seshat

A data and software repository system

Roel Janssen

May 30, 2026

Table of Contents

1 Introduction ... **1**
1.1 Obtaining the source code ... 1
1.2 Installing the prerequisites ... 1
1.2.1 Optional installation requirements depending on configuration ... 1
1.3 Installation instructions ... 2
1.4 RPM packages ... 2
1.5 Debian packages ... 2
2 Configuring `seshat` ... **3**
2.1 Essential options ... 3
2.2 Configuring the Database ... 4
2.3 Audit trails and database reconstruction ... 4
2.3.1 Reconstructing the database from the query audit log ... 4
2.4 Configuring storage ... 5
2.4.1 Configuring S3 buckets ... 5
2.5 Configuring an identity provider ... 6
2.5.1 SAML2.0 ... 6
2.5.2 ORCID ... 7
2.6 Configuring an e-mail server ... 8
2.7 Configuring DOI registration ... 8
2.8 Configuring Handle registration ... 8
2.9 Configuring IIIF support ... 9
2.10 Configuring OCI registry support ... 9
2.11 Customizing looks ... 9
2.11.1 Customizing colors ... 10
2.12 Configuring privileged users ... 10
3 Running `seshat` ... **12**
3.1 Running `seshat` ... 12
3.2 Running `seshat` behind an `nginx` reverse-proxy ... 12
3.3 Running `seshat` with feature isolation ... 12
3.3.1 Isolating Git operations ... 12
3.3.2 Isolating IIIF operations ... 13
4 Knowledge graph ... **14**
4.1 Use of vocabularies ... 14
4.2 Notational shortcuts ... 14
4.2.1 Notation for typed triples ... 14
4.2.2 Notation for `rdf:List` ... 14
4.3 Datasets ... 15
4.4 Collections ... 16
4.5 Authors ... 17
4.6 Accounts ... 17
4.7 Funding ... 17
4.8 Categories ... 17
4.9 Institutions/groups ... 18
4.10 Files ... 18
4.11 Private links ... 18
4.12 Collaborators ... 18
5 Contributing ... **20**
5.1 Setting up a development environment ... 20
5.2 Configuring `seshat` ... 20
5.2.1 Modifications to the example configuration for developers ... 20
5.2.2 Invoking `seshat` ... 21
5.3 Navigating the source code ... 21
5.3.1 Starting point ... 21
5.3.2 How `seshat` initializes ... 21

5.3.3 Translating URI paths to internal procedures ... 21
5.3.4 Diving into the code that displays the homepage ... 22
5.3.5 Database communication ... 22
6 Application Programming Interface ... 23
6.1 Published datasets ... 23
6.1.1 /v2/articles (GET) ... 23
6.1.2 /v2/articles/search (POST) ... 24
6.1.3 /v2/articles/<dataset-id> (GET) ... 24
6.1.4 /v2/articles/<dataset-id>/versions (GET) ... 25
6.1.5 /v2/articles/<dataset-id>/versions/<version> (GET) ... 25
6.1.6 /v2/articles/<dataset-id>/versions/<version>/embargo (GET) ... 25
6.1.7 /v2/articles/<dataset-id>/files (GET) ... 26
6.1.8 /v2/articles/<dataset-id>/files/<file-id> (GET) ... 26
6.2 Published collections ... 26
6.2.1 /v2/collections (GET) ... 26
6.2.2 /v2/collections/search (POST) ... 27
6.2.3 /v2/collections/<collection-id> (GET) ... 28
6.2.4 /v2/collections/<collection-id>/versions (GET) ... 28
6.2.5 /v2/collections/<collection-id>/versions/<version> (GET) ... 29
6.2.6 /v2/collections/<collection-id>/articles (GET) ... 29
6.3 Categories ... 30
6.3.1 /v2/categories (GET) ... 30
6.4 Licences ... 30
6.4.1 /v2/licenses (GET) ... 30
6.5 Account properties ... 31
6.5.1 /v2/account (GET) ... 31
6.6 Authors ... 31
6.6.1 /v2/account/authors/search (POST) ... 31
6.6.2 /v2/account/authors/<author-id> (GET) ... 32
6.7 Institutions ... 32
6.7.1 /v2/account/institution (GET) ... 32
6.7.2 /v2/account/institution/users/<account-id> (GET) ... 32
6.8 Git repositories ... 33
6.8.1 /v3/datasets/<git-uuid>.git (GET) ... 33
6.8.2 /v3/datasets/<git-uuid>.git/zip (GET) ... 33
6.8.3 /v3/datasets/<git-uuid>.git/languages (GET) ... 33
6.8.4 /v3/datasets/<git-uuid>.git/contributors (GET) ... 34
6.9 Creating and editing datasets ... 34
6.9.1 /v2/account/articles (GET) ... 34
6.9.2 /v2/account/articles (POST) ... 35
6.9.3 /v2/account/articles/<dataset-id> (GET) ... 36
6.9.4 /v2/account/articles/<dataset-id> (PUT) ... 36
6.9.5 /v2/account/articles/<dataset-id> (DELETE) ... 38
6.9.6 /v2/account/articles/<dataset-id>/authors (GET) ... 38
6.9.7 /v2/account/articles/<dataset-id>/authors (POST) ... 39
6.9.8 /v2/account/articles/<dataset-id>/authors (PUT) ... 39
6.9.9 /v2/account/articles/<dataset-id>/authors/<author-id> (DELETE) ... 40
6.9.10 /v2/account/articles/<dataset-id>/funding (GET) ... 40
6.9.11 /v2/account/articles/<dataset-id>/funding (POST) ... 41
6.9.12 /v2/account/articles/<dataset-id>/funding (PUT) ... 41
6.9.13 /v2/account/articles/<dataset-id>/funding/<funding-id> (DELETE) ... 41
6.9.14 /v2/account/articles/<dataset-id>/categories (GET) ... 42
6.9.15 /v2/account/articles/<dataset-id>/categories (POST) ... 42
6.9.16 /v2/account/articles/<dataset-id>/categories (PUT) ... 42
6.9.17 /v2/account/articles/<dataset-id>/categories/<category-id> (DELETE) ... 43

6.9.18 /v2/account/articles/<dataset-id>/embargo (GET) ... 43
6.9.19 /v2/account/articles/<dataset-id>/embargo (DELETE) ... 43
6.9.20 /v2/account/articles/<dataset-id>/files (GET) ... 43
6.9.21 /v2/account/articles/<dataset-id>/files (DELETE) ... 44
6.9.22 /v2/account/articles/<dataset-id>/files/<file-id> (GET) ... 44
6.9.23 /v2/account/articles/<dataset-id>/private_links (GET) ... 44
6.9.24 /v2/account/articles/<dataset-id>/private_links (POST) ... 45
6.9.25 /v2/account/articles/<dataset-id>/private_links/<link-id> (GET) ... 45
6.9.26 /v2/account/articles/<dataset-id>/private_links/<link-id> (PUT) ... 45
6.9.27 /v2/account/articles/<dataset-id>/private_links/<link-id> (DELETE) ... 46
6.9.28 /v2/account/articles/<dataset-id>/reserve_doi (POST) ... 46
6.9.29 /v2/account/articles/<dataset-id>/publish (POST) ... 46
6.10 Creating and editing collections ... 47
6.10.1 /v2/account/collections (GET) ... 47
6.10.2 /v2/account/collections (POST) ... 47
6.10.3 /v2/account/collections/<collection-id> (GET) ... 48
6.10.4 /v2/account/collections/<collection-id> (PUT) ... 49
6.10.5 /v2/account/collections/<collection-id> (DELETE) ... 50
6.10.6 /v2/account/collections/search (POST) ... 50
6.10.7 /v2/account/collections/<collection-id>/authors (GET) ... 50
6.10.8 /v2/account/collections/<collection-id>/authors (POST) ... 51
6.10.9 /v2/account/collections/<collection-id>/authors (PUT) ... 51
6.10.10 /v2/account/collections/<collection-id>/authors/<author-id> (DELETE) ... 52
6.10.11 /v2/account/collections/<collection-id>/categories (GET) ... 52
6.10.12 /v2/account/collections/<collection-id>/categories (POST) ... 53
6.10.13 /v2/account/collections/<collection-id>/categories (PUT) ... 53
6.10.14 /v2/account/collections/<collection-id>/categories/<category-id> (DELETE) ... 53
6.10.15 /v2/account/collections/<collection-id>/articles (GET) ... 53
6.10.16 /v2/account/collections/<collection-id>/articles (POST) ... 54
6.10.17 /v2/account/collections/<collection-id>/articles (PUT) ... 54
6.10.18 /v2/account/collections/<collection-id>/articles/<dataset-id> (DELETE) ... 54
6.10.19 /v2/account/collections/<collection-id>/private_links (GET) ... 54
6.10.20 /v2/account/collections/<collection-id>/private_links (POST) ... 55
6.10.21 /v2/account/collections/<collection-id>/private_links/<link-id> (GET) ... 55
6.10.22 /v2/account/collections/<collection-id>/private_links/<link-id> (PUT) ... 55
6.10.23 /v2/account/collections/<collection-id>/private_links/<link-id> (DELETE) ... 56
6.10.24 /v2/account/collections/<collection-id>/reserve_doi (POST) ... 56
6.10.25 /v2/account/collections/<collection-id>/funding (GET) ... 56
6.10.26 /v2/account/collections/<collection-id>/funding (POST) ... 57
6.10.27 /v2/account/collections/<collection-id>/funding (PUT) ... 57
6.10.28 /v2/account/collections/<collection-id>/funding/<funding-id> (DELETE) ... 57
Bibliography ... 59
Contact ... 60
The team ... 60
Mailing lists ... 60
Reporting security vulnerabilities ... 60
News ... 61
Release notes for v26.5. ... 61
New features ... 61
Bugfixes ... 61
Incremental improvements ... 61
Technical debt ... 61
Release notes for v26.4. ... 61
New features ... 61
Security ... 62

Bugfixes . . . 62
Incremental improvements . . . 62
Technical debt . . . 62
Release notes for `v25.6` . . . 62
New features . . . 63
Bugfixes . . . 63
Incremental improvements . . . 63
Release notes for `v25.5` . . . 63
UI revisions . . . 63
Security . . . 63
Bugfixes . . . 63
Technical debt . . . 63
Release notes for `v25.4` . . . 63
New features . . . 63
Security . . . 63
Bugfixes . . . 64
Technical debt . . . 64
Release notes for `v25.3` . . . 64
New features . . . 64
UI revisions . . . 64
Security . . . 64
Bugfixes . . . 64
Technical debt . . . 64
Release notes for `v25.2` . . . 64
New features . . . 65
Security . . . 65
Bugfixes . . . 65
Technical debt . . . 65
Release notes for `v25.1` . . . 65
New features . . . 65
Bugfixes . . . 65
Technical debt . . . 65
Release notes for `v24.12` . . . 66
New features . . . 66
Bugfixes . . . 66
Technical debt . . . 66

1 Introduction

`seshat` is a data and software repository system. The name finds its inspiration in Thoth, the Egyptian entity that introduced the idea of writing.

1.1 Obtaining the source code

The source code can be downloaded at the Releases[1] page. Make sure to download the `seshat-26.5.tar.gz` file.

Or, directly download the tarball using the command-line:

```
curl -LO https://codeberg.org/seshat/seshat/releases/download/v26.5/seshat-26.5.tar.gz
```

After obtaining the tarball, it can be unpacked using the `tar` command:

```
tar zxvf seshat-26.5.tar.gz
```

1.2 Installing the prerequisites

The `seshat` program needs Python (version 3.9 or higher) and Git to be installed. Additionally, a couple of Python packages need to be installed. The following sections describe installing the prerequisites on various GNU/Linux distributions. To put the software in the context of its environment, Figure 1 displays the complete run-time dependencies from `seshat` to `glibc`.

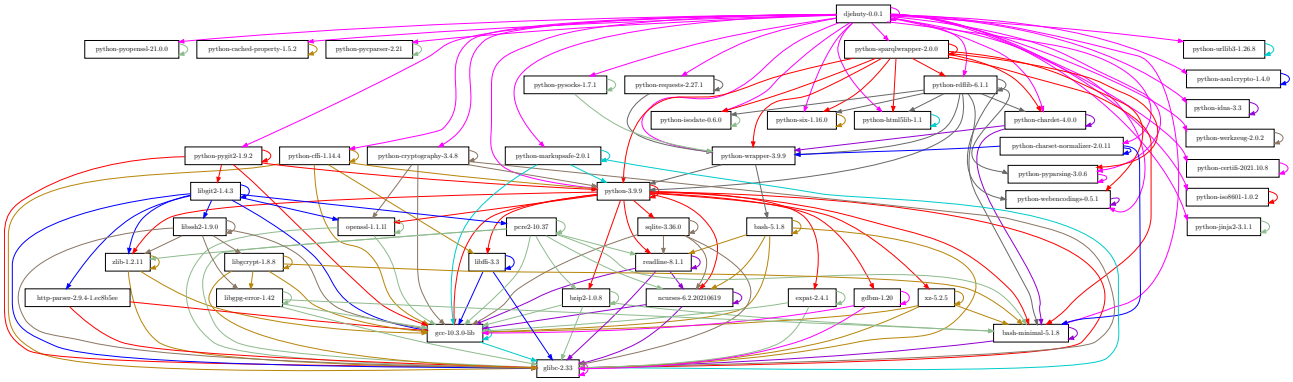

Figure 1: Run-time references when constructed with the packages from GNU Guix.

The web service of `seshat` stores its information in a SPARQL 1.1 (*SPARQL 1.1 Overview*, 2013) endpoint. We recommend either Blazegraph[2] or Virtuoso open-source edition[3].

1.2.1 Optional installation requirements depending on configuration

For specific features `seshat` may require additional packages to be installed. Whether this is the case depends on the run-time configuration. When an optional package is required `seshat` will report which one in its logs. There are three configuration scenarios that require the additional packages: SAML, S3 and IIIF.

1.2.1.1 SAML

When configuring the use of an identity provider via SAML `seshat` requires the `python3-saml` Python package to be installed. This package provides the implementation of the SAML protocol.

1.2.1.2 S3

When configuring file access in S3 buckets `seshat` requires the `boto3` Python package to be installed. This package is used to authenticate to the S3 endpoints and to download (or stream) data.

[1] https://codeberg.org/seshat/seshat/releases
[2] https://blazegraph.com/
[3] http://vos.openlinksw.com/owiki/wiki/VOS

1.2.1.3 IIIF

When enabling the IIIF functionality `seshat` requires the `pyvips` Python package to be installed. This package is used to perform image transformations.

1.3 Installation instructions

After obtaining the source code (see Section 1.1) and installing the required tools (see Section 1.2), building involves running the following commands:

```
cd seshat-26.5
autoreconf -vif # Only needed if the "./configure" step does not work.
./configure
make
make install
```

To run the `make install` command, super user privileges may be required. Specify a `--prefix` to the `configure` script to install the tools to a user-writeable location to avoid needing super user privileges.

After installation, the `seshat` program will be available.

1.4 RPM packages

RPM packages are provided and built for Enterprise Linux 10. This RPM depends on packages in the Extra Packages for Enterprise Linux (EPEL) repository.

Filename	Description
seshat-26.5–1.el10.noarch.rpm	Binary RPM, to install and run `seshat`.
seshat-26.5–1.el10.src.rpm	Source RPM, to (re)build from source code.

RPM packages for more distributions, including Enterprise Linux 9, are built via Copr.

1.5 Debian packages

A Debian package is provided and built for Debian 13.

Filename	Description
seshat_26.5–1_all.deb	Binary DEB, to install and run `seshat`.

2 Configuring seshat

Now that seshat is installed, it's a good moment to look into its run-time configuration options. All configuration can be done through a configuration file, for which an example is available at etc/seshat.xml.

2.1 Essential options

Option	Description
bind-address	The address to bind a TCP socket on.
port	The port to bind a TCP socket on.
alternative-port	A fall-back port to bind on when port is already in use.
base-url	The URL on which the instance will be available to the outside world.
documentation-url	The URL on which the documentation will be available to the outside world.
ontology-url	The URL for the internal ontology.
allow-crawlers	Set to 1 to allow crawlers in the robots.txt, otherwise set to 0.
production	Performs extra checks before starting. Enable this when running a production instance.
live-reload	When set to 1, it reloads Python code on-the-fly. We recommend to set it to 0 when running in production.
debug-mode	When set to 1, it will display backtraces and error messages in the web browser. When set to 0, it will only show backtraces and error messages in the web browser.
use-x-forwarded-for	When running seshat behind a reverse-proxy server, use the HTTP header X-Forwarded-For to log IP address information. Set to 1 when seshat should use the X-Forwarded-For HTTP header.
static-resources-cache	When running seshat behind a reverse-proxy server, it can write images, fonts, stylesheets and JavaScript resources to a folder so it can be served by the reverse-proxy server. Specify a filesystem directory to store the resources at.
disable-collaboration	When set to 1, it disables the "collaborators" feature.
allowed-depositing-domains	When unset, any authenticated user may deposit data. Otherwise, this option limits the ability to deposit to users with an e-mail address of the listed domain names.
cache-root	seshat can cache query responses to lower the load on the database server. Specify the directory where to store cache files. This element takes an attribute clear-on-start, and when set to 1, it will remove all cache files on start-up of seshat.
profile-images-root	Users can upload a profile image in seshat. This option should point to a filesystem directory where these profile images can be stored.
disable-2fa	Accounts with privileges receive a code by e-mail as a second factor when logging in. Setting this option to 1 disables the second factor authentication.
disable-account-creation	When set to 1, new accounts cannot be created, effectively disabling new users from signing up. The default is 0.
sandbox-message	Display a message on the top of every page.
notice-message	Display a message on the main page.

Option	Description
`maintenance-mode`	When set to 1, all HTTP requests result in the displayment of a maintenance message. Use this option while backing up the database, or when performing major updates.

2.2 Configuring the Database

The `seshat` program stores its state in a SPARQL 1.1 compliant RDF store. Configuring the connection details is done in the `rdf-store` node.

Option	Description
`state-graph`	The graph name to store triplets in.
`sparql-uri`	The URI at which the SPARQL 1.1 endpoint can be reached. When the `sparql-uri` begins with `bdb://`, followed by a path to a filesystem directory, it will use the BerkeleyDB back-end, for which the `berkeleydb` Python package needs to be installed.
`sparql-update-uri`	The URI at which the SPARQL 1.1 Update endpoint can be reached (in case it is different from the `sparql-uri`).
`seconds-to-wait-for-online`	Number of seconds to retry in case the SPARQL endpoint is down when initializing. A retry is done every second, and after the configured amount, `seshat` will give up.
`read-only-mode`	When set to `1`, no queries that change the state of the database shall be made. This is useful for running secondary instances to balance the load of specific endpoints. Defaults to `0`.

2.3 Audit trails and database reconstruction

The `seshat` program can keep an audit log of all database modifications made by itself from which a database state can be reconstructed. Whether `seshat` keeps such an audit log can be configured with the `enable-query-audit-log` configuration option.

The database can receive a large amount of requests due to inserting log entries to calculate views and downloads statistics from. The configuration option `delay-inserting-log-entries` can be used to prevent these queries from being sent to the database server. To maintain accurate statistics, these log entry queries must be replayed in due time.

Option	Description
`delay-inserting-log-entries`	When set to 1, it writes the `INSERT` queries for log entries only to the audit log, rather than executing them on the SPARQL endpoint.
`enable-query-audit-log`	When set to 1, it writes every SPARQL query that modifies the database in the web logs. This can be replayed to reconstruct the database at a later time. Setting this option to 0 disables this feature. This element takes an attribute `transactions-directory` that should specify an empty directory to which transactions can be written that are extracted from the audit log.

2.3.1 Reconstructing the database from the query audit log

Each query that modifies the database state while the query audit logs are enabled can be extracted from the query audit log using the `--extract-transactions-from-log` command-line option. A timestamp to specify the starting point to extract from can be specified as an argument. The following example displays its use:

```
seshat web --config-file=config.xml --extract-transactions-from-log="YYYY-MM-DD HH:MM:SS"
```

This will create a file for each query in the folder specified in the `transactions-directory` attribute.

To replay the extracted transactions, use the `--apply-transactions` command-line option:

```
seshat web --config-file=config.xml --apply-transactions
```

When a query cannot be executed, the command stops, allowing to fix or remove the query to-be-replayed. Invoking the `--apply-transactions` command a second time will continue replaying where the previous run stopped.

If the `delay-inserting-log-entries` configuration option is set to 1, use the `--extract-delayed-transactions-from-log` command-line option to write the log entries queries to the transactions-directory. Then re-run `--apply-transactions` to apply the log entries queries.

2.4 Configuring storage

Storage locations can be configured with the `storage` node. When configuring multiple locations, `seshat` attempts to find a file by looking at the first configured location, and in case it cannot find the file there, it will look at the second configured location, and so on, until it has tried each storage location.

This allows for moving files between storage systems transparently without requiring specific interactions with `seshat` other than having the files made available as a POSIX filesystem or in an S3 bucket.

One use-case that suits this mechanism is letting uploads write to fast online storage and later move the uploaded files to a slower but less costly storage.

Option	Description
`location`	A filesystem path to where files are stored. This is a repeatable property.
`s3-bucket`	An S3 bucket configuration. See Section 2.4.1. This is a repeatable property.

2.4.1 Configuring S3 buckets

Other than configuring storage locations on a POSIX filesystem, `seshat` can be configured to serve files from an S3 bucket. To do so, the following parameters must be configured within a `s3-bucket` node.

Option	Description
`endpoint`	Endpoint URL to connect to.
`name`	Name of the bucket.
`key-id`	Key ID for the bucket.
`secret-key`	Secret key for the bucket.

For example, configuring one filesystem location and one S3 bucket as storage locations looks as following:

```
<storage>
  <location>/data</location>
  <s3-bucket>
    <endpoint>https://some.example</endpoint>
    <n>example-bucket</n>
    <key-id>...</key-id>
    <secret-key>...</secret-key>
  </s3-bucket>
</storage>
```

There are a few scenarios in which `seshat` downloads an S3 object to perform some operation on: creating thumbnails and IIIF image transformations. To direct where these temporary files will be stored, the `s3-cache-root` can be configured.

Option	Description
`s3-cache-root`	The directory to store the S3 objects while performing some operation on the objects. This option can only be configured globally and applies to all S3 buckets.

2.5 Configuring an identity provider

Ideally, `seshat` makes use of an external identity provider. `seshat` can use SAML2.0, ORCID, or an internal identity provider (for testing and development purposes only).

This section will outline the configuration options for each identity provision mechanism.

2.5.1 SAML2.0

For SAML 2.0, the configuration can be placed in the `saml` section under `authentication`. That looks as following:

```
<authentication>
  <saml version="2.0">
    <!-- Configuration goes here. -->
  </saml>
</authentication>
```

The options outlined in the remainder of this section should be placed where the example shows `<!-- Configuration goes here. -->`.

Option	Description
`strict`	When set to 1, SAML responses must be signed. **Never disable 'strict' mode in a production environment.**
`debug`	Increases logging verbosity for SAML-related messages.
`attributes`	In this section the attributes provided by the identity provider can be aligned to the attributes `seshat` expects.
`service-provider`	The `seshat` program fulfills the role of service provider. In this section the certificate and service provider metadata can be configured.
`identity-provider`	In this section the certificate and single-sign-on URL of the identity provider can be configured.
`sram`	In this section, SURF Research Access Management-specific attributes can be configured.

2.5.1.1 The `attributes` configuration

To create account and author records and to authenticate a user, `seshat` stores information provided by the identity provider. Each identity provider may provide this information using different attributes. Therefore, the translation from attributes used by `seshat` and attributes given by the identity provider can be configured. The following attributes must be configured.

Option	Description
`first-name`	A user's first name.
`last-name`	A user's last name.
`common-name`	A user's full name.
`email`	A user's e-mail address.
`groups`	The attribute denoting groups.
`group-prefix`	The prefix for each group short name.

As an example, the attributes configuration for SURFConext looks like this:

```
<attributes>
  <first-name>urn:mace:dir:attribute-def:givenName</first-name>
  <last-name>urn:mace:dir:attribute-def:sn</last-name>
  <common-name>urn:mace:dir:attribute-def:cn</common-name>
  <email>urn:mace:dir:attribute-def:mail</email>
</attributes>
```

And for SURF Research Access Management (SRAM), the attributes configuration looks like this:

```
<attributes>
  <first-name>urn:oid:2.5.4.42</first-name>
  <last-name>urn:oid:2.5.4.4</last-name>
  <common-name>urn:oid:2.5.4.3</common-name>
  <email>urn:oid:0.9.2342.19200300.100.1.3</email>
  <groups>urn:oid:1.3.6.1.4.1.5923.1.1.1.7</groups>
  <group-prefix>urn:mace:surf.nl:sram:group:[organisation]:[service]</group-prefix>
</attributes>
```

2.5.1.2 The `sram` configuration

When using SURF Research Access Management (SRAM), `seshat` can persuade SRAM to send an invitation to anyone inside or outside the institution to join the SRAM collaboration that provides access to the `seshat` instance. To do so, the following attributes must be configured.

Option	Description
`organization-api-token`	An organization-level API token.
`collaboration-id`	The UUID of the collaboration to invite users to.

2.5.1.3 The `service-provider` configuration

Option	Description
`x509-certificate`	Contents of the public certificate without whitespacing.
`private-key`	Contents of the private key belonging to the `x509-certificate` to sign messages with.
`metadata`	This section contains metadata that may be displayed by the identity provider to users before authorizing them.
`display-name`	The name to be displayed by the identity provider when authorizing the user to the service.
`url`	The URL to the service.
`description`	Textual description of the service.
`organization`	This section contains metadata to describe the organization behind the service.
`name`	The name of the service provider's organization.
`url`	The URL to the web page of the organization.
`contact`	A repeatable section to list contact persons and their roles within the organization. The role can be configured by setting the `type` attribute.
`first-name`	The first name of the contact person.
`last-name`	The last name of the contact person.
`email`	The e-mail address of the contact person. Note that some identity providers prefer functional e-mail addresses (e.g. support@… instead of jdoe@…).

2.5.2 ORCID

ORCID.org plays a key role in making researchers findable. Its identity provider service can be used by `seshat` in two ways:

1. As primary identity provider to log in and deposit data;
2. As additional identity provider to couple an author record to its ORCID record.

When another identity provider is configured in addition to ORCID, that identity provider will be used as primary and ORCID will only be used to couple author records to the author's ORCID record.

To configure ORCID, the configuration can be placed in the `orcid` section under `authentication`. That looks as following:

```
<authentication>
  <orcid>
    <!-- Configuration goes here. -->
  </orcid>
</authentication>
```

Then the following parameters can be configured:

Option	Description
`client-id`	The client ID provided by ORCID.
`client-secret`	The client secret provided by ORCID.
`endpoint`	The URL to the ORCID endpoint to use.

2.6 Configuring an e-mail server

On various occassions, `seshat` will attempt to send an e-mail to either an author, a reviewer or an administrator. To be able to do so, an e-mail server must be configured from which the instance may send e-mails.

The configuration is done under the `email` node, and the following items can be configured:

Option	Description
`server`	Address of the e-mail server without protocol specification.
`port`	The port the e-mail server operates on.
`starttls`	When 1, `seshat` attempts to use StartTLS.
`username`	The username to authenticate with to the e-mail server.
`password`	The password to authenticate with to the e-mail server.
`from`	The e-mail address used to send e-mail from.
`subject-prefix`	Text to prefix in the subject of all e-mails sent from the instance of `seshat`. This can be used to distinguish a test instance from a production instance.

2.7 Configuring DOI registration

When publishing a dataset or collection, `seshat` can register a persistent identifier with DataCite. To enable this feature, configure it under the `datacite` node. The following parameters can be configured:

Option	Description
`api-url`	The URL of the API endpoint of DataCite.
`repository-id`	The repository identifier given by DataCite.
`password`	The password to authenticate with to DataCite.
`prefix`	The DOI prefix to use when registering a DOI.

2.8 Configuring Handle registration

Each uploaded file can be assigned a persistent identifier using the Handle system. To enable this feature, configure it under the `handle` node. The following parameters can be configured:

Option	Description
`url`	The URL of the API endpoint of the Handle system implementor.
`certificate`	Certificate to use for authenticating to the endpoint.
`private-key`	The private key paired with the certificate used to authenticate to the endpoint.
`prefix`	The Handle prefix to use when registering a handle.
`index`	The index to use when registering a handle.

2.9 Configuring IIIF support

When publishing images, `seshat` can enable the IIIF Image API for the images. It uses `libvips` and `pyvips` under the hood to perform image manipulation. The following parameters can be configured:

Option	Description
`enable-iiif`	Enable support for the IIIF image API. This requires the `pyvips` package to be available in the run-time environment.
`iiif-cache-root`	The directory to store the output of IIIF Image API requests to avoid re-computing the image.

2.10 Configuring OCI registry support

`seshat` has built-in support for the OCI registry v2 API. The following configuration options are available:

Option	Description
`enable-oci-registry`	Enable support for the built-in OCI registry (defaults to `0`.
`oci-registry-root`	The directory to store the container images in.

2.11 Customizing looks

With the following options, the instance can be branded as necessary.

Option	Description
`site-name`	Name for the instance used in the title of a browser window and as default value in the publisher field for new datasets.
`site-description`	Description used as a meta-tag in the HTML output.
`site-shorttag`	Used as keyword and as Git remote name.
`ror-url`	The ROR URL for this instance's organization.
`support-email-address`	E-mail address used in e-mails sent to users in automated messages.
`custom-logo-path`	Path to a PNG image file that will be used as logo on the website.
`custom-favicon-path`	Path to an ICO file that will be used as favicon.
`small-footer`	HTML that will be used as footer for all pages except for the main page.
`large-footer`	HTML that will be used as footer on the main page.
`show-portal-summary`	When set to 1, it shows the repository summary of number of datasets, authors, collections, files and bytes on the main page.
`show-latest-datasets`	When set to 1, it shows the list of latest published datasets on the main page.
`colors`	Colors used in the HTML output. See Section 2.11.1.

2.11.1 Customizing colors

The following options can be configured in the `colors` section.

Option	Description
`primary-color`	The main background color to use.
`primary-foreground-color`	The main foreground color to use.
`primary-color-hover`	Color to use when hovering a link.
`primary-color-active`	Color to use when a link is clicked.
`privilege-button-color`	The background color of buttons for privileged actions.
`footer-background-color`	Color to use in the footer.
`background-color`	Background color for the content section.

2.12 Configuring privileged users

By default an authenticated user may deposit data. But users can have additional roles; for example: a dataset reviewer, a technical administrator or a quota reviewer.

Such additional roles are configured in terms of privileges. The following privileges can be configured in the `privileges` section:

Option	Description
`may-administer`	Allows access to perform maintenance tasks, view accounts and view reports on restricted and embargoed datasets.
`may-run-sparql-queries`	Allows to run arbitrary SPARQL queries on the database.
`may-impersonate`	Allows to log in to any account and therefore perform any action as that account.
`may-review`	Allows to see which datasets are sent for review, and allows to perform reviews.
`may-review-quotas`	Allows access to see requests for storage quota increases and approve or decline them.
`may-review-integrity`	Allows access to an API call that provides statistics on the accessibility of files on the filesystem.
`may-process-feedback`	Accounts with this privilege will receive e-mails with the information entered into the feedback form by other users.
`may-recalculate-statistics`	Views and downloads statistics are not calculated in real time. An administrator with this additional privilege can trigger a recalculation of these statistics, which can be a database-intensive action.
`may-receive-email-notifications`	This "privilege" can be used to disable sending any e-mails to an account by setting it to `0`. The default is `1`.

To enable a privilege for an account, set the value of the desired privilege to `1`. Privileges are disabled by default, except for `may-receive-email-notifications` which defaults to `1`.

```
<privileges>
  <account email="you@example.com" orcid="0000-0000-0000-0001">
    <may-administer>1</may-administer>
    <may-run-sparql-queries>1</may-run-sparql-queries>
    <may-impersonate>1</may-impersonate>
    <may-review>0</may-review>
    <may-review-quotas>0</may-review-quotas>
    <may-review-integrity>0</may-review-integrity>
```

```
    <may-process-feedback>0</may-process-feedback>
    <may-receive-email-notifications>1</may-receive-email-notifications>
  </account>
</privileges>
```

3 Running seshat

Before running seshat, consider Section 2 which provides the configuration options to enable or disable features, where data will be stored and a way to adapt seshat to your organization's style.

3.1 Running seshat

Invoking seshat web starts the web interface of seshat. On what port it makes itself available can be configured in its configuration file.

```
seshat web --config-file=your-seshat-config.xml
```

3.2 Running seshat behind an nginx reverse-proxy

While seshat itself does not support SSL/TLS, it is designed to work together with a reverse-proxy HTTP server like nginx. When seshat starts, it will bind on a pre-configured address and port, which in turn can be proxy_passed to using nginx.

The following snippet shows how to configure nginx.

```
server {
    listen              443 ssl;
    listen              [::]:443 ssl;
    server_name         example.domain;

    ssl_certificate     /etc/letsencrypt/live/example.domain/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/example.domain/privkey.pem;

    location / {
       # Set 'use-x-forwarded-for' in the seshat configuration.
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
       # The values for address and port depend on what is configured in the
       # seshat configuration file.
       proxy_pass http://127.0.0.1:8080;
       root /usr/share/nginx/html;
    }
}
```

To ensure seshat receives the actual client IP address so it can log this information, one can set the use-x-forwarded-for option described in Section 2.1.

3.3 Running seshat with feature isolation

Some features are more prone to denial-of-service situations because they require more CPU, memory, or disk space to complete. To mitigate outages while using these features, multiple instances of seshat can run simultaneous as long as all but one instance runs in read-only-mode in the rdf-store section.

Only one instance may write to the database, but other instances can be useful to handle specific endpoints (like IIIF image processing or Git operations). The read-only instances need to be able to query the database, but need not to write to it. They can run on completely different machines, as long as a database connection can be established.

3.3.1 Isolating Git operations

To take advantage of the multi-instance deployment, we are going to use the nginx configuration file to let a second instance of seshat handle all Git operations by defining multiple location blocks in the nginx configuration.

So we create two configuration files, differing only in the port, log-file and rdf-store/read-only-mode options.

- seshat-primary.xml: Set the rdf-store/read-only-mode to 0 and port to 8080.
- seshat-git.xml: Set the rdf-store/read-only-mode to 1 and port to 8081.

Setting two different log files makes it easier to track which request goes where.

With both instances of `seshat` running (one with the 'primary' and one 'git' configuration file), the accompanying `nginx` `server`-block configuration looks as following:

```
server {
    listen              443 ssl;
    listen              [::]:443 ssl;
    server_name         example.domain;

    ssl_certificate     /etc/letsencrypt/live/example.domain/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/example.domain/privkey.pem;

    # Here we group the endpoints that are used to push/pull with Git.
    # These requests will then use an instance bound to port 8081.
    location ~* ^/v3/datasets/(.*).git/(info/refs|git-upload-pack|git-receive-pack)$ {
       root /usr/share/nginx/html;
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
       proxy_pass http://127.0.0.1:8081;
    }

    # Additionally, we group the endpoints that have a Git interaction.
    # You can also concatenate the last part of the regexp to the previous
    # location block.
    location ~* ^/v3/datasets/(.*).git/(languages|contributors|zip)$ {
       root /usr/share/nginx/html;
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
       proxy_pass http://127.0.0.1:8081;
    }

    # All other locations use the instance of Seshat bound to port 8080.
    location / {
       root /usr/share/nginx/html;
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
       proxy_pass http://127.0.0.1:8080;
     }
}
```

3.3.2 Isolating IIIF operations

The IIIF endpoints can be captured by the following regular expression for the `location` block in `nginx`:

```
^/iiif/v3/(.*)$
```

The rest of the instructions are identical to isolating Git requests (see Section 3.3.1).

4 Knowledge graph

Seshat processes its information using the Resource Description Framework (Lassila, 1999). This chapter describes the parts that make up the data model of `seshat`.

This chapter dives into the structure of the data model, but does not describe every property. When running an instance of `seshat`, the "Exploratory" available in the "Admin panel" can be used to explore every property.

4.1 Use of vocabularies

Throughout this chapter, abbreviated references to ontologies are used. Table 1 lists these abbreviations.

Abbreviation	Ontology URI
`shst`	Internal and unpublished ontology.
`rdf`	http://www.w3.org/1999/02/22-rdf-syntax-ns#
`rdfs`	http://www.w3.org/2000/01/rdf-schema#
`xsd`	http://www.w3.org/2001/XMLSchema#

Table 1: Lookup table for vocabulary URIs and their abbreviations.

4.2 Notational shortcuts

In addition to abbreviating ontologies with their prefix we use another notational shortcut. To effectively communicate the structure of the RDF graph used by `seshat` we introduce a couple of shorthand notations.

4.2.1 Notation for typed triples

When the `object` in a triple is *typed*, we introduce the shorthand to only show the type, rather than the actual value of the `object`. Figure 2 displays this for URIs, and Figure 3 displays this for literals.

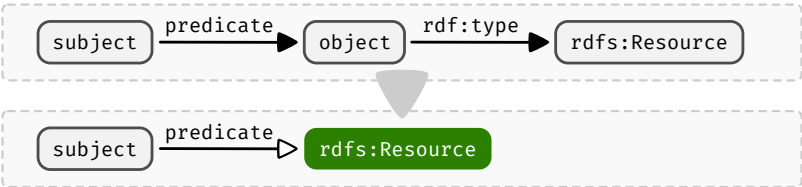

Figure 2: Shorthand notation for triples with an `rdf:type` which features a hollow predicate arrow and a colored type specifier with rounded corners.

Literals are depicted by rectangles (with sharp edges) in contrast to URIs which are depicted as rectangles with rounded edges.

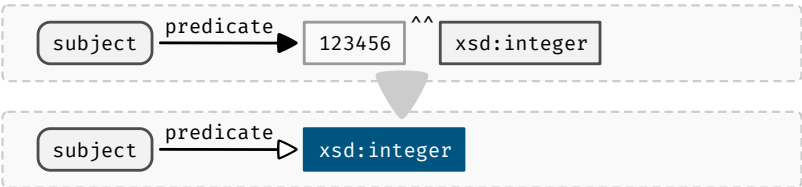

Figure 3: Shorthand notation for triples with a literal, which features a hollow predicate arrow and a colored rectangular type specifier.

When the subject of a triple is the shorthand type, assume the subject is not the type itself but the subject which has that type.

4.2.2 Notation for `rdf:List`

To preserve the order in which lists were formed, the data model makes use of `rdf:List` with numeric indexes. This pattern will be abbreviated in the remainder of the figures as displayed in Figure 4.

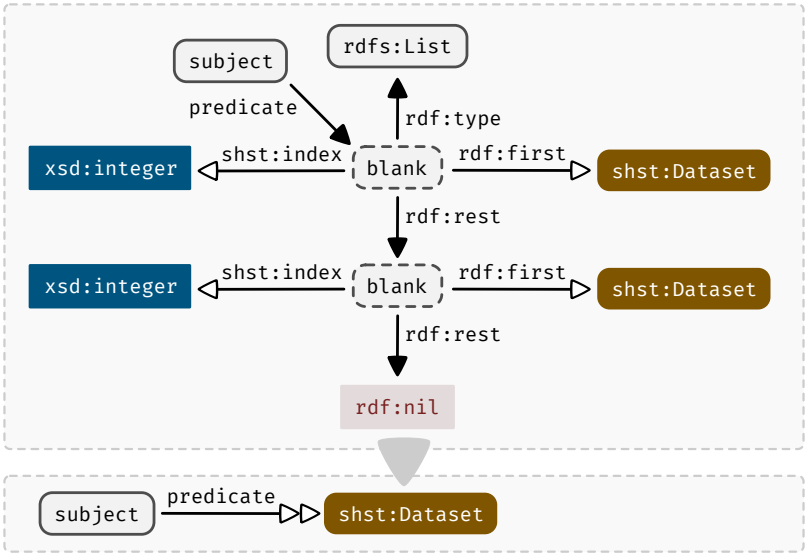

Figure 4: Shorthand notation for `rdf:List` with numeric indexes, which features a hollow double-arrow. Lists have arbitrary lengths, and the numeric indexes use 1-based indexing.

The hollow double-arrow depicts the use of an `rdf:List` with numeric indexes.

4.3 Datasets

Datasets play a central role in the repository system because every other type links in one way or another to it. The user submits files along with data about those bytes as a single record which we call a `shst:Dataset`. Figure 5 shows how the remainder of types in this chapter relate to a `shst:Dataset`.

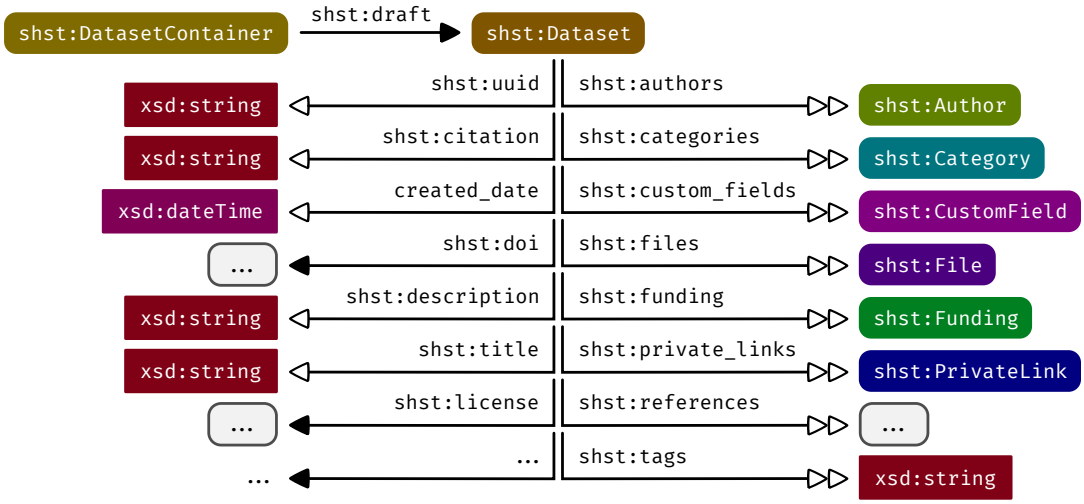

Figure 5: The RDF pattern for a `shst:Dataset`. For a full overview of `shst:Dataset` properties, use the exploratory from the administration panel.

Datasets are versioned records. The data and metadata between versions can differ, except all versions of a dataset share an identifier. We use `shst:DatasetContainer` to describe the version-unspecific properties of a set of versioned datasets.

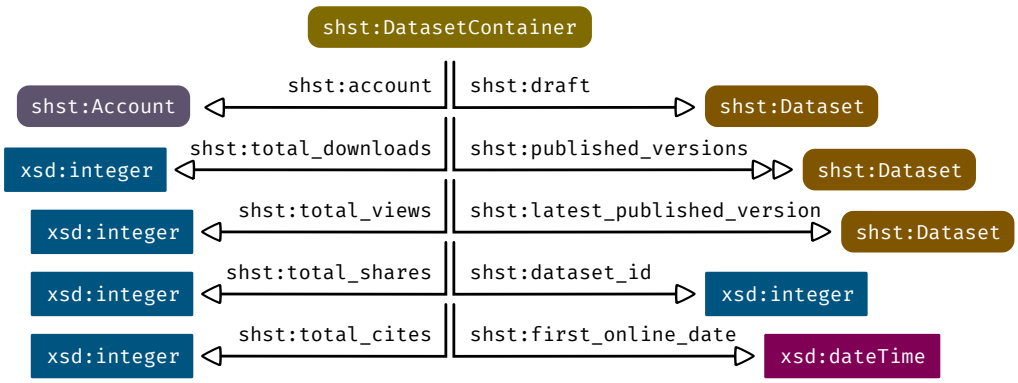

Figure 6: The RDF pattern for a `shst:DatasetContainer`. All versions of a dataset share a `shst:dataset_id` and a UUID in the container URI.

The data model follows a natural expression of published versions as a linked list. Figure 6 further reveals that the *view*, *download*, *share* and *citation* counts are stored in a version-unspecific way.

4.4 Collections

Collections provide a way to group `shst:Dataset` objects.

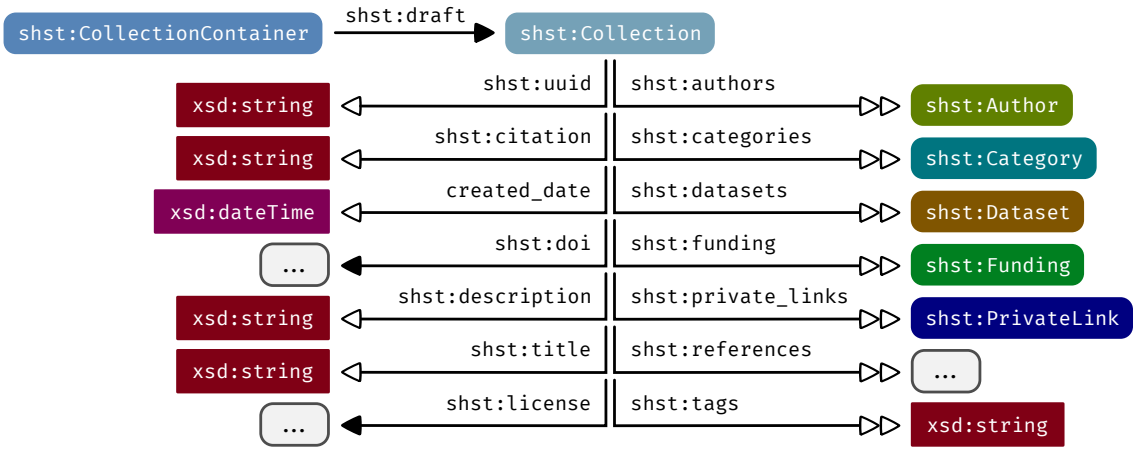

Figure 7: The RDF pattern for a `shst:Collection`. For a full overview of `shst:Collection` properties, use the exploratory from the administration panel.

Collections are (just like Datasets) versioned records. The metadata between versions can differ, except all versions of a collection share an identifier. We use `shst:CollectionContainer` to describe the version-unspecific properties of a set of versioned collections.

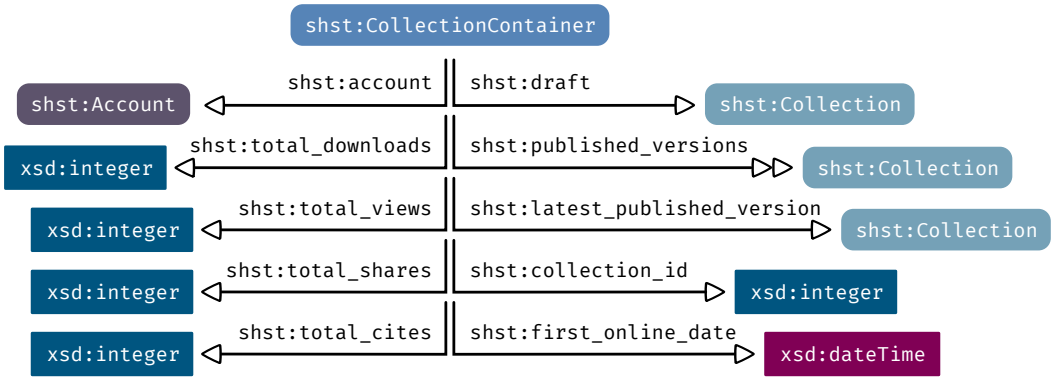

Figure 8: The RDF pattern for a `shst:CollectionContainer`. All versions of a collection share a `shst:collection_id` and a UUID in the container URI.

The data model follows a natural expression of published versions as a linked list. Figure 8 further reveals that the *view*, *download*, *share* and *citation* counts are stored in a version-unspecific way.

4.5 Authors

`seshat` keeps records of authors including their full name, ORCID, and e-mail address. Furthermore, each `shst:Account` has a linked `shst:Author` record.

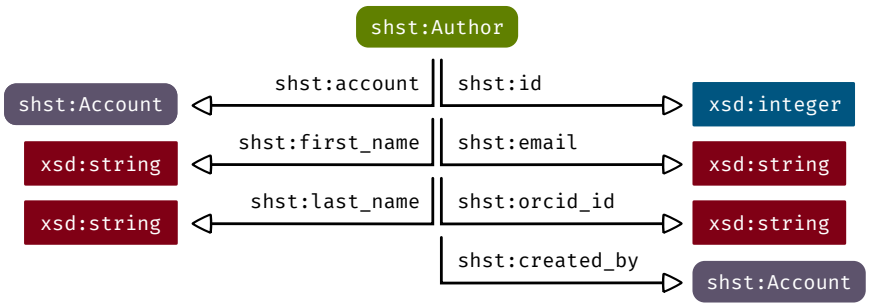

Figure 9: The RDF pattern for a `shst:Author`.

4.6 Accounts

`seshat` uses an external identity provider, but stores an e-mail address, full name, and preferences for categories.

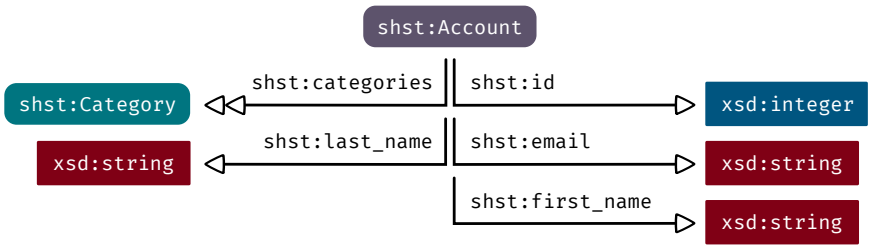

Figure 10: The RDF pattern for a `shst:Account`.

4.7 Funding

When the `shst:Dataset` originated out of a funded project, the funders can be listed using `shst:Funding`. Figure 11 displays the details for this structure.

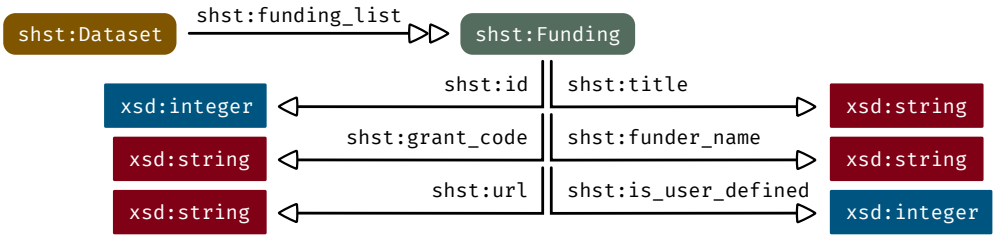

Figure 11: The RDF pattern for a `shst:Funding`.

4.8 Categories

Categories in `seshat` are a controlled vocabulary based on the Australian and New Zealand Standard Research Classification (ANZSRC). The hierarchical structure is captured by using `id` and `parent_id` properties.

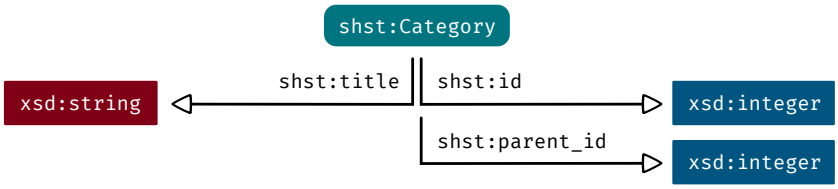

Figure 12: The RDF pattern for a `shst:Category`.

4.9 Institutions/groups

A `shst:Account` has an affiliation with an institute or research group. The `shst:InstitutionGroup` is stored per `shst:Dataset` and `shst:Collection`. The groups can be structured hierarchically by using the `id` and `parent_id` properties.

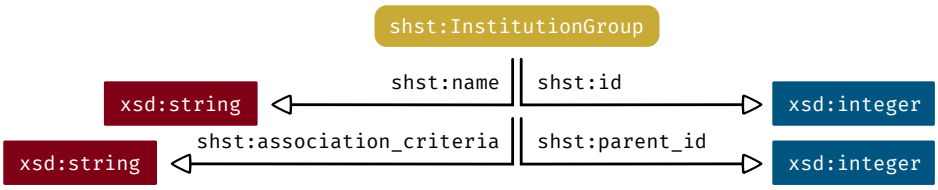

Figure 13: The RDF pattern for a `shst:InstitutionGroup`.

4.10 Files

A `shst:Dataset` keeps a list of `shst:File` records. The file metadata is stored in the knowledge graph while the file contents are stored on a filesystem. The location of the file data is tracked via the `shst:filesystem_location` property.

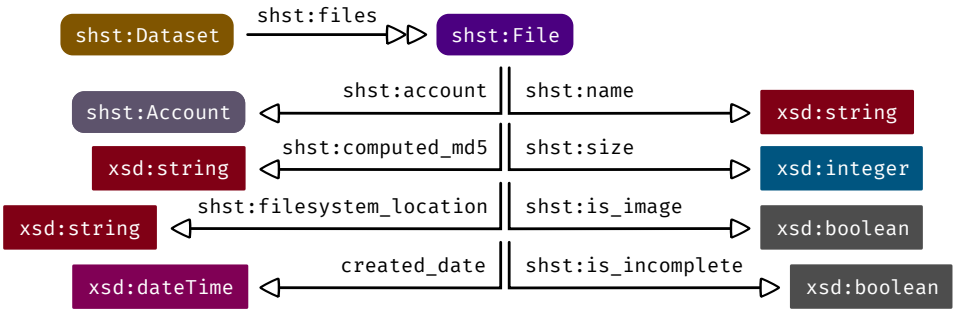

Figure 14: The RDF pattern for a `shst:File`.

4.11 Private links

Before a `shst:Dataset` or a `shst:Collection` is made publically available, it can be shared using a private link.

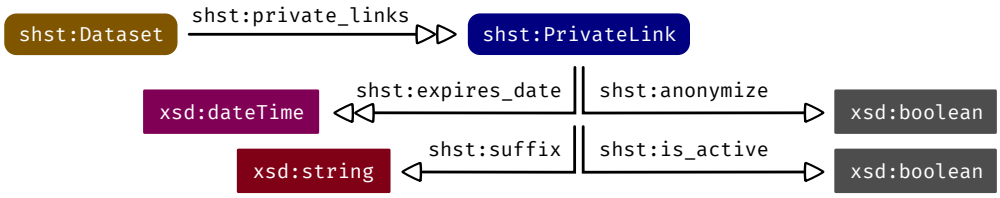

Figure 15: The RDF pattern for a `shst:PrivateLink`.

Figure 15 displays how private links are stored for a `shst:Dataset`, and it works the same for a `shst:Collection`.

4.12 Collaborators

To enable multiple accounts collaborating on a dataset before it's published, each `shst:Dataset` can have a list of `shst:Collaborator` objects.

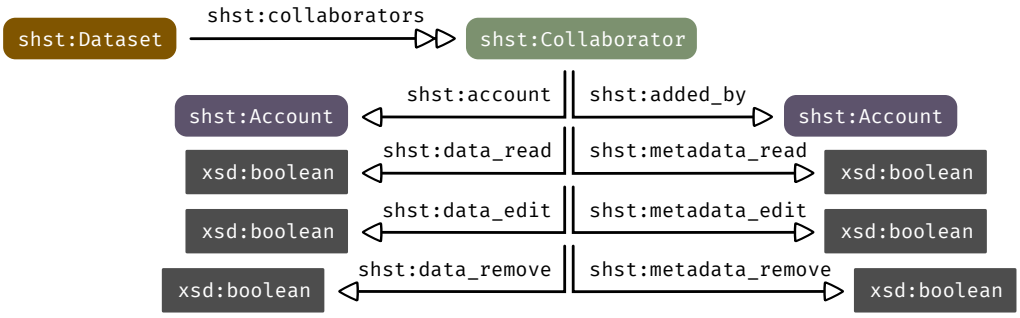

Figure 16: The RDF pattern for a `shst:Collaborator`.

As shown in Figure 16, a `shst:Collaborator` can be given read, edit, and/or remove rights independently for both metadata (the form fields) and data (the files).

5 Contributing

This chapter outlines how to set up an instance of `seshat` with the goal of modifying its source code. Or in other words: this is the developer setup.

5.1 Setting up a development environment

First, we need to obtain the latest version of the source code:

```
$ git clone https://codeberg.org/seshat/seshat
```

Next, we need to create a somewhat isolated Python environment:

```
$ python -m venv seshat-env
$ . seshat-env/bin/activate
[env]$ cd seshat
```

And finally, we can install `seshat` in the virtual environment to make the `seshat` command available:

```
[env]$ autoreconf -if && ./configure
[env]$ pip install --editable .
```

If all went well, we will now be able to run `seshat`:

```
[env]$ seshat --help
```

5.2 Configuring `seshat`

Invoking `seshat web` starts the web interface of `seshat`. On what port it makes itself available can be configured in its configuration file. An example of a configuration file can be found in `etc/seshat.xml`. We will use this file as the basis to configure our development environment.

```
[env]$ cp etc/seshat.xml config.xml
```

In the remainder of the chapter we will assume a value of `127.0.0.1` for `bind-address` and a value of `8080` for `port`.

5.2.1 Modifications to the example configuration for developers

Section 2 describes each configuration option for `seshat`. The remainder of sections here contain a fast-path through configuring `seshat` for use in a development setup.

5.2.1.1 Live reload

The `seshat` program can be configured to automatically reload itself when a change is detected by setting `live-reload` to `1`.

5.2.1.2 Configuring authentication with ORCID

The `seshat` program does not have Identity Provider (IdP) capabilities, so in order to log into the system we must configure an external IdP. With an ORCID account comes the ability to set up an OAuth endpoint. Go to developer-tools at orcid.org. When setting up the OAuth at ORCID, choose `http://127.0.0.1:8080/login` as redirect URI.

Modify the following bits to reflect the settings obtained from ORCID.

```
<authentication>
  <orcid>
    <client-id>APP-XXXXXXXXXXXXXXXX</client-id>
    <client-secret>XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX</client-secret>
    <endpoint>https://orcid.org/oauth</endpoint>
  </orcid>
</authentication>
```

To limit who can log into a development system, accounts are not automatically created for ORCID as IdP. So we need to configure who can log in by creating a record in the `privileges` section of the configuration file.

This is also a good moment to configure additional privileges for your account. In the following snippet, configure the ORCID with which you will log into the system in the `orcid` argument.

```
<privileges>
  <account email="you@example.com" orcid="0000-0000-0000-0001">
    <may-administer>1</may-administer>
    <may-impersonate>1</may-impersonate>
    <may-review>1</may-review>
  </account>
</privileges>
```

5.2.2 Invoking `seshat`

Once we've configured `seshat` for development use, we can start the web interface by running:

```
[env]$ seshat web --initialize --config-file=config.xml
```

The `--initialize` option creates the internal account record and associates the specified ORCID with it. We only need to run `seshat` with the `--initialize` option once.

By now, we should be able to visit `seshat` through a web browser at localhost:8080, unless configured differently. We should be able to log in through ORCID, and access all features of `seshat`.

5.3 Navigating the source code

In this section, we trace the path from invoking `seshat` to responding to a HTTP request.

5.3.1 Starting point

Because `seshat` is installable as a Python package, we can find the starting point for running `seshat` in `pyproject.toml`. It reads:

```
[project.scripts]
seshat = seshat.ui:main
```

So, we start our tour at `src/seshat/ui.py` in the procedure called `main`.

5.3.2 How `seshat` initializes

The `main` procedure calls `main_inner`, which handles the command-line arguments. When invoking `seshat`, we usually invoke `seshat *web*`, which is handled by the following snippet:

```
import seshat.web.ui as web_ui
...
if args.command == "web":
    web_ui.main (args.config_file, True, args.initialize,
                 args.extract_transactions_from_log,
                 args.apply_transactions)
```

So, the entry-point for the `web` subcommand is found in `src/seshat/web/ui.py` at the `main` procedure.

This procedure essentially sets up an instance of `WebServer` (found in `src/seshat/web/wsgi.py`) and uses `werkzeug`'s `run_simple` to start the web server.

5.3.3 Translating URI paths to internal procedures

An instance of the `WebServer` is passed along in werkzeug's `run_simple` procedure. Werkzeug calls the instance directly, which is handled by the `__call__` procedure of the `WebServer` class. The `__call__` procedure invokes its `wsgi` instance, which is configured as following:

```
self.wsgi = SharedDataMiddleware(self.__respond, self.static_roots)
```

The `__respond` procedure calls `__dispatch_request`.

In `__dispatch_request`, the requested URI is translated into the procedure name using the `url_map`. So, except for static resources in the `src/seshat/web/resources` folder and pre-configured static pages, URIs are handled by a procedure in the `WebServer` instance.

A mapping between a URI and the procedure that is executed to handle the request to that URI can be found in the `url_map` defined in the `WebServer` class in `wsgi.py`.

5.3.4 Diving into the code that displays the homepage

As an example, in the `url_map`, we can find the following line:

```
R("/", self.ui_home),
```

In this case, `self` is a reference to an instance of the `WebServer` class, so we look for a procedure called `ui_home` inside the `WebServer` class. Some code editors have a feature to "go to definition" which helps navigating.

The `ui_home` gathers the summary numbers from the SPARQL endpoint with the following line:

```
summary_data = self.db.repository_statistics()
```

And a list of the latest datasets with the following line:

```
records = self.db.latest_datasets_portal(30)
```

It then passes that information to the `__render_template` procedure which renders the `portal.html` in the `src/seshat/web/resources/html_templates` folder. The Jinja[4] package is used to interpret the template.

```
return self.__render_template (request, "portal.html",
                               summary_data = summary_data,
                               latest = records, ...)
```

5.3.5 Database communication

In the `ui_home` procedure, we found a call to the `self.db.repository_statistics` procedure. To find out by hand where that procedure can be found, we can look for the place where `self.db` is assigned a value:

```
self.db = database.SparqlInterface()
```

And from there look up where `database` comes from:

```
from seshat.web import database
```

From which we can conclude that it can be found in `src/seshat/web/database.py`.

In the `repository_statistics` procedure, we find a call to `self.__query_from_template` followed by a call to `__run_query` which takes the output of the former procedure as its input.

As the name implies, `__run_query` sends the query to the SPARQL endpoint and retrieves the results by putting them in a list of Python dictionaries.

The `self.__query_from_template` procedure takes one parameter, which is the name of the template file (minus the extension) that contains a SPARQL query. These templates can be found in the `src/seshat/web/resources/sparql_templates` folder.

[4] https://jinja.palletsprojects.com/en/3.1.x/

6 Application Programming Interface

The application programming interface (API) provided by `seshat` allows for automating tasks otherwise done through the user interface. In addition to automation, the API can also be used to gather additional information, like statistics on Git repositories.

Throughout this chapter we provide examples for using the API using `curl` and `jq`. Another way of seeing the API in action is to use the developer tools in a web browser while performing the desired action using the web user interface.

6.1 Published datasets

The `v2` API was designed by Figshare[5]. `seshat` implements a backward-compatible version of it, with the following differences:

1. The `id` property is superseded by the `uuid` property.
2. Error handling is done through precise HTTP error codes, rather than always returning `400` on a usage error.

Unless specified otherwise, the HTTP `Content-Type` to interact with the API is `application/json`. In the case an API call returns information, don't forget to set the HTTP `Accept` header appropriately.

6.1.1 `/v2/articles` (`GET`)

This API endpoint can be used to retrieve a list of published datasets. Passing (one of) the following parameters will filter or sort the list of datasets:

Parameter	Required	Description
`order`	Optional	Field to use for sorting.
`order_direction`	Optional	Can be either `asc` or `desc`.
`institution`	Optional	The institution identifier to filter on.
`published_since`	Optional	When set, datasets published before this timestamp are dropped from the results.
`modified_since`	Optional	When set, only datasets modified after this timestamp are shown from the results.
`group`	Optional	The group identifier to filter on.
`resource_doi`	Optional	The URL of the DOI of an associated peer-reviewed journal publication. When set, only returns datasets associated with this DOI.
`item_type`	Optional	Either `3` for datasets or `9` for software.
`doi`	Optional	The DOI of the dataset to search for.
`handle`	Optional	Unused.
`page`	Optional	The page number used in combination with `page_size`.
`page_size`	Optional	The number of datasets per page. Used in combination with `page`.
`limit`	Optional	The maximum number of datasets to output. Used together with `offset`.
`offset`	Optional	The number of datasets to skip in the output. Used together with `limit`.

Example usage:

```
curl "https://demo.seshat.software/v2/articles?limit=100&published_since=2024-07-25" | jq
```

Output of the example:

[5] https://figshare.com

```
[ /* Example output has been shortened. */
  {
    "id": null,
    "uuid": "4f8a9423-83fc-4263-9bb7-2aa83d73865d",
    "title": "Measurement data of a Low Speed Field Test of Tractor Se...",
    "doi": "10.4121/4f8a9423-83fc-4263-9bb7-2aa83d73865d.v1",
    "handle": null,
    "url": "https://demo.seshat.software/v2/articles/4f8a...865d",
    "published_date": "2024-07-26T10:39:57",
    "thumb": null,
    "defined_type": 3,
    "defined_type_name": "dataset",
    "group_id": 28589,
    "url_private_api": "https://demo.seshat.software/v2/account/articles/4f8a...865d",
    "url_public_api": "https://demo.seshat.software/v2/articles/4f8a...865d",
    "url_private_html": "https://demo.seshat.software/my/datasets/4f8a...865d/edit",
    "url_public_html": "https://demo.seshat.software/datasets/4f8a...865d/1",
    ...
  }
]
```

6.1.2 `/v2/articles/search` (`POST`)

In addition to the parameters of Section 6.1.1, the following parameters can be used.

Parameter	Required	Description
`search_for`	Optional	The terms to search for.

Example usage:

```
curl --request POST \
     --header "Content-Type: application/json"\
     --data '{ "search_for": "seshat" }'\
     https://demo.seshat.software/v2/articles/search | jq
```

Output of the example:

```
[ /* Example output has been shortened. */
  {
    "id": null,
    "uuid": "342efadc-66f8-4e9b-9d27-da7b28b849d2",
    "title": "Source code of the 4TU.ResearchData repository",
    "doi": "10.4121/342efadc-66f8-4e9b-9d27-da7b28b849d2.v1",
    "handle": null,
    "url": "https://demo.seshat.software/v2/articles/342e...49d2",
    "published_date": "2023-03-20T11:29:10",
    "thumb": null,
    "defined_type": 9,
    "defined_type_name": "software",
    "group_id": 28586,
    "url_private_api": "https://demo.seshat.software/v2/account/articles/342e...49d2",
    "url_public_api": "https://demo.seshat.software/v2/articles/342e...49d2",
    "url_private_html": "https://demo.seshat.software/my/datasets/342e...49d2/edit",
    "url_public_html": "https://demo.seshat.software/datasets/342e...49d2/1",
    ...
  }
]
```

6.1.3 `/v2/articles/<dataset-id>` (`GET`)

This API endpoint can be used to retrieve detailed metadata for the dataset identified by `dataset-id`.

Example usage:

```
curl https://demo.seshat.software/v2/articles/342efadc-66f8-4e9b-9d27-da7b28b849d2 | jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "files": ...,
  "custom_fields": ...,
  "authors": ...,
  "description": "<p>This dataset contains the source code of the 4TU...",
  "license": ...,
  "tags": ...,
  "categories": ...,
  "references": ...,
  "id": null,
  "uuid": "342efadc-66f8-4e9b-9d27-da7b28b849d2",
  "title": "Source code of the 4TU.ResearchData repository",
  "doi": "10.4121/342efadc-66f8-4e9b-9d27-da7b28b849d2.v1",
  "url": "https://demo.seshat.software/v2/articles/342e...49d2",
  "published_date": "2023-03-20T11:29:10",
  "timeline": ...,
  ...
}
```

6.1.4 `/v2/articles/<dataset-id>/versions` (GET)

This API endpoint can be used to retrieve a list of versions for the dataset identified by `dataset-id`.

Example usage:

```
curl https://demo.seshat.software/v2/articles/342efadc-66f8-4e9b-9d27-da7b28b849d2/
versions | jq
```

Output of the example:

```
[
  {
    "version": 1,
    "url": "https://demo.seshat.software/v2/articles/342e...49d2/versions/1"
  }
]
```

6.1.5 `/v2/articles/<dataset-id>/versions/<version>` (GET)

This API endpoint can be used to retrieve detailed metadata of the version `version` for the dataset identified by `dataset-id`.

Example usage:

```
curl https://demo.seshat.software/v2/articles/342e...49d2/versions/1 | jq
```

The output of the example is identical to the example output of Section 6.1.3.

6.1.6 `/v2/articles/<dataset-id>/versions/<version>/embargo` (GET)

This API endpoint can be used to retrieve embargo information of the version `version` for the dataset identified by `dataset-id`.

Example usage:

```
curl https://demo.seshat.software/v2/articles/c127...8fd7/versions/2/embargo | jq
```

Output of the example:

```
{
  "is_embargoed": true,
  "embargo_date": "2039-06-30",
  "embargo_type": "article",
  "embargo_title": "Under embargo",
  "embargo_reason": "<p>Need consent to publish the data</p>",
```

```
  "embargo_options": []
}
```

6.1.7 /v2/articles/<dataset-id>/files (GET)

This API endpoint can be used to retrieve the list of files associated with the dataset identified by `dataset-id`.

Example usage:

```
curl https://demo.seshat.software/v2/articles/342efadc-66f8-4e9b-9d27-da7b28b849d2/files
```

Output of the example:

```
[ /* Example output has been shortened. */
  {
    "id": null,
    "uuid": "d3e1c325-7fa9-4cb9-884e-0b9cd2059292",
    "name": "seshat-0.0.1.tar.gz",
    "size": 3713709,
    "is_link_only": false,
    "is_incomplete": false,
    "download_url": "https://demo.seshat.software/file/342e...49d2/d3e1...9292",
    "supplied_md5": null,
    "computed_md5": "910e9b0f79a0af548f59b3d8a56c3bf4"
  }
]
```

6.1.8 /v2/articles/<dataset-id>/files/<file-id> (GET)

This API endpoint can be used to retrieve all metadata of the file identified by `file-id` associated with the dataset identified by `dataset-id`.

Example usage:

```
curl https://demo.seshat.software/v2/articles/342e...49d2/files/d3e1...9292 | jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "id": null,
  "uuid": "d3e1c325-7fa9-4cb9-884e-0b9cd2059292",
  "name": "seshat-0.0.1.tar.gz",
  "size": 3713709,
  "is_link_only": false,
  "is_incomplete": false,
  "download_url": "https://demo.seshat.software/file/342e...49d2/d3e1...9292",
  "supplied_md5": null,
  "computed_md5": "910e9b0f79a0af548f59b3d8a56c3bf4"
}
```

6.2 Published collections

6.2.1 /v2/collections (GET)

This API endpoint can be used to retrieve a list of collections published in the data repository.

The following parameters can be used:

Parameter	Required	Description
`order`	Optional	Field to use for sorting.
`order_direction`	Optional	Can be either `asc` or `desc`.
`institution`	Optional	The institution identifier to filter on.

Parameter	Required	Description
`published_since`	Optional	When set, collections published before this timestamp are dropped from the results.
`modified_since`	Optional	When set, only collections modified after this timestamp are shown from the results.
`group`	Optional	The group identifier to filter on.
`resource_doi`	Optional	The URL of the DOI of an associated peer-reviewed journal publication. When set, only returns collections associated with this DOI.
`doi`	Optional	The DOI of the collection to search for.
`handle`	Optional	Unused.
`page`	Optional	The page number used in combination with `page_size`.
`page_size`	Optional	The number of collections per page. Used in combination with `page`.
`limit`	Optional	The maximum number of collections to output. Used together with `offset`.
`offset`	Optional	The number of collections to skip in the output. Used together with `limit`.

Example usage:

```
curl "https://demo.seshat.software/v2/collections?limit=100&published_since=2024-07-25" |
jq
```

Output of the example:

```
[ /* Example output has been shortened. */
  {
    "id": null,
    "uuid": "0fe9ab80-6e6a-4087-a509-ce09dddfa3d9",
    "title": "PhD research 'Untangling the complexity of local water ...'",
    "doi": "10.4121/0fe9ab80-6e6a-4087-a509-ce09dddfa3d9.v1",
    "handle": "",
    "url": "https://demo.seshat.software/v2/collections/0fe9...fa3d9",
    "timeline": {
      "posted": "2024-08-13T14:09:52",
      "firstOnline": "2024-08-13T14:09:51",
      ...
    },
    "published_date": "2024-08-13T14:09:52"
  },
  ...
]
```

6.2.2 `/v2/collections/search` (`POST`)

This API endpoint can be used to search for collections published in the data repository.

In addition to the parameters of Section 6.2.1, the following parameters can be used.

Parameter	Required	Description
`search_for`	Optional	The terms to search for.

Example usage:

```
curl --request POST \
     --header "Content-Type: application/json"\
     --data '{ "search_for": "wingtips" }'\
     https://demo.seshat.software/v2/collections/search | jq
```

Output of the example:

```
[  /* Example output has been shortened. */
  {
    "id": 6070238,
    "uuid": "3dfc4ef2-7f79-4d33-81a7-9c6ae09a2782",
    "title": "Flared Folding Wingtips - TU Delft",
    "doi": "10.4121/c.6070238.v1",
    "handle": "",
    "url": "https://demo.seshat.software/v2/collections/3dfc...2782",
    "timeline": {
      "posted": "2023-04-05T15:05:04",
      "firstOnline": "2023-04-05T15:05:03",
      ...
    },
    "published_date": "2023-04-05T15:05:04"
  },
  ...
]
```

6.2.3 `/v2/collections/<collection-id>` (GET)

This API endpoint can be used to retrieve detailed metadata for the collection identified by `collection-id`.

Example usage:

```
curl https://demo.seshat.software/v2/collections/3dfc4ef2-7f79-4d33-81a7-9c6ae09a2782 | jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "version": 3,
  ...
  "description": "<p>This collection contains the results of the work ...",
  "categories": [ ... ],
  "references": [],
  "tags": [ ... ],
  "created_date": "2024-08-08T15:48:55",
  "modified_date": "2024-08-12T11:24:39",
  "id": 6070238,
  "uuid": "3dfc4ef2-7f79-4d33-81a7-9c6ae09a2782",
  "title": "Flared Folding Wingtips - TU Delft",
  "doi": "10.4121/c.6070238.v3",
  "published_date": "2024-08-12T11:24:40",
  "timeline": ...
  ...
}
```

6.2.4 `/v2/collections/<collection-id>/versions` (GET)

This API endpoint can be used to retrieve a list of versions for the collection identified by `collection-id`.

Example usage:

```
curl https://demo.seshat.software/v2/collections/3dfc4ef2-7f79-4d33-81a7-9c6ae09a2782/
versions | jq
```

Output of the example:

```
[ /* Example output has been shortened. */
  {
    "version": 3,
    "url": "https://demo.seshat.software/v2/collections/3dfc...2782/versions/3"
  },
  {
```

```
    "version": 2,
    "url": "https://demo.seshat.software/v2/collections/3dfc...2782/versions/2"
  },
  {
    "version": 1,
    "url": "https://demo.seshat.software/v2/collections/3dfc...2782/versions/1"
  }
]
```


6.2.5 /v2/collections/<collection-id>/versions/<version> (GET)

This API endpoint can be used to retrieve detailed metadata of the version `version` for the collection identified by `collection-id`.

Example usage:

```
curl https://demo.seshat.software/v2/collections/3dfc...2782/versions/2 | jq
```

Output of the example:


```
{ /* Example output has been shortened. */
  "version": 2,
  ...
  "description": "<p>This collection contains the results of the work ...",
  "categories": [ ... ],
  "references": [],
  "tags": [ ... ],
  "references": [],
  "tags": [ ... ],
  "authors": [ ... ],
  "created_date": "2023-04-05T15:07:35",
  "modified_date": "2023-05-26T15:19:11",
  "id": 6070238,
  "uuid": "3dfc4ef2-7f79-4d33-81a7-9c6ae09a2782",
  "title": "Flared Folding Wingtips - TU Delft",
  "doi": "10.4121/c.6070238.v2",
  ...
}
```


6.2.6 /v2/collections/<collection-id>/articles (GET)

This API endpoint can be used to retrieve the list of datasets in the collection identified by `collection-id`.

Example usage:

```
curl https://demo.seshat.software/v2/collections/3dfc...2782/articles | jq
```

Output of the example:


```
[ /* Example output has been shortened. */
  {
    "id": 20222334,
    "uuid": "c5fde4a2-798a-456e-b793-cf64e486c0e8",
    "title": "E001 - Stiffness and Hinge Release Study (October 2021) ...",
    "doi": "10.4121/20222334.v2",
    "published_date": "2023-05-31T08:57:54",
    "defined_type": 3,
    "defined_type_name": "dataset",
    "group_id": 28586,
    "timeline": {
      "posted": "2023-05-31T08:57:54",
      "firstOnline": "2023-05-26T15:08:09",
      "revision": null
    },
    "resource_title": "Effect of Wing Stiffness and Folding Wingtip ...",
    "resource_doi": "https://doi.org/10.2514/1.C037108"
```

```
  },
  {
    "id": null,
    "uuid": "984090ea-26fd-4809-8dac-f41367bf8916",
    "title": "M001 - GVT Data and Nastran models (August 2024) ...",
    "doi": "10.4121/984090ea-26fd-4809-8dac-f41367bf8916.v1",
    "published_date": "2024-08-12T11:21:47",
    "defined_type": 3,
    "defined_type_name": "dataset",
    "group_id": 28586,
    "timeline": {
      "posted": "2024-08-12T11:21:47",
      "firstOnline": "2024-08-12T11:21:46",
      "revision": null
    },
    "resource_title": "Effect of Wing Stiffness and Folding Wingtip ...",
    "resource_doi": "https://doi.org/10.2514/1.C037108"
  }
]
```


6.3 Categories

6.3.1 `/v2/categories` (GET)

Each dataset and collection is categorized using a controlled vocabulary of categories. This API endpoint provides those categories.

Example usage:

```
curl https://demo.seshat.software/v2/categories | jq
```

Output of the example:


```
[ /* Example output has been shortened. */
  {
    "id": 13622,
    "uuid": "01fddd41-68d2-4e28-9d9c-18347847e7d1",
    "title": "Mining and Extraction of Energy Resources",
    "parent_id": 13620,
    "parent_uuid": "6e5bdc69-96db-41e4-ac0b-18812b46c49c",
    "path": "",
    "source_id": null,
    "taxonomy_id": null
  },
  {
    "id": 13443,
    "uuid": "026f555c-2826-4a83-97ff-0f230fb54ddb",
    "title": "Livestock Raising",
    "parent_id": 13440,
    "parent_uuid": "45a8c849-ab59-4302-af79-09b8c0677df8",
    "path": "",
    "source_id": null,
    "taxonomy_id": null
  },
  ...
]
```


6.4 Licences

6.4.1 `/v2/licenses` (GET)

Publishing a dataset involves communicating under which conditions it can be re-used. The licenses under which you can publish a dataset can be found with this API endpoint.

Example usage:

```
curl https://demo.seshat.software/v2/licenses | jq
```

Output of the example:

```
[ /* Example output has been shortened. */
  {
    "value": 1,
    "name": "CC BY 4.0",
    "url": "https://creativecommons.org/licenses/by/4.0/",
    "type": "data"
  },
  {
    "value": 10,
    "name": "CC BY-NC 4.0",
    "url": "https://creativecommons.org/licenses/by-nc/4.0/",
    "type": "data"
  },
  ...
]
```

6.5 Account properties

The interaction with the `v2` private interface API requires an API token. Such a token can be obtained from the dashboard page after logging in. This token can then be passed along in the `Authorization` HTTP header as:

```
Authorization: token YOUR_TOKEN_HERE
```

6.5.1 `/v2/account` (`GET`)

This API endpoint can be used to retrieve information about the account identified with the API token.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account | jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "id": null,
  "uuid": "df7c0e54-b988-42b1-a815-308513d2f269",
  "is_active": true,
  "is_public": false,
  ...
}
```

6.6 Authors

6.6.1 `/v2/account/authors/search` (`POST`)

This API endpoint can be used to search for authors known to the data repository.

The following parameter can be used:

Parameter	Data type	Required	Description
`search`	`string`	Yes	The string to search for.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --data '{ "search": "John Doe" }' \
     https://demo.seshat.software/v2/account/authors/search | jq
```

Output of the example:

```
[ /* This example output has been shortened. */
  {
    "full_name": "John Doe Jr",
    "uuid": "08f4d496-67b5-4b7c-b2d2-923458d1f450",
    "orcid_id": "",
    ...
  },
  {
    "full_name": "John Doe",
    "uuid": "6815031c-21dc-4873-93c9-f6539da482ce",
    "orcid_id": "",
    ...
  }
]
```

6.6.2 `/v2/account/authors/<author-id>` (`GET`)

This API endpoint returns a detailed author record for the author identified by `author-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/authors/5c75...94aa | jq
```

Output of the example:

```
{ /* This example output has been shortened. */
  "first_name": "Roel",
  "full_name": "Roel Janssen",
  "uuid": "5c752155-60ff-41d7-9b88-b7112afc94aa",
  "last_name": "Janssen",
  "orcid_id": "0000-0003-4324-5350",
  ...
}
```

6.7 Institutions

6.7.1 `/v2/account/institution` (`GET`)

This API endpoint returns the accounts within your institution.

Example usage:

```
curl --header "Accept: application/json" \
     --header "Authorization: token ${API_TOKEN}" \
     https://demo.seshat.software/v2/account/institution/accounts | jq
```

Output of the example:

```
[
  { /* This example output has been shortened. */
    "uuid": "485a04c8-7fb0-4361-856f-470930c5fec0",
    "first_name": "Roel",
    "last_name": "Janssen",
    "full_name": "Roel Janssen",
    "is_active": true,
    "is_public": false,
    ...
  },
  ...
]
```

6.7.2 `/v2/account/institution/users/<account-id>` (`GET`)

This API endpoint returns account information for the specified `account-id`.

Example usage:

```
curl --header "Accept: application/json" \
     --header "Authorization: token ${API_TOKEN}" \
     https://demo.seshat.software/v2/account/institution/users/485a...fec0 | jq
```

Output of the example:

```
{ /* This example output has been shortened. */
  "uuid": "485a04c8-7fb0-4361-856f-470930c5fec0",
  "first_name": "Roel",
  "last_name": "Janssen",
  "full_name": "Roel Janssen",
  "is_active": true,
  "is_public": false,
  ...
}
```

6.8 Git repositories

A published dataset may include a Git repository in its publication. The Git repository has a unique UUID that isn't re-used between versions of the same publication.

6.8.1 `/v3/datasets/<git-uuid>.git` (GET)

This endpoint can be used to pass as a URL to `git clone`.

Example usage:

```
git clone https://demo.seshat.software/v3/datasets/de82...20b7.git
```

Output of the example:

```
Cloning into 'seshat'...
remote: Enumerating objects: 24850, done.
remote: Counting objects: 100% (4171/4171), done.
remote: Compressing objects: 100% (599/599), done.
remote: Total 24850 (delta 3963), reused 3647 (delta 3571), pack-reused 20679 (from 2)
Receiving objects: 100% (24850/24850), 11.41 MiB | 569.00 KiB/s, done.
Resolving deltas: 100% (17094/17094), done.
```

6.8.2 `/v3/datasets/<git-uuid>.git/zip` (GET)

This endpoint offers a ZIP file to download the files in the Git repository without using Git. The ZIP file is generated on the spot, and therefore, the timestamps of files within the ZIP file are set to *January 1, 1980* to ensure that downloading a ZIP file multiple times results in an identical ZIP file.

6.8.3 `/v3/datasets/<git-uuid>.git/languages` (GET)

This API endpoint can be used to gather which programming languages are used in which ratio to each other for the Git repository identified by the `git-uuid`.

The number returned per programming language is the sum of the number of bytes of files identified to belong to that programming language.

Example usage:

```
curl --header "Accept: application/json" \
     https://demo.seshat.software/v3/datasets/de82...20b7.git/languages
```

Output of the example:

```
{ /* Example output has been shortened. */
  "Python": 963065,
  "JavaScript": 188239,
  "HTML": 186766,
  ...
}
```

6.8.4 /v3/datasets/<git-uuid>.git/contributors (GET)

This API endpoint provides contributions in the form of additions, deletions and commits per week per author.

Example usage:

```
curl --header "Accept: application/json" \
     https://demo.seshat.software/v3/datasets/de82...20b7.git/contributors
```

Output of the example:

```
[ /* Example output has been annotated and shortened. */
  {
    "total": 2769,
    "additions": 94508,
    "deletions": 62028,
    "weeks": [
      {
        "w": 1624831200,   /* Timestamp for the week */
        "a": 100,          /* Additions */
        "d": 0,            /* Deletions */
        "c": 2             /* Commits */
      },
      ...
    ],
    "author": {
      "name": "Roel Janssen",
      "email": "..."       /* Omitted from the example. */
    }
  },
  ...
]
```

6.9 Creating and editing datasets

6.9.1 /v2/account/articles (GET)

This API endpoint lists the draft datasets of the account to which the authorization token belongs.

The following parameters can be used:

Parameter	Required	Description
`page`	Optional	The page number used in combination with `page_size`.
`page_size`	Optional	The number of datasets per page. Used in combination with `page`.
`limit`	Optional	The maximum number of datasets to output. Used together with `offset`.
`offset`	Optional	The number of datasets to skip in the output. Used together with `limit`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles | jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "id": null,
  "uuid": "6ddd7a31-8ad8-4c20-95a3-e68fe716fa42",
  "title": "Example draft dataset",
  "doi": null,
  "handle": null,
  "url": "https://demo.seshat.software/v2/articles/6ddd7a31-8ad8-4c20-95a3-e68fe716fa42",
  "published_date": null,
```

```
  ...
}
```

6.9.2 /v2/account/articles (POST)

This API endpoint can be used to create a new dataset.

The following parameters can be used:

Parameter	Data type	Description
title	string	The title of the dataset.
description	string	A description of the dataset.
tags	list of strings	Keywords to enhance the findability of the dataset. Instead of using the key tags, you may also use the key keywords.
keywords	list of strings	See tags.
references	list of strings	URLs to resources referring to this dataset, or resources that this dataset refers to.
categories	list of strings	Categories are a controlled vocabulary and can be used to make the collection findable in the categorical overviews. The string values expected here can be found under the uuid property with a call to \/v2/categories. For more details, see Section 6.3.1.
authors	list of author records	
defined_type	string	One of: figure, online resource, preprint, book, conference contribution, media, dataset, poster, journal contribution, presentation, thesis or software.
funding	string	One-liner to cite funding.
funding_list	list of funding records	
license	integer	Licences communicate under which conditions the dataset can be re-used. The integer value to submit here can be found as the value property in a call to \/v2/licences. For more details, see Section 6.4.1.
language	string	An ISO 639-1 language code.
doi	string	Do not use this field as a DOI will be automatically assigned upon publication.
handle	string	Do not use this field as it is deprecated.
resource_doi	string	The URL of the DOI of an associated peer-reviewed journal publication.
resource_title	string	The title of the associated peer-reviewed journal publication.
publisher	string	The name of the data repository publishing the dataset.
custom_fields	list of key-value pairs	An Object where each key is a field name and each value is the corresponding value. Allowed values are: contributors, data_link, derived_from, format, geolocation, language, latitude, longitude, organizations, publisher, same_as, time_coverage.

Parameter	Data type	Description
custom_fields_list	list of Objects	Each Object should have two keys: name and value. For allowed keys, see custom_fields.
timeline		Do not use this field because it will be automatically populated during the publication process.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "title": "Example dataset" }' \
     https://demo.seshat.software/v2/account/articles | jq
```

Output of the example:

```
{ /* The UUID in this example has been shortened. */
  "location": "https://demo.seshat.software/v2/account/articles/d7b3...995b1",
  "warnings": []
}
```

6.9.3 /v2/account/articles/<dataset-id> (GET)

This API endpoint lists details of the dataset identified by dataset-id.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b1
| jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "files": [],
  "authors": [],
  "id": null,
  "uuid": "637e9a3b-3e6d-4810-bc8d-f15ab1d6a4d7",
  "title": "Example dataset",
  ...
}
```

6.9.4 /v2/account/articles/<dataset-id> (PUT)

This API endpoint can be used to update the metadata of the dataset identified by dataset-id.

The following parameters can be used:

Parameter	Required	Description
title	string	The title of the dataset.
description	string	A description of the dataset.
resource_doi	string	The URL of the DOI of an associated peer-reviewed journal publication.
resource_title	string	The title of the associated peer-reviewed journal publication.
license	integer	Licences communicate under which conditions the dataset can be re-used. The integer value to submit here can be found as the value property in a call to \/v2/licences. For more details, see Section 6.4.1.
group_id	integer	

Parameter	Required	Description
`time_coverage`	`string`	Free-text field to describe the time coverage of the dataset.
`publisher`	`string`	The name of the data repository publishing the dataset.
`language`	`string`	An ISO 639-1 language code.
`contributors`	`string`	Free-text field to indicate contributors to the dataset other than direct authors.
`license_remarks`	`string`	Free-text field to clarify licensing details.
`geolocation`	`string`	Free-text field to specify a location.
`longitude`	`string`	The longitude coordinate of the location.
`latitude`	`string`	The latitude coordinate of the location.
`format`	`string`	Free-text field to indicate the data format(s) used in the dataset.
`data_link`	`string`	URL to where the data can be found. This is only applicable when data is not directly uploaded.
`derived_from`	`string`	DOI or URL of a dataset from which this dataset is derived from.
`same_as`	`string`	DOI or URL of the dataset that is the same as this one.
`organizations`	`string`	Free-text field to specify organizations that contributed or are associated with the dataset.
`is_embargoed`	`boolean`	Set to `true` when the dataset is under embargo.
`embargo_options`	`Object`	An `Object` with an `id` property that can have either the integer value `1000` to indicate the dataset has no end-date for the embargo or the integer value `1001` to indicate that the dataset is permanently closed-access.
`embargo_until_date`	`string`	A date indicator for when the dataset will be available publically.
`embargo_type`	`string`	Either `file` for files-only embargo or `article` to also hide the metadata, except for the title and authors of the dataset.
`embargo_title`	`string`	Title of the embargo.
`embargo_reason`	`string`	Reason for the embargo.
`is_metadata_record`	`boolean`	Set to `true` when no data is associated with this dataset.
`metadata_reason`	`string`	Reason why the dataset is metadata-only.
`eula`	`string`	An End-User-License-Agreement.
`defined_type`	`string`	Either `software` to indicate the dataset is software or `dataset` to indicate the dataset is data (not software).
`git_repository_name`	`string`	Title of the Git repository (for software datasets only). This is a seshat-extension to the original API specification.

Parameter	Required	Description
`git_code_hosting_url`	`string`	Link to the code hosting platform (e.g. Gitlab, or any other). This is a seshat-extension to the original API specification.
`agreed_to_deposit_agreement`	`boolean`	Set to `true` when you agree to the repository's deposit agreement. This is a seshat-extension to the original API specification.
`agreed_to_publish`	`boolean`	Set to `true` to indicate the dataset may be published. This is a seshat-extension to the original API specification.
`categories`	list of `strings`	Categories are a controlled vocabulary and can be used to make the collection findable in the categorical overviews. The `string` values expected here can be found under the `uuid` property with a call to `\/v2/categories`. For more details, see Section 6.3.1.

Example usage:

```
curl --verbose --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --data '{ "title": "Updated title" }'
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b1
| jq
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.9.5 `/v2/account/articles/<dataset-id>` (`DELETE`)

This API endpoint can be used to delete a draft dataset.

Example usage:

```
curl --verbose --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b1
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.9.6 `/v2/account/articles/<dataset-id>/authors` (`GET`)

This API endpoint lists the authors of the dataset identified by `dataset-id`. The following URL parameters can be used:

Parameter	Required	Description
`order`	Optional	Field to use for sorting.
`order_direction`	Optional	Can be either `asc` or `desc`.
`limit`	Optional	The maximum number of datasets to output. Used together with `offset`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b1
| jq
```

Output of the example:

```
[
  {
    "id": null,
    "uuid": "08f4d496-67b5-4b7c-b2d2-923458d1f450",
    "full_name": "John Doe Jr",
    "is_active": false,
    "url_name": null,
    "orcid_id": ""
  },
  {
    "id": null,
    "uuid": "6815031c-21dc-4873-93c9-f6539da482ce",
    "full_name": "John Doe",
    "is_active": false,
    "url_name": null,
    "orcid_id": ""
  }
]
```

6.9.7 `/v2/account/articles/<dataset-id>/authors` (`POST`)

This API endpoint can be used to append authors to the dataset identified by `dataset-id`.

The following parameters can be used:

Parameter	Data type	Required	Description
`full_name`	`string`	No	The full name of the author.
`first_name`	`string`	Yes	The first name of the author.
`last_name`	`string`	Yes	The last name of the author.
`email`	`string`	No	The e-mail address of the author.
`orcid_id`	`string`	No	The ORCID identifier for the author.
`job_title`	`string`	No	The job title of the author.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "authors": [{ "name": "John Doe" }]}' \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b
1/authors
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "authors": [{ "name": "John Doe Jr" }]}' \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b
1/authors
```

The following is an example of the output of the HTTP `POST` calls:

```
HTTP/1.1 205 RESET CONTENT
```

An example of the output of the HTTP `GET` call can be found in Section 6.9.6.

6.9.8 `/v2/account/articles/<dataset-id>/authors` (`PUT`)

In contrast to Section 6.9.7, this API endpoint can be used to **overwrite** the list of authors of the dataset identified by `dataset-id`.

Example usage:

```
curl --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
```

```
     --header "Content-Type: application/json" \
     --data '{ "authors": [{ "name": "John Doe" }]}' \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b
1/authors

curl --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "authors": [{ "name": "John Doe Jr" }]}' \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b
1/authors

curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b1
| jq
```

Output of the example:

```
[
  {
    "id": null,
    "uuid": "61751fe3-53a1-477f-a46f-e534cbd0b618",
    "full_name": "John Doe Jr",
    "is_active": false,
    "url_name": null,
    "orcid_id": ""
  },
]
```

6.9.9 `/v2/account/articles/<dataset-id>/authors/<author-id>` (DELETE)

This API endpoint can be used to delete an author's association with a dataset.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/authors/6175...0b618
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.9.10 `/v2/account/articles/<dataset-id>/funding` (GET)

This API endpoint lists the funding of the dataset identified by `dataset-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...95b1/funding | jq
```

Output of the example:

```
[
  {
    "id": null,
    "uuid": "6f605fe1-e87a-43f5-8b67-70ebe3f9b868",
    "title": "Example cases fund",
    "grant_code": "EXA-001",
    "funder_name": "Example",
    "is_user_defined": null,
    "url": "https://example.exa"
  }
]
```

6.9.11 /v2/account/articles/<dataset-id>/funding (POST)

This API endpoint can be used to append funders to the dataset identified by `dataset-id`.

The following parameters can be used:

Parameter	Data type	Required	Description
`uuid`	`string`	No	The uuid of an existing funding record. When this parameter is set, other parameters will be ignored.
`title`	`string`	Yes	The title of the funding project.
`grant_code`	`string`	No	An optional grant code of the funding.
`funder_name`	`string`	No	The name of the funder.
`url`	`string`	No	A URL to the funding project or funding organization.

Example usage:

```
curl --verbose --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "funders": [{ "title": "Example cases fund", \
                             "grant_code": "EXA-001", \
                             "funder_name": "Example", \
                             "url": "https://example.exa" }]}' \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b
1/funding
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.9.12 /v2/account/articles/<dataset-id>/funding (PUT)

In contrast to Section 6.9.11, this API endpoint can be used to **overwrite** the list of funders of the dataset identified by `dataset-id`.

```
curl --verbose --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "funders": [{ "title": "Example cases fund",
                             "grant_code": "EXA-001",
                             "funder_name": "Example",
                             "url": "https://example.exa" }]}' \
     https://demo.seshat.software/v2/account/articles/d7b3daa5-45e2-47b0-9910-0f7fa6a995b
1/funding
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.9.13 /v2/account/articles/<dataset-id>/funding/<funding-id> (DELETE)

This API endpoint can be used to delete an funder's association with a dataset.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/funding/d50e...7500
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.9.14 /v2/account/articles/<dataset-id>/categories (GET)

This API endpoint lists the categories of the dataset identified by `dataset-id`. The identifiers for the categories can be found by using the API endpoint described at Section 6.3.1.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...95b1/categories | jq
```

Output of the example:

```
[
  {
    "id": 13558,
    "uuid": "8f27eb44-0a63-4496-ba6d-e3cbf4efa6c7",
    "title": "Other Earth Sciences",
    "parent_id": 13551,
    "parent_uuid": "dd4dbaaf-0610-4d8d-8b07-e1eeb32dd11c",
    "path": "",
    "source_id": null,
    "taxonomy_id": null
  },
  {
    "id": 13551,
    "uuid": "dd4dbaaf-0610-4d8d-8b07-e1eeb32dd11c",
    "title": "Earth Sciences",
    "parent_id": null,
    "parent_uuid": null,
    "path": "",
    "source_id": null,
    "taxonomy_id": null
  }
]
```

6.9.15 /v2/account/articles/<dataset-id>/categories (POST)

This API endpoint can be used to append categories to the dataset identified by `dataset-id`. The parameters sent to the server should be a JSON object with a single key named `categories`, with as value a list of either the numeric or the UUID identifiers for a category. The API endpoint described in Section 6.9.14 shows how to obtain the category identifiers.

Example usage:

```
curl --verbose --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "categories": [13551, 13558]}' \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/categories
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.9.16 /v2/account/articles/<dataset-id>/categories (PUT)

In contrast to Section 6.9.15, this API endpoint can be used to **overwrite** the list of categories of the dataset identified by `dataset-id`.

Example usage:

```
curl --verbose --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "categories": ["dd4dbaaf-0610-4d8d-8b07-e1eeb32dd11c"]}' \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/categories
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.9.17 /v2/account/articles/<dataset-id>/categories/<category-id> (DELETE)

This API endpoint can be used to delete a category's association with a dataset. The `category-id` can be either the `uuid` or the `id` property.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/categories/5c61...b668
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.9.18 /v2/account/articles/<dataset-id>/embargo (GET)

This API endpoint lists the embargo status of the dataset identified by `dataset-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/embargo | jq
```

Output of the example:

```
{
  "is_embargoed": false,
  "embargo_date": null,
  "embargo_type": "file",
  "embargo_title": "",
  "embargo_reason": "",
  "embargo_options": []
}
```

6.9.19 /v2/account/articles/<dataset-id>/embargo (DELETE)

This API endpoint can be used to remove an embargo on the dataset identified by `dataset-id`.

Example usage:

```
curl --verbose --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3d...995b1/embargo
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.9.20 /v2/account/articles/<dataset-id>/files (GET)

This API endpoint lists files associated with the dataset identified by `dataset-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/files | jq
```

Output of the example:

```
[ /* Example output has been shortened. */
  {
    "id": null,
    "uuid": "d112d0cd-bc15-4f8e-9013-930750fc017a",
    "name": "README.md",
    "size": 3696,
```

```
    "is_link_only": false,
    "is_incomplete": false,
    "download_url": "https://demo.seshat.software/file/d7b3...995b1/d112...c017a",
    "supplied_md5": null,
    "computed_md5": "c5b36584a0d62d28e9bf9e6892d9ebac"
  }
]
```

6.9.21 /v2/account/articles/<dataset-id>/files (DELETE)

> **Note:** This API endpoint is a seshat extension to the original specification.

This API endpoint can be used to delete all files associated with the dataset identified by `dataset-id`.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "remove_all": true }' \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/files
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.9.22 /v2/account/articles/<dataset-id>/files/<file-id> (GET)

This API endpoint lists files associated with the dataset identified by `dataset-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/files | jq
```

Output of the example:

```
[ /* Example output has been shortened. */
  {
    "id": null,
    "uuid": "d112d0cd-bc15-4f8e-9013-930750fc017a",
    "name": "README.md",
    "size": 3696,
    "is_link_only": false,
    "is_incomplete": false,
    "download_url": "https://demo.seshat.software/file/d7b3...995b1/d112...c017a",
    "supplied_md5": null,
    "computed_md5": "c5b36584a0d62d28e9bf9e6892d9ebac"
  }
]
```

6.9.23 /v2/account/articles/<dataset-id>/private_links (GET)

This API endpoint lists the private links associated with the dataset identified by `dataset-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/private_links | jq
```

Output of the example:

```
[
  {
    "id": "8G2f...IJP0",
    "is_active": true,
    "expires_date": "2032-01-01T00:00:00"
```

```
  },
  {
    "id": "Hb0a...diitg",
    "is_active": true,
    "expires_date": "2026-01-01T00:00:00"
  }
]
```

6.9.24 /v2/account/articles/<dataset-id>/private_links (POST)

This API endpoint can be used to append a private link to the dataset identified by `dataset-id`.

The following parameter can be used:

Parameter	Data type	Required	Description
`expires_date`	`string`	No	The format of the date string should be `YYYY-MM-DD`.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --data '{ "expires_date": "2032-01-01", "read_only": false }' \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/private_links | jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "location": "https://demo.seshat.software/private_datasets/8G2fk..."
}
```

6.9.25 /v2/account/articles/<dataset-id>/private_links/<link-id> (GET)

This API endpoint can be used to view the details of a private link for the dataset identified by `dataset-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/private_links/8G2fk...
| jq
```

Output of the example:

```
[
  {
    "id": "8G2f...IJP0",
    "is_active": true,
    "expires_date": "2032-01-01T00:00:00"
  }
]
```

6.9.26 /v2/account/articles/<dataset-id>/private_links/<link-id> (PUT)

This API endpoint can be used to update the expiry date of a private link and whether the private link is active or not for the dataset identified by `dataset-id`.

The following parameter can be used:

Parameter	Data type	Required	Description
`expires_date`	`string`	No	The format of the date string should be `YYYY-MM-DD`.
`is_active`	`boolean`	No	Defaults to `false`.

Example usage:

```
curl --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --data '{ "expires_date": "2034-01-01", "is_active": true }' \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/private_links/8G2fk...
| jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "location": "https://demo.seshat.software/private_datasets/8G2fk..."
}
```

6.9.27 /v2/account/articles/<dataset-id>/private_links/<link-id> (DELETE)

This API endpoint can be used to remove a private link for the dataset identified by `dataset-id`.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/private_links/8G2fk...
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.9.28 /v2/account/articles/<dataset-id>/reserve_doi (POST)

This API endpoint can be used to obtain the DOI before the dataset is published and the DOI is activated.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/reserve_doi | jq
```

Output of the example:

```
{
  "doi": "10.5074/d7b3daa5-45e2-47b0-9910-0f7fa6a995b1"
}
```

6.9.29 /v2/account/articles/<dataset-id>/publish (POST)

This API endpoint can be used to publish the dataset identified by `dataset-id`.

> **Note:** Only users with the "review" privilege can succesfully use this API call.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/articles/d7b3...995b1/publish | jq
```

HTTP response of the example:

```
HTTP/1.1 201 CREATED
```

Output of the example:

```
{ /* Example output has been shortened. */
  "location": "https://demo.seshat.software/review/published/9ce6...3976"
}
```

6.10 Creating and editing collections

6.10.1 `/v2/account/collections` (`GET`)

This API endpoint lists the draft collections of the account to which the authorization token belongs.

The following parameters can be used:

Parameter	Required	Description
`page`	Optional	The page number used in combination with `page_size`.
`page_size`	Optional	The number of datasets per page. Used in combination with `page`.
`limit`	Optional	The maximum number of datasets to output. Used together with `offset`.
`offset`	Optional	The number of datasets to skip in the output. Used together with `limit`.
`order`	Optional	Field to use for sorting.
`order_direction`	Optional	Can be either `asc` or `desc`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections | jq
```

Output of the example:

```
[ /* This example output has been shortened. */
  {
    "id": null,
    "uuid": "fc03a4c3-cba4-4a88-a8a6-eb38924eeb6d",
    "title": "Test collection",
    "doi": null,
    "handle": "",
    "url": "https://demo.seshat.software/v2/collections/fc03...eb6d",
    "published_date": null,
    ...
  }
]
```

6.10.2 `/v2/account/collections` (`POST`)

This API endpoint can be used to create a new collection.

The following parameters can be used:

Parameter	Data type	Description
`title`	`string`	The title of the collection.
`description`	`string`	A description of the collection.
`tags`	list of `strings`	Keywords to enhance the findability of the collection. Instead of using the key `tags`, you may also use the key `keywords`.
`keywords`	list of `strings`	See `tags`.
`references`	list of `strings`	URLs to resources referring to this collection, or resources that this collection refers to.
`categories`	list of `strings`	Categories are a controlled vocabulary and can be used to make the collection findable in the categorical overviews. The `string` values expected here can be found under the `uuid` property with

Parameter	Data type	Description
		a call to `\/v2/categories`. For more details, see Section 6.3.1.
`authors`	list of author records	
`funding`	`string`	One-liner to cite funding.
`funding_list`	list of funding records	
`license`	`integer`	Licences communicate under which conditions the collection can be re-used. The `integer` value to submit here can be found as the `value` property in a call to `\/v2/licences`. For more details, see Section 6.4.1.
`doi`	`string`	Do not use this field as a DOI will be automatically assigned upon publication.
`handle`	`string`	Do not use this field as it is deprecated.
`resource_doi`	`string`	The URL of the DOI of an associated peer-reviewed journal publication.
`resource_title`	`string`	The title of the associated peer-reviewed journal publication.
`custom_fields`	`Object`	An `Object` where each key is a field name and each value is the corresponding value. Allowed values are: `contributors`, `data_link`, `derived_from`, `format`, `geolocation`, `language`, `latitude`, `longitude`, `organizations`, `publisher`, `same_as`, `time_coverage`.
`custom_fields_list`	list of `Objects`	Each `Object` should have two keys: `name` and `value`. For allowed keys, see `custom_fields`.
`timeline`		Do not use this field because it will be automatically populated during the publication process.
`articles`	list of strings or integers	The articles to include in the collection.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "title": "Example collection" }' \
     https://demo.seshat.software/v2/account/collections | jq
```

Output of the example:

```
{
  "location": "https://demo.seshat.software/v2/account/collections/08b7...cfa8",
  "warnings": []
}
```

6.10.3 `/v2/account/collections/<collection-id>` (`GET`)

This API endpoint lists details of the collection identified by `collection-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/08b7...cfa8 | jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "articles_count": 0,
  "authors": [],
  "id": null,
  "uuid": "08b702d6-98a0-4081-9445-5aeae720cfa8",
  "title": "Example collection",
  ...
}
```

6.10.4 `/v2/account/collections/<collection-id>` (`PUT`)

This API endpoint can be used to update the metadata of the collection identified by `collection-id`.

The following parameters can be used:

Parameter	Data type	Required	Description
`title`	`string`	Yes	The title of the collection.
`description`	`string`	No	A description of the collection.
`resource_doi`	`string`	No	The URL of the DOI of an associated peer-reviewed journal publication.
`resource_title`	`string`	No	The title of the associated peer-reviewed journal publication.
`group_id`	`integer`	No	
`time_coverage`	`string`	No	Free-text field to describe the time coverage of the collection.
`publisher`	`string`	No	The name of the data repository publishing the collection.
`language`	`string`	No	An ISO 639-1 language code.
`contributors`	`string`	No	Free-text field to indicate contributors to the collection other than direct authors.
`geolocation`	`string`	No	Free-text field to specify a location.
`longitude`	`string`	No	The longitude coordinate of the location.
`latitude`	`string`	No	The latitude coordinate of the location.
`organizations`	`string`	No	Free-text field to specify organizations that contributed or are associated with the collection.
`categories`	list of `strings`	No	Categories are a controlled vocabulary and can be used to make the collection findable in the categorical overviews. The `string` values expected here can be found under the `uuid` property with a call to `\/v2/categories`. For more details, see Section 6.3.1.

Example usage:

```
curl --verbose --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "title": "Updated title" }' \
     https://demo.seshat.software/v2/account/collections/08b702d6-98a0-4081-9445-5aeae720
cfa8 | jq
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.10.5 /v2/account/collections/<collection-id> (DELETE)

This API endpoint can be used to delete a draft collection.

Example usage:

```
curl --verbose --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/08b702d6-98a0-4081-9445-5aeae720
cfa8
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.10.6 /v2/account/collections/search (POST)

This API call searches for collections, including drafts created by the account performing the search.

Parameter	Required	Description
search_for	Optional	The terms to search for.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "search_for": "Example" }' \
     https://demo.seshat.software/v2/account/collections/search | jq
```

Output of the example:

```
[ /* Example output has been shortened. */
  {
    "id": null,
    "uuid": "08b702d6-98a0-4081-9445-5aeae720cfa8",
    "title": "Example collection",
    "url": https://demo.seshat.software/v2/collections/08b7...cfa8
    ...
  }
]
```

6.10.7 /v2/account/collections/<collection-id>/authors (GET)

Similar to Section 6.9.6, this API endpoint lists the authors of the collection identified by `collection-id`. The following URL parameters can be used:

Parameter	Required	Description
order	Optional	Field to use for sorting.
order_direction	Optional	Can be either `asc` or `desc`.
limit	Optional	The maximum number of datasets to output. Used together with `offset`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/3760c457-d4f3-4d58-8b94-af089a97a
9b4 | jq
```

Output of the example:

```
[
  {
    "id": null,
    "uuid": "08f4d496-67b5-4b7c-b2d2-923458d1f450",
```

```
    "full_name": "John Doe Jr",
    "is_active": false,
    "url_name": null,
    "orcid_id": ""
  },
  {
    "id": null,
    "uuid": "6815031c-21dc-4873-93c9-f6539da482ce",
    "full_name": "John Doe",
    "is_active": false,
    "url_name": null,
    "orcid_id": ""
  }
]
```


6.10.8 `/v2/account/collections/<collection-id>/authors` (`POST`)

Similar to Section 6.9.7, this API endpoint can be used to append authors to the collection identified by `collection-id`.

The following parameters can be used:

Parameter	Data type	Required	Description
`full_name`	`string`	No	The full name of the author.
`first_name`	`string`	Yes	The first name of the author.
`last_name`	`string`	Yes	The last name of the author.
`email`	`string`	No	The e-mail address of the author.
`orcid_id`	`string`	No	The ORCID identifier for the author.
`job_title`	`string`	No	The job title of the author.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "authors": [{ "name": "John Doe" }]}' \
     https://demo.seshat.software/v2/account/collections/3760c457-d4f3-4d58-8b94-af089a97a
9b4/authors
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "authors": [{ "name": "John Doe Jr" }]}' \
     https://demo.seshat.software/v2/account/collections/3760c457-d4f3-4d58-8b94-af089a97a
9b4/authors
```

The following is an example of the output of the HTTP `POST` calls:

```
HTTP/1.1 205 RESET CONTENT
```

An example of the output of the HTTP `GET` call can be found in Section 6.10.7.

6.10.9 `/v2/account/collections/<collection-id>/authors` (`PUT`)

In contrast to Section 6.10.8, this API endpoint can be used to **overwrite** the list of authors of the collection identified by `collection-id`.

Example usage:

```
curl --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "authors": [{ "name": "John Doe" }]}' \
     https://demo.seshat.software/v2/account/collections/3760c457-d4f3-4d58-8b94-af089a97a
```

```
9b4/authors

curl --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "authors": [{ "name": "John Doe Jr" }]}' \
     https://demo.seshat.software/v2/account/collections/3760c457-d4f3-4d58-8b94-af089a97a
9b4/authors

curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/3760c457-d4f3-4d58-8b94-af089a97a
9b4/authors | jq
```

Output of the example:

```
[
  {
    "id": null,
    "uuid": "61751fe3-53a1-477f-a46f-e534cbd0b618",
    "full_name": "John Doe Jr",
    "is_active": false,
    "url_name": null,
    "orcid_id": ""
  },
]
```

6.10.10 /v2/account/collections/<collection-id>/authors/<author-id> (DELETE)

This API endpoint can be used to delete an author's association with a collection.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d//authors/5c75...94aa
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.10.11 /v2/account/collections/<collection-id>/categories (GET)

This API endpoint can be used to retrieve the categories associated with the collection identified by `collection-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/categories | jq
```

Output of the example:

```
[
  {
    "id": 13376,
    "uuid": "2bdba8f2-5914-4d82-bfe8-c938cccab71f",
    "title": "Agricultural and Veterinary Sciences",
    "parent_id": null,
    "parent_uuid": null,
    "path": "",
    "source_id": null,
    "taxonomy_id": null
  }
]
```

6.10.12 /v2/account/collections/<collection-id>/categories (POST)

Similar to Section 6.9.15 this API endpoint can be used to append categories to the collection identified by `collection-id`. The parameters sent to the server should be a JSON object with a single key named `categories`, with as value a list of either the numeric or the UUID identifiers for a category. The API endpoint described in Section 6.9.14 shows how to obtain the category identifiers.

Example usage:

```
curl --verbose --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "categories": [13551, 13558]}' \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/categories
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.10.13 /v2/account/collections/<collection-id>/categories (PUT)

In contrast to Section 6.10.12, this API endpoint can be used to **overwrite** the list of categories of the collection identified by `collection-id`.

Example usage:

```
curl --verbose --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "categories": ["dd4dbaaf-0610-4d8d-8b07-e1eeb32dd11c"]}' \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/categories
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.10.14 /v2/account/collections/<collection-id>/categories/<category-id> (DELETE)

This API endpoint can be used to delete a category's association with a collection. The `category-id` can be either the `uuid` or the `id` property.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/categories/13558
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.10.15 /v2/account/collections/<collection-id>/articles (GET)

This API endpoint can be used to retrieve the datasets associated with the collection identified by `collection-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/articles | jq
```

Output of the example:

```
[ /* This example has been shortened. */
  {
    "id": null,
    "uuid": "8050f9cb-d0b0-4149-bd24-02f13c2410db",
    "doi": "10.4121/8050f9cb-d0b0-4149-bd24-02f13c2410db.v1",
    ...
```

```
  },
  {
    "id": 14309234,
    "uuid": "06431360-776c-45c6-bcca-ec898f2870ff",
    "doi": "10.4121/14309234.v1",
    ...
  }
]
```

6.10.16 `/v2/account/collections/<collection-id>/articles` (`POST`)

This API endpoint can be used to append datasets to the collection identified by `collection-id`. The API endpoint accepts both the `id` property and the `uuid` property of a dataset as identifier.

The parameters sent to the server should be a JSON object with a single key named `articles`, with as value a list of either the `id` or the `uuid` identifiers for datasets. The API endpoint described in Section 6.1.2 shows how to obtain the dataset identifiers.

Example usage:

```
curl --verbose --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --header "Accept: application/json" \
     --data '{ "articles": ["8050...10db", 14309234 ]}' \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/articles
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.10.17 `/v2/account/collections/<collection-id>/articles` (`PUT`)

In contrast to Section 6.10.16, this API endpoint can be used to **overwrite** the list of datasets associated with a collection.

```
curl --verbose --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --header "Accept: application/json" \
     --data '{ "articles": [ 14309234 ]}' \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/articles
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.10.18 `/v2/account/collections/<collection-id>/articles/<dataset-id>` (`DELETE`)

This API endpoint can be used to delete a dataset's association with a collection. The `dataset-id` can be either the `uuid` or the `id` property.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/articles/8050...10db
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.10.19 `/v2/account/collections/<collection-id>/private_links` (`GET`)

This API endpoint lists the private links associated with the collection identified by `collection-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/f000...3c16/private_links | jq
```

Output of the example:

```
[
  {
    "id": "xk84wQ0Vxn5Xf44cAjPlg4Bjw-sbrHoia_SgboWF6eo",
    "is_active": true,
    "expires_date": null
  },
  {
    "id": "EYgjp5RGul2maCblF6Mlx0lfFombnPLLYniTS7hLD2E",
    "is_active": true,
    "expires_date": "2027-01-01T00:00:00"
  }
]
```

6.10.20 `/v2/account/collections/<collection-id>/private_links` (`POST`)

This API endpoint can be used to append a private link to the collection identified by `collection-id`.

The following parameter can be used:

Parameter	Data type	Required	Description
`expires_date`	`string`	No	The format of the date string should be `YYYY-MM-DD`.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --data '{ "expires_date": "2032-01-01", "read_only": false }' \
     https://demo.seshat.software/v2/account/collections/f000...3c16/private_links | jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "location": "https://demo.seshat.software/private_collections/Jk2I..."
}
```

6.10.21 `/v2/account/collections/<collection-id>/private_links/<link-id>` (`GET`)

This API endpoint can be used to view the details of a private link for the collection identified by `collection-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/f000...3c16/private_links/Jk2I...
| jq
```

Output of the example:

```
[
  {
    "id": "Jk2Iw1cOKqQ0EJbx01feLKcJOukIcBCDJ20HFbe9F8g",
    "is_active": true,
    "expires_date": "2032-01-01T00:00:00"
  }
]
```

6.10.22 `/v2/account/collections/<collection-id>/private_links/<link-id>` (`PUT`)

This API endpoint can be used to update the expiry date of a private link and whether the private link is active or not for the collection identified by `collection-id`.

The following parameter can be used:

Parameter	Data type	Required	Description
`expires_date`	`string`	No	The format of the date string should be `YYYY-MM-DD`.
`is_active`	`boolean`	No	Defaults to `false`.

Example usage:

```
curl --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --data '{ "expires_date": "2034-01-01", "is_active": true }' \
     https://demo.seshat.software/v2/account/collections/f000...3c16/private_links/Jk2I...
| jq
```

Output of the example:

```
{ /* Example output has been shortened. */
  "location": "https://demo.seshat.software/private_collections/Jk2I..."
}
```

6.10.23 `/v2/account/collections/<collection-id>/private_links/<link-id>` (`DELETE`)

This API endpoint can be used to remove a private link for the collection identified by `collection-id`.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/f000...3c16/private_links/Jk2I...
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

6.10.24 `/v2/account/collections/<collection-id>/reserve_doi` (`POST`)

This API endpoint can be used to obtain the DOI before the collection is published and the DOI is activated.

Example usage:

```
curl --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/reserve_doi | jq
```

Output of the example:

```
{
  "doi": "10.5074/fc03a4c3-cba4-4a88-a8a6-eb38924eeb6d"
}
```

6.10.25 `/v2/account/collections/<collection-id>/funding` (`GET`)

This API endpoint lists the funding of the collection identified by `collection-id`.

Example usage:

```
curl --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/funding | jq
```

Output of the example:

```
[
  {
    "id": null,
    "uuid": "6f605fe1-e87a-43f5-8b67-70ebe3f9b868",
    "title": "Example cases fund",
    "grant_code": "EXA-001",
    "funder_name": "Example",
```

```
    "is_user_defined": null,
    "url": "https://example.exa"
  }
]
```

6.10.26 /v2/account/collections/<collection-id>/funding (POST)

This API endpoint can be used to append funders to the collection identified by `collection-id`.

The following parameters can be used:

Parameter	Data type	Required	Description
`uuid`	`string`	No	The uuid of an existing funding record. When this parameter is set, other parameters will be ignored.
`title`	`string`	Yes	The title of the funding project.
`grant_code`	`string`	No	An optional grant code of the funding.
`funder_name`	`string`	No	The name of the funder.
`url`	`string`	No	A URL to the funding project or funding organization.

Example usage:

```
curl --verbose --request POST \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "funders": [{ "title": "Example cases fund", \
                             "grant_code": "EXA-001", \
                             "funder_name": "Example", \
                             "url": "https://example.exa" }]}' \
     https://demo.seshat.software/v2/account/collections/fc03a4c3-cba4-4a88-a8a6-eb38924
eeb6d/funding
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.10.27 /v2/account/collections/<collection-id>/funding (PUT)

In contrast to Section 6.10.26, this API endpoint can be used to **overwrite** the list of funders of the collection identified by `collection-id`.

```
curl --verbose --request PUT \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     --header "Content-Type: application/json" \
     --data '{ "funders": [{ "title": "Example cases fund",
                             "grant_code": "EXA-001",
                             "funder_name": "Example",
                             "url": "https://example.exa" }]}' \
     https://demo.seshat.software/v2/account/collections/fc03a4c3-cba4-4a88-a8a6-eb38924
eeb6d/funding
```

HTTP response of the example:

```
HTTP/1.1 205 RESET CONTENT
```

6.10.28 /v2/account/collections/<collection-id>/funding/<funding-id> (DELETE)

This API endpoint can be used to delete an funder's association with a collection.

Example usage:

```
curl --request DELETE \
     --header "Authorization: token YOUR_TOKEN_HERE" \
     https://demo.seshat.software/v2/account/collections/fc03...eb6d/funding/9b43...e6cd
```

HTTP response of the example:

```
HTTP/1.1 204 NO CONTENT
```

Bibliography

Lassila, O. (1999). *Resource Description Framework (RDF) Model and Syntax Specification* [W3C Recommendation].

SPARQL 1.1 Overview. (2013). [W3C Recommendation].

Contact

The team

This software is voluntarily maintained by Malaika Newman, Roel Janssen and Guillaume Dubois.

Mailing lists

You can best reach us by e-mailing development@seshat.software.

Reporting security vulnerabilities

For security-related matters, please e-mail security@seshat.software.

News

Release notes for `v26.5.`

The May release of 2026 consists of 39 commits made by 3 authors.

New features

- Add OCI registry v2 API (199b30be5, 92760e5a0, 5046a0131, d88892d9a).
- Add a v3 API call to revoke API tokens from the dashboard (f5eba830e, bda7af831).
- Add a `disable-account-creation` configuration option (524ccdf9b).
- Add a dark-mode theme to the HTML documentation (90097098e, 480b383af).

Bugfixes

- Restore logging in via an external identity provider by relaxing the cookie's `SameSite` attribute during the login redirect (776bbf888).
- Prevent duplicate account records when logging in with ORCID (c5a0c6301).
- Return the proper error in `api_v3_dataset_upload_file` instead of silently ignoring it (bc1cfbb8a).
- Install the systemd service file and example configuration into the standard system locations for distribution packages (90acbcfaf, 555a0e55f, 30bd6d86c).
- Use predictable links to chapters in the documentation (99fe2dba4).

Incremental improvements

- Update project website and e-mail addresses (722238f2e, 7a4fc5c53)
- Consistently use rounded corners on chapters in the documentation (9738ff433).

Technical debt

- Remove the OpenDAP-to-DOI mapping functionality (b8d3f6240).
- Remove obsolete endpoints, including `/portal`, `/browse`, the account home page, and the collection private-link delete endpoint (10f2d7912, 2f479a600, 4b9bbf29b).
- Finish moving `sparql_is_up` to the configuration module (9829cfed5, b21f9dd5e).
- Remove dead code and address linter complaints (b50bd84b8, e3b8b2474).
- Remove the unused "AUTHORS" lock (9655448a3).
- Remove a legacy license from the license list (8f72508d1).

Release notes for `v26.4.`

After a delay of 10 months, we are delighted to announce the April release of 2026. It consists of 167 commits made by 4 authors.

This is a special moment, because this is the first release under the new name: Seshat. In Egyptian mythology, Seshat is the true keeper of wisdom; the "Mistress of the House of Books," goddess of writing, measurement, and record. She was venerated long before Djehuty, and the scribal domains he is now celebrated for were hers first. She measured the foundations of temples with her own cord and kept the archives of what was true. Djehuty's later fame is largely inherited renown, built on what Seshat had already established.

New features

- Add folder uploading via drag-and-drop in the depositor UI (de693d0e4, 41fa6d401).
- Bring collection private links on par with dataset private links (6ec74fcc8, 337dd990a, ce72a1fee, 8133eed81, 444fde680, 3a1db7b31).
- Display banner automatically when database is down (cfeec92cf).
- Add ability to build a Debian package. (9b240c39d, 110bb308c, 339648c32, 9951c0356, 2d584f812, fc1d39804, 4273b51d2).
- Make cloud-native deployments more convenient to implement (aa4633db2, 408de87dc, 9981890ca).
- Document multi-instance and feature-isolation deployments (63fe1a395).

Security

- Constrain resource usage upon cloning a Git repository (d2f0d61ec).
- Do not resolve symbolic links when producing a ZIP file (e27148dde, 96d7eed0b).
- Validate the value of the "Git-Protocol" HTTP header (a0b63f501).
- Avoid creating Git repository directories before input validation (0de6bba24).
- Strictly validate the "order" value in 'sparql_suffix' (2bc0f0ac9).
- Prevent SPARQL injection of key in search API (42e48983c).
- Prevent SPARQL injection of faking an operator in search API (dc327f196).
- Add `SameSite=Strict` attribute when setting cookies (8edd18b11).

Bugfixes

- Restore compatibility with Python 3.9 (dde1766d2).
- Avoid creating invalid expiry dates for private links (dbf3b3e0d).
- Clean up temporary directory of a Git clone when an error occurs (4cbb2856f).
- Fix rendering of DOI in private viewing (d93ff4b31).
- Preserve updates on statistics upon rebuilding the database (811c5a959).
- Restore styling of the maintenance page (5d8aa1db9).
- Always clean up the temporary directory when zipping a Git repo fails (9b31d5f51).
- Repair ability to remove references with quotes and ampersands (ab2bd85b2).
- Avoid SPARQL FILTER bug in Virtuoso when querying private links (b584dcbee).
- Avoid displaying error messages when querying an empty Git repository (5523164c8).
- Don't assume 'last_name' is set (87cabf943).
- Fix error message in 'api_v3_dataset_upload_file' (345c3dc65).
- Produce valid HTML on the dataset page (332345080).

Incremental improvements

- Simplify steps to install `seshat` (0a6cf9685).
- Rename project to `seshat` (5b12bf64e, d3c2d88c0, b2901d862, 68b2cf073, defdb8c25, 0abcac39d, 379aad157, f5bf0b6a0, 998414a45, c71577a2a).
- Add a manpage (542a053d5).
- Migrate documentation from LaTeX to Typst (a0194d045, 7629ff826, 932b85cc1, f6164b16f).
- Improve documentation theming and navigation (cbdae4306, b162e054b, c0ec49b47, 6890793cb, b74da37f6, b585cadaf, 09755468d, 1e325628b, fdbbdc337).
- Update Quill to 2.0.3 (86bdc5c5f, 03136bed4).
- Make deleting datasets and collections HTTP `DELETE` operations. (f36ae865a, 239b85139).
- Refactor inline JavaScript on the My Datasets page (0a706e3ed).
- Reduce duplicated code for 'new_author' and 'new_funding' (83e634834, 8e7a36264).
- Various small JavaScript cleanups (d5322b576, 1928b5a52, ec8f82a20, 72ffe9c19, 14516add8).
- Clean up HTML templates (b284b938d, 65d9a8b68, 0bdf999c9, 89065c2f0, b0aca1801, 2cf682357).
- Unify timestamp format in the WSGI layer (0e9c5cd76).
- Avoid implicit variable use in a 'map' call inside a loop (a67bf3aa5).
- Apply minor Python and linter cleanups (0ce3563cb, 57a7798fd, 13c52e885, 652c9f1d4).
- Include "python-pip" in the Windows installation instructions (cc603327a).
- Directly refer to the documentation for API usage (51860bfbd).
- Improve the build system for RPM packaging (54ff37550, a476312f9, 163017179, b95c12314, 34f7c789f).
- Improve the build system in general (6c74d8e4b, 4ee1e7872, c3c52f253, 3914b5d57).

Technical debt

- Remove the "backup" subcommand (e688f1cd7).
- Remove hard-coded 4TU.ResearchData-isms (18e3e79f2, b0bd9d974, e08c1f019, 1edd40f52).
- Link to `choosealicense.com` in help text (404f713f6).

Release notes for `v25.6.`

The June release of 2025 consists of 15 commits made by 3 authors.

New features

- Add initial support to display a CodeWorks badge (`4b3d5212b`).
- Add auto-completion support for keywords (`032445341`).

Bugfixes

- Return empty list upon error in `/v3/datasets/<uuid>.git/contributors` (`5a7c70c06`).
- Avoid duplicating entries in the funding autocompletion (`a53887390`).

Incremental improvements

- Improve documentation (`3458987db`, `9edb43e9c`, `f15e95f9d`).
- Improve the caching system (`4b1c054f7`, `4cee56a6f`).
- Introduce permission to recalculate statistics (`5c6beab77`).
- Improve contrast of the "connect with ORCID" button (`c1b4d1bfc`).

Release notes for `v25.5.`

The May release of 2025 consists of 14 commits made by 3 authors.

This release contains a security fix for a SPARQL injection found by Thomas Thelen and a security fix for a HTML injection found by Anass Ksiber. Many thanks to both for reporting and assisting in resolving these vulnerabilities.

UI revisions

- Introduce an "Interoperability" section with links to the RO-Crate metadata API and the IIIF manifest (`2a49687d0`).

Security

- Properly escape session cookie value (`da1cbf2b1`).
- Avoid possibility of HTML injection in the search page (`4f479f686`).

Bugfixes

- Avoid re-creating the Handle configuration (`80f1f2e3e`).
- Ensure the v2 API respects the depositing-domains property (`45941d2d9`).
- Don't show file metadata for restricted datasets in RO-Crate output (`b81c730ec`).
- Improve render quality of PDF files in the IIIF Image API (`6556bf2ff`, `ffb35961a`).
- Document acceptable parameters for various API endpoints (`b5dea019b`).
- Distribute missing files in the release tarball (`76b29514e`).

Technical debt

- Simplify the `dist-docker` target (`7b08aae0f`).

Release notes for `v25.4.`

The April release of 2025 consists of 51 commits made by 4 authors.

New features

- Implement IIIF Presentation API (`1eb35f4f4`, `b052e5dc7`, `9dca2ffa5`, `68f2bbacd`, `ed0a39e19`, `8f975910f`, `ce43c4590`, `a4b43cdd7`).
- Implement tiles property for IIIF Image API (`e9453e1ec`).
- Implement `static-resources-cache` option (`1529d961a`).

Security

- Harden the `Content-Security-Policy` as an extra layer of defence against cross-site scripting (`3c2c53290`, `ce428d824`, `4a1abf60e`, `da39ace8e`, `1bcc4f92d`, `13c06ef7b`, `b5434f969`, `7a157070a`, `9d84583c8`).

Bugfixes

- Ensure ZIP files of Git repositories are bit-reproducible (586c30458).
- Fix alignment of search results when viewed as a list (2463ebbcf).
- Document the institution API endpoints (fc27ddf7a).
- Show reviews for institutional reviewers based on the group rather than accounts (6ef0eca6c).

Technical debt

- Code clean-ups (02ce6b383, a1e7c2fe6, 1a3adcfcc, c61f27247, 9d6307afa, a4e5b38a7, e60e6f373, 51817a2b5, 96d338254, 2de1e417f, 35afe28e9)

Release notes for v25.3.

The March release of 2025 consists of 57 commits made by 2 authors.

This release contains various bugfixes, minor UI revisions, minor feature updates, and contains the foundation for an extra security layer to prevent cross-site scripting vulnerabilities.

The release date slipped a couple of days because yours truly wanted to give last-minute changes a little bit of time to make sure no regressions occurred before formalizing the release.

New features

- Implement API endpoints for reviewers (1de7f6808, 038e931b9, 2f5963553).
- Report number of search results in the `/v2/articles/search` endpoint. (a8917a837).
- Add `SoftwareSourceCode` to RO-Crate output (79cf0b32a).

UI revisions

- Revise the "Cite" and "Collect" buttons on landing pages (b0b9dbd1f).
- Remove the need for a "save URL" button in the dataset metadata form (a853085c9).
- Revise the versions drop-down menu on landing pages (e5b89ce23).
- Fix tile scaling on the main page for different zoom levels (7a30bfa47).

Security

- Addressed a Cross-Site-Scripting vulnerability in the search functionality (40b12a559).
- Only display e-mail address of authors to the creators of such records. (05a56fa18).

Bugfixes

- Fix author ordering for collections. (244017a01).
- Fix bug in cached responses in the IIIF Image API implementation (88d68c787).
- Fix bug with proportional scaling in the IIIF Image API implementation (05d5c7a85).
- Fix various bugs with rendering HTML entities and tags. (d3667ed8b, 3b19d7de8, 08e4fc77a, 3a1f3dde5).
- Avoid a divide-by-zero situation with quota usage calculation (cefde15dd).
- Fix creating datasets with repeated fields using the v2 API (74fe025db, 87127c1f8).
- Fix setting default fields when creating a dataset using the API (7f183389d).
- Fix returning Git statistics for empty Git repositories (be3630a63, 2cffe955a, c3227a768).

Technical debt

- Work towards a stricter Content-Security-Policy by avoiding inline use of `style` attributes, `script` elements, and event handlers (b862fdf3d, 18b3bbe3e, f08542ecb, 1c248a1e2, 99cf348f8, 7524bbbd2, 4d6696335).
- Avoid hard-coded versions in the documentation for the RPM download links (21be87dc0).
- Avoid repetitive text in the documentation by using macros (088f8a13c, f3dc9c8cd).
- Build Docker images with C development libraries to work around "xmlsec" build issues. (2c03cb3cb).

Release notes for v25.2.

The February release of 2025 consists of 75 commits made by 2 authors.

The major new features in this release are initial support for RO-Crate and direct support for S3 buckets. The release was delayed to include an important security fix for a Cross-Site-Scripting vulnerability found by Aaron Liebig. Many thanks for reporting and assisting in resolving this vulnerability.

New features

- Implement initial support for RO-Crates (`5ffee87a7`, `96ee261ff`).
- Implement support for S3 buckets via `boto3` (`1077e2503`, `67fe48dbc`, `2e5394b41`, `09e20e405`, `30209a754`, `7e6d147fc`).
- Implement ability to extract/replay log entries (`a5a2c0903`).

Security

- Addressed a Cross-Site-Scripting vulnerability (`38e89a0c8`).

Bugfixes

- Improve rendering for accessibility (`a3e456cf1`, `7cb7654fc`, `11bba9598`, `c74ef45f1`).
- Apply user interface updates (`b2c213ac9`, `3f13e0c75`, `98606dfcc`, `858f7b7aa`, `d4db44603`, `c327f196f`, `22107051e`, `c87550cc5`, `6642dcf94`).
- Enhance the documentation (`57192bbb8`, `3a864f07a`, `be65be335`, `ea4e623db`, `1c6267ec4`, `8d4d81266`).
- Show precise error messages when input validation fails in the edit-dataset and edit-collection forms (`bfd2110f7`).

Technical debt

- Remove `urllib3` as explicit dependency (`672eaff0c`).
- Prevent comments in query templates from being sent to the SPARQL endpoint (`9ac703556`).

Release notes for `v25.1.`

The January release of 2025 consists of 85 commits made by 3 authors.

In this release we included an RPM package for Enterprise Linux 9. This RPM depends on packages in the Extra Packages for Enterprise Linux (EPEL) repository.

New features

- CodeMeta API output is more complete (`91ed59d86`, `860edfec5`, `3f150a246`, `5daa47e5f`).
- A second port to bind the web service on can now be configured (`dff68a6d6`).
- Enable searching by author (`18a242fcd`, `fe0957865`).
- Enable institution reviewing (`cbd1092cc`, `cb67f83bf`).
- Improve indexability by search engines (`cc5848390`, `20fe6fd9a`).

Bugfixes

- Related versions of a dataset are communicated to DataCite (`1539117be`).
- HTML output of the documentation is responsive to browser widths (`934ed9310`).
- Restore ability to create new collection versions (`545de472c`).
- Display embargoed datasets in the search results (`c86dcf85b`).
- Fixed building RPM packages (`1f629fa0d`, `e4c10571a`).
- Fixed HTTP `PUT` behavior for `\/v2\/account\/collections\/<id>\/articles` (`107ea693e`).

Technical debt

- Unified the development environment instructions between GNU/Linux, Windows and macOS (`8921d35c1`, `ca9b5831c`).
- Run-time configurable properties are stored in a separate module (`a8e353db6`).
- Improve error handling (`bdf77edd8`, `23c3b53d2`, `609c9864e`).
- Embed simplified 'zipfly' (`0fe0904a2`).

Release notes for `v24.12`.

New features

- Add specific logging for when the server would respond an HTTP 500 error (`1dcd141f7`).

Bugfixes

- Fix a problem with downloading Git repositories as ZIP (`a8c2da4c5`).
- Avoid returning an internal server error when using paging in the API (`2629ef56f`).
- Fix lay-out bug on the landing pages (`a3ff6a54e`).
- Fix bug when filtering on groups in the API (`187f4344f`).

Technical debt

- Refactor parts of the codebase (`0370836d5`, `dd7602a66`, `aa8eda2fc`, `3fb51635c`, `dd6ccca45`, `58c1fcedc`)
- Reduce build-system files (`e73966fed`).
- Bump the minimal required Python version to 3.9 (`208c7e08f`).