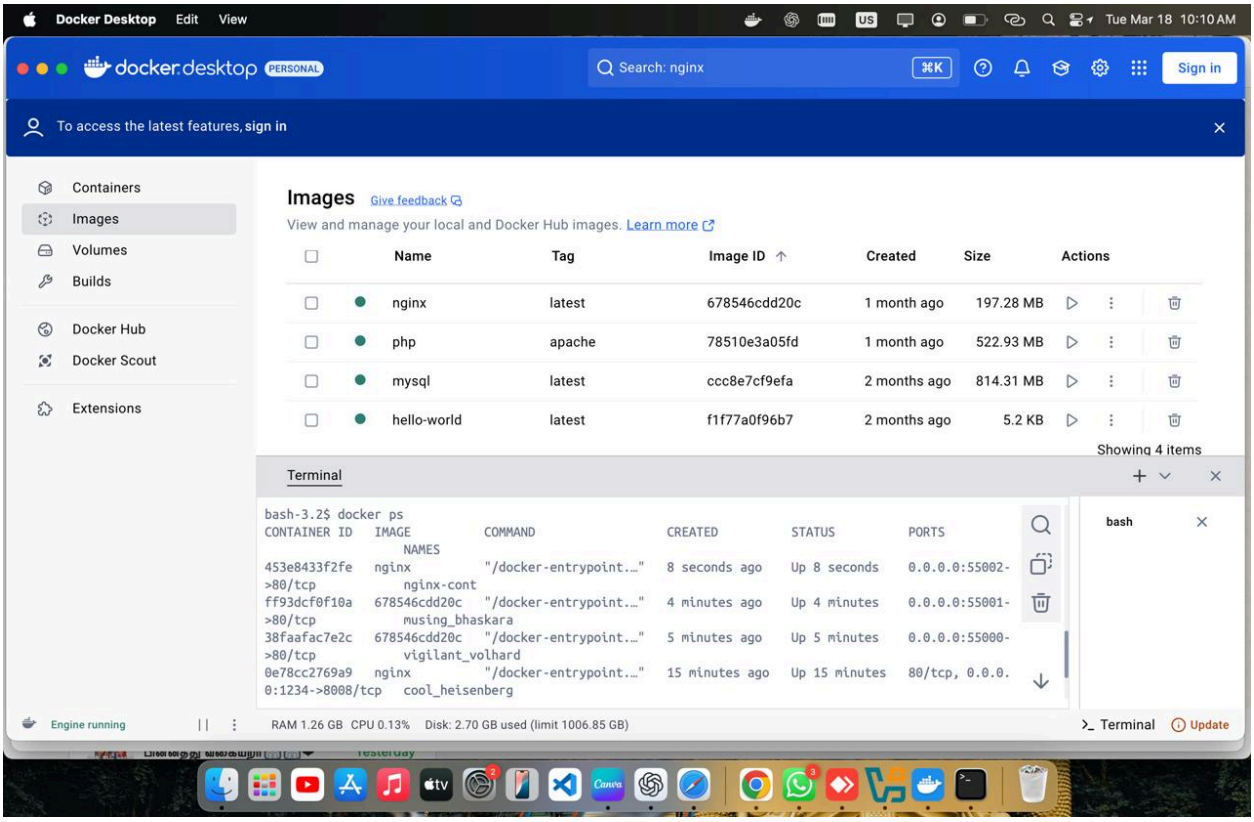
DevOps Class- Guvi

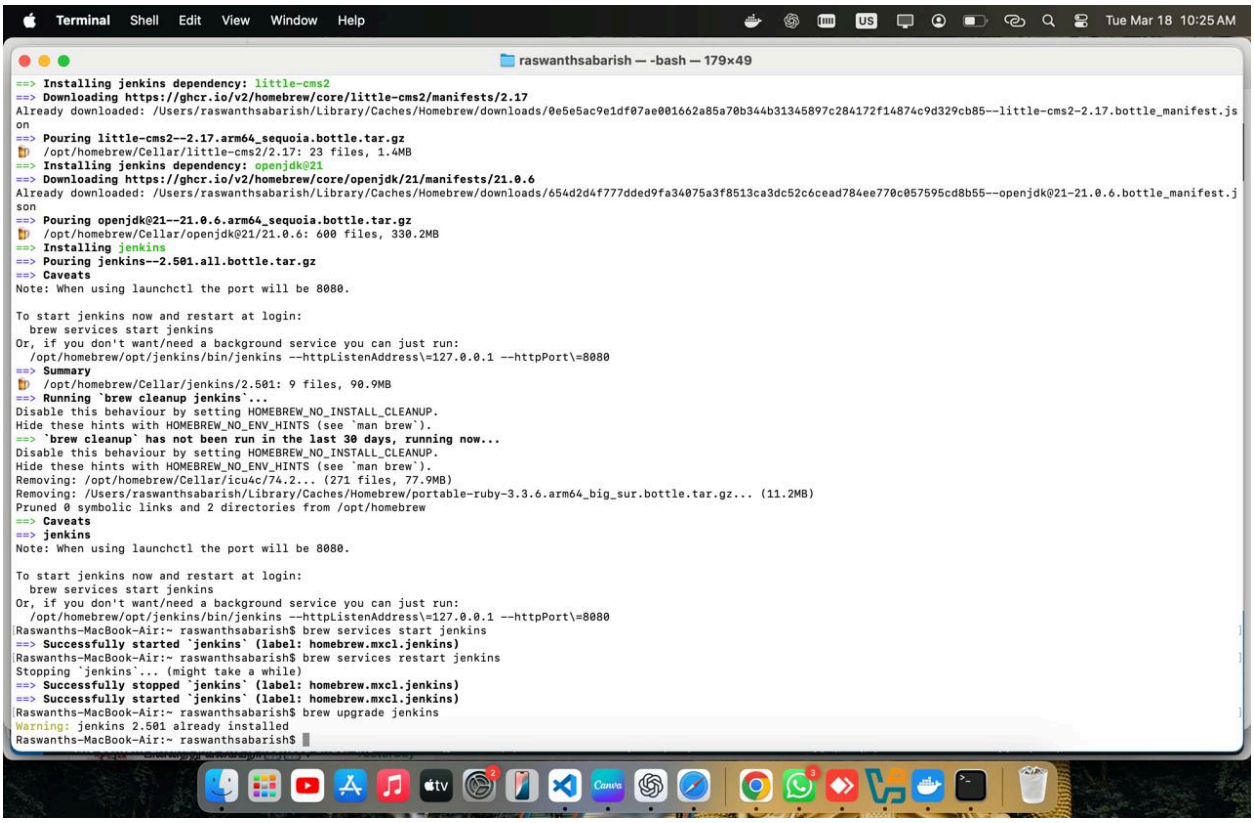
Github link: https://github.com/RASWANTHSABARISH/devops_class_guvi.git

17 March 2025 (DAY-1)



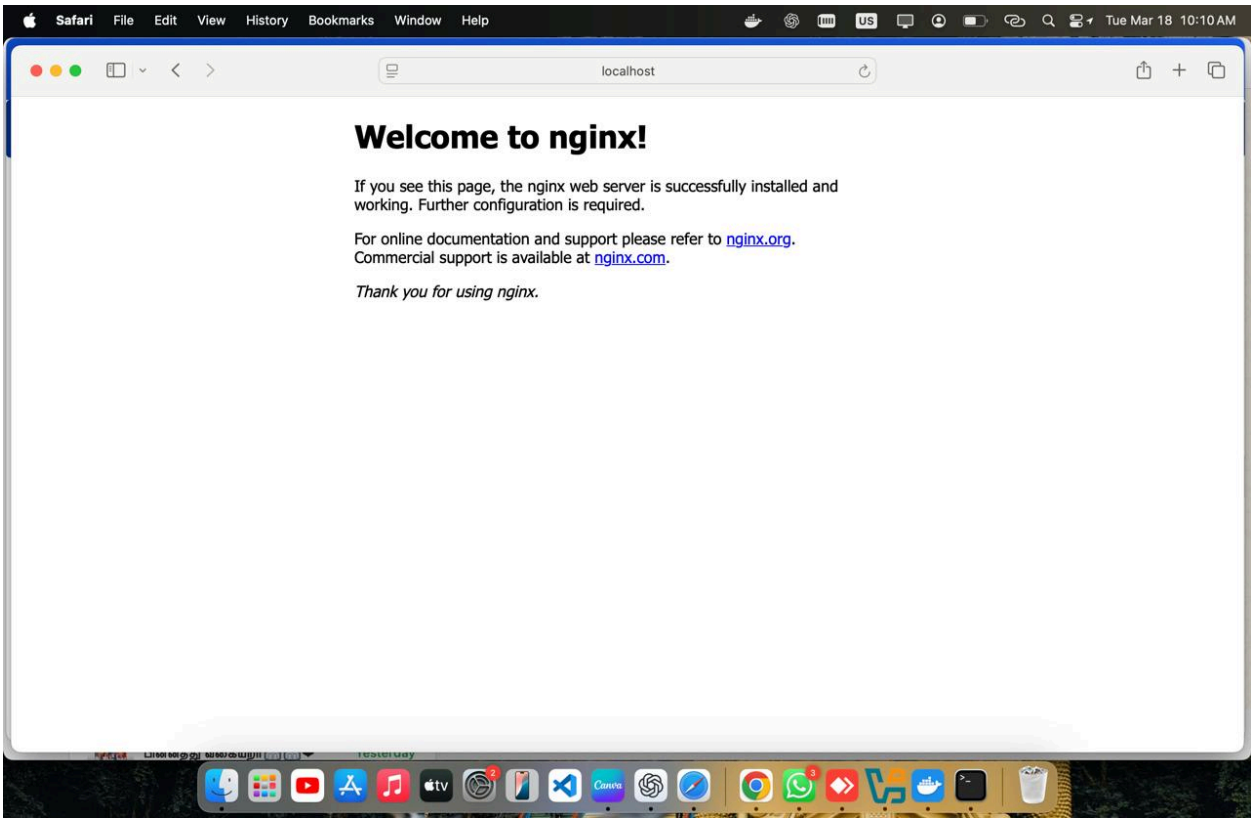
Displays various downloaded Docker image **nginx**

Running Containers – The docker ps command reveals multiple **nginx containers** running with different names, ports, and uptime statuses.



Jenkins Installation via Homebrew – Jenkins being installed using Homebrew, with dependencies like OpenJDK

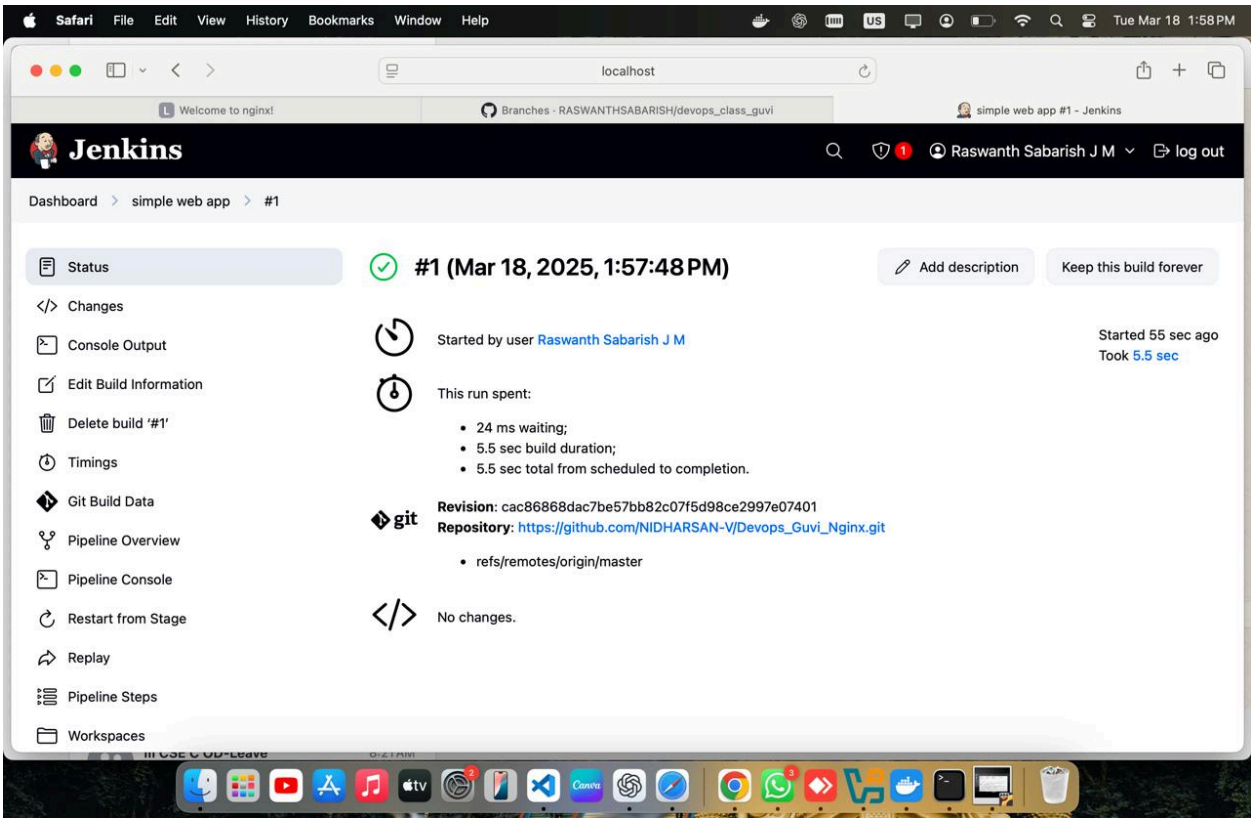
start jenkins and brew services restart jenkins are used to manage the Jenkins background service, running on port **8080**.



Default **Nginx welcome page**, confirming that the containerized Nginx server is running properly.

The Nginx server is accessible at **localhost**, meaning it's correctly mapped from the container's port to the host system.

18 March 2025 (DAY-2)

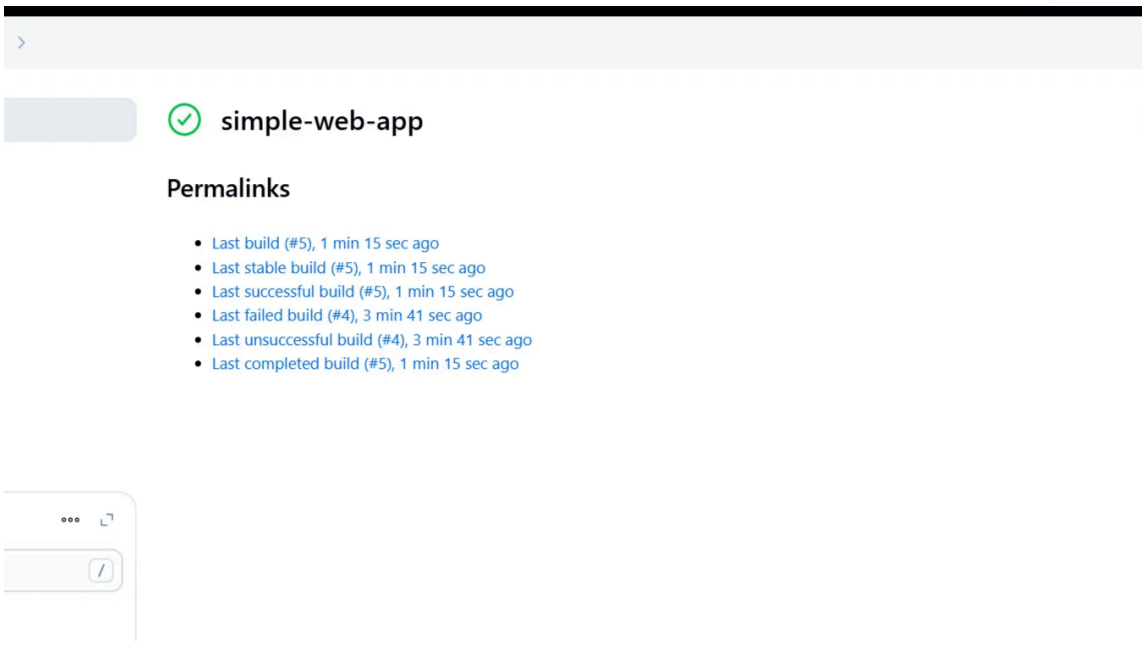


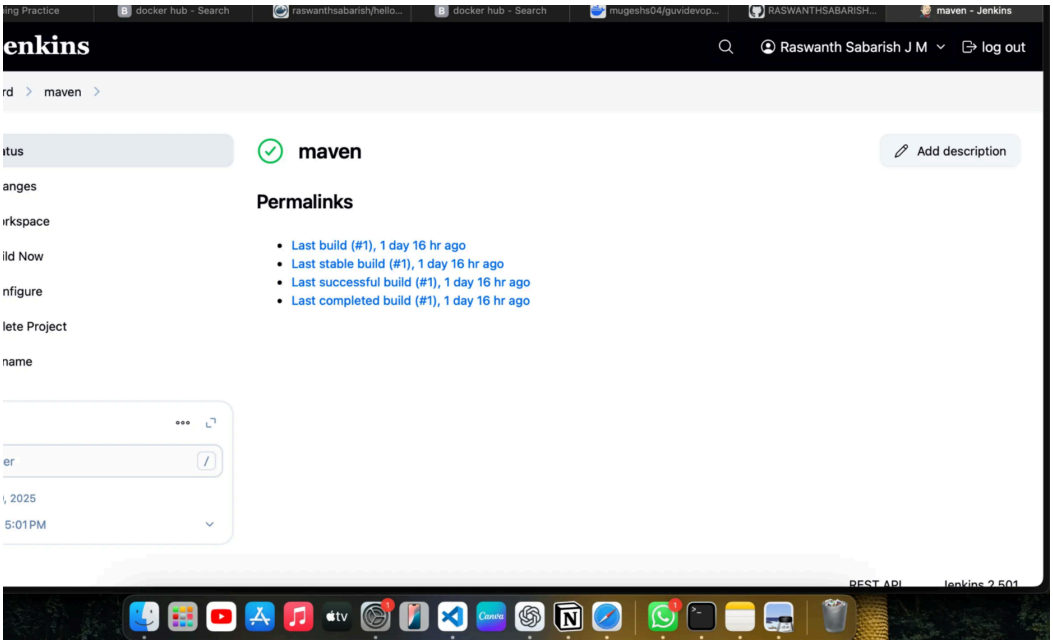
simple-web-app > #5

```
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Run Docker Container)
[Pipeline] script
[Pipeline] {
[Pipeline] bat

C:\ProgramData\Jenkins\.jenkins\workspace\simple-web-app>docker run -d -p 8888:8080 --name war-container warimage-jenkins
1aa7b7e48602067595e088b0ee5f3404050d735b925d69fa604a2686b37b95c6
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

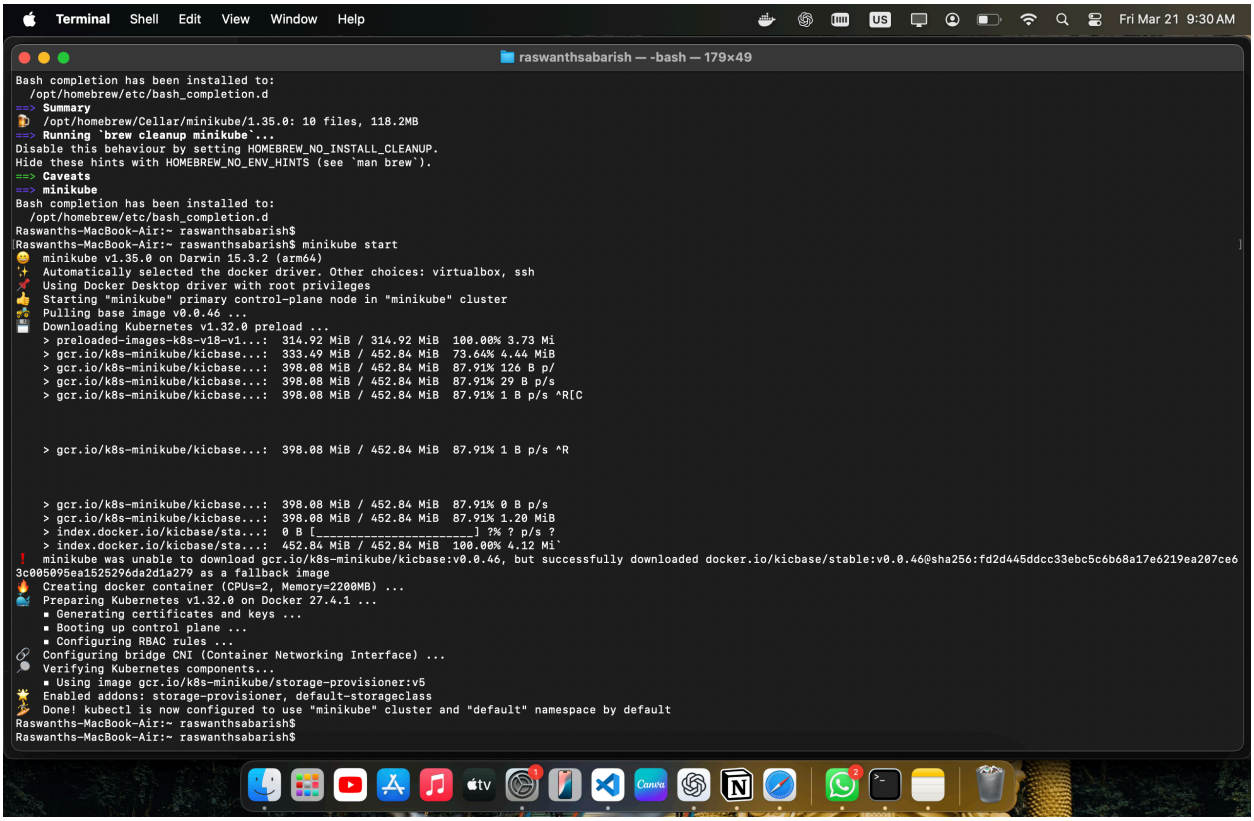
REST API Jenkins 2.501





19 March 2025 (DAY-3)

I**nstall Minikube**- on macOS using Homebrew:



wsl tool. (only for windows)

Installing java- sudo apt install fontconfig openjdk-17-jre java -version

Installing Jenkins on Ubuntu/Debian

- Follow the official Jenkins installation guid Jenkins Installation Guide
- Restart and check Jenkins service status

  sudo service jenkins restart

  sudo service jenkins status

Installing Docker

sudo apt install docker.io -y

sudo service docker restart

sudo service docker status

1. Add user to the Docker group
    a. sudo usermod -aG docker $USER
2. Check Docker images and running containers.
    a. sudo chmod 666 /var/run/docker.sock

Installing Kubernetes (kubectl)

Download and install kubectl.

curl -LO https://dl.k8s.io/release/v1.32.0/bin/linux/amd64/kubectl

sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

chmod +x kubectl

mkdir -p ~/.local/bin

mv ./kubectl ~/.local/bin/kubectl

kubectl version --client

Installing Minikube (Kubernetes)

Download and install Minikube

curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64

Start Minikube and check status.

minikube start

```
minikube status
```

Check Kubernetes resources.

```
kubectl get pod
kubectl get deploy
kubectl get replica
kubectl get pod -o wide
```

Docker Compose (Managing Multi-Container Applications)

Install Docker Compose.

```
sudo apt install docker-compose -y
```

Download the latest Docker Compose binary.

```
sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

Example docker-compose.yml file for running **NGINX and MySQL**.

yaml code:

```
version: '3'

services:
web:
image: nginx:latest
ports:
- 80:80

db:
image: mysql:latest
environment:
- MYSQL_ROOT_PASSWORD=secret
```

Running MySQL Inside Docker Container

Enter the MySQL container shell.

```
docker exec -it david-db-1 /bin/bash
```

Login to MySQL

```
mysql -u root -p
```

Jenkins Workspace and Maven Build Location

Path where Jenkins builds and stores the .war file.

```
/var/lib/jenkins/workspace/maven/target/my-app.war
```

Pipelining code for Tomcat

```
pipeline {
agent any
environment {
   DOCKER_CREDENTIALS = credentials('docker-hub-cred')  // Docker Hub
Credentials ID
}
stages {
   stage('SCM') {
      steps {
         git branch: 'main', url: '<https://github.com/MugeshS-
04/guvidevopsday1.git>'
      }
   }
   stage('Build') {
      steps {
         sh "mvn clean"
         sh "mvn install"
      }
   }
   stage('Build Docker Image') {
      steps {
         script {
            sh 'docker build -t mugeshs04/guvidevopsday1 .'
         }
```

```
        }
    }
    stage('Push to Docker Hub') {
        steps {
            script {
                docker.withRegistry('<https://index.docker.io/v1/>', 'docker-hub-
cred') {
                    sh 'docker push mugeshs04/guvidevopsday1'
                }
            }
        }
    }
}}
```

20 March 2025 (DAY-4)

Kubernetes (K8s) Notes

Kubernetes is an open-source container orchestration platform for automating deployment, scaling, and management of containerized applications.

Originally developed by **Google** as **Borg** in the early 2000s and open-sourced in 2014 under CNCF.

Architecture: Kubernetes follows a **master-worker** model.

Control Plane (Master Node)

API Server – Entry point for all Kubernetes interactions.

•**Scheduler** – Assigns workloads to nodes based on resource availability.

Controller Manager – Ensures the cluster's desired state (e.g., scaling, replication).

etcd – Stores all cluster data persistently.

Worker Nodes (Slave Nodes)

Kubelet – Manages pods and communicates with the master node.

Kube Proxy – Handles networking and load balancing.

Container Runtime – Runs containers (e.g., Docker, containerd).

Key Kubernetes Concepts

Pods – Smallest deployable unit in Kubernetes, containing one or more containers.

- **Deployments** – Manage and scale pod replicas.

Services – Expose applications within and outside the cluster.

- **Ingress** – Manages external access to services using HTTP/HTTPS.