DEVOPS EXPERIMENT 4

Maven installation in ubunti aws

To install Maven on an AWS EC2 instance running Ubuntu 20.04 and configure it in Jenkins, you can follow these steps:

\*\*Step 1: Connect to your EC2 instance\*\*

Use SSH to connect to your AWS EC2 instance. You can use the SSH key pair you used when launching the instance.

```bash

ssh -i your-key.pem ubuntu@your-ec2-instance-ip

```

\*\*Step 2: Update and Upgrade the System\*\*

Update the package list and upgrade installed packages to ensure you have the latest software information.

```bash

sudo apt update

sudo apt upgrade

```

\*\*Step 3: Install Apache Maven\*\*

You can install Apache Maven from the Ubuntu package repository using `apt`. This will install the default Maven version available in the Ubuntu repository.

```bash

sudo apt install maven

```

To verify the installation, you can check the installed Maven version:

```bash

mvn -version

```

\*\*Step 4: Configure Jenkins\*\*

Assuming you have Jenkins installed and running on your EC2 instance (you can follow the previous steps mentioned in the first response to set up Jenkins), follow these steps to configure Jenkins to use the newly installed Maven:

1. Open Jenkins in your web browser by navigating to `http://your-ec2-instance-ip:8080`.

2. Log in to Jenkins with your admin account.

3. Click on "Manage Jenkins" in the left-hand sidebar.

4. Click on "Global Tool Configuration."

5. Scroll down to the "Maven" section, and click on the "Add Maven" button.

6. Provide a name for the Maven installation (e.g., "Maven 3.6.3").

7. In the "MAVEN\_HOME" field, provide the path to the Maven installation. By default, on Ubuntu, it should be `/usr/share/maven`.

8. Click "Save" to save the Maven configuration.

\*\*Step 5: Configure Jenkins Job to use Maven\*\*

Now that you've configured Maven in Jenkins, you can create or modify a Jenkins job to use Maven as the build tool:

1. Create a new Jenkins job or edit an existing one.

2. In the job configuration, under the "Build" section, choose "Invoke top-level Maven targets."

3. In the "Goals" field, specify the Maven goals you want to execute, e.g., "clean install."

4. In the "Maven Version" field, select the Maven installation you configured earlier (e.g., "Maven 3.6.3").

5. Save your job configuration.

\*\*Step 6: Build the Project\*\*

You can now manually trigger the Jenkins job, and it will use Maven for the build process as specified in the job configuration.

This setup allows Jenkins to use the installed Maven for your project's builds. Make sure to configure your Jenkins job's source code management and other build-related settings as needed based on your project requirements.

EXPERIMENT 5\

To install Apache Tomcat on an AWS EC2 instance running Ubuntu, you can follow these steps:

\*\*Step 1: Connect to your EC2 Instance\*\*

Use SSH to connect to your AWS EC2 instance. You can use the SSH key pair you used when launching the instance.

```bash

ssh -i your-key.pem ubuntu@your-ec2-instance-ip

```

\*\*Step 2: Update and Upgrade the System\*\*

Update the package list and upgrade installed packages to ensure you have the latest software information.

```bash

sudo apt update

sudo apt upgrade

```

\*\*Step 3: Install Tomcat\*\*

To install Apache Tomcat on Ubuntu, you can use the `apt` package manager. Run the following command:

```bash

sudo apt install tomcat9

```

During the installation, you will be asked to confirm whether you want to continue. Type 'Y' and press Enter.

\*\*Step 4: Start and Enable Tomcat\*\*

Tomcat should start automatically after installation, but you can ensure it's running and set to start at boot with the following commands:

```bash

sudo systemctl start tomcat9

sudo systemctl enable tomcat9

```

\*\*Step 5: Adjust Firewall Rules\*\*

If you have a firewall enabled (e.g., UFW), you'll need to allow traffic to Tomcat's default port (8080). Run the following command to open the port:

```bash

sudo ufw allow 8080/tcp

```

\*\*Step 6: Access the Tomcat Web Application Manager (Optional)\*\*

Tomcat comes with a web application manager that you can use to deploy and manage applications. By default, it's not accessible from the internet. To access it remotely, you can edit the `tomcat-users.xml` file and configure user roles and access.

Edit the `tomcat-users.xml` file:

```bash

sudo nano /etc/tomcat9/tomcat-users.xml

```

Add the following lines inside the `<tomcat-users>` section, replacing `your\_username` and `your\_password` with your desired credentials:

```xml

<role rolename="manager-gui"/>

<user username="your\_username" password="your\_password" roles="manager-gui"/>

```

Save and close the file.

\*\*Step 7: Restart Tomcat\*\*

After making changes to the `tomcat-users.xml` file, you need to restart Tomcat for the changes to take effect:

```bash

sudo systemctl restart tomcat9

```

\*\*Step 8: Access Tomcat Manager\*\*

You can access the Tomcat Manager web application by going to:

`http://your-ec2-instance-ip:8080/manager/html`

You will be prompted to enter the username and password you configured in the `tomcat-users.xml` file. Once logged in, you can deploy and manage web applications.

That's it! You've successfully installed and configured Apache Tomcat on your AWS EC2 instance running Ubuntu. You can now deploy your web applications to Tomcat as needed.

pipeline {

agent any

tools {

jdk 'JDK 8' // Use the name you configured for the JDK

maven 'Maven' // Use the name you configured for Maven

}

stages {

stage('Checkout') {

steps {

git 'https://github.com/yourusername/your-repo.git'

}

}

stage('Build and Test') {

steps {

sh 'mvn clean install'

}

}

stage('Deploy to Tomcat') {

steps {

sh 'cp target/your-app.war /path/to/tomcat/webapps/'

}

}

}

}

stage('Deploy to Tomcat') {

steps {

sh 'cp target/drools-hello-world-javainuse2-0.0.1-SNAPSHOT.war /path/to/tomcat/webapps/'

}

}

**/var/lib/tomcat9**