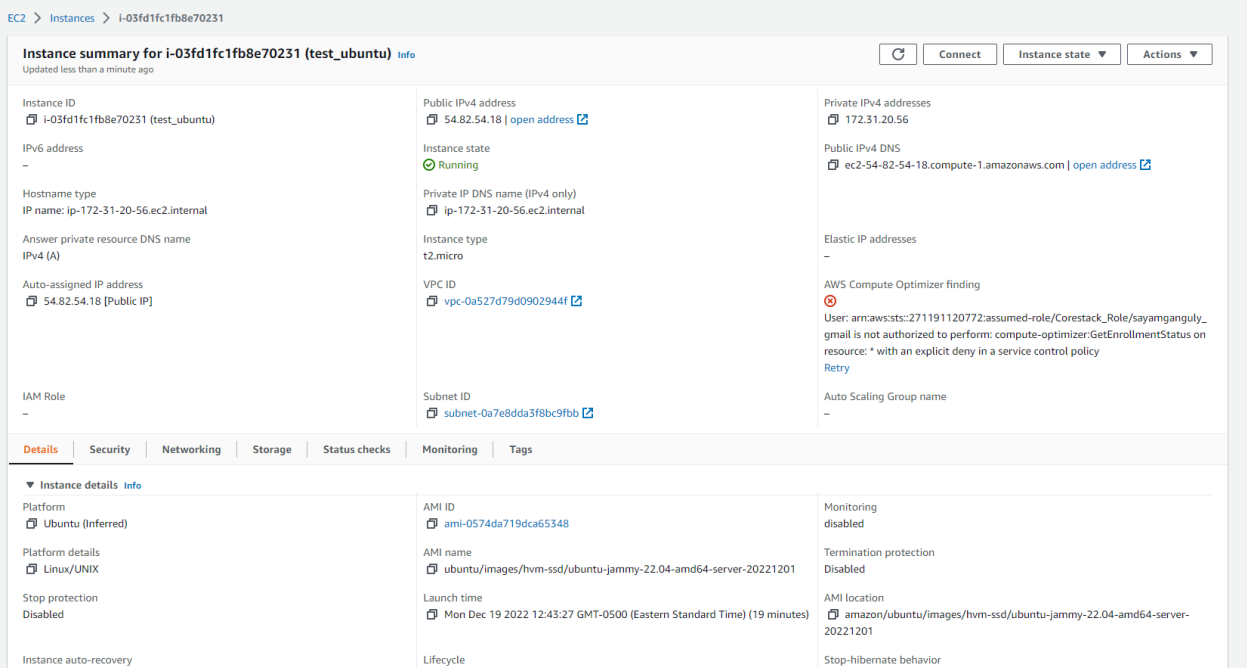
Infra Optimization

All the files used in here are uploaded to the repo - https://github.com/SayamGanguly/devops_capstone_project_infra_optimization

I have made use of AWS, Azure Lab framework and the Kubernetes labs to complete different aspects of the project.

Create an EC2 Instance and Install Docker and Kubernetes

First Create an EC2 Instance



Connect to the EC2 Instance

Install Docker

Verify if Docker is installed

```
root@ip-172-31-20-56:~# service docker status
Unit docker.service could not be found.
root@ip-172-31-20-56:~# docker version
Command 'docker' not found, but can be installed with:
snap install docker         # version 20.10.17, or
apt  install docker.io      # version 20.10.12-0ubuntu4
apt  install podman-docker  # version 3.4.4+ds1-1ubuntu1
See 'snap info docker' for additional versions.
root@ip-172-31-20-56:~#
```

Now I ran the following commands to install Docker

- `sudo su -`
- `apt-get update`
- `apt-get install ca-certificates curl gnupg lsb-release --assume-yes`
- `mkdir -p /etc/apt/keyrings`
- `curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --dearmor -o /etc/apt/keyrings/docker.gpg`
- echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | tee /etc/apt/sources.list.d/docker.list > /dev/null

- `apt-get update`
- `apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin --assume-yes`

Finally verify that Docker has been installed

```
root@ip-172-31-20-56:~# docker version
Client: Docker Engine - Community
 Version:           20.10.22
 API version:       1.41
 Go version:        go1.18.9
 Git commit:        3a2c30b
 Built:             Thu Dec 15 22:28:04 2022
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true

Server: Docker Engine - Community
 Engine:
  Version:          20.10.22
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.18.9
  Git commit:       42c8b31
  Built:            Thu Dec 15 22:25:49 2022
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.13
  GitCommit:        78f51771157abb6c9ed224c22013cdf09962315d
 runc:
  Version:          1.1.4
  GitCommit:        v1.1.4-0-g5fd4c4d
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
root@ip-172-31-20-56:~#
```

```
root@ip-172-31-20-56:~# service docker status
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2022-12-19 17:50:46 UTC; 1min 13s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 7607 (dockerd)
      Tasks: 7
     Memory: 33.5M
        CPU: 275ms
     CGroup: /system.slice/docker.service
             └─7607 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Dec 19 17:50:45 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:45.856270124Z" level=info msg="scheme \"unix\" not registered, fallback to default scheme" module=grpc
Dec 19 17:50:45 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:45.856452127Z" level=info msg="ccResolverWrapper: sending update to cc: {[{unix:///run/containerd/containerd.sock  <nil> 0 <nil>}] <nil> <nil>
Dec 19 17:50:45 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:45.856612843Z" level=info msg="ClientConn switching balancer to \"pick_first\"" module=grpc
Dec 19 17:50:45 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:45.939192634Z" level=info msg="Loading containers: start."
Dec 19 17:50:46 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:46.173163019Z" level=info msg="Default bridge (docker0) is assigned with an IP address 172.17.0.0/16. Daemon option --bip can be used to set
Dec 19 17:50:46 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:46.291318548Z" level=info msg="Loading containers: done."
Dec 19 17:50:46 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:46.371701715Z" level=info msg="Docker daemon" commit=42c8b31 graphdriver(s)=overlay2 version=20.10.22
Dec 19 17:50:46 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:46.372148511Z" level=info msg="Daemon has completed initialization"
Dec 19 17:50:46 ip-172-31-20-56 systemd[1]: Started Docker Application Container Engine.
Dec 19 17:50:46 ip-172-31-20-56 dockerd[7607]: time="2022-12-19T17:50:46.411146141Z" level=info msg="API listen on /run/docker.sock"
lines 1-22/22 (END)
```

Install Kubernetes

First verify that none of the Kubernetes components are already installed

```
root@ip-172-31-20-56:~# kubectl version --short --client
Command 'kubectl' not found, but can be installed with:
snap install kubectl
root@ip-172-31-20-56:~# kubeadm version -o short
Command 'kubeadm' not found, but can be installed with:
snap install kubeadm
root@ip-172-31-20-56:~# kubelet --version
Command 'kubelet' not found, but can be installed with:
snap install kubelet
root@ip-172-31-20-56:~#
```

Then I ran the following commands –

- `curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -`
- `echo 'deb http://apt.kubernetes.io/ kubernetes-xenial main' | sudo tee /etc/apt/sources.list.d/kubernetes.list`
- `apt-get update`
- `apt-get install -y kubelet kubeadm kubectl`

Verify the installation status

```
root@ip-172-31-20-56:~# kubectl version --short --client
Flag --short has been deprecated, and will be removed in the future. The --short output will become the default.
Client Version: v1.26.0
Kustomize Version: v4.5.7
root@ip-172-31-20-56:~# kubeadm version -o short
v1.26.0
root@ip-172-31-20-56:~# kubelet --version
Kubernetes v1.26.0
root@ip-172-31-20-56:~#
```

Kubectl, kubeadm and Kubelet have been installed successfully

Note – I was unable to use AWS for the rest of the project as our lab accounts do not have access to AKS. Hence could not create the cluster in AWS

Implement Database Policy to restrict ingress traffic to db container from frontend container only

For this I have used our Kubernetes lab setup as it doesn't require access from outside world. All the relevant files can be found here - https://github.com/SayamGanguly/devops_capstone_project_infra_optimization/tree/main/NetworkPolicy

Frontend files

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: knote
spec:
  replicas: 2
  selector:
    matchLabels:
      app: knote
      tier: frontend
  template:
    metadata:
      labels:
        app: knote
        tier: frontend
    spec:
      containers:
        - name: app
          image: learnitguide/knotejs:1.0
          ports:
            - containerPort: 3000
          env:
            - name: MONGO_URL
              value: mongodb://mongo:27017/dev
```

Deployment

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: knote
spec:
  selector:
    app: knote
    tier: frontend

  ports:
    - port: 80
      targetPort: 3000
      nodePort: 30000
  type: NodePort
```

Service

Backend Files

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongo
spec:
  replicas: 2
  selector:
    matchLabels:
      app: mongo
      tier: backend

  template:
    metadata:
      labels:
        app: mongo
        tier: backend

    spec:
      containers:
        - name: mongo
          image: mongo
          ports:
            - containerPort: 27017
```

Deployment

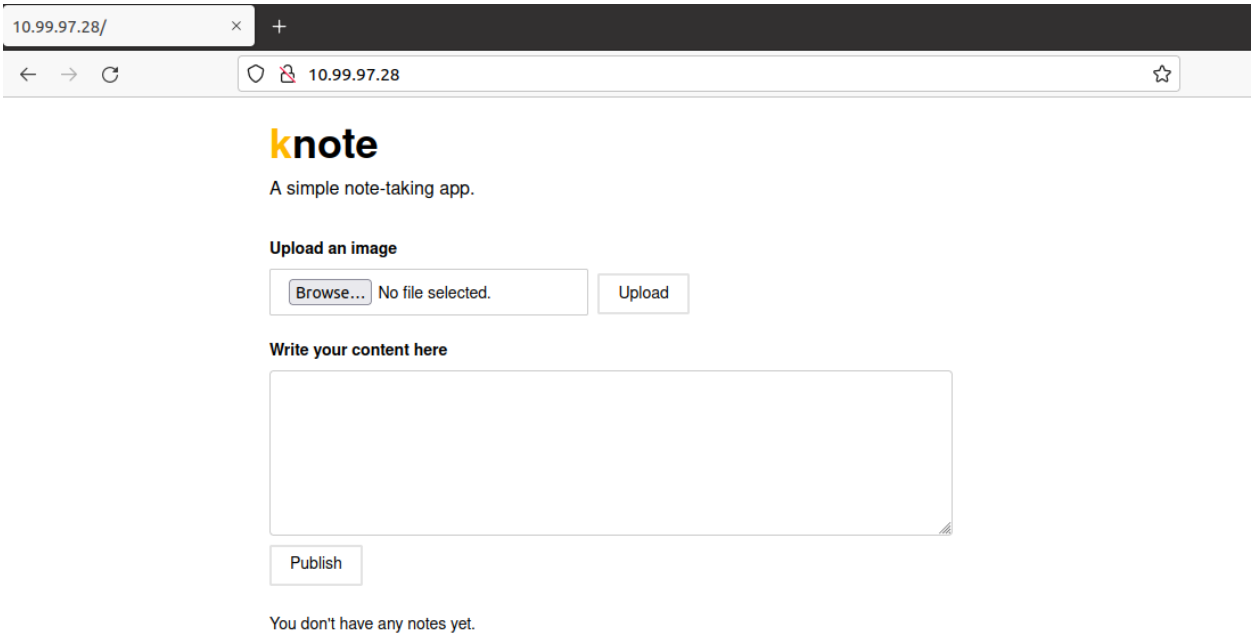
```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: mongo
spec:
  selector:
    app: mongo
    tier: backend

  ports:
    - port: 27017
      targetPort: 27017
  type: ClusterIP
```

Service

Create the Deployment and the Services

```
root@master:~/project# kubectl apply -f mongo.yaml
deployment.apps/mongo created
service/mongo created
root@master:~/project# kubectl apply -f knote.yaml
deployment.apps/knote created
service/knote created
root@master:~/project# kubectl get pods -o wide
NAME                     READY   STATUS    RESTARTS   AGE   IP                NODE      NOMINATED NODE   READINESS GATES
knote-64c9946d6b-jhx5l   1/1     Running   0          12s   192.168.235.135   worker1   <none>           <none>
knote-64c9946d6b-rxkpl   1/1     Running   0          12s   192.168.189.71    worker2   <none>           <none>
mongo-bb6b99b68-r2gbx    1/1     Running   0          18s   192.168.189.70    worker2   <none>           <none>
mongo-bb6b99b68-t4t5x    1/1     Running   0          18s   192.168.235.134   worker1   <none>           <none>
root@master:~/project# kubectl get services -o wide
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE   SELECTOR
knote        NodePort    10.99.97.28     <none>        80:30000/TCP   18s   app=knote,tier=frontend
kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP        52m   <none>
mongo        ClusterIP   10.103.47.125   <none>        27017/TCP      24s   app=mongo,tier=backend
root@master:~/project#
```

The App is accessible



Create another set of frontend pod/service with different labels

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: knote-blocked
spec:
  replicas: 2
  selector:
    matchLabels:
      app: knote-blocked
  template:
    metadata:
      labels:
        app: knote-blocked
    spec:
      containers:
        - name: app-blocked
          image: learnitguide/knotejs:1.0
          ports:
            - containerPort: 3000
          env:
            - name: MONGO_URL
              value: mongodb://mongo:27017/dev

---
```

Deployment

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: knote-blocked
spec:
  selector:
    app: knote-blocked

  ports:
    - port: 81
      targetPort: 3000
      nodePort: 30001
  type: NodePort
```

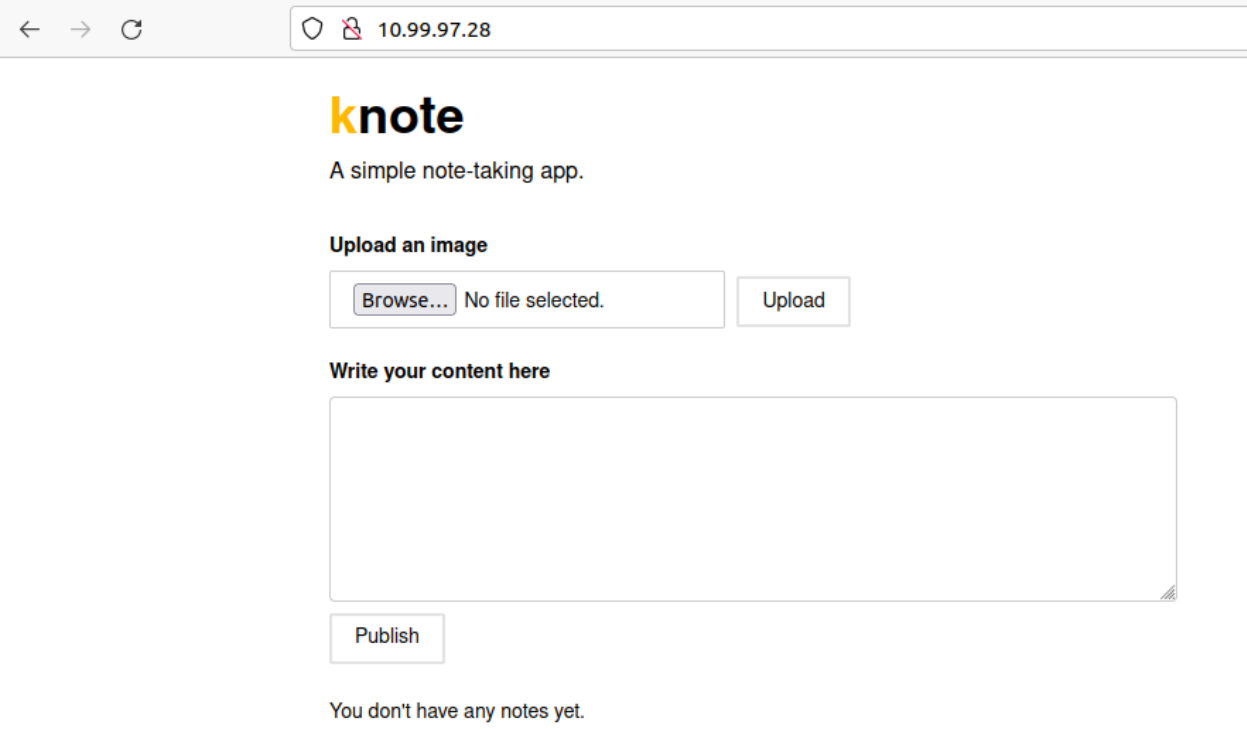
Service

Notice that the label “tier: frontend” is missing from this deployment

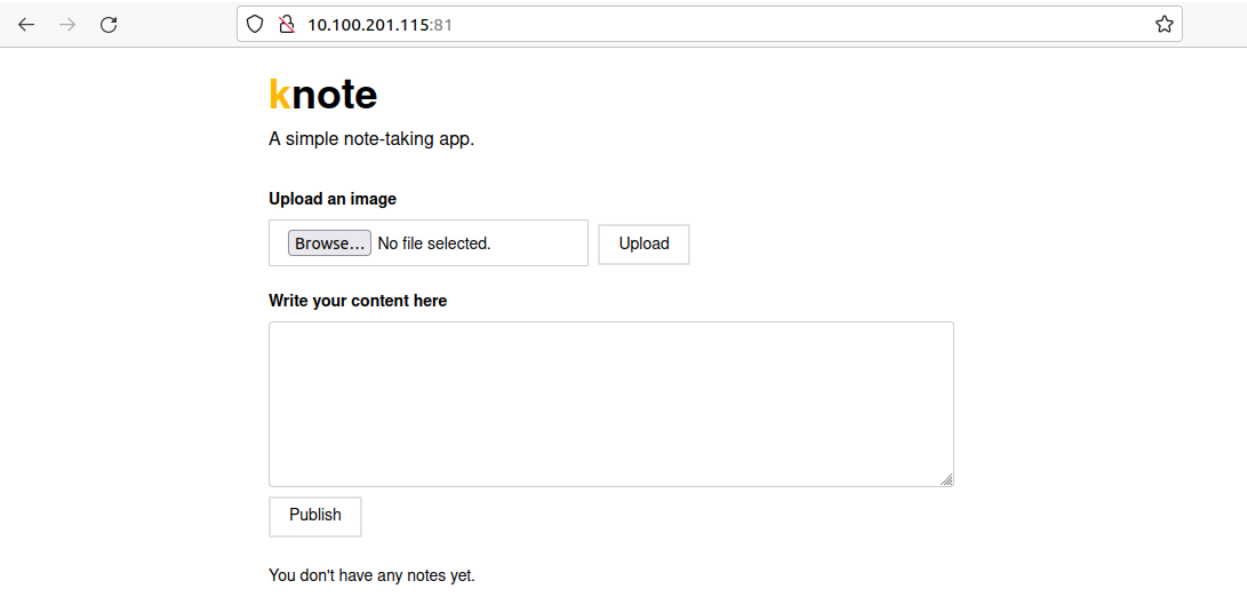
```
root@master:~/project# kubectl apply -f knote_blocked.yaml
deployment.apps/knote-blocked created
service/knote-blocked created
root@master:~/project# kubectl get services -o wide
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE     SELECTOR
knote           NodePort    10.99.97.28      <none>        80:30000/TCP   4m55s   app=knote,tier=frontend
knote-blocked   NodePort    10.100.201.115   <none>        81:30001/TCP   8s      app=knote-blocked
kubernetes      ClusterIP   10.96.0.1        <none>        443/TCP        56m     <none>
mongo           ClusterIP   10.103.47.125    <none>        27017/TCP      5m1s    app=mongo,tier=backend
root@master:~/project#
```

Verify that the app is currently accessible from both services

App with tier=frontend



App without tier=frontend



Also verify from the terminal itself

```
root@master:~/project# kubectl get services -o wide
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE     SELECTOR
knote           NodePort    10.99.97.28      <none>        80:30000/TCP   4m55s   app=knote,tier=frontend
knote-blocked   NodePort    10.100.201.115   <none>        81:30001/TCP   8s      app=knote-blocked
kubernetes      ClusterIP   10.96.0.1        <none>        443/TCP        56m     <none>
mongo           ClusterIP   10.103.47.125    <none>        27017/TCP      5m1s    app=mongo,tier=backend
root@master:~/project#
root@master:~/project#
root@master:~/project# curl 10.100.201.115:81
<html><head><title></title><link rel="stylesheet" href="tachyons.min.css"/></head><body class="ph3 pt0 pb4 mw7 center sans-serif"><h1 class="f2 mb0"><span class="gold">k</span>note</h1><p class="f5 mt1 mb4 lh-copy">
A simple note-taking app.</p><form action="/note" method="POST" enctype="multipart/form-data"><ol class="list pl0"><li class="mv3"><label class="f6 b db mb2" for="image">Upload an image</label><input class="f6 link
dim br1 ba b--black-20 ph3 pv2 mb2 dib black bg-white pointer" type="file" name="image"/><input class="f6 link dim br1 ba bw1 ph3 pv2 mb2 dib black bg-white pointer ml2" type="submit" value="Upload" name="upload"/><
/li><li class="mv3"><label class="f6 b db mb2" for="description">Write your content here</label><textarea class="f4 db border-box hover-black w-100 measure ba b--black-20 pa2 br2 mb2" rows="5" name="description"></t
extarea><input class="f6 link dim br1 ba bw1 ph3 pv2 mb2 dib black bg-white pointer" type="submit" value="Publish" name="publish"/></li></ol></form><p class="lh-copy f6">You don't have any notes yet.</p></body></htm
l>root@master:~/project#
root@master:~/project#
root@master:~/project# curl 10.99.97.28:80
<html><head><title></title><link rel="stylesheet" href="tachyons.min.css"/></head><body class="ph3 pt0 pb4 mw7 center sans-serif"><h1 class="f2 mb0"><span class="gold">k</span>note</h1><p class="f5 mt1 mb4 lh-copy">
A simple note-taking app.</p><form action="/note" method="POST" enctype="multipart/form-data"><ol class="list pl0"><li class="mv3"><label class="f6 b db mb2" for="image">Upload an image</label><input class="f6 link
dim br1 ba b--black-20 ph3 pv2 mb2 dib black bg-white pointer" type="file" name="image"/><input class="f6 link dim br1 ba bw1 ph3 pv2 mb2 dib black bg-white pointer ml2" type="submit" value="Upload" name="upload"/><
/li><li class="mv3"><label class="f6 b db mb2" for="description">Write your content here</label><textarea class="f4 db border-box hover-black w-100 measure ba b--black-20 pa2 br2 mb2" rows="5" name="description"></t
extarea><input class="f6 link dim br1 ba bw1 ph3 pv2 mb2 dib black bg-white pointer" type="submit" value="Publish" name="publish"/></li></ol></form><p class="lh-copy f6">You don't have any notes yet.</p></body></htm
l>root@master:~/project#
root@master:~/project#
```

Apply the network policy

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: mongo-netpol
spec:
  podSelector:
    matchLabels:
      tier: backend
  policyTypes:
    - Ingress
  ingress:
    - from:
        - podSelector:
            matchLabels:
              tier: frontend
~
```

```
root@master:~/project# kubectl apply -f netpol_mongo.yaml
networkpolicy.networking.k8s.io/mongo-netpol created
root@master:~/project# kubectl get netpol -o wide
NAME            POD-SELECTOR   AGE
mongo-netpol    tier=backend   8s
root@master:~/project#
```

Now retry to access the app

```
root@master:~/project# kubectl get services -o wide
NAME            TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE   SELECTOR
knote           NodePort   10.99.97.28      <none>        80:30000/TCP   42m   app=knote,tier=frontend
knote-blocked   NodePort   10.100.201.115   <none>        81:30001/TCP   37m   app=knote-blocked
kubernetes      ClusterIP  10.96.0.1        <none>        443/TCP        93m   <none>
mongo           ClusterIP  10.103.47.125    <none>        27017/TCP      42m   app=mongo,tier=backend
root@master:~/project#
root@master:~/project#
root@master:~/project# curl 10.99.97.28:80
<html><head><title></title><link rel="stylesheet" href="tachyons.min.css"/></head><body class="ph3 pt0 pb4 mw7 center sans-serif"><h1 class="f2 mb0"><span class="gold">k</span>note</h1><p class="f5 mt1 mb4 lh-copy">
A simple note-taking app.</p><form action="/note" method="POST" enctype="multipart/form-data"><ol class="list pl0"><li class="mv3"><label class="f6 b db mb2" for="image">Upload an image</label><input class="f6 link
dim br1 ba b--black-20 ph3 pv2 mb2 dib black bg-white pointer" type="file" name="image"/><input class="f6 link dim br1 ba bw1 ph3 pv2 mb2 dib black bg-white pointer ml2" type="submit" value="Upload" name="upload"/><
/li><li class="mv3"><label class="f6 b db mb2" for="description">Write your content here</label><textarea class="f4 db border-box hover-black w-100 measure ba b--black-20 pa2 br2 mb2" rows="5" name="description"></t
extarea><input class="f6 link dim br1 ba bw1 ph3 pv2 mb2 dib black bg-white pointer" type="submit" value="Publish" name="publish"/></li></ol></form><ul class="list pl0"><p class="f6 b db mb2">Notes</p><li class="mv3
 bb bw2 b--light-yellow bg-washed-yellow ph4 pv2"><p class="measure"><p> <img src="/uploads/ec7328738cd1ec14dee471ed779335af" alt=""></p>
</p></li></ul></body></html>root@master:~/project#
root@master:~/project#
root@master:~/project#
root@master:~/project# curl 10.100.201.115:81



^C
root@master:~/project#
```

The Network Policy is able to successfully block traffic to mongodb from any other apps and as a result the pods are not responding

Create a new user with permissions to create, list, get, update, and delete pods

I have used our Kubernetes lab setup for this one as well as I was not able to get root access to our Kubernetes cluster in Azure and certain steps in this process requires root level access.

We will create a user called “poduser”

Start by creating a specified folder and navigating to the folder

```
root@master:~# mkdir -p /home/certs
root@master:~# cd /home/certs/
root@master:/home/certs# ls -lrt
total 0
root@master:/home/certs#
```

Create rsa key

```
root@master:/home/certs# openssl genrsa -out poduser.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....................................+++++
.......................................................+++++
e is 65537 (0x010001)
root@master:/home/certs# less poduser.key
root@master:/home/certs# cat poduser.key
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAv2caxF1SsQmCZRlZnlKwHsLTU0pcsrHaK8hKr7CLkKwJMuD/
AvqE780jGyIA54F7ZtP/Mg1zT9tK9Eqp3PB5N0xQQCnB9TTCwgeKdSDFnwd5Va+t
doQsmoaVbM2zeQjmsel5p9jFStWmaOTj5l87mIx+lo//KCbeM7QQfBXt6ha6q7RQ
BKWaXhHzIXojsH6GiWri/AvwH4vNv8+wlAjGjZKimxVJNbZFVmPejmFgI5M0Gx8f
vMlAR+mdU8YzWITuhA2Z6OMnMAleEMmuQu6zQWH5Q4cdc3yLFvoOm+/ZSiXEzODF
cCVfVJwqrlDVoh8PjOUHpwuvlq4ueyJNOto+sQIDAQABAoIBAQCdvODoV2igkg+u
NdgPjGGs2C7Eg5IlikEct/IlHD/IuJ7UGWzQDZmk7tAxujI4mBMUyT2VCFSxyS3a
afXXKS8cjX3Ngxxfcl3W/dPzd1CmIMiz/xXbiE4iAyv+AQAZjWZh1OH3m8PsSmrc
I1ZWMsGwlIh2mLFVa8LPpWO4f3IU6b018SYGg4yefpAftePtvGi0BNTK0vYNhlM9
l8DNJhbdte503Xjd5TdNpSDY2Xp03mvZuiBaLD37SX4QwSuOTJ//Q7d1jFPw+zbh
Zwq+26WwXp/48VmS7cV4DHU91xUi7msnWiBLpG5k8eb4cL05D1xnlQDmh7lbYfZ6
4am711JJAoGBAOc8FD2Ti04Xw87tyh7kJxvLLkS9nCTBqHlFQ8bLITEOGilfSlgV
5n3N9vOK/UKvUxpqUC8pnB75NcfXSPSR/vEp4Rarh3hdBmxOZeHuGRYe/G6FzuxT
GjqzHWRHjtCGCEO5mZLf/5l2tTF5zb6ON2gtE0NaFNiTDieEL0F/CpJLAoGBANPm
7QnAILNsXsaAxGRzTmi9qSXOPF9JPUeMejtOX/xkTrIUZ6IwiJJPRShgKHs8QkvW
c9RowtyU5v/Dwu3gxZtQPZAxRcI7h4wkqqFGYq3T/QRSFJadeN1JwVHYLJQKyelc
5D/zFYFqqlAmy9lAfqitky3qnltO1yl1FgEP0zVzAoGBALTX6OCLnvxRW8Nfd58X
QnrbOfFq9U5N4uwNPsefn/LUm6NVsjgX0K79QTERt6bKpczX7qQj/8S9lNEZTyZv
dBq6bR6kwgthfZ+A5O+AnkANtdDed4meKLpV0RvXVxtAaABHjJMw+p7UFzt9jmqE
oVKpRhJ7xifnQr+nAqpPqD3nAoGAc9wKdM9Y0eGJ/G0BdI7bK0ewCVy3A75uxAUZ
+/1BXCWYJaQkebiho6W2AXp6B8+NzBtoX5YcHTFJEET/+sJwKRh5YLKgGrAjz8to
hB+Q6sxCxi7+1M8q+4tNKSvSS+uONIs2vYRymL0ye/qKcLHjzuSA2ZUzmmcEG8XK
DzCQ4eMCgYEAurRxbihfs+NIs2gTTdPCKHf8ID2TwQ5mA+ZiSbrYJ3LRST+oeO9F
FD4nm2QDJCgimZxOD9RbpvzM05jQKL/dEbyz7nNUGN34Ev99OotvESVG7ICo5kTH
SJWcjuiNkts2duWKCTBm2P6NYZy+sG+aM9RhnZGWRnvXe5w3nSgy7j8=
-----END RSA PRIVATE KEY-----
root@master:/home/certs# ls -lrt
total 4
-rw------- 1 root root 1679 Dec 13 16:20 poduser.key
root@master:/home/certs#
```

Create csr

```
root@master:/home/certs# openssl req -new -key poduser.key -out poduser.csr -subj "/CN=poduser/O=devops"
root@master:/home/certs# ls -lrt
total 8
-rw------- 1 root root 1679 Dec 13 16:20 poduser.key
-rw-r--r-- 1 root root  911 Dec 13 16:22 poduser.csr
root@master:/home/certs#
```

Generate Certificate

```
root@master:/home/certs# openssl x509 -req -in poduser.csr -CA /etc/kubernetes/pki/ca.crt -CAkey /etc/kubernetes/pki/ca.key -CAcreateserial -out poduser.crt -days 1000 ; ls -ltr
Signature ok
subject=CN = poduser, O = devops
Getting CA Private Key
total 12
-rw------- 1 root root 1679 Dec 13 16:20 poduser.key
-rw-r--r-- 1 root root  911 Dec 13 16:22 poduser.csr
-rw-r--r-- 1 root root 1013 Dec 13 16:24 poduser.crt
root@master:/home/certs#
```

Create config for poduser

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /etc/kubernetes/pki/ca.crt
    server: https://172.31.55.216:6443
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: poduser
  name: poduser@kubernetes
current-context: poduser@kubernetes
kind: Config
preferences: {}
users:
- name: poduser
  user:
    client-certificate: /home/certs/poduser.crt
    client-key: /home/certs/poduser.key

~
```

Verify that “poduser” is now authenticated to the cluster

```
root@master:/home/certs# ls -lrt
total 16
-rw------- 1 root root 1679 Dec 13 16:20 poduser.key
-rw-r--r-- 1 root root  911 Dec 13 16:22 poduser.csr
-rw-r--r-- 1 root root 1013 Dec 13 16:24 poduser.crt
-rw-r--r-- 1 root root  423 Dec 13 16:32 poduser.conf
root@master:/home/certs# kubectl --kubeconfig=/home/certs/poduser.conf version --short
Client Version: v1.23.4
Server Version: v1.23.15
root@master:/home/certs#
```

Also verify that “poduser” is still not authorized to access resources

```
root@master:/home/certs# kubectl --kubeconfig=/home/certs/poduser.conf get nodes
Error from server (Forbidden): nodes is forbidden: User "poduser" cannot list resource "nodes" in API group "" at the cluster scope
root@master:/home/certs#
```

Now create a clusterRole “podadmin” with create, list, get, update, and delete pods authorization

```
root@master:/home/certs# kubectl --kubeconfig=/etc/kubernetes/admin.conf create clusterrole podadmin --verb=create,list,get,update,delete --resource=pods
clusterrole.rbac.authorization.k8s.io/podadmin created
root@master:/home/certs# kubectl get clusterroles
NAME                                                   CREATED AT
admin                                                  2022-12-13T14:23:21Z
calico-kube-controllers                                2022-12-13T14:24:29Z
calico-node                                            2022-12-13T14:24:29Z
cluster-admin                                          2022-12-13T14:23:21Z
edit                                                   2022-12-13T14:23:21Z
kubeadm:get-nodes                                      2022-12-13T14:23:23Z
podadmin                                               2022-12-13T16:49:52Z
```

Now create clusterRoleBinding “poduserbinding” to assign the “podamin” role to “poduser”

```
root@master:/home/certs# kubectl --kubeconfig=/etc/kubernetes/admin.conf create clusterrolebinding poduserbinding --user=poduser --clusterrole=podadmin
clusterrolebinding.rbac.authorization.k8s.io/poduserbinding created
root@master:/home/certs# kubectl get clusterrolebinding
NAME                                                   ROLE                                                                   AGE
calico-kube-controllers                                ClusterRole/calico-kube-controllers                                    148m
calico-node                                            ClusterRole/calico-node                                                148m
cluster-admin                                          ClusterRole/cluster-admin                                              149m
kubeadm:get-nodes                                      ClusterRole/kubeadm:get-nodes                                          149m
kubeadm:kubelet-bootstrap                              ClusterRole/system:node-bootstrapper                                   149m
kubeadm:node-autoapprove-bootstrap                     ClusterRole/system:certificates.k8s.io:certificatesigningrequests:nodeclient   149m
kubeadm:node-autoapprove-certificate-rotation          ClusterRole/system:certificates.k8s.io:certificatesigningrequests:selfnodeclient   149m
kubeadm:node-proxier                                   ClusterRole/system:node-proxier                                        149m
poduserbinding                                         ClusterRole/podadmin                                                   15s
```

Now retry the to get pods using the “poduser”

```
root@master:/home/certs# kubectl --kubeconfig=/home/certs/poduser.conf get pods
NAME                          READY   STATUS    RESTARTS   AGE
knote-64c9946d6b-jhx5l        1/1     Running   0          98m
knote-64c9946d6b-rxkpl        1/1     Running   0          98m
knote-blocked-994c5ccd-9twpb  1/1     Running   0          93m
knote-blocked-994c5ccd-ncbnn  1/1     Running   0          93m
mongo-bb6b99b68-r2gbx         1/1     Running   0          98m
mongo-bb6b99b68-t4t5x         1/1     Running   0          98m
root@master:/home/certs#
```

Also make sure that we’re not able to get other resources using the same “poduser”

```
root@master:/home/certs# kubectl --kubeconfig=/home/certs/poduser.conf get deployments
Error from server (Forbidden): deployments.apps is forbidden: User "poduser" cannot list resource "deployments" in API group "apps" in the namespace "default"
root@master:/home/certs# kubectl --kubeconfig=/home/certs/poduser.conf get services
Error from server (Forbidden): services is forbidden: User "poduser" cannot list resource "services" in API group "" in the namespace "default"
root@master:/home/certs# kubectl --kubeconfig=/home/certs/poduser.conf get nodes
Error from server (Forbidden): nodes is forbidden: User "poduser" cannot list resource "nodes" in API group "" at the cluster scope
root@master:/home/certs# kubectl --kubeconfig=/home/certs/poduser.conf get pods
NAME                          READY   STATUS    RESTARTS   AGE
knote-64c9946d6b-jhx5l        1/1     Running   0          100m
knote-64c9946d6b-rxkpl        1/1     Running   0          100m
knote-blocked-994c5ccd-9twpb  1/1     Running   0          95m
knote-blocked-994c5ccd-ncbnn  1/1     Running   0          95m
mongo-bb6b99b68-r2gbx         1/1     Running   0          100m
mongo-bb6b99b68-t4t5x         1/1     Running   0          100m
root@master:/home/certs#
```

We can see that we’re able to get pods but not deployment, service, nodes etc

Take snapshot of ETCD database

For this one as well I have used our Kubernetes lab setup as root access is required to install “etcdctl” which I couldn’t do in Azure cluster

Install etcdctl

```
export RELEASE="3.3.13"
wget https://github.com/etcd-io/etcd/releases/download/v${RELEASE}/etcd-v${RELEASE}-linux-amd64.tar.gz
tar xvf etcd-v${RELEASE}-linux-amd64.tar.gz
cd etcd-v${RELEASE}-linux-amd64
sudo mv etcdctl /usr/local/bin
```

```
root@master:~# etcdctl --version
etcdctl version: 3.3.13
API version: 2
root@master:~#
```

Take the backup

Create a folder “etcd-backup” for back up

Get the IP address of the etcd-master running

```
root@master:~# kubectl get pod etcd-master -n kube-system -o wide
NAME          READY   STATUS    RESTARTS   AGE    IP              NODE     NOMINATED NODE   READINESS GATES
etcd-master   1/1     Running   0          172m   172.31.55.216   master   <none>           <none>
root@master:~#
```

```
root@master:~# ETCDCTL_API=3 etcdctl --endpoints=172.31.55.216:2379 --cacert /etc/kubernetes/pki/etcd/ca.crt --cert /etc/kubernetes/pki/etcd/server.crt --key /etc/kubernetes/pki/etcd/server.key snapshot save /etcd-backup/etcd-snapshot-latest.db
Snapshot saved at /etcd-backup/etcd-snapshot-latest.db
root@master:~# ls -lrt /etcd-backup/
total 4728
-rw-r--r-- 1 root root 4837408 Dec 13 17:17 etcd-snapshot-latest.db
root@master:~#
```

Here’s the command -
ETCDCTL_API=3 etcdctl --endpoints=172.31.55.216:2379 --cacert /etc/kubernetes/pki/etcd/ca.crt --cert /etc/kubernetes/pki/etcd/server.crt --key /etc/kubernetes/pki/etcd/server.key snapshot save /etcd-backup/etcd-snapshot-latest.db

Provisioning Cluster in Azure and Autoscaling

For this one I have used the Azure Cloud framework which will help in demonstrating the cluster Autoscaler feature as one of the Autoscaling options. All the relevant files can be found here - https://github.com/SayamGanguly/devops_capstone_project_infra_optimization/tree/main/AutoScaling

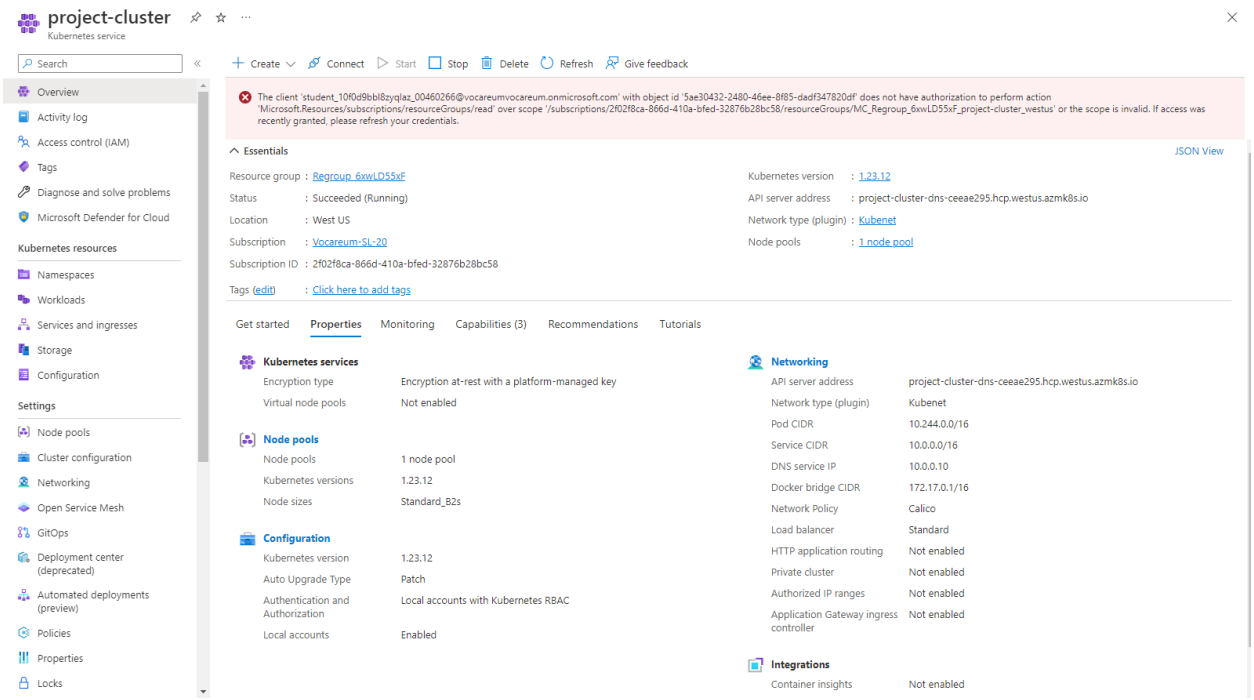
I have tried here scaling with 2 different modes -

1. Cluster Autoscaling

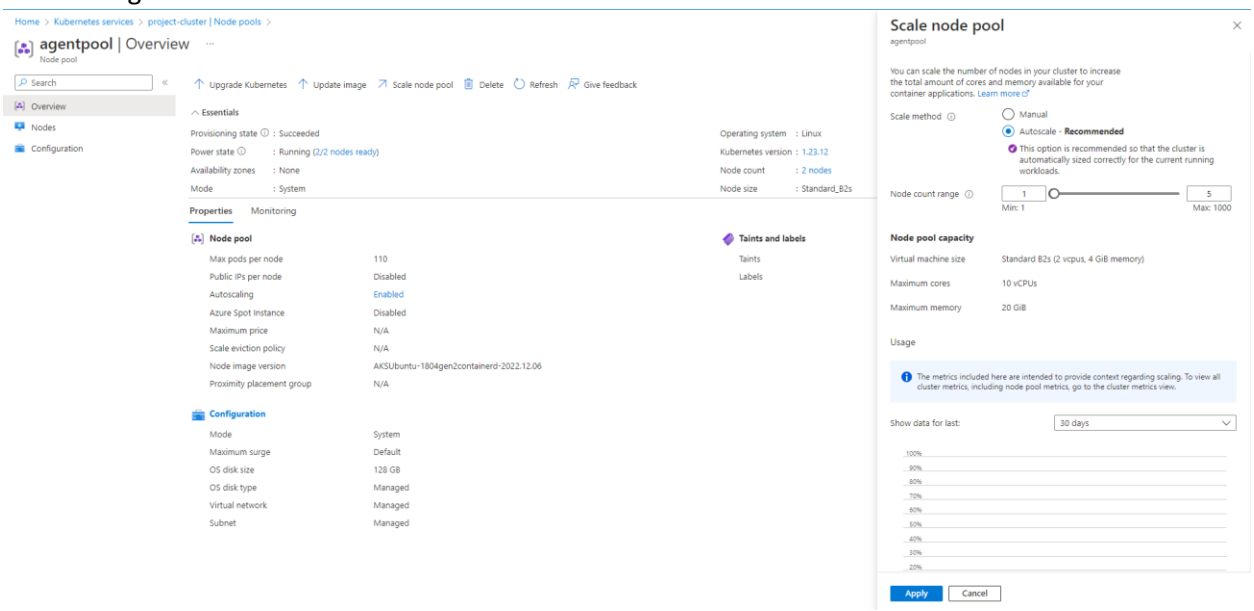
2. Horizontal Pod Autoscaler

Cluster Autoscaling

Created a cluster in the Azure lab and with autoscaling turned on (min-1, max-5)



Autoscaling have been turned on -



Initially the cluster got created with 2 nodes

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get nodes
NAME                                STATUS   ROLES   AGE   VERSION
aks-agentpool-31228956-vmss000000   Ready    agent   12m   v1.23.12
aks-agentpool-31228956-vmss000001   Ready    agent   12m   v1.23.12
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Deployed a sample app that when running simulates a load on the cpu

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get nodes
NAME                                STATUS   ROLES   AGE   VERSION
aks-agentpool-31228956-vmss000000   Ready    agent   12m   v1.23.12
aks-agentpool-31228956-vmss000001   Ready    agent   12m   v1.23.12
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods -o wide
NAME                                  READY   STATUS    RESTARTS   AGE     IP          NODE                                NOMINATED NODE   READINESS GATES
application-cpu-7749fd9d77-4llvx      1/1     Running   0          3m13s   10.244.2.3  aks-agentpool-31228956-vmss000000   <none>           <none>
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get deployments -o wide
NAME              READY   UP-TO-DATE   AVAILABLE   AGE     CONTAINERS        IMAGES                              SELECTOR
application-cpu   1/1     1            1           4m50s   application-cpu   aimvector/application-cpu:v1.0.2    app=application-cpu
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Scaled the deployment first to deploy 5 pods

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl scale deployment application-cpu --replicas 5
deployment.apps/application-cpu scaled
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Upon describing both the nodes

```
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource           Requests     Limits
  --------           --------     ------
  cpu                1810m (95%)  9 (473%)
  memory             390Mi (18%)  3512Mi (162%)
  ephemeral-storage  0 (0%)       0 (0%)
  hugepages-1Gi      0 (0%)       0 (0%)
  hugepages-2Mi      0 (0%)       0 (0%)
```

```
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource           Requests     Limits
  --------           --------     ------
  cpu                1458m (76%)  11200m (589%)
  memory             500Mi (23%)  9560Mi (443%)
  ephemeral-storage  0 (0%)       0 (0%)
  hugepages-1Gi      0 (0%)       0 (0%)
  hugepages-2Mi      0 (0%)       0 (0%)
Events:
```

Both the nodes are almost filled up

Now scale the application further to deploy 10 pods

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl scale deployment application-cpu --replicas 10
deployment.apps/application-cpu scaled
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
application-cpu-7749fd9d77-4llvx    1/1     Running   0          7m25s
application-cpu-7749fd9d77-7cpn4    0/1     Pending   0          5s
application-cpu-7749fd9d77-7g8z2    0/1     Pending   0          5s
application-cpu-7749fd9d77-b6wvp    1/1     Running   0          2m41s
application-cpu-7749fd9d77-cpbgj    1/1     Running   0          2m41s
application-cpu-7749fd9d77-jffcx    0/1     Pending   0          5s
application-cpu-7749fd9d77-ltcnx    1/1     Running   0          2m41s
application-cpu-7749fd9d77-q468f    1/1     Running   0          2m41s
application-cpu-7749fd9d77-ttxk9    0/1     Pending   0          5s
application-cpu-7749fd9d77-tzz5r    0/1     Pending   0          5s
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

I see that several pods are in “pending” state. Let’s describe one such pod -

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl describe pod application-cpu-7749fd9d77-7cpn4
Name:             application-cpu-7749fd9d77-7cpn4
Namespace:        default
Priority:         0
Service Account:  default
Node:             <none>
Labels:           app=application-cpu
                  pod-template-hash=7749fd9d77
Annotations:      <none>
Status:           Pending
IP:
IPs:              <none>
Controlled By:    ReplicaSet/application-cpu-7749fd9d77
Containers:
  application-cpu:
    Image:      aimvector/application-cpu:v1.0.2
    Port:       80/TCP
    Host Port:  0/TCP
    Limits:
      cpu:     2
      memory:  500Mi
    Requests:
      cpu:        500m
      memory:     50Mi
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-dp56d (ro)
Conditions:
  Type           Status
  PodScheduled   False
Volumes:
  kube-api-access-dp56d:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   Burstable
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/memory-pressure:NoSchedule op=Exists
                             node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason            Age   From                Message
  ----     ------            ----  ----                -------
  Warning  FailedScheduling  47s   default-scheduler   0/2 nodes are available: 2 Insufficient cpu.
  Normal   TriggeredScaleUp  32s   cluster-autoscaler  pod triggered scale-up: [{aks-agentpool-31228956-vmss 2->4 (max: 5)}]
```

Clearly, we can see that the deployment failed due to “Insufficient cpu” in both the available nodes

After waiting a few minutes let’s check the nodes again -

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get nodes
NAME                                STATUS     ROLES    AGE   VERSION
aks-agentpool-31228956-vmss000000   Ready      agent    19m   v1.23.12
aks-agentpool-31228956-vmss000001   Ready      agent    19m   v1.23.12
aks-agentpool-31228956-vmss000003   NotReady   <none>   14s   v1.23.12
aks-agentpool-31228956-vmss000004   NotReady   <none>   16s   v1.23.12
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

As we can see that 2 new nodes have come up automatically

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get nodes
NAME                                STATUS   ROLES    AGE   VERSION
aks-agentpool-31228956-vmss000000   Ready    agent    19m   v1.23.12
aks-agentpool-31228956-vmss000001   Ready    agent    19m   v1.23.12
aks-agentpool-31228956-vmss000003   Ready    <none>   33s   v1.23.12
aks-agentpool-31228956-vmss000004   Ready    <none>   35s   v1.23.12
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

The 2 new nodes are ready now

Now let's check the pods again -

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods
NAME                                    READY   STATUS    RESTARTS   AGE
application-cpu-7749fd9d77-4llvx        1/1     Running   0          9m57s
application-cpu-7749fd9d77-7cpn4        1/1     Running   0          2m37s
application-cpu-7749fd9d77-7g8z2        1/1     Running   0          2m37s
application-cpu-7749fd9d77-b6wvp        1/1     Running   0          5m13s
application-cpu-7749fd9d77-cpbgj        1/1     Running   0          5m13s
application-cpu-7749fd9d77-jffcx        1/1     Running   0          2m37s
application-cpu-7749fd9d77-ltcnx        1/1     Running   0          5m13s
application-cpu-7749fd9d77-q468f        1/1     Running   0          5m13s
application-cpu-7749fd9d77-ttxk9        1/1     Running   0          2m37s
application-cpu-7749fd9d77-tzz5r        1/1     Running   0          2m37s
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

All the pods are now up and running

If we describe the pod that was pending before again -

```
Events:
  Type    Reason             Age                  From                Message
  ----    ------             ----                 ----                -------
  Normal  TriggeredScaleUp   2m51s                cluster-autoscaler  pod triggered scale-up: [{aks-agentpool-31228956-vmss 2->4 (max: 5)}]
  Warning FailedScheduling   118s (x2 over 3m6s)  default-scheduler   0/2 nodes are available: 2 Insufficient cpu.
  Warning FailedScheduling   67s (x2 over 72s)    default-scheduler   0/4 nodes are available: 2 Insufficient cpu, 2 node(s) had taint {node.kubernetes.io/not-ready: }, that the pod didn't tolerate.
  Warning FailedScheduling   45s                  default-scheduler   0/4 nodes are available: 1 node(s) had taint {node.kubernetes.io/network-unavailable: }, that the pod didn't tolerate, 3 Insufficient cpu.
  Normal  Scheduled          34s                  default-scheduler   Successfully assigned default/application-cpu-7749fd9d77-7cpn4 to aks-agentpool-31228956-vmss000004
  Normal  Pulling            33s                  kubelet             Pulling image "aimvector/application-cpu:v1.0.2"
  Normal  Pulled             32s                  kubelet             Successfully pulled image "aimvector/application-cpu:v1.0.2" in 722.609464ms
  Normal  Created            32s                  kubelet             Created container application-cpu
  Normal  Started            32s                  kubelet             Started container application-cpu
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

It can be seen that it ultimately was picked up by one the nodes newly brought up

Now let's scale down

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl scale deployment application-cpu --replicas 2
deployment.apps/application-cpu scaled
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods
NAME                                    READY   STATUS    RESTARTS   AGE
application-cpu-7749fd9d77-b6wvp        1/1     Running   0          7m28s
application-cpu-7749fd9d77-q468f        1/1     Running   0          7m28s
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

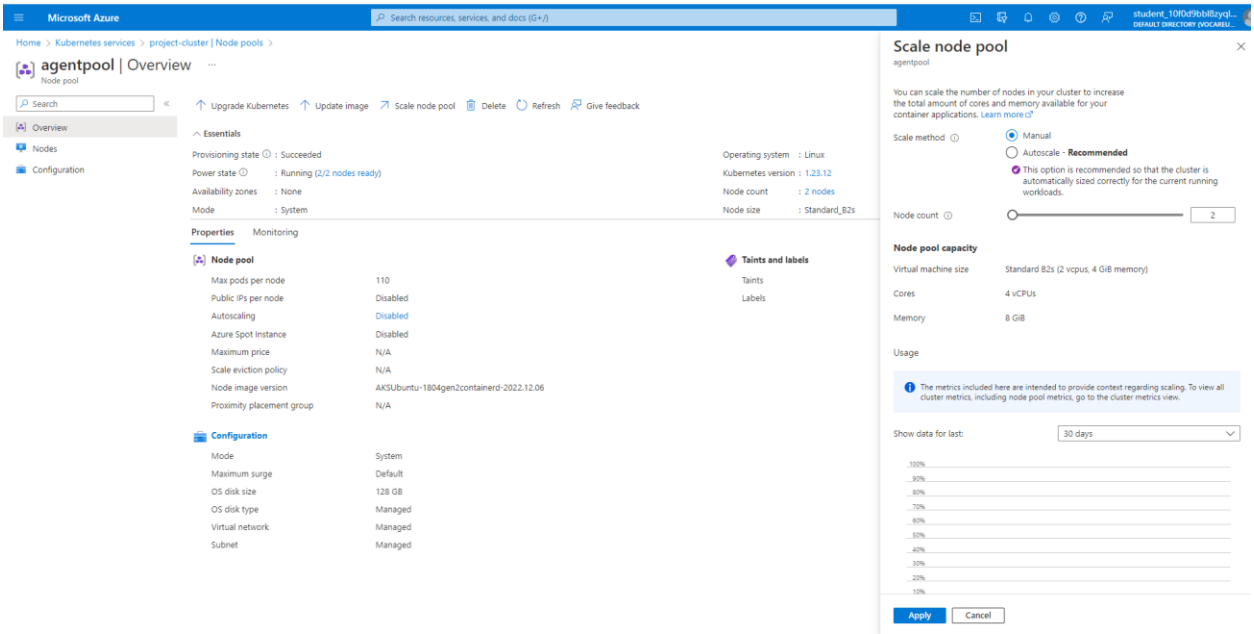
Again, after waiting a few minutes, let's check the nodes again -

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get nodes
NAME                                    STATUS   ROLES   AGE   VERSION
aks-agentpool-31228956-vmss000000       Ready    agent   33m   v1.23.12
aks-agentpool-31228956-vmss000001       Ready    agent   33m   v1.23.12
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

The nodes have also scaled down as the load has reduced on the system.

Horizontal Pod Autoscaler

At first disabled autoscaling feature of the cluster –



Currently there are 2 nodes -

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get nodes
NAME                                STATUS   ROLES   AGE     VERSION
aks-agentpool-31228956-vmss000000   Ready    agent   52m     v1.23.12
aks-agentpool-31228956-vmss000005   Ready    agent   2m28s   v1.23.12
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

And 1 pod -

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
application-cpu-7749fd9d77-wgc29    1/1     Running   0          8m6s
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Create another pod which is a simple alpine pod

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl apply -f traffic-generator.yaml
pod/traffic-generator created
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP           NODE                                NOMINATED NODE   READINESS GATES
application-cpu-7749fd9d77-wgc29    1/1     Running   0          10m   10.244.2.12  aks-agentpool-31228956-vmss000000   <none>           <none>
traffic-generator                   1/1     Running   0          11s   10.244.5.2   aks-agentpool-31228956-vmss000005   <none>           <none>
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Load at this moment in time

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl top pods
NAME                                CPU(cores)   MEMORY(bytes)
application-cpu-7749fd9d77-wgc29    1m           5Mi
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Now let's generate some load on the sample application –

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl exec -it traffic-generator sh
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl exec [POD] -- [COMMAND] instead.
/ # apk add --no-cache wrk
fetch https://dl-cdn.alpinelinux.org/alpine/v3.17/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.17/community/x86_64/APKINDEX.tar.gz
(1/3) Installing libgcc (12.2.1_git20220924-r4)
(2/3) Installing luajit (2.1_p20210510-r3)
(3/3) Installing wrk (4.1.0-r5)
Executing busybox-1.35.0-r29.trigger
OK: 9 MiB in 18 packages
/ # wrk -c 5 -t 5 -d 99999 -H "Connection: Close" http://application-cpu
Running 1667m test @ http://application-cpu
  5 threads and 5 connections
```

Let's check load on the pods -

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl top pods
NAME                                    CPU(cores)   MEMORY(bytes)
application-cpu-7749fd9d77-wgc29        1139m        8Mi
traffic-generator                       167m         3Mi
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

It is evident that the single application pod is overwhelmed

Now we deploy the Autoscaler with a specification of minimum = 1, maximum = 10 and the criteria is cpu-percent for the pods should be around 50%

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl autoscale deploy/application-cpu --cpu-percent=50 --min=1 --max=10
horizontalpodautoscaler.autoscaling/application-cpu autoscaled
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Immediately we see more pods getting created

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
application-cpu-7749fd9d77-7h816    1/1     Running   0          5s
application-cpu-7749fd9d77-bj6fv    1/1     Running   0          5s
application-cpu-7749fd9d77-gv8mg    1/1     Running   0          5s
application-cpu-7749fd9d77-wgc29    1/1     Running   0          28m
traffic-generator                   1/1     Running   0          17m
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get hpa/application-cpu  -owide
NAME              REFERENCE                    TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
application-cpu   Deployment/application-cpu   246%/50%   1         10        4          40s
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
application-cpu-7749fd9d77-7h816    1/1     Running   0          36s
application-cpu-7749fd9d77-bj6fv    1/1     Running   0          36s
application-cpu-7749fd9d77-gv8mg    1/1     Running   0          36s
application-cpu-7749fd9d77-kqgfq    0/1     Pending   0          21s
application-cpu-7749fd9d77-kxtfb    1/1     Running   0          21s
application-cpu-7749fd9d77-wgc29    1/1     Running   0          28m
traffic-generator                   1/1     Running   0          18m
```

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get hpa
NAME              REFERENCE                    TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
application-cpu   Deployment/application-cpu   67%/50%    1         10        8          6m21s
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
application-cpu-7749fd9d77-7gqgf      1/1     Running   0          5m13s
application-cpu-7749fd9d77-7h816      1/1     Running   0          6m13s
application-cpu-7749fd9d77-bj6fv      1/1     Running   0          6m13s
application-cpu-7749fd9d77-gv8mg      1/1     Running   0          6m13s
application-cpu-7749fd9d77-kqgfq      1/1     Running   0          5m58s
application-cpu-7749fd9d77-kxtfb      1/1     Running   0          5m58s
application-cpu-7749fd9d77-s24mw      1/1     Running   0          5m13s
application-cpu-7749fd9d77-wgc29      1/1     Running   0          34m
traffic-generator                     1/1     Running   0          24m
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get deployments
NAME              READY   UP-TO-DATE   AVAILABLE   AGE
application-cpu   8/8     8            8           70m
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get hpa
NAME              REFERENCE                    TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
application-cpu   Deployment/application-cpu   53%/50%    1         10        8          7m30s
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Finally, we see that the HPA is able to hit the target of around 50% with 8 pods

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl describe hpa application-cpu
Warning: autoscaling/v2beta2 HorizontalPodAutoscaler is deprecated in v1.23+, unavailable in v1.26+; use autoscaling/v2 HorizontalPodAutoscaler
Name:                                                 application-cpu
Namespace:                                            default
Labels:                                               <none>
Annotations:                                          <none>
CreationTimestamp:                                    Tue, 13 Dec 2022 19:42:29 +0000
Reference:                                            Deployment/application-cpu
Metrics:                                              ( current / target )
  resource cpu on pods  (as a percentage of request):  46% (232m) / 50%
Min replicas:                                         1
Max replicas:                                         10
Deployment pods:                                      8 current / 8 desired
Conditions:
  Type            Status  Reason              Message
  ----            ------  ------              -------
  AbleToScale     True    ReadyForNewScale    recommended size matches current size
  ScalingActive   True    ValidMetricFound    the HPA was able to successfully calculate a replica count from cpu resource utilization (percentage of request)
  ScalingLimited  False   DesiredWithinRange  the desired count is within the acceptable range
Events:
  Type    Reason             Age    From                       Message
  ----    ------             ----   ----                       -------
  Normal  SuccessfulRescale  7m48s  horizontal-pod-autoscaler  New size: 4; reason: cpu resource utilization (percentage of request) above target
  Normal  SuccessfulRescale  7m33s  horizontal-pod-autoscaler  New size: 6; reason: cpu resource utilization (percentage of request) above target
  Normal  SuccessfulRescale  6m48s  horizontal-pod-autoscaler  New size: 8; reason: cpu resource utilization (percentage of request) above target
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

The activity is evident from the describing the HPA as well

Deploying and configure a sample application

For this one as well I have used the Azure Cluster setup from previous one as I wanted to deploy the app as accessible from outside world as well. All the relevant files are here - https://github.com/SayamGanguly/devops_capstone_project_infra_optimization/tree/main/WebApp

Building the docker image from the app -

```
FROM openjdk:8-jdk-alpine
COPY target/spring-boot-docker-complete-0.0.1-SNAPSHOT.jar app.jar
ENTRYPOINT ["java","-jar","/app.jar"]
```

Pushed the docker image to my own repository –

https://hub.docker.com/repository/docker/sayamganguly/gs-spring-boot-docker

Prepare the deployment and the service

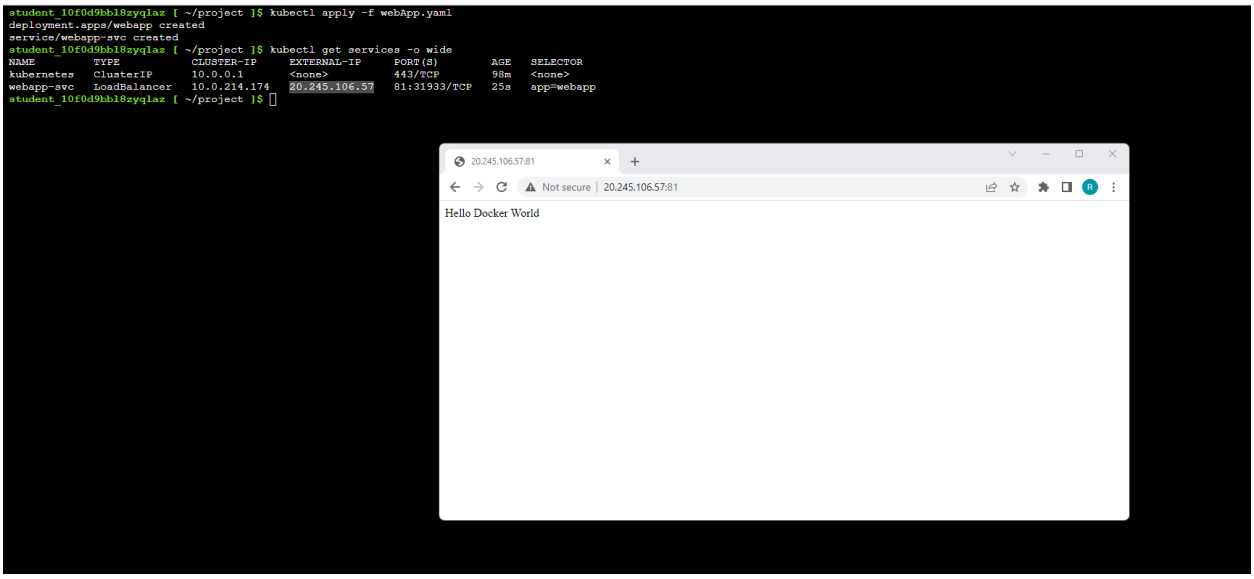
```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webapp
spec:
  replicas: 2
  selector:
    matchLabels:
      app: webapp
  template:
    metadata:
      labels:
        app: webapp
    spec:
      containers:
        - name: webapp
          image: sayamganguly/gs-spring-boot-docker:latest
          ports:
            - containerPort: 8081
```

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: webapp-svc
spec:
  selector:
    app: webapp

  ports:
    - port: 81
      targetPort: 8081
  type: LoadBalancer
```

Deploy the app and service -

```
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl apply -f webApp.yaml
deployment.apps/webapp created
service/webapp-svc created
student_10f0d9bbl8zyqlaz [ ~/project ]$ kubectl get services -o wide
NAME          TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)        AGE   SELECTOR
kubernetes    ClusterIP      10.0.0.1       <none>          443/TCP        98m   <none>
webapp-svc    LoadBalancer   10.0.214.174   20.245.106.57   81:31933/TCP   25s   app=webapp
student_10f0d9bbl8zyqlaz [ ~/project ]$
```

Verify using the LoadBalancer public IP -



The app is accessible. Hence the deployment is successful

Conclusion

In this project I was able to-

- Create an EC2 Instance and install Docker and Kubernetes on it.
- Create an Azure EKS cluster successfully and deployed a sample app on it.
- Create Network Policy that would allow db access only from certain UI apps and not from other apps.
- Create a user that only has access to pods and not to any other resources on the cluster
- Take a snapshot of the ETCD database
- Demonstrate Azure EKS's Cluster Scaling property
- Implement a Horizontal Auto Scaler which will scale the number of Pods when the cpu-utilization crosses a certain threshold