DevOps Lab Program 8

Practical Exercise: Set Up a Jenkins CI Pipeline for a Maven Project, Use Ansible to Deploy Artifacts Generated by Jenkins

🔧 Environment Overview

Assuming the following:

- Jenkins is installed and accessible via web interface.
- Java JDK 17 and Maven 3.9.9 are installed.
- You have WSL (Ubuntu) installed and Ansible configured there.
- You're on a Windows machine.

🧩 Goal Breakdown

✅ Part 1: Jenkins CI Pipeline

- Install necessary Jenkins plugins
- Create a Maven project in Jenkins
- Configure SCM (e.g., GitHub)
- Build Maven project
- Archive artifacts

✅ Part 2: Ansible Deployment via WSL

- Configure Ansible inside WSL
- Pull artifacts from Jenkins (or access shared location)
- Deploy the artifact to a target (e.g., local server or VM)

🔹 Step 1: Jenkins Initial Configuration

1.1 Launch Jenkins

Open your browser:

http://localhost:8080

1.2 Install Jenkins Plugins

Go to: Manage Jenkins → Plugins
Install these:

- **Git Plugin**
- **Pipeline**
- **Maven Integration**
- **Ansible Plugin** (optional, if you want Jenkins to run Ansible directly)

1.3 Configure Global Tools

Go to: Manage Jenkins → Global Tool Configuration

Maven:

- Add Maven 3.9.9 path

Name: Maven-3.9.9

MAVEN_HOME: C:\apache-maven-3.9.9

JDK:

- Add JDK 17 path

Name: jdk-17

JAVA_HOME: C:\Program Files\Java\jdk-17\

🔹 Step 2: Create a Sample Maven Project

2.1 Use a Sample GitHub Repo

You can fork or clone a Maven project like:

https://github.com/spring-projects/spring-petclinic

or create your own:

bash

CopyEdit

mvn archetype:generate -DgroupId=com.example -DartifactId=myapp -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

Push this to GitHub so Jenkins can pull it.

🔹 Step 3: Configure Jenkins Job

3.1 New Job

- Go to: New Item
- Name: maven-ci-pipeline
- Type: **Freestyle Project**

3.2 Source Code Management

- Git → Add your repo URL
- Add credentials if needed

3.3 Build Environment

- Check: "Delete workspace before build starts"

3.4 Build

- Select **Invoke top-level Maven targets**
- Maven Version: Maven-3.9.9
- Goals: clean package

3.5 Post-Build Actions

- **Archive the artifacts**

bash

CopyEdit

Files to archive: target/*.jar

Save and **Build Now**

✓ You should now have a JAR file generated and archived by Jenkins.

🔹 Step 4: Share Artifact with Ansible (via WSL)

You need to make the artifact accessible to Ansible. Easiest way is to copy it to a known shared folder.

4.1 Create a Shared Folder

E.g., create:

C:\ci-artifacts\

Use a Jenkins post-build script (add this as a post-build "Execute Windows Batch Command"):

bat

CopyEdit

copy target*.jar C:\ci-artifacts\

Now Ansible in WSL can access it at:

swift

CopyEdit

/mnt/c/ci-artifacts/

🔹 **Step 5: Configure Ansible in WSL**

5.1 Install Ansible (if not done)

bash

CopyEdit

sudo apt update

sudo apt install ansible

5.2 Inventory File

Create /home/sunil/ansible/inventory:

ini

CopyEdit

[local]

localhost ansible_connection=local

5.3 Ansible Playbook

Create /home/sunil/ansible/deploy.yml:

yaml

CopyEdit

```
- hosts: local
  tasks:
    - name: Copy JAR to deployment directory
      copy:
        src: /mnt/c/ci-artifacts/ spring-petclinic-3.4.0-SNAPSHOT.jar
        dest: /home/sunil/deploy/app.jar
```

Ensure the deploy folder exists:

bash

CopyEdit

mkdir -p ~/deploy

🔹 **Step 6: Run Ansible Playbook**

bash

CopyEdit

cd ~/ansible

ansible-playbook -i inventory deploy.yml

✓ You've now deployed the artifact from Jenkins to a target using Ansible.

✅ **Optional: Run Ansible From Jenkins**

If you want Jenkins to run the Ansible playbook:

1. Install **WSL Plugin** or configure Execute shell:

bash

CopyEdit

wsl ansible-playbook /home/sunil/ansible/deploy.yml

You can put this in a "Post-build" step.

🧠 **Recap**

Step	Task
1	Install Jenkins plugins and configure Maven, JDK
2	Create/push Maven project to GitHub
3	Set up Jenkins pipeline to build and archive
4	Copy artifacts to shared folder
5	Set up Ansible inside WSL
6	Write and run a playbook to deploy artifacts
