Terdapat beberapa tugas yang perlu dikerjakan sebagai berikut:

Deploy service demo microservice https://github.com/docker-archive/swarm-microservice-demo-v1 dengan spesifikasi environment sebagai berikut (Point 80) :

1. Siapkan 3 unit server. Bisa dalam bentuk VM di laptop/PC atau VPS. **(Berhasil)**
2. Setup cluster kubernetes di 3 server diatas. **(Berhasil)**
3. Sample service diatas dirancang untuk dideploy di docker swarm. Sesuaikan agar service berjalan lancar diatas kubernetes. **(Berhasil)**
4. Setup sistem CI untuk service diatas. **(Tidak Berhasil)**
5. Kemudian dokumentasikan secara lengkap langkah-langkah deployment tadi mulai dari instalasi OS VM/VPS, setup kubernetes, deployment service, deployment ci/cd, sampai ke demonstrasi fitur CI.
6. Dokumen diatas bisa dalam bentuk pdf/doc & script deployment harus ikut disertakan dengan ketentuan yang telah dijelaskan di halaman terakhir dokumen soal ini.

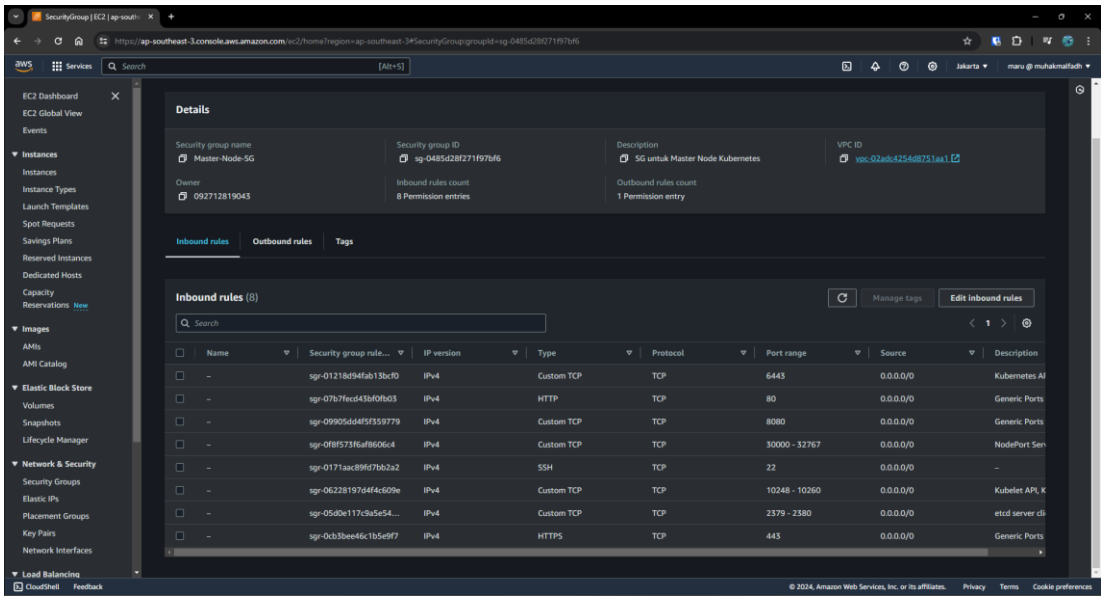
Bonus poin apabila menggunakan :

1. terraform/ansible untuk deployment & setup VM/VPS **(Berhasil)**
2. Rancher untuk deployment Kubernetes
3. Gitlab / jenkins untuk CI
4. Setup logging service dengan ELK/grafana loki
5. Setup monitoring & alerting dengan grafana & Prometheus

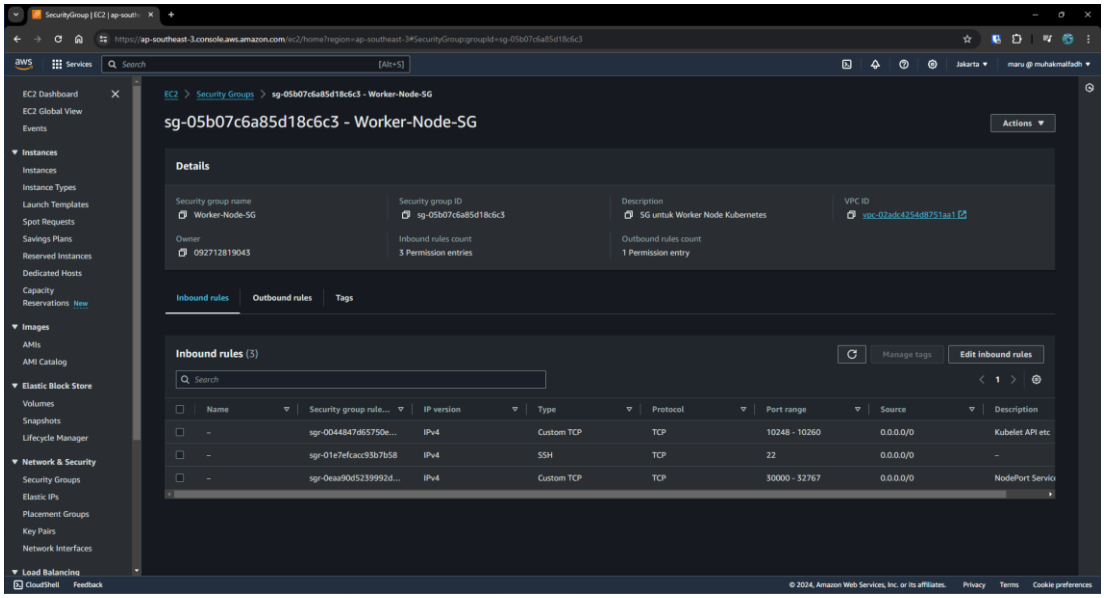
Git Repository: https://github.com/muhakmalfadh/microservice-demo.git

Membuat Cluster Kubernetes pada EC2 Instance

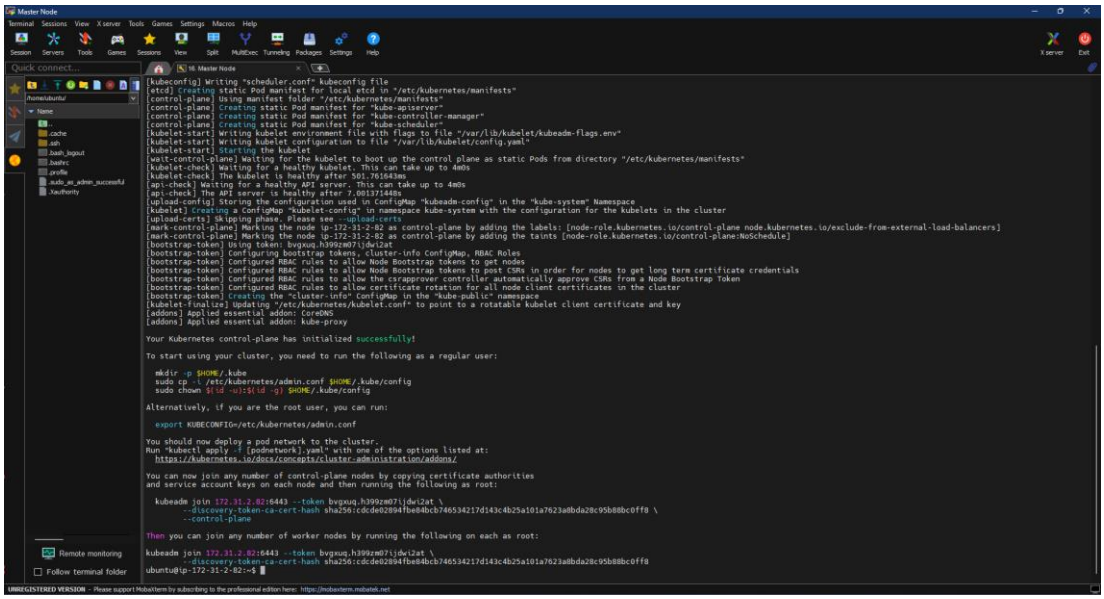
1. Membuat Security Group untuk Master Node



2. Membuat Security Group untuk Worker Node



3. Membuat 3 instance (1 Master Node dan 2 Worker Node) serta menginstal docker, kubeadm, kubelet, kubectl, dan kubernetes-cni pada masing-masing instance menggunakan Terraform
4. Inisialisasi Kubernetes Cluster pada Master Node menggunakan command berikut:
   `sudo kubeadm init --control-plane-endpoint=172.31.2.82 --pod-network-cidr=10.244.0.0/16`



5. Untuk menggunakan cluster, jalankan command berikut:
   `mkdir -p $HOME/.kube`
   `sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config`
   `sudo chown $(id -u):$(id -g) $HOME/.kube/config`
6. Menginstal plugin network Flannel
   `kubectl apply -f https://github.com/flannel-io/flannel/releases/latest/download/kube-flannel.yml`

```
ubuntu@ip-172-31-2-82:~$ kubectl apply -f https://github.com/flannel-io/flannel/releases/latest/download/kube-flannel.yml
namespace/kube-flannel created
serviceaccount/flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

7. Memastikan seluruh pods telah berjalan dengan baik
   `kubectl get pods --all-namespaces`

```
ubuntu@ip-172-31-2-82:~$ kubectl get pods --all-namespaces
NAMESPACE      NAME                                     READY   STATUS    RESTARTS   AGE
kube-flannel   kube-flannel-ds-2q88d                    1/1     Running   0          100s
kube-system    coredns-7db6d8ff4d-fwnzp                 1/1     Running   0          3m7s
kube-system    coredns-7db6d8ff4d-kr5pg                 1/1     Running   0          3m7s
kube-system    etcd-ip-172-31-2-82                      1/1     Running   0          3m22s
kube-system    kube-apiserver-ip-172-31-2-82            1/1     Running   0          3m22s
kube-system    kube-controller-manager-ip-172-31-2-82   1/1     Running   0          3m22s
kube-system    kube-proxy-t5ddz                         1/1     Running   0          3m7s
kube-system    kube-scheduler-ip-172-31-2-82            1/1     Running   0          3m22s
```

8. Menggabungkan Worker Node ke dalam cluster menggunakan command berikut dengan user root:

```
sudo su –
kubeadm join 172.31.2.82:6443 --token bvgxuq.h399zm07ijdwi2at
--discovery-token-ca-cert-hash
sha256:cdcde02894fbe84bcb746534217d143c4b25a101a7623a8bda28c95
b88bc0ff8
```

```
root@ip-172-31-8-17:~# kubeadm join 172.31.2.82:6443 --token bvgxuq.h399zm07ijdwi2at --discovery-token-ca-cert-hash sha256:cdcde02894fbe84bcb746534217d143c4b25a101a7623a8bda28c95b88bc0ff8
[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 kubeadm-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/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 501.05099ms
[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.
```

```
root@ip-172-31-15-85:~# kubeadm join 172.31.2.82:6443 --token bvgxuq.h399zm07ijdwi2at --discovery-token-ca-cert-hash sha256:cdcde02894fbe84bcb746534217d143c4b25a101a7623a8bda28c95b88bc0ff8
[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 kubeadm-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/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 509.473885ms
[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.
```

9. Memastikan Worker Nodes telah masuk ke dalam cluster dengan command:

```
kubectl get nodes
```

```
ubuntu@ip-172-31-2-82:~$ kubectl get nodes
NAME              STATUS   ROLES           AGE    VERSION
ip-172-31-15-85   Ready    <none>          117s   v1.30.1
ip-172-31-2-82    Ready    control-plane   9m2s   v1.30.1
ip-172-31-8-17    Ready    <none>          2m3s   v1.30.1
```

10. Memberikan ROLES pada worker node:

```
kubectl label node ip-172-31-8-17 node-
role.kubernetes.io/worker=worker
kubectl label node ip-172-31-15-85 node-
role.kubernetes.io/worker=worker
```

```
ubuntu@ip-172-31-2-82:~$ kubectl get nodes
NAME              STATUS   ROLES           AGE     VERSION
ip-172-31-15-85   Ready    worker          5m41s   v1.30.1
ip-172-31-2-82    Ready    control-plane   12m     v1.30.1
ip-172-31-8-17    Ready    worker          5m47s   v1.30.1
```

Menginstal Jenkins di Kubernetes

1. Git clone Jenkins Kubernetes Manifest Files yang tersedia di dokumentasi Jenkins:

   `git clone https://github.com/scriptcamp/kubernetes-jenkins`

```
ubuntu@ip-172-31-2-82:~/scripts$ git clone https://github.com/scriptcamp/kubernetes-jenkins
Cloning into 'kubernetes-jenkins'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 16 (delta 1), reused 0 (delta 0), pack-reused 9
Receiving objects: 100% (16/16), done.
Resolving deltas: 100% (1/1), done.
```

2. Buat Namespace untuk Jenkins:

   `kubectl create namespace devops-tools`

```
ubuntu@ip-172-31-2-82:~/scripts$ kubectl create namespace devops-tools
namespace/devops-tools created
```

3. Buat service account menggunakan script serviceAccount.yaml

   `kubectl apply -f serviceAccount.yaml`

```
ubuntu@ip-172-31-15-78:~/scripts/kubernetes-jenkins$ kubectl apply -f serviceAccount.yaml
clusterrole.rbac.authorization.k8s.io/jenkins-admin created
serviceaccount/jenkins-admin created
clusterrolebinding.rbac.authorization.k8s.io/jenkins-admin created
```

4. Membuat volume menggunakan script volume.yaml (ganti “worker-node01” dengan nama worker node yang digunakan)

   `kubectl create -f volume.yaml`

```
ubuntu@ip-172-31-2-82:~/scripts/kubernetes-jenkins$ vim volume.yaml
ubuntu@ip-172-31-2-82:~/scripts/kubernetes-jenkins$ kubectl create -f volume.yaml
storageclass.storage.k8s.io/local-storage created
persistentvolume/jenkins-pv-volume created
persistentvolumeclaim/jenkins-pv-claim created
```

5. Membuat deployment menggunakan script deployment.yaml

   `kubectl apply -f deployment.yaml`

```
ubuntu@ip-172-31-2-82:~/scripts/kubernetes-jenkins$ kubectl apply -f deployment.yaml
deployment.apps/jenkins created
```

6. Memeriksa status deployment

   `kubectl get deployments -n devops-tools`

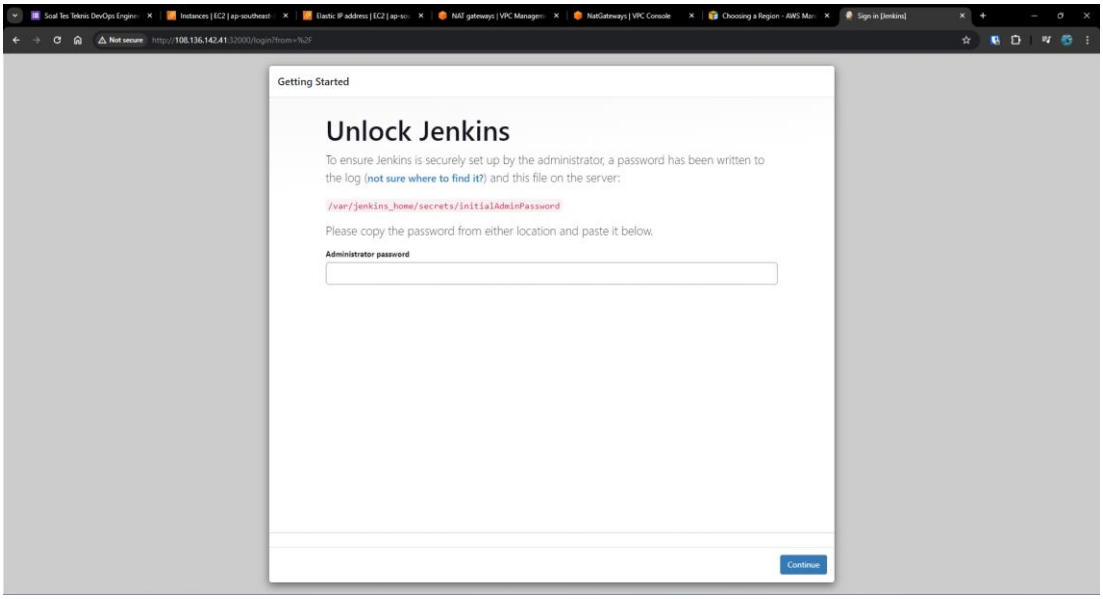
```
Every 2.0s: kubectl get deployments -n devops-tools

NAME      READY   UP-TO-DATE   AVAILABLE   AGE
jenkins   1/1     1            1           2m7s
```

7. Membuat service Jenkins menggunakan kubectl

   `kubectl apply -f service.yaml`

```
ubuntu@ip-172-31-2-82:~/scripts/kubernetes-jenkins$ kubectl apply -f service.yaml
service/jenkins-service created
```

8. Akses Jenkins melalui Alamat IP Worker Node yang digunakan pada Langkah 4. Akses dilakukan ke alamat port 32000 (http://<node-ip>:32000)



9. Administrator password untuk masuk ke dalam Jenkins didapatkan melalui pod yang menjalankan Jenkins

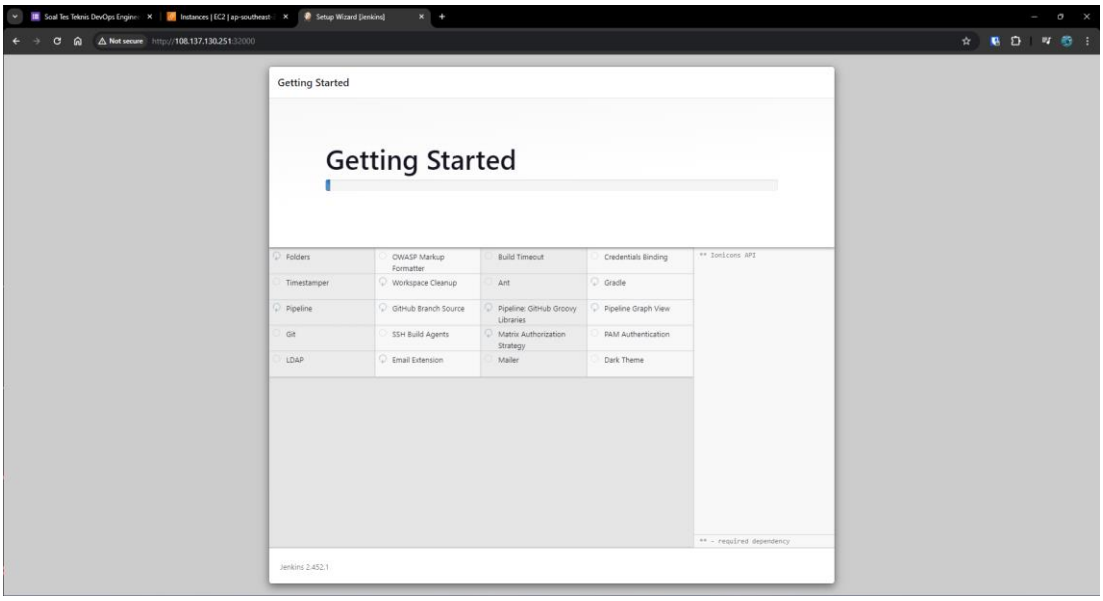
```
kubectl get pods --namespace=devops-tools
```

```
ubuntu@ip-172-31-2-82:~/scripts/kubernetes-jenkins$ kubectl get pods --namespace=devops-tools
NAME                       READY   STATUS    RESTARTS   AGE
jenkins-68c8b7f55-qrthg    1/1     Running   0          7m9s
```

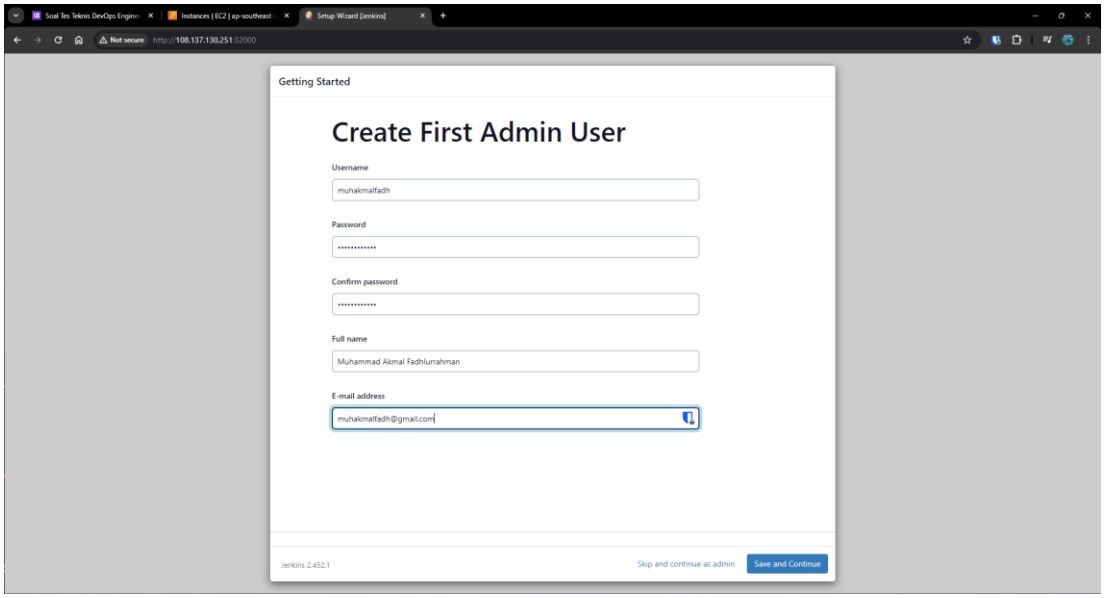
```
kubectl exec -it jenkins-68c8b7f55-lqp9f -n devops-tools --
cat /var/jenkins_home/secrets/initialAdminPassword
```

```
ubuntu@ip-172-31-2-82:~/scripts/kubernetes-jenkins$ kubectl exec -it jenkins-68c8b7f55-qrthg -n devops-tools -- cat /var/jenkins_home/secrets/initialAdminPassword
14264f60efbb4bbfbad49318e732a825
```

10. Menginstal plugins untuk digunakan oleh Jenkins



11. Membuat akun user admin



Mencoba menjalankan aplikasi di local

1. Ketika mencoba menjalankan docker compose, terdapat error karena image java:7 sudah tidak bisa digunakan sehingga saya coba ganti menggunakan openjdk:8 dan seluruh image serta container berhasil dibuat

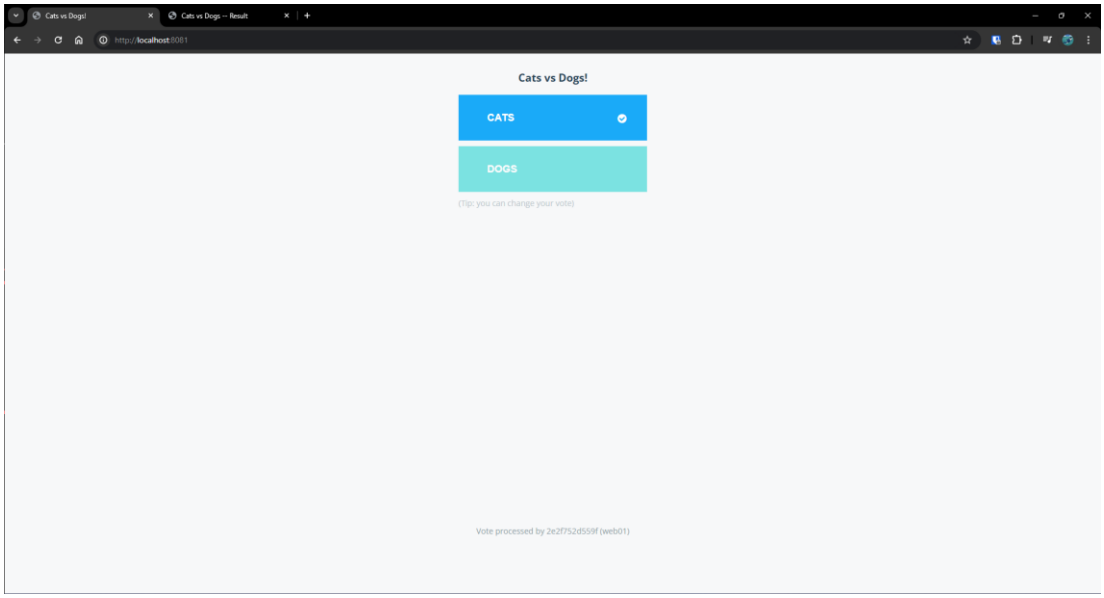
```
C:\Users\muaf1\OneDrive\Documents\Work\6. TLab - DevOps Engineer\microservice-demo\kube-deployment.yaml
service-demo> docker image ls
                                                    SIZE
                                                    657MB
microservice-demo-web-vote-app   latest    2374ee83ae88   15 minutes ago   907MB
microservice-demo-vote-worker    latest    bfb9162f1279   18 hours ago     844MB
postgres                         9.5       6d176851b77f   3 years ago      197MB
redis                            3         87856cc39862   5 years ago      76MB
PS C:\Users\muaf1\OneDrive\Documents\Work\6. TLab - DevOps Engineer\microservice-demo> docker container ls -a
CONTAINER ID   IMAGE                          COMMAND                  CREATED         STATUS    PORTS     NAMES
31fb85b6e709   redis:3                        "docker-entrypoint.s…"   3 minutes ago   Created             microservice-demo-redis01-1
f333ba9a6fe3   microservice-demo-vote-worker  "/usr/lib/jvm/java-7…"   3 minutes ago   Created             microservice-demo-vote-worker-1
650cc355201f   microservice-demo-web-vote-app "python app.py"          3 minutes ago   Created             microservice-demo-web-vote-app-1
4d1f15848186   microservice-demo-results-app  "node server.js"         3 minutes ago   Created             microservice-demo-results-app-1
01e51dedae64   postgres:9.5                   "docker-entrypoint.s…"   3 minutes ago   Created             microservice-demo-store-1
```

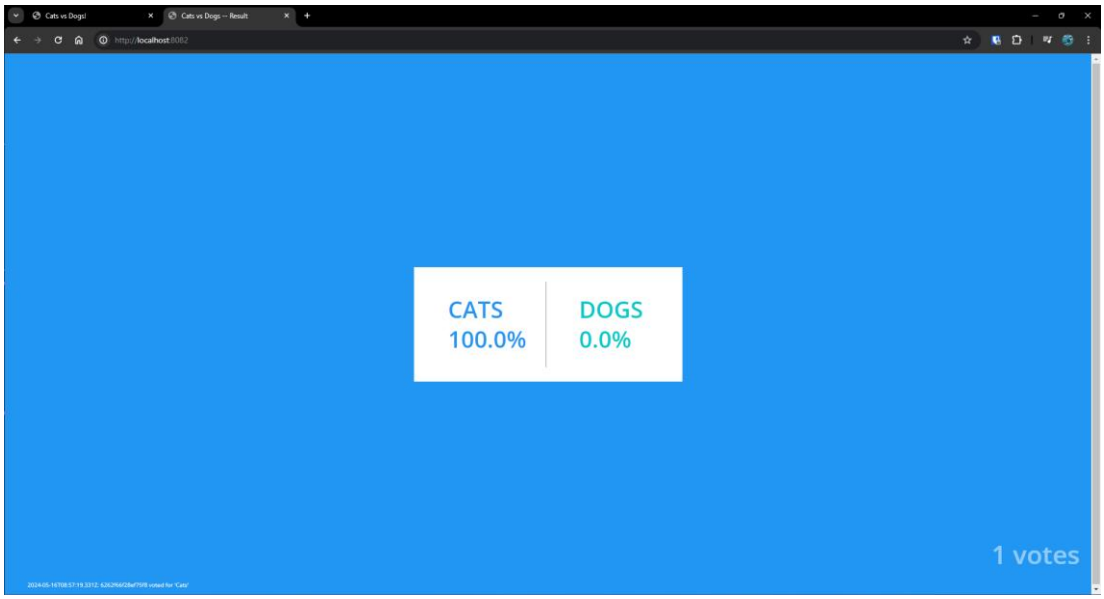
2. Adapun ketika dijalankan, container microservice-demo-results-app tidak bisa dijalankan

```
PS C:\Users\muaf1\OneDrive\Documents\Work\6. TLab - DevOps Engineer\microservice-demo> docker compose -f .\docker-compose.yml -f .\docker-compose.override.yml start
[+] Running 5/5
 ✓ Container microservice-demo-redis01-1       Started
 ✓ Container microservice-demo-vote-worker-1   Started
 ✓ Container microservice-demo-store-1         Started
 ✓ Container microservice-demo-web-vote-app-1  Started
 ✓ Container microservice-demo-results-app-1   Started
PS C:\Users\muaf1\OneDrive\Documents\Work\6. TLab - DevOps Engineer\microservice-demo> docker container ls
CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS         PORTS                    NAMES
c0685381ddb0   postgres:9.5                   "docker-entrypoint.s…"   12 seconds ago   Up 5 seconds   5432/tcp                 microservice-demo-store-1
99ac011bf5ea   microservice-demo-vote-worker  "java -jar target/wo…"   12 seconds ago   Up 5 seconds                            microservice-demo-vote-worker-1
9240fe8e55af   microservice-demo-web-vote-app "python app.py"          12 seconds ago   Up 4 seconds   0.0.0.0:8081->80/tcp     microservice-demo-web-vote-app-1
1e892e16164c   redis:3                        "docker-entrypoint.s…"   12 seconds ago   Up 5 seconds   0.0.0.0:6379->6379/tcp   microservice-demo-redis01-1
PS C:\Users\muaf1\OneDrive\Documents\Work\6. TLab - DevOps Engineer\microservice-demo> docker compose -f .\docker-compose.yml -f .\docker-compose.override.yml start
[+] Running 1/1
 ✓ Container microservice-demo-results-app-1  Started
PS C:\Users\muaf1\OneDrive\Documents\Work\6. TLab - DevOps Engineer\microservice-demo> docker container ls
CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS          PORTS                    NAMES
c0685381ddb0   postgres:9.5                   "docker-entrypoint.s…"   35 seconds ago   Up 27 seconds   5432/tcp                 microservice-demo-store-1
99ac011bf5ea   microservice-demo-vote-worker  "java -jar target/wo…"   35 seconds ago   Up 28 seconds                            microservice-demo-vote-worker-1
9240fe8e55af   microservice-demo-web-vote-app "python app.py"          35 seconds ago   Up 27 seconds   0.0.0.0:8081->80/tcp     microservice-demo-web-vote-app-1
1e892e16164c   redis:3                        "docker-entrypoint.s…"   35 seconds ago   Up 27 seconds   0.0.0.0:6379->6379/tcp   microservice-demo-redis01-1
```

3. Service results-app menggunakan node:0.10, terdapat penggunaan library yang tidak didukung sehingga saya coba ganti menggunakan node:4.0 sehingga semua service berhasil dijalankan

```
PS C:\Users\muaf1\OneDrive\Documents\Work\6. TLab - DevOps Engineer\microservice-demo> docker compose -f .\docker-compose.yml -f .\docker-compose.override.yml start
 [+] Running 5/5
  ✓ Container microservice-demo-results-app-1   Started
  ✓ Container microservice-demo-redis01-1       Started
  ✓ Container microservice-demo-web-vote-app-1  Started
  ✓ Container microservice-demo-store-1         Started
  ✓ Container microservice-demo-vote-worker-1   Started
PS C:\Users\muaf1\OneDrive\Documents\Work\6. TLab - DevOps Engineer\microservice-demo> docker container ls
CONTAINER ID   IMAGE                        COMMAND                  CREATED             STATUS          PORTS                    NAMES
dacfdf7d72ec   postgres:9.5                 "docker-entrypoint.s…"   About a minute ago  Up 13 seconds   5432/tcp                 microservice-demo-store-1
2e2f752d559f   microservice-demo-web-vote-app "python app.py"        About a minute ago  Up 13 seconds   0.0.0.0:8081->80/tcp     microservice-demo-web-vote-app-1
79d102b4d730   microservice-demo-results-app  "node server.js"         About a minute ago  Up 13 seconds   0.0.0.0:8082->80/tcp     microservice-demo-results-app-1
311ee5524cff   microservice-demo-vote-worker  "java -jar target/wo…"   About a minute ago  Up 14 seconds                            microservice-demo-vote-worker-1
700602c57d40   redis:3                      "docker-entrypoint.s…"   About a minute ago  Up 14 seconds   0.0.0.0:6379->6379/tcp   microservice-demo-redis01-1
```





Deployment aplikasi ke Kubernetes Cluster

1. Membuat namespace microservice-demo untuk service yang akan di-deploy
   kubectl create namespace microservice-demo

```
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl create namespace microservice-demo
namespace/microservice-demo created
```

2. Membuat file Kubernetes Manifest untuk masing-masing service dan melakukan deployment database dan redis terlebih dahulu
   kubectl apply -f <nama-manifest-file>.yaml

```
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl get all --namespace=microservice-demo
NAME                                    READY   STATUS    RESTARTS   AGE
pod/redis01                             1/1     Running   0          12m
pod/results-app-866d4bdbd9-tr798        1/1     Running   0          14s
pod/store                               1/1     Running   0          10m
pod/vote-worker-695f6c554-749jd         1/1     Running   0          9m34s
pod/web-vote-app-66b8bb46bc-jrmn6       1/1     Running   0          4m18s

NAME                   TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
service/results-app    NodePort   10.99.250.70   <none>        80:30081/TCP   14s
service/web-vote-app   NodePort   10.110.39.48   <none>        80:30080/TCP   4m18s

NAME                              READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/results-app       1/1     1            1           14s
deployment.apps/vote-worker       1/1     1            1           9m34s
deployment.apps/web-vote-app      1/1     1            1           4m18s

NAME                                         DESIRED   CURRENT   READY   AGE
replicaset.apps/results-app-866d4bdbd9       1         1         1       14s
replicaset.apps/vote-worker-695f6c554        1         1         1       9m34s
replicaset.apps/web-vote-app-66b8bb46bc      1         1         1       4m18s
```

3. Mendapatkan alamat IP pod milik database dan redis
   kubectl get pod -n microservice-demo -o wide

```
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl get pod -n microservice-demo -o wide
NAME                      READY   STATUS    RESTARTS   AGE    IP            NODE              NOMINATED NODE   READINESS GATES
redis01                   1/1     Running   0          70m    10.244.2.12   ip-172-31-15-85   <none>           <none>
store                     1/1     Running   0          79m    10.244.2.11   ip-172-31-15-85   <none>           <none>
```

4. Perbarui kode service vote-worker untuk terhubung ke database dan redis menggunakan alamat IP (worker.java)

```java
static Jedis connectToRedis(String host) {

  //For production
  String redisServiceHost = "10.244.2.12";
  Jedis conn = new Jedis(redisServiceHost);
```

```java
static Connection connectToDB(String host) throws SQLException {
  Connection conn = null;
  String password = "pg8675309";

  try {

    Class.forName(className:"org.postgresql.Driver");
    //For production
    String url = "jdbc:postgresql://10.244.2.11/postgres";
```

5. Perbarui kode service web-vote-app untuk terhubung ke redis menggunakan alamat IP (app.py)

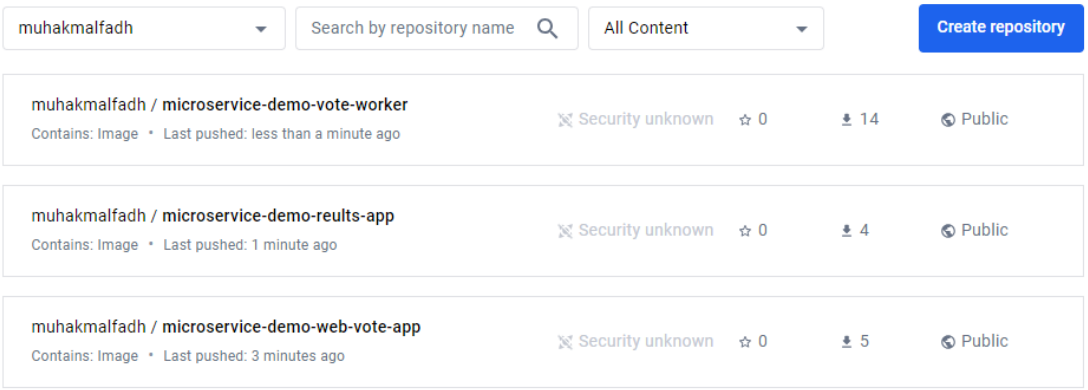
```
db_server = "10.244.2.12"
```

6. Perbarui kode service results-app untuk terhubung ke database menggunakan alamat IP (server.js)

```
async.retry(
  {times: 1000, interval: 1000},
  function(callback) {
    //for development
    //pg.connect('postgres://postgres:pg8675309@store/postgres', function(err, client, done) {

    //for production
    pg.connect('postgres://postgres:pg8675309@10.244.2.11/postgres', function(err, client, done) {
```

7. Mengunggah image database dan redis ke Docker Hub

   docker push <nama-image>



8. Melakukan deployment vote-worker, web-vote-app, dan results-app

   kubectl apply -f <nama-file>.yaml

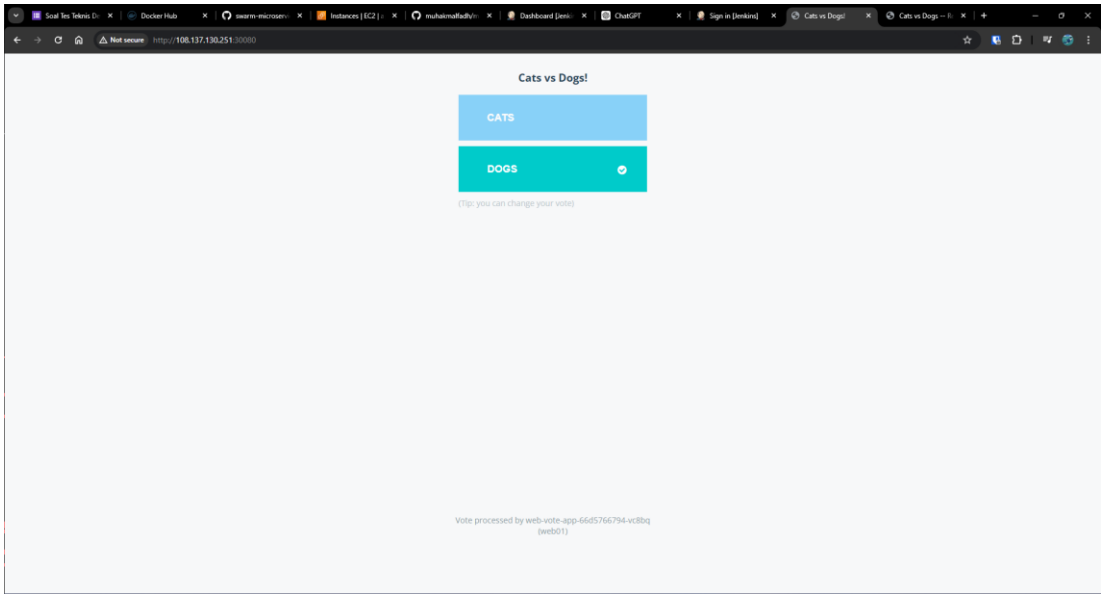
```
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl apply -f vote-worker.yaml
deployment.apps/vote-worker created
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl apply -f results-app.yaml
deployment.apps/results-app created
service/results-app created
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl apply -f web-vote-app.yaml
deployment.apps/web-vote-app created
service/web-vote-app created
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl get pod -n microservice-demo
NAME                              READY   STATUS    RESTARTS   AGE
redis01                           1/1     Running   0          104m
results-app-5cb5cc9f95-n7mp4      1/1     Running   0          38s
store                             1/1     Running   0          112m
vote-worker-64f8d9846f-nckn4      1/1     Running   0          109s
web-vote-app-66d5766794-vc8bq     1/1     Running   0          23s
```

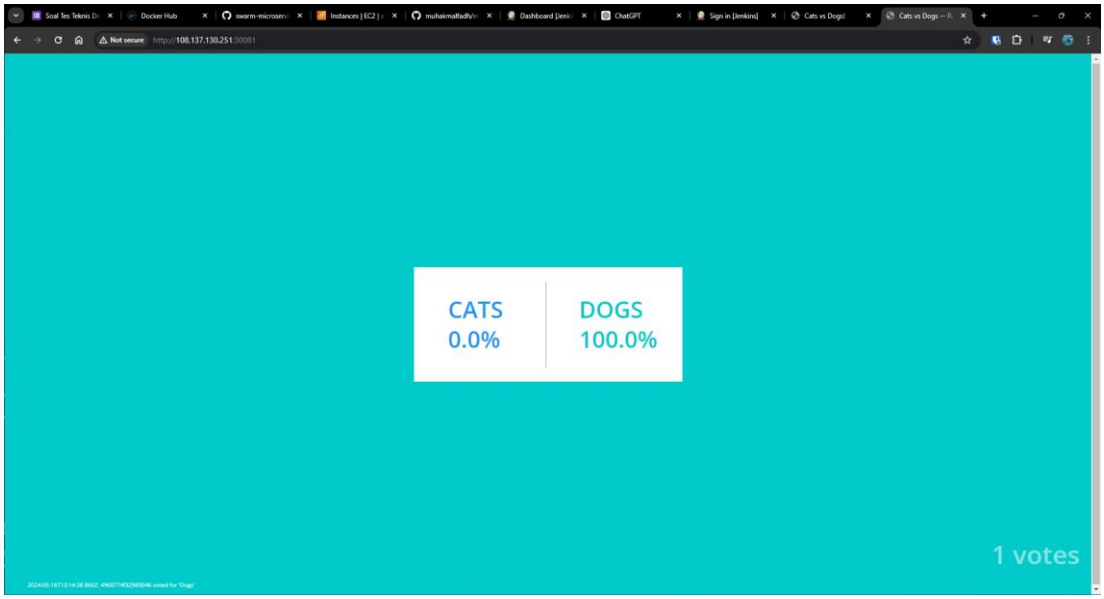
9. Periksa log masing-masing pod dan pastikan berjalan dengan baik

   kubectl logs <nama-pod> -n microservice-demo

```
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl logs -n microservice-demo vote-worker-64f8d9846f-nckn4
1 redis hosts
  redisHosts[0] = 'redis01'
Connected to redis
Watching vote queue
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl logs -n microservice-demo web-vote-app-66d5766794-vc8bq
Connecting to redis
Connected to redis
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 241-128-645
ubuntu@ip-172-31-2-82:~/apps/microservice-demo/kube deployment$ kubectl logs -n microservice-demo results-app-5cb5cc9f95-n7mp4
Thu, 16 May 2024 13:09:17 GMT body-parser deprecated bodyParser: use individual json/urlencoded middlewares at server.js:77:9
Thu, 16 May 2024 13:09:17 GMT body-parser deprecated undefined extended: provide extended option at ../node_modules/body-parser/index.js:104:29
App running on port 80
Connected to db
```

10. Mencoba mengakses dan menggunakan aplikasi


Cats vs Dogs!
CATS
DOGS
(Tip: you can change your vote)
Vote processed by web-vote-app-66d5766794-vc8bq
(web01)


CATS
0.0%
DOGS
100.0%
1 votes

LAMPIRAN

Script main.tf

```
provider "aws" {
  region = "ap-southeast-3"
}

data "aws_security_group" "master_node_sg" {
  id = "sg-0485d28f271f97bf6"
  name = "Master-Node-SG"
}

data "aws_security_group" "worker_node_sg" {
  id = "sg-05b07c6a85d18c6c3"
  name = "Worker-Node-SG"
}

data "aws_ami" "ubuntu" {
  most_recent = true

  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-*"]
  }

  filter {
    name   = "virtualization-type"
    values = ["hvm"]
  }

  owners = ["099720109477"] # Canonical
}

resource "aws_instance" "master_node" {
  count         = 1
  ami           = data.aws_ami.ubuntu.id
  instance_type = "t3.medium"
  key_name      = "TLab"
  security_groups = [data.aws_security_group.master_node_sg.name]

  root_block_device {
    volume_size = 20
    volume_type = "gp2"
  }

  connection {
    host = self.public_ip
    type = "ssh"
    user = "ubuntu"
    private_key = file("../TLab.pem")
    timeout = "2m"
  }
```

```
  provisioner "remote-exec" {
    inline = [
      "echo 'net.ipv4.ip_forward = 1' | sudo tee /etc/sysctl.d/k8s.conf >
/dev/null",
      "sudo sysctl --system",
      "sysctl net.ipv4.ip_forward",
      "sudo apt-get update",
      "sudo apt-get install ca-certificates curl -y",
      "sudo install -m 0755 -d /etc/apt/keyrings",
      "sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc",
      "sudo chmod a+r /etc/apt/keyrings/docker.asc",
      "echo \"deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu $(.
/etc/os-release && echo \"$VERSION_CODENAME\") stable\" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null",
      "sudo sed -i 's/\"//g' /etc/apt/sources.list.d/docker.list",
      "sudo apt-get update",
      "sudo apt-get install docker-ce docker-ce-cli containerd.io docker-
buildx-plugin docker-compose-plugin -y",
      "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",
      "sudo systemctl restart containerd",
      "sudo systemctl enable containerd",
      "sudo apt-get update",
      "sudo apt-get install -y apt-transport-https ca-certificates curl gpg",
      "curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key |
sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg",
      "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",
      "sudo apt-get update",
      "sudo apt-get install -y kubelet kubeadm kubectl kubernetes-cni",
      "sudo apt-mark hold kubelet kubeadm kubectl",
      "sudo systemctl enable --now kubelet",
      "swapoff -a",
      "sudo echo 'overlay' | sudo tee /etc/modules-load.d/containerd.conf >
/dev/null",
      "sudo echo 'br_netfilter' | sudo tee -a /etc/modules-
load.d/containerd.conf > /dev/null",
      "sudo modprobe overlay",
      "sudo modprobe br_netfilter",
      "sudo echo 'net.bridge.bridge-nf-call-ip6tables = 1' | sudo tee
/etc/sysctl.d/kubernetes.conf > /dev/null",
      "sudo echo 'net.bridge.bridge-nf-call-iptables = 1' | sudo tee -a
/etc/sysctl.d/kubernetes.conf > /dev/null",
      "sudo echo 'net.ipv4.ip_forward = 1' | sudo tee -a
/etc/sysctl.d/kubernetes.conf > /dev/null",
      "sudo sysctl --system"
     ]
  }

  tags = {
    Name = "Master-Node"
  }
}
```

```
resource "aws_instance" "worker_node" {
  count         = 2
  ami           = data.aws_ami.ubuntu.id
  instance_type = "t3.medium"
  key_name      = "TLab"
  security_groups = [data.aws_security_group.worker_node_sg.name]

  root_block_device {
    volume_size = 20
    volume_type = "gp2"
  }

  connection {
    host = self.public_ip
    type = "ssh"
    user = "ubuntu"
    private_key = file("../TLab.pem")
    timeout = "2m"
  }

  provisioner "remote-exec" {
    inline = [
      "echo 'net.ipv4.ip_forward = 1' | sudo tee /etc/sysctl.d/k8s.conf >
/dev/null",
      "sudo sysctl --system",
      "sysctl net.ipv4.ip_forward",
      "sudo apt-get update",
      "sudo apt-get install ca-certificates curl -y",
      "sudo install -m 0755 -d /etc/apt/keyrings",
      "sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc",
      "sudo chmod a+r /etc/apt/keyrings/docker.asc",
      "echo \"deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu $(.
/etc/os-release && echo \"$VERSION_CODENAME\") stable\" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null",
      "sudo sed -i 's/\"//g' /etc/apt/sources.list.d/docker.list",
      "sudo apt-get update",
      "sudo apt-get install docker-ce docker-ce-cli containerd.io docker-
buildx-plugin docker-compose-plugin -y",
      "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",
      "sudo systemctl restart containerd",
      "sudo systemctl enable containerd",
      "sudo apt-get update",
      "sudo apt-get install -y apt-transport-https ca-certificates curl gpg",
      "curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key |
sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg",
      "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",
      "sudo apt-get update",
      "sudo apt-get install -y kubelet kubeadm kubectl kubernetes-cni",
      "sudo apt-mark hold kubelet kubeadm kubectl",
      "sudo systemctl enable --now kubelet",
      "swapoff -a",
```

```
      "sudo echo 'overlay' | sudo tee /etc/modules-load.d/containerd.conf >
/dev/null",
      "sudo echo 'br_netfilter' | sudo tee -a /etc/modules-
load.d/containerd.conf > /dev/null",
      "sudo modprobe overlay",
      "sudo modprobe br_netfilter",
      "sudo echo 'net.bridge.bridge-nf-call-ip6tables = 1' | sudo tee
/etc/sysctl.d/kubernetes.conf > /dev/null",
      "sudo echo 'net.bridge.bridge-nf-call-iptables = 1' | sudo tee -a
/etc/sysctl.d/kubernetes.conf > /dev/null",
      "sudo echo 'net.ipv4.ip_forward = 1' | sudo tee -a
/etc/sysctl.d/kubernetes.conf > /dev/null",
      "sudo sysctl --system"
     ]
  }

  tags = {
    Name = "Worker-Node-${count.index + 1}"
  }
}
```

Script redis01.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: redis01
  namespace: microservice-demo
spec:
  containers:
  - name: redis01
    image: redis:3
    ports:
    - containerPort: 6379
---
apiVersion: v1
kind: Service
metadata:
  name: redis01
  namespace: microservice-demo
spec:
  selector:
    app: redis01
  ports:
    - protocol: TCP
      port: 6379
      targetPort: 6379
```

Script store.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: store
  namespace: microservice-demo
spec:
  containers:
  - name: store
    image: postgres:9.5
    env:
    - name: POSTGRES_USER
      value: postgres
    - name: POSTGRES_PASSWORD
      value: pg8675309
---
apiVersion: v1
kind: Service
metadata:
  name: store-service
  namespace: microservice-demo
spec:
  selector:
    app: store
  ports:
    - protocol: TCP
      port: 5432
      targetPort: 5432
```

Script results-app.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: results-app
  namespace: microservice-demo
spec:
  replicas: 1
  selector:
    matchLabels:
      app: results-app
  template:
    metadata:
      labels:
        app: results-app
    spec:
      containers:
      - name: results-app
        image: muhakmalfadh/microservice-demo-reults-app:latest
        ports:
        - containerPort: 80
        env:
        - name: REDIS_HOST
          value: "redis01.microservice-demo.svc.cluster.local"
        - name: DATABASE_HOST
          value: "store-service.microservice-demo.svc.cluster.local"
---
apiVersion: v1
kind: Service
metadata:
  name: results-app
  namespace: microservice-demo
spec:
  type: NodePort
  selector:
    app: results-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
      nodePort: 30081
```

Script vote-worker.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: vote-worker
  namespace: microservice-demo
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vote-worker
  template:
    metadata:
      labels:
        app: vote-worker
    spec:
      containers:
      - name: vote-worker
        image: muhakmalfadh/microservice-demo-vote-worker:latest
        imagePullPolicy: Always
        env:
        - name: FROM_REDIS_HOST
          value: "1"
        - name: TO_REDIS_HOST
          value: "1"
        - name: REDIS_HOST
          value: "redis01.microservice-demo.svc.cluster.local"
        - name: DATABASE_HOST
          value: "store-service.microservice-demo.svc.cluster.local"
```

Script web-vote-app.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web-vote-app
  namespace: microservice-demo
spec:
  replicas: 1
  selector:
    matchLabels:
      app: web-vote-app
  template:
    metadata:
      labels:
        app: web-vote-app
    spec:
      containers:
      - name: web-vote-app
        image: muhakmalfadh/microservice-demo-web-vote-app:latest
        ports:
        - containerPort: 80
        env:
        - name: WEB_VOTE_NUMBER
          value: "01"
        - name: REDIS_HOST
          value: "redis01.microservice-demo.svc.cluster.local"
        - name: DATABASE_HOST
          value: "store-service.microservice-demo.svc.cluster.local"
---
apiVersion: v1
kind: Service
metadata:
  name: web-vote-app
  namespace: microservice-demo
spec:
  type: NodePort
  selector:
    app: web-vote-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
      nodePort: 30080
```