DevOps Shack

Sample Kubernetes Manifests Files for DevOps interviews

[Click Here To Enrol To Batch-5 | DevOps & Cloud DevOps]

1. Deployment

A Deployment ensures that a specified number of pod replicas are running at any given time. It can also be used to roll out updates to applications.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: example-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: example
  template:
    metadata:
      labels:
        app: example
    spec:
      containers:
      - name: example-container
        image: nginx:latest
        ports:
        - containerPort: 80
```

Explanation:

- **apiVersion:** The version of the Kubernetes API to use.
- **kind:** The type of resource (Deployment).
- **metadata:** Data to help uniquely identify the object, including a name string.
- **spec:** Specification of the desired behavior of the deployment.
    - **replicas:** The number of desired pod replicas.

  - **selector:** A label query to identify the set of pods.
  - **template:** The pod template describes the pods that will be created.
    - **metadata:** Labels for the pods.
    - **spec:** Specification for the container(s) in the pod.
      - **containers:** A list of containers, each with a name, image, and port mappings.

2. Service

A Service provides a stable endpoint (IP address and port) to access a set of pods.

```
apiVersion: v1
kind: Service
metadata:
  name: example-service
spec:
  selector:
    app: example
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
  type: ClusterIP
```

Explanation:

- **apiVersion:** The version of the Kubernetes API to use.
- **kind:** The type of resource (Service).
- **metadata:** Data to help uniquely identify the object, including a name string.
- **spec:** Specification of the desired behavior of the service.
  - **selector:** Selects the pods that the service will target.
  - **ports:** List of ports that the service will expose.
    - **protocol:** Protocol used by the service (TCP).
    - **port:** Port exposed by the service.
    - **targetPort:** Port on the container that the traffic will be directed to.
  - **type:** Type of service (ClusterIP is the default, providing an internal IP).

3. PersistentVolume (PV)

A PersistentVolume is a piece of storage in the cluster that has been provisioned by an administrator or dynamically provisioned using Storage Classes.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: example-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
```

Explanation:

- **apiVersion:** The version of the Kubernetes API to use.
- **kind:** The type of resource (PersistentVolume).
- **metadata:** Data to help uniquely identify the object, including a name string.
- **spec:** Specification of the desired behavior of the PV.
  - **capacity:** Storage capacity of the volume.
  - **accessModes:** The ways the volume can be accessed (ReadWriteOnce allows read/write by a single node).
  - **hostPath:** Path on the host node’s filesystem.

4. PersistentVolumeClaim (PVC)

A PersistentVolumeClaim is a request for storage by a user. It is similar to a pod.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: example-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Explanation:

- **apiVersion:** The version of the Kubernetes API to use.
- **kind:** The type of resource (PersistentVolumeClaim).
- **metadata:** Data to help uniquely identify the object, including a name string.
- **spec:** Specification of the desired behavior of the PVC.
  - **accessModes:** The ways the volume can be accessed.
  - **resources:** Specifies the resources required by the PVC.
    - **requests:** Minimum amount of storage requested.

5. Secret

A Secret is an object that contains a small amount of sensitive data such as a password, a token, or a key.

```
apiVersion: v1
kind: Secret
metadata:
  name: example-secret
type: Opaque
data:
  username: YWRtaW4=
  password: MWYyZDFlMmU2N2Rm
```

Explanation:

- **apiVersion:** The version of the Kubernetes API to use.
- **kind:** The type of resource (Secret).
- **metadata:** Data to help uniquely identify the object, including a name string.
- **type:** The type of secret (Opaque is a generic type).
- **data:** The secret data, where each key must be base64 encoded.

6. ConfigMap

A ConfigMap is an API object used to store non-confidential data in key-value pairs.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-configmap
data:
  database_url: postgres://user:password@hostname:5432/database
  log_level: DEBUG
```

Explanation:

- **apiVersion:** The version of the Kubernetes API to use.
- **kind:** The type of resource (ConfigMap).
- **metadata:** Data to help uniquely identify the object, including a name string.
- **data:** The configuration data, in key-value pairs.

These examples provide basic templates that you can expand upon or modify based on your specific needs in a Kubernetes environment.