Summary

- The code for Kubernetes is written in the YAML language which works on a declarative approach
- Scaling of pods
    - Vertical scaling:- Increasing physical storage such as CPU, RAM, and storage of a server to handle traffic is called vertical scaling.
        - Increasing resources → scale up
          Decreasing resources→scale down.
        - Disadvantage:- Single point of failure
    - Horizontal scaling:- Creating similar replicas of the server to handle the traffic is known as horizontal scaling
        - Creating more replicas → scale out
          Removing replicas→ scale in
- The load balancer can check the health of the server
- Load balancer works on Round Robin algorithm
- The load balancer also works as a reverse proxy between the client & server
- Scaling of a replication controller
    - From the YAML file

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: myrc1
spec:
  replicas: 1
  selector:
   dc: IN
  template:
    metadata:
      name: "mypod"
      labels:
        dc: IN
    spec:
      containers:
      - name: "myc2"
        image: "httpd"
```

- From command
  - **Command: kubectl scale --replicas=3 rc/(name or rc)**

```
C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl scale --replicas=3   rc/myrc1
replicationcontroller/myrc1 scaled

C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl  get rc
NAME    DESIRED   CURRENT   READY   AGE
myrc1   3         3         1       3m18s

C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl  get pods
NAME          READY   STATUS              RESTARTS   AGE
myrc1-8l9mj   0/1     ContainerCreating   0          6s
myrc1-8vzw9   1/1     Running             0          2m40s
myrc1-w58zs   0/1     ContainerCreating   0          6s

C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl  get pods
NAME          READY   STATUS    RESTARTS   AGE
myrc1-8l9mj   1/1     Running   0          15s
myrc1-8vzw9   1/1     Running   0          2m49s
myrc1-w58zs   1/1     Running   0          15s

C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl  get rc
NAME    DESIRED   CURRENT   READY   AGE
myrc1   3         3         3       3m33s
```

- How to expose and create service
  - **Command:- kubectl expose --type=NodePort --port=80 rc/(name of rc)**

```
C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl  get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   2d1h

C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl  expose --type=NodePort  --port=80  rc/myrc1
service/myrc1 exposed

C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl  get svc
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP        2d1h
myrc1        NodePort    10.109.90.113   <none>        80:31901/TCP   8s

C:\Users\Vimal Daga\Documents\Container2021-ws>
```

- How to check the details of the service
  - **Command: kubectl describe svc (name of svc)**
  - Kubernetes service automatically registers the pods running in the replication controller

```
C:\Users\Vimal Daga\Documents\Container2021-ws>kubectl  describe svc  myrc1
Name:                     myrc1
Namespace:                default
Labels:                   dc=IN
Annotations:              <none>
Selector:                 dc=IN
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.109.90.113
IPs:                      10.109.90.113
Port:                     <unset>  80/TCP
TargetPort:               80/TCP
NodePort:                 <unset>  31901/TCP
Endpoints:                172.17.0.2:80,172.17.0.3:80,172.17.0.5:80
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
```