



Tutorial session

Setting-up a VESPA service using custom API from MongoDB

Setting up an EPN-TAB service and MongoDB

get instruction from:

https://git.cbk.waw.pl/tomasik/vespa_tutorial

Result expected:

*Running VO server

*Running mongoBD

Add some software to Dachs container

```
docker exec -it dachs /bin/bash
```

- apt update
- apt-get install mc | nano python iputils-ping python3-pip screen openssh-server
- pip3 install pymongo pandas flask waitress

Create user and activate ssh

```
useradd -m -d /home/myuser myuser
```

```
usermod -aG sudo myuser
```

```
passwd myuser
```

```
/etc/init.d/ssh start
```

now you can ssh to 127.0.0.1:2022 to have an nice console

create our REST API for databases in mongoDB

```
cd $home
```

```
mkdir mongoapi
```

```
cd mongoapi
```

```
wget --no-check-certificate -O mongo_create_rest_api.py
```

```
https://git.cbk.waw.pl/tomasik/vespa\_tutorial/-/raw/main/mongo\_create\_rest\_api.py?inline=false
```

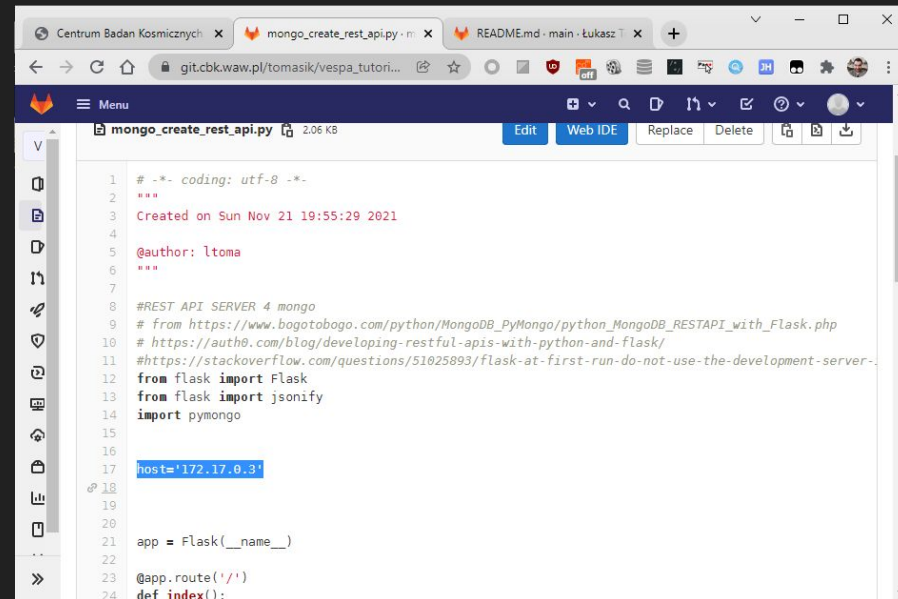
To find your mongoDB IP

run in cmd

```
docker inspect -f '{{range.NetworkSettings.Networks}}{{.IPAddress}}{{end}}'  
mongo4vespa
```

Correct the mongo Ip address

nano +17 mongo_create_rest_api.py



```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Sun Nov 21 19:55:29 2021
4
5 @author: ltoma
6 """
7
8 #REST API SERVER 4 mongo
9 # from https://www.bogotobogo.com/python/MongoDB_PyMongo/python_MongoDB_RESTAPI_with_Flask.php
10 # https://auth0.com/blog/developing-restful-apis-with-python-and-flask/
11 #https://stackoverflow.com/questions/51025893/flask-at-first-run-do-not-use-the-development-server-...
12 from flask import Flask
13 from flask import jsonify
14 import pymongo
15
16
17 host='172.17.0.3'
18
19
20
21 app = Flask(__name__)
22
23 @app.route('/')
24 def index():
```

Run the REST API for mongo server

run command

```
screen -d -m python3 mongo_create_rest_api.py
```

and see the world:

<http://127.0.0.1:5000/>

Create our first database in mongoDB

```
cd $home
```

```
mkdir mongodatabase
```

```
cd mongodatabase
```

```
wget --no-check-certificate -O mongo_create_database.py
```

```
https://git.cbk.waw.pl/tomasik/vespa\_tutorial/-/raw/main/mongo\_create\_database.py?inline=false
```

and edit the ip address as well:

```
nano +17 mongo_create_database.py
```

run command

```
python3 mongo_create_database.py
```

Visit the Mars :)

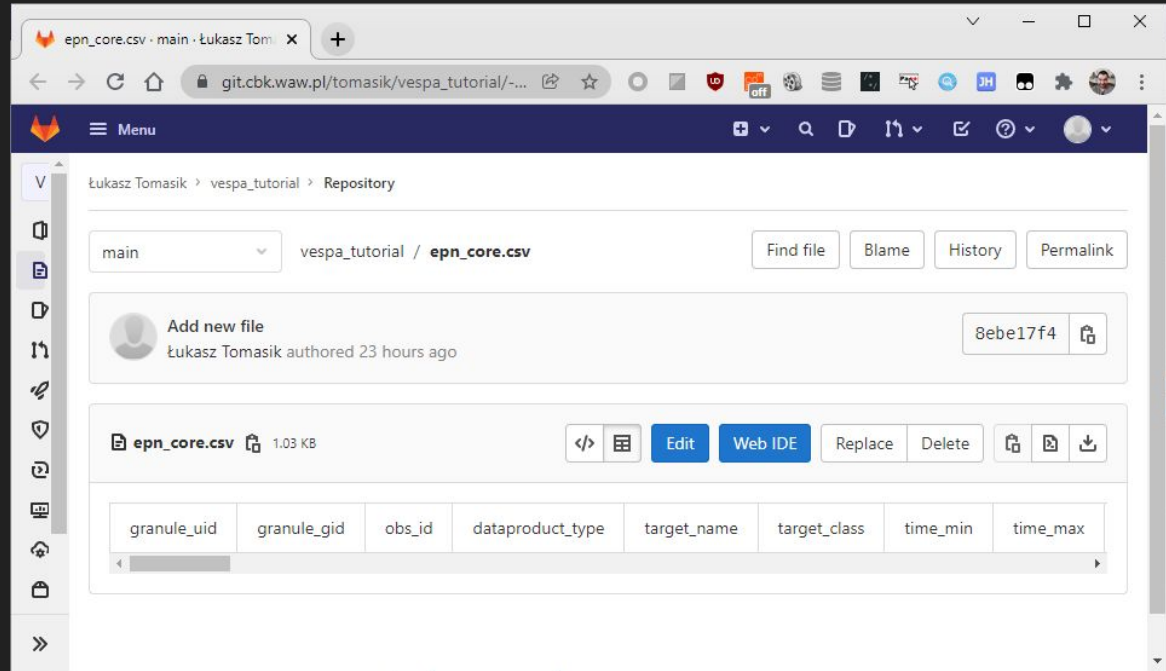
```
http://127.0.0.1:5000/planet/Mars
```

Import data using empty csv trik

```
cd /var/gavo/inputs
```

```
mkdir mongoplanet
```

```
cd mongoplanet
```



The screenshot shows a web browser window displaying a GitHub repository page for the file 'epn_core.csv'. The browser's address bar shows the URL 'git.cbk.waw.pl/tomasik/vespa_tutorial/~...'. The repository page includes a navigation bar with 'Menu', search, and other icons. The main content area shows the file path 'main vespa_tutorial / epn_core.csv' and buttons for 'Find file', 'Blame', 'History', and 'Permalink'. Below this, there is a section for 'Add new file' by 'Łukasz Tomasiak' with a commit hash '8ebe17f4'. The file 'epn_core.csv' is listed with a size of 1.03 KB and options for 'Edit', 'Web IDE', 'Replace', and 'Delete'. At the bottom, a table header is visible with columns: 'granule_uid', 'granule_gid', 'obs_id', 'dataprodect_type', 'target_name', 'target_class', 'time_min', and 'time_max'.

download files:

```
wget --no-check-certificate -O epn_core.csv  
https://git.cbk.waw.pl/tomasik/vespa\_tutorial/-/raw/main/epn\_core.csv?inline=false
```

```
wget --no-check-certificate -O input_template.json  
https://git.cbk.waw.pl/tomasik/vespa\_tutorial/-/raw/main/input\_template.json?inline=false
```

```
wget --no-check-certificate -O qmongo.rd  
https://git.cbk.waw.pl/tomasik/vespa\_tutorial/-/raw/main/qmongo.rd?inline=false
```

```
wget --no-check-certificate -O mongo2psql.py  
https://git.cbk.waw.pl/tomasik/vespa\_tutorial/-/raw/main/mongo2psql.py?inline=false
```

Check the `input_template.json`

`nano input_template.json`

it is helping generate query

```
mc [root@2f1dfac3027c]:/var/gavo/inputs/mongoplanet
GNU nano 5.4 input_template.json
(
  "granule_uid": "@mongoapi:target_name@",
  "granule_gid": "Planet",
  "obs_id": "@mongoapi:obs_id@",
  "dataproduct_type": "ci",
  "target_name": "@mongoapi:target_name@",
  "target_class": "planet",
  "time_min": "NULL",
  "time_max": "NULL",
  "time_sampling_step_min": "NULL",
  "time_sampling_step_max": "NULL",
  "time_exp_min": "NULL",
  "time_exp_max": "NULL",
  "spectral_range_min": "NULL",
  "spectral_range_max": "NULL",
  "spectral_sampling_step_min": "NULL",
  "spectral_sampling_step_max": "NULL",
  "spectral_resolution_min": "NULL",
  "spectral_resolution_max": "NULL",
)
[ Read 67 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste    ^J Justify  ^_ Go To Line
```

now set up the mongoplanet:

```
gavo val qmongo.rd
```

```
gavo imp qmongo.rd
```

```
dachs serve restart
```

see your empty table

http://127.0.0.1:8080/_system_/adql/query/form?nevow_form=genForm&query=select%20*%20from%20mongoplanet.epn_core

Run the script

```
python3 mongo2psql.py
```

and check your mongoplanet page:

http://127.0.0.1:8080/system/adql/query/form?nevow_form=genForm&query=select%20*%20from%20mongoplanet.epn_core