1. Experiment 1 - Installation of Maven and Gradle

**1. Install Java 17 (or higher)**

sudo apt update

sudo apt install openjdk-17-jdk -y

**2. Verify Java Installation**

java --version

**3. Set Java 17 (or higher) as Default if it’s not already set**

sudo update-alternatives --config java

*Follow the prompts to select Java 17 (or higher).*

**1. Check if Gradle is Installed**

gradle -v

**2. Remove Gradle Installed via APT**

sudo apt remove gradle -y

**3. Remove Gradle Installed via SDKMAN**

ls ~/.sdkman

If it exists, then uninstall Gradle using:

sdk uninstall gradle <version?> --force

#Replace <version> with the installed version for example if version was 8.13 the line would be -

#sdk uninstall gradle 8.13 —force

**1. Install Maven**

sudo apt install maven -y

**2. Verify Maven Installation**

mvn -version

**1. Download and Extract Gradle**

wget https://services.gradle.org/distributions/gradle-8.14-bin.zip -P /tmp

sudo unzip -d /opt/gradle /tmp/gradle-8.14-bin.zip

**2. Set Environment Variables Permanently (for Bash)**

Modify bash command file by running the following command:

gedit ~/.bashrc

**At the end of the file, add the following lines:**

export GRADLE\_HOME=/opt/gradle/gradle-8.14

export PATH=${GRADLE\_HOME}/bin:${PATH}

--

source ~/.bashrc

**3. Verify Gradle Installation**

gradle -v

2.working with maven

mvn archetype:generate -DgroupId=com.example -DartifactId=MyMavenApp -DarchetypeArtifactId=maven-archetype-quickstart

-DinteractiveMode=false

-----

cd MyMavenApp

---

sudo snap install tree

tree

gedit pom.xml

after url and before dependencies put this code

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

   </properties>

mvn compile

mvn test

mvn package

java -cp target/classes com.example.App

3. 3rd prgm

1. gradle -v (if not installed install)

2. mkdir HelloGradle

3. cd HelloGradle

4. gradle init --type java-application –dsl groovy --overwrite

5. gedit app/build.gradle

task hello {

doLast {

println 'Hello, Gradle!'

}

}

6. gradle build

7. gradle run

8.gradle hello

4. 4th prgm

1. mvn archetype:generate -DgroupId=com.example -DartifactId=HelloMaven

DarchetypeArtifactId=maven-archetype-quickstart

DinteractiveMode=false

2. cd HelloMaven

2nd prgm

3. gradle init

6. gedit build.gradle

add id 'application' to plugins and

application

{

mainClass = 'com.example.App'

}

7. gradle build

8. gradle run

5.installing Jenkins

**1. Update Your System**

sudo apt update

sudo apt upgrade -y

**2. Add the Jenkins Repository Key**

sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc

\ <https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key>

**3. Add the Jenkins Repository to Your Sources List**

echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc]" \

https://pkg.jenkins.io/debian-stable binary/ | sudo tee \

/etc/apt/sources.list.d/jenkins.list > /dev/null

**4. Update Your Package List and Install Jenkins**

sudo apt update

sudo apt install jenkins -y

**5. Start and Enable the Jenkins Service**

sudo systemctl start Jenkins

sudo systemctl enable Jenkins

**check-**

sudo systemctl status Jenkins

**Access Jenkins and Complete Initial Setup**

1. Open a web browser and navigate to the Jenkins default port:

[localhost:8080](http://localhost:8080)

1. sudo cat /var/lib/jenkins/secrets/initialAdminPassword
2. Copy the displayed password and paste it into the "Administrator password" field in your browser. Click "Continue".
3. On the next screen, click "Install suggested plugins" to install a recommended set of plugins.
4. After the plugins are installed, you will be asked to create an administrator user. Fill in the required details and click "Save and Finish".
5. You should now see the Jenkins dashboard, ready for use.

6.jenkins and github

Start with 5th project

localhost:8080

enter the password by creating through command

enter item name: Maven-GitHub-Freestyle

freestyle project

ok

git

<https://github.com/devops-ds/your-maven-project.git>

change master to main

add build step

execute shell

#/path/to/your/maven/bin/mvn clean install

# Example:

/usr/bin/mvn clean install

Add post build action:publish junit test result report

\*\*/target/surefire-reports/\*.xml

Save

Build project

New item

Maven-GitHub-Pipeline

--under script

pipeline {

agent any

stages {

stage('Checkout') {

steps {

git url: 'https://github.com/devops-ds/your-maven-project.git', branch: 'main'

}

}

stage('Build') {

steps {

sh '/usr/bin/mvn clean package'

}

}

stage('Test') {

steps {

echo 'Tests are typically run during the Build stage with Maven.'

}

}

}

post {

always {

junit '\*\*/target/surefire-reports/\*.xml'

}

success {

echo 'Build and tests succeeded!'

}

failure {

echo 'Build or tests failed.'

}

}

}

--Ok

Build

7.using ansible

sudo apt update

sudo apt upgrade -y

--

sudo apt install ansible -y

--

ansible –version

--creating a ansible inventory

-- go to hosts.ini

gedit hosts.ini

[local]

localhost ansible\_connection=local

--

gedit setup.yml

---

- name: Basic Server Setup

hosts: local

become: yes # Use privilege escalation (sudo)

tasks:

- name: Update apt cache

apt:

update\_cache: yes

- name: Install curl

apt:

name: curl

state: present

--

sudo ansible-playbook -i hosts.ini setup.yml

8 . maven and gradle

Localhost:8080

HelloMaven-CI

Freestyle project

--ok

Git

<https://github.com/devops-ds/your-maven-project.git>

master to main

Click "Add build step" and select **"Execute shell".**

#/path/to/your/maven/bin/mvn clean package

# Example:

/usr/bin/mvn clean package

Click "Add build step" and select **"Execute shell".**

ansible-playbook -i hosts.ini deploy.yml

Add post build action: archive artifacts

target/\*.jar

--save

9. azure

Login

Search- azure devops organization

My azure devops organisations

Create new organisation

--org name

--project name

--private

done