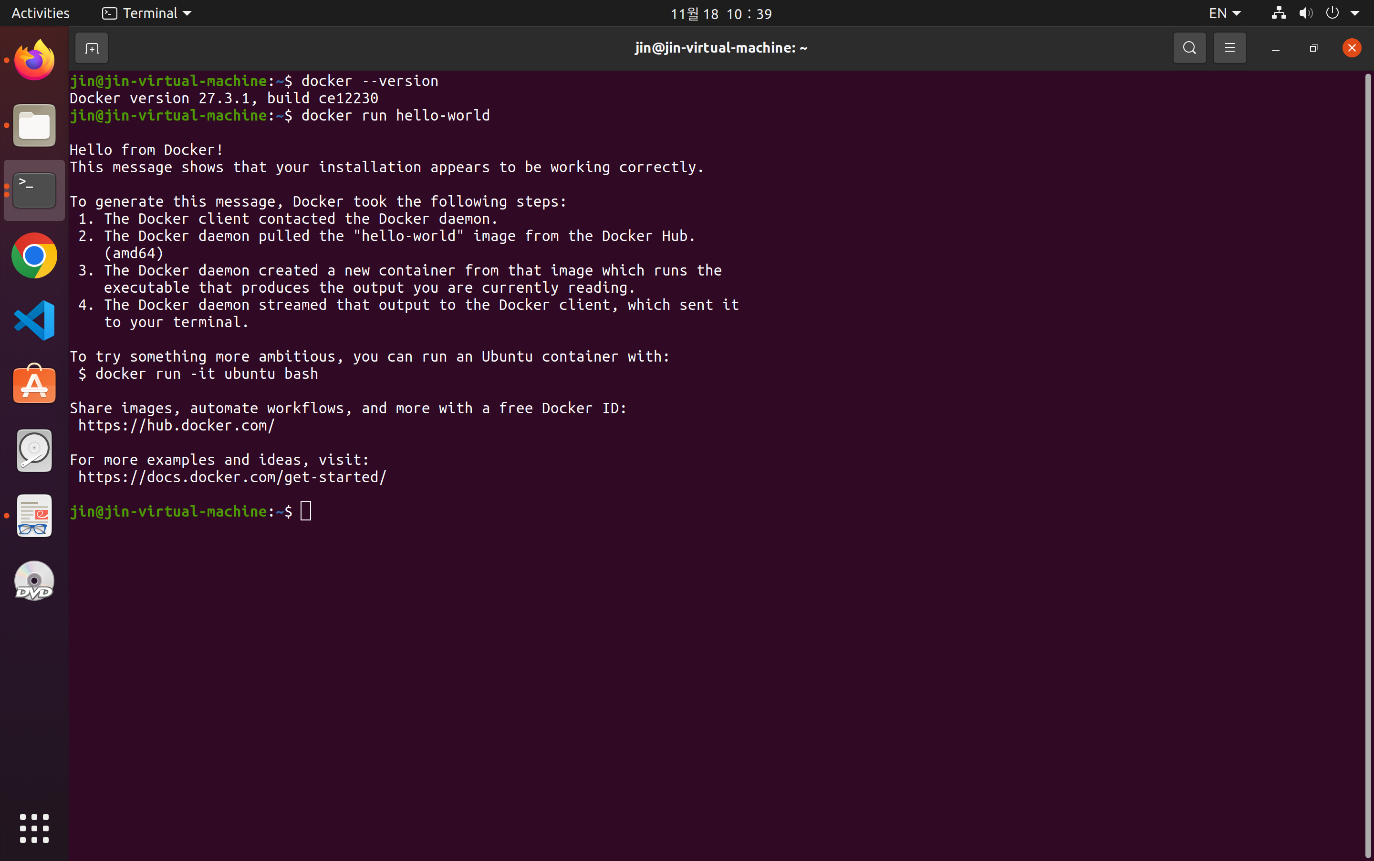
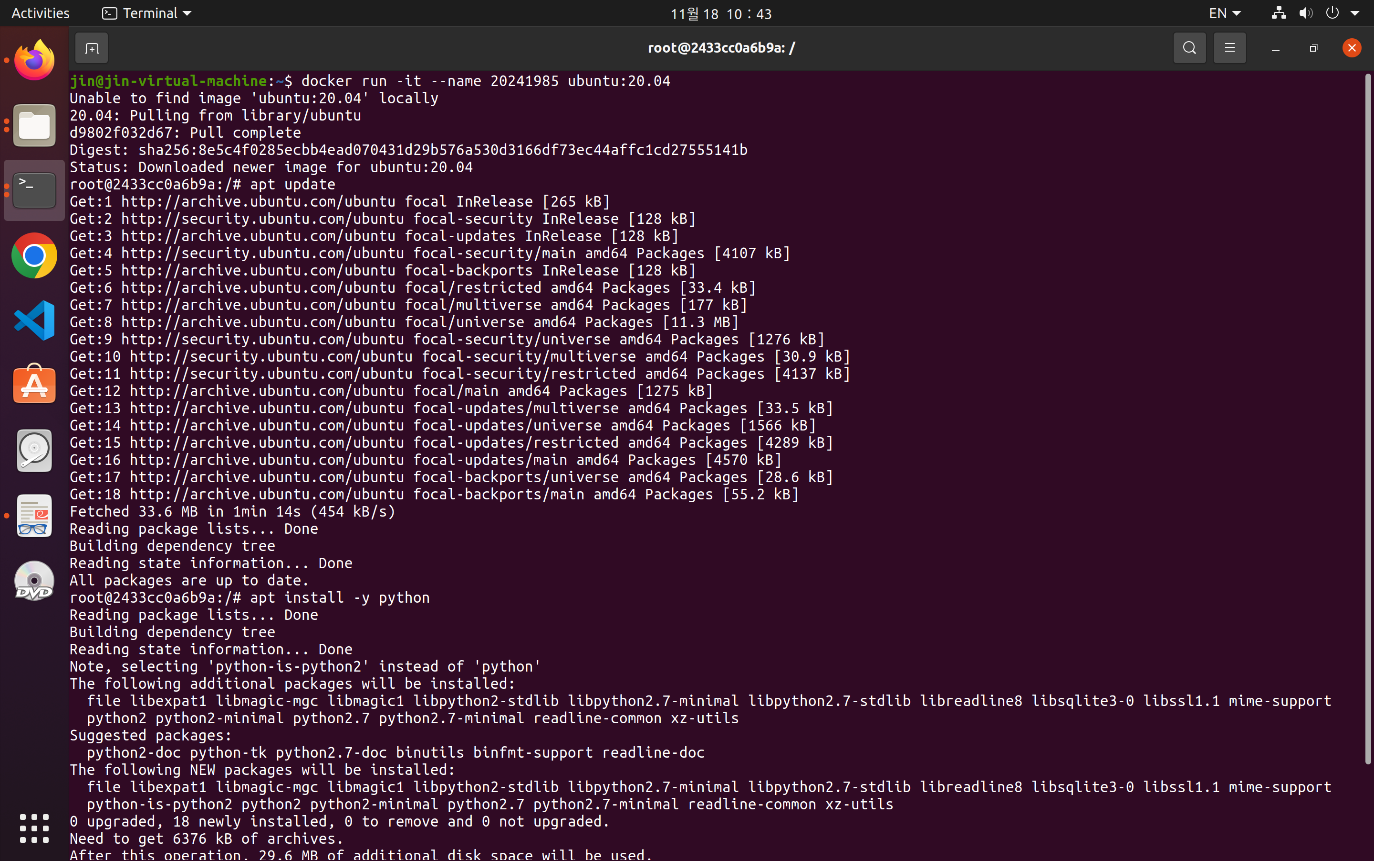
**도커 실습 보고서** 1조 20241985 조혁진, 20231958 김시현, 20243302 최연하

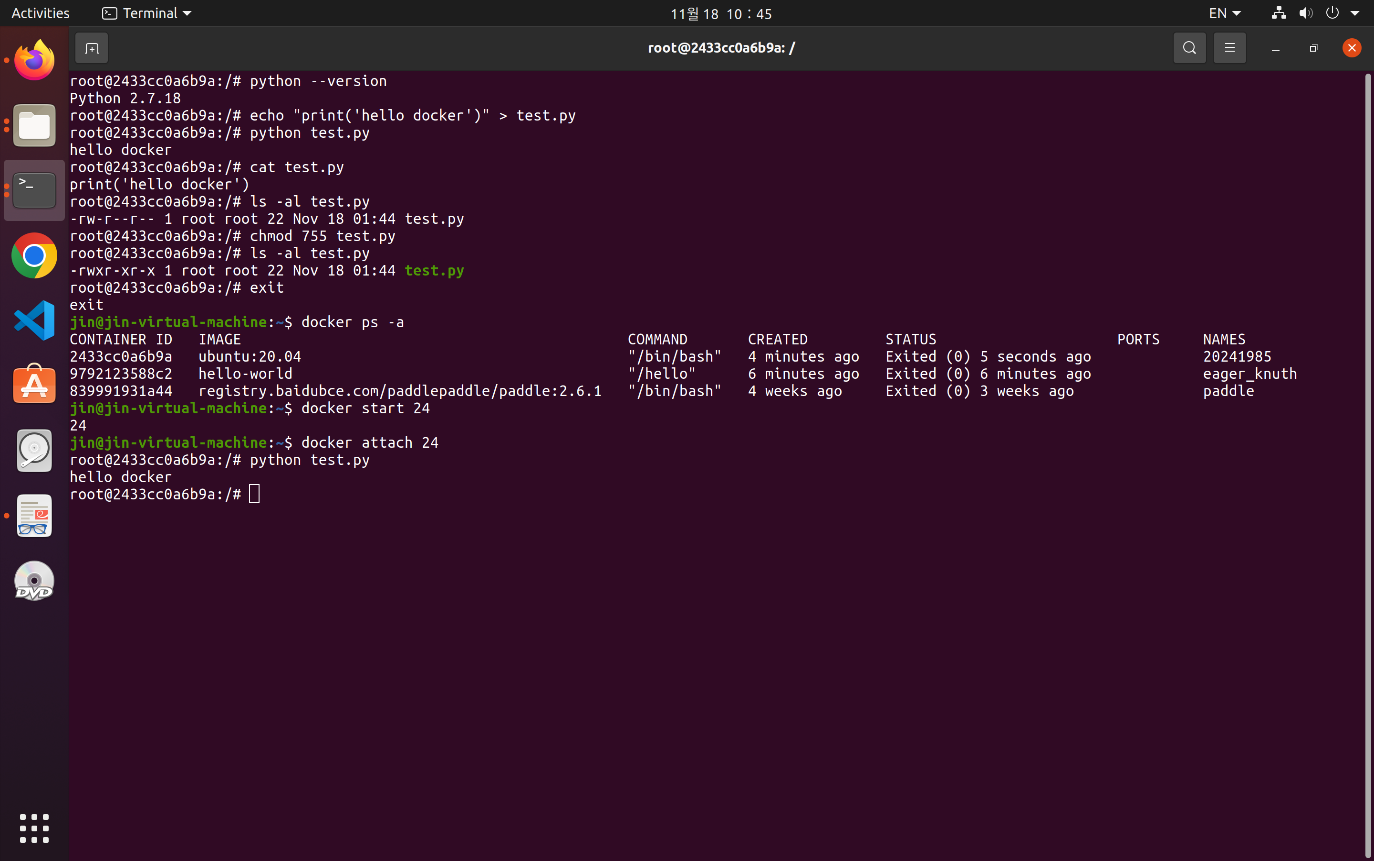
실습 1: docker --version을 통해서 도커의 버전을 확인해주고 docker run hello-world를 통해서 기능 테스트를 진행했다.



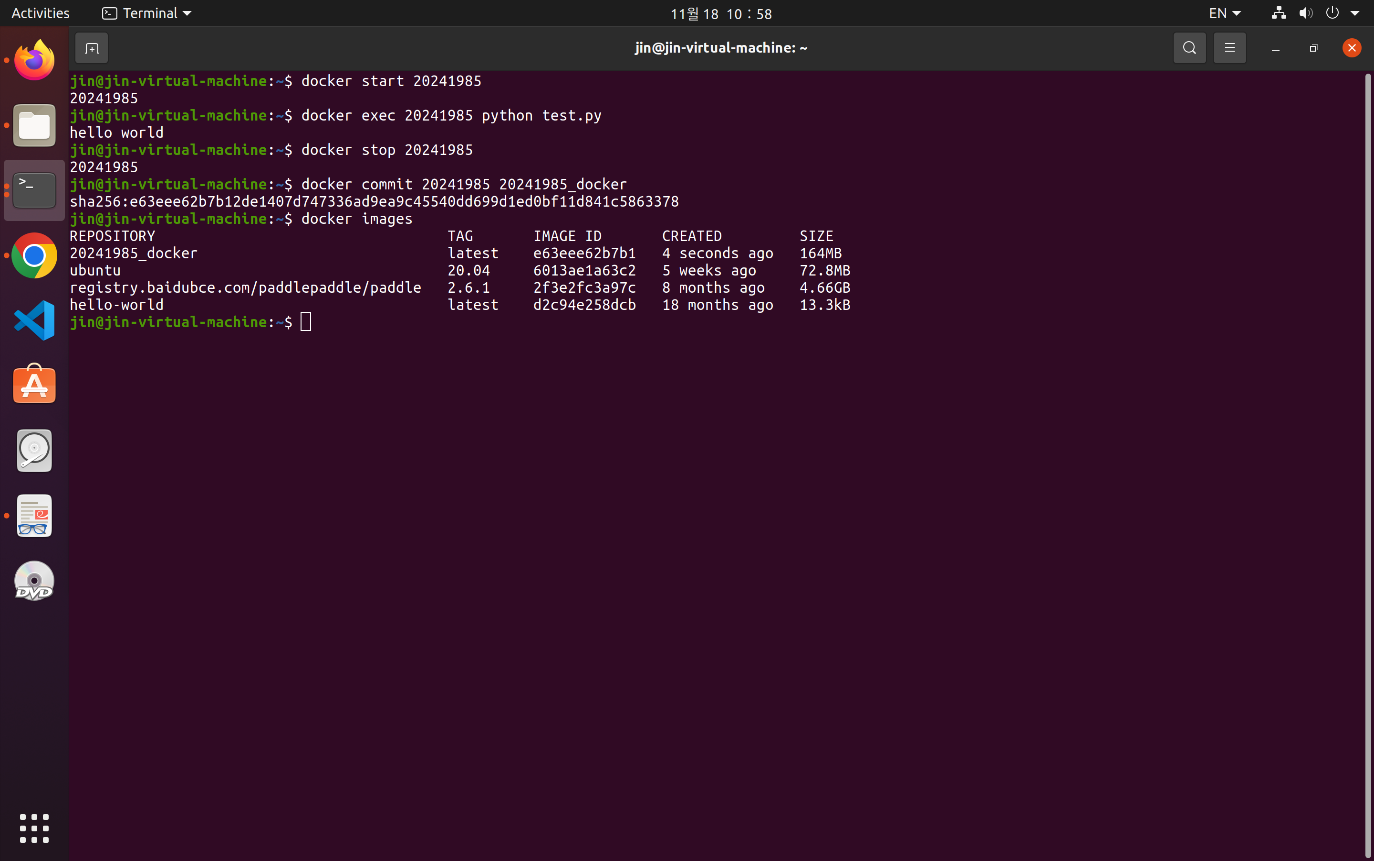
실습 2: docker run -it --name 20241985 ubuntu:20.04를 통해서 우분투 20.04 이미지를 실행하고 이름은 20241985로 지정했다. apt update를 통해서 업데이트를 진행했다.



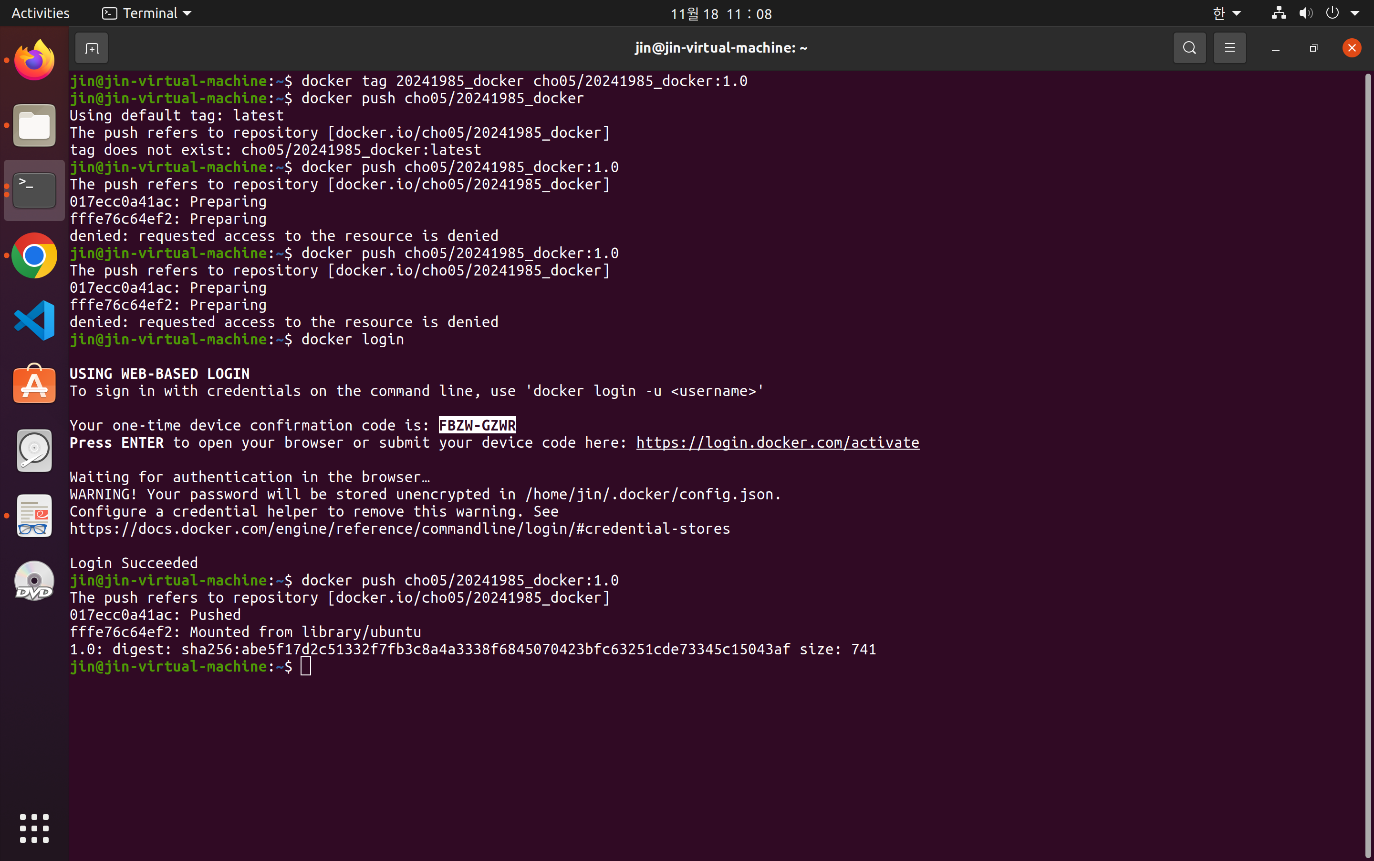
파이썬을 설치하고 test.py를 작성했다. 이후 실행도 해보고 권한도 부여하는 등 작업을 수행했다. exit를 통해서 나오면서 컨테이너를 종료하고 이후 docker start 24를 통해서 다시 실행하고 attach로 컨테이너로 진입했다.



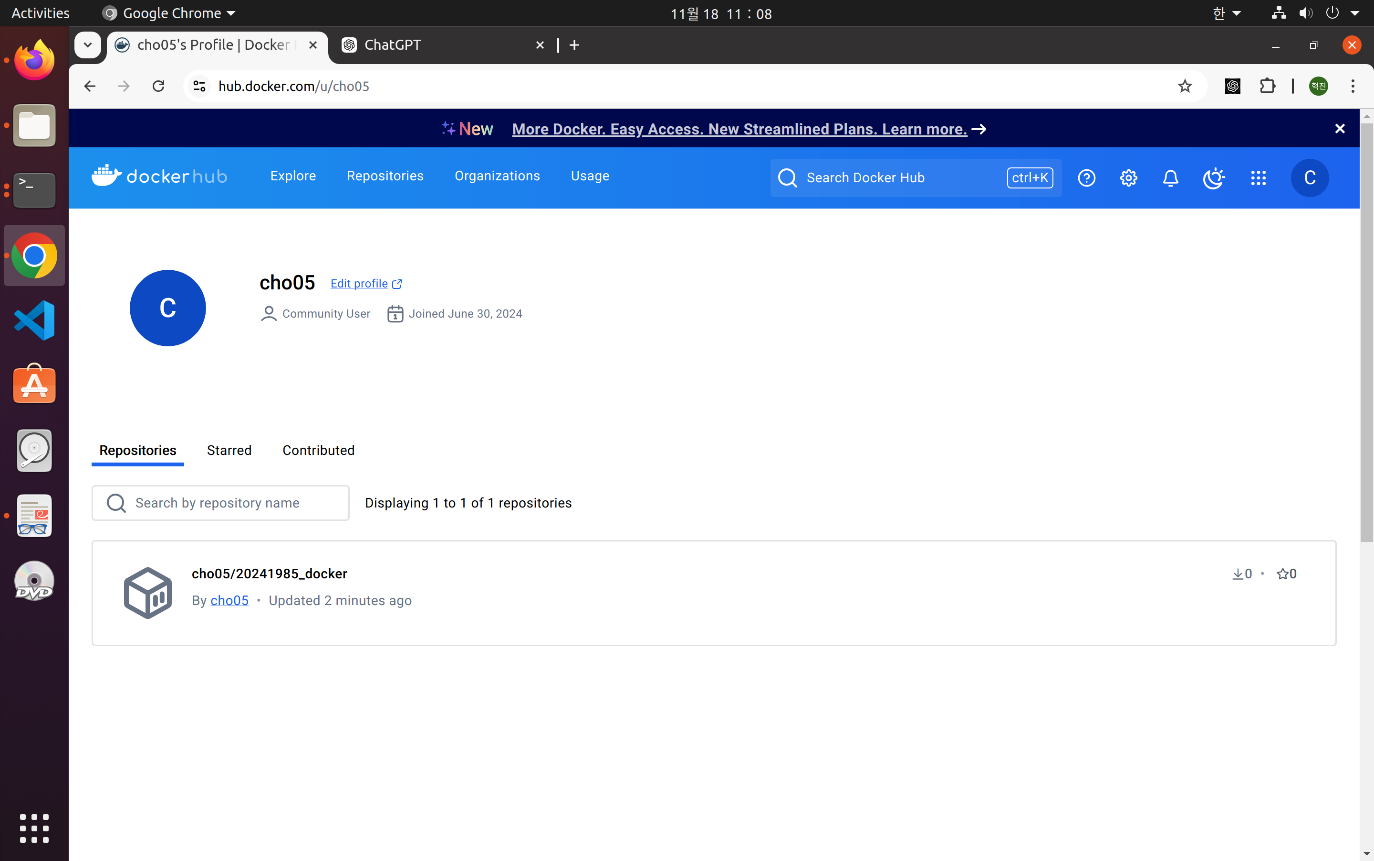
실습 3: start 명령어로 20241985 이름을 가진 컨테이너를 실행시키고 exec 명령어로 컨테이너에 있는 test.py를 실행시켰다. 컨테이너를 종료시킨 후, commit 명령어로 20241985 컨테이너를 20241985\_docker 이미지로 만들었다.



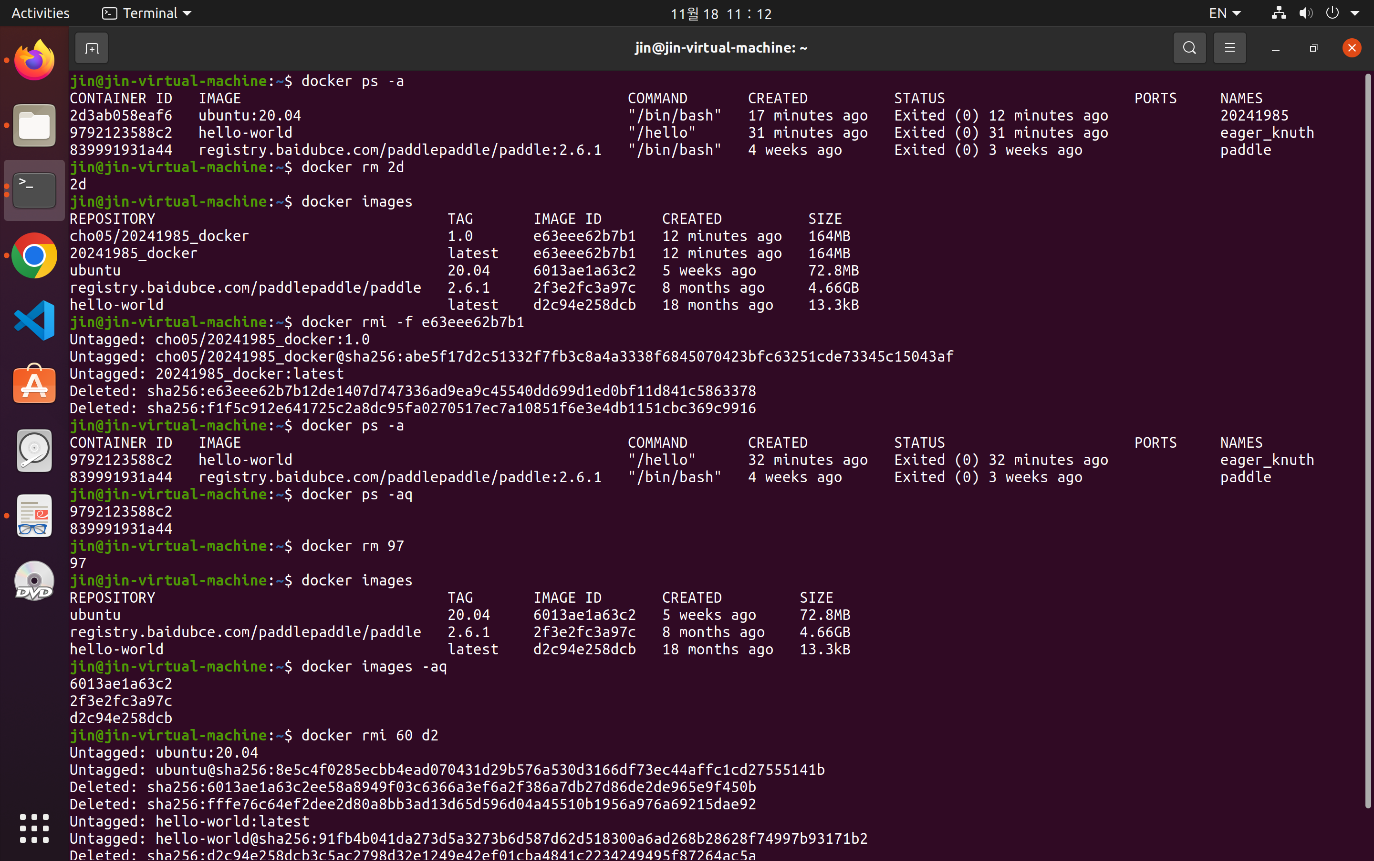
Docker Hub로 로컬 이미지를 푸시하기 위해서 tag 명령어로 태그한 뒤 push를 시행했지만 에러가 발생했다. 알고 보니, 로컬 터미널에서 docker login을 해야 했고 이를 수행한 뒤 다시 시도하여 성공했다.



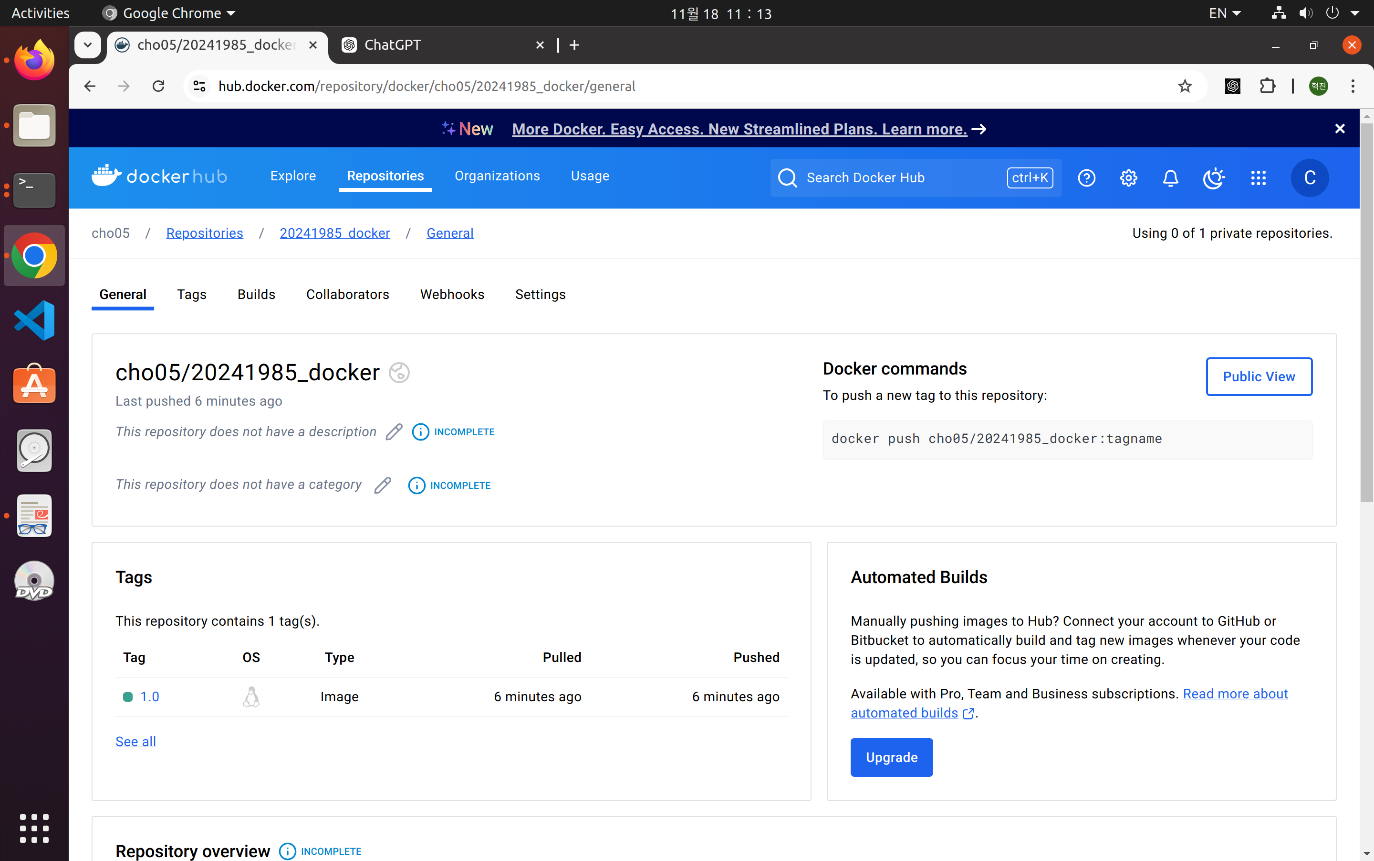
cho05 계정의 repositories에 푸시가 된 것을 확인할 수 있다.



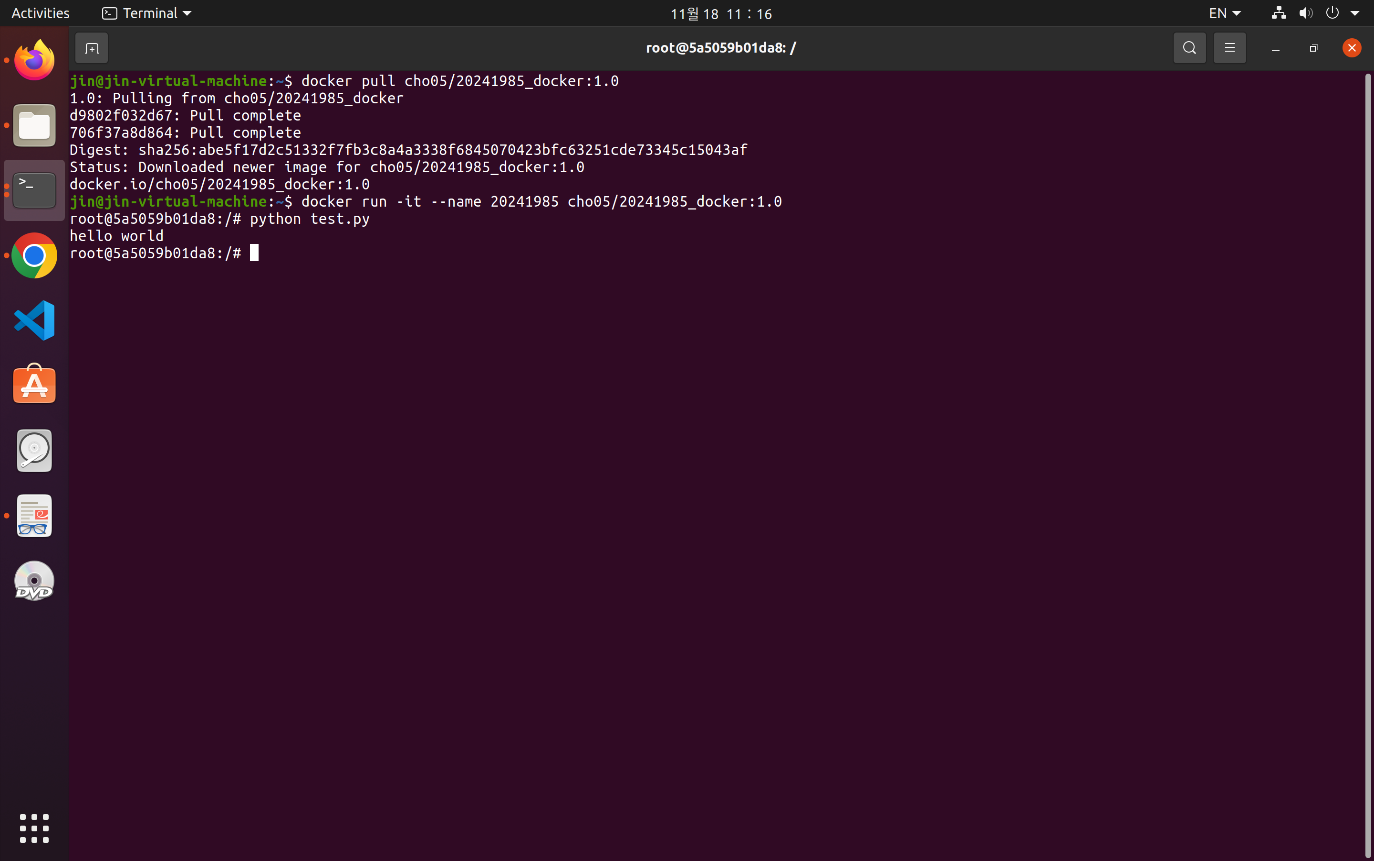
실습 4에서 pull을 하기 위해서 기존에 있던 로컬 컨테이너 및 이미지를 제거하는 사진이다. 컨테이너를 삭제하기 위해서 rm 이라는 명령어를 사용하고 이미지를 삭제하기 위해서는 rmi 명령어를 사용하여 제거를 진행했다.



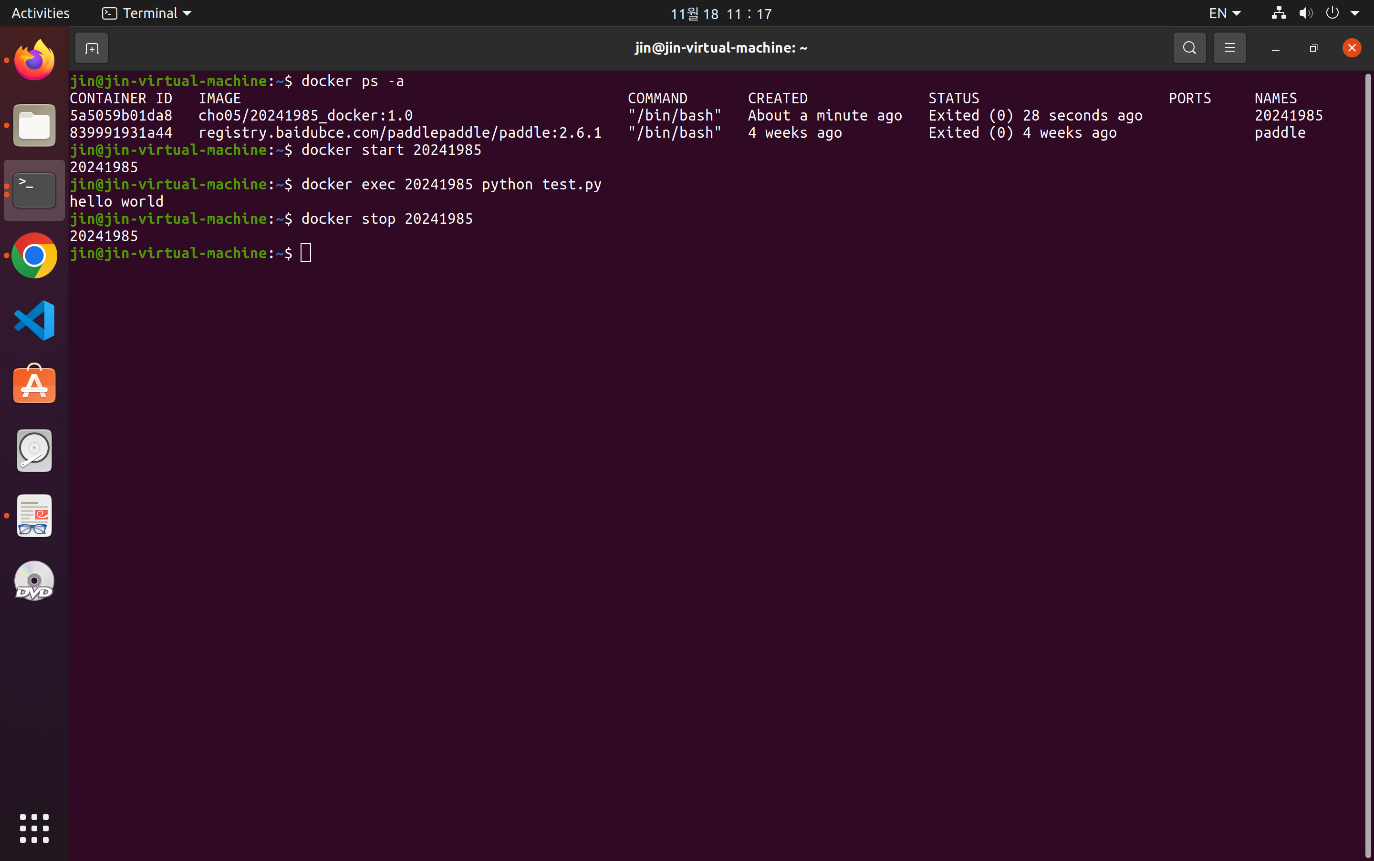
cho05 계정의 repositories에 들어가면 20241985\_docker 이미지가 Docker Hub에 잘 올라왔음을 확인할 수 있다.



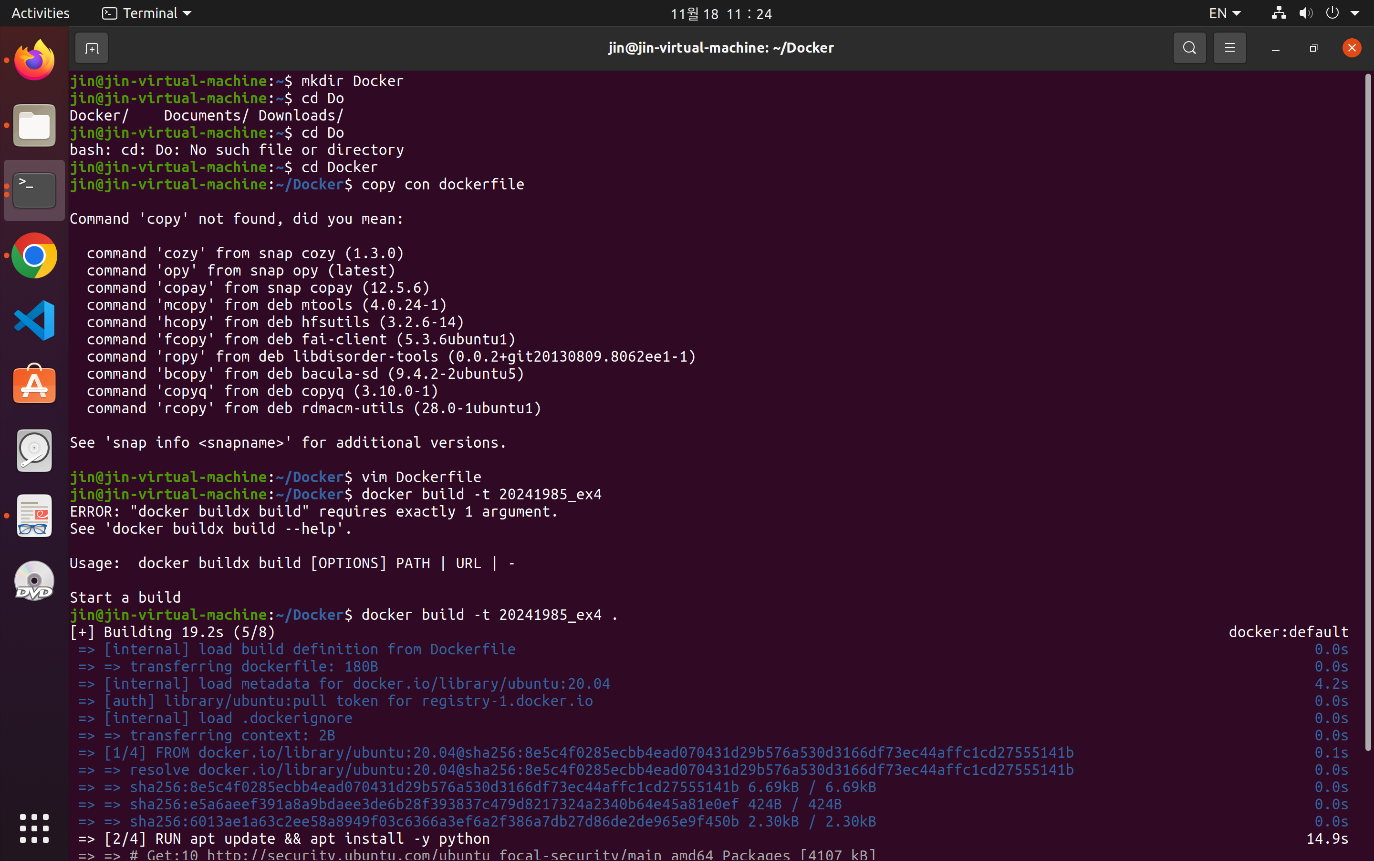
실습 4: pull 명령어를 통해서 cho05 계정에 있는 20241985\_docker:1.0 이미지를 가져왔다. 이미지를 잘 가져왔는지 확인하기 위해서 test.py 파일을 실행했고 결과가 잘 나왔다.



start 명령어로 20241985 이름을 가진 컨테이너를 실행시키고 exec 명령어로 컨테이너에 있는 test.py를 실행시켰다. 그 다음, stop 명령어로 컨테이너를 종료시켰다.



실습 5: Docker 폴더를 만들고 vim Dockerfile을 통해서 도커 파일을 작성했다. 이후 docker build를 통해서 이미지를 도커 파일을 이용하여 빌드했다.



빌드가 성공적으로 완료되었다. docker run 20241985\_ex4 명령어를 실행하여 빌드된 이미지를 기반으로 컨테이너를 실행했다. 그리고 hello world라는 출력이 표시되었다.

