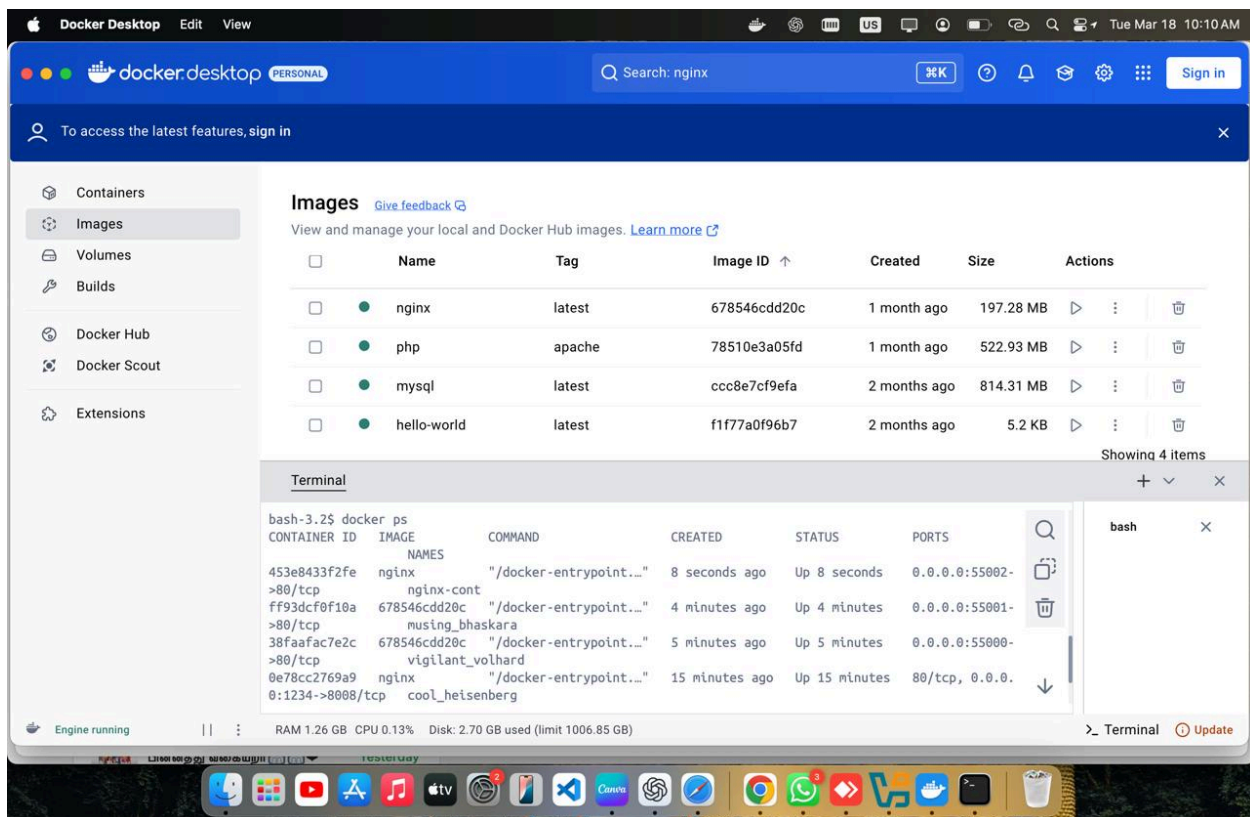


DevOps Class- Guvi

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.

```
Terminal Shell Edit View Window Help
raswanthsabarish - -bash - 179x49

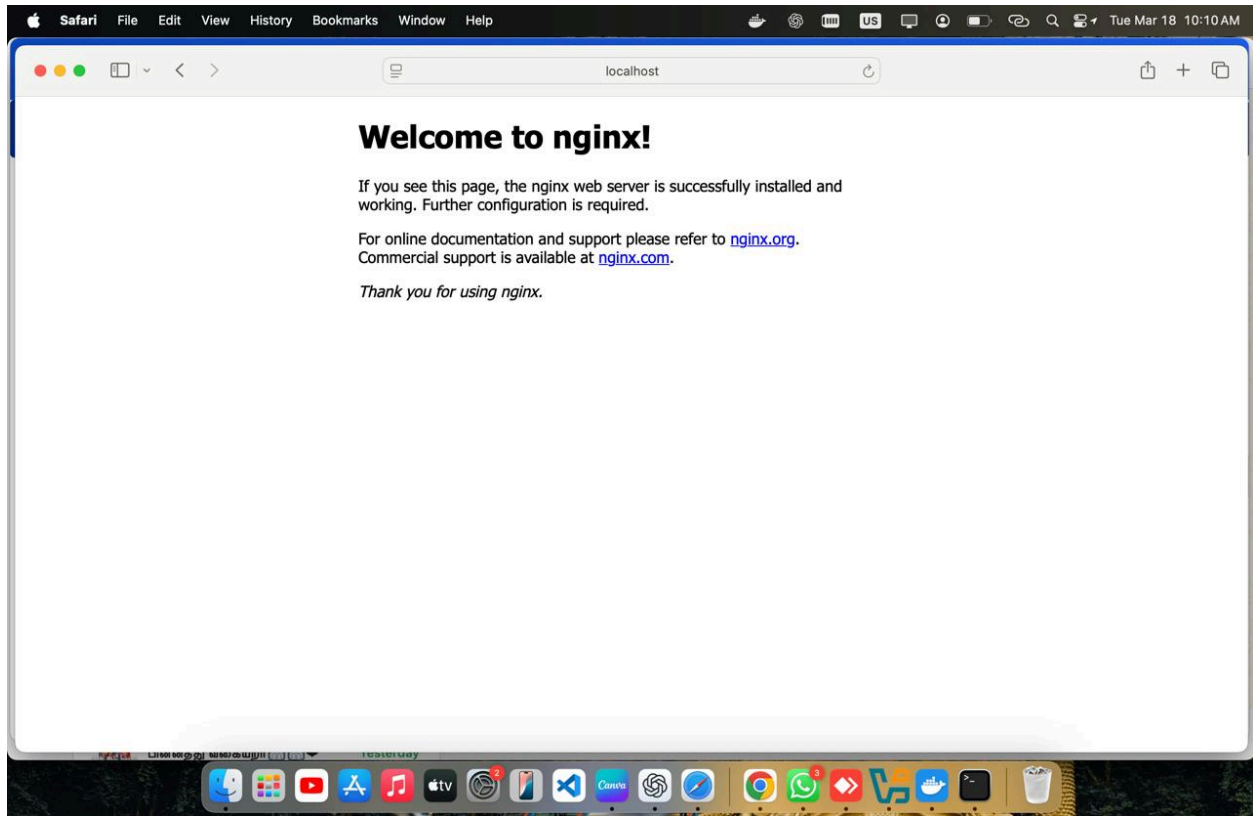
==> Installing jenkins dependency: little-cms2
==> Downloading https://ghcr.io/v2/homebrew/core/little-cms2/manifests/2.17
Already downloaded: /Users/raswanthsabarish/Library/Caches/Homebrew/downloads/0e5e5ac9e1df07ae001662a85a78b344b31345897c284172f14874c9d329cb85--little-cms2-2.17.bottle_manifest.js
on
==> Pouring little-cms2--2.17.arm64_sequoia.bottle.tar.gz
  /opt/homebrew/Cellar/little-cms2/2.17: 23 files, 1.4MB
==> Installing jenkins dependency: openjdk21
==> Downloading https://ghcr.io/v2/homebrew/core/openjdk/21/manifests/21.0.6
Already downloaded: /Users/raswanthsabarish/Library/Caches/Homebrew/downloads/654d2d4f777dded9fa34075a3f8513ca3dc52c6cead784ee770c057595cd8b55--openjdk@21-21.0.6.bottle_manifest.j
son
==> Pouring openjdk@21--21.0.6.arm64_sequoia.bottle.tar.gz
  /opt/homebrew/Cellar/openjdk@21/21.0.6: 600 files, 330.2MB
==> Installing jenkins
==> Pouring jenkins--2.501.all.bottle.tar.gz
==> Caveats
Note: When using launchctl the port will be 8080.

To start jenkins now and restart at login:
  brew services start jenkins
Or, if you don't want/need a background service you can just run:
  /opt/homebrew/opt/jenkins/bin/jenkins --httpListenAddress=127.0.0.1 --httpPort=8080
==> Summary
  /opt/homebrew/Cellar/jenkins/2.501: 9 files, 90.9MB
==> Running 'brew cleanup jenkins'...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see 'man brew').
==> 'brew cleanup' has not been run in the last 30 days, running now...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see 'man brew').
Removing: /opt/homebrew/Cellar/icu4c/74.2... (271 files, 77.9MB)
Removing: /Users/raswanthsabarish/Library/Caches/Homebrew/portable-ruby-3.3.6.arm64_big_sur.bottle.tar.gz... (11.2MB)
Pruned 0 symbolic links and 2 directories from /opt/homebrew
==> Caveats
jenkins
Note: When using launchctl the port will be 8080.

To start jenkins now and restart at login:
  brew services start jenkins
Or, if you don't want/need a background service you can just run:
  /opt/homebrew/opt/jenkins/bin/jenkins --httpListenAddress=127.0.0.1 --httpPort=8080
Raswanths-MacBook-Air:~ raswanthsabarish$ brew services start jenkins
==> Successfully started 'jenkins' (label: homebrew.mxcl.jenkins)
Raswanths-MacBook-Air:~ raswanthsabarish$ brew services restart jenkins
Stopping 'jenkins'... (might take a while)
==> Successfully stopped 'jenkins' (label: homebrew.mxcl.jenkins)
==> Successfully started 'jenkins' (label: homebrew.mxcl.jenkins)
Raswanths-MacBook-Air:~ raswanthsabarish$ brew upgrade jenkins
Warning: jenkins 2.501 already installed
Raswanths-MacBook-Air:~ raswanthsabarish$
```

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

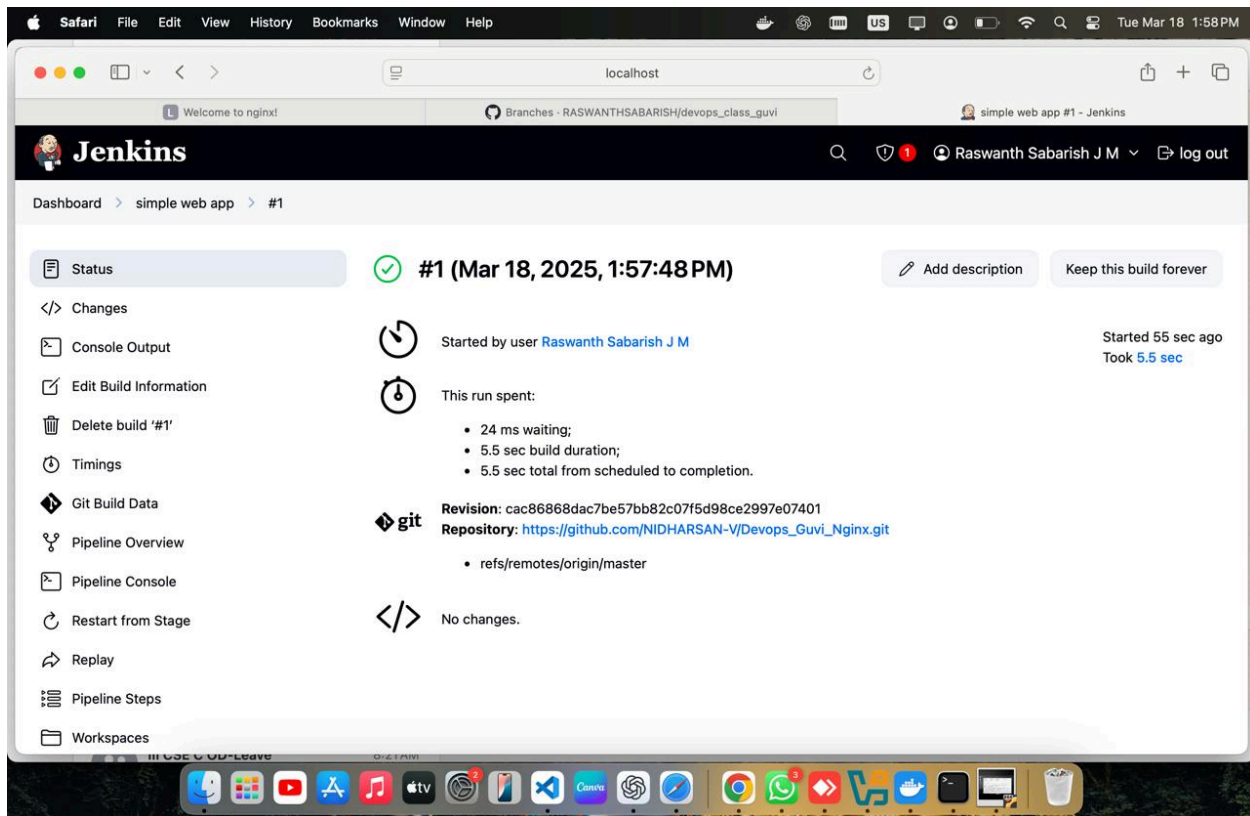
start jenkins and brew services restart jenkins are used to manage the Jenkins background service, running on port **8080**.



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)



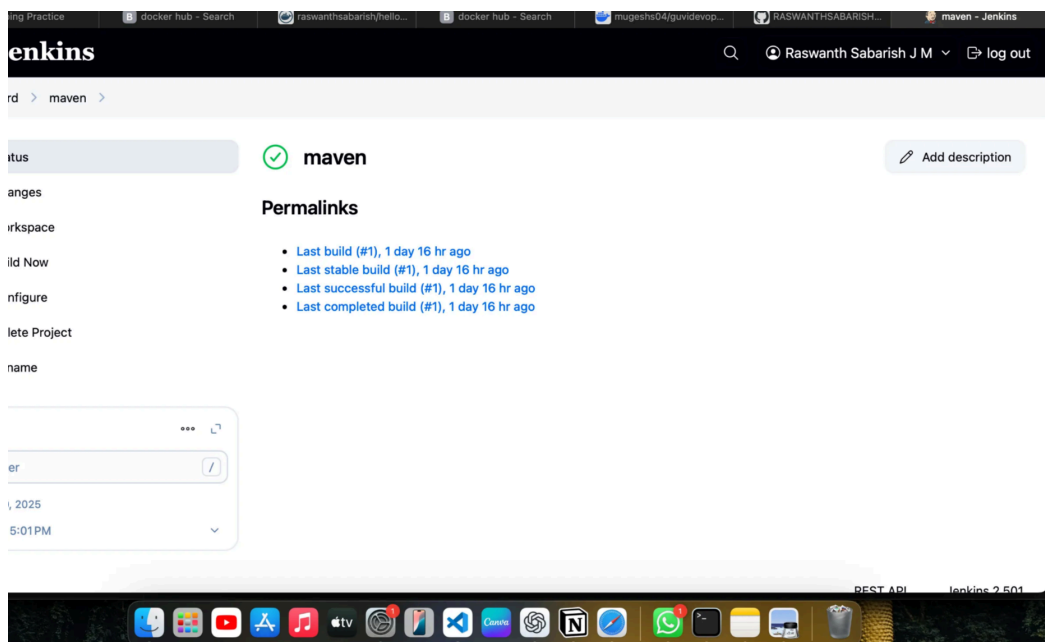
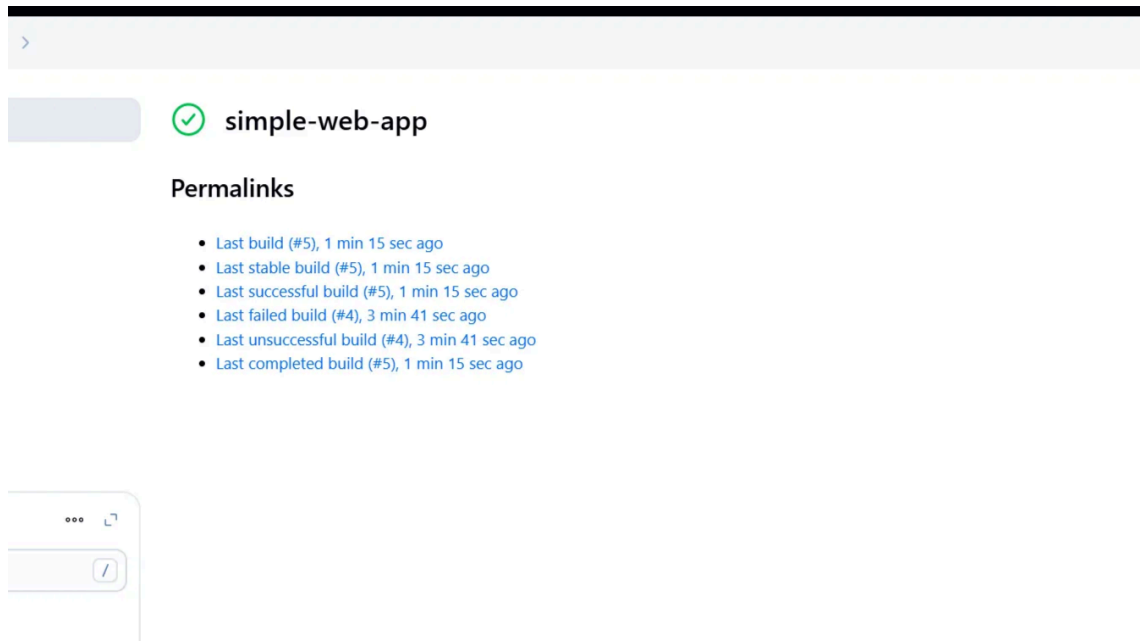
simple-web-app > #5

```
[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)

Install Minikube- on macOS using Homebrew:

```
Terminal Shell Edit View Window Help
raswanthsabarish -- -bash -- 179x49

Bash completion has been installed to:
/opt/homebrew/etc/bash_completion.d
=> Summary
  /opt/homebrew/Cellar/minikube/1.35.0: 10 files, 118.2MB
=> Running 'brew cleanup minikube' ...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see 'man brew').
=> Caveats
=> minikube
Bash completion has been installed to:
/opt/homebrew/etc/bash_completion.d
Raswanths-MacBook-Air:~ raswanthsabarish$
Raswanths-MacBook-Air:~ raswanthsabarish$ minikube start
  minikube v1.35.0 on Darwin 15.3.2 (arm64)
  Automatically selected the docker driver. Other choices: virtualbox, ssh
  Using Docker Desktop driver with root privileges
  Starting "minikube" primary control-plane node in "minikube" cluster
  Pulling base image v0.0.46 ...
  Downloading Kubernetes v1.32.0 preload ...
  > gcr.io/k8s-minikube/kicbase...: 314.92 MiB / 314.92 MiB 100.00% 3.73 MiB/s
  > gcr.io/k8s-minikube/kicbase...: 333.49 MiB / 452.84 MiB 73.64% 4.44 MiB/s
  > gcr.io/k8s-minikube/kicbase...: 398.08 MiB / 452.84 MiB 87.91% 126 B p/s
  > gcr.io/k8s-minikube/kicbase...: 398.08 MiB / 452.84 MiB 87.91% 29 B p/s
  > gcr.io/k8s-minikube/kicbase...: 398.08 MiB / 452.84 MiB 87.91% 1 B p/s ^R[C

  > gcr.io/k8s-minikube/kicbase...: 398.08 MiB / 452.84 MiB 87.91% 1 B p/s ^R

  > gcr.io/k8s-minikube/kicbase...: 398.08 MiB / 452.84 MiB 87.91% 0 B p/s
  > gcr.io/k8s-minikube/kicbase...: 398.08 MiB / 452.84 MiB 87.91% 1.20 MiB/s
  > index.docker.io/kicbase/sta...: 0 B [ ] 7% ? p/s ?
  > index.docker.io/kicbase/sta...: 452.84 MiB / 452.84 MiB 100.00% 4.12 MiB/s
! minikube was unable to download gcr.io/k8s-minikube/kicbase:v0.0.46, but successfully downloaded docker.io/kicbase/stable:v0.0.46@sha256:fd2d445ddcc33ebc5c6b68a17e6219ea207ce63c005095ea1525296da2d1a2779 as a fallback image
  Creating docker container (CPUs=2, Memory=2200MB) ...
  Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
  ■ Generating certificates and keys ...
  ■ Booting up control plane ...
  ■ Configuring RBAC rules ...
  Configuring bridge CNI (Container Networking Interface) ...
  Verifying Kubernetes components...
  ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
  Enabled addons: storage-provisioner, default-storageclass
  Done! kubect1 is now configured to use "minikube" cluster and "default" namespace by default
Raswanths-MacBook-Air:~ raswanthsabarish$
Raswanths-MacBook-Air:~ raswanthsabarish$
```

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.