**DEPLOY WAR/EAR file in to tomcat by Jenkins**

**TOMCAT**: A Java-based web server and servlet container. It's mainly used to host Java web applications, providing an environment to run applications that respond to web requests, like a website or API. If you need a server to handle HTTP requests and serve applications to users, Tomcat is your choice.

STEP:1 CREATE TWO INSTANCES i.e., one instance for Jenkins installation and the other for tomcat installation

* STEP:2 Connect the Jenkins instance with gitbash/linux/putty then run the following commands

**sudo su**

**sudo dnf update -y**

**yum list java\***

**sudo dnf install -y java-17-amazon-corretto.x86\_64**

**sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo**

**sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key**

**yum install jenkins**

**service Jenkins start**

**systemctl start Jenkins**

**systemctl status Jenkins**

**cat <link>**

readlink -f $(which java): By giving this command in the git bash we will be getting JDK path like this /usr/lib/jvm/java-17-amazon-corretto.x86\_64/bin/bash

From this path remove bin/bash. Then paste the remaining path obtained in the JDK section wherever you need

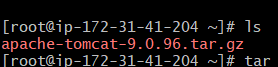
/usr/lib/jvm/java-17-amazon-corretto.x86\_64

* Install the necessary plugins and get connect to dashboard

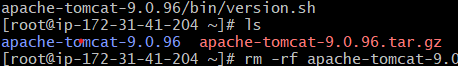
STEP:3 Maven should be there in your local system

STEP:4 Connect the Tomcat instance with gitbash/linux/putty then run the following commands

* wget <https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.96/bin/apache-tomcat-9.0.96.tar.gz>
* ls



* tar xvzf <give the unzip file that is displayed in the above image>
* ls



* rm -rf <give the unzip file link over here>
* ls 🡪 By giving ls command we can see that the unzip file has been deleted.



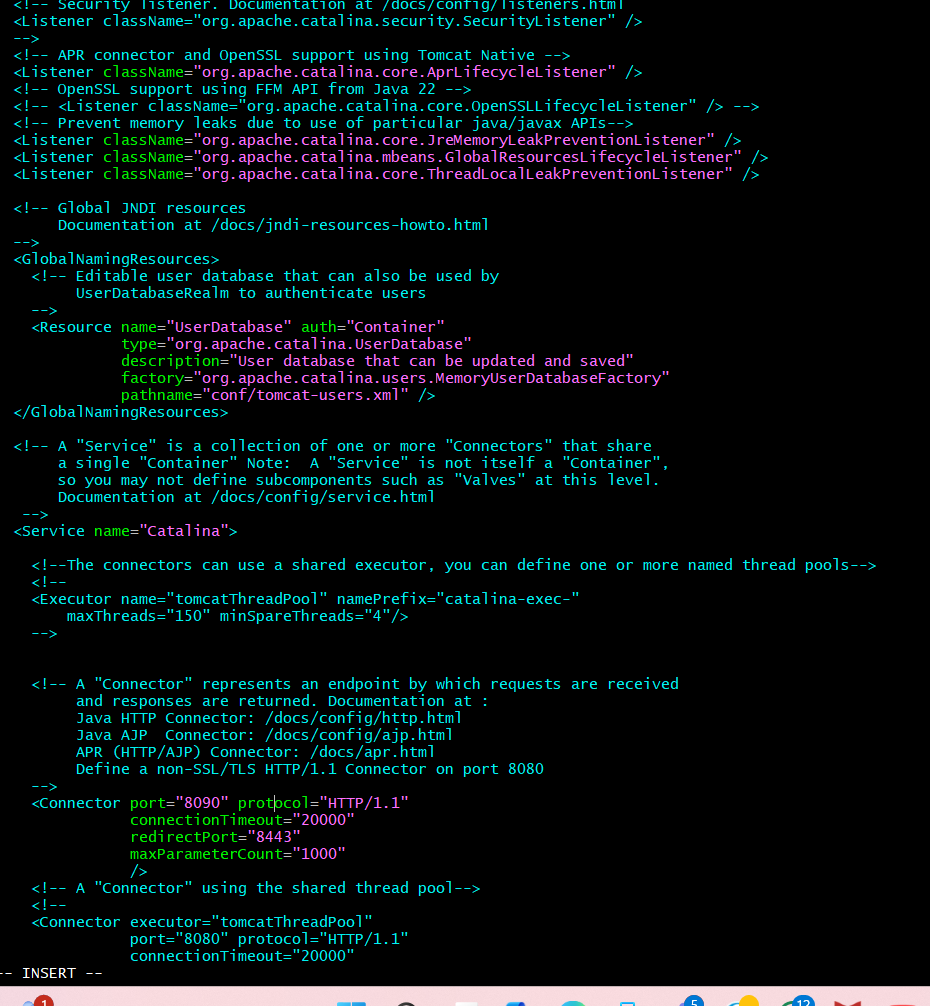
* mv apache-tomcat-9.0.96 tomcat 🡪mv command will change the file name from original name. Here I am changing “apache-tomcat-9.0.96” to “tomcat” as shortform
* cd tomcat 🡪 Changing to tomcat directory
* yum install java -y 🡪Install java
* ls 🡪The following files and directories gets displayed



* cd conf 🡪Navigating into the conf directory
* ls

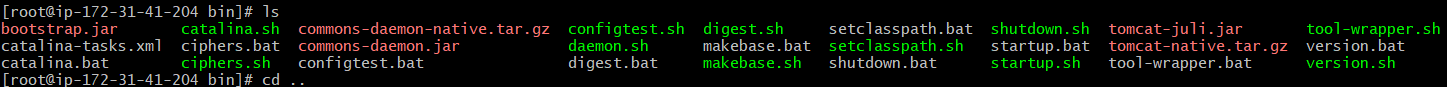


* vi server.xml 🡪 vi editor gets opened and here we need to change the port. The default port will be 8080. As we are opening Jenkins in the same port the tomcat port needs to be changed. Here I have changed the port to 8090.



* cd .. 🡪 from the conf directory we are getting back to tomcat directory
* cd bin
* ls 🡪 The files and directories in the bin gets displayed. Like this when we navigate in to any directory we can check the inside files and directories by giving ls command.





* sh shutdown.sh
* sh startup.sh
* cd .. 🡪From bin dir we will go back to tomcat dir.
* cd conf
* ls
* vi tomcat-users.xmls

<role rolename="manager-gui"/>  
<user username="tomcat" password="s3cret" roles="manager-gui"/>  
<role rolename="manager-script"/>  
<user username="tomcat" password="s3cret" roles="manager-gui,manager script"/>

Add the given roles, username and password as shown in the figure

* cd .. 🡪 From conf getting back to tomcat dir
* cd bin 🡪 Navigate to bin
* sh shutdown.sh
* sh startup.sh
* cd .. 🡪 From bin getting back to tomcat dir
* ls
* cd webapps 🡪Navigate to webapps
* ls



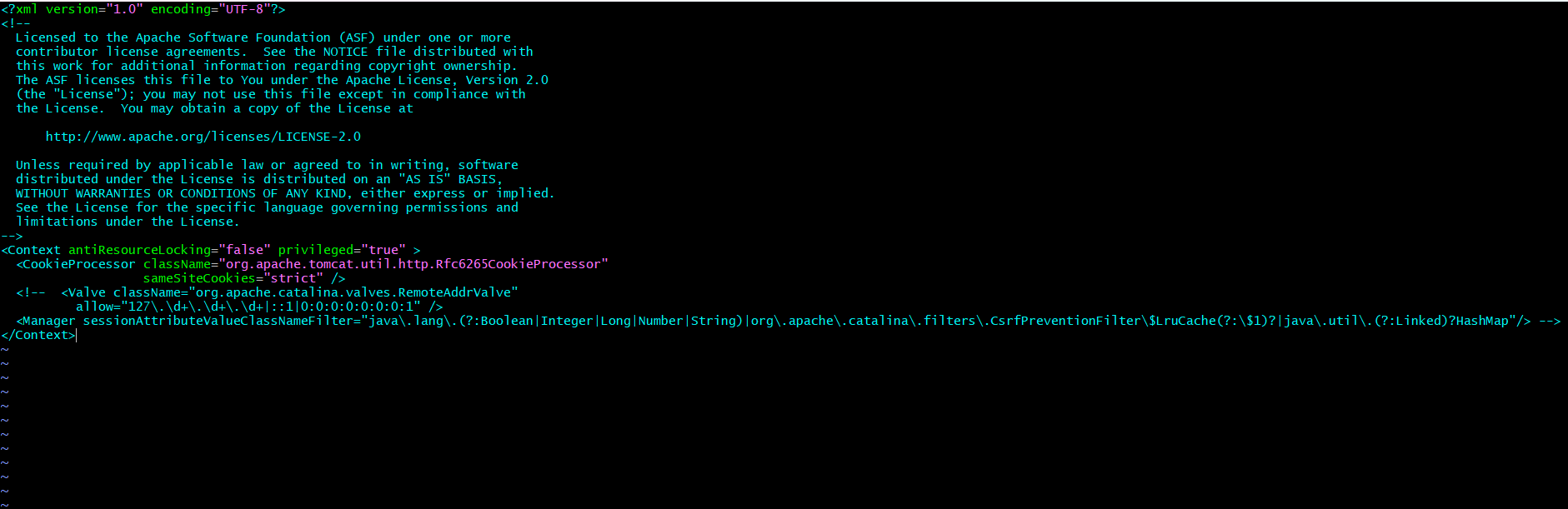
* cd manager 🡪Navigate to manager
* ls



* cd META-INF 🡪Navigate to META-INF
* ls



* vi context.xml

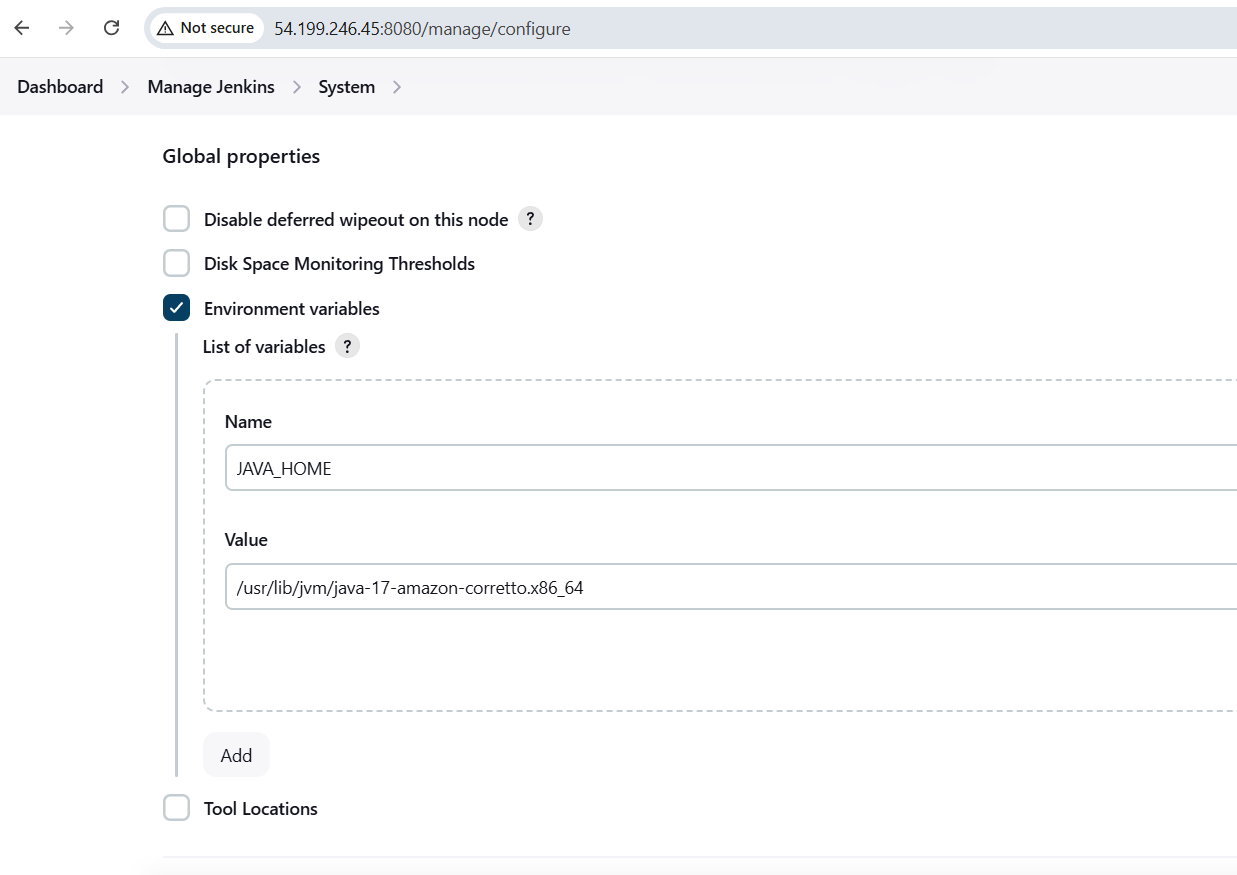


Add <!-- symbol before Valve and “-->” symbol after HashMap to disable it.

* cd .. 🡪 back to manager dir
* cd .. 🡪 back to webapps dir
* cd .. 🡪 back to tomcat dir
* cd bin 🡪 after getting into tomcat dir we will get navigate to bin dir
* sh shutdown.sh
* sh startup.sh 🡪 Tomcat gets started

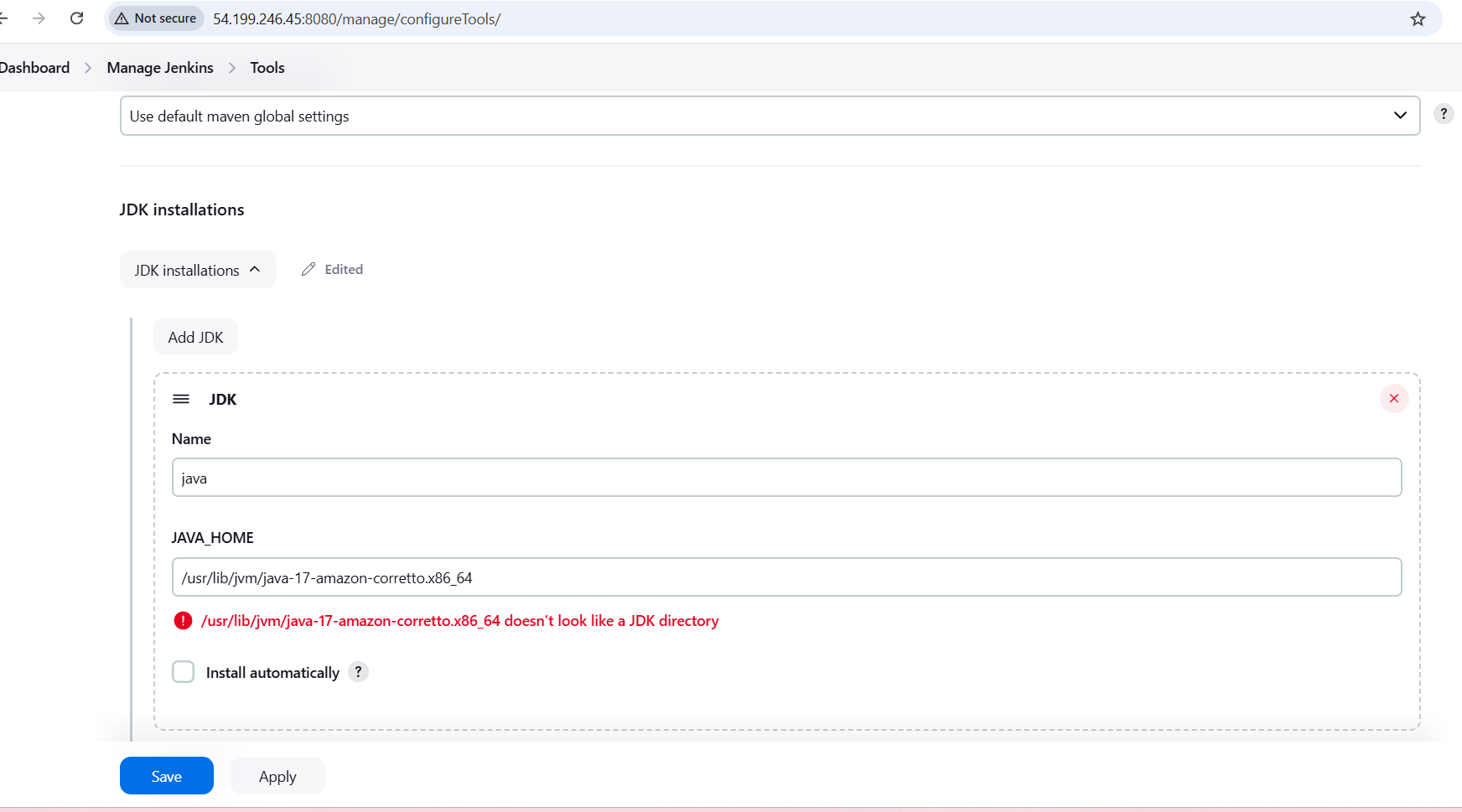
STEP5: In the Jenkins Dashboard install some plugins like container to deploy plugin, Maven Integration plugin, Pipeline stage view plugin and git plugin.

STEP6: Go to manage Jenkins 🡪 Systems 🡪 Global properties 🡪 Environment variables 🡪Under the name section add JAVA\_HOME and value as /usr/lib/jvm/java-17-amazon-corretto.x86\_64



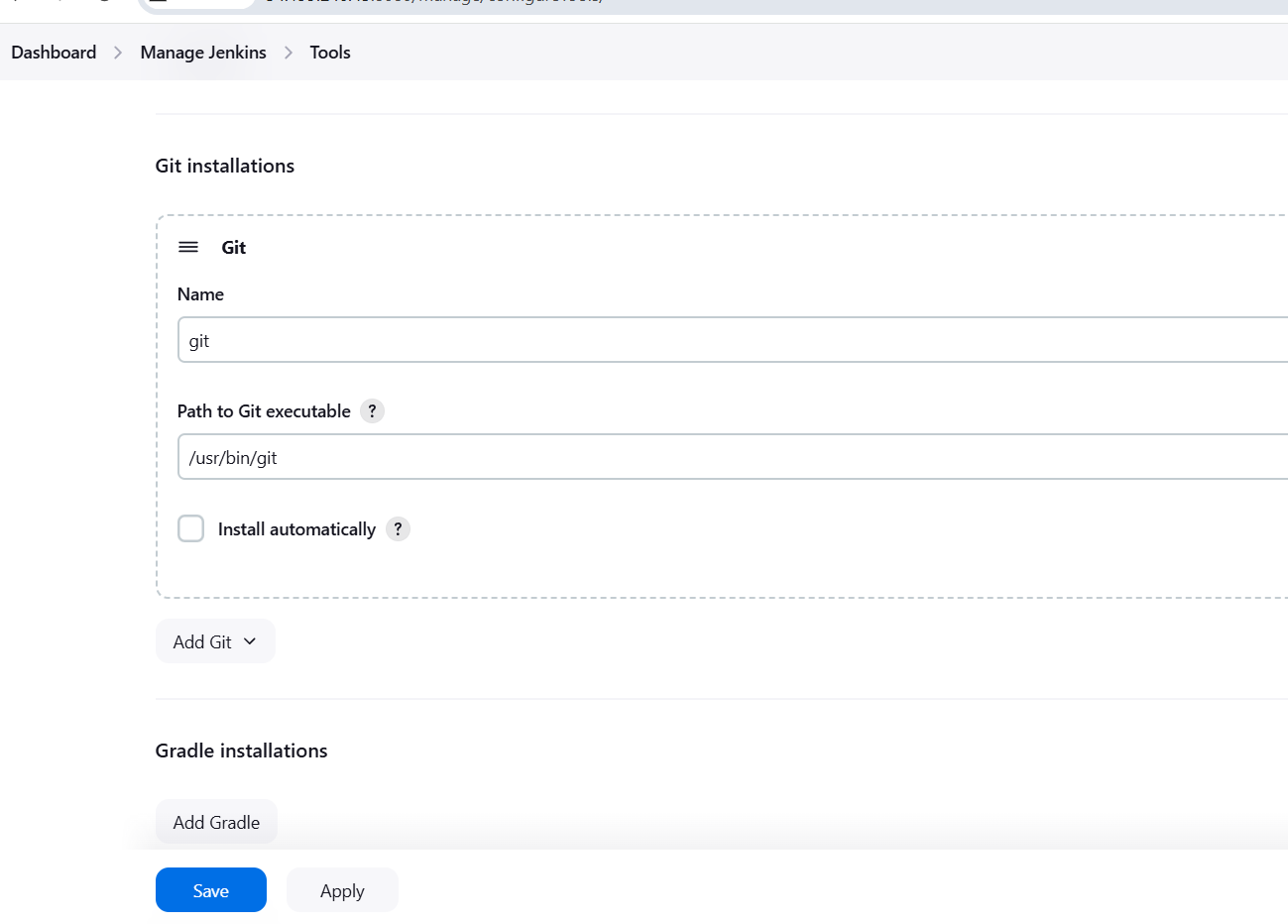
STEP7: Again manage Jenkins 🡪 Tools 🡪 jdk installations 🡪 Name:java 🡪

JAVA\_HOME: /usr/lib/jvm/java-17-amazon-corretto.x86\_64



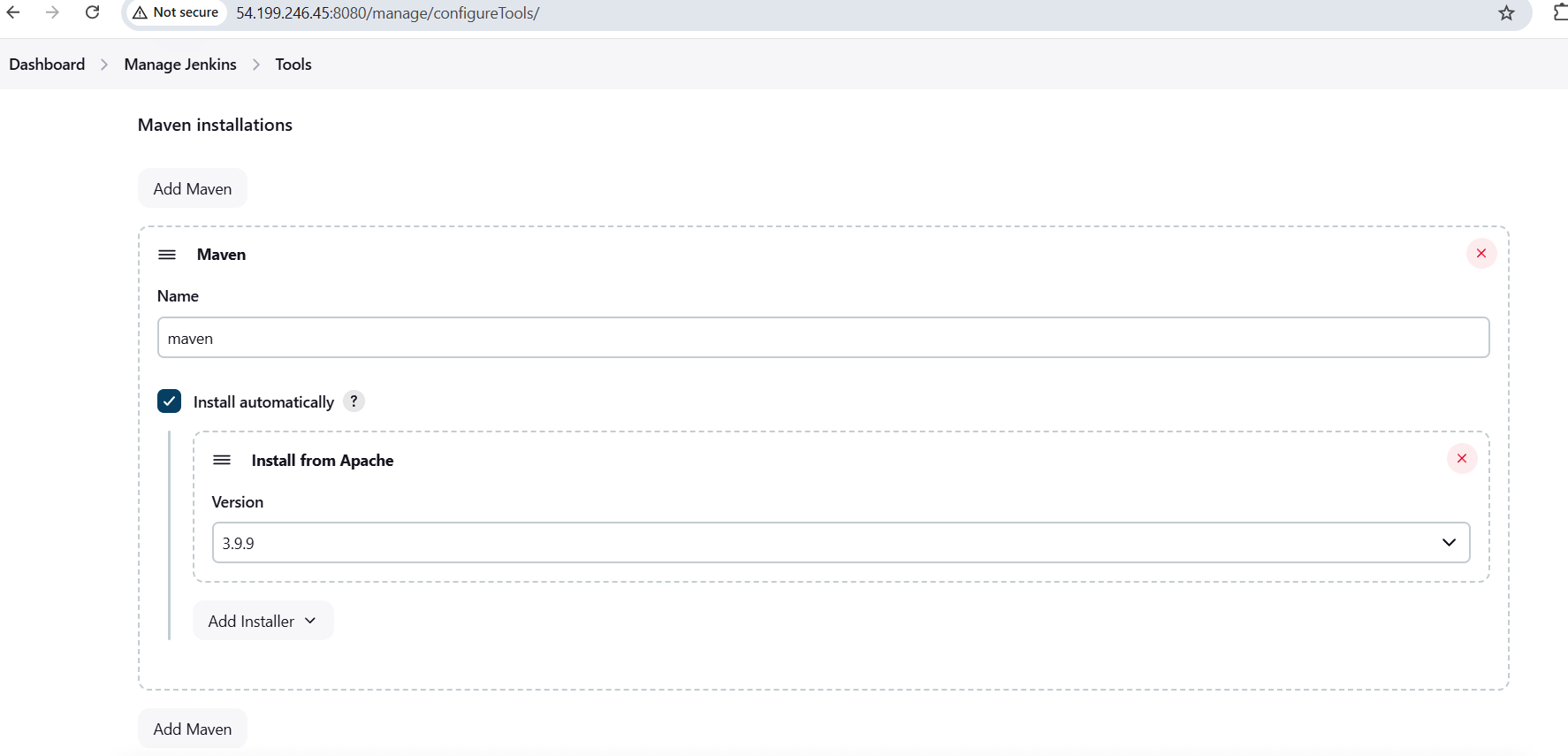
STEP8: Manage Jenkins 🡪 Tools 🡪 Git installations 🡪 Name: git 🡪 Path: user/bin/git

\*\*To know the git path use the command “ which git “ in the git bash. Then it shows the path.\*\*



STEP9: Manage Jenkins 🡪 Tools 🡪 Maven Installations 🡪 name should be selected from dropdown 🡪 and the version should be given that has been installed in our local system

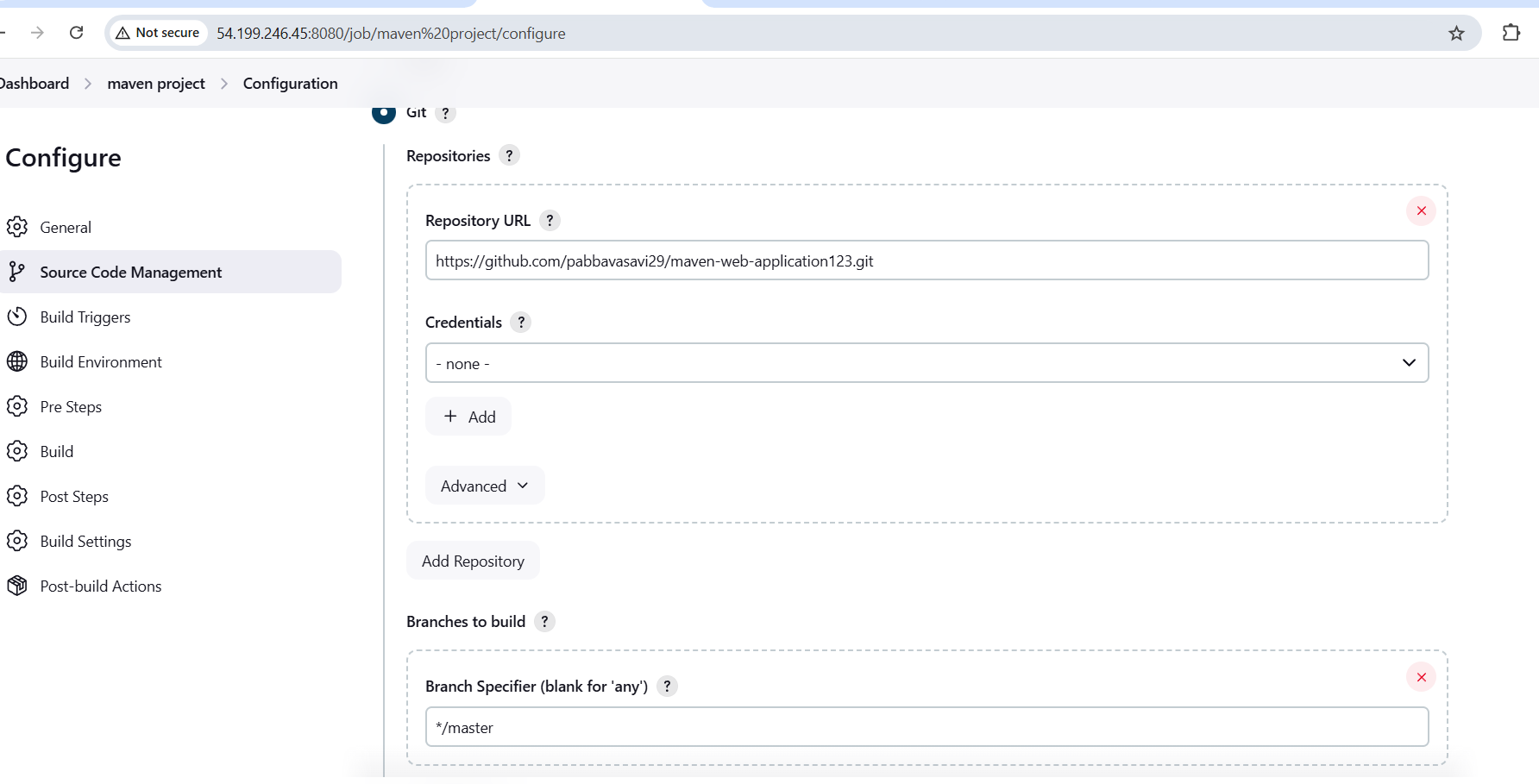
In my case I have installed 3.9.9 version



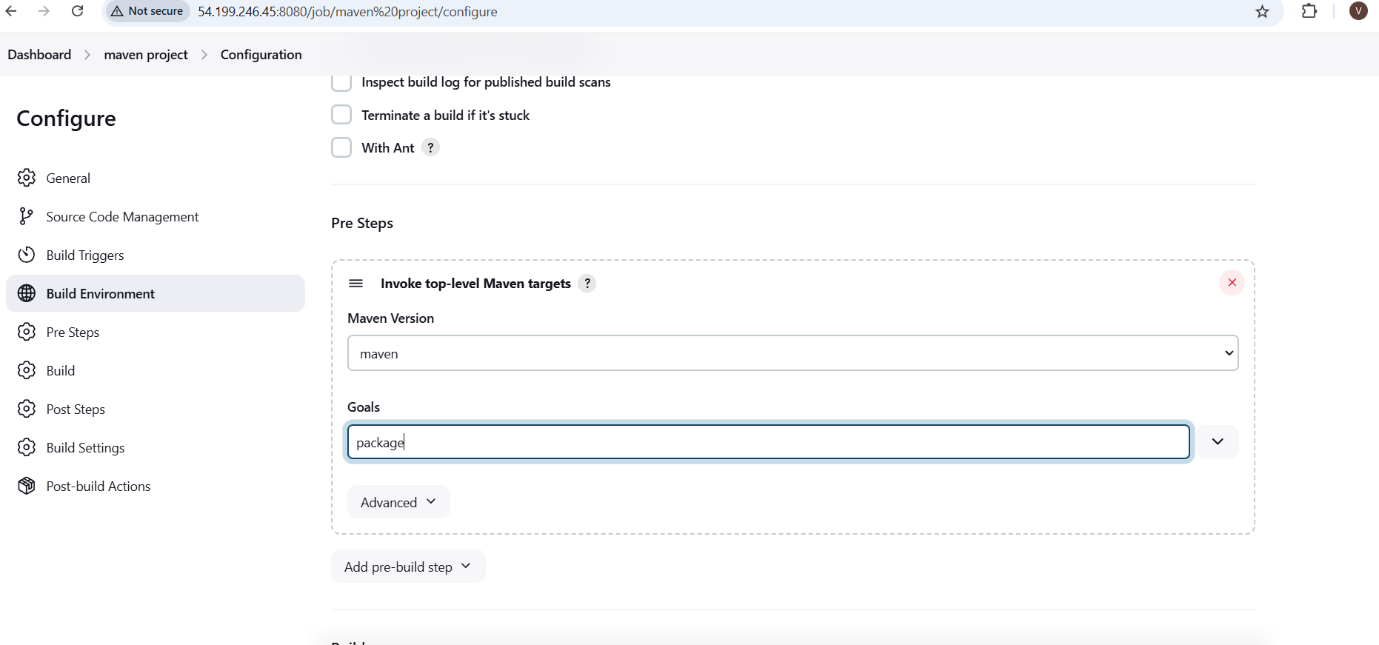
Click on Save and apply

STEP10: Jenkins dashboard 🡪 New Item 🡪Give some name🡪 Select maven Project 🡪 Click on ok

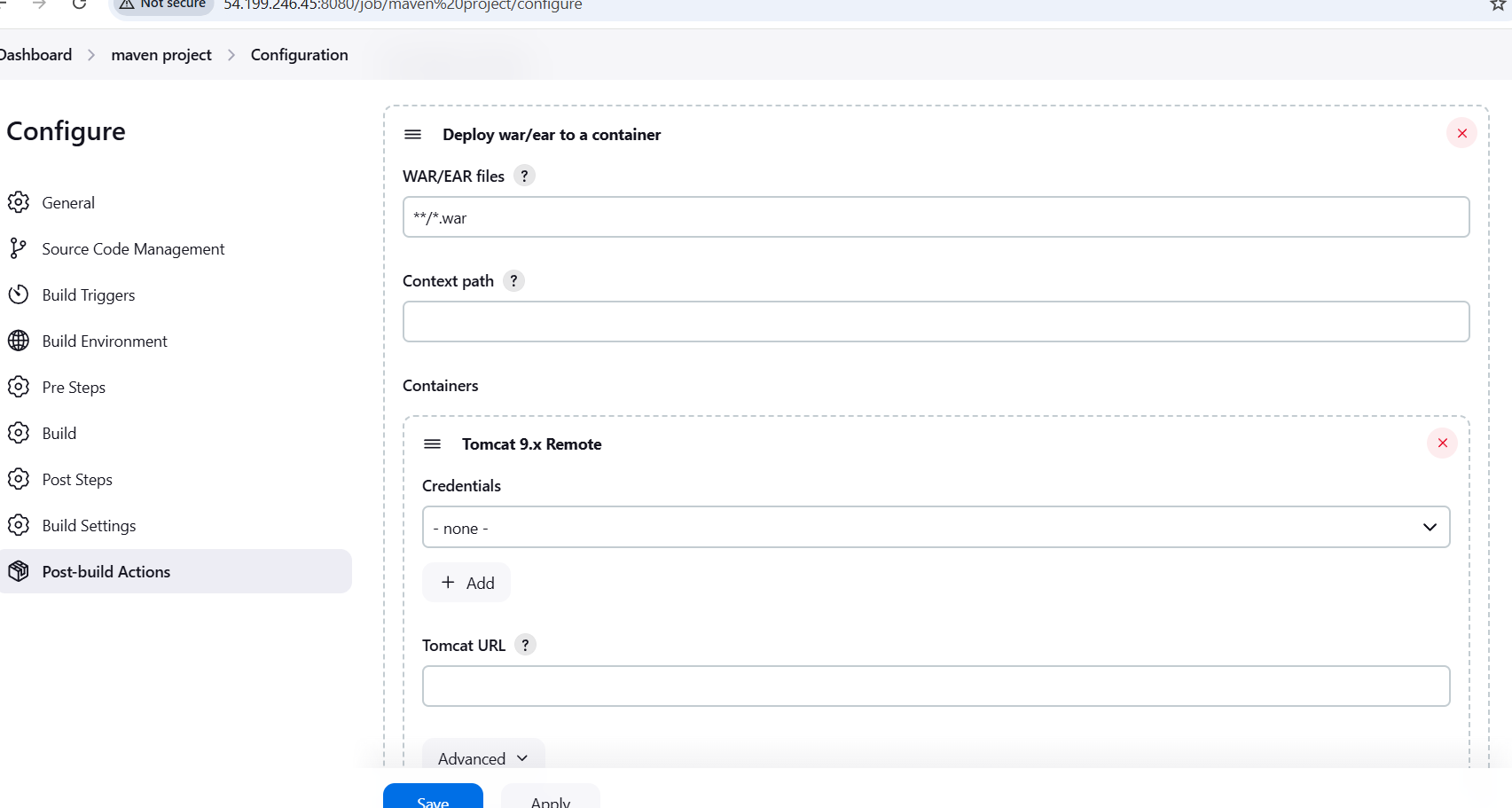
STEP11: Under the source code management select Git and add repository url. Also specify the default branch of that repository i.e., \*/master or \*/main



STEP12: Pre-steps 🡪 Invoke top level maven targets 🡪 select maven version from dropdown 🡪 add package under the goals section



STEP13: Post-Build actions 🡪 select deploy war/ear to a container from the dropdown 🡪 under the WAR/EAR files section give \*\*/\*.war



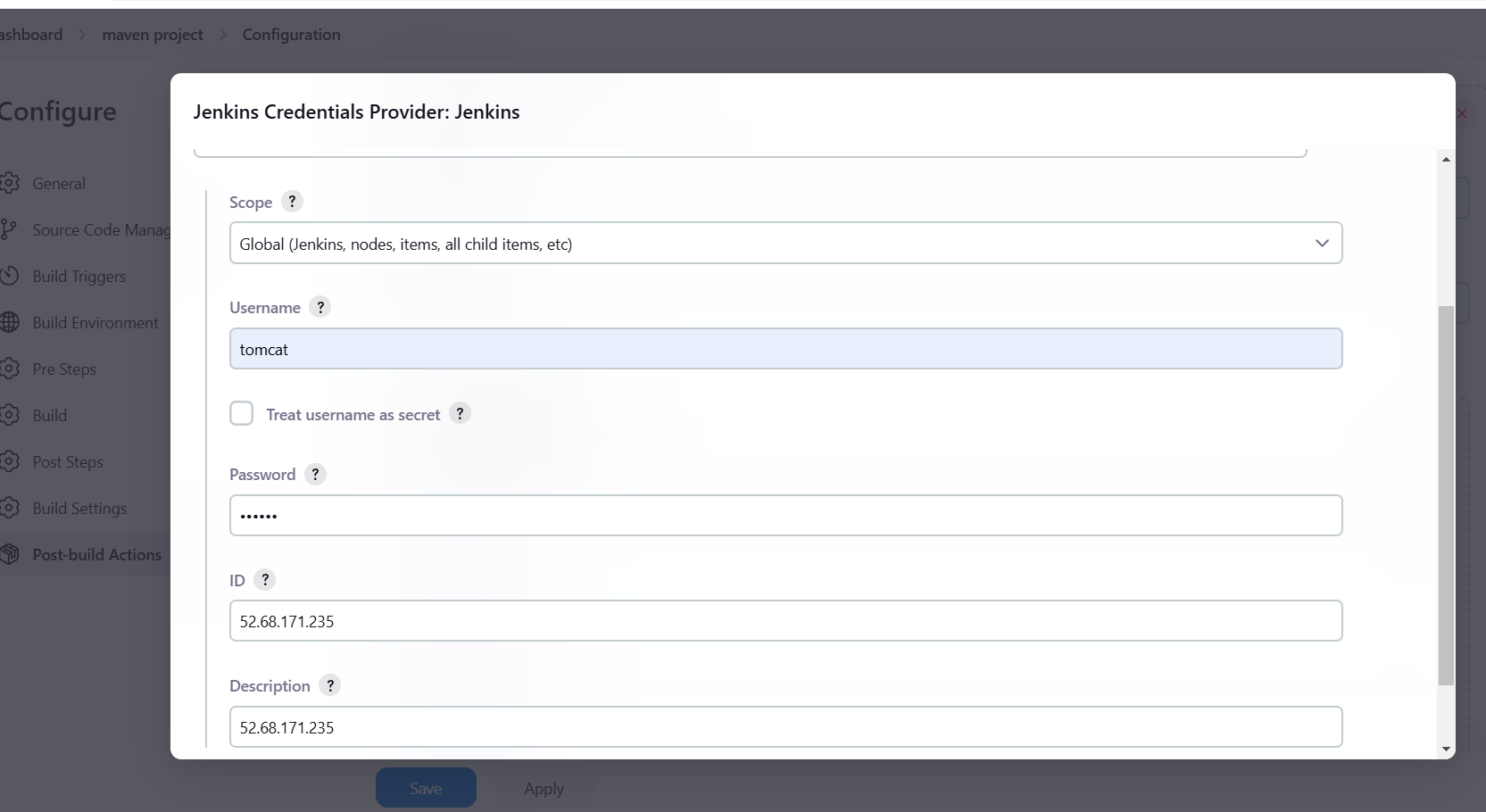
STEP14: In the same section add tomcat credentials i.e., username and password which we have added in the tomcat-users.xml file.

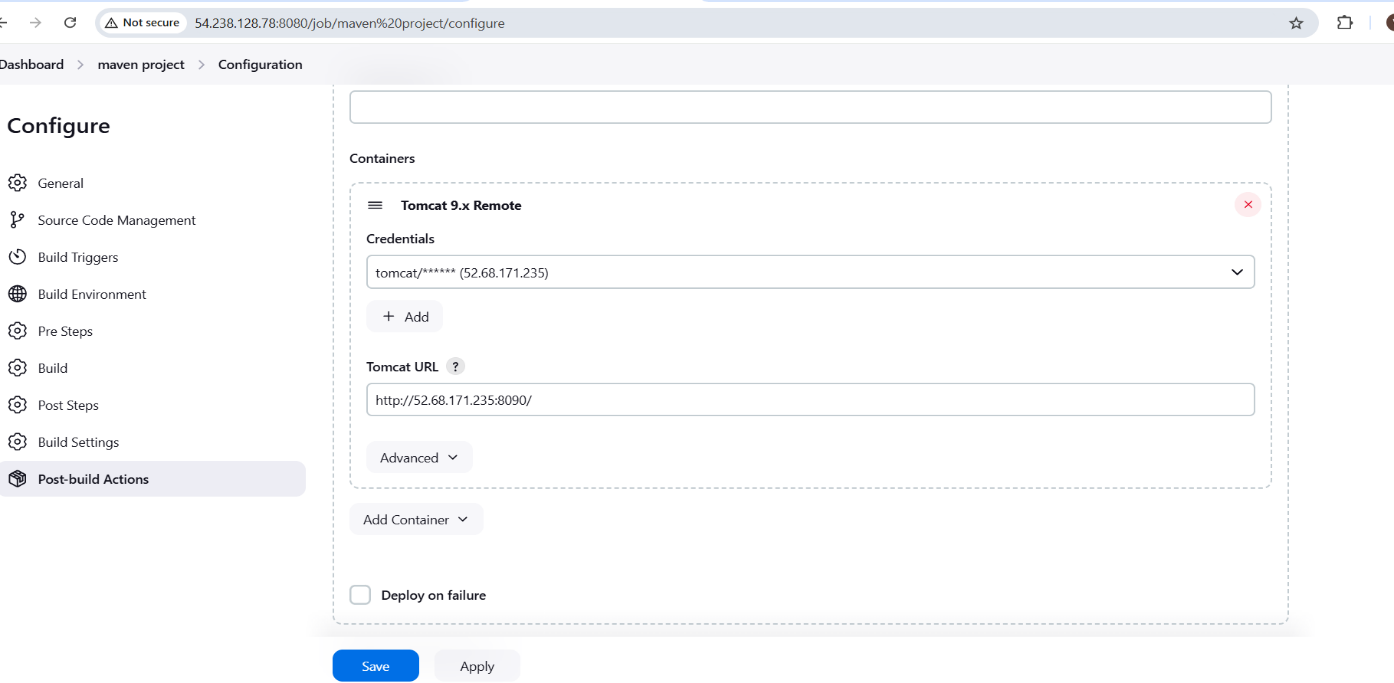
In my case username is tomcat

password is s3cret.

Give the Public Ip of our tomcat instance in the ID and description section

And click on add. Then the credentials gets added



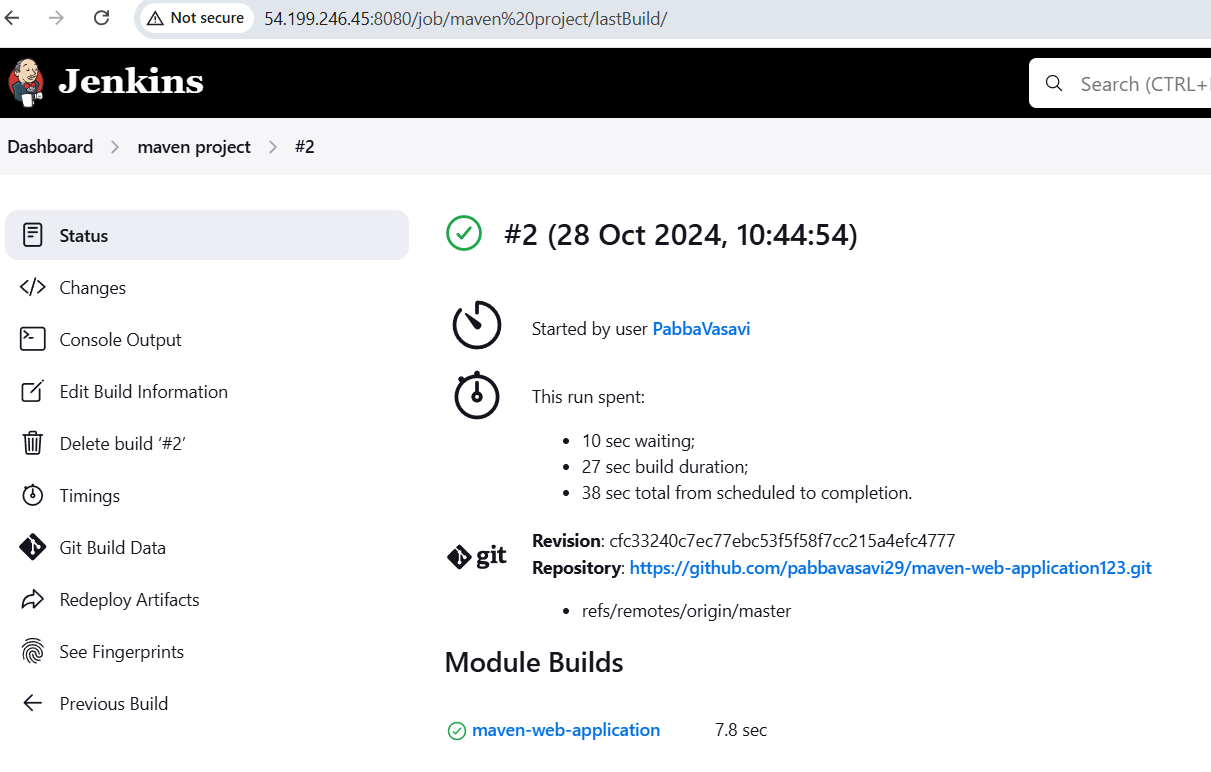


After adding Credentials we have to select the added credentials from the dropdown in the CREDENTIALS section and also we have specify tomcat url in the TOMCAT URL section.

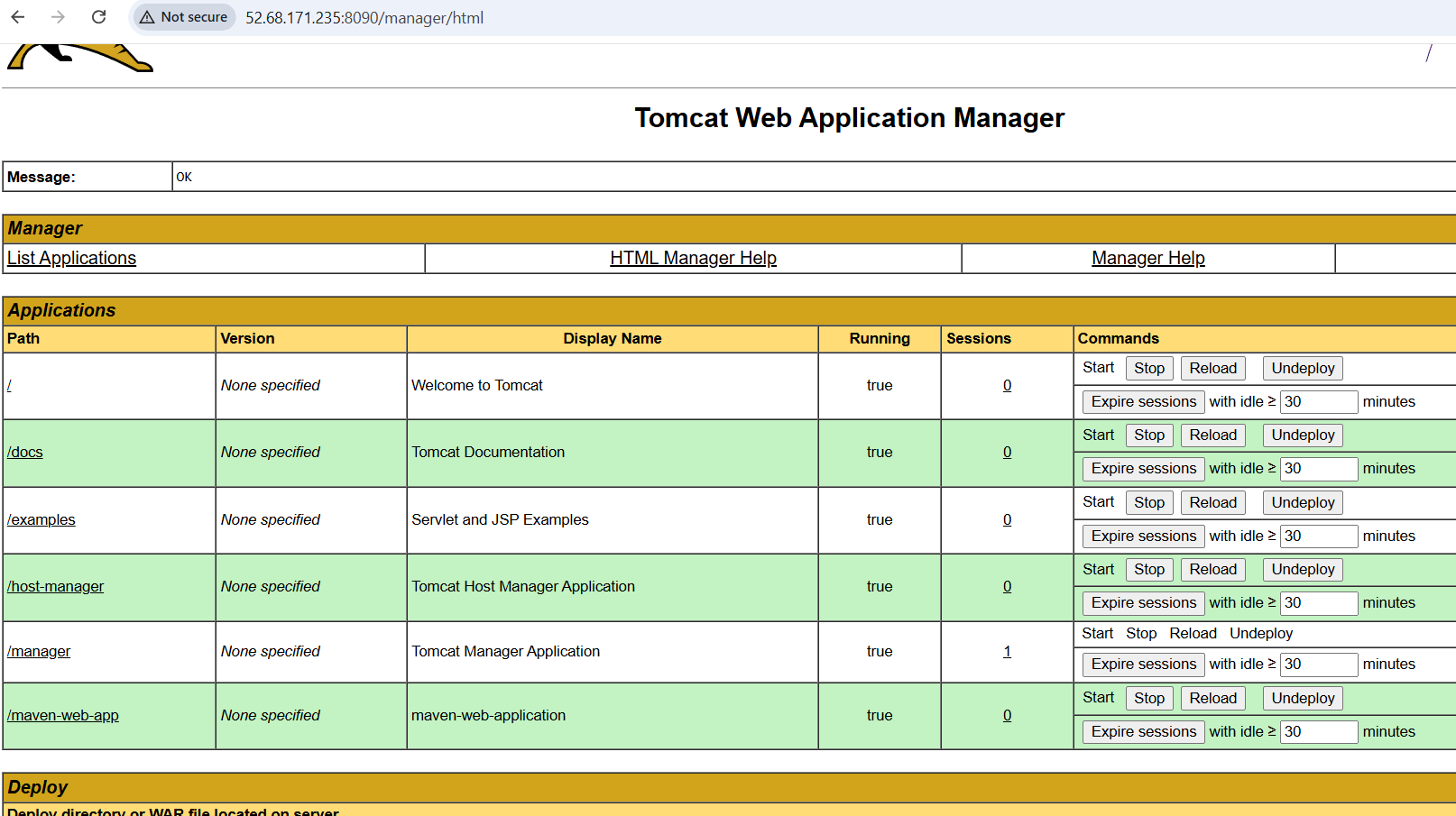
STEP15: Click on save and apply

STEP16: Build now

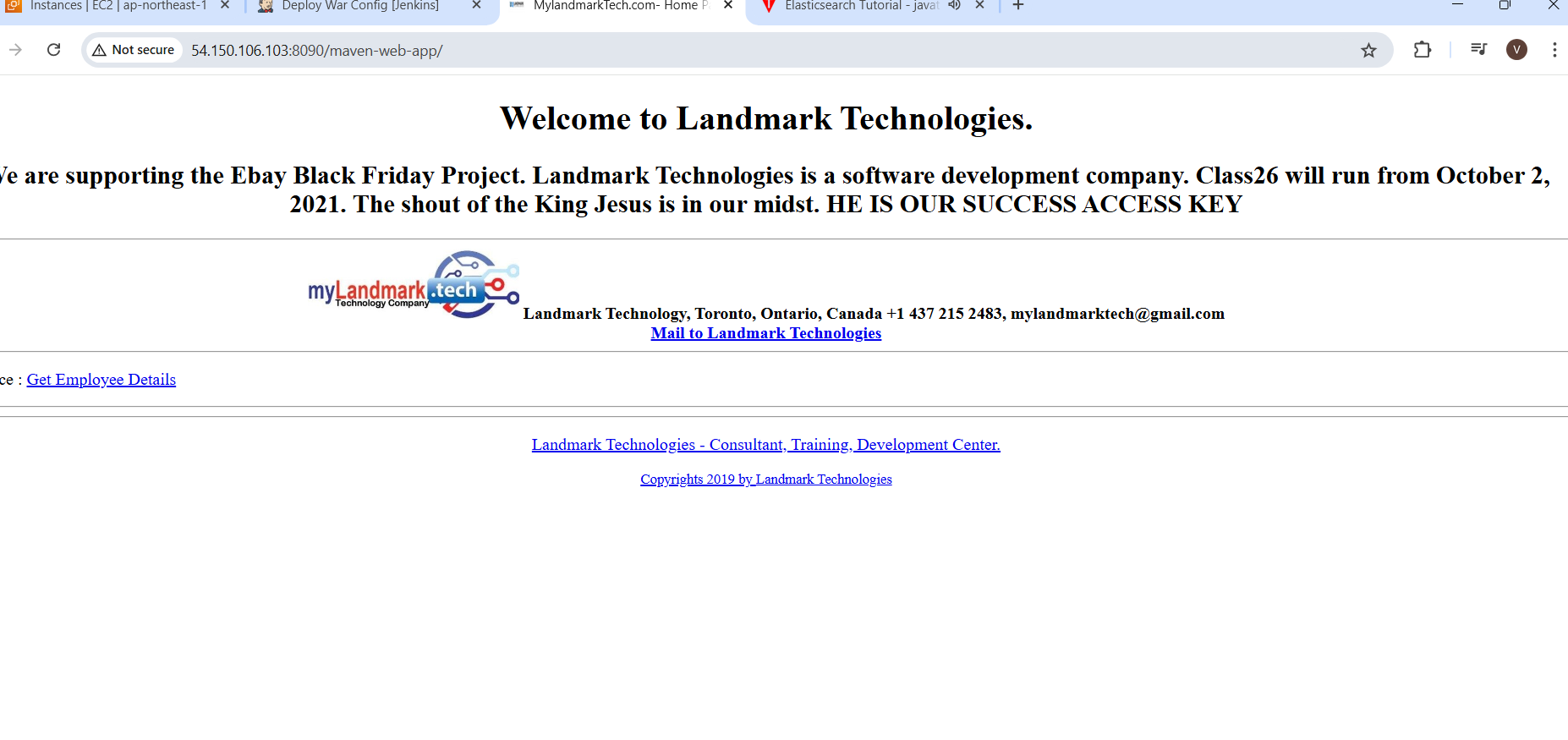
STEP17: Check the status of the project.



STEP18: After Successful completion of the build navigate to tomcat dashboard🡪Click on Manager-apps🡪It will asks for username and password that we have mentioned in the credentials section🡪 If our application gets succeed it gets reflected in the dashboard as shown in the figure.



Here we can see maven-web-app application that has been reflected. Just give a click on the displayed application of your dashboard. We can see the output in the dashboard.



Conclusion: We will pull the code from SCM(Git)🡪Integrate with Jenkins🡪 Jenkins gets interacts with maven as it is a build tool 🡪 Here we will build a WAR/JAR FILE 🡪After successful build of WAR/JAR file we will deploy the file in Tomcat server.