### **Project 01**

### **Deploying a Node.js App Using Minikube Kubernetes**

#### **Overview**

This project guides you through deploying a Node.js application using Minikube Kubernetes. You'll use Git for version control, explore branching and fast-forward merges, and set up Kubernetes services and deployment pods, including ClusterIP and NodePort service types.

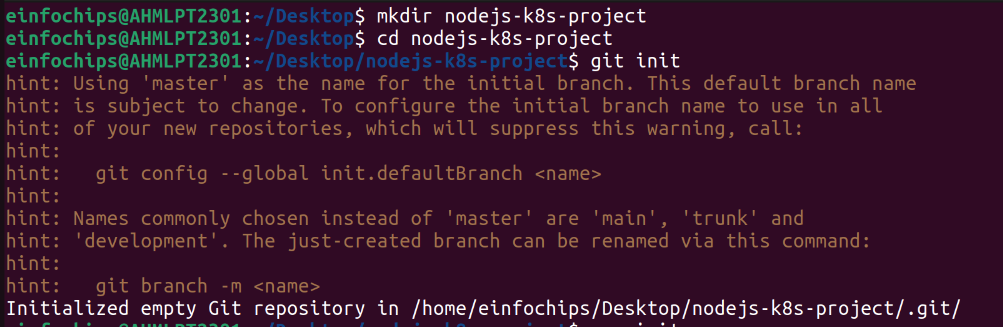
#### **Prerequisites**

* Minikube installed
* kubectl installed
* Git installed
* Node.js installed ([https://nodejs.org/en/download/package-manager/all#debian-and-ubuntu-based-linux-distributions](https://nodejs.org/en/download/package-manager/all" \l "debian-and-ubuntu-based-linux-distributions))

#### **Project Steps**

### **1. Set Up Git Version Control**

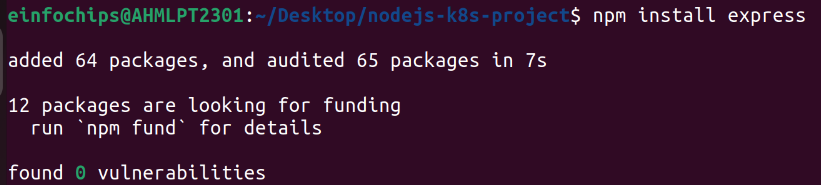
**1.1. Initialize a Git Repository**

****

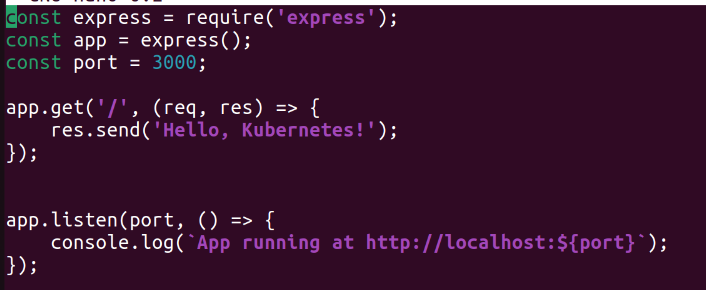
**1.2. Create a Node.js Application**



Install Express.js:



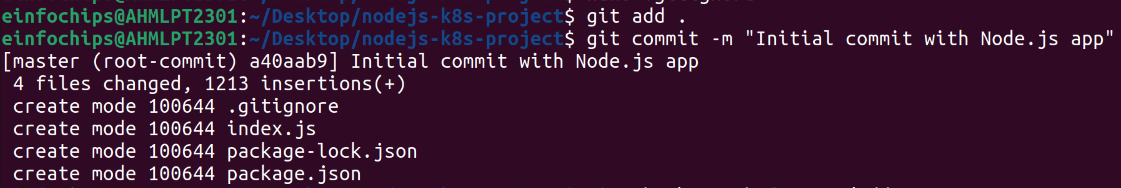
Create an index.js file with the following content:



Create a .gitignore file to ignore node\_modules:



**1.3. Commit the Initial Code**



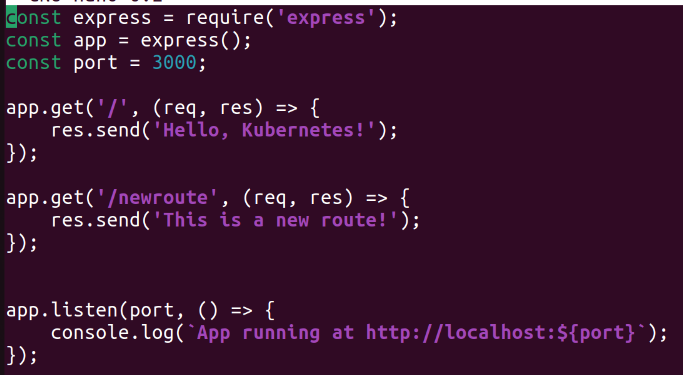
### **2. Branching and Fast-Forward Merge**

**2.1. Create a New Branch**



**2.2. Implement a New Route**

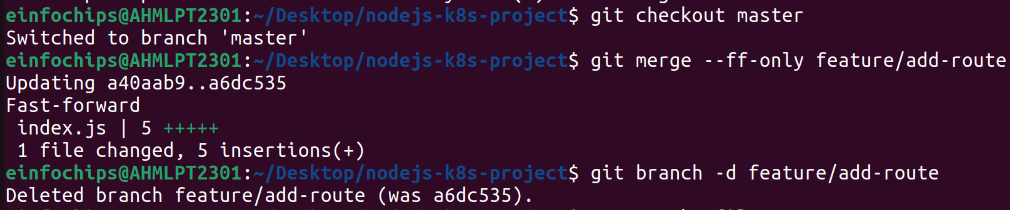
Modify index.js to add a new route:



Commit the changes:

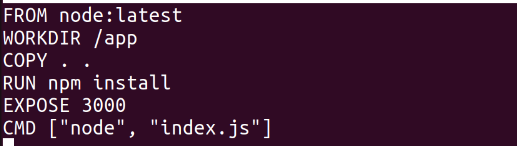


**2.3. Merge the Branch Using Fast-Forward**



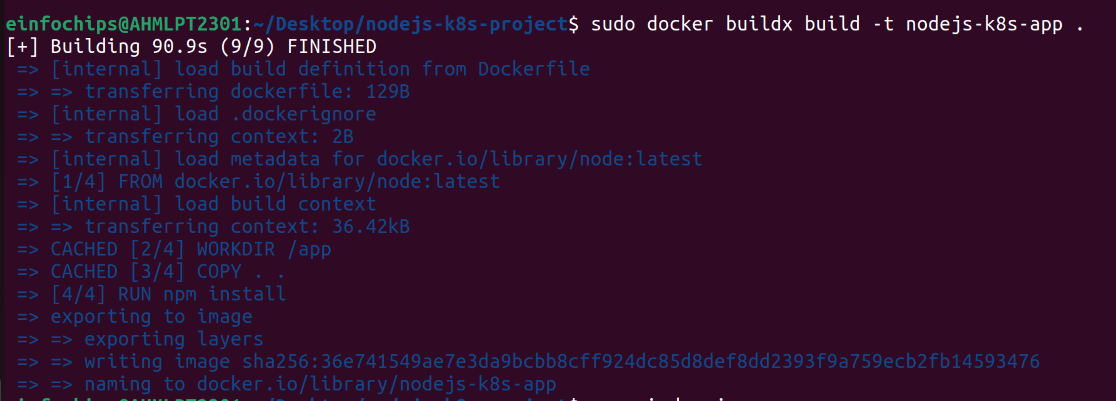
### **3. Containerize the Node.js Application**

**3.1. Create a Dockerfile**

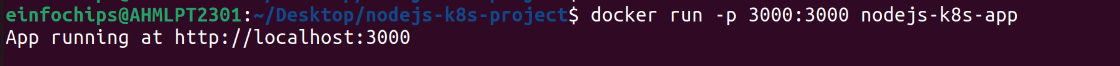
****

**3.2. Build and Test the Docker Image**

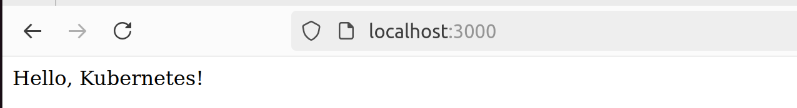
Build the Docker image:



Run the Docker container to test:

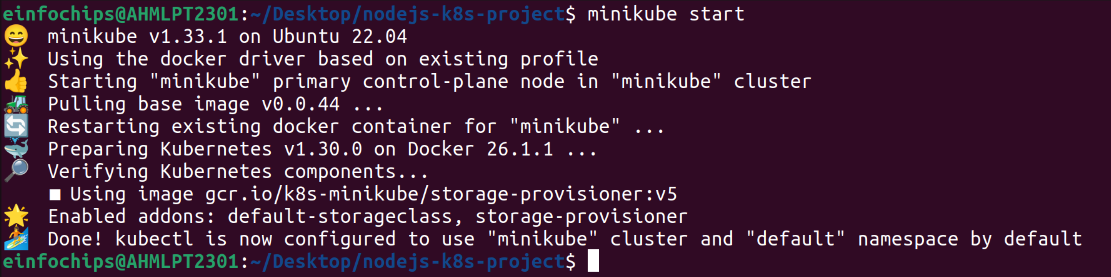


Access http://localhost:3000 to see the app running.



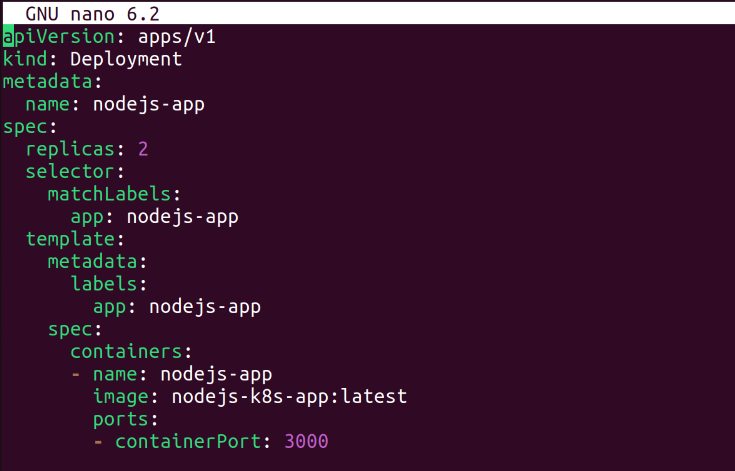
### **4. Deploying to Minikube Kubernetes**

**4.1. Start Minikube**

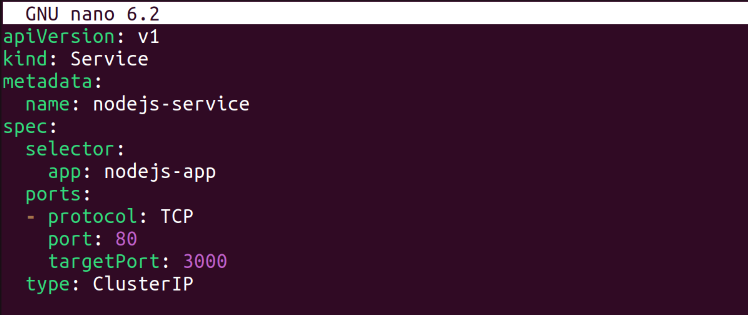


**4.2. Create Kubernetes Deployment and Service Manifests**

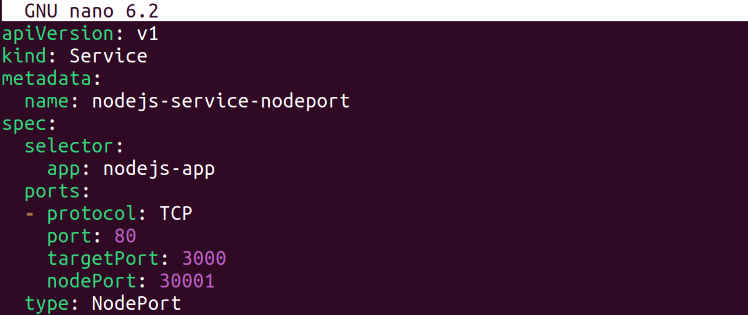
Create a deployment.yaml file:



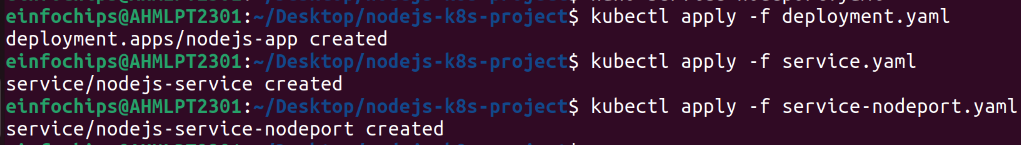
Create a service.yaml file for ClusterIP:



Create a service-nodeport.yaml file for NodePort:



**4.3. Apply Manifests to Minikube**

****

First Apply the Deployment

Then Apply the ClusterIP service

Then Apply the NodePort service

**4.4. Access the Application**

****

First we Get the Minikube IP



Then Access the application using the NodePort.

### **Making Changes to the App and Redeploying Using Kubernetes**

### **6. Making Changes to the Node.js Application**

**6.1. Create a New Branch for Changes**

Create and switch to a new branch feature/update-message:



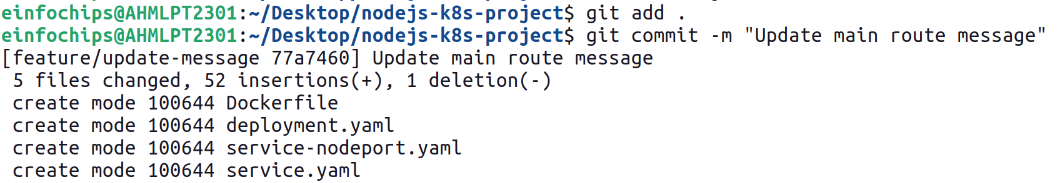
**6.2. Update the Application**

Modify index.js to change the message:

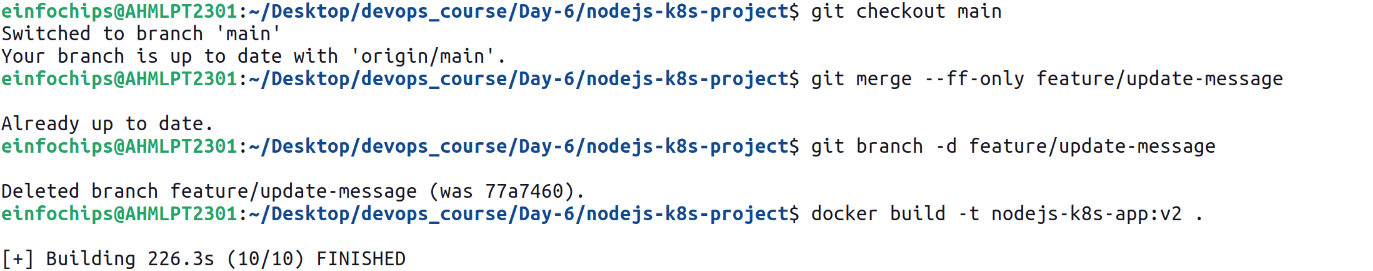


**6.3. Commit the Changes**

Add and commit the changes:



### **7. Merge the Changes and Rebuild the Docker Image**

****

### **8. Update Kubernetes Deployment**

**8.1. Update the Deployment Manifest**



**8.2. Apply the Updated Manifest**



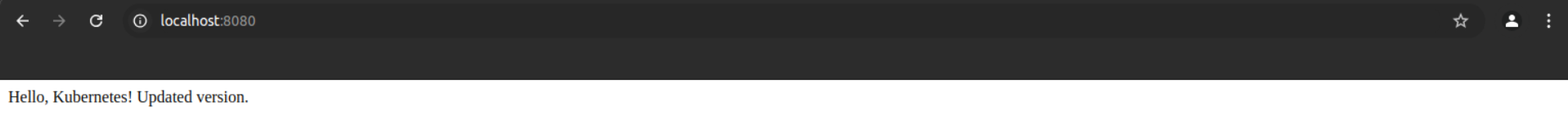
**8.3. Verify the Update**

****

### **9. Access the Updated Application**

**9.1. Access Through ClusterIP Service**



****

**9.2. Access Through NodePort Service**



**Project 02**

### **Deploying a Python Flask App Using Minikube Kubernetes**

#### **Overview**

This project guides you through deploying a Python Flask application using Minikube Kubernetes. You'll use Git for version control, explore branching and fast-forward merges, and set up Kubernetes services and deployment pods, including ClusterIP and NodePort service types.

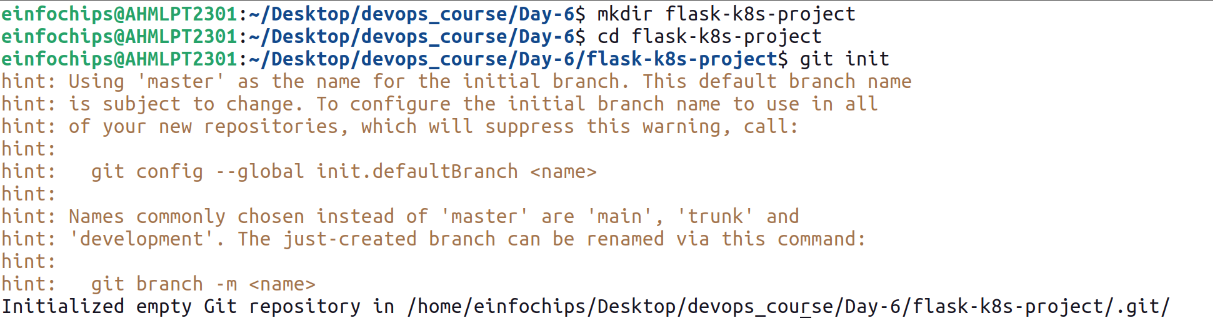
#### **Prerequisites**

* Minikube installed
* kubectl installed
* Git installed
* Python installed

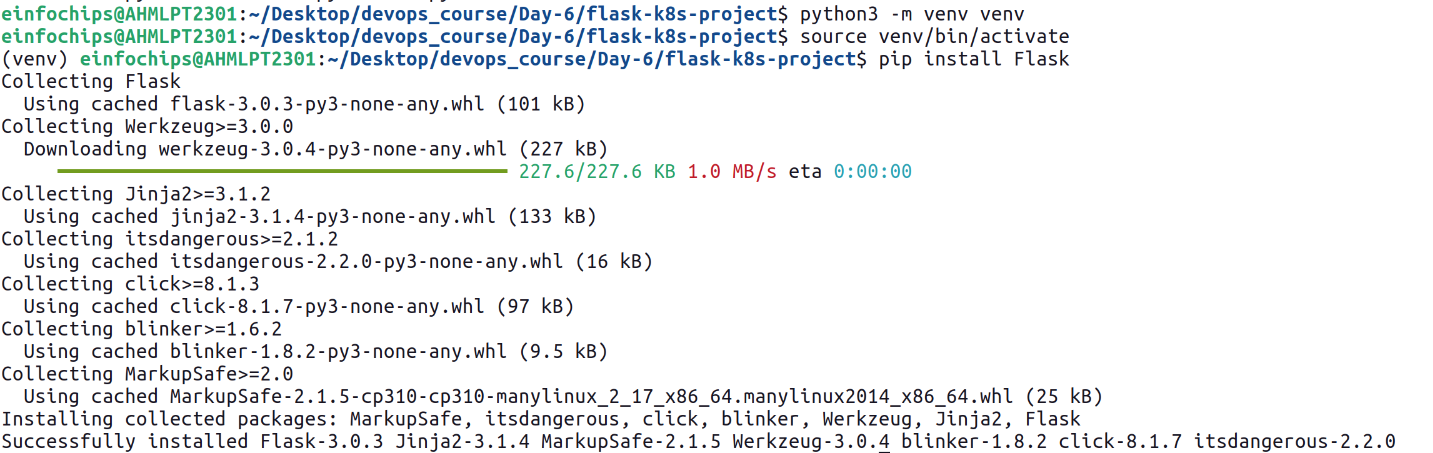
#### **Project Steps**

### **1. Set Up Git Version Control**

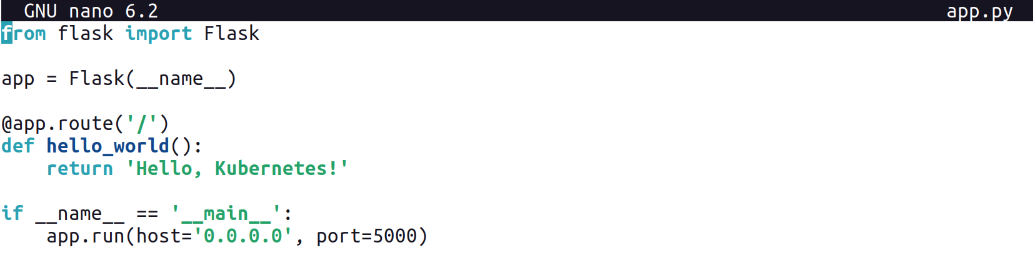
**1.1. Initialize a Git Repository**



**1.2. Create a Python Flask Application**



Create an app.py file with the following content:

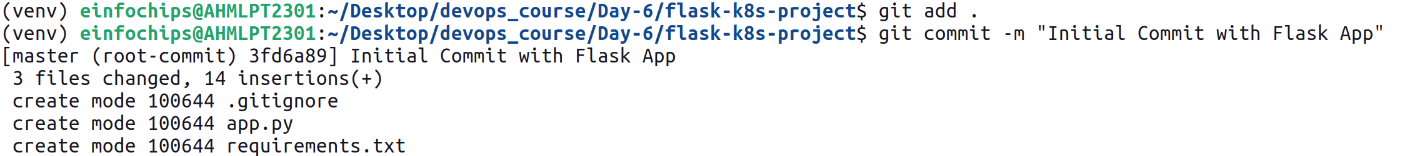
Create a requirements.txt file to list the dependencies:



Create a .gitignore file to ignore venv:

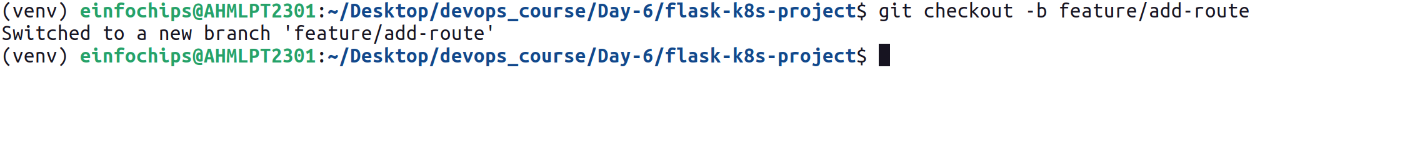


**1.3. Commit the Initial Code**

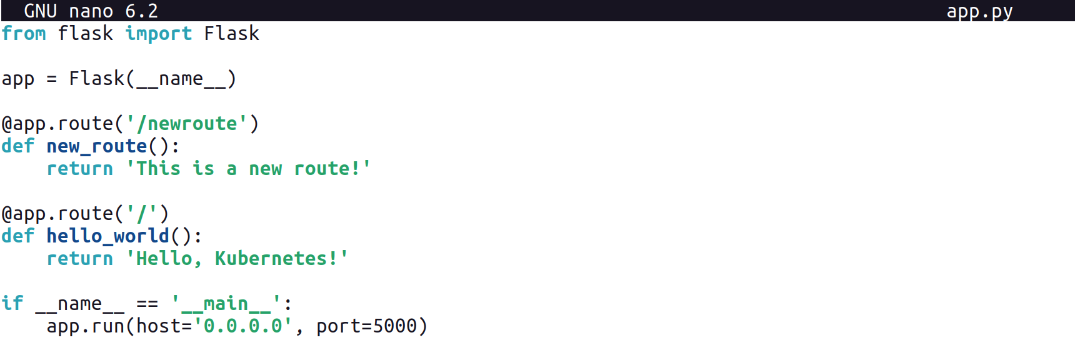


### **2. Branching and Fast-Forward Merge**

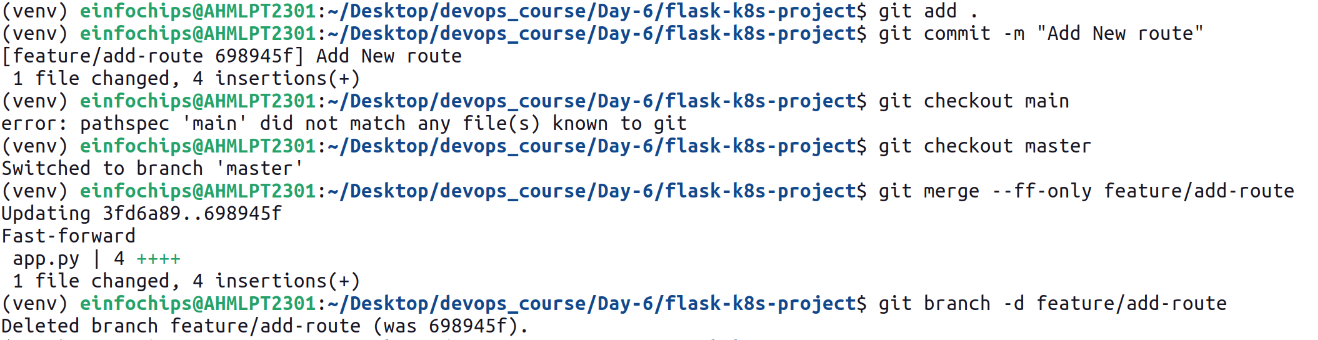
**2.1. Create a New Branch**

****

**2.2. Implement a New Route**

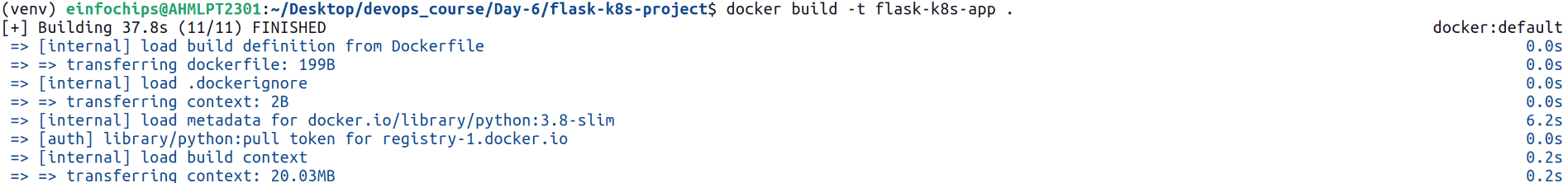


**2.3. Merge the Branch Using Fast-Forward**



### **3. Containerize the Flask Application**

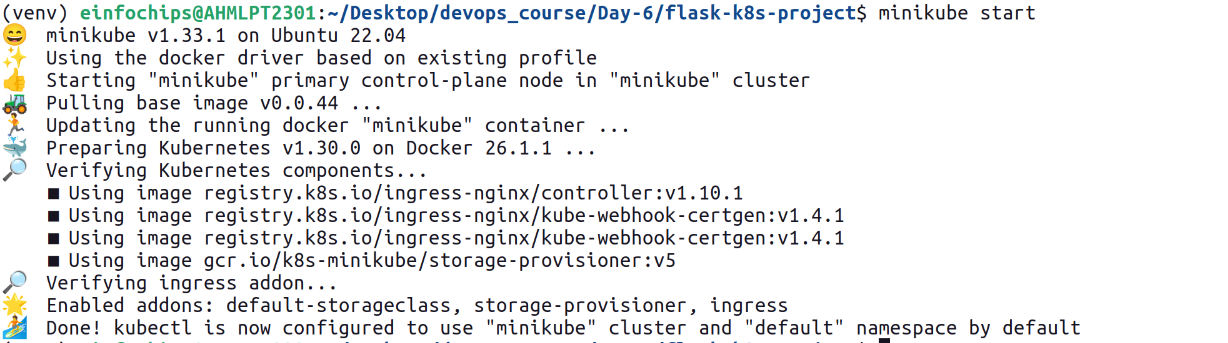
**3.1. Create a Dockerfile**

**3.2. Build and Test the Docker Image**



### **4. Deploying to Minikube Kubernetes**

**4.1. Start Minikube**

****

**4.2. Create Kubernetes Deployment and Service Manifests**

Create a deployment.yaml file:

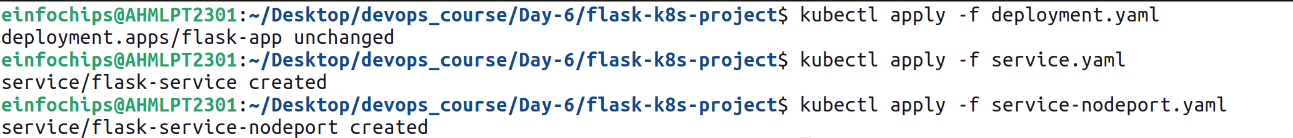


Create a service.yaml file for ClusterIP:

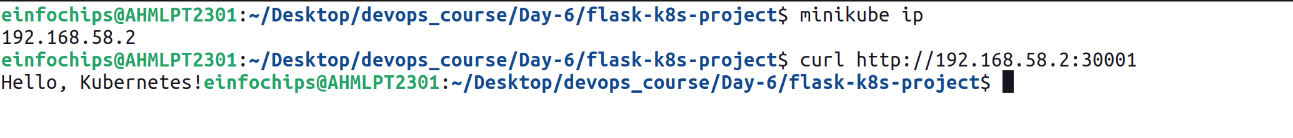


Create a service-nodeport.yaml file for NodePort:

**4.3. Apply Manifests to Minikube**



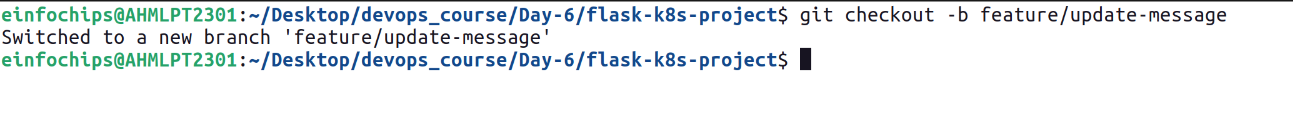
**4.4. Access the Application**



### **5. Making Changes to the Flask Application**

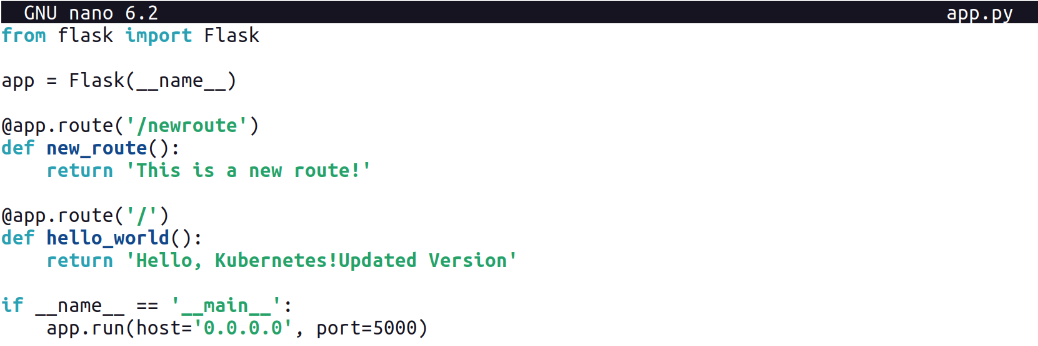
**5.1. Create a New Branch for Changes**

Create and switch to a new branch feature/update-message:

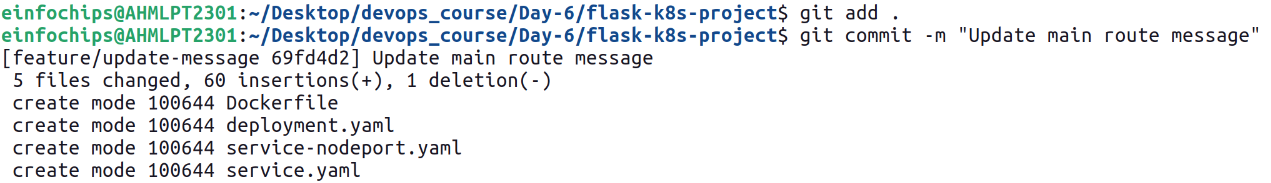


**5.2. Update the Application**

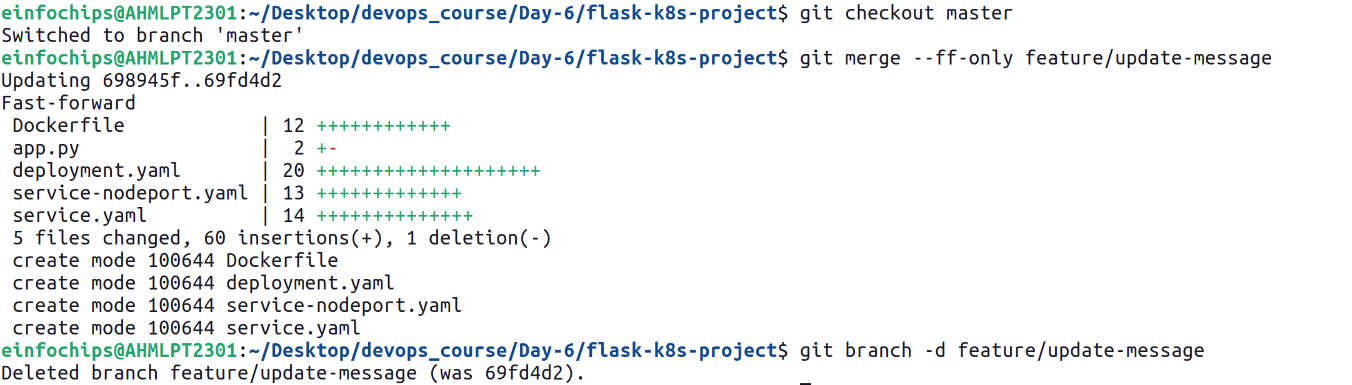
Modify app.py to change the message:



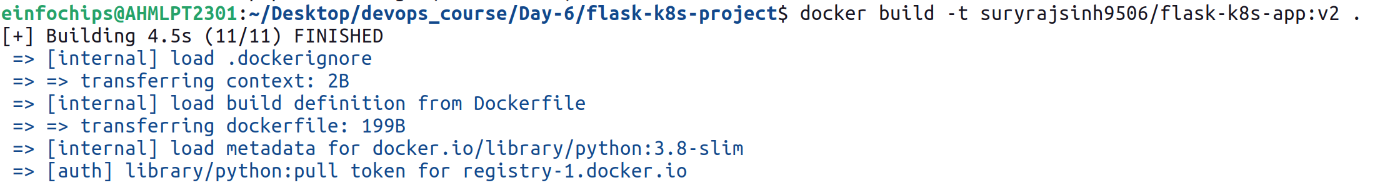
**5.3. Commit the Changes**

**6. Merge the Changes and Rebuild the Docker Image**

**6.1. Merge the Feature Branch**

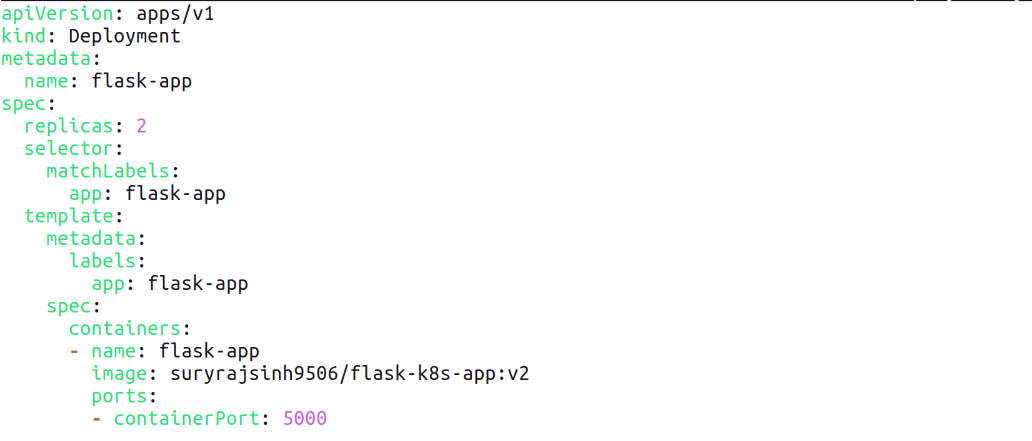


**6.2. Rebuild the Docker Image**



### **7. Update Kubernetes Deployment**

**7.1. Update the Deployment Manifest**



**7.2. Apply the Updated Manifest**

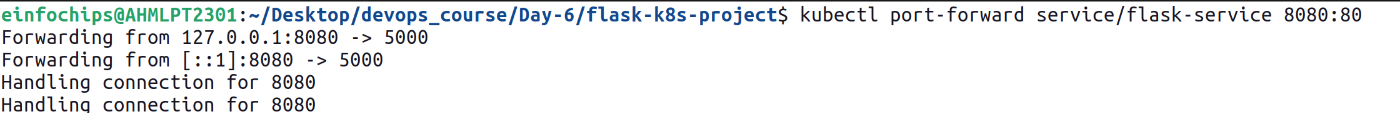


**7.3. Verify the Update**



### **8. Access the Updated Application**

**8.1. Access Through ClusterIP Service**





**8.2. Access Through NodePort Service**

