Important kubectl Commands for CKAD Certification

# Basic Commands

## Get Cluster Information

* kubectl cluster-info

## Get API Resources

* kubectl api-resources

## Get All Resources in a Namespace

* kubectl get all -n <namespace>

## Get Resource Details

* kubectl get pods
* kubectl get svc
* kubectl get deployments
* kubectl get nodes

## Describe Resources

* kubectl describe pod <pod-name>
* kubectl describe svc <service-name>
* kubectl describe node <node-name>

## Create Resources

* kubectl create -f <file.yaml>
* kubectl run nginx --image=nginx

## Delete Resources

* kubectl delete pod <pod-name>
* kubectl delete svc <service-name>
* kubectl delete -f <file.yaml>

## Edit Resources

* kubectl edit pod <pod-name>

# Resource Management

## Scale Deployments

* kubectl scale deployment <deployment-name> --replicas=3

## Autoscale Deployments

* kubectl autoscale deployment <deployment-name> --min=2 --max=5 --cpu-percent=80

## Rollout Management

* kubectl rollout status deployment/<deployment-name>
* kubectl rollout history deployment/<deployment-name>
* kubectl rollout undo deployment/<deployment-name>

# Debugging and Troubleshooting

## Logs

* kubectl logs <pod-name>
* kubectl logs -f <pod-name>
* kubectl logs <pod-name> --previous

## Execute Commands in a Pod

* kubectl exec <pod-name> -- <command>
* kubectl exec -it <pod-name> -- /bin/bash

## Port Forwarding

* kubectl port-forward <pod-name> 8080:80

## Get Events

* kubectl get events

## Resource Usage (Top)

* kubectl top pod
* kubectl top node

## Debugging Containers

* kubectl debug <pod-name> --image=busybox --target=<container-name> -- /bin/sh

# Networking and Services

## Expose a Pod/Deployment as a Service

* kubectl expose pod <pod-name> --port=8080 --target-port=80 --type=NodePort
* kubectl expose deployment <deployment-name> --port=8080 --target-port=80 --type=LoadBalancer

## View Service Endpoints

* kubectl get endpoints

## Apply Network Policies

* kubectl apply -f <network-policy.yaml>

# ConfigMaps and Secrets

## Create ConfigMap

* kubectl create configmap <config-name> --from-literal=key=value
* kubectl create configmap <config-name> --from-file=<file>

## Create Secret

* kubectl create secret generic <secret-name> --from-literal=key=value
* kubectl create secret generic <secret-name> --from-file=<file>

## View ConfigMap/Secret

* kubectl get configmap <config-name> -o yaml
* kubectl get secret <secret-name> -o yaml

# Persistent Storage

## Create PersistentVolumeClaim

* kubectl apply -f <pvc.yaml>

## View PersistentVolumes and PersistentVolumeClaims

* kubectl get pv
* kubectl get pvc

# Namespaces

## List Namespaces

* kubectl get namespaces

## Switch Context to a Namespace

* kubectl config set-context --current --namespace=<namespace>

## Create/Delete Namespace

* kubectl create namespace <namespace>
* kubectl delete namespace <namespace>

# Role-Based Access Control (RBAC)

## View Roles and RoleBindings

* kubectl get roles
* kubectl get rolebindings

## Create Role/RoleBinding

* kubectl apply -f <role.yaml>
* kubectl apply -f <rolebinding.yaml>

# Advanced Resource Management

## Apply Resource Configuration

* kubectl apply -f <file.yaml>

## Patch a Resource

* kubectl patch deployment <deployment-name> -p '{"spec":{"replicas":5}}'

## Replace a Resource

* kubectl replace -f <file.yaml>

## Dry Run Commands

* kubectl apply -f <file.yaml> --dry-run=client

# Miscellaneous

## Save Resource Configuration to YAML/JSON

* kubectl get pod <pod-name> -o yaml
* kubectl get pod <pod-name> -o json

## Label Resources

* kubectl label pod <pod-name> key=value

## Annotate Resources

* kubectl annotate pod <pod-name> key=value

## Taint/Tolerate Nodes

* kubectl taint nodes <node-name> key=value:NoSchedule