18. Demonstrate how to build and configure CI/CD pipeline with Selenium WebDriver.

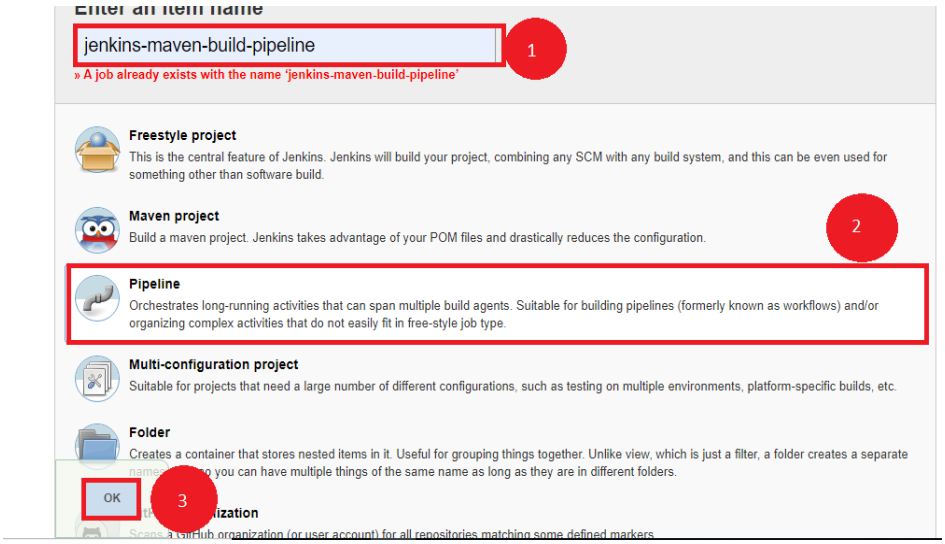
Step 1: Foking the git repository

- Fork the following repository

  https://github.com/canindit75/JenkinsDemo

Step 2: Create a Jenkins pipeline job

- Java 1.8 is already installed in your practice lab. Refer to QA to QE lab guide for Phase 1 for more information.
- Jenkins.war file is already present in your practice lab in cd /usr/share/jenkins directory.
- Go to jenkins.war location Start the Jenkins by using command on command prompt:**java -jar jenkins.war.**
- Open browser and type **localhost:8080.**
- Enter the password.
- Create a job.
- Pass a name.
- Select **Pipeline.**
- Click on Ok.



- Create a text file name it **run.sh** in your lab and keep the below given code in it.

```
java -cp bin;lib/* org.testng.TestNG testng.xml
```

- Give executable permission to **run.sh** using the commands below:

  chmod 755 run.sh

  chmod 777 run.sh

- Push **run.sh in your repository** under master branch.

git push <reponame> master

git status

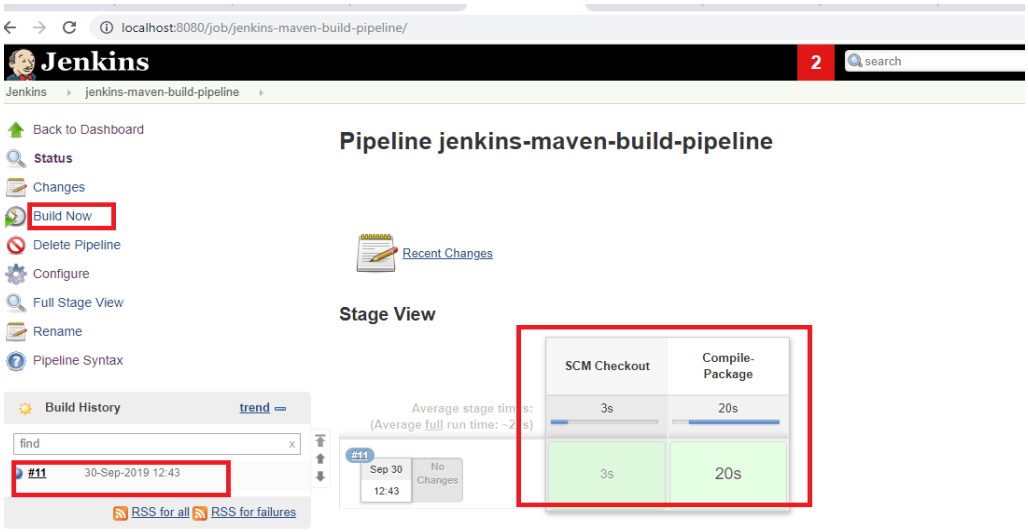
- Go to Jenkins pipeline job.
- Write a groovy script in the pipeline.

```
node {
   def mvnHome
   stage('Preparation') { // for display purposes
      // Get some code from a GitHub repository
      git 'https://github.com/jglick/simple-maven-project-with-tests.git'
      // Get the Maven tool.
      // ** NOTE: This 'M3' Maven tool must be configured
      // **       in the global configuration.
      mvnHome = tool 'maven3'
   }
   stage('Build') {
      // Run the maven build
      withEnv(["MVN_HOME=$mvnHome"]) {
         if (isUnix()) {
            sh '"$MVN_HOME/bin/mvn" -Dmaven.test.failure.ignore clean
package'
         } else {
```

```
            sh  '"%MVN_HOME%\bin\mvn" -Dmaven.test.failure.ignore clean
package'
          }
      }
   }
   stage('Results') {
      junit '**/target/surefire-reports/TEST-*.xml'
      archiveArtifacts 'target/*.jar'
   }}
```

- Click on Apply and Save.

- Click on Build now.



Step 3: Pushing the code to GitHub repositories

Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add .

Commit the changes using the following command:

git commit . -m “Changes have been committed.”

Push the files to the folder you initially created using the following command:

git push -u origin master