1. 테스트 환경 정보

- OS : Ubuntu 22.04
- GPU : NVIDIA H100 80GB
- CUDA : 12.2
- cuDNN : 8.9.6
- Driver Version : 535.104.12

2. 개발 환경 구성

(2-1) 컨테이너 관련 설정

- 이미지 다운로드
  - docker pull nvidia/cuda:12.2.2-cudnn8-devel-ubuntu22.04
    - ✓ 사용자 환경에 따라 상이하며, Driver Version, Cuda Version등 호환 여부 확인 필요

```
(base) ubuntu@gpu-1:~$ docker images
REPOSITORY                                TAG                       IMAGE ID       CREATED         SIZE
node                                      lts                       4c2ff0421257   4 weeks ago     1.1GB
winglian/axolotl                          <none>                    92c3259fb975   7 months ago    20.1GB
kalilinux/kali-rolling                    latest                    74603322e656   7 months ago    127MB
ghcr.io/huggingface/text-generation-inference   1.4.2                 d30bf5cea0a4   8 months ago    10.3GB
nvidia/cuda                               12.2.2-cudnn8-devel-ubuntu22.04   589cdbe398c6   12 months ago   9.33GB
```

- 컨테이너 생성
  - docker run -it -d --gpus all -p 10088:8888 -p 10555:5000 --name test_workspace <image id> /bin/bash

```
(base) ubuntu@gpu-1:~$ docker run -it -d --gpus all -p 10088:8888 -p 10555:5000 --name test_workspace 589cdbe398c6 /bin/bash
32bb99076cff892b31317a8b07da63c8168a1315d03eb65c88ac448b20b692ec
(base) ubuntu@gpu-1:~$ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED         STATUS         PORTS                                                                                  NAMES
32bb99076cff   589cdbe398c6           "/opt/nvidia/nvidia_…"   4 seconds ago   Up 4 seconds   0.0.0.0:10555->5000/tcp, :::10555->5000/tcp, 0.0.0.0:10088->8888/tcp, :::10088->8888/tcp   test_workspace
```

- 컨테이너 접속
  - docker exec -it test_workspace /bin/bash

```
(base) ubuntu@gpu-1:~$ docker exec -it test_workspace /bin/bash
root@32bb99076cff:/#
```

- 디렉토리 생성 및 이동
  - 생성 예) mkdir workspace
  - 이동 예) cd workspace

```
root@32bb99076cff:/# ls
NGC-DL-CONTAINER-LICENSE  boot                     dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
bin                       cuda-keyring_1.0-1_all.deb  etc  lib   lib64  media  opt  root  sbin  sys  usr
root@32bb99076cff:/# mkdir workspace
root@32bb99076cff:/# cd workspace/
```

- 컨테이너 내부 패키지 업데이트 및 설치
  - apt-get update
  - apt-get install -y git curl wget python3-pip

(2-2) 코드 가져오기

- git lfs 설치
  - wget https://github.com/git-lfs/git-lfs/releases/download/v2.13.3/git-lfs-linux-amd64-v2.13.3.tar.gz
  - tar xvfz git-lfs-linux-amd64-v2.13.3.tar.gz
  - ./install.sh

```
root@32bb99076cff:/workspace# ./install.sh
Git LFS initialized.
```

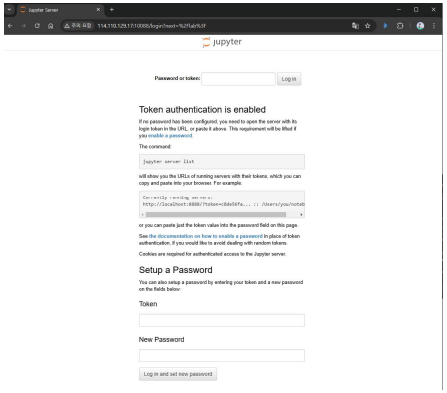
- git repo clone 수행
  - git clone https://github.com/etri/FeDiT.git
- 경로 이동
  - 대상 경로 : FeDiT/OSFW/DTAI_m2m/Source/

(2-3) 주피터 사용 환경 구축

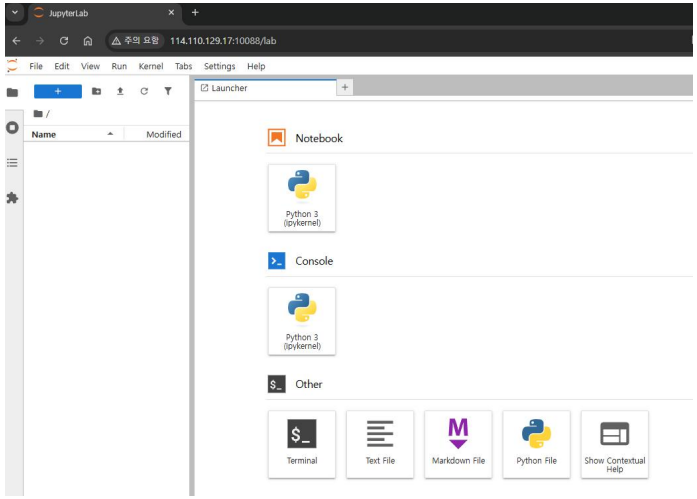
- 라이브러리 설치
  - pip install jupyterlab
- 실행
  - jupyter lab --ip=0.0.0.0 —port=8888 --allow-root
  - 실행 시 생성된 Token은 접속 시 요구되는 패스워드 확인

```
root@e38dcc572ec4:/workspace# jupyter lab --ip=0.0.0.0 --port=8888 --allow-root
[I 2024-10-31 09:31:11.235 ServerApp] jupyter_lsp | extension was successfully linked.
[I 2024-10-31 09:31:11.238 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2024-10-31 09:31:11.242 ServerApp] jupyterlab | extension was successfully linked.
[I 2024-10-31 09:31:11.452 ServerApp] notebook_shim | extension was successfully linked.
[I 2024-10-31 09:31:11.466 ServerApp] notebook_shim | extension was successfully loaded.
[I 2024-10-31 09:31:11.468 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2024-10-31 09:31:11.469 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2024-10-31 09:31:11.470 LabApp] JupyterLab extension loaded from /usr/local/lib/python3.10/dist-packages/jupyterlab
[I 2024-10-31 09:31:11.470 LabApp] JupyterLab application directory is /usr/local/share/jupyter/lab
[I 2024-10-31 09:31:11.470 LabApp] Extension Manager is 'pypi'.
[I 2024-10-31 09:31:11.540 ServerApp] jupyterlab | extension was successfully loaded.
[I 2024-10-31 09:31:11.540 ServerApp] Serving notebooks from local directory: /workspace
[I 2024-10-31 09:31:11.540 ServerApp] Jupyter Server 2.14.2 is running at:
[I 2024-10-31 09:31:11.540 ServerApp] http://e38dcc572ec4:8888/lab?token=f2b0c40506a31bee5111463418d566460fb756468d5ed56c
[I 2024-10-31 09:31:11.540 ServerApp]     http://127.0.0.1:8888/lab?token=f2b0c40506a31bee5111463418d566460fb756468d5ed56c
[I 2024-10-31 09:31:11.540 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[W 2024-10-31 09:31:11.544 ServerApp] No web browser found: Error('could not locate runnable browser').
```

- 접속
  - 토큰 입력 및 패스워드 생성 후 접속 가능



1) 접속 화면



(2-4) 관련 필수 라이브러리 설치

- 라이브러리 관련 툴 poetry 설치
  - curl -sSL https://install.python-poetry.org ¦ python3 -
- 환경변수에 추가
  - export PATH="$HOME/.local/bin:$PATH"
- vim 설치
  - apt-get install -y vim
- bashrc 열기
  - vim ~/.bashrc
- 다음 명령어 추가 후 저장
  - export PATH="$HOME/.local/bin:$PATH"
- 현재 셀 세션에 적용
  - source ~/.bashrc
- 관련 라이브러리 설치
  - poetry install
- my_poetry_env 라는 이름으로 ipykernel 추가
  - poetry run python -m ipykernel install --user --name=my_poetry_env
- python-opencv 관련 라이브러리 추가설치
  - apt-get install -y libgl1-mesa-glx
  - apt-get install -y libglib2.0-0

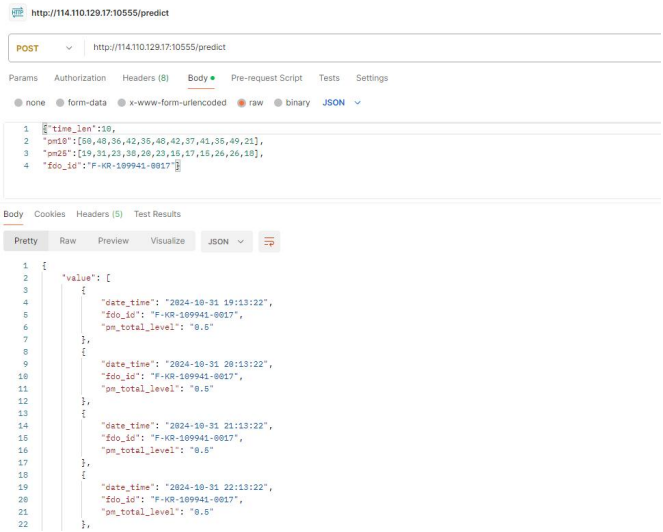
3. 실행

(3-1) 학습 및 테스트 수행

- 주피터 노트북에서 실행 시, 이전에 생성한 my_poetry_env 라는 ipykernel 선택 필수
- 경로 이동
  - FeDiT/OSFW/DTAI_m2m/Source/CFD
- 0_데이터세트생성.ipynb 순차적 실행
  - 라이브러리 import
  - 데이터세트 가져오기
  - 데이터세트 전처리
- 1_학습 및 테스트.ipynb 순차적 실행
  - 라이브러리 import
  - 데이터세트 로드
  - 모델 지정
  - 각종 파라미터 지정
  - 학습 및 테스트 수행

(3-2) 1차 PoC 테스트

- 경로 이동
  - FeDiT/OSFW/DTAI_m2m/Source/POC
- 실행
  - poetry run python poc.py
- request 테스트



- request 파라미터
  - ✓ time_len : 예측할 시간 (단위:hour)
  - ✓ pm10 : 12개 포인트에 대한 pm10 수치값
  - ✓ pm25 : 12개 포인트에 대한 pm2.5 수치값
  - ✓ fdo_id : 관광지 위치 정보
- response 파라미터
  - ✓ data_time : 예측 시간
  - ✓ fdo_id : 관광지 위치 정보
  - ✓ pm_total_level : pm 수치 정도