Service and Its Types in Kubernetes

A **Service** in Kubernetes is an abstraction that provides **stable network access** to a group of Pods.

Since Pods are:

- Ephemeral (they can restart anytime)
- Assigned dynamic IP addresses

A Service ensures:

- Stable IP address
- Stable DNS name
- Load balancing across Pods

Need of Service

Without a Service:

- Pods communicate using Pod IPs
- If Pod restarts → IP changes
- Application breaks

With a Service:

- Pods are accessed using a fixed DNS name
- Traffic is automatically load balanced

How Service Works

Flow:

1. Service selects Pods using labels.
2. kube-proxy configures networking rules.
3. Traffic is forwarded to matching Pods.
4. Load balancing happens automatically.

Types of Services in Kubernetes

There are **4 main types**:

ClusterIP (Default)

NodePort

LoadBalancer

ClusterIP (Default)

What it does:

- Exposes Service **inside the cluster only**
- Gets internal cluster IP
- Not accessible externally

Use case:

- Internal microservices
- Backend APIs
- Database communication

Example YAML:

```yaml
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  type: ClusterIP
  selector:
    app: backend
  ports:
    - port: 80
      targetPort: 8080
```

NodePort

What it does:

- Exposes Service on a static port on each Node
- Accessible using:

<NodeIP>:<NodePort>

- Port range: 30000–32767

Use case:

- Testing
- On-prem clusters
- Small environments

LoadBalancer

What it does:

- Exposes Service externally using cloud provider load balancer
- Automatically provisions external IP

Use case:

- Production applications
- Public-facing apps

Supported in:

- AWS
- Azure
- GCP

Real-World Example

Application:

- Backend API → ClusterIP
- Frontend Testing → NodePort
- Production App → LoadBalancer