I. Install softwares and set up

Supposing you already have a configured and functional Ubuntu installed on your laptop, and Python version 3.7 installed, I strongly advise you to use the Pycharm IDE for all the python related sections. Let's start by installing all the software required.

I.1 InfluxDB

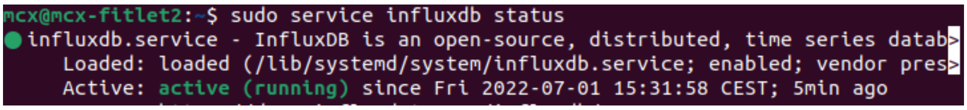
First, go to https://portal.influxdata.com/downloads/ to download latest Ubuntu version of influxDB Open-Source.

Once you successfully downloaded it through the bash, you may check everything's ok:

Launch the influxDB service through the command: "`sudo service influxdb start`"

Then check if it has been done properly by executing: "`sudo service influxdb status`"

You should see something similar to:

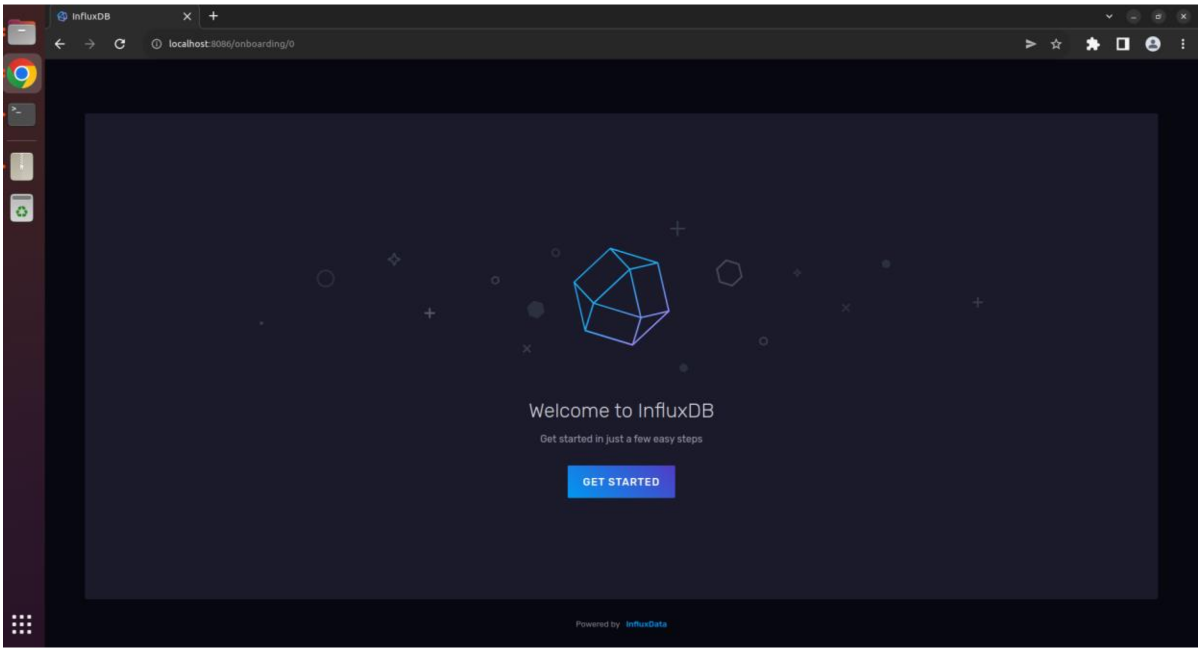


```
mcx@mcx-fitlet2:~$ sudo service influxdb status
● influxdb.service - InfluxDB is an open-source, distributed, time series datab>
     Loaded: loaded (/lib/systemd/system/influxdb.service; enabled; vendor pres>
     Active: active (running) since Fri 2022-07-01 15:31:58 CEST; 5min ago
```

I.1 InfluxDB

Open a browser and type: localhost:8086

This page should be loaded:



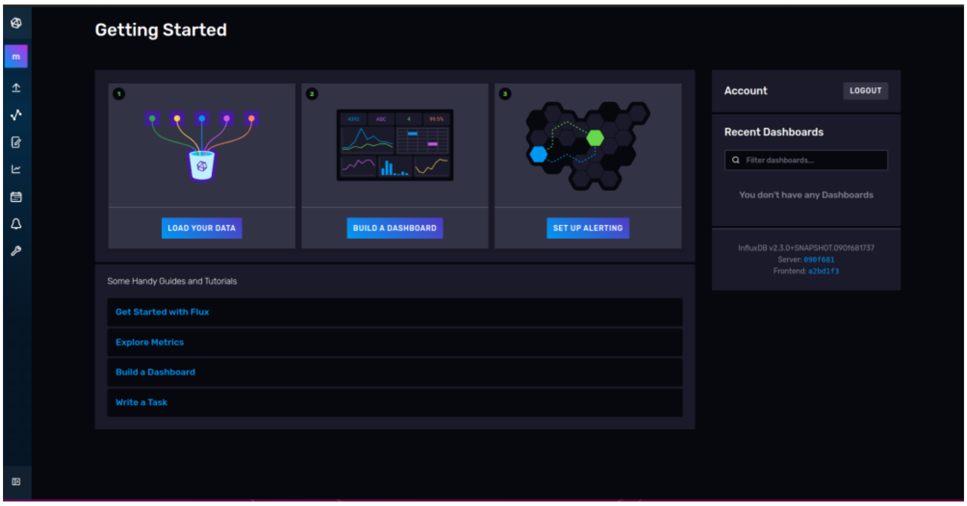
Welcome to InfluxDB

Get started in just a few easy steps

You simply have to follow the step in the page. Name your first bucket ”RawData” and your organization ”mcx”.

Make sure to store your credentials somewhere because it is a pain to recover if you lost them

You should access the InfluxDB User Interface by clicking on “Configure Later” option at the end.



You don't have any Dashboards

InfluxDB v2.3.0+SNAPSHOT.090f681737
Server: 090f681
Frontend: a2bd1f3

Now, we will create the different buckets used to store the data with different update rate and configure the tasks that will run to preprocess the data and store them automatically in the different buckets.

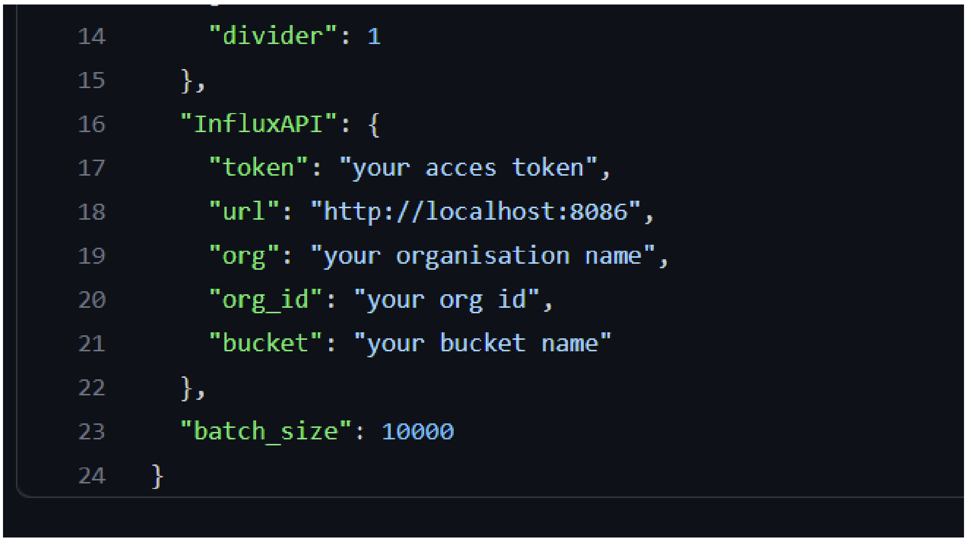
This process will be done automatically by running a Python script.

1st step:

download the whole repository on this link: https://github.com/timphil174/motorcortexAPI

And open the config.json file

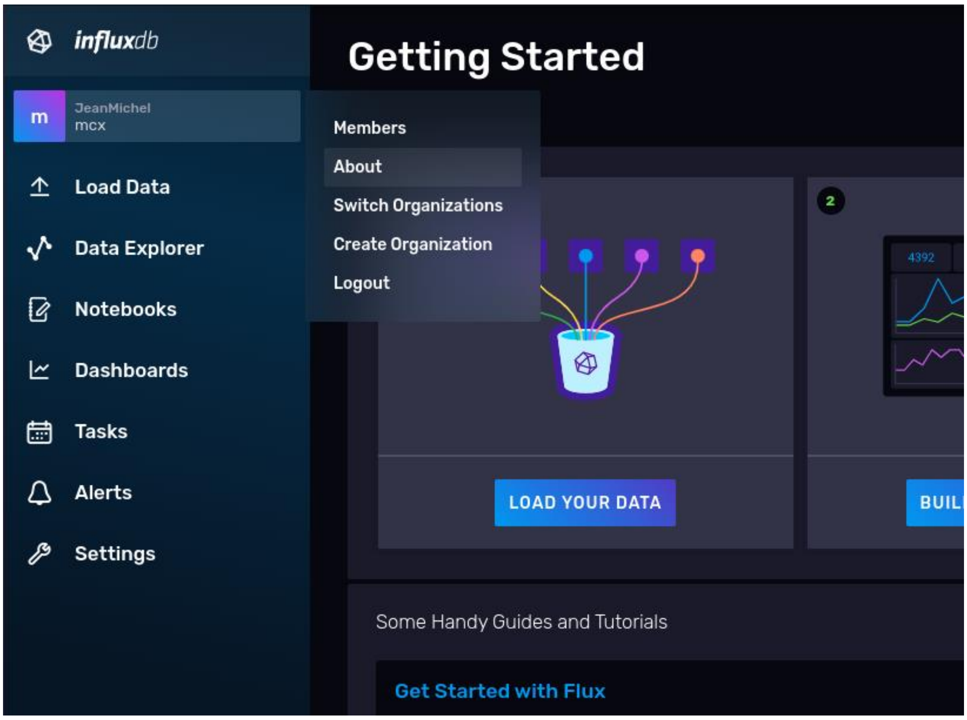
You’ll need to fill some information regarding your influxDB account that will make the settings easier.



```
      "divider": 1
    },
    "InfluxAPI": {
      "token": "your acces token",
      "url": "http://localhost:8086",
      "org": "your organisation name",
      "org_id": "your org id",
      "bucket": "your bucket name"
    },
    "batch_size": 10000
  }
```

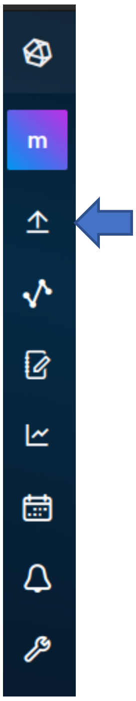
Your organization name should be “mcx” and your bucket name “RawData”

To get your organization id, get on the navigation menu and click on the “about” section:

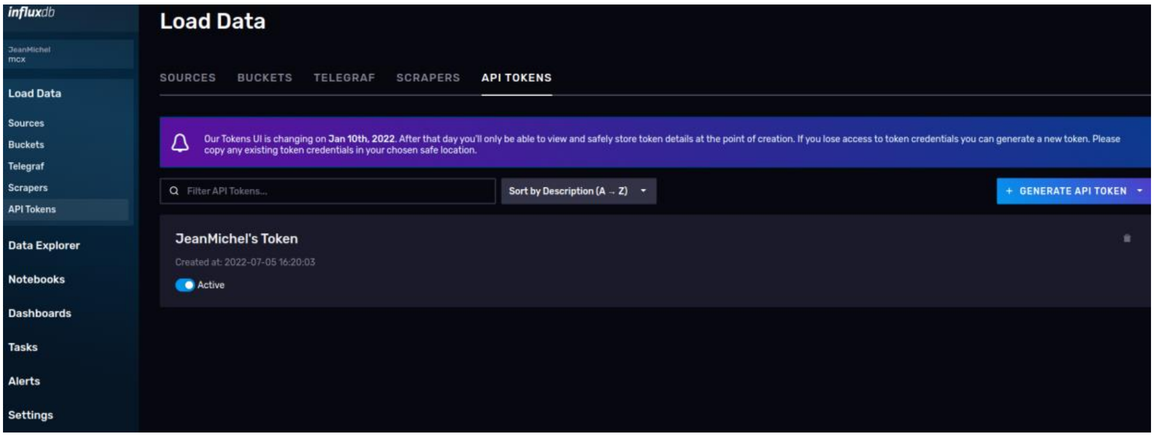


Simply copy to clipboard the organization id and paste it in the "org_id" section in the config file.

In order to get your token, get on the navigation menu and click on the “load data” section



Then Click on the “API TOKENS” section:



Load Data

Our Tokens UI is changing on **Jan 10th, 2022**. After that day you'll only be able to view and safely store token details at the point of creation. If you lose access to token credentials you can generate a new token. Please copy any existing token credentials in your chosen safe location.

JeanMichel's Token

Created at: 2022-07-05 16:20:03

Active

Click on your token and copy it to your clipboard in order to paste it in the corresponding section in the config.json file

Finally, you can save and close your config file.

2nd Step:

In order to run the Python script, you need to install the packages used.

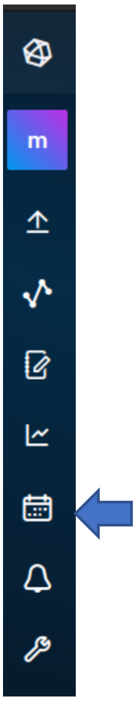
In Pycharm, with the Python Packages widget, search for influxdb-client package and install it.

Then execute the script createBucketAndTasks.py on pycharm.

3rd Step: Verification

To verify that the process has been successfully done, you can click again on the “load data” section in the navigation tab and get in the “bucket” section. You should be able to see 9 buckets including the one you created at the beginning called “RawData”.

Also verify that the tasks have been successfully created by going on the “tasks” section in the navigation menu:



You should be able to see a total of 8 tasks, all already active.

You have to get on this tab, edit the task called “1s_min” and change the last line :

From “|> to(bucket: “1s_max”, org : “mcx”)” to “|> to(bucket: “1s_min”, org : “mcx”)”

Sorry for the inconvenience.

I.2 MySQL

The datalogger python script also comes with the possibility to write your data into MySQL databases. The MySQL setup also provides a lot of convenience for the data visualization on Grafana.

The installation and setup of MySQL is pretty quick:

1. Open a terminal, type “sudo apt update” then “sudo apt upgrade”

When done, type “sudo apt install mysql-server"

then check the installation by typing “mysql --version”

At this point, mysql is already installed.

We will now setup your mysql password:

Execute: 1. sudo mysql

2. ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password by 'mynewpassword';

3. quit;

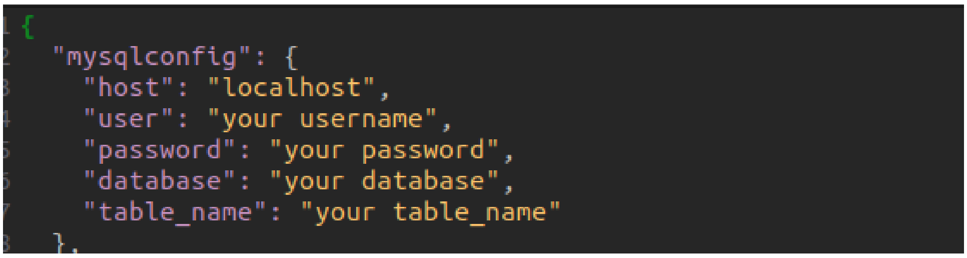
4. sudo mysql_secure_installation

The bash will ask you for a password which is ‘mynewpassword’, you can now change your password by following the instructions in the bash.

We will finish the Mysql setup with the creation of a database and the completion of your config file:

1. Type sudo mysql –u root --password
2. Type “create database mydatabase;”

You can now open config.json file and get in the first section:



```
{
  "mysqlconfig": {
    "host": "localhost",
    "user": "your username",
    "password": "your password",
    "database": "your database",
    "table_name": "your table_name"
  },
```

At this point, your username should be “root”, your database should be “mydatabase”.

Simply fill in your password and the name of your choice of a table which is going to be created by the python script soon.

I.3 Grafana

First, go to https://grafana.com/grafana/download?pg=get&plcmt=selfmanaged-box1-cta1 to download latest Linux version of Grafana

It should be done for the grafana downloading if no error is showing up after you executed the command line in the bash.

Grafana Setup

Open another tab and type: localhost:3000

The default credentials are: -user : admin

-password : admin

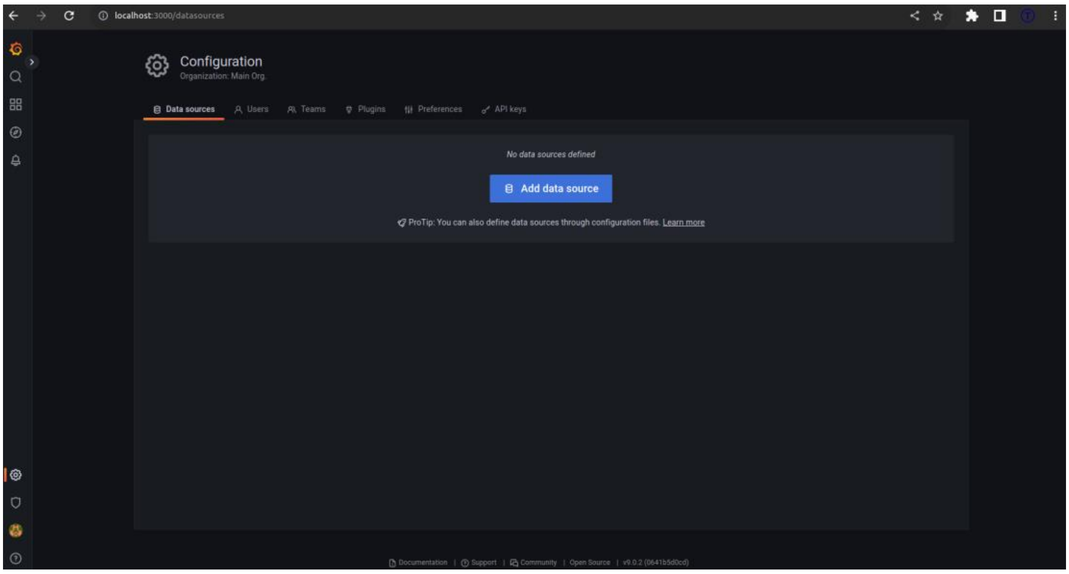
Once you are connected, you can change your password.

Then you should be sent on the User Interface.

The next step is to configure the data sources to link Grafana with InfluxDB and MySQL.

1. InfluxDB connection

On the left navigation menu, pass on the settings widget and click on “Data Sources”:

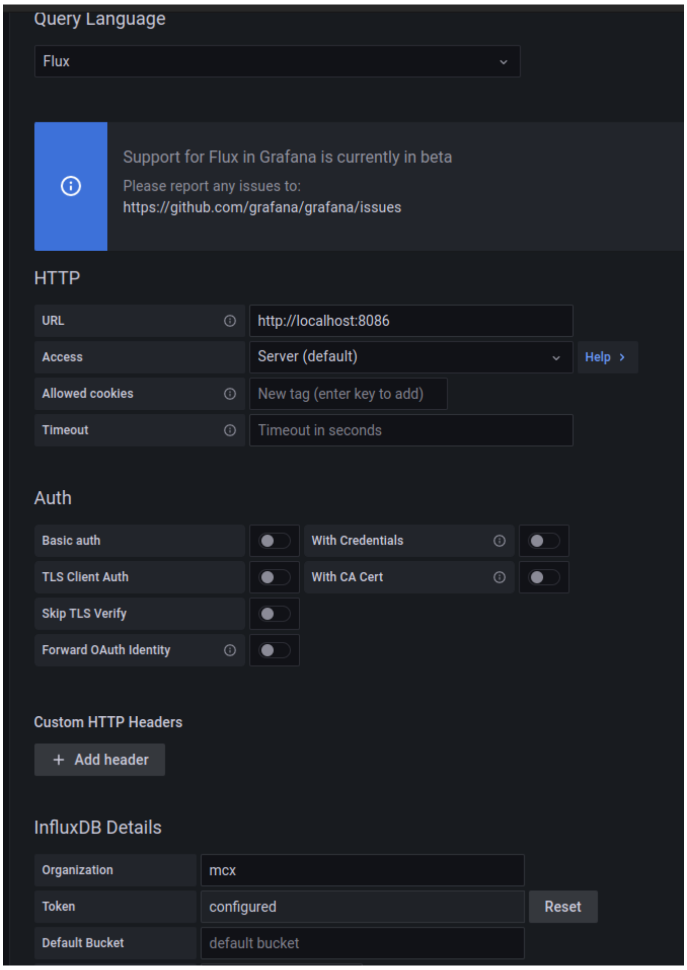


No data sources defined

ProTip: You can also define data sources through configuration files. Learn more

Click on “Add data source” and select InfluxDB.

Select “Flux” for the query language, simply reproduce this setup, with your token that should be stored in your config file.



Query Language

Flux

Support for Flux in Grafana is currently in beta

Please report any issues to:
https://github.com/grafana/grafana/issues

HTTP

URL	http://localhost:8086
Access	Server (default)
Allowed cookies	New tag (enter key to add)
Timeout	Timeout in seconds

Auth

Basic auth	○	With Credentials	○
TLS Client Auth	○	With CA Cert	○
Skip TLS Verify	○		
Forward OAuth Identity	○		

Custom HTTP Headers

InfluxDB Details

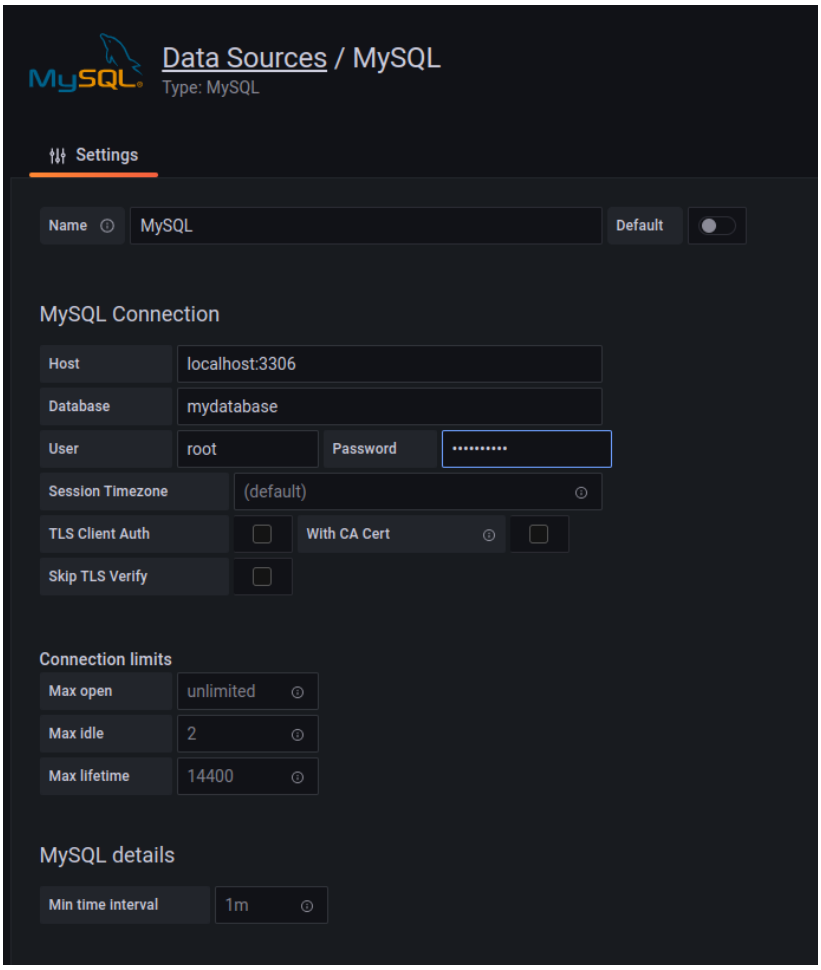
Organization	mcx
Token	configured
Default Bucket	default bucket

Once you pressed the “Save and Test” button, Grafana should be able to find your several buckets. If so, the InfluxDB connection is successfully done.

2. MySQL connection

Click again on “Add data source”, scroll down in order to select “MySQL”.

Simply reproduce this setup with your password that should be stored in your config file.



Data Sources / MySQL
Type: MySQL

Name: MySQL
Default: ○

MySQL Connection

Host: localhost:3306
Database: mydatabase
User: root
Password: ••••••••••
Session Timezone: (default)
TLS Client Auth: □
With CA Cert: □
Skip TLS Verify: □

Connection limits

Max open: unlimited
Max idle: 2
Max lifetime: 14400

MySQL details

Min time interval: 1m

Click “Save & test”. You should see a “Database Connection OK” message. If so, the MySQL connection is successfully done.

II. Data Writing and Visualization

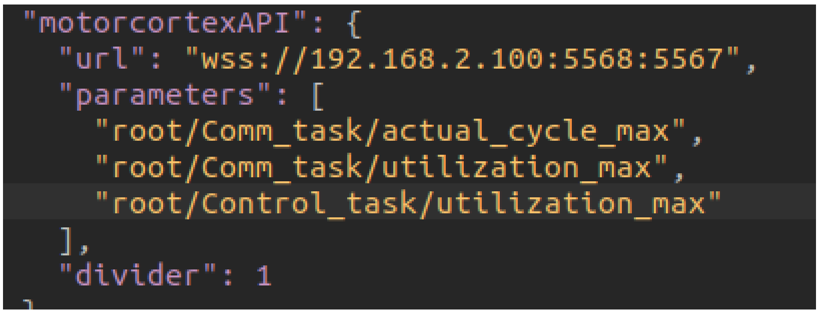
II.1 Data Writing

At this point, the software installation and setup are almost done.

You now must fill the list of signals you want to record and write into the databases.

To do so, open the config file and fill in the “parameters” section with the signals name.

For example, here is a configuration of a 3 signals record:



```
"motorcortexAPI": {
  "url": "wss://192.168.2.100:5568:5567",
  "parameters": [
    "root/Comm_task/actual_cycle_max",
    "root/Comm_task/utilization_max",
    "root/Control_task/utilization_max"
  ],
  "divider": 1
```

You can fill as much signals as you want. To get a list of available signals generated by Motorcortex, simply go on the Motorcortex Desk. Once you are done, save and close the config file.

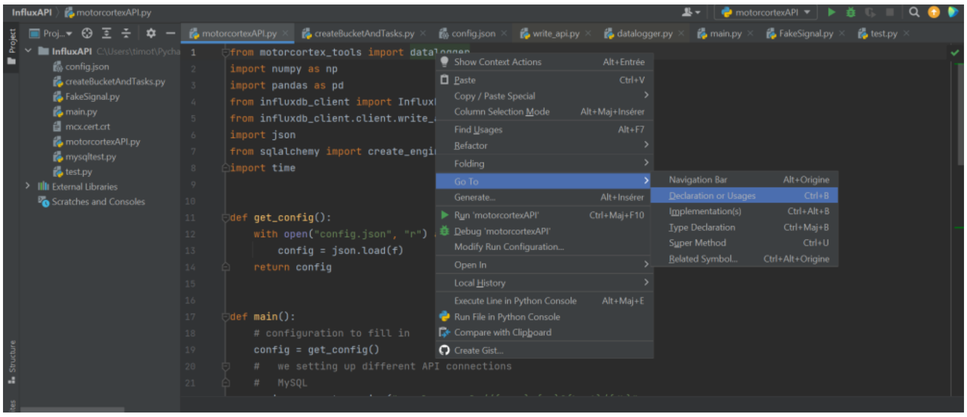
We are now in the last step before launching the data writing script. We have to install the Python package used in the python script.

In Pycharm, get on the "Python Package" widget and install the packages in this order:

1. pyngg
2. motorcortex-python-tools
3. pymysql
4. Sqlalchemy

You also have to edit a a little formula in the datalogger.py file.

To access this file, simply get on the motorcortexAPI.py, and in the first line of the code, right click on "datalogger", then "Go to" then "Declarations or Usages"

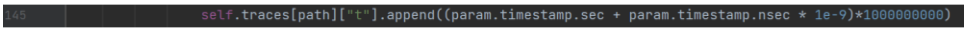


```
from motorcortex_tools import datalogger
import numpy as np
import pandas as pd
from influxdb_client import Influx
from influxdb_client.client.write_
import json
from sqlalchemy import create_engi
import time


def get_config():
    with open("config.json", "r")
        config = json.load(f)
    return config


def main():
    # configuration to fill in
    config = get_config()
    #   we setting up different API connections
    #   MySQL
```

Then at the line 145, just modify the formula.



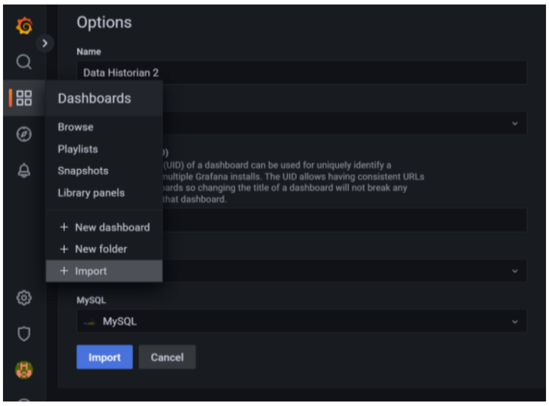
```
145                 self.traces[path]["t"].append((param.timestamp.sec + param.timestamp.nsec * 1e-9)*1000000000)
```

So the timestamps are in a proper way to get ingested by influxdb

You should be able to run the motorcortexAPI.py script and see the data writing in real-time.

To be able to properly visualize your signals, you have to import the following json template in grafana located in the github repository.

To do so, click on this "import" widget and import the JSON file called "Data Historian-1659619809016.json"



Options

Name

Data Historian 2

)
(UID) of a dashboard can be used for uniquely identify a
multiple Grafana installs. The UID allows having consistent URLs
ards so changing the title of a dashboard will not break any
that dashboard.

MySQL

MySQL

You should be now able to visualize properly the signals of your choice.