**8. Kubernetes Namespaces - Create Imperatively using kubectl**

--- **Reference** - <https://github.com/stacksimplify/aws-eks-kubernetes-masterclass/tree/master/05-Kubernetes-Important-Concepts-for-Application-Deployments/05-05-Kubernetes-Namespaces>

--- **References** - <https://kubernetes.io/docs/tasks/administer-cluster/namespaces-walkthrough/>

**Introduction**

--- Namespaces allow to split-up resources into different groups.

--- Resource names should be unique in a namespace

--- We can use namespaces to create multiple environments like dev, staging and production etc

--- Kubernetes will always list the resources from default namespace unless we provide exclusively from which namespace, we need information from.

**Namespaces Generic - Deploy in Dev1 and Dev2**

**Create Namespace**

**# List Namespaces**

--- **kubectl get ns**

**# Create Namespace**

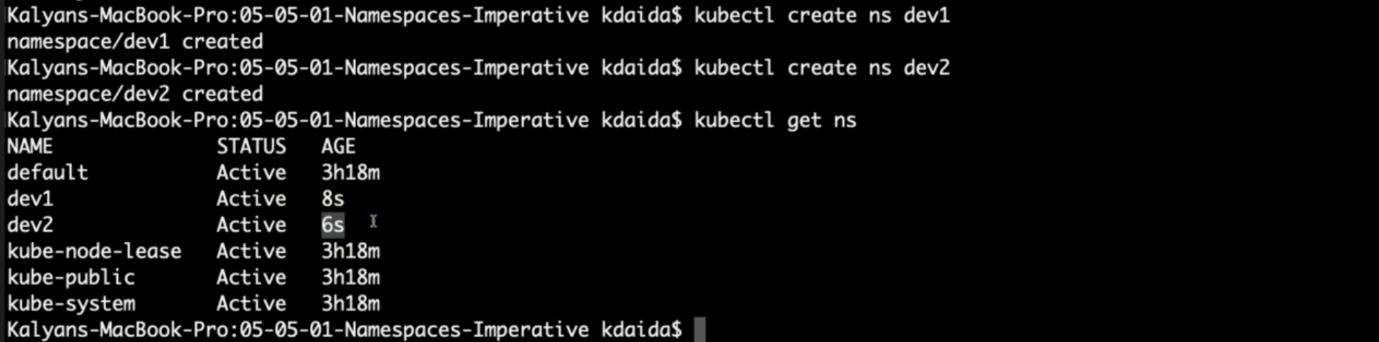
--- **kubectl create namespace <namespace-name>**

--- **kubectl create namespace dev1**

--- **kubectl create namespace dev2**

**# List Namespaces**

--- **kubectl get ns**



--- **note** – name space dev1 and dev2 are created.

--- **very important** - Comment NodePort in UserMgmt NodePort Service

--- 07-UserManagement-Service.yml

--- **Why? -** Whenever we create with same manifests multiple environments like dev1, dev2 with namespaces, we cannot have same worker node port for multiple services.

We will have port conflict. It’s good for k8s system to provide dynamic nodeport for us in such situations.

#nodePort: 31231

--- **07-UserManagement-Service.yml**

apiVersion: v1

kind: Service

metadata:

  name: usermgmt-restapp-service

  labels:

    app: usermgmt-restapp

spec:

  type: NodePort

  selector:

    app: usermgmt-restapp

  ports:

    - port: 8095

      targetPort: 8095

      #nodePort: 31231

--- **Error if not commented** - The Service "usermgmt-restapp-service" is invalid: spec.ports[0].nodePort: Invalid value: 31231: provided port is already allocated

**Deploy All k8s Objects**

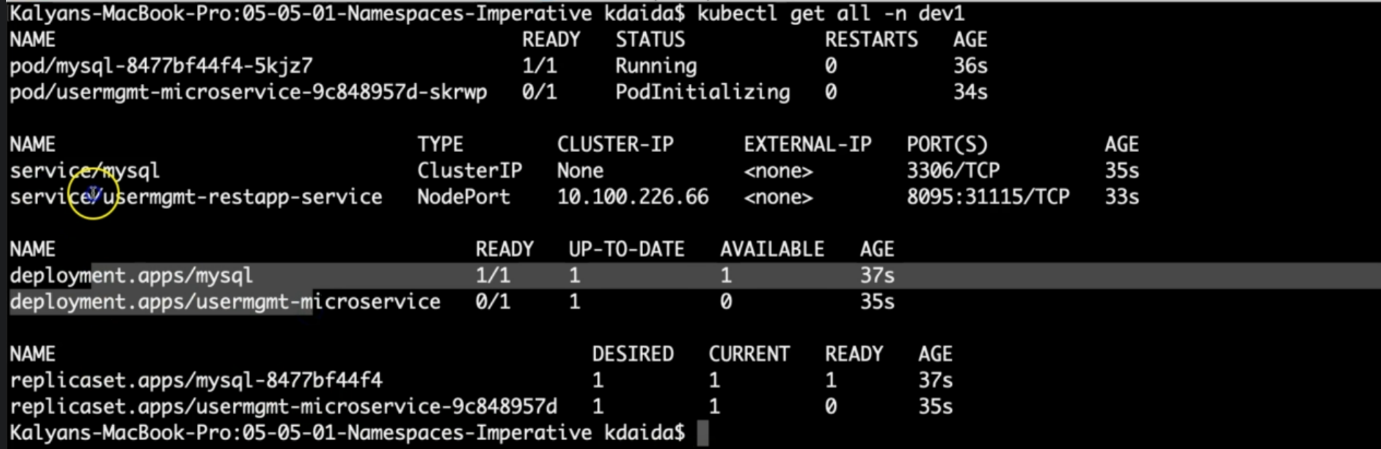
**# Deploy All k8s Objects**

--- **kubectl apply -f kube-manifests/ -n dev1**

--- **kubectl apply -f kube-manifests/ -n dev2**

**# List all objects from dev1 & dev2 Namespaces**

--- **kubectl get all -n dev1**



--- **kubectl get all -n dev2**



**Verify SC, PVC and PV**

--- **Shorter Note**

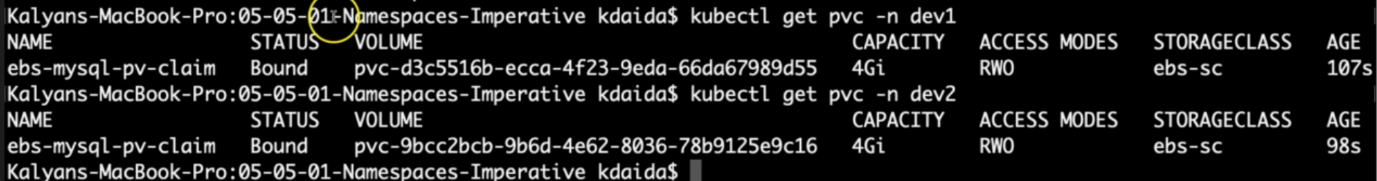
1. PVC is a namespace specific resource
2. PV and SC are generic

**Observation-1**: Persistent Volume Claim (PVC) gets created in respective namespaces

**# List PVC for dev1 and dev2**

--- **kubectl get pvc -n dev1**

--- **kubectl get pvc -n dev2**



**Observation-2**: Storage Class (SC) and Persistent Volume (PV) gets created generic. No specific namespace for them

**# List sc,pv**

--- **kubect get sc,pv**

**Access Application**

**Dev1 Namespace**

**# Get Public IP**

--- **kubectl get nodes -o wide**

**# Get NodePort for dev1 usermgmt service**

--- **kubectl get svc -n dev1**

**# Access Application**

--- **http://<Worker-Node-Public-Ip>:<Dev1-NodePort>/usermgmt/health-stauts**

**Dev2 Namespace**

**# Get Public IP**

--- **kubectl get nodes -o wide**

**# Get NodePort for dev2 usermgmt service**

--- **kubectl get svc -n dev2**

**# Access Application**

--- **http://<Worker-Node-Public-Ip>:<Dev2-NodePort>/usermgmt/health-stauts**

**Clean-Up**

**# Delete namespaces dev1 & dev2**

--- **kubectl delete ns dev1**

--- **kubectl delete ns dev2**

**# List all objects from dev1 & dev2 Namespaces**

--- **kubectl get all -n dev1**

--- **kubectl get all -n dev2**

**# List Namespaces**

--- **kubectl get ns**

**# List sc,pv**

--- **kubectl get sc,pv**

**# Delete Storage Class**

--- **kubectl delete sc ebs-sc**

**# Get all from All Namespaces**

--- **kubectl get all -all-namespaces**