**Your Local Jenkins CI/CD Lab**

This guide will walk you through setting up a **Jenkins instance using Docker**. You can use this lab to practice and understand the core concepts of CI/CD pipelines.

**Step 1: Get Jenkins Running with Docker**

The easiest way to start is to pull the official **Jenkins Long-Term Support (LTS)** image and run it as a container. Open your terminal or command prompt and run the following command.

$ docker run -d --name my-jenkins \

-p 8080:8080 -p 50000:50000 \

-v jenkins\_home:/var/jenkins\_home \

jenkins/jenkins:lts-jdk11

**What this command does:**

* docker run -d: Runs the container in detached mode (in the background).
* --name my-jenkins: Assigns a human-readable name to your container.
* -p 8080:8080: Maps port 8080 on your machine to port 8080 in the container, so you can access the UI.
* -p 50000:50000: Maps port 50000 for Jenkins agents (if you want to scale up later).
* -v jenkins\_home:/var/jenkins\_home: Creates a volume to persist your Jenkins data, so it won't be lost when you stop the container.

**Step 2: Initial Setup & Admin Password**

Once the container is running, you need to retrieve the initial administrator password from the Docker logs.

$ docker logs my-jenkins

Look for a line that says \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* followed by a long alphanumeric password. Copy this password. Now, open your browser and navigate to **http://localhost:8080**. You will be prompted to enter this password to unlock Jenkins and then follow the steps to install recommended plugins and create your first admin user.

**Step 3: Creating a CI/CD Pipeline**

Now that Jenkins is ready, let's create a simple CI/CD pipeline job.

1. From the Jenkins dashboard, click **"New Item"**.
2. Give your project a name (e.g., my-first-ci-cd).
3. Select **"Pipeline"** and click **"OK"**.
4. On the next page, scroll down to the **"Pipeline"** section.
5. Choose **"Pipeline script"** and paste the following Jenkinsfile script into the text area.

This example script runs a simple "build" and "deploy" process using basic shell commands.

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Building the application...'

// You would add your real build commands here, e.g., 'npm install' or 'make'

sh 'echo "Build successful."'

}

}

stage('Deploy') {

steps {

echo 'Deploying to a target environment...'

// You would add your deployment commands here, e.g., 'scp' or 'docker run'

sh 'echo "Deployment successful."'

}

}

}

}

**Step 4: Run the Pipeline & Monitor**

After saving the project, you'll be taken to its dashboard. Click **"Build Now"** on the left menu. Jenkins will run the pipeline, and you can monitor its progress in real-time. To see the output of each command, click on the running build in the **"Build History"** and then select **"Console Output"**. This is where you'll find the logs from your "build" and "deploy" steps.

**Cleanup**

To stop and remove the Jenkins container when you are finished, run:

docker stop my-jenkins and docker rm my-jenkins.

The Jenkins data volume will persist, so you can start a new container with the same volume name to resume your work.