**Lab: Build Pipelines**

**Pre-Requisites: -**

1. Git Repository with the code to be deployed should be available. The git repo to be used in training: https://github.com/LovesCloud/java-tomcat-build-pipeline

Please fork the repo in your GitHub account.

1. [Configure GitHub Webhook for Jenkins](#_Configure_GitHub_Webhook) on your forked repo. follow the steps mentioned in [**Configure GitHub Webhook for Jenkins**](#_Configure_GitHub_Webhook_1)
2. Deployment server i.e. Tomcat Server should be set up. For Lab, We have the Tomcat server ready.
3. Create Tomcat ssh user in your Jenkins following the steps mentioned in [**Create Tomcat SSH User**](#_toc77)
4. Install Plugin SSH Agent Plugin
5. maven 3.5.4 should be configured in global tools configuration. This you should have done in one lab earlier. Please ensure that it is present.
6. Ensure to run this on jenkins master only as for pipeline project we do not have option to select slave.

**Steps to Follow at GitHub end:**

1. Open **pom.xml** from root directory, click pencil icon to edit it, Replace line <finalName>**demojavapipeline\_newuser**</finalName> with <finalName>**demojavapipeline\_<yourname>**</finalName> and commit the changes.
2. Open **Jenkinsfile** in your root directory of your repo, click pencil icon to edit it.
3. In **SCM Checkout** stage, Replace the https://github.com/LovesCloud/java-tomcat-build-pipeline with url of your forked repo.
4. In stage **Deploy**, Replace IP **demojavapipeline\_newuser** with your war name given in step 1 above and commit the changes.
5. Keep a note of the Tomcat Server IP mentioned under **Deploy** stage

**Steps to Follow at Jenkins end:**

**Section A:**

1. Click **New Item** link on left panel
2. Enter an **Item name** like <yourname>\_pipeline
3. **Note: PLEASE DO NOT INCLUDE CAPITAL LETTER IN THE NAME IN STEP-8**
4. Select **Pipeline**
5. Click OK
6. Under Pipeline section, Select **Definition** as “Pipe line script from SCM”
7. Select **SCM** as Git
8. Enter Repository URL as <repo to be used>
9. Keep **Script Path** as “Jenkinsfile”
10. Click Save
11. On Left Panel, Click on **Build now**
12. View the Pipeline being build and showing Stages.
13. Once all stages are successfully completed. Click on the latest triggered build number from left panel.
14. View the Console Output

**Section B:**

**Verifying the deployed application:**

1. Take the Tomcat Server IP from the Jenkinsfile **Deploy** stage
2. Open any browser
3. Type http://**<Tomcat Server IP>:8080**/**<file name of the war given in pom.xml>**
4. Hit Enter;
5. It will dispaly the web page of the deployed application.

**Section C:**

**Steps to Follow for automated CICD pipeline:**

1. On GitHub end, edit pom.xml, Change the name of the war file name again of your choice and Commit the changes.
2. On the Jenkins end, Open the Job in Configure mode, *Under Build Triggers section, Select* ***GitHub hook trigger for GITScm polling*** *checkbox.*
3. Click **Save.**
4. On GitHub end, edit Jenkinsfile,Replace the name of war under Stage **Deploy**  with the one used in step 1 under **Section C:**  and commit the changes.
5. Open the Job Details page and observe that the build is triggered automatically as soon as you committed the file in GitHub in your repo.
6. View the Pipeline being build and showing Stages.
7. Once all stages are successfully completed. Click on the latest triggered build number from left panel.
8. View the Console Output

*The complete log of the steps involved in successful building of job are displayed.*

1. **I**n order to view the latest deployment, repeat the steps mentioned in section **Verifying the deployed application** above and use the war file name you gave in Section C:-> step 1

# **Create Tomcat SSH User**

1. Login to Jenkins;
2. On the Left Navigation panel, Click on  **Credentials** link
3. Under **Stores Copies to Jenkins** section,Click on **Global** link
4. Click **Add Credentials**
5. Keep **Kind** as SSH Username with private key
6. Enter **ID**  as tomcatsshuser
7. Enter **Description**  as tomcatsshuser
8. Enter **Username**  as jenkins
9. For Private key, Select **Enter directly**  radio button
10. Click **Add** under Key textarea
11. Copy a Private key

**Note: The Private key is available with Trainer.**

1. Click **OK**

# **Configure GitHub Webhook for Jenkins**

1. Open GitHub
2. Navigate to Git Repo;
3. Navigate to Settings of repository
4. Click Webhook
5. Click Add Webhook
6. Enter Payload URL-http://<Public IP of Jenkins Server>:8080/github-webhook/
7. Click Save Webhook