Day 4 Task: Kubernetes

- *Learned Kubernetes architecture*
- *Deployed applications using Kubernetes*
- *Worked with Kubernetes Services and Namespaces*

Kubernetes Control Plane

*The **Kubernetes Control Plane** manages the state of the cluster, making decisions about scheduling and responding to cluster events. It includes key components:*

Components of Control Plane

1. ***Kube-API Server** – Handles user requests, authenticates them, and updates the cluster state.*
2. ***Kube-Scheduler** – Decides where to place new pods based on resource availability.*
3. ***Controller Manager** – Ensures the cluster matches the desired state by managing controllers.*
4. ***Etcd Database** – Stores all cluster data and configuration for reliability.*

Workflow of Kubernetes Control Plane

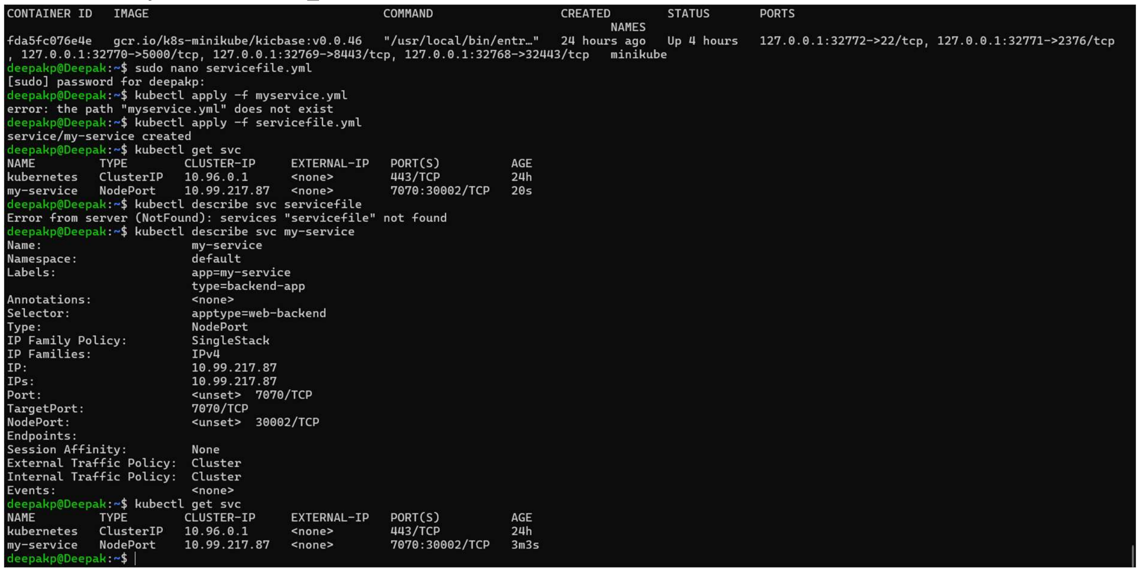
- ***User Interaction**: Users interact via kubectl or APIs to deploy and manage workloads.*
- ***API Server Processing**: Authenticates and validates requests before updating etcd.*
- ***Etcd Update**: Stores the cluster's state and ensures consistency.*

- ***Controller Manager Actions****: Detects deviations and corrects them, managing pod replicas, nodes, and endpoints.*
- ***Scheduler Decisions****: Assigns pods to nodes based on available resources.*

Key Kubernetes Concepts

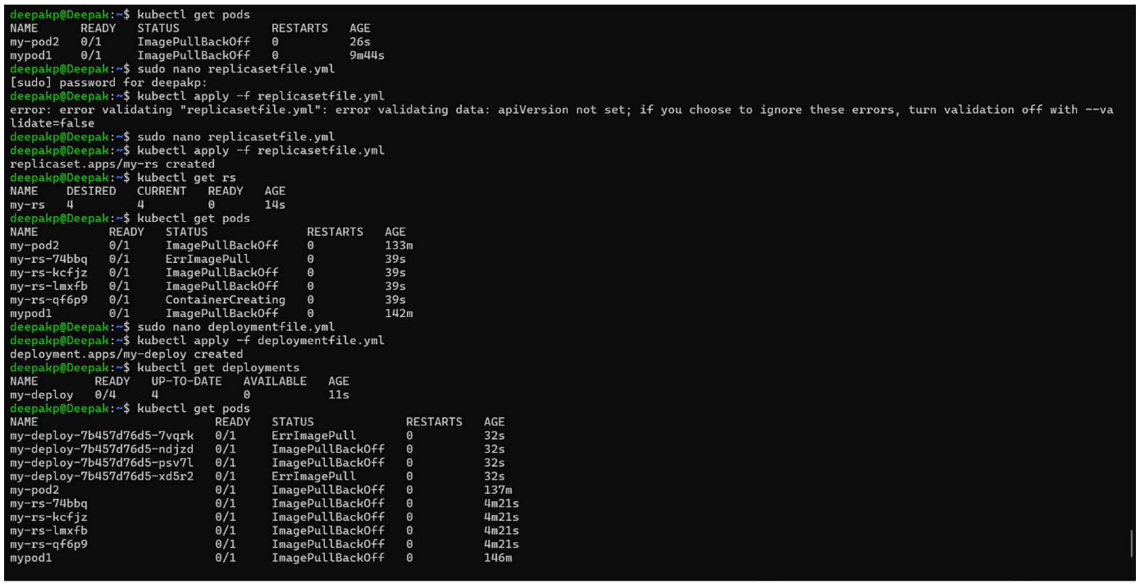
- ***Pods****: The smallest unit in Kubernetes, typically running a single container.*
- ***ReplicaSets****: Ensures a specified number of pod replicas are running.*
- ***Service****: Provides network access to a set of pods.*
- ***Deployment****: Manages ReplicaSets, allows updates and rollbacks for seamless application deployment.*

Service yaml output



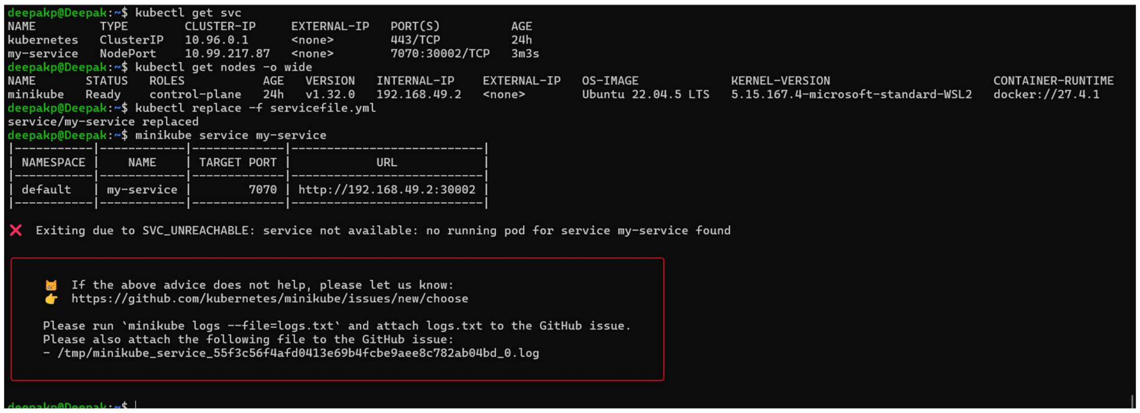
```
CONTAINER ID   IMAGE                                 COMMAND                  CREATED        STATUS        PORTS
                                                                                   NAMES
fda5fc076e4e   gcr.io/k8s-minikube/kicbase:v0.0.46   "/usr/local/bin/entr…"   24 hours ago   Up 4 hours    127.0.0.1:32772->22/tcp, 127.0.0.1:32771->2376/tcp
, 127.0.0.1:32770->5000/tcp, 127.0.0.1:32769->8443/tcp, 127.0.0.1:32768->32443/tcp   minikube
deepakp@Deepak:~$ sudo nano servicefile.yml
[sudo] password for deepakp:
deepakp@Deepak:~$ kubectl apply -f myservice.yml
error: the path "myservice.yml" does not exist
deepakp@Deepak:~$ kubectl apply -f servicefile.yml
service/my-service created
deepakp@Deepak:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP          24h
my-service   NodePort    10.99.217.87   <none>        7070:30002/TCP   20s
deepakp@Deepak:~$ kubectl describe svc servicefile
Error from server (NotFound): services "servicefile" not found
deepakp@Deepak:~$ kubectl describe svc my-service
Name:                     my-service
Namespace:                default
Labels:                   app=my-service
                          type=backend-app
Annotations:              <none>
Selector:                 apptype=web-backend
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.99.217.87
IPs:                      10.99.217.87
Port:                     <unset>  7070/TCP
TargetPort:               7070/TCP
NodePort:                 <unset>  30002/TCP
Endpoints:
Session Affinity:         None
External Traffic Policy:  Cluster
Internal Traffic Policy:  Cluster
Events:                   <none>
deepakp@Deepak:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP          24h
my-service   NodePort    10.99.217.87   <none>        7070:30002/TCP   3m3s
deepakp@Deepak:~$ |
```

Pods output



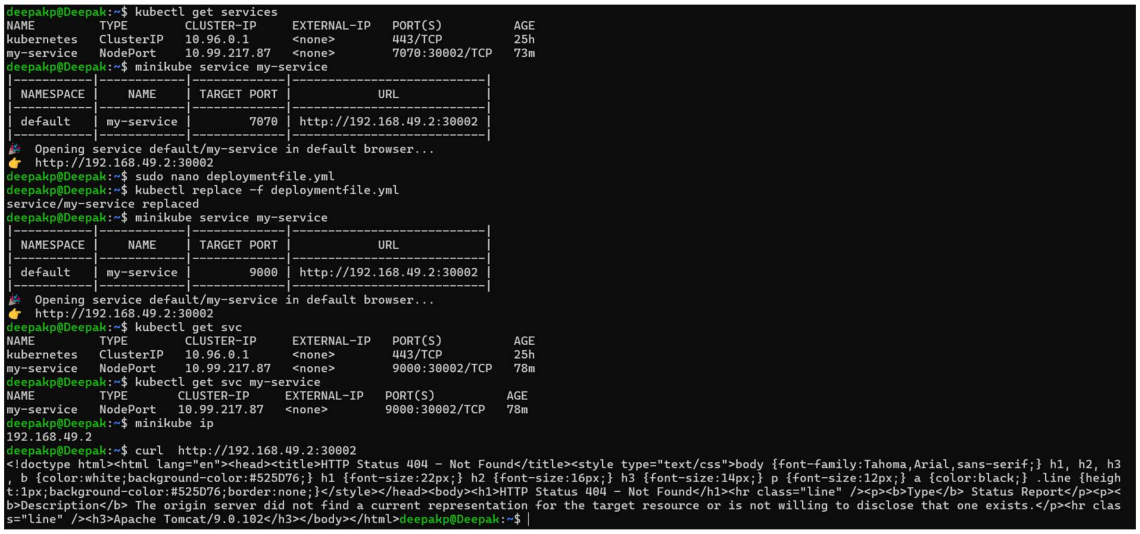
```
deepakp@Deepak:~$ kubectl get pods
NAME      READY   STATUS             RESTARTS   AGE
my-pod2   0/1     ImagePullBackOff   0          26s
mypod1    0/1     ImagePullBackOff   0          9m44s
deepakp@Deepak:~$ sudo nano replicasetfile.yml
[sudo] password for deepakp:
deepakp@Deepak:~$ kubectl apply -f replicasetfile.yml
error: error validating "replicasetfile.yml": error validating data: apiVersion not set; if you choose to ignore these errors, turn validation off with --va
lidate=false
deepakp@Deepak:~$ sudo nano replicasetfile.yml
deepakp@Deepak:~$ kubectl apply -f replicasetfile.yml
replicaset.apps/my-rs created
deepakp@Deepak:~$ kubectl get rs
NAME    DESIRED   CURRENT   READY   AGE
my-rs   4         4         0       14s
deepakp@Deepak:~$ kubectl get pods
NAME          READY   STATUS              RESTARTS   AGE
my-pod2       0/1     ImagePullBackOff    0          133m
my-rs-74bbq   0/1     ErrImagePull        0          39s
my-rs-kcfjz   0/1     ImagePullBackOff    0          39s
my-rs-lmxfb   0/1     ImagePullBackOff    0          39s
my-rs-qf6p9   0/1     ContainerCreating   0          39s
mypod1        0/1     ImagePullBackOff    0          142m
deepakp@Deepak:~$ sudo nano deploymentfile.yml
deepakp@Deepak:~$ kubectl apply -f deploymentfile.yml
deployment.apps/my-deploy created
deepakp@Deepak:~$ kubectl get deployments
NAME        READY   UP-TO-DATE   AVAILABLE   AGE
my-deploy   0/4     4            0           11s
deepakp@Deepak:~$ kubectl get pods
NAME                         READY   STATUS             RESTARTS   AGE
my-deploy-7b457d76d5-7vqrk   0/1     ErrImagePull       0          32s
my-deploy-7b457d76d5-ndjzd   0/1     ImagePullBackOff   0          32s
my-deploy-7b457d76d5-psv7l   0/1     ImagePullBackOff   0          32s
my-deploy-7b457d76d5-xd5r2   0/1     ErrImagePull       0          32s
my-pod2                      0/1     ImagePullBackOff   0          137m
my-rs-74bbq                  0/1     ImagePullBackOff   0          4m21s
my-rs-kcfjz                  0/1     ImagePullBackOff   0          4m21s
my-rs-lmxfb                  0/1     ImagePullBackOff   0          4m21s
my-rs-qf6p9                  0/1     ImagePullBackOff   0          4m21s
mypod1                       0/1     ImagePullBackOff   0          146m
```

Minikube service output



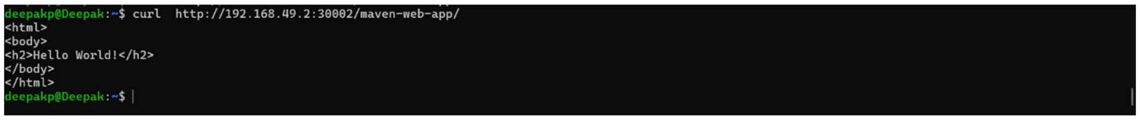
```
deepakp@Deepak:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP          24h
my-service   NodePort    10.99.217.87   <none>        7070:30002/TCP   3m3s
deepakp@Deepak:~$ kubectl get nodes -o wide
NAME       STATUS   ROLES           AGE   VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION                       CONTAINER-RUNTIME
minikube   Ready    control-plane   24h   v1.32.0   192.168.49.2   <none>        Ubuntu 22.04.5 LTS   5.15.167.4-microsoft-standard-WSL2   docker://27.4.1
deepakp@Deepak:~$ kubectl replace -f servicefile.yml
service/my-service replaced
deepakp@Deepak:~$ minikube service my-service
|-----------|------------|-------------|---------------------------|
| NAMESPACE |    NAME    | TARGET PORT |            URL            |
|-----------|------------|-------------|---------------------------|
| default   | my-service |        7070 | http://192.168.49.2:30002 |
|-----------|------------|-------------|---------------------------|

❌  Exiting due to SVC_UNREACHABLE: service not available: no running pod for service my-service found

    😿  If the above advice does not help, please let us know:
    👉  https://github.com/kubernetes/minikube/issues/new/choose

    Please run `minikube logs --file=logs.txt` and attach logs.txt to the GitHub issue.
    Please also attach the following file to the GitHub issue:
    - /tmp/minikube_service_55f3c56f4afd0413e69b4fcbe9aee8c782ab04bd_0.log

deepakp@Deepak:~$
```

Minikube HTML output



```
deepakp@Deepak:~$ kubectl get services
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP          25h
my-service   NodePort    10.99.217.87   <none>        7070:30002/TCP   73m
deepakp@Deepak:~$ minikube service my-service
|-----------|------------|-------------|---------------------------|
| NAMESPACE |    NAME    | TARGET PORT |            URL            |
|-----------|------------|-------------|---------------------------|
| default   | my-service |        7070 | http://192.168.49.2:30002 |
|-----------|------------|-------------|---------------------------|
🎉  Opening service default/my-service in default browser...
👉  http://192.168.49.2:30002
deepakp@Deepak:~$ sudo nano deploymentfile.yml
deepakp@Deepak:~$ kubectl replace -f deploymentfile.yml
service/my-service replaced
deepakp@Deepak:~$ minikube service my-service
|-----------|------------|-------------|---------------------------|
| NAMESPACE |    NAME    | TARGET PORT |            URL            |
|-----------|------------|-------------|---------------------------|
| default   | my-service |        9000 | http://192.168.49.2:30002 |
|-----------|------------|-------------|---------------------------|
🎉  Opening service default/my-service in default browser...
👉  http://192.168.49.2:30002
deepakp@Deepak:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP          25h
my-service   NodePort    10.99.217.87   <none>        9000:30002/TCP   78m
deepakp@Deepak:~$ kubectl get svc my-service
NAME         TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
my-service   NodePort   10.99.217.87   <none>        9000:30002/TCP   78m
deepakp@Deepak:~$ minikube ip
192.168.49.2
deepakp@Deepak:~$ curl  http://192.168.49.2:30002
<!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
s="line" /><h3>Apache Tomcat/9.0.102</h3></body></html>deepakp@Deepak:~$ |
```

Minikube Hello World HTMl File output



```
deepakp@Deepak:~$ curl  http://192.168.49.2:30002/maven-web-app/
<html>
<body>
<h2>Hello World!</h2>
</body>
</html>
deepakp@Deepak:~$ |
```

Deployment YAML Configuration

Deployment Example:

apiVersion: apps/v1

kind: Deployment

metadata:

  name: my-deploy

  labels:

    name: my-deploy

spec:

  replicas: 4

```yaml
 selector:
   matchLabels:
     apptype: web-backend
 strategy:
   type: RollingUpdate
 template:
   metadata:
     labels:
       apptype: web-backend
   spec:
     containers:
     - name: my-app
       image: deepakp2003/app:latest
       ports:
         - containerPort: 7070
```

Namespace Creation:

```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: my-demo-ns
```

Pod Management Commands

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 Pods***
   - ***In Default Namespace:***
     kubectl get pods
   - ***In All Namespaces:***
     kubectl get pods -A
   - ***For a Specific Namespace:***
     kubectl get pods -n kube-system
   - ***For a Specific Pod:***
     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

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

Pod & ReplicaSet YAML Configuration

Pod Example:

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
  labels:
      app: my-web-app
spec:
  containers:
    - name: nginx-container
      image: deepakp2003/app:latest
      ports:
        - containerPort: 80
```

Pod with Namespace:

```
apiVersion: v1
kind: Pod
metadata:
  name: my-deploy
  namespace: mydeploy
spec:
  containers:
  - name: my-container
    image: nginx:latest
```

ReplicaSet Example:

```
apiVersion: apps/v1
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: deepakp/app:latest
        ports:
          - containerPort: 8081
```