**docker 기초 실습 (팀별)**

팀원: 권유진(20243261), 김지율(20243269), 김혜령(20243273)

**실습#1 : 도커 버전 확인 및 기능 테스트**

**텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명**

**설명:** *docker --version* 명령어를 통해 Docker 버전이 "27.3.1”임을 확인한다. 이후 *docker run hello-world* 명령어로 Docker의 테스트 이미지를 실행하여 설치가 올바르게 되었는지 확인한다. 출력된 *‘Hello from Doker!’* 란 결과값을 통해 Docker가 성공적으로 설치 및 작동 중임을 확인할 수 있다.

**실습#2 : 도커 환경에서의 코드 작성**

**텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명**

**설명:** 환경 구축 후 python으로 코드를 작성하고 Docker 컨테이너의 관리와 내부에서 프로그램을 실행한다. 먼저 *docker ps -a* 명령어를 이용해 현재 존재하는 모든 컨테이너의 ID, 이미지 이름, 실행 명령어, 생성 시간, 상태 등을 확인한다. 이후 *docker start 65e600658f16* 명령어를 통해 지정한 컨테이너 ID로 Ubuntu 컨테이너를 시작한다. *docker attach 65e600658f16* 명령어를 입력하여 해당 컨테이너의 터미널 세션에 연결하고 컨테이너 내부에서 Python 스크립트 test.py 실행하여 “hello docker"라는 출력을 확인한다.

**실습#3 : 도커 환경을 배포**

**텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명**

**텍스트, 스크린샷, 폰트, 블랙이(가) 표시된 사진

자동 생성된 설명**

**설명:** Docker Hub를 통해 다른 사용자와 Docker 이미지를 쉽게 공유하도록 배포한다. *docker tag 20243261\_ex5 yujin818/20243261\_ex5:1.0* 명령어를 이용하여 *yujin818*가 사용자명인 Docker Hub에서 *20243261\_ex5*의 이미지에 *yujin818/20243261\_ex5:1.0* 태그를 추가한다. 이후 *docker push yujin818/20243261\_ex5:1.0* 명령어를 입력해 태그가 붙은 이미지를 Docker Hub 저장소로 업로드한다. 이미지를 통해 Docker Hub에서 사용자가 Docker Hub에 푸시한 이미지가 성공적으로 저장소에 등록되었음을 알 수 있다.

**실습#4 : 배포한 환경을 검증**

**텍스트, 영수증, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명**

**설명:** Docker 컨테이너를 실행하고 Python 스크립트를 실행한다.

앞서서 *docker pull yujin818/20243261\_ex5:1.0* 명령을 실행하여 docker hub에서 이미지를 가져온다.

이후 *docker run -it --name 20243261 yujin818/20243261\_ex5:1.0* 명령을 실행하여 컨테이너 내부에서 작업을 수행한다. -*it* 로 대화형(interactive) 모드로 실행하고 *--name 20243261*으로 실행되는 컨테이너 이름을 20243261로 지정한다. 그리고 *yujin818/20243261\_ex5:1.0* 은 Docker Hub에서 해당 태그가 붙은 이미지를 가져와 실행한다.

이때 Python 스크립트 실행을 해야한다. 컨테이너 내부에서 *python test.py* 명령어를 실행하면 출력 결과 *hello docker* 가 나온다. 컨테이너는 *exit* 명령어로 종료한다.

이 과정은 Docker Hub에서 이미지를 가져와 실행한 뒤, 컨테이너 내부에서 작업을 수행하고 종료하는 과정을 설명하고, 컨테이너 실행과 테스트가 정상적으로 완료되었음을 보여준다.

**실습#5 : 도커 파일 작성 실습**

**텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명**

**설명**: Docker 이미지를 생성하고 실행한다. 먼저, Dockerfile을 사용하여 *ubuntu:20.04* 이미지를 기반으로 새로운 Docker 이미지를 빌드한다. 빌드 과정에서는 Python을 설치하고, *test.py*라는 파일에 간단한 Python 코드*(print('hello docker'))*를 작성한 후, 해당 파일에 실행 권한을 부여한다. 빌드가 완료된 이미지는 *20243261\_ex5* 라는 태그로 저장된다. 그 후, 생성된 이미지를 실행하기 위해 *docker run 20243261\_ex5* 명령을 사용하며, 이 과정에서 컨테이너 내부의 *test.py*가 실행되어 *hello docker*라는 출력이 표시된다. 이로써 Docker 이미지 생성 및 실행이 정상적으로 작동함을 확인할 수 있다.