DEVOPS LAB EXAM – NEW SYSTEM CHECKLIST (Q1)

⏱️ First 5 Minutes: Basic Setup (MOST IMPORTANT)

Step 1: Check Internet

```
ping google.com
```

Step 2: Install Java (JDK 21)

```
sudo apt update
sudo apt install openjdk-21-jdk -y
java -version
```

Step 3: Install Git

```
sudo apt install git -y
git --version
```

Step 4: Install Maven

```
sudo apt install maven -y
mvn -version
```

Step 5: Install Docker

```
sudo apt install docker.io -y
sudo systemctl start docker
sudo systemctl enable docker
docker --version
```

Add user to Docker group:

```
sudo usermod -aG docker $USER
newgrp docker
```

Step 6: Install Jenkins

```
sudo apt install jenkins -y
sudo systemctl start jenkins
sudo systemctl enable jenkins
```

Open browser:

```
http://localhost:8080
```

⏱ Next 5 Minutes: Jenkins Setup

Step 7: Install Jenkins Plugins

Jenkins → Manage Jenkins → Plugins:

- Git
- Pipeline
- Docker Pipeline
- Credentials Binding

Restart Jenkins.

Step 8: Add Docker Hub Credentials

Jenkins → Credentials → Global:

- Kind: Username & Password
- ID: dockerhub

Step 9: Jenkins ↔ Docker Permission

```
sudo usermod -aG docker jenkins
sudo systemctl restart docker
sudo systemctl restart jenkins
```

✅ After Question 1 – What to Show the Invigilator

Think of this as your **“proof list”**.

1️⃣ Jenkins Pipeline – SUCCESS

Open Jenkins → your job → latest build.

Show:

- All stages **green ✔**

- Console Output ends with:

```
Pipeline Successful
```

👉 Say:

> This shows the CI/CD pipeline executed successfully.

2️⃣ Docker Hub – Image Pushed

Open Docker Hub in browser.

Show:

- Repository: `my_maven_app`
- Tag: `latest`
- Updated time = **just now**

👉 Say:

> The Docker image was pushed to Docker Hub by Jenkins.

3️⃣ Docker Swarm – Service Running

In terminal, run:

```
docker service ls
```

Show:

- Service name
- Replicas: `1/1`

Then:

```
docker service ps my_maven_service
```

Status:

```
Running
```

👉 Say:

> The application is deployed as a Docker Swarm service.

4️⃣ Browser Output (MOST IMPORTANT)

Open browser:

```
http://localhost:8080
```

Show:

```
Hello from Spring Boot Maven App!
```

☞ Say:

> This confirms successful deployment and port mapping.

3. Maven + Java + Kubernetes (Manual / On-Demand Pipeline)

Create a CI/CD pipeline for a Maven-based Java application using Git/GitHub, Jenkins, and Docker. Deploy the containerized application on a Kubernetes cluster and demonstrate the execution of each pipeline stage.

---

4. Maven + Java + Kubernetes (Cron-based Automated Pipeline)

Build a Java application using Maven and configure a fully automated CI/CD pipeline with Git/GitHub, Jenkins, and Docker. Use Jenkins cron jobs to trigger the pipeline automatically and deploy the application on Kubernetes. Explain all automated stages of the pipeline.