

VIETTEL DIGITAL TALENT 2025

BÁO CÁO

BÀI TẬP LỚN CUỐI KỲ  
LĨNH VỰC CLOUD - GIAI ĐOẠN 1

NGƯỜI HIỆN THỰC: NGUYỄN TRUNG VƯƠNG

TP.HỒ CHÍ MINH, 06/2025

# MỤC LỤC

## DANH MỤC HÌNH VẼ

iv

|          |  |          |
|----------|--|----------|
| <b>1</b> | <b>Triển khai Kubernetes (1đ)</b>  | <b>1</b> |
| 1.1      | Yêu cầu . . . . .  | 1        |
| 1.2      | Output . . . . .   | 1        |
| 1.2.1    | Tài liệu cài đặt . . . . .   | 1        |
| 1.2.2    | Ảnh chụp Log của các lệnh kiểm tra hệ thống . . . . .                      | 3        |
| <b>2</b> | <b>Triển khai web application sử dụng các DevOps tools &amp; practices</b> | <b>5</b> |
| 2.1      | K8S Helm Chart (1.5đ) . . . . .  | 5        |
| 2.1.1    | Yêu cầu 1 . . . . .  | 5        |
| 2.1.2    | Output 1 . . . . .   | 5        |
| 2.1.3    | Yêu cầu 2 . . . . .  | 8        |
| 2.1.4    | Output 2 . . . . .   | 9        |
| 2.2      | CI/CD (1.5đ) . . . . .   | 12       |
| 2.2.1    | Yêu cầu . . . . .  | 12       |
| 2.2.2    | Output . . . . .   | 13       |
| 2.3      | Monitoring (1.5đ) . . . . .  | 20       |
| 2.3.1    | Yêu cầu . . . . .  | 20       |
| 2.3.2    | Output . . . . .   | 20       |
| 2.4      | Logging (1.5đ) . . . . .   | 23       |
| 2.4.1    | Yêu cầu . . . . .  | 23       |
| 2.4.2    | Output . . . . .   | 24       |
| 2.5      | Security . . . . .   | 27       |
| 2.5.1    | Yêu cầu 1 (1đ) . . . . .   | 27       |
| 2.5.2    | Output 1 . . . . .   | 27       |
| 2.5.3    | Yêu cầu 2 (1đ) . . . . .   | 28       |
| 2.5.4    | Output . . . . .   | 29       |
| 2.5.5    | Yêu cầu 3 (1đ) . . . . .   | 30       |

|                          |    |
|--------------------------|----|
| 2.5.6 Output 3 . . . . . | 30 |
|--------------------------|----|

# DANH MỤC HÌNH VẼ

|  |    |
|--|----|
| Hình 1.1. Ảnh chụp khi chạy ansible hoàn tất. . . . .  | 3  |
| Hình 1.2. Ảnh chụp khi các lệnh kiểm tra hệ thống. . . . .   | 3  |
| Hình 2.3. Ảnh chụp giao diện màn hình hệ thống ArgoCD khi truy cập qua trình duyệt trình duyệt. . . . .  | 5  |
| Hình 2.4. Ảnh chụp giao diện màn hình hệ thống Jenkins khi truy cập qua trình duyệt trình duyệt. . . . . | 8  |
| Hình 2.5. Ảnh chụp giao diện màn hình hệ thống ArgoCD trên trình duyệt. .                                | 11 |
| Hình 2.6. Ảnh chụp giao diện màn hình trình duyệt khi truy cập vào Web URL.                              | 11 |
| Hình 2.7. Ảnh chụp giao diện màn hình trình duyệt khi truy cập vào API URL.                              | 12 |
| Hình 2.8. Luồng CI/CD. . . . .   | 13 |
| Hình 2.9. Luồng CI/CD khi tạo tag mới trên repo app. . . . .   | 13 |
| Hình 2.10. web application argoCD. . . . .   | 13 |
| Hình 2.11. log chứng minh jenkin đã chạy đúng. . . . .   | 14 |
| Hình 2.12. Docker image được tạo từ CD. . . . .  | 18 |
| Hình 2.13. Hình ảnh diff khi argoCD phát hiện thay đổi ở config repo. . . . .                            | 18 |
| Hình 2.14. Hình ảnh app trước khi sửa code. . . . .  | 19 |
| Hình 2.15. Hình ảnh app sau khi sửa code. . . . .  | 19 |
| Hình 2.16. Add module prometheus vào NestJS. . . . .   | 20 |
| Hình 2.17. prometheus.yml. . . . .   | 22 |
| Hình 2.18. Hình ảnh khi truy cập vào Prometheus UI thông qua trình duyệt. .                              | 22 |
| Hình 2.19. Hình ảnh danh sách target của App được giám sát bởi Prometheus.                               | 23 |
| Hình 2.20. Hình ảnh metric được expose từ api. . . . .   | 23 |
| Hình 2.21. Trạng thái của các pods. . . . .  | 25 |
| Hình 2.22. Giao diện kibana. . . . .   | 26 |
| Hình 2.23. Log từ kibana. . . . .  | 26 |
| Hình 2.24. Cài đặt hosts. . . . .  | 28 |
| Hình 2.25. Kết quả api trả về khi đăng nhập với admin. . . . .   | 29 |
| Hình 2.26. Kết quả api trả về khi đăng nhập với user. . . . .  | 29 |
| Hình 2.27. Kết quả khi không đăng nhập nhưng gọi API. . . . .  | 29 |

|  |    |
|--|----|
| Hình 2.28. Kết quả gọi API không dành cho admin. . . . . | 30 |
| Hình 2.29. Kết quả gọi API không dành cho user. . . . .  | 30 |
| Hình 2.30. Kết quả kiểm tra. . . . .                     | 31 |

# 1 Triển khai Kubernetes (1d)

## 1.1 Yêu cầu

- Triển khai được Kubernetes thông qua công cụ minikube trên 1 node: 0.5 điểm

Hoặc

- Triển khai được Kubernetes thông qua công cụ kubeadm hoặc kubespray lên 1 master node VM + 1 worker node VM: 1 điểm

## 1.2 Output

### 1.2.1 Tài liệu cài đặt

Tài liệu Kubespray: <https://kubespray.io>

### Bảng các máy sử dụng

| Máy     | Vai trò                                      | RAM |
|---------|--|-----|
| ansible | Node control, cài ansible, docker, kubespray | 2GB |
| master1 | master + etcd                                | 2GB |
| worker1 | worker                                       | 4GB |
| worker2 | worker                                       | 4GB |

Bảng 1.1 Bảng các máy sử dụng.

\*\*Lưu ý\*\* : Đã cấu hình SSH từ thông tin Ansible sang master1 và worker1, worker2 sử dụng ssh-copy-id.

### Các lệnh cài đặt

- Clone Kubespray

```
git clone https://github.com/kubernetes-sigs/kubespray  
cd kubespray
```

- Chạy container Kubespray trên máy **amd64 hoặc x86**

```
docker run --rm -it \  
  --mount type=bind,source="${pwd}"/inventory/sample,dst=/inventory \  
  \  
  --mount type=bind,source="${HOME}/.ssh/id_rsa,dst=/root/.ssh/ \  
    id_rsa \  
  quay.io/kubespray/kubespray:v2.28.0 bash
```

- Chạy container Kubespray trên máy **arm64**

```

docker run --privileged --rm tonistiigi/binfmt --install all

docker run --rm -it --platform=linux/amd64 --mount type=bind,source=
"${pwd}"/inventory/sample,dst=/inventory --mount type=bind,source
="${HOME}"/.ssh/id_rsa,dst=/root/.ssh/id_rsa quay.io/kubespray/
kubespray:v2.28.0 bash

```

- **Sửa nội dung file /inventory/inventory.ini**

```

[kube_control_plane]
master1 ansible_host=<ip_master> ansible_port=<port_ssh_sang_master1
> ansible_user=<user_c_quyn_ln_root> #vit lin 1 dng

[etcd:children]
kube_control_plane

[kube_node]
worker1 ansible_host=<ip_worker> ansible_port=<port_ssh_sang_worker1
> ansible_user=<user_c_quyn_ln_root>

worker2 ansible_host=<ip_worker> ansible_port=<port_ssh_sang_worker1
> ansible_user=<user_c_quyn_ln_root>

[k8s_cluster:children]
kube_control_plane
kube_node

```

- **Chạy cài đặt K8s**

```

ansible-playbook -i /inventory/inventory.ini cluster.yml --become --
ask-pass --ask-become-pass

```

\*\* Lưu ý \*\*: Nếu đã kết nối ssh không cần –ask-pass và tại đây để thuận tiện các password của các máy đều giống nhau.

- **Cài kubectl (trên Ansible hoặc worker1)**

Link: <https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/>

```

// amd64
curl -LO "https://dl.k8s.io/release/$(curl -sL https://dl.k8s.io/
release/stable.txt)/bin/linux/amd64/kubectl"
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

//arm64

```

```

curl -LO "https://dl.k8s.io/release/$(curl -sL https://dl.k8s.io/
release/stable.txt)/bin/linux/arm64/kubectl"
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

```

## • Cấu hình kubeconfig

Trên master1

```
sudo cat /etc/kubernetes/admin.conf
```

- Copy file kubeconfig về máy Ansible, sửa phần ip của API Server, lưu vào file, ví dụ k8s-config.yaml:

```
server: https://127.0.0.1:6443 => https://<ip_master1>:6443
```

- Gán env variable KUBECONFIG:

```

export KUBECONFIG=k8s-config.yaml
kubectl get nodes -o wide
kubectl get pods -A -o wide

```

### 1.2.2 Ảnh chụp Log của các lệnh kiểm tra hệ thống

```

PLAY RECAP ****
master1 : ok=644 changed=145 unreachable=0 failed=0 skipped=1000 rescued=0 ignored=6
worker1 : ok=442 changed=89 unreachable=0 failed=0 skipped=630 rescued=0 ignored=1

```

Hình 1.1. Ảnh chụp khi chạy ansible hoàn tất.

```

[vun@ansible:~/kubespray$ kubectl get nodes -o wide
NAME      STATUS   ROLES    AGE     VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE       KERNEL-VERSION   CONTAINER-RUNTIME
master1   Ready    control-plane   9m55s   v1.32.5   192.168.1.14   <none>        Ubuntu 22.04.5 LTS   5.15.0-119-generic   containerd://2.0.5
worker1   Ready    <none>    9m14s   v1.32.5   192.168.1.18   <none>        Ubuntu 22.04.5 LTS   5.15.0-142-generic   containerd://2.0.5
[vun@ansible:~/kubespray$ kubectl get pods -A -o wide
NAMESPACE   NAME          READY   STATUS    RESTARTS   AGE     IP           NODE   NOMINATED-NODE   READINESS   GATES
kube-system  calico-kube-controllers-588dd6f6c9-h9nl2   1/1    Running   0          8m30s   192.233.105.129   worker1   <none>        <none>
kube-system  calico-node-5q5hl   1/1    Running   0          8m61s   192.168.1.18    worker1   <none>        <none>
kube-system  calico-node-fkkkr   1/1    Running   0          8m61s   192.168.1.14    master1   <none>        <none>
kube-system  coredns-5c54f84c97-56b8w   1/1    Running   0          8m23s   10.233.104.65   master1   <none>        <none>
kube-system  coredns-5c54f84c97-xxxhv   1/1    Running   0          8m13s   10.233.105.131   worker1   <none>        <none>
kube-system  dns-autoscaler-56ccb45595c-8t596   1/1    Running   0          8m21s   10.233.105.130   worker1   <none>        <none>
kube-system  kube-apiserver-master1   1/1    Running   1          10m     192.168.1.14    master1   <none>        <none>
kube-system  kube-controller-manager-master1   1/1    Running   2          10m     192.168.1.14    master1   <none>        <none>
kube-system  kube-proxy-2s46b   1/1    Running   0          9m19s   192.168.1.14    master1   <none>        <none>
kube-system  kube-proxy-w8zb6   1/1    Running   0          9m19s   192.168.1.18    worker1   <none>        <none>
kube-system  kube-scheduler-master1   1/1    Running   1          10m     192.168.1.14    master1   <none>        <none>
kube-system  nginx-proxy-worker1   1/1    Running   0          9m22s   192.168.1.18    worker1   <none>        <none>
kube-system  nodelocaldns-hqtds   1/1    Running   0          8m16s   192.168.1.14    master1   <none>        <none>
kube-system  nodelocaldns-wqfkf   1/1    Running   0          8m16s   192.168.1.18    worker1   <none>        <none>
vun@ansible:~/kubespray$ ]

```

Hình 1.2. Ảnh chụp khi các lệnh kiểm tra hệ thống.

Trong quá trình cài đặt, có các vấn đề phát sinh như sau:

- Khi sử dụng máy ảo mạng bridge có lúc IP của master thay đổi khi đó cần phải cập nhật lại ip của master tại file `/etc/kubernetes/manifests/kube-apiserver.yaml`

- Reset etcd và systemd

```
sudo systemctl daemon-reexec  
sudo systemctl daemon-reload  
sudo systemctl restart etcd
```

- Trở lại bước 1.2.1 và làm lại các bước trên.

## 2 Triển khai web application sử dụng các DevOps tools & practices

### 2.1 K8S Helm Chart (1.5đ)

#### 2.1.1 Yêu cầu 1

- Cài đặt ArgoCD lên Kubernetes Cluster, expose được ArgoCD qua NodePort
- Cài đặt Jenkins lên Kubernetes Cluster, expose được Jenkins qua NodePort

#### 2.1.2 Output 1

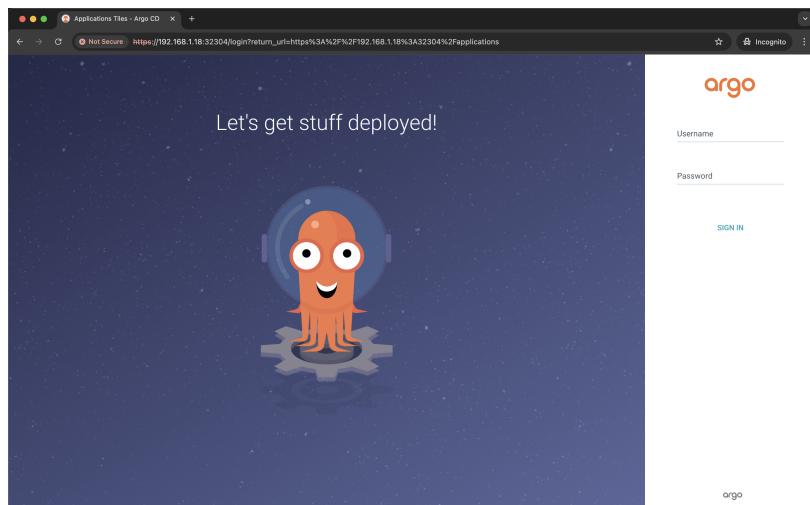
- File manifests sử dụng để triển khai **ArgoCD** lên K8S Cluster: <https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml>

```
kubectl apply -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml
```

- Chuyển ClusterIP thành NodePort để truy cập **ArgoCD** từ trình duyệt bên ngoài (browser).

```
kubectl patch svc argocd-server -n default -p '{"spec": {"type": "NodePort"}}'
```

- Ảnh chụp giao diện màn hình hệ thống **ArgoCD** khi truy cập qua trình duyệt trình duyệt.



Hình 2.3. Ảnh chụp giao diện màn hình hệ thống ArgoCD khi truy cập qua trình duyệt trình duyệt.

- Lấy mật khẩu của **argo-cd** (*username = admin*)

```
kubectl -n default get secret argocd-initial-admin-secret -o jsonpath='{.data.password}' | base64 -d && echo
```

- Manifest sử dụng để triển khai Jenkins lên K8S Cluster

```

apiVersion: v1
kind: Namespace
metadata:
  name: jenkins
---
apiVersion: v1
kind: PersistentVolume
metadata:
  name: jenkins-pv
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  hostPath:
    path: /data/jenkins
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: jenkins-pvc
  namespace: jenkins
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: jenkins
  namespace: jenkins
spec:
  replicas: 1
  selector:
    matchLabels:
      app: jenkins
  template:

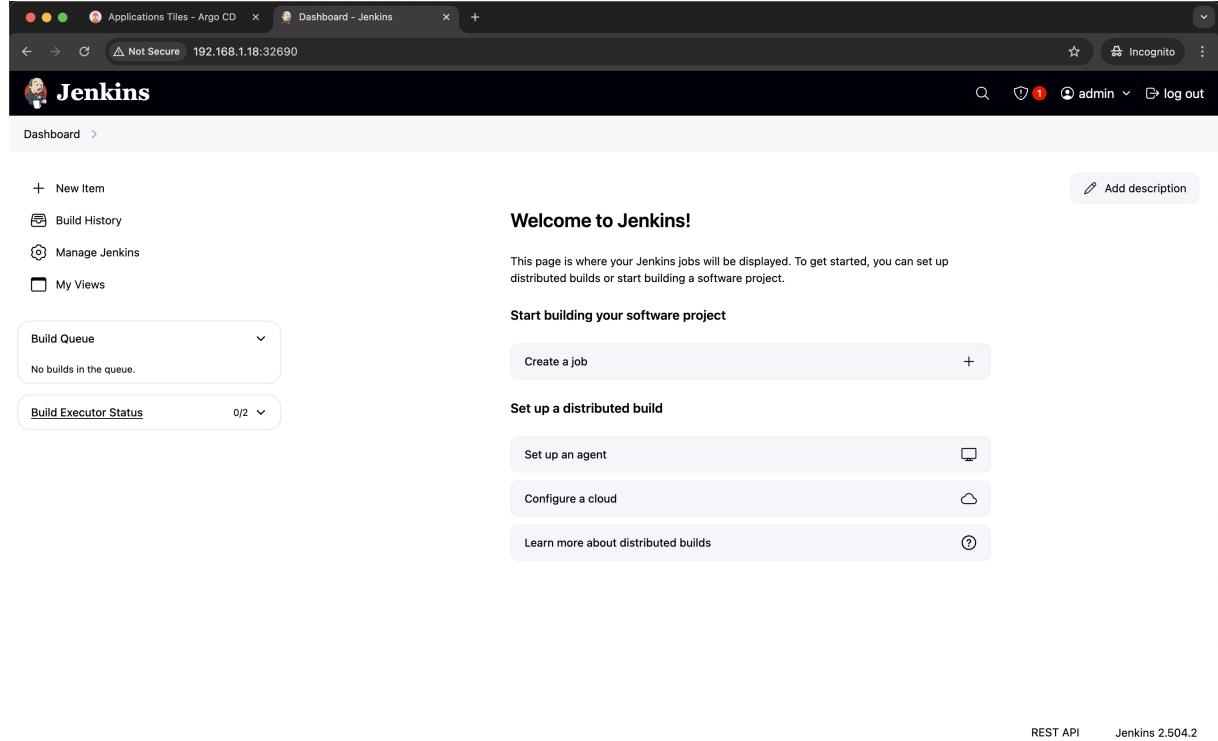
```

```

metadata:
  labels:
    app: jenkins
spec:
  containers:
    - name: jenkins
      image: jenkins/jenkins:lts
      ports:
        - containerPort: 8080
        - containerPort: 50000
      volumeMounts:
        - name: jenkins-storage
          mountPath: /var/jenkins_home
      securityContext:
        runAsUser: 0
  volumes:
    - name: jenkins-storage
      persistentVolumeClaim:
        claimName: jenkins-pvc
---
apiVersion: v1
kind: Service
metadata:
  name: jenkins-service
  namespace: jenkins
spec:
  type: NodePort
  selector:
    app: jenkins
  ports:
    - name: http
      port: 8080
      targetPort: 8080
      nodePort: 32000
    - name: agent
      port: 50000
      targetPort: 50000
      nodePort: 32001

```

- Ánh chụp giao diện màn hình hệ thống **Jenkins** khi truy cập qua trình duyệt trình duyệt



Hình 2.4. Ánh chụp giao diện màn hình hệ thống Jenkins khi truy cập qua trình duyệt trình duyệt.

- **Lấy password trong jenkins**

```
kubectl exec -n jenkins -it <jenkins-name> -- cat /var/jenkins_home/secrets/initialAdminPassword
```

### 2.1.3 Yêu cầu 2

- Viết hoặc tìm mẫu Helm Chart cho app bất kỳ, để vào 1 folder riêng trong repo app.
- Tạo Repo Config cho app trên, trong repo này chứa các file values.yaml với nội dung của cá file values.yaml là các config cần thiết để chạy ứng dụng trên k8s bằng Helm Chart .

## 2.1.4 Output 2

### Các Helm Chart sử dụng để triển khai app lên K8S Cluster

- API helm chart: [API helm chart](#)
- Web helm chart: [Web helm chart](#)

### Các file values.yaml trong config repo của app

- Config-api: [Config-api](#)
- Config-web: [Config-web](#)

**Manifest của ArgoCD Application:** Sử dụng tính năng multiple sources của ArgoCD để triển khai các service web và api service lên K8S Cluster

- **Manifest cho frontend**

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: web
  namespace: default
spec:
  project: default
  destination:
    server: https://kubernetes.default.svc
    namespace: vdt-final
  sources:
    - repoURL: "https://github.com/nguyenvuong310/student-management-frontend.git"
      targetRevision: main
      path: helm-chart
      helm:
        valueFiles:
          - $values/values.yaml
    - repoURL: "https://github.com/nguyenvuong310/config-web.git"
      targetRevision: main
      ref: values
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
```

- **Manifest cho backend**

```
apiVersion: argoproj.io/v1alpha1
```

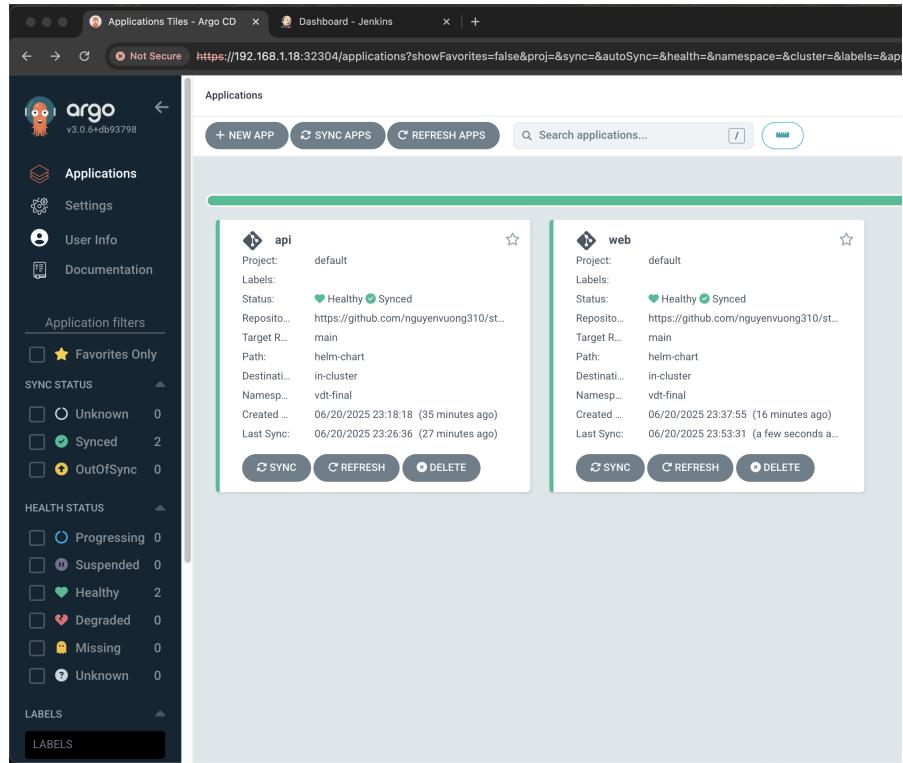
```
kind: Application
metadata:
  name: api
  namespace: argocd
spec:
  project: default
  destination:
    server: https://kubernetes.default.svc
    namespace: vdt-final

  sources:
    - repoURL: "https://github.com/nguyenvuong310/student-management-backend.git"
      targetRevision: main
      path: helm-chart
      helm:
        valueFiles:
          - $values/values.yaml
    - repoURL: "https://github.com/nguyenvuong310/config-api.git"
      targetRevision: main
      ref: values

  syncPolicy:
    automated:
      selfHeal: true
      prune: true
```

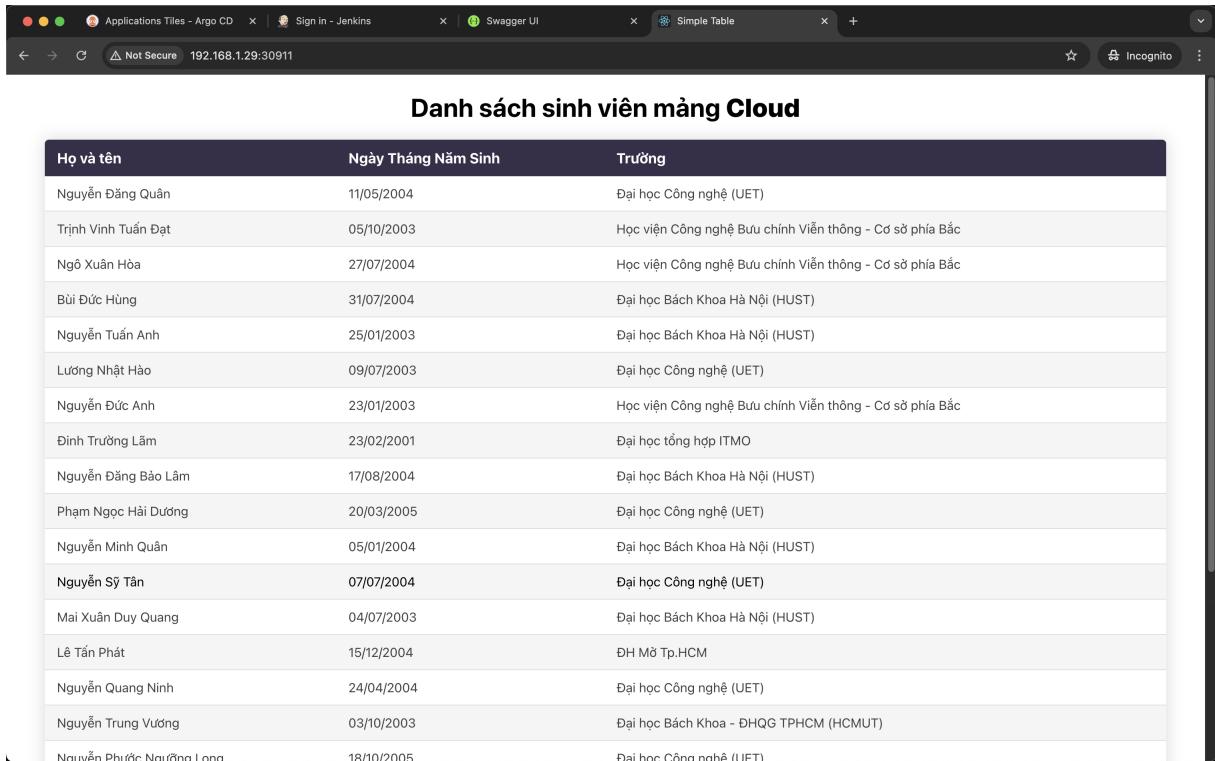
\*\*Lưu ý\*\*: Cần tạo thêm file manifest cho database posgres để backend có thể kết nối.

## Ảnh chụp giao diện màn hình hệ thống ArgoCD trên trình duyệt



Hình 2.5. Ảnh chụp giao diện màn hình hệ thống ArgoCD trên trình duyệt.

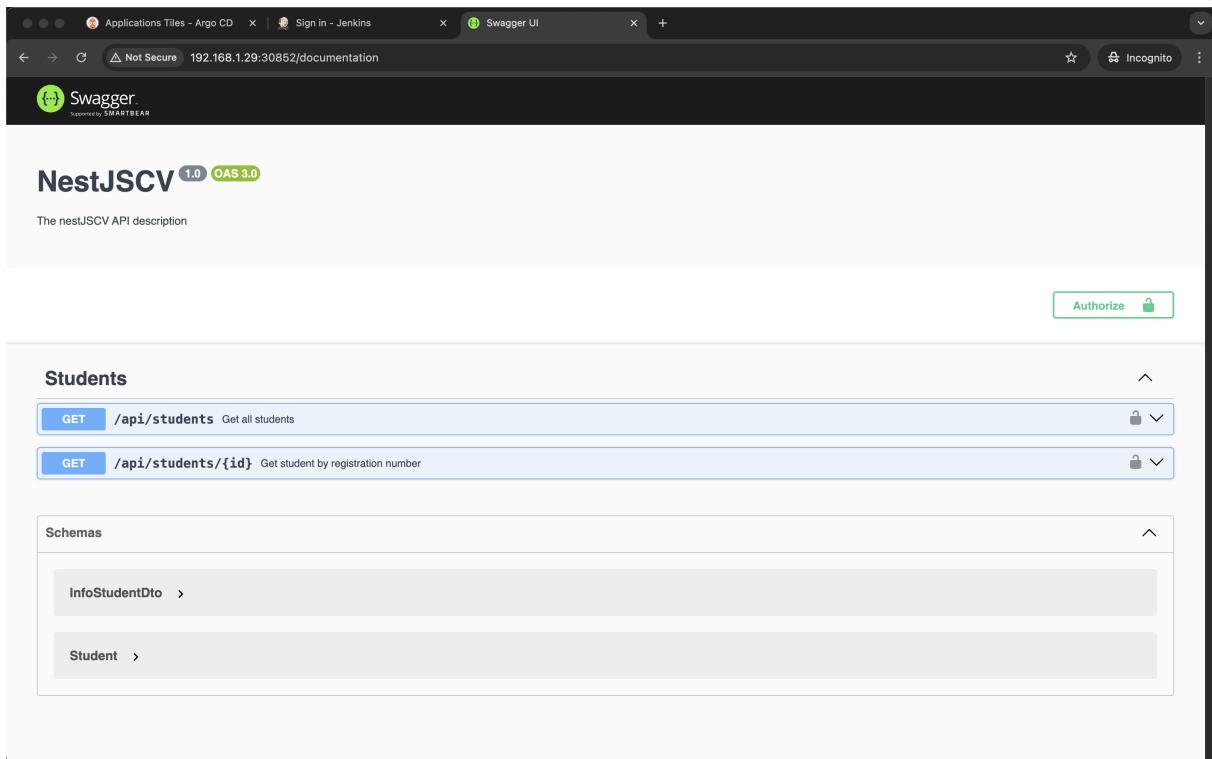
## Ảnh chụp giao diện màn hình trình duyệt khi truy cập vào Web URL



| Họ và tên                 | Ngày Tháng Năm Sinh | Trưởng   |
|---------------------------|---------------------|--|
| Nguyễn Đăng Quân          | 11/05/2004          | Đại học Công nghệ (UET)                                  |
| Trịnh Vinh Tuấn Đạt       | 05/10/2003          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Ngô Xuân Hòa              | 27/07/2004          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Bùi Đức Hùng              | 31/07/2004          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Nguyễn Tuấn Anh           | 25/01/2003          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Lương Nhật Hào            | 09/07/2003          | Đại học Công nghệ (UET)                                  |
| Nguyễn Đức Anh            | 23/01/2003          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Đinh Trường Lâm           | 23/02/2001          | Đại học tổng hợp ITMO                                    |
| Nguyễn Đăng Bảo Lâm       | 17/08/2004          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Phạm Ngọc Hải Dưỡng       | 20/03/2005          | Đại học Công nghệ (UET)                                  |
| Nguyễn Minh Quân          | 05/01/2004          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Nguyễn Sỹ Tân             | 07/07/2004          | Đại học Công nghệ (UET)                                  |
| Mai Xuân Duy Quang        | 04/07/2003          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Lê Tân Phát               | 15/12/2004          | ĐH Mở Tp.HCM   |
| Nguyễn Quang Ninh         | 24/04/2004          | Đại học Công nghệ (UET)                                  |
| Nguyễn Trung Vương        | 03/10/2003          | Đại học Bách Khoa - ĐHQG TPHCM (HCMUT)                   |
| Nguyễn Phi Út Nguyễn Lona | 18/10/2005          | Đại học Công nghệ (UET)                                  |

Hình 2.6. Ảnh chụp giao diện màn hình trình duyệt khi truy cập vào Web URL.

## Ảnh chụp giao diện màn hình trình duyệt khi truy cập vào API URL



Hình 2.7. *Ảnh chụp giao diện màn hình trình duyệt khi truy cập vào API URL.*

## 2.2 CI/CD (1.5đ)

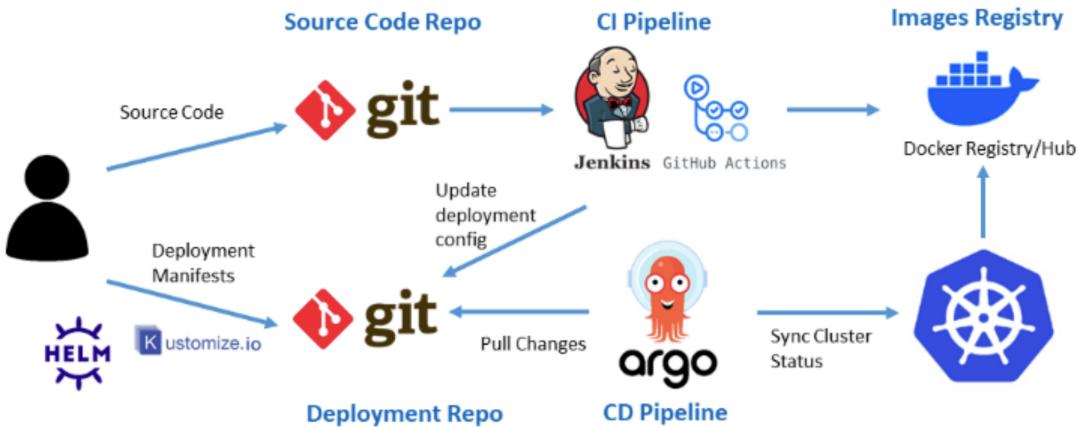
### 2.2.1 Yêu cầu

Viết 1 luồng CI/CD cho app, khi có thay đổi từ source code, 1 tag mới được tạo ra trên repo này thì luồng CI/CD tương ứng của repo đó thực hiện các công việc sau:

- Sửa code trong source code
- Thực hiện build source code trên jenkin bằng docker với image tag là tag name đã được tạo ra trên gitlab/github và push docker image sau khi build xong lên Docker Hub
- Sửa giá trị Image version trong file values.yaml trong config repo và push thay đổi lên config repo.
- Cấu hình ArgoCD tự động triển khai lại web Deployment và api Deployment khi có sự thay đổi trên config repo.

## 2.2.2 Output

### Luồng CI/CD



Hình 2.8. Luồng CI/CD.

### Output log của luồng CI/CD khi tạo tag mới trên repo app

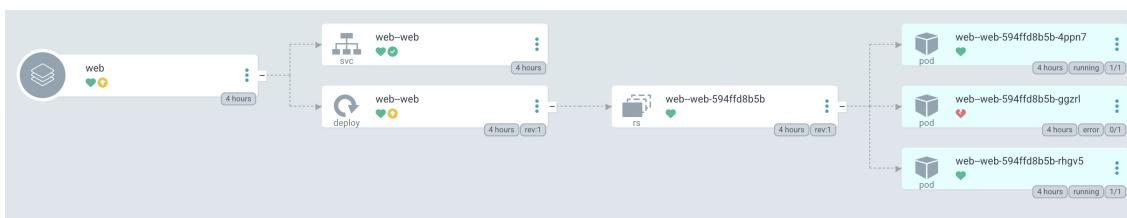
```

graph LR
    Start((Start)) --> CheckoutSCM[Checkout SCM]
    CheckoutSCM --> BuildPush[Build & Push with...]
    BuildPush --> CheckConfig[Checkout Config Repo]
    CheckConfig --> UpdateImageTag[Update Image Tag in values.yaml]
    UpdateImageTag --> CommitPush[Commit and Push...]
    CommitPush --> PostActions[Post Actions]
    PostActions --> End((End))
    
```

Log details:

- Checkout SCM: 3,0 giây
- 1. Checkout Source Code: 1,4 giây
- 2. Build & Push with Kaniko: 4 phút
- 3. Checkout Config Repo: 5,3 giây
- 4. Update Image Tag in values.yaml: 2,0
- 5. Commit and Push Changes: 2,0
- Post Actions: 72mili giây

Hình 2.9. Luồng CI/CD khi tạo tag mới trên repo app.



Hình 2.10. web application argoCD.

## Show log chứng minh jenkin đã chạy đúng

```
✓ Pipeline đã kết thúc.  
[Pipeline] cleanWs  
[WS-CLEANUP] Deleting project workspace...  
[WS-CLEANUP] Deferred wipeout is used...  
[WS-CLEANUP] done  
[Pipeline] }  
[Pipeline] // stage  
[Pipeline] }  
[Pipeline] // withEnv  
[Pipeline] }  
[Pipeline] // withCredentials  
[Pipeline] }  
[Pipeline] // container  
[Pipeline] }  
[Pipeline] // withEnv  
[Pipeline] }  
Agent vdt-final-25-0gskm-9kk6k-fw3x6 was deleted, but do not have a node body to cancel  
[Pipeline] // node  
[Pipeline] }  
[Pipeline] // podTemplate  
[Pipeline] End of Pipeline  
Finished: SUCCESS
```

Hình 2.11. log chứng minh jenkin đã chạy đúng.

## Jenkin file cấu hình các luồng

```
pipeline {  
    agent {  
        kubernetes {  
            cloud 'test-k8s-cloud'  
            defaultContainer 'kaniko'  
            yaml """  
apiVersion: v1  
kind: Pod  
spec:  
    containers:  
        - name: shell  
          image: alpine/git  
          command: [sleep]  
          args: ["999999"]  
    volumeMounts:  
        - name: workspace-volume  
          mountPath: /home/jenkins/agent  
  
        - name: kaniko  
          image: gcr.io/kaniko-project/executor:debug
```

```

    imagePullPolicy: Always
    command: [sleep]
    args: ["999999"]
    volumeMounts:
      - name: workspace-volume
        mountPath: /home/jenkins/agent
      - name: docker-config
        mountPath: /kaniko/.docker/
  volumes:
    - name: workspace-volume
      emptyDir: {}
    - name: docker-config
      secret:
        secretName: dockerhub-credentials
        items:
          - key: .dockerconfigjson
            path: config.json
    ...
  }
}

environment {
  DOCKER_IMAGE_NAME = 'vuong676/listvdt-frontend-web'
  G_TOKEN = credentials('github-token-write')
}

parameters {
  string(name: 'TAG_NAME', defaultValue: '', description: 'Tag name (e.g
    . 1.0)')
}

stages {
  stage('1. Checkout Source Code') {
    steps {
      container('kaniko') {
        git(
          url: 'https://github.com/nguyenvuong310/student-management-
            frontend-.git',
          branch: 'main',
          credentialsId: 'github-cred'
        )
        echo "  Checkout thnh cng ."
      }
    }
  }
}

```

```

        sh "ls -la"
    }
}
}

stage('2. Build & Push with Kaniko') {
when {
    expression { return params.TAG_NAME?.trim() }
}
steps {
    container('kaniko') {
        script {
            def fullTag = "${DOCKER_IMAGE_NAME}:${TAG_NAME}"
            echo "      Building and pushing image: ${fullTag}"
            sh """
                /kaniko/executor \
                --context 'pwd' \
                --dockerfile 'pwd'/dockerfile \
                --destination=${fullTag}
            """
        }
    }
}
}

stage('3. Checkout Config Repo') {
steps {
    container('shell') {
        dir('config-repo') {
            git url: "https://github.com/nguyenvuong310/config-web.git",
                branch: "main",
                credentialsId: 'github-token-write'
        }
    }
}
}

stage('4. Update Image Tag in values.yaml') {
steps {
    container('shell') {
        dir('config-repo') {
            sh """
                ls -la
            """
        }
    }
}
}

```

```

        sed -i 's/^ *tag:.*/ tag: ${TAG_NAME}/g' values.yaml
    """
}

}

}

}

stage('5. Commit and Push Changes') {
    steps {
        container('shell') {
            dir('config-repo') {
                withEnv(["GITHUB_TOKEN=${G_TOKEN}"]) {
                    sh '''
                        git config --global --add safe.directory $(pwd)
                        git add values.yaml
                        git config --global user.email "trungvuong2169@gmail.com"
                        git config --global user.name "nguyenvuong310"
                        git commit -m "Update image tag to $TAG_NAME" || echo "Nothing to
commit"
                        git push https://$GITHUB_TOKEN@github.com/nguyenvuong310/config-web.
git HEAD:main
'''
                }
            }
        }
    }
}

post {
    always {
        echo 'Pipeline done.'
        cleanWs()
    }
}
}

```

**\*\* Lưu ý \*\***: jenkins chưa bật github webhook (Vì github chỉ nhận https hoặc public ip -> có thể sử dụng ngrok để chuyển đổi cổng thành https) để tự động build. Do đó cần phải build thủ công trên giao diện.

## Docker image

The screenshot shows the Docker Hub interface with the 'Tags' tab selected. It lists three Docker images:

- 3.0**: Last pushed 2 minutes ago by [vuong676](#). Digest: [437c978c352c](#), OS/ARCH: linux/arm64/v8, Last pull: less than 1 day, Compressed size: 223.03 MB. Pull command: docker pull vuong676/listvdt-frontend-web:3.0
- 2.0**: Last pushed about 1 hour ago by [vuong676](#). Digest: [9564039b6525](#), OS/ARCH: linux/arm64/v8, Last pull: less than 1 day, Compressed size: 223.06 MB. Pull command: docker pull vuong676/listvdt-frontend-web:2.0
- latest**: Last pushed about 13 hours ago by [vuong676](#). Digest: [af43d26234ed](#), OS/ARCH: linux/arm64/v8, Last pull: less than 1 day, Compressed size: 191.25 MB. Pull command: docker pull vuong676/listvdt-frontend-web:latest

Hình 2.12. Docker image được tạo từ CD.

## Hình ảnh diff khi argoCD phát hiện thay đổi ở config repo

The screenshot shows the argoCD interface with the 'DIFF' tab selected. It displays a comparison of two configuration files, specifically focusing on the 'replicas' field in the 'spec' section of the 'apps/Deployment/vdt-final/web--web' resource. The changes are highlighted with red and green bars:

```
apps/Deployment/vdt-final/web--web
103 spec:
104   progressDeadlineSeconds: 600
105   replicas: 1
106   revisionHistoryLimit: 10
107   selector:
108
109 spec:
110   progressDeadlineSeconds: 600
111   replicas: 1
112   revisionHistoryLimit: 10
113   selector:
```

Compact diff  Inline diff

Hình 2.13. Hình ảnh diff khi argoCD phát hiện thay đổi ở config repo.

## Hình ảnh app trước khi sửa code và sau khi sửa code.

| Họ và tên                     | Ngày Tháng Năm Sinh | Trường   |
|-------------------------------|---------------------|--|
| Nguyễn Đăng Quân              | 11/05/2004          | Đại học Công nghệ (UET)                                  |
| Trịnh Vinh Tuấn Đạt           | 05/10/2003          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Ngô Xuân Hòa                  | 27/07/2004          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Bùi Đức Hùng                  | 31/07/2004          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Nguyễn Tuấn Anh               | 25/01/2003          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Lương Nhật Hào                | 09/07/2003          | Đại học Công nghệ (UET)                                  |
| Nguyễn Đức Anh                | 23/01/2003          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Đinh Trường Lâm               | 23/02/2001          | Đại học tổng hợp ITMO                                    |
| Nguyễn Đăng Bảo Lâm           | 17/08/2004          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Phạm Ngọc Hải Dương           | 20/03/2005          | Đại học Công nghệ (UET)                                  |
| Nguyễn Minh Quân              | 05/01/2004          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Nguyễn Sỹ Tân                 | 07/07/2004          | Đại học Công nghệ (UET)                                  |
| Mai Xuân Duy Quang            | 04/07/2003          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Lê Tấn Phát                   | 15/12/2004          | ĐH Mở Tp.HCM   |
| Nguyễn Quang Ninh             | 24/04/2004          | Đại học Công nghệ (UET)                                  |
| Nguyễn Trung Vương            | 03/10/2003          | Đại học Bách Khoa - ĐHQG TPHCM (HCMUT)                   |
| Nairvien Phuoc; Nairvina Long | 18/10/2005          | Đại học Công nghệ (UET)                                  |

Hình 2.14. Hình ảnh app trước khi sửa code.

| Họ và tên           | Ngày Tháng Năm Sinh | Trường   |
|---------------------|---------------------|--|
| Nguyễn Đăng Quân    | 11/05/2004          | Đại học Công nghệ (UET)                                  |
| Trịnh Vinh Tuấn Đạt | 05/10/2003          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Ngô Xuân Hòa        | 27/07/2004          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Bùi Đức Hùng        | 31/07/2004          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Nguyễn Tuấn Anh     | 25/01/2003          | Đại học Bách Khoa Hà Nội (HUST)                          |
| Lương Nhật Hào      | 09/07/2003          | Đại học Công nghệ (UET)                                  |
| Nguyễn Đức Anh      | 23/01/2003          | Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc |
| Đinh Trường Lâm     | 23/02/2001          | Đại học tổng hợp ITMO                                    |
| Nguyễn Đăng Bảo Lâm | 17/08/2004          | Đại học Bách Khoa Hà Nội (HUST)                          |

Hình 2.15. Hình ảnh app sau khi sửa code.

## 2.3 Monitoring (1.5đ)

### 2.3.1 Yêu cầu

- Expose metric của app ra 1 http path. Tham khảo: <https://github.com/korfuri/django-prometheus>
- Sử dụng ansible playbooks để triển khai container Prometheus server. Sau đó cấu hình prometheus add target giám sát các metrics đã expose ở trên.

### 2.3.2 Output

Để expose metric trên api, sử dụng nestjs-prometheus



```
You, 34 minutes ago | 1 author (You)
@Module({
  imports: [
    StudentsModule,
    ConfigModule.forRoot({
      isGlobal: true,
    }),
    PrometheusModule.register({
      path: '/metrics', // default là /metrics
    }),
  ],
})
```

Hình 2.16. Add module prometheus vào NestJS.

**Ansible playbooks để triển khai container Prometheus server :** chỉ cấu hình để cài đặt git, docker, và clone repository từ github.

```
- name: Update apt cache
  ansible.builtin.apt:
    update_cache: yes
    become_user: root

- name: Install required packages for Docker, Git
  ansible.builtin.apt:
    name:
      - git
      - apt-transport-https
      - ca-certificates
      - curl
      - gnupg
      - lsb-release
    state: present

- name: Add Docker's official GPG key
  ansible.builtin.apt_key:
    url: https://download.docker.com/linux/ubuntu/gpg
```

```

    state: present

- name: Add Docker repository
  ansible.builtin.apt_repository:
    repo: "deb https://download.docker.com/linux/ubuntu jammy stable"
    state: present
    filename: docker

- name: Update apt cache
  ansible.builtin.apt:
    update_cache: yes

- name: Install Docker Engine and Docker Compose plugin
  ansible.builtin.apt:
    name:
      - docker-ce
      - docker-ce-cli
      - containerd.io
      - docker-buildx-plugin
      - docker-compose-plugin
    state: present

- name: adding ubuntu to docker group
  user:
    name: ubuntu
    groups: docker
    append: yes

- name: Ensure Docker service is running
  ansible.builtin.service:
    name: docker
    state: started
    enabled: yes

- name: Clone GitHub repository
  ansible.builtin.git:
    repo: "https://github.com/nguyenvuong310/vdt-prometheus"
    dest: "{{ repo_dest }}"
    version: master

```

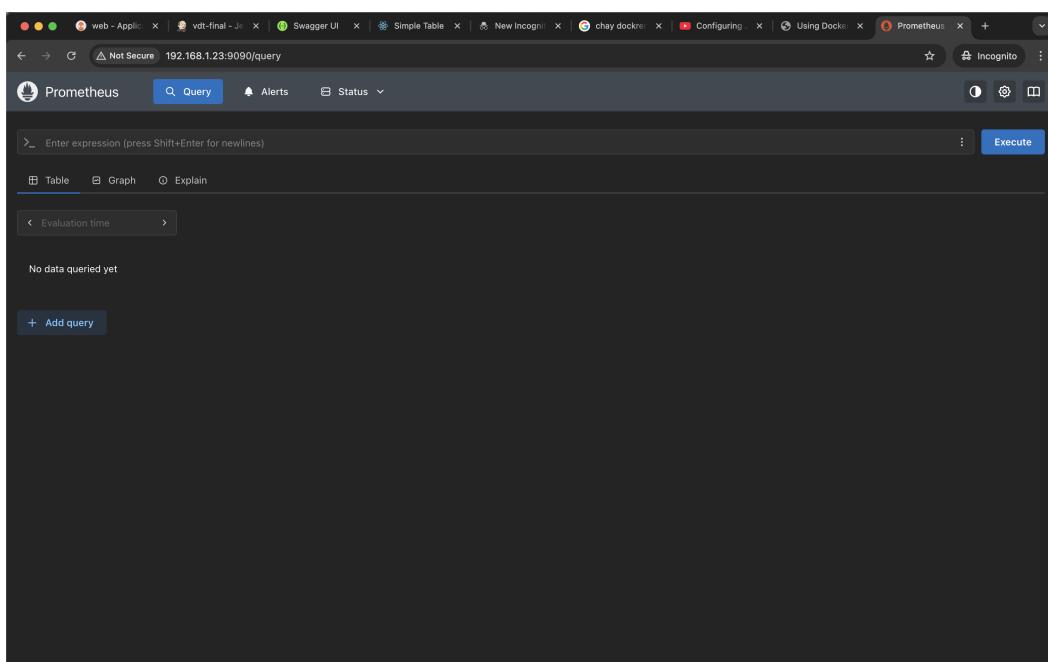
## Thay đổi nội dung trong file prometheus.yml để giám sát các ứng dụng

```
# A scrape configuration containing exactly one endpoint to scrape.
scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: [
        'localhost:9090',
        'localhost:9091'
      ]
  - job_name: 'node-exporter'
    static_configs:
      - targets: [
        'localhost:9100'
      ]
  - job_name: 'cadvisor'
    static_configs:
      - targets: [
        'localhost:8080'
      ]
```

Hình 2.17. *prometheus.yml*.

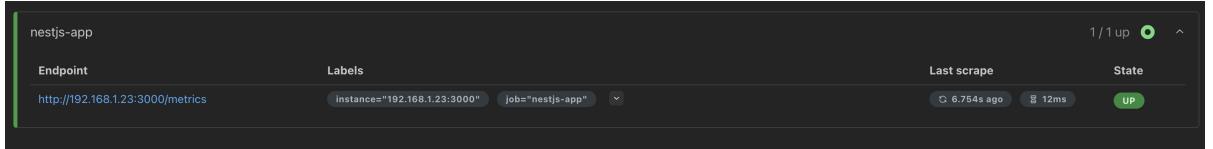
\*\* Lưu ý\*\*: Lưu ý: Vì đây là môi trường thử nghiệm nên các metric có thể được đọc bởi bất kỳ ứng dụng nào. Tuy nhiên, trong môi trường thực tế, cần cấu hình reverse proxy hoặc firewall để đảm bảo an toàn và giới hạn quyền truy cập vào các endpoint metric.

## Hình ảnh khi truy cập vào Prometheus UI thông qua trình duyệt

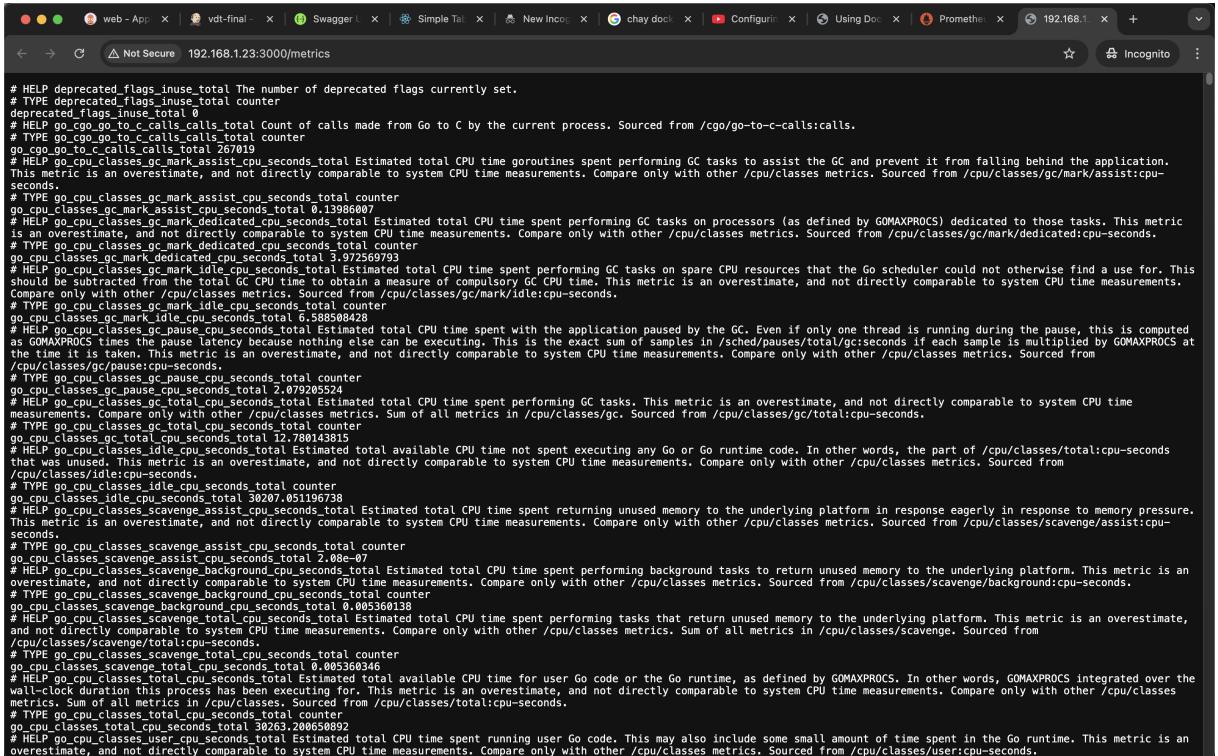


Hình 2.18. *Hình ảnh khi truy cập vào Prometheus UI thông qua trình duyệt*.

## Hình ảnh danh sách target của App được giám sát bởi Prometheus



Hình 2.19. Hình ảnh danh sách target của App được giám sát bởi Prometheus.



```
# HELP deprecated_flags_inuse_total The number of deprecated flags currently set.
# TYPE deprecated_flags_inuse_total counter
deprecated_flags_inuse_total 0
# HELP go_cgo_go_to_c_calls_calls total Count of calls made from Go to C by the current process. Sourced from /cgo/go-to-c-calls:calls.
# TYPE go_cgo_go_to_c_calls_calls_total counter
go_cgo_go_to_c_calls_calls_total 267019
# HELP go_gc_classes_gc_mark_assist_cpu_seconds_total Estimated total CPU time goroutines spent performing GC tasks to assist the GC and prevent it from falling behind the application.
# TYPE go_gc_classes_gc_mark_assist_cpu_seconds_total counter
go_gc_classes_gc_mark_assist_cpu_seconds_total 0.13986007
# HELP go_gc_classes_gc_dedicated_cpu_seconds_total Estimated total CPU time spent performing GC tasks on processors (as defined by GOMAXPROCS) dedicated to those tasks. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sourced from /cpu/classes/gc/mark/dedicated:cpu-seconds.
# TYPE go_gc_classes_gc_dedicated_cpu_seconds_total counter
go_gc_classes_gc_dedicated_cpu_seconds_total 0
# HELP go_gc_classes_gc_mark_idle_cpu_seconds_total Estimated total CPU time spent performing GC tasks on spare CPU resources that the Go scheduler could not otherwise find a use for. This should be subtracted from the total GC CPU time to obtain a measure of compulsory GC CPU time. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sourced from /cpu/classes/gc/mark/idle:cpu-seconds.
# TYPE go_gc_classes_gc_mark_idle_cpu_seconds_total counter
go_gc_classes_gc_mark_idle_cpu_seconds_total 6.588508428
# HELP go_gc_classes_gc_pause_cpu_seconds_total Estimated total CPU time spent with the application paused by the GC. Even if only one thread is running during the pause, this is computed as GOMAXPROCS times the pause duration because nothing else can be executing. This is the exact sum of samples in /sched/pauses/total/gc:seconds if each sample is multiplied by GOMAXPROCS at the time it is taken. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sourced from /cpu/classes/gc/pause:cpu-seconds.
# TYPE go_gc_classes_gc_pause_cpu_seconds_total counter
go_gc_classes_gc_pause_cpu_seconds_total 2.079205524
# HELP go_gc_classes_gc_total_cpu_seconds_total Estimated total CPU time spent performing GC tasks. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sum of all metrics in /cpu/classes/gc. Sourced from /cpu/classes/gc/total:cpu-seconds.
# TYPE go_gc_classes_gc_total_cpu_seconds_total counter
go_gc_classes_gc_total_cpu_seconds_total 12.789143915
# HELP go_gc_classes_idle_cpu_seconds_total Estimated total available CPU time not spent executing any Go or Go runtime code. In other words, the part of /cpu/classes/total:cpu-seconds that was unused. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sourced from /cpu/classes/idle:cpu-seconds.
# TYPE go_gc_classes_idle_cpu_seconds_total counter
go_gc_classes_idle_cpu_seconds_total 30207.051196738
# HELP go_gc_classes_scavenge_assist_cpu_seconds_total Estimated total CPU time spent returning unused memory to the underlying platform in response eagerly in response to memory pressure. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sourced from /cpu/classes/scavenge/assist:cpu-seconds.
# TYPE go_gc_classes_scavenge_assist_cpu_seconds_total counter
go_gc_classes_scavenge_assist_cpu_seconds_total 2.08e-07
# HELP go_gc_classes_scavenge_background_cpu_seconds_total Estimated total CPU time spent performing background tasks to return unused memory to the underlying platform. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sourced from /cpu/classes/scavenge/background:cpu-seconds.
# TYPE go_gc_classes_scavenge_background_cpu_seconds_total counter
go_gc_classes_scavenge_background_cpu_seconds_total 0.005360198
# HELP go_gc_classes_scavenge_total_cpu_seconds_total Estimated total CPU time spent performing tasks that return unused memory to the underlying platform. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sum of all metrics in /cpu/classes/scavenge. Sourced from /cpu/classes/scavenge/total:cpu-seconds.
# TYPE go_gc_classes_scavenge_total_cpu_seconds_total counter
go_gc_classes_scavenge_total_cpu_seconds_total 0.005360346
# HELP go_gc_classes_user_cpu_seconds_total Estimated total available CPU time for user Go code or the Go runtime, as defined by GOMAXPROCS. In other words, GOMAXPROCS integrated over the wall-clock duration this process has been executing for. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sum of all metrics in /cpu/classes. Sourced from /cpu/classes/total:cpu-seconds.
# TYPE go_gc_classes_total_cpu_seconds_total counter
go_gc_classes_total_cpu_seconds_total 30263.200650892
# HELP go_gc_classes_user_cpu_seconds_total Estimated total CPU time spent running user Go code. This may also include some small amount of time spent in the Go runtime. This metric is an overestimate, and not directly comparable to system CPU time measurements. Compare only with other /cpu/classes metrics. Sourced from /cpu/classes/user:cpu-seconds.
```

Hình 2.20. Hình ảnh metric được expose từ api.

## 2.4 Logging (1.5đ)

### 2.4.1 Yêu cầu

Sử dụng ansible playbooks để triển khai stack EFK (elasticsearch, fluentd, kibana), sau đó cấu hình logging cho web service và api service, đảm bảo khi có http request gửi vào web service hoặc api service thì trong các log mà các service này sinh ra, có ít nhất 1 log có các thông tin:

- Request Path(VD: /api1/1, /api2/3 ..)
- HTTP Method VD: (GET PUT POST...)
- Response Code: 302, 200, 202, 201...

#### 2.4.2 Output

**Ansible playbooks để triển khai container Prometheus server :** chỉ cấu hình để cài đặt git, docker, và clone repository từ github.

```
- name: Update apt cache
ansible.builtin.apt:
  update_cache: yes
  become_user: root

- name: Install required packages for Docker, Git
ansible.builtin.apt:
  name:
    - git
    - apt-transport-https
    - ca-certificates
    - curl
    - gnupg
    - lsb-release
  state: present

- name: Add Docker's official GPG key
ansible.builtin.apt_key:
  url: https://download.docker.com/linux/ubuntu/gpg
  state: present

- name: Add Docker repository
ansible.builtin.apt_repository:
  repo: "deb https://download.docker.com/linux/ubuntu jammy stable"
  state: present
  filename: docker

- name: Update apt cache
ansible.builtin.apt:
  update_cache: yes

- name: Install Docker Engine and Docker Compose plugin
ansible.builtin.apt:
  name:
    - docker-ce
    - docker-ce-cli
    - containerd.io
    - docker-buildx-plugin
    - docker-compose-plugin
```

```

state: present

- name: adding ubuntu to docker group
  user:
    name: ubuntu
    groups: docker
    append: yes

- name: Ensure Docker service is running
  ansible.builtin.service:
    name: docker
    state: started
    enabled: yes

- name: Clone GitHub repository
  ansible.builtin.git:
    repo: "https://github.com/nguyenvuong310/vdt-efk"
    dest: "{{ repo_dest }}"
    version: master

```

Xem README để cài đặt: <https://github.com/nguyenvuong310/vdt-efk/blob/main/README.md>

## Cài đặt thành công EFK

| NAME  | READY | STATUS  | RESTARTS     | AGE | IP             | NODE    | NOMINATED NODE | READINESS GATES |
|---|-------|---------|--------------|-----|----------------|---------|----------------|-----------------|
| argocd-application-controller-0                   | 1/1   | Running | 1 (158m ago) | 41h | 10.233.125.50  | worker2 | <none>         | <none>          |
| argocd-applicationset-controller-655cc58ff8-z4p65 | 1/1   | Running | 1 (158m ago) | 41h | 10.233.125.56  | worker2 | <none>         | <none>          |
| argocd-dex-server-7d9dfb4fb8-8pwtd                | 1/1   | Running | 1 (158m ago) | 41h | 10.233.125.60  | worker2 | <none>         | <none>          |
| argocd-notifications-controller-6c6848bc4c-52bxk  | 1/1   | Running | 1 (158m ago) | 41h | 10.233.125.52  | worker2 | <none>         | <none>          |
| argocd-redis-656c79549c-xtj5n                     | 1/1   | Running | 1 (158m ago) | 20h | 10.233.125.55  | worker2 | <none>         | <none>          |
| argocd-repo-server-856b7687fd9-5lb2g              | 1/1   | Running | 1 (158m ago) | 41h | 10.233.125.53  | worker2 | <none>         | <none>          |
| argocd-server-99c485944-cav46                     | 1/1   | Running | 1 (158m ago) | 41h | 10.233.125.54  | worker2 | <none>         | <none>          |
| es-cluster-0                                      | 1/1   | Running | 0            | 57m | 10.233.105.147 | worker1 | <none>         | <none>          |
| fluentd-rvgnw                                     | 1/1   | Running | 0            | 71m | 10.233.105.165 | worker1 | <none>         | <none>          |
| fluentd-vlft4                                     | 1/1   | Running | 1 (55m ago)  | 92m | 10.233.125.14  | worker2 | <none>         | <none>          |
| kibana-79d4c987b9-lzbrh                           | 1/1   | Running | 1 (61m ago)  | 81m | 10.233.125.15  | worker2 | <none>         | <none>          |

Hình 2.21. Trạng thái của các pods.

**Enterprise Search**  
Search everything →  
Build a powerful search experience.  
Connect your users to relevant data.  
Unify your team content.

**Observability**  
Centralize & monitor →  
Monitor infrastructure metrics.  
Trace application requests.  
Measure SLAs and react to issues.

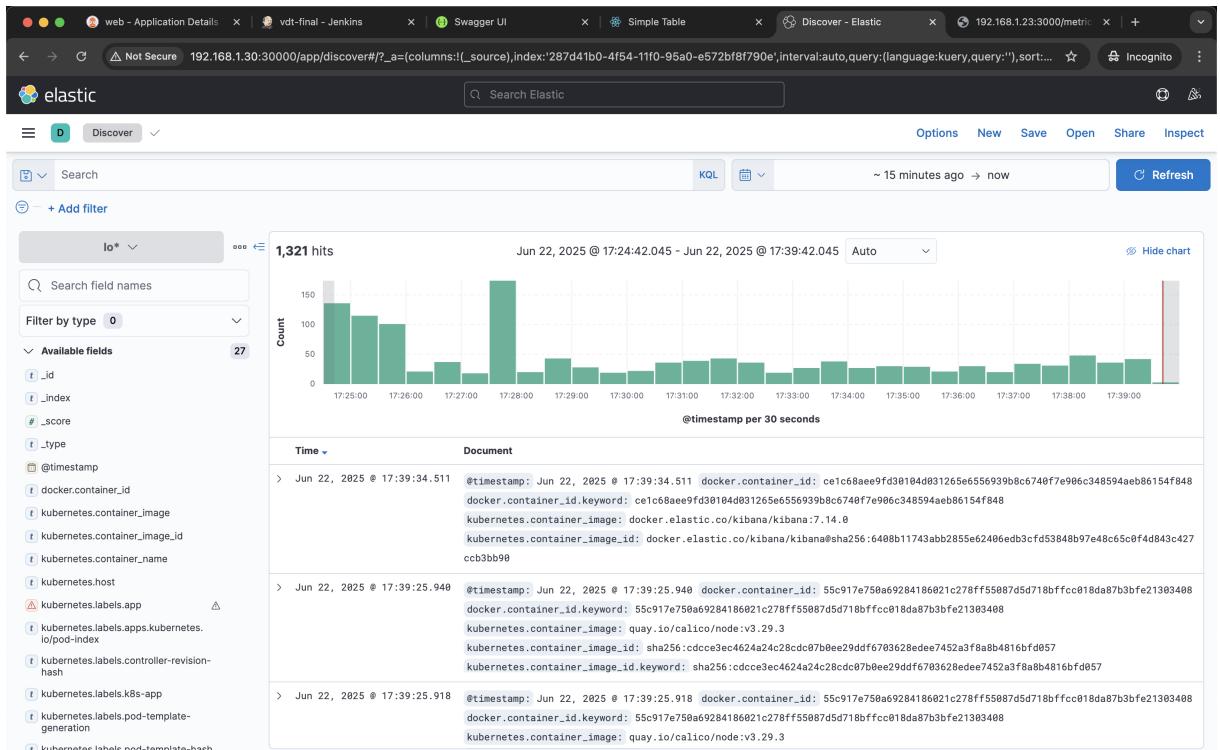
**Kibana**  
Visualize & analyze →  
Analyze data in dashboards.  
Search and find insights.  
Design pixel-perfect presentations.  
Plot geographic data.  
Model, predict, and detect.

**Security**  
SIEM & Endpoint Security →  
Prevent threats autonomously.  
Detect and respond.  
Investigate incidents.

**Ingest your data**

- Add data
- Add Elastic Agent integrations
- Upload a file

Hình 2.22. Giao diện kibana.



Hình 2.23. Log từ kibana.

Thiếu sót: Chưa cấu hình được fluentd lấy log từ service.

## 2.5 Security

### 2.5.1 Yêu cầu 1 (Id)

- Dựng HAProxy Loadbalancer trên 1 VM riêng (trong trường hợp cụm lab riêng của sinh viên) với mode TCP, mở port trên LB trả về NodePort của App trên K8S Cluster. (0.5)
- Sử dụng giải pháp Ingress cho các deployment, đảm bảo các truy cập đến các port App sử dụng https (0.5)
- Cho phép sinh viên sử dụng self-signed cert để làm bài

### 2.5.2 Output 1

#### Ansible playbooks để triển khai HA proxy

```
---
- name: Update apt cache
  ansible.builtin.apt:
    update_cache: yes
  become_user: root

- name: Installs haproxy load balancer
  ansible.builtin.apt:
    name:
      - haproxy
    state: present

- name: Pushes configuration
  template:
    src: templates/haproxy.cfg.j2
    dest: /etc/haproxy/haproxy.cfg
    mode: 0640
    owner: root
    group: root
  notify:
    - restart haproxy

- name: Sets default starting flag to 1
  lineinfile:
    dest: /etc/default/haproxy
    regexp: "^\$ENABLED"
    line: "\$ENABLED=1"
  notify:
    - restart haproxy
```

## Templates haproxy

```
global
    daemon
    maxconn 256

defaults
    mode http
    timeout connect 5000ms
    timeout client 50000ms
    timeout server 50000ms

listen cluster
    bind {{ ansible_host }}:80
    mode http
    stats enable
    balance roundrobin
{% for backend in groups['web'] %}
    server {{ hostvars[backend]['ansible_hostname'] }} {{ hostvars[backend]['ansible_facts']['default_ipv4']['address'] }}:3000 check port
        3000
{% endfor %}
    option httpchk HEAD / HTTP/1.0
```

```
1 [web]
2 vm1 ansible_host=192.168.1.12 ansible_user=vm1
3 vm2 ansible_host=192.168.1.13 ansible_user=vm2
4
5 [haproxy]
6 vm3 ansible_host=192.168.1.7 ansible_user=vm3
```

Hình 2.24. Cài đặt hosts.

\*\* Lưu ý \*\*: Thay đổi host, ip và port.

### 2.5.3 Yêu cầu 2 (Id)

Đảm bảo 1 số URL của api service khi truy cập phải có xác thực thông qua 1 trong số các phương thức cookie, basic auth, token auth, nếu không sẽ trả về HTTP response code 403. (0.5)

Thực hiện phân quyền cho 2 loại người dùng trên API:

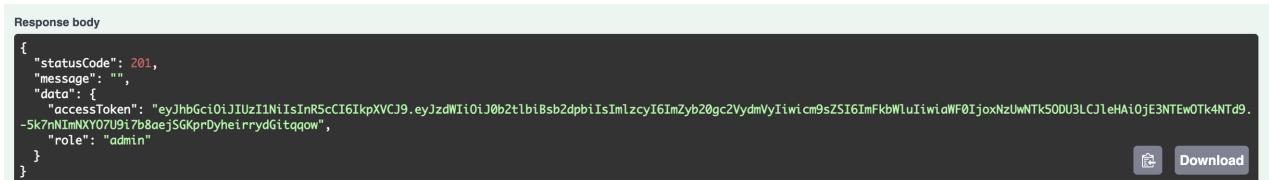
- Nếu người dùng có role là user thì truy cập vào GET request trả về code 200, còn truy cập vào POST/DELETE thì trả về 403
- Nếu người dùng có role là admin thì truy cập vào GET request trả về code 200, còn truy cập vào POST/DELETE thì trả về 2xx

#### 2.5.4 Output

Sử dụng LocalAuthGuard để xác thực đăng nhập và JWT Guard để bảo vệ các route, đồng thời triển khai custom decorator nhằm phân quyền người dùng trong NestJS. Để chi tiết vui lòng xem [source code](#)

Để thuận tiện cho việc demo, chức năng đăng nhập chỉ cần thực hiện xác thực cơ bản với username và password cố định, không cần lưu trữ trong cơ sở dữ liệu.

- Nếu username và password là **admin**:

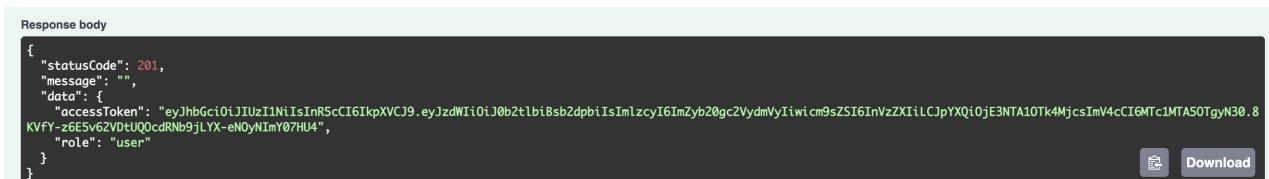


```
Response body
{
  "statusCode": 201,
  "message": "",
  "data": {
    "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJB2t1biBsb2dpbiIsImLzcyI6ImZyb20gc2VydmlvIiwicm9sZSI6ImFkbWluIiwiWF0IjoxNzUwNTk5ODU3LCJleHaiOjE3NTExOTk4NTd9.-5krhInNXY07U9t7b8aejSGKprDyheirrydGltqqow",
    "role": "admin"
  }
}
```

Copy Download

Hình 2.25. Kết quả api trả về khi đăng nhập với admin.

- Nếu username và password là **user**:

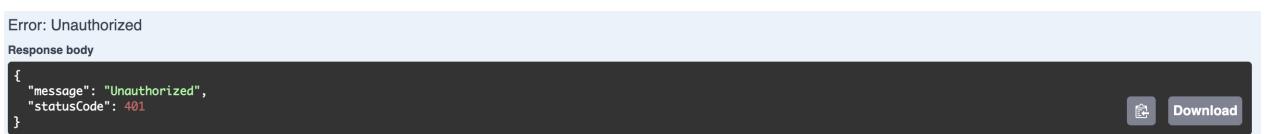


```
Response body
{
  "statusCode": 201,
  "message": "",
  "data": {
    "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJB2t1biBsb2dpbiIsImLzcyI6ImZyb20gc2VydmlvIiwicm9sZSI6InVZXIiLCJpYXQiOjE3NTA1OTk4MjcsImV4cCI6MTC1MTA5OTgyN30.8KVFY-zG65vG2VdtUQ0cdRN9jLYX-eNoYNImY07HU4",
    "role": "user"
  }
}
```

Copy Download

Hình 2.26. Kết quả api trả về khi đăng nhập với user.

- Khi không đăng nhập nhưng gọi API



```
Error: Unauthorized
Response body
{
  "message": "Unauthorized",
  "statusCode": 401
}
```

Copy Download

Hình 2.27. Kết quả khi không đăng nhập nhưng gọi API.

- Khi đăng nhập nhưng sai role

Error: Forbidden  
Response body

```
{  
  "message": "Access denied for role: admin",  
  "error": "Forbidden",  
  "statusCode": 403  
}
```

Download

Hình 2.28. Kết quả gọi API không dành cho admin.

Error: Forbidden  
Response body

```
{  
  "message": "Access denied for role: user",  
  "error": "Forbidden",  
  "statusCode": 403  
}
```

Download

Hình 2.29. Kết quả gọi API không dành cho user.

### 2.5.5 Yêu cầu 3 (Id)

Sử dụng 1 trong số các giải pháp để ratelimit cho Endpoint của api Service, sao cho nếu có quá 10 request trong 1 phút gửi đến Endpoint của api service thì các request sau đó bị trả về HTTP Response 409

### 2.5.6 Output 3

Sử dụng Rate Limit của NestJS là ThrottlerModule.

Đây là script viết bằng Python để tự động gọi liên tục 15 request GET vào API endpoint. Vì chỉ giới hạn 10 request trong một phút nên các request từ 11-15 sẽ bị reject

```
import requests

api_key = <ACCESS_TOKEN>
url = 'http://localhost:3000/api/students'
headers = {'Authorization': f'Bearer {api_key}'}

for i in range(15):
    response = requests.get(url, headers=headers)
    if response.text:
        print(f'Request {i+1}: Status Code: {response.status_code},  
          Response: {response.text}')
    else:
        print(f'Request {i+1}: Status Code: {response.status_code},  
          Response: {"You have exceeded 10 requests per minute"}')
```

```

": "Đại học Bách Khoa Hà Nội (HUST)"}, {"id": "S012", "name": "Nguyễn Sỹ Tân", "birth_date": "07/07/2004", "university": "Đại học Công nghệ (UET)"}, {"id": "S013", "name": "Mai Xuân Duy Quang", "birth_date": "04/07/2003", "university": "Đại học Bách Khoa Hà Nội (HUST)"}, {"id": "S014", "name": "Lê Tân Phát", "birth_date": "15/12/2004", "university": "ĐH Mở Tp.HCM"}, {"id": "S015", "name": "Nguyễn Quang Ninh", "birth_date": "24/04/2004", "university": "Đại học Công nghệ (UET)"}, {"id": "S016", "name": "Nguyễn Trung Vương", "birth_date": "03/10/2003", "university": "Đại học Bách Khoa - ĐHQG TPHCM (HCMUT)"}, {"id": "S017", "name": "Nguyễn Phước Ngưỡng Long", "birth_date": "18/10/2005", "university": "Đại học Công nghệ (UET)"}, {"id": "S018", "name": "Nguyễn Văn Dương", "birth_date": "30/10/2003", "university": "Đại học Công nghệ (UET)"}, {"id": "S019", "name": "Lê Minh Hoàng", "birth_date": "17/05/2004", "university": "Đại học Khoa học tự nhiên - ĐHQG TPHCM (HCMUS)"}, {"id": "S020", "name": "Nguyễn Đức Thịnh", "birth_date": "10/09/2001", "university": "Đại học Thủy Lợi"}, {"id": "S021", "name": "Hoàng Minh Thành", "birth_date": "09/06/1999", "university": "Đại học tổng hợp ITMO"}, {"id": "S022", "name": "Vũ Đình Ngọc Bảo", "birth_date": "29/01/2005", "university": "Đại học Khoa học tự nhiên - ĐHQG TPHCM (HCMUS)"}, {"id": "S023", "name": "Nguyễn Hồng Linh", "birth_date": "08/12/2003", "university": "Đại học Công nghệ (UET)"}]
Request 10: Status Code: 200, Response: {"statusCode":200,"message": "", "data": [{"id": "S001", "name": "Nguyễn Đăng Quân", "birth_date": "11/05/2004", "university": "Đại học Công nghệ (UET)"}, {"id": "S002", "name": "Trịnh Vinh Tuấn Đạt", "birth_date": "05/10/2003", "university": "Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc"}, {"id": "S003", "name": "Ngô Xuân Hòa", "birth_date": "27/07/2004", "university": "Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc"}, {"id": "S004", "name": "Bùi Đức Hùng", "birth_date": "31/07/2004", "university": "Đại học Bách Khoa Hà Nội (HUST)"}, {"id": "S005", "name": "Nguyễn Tuấn Anh", "birth_date": "25/01/2003", "university": "Đại học Bách Khoa Hà Nội (HUST)"}, {"id": "S006", "name": "Lương Nhật Hào", "birth_date": "09/07/2003", "university": "Đại học Công nghệ (UET)"}, {"id": "S007", "name": "Nguyễn Đức Anh", "birth_date": "23/01/2003", "university": "Học viện Công nghệ Bưu chính Viễn thông - Cơ sở phía Bắc"}, {"id": "S008", "name": "Đinh Trường Lâm", "birth_date": "23/02/2001", "university": "Đại học tổng hợp ITMO"}, {"id": "S009", "name": "Nguyễn Đăng Bả o Lâm", "birth_date": "17/08/2004", "university": "Đại học Bách Khoa Hà Nội (HUST)"}, {"id": "S010", "name": "Phạm Ngọc Hải Dương", "birth_date": "05/01/2004", "university": "Đại học Công nghệ (UET)"}, {"id": "S011", "name": "Nguyễn Minh Quân", "birth_date": "07/07/2004", "university": "Đại học Công nghệ (UET)"}, {"id": "S012", "name": "Nguyễn Sỹ Tân", "birth_date": "07/07/2004", "university": "Đại học Công nghệ (UET)"}, {"id": "S013", "name": "Mai Xuân Duy Quang", "birth_date": "04/07/2003", "university": "Đại học Bách Khoa Hà Nội (HUST)"}, {"id": "S014", "name": "Lê Tân Phát", "birth_date": "15/12/2004", "university": "ĐH Mở Tp.HCM"}, {"id": "S015", "name": "Nguyễn Quang Ninh", "birth_date": "24/04/2004", "university": "Đại học Công nghệ (UET)"}, {"id": "S016", "name": "Nguyễn Trung Vương", "birth_date": "03/10/2003", "university": "Đại học Bách Khoa - ĐHQG TPHCM (HCMUT)"}, {"id": "S017", "name": "Nguyễn Phước Ngưỡng Long", "birth_date": "18/10/2005", "university": "Đại học Công nghệ (UET)"}, {"id": "S018", "name": "Nguyễn Văn Dương", "birth_date": "30/10/2003", "university": "Đại học Công nghệ (UET)"}, {"id": "S019", "name": "Lê Minh Hoàng", "birth_date": "17/05/2004", "university": "Đại học Khoa học tự nhiên - ĐHQG TPHCM (HCMUS)"}, {"id": "S020", "name": "Nguyễn Đức Thịnh", "birth_date": "10/09/2001", "university": "Đại học Thủy Lợi"}, {"id": "S021", "name": "Hoàng Minh Thành", "birth_date": "09/06/1999", "university": "Đại học tổng hợp ITMO"}, {"id": "S022", "name": "Vũ Đình Ngọc Bảo", "birth_date": "29/01/2005", "university": "Đại học Khoa học tự nhiên - ĐHQG TPHCM (HCMUS)"}, {"id": "S023", "name": "Nguyễn Hồng Linh", "birth_date": "08/12/2003", "university": "Đại học Công nghệ (UET)"}]
Request 11: Status Code: 429, Response: {"statusCode":429,"message": "ThrottlerException: Too Many Requests"}
Request 12: Status Code: 429, Response: {"statusCode":429,"message": "ThrottlerException: Too Many Requests"}
Request 13: Status Code: 429, Response: {"statusCode":429,"message": "ThrottlerException: Too Many Requests"}
Request 14: Status Code: 429, Response: {"statusCode":429,"message": "ThrottlerException: Too Many Requests"}
Request 15: Status Code: 429, Response: {"statusCode":429,"message": "ThrottlerException: Too Many Requests"}

```

Hình 2.30. Kết quả kiểm tra.

Em nghĩ khi gửi quá nhiều request trong một khoảng thời gian ngắn, lỗi phù hợp là 429 Too Many Requests thay vì 409 Conflict, vì đây là tình huống liên quan đến giới hạn tần suất truy cập chứ không phải xung đột logic dữ liệu.