DevOps Lab Experiment 10: Creating Build Pipelines for a Maven Project with Azure Pipelines

1. Install IntelliJ IDEA (Community Edition)

- Download: https://www.jetbrains.com/idea/download/?section=linux
- Or install via terminal:

  $ sudo snap install intellij-idea-community --classic

  $ intellij-idea-community

- Note: Java SDK must be pre-installed.

2. Create a Maven Project in IntelliJ

- Open IntelliJ > New Project
- Select:
  - Build System: Maven
  - Language: Java
- Name your project.

3. Install and Configure Git

- Check Git:

  $ git --version

- If not installed:

  $ sudo apt install git

- Initialize Git in project folder:

  $ git init

  $ git add .

  $ git commit -m "First Commit"

  $ git branch -M main

4. Push Project to GitHub

- Create a private GitHub repository (same name as your project).

- Configure Git:

  $ git config --global user.name "Your_Name"

  $ git config --global user.email "Your_Email"

- Set up SSH:

  $ ssh-keygen -t rsa -b 4096 -C youremail@gmail.com

  $ cat ~/.ssh/id_rsa.pub

- Add SSH key in GitHub: Settings > SSH and GPG keys > New SSH key

- Link remote:

  $ git remote set-url origin git@github.com:username/repo_name.git

  $ git push --set-upstream origin main

  $ git push

5. Create Azure Pipeline

- Go to https://dev.azure.com/

- Project > Pipelines > Create Pipeline

- Select:

  - GitHub YAML

  - Your repo

  - Maven pipeline

- Azure generates a YAML file (based on pom.xml)

- Save and Run

6. Configure YAML for Unit Testing and Reports

Include in the YAML file:

```
trigger:
- main

pool:
  vmImage: 'ubuntu-latest'

steps:
- task: Maven@3
  inputs:
    mavenPomFile: 'pom.xml'
    goals: 'clean package'

- task: PublishTestResults@2
  inputs:
    testResultsFiles: '**/target/surefire-reports/TEST-*.xml'
    mergeTestResults: true
    testRunTitle: 'Maven Unit Test Results'
```

7. Run and Monitor Pipeline

- Commit YAML file to repo
- Azure Pipelines triggers automatically or run manually
- View Logs, Build status, and Test summary under Tests tab