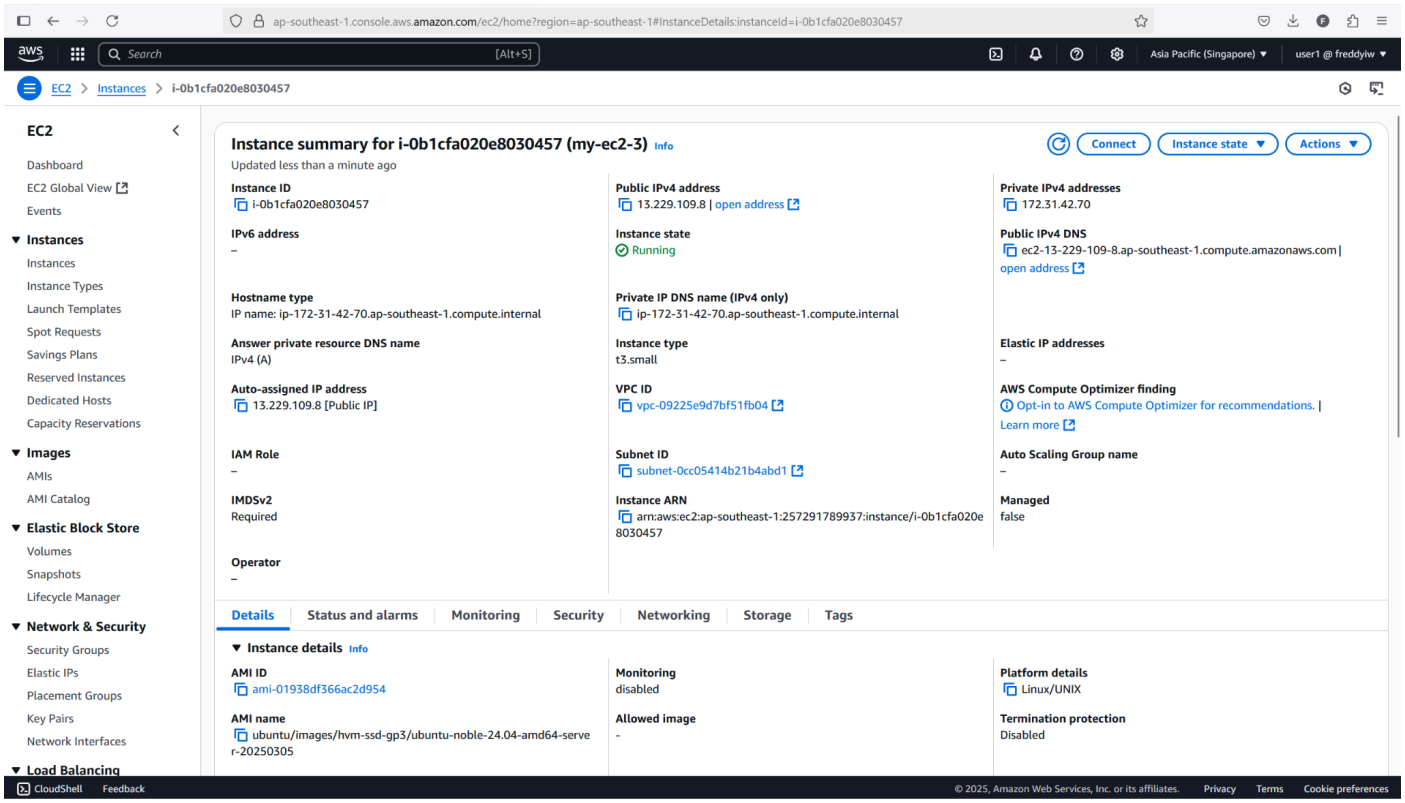
DevOps Take-home Assignment Documentation
Freddy Indra Wiryadi

Steps:
1) Create Dockerfile to dockerize the app

Dockerfile:
```
FROM openjdk:19-jdk-alpine
EXPOSE 8080
COPY build/libs/spring-petclinic-3.1.0.jar spring-petclinic-3.1.0-SNAPSHOT.jar
ENTRYPOINT ["java","-jar","/spring-petclinic-3.1.0-SNAPSHOT.jar"]
```

2) Provision a VM (EC2 instance) on AWS



3) Install k3s on the VM

```
curl -sfL https://get.k3s.io | INSTALL_K3S_EXEC="--tls-san 13.229.109.8" sh -
```

Modify /etc/rancher/k3s/k3s.yaml, change the cluster server value to the public IP address of the VM (https://13.229.109.8:6443) so we can execute kubectl remotely.

4) Create kubernetes manifests

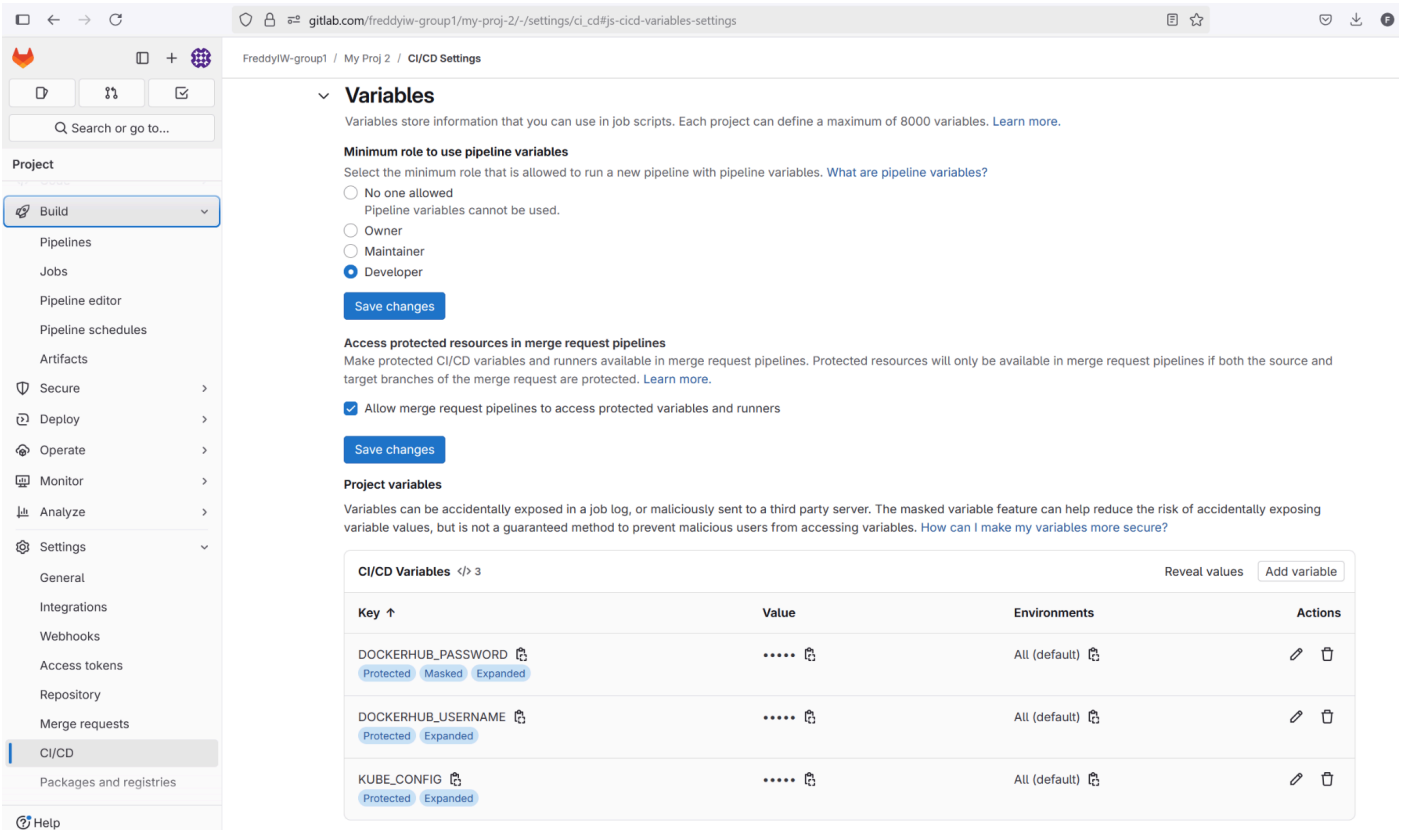
petclinic-deployment.yaml:
```
apiVersion: apps/v1
kind: Deployment
```

```
metadata:
  name: petclinic
  namespace: petclinic
  labels:
    app: petclinic
spec:
  replicas: 1
  selector:
    matchLabels:
      app: petclinic
  template:
    metadata:
      labels:
        app: petclinic
    spec:
      containers:
      - name: petclinic
        image: freddyiw/my-pet-clinic-3:latest
        ports:
        - containerPort: 8080
```

petclinic-service.yaml:

```
apiVersion: v1
kind: Service
metadata:
  name: petclinic-service
  namespace: petclinic
spec:
  type: NodePort
  selector:
    app: petclinic
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 8080
      nodePort: 30080
```

5) Set up CI/CD variables on GitLab
DOCKERHUB_USERNAME
DOCKERHUB_PASSWORD
KUBE_CONFIG (*contain the base64-encoded kubernetes config file content from /etc/rancher/k3s/k3s.yaml (equivalent to ~/.kube/config in k8s)*)



6) Set up CI/CD pipeline on GitLab

.gitlab-ci.yml:

```
stages:
  - build
  - deploy

variables:
  DOCKER_IMAGE: freddyiw/my-pet-clinic-3
  DOCKER_TAG: latest

build_job_1:
  stage: build
  image: gradle:7.6-jdk17
  variables:
    DOCKER_HOST: tcp://docker:2375
    DOCKER_TLS_CERTDIR: ""
    TESTCONTAINERS_RYUK_DISABLED: true
  services:
    - docker:dind
  script:
    - ./gradlew build -Dtestcontainers.use.docker.host=true
  artifacts:
    paths:
      - build/libs/spring-petclinic-3.1.0.jar
    expire_in: 1 hour
```

```
build_job_2:
  stage: build
  image: docker:latest
  services:
    - docker:dind
  variables:
    DOCKER_HOST: tcp://docker:2375
    DOCKER_TLS_CERTDIR: ""
  script:
    - docker login -u "$DOCKERHUB_USERNAME" -p "$DOCKERHUB_PASSWORD"
    - docker build -t $DOCKER_IMAGE:$DOCKER_TAG .
    - docker push $DOCKER_IMAGE:$DOCKER_TAG
  needs:
    - job: build_job_1
      artifacts: true

deploy:
  stage: deploy
  image: bitnami/kubectl:latest
  before_script:
    - echo "$KUBE_CONFIG" | base64 -d > kubeconfig
    - export KUBECONFIG=$CI_PROJECT_DIR/kubeconfig
  script:
    - kubectl apply -f petclinic-deployment.yaml
    - kubectl rollout restart deployment petclinic
  only:
    - main
```

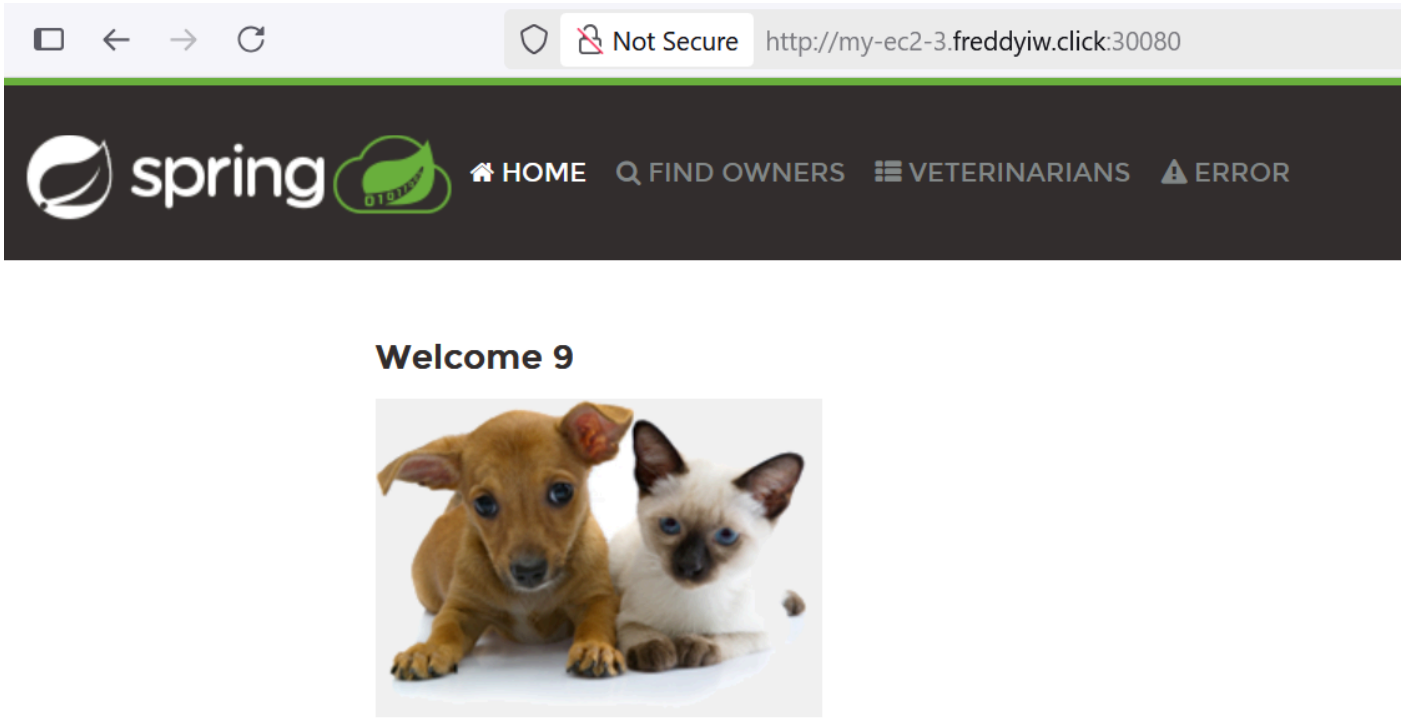
7) The CI/CD pipeline will run when:

- There is a code change on the main branch (e.g. a pull request is merged to the main branch)

  or

- Triggered manually from the GitLab UI

8) After GitLab CI/CD deploys the app to the kubernetes cluster on EC2 VM, access the app on http://my-ec2-3.freddyiw.click:30080/



9) For monitoring, install Prometheus and Grafana on the k3s cluster using Helm

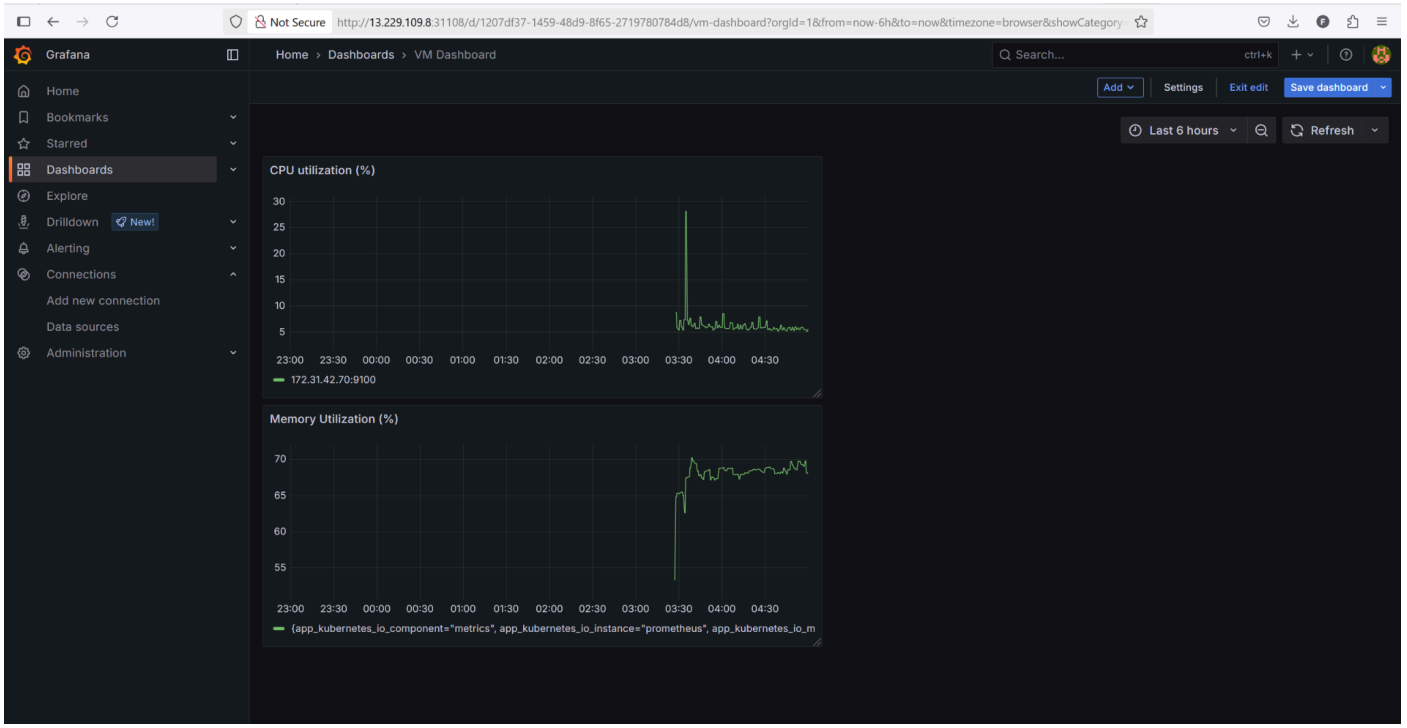
```
helm repo add prometheus-community
https://prometheus-community.github.io/helm-charts
helm repo update
kubectl create namespace monitoring
helm install prometheus-stack prometheus-community/kube-prometheus-stack
--namespace monitoring
```

Edit Grafana service to use NodePort:
`kubectl -n monitoring edit svc prometheus-stack-grafana`
Change `type: ClusterIP` to `type: NodePort`

10) Set up Grafana to use Prometheus as the data source, then create a dashboard to show:

CPU utilization:
```
100 - (avg by(instance) (irate(node_cpu_seconds_total{mode="idle"}[5m])) * 100)
```

Memory utilization:
```
100 * (1 - (node_memory_MemAvailable_bytes / node_memory_MemTotal_bytes))
```



11) For Logging, install Loki and Promtail
(Promtail is installed as DaemonSet to scrape container logs and send to Loki for storing)

```
helm repo add grafana https://grafana.github.io/helm-charts
helm repo update
kubectl create namespace logging
helm install loki grafana/loki-stack \
  --namespace logging \
  --set loki.enabled=true \
  --set promtail.enabled=true \
  --set grafana.enabled=false \
  --set fluent-bit.enabled=false
```

12) Add Loki as a data source in Grafana
URL: http://loki.logging.svc.cluster.local:3100

Go to Explore tab in Grafana, select Loki as the data source, then use this query:

```
{app="petclinic"}
```

Source code public repository: https://github.com/FreddyIW/my-proj-2