**Day 7**

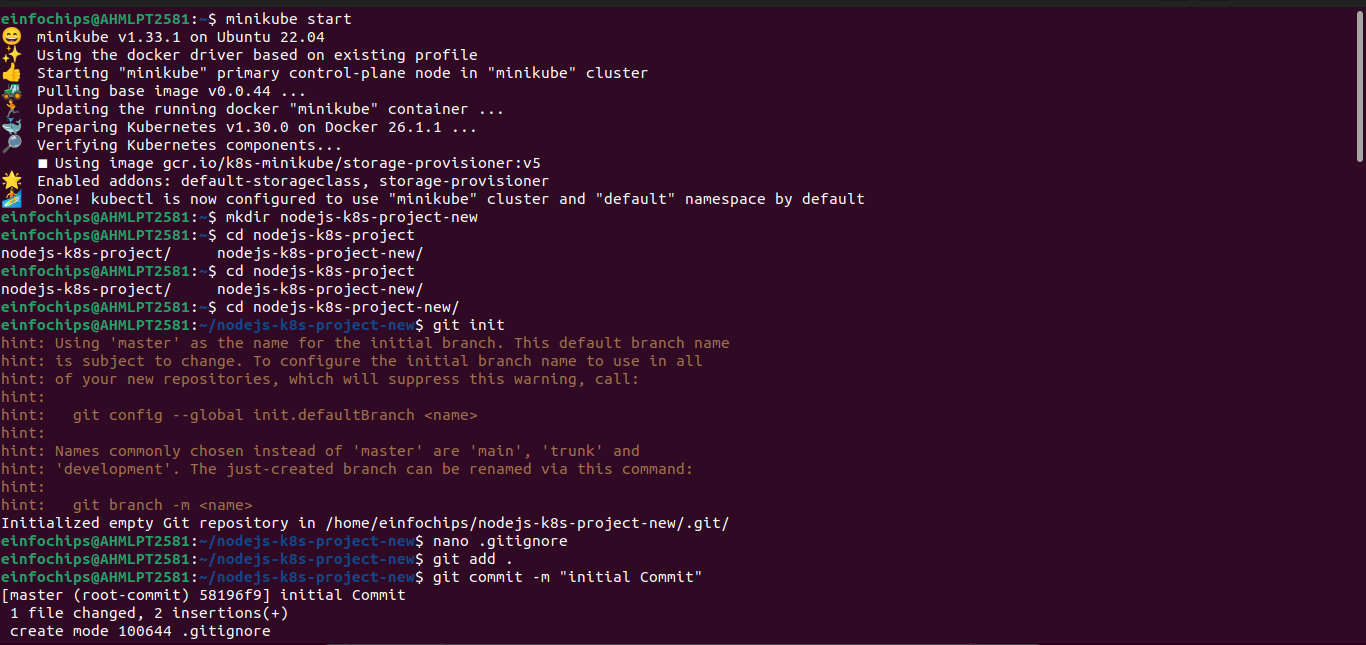
**Project 01**

In this project, you will develop a simple Node.js application, deploy it on a local Kubernetes cluster using Minikube, and configure various Kubernetes features. The project includes Git version control practices, creating and managing branches, and performing rebases. Additionally, you will work with ConfigMaps, Secrets, environment variables, and set up vertical and horizontal pod autoscaling.

## **Project 01**

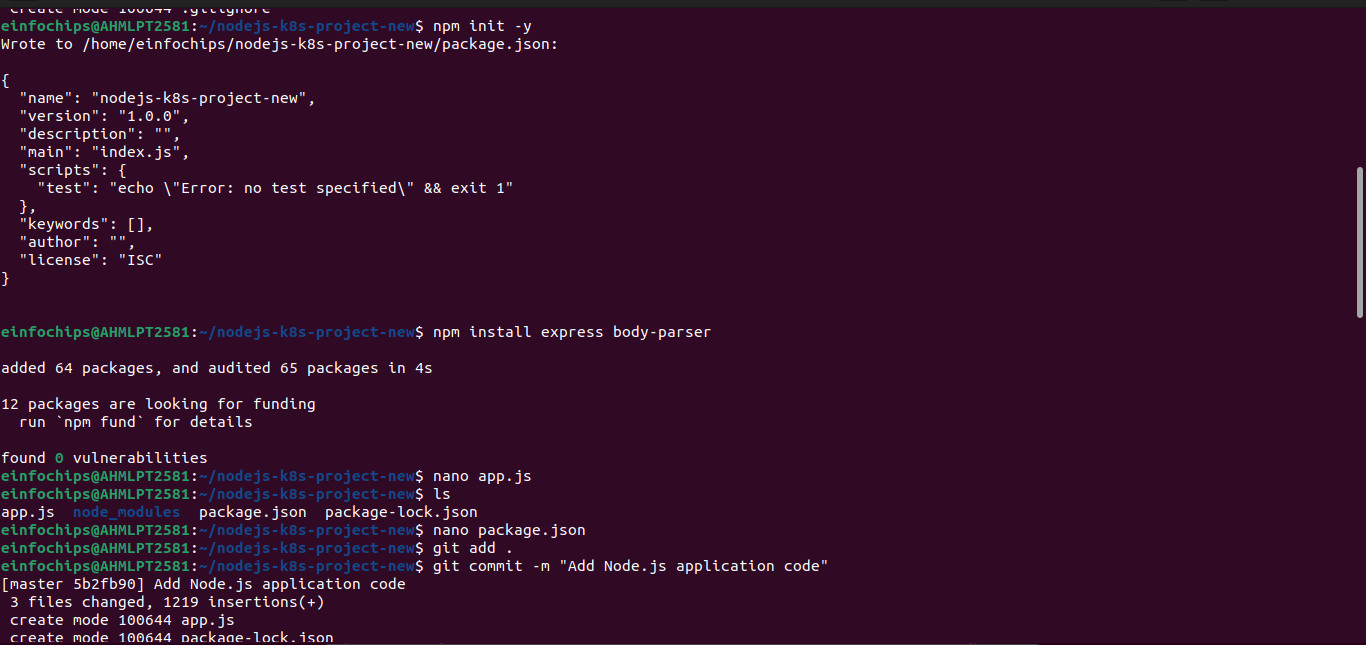
## **Project Steps**

### **1. Setup Minikube and Git Repository**

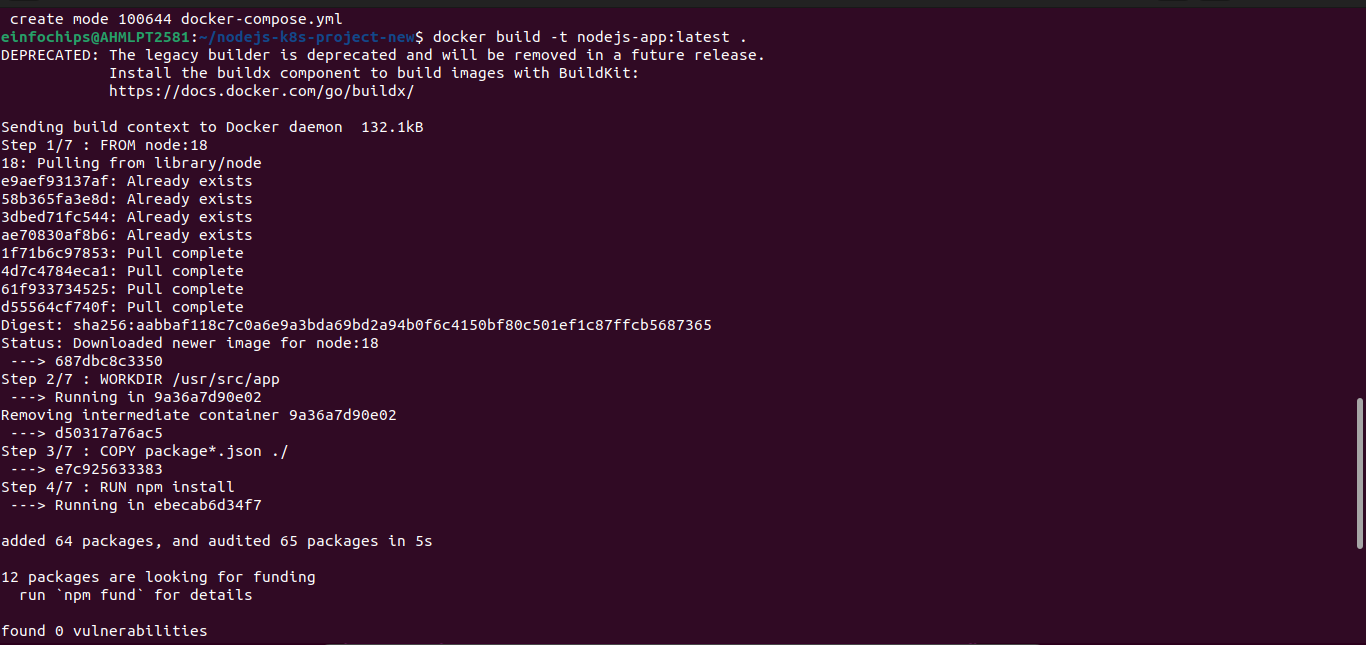


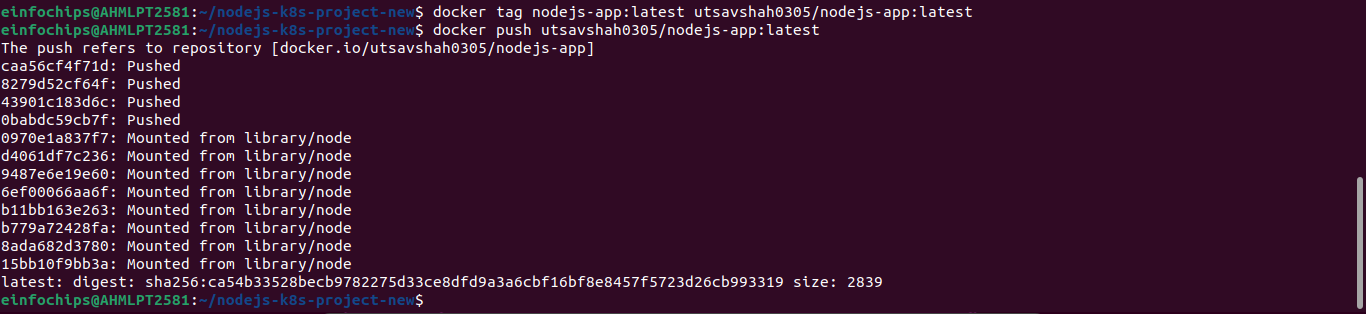
### **2. Develop a Node.js Application**

#### **2.1 Create the Node.js App**

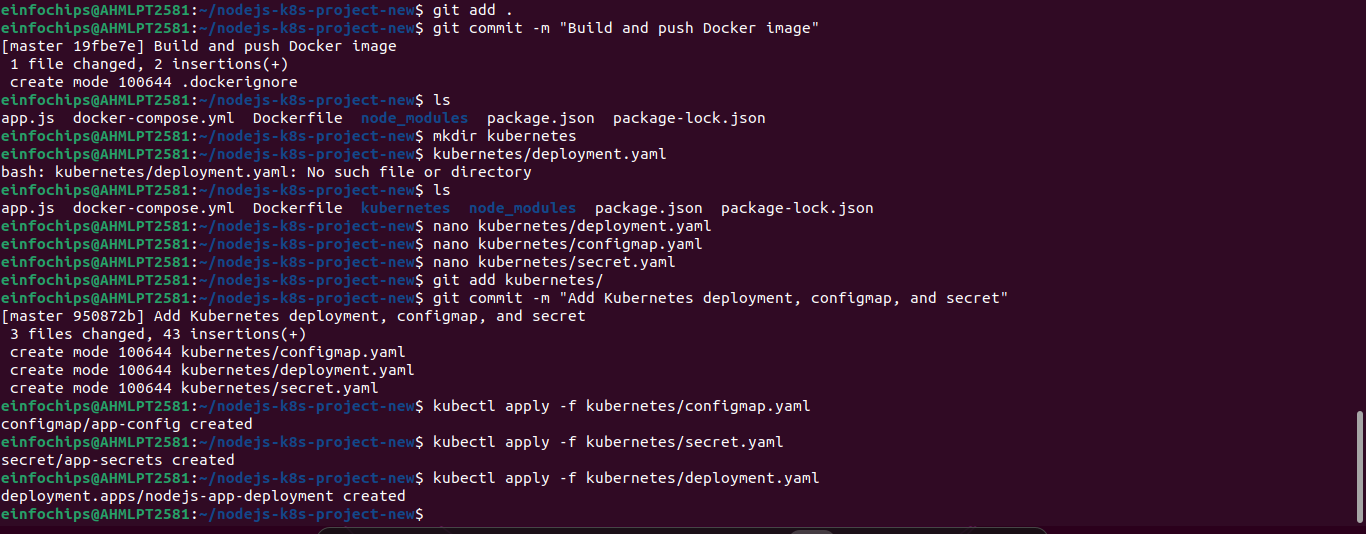
**Initialize the Node.js project**:

### **4. Build and Push Docker Image**



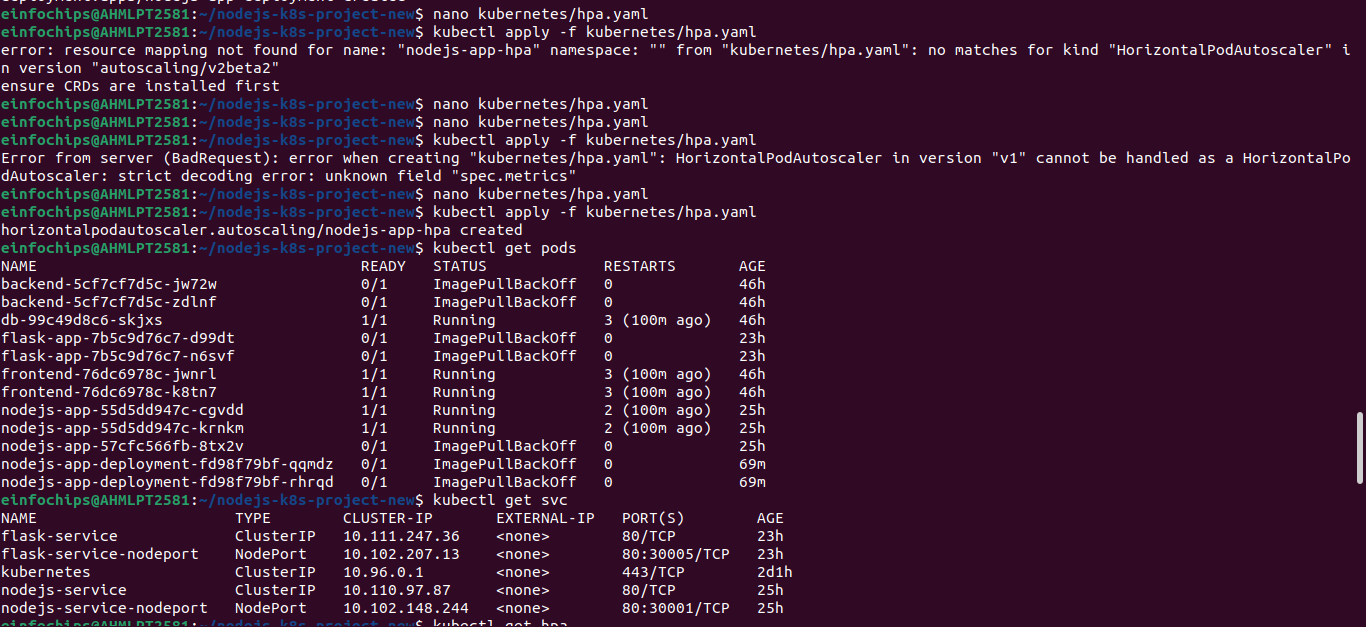


### **5. Create Kubernetes Configurations**

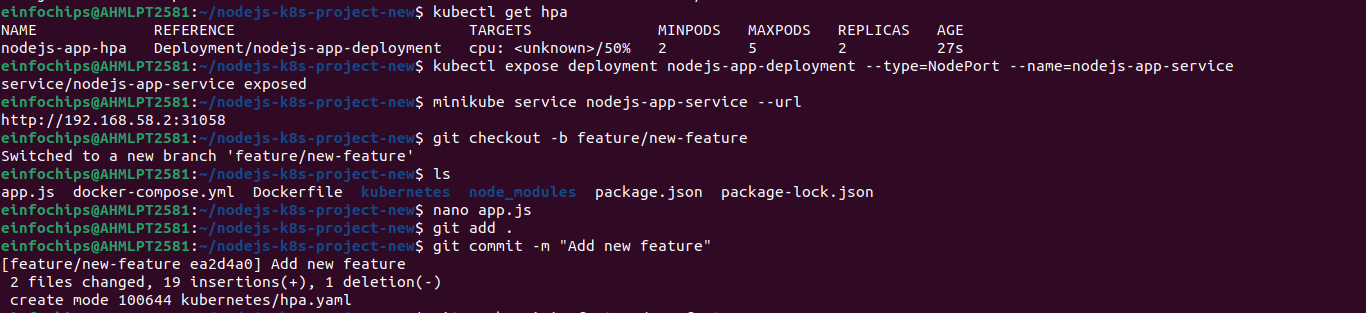


### **6. Implement Autoscaling**

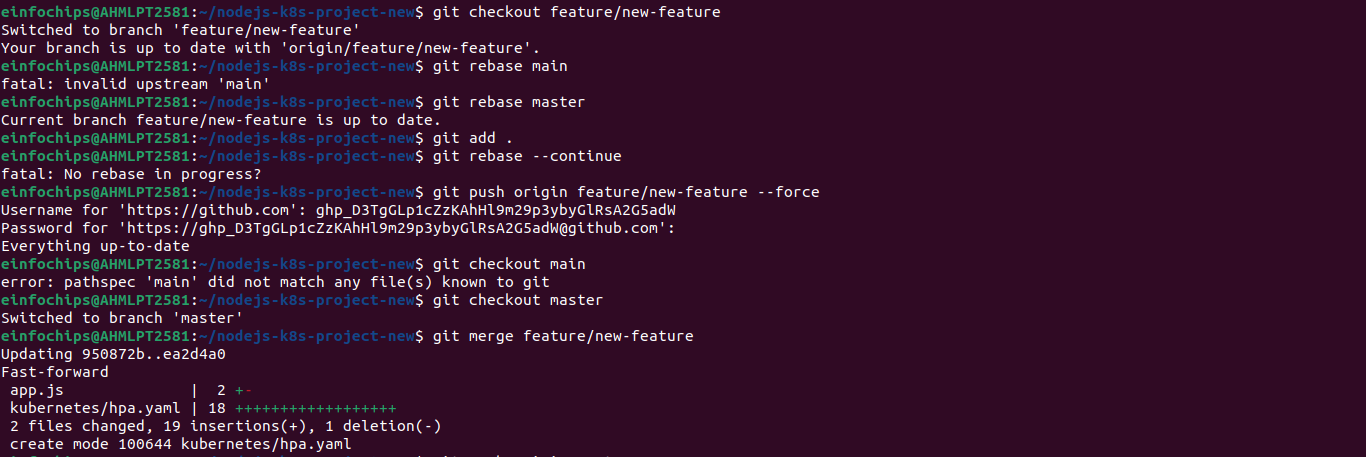
### **7. Test the Deployment**

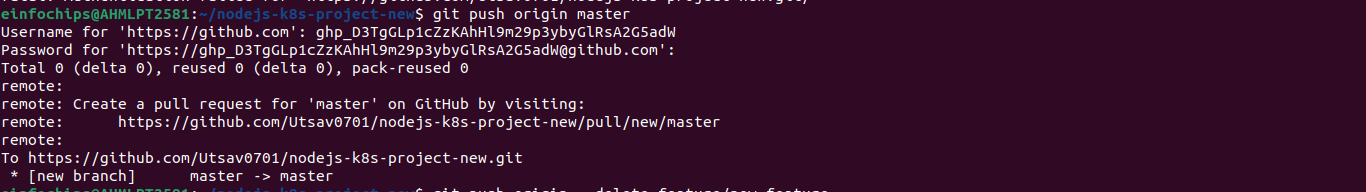


### **8. Git Version Control**



### **9. Final Commit and Cleanup**



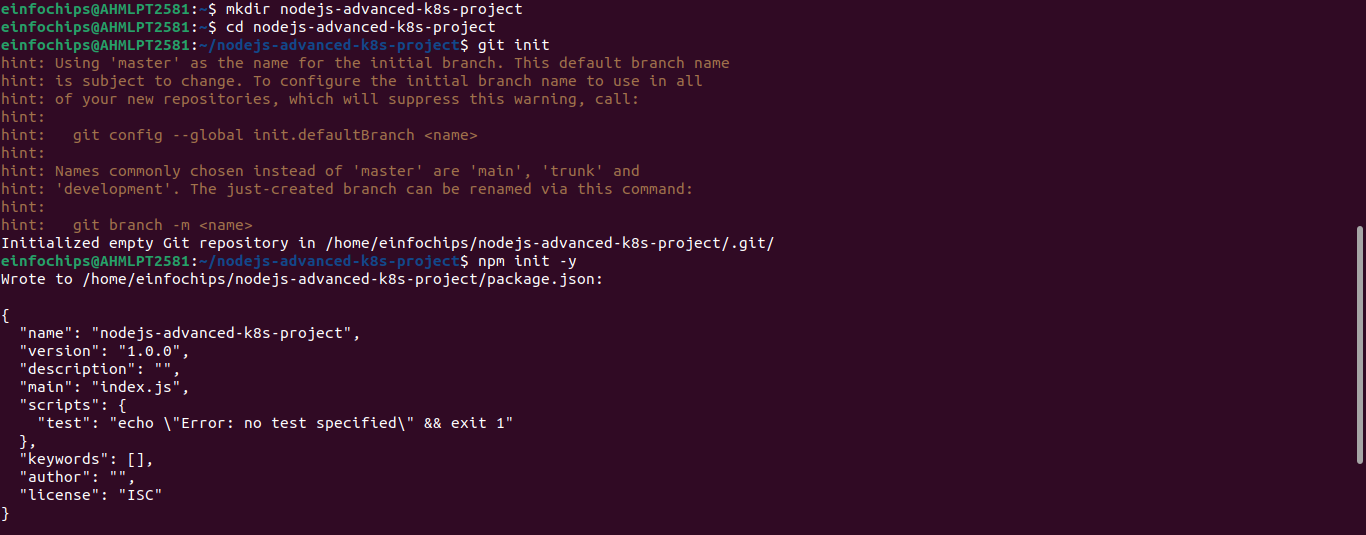


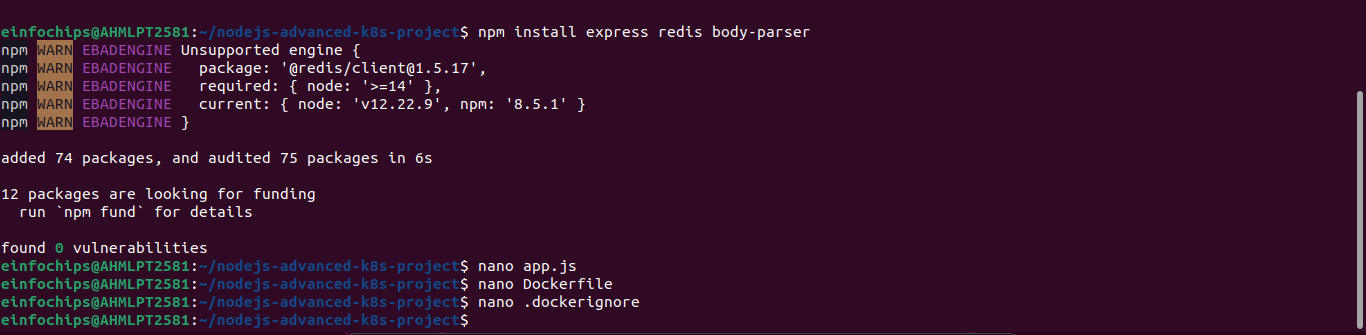
**Project 02**

Deploy a Node.js application to Kubernetes with advanced usage of ConfigMaps and Secrets. Implement Horizontal Pod Autoscaler (HPA) with both scale-up and scale-down policies. The project will include a multi-environment configuration strategy, integrating a Redis cache, and monitoring application metrics.

## **Project Setup**

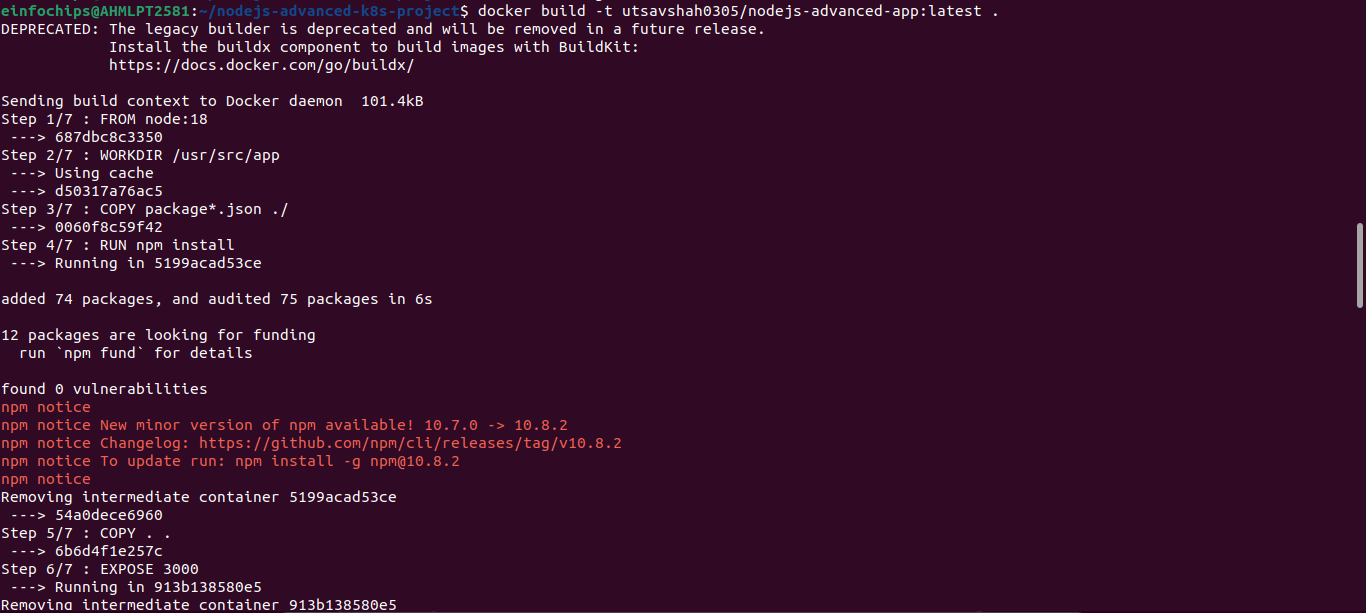
### **1.1 Initialize a Git Repository**





**1. Build and push Docker image:**

docker build -t utsavshah0305/nodejs-advanced-app:latest .

****

**2. Advanced Kubernetes Configuration**

**2.1 Deployment Configuration**

### **2.2 ConfigMap for Application and Redis**

### **2.3 Secret for Sensitive Data**

### **2.4 Service Configuration**

### **2.5 Horizontal Pod Autoscaler with Scale-Up and Scale-Down Policies**

### **2.6 Vertical Pod Autoscaler Configuration**

### **2.7 Redis Deployment**

### **2.8 Apply Kubernetes Configurations**

Apply all configurations to your Minikube cluster:

kubectl apply -f kubernetes/redis-deployment.yaml

kubectl apply -f kubernetes/redis-service.yaml

kubectl apply -f kubernetes/configmap.yaml

kubectl apply -f kubernetes/secret.yaml

kubectl apply -f kubernetes/deployment.yaml

kubectl apply -f kubernetes/service.yaml

kubectl apply -f kubernetes/hpa.yaml

kubectl apply -f kubernetes/vpa.yaml

### **2.9 Verify Deployments and Services**

Check the status of your deployments and services:

kubectl get all

