

Organize Kubernetes Pods using Labels and Namespaces

◆ Step 1: Create Namespace YAML

```
nano namespaces.yaml
```

```
apiVersion: v1
kind: Namespace
metadata:
  name: dev
---
apiVersion: v1
kind: Namespace
metadata:
  name: prod
```

◆ Step 2: Create Pods YAML

```
nano pods.yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  name: frontend-pod
  namespace: dev
  labels:
    app: frontend
spec:
  containers:
    - name: frontend
      image: nginx
---
apiVersion: v1
kind: Pod
metadata:
  name: backend-pod
  namespace: dev
  labels:
    app: backend
spec:
  containers:
    - name: backend
```

```

        image: nginx
---
apiVersion: v1
kind: Pod
metadata:
  name: frontend-pod
  namespace: prod
  labels:
    app: frontend
spec:
  containers:
  - name: frontend
    image: nginx
---
apiVersion: v1
kind: Pod
metadata:
  name: backend-pod
  namespace: prod
  labels:
    app: backend
spec:
  containers:
  - name: backend
    image: nginx

```

◆ Step 3: Apply YAML files

```

kubectl apply -f namespaces.yaml
kubectl apply -f pods.yaml

```

◆ Step 4: Verify Pods by Namespace

```

kubectl get pods -n dev
kubectl get pods -n prod

```

◆ Step 5: Filter Pods using Labels

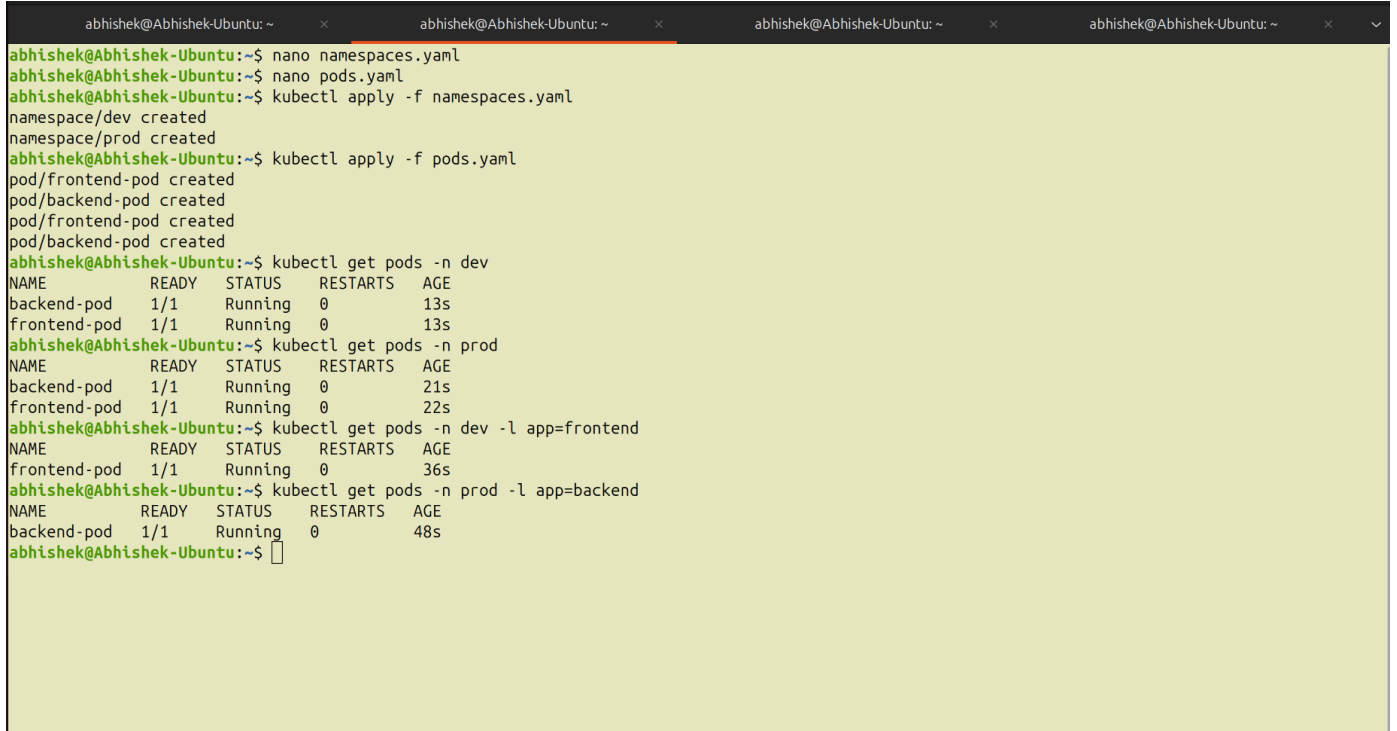
```

kubectl get pods -n dev -l app=frontend
kubectl get pods -n prod -l app=backend

```

◆ Step 6: Cleanup

```
kubectl delete pods --all -n dev
kubectl delete pods --all -n prod
kubectl delete namespace dev
kubectl delete namespace prod
```

A terminal window with four tabs, all titled 'abhishek@Abhishek-Ubuntu: ~'. The terminal shows a series of commands and their outputs. First, 'nano namespaces.yaml' is run, followed by 'nano pods.yaml'. Then, 'kubectl apply -f namespaces.yaml' is executed, resulting in 'namespace/dev created' and 'namespace/prod created'. Next, 'kubectl apply -f pods.yaml' is run, resulting in 'pod/frontend-pod created', 'pod/backend-pod created', 'pod/frontend-pod created', and 'pod/backend-pod created'. The user then runs 'kubectl get pods -n dev', which shows a table with columns NAME, READY, STATUS, RESTARTS, and AGE, containing two rows for 'backend-pod' and 'frontend-pod'. This is followed by 'kubectl get pods -n prod', showing a similar table with two rows. Then, 'kubectl get pods -n dev -l app=frontend' shows one row for 'frontend-pod'. Finally, 'kubectl get pods -n prod -l app=backend' shows one row for 'backend-pod'. The terminal ends with a prompt character.

```
abhishek@Abhishek-Ubuntu:~$ nano namespaces.yaml
abhishek@Abhishek-Ubuntu:~$ nano pods.yaml
abhishek@Abhishek-Ubuntu:~$ kubectl apply -f namespaces.yaml
namespace/dev created
namespace/prod created
abhishek@Abhishek-Ubuntu:~$ kubectl apply -f pods.yaml
pod/frontend-pod created
pod/backend-pod created
pod/frontend-pod created
pod/backend-pod created
abhishek@Abhishek-Ubuntu:~$ kubectl get pods -n dev
NAME          READY   STATUS    RESTARTS   AGE
backend-pod    1/1     Running   0           13s
frontend-pod   1/1     Running   0           13s
abhishek@Abhishek-Ubuntu:~$ kubectl get pods -n prod
NAME          READY   STATUS    RESTARTS   AGE
backend-pod    1/1     Running   0           21s
frontend-pod   1/1     Running   0           22s
abhishek@Abhishek-Ubuntu:~$ kubectl get pods -n dev -l app=frontend
NAME          READY   STATUS    RESTARTS   AGE
frontend-pod   1/1     Running   0           36s
abhishek@Abhishek-Ubuntu:~$ kubectl get pods -n prod -l app=backend
NAME          READY   STATUS    RESTARTS   AGE
backend-pod    1/1     Running   0           48s
abhishek@Abhishek-Ubuntu:~$
```