BÁO CÁO KẾT QUẢ HỌC TẬP: DOCKER TUTORIAL FOR BEGINNERS

1. Tổng quan những nội dung đã học

Thông qua chuỗi 12 bài học từ khóa "Docker Tutorial For Beginners" trên Simplilearn, em đã nắm bắt được các kiến thức cốt lõi về công nghệ Containerization và cách sử dụng Docker. Các kiến thức chính bao gồm:

A. Khái niệm cơ bản và Kiến trúc (Lessons 1, 4, 5)

- **Docker là gì:** Là một nền tảng mã nguồn mở giúp tự động hóa việc triển khai các ứng dụng bên trong các "container" phần mềm. Nó cho phép đóng gói ứng dụng cùng với tất cả các thư viện và dependencies cần thiết để chạy được ở mọi nơi.
- **Docker vs Virtual Machine (Máy ảo):** Đây là kiến thức quan trọng.
    - *VM:* Ảo hóa phần cứng, mỗi VM cần một hệ điều hành (Guest OS) đầy đủ, gây tốn tài nguyên và khởi động chậm.
    - *Docker:* Ảo hóa ở cấp hệ điều hành (OS-level virtualization), các container chia sẻ kernel của Host OS. Điều này giúp Docker nhẹ hơn, khởi động nhanh hơn và hiệu quả hơn về tài nguyên.
- **Docker Container:** Là một instance (phiên bản đang chạy) của Docker Image. Nó độc lập, an toàn và bị cô lập với môi trường bên ngoài.

B. Các thành phần cốt lõi (Lessons 6, 10)

- **Docker Image:** Là một template chỉ đọc (read-only) chứa source code, libraries, dependencies, tools,... cần thiết để ứng dụng chạy.
- **Dockerfile:** Là một file văn bản chứa tập hợp các lệnh để Docker build ra một Docker Image. Em đã học các từ khóa quan trọng như FROM, RUN, COPY, CMD, EXPOSE.
- **Docker Commands:** Nắm vững các lệnh CLI cơ bản để quản lý vòng đời container: docker run, docker ps, docker stop, docker rm, docker rmi, docker build.

C. Quản lý và Điều phối (Lessons 7, 8, 9)

- **Docker Networking:** Hiểu cách các container giao tiếp với nhau và với thế giới bên ngoài thông qua các driver mạng như Bridge (mặc định), Host, và None.
- **Docker Compose:** Công cụ giúp định nghĩa và chạy các ứng dụng multi-container (nhiều container). Thay vì chạy từng lệnh lẻ tẻ, ta sử dụng file docker-compose.yml để khởi tạo toàn bộ stack ứng dụng chỉ với một lệnh docker-compose up.

- **Docker Swarm:** Tìm hiểu khái niệm về clustering và orchestration, cho phép quản lý một cụm các Docker engines như một hệ thống thống nhất (tuy nhiên trong thực tế lab MLOps hiện tại em tập trung vào Docker Desktop trên máy đơn).

2. Các bước thực hành

Dưới đây là quy trình em đã thực hiện trên môi trường **Docker Desktop** (Windows):

Bước 1: Cài đặt và Kiểm tra môi trường

- Tải và cài đặt Docker Desktop cho Windows.
- Kích hoạt WSL 2 (Windows Subsystem for Linux) để tối ưu hóa hiệu suất Docker trên Windows.
- Kiểm tra phiên bản đã cài đặt bằng terminal.

Lệnh thực hiện:

docker --version

docker run hello-world

```
PS E:\docker\TimHieuVeDocker> docker --version
Docker version 28.3.2, build 578ccf6
PS E:\docker\TimHieuVeDocker> docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Bước 2: Viết Dockerfile cho một ứng dụng Python đơn giản

Em đã tạo một thư mục dự án và viết file Dockerfile để đóng gói một script Python (mô phỏng một tác vụ MLOps đơn giản).

Nội dung Dockerfile:

Dockerfile

```
# Sử dụng Python base image
FROM python:3.9-slim

# Thiết lập thư mục làm việc
WORKDIR /app

# Copy file code vào container
COPY main.py .

# Chạy file python
CMD ["python", "main.py"]
```

Bước 3: Build Image từ Dockerfile

Sử dụng lệnh docker build để tạo image từ Dockerfile trên.

Lệnh thực hiện:

Bash

```
docker build -t my-python-app:v1 .
```

```
PS E:\docker\TimHieuVeDocker\docker-mlops-lab> docker build -t my-python-app:v
1 .
[+] Building 9.1s (8/8) FINISHED                          docker:desktop-linux
 => [internal] load build definition from Dockerfile                      0.0s
 => => transferring dockerfile: 326B                                      0.0s
 => [internal] load metadata for docker.io/library/python:3.9-slim        1.2s
 => [internal] load .dockerignore                                         0.0s
 => => transferring context: 2B                                           0.0s
 => [1/3] FROM docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd3   7.4s
 => => resolve docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd3   0.0s
 => => sha256:ea56f685404adf81680322f152d2cfec62115b30dda48 251B / 251B   1.0s
 => => sha256:fc74430849022d13b0d44b8969a953f842f59c6 13.88MB / 13.88MB   6.3s
 => => sha256:38513bd7256313495cdd83b3b0915a633cfa475 29.78MB / 29.78MB   6.0s
 => => sha256:b3ec39b36ae8c03a3e09854de4ec4aa08381dfed8 1.29MB / 1.29MB   3.6s
 => => extracting sha256:38513bd7256313495cdd83b3b0915a633cfa475dc2a070   0.7s
 => => extracting sha256:b3ec39b36ae8c03a3e09854de4ec4aa08381dfed84a9da   0.1s
 => => extracting sha256:fc74430849022d13b0d44b8969a953f842f59c6e9d1a0c   0.4s
 => => exporting config sha256:2352ceeb0208bada2183c3f3fe913b875498ae1e   0.0s
 => => exporting attestation manifest sha256:5dfaef52f1ab77618e2c27139c   0.0s
 => => exporting manifest list sha256:ec55f9fe9e008ff10f48f7fb69fe2386b   0.0s
 => => naming to docker.io/library/my-python-app:v1                       0.0s
 => => unpacking to docker.io/library/my-python-app:v1                    0.0s
```

Bước 4: Run Container và Kiểm tra

Chạy container từ image vừa build và kiểm tra trạng thái hoạt động.

Lệnh thực hiện:

Bash

docker run my-python-app:v1

```
PS E:\docker\TimHieuVeDocker\docker-mlops-lab> docker run my-python-app:v1
=== MLOps Lab: Docker Training ===
Step 1: Load Data... Done
Step 2: Training Model... Done
Step 3: Model Accuracy: 80%
Hello from Docker Container!
```

Bước 5: Sử dụng Docker Compose (Tùy chọn)

Thử nghiệm tạo file docker-compose.yml để quản lý cấu hình chạy thay vì gõ lệnh dài dòng.

```yaml
version: '3'
services:
  my-mlops-app:
    image: my-python-app:v1
```

```
    container_name: mlops-container-test
```

```
PS E:\docker\TimHieuVeDocker\docker-mlops-lab> docker compose up
time="2025-11-22T10:51:11+07:00" level=warning msg="E:\\docker\\TimHieuVeDocke
r\\docker-mlops-lab\\docker-compose.yml: the attribute `version` is obsolete,
it will be ignored, please remove it to avoid potential confusion"
[+] Running 2/2
 ✓ Network docker-mlops-lab_default  Created                              0.0s
 ✓ Container mlops-container-test    Created                              0.1s
Attaching to mlops-container-test
mlops-container-test  | === MLOps Lab: Docker Training ===
mlops-container-test  | Step 1: Load Data... Done
mlops-container-test  | Step 2: Training Model... Done
mlops-container-test  | Step 3: Model Accuracy: 80%
mlops-container-test  | Hello from Docker Container!
mlops-container-test exited with code 0
PS E:\docker\TimHieuVeDocker\docker-mlops-lab>

v View in Docker Desktop   o View Config   w Enable Watch
```

3. Khó khăn và Lỗi gặp phải

Trong quá trình học và thực hành, em đã gặp một số vấn đề sau:

1. **Vấn đề về WSL 2:**
    - *Lỗi:* Khi mới cài đặt Docker Desktop, Docker Engine không khởi động được do chưa update WSL kernel.
    - *Cách khắc phục:* Em đã tải gói update WSL 2 từ trang chủ Microsoft và chạy lệnh wsl --set-default-version 2 trong PowerShell.
2. **Xung đột Port (Cổng):**
    - *Lỗi:* Khi thử chạy một container Nginx web server map vào port 80 (-p 80:80), báo lỗi port đã được sử dụng.
    - *Nguyên nhân:* Port 80 trên Windows thường bị chiếm bởi System process hoặc IIS.
    - *Cách khắc phục:* Đổi sang map vào port khác (ví dụ: -p 8080:80).

4. Định hướng tìm hiểu thêm

Từ nền tảng kiến thức này, để phục vụ tốt hơn cho môn học **MLOps**, em dự định tìm hiểu sâu hơn về 2 chủ đề sau:

1. **Docker for GPU (NVIDIA Container Toolkit):**
    - Trong MLOps, việc huấn luyện mô hình Deep Learning cần GPU. Em muốn tìm hiểu cách cấu hình để Docker container có thể truy cập và sử dụng GPU của máy host (thông qua cờ --gpus all), điều này rất quan trọng cho các thư viện như TensorFlow hay PyTorch.
2. **Kubernetes (K8s):**
    - Mặc dù đã học về Docker Swarm, nhưng em nhận thấy Kubernetes mới là chuẩn công nghiệp hiện nay cho việc điều phối container ở quy mô lớn. Em muốn tìm hiểu cách deploy các mô hình Machine Learning lên cụm K8s để đảm bảo khả năng mở rộng (scalability) và tính sẵn sàng cao (high availability).

Tài liệu tham khảo

1. Simplilearn - Docker Tutorial For Beginners Playlist.
2. Docker Official Documentation (docs.docker.com).