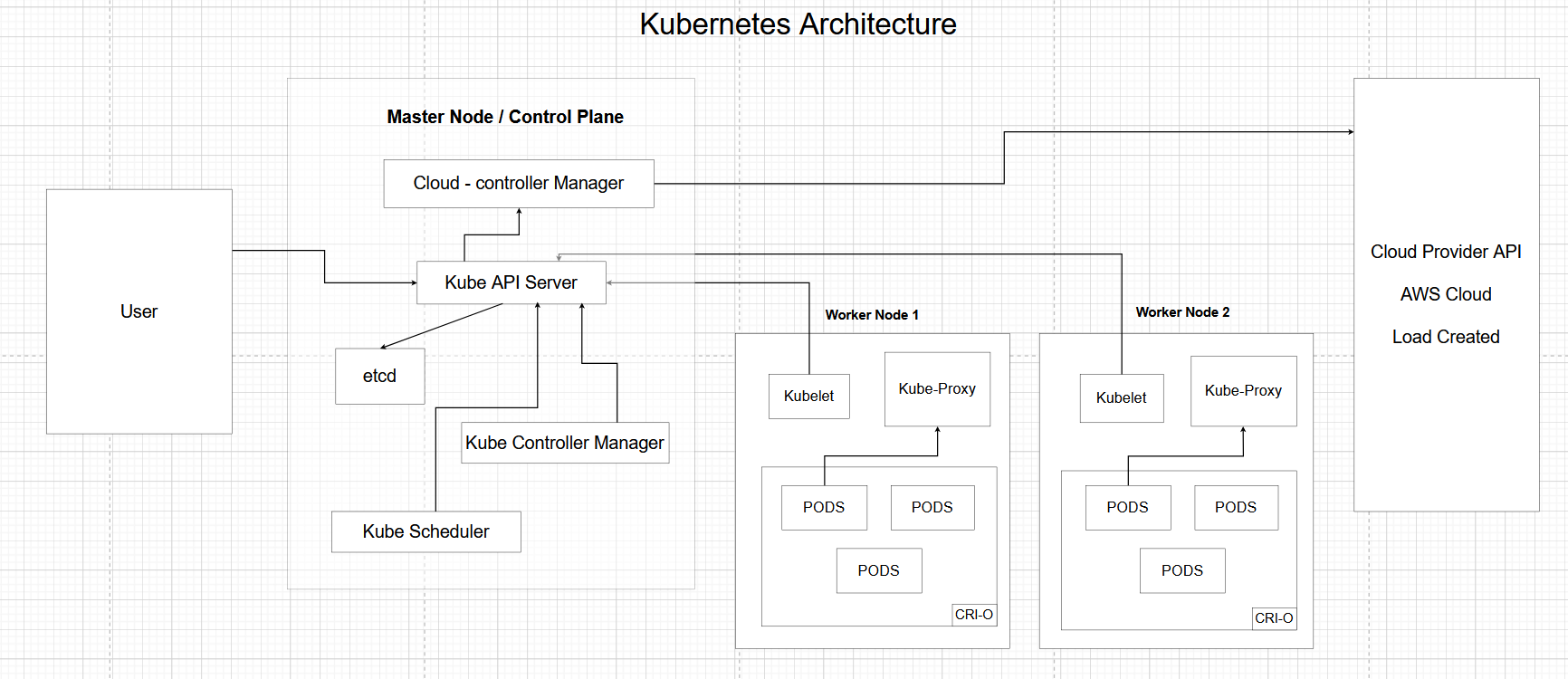
# **Kubernetes**



**Pod**

Pods are the smallest deployable units of computing that you can create and manage in Kubernetes.

pod.yml

apiVersion: v1

kind: Pod

metadata:

name: bimal-pod # pod name

namespace: bimal

labels:

app: bimal-nginx

spec:

containers:

- name: bimal-nginx

image: nginx:latest

ports:

- containerPort: 80

API version: k8s API used.

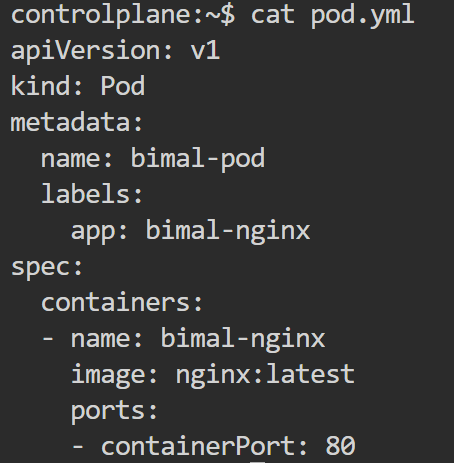
Kind: define type of object.

Metadata: stores information about the object.

Spec: defines the state of object or resource.

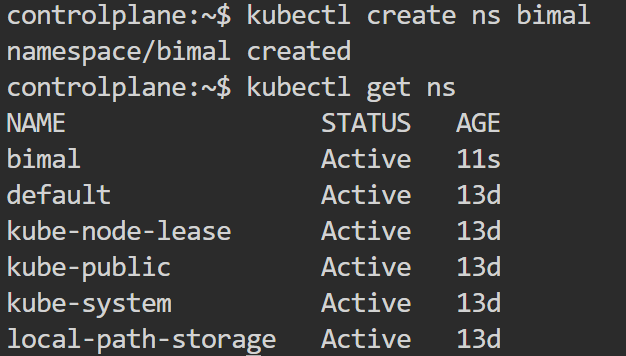
Selector: to find the object with specific labels.

Template: blueprint for creating each pod.



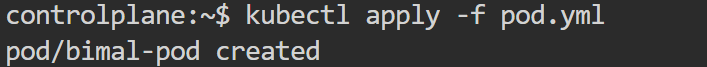
kubectl create ns bimal - to create namespace

kubectl get ns – to get all namespaces

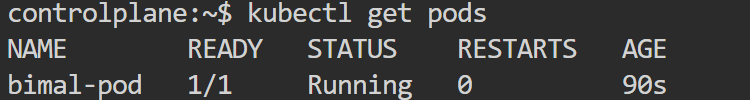
****

**Pod commands**

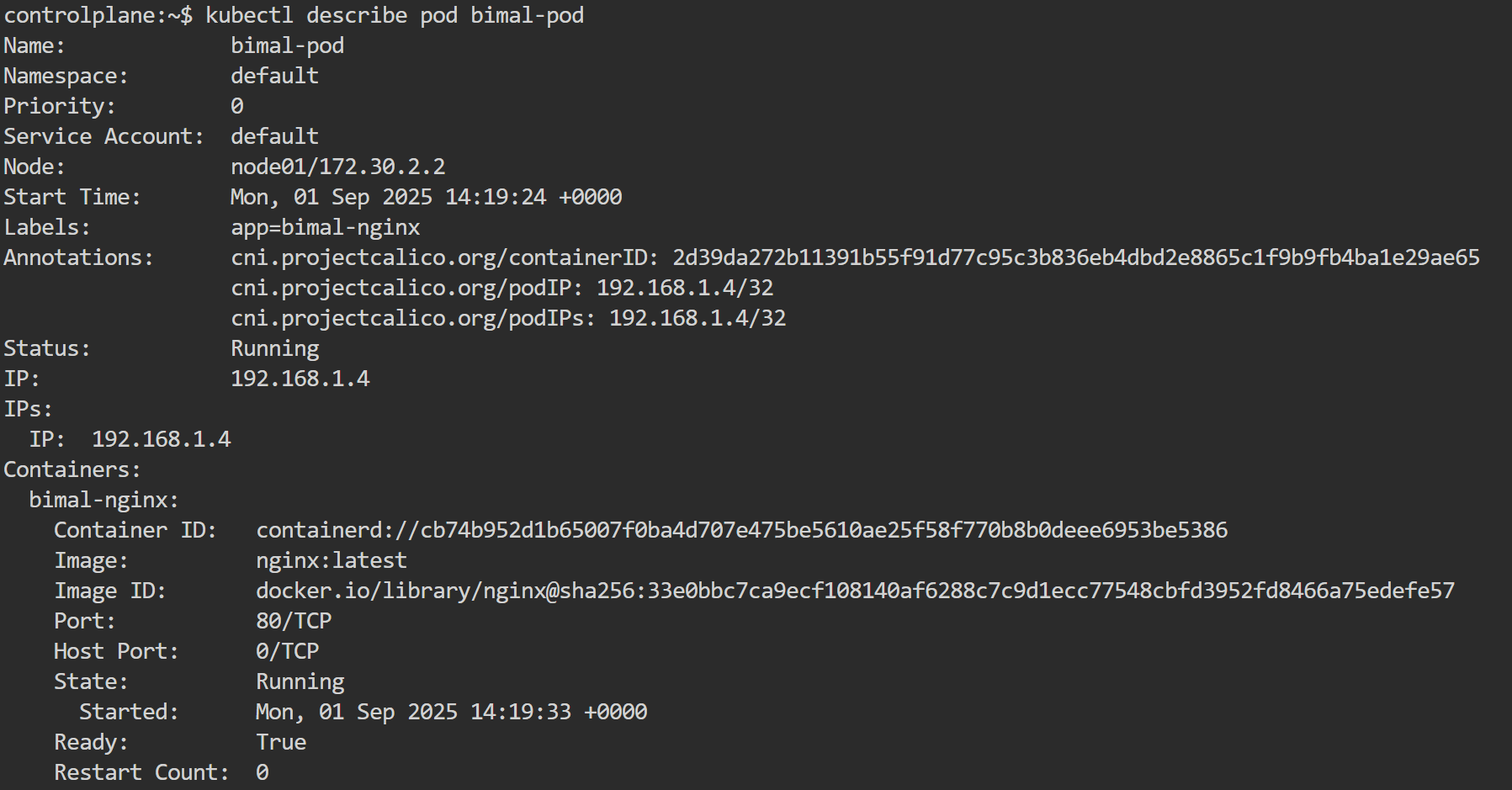
kubectl apply -f <filename> - applies configuration



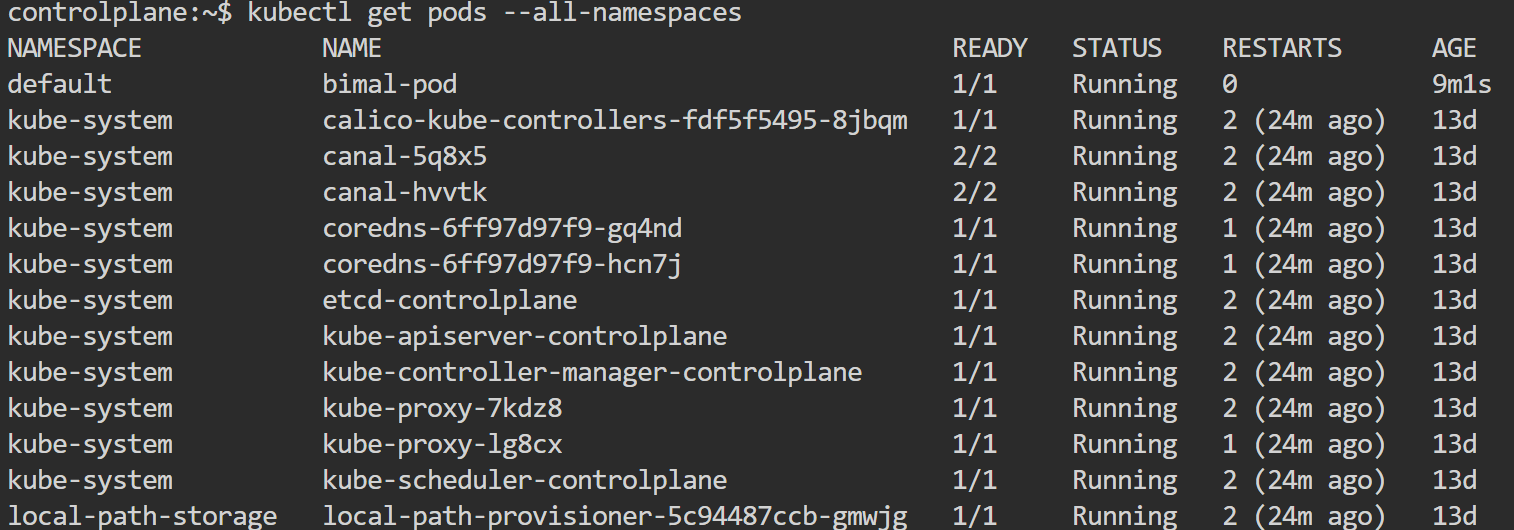
kubectl get pods - List all pods



kubectl describe pod <pod-name> - Get detailed information about pods



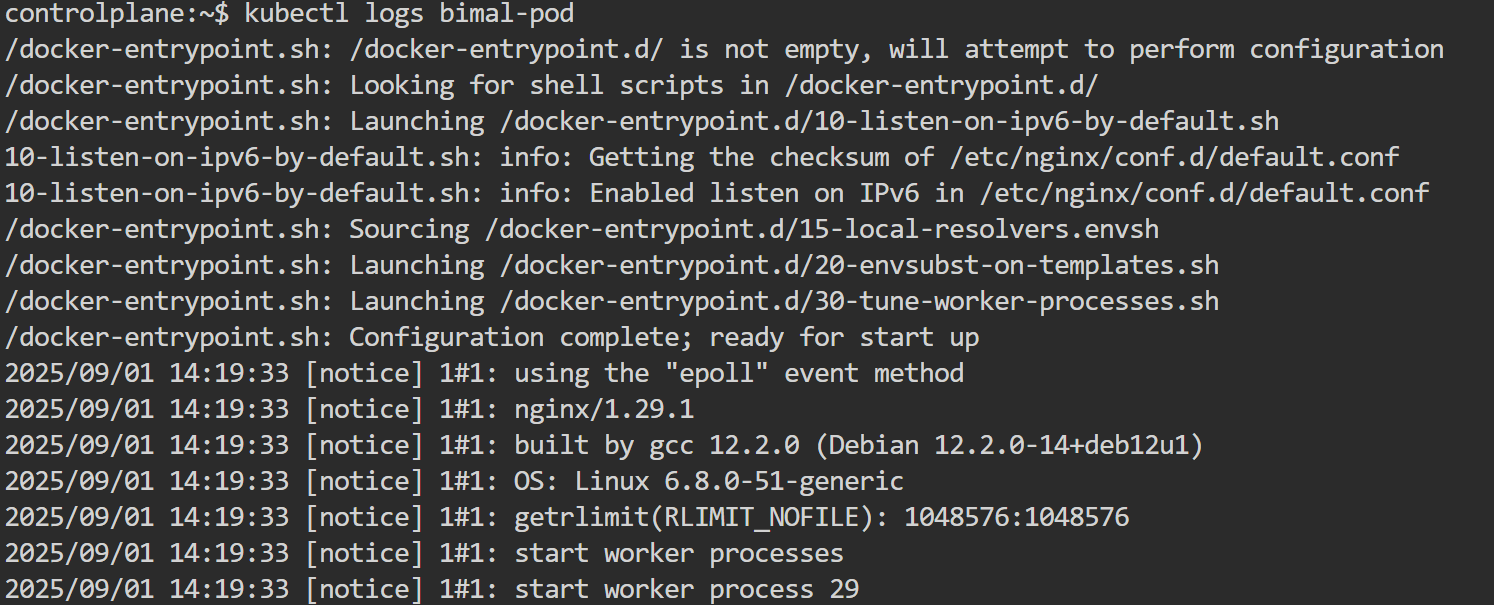
kubectl get pods --all-namespaces - List all Pods across all namespaces



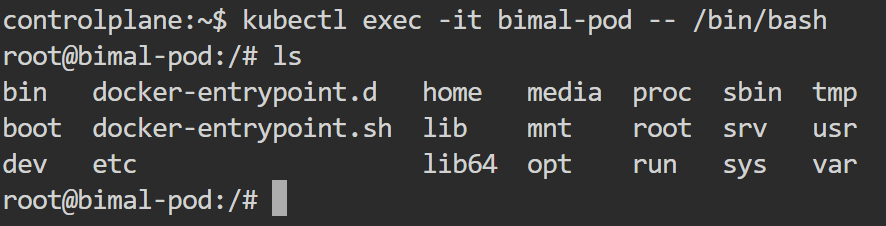
kubectl get pods -o wide - List Pods with more details

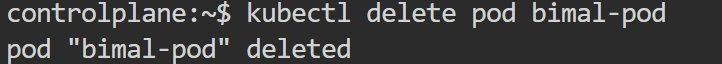


kubectl logs <pod-name> - View logs from a Pod



kubectl exec -it <pod-name> -- /bin/bash - Execute a shell inside a Pod



kubectl delete pod <pod-name> - Delete a specific Pod

**Deployment**

It defines how to create or modify instances of pods that hold a containerized application.

deployment.yml

apiVersion: apps/v1 # API group

kind: Deployment # type of object

metadata: # info to identify the deployment

name: bimal-deployment

namespace: bimal

spec: # describe the state of app

replicas: 3 # create replicas of the pod

selector: # to find the pods

matchLabels:

app: my-pod

template: # blueprint for creating each pod

metadata:

labels:

app: my-pod

spec: # Specification for pods

containers: # container for pod

- name: c-1 # container name

image: muralisocial123/ipl-app:v1

ports:

- containerPort: 3000 # listen for traffic on this port

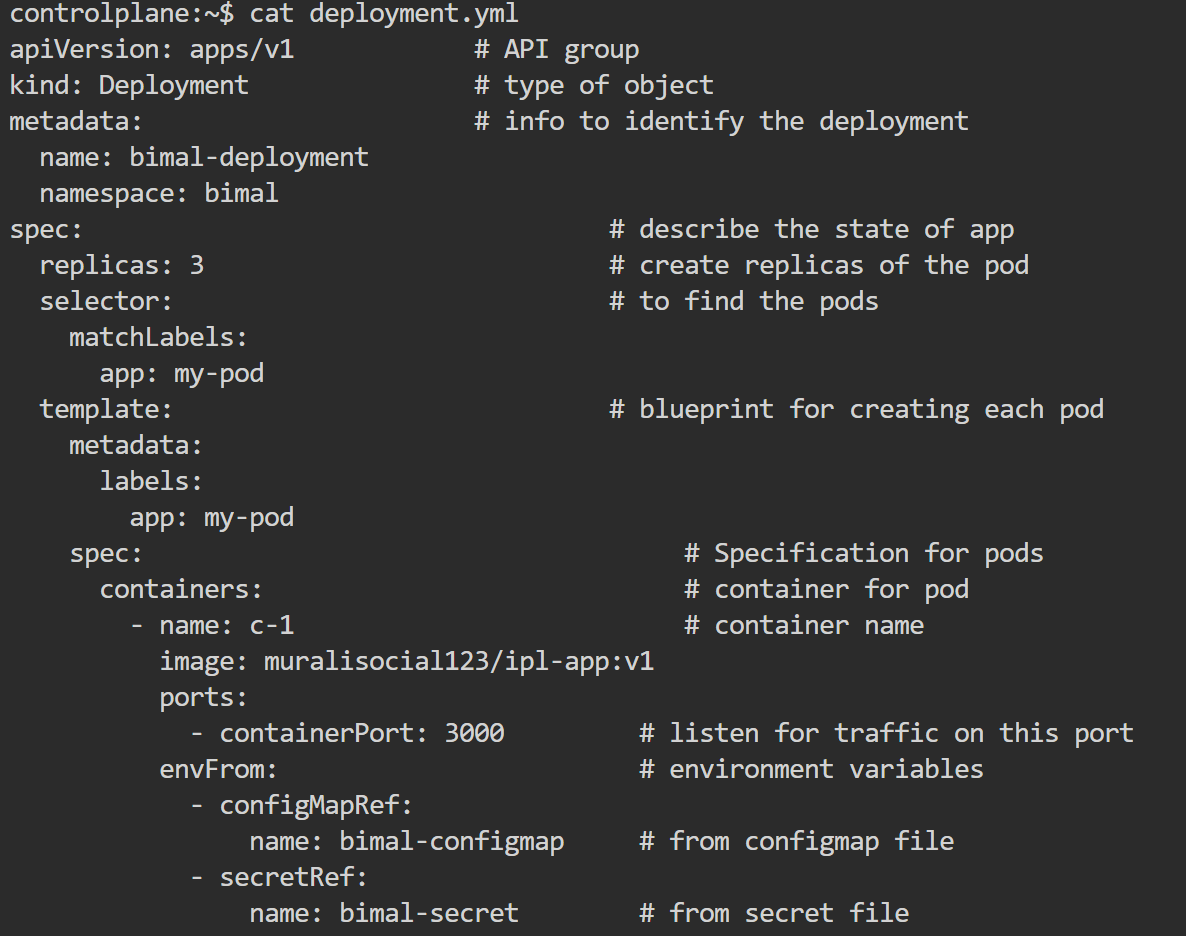
envFrom: # inject environment variables

- configMapRef:

name: bimal-configmap # from configmap file

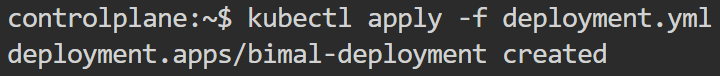
- secretRef:

name: bimal-secret # from secret file

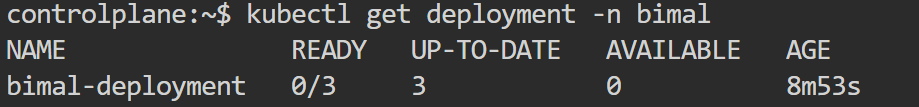


**Deployment commands**

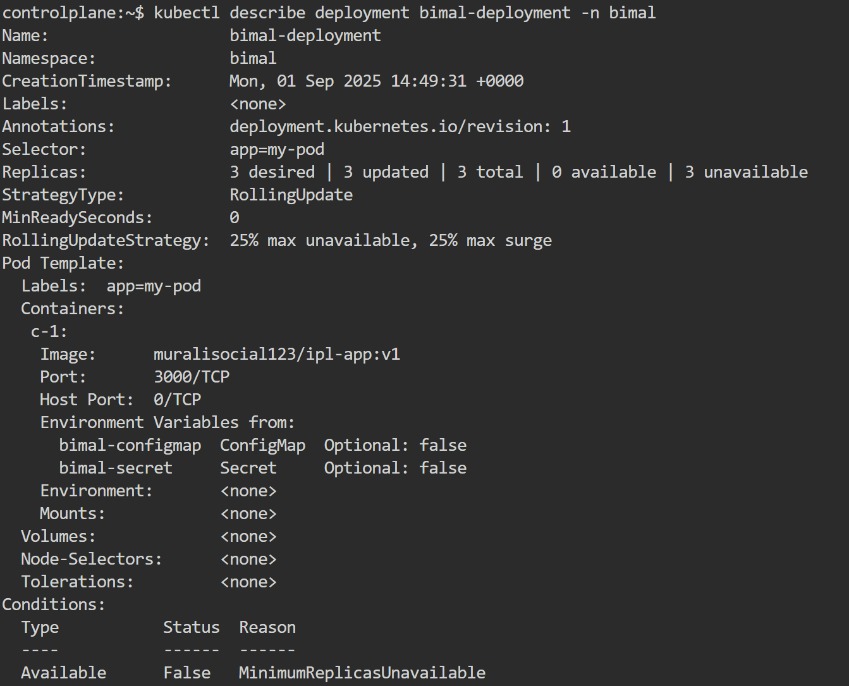
kubectl apply -f <your-deployment-file.yaml> - Apply a configuration



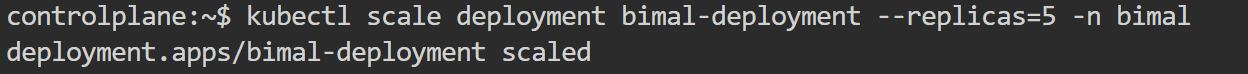
kubectl get deployments - List all Deployments



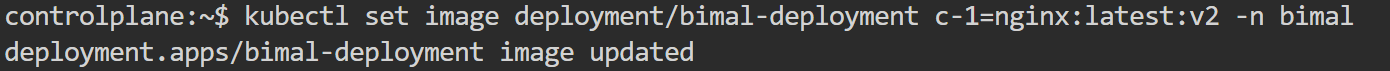
kubectl describe deployment <deployment-name> - Get detailed information about a deployment



kubectl scale deployment <deployment-name> --replicas=<number> - Scale a Deployment manually

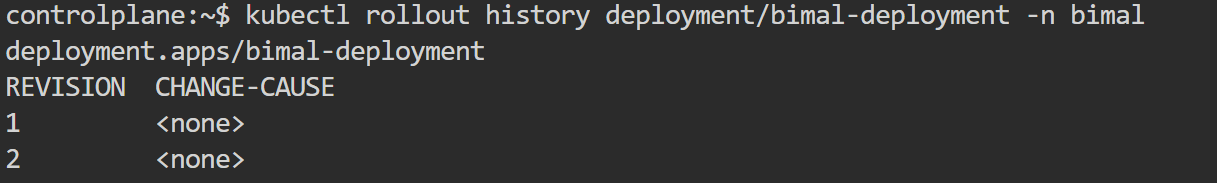


kubectl set image deployment <deployment-name> <container-name>=<new-image>:<new-tag> - Roll out a new version of an image

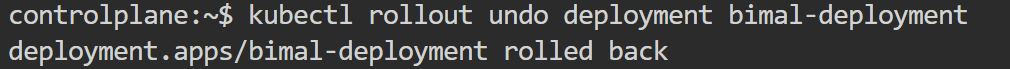


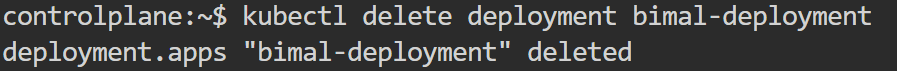
kubectl rollout status deployment <deployment-name> - Check the status of a rollout

kubectl rollout history deployment <deployment-name> - View the history of a Deployment



kubectl rollout undo deployment <deployment-name> - Roll back to the previous version



kubectl delete deployment <deployment-name> - Delete a Deployment

**Replica Set**

A ReplicaSet is a controller that ensures a specified number of identical Pod replicas are running at all times.

replica.yml

apiVersion: apps/v1

kind: ReplicaSet

metadata:

name: bimal-replica

namespace: bimal

labels:

app: bimal-nginx-rs

spec:

replicas: 3 # number of pod replicas

selector:

matchLabels:

app: bimal-nginx-rs

template:

metadata:

labels:

app: bimal-nginx-rs

spec:

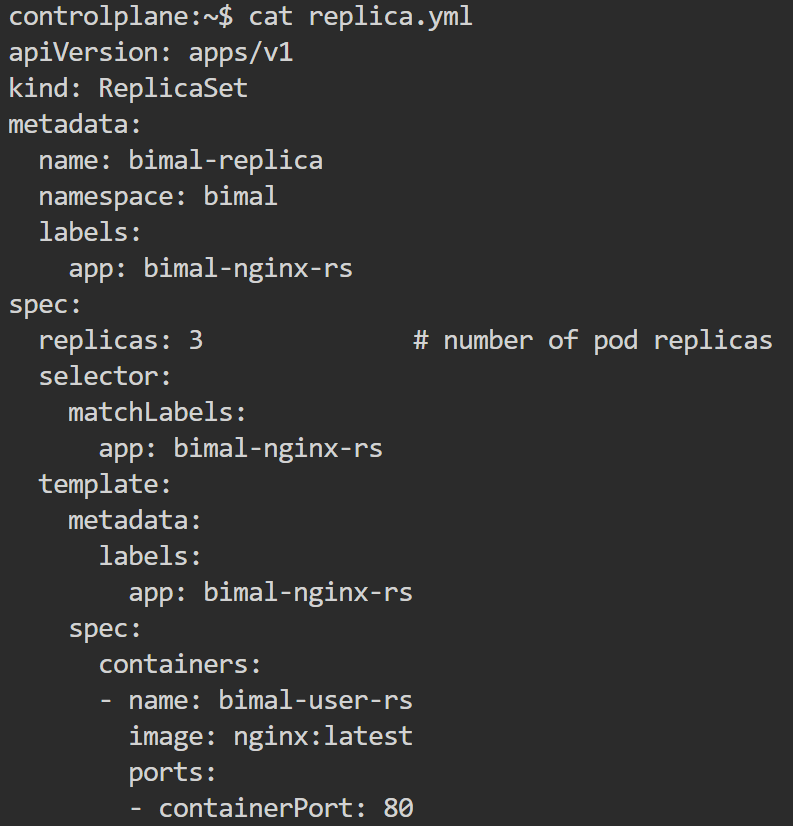
containers:

- name: bimal-user-rs

image: nginx:latest

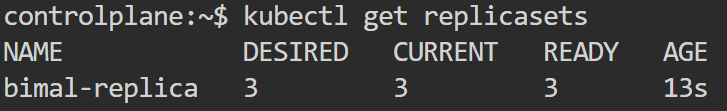
ports:

- containerPort: 80

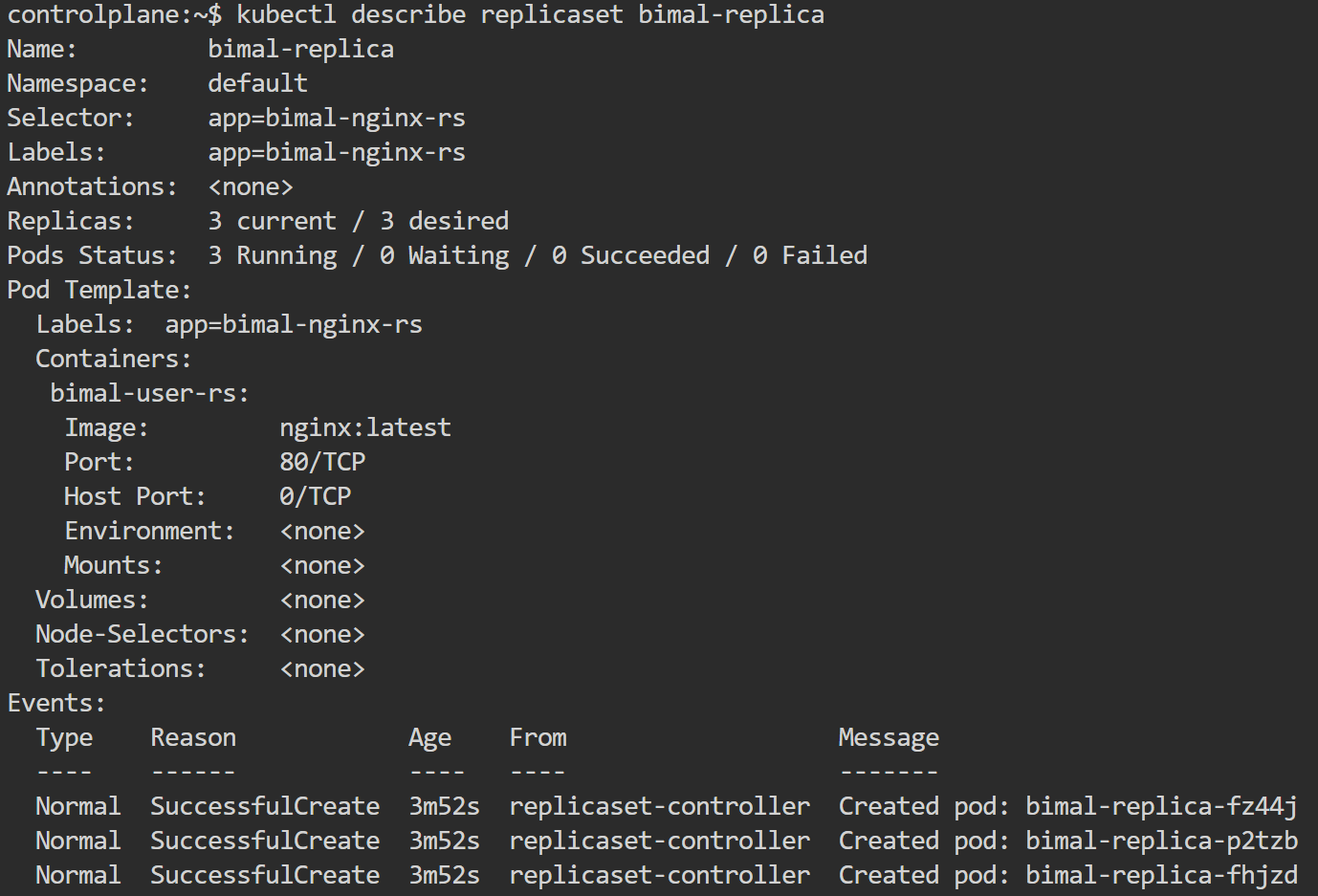


**Replica set command**

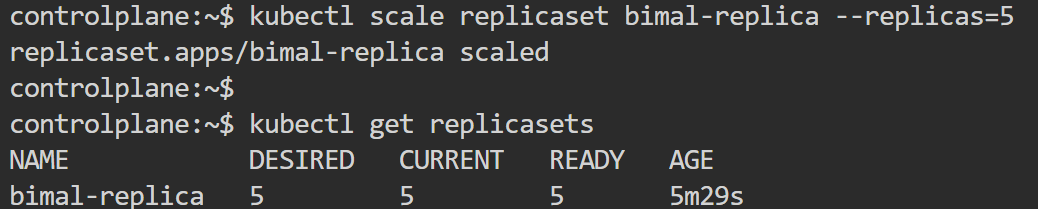
kubectl get replicasets - List all ReplicaSets



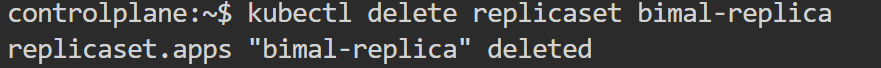
kubectl describe replicaset <replicaset-name> - Get detailed information about a ReplicaSet



kubectl scale replicaset <replicaset-name> --replicas=<number> - Scale a ReplicaSet manually



kubectl delete replicaset <replicaset-name> - Delete a ReplicaSet



**Service**

**Cluster IP:**

* It will expose the app within a cluster only but you can't access app outside the cluster.
* It is for production use.

apiVersion: v1

kind: Service

metadata:

name: bimal-service

namespace: bimal

spec:

selector:

app: bimal-pod

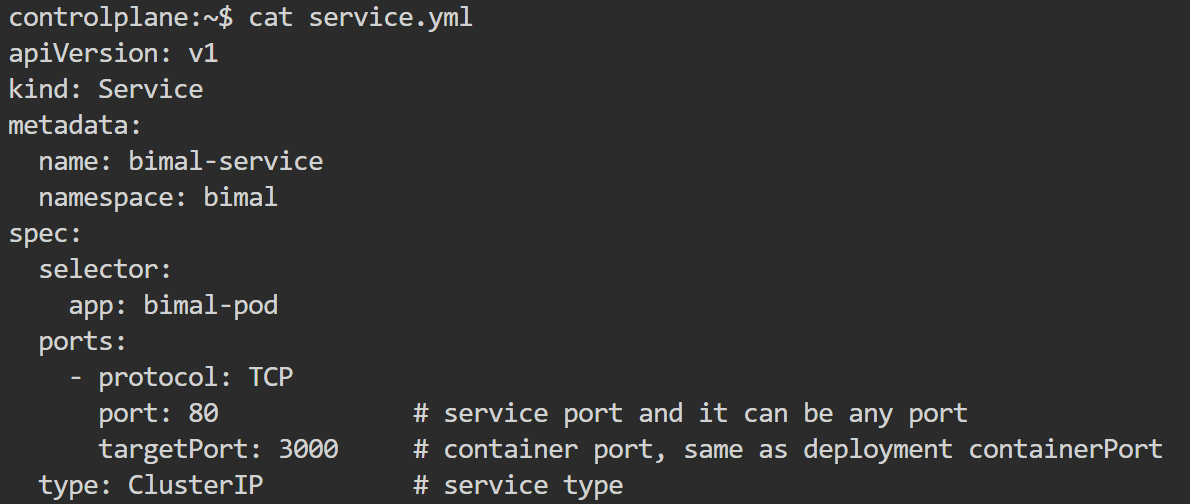
ports:

- protocol: TCP

port: 80 # service port and it can be any port

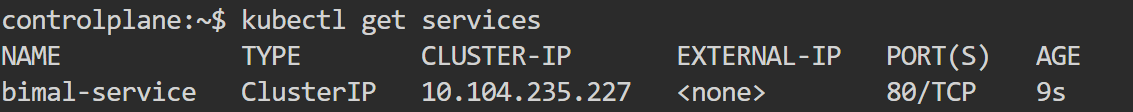
targetPort: 3000 # container port, same as deployment containerPort

type: ClusterIP # service type

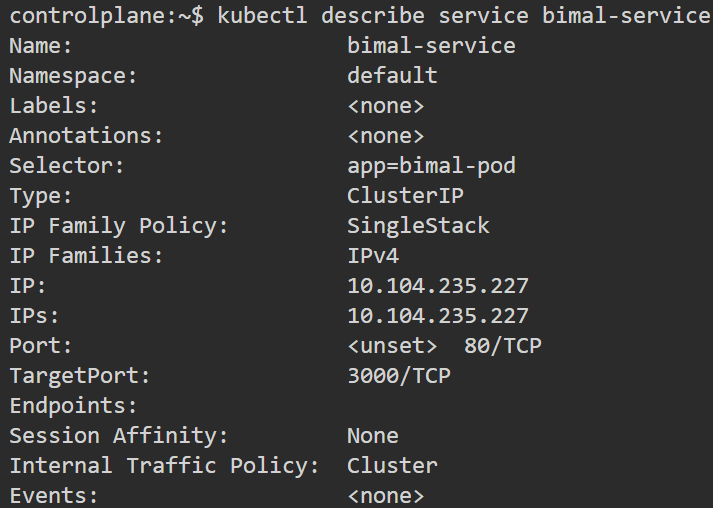


**Commands**

kubectl get services - List all services



kubectl describe service <service-name> - Get detailed information about Cluster IP



kubectl delete service <service-name> - Delete a service



**Node Port**

* It will expose the app on each worker node within the cluster and outside the cluster.

apiVersion: v1

kind: Service

metadata:

name: bimal-service

namespace: bimal

spec:

selector:

app: bimal-pod

ports:

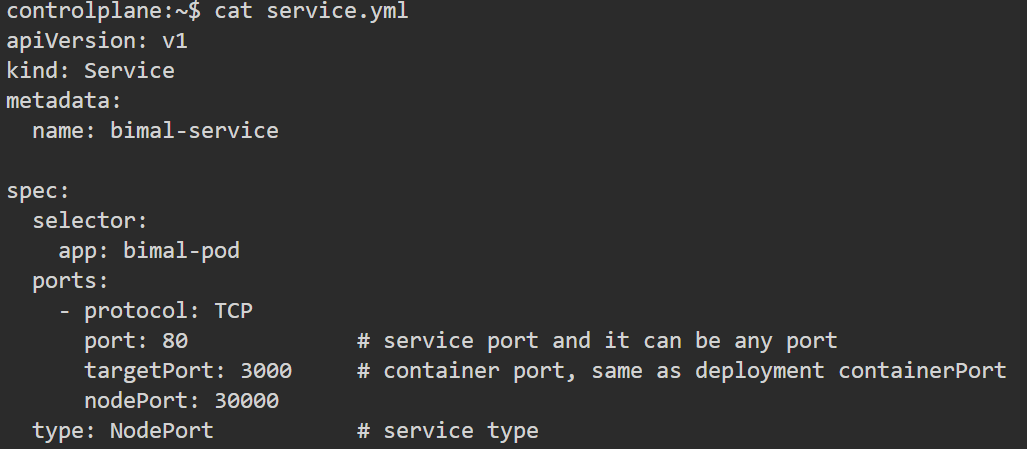
- protocol: TCP

port: 80 # service port and it can be any port

targetPort: 3000 # container port, same as deployment containerPort

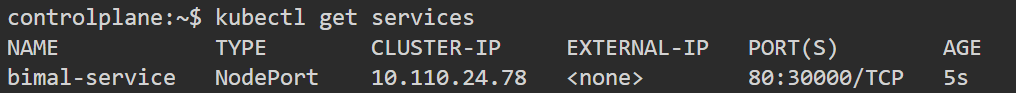
nodePort: 30000

type: NodePort

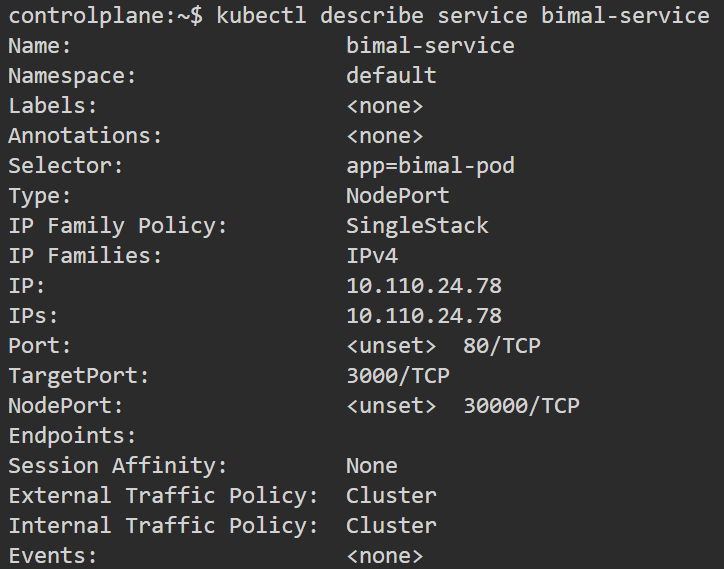


**Commands**

kubectl get services - List all services



kubectl describe service <service-name> - Get detailed information about Node Port



kubectl delete service <service-name> - Delete a service



**Load Balancer**

* It will expose the app outside the container.
* It will give one external ip add for accessing your app outside the cluster.

apiVersion: v1

kind: Service

metadata:

name: bimal-service

namespace: bimal

spec:

selector:

app: bimal-pod

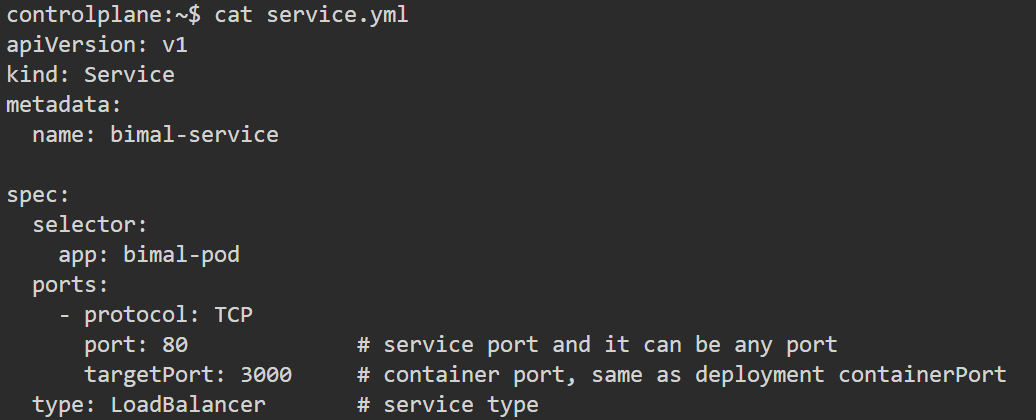
ports:

- protocol: TCP

port: 80 # service port number

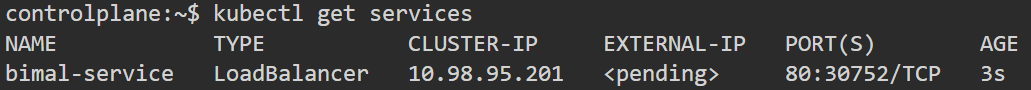
targetPort: 3000

type: LoadBalancer

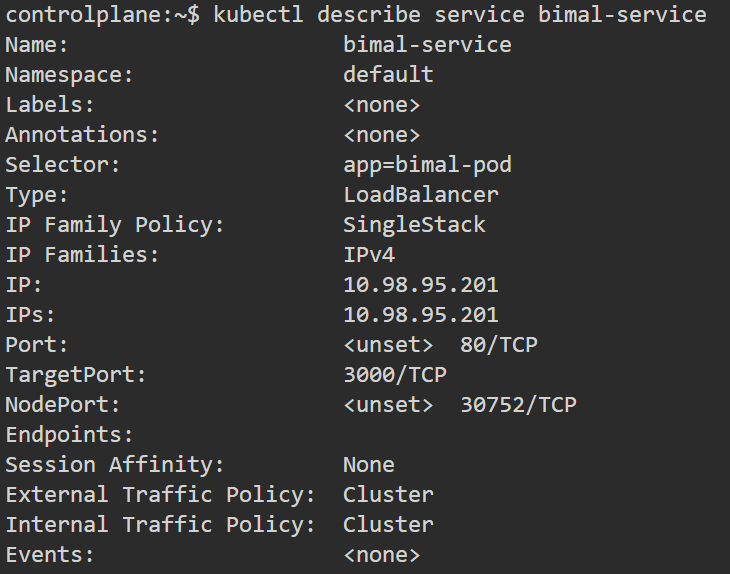


**Commands**

kubectl get services - List all services



kubectl describe service <service-name> - Get detailed information about Load Balancer



kubectl delete service <service-name> - Delete a service



**External Name**

* It is a specialized service type in k8s.
* Instead of routing traffic to pods inside the cluster but it maps the service name to an external DNS/ Host name (outside or inside the cluster).

apiVersion: v1

kind: Service

metadata:

name: bimal-service

namespace: bimal

spec:

selector:

app: ipl

ports:

- protocol: TCP

port: 80

externalName: flipkart.com, amazon.com, google.com

type: ExternalName