Assignment 1

Tasks to do:

- Create a maven project
- Push maven project to GitHub
- Create Jenkinsfile in the project
- Configure Maven, Jenkins server and Nexus Server

Prerequisites (installed software):

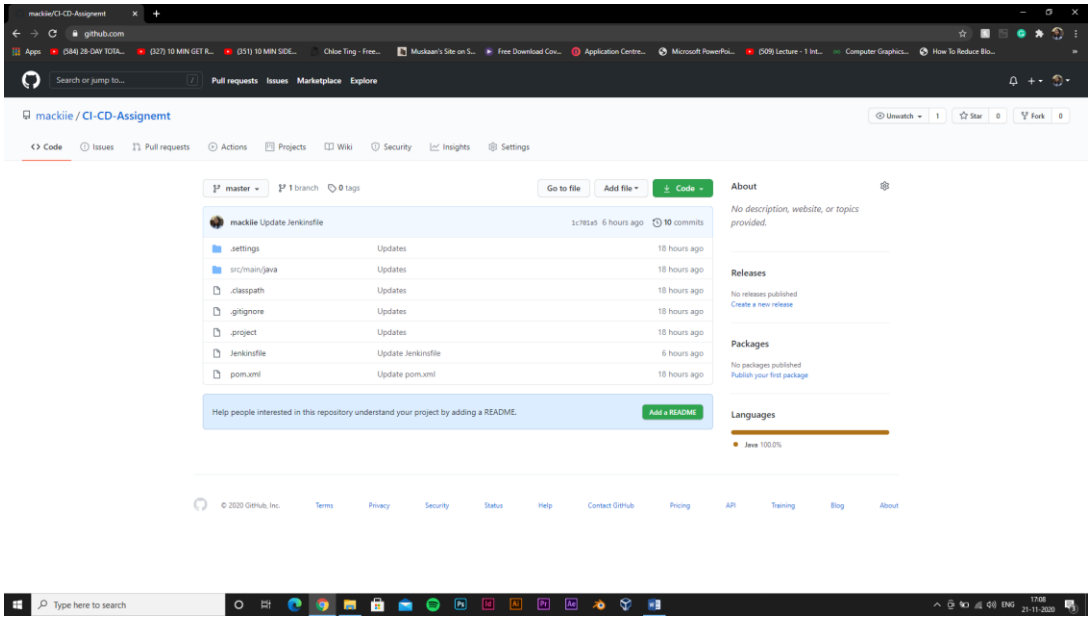
- Java (java 11)
- Maven
- Jenkins
- Sonatype Nexus

Solution:

Step 1:

Upload a maven project on GitHub and create a Jenkinsfile in it

The Jenkinsfile will be used by Jenkins to create and run the pipeline

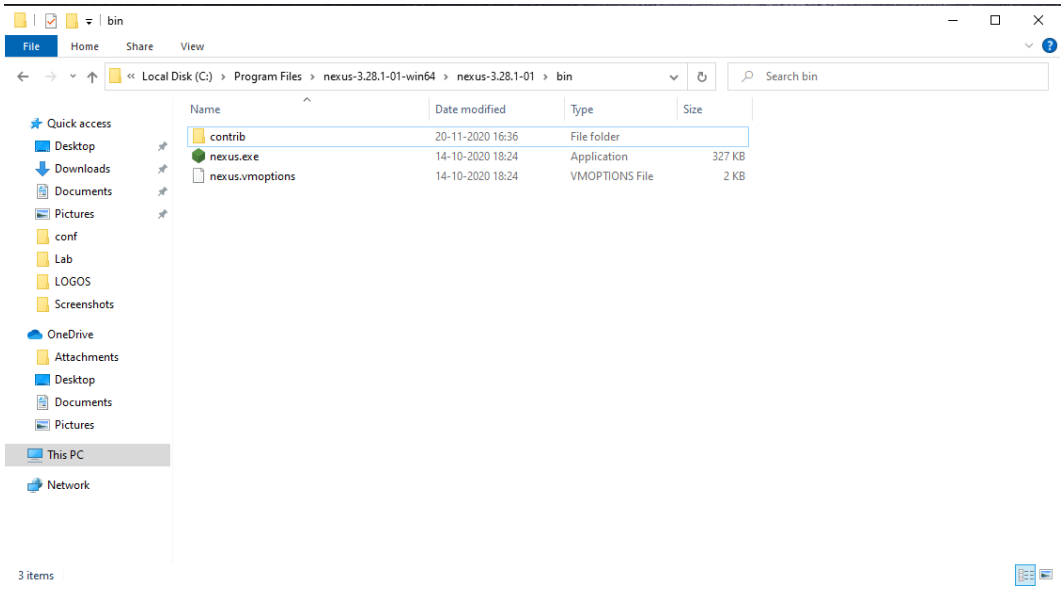


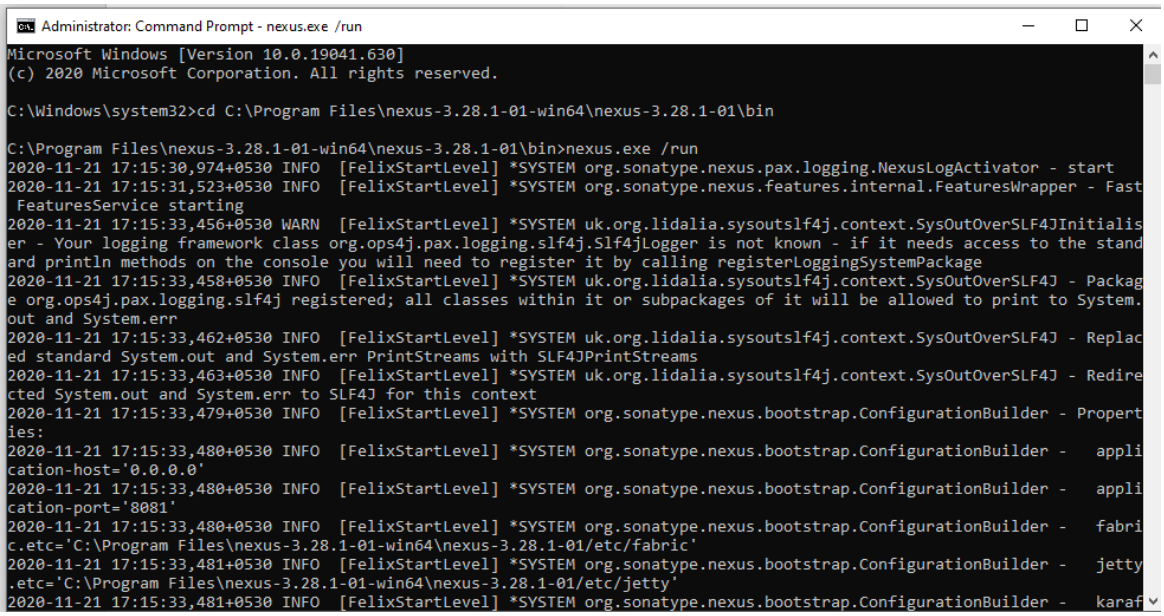
Step 2:

Go to the location on your system where nexus is installed.

Open the bin folder to find nexus.exe file

Open command prompt in the same folder and run the following command: nexus.exe /run

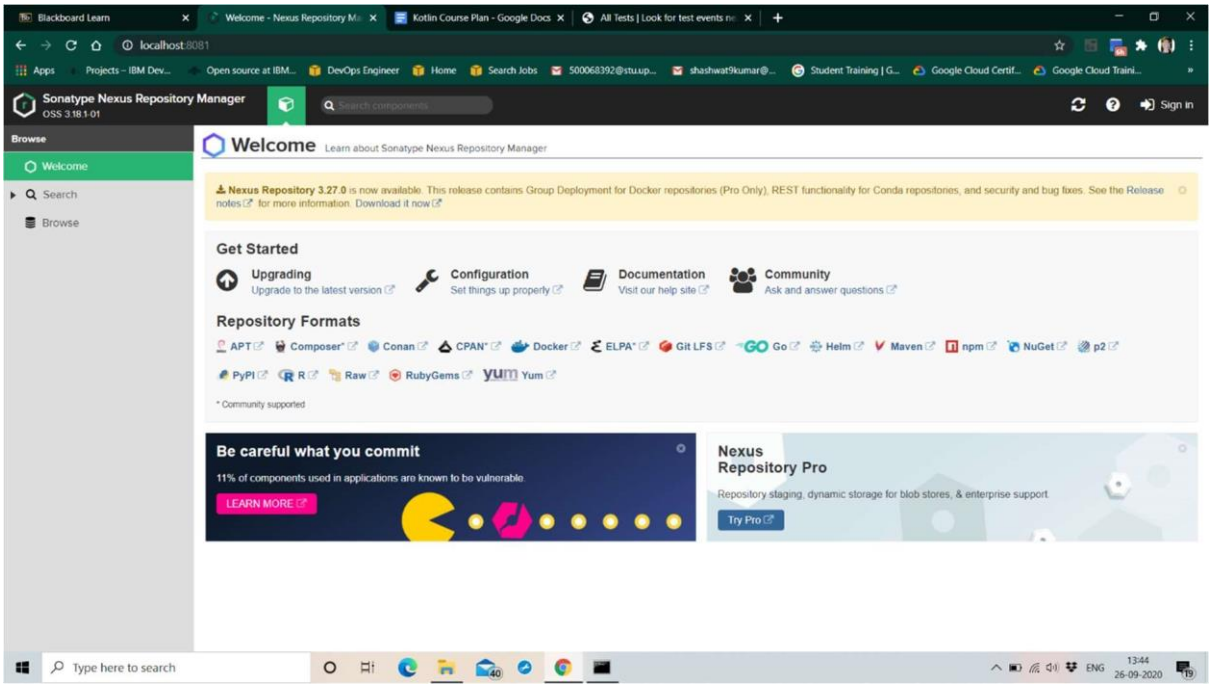




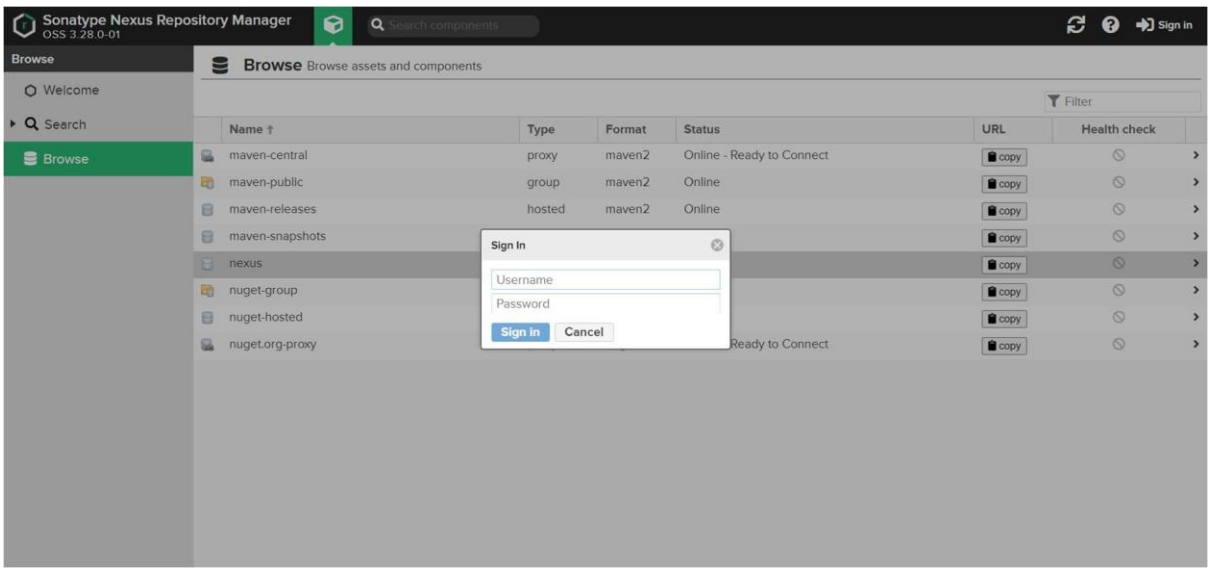
On successful start-up of nexus the terminal shows the following:

```
2020-11-21 17:16:14,866+0530 INFO  [jetty-main-1] *SYSTEM org.eclipse.jetty.server.AbstractConnector - Started ServerCon
nector@77868ff7{HTTP/1.1, (http/1.1)}{0.0.0.0:8081}
2020-11-21 17:16:14,866+0530 INFO  [jetty-main-1] *SYSTEM org.eclipse.jetty.server.Server - Started @54752ms
2020-11-21 17:16:14,866+0530 INFO  [jetty-main-1] *SYSTEM org.sonatype.nexus.bootstrap.jetty.JettyServer -
--------------------------------------------------

Started Sonatype Nexus OSS 3.28.1-01

--------------------------------------------------
2020-11-21 17:16:15,725+0530 INFO  [qtp322277527-51] *UNKNOWN org.apache.shiro.session.mgt.AbstractValidatingSessionMana
```

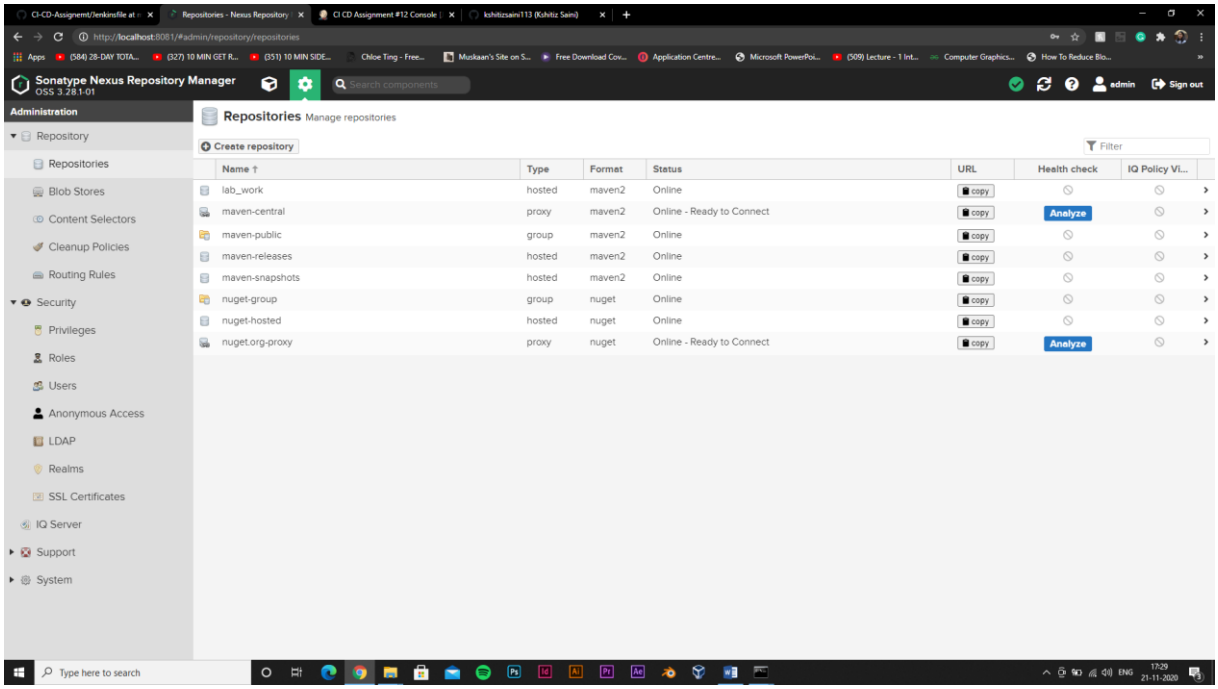
Once the nexus server runs head to localhost:8081, the port where Nexus server runs



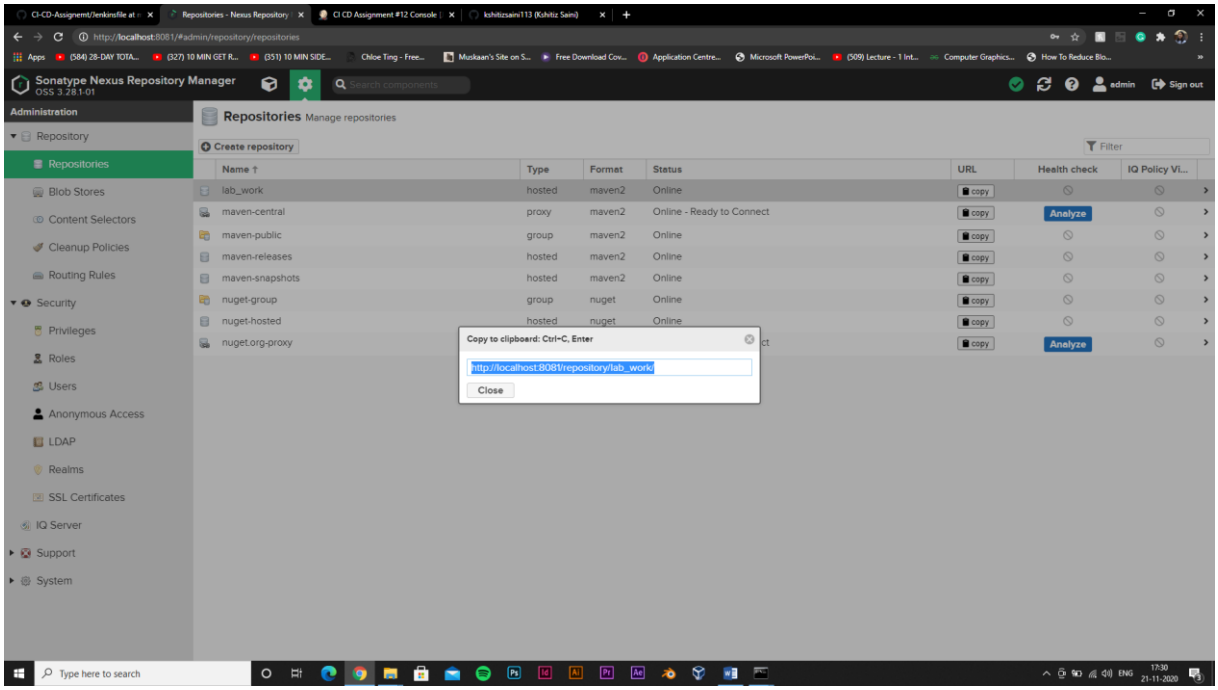
Log in into nexus server using your username and password



On login, Nexus server dashboard should look similar to this



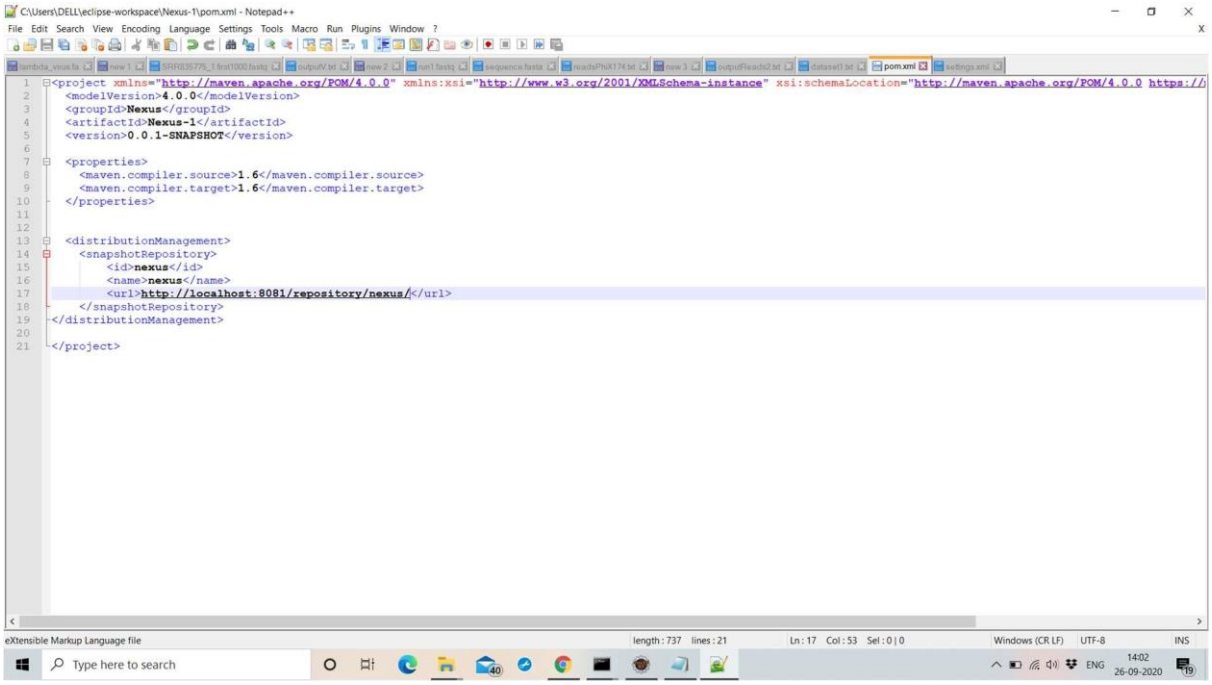
Create a new repository in Nexus (here called “nexus”) and copy its URL



Step 3:

Open the pom.xml of the Maven project whose artifact is to be published to Nexus

Add the snapshot Repository tag into the pom.xml and give the URL of Nexus repository copied

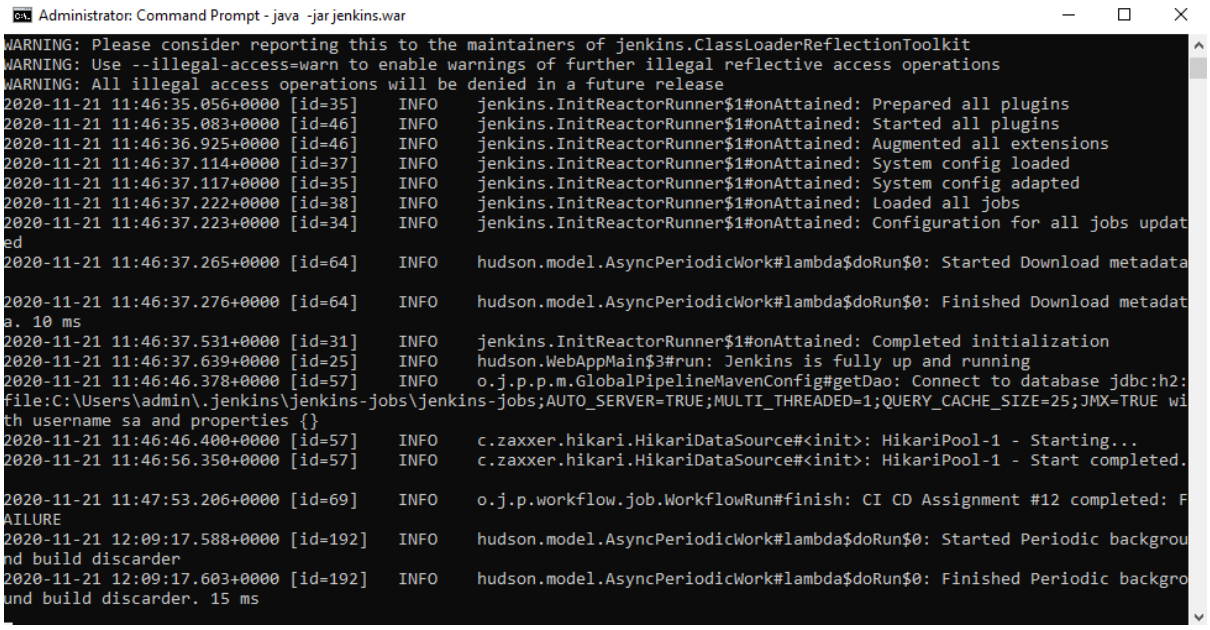


Open the settings.xml file from the place where maven is installed and edit the <servers> tag in it to add the username and password for the deployment server

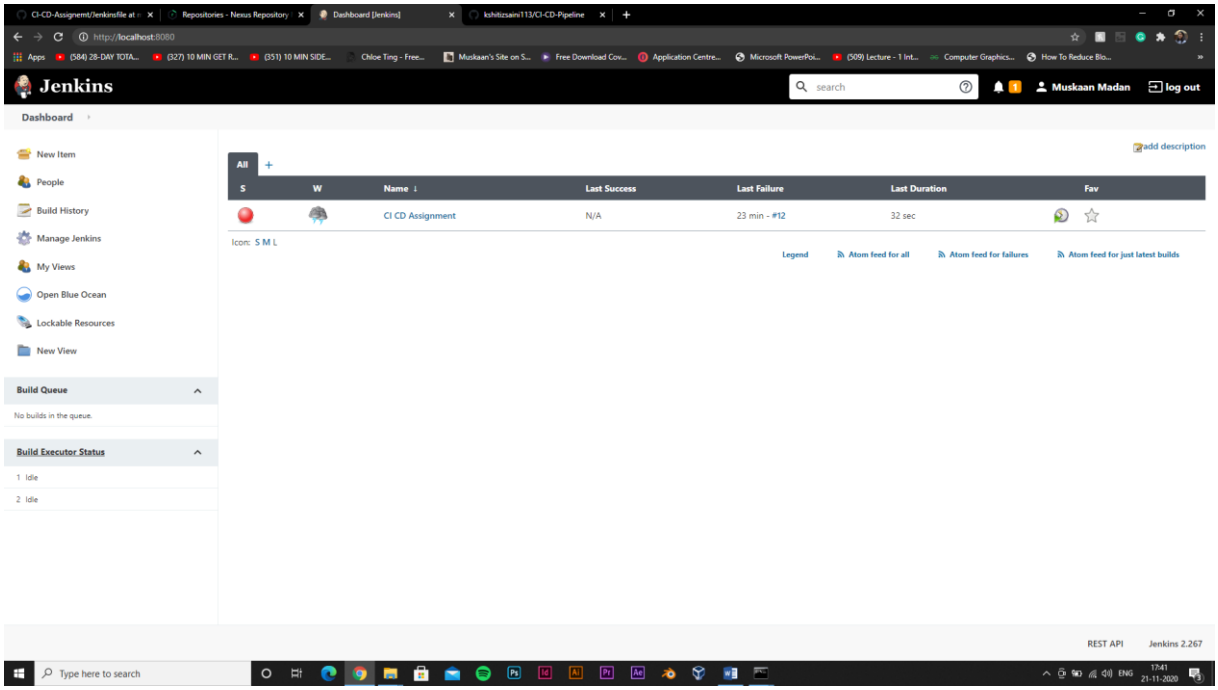
```
111  <servers>
112    <!-- server
113     | Specifies the authentication information to use when connecting to a particular server, identified by
114     | a unique name within the system (referred to by the 'id' attribute below).
115     |
116     | NOTE: You should either specify username/password OR privateKey/passphrase, since these pairings are
117     |       used together.
118     |
119    <server>
120      <id>deploymentRepo</id>
121      <username>repouser</username>
122      <password>repopwd</password>
123    </server>
124    -->
125
126    <!-- Another sample, using keys to authenticate.
127    <server>
128      <id>siteServer</id>
129      <privateKey>/path/to/private/key</privateKey>
130      <passphrase>optional; leave empty if not used.</passphrase>
131    </server>
132    -->
133
134    <server>
135       <id>nexus</id>
136       <username>admin</username>
137       <password>123456</password>
138     </server>
139
140
141  </servers>
142
```

Step 4:

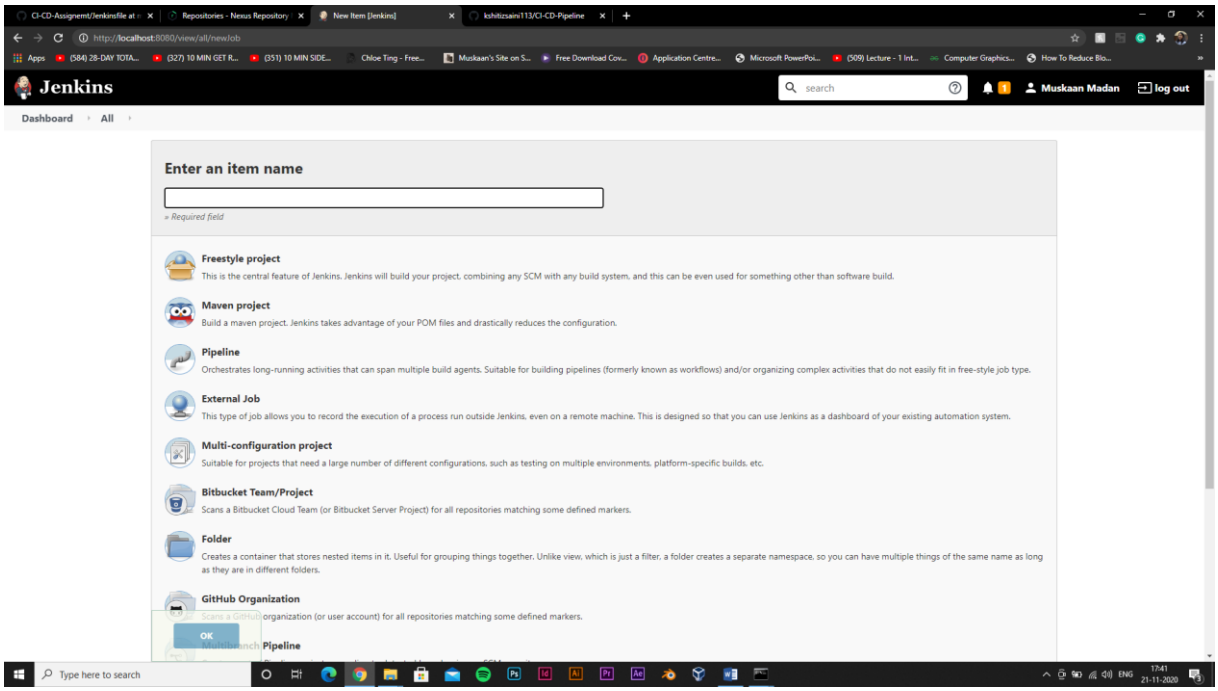
Open the location in terminal where Jenkins is installed and run the following command: java -jar jenkins.war



Log in into Jenkins dashboard



Click on new Item to create a new pipeline

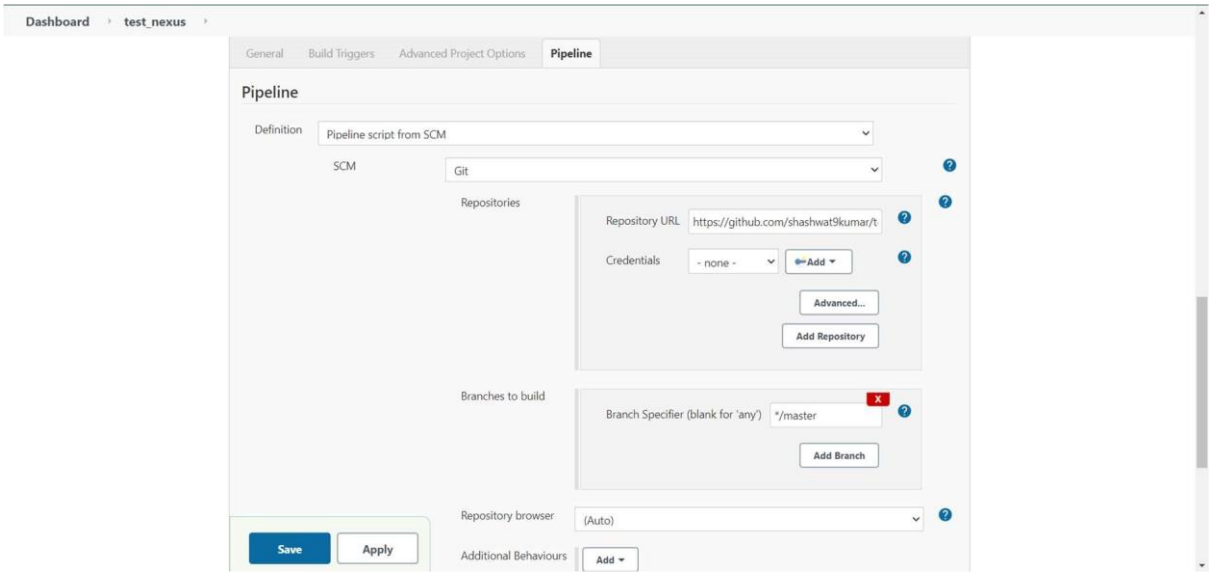


On the page that opens head over to ‘Pipeline’ section and select ‘Pipeline Script from SCM’

In the SCM, select ‘Git’

Provide the URL of the GitHub repository where your project is hosted

Click on ‘Save’ and ‘Apply’



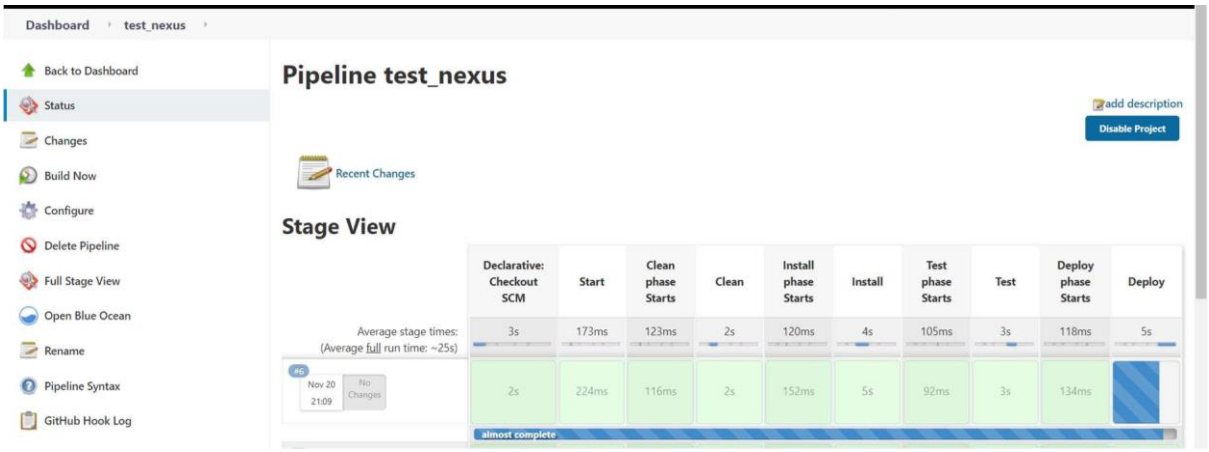
The GitHub repository must contain a Jenkins file that contains “mvn deploy” stage. This would help us deploy the artifact on Nexus server.

Step 5:

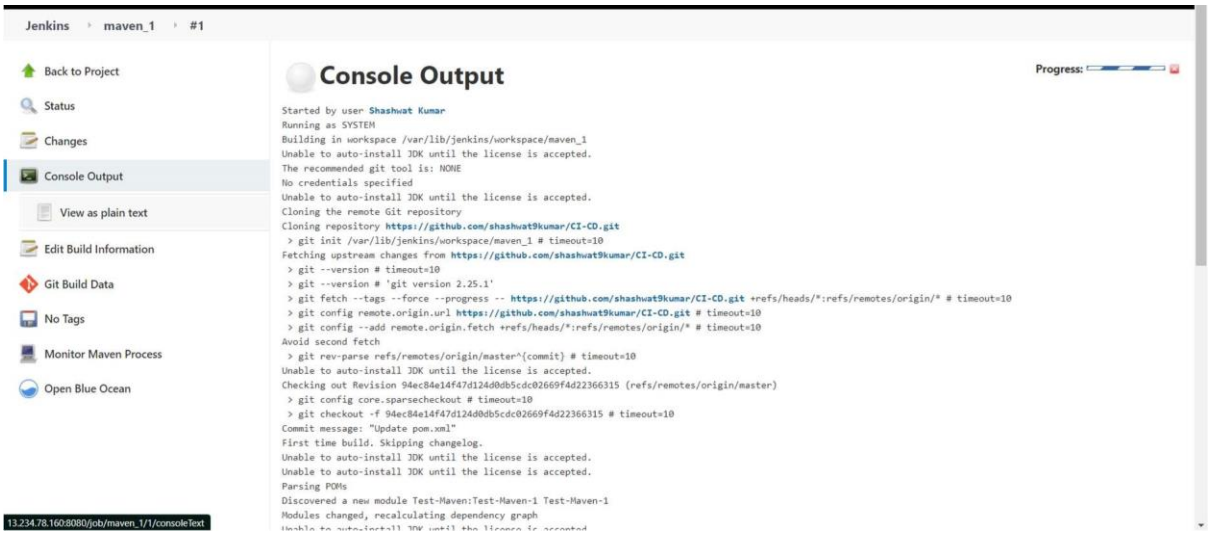
In the screen that pops up, click on Build Now option to build the project.

The project starts to build and run

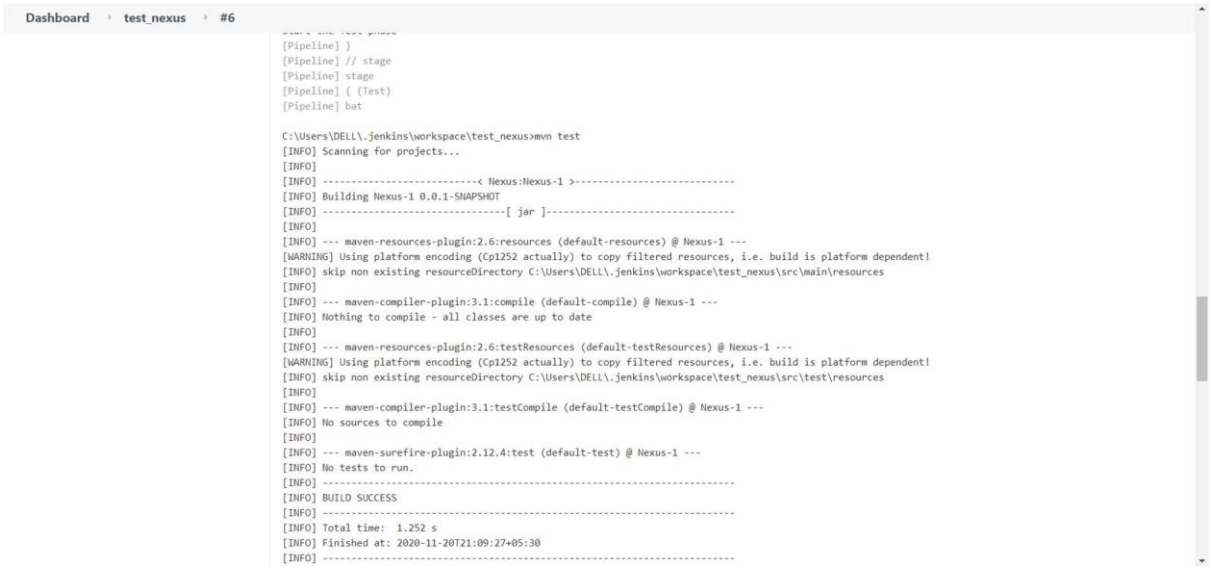
The project pipeline can be seen on the screen depicting the stages defined in the Jenkinsfile



Select the console output option to check how the build is running



If the build runs successfully it would return “BUILD SUCCESS”. It would return failure in case of any errors in the project



Browse the Nexus repository on the running nexus server now to see the deployed jar file