## 📄 Task 3: Expose with LoadBalancer

**Objective:** Expose the Node.js application to external traffic by creating a LoadBalancer-type service.

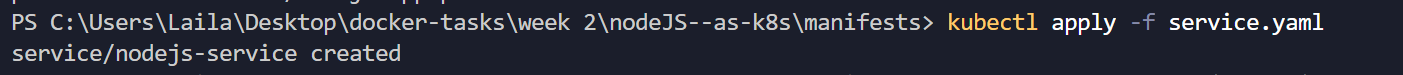
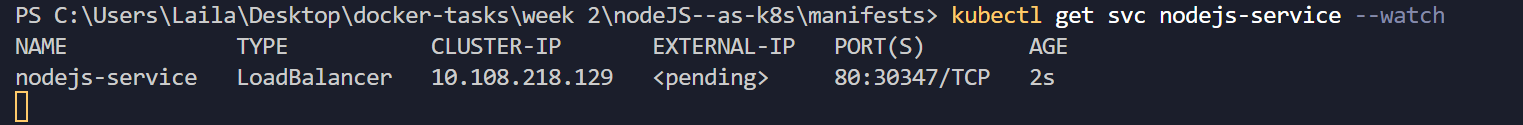
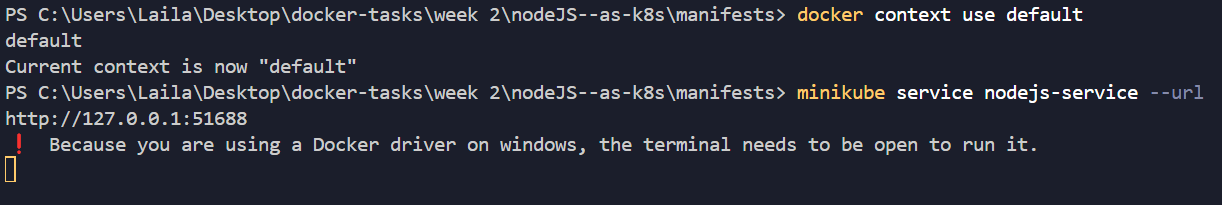
### 🛠 Steps Taken

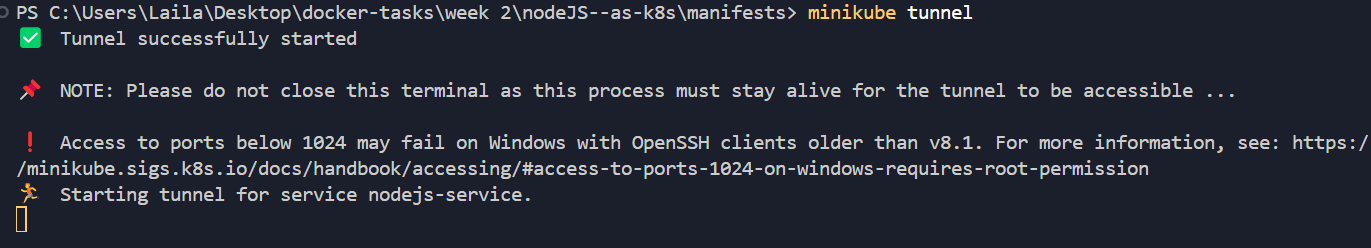
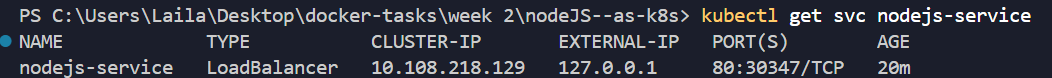
1. Created a Service manifest (service.yaml) of type LoadBalancer to expose port **80** mapped to container port **3000**.
2. Applied the Service manifest using **kubectl apply -f service.yaml**
3. **On local device:**

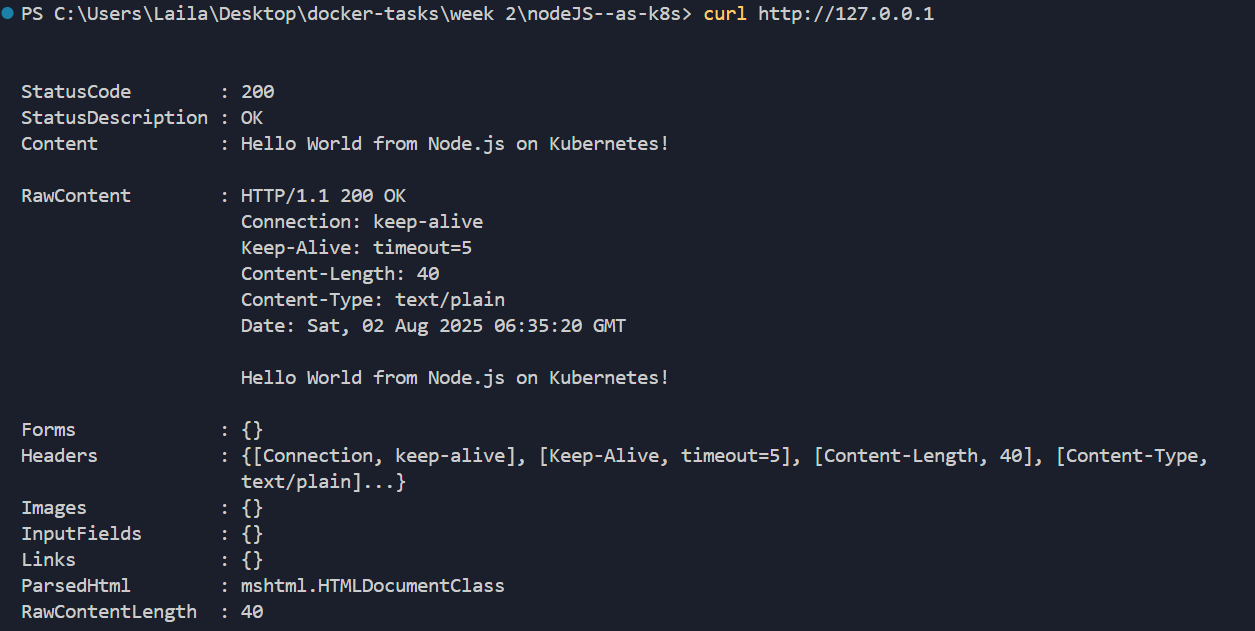
Minikube automatically routes loadbalancer on local devices, which makes the external IP appear pending forever. The external IP can be retrieved using **minikube service nodejs-service --url**, which opens a temporary local proxy to expose the service. Alternatively, **minikube tunnel** can be used to simulate a real cloud LoadBalancer by assigning a routable external IP.

1. Confirmed LoadBalancer service routing using **curl <external-ip>**

### 📸 Screenshots

* **Screenshot 1:** kubectl apply -f service.yaml  
  
* **Screenshot 2:** kubectl get svc nodejs-service –watch  
  - The External-IP will be pending forever if minikube tunnel isn’t enabled
* **Screenshot 3 (Option A):** minikube service nodejs-serivce --url  
  Get the external IP  
  

**Screenshot 4 (Option B):**  
a - minikube tunnel  
  
b- kubectl get svc nodejs-service  
 

**Screenshot 5:** curl <http://127.0.01>  
Verifying the external IP  


### ✅ Outcome

The application was successfully exposed to external traffic using a LoadBalancer service. It was accessible via the provided external URL.