CKAD Ders Dökümanı #04 — Namespace

Dil: Türkçe • CKAD Kategorisi: Application Design and Build (20%) • Odak: Konu anlatımı + en sık kullanılan komutlar

1) Konu Anlatımı

Namespace, Kubernetes kaynaklarını (Pod, Service, ConfigMap, Secret vb.) mantıksal olarak izole etmek ve organize etmek için kullanılan bir sınırdır. Aynı cluster içinde farklı ekipler, ortamlar (dev/test/prod) veya projeler için kaynakları ayırmak için kullanılır.

Birçok Kubernetes kaynağı namespaced'dir (Pod, Service, Deployment gibi). Bazı kaynaklar ise cluster-scoped'dur (Node, PersistentVolume, ClusterRole, Namespace gibi). CKAD'de görevler sıklıkla belirli bir namespace içinde yapılır; bu yüzden doğru namespace'te çalışmak kritik önemdedir.

Önemli not: Namespace "güvenlik" tek başına değildir. İzolasyon ve yetkilendirme için RBAC, NetworkPolicy, ResourceQuota gibi mekanizmalarla birlikte kullanılmalıdır.

Namespace ile ilgili temel kavramlar

Namespaced kaynaklar: Pod, Deployment, Service, ConfigMap, Secret, Ingress, NetworkPolicy, Role/RoleBinding vb. Cluster-scoped kaynaklar: Node, Namespace, PersistentVolume, ClusterRole/ClusterRoleBinding vb. Default namespace: Namespace belirtmezseniz komutlar varsayılan olarak default namespace'i hedefler. Context: kubectl context içinde varsayılan namespace ayarlanabilir (sınavda zaman kazandırır).

2) En Sık Karşılaşılan YAML Örnekleri

Namespace oluşturma

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

Belirli bir namespace'e kaynak tanımlama (metadata.namespace)

```
apiVersion: v1
kind: Pod
metadata:
  name: web
  namespace: dev
spec:
  containers:
  - name: nginx
    image: nginx:1.25
```

3) Sık Kullanılan Alanlar (Kısa Açıklamalar)

Alan	Ne işe yarar?
kind: Namespace	Namespace kaynağının türü (cluster-scoped).
metadata.name	Namespace adı.
metadata.labels	Organizasyon/politika amaçlı etiketler (ör. network policy seçimleri).
metadata.namespace	Namespaced kaynakları hangi namespace’te oluşturacağınızı belirtir (P

4) En Sık Kullanılan kubectl Komutları

Namespace yönetimi ve namespace’te çalışmayı hızlandıran komutlar:

4.1 Namespace listeleme / görüntüleme

Mevcut namespace’leri listeler ve detaylarını görürsünüz.

```
kubectl get ns
kubectl get namespace
kubectl describe ns dev
```

4.2 Namespace oluşturma / silme

Yeni namespace oluşturur veya kaldırır.

```
# Oluştur
kubectl create ns dev
# veya
kubectl apply -f namespace.yaml

# Sil (dikkat: içindeki kaynakları da kaldırır)
kubectl delete ns dev
```

4.3 Komutları belirli namespace’te çalıştırma (-n)

Sınavda en kritik alışkanlık: her komutta doğru namespace’i hedeflemek.

```
# Pod'ları dev namespace'te listele
kubectl get pod -n dev

# Kaynak oluştururken namespace belirt
kubectl apply -f pod.yaml -n dev

# Service oluştururken namespace belirt
kubectl expose deployment web --port=80 --name=web-svc -n dev
```

4.4 Mevcut context için varsayılan namespace ayarlama

Sürekli -n yazmamak için context üzerinde varsayılan namespace ayarlanabilir.

```
# Aktif context'i gör
kubectl config current-context

# Aktif context'te varsayılan namespace'i dev yap
kubectl config set-context --current --namespace=dev

# Geri default'a dönmek istersen:
kubectl config set-context --current --namespace=default
```

4.5 Belirli namespace’te tüm kaynakları görmek

Sınavda hızlı envanter için kullanışlı.

```
kubectl get all -n dev
kubectl get deploy,rs,pod,svc,ingress -n dev
```

4.6 Kaynakları namespace’ler arası kopyalama (pratik yaklaşım)

Doğrudan 'move' yoktur. Genelde YAML alıp namespace’i değiştirerek apply edilir.

```
# Kaynağın YAML'ını al
kubectl get cm app-config -n dev -o yaml

# Çıktıda metadata.namespace alanını değiştirip apply et (örn. test)
# veya pipeline ile: yq/sed kullan (sınavda genelde manuel daha hızlı)
```

5) Troubleshooting Hızlı Rehber

Kaynağı bulamıyorum: Yanlış namespace’te olabilirsiniz. kubectl get ... -n ... veya -A kullanın.Ben oluşturdum ama görünmüyor: Komutun hangi namespace’te çalıştığını kontrol edin (context namespace).Delete ettim ama bitmedi: Namespace terminating’de kalabilir (finalizer). CKAD’de nadir; genelde beklemek yeterli.Permission denied: RBAC kapsamında ilgili namespace’te yetkiniz olmayabilir.

6) CKAD İpuçları

Soruyu okur okumaz namespace’i belirle. En çok hata sebebi: yanlış namespace.Zaman kazanmak için: kubectl config set-context --current --namespace=...Tüm namespace’lerde arama: kubectl get pod -A (kayıp kaynak bulmak için).Cluster-scoped vs namespaced farkını bil: Namespace’in kendisi cluster-scoped’tur.