NAMESPACES

❖ NAMESPACES:

- In Kubernetes, namespaces provide a mechanism for isolating groups of resources within a single cluster.
- Names of resources need to be unique within a namespace, but not across namespaces.
- Namespace-based scoping is applicable only for namespace objects (e.g., Deployments, Services, etc) and not for cluster-wide objects (e.g., StorageClass, Nodes, PersistentVolumes, etc).
- Namespaces cannot be nested inside one another and each Kubernetes resource can only be in one namespace.
- Namespaces are a way to divide cluster resources between multiple users.

KUBERNETES INITIAL NAMESPACES:

DEFAULT: The default namespace for objects with no other namespace

KUBE-SYSTEM: The namespace for objects created by the Kubernetes system

KUBE-PUBLIC: This namespace is created automatically and is readable by all users. This namespace is mostly reserved for cluster usage, in case that some resources should be visible and readable publicly throughout the whole cluster.

KUBE-NODE-LEASE: This namespace holds Lease objects associated with each node. Node leases allow the kubelet to send heartbeats so that the control plane can detect node failure.

→ **List the current namespaces in a cluster using:**

$kubectl get namespace

$kubectl get ns

→ **Create a new Namespace:**

$kubectl create namespace dev-ns

$kubectl get namespace

→ **To change namespace:**

$kubectl config set-context --current --namespace=dev-ns

→ **To set the namespace for a current request, use the --namespace flag:**

$kubectl run nginx --image=nginx --namespace=<insert-namespace-here>

$kubectl get pods --namespace=<insert-namespace-name-here>

→ **You can permanently save the namespace for all subsequent kubectl commands in that context:**

$kubectl config set-context --current --namespace=<insert-namespace-here>

$kubectl config view --minify | grep namespace:

→ **To list all namespaces pods:**

$kubectl get all --all-namespaces (or) $kubectl get all -A

→ **To list specific namespaes:**

$kubectl get all -n dev-ns [to get pods in dev-namespace]

→ **To change namespace:**

$kubectl config set-context --current --namespace=dev-ns

$kubectl get pods

→ **To delete a name space:**

$kubectl delete namespaces dev-ns

$kubectl get namespaces

EXAMPLE: CREATE AN OBJECT ON DEV-NS NAMESPACE:

$vim namespace.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
  namespace: dev-ns
  labels:
    app: myapp
spec:
  containers:
  - name: myapp-container
    image: nginx
    command: ['sh', '-c', 'echo Hello Ram... && sleep 3600']
```

→ **Apply for creating an object:**

$kubectl apply -f namespace.yml

$kubectl get pods -n dev-ns

$kubectl describe namespace.yml -n dev-ns

→ **To delete a name space:**

$kubectl delete namespaces dev-ns

$kubectl get namespaces