**🧪 LAB 4: Container Orchestration with Kubernetes using Minikube**

**✅ 1. Pre-requisites**

* **Windows 10/11**
* **Docker Desktop (installed and running)**
* **Minikube installed and added to system PATH**
* **kubectl (comes with Minikube)**

**⚙️ 2. Start Minikube Cluster**

**Open PowerShell as Administrator:**

**minikube start --driver=docker**

**minikube status**

**kubectl get nodes**

**You should see a minikube node in Ready state.**

**🧪 3. Create & Deploy NGINX Pod (Basic Test)**

**🔹 pod.yaml**

**apiVersion: v1**

**kind: Pod**

**metadata:**

**name: my-nginx**

**spec:**

**containers:**

**- name: nginx**

**image: nginx:latest**

**ports:**

**- containerPort: 80**

**🔹 Apply and Verify**

**kubectl apply -f pod.yaml**

**kubectl get pods**

**🐍 4. Create a Flask App**

**🔹 app.py**

**from flask import Flask**

**app = Flask(\_\_name\_\_)**

**@app.route('/')**

**def hello():**

**return "Hello from App 1!! Kubernetes is awesome!"**

**if \_\_name\_\_ == '\_\_main\_\_':**

**app.run(host='0.0.0.0', port=5000)**

**🔹 requirements.txt**

**flask==3.0.0**

**📦 5. Create Docker Image**

**🔹 Dockerfile**

**FROM python:3.12-slim**

**WORKDIR /app**

**COPY requirements.txt .**

**RUN pip install --no-cache-dir -r requirements.txt**

**COPY app.py .**

**EXPOSE 5000**

**CMD ["python", "app.py"]**

**🐳 6. Build & Push Docker Image to Docker Hub**

**Replace sarvesh8393 with your own Docker Hub username.**

**docker build -t sarvesh8393/app1-k8s:latest .**

**docker login**

**docker push sarvesh8393/app1-k8s:latest**

**☸️ 7. Create Kubernetes Deployment**

**🔹 deployment.yaml**

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: hw-deployment**

**spec:**

**replicas: 2**

**selector:**

**matchLabels:**

**app: hello-world**

**template:**

**metadata:**

**labels:**

**app: hello-world**

**spec:**

**containers:**

**- name: hw-container**

**image: sarvesh8393/app1-k8s:latest**

**ports:**

**- containerPort: 5000**

**🔹 Apply Deployment**

**kubectl apply -f deployment.yaml**

**🌐 8. Create Kubernetes Service**

**🔹 service.yaml**

**apiVersion: v1**

**kind: Service**

**metadata:**

**name: hello-world**

**spec:**

**type: NodePort**

**selector:**

**app: hello-world**

**ports:**

**- port: 5000**

**targetPort: 5000**

**🔹 Apply Service**

**kubectl apply -f service.yaml**

**🔍 9. Verify Setup**

**kubectl get pods**

**kubectl get svc**

**You should see 2 running pods and a service hello-world with a NodePort assigned.**

**🌐 10. Access Your Flask App**

**🔹 Forward port to localhost**

**kubectl port-forward svc/hello-world 5000:5000**

**🔹 Visit in Browser**

**Open:** [**http://localhost:5000**](http://localhost:5000/) **You should see:**

**Hello from App 1!! Kubernetes is awesome!**

**⚙️ 11. Scale the Deployment**

**kubectl scale deployment hw-deployment --replicas=3**

**kubectl get deployment**

**kubectl get pods**

**🧹 12. Cleanup Commands (Optional)**

**kubectl delete all --all # Delete all resources in the cluster**

**kubectl delete deployment hw-deployment**

**kubectl delete svc hello-world**

**minikube stop**

**minikube delete**

**🧠 Summary Table**

| **Task** | **Command** |
| --- | --- |
| **Start cluster** | **minikube start --driver=docker** |
| **Build image** | **docker build -t sarvesh8393/app1-k8s:latest .** |
| **Push image** | **docker push sarvesh8393/app1-k8s:latest** |
| **Apply deployment** | **kubectl apply -f deployment.yaml** |
| **Apply service** | **kubectl apply -f service.yaml** |
| **Check pods** | **kubectl get pods** |
| **Check services** | **kubectl get svc** |
| **Port forward** | **kubectl port-forward svc/hello-world 5000:5000** |
| **Scale up** | **kubectl scale deployment hw-deployment --replicas=3** |
| **Cleanup** | **kubectl delete all --all** |

**Let me know if you want this as a PDF or Word doc for printing/studying!**