CLASS ACTIVITY: “From Code to Cluster”

Docker + Kubernetes Integrated Challenge

Goal of the Activity

Students will:

- Build containers from scratch
- Use Docker Compose
- Deploy on Kubernetes
- Run CI/CD using Jenkins
- Debug failures like software engineers

Theme:

You are hired by a startup, DevSolutions Ltd.

Their web app keeps crashing.

Your job as a DevOps Engineer is to:

- Containerize the app
- Run multi-service architecture
- Deploy it to Kubernetes
- Automate updates using Jenkins

TOTAL DURATION: 2 Hours

PHASE 1: Container Bootcamp

Your mission:

Build and run containers manually (no compose, no k8s yet).

Tasks:

1. Pull NGINX image
2. Run it on custom port
3. Stop it, remove it
4. Build your own Dockerfile

5. Run custom container

Challenge:

- Change HTML content inside container
- Rebuild image with version tag v2

OUTPUT REQUIRED:

docker ps

docker images

- Working web app screenshot

PHASE 2: App to Containers (LAB 02)

40 Minutes

Your mission:

Run app in Docker → then Kubernetes.

Tasks:

1. Dockerize simple web app
2. Push image (optional)
3. Create Kubernetes:
    - Deployment
    - Service
4. Verify Pod running
5. Access using NodePort / Port Forwarding

OUTPUT REQUIRED:

- YAML file
- Pod screenshot
- App accessible proof

PHASE 3: Multi-Service App (LAB 03)

Your mission:

Run frontend + backend + database together.

Tasks:

1. Build docker-compose.yml
2. Run:

docker-compose up

3. Stop, remove, and rebuild
4. Change DB name or port
5. Check logs

Challenges:

- Add volume
- Restart only 1 service
- Kill backend then recover it

OUTPUT REQUIRED:

- docker-compose.yml
- All containers running screenshot

PHASE 4: CI/CD WAR ZONE (LAB 04)

Your mission:

Push → Build → Test → Deploy like Google engineers.

Tasks:

1. Jenkins Pipeline:
    - Build image
    - Push
    - Deploy to K8s
2. Make a commit
3. Trigger Jenkins
4. Watch logs
5. Confirm pod updated

OUTPUT REQUIRED:

- Jenkins screenshot
- Git commit
- New pod running proof

BONUS ROUND: DEBUG BATTLE

Each group gets:

- One broken deployment
- One broken pipeline
- One broken compose

You MUST:

- Identify issue
- Fix
- Explain solution