Storage

Weight : 4

1) For this question, please set this context (In exam, diff cluster name)

```
kubectl config use-context kubernetes-admin@kubernetes
```

- An existing nginx pod, `my-pod-cka` and Persistent Volume Claim (PVC) named `my-pvc-cka` are available. Your task is to implement the following modifications:
- NOTE:- PVC to PV binding and `my-pod-cka` pods sometimes takes around 2Mins to Up & Running So Please wait
- Update the pod to include a sidecar container that uses the `busybox` image. Ensure that this sidecar container remains operational by including an appropriate command `"tail -f /dev/null"` .
- Share the `shared-storage` volume between the main application and the sidecar container, mounting it at the path `/var/www/shared` . Additionally, ensure that the sidecar container has `read-only` access to this shared volume.

Solution:-

Step 1: run edit pod my-pod-cka command

```
kubectl edit po my-pod-cka
```

Step 2: Update sidecontainer and save it

```
  - name: sidecar-container
    image: busybox
    command: ["sh", "-c", "tail -f /dev/null"]
    volumeMounts:
      - name: shared-storage
        mountPath: /var/www/shared
        readOnly: true
```

Step 3: run kubectl replace command

```
kubectl replace -f /tmp/kubectl-edit-1047923679.yaml --force
```

Weight : 2

2) For this question, please set this context (In exam, diff cluster name)

```
kubectl config use-context kubernetes-admin@kubernetes
```

Modify the size of the existing Persistent Volume Claim (PVC) named `yellow-pvc-cka` to request `60Mi` of storage from the `yellow-pv-cka` volume. Ensure that the PVC successfully resizes to the new size and remains in the `Bound` state.

Solution:-

Step 1: run edit pvc yellow-pvc-cka command

```
kubectl edit pvc yellow-pvc-cka
```

Step 2: replace from-

```
  resources:
    requests:
      storage: 40Mi
```

to-

```
  resources:
    requests:
      storage: 60Mi
```

Weight : 10

3) For this question, please set this context (In exam, diff cluster name)

```
kubectl config use-context kubernetes-admin@kubernetes
```

- Create a Storage Class named `fast-storage` with a provisioner of `kubernetes.io/no-provisioner` and a `volumeBindingMode` of `Immediate` .
- Create a Persistent Volume (PV) named `fast-pv-cka` with a storage capacity of 50Mi using the fast-storage Storage Class.
- Create a Persistent Volume Claim (PVC) named `fast-pvc-cka` that requests `30Mi` of storage from the `fast-pv-cka` PV.
- Create a Pod named fast-pod-cka that uses the fast-pvc-cka PVC and mounts the volume at the path /app/data.

Solution:-

Step 1: create storage class

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: fast-storage
provisioner: kubernetes.io/no-provisioner
volumeBindingMode: Immediate
```

Step 2: create pv

```
apiVersion: v1
kind: PersistentVolume
```

```
metadata:
  name: fast-pv-cka
spec:
  capacity:
    storage: 50Mi
  accessModes:
    - ReadWriteOnce
  storageClassName: fast-storage
  hostPath:
    path: /tmp/fast-data
```

Step 3: create pvc

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: fast-pvc-cka
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: fast-storage
  resources:
    requests:
      storage: 30Mi
```

Step 4: create pod

```
apiVersion: v1
kind: Pod
metadata:
  name: fast-pod-cka
spec:
  containers:
    - name: my-container
      image: nginx:latest
      volumeMounts:
        - name: shared-volume
          mountPath: /app/data
  volumes:
    - name: shared-volume
      persistentVolumeClaim:
        claimName: fast-pvc-cka
```

Step 5: `kubectl apply -f file.yaml`

Weight : 8

4) For this question, please set this context (In exam, diff cluster name)

```
kubectl config use-context kubernetes-admin@kubernetes
```

Your task involves setting up storage components in a Kubernetes cluster. Follow these steps:

Step 1: Create a Storage Class named `blue-stc-cka` with the following properties:

- Provisioner: `kubernetes.io/no-provisioner`
- Volume binding mode: `WaitForFirstConsumer`

Step 2: Create a Persistent Volume (PV) named `blue-pv-cka` with the following properties:

- Capacity: `100Mi`
- Access mode: `ReadWriteOnce`
- Reclaim policy: `Retain`
- Storage class: `blue-stc-cka`
- Local path: `/opt/blue-data-cka`
- Node affinity: Set node affinity to create this PV on `controlplane` .

Step 3: Create a Persistent Volume Claim (PVC) named `blue-pvc-cka` with the following properties:

- Access mode: `ReadWriteOnce`
- Storage class: `blue-stc-cka`
- Storage request: `50Mi`
- The volume should be bound to `blue-pv-cka` .

Solution:-

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: blue-stc-cka
provisioner: kubernetes.io/no-provisioner
volumeBindingMode: WaitForFirstConsumer
---
apiVersion: v1
kind: PersistentVolume
metadata:
  name: blue-pv-cka
spec:
  capacity:
    storage: 100Mi
  accessModes:
    - ReadWriteOnce
  storageClassName: blue-stc-cka
  persistentVolumeReclaimPolicy: Retain
  local:
    path: /opt/blue-data-cka
```

```
  nodeAffinity:
    required:
      nodeSelectorTerms:
        - matchExpressions:
            - key: kubernetes.io/hostname
              operator: In
              values:
                - controlplane
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: blue-pvc-cka
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: blue-stc-cka
  resources:
    requests:
      storage: 50Mi
  volumeName: blue-pv-cka

kubectl  apply -f <filename>.yaml
```

Weight : 5

5) For this question, please set this context (In exam, diff cluster name)

`kubectl config use-context kubernetes-admin@kubernetes`

A Kubernetes pod definition file named `nginx-pod-cka.yaml` is available. Your task is to make the following modifications to the manifest file:

- Create a Persistent Volume Claim (PVC) with the name `nginx-pvc-cka` . This PVC should request `80Mi` of storage from an existing Persistent Volume (PV) named `nginx-pv-cka` and Storage Class named`nginx-stc-cka` . Use the access mode `ReadWriteOnce` .
- Add the created `nginx-pvc-cka` PVC to the existing `nginx-pod-cka` POD definition.
- Mount the volume claimed by `nginx-pvc-cka` at the path `/var/www/html` within the `nginx-pod-cka` POD.
- Add tolerations with the key `node-role.kubernetes.io/control-plane` set to `Exists` and effect `NoSchedule` to the `nginx-pod-cka` Pod
- Ensure that the `peach-pod-cka05-str` POD is running and that the Persistent Volume (PV) is successfully `bound` .

Solution:-

Step 1: create pvc

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nginx-pvc-cka
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: nginx-stc-cka
  resources:
    requests:
      storage: 80Mi
  volumeName: nginx-pv-cka
```

Run `kubectl apply -f <filename>.yaml`

Step 2: edit nginx-pod-cka.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod-cka
spec:
  containers:
    - name: my-container
      image: nginx:latest
      volumeMounts:
        - name: shared-volume
          mountPath: /var/www/html
  volumes:
    - name: shared-volume
      persistentVolumeClaim:
        claimName: nginx-pvc-cka
  tolerations:
    - key: node-role.kubernetes.io/control-plane
      operator: Exists
      effect: NoSchedule
```

Run `kubectl apply -f nginx-pod-cka.yaml`

Weight : 4

6) For this question, please set this context (In exam, diff cluster name)

```
kubectl config use-context kubernetes-admin@kubernetes
```

A persistent volume named `red-pv-cka` is available. Your task is to create a PersistentVolumeClaim (PVC) named `red-pvc-cka` and request `30Mi` of storage from the `red-pv-cka` PersistentVolume (PV).

Ensure the following criteria are met:

- Access mode: `ReadWriteOnce`
- Storage class: `manual`

Solution:-

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: red-pvc-cka
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: manual
  resources:
    requests:
      storage: 30Mi
  volumeName: red-pv-cka
```

And run `kubectl apply -f <filename>.yaml`

Weight : 4

7) For this question, please set this context (In exam, diff cluster name)

```
kubectl config use-context kubernetes-admin@kubernetes
```

Create a PersistentVolume (PV) named `black-pv-cka` with the following specifications:

- Volume Type: `hostPath`
- Path: `/opt/black-pv-cka`
- Capacity: `50Mi`

Solution:-

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: black-pv-cka
spec:
  capacity:
    storage: 50Mi
  accessModes:
```

```
  - ReadWriteOnce
 hostPath:
  path: /opt/black-pv-cka
```

And run `kubectl apply -f <filename>.yaml`

Weight : 8

8) For this question, please set this context (In exam, diff cluster name)

`kubectl config use-context kubernetes-admin@kubernetes`

Create a PersistentVolume (PV) and a PersistentVolumeClaim (PVC) using an existing storage class named `gold-stc-cka` to meet the following requirements:

Step 1: Create a Persistent Volume (PV)

- Name the PV as `gold-pv-cka` .
- Set the capacity to `50Mi` .
- Use the volume type hostpath with the path `/opt/gold-stc-cka` .
- Assign the storage class as `gold-stc-cka` .
- Ensure that the PV is created on `node01` , where the `/opt/gold-stc-cka` directory already exists.
- Apply a label to the PV with key `tier` and value `white` .

Step 2: Create a Persistent Volume Claim (PVC)

- Name the PVC as `gold-pvc-cka` .
- Request `30Mi` of storage from the PV `gold-pv-cka` using the matchLabels criterion.
- Use the `gold-stc-cka` storage class.
- Set the access mode to `ReadWriteMany` .

Solution:-

Step 1: Create PV

```
apiVersion: v1
kind: PersistentVolume
metadata:
 name: gold-pv-cka
 labels:
  tier: white
spec:
 capacity:
  storage: 50Mi
 volumeMode: Filesystem
```

```
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  storageClassName: gold-stc-cka
  hostPath:
    path: /opt/gold-stc-cka
  nodeAffinity:
    required:
      nodeSelectorTerms:
        - matchExpressions:
            - key: kubernetes.io/hostname
              operator: In
              values:
                - node01
```

And run `kubectl apply -f <filename>.yaml`

Step 2: Create PVC

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: gold-pvc-cka
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: gold-stc-cka
  selector:
    matchLabels:
      tier: white
  volumeName: gold-pv-cka
  resources:
    requests:
      storage: 30Mi
```

And run `kubectl apply -f <filename>.yaml`

Weight : 2

9) For this question, please set this context (In exam, diff cluster name)

`kubectl config use-context kubernetes-admin@kubernetes`

Create a storage class called `green-stc` as per the properties given below:

- Provisioner should be `kubernetes.io/no-provisioner` . - Volume binding mode should be `WaitForFirstConsumer` .

- Volume expansion should be `enabled` .

Solution:-

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: green-stc
provisioner: kubernetes.io/no-provisioner
volumeBindingMode: WaitForFirstConsumer
allowVolumeExpansion: true
```

And run `kubectl apply -f <filename>.yaml`

Weight : 2

10) For this question, please set this context (In exam, diff cluster name)

```
kubectl config use-context kubernetes-admin@kubernetes
```

You are responsible for provisioning storage for a Kubernetes cluster. Your task is to create a PersistentVolume (PV), a PersistentVolumeClaim (PVC), and deploy a pod that uses the PVC for shared storage.

Here are the specific requirements:

- Create a PersistentVolume (PV) named `my-pv-cka` with the following properties:
  - Storage capacity: `100Mi`
  - Access mode: `ReadWriteOnce`
  - Host path: `/mnt/data`
  - Storage class: `standard`
- Create a PersistentVolumeClaim (PVC) named `my-pvc-cka` to claim storage from the `my-pv-cka` PV, with the following properties:
  - Storage class: `standard`
- Deploy a pod named `my-pod-cka` using the `nginx` container image.
- Mount the PVC, `my-pvc-cka` , to the pod at the path `/var/www/html` . Ensure that the PV, PVC, and pod are successfully created, and the pod is in a Running state.

Note: Binding and Pod might take time to come up, please have patience

Solution:-

Step 1: Create PV

```
apiVersion: v1
kind: PersistentVolume
metadata:
```

```
  name: my-pv-cka
spec:
  capacity:
    storage: 100Mi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: /mnt/data
  storageClassName: standard
```

Step 2: Create PVC

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc-cka
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  volumeName: my-pv-cka
  storageClassName: standard
```

Step 3: Create pod

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod-cka
spec:
  containers:
    - name: nginx-container
      image: nginx
      volumeMounts:
        - name: shared-storage
          mountPath: /var/www/html
  volumes:
    - name: shared-storage
      persistentVolumeClaim:
        claimName: my-pvc-cka
```
