## 📄 Task 4: Implement Rolling Updates

### 🎯 Objective

Configure the Deployment to use **RollingUpdate** strategy with controlled surge and availability, then trigger an image version change and observe the rollout.

### 🛠 Steps Taken

1. Edited the Deployment manifest to include the following rolling update strategy:

strategy:

type: RollingUpdate

rollingUpdate:

maxSurge: 1

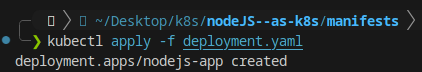
maxUnavailable: 0

1. Applied the updated Deployment using:  
   kubectl apply -f deployment.yaml
2. Triggered a new rollout by updating the container image:  
   kubectl set image deployment/nodejs-app nodejs=aw033/nodejs-hello:v1
3. Monitored the rollout progress:  
   kubectl rollout status deployment/my-app
4. Verified updated pods and running image version:  
   kubectl get pods -o wide
5. Described the Deployment to inspect rollout details:  
   kubectl describe deploy nodejs-app
6. Performed a rollback to the previous version:  
   kubectl rollout undo deployment/nodejs-app
7. Verified the rollback via Deployment description:  
   kubectl describe deploy nodejs-app

### 📸 Screenshots

• **Screenshot 1:**

kubectl apply -f deployment.yaml



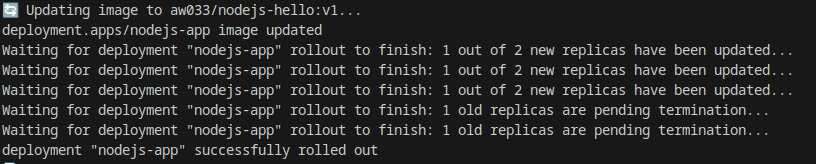
• **Screenshot 2:**

kubectl set image deployment/nodejs-app nodejs=aw033/nodejs-hello:v1



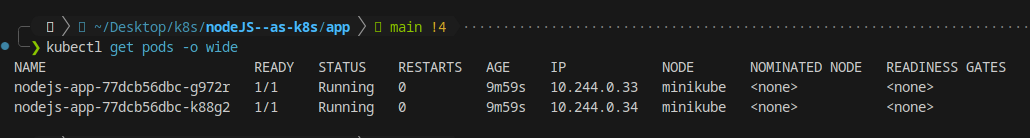
• **Screenshot 3:**

kubectl rollout status deployment/my-app



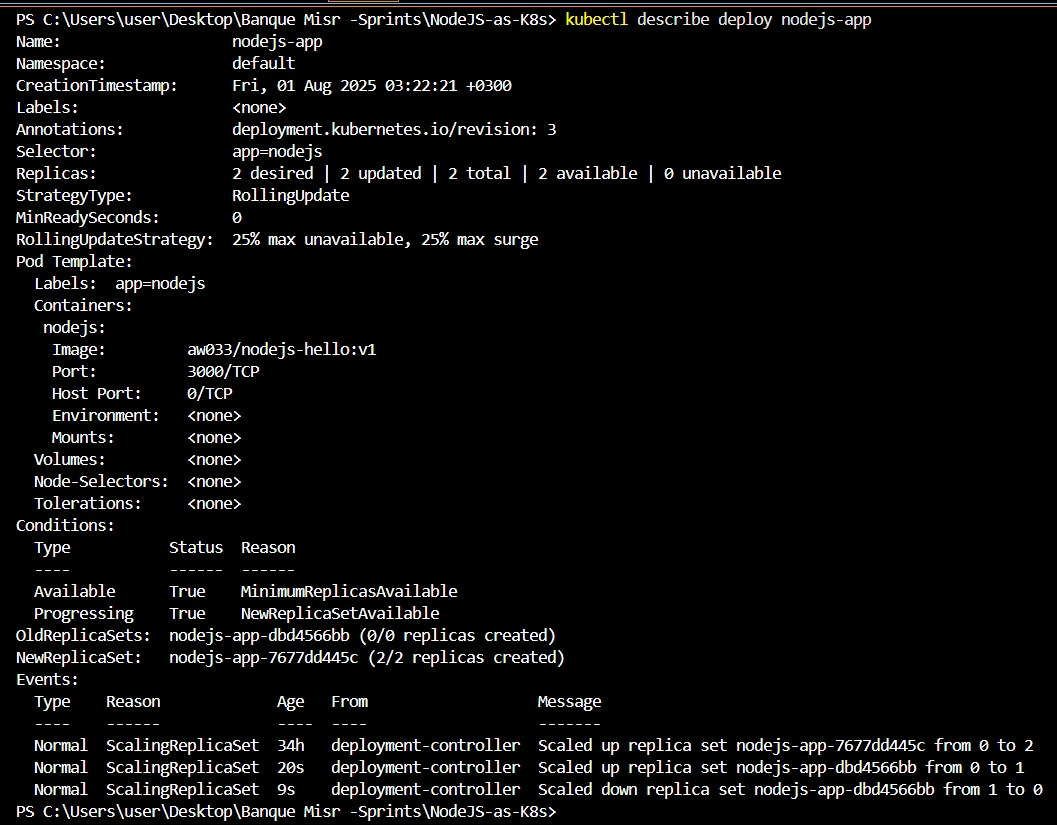
• **Screenshot 4:**

kubectl get pods -o wide



• **Screenshot 5:**

kubectl describe deploy nodejs-app



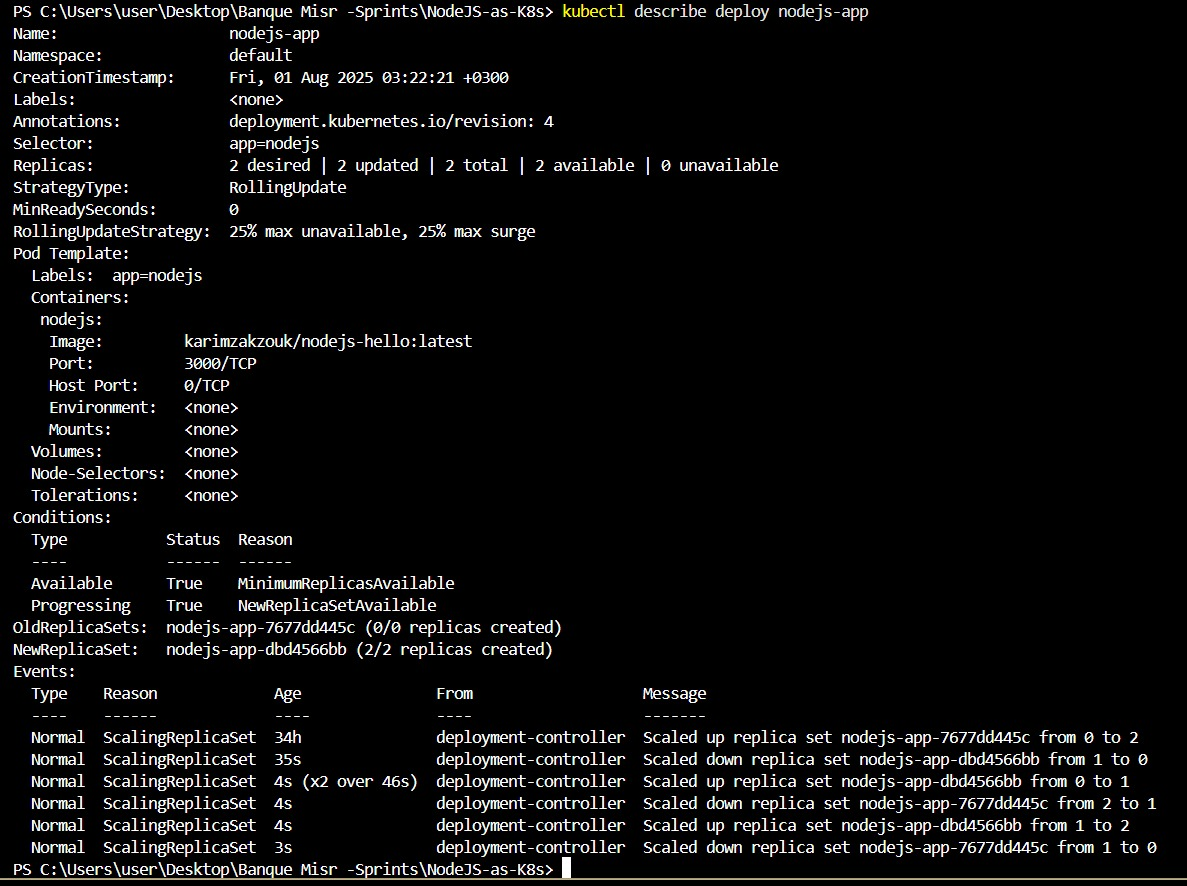
• **Screenshot 6:**

kubectl rollout undo deployment/nodejs-app



• **Screenshot 7:**

kubectl describe deploy nodejs-app



### ✅ Outcome

The Deployment used a **RollingUpdate** strategy successfully. The application was updated incrementally with zero downtime, and all pods transitioned to the new version while maintaining availability.