Department of Computer Science and Engineering
Cloud Architecture

Lab5B: Kubernetes using Minikube
Testing

UCID:
Student Name:
Branch:

Configure Horizontal Pod Autoscaling (HPA)
Test Case 1: Validate Load Balancing Across Pods
Test Case 2: High Load Performance
Test Case 3: Pod Failure and Recovery
Test Case 4: Service Degradation Under Node Failure
Test Case 5: Load Balancing with StatefulSets
Test Case 6: Network Latency Simulation

Conclusion:

Horizontal Pod Autoscaling (HPA) is an essential feature in Kubernetes that helps maintain scalability and reliability for applications under changing loads. It automatically adjusts the number of pods based on resource usage, making services more responsive while optimizing resources.

In this experiment, we tested HPA in various scenarios like handling high-load performance, recovering from pod failures, and dealing with service degradation caused by node failures. The results showed that HPA efficiently distributes the load across pods, manages unexpected failures smoothly, and keeps performance consistent even under tough conditions. We also tested load balancing with StatefulSets and simulated network latency, which further proved HPA's adaptability and strength in different environments.

We set up and executed the experiment using Minikube and Kubernetes, which gave a hands-on understanding of deploying and managing scalable applications. Commands like `kubectl create deployment` and `kubectl expose deployment` helped us set up the services, while the Minikube dashboard provided insights into how the system behaved during the tests.

This experiment reinforced how important it is to configure HPA for ensuring high availability and performance in cloud-native applications. Overall, it showed that HPA plays a key role in building a resilient and scalable infrastructure that can handle dynamic user demands effectively.

Department of Computer Science and Engineering
Cloud Architecture

References:

[1] https://minikube.sigs.k8s.io/docs/start/

[2] https://medium.com/cloud-native-daily/how-to-run-nginx-on-kubernetes-using-minikube-df3319b80511

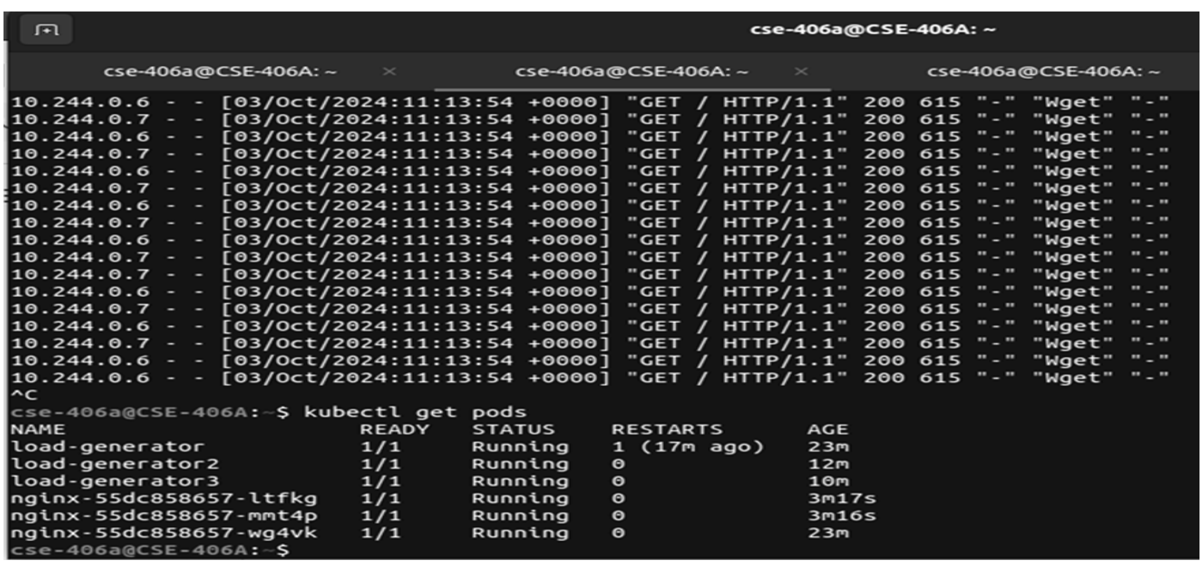
[3] https://youtu.be/s_o8dwzRlu4

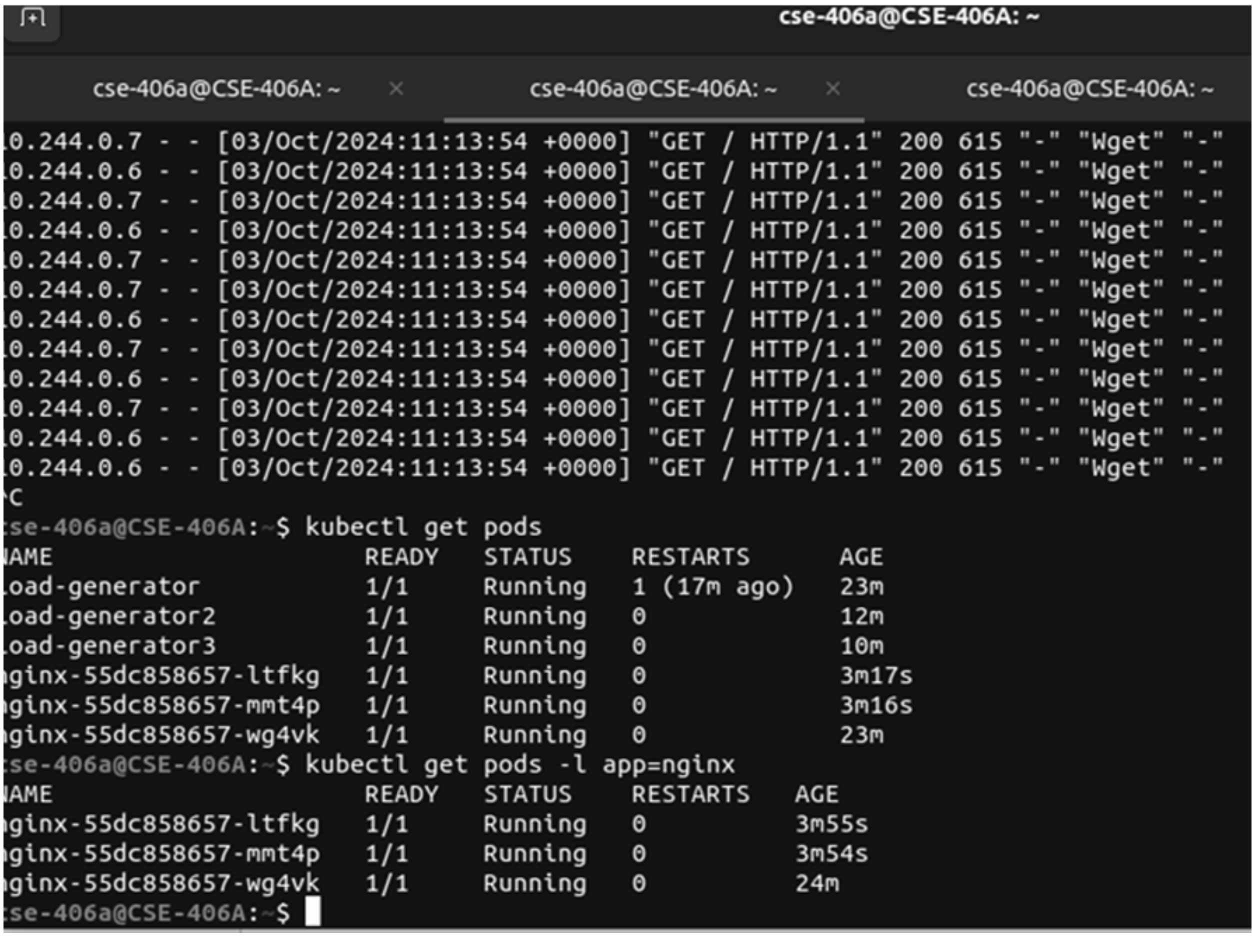
[4] https://youtu.be/E2pP1MOfo3g

List of commands on my setup: history command

```
    sudo apt update
 21  curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
 22  sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
 28  minikube start
 29  sudo chmod 777 /var/run/docker.sock
 30  minikube start
 31  kubectl get po -A
 32  docker ps
 33  docker ps -a
 34  docker ps -aq
 35  docker ps -a
 36  minikube kubectl -- get po -A
 37  docker ps -a
 38  alias kubectl="minikube kubectl --"
 39  minikube dashboard
 40  sudo minikube dashboard
 41  sudo minikube start
```

42 minikube dashboard
43 kubectl create deployment hello-minikube --image=kicbase/echo-server:1.0
44 kubectl expose deployment hello-minikube --type=NodePort --port=8080
45 kubectl get services hello-minikube
46 minikube service hello-minikube
47 kubectl port-forward service/hello-minikube 7080:8080
48 ifconfig
49 kubectl port-forward service/hello-minikube 7080:8080
50 minikube kubectl -- get pods
51 mkdir my_directory
52 cd my_directory/
53 nano service.yaml
54 nano deployment.yaml
55 kubectl create -f service.yaml
56 kubectl create -f deployment.yaml
57 Kubectl get pods
58 kubectl get pods
59 minikube service nginx-service
60 kubectl get pods
61 docker ps -a




cse-406a@CSE-406A: ~
cse-406a@CSE-406A: ~
cse-406a@CSE-406A: ~
cse-406a@CSE-406A: ~
.0.244.0.7 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.6 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.7 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.6 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.7 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.7 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.6 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.7 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.6 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.7 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.6 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
.0.244.0.6 - - [03/Oct/2024:11:13:54 +0000] "GET / HTTP/1.1" 200 615 "-" "Wget" "-"
^C
se-406a@CSE-406A:~$ kubectl get pods
IAME READY STATUS RESTARTS AGE
oad-generator 1/1 Running 1 (17m ago) 23m
oad-generator2 1/1 Running 0 12m
oad-generator3 1/1 Running 0 10m
ginx-55dc858657-ltfkg 1/1 Running 0 3m17s
ginx-55dc858657-mmt4p 1/1 Running 0 3m16s
ginx-55dc858657-wg4vk 1/1 Running 0 23m
se-406a@CSE-406A:~$ kubectl get pods -l app=nginx
IAME READY STATUS RESTARTS AGE
ginx-55dc858657-ltfkg 1/1 Running 0 3m55s
ginx-55dc858657-mmt4p 1/1 Running 0 3m54s
ginx-55dc858657-wg4vk 1/1 Running 0 24m
se-406a@CSE-406A:~$

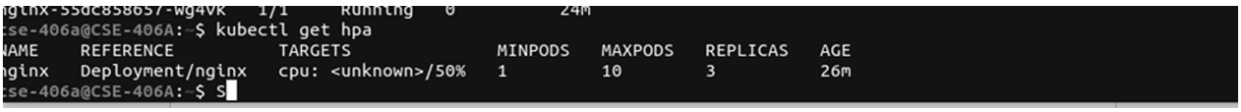
Deployment yaml

```
cse-406a@CSE-406A:~$ kubectl get deployment nginx -o yaml
apiVersion: apps/v1
kind:
Deployment
metadata:
annotations:
        deployment.kubernetes.io/revisio
 n:          "2"          creationTimestamp:
 "2024-10-03T10:54:07Z" generation: 3

 labels:

        app:
 nginx name:
 nginx
 namespace:
 default
 resourceVersion: "1603"
 uid:
44a44b04-16b2-43a2-97c1-eed68420ff02
spec:

 progressDeadlineSeconds: 600

 replicas: 3
 revisionHistoryLimit: 10
 selector:

        matchLabels:

        app:
 nginx
 strategy:
```

```
        rollingUpdate:

        maxSurge: 25%
        maxUnavailable
        : 25% type:
        RollingUpdate

template:

        metadata:
        creationTimestamp
        : null labels:
        app:
        nginx
        spec:

        containers:

        - image: nginx
        imagePullPolicy:
        Always name:
        nginx

        resources:

        limits:
        cpu:
        200m
        reque
        sts:

        cpu: 100m

        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
        dnsPolicy: ClusterFirst
        restartPolicy: Always
```

```
        schedulerName:
        default-scheduler
        securityContext: {}
        terminationGracePeriodSecon
        ds: 30

 status:

  availableRepli
  cas: 3
  conditions:
        - lastTransitionTime:
              "2024-10-03T10:54:07Z"
              lastUpdateTime:
              "2024-10-03T10:54:46Z"
        message: ReplicaSet "nginx-55dc858657" has successfully progressed.
        reason: NewReplicaSetAvailable
        status:
        "True"
        type:
        Progressing
        - lastTransitionTime:
              "2024-10-03T11:15:24Z"
              lastUpdateTime:
              "2024-10-03T11:15:24Z" message:
              Deployment has minimum availability.
              reason: MinimumReplicasAvailable
        status:
        "True"
        type:
        Available

observedGeneration: 3

  readyReplicas: 3
  replicas: 3
  updatedReplicas: 3
```

Test case 1



Test case 3