PARSHVANATH CHARITABLE TRUST'S

A. P. SHAH INSTITUTE OF TECHNOLOGY

Department of Information Technology

(NBA Accredited)

Academic Year: 2025-26 **Semester: V**

Class / Branch: TEIT

Subject: DevOps Lab

Name of Instructor: Prof. Sujata Oak

Experiment No. 5

Aim: To create and build a CI/CD pipeline in Jenkins to test and deploy an application over the tomcat server.

Theory:

Continuous Integration (CI) and Continuous Deployment (CD) are essential practices in modern software engineering. **Jenkins**, an open-source automation server, is widely used to implement CI/CD pipelines. A **pipeline** in Jenkins defines the automated sequence of tasks needed to build, test, and deploy an application.
Jenkins pipelines are configured to automate the deployment of a **web application** onto an **Apache Tomcat server**.

Jenkins Pipelines

A Jenkins **Pipeline** is a suite of plugins that supports integrating and implementing continuous delivery. Pipelines can be defined as:

- **Declarative Pipeline** – uses a simple, predefined syntax.
- **Scripted Pipeline** – more flexible, written in Groovy scripts.

The pipeline describes **stages** such as:

1. **Checkout Code** (from GitHub or another repository)
2. **Build** (compile code, package into .war)
3. **Test** (unit tests, integration tests)
4. **Deploy** (copy artifact to Tomcat server)

Apache Tomcat Server

Apache Tomcat is an open-source Java Servlet container that runs Java-based web applications packaged as .war (Web Application Archive) files. In deployment pipelines:

- Tomcat provides the runtime environment for the application.
- Jenkins deploys the .war file automatically to Tomcat's webapps directory.

Jenkins–Tomcat Integration

To automate deployment:

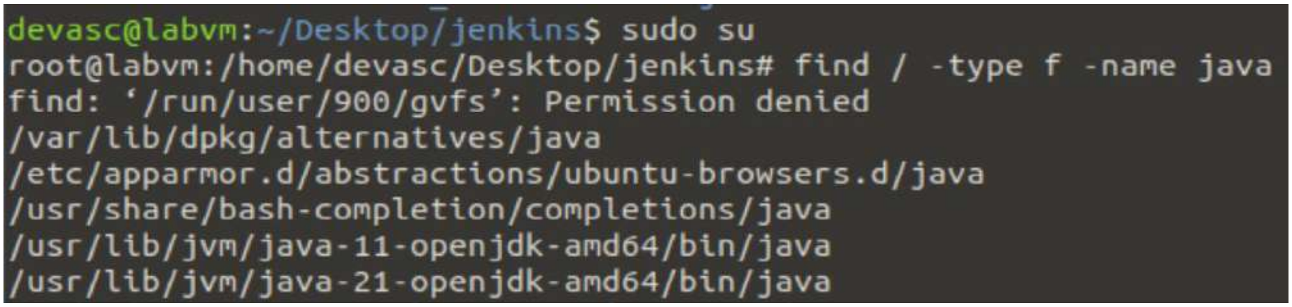
- Jenkins is configured with **credentials** to connect to the Tomcat server.
- Deployment is done using methods like:
  - **Jenkins Deploy to Container Plugin** (direct WAR deployment).
  - **scp/rsync** to copy .war files into Tomcat's webapps.
  - **Pipeline Groovy scripts** for custom deployment logic.

IMPLEMENTATION: CONFIGURING JENKINS PIPELINE

Once Jenkins is installed follow the below steps:

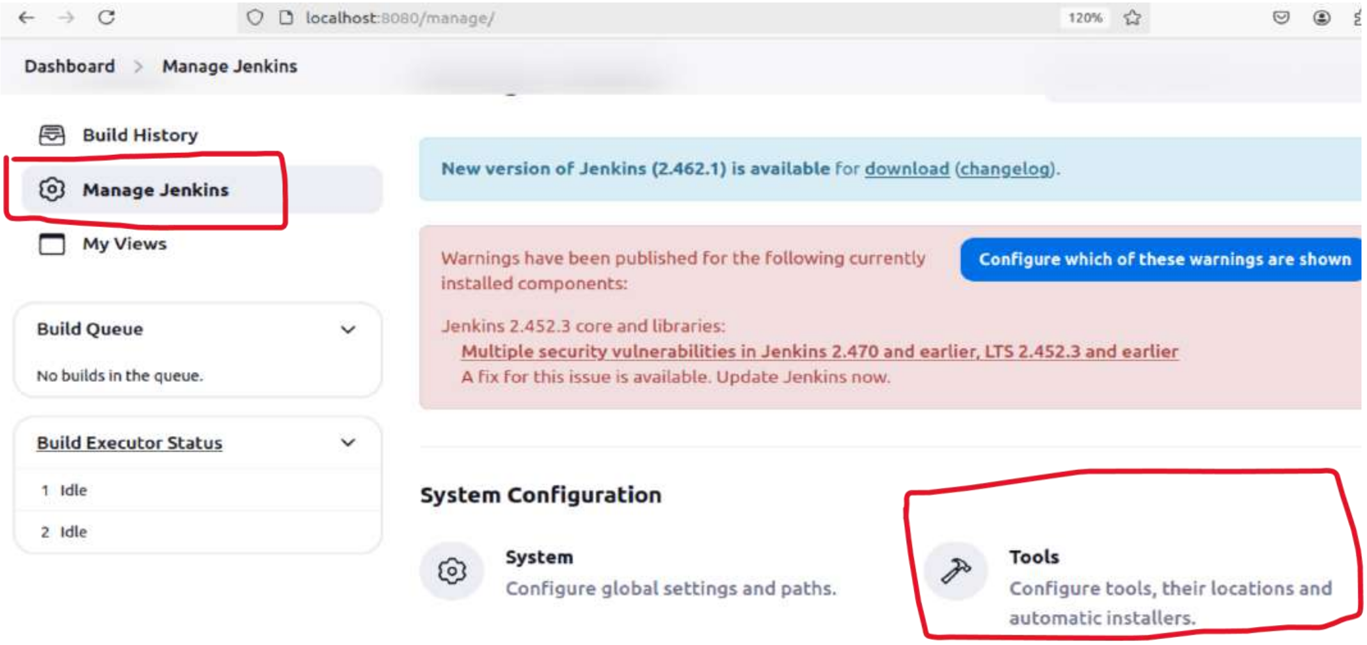
Step1: Now, we need to specify the Java location to Jenkins. Go back to your server command prompt and use the code below to fetch the directory of Java. Multiple directories will be listed using the below code. In our case, the directory is: '/usr/lib/jvm/java-11-openjdk-amd64/bin/java'.

find / -type f -name java



```
devasc@labvm:~/Desktop/jenkins$ sudo su
root@labvm:/home/devasc/Desktop/jenkins# find / -type f -name java
find: '/run/user/900/gvfs': Permission denied
/var/lib/dpkg/alternatives/java
/etc/apparmor.d/abstractions/ubuntu-browsers.d/java
/usr/share/bash-completion/completions/java
/usr/lib/jvm/java-11-openjdk-amd64/bin/java
/usr/lib/jvm/java-21-openjdk-amd64/bin/java
```

Step 2) Copy the above location and go back to your Jenkins Dashboard. Look for Global Tool Configuration under the Manage Jenkins menu, as shown in the image below.



No builds in the queue.

New version of Jenkins (2.462.1) is available for download (changelog).

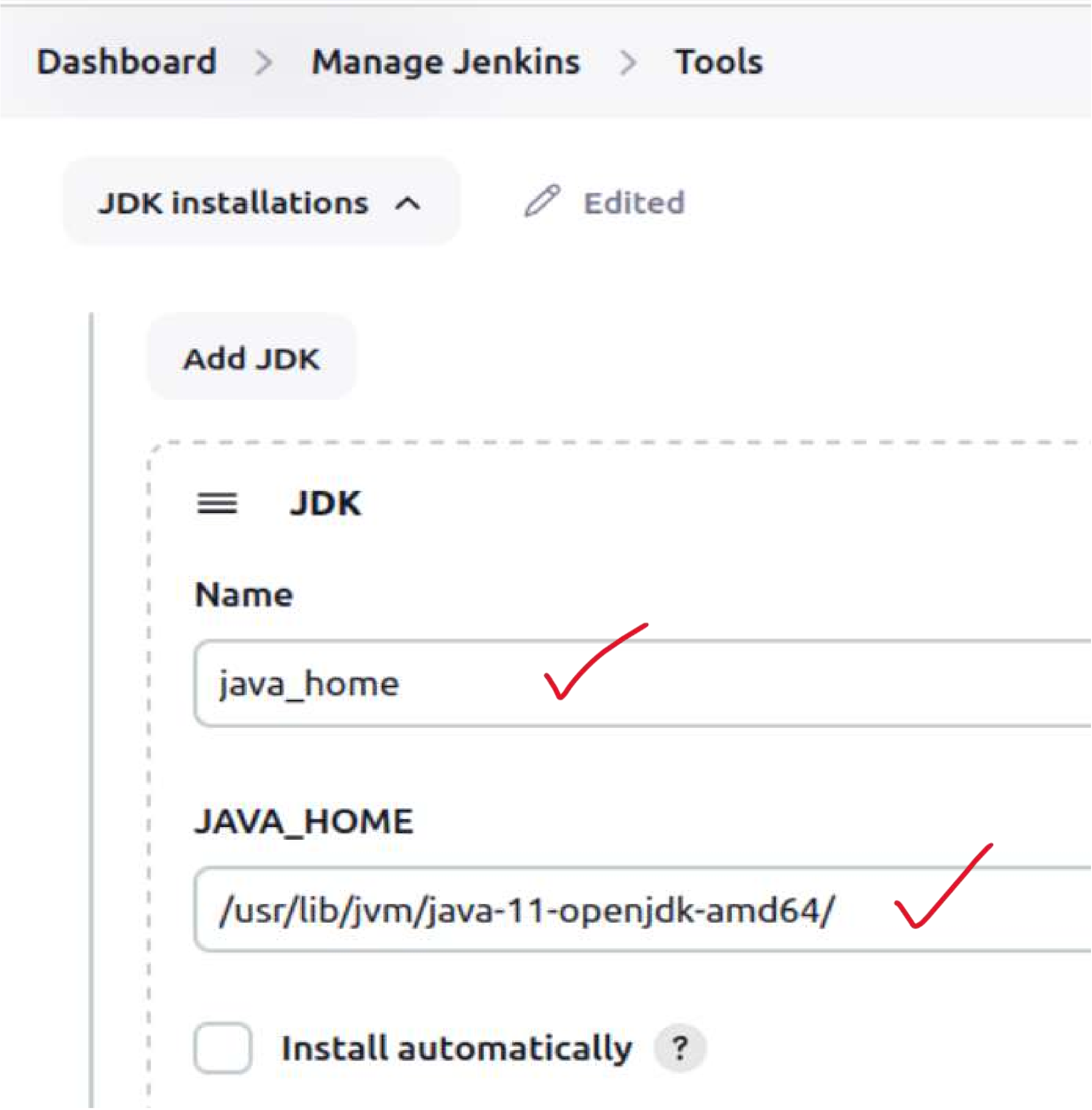
Warnings have been published for the following currently installed components:

Jenkins 2.452.3 core and libraries:
Multiple security vulnerabilities in Jenkins 2.470 and earlier, LTS 2.452.3 and earlier
A fix for this issue is available. Update Jenkins now.

Configure global settings and paths.

Configure tools, their locations and automatic installers.

Step 3) Unselect the Install automatically button from the JDK window and fill the fields. Paste the java location path and trim it as shown in the image below. In the below Git window, select the install automatically checkbox.



Dashboard > Manage Jenkins > Tools

JDK installations ^ Edited

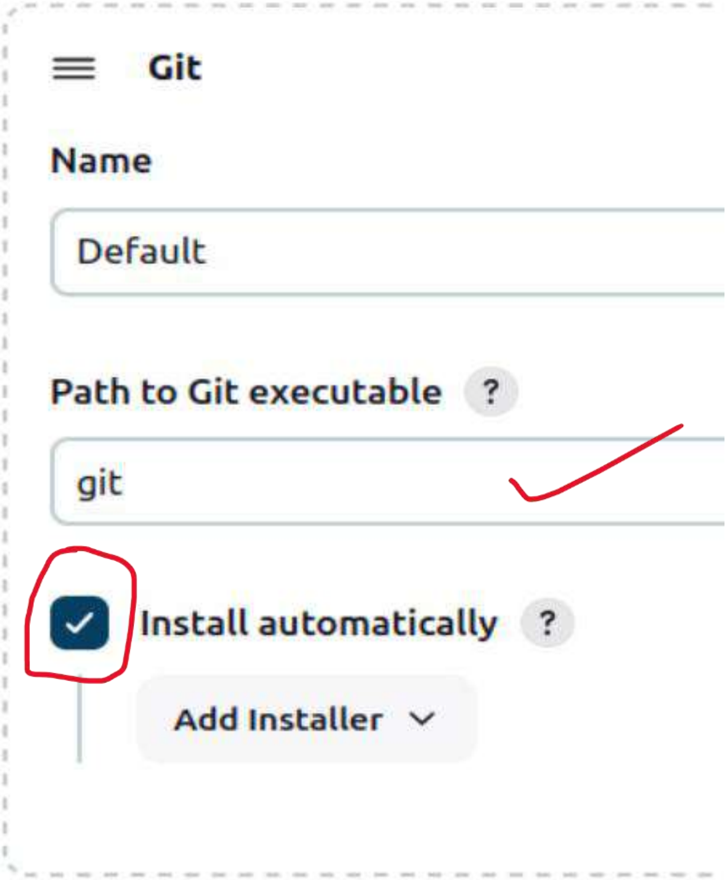
Add JDK

JDK

Name: java_home

JAVA_HOME: /usr/lib/jvm/java-11-openjdk-amd64/

□ Install automatically ?



≡ Git

Name

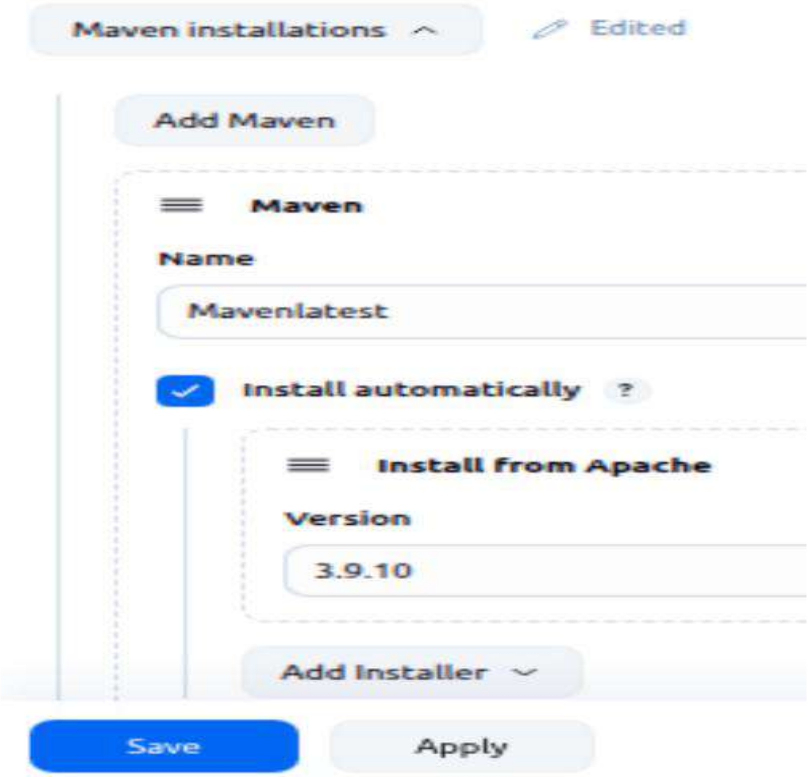
Default

Path to Git executable ?

git

☑ Install automatically ?

Step 4) Similarly, check the box for Maven and fill the name field. Save all the settings, and now the configuration of Jenkins is completed.



Name: Mavenlatest

☑ Install automatically ?

Install from Apache

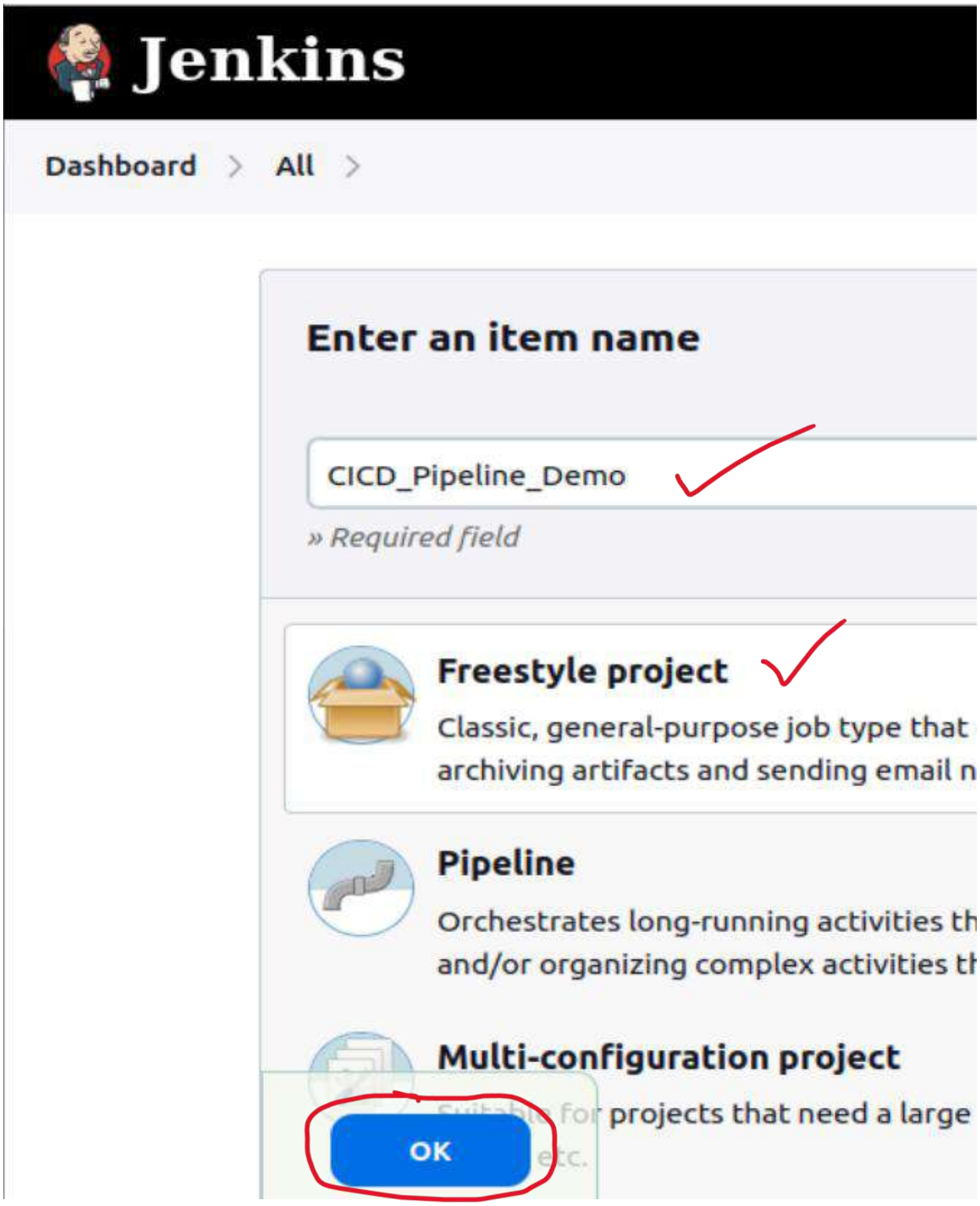
Version: 3.9.10

Create CI/CD Pipeline Jenkins

We can now start creating pipelines using Jenkins after all the configuration and setup.

Continuous Integration

Step 1) Create New Item, select Freestyle Project and provide a name to your item.



Jenkins
Dashboard > All >
Enter an item name

CICD_Pipeline_Demo ✓

» Required field
Freestyle project ✓
Classic, general-purpose job type that
archiving artifacts and sending email n
Pipeline
Orchestrates long-running activities th
and/or organizing complex activities th
Multi-configuration project
Suitable for projects that need a large
etc.
OK

Step 2) Switch to the Source Code Management window and paste your Github repository link. Specify your branch name of the repository below and Save it.

[Note: The above-linked Github repository ‘*https://github.com/sujataoak799/hello-world2025.git* ‘ contains a ‘*pom.xml*‘ file used for Java compilation and generates a web app. It will be deployed to the server]

The **pom.xml** file is used in **Apache Maven** (a popular Java build automation tool).

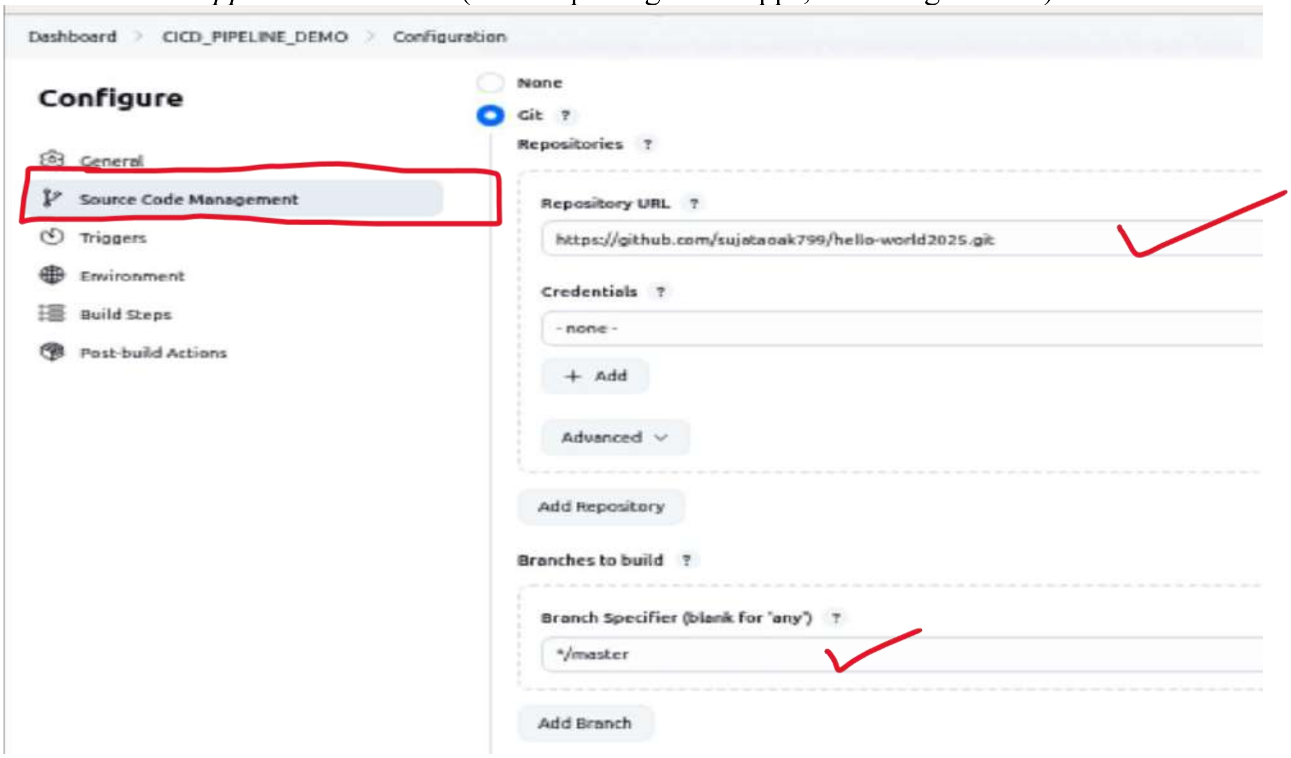
- **POM → Project Object Model**
- **pom.xml** → An XML file that contains the configuration and metadata about the project.

It usually defines:

- Project coordinates (groupId, artifactId, version)
- Dependencies (other libraries required)
- Build plugins and goals
- Project info (name, description, URL, licenses, developers, etc.)

In short:
pom.xml = the **blueprint** for how Maven builds and manages your Java project.

- **WAR** → *Web Application Archive* (used to package web apps, including Jenkins)



Dashboard > CICD_PIPELINE_DEMO > Configuration

Configure

○ None
● Git ?

Repositories ?

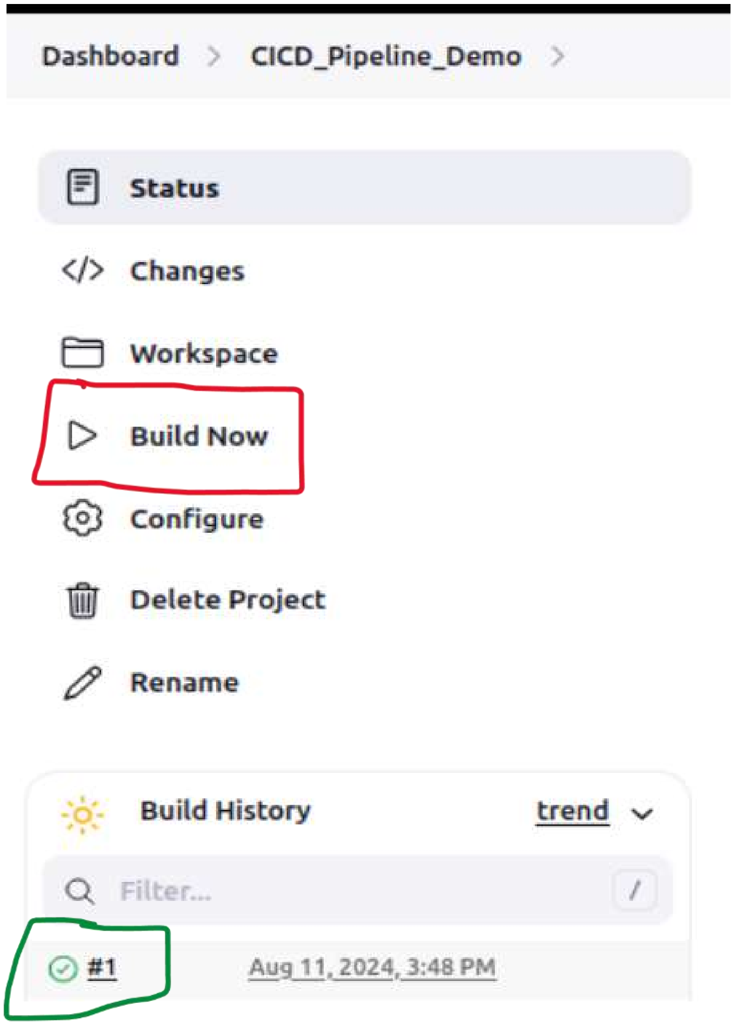
Repository URL ?
https://github.com/sujataoak799/hello-world2025.git

Credentials ?
- none -

Branches to build ?

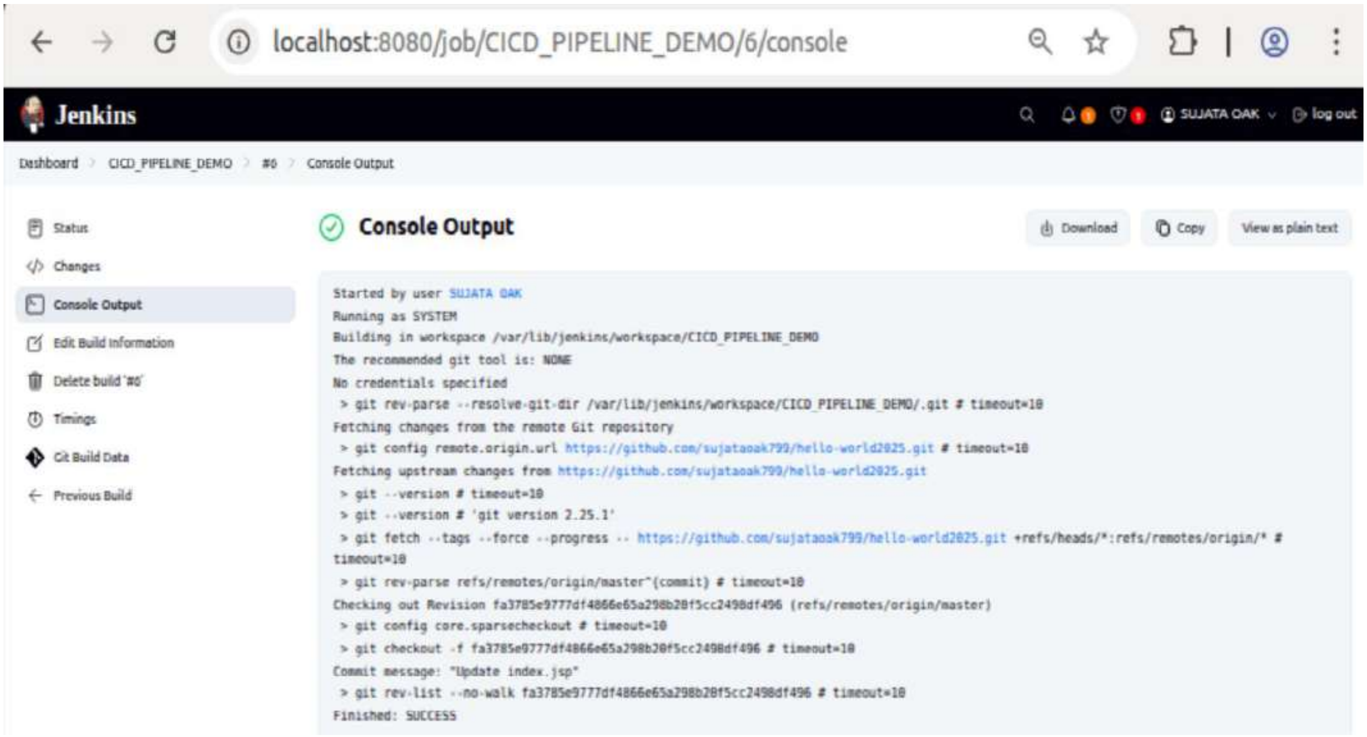
Branch Specifier (blank for 'any') ?
*/master

Step 3) Now click on Build Now button from the menu. With this step, all the repository files will be fetched by Jenkins. Click on Configure to go back to the same settings page.



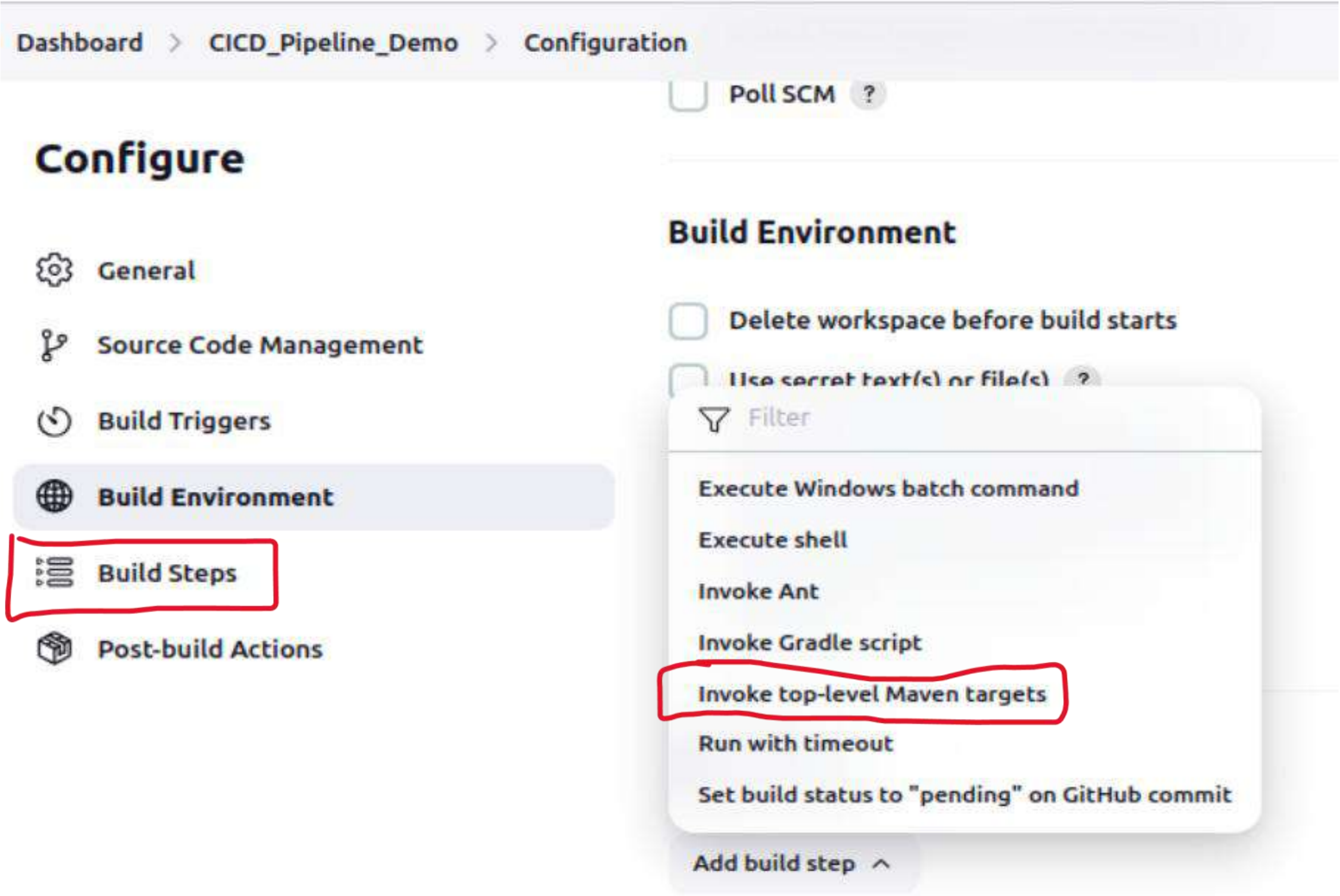
Dashboard > CICD_Pipeline_Demo >

✓ #1 Aug 11, 2024, 3:48 PM



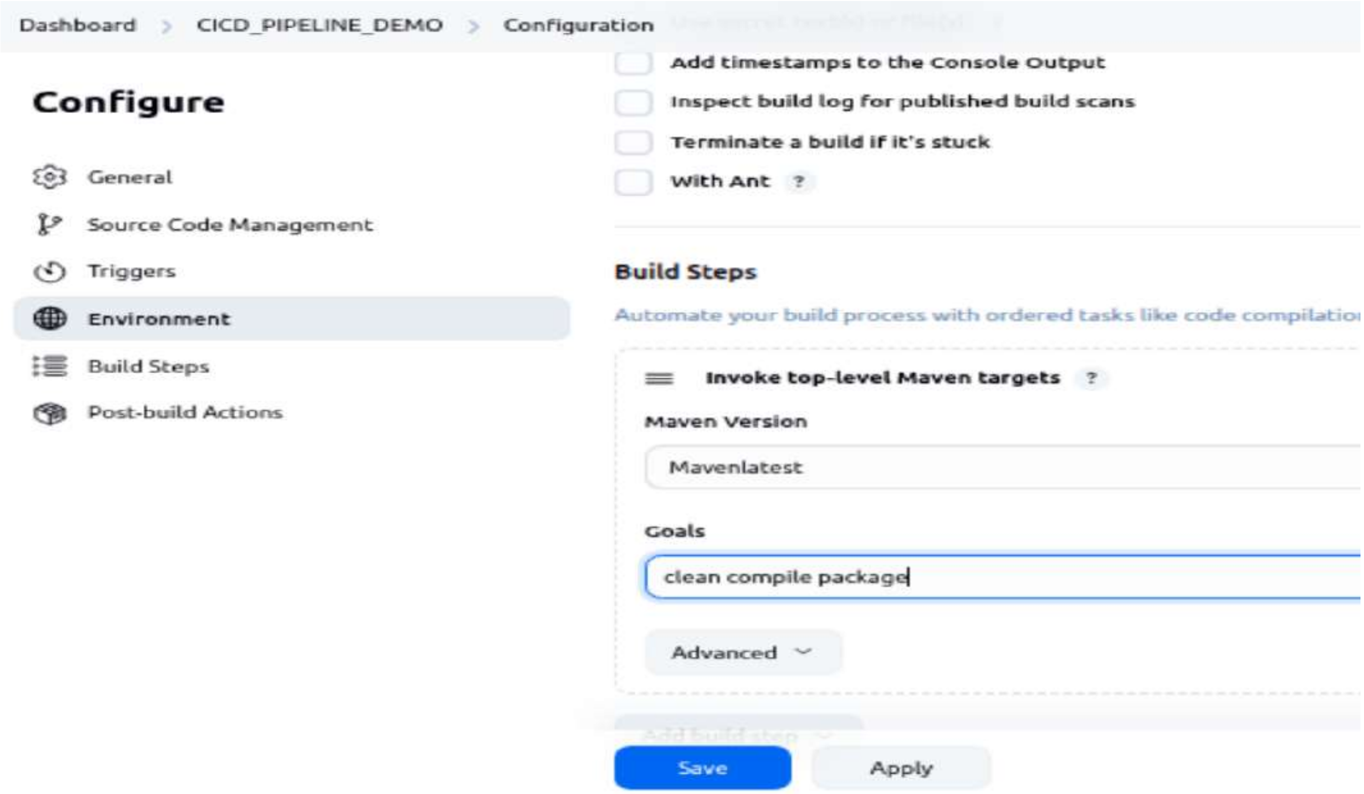
```
Started by user SUJATA OAK
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/CICD_PIPELINE_DEMO
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/CICD_PIPELINE_DEMO/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/sujataoak799/hello-world2025.git # timeout=10
Fetching upstream changes from https://github.com/sujataoak799/hello-world2025.git
 > git --version # timeout=10
 > git --version # 'git version 2.25.1'
 > git fetch --tags --force --progress -- https://github.com/sujataoak799/hello-world2025.git +refs/heads/*:refs/remotes/origin/* #
timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision fa3785e9777df4866e65a298b20f5cc2498df496 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f fa3785e9777df4866e65a298b20f5cc2498df496 # timeout=10
Commit message: "Update index.jsp"
 > git rev-list --no-walk fa3785e9777df4866e65a298b20f5cc2498df496 # timeout=10
Finished: SUCCESS
```

Step 4) Click on Build Tab and select build step as '*Invoke top-level Maven targets*'.



Dashboard > CICD_Pipeline_Demo > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
□ Poll SCM ?
Build Environment
□ Delete workspace before build starts
□ Use secret text(s) or file(s) ?
Filter
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit
Add build step ^

Step 5) **Select your maven name from the drop-down menu. Fill the goals with the multiple jobs you need to perform and separate them with one space. These goals are available in your repository, and you need to invoke them using Maven. Save it and again click on the '*Build Now*' button from the menu as we did in the previous steps. Now the maven commands will be executed that will generate a war file.**



☐ Add timestamps to the Console Output

☐ Inspect build log for published build scans

☐ Terminate a build if it's stuck

☐ With Ant ?

Build Steps

Automate your build process with ordered tasks like code compilatio

Invoke top-level Maven targets ?

Maven Version

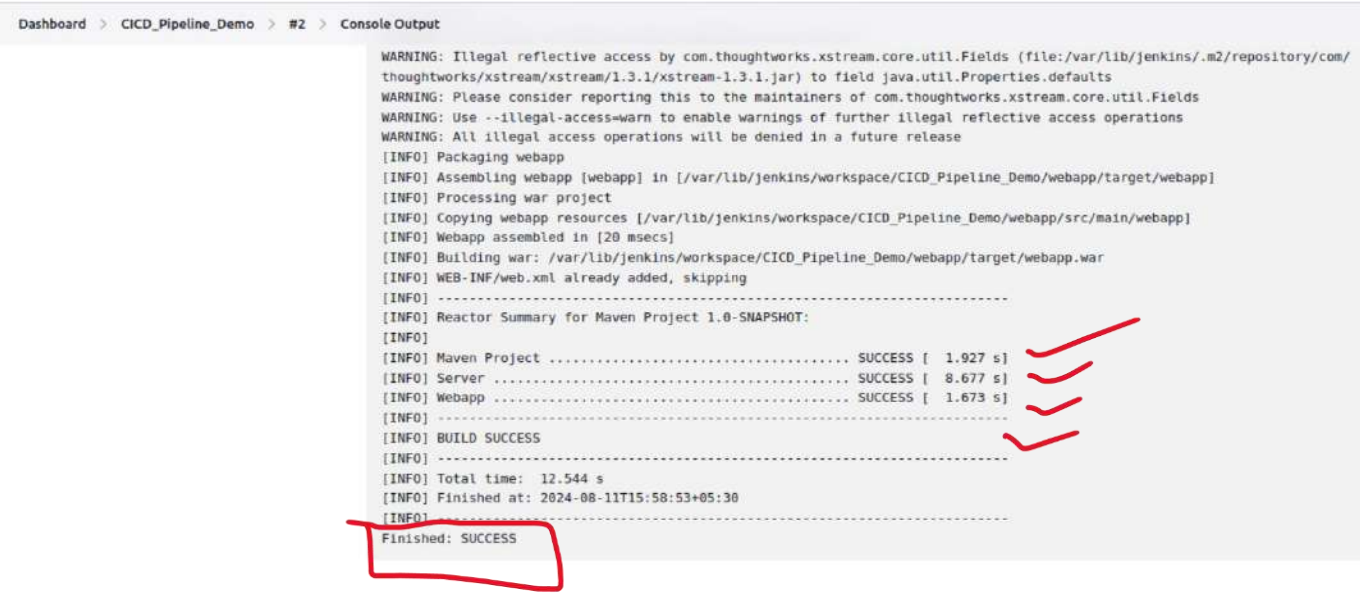
Mavenlatest

Goals

clean compile package

Advanced ⌄

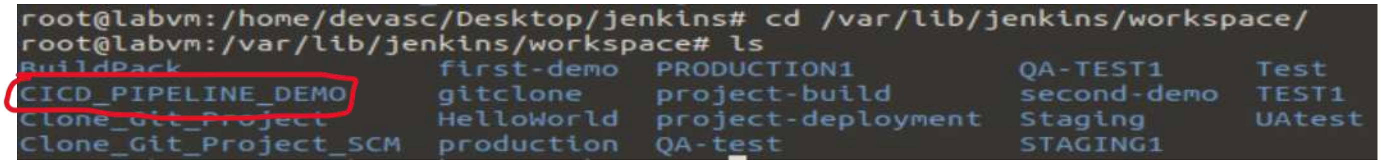
Click-On Build Now:



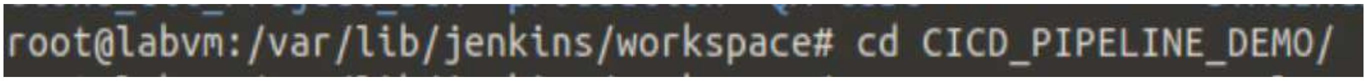
```
WARNING: Illegal reflective access by com.thoughtworks.xstream.core.util.Fields (file:/var/lib/jenkins/.m2/repository/com/
thoughtworks/xstream/xstream/1.3.1/xstream-1.3.1.jar) to field java.util.Properties.defaults
WARNING: Please consider reporting this to the maintainers of com.thoughtworks.xstream.core.util.Fields
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
[INFO] Packaging webapp
[INFO] Assembling webapp [webapp] in [/var/lib/jenkins/workspace/CICD_Pipeline_Demo/webapp/target/webapp]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/CICD_Pipeline_Demo/webapp/src/main/webapp]
[INFO] Webapp assembled in [20 msecs]
[INFO] Building war: /var/lib/jenkins/workspace/CICD_Pipeline_Demo/webapp/target/webapp.war
[INFO] WEB-INF/web.xml already added, skipping
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for Maven Project 1.0-SNAPSHOT:
[INFO]
[INFO] Maven Project ...................................... SUCCESS [  1.927 s]
[INFO] Server ............................................. SUCCESS [  8.677 s]
[INFO] Webapp ............................................. SUCCESS [  1.673 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  12.544 s
[INFO] Finished at: 2024-08-11T15:58:53+05:30
[INFO] ------------------------------------------------------------------------
Finished: SUCCESS
```

Step 6) If you want to check the war file created in the previous steps, visit the workspace on your Jenkins dashboard or just run the directory commands in your server. Your directories and project name can vary, so you can use the '*ls*' command to see the list inside that directory and also keep in mind the directory name is case sensitive.

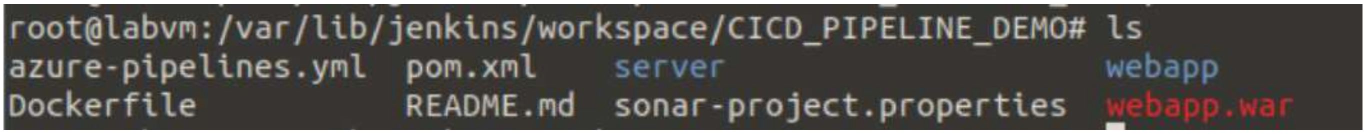
# cd /var/lib/jenkins/workspace/
#ls



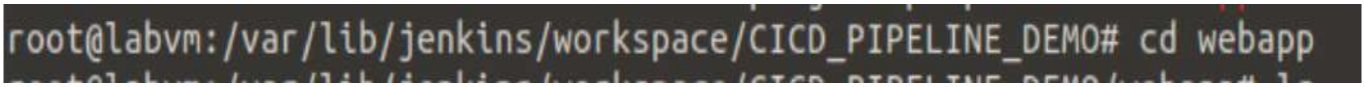
```
root@labvm:/home/devasc/Desktop/jenkins# cd /var/lib/jenkins/workspace/
root@labvm:/var/lib/jenkins/workspace# ls
BuildPack                first-demo   PRODUCTION1          QA-TEST1     Test
CICD_PIPELINE_DEMO       gitclone     project-build        second-demo  TEST1
Clone_Git_Project        HelloWorld   project-deployment   Staging      UAtest
Clone_Git_Project_SCM    production   QA-test              STAGING1
```



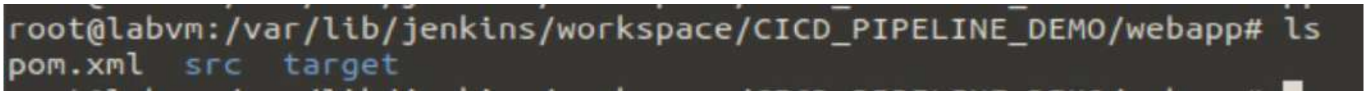
```
root@labvm:/var/lib/jenkins/workspace# cd CICD_PIPELINE_DEMO/
```



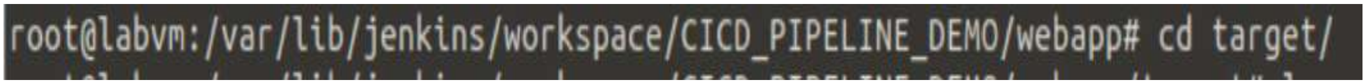
```
root@labvm:/var/lib/jenkins/workspace/CICD_PIPELINE_DEMO# ls
azure-pipelines.yml  pom.xml    server                    webapp
Dockerfile           README.md  sonar-project.properties  webapp.war
```



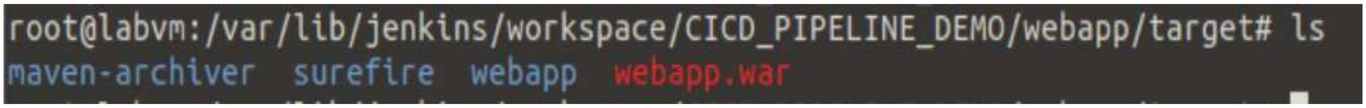
```
root@labvm:/var/lib/jenkins/workspace/CICD_PIPELINE_DEMO# cd webapp
```



```
root@labvm:/var/lib/jenkins/workspace/CICD_PIPELINE_DEMO/webapp# ls
pom.xml  src  target
```



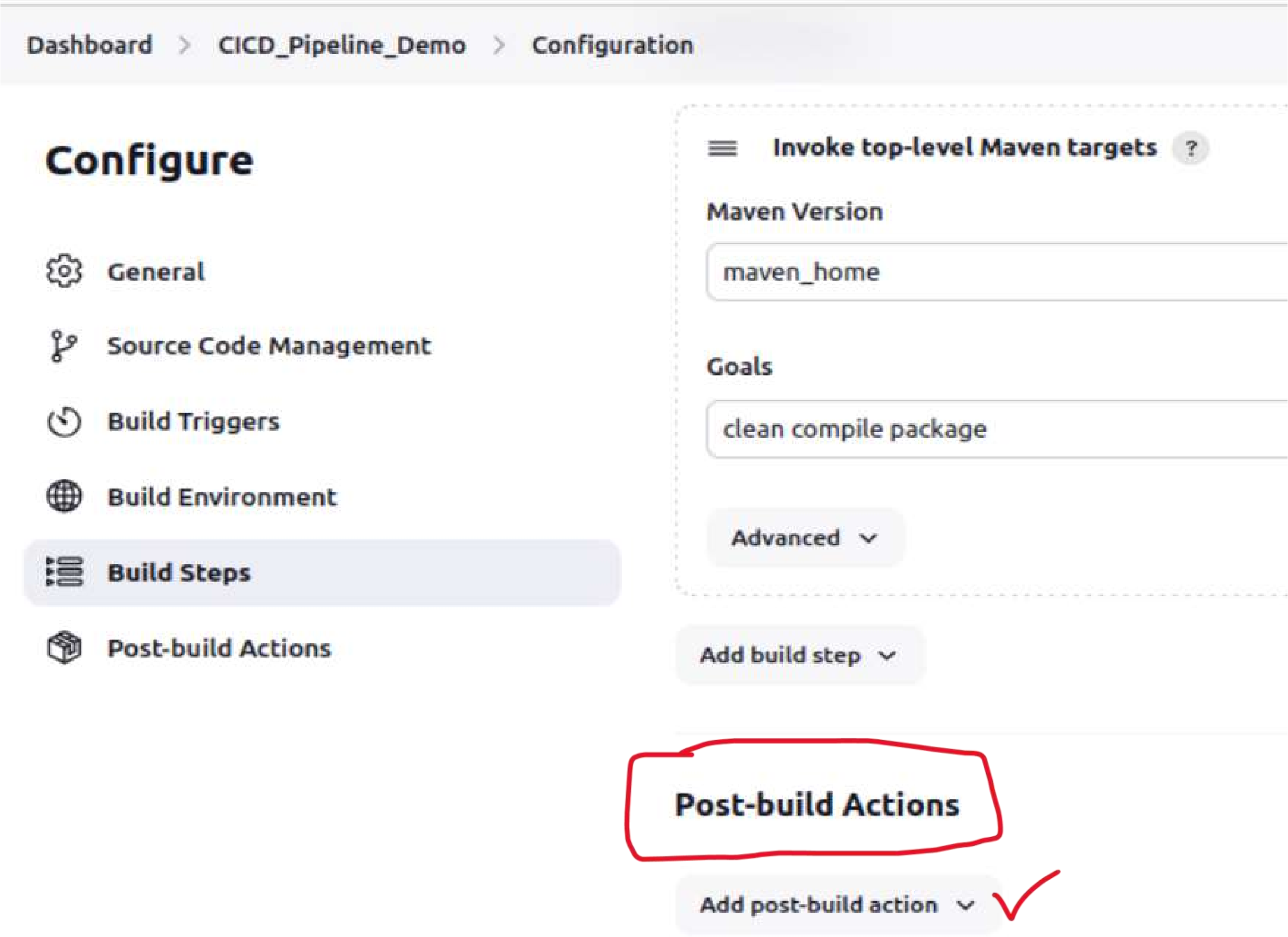
```
root@labvm:/var/lib/jenkins/workspace/CICD_PIPELINE_DEMO/webapp# cd target/
```



```
root@labvm:/var/lib/jenkins/workspace/CICD_PIPELINE_DEMO/webapp/target# ls
maven-archiver  surefire  webapp  webapp.war
```

Step 7) Now go back to Configure and visit the '*Post Build Actions*' tab. Click the drop-down and select '*Archive the Artifacts*' from the options. In the field, write down '/*.war' as shown in the image below. It will fetch all the directories and get the war file wherever it is present. Click again on Build Now button, and you will now see the**

Artifacts in the Jenkins dashboard.



Dashboard > CICD_Pipeline_Demo > Configuration

Configure

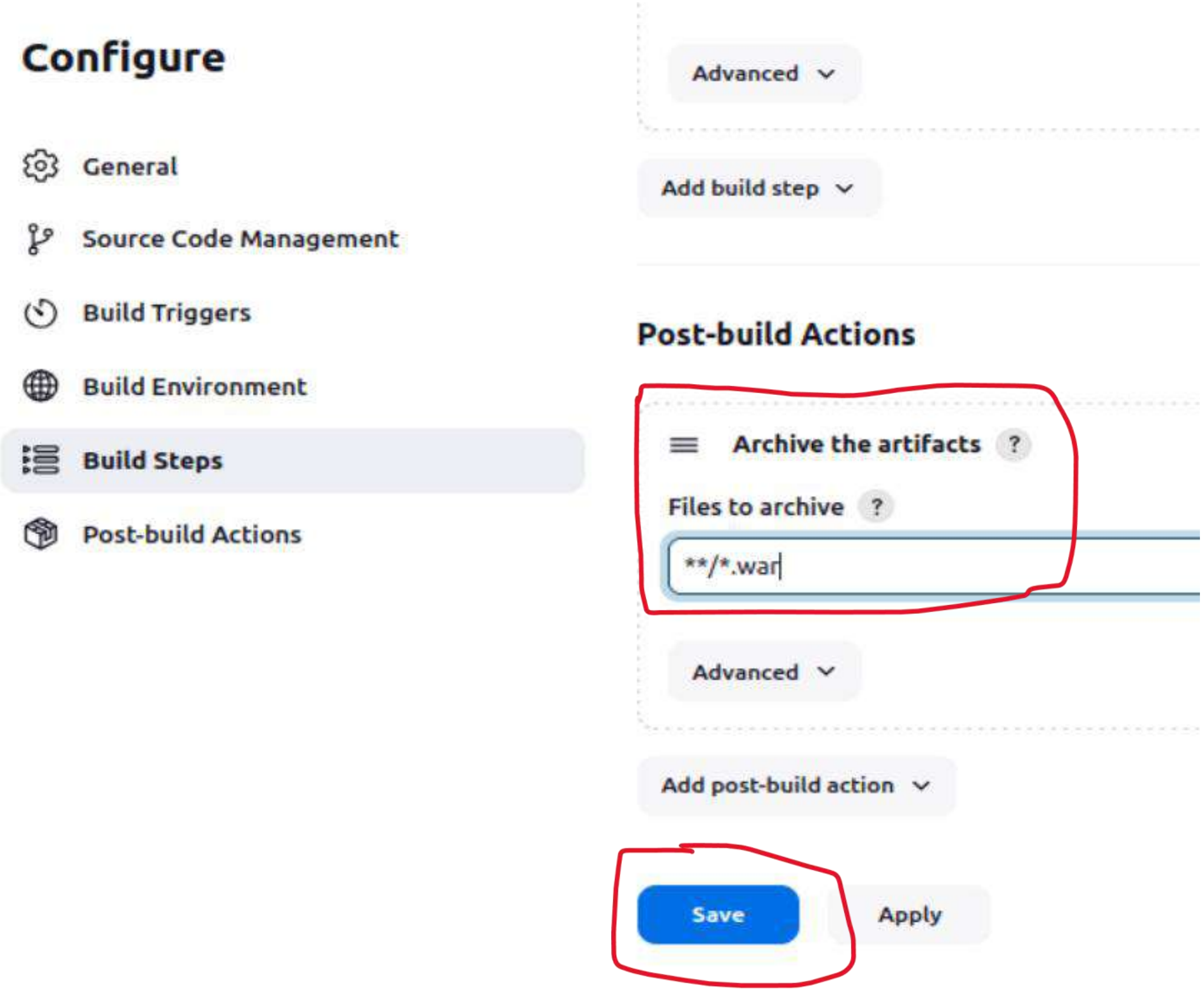
- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Invoke top-level Maven targets

Maven Version: maven_home

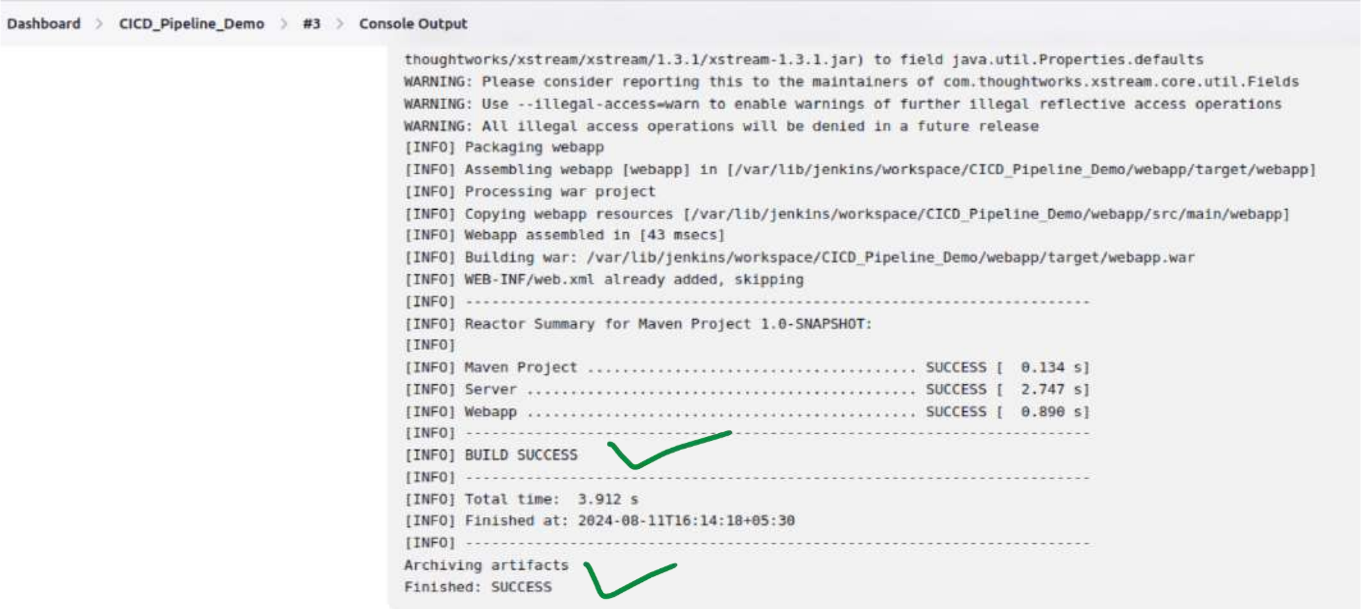
Goals: clean compile package

Post-build Actions



Archive the artifacts

Files to archive: **/*.war



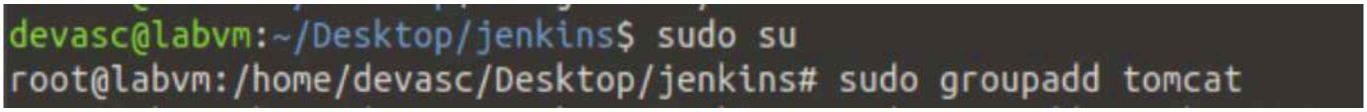
```
thoughtworks/xstream/xstream/1.3.1/xstream-1.3.1.jar) to field java.util.Properties.defaults
WARNING: Please consider reporting this to the maintainers of com.thoughtworks.xstream.core.util.Fields
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
[INFO] Packaging webapp
[INFO] Assembling webapp [webapp] in [/var/lib/jenkins/workspace/CICD_Pipeline_Demo/webapp/target/webapp]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/CICD_Pipeline_Demo/webapp/src/main/webapp]
[INFO] Webapp assembled in [43 msecs]
[INFO] Building war: /var/lib/jenkins/workspace/CICD_Pipeline_Demo/webapp/target/webapp.war
[INFO] WEB-INF/web.xml already added, skipping
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for Maven Project 1.0-SNAPSHOT:
[INFO]
[INFO] Maven Project ...................................... SUCCESS [  0.134 s]
[INFO] Server ............................................. SUCCESS [  2.747 s]
[INFO] Webapp ............................................. SUCCESS [  0.890 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  3.912 s
[INFO] Finished at: 2024-08-11T16:14:18+05:30
[INFO] ------------------------------------------------------------------------
Archiving artifacts
Finished: SUCCESS
```

Continuous Deployment

Step 8) We need to install Apache Tomcat, and for this, you need to visit the Tomcat Download page. In the core section, hover over the ‘*tar.gz’* link and copy it. Now, use the below commands in your server one by one.

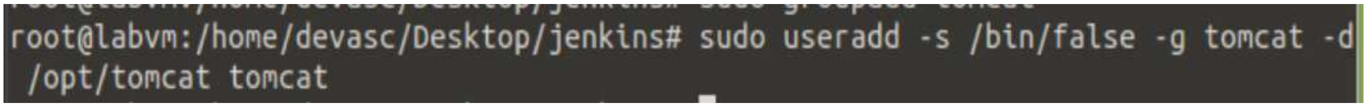
- **First, four commands will create one temporary directory and user group to access the file. Here, use the command curl -O ‘*paste tomcat download link*‘ as shown in the command below.**
- **Use further commands to create a tomcat directory and extract the gzip file. Just cross-check the version number of the Tomcat that you are downloading and extracting.**
- **Now the permission of the files needs to be configured with the below commands. In the last command, replace it with your username by which you are accessing the server.**

sudo groupadd tomcat



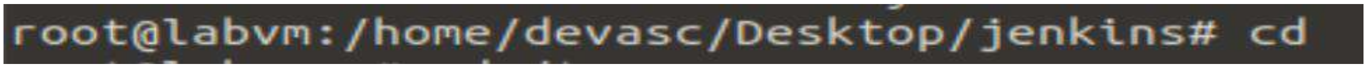
```
devasc@labvm:~/Desktop/jenkins$ sudo su
root@labvm:/home/devasc/Desktop/jenkins# sudo groupadd tomcat
```

sudo useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat



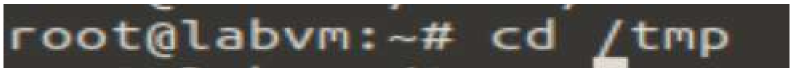
```
root@labvm:/home/devasc/Desktop/jenkins# sudo useradd -s /bin/false -g tomcat -d
 /opt/tomcat tomcat
```

cd



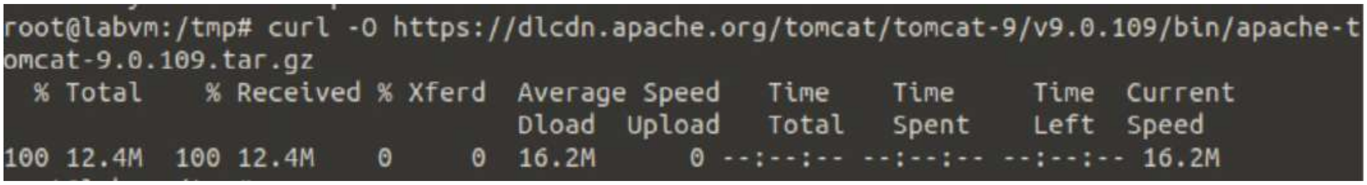
```
root@labvm:/home/devasc/Desktop/jenkins# cd
```

cd /tmp



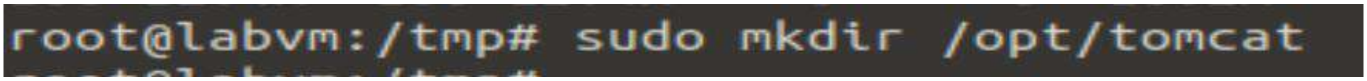
```
root@labvm:~# cd /tmp
```

curl -O https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.109/bin/apache-tomcat-9.0.109.tar.gz



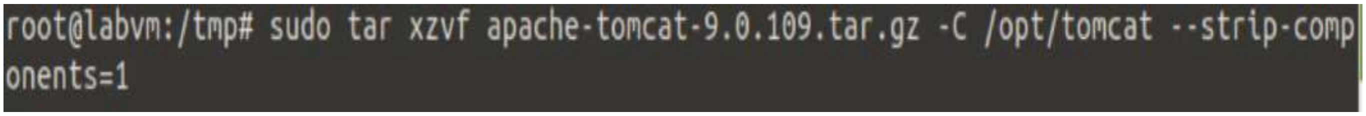
```
root@labvm:/tmp# curl -O https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.109/bin/apache-t
omcat-9.0.109.tar.gz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 12.4M  100 12.4M    0     0  16.2M      0 --:--:-- --:--:-- --:--:-- 16.2M
```

sudo mkdir /opt/tomcat

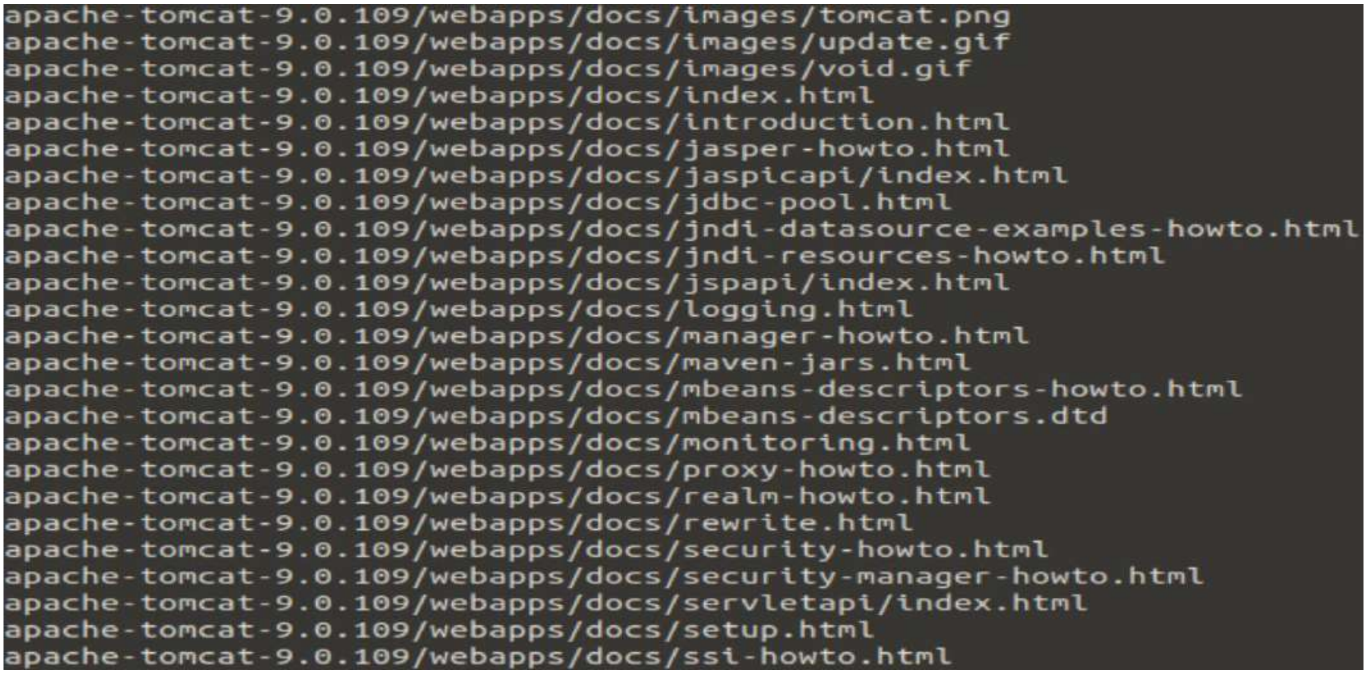


```
root@labvm:/tmp# sudo mkdir /opt/tomcat
```

sudo tar xzvf apache-tomcat-9.0.109.tar.gz -C /opt/tomcat --strip-components=1



```
root@labvm:/tmp# sudo tar xzvf apache-tomcat-9.0.109.tar.gz -C /opt/tomcat --strip-comp
onents=1
```



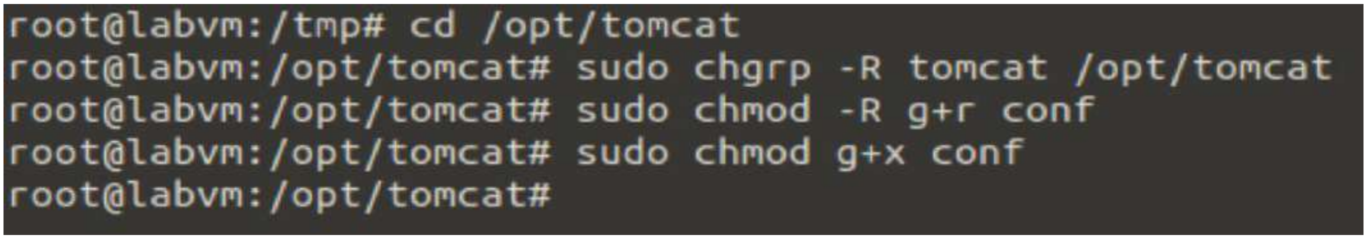
```
apache-tomcat-9.0.109/webapps/docs/images/tomcat.png
apache-tomcat-9.0.109/webapps/docs/images/update.gif
apache-tomcat-9.0.109/webapps/docs/images/void.gif
apache-tomcat-9.0.109/webapps/docs/index.html
apache-tomcat-9.0.109/webapps/docs/introduction.html
apache-tomcat-9.0.109/webapps/docs/jasper-howto.html
apache-tomcat-9.0.109/webapps/docs/jaspicapi/index.html
apache-tomcat-9.0.109/webapps/docs/jdbc-pool.html
apache-tomcat-9.0.109/webapps/docs/jndi-datasource-examples-howto.html
apache-tomcat-9.0.109/webapps/docs/jndi-resources-howto.html
apache-tomcat-9.0.109/webapps/docs/jspapi/index.html
apache-tomcat-9.0.109/webapps/docs/logging.html
apache-tomcat-9.0.109/webapps/docs/manager-howto.html
apache-tomcat-9.0.109/webapps/docs/maven-jars.html
apache-tomcat-9.0.109/webapps/docs/mbeans-descriptors-howto.html
apache-tomcat-9.0.109/webapps/docs/mbeans-descriptors.dtd
apache-tomcat-9.0.109/webapps/docs/monitoring.html
apache-tomcat-9.0.109/webapps/docs/proxy-howto.html
apache-tomcat-9.0.109/webapps/docs/realm-howto.html
apache-tomcat-9.0.109/webapps/docs/rewrite.html
apache-tomcat-9.0.109/webapps/docs/security-howto.html
apache-tomcat-9.0.109/webapps/docs/security-manager-howto.html
apache-tomcat-9.0.109/webapps/docs/servletapi/index.html
apache-tomcat-9.0.109/webapps/docs/setup.html
apache-tomcat-9.0.109/webapps/docs/ssi-howto.html
```

cd /opt/tomcat

sudo chgrp -R tomcat /opt/tomcat

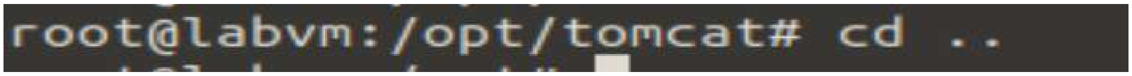
sudo chmod -R g+r conf

sudo chmod g+x conf



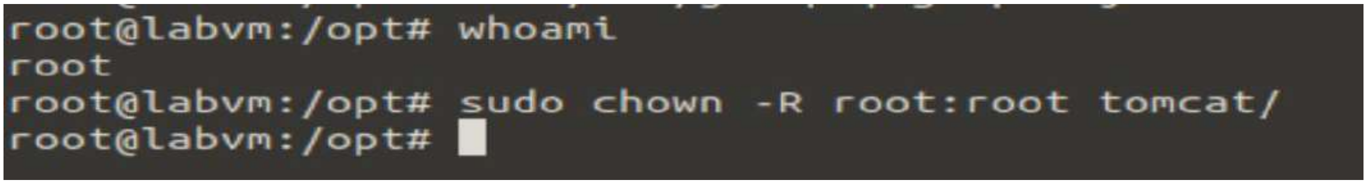
```
root@labvm:/tmp# cd /opt/tomcat
root@labvm:/opt/tomcat# sudo chgrp -R tomcat /opt/tomcat
root@labvm:/opt/tomcat# sudo chmod -R g+r conf
root@labvm:/opt/tomcat# sudo chmod g+x conf
root@labvm:/opt/tomcat#
```

cd ..



```
root@labvm:/opt/tomcat# cd ..
```

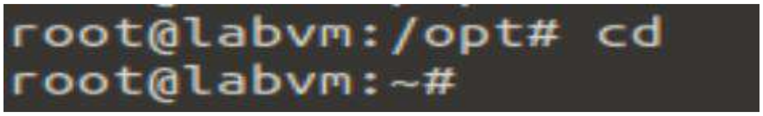
sudo chown -R root:root tomcat/



```
root@labvm:/opt# whoami
root
root@labvm:/opt# sudo chown -R root:root tomcat/
root@labvm:/opt#
```

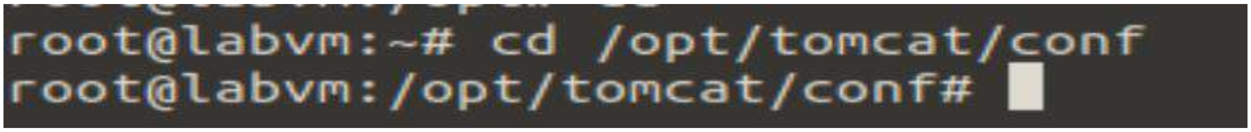
Step 9) We need to update the port number from 8080 to 8090 in the server.xml file. We are updating it as this port number is already in use by Jenkins, and we have created 8090 in Azure VM for Tomcat. Use the below commands to edit the file. When you enter the file, click the INSERT button to edit. Now search for a similar code, as shown in the image below. Update the port number to 8090. To save the file, press the *Esc* key, type *:wq* and click on *Enter* button.

cd



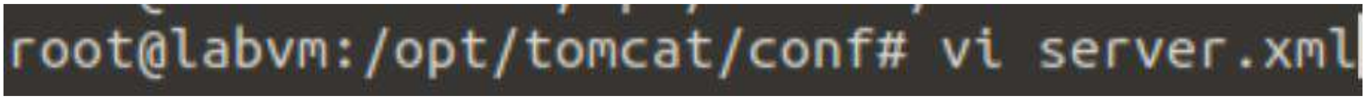
```
root@labvm:/opt# cd
root@labvm:~#
```

cd /opt/tomcat/conf

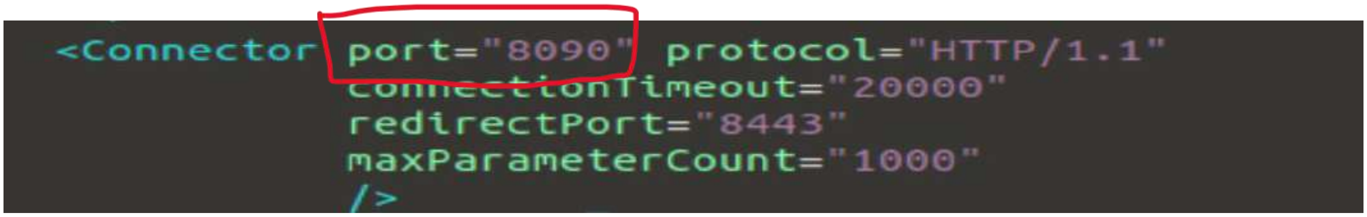


```
root@labvm:~# cd /opt/tomcat/conf
root@labvm:/opt/tomcat/conf#
```

vi server.xml



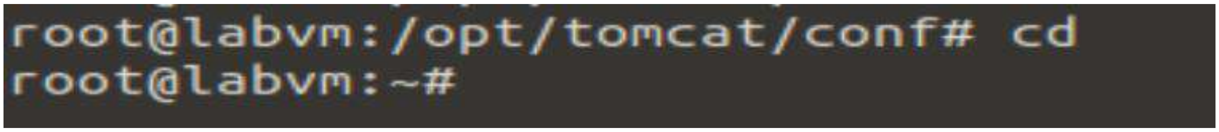
```
root@labvm:/opt/tomcat/conf# vi server.xml
```



```
<Connector port="8090" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
           maxParameterCount="1000"
           />
```

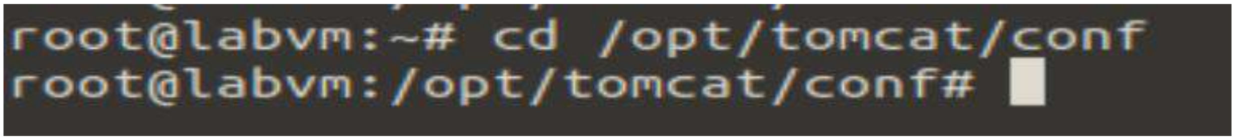
Step 10) Similarly, we need to edit the '*tomcat-users.xml'* file to update the roles that enable us to deploy files using Tomcat. In the file before tomcat-users ending code, paste the below roles code. To save the file press, the *Esc* key, type *:wq* and press *Enter*.

cd



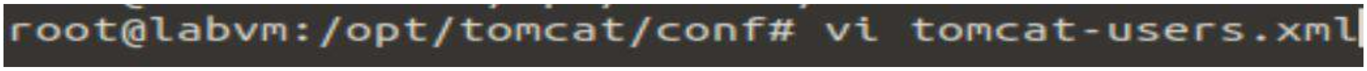
```
root@labvm:/opt/tomcat/conf# cd
root@labvm:~#
```

cd /opt/tomcat/conf



```
root@labvm:~# cd /opt/tomcat/conf
root@labvm:/opt/tomcat/conf#
```

vi tomcat-users.xml



```
root@labvm:/opt/tomcat/conf# vi tomcat-users.xml
```

<role rolename="manager-gui"/>

<role rolename="manager-script"/>

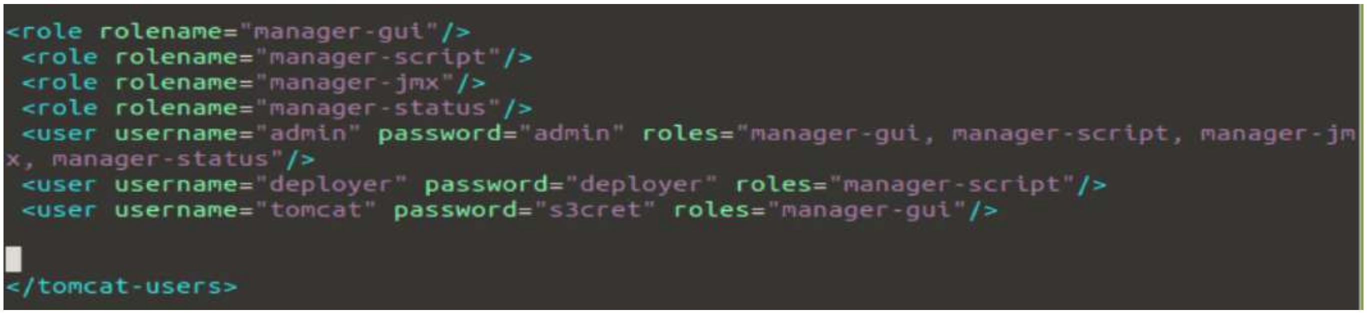
<role rolename="manager-jmx"/>

<role rolename="manager-status"/>

<user username="admin" password="admin" roles="manager-gui, manager-script, manager-jmx, manager-status"/>

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

<user username="tomcat" password="s3cret" roles="manager-gui"/>

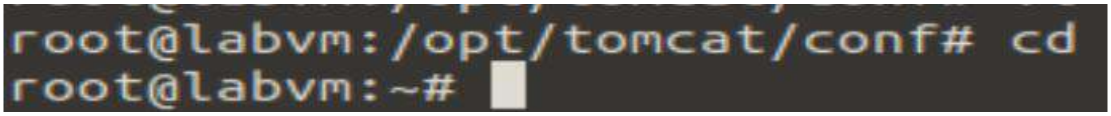


```
<role rolename="manager-gui"/>
 <role rolename="manager-script"/>
 <role rolename="manager-jmx"/>
 <role rolename="manager-status"/>
 <user username="admin" password="admin" roles="manager-gui, manager-script, manager-jm
x, manager-status"/>
 <user username="deployer" password="deployer" roles="manager-script"/>
 <user username="tomcat" password="s3cret" roles="manager-gui"/>

</tomcat-users>
```

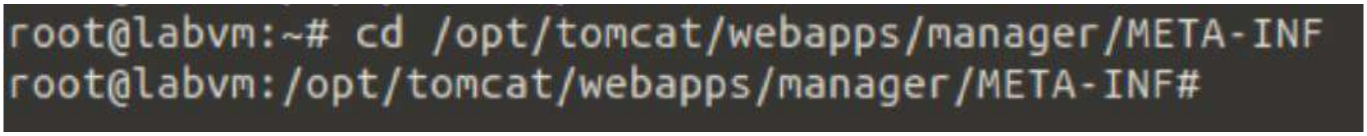
Step 11) We also need to update the context.xml file to remove the IP restriction. Use the same steps to edit the file with the below commands.

cd



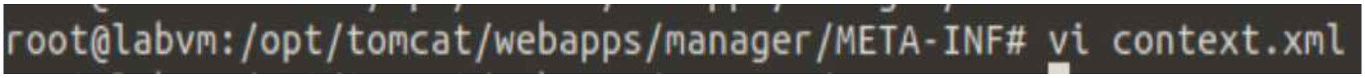
```
root@labvm:/opt/tomcat/conf# cd
root@labvm:~#
```

cd /opt/tomcat/webapps/manager/META-INF



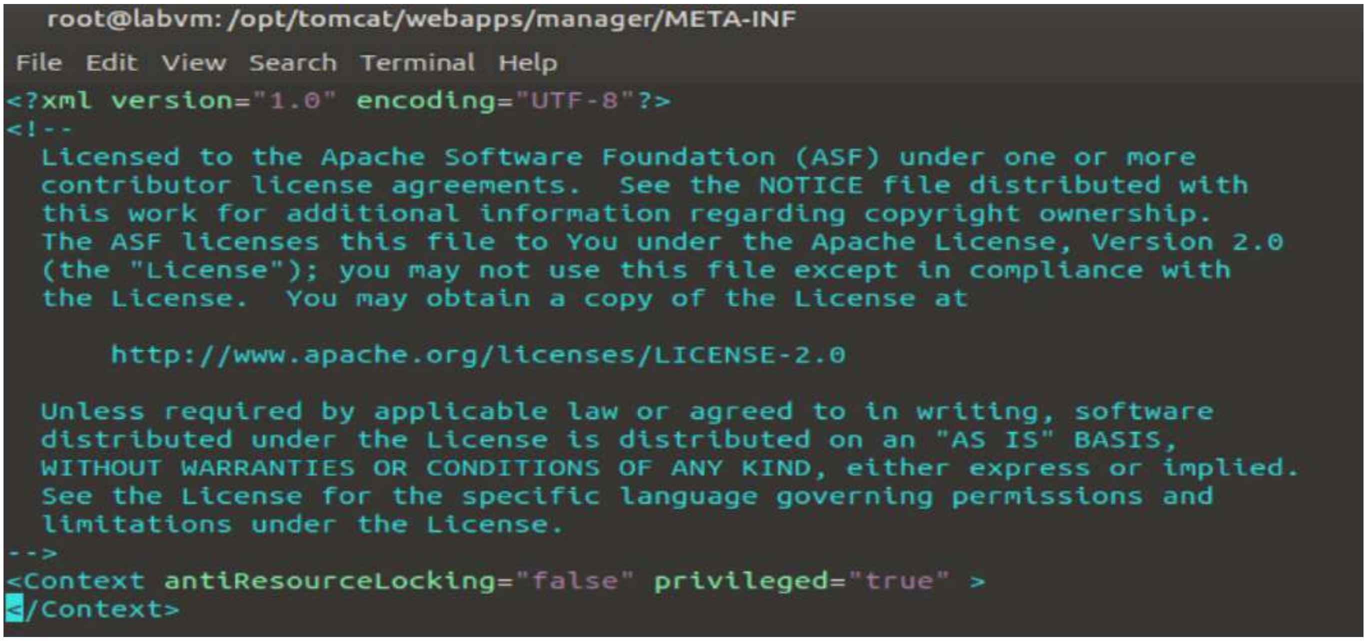
```
root@labvm:~# cd /opt/tomcat/webapps/manager/META-INF
root@labvm:/opt/tomcat/webapps/manager/META-INF#
```

vi context.xml



```
root@labvm:/opt/tomcat/webapps/manager/META-INF# vi context.xml
```

Remove all the content present inside '*context*' as shown in the image below and save it. :wq <enter key>


root@labvm: /opt/tomcat/webapps/manager/META-INF
File Edit View Search Terminal Help

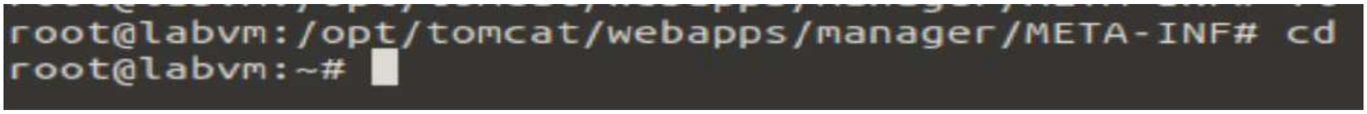
```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<Context antiResourceLocking="false" privileged="true" >
</Context>
```

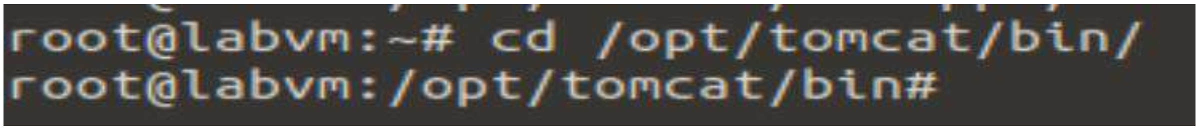
Step 12) Now, all the files are edited successfully. To update the Tomcat, we need to restart the system to accept all our changes. Use the below commands in the server for shutdown and startup of Tomcat. With this step, Tomcat is ready to deploy our container.

cd



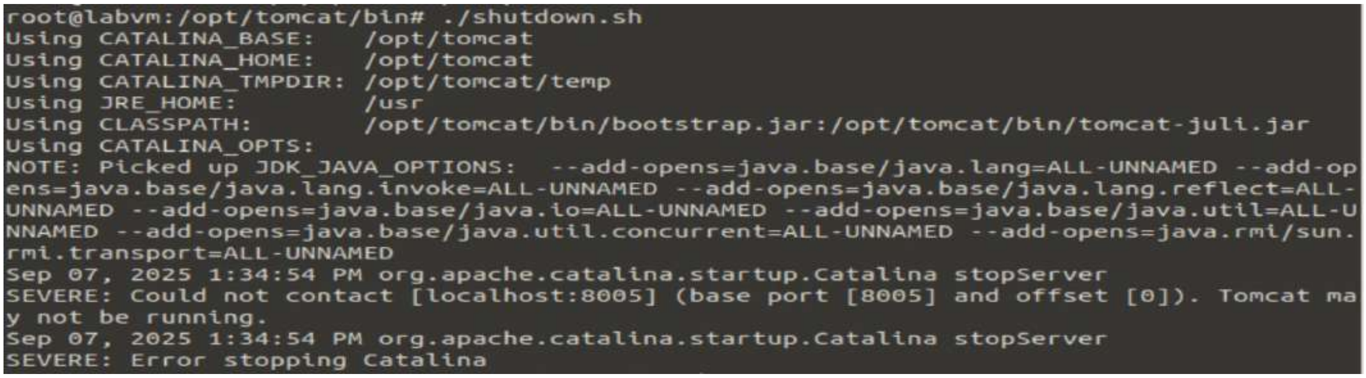
```
root@labvm:/opt/tomcat/webapps/manager/META-INF# cd
root@labvm:~#
```

cd /opt/tomcat/bin/



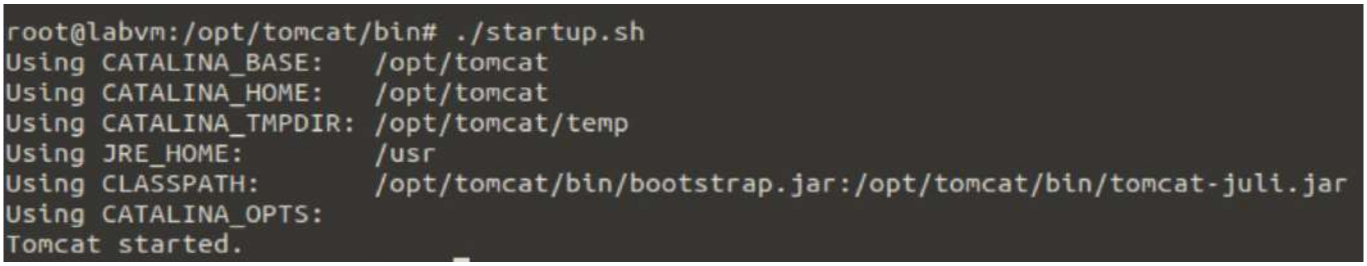
```
root@labvm:~# cd /opt/tomcat/bin/
root@labvm:/opt/tomcat/bin#
```

./shutdown.sh



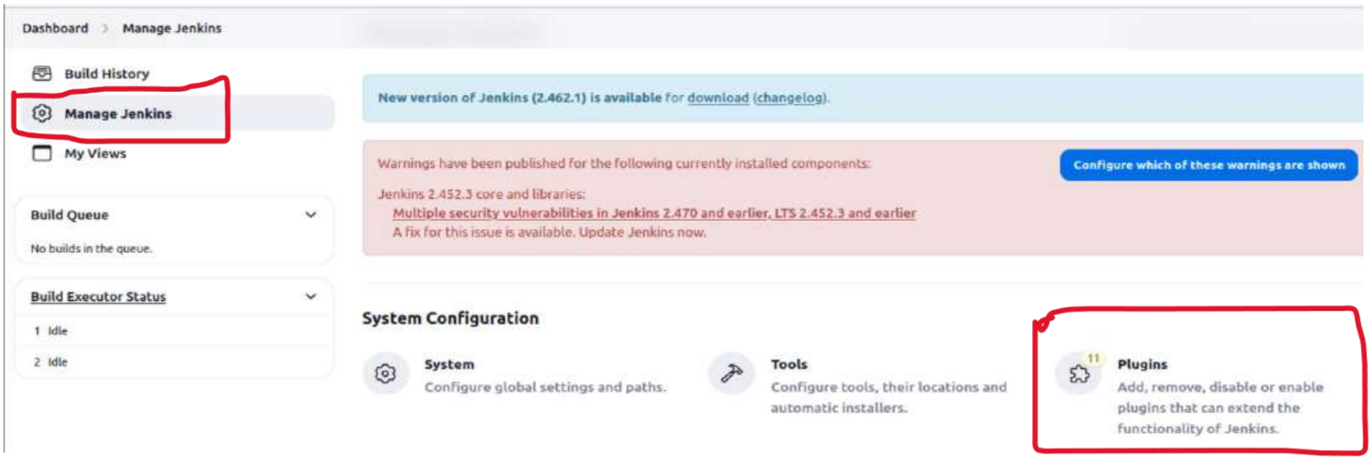
```
root@labvm:/opt/tomcat/bin# ./shutdown.sh
Using CATALINA_BASE:   /opt/tomcat
Using CATALINA_HOME:   /opt/tomcat
Using CATALINA_TMPDIR: /opt/tomcat/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /opt/tomcat/bin/bootstrap.jar:/opt/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
NOTE: Picked up JDK_JAVA_OPTIONS:  --add-opens=java.base/java.lang=ALL-UNNAMED --add-op
ens=java.base/java.lang.invoke=ALL-UNNAMED --add-opens=java.base/java.lang.reflect=ALL-
UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.base/java.util=ALL-U
NNAMED --add-opens=java.base/java.util.concurrent=ALL-UNNAMED --add-opens=java.rmi/sun.
rmi.transport=ALL-UNNAMED
Sep 07, 2025 1:34:54 PM org.apache.catalina.startup.Catalina stopServer
SEVERE: Could not contact [localhost:8005] (base port [8005] and offset [0]). Tomcat ma
y not be running.
Sep 07, 2025 1:34:54 PM org.apache.catalina.startup.Catalina stopServer
SEVERE: Error stopping Catalina
```

./startup.sh



```
root@labvm:/opt/tomcat/bin# ./startup.sh
Using CATALINA_BASE:   /opt/tomcat
Using CATALINA_HOME:   /opt/tomcat
Using CATALINA_TMPDIR: /opt/tomcat/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /opt/tomcat/bin/bootstrap.jar:/opt/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
```

Step 13) In the Jenkins Dashboard, click on Manage Jenkins and then visit Manage Plugins.



Dashboard > Manage Jenkins

Build History
Manage Jenkins
My Views

Build Queue
No builds in the queue.

Build Executor Status
1 Idle
2 Idle

New version of Jenkins (2.462.1) is available for download (changelog).

Warnings have been published for the following currently installed components:
Jenkins 2.452.3 core and libraries:
Multiple security vulnerabilities in Jenkins 2.470 and earlier, LTS 2.452.3 and earlier
A fix for this issue is available. Update Jenkins now.

Configure which of these warnings are shown

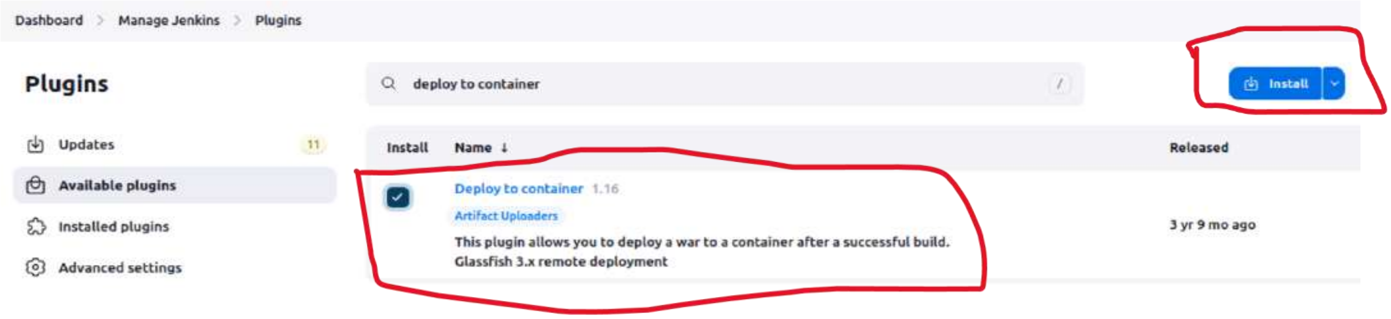
System Configuration

System
Configure global settings and paths.

Tools
Configure tools, their locations and automatic installers.

Plugins (11)
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

Step 14) Click on the *Available* tab and search for the '*Deploy to Container*' plugin. Select the plugin and click on the '*Install without restart*' button.



Dashboard > Manage Jenkins > Plugins

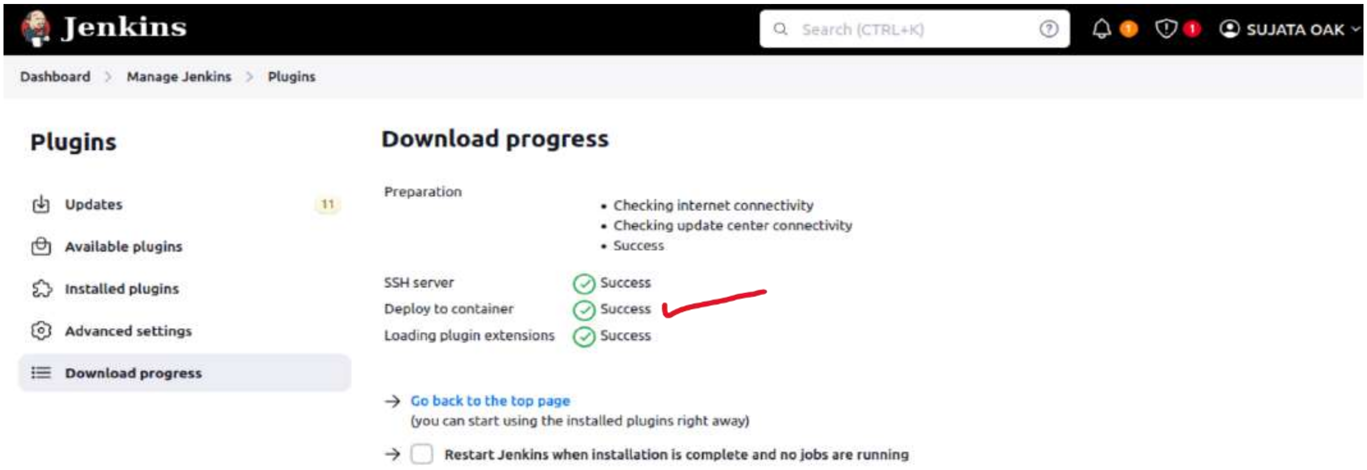
Plugins

Updates 11
Available plugins
Installed plugins
Advanced settings

deploy to container

Install

Install	Name ↓	Released
☑	Deploy to container 1.16 Artifact Uploaders This plugin allows you to deploy a war to a container after a successful build. Glassfish 3.x remote deployment	3 yr 9 mo ago



Jenkins

Search (CTRL+K)

SUJATA OAK

Dashboard > Manage Jenkins > Plugins

Plugins

Updates 11
Available plugins
Installed plugins
Advanced settings
Download progress

Download progress

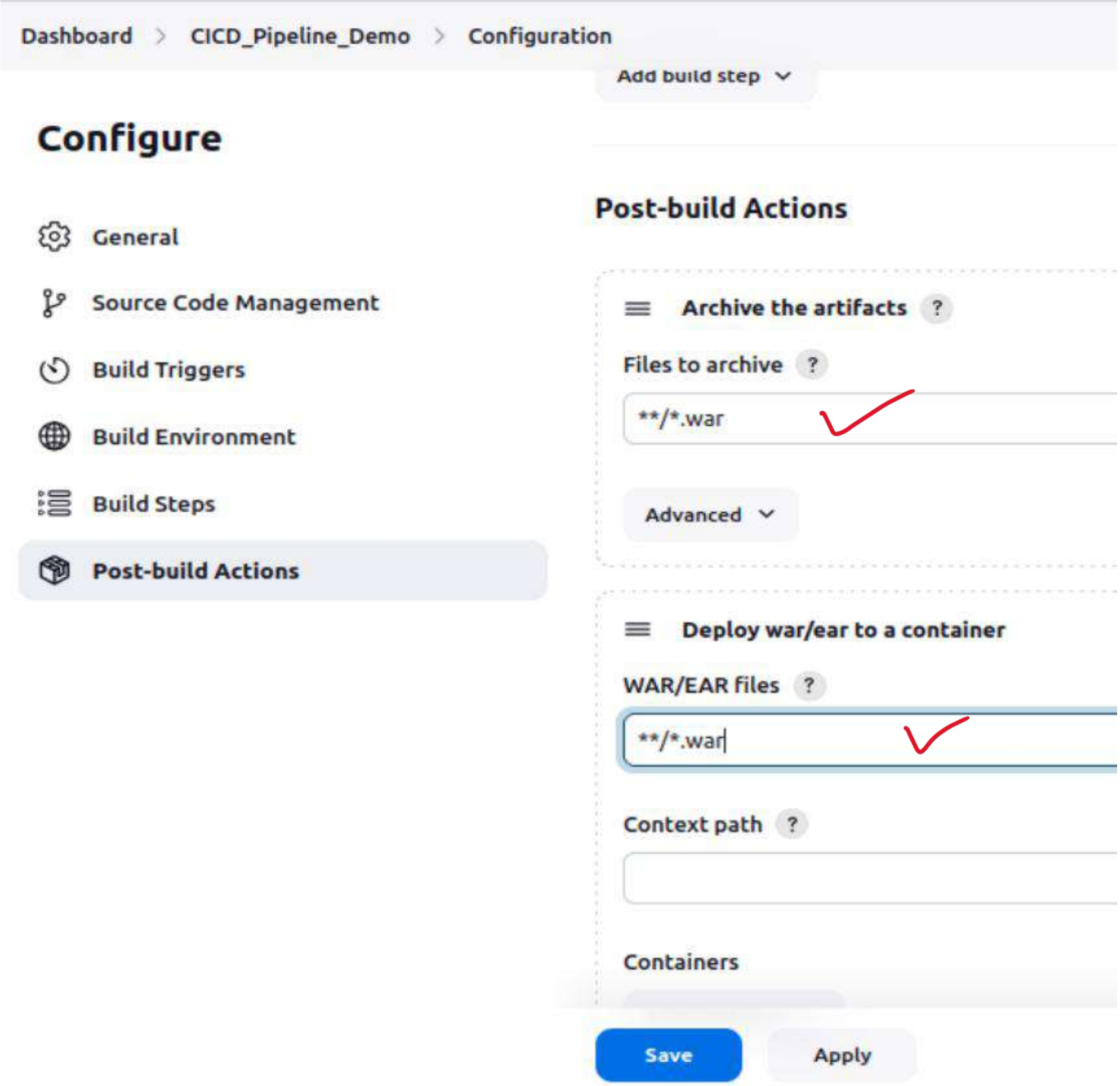
Preparation
- Checking internet connectivity
- Checking update center connectivity
- Success

SSH server ✓ Success
Deploy to container ✓ Success
Loading plugin extensions ✓ Success

→ Go back to the top page
(you can start using the installed plugins right away)

→ □ Restart Jenkins when installation is complete and no jobs are running

Step 15) Go back to your Configure window and select the Post-build Actions tab. Select the '*Add post-build action*' drop-down button and select the '*Deploy war/ear to a container*' plugin. Fill the same path of your war file here, as shown in the image below.



Post-build Actions

Archive the artifacts

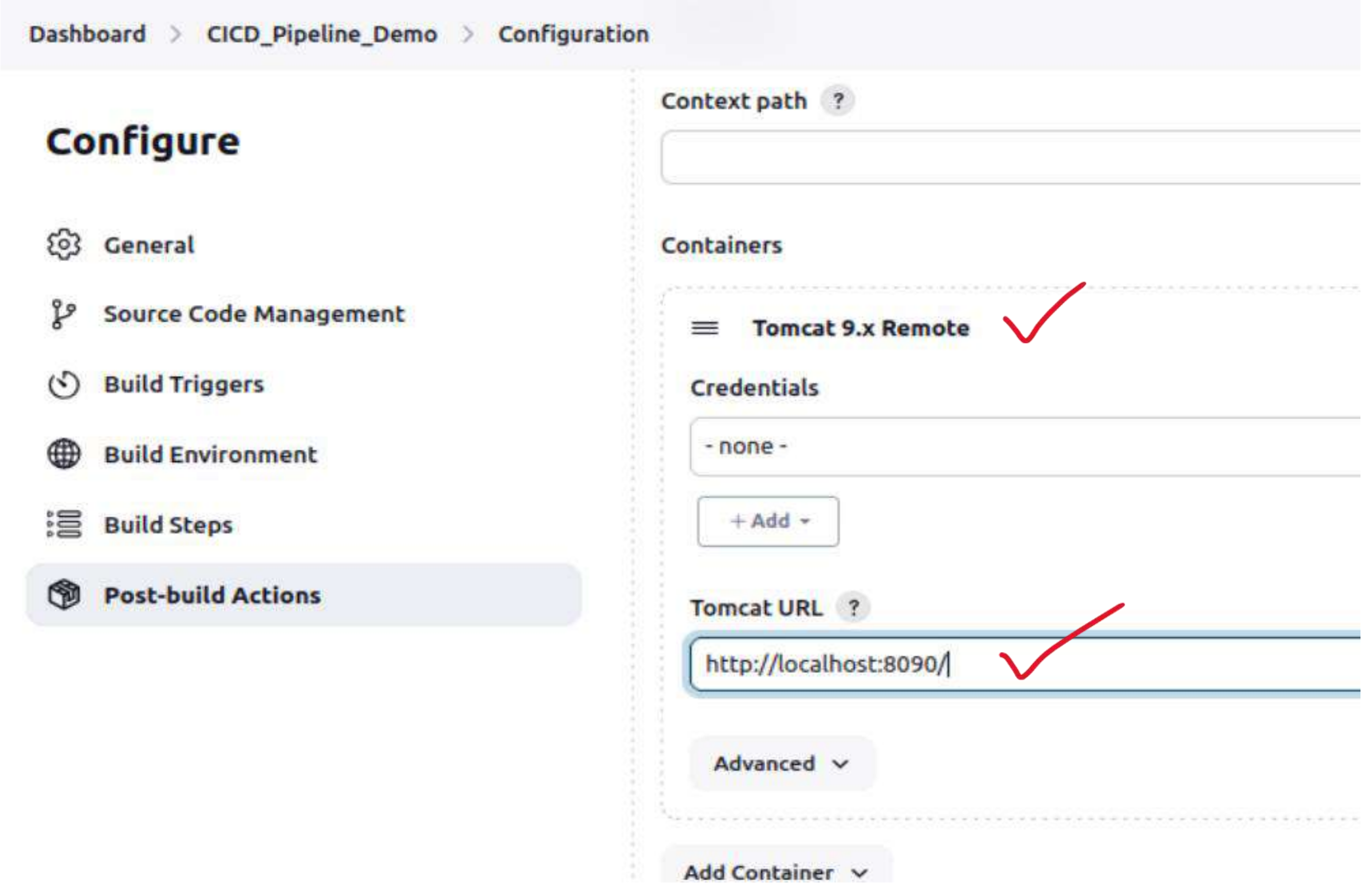
Files to archive: **/*.war

Deploy war/ear to a container

WAR/EAR files: **/*.war

Context path:

Step 16) Now click on the '*Add Container*' button and select the '*Tomcat 9.x Remote*' as we are using version 9 of the Tomcat. Fill in the URL of the same virtual machine with the new port number 8090.


Dashboard > CICD_Pipeline_Demo > Configuration

Configure

Context path ?

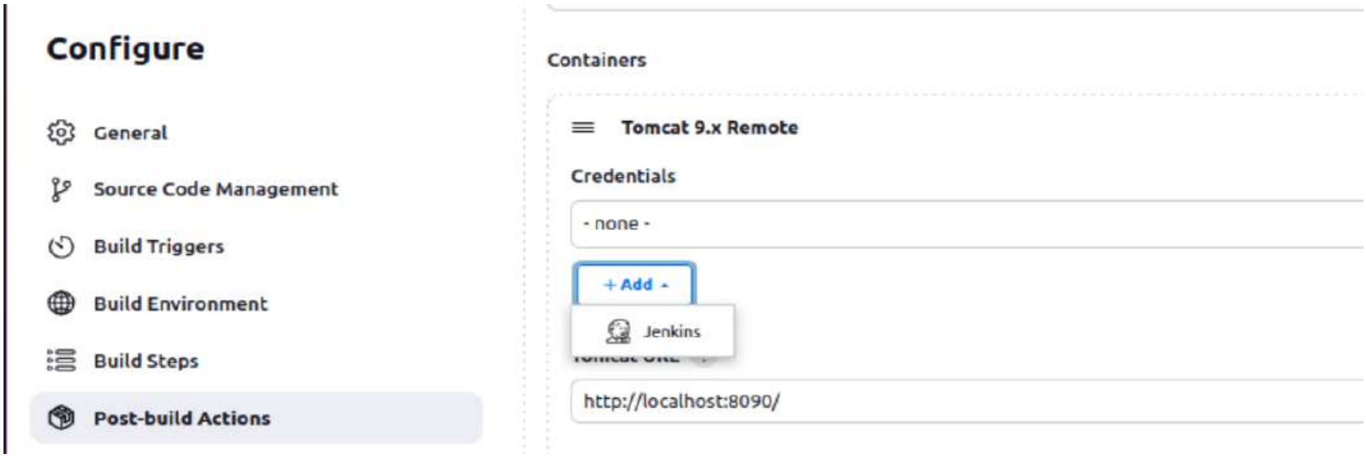
Containers

Tomcat 9.x Remote

Credentials: - none -

Tomcat URL ?: http://localhost:8090/

On the credentials drop-down button, select Jenkins.



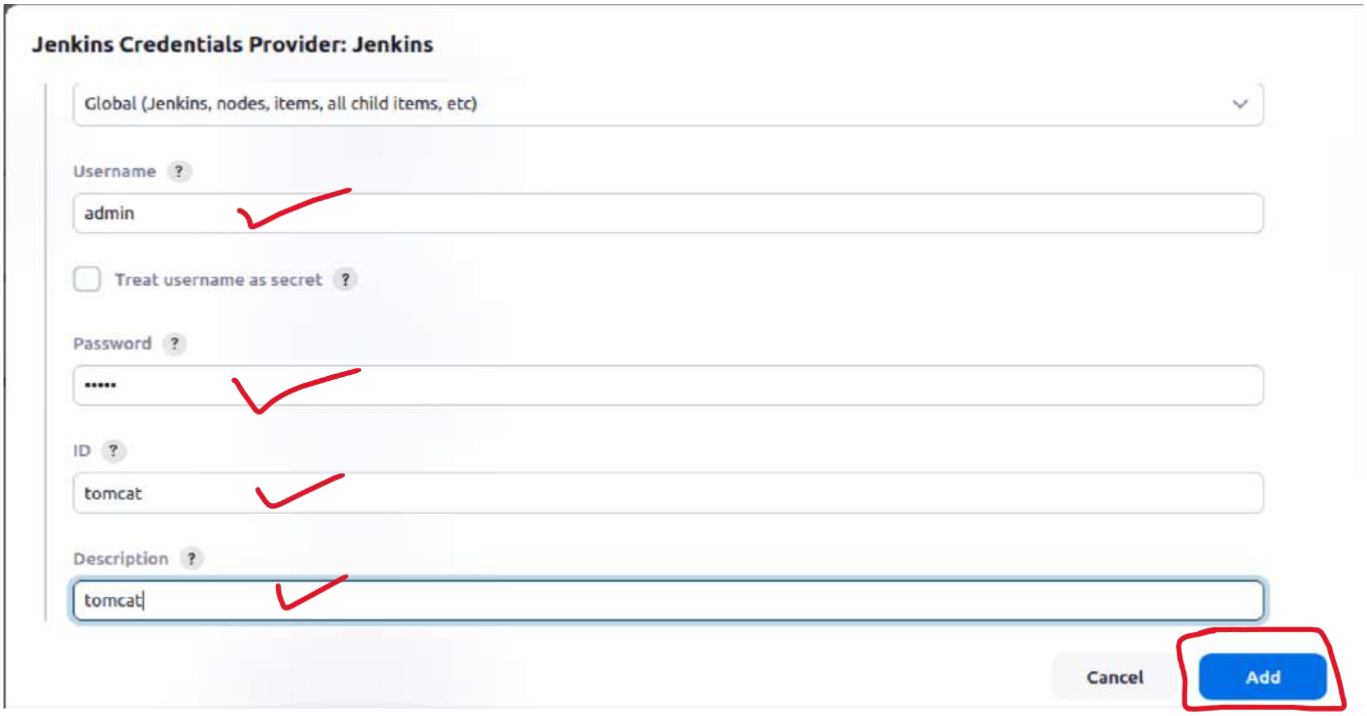
Containers

Tomcat 9.x Remote

Credentials: - none -

http://localhost:8090/

Step 17) In this window, fill in the username and password that we have used in the '*tomcat-users.xml*' file (in Step 10). Fill in the ID, description and click on the button '*Add*'.



Jenkins Credentials Provider: Jenkins

Global (Jenkins, nodes, items, all child items, etc)

Username: admin

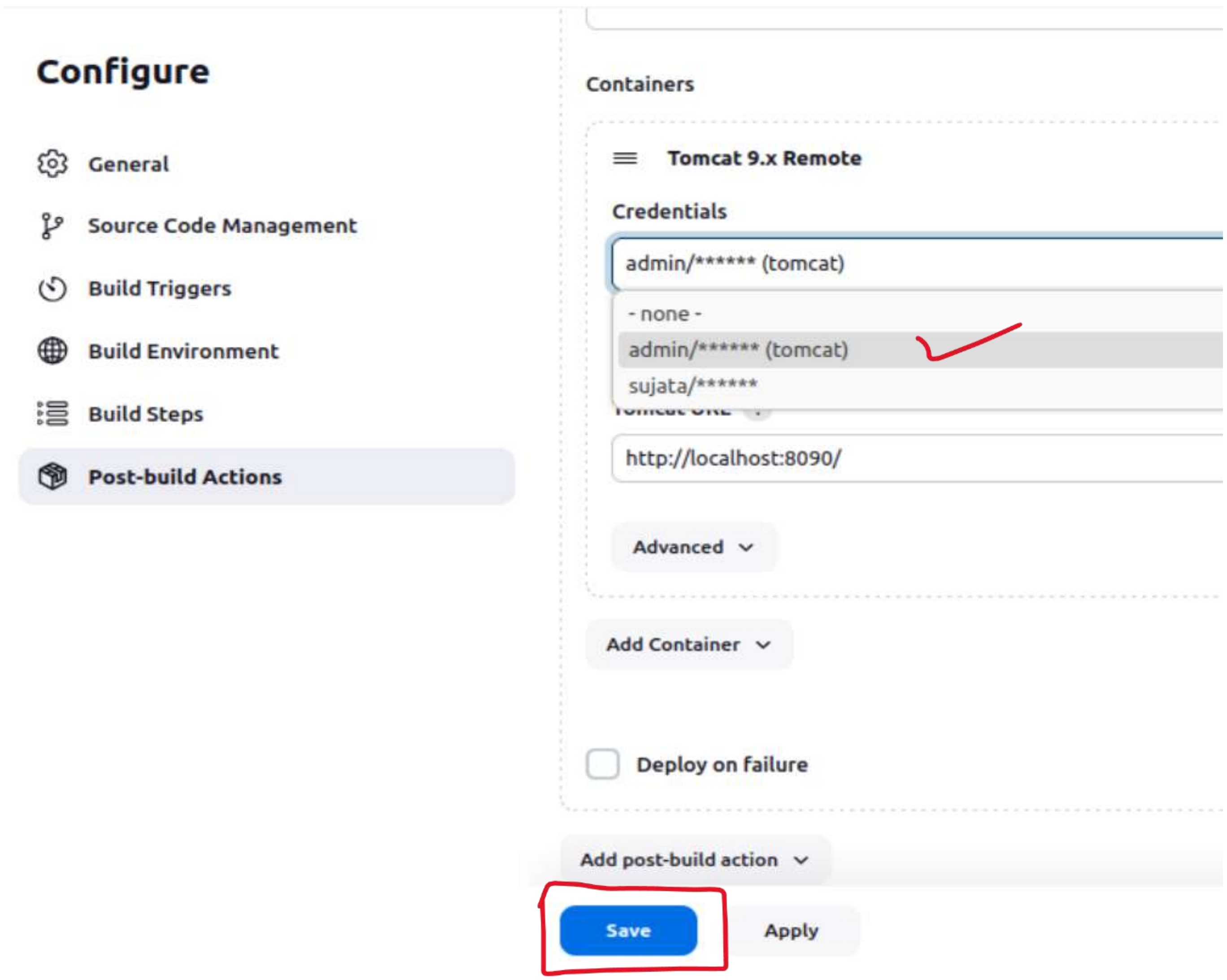
□ Treat username as secret

Password: •••••

ID: tomcat

Description: tomcat

Step 18) Click on the credentials drop-down button and select the recently created credential. Save all the settings and again click on the '*Build Now*' button from the Jenkins dashboard. If the Build is successful, the war file will get deployed.



Containers

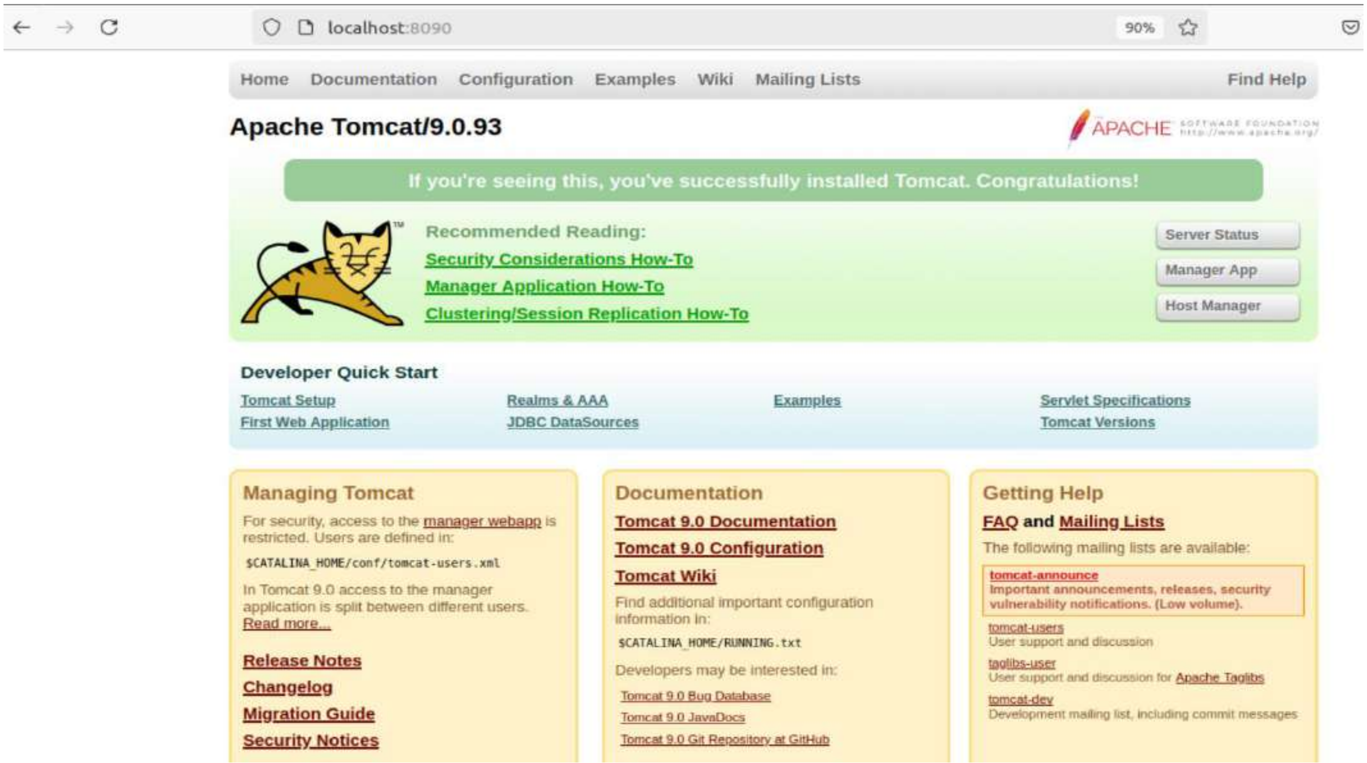
Tomcat 9.x Remote

Credentials: admin/****** (tomcat)

- none -
admin/****** (tomcat)
sujata/******

http://localhost:8090/

□ Deploy on failure



Apache Tomcat/9.0.93

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:
Security Considerations How-To
Manager Application How-To
Clustering/Session Replication How-To

Developer Quick Start

Tomcat Setup	Realms & AAA	Examples	Servlet Specifications
First Web Application	JDBC DataSources		Tomcat Versions

Managing Tomcat

For security, access to the manager webapp is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 9.0 access to the manager application is split between different users. Read more...

Release Notes
Changelog
Migration Guide
Security Notices

Documentation

Tomcat 9.0 Documentation
Tomcat 9.0 Configuration
Tomcat Wiki

Find additional important configuration information in:

```
$CATALINA_HOME/RUNNING.txt
```

Developers may be interested in:

Tomcat 9.0 Bug Database
Tomcat 9.0 JavaDocs
Tomcat 9.0 Git Repository at GitHub

Getting Help

FAQ and Mailing Lists

The following mailing lists are available:

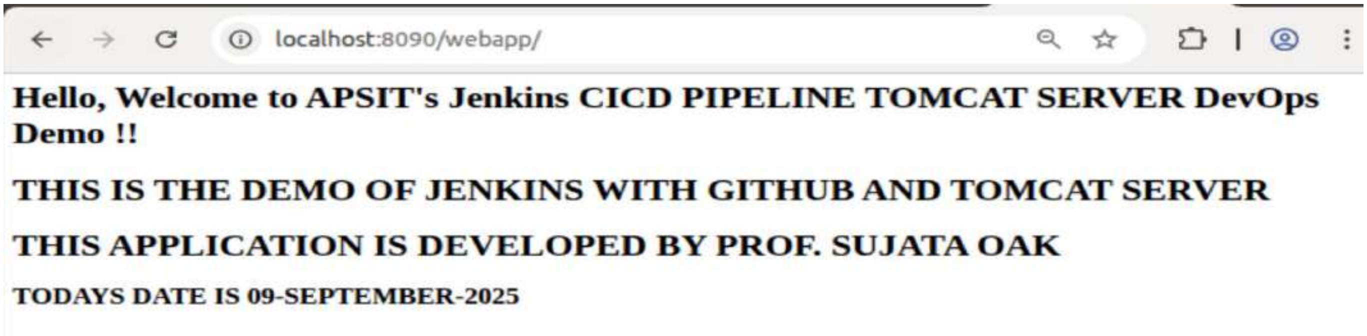
tomcat-announce
Important announcements, releases, security vulnerability notifications. (Low volume).

tomcat-users
User support and discussion

taglibs-user
User support and discussion for Apache Taglibs

tomcat-dev
Development mailing list, including commit messages

http://localhost:8090/webapp/



Hello, Welcome to APSIT's Jenkins CICD PIPELINE TOMCAT SERVER DevOps Demo !!

THIS IS THE DEMO OF JENKINS WITH GITHUB AND TOMCAT SERVER

THIS APPLICATION IS DEVELOPED BY PROF. SUJATA OAK

TODAYS DATE IS 09-SEPTEMBER-2025

Conclusion: By implementing Jenkins pipelines for Tomcat deployment, software teams achieve full automation of the build, test, and deployment lifecycle of web applications. This experiment demonstrates the practical application of CI/CD concepts in a real-world scenario.

```
root@labvm:/home/devasc/Desktop/jenkins# cd
root@labvm:~# cd /opt/tomcat/bin
root@labvm:/opt/tomcat/bin# ./startup.sh
root@labvm:/opt/tomcat/bin# ./shutdown.sh
```