Lab 7. Tomcat installation and configuration

Objectives

- Install and Configure Tomcat on local VM
- Verify Tomcat Working
- Configure Maven to be used in Jenkins
- Install Maven Plugin to Create Maven Jobs
- Pull the code from GITHUB and deploy it to Tomcat Application Server (Servlet Container). Configure Jenkins to use Tomcat.
- Configure Post Build Job to Deploy App on Tomcat

Prerequisites

- Some Basic Knowledge on Jenkins and Tomcat
- Working Setup of Jenkins
- Maven Working as Installed in Lab 2
- Administration Privileges on Jenkins and Tomcat

Sequence 1. Install and Configure Tomcat

1. Install Tomcat and other packages on Oracle Linux 7 with following command:

   # yum -y install tomcat tomcat-admin-webapps tomcat-webapps jakarta-taglibs-standard

   This will install Tomcat and its dependencies, including:

   - The Tomcat Web Admin Manager (tomcat-admin-webapps)
   - The official Tomcat documentation and Default Home Page (tomcat-webapps)

2. Change Tomcat Port so that it does not conflict with Jenkins. Say, 8081

   # vi /etc/tomcat/server.xml

```
<!-- A "Connector" represents an endpoint by which requests are received
     and responses are returned. Documentation at :
     Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
     Java AJP  Connector: /docs/config/ajp.html
     APR (HTTP/AJP) Connector: /docs/apr.html
     Define a non-SSL HTTP/1.1 Connector on port 8080
-->
<Connector port="8081" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
<!-- A "Connector" using the shared thread pool-->
<!--
<Connector executor="tomcatThreadPool"
           port="8081" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
-->
```

3. In order to use Tomcat's web management interface, you will need to create a user. Open the tomcat-users.xml file with the command and scroll down to below the line, which reads ***<tomcat-users>***, and add the information for your user account as given below.

   # vi /usr/share/tomcat/conf/tomcat-users.xml

   <tomcat-users>

   <user username="tomcatmanager" password="En4EW25eI0" roles="manager-gui"/>

   <user username="deployer" password="En4EW25eI0" roles="manager-script"/>

```
-->
<tomcat-users>
<user username="tomcatmanager" password="En4EW25eI0" roles="manager-gui"/>
<user username="deployer" password="En4EW25eI0" roles="manager-script"/>

<!--
  NOTE:  By default, no user is included in the "manager-gui" role required
  to operate the "/manager/html" web application.  If you wish to use this app,
  you must define such a user - the username and password are arbitrary. It is
  strongly recommended that you do NOT use one of the users in the commented out
```

The ***deployer*** account would be used to deploy the WAR file over http. The ***manager-gui*** based ***tomcatmanager*** user would be used to manage the manager web application at http://10.10.0.100:8081/manager

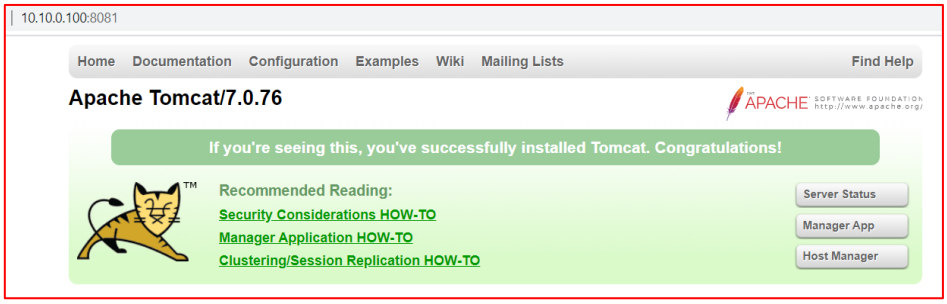
4. Open ports in Firewall

   # firewall-cmd --zone=public --permanent --add-port=8081/tcp
   # firewall-cmd --reload

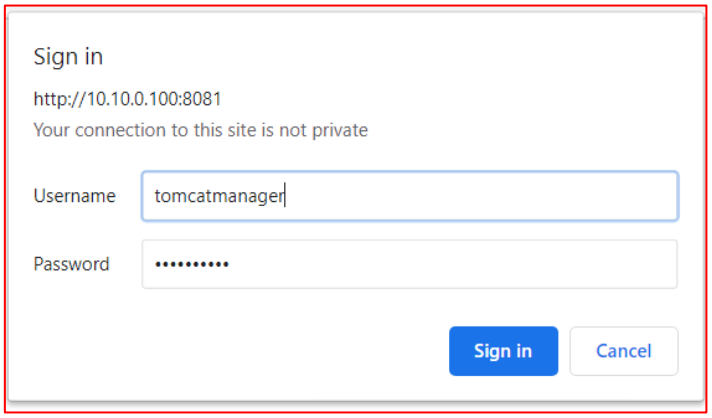
5. Start Tomcat and enable Tomcat to automatically start if the server is rebooted:

   # systemctl --now enable tomcat

6. In a browser, visit the URL http://10.10.0.100:8081 to see the Tomcat welcome page.



7. Click the Manager App link. It will ask you Credentials. Enter User ID and Password as created in previous step.

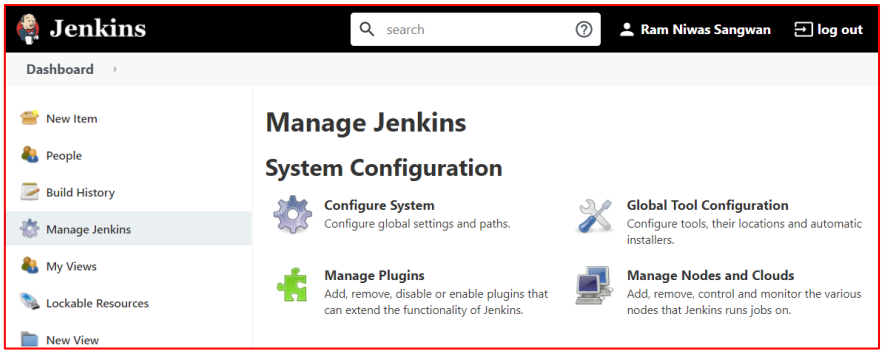


8. Now we are ready with the Tomcat Servlet Container aka Application server and it is ready to be connected from Jenkins.

Sequence 2. Jenkins Configuration Install/Verify Git and Maven

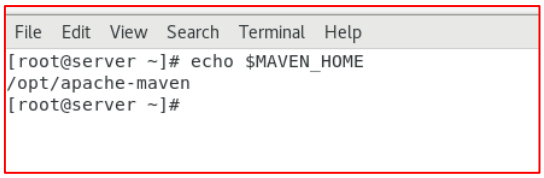
1. Log in to Jenkins as admin user.

2. Go to ***Manage Jenkins*** -> ***Global Tool Configuration*** Section of Jenkins.



3. Git and Maven are already installed. Check MAVEN_HOME with

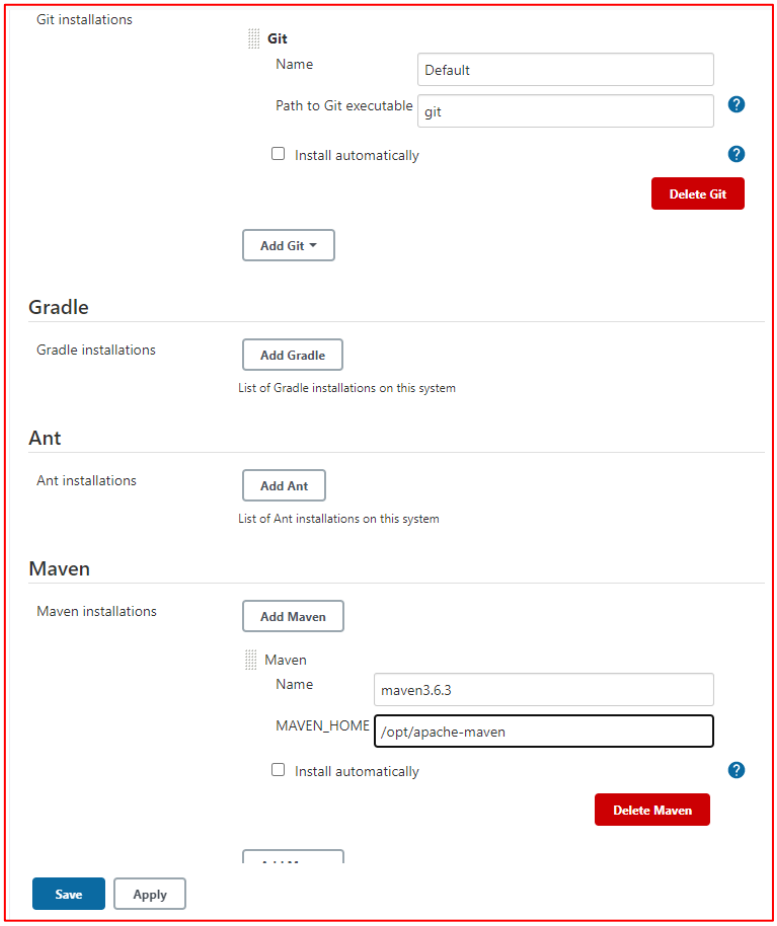
   # echo $MAVEN_HOME



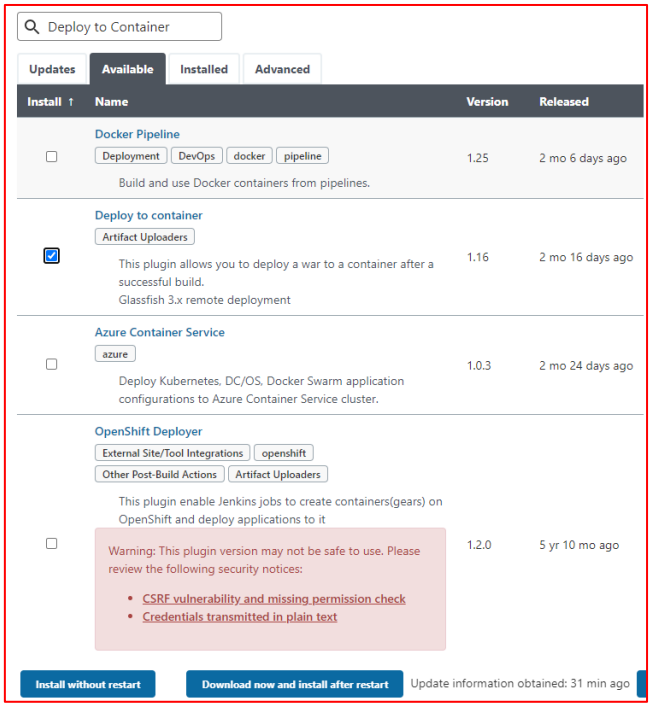
4. Change the permissions on apache-maven directory so that it is accessible by jenkins

   # chmod 777 /opt/apache-maven/

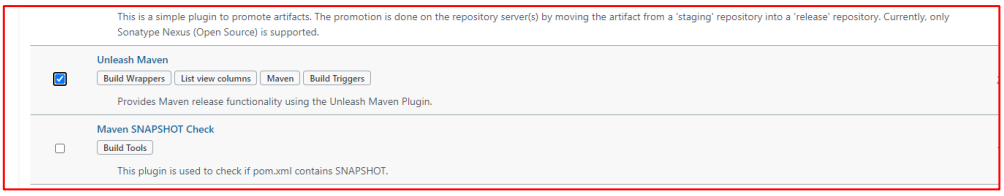
5. Use the Same in the Next Screen and Click Apply and Save.



6. Now **Install Deploy to Container Plugin**. Go to ***Manage Jenkins*** -> ***Manage Plugins*** -> ***Available*** -> ***Deploy to Container Plugin***



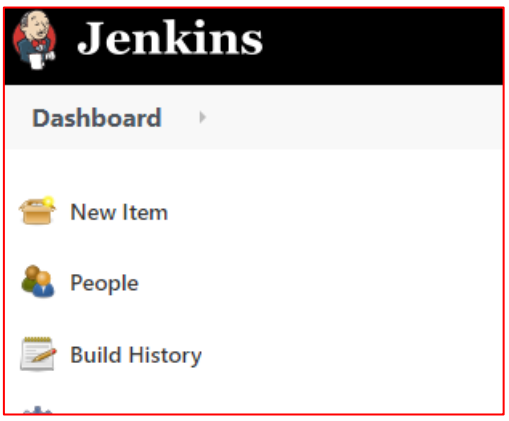
7. Similarly, search for "Maven plugin" and you will get search result as "Unleash Maven Plugin", enable the check-box, click on "Download now and install after restart"



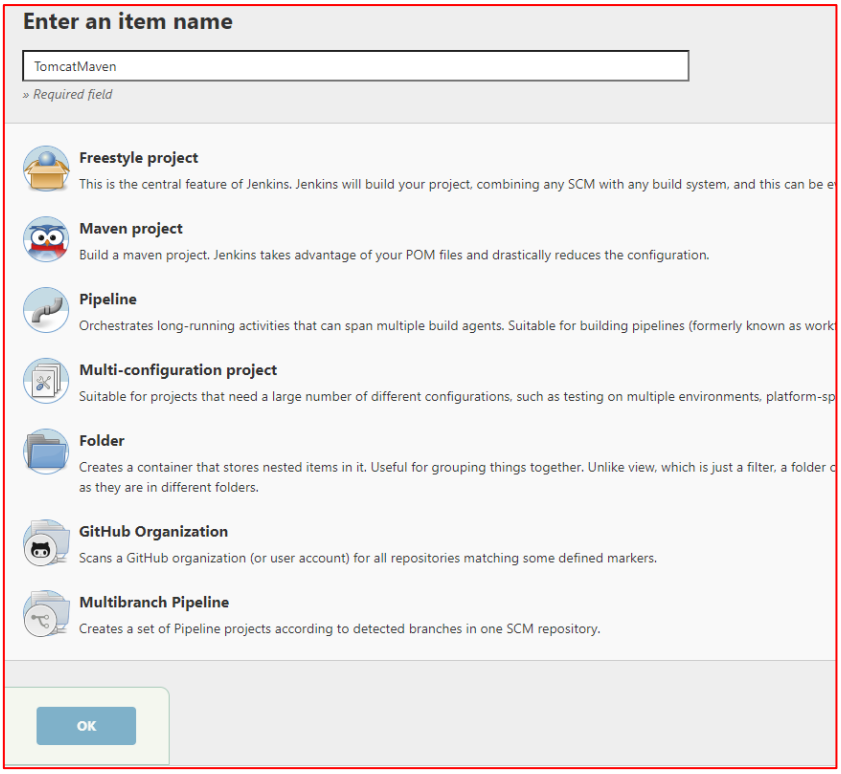
This will help us to select a Maven Job as our Choice.

Sequence 3. Create and Configure a Maven Job with Github.

1. From Menu on left select New Item -> Maven Project



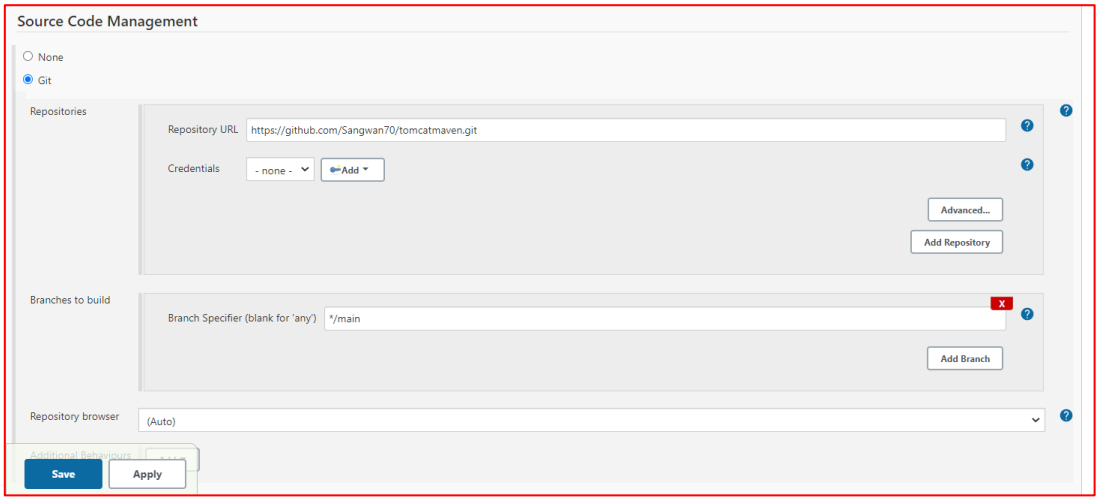
2. Enter Name of your Project “TomcatMaven” and Click OK



3. In the Configuration Section, Under Source Code Management Fill your Github Repository URL. For testing, use one of our Sample Application Named Tomcat Maven App from our Github public Repository.

   [You can use this GITHUB Repository]

   For private repositories, click on Add button displayed near the Credentials drop-down and enter the username and password of your SCM Repo and once it is saved, it would be available on the Dropdown for you to select. If using above Git, Change the Branch Specifier to "***/main***"

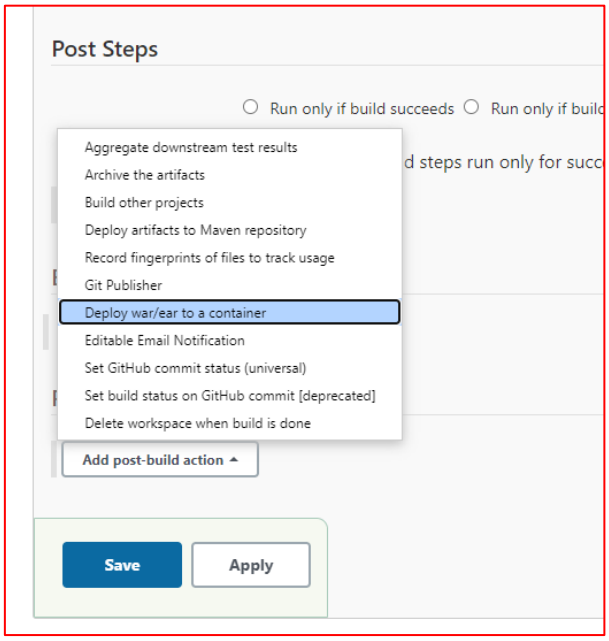


Sequence 4. Configure the Post-build Action and Specify the Tomcat Server Details

1. Drag to the bottom and Go to the ***Post-build Actions*** section

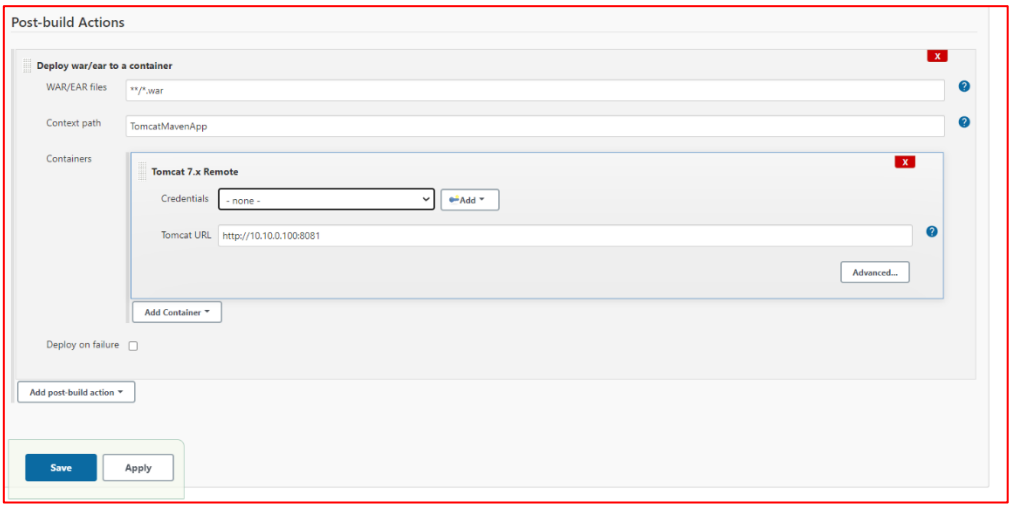
2. Click on ***Add post-build action*** button

3. On the available options click on the Deploy war/ear to container



4. Fill the required parameters for the plugin. After entering these details, click on Add Icon in the screen shot to add tomcat Credentials.

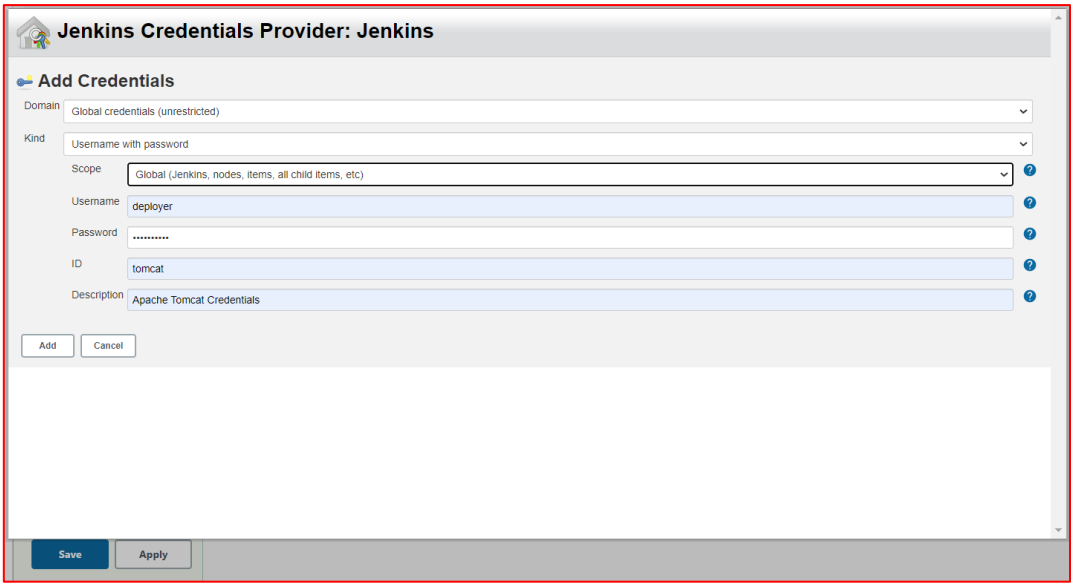
   - WAR/EAR files: **/*.war
   - Context Path: TomcatMavenApp
   - Tomcat URL: http://10.10.0.100:8081



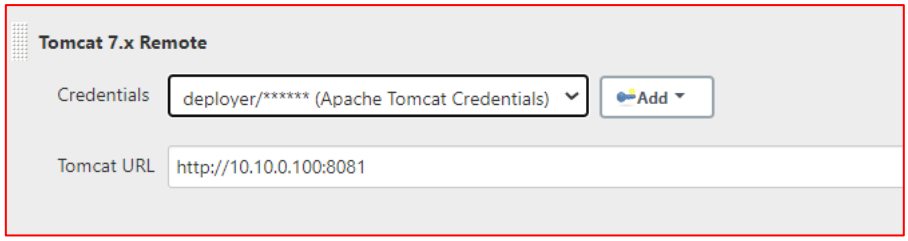
5. Enter the details as given in the screen shot.

   Username: deployer
   Password : *En4EW25eI0*
   ID: tomcat

Description: Apache Tomcat Credentials

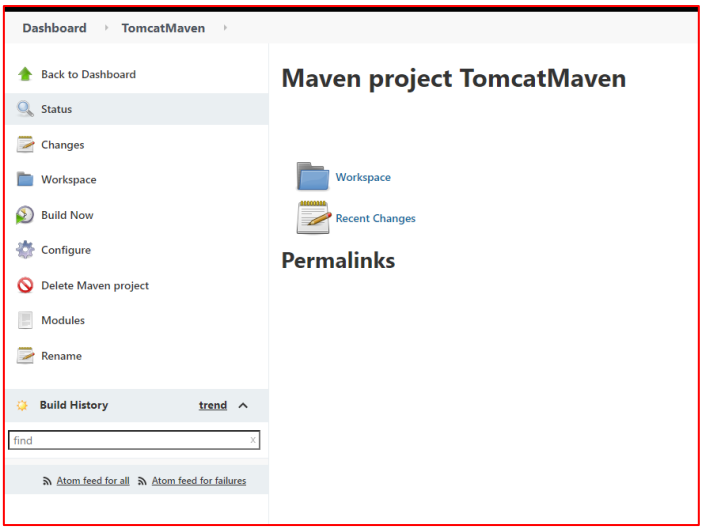


6. Click on Add Button and select the credentials as given below.



7. Finally Click Apply and Save.

8. Execute the Job you have created by clicking on the **Build Now** button



9. Click on Console Output after the Successful build. At the last line you can see that the WAR file has been generated and deployed on the remote server. in our case, http://10.10.0.100:8081/

```
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  4.905 s
[INFO] Finished at: 2021-01-18T12:10:37-05:00
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/lib/jenkins/workspace/TomcatMaven/pom.xml to com.sarav/TomcatMavenApp/2.0/TomcatMavenApp-2.0.pom
[JENKINS] Archiving /var/lib/jenkins/workspace/TomcatMaven/target/TomcatMavenApp-2.0.war to com.sarav/TomcatMavenApp/2.0/TomcatMavenApp-2.0.war
channel stopped
[DeployPublisher][INFO] Attempting to deploy 1 war file(s)
[DeployPublisher][INFO] Deploying /var/lib/jenkins/workspace/TomcatMaven/target/TomcatMavenApp-2.0.war to container Tomcat 7.x Remote with context TomcatMavenApp
  [/var/lib/jenkins/workspace/TomcatMaven/target/TomcatMavenApp-2.0.war] is not deployed. Doing a fresh deployment.
  Deploying [/var/lib/jenkins/workspace/TomcatMaven/target/TomcatMavenApp-2.0.war]
Finished: SUCCESS
```

Sequence 5. Test the Application.

1. As the deployment is completed and the Jenkins Job ran Successfully without issues, let us test our application. In our case, the URL should be

   http://10.10.0.100:8081/TomcatMavenApp

2. You have successfully configured Tomcat with Jenkins Continuous Deployment