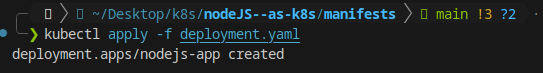
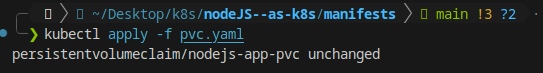
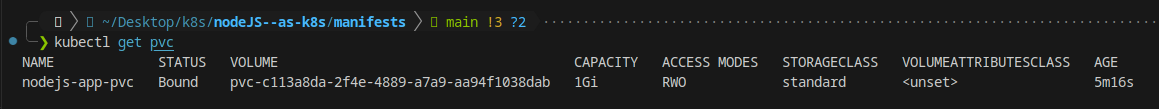
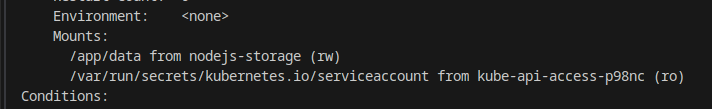
## 📄 Task 2: Provision and Mount Storage

**Objective:** Provision a PersistentVolumeClaim (PVC), mount it inside the container, and verify persistent storage.

### 🛠 Steps Taken

1. Deploy new deployment with volume mounted
2. Defined a PVC in pvc.yaml requesting 1Gi of storage.
3. Applied the PVC manifest.
4. Verified the PVC reached **Bound** status.
5. Updated the Deployment to mount the volume inside the container at /app/data.
6. Verified volume mounting via kubectl describe pod.
7. Wrote a file inside the mounted volume using kubectl exec, then read it to verify persistence.

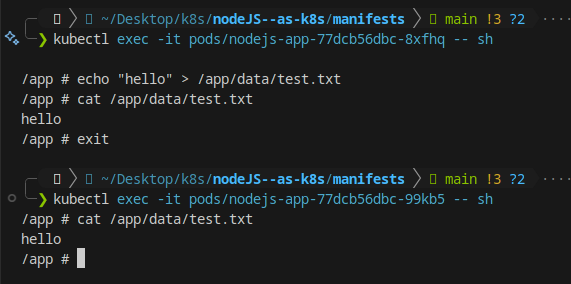
### 📸 Screenshots

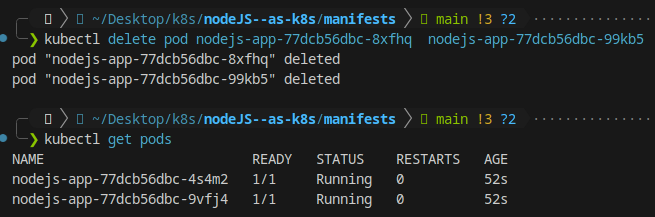
* **Screenshot 1:** kubectl apply -f deployment.yaml  
  (Deploy new deployment with volume mounted)
* **Screenshot 2:** kubectl apply -f pvc.yaml  
  (PVC creation)
* **Screenshot 3:** kubectl get pvc  
  (Shows that PVC is in Bound state)
* **Screenshot 4:** kubectl describe pod <pod-name>  
  (Verifies volume is mounted at /app/data)
* **Screenshot 5:** kubectl exec and file operations  
  Commands:

kubectl exec -it <pod-name> -- sh

echo "hello" > /app/data/test.txt

cat /app/data/test.txt



* **Screenshot 6 (optional):** Recreate Pod and check file again  
  (Proves data persisted across Pod restarts)

### ✅ Outcome

The application successfully mounted the persistent volume. Files written to the volume remained intact even after restarting the Pod, proving that persistent storage works correctly.