**Docker Guide-Book**

**Ubuntu 실행 후, 회사 IP에 접근하기**

* **MobaXterm으로 작업하면 Server 폴더 및 경로를 시각적으로 확인할 수 있어서**

**가능하다면 MobaXterm으로 작업할 것을 추천!**

**MobaXterm에서 Linux환경의 Server 접근 및 제어 모두 Ubuntu에서와 동일하게 가능!**

root@SVhd08:~# **ssh sehyun\_2022si@10.50.20.37**

sehyun\_2022si@10.50.20.37's password: **ss2022**

**Ubuntu Docker 명령**

**ls**  현재 위치의 모든 폴더 표시

**cd ..** 직전 폴더로 이동

**sudo /bin/bash** 매번 sudo 명령어 입력하기 싫으면 이 코드 추가하면 된다. – **이 코드로 서버 접근!**

**cd media/hdd/**  media/hdd 폴더로 이동

**sudo mkdir sehyun** sehyun 폴더 생성

**docker ps** 현재 작동중인 Container가 어떤 것이 있는지 확인

**docker ps -a** 전체 Container 확인

**docker images**  현재 Local Docker Server안에 다운로드 되어 있는 Image는 어떤 것이 있는지 확인가능

**docker run -it -v /media/hdd/sehyun:/sehyun --name "sehyun" --shm-size 16G python:3.9.12 /bin/bash**

docker run -it container를 만든다는 의미

-v Volume으로 연결하겠다는 의미, host Directory를 Container에서 쓸 수 있도록 설정

Sehyun: host 이름

Sehyun container에서의 경로 설정

--name container이름 설정

--shm-size 16G shared memory를 지정해서, Deep Learning 모델을 돌릴 때 에러가 나지 않도록 설정

python:3.9.12 사용할 이미지 이름 지정

**다양한 옵션**

**docker run -it -rm -v /media/hdd/sehyun:/sehyun –name ‘sehyun’ --shm-size 16G python:3.9.12 /bin/bash**

**-rm**  container열려 있는 동안만 사용하고, container를 내리면 docker도 같이 종료되게 지정

**docker container를 만들고 확인해볼 것**

[GPU 환경 확인]  **nvidia-smi**

[cuda 환경 확인]  **nvcc --version**

이 두 개가 정상적으로 출력이 되어야 Container가 정상적으로 만들어진 것!

**Ctrl + P + Q Docker Container 활성화시킨 채로 나가기**

**Ctrl + D Docker Container 종료하고 나가기**

**Docker stop Sehyun Sehyun 컨테이너 종료하기**

**Docker rm Sehyun Sehyun 컨테이너 삭제하기**

**GPU를 사용할 수 있는 Docker Container 생성하기**

nvidia-docker run --gpus='"device=6,7"' -it -v /media/hdd/sehyun:/sehyun --name "sehyun" --shm-size 16G python:3.9.12 /bin/bash

**NVCC활성화 시키는 Docker Container 생성하기**

**nvidia-docker run --gpus='"device=6,7"' -it -v /media/hdd/sehyun:/sehyun --name "sehyun" --shm-size 16G nvcr.io/nvidia/pytorch:21.12-py3 /bin/bash**

**Linux Command 창에서 코드 실행하기**

**입력 :** python

Import torch

Torch.cuda.is\_available()

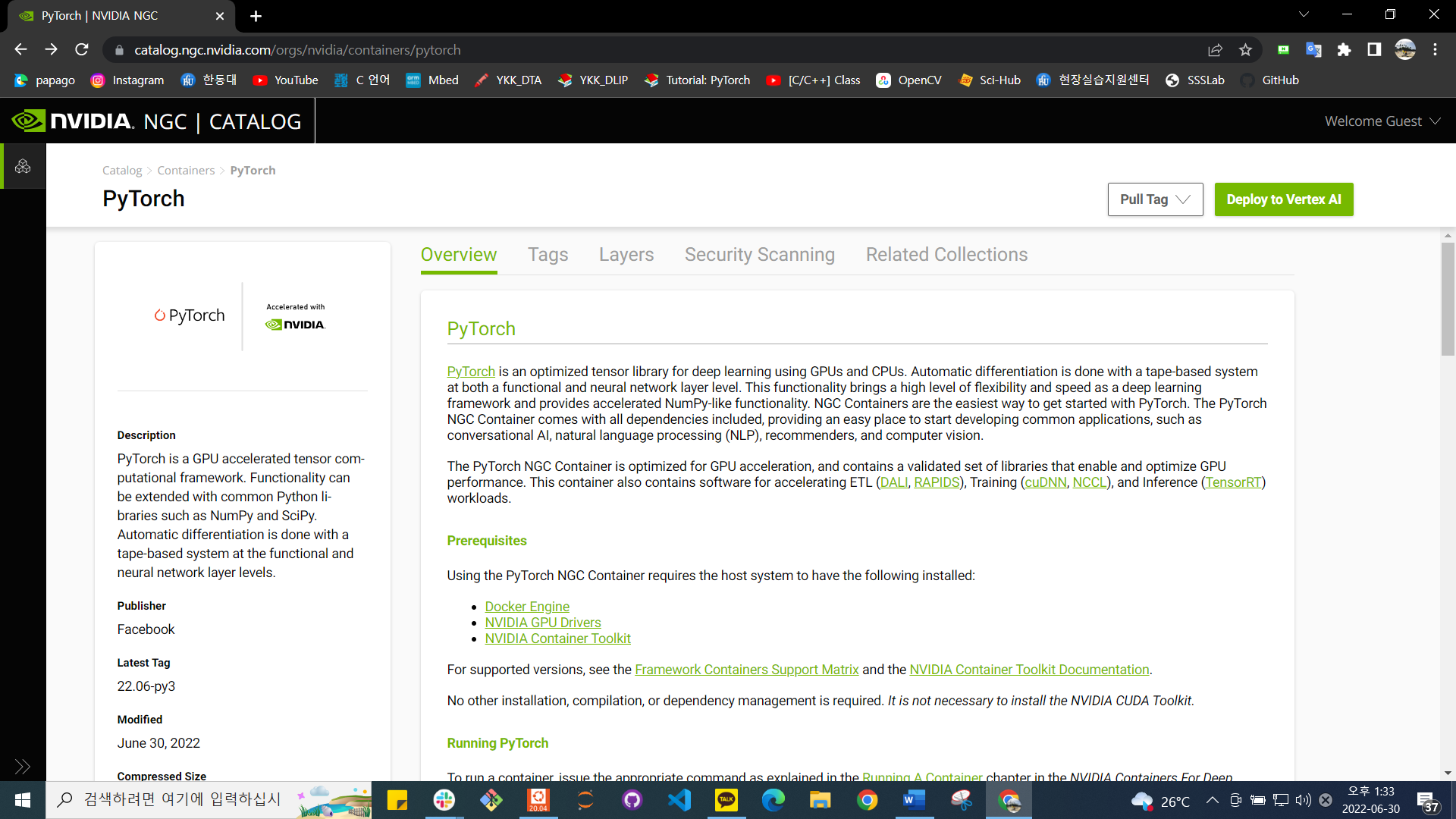
여기서 나오는 것이 **True**라면, **Cuda/GPU Setting**이 모두 **완료**!

**환경설정 방법 – Image Download Site**

**NV NGC** = Docker images를 받을 수 있는 곳

(+) Docker hub 에서도 Image Downloading 가능!

**[1]** Google - **nvcr pytorch** 검색 – 사이트 접속

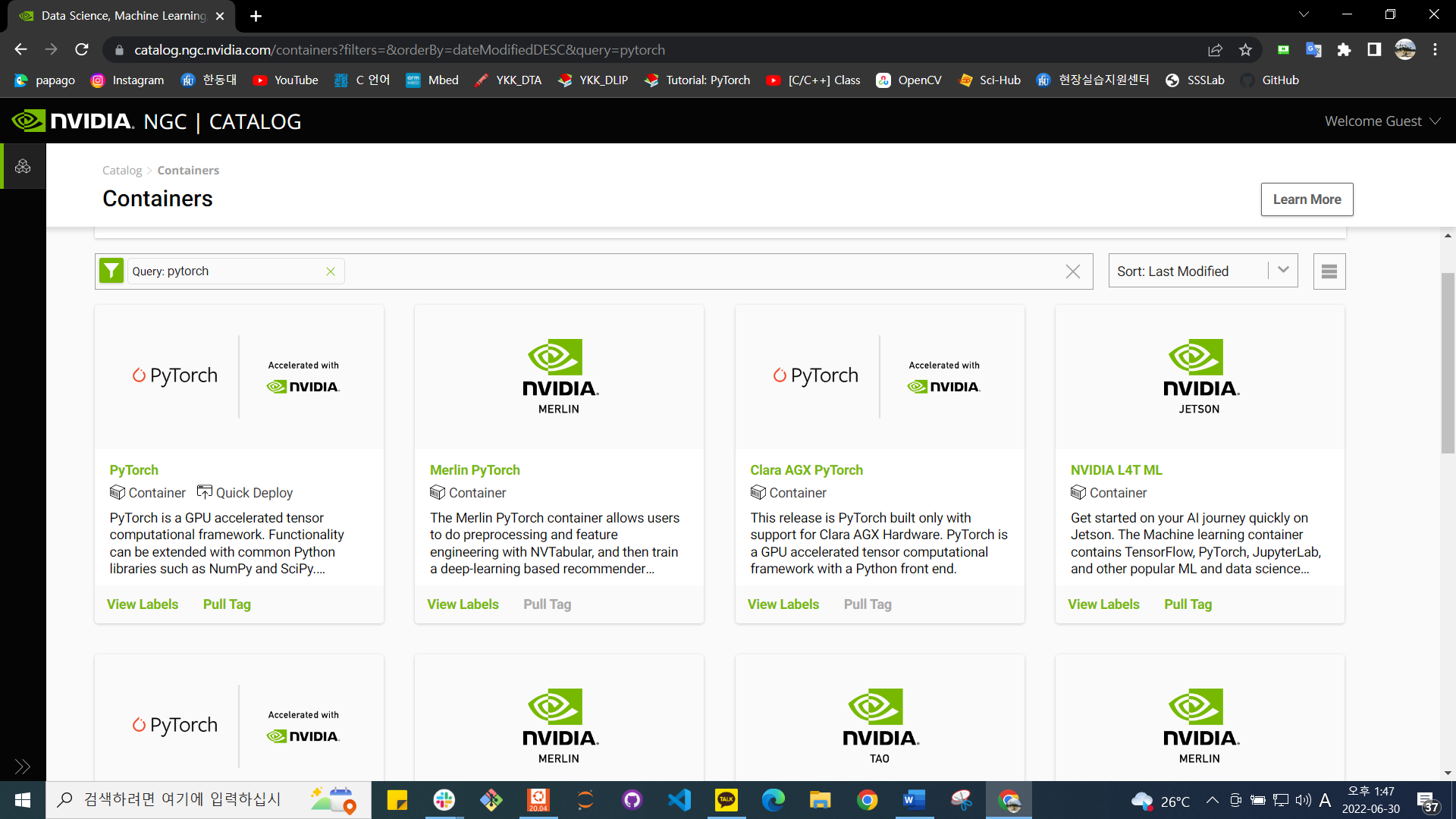


**[2]** (좌측 상단의) **Containers** 클릭하여, **Containers** 진입

텍스트이(가) 표시된 사진

자동 생성된 설명

**[3]** 검색창 **– ‘PyTorch’** 검색 **–** 제일 첫번째에 있는 **PyTorch** **Library** 클릭**!**

텍스트이(가) 표시된 사진

자동 생성된 설명

**[4] Tags** 클릭 **– ‘Pull Tags’**를 통해 **Images Code 복사**하기!

https://catalog.ngc.nvidia.com/orgs/nvidia/containers/tritonserver/tags

텍스트, 스크린샷, 모니터, 컴퓨터이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 모니터, 컴퓨터이(가) 표시된 사진

자동 생성된 설명

**[5]** 복사한 **Images Code**를 그대로 **Ubuntu Docker 명령창**에 입력!

단, Images Code에는 **PyTorch랑 Python어떤 버전이 사용되었는지 까지는 나와있지 않음.**

따라서, 버전 확인을 위해 NVIDIA Release Notes를 참고해야 한다.

**[6]** Google - **Ngc pytorch release notes** 검색 – 사이트 접속

가져오려는 Image가 **어떤 버전의 Library가 담긴 image인지** 알 수 있음.

https://docs.nvidia.com/deeplearning/frameworks/pytorch-release-notes/index.html

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

**(+)**

**Docker container의 포트와 호스트의 포트를 같게 만들어주는 방법**

nvidia-docker run --gpus='"device=6,7"' -it -v /media/hdd/sehyun:/Sehyun **-p 6008:6008** --name "sehyun" --shm-size 16G nvcr.io/nvidia/pytorch:21.12-py3 /bin/bash

**(+)**

**docker logs ‘컨테이너이름’** 컨테이너의 동작 로그 확인

**docker logs -f ‘컨테이너이름’** 컨테이너의 동작 로그 실시간으로 확인

**docker start ‘컨테이너이름’** 컨테이너 작동시키기

**(+)**

**ls -l | grep ^- | wc -l** 현재 위치에서의 파일 개수

**Visual Studio Code에 원격 가상 Server 연동하기**

**[0]** Remote Tool 모두 설치

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**[1]** Ctrl + Shift + P

**[2]** 작업 Command창에 ‘Remote-SSH: Connect to Host’ 입력

**[3]** ‘+ 새 호스트 추가’ 클릭

**[4] ssh** [**sehyun\_2022si@10.50.20.37**](mailto:sehyun_2022si@10.50.20.37)입력 (본인 계정)

**[5]** 비밀번호 입력

**[6]** 원격 탐색기에서 본인의 Container 지정

* 여기까지 하면, 원격 Server를 VS code에서 사용할 수 있게 되며,

파일 경로 및 복사 붙여넣기 코드 작성 및 실행 전부 VS code 상에서 진행이 가능!