

Đề tài: Tìm Hiểu và Triển khai Docker

1. Danh sách thành viên & Công việc

Họ & tên	MSSV	Công việc	Tiến độ
Đào Ninh Thái	175A071427	- Cài đặt docker và triển khai web - Hướng dẫn sử dụng	-Xong cài đặt docker trên linux, -Chạy web wordpress bằng cách cài các container qua docker-compose
Nguyễn Thanh Tùng	175A071423	- Tìm hiểu tài liệu về Docker. - Phân tích Ưu điểm / Nhược điểm của Docker. - So sánh Docker và các phần mềm tương tự khác	100%

2. Nội dung nghiên cứu

a.Giới thiệu

a. Giới thiệu

1) Tại sao cần Docker

- Phát triển ứng dụng ngày yêu cầu không những khả năng viết code mà còn đòi hỏi rất nhiều công cụ khác. Sử dụng nhiều ngôn ngữ lập trình bậc cao như C#,C++,Python,....., hay những phần mềm cơ sở dữ liệu như mySQL, MariaDB..., hoặc hàng tỷ khác như khung phần mềm, hay kiến trúc phần mềm được update ngày qua ngày như PHP, Nodejs, MongoDB, Laravel, Redis....., việc cài đặt cùng lúc nhiều phần mềm này để chạy cho cùng lúc nhiều dự án cũng như triển khai chúng trên nhiều máy tính khác nhau hoặc từ môi trường local lên sever là cực kỳ khó khăn, tốn thời gian, tốn tiền do tốn thời gian của nhà phát triển, và thế là Docker ra đời, phần mềm này giống như một thứ tôn giáo của nhà phát triển, Docker sẽ tạo ra một môi trường ảo hóa cho phép chạy nhiều ứng dụng của cả Windows và Linux trên đó mà ko làm ảnh hưởng tới môi trường gốc do đó không phải sợ hiện tượng CSDL dẫm chân lên nhau rồi lẫn ra chết nữa .

2) Docker là gì và cách thức hoạt động lý do vì sao chọn Docker

- Docker là một chương trình mà trong đó Docker sẽ tạo ra một môi trường ảo hóa cho phép chạy nhiều ứng dụng của cả Windows, Linux, MacOS trên đó mà không làm ảnh hưởng tới môi trường của HĐH “Gốc”
- Docker được cấu tạo bởi những thành phần chính sau :

Docker Image	là OS, một ứng dụng(PHP,SQL...) đã được cài đặt và đóng gói. Image chỉ có quyền đọc.
Docker Container	là bản thực thể của một image, được clone ra từ image, mọi người sẽ sử dụng và làm việc trên container là chính
Docker Engine	Tạo, vận chuyển và chạy các container Docker có thể triển khai trên vật lý hoặc ảo, lưu trữ cục bộ, trong trung tâm dữ liệu hoặc nhà cung cấp dịch vụ đám mây, bao gồm cả Daemon
Docker HUB	nằm ở server internet, còn images và container nằm ở máy người dùng

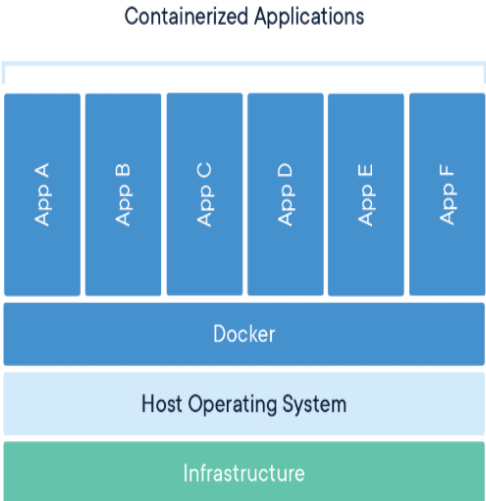
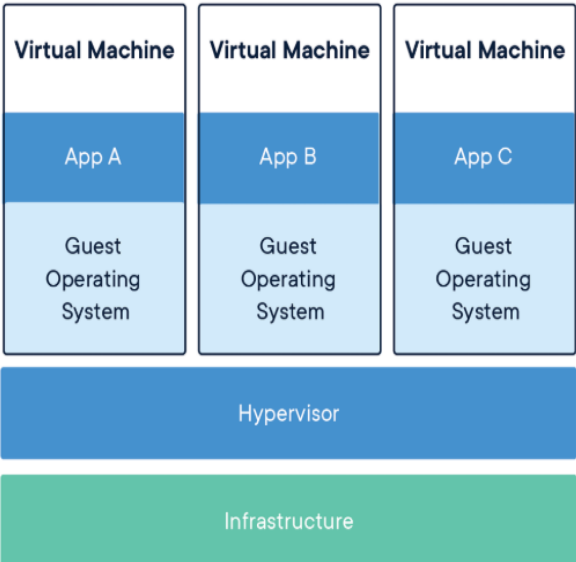
- Ngoài ra trong quá trình sử dụng Docker cũng có những tính năng khác dùng cho những công việc cụ thể sau:

Docker Compose	Docker-compose là tool để cấu hình và chạy nhiều docker container cùng lúc. Dùng docker-compose sẽ giúp ta dễ dàng hơn trong việc chạy cùng lúc 1 hoặc 1 số container cần thiết cho project, đồng thời giúp ta dễ dàng visualize (nhìn) tổng quan về project.
Dockerfile	Dockerfile đơn giản là một file (dạng text nhưng không có Extension) chứa một tập hợp các dòng lệnh dùng để khởi tạo một docker image. Nó quy định image sẽ được khởi tạo như thế nào, gồm các ứng dụng gì trong đó.
DOCKER SWARM	Docker Swarm là một công cụ phân cụm và lập lịch cho các container Docker. Với Swarm, các quản trị viên và nhà phát triển CNTT có thể thiết lập và quản lý một cụm các nút Docker dưới dạng một hệ thống ảo duy nhất.
PORT	<p>Vì môi trường trong Docker độc lập hoàn toàn so với môi trường gốc. Nên để có thể sử dụng được ứng dụng chạy trong Docker thì ta cần mở port từ Docker để bên ngoài có thể gọi vào được.</p> <p>Ở container ta mở port 8080 cho môi trường ngoài truy</p>

	cập, từ môi trường ngoài ta mở cổng 8888 để user thật có thể truy vấn. Do đó user sẽ gọi đến cổng 8888 để giao tiếp với ứng dụng chạy trong Docker container.
VOLUME	Vì môi trường Docker độc lập nên hệ thống file của ứng dụng chạy trong Docker cũng độc lập, mà thực tế thì hầu như ta luôn cần lưu lại file: lưu trữ DB, lưu trữ log, ảnh,... Do đó để môi trường gốc truy cập được vào file system của ứng dụng chạy trong Docker thì ta cần tới Volume

- Nói sơ qua thì nhìn chung Docker khá giống với những phần mềm máy ảo khác như Virtual Box.

Hãy xem qua bảng so sánh sau :

	
<p>- Với Docker bạn có thể chạy nhiều Project với các Container ngoài ra để phân cứng để chạy Docker với máy thật là như nhau bạn không cần phải share 1 lượng lớn tài nguyên của mình. Các kích thước của các Container cũng rất nhẹ giúp chương trình chạy tốt hơn.</p>	<p>- Với VB với mỗi Project bạn sẽ lại phải chạy 1 máy ảo bạn sẽ lại phải chia tài nguyên phần cứng dành cho máy ảo ngoài ra ổ cứng của bạn cũng phải nhận thêm dung lượng của hệ điều hành máy ảo, nó rất nặng nề và chậm chạp ngay cả Core i9 chắc cũng chỉ giúp bạn chạy được 10 ứng dụng là hết đất</p> <p>Ngoài ra nếu máy ảo bị lỗi bạn phải xóa đi cài lại lúc đó mới thấm đau.</p>

VD: Docker bạn ở trong một ngôi nhà có nhiều phòng cần ăn uống thì vào nhà bếp cần đi vệ sinh vào phòng vệ sinh cần giải trí vào phòng giải trí rất tiện lợi, nhanh chỉ trong diện tích của 1 ngôi nhà. Còn với VB bạn đang ở trong 1 dãy phố siêu to khổng

lò muốn ăn bạn phải vào nhà hàng muốn đi vệ sinh thì cần vô nhà vệ sinh công cộng muốn giải trí thì tới casino việc này làm bạn tốn rất nhiều Tiền diện tích xây dựng thì nhiều cũng như thời gian di chuyển thì lâu.

Docker còn có rất nhiều tính năng hữu ích bắt đầu bằng chữ TỰ ĐỘNG .

Docker thật sự là một phần mềm nhanh tiện lợi và miễn phí ngoài ra cộng đồng đông đảo sẵn sàng Support tận răng người dùng mới tuy vậy Jerusalem cũng có lúc thất thủ nên thật ra Docker vẫn có một số nhược điểm sau:

- + Các Container của Docker hoạt động mặc dù nhanh hơn rất nhiều so với máy ảo nhưng nó vẫn không thể chạy với 100% công suất. lý do có thể do bị quá tải
- + Hệ sinh thái của Docker vẫn chưa đủ mạnh mặc dù là một phần mềm mã nguồn mở nhưng nó không có được sự hỗ trợ từ một số phần mềm do liên quan tới các công ty đối thủ .
- + Lưu trữ dữ liệu trong Docker khá khó khăn, các dữ liệu trong Container sẽ biến mất khi bạn tắt nó đi trừ khi bạn lưu nó ở một nơi nào đó trước đó bạn có thể sử dụng Docker Volume nhưng đây vẫn là một vấn đề mà Docker chưa xử lý được triệt để.
- + Các ứng dụng đồ họa không hoạt động tốt trong các Container mặc dù môi trường Docker cũng như SV không dung GUI nhưng có một số nhà sáng tạo đã làm một số cách để GUI hoạt động trong Container nhưng nhìn chung chúng khá khó hiểu và phức tạp .
- + Không phải ứng dụng nào cũng hưởng lợi từ Docker nhìn chung các phần mềm với kiến trúc microservices sẽ chạy tốt trên Docker .
- + Docker khá khó sử dụng và cài đặt vì ko có GUI đòi hỏi người dung có kiến thức về máy tính .

b. Hướng dẫn cài đặt

B1: Kiểm tra kiến trúc và hệ điều hành để chuẩn bị cài đặt

Platform	x86_64 / amd64	ARM	ARM64 / AARCH64	IBM Power (ppc64le)	IBM Z (s390x)
CentOS	✓		✓		
Debian	✓	✓	✓		
Fedora	✓		✓		
Raspbian		✓	✓		
Ubuntu	✓	✓	✓	✓	✓

Cách 1: cài đặt theo Official Repository (Kho lưu trữ chính thức của Docker)

B2: Nếu máy bạn đã có Docker bản cũ thì hãy tiến hành gỡ bỏ.

```
sudo apt-get remove docker docker-engine docker.io containerd runc
```

B3: Có nhiều cách để cài Docker ở đây sẽ cài đặt theo Repository (Kho).

Cập nhật chỉ apt và cài đặt các gói để cho phép apt sử dụng kho lưu trữ qua HTTPS:

```
$ sudo apt-get update

$ sudo apt-get install \
apt-transport-https \
ca-certificates \
curl \
gnupg-agent \
software-properties-common
```

B4: Thêm khóa GPG chính thức của Docker dùng để mã hóa.

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

B5: Khóa GPG của Docker official là 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88, bằng cách tìm kiếm 8 ký tự cuối cùng của dấu vân tay.

```
$ sudo apt-key fingerprint 0EBFCD88

pub  rsa4096 2017-02-22 [SCEA]
9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88
uid      [ unknown] Docker Release (CE deb) <docker@docker.com>
sub  rsa4096 2017-02-22 [S]
```

B6: Kiến trúc người dung hiện tại là x86_64/amd64 sử dụng lệnh sau để thiết lập kho lưu trữ ổn định .

```
$ sudo add-apt-repository \  
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \  
$(lsb_release -cs) \  
stable"
```

B7: Cài đặt Docker Engine hoặc nếu bạn muốn cài đặt cho một phiên bản tùy ý thì tới B8

```
$ sudo apt-get update  
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

B8: Cài đặt cho một bản tùy chỉnh .
Để kiểm tra các phiên bản hỗ trợ trên máy .

```
$ apt-cache madison docker-ce  
  
docker-ce | 5:18.09.1~3-0~ubuntu-xenial | https://download.docker.com/linux/ubuntu  
xenial/stable amd64 Packages  
docker-ce | 5:18.09.0~3-0~ubuntu-xenial | https://download.docker.com/linux/ubuntu  
xenial/stable amd64 Packages  
docker-ce | 18.06.1~ce~3-0~ubuntu      | https://download.docker.com/linux/ubuntu  
xenial/stable amd64 Packages  
docker-ce | 18.06.0~ce~3-0~ubuntu      | https://download.docker.com/linux/ubuntu  
xenial/stable amd64 Packages  
...
```

VD : Bạn dùng 5:18.09.1~3-0~ubuntu-xenial thì chạy

```
$ sudo apt-get install docker-ce=5:18.09.1~3-0~ubuntu-xenial docker-ce-cli=5:18.09.1~3-  
0~ubuntu-xenial containerd.io
```

Bước 9: Kiểm tra xem bạn đã cài đặt docker thành công chưa

```
$ sudo docker run hello-world
```

Bước 10: Khi chạy dự án thực tế thì hầu như ta sẽ sử dụng Docker Compose để chạy project, vì thực tế hầu như ta luôn cần nhiều hơn 1 container cho 1 project, tiến hành cài đặt Docker Compose.

```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.25.5/docker-  
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

Bước 11: Thiết lập quyền.

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

Bước 12: Kiểm tra

```
$ docker-compose --version
```

Cách 2: cài đặt theo Default Repositories (Kho lưu trữ mặc định)

B1: Nâng cấp các ứng dụng

```
$ sudo apt-get update
```

B2: Gỡ bỏ bản docker cũ

```
$ sudo apt-get remove docker docker-engine docker.io
```

B3: Cài docker

```
$ sudo apt install docker.io
```

B4: Khởi động Docker

```
$ sudo systemctl start docker
```

```
$ sudo systemctl enable docker
```

B5: Cài đặt Docker-compose

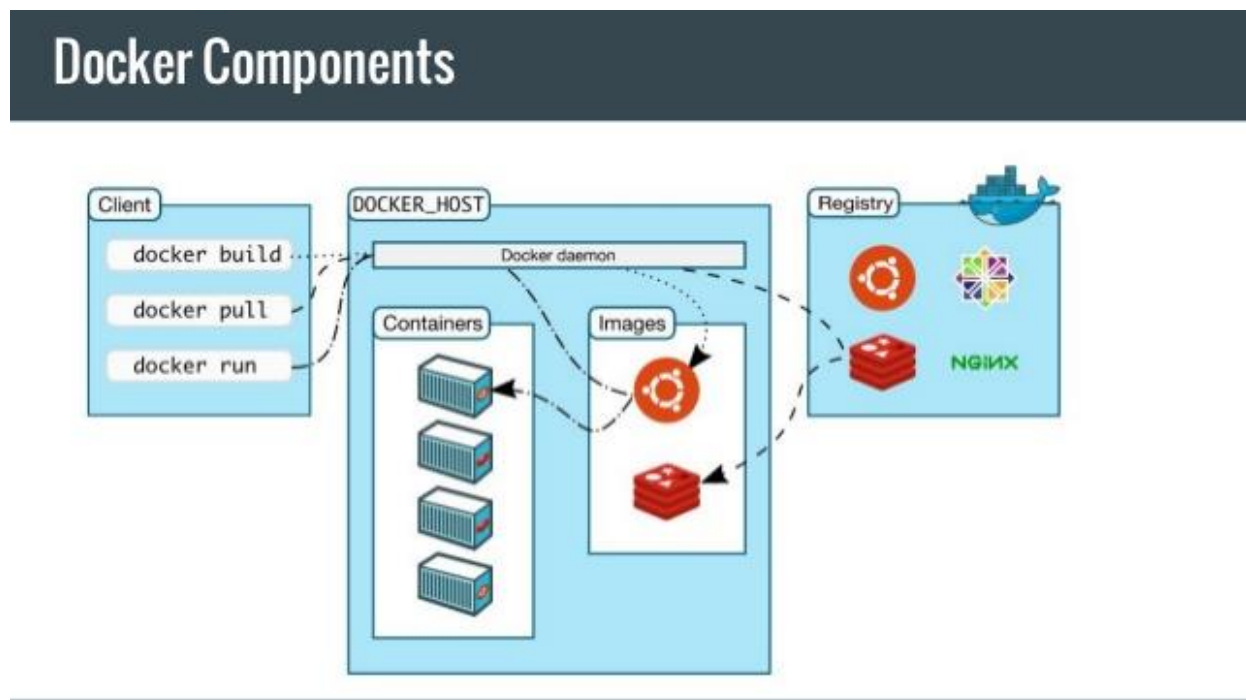
```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.25.5/docker-  
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

B6: Xét quyền cho Docker-compose

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

c. Hướng dẫn sử dụng / quản trị

1. Các thành phần của docker



Registry => Images => Container

Registry (Docker Hub) là server trung tâm nơi chứa các images original, hoặc các bản images đã được cài đặt chỉnh sửa theo nhu cầu riêng biệt.

Images: là OS, một ứng dụng đã được cài đặt và đóng gói. Image chỉ có quyền đọc.

Container là bản thực thể của một image, được clone ra từ image, mọi người sẽ sử dụng và làm việc trên container là chính

Registry (Docker Hub) nằm ở server internet, còn images và container nằm ở máy người dùng.

2. các lệnh cơ bản thường dùng

- Để hiển thị các images ta sử dụng lệnh

```
$ docker images
```


- Để hiển thị các container ta sử dụng lệnh

*Liệt kê các container

```
$ docker ps -a
```

*Chỉ liệt kê container đang chạy background

```
$ docker ps
```

- Để khởi động / dừng các container ta sử dụng lệnh

```
$ docker start/stop <container name>
```

- Để xóa container ta sử dụng lệnh

```
$ docker rm -f < container name>
```

*Xóa tất cả container

```
$ docker rm $(docker ps -a -q)
```

- Để truy cập vào container đang chạy

```
$ docker exec -it <id / container name>
```

- Để xem danh sách volume

```
$ docker volume ls
```

- Để xóa volume

```
$ docker rm <volume name>
```

- Để xóa image

```
$ docker rmi <id / image name>
```

- Để xóa object không sử dụng

```
$ docker system prune
```

- Để xóa tất cả container không sử dụng

```
$ docker container prune
```

- Để xem thông tin network

```
$ docker network ls
```

- Để chạy docker-compose

```
$ docker-compose up
```

- Để pull image từ docker hub

```
$ docker pull <author>/<image name>:<tag>
```

- Ngoài ra còn rất nhiều lệnh liên quan đến docker khác.

3.Chạy thử

Muốn sử dụng docker có nhiều hướng , trong bài này nhóm em sẽ sử dụng docker-compose để build một web server bao gồm mysql , phpmyadmin, wordpress

Bước 1 : Để sử dụng docker-compose , chúng ta cần tải docker-compose xuống

```
$ sudo curl -L https://github.com/docker/compose/releases/download/1.21.2/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose
```

Bước 2: Thêm group để không cần sudo

```
$ sudo groupadd docker
```

```
$ sudo usermod -aG docker $USER
```

```
$ newgrp docker
```

Bước 3: tạo một thư mục trong home đặt tên là test3

```
mkdir test3
```

```
$ mkdir test3
```

Bước 4: di chuyển vào thư mục vừa tạo và tạo file docker-compose.yml

```
$ cd test3
```

```
nano docker-compose.yml
```

```
$ nano docker-compose.yml
```

Bước 5 :Trong file docker-compose.yml ta viết

```
version: '3.3'
```

```
services:
```

```
  db:
```

```
    image: mysql:5.7
```

```
    volumes:
```

```
      - ./db:/var/lib/mysql
```

restart: always

environment:

MYSQL_ROOT_PASSWORD: root

MYSQL_DATABASE: wordpress

MYSQL_USER: root

MYSQL_PASSWORD: root

wordpress:

depends_on:

- db

image: wordpress:latest

ports:

- 8008:80

restart: always

environment:

WORDPRESS_DB_HOST: db:3306

WORDPRESS_DB_USER: root

WORDPRESS_DB_PASSWORD: root

WORDPRESS_DB_NAME: wordpress

phpmyadmin:

image: phpmyadmin/phpmyadmin

restart: always

depends_on:

- db

ports:

- 3333:80

environment:

PMA_HOST: db

MYSQL_ROOT_PASSWORD: root

Giải thích :

version: '3.3'

phiên bản của docker-compose

services:

```

db:                                # tên thành phần (tên này được kết hợp với thư mục
                                cha để đặt tên container nếu trong docker-
                                compose không có container_name
image: mysql:5.7                  # image mysql phiên bản 5.7
volumes:
- ./db:/var/lib/mysql            # tạo một thư mục là db để lưu dữ liệu, thư mục
                                này được ánh xạ đến /var/lib/mysql
restart: always
environment:                      # môi trường của mysql
MYSQL_ROOT_PASSWORD: root
MYSQL_DATABASE: wordpress
MYSQL_USER: root
MYSQL_PASSWORD: root

wordpress:
depends_on:                        # có nghĩa là phụ thuộc vào . Ở đây wordpress có
- db                             chung nơi lưu dữ liệu với db ( cùng là thư mục db)
image: wordpress:latest
ports:
- 8008:80                        # cổng 8008 được ánh xạ đến cổng mặc định
restart: always
environment:                     # môi trường của wordpress
WORDPRESS_DB_HOST: db:3306
WORDPRESS_DB_USER: root
WORDPRESS_DB_PASSWORD: root
WORDPRESS_DB_NAME: wordpress

phpmyadmin:                      # giống như trên , đây là tên thành phần
image: phpmyadmin/phpmyadmin     # tên image
restart: always                  # luôn được khởi động lại
depends_on:                      # nơi lưu trữ giống với db
- db
ports:                           # cổng
- 3333:80
environment:                    # môi trường
PMA_HOST: db
MYSQL_ROOT_PASSWORD: root

```

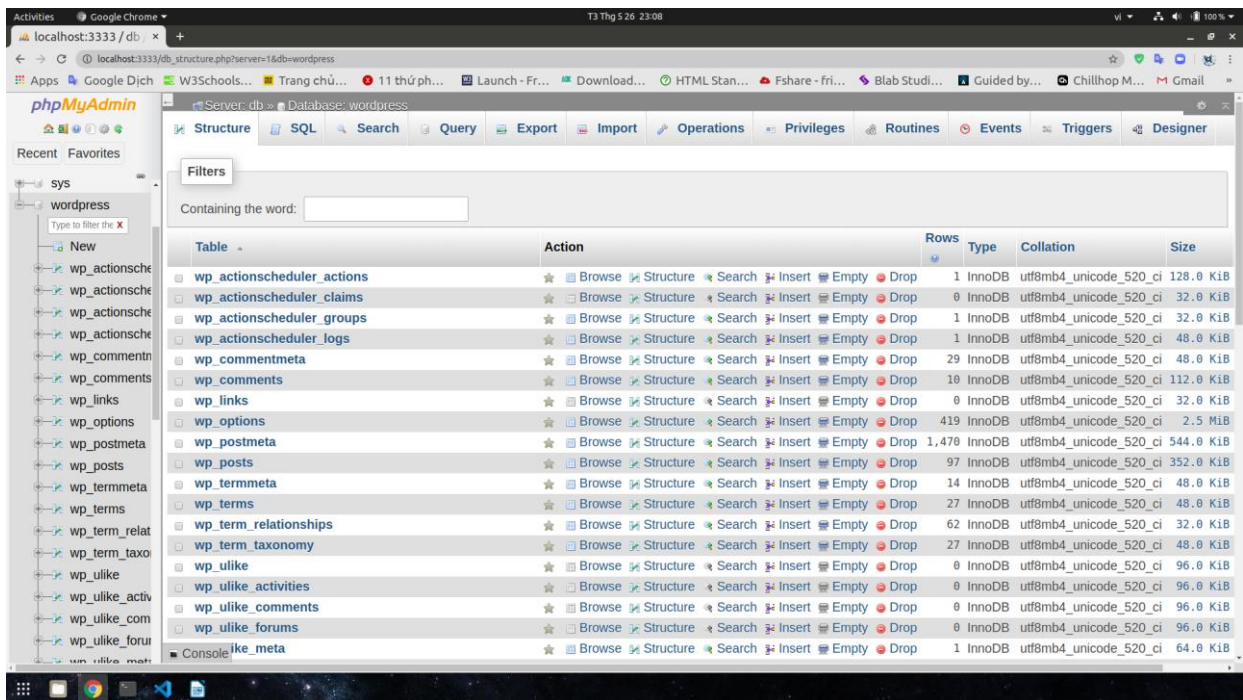
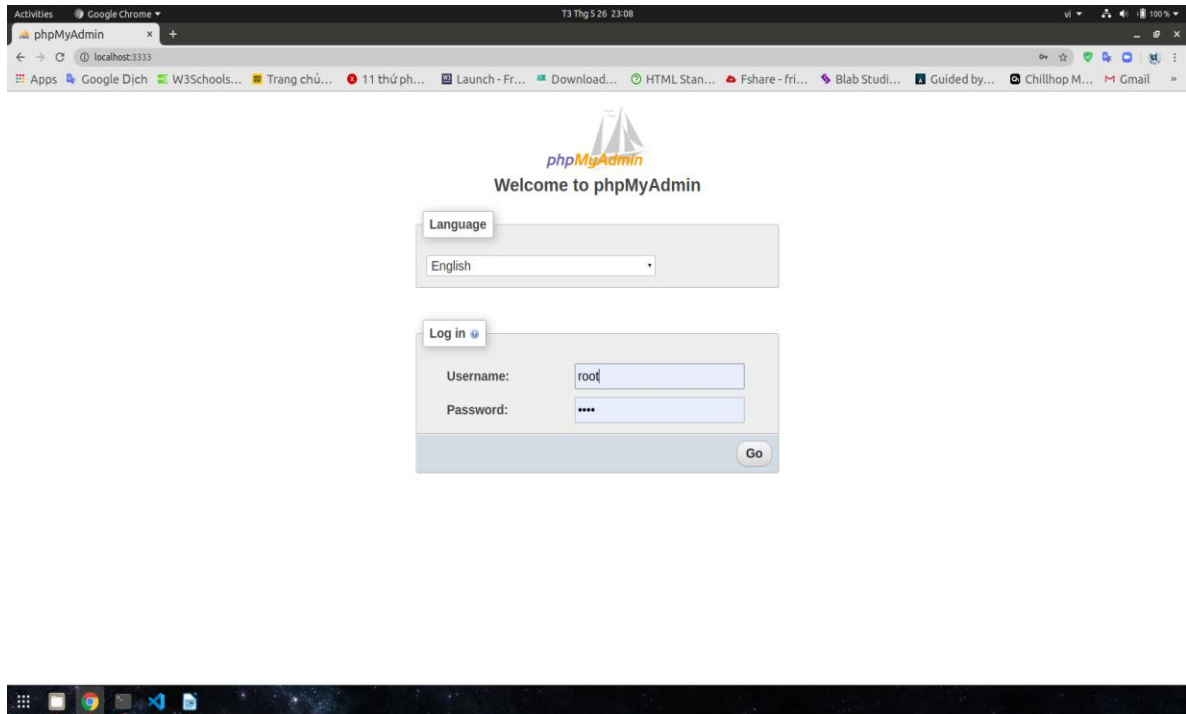
Bước 6: Chạy docker-compose bằng lệnh docker-compose up -d (-d để chạy ngầm, không hiện lên màn hình)

```

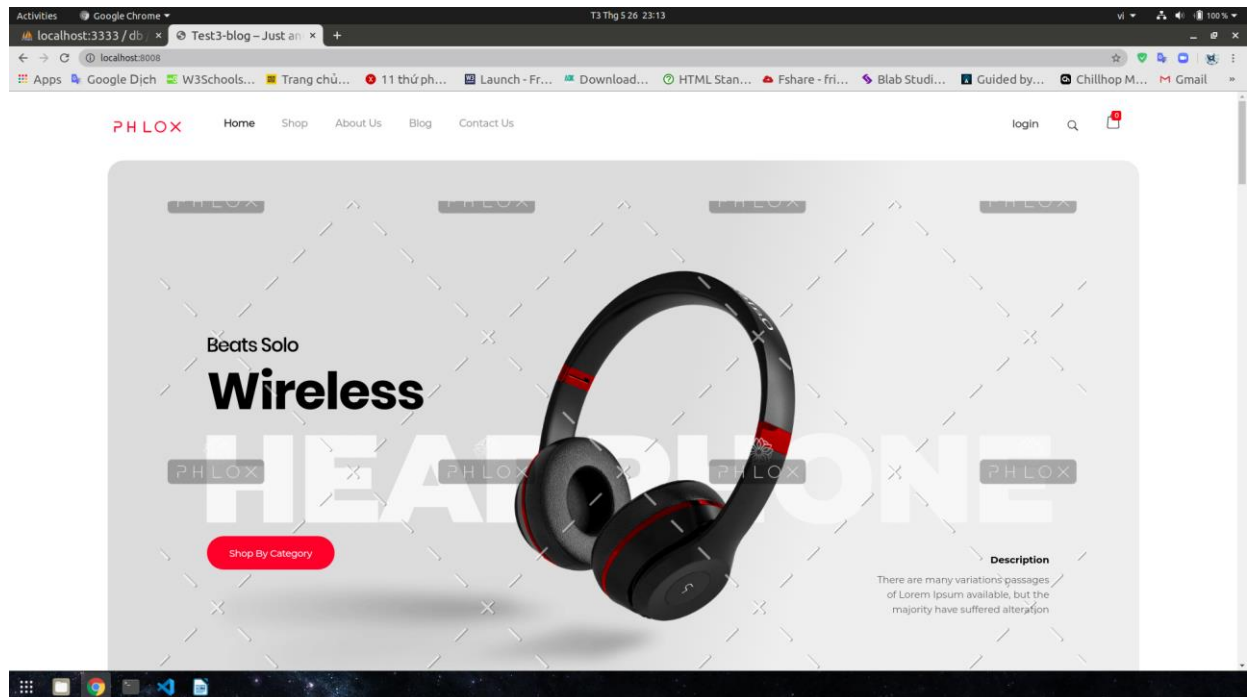
daoninhthai@doffy-Inspiron-7559:~/test3$ docker-compose up -d
Starting test3_db_1 ... done
Starting test3_wordpress_1 ... done
Starting test3_phpmyadmin_1 ... done

```

Bước 7 : lên trình duyệt gõ localhost:3333 để xem container phpmyadmin đã được tạo chưa , sau đó đăng nhập vào và xem database wordpress đã được tạo chưa



Bước 9 : Chuyển đến localhost:8008 để xem trang bằng wordpress , làm theo hướng dẫn tạo tài khoản và tải một theme là sẽ có một trang web tương tự như hình.



d. Hướng dẫn phát triển

Một vài lưu ý khi sử dụng docker :

- Nên tìm hiểu rõ các phiên bản docker , docker-compose được cho phép trước khi cài đặt
- Không nên update docker lên phiên bản mới nhất ngay khi nhận thông báo vì có thể gặp lỗi
- Nếu cần lưu dữ liệu thì nên dùng volume vì nếu xoá container thì sẽ mất dữ liệu
- Không nên dùng latest tag cho image vì có thể về sau build lại có thể bị sai và không chạy được