Solar Visualization Simulator

Introduction

This project aims to provide a simulation of a visualization system for a solar grid. The system is divided in 3 docker containers:

- Grafana
- InfluxDB
- Python (optional)

Installation

To run the containers, an installation of docker is required to proceed.

Linux (or WSL) users might install Docker Engine and Docker Compose. It is recommended to follow *https://docs.docker.com/engine/install/linux-postinstall*, and add the desired local user to the *docker* group.

Otherwise, in **Windows** and **macOS**, users can install Docker Desktop to manage the containers with a GUI. For more details, user should follow the official instructions at *https://docs.docker.com/get-docker/*. It is worth mentioning that Windows users should install WSL previously.

Make sure to create a network in which the containers will communicate with. In a terminal (WSL terminal in Windows):

```
docker network create <name_network_in_docker_compose_file>
```

To run *main.py* users are encouraged to create a python virtual environment. To do so, you should enter in the directory where the files of this project are located.

```
python -m venv solar_venv
source solar_venv/bin/activate
pip install -r requirements.txt
```

To activate the environment, you should execute the following command:

```
source ./solar_venv/bin/activate
```

To deactivate it

```
deactivate
```

Optionally, users may select to run the code in an additional container (optional part).

Running

Prior to run the application, you may proceed to complete the Roadmap section.
After completing it, it is time to run the application and configure the final aspects, especially the ones in Grafana.

In *deployment/*:

```
docker compose up
python ../main.py
```

Roadmap

1. **Download weather data.**
   You must filter (do a Query) by DATA_LECTURA. These values must be compressed between 01/01/2025 to 31/12/2025. Also, you must specify CODI_ESTACIO that must be XV. When these filters are established, you can proceed to download the data as a CSV file. The link from where you can download the data is: https://analisi.transparenciacatalunya.cat/en/Medi-Ambient/Dades-meteorol-giques-de-la-XEMA/nzvn-apee/data_preview Useful information:

   - https://analisi.transparenciacatalunya.cat/Medi-Ambient/Metadades-estacions-meteorol-giques-autom-tiques/yqwd-vj5e/data_preview
   - https://analisi.transparenciacatalunya.cat/en/Medi-Ambient/Metadades-variables-meteorol-giques/4fb2-n3yi/data_preview (in Catalan, you may translate it).

   You must store the downloaded csv file as weather_data_unprocessed.csv in WeatherData/Data/ folder.
2. **Preprocess the data.**
   Process the measurements from *WeatherData/Data/weather_data_unprocessed.csv* and put them into a more usable format. For that purpose, you need to complete the TODO from the Python Script *data_preprocessing.py* in *WeatherData* folder.
3. **Finish TODOs in main.py and influx_conf.py**
   Prior to configure the stack of containers, the TODOs in the main.py file must be completed. The same applies for the influx_conf.py.

4. **Finish docker_compose.yml configuration**
   To construct the stack of containers, you have to open the docker-compose.yml file in *deployment/* folder and complete the TODOs.

For more info on Grafana, see https://grafana.com/docs/grafana/latest/ and https://grafana.com/docs/grafana/latest/setup-grafana/installation/docker/.

For more info on InfluxDB, see https://docs.influxdata.com/influxdb/v2/ and https://hub.docker.com/_/influxdb.

5. **Build the application stack**

   Now, it is time to run the application. As a result, you will have two containers running (Grafana + InfluxDB). Enter in Grafana container through a web browser (http://localhost:<PORT OF GRAFANA>) and enter the defined credentials. Once you are inside the service, you need to configure the data sources. Since Grafana is in the same network as InfluxDB container, you only need to specify the name of the container as the host.

6. **Build your dashboard**

   Once you have the connection of the data sources with Grafana, it is time to build your own dashboards. Be creative!

7. **(Optional) Dockerize the python scripts.**

   Now, we are going to Dockerize the python script. With this, we will have our application fully based on containers. This will allow us to transfer it wherever we want and ensure its correct behavior.

   a. Create a Python_Script folder into deployment folder.
   b. Copy *influx_config.py, main.py, pv_system_config.py* and *requirements.txt* into the Python_Script folder.
   c. Modify *influx_config.py* and *main.py* to store information in the InfluxDB container (specifying the name of the container instead of localhost).
   d. Inside this folder, create a Dockerfile to execute the python script. You must complete (...) with the correct values.

```
FROM python:3.10.13-slim

WORKDIR /solar_simulation

COPY ./Python_Script/(...) .
COPY ./Python_Script/(...) .
COPY ./Python_Script/(...) .
COPY ./Python_Script/(...) .

RUN pip install --upgrade pip RUN
pip install -r (...)
CMD [ "python3", "-u", "(...)" ]
```

   e. Add the following into the docker-compose.yml

```
 python_code:
  container_name: python
depends_on:
    - influxdb
build:
context: ./
```

```
   dockerfile: ./Python_Script/(...)
restart: (...)
```

f. Now, in *deployment/*

```
docker compose up
```

g. Check that all is correctly working

Extra information:
https://docs.docker.com/guides/python/containerize/