==========
Kubernetes (K8S)
==========

=> It is free & open source s/w

=> Google developed this k8s

=> K8s developed using GO programming language

=> K8S provides Orchestration (Management)

=> K8S is used to manage containers
  (create/stop/start/delete/scale-up/scale-down)

==========
Advantages
==========

1) Auto Scaling : Based on demand containers count will be increased or decreased.

2) Load Balancing : Load will be distributed to all containers which are up and running.

3) Self Healing : If any container got crashed then it will be replaced with new container.

==========
Docker Vs Kubernetes
==========

Docker : Containerization platform

Note: Packaging our app code and dependencies as single unit for execution is called as Containerization.

Kubernetes : Orchestration platform

Note: Orchestation means managing the containers.

==========
Kubernetes Architecture
==========

=> K8S will follow cluster architecture.

=> Cluster means group of machines will be available.

=> In K8s Cluster we will have master node (control plane) and worker nodes

==========
K8S Cluster Components
==========

1) Control Node (Master Node)

- API Server
- Schedular
- Controller Manager
- ETCD

2) Worker Node

- Kubelet

- Kube Proxy
- Docker Runtime
- POD
- Container

=> To deploy our application using k8s we need to communicate with control node.

=> We will use KUBECTL (CLI) to communicate with control plane.

=> API Server will recieve the request given by kubectl and it will store that request in ETCD with pending status.

=> Schedular will identify pending requests available in ETCD and it will identify worker node to schedule the task.

=> Kubelet is called as Node Agent. It will maintain all the worker node information.

=> Kube Proxy will provide network for the cluster communication.

=> Controller Manager will verify all the taks are working as expected or not.

=> In Worker Node, Docker Engine will be available to run docker container.

=> In K8s, container will be created inside POD.

=> POD is a smallest building block that we can create in k8s cluster.

=> Inside POD, docker container will be created.

Note: In K8s, everything will be represented as POD only.

==========
K8S Cluster Setup
==========

1) Mini Kube => Single Node cluster => Only for practice => Only for beginners

2) Kubeadm => Multi Node Cluster => Self Managed Cluster => We are responsible for everything

3) Cloud Provider Managed Cluster => Ready Made Cluster => Provider will take care of everything

Ex : AWS EKS, Azure AKS, GCP GKE etc...

Note: Provider Managed Clusters are chargable.

EKS Setup Video : https://www.youtube.com/watch?v=is99tq4Zwsc

EKS Setup Steps: https://github.com/ashokitschool/DevOps-Documents/blob/main/05-EKS-Setup.md

==========
What is POD?
==========

=> POD is a smallest building block in the k8s cluster.

=> Application will be deployed as a pod in k8s.

=> We can create multiple pods for one application (for load balancing and High Availability)

=> To create a POD we will use YML file (Manifest YML).

=> In POD manifest YML we will configure our Docker image.

=> If POD is damaged/crashed/deleted then k8s will create new pod (self-healing).

=> If application running in multiple pods, then k8s will distribute the load to all running pods (Load Balancer).

=> PODS can be increased or decreased automatically based on the load (Scalability).

==========
K8S Services
==========

=> Service is used to expose PODS.

=> We have 3 types of services in k8s

1) Cluster IP

2) Node Port

3) Load Balancer

==========
What is Cluster IP ?
==========

=> POD is a short lived object.

=> When pod is crashed/damaged k8s will replace that with new pod.

=> When POD is re-created pod IP will be changed.

Note: It is not recommended to access pods using POD IP.

=> Cluster IP service is used to link all PODS to single ip address.

=> Cluster IP is a static ip to access pods.

=> Using Cluster IP we can access pods only with in the cluster.

Ex: DB Pods we can map to Cluster IP.

==========
What is NodePort service ?
==========

=> NodePort service is used to expose our pods outside the cluster.

=> Using NodePort we can access our application with Worker Node Public IP address.

=> When we use Node Public IP to access our pod then all requests will go same worker node (burden will be increased on the node).

Note : To distribute load to multiple worker nodes we will use LBR service.

==========
What is Load Balancer Service ?
==========

=> It is used to expose our pods outside cluster using AWS Load Balancer

=> When we access load balancer url, requests will be distributed to all pods running in all worker nodes.

Note: It works only in provider managed cluster like (EKS, AKS, GKE...)

```
================================================================================
kubernetes Manifest YML to deploy SpringBoot app and expose it using Load Balancer Service
================================================================================

---
apiVersion: apps/v1
kind: Deployment
metadata:
 name: javawebdeploy
spec:
 replicas: 10
 strategy:
  type: RollingUpdate
 selector:
  matchLabels:
   app: javawebapp
 template:
  metadata:
   name: javawebpod
   labels:
    app: javawebapp
  spec:
   containers:
   - name: javawebappcontainer
     image: ashokit/sb-logger-app
     ports:
     - containerPort: 8080
---
apiVersion: v1
kind: Service
metadata:
 name: javawebsvc
spec:
 type: LoadBalancer
 selector:
  app: javawebapp
 ports:
  - port: 80
    targetPort: 8080
...


# Execute YML to deploy
$ kubectl apply -f <yml>


##### Note: We can acces our application in browser using Load Balancer DNS URL. ###

# check pods status
$ kubectl get pods

# Check pods running in which worker node
$ kubectl get pods -o wide

# Check pod logs
$ kubectl logs <pod-name>

# Delete pod with pod name
$ kubectl delete pod <pod-name>

$ kubectl get pods
```

```
# increae pods count
$ kubectl scale deployment javawebdeploy --replicas 5

$ kubectl get pods

# decrease pods count
$ kubectl scale deployment javawebdeploy --replicas 2

$ kubectl get pods



##### Note: once practice is completed delete EKS cluster. ####

================================================================================

*EFK Stack setup in KUBERNETES EKS Cluster for Logs Monitoring* : https://www.youtube.com/watch?
v=8MLcbbfEL1U

================================================================================

1) What is Orchestration

2) Containerization vs Orchestration

3) Kubernetes Introduction

4) K8S Advantages

5) K8S Architecture

6) K8S Cluster Setup (EKS)

7) What is POD

8) What is Service (cluster ip, nodeport, load balancer)

9) Labels and Selectors

10) SpringBoot app deployment in K8S cluster

11) Centralize Log Monitoring using EFK
```