

DAY 4

```
GNU nano 7.2 pod.yml Restore down
apiVersion: v1
kind: Pod
metadata:
  name: my-app
  labels:
    app: my-web-app
    type: backend
spec:
  containers:
  - name: my-app-container
    image: roshni2108/devops:latest
    ports:
    - containerPort: 9090

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo     M-A Set Mark
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo     M-6 Copy
```

```
roshni@LAPTOP-1VNKAF6I: ~$ kubectl delete all --all
pod "web-nginx-546f4997bb-t5wvw" deleted
service "kubernetes" deleted
deployment.apps "web-nginx" deleted
replicaset.apps "web-nginx-546f4997bb" deleted
roshni@LAPTOP-1VNKAF6I: ~$ kubectl get pod
No resources found in default namespace.
roshni@LAPTOP-1VNKAF6I: ~$ sudo nano deployment.yml
[sudo] password for roshni:
roshni@LAPTOP-1VNKAF6I: ~$ kubectl apply -f deployment.yml
deployment.apps/my-deploy created
service/my-service created
roshni@LAPTOP-1VNKAF6I: ~$ minikube service my-service
-----
| NAMESPACE | NAME       | TARGET PORT | URL                               |
|-----|-----|-----|-----|
| default   | my-service | 7070        | http://192.168.49.2:30002       |
|-----|-----|-----|-----|
🚀 Starting tunnel for service my-service.
-----
| NAMESPACE | NAME       | TARGET PORT | URL                               |
|-----|-----|-----|-----|
| default   | my-service |              | http://127.0.0.1:40331         |
|-----|-----|-----|-----|
🌐 Opening service default/my-service in default browser...
👉 http://127.0.0.1:40331
! Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```

1. Create a pod using run command

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

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>
```

4. View Logs of a pod

```
$ kubectl logs <pod-name>
```

5. Execute any command inside Pod (Inside Pod OS)

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

6. Delete Deployment

```
$ kubectl delete deploy <deployment-name>
```

POD.YML

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: my-app
```

```
spec:
```

containers:

- name: my-app-container

image: <images> ports:

- containerPort: 9090

DEPLOYMENT.YML

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

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

1. 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
```

2. View ReplicaSets

```
$ kubectl get replicaset
```

```
$ kubectl get rs
```

```
$ kubectl get rs -o wide
```

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

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

3. View ReplicaSet Description

```
$ kubectl describe rs <replica-set-name>
```

4. We can modify generated/updated YAML file

```
$ kubectl edit rs <replica-set-name>
```

DEPLOYMENT.YML (service)

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