Lab: CICDPipelineUsingDockerAndJenkins

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-demo-app
   Please fork the repo in your GitHub account.
2. Configure GitHub Webhook for Jenkins on your forked repo. follow the steps mentioned in **Configure GitHub Webhook for Jenkins**
3. To publish docker image on DockerHub, Follow the settings mentioned in section **Configuring Jenkins for publishing docker image into DockerHub .**
4. Keep a note of your AWS Instance Public IP, it will be used as Deployment Server IP in this lab.

Steps to Follow at GitHub end:

1. Open **Jenkinsfile** in your root directory of your repo, click pencil icon to edit it.
2. In line def dockerImageName= 'rajnikhattarrsinha/javademoapp_$JOB_NAME:$BUILD_NUMBER'
   Replace **rajnikhattarrsinha** with **<your dockerhub username>.**
3. In **SCM Checkout** stage, Replace the https://github.com/LovesCloud/java-tomcat-demo-app with url of your forked repo.
4. In stage **Publish Docker Image**, Replace "**dockerpwd**" with "**dockerpwd<yourname>**"
5. In stage **Run Docker Image**, Replace IP **34.229.191.73** with **your AWS Instance Public IP** and commit the changes.
6. Open **pom.xml** from root directory, click pencil icon to edit it,
   Replace line <finalName>**demopipeline_rajni**</finalName> with <finalName>**demopipeline_<yourname>**</finalName> and commit the changes.

Steps to Follow at Jenkins end:

7. Click **New Item** link on left panel
8. Enter an **Item name** like <yourname>_pipeline

Note: PLEASE DO NOT INCLUDE CAPITAL LETTER IN THE NAME IN STEP-8

9. Select **Pipeline**
10. Click OK
11. Select General Tab, **GitHub Project** as <repo to be deployed>
12. Under Build Triggers section, Select **GitHub hook trigger for GITScm polling** checkbox.
13. Under Pipeline section, Select **Definition** as "Pipe line script from SCM"
14. Select **SCM** as Git
15. Enter Repository URL as <repo to be used>
16. Keep **Script Path** as "Jenkinsfile"

17. Click Save
18. On Left Panel, Click on **Build now**
19. View the Pipeline being build and showing Stages.
20. Once all stages are successfully completed. Click on the latest triggered build number from left panel.
21. View the Console Output

Verifying the deployed application:

A. Take the **Public IP** of your **AWS Instance** from **AWS Console**.
B. Open any browser
C. Type http://**<Public IP of the AWS Instance>**:**8082/<file name of the war given in pom.xml>**
D. Hit Enter;

Steps to Follow for automated CICD pipeline:

22. On GitHub end, edit pom.xml, Change the name of the war file name again of your choice and Commit the changes.
23. Open the Jenkins end, 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.
24. View the Pipeline being build and showing Stages.
25. Once all stages are successfully completed. Click on the latest triggered build number from left panel.
26. View the Console Output

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

27. **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 step 22.

Configuring Jenkins for publishing docker image into DockerHub

1.) Navigate to **Credentials** in Jenkins opened in browser
2.) Click "**global**" link in section Stores scoped to Jenkins
3.) Click **Add Credentials** link
4.) Select **Kind**= Secret text
5.) Enter **Secret** =<Type the docker hub password>
Note: If your dockerhub password has $ character in it, please prefix it with \ (backward slash)
For Example: Modify **Dev0p$!!/** with **Dev0p\$!!/** and then enter in **Secret** field.
6.) Enter **ID** as "dockerpwd<yourname>"
Note: please keep value as mentioned. This is used in pipeline script.
7.) Enter **Description** as "dockerpwd<yourname>"

8.) 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