

Bộ Giáo Dục Và Đào Tạo
Trường Đại Học Ngoại Ngữ - Tin Học Thành Phố Hồ Chí Minh
Khoa Công Nghệ Thông Tin



MÔN HỌC: ĐIỆN TOÁN ĐÁM MÂY

ĐỀ TÀI: TÌM HIỂU VỀ KUBERNETES

Giảng viên: ThS. Cao Tiến Thành

Thành viên:

Trần Quang Khải – MSSV: 22DH114583

Tp. Hồ Chí Minh, Ngày ... tháng ... năm 2025

LỜI CẢM ƠN

Em xin chân thành gửi lời cảm ơn đến Thầy ThS. Cao Tiến Thành – giảng viên phụ trách môn *Điện toán đám mây* – đã tận tình giảng dạy, truyền đạt kiến thức và hướng dẫn em trong suốt quá trình học tập cũng như thực hiện báo cáo môn học.

Nhờ sự hướng dẫn tận tâm, cùng những bài giảng súc tích, dễ hiểu của Thầy, em đã có cơ hội tiếp cận và ứng dụng nhiều kiến thức thực tiễn về công nghệ điện toán đám mây – một lĩnh vực hiện đại và có tính ứng dụng cao trong thực tế.

Báo cáo này là kết quả từ quá trình học tập nghiêm túc và vận dụng kiến thức đã học, tuy nhiên do thời gian và kiến thức còn hạn chế, bài báo cáo chắc chắn vẫn còn nhiều thiếu sót. Em rất mong nhận được sự góp ý từ Thầy để em có thể hoàn thiện hơn trong những lần sau.

Một lần nữa, em xin chân thành cảm ơn Thầy và kính chúc Thầy nhiều sức khỏe, thành công trong công việc giảng dạy và nghiên cứu.

NHẬN XÉT CỦA GIẢNG VIÊN

MỤC LỤC

CHƯƠNG I. GIỚI THIỆU CHUNG	7
1. Lý do trọng đề tài	7
2. Mục tiêu của đề tài	7
3. Phạm vi và đối tượng nghiên cứu	7
4. Phương pháp thực hiện	7
5. Cấu trúc báo cáo.....	7
CHƯƠNG II. CƠ SỞ LÝ THUYẾT VÀ CÀI ĐẶT	9
1. Kubernetes là gì?.....	9
2. Tại sao bạn cần kubernets và nó có thể làm gì?.....	9
3. Những khái niệm cơ bản trong Kubernetes	10
4. Ưu và nhược điểm của Kubernetes[5]	12
4.1. Ưu điểm.....	12
4.2. Nhược điểm.....	12
5. Cài đặt Kubernetes	13
III.Triển khai dịch vụ (Static web).....	22
Tài liệu tham khảo.....	29

DANH MỤC HÌNH ẢNH

Hình 1. Kubernetes	9
Hình 2. MasterNode _ WorkerNode	10
Hình 3. Thành phần của Kubernetes.....	11
Hình 4. Mô hình triển khai k8s	15
Hình 5. Nội dung hosts	16
Hình 6. Nội dung file containerd.conf	17
Hình 7. Hoàn thành cấu hình cơ bản.....	19
Hình 8. Hoàn thành cấu hình k8s-master.....	20
Hình 9. Hoàn thành cấu hình k8s-worker1	20
Hình 10. Hoàn thành cấu hình k8s-worker2	21
Hình 11 Nội dung file web.....	22
Hình 12 Nội dung Dockerfile	23
Hình 13 Image trên Docker Hub.....	23
Hình 14 Các node của cụm K8s.....	24
Hình 15 Thư mục quản lý web trên K8s.....	24
Hình 16 Nội dung file deployment	25
Hình 17 Các pod được khởi tạo trên cụm k8s	25
Hình 18. Nội dung file service YAML.....	26
Hình 19. Kiểm tra service	26
Hình 20. Thông tin về các máy chủ k8s.....	27
Hình 21. Static web	27
Hình 22 mysql-pv.yaml.....	29
Hình 23 wordpress-pv.yaml	30
Hình 24 mysql-deployment.yaml.....	30
Hình 25 wordpress-deployment.yaml.....	31
Hình 26 Apply file yaml	31
Hình 27 Truy cập web thành công	32
Hình 28 Tạo user mới	33
Hình 29 Database mysql	34
Hình 30 Server backup.....	35
Hình 31 Docker-compose file	35
Hình 32 docker-compose up -d.....	36
Hình 33 Web Minio.....	36
Hình 34 Tạo bucket.....	37
Hình 35 Setup velero.....	37
Hình 36 credentials-velero	37

Hình 37 Kết nối velero đến minio.....	38
Hình 38 Backup	38
Hình 39 Minio backup	38
Hình 40 Restore	39
Hình 41 Daily backup	39

DANH MỤC BẢNG

Bảng 1 So sánh Kubernetes với các công cụ khác	13
Bảng 2 So sánh các cách cài đặt Kubernetes	14

I. GIỚI THIỆU CHUNG

1. Lý do chọn đề tài

Trong thời đại công nghệ số phát triển mạnh mẽ, các doanh nghiệp ngày càng đòi hỏi hệ thống công nghệ thông tin phải có khả năng mở rộng linh hoạt, tự động hóa và đáp ứng nhanh chóng với nhu cầu thay đổi. Các nền tảng điện toán đám mây (Cloud Platform) đã trở thành giải pháp tối ưu giúp doanh nghiệp tiết kiệm chi phí đầu tư, tăng hiệu quả vận hành và nâng cao khả năng cạnh tranh. Trong số các công nghệ nổi bật hỗ trợ triển khai cloud-native applications, Kubernetes đã chứng minh được vai trò quan trọng với khả năng quản lý container hiệu quả, tự động hóa việc triển khai và mở rộng ứng dụng.

Vì lý do đó, nhóm chúng em chọn đề tài "Cloud Platform - Kubernetes" nhằm tìm hiểu cách thức hoạt động, lợi ích và quy trình triển khai hệ thống Kubernetes trong môi trường cloud thực tế.

2. Mục tiêu của đề tài

- Tìm hiểu tổng quan về kiến trúc và cơ chế hoạt động của Kubernetes.
- Phân tích lợi ích của việc sử dụng Kubernetes trong môi trường cloud.
- Thực hành triển khai cụm Kubernetes cơ bản trên nền tảng cloud (GCP, AWS hoặc local).
- Đánh giá khả năng mở rộng, tự phục hồi và tính linh hoạt khi sử dụng Kubernetes.

3. Phạm vi và đối tượng nghiên cứu

- **Đối tượng nghiên cứu:** Nền tảng Kubernetes và mô hình triển khai ứng dụng container hóa trên cloud.
- **Phạm vi nghiên cứu:** Tập trung vào cài đặt, quản lý và triển khai web trên cụm Kubernetes cơ bản.

4. Phương pháp thực hiện

Nhóm áp dụng phương pháp học qua dự án (project-based learning), kết hợp giữa nghiên cứu tài liệu, cài đặt thực tế và đánh giá hiệu quả. Các công cụ sử dụng bao gồm Google Cloud Platform (GCP), Minikube, kubectl, Docker, v.v.

5. Cấu trúc báo cáo

Báo cáo gồm:

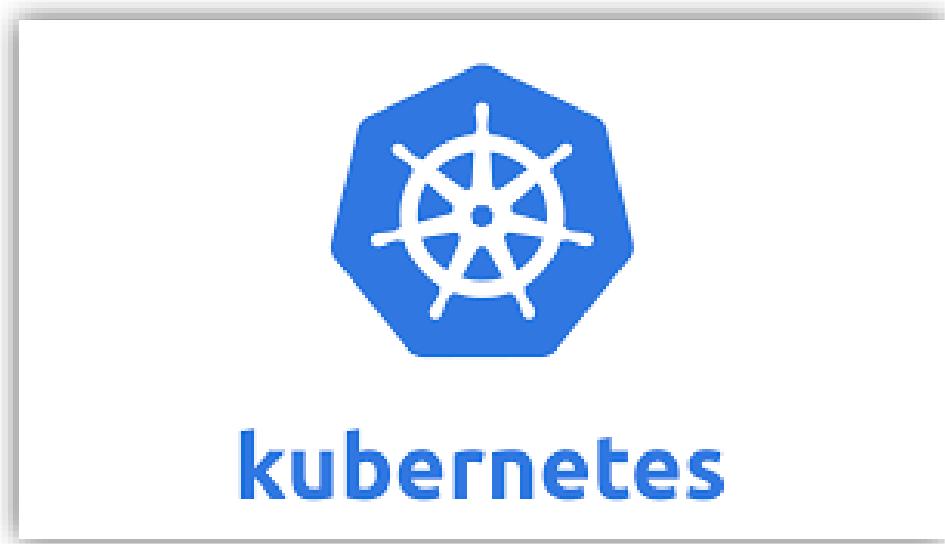
- I. Giới thiệu chung.

- II. Cơ sở lý thuyết và cài đặt.
- III. Triển khai dịch vụ (Static web).
- IV. Triển khai dịch vụ gia tăng (Dynamic web).
- V. Backup và restore.
- VI. Tìm hiểu và triển khai các công cụ bảo mật.
- Kết luận.
- Tài liệu tham khảo.

Từ những nội dung đã trình bày ở phần I, có thể thấy tầm quan trọng của Kubernetes trong việc triển khai ứng dụng trên nền tảng cloud. Để hiểu rõ hơn về công nghệ này và chuẩn bị cho phần thực hành triển khai, phần II sẽ trình bày các kiến thức lý thuyết cơ bản và quá trình cài đặt môi trường Kubernetes.

II. CƠ SỞ LÝ THUYẾT VÀ CÀI ĐẶT

1. Kubernetes là gì?



Hình 1. Kubernetes

Kubernetes (K8s) là một hệ thống mã nguồn mở giúp tự động hóa việc quản lý, mở rộng quy mô và triển khai ứng dụng dưới dạng container. Về cơ bản, đây là hệ điều hành cho nền tảng đám mây, cung cấp các công cụ để quản lý và điều phối các ứng dụng trên nhiều máy chủ, đảm bảo chúng luôn hoạt động hiệu quả và ổn định.[1]

Bên cạnh đó K8s còn được gọi là Container Orchestration Engine (Công cụ điều phối container). Kubernetes loại bỏ rất nhiều các quy trình thủ công liên quan đến việc triển khai và mở rộng các containerized applications.[1]

2. Tại sao bạn cần kubernets và nó có thể làm gì?

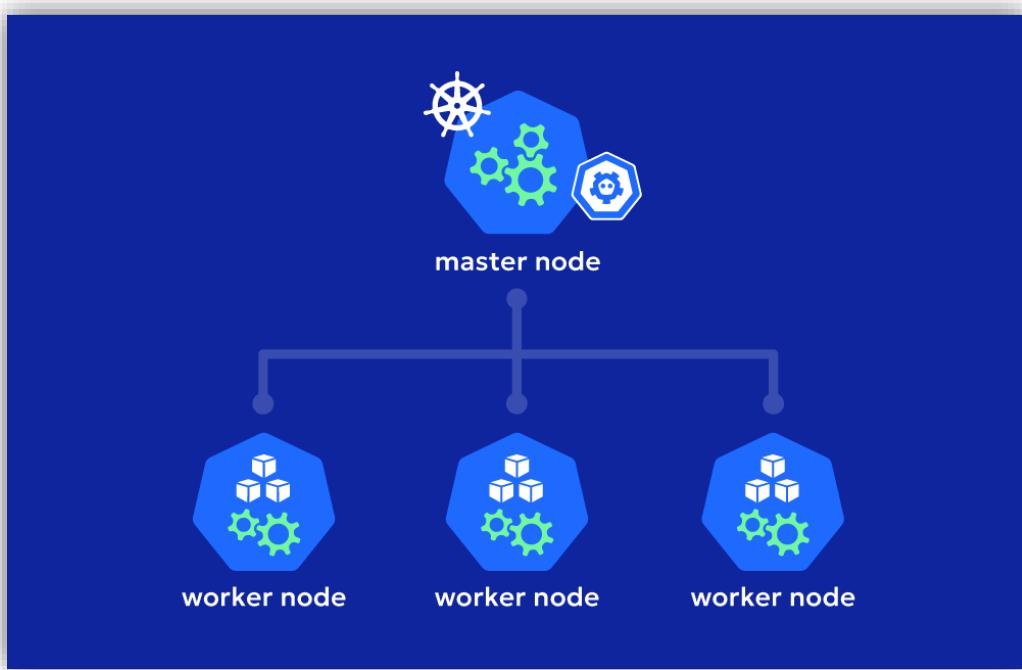
Kubernetes cung cấp cho bạn một framework để chạy các hệ phân tán một cách mạnh mẽ. Nó đảm nhiệm việc nhân rộng và chuyển đổi dự phòng cho ứng dụng của bạn, cung cấp các mẫu deployment và hơn thế nữa.[2]

Kubernetes cung cấp cho bạn:

- **Service Discovery & Cân bằng tải:** Kubernetes sử dụng DNS hoặc IP để expose container và tự động phân phối lưu lượng truy cập để đảm bảo ổn định hệ thống.[2]
- **Điều phối bộ nhớ:** Tự động mount hệ thống lưu trữ từ nhiều nguồn như ổ cứng local hoặc cloud provider.[2]

- **Tự động Rollout & Rollback:** Cho phép cập nhật hoặc quay lại phiên bản cũ một cách an toàn theo trạng thái mong muốn đã định nghĩa.[2]
- **Đóng gói tự động:** Tự động phân bổ container đến các node dựa trên tài nguyên yêu cầu (CPU, RAM) để tối ưu hiệu suất.[2]
- **Tự phục hồi:** Tự động khởi động lại hoặc thay thế các container gặp lỗi, đảm bảo tính sẵn sàng cao.[2]
- **Quản lý cấu hình & bảo mật:** Hỗ trợ lưu trữ và cập nhật thông tin nhạy cảm (như mật khẩu, token...) mà không cần rebuild container.[2]

3. Những khái niệm cơ bản trong Kubernetes



Hình 2. MasterNode_WorkerNode

Master Node – Điều khiển hệ thống Kubernetes[3]

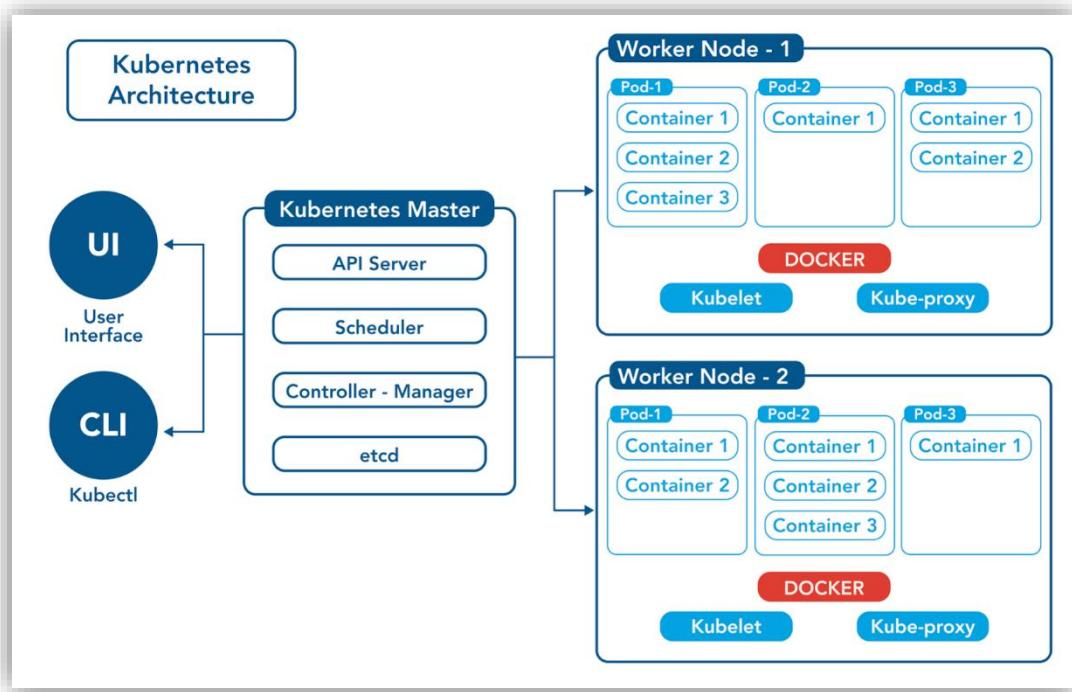
Gồm 4 thành phần chính:

- API Server: Giao tiếp giữa các thành phần và với lập trình viên qua API.
- Scheduler: Lập lịch, chọn node phù hợp để chạy ứng dụng.
- Controller Manager: Quản lý trạng thái hệ thống, kiểm tra node, nhân bản ứng dụng.
- etcd: Cơ sở dữ liệu lưu trữ toàn bộ trạng thái của cluster.

Worker Node – Chạy ứng dụng (Container) [3]

Gồm 3 thành phần chính:

- Container Runtime: Chạy container (thường là Docker).
- Kubelet: Giao tiếp với API Server, quản lý container trên node.
- Kube-Proxy: Cân bằng tải, phân phối lưu lượng đến các container.



Hình 3. Thành phần của Kubernetes

Kubelet:

Đảm bảo các Pod đang chạy, bao gồm cả Container của chúng.[4]

Kube-proxy:

Duy trì các quy tắc mạng trên các Node để thực hiện các Service.[4]

Pod

Là đơn vị triển khai nhỏ nhất trong Kubernetes, chứa một hoặc nhiều container. Pod chạy trên các Worker Node và có tài nguyên riêng như CPU, RAM, volume, IP mạng,... Là nơi trực tiếp thực thi ứng dụng.

Image

Là phần mềm ứng dụng đã được đóng gói dưới dạng container. Pod sử dụng các Image này để chạy container. Image thường được lưu trữ ở các registry như Docker Hub (ví dụ: nginx, mysql, wordpress...).

Deployment

Đối tượng quản lý việc triển khai và cập nhật Pod. Deployment hỗ trợ tự động cập nhật (rollout), khôi phục (rollback), và đảm bảo số lượng Pod hoạt động đúng yêu cầu.

Replica Controller (ReplicaSet)

Dùng để quản lý số lượng bản sao (replica) của Pod. Tự động tạo thêm hoặc xoá bớt Pod để đảm bảo hệ thống luôn có đủ số lượng Pod cần thiết.

Service

Cung cấp địa chỉ mạng ổn định để các Pod giao tiếp với nhau hoặc để người dùng truy cập vào ứng dụng. Hỗ trợ cân bằng tải (load balancing) giữa các bản sao Pod, giúp ứng dụng hoạt động ổn định.

Label

Là cặp key-value dùng để phân loại và quản lý các đối tượng như Pod. Labels giúp dễ dàng lọc, chọn hoặc nhóm các Pod theo mục đích sử dụng (frontend, backend) hoặc theo môi trường (dev, test, prod,...).

4. Ưu và nhược điểm của Kubernetes[5]

4.1. Ưu điểm

- Khả năng mở rộng: Kubernetes cho phép dễ dàng tăng giảm số lượng container, cân bằng tải và tự động mở rộng quy mô mà không gián đoạn hệ thống.
- Tự động hóa và linh hoạt: Containers có tính di động cao, có thể chạy trên mọi cơ sở hạ tầng đám mây, giúp dễ dàng di chuyển ứng dụng giữa các môi trường khác nhau.
- Tận dụng tài nguyên hiệu quả: Kubernetes tối ưu hóa việc sử dụng tài nguyên máy chủ, đảm bảo các container hoạt động với tài nguyên cần thiết, nâng cao hiệu suất.
- Triển khai nhanh chóng: K8s chuẩn hóa quy trình đóng gói ứng dụng, tăng tính di động và nhất quán, giúp triển khai nhanh hơn và tiết kiệm công sức.

4.2. Nhược điểm

- Mức độ phức tạp: Kubernetes có cấu trúc phức tạp, yêu cầu người dùng nắm rõ nhiều khái niệm và thành phần, gây khó khăn cho người mới bắt đầu trong việc thiết lập và quản lý.

- Khả năng bảo mật:** Các container nhẹ và di động có thể dễ bị đe dọa an toàn nếu không được bảo vệ đúng cách, đòi hỏi nhà phát triển phải đảm bảo việc bảo mật để ngăn chặn truy cập trái phép.
- Chi phí tài nguyên:** Việc triển khai và vận hành Kubernetes cần tài nguyên bổ sung như CPU, bộ nhớ và lưu trữ, có thể làm tăng chi phí nếu không được quản lý hợp lý.

Tiêu chí	Kubernetes	Docker Swarm	Nomad
Độ phổ biến	Rất cao (chuẩn công nghiệp)	Trung bình	Đang phát triển
Để sử dụng	Khó hơn, cần thời gian học	Dễ, đơn giản	Trung bình
Khả năng mở rộng	Rất tốt	Tốt ở mức nhỏ và vừa	Tốt, nhẹ
Tự phục hồi	Có, mạnh mẽ	Có, đơn giản	Có
Quản lý trạng thái	Tốt (StatefulSets)	Hạn chế	Có
Hệ sinh thái hỗ trợ	Rộng, nhiều công cụ tích hợp	Hạn chế	Tốt (Vault, Consul)

Bảng 1 So sánh Kubernetes với các công cụ khác

5. Cài đặt Kubernetes

Để sử dụng Kubernetes, chúng ta cần cài đặt một cụm (cluster) gồm ít nhất một máy điều khiển (*master*) và một hoặc nhiều máy làm việc (*worker*). Việc cài đặt Kubernetes có thể thực hiện theo nhiều cách khác nhau, tùy theo mục đích sử dụng như học tập, thử nghiệm hay triển khai thực tế.

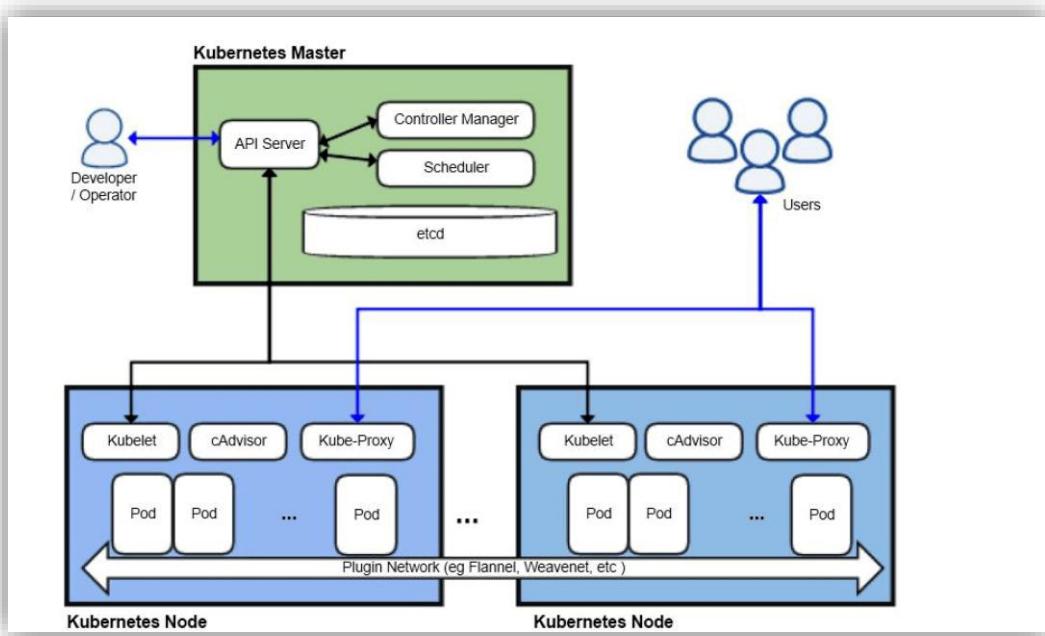
Các cách cài đặt Kubernetes phổ biến:

- Cài đặt đơn giản để học tập:** Dùng các công cụ như Minikube, Kind, MicroK8s hoặc k3s để chạy Kubernetes trên một máy tính. Cách này nhanh, dễ làm và không cần nhiều tài nguyên.
- Cài đặt thủ công với kubeadm:** Dành cho những ai muốn tìm hiểu sâu hơn hoặc thử nghiệm cụm Kubernetes thật trên nhiều máy. Yêu cầu người dùng có kiến thức hệ thống cơ bản.
- Dùng dịch vụ đám mây:** Các nền tảng như Google Cloud (GKE), Amazon AWS (EKS) hay Microsoft Azure (AKS) cho phép tạo cụm Kubernetes dễ dàng mà không cần tự cài đặt thủ công. Phù hợp với doanh nghiệp và triển khai thực tế.

Tiêu chí	Minikube / Kind / k3s / MicroK8s	Kubeadm (cài thủ công)	Dịch vụ đám mây (GKE, EKS, AKS)
Mục đích sử dụng	Học tập, thử nghiệm cá nhân	Tìm hiểu sâu, mô phỏng môi trường thật	Triển khai thật, sản phẩm/doanh nghiệp
Độ phức tạp	Thấp, dễ cài đặt	Trung bình – cao	Rất thấp (hầu hết thao tác qua giao diện)
Tài nguyên yêu cầu	Thấp (chạy trên 1 máy)	Trung bình – cần nhiều máy ảo hoặc server	Do nhà cung cấp đám mây chịu trách nhiệm
Mức độ tùy biến	Thấp – trung bình	Cao (tùy chỉnh cấu hình)	Thấp – theo giới hạn của nhà cung cấp
Tính thực tế	Thấp – mô phỏng	Cao – giống môi trường thật	Rất cao – mô trường triển khai thực tế
Chi phí	Miễn phí (trừ tài nguyên cá nhân)	Miễn phí (nếu tự động trên máy ảo)	Trả phí theo giờ/tài nguyên sử dụng
Phù hợp với ai?	Sinh viên, người mới học	Kỹ sư hệ thống, DevOps học chuyên sâu	Doanh nghiệp, dự án thực tế

Bảng 2 So sánh các cách cài đặt Kubernetes

Trong bài báo cáo này, em lựa chọn phương pháp cài đặt Kubernetes thủ công bằng kubeadm, triển khai trên các máy ảo của Google Cloud Platform (GCP). Cách làm này giúp em hiểu rõ hơn về quá trình thiết lập và vận hành một cụm Kubernetes thực tế, từ bước cấu hình hệ thống cho đến khởi tạo và kết nối các node trong cụm.



Hình 4. Mô hình triển khai k8s

Chuẩn bị hệ thống

Tên VM	IP	VCPU	RAM	Disk	OS	Firewall
k8s-master	10.148.0.26	2	4GB	100GB	Ubuntu 22.04 LTS	Cho phép HTTP + HTTPS
k8s-worker1	10.148.0.27	2	4GB	100GB	Ubuntu 22.04 LTS	Cho phép HTTP + HTTPS
data	10.148.0.28	2	4GB	50GB	Ubuntu 22.04 LTS	Cho phép HTTP + HTTPS

Thực hiện trên cả 2 servers

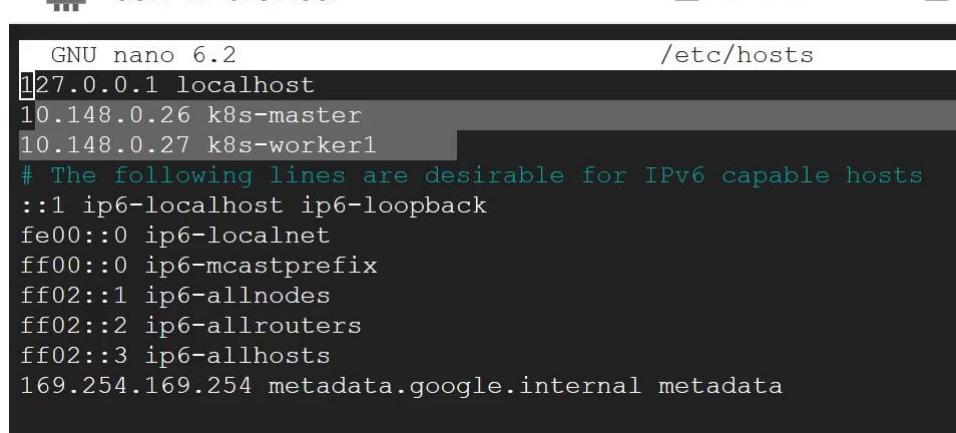
Thêm hosts

```
# vi /etc/hosts/
```

Nội dung cấu hình

10.148.0.26 k8s-master

10.148.0.27 k8s-worker1



```
GNU nano 6.2                               /etc/hosts
127.0.0.1 localhost
10.148.0.26 k8s-master
10.148.0.27 k8s-worker1
# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
169.254.169.254 metadata.google.internal metadata
```

Hình 5. Nội dung hosts

Cập nhật và nâng cấp hệ thống

```
# sudo apt update -y && sudo apt upgrade -y
```

Tắt swap

```
# sudo swapoff -a
```

```
# sudo sed -i '/swap.img/s/^/#/' /etc/fstab
```

Cấu hình module kernel

```
# vi /etc/modules-load.d/containerd.conf
```

Nội dung sau:

overlay

br_nf

The screenshot shows a terminal window titled "SSH-in-browser". The terminal has a dark background and displays the following text:

```
overlay
br_netfilter
```

At the bottom right of the terminal window, there are two status indicators: "2,12" and "All".

Hình 6. Nội dung file containerd.conf

Tải module kernel

```
# sudo modprobe overlay
# sudo modprobe br_netfilter
```

Cấu hình hệ thống mạng

```
# echo "net.bridge.bridge-nf-call-ip6tables = 1" | sudo tee -a
/etc/sysctl.d/kubernetes.conf
# echo "net.bridge.bridge-nf-call-iptables = 1" | sudo tee -a
/etc/sysctl.d/kubernetes.conf
# echo "net.ipv4.ip_forward = 1" | sudo tee -a /etc/sysctl.d/kubernetes.conf
```

Áp dụng cấu hình sysctl

```
# sudo sysctl -system
```

Cài đặt các gói cần thiết và thêm kho Docker

```
# sudo apt install -y curl gnupg2 software-properties-common apt-transport-
https ca-certificates
# sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --
dearmour -o /etc/apt/trusted.gpg.d/docker.gpg
```

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

Cài đặt containerd

```
# sudo apt update -y  
# sudo apt install -y containerd.io
```

Cấu hình containerd

```
# containerd config default | sudo tee /etc/containerd/config.toml >/dev/null  
2>&1  
  
# sudo sed -i 's/SystemdCgroup = false/SystemdCgroup = true/g'  
/etc/containerd/config.toml
```

Khởi động containerd

```
# sudo systemctl restart containerd  
# sudo systemctl enable containerd
```

Thêm kho lưu trữ Kubernetes

```
# echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]  
https://pkgs.k8s.io/core:/stable:/v1.30/deb/ /" | sudo tee  
/etc/apt/sources.list.d/kubernetes.list  
  
# curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key | sudo gpg --  
dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
```

Cài đặt các gói Kubernetes

```
# sudo apt update -y  
# sudo apt install -y kubelet kubeadm kubectl  
# sudo apt-mark hold kubelet kubeadm kubectl
```

```

SSH-in-browser
SSH-in-browser
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.30
/distro/deb_kubernetes-cni_1.4.0-1.1 [32.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.30
/distro/deb_kubelet_1.30.14-1.1 [18.2 MB]
Fetched 93.8 MB in 2s (49.3 MB/s)
Selecting previously unselected package conntrack.
(Reading database ... 67012 files and directories currently installed.)
Preparing to unpack .../0-conntrack_1%3al.4.6-2build2_amd64.deb ...
Unpacking conntrack (1:1.4.6-2build2) ...
Selecting previously unselected package cri-tools.
Preparing to unpack .../1-cri-tools_1.30.1-1.1_amd64.deb ...
Unpacking cri-tools (1.30.1-1.1) ...
Selecting previously unselected package kubeadm.
Preparing to unpack .../2-kubeadm_1.30.14-1.1_amd64.deb ...
Unpacking kubeadm (1.30.14-1.1) ...
Selecting previously unselected package kubelet.
Preparing to unpack .../3-kubelet_1.30.14-1.1_amd64.deb ...
Unpacking kubelet (1.30.14-1.1) ...
Selecting previously unselected package kubernetes-cni.
Preparing to unpack .../4-kubernetes-cni_1.4.0-1.1_amd64.deb ...
Unpacking kubernetes-cni (1.4.0-1.1) ...
Selecting previously unselected package kubelet.
Preparing to unpack .../5-kubelet_1.30.14-1.1_amd64.deb ...
Unpacking kubelet (1.30.14-1.1) ...
Setting up conntrack (1:1.4.6-2build2) ...
Setting up kubelet (1.30.14-1.1) ...
Setting up cri-tools (1.30.1-1.1) ...
Setting up kubernetes-cni (1.4.0-1.1) ...
Setting up kubelet (1.30.14-1.1) ...
Setting up kubelet (1.30.14-1.1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...
Running kernel seems to be up-to-date.
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
quangkhai1692004@k8s-master:~$ sudo apt-mark hold kubelet kubeadm kubelet
kubelet set on hold.
kubeadm set on hold.
kubelet set on hold.

SSH-in-browser
SSH-in-browser
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.30
/distro/deb_kubernetes-cni_1.4.0-1.1 [32.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.30
/distro/deb_kubelet_1.30.14-1.1 [18.2 MB]
Fetched 93.8 MB in 2s (49.3 MB/s)
Selecting previously unselected package conntrack.
(Reading database ... 67012 files and directories currently installed.)
Preparing to unpack .../0-conntrack_1%3al.4.6-2build2_amd64.deb ...
Unpacking conntrack (1:1.4.6-2build2) ...
Selecting previously unselected package cri-tools.
Preparing to unpack .../1-cri-tools_1.30.1-1.1_amd64.deb ...
Unpacking cri-tools (1.30.1-1.1) ...
Selecting previously unselected package kubeadm.
Preparing to unpack .../2-kubeadm_1.30.14-1.1_amd64.deb ...
Unpacking kubeadm (1.30.14-1.1) ...
Selecting previously unselected package kubelet.
Preparing to unpack .../3-kubelet_1.30.14-1.1_amd64.deb ...
Unpacking kubelet (1.30.14-1.1) ...
Setting up conntrack (1:1.4.6-2build2) ...
Setting up kubelet (1.30.14-1.1) ...
Setting up cri-tools (1.30.1-1.1) ...
Setting up kubernetes-cni (1.4.0-1.1) ...
Setting up kubelet (1.30.14-1.1) ...
Setting up kubelet (1.30.14-1.1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...
Running kernel seems to be up-to-date.
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
quangkhai1692004@k8s-worker1:~$ sudo apt-mark hold kubelet kubeadm kubelet
kubelet set on hold.
kubeadm set on hold.
kubelet set on hold.

```

Hình 7. Hoàn thành cấu hình cơ bản

Cấu hình máy chủ theo mô hình 1 master – 1 worker

Thực hiện trên server k8s-master-1

```

# sudo kubeadm init

# mkdir -p $HOME/.kube

# sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

# sudo chown $(id -u):$(id -g) $HOME/.kube/config

# kubectl apply -f

```

<https://raw.githubusercontent.com/projectcalico/calico/v3.25.0/manifests/calico.yaml>

To start using your cluster, you need to run the following as a regular user:

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Alternatively, if you are the root user, you can run:

```
export KUBECONFIG=/etc/kubernetes/admin.conf
```

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
<https://kubernetes.io/docs/concepts/cluster-administration/addons/>

Then you can join any number of worker nodes by running the following on each as root:

```
kubeadm join 10.148.0.3:6443 --token 0bt1w2.5dwflqlavrtgdyho \
--discovery-token-ca-cert-hash sha256:51392a8a99cb5524c64eb8223df9a3bf9e21b70eb6
7a3ca0e84c3032f27b7f70
quanghai692004@k8s-master:~$
```

Hình 8. Hoàn thành cấu hình k8s-master

Thực hiện trên server k8s-worker1 và k8s-worker2

```
# sudo kubeadm join 10.148.0.3:6443 --token your_token --discovery-token-ca-
cert-hash your_sha
```

```
7a3ca0e84c3032f27b7f70
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm ku
beadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/ku
beadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This ca
n take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.001712952s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

quanghai692004@k8s-worker2:~$
```

Hình 9. Hoàn thành cấu hình k8s-worker1

The screenshot shows a terminal window titled "SSH-in-browser". The terminal displays the output of a "kubeadm init" command. The output includes pre-flight checks, configuration reading from the cluster, and the kubelet starting up. It also shows the node joining the cluster successfully and provides instructions to run "kubectl get nodes" to see the node join the cluster. The terminal prompt at the bottom is "quangkhai692004@k8s-worker1:~\$".

```
7a3ca0e84c3032f27b7f70
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeconfig -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.001552532s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

quangkhai692004@k8s-worker1:~$
```

Hình 10. Hoàn thành cấu hình k8s-worker2

[+] https://youtu.be/Ek9_FMwO1Gs, Setup K8S.

Sau khi hoàn tất các bước cài đặt Kubernetes bằng kubeadm trên các máy ảo GCP, cụm Kubernetes đã sẵn sàng để sử dụng. Cụm bao gồm một master node và các worker node được kết nối thông qua kubeadm, cùng với plugin mạng đã được cấu hình đầy đủ.

Tiếp theo, ở phần III sẽ tiến hành triển khai dịch vụ (Static web) lên cụm Kubernetes để kiểm tra hoạt động của hệ thống, sử dụng các công cụ như kubectl để quản lý Pod, Service và kiểm soát lưu lượng mạng nội bộ

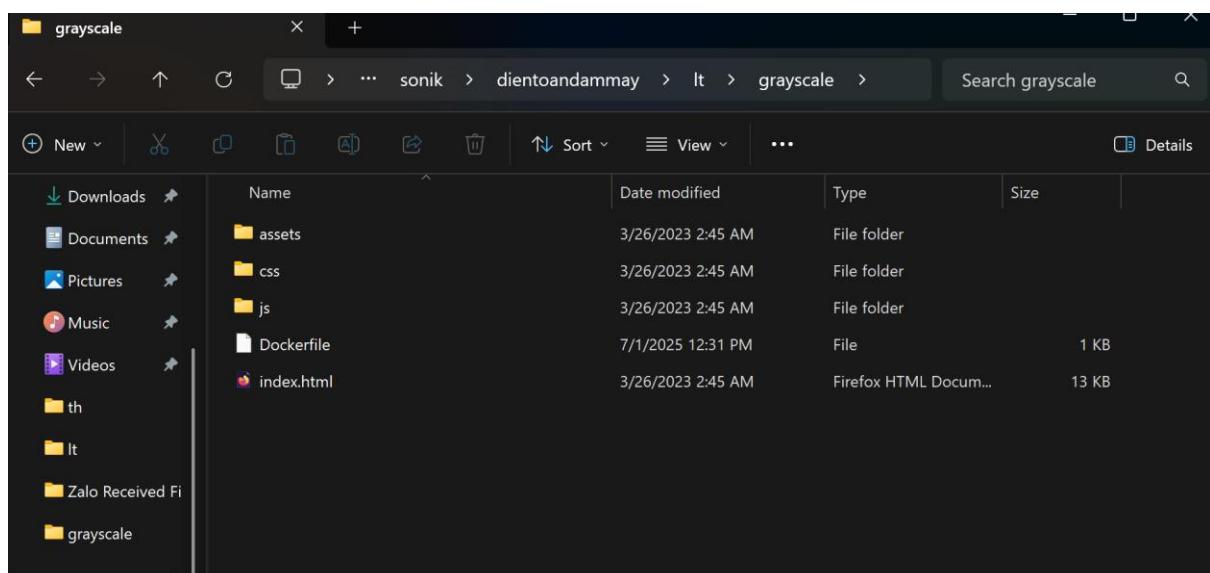
III. Triển khai dịch vụ (Static web)

Trước khi làm việc với các ứng dụng web phức tạp, việc triển khai một website tĩnh là bước khởi đầu quan trọng để làm quen với cách Kubernetes vận hành dịch vụ. Trong chương này, chúng ta sẽ học cách triển khai một ứng dụng web tĩnh (static web), chẳng hạn như một trang HTML đơn giản, sử dụng Pod, Deployment và Service.

Thông qua chương này, bạn sẽ hiểu được quy trình cơ bản để đưa một ứng dụng lên Kubernetes, cách truy cập dịch vụ từ bên ngoài cụm, và nền tảng để triển khai các ứng dụng web nâng cao ở các chương tiếp theo.

Chuẩn bị image trên DockerHub

Các file của web



Hình 11 Nội dung file web

Nội dung file Docker

```
# Dùng image nginx nhẹ
FROM nginx:alpine

# Xóa file mặc định
RUN rm -rf /usr/share/nginx/html/*

# Copy toàn bộ source vào thư mục HTML của nginx
COPY . /usr/share/nginx/html

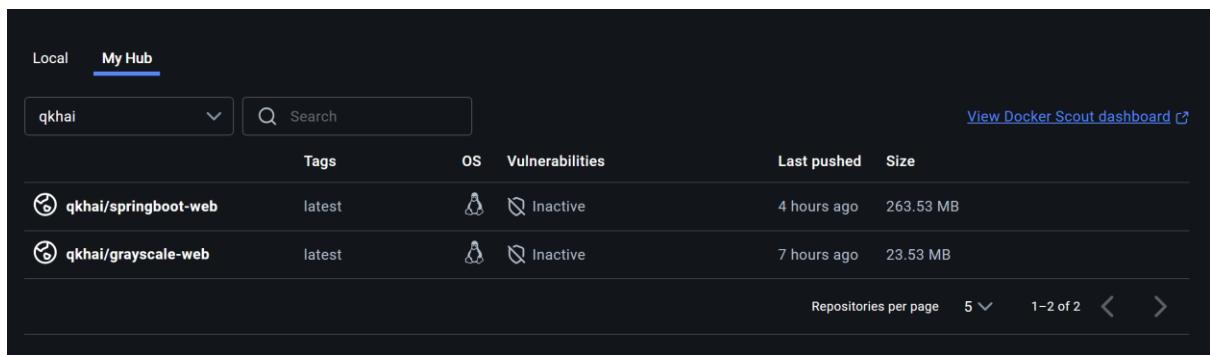
# Mở port 80
EXPOSE 80
```

Hình 12 Nội dung Dockerfile

Build và đẩy image lên Docker Hub

```
# docker build -t yourdockerhub/your-web-app .
```

```
# docker push yourdockerhub/your-web-app
```



Hình 13 Image trên Docker Hub

Kiểm tra lại kết nối với cluster

```
# kubectl get nodes
```

```
see 'snap info <snapshot>' for additional versions.  
quanghai692004@master-k8s:~$ kubectl get no  
NAME        STATUS   ROLES      AGE     VERSION  
master-k8s   Ready    control-plane   2d21h   v1.30.14  
worker1-k8s  NotReady   <none>    2d21h   v1.30.14  
worker2-k8s  NotReady   <none>    2d21h   v1.30.14  
quanghai692004@master-k8s:~$
```

Hình 14 Các node của cụm K8s

Tiến hành tạo một thư mục để để quản lý

```
# sudo mkdir grayscale
```

```
quanghai692004@master-k8s:~$ ls  
grayscale  
quanghai692004@master-k8s:~$
```

Hình 15 Thư mục quản lý web trên K8s

Tạo file YAML mô tả Deployment

```
# sudo vi deployment.yaml
```



SSH-in-browser

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: grayscale-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: grayscale
  template:
    metadata:
      labels:
        app: grayscale
    spec:
      containers:
        - name: grayscale
          image: qkhai/grayscale-web:latest
          ports:
            - containerPort: 80
```

~

Hình 16 Nội dung file deployment

Áp dụng Deployment

```
# kubectl apply -f deployment.yaml
```

Kiểm tra Deployment

```
# kubectl get pods
```

```
quangkhai692004@master-k8s:~/grayscale$ kubectl get pod
NAME                               READY   STATUS        RESTARTS   AGE
grayscale-deployment-66df66784f-225qz   1/1    Terminating   0          7h7m
grayscale-deployment-66df66784f-69lpr    0/1    Pending       0          12m
grayscale-deployment-66df66784f-8pws4    0/1    Pending       0          12m
grayscale-deployment-66df66784f-k16t2    1/1    Terminating   0          7h7m
quangkhai692004@master-k8s:~/grayscale$ █
```

Hình 17 Các pod được khởi tạo trên cụm k8s

Tạo Service kiểu NodePort để expose ra ngoài

```
# sudo vi service.yaml
```



SSH-in-browser

```
apiVersion: v1
kind: Service
metadata:
  name: grayscale-service
spec:
  type: NodePort
  selector:
    app: grayscale
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30080
```

```
~  
~  
~
```

Hình 18. Nội dung file service YAML

Áp dụng Service

```
# kubectl apply -f web-ser.yaml
```

Kiểm tra service:

```
# kubectl get svc
```

```
quangkhai692004@master-k8s:~/grayscale$ kubectl get svc
NAME         TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)        AGE
grayscale-service   NodePort   10.104.121.31 <none>        80:30080/TCP   7h9m
kubernetes   ClusterIP  10.96.0.1    <none>        443/TCP       2d21h
quangkhai692004@master-k8s:~/grayscale$ █
```

Hình 19. Kiểm tra service

Truy cập ứng dụng từ bên ngoài

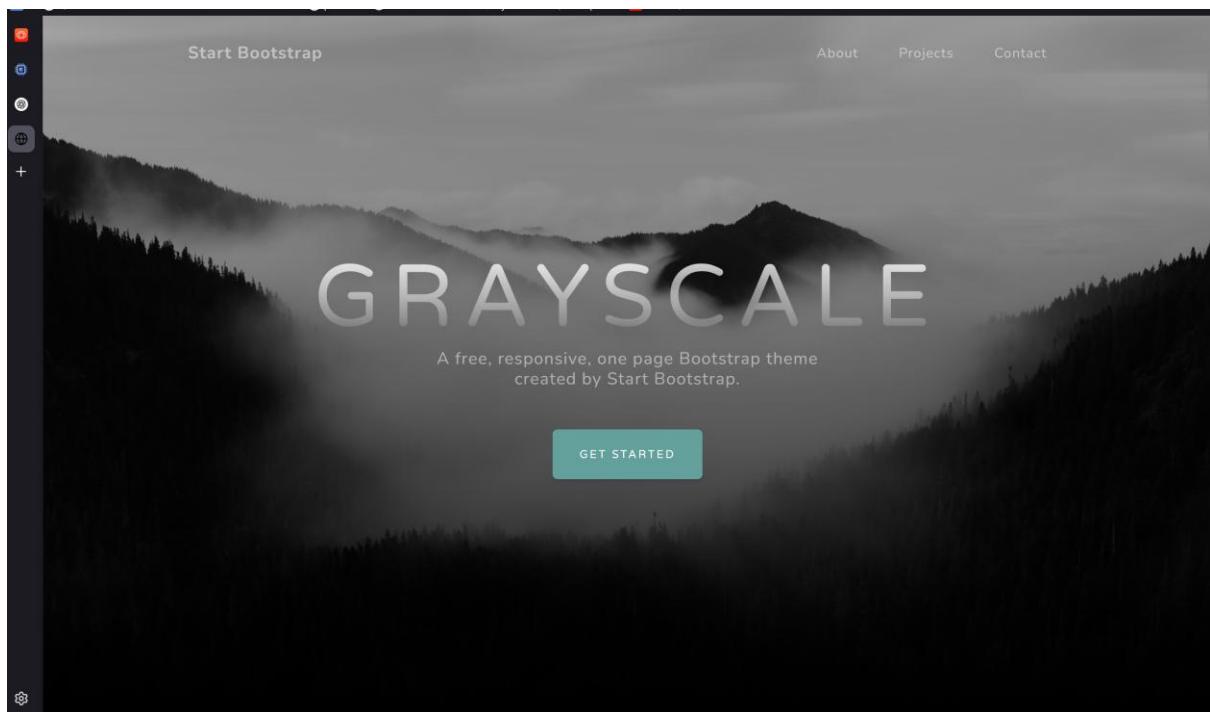
http://<Node_IP>:30080

Node IP là IP public của 1 trong các **worker node** (hoặc master nếu có gán IP public).

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	master-k8s	asia-northeast1-c	10.146.0.6 (nic0)	35.221.112.1 (nic0)	SSH
<input checked="" type="checkbox"/>	<input type="radio"/>	worker1-k8s	asia-northeast1-c	10.146.0.7 (nic0)		SSH
<input checked="" type="checkbox"/>	<input type="radio"/>	worker2-k8s	asia-northeast1-c	10.146.0.8 (nic0)		SSH

Hình 20. Thông tin về các máy chủ k8s

Sau khi triển khai thành công website tĩnh lên cụm Kubernetes, em đã có thể truy cập trang web thông qua địa chỉ IP của node và cổng được mở (NodePort). Điều này cho thấy cụm Kubernetes hoạt động ổn định, các thành phần như Pod, Service được cấu hình đúng, và có thể phục vụ nội dung web một cách hiệu quả.



Hình 21. Static web

[+] <https://youtu.be/wXGXspB8X8Q>, StaticWeb.

Sau khi hoàn thành các bước trên, em đã triển khai thành công một website tĩnh trên nền tảng Kubernetes.

Việc triển khai này giúp em hiểu được cách chạy một ứng dụng đơn giản trong cụm Kubernetes, cách tạo Pod, Service và mở cổng truy cập từ bên ngoài.

Ở phần tiếp theo – Phần IV – em sẽ tiếp tục triển khai một trang web động là WordPress, có khả năng kết nối với cơ sở dữ liệu MariaDB.

Thông qua phần này, em muốn tìm hiểu cách triển khai một ứng dụng thực tế gồm cả web và database trong Kubernetes, cũng như cách các thành phần này giao tiếp với nhau trong môi trường container.

IV. Triển khai dịch vụ gia tăng (Dynamic web).

Trong chương này, chúng ta sẽ tìm hiểu cách triển khai một dịch vụ web động trên Kubernetes, cụ thể là cài đặt WordPress – một nền tảng quản trị nội dung phổ biến. Bên cạnh đó, chương cũng hướng dẫn cách kết nối với cơ sở dữ liệu MySQL, cấu hình lưu trữ dữ liệu ổn định (Persistent Volume), và thiết lập truy cập an toàn qua HTTPS.

Mục tiêu của chương là giúp bạn biết cách triển khai một website có thể hoạt động thực tế, từ đó mở rộng và áp dụng vào các dự án khác trong tương lai.

Tạo thư mục để dễ quản lý dự án

```
mkdir wordpress && cd wordpress
```

Cấu hình file mysql-pv.yaml



```
GNU nano 6.2                                     mysql-pv.yaml *
apiVersion: v1
kind: PersistentVolume
metadata:
  name: mysql-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data/mysql"
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Hình 22 mysql-pv.yaml

Cấu hình file wordpress-pv.yaml

```
GNU nano 6.2
apiVersion: v1
kind: PersistentVolume
metadata:
  name: wordpress-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data/wordpress"
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: wordpress-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Hình 23 wordpress-pv.yaml

Cấu hình fiel mysql-deployment.yaml

```
GNU nano 6.2
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
spec:
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:5.7
          env:
            - name: MYSQL_ROOT_PASSWORD
              value: wordpresspass
            - name: MYSQL_DATABASE
              value: wordpress
            - name: MYSQL_USER
              value: wordpress
            - name: MYSQL_PASSWORD
              value: wordpresspass
        ports:
          - containerPort: 3306
      volumeMounts:
        - name: mysql-storage
          mountPath: /var/lib/mysql
      volumes:
        - name: mysql-storage
          persistentVolumeClaim:
            claimName: mysql-pvc
---
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  selector:
    app: mysql
  ports:
    - port: 3306
      targetPort: 3306
```

Hình 24 mysql-deployment.yaml

Cấu hình file wordpress-deployment.yaml

```
  GNU nano 0.2
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
  template:
    metadata:
      labels:
        app: wordpress
    spec:
      containers:
        - name: wordpress
          image: wordpress:6.5
          env:
            - name: WORDPRESS_DB_HOST
              value: mysql:3306
            - name: WORDPRESS_DB_USER
              value: wordpress
            - name: WORDPRESS_DB_PASSWORD
              value: wordpresspass
            - name: WORDPRESS_DB_NAME
              value: wordpress
          ports:
            - containerPort: 80
          volumeMounts:
            - name: wordpress-storage
              mountPath: /var/www/html
      volumes:
        - name: wordpress-storage
          persistentVolumeClaim:
            claimName: wordpress-pvc
---
apiVersion: v1
kind: Service
metadata:
  name: wordpress
spec:
  type: NodePort
  selector:
    app: wordpress
  ports:
    - port: 80
      nodePort: 30080
Save modified buffer?
y Yes
```

Hình 25 wordpress-deployment.yaml

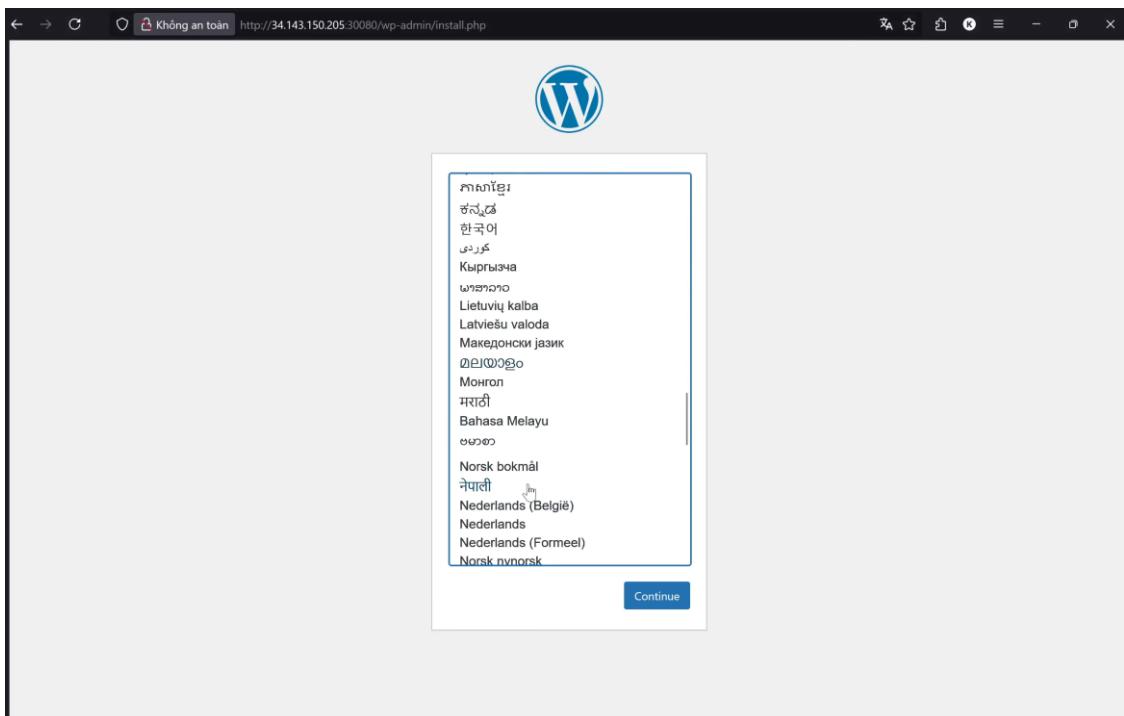
Áp dụng các file yaml

```
mysql-deployment.yaml mysql-pv.yaml wordpress-deployment.yaml wordpress-pv.yaml
quangkhai692004@k8s-master:~/wordpress$ kubectl apply -f mysql-pv.yaml
kubectl apply -f wordpress-pv.yaml
kubectl apply -f mysql-deployment.yaml
kubectl apply -f wordpress-deployment.yaml
persistentvolume/mysql-pv created
persistentvolumeclaim/mysql-pvc created
persistentvolume/wordpress-pv created
persistentvolumeclaim/wordpress-pvc created
deployment.apps/mysql created
service/mysql created
deployment.apps/wordpress created
service/wordpress created
quangkhai692004@k8s-master:~/wordpress$
```

Hình 26 Apply file yaml

Và cuối cùng có thể truy cập web vs địa chỉ:

http://<ip node>:<NodePort>



Hình 27 Truy cập web thành công

Thứ tiên hành tạo một user mới trong trên giao diện web

Tạo một người sử dụng mới và thêm vào trang mạng này.

Tên người dùng (bắt buộc)	guner
Email (bắt buộc)	guner@arsenal.com
Tên	guner
Họ	guner
Trang web	guner
Ngôn ngữ 	Trang web mặc định 
Mật khẩu	<input type="button" value="Tạo mật khẩu"/> <input type="text" value="C1zGE)kyIKuF\$&CYI)tdZCrd"/>   Mạnh
Gửi thông báo đến thành viên	<input checked="" type="checkbox"/> Gửi cho người dùng mới một email về tài khoản của họ
Vai trò	Thành viên đăng ký 
<input type="button" value="Thêm người dùng"/>	

Cảm ơn bạn đã khởi tạo với [WordPress](#).

Hình 28 Tạo user mới

Tiến hành truy cập mysql trong node để kiểm tra user:

```

Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 465
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, 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> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| wordpress |
+-----+
5 rows in set (0.00 sec)

mysql> USE wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from wp_users;
+----+-----+-----+-----+-----+-----+-----+-----+
| ID | user_login | user_pass           | user_nicename | user_email          | user_url           | user_registered |
|----+-----+-----+-----+-----+-----+-----+-----+
| 1  | sonik    | $wp$2y8109gfdMY1twBzzy7KeGFeNEokVPCwbulx.AchVRo3zyqt5tnHanrq | sonik          | quangkhai692004@gmail.com | http://34.143.150.205:30080 | 2025-07-27 10:37:33 |
| 2  | guner    | $wp$2y8109gfdMY1twBzzy7KeGFeNEokVPCwbulx.AchVRo3zyqt5tnHanrq | guner          | guner@arsenal.com   | http://guner       | 2025-07-27 11:27:55 |
+----+-----+-----+-----+-----+-----+-----+-----+

```

Hình 29 Database mysql

[+] <https://youtu.be/rPaMgxh2Mfk> ,Wordpress.

Qua chương IV, chúng ta đã tìm hiểu và triển khai thành công các dịch vụ web động trên nền tảng Kubernetes, bao gồm cài đặt WordPress, cấu hình cơ sở dữ liệu và bảo mật giao tiếp thông qua HTTPS. Những kiến thức này không chỉ giúp hệ thống hoạt động ổn định mà còn tạo nền tảng cho việc vận hành và mở rộng trong thực tế.

Tuy nhiên, để đảm bảo tính sẵn sàng và khả năng khôi phục khi xảy ra sự cố, việc sao lưu (backup) và phục hồi (restore) dữ liệu là yếu tố không thể thiếu trong bất kỳ hệ thống nào. Do đó, trong chương tiếp theo, chúng ta sẽ đi sâu vào quy trình xây dựng chiến lược backup và restore trên Kubernetes – giúp đảm bảo dữ liệu luôn được bảo vệ và có thể phục hồi một cách nhanh chóng khi cần thiết.

V. Backup và restore

Trong chương này, chúng ta sẽ tìm hiểu cách sao lưu và phục hồi dữ liệu trong Kubernetes để đảm bảo hệ thống luôn an toàn khi gặp sự cố.

Chúng ta sẽ sử dụng **Velero**, một công cụ hỗ trợ backup và restore các tài nguyên trong cụm Kubernetes. Dữ liệu sao lưu sẽ được lưu trữ trên **MinIO**, một dịch vụ lưu trữ đơn giản, tương thích với chuẩn S3.

Qua chương này, bạn sẽ biết cách cài đặt Velero, cấu hình MinIO, thực hiện backup và khôi phục lại dữ liệu khi cần thiết.

Trước tiên em sẽ chuẩn bị trước một server để lưu trữ các bản backup.

Filter Enter property name or value							
Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
<input checked="" type="checkbox"/> <input checked="" type="checkbox"/>	data	asia-southeast1-b			10.148.0.28 (nic0)	34.142.185.177 (nic0)	SSH
<input checked="" type="checkbox"/>							

Hình 30 Server backup

Trên server này đầu tiên tiến hành cài đặt Minio bằng docker-compose.

```
GNU nano 6.2
version: '3'
services:
  minio:
    image: minio/minio
    container_name: minio
    ports:
      - "9000:9000"
      - "9001:9001"
    volumes:
      - ./storage:/data
    environment:
      MINIO_ROOT_USER: devopseduvn
      MINIO_ROOT_PASSWORD: devopseduvn
    command: server --console-address ":9001" /data
```

Hình 31 Docker-compose file

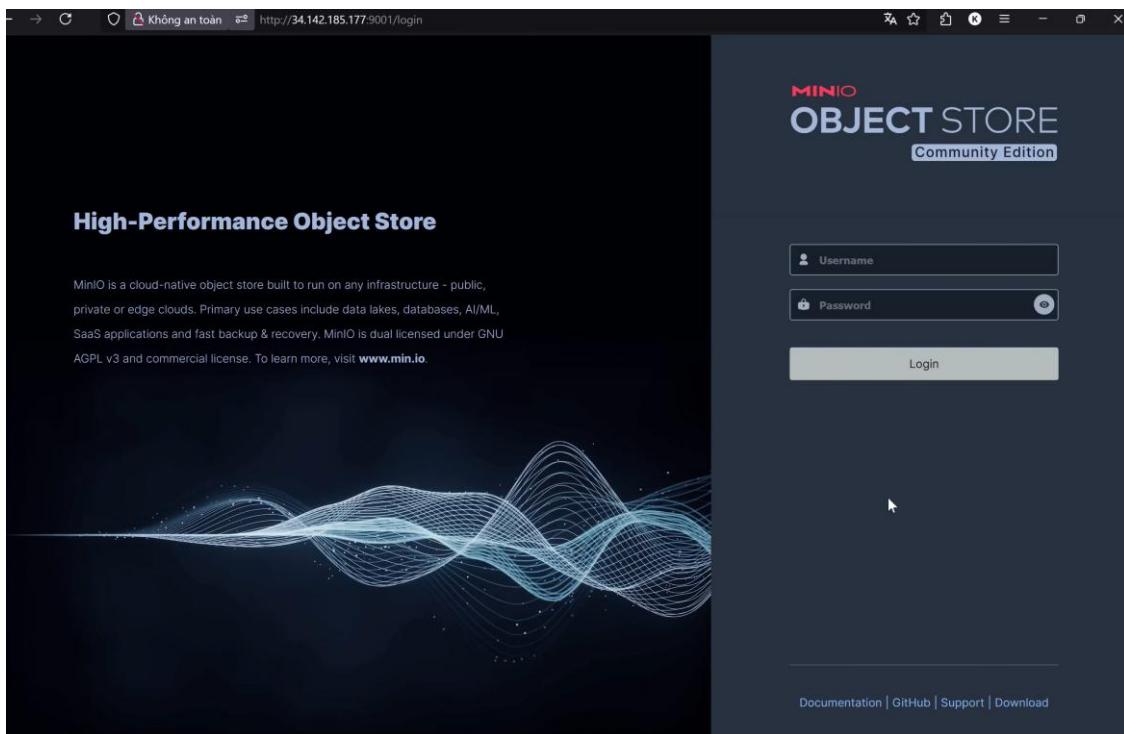
Bắt đầu khởi tạo minio bằng lệnh docker-compose up -d

```
raise DockerException(  
docker.errors.DockerException: Error while fetching server A  
quangkhai692004@data:~/minio$ sudo docker-compose up -d  
Creating network "minio_default" with the default driver  
Pulling minio (minio/minio:)...  
latest: Pulling from minio/minio  
dd15713b0f22: Pull complete  
fd312a3c94fc: Pull complete  
02e05d20c69e: Pull complete  
1f08d9fb7d7d: Pull complete  
b452b5fb7338: Pull complete  
e14ab7c7ae26: Pull complete  
3bf7f4428838: Pull complete  
adc9ba906e83: Pull complete  
1611ffcaf8be: Pull complete  
Digest: sha256:d249d1fb6966de4d8ad26c04754b545205ff15a62e4fd
```

Hình 32 docker-compose up -d

Truy cập giao diện web bằng ip server backup và port 9001

Tài khoản mật khẩu sẽ nằm trong file docker-compose lúc đầu cấu hình.

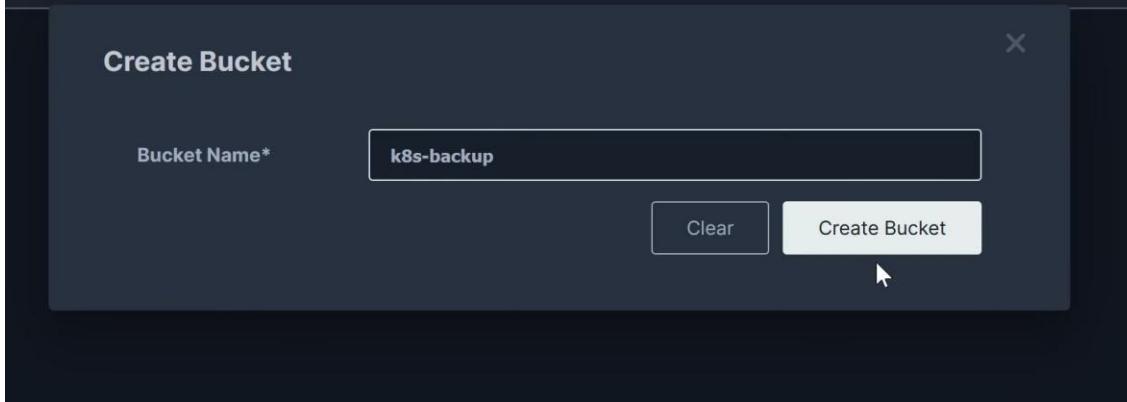


Hình 33 Web Minio

Tiến hành tạo bucket để lưu trữ backup

MinIO uses buckets to organize objects. A bucket is similar to a folder or directory in a file system, where each bucket can contain an arbitrary number of objects.

To get started, Create a Bucket.



Hình 34 Tạo bucket

Sau khi triển khai xong mini thì sẽ truyền qua cụm k8s để cài đặt velero

```
quangkhai692004@k8s-master:~$ curl -LO https://github.com/vmware-tanzu/velero/releases/download/v1.16.1/velero-v1.16.1-linux-amd64.tar.gz
% Total    % Received  % Xferd  Average Speed   Time     Time      Current
          Dload  Upload   Total Spent  Left  Speed
0     0     0     0     0     0     0 --:--:-- --:--:-- --:--:-- 0
100 52.8M 100 52.8M 0     0 17.8M 0:00:02 0:00:02 22.3M
quangkhai692004@k8s-master:~$ ls
ssl  velero-v1.16.1-linux-amd64.tar.gz  web  wordpress
quangkhai692004@k8s-master:~$ tar -xvf velero-v1.16.1-linux-amd64.tar.gz
velero-v1.16.1-linux-amd64/LICENSE
velero-v1.16.1-linux-amd64/examples/minio/00-minio-deployment.yaml
velero-v1.16.1-linux-amd64/examples/nginx-app/README.md
velero-v1.16.1-linux-amd64/examples/nginx-app/base.yaml
velero-v1.16.1-linux-amd64/examples/nginx-app/with-pv.yaml
velero-v1.16.1-linux-amd64/velero
quangkhai692004@k8s-master:~$ ls
ssl  velero-v1.16.1-linux-amd64  velero-v1.16.1-linux-amd64.tar.gz  web  wordpress
quangkhai692004@k8s-master:~$ sudo mv velero-v1.16.1-linux-amd64/velero /usr/local/bin
quangkhai692004@k8s-master:~$ velero version
Client:
      Version: v1.16.1
      Git commit: 2eb97fa8b187f9ed0aeb49f216565eddf93a0b08
<error getting server version: no matches for kind "ServerStatusRequest" in version "velero.io/v1">
quangkhai692004@k8s-master:~$
```

Hình 35 Setup velero

Sau đó khởi tạo file credentials-velero để lưu key

```
GNU nano 6.2                                         credentials-velero *
[default]
aws_access_key_id = devopseduvn
aws_secret_access_key = devopseduvn
```

Hình 36 credentials-velero

Tiến hành cài đặt và kết nối đến database. Nhưng phải nhập chính xác các thông tin về ip và bucket

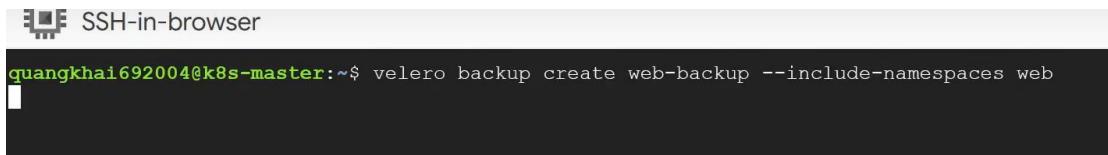
```

[1]: getting server version. No matches for kind "ServiceStatusRequest" in version "v1beta1"
quangkhai692004@k8s-master:~$ sudo nano credentials-velero
quangkhai692004@k8s-master:~$ velero install \
--provider aws \
--plugins velero/velero-plugin-for-aws:v1.6.0 \
--bucket k8s-backup \
--secret-file ./credentials-velero \
--use-volume-snapshots=false \
--backup-location-config region=minio,s3ForcePathStyle="true",s3Url=http://[34.142.185.177]:9000

```

Hình 37 Kết nối velero đến minio

Và cuối cùng có thể thực hiện backup và restore.



Hình 38 Backup

Sau đó bản backup đã được lưu trong minio

k8s-backup		
Created on: Mon, Jul 28 2025 03:36:43 (GMT+7) Access: PRIVATE		
<input type="button" value="Rewind ⏪"/> <input type="button" value="Refresh ⏮"/> <input type="button" value="Upload ⬆"/>		
◀	k8s-backup / backups / web-backup	Create new path ⌂
□	▲ Name	Last Modified
□	velero-backup.json	Today, 04:06
□	web-backup-csi-volumesnapshotclasses.json.gz	Today, 04:06
□	web-backup-csi-volumesnapshotcontents.json.gz	Today, 04:06
□	web-backup-csi-volumesnapshots.json.gz	Today, 04:06
□	web-backup-itemoperations.json.gz	Today, 04:06
□	web-backup-logs.gz	Today, 04:06
□	web-backup-podvolumebackups.json.gz	Today, 04:06
□	web-backup-resource-list.json.gz	Today, 04:06
□	web-backup-results.gz	Today, 04:06
□	web-backup-volumeinfo.json.gz	Today, 04:06
□	web-backup-volumesnapshots.json.gz	Today, 04:06
□	web-backup.tar.gz	Today, 04:06

Hình 39 Minio backup

Tiến hành xóa thử deployment của web và restore

```
Run `velero backup describe web-backup` or `velero backup logs web-backup` for more details.  
quangkhai1692004@k8s-master:~$ kubectl get deployment -n web  
NAME      READY   UP-TO-DATE   AVAILABLE   AGE  
grayscale-web   2/2     2           2           9h  
quangkhai1692004@k8s-master:~$ kubectl delete deployment grayscale-web -n web  
deployment.apps "grayscale-web" deleted  
quangkhai1692004@k8s-master:~$ kubectl get deployment -n web  
No resources found in web namespace.  
quangkhai1692004@k8s-master:~$ kubectl get deployment -n web  
No resources found in web namespace.  
quangkhai1692004@k8s-master:~$ kubectl get deployment -n web  
No resources found in web namespace.  
quangkhai1692004@k8s-master:~$ kubectl get deployment -n web  
No resources found in web namespace.  
quangkhai1692004@k8s-master:~$ velero restore create --from-backup web-backup  
Restore request "web-backup-20250727211053" submitted successfully.  
Run `velero restore describe web-backup-20250727211053` or `velero restore logs web-backup-20250727211053` for more details.  
quangkhai1692004@k8s-master:~$ kubectl get deployment -n web  
NAME      READY   UP-TO-DATE   AVAILABLE   AGE  
grayscale-web   2/2     2           2           6s  
quangkhai1692004@k8s-master:~$ █
```

Hình 40 Restore

Backup theo lịch 2h sáng mỗi ngày

```
grayscale-web   2/2     2           2           6s  
quangkhai1692004@k8s-master:~$ velero schedule create daily-backup --schedule="0 2 * * *" --include-namespaces '*'  
Schedule "daily-backup" created successfully.  
quangkhai1692004@k8s-master:~$ █
```

Hình 41 Daily backup

[+] <https://www.youtube.com/watch?v=oQjxfalZ6ec> , Backup - Restore

Kết luận

Qua quá trình thực hiện đề tài, em đã triển khai thành công các dịch vụ web trên nền tảng Kubernetes, bao gồm:

- **Triển khai web tĩnh:** Giúp hiển thị nội dung đơn giản, dễ dàng triển khai và quản lý.
- **Triển khai web động (WordPress):** Cho phép người dùng tương tác với hệ thống, thể hiện khả năng xử lý backend và tích hợp cơ sở dữ liệu.
- **Backup và Restore với Velero:** Đảm bảo an toàn cho dữ liệu, giúp hệ thống có khả năng phục hồi khi xảy ra sự cố.

Tuy nhiên, trong khuôn khổ thời gian và kiến thức có hạn, em vẫn chưa kịp hoàn thành phần triển khai **HTTPS** cho các dịch vụ, một yếu tố quan trọng nhằm tăng cường bảo mật kết nối giữa người dùng và hệ thống. Đây sẽ là nội dung em tiếp tục nghiên cứu và hoàn thiện trong giai đoạn tiếp theo.

Qua đề tài này, em đã hiểu rõ hơn về cách thức triển khai, vận hành và bảo vệ hệ thống dịch vụ web trong môi trường Kubernetes — một kỹ năng thiết yếu trong lĩnh vực DevOps và điện toán đám mây hiện nay. Đây cũng là cơ hội để em củng cố kiến thức và tiếp cận gần hơn với các mô hình triển khai thực tế.

Tài liệu tham khảo

[1] <https://bizflycloud.vn/tin-tuc/kubernetes-la-gi-vai-tro-cua-kubernetes-la-gi-20181015094513924.htm>, Kubernetes (K8s) là gì? Tìm hiểu cơ bản về Kubernetes, 9/6/2025

[2] <https://kubernetes.io/vi/docs/concepts/overview/what-is-kubernetes/>, Kubernetes là gì ?, 9/6/2025

[3] <https://topdev.vn/blog/kubernetes-la-gi/#3d6a>, Kubernetes là gì? Cùng tìm hiểu cách hoạt động của K8s, 9/6/2025

[4] <https://kubernetes.io/docs/concepts/overview/components/>, Kubernetes Components, 9/6/2025

[5]

https://fptcloud.com/kubernetes-la-gi/#3_Uu_va_nhuoc_diem_cua_Kubernetes, Kubernetes (K8s) là gì? Chức năng và cơ chế hoạt động chi tiết, 9/6/2025