

Experiment -2

(Creating a Jenkins Pipeline with a Jenkins File)

Objective: Create a Jenkins pipeline using a Jenkins file that builds a simple project, runs tests, and deploys the project to a designed environment.

1. Create a simple maven Project (the same one used in Experiment -1) and push the project to GitHub.

Vidu7060 Update MyCalss.java 8b36850 on Sep 26 13 commits		
📁 .github/workflows	Update cicd.yml	3 months ago
📁 .settings	GitHub Actions	3 months ago
📁 src/main/java/devops/b3/cicd/lab3	Update MyCalss.java	3 months ago
📄 .classpath	GitHub Actions	3 months ago
📄 .gitignore	GitHub Actions	3 months ago
📄 .project	GitHub Actions	3 months ago
📄 Dockerfile	Update Dockerfile	3 months ago
📄 pom.xml	GitHub Actions	3 months ago

2. Create a Jenkins file and push it to GitHub repo (in the root dir) Note: the following Jenkins file includes stages associated with maven project


```
pipeline {
    agent any
    tools {
        maven 'MAVEN_HOME'
    }
    stages {
        stage('Stage1: Hello Clean Stage 1') {
            steps {
                bat 'mvn clean'
            }
        }
        stage('Stage 2: Test Stage') {
            steps {
                bat 'mvn test'
            }
        }
        stage('Stage 3: My Package'){
            steps {
                bat 'mvn package'
            }
        }
        stage('Stage 4: My Final Build Stage'){
            steps {
                bat 'mvn install'
            }
        }
        stage('Stage Final: Build Success'){
            steps {
                echo 'Build Success!'
            }
        }
    }
}
```

3. Create a Jenkins Pipeline CONFIGURATION:


Enter an item name

Lab2


» Required field

**Freestyle project**


This is the central feature of Jenkins. Jenkins will build your project, compile it, and run tests for something other than software build.

**Maven project**


Build a maven project. Jenkins takes advantage of your POM files and drives the build process.

**Pipeline**

Orchestrates long-running activities that can span multiple build agents. Useful for organizing complex activities that do not easily fit in free-style jobs.

**Multi-configuration project**

Suitable for projects that need a large number of different configurations. Useful for building multiple variants of a product.

**Folder**

Creates a container that stores nested items in it. Useful for grouping things that share a common namespace, so you can have multiple things of the same name.

OK

4. Select pipeline Script from SCM (git) and provide GitHub repo URL.

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/Vidu7060/CICD-LAB-3-4

5. Provide the Jenkins file path (path in the repo) in the Script Path.

Script Path ?

Jenkinsfile

☒ Lightweight checkout ?

[Pipeline Syntax](#)

6. Now if we make any changes to the GitHub repo and push it, it will trigger a new pipeline build.

