## **Project 01 - 1 Hour**

## **Deploying a Scalable Web Application with Persistent Storage and Advanced Automation**

### **Objective:**

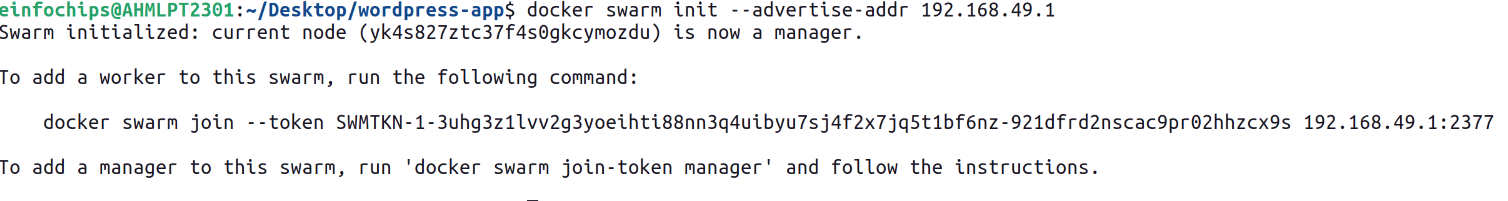
Deploy a scalable web application using Docker Swarm and Kubernetes, ensuring data persistence using a single shared volume, and automate the process using advanced shell scripting.

### **Overview:**

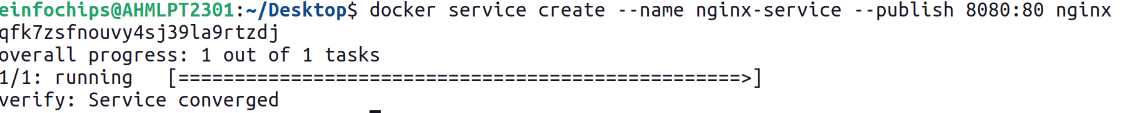
1. **Step 1**: Set up Docker Swarm and create a service.
2. **Step 2**: Set up Kubernetes using Minikube.
3. **Step 3**: Deploy a web application using Docker Compose.
4. **Step 4**: Use a single shared volume across multiple containers.
5. **Step 5**: Automate the entire process using advanced shell scripting.

### **Step 1: Set up Docker Swarm and Create a Service**

#### **1.1 Initialize Docker Swarm**

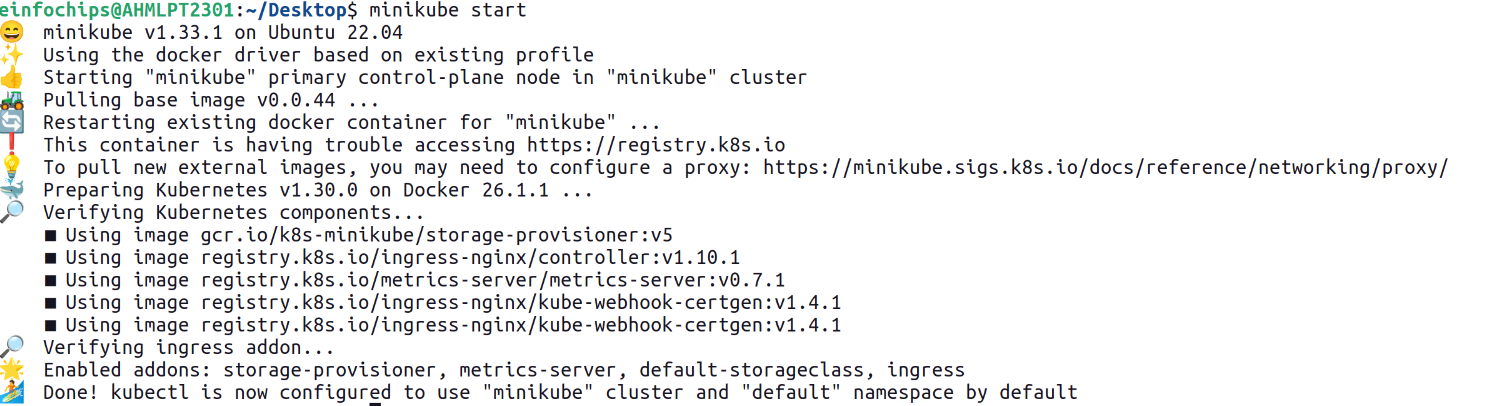
****

#### **1.2 Create a Docker Swarm Service**



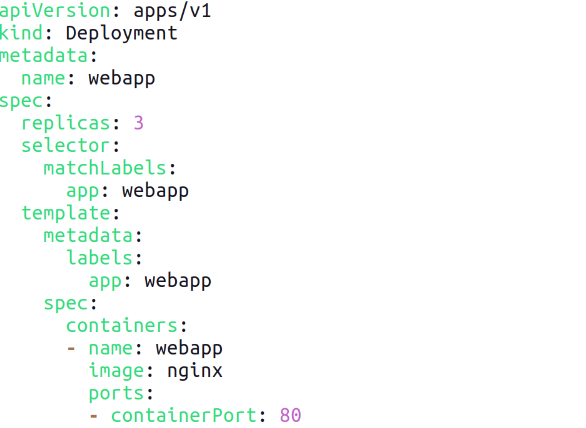
### **Step 2: Set up Kubernetes Using Minikube**

#### **2.1 Start Minikube**



#### **2.2 Deploy a Web App on Kubernetes**

Create a deployment file named webapp-deployment.yml:



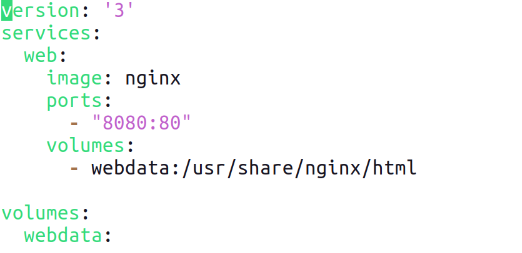
Apply the deployment:

#### **2.3 Expose the Deployment**



### **Step 3: Deploy a Web Application Using Docker Compose**

#### **3.1 Create a docker-compose.yml File**

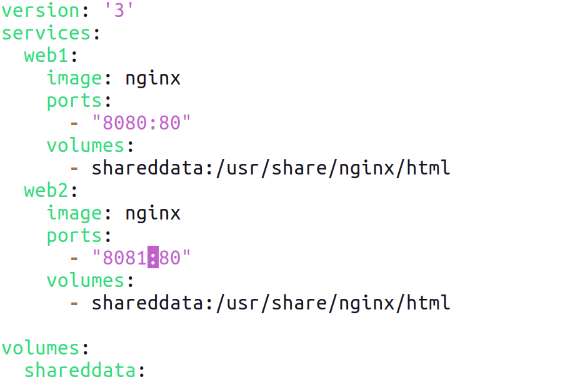


#### **3.2 Deploy the Web Application**

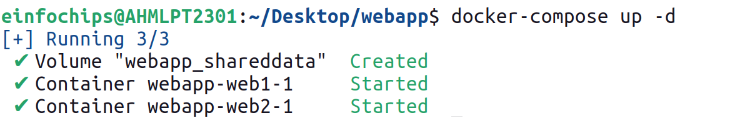


### **Step 4: Use a Single Shared Volume Across Multiple Containers**

#### **4.1 Update docker-compose.yml to Use a Shared Volume**



#### **4.2 Deploy with Docker Compose**



### **Step 5: Automate the Entire Process Using Advanced Shell Scripting**

#### **5.1 Create a Shell Script deploy.sh**

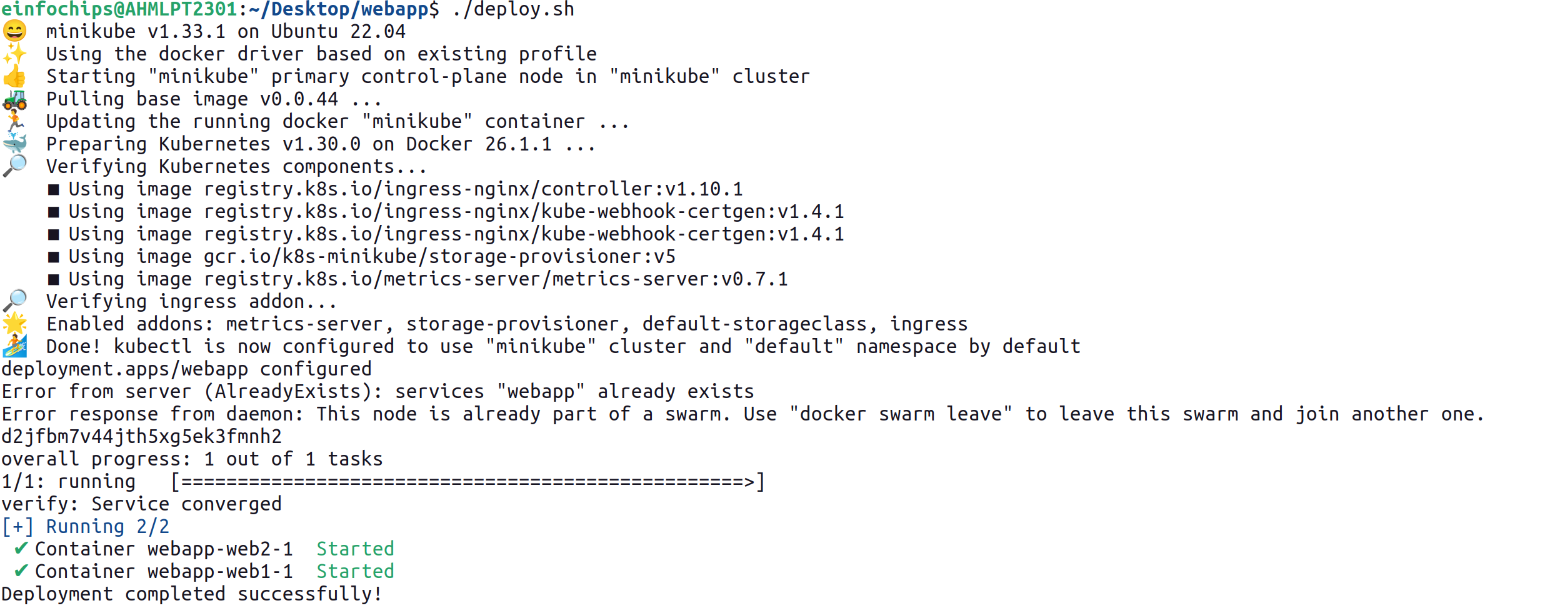
Here we change port numbers because of conflict in docker-compose.yaml file.



#### **5.2 Make the Script Executable**



#### **5.3 Run the Script**

****

### **Project 02 - 1 Hour**

### **Comprehensive Deployment of a Multi-Tier Application with CI/CD Pipeline**

### **Objective:**

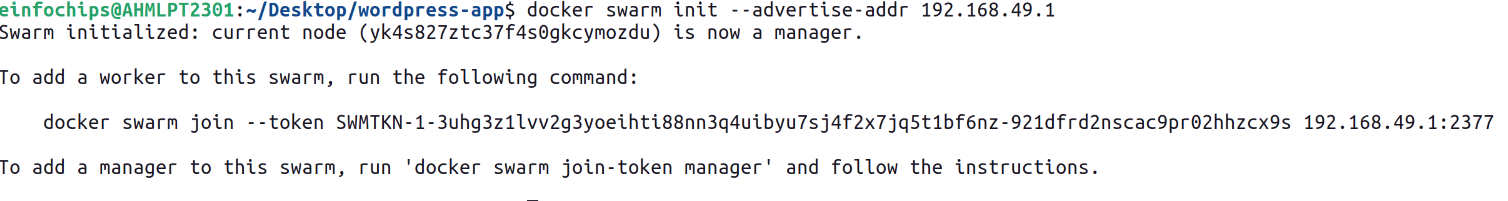
Deploy a multi-tier application (frontend, backend, and database) using Docker Swarm and Kubernetes, ensuring data persistence using a single shared volume across multiple containers, and automating the entire process using advanced shell scripting and CI/CD pipelines.

### **Overview:**

1. **Step 1**: Set up Docker Swarm and create a multi-tier service.
2. **Step 2**: Set up Kubernetes using Minikube.
3. **Step 3**: Deploy a multi-tier application using Docker Compose.
4. **Step 4**: Use a single shared volume across multiple containers.
5. **Step 5**: Automate the deployment process using advanced shell scripting.

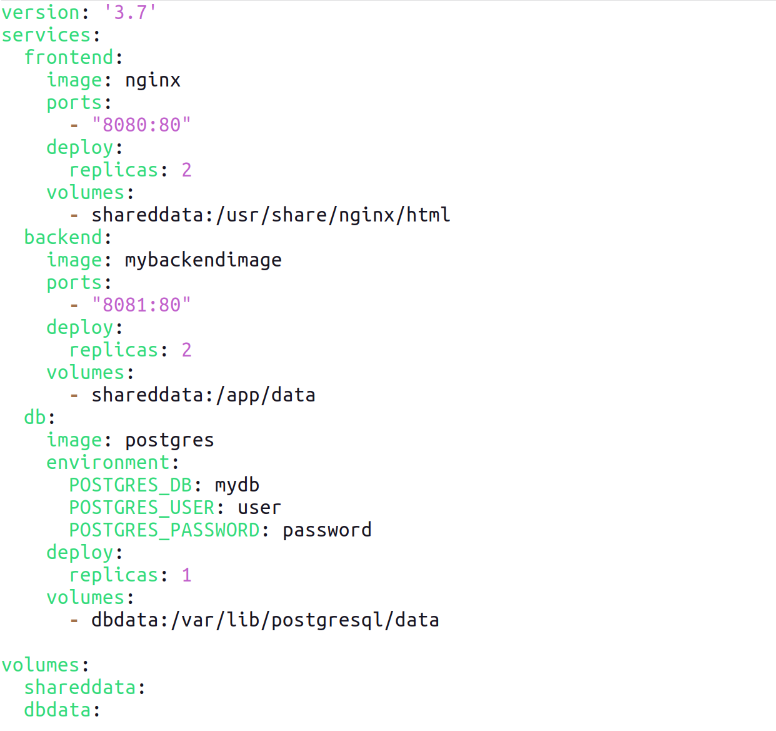
### **Step 1: Set up Docker Swarm and Create a Multi-Tier Service**

#### **1.1 Initialize Docker Swarm**

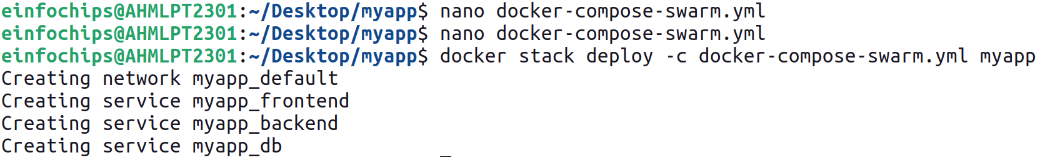


#### **1.2 Create a Multi-Tier Docker Swarm Service**

Create a docker-compose-swarm.yml file:

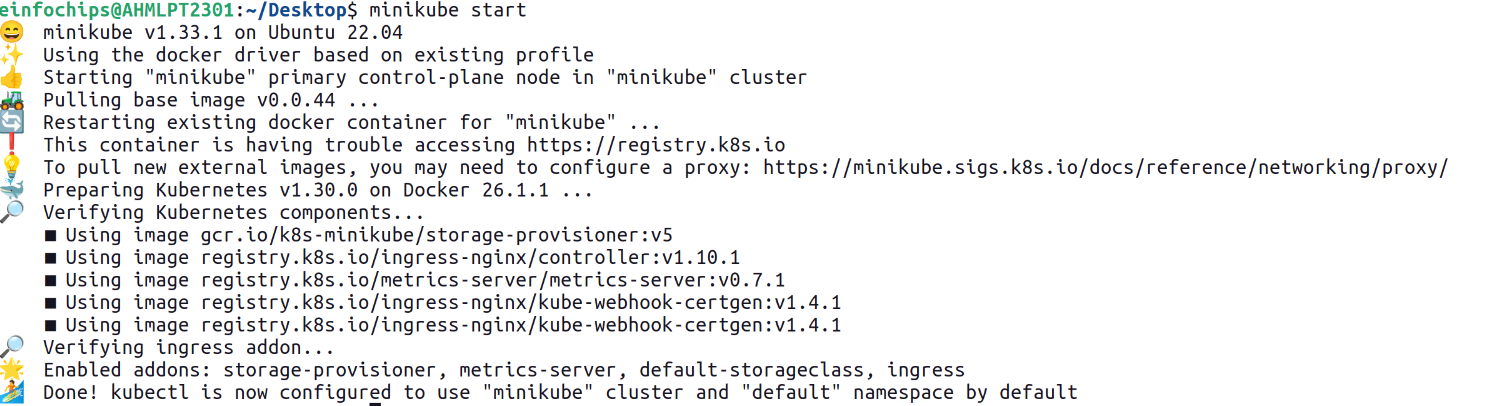


Deploy the stack:



### **Step 2: Set up Kubernetes Using Minikube**

#### **2.1 Start Minikube**

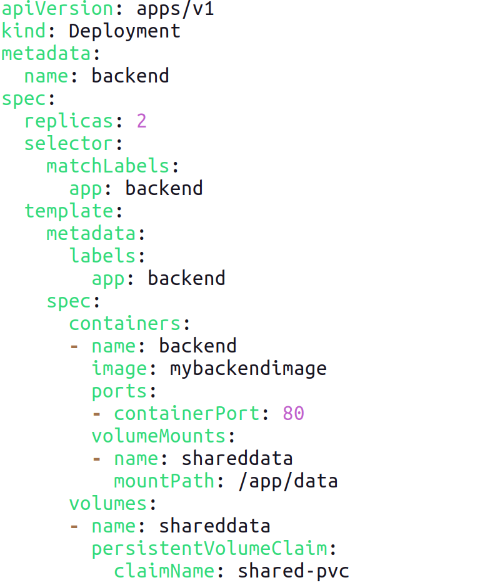


#### **2.2 Create Kubernetes Deployment Files**

Create frontend-deployment.yaml:



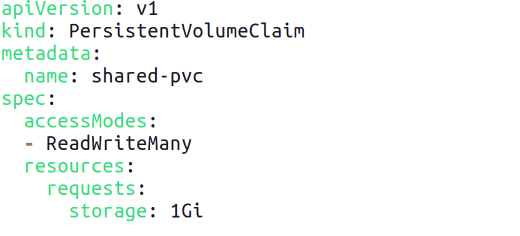
Create backend-deployment.yaml:



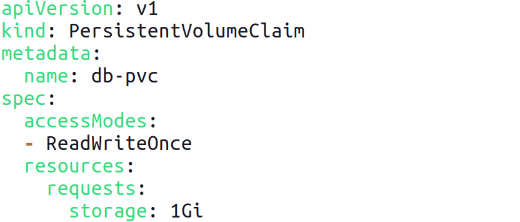
Create db-deployment.yaml:



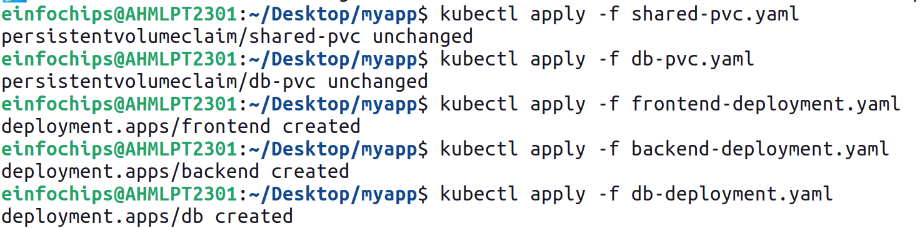
Create shared-pvc.yaml:



Create db-pvc.yaml:

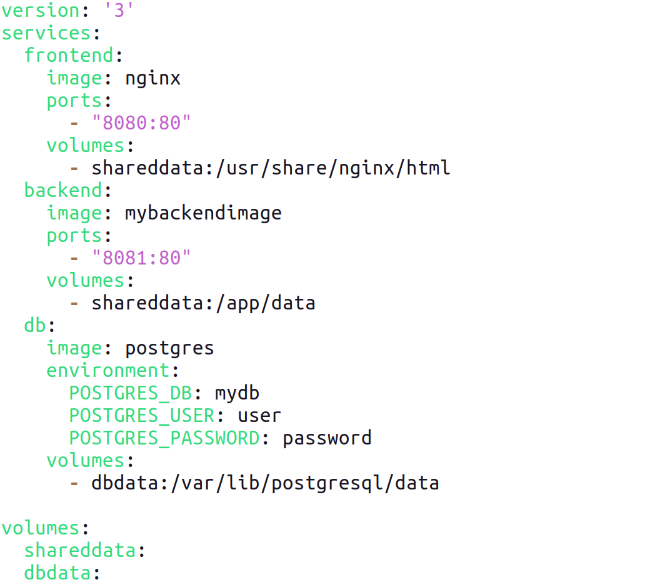


Apply the deployments:

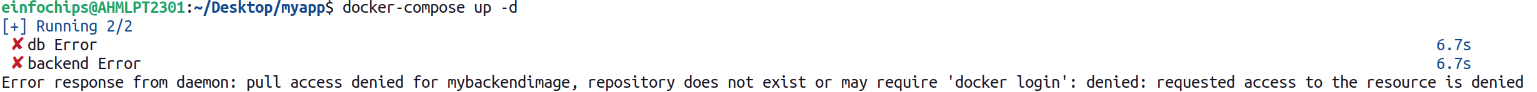


### **Step 3: Deploy a Multi-Tier Application Using Docker Compose**

#### **3.1 Create a docker-compose.yml File**



#### **3.2 Deploy the Application**



Here mybackend image is not available so it is throwing an error.