**DOCKER**

1. **Docker là gì?**

Docker là mã nguồn mỡ giúp tự động triển khai các ứng dụng linux và windown trong các container ảo hóa

Docker cung cấp lớp trừu tượng và tự động ảo hóa trên linux. Docker sử dụng các tài nguyên cô lập của linux như cgroup, kenel, quản lý tiệp cho phép các container chạy độc lập trong một thực thể linux

Link tham khảo: https://docs.docker.com/

1. **Tại sao phải dùng Docker**

Giả sử trước đây khi chạy một website laravel cần phải cài Linux, Apche, Mysql, PHP… và để chạy được ứng dụng phải đáp ứng đầy đủ và đúng version của các service. Docker cho phép đóng gói các dịch vụ và hệ điều hành thành 1 container, chúng ta chỉ cần chạy Dockerfile hoặc docker-composer up thì Docker sẽ tự động build cho chúng ta 1 môi trường tương tự.

1. **Docker compose**
   1. **Định nghĩa**

Là công cụ định nghĩa và khởi chạy multiple-compose application. Trong compose file có thể cấu hình các ứng dụng cần thiết cho môi trường phát triển.

Chỉ với một câu lệnh dev có thể dễ dàng tạo và chạy toàn bộ dịch vụ.

Trong docker compose bạn có thể khai báo tất cả các dịch vụ cần thiết và khởi chạy, hoặc có thể gọi đến từng Dockerfile (nơi chứa các dịch vụ) riêng.

* 1. **Các câu lệnh thường dùng**
     1. **Kiểm tra version**

Docker-compose -v

* + 1. **Chạy file docker**

Docker-compose up -d

- Đối với câu lệnh này docker sẽ trả lại con trỏ trong command

Docker-compose up

- Đối với câu lệnh này docker sẽ không trả lại con trỏ trong command, tạo chúng ta cảm giác đang chạy bị đứng. Nhưng ứng dụng vẫn hoạt động bình thường.

* + 1. **Điều kiện chạy được Docker compose**

Để chạy được Docker compose bắc buộc phải có file có tên docker-compose.yml nằm ở nơi cần chạy ứng dụng

* + 1. **Một số nội dung bên trong Docker compose file**

\* Ví dụ về 1 file Docker-compose cơ bản:

*version: '2'*

*services:*

*web:*

*image: nginx:latest*

*ports:*

*- "80:80"*

*volumes:*

*- ./code:/code*

*- ./site.conf:/etc/nginx/conf.d/default.conf*

*networks:*

*- code-network*

*php:*

*image: php:fpm*

*volumes:*

*- ./code:/code*

*- ./log.conf:/usr/local/etc/php-fpm.d/zz-log.conf*

*networks:*

*- code-network*

\*Thuộc tính:

- Version: phiên bản docker composer

- Serives: dịch vụ chứa các container cần chạy

- Image: là các gói cài đặc sẽ được download và cài đặc trong quá trình chạy

- Ports: cổng của các image, ở đây chúng ta có thể map nó với một port khác khi access từ máy thật. Nếu không khai báo thì khi truy cập phải sửa dụng port mặc định.

+ Ví dụ: chúng ta có thể khai báo - "90:80" thay cho - "80:80" ở trên

Nó có ý nghĩa là nếu chúng ta access từ máy thật với port 90 docker sẽ ánh xạ port 90 thành port 80 trong image nginx.

- Volumes: là nơi ánh xạ folder hoặc file từ máy thật đến máy ảo

Nếu bạn cần tìm hiểu thêm các thuộc tính của nó có thể tham khảo link:

https://docs.docker.com/compose/compose-file/#compose-and-docker-compatibility-matrix

1. **Docker machine**
   1. **Định nghĩa**

Docker machine là 1 máy ảo Docker dùng để build trường ảo.

Docker sẽ liên kết với VM, Vitualbox hoặc Hyper-v của windown (từ phiên bản 8.1 windows đã tích hợp tính năng này) để tạo môi trường.

Khi start được Docker thì Docker sẽ tạo 1 máy ảo mặc định tên default .

Tuy nhiên trong quá trình phát trển dev có thể tạo nhiều máy ảo, mỗi máy ảo dùng cho 1 project để dễ quản lý, hoặc có thể sử dụng 1 máy ảo.

\* Chú ý ip máy ảo là IP máy thật tương tác được, điều kiện máy thật phải ping thấy được máy ảo thì docker mới hoạt động được.

* 1. **Các hàm thông dụng**
     1. **Tạo máy ảo:**

Docker-machine create dev - - driver virtualbox

Chú thích:

* Dev: tên máy ảo
* - -driver: máy ảo mới sẽ dùng driver của virtualbox
* Virtualbox: loại máy ảo
  + 1. **List máy ảo**

Docker-machine ls

* + 1. **Start**

Docker -machine start dev

* + 1. **Stop**

Docker-machine stop dev

* + 1. **Remove**

Docker-machine rm dev

* + 1. **Show IP**

Docker-machine tenmayao ip

**Sử dụng Docker Machine để chạy Docker**

B1: vào đúng thư mục chứa file docker-compose.yml > mỡ command line

B2: Tạo máy ảo

B3: chạy môi trường trên máy ảo với câu lện:

Docker-machine env tenmayao

B4: copy đoạn mã REM ở cuối pass vào con trỏ > enter

B5: chạy: docker-composer up –d

Link tham khảo: Link tham khảo: https://docs.docker.com/machine/get-started/#prerequisite-information

1. **Dockerhub là gì?**

Docker hub cũng giống như Github dùng để lưu trữ các image của chúng ta và download các image khác đã được public từ cộng đồng.

Dockerhub còn có khả năng link kết nối với code repository, build image của bạn và chạy thử trên image đó, lưu trữ các push image…

Link tham khảo: https://hub.docker.com/explore/

1. **Docker Toolbox**

Là gói phần mềm thu gọn của docker. Khi cài đặc đựa chia ra làm 2 phần

Dùng command line mà docker đã xây dựng sẵn, khi khởi chạy nó tạo sẵn cho mình một Docker machine luôn.

Dùng giao diện, giúp chúng ta tương tác cài đặc từ Dockerhub, cấu hình port, thư mục… của các image bằng giao diện

Link tham khảo: https://docs.docker.com/toolbox/toolbox\_install\_windows/

1. **Docker container**
   1. **Định nghĩa**

1 Container là 1 runable instance của image. Bạn có thể create, run, stop, delete or move container sử dụng Docker API or CLI. Bạn có thể kết nối 1 hoặc nhiều network, lưu trữ nó, hoặc thậm chí tạo ra 1 image mới dựa trên trạng thái của nó. Default thì một container được cách ly tương đối với các container và host machine. Bạn có thể control được việc cách ly network, storage, hoặc các sub system khác nằm dưới các containers hoặc các host machine.

* 1. **Các hàm thông dụng**

- List: docker container ls

- Stop: docker stop containerId

- Remove: docker rm containerId

Link tham khảo: https://docs.docker.com/edge/engine/reference/commandline/container/#child-commands

1. **Docker images**
   1. **Định nghĩa**

là các template read-only hướng dẫn cách tạo ra các Docker container. image được sử dụng để đóng gói ứng dụng và các thành phần phụ thuộc của ứng dụng. Image có thể được lưu trữ ở local hoặc trên một registry. Ví dụ ban có thể xây dựng 1 image trên ubuntu, cài Apache server , cũng như cấu hình chi tiết nhưng thứ cần thiết cho viêc running ứng dụng của bạn.

* 1. **Một số hàm thông dụng**

- List: docker images ls

- Remove: docker rmi ImageId

Link tham khảo: https://docs.docker.com/engine/reference/commandline/images/

1. **Docker file**
   1. **Định nghĩa**

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.

* 1. **Cách sử dụng docker file**
     1. **Cách khai báo**

Trong Dockerfile khai báo các lệnh cơ bản như:

FROM

RUN

CMD

……

- FROM: dùng để chỉ ra image được build từ image gốc nào

Vd: FROM Ubuntu

Nếu không chỉ rõ version nó sẽ lấy bản cuối cùng lastest

- RUN: dùng để chạy 1 lệnh nào đó khi build image

Vd: RUN apt-get update

RUN apt-get install curl –y

- CMD: dùng để truyền một lệnh của Linux mỗi khi thực hiện khởi tạo một container từ image (image này được build từ Dockerfile) Có các cách (trong docs nói có 3 cách) sử dụng lệnh CMD

Vd: CMD [‘curl’, ‘info.io’] hoặc CMD curl info.io

Link tham khảo: https://docs.docker.com/engine/reference/commandline/images/

1. **Bài tập demo build 1 con docker chạy môi trường Nginx sử dụng docker compose**

- Mkdir app

- Cd app

- Touch docker-compose.yml với nội dung:

version: '2'

services:

web:

image: nginx:latest

ports:

- "80:80"

volumes:

- ./code:/code

- ./site.conf:/etc/nginx/conf.d/default.conf

networks:

- code-network

php:

image: php:fpm

volumes:

- ./code:/code

- ./log.conf:/usr/local/etc/php-fpm.d/zz-log.conf

networks:

- code-network

networks:

code-network:

driver: bridge

- Tạo file log.conf với nội dung

php\_admin\_flag[log\_errors] = on

php\_flag[display\_errors] = off

- Tạo file site.conf với nội dung:

server {

listen 80;

index index.php index.html;

server\_name docker.test;

error\_log /var/log/nginx/error.log;

access\_log /var/log/nginx/access.log;

root /code;

location ~ \.php$ {

try\_files $uri =404;

fastcgi\_split\_path\_info ^(.+\.php)(/.+)$;

fastcgi\_pass php:9000;

fastcgi\_index index.php;

include fastcgi\_params;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param PATH\_INFO $fastcgi\_path\_info;

}

}

- Tạo folder chưa tên “source” trong folder source tạo file index.php(file test)

- Vào command line gõ: docker-compose up –d

- Config trong host window ip máy ảo với domain là docker.test

- Vào trình duyệt test http://docker.test