```
deepa@ubuntu:~/devops/K8s$ minikube stop
* Stopping node "minikube"  ...
* Powering off "minikube" via SSH ...
* 1 node stopped.
```

```
deepa@ubuntu:~/devops/K8s$ minikube delete
* Deleting "minikube" in docker ...
* Deleting container "minikube" ...
* Removing /home/deepa/.minikube/machines/minikube ...
* Removed all traces of the "minikube" cluster.
```

```
deepa@ubuntu:~/devops/K8s$ minikube start --driver=docker
* minikube v1.37.0 on Ubuntu 20.04
* Using the docker driver based on user configuration
* Using Docker driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.48 ...
* Creating docker container (CPUs=2, Memory=3072MB) ...
* Preparing Kubernetes v1.34.0 on Docker 28.4.0 ...
* Configuring bridge CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
deepa@ubuntu:~/devops/K8s$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED              STATUS              PORTS
                                                                                                                    NAMES
6f200f665dcf   gcr.io/k8s-minikube/kicbase:v0.0.48   "/usr/local/bin/entr…"   About a minute ago   Up About a minute   127.0.0.1:32777→22/tcp, 127.0.0.1:32776→2376/tcp,
127.0.0.1:32775→5000/tcp, 127.0.0.1:32774→8443/tcp, 127.0.0.1:32773→32443/tcp   minikube
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get nodes
NAME       STATUS   ROLES           AGE     VERSION
minikube   Ready    control-plane   3m29s   v1.34.0
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /home/deepa/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   7m49s
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl create deployment mydeploy --image=nginx
deployment.apps/mydeploy created
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get all
NAME                                 READY   STATUS              RESTARTS   AGE
pod/mydeploy-8d4c44bfc-spg4f         0/1     ContainerCreating   0          13s

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   11m

NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mydeploy     0/1     1            0           13s

NAME                                    DESIRED   CURRENT   READY   AGE
replicaset.apps/mydeploy-8d4c44bfc      1         1         0       13s
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get all
NAME                                 READY   STATUS             RESTARTS   AGE
pod/mydeploy-8d4c44bfc-spg4f         0/1     ImagePullBackOff   0          3m55s

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   14m

NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mydeploy     0/1     1            0           3m55s

NAME                                    DESIRED   CURRENT   READY   AGE
replicaset.apps/mydeploy-8d4c44bfc      1         1         0       3m55s
deepa@ubuntu:~/devops/K8s$
```

- Common issues:
  - Typo in image name
  - Missing tag (e.g., nginx vs nginx:latest)
  - Private registry without credentials

“kubectl describe pod mydeploy-8d4c44bfc-spg4f”

-==========

```
deepa@ubuntu:~/devops/K8s$ kubectl create deploy deepadeploy --image=deepat1128/gitdemo1imagedt/1.0 --port=9090
deployment.apps/deepadeploy created
deepa@ubuntu:~/devops/K8s$ kubectl get deploy
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
deepadeploy  0/1     1            0           16s
deepa@ubuntu:~/devops/K8s$ kubectl get poda
error: the server doesn't have a resource type "poda"
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                          READY   STATUS             RESTARTS   AGE
deepadeploy-6575f694c-4m4n9   0/1     ImagePullBackOff   0          36s
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                          READY   STATUS         RESTARTS   AGE
deepadeploy-6575f694c-4m4n9   0/1     ErrImagePull   0          45s
```

```
deepa@ubuntu:~/devops/K8s$ kubectl create deploy deepadeploy --image=deepat1128/gitdemo1imagedt:1.0 --port=9090
deployment.apps/deepadeploy created
deepa@ubuntu:~/devops/K8s$ kubectl get deploy
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deepadeploy   0/1     1            0           9s
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                           READY   STATUS              RESTARTS   AGE
deepadeploy-8495f7f86b-wl9gs   0/1     ContainerCreating   0          20s
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                           READY   STATUS              RESTARTS   AGE
deepadeploy-8495f7f86b-wl9gs   0/1     ContainerCreating   0          33s
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                           READY   STATUS    RESTARTS   AGE
deepadeploy-8495f7f86b-wl9gs   1/1     Running   0          5m56s
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl describe pod mydeploy-8d4c44bfc-spg4f
Name:             mydeploy-8d4c44bfc-spg4f
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Sat, 04 Oct 2025 10:07:10 -0700
Labels:           app=mydeploy
                  pod-template-hash=8d4c44bfc
Annotations:      <none>
Status:           Running
IP:               10.244.0.5
IPs:
  IP:           10.244.0.5
Controlled By:  ReplicaSet/mydeploy-8d4c44bfc
Containers:
  nginx:
    Container ID:   docker://226814d3f5a398b183e54a078b62be9b0152e99c78547a93cdf764e458811b12
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:8adbdcb969e2676478ee2c7ad333956f0c8e0e4c5a7463f4611d7a2e7a7ff5dc
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Sat, 04 Oct 2025 10:11:29 -0700
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-2xdks (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-2xdks:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    Optional:                false
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s

Events:
  Type     Reason          Age                From               Message
  ----     ------          ----               ----               -------
  Normal   Scheduled       13m                default-scheduler  Successfully assigned default/mydeploy-8d4c44bfc-spg4f to minikube
  Warning  Failed          11m                kubelet            Failed to pull image "nginx": dial tcp: lookup docker-images-prod.6aa30f8b08e16409b46e0173
dflarestorage.com on 192.168.49.1:53: server misbehaving
  Warning  Failed          11m                kubelet            Error: ErrImagePull
  Normal   SandboxChanged  11m                kubelet            Pod sandbox changed, it will be killed and re-created.
  Normal   BackOff         11m (x2 over 11m)  kubelet            Back-off pulling image "nginx"
  Warning  Failed          11m (x2 over 11m)  kubelet            Error: ImagePullBackOff
  Normal   Pulling         10m (x2 over 13m)  kubelet            Pulling image "nginx"
  Normal   Pulled          9m11s              kubelet            Successfully pulled image "nginx" in 1m37.986s (1m37.986s including waiting). Image size:
  Normal   Created         9m11s              kubelet            Created container: nginx
  Normal   Started         9m11s              kubelet            Started container nginx
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get all
NAME                              READY   STATUS    RESTARTS   AGE
pod/mydeploy-8d4c44bfc-spg4f      1/1     Running   0          17m

NAME                 TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1     <none>        443/TCP   28m

NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mydeploy     1/1     1            1           17m

NAME                                    DESIRED   CURRENT   READY   AGE
replicaset.apps/mydeploy-8d4c44bfc      1         1         1       17m
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get deploy
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
mydeploy   1/1     1            1           20m
deepa@ubuntu:~/devops/K8s$ kubectl get deploy -o wide
NAME       READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES   SELECTOR
mydeploy   1/1     1            1           20m   nginx        nginx    app=mydeploy
deepa@ubuntu:~/devops/K8s$
```

label

```
deepa@ubuntu:~/devops/K8s$ kubectl rollout history deploy mydeploy
deployment.apps/mydeploy
REVISION  CHANGE-CAUSE
1         <none>

deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl scale deploy mydeploy --replicas=2
deployment.apps/mydeploy scaled
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get pod
NAME                        READY   STATUS    RESTARTS   AGE
mydeploy-8d4c44bfc-klkr4    1/1     Running   0          38s
mydeploy-8d4c44bfc-spg4f    1/1     Running   0          27m
deepa@ubuntu:~/devops/K8s$
```

Kubectl get deploy mydeploy -o yaml >mydeploy.yml

```
deepa@ubuntu:~/devops/K8s$ kubectl get deploy mydeploy -o yaml > mydeploy.yml
deepa@ubuntu:~/devops/K8s$ ls
kubectl  minikube-linux-amd64  mydeploy.yml  ns.yaml
deepa@ubuntu:~/devops/K8s$ cat mydeploy.yml
```

```
deepa@ubuntu:~/devops/K8s$ minikube stop
* Stopping node "minikube"  ...
* Powering off "minikube" via SSH ...
* 1 node stopped.
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ cat mydeploy.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: "2025-10-04T17:07:10Z"
  generation: 2
  labels:
    app: mydeploy
  name: mydeploy
  namespace: default
  resourceVersion: "2315"
  uid: 0d3829a6-f811-47c1-99c8-30cb089a3e27
spec:
  progressDeadlineSeconds: 600
  replicas: 2
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: mydeploy
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: mydeploy
    spec:
      containers:
      - image: nginx
        imagePullPolicy: Always
        name: nginx
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30
status:
  availableReplicas: 2
  conditions:
  - lastTransitionTime: "2025-10-04T17:07:10Z"
    lastUpdateTime: "2025-10-04T17:11:29Z"
    message: ReplicaSet "mydeploy-8d4c44bfc" has successfully progressed.
    reason: NewReplicaSetAvailable
    status: "True"
    type: Progressing
  - lastTransitionTime: "2025-10-04T17:34:17Z"
    lastUpdateTime: "2025-10-04T17:34:17Z"
    message: Deployment has minimum availability.
    reason: MinimumReplicasAvailable
    status: "True"
    type: Available
  observedGeneration: 2
  readyReplicas: 2
  replicas: 2
  updatedReplicas: 2
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ vi mydeploy.yml
deepa@ubuntu:~/devops/K8s$
```

```
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: mydeploy
    spec:
      containers:
      - image: nginx:alpine
        imagePullPolicy: Always
        name: nginx
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30
status:
  availableReplicas: 2
```

edited

```
deepa@ubuntu:~/devops/K8s$ kubectl apply -f mydeploy.yml
Warning: resource deployments/mydeploy is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is required by kubectl apply. kubectl apply should only be used on resources created declaratively by either kubectl create --save-config or kubectl apply. The missing annotation will be patched automatically.
deployment.apps/mydeploy configured
deepa@ubuntu:~/devops/K8s$ kubectl rollout history deploy mydeploy
deployment.apps/mydeploy
REVISION  CHANGE-CAUSE
1         <none>
2         <none>
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get all
NAME                             READY   STATUS    RESTARTS   AGE
pod/mydeploy-86b956b58d-b6qlm    1/1     Running   0          16m
pod/mydeploy-86b956b58d-cndcx    1/1     Running   0          16m
```

```
deepa@ubuntu:~/devops/K8s$ kubectl delete deployment mydeploy
deployment.apps "mydeploy" deleted from default namespace
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   92m
```

```
deepa@ubuntu:~/devops/K8s$ kubectl exec -it deepadeploy-8495f7f86b-b8sbv -- bash
OCI runtime exec failed: exec failed: unable to start container process: exec: "bash": executable file not found in $PATH: unknown
command terminated with exit code 127
deepa@ubuntu:~/devops/K8s$ kubectl exec -it deepadeploy-8495f7f86b-b8sbv -- sh
/app #
```

07.10.2025

“minikube start”

```
deepa@ubuntu:~/devops/K8s$ kubectl run pod1 --image=deepat1128/gitdemo1imagedt:1.0 --port=9090
pod/pod1 created
deepa@ubuntu:~/devops/K8s$ kubectl run pod2 --image=deepat1128/gitprojectimage:1.0 --port=9080
pod/pod2 created
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                              READY   STATUS    RESTARTS      AGE
deepadeploy-8495f7f86b-d27x2      1/1     Running   2 (13m ago)   39h
deepadeploy-8495f7f86b-hkp6k      1/1     Running   0             22s
deepadeploy-8495f7f86b-nz4lt      1/1     Running   2 (13m ago)   39h
pod1                              1/1     Running   0             5m9s
pod2                              1/1     Running   0             3m57s
```

Number of pods reduced by scaling down:

```
deepa@ubuntu:~/devops/K8s$ kubectl scale deploy deepadeploy --replicas=1
deployment.apps/deepadeploy scaled
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                              READY   STATUS    RESTARTS      AGE
deepadeploy-8495f7f86b-nz4lt      1/1     Running   2 (20m ago)   39h
pod1                              1/1     Running   0             11m
pod2                              1/1     Running   0             10m
deepa@ubuntu:~/devops/K8s$
```

In logs of pod1, end point is shown: /demo1/h2

```
deepa@ubuntu:~/devops/K8s$ kubectl logs pod1

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.7.0)

2025-10-07 09:59:10.037  INFO 1 --- [           main] com.example.demo1.Demo1Application       : Starting Demo1Application v0.0.1-SNAPSHOT using Java 17.0.16 on pod1 with
PID 1 (/app/app.jar started by root in /app)
2025-10-07 09:59:10.059  INFO 1 --- [           main] com.example.demo1.Demo1Application       : No active profile set, falling back to 1 default profile: "default"
2025-10-07 09:59:15.457  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2025-10-07 09:59:15.856  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 307 ms. Found 1 JPA repository
 interfaces.
2025-10-07 09:59:19.840  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2025-10-07 09:59:19.872  INFO 1 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2025-10-07 09:59:19.873  INFO 1 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.63]
2025-10-07 09:59:20.177  INFO 1 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2025-10-07 09:59:20.178  INFO 1 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 9521 ms
2025-10-07 09:59:20.362  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2025-10-07 09:59:21.369  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2025-10-07 09:59:21.386  INFO 1 --- [           main] o.s.b.a.h2.H2ConsoleAutoConfiguration    : H2 console available at '/demo1/h2'. Database available at 'jdbc:h2:mem:de
mo1'
2025-10-07 09:59:22.682  INFO 1 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2025-10-07 09:59:23.093  INFO 1 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.6.9.Final
```

```
deepa@ubuntu:~/devops/K8s$ kubectl exec -it pod1 -- sh
/app # env
KUBERNETES_SERVICE_PORT=443
KUBERNETES_PORT=tcp://10.96.0.1:443
LANGUAGE=en_US:en
HOSTNAME=pod1
SHLVL=1
HOME=/root
JAVA_VERSION=jdk-17.0.16+8
TERM=xterm
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
PATH=/opt/java/openjdk/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_PROTO=tcp
LANG=en_US.UTF-8
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
LC_ALL=en_US.UTF-8
KUBERNETES_SERVICE_HOST=10.96.0.1
JAVA_HOME=/opt/java/openjdk
PWD=/app
/app #
```

Helm Installation:

deepa@ubuntu:~/devops/K8s$ curl https://baltocdn.com/helm/signing.asc | gpg --dearmor | sudo tee /usr/share/keyrings/helm.gpg > /dev/null

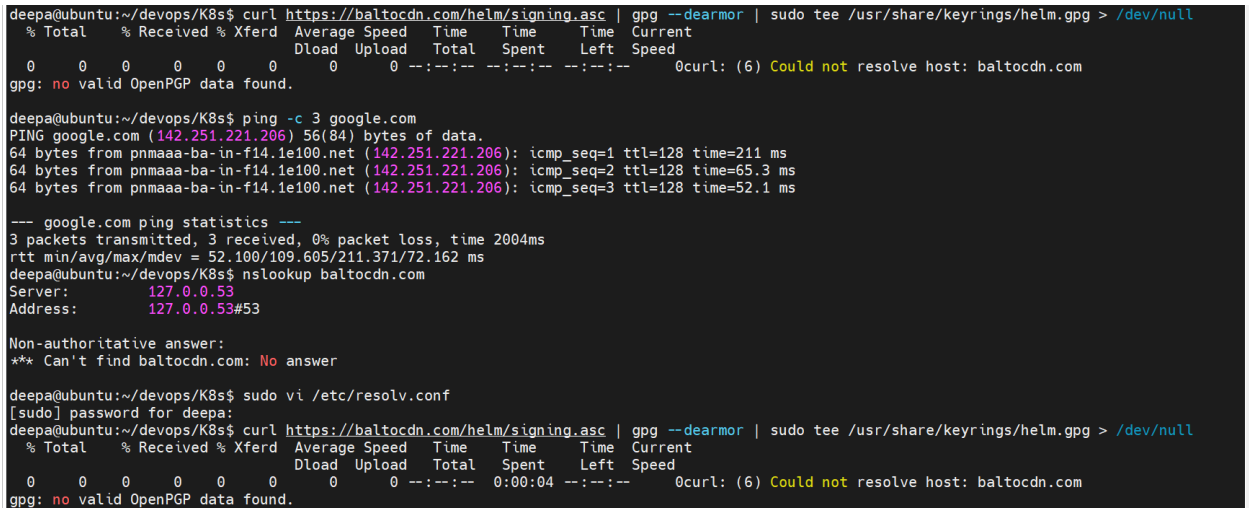
```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--
0curl: (6) Could not resolve host: baltocdn.com
gpg: no valid OpenPGP data found.
```

deepa@ubuntu:~/devops/K8s$ ping -c 3 google.com

```
PING google.com (142.251.221.206) 56(84) bytes of data.
64 bytes from pnmaaa-ba-in-f14.1e100.net (142.251.221.206): icmp_seq=1 ttl=128 time=211 ms
64 bytes from pnmaaa-ba-in-f14.1e100.net (142.251.221.206): icmp_seq=2 ttl=128 time=65.3 ms
64 bytes from pnmaaa-ba-in-f14.1e100.net (142.251.221.206): icmp_seq=3 ttl=128 time=52.1 ms

--- google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 52.100/109.605/211.371/72.162 ms
```

```
deepa@ubuntu:~/devops/K8s$ nslookup baltocdn.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
*** Can't find baltocdn.com: No answer

deepa@ubuntu:~/devops/K8s$ sudo vi /etc/resolv.conf
[sudo] password for deepa:
deepa@ubuntu:~/devops/K8s$ curl https://baltocdn.com/helm/signing.asc | gpg
--dearmor | sudo tee /usr/share/keyrings/helm.gpg > /dev/null
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
Time  Current
                                 Dload  Upload   Total   Spent
Left  Speed
  0     0    0     0    0     0      0      0 --:--:--  0:00:04 --:--:--
0curl: (6) Could not resolve host: baltocdn.com
gpg: no valid OpenPGP data found.
deepa@ubuntu:~/devops/K8s$ sudo rm /etc/resolv.conf
deepa@ubuntu:~/devops/K8s$ sudo ln -s /run/systemd/resolve/resolv.conf
/etc/resolv.conf
deepa@ubuntu:~/devops/K8s$ sudo vi /etc/systemd/resolved.conf
deepa@ubuntu:~/devops/K8s$ sudo vi /etc/systemd/resolved.conf
deepa@ubuntu:~/devops/K8s$ sudo systemctl restart systemd-resolved
deepa@ubuntu:~/devops/K8s$ nslookup baltocdn.com
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
*** Can't find baltocdn.com: No answer

deepa@ubuntu:~/devops/K8s$ curl -fsSL
https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
Downloading https://get.helm.sh/helm-v3.19.0-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
helm installed into /usr/local/bin/helm
```

deepa@ubuntu:~/devops/K8s$ helm version
version.BuildInfo{Version:"v3.19.0",
GitCommit:"3d8990f0836691f0229297773f3524598f46bda6",
GitTreeState:"clean", GoVersion:"go1.24.7"}
deepa@ubuntu:~/devops/K8s$



---

deepa@ubuntu:~/devops/K8s$ kubectl describe pod pod1

```
deepa@ubuntu:~/devops/K8s$ kubectl describe pod pod1
Name:             pod1
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Tue, 07 Oct 2025 02:59:00 -0700
Labels:           run=pod1
Annotations:      <none>
Status:           Running
IP:               10.244.0.31
IPs:
  IP:  10.244.0.31
Containers:
  pod1:
    Container ID:   docker://6d7279bee720697b00b2d353654bfea0e62a8967c7dea7d278ba4abc3caa4ca4
    Image:          deepat1128/gitdemo1imagedt:1.0
    Image ID:       docker-pullable://deepat1128/gitdemo1imagedt@sha256:6dce46ec3cc67feafc1552a8b58468717a22f55924acd7dc3c8019c7db45b619
    Port:           9090/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 07 Oct 2025 06:31:55 -0700
    Last State:     Terminated
      Reason:       Error
      Exit Code:    143
      Started:      Tue, 07 Oct 2025 02:59:02 -0700
      Finished:     Tue, 07 Oct 2025 06:31:30 -0700
    Ready:          True
    Restart Count:  1
    Environment:    <none>
```

```
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-zmk94 (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-zmk94:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    Optional:                false
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason          Age                From     Message
  ----    ------          ----               ----     -------
  Normal  SandboxChanged  28m                kubelet  Pod sandbox changed, it will be killed and re-created.
  Normal  Pulled          28m (x2 over 4h1m)  kubelet  Container image "deepat1128/gitdemo1imagedt:1.0" already present on machine
  Normal  Created         28m (x2 over 4h1m)  kubelet  Created container: pod1
  Normal  Started         28m (x2 over 4h1m)  kubelet  Started container pod1
deepa@ubuntu:~/devops/K8s$ ^C
deepa@ubuntu:~/devops/K8s$
```

08.10.2025

```
deepa@ubuntu:~/devops/K8s$ kubectl get deploy
No resources found in default namespace.
deepa@ubuntu:~/devops/K8s$ kubectl get deploy -A
NAMESPACE     NAME      READY   UP-TO-DATE   AVAILABLE   AGE
kube-system   coredns   1/1     1            1           3m28s
deepa@ubuntu:~/devops/K8s$
```

kubectl get deploy (default namespace): No deployments exist in the default namespace yet.

kubectl get deploy -A (all namespaces): You see coredns running in the kube-system namespace, which is expected—it's a core Kubernetes DNS service.

Kube-system: It houses core components that keep your Kubernetes cluster running smoothly.
Managed by Kubernetes itself—cluster admins and system processes deploy resources here.
Keeps system-level workloads separate from user workloads for clarity and security.
- coredns: Handles internal DNS resolution so pods can find each other by name.
- kube-proxy: Manages networking rules for communication between pods and services.
- etcd: Stores cluster state (used in control plane nodes).
- kube-controller-manager**,** kube-scheduler: Control plane components managing cluster behavior.
- **Add-ons**: Metrics server, dashboard, network plugins (like Calico or Flannel), and cloud provider integrations.

The kube-system namespace is a **reserved space** in every Kubernetes cluster for **system-level components** that support cluster operations. It's created automatically when the cluster is initialized.
shouldn't deploy our own apps here, because
Risk of interference: Your workloads could affect system stability.
Security concerns: System pods often run with elevated privileges.
Audit clarity: Mixing system and user workloads complicates monitoring and debugging.
kubectl get all -n kube-system

kubectl get daemonset -A

```
deepa@ubuntu:~/devops/K8s$ kubectl get daemonset -A
NAMESPACE     NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-system   kube-proxy   1         1         1       1            1           kubernetes.io/os=linux   26m
deepa@ubuntu:~/devops/K8s$
```

kubectl get replicaset -A, coredns is deployment created

```
deepa@ubuntu:~/devops/K8s$ kubectl get replicaset -A
NAMESPACE     NAME                 DESIRED   CURRENT   READY   AGE
kube-system   coredns-66bc5c9577   1         1         1       30m
deepa@ubuntu:~/devops/K8s$
```

use this command to inspect the ownership chain:
kubectl get replicaset coredns-66bc5c9577 -n kube-system -o yaml

```
deepa@ubuntu:~/devops/K8s$ kubectl get replicaset coredns-66bc5c9577 -n kube-system -o yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  annotations:
    deployment.kubernetes.io/desired-replicas: "1"
    deployment.kubernetes.io/max-replicas: "2"
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: "2025-10-08T04:33:53Z"
  generation: 2
  labels:
    k8s-app: kube-dns
    pod-template-hash: 66bc5c9577
  name: coredns-66bc5c9577
  namespace: kube-system
  ownerReferences:
  - apiVersion: apps/v1
    blockOwnerDeletion: true
    controller: true
    kind: Deployment
    name: coredns
    uid: 4d820529-ebde-41c4-9fbb-a1469fd244c4
  resourceVersion: "538"
  uid: cc3bd61f-b3b2-4ec5-9b0a-74a17feaceae
spec:
  replicas: 1
  selector:
```

kubectl create job test --image=hello-world --dry-run=client -o yaml

```
deepa@ubuntu:~/devops/K8s$ kubectl create job test --image=hello-world --dry-run=client -o yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: test
spec:
  template:
    metadata: {}
    spec:
      containers:
      - image: hello-world
        name: test
        resources: {}
      restartPolicy: Never
status: {}
```

kubectl get pods

```
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME         READY   STATUS      RESTARTS   AGE
test-tv4mm   0/1     Completed   0          107s
deepa@ubuntu:~/devops/K8s$
```

kubectl logs test-tv4mm

```
deepa@ubuntu:~/devops/K8s$ kubectl logs test-tv4mm

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

```
deepa@ubuntu:~/devops/K8s$ kubectl exec -it test-tv4mm -- bash
error: cannot exec into a container in a completed pod; current phase is Succeeded
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME         READY   STATUS      RESTARTS   AGE
test-tv4mm   0/1     Completed   0          10m
deepa@ubuntu:~/devops/K8s$ kubectl exec -it test-tv4mm -- bash
error: cannot exec into a container in a completed pod; current phase is Succeeded
deepa@ubuntu:~/devops/K8s$
```

sir, is this a state in the life cycle of a pod (created for job)

```
deepa@ubuntu:~/devops/K8s$ cp mydeploy.yml statefulset.yaml
deepa@ubuntu:~/devops/K8s$ ls
deepa.yaml  job.yaml  kubectl  minikube-linux-amd64  mydeploy.yml  ns.yaml  pod.yaml  statefulset.yaml
deepa@ubuntu:~/devops/K8s$ vi statefulset.yaml
deepa@ubuntu:~/devops/K8s$ vi statefulset.yaml
deepa@ubuntu:~/devops/K8s$
```

In deployment yaml file , kind changed, spec progressDeadlineSeconds – removed and status block made empty, replica related lines removed. Then applied , it created. Strategy also removed.

kubectl apply -f statefulset.yaml
kubectl explain sts

```
deepa@ubuntu:~/devops/K8s$ kubectl explain sts
GROUP:      apps
KIND:       StatefulSet
VERSION:    v1

DESCRIPTION:
    StatefulSet represents a set of pods with consistent identities. Identities
    are defined as:
      - Network: A single stable DNS and hostname.
      - Storage: As many VolumeClaims as requested.

    The StatefulSet guarantees that a given network identity will always map to
    the same storage identity.
```

Now should add replica in yaml file, and apply

```
deepa@ubuntu:~/devops/K8s$ cat cron.yaml
apiVersion: batch/v1
kind: CronJob
metadata:
  name: cronjobname
spec:
  jobTemplate:
    metadata:
      name: cronjobname
    spec:
      template:
        metadata: {}
        spec:
          containers:
          - image: hello-world
            name: cronjobname
            resources: {}
          restartPolicy: OnFailure
  schedule: '* * * * *'
status: {}
```

```
deepa@ubuntu:~/devops/K8s$ vi cron1.yaml
deepa@ubuntu:~/devops/K8s$ kubectl apply -f cron.yaml
cronjob.batch/cronjobname created
deepa@ubuntu:~/devops/K8s$ kubectl get cronjob
```

```
deepa@ubuntu:~/devops/K8s$ kubectl get cronjob
NAME          SCHEDULE    TIMEZONE   SUSPEND   ACTIVE   LAST SCHEDULE   AGE
cronjobname   * * * * *   <none>     False     0        <none>          28s
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                           READY   STATUS      RESTARTS   AGE
cronjobname-29331821-2n9xm     0/1     Completed   0          20s
mydeploy-0                     1/1     Running     0          53m
test-tv4mm                     0/1     Completed   0          92m
deepa@ubuntu:~/devops/K8s$
```

Command : watch “kubectl get pods”

```
Every 2.0s: kubectl get pods

NAME                           READY   STATUS      RESTARTS   AGE
cronjobname-29331821-2n9xm     0/1     Completed   0          78s
cronjobname-29331822-4jd9c     0/1     Completed   0          18s
mydeploy-0                     1/1     Running     0          54m
test-tv4mm                     0/1     Completed   0          93m
```

```
Every 2.0s: kubectl get pods

NAME                           READY   STATUS      RESTARTS   AGE
cronjobname-29331822-4jd9c     0/1     Completed   0          2m11s
cronjobname-29331823-ltg8t     0/1     Completed   0          71s
cronjobname-29331824-xnks2     0/1     Completed   0          11s
mydeploy-0                     1/1     Running     0          56m
test-tv4mm                     0/1     Completed   0          95m
```

```
deepa@ubuntu:~/devops/K8s$ kubectl delete cronjob cronjobname
cronjob.batch "cronjobname" deleted from default namespace
deepa@ubuntu:~/devops/K8s$
```

```
deepa@ubuntu:~/devops/K8s$ git init
Initialized empty Git repository in /home/deepa/devops/K8s/.git/
```

```
deepa@ubuntu:~/devops/K8s$ git add .
deepa@ubuntu:~/devops/K8s$ git commit -m "Initial commit: K8s YAMLs and binaries"
[master (root-commit) ed23849] Initial commit: K8s YAMLs and binaries
 9 files changed, 178 insertions(+)
 create mode 100644 cron.yaml
 create mode 100644 deepa.yaml
 create mode 100644 job.yaml
 create mode 100644 kubectl
 create mode 100644 minikube-linux-amd64
 create mode 100644 mydeploy.yml
 create mode 100644 ns.yaml
 create mode 100644 pod.yaml
 create mode 100644 statefulset.yaml
deepa@ubuntu:~/devops/K8s$ ls
cron1.yaml  cron.yaml  deepa.yaml  job.yaml  kubectl  minikube-linux-amd64  mydeploy.yml  ns.yaml  pod.yaml  statefulset.yaml
deepa@ubuntu:~/devops/K8s$ git add cron1.yaml
deepa@ubuntu:~/devops/K8s$ git commit -m "cron1.yaml"
[master 4811f30] cron1.yaml
 1 file changed, 19 insertions(+)
 create mode 100644 cron1.yaml
```

```
deepa@ubuntu:~/devops/K8s$ ssh -T git@github.com
Hi deepat1128! You've successfully authenticated, but GitHub does not provide shell access.
deepa@ubuntu:~/devops/K8s$ git branch
* master
deepa@ubuntu:~/devops/K8s$ git remote add origin git@github.com:deepat1128/K8s.git
deepa@ubuntu:~/devops/K8s$
deepa@ubuntu:~/devops/K8s$ git branch -M main
deepa@ubuntu:~/devops/K8s$ git branch
* main
```

```
deepa@ubuntu:~/devops/K8s$ git push -u origin main
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
^Citing objects:  57% (8/14), 17.14 MiB | 11.00 KiB/s
deepa@ubuntu:~/devops/K8s$ git reset HEAD~1
deepa@ubuntu:~/devops/K8s$ ls
cron1.yaml  cron.yaml  deepa.yaml  job.yaml  kubectl  minikube-linux-amd64  mydeploy.yml  ns.yaml  pod.yaml  statefulset.yaml
```

```
deepa@ubuntu:~/devops/K8s$ git add pod.yaml
deepa@ubuntu:~/devops/K8s$ git commit -m "Add pod.yaml: basic pod spec for demo"
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        cron1.yaml

nothing added to commit but untracked files present (use "git add" to track)
deepa@ubuntu:~/devops/K8s$ git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 67.66 MiB | 893.00 KiB/s, done.
Total 11 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
remote: warning: File kubectl is 57.75 MB; this is larger than GitHub's recommended maximum file size of 50.00 MB
remote: error: Trace: a17c18106afc3bac81082d4b381ad484b1f4ad3b41c1605e8a84a42224ebfe41
remote: error: See https://gh.io/lfs for more information.
remote: error: File minikube-linux-amd64 is 133.41 MB; this exceeds GitHub's file size limit of 100.00 MB
remote: error: GH001: Large files detected. You may want to try Git Large File Storage - https://git-lfs.github.com.
To github.com:deepat1128/K8s.git
 ! [remote rejected] main -> main (pre-receive hook declined)
error: failed to push some refs to 'git@github.com:deepat1128/K8s.git'
deepa@ubuntu:~/devops/K8s$ git reset
deepa@ubuntu:~/devops/K8s$ git add *.yaml
deepa@ubuntu:~/devops/K8s$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   cron1.yaml
```

```
deepa@ubuntu:~/devops/K8s$ git commit -m "cron1.yaml"
[main c6cd9b7] cron1.yaml
 1 file changed, 19 insertions(+)
 create mode 100644 cron1.yaml
deepa@ubuntu:~/devops/K8s$ git push origin main
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
Writing objects:  50% (7/14), 5.12 MiB | 17.00 KiB/s
```

09.10.2025

Create pod in declarative approach using yaml file:

```
deepa@ubuntu:~$ cd devops/K8s
deepa@ubuntu:~/devops/K8s$ ls
cron1.yaml  cron.yaml  deepa.yaml  job.yaml  kubectl  minikube-linux-amd64  mydeploy.yml  ns.yaml  pod2.yaml  pod.yaml  statefulset.yaml
deepa@ubuntu:~/devops/K8s$ vi pod2.yaml
deepa@ubuntu:~/devops/K8s$ kubectl apply -f pod2.yaml
pod/mypod created
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME         READY   STATUS              RESTARTS      AGE
mydeploy-0   1/1     Running             1 (25m ago)   28h
mypod        0/1     ContainerCreating   0             13s
test-tv4mm   0/1     Completed           0             28h
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME         READY   STATUS              RESTARTS      AGE
mydeploy-0   1/1     Running             1 (26m ago)   28h
mypod        0/1     ContainerCreating   0             21s
test-tv4mm   0/1     Completed           0             28h
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME         READY   STATUS              RESTARTS      AGE
mydeploy-0   1/1     Running             1 (26m ago)   28h
mypod        0/1     ContainerCreating   0             32s
test-tv4mm   0/1     Completed           0             28h
deepa@ubuntu:~/devops/K8s$ cat pod2.yaml
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
    - name: my-first-container
      image: nginx
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME         READY   STATUS      RESTARTS      AGE
mydeploy-0   1/1     Running     1 (28m ago)   28h
mypod        1/1     Running     0             2m39s
test-tv4mm   0/1     Completed   0             28h
deepa@ubuntu:~/devops/K8s$
```

Create a pod named mypod using nginx:alpine image:

Create a pod named pod-2 using image redis with label name=pod-2

Create a pod with name nginx and image nginx and expose it on port 8080

Class date: 04.10.2025

Kubectl get pods

Kubectl get deployment

```
deepa@ubuntu:~/devops/K8s$ kubectl get pods
NAME                        READY   STATUS      RESTARTS        AGE
mydeploy-0                  1/1     Running     2 (3m41s ago)   2d1h
mydeploy-86b956b58d-5q9t5   1/1     Running     1 (3m41s ago)   16h
mydeploy-86b956b58d-b9tzs   1/1     Running     1 (3m41s ago)   15h
mydeploy-86b956b58d-dpjcr   1/1     Running     1 (3m41s ago)   16h
test-tv4mm                  0/1     Completed   0               2d1h
deepa@ubuntu:~/devops/K8s$ ls
cron1.yaml  job.yaml              mydeploy.yml  pod.yaml
cron.yaml   kubectl               ns.yaml       statefulset.yaml
deepa.yaml  minikube-linux-amd64  pod2.yaml
deepa@ubuntu:~/devops/K8s$ kubectl get deployment
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
mydeploy   3/3     3            3           16h
```

Kubectl scale deploy deploymentname –replicas=1

```
deepa@ubuntu:~/devops/K8s$ kubectl scale deploy mydeploy --replicas=1
deployment.apps/mydeploy scaled
deepa@ubuntu:~/devops/K8s$ kubectl get deployment
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
mydeploy   1/1     1            1           16h
```

Here deployment is scaled down through cli, so yaml file need to be updated manually or using command.

Kubectl get deploy deploymentname -o yaml > updated-mydeploy.yml

Now, creating deployment with config , not command(like earlier), and contents of the yaml config to explore.

```
deepa@ubuntu:~/devops/K8s$ kubectl delete deploy mydeploy -n my-namespace
deployment.apps "mydeploy" deleted from my-namespace namespace
```

For the same command, add - - dry-run=client

Then add -o yaml or json or table

Kubectl create deploy deploymentname - - image=imagename - - port=ext:int - -dry-run=client -o yaml

```
deepa@ubuntu:~/devops/K8s$ kubectl create deployment mydeploy --image=nginx --po
rt=8050 --dry-run=client -n my-namespace
deployment.apps/mydeploy created (dry run)
```

Kubectl create deploy deploymentname - - image=imagename - - port=ext:int - -dry-run=client -o yaml > deploy.yaml

To Actually Create the Deployment, kubectl apply -f

```
deepa@ubuntu:~/devops/K8s$ kubectl apply -f mydeploy.yaml
deployment.apps/mydeploy created
deepa@ubuntu:~/devops/K8s$ kubectl get deploy mydeploy -n my-namespace
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
mydeploy   1/1     1            1           11s
deepa@ubuntu:~/devops/K8s$ kubectl get deploy mydeploy -n my-namespace -o wide
NAME       READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES   SELECTOR
mydeploy   1/1     1            1           23s   nginx        nginx    app=myde
ploy
```

kubectl get pods -n my-namespace -o wide

```
deepa@ubuntu:~/devops/K8s$ kubectl get pods -n my-namespace -o wide
NAME                          READY   STATUS    RESTARTS   AGE   IP             NOD
E       NOMINATED NODE   READINESS GATES
mydeploy-5b8477667f-6kc9f   1/1     Running   0          98s   10.244.0.30   min
ikube   <none>           <none>
deepa@ubuntu:~/devops/K8s$
```

Use this yaml file to modify further

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: mydeploy
  name: mydeploy
  namespace: my-namespace
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mydeploy
  strategy: {}
  template:
    metadata:
      labels:
        app: mydeploy
    spec:
      containers:
      - image: nginx
        name: nginx
        ports:
        - containerPort: 8050
        resources: {}
status: {}
~
```

Delete the deployment older

Now deployment.yaml> replica = 3

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: mydeploy
  name: mydeploy
  namespace: my-namespace
spec:
  replicas: 3
  selector:
    matchLabels:
      app: mydeploy
  strategy: {}
  template:
    metadata:
      labels:
        app: mydeploy
    spec:
      containers:
      - image: nginx
        name: nginx
        ports:
        - containerPort: 8050
        resources: {}
status: {}
```

Kubectl apply -f yamlfile

```
deepa@ubuntu:~/devops/K8s$ kubectl get deploy mydeploy -n my-namespace
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
mydeploy   1/1     1            1           8m57s
deepa@ubuntu:~/devops/K8s$ vi mydeploy.yaml
deepa@ubuntu:~/devops/K8s$ kubectl apply -f mydeploy.yaml
deployment.apps/mydeploy configured
deepa@ubuntu:~/devops/K8s$ kubectl get pods -A
NAMESPACE      NAME                                 READY   STATUS      RESTARTS      AGE
default        mydeploy-0                           1/1     Running     2 (47m ago)   2d1h
default        mydeploy-86b956b58d-b9tzs            1/1     Running     1 (47m ago)   16h
default        test-tv4mm                           0/1     Completed   0             2d2h
kube-system    coredns-66bc5c9577-5d64k             1/1     Running     2 (47m ago)   2d4h
kube-system    etcd-minikube                        1/1     Running     2 (47m ago)   2d4h
kube-system    kube-apiserver-minikube              1/1     Running     2 (47m ago)   2d4h
kube-system    kube-controller-manager-minikube     1/1     Running     3 (47m ago)   2d4h
kube-system    kube-proxy-fhq5p                     1/1     Running     2 (47m ago)   2d4h
kube-system    kube-scheduler-minikube              1/1     Running     2 (47m ago)   2d4h
kube-system    storage-provisioner                  1/1     Running     5 (45m ago)   2d4h
my-namespace   mydeploy-5b8477667f-6kc9f            1/1     Running     0             9m47s
my-namespace   mydeploy-5b8477667f-l5qlw            1/1     Running     0             9s
my-namespace   mydeploy-5b8477667f-p2x66            1/1     Running     0             9s
deepa@ubuntu:~/devops/K8s$
```

Kubectl logs podname

```
deepa@ubuntu:~/devops/K8s$ kubectl logs mydeploy-5b8477667f-l5qlw -n my-namespace
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/10/10 08:43:55 [notice] 1#1: using the "epoll" event method
2025/10/10 08:43:55 [notice] 1#1: nginx/1.29.2
2025/10/10 08:43:55 [notice] 1#1: built by gcc 14.2.0 (Debian 14.2.0-19)
2025/10/10 08:43:55 [notice] 1#1: OS: Linux 5.15.0-139-generic
2025/10/10 08:43:55 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/10/10 08:43:55 [notice] 1#1: start worker processes
2025/10/10 08:43:55 [notice] 1#1: start worker process 29
2025/10/10 08:43:55 [notice] 1#1: start worker process 30
2025/10/10 08:43:55 [notice] 1#1: start worker process 31
2025/10/10 08:43:55 [notice] 1#1: start worker process 32
2025/10/10 08:43:55 [notice] 1#1: start worker process 33
2025/10/10 08:43:55 [notice] 1#1: start worker process 34
2025/10/10 08:43:55 [notice] 1#1: start worker process 35
2025/10/10 08:43:55 [notice] 1#1: start worker process 36
```

Kubectl exec -it podname - - bash

kubectl exec -it mydeploy-5b8477667f-6kc9f -n my-namespace – bash

```
deepa@ubuntu:~/devops/K8s$ kubectl exec -it mydeploy-5b8477667f-6kc9f -n my-namespace -- bash
root@mydeploy-5b8477667f-6kc9f:/#
```

Inside container, curl localhost:port/endpoint , to check response , end point from the logs

root@mydeploy-5b8477667f-6kc9f:/# curl localhost:80/

```
deepa@ubuntu:~/devops/K8s$ kubectl exec -it mydeploy-5b8477667f-6kc9f -n my-namespace -- bash
root@mydeploy-5b8477667f-6kc9f:/# curl localhost:80/
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@mydeploy-5b8477667f-6kc9f:/#
```

Using private ip of pod , instead of localhost:

```
root@mydeploy-5b8477667f-6kc9f:/# curl 10.244.0.30:80/
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@mydeploy-5b8477667f-6kc9f:/#
```

Now curl other containers/ ping other containers from this container only, port will change and take private ip of the pod from command

```
root@mydeploy-5b8477667f-6kc9f:/# curl 10.244.0.31:80/
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@mydeploy-5b8477667f-6kc9f:/#
```

```
root@mydeploy-5b8477667f-6kc9f:/# curl 10.244.0.32:80/
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@mydeploy-5b8477667f-6kc9f:/#
```

Kubectl get pods -o wide

Inplaceof localhost add private ip of pod

Delete the pod, a new container will run again, and new ip will be assigned. Because it is part of deployment replicaset.

pods communicate with eachother in network using private ip and port

**********docker run -d --name sonarqube -p 9000:9000 sonarqube:latest******

Creating container using the yaml file:

Add config in deployment yaml file, below container details, copy -paste the lines , edit image and port.

Kubectl exec -it podname - -bash, it will select the first container out of many.

If you want to log in to other containers, command is

Kubectl exec -it podname -c containername - -bash

```
deepa@ubuntu:~/devops/K8s$ kubectl create deploy mydeploy --image=deepat1128/gitdemo1imagedt:1.0 --port=8050 --dry-run=client
deployment.apps/mydeploy created (dry run)
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: mydeploy
  name: mydeploy
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mydeploy
  strategy: {}
  template:
    metadata:
      labels:
        app: mydeploy
    spec:
      containers:
      - image: deepat1128/gitdemo1imagedt:1.0
        name: gitdemo1imagedt
        ports:
        - containerPort: 8050
        resources: {}
status: {}
~
```

New container creation in the pod:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: mydeploy
  name: mydeploy
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mydeploy
  strategy: {}
  template:
    metadata:
      labels:
        app: mydeploy
    spec:
      containers:
      - image: deepat1128/gitdemo1imagedt:1.0
        name: gitdemo1imagedt
        ports:
        - containerPort: 8050
        resources: {}
      - image: deepat1128/gitprojectimage:1.0
        name: gitprojectimage
        ports:
        - containerPort: 8060
        resources: {}
status: {}
~
```

deepa@ubuntu:~/devops/K8s$ kubectl exec -it mydeploy-77dc9c87c6-wvrqg – bash

deepa@ubuntu:~/devops/K8s$ kubectl exec -it mydeploy-77dc9c87c6-wvrqg -- sh

```
deepa@ubuntu:~/devops/K8s$ kubectl exec -it mydeploy-77dc9c87c6-wvrqg -- bash
Defaulted container "gitdemo1imagedt" out of: gitdemo1imagedt, gitprojectimage
OCI runtime exec failed: exec failed: unable to start container process: exec: "bash": executable file not found in $PATH: unknown
command terminated with exit code 127
deepa@ubuntu:~/devops/K8s$ kubectl exec -it mydeploy-77dc9c87c6-wvrqg -- sh
Defaulted container "gitdemo1imagedt" out of: gitdemo1imagedt, gitprojectimage
/app #
```

Port 8060, failed

```
Connecting to localhost (localhost)|::1|:8060 ... failed: Connection refused.
Connecting to localhost (localhost)|127.0.0.1|:8060 ... failed: Connection refused.
root@mydeploy-77dc9c87c6-wvrqg:/app# wget 10.244.0.34:8060/docs
--2025-10-10 10:11:31--  http://10.244.0.34:8060/docs
Connecting to 10.244.0.34:8060 ... failed: Connection refused.
```

From logs, tomcat started at 8086, so tried 8086

```
root@mydeploy-77dc9c87c6-wvrqg:/app# wget 10.244.0.34:8086/docs
--2025-10-10 10:13:27--  http://10.244.0.34:8086/docs
Connecting to 10.244.0.34:8086... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/json]
Saving to: 'docs'

docs                                    [ <=>                                                                   ]     385  --.-KB/s    in 0.007s

2025-10-10 10:13:27 (53.1 KB/s) - 'docs' saved [385]

root@mydeploy-77dc9c87c6-wvrqg:/app#
```

Here code 200 means, ok

Inter-pod networking is working inside the cluster.

```
root@mydeploy-77dc9c87c6-wvrqg:/app# wget 10.244.0.34:8080
--2025-10-10 10:19:39--  http://10.244.0.34:8080/
Connecting to 10.244.0.34:8080... connected.
HTTP request sent, awaiting response... 200
Length: 17 [text/plain]
Saving to: 'index.html'

index.html                              100%[=================================================================================>]      17  --.-KB/s    in 0s

2025-10-10 10:19:39 (686 KB/s) - 'index.html' saved [17/17]

root@mydeploy-77dc9c87c6-wvrqg:/app# wget 10.244.0.34:8080/demo1/h2
--2025-10-10 10:20:13--  http://10.244.0.34:8080/demo1/h2
Connecting to 10.244.0.34:8080... connected.
HTTP request sent, awaiting response... 302
Location: http://10.244.0.34:8080/demo1/h2/ [following]
--2025-10-10 10:20:13--  http://10.244.0.34:8080/demo1/h2/
Reusing existing connection to 10.244.0.34:8080.
HTTP request sent, awaiting response... 200
Length: 938 [text/html]
Saving to: 'h2'

h2                                      100%[=================================================================================>]     938  --.-KB/s    in 0s

2025-10-10 10:20:13 (79.7 MB/s) - 'h2' saved [938/938]

root@mydeploy-77dc9c87c6-wvrqg:/app#
```

```
deepa@ubuntu:~/devops/K8s$ cat mydeploy.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: mydeploy
  name: mydeploy
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mydeploy
  strategy: {}
  template:
    metadata:
      labels:
        app: mydeploy
    spec:
      containers:
      - image: deepat1128/gitdemo1imagedt:1.0
        name: gitdemo1imagedt
        ports:
        - containerPort: 8050
        resources: {}
      - image: deepat1128/gitprojectimage:1.0
        name: gitprojectimage
        ports:
        - containerPort: 8060
        resources: {}

status: {}
deepa@ubuntu:~/devops/K8s$
```

Done:

Container to container communication

Creating pod using deployment yaml file, declaratively

Creating one more container using yaml file.