their respective digital signatures exchanged between actors are expressed using Javascript Object Notation (JSON). Digital signatures are expressed as JSON Web Signature (JWS)-structures and cryptographic keys as JSON Web Key (JWK)-structures.
In this document, JSON definitions of the data records are presented without JWS structures.
All Timestamps are in UTC in the NumericDate format as defined in [RFC7519].

2 MyData Account model

MyData Account is a key enabler in service linking and data flow authorisation. It stores all Account Owner's service links and consents along with their history in a single place. This helps provide a unified view to all data flow authorizations and it enables the Account Owner to manage and control the authorized data flows.

To perform its function, MyData Account has to process and store some personal data about the Account Owner. Allowing the Operator to use this additional data may enable a more personalised user experience for the Account Owner.

Typically, MyData Account contains at least the following information about Account Owner:

- local credentials
- personal details (first name, last name and date of birth)
- contact details (email address)
- Account Owner's cryptographic keys (at least the public keys, but in some implementations also the private keys)
- Account Owner's Service Links and Authorisations with corresponding Status Records

Additional information that MyData Account MAY contain to provide more personalised user experience

- Account Owner's linked identities, e.g. for single sign-on (SSO) purposes
- preferences for user interfaces
- presets for data flow authorisation
- detailed contact details

The MyData Account has to be hosted by MyData Operator.

Some operations (e.g. exporting the contents of the Account or permanently deleting the Account) may require further verification steps from the Account Owner. Depending on the identity used with the account, this can be implemented e.g. replying to a verification email.

All the information in MyData Account is designed to be portable allowing Account Owner to change between MyData Operators by moving their data from the one Operator to the other one. The implementation of MyData Account portability as a service between Operators is deferred to a later architecture release.

3 MyData Account Transactions

MyData Account supports a number of transactions, which have been presented in the following two sections: Account Management and Operational Support. Account Management contains transactions for managing the MyData Account itself, whereas Operational Support contains transactions to enable Service Linking, Authorisation and Data Connection as described in MyData Architecture specification.

3.1 Account Management

There are three transactions for MyData Account management: creating an Account, exporting the contents of an Account, and deleting an Account.

3.1.1 Account Creation

Motivation
Individual wants to become Account Owner in MyData Ecosystem and start to manage his/her data and related policies.

Prerequisites

- The party creating or using a MyData Account MUST be a natural person, as legal persons are not allowed to be Account Owners

Process (steps refer to Figure 3.1)[1]

- *Step 1*: Operator front-end sends required information about new Account to MyData Account Service
- *Step 2*: MyData Account Service validates[2] delivered information
- *Step 3*: MyData Account Service checks that proposed username is not already taken
- *Step 4*: MyData Account Service creates a new MyData Account
- *Step 5*: MyData Account Service sends verification email to provided email address (step 1)

Outcome

- New MyData Account

Additional info

- MyData Account MUST be activated via verification email before Account can be used. This minimum verification procedure may be overridden - also in other transactions later in this specification - with a more complicated process per applied identity assurance mechanism chosen by the Operator.

[1] This process refers to creating a local identity; Operator MAY also support the use of external identities.
[2] Validation rules have been defined in section 5.1. Internal API specification

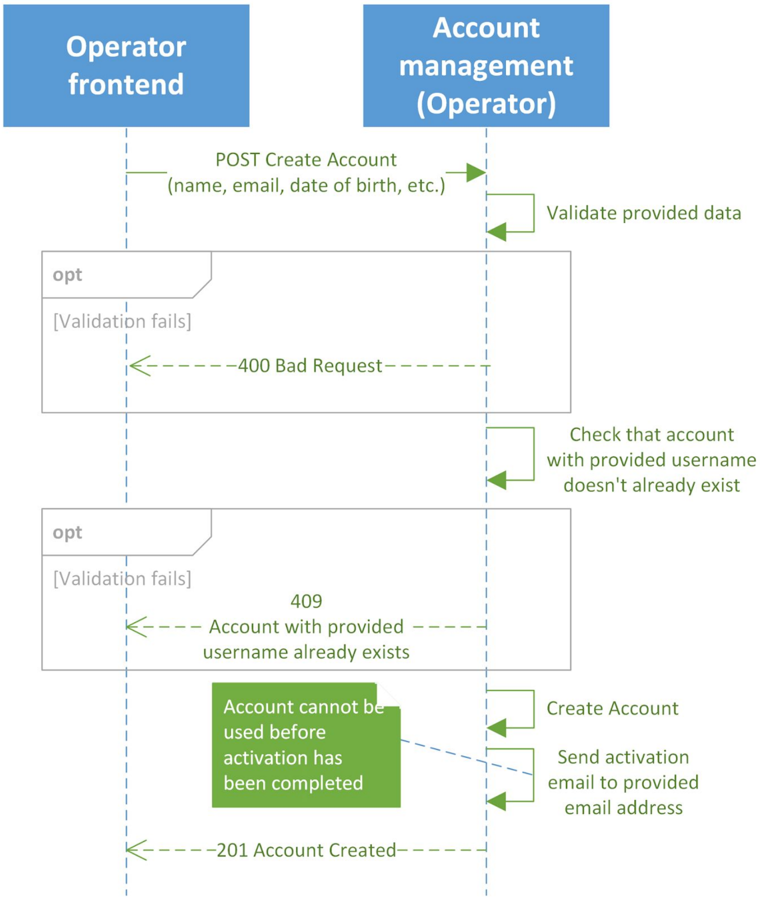
Operator frontend
Account management (Operator)
POST Create Account (name, email, date of birth, etc.)
Validate provided data
opt
[Validation fails]
400 Bad Request
Check that account with provided username doesn't already exist
opt
[Validation fails]
409 Account with provided username already exists
Account cannot be used before activation has been completed
Create Account
Send activation email to provided email address
201 Account Created

Figure 3.1: A simplified Account creation flow

3.1.2 Data Export

Motivation

Account Owner wants to export all data related to his/her MyData Account.

Prerequisites

- Individual has MyData Account at MyData Operator

Process (steps refer to Figure 3.2)

- *Step 1*: Account Owner MUST authenticate himself / herself
- *Step 2*: Operator front-end sends request to export specified MyData Account
- *Step 3*: MyData Account Service checks that authenticated user is authorised to export data of specified MyData Account
- *Step 4*: MyData Account Service sends verification request to the Account Owner with appropriate method, e.g. an email to the verified email address of the MyData Account
- *Step 5*: MyData Account Service collects all data related to specified MyData Account
- *Step 6*: MyData Account Service encapsulates collected data into format preferred by Account Owner
- *Step 7*: MyData Account Service sends export instructions to the Account Owner, in our basic example, via email to the verified email address of the MyData Account

Outcome

- Exported representation of MyData Account in the Account Owner's preferred format (default: JSON)

Additional info

- MyData Account data collection process will be started after email verification has been completed
- Export instructions contain a download link for the MyData Account data in the preferred format

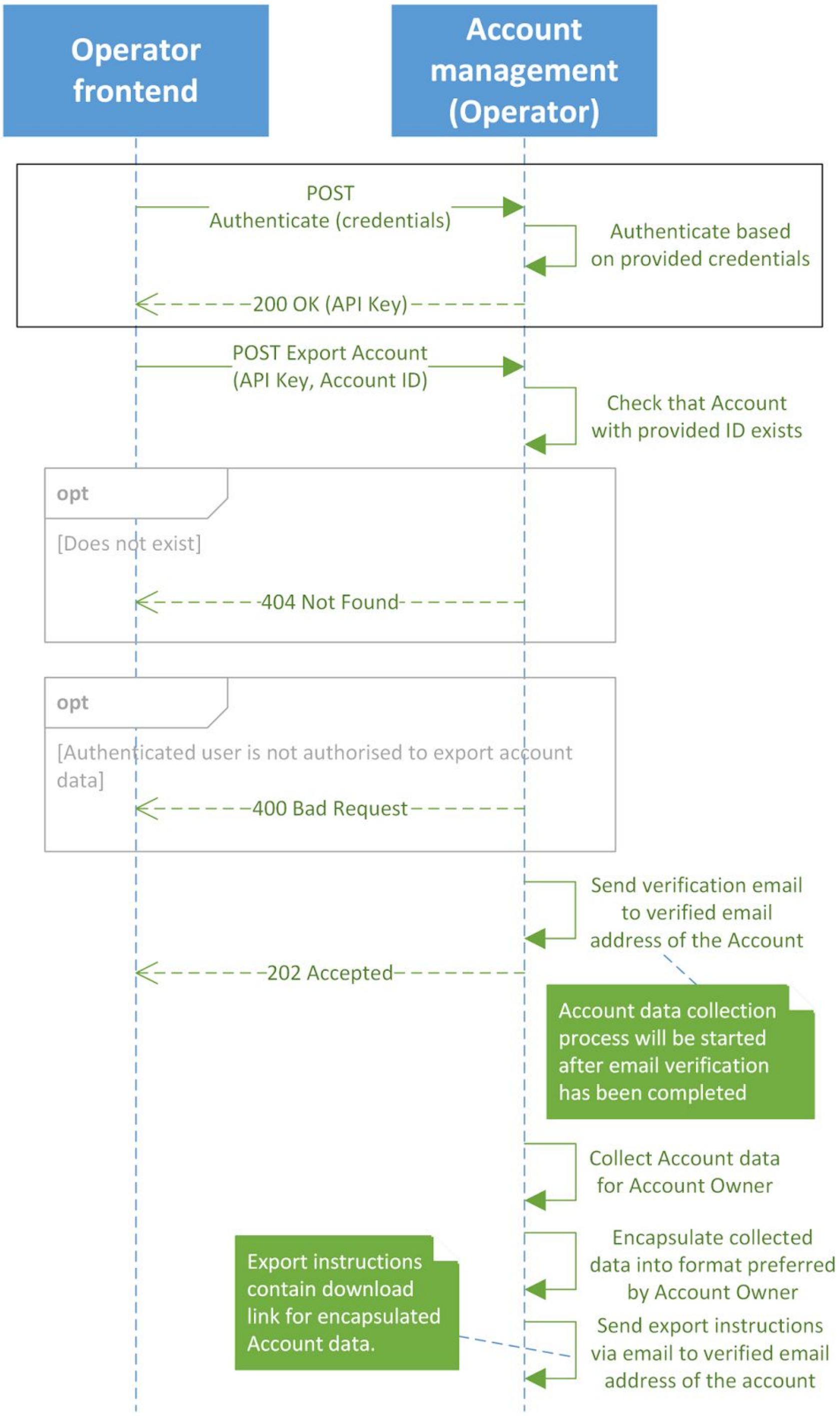
Operator frontend
Account management (Operator)
POST
Authenticate (credentials)
Authenticate based on provided credentials
200 OK (API Key)
POST Export Account
(API Key, Account ID)
Check that Account with provided ID exists
opt
[Does not exist]
404 Not Found
opt
[Authenticated user is not authorised to export account data]
400 Bad Request
Send verification email to verified email address of the Account
202 Accepted
Account data collection process will be started after email verification has been completed
Collect Account data for Account Owner
Encapsulate collected data into format preferred by Account Owner
Send export instructions via email to verified email address of the account
Export instructions contain download link for encapsulated Account data.

Figure 3.2: A simplified Account export flow

3.1.3 Account Deletion

Motivation

Account Owner wants to delete all data related to his/her MyData Account.

Prerequisites

- Individual has a MyData Account at MyData Operator

Process (steps refer to Figure 3.3)

- *Step 1*: Account Owner MUST authenticate himself / herself
- *Step 2*: Client sends request to delete specified MyData Account
- *Step 3*: MyData Account Service checks that authenticated user is authorised to delete data of specified MyData Account
- *Step 4*: MyData Account Service sends verification request to the Account Owner with appropriate method, e.g. an email to the verified email address of the MyData Account
- *Step 5*: MyData Account Service deletes all data related to specified MyData Account

Outcome

- MyData Account is deleted

Additional info

- MyData Account data deletion process will be started after verification has been completed
- Operator MAY provide a grace period during which the deletion can be undone.

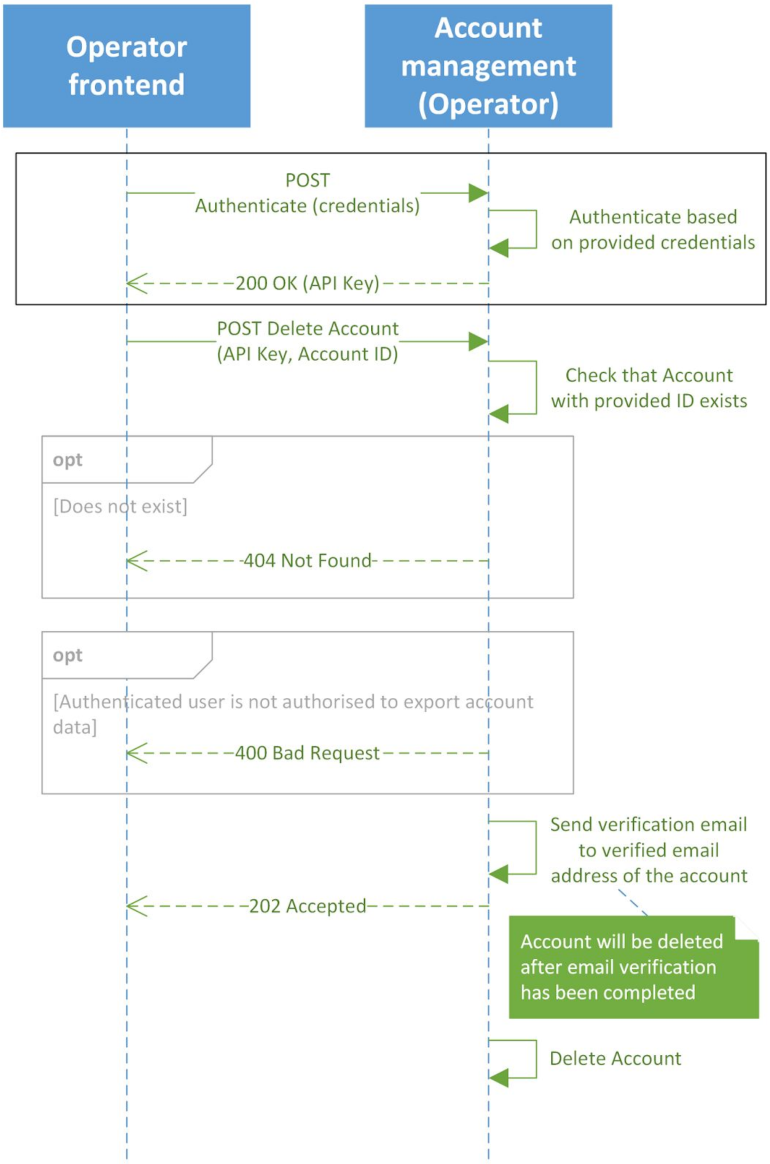
Operator frontend
Account management (Operator)
POST Authenticate (credentials)
Authenticate based on provided credentials
200 OK (API Key)
POST Delete Account (API Key, Account ID)
Check that Account with provided ID exists
opt
[Does not exist]
404 Not Found
opt
[Authenticated user is not authorised to export account data]
400 Bad Request
Send verification email to verified email address of the account
202 Accepted
Account will be deleted after email verification has been completed
Delete Account

Figure 3.3: A simplified Account deletion flow

3.2 Operational Support

There are three transactions: Service Linking, Authorisation, and Logging.

3.2.1 Service Linking

A Service Link consists of two records: a Service Link Record (SLR) and a Service Link Status Record (SSR).

3.2.1.1 Constructing Service Link Record

Motivation

Account Owner wants to manage access to data at a service through the Operator.

Prerequisites

- Individual has MyData Account
- Account Owner has started the linking process at the MyData Operator

Process (steps refer to Figure 3.4)

- *Step 1*: MyData Operator requests MyData Account Service to fill the missing fields to a partial SLR payload and to construct and sign SLR payload
- *Step 2*: MyData Account Service validates the request's payload
- *Step 3*: MyData Account Service fetches the public part of Account Owner's cryptographic key from MyData Operator's Key Management
- *Step 4*: MyData Account Service fills missing fields to partial SLR payload
- *Step 5*: MyData Account Service requests MyData Operator's Key Management to encapsulate SLR payload as JWS and to sign it with Account Owner's cryptographic key
- *Step 6*: MyData Account Service returns Account signed SLR to MyData Operator

Outcome

- Partial Service Link Record (signed only by Account Owner)

Additional info

- See *Service Linking Specification*

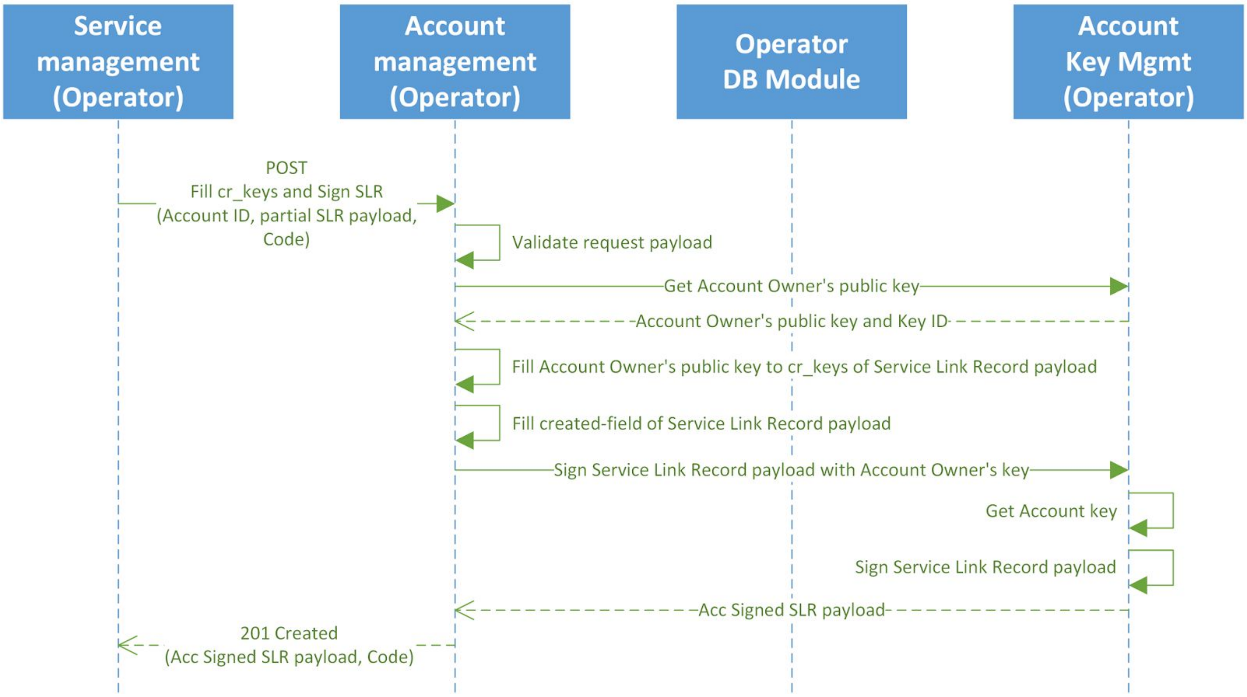
Service management (Operator)
Account management (Operator)
Operator DB Module
Account Key Mgmt (Operator)
POST
Fill cr_keys and Sign SLR
(Account ID, partial SLR payload, Code)
Validate request payload
Get Account Owner's public key
Account Owner's public key and Key ID
Fill Account Owner's public key to cr_keys of Service Link Record payload
Fill created-field of Service Link Record payload
Sign Service Link Record payload with Account Owner's key
Get Account key
Sign Service Link Record payload
Acc Signed SLR payload
201 Created
(Acc Signed SLR payload, Code)

Figure 3.4 : A simplified flow of construction and signing a Service Link Record

3.2.1.2 Constructing a Service Link Status Record

Motivation

Account Owner wants to manage access to data at a service through the Operator.

Prerequisites

- Account Owner has started the linking process at the MyData Operator
- Service Link Record has been constructed and signed (*see previous section*)

Process (steps refer to Figure 3.5)

- *Step 1*: MyData Operator requests MyData Account Service to store SLR and to construct, sign and store SSR payload
- *Step 2*: MyData Account Service validates request's payload
- *Step 3*: MyData Account Service requests MyData Operator's Key Management to verify Account Owner's signature in SLR
- *Step 4*: MyData Account Service fills missing fields to partial SSR payload
- *Step 5*: MyData Account Service requests MyData Operator's Key Management to encapsulate SSR payload as JWS and to sign it with Account Owner's cryptographic key
- *Step 6*: MyData Account Service requests MyData Operator's Database Service to store SLR and SSR to persistent storage
- *Step 7*: MyData Account Service returns Account Owner signed SLR and SSR to MyData Operator

Outcome

- Service Link Record and Service Link Status Record

Additional info

- See *Service Linking Specification*

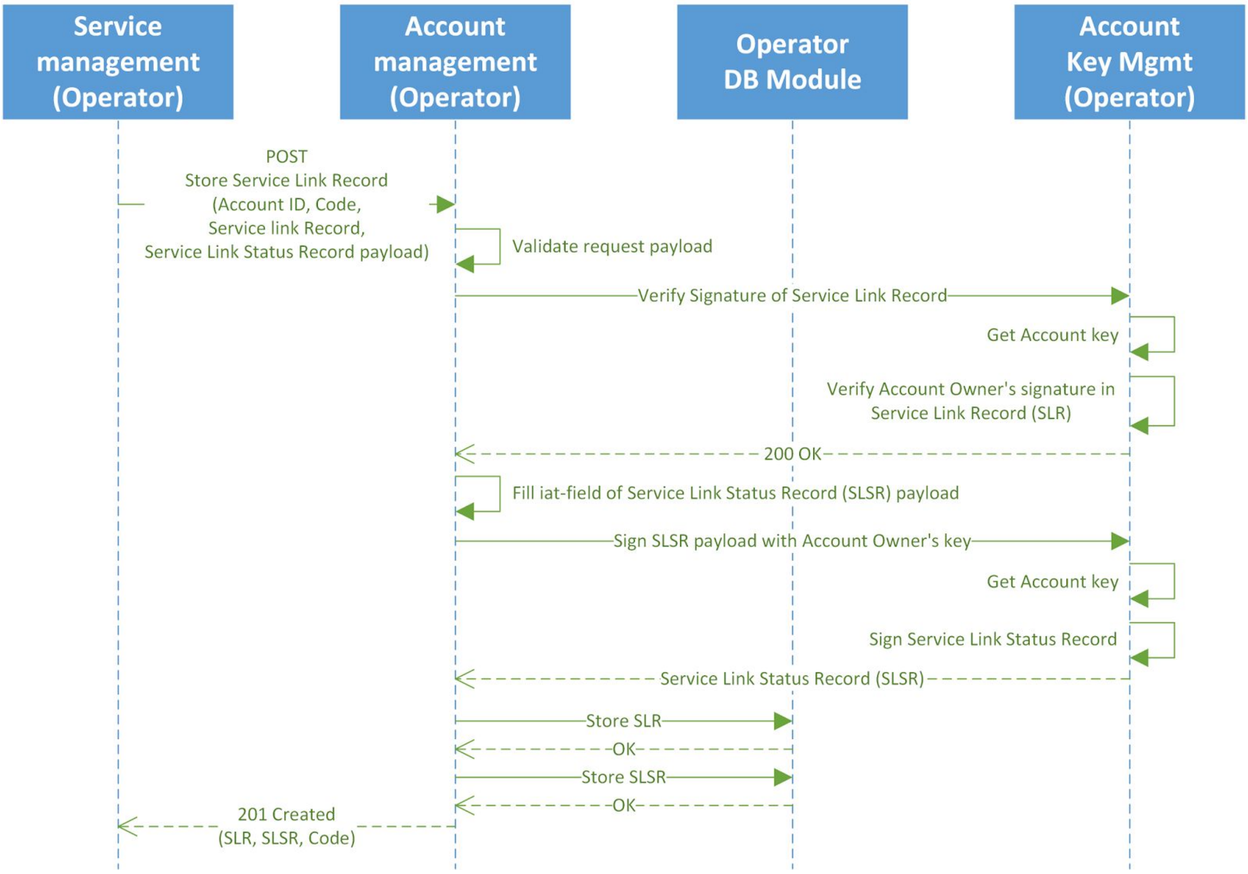
Service management (Operator)
Account management (Operator)
Operator DB Module
Account Key Mgmt (Operator)
POST
Store Service Link Record
(Account ID, Code,
Service link Record,
Service Link Status Record payload)
Validate request payload
Verify Signature of Service Link Record
Get Account key
Verify Account Owner's signature in
Service Link Record (SLR)
200 OK
Fill iat-field of Service Link Status Record (SLSR) payload
Sign SLSR payload with Account Owner's key
Get Account key
Sign Service Link Status Record
Service Link Status Record (SLSR)
Store SLR
OK
Store SLSR
OK
201 Created
(SLR, SLSR, Code)

Figure 3.5 , A simplified flow of validating Account Owner's signature in a Service Link Record and constructing and signing a Service Link Status Record

3.2.2 Authorisation

MyData Authorisation consists of a Consent Record (CR) and a Consent Status Record (CSR).

Motivation
Account Owner wants to authorise data flow from one service to another service.

Prerequisites

- Account Owner has completed Service Linking at least one Source and one Sink service
- Account Owner has started the consenting process at the MyData Operator

Process (steps refer to Figure 3.6)

- *Step 1*: MyData Operator requests Surrogate ID and Service Link Record ID based on Service ID and Account ID from MyData Account Service. This step is executed for both Source and Sink Services.
- *Step 2*: MyData Operator generates CR payloads
- *Step 3*: MyData Operator generates CSR payloads
- *Step 4*: MyData Operator requests MyData Account Service to sign and store CR and CSR
- *Step 5*: MyData Account Service validates request's payload
- *Step 6*: MyData Account Service requests MyData Operator's Key Management to encapsulate CR payloads and CSR payloads as JWS' and to sign those with Account Owner's cryptographic key. This step is executed for both Source and Sink Services.
- *Step 7*: MyData Account Service requests MyData Operator's Database Service to store CR and CSR to persistent storage. This step is executed for both Source and Sink Services.
- *Step 8*: MyData Account Service returns Account Owner signed CRs and CSRs to MyData Operator

Outcome

- Consent Records and Consent Status Records

Additional info

- See *Authorisation Specification*
- Support for granting permission for a service to (further) process data it already has (use case listed in Terminology/Authorisation section and in *MyData Architecture specification*) is deferred to a later architecture release.

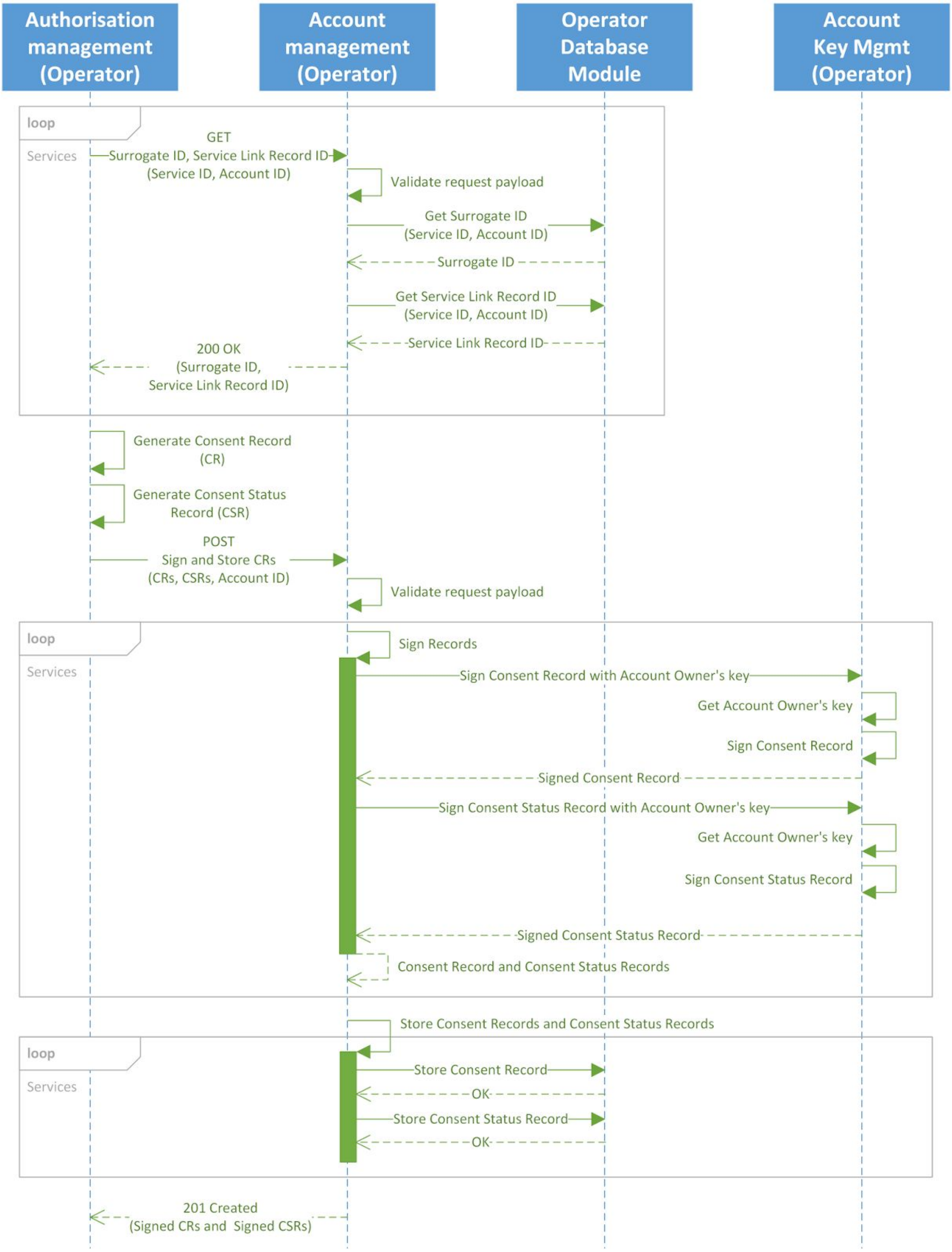

Figure 3.6: A Simplified MyData Authorisation flow

3.2.3 Logging

Logging is deferred to a later architecture release.

4 Data model

This section describes the MyData Account data model. A detailed example of a database implementation is also presented.

4.1 Identities

MyData Account's identity model is shown in Figure 4.1. MyData Account has a mandatory local identity. Account Owner may also link identities provided by third parties to his/her Account, which can be used to enable features such as single-sign-on (SSO).

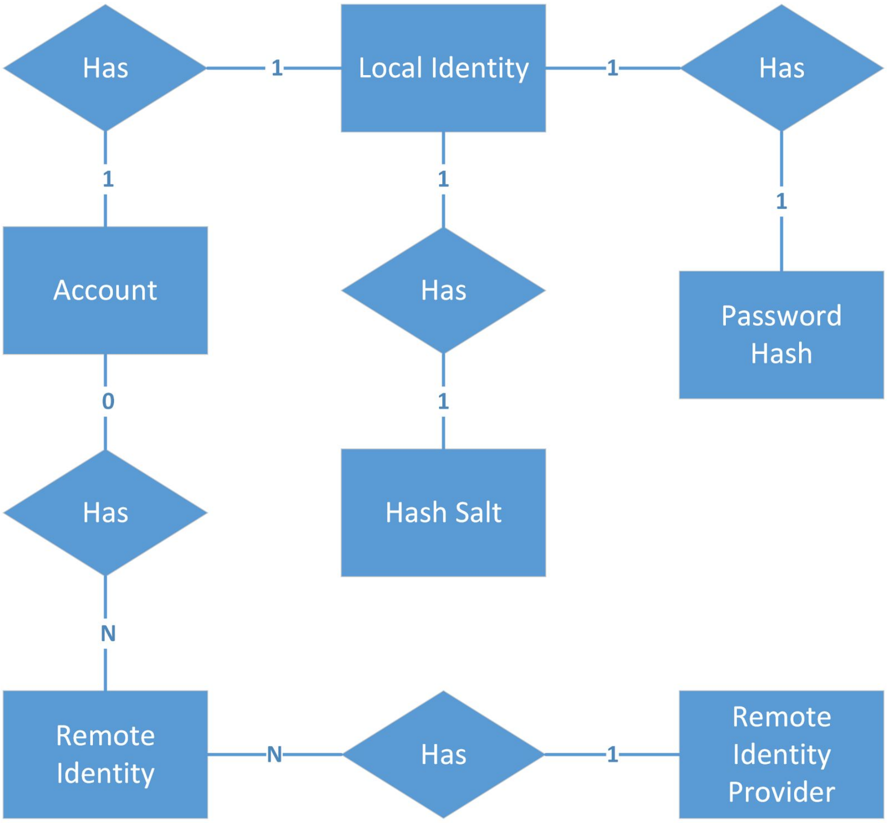

Figure 4.1: ER-model of identity information related to MyData Account

4.2 Personal details and settings

This section describes the personal details in MyData Account. As described in section 2 and in Figure 4.2, not all information in personal details is mandatory.

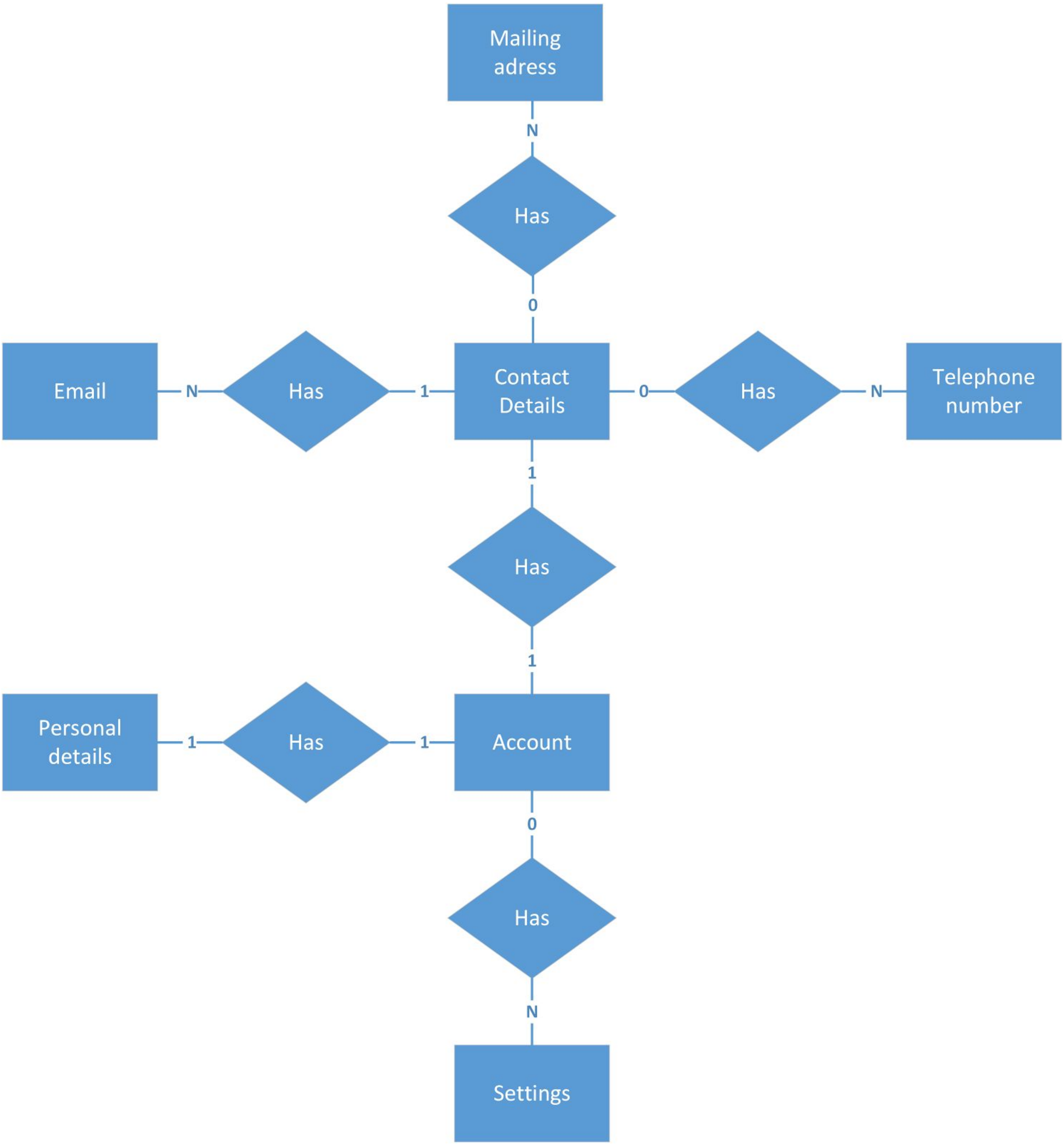

Figure 4.2: ER-model of personal details related to MyData Account

4.3 Service Links and Consent Records

Figure 4.3 shows, how Service Link Records, Consent Records and the related Status Records are used in MyData Account. For more information see *Service Linking Specification* and *Authorisation Specification*.

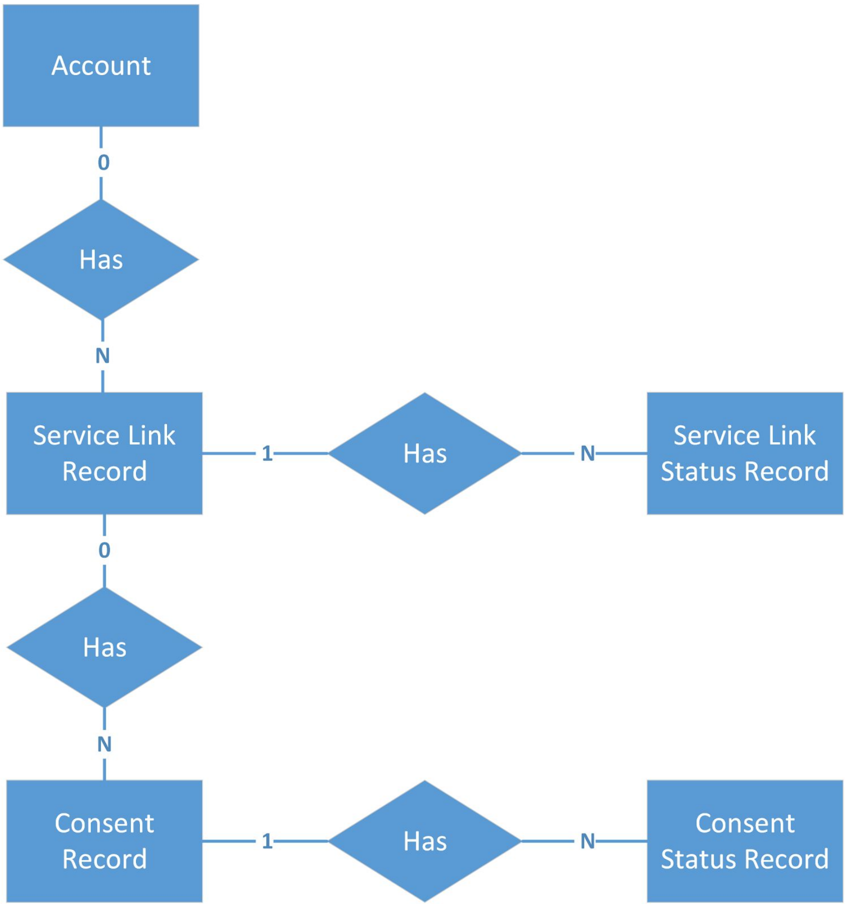

Figure 4.3: ER-model of Service Links and Consents related to MyData Account

4.4 Database model

Example implementation of version 1.2.1 of MyData Account Database with MySQL is shown in Figure 4.4.

Link to the EER model at MyData SDK

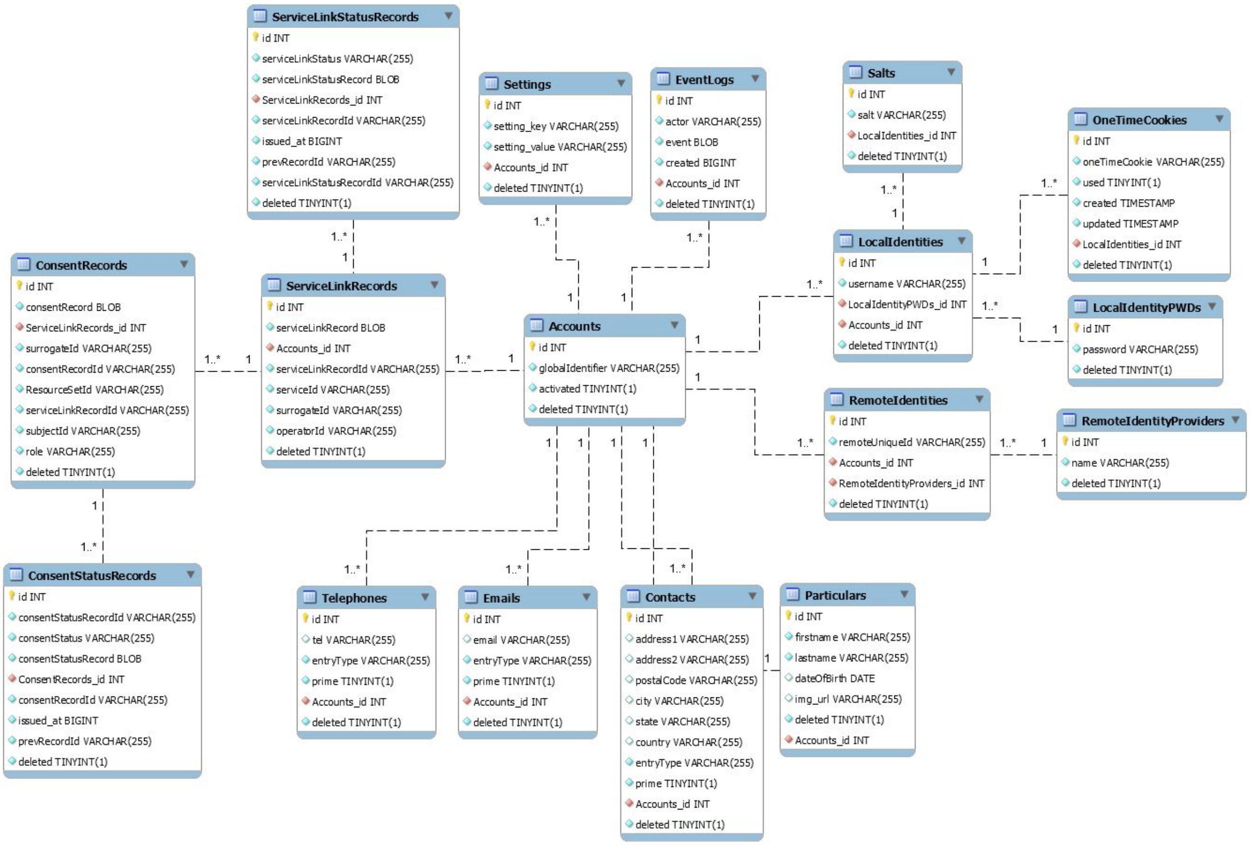

Figure 4.4: EER model of Account as MySQL database

4.5 Data Export

A default JSON data structure of MyData Account Export is shown in Table 4.1. The actual data format of the export can vary depending on implementation, for the purposes of account data interoperability and data exchange it is suggested to only use the schema described here.

Table 4.1: JSON Schema presentation of an exported MyData Account.

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "Account": {
      "type": "object",
      "properties": {
        "account_id": {
          "type": "string"
        },
        "activated": {
          "type": "string"
        },
        "ConsentRecords": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "consentRecord": {
                "type": "string"
              }
            }
          }
        },
        "ConsentStatusRecords": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "consentStatusRecord": {
                "type": "string"
              }
            }
          }
        },
        "Contacts": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "address1": {
                "type": "string"
              },
              "address2": {
                "type": "string"
              },
              "postalCode": {
                "type": "string"
              },
              "city": {
                "type": "string"
              },
              "state": {
                "type": "string"
              },
              "country": {
                "type": "string"
              },
              "type": {
                "type": "string"
              },
              "prime": {
                "type": "string"
              }
            }
          }
        },
        "Emails": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "email": {
                "type": "string"
              },
              "type": {
                "type": "string"
              },
              "prime": {
                "type": "string"
              }
            }
          }
        },
        "EventLogs": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "actor": {
                "type": "string"
              },
              "event": {
                "type": "string"
              },
              "timestamp": {
                "type": "string"
              }
            }
          }
        },
        "LocalIdentities": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "username": {
                "type": "string"
              }
            }
          }
        },
        "PersonalDetails": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "firstname": {
                "type": "string"
              },
              "lastname": {
                "type": "string"
              },
              "dateOfBirth": {
                "type": "string"
              },
              "img_url": {
                "type": "string"
              }
            }
          }
        },
        "ServiceLinkRecords": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "serviceLinkRecord": {
                "type": "object",
                "properties": {}
              }
            }
          }
        },
        "ServiceLinkStatusRecords": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "serviceLinkStatusRecord": {
                "type": "string"
              }
            }
          }
        },
        "Settings": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "key": {
                "type": "string"
              },
              "value": {
                "type": "string"
              }
            }
          }
        },
        "Telephones": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "tel": {
                "type": "string"
              },
              "type": {
                "type": "string"
              },
              "prime": {
                "type": "string"
              }
            }
          }
        }
      }
    }
  }
}
```

5 Account APIs

Account has been implemented as a service in MyData SDK. MyData Account Service's API specifications are provided as Swagger YAML.

5.1 Internal Account API specification

API exposed for MyData Operator's internal functions and components. The YAML file can be found at https://github.com/HIIT/mydata-sdk/blob/v1.2.1/Account/doc/api/account_api_internal.yaml

and a viewer-friendly version behind Swagger Editor at

http://editor.swagger.io/#/?import=https://raw.githubusercontent.com/HIIT/mydata-sdk/v1.2.1/Account/doc/api/account_api_internal.yaml.

5.2 External Account API specification

API exposed for realising an Operator front-end e.g. as a mobile native app or a web app. The YAML file can be found at

https://github.com/HIIT/mydata-sdk/blob/v1.2.1/Account/doc/api/account_api_external.yaml

and a viewer-friendly version behind Swagger Editor at

http://editor.swagger.io/#/?import=https://raw.githubusercontent.com/HIIT/mydata-sdk/v1.2.1/Account/doc/api/account_api_external.yaml

References

[RFC2119] Bradner, S, "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC7515] Jones, M, Bradley, J, Sakimura, N, JSON Web Signature”, RFC 7515, May 2015.