Lab 12 : Horizontal Pod Autoscaling

1. Create a Deployment named scaler-challenge.
    a. Use the image as **registry.k8s.io/hpa-example** and port **80**.
    b. Define a CPU request of 100m and a limit of 200m.
    c. Expose it via a Service named scaler-service on port **80**.

`workload.yaml`

```shell
apiVersion: v1
kind: Service
metadata:
  name: scaler-service
spec:
  type: ClusterIP
  selector:
    app: scaler-challenge
  ports:
  - port: 80
    targetPort: 80
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: scaler-challenge
spec:
  selector:
    matchLabels:
      app: scaler-challenge
  template:
    metadata:
      labels:
        app: scaler-challenge
    spec:
      containers:
      - name: scaler-challenge
        image: registry.k8s.io/hpa-example
        resources:
          requests:
            cpu: "100m"
          limits:
            cpu: "200m"
        ports:
        - containerPort: 80
```

```Shell
k apply -f workload.yaml
service/scaler-service created
deployment.apps/scaler-challenge created
```

2. Create the HPA.
    a. Metrics: Average CPU Utilization at 60%.

`hpa.yaml`

```Shell
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: scaler-challenge-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: scaler-challenge
  minReplicas: 2
  maxReplicas: 6
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 60
```

```
Shell
k apply -f hpa.yaml
horizontalpodautoscaler.autoscaling/scaler-challenge-hpa created
```

```
Shell
k get hpa
NAME                   REFERENCE                        TARGETS      MINPODS
MAXPODS   REPLICAS   AGE
scaler-challenge-hpa   Deployment/scaler-challenge   cpu: 1%/60%   2         6
2          58s
```

3. Stress the workload.
    a. Open a terminal and watch the HPA.

```
Shell
kubectl get hpa scaler-challenge-hpa -w
```

    b. Run the stress test.

```
Shell
kubectl run load-gen --image=busybox:1.28 --restart=Never -- /bin/sh -c "while
true; do wget -q -O- http://scaler-service; done"
```

4. Answer the questions below:
    a. Why did the Pod count immediately jump to 2 even before you started the load generator?
        i. **Min Replicas = 2**
    b. What was the highest number of replicas reached during the stress test?
        i. **6 replicas.**
    c. After you delete the load-gen pod, how long does it take for the replicas to scale back down to 2?
        i. **Around 5 minutes.**

Shell

```
k get deploy
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
scaler-challenge   6/6     6            6           17m
```

Shell

```
kubectl get hpa scaler-challenge-hpa -w
NAME                   REFERENCE                     TARGETS       MINPODS
MAXPODS   REPLICAS   AGE
scaler-challenge-hpa   Deployment/scaler-challenge   cpu: 1%/60%   2         6
2          11m
scaler-challenge-hpa   Deployment/scaler-challenge   cpu: 53%/60%   2         6
2          12m

scaler-challenge-hpa   Deployment/scaler-challenge   cpu: 254%/60%   2
6         2          13m
scaler-challenge-hpa   Deployment/scaler-challenge   cpu: 254%/60%   2
6         4          13m
scaler-challenge-hpa   Deployment/scaler-challenge   cpu: 254%/60%   2
6         6          13m
scaler-challenge-hpa   Deployment/scaler-challenge   cpu: 87%/60%    2
6         6          14m
```

Shell

```
k delete pod load-gen
```