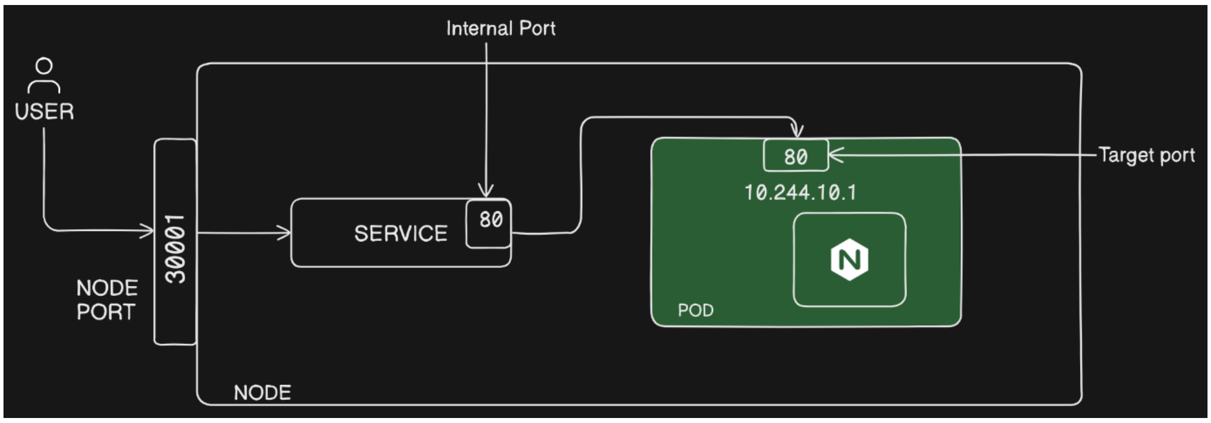
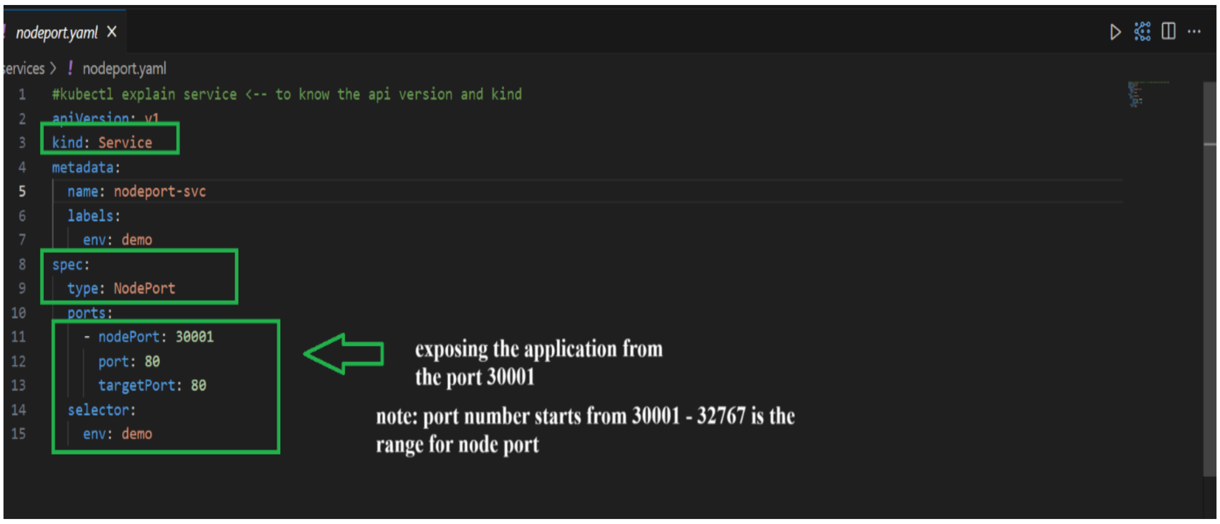
NODE-PORT AND CLUSTER-IP SERVICE IN KUBERNETES

NodePort is a way to expose a service to external traffic. When you define a service as a **NodePort** type, Kubernetes opens a specific port on each worker node, which can be accessed from outside the cluster. This NodePort will redirect traffic to the corresponding service running inside the cluster on a specified port.



How it works:

1. **NodePort range**: By default, the port number is chosen from a range (usually 30000-32767), though you can specify a custom port within this range.
2. **External access**: The service becomes accessible via <NodeIP>:<NodePort>. For example, if the Node's IP is 192.168.1.10 and the assigned NodePort is 30001, you can access the service at http://192.168.1.10:30001 or localhost:30001.
3. **Redirection**: Once the request hits the NodePort, it is forwarded to the target ClusterIP service inside the Kubernetes cluster, which in turn routes it to the appropriate pods.

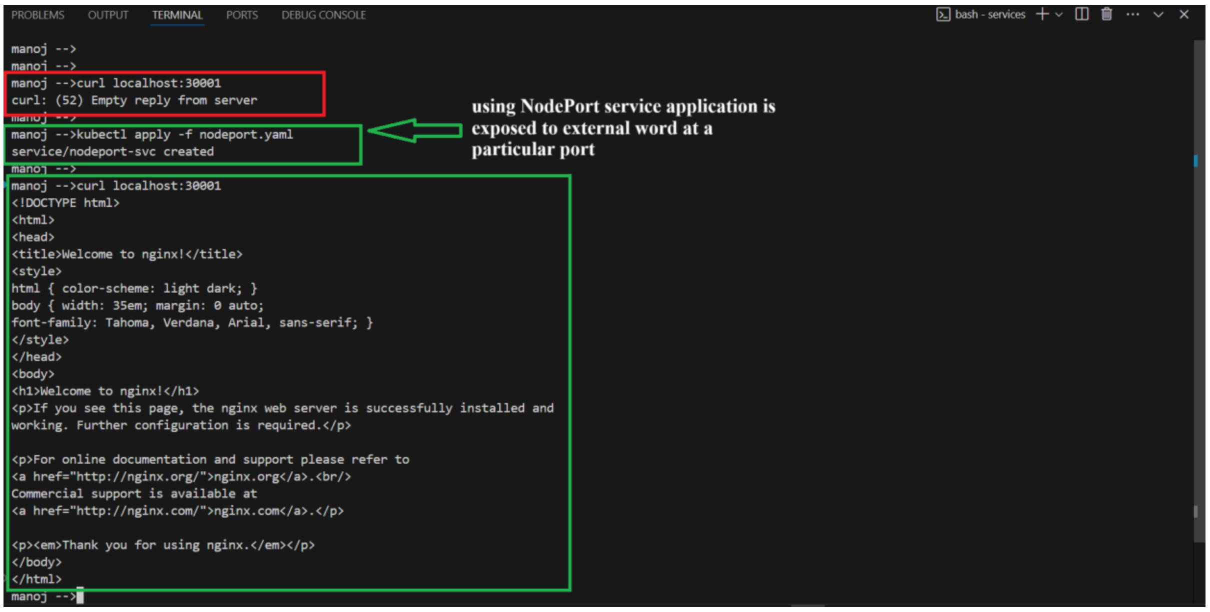


```yaml
#kubectl explain service <-- to know the api version and kind
apiVersion: v1
kind: Service
metadata:
  name: nodeport-svc
  labels:
    env: demo
spec:
  type: NodePort
  ports:
    - nodePort: 30001
      port: 80
      targetPort: 80
  selector:
    env: demo
```

exposing the application from the port 30001

note: port number starts from 30001 - 32767 is the range for node port

NODE-PORT AND CLUSTER-IP SERVICE IN KUBERNETES

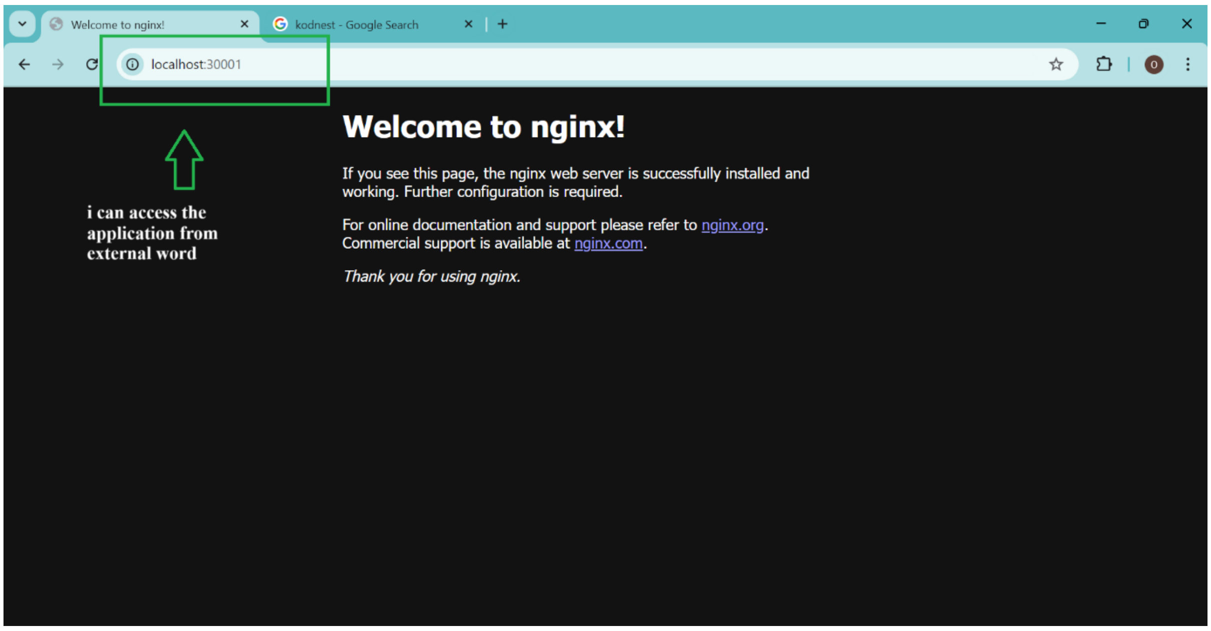


```
manoj -->
manoj -->
manoj -->curl localhost:30001
curl: (52) Empty reply from server
manoj -->
manoj -->kubectl apply -f nodeport.yaml
service/nodeport-svc created
manoj -->
manoj -->curl localhost:30001
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light 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>
manoj -->
```

using NodePort service application is exposed to external word at a particular port



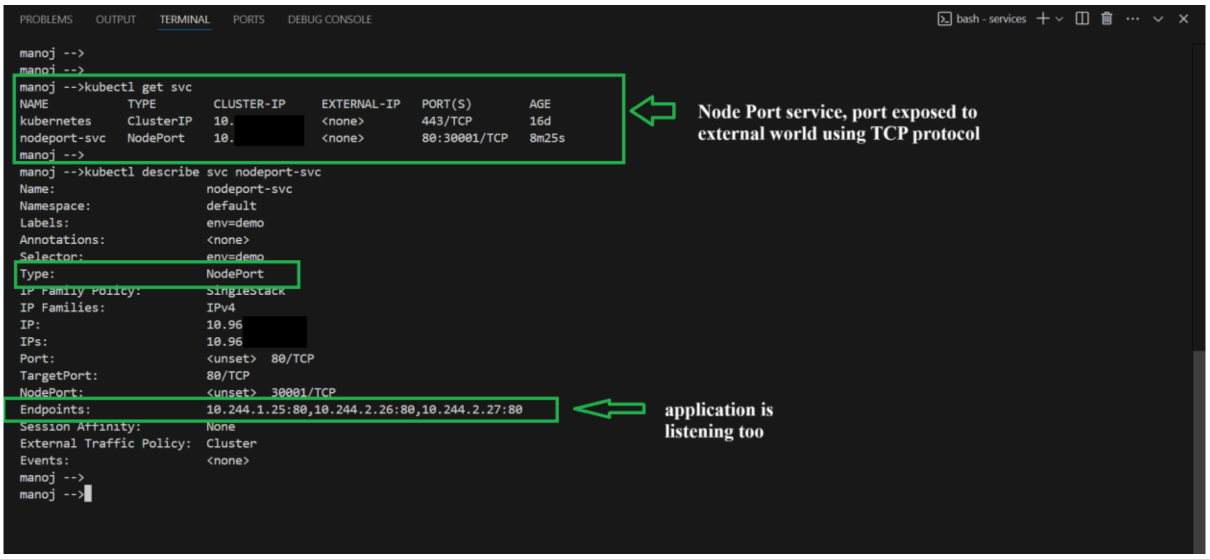
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

i can access the application from external word



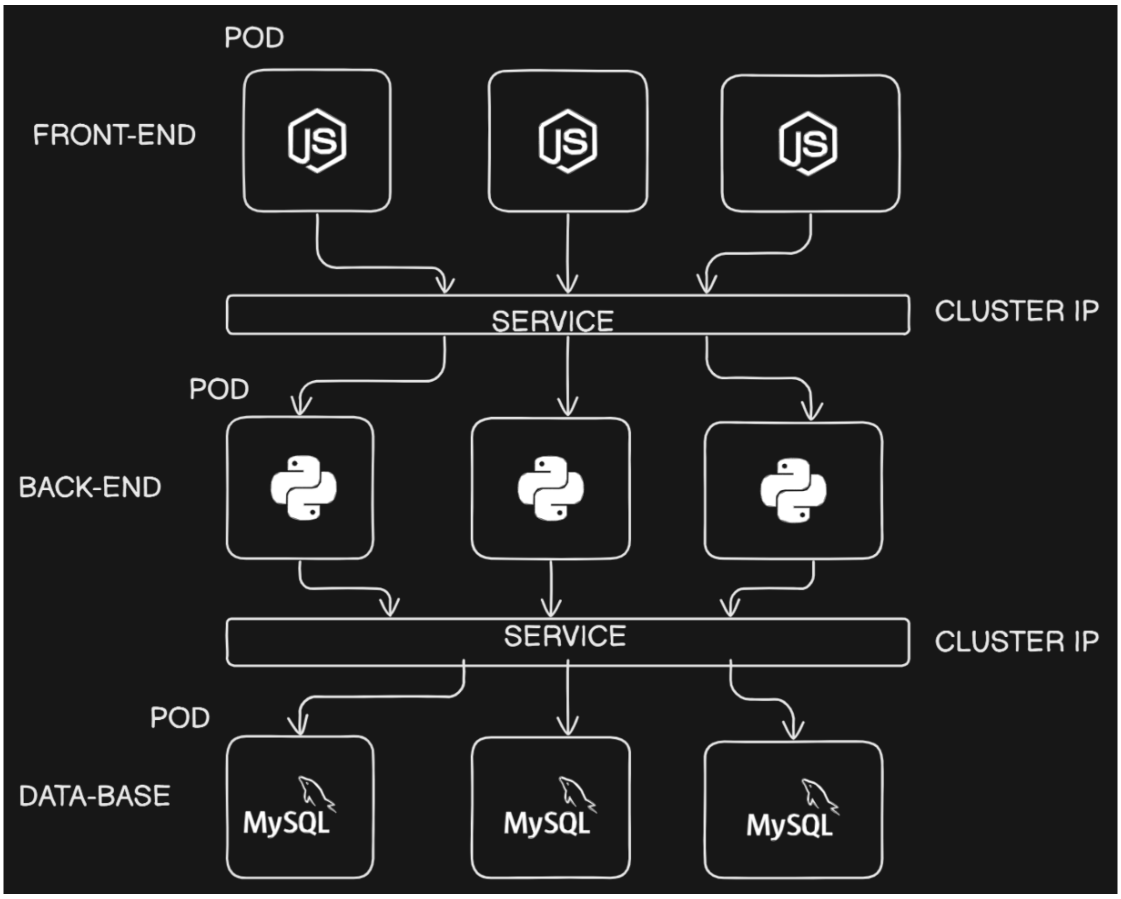
```
manoj -->
manoj -->
manoj -->kubectl get svc
NAME           TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)         AGE
kubernetes     ClusterIP   10.[redacted]  <none>        443/TCP         16d
nodeport-svc   NodePort    10.[redacted]  <none>        80:30001/TCP    8m25s
manoj -->
manoj -->kubectl describe svc nodeport-svc
Name:                     nodeport-svc
Namespace:                default
Labels:                   env=demo
Annotations:              <none>
Selector:                 env=demo
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.96[redacted]
IPs:                      10.96[redacted]
Port:                     <unset>  80/TCP
TargetPort:               80/TCP
NodePort:                 <unset>  30001/TCP
Endpoints:                10.244.1.25:80,10.244.2.26:80,10.244.2.27:80
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
manoj -->
manoj -->
```

Node Port service, port exposed to external world using TCP protocol

application is listening too

NODE-PORT AND CLUSTER-IP SERVICE IN KUBERNETES

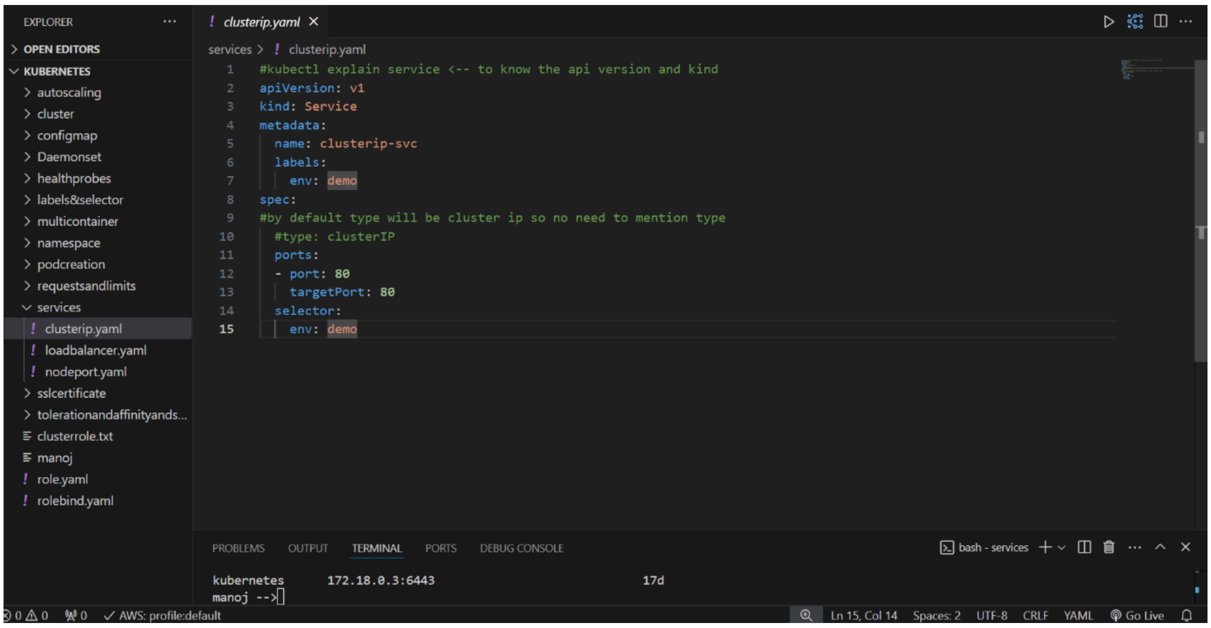
ClusterIP is the default service type used to expose services internally within the cluster. It provides a stable internal IP address (the ClusterIP) for communication between different components (like Pods) inside the Kubernetes cluster, without exposing the service to external traffic.



How ClusterIP Works:

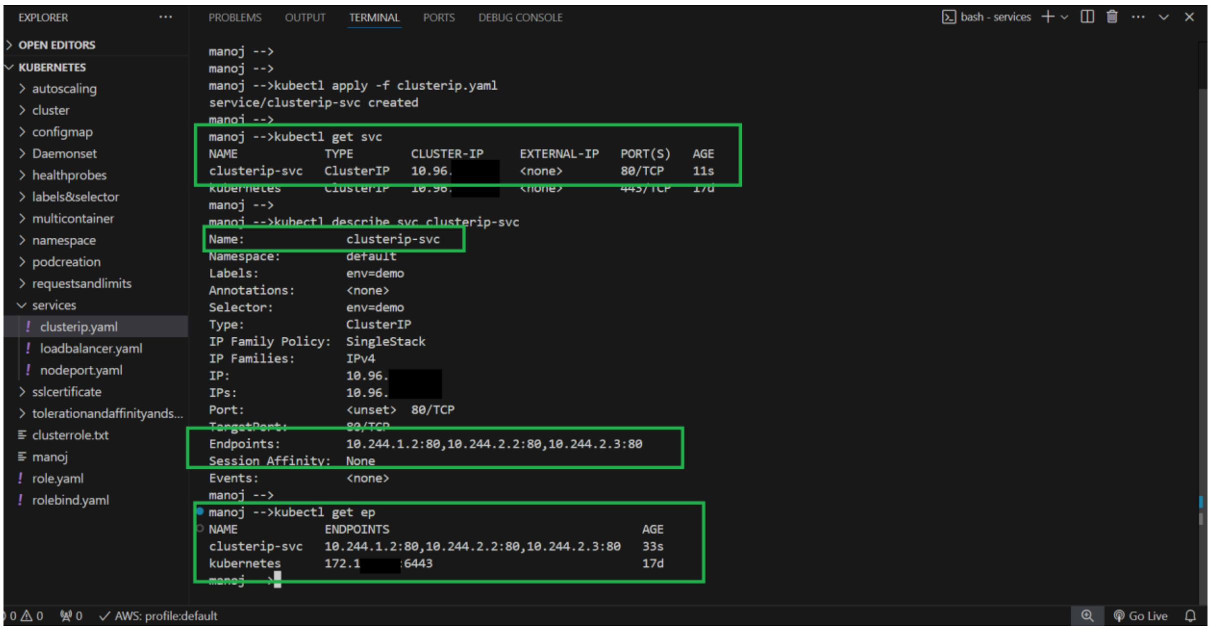
1. **Internal Load Balancer**: The ClusterIP acts as an internal load balancer that distributes incoming requests across the Pods associated with the service.
2. **DNS Integration**: Kubernetes automatically assigns a DNS name for the service, such as my-clusterip-service.default.svc.cluster.local, which makes it easy for Pods to reference services by name instead of IP.
3. **Pod to Pod Communication**: If multiple Pods need to communicate within the cluster, they can use the ClusterIP service to reach each other without needing to know the individual Pod IP addresses.

NODE-PORT AND CLUSTER-IP SERVICE IN KUBERNETES

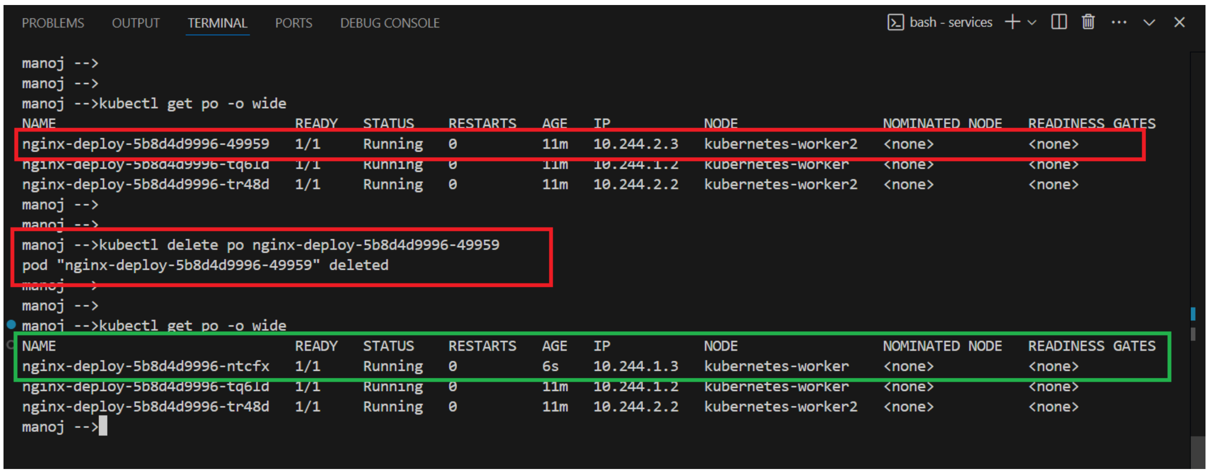


```yaml
#kubectl explain service <-- to know the api version and kind
apiVersion: v1
kind: Service
metadata:
  name: clusterip-svc
  labels:
    env: demo
spec:
#by default type will be cluster ip so no need to mention type
  #type: clusterIP
  ports:
  - port: 80
    targetPort: 80
  selector:
    env: demo
```

```
kubernetes      172.18.0.3:6443                       17d
manoj -->
```



```
manoj -->
manoj -->
manoj -->kubectl apply -f clusterip.yaml
service/clusterip-svc created
manoj -->
manoj -->kubectl get svc
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP    PORT(S)    AGE
clusterip-svc   ClusterIP   10.96.[redacted]    <none>         80/TCP     11s
kubernetes      ClusterIP   10.96.[redacted]    <none>         443/TCP    17d
manoj -->
manoj -->kubectl describe svc clusterip-svc
Name:                 clusterip-svc
Namespace:            default
Labels:               env=demo
Annotations:          <none>
Selector:             env=demo
Type:                 ClusterIP
IP Family Policy:     SingleStack
IP Families:          IPv4
IP:                   10.96.[redacted]
IPs:                  10.96.[redacted]
Port:                 <unset>  80/TCP
TargetPort:           80/TCP
Endpoints:            10.244.1.2:80,10.244.2.2:80,10.244.2.3:80
Session Affinity:     None
Events:               <none>
manoj -->
manoj -->kubectl get ep
NAME            ENDPOINTS                                     AGE
clusterip-svc   10.244.1.2:80,10.244.2.2:80,10.244.2.3:80     33s
kubernetes      172.1[redacted]:6443                               17d
manoj -->
```



```
manoj -->
manoj -->
manoj -->kubectl get po -o wide
NAME                           READY   STATUS    RESTARTS   AGE   IP           NODE                 NOMINATED NODE   READINESS GATES
nginx-deploy-5b8d4d9996-49959  1/1     Running   0          11m   10.244.2.3   kubernetes-worker2   <none>           <none>
nginx-deploy-5b8d4d9996-tq61d  1/1     Running   0          11m   10.244.1.2   kubernetes-worker    <none>           <none>
nginx-deploy-5b8d4d9996-tr48d  1/1     Running   0          11m   10.244.2.2   kubernetes-worker2   <none>           <none>
manoj -->
manoj -->
manoj -->kubectl delete po nginx-deploy-5b8d4d9996-49959
pod "nginx-deploy-5b8d4d9996-49959" deleted
manoj -->
manoj -->
manoj -->kubectl get po -o wide
NAME                           READY   STATUS    RESTARTS   AGE   IP           NODE                 NOMINATED NODE   READINESS GATES
nginx-deploy-5b8d4d9996-ntcfx  1/1     Running   0          6s    10.244.1.3   kubernetes-worker    <none>           <none>
nginx-deploy-5b8d4d9996-tq61d  1/1     Running   0          11m   10.244.1.2   kubernetes-worker    <none>           <none>
nginx-deploy-5b8d4d9996-tr48d  1/1     Running   0          11m   10.244.2.2   kubernetes-worker2   <none>           <none>
manoj -->
```

Key Characteristics of a ClusterIP Service:

1. **Internal Access Only**:
   - The service is only accessible from within the cluster. It cannot be accessed from outside the cluster unless combined with other service types like **NodePort** or **LoadBalancer**.
2. **Stable IP Address**:
   - Kubernetes assigns a static IP (ClusterIP) to the service, which remains the same throughout the lifecycle of the service. This allows other Pods to communicate with the service using this internal IP address.
3. **Service Discovery**:
   - The ClusterIP is also registered with Kubernetes DNS. Pods can access the service using the service name, which simplifies service discovery within the cluster.
4. **Traffic Routing**:
   - The service acts as a load balancer that forwards requests to one of the backend Pods (selected based on labels) listening on the specified target port.