1. Batch upload of data

In case you need to upload large amount of data either historical or non-time critical data it is beneficial to use the batch upload methods. There is a batch upload procedure which either can be called as HTTP command using “curl” – this is intended for manual upload, or it can be imbedded in a Python script.

2. Batch import HTTP API

To import data from csv-files you need to:

1. format the .csv files according to the instructions below.
2. create the upload in the database
3. get the upload url
4. upload the file to s3
5. validate the file
6. ingest the file

There is a python helper class that makes this easier, see the next section.

2.1 Authorization and headers

In all the http requests to http://admin.energydata.dk/api/v1/import below you need to add a authorization header and a accept header. The authorization must be a bearer token, where the token is an batch import data token, that the user can create in the api token section of the energydata.dk ui.

Here is an example of the headers, with a fictional token:

```
Authorization: Bearer bzkMFuZgTzsuOldud98fJ5RkRHBI4Uoprdub1R"
Accept: application/json
```

2.2 Format file

The file must be formatted as CSV according to RFC4180 RFC4180. Additionally, if the file contains non-ascii characters, the file should be encoded as UTF-8.

The first row of the CSV file must contain headers. The header of a given column identifies the datastream to which the column’s data must be imported. The header can either be the

DEN EUROPÆISKE UNION
Den Europæiske Socialfond
DEN EUROPÆISKE UNION
Den Europæiske Fond for Regionaludvikling
Finansieret som et led i EU's reaktion på COVID-19-pandemien
Vi investerer i din fremtid

topic associated with the given datastream, e.g. my/topic or the datastream ID, e.g. 19323. The first column of the first row is ignored.

Every row after the header contains a timestamp in the first column. The rest of the columns contain the values applicable for the given datastream at the given timestamp.

The timestamps must be formatted as YYYY-MM-DD[T]HH:mm:ss.SSS[Z], e.g. 2021-05-01T10:12:44.432Z for May 1st 2023, 10:12:44.432 UTC. The timestamp must be in UTC timezone.

The type of the data in a column must match the datatype of the datastream to which the column belongs. If a datastream is an integer, any values for that datastream must be integers as well.

If a field is left empty, the handling depends on the datatype of the corresponding datastream. If the datastream is of type string, an empty string will be imported. If the datastream is of type integer or double, no value will be imported.

Below is an example file. Note that the first datastream (117217) is of type string, the second datastream (my/topic) is of type double and the last datastream (119221) is of type integer.

```
;117217;my/topic;119221
2021-03-10T20:24:30.139Z;a_string;23.4121;-10
2021-03-10T20:24:31.144Z;"another string";999888777.121;0
2021-03-10T20:24:32.161Z;a third string;-1.33e-16;45
2021-03-10T20:24:33.186Z;;54.1;11
2021-03-10T20:24:34.201Z;a-fourth-string;;45
```

Note the empty fields in row 5 (datastream 117217) and row 6 (my/topic). For datastream 117217, an empty string will be imported with timestamp 2021-03-10T20:24:33.186Z. For datastream my/topic, no value will be imported with timestamp 2021-03-10T20:24:34.201Z.

2.3 Create the upload

To create an import, post an import name to https://admin.energydata.dk/api/v1/import, with the import as an url parameter.

```
curl -H "Authorization: Bearer
bzkMFuZgTzsuOldud98fJ5RkRHBI4Uoprdub1R" -H "Accept:
application/json" -X POST
https://admin.energydata.dk/api/v1/import -G --data-urlencode
"importname=batch-import-example"

# Response example
```

DEN EUROPÆISKE UNION
Den Europæiske Socialfond
DEN EUROPÆISKE UNION
Den Europæiske Fond for Regionaludvikling
Finansieret som et led i EU's reaktion på COVID-19-pandemien
Vi investerer i din fremtid

```
# {"user_id":11,"status":"stored","name":"batch-import-
example2","updated_at":"2023-03-
27T08:20:36.000000Z","created_at":"2023-03-
27T08:20:36.000000Z","id":1340}
```

2.3.1 Show and delete

By calling `get` or `delete` on the endpoint `https://admin.energydata.dk/api/v1/import/1340` you can see or delete an import with the id `1340`

2.4 Get the upload url

Get the upload url with the id returned in the creation above:

```
curl -H "Authorization: Bearer
zGefiozgtCOfW70jnydNkTF35WOctQUnEMGAYKpi" -H "Accept:
application/json"
https://admin.energydata.dk/api/v1/import/1340/upload_url

# Response example
# {
#  "upload_url":
"https://s3.energydata.dk/import/inbox/example-user/b23sdfb23-
b163-4dfc-84c8-87ded19fdse1a.csv?X-Amz-Content-
Sha256=UNSIGNED-PAYLOAD&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-
Amz-Credential=edk-s3-
root%2F20230327%2Fdefault%2Fs3%2Faws4_request&X-Amz-
Date=20230327T083455Z&X-Amz-SignedHeaders=host&X-Amz-
Expires=7200&X-Amz-
Signature=16697645e0f009c791e62f237dfd80e74d9ddfda83feb8801sd5
af186c81e0c0b"
}
```

2.5 Upload the file

Upload the file using curl, the python helper script in or see the minio documentation for alternatives.

In the following curl example the file-path used is `batch-import-example.csv`

```
curl -X PUT -H "Content-Type: application/octet-stream" --
data-binary "@batch-import-example.csv"
"https://s3.energydata.dk/import/inbox/example-user/b23sdfb23-
```

DEN EUROPÆISKE UNION
Den Europæiske Socialfond
DEN EUROPÆISKE UNION
Den Europæiske Fond for Regionaludvikling
Finansieret som et led i EU's reaktion på COVID-19-pandemien
Vi investerer i din fremtid

```
b163-4dfc-84c8-87ded19fdse1a.csv?X-Amz-Content-
Sha256=UNSIGNED-PAYLOAD&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-
Amz-Credential=edk-s3-
root%2F20230327%2Fdefault%2Fs3%2Faws4_request&X-Amz-
Date=20230327T083455Z&X-Amz-SignedHeaders=host&X-Amz-
Expires=7200&X-Amz-
Signature=16697645e0f009c791e62f237dfd80e74d9ddfda83feb8801sd5
af186c81e0c0b" > /dev/null
```

2.6 Validate the file

To validate the file format and data, send a PUT to `https://admin.energydata.dk/api/v1/import/1340/validate`, in the following example the import id 1340 is used

```
curl -H "Authorization: Bearer
bzkMFuZgTzsuOldud98fJ5RkRHBI4Uoprdub1R" -H "Accept:
application/json" -X PUT
https://admin.energydata.dk/api/v1/import/1340/validate"
```

A correct request will just accept the task. To see the status of the validation you need to post a GET request as specified in section 2.1. On the returned import will be a status and any errors.

2.7 Ingest the file

To ingest the file format and data, send a PUT to `https://admin.energydata.dk/api/v1/import/1340/ingest`, in the following example the import id 1340 is used

```
curl -H "Authorization: Bearer
bzkMFuZgTzsuOldud98fJ5RkRHBI4Uoprdub1R" -H "Accept:
application/json" -X PUT
https://admin.energydata.dk/api/v1/import/1340/ingest"
```

A correct request will just accept the task. To see the status of the ingestion you need to post a GET request as specified in section 2.1. On the returned import will be a status and any errors.

DEN EUROPÆISKE UNION
Den Europæiske Socialfond
DEN EUROPÆISKE UNION
Den Europæiske Fond for Regionaludvikling
Finansieret som et led i EU's reaktion på COVID-19-pandemien
Vi investerer i din fremtid

3. Batch import python API

This is the documentation for a class made to make importing into Energydata.dk easier from Python. The code can be found in this repository:
https://git.elektro.dtu.dk/energydatadk/batch-import-python-api

3.1 EnergyDataImport

```
class EnergyDataImport()
```

A simple class to make batch importing to Energydata.dk easier from Python

This class handles building a csv file in the correct format to be used for importing to Energydata.dk via the batch API. The batch API itself is documented seperatley here: https://docs.energydata.dk/http-api/importing_data.html This class can handle everything from creating the propper csv to uploading, validating and ingesting the file. You can check the 'example_import.py' script for an example of how to use this class.

3.1.1 __init__

```
__init__(upload_filename: str, properties: List,
sftp_username: str, sftp_password: str, energydata_api_token:
str, overwrite: bool = False, tmp_dir: str =
'/tmp/energydata_batch_uploud', autoclean_tmp_files: bool =
True)
```

Constructor for class. Create an instance of EnergydataImport for each batch of data you want to upload. An instance can not be reused between multiple imports.

Arguments:

- `upload_filename` *str* - A filename for the import file generated behind the scenes. A local file in tmp_dir will be created with this name, and the same filename is used when uploading to the SFTP server.
- `properties` *List* - List of properties to which data will be added. Can be either property id's or topics.
- `sftp_username` *str* - Username for the sftpserver on 'api.energydata.dk' from where batch imports run. Ask an admin for access credentials.
- `sftp_password` *str* - Password for the sftpserver on 'api.energydata.dk' from where batch imports run. Ask an admin for access credentials.
- `energydata_api_token` *str* - Api token as found in: https://portal.energydata.dk/user#accesstokens

DEN EUROPÆISKE UNION
Den Europæiske Socialfond
DEN EUROPÆISKE UNION
Den Europæiske Fond for Regionaludvikling
Finansieret som et led i EU's reaktion på COVID-19-pandemien
Vi investerer i din fremtid

- `overwrite` *bool, optional* - If a local file with filename {tmp_dir}/{upload_filename} or a file with on the SFTP upload folder with {upload_filename} is found, this setting will control whatever we should overwrite the file or not. Defaults to False.
- `tmp_dir` *str, optional* - A directory for temporarily storing data file before they are uploaded to SFTP. Defaults to '/tmp/energydata_batch_uploud'.
- `autoclean_tmp_files` *bool, optional* - If this parameter is set to False the files generated in tmp_dir will not be removed once import context is closed. Defaults to True.

3.1.2 add_values

```
add_values(time: datetime, values: List)
```

Call while in open state to keep adding values to be batch imported.

Arguments:

- `time` *datetime* - Timestamp for values to be added
- `values` *List* - A list of the values to be added

Raises:

- `Exception` - An exception is raised if the time argument is a datetime object without any timezone information
- `Exception` - The length of the values list must be equal to the number of properties passed in the constructor, otherwise an exception is raised.

3.1.3 Upload

```
upload(print_progress=True)
```

When all values have been added using add_values method, this method can be called to uploud the generated csv file to the SFTP server, and create a import job with the energydata.dk API. Once called, state of the job will transition from open to uploading, uploaded and finally to stored once the job is created.

Arguments:

- `print_progress` *bool, optional* - Whatever the upload progress should be continiously printed. Defaults to True.

3.1.4 Validate

```
validate(block=True, print_progress=True)
```

When in stored state, this method can be called to trigger validation of the data to be imported. This will happen by calling the corresponding validate endpoint on energydata.dk. State will transition from stored to validating and finally to ready.

Arguments:

- `block` *bool, optional* - Whatever this method should block while validation is in progress. Defaults to True.
- `print_progress` *bool, optional* - Whatever the upload progress should be continiously printed. Defaults to True.

3.1.5 Ingest

```
ingest(block=True, print_progress=True)
```

When in ready state, this method can be called to trigger ingestion of the data to be imported. This will happen by calling the corresponding ingest endpoint on energydata.dk. State will transition from ready to ingesting and finally to done.

Arguments:

- `block` *bool, optional* - Whatever this method should block while ingestion is in progress. Defaults to True.
- `print_progress` *bool, optional* - Whatever the upload progress should be continiously printed. Defaults to True.

3.1.6 refresh_status

```
refresh_status()
```

This method can be used to manually refresh and sync the internal state of the import job with the state in energydata.dk. If for instance validate is called with block=False, this method must be used to refresh the state before ingest can be called, once validation has completed on the server.

Returns:

- `[type]` - Return the current status as in the form of an enum of type EnergyDataImport.Status.

DEN EUROPÆISKE UNION
Den Europæiske Socialfond
DEN EUROPÆISKE UNION
Den Europæiske Fond for Regionaludvikling
Finansieret som et led i EU's reaktion på COVID-19-pandemien
Vi investerer i din fremtid