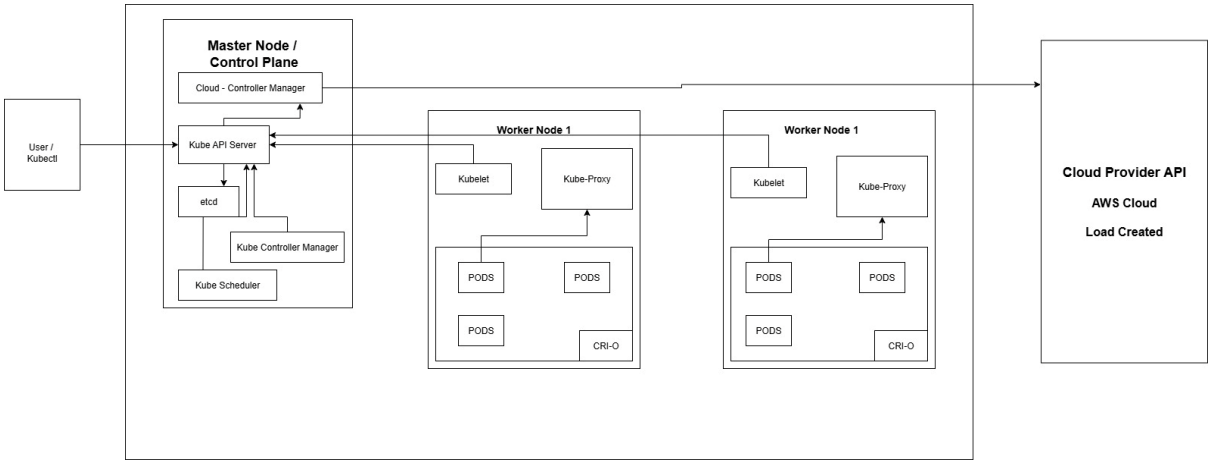
KUBERNETES ARCHITECTURE



KUBERNETES RESOURCES

POD

Definition: Pod is the smallest and most basic unit of deployment. It represents a single instance of a running process in your cluster. A Pod encapsulates one or more containers (such as Docker containers), storage resources, a unique network IP, and options that govern how the containers should run.

pod yaml file

```
apiVersion: v1
kind: Pod
metadata:
  name: ipl-pod
  labels:
    app: ipl
spec:
  containers:
    - name: ipl-container
      image: muralisocial123/ipl:latest
      ports:
        - containerPort: 3000
```

```
controlplane:~$ mkdir yashu
controlplane:~$ cd yashu
controlplane:~/yashu$ vi pod.yml
controlplane:~/yashu$ cat pod.yml
apiVersion: v1
kind: Pod
metadata:
  name: ipl-pod
  labels:
    app: ipl
spec:
  containers:
    - name: ipl-container
      image: muralisocial123/ipl:latest
      ports:
        - containerPort: 3000
```

Explanation of Pod Yaml file

1. **apiVersion: v1**

- Tells Kubernetes which version of API to use.
- v1 is used for core resources like Pod, Service, ConfigMap.

1. **kind: Pod**

- Tells Kubernetes what resource we are creating (here, a Pod).

2. **metadata:**

- Information about the Pod.
- name: Unique name of the Pod.
- labels: Key-value pairs to group/select Pods later.

3. **spec:**

- Defines the desired state of the Pod.

4. **containers:**

- List of containers inside the Pod.
- name: Name of the container.
- image: Docker image used to run the app.
- ports.containerPort: Port where container listens.

Commands to Work with Pods

1️⃣ Create a Pod

>> kubectl apply -f pod.yml

- Creates the Pod defined in your YAML file.

```
controlplane:~/yashu$ kubectl apply -f pod.yml
pod/ipl-pod created
controlplane:~/yashu$
```

---

2 Get Pods

>> kubectl get pods

- Lists all Pods in the current namespace.

```
controlplane:~/yashu$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
ipl-pod   1/1     Running   0          3m43s
```

>> kubectl get pods -o wide

- Shows extra details (Node name, IP, etc).

```
controlplane:~/yashu$ kubectl get pods -o wide
NAME      READY   STATUS    RESTARTS   AGE    IP             NODE     NOMINATED NODE   READINESS GATES
ipl-pod   1/1     Running   0          7m3s   192.168.1.4    node01   <none>           <none>
```

---

3 Describe a Pod

>> kubectl describe pod ipl-pod

- Shows detailed info: labels, IP, events, node, etc.

```
controlplane:~/yashu$ kubectl describe pod ipl-pod
Name:             ipl-pod
Namespace:        default
Priority:         0
Service Account:  default
Node:             node01/172.30.2.2
Start Time:       Mon, 01 Sep 2025 16:52:27 +0000
Labels:           app=ipl
Annotations:      cni.projectcalico.org/containerID: 7c0ca675dc8fb1eb55005601ea7a9964225bee7217fba73a833735be6f518cfd
                  cni.projectcalico.org/podIP: 192.168.1.4/32
                  cni.projectcalico.org/podIPs: 192.168.1.4/32
Status:           Running
IP:               192.168.1.4
IPs:
  IP:  192.168.1.4
Containers:
  ipl-container:
    Container ID:   containerd://8769aa0e38668a31738c6e8ab2a9d05013a0b86f1cb8504d3607589e2168e53e
    Image:          muralisocial123/ipl:latest
    Image ID:       docker.io/muralisocial123/ipl@sha256:77030f01a2375683dde323945e1752d7d7d145e6cbd4b431f4c1b4b32d8117a5
    Port:           3000/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Mon, 01 Sep 2025 16:52:55 +0000
    Ready:          True
    Restart Count:  0
```

```
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-8zpbp (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-8zpbp:
    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  8m16s  default-scheduler  Successfully assigned default/ipl-pod to node01
  Normal  Pulling    8m16s  kubelet            Pulling image "muralisocial123/ipl:latest"
  Normal  Pulled     7m49s  kubelet            Successfully pulled image "muralisocial123/ipl:latest" in 26.642s (26.642s includ
ing waiting). Image size: 445562584 bytes.
  Normal  Created    7m49s  kubelet            Created container: ipl-container
  Normal  Started    7m49s  kubelet            Started container ipl-container
```

Check Logs of a Pod

>> kubectl logs ipl-pod

- Prints logs from the main container inside the Pod.

```
controlplane:~/yashu$ kubectl logs ipl-pod

> ipl-nodejs-project@1.0.0 start
> node server.js

(node:19) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver v
ersion 4.0.0 and will be removed in the next major version
(Use `node --trace-warnings ...` to show where the warning was created)
(node:19) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Dr
iver version 4.0.0 and will be removed in the next major version
Server is running on http://localhost:3000
MongoDB connection error: MongooseServerSelectionError: getaddrinfo ENOTFOUND mongodb
    at _handleConnectionErrors (/app/node_modules/mongoose/lib/connection.js:897:11)
    at NativeConnection.openUri (/app/node_modules/mongoose/lib/connection.js:848:11) {
  reason: TopologyDescription {
    type: 'Unknown',
    servers: Map(1) { 'mongodb:27017' => [ServerDescription] },
    stale: false,
    compatible: true,
    heartbeatFrequencyMS: 10000,
    localThresholdMS: 15,
    setName: null,
    maxElectionId: null,
    maxSetVersion: null,
    commonWireVersion: 0,
    logicalSessionTimeoutMinutes: null
```

If multiple containers:

>> kubectl logs ipl-pod -c ipl-container

```
controlplane:~/yashu$ kubectl logs ipl-pod -c ipl-container

> ipl-nodejs-project@1.0.0 start
> node server.js

(node:19) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver v
ersion 4.0.0 and will be removed in the next major version
(Use `node --trace-warnings ...` to show where the warning was created)
(node:19) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Dr
iver version 4.0.0 and will be removed in the next major version
Server is running on http://localhost:3000
MongoDB connection error: MongooseServerSelectionError: getaddrinfo ENOTFOUND mongodb
    at _handleConnectionErrors (/app/node_modules/mongoose/lib/connection.js:897:11)
    at NativeConnection.openUri (/app/node_modules/mongoose/lib/connection.js:848:11) {
  reason: TopologyDescription {
    type: 'Unknown',
    servers: Map(1) { 'mongodb:27017' => [ServerDescription] },
    stale: false,
    compatible: true,
    heartbeatFrequencyMS: 10000,
    localThresholdMS: 15,
    setName: null,
    maxElectionId: null,
    maxSetVersion: null,
    commonWireVersion: 0,
```

---

5 Execute Command Inside Pod

>> kubectl exec -it ipl-pod -- /bin/bash

- Opens a shell inside the container.

```
controlplane:~/yashu$ kubectl exec -it ipl-pod -- /bin/bash
root@ipl-pod:/app#
```

>> kubectl exec -it ipl-pod -- ls /app

- Runs a command inside container.

```
controlplane:~/yashu$ kubectl exec -it ipl-pod -- ls /app
Dockerfile        docker-compose.yml  node_modules       package.json   public     services.yaml
deployments.yaml  mongo-service.yaml  package-lock.json  play-book.yml  server.js  uploads
controlplane:~/yashu$
```

---

6 Delete a Pod

>> kubectl delete pod ipl-pod

- Deletes the Pod.

```
controlplane:~/yashu$ kubectl delete pod ipl-pod
pod "ipl-pod" deleted
controlplane:~/yashu$
```

>> kubectl delete -f pod.yml

- Deletes Pod using YAML file.

```
controlplane:~/yashu$ kubectl delete -f pod.yml
pod "ipl-pod" deleted
```

---

7 Get Pod YAML (Generated)

>> kubectl get pod ipl-pod -o yaml

- Shows full YAML definition of a running Pod.

```
controlplane:~/yashu$ kubectl get pod ipl-pod -o yaml
apiVersion: v1
kind: Pod
metadata:
  annotations:
    cni.projectcalico.org/containerID: 4277846e0115c544a1fd5a39e5859b89c62c21f16c261b188e23cfe90d678f34
    cni.projectcalico.org/podIP: 192.168.1.6/32
    cni.projectcalico.org/podIPs: 192.168.1.6/32
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Pod","metadata":{"annotations":{},"labels":{"app":"ipl"},"name":"ipl-pod","namespace":"default"
},"spec":{"containers":[{"image":"muralisocial123/ipl:latest","name":"ipl-container","ports":[{"containerPort":3000}]}]}}
  creationTimestamp: "2025-09-01T17:15:40Z"
  generation: 1
  labels:
    app: ipl
  name: ipl-pod
  namespace: default
  resourceVersion: "5719"
  uid: 5031fcd5-d280-488d-b3ab-4807368c696d
spec:
  containers:
  - image: muralisocial123/ipl:latest
    imagePullPolicy: Always
    name: ipl-container
    ports:
    - containerPort: 3000
```

---

8 Port Forward Pod (Access App Locally)

>> kubectl port-forward pod/ipl-pod 8080:3000

- Maps **localhost:8080 → pod:3000**, so you can access app in browser.

```
controlplane:~/yashu$ kubectl port-forward pod/ipl-pod 8080:3000
Forwarding from 127.0.0.1:8080 -> 3000
Forwarding from [::1]:8080 -> 3000
```

---

9 Get Pod's IP

>> kubectl get pod ipl-pod -o wide

- Shows the internal Pod IP inside the cluster.

```
^Ccontrolplane:~/yashu$ kubectl get pod ipl-pod -o wide
NAME      READY   STATUS    RESTARTS   AGE     IP            NODE     NOMINATED NODE   READINESS GATES
ipl-pod   1/1     Running   0          4m43s   192.168.1.6   node01   <none>           <none>
```

---

10 Restart a Pod

Pods don't restart automatically when deleted (unless part of a Deployment/ReplicaSet).

>> kubectl delete pod ipl-pod

>> kubectl apply -f pod.yml

- This recreates it manually.

```
controlplane:~/yashu$ kubectl delete pod ipl-pod
pod "ipl-pod" deleted
controlplane:~/yashu$ kubectl apply -f pod.yml
pod/ipl-pod created
controlplane:~/yashu$
```

REPLICA SET

Definition: A **ReplicaSet** is a Kubernetes controller that ensures a specified number of identical Pod replicas are running at all times. If a Pod fails, is deleted, or an entire node fails, the ReplicaSet will automatically create new Pods to maintain the desired count. It's a key component for ensuring high availability and scalability of your applications.

Replica set yaml file

---

apiVersion: apps/v1

kind: ReplicaSet

metadata:

```
  name: ipl-replicaset
  labels:
    app: ipl
spec:
  replicas: 3
  selector:
    matchLabels:
      app: ipl
  template:
    metadata:
      labels:
        app: ipl
    spec:
      containers:
        - name: ipl-container
          image: muralisocial123/ipl:latest
          ports:
            - containerPort: 3000
```

```
controlplane:~$ vi replica.yml
controlplane:~$ cat replica.yml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: ipl-replicaset
  labels:
    app: ipl
spec:
  replicas: 3
  selector:
    matchLabels:
      app: ipl
  template:
    metadata:
      labels:
        app: ipl
    spec:
      containers:
        - name: ipl-container
          image: muralisocial123/ipl:latest
          ports:
            - containerPort: 3000
```

Explanation of Each Section

---

apiVersion: apps/v1

- ReplicaSet belongs to apps/v1 API group.

kind: ReplicaSet

- Defines that we are creating a ReplicaSet resource.

metadata:

- Info about the ReplicaSet.
- name: Unique name of RS.
- labels: Key-value pairs for grouping/identification.

spec:

- Desired state of RS.

replicas: 3

- Number of Pod copies that must always run.

selector:

- Defines how RS finds Pods to manage.
- matchLabels: app: ipl means RS will manage Pods with label app=ipl.

template:

- Pod definition (just like Pod YAML).
- ReplicaSet uses this template to create Pods if none exist.

Commands for ReplicaSet in Kubernetes

1 Create ReplicaSet

>> kubectl apply -f replicaset.yml

- Creates RS with 3 Pods (from replicas: 3).

```
controlplane:~$ kubectl apply -f replicaset.yml
replicaset.apps/ipl-replicaset created
controlplane:~$
```

---

2 Get ReplicaSets

>> kubectl get rs

- Shows list of all ReplicaSets and number of replicas.

```
controlplane:~$ kubectl get rs
NAME             DESIRED   CURRENT   READY   AGE
ipl-replicaset   3         3         3       74s
```

---

3 Get Pods Created by RS

>> kubectl get pods -l app=ipl

- Lists Pods managed by RS using label app=ipl.

```
controlplane:~$ kubectl get pods -l app=ipl
NAME                   READY   STATUS    RESTARTS   AGE
ipl-replicaset-m5d78   1/1     Running   0          2m15s
ipl-replicaset-tg2md   1/1     Running   0          2m15s
ipl-replicaset-v69gm   1/1     Running   0          2m15s
controlplane:~$
```

---

4 Describe ReplicaSet

>> kubectl describe rs ipl-replicaset

- Shows detailed info: selectors, events, how many replicas running.

```
controlplane:~$ kubectl describe rs ipl-replicaset
Name:         ipl-replicaset
Namespace:    default
Selector:     app=ipl
Labels:       app=ipl
Annotations:  <none>
Replicas:     3 current / 3 desired
Pods Status:  3 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=ipl
  Containers:
   ipl-container:
    Image:         muralisocial123/ipl:latest
    Port:          3000/TCP
    Host Port:     0/TCP
    Environment:   <none>
    Mounts:        <none>
  Volumes:         <none>
  Node-Selectors:  <none>
  Tolerations:     <none>
Events:
  Type    Reason            Age    From                   Message
  ----    ------            ----   ----                   -------
  Normal  SuccessfulCreate  3m33s  replicaset-controller  Created pod: ipl-replicaset-m5d78
  Normal  SuccessfulCreate  3m33s  replicaset-controller  Created pod: ipl-replicaset-tg2md
  Normal  SuccessfulCreate  3m33s  replicaset-controller  Created pod: ipl-replicaset-v69gm
```

---

5 Scale ReplicaSet

>> kubectl scale rs ipl-replicaset --replicas=5

- Increases Pods from 3 → 5.
- RS will immediately create 2 more Pods.

```
controlplane:~$ kubectl scale rs ipl-replicaset --replicas=5
replicaset.apps/ipl-replicaset scaled
controlplane:~$
```

---

6 Edit ReplicaSet

>> kubectl edit rs ipl-replicaset

- Opens editor to change YAML live (e.g., replicas count).

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"apps/v1","kind":"ReplicaSet","metadata":{"annotations":{},"labels":{"app":"ipl"},"name":"ipl-repl
  creationTimestamp: "2025-09-01T17:40:17Z"
  generation: 2
  labels:
    app: ipl
  name: ipl-replicaset
  namespace: default
  resourceVersion: "4614"
  uid: 0373203a-d41c-4658-8cea-fc0bfabb0e7b
spec:
  replicas: 5
  selector:
    matchLabels:
      app: ipl
  template:
    metadata:
```

---

7 Delete ReplicaSet

>> kubectl delete rs ipl-replicaset

- Deletes RS and all its Pods.

```
controlplane:~$ kubectl delete rs ipl-replicaset
replicaset.apps "ipl-replicaset" deleted
controlplane:~$
```

If you want to delete RS but **keep Pods**:

>> kubectl delete rs ipl-replicaset --cascade=orphan

```
controlplane:~$ kubectl delete rs ipl-replicaset --cascade=orphan
replicaset.apps "ipl-replicaset" deleted
controlplane:~$
```

---

8 Check Events

>> kubectl describe rs ipl-replicaset | grep Events -A5

- Shows events like Pod creation/deletion by RS.

```
controlplane:~$ kubectl scale rs ipl-replicaset --replicas=5
replicaset.apps/ipl-replicaset scaled
controlplane:~$ kubectl describe rs ipl-replicaset | grep Events -A5
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  34s   replicaset-controller  Created pod: ipl-replicaset-t2592
  Normal  SuccessfulCreate  34s   replicaset-controller  Created pod: ipl-replicaset-lhd8v
controlplane:~$
```

---

9 Get RS YAML

>> kubectl get rs ipl-replicaset -o yaml

- Shows full YAML definition of running RS.

```
controlplane:~$ kubectl get rs ipl-replicaset -o yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"apps/v1","kind":"ReplicaSet","metadata":{"annotations":{},"labels":{"app":"ipl"},"name":"ipl-repl
icaset","namespace":"default"},"spec":{"replicas":3,"selector":{"matchLabels":{"app":"ipl"}},"template":{"metadata":{"
labels":{"app":"ipl"}},"spec":{"containers":[{"image":"muralisocial123/ipl:latest","name":"ipl-container","ports":[{"c
ontainerPort":3000}]}]}}}}
  creationTimestamp: "2025-09-01T17:50:52Z"
  generation: 2
  labels:
    app: ipl
  name: ipl-replicaset
  namespace: default
  resourceVersion: "5296"
  uid: ee76717d-88a5-4b88-be04-b30b0dfc0cf4
spec:
  replicas: 5
  selector:
    matchLabels:
      app: ipl
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: ipl
    spec:
      containers:
      - image: muralisocial123/ipl:latest
        imagePullPolicy: Always
        name: ipl-container
        ports:
        - containerPort: 3000
          protocol: TCP
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30
status:
  availableReplicas: 5
  fullyLabeledReplicas: 5
  observedGeneration: 2
  readyReplicas: 5
  replicas: 5
controlplane:~$
```

DEPLOYMENT

Definition: A **Deployment** is a higher-level abstraction on top of ReplicaSets. While a ReplicaSet ensures a specified number of Pods are running, a Deployment provides declarative updates for Pods and ReplicaSets. It allows you to:

1. **Define the desired state** of your application (e.g., image, number of replicas).
2. **Automatically create and manage ReplicaSets** underneath.

3. **Perform rolling updates** to your application without downtime.
4. **Rollback** to previous versions if an update goes wrong.

In essence, a Deployment manages ReplicaSets, and ReplicaSets manage Pods. You typically interact with Deployments, and Kubernetes handles the underlying ReplicaSet and Pod creation and management.

Deployment yaml file

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ipl-deployment
  labels:
    app: ipl
spec:
  replicas: 3
  selector:
    matchLabels:
      app: ipl
  template:
    metadata:
      labels:
        app: ipl
    spec:
      containers:
        - name: ipl-container
          image: muralisocial123/ipl:latest
          ports:
            - containerPort: 3000
```

```
controlplane:~$ vi deploment.yml
controlplane:~$ cat deploment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ipl-deployment          # Name of Deployment
  labels:
    app: ipl
spec:
  replicas: 3                   # Number of Pods to run
  selector:
    matchLabels:
      app: ipl
  template:                     # Pod template (same as Pod definition)
    metadata:
      labels:
        app: ipl
    spec:
      containers:
        - name: ipl-container
          image: muralisocial123/ipl:latest
          ports:
            - containerPort: 3000
```

Explanation of Each Section

1. **apiVersion: apps/v1**
    - Deployment belongs to the apps/v1 API group.
2. **kind: Deployment**
    - Defines that we are creating a Deployment resource.
3. **metadata:**
    - name: Unique name of Deployment.
    - labels: Key-value identifiers.
4. **spec:**
    - Desired state of Deployment.
5. **replicas: 3**
    - Ensures 3 Pods are running at all times.
6. **selector:**
    - Used by Deployment to find Pods to manage.
    - Must match Pod template labels.
7. **template:**
    - Defines Pod specification (like Pod YAML).
    - Includes labels, containers, images, and ports.

Commands for Deployment in Kubernetes

1 Create Deployment

>> kubectl apply -f deployment.yml

- Creates a Deployment and underlying ReplicaSet + Pods.

```
controlplane:~$ kubectl apply -f deployment.yml
deployment.apps/ipl-deployment created
controlplane:~$
```

---

2 Get Deployments

>> kubectl get deployments

- Lists Deployments with replicas count.

```
controlplane:~$ kubectl get deployments
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
ipl-deployment   3/3     3            3           54s
controlplane:~$
```

---

3 Get ReplicaSets (created by Deployment)

>> kubectl get rs

- Shows ReplicaSets managed by the Deployment.

```
controlplane:~$ kubectl get rs
NAME                       DESIRED   CURRENT   READY   AGE
ipl-deployment-dc979677f   3         3         3       102s
```

---

4 Get Pods (created by ReplicaSet under Deployment)

>> kubectl get pods -l app=ipl

- Lists all Pods controlled by Deployment.

```
controlplane:~$ kubectl get pods -l app=ipl
NAME                             READY   STATUS    RESTARTS   AGE
ipl-deployment-dc979677f-dv85k   1/1     Running   0          3m14s
ipl-deployment-dc979677f-krlst   1/1     Running   0          3m14s
ipl-deployment-dc979677f-mstjt   1/1     Running   0          3m14s
controlplane:~$
```

---

5 Describe Deployment

>> kubectl describe deployment ipl-deployment

- Shows detailed info about the Deployment, strategy, events, and Pods.

```
controlplane:~$ kubectl describe deployment ipl-deployment
Name:                   ipl-deployment
Namespace:              default
CreationTimestamp:      Mon, 01 Sep 2025 18:14:21 +0000
Labels:                 app=ipl
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=ipl
Replicas:               3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=ipl
  Containers:
   ipl-container:
    Image:         muralisocial123/ipl:latest
    Port:          3000/TCP
    Host Port:     0/TCP
    Environment:   <none>
    Mounts:        <none>
  Volumes:         <none>
  Node-Selectors:  <none>
  Tolerations:     <none>
```

```
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   ipl-deployment-dc979677f (3/3 replicas created)
Events:
  Type    Reason             Age   From                   Message
  ----    ------             ----  ----                   -------
  Normal  ScalingReplicaSet  4m5s  deployment-controller  Scaled up replica set ipl-deployment-dc979677f from 0 to 3
controlplane:~$
```

---

6 Scale Deployment

>> kubectl scale deployment ipl-deployment --replicas=5

- Increases replicas from 3 → 5.

```
controlplane:~$ kubectl scale deployment ipl-deployment --replicas=5
deployment.apps/ipl-deployment scaled
```

---

7 Update Deployment (e.g., new image)

>> kubectl set image deployment/ipl-deployment ipl-container=muralisocial123/ipl:v2

- Updates container image (rolling update with zero downtime).

```
controlplane:~$ kubectl set image deployment/ipl-deployment ipl-container=muralisocial123/ipl:v2
deployment.apps/ipl-deployment image updated
```

---

8 Check Rollout Status

>> kubectl rollout status deployment ipl-deployment

- Shows progress of deployment update.

```
controlplane:~$ kubectl rollout status deployment ipl-deployment
Waiting for deployment "ipl-deployment" rollout to finish: 3 out of 5 new replicas have been updated...
```

---

9 Rollback Deployment

>> kubectl rollout undo deployment ipl-deployment

- Rolls back to previous version if new update fails.

```
controlplane:~$ kubectl rollout undo deployment ipl-deployment
deployment.apps/ipl-deployment rolled back
```

---

10 Edit Deployment

>> kubectl edit deployment ipl-deployment

- Opens live YAML for editing.

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "4"
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"apps/v1","kind":"Deployment","metadata":{"annotations":{},"labels":{"app":"ipl"},"name":"ipl-deployment","na
  creationTimestamp: "2025-09-01T18:14:21Z"
  generation: 5
  labels:
    app: ipl
  name: ipl-deployment
  namespace: default
  resourceVersion: "5063"
  uid: 5f81f0cb-f97a-4431-8142-1e118f098338
spec:
  progressDeadlineSeconds: 600
  replicas: 5
  revisionHistoryLimit: 10
  selector:
    matchLabels:
"/tmp/kubectl-edit-3155683960.yaml" 72L, 2322B
```

---

11 Delete Deployment

>> kubectl delete deployment ipl-deployment

- Deletes Deployment, ReplicaSet, and Pods.

```
controlplane:~$ kubectl delete deployment ipl-deployment
deployment.apps "ipl-deployment" deleted
```

1️⃣2️⃣ Get YAML of Deployment

>> kubectl get deployment ipl-deployment -o yaml

- Shows the live YAML configuration of the Deployment.

```
controlplane:~$ kubectl get deployment ipl-deployment -o yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"apps/v1","kind":"Deployment","metadata":{"annotations":{},"labels":{"app":"ipl"},"name":"ipl-deployment","na
mespace":"default"},"spec":{"replicas":3,"selector":{"matchLabels":{"app":"ipl"}},"template":{"metadata":{"labels":{"app":"ipl"}}
,"spec":{"containers":[{"image":"muralisocial123/ipl:latest","name":"ipl-container","ports":[{"containerPort":3000}]}]}}}}
  creationTimestamp: "2025-09-01T18:30:42Z"
  generation: 1
  labels:
    app: ipl
  name: ipl-deployment
  namespace: default
  resourceVersion: "5563"
  uid: 8a7134d1-ea95-4974-a66f-71b17291668f
spec:
  progressDeadlineSeconds: 600
  replicas: 3
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: ipl
  strategy:
```

```
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: ipl
    spec:
      containers:
      - image: muralisocial123/ipl:latest
        imagePullPolicy: Always
        name: ipl-container
        ports:
        - containerPort: 3000
          protocol: TCP
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30
```

```
status:
  availableReplicas: 3
  conditions:
  - lastTransitionTime: "2025-09-01T18:30:45Z"
    lastUpdateTime: "2025-09-01T18:30:45Z"
    message: Deployment has minimum availability.
    reason: MinimumReplicasAvailable
    status: "True"
    type: Available
  - lastTransitionTime: "2025-09-01T18:30:42Z"
    lastUpdateTime: "2025-09-01T18:30:45Z"
    message: ReplicaSet "ipl-deployment-dc979677f" has successfully progressed.
    reason: NewReplicaSetAvailable
    status: "True"
    type: Progressing
  observedGeneration: 1
  readyReplicas: 3
  replicas: 3
  updatedReplicas: 3
```

SERVICE

Definition: Service is an abstract way to expose an application running on a set of Pods as a network service. Services provide a stable IP address and DNS name for your Pods, even if the underlying Pods are constantly changing (e.g., due to scaling, updates, or failures).

Key problems Services solve:

1. **Pod Volatility:** Pods are ephemeral. They are created, deleted, and restarted, and their IP addresses change. Services provide a persistent endpoint.
2. **Load Balancing:** A Service can distribute network traffic across multiple Pods (replicas) that belong to it, acting as a simple load balancer.
3. **Discovery:** Services make it easy for other applications (or users) to discover and connect to your application, either internally within the cluster or externally.

Types of Services in Kubernetes

1. **ClusterIP** (default) → Exposes app only inside the cluster.
2. **NodePort** → Exposes app outside cluster using <NodeIP>:<NodePort>.
3. **LoadBalancer** → Creates external load balancer (on cloud providers like AWS, GCP, Azure).
4. **ExternalName** → Maps service to an external DNS name.

ClusterIP Service YAML

```
apiVersion: v1
kind: Service
metadata:
  name: ipl-clusterip-svc
  namespace: default
spec:
  selector:
    app: ipl-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
  type: ClusterIP
```

```
controlplane:~$ vi service_Clusterip.yml
controlplane:~$ cat service_Clusterip.yml
apiVersion: v1
kind: Service
metadata:
  name: ipl-clusterip-svc
  namespace: default
spec:
  selector:
    app: ipl-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
  type: ClusterIP
```

NodePort Service YAML

apiVersion: v1

kind: Service

metadata:

  name: ipl-nodeport-svc

  namespace: default

spec:

  selector:

    app: ipl-app

  ports:

    - protocol: TCP

      port: 80

      targetPort: 3000

      nodePort: 30080   # Must be between 30000–32767

  type: NodePort

```
controlplane:~$ cat service_Nodeport.yml
apiVersion: v1
kind: Service
metadata:
  name: ipl-nodeport-svc
  namespace: default
spec:
  selector:
    app: ipl-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
      nodePort: 30080   # Must be between 30000-32767
  type: NodePort
```

LoadBalancer Service YAML

```
apiVersion: v1
kind: Service
metadata:
  name: ipl-loadbalancer-svc
  namespace: default
spec:
  selector:
    app: ipl-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
  type: LoadBalancer
```

```
controlplane:~$ vi service.yml
controlplane:~$ cat service.yml
apiVersion: v1
kind: Service
metadata:
  name: ipl-loadbalancer-svc
  namespace: default
spec:
  selector:
    app: ipl-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 3000
  type: LoadBalancer
```

ExternalName Service YAML

```
apiVersion: v1
kind: Service
metadata:
  name: ipl-externalname-svc
  namespace: default
spec:
  type: ExternalName
  externalName: example.com
```

```
controlplane:~$ vi service_exte.yml
controlplane:~$ cat service_exte.yml
apiVersion: v1
kind: Service
metadata:
  name: ipl-externalname-svc
  namespace: default
spec:
  type: ExternalName
  externalName: example.com
controlplane:~$
```

Explanation of YAML fields

- **apiVersion**: API version (v1 for Service).
- **kind**: Type of object (Service).
- **metadata.name**: Name of the Service.
- **metadata.namespace**: Namespace (default if not given).
- **spec.selector**: Labels of Pods to which traffic will be sent.
- **spec.ports.port**: Port exposed by Service.
- **spec.ports.targetPort**: Port on Pod containers.
- **spec.ports.nodePort**: Port on Node (only for NodePort).
- **spec.type**: Defines type (ClusterIP, NodePort, LoadBalancer, ExternalName).

Commands for Services in Kubernetes

1. **Create Service**

>> kubectl apply -f service.yml

→Creates the Service from YAML file.

```
controlplane:~$ kubectl apply -f service.yml
deployment.apps/ipl-deployment created
service/ipl-clusterip-svc created
service/ipl-nodeport-svc created
service/ipl-loadbalancer-svc created
service/ipl-externalname-svc created
```

2. List all Services

>> kubectl get svc

→Shows all services with type, cluster IP, external IP, and ports.

```
controlplane:~$ kubectl get svc
NAME                  TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
ipl-clusterip-svc     ClusterIP      10.99.120.254    <none>        80/TCP         26s
ipl-externalname-svc  ExternalName   <none>           example.com   <none>         26s
ipl-loadbalancer-svc  LoadBalancer   10.103.171.200   <pending>     80:32037/TCP   26s
ipl-nodeport-svc      NodePort       10.103.1.184     <none>        80:30080/TCP   26s
kubernetes            ClusterIP      10.96.0.1        <none>        443/TCP        13d
```

3. Describe a Service

>> kubectl describe svc ipl-nodeport-svc

→Gives detailed info (selectors, endpoints, events).

```
controlplane:~$ kubectl describe svc ipl-nodeport-svc
Name:                     ipl-nodeport-svc
Namespace:                default
Labels:                   <none>
Annotations:              <none>
Selector:                 app=ipl-app
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.103.1.184
IPs:                      10.103.1.184
Port:                     <unset>  80/TCP
TargetPort:               3000/TCP
NodePort:                 <unset>  30080/TCP
Endpoints:                192.168.1.5:3000,192.168.1.4:3000,192.168.0.4:3000
Session Affinity:         None
External Traffic Policy:  Cluster
Internal Traffic Policy:  Cluster
Events:                   <none>
```

4. Delete a Service

>> kubectl delete svc ipl-nodeport-svc

→Deletes the Service.

```
^Ccontrolplane:~$ kubectl delete svc ipl-nodeport-svc
service "ipl-nodeport-svc" deleted
```

5. Access Pods via Service

→ClusterIP → curl http://<ClusterIP>:80 (inside cluster).

→NodePort → http://<NodeIP>:30080.

→LoadBalancer → http://<External-IP>.

→ExternalName → resolves DNS (e.g., example.com).

6. Get Endpoints (Pods linked to Service)

>> kubectl get endpoints ipl-clusterip-svc

→Shows which Pods are selected by the Service.

```
controlplane:~$ kubectl get endpoints ipl-clusterip-svc
Warning: v1 Endpoints is deprecated in v1.33+; use discovery.k8s.io/v1 EndpointSlice
NAME                ENDPOINTS                                                  AGE
ipl-clusterip-svc   192.168.0.4:3000,192.168.1.4:3000,192.168.1.5:3000   10m
```

7. Port-forward to Service

>> kubectl port-forward svc/ipl-clusterip-svc 8080:80

→Access Service locally at http://localhost:8080.

```
controlplane:~$ kubectl get endpoints ipl-clusterip-svc
Warning: v1 Endpoints is deprecated in v1.33+; use discovery.k8s.io/v1 EndpointSlice
NAME                ENDPOINTS                                                  AGE
ipl-clusterip-svc   192.168.0.4:3000,192.168.1.4:3000,192.168.1.5:3000   10m
controlplane:~$ kubectl port-forward svc/ipl-clusterip-svc 8080:80
Forwarding from 127.0.0.1:8080 -> 3000
Forwarding from [::1]:8080 -> 3000
```