Nama: Rayhand Fernanda Viantama

Repo : https://github.com/Rayhand117/swarm-to-k8s

**UPDATE, OVERDUE: Mulai Halaman 20**

**Tahap 0 pengecekan:**

**Sebelum convert dari docker swarm ke kubernetes, dilakukan pengecekan menggunakan developmen env (WSL2 KaliLinux & Docker Desktop):**

*git clone https://github.com/docker-archive/swarm-microservice-demo-v1.git*

git push to my repo:

https://github.com/Rayhand117/swarm-to-k8s

git clone https://github.com/Rayhand117/swarm-to-k8s.git

cd swarm-to-k8s

*docker compose up -d*

+ terdapat error pada Dockerfile, base image service vote-worker.

- java:7, (base image unavailable di dockerhub) update ke openjdk:8-jdk.

*docker compose up -d*

*docker ps*

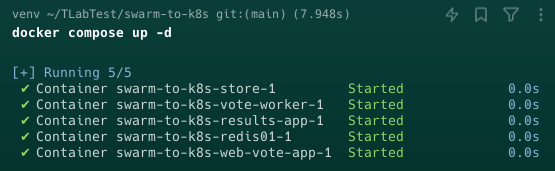
+ service results-app status "exited".

- base image Dockerfile node:0.10 terlalu tua, error: tidak bisa menggunakan variabel "let".

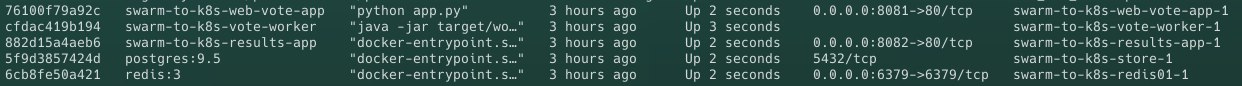
- update base image ke node:14.

*docker compose up -d*

+ semua service Ready, services berjalan lancar.



Gambar 1. 1 docker compose up -d

**

Gambar 1. 2 docker ps

**Memberi tag & push ke dockerhub**

*docker tag eb35bc8f57f0 ray117/swarm-microservice-demo-v1-results-app:latest*

*docker tag e2ba538e8f10 ray117/swarm-microservice-demo-v1-vote-worker:latest*

*docker tag 9a17b3e4787f ray117/swarm-microservice-demo-v1-web-vote-app:latest*

*docker tag 6d176851b77f ray117/postgres:9.5*

*docker tag 87856cc39862 ray117/redis:3*

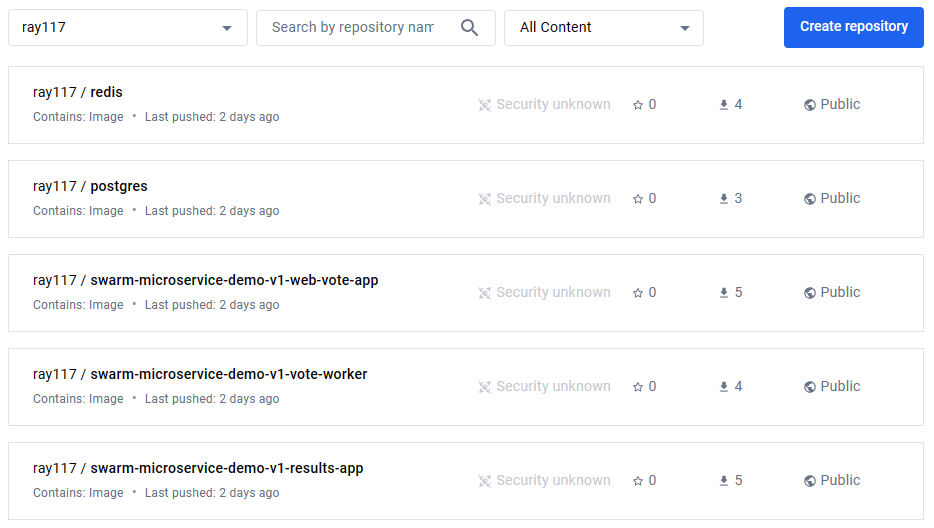
*docker push ray117/swarm-microservice-demo-v1-results-app:latest*

*docker push ray117/swarm-microservice-demo-v1-vote-worker:latest*

*docker push ray117/swarm-microservice-demo-v1-web-vote-app:latest*

*docker push ray117/postgres:9.5*

*docker push ray117/redis:3*



**Melakukan konversi docker-compose.yml dan docker-compose.override.yml menjadi deployment.yml**

- memberi tambahan NodePort pada service web-vote-app, results-app, & redis01 agar server juga bisa diakses oleh Host.

- menggunakan image yang telah dipush

- setiap deployment diberi cpu 0.5 & memory 250Mi

**Tahap 1 Menyiapkan 3 unit server menggunakan VM VirtualBox.:**

+ Menggunakan Distro Ubuntu 22.04.4 LTS untuk ketiga unit server, instalasi menggunakan ISO file.

- Konfigurasi Jaringan: Bridge Adapter, alasan: bisa diakses Host, sesama distro VM, & akses internet.

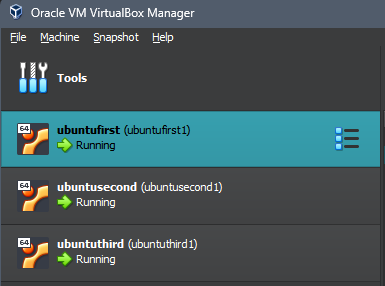
- Spesifikasi:

- ubuntuthird (Kontrol node):

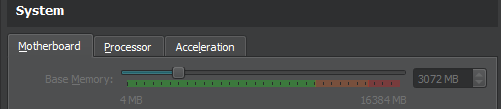
- 4 Cores & 3GB RAM

- ubuntusecond (worker 1), & ubuntufirst(worker 2):

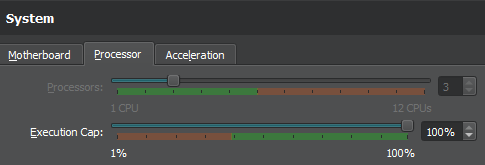
- 3 Cores & 3GB RAM



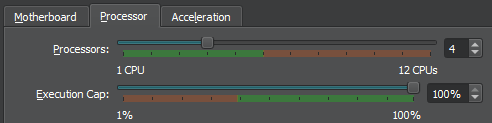
Gambar 1. 3 Ketiga Running Server



Gambar 1. 4 Set Jumlah RAM 3GB (Worker Nodes – ubuntusecond & ubuntufirst)



Gambar 1. 5 Set Jumlah CPU 3 Cores (Worker Nodes ubuntusecond & ubuntufirst)



Gambar 1. 6 Set Jumlah CPU 4 Cores (Master Node - ubuntuthird)

**Tahap 2 Setup Cluster:**

**execute commands pada KETIGA SERVER (persiapan env):**

*sudo apt update*

*sudo apt upgrade*

*sudo apt install git*

*sudo apt install curl*

*sudo snap install code --classic*

*sudo snap install microk8s --classic*

*sudo microk8s status*

*git clone*

## karena tidak bisa copy paste VM meski fitur clipboard bidirectional, jadi banyak git push & git pull:)

**# ubuntuthird (master)**

*sudo microk8s add-node*

**# ubuntusecond (worker1)**

*sudo microk8s join 192.168.1.10:25000/430d6d79606025ca789ea7076c2d6cb5/75268e6c1b30 --worker*

**# ubuntuthird (master)**

*sudo microk8s add-node*

**# ubuntufirst (worker2)**

*sudo microk8s join 192.168.1.10:25000/5f912ea23847d2e8aa2d1755e5c68a38/75268e6c1b30 --worker*

**# KETIGA SERVER enable forward packets & persistent iptables (IP aman ketika reboot)**

*sudo iptables -P FORWARD ACCEPT*

*sudo apt-get install iptables-persistent*

**# prompt: yes, yes**

**# karena ada masalah limit file watchers, execute:**

*echo fs.inotify.max\_user\_watches=1048576 | sudo tee -a /etc/sysctl.conf*

*sudo sysctl --system*

**# ubuntuthird (master) cek status**

*sudo microk8s kubectl get services*

*sudo microk8s kubectl get services -o wide*

*sudo microk8s kubectl get deployments*

*sudo microk8s kubectl get get nodes*

*sudo microk8s kubectl get get pods*

*sudo microk8s kubectl apply -f deployment.yml*

**# ubuntuthird (master) cek status lagi**

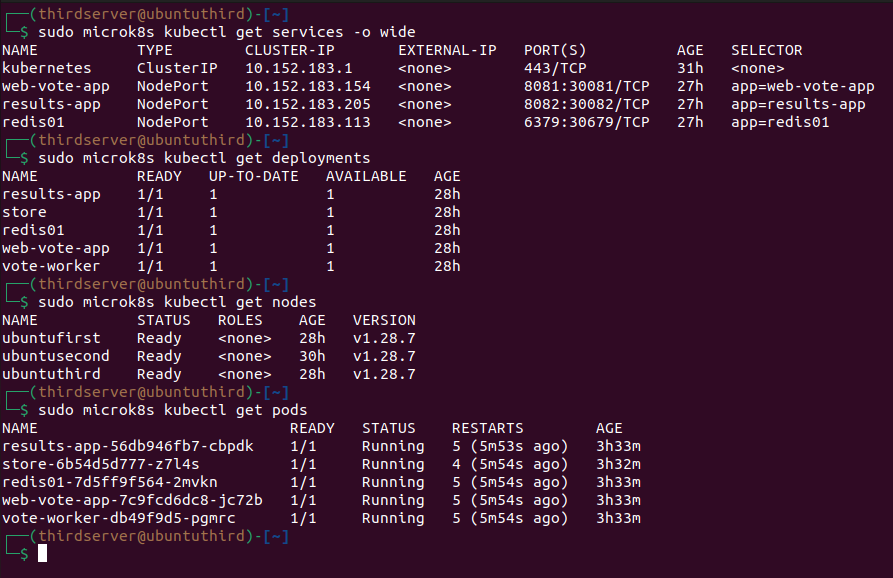
*sudo microk8s kubectl get services*

*sudo microk8s kubectl get services -o wide*

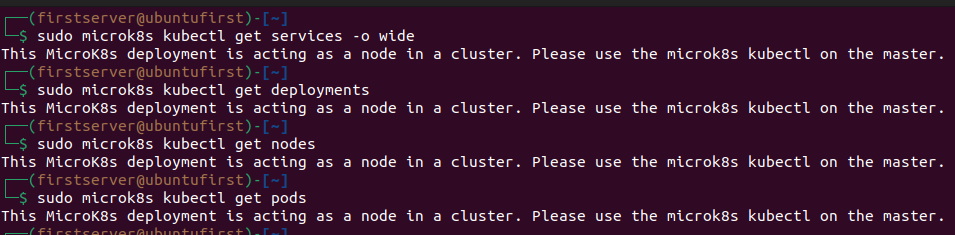
*sudo microk8s kubectl get deployments*

*sudo microk8s kubectl get get nodes*

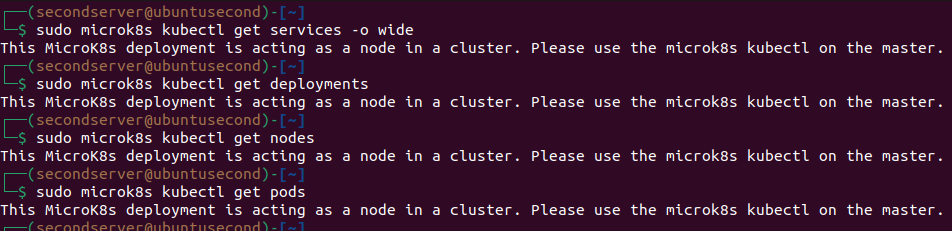
*sudo microk8s kubectl get get pods*



Gambar 1. 7 Status ubuntuthird - master



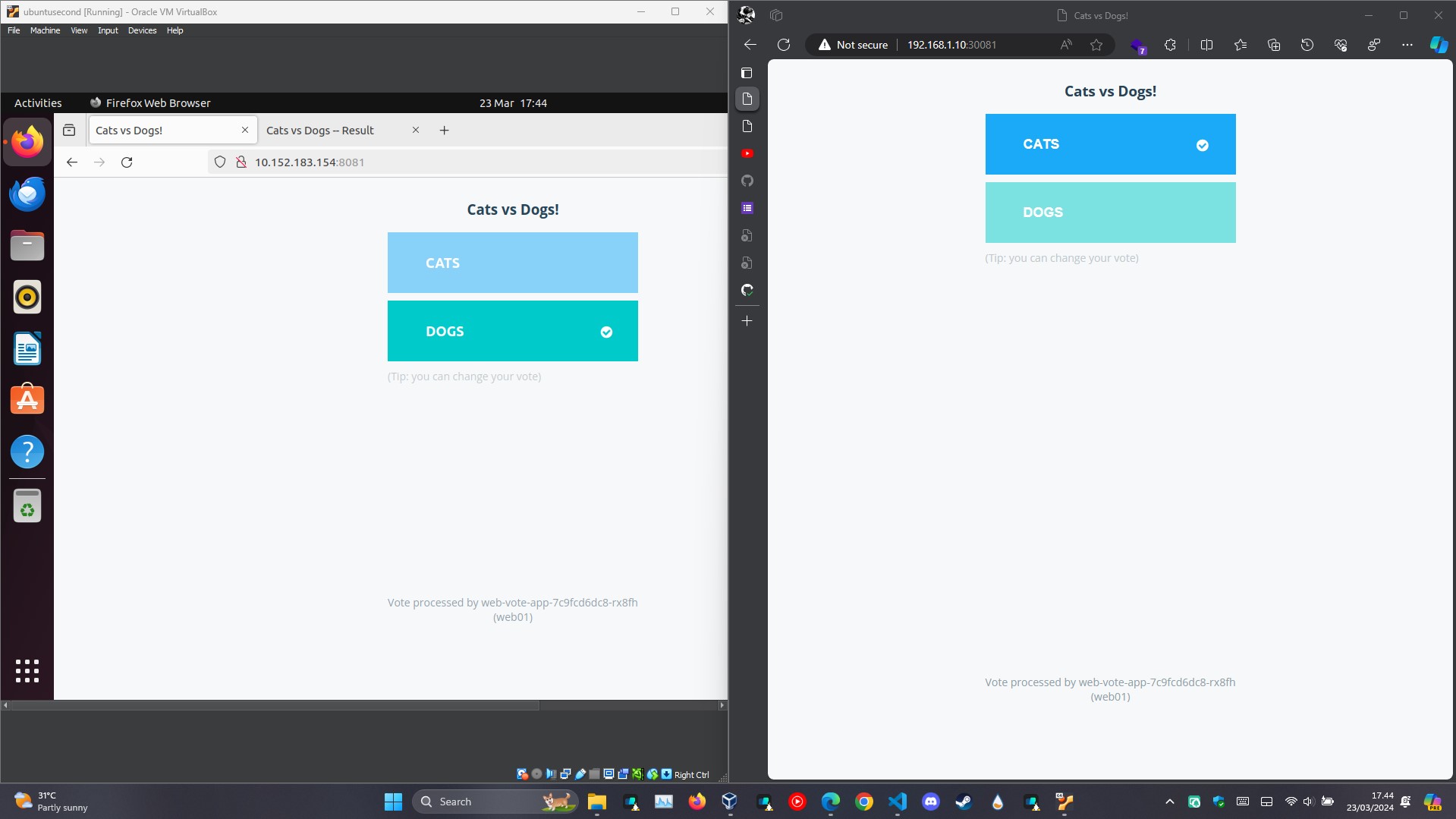
Gambar 1. 8 Status ubuntufirst – worker



Gambar 1. 9 Status ubuntusecond - worker

**Test akses services dalam cluster (node master & worker), 200 OK:**

**web-vote-app <IP>:<Port>10.152.183.154:8081:**



Gambar 1. 10 Service web-vote-app dalam cluster

*curl -I 10.152.183.154:8081*

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

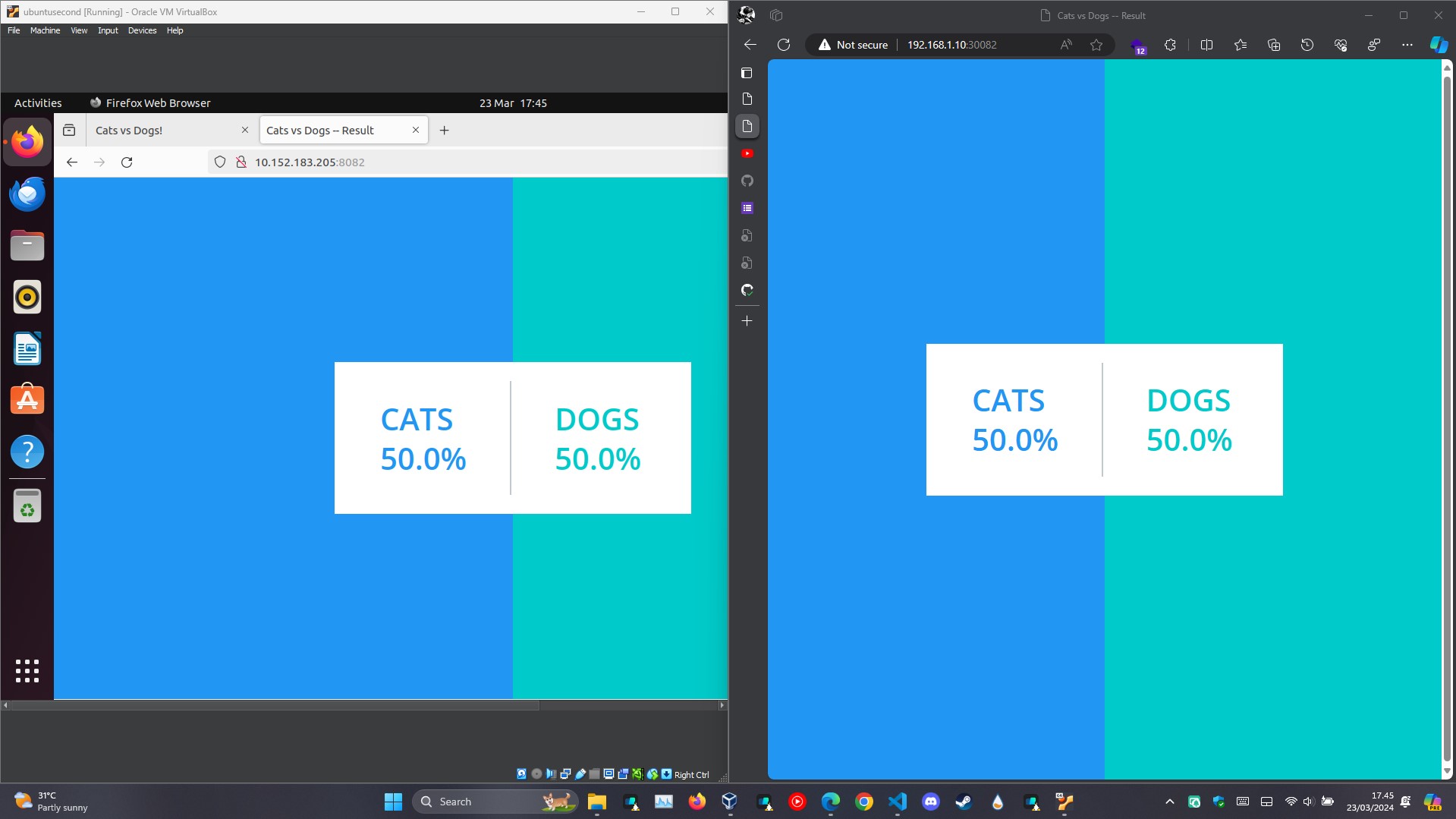
Content-Length: 1293

Set-Cookie: voter\_id=7b30da38b7fe6b25; Path=/

Server: Werkzeug/1.0.1 Python/2.7.18

Date: Sat, 23 Mar 2024 13:55:03 GMT

**results-app <IP>:<Port> 10.152.183.205:8082:**



Gambar 1. 11 Service results-app dalam cluster

*curl -I 10.152.183.205:8082*

HTTP/1.1 200 OK

X-Powered-By: Express

Access-Control-Allow-Origin: \*

Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept

Access-Control-Allow-Methods: PUT, GET, POST, DELETE, OPTIONS

Accept-Ranges: bytes

Cache-Control: public, max-age=0

Last-Modified: Fri, 22 Mar 2024 05:57:53 GMT

ETag: W/"614-18e64bcb6e8"

Content-Type: text/html; charset=UTF-8

Content-Length: 1556

Date: Sat, 23 Mar 2024 13:56:02 GMT

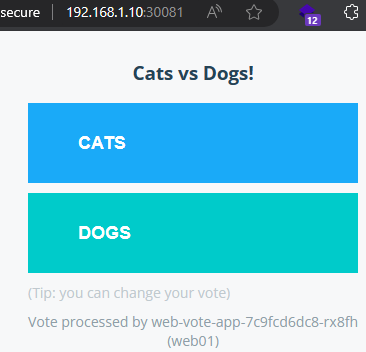
Connection: keep-alive

Keep-Alive: timeout=5

**Test akses services pada Host PC <IP>:<NodePort>, 200 OK:**

**web-vote-app 192.168.1.8 to 192.168.1.10 (all on port 30081)**

**192.168.1.10:30081:**

****

Gambar 1. 12 Service web-vote-app pada Host

*curl -I 192.168.1.10:30081*

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

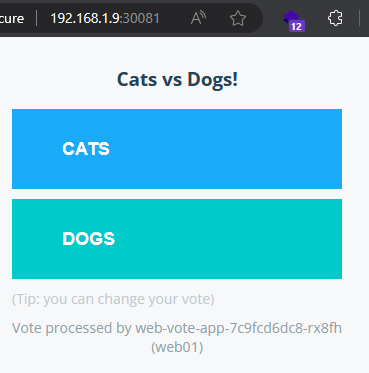
Content-Length: 1293

Set-Cookie: voter\_id=c22fb4829e7428b; Path=/

Server: Werkzeug/1.0.1 Python/2.7.18

Date: Sat, 23 Mar 2024 13:51:05 GMT

**192.168.1.9:30081:**



Gambar 1. 13 Service web-vote-app pada Host

*curl -I 192.168.1.9:30081*

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

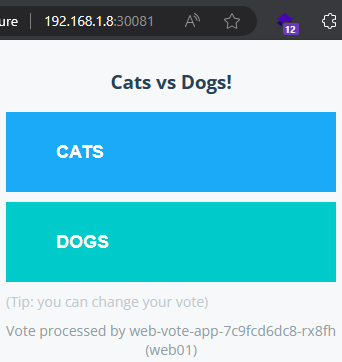
Content-Length: 1293

Set-Cookie: voter\_id=8a96623f9db27d05; Path=/

Server: Werkzeug/1.0.1 Python/2.7.18

Date: Sat, 23 Mar 2024 13:51:10 GMT

**192.168.1.8:30081:**



Gambar 1. 14 Service web-vote-app pada Host

- curl -I 192.168.1.8:30081

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

Content-Length: 1293

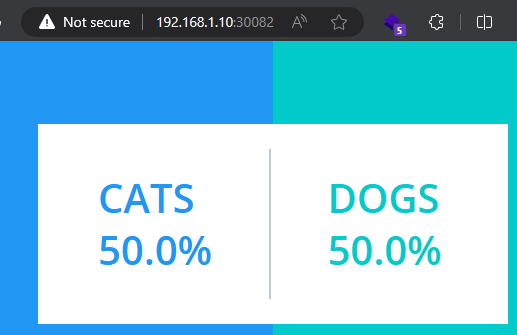
Set-Cookie: voter\_id=a2b58a5179924e54; Path=/

Server: Werkzeug/1.0.1 Python/2.7.18

Date: Sat, 23 Mar 2024 13:51:16 GMT

**results-app 192.168.1.8 to 192.168.1.10 (all on port 30082):**

**192.168.1.10:30082:**

******

Gambar 1. 15 Service results-app pada Host

*curl -I 192.168.1.10:30082*

HTTP/1.1 200 OK

X-Powered-By: Express

Access-Control-Allow-Origin: \*

Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept

Access-Control-Allow-Methods: PUT, GET, POST, DELETE, OPTIONS

Accept-Ranges: bytes

Cache-Control: public, max-age=0

Last-Modified: Fri, 22 Mar 2024 05:57:53 GMT

ETag: W/"614-18e64bcb6e8"

Content-Type: text/html; charset=UTF-8

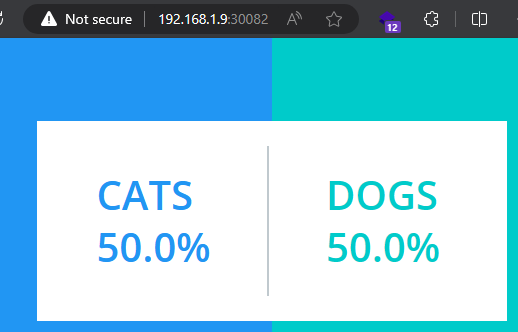
Content-Length: 1556

Date: Sat, 23 Mar 2024 13:52:10 GMT

Connection: keep-alive

Keep-Alive: timeout=5

**192.168.1.9:30082:**

******

Gambar 1. 16 Service results-app pada Host

*curl -I 192.168.1.9:30082*

HTTP/1.1 200 OK

X-Powered-By: Express

Access-Control-Allow-Origin: \*

Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept

Access-Control-Allow-Methods: PUT, GET, POST, DELETE, OPTIONS

Accept-Ranges: bytes

Cache-Control: public, max-age=0

Last-Modified: Fri, 22 Mar 2024 05:57:53 GMT

ETag: W/"614-18e64bcb6e8"

Content-Type: text/html; charset=UTF-8

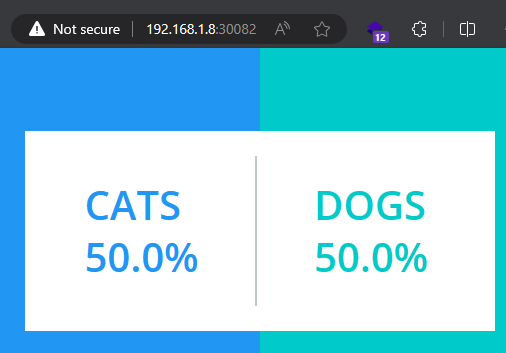
Content-Length: 1556

Date: Sat, 23 Mar 2024 13:52:41 GMT

Connection: keep-alive

Keep-Alive: timeout=5

**192.168.1.8:30082:**

******

Gambar 1. 17 Service results-app pada Host

*curl -I 192.168.1.8:30082*

HTTP/1.1 200 OK

X-Powered-By: Express

Access-Control-Allow-Origin: \*

Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept

Access-Control-Allow-Methods: PUT, GET, POST, DELETE, OPTIONS

Accept-Ranges: bytes

Cache-Control: public, max-age=0

Last-Modified: Fri, 22 Mar 2024 05:57:53 GMT

ETag: W/"614-18e64bcb6e8"

Content-Type: text/html; charset=UTF-8

Content-Length: 1556

Date: Sat, 23 Mar 2024 13:53:00 GMT

Connection: keep-alive

Keep-Alive: timeout=5

**Tahap 3 - Continuous Integrations:**

**Note: Menggunakan second account JustRay117 untuk membuat PR, karena env distro sudah terconfig venv**

Development env: WSL2 AlmaLinux & Docker Desktop

**CI GitHub Actions:**

- Menggunakan Docker Compose, karena hanya butuh 1 node

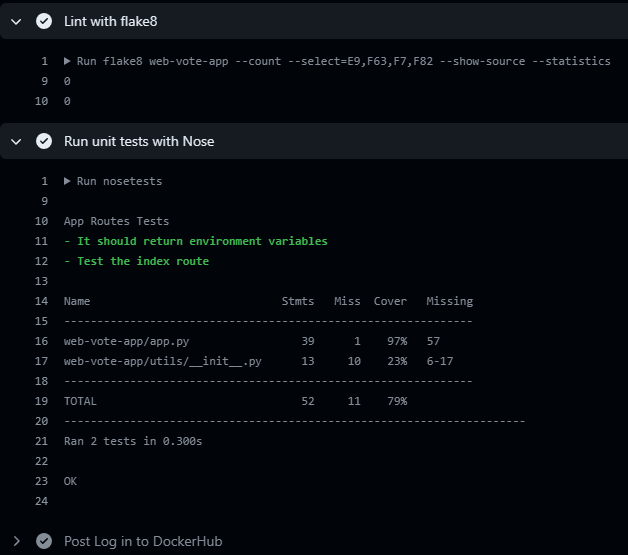
- Set Hostname environment variable agar bisa dijalankan pada GH Actions

- Linting & Test pada app.py menggunakan flake8 & nose

Membuat Unit Tests pada file ./tests/test\_app.py untuk menguji web-vote-app/app.py

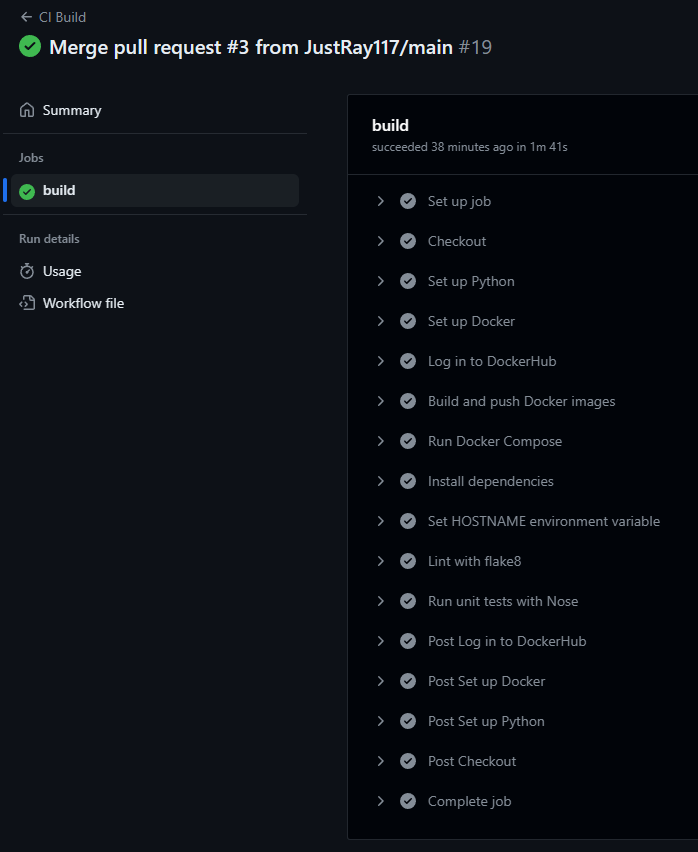
- Membuat test case test\_dump\_env untuk send GET request pada /env untuk mengetahui response status code 200.

- Membuat test case test\_index\_route menggunakan mock, kemudian mengirim GET request & POST request untuk mengetahui response status code 200.



Gambar 1. 18 Flake8 & Nosetests

**Berikut steps - steps yang dilakukan Workflow file ci-build.yaml:**



Gambar 1. 19 Workflow

Step 1. Checkout:

- Memeriksa repository menggunakan actions/checkout@v2 bertujuan untk memberi akses workflow pada konten repository.

Step 2. Set up Python:

- Menggunakan action actions/setup-python@v2 untuk menginstall python versi 3.9.

Step 3. Set up Docker

- Menggunakan action docker/setup-buildx-action@v1 untuk set up Docker Buildx untuk extend docker build.

Step 4. Log in to DockerHub

- Log in ke docker menggunakan action docker/login-action@v1 untuk menggunakan username & passsword yang disimpan pada secrets.

Step 5. Build and push Docker images

- run: docker-compose build

Step 6. Run Docker Compose

- run: docker-compose up -d

Step 7. Install dependencies

- Install python dependencies pada requirements.txt.

Step 8. Set HOSTNAME environment variable

- melakukan set HOSTNAME pada env variable pada runner github.

Step 9. Lint with flake8

- Melakukan linting menggunakan flake8 pada directori web-vote-app

Step 10. Run unit tests with Nose

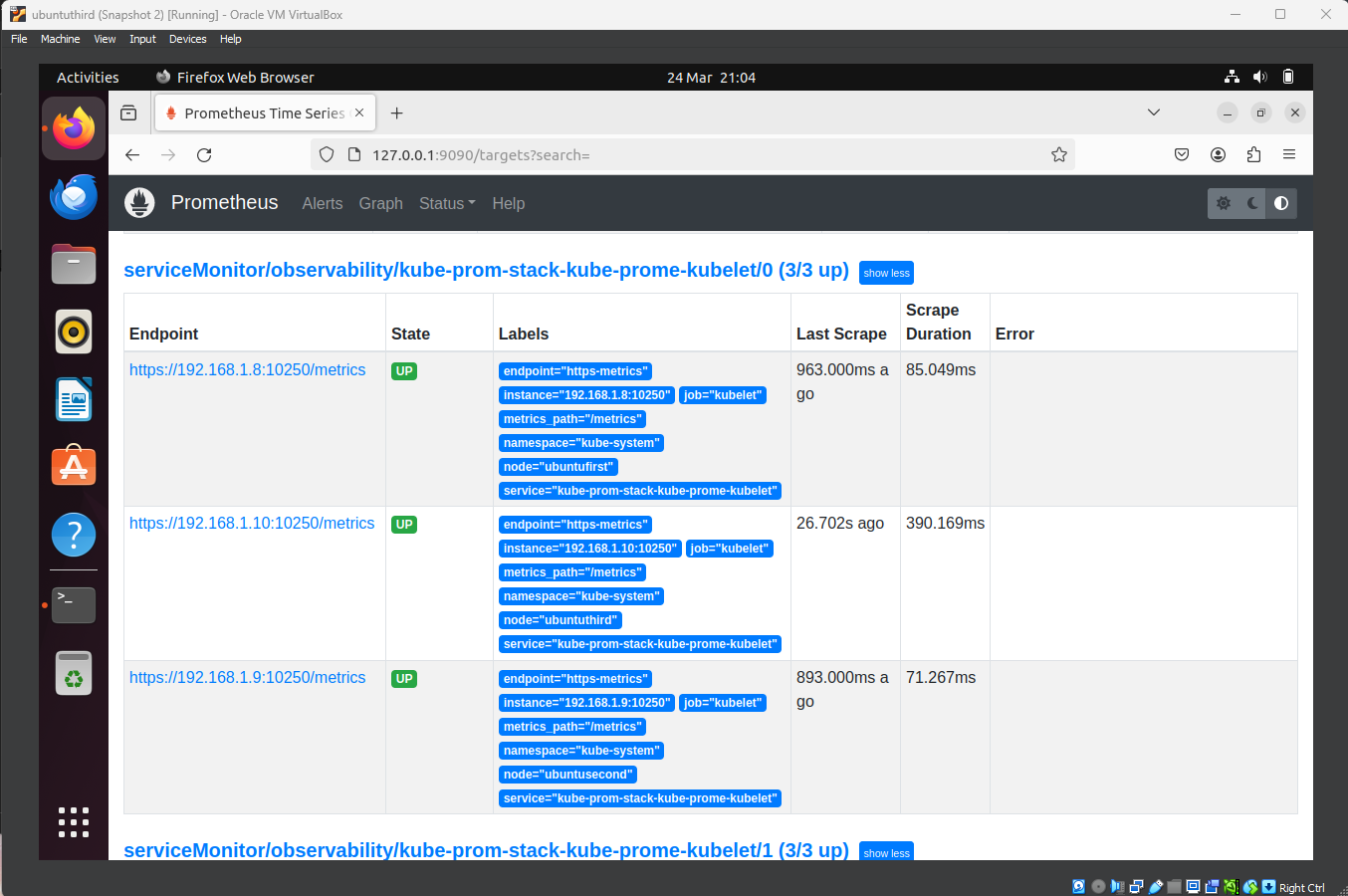
- Menjalankan unit tests menggunakan nose

**Tahap 4 – Prometheus & Grafana:**

**Enabling prometheus**

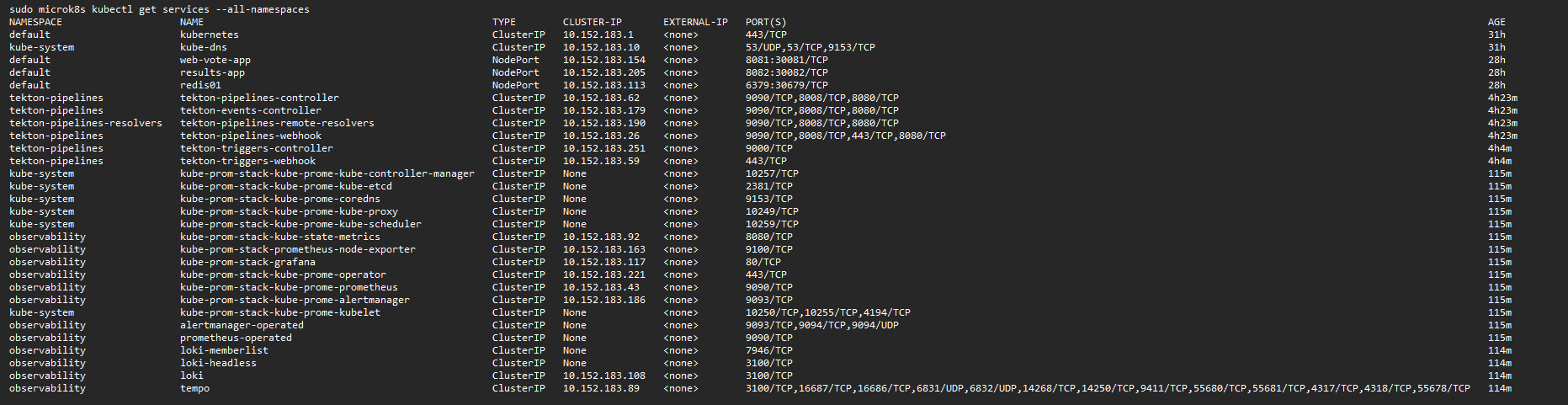
*sudo microk8s enable prometheus*

*sudo microk8s kubectl port-forward service/prometheus-operated 9090:9090 -n observability*

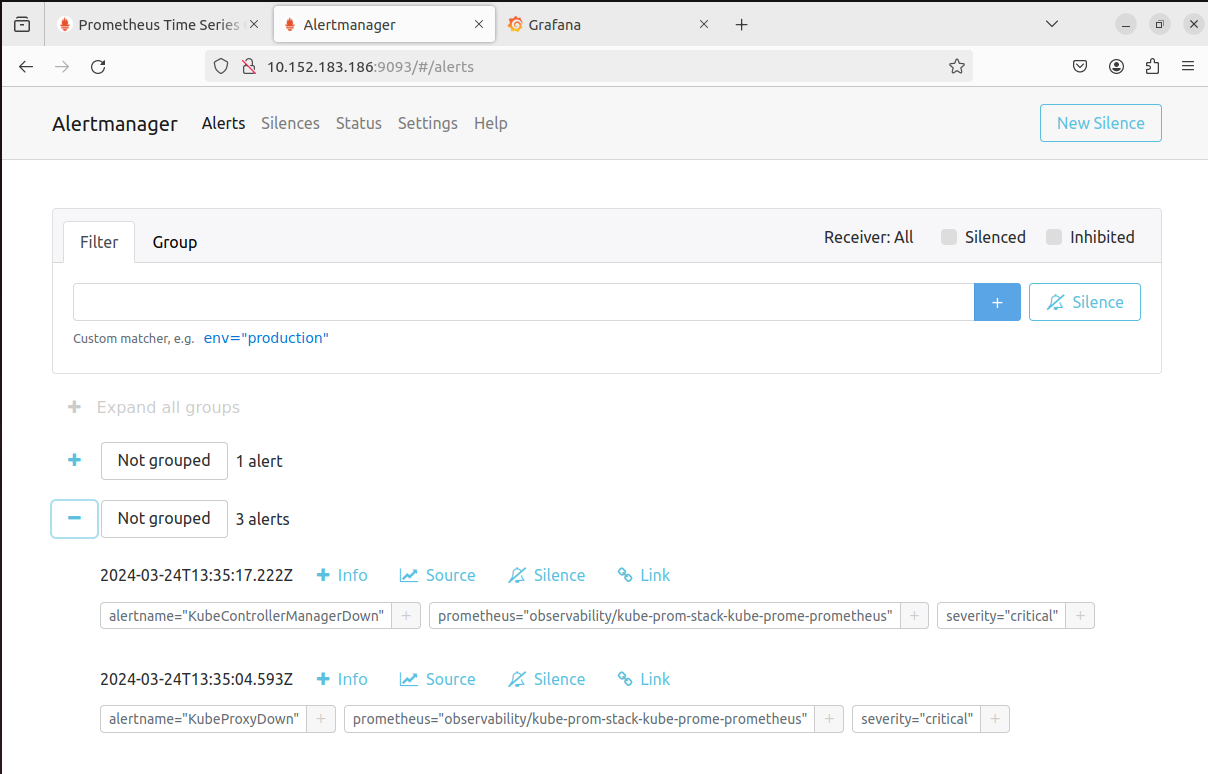


atau bisa langsung diakses menggunakan cluster ip (10.152.183.43:9090)

*sudo microk8s kubectl get services --all-namespaces*



Gambar 1. 20 Services all namespaces



Gambar 1. 21 Prometheus Alertmanager

**Grafana:**

Mencari password:

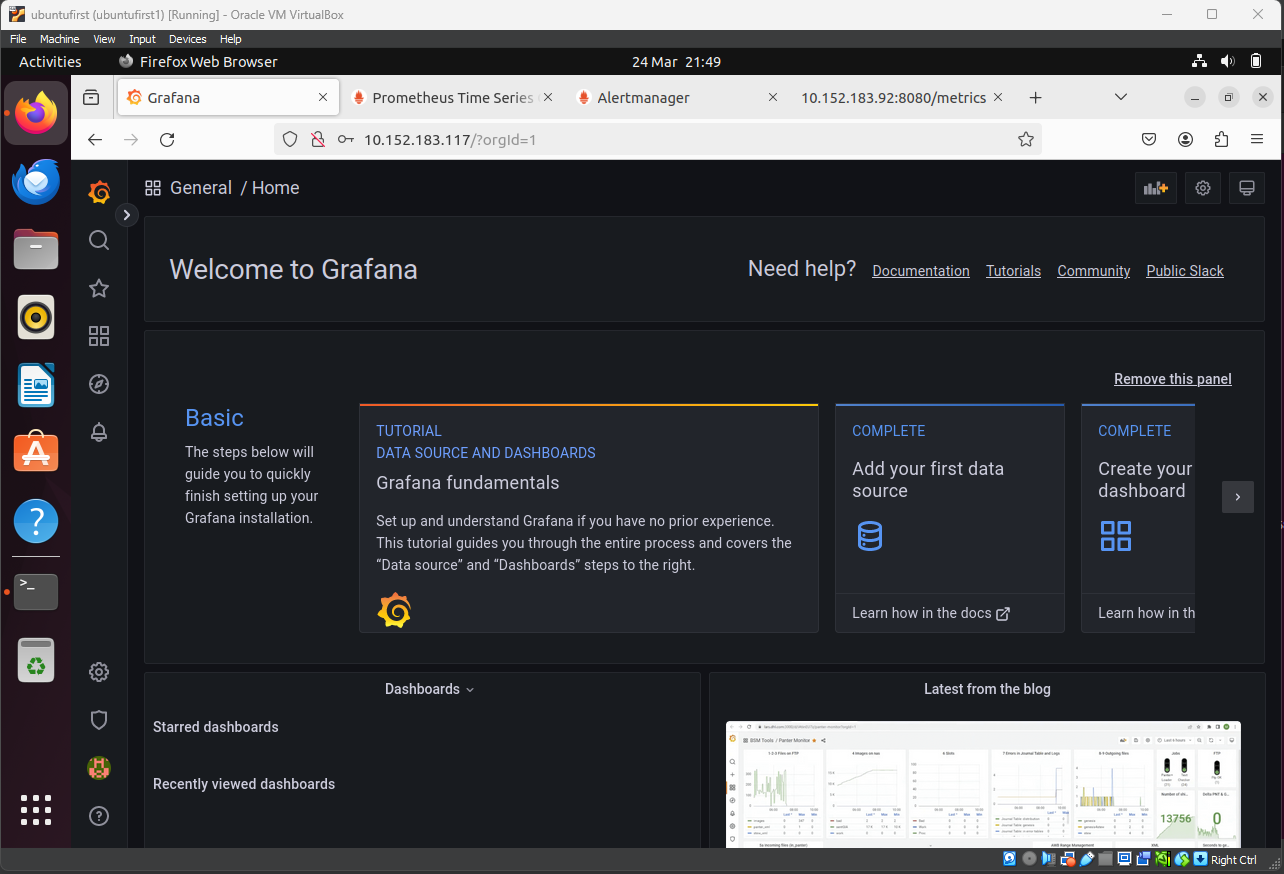
*sudo microk8s kubectl get secrets -n observability*

*sudo microk8s kubectl get secret kube-prom-stack-grafana -n observability -o jsonpath="{.data.admin-password}" | base64 –decode*

username: admin password: prom-operator



Gambar 1. 22 Grafana password



Gambar 1. 23 Grafana Home

Masih belum ter-setup:(

**Tahap 5 – Tekton:**

**Pada direktori tekton, ada pipeline.yaml, pvc.yaml, & tasks.yaml. hanya saja belum ter-apply dan belum dicoba:(**

*sudo microk8s kubectl apply --filename https://storage.googleapis.com/tekton-releases/pipeline/latest/release.yaml*

*sudo microk8s kubectl apply --filename https://storage.googleapis.com/tekton-releases/triggers/latest/release.yaml*

*sudo microk8s kubectl get pods --namespace tekton-pipelines*

*sudo microk8s kubectl create namespace mynamespace*

*sudo microk8s kubectl get namespaces*

*openssl rand -hex 32*

*sudo microk8s kubectl create secret generic github-secret --from-literal=secretToken=db6ac55208a5d48509dee6361fdca5ae5035acd3670b475f0b7b5f996fd9e8ed -n mytekton*

*##secret/github-secret created*

*sudo microk8s kubectl get secrets -n mytekton*

**pipeline.yaml berisikan:**

* **Jenis:** Pipeline
* **Workspace:** pipeline-workspace
* **Tasks:** init (cleanup workspace), git clone, flake8 lint, nose, #deploy

**tasks.yaml berisikan:**

* **Jenis:** Task
* **Name Tasks:** echo, cleanup, & nose

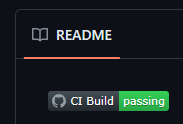
**pvc.yaml berisikan:**

* **Jenis:** PersistentVolumeClaim
* **name:**pipelinerun-pvc
* **Storage:** 500Mi
* **accessModes:** ReadWriteMany

Missing: full-setup prometheus, grafana, & tekton.

README.md:

![Build Status](https://github.com/Rayhand117/swarm-to-k8s/actions/workflows/ci-build.yaml/badge.svg)



**Enable Kubernetes Dashboard OVERDUE**

*sudo microk8s enable dashboard*

*sudo microk8s kubectl get all –all-namespaces*

*TOKEN=$(openssl rand -base64 32)*

**Create secret**

*sudo microk8s kubectl create secret generic kubernetes-dashboard-token -n kube-system –from-literal=token=$TOKEN*

**Patching dashboard**

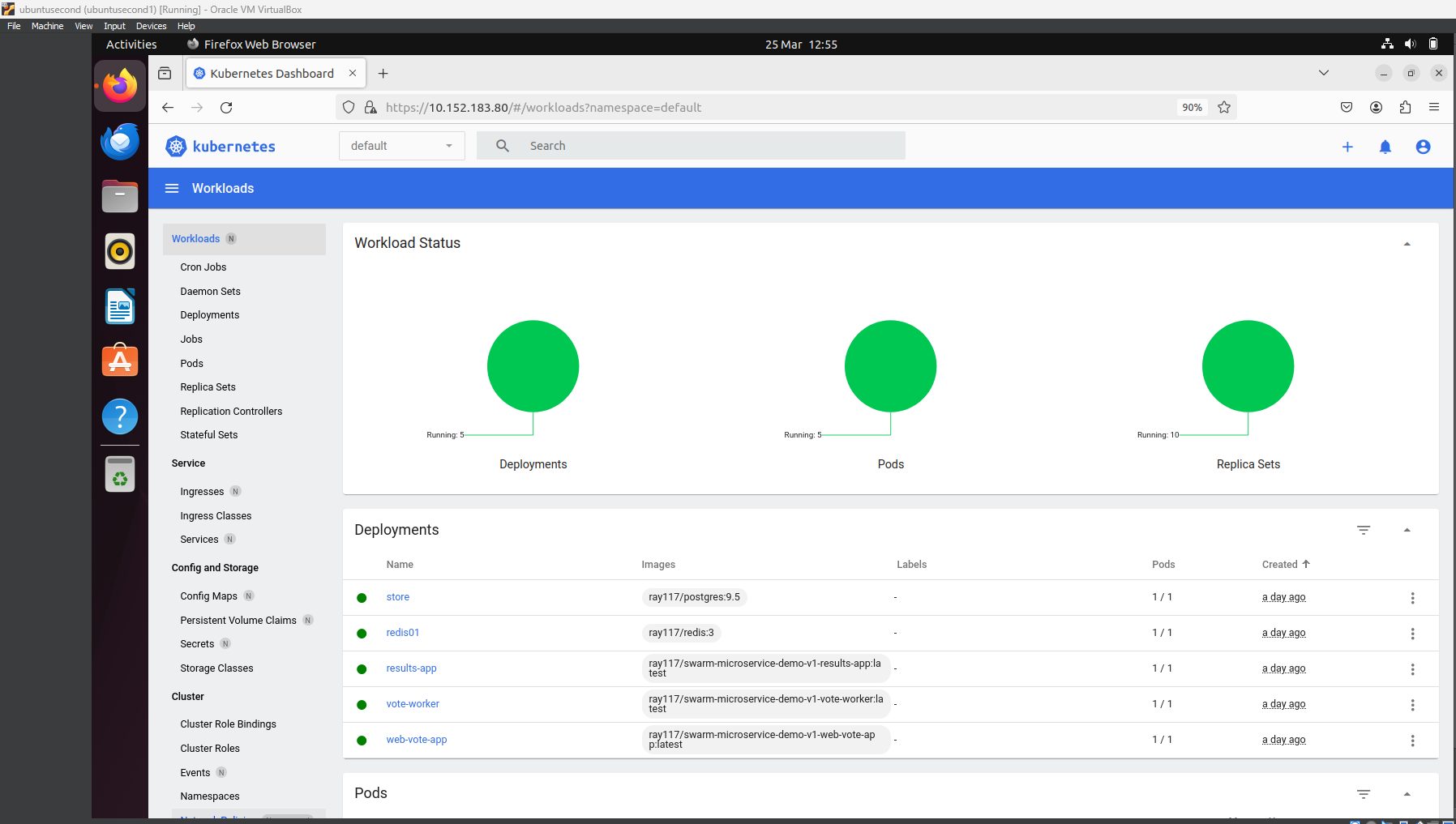
*sudo microk8s kubectl patch serviceaccount kubernetes-dashboard -n kube-system -p ‘{“secrets”: [{“name”: “kubernetes-dashboard-token”}]}’*

**Retrieve secret name**

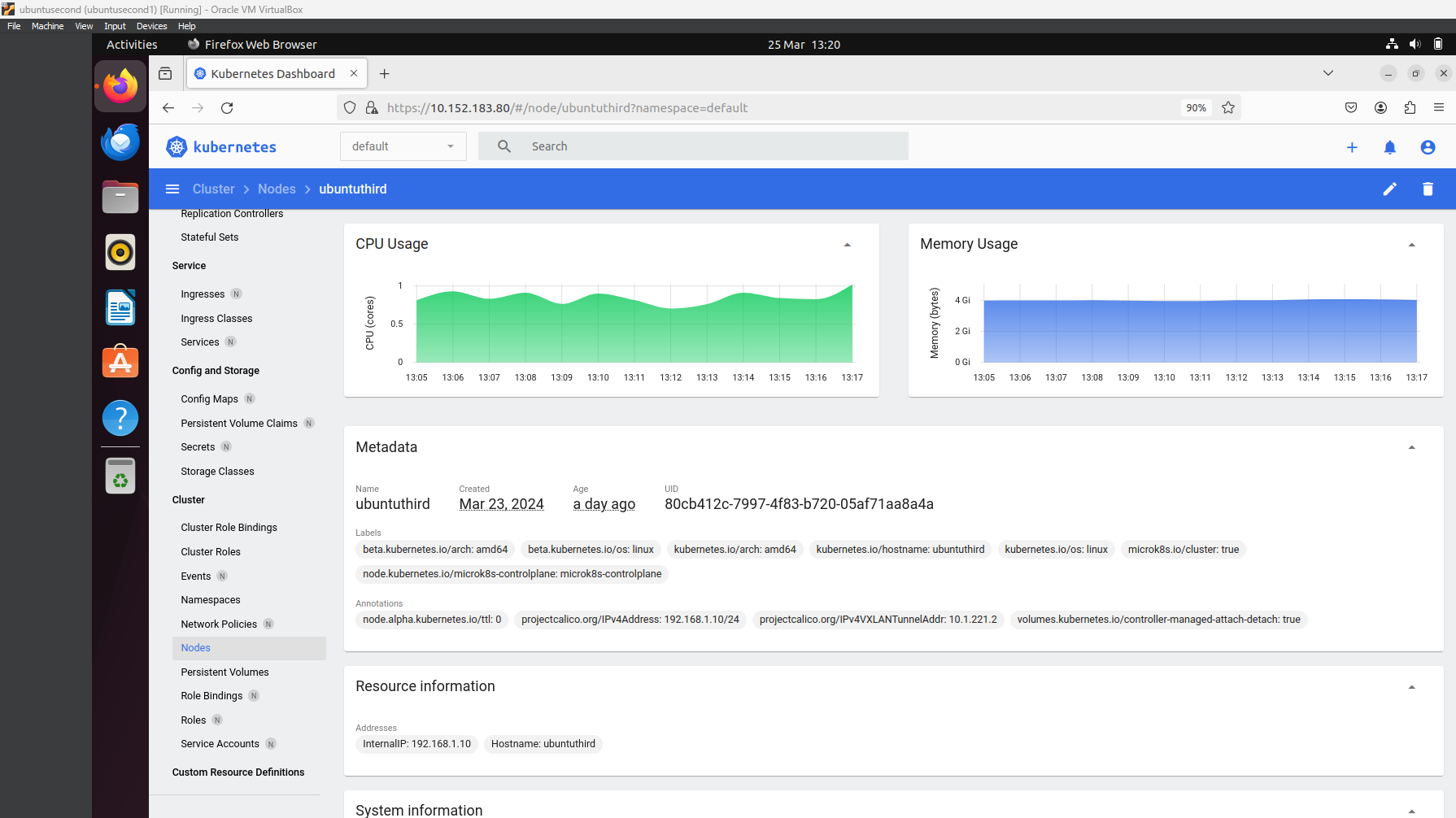
*sudo microk8s kubectl -n kube-system get serviceaccount/kubernetes-dashboard -o jsonpath=”{.secrets[0].name}”*

**Access Kubernetes dashboard & show token**

*sudo microk8s dashboard-proxy*



Gambar 1. 24 Kubernetes Dashboard

****

Gambar 1. 25 Ubuntuthird node