HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA AN TOÀN THÔNG TIN

BÁO CÁO THỰC TẬP TỐT NGHIỆP

ĐỀ TÀI: TÌM HIỂU VÀ THỰC HÀNH VỀ DOCKER

Giảng viên hướng dẫn : TS. Nguyễn Ngọc Điệp
Sinh viên thực hiện : Nguyễn Văn Quyến
Mã Sinh viên : B21DCAT161
Lớp : D21DCQT01-B

HÀ NỘI 7-2025

MỤC LỤC

DANH MỤC HÌNH VẼ **2**
CHƯƠNG 1: TỔNG QUAN VỀ DOCKER **3**
1. Docker là gì? 3
2. Những lợi ích và hạn chế của Docker 4
2.1 Lợi ích 4
2.2 Hạn chế 4
3. Các thành phần Docker 4
3.1 Docker Engine 4
3.2 Distribution tools 5
3.3 Orchestration tools 5
4. Kiến trúc của Docker 6
4.1 Docker daemon 6
4.2 Docker registry 6
4.3 Docker client 7
4.4 Docker Desktop 7
4.5 Docker object 7
CHƯƠNG 2: THỰC HÀNH MỘT SỐ LỆNH CƠ BẢN **8**
1. Cài đặt Docker Destokp trên Windows 11 8
1.1 Cách bật WSL2 8
1.2 Cách cài đặt Docker 10
2. Thực hành một số lệnh cơ bản trên Docker 12
2.1 Thực hành theo hướng dẫn trên Docker Desktop 12
2.2 Thực hành với một số ví dụ cơ bản 15

DANH MỤC HÌNH VẼ

Hình 1: Docker là gì? 3
Hình 2: Docker Engine 5
Hình 3: Kiến trúc của Docker 6
Hình 4: Bật các tính năng trong Windows Features 8
Hình 5: Các distro có thể cài đặt 9
Hình 6: Cài đặt distro Ubuntu 9
Hình 7: Chọn "Use WSL 2 instead of Hyper-V" 10
Hình 8: nhấn "Close and log out" 11
Hình 9: Nhấn "Accept" 11
Hình 10: Cài đặt thành công Docker Desktop 12
Hình 11: Clone repository welcome-to-docker 12
Hình 12: Dockerfile của thư mục welcome-to-docker 13
Hình 13: Build thành công 13
Hình 14: Run container 14
Hình 15: Thực hành thành công 14
Hình 16: Sử dụng lệnh docker search 15
Hình 17: Sử dụng lệnh docker pull để tải mysql 15
Hình 18: file ex01 15
Hình 19: Dockerfile 16
Hình 20: Build thành công 16
Hình 21: Chạy thành công container 16
Hình 22: file ex02 17
Hình 23: Thêm dòng cài đặt flask vào Dockerfile 17
Hình 24: Build thành công 18
Hình 25: Chạy container từ image vừa build 18
Hình 26: Thực hành thành công 18
Hình 27: file ex03 19
Hình 28: Dockerfile 19
Hình 29: file setup.txt chứa các thư viện cần cài đặt 20
Hình 30: Build thành công 20
Hình 31: Danh sách các mạng trong Docker network 20
Hình 32: Tạo thêm mạng mynet 20
Hình 33: Chạy và cấu hình mysql 21
Hình 34: Chạy container từ image ex03 21
Hình 35: Thực hành thành công 21

CHƯƠNG 1: TỔNG QUAN VỀ DOCKER

1. Docker là gì?

Docker là một ứng dụng mã nguồn mở cho phép đóng gói các ứng dụng, các phần mềm phụ thuộc lẫn nhau vào trong cùng một container. Container này sau đó có thể mang đi triển khai trên bất kỳ một hệ thống Linux phổ biến nào. Các container này hoàn toàn độc lập với các container khác.

- Với Docker, người dùng có thể đóng gói mọi ứng dụng ví dụ như webapp, backend, MySQL, BigData…thành các containers và có thể chạy ở "hầu hết" các môi trường như Linux, Mac, Window…
- Docker Containers có một API cho phép quản trị các container từ bên ngoài. Giúp cho người dùng có thể dễ dàng quản lí, thay đổi, chỉnh sửa các container.
- Hầu hết các ứng dụng Linux có thể chạy với Docker Containers.

Hình 1: Docker là gì?

2. Những lợi ích và hạn chế của Docker

2.1 Lợi ích

- **Sử dụng ít tài nguyên:** Thay vì phải ảo hóa toàn bộ hệ điều hành thì chỉ cần build và chạy các container độc lập sử dụng chung kernel duy nhất.
- **Tính đóng gói và di động:** Tất cả các gói dependencies cần thiết đều được đóng gói vừa đủ trong container. Và sau đó có thể mang đi triển khai trên các server khác.
- **Cô lập tài nguyên:** Server không biết ở trong container chạy gì và container cũng không cần biết nó là CentOs hay Ubuntu. Các container độc lập với nhau và có thể giao tiếp với nhau bằng một interface
- **Hỗ trợ phát triển và quản lý ứng dụng nhanh:** Đối với Dev, sử dụng docker giúp họ giảm thiểu thời gian setup môi trường, đóng gói được các môi trường giống nhau từ Dev - Staging - Production
- **Mã nguồn mở:** Cộng đồng support lớn, các tính năng mới được release liên tục.

2.2 Hạn chế

- Docker base trên Linux 64bit và các tính năng cgroup, namespaces. Vì thế Linux 32bit hoặc môi trường Window không thể chạy được docker (đối với phiên bản CE).
- Sử dụng container tức là người dùng sử dụng chung kernel của hệ điều hành. Trong trường hợp người dùng download image có sẵn và trong đó có một số công cụ có thể kiểm soát được kernel thì server có thể bị mất kiểm soát hoàn toàn.
- Các tiến trình chạy container một khi bị stop thì sẽ mất hoàn toàn dữ liệu nếu không được mount hoặc backup. Điều này có thể sẽ gây ra một số bất tiện…

3. Các thành phần Docker

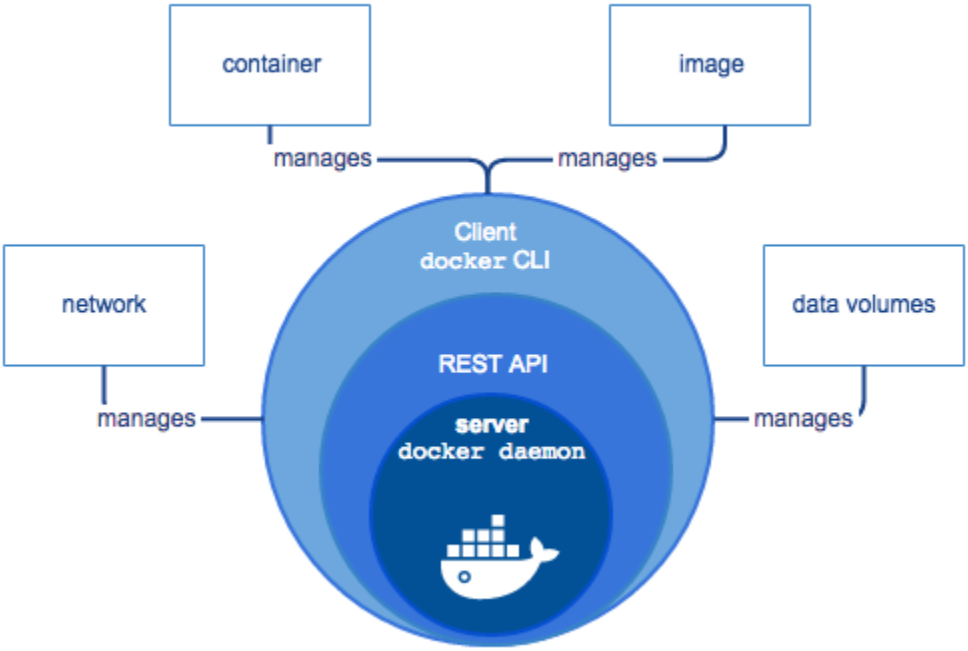
3.1 Docker Engine

Docker Engine là một ứng dụng client-server. Có hai phiên bản Docker Engine phổ biến là:

- Docker Community Edition (CE): Là phiên bản miễn phí và chủ yếu dựa vào các sản phẩm nguồn mở khác.
- Docker Enterprise: Khi sử dụng phiên bản này người dùng sẽ nhận được sự support của nhà phát hành, có thêm các tính năng quản lý và security.

Các thành phần chính của Docker Engine gồm có:

- Server hay còn được gọi là docker daemon (dockerd): chịu trách nhiệm tạo, quản lý các Docker objects như images, containers, networks, volume.
- REST API: docker daemon cung cấp các api cho Client sử dụng để thao tác với Docker
- Client là thành phần đầu cuối cung cấp một tập hợp các câu lệnh sử dụng api để người dùng thao tác với Docker. (Ví dụ: *docker images*, *docker ps*, *docker rmi image, ...*)



Hình 2: Docker Engine

3.2 Distribution tools

Là các công cụ phân tán giúp người dùng lưu trữ và quản lý các Docker Images như*: Docker Registry, Docker Trusted Registry, Docker Hub*.

Docker Hub là một công cụ phần mềm như một dịch vụ cho phép người dùng public hay private các images của họ. Dịch vụ cung cấp hơn 100.000 ứng dụng có sẵn công khai, cũng như các cơ quan đăng ký container công cộng và tư nhân

3.3 Orchestration tools

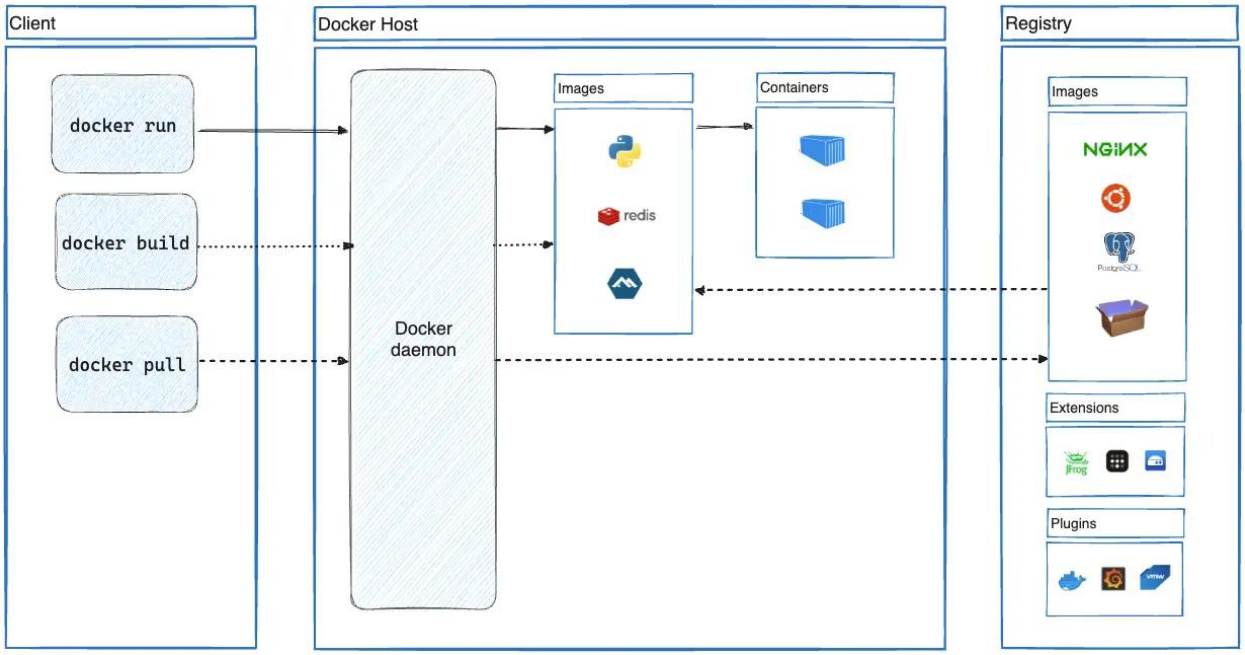
- **Docker Machine**: Machine tạo Docker Engine trên laptop của người dùng hoặc trên bất cứ dịch vụ cloud phổ biến nào như AWS, Azure, Google Cloud, Softlayer hoặc trên hệ thống data center như VMware, OpenStack. Docker Machine sẽ tạo

các máy ảo và cài Docker Engine lên chúng và cuối cùng nó sẽ cấu hình Docker Client để giao tiếp với Docker Engine một cách bảo mật

- **Docker Compose**: là công cụ giúp định nghĩa và khởi chạy multi-container Docker applications
- **Docker Swarm**: là một công cụ giúp người dùng tạo ra một clustering Docker. Nó giúp gom nhiều Docker Engine lại với nhau và ta có thể "nhìn" nó như duy nhất một virtual Docker Engine

4. Kiến trúc của Docker

Docker sử dụng kiến trúc client-server. Docker server (hay còn gọi là daemon) sẽ chịu trách nhiệm build, run, distrubute Docker container. Docker client và Docker server có thể nằm trên cùng một server hoặc khác server. Chúng giao tiếp với nhau thông qua REST API dựa trên UNIX sockets hoặc network interface.



Hình 3: Kiến trúc của Docker

4.1 Docker daemon

Docker daemon (dockerd) là thành phần core, lắng nghe API request và quản lý các Docker object. Docker daemon host này cũng có thể giao tiếp được với Docker daemon ở host khác.

4.2 Docker registry

Docker registry là một kho chứa các Image. Nổi tiếng nhất chính là Docker Hub, ngoài ra người dùng có thể tự xây dựng một Docker registry cho riêng mình.

4.3 Docker client

Docker client (docker) là cách chính mà nhiều người dùng Docker tương tác với Docker. Khi người dùng sử dụng các lệnh như *docker run*, client sẽ gửi các lệnh này đến dockerd, daemon sẽ thực hiện. Lệnh docker sử dụng Docker API. Docker client có thể giao tiếp với nhiều daemon.

4.4 Docker Desktop

Docker Desktop là một ứng dụng dễ cài đặt cho môi trường Mac, Windows hoặc Linux, cho phép người dùng build và chia sẻ các ứng dụng và microservices được container hóa. Docker Desktop bao gồm Docker daemon (dockerd), Docker client (docker), Docker Compose, Docker Content Trust, Kubernetes và Credential Helper. Để biết thêm thông tin, xem Docker Desktop.

4.5 Docker object

Các object này chính là các đối tượng mà người dùng thường xuyên gặp phải khi sử dụng Docker gồm có:

Images

- Image là một *template read-only* sử dụng để chạy container.
- Một image có thể base trên một image khác. Ví dụ người dùng muốn tạo một image *nginx*, tất nhiên nginx phải chạy trên linux ubuntu chẳng hạn. Khi đó image nginx trước hết sẽ phải base trên ubuntu trước đã.
- Người dùng cũng có thể tự build image cho riêng mình hoặc tải các image có sẵn của người khác trên Docker registry.

Container

- Container được chạy dựa trên 1 image cụ thể. Người dùng có thể tạo, start, stop, move, delete container.
- Người dùng cũng có thể kết nối các container với nhau hoặc attach storage cho nó, thậm chí là tạo lại một image từ chính state hiện tại của container
- Container cô lập tài nguyên với host và các container khác.

CHƯƠNG 2: THỰC HÀNH MỘT SỐ LỆNH CƠ BẢN

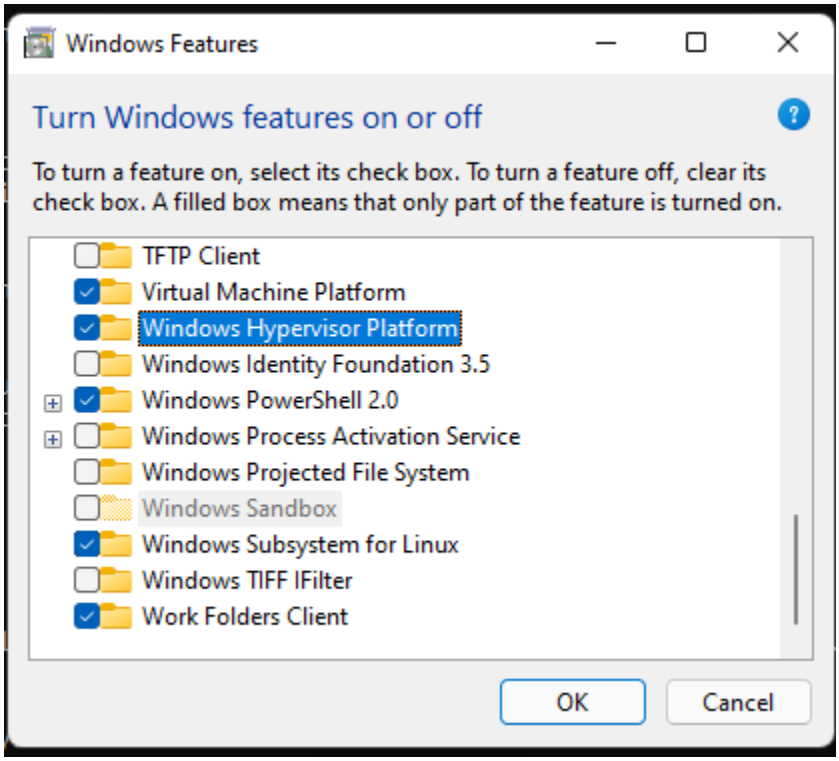
1. Cài đặt Docker Destokp trên Windows 11

Để cài đặt Docker trên Windows 11, cần bật WSL2, vẫn có thể sử dụng Docker mà không cần bật WSL2, tuy nhiên Docker sẽ sử dụng HyperV để ảo hoá nhưng hiệu quả sẽ kém hơn so với sử dụng WSL2.

1.1 Cách bật WSL2

Bước 1: Mở "Windows Features" trong Start Menu.

Bước 2: Bật các tính năng sau: Virtual Machine Platform, Windows Hypervisor Platform và Windows Subsystem for Linux.



Hình 4: Bật các tính năng trong Windows Features

Sau khi bật xong những tính năng trên thì restart lại máy.

Bước 3: Mở CMD bằng quyền Admin và nhập lệnh sau để cập nhật WSL

wsl –update

Bước 4: Nhập lệnh sau để xem các distro có thể cài đặt trên WSL2

wsl --list –online

```
C:¥Windows¥System32>wsl --list --online
The following is a list of valid distributions that can be installed.
Install using 'wsl.exe --install <Distro>'.

NAME                            FRIENDLY NAME
AlmaLinux-8                     AlmaLinux OS 8
AlmaLinux-9                     AlmaLinux OS 9
AlmaLinux-Kitten-10             AlmaLinux OS Kitten 10
AlmaLinux-10                    AlmaLinux OS 10
Debian                          Debian GNU/Linux
FedoraLinux-42                  Fedora Linux 42
SUSE-Linux-Enterprise-15-SP6    SUSE Linux Enterprise 15 SP6
SUSE-Linux-Enterprise-15-SP7    SUSE Linux Enterprise 15 SP7
Ubuntu                          Ubuntu
Ubuntu-24.04                    Ubuntu 24.04 LTS
archlinux                       Arch Linux
kali-linux                      Kali Linux Rolling
openSUSE-Tumbleweed             openSUSE Tumbleweed
openSUSE-Leap-15.6              openSUSE Leap 15.6
Ubuntu-18.04                    Ubuntu 18.04 LTS
Ubuntu-20.04                    Ubuntu 20.04 LTS
Ubuntu-22.04                    Ubuntu 22.04 LTS
OracleLinux_7_9                 Oracle Linux 7.9
OracleLinux_8_10                Oracle Linux 8.10
OracleLinux_9_5                 Oracle Linux 9.5
```

Hình 5: Các distro có thể cài đặt

Bước 5: Nhập lệnh sau để tiến hành cài đặt distro mong muốn. Nếu không điền distro nào thì nó sẽ mặc định là Ubuntu:

wsl --install -d <Distribution>

```
Microsoft Windows [Version 10.0.22000.318]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>wsl --install
Installing: Virtual Machine Platform
Virtual Machine Platform has been installed.
Installing: Windows Subsystem for Linux
Windows Subsystem for Linux has been installed.
Downloading: WSL Kernel
Installing: WSL Kernel
WSL Kernel has been installed.
Downloading: GUI App Support
[==========                18.5%
```

Hình 6: Cài đặt distro Ubuntu

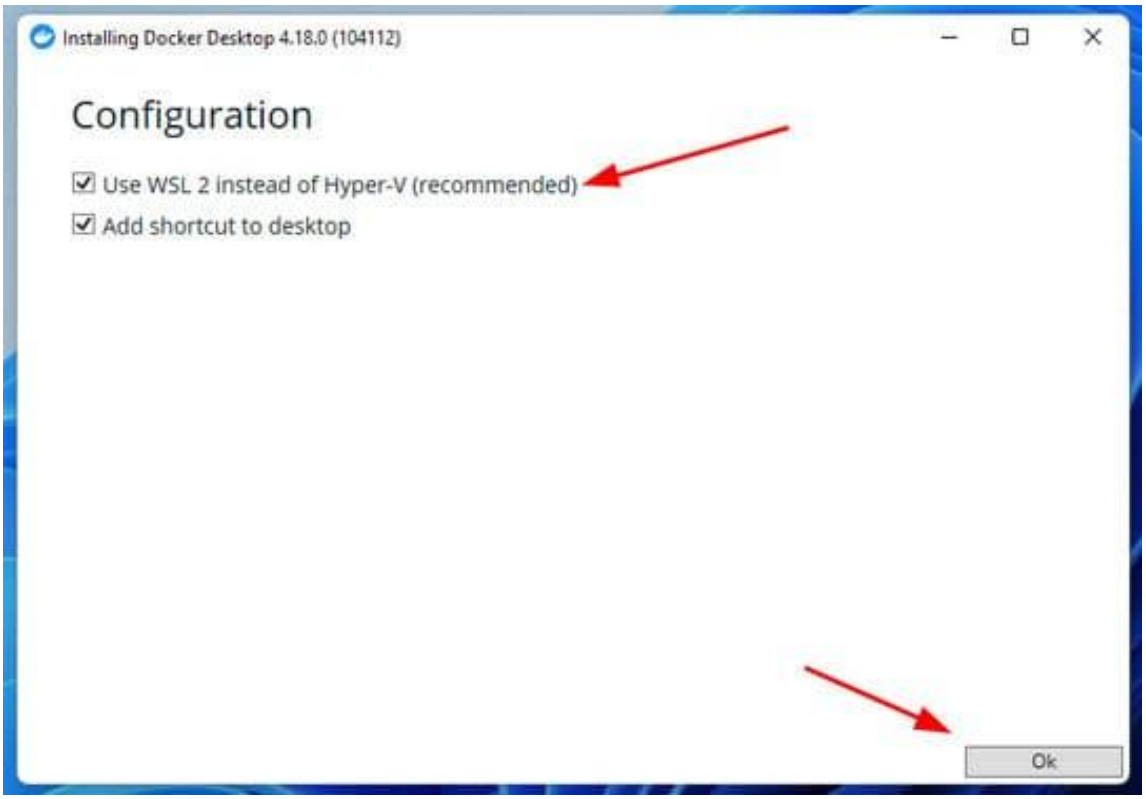
Sau khi cài đặt xong distro, tiến hành thiết lập user và password.

1.2 Cách cài đặt Docker

Bước 1: Tải file Docker bằng cách truy cập vào trang chủ của Docker

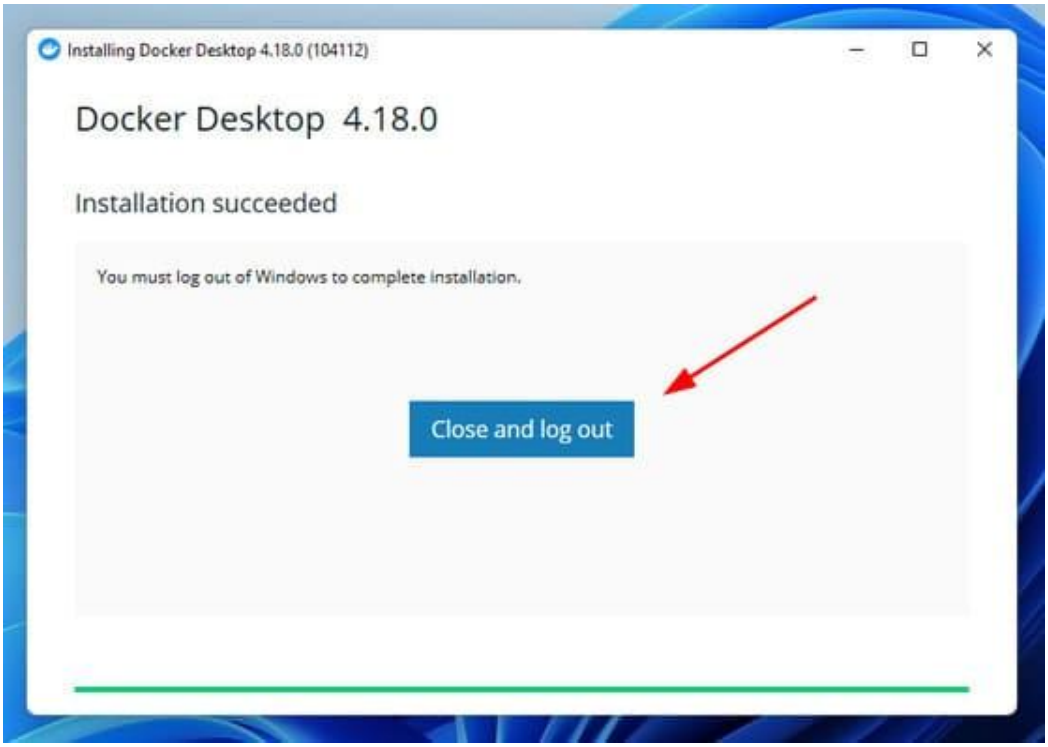
https://docs.docker.com/get-started/get-docker/

Bước 2: Chọn "Use WSL 2 instead of Hyper-V" và nhấn OK.



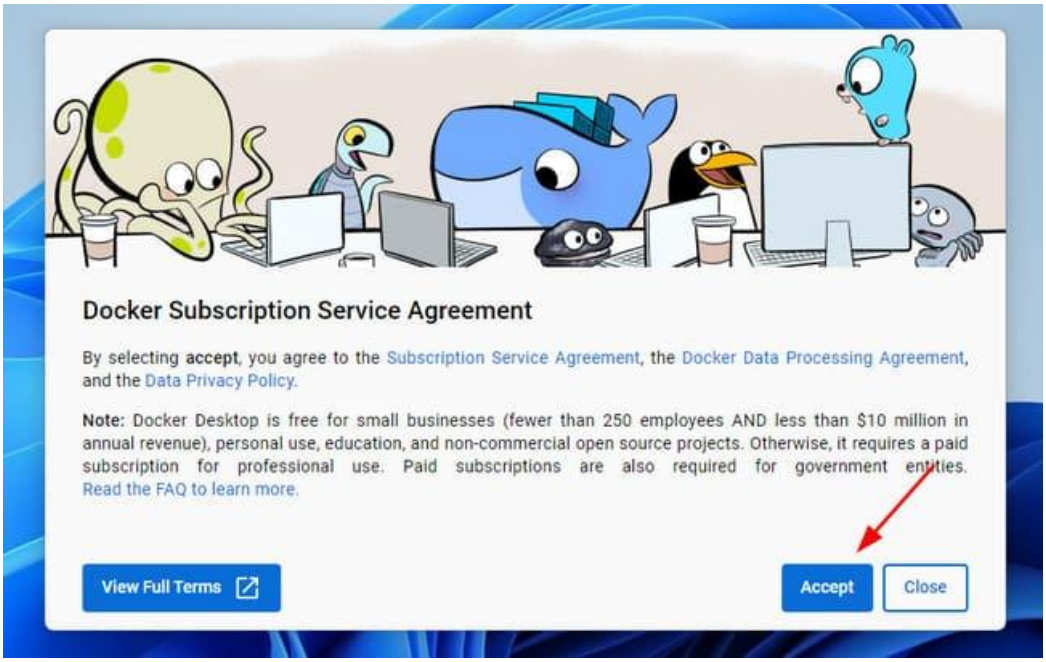
Hình 7: Chọn "Use WSL 2 instead of Hyper-V"

Bước 3: Sau khi cài đặt xong thì nhấn "Close and log out".



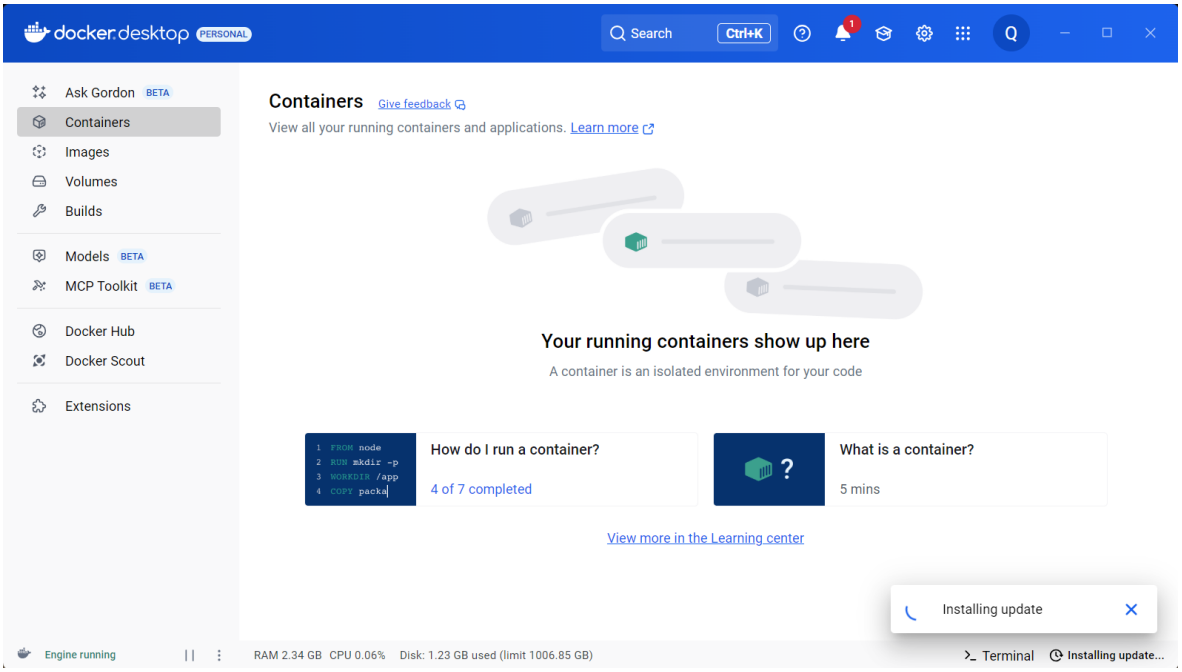
Hình 8: nhấn "Close and log out"

Bước 4: Nhấn "Accept".



Hình 9: Nhấn "Accept"

Cài đặt thành công:



Hình 10: Cài đặt thành công Docker Desktop

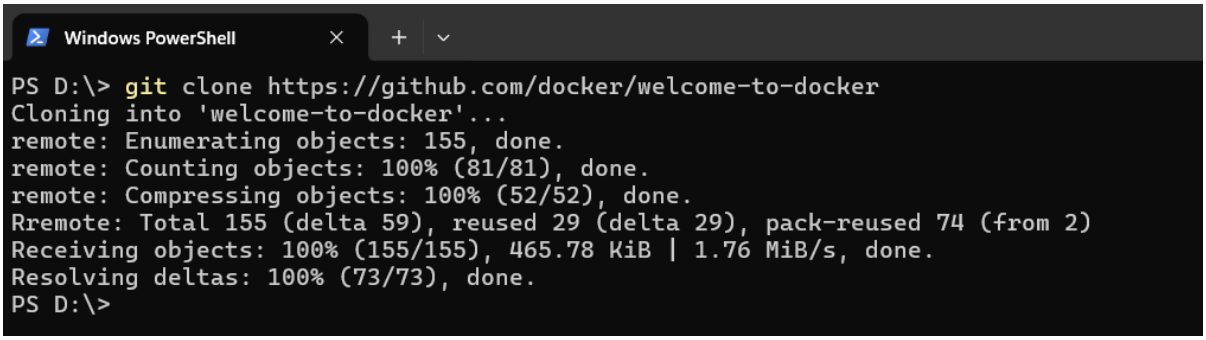
2. Thực hành một số lệnh cơ bản trên Docker

2.1 Thực hành theo hướng dẫn trên Docker Desktop

Bước 1: Clone repository theo hướng dẫn

Clone repository bằng lệnh dưới đây:

git clone https://github.com/docker/welcome-to-docker



Hình 11: Clone repository welcome-to-docker

Di chuyển tới thư mục vừa clone:

cd welcome-to-docker

Bước 2: Kiểm tra Dockerfile

hosts | setup.txt | dockerfile | Dockerfile

File Edit View H1

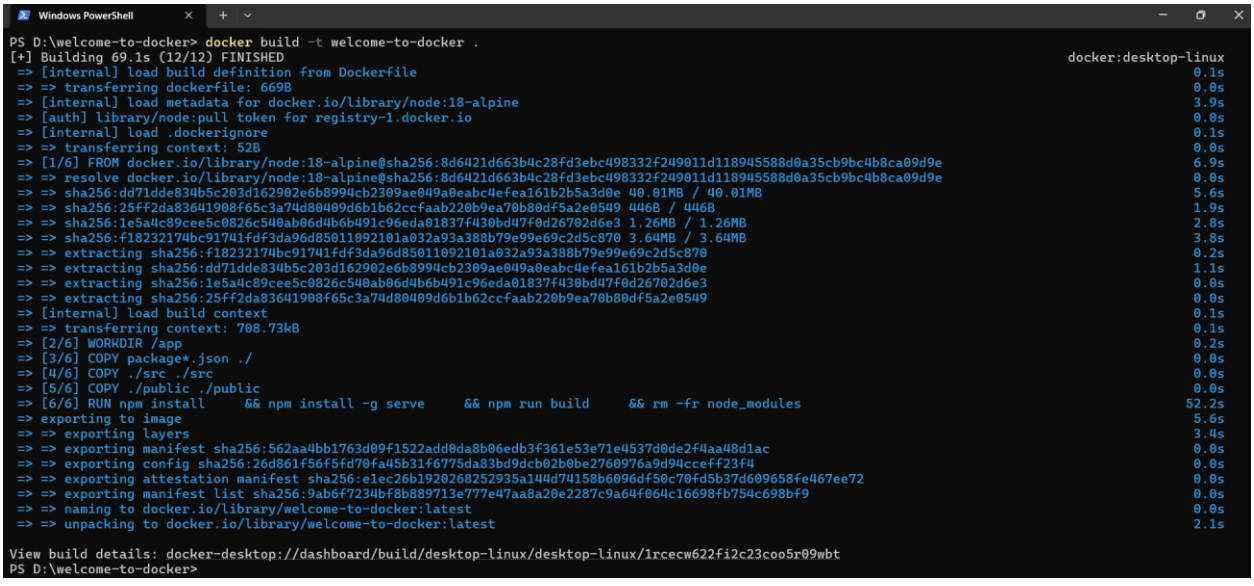
```
# Start your image with a node base image
FROM node:18-alpine

# The /app directory should act as the main application directory
WORKDIR /app

# Copy the app package and package-lock.json file
COPY package*.json ./

# Copy local directories to the current local directory of our docker image (/app)
COPY ./src ./src
COPY ./public ./public

# Install node packages, install serve, build the app, and remove dependencies at the end
RUN npm install \
    && npm install -g serve \
    && npm run build \
    && rm -fr node_modules

EXPOSE 3000

# Start the app using serve command
CMD [ "serve", "-s", "build" ]
```

Hình 12: Dockerfile của thư mục welcome-to-docker

Bước 3: Build image

Sử dụng câu lệnh:

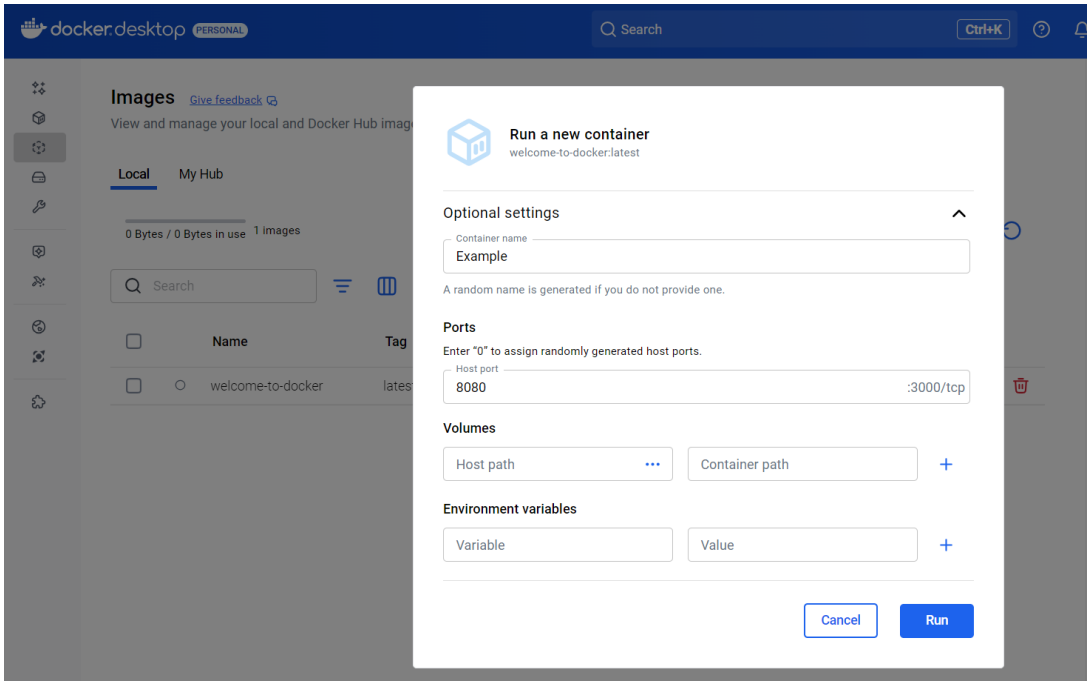
docker build -t welcome-to-docker .



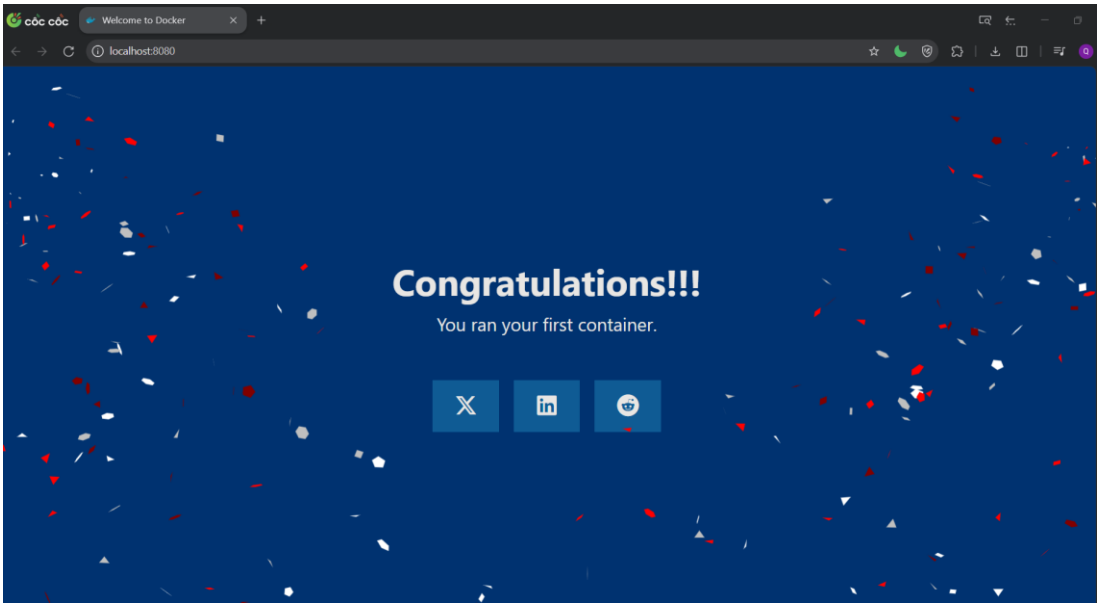
Hình 13: Build thành công

Bước 4: Run container

Khi quá trình build hoàn tất, một image sẽ xuất hiện trong tab Images. Chọn tên image để xem chi tiết. Chọn Run để chạy nó dưới dạng một container. Trong phần Optional settings, chỉ định một số cổng để container có thể được truy cập từ bên ngoài.



Hình 14: Run container



Hình 15: Thực hành thành công

2.2 Thực hành với một số ví dụ cơ bản

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS D:\Docker> docker search mysql
NAME                  DESCRIPTION                                     STARS     OFFICIAL
mysql                 MySQL is a widely used, open-source relation…   15850     [OK]
bitnami/mysql         Bitnami container image for MySQL               139
circleci/mysql        MySQL is a widely used, open-source relation…   31
bitnamicharts/mysql   Bitnami Helm chart for MySQL                     0
cimg/mysql                                                            3
ubuntu/mysql          MySQL open source fast, stable, multi-thread…   68
google/mysql          MySQL server for Google Compute Engine          26
linuxserver/mysql     A Mysql container, brought to you by LinuxSe…   44
elestio/mysql         Mysql, verified and packaged by Elestio         1
docksal/mysql         MySQL service images for Docksal - https://d…   0
alpine/mysql          mysql client                                    3
eclipse/mysql         Mysql 5.7, curl, rsync                          2
datajoint/mysql       MySQL image pre-configured to work smoothly …   2
ilios/mysql           Mysql configured for running Ilios              1
ddev/mysql            ARM64 base images for ddev-dbserver-mysql-8.…   1
mirantis/mysql                                                        0
corpusops/mysql       https://github.com/corpusops/docker-images/     0
mysql/mysql-server    Optimized MySQL Server Docker images. Create…   1032
vulhub/mysql                                                          1
cbioportal/mysql      This repository hosts MySQL database images …   1
```

Hình 16: Sử dụng lệnh docker search

```
PS D:\Docker> docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
237611fb5faf: Pull complete
ccb04e1eb0a7: Pull complete
8d917159a675: Pull complete
cb6b011730e2: Pull complete
90dac1e734aa: Pull complete
f3e3a6933245: Pull complete
f4b7ec9de513: Pull complete
f6e58dad121f: Pull complete
6be37253424c: Pull complete
f1d28d66c159: Pull complete
Digest: sha256:297f5ead7043a440ce84b3b0f3b77430a4f233c2578ff15fff8f0de54f67f22d
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
PS D:\Docker>
```

Hình 17: Sử dụng lệnh docker pull để tải mysql

Ví dụ 1: Tạo và chạy Docker image từ file ex01

```
ex01.py
ex01.py
1   print("Hello Docker!")
2
```

Hình 18: file ex01

dockerfile

```dockerfile
# Dựa trên image cơ bản nào
FROM python:3

# Khai báo thư mục làm việc
WORKDIR /app

# Copy toàn bộ file mã nguồn và các file khác vào image
COPY . .

# Thực hiện chạy lệnh
CMD ["python","./ex01.py"]
```

Hình 19: Dockerfile

```
PS D:\Docker> docker build -t quyennv/image_ex01 .
[+] Building 1.6s (8/8) FINISHED
 => [internal] load build definition from dockerfile
 => => transferring dockerfile: 273B
 => [internal] load metadata for docker.io/library/python:3
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/3] FROM docker.io/library/python:3@sha256:28f60ab75da2183870846130cead1f6af30162148d3238348f78f89cf6160b5d
 => => resolve docker.io/library/python:3@sha256:28f60ab75da2183870846130cead1f6af30162148d3238348f78f89cf6160b5d
 => [internal] load build context
 => => transferring context: 381B
 => CACHED [2/3] WORKDIR /app
 => [3/3] COPY . .
 => exporting to image
 => => exporting layers
 => => exporting manifest sha256:664e2464d305262e612d1f9cf6a331a55ab2f2ca245a9d2acc28499bcf3aa0df
 => => exporting config sha256:334aeaf5ea24c55fd18f066b1bd0c6bc5bddceec68eafb73c582b4dc69a7c078
 => => exporting attestation manifest sha256:6721221767ed54bab2d52d874f67a98ac16e9f0b5d96fbfeef7092e3f16b34ef
 => => exporting manifest list sha256:282781576377141357b77c498d3b034138cdfcf398b1b12658378a5a7e88c169
 => => naming to docker.io/quyennv/image_ex01:latest
 => => unpacking to docker.io/quyennv/image_ex01:latest

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/xtlyrgf9wdcybanuz9ly0t3g2
```

Hình 20: Build thành công

```
PS D:\Docker> docker run --name ex01 quyennv/image_ex01
Hello Docker!
PS D:\Docker>
```

Hình 21: Chạy thành công container

Ví dụ 2: Tạo và chạy Docker image từ file ex02

ex01.py | ex02.py 1 | dockerfile

ex02.py > ...

```python
# Triển khai Flask HTTP API
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    return "Hello from Flask (in Docker)"

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port='5000', use_reloader=False)
```

Hình 22: file ex02

ex01.py | ex02.py 1 | dockerfile

dockerfile

```
# Dựa trên image cơ bản nào
FROM python:3

# Khai báo thư mục làm việc
WORKDIR /app

# Copy toàn bộ file mã nguồn và các file khác vào image
COPY . .

# Cài đặt flask
RUN pip install flask

# Thực hiện chạy lệnh
CMD ["python","./ex02.py"]
```

Hình 23: Thêm dòng cài đặt flask vào Dockerfile

```
PS D:\Docker> docker build -t quyennv/image_ex02 .
[+] Building 1.4s (9/9) FINISHED
 => [internal] load build definition from dockerfile
 => => transferring dockerfile: 316B
 => [internal] load metadata for docker.io/library/python:3
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/4] FROM docker.io/library/python:3@sha256:28f60ab75da2183870846130cead1f6af30162148d3238348f78f89cf6160b5d
 => => resolve docker.io/library/python:3@sha256:28f60ab75da2183870846130cead1f6af30162148d3238348f78f89cf6160b5d
 => [internal] load build context
 => => transferring context: 424B
 => CACHED [2/4] WORKDIR /app
 => CACHED [3/4] COPY . .
 => CACHED [4/4] RUN pip install flask
 => exporting to image
 => => exporting layers
 => => exporting manifest sha256:09432ed734ce33fcb1aecee509c7e5986c224952b06c6c846ad4369fb826e02f
 => => exporting config sha256:5c7682f5fd5956bdaa9753f17347af1978a80c4ccbbef5082bbae7ea55a3fe12
 => => exporting attestation manifest sha256:c7ad77a2c91aff4055076b1ccf4d0ca5d4e7f8f80a52b69e0961953cba8294b0
 => => exporting manifest list sha256:439e400f4f2368bfb97bf1278eba659c376f0054b205046169e74c866bc4412b
 => => naming to docker.io/quyennv/image_ex02:latest
 => => unpacking to docker.io/quyennv/image_ex02:latest

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/8n6g7cgtaewfdnhr9y3yfxh8o
```

Hình 24: Build thành công

Để chạy container sử dụng câu lệnh:

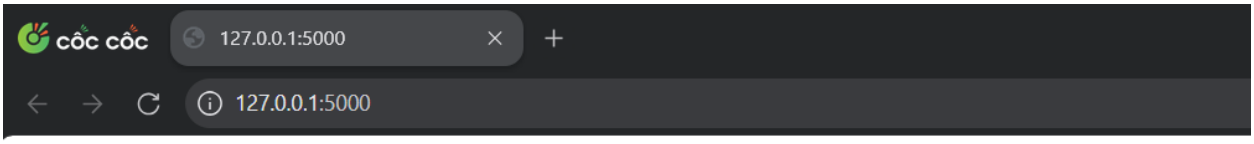
docker run --name ex02 -p 5000:5000 quyennv/image_ex02

Trong đó:

- *-p 5000:5000* - Cổng 5000 bên trong container (nơi Flask chạy) được map ra cổng 5000 bên ngoài máy host.

```
PS D:\Docker> docker run --name ex02 -p 5000:5000 quyennv/image_ex02
 * Serving Flask app 'ex02'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.17.0.3:5000
Press CTRL+C to quit
```

Hình 25: Chạy container từ image vừa build



Hình 26: Thực hành thành công

Ví dụ 3: Tạo và chạy Docker image từ file ex03

ex01.py | ex02.py 1 | ex03.py 2 | dockerfile

ex03.py > hello

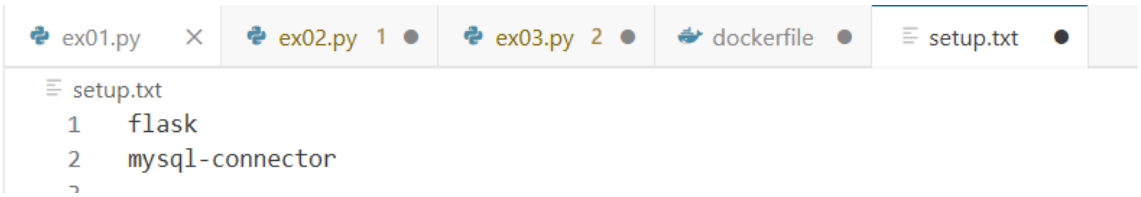
```python
from flask import Flask
import mysql.connector

app = Flask(__name__)

@app.route('/')
def hello():
    # MySQL param
    MYSQL_HOST = 'mysql'
    MYSQL_DB = 'mysql'
    MYSQL_USER = 'root'
    MYSQL_PASS = 'password'
    rowcount = -1
    connection = None
    try:
        connection = mysql.connector.connect(host=MYSQL_HOST, database=MYSQL_DB, user=MYSQL_USER,
                                             password=MYSQL_PASS, port=3306, auth_plugin='mysql_native_password')
        mysql_insert_query = "SELECT * FROM mysql.user"
        print(mysql_insert_query)
        cursor = connection.cursor()
        cursor.execute(mysql_insert_query)
        records = cursor.fetchall()
        rowcount = cursor.rowcount

    except mysql.connector.Error as error:
        print("Fail login {}".format(error))
        ret_result = -1
    finally:
        if connection is not None:
            connection.close()
        del connection

    return "Total user numers = : "+  str(rowcount)

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port='5000', use_reloader=False)
```

Hình 27: file ex03

dockerfile

```
# Dựa trên image cơ bản nào
FROM python:3

# Khai báo thư mục làm việc
WORKDIR /app

# Copy toàn bộ file mã nguồn và các file khác vào image
COPY . .

# Cài đặt cac thư viện cần thiết
RUN pip install -r setup.txt

# Thực hiện chạy lệnh
CMD ["python","./ex03.py"]
```

Hình 28: Dockerfile



Hình 29: file setup.txt chứa các thư viện cần cài đặt

```
PS D:\Docker> docker build -t quyennv/image_ex03 .
[+] Building 36.5s (10/10) FINISHED
 => [internal] load build definition from dockerfile
 => => transferring dockerfile: 349B
 => [internal] load metadata for docker.io/library/python:3
 => [auth] library/python:pull token for registry-1.docker.io
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/4] FROM docker.io/library/python:3@sha256:28f60ab75da2183870846130cead1f6af30162148d3238348f78f89cf6160b5d
 => => resolve docker.io/library/python:3@sha256:28f60ab75da2183870846130cead1f6af30162148d3238348f78f89cf6160b5d
 => [internal] load build context
 => => transferring context: 1.56kB
 => CACHED [2/4] WORKDIR /app
 => [3/4] COPY . .
 => [4/4] RUN pip install -r setup.txt
 => exporting to image
 => => exporting layers
 => => exporting manifest sha256:3a36b2f3000f6d97ecc47936e6901f97dcb5aec5f65a27a5267e08665dfb99e4
 => => exporting config sha256:fb1e6201c4f88368ea7afa50b8637137ae7c20d89f24b9bb9b90bbaaca234977
 => => exporting attestation manifest sha256:042fc288b5a12993fd51ad654f1e4ef390cf0f57057cc754c29ca045b4ae4b48
 => => exporting manifest list sha256:6c9fd8bb2c1b84d56c43525a89b6d91c4c14f320086ff5001dccd1f98af78e76
 => => naming to docker.io/quyennv/image_ex03:latest
 => => unpacking to docker.io/quyennv/image_ex03:latest

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/xels7c02eburman278c9tvt6e
```

Hình 30: Build thành công

Để kết nối được ex03 với mysql, thực hiện nhóm cả 2 vào một Docker network:

```
PS D:\Docker> docker network list
NETWORK ID     NAME      DRIVER    SCOPE
d88b8d11b2c6   bridge    bridge    local
11fd4a212cc0   host      host      local
60e67caa0574   none      null      local
```

Hình 31: Danh sách các mạng trong Docker network

```
PS D:\Docker> docker network create mynet
094e278bb0aab5427e36e5dae2178c246830b88099df9e6bc468e008bb5ab535
PS D:\Docker> docker network list
NETWORK ID     NAME      DRIVER    SCOPE
d88b8d11b2c6   bridge    bridge    local
11fd4a212cc0   host      host      local
094e278bb0aa   mynet     bridge    local
60e67caa0574   none      null      local
```

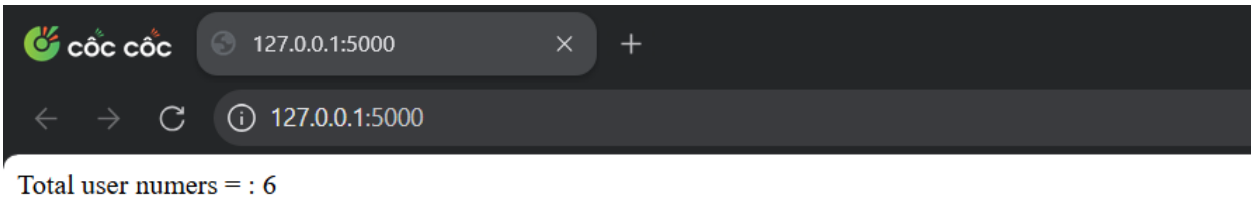
Hình 32: Tạo thêm mạng mynet

```
PS D:\Docker> docker run --name mysql --net mynet -e MYSQL_ROOT_PASSWORD=root -d mysql
13a41d8233d01ec2eb48dbf23c8d965299e8d923312665ad6aac175b500adb1c
PS D:\Docker> docker exec -it mysql bash
bash-5.1# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 9.3.0 MySQL Community Server - GPL

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ALTER USER 'root'@'%' IDENTIFIED BY 'quyen161';
Query OK, 0 rows affected (0.007 sec)

mysql> flush privileges;
Query OK, 0 rows affected, 1 warning (0.010 sec)
```

Hình 33: Chạy và cấu hình mysql

```
PS D:\Docker> docker run --name ex03 -p 5000:5000 --net mynet quyennv/image_ex03
 * Serving Flask app 'ex03'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.18.0.3:5000
Press CTRL+C to quit
```

Hình 34: Chạy container từ image ex03



Hình 35: Thực hành thành công

TÀI LIỆU THAM KHẢO

[1] https://blog.cloud365.vn/container/tim-hieu-docker-phan-3/
[2] https://viblo.asia/p/docker-va-nhung-kien-thuc-co-ban-YWOZrp075Q0
[3] https://luanbn.wordpress.com/2015/08/27/docker-part3-cai-dat-docker-tren-ubuntu-14-04/