



Guidance

Upload data to the OpenEnergy Platform and initiate review process

REST-API via python

v.0.3

This document describes how to upload data to the [OpenEnergy Platform \(OEP\)](#) using [Python](#) and the [REST-API](#) as well as a description on how to initiate the review process of that data.

Several steps are needed to contribute actively to the platform. Some of them you only need to do once to get started. The steps described in this document are the following:

- **(once)** [Register at the OpenEnergy Platform](#)
- **(once)** [Register at Github and get invited to the OpenEnergy Platform Group](#)
- [Create data tables\(s\) and upload data table\(s\)](#)
- [Create metadata](#) and [upload metadata to Github](#)
- [Initiate review](#)
- [Create factsheets](#)

Structure

1.	Things you only need to do once	3
1.1.	Register at the OpenEnergy Platform	3
1.2.	Register at github and get invited to OpenEnergyPlatform Group	4
2.	Things you will need to do any time you contribute new data and metadata	5
2.1.	Create and upload data table(s)	5
2.1.1.	Create a new table	5
2.1.2.	Upload data	7
2.1.3.	Starting over: Deleting your table	7
2.1.4.	Complete code for one example	8
2.2.	Create metadata	9
2.3.	Initiate review	11
3.	Complete your contribution	12
3.1.	Create Factsheets	12

1. Things you only need to do once

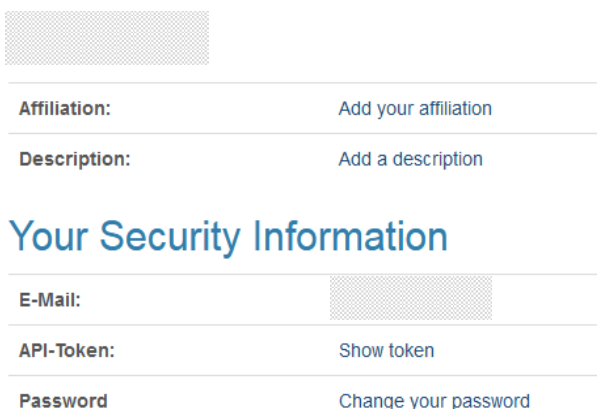
1.1. Register at the OpenEnergy Platform

- Registration
 - Click [Login](#) button on Homepage and choose the option **Create a new account**.
 - OR: go directly to <https://openenergy-platform.org/user/register:>



The screenshot shows the 'Create new user' registration form on the OpenEnergy Platform. It includes input fields for Name*, Affiliation, Email address*, Password*, and Password confirmation*. A green 'Submit' button is highlighted with a red box and a red arrow pointing to it.

- Fill out the required fields and click on **Submit**.
- You will receive an activation email, sent to your email address.
- Follow the instructions received in the email.
- After successfully activating and logging into your account, you will see your information as in the screenshot below:



The screenshot shows the user profile page after registration. It includes a header section with a blurred profile picture. Below it are two rows: 'Affiliation:' with a link 'Add your affiliation' and 'Description:' with a link 'Add a description'. The main section is titled 'Your Security Information' and contains three rows: 'E-Mail:' with a blurred email address, 'API-Token:' with a link 'Show token', and 'Password' with a link 'Change your password'.

- API-Token
 - The API token is necessary to make authenticated requests to the API.
 - After you logged into the OEP, click the member icon at the upper right corner:



- You can find your API-Token under “Your Security Information” by clicking on **Show token**:

 Affiliation: [Add your affiliation](#)

 Description: [Add a description](#)

Your Security Information

 E-Mail:

 API-Token: [Show token](#)

 Password [Change your password](#)

1.2. Register at github and get invited to OpenEnergyPlatform Group

- If you do not yet have a github account, you need to register: <https://github.com/join>.
- You will then need to become a Member of the group OpenEnergyPlatform: <https://github.com/OpenEnergyPlatform> by getting invited to the group. This works as follows:
 - Get in touch via the contact form on the OpenEnergyPlatform: <https://openenergy-platform.org/contact/>.
 - Fill in the form with your request to be added to the OpenEnergyPlatform group on github. An example text is provided in the screenshot below.

OEP - Contact

General Matter:

Name:

Subject:

Email:

Content:

- A response will not be immediate. Please wait until you receive an invitation to the github group. Only then you will be able to [upload metadata](#) and [initiate data review](#).

2. Things you will need to do any time you contribute new data and metadata

2.1. Create and upload data table(s)

Prerequisite for this step is [that you are a registered user on openenergy-platform.org](#).

Please note: code snippets below are shown to highlight specific instances of a sequence of code. You find a complete sample code that you can copy&paste and try under 2.1.4 Complete code for one example.

- The REST-API can be used with any language than can make HTTP(s) requests.
- Most requests require you to add an authorization header:
Authorization: "Token API_TOKEN", where you substitute API_TOKEN with your token [as described above](#).
- All requests (and most responses) will use json data as payload¹
- In the examples below, we use python and the [requests package](#). All requests will use a requests session with the authorization header.

```
import requests
API_URL = 'https://openenergy-platform.org/api/v0'
session = requests.Session()
session.headers = {'Authorization': 'Token %s' % API_TOKEN}
```

- The requests in the following sections use roughly the same pattern:
 - Prepare your request payload as a json object
 - Prepare your request url
 - Send your request using the correct [verb](#) (get, post, put, delete)
 - Check if the request was successful

2.1.1. Create a new table

- You will need to create the tables at first in the *model_draft* schema. This schema is used for data that is still preliminary and subject to changes. After a successful [review](#) later, the table will be moved to the final target schema.
- You need to specify the name of the new table (TABLE_NAME), which should be a valid post-gresql table name, without spaces, ideally only containing lower case letters, numbers and underscores.

¹ Payload: the actual data content of the request

- You also need to specify names and data types of your columns, which also must be [valid postgres data types](#).

```
# prepare request payload
data = {'query': {
    'columns': [
        {
            'name': 'id',
            'data_type': 'bigserial'
        },
        # add more columns here
    ],
    'constraints': [
        {'constraint_type': 'PRIMARY KEY', 'constraint_parameter': 'id'}
    ]
}}

# prepare api url
url = API_URL + '/schema/model_draft/tables/' + TABLE_NAME

# make request and check using PUT
res = session.put(url, json=data)
res.raise_for_status() # check: throws exception if not successful
```

2.1.2. Upload data

- To upload data, you must first load it into a json structure as a [list](#) representing data rows, each of which is a [dictionary](#) mapping column names to values.
- In the example, we will use [pandas](#) to read data from an Excel workbook (WORKBOOK, WORKSHEET) into a [data frame](#) which we will then convert into a json object. Please note that this step will most likely require some modification to accommodate the specifics of your input data.
- In addition to that, at the end, you need to load your data into the specified json structure.
- After that, the data can be uploaded making a request to the API:

```
# load data into dataframe, convert into json
df = pd.read_excel(WORKBOOK, WORKSHEET)
records = df.to_json(orient='records')
records = json.loads(records)

# prepare request payload
data = {'query': records}

# prepare api url
url = API_URL + '/schema/model_draft/tables/' + TABLE_NAME + '/rows/new'

# make request
res = session.post(url, json=data)
res.raise_for_status() # check
```

- You can repeat this if you want to upload your data in multiple batches.

2.1.3. Starting over: Deleting your table

- While the table is still in the model draft, you can always delete the table and start over:

```
# prepare api url
url = API_URL + '/schema/model_draft/tables/' + TABLE_NAME

# make request
res = session.delete(url)
res.raise_for_status() # check
```

2.1.4. Complete code for one example

- This is an example code that you can copy and paste to execute the workflow we have described above. The workflow described does not take into account any specific settings you will need to make in your personal work environment. It can, however, simply be executed in the python prompt.
- Let's say you have an Excel workbook named "CountryValues.xlsx", with a worksheet named "data"² as depicted below:

	A	B	C	D
1	id	country	year	value
2	1	DE	2010	13.0
3	2	DE	2011	14.0
4	3	DE	2012	12.0
5	4	FR	2010	15.0
6	5	FR	2011	12.0
7				
8				
9				
10				
11				
12				

- Your API-Token is "xxxxxxxxxxxx", and your desired table name shall be "my_example_table". Please note that, while it is highly unlikely, there may already exist a table with this particular name created by another person. If this is the case, the example below will not work. Then, please choose another name for your table and substitute every instance with 'my_example_table' below with the name you have chosen.
- Save the example code below as "example.py" in the same folder as the Excel workbook, then [open a command line](#), [navigate to the folder](#), and type "python3 example.py"

```
import json
import requests
import pandas as pd

API_TOKEN = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
TABLE_NAME = 'my_example_table'

API_URL = 'https://openenergy-platform.org/api/v0'
session = requests.Session()
session.headers = {'Authorization': 'Token %s' % API_TOKEN}

# -----
# delete table (in case it already exists)
# -----

# prepare api url
url = API_URL + '/schema/model_draft/tables/' + TABLE_NAME
# make request
res = session.delete(url)

# -----
# create table
# -----

# prepare request payload
data = {'query': {
```

² To be on the safe side we suggest you use other values


```

    'columns': [
        {'name': 'id', 'data_type': 'bigserial'},
        {'name': 'country', 'data_type': 'varchar'},
        {'name': 'year', 'data_type': 'int'},
        {'name': 'value', 'data_type': 'float'}
    ],
    'constraints': [
        {'constraint_type': 'PRIMARY KEY', 'constraint_parameter': 'id'}
    ]
}
}

# prepare api url
url = API_URL + '/schema/model_draft/tables/' + TABLE_NAME + '/'
# make request and check using PUT
res = session.put(url, json=data)
res.raise_for_status() # check: throws exception if not successful

# -----
# read data
# -----

# load data into dataframe, convert into json
df = pd.read_excel('CountryValues.xlsx', 'data')
records = df.to_json(orient='records')
records = json.loads(records)

# -----
# upload data
# -----

# prepare request payload
# 'records' is a list of dictionaries (field name: value)
data = {'query': records}
# prepare api url
url = API_URL + '/schema/model_draft/tables/' + TABLE_NAME + '/rows/new'
# make request
res = session.post(url, json=data)
res.raise_for_status() # check

# -----
# read data (to check if everything was ok)
# or search for your table here:
# https://openenergy-platform.org/dataedit/view/model_draft/tables/my_example_table
# -----

# prepare api url
url = API_URL + '/schema/model_draft/tables/' + TABLE_NAME + '/rows'
# make request
res = session.get(url)
print(res.json())

# -----
# delete table (because it was just a test)
# -----

# prepare api url
url = API_URL + '/schema/model_draft/tables/' + TABLE_NAME
# make request
res = session.delete(url)
res.raise_for_status() # check

```

2.2. Create metadata

- You must save your meta data in a json file³ following the examples from [github](#).

³ A plain text file with the suffix .json, following the JSON specifications. Here is one of many tutorials for that: <https://www.digitalocean.com/community/tutorials/an-introduction-to-json>

- A documentation of the OEP metadata standard can be found [here](#). This can guide you what information to put where and in which format.
- Although optional, we recommend that you fill out as many fields as possible to make the data meaningful to other users.
- Then upload metadata to github as described below.

Prerequisite for this step is that you have a [github account](#) and you are member of the [OpenEnergy Platform group on github](#).


- Once your data table(s) have been uploaded to *model_draft* schema on the OpenEnergy Platform, you will need to submit this information and the metadata to github into the *data-preprocessing* repository:
<https://github.com/OpenEnergyPlatform/data-preprocessing/tree/master/data-review>
- In the [data-review folder](#) you need to **create a new branch** with a suitable name for your data:
 - Click [upload files](#)
 - Select your meta data json file(s)
 - Describe the files and context, to make it easier for the reviewer.
 - At the end of the form create the new branch `/review/###NAME###` and give it a suitable name (e.g. name of the scenario)

[data-preprocessing](#) / [data-review](#)



Drag files here to add them to your repository

Or [choose your files](#)



Commit changes

Add files via upload

Add an optional extended description...

☐ Commit directly to the `master` branch.
 ☒ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

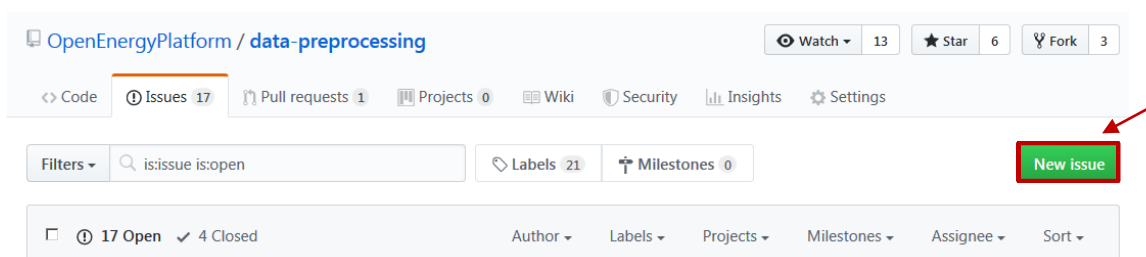
Will be created as `review/###NAME##`

Propose changes
Cancel

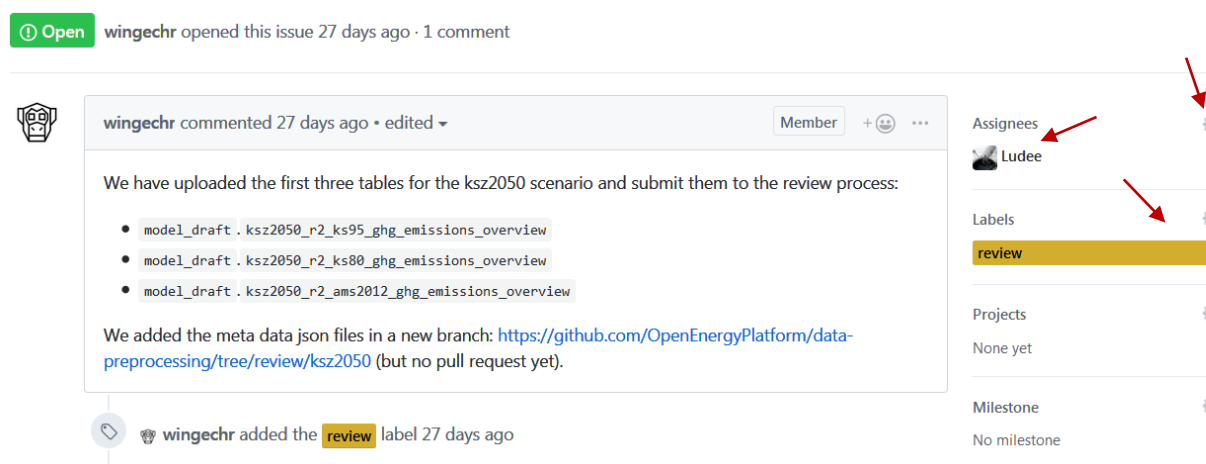
2.3. Initiate review

Prerequisite for this step is that you have a [github account](#) and you are member of the [OpenEnergy Platform group](#).

- Create a new issue in the [data-preprocessing](#) repository to describe your recent upload and the context:



- See an example issue here: <https://github.com/OpenEnergyPlatform/data-preprocessing/issues/27>
- Assign the user *Ludee* to this issue. This user currently is the one who reviews all data.
- Select the label *review*.



- Now you have completed the whole process from your side. All you need to do now is to wait until a reviewer has dedicated time to review your data. If you have not enabled email-notifications in github, we suggest you check back into the issue you created on a regular basis.

3. Complete your contribution

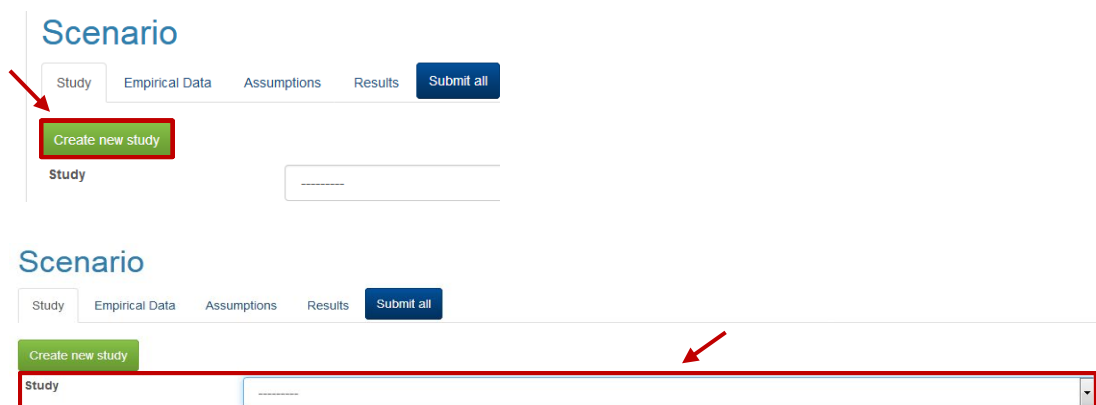
This document described how to upload data and metadata to the OpenEnergy Platform. For your contribution to become even more useful to other users of the platform and as further support for the community, we encourage you to create factsheets to describe the data you provided and their context in more detail.

3.1. Create Factsheets

- The drop-down menu shows you the 3 different Factsheet categories and the tab “Overview”, which offers an introduction and explanation of the Factsheet options
- Select a suitable factsheet category from the drop-down menu
 - Frameworks: to describe the fundamental structure or toolbox to build a model, e.g. TIMES
 - Models: to describe a model that has been used to run a scenario; ideally this relates to a scenario for which you have uploaded data to the OEP.
 - Scenarios: to describe a scenario; ideally one for which you have uploaded data to the OEP already.

Framework Factsheets 

- Each category has a plus button next to the title to access the fill out form
- Scenario Factsheet option offers you to create a new Scenario Factsheet for a new study or add a Scenario Factsheet to an existing study.



The image shows two screenshots of the 'Scenario' factsheet form. The top screenshot shows the 'Study' tab selected, with a red box around the 'Create new study' button and a red arrow pointing to it. The bottom screenshot shows the same form with a red box around the 'Study' dropdown menu and a red arrow pointing to it.

- Fill out the interactive fields and choose from the pre-defined responses
 - There are different tabs including further required fields marked with a *.

- When you are done – click on “Submit all”.
- Ideally you will /continue to create one Factsheet for each category to complete your contribution.