Học về docker

1. Khái niệm chung và các lệnh cơ bản về container:

- Khái niệm: Docker là một nền tảng giúp đóng gói sản phẩm => khi đưa cho client, khách hàng chạy container trên docker là chạy được sản phẩm (không cần cài thêm gì cả) => client chạy sản phẩm mà không cần biết source code ntn

Lưu ý: docker yêu cầu chạy trên linux => sử dụng wsl2 để có thể giả lập môi trường linux

- Docker Hub: nền tảng các người dùng sử dụng docker chia sẻ file cho nhau

- Container: package code và dependencies lai với nhau => multiple container có thể chạy trên 1 machine và share kernel với nhau. Container không sử dụng toàn bộ operating system, chúng dùng chung OS của máy chính => nhanh hơn với VM

- Virtual machine: hệ thống máy ảo, có thể dùng hypervisor để chạy nhiều VM trên 1 máy. Tuy nhiên VM copy full OS của máy

A screenshot of a diagram

AI-generated content may be incorrect.

Container vs VM applications

- Một số command cơ bản với docker:

* Docker –-version: kiểm tra version hiện tại docker
* Docker ps: kiểm tra các container đang chạy
* Docker ls: list các command trong docker
* Docker images: kiểm tra các images ta đang có locally
* Docker container run “image name”: “tag” : chạy một image
* Docker container run -p 8080:80 -d “images name”: tag: map cổng local host với port 80 trên images name (port phải là số có 4 chữ số và lớn hơn 1000, có thể map nhiều host vào một port)
* Docker container stop “container id”/ “container name”: dừng một container
* Docker container ls: xem tất cả các container đang chạy (tương tự docker ps)
* Docker container rm “container id”/ “container name”: remove một container (lưu ý không thể remove running container)
* Export a = “string”: gắn một chuỗi string vào a

- Image: là một template tạo môi trường cho sản phẩm, bao gồm tất cả những gì app cần có để chạy: os system, software environment, app code

- Container: là instance của image

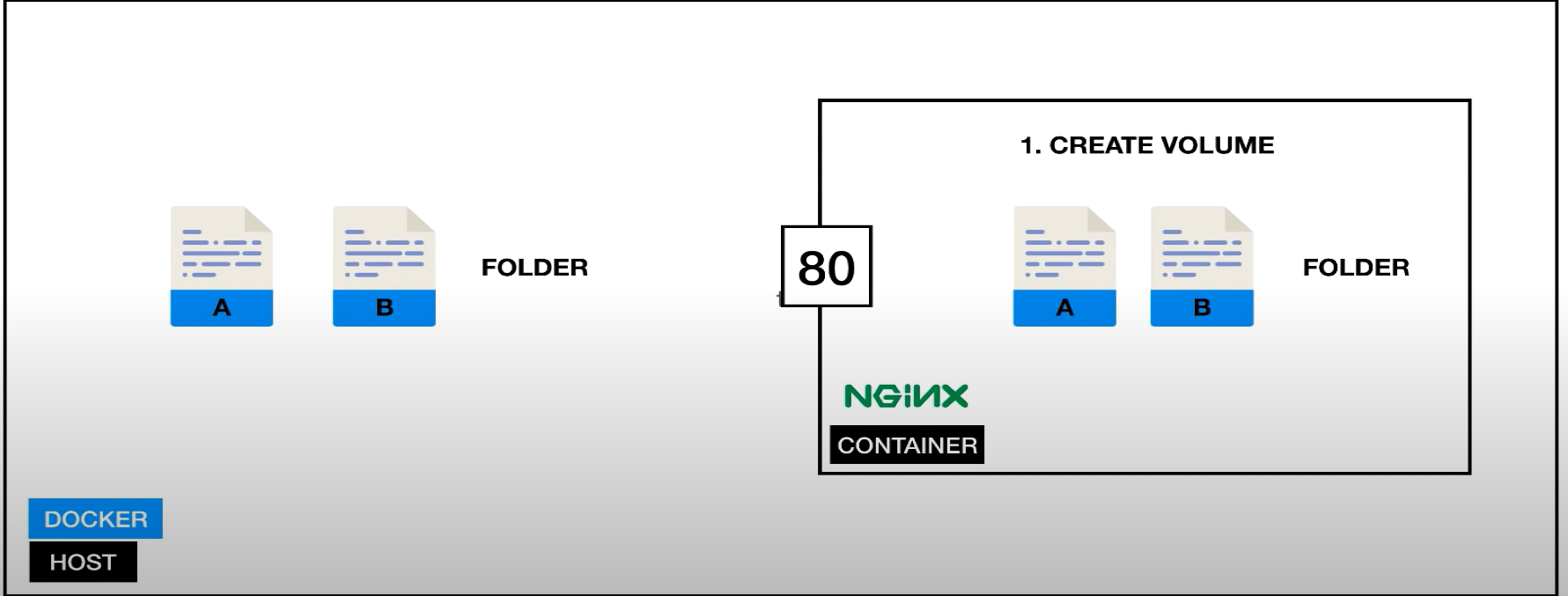
Lưu ý: khi ta chạy cmd, ta muốn xem các lệnh hoàn chỉnh có thể xây dựng từ command ta đang nhập, thêm - - help vào sau command ta đang nhập

- Khi một container bị stop, ta có thể chạy lại container đó sử dụng docker start container ID và hoặc container name của nó.

- sử dụng %A% cho command A nếu A là parameter của command B

2. Volumes:

- Giúp chia sẻ dữ liệu giữa host (docker) và container



Khi ta thêm một file vào host, nó cũng sẽ thêm vào container và ngược lại

Lệnh khởi tạo một container và thêm volumes:

* Docker container run -v “path\_to\_folder\_host”: “path\_to\_folder\_nginx” (path to folder nginx sẽ là thư mục trong container lưu các file được volume)

=> Mọi file trong folder sẽ được gắn vào nginx

* Docker exec -it “tên container” bash: mở bash trong container (nếu container chạy dựa trên linux)

Lệnh tạo volume giữa các container với nhau:

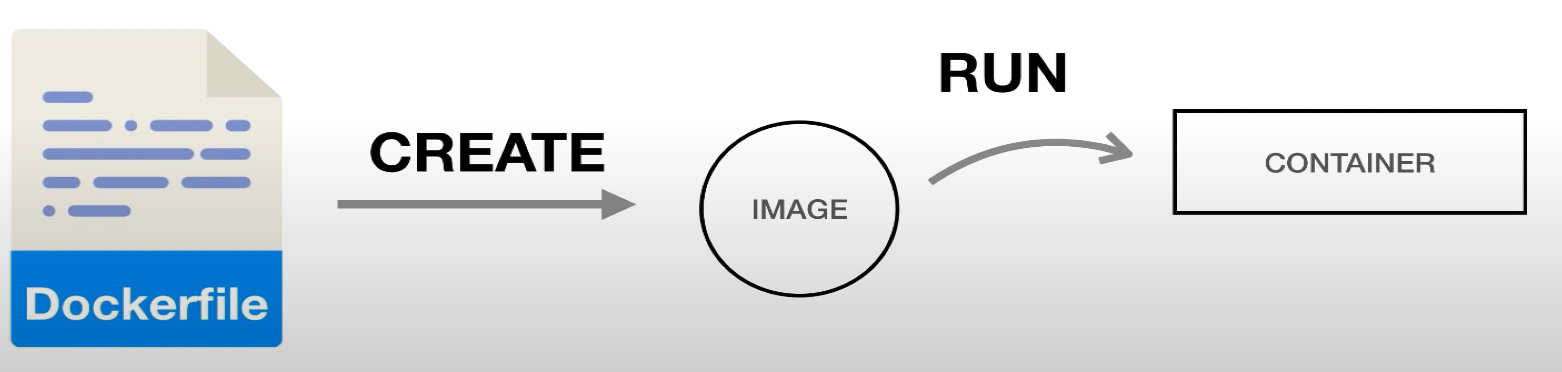
* Docker container run —volumes-from “tên container chứa file ta muốn tạo volume)

Lệnh xem chi tiết thông tin của một container:

* Docker container inspect “tên container”
* Docker container logs -f “tên container”: xem thông tin về logs của một container (xem tần suất truy cập ntn và debug tốt hơn)

3. Docker file:

- Docker file cho phép ta tạo một images (là một template gồm OS, app code, software environment)



- Lệnh chạy một dockerfile để tạo images:

* Docker build –tag “name:tag” . : gắn name và tag vào cho docker

- Các instruction dùng để build một dockerfile:

* From “base image name”: base image của image ta build (bắt buộc phải có)
* ADD “source folder” “destination folder”: add các file của host vào trong images
* WORKDIR “destination folder”: thiết lập thư mục làm việc trong container (destination của các lệnh add copy entrypoint,run, cmd,...)
* CMD “cmd command”: thực hiện một command
* RUN “build command”: thực hiện một build command

.dockerignore: file list tất cả các file ta sẽ loại bỏ ra khi thực hiện lệnh add, copy,.... (các file không cần đưa vào container)

4. Caching:

- Cơ chế giúp cho dockerfile không phải chạy lại các lệnh từ lần chạy trước

- Trong dockerfile, các lệnh nào đã chạy lần trước mà giống hệt ở lần chạy này thì sẽ được cache (không cần chạy lại nữa) => khi add, ta ưu tiên để các file ít thay đổi ra một lệnh add riêng => có thể kph chạy lại nữa khi ta thay đổi sau này

5. Alpine:

- Là phiên bản Linux Distribution (thường sử dụng cho Docker Images)

- Các Images được cấu hình theo alpine có size nhỏ hơn => chạy nhanh hơn rất nhiều (được tag là alpine)

Mẹo: khi pull image, ta không nên dùng tag latest mà nên chỉ rõ version => kiểm soát lỗi và phiên bản tốt hơn

6. Tagging:

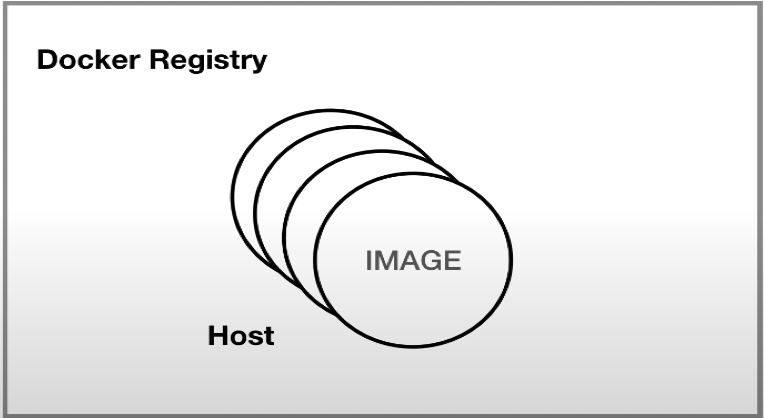
- Lệnh để tạo tag cho một phiên bản images của mình:

* **Docker tag website:latest website:2**: gắn image với tag latest thành tag 1

=> khi ta tạo một phiên bản website mới, lại gắn tag latest và tag 2

7. Docker Registry:

- Là 1 server giúp ta lưu và triển khai images. Thực tế, các images chính thức được host tại registry tại docker hub (server chính thức, máy chủ nằm ở nước ngoài). Docker Registry cho phép ta chia sẻ images của mình với các người khác



- Các server docker registry hiện nay:

* Docker Hub (registry chuyên lưu các official images)
* [quay.io](http://quay.io)
* Amazon ECR

- Lệnh upload một images của mình lên docker registry (để có thể share với người khác):

* Docker push “username/tên images:tag”

Lưu ý: tên images như thế nào sẽ up lên repo có tên giống hệt như vậy

7. Network:

- Network: là một cụm các container. Các container trong cùng một network sẽ kết nối với nhau qua port nội bộ hoặc name chứ không cần port mapping ra ngoài

- Các lệnh với network:

* Chạy một container và gắn vào network: docker container run –network [network name] [image:tag]
* Command tạo network: **docker network create -d network [type] [network name]**
* Xem chi tiết một network: **docker network inspect [network name]**
* Gán thêm vào network: **docker network connect [network name] [container name]**
* Bỏ ra khỏi network: **docker network disconnect [network name] [container name]**

- Có các loại network sau:

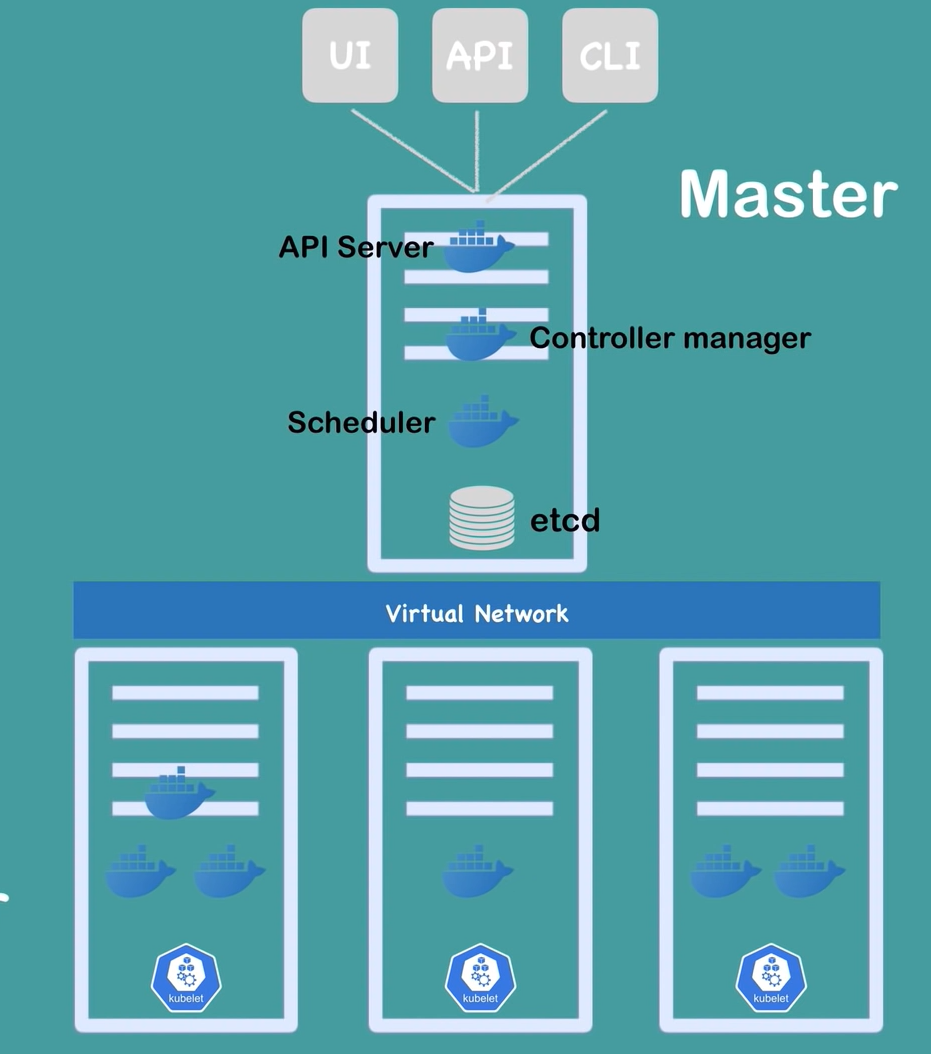
* Bridge network: chỉ truy cập được container trong network nếu nó được map ra ngoài.
* Host network: có thể truy cập trực tiếp vào container trong network mà không cần map ra ngoài
* None network: container chạy biệt lập (không thể kết nối container khác)
* Overlay network: dùng cho docker swarm hoặc k8s. Container trên nhiều máy khác nhau có thể giao tiếp được với nhau nếu cùng 1 mạng.

8. Docker compose.yaml:

- Đây là file có tác dụng tự động khởi tạo và chạy container (khác với dockerfile dùng để tạo docker images)

Kubertenes

- Là công cụ giúp quản lý sự hoạt động của các containers (tăng availability và scalability) của hệ thống hoạt động bởi nhiều containers



Cấu trúc của 1 cụm k8s

- Master: Lưu các process (container) quan trọng như:

* API Server: tương tác với các UI/UX, API Call, CLI từ người dùng (entrypoint)
* Controller Manager: Kiểm soát trạng thái của các container
* Scheduler: schedule hoạt động của các containers trên các cubelets (nodes)
* Etcd: lưu các states, tham số của kubernetes cluster qua các thời điểm khác nhau => backup

- Virtual network: kênh trao đổi giữa master node và worker node

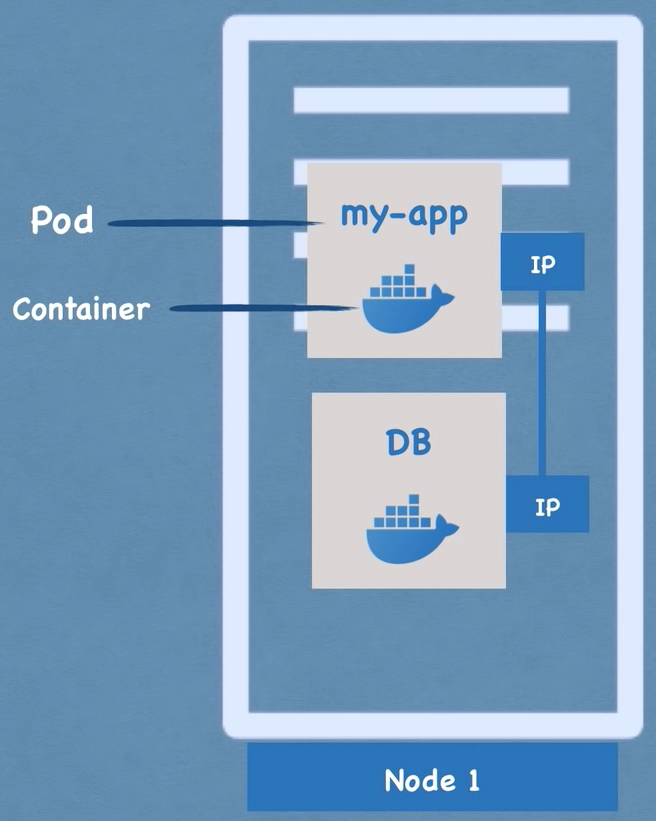
- Workernode: chứa các containers thực thi chương trình

=> Thông thường, trong các hệ thống sẽ có 1-2 backup masternodes tại nó quan trọng

1. Pod, services, ingress:

- Pod: đơn vị nhỏ nhất trong k8s, mỗi pod chứa 1 container

- Các pod trao đổi với nhau dựa trên IP Address. Mỗi pod có 1 IP Add riêng, bản chất là static => khi 1 pod thay đổi ở 1 chỗ, đ/c IP Address cũng bị thay đổi => bất tiện



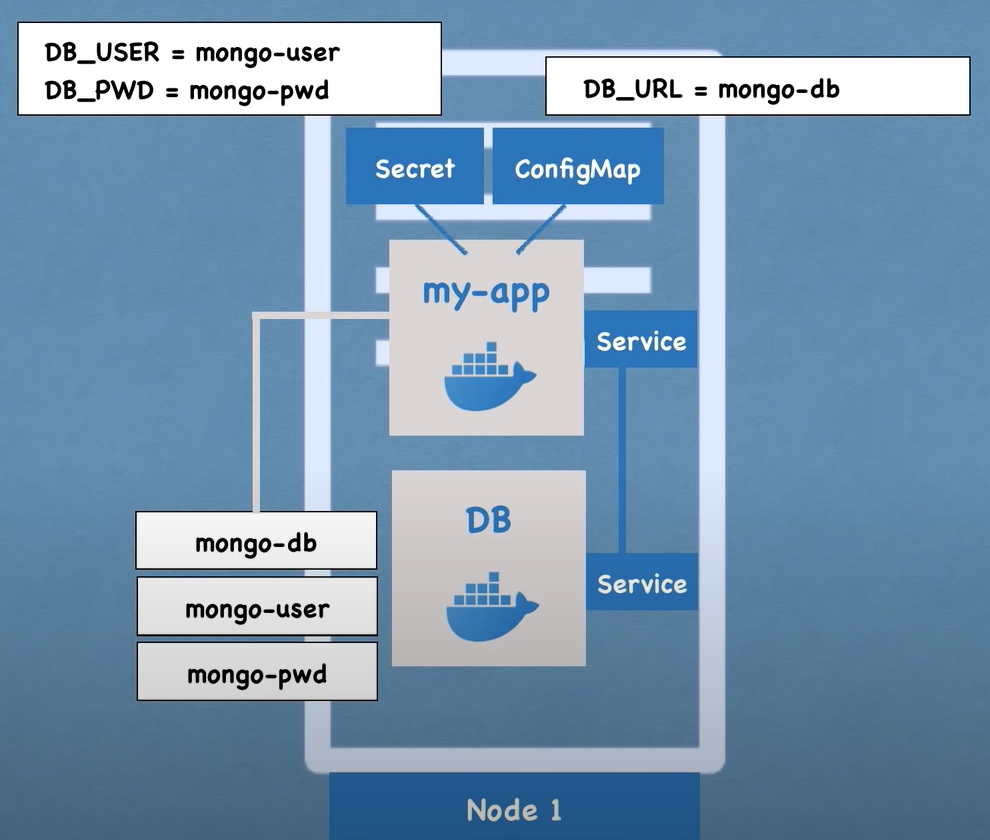
Communicate with IP (internal address)

- Service: dynamic IP Address để assign cho các pod. Khi ta thay đổi pod ở một chỗ, IP Address thay đổi nhưng service không đổi => tiện hơn

- Ingress: cổng vào, định hướng các request đến đúng pod mà không cho pod tiếp xúc với bên ngoài

Để tiện cho việc kết nối đến database, pod của container có 2 file:

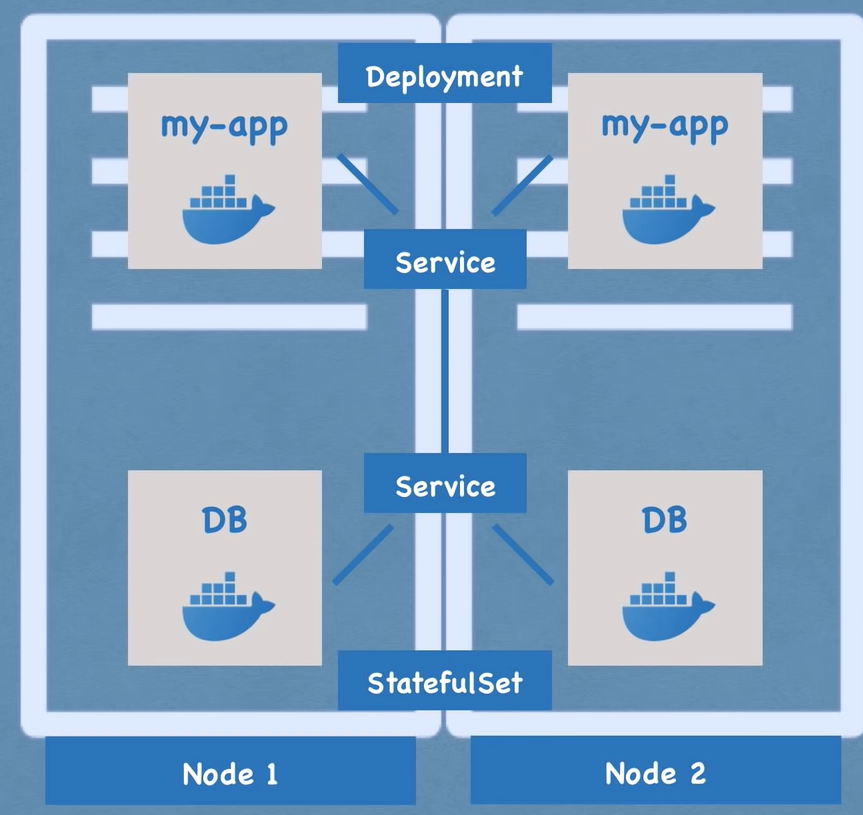
* ConfigMap: File lưu các thông tin config(biến môi trường) của hệ thống (database\_url)
* Secret: file lưu các thông tin bảo mật (database password) được encode dưới base64



- Volumes: tác dụng tương tự như pod. Tuy nhiên nó thực sự được assign bộ nhớ ở máy host hoặc một remote machine => giữ liệu vẫn còn nếu hệ thống restart => dùng để lưu database

Lưu ý: k8s chỉ lưu data ảo (tức restart hệ thống là mất), ta sử dụng bộ nhớ thì được coi là storage ngoài k8s

- Tương tự như các hệ thống distributed khác, K8S hỗ trợ fault tolerance bằng cách tạo bản sao của các pod => Deployment: tạo bản sao của các pod; StatefulSet: tạo bản sao của volumes, tuy nhiên có quản lý truy cập vào storage rõ ràng.



2. Minikube and Kubectl:

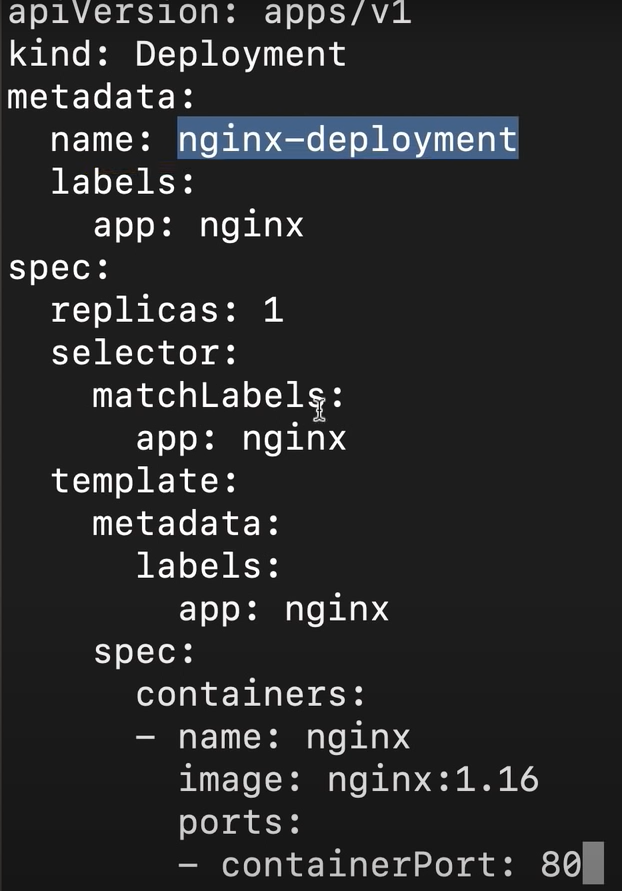
- Minikube: Mô phỏng cả 1 cụm k8s ở trên 1 máy ảo => thích hợp cho việc kiểm thử, test cụm k8s

- Kubectl: Giao diện command line cho phép người dùng tương tác, edit cụm k8s thông qua k8s API Server ở master node.

- Một số lệnh cơ bản với kubectl:

* Minikube start: lệnh khởi động minikube. Phải khởi động cái này thì kubectl mới trỏ đến được
* Kubectl get nodes: get status về node trong kubectl
* Kubectl get “Component”: liệt kê các component thuộc loại cần tìm đang chạy trong k8s
* Kubectl get pod: liệt kê tất cả các pod đang chạy trong k8s
* Kubectl get service: in ra các service đang có trong k8s
* Kubectl create “component” NAME : tạo 1 component và đặt tên là NAME trong k8s
* kubectl create deployment nginx-depl --image=nginx:latest: tạo một pod với container là instance của image nginx và đặt tên là nginx-depl
* Kubectl logs “tên pod”: xem log của container đang chạy trong pod (để debugging)
* Kubectl describe pod “tên pod”: in tất cả các thông tin (tên, ips, images,state changes, ... của pod)
* Kubectl exec -it “podname” – bin/bash: vào bash của container đang chạy trong pod
* Kubectl apply -f “tên file”: tạo một component với cấu hình được viết trước trong file yaml hoặc json (khi ta thay đổi file config, chạy lại lệnh này sẽ tự động update các pod theo config)
* Minikube service “tên của external service”: khởi động một external service nhận request bên ngoài

- File yaml: lưu thông tin dạng key - value tương tự định dạng json. Tuy nhiên dùng intend cho định hình cấu trúc dữ liệu, dùng dấu gạch đầu dòng cho list, key:value cho các cặp khóa: giá trị



File YAML config cấu hình của 1 deployment

**Node: học về quy tắc lấy đường dẫn khi import module trong python**