CHIRIN BANU M

22CSR035

DEVOPS

DAY4 -TASK

Kubernetes (K8s):

Kubernetes is an open source container orchestration engine for automating deployment, scaling, and management of containerized applications. The open source project is hosted by the Cloud Native Computing Foundation (CNCF).

It provides a scalable and resilient framework for automating the deployment, scaling, and management of applications across clusters of servers.

A SMALL HISTORY OF K8S:

In the early 2000s, Google started developing a system called Borg to manage their internal containerized applications.

Borg enabled Google to run applications at scale, providing features such as automatic scaling, service discovery, and fault tolerance.

In 2014, Google open-sourced a version of Borg called Kubernetes.

Kubernetes was donated to the Cloud Native Computing Foundation (CNCF), a neutral home for open-source cloud-native projects, in July 2015.

Kubernetes 1.8 added significant enhancements for storage, security, and networking. Key features included the stable release of the stateful sets API, expanded support for volume plugins, and improvements in security policies.

Check URL: https://kubernetes.io/releases/ for more release details.

Control Plane /Master Node:

The control plane's components make global decisions about the cluster (for example, scheduling), as well as detecting and responding to cluster events (for example, starting up a new pod when a deployment's replicas field is unsatisfied).

Control plane components can be run on any machine in the cluster. Do not run user containers on this machine.

Node Components / Worker Nodes

Node components run on every node, maintaining running pods and providing the Kubernetes runtime environment.

1. Master Node: The master node is responsible for managing the cluster and coordinating the overall state of the system. It includes the following components:

a. API Server: The API server is the central control point for all interactions with the cluster. It exposes the Kubernetes API and handles requests from users and other components.

b. Scheduler: The scheduler is responsible for assigning workloads (pods) to individual worker nodes based on resource requirements, constraints, and other policies.

c. Controller Manager: The controller manager runs various controllers that monitor the cluster state and drive it towards the desired state. Examples include the replication controller, node controller, and service controller.

d. etcd: etcd is a distributed key-value store used by Kubernetes to store cluster state and configuration data.

Create Deployment by executing above YAML file:

$ kubectl create -f web-deploy.yml

Do necessary modifications if exist, else create new

$ kubectl create -f web-deploy.yml

Completely Modify Pod Template

$ kubectl replace –f web-deploy.yml

3. View Deployments

$ kubectl get deployments

$ kubectl get deploy

$ kubectl get deploy -o wide

$ kubectl get deploy <deployment-name> -o json

$ kubectl get deploy <deployment-name> -o yaml

4. View Deployment Description

$ kubectl describe deploy <deployment-name>

5. We can modify generated/updated YAML file

$ kubectl edit deploy <deployment-name>

change replicas: count to any other value then (ESC):wq

We can modify our YAML file and then execute apply command

$ kubectl apply -f web-deploy.yml

We can Even scale using command also

$ kubectl scale deploy <deployment-name> --replicas=<desired-replica-count>

6. Delete Deployment

$ kubectl delete deploy <deployment-name>

$ kubectl delete -f web-deploy.yml

[2:33 PM, 3/20/2025] +91 90928 13114: 2. Create ReplicaSet by executing above YAML file

$ kubectl create -f rs-test.yml

Do necessary modifications if exist, else create new

$ kubectl apply -f rs-test.yml

Completely Modify Pod Template

$ kubectl replace –f rs-test.yml

3. View ReplicaSets

$ kubectl get replicasets

$ kubectl get rs

$ kubectl get rs –o wide

$ kubectl get rs <replica-set-name> –o json

$ kubectl get rs <replica-set-name> –o yaml

4. View ReplicaSet Description

$ kubectl describe rs <replica-set-name>

5. We can modify generated/updated YAML file

$ kubectl edit rs <replica-set-name>

change replicas: count to any other value then (ESC):wq

We can modify our YAML file and then execute apply command

$ kubectl apply -f rs-test.yml

We can Even scale using command also

$ kubectl scale replicaset <replicaset-name> --replicas=<desired-replica-count>

6. Delete ReplicaSet

$ kubectl delete rs <replica-set-name>

$ kubectl delete -f rs-test.yml

[2:38 PM, 3/20/2025] +91 90928 13114: apiVersion: apps/v1

```
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 1
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image:
        ports:
        - containerPort: 7070
---
```

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
  labels:
    app: my-service
    type: backend-app
spec:
  type: NodePort
  ports:
   - targetPort: 7070
     port: 7070
     nodePort: 30002
  selector:
    apptype: web-backend
```

[3:46 PM, 3/20/2025] +91 90928 13114: apiVersion: apps/v1

```
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 1
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
```

```
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image:
        ports:
        - containerPort: 9000

---

apiVersion: v1
kind: Service
metadata:
  name: my-service
  labels:
    app: my-service
spec:
  type: NodePort
  ports:
    - port: 9000
      targetPort: 8080
      nodePort: 30002
  selector:
    apptype: web-backend
```

```
chirinbanum@ChirinAnifa: ~
my-replicaset-66chn          1/1     Running   0          155m
my-replicaset-f4f48          1/1     Running   0          155m
my-replicaset-jdw7r          1/1     Running   0          155m
my-replicaset-tq7gs          1/1     Running   0          40m
chirinbanum@ChirinAnifa:~$ kubectl apply -f deployment.yml
deployment.apps/deployment configured
chirinbanum@ChirinAnifa:~$ kubectl get deployments
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
deployment   3/5     5            3           2m3s
chirinbanum@ChirinAnifa:~$ kubectl get pods
NAME                         READY   STATUS    RESTARTS   AGE
deployment-7678898557-68cjv  1/1     Running   0          15s
deployment-7678898557-bwxpd  1/1     Running   0          2m14s
deployment-7678898557-ngqcl  1/1     Running   0          15s
deployment-7678898557-nzss5  1/1     Running   0          2m14s
deployment-7678898557-v96j6  1/1     Running   0          2m14s
my-app                       1/1     Running   0          3h5m
my-pod                       1/1     Running   0          4h16m
my-replicaset-66chn          1/1     Running   0          155m
my-replicaset-f4f48          1/1     Running   0          155m
my-replicaset-jdw7r          1/1     Running   0          155m
my-replicaset-tq7gs          1/1     Running   0          41m
chirinbanum@ChirinAnifa:~$ kubectl scale deployment deployment --replicas=3
deployment.apps/deployment scaled
chirinbanum@ChirinAnifa:~$ kubectl get deployments
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
deployment   3/3     3            3           2m38s
chirinbanum@ChirinAnifa:~$ kubectl get pods
NAME                         READY   STATUS    RESTARTS   AGE
deployment-7678898557-bwxpd  1/1     Running   0          2m43s
deployment-7678898557-nzss5  1/1     Running   0          2m43s
deployment-7678898557-v96j6  1/1     Running   0          2m43s
my-app                       1/1     Running   0          3h6m
my-pod                       1/1     Running   0          4h16m
my-replicaset-66chn          1/1     Running   0          156m
my-replicaset-f4f48          1/1     Running   0          156m
my-replicaset-jdw7r          1/1     Running   0          156m
my-replicaset-tq7gs          1/1     Running   0          41m
chirinbanum@ChirinAnifa:~$
```

PODS COMMANDS:

Pod: The basic building block of Kubernetes. A pod represents a single instance of a running process within the cluster. It can encapsulate one or more containers that share the same network and storage resources.

[10:49 AM, 3/20/2025] +91 90928 13114: 1. Create a pod using run command

$ kubectl run <pod-name> --image=<image-name> --port=<container-port>

$ kubectl run my-pod --image=nginx --port=80

2. View all the pods

(In default namespace)

$ kubectl get pods

(In All namespace)

$ kubectl get pods -A

For a specific namespace

$ kubectl get pods -n kube-system

For a specific type

$ kubectl get pods <pod-name>

$ kubectl get pods <pod-name> -o wide

$ kubectl get pods <pod-name> -o yaml

$ kubectl get pods <pod-name> -o json

3. Describe a pod (View Pod details)

$ kubectl describe pod <pod-name>

$ kubectl describe pod my-pod

4. View Logs of a pod

$ kubectl logs <pod-name>

$ kubectl logs my-pod

[10:49 AM, 3/20/2025] +91 90928 13114: 5. Execute any command inside Pod (Inside Pod OS)

$ kubectl exec <pod-name> -- <command>

[10:58 AM, 3/20/2025] +91 90928 13114: apiVersion: v1

```
kind: Pod
metadata:
  name: my-pod
  labels:
      app: my-web-app
              type: backend
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
        - containerPort: 80
```

[11:05 AM, 3/20/2025] +91 90928 13114: apiVersion: v1

```
kind: Pod
metadata:
  name: my-app
```

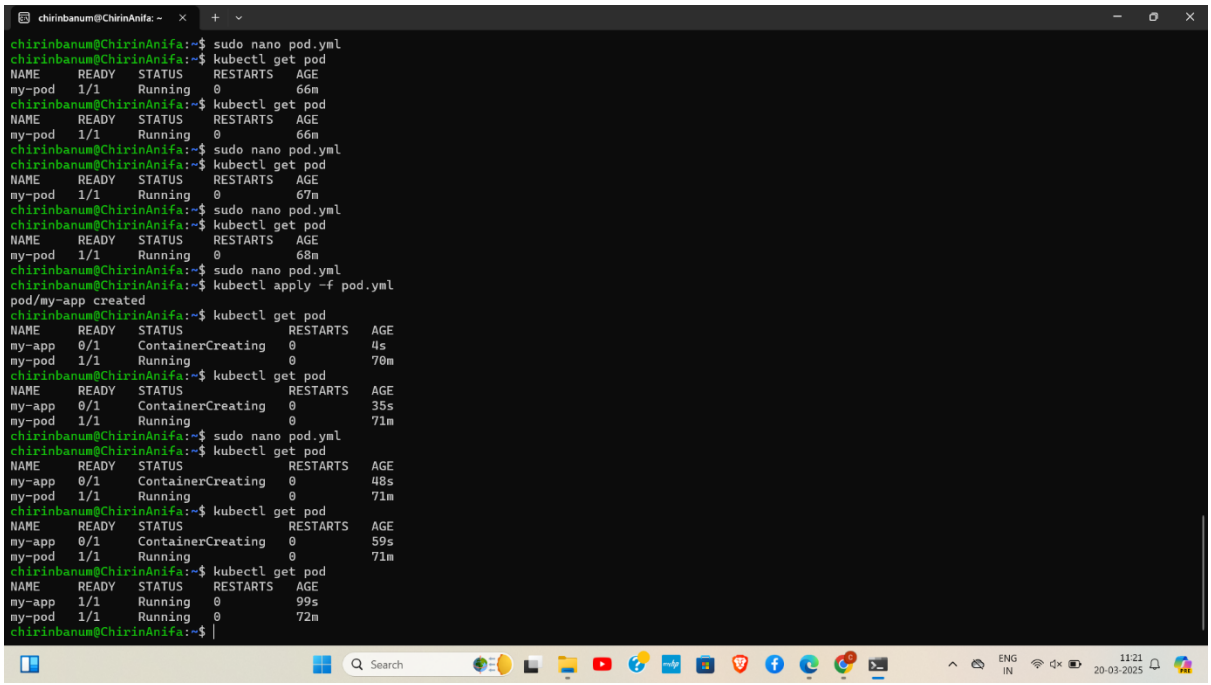
spec:

containers:

- name: my-app-container

image: <images> ports:

- containerPort: 9090



REPLICASET:

kind: ReplicaSet

metadata:

name: my-rs

labels:

name: my-rs

spec:

replicas: 4

selector:

matchLabels:

apptype: web-backend

template:

metadata:

labels:

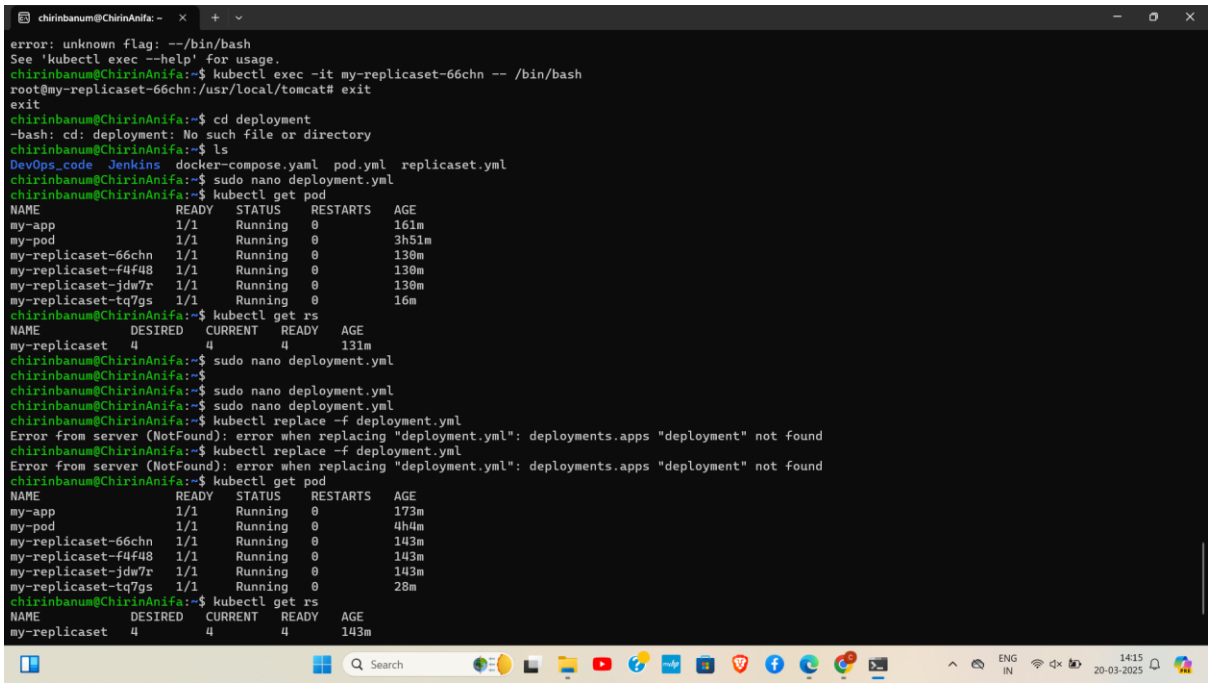
```
      apptype: web-backend
    spec:
      containers:
      - name: my-app
        image:
        ports:
          - containerPort: 8080
```

[1:56 PM, 3/20/2025] +91 90928 13114: apiVersion: apps/v1

```
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 4
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image:
        ports:
            - containerPort: 7070
```

[2:03 PM, 3/20/2025] +91 90928 13114: kubectl create deployment webnginx2 --image=nginx:latest --replicas=1

[2:12 PM, 3/20/2025] +91 90928 13114: kubectl scale deploy <deployment-name> --replicas=<desired-replica-count>



```
chirinbanum@ChirinAnifa: ~
error: unknown flag: --/bin/bash
See 'kubectl exec --help' for usage.
chirinbanum@ChirinAnifa:~$ kubectl exec -it my-replicaset-66chn -- /bin/bash
root@my-replicaset-66chn:/usr/local/tomcat# exit
exit
chirinbanum@ChirinAnifa:~$ cd deployment
-bash: cd: deployment: No such file or directory
chirinbanum@ChirinAnifa:~$ ls
DevOps_code  Jenkins  docker-compose.yaml  pod.yml  replicaset.yml
chirinbanum@ChirinAnifa:~$ sudo nano deployment.yml
chirinbanum@ChirinAnifa:~$ kubectl get pod
NAME                   READY   STATUS    RESTARTS   AGE
my-app                 1/1     Running   0          161m
my-pod                 1/1     Running   0          3h51m
my-replicaset-66chn    1/1     Running   0          130m
my-replicaset-f4f48    1/1     Running   0          130m
my-replicaset-jdw7r    1/1     Running   0          130m
my-replicaset-tq7gs    1/1     Running   0          16m
chirinbanum@ChirinAnifa:~$ kubectl get rs
NAME            DESIRED   CURRENT   READY   AGE
my-replicaset   4         4         4       131m
chirinbanum@ChirinAnifa:~$ sudo nano deployment.yml
chirinbanum@ChirinAnifa:~$
chirinbanum@ChirinAnifa:~$ sudo nano deployment.yml
chirinbanum@ChirinAnifa:~$ sudo nano deployment.yml
chirinbanum@ChirinAnifa:~$ kubectl replace -f deployment.yml
Error from server (NotFound): error when replacing "deployment.yml": deployments.apps "deployment" not found
chirinbanum@ChirinAnifa:~$ kubectl replace -f deployment.yml
Error from server (NotFound): error when replacing "deployment.yml": deployments.apps "deployment" not found
chirinbanum@ChirinAnifa:~$ kubectl get pod
NAME                   READY   STATUS    RESTARTS   AGE
my-app                 1/1     Running   0          173m
my-pod                 1/1     Running   0          4h4m
my-replicaset-66chn    1/1     Running   0          143m
my-replicaset-f4f48    1/1     Running   0          143m
my-replicaset-jdw7r    1/1     Running   0          143m
my-replicaset-tq7gs    1/1     Running   0          28m
chirinbanum@ChirinAnifa:~$ kubectl get rs
NAME            DESIRED   CURRENT   READY   AGE
my-replicaset   4         4         4       143m
```

SERVICE NOTES:

Services (short name = svc):

Service is an abstraction that defines a logical set of pods and a policy to access them. Services enable network connectivity and load balancing to the pods that are part of the service, allowing other components within or outside the cluster to interact with the application.

Service Types: Kubernetes supports different types of services:

1. NodePort: Exposes the service on a static port on each selected node's IP. This type makes the service accessible from outside the cluster by the <NodeIP>:<NodePort> combination.

2. ClusterIP: Exposes the service on a cluster-internal IP. This type makes the service only reachable within the cluster.

3. LoadBalancer: Creates an external load balancer in cloud environments, which routes traffic to the service.

```
chirinbanum@ChirinAnifa: ~
20-Mar-2025 05:49:35.976 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.base=/usr/local/tomcat
20-Mar-2025 05:49:35.976 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dcatalina.home=/usr/local/tomcat
20-Mar-2025 05:49:35.976 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.io.tmpdir=/usr/local/tomcat/temp
20-Mar-2025 05:49:35.985 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent Loaded Apache Tomcat Native library [1.3.1] using APR vers
ion [1.7.2].
20-Mar-2025 05:49:35.985 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent APR capabilities: IPv6 [true], sendfile [true], accept fil
ters [false], random [true], UDS [true].
20-Mar-2025 05:49:35.986 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent APR/OpenSSL configuration: useAprConnector [false], useOpe
nSSL [true]
20-Mar-2025 05:49:35.991 INFO [main] org.apache.catalina.core.AprLifecycleListener.initializeSSL OpenSSL successfully initialized [OpenSSL 3.0.13 30 Jan 202
4]
20-Mar-2025 05:49:36.917 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["http-nio-8080"]
20-Mar-2025 05:49:37.015 INFO [main] org.apache.catalina.startup.Catalina.load Server initialization in [1327] milliseconds
20-Mar-2025 05:49:37.181 INFO [main] org.apache.catalina.core.StandardService.startInternal Starting service [Catalina]
20-Mar-2025 05:49:37.181 INFO [main] org.apache.catalina.core.StandardEngine.startInternal Starting Servlet engine: [Apache Tomcat/9.0.102]
20-Mar-2025 05:49:37.196 INFO [main] org.apache.catalina.startup.HostConfig.deployWAR Deploying web application archive [/usr/local/tomcat/webapps/maven-web
-app.war]
20-Mar-2025 05:49:37.808 INFO [main] org.apache.catalina.startup.HostConfig.deployWAR Deployment of web application archive [/usr/local/tomcat/webapps/maven
-web-app.war] has finished in [611] ms
20-Mar-2025 05:49:37.811 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-nio-8080"]
20-Mar-2025 05:49:37.825 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in [752] milliseconds
chirinbanum@ChirinAnifa:~$ curl http://192.168.49.2:31130/maven-web-app
chirinbanum@ChirinAnifa:~$ curl http://192.168.49.2:31130/maven-web-app/
<html>
<body>
<h2>Hello World!</h2>
</body>
</html>
chirinbanum@ChirinAnifa:~$
```

```
chirinbanum@ChirinAnifa: ~
🎉  Opening service default/my-service in default browser...
👉  http://127.0.0.1:45977
❗  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
cd /path/to/your/file
^C✋  Stopping tunnel for service my-service.
chirinbanum@ChirinAnifa:~$ cd /path/to/your/file
-bash: cd: /path/to/your/file: No such file or directory
chirinbanum@ChirinAnifa:~$ kubectl apply -f service.yml
service/my-service unchanged
chirinbanum@ChirinAnifa:~$ kubectl apply -f service.yml -n myns
Error from server (NotFound): error when creating "service.yml": namespaces "myns" not found
chirinbanum@ChirinAnifa:~$ nano service.yml


chirinbanum@ChirinAnifa:~$ minikube service my-service
|-----------|------------|-------------|---------------------------|
| NAMESPACE |    NAME    | TARGET PORT |            URL            |
|-----------|------------|-------------|---------------------------|
| default   | my-service |          80 | http://192.168.49.2:31130 |
|-----------|------------|-------------|---------------------------|
🏃  Starting tunnel for service my-service.
|-----------|------------|-------------|------------------------|
| NAMESPACE |    NAME    | TARGET PORT |          URL           |
|-----------|------------|-------------|------------------------|
| default   | my-service |             | http://127.0.0.1:44079 |
|-----------|------------|-------------|------------------------|
🎉  Opening service default/my-service in default browser...
👉  http://127.0.0.1:44079
❗  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
^C✋  Stopping tunnel for service my-service.
chirinbanum@ChirinAnifa:~$ curl http://192.168.49.2:31130
<!doctype html><html lang="en"><head><title>HTTP Status 404 – Not Found</title><style type="text/css">body {font-family:Tahoma,Arial,sans-serif;} h1, h2, h3
, b {color:white;background-color:#525D76;} h1 {font-size:22px;} h2 {font-size:16px;} h3 {font-size:14px;} p {font-size:12px;} a {color:black;} .line {heigh
t:1px;background-color:#525D76;border:none;}</style></head><body><h1>HTTP Status 404 – Not Found</h1><hr class="line" /><p><b>Type</b> Status Report</p><p><
b>Description</b> The origin server did not find a current representation for the target resource or is not willing to disclose that one exists.</p><hr clas
chirinbanum@ChirinAnifa:~$
```

```
chirinbanum@ChirinAnifa: ~
chirinbanum@ChirinAnifa:~$ kubectl create ns mydeploy
namespace/mydeploy created
chirinbanum@ChirinAnifa:~$  kubectl apply -f deploy.yml -n mydeploy
error: the path "deploy.yml" does not exist
chirinbanum@ChirinAnifa:~$  kubectl apply -f deploy.yml -n mydeploy
error: the path "deploy.yml" does not exist
chirinbanum@ChirinAnifa:~$  kubectl apply -f deployment.yml -n mydeploy
deployment.apps/deployment created
chirinbanum@ChirinAnifa:~$ kubectl get pod
NAME                          READY   STATUS    RESTARTS   AGE
deployment-7678898557-b2brr   1/1     Running   0          26m
deployment-7678898557-bwxpd   1/1     Running   0          135m
deployment-7678898557-cw957   1/1     Running   0          26m
deployment-7678898557-nzss5   1/1     Running   0          135m
deployment-7678898557-v96j6   1/1     Running   0          135m
my-app                        1/1     Running   0          5h19m
my-pod                        1/1     Running   0          6h30m
my-replicaset-66chn           1/1     Running   0          4h49m
my-replicaset-f4f48           1/1     Running   0          4h49m
my-replicaset-jdw7r           1/1     Running   0          4h49m
my-replicaset-tq7gs           1/1     Running   0          174m
chirinbanum@ChirinAnifa:~$ kubectl get deploy
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
deployment   5/5     5            5           135m
chirinbanum@ChirinAnifa:~$ kubectl get pod -n deploy
No resources found in deploy namespace.
chirinbanum@ChirinAnifa:~$ kubectl get pod -n mydeploy
NAME                          READY   STATUS    RESTARTS   AGE
deployment-7678898557-2f592   1/1     Running   0          114s
deployment-7678898557-bxf2r   1/1     Running   0          114s
deployment-7678898557-h6tg7   1/1     Running   0          114s
deployment-7678898557-lb4cb   1/1     Running   0          114s
deployment-7678898557-wpq8p   1/1     Running   0          114s
chirinbanum@ChirinAnifa:~$ kubectl get pod
NAME                          READY   STATUS    RESTARTS   AGE
deployment-7678898557-b2brr   1/1     Running   0          27m
deployment-7678898557-bwxpd   1/1     Running   0          137m
deployment-7678898557-cw957   1/1     Running   0          27m
deployment-7678898557-nzss5   1/1     Running   0          137m
deployment-7678898557-v96j6   1/1     Running   0          137m
```

NAMESPACE NOTES:

Namespace (short name = ns):

namespace is a virtual cluster or logical partition within a cluster that provides a way to organize and isolate resources. It allows multiple teams or projects to share the same physical cluster while maintaining resource separation and access control.

[4:34 PM, 3/20/2025] +91 90928 13114: # To create a namespace:

$ kubectl create namespace <namespace-name>

$ kubectl create ns my-bank

To switch to a specific namespace: (make this as default type)

$ kubectl config set-context --current --namespace=<namespace-name>

To list all namespaces:

$ kubectl get namespaces

To get resources within a specific namespace:

$ kubectl get <resource-type> -n <namespace-name>

$ kubectl get deploy -n my-bank

$ kubectl get deploy --namespace my-bank

$ kubectl get all --namespace my-bank

To delete a namespace and all associated resources:

$ kubectl delete namespace <namespace-name>

$ kubectl delete ns my-bank

[4:34 PM, 3/20/2025] +91 90928 13114: kubectl create ns mydeploy

[4:42 PM, 3/20/2025] +91 90928 13114: kubectl apply -f deploy.yml -n mydeploy

[4:42 PM, 3/20/2025] +91 90928 13114: apiVersion: v1

kind: Namespace

metadata:

  name: my-demo-ns

apiVersion: v1

kind: Pod

metadata:

  name: my-pod

  namespace: my-demo-ns

spec:

  containers:

  - name: my-container

    image: nginx:latest