ECAS-B2SHARE

Sep 05, 2019

Contents:

This is a short documentation about the ecasb2share python library. You can use this client to create a draft record with minimal metadata in B2SHARE.

Important: the client is related to the ECAS use case and should be used in ECASLab.

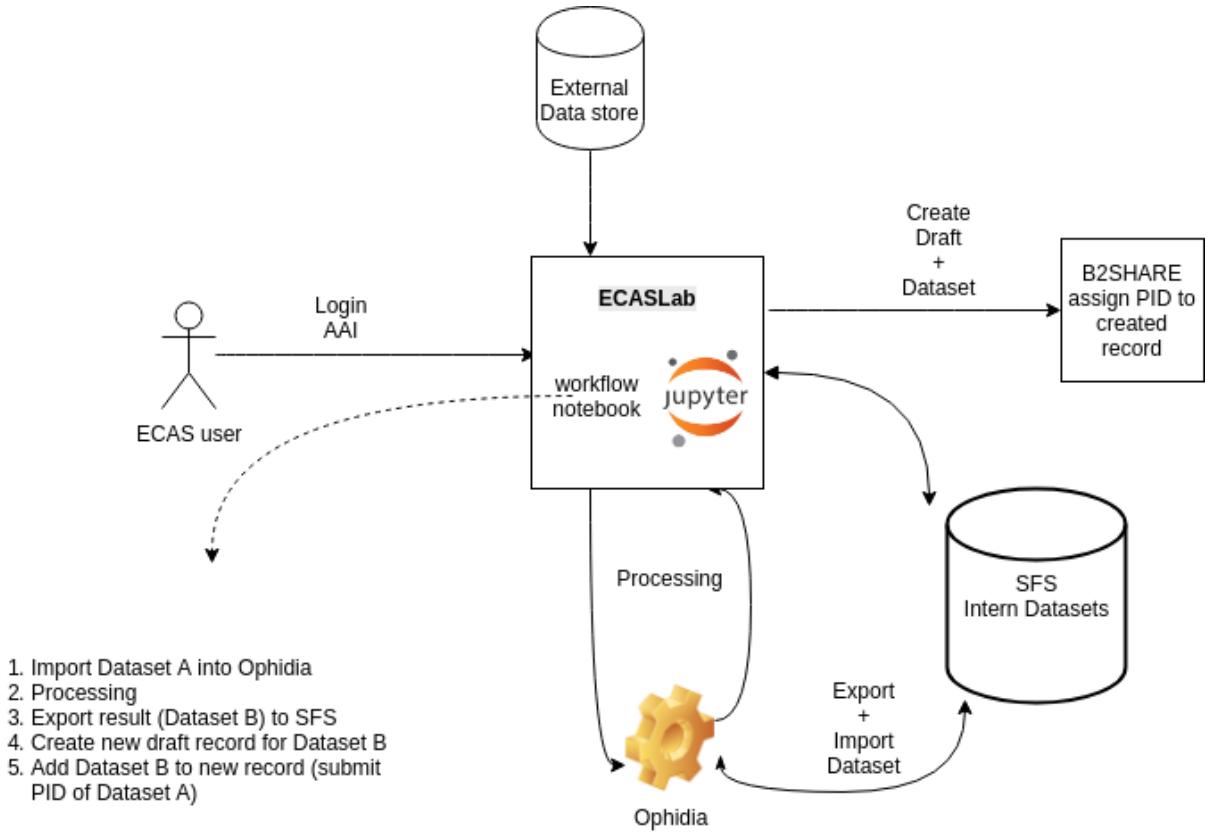
CHAPTER 1

How to install?

```
pip install ecasb2share
```

CHAPTER 2

Sharing with B2SHARE



B2SHARE is a user-friendly, reliable and trustworthy way for researchers, scientific communities and citizen scientists to store and share small-scale research data from diverse contexts.

In the context of ECAS, B2SHARE is integrated for two major reasons:

- enable publishing of results (data, notebooks, etc) directly from the **ECAS** environment
- enhance research reproducibility

Sharing data from ECAS to B2SHARE is performed using the ecasb2share Python library. The library is already installed in each user environment and can be called directly from the jupyter notebook. It helps to create draft records with minimal metadata and upload different types of files.

Note: After the creation of a draft record, it is recommended to switch to the B2SHARE graphical user interface to check the correctness of the metadata and submit request for publication. These features are available in the user workspace.

Note: The ECAS-B2SHARE python client covers only the mandatory steps when creating new records in B2SHARE. Please consult user documentation for more details about the service.

In B2SHARE there are different research communities to store the records. An extra community has been added to cover the ECAS use case with a specific metadata schema.

2.1 Requirements

In order to upload and share ECAS data with B2SHARE, users need to register for an account. Please use this link to create a new account.

2.2 How to use the client?

1. initialize the client:

```
from ecasb2share.ecasb2shareclient import EcasShare as Client
client = Client(url='{b2share-url}', token_file='{path to token}')
```

default url is https://b2share.eudat.eu

```
help(client)
```

2. create a draft record

This methods reads metadata from a json file, creates a draft record and returns useful informations.

```
client.create_draft_record_with_pid(metadata_json='{metada json file}')
```

Note: it is possible to pass the metadata to the method manually but it is recommended to use the json file to avoid syntax validation issues in case of multiple related identifiers.

2.2.1 Example of metadata:

```
{"titles": [{"title": "ECAS TEST Multiple PIDs"}],
 "community": "d2c6e694-0c0a-4884-ad15-ddf498008320",
 "related_identifiers": [
                          {
                            "related_identifier": "original_pid1",
                            "related_identifier_type": "Handle",
                            "relation_type": "IsDerivedFrom"
                          },
                          {
                            "related_identifier": "original_pid",
                            "related_identifier_type": "Handle",
                            "relation_type": "IsDerivedFrom"
                          }
                        ],

 "open_access": true
 }
```

CHAPTER 3

API

EcasShare.**__init__**(*url=None*, *token_file=None*)
Initialize the client.

Parameters **url** – URL of the B2SHARE instance. One of the following

hostnames can be used to identify the B2SHARE instance:

- b2share.eudat.eu - the hostname of the production site.
- trng-b2share.eudat.eu - the base url of the training

site. Use this URL for testing.

Parameters **token_file** – B2SHARE API ACCESS token

EcasShare.**retrieve_access_token**()
Read the token from a given file named 'token'

EcasShare.**list_communities**(*token=None*)
List all the communities, without any filtering.

Parameters **token** – Optional: B2SHARE API ACCESS token

Returns list of communities in json

EcasShare.**retrieve_community_specific_records**(*community_id*)
List all records of a specific community.

Parameters **community_id** – community id. Can be retrieved from the list of

communities *list_communities* :return: the list of records (in JSON format) or an exception message

EcasShare.**get_community_schema**(*community_id*)
Retrieves the JSON schema of records approved by a specific community.

Parameters **community_id** – community id. Can be found from the list of

communities *list_communities*

Returns community schema in json format.

EcasShare.**list_all_records**(*size=None*)
List all the records, without any filtering TODO add pagination :param size: Optional :return: list of records in json format.

EcasShare.**get_specific_record**(*record_id*, *draft=True*)
List the metadata of the record specified by RECORD_ID.

Parameters

- **record_id** – record id.
- **draft** – True/False to specify which type of record to search for. Default: True.

Returns list of records.

EcasShare.**get_record_pid**(*record_id*)
Get the pid from the record metadata (published).

Parameters **record_id** – record id

Returns epicPID, prefix/suffix

EcasShare.**create_draft_record**(*community_id*, *title*)
Create a new record with minimal metadata, in the draft state.

Parameters

- **community_id** –
- **title** – title for the record

Returns record_id, filebucket_id

EcasShare.**create_draft_record_with_pid**(*title=None*, *original_pid=None*, *metadata_json=None*)
Create a draft record and specifying the original pid. Adapted from DataCite schema:

- relatedIdentifierType: Handle
- relationType: isDerivedFrom

Parameters

- **title** – title for the record.
- **original_pid** – PID (prefix/suffix) of the input Dataset.

Returns record_id and filebucket_id

EcasShare.**submit_draft_for_publication**(*record_id*)

Parameters **record_id** – record id

Returns request status (HTTP response)

EcasShare.**delete_draft_record**(*record_id*)

Parameters **record_id** – record id

Returns request status

EcasShare.**search_records**()
List all the records, without any filtering

EcasShare.**get_filebucketid_from_record**(*record_id*)
TODO add exception when record not found

Parameters **record_id** – identifier for a specific record, which can be in draft or published state

Returns filebucket id.

EcasShare.**search_drafts**()
Search for all drafts (unpublished records) that are accessible by the requestor. Usually this means own records only.

Returns the list of matching drafts (in JSON format).

EcasShare.**search_specific_record**(*search_value*)

EcasShare.**add_file_to_draft_record**(*file_path*, *filebucket_id*)

Parameters

- **file_path** – path to the file to be uploaded.
- **filebucket_id** – identifier for a set of files. Each record has its own file set, usually found in the links -> files section

Returns request status

EcasShare.**list_files_in_bucket**(*filebucket_id*)
List the files uploaded into a record object.

Parameters **filebucket_id** – identifier for a set of files. Each record has its own file set, usually found in the links -> files section

Returns information about all the files in the record object

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index

Symbols

A

C

D

G

L

R

S