B2SHARE HTTP REST API

About

User documentation about the EUDAT B2SHARE HTTP REST API.

Modified: 26 March 2019

B2SHARE version: 2.1.x

Synopsis

B2SHARE is the EUDAT service for storing and publishing data sets. In addition to its web-based GUI, B2SHARE offers an HTTP REST API. External applications or workflows can use this API, for example, to integrate B2SHARE with other websites like research community portals, or to allow uploads and downloads of large data sets that are not easily handled via a web browser. The API functionality is also used for metadata harvesting by external metadata catalogue services, including B2FIND.

What is B2SHARE?

B2SHARE is the EUDAT service for storing and publishing scientific data. B2SHARE is designed to be easy to use and currently supports access restrictions, registration of a PID for any uploaded data object, add on of checksums and transition of all metadata information to the EUDAT metadata search. Importantly, B2SHARE enforces the inclusion of metadata accompanying the deposited data, so as to increase the value and facilitate sharing of your assets. B2SHARE fosters Open Access to data and includes a tool to help the user choose the correct licence for their data. B2SHARE should be used to publish data that are not meant to change. For transient data, please consider B2DROP.

The REST API

B2SHARE provides a graphical, web-based interface, whose functionality is discussed in the relevant User Documentation page.

The B2HARE HTTP REST API can be used for interaction with B2SHARE via external services or applications, for example for integration with other web-sites (research community portals) or for uploading or downloading large data sets that are not easily handled via a web browser. The API can also be used for metadata harvesting, although an OAI-PMH API endpoint is also provided for this purpose. This latter API will not be discussed here.

This page will explain the basic concepts, authentication and all existing HTTP requests that can currently be used. The given examples are explained using curl commands. For usage of the API with Python, please follow the training material provided by EUDAT.

Basic concepts

The B2SHARE service uses several concepts which are briefly explained below.

A scientific **community** has the roles of creating and maintaining metadata schemas and curating the datasets which are part of a scientific domain or research project. B2SHARE users can be part of one or more communities. Some selected members of a community are also given the role of community administrators, which grants them

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

the special rights needed for the schema definitions and record curation tasks.
Any user can upload scientific datasets into B2SHARE and thus create data **records**. A record is comprised of data files and associated metadata. The record’s **metadata** consists of a set of common fixed metadata fields and a set of custom metadata blocks, each block containing related metadata fields. A record is always connected to one scientific community which has the role of curating and maintaining it.

A record contains a set of common metadata fields and a set of custom metadata blocks. This metadata is not free form, however, but is governed by static schemas; the common metadata schema is set by B2SHARE and defines a superset of Dublin Core elements, while the schema for the custom metadata block is specific to each community and can be customized by the community administrators. The schemas are formally defined in the JSON Schema format. A special HTTP REST API call is available for retrieving the JSON Schema of a record in a specific community. In order to be accepted, the records submitted to a community must conform to the schema required by the community.

Editing and versioning records

A data record can exist in several states. Immediately after creation a record enters the 'draft' state. In this state the record is only accessible by its owner and can be freely modified: its metadata can be changed and files can be uploaded into or removed from it. A draft can be published at any time, and through this action it changes its state from 'draft' to 'published', is assigned Persistent Identifiers, and becomes publicly accessible. **Please note that the list of files in a *published record* cannot be changed without versioning the record**.

To update the metadata of a record through the API, a JSON Patch must be supplied with the request. Please read the documentation on this website carefully to fully understand how these patches work. In the request below, the term 'JSONPath' is used which indicates a path in the metadata relative to the root of the structure.

Existing published records can be versioned by creating a derivative draft that initially is a clone of the original record. This draft record can be changed in metadata but also files. A link will be established to the original record so that anyone can find and compare the contents of the versioned and original record. There is no limit to the number of versions created per record. A new versioned record needs to be published before it becomes available to other users.

Using the API

The REST API is used by making API requests using the GET or POST methods of the standard HTTP protocol. Typically an application like curl is used to make requests, but it is also possible to do this from within your custom-built applications. To intergrate the API in your application using Python, please refer to the B2SHARE API training module to learn how to use the API.

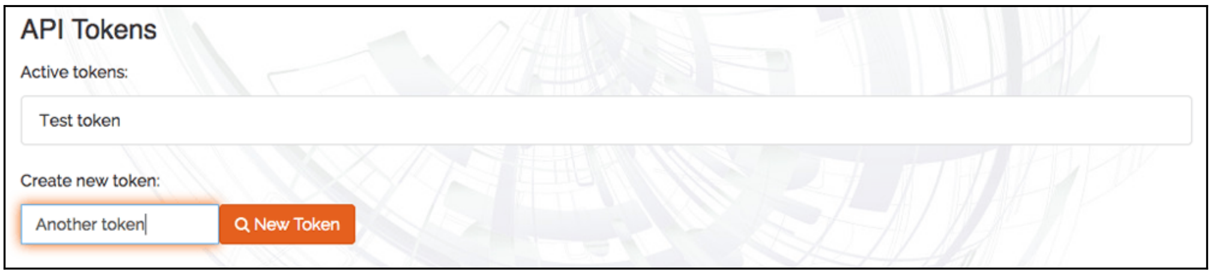
Authentication

Although listing and accessing public data is not access-controlled, only authenticated users can use the API to its full extent. Authentication during requests is done by passing an access token along with the request. The access token is an encrypted string which can be created in the B2SHARE user profile when logged in to the web user interface. B2SHARE’s access tokens follow the OAuth 2.0 standard.

Creating an access token

After logging in to the B2SHARE service click your user name and select 'Profile'. Under 'API tokens', enter a token name (Figure 1).

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.



API Tokens

Active tokens:

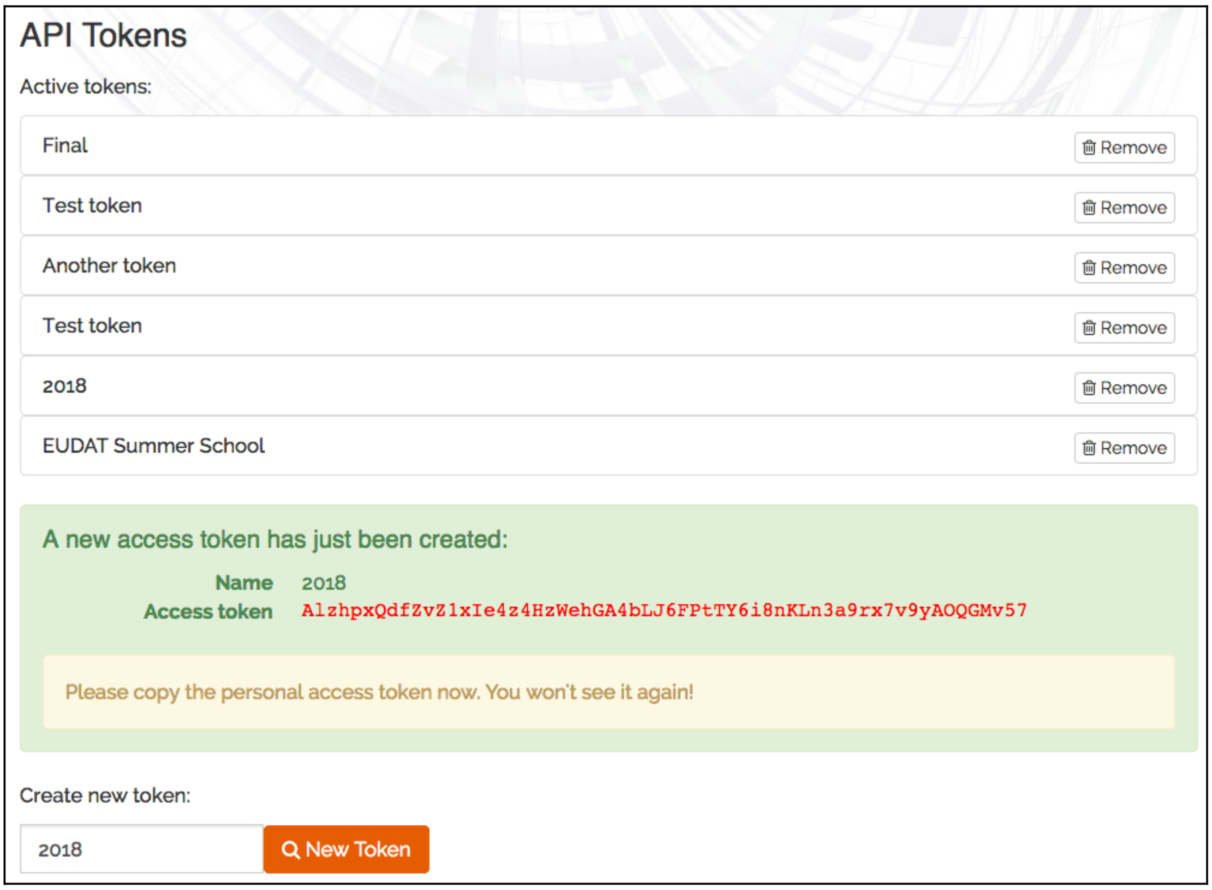
Test token

Create new token:

Another token

Figure 1. B2SHARE Access Token Step 1: Applications.

Click “New token” on the right of the text box. Your token is now created and immediately displayed (Figure 2). Please note that this is the only time the access token is visible, so copy it to a safe place.



API Tokens

Active tokens:

Final

Test token

Another token

Test token

2018

EUDAT Summer School

A new access token has just been created:

Name 2018
Access token AlzhpxQdfZvZ1xIe4z4HzWehGA4bLJ6FPtTY6i8nKLn3a9rx7v9yAOQGMv57

Please copy the personal access token now. You won't see it again!

Create new token:

2018

Figure 2. B2SHARE Access Token Step 2: Create new token.

The access token is now ready for use. Access tokens can be removed by clicking on the remove button next to each access token name.

The following shell commands will expect that the `ACCESS_TOKEN` environment variable is defined and contains the actual access_token. The command to define this variable looks like this:

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
Command:
export ACCESS_TOKEN='7028DlvgCatQV0pkS6jLw947tbo123oztkU4dPw6fnqmJ8inOYAi7dYhF0d04'
```

Notes:

- Please remember to use your actual token instead of the one given as an example above.
- Your token can only be used for the instance you created the token in. That means that a token for the training instance of B2SHARE will not work with the production instance of B2SHARE and vice-versa!

Requests

The API requests are made to a URL with parameters as described below. Each URL consists of a protocol part (always 'https://'), a hostname and a path. One of the following hostnames can be used to identify the B2SHARE instance:

- https://b2share.eudat.eu - the hostname for the production site.
- https://trng-b2share.eudat.eu - the hostname for the training site. Use this base URL for testing.

Notes:

- Please make sure that you are not using production instances for creating test records or testing the API in general.
- Make sure to add a forward-slash ('/') to the URL if that is required. If you forget the slash then the request is interpreted differently and you might get other results than expected. In many cases, a redirect (status code 302) will be returned, a result that in the browser will be handled automatically, but not in a typical API request.

The curl commands in the examples of each request will expect that the HOST environment variable is defined and contains the host part of the targeted B2SHARE site, e.g.:

```
Command:
export B2SHARE_HOST='trng-b2share.eudat.eu'
```

Responses

All request response bodies are in JSON format and UTF-8 encoded.

A record is represented as a JSON object:

```
{
  "field1": "value"
}
```

A collection of records is represented as a JSON array of objects:

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
{
  "collection": [
    {
      "field1": "value",
      "field2": "value"
    },
    {
      "field1": "value",
      "field2": "value"
    }
  ]
}
```

Timestamps are in UTC and formatted according to ISO 8601:

```
{
  "updated": "YYYY-MM-DDTHH:MM:SS.ssssss+00:00"
}
```

Errors

In case a request fails, the body of the response body contains details about the error, for example:

```
{
  "message": "The requested URL was not found on the server.  If you entered the URL
manually please check your spelling and try again.",
  "status": 404
}
```

Herein the message field provides a detailed description of what went wrong, while the code indicates the HTTP status code (equivalent to the request response status code).

Status codes

The request status codes indicate whether the request was successfully received, processed and/or executed. B2SHARE follows the HTTP status codes where possible, a complete list can be found here.

One of the following status codes is returned in case the request was successful:

- `200` - Request was successfully received and executed, see body for results
- `201` - Object created, see body for results
- `204` - No contents, this occurs when for example an object is successfully deleted

In case the request failed, the body of the response usually contains details, and one of the following status codes is returned:

- `400` - Request was not understood
- `401` - User must authenticate first, usually because no access token was provided with the request

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

- `403` - User is not authorized to perform request, missing permission to do so
- `404` - Requested object not found or API endpoint does not exist

Any status code greater then or equal to 500 indicates that internally something went wrong in the server. If in this case the problem persists, kindly report this to EUDAT.

A publication workflow

The HTTP REST API does not impose a specific workflow for creating a record. The following example workflow only defines the most basic steps:

1. Identify a target community for your data by using the HTTP REST API List all communities function
2. Using the community's identifier, retrieve the JSON Schema of the record's metadata. The submitted metadata will have to conform to this schema. Use the Get community schema function
3. Create a draft record: use the Create draft record function
4. Upload the files into the draft record. You will have to use one HTTP request per file. Use the Upload file function
5. Set the complete metadata and publish the record. Use the Submit draft for publication function

Migrating to the B2SHARE v2 HTTP REST API

The following changes are needed for a B2SHARE version 1 client using the old HTTP REST API in order to make it work with B2SHARE version 2 for creating and publishing a record:

1. Identify the unique ID of your target community or communities: see List all communities function
2. Update the URL for creating a new record, from `/api/deposition/` to `/api/records/`; see Create draft record function
3. Update the JSON structure of the newly created records to match the required JSON schema structure, see the Get community schema function
4. Update the file upload calls, making sure that the file bucket url is used instead of the old record URL, see the Upload file function
5. Update the old 'commit' action as described in the Submit draft for publication function

Available HTTP REST API requests

Each allowed request is described as follows:

- Description - A description of the function of the request.
- HTTP method - which HTTP protocol such as GET or POST method is used.
- URL path - grammar for the allowed paths used together with one of the base URLs above.
- Status code - the returned status code upon a successful request.
- Returns - the returned data in the body of the response upon a successful request.
- Example - an example of usage using the program curl from the command line.

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

Some of the requests additionally might have the following information:

- Required parameters - the parameters that need to be added to the URL.
- Required data - the data that needs to be sent with the request, the expected structure is shown in the example.

Variables in the descriptions:

- `RECORD_ID` - identifier for a specific record, which can be in draft or published state
- `RECORD_HEAD_ID` - head identifier for a group of record that are versions of each other
- `FILE_BUCKET_ID` - identifier for a set of files. Each record has its own file set, usually found in the links -> files section
- `COMMUNITY_ID` - identifier of a user community in B2SHARE
- `SCHEMA_ID` - identifier of a metadata schema in B2SHARE
- `FILE_NAME` - name of a file in a specific file bucket
- `FIELD_NAME` - name of a metadata field

For most requests, an example is shown using a curl command. If a payload is sent with the request, this is shown in a structured way below the example. The returned response body and request-specific errors are shown if applicable.

Object retrieval

The following requests concern the retrieval of information about records and communities. Click on a title to show details.

List all communities

List all the communities, without any filtering.

- HTTP method: GET
- URL path: `/api/communities/`
- Required parameters: `access_token`
- Status code on success: `200`
- Returns: the list of communities (in JSON format) or an error message.

```
Command:
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
curl https://$B2SHARE_HOST/api/communities/?access_token=$ACCESS_TOKEN
```

```
Returns:
{
  "hits": {
    "hits": [
      {
        "created": "Tue, 18 Oct 2016 08:30:47 GMT",
        "description": "The big Eudat community. Use this community if no other is suited for
you",
        "id": "e9b9792e-79fb-4b07-b6b4-b9c2bd06d095",
        "links": {
          "self":
"https://trng-b2share.eudat.eu/api/communities/e9b9792e-79fb-4b07-b6b4-b9c2bd06d095"
        },
        "logo": "/img/communities/eudat.png",
        "name": "EUDAT",
        "updated": "Tue, 18 Oct 2016 08:30:47 GMT"
      },
      ...
    ],
    "total": 11
  },
  "links": {
    "self": "https://trng-b2share.eudat.eu/api/communities/"
  }
}
```

Get community schema

Retrieves the JSON schema of records approved by a specific community.

- HTTP method: GET
- URL path: `/api/communities/$COMMUNITY_ID/schemas/last`
- Required parameters: `access_token`
- Status code on success: `200`
- Returns: the community metadata schema, embedded in a JSON object, or an error message.

```
Command:
curl
https://$B2SHARE_HOST/api/communities/$COMMUNITY_ID/schemas/last?access_token=$ACCESS_
TOKEN
```

```
Returns:
{
  "community": "e9b9792e-79fb-4b07-b6b4-b9c2bd06d095",
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
  "draft_json_schema": {
    "$ref":
"https://trng-b2share.eudat.eu/api/communities/e9b9792e-79fb-4b07-b6b4-b9c2bd06d095/schemas/0#/
json_schema",
    "$schema": "http://json-schema.org/draft-04/schema#"
  },
  "json_schema": {
    "allOf": [
      ...
    ]
  },
  "links": {
    "self":
"https://trng-b2share.eudat.eu/api/communities/e9b9792e-79fb-4b07-b6b4-b9c2bd06d095/schemas/0"
  },
  "version": 0
}
```

List all records

List all the records, without any filtering.

- HTTP method: GET
- URL path: `/api/records`
- Required parameters: `access_token`
- Status code on success: `200`
- Returns: the list of records (in JSON format) or an error message.

```
Command:
curl https://$B2SHARE_HOST/api/records/?access_token=$ACCESS_TOKEN
```

```
Returns:
{
  "aggregations": {
    "type": {
      "buckets": [],
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 0
    }
  },
  "hits": {
    "hits": [
      {
        "created": "2016-10-19T11:32:46.095143+00:00",
        "files": [
          {
            "bucket": "473086fc-e125-4389-8483-b8a4f130e181",
            "checksum": "md5:c8afdb36c52cf4727836669019e69222",
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
            "key": "myfile",
            "size": 9,
            "version_id": "324fdf1d-0005-41b1-a9c5-26a8eabd05a2"
          }
        ],
        "id": "a1c2ef96a1e446fa9bd7a2a46d2242d4",
        "links": {
          "files":
"https://trng-b2share.eudat.eu/api/files/473086fc-e125-4389-8483-b8a4f130e181",
          "self": "https://trng-b2share.eudat.eu/api/records/a1c2ef96a1e446fa9bd7a2a46d2242d4"
        },
        "metadata": {
          ...: ...
        },
        "updated": "2016-10-19T11:32:46.095152+00:00"
      },
      ...
    ],
    "total": 51
  },
  "links": {
    "next": "https://trng-b2share.eudat.eu/api/records/?sort=mostrecent&q=&page=2",
    "self": "https://trng-b2share.eudat.eu/api/records/?sort=mostrecent&q=&page=1"
  }
}
```

List records per community

List all records of a specific community.

- HTTP method: GET
- URL path: `/api/records/?q=community:COMMUNITY_ID`
- Required parameters: `access_token`
- Status code on success: `200`
- Returns: the list of records (in JSON format) or an error message.

```
Command:
curl
https://$B2SHARE_HOST/api/records/?q=community:$COMMUNITY_ID?access_token=$ACCESS_TOKE
N
```

```
Returns:
{
  "aggregations": {
    "type": {
      "buckets": [],
      "doc_count_error_upper_bound": 0,
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
      "sum_other_doc_count": 0
    }
  },
  "hits": {
    "hits": [
      {
        "created": "2016-10-24T11:29:27.016892+00:00",
        "id": "f7fddf6f111f4362a9e4661294e2b59e",
        "links": {
          "files":
"https://trng-b2share.eudat.eu/api/files/90ea3483-2792-4483-9392-7d624b610398",
          "self": "https://trng-b2share.eudat.eu/api/records/f7fddf6f111f4362a9e4661294e2b59e",
          "versions":
"https://trng-b2share.eudat.eu/api/records/d855e187e3864ddcaa1b68625866dd78/versions"
        },
        "updated": "2016-10-24T11:29:27.016900+00:00",
        ...: ...
      },
      ...
    ],
    "total": 32
  },
  "links": {
    "next":
"https://trng-b2share.eudat.eu/api/records/?sort=bestmatch&q=community%3Ae9b9792e-79fb-4b07-b6b
4-b9c2bd06d095&size=10&page=2",
    "self":
"https://trng-b2share.eudat.eu/api/records/?sort=bestmatch&q=community%3Ae9b9792e-79fb-4b07-b6b
4-b9c2bd06d095&size=10&page=1"
  }
}
```

Search records

Search all the published records for a query string.

- HTTP method: GET
- URL path: `/api/records/?q=$QUERY_STRING`
- Required parameters: `access_token`
- Status code on success: `200`
- Returns: the list of matching records (in JSON format) or an error message.

```
Command:
curl https://$B2SHARE_HOST/api/records/?q=$QUERY_STRING?access_token=$ACCESS_TOKEN
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

Search drafts

- HTTP method: GET
- URL path: `/api/records/?drafts`
- Required parameters: `access_token`
- Status code on success: `200`
- Returns: the list of matching drafts (in JSON format) or an error message.

```
Command:
curl https://$B2SHARE_HOST/api/records/?drafts&access_token=$ACCESS_TOKEN
```

Get specific record

List the metadata of the record specified by RECORD_ID

- HTTP method: GET
- URL path: `/api/records/RECORD_ID`
- Required parameters: `access_token`
- Status code on success: `200`
- Notes: the access token is only required when a record is not publicly available.

```
Command:
curl
https://$B2SHARE_HOST/api/records/47077e3c4b9f4852a40709e338ad4620?access_token=$ACCES
S_TOKEN
```

Record administration

The following requests concern the creation, update and management of records. Click on a title to show details.

Create draft record

Create a new record, in the draft state.

- HTTP method: POST
- URL path: `/api/records/`

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

- Required parameters: `access_token`
- Payload data: JSON object with basic metadata of the object, at least the required fields of the basic metadata schema of each new record: `titles`, `community` and `open_access`.
- Status code on success: `201`
- Returns: the new draft record metadata including new URL location. Please note that the returned JSON object contains also the URL of the file bucket used for the record. Also note that the URL of the draft record, needed for setting record metadata, will end in '/draft/'
- Notes: you cannot change the community the record resides in after you have created the record.

Example 1

The following example creates an open-access record for a community with identifier `e9b9792e-79fb-4b07-b6b4-b9c2bd06d095` with title 'My dataset record', creators 'John Smith' and 'Jane Smith' and description of type abstract 'A simple description'.

```
Command:
curl -X POST -H "Content-Type:application/json" -d '{"titles":[{"title":"My dataset
record"}], "creators":[{"creator_name": "John Smith"}, {"creator_name": "Jane
Smith"}], "descriptions":[{"description": "A simple description", "description_type":
"Abstract"}], "community":"e9b9792e-79fb-4b07-b6b4-b9c2bd06d095", "open_access":true}'
https://$B2SHARE_HOST/api/records/?access_token=$ACCESS_TOKEN
```

```
Payload:
{
  "titles": [
    {
      "title": "My dataset record"
    }
  ],
  "creators": [
    {
      "creator_name": "John Smith"
    },
    {
      "creator_name": "Jane Smith"
    }
  ],
  "descriptions": [
    {
      "description": "A simple description",
      "description_type": "Abstract"
    }
  ],
  "community": "e9b9792e-79fb-4b07-b6b4-b9c2bd06d095",
  "open_access": true,
  "community_specific": {
    "field_1": "value_1",
    "field_2": "value_2"
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
  }
}
```

```
Returns:
{
  "created": "2016-10-24T12:21:21.697737+00:00",
  "id": "01826ff3e4974415afdb2574a7ea5a91",
  "links": {
    "files": "https://trng-b2share.eudat.eu/api/files/5594a1bf-1484-4a01-b7d3-f1eb3d2e1dc6",
    "publication":
"https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91",
    "self": "https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91/draft",
    "versions":
"https://trng-b2share.eudat.eu/api/records/d855e187e3864ddcaa1b68625866dd78/versions"
  },
  "metadata": {
    "$schema":
"https://trng-b2share.eudat.eu/api/communities/e9b9792e-79fb-4b07-b6b4-b9c2bd06d095/schemas/0#/
draft_json_schema",
    "community": "e9b9792e-79fb-4b07-b6b4-b9c2bd06d095",
    "community_specific": {
      "field_1": "value_1",
      "field_2": "value_2"
    },
    "open_access": true,
    "owners": [
      8
    ],
    "publication_state": "draft",
    "titles": [
      {
        "title": "My dataset record"
      }
    ],
    "creators": [
      {
        "creator_name": "John Smith"
      },
      {
        "creator_name": "Jane Smith"
      }
    ]
  },
  "updated": "2016-10-24T12:21:21.697744+00:00"
}
```

Example 2

The next example creates an open-access record for a community with identifier `94a9567e-2fba-4677-8fde-a8b68bdb63e8` with title 'My community record', creator 'John Smith'. The following community-specific fields are added: 'field_1' and 'field_2'.

For this to work, the schema identifier of the community metadata schema is required. You can get this information from the community metadata using the Get community schema request, although it is a bit hidden. The correct JSONPath for this metadata is

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

/json_schema/allOf/1/properties/community_specific/required, in this example 5108aff5-be5b-4d92-968a-22930ee65e94.

```
Command:
curl -X POST -H "Content-Type:application/json" -d '{"titles":[{"title":"My community
record"}], "creators":[{"creator_name": "John Smith"}],
"community":"94a9567e-2fba-4677-8fde-a8b68bdb63e8", "open_access":true,
"community_specific": {"5108aff5-be5b-4d92-968a-22930ee65e94": {"field_1": "value",
"field_2": "value"}}}' https://$B2SHARE_HOST/api/records/?access_token=$ACCESS_TOKEN
```

```
Payload:
{
  "titles": [
    {
      "title": "My community record"
    }
  ],
  "creators": [
    {
      "creator_name": "John Smith"
    }
  ],
  "community": "94a9567e-2fba-4677-8fde-a8b68bdb63e8",
  "open_access": true,
  "community_specific": {
    "5108aff5-be5b-4d92-968a-22930ee65e94": {
      "field_1": "value",
      "field_2": "value"
    }
  }
}
```

```
Returns:
{
  "created": "2016-10-24T12:21:21.697737+00:00",
  "id": "01826ff3e4974415afdb2574a7ea5a91",
  "links": {
    "files": "https://trng-b2share.eudat.eu/api/files/5594a1bf-1484-4a01-b7d3-f1eb3d2e1dc6",
    "publication":
"https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91",
    "self": "https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91/draft",
    "versions":
"https://trng-b2share.eudat.eu/api/records/d855e187e3864ddcaa1b68625866dd78/versions"
  },
  "metadata": {
    "$schema":
"https://trng-b2share.eudat.eu/api/communities/94a9567e-2fba-4677-8fde-a8b68bdb63e8/schemas/0#/
draft_json_schema",
    "community": "94a9567e-2fba-4677-8fde-a8b68bdb63e8",
    "community_specific": {
      "5108aff5-be5b-4d92-968a-22930ee65e94": {
        "field_1": "value_1",
        "field_2": "value_2"
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
      }
    },
    "open_access": true,
    "owners": [
      8
    ],
    "publication_state": "draft",
    "titles": [
      {
        "title": "My community record"
      }
    ],
    "creators": [
      {
        "creator_name": "John Smith"
      },
      {
        "creator_name": "Jane Smith"
      }
    ]
  },
  "updated": "2016-10-24T12:21:21.697744+00:00"
}
```

Common errors

```
On metadata validation error:
{
  "message": "Validation error.",
  "status": 400
}
```

The supplied metadata is invalid or incorrectly structured. This means that either a specified field does not exist in the metadata schema, or that one of the values for a given field is invalid.

Upload file into draft record

To upload a new file into a draft record object, first you need to identify the file bucket URL. This URL can be found in the information returned when querying a draft record, in the 'links/files' section of the returned data.

- HTTP method: PUT
- URL path: `/api/files/FILE_BUCKET_ID/FILE_NAME`
- Required parameters: `access_token`
- Payload data: the file, sent as direct stream, for curl use the `--data-binary @FILE_NAME` option for this.
- Status code on success: `200`
- Returns: informations about the newly uploaded file

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
Command:
curl -X PUT -H 'Accept:application/json' -H 'Content-Type:application/octet-stream' --
data-binary @$FILE_NAME
https://$B2SHARE_HOST/api/files/$FILE_BUCKET_ID/$FILE_NAME?access_token=$ACCESS_TOKEN
```

Delete file from draft record

Send a DELETE request to the file's URL, which is the same URL used for uploading.

- HTTP method: `DELETE`
- URL path: `/api/files/FILE_BUCKET_ID/FILE_NAME`
- Required parameters: `access_token`
- Status code on success: `204`
- Returns: no content

```
Command:
curl -X DELETE -H 'Accept:application/json'
https://$B2SHARE_HOST/api/files/$FILE_BUCKET_ID/FileToBeRemoved.txt?access_token=$ACCE
SS_TOKEN
```

List files of record

List the files uploaded into a record object. For this request you need the `FILE_BUCKET_ID` which can be found in the metadata of the record.

- HTTP method: `GET`
- URL path: `/api/files/FILE_BUCKET_ID`
- Required parameters: `access_token`
- Status code on success: `200`
- Returns: information about all the files in the record object

```
Command:
curl https://$B2SHARE_HOST/api/files/$FILE_BUCKET_ID?access_token=$ACCESS_TOKEN
```

Update draft record metadata

This action updates the draft record with new information.

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

- HTTP method: PATCH
- URL path: `/api/records/RECORD_ID/draft`
- Required parameters: `access_token`
- Payload data: the metadata for the draft record to be updated, in the JSON Patch format (see http://jsonpatch.com/)
- Status code on success: `200`
- Returns: the updated metadata of the draft record.
- Notes: The JSON Patch format contains one or more JSONPath strings. The root of these paths are the *metadata* object, as this is the only mutable object. For instance, to update the *title* field of the record, use this JSONPath: `/titles/title`

Example 1

The following example adds two values to the metadata field `keywords` of an existing draft record.

```
Command:
curl -X PATCH -H 'Content-Type:application/json-patch+json' -d '[{"op": "add",
"path":"/keywords", "value": ["keyword1", "keyword2"]}]'
https://$B2SHARE_HOST/api/records/$RECORD_ID/draft?access_token=$ACCESS_TOKEN
```

```
Returns:
{
  "created": "2016-10-24T12:21:21.697737+00:00",
  "id": "01826ff3e4974415afdb2574a7ea5a91",
  "links": {
    "files": "https://trng-b2share.eudat.eu/api/files/5594a1bf-1484-4a01-b7d3-f1eb3d2e1dc6",
    "publication":
"https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91",
    "self": "https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91/draft",
    "versions":
"https://trng-b2share.eudat.eu/api/records/d855e187e3864ddcaa1b68625866dd78/versions"
  },
  "metadata": {
    "$schema":
"https://trng-b2share.eudat.eu/api/communities/e9b9792e-79fb-4b07-b6b4-b9c2bd06d095/schemas/0#/
draft_json_schema",
    "community": "e9b9792e-79fb-4b07-b6b4-b9c2bd06d095",
    "community_specific": {},
    "keywords": [
      "keyword1",
      "keyword2"
    ],
    "open_access": true,
    "owners": [
      8
    ],
    "publication_state": "draft",
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
    "titles": [
      {
        "title": "My community title"
      }
    ]
  },
  "updated": "2016-10-24T12:23:59.454951+00:00"
}
```

Example 2

This example replaces the value of the title of a record. This requires a JSONPath `/titles/0/title` as we are updated an existing value of multivalued field.

```
Command:
curl -X PATCH -H 'Content-Type:application/json-patch+json' -d '[{"op": "replace",
"path":"/titles/0/title", "value": ["The new title"]}]'
https://$B2SHARE_HOST/api/records/$RECORD_ID/draft?access_token=$ACCESS_TOKEN
```

```
Returns:
{
  "created": "2016-10-24T12:21:21.697737+00:00",
  "id": "01826ff3e4974415afdb2574a7ea5a91",
  "links": {
    "files": "https://trng-b2share.eudat.eu/api/files/5594a1bf-1484-4a01-b7d3-f1eb3d2e1dc6",
    "publication":
"https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91",
    "self": "https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91/draft",
    "versions":
"https://trng-b2share.eudat.eu/api/records/d855e187e3864ddcaa1b68625866dd78/versions"
  },
  "metadata": {
    "$schema":
"https://trng-b2share.eudat.eu/api/communities/e9b9792e-79fb-4b07-b6b4-b9c2bd06d095/schemas/0#/
draft_json_schema",
    "community": "e9b9792e-79fb-4b07-b6b4-b9c2bd06d095",
    "community_specific": {},
    "open_access": true,
    "owners": [
      8
    ],
    "publication_state": "draft",
    "titles": [
      {
        "title": "The new title"
      }
    ]
  },
  "updated": "2016-10-24T12:23:59.454951+00:00"
}
```

Example 3

The next example updates the community-specific metadata fields `field_1` and `field_2` of an existing draft

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

record of community with identifier `e9b9792e-79fb-4b07-b6b4-b9c2bd06d095`. Note that in order to update a community-specific field, the JSONPath `/community-specific/SCHEMA_ID/FIELD_NAME` is required which contains the schema identifier used by the community.

For this to work, the schema identifier of the community metadata schema is required. You can get this information from the community metadata using the Get community schema request, although it is a bit hidden. The correct JSONPath for this metadata is `/json_schema/allOf/1/properties/community_specific/required`.

```
Command:
curl -X POST -H "Content-Type:application/json-patch+json" -d '[{"op": "add", "path":
"/community_specific/$SCHEMA_ID/field_1", "value": "value_1"}, {"op": "add", "path":
"/community_specific/$SCHEMA_ID/field_2", "value": "value_2"}]'
https://$B2SHARE_HOST/api/records/$RECORD_ID/draft?access_token=$ACCESS_TOKEN
```

```
Returns:
{
  "created": "2016-10-24T12:21:21.697737+00:00",
  "id": "01826ff3e4974415afdb2574a7ea5a91",
  "links": {
    "files": "https://trng-b2share.eudat.eu/api/files/5594a1bf-1484-4a01-b7d3-f1eb3d2e1dc6",
    "publication":
"https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91",
    "self": "https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91/draft",
    "versions":
"https://trng-b2share.eudat.eu/api/records/d855e187e3864ddcaa1b68625866dd78/versions"
  },
  "metadata": {
    "$schema":
"https://trng-b2share.eudat.eu/api/communities/e9b9792e-79fb-4b07-b6b4-b9c2bd06d095/schemas/0#/
draft_json_schema",
    "community": "e9b9792e-79fb-4b07-b6b4-b9c2bd06d095",
    "community_specific": {
      "field_1": "value_1",
      "field_2": "value_2"
    },
    "open_access": true,
    "owners": [
      8
    ],
    "publication_state": "draft",
    "titles": [
      {
        "title": "My dataset record"
      }
    ]
  },
  "updated": "2016-10-24T12:21:21.697744+00:00"
}
```

Common errors

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
On JSON Patch operation error:
{
  "message": "Invalid Operation.",
  "errors": [
    {
      "message": "Invalid JSON Pointer"
    }
  ],
  "status": 400
}
```

One of the JSON Patch operations is invalid.

```
On JSON Patch content type error:
{
  "message": "Invalid 'Content-Type' header. Expected one of: application/json-patch+json",
  "status": 415
}
```

The supplied content type header value is invalid.

```
On metadata validation error:
{
  "message": "Validation error.",
  "errors": [
    {
      "message": "{'title': 'Some title'} is not of type 'array'",
      "field": "titles"
    }
  ],
  "status": 400
}
```

The supplied value for the metadata field is invalid.

Submit draft record for publication

This action marks the draft record as complete and submits it for publication. Currently B2SHARE automatically publishes all the submitted drafts. Please be advised that publishing the draft **will make its files immutable**.

A draft record is submitted for publication if a special metadata field, called 'publication_state' is set to 'submitted'. This field can be set using the metadata update request described above.

Depending on the community specification, other fields could be required in order to successfully publish a record. In case one of the required fields is missing the request fails and an error message is returned with further details.

- HTTP method: GET
- URL path: `/api/records/$RECORD_ID/draft`
- Required parameters: `access_token`
- Payload data: JSON Patch operation that alters the `publication_state` metadata field of the record

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

metadata, see example below.

- Status code on success: `200`
- Notes: this request is essentially a metadata update request as described above.

```
Command:
curl -X PATCH -H 'Content-Type:application/json-patch+json' -d '[{"op": "add",
"path":"/publication_state", "value": "submitted"}]'
https://$B2SHARE_HOST/api/records/$RECORD_ID/draft?access_token=$ACCESS_TOKEN
```

```
Returns:
{
  "created": "2016-10-24T12:21:21.697737+00:00",
  "id": "01826ff3e4974415afdb2574a7ea5a91",
  "links": {
    "files": "https://trng-b2share.eudat.eu/api/files/5594a1bf-1484-4a01-b7d3-f1eb3d2e1dc6",
    "publication":
"https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91",
    "versions":
"https://trng-b2share.eudat.eu/api/records/c1d28e53db104cb286425902af134579/versions",
    "self": "https://trng-b2share.eudat.eu/api/records/01826ff3e4974415afdb2574a7ea5a91/draft"
  },
  "metadata": {
    "$schema":
"https://trng-b2share.eudat.eu/api/communities/e9b9792e-79fb-4b07-b6b4-b9c2bd06d095/schemas/0#/
draft_json_schema",
    "DOI": "10.5072/b2share.cdb15c27-326e-4e95-b812-6b1c6b54c299",
    "community": "e9b9792e-79fb-4b07-b6b4-b9c2bd06d095",
    "community_specific": {},
    "keywords": [
      "keyword1",
      "keyword2"
    ],
    "ePIC_PID": "http://hdl.handle.net/11304/2c473f04-d997-47d9-9bdb-d3d71800f870",
    "open_access": true,
    "owners": [
      8
    ],
    "publication_state": "published",
    "titles": [
      {
        "title": "TestRest"
      }
    ]
  },
  "updated": "2016-10-24T12:26:51.538025+00:00"
}
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

Update published record metadata

This request updates the metadata of an already published record without creating a new version.

- HTTP method: `PATCH`
- URL path: `/api/records/RECORD_ID/`
- Required parameters: `access_token`
- Payload data: the metadata for the published record object to be updated, in the JSON Patch format (see http://jsonpatch.com/)
- Status code on success: `200`
- Notes: The JSON Patch format contains one or more JSONPath strings. The root of these paths are the *metadata* object, as this is the only mutable object. For instance, to update the *title* field of the record, use this JSONPath: `/titles/title`

See the Update draft record metadata request for examples.

Record versioning

The following requests concern the versioning of published records. Click on a title to show details.

Create new version of published record

Create a new version of an existing published record into a new draft.

- HTTP method: `POST`
- URL path: `/api/records/RECORD_ID/draft`
- Required parameters: `access_token`
- Status code on success: `201`
- Returns: the new draft record metadata including new URL location. The metadata will be exactly the same as the original record with the exception of the links and persistent identifiers. Since the new record is in draft state, you can freely alter it, including the files.
- Notes: the output of the request is the same as the output of the Create new record request. You cannot create a new version of a draft record itself.

```
Command:
curl -X POST -H "Content-Type:application/json"
https://$B2SHARE_HOST/api/records/$RECORD_ID/draft?access_token=$ACCESS_TOKEN
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

Get all record versions

Get all versions of a specific record by using the record head identifier.

- HTTP method: GET
- URL path: `/api/records/RECORD_HEAD_ID/versions`
- Required parameters: `access_token`
- Status code on success: `200`
- Returns: a JSON structure containing a list of all record versions with identifier, version number and URL.
- Notes: the record head identifier is not the same as the record identifier. Use the metadata of the record to find the record head identifier, located in the JSONPath `/links/versions`.

  If a record is not versioned, the result will be empty.

```
Command:
curl -H "Content-Type:application/json"
https://$B2SHARE_HOST/api/records/$RECORD_HEAD_ID/versions?access_token=$ACCESS_TOKEN
```

```
Returns:
{
  "versions": [
    {
      "created": "Wed, 05 Jul 2017 09:40:14 GMT",
      "id": "a766efd2e5d543968fff9dd7bf3783c5",
      "updated": "Tue, 19 Dec 2017 12:15:06 GMT",
      "url": "https://trng-b2share.eudat.eu/api/records/a766efd2e5d543968fff9dd7bf3783c5",
      "version": 1
    },
    {
      "created": "Fri, 12 Jan 2018 16:43:33 GMT",
      "id": "2ff3f5815db3494a840e6b3f1e6a6542",
      "updated": "Fri, 12 Jan 2018 16:43:33 GMT",
      "url": "https://trng-b2share.eudat.eu/api/records/2ff3f5815db3494a840e6b3f1e6a6542",
      "version": 2
    }
  ]
}
```

```
On not found error:
{
  "message": "PID is not registered.",
  "status": 404
}
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

Other requests

The following requests are the remaining requests possible in B2SHARE. Click on a title to show details.

Report a record as an abuse record

If there is anything wrong with the record users can report it as an abuse record. An email will be send to the related admin and it will be followed up. There are 4 different reasons listed on the report abuse form and the reporter should choose one of:

1. Abuse or Inappropriate content
2. Copyrighted material
3. Not research data
4. Illegal content

The reporter can also send a message to explain more about the problem. It is possible for an anonymous user to send the report and authentication is not required.

Report an abuse record.

- HTTP method: `POST`
- URL path: `/api/records/$RECORD_ID/abuse`
- Required parameters: `access_token`
- Payload data: JSON object with information about reporter, the reason indicated by booleans and a message.
- Status code on success: `200`
- Returns: a message that an email was sent and the record is reported

```
Command:
curl -X POST -H 'Content-Type:application/json' -d '{"noresearch":true,
"abusecontent":false, "copyright":false, "illegalcontent":false,"message":"It is not
research data...", "name":"John Smith", "affiliation":"Example University",
"email":"j.smith@example.com", "address":"Example street", "city":"Example City",
"country":"Example country", "zipcode":"12345", "phone":"7364017452"}'
https://$B2SHARE_HOST/api/records/$RECORD_ID/abuse?access_token=$ACCESS_TOKEN
```

```
Payload:
{
  "noresearch": true,
  "abusecontent": false,
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
  "copyright": false,
  "illegalcontent": false,
  "message": "It is not research data...",
  "name": "John Smith",
  "affiliation": "Example University",
  "email": "j.smith@example.com",
  "address": "Example street",
  "city": "Example City",
  "country": "Example country",
  "zipcode": "12345",
  "phone": "7364017452"
}
```

```
Returns:
{
  "message": "The record is reported.",
  "status": 200
}
```

Send record access request

For the records with restricted access to data, a user (either authenticated or anonymous) can send a request to the record owner and ask for it. Send request to access closed data.

- HTTP method: POST
- URL path: `/api/records/$RECORD_ID/accessrequests`
- Required parameters: `access_token`
- Payload data: JSON object with information about who is sending the request
- Status code on success: `200`
- Returns: a message that an email was sent

```
Command:
curl -X POST -H 'Content-Type:application/json' -d '{"message":"Explain the
request...", "name":"John Smith", "affiliation":"Example University",
"email":"j.smith@example.com", "address":"Example street", "city":"Example City",
"country":"Example country", "zipcode":"12345", "phone":"7364017452"}'
https://$B2SHARE_HOST/api/records/$RECORD_ID/accessrequests?access_token=$ACCESS_TOKEN
```

```
Payload:
{
  "message": "Explain the request...",
  "name": "John Smith",
  "affiliation": "Example University",
  "email": "j.smith@example.com",
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
  "address": "Example street",
  "city": "Example City",
  "country": "Example country",
  "zipcode": "12345",
  "phone": "7364017452"
}
```

```
Returns:
{
  "message": "An email was sent to the record owner.",
  "status": 200
}
```

Get record statistics

Returns statistics of a record, indicated by statistic type.

Supported statistics:

- `bucket-file-download-total` - total downloads per file in a bucket

More statistics might be added in future releases of B2SHARE.

- HTTP method: POST
- URL path: `/api/stats`
- Required parameters: `access_token`
- Payload data: JSON structure containing the statistic and required elements for that statistic, see below
- Status code on success: `200`
- Returns: JSON structure containing requested statistics

```
Command:
curl -X POST -H 'Content-Type:application/json' -d '{"fileDownloads": {"params":
{"bucket_id": "b0377611-d5a4-4683-9781-b83edcb86324"}, "stat": "bucket-file-download-
total"}}' https://$B2SHARE_HOST/api/stats?access_token=$ACCESS_TOKEN
```

```
Payload:
{
  "fileDownloads": {
    "params": {
      "bucket_id": "b0377611-d5a4-4683-9781-b83edcb86324"
    },
    "stat": "bucket-file-download-total"
  }
```

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

```
}
```

```
Returns:
{
  "fileDownloads": {
    "key_type": "terms",
    "field": "file_key",
    "buckets": [
      {
        "value": 1,
        "key": "file.dat"
      }
    ],
    "type": "bucket"
  }
}
```

Delete draft record

Delete a draft record.

- HTTP method: DELETE
- URL path: `/api/records/RECORD_ID/draft`
- Required parameters: `access_token`
- Status code on success: 204
- Returns: no contents.
- Notes: you can only delete draft records that you own, not published records.

```
Command:
curl -X DELETE
https://$B2SHARE_HOST/api/records/$RECORD_ID/draft?access_token=$ACCESS_TOKEN
```

Delete published record

Delete a published record.

- HTTP method: DELETE
- URL path: `/api/records/RECORD_ID`
- Required parameters: `access_token`
- Status code on success: 204

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.

- Returns: no contents.
- Notes: only a site administrator can delete a published record.

```
Command:
curl -X DELETE
https://$B2SHARE_HOST/api/records/$RECORD_ID/?access_token=$ACCESS_TOKEN
```

Support

Hands-on material is available from the EUDAT Training Repository.

Our B2SHARE presentations offer training material for the service.

Support for B2SHARE and its REST API is available via the EUDAT ticketing system through the webform.

If you have comments on this page, please submit them though the EUDAT ticketing system.

Document Data

Version: 1.4.1

Authors:

Hans van Piggelen, hans.vanpiggelen@surfsara.nl

Emanuel Dima, emanuel.dima@uni-tuebingen.de

Editors:

Kostas Kavoussanakis, k.kavoussanakis@epcc.ed.ac.uk

Read more

EUDAT receives funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 654065.