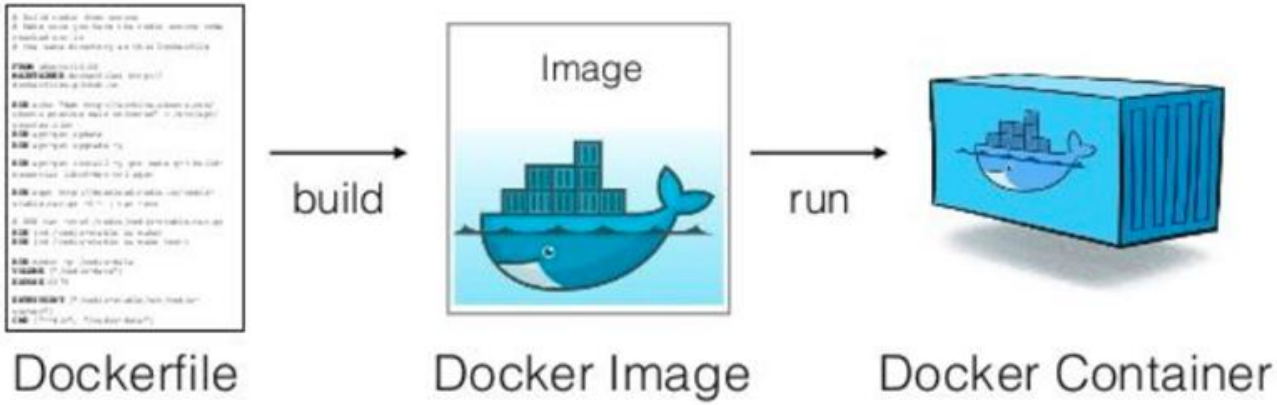
TEL4VN

Docker Image

Agenda

- **Docker Image là gì ?**
- **Docker hub - Registry Image**
- **Xây dựng Docker Image**

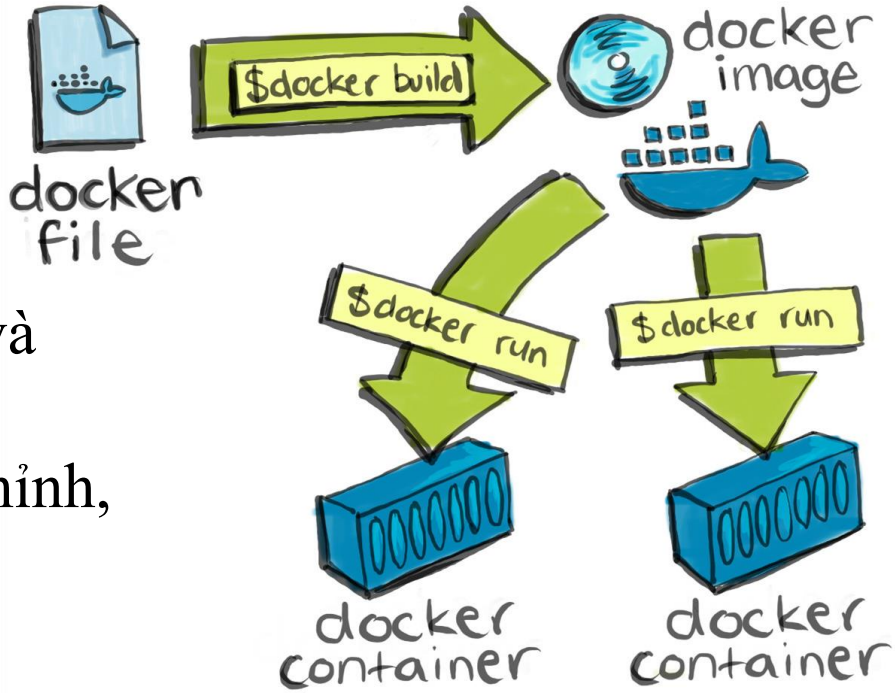
Docker Image là gì ?



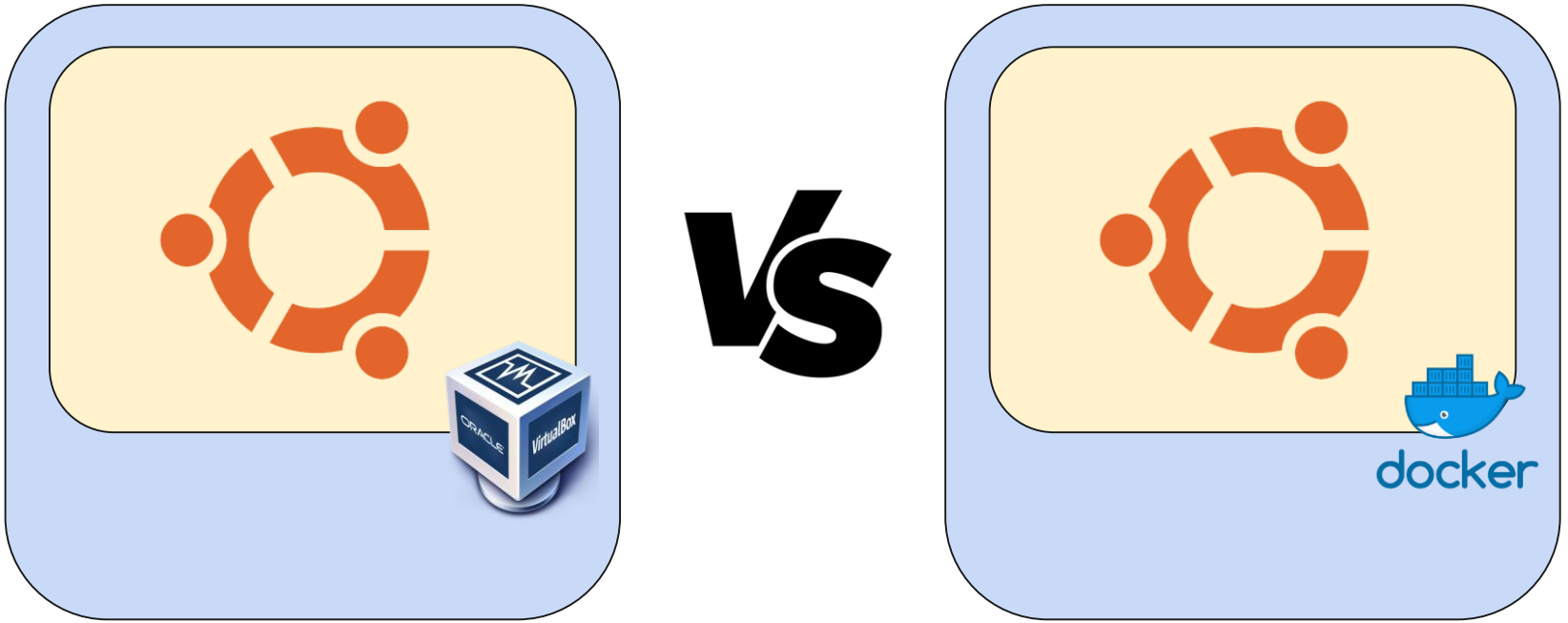
Định nghĩa: "An Image is an ordered collection of root filesystem changes and the corresponding execution parameters for use within a container runtime."

Docker Image

- App binaries và dependencies
- Metadata về dữ liệu của Image và cách vận hành chúng
- Không bao gồm một OS hoàn chỉnh, không chứa kernel và kernel modules (drivers...)



The Ubuntu Image vs Ubuntu OS



Docker Hub Registry Images

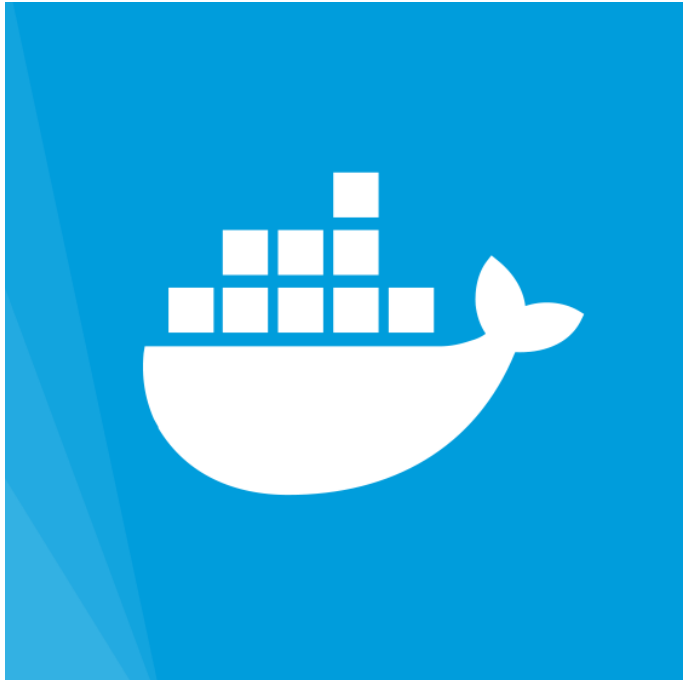
- Public và Docker Registry mặc định
- Docker Hub:

+ Hệ thống quản lý "package" của Containers

```
docker run httpd
```

+ Câu lệnh sử dụng giống git command

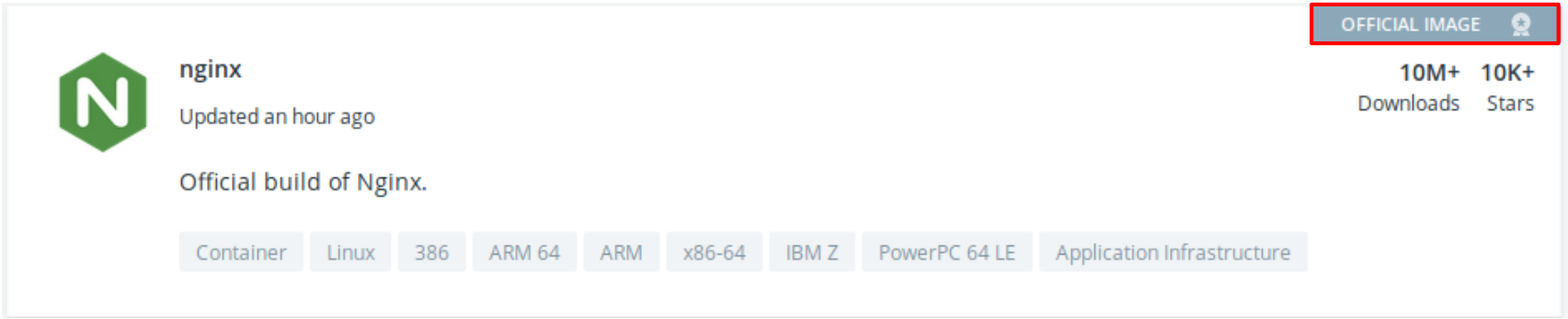
```
docker pull
docker push
docker tag
```



Docker Hub

Docker Hub Registry Images

Official Image:

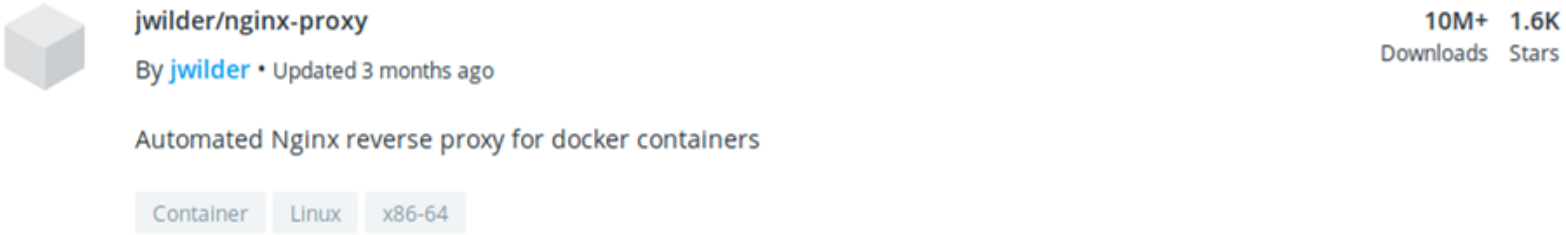


Official Image: docker pull nginx

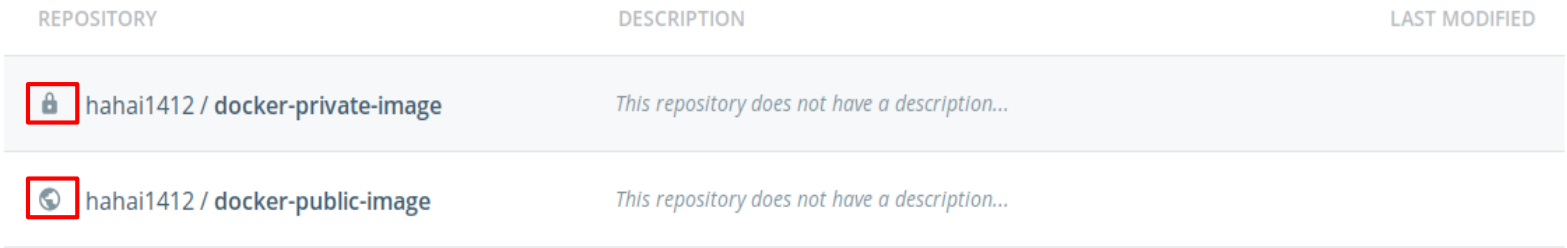
Unofficial Image: docker pull <*account-name*>/nginx

Docker Hub Registry Images

Public Image:



Private Image:



Private Docker Registry Images

TreeScale

JFrog ARTIFACTORY

canister

GitLab

Build Private Docker Registry

Docker Registry

```
docker run -d -p 5000:5000 --net mybridge --restart always --name registry registry:2
```

Docker Registry UI

```
docker run -d -p 8080:80 --net mybridge -e DELETE_IMAGES=true -e
REGISTRY_URL=http://registry:5000 joxit/docker-registry-ui:static
```

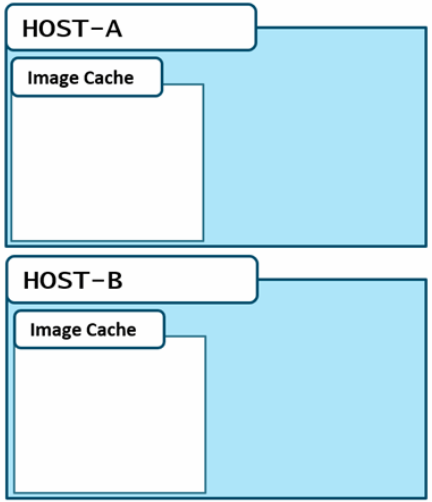
Test command

```
docker tag ubuntu localhost:5000/ubuntu
docker push localhost:5000/ubuntu
```

Docker Image Tagging

1

```
docker build -t web:1
```



Docker Image Tagging

Supported tags and respective `Dockerfile` links

- `1.15.12`, `mainline`, `1`, `1.15`, `latest` (*mainline/stretch/Dockerfile*)
- `1.15.12-perl`, `mainline-perl`, `1-perl`, `1.15-perl`, `perl` (*mainline/stretch-perl/Dockerfile*)
- `1.15.12-alpine`, `mainline-alpine`, `1-alpine`, `1.15-alpine`, `alpine` (*mainline/alpine/Dockerfile*)
- `1.15.12-alpine-perl`, `mainline-alpine-perl`, `1-alpine-perl`, `1.15-alpine-perl`, `alpine-perl` (*mainline/alpine-perl/Dockerfile*)
- `1.16.0`, `stable`, `1.16` (*stable/stretch/Dockerfile*)
- `1.16.0-perl`, `stable-perl`, `1.16-perl` (*stable/stretch-perl/Dockerfile*)
- `1.16.0-alpine`, `stable-alpine`, `1.16-alpine` (*stable/alpine/Dockerfile*)
- `1.16.0-alpine-perl`, `stable-alpine-perl`, `1.16-alpine-perl` (*stable/alpine-perl/Dockerfile*)

Building Images: The Dockerfile Basics

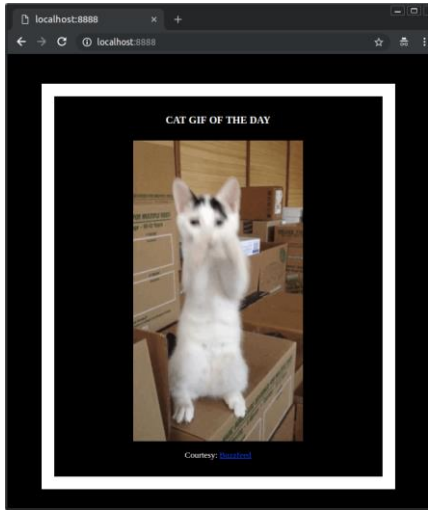
Dockerfile:

- Một dạng text-file đơn giản chứa một danh sách câu lệnh mà Docker client dùng khi tạo ra một Docker Image
- Một cách đơn giản để tự động hóa quá trình tạo ra Image
- Câu lệnh sử dụng trong Dockerfile tư tượng như câu lệnh trong Linux và như cách chúng ta viết một shellscript

Building Images: The Dockerfile Basics

Ví dụ:

```
# all images must have a FROM
FROM python:3-onbuild

# expose these ports on the docker virtual network
# you still need to use -p or -P to open/forward these ports on host
EXPOSE 5000

# required: run this command when container is launched
# only one CMD allowed, so if there are multiple, last one wins
CMD ["python", "./app.py"]
```




```
cd dockerfile-sample-1
docker build -t dockerfile-sample-01:0.1 .
docker run -p 8888:5000 dockerfile-sample-01:0.1
```

Building Images: The Dockerfile Basics

FROM	Image gốc
ENV	Biến môi trường
RUN	Câu lệnh shell
COPY	Sao chép file hay thư mục từ host vào container
WORKDIR	Chỉnh đường dẫn để chạy các câu lệnh RUN, CMD, ENTRYPOINT, COPY và ADD
EXPOSE	Mở port trên container ra Docker Network
CMD	Câu lệnh chạy khi khởi động container

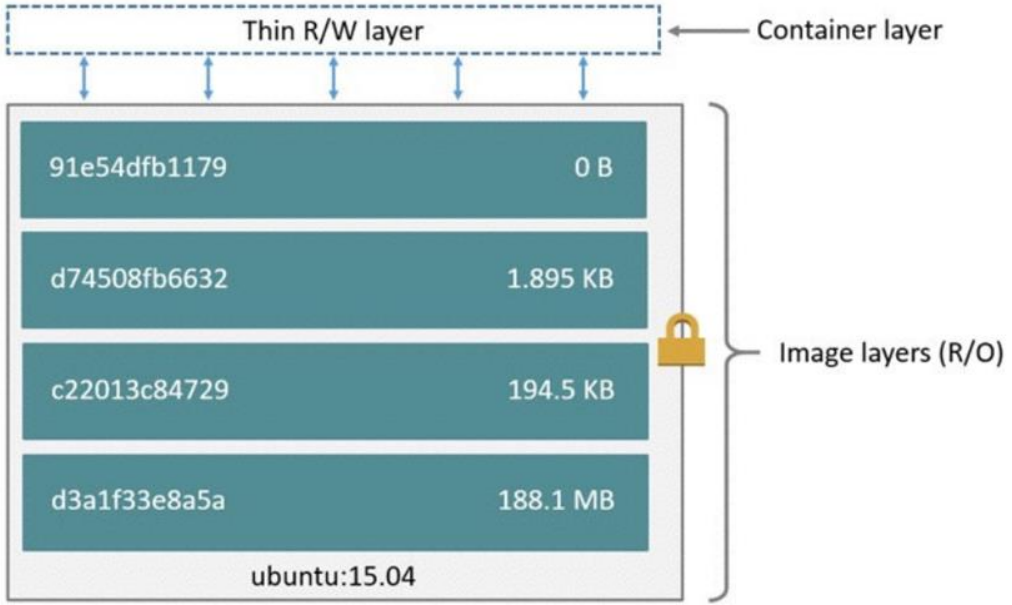
Docker Image Layer

```
# all images must have a FROM
FROM python:3-onbuild



# expose these ports on the docker virtual network
# you still need to use -p or -P to open/forward these ports on host
EXPOSE 5000



# required: run this command when container is launched
# only one CMD allowed, so if there are multiple, last one wins
CMD ["python", "./app.py"]
```

```
Step 1/3 : FROM python:3-onbuild
# Executing 3 build triggers
 ---> Running in 413002714d75
...
Removing intermediate container 413002714d75
 ---> ebc05e370845
Step 2/3 : EXPOSE 5000
 ---> Running in 8680570291f7
Removing intermediate container 8680570291f7
 ---> a1e3d44e6319
Step 3/3 : CMD ["python", "./app.py"]
 ---> Running in 60dcb5e1d718
Removing intermediate container 60dcb5e1d718
 ---> 3cc449c34314
Successfully built 3cc449c34314
Successfully tagged dockerfile-sample-01:0.1
```

Docker Image Layer

Image



Container
(based on ubuntu:15.04 image)

Docker Image Layer Cache

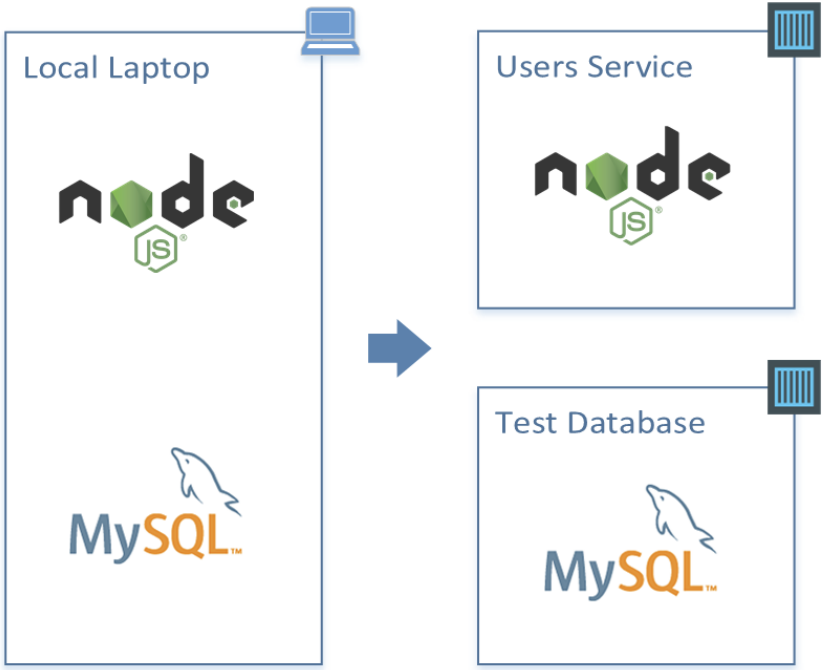
Step 1/3 : FROM python:3-onbuild
Executing 3 build triggers
 ---> Running in 413002714d75
...
Removing intermediate container 413002714d75
 ---> ebc05e370845
Step 2/3 : EXPOSE 5000
 ---> Running in 8680570291f7
Removing intermediate container 8680570291f7
 ---> a1e3d44e6319
Step 3/3 : CMD ["python", "./app.py"]
 ---> Running in 60dcb5e1d718
Removing intermediate container 60dcb5e1d718
 ---> 3cc449c34314
Successfully built 3cc449c34314
Successfully tagged dockerfile-sample-01:0.1

Step 1/3 : FROM python:3-onbuild
 # Executing 3 build triggers
 ---> *Using cache*
 ---> *Using cache*
 ---> *Using cache*
 ---> *ebc05e370845*
Step 2/3 : EXPOSE 5000
 ---> *Using cache*
 ---> *a1e3d44e6319*

Step 3/3 : CMD ["python", "./app.py"]
 ---> *Using cache*
 ---> *3cc449c34314*

Successfully built 3cc449c34314
Successfully tagged dockerfile-sample-01:0.1

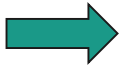
Best practices for building containers



Một ứng dụng trên một container

Best practices for building containers

```
FROM python:3.5
COPY my_code/ /src
RUN pip install my_requirements
```



```
FROM python:3.5
RUN pip install my_requirements
COPY my_code/ /src
```

```
FROM debian:9
RUN apt-get update
RUN apt-get install nginx -y
```

```
FROM debian:9
RUN apt-get update \
    && apt-get install nginx -y
```

Tối ưu hóa cache khi build Docker Image

Best practices for building containers

```
RUN apt-get update \
    && apt-get install --no-install-recommends --no-install-suggests -y \
                                                    nginx=${NGINX_VERSION}
\
                                                    nginx-module-
xslt=${NGINX_VERSION} \
                                                    nginx-module-
geoip=${NGINX_VERSION} \
                                                    nginx-module-image-
filter=${NGINX_VERSION} \
                                                    nginx-module-
njs=${NJS_VERSION} \
                                                    gettext-base \
&& rm -rf /var/lib/apt/lists/*
```

Tối ưu hóa Docker Image Size:

- Loại bỏ những tools không cần thiết
- Loại bỏ những file hoặc thư mục không sử dụng
- Loại bỏ những thư viện không sử dụng của ứng dụng
- Sử dụng các OS có dung lượng nhỏ (Alpine)

Best practices for building containers

`1.15.12` , `mainline` , `1` , `1.15` , `latest` (*mainline/stretch/Dockerfile*)

`1.15.12-perl` , `mainline-perl` , `1-perl` , `1.15-perl` , `perl` (*mainline/stretch-perl/Dockerfile*)

`1.15.12-alpine` , `mainline-alpine` , `1-alpine` , `1.15-alpine` , `alpine` (*mainline/alpine/Dockerfile*)

`1.15.12-alpine-perl` , `mainline-alpine-perl` , `1-alpine-perl` , `1.15-alpine-perl` , `alpine-perl` (*mainline/alpine-perl/Dockerfile*)

`1.16.0` , `stable` , `1.16` (*stable/stretch/Dockerfile*)

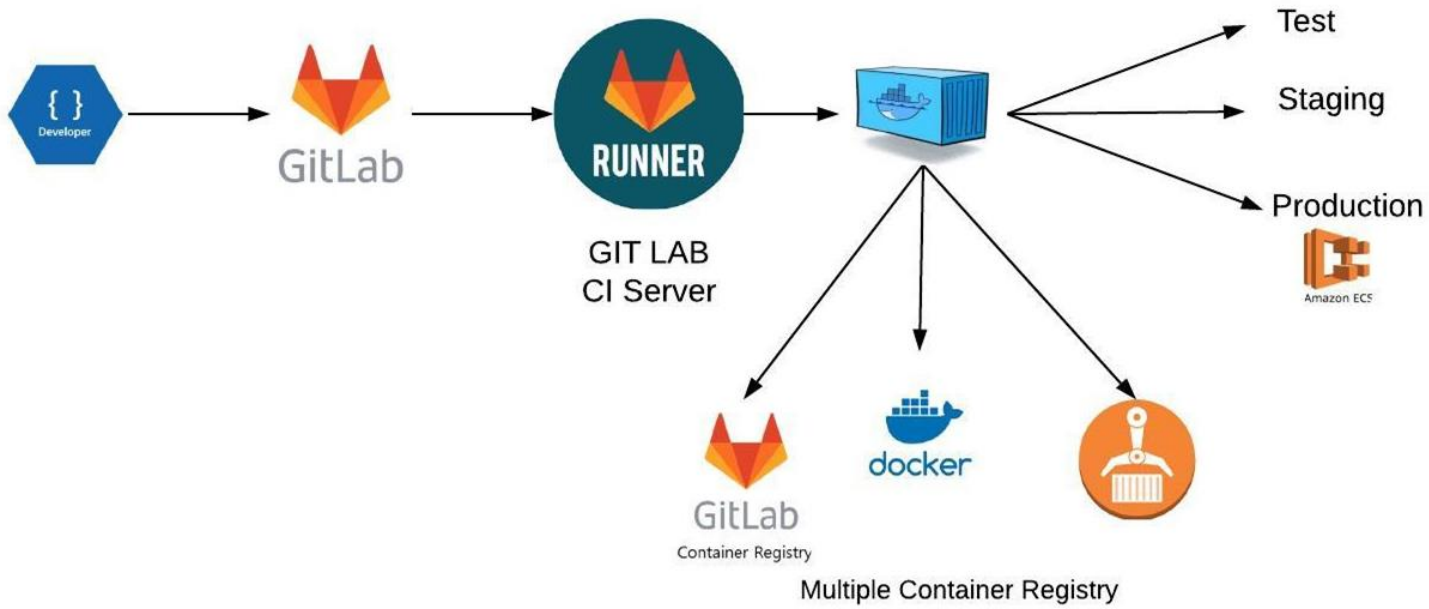
`1.16.0-perl` , `stable-perl` , `1.16-perl` (*stable/stretch-perl/Dockerfile*)

`1.16.0-alpine` , `stable-alpine` , `1.16-alpine` (*stable/alpine/Dockerfile*)

`1.16.0-alpine-perl` , `stable-alpine-perl` , `1.16-alpine-perl` (*stable/alpine-perl/Dockerfile*)

Sử dụng tagging một cách hợp lý

Best practices for building containers



Cẩn thận khi sử dụng Public Image và Public Registry

Bài tập: Build Your Own Image

Sử dụng app Node.js có sẵn và Containerize app đó
Dùng Dockerfile:

- Build Image
- Test Image sau khi build
- Push lên Docker Hub
- Remove Image ở local
- Run lại Container với Image vừa push

Chi tiết trong dockerfile-assignment-1/Dockerfile
Sử dụng Image Nodejs alpine 6.x bản official
Kiểm tra bằng http://localhost


Q
A