Lab 19: End-to-End DevOps Pipeline (Basic Level)

Author: Dr. Sandeep Kumar Sharma
Level: Beginner
Platform: Ubuntu Linux + GitHub + Maven + Docker + Terraform + Ansible + Azure
Prerequisite: Basic Linux, Git, Docker, Terraform, Ansible

Learning Objective

Participants will learn:

- How an end-to-end DevOps pipeline works
- How code moves from developer to production
- How different DevOps tools connect
- Basic CI/CD flow
- Infrastructure + Application automation

Learning Outcome

After completing this lab, participants will:

- Understand complete DevOps lifecycle
- Build a simple DevOps pipeline
- Deploy a live application
- Understand tool integration

Pipeline Architecture

```
Developer
   ↓
GitHub
   ↓
Maven (Build)
   ↓
Docker (Image)
   ↓
Terraform (Infra)
   ↓
Ansible (Config + Deploy)
```

```
    ↓
Live Application
```

Project Overview

We will build:

- Hello Java application
- Build with Maven
- Create Docker image
- Create VM using Terraform
- Configure VM using Ansible
- Run Docker container
- Access live app in browser

Step 1: Create Hello Java Application

Create project

```
mkdir hello-devops
cd hello-devops
```

Create structure:

```
src/main/java/com/example/
```

Create file:

```
nano src/main/java/com/example/HelloApp.java
```

```java
package com.example;

public class HelloApp {
    public static void main(String[] args) {
        System.out.println("Hello DevOps World");
    }
}
```

Step 2: Create Maven pom.xml

```
nano pom.xml
```

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://
maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>hello-devops</artifactId>
    <version>1.0</version>
    <packaging>jar</packaging>
</project>
```

Build:

```
mvn clean package
```

Step 3: Create Dockerfile

```
nano Dockerfile
```

```
FROM openjdk:11-jre-slim
WORKDIR /app
COPY target/hello-devops-1.0.jar app.jar
CMD ["java", "-jar", "app.jar"]
```

Build image:

```
docker build -t hello-devops:1.0 .
```

Run locally:

```
docker run hello-devops:1.0
```

Step 4: Infrastructure with Terraform (Basic VM)

Create folder:

```
mkdir terraform
cd terraform
```

Create `main.tf`:

```
provider "azurerm" {
  features {}
}

resource "azurerm_resource_group" "rg" {
  name     = "rg-devops-pipeline"
  location = "East US"
}
```

```
terraform init
terraform apply
```

Step 5: Ansible for Configuration

Create inventory:

```
nano inventory
```

```
[app]
<vm-public-ip>
```

Create playbook:

```
nano deploy.yml
```

```
- hosts: app
  become: yes
  tasks:
    - name: Install Docker
      apt:
        name: docker.io
        state: present
        update_cache: yes

    - name: Start Docker
      service:
        name: docker
        state: started

    - name: Copy Docker Image
      shell: docker load < hello-devops.tar

    - name: Run Container
      shell: docker run -d -p 8080:8080 hello-devops:1.0
```

Step 6: Deploy Application

```
ansible-playbook -i inventory deploy.yml
```

Step 7: Access Live Application

Browser:

```
http://<vm-public-ip>:8080
```

Output:

```
Hello DevOps World
```

End-to-End Flow

```
Code → Build → Image → Infra → Config → Deploy → Live App
```

Tools Used

- GitHub
- Maven
- Docker
- Terraform
- Ansible
- Azure

Cleanup

- Stop container
- Destroy VM
- Terraform destroy

Learning Summary

This lab demonstrates:

- DevOps lifecycle
- Toolchain integration
- Automation mindset
- Infrastructure + Application automation
- Real DevOps workflow

> "DevOps is not a toolchain. DevOps is flow, automation, and culture."