

Manual for Unified Messaging / Hector

Version 7.24.3026, 2026-05-12T15:44:05Z

Table of Contents

1. A word on versioning	1
1.1. Major version	1
1.2. Minor version	1
1.3. Patch version	1
2. APIs	2
2.1. SOAP	2
2.2. REST	2
2.2.1. V2 and V3	2
2.2.2. V4	2
3. Recip-e	4
3.1. General	4
3.2. Environments	5
3.3. As a prescriber	5
3.3.1. General	5
3.3.2. Prerequisites	5
3.3.3. Frontend module	6
3.3.4. Backend Module	9
3.3.5. Older versions	17
3.4. As an executor	17
3.4.1. General	17
3.4.2. Prerequisites	18
3.4.3. Backend module	19
4. Consulting Vitalink	25
4.1. Authentication & error handling	25
4.2. Patient consent	26
4.2.1. Get patient consent	27
4.2.2. Revoke patient consent	27
4.2.3. Register patient consent	28
4.3. Therapeutic link	29
4.3.1. Get therapeutic link	30
4.3.2. Revoke therapeutic link	31
4.3.3. Register therapeutic link	31
4.4. Medication scheme	32
4.5. SUMEHR	33
4.6. Sumehr	35
5. Get GMF information	37
5.1. Error flows	38
6. Querying Hector	40

7. Consulting Credentials	41
7.1. Authentication & error handling	41
7.2. Get credentials	42
7.3. Get credentials with filtering	44
7.4. Get credentials for organisations	44
8. Get insurability information	46
8.1. Consult for a specific date	47
8.2. Error flows	47
9. Helena	48
9.1. Authentication & error handling	48
9.2. Upload a laboratory result	48
9.3. Retrieve a patient registration link	50
9.4. Retrieve a patient's application link	51
9.5. Retrieve a patient's status	54
10. wit-gele kruis (WGK)	56
10.1. Authentication & error handling	56
10.1.1. Authentication	56
10.1.2. Hospital Credential	56
10.1.3. Traceability	56
10.1.4. Errors	57
10.2. Therapeutic relationship	57
10.3. General request data-types	57
10.4. General response data-types	58
10.5. Verify data availability	58
10.6. Get application link	59
11. Context Integration (UM Only)	62
11.1. Authentication & error handling	62
11.2. Create an OTP session	62
11.3. Consume the OTP session	64
12. Security	66
12.1. Legacy Mode	66
12.1.1. Configuration	66
12.1.2. Hector	66
12.1.3. Unified Messaging	66
12.1.4. Disabled APIs	66
13. V4 Insurability	68
13.1. Get Insurability (member data)	68
13.1.1. Request by Patient ssin	68
13.1.2. Request by mutuality info	68
13.1.3. Example response	69
13.2. Get Global Medical File Record	72

13.2.1. Request by Patient ssin	72
13.2.2. Request by mutuality info	72
13.2.3. Example response when a doctor is the global medical file record holder	73
13.2.4. Example response when a medical house is the global medical file record holder	73
14. V4 Recip-E	75
14.1. General	75
14.1.1. Supported qualities	75
14.2. Recipe Prescriber	75
14.2.1. Register a prescription	75
14.2.2. Fetch the Prescription / pdf / kmehr / preview	79
14.2.3. Fetch all prescriptions	80
14.2.4. Refresh the status	81
14.2.5. Revoke the prescription	82
14.2.6. Delete prescription	82
14.2.7. Send notification	82
14.2.8. Get feedbacks	84
15. V4 Rn-Consult	87
15.1. Supported Qualities	87
15.2. Search Phonetically	87
15.2.1. Example Request	87
15.2.2. Example Response	88
15.3. Fetch by SSIN	91
15.3.1. Example Request	91
15.3.2. Example Response	92
15.4. Fetch person history by SSIN	98
15.4.1. Example Request	98
15.4.2. Example Response	98
15.5. Add or remove subscription	104
15.5.1. Example Request	105
15.5.2. Example Response	105
15.6. Get notification	105
15.6.1. Example Request	105
15.6.2. Example Response	106
15.7. Ack notification	118
15.7.1. Example Request	118
15.7.2. Example Response	118
15.8. Get current ssin history	119
15.8.1. Example Request	119
15.8.2. Example Response	119
15.9. Get related ssins history	119
15.9.1. Example Request	119

15.9.2. Example Response	120
15.9.3. Error Codes	120
16. HealthPages	122
16.1. Search Contacts	122
16.2. Search contact cards	122
16.3. Manage caregivers for your organization	122
16.4. Manage departments for your organization	122
16.5. Resolve Addressees	123
16.5.1. Request details	123
16.5.2. Response details	124
16.6. Models	125
16.6.1. <i>ADR</i>	125
16.6.2. <i>Address</i>	125
16.6.3. <i>AddresseeRequest</i>	125
17. V4 Assurmed	126
17.1. Assurmed credential configuration in MyBox	126
17.2. Get configured Assurmed credentials	126
17.2.1. Example Request	126
17.2.2. Example Response	126
17.3. Assurmed proxy endpoints	127
17.3.1. List all attestations	127
17.3.2. Get a single attestation	127
17.3.3. Submit/update a reimbursement request	127
17.3.4. Reference Data	128
18. V4 Vsb	129
18.1. Vsb credential configuration in MyBox	129
18.2. Get configured Vsb credentials	129
18.2.1. Example Request	129
18.2.2. Example Response	129
18.3. Vsb Gen Async proxy endpoints	130
18.3.1. Get	130
18.3.2. Post	131
18.3.3. Confirm	131
18.3.4. Get Insurability	131
19. V4 Timestamping	132
19.1. Timestamping credential configuration in MyBox	132
19.2. Configure user for timestamping	132
19.3. Scheduling timestamping files	133
19.3.1. File system	133
19.3.2. Soap service	133
19.4. Processing files to timestamp	133

19.5. Archiving timestamped files	133
---	-----

Chapter 1. A word on versioning

Every installation is bound to a snapshot in time of our product and every snapshot has a version.

This version is noted using 3 numbers separated by a dot (.), resulting in the following format:

major.minor.patch

Examples:

- 7.13.1679
- 7.14.1688
- 7.15.1686

1.1. Major version

When the major version changes, you should review the entire application.

A change of this version indicates a change in philosophy and/or architecture of the software.

When this version increments, the application is not guaranteed to be backward compatible.

1.2. Minor version

When the minor version changes, you should review existing integrations with the application.

When this version increments, the application is not guaranteed to be backward compatible.

1.3. Patch version

This number is a forever-increasing number, regardless of the major and minor version.

When this version increments, the application is guaranteed to be backward compatible for the same major & minor version, unless otherwise communicated.

Chapter 2. APIs

2.1. SOAP

WARNING

These APIs are considered END OF LIFE.

If you already integrate these services we encourage you to implement the equivalent using the REST APIs as soon as possible.

Support is only available if you already implemented these APIs.

For more information browse to </unified-messaging/services/>

2.2. REST

2.2.1. V2 and V3

NOTE

No new development will be performed for V2 and V3.

All noteworthy APIs for version 2 and 3 are described further down this document.

This document serves as a guide both functionally as technically for these APIs.

eHealth credential identification is performed by passing the X-Credential-Id HTTP Header.
The value for this header is a database identifier.

NOTE

The root endpoint mapping differs for Hector and Unified-Messaging.

Endpoint mapping on Hector:

- </rest/api/v2>
- </rest/api/v3>

Endpoint mapping on Unified-Messaging:

- </unified-messaging/api/v2>
- </unified-messaging/api/v3>

2.2.2. V4

IMPORTANT

New endpoints will be exposed over this api.

This document serves as a functional guide for these APIs.

For technical details please consult the OpenAPI documentation.

eHealth credential identification is performed by passing the Default-Ehealth-Credential-Id HTTP Header.

The value for this header is a functional identifier which can be found on eg. HealthPages.

Root endpoint mapping on Hector **and** Unified-Messaging is equal

- /api/v4

Chapter 3. Recip-e

3.1. General

Our Recip-e module can be used to create, process, and manage prescriptions. There are two types of actors:

- **Prescriber:** The creation, registration and management of prescriptions.
 - To create a prescription, the frontend Recip-e module is used, which is a part of our E-forms Cloud service.
 - To register a prescription, the backend Recip-e module is used, which is integrated in Unified Messaging and Hector.
- **Executor:** The reading, processing and management of prescriptions.
 - To read and process a prescription, the backend Recip-e module is used, which is integrated in Unified Messaging and Hector.

A prescription can have the following statuses:

Status	Description
NOT_DELIVERED	The prescription is registered at Recip-e but not yet delivered to the patient or executed by the executor.
IN_PROCESS	The prescription is being processed by an executor. The prescription typically transitions into this state by scanning the prescription's barcode.
DELIVERED	The prescription is delivered to the patient, it's contents are no longer retrievable from Recip-e.
ARCHIVED	The prescription has been archived by the executor, it's contents are no longer retrievable from Recip-e.
REVOKED	The prescription has been revoked by either the prescriber or an executor. It's contents are no longer retrievable from Recip-e.
EXPIRED	The prescription is expired and can no longer be processed. It's contents are no longer retrievable from Recip-e.

Some of the backend Recip-e module's tasks are delegated to our E-forms Cloud services, so access to these servers is required for this module to work.

The frontend Recip-e module can also be used through MyBox.

3.2. Environments

- Acceptation environment endpoints
 - Cloud core: <https://acc-services.e-forms.be/cloud/api/v1>
 - Catalogue: <https://acc-services.e-forms.be/catalogue/api/v1>
 - Authentication: <https://acc.healthconnect.be/iamWeb>
- Production environment endpoints
 - Cloud core: <https://services.e-forms.be/cloud/api/v1>
 - Catalogue: <https://services.e-forms.be/catalogue/api/v1>
 - Authentication: <https://services.healthconnect.be/iamWeb>

3.3. As a prescriber

3.3.1. General

This chapter will explain the flow to create, register and manage prescriptions as a prescriber. These are the high level steps:

- Create and fill a Medication Prescription eForm
- Export that form as a kmehr
- Register the kmehr at Recip-e

Once the kmehr is registered at Recip-e, it can be managed (follow-up, revocation, ...)

3.3.2. Prerequisites

eHealth certificate

The first requirement is a valid eHealth certificate for the eHealth acceptance environment. The certificate holder needs

to be one of the following qualities in order to be able to prescribe. Without one of these qualities, prescribing will fail.

- Hospital
- Doctor
- Dentist
- Midwife

Functional Unified messaging or Hector installation

The certificate that will be used to prescribe needs to be loaded correctly into the Unified Messaging or Hector installation.

A test procedure is available in Hector to verify that the certificate was loaded correctly.

Authentication on the REST services is done using basic authentication. A username and password

are available on request.

An IAM account for the Healthconnect Cloud

If you want to use our front end Recip-e module, you will need an account for the Healthconnect Cloud.

This account is required for both our acceptance and our production environment. It consists of a username and password that is needed to fetch an authentication token.

TIP | If you are not using the front end module, you can safely skip this step.

3.3.3. Frontend module

The goal is to generate a kmehr for a prescription. In order to do so, the Medication Prescription eForm can be used.

A successful integration of our front end Medication Prescription eForm executes the following steps:

Acquire a Healthconnect Cloud token

Either call the Unified Messaging/Hector to fetch a token, or contact our Cloud IAM directly. The second option is a bit more complicated.

Construct a basic data set

The form is pre-filled by passing a basic data set.

Create and visualize a form instance

The instance will be pre-filled with the basic data set. The user can enter more information.

Export to Kmehr

Once the user signals that the form is filled correctly, export it to a Kmehr.

This Kmehr will be handed off to our Unified Messaging or Hector for registration with Recip-e.

Acquire a Healthconnect Cloud token

There are two ways to acquire a Healthconnect Cloud token. The easiest way is by invoking the Unified Messaging or Hector. They will use the certificate loaded in the prerequisites step to initialize an STS session. Using the STS token, our application will contact our cloud authorization service to retrieve a token.

Any integrator is free to bypass our Unified Messaging or Hector and invoke the authorization service directly.

The resulting token is exactly the same so both solutions are equivalent.

We will document only the 'easiest' of both solutions here.

For more information about directly contacting our authorization service, please consult the [E-forms cloud documentation](#).

A token can be obtained by performing a **POST** to the endpoint **api/v2/cloud-token**.

Since our applications support several certificates, you must specify which credential to use to retrieve the token.

This can be achieved in two ways:

- Use the 'X-Credential-Id' header (preferred)
- Pass a selector as request-body

You can only provide one of the two. If both are supplied, an error-response will be returned.

To use the 'X-Credential-Id' header, simply provide a header with the request containing 'X-Credential-Id' as key, and the id of the credential as value.

To obtain the id of the desired credential, the [credential API](#) can be used.

The selector is constructed by the following triplet: an identifier, a type for the identifier (SSIN, NIHII-*, CBE, EHP)

and an application id. The application id is not allowed for person based certificates and optional for organization certificates.

IMPORTANT

Try to construct a selector matching the prescriber as close as possible. The cloud token contains information about the originating certificate. This is used to help pre-fill the form and store commonly used prescriptions. For an optimal user experience, the user must be the owner of the certificate.

IMPORTANT

There are a few cases, where the selector, although fully provided, will match multiple certificates (e.g. two credentials for the same person (one person having two qualities)). In that case, an error response will occur with a 409 status and a matching message. To prevent this, please use the 'X-Credential-Id'.

Ehealth certificate selector

```
{
  "identifier": "86072121285",
  "type": "SSIN"
}
```

Organizations can pass an application id.

Ehealth certificate selector

```
{
  "identifier": "71071328",
  "type": "NIHII-HOSPITAL",
  "applicationId": "FLOWTCT"
}
```

If you are targeting an organizational certificate, you can specify an **applicationId**. This is not supported for certificates where the holder is a person.

Supported types are:

For organisations

CBE, NIHII-LABO, NIHII-HOSPITAL, NIHII-PHARMACY, EHP, NIHII-PSYCH_HOUSE, NIHII-GUARD_POST, NIHII-GROUP_DOCTORS, NIHII-OFFICE_DOCTORS, NIHII-RETIREMENT, NIHII-OTD_PHARMACY, NIHII-GROUP, CBE-TREAT_CENTER, NIHII-MEDICAL_HOUSE

For individuals

SSIN

This list can grow over time. For the most up to date values, consult the eHealth documentation.

Example request

```
POST /unified-messaging/api/v2/cloud-token HTTP/1.1
Authorization: Basic base64("username:password")
Content-Type: application/json; charset=UTF-8

{
  "identifier": "86072121285",
  "type": "SSIN"
}
```

Path	Type	Description
identifier	String	The identifier of the certificate.
type	String	The type of the identifier of the certificate.
applicationId	String	The application id of the certificate.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "accessToken": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOiJhYmZkYmQ..."
}
```

Construct a basic data set

For more information on how to construct a basic data set, please consult the [E-forms cloud documentation](#).

Create and visualize form instance

The front end form is built entirely on E-forms cloud.

All the steps are documented in detail in the [E-forms cloud documentation](#) under *integrate*.

The relevant sections are given for each step.

1. A first step here is to consult the catalogue to retrieve the latest version of the **MEDICATION-PRESCRIPTION** form. **(section 3.1.3 & section 3.2.2)**
2. With the latest version number, a form instance can be constructed. **(section 3.1.4)**
3. The basic data set created in the previous step needs to be uploaded and merged into the form. **(section 3.1.5 and 3.1.6)**
4. Once the basic data set is properly uploaded and merged, the lock on the form can be deleted. **(section 3.1.7)**

The form instance is now ready to be displayed to the user for further input. This needs to be done in an iFrame or an embedded browser.

There is no need to set a sender or recipients for this form. The Recip-e module will handle this.

Attachments are not supported by this form.

Export to Kmehr

Once the user indicates that the form is properly filled, a Kmehr export can be generated. **(section 3.2.1)**

If the Kmehr export succeeded, remove the form instance. **(section 3.1.12)**

3.3.4. Backend Module

The main goal is to register the prescription at Recip-e. This is achieved by passing the kmehr to Unified Messaging / Hector.

Back end authentication does not require a token but is done using basic authentication. The username and password are available on request.

The backend services are exposed on the following endpoints:

- Unified Messaging / Hector: **/api/v4/ recip-e/ prescriber/ prescriptions**

These services are further explained in the V4 API section of this documentation.

NOTE

There is also a deprecated V2 API, which is exposed on the following endpoints: +

- Unified Messaging: **<your_context_path>/api/v2/ recip-e/ prescriptions**
- Hector: **/rest/api/v2/ recip-e/ prescriptions**

The V3 API is also deprecated, and is exposed on the following endpoints: +

- Unified Messaging: **<your_context_path>/api/v3/ recip-**

e/prescriber/prescriptions

- Hector: /rest/api/v3/recipe/prescriber/prescriptions

A successful integration of our V3 back end Medication Prescription module executes the following steps:

Register the Kmehr with Recipe

The Kmehr that was constructed using E-forms Cloud or via your own application can be sent to Recip-e for registration.

This Kmehr needs to be XSD complaint following the guidelines set out by Recip-e.

Retrieve the PDF

The PDF can be visualized and printed using the PDF presentation.

Register a prescription (V3)

To register a prescription, send the kmehr of the prescription together with the Ehealth Credential to use to the Unified Messaging or Hector.

- See the [Frontend module](#) to retrieve the kmehr of the prescription.
- The Ehealth Credential to use is provided by setting the X-Credential-Id header with the id of the desired credential.
 - The available credentials can be retrieved from the backend by using the credential API (see section 5. Consulting Credentials).

Example request

```
POST /unified-messaging/api/v3/recipe/prescriber/prescriptions HTTP/1.1
X-Credential-Id: 23
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/xml; charset=UTF-8
Content-Length: 4505

<?xml version="1.0" encoding="UTF-8"?>
<kmehrmessage xmlns="http://www.ehealth.fgov.be/standards/kmehr/schema/v1">
  <header>
    <standard>
      <cd S="CD-STANDARD" SV="1.20">20161201</cd>
    </standard>
    <id S="ID-KMEHR" SV="1.0">19006951001.200901100900000000</id>
    <id S="LOCAL" SL="ID-MEDISOFT" SV="versie 1.23.25.0">8e1c4ea4-3825-48e4-
bcc2b8cadfa7a897</id>
    <date>2016-09-28</date>
    <time>09:00:00</time>
    <sender>
      <hcparty>
        <id S="ID-HCPARTY" SV="1.0">39015378000</id>
        <cd S="CD-HCPARTY" SV="1.11">persdentist</cd>
```

```

        <firstname>Tom</firstname>
        <familyname>Vleminckx</familyname>
    </hparty>
    <hparty>
        <cd S="CD-HCPARTY" SV="1.1">application</cd>
        <name>MySoftware</name>
        <telecom>
            <cd S="CD-ADDRESS" SV="1.1">work</cd>
            <cd S="CD-TELECOM" SV="1.0">phone</cd>
            <telecomnumber>02/100.11.12</telecomnumber>
        </telecom>
        <telecom>
            <cd S="CD-ADDRESS" SV="1.1">work</cd>
            <cd S="CD-TELECOM" SV="1.0">email</cd>
            <telecomnumber>tom@mysoftware.com</telecomnumber>
        </telecom>
    </hparty>
</sender>
<recipient>
    <hparty>
        <id S="ID-HCPARTY" SV="1.0">RECIPE</id>
        <cd S="CD-HCPARTY" SV="1.1">orgpublichealth</cd>
        <name>Recip-e</name>
    </hparty>
</recipient>
</header>
<folder>
    <id S="ID-KMEHR" SV="1.0">1</id>
    <patient>
        <id S="ID-PATIENT" SV="1.0">76020727360</id>
        <firstname>Fred</firstname>
        <familyname>Flintstone</familyname>
        <birthdate>
            <date>1976-02-07</date>
        </birthdate>
        <sex>
            <cd S="CD-SEX" SV="1.1">male</cd>
        </sex>
    </patient>
    <transaction>
        <id S="ID-KMEHR" SV="1.0">1</id>
        <cd S="CD-TRANSACTION" SV="1.10">pharmaceuticalprescription</cd>
        <date>2016-09-28</date>
        <time>09:00:00</time>
        <author>
            <hparty>
                <id S="ID-HCPARTY" SV="1.0">39015378000</id>
                <cd S="CD-HCPARTY" SV="1.11">persdentist</cd>
                <name>Vleminckx Tom</name>
                <address>
                    <cd S="CD-ADDRESS" SV="1.1">work</cd>

```

```

        <country>
            <cd S="CD-FED-COUNTRY" SV="1.2">be</cd>
        </country>
        <zip>1000</zip>
        <city>Brussel</city>
        <street>Grote Markt</street>
        <houzenumber>7</houzenumber>
    </address>
    <telecom>
        <cd S="CD-ADDRESS" SV="1.1">work</cd>
        <cd S="CD-TELECOM" SV="1.0">phone</cd>
        <telecomnumber>02/221.21.21</telecomnumber>
    </telecom>
</hccparty>
</author>
<iscomplete>true</iscomplete>
<isvalidated>true</isvalidated>
<heading>
    <id S="ID-KMEHR" SV="1.0">1</id>
    <cd S="CD-HEADING" SV="1.2">prescription</cd>
    <item>
        <id S="ID-KMEHR" SV="1.0">1</id>
        <cd S="CD-ITEM" SV="1.11">medication</cd>
        <content>
            <medicinalproduct>
                <intendedcd S="CD-DRUG-CNK"
SV="LOCALDB">0318717</intendedcd>
                <intendedname>ADALAT OROS 30 COMP 28 X 30
MG</intendedname>
            </medicinalproduct>
        </content>
        <beginmoment>
            <date>2016-09-28</date>
        </beginmoment>
        <lifecycle>
            <cd S="CD-LIFECYCLE" SV="1.9">prescribed</cd>
        </lifecycle>
        <quantity>
            <decimal>1</decimal>
        </quantity>
        <posology>
            <text L="nl">1 tablet per dag</text>
        </posology>
    </item>
</heading>
</transaction>
</folder>
</kmehrmessage>

```

Example response

```
HTTP/1.1 201 Created
Content-Type: application/json; charset=UTF-8
Content-Length: 179
```

```
{
  "id": "BEP06YKWZE7X",
  "created": "2017-07-13T12:35:46Z",
  "status": "NOT_DELIVERED",
  "patient": {
    "ssin": "76020727360",
    "name": "Flintstone",
    "givenName": "Fred",
    "gender": "MALE",
    "dateOfBirth": "1976-02-06T23:00:00Z"
  }
}
```

Path	Type	Description
<code>id</code>	String	The identifier of the certificate.
<code>created</code>	String	Date that the prescription was created.
<code>status</code>	String	The status of the credential.
<code>patient.name</code>	String	The patient name.
<code>patient.givenName</code>	String	The patient given name.
<code>patient.ssin</code>	String	The patient ssin.
<code>patient.gender</code>	String	The patient gender.
<code>patient.dateOfBirth</code>	String	The patient date of birth.

Fetch the Prescription / pdf / kmehr / preview (V3)

This will only look in the local Database. In other words, you can only retrieve prescriptions, that were made by the authenticator.

It's possible to extract the PDF and/or preview (html-representation) from the prescription.

The supplied Accept-Language header will be used for the language of the transcript: *nl*, *en* and *fr* are available.

- To retrieve the json of the prescription, set the Accept header to *application/json*
- To retrieve the PDF, set the Accept header to *application/pdf*
- To retrieve the preview, set the Accept header to *text/html*
- To retrieve the kmehr, set the Accept header to *application/xml*

It is possible for the pdf, preview and kmehr to request it inline with the request parameter inline

set to true

Example request as json

```
GET /unified-messaging/api/v3/recipient/prescriptions/BEP06YKWZE7X HTTP/1.1
X-Credential-Id: 23
Accept-Language: nl-BE
Accept: application/json
Authorization: Basic *****
```

Example response for json

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 179

{
  "id": "BEP06YKWZE7X",
  "created": "2017-07-13T12:35:46Z",
  "status": "DELIVERED",
  "patient": {
    "ssin": "76020727360",
    "name": "Flintstone",
    "givenName": "Fred",
    "gender": "MALE",
    "dateOfBirth": "1976-02-06T23:00:00Z"
  }
}
```

Example request as pdf

```
GET /unified-messaging/api/v3/recipient/prescriptions/BEP06YKWZE7X?inline=true HTTP/1.1
X-Credential-Id: 23
Accept-Language: nl-BE
Accept: application/pdf
Authorization: Basic *****
```

Example response for pdf

```
HTTP/1.1 200 OK
Content-Type: application/pdf
Content-Disposition: inline; filename=BEP06YKWZE7X.pdf
Content-Length: 8309

%PDF-1.4
%
4 0 obj
<<
```

...

startxref
7835
%%EOF

Fetch all prescriptions (V3)

Only local prescriptions will be retrieved.

Example request

```
GET /unified-messaging/api/v3/recipient/prescriber/prescriptions HTTP/1.1
X-Credential-Id: 23
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json
```

Example response for json

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

[{"id": "BEP06YKWZE7X",
  "created": "2017-07-13T12:35:46Z",
  "status": "DELIVERED",
  "patient": {
    "ssin": "76020727360",
    "name": "Flintstone",
    "givenName": "Fred",
    "gender": "MALE",
    "dateOfBirth": "1976-02-06T23:00:00Z"
  }
}, {"id": "BEP13HGZES5B",
  "created": "2017-07-12T11:31:32Z",
  "status": "DELIVERED",
  "patient": {
    "ssin": "76020727360",
    "name": "Flintstone",
    "givenName": "Fred",
    "gender": "MALE",
    "dateOfBirth": "1976-02-06T23:00:00Z"
  }
}]
```

Refresh the status (V3)

The statuses of the prescriptions are synchronized with Recip-e on a regular interval. If you want to be sure that the status is up to date, an explicit refresh of the status is possible.

Example request

```
POST /unified-messaging/api/v3/ recip-e/prescriber/prescriptions/BEP06YKWZE7X/status
HTTP/1.1
X-Credential-Id: 23
Accept-Language: nl-BE
Authorization: Basic *****
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
  "status": "NOT_DELIVERED"
}
```

Revoke the prescription (V3)

It is only possible to revoke a prescription at Recip-e if it has the status *NOT_DELIVERED*. To revoke a prescription, a reason is required.

Example request

```
POST /unified-messaging/api/v3/ recip-e/prescriber/prescriptions/BEP06YKWZE7X/revocation
HTTP/1.1
X-Credential-Id: 23
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json

{
  "reason": "The patient is healthy again, and doesn't need pills anymore."
}
```

Example response

```
HTTP/1.1 204 No Content
Content-Type: application/json; charset=UTF-8
```

Delete prescription (V3)

This endpoint will only delete the prescription from the local database. No actions at Recip-e are taken.

Example request

```
DELETE /unified-messaging/api/v3/ recip-e/prescriber/prescriptions/BEP06YKWZE7X
HTTP/1.1
X-Credential-Id: 23
Accept-Language: nl-BE
Authorization: Basic *****
```

Example response

```
HTTP/1.1 204 No Content
Content-Type: application/json; charset=UTF-8
```

3.3.5. Older versions

There is also an old V2 API for the prescriptions. Here the X-Credential-Id is not required.

The Unified Messaging or Hector will attempt to resolve the credential based on the prescriber and sender in the provided kmehr.

However, if there are multiple credentials that match for the Kmehr an exception will be thrown.

All endpoints are listed under *api/v2/ recip-e/prescriptions/* and match the V3 signatures with exception of the following three:

- Get pdf: *api/v2/ recip-e/prescriptions/{rid}/pdf*
- Get Kmehr: *api/v2/ recip-e/prescriptions/{rid}/kmehr*
- Get Preview: *api/v2/ recip-e/prescriptions/{rid}/preview/{locale}*
 - {locale} must be *nl-BE*, *fr-BE* or *en-BE*

The V3 API requires the X-Credential-Id and resolves the credential using this database identifier. This API is also deprecated.

It is highly recommended using the V4, which requires the Default-Ehealth-Credential-Id: a **HealthcareActorId**.

3.4. As an executor

3.4.1. General

An executor (e.g. a pharmacist in a pharmacy or a pharmacy in a hospital) can read & process a registered prescription at Recip-e.

This chapter will explain the flow to retrieve and process a prescription.

These are the high level steps:

- Fetch the prescription from Recip-e
 - By reading the RID from the barcode, the prescription can be fetched

- Process the prescription
 - After the patient has received the medication, the prescription must be marked as delivered
- Archive the prescription
 - Export the archive-content and store it on a safe external location
 - Mark the prescription as archived.

3.4.2. Prerequisites

eHealth certificate

The first requirement is valid eHealth certificates.

Recip-e support both regular pharmacy and pharmacy inside a hospital.

There are two flows available.

Regular Pharmacy

For the first flow the following qualities must be available in certificates in the Unified Messaging or Hector:

- Pharmacist
- Pharmacy

For every call to the backend module, the desired credential must be supplied through the next headers:

- **X-Credential-Id**: contains the id of the pharmacist credentials.
- **X-Pharmacy-Credential-Id** contains the id of the pharmacy credentials.

For every call to the backend module, the desired credential must be supplied through the above headers.

The backend will use them to retrieve the prescription.

Pharmacy in hospital

For the second flow, request headers identifying the hospital, the pharmacy and the pharmacy holder has to be provided via the request headers:

- The **X-Credential-Id** header contains the id of the hospital.
- The **X-Pharmacy-Nihii** contains the nihii number of the pharmacy in the hospital.
- The **X-Pharmacy-Holder-Ssin** contains the SSIN of the pharmacy holder

For every call to the backend module, the desired credential must be supplied through the above headers.

The backend will use the X-Credential-id combined with the two other headers to retrieve the prescription.

Flow Algorithm

The application always checks if the `X-Pharmacy-Credential-Id` is provided. If so the `X-Credential-Id` is used to retrieve the pharmacist and the `X-Pharmacy-Credential-Id` to retrieve the pharmacy. If not the application uses the `X-Credential-Id` to retrieve the hospital credentials and check that `X-Pharmacy-Nihii` and `X-Pharmacy-Holder-Ssin` are provided.

WARNING Always the first flow will be used if the two flows are mixed up.

Functional Unified messaging or Hector installation

The certificates that will be used to retrieve the prescriptions need to be loaded correctly into the Unified Messaging or Hector installation. A test procedure is available in Hector to verify that the certificate was loaded correctly.

Authentication on the REST services is done using basic authentication. The username and password are available on request and are configured in our cloud authentication module.

If you want to retrieve the PDF or Transcript of the prescription, a working connection to the E-forms Cloud service is required.

3.4.3. Backend module

NOTE Please note that for pharmacy in hospital requests the headers (`X-Pharmacy-Nihii` and `X-Pharmacy-Holder-Ssin`) are mandatory, but optional for the regular pharmacy.

Back end authentication does not require a token but is done using basic authentication. The username and password are available on request.

The backend services are exposed on the following endpoints:

- Unified Messaging: `<your_context_path>/api/v3/recip-e/executor/prescriptions`
- Hector: `/rest/api/v3/recip-e/executor/prescriptions`

A successful integration of our back end Medication Prescription module executes the following steps:

Retrieving the prescription based on a RID (typically from a barcode)

As long as the prescription has the status `NOT_DELIVERED` a qualified person can retrieve it. Once the prescription is retrieved, the status will be updated to `IN_PROCESS`.

Retrieve the PDF or Preview

The PDF or Preview can be retrieved to visually see the contents of the prescription. If the integrating software decides to render the prescription in another way, this step can be skipped.

Update the status

After the prescription is processed, the status must be updated to `DELIVERED`

Archive the prescription

After the prescription is delivered, a backup must be stored on an external fail-safe storage.

Update the status

Once the prescription is locally backed up, the status must be set to *ARCHIVED*

Get a prescription / PDF / preview / kmehr

This is typically done when reading the barcode from the prescription of the patient.

If the prescription has already been retrieved once, it will be fetched from the local Database, otherwise Recip-e is contacted.

Only new prescriptions in *NOT_DELIVERED* can be retrieved from Recip-e.

When a prescription is retrieved with status *NOT_DELIVERED*, **the status will automatically be updated to *IN_PROCESS***.

This can be made undone by setting the status back to *NOT_DELIVERED*.

It's possible to extract the PDF and/or preview (html-representation) from the prescription.

This will contact to Eforms Cloud Service to generate the PDF / preview.

The supplied Accept-Language header will be used for the language of the transcript: nl, en and fr are available.

- To retrieve the json of the prescription, set the Accept header to *application/json*
- To retrieve the PDF, set the Accept header to *application/pdf*
- To retrieve the preview, set the Accept header to *text/html*
- To retrieve the kmehr, set the Accept header to *application/xml*

It is possible for the pdf, preview and kmehr to request it inline with the request parameter inline set to true

Example request for hospital

```
GET /unified-messaging/api/v3/recip-e/executor/prescriptions/BEP06YKWZE7X HTTP/1.1
X-Credential-Id: 3
X-Pharmacy-Nihii: 11010280          <-- mandatory for pharmacy in hospital
X-Pharmacy-Holder-Ssin: 12090822841 <-- mandatory for pharmacy in hospital
Accept-Language: nl-BE
Accept: application/json
Host: localhost:8081
```

Example request for regular pharmacy

```
GET /unified-messaging/api/v3/recip-e/executor/prescriptions/BEP06YKWZE7X HTTP/1.1
X-Credential-Id: 2
X-Pharmacy-Credential-Id: 1 <-- mandatory for pharmacy
Accept-Language: nl-BE
Accept: application/json
```

Host: localhost:8081

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 179
```

```
{
  "id": "BEP06YKWZE7X",
  "created": "2017-07-13T12:35:46Z",
  "status": "IN_PROCESS",
  "patient": {
    "ssin": "76020727360",
    "name": "Flintstone",
    "givenName": "Fred",
    "gender": "MALE",
    "dateOfBirth": "1976-02-06T23:00:00Z"
  }
}
```

Example request for Hospital

```
GET /unified-messaging/api/v3/recv-e/executor/prescriptions/BEP06YKWZE7X HTTP/1.1
X-Credential-Id: 13
X-Pharmacy-Nihii: 11010280          <-- mandatory for pharmacy in hospital
X-Pharmacy-Holder-Ssin: 12090822841 <-- mandatory for pharmacy in hospital
Accept-Language: nl-BE
Accept: application/pdf
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/pdf
Content-Disposition: attachment; filename=BEP06YKWZE7X.pdf
Content-Length: 8309
```

%PDF-1.4

%

4 0 obj

<<

...

startxref

7835

%%EOF

Update the status

There are several cases where the status must be set:

- If for some reason, the prescription cannot be processed (e.g. medicine not available), it must be set back to *NOT_DELIVERED*.
- After the prescription is processed, it must be set to *DELIVERED*.
- The prescription cannot stay in *DELIVERED*: the prescription must be archived in a safe place. Afterwards, those prescriptions must be set to *ARCHIVED*.

Example request for Hospital

```
POST /unified-messaging/api/v3/recipe/executor/prescriptions/BEP06YKWZE7X/status/NOT_DELIVERED HTTP/1.1
X-Pharmacy-Nihii: 11010280          <-- mandatory for pharmacy in hospital
X-Pharmacy-Holder-Ssin: 12090822841  <-- mandatory for pharmacy in hospital
X-Credential-Id: 23
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8

{
  "status": "NOT_DELIVERED"
}
```

Get the archive content

Once the prescription is processed, the prescription must be archived. The Unified Messaging / Hector cannot be used for this.

Typically this data is stored on an external data center with guarantees for no data loss.

Do not forget to update the status of the prescription as *ARCHIVED* afterwards.

WARNING

It is legally required to store archived prescription in a safe place. Do not leave the prescription in *DELIVERED*.

Example request for Hospital

```
GET /unified-messaging/api/v3/recipe/executor/prescriptions/BEP06YKWZE7X/archive-content HTTP/1.1
X-Credential-Id: 23
X-Pharmacy-Nihii: 11010280          <-- mandatory for pharmacy in hospital
X-Pharmacy-Holder-Ssin: 12090822841 <-- mandatory for pharmacy in hospital
Accept-Language: nl-BE
```

```
Authorization: Basic *****
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/octet-stream;charset=UTF-8
Content-Disposition: attachment; filename=BEP06YKWZE7X-archive.bak
Content-Length: 8309

%
%
4 0 obj
<<

...

8489
7835
%%EOF
```

Revoke a prescription

It is only possible to revoke a prescription at Recip-e if it has the status *IN_PROCESS*.
To revoke a prescription, a reason is required.

Example request for Hospital

```
POST /unified-messaging/api/v3/ recip-e/executor/prescriptions/BEP06YKWZE7X/revocation
HTTP/1.1
X-Credential-Id: 23
X-Pharmacy-Nihii: 11010280          <-- mandatory for pharmacy in hospital
X-Pharmacy-Holder-Ssin: 12090822841  <-- mandatory for pharmacy in hospital
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json

{
  "reason": "The patient recognized the medication: he already has it at home."
}
```

Example response

```
HTTP/1.1 204 No Content
Content-Type: application/json; charset=UTF-8
```

Delete a prescription

This endpoint will only delete the prescription from the local database. No actions at Recip-e are

taken.

Example request for Hospital

```
DELETE /unified-messaging/api/v3/recipient/exectuor/prescriptions/BEP06YKWZE7X HTTP/1.1
X-Credential-Id: 23
X-Pharmacy-Nihii: 11010280          <-- mandatory for pharmacy in hospital
X-Pharmacy-Holder-Ssin: 12090822841 <-- mandatory for pharmacy in hospital
Accept-Language: nl-BE
Authorization: Basic *****
```

Example response

```
HTTP/1.1 204 No Content
Content-Type: application/json; charset=UTF-8
```

Chapter 4. Consulting Vitalink

This document describes the REST endpoints for the retrieval of the medication scheme. In addition, it describes how to create a patient-consent and a therapeutic relation, which are needed to consult the medication scheme.

4.1. Authentication & error handling

Authentication against the Hector or UM services is done using basic authentication. A username and password are available on request.

Using these credentials, you can authenticate yourself as an integrator of our services. The next step is to specify which Ehealth credential to use to perform an operation. Using the existing Credential service, you can retrieve the id of a credential. This id needs to be passed along with every call by means of a custom HTTP header `X-Credential-Id`.

If for some reason we cannot identify a single Ehealth credential to use, one of the following errors will be returned.

Table 1. Credential error codes

Code	Message	Description	HTTP code
missing_credential_id	There is no credential id passed in the request.	Without a correct valid credential id, we cannot determine which credential to use.	400
no_credentials	There are no active credentials configured in this installation.	Verify that there is at least one active credential configured in the system.	409
several_matching_credentials	There are several matching credentials configured in this installation.	Verify that the correct parameters are passed so that the system can uniquely resolve to a single credential.	409
no_matching_credential	There is no credential configured in this installation that matches with the given input.	Verify that the correct parameters are passed so that the system can uniquely resolve to a single credential.	409

The error responses follow the same principles as [E-forms Cloud](#). For more details consult that documentation.

Table 2. General error codes

Code	Message	Description	HTTP code
authentication_error	Could not perform the operation because the service call is not authenticated.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	401

Code	Message	Description	HTTP code
authorization_error	Could not perform the operation because the sender is not allowed to perform this operation.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	403
internal_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

4.2. Patient consent

A patient consent is an informed consent given by the patient to share medical information with health care providers.

The consent is given once by the patient for all health care stakeholders.

The outcome of these services will be the same regardless of the selected Ehealth credential.

The choice is mostly relevant in the context of auditing, who registered the consent.

We provide an endpoint that supports three methods: retrieve the existing consent, revoke a consent and create a consent.

Table 3. Patient consent error codes

Code	Message	Description	HTTP code
invalid_request_sender	Could not complete the operation because the sender is invalid.	Please make sure valid data is provided. Do not attempt to retry the same request without changing the data.	422
invalid_patient_identifier	Could not complete the operation because the patient could not be identified (e.g. invalid combination of ssin and cardNo).	Please make sure valid data is provided. Do not attempt to retry the same request without changing the data.	422
consent_already_exists	Could not complete the operation because a consent already exists.	A patient consent is registered once for all health caregivers and remains valid until revoked.	409
no_active_consent	Could not complete the operation because there is no active consent between the concerned parties.	Please create a consent before repeating the operation.	409

Code	Message	Description	HTTP code
service_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

4.2.1. Get patient consent

Request

```
GET /rest/api/v3/patients/85041231542/patient-consent HTTP/1.1
Accept-Language: nl-BE
X-Credential-Id: 2
Authorization: Basic ....
```

The SSIN of the patient serves as the identifier in the URL. Replace the one in the example with your patient's SSIN.

Response if there is no patient consent

```
HTTP/1.1 404 Not Found
```

An empty response is returned

Response if there is a patient consent

```
HTTP/1.1 200 OK

{
  "start": "2017-11-02T23:00:00Z"
}
```

The start date when the patient consent was registered.

4.2.2. Revoke patient consent

Request

```
DELETE /rest/api/v3/patients/85041231542/patient-consent HTTP/1.1
Accept-Language: nl-BE
X-Credential-Id: 1
Authorization: Basic ....
Content-Type: application/json

{
  "supportCardNumber": "592130453362"
```

```
}
```

Response if there is no patient consent

```
HTTP/1.1 409 Conflict

{
  "code": "no_active_consent",
  "message": "Could not complete the operation because ...",
  "description": "Please create a consent before repeating the operation.",
  "uniqueId": "d69877a7cb2c451b92f82a0dc62a681d"
}
```

An error is returned with the code `no_active_consent`.

Response if there is a patient consent

```
HTTP/1.1 204 No Content
```

An empty response is returned.

4.2.3. Register patient consent

Request

```
POST /rest/api/v3/patients/85041231542/patient-consent HTTP/1.1
Accept-Language: nl-BE
X-Credential-Id: 1
Authorization: Basic ....
Content-Type: application/json

{
  "supportCardNumber": "592130453362"
}
```

Response if there is no patient consent

```
HTTP/1.1 201 Created
```

An empty response is returned.

Response if there is a patient consent

```
HTTP/1.1 409 Conflict

{
  "code": "consent_already_exists",
  "message": "Could not complete the operation because ...",
}
```

```

    "description": "Could not complete the operation ...",
    "uniqueId": "9b5a2029b3b24595ac6da58d6d0214c7"
  }

```

An error is returned with the code `consent_already_exists`.

4.3. Therapeutic link

A therapeutic link is a relation between the caregiver and the patient.

There are different types of links.

The therapeutic link exists only between the caregiver and the patient.

There can be multiple active links between the same caregiver and patient.

We provide an endpoint that supports three methods: retrieve the existing links, revoke a link (default link type) and create a link (default link type).

Table 4. Therapeutic link error codes

Code	Message	Description	HTTP code
excluded_by_patient	Could not complete the operation because the health caregiver is excluded by the patient.	Make sure that a patient consent exists before trying to call this service.	409
invalid_patient_identifier	Could not complete the operation because the patient could not be identified (e.g. invalid combination of ssin and cardNo).	Please make sure valid data is provided. Do not attempt to retry the same request without changing the data.	422
invalid_caregiver_identifier	Could not complete the operation because the health-caregiver could not be identified (e.g. invalid combination of ssin and nihii).	Please make sure valid data is provided. Do not attempt to retry the same request without changing the data.	422
therapeutic_link_exists	Could not complete the operation because a therapeutic link already exists between the concerned parties.	If an active therapeutic link of the default type exists between the patient and the health caregiver, it has to be removed first before a new one of the same type can be created.	409
no_active_therapeutic_link	Could not complete the operation because there is no active therapeutic link between the concerned parties.	Please create a therapeutic link before repeating the operation.	409

Code	Message	Description	HTTP code
service_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

4.3.1. Get therapeutic link

Request

```
GET /rest/api/v3/patients/85041231542/therapeutic-link HTTP/1.1
Accept-Language: nl-BE
X-Credential-Id: 2
Authorization: Basic ....
```

The SSIN of the patient serves as the identifier in the URL. Replace the one in the example with your patient's SSIN.

Response

```
HTTP/1.1 200 OK

{
  "caregiver" {
    "nihii": "13602140001",
    "ssin": "90062414715",
    "quality": "DOCTOR"
  },
  "patientSsin": "85041231542",
  "therapeuticLinks": [{
    "type": "gmd"
  }, {
    "type": "gpconsultation"
    "period": {
      "start": "2017-11-02T23:00:00Z",
      "end": "2018-11-02T23:00:00Z"
    }
  }
],
  "defaultType": "gpconsultation"
}
```

Therapeutic links are ordered by end-date (longest lasting first).

The start date is the date when the therapeutic link is registered and the end date depends on the link type chosen.

A gmd relation never has a start or end date.

It remains active as long as the GMD relationship is in place.

The defaultType is the relation type that will be used when you create or revoke a relation with that caregiver for that patient.

4.3.2. Revoke therapeutic link

Request

```
DELETE /rest/api/v3/patients/85041231542/therapeutic-link HTTP/1.1
Accept-Language: nl-BE
X-Credential-Id: 1
Authorization: Basic ....
Content-Type: application/json

{
  "supportCardNumber": "592130453362"
}
```

Response if there is no patient consent

```
HTTP/1.1 409 Conflict

{
  "code": "no_active_therapeutic_link",
  "message": "Could not complete the operation because there is no active
therapeutic...",
  "description": "Please create a therapeutic link before repeating the operation.",
  "uniqueId": "4be0fbec91ed429e86e495e0746d8a07"
}
```

An error is returned with the code `no_active_therapeutic_link`.

Response if there is a therapeutic link

```
HTTP/1.1 204 No Content
```

An empty response is returned.

4.3.3. Register therapeutic link

Request

```
POST /rest/api/v3/patients/85041231542/therapeutic-link HTTP/1.1
Accept-Language: nl-BE
X-Credential-Id: 1
Authorization: Basic ....
Content-Type: application/json

{
  "supportCardNumber": "592130453362",
```

```
"firstName": "Thomas",  
"lastName": "De Rooie"  
}
```

Response if there is no therapeutic link of the default type between the patient and the caregiver

HTTP/1.1 201 Created

An empty response is returned.

Response if there is a therapeutic link of the default type between the patient and the caregiver

HTTP/1.1 409 Conflict

```
{  
  "code": "therapeutic_link_exists",  
  "message": "Could not complete the operation because a therapeutic link...",  
  "description": "If an active therapeutic link of the default type exists...",  
  "uniqueId": "069b029aae8d44cc93f92be81e8220bc"  
}
```

An error is returned with the code `therapeutic_link_exists`.

4.4. Medication scheme

Request

```
GET /rest/api/v3/patients/17010151711/medication-scheme HTTP/1.1  
Accept-Language: nl-BE  
X-Credential-Id: 1  
Authorization: Basic ....
```

It is important not to specify any preferred type in the `Accept` header.

The service endpoint will return or

`application/json` in case of an error, or `application/pdf` in case of a correct response.

PDF response

```
HTTP/1.1 200 OK  
Content-Type: application/pdf
```

A PDF is returned in the response body.

Response if there is no patient consent

```
HTTP/1.1 403 Forbidden  
Content-Type: application/json
```

```
{
  "code": "no_patient_consent",
  "message": "Could not complete the operation because ...",
  "description": "Please register a patient consent before ...",
  "uniqueId": "a1b02c681a034316834be1cb588647ea"
}
```

The HTTP status code will depend on the returned code.

Table 5. Medication scheme error codes

Code	Message	Description	HTTP code
invalid_patient_identifier	Could not complete the operation because the patient could not be identified (e.g. invalid ssin or the ssin does not correspond to a real person).	Please make sure valid data is provided. Do not attempt to retry the same request without changing the data.	422
no_medication_scheme	There is no medication scheme registered for the patient matching the identifier.	This is normal behaviour. Not every citizen in Belgium has a medication scheme registered with Vitalink.	404
unauthorized	The requester is not authorized to consult the vitalink resource.	This can be caused by a variety of reasons and is usually an administrative issue. For more details, consult our support division.	403
no_therapeutic_link	Could not complete the operation because there is no active therapeutic link between the concerned parties.	Please create a therapeutic link before repeating the operation.	403
no_patient_consent	Could not complete the operation because there this patient has not given its consent.	Please register a patient consent before repeating the operation..	403
service_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

4.5. SUMEHR

Request

```
GET /rest/api/v3/patients/17010151711/sumehr HTTP/1.1
Accept-Language: nl-BE
X-Credential-Id: 1
Authorization: Basic ....
```

It is important not to specify any preferred type in the **Accept** header.
The service endpoint will return or **application/json** in case of an error, or **application/html** in case of a correct response.

PDF response

```
HTTP/1.1 200 OK
Content-Type: application/html
```

An HTML is returned in the response body.

Response if there is no patient consent

```
HTTP/1.1 403 Forbidden
Content-Type: application/json

{
  "code": "no_patient_consent",
  "message": "Could not complete the operation because ...",
  "description": "Please register a patient consent before ...",
  "uniqueId": "a1b02c681a034316834be1cb588647ea"
}
```

The HTTP status code will depend on the returned code.

Table 6. SUMEHR error codes

Code	Message	Description	HTTP code
invalid_patient_identifier	Could not complete the operation because the patient could not be identified (e.g. invalid ssin or the ssin does not correspond to a real person).	Please make sure valid data is provided. Do not attempt to retry the same request without changing the data.	422
no_sumehr	There is no sumehr registered for the patient matching the identifier.	This is normal behaviour. Not every citizen in Belgium has a sumehr in Vitalink.	404

Code	Message	Description	HTTP code
unauthorized	The requester is not authorized to consult the vitalink resource.	This can be caused by a variety of reasons and is usually an administrative issue. For more details, consult our support devision.	403
no_therapeutic_link	Could not complete the operation because there is no active therapeutic link between the concerned parties.	Please create a therapeutic link before repeating the operation.	403
no_patient_consent	Could not complete the operation because there this patient has not given its consent.	Please register a patient consent before repeating the operation..	403
service_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

4.6. Sumehr

Request the sumehr of a patient.

Request

```
GET rest/api/v3/patients/{ssin}/sumehr HTTP/1.1
Accept-Language: nl-BE
X-Credential-Id: 1
Authorization: Basic ....
```

As mentioned in the [Authentication & error handling](#) section.

It is required to provide a valid **X-Credential-Id** to request the sumehr.

The ssin shown in the request example has to be replaced by the patient's ssin the sumehr is requested for.

It's also important the integrator verifies in advance if a consent is granted between patient and requestor.

Otherwise an error message may be returned.

Response

```
HTTP/1.1 200 OK
Content-Type: text/html
```

Returns the sumehr html page in the body of the response.

Table 7. Error codes

Code	Message	Description	HTTP code
validation_errors	Could not complete the operation because the patient could not be identified (e.g. invalid ssin).	Please make sure valid data is provided. Do not attempt to retry the same request without changing the data.	422
no_patient_consent	Could not complete the operation because the patient has not given its consent.	Please register a patient consent before repeating the operation. See the Patient consent section for a more detailed description.	403

Chapter 5. Get GMF information

The global medical file (GMF) service allows the integrator to consult who is the holder of the GMF of a patient.

To be able to consult this information, a healthcare professional needs to have an active therapeutic link with the patient.

This service does not allow setting the GMF holder of a patient.

Example request

```
GET /rest/api/v3/patients/{ssin}/gmf
Accept-Language: nl-BE
Authorization: Basic ...
```

The SSIN of the patient serves as the identifier in the URL. Replace the one in the example with your patient's SSIN.

Example response

```
HTTP/1.1 200 Ok
Content-Type: application/json; charset=UTF-8

{
  "patient" : {
    "ssin" : "86120131345",
    "firstName": "patient first name",
    "lastName": "patient last name"
  },
  "holder" : {
    "ssin" : "86120131345",
    "firstName": "holder first name",
    "lastName": "holder last name",
    "nihii": "19903410004"
  },
  "startDate" : "2017-10-23T21:31:18Z",
  "endDate" : "2018-01-24T22:01:18Z"
}
```

Path	Type	Description
patient	Patient	The patient.
holder	Holder	The gmf holder.
startDate	Date	The gmf start date
endDate	Date	The gmf end date

Table 8. Patient

Path	Type	Description
ssin	String	The ssin.
firstName	String	The first name.
lastName	String	The last name.

Table 9. Holder

Path	Type	Description
ssin	String	The ssin.
firstName	String	The first name.
lastName	String	The last name.
nihii	String	The nihii.

5.1. Error flows

A global error response is possible. Examples are bad requests (validation failure of the ssin), internal server errors, etc. In these cases, the HTTP status code will reflect the nature of the error. A 422 indicates a validation issue. A 500 a technical server error.

A 403 error denotes an authorization error. An authorization error can have different causes, specified in the error.

Table 10. error codes

Code	Message	Description	HTTP code
unauthorized	You do not have enough rights to request this information. Make sure the patient has given his consent, and that you have a valid therapeutic relationship	/	403
no_active_therapeutic_link	Could not complete the operation because there is no valid therapeutic relationship	/	403
validation_errors	The ssin is not valid.	Please provide a valid ssin.	422

Code	Message	Description	HTTP code
service_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500
gateway_error	A gateway error occurred	Could not complete the operation because an error occurred in the external service.	503

Chapter 6. Querying Hector

We are adding a number of support endpoints to the Hector API to better query the installations current state. The first endpoint that is available for an integrator is the status endpoint.

Request

```
GET /rest/api HTTP/1.1
```

Response

```
HTTP/1.1 200 OK
```

```
{
  "server": "Hector",
  "version": "6.6.0-SNAPSHOT",
  "environment": "acc",
  "installationId": "8bb31c99-9523-4f8a-8190-ef7e10b125c0"
}
```

The server variable indicates if we are dealing with a *Hector* or *Unified Messaging* installation. In case of a Unified Messaging installation, the value would be **Unified Messaging**. We return the version, the environment we are running on (this affects which of our servers are called as well as external services), and also the installation ID. This value is unique for every Hector installation and forms the basis for our update services.

Based on the installation ID, Hector will query our centralized services to see if an update is available. This value can be useful to help debug problems.

Chapter 7. Consulting Credentials

This document describes the REST endpoints for the retrieval of the credentials.

There are several types of credentials in Hector / Unified Messaging.

For now, the only type returned by this API are **EHEALTH** credentials.

This type of credential is a combination of a certificate and the quality that it is configured with.

This means that there can be several credentials with the same certificate and different qualities.

7.1. Authentication & error handling

Authentication against the Hector or Unified Messaging services is done using basic authentication.

A username and password are available on request.

Using these credentials, you can authenticate yourself as an integrator of our services.

To access most of the REST endpoints, you also need a credential-id: the Ehealth credential to use to perform a operation.

This credential-id needs to be passed along with every call by means of a custom HTTP header **X-Credential-Id**.

To retrieve the credential id, you can consult the Credential REST API. This API does not require a **X-Credential-Id**.

There are no specific error codes for the credential API. If there are no credentials, an empty list is returned.

Table 11. General error codes

Code	Message	Description	HTTP code
authentication_error	Could not perform the operation because the service call is not authenticated.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	401
authorization_error	Could not perform the operation because the sender is not allowed to perform this operation.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	403
internal_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

7.2. Get credentials

To list all credentials known to a Hector or Unified Messaging installation execute the below request.

Request

```
GET /rest/api/v3/credentials HTTP/1.1
Accept-Language: nl-BE
Authorization: Basic ....
```

When there are no found credentials, the response will be an empty list.

When the password of a keystore is invalid, its status will be unavailable and the certificate will be null.

Response

```
HTTP/1.1 200 OK

[{"id": 1,
  "type": "EHEALTH",
  "status": "AVAILABLE",
  "certificate": {
    "identifier": {
      "value": "81020125549",
      "type": "SSIN"
    },
    "applicationId": null,
    "expirationDate": "2018-05-29T10:17:00Z"
  },
  "quality": {
    "name": "DOCTOR",
    "identifier": {
      "value": "14032237",
      "type": "NIHII"
    }
  },
  "healthcareActorId":
  "0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f5
  353d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7"
}, {"id": 2,
  "type": "EHEALTH",
  "status": "AVAILABLE",
  "certificate": {
    "identifier": {
      "value": "92060428727",
      "type": "SSIN"
    },
    "applicationId": null,
    "expirationDate": "2018-05-29T10:17:00Z"
  },
  "quality": {
    "name": "DOCTOR",
    "identifier": {
      "value": "14032237",
      "type": "NIHII"
    }
  },
  "healthcareActorId":
  "0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f5
  353d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7"
}]
```

```

    "applicationId": null,
    "expirationDate": "2018-04-18T11:16:00Z"
  },
  "quality": {
    "name": "DIETICIAN",
    "identifier": {
      "value": "56222855",
      "type": "NIHII"
    }
  },
  "healthcareActorId":
"3ec15c3ca4962267dff303855ede39190043db1ef20f403c7ec9d1b4b2c26c8048e9ce42b9fc4f2083e38
2d8b2aaaa4475a820544906d16843f36ff2fc6dfeff"
}, {
  "id": 3,
  "status": "AVAILABLE",
  "certificate": {
    "identifier": {
      "type": "NIHII",
      "value": "81160492"
    },
    "applicationId": "",
    "expirationDate": "2018-07-03T11:09:51.000+0000"
  },
  "quality": {
    "name": "LABO",
    "identifier": {
      "type": "NIHII",
      "value": "81160492000"
    }
  },
  "healthcareActorId":
"5f38852b57391b0c6ac4618c2cf73b4c19d2671d82b45ef4908bb3f1bd3591bbb332947ead1dd5fc105c1
2c61278433b22651968192e73588bf562cfce8803c7",
  "type": "EHEALTH"
}, {
  "id": 4,
  "status": "AVAILABLE",
  "certificate": {
    "identifier": {
      "type": "NIHII",
      "value": "71030130"
    },
    "applicationId": "",
    "expirationDate": "2018-05-24T13:45:03.000+0000"
  },
  "quality": {
    "name": "HOSPITAL",
    "identifier": {
      "type": "NIHII",
      "value": "71030130000"
    }
  }
}

```

```

    },
    "healthcareActorId":
    "2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d69
    2e43cd65ae09817c87d020d66028e69610770a0a7d7",
    "type": "EHEALTH"
  }]

```

7.3. Get credentials with filtering

On all fields, except for `expirationDate`, there can be filtered.

To add a filter, simply add a query parameter containing the field and the value.

Request with filtering

```

GET
/rest/api/v3/credentials?certificate.identifier.value=12132237&certificate.identifier.
type=SSIN HTTP/1.1
Accept-Language: nl-BE
Authorization: Basic ....

```

Status has 2 possible values:

AVAILABLE and UNAVAILABLE. Both can be added.

Filtering the status

```

GET /rest/api/v3/credentials?status=AVAILABLE,status=UNAVAILABLE HTTP/1.1
Accept-Language: nl-BE
Authorization: Basic ....

```

The response schema is the same as *get credentials* without filtering.

7.4. Get credentials for organisations

Use filtering to get credentials for a specific organisation, for instance a hospital or a laboratory.

Specify the *health care quality* by using query parameter `quality.name=HOSPITAL` or `quality.name=LABO`.

Filtering on health care quality for laboratories

```

GET /rest/api/v3/credentials?quality.name=LABO HTTP/1.1
Accept-Language: nl-BE
Authorization: Basic ....

```

Find a specific health care organisation by filtering with query parameters `quality.identifier.type` and `quality.identifier.value`.

Filtering on health care quality identifier for a hospital

```
GET  
/rest/api/v3/credentials?quality.identifier.type=NIHII&quality.identifier.value=710301  
30000 HTTP/1.1  
Accept-Language: nl-BE  
Authorization: Basic ....
```

Additionally you can specify the *applicationId* of your health care organisation with query parameter **certificate.applicationId**.

Filtering on applicationId for 'EHEALTHBOX'

```
GET /rest/api/v3/credentials?certificate.applicationId=EHEALTHBOX HTTP/1.1  
Accept-Language: nl-BE  
Authorization: Basic ....
```

Chapter 8. Get insurability information

The MCN insurability service allows the integrator to consult key insurability numbers.

Example request

```
GET /rest/api/v3/patients/{ssin}/insurability
Accept-Language: nl-BE
X-Credential-Id: 1
Authorization: Basic ...
```

The SSIN of the patient serves as the identifier in the URL.
Replace the one in the example with your patient's SSIN.

As seen in the example above.

It is also possible to explicitly provide an Ehealth credential id using custom HTTP header **X-Credential-Id**.

Used to authenticate the MCN request on behalf of.

To retrieve a credential id, you can consult the Credential REST API.

Described in section [5.4 Get credentials for organisations](#) for more information.

It is also possible to omit the credential id.

Aldo this is strongly discouraged.

The application will try to resolve a credential by itself based upon configured credentials and authenticated user.

But always keep in mind this is a best effort.

Example response

```
HTTP/1.1 200 Ok
Content-Type: application/json; charset=UTF-8

{
  "registrationNumber": "2018987014365",
  "mutuality": {
    "code": "304",
    "name": "De Voorzorg Socialistische Mutualiteit van de Provincie Antwerpen"
  },
  "ct1": "110",
  "ct2": "110",
  "payment": {
    "byInsurabilityOrganisation": false,
    "specialSocialCategory": false
  }
}
```

8.1. Consult for a specific date

If no date is specified, today's insurability information is retrieved.

You can specify a query parameter `queryForDate` that accepts an ISO 8601 date.

Example request

```
GET /rest/api/v3/patients/{ssin}/insurability?queryForDate=2021-01-15
Accept-Language: nl-BE
Authorization: Basic ...
```

8.2. Error flows

A global error response is possible.

Examples are bad requests (validation failure of the `ssin`), internal server errors, etc.

In these cases, the HTTP status code will reflect the nature of the error.

A 422 indicates a validation issue.

A 500 a technical server error.

A 403 error denotes an authorization error.

An authorization error can have different causes, specified in the error.

Table 12. error codes

Code	Message	Description	HTTP code
unauthorized	You do not have enough rights to request this information.	/	403
validation_errors	The ssin is not valid.	Please provide a valid ssin.	422
service_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500
gateway_error	A gateway error occurred	Could not complete the operation because an error occurred in the external service.	503

Chapter 9. Helena

Helena provides an online platform in which messages or medical documents can be communicated from a medical professional/organization to a patient.

9.1. Authentication & error handling

Authentication against the Hector or UM services is done using basic authentication.

A username and password are available on request.

Using these credentials, you can authenticate yourself as an integrator of our services.

You also need a credential-id: the Ehealth credential to use to perform a operation.

This credential-id needs to be passed along with every call by means of a custom HTTP header **X-Credential-Id**.

To retrieve the credential id, you can consult the Credential REST API. Refer to section [5.4 Get credentials for organisations](#) for more information.

Table 13. General error codes

Code	Message	Description	HTTP code
authentication_error	Could not perform the operation because the service call is not authenticated.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	401
authorization_error	Could not perform the operation because the sender is not allowed to perform this operation.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	403
validation_error	Your input contains errors.	Description of the field(s) in error.	422
internal_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

9.2. Upload a laboratory result

A PDF representing a laboratory result can be uploaded to Helena using the example request below. The PDF must be BASE64 encoded with padding enabled.

You must specify the SSIN of the patient as "patientSsin" and specify NIHII number of the doctor who requested this laboratory result as "onBehalfOfNihii".

Table 14. Request body details

Field	Description
title	Title of the document. Max length 255 characters.
description	Explanatory notes for the document. Optional, a free-text message.
type	Type of Helena document being sent. This is required and only one of following is allowed: ATTESTATION , DOCUMENT , LAB_RESULT , REFERRAL_LETTER , REPORT .
patientSsin	Ssin of the patient receiving the Helena document. The patient must have an account for Helena.
patientEmail	Email of the patient receiving the Helena document. Optional but if specified in the request, this has to be a valid email address. Otherwise, the upload will fail.
patientLanguage	Language of the patient receiving the Helena document. Optional but if specified in the request, this has to be one of NL, FR or EN. Otherwise, the upload will fail.
onBehalfOfNihii	NIHII number of the doctor who requested this laboratory result.
content	The document which must be represented as a padded base64 encoded string.
mediaType	The media-type of the document, this will be validated against the kmehr CDMEDIATYPEvalues.

Request

```
POST /rest/api/v3/helena/document HTTP/1.1
Accept-Language: nl-BE
Authorization: Basic ....
Content-Type: application/json
X-Credential-Id: 1

{
  "title": "Resultaten Bloedonderzoek",
  "description": "De resultaten van het bloedonderzoek naar aanleiding van uw laatste consult.",
  "type": "LAB_RESULT",
  "patientSsin": "89121824700",
  "patientEmail": "patient@email.be",
  "patientLanguage": "NL",
```

```
"onBehalfOfNihii": "14032237730",
"content": "BASE64_ENCODED_PDF_CONTENT",
"mediaType": "application/pdf"
}
```

Response

```
HTTP/1.1 204 No Content
Content-Type: application/json; charset=UTF-8
```

If the request to Helena does not succeed a HTTP error '400 Bad Request' is returned.

For example when a patient is not on Helena.

When the request to Helena fails a HTTP error '502 Bad Gateway' is returned.

Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=UTF-8
```

```
{
  "code": "not_found",
  "message": "Patient [SSIN=*****4700] not known by Helena.",
  "description": "Verify that the identifier is valid and is in use. If the
  identifier was fetched a while back, or is constructed on your side, it most likely
  became stale. Query the corresponding list endpoint to retrieve all identifiers for
  this resource type. If you have just acquired this identifier from our system, contact
  our support services.",
  "uniqueId": "h8TrKj8M080Evrbdncv"
}
```

9.3. Retrieve a patient registration link

The patient has to be registered before being able to access Helena.

A new link to the registration form can be retrieved via following POST request.

No specific data is required and the response will always contain a unique URL allowing for a 'new' patient registration to be processed.

Via Hector users can always retrieve a patient registration link.

UM is more strict and only allows users granted user, department administrator or invoicing clerk rights.

Also Hector and UM require the application to be configured with a valid Lab or Hospital eHealth Credential.

Otherwise no registration link will be able to be retrieved resulting in error.

Request

```
POST /rest/api/v3/helena/registration-link HTTP/1.1
Accept-Language: nl-BE
```

Authorization: Basic

Response

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8

"https://services.healthconnect.be/helena/integrated/registration/patient?type=WEB&token=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJIT1NQSVRBTERFViIsImF1ZCI6WyJodHRwczovL2hlbGVuYS5jYXJlL3dlYiJdLCJpZFR5cGU0Ij0SUhJSSiIsImVoZWZsdGhRdWFSaXR5IjoieSE9TUElUQUwiLCJhbXCI6Ij0JTVFmLCJzY29wZSI6WyJyb2x1Om9yZ2FuaXNhdGlvbiIsImZ1bmN0aW9uOnBhdGllbnQ6Y3JlYXRlIiwiaWF0Ij0iMTAzMDEzMDEwMCIsImV4cCI6MTU3OTE4ODc4NSwianRpIjoieY2M5MDhjN2Q2YzQxNi00YzFhLTkyNjEtMGU0YzdkYjI3N2E2IiwiaWY2xpZW50X2lkIjoiaGVsZW5hIiwidXNlcm5hbWUiOiJIT1NQSVRBTERFViJ9.B_Rf2kXLN4btySqJKJz8KdPEwS6Uurn40sJm0iw-NHEXzg8Pn1FFF0XEt8DphtCudGc52SKvCjB_pFG41Hw0dcRg7AVylvcpoSDtYN9byq7-7ixUC8GbC4-177kaUJt_eX0IsL8n0_WL9Da-Ui2swl8gZS3tSnk0Fj5ewpCYLPuTKTvL08G0X6aXl9Gcdm119VkcOu_Sy2F3w6LYVT_ZaMvAFFVHVAXFaT9Uio3Rraw2yTXa5HFCFY8we786511PPUxw1TjrUo-NlhxWQva-qh5MQ41h2PZyJo0R374JTC-L4yoHnwHAgmmT3iJZ9A6rR3mwoN7xvEBLzCUZodvPCQ&idpId&firstName&lastName&email&phoneNumber&language"
```

Table 15. Error codes

Code	Message	Description	HTTP code
helena_credential_error	No valid credential configured for authentication with Helena.	Please add a valid credential for authentication with Helena.	409

9.4. Retrieve a patient's application link

Returns a link to open the patient's portal management screen.

The management screens allows to start a video consult, view published documents and to update patient registration details.

To retrieve a link the integrator has to provide a valid 'X-Credential-Id' header of a configured ehealth credential.

Credentials have to be of quality 'DOCTOR', 'HOSPITAL' or 'LABO' otherwise subsequent calls to the system will be refused.

The request's document body has to contain a JSON Object with mandatory 'patientSsin'. Invalid SSIN or NIHII numbers will be refused.

Table 16. Request body details

Field	Description
patientSsin	Required patient's SSIN number.

Field	Description
patientDetails	Optional patient registration details. Which when provide will be passed on to the Helena API.
onBehalfOfNihii	NIHII number of the caregiver requesting portal management. Can only be provided when using an organization eHealth Credential with qualities 'HOSPITAL' or 'LABO'.

Table 17. Request Patient details

Field	Description
firstName	E.g. John.
lastName	E.g. Doe.
language	'NL' or 'FR', used to determine the language of the patient's invite email/SMS.
cellPhoneNumber	E.g. +32474567890.
email	E.g. john.doe@healthconnect.be .
birthDate	E.g. 2000-05-29.

When a request is made using a professional eHealth Credential with 'DOCTOR' quality. The 'onBehalfOfNihii' should never be passed. As well as the caregiver object.

In case of an organization, containing one of the allowed 'HOSPITAL' or 'LABO' qualities. The 'onBehalfOfNihii' is required.

Patient details are always optional.

Request using a 'DOCTOR' eHealth Credential

```
POST /unified-messaging//api/v3/helena/application-link HTTP/1.1
Host: http://localhost:8081/unified-messaging
Accept-Language: nl-BE
Accept: text/plain
X-Credential-Id: 4
Content-Type: application/json
Authorization: Basic ....
```

```
{
  "patientSsin": "85020125559",
  "patientDetails":
  {
    "firstName": "",
    "lastName": "",
    "language": "",
    "cellPhoneNumber": "",
```

```

        "email": "",
        "birthDate": ""
    }
}

```

Request using a 'HOSPITAL' OR 'LABO' eHealth Credential

```

POST /unified-messaging/api/v3/helena/application-link HTTP/1.1
Host: http://localhost:8081/unified-messaging
Accept-Language: nl-BE
Accept: text/plain
X-Credential-Id: 7
Content-Type: application/json
Authorization: Basic ....

```

```

{
  "patientSsin": "85020125559",
  "patientDetails":
  {
    "firstName": "",
    "lastName": "",
    "language": "",
    "cellPhoneNumber": "",
    "email": "",
    "birthDate": ""
  },
  "onBehalfOfNihii": "56244855601"
}

```

Response

```

HTTP/1.1 200 OK
Content-Type: text/plain;charset=UTF-8

"https://example.helena.care/caregiver?token=abc.xyz-test&lng=nl"

```

Table 18. Error codes

Code	Message	Description	HTTP code
invalid_input	Unable to resolve a caregiver for given NIHII number.	The caregivers details could not be retrieved, make sure the NIHII number is still valid.	400
helena_credential_error	Missing valid Helena credential, please configure before retrying.	Please provide a valid Helena credential via the 'X-Credential-Id' header.	409

9.5. Retrieve a patient's status

Returns the current status of a patient in the Helena application even when not yet registered.

It's required to provide a registered ehealth credential id via the 'X-Credential-Id' header.

Which will be used to authenticate against the Helena API.

Verify that the credential is configured.

Otherwise the request will be declined.

Also provide a valid SSIN in the path to retrieve the current state for.

Invalid SSIN numbers will be rejected.

As a result one of following patient statuses should be returned:

- REGISTERED_WITHOUT_ACCOUNT (before 'REGISTERED_ADULT')
- REGISTERED_WITH_ACCOUNT (before 'REGISTERED_CHILD')
- NOT_REGISTERED
- DELETED
- BLOCKED_CAREGIVER (before 'EXCLUSION')
- NOT_FOUND

Note: /status endpoint deprecated and replaced by /registrationstatus

▼ *GET /unified-messaging//api/v3/helena/patient/<patientSsin>/status*

Deprecated request

```
GET /unified-messaging//api/v3/helena/patient/<patientSsin>/status HTTP/1.1
Accept-Language: nl-BE
Accept: text/plain
X-Credential-Id: 7
Authorization: Basic ....
```

Request

```
GET /unified-messaging//api/v3/helena/patient/<patientSsin>/registrationstatus
HTTP/1.1
Accept-Language: nl-BE
Accept: text/plain
X-Credential-Id: 7
Authorization: Basic ....
```

Request

```
GET /unified-messaging//api/v3/helena/patient/<patientSsin>/registrationstatus
HTTP/1.1
Accept-Language: nl-BE
Accept: text/plain
X-Credential-Id: 7
```

Authorization: Basic

Response

HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8

"NOT_REGISTERED"

Table 19. Error codes

Code	Message	Description	HTTP code
invalid_input	Invalid patient SSIN.	Please provide a valid SSIN number.	400
authentication_error	No valid credential found for given 'X-Credential-Id'.	Please provide a valid and existing credential id.	401
not_found	No patient found for given SSIN in Helena.	Please register the patient first before continuing.	404

Chapter 10. wit-gele kruis (WGK)

WGK allows hospitals to consult patient nursing files.

'Mijn Thuisverpleging' (mijnTVP) acts as a gateway that allows the nursing file of a patient to be consulted at mijnWGK or one of the similar partner applications.

In order to make this possible two services are provided.

The first, allowing to verify if a patient is known by any of their applications.

The second, providing a temporarily link which allows to open 'mijnTVP' in a browser.

Allowing web access to the available nursing files of a patient.

10.1. Authentication & error handling

10.1.1. Authentication

These services are only accessible for integrators using UM. To access them, basic authentication by username and password is required.

Credentials are available upon request.

Basic Authorization example

```
Authorization: Basic dGVzdC11c2VyOnRlc3QtcGFzc3dvcmQ=
```

10.1.2. Hospital Credential

UM needs a valid eHealth Credential of quality 'HOSPITAL'.

- in the first place, a valid hospital ehealth certificate must be configured in UM
- secondly, the identifier bound to this hospital ehealth certificate must be provided using the custom HTTP header X-Credential-Id

Refer to section [5.4 Get credentials for organisations](#) for more information concerning credentials.

X-Credential-Id header example

```
X-Credential-Id: 3
```

10.1.3. Traceability

For traceability every request can be handed an id, identifying the request using HTTP header Correlation-Id.

It is send back in the response message.

Correlation-Id header example

```
Correlation-Id: REQ93402532464322
```

10.1.4. Errors

Table 20. General error codes

Code	Message	Description	HTTP code
invalid_input	Invalid input field. Please verify the documentation.	Description of the field(s) in error.	400
invalid_credential	Invalid eHealth Credential to obtain an STS token. Should be of quality HOSPITAL.	Verify if a valid 'X-Credential-Id' is provided.	400
authentication_error	Could not perform the operation because the service call is not authenticated.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	401
authorization_error	Could not perform the operation because the sender is not allowed to perform this operation.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	403
validation_error	Your input contains errors.	Description of the field(s) in error.	422
internal_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

10.2. Therapeutic relationship

WGK performs no check on the therapeutic relationship between patient and applicant.

IMPORTANT

It is up to the integrator to verify whether the applicant has access to the requested patient or not!

10.3. General request data-types

Table 21. Role of the physical person who is sending the request.

Value	Description
PERSPHYSICIAN	Persphysician
PERSNURSE	Nurse
PERSSOCIALWORKER	Social worker
PERSPHARMACIST	Pharmacist

10.4. General response data-types

Table 22. Data available types which indicates if data is available for the requested patient in mijnTVP.

Value	Description
DATA_AVAILABLE	Patient is known in mijnTVP.
NO_DATA_AVAILABLE	Patient is not known in mijnTVP.

10.5. Verify data availability

Allows to verify if data is available in the mijnTVP application for a given patient.

Table 23. Request body details

Field	Required	Description
careProviderFirstName	no	First name of the physical person who is sending the request.
careProviderLastName	no	Last name of the physical person who is sending the request.
careProviderRole	yes	Role of the physical person who is sending the request. Possible values are provided in the Role types table.
careProviderSsin	Required when no careProviderNihii provided.	SSIN of the physical person who is sending the request.
careProviderNihii	Required when no careProviderSsin provided.	Nihii-number of the physical person who is sending the request.
patientSsin	yes	Identification of the sociale security (INSZ) of the patient which is being verified if data is available.

Request example

```
GET /unified-messaging/api/v3/mijnwgk/verify-data-availability HTTP/1.1
Accept-Language: nl-BE
Authorization: Basic ....
Content-Type: application/json
X-Credential-Id: 1
```

```
{
  "careProviderFirstName": "John",
  "careProviderLastName": "Doe",
  "careProviderRole": "PERSPHYSICIAN",
  "careProviderNihii": "14032237730",
  "patientSsin": "00010100105"
}
```

Table 24. Response body details

Field	Required	Description
status	yes	Message wich indicates if the request is successfully handled.
correlationId	yes	Unique ID assigned by the WGK server to enable traceability.
ssin	no	Identification of the sociale security (INSZ) of the patient which is being verified if data is available.
dataAvailable	no	Parameter which indicates if data is available for the requested patient in mijnTVP. Possible values are provide in the Data available types table.

Response example

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
  "status": {
    "code": "200",
    "message": "Success"
  },
  "requestId": "REQ123",
  "correlationId": "0ff2a2cd-df7e-4f20-9f84-ae166b48924e",
  "ssin": "00010100105",
  "dataAvailable": "DATA_AVAILABLE"
}
```

10.6. Get application link

Request an application link to access the nursing files for given patient.

A link is only returned if the patient is available in the mijnTVP application.

The returned link will be accessable for 30 minutes and will expire thereafter.

Table 25. Request body details

Field	Required	Description
careProvider FirstName	no	First name of the physical person who is sending the request.

Field	Required	Description
careProvider LastName	no	Last name of the physical person who is sending the request.
careProvider Role	yes	Role of the physical person who is sending the request. Possible values are provided in the Role types table.
careProvider Ssin	Required when no careProviderNihii provided. Or role equals PERSSOCIALWORKER!	SSIN of the physical person who is sending the request.
careProvider Nihii	Required when no careProviderSsin provided.	Nihii-number of the physical person who is sending the request.
patientSsin	yes	Identification of the sociale security (INSZ) of the patient which is being verified if data is available.

Request example

```
GET /unified-messaging//api/v3/mijnwvk/application-link HTTP/1.1
Accept-Language: nl-BE
Authorization: Basic ....
Content-Type: application/json
X-Credential-Id: 1

{
  "careProviderFirstName": "John",
  "careProviderLastName": "Doe",
  "careProviderRole": "PERSPHYSICIAN",
  "careProviderSsin": "90010100123",
  "patientSsin": "00010100105"
}
```

Table 26. Response body details

Field	Required	Description
status	yes	Message wich indicates if the request is successfully handled.
correlationId	yes	Unique ID assigned by the WGK server to enable traceability.
ssin	no	Identification of the sociale security (INSZ) of the patient which is being verified if data is available.
dataAvailable	no	Parameter which indicates if data is available for the requested patient in mijnTVP. Possible values are provide in the Data available types table.
link	no	Direct link which can be used in a browser to gain access to the file of a patient in mijnTVP.

Response example

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
  "status": {
    "code": "200",
    "message": "Success"
  },
  "requestId": "REQ123",
  "correlationId": "0ff2a2cd-df7e-4f20-9f84-ae166b48924e",
  "ssin": "00010100105",
  "dataAvailable": "DATA_AVAILABLE",
  "link": "https://example.mijntvp.be/#/select-partner?token=xYz_token-1234"
}
```

Chapter 11. Context Integration (UM Only)

Allows an integrator to initialize and visualize the Mybox application in a given authenticated state.

This enables the integrator to seamlessly provide functionalities out-of-the-box towards their end-users which are already present in Mybox.

The aim is to reduce cost of development and eliminate the need to (re-)implement a user interface for each specific integrator.

We provide this integration using the OTP (One Time Password) mechanism.

Example use case: Visualize the messaging inbox for user John Doe filtered on patient having a specified SSIN.

This will result in Mybox being opened as user John Doe on the messaging inbox screen with the messages filtered for the given patient.

11.1. Authentication & error handling

Authentication against the UM services is done using basic authentication.

A username and password are available on request.

Using these credentials, you can authenticate yourself as an integrator of our services.

Table 27. General error codes

Code	Message	Description	HTTP code
authentication_error	Could not perform the operation because the service call is not authenticated.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	401
authorization_error	Could not perform the operation because the sender is not allowed to perform this operation.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	403
validation_error	Your input contains errors.	Description of the field(s) in error.	422
internal_error	An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500

11.2. Create an OTP session

Important: the OTP session will be valid for 15seconds, after that the resulting OTP code will no

longer be useable.

You must therefore make sure that when you create an OTP session you consume it within this time-frame.

To create an OTP session you must provide the username of the user you want to work with in Mybox and a context.

The context contains a target and additional parameters.

The target determines which page in Mybox you open, the other parameters define the behavior of the requested page.

These parameters can differ depending on the target you wish to use.

Currently we support 1 target: 'MESSAGING/INBOX' and for this target we support 1 parameter 'patient' which is an SSIN. (See example use case above)

The response will contain your OTP code, which you can later use to consume your OTP session.

Table 28. Request body details

Field	Description
user.username	The username of the user. As an administrator consult the users in mybox to see which usernames you configured.
context.target	The identifier of the page you which to present to your user.
context.patient	The SSIN of the patient you which the messages should be automatically filtered on.

Request

Please take note that we masked the used SSIN for patient, in reality provide a complete and valid SSIN as String

```
POST /unified-messaging/api/v3/mybox-sessions? HTTP/1.1
Content-Type: application/json
Authorization: Basic ...

{
  "user": {
    "username": "john.doe"
  },
  "context": {
    "target": "MESSAGING/INBOX",
    "patient": "*****5559"
  }
}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
{
  "code": "6dJB7QshojY_-ZdoUUqsn2HmlVZfBB3P"
}
```

If you provide any errors on your context, '400 Bad Request' is returned.
For example when a patient SSIN is not valid.

Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=UTF-8

{
  "code": "validation_errors",
  "message": "Check your input.",
  "uniqueId": "cSltzs2UDTUPgo9JF8wS",
  "validationErrors": [
    {
      "field": "context.patient",
      "value": "*****5558",
      "message": "Invalid SSIN."
    }
  ]
}
```

If you provided an unknown username, '400 Bad Request' is returned.

Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=UTF-8
{
  "code": "invalid_input",
  "message": "Could not authorize user with username [jane.doe].",
  "description": "Consult the documentation on how to invoke this endpoint.",
  "uniqueId": "LyXrSxgeNFOUPVX3F5tS"
}
```

11.3. Consume the OTP session

The OTP session, which you created using the documentation above, can be opened in a browser or an equivalent.

No HTTP Authorization header is required! Your OTP code is considered to be your authentication.

As with any other OTP implementation the code can only be consumed **once and only once**.

Any attempt to request a non-existing OTP code will result in an error page to be shown.

Request

```
GET /unified-messaging/api/v3/mybox-sessions/6dJB7QshojY_-ZdoUUqsn2Hm1VZfBB3P HTTP/1.1
```

Response (Success)

```
HTTP/1.1 302 Moved Permanently
Set-Cookie: JSESSIONID=568DD364571CA9B6E8A4DAAD1CA7B97F; Path=/; HttpOnly
Location: https://yourhostname/mybox/index.html
Content-Length: 0
```

Response (Failure)

```
HTTP/1.1 302 Moved Permanently
Set-Cookie: JSESSIONID=568DD364571CA9B6E8A4DAAD1CA7B97F; Path=/; HttpOnly
Location: https://yourhostname/mybox/error.html
Content-Length: 0
```

Response (After following the redirect)

```
HTTP/1.1 200 OK
Content-Type: text/html

<html>
... Mybox...
</html>
```

Chapter 12. Security

12.1. Legacy Mode

Both Unified Messaging and Hector are shipped with a Legacy Mode.

Legacy Mode is a master switch which enables or disables the ability to make use of less secure APIs.

To ensure backward compatibility, both Unified Messaging and Hector are currently (v7) shipped with Legacy Mode turned on by default.

This means that all less secure APIs are by default accessible.

CAUTION

Starting from version 8 Legacy Mode will be turned **OFF** by default.
Resulting in less secure endpoints to no longer be accessible!

12.1.1. Configuration

The only way to currently change the Legacy Mode is by manually executing the following SQL statement:

```
UPDATE SETTING SET SETTING_VALUE = 'true' where SETTING_FIELD = 'LEGACY_MODE_ENABLED';  
COMMIT;
```

The SETTING_VALUE column accepts the following values: 'true' and 'false'.

If you require guidance, please contact our support service.

12.1.2. Hector

When Legacy Mode is turned **OFF** Hector will enforce only listening on the loopback interface. Meaning that only connections for the same physical machine will be possible.

If you decide to turn Legacy Mode **ON** we strongly advise you to configure `server.address` in `hector.properties`.

The value `0.0.0.0` is strongly discouraged for `server.address` but **is** allowed.

12.1.3. Unified Messaging

When Legacy Mode is turned **OFF** Unified Messaging will enforce the use of an SSL certificate. Meaning that only encrypted communication will be accepted.

12.1.4. Disabled APIs

When Legacy Mode is turned **OFF** the following endpoints will **NOT** be accessible:

- `/services/*`

- /unified-messaging/services/*
- /*/api/v1/credentials/**
- /*/api/v1/em/**
- /*/api/v1/users/**
- /*/api/v1/oauth2-token/**
- /*/api/v3/installation/**
- /*/api/v3/status/connectivity
- /*/api/mybox/authenticate

Chapter 13. V4 Insurability

13.1. Get Insurability (member data)

MyCaretet exposes the patient's insurability over a given period of time in a structured way.

There are two options of requesting a patient's insurability: via SSIN or mutuality info. Only one of these two options can be provided.

In any case, the **Default-Ehealth-Credential-Id** header should be provided. This is the EhealthActorId associated with the ehealth credential of the user requesting the insurability. Without this you will not be authorized to call the v4 endpoints!

13.1.1. Request by Patient ssin

```
GET /api/v4/patients/mcn-insurability HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Patient-ssin: 72081061175
```

13.1.2. Request by mutuality info

The following pattern must be respected: <registrationNumber>@<mutualityId>

```
GET /api/v4/patients/mcn-insurability HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Mutuality-info: 2018987010285@304
```

The request for insurability information can be filtered further by means of query parameters. The following table gives an overview of the parameters as well as their default value.

Table 29. Insurability filter fields

Field	Default value
startDate	Today
endDate	Today
requestType	INFORMATION
contactType	OTHER

You can find an example of a fully configured filter hereunder.

```
GET /api/v4/patients/mcn-
```

```
insurability?requestType=INFORMATION&contactType=OTHER&startDate=2021-02-09&endDate=2021-02-09 HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Patient-ssin: 72081061175
```

13.1.3. Example response

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: ...

{
  "patient": {
    "ssin": 93051822361,
    "firstName": "John",
    "lastName": "Doe",
    "birthDate": "2016-08-29",
    "gender": "MALE",
    "deceasedOn": "2016-08-29"
  },
  "payment": {
    "byInsurabilityOrganisation": false,
    "specialSocialCategory": false,
    "maxInvoicedForYears": [
      "string"
    ]
  },
  "generalSituation": {
    "transfers": [
      {
        "mutualityId": "900",
        "date": "2016-08-29",
        "direction": "TO_OTHER_IO"
      }
    ],
    "eventType": "CHANGE_DURING"
  },
  "hospitalization": {
    "hospital": "string",
    "admissionDate": "2016-08-29",
    "admissionService": "string"
  },
  "medicalHouses": [
    {
      "medicalHouse": {
        "nihii": "string",
        "label": "string",
```

```

      "contactId":
"0492d47a23d5ad11e94107cfa88e896f768d8e12ola6f2f9c3b7fed8baacc3304358e8314397c2d9cb9f5
353d52f9b7496e9ae2cf94e18edd47c4925e69ac8b",
      "name": "string"
    },
    "dateTimePeriod": {
      "startDate": "2016-01-01T00:12:33.001Z",
      "endDate": "2020-12-31T09:12:33.001Z"
    },
    "nurseCare": true,
    "medicalCare": true,
    "kineCare": true
  }
],
"insurabilityRecords": [
  {
    "registrationNumber": "string",
    "mutualityId": "900",
    "datePeriod": {
      "startDate": "2016-08-29",
      "endDate": "2016-08-29"
    },
    "ct1": "string",
    "ct2": "string",
    "paymentApproval": "string",
    "communicationDate": "2016-08-29"
  }
],
"carePaths": [
  {
    "carePathType": "DIABETES",
    "generalist": {
      "nihii": "string",
      "label": "string",
      "contactId":
"0492d47a23d5ad11e94107cfa88e896f768d8e12ola6f2f9c3b7fed8baacc3304358e8314397c2d9cb9f5
353d52f9b7496e9ae2cf94e18edd47c4925e69ac8b",
      "firstName": "string",
      "lastName": "string"
    },
    "medicalHouse": {
      "nihii": "string",
      "label": "string",
      "contactId":
"0492d47a23d5ad11e94107cfa88e896f768d8e12ola6f2f9c3b7fed8baacc3304358e8314397c2d9cb9f5
353d52f9b7496e9ae2cf94e18edd47c4925e69ac8b",
      "name": "string"
    },
    "specialist": {
      "nihii": "string",
      "label": "string",

```

```

      "contactId":
"0492d47a23d5ad11e94107cfa88e896f768d8e1201a6f2f9c3b7fed8baacc3304358e8314397c2d9cb9f5
353d52f9b7496e9ae2cf94e18edd47c4925e69ac8b",
      "firstName": "string",
      "lastName": "string"
    },
    "rightPeriod": {
      "startDate": "2016-08-29",
      "endDate": "2016-08-29"
    },
    "contractEnd": "2016-08-29"
  }
],
"chronicCondition": {
  "years": [
    "string"
  ]
},
"referencePharmacy": {
  "pharmacy": {
    "nihii": "string",
    "label": "string",
    "contactId":
"0492d47a23d5ad11e94107cfa88e896f768d8e1201a6f2f9c3b7fed8baacc3304358e8314397c2d9cb9f5
353d52f9b7496e9ae2cf94e18edd47c4925e69ac8b",
    "name": "string"
  },
  "startDate": "2016-08-29"
},
"mediPrima": {
  "carmedNumber": "string",
  "versionNumber": 0,
  "names": {
    "additionalProp1": "string",
    "additionalProp2": "string",
    "additionalProp3": "string"
  },
  "validityPeriod": {
    "startDate": "2016-08-29",
    "endDate": "2016-08-29"
  },
  "cbeNumber": "string",
  "thirdPayer": true,
  "agreementNumber": "string",
  "percentageGouvernement": 0,
  "percentagePatient": 0,
  "ambulantGov": 0,
  "ambulantPatient": 0,
  "hospitalizationGov": 0,
  "hospitalizationPatient": 0,
  "medicalUrgencyInd": true
}

```

```

    },
    "specialStatus": {
      "code": "NoTMAndPaybyIOM"
    }
  },
  "errorFacets": [
    "CARE_PATH"
  ]
}

```

13.2. Get Global Medical File Record

MyCarenet exposes the patient's global medical file record facet over a given period of time in a structured way.

The GMD holder will be returned if a GMD is known on the first day of the requested period.

If the GMD holder is known it will represent either the family doctor or a medical house.

A patient has the option of registering in a medical house that has concluded a lump-sum payment agreement with his or her health insurance fund.

In this case, the patient's health will globally be taken into account in the medical house.

There are two options of requesting a patient's global medical file record: via SSIN or mutuality info.

Only one of these two options can be provided.

In any case, the **Default-Ehealth-Credential-Id** header should be provided.

This is the EhealthActorId associated with the ehealth credential of the user requesting the global medical file record.

Without this you will not be authorized to call the v4 endpoints!

13.2.1. Request by Patient ssin

```

GET /api/v4/patients/mcn-global-medical-file-record HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Patient-ssin: 72081061175

```

13.2.2. Request by mutuality info

The following pattern must be respected: <registrationNumber>@<mutualityId>

```

GET /api/v4/patients/mcn-global-medical-file-record HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Mutuality-info: 2018987010285@304

```

The request for insurability information can be filtered further by means of query parameters. The following table gives an overview of the parameters as well as their default value.

Table 30. Insurability filter fields

Field	Default value
startDate	Today
endDate	Today

You can find an example of a fully configured filter hereunder.

```
GET /api/v4/patients/mcn-global-medical-file-record?startDate=2021-02-09&endDate=2021-02-09 HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Patient-ssin: 72081061175
```

13.2.3. Example response when a doctor is the global medical file record holder

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: ...

{
  "doctor": {
    "nihii": "14032237",
    "contactId":
"0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f5
353d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7",
    "label": "Arts Carl Molemans"
  },
  "medicalHouse": null
}
```

13.2.4. Example response when a medical house is the global medical file record holder

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: ...

{
  "doctor": null,
  "medicalHouse": {
```

```
"name": "MEDISCH HUIS COLIN",  
"nihii": "81150594",  
"contactId":  
"235fcc3b4458f7b910083d8f32c0425de524e55de30f3421a7608fb3292a4678b7ee7e6c046217f3f8226  
09b173afc912c8e5b0e6a1863b6925baa9831fa7b9f",  
"label": "Label Medisch Huis Colin"  
}  
}
```

Chapter 14. V4 Recip-E

14.1. General

A prescription can have the following statuses:

Status	Description
NOT_DELIVERED	The prescription is registered at Recip-e but not yet delivered to the patient or executed by the executor.
IN_PROCESS	The prescription is being processed by an executor. The prescription typically transitions into this state by scanning the prescription's barcode.
DELIVERED	The prescription is delivered to the patient, it's contents are no longer retrievable from Recip-e.
ARCHIVED	The prescription has been archived by the executor, it's contents are no longer retrievable from Recip-e.
REVOKED	The prescription has been revoked by either the prescriber or an executor. It's contents are no longer retrievable from Recip-e.
EXPIRED	The prescription is expired and can no longer be processed. It's contents are no longer retrievable from Recip-e.

14.1.1. Supported qualities

The Recip-E endpoints can be used only with credentials having one of following qualities:

- DOCTOR, DENTIST, HOSPITAL, PHYSIOTHERAPIST, NURSE, MIDWIFE

14.2. Recipe Prescriber

This chapter will explain the flow to create, register and manage prescriptions as a prescriber.

14.2.1. Register a prescription

To register a prescription, send the kmehr of the prescription together with the Ehealth Credential to use to the Unified Messaging or Hector.

- See the [Frontend module](#) to retrieve the kmehr of the prescription.
- The Ehealth Credential to use is provided by setting the Default-Ehealth-Credential-Id header with the actor id of the desired credential.

- The available credentials can be retrieved from the backend by using the credential API (see section 5. Consulting Credentials).
- [Optional]: The Prescription session id. When creating multiple prescriptions for the same patient, you can use the prescription session id to speed up subsequent calls.
- [Optional]: The boolean (default value: false) Prescription locked. When creating a prescription with Prescription-Locked set to true, it will only be visible in the pharmacy where the patient presents the (paper) proof of prescription.

Example request

```
POST /api/v4/recipe/prescriber/prescriptions HTTP/1.1
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/xml; charset=UTF-8
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Prescription-Session-Id: li9BfuQBxi4dMK6CNcSjLA==
Prescription-Locked: true
Content-Length: 5385

<?xml version="1.0" encoding="UTF-8"?>
<kmehrmessage xmlns="http://www.ehealth.fgov.be/standards/kmehr/schema/v1">
  <header>
    <standard>
      <cd S="CD-STANDARD" SV="1.20">20161201</cd>
    </standard>
    <id S="ID-KMEHR" SV="1.0">19006951001.200901100900000000</id>
    <id S="LOCAL" SL="ID-MEDISOFT" SV="versie 1.23.25.0">8e1c4ea4-3825-48e4-
bcc2b8cadfa7a897</id>
    <date>2016-09-28</date>
    <time>09:00:00</time>
    <sender>
      <hcparty>
        <id S="ID-HCPARTY" SV="1.0">39015378000</id>
        <cd S="CD-HCPARTY" SV="1.11">persdentist</cd>
        <firstname>Tom</firstname>
        <familyname>Vleminckx</familyname>
      </hcparty>
      <hcparty>
        <cd S="CD-HCPARTY" SV="1.11">application</cd>
        <name>MySoftware</name>
        <telecom>
          <cd S="CD-ADDRESS" SV="1.1">work</cd>
          <cd S="CD-TELECOM" SV="1.0">phone</cd>
          <telecomnumber>02/100.11.12</telecomnumber>
        </telecom>
        <telecom>
          <cd S="CD-ADDRESS" SV="1.1">work</cd>
          <cd S="CD-TELECOM" SV="1.0">email</cd>
```

```

        <telecomnumber>tom@mysoftware.com</telecomnumber>
    </telecom>
</hcparty>
</sender>
<recipient>
    <hcparty>
        <id S="ID-HCPARTY" SV="1.0">RECIPE</id>
        <cd S="CD-HCPARTY" SV="1.11">orgpublichealth</cd>
        <name>Recip-e</name>
    </hcparty>
</recipient>
</header>
<folder>
    <id S="ID-KMEHR" SV="1.0">1</id>
    <patient>
        <id S="ID-PATIENT" SV="1.0">76020727360</id>
        <firstname>Fred</firstname>
        <familyname>Flintstone</familyname>
        <birthdate>
            <date>1976-02-07</date>
        </birthdate>
        <sex>
            <cd S="CD-SEX" SV="1.1">male</cd>
        </sex>
    </patient>
    <transaction>
        <id S="ID-KMEHR" SV="1.0">1</id>
        <cd S="CD-TRANSACTION" SV="1.10">pharmaceuticalprescription</cd>
        <date>2016-09-28</date>
        <time>09:00:00</time>
        <author>
            <hcparty>
                <id S="ID-HCPARTY" SV="1.0">39015378000</id>
                <cd S="CD-HCPARTY" SV="1.11">persdentist</cd>
                <name>Vleminckx Tom</name>
                <address>
                    <cd S="CD-ADDRESS" SV="1.1">work</cd>
                    <country>
                        <cd S="CD-FED-COUNTRY" SV="1.2">be</cd>
                    </country>
                    <zip>1000</zip>
                    <city>Brussel</city>
                    <street>Grote Markt</street>
                    < housenumber>7</ housenumber>
                </address>
                <telecom>
                    <cd S="CD-ADDRESS" SV="1.1">work</cd>
                    <cd S="CD-TELECOM" SV="1.0">phone</cd>
                    <telecomnumber>02/221.21.21</telecomnumber>
                </telecom>
            </hcparty>

```

```

</author>
<iscomplete>true</iscomplete>
<isvalidated>true</isvalidated>
<heading>
  <id S="ID-KMEHR" SV="1.0">1</id>
  <cd S="CD-HEADING" SV="1.2">prescription</cd>
  <item>
    <id S="ID-KMEHR" SV="1.0">1</id>
    <cd S="CD-ITEM" SV="1.1">medication</cd>
    <content>
      <medicinalproduct>
        <intendedcd S="CD-DRUG-CNK"
SV="LOCALDB">0318717</intendedcd>
        <intendedname>ADALAT OROS 30 COMP 28 X 30
MG</intendedname>
      </medicinalproduct>
    </content>
    <beginmoment>
      <date>2016-09-28</date>
    </beginmoment>
    <lifecycle>
      <cd S="CD-LIFECYCLE" SV="1.9">prescribed</cd>
    </lifecycle>
    <quantity>
      <decimal>1</decimal>
    </quantity>
    <posology>
      <text L="nl">1 tablet per dag</text>
    </posology>
  </item>
</heading>
</transaction>
</folder>
</kmehrmessage>

```

Example response

```

HTTP/1.1 201 Created
Content-Type: application/json; charset=UTF-8
Content-Length: 179
Prescription-Session-Id: li9BfuQBXi4dMK6CNcSjLA==

```

```

{
  "id": "BEP06YKWZE7X",
  "created": "2017-07-13T12:35:46Z",
  "status": "NOT_DELIVERED",
  "patient": {
    "ssin": "76020727360",
    "name": "Flintstone",
    "givenName": "Fred",

```

```
"gender": "MALE",  
"dateOfBirth": "1976-02-06T23:00:00Z"  
}  
}
```

14.2.2. Fetch the Prescription / pdf / kmehr / preview

This will only look in the local Database. In other words, you can only retrieve prescriptions, that were made by the authenticator.

It's possible to extract the PDF and/or preview (html-representation) from the prescription.

The supplied Accept-Language header will be used for the language of the transcript: *nl*, *en* and *fr* are available.

- To retrieve the json of the prescription, set the Accept header to *application/json*
- To retrieve the PDF, set the Accept header to *application/pdf*
- To retrieve the preview, set the Accept header to *text/html*
- To retrieve the kmehr, set the Accept header to *application/xml*

It is possible for the pdf, preview and kmehr to request it inline with the request parameter *inline* set to *true*

Example request as json

```
GET /api/v4/ recip-e/prescriber/prescriptions/BEP06YKWZE7X HTTP/1.1  
Default-Ehealth-Credential-Id:  
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53  
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7  
Accept-Language: nl-BE  
Accept: application/json  
Authorization: Basic *****
```

Example response for json

```
HTTP/1.1 200 OK  
Content-Type: application/json; charset=UTF-8  
Content-Length: 179  
  
{  
  "id": "BEP06YKWZE7X",  
  "created": "2017-07-13T12:35:46Z",  
  "status": "DELIVERED",  
  "patient": {  
    "ssin": "76020727360",  
    "name": "Flintstone",  
    "givenName": "Fred",  
    "gender": "MALE",  
    "dateOfBirth": "1976-02-06T23:00:00Z"
```

```
}  
}
```

Example request as pdf

```
GET /api/v4/ recip-e/prescriber/prescriptions/BEP06YKWZE7X?inline=true HTTP/1.1  
Default-Ehealth-Credential-Id:  
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53  
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7  
Accept-Language: nl-BE  
Accept: application/pdf  
Authorization: Basic *****
```

Example response for pdf

```
HTTP/1.1 200 OK  
Content-Type: application/pdf  
Content-Disposition: inline; filename=BEP06YKWZE7X.pdf  
Content-Length: 8309  
  
%PDF-1.4  
%  
4 0 obj  
<<  
  
...  
  
startxref  
7835  
%%EOF
```

14.2.3. Fetch all prescriptions

Only local prescriptions will be retrieved.

Example request

```
GET /api/v4/ recip-e/prescriber/prescriptions HTTP/1.1  
Default-Ehealth-Credential-Id:  
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53  
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7  
Accept-Language: nl-BE  
Authorization: Basic *****  
Content-Type: application/json
```

Example response

```
HTTP/1.1 200 OK
```

Content-Type: application/json; charset=UTF-8

```
[{
  "id": "BEP06YKWZE7X",
  "created": "2017-07-13T12:35:46Z",
  "status": "DELIVERED",
  "patient": {
    "ssin": "76020727360",
    "name": "Flintstone",
    "givenName": "Fred",
    "gender": "MALE",
    "dateOfBirth": "1976-02-06T23:00:00Z"
  }
}, {
  "id": "BEP13HGZES5B",
  "created": "2017-07-12T11:31:32Z",
  "status": "DELIVERED",
  "patient": {
    "ssin": "76020727360",
    "name": "Flintstone",
    "givenName": "Fred",
    "gender": "MALE",
    "dateOfBirth": "1976-02-06T23:00:00Z"
  }
}]
```

14.2.4. Refresh the status

The statuses of the prescriptions are synchronized with Recip-e on a regular interval. If you want to be sure that the status is up to date, an explicit refresh of the status is possible.

Example request

```
POST /api/v4/ recip-e/prescriber/prescriptions/BEP06YKWZE7X/status HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Accept-Language: nl-BE
Authorization: Basic *****
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
  "status": "NOT_DELIVERED"
}
```

14.2.5. Revoke the prescription

It is only possible to revoke a prescription at Recip-e if it has the status *NOT_DELIVERED*. To revoke a prescription, a reason is required.

Example request

```
POST /api/v4/ recip-e/prescriber/prescriptions/BEP06YKWZE7X/revocation HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json

{
  "reason": "The patient is healthy again, and doesn't need pills anymore."
}
```

Example response

```
HTTP/1.1 204 No Content
Content-Type: application/json; charset=UTF-8
```

14.2.6. Delete prescription

This endpoint will only delete the prescription from the local database. No actions at Recip-e are taken.

Example request

```
DELETE /api/v4/ recip-e/prescriber/prescriptions/BEP06YKWZE7X HTTP/1.1
Default-Ehealth-Credential-Id:
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Accept-Language: nl-BE
Authorization: Basic *****
```

Example response

```
HTTP/1.1 204 No Content
Content-Type: application/json; charset=UTF-8
```

14.2.7. Send notification

This endpoint will send a notification to a given caregiver for a given patient


```
0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f53
53d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7
Accept-Language: nl-BE
Authorization: Basic *****
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 1247

[ {
  "rid" : "BEP0A4EHT8SA",
  "feedback" : {
    "text" : "This is a feedback from pharmacy XXX"
  },
  "sentOn" : "2021-05-14T14:18:12.4687394Z",
  "executorNihii" : "14032237730",
  "executor" : {
    "identifyingEhealthQuality" : {
      "name" : "DOCTOR",
      "label" : "Doctor",
      "identifier" : {
        "value" : "14032237730",
        "type" : "NIHII"
      }
    }
  },
  "healthcareActorId" :
  "0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f5
  353d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7",
  "firstName" : "Carl",
  "lastName" : "Molemans"
}, {
  "rid" : "BEP0A4EHT8SB",
  "feedback" : {
    "text" : "This is a feedback from pharmacy XXX"
  },
  "sentOn" : "2021-05-14T14:18:12.4687394Z",
  "executorNihii" : "15964913001",
  "executor" : {
    "identifyingEhealthQuality" : {
      "name" : "DOCTOR",
      "label" : "Doctor",
      "identifier" : {
        "value" : "15964913001",
        "type" : "NIHII"
      }
    }
  },
  "healthcareActorId" :
```

```
"ee960435311b4cc2869853f6a268f01cc99d755413841b452f860e1e9404bff6161155754922b9a277b14  
036dabc3d0176a878a3c413fc97753189f9621623e6",  
  "firstName" : "Elhassan",  
  "lastName" : "Baazizi"  
}  
} ]
```

Chapter 15. V4 Rn-Consult

15.1. Supported Qualities

We currently support the following eHealth qualities:

- HOSPITAL
- LABO

15.2. Search Phonetically

Rn-Consult provides the ability to search for a citizen by name and birthdate.

A list of citizens which match "best" to your criteria will be returned.

WARNING

Be warned, whenever there are more than 20 results from the crossroads database, the eHealth endpoint will crash.

There is no way to guarantee the amount of results.

We therefore encourage you to restrict the birthdate period as much as possible.

Also mention be as specific as possible providing the actual search criteria.

15.2.1. Example Request

```
POST http://localhost:8080/api/v4/rn-consult/search
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692
e43cd65ae09817c87d020d66028e69610770a0a7d7

{
  "name": {
    "lastName": "string",
    "givenNames": [
      "string"
    ],
    "givenNameMatching": "COMPLETE_FIRST_GIVENNAME"
  },
  "birth": {
    "birthDate": "string",
    "variation": 0
  },
  "gender": {
    "genderCode": "M"
  },
}
```

```
"address": {
  "countryCode": 0,
  "cityCode": "string"
},
"maximumResultCount": 0
}
```

15.2.2. Example Response

CAUTION

Because of the sensitivity of the data returned for this service, this response has been heavily anonymized.

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 5064
[
  {
    "register": "string",
    "deceased": true,
    "ssin": "00010112345",
    "name": {
      "lastName": "string",
      "givenNames": [
        "string"
      ],
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  {
    "birthDate": "string",
    "gender": {
      "genderCode": "string",
      "inceptionDate": "2021-01-01",
      "additionalProp1": {}
    },
    "address": {
      "diplomaticPost": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "diplomaticPostCode": 0,
        "diplomaticPostName": "string"
      },
      "diplomaticAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "address": "string",
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
      }
    }
  }
]
```

```

},
"postAddress": {
  "countryCode": 0,
  "countryIsoCode": "string",
  "countryName": "string",
  "address": "string",
  "inceptionDate": "2021-01-01",
  "additionalProp1": {}
},
"referenceAddress": {
  "countryCode": 0,
  "countryIsoCode": "string",
  "countryName": "string",
  "regionCode": "string",
  "regionName": "string",
  "cityCode": "string",
  "cityRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "cityName": "string",
  "postalCode": "string",
  "streetCode": "string",
  "streetRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "streetName": "string",
  "houseNumber": "string",
  "boxNumber": "string",
  "addressRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "inceptionDate": "2021-01-01",
  "additionalProp1": {}
},
"residentialAddress": {
  "countryCode": 0,
  "countryIsoCode": "string",
  "countryName": "string",
  "regionCode": "string",
  "regionName": "string",
  "cityCode": "string",
  "cityRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  }
}

```

```

    },
    "cityName": "string",
    "postalCode": "string",
    "streetCode": "string",
    "streetRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "streetName": "string",
    "houseNumber": "string",
    "boxNumber": "string",
    "addressRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "temporaryAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "address": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  }
},
"contactAddress": {
  "countryCode": 0,
  "countryIsoCode": "string",
  "countryName": "string",
  "regionCode": "string",
  "regionName": "string",
  "cityCode": "string",
  "cityRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "cityName": "string",
  "postalCode": "string",
  "streetCode": "string",
  "streetRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "streetName": "string",
  "houseNumber": "string",

```

```

    "boxNumber": "string",
    "addressRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "typeCode": 0,
    "typeDescription": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "administrator": {
    "specialNotion": {
      "specialNotionCode": "string",
      "specialNotionDescription": "string"
    },
    "location": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "anomalies": [
    {
      "code": "string",
      "description": "string"
    }
  ]
}
]

```

15.3. Fetch by SSIN

Rn-Consult provides the ability to fetch a citizen provided their SSIN.

A matching record will be returned for the provided SSIN if one exists.

15.3.1. Example Request

```

GET http://localhost:8080/api/v4/rn-consult/00010112345
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692

```

15.3.2. Example Response

CAUTION

Because of the sensitivity of the data returned for this service, this response has been heavily anonymized.

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 8114
{
  "ssin": {
    "value": "00010112345",
    "cancelled": true,
    "replaces": "00010112345"
  },
  "result": {
    "register": "string",
    "registerInceptionDate": "2021-01-01",
    "ssin": "00010112345",
    "nobilityTitle": {
      "nobilityTitleCode": "string",
      "nobilityTitleDescription": "string",
      "inceptionDate": "2021-01-01"
    },
    "name": {
      "lastName": "string",
      "givenNames": [
        "string"
      ],
      "inceptionDate": "2021-01-01",
      "additionalProp1": {}
    },
    "nationalities": [
      {
        "nationalityCode": 0,
        "nationalityDescription": "string",
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
      }
    ],
    "birth": {
      "birthDate": "string",
      "birthPlace": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
      }
    }
  }
}
```

```

    },
    "actType": {
      "actTypeCode": 0,
      "actTypeDescription": "string"
    },
    },
    "additionalProp1": {}
  },
  "decease": {
    "deceaseDate": "string",
    "deceasePlace": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    },
    "additionalProp1": {}
  },
  "gender": {
    "genderCode": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "civilStates": [
    {
      "civilStateCode": 0,
      "civilStateDescription": "string",
      "partner": {
        "partnerFictionalIdentificationNumber": "string",
        "partnerSsin": "00010112345",
        "partnerName": {
          "lastName": "string",
          "givenNames": [
            "string"
          ],
          "inceptionDate": "2021-01-01",
          "additionalProp1": {}
        },
      },
    },
    "location": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    },
    "judgment": {
      "judgementDate": "string",
      "judgmentLocation": {
        "countryCode": 0,
        "countryIsoCode": "string",

```

```

        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
    },
    "additionalProp1": {}
},
"transcription": {
    "transcriptionDate": "string",
    "transcriptionLocation": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
    }
},
"inceptionDate": "2021-01-01",
"additionalProp1": {}
}
],
"address": {
    "diplomaticPost": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "diplomaticPostCode": 0,
        "diplomaticPostName": "string"
    },
    "diplomaticAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "address": "string",
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
    },
    "postAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "address": "string",
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
    },
    "referenceAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "regionCode": "string",
        "regionName": "string",
        "cityCode": "string",

```

```

    "cityRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "cityName": "string",
    "postalCode": "string",
    "streetCode": "string",
    "streetRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "streetName": "string",
    "houseNumber": "string",
    "boxNumber": "string",
    "addressRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "residentialAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "regionCode": "string",
    "regionName": "string",
    "cityCode": "string",
    "cityRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "cityName": "string",
    "postalCode": "string",
    "streetCode": "string",
    "streetRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "streetName": "string",
    "houseNumber": "string",
    "boxNumber": "string",
    "addressRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    }
  }
}

```

```

    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "temporaryAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "address": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  }
},
"contactAddress": {
  "countryCode": 0,
  "countryIsoCode": "string",
  "countryName": "string",
  "regionCode": "string",
  "regionName": "string",
  "cityCode": "string",
  "cityRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "cityName": "string",
  "postalCode": "string",
  "streetCode": "string",
  "streetRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "streetName": "string",
  "houseNumber": "string",
  "boxNumber": "string",
  "addressRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "typeCode": 0,
  "typeDescription": "string",
  "inceptionDate": "2021-01-01",
  "additionalProp1": {}
},
"administrator": {
  "specialNotion": {
    "specialNotionCode": "string",
    "specialNotionDescription": "string"
  }
},

```

```

    "location": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "subregister": {
    "subregisterCode": 0,
    "subregisterDescription": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "legalCohabitation": {
    "partner": {
      "partnerFictionalIdentificationNumber": "string",
      "partnerSsin": {
        "type": "00010112345",
        "dateOfBirth": "2000-01-01",
        "female": false,
        "male": true,
        "anonymizedSsin": "*****345",
        "baseIdentifier": {
          "type": "NIHII",
          "baseIdentifier": "string"
        }
      },
      "partnerName": {
        "lastName": "string",
        "givenNames": [
          "string"
        ],
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
      }
    },
    "registration": {
      "registrationDate": "2021-01-01",
      "location": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
      },
      "registrationBailiff": {
        "bailiffName": "string",
        "location": {

```

```

        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
      }
    },
    "inceptionDate": "2021-01-01"
  },
  "anomalies": [
    {
      "code": "string",
      "description": "string"
    }
  ]
}

```

15.4. Fetch person history by SSIN

Rn-Consult provides the ability to fetch a citizen's history provided their SSIN matching the desired criteria.

A matching record will be returned for the provided SSIN if one exists.

15.4.1. Example Request

```

GET http://localhost:8080/api/v4/rn-consult/00010112345/history?criteria=NAMES,NATIONALITIES,BIRTHS,DECEASES,GENDERS,CIVIL_STATES,ADDRESSES,CONTACT_ADDRESSES,ADMINISTRATORS,SUBREGISTERS
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692e43cd65ae09817c87d020d66028e69610770a0a7d7

```

15.4.2. Example Response

CAUTION

Because of the sensitivity of the data returned for this service, this response has been heavily anonymized.

```

HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 8114
{

```

```

"ssin": {
  "value": "00010112345",
  "cancelled": true,
  "replaces": "00010112345"
},
"result": {
  "ssin": "00010112345",
  "names": [
    {
      "lastName": "string",
      "givenNames": [
        "string"
      ],
      "inceptionDate": "2021-01-01",
      "expiryDate": "2021-01-01",
      "source": "string",
      "additionalProp1": {}
    }
  ],
  "nationalities": [
    {
      "nationalityCode": 0,
      "nationalityDescription": "string",
      "inceptionDate": "2021-01-01",
      "expiryDate": "2021-01-01",
      "source": "string",
      "additionalProp1": {}
    }
  ],
  "births": [
    {
      "birthDate": "string",
      "birthPlace": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
      },
      "actType": {
        "actTypeCode": 0,
        "actTypeDescription": "string"
      },
      "source": "string",
      "additionalProp1": {}
    }
  ],
  "deceases": [
    {
      "deceaseDate": "string",
      "deceasePlace": {

```

```

        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
    },
    "source": "string",
    "additionalProp1": {}
}
],
"genders": [
    {
        "genderCode": "string",
        "inceptionDate": "2021-01-01",
        "expiryDate": "2021-01-01",
        "source": "string",
        "additionalProp1": {}
    }
],
"civilStates": [
    {
        "civilStateCode": 0,
        "civilStateDescription": "string",
        "partner": {
            "partnerFictionalIdentificationNumber": "string",
            "partnerSsin": "00010112345",
            "partnerName": {
                "lastName": "string",
                "givenNames": [
                    "string"
                ],
                "inceptionDate": "2021-01-01",
                "additionalProp1": {}
            }
        },
        "location": {
            "countryCode": 0,
            "countryIsoCode": "string",
            "countryName": "string",
            "cityCode": "string",
            "cityName": "string"
        },
        "judgment": {
            "judgementDate": "string",
            "judgmentLocation": {
                "countryCode": 0,
                "countryIsoCode": "string",
                "countryName": "string",
                "cityCode": "string",
                "cityName": "string"
            }
        },
    }
],

```

```

    "additionalProp1": {}
  },
  "transcription": {
    "transcriptionDate": "string",
    "transcriptionLocation": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    }
  },
  "inceptionDate": "2021-01-01",
  "expiryDate": "2021-01-01",
  "source": "string",
  "additionalProp1": {}
}
],
"addresses": [
  {
    "diplomaticPost": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "diplomaticPostCode": 0,
      "diplomaticPostName": "string"
    },
    "diplomaticAddress": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "address": "string",
      "inceptionDate": "2021-01-01",
      "expiryDate": "2021-01-01",
      "additionalProp1": {}
    },
    "postAddresses": [
      {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "address": "string",
        "inceptionDate": "2021-01-01",
        "expiryDate": "2021-01-01",
        "additionalProp1": {}
      }
    ],
    "referenceAddress": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",

```

```

"regionCode": "string",
"regionName": "string",
"cityCode": "string",
"cityRegionalCode": {
  "namespace": "string",
  "objectIdentifier": "string",
  "versionIdentifier": "string"
},
"cityName": "string",
"postalCode": "string",
"streetCode": "string",
"streetRegionalCode": {
  "namespace": "string",
  "objectIdentifier": "string",
  "versionIdentifier": "string"
},
"streetName": "string",
"houseNumber": "string",
"boxNumber": "string",
"addressRegionalCode": {
  "namespace": "string",
  "objectIdentifier": "string",
  "versionIdentifier": "string"
},
"inceptionDate": "2021-01-01",
"expiryDate": "2021-01-01",
"additionalProp1": {}
},
"residentialAddress": {
  "countryCode": 0,
  "countryIsoCode": "string",
  "countryName": "string",
  "regionCode": "string",
  "regionName": "string",
  "cityCode": "string",
  "cityRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "cityName": "string",
  "postalCode": "string",
  "streetCode": "string",
  "streetRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "streetName": "string",
  "houseNumber": "string",
  "boxNumber": "string",

```

```

        "addressRegionalCode": {
            "namespace": "string",
            "objectIdentifier": "string",
            "versionIdentifier": "string"
        },
        "inceptionDate": "2021-01-01",
        "expiryDate": "2021-01-01",
        "additionalProp1": {}
    },
    "temporaryAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "address": "string",
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
    },
    "source": "string"
}
],
"contactAddresses": [
    {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "regionCode": "string",
        "regionName": "string",
        "cityCode": "string",
        "cityRegionalCode": {
            "namespace": "string",
            "objectIdentifier": "string",
            "versionIdentifier": "string"
        },
        "cityName": "string",
        "postalCode": "string",
        "streetCode": "string",
        "streetRegionalCode": {
            "namespace": "string",
            "objectIdentifier": "string",
            "versionIdentifier": "string"
        },
        "streetName": "string",
        "houseNumber": "string",
        "boxNumber": "string",
        "addressRegionalCode": {
            "namespace": "string",
            "objectIdentifier": "string",
            "versionIdentifier": "string"
        },
        "typeCode": 0,
        "typeDescription": "string",

```

```

        "inceptionDate": "2021-01-01",
        "expiryDate": "2021-01-01",
        "source": "string",
        "additionalProp1": {}
    }
],
"administrators": [
    {
        "specialNotion": {
            "specialNotionCode": "string",
            "specialNotionDescription": "string"
        },
        "location": {
            "countryCode": 0,
            "countryIsoCode": "string",
            "countryName": "string",
            "cityCode": "string",
            "cityName": "string"
        },
        "inceptionDate": "2021-01-01",
        "expiryDate": "2021-01-01",
        "source": "string",
        "additionalProp1": {}
    }
],
"subregisters": [
    {
        "subregisterCode": 0,
        "subregisterDescription": "string",
        "inceptionDate": "2021-01-01",
        "expiryDate": "2021-01-01",
        "source": "string",
        "additionalProp1": {}
    }
],
"anomalies": [
    {
        "code": "string",
        "description": "string"
    }
],
"register": "string",
"registerInceptionDate": "2021-01-01"
}
}

```

15.5. Add or remove subscription

Based on a unique identifier SSIN, users will have the opportunity to subscribe to or unsubscribe from the notifications following from changes on a person's data.

15.5.1. Example Request

```
POST http://localhost:8080/api/v4/rn-consult/00010112345/subscription
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692
e43cd65ae09817c87d020d66028e69610770a0a7d7
```

```
DELETE http://localhost:8080/api/v4/rn-consult/00010112345/subscription
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692
e43cd65ae09817c87d020d66028e69610770a0a7d7
```

15.5.2. Example Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 44
{
  "value": "00010112345",
  "replacing": false
}
```

15.6. Get notification

Based on the subscriptions someone has added, that person can fetch all notifications related to all the people someone is subscribed to.

15.6.1. Example Request

```
GET http://localhost:8080/api/v4/rn-consult/notification
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692
e43cd65ae09817c87d020d66028e69610770a0a7d7
```

15.6.2. Example Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 19496
{
  "count": 3,
  "ackId": "123",
  "cancelationNotifications": [
    {
      "ssin": "00010112345",
      "notificationInformation": {
        "timestamp": "2021-05-05T09:12:33Z",
        "reason": "string",
        "notificationId": "string"
      }
    }
  ],
  "replacementNotifications": [
    {
      "notificationInformation": {
        "timestamp": "2021-05-05T09:12:33Z",
        "reason": "string",
        "notificationId": "string"
      },
      "ssin": {
        "value": "00010112345",
        "canceled": true,
        "replacedBy": "00010112345"
      },
      "replacingPerson": {
        "ssin": "00010112345",
        "nobilityTitle": {
          "nobilityTitleCode": "string",
          "nobilityTitleDescription": "string",
          "inceptionDate": "2021-01-01"
        },
        "name": {
          "lastName": "string",
          "givenNames": [
            "string"
          ],
          "inceptionDate": "2021-01-01",
          "additionalProp1": {}
        },
        "nationalities": [
          {
            "nationalityCode": 0,
            "nationalityDescription": "string",
            "inceptionDate": "2021-01-01",
```

```

    "additionalProp1": {}
  },
  "birth": {
    "birthDate": "string",
    "birthPlace": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    "actType": {
      "actTypeCode": 0,
      "actTypeDescription": "string"
    },
    "additionalProp1": {}
  },
  "decease": {
    "deceaseDate": "string",
    "deceasePlace": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    "additionalProp1": {}
  },
  "gender": {
    "genderCode": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "civilStates": [
    {
      "civilStateCode": 0,
      "civilStateDescription": "string",
      "partner": {
        "partnerFictionalIdentificationNumber": "string",
        "partnerSsin": "00010112345",
        "partnerName": {
          "lastName": "string",
          "givenNames": [
            "string"
          ],
          "inceptionDate": "2021-01-01",
          "additionalProp1": {}
        }
      },
      "location": {

```

```

        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
    },
    "judgment": {
        "judgementDate": "string",
        "judgmentLocation": {
            "countryCode": 0,
            "countryIsoCode": "string",
            "countryName": "string",
            "cityCode": "string",
            "cityName": "string"
        },
        "additionalProp1": {}
    },
    "transcription": {
        "transcriptionDate": "string",
        "transcriptionLocation": {
            "countryCode": 0,
            "countryIsoCode": "string",
            "countryName": "string",
            "cityCode": "string",
            "cityName": "string"
        }
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
}
],
"address": {
    "diplomaticPost": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "diplomaticPostCode": 0,
        "diplomaticPostName": "string"
    },
    "diplomaticAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "address": "string",
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
    },
    "postAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",

```

```

    "address": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "referenceAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "regionCode": "string",
    "regionName": "string",
    "cityCode": "string",
    "cityRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "cityName": "string",
    "postalCode": "string",
    "streetCode": "string",
    "streetRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "streetName": "string",
    "houseNumber": "string",
    "boxNumber": "string",
    "addressRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "residentialAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "regionCode": "string",
    "regionName": "string",
    "cityCode": "string",
    "cityRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "cityName": "string",
    "postalCode": "string",
    "streetCode": "string",
    "streetRegionalCode": {

```

```

        "namespace": "string",
        "objectIdentifier": "string",
        "versionIdentifier": "string"
    },
    "streetName": "string",
    "houseNumber": "string",
    "boxNumber": "string",
    "addressRegionalCode": {
        "namespace": "string",
        "objectIdentifier": "string",
        "versionIdentifier": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
},
"temporaryAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "address": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
}
},
"contactAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "regionCode": "string",
    "regionName": "string",
    "cityCode": "string",
    "cityRegionalCode": {
        "namespace": "string",
        "objectIdentifier": "string",
        "versionIdentifier": "string"
    },
    "cityName": "string",
    "postalCode": "string",
    "streetCode": "string",
    "streetRegionalCode": {
        "namespace": "string",
        "objectIdentifier": "string",
        "versionIdentifier": "string"
    },
    "streetName": "string",
    "houseNumber": "string",
    "boxNumber": "string",
    "addressRegionalCode": {
        "namespace": "string",
        "objectIdentifier": "string",
        "versionIdentifier": "string"
    }
}

```

```

    },
    "typeCode": 0,
    "typeDescription": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "administrator": {
    "specialNotion": {
      "specialNotionCode": "string",
      "specialNotionDescription": "string"
    },
    "location": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "subregister": {
    "subregisterCode": 0,
    "subregisterDescription": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "legalCohabitation": {
    "partner": {
      "partnerFictionalIdentificationNumber": "string",
      "partnerSsin": "00010112345",
      "partnerName": {
        "lastName": "string",
        "givenNames": [
          "string"
        ],
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
      }
    },
    "registration": {
      "registrationDate": "2021-01-01",
      "location": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
      },
      "registrationBailiff": {
        "bailiffName": "string",

```

```

        "location": {
            "countryCode": 0,
            "countryIsoCode": "string",
            "countryName": "string",
            "cityCode": "string",
            "cityName": "string"
        }
    },
    "inceptionDate": "2021-01-01"
},
"anomalies": [
    {
        "code": "string",
        "description": "string"
    }
],
"register": "string",
"registerInceptionDate": "2021-01-01"
}
},
],
"updateNotifications": [
    {
        "notificationInformation": {
            "timestamp": "2021-05-05T09:12:33Z",
            "reason": "string",
            "notificationId": "string"
        },
        "ssin": "00010112345",
        "person": {
            "ssin": "00010112345",
            "nobilityTitle": {
                "nobilityTitleCode": "string",
                "nobilityTitleDescription": "string",
                "inceptionDate": "2021-01-01"
            },
            "name": {
                "lastName": "string",
                "givenNames": [
                    "string"
                ],
                "inceptionDate": "2021-01-01",
                "additionalProp1": {}
            },
            "nationalities": [
                {
                    "nationalityCode": 0,
                    "nationalityDescription": "string",
                    "inceptionDate": "2021-01-01",
                    "additionalProp1": {}
                }
            ]
        }
    }
]

```

```

    }
  ],
  "birth": {
    "birthDate": "string",
    "birthPlace": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    "actType": {
      "actTypeCode": 0,
      "actTypeDescription": "string"
    },
    "additionalProp1": {}
  },
  "decease": {
    "deceaseDate": "string",
    "deceasePlace": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    "additionalProp1": {}
  },
  "gender": {
    "genderCode": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "civilStates": [
    {
      "civilStateCode": 0,
      "civilStateDescription": "string",
      "partner": {
        "partnerFictionalIdentificationNumber": "string",
        "partnerSsin": "00010112345",
        "partnerName": {
          "lastName": "string",
          "givenNames": [
            "string"
          ],
          "inceptionDate": "2021-01-01",
          "additionalProp1": {}
        }
      },
      "location": {
        "countryCode": 0,

```

```

        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
    },
    "judgment": {
        "judgementDate": "string",
        "judgmentLocation": {
            "countryCode": 0,
            "countryIsoCode": "string",
            "countryName": "string",
            "cityCode": "string",
            "cityName": "string"
        },
        "additionalProp1": {}
    },
    "transcription": {
        "transcriptionDate": "string",
        "transcriptionLocation": {
            "countryCode": 0,
            "countryIsoCode": "string",
            "countryName": "string",
            "cityCode": "string",
            "cityName": "string"
        }
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
}
],
"address": {
    "diplomaticPost": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "diplomaticPostCode": 0,
        "diplomaticPostName": "string"
    },
    "diplomaticAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "address": "string",
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
    },
    "postAddress": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "address": "string",

```

```

    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "referenceAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "regionCode": "string",
    "regionName": "string",
    "cityCode": "string",
    "cityRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "cityName": "string",
    "postalCode": "string",
    "streetCode": "string",
    "streetRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "streetName": "string",
    "houseNumber": "string",
    "boxNumber": "string",
    "addressRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "residentialAddress": {
    "countryCode": 0,
    "countryIsoCode": "string",
    "countryName": "string",
    "regionCode": "string",
    "regionName": "string",
    "cityCode": "string",
    "cityRegionalCode": {
      "namespace": "string",
      "objectIdentifier": "string",
      "versionIdentifier": "string"
    },
    "cityName": "string",
    "postalCode": "string",
    "streetCode": "string",
    "streetRegionalCode": {
      "namespace": "string",

```

```

    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "streetName": "string",
  "houseNumber": "string",
  "boxNumber": "string",
  "addressRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "inceptionDate": "2021-01-01",
  "additionalProp1": {}
},
"temporaryAddress": {
  "countryCode": 0,
  "countryIsoCode": "string",
  "countryName": "string",
  "address": "string",
  "inceptionDate": "2021-01-01",
  "additionalProp1": {}
}
},
"contactAddress": {
  "countryCode": 0,
  "countryIsoCode": "string",
  "countryName": "string",
  "regionCode": "string",
  "regionName": "string",
  "cityCode": "string",
  "cityRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "cityName": "string",
  "postalCode": "string",
  "streetCode": "string",
  "streetRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },
  "streetName": "string",
  "houseNumber": "string",
  "boxNumber": "string",
  "addressRegionalCode": {
    "namespace": "string",
    "objectIdentifier": "string",
    "versionIdentifier": "string"
  },

```

```

    "typeCode": 0,
    "typeDescription": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "administrator": {
    "specialNotion": {
      "specialNotionCode": "string",
      "specialNotionDescription": "string"
    },
    "location": {
      "countryCode": 0,
      "countryIsoCode": "string",
      "countryName": "string",
      "cityCode": "string",
      "cityName": "string"
    },
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "subregister": {
    "subregisterCode": 0,
    "subregisterDescription": "string",
    "inceptionDate": "2021-01-01",
    "additionalProp1": {}
  },
  "legalCohabitation": {
    "partner": {
      "partnerFictionalIdentificationNumber": "string",
      "partnerSsin": "00010112345",
      "partnerName": {
        "lastName": "string",
        "givenNames": [
          "string"
        ],
        "inceptionDate": "2021-01-01",
        "additionalProp1": {}
      }
    },
    "registration": {
      "registrationDate": "2021-01-01",
      "location": {
        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
      },
      "registrationBailiff": {
        "bailiffName": "string",
        "location": {

```

```

        "countryCode": 0,
        "countryIsoCode": "string",
        "countryName": "string",
        "cityCode": "string",
        "cityName": "string"
    }
}
},
"inceptionDate": "2021-01-01"
},
"anomalies": [
    {
        "code": "string",
        "description": "string"
    }
],
"register": "string",
"registerInceptionDate": "2021-01-01"
},
"mutationEvents": [
    {
        "modificationTimestamp": "2021-05-05T09:12:33Z",
        "modifiedField": "string"
    }
]
}
]
}

```

15.7. Ack notification

Send an acknowledgement based on the acknowledgement id accompanied by the notifications that are fetched.

15.7.1. Example Request

```

POST http://localhost:8080/api/v4/rn-consult/notification/123
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbc5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692
e43cd65ae09817c87d020d66028e69610770a0a7d7

```

15.7.2. Example Response

```

HTTP/1.1 204 No Content

```

```
Content-Type: application/json; charset=UTF-8
```

15.8. Get current ssin history

Based on a unique identifier SSIN, users will have the opportunity to obtain the history of the unique identifier related to this person and to identify the current active identifier.

15.8.1. Example Request

```
GET http://localhost:8080/api/v4/rn-consult/00010112346/ssin-history/current
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692
e43cd65ae09817c87d020d66028e69610770a0a7d7
```

15.8.2. Example Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 90
{
  "lastActiveSsin": "00010112346",
  "canceled": true,
  "replacesSsin": "00010112345"
}
```

15.9. Get related ssins history

Based on a unique identifier SSIN, users will have the opportunity to obtain the history of the unique identifier related to this person and to identify the current active identifier.

Includes a list of identifiers related to the given person ssin.

15.9.1. Example Request

```
GET http://localhost:8080/api/v4/rn-consult/00010112347/ssin-history/related
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8
Rn-Consult-Ehealth-Credential-Id:
2b45fbd6111002ccbca5042592e30abce763630318ac17253bf1b9c19db380dd18989f1ddf4cb4f8d1d692
e43cd65ae09817c87d020d66028e69610770a0a7d7
```

15.9.2. Example Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 133
{
  "lastActiveSsin": "00010112347",
  "canceled": true,
  "replacesSsin": "00010112346",
  "relatedSsins": [
    "00010112345"
  ]
}
```

15.9.3. Error Codes

Table 31. General error codes

Code	Message	Description	HTTP code
authentication_error	Could not perform the operation because the service call is not authenticated.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	401
application_id_not_configured	The RN application identifier is not configured at eHealth.	Request eHealth to configure this application identifier.	401
authorization_error	Could not perform the operation because the sender is not allowed to perform this operation.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	403
not_found	The request completed successfully, but there was no data to return.	404	validation_error
Your input contains errors.	Description of the field(s) in error.	422	internal_error

Code	Message	Description	HTTP code
An unknown error occurred.	The reason and origin of this error cannot be processed automatically. Contact support to further investigate this problem. Do not forget to pass the unique id returned in the response.	500	too_many_results

Chapter 16. HealthPages

16.1. Search Contacts

HealthPages provides the ability to search for contacts using a query parameters.

Any string or number can be supplied to search for a contact to the query parameter "q", which is mandatory.

This can be the name of a caregiver or an organization.

But also an identifier (of type NIHIL, CBE, ...) associated to the contact.

The query parameter maxResults specifies the maximum number of results to be returned.

This is not a mandatory parameter, is 5 by default and has 50 as maximum.

A list of contacts which match "best" to your criteria will be returned.

Searching for contacts on HealthPages is exposed under [GET /api/v4/healthpages/contacts](#).

This endpoint is a pure proxy to the HealthPages 'Search for contacts using a simplified API.' endpoint.

The OpenAPI specification together with an example request and response of this endpoint can be found on [HealthPages API: Search Contacts](#)

16.2. Search contact cards

Searching for contact cards on HealthPages is exposed under [POST /api/v4/healthpages/contact-cards](#).

This endpoint is a pure proxy to the HealthPages 'Search Contact Cards' endpoint.

The OpenAPI specification together with an example request and response of this endpoint can be found on [HealthPages API: Search Contact Cards](#)

16.3. Manage caregivers for your organization

Updating the caregivers for an organization on HealthPages is exposed under [POST /api/v4/healthpages/organizations/{entityId}/caregivers/bulk](#).

This endpoint is a pure proxy to the HealthPages 'Add caregivers in bulk to a parent organization.' endpoint.

The OpenAPI specification together with an example request and response of this endpoint can be found on [HealthPages API: Add caregivers in bulk to a parent organization](#).

16.4. Manage departments for your organization

Updating the departments for an organization on HealthPages is exposed under [PUT /api/v4/healthpages/organizations/{entityId}/departments](#).

This endpoint is a pure proxy to the HealthPages 'Update departments linked to a parent organization' endpoint.

The OpenAPI specification together with an example request and response of this endpoint can be found on [HealthPages API: Update departments linked to a parent organization](#)

16.5. Resolve Addressees

Converts Healthcare actor identifiers to ADR addressees.

This enables you to construct an ADR xml with addressees, which are structurally determined to be correct.

NOTE

You can use the 'Search Contacts' endpoint to fetch the actor ids that you want to pass in the body of this request.

16.5.1. Request details

Converts Healthcare actor identifiers to ADR addressees.

This enables you to construct an ADR xml with addressees, which are structurally determined to be correct.

Given a list of max 15 healthcare actor identifiers, resolve as ADR.Addressee.

Opt-in possibilities to map unreachable contacts or to (not) fail when a healthcare actor identifier is missing from our datasource.

Parameters

Table 32. Body Parameter

Name	Description	Required
AddresseeRequest	AddresseeRequest	X

Table 33. Header Parameters

Name	Description	Required
Accept-Language	Language of the labels and descriptions returned.	X

Example Request

```
POST http://localhost:8080/api/v4/healthpages/addressees?q=14032237730
Accept-Language: nl-BE
Authorization: Basic *****
Content-Type: application/json; charset=UTF-8

{
  "actorIds": [

    "0492d47a07d5ad11e93107cfa88e896f768d8e12dba6f2f9c3b7fed8baacc3304358e3794397c2d9cb9f5
    353d52f9b7496e9ae2cf94e18edd47c4925e69cc8b7",
```

```
"003e23b43bb4a2dd18db2e7e96216eb0c705c0ee736c7614fcfe6be7d33500c8fbd1cccc3ca19079038d
4cd91adeda83e82e49ee4ed07c83c02eeb7c0102718"
]
}
```

16.5.2. Response details

Return Type

[ADR](#)

Content Type

- application/xml
- application/json

Responses

Table 34. http response codes

Code	Message
200	An ADR xml file which represent the healthcare actor identifiers as addressee elements.
400	Invalid input, the error code will be 'invalid_input' and the body will specify the invalid field.
500	The service failed while processing your request

Example Response

```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=UTF-8
Content-Length: 782
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ADR xmlns="urn:be:healthconnect:adr:1_0">
  <Addressee>
    <Quality>LABO</Quality>
    <Identifier>81160492</Identifier>
    <EhboxIdType>NIHII</EhboxIdType>
    <Name>HealthConnect</Name>
    <SubAddressee>
      <SubQualityType>DOCTOR</SubQualityType>
      <Identifier>14032237</Identifier>
      <FirstName>Carl</FirstName>
      <LastName>Molemans</LastName>
    </SubAddressee>
    <ApplicationId></ApplicationId>
  </Addressee>
  <Addressee>
```

```
<Quality>DOCTOR</Quality>
<Identifier>14032237730</Identifier>
<EhboxIdType>NIHII</EhboxIdType>
<FirstName>Carl</FirstName>
<LastName>Molemans</LastName>
</Addressee>
</ADR>
```

16.6. Models

16.6.1. ADR

ADR is an XML format which enables software to send EhealthBox messages using our proprietary software Hector and Unified-Messaging.

Field Name	Required	Type	Description
Addressee		List of [Addressee]	

16.6.2. Address

Field Name	Required	Type	Description
street	X	String	
houseNumber	X	String	
boxNumber		String	
postalCode	X	String	
city	X	String	
country	X	String	Uppercase two-letter country codes, as defined by ISO 3166-1.

16.6.3. AddresseeRequest

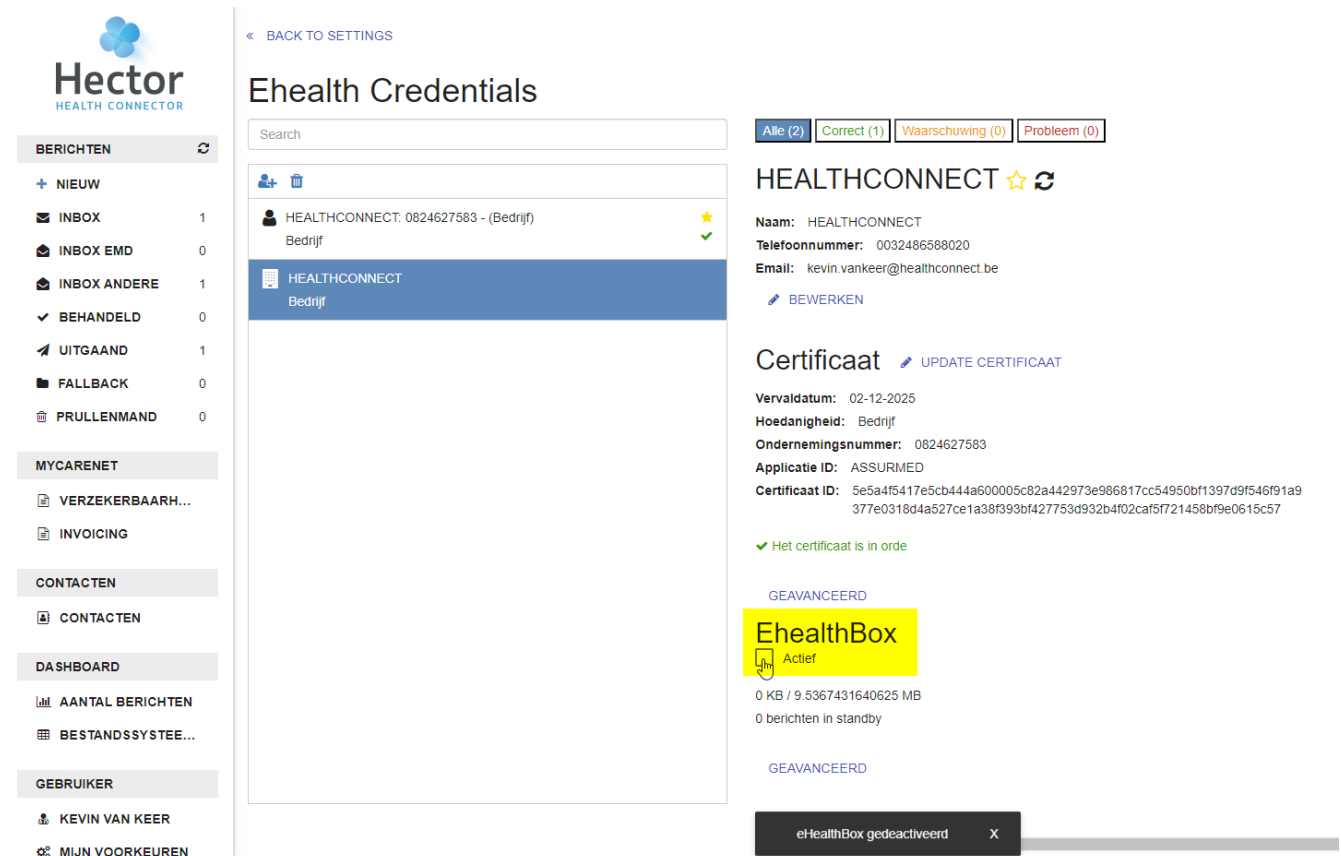
This is the filter object to which the ADR addressees will be restricted too.

Field Name	Required	Type	Description
actorIds	X	List of String	
ignoreInvalid		Boolean	
mapUnreachable		Boolean	

Chapter 17. V4 Assurmed

17.1. Assurmed credential configuration in MyBox

To be able to call Assurmed with your credential, you have to configure this in the MyBox settings. Assurmed can only be called using certificates with an application id with value 'ASSURMED'. When uploading this certificate, make sure to disable the eHealthBox communication for the Assurmed credential in the eHealth credentials configuration screen.



17.2. Get configured Assurmed credentials

Mybox offers the possibility to configure ehealth credentials to use Assurmed. This endpoint lists all the credentials that have been configured to use Assurmed.

17.2.1. Example Request

```
GET http://localhost:8080/api/v4/assurmed/credentials
Accept-Language: nl-BE
Authorization: Basic *****
```

17.2.2. Example Response

```
HTTP/1.1 200 OK
```

```
Content-Type: application/json; charset=UTF-8
Content-Length: 448
[
  {
    "healthcareActorId":
    "fp0zffuf8vxx303m884paczjfl51z3cojrenra7ovysy3ymiakmurtv4ixb1zlgpfp0zffuf8vxx303m884paczjfl51z3cojrenra7ovysy3ymiakmurtv4ixb1zlgp",
    "credentialLabel": "Dr John Doe"
  },
  {
    "healthcareActorId":
    "8f6b58a20ba89476680dfb1f65521e0607f7d0169ea049b16a338b0dd8dd5cd204ddc6930be44b709e9ce52f8bf0274cfcf18aa13feb3b7e5a3bcda83d2b7b75",
    "credentialLabel": "Ziekenhuis Sint-Maria Halle"
  }
]
```

17.3. Assumed proxy endpoints

Unified Messaging exposes a few Assumed endpoints as a pure proxy.

This means that responses and errors are returned as if the endpoint was called on Assumed directly.

Because of this, it is possible that the structured error response may diverge from the error response structure used by Unified Messaging.

Consult the [Assumed response documentation](#) for further information on the structured error response that can be returned.

17.3.1. List all attestations

Listing all assigned attestations on Assumed is exposed under **GET** `/api/v4/assumed/{healthcareActorId}/attestations`.

This endpoint is a pure proxy to the Assumed [Get all attestations](#) endpoint.

17.3.2. Get a single attestation

Retrieving a single attestation on Assumed is exposed under **GET** `/api/v4/assumed/{healthcareActorId}/attestations/{attestationId}`.

This endpoint is a pure proxy to the Assumed [Get an attestation](#) endpoint.

17.3.3. Submit/update a reimbursement request

Submitting or updating a reimbursement request on Assumed is exposed under **PUT** `/api/v4/assumed/{healthcareActorId}/attestations/{attestationId}/reimbursement-requests`.

This endpoint is a pure proxy to the Assumed [Submit/Update a reimbursement request](#) endpoint.

17.3.4. Reference Data

Assurmed provides some endpoints which provide textual descriptions of certain properties. These can then be visualized in the same way by all integrators. We highly recommend making use of these endpoints.

Consent Texts

Retrieving consent texts is exposed under `GET /api/v4/assurmed/{healthcareActorId}/reference-data/consent-texts`.

This endpoint is a pure proxy to the Assurmed [Get consent texts](#) endpoint.

Expense Categories

Retrieving expense categories is exposed under `GET /api/v4/assurmed/{healthcareActorId}/reference-data/expense-categories`.

This endpoint is a pure proxy to the Assurmed [Get expense categories](#) endpoint.

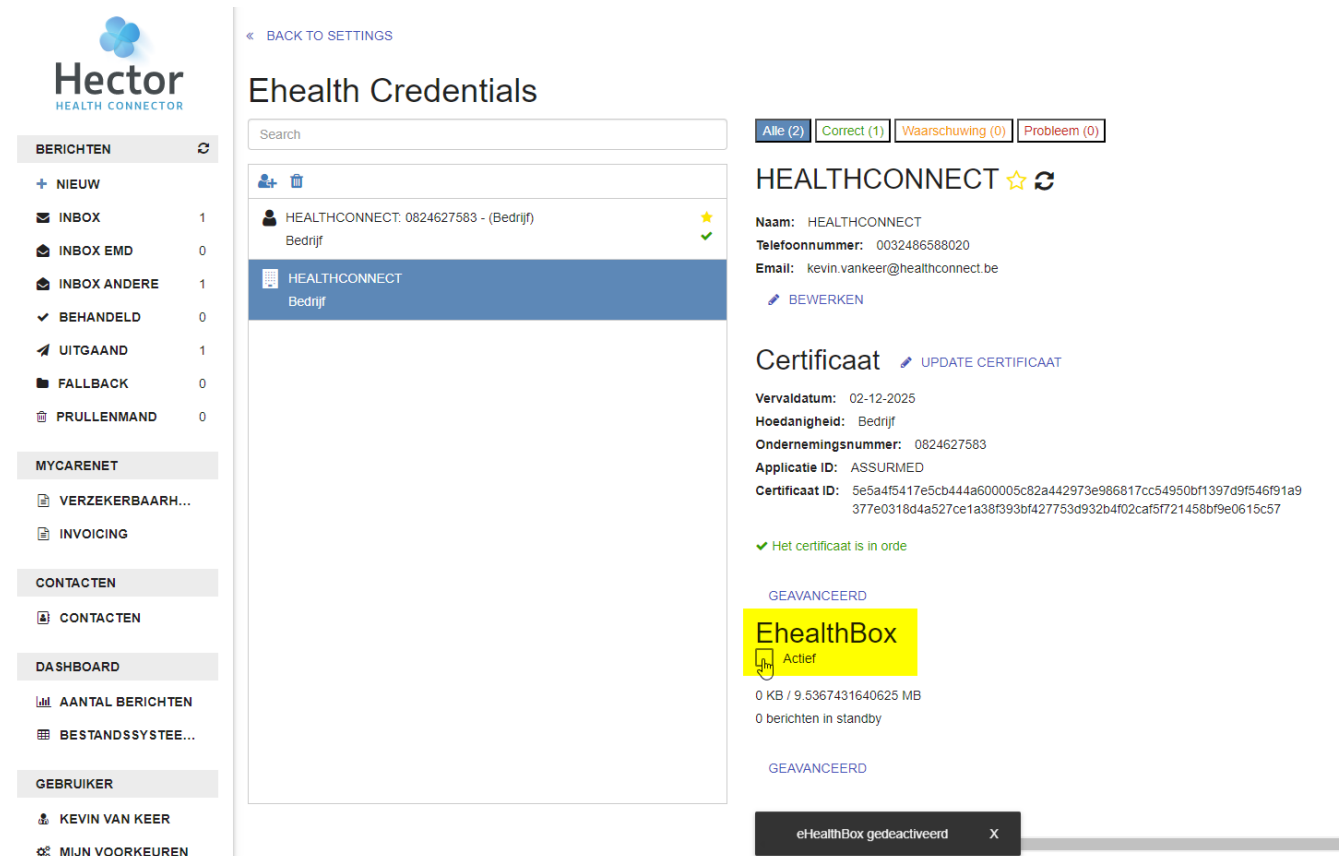
Chapter 18. V4 Vsb

18.1. Vsb credential configuration in MyBox

To be able to call Vsb with your credential, you have to configure this in the MyBox settings.

Vsb can only be called using certificates with the quality 'REEDUCATION'.

When uploading this certificate, make sure to disable the eHealthBox communication for the Vsb credential in the eHealth credentials configuration screen.



18.2. Get configured Vsb credentials

Mybox offers the possibility to configure ehealth credentials to use Vsb.

This endpoint lists all the credentials that have been configured to use Vsb.

18.2.1. Example Request

```
GET http://localhost:8080/api/v4/vsb/credentials
Accept-Language: nl-BE
Authorization: Basic *****
```

18.2.2. Example Response

```
HTTP/1.1 200 OK
```

```

Content-Type: application/json; charset=UTF-8
Content-Length: 448
[
  {
    "healthcareActorId":
"be5876cc9eb0afc3e98accc183063cc10dbaec799980235d325d07c08a5ee49967acad36afe64501e3011
a900c7968d8d053744211d31e8a4348c805d83d2ebf",
    "credentialLabel": "UNIVERSITAIR REVALIDATIECENTRUM VOOR
COMMUNICATIE TOORNISSEN UZA",
    "hcoNumber": "29615",
    "identifier": {
      "value": "95342189",
      "type": "NIHII"
    }
  },
  {
    "healthcareActorId":
"ea1c47f6be3e0390ce8184bbd798ae7926f19e5ba25d2ed7f2251fef32f4649e6586457b455c5c72f4c21
cfff12866aefa07f78195d7779be92f0bd4503c2aeaf",
    "credentialLabel": "VISUELE REVALIDATIE EN LOW VISION UZ ANTWERPEN",
    "hcoNumber": "29589",
    "identifier": {
      "value": "96900921",
      "type": "NIHII"
    }
  }
]

```

18.3. Vsb Gen Async proxy endpoints

Unified Messaging exposes a few Vsb endpoints as a pure proxy to Healthservices.

This means that responses and errors are returned as if the endpoint was called on Healthservices directly.

To be able to call these proxy endpoints, you need to have a credential configured as VSB credential through the MyBox settings.

Because of this, it is possible that the structured error response may diverge from the error response structure used by Unified Messaging.

Consult the [Healthservices response documentation](#) for further information on the structured error response that can be returned.

18.3.1. Get

Performing a get request to the MyVsbNet async bus is exposed under `GET /api/v4/vsb/gen-async/{service}/get`.

This endpoint is a pure proxy to the Healthservices [Vsb Gen Async Get](#) endpoint.

18.3.2. Post

Performing a post request to the MyVsbNet async bus is exposed under `POST /api/v4/vsb/gen-async/{service}/post`.

This endpoint is a pure proxy to the Healthservices [Vsb Gen Async Post](#) endpoint.

18.3.3. Confirm

Performing a confirm request to the MyVsbNet async bus is exposed under `POST /api/v4/vsb/gen-async/{service}/confirm`.

This endpoint is a pure proxy to the Healthservices [Vsb Gen Async Confirm](#) endpoint.

18.3.4. Get Insurability

Performing a get insurability request to MyVsbNet is exposed under `POST /api/v4/vsb/patient-insurability`.

This endpoint is a pure proxy to the Healthservices [Vsb Retrieve Insurability Sync](#) endpoint.

Chapter 19. V4 Timestamping

19.1. Timestamping credential configuration in MyBox

To be able to call Timestamping with your credential, you have to upload your timestamping ehealth credential with the ehealth credentials settings.

When uploading this certificate, make sure to disable the eHealthBox communication for the Timestamping credential in the eHealth credentials configuration screen.

Timestamping can only be called using certificates with the quality 'HOSPITAL' OR 'LABO'.

You can configure the Timestamping credential in the Timestamping settings screen together with the input path and the archive path.

These are all mandatory.

The input path is a parent directory for sub directories 'IN', 'PROCESSING' and 'ERROR', which will be created by the system.

The files to timestamp need to be written in the IN-folder by a third party system and will be moved to the PROCESSING-folder while executing.

If the execution is successful the timestamped file will be moved to the folder specified by the archive path.

Otherwise, the file will be moved to the ERROR-folder.

The archive folder may not be a subdirectory of the input folder and vice versa the input folder may not be a subdirectory of the archive folder.

Now your Timestamping certificate is configured.

Only one timestamping certificate can be used for an installation.

This certificate can be removed if necessary by clicking on the remove configuration button.

19.2. Configure user for timestamping

In the user settings, an administrator can allow users to use the timestamping features.

The screenshot shows the 'GEBRUIKERS' (Users) tab in the MyBox settings. Under 'Rollen' (Roles), the 'Gebruiker' (User) role is selected. The 'Permissies' (Permissions) section contains a warning message: 'De gebruiker waarvoor deze permissie geactiveerd wordt moet kennis genomen hebben van de gedragsregels m.b.t. gebruik van het rijksregister, opgelegd door de overheid. Meer info kan u hier terugvinden: <https://www.ehealth.fgov.be/ehealthplatform/nl/service-webservices-consultr>'. Below this, two permissions are listed: 'Kan het rijksregister consulteren' (checked) and 'Kan timestamping consulteren' (checked and highlighted in yellow).

Now the logged in user will see the active module for Timestamping and will be able to use the timestamping features.

19.3. Scheduling timestamping files

19.3.1. File system

When a file (kmehr.xml) is transferred in the configured input folder together with the ok-file (kmehr.xml.ok or kmehr.ok). The file to timestamp will be moved to the configured processing folder and the ok-file will be removed.

19.3.2. Soap service

The list of soap services can be found here: `{{host}}/unified-messaging/services/` :

- For timestamping services for GenericDocumentServiceSoap: `{{host}}/unified-messaging/services/timestamping/GenericDocumentService?wsdl`
- For timestamping services for JournalEntryServiceSoap: `{{host}}/unified-messaging/services/timestamping/JournalEntryService?wsdl`

The GenericDocumentServiceSoap can be used to timestamp all kind of documents. The document will be hashed without interpretation and queued for timestamping. An exception is returned if something goes wrong.

The AddJournalEntry method in the JournalEntryServiceSoap can be used to timestamp KMEHR-files. The kmehr document will be validated. The metadata of the kmehr file will be retrieved and used for searching purposes in UM. The document will be hashed and queued for timestamping. An exception is returned if something goes wrong.

The AddAttachedJournalEntry method in the JournalEntryServiceSoap is not used anymore and will always throw an exception.

19.4. Processing files to timestamp

Every 10 minutes a scheduling job will gather all the files in the processing folder and send a timestamping request to eHealth.

These files will be timestamped together in a eHealth bag. The result will be one timestamp token for all the files in the bag.

19.5. Archiving timestamped files

When a bag of files has successfully been timestamped, these files will be moved to the archive folder. The archive folder consists of the archive path specified in the timestamp configuration and subfolders depending of the date and bag are created. In this case the timestamped file and the corresponding timestamp token will be moved in folder `\um\archive\yyyy\mm\dd\request_id\` resulting in `\archive\yyyy\mm\dd\request_id\example.pdf` and timestamp token file `\archive\yyyy\mm\dd\request_id\request_id`