Kalaivani Ravishankar
Vani212008@gmail.com

Deployment of Java Application on Tomcat Apache Using Jenkins Pipeline

Deployment of Java Application on Tomcat Apache Using Pipeline

Introduction:

In this project I will apply the skills and knowledge which were developed throughout PG DO - CI/CD Pipeline with Jenkins. These include:

- **Working in AWS**
- **Building pipelines**
- **Git, GitHub,**
- **Jenkins,**
- **Tomcat Apache and**
- **Maven**

Application:

The application I used sample java application.

I used ec2 instance to create tomcat server and Jenkins server

Tools used:

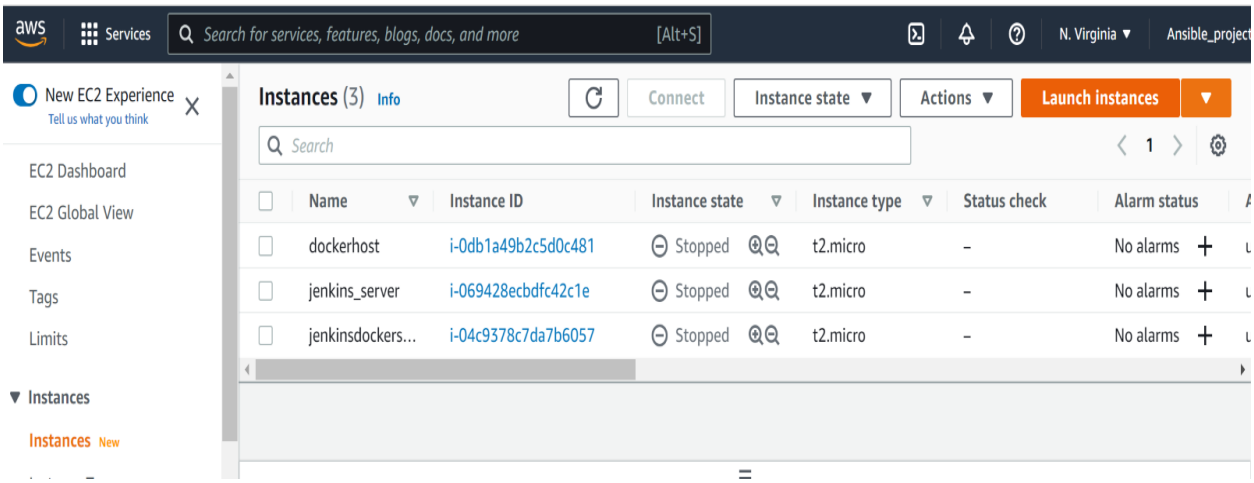
- **Git**
- **Maven**
- **Jenkins**
- **Tomcat Apache**

- **GitHub**

Steps to create ec2-instance:

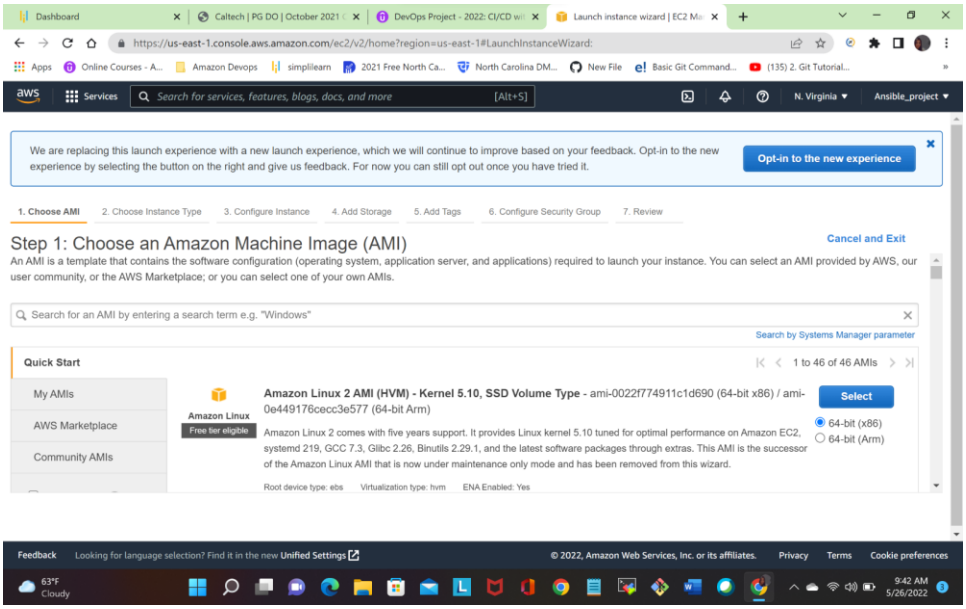
Step 1: Login to the aws console: -
I use this link https://console.aws.amazon.com/ec2/ to login to aws console account.

Step 2: Choose Launch Instance
First of all, click on 'Launch Instance' button shows in the below picture for launch/create new ec2 instance in aws:



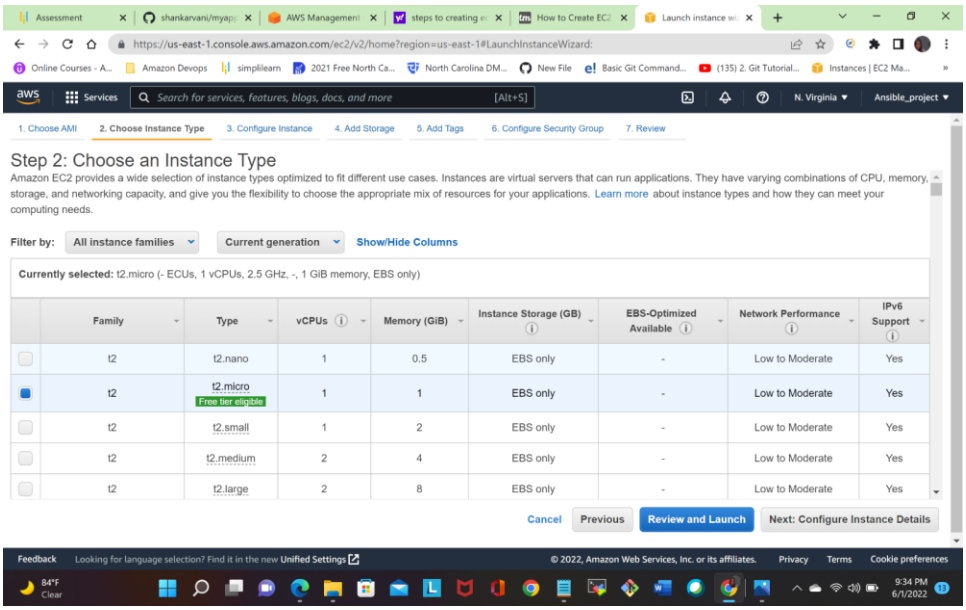
In this step, choose AMI (I chose Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type - ami-0022f774911c1d690 (64-bit x86) / ami-0e449176cecc3e577 (64-bit Arm)

Shown in the below fig

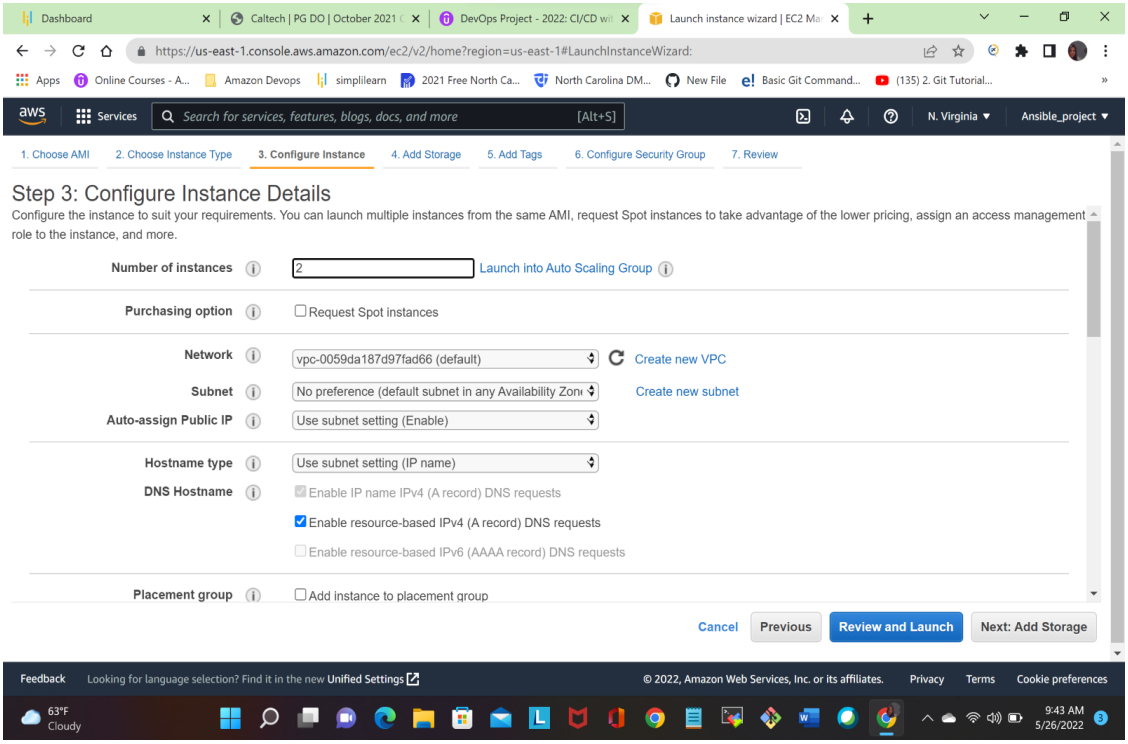


Step 4: Choose EC2 Instance Types

In this step, choose ec2 instance type shown in below picture:

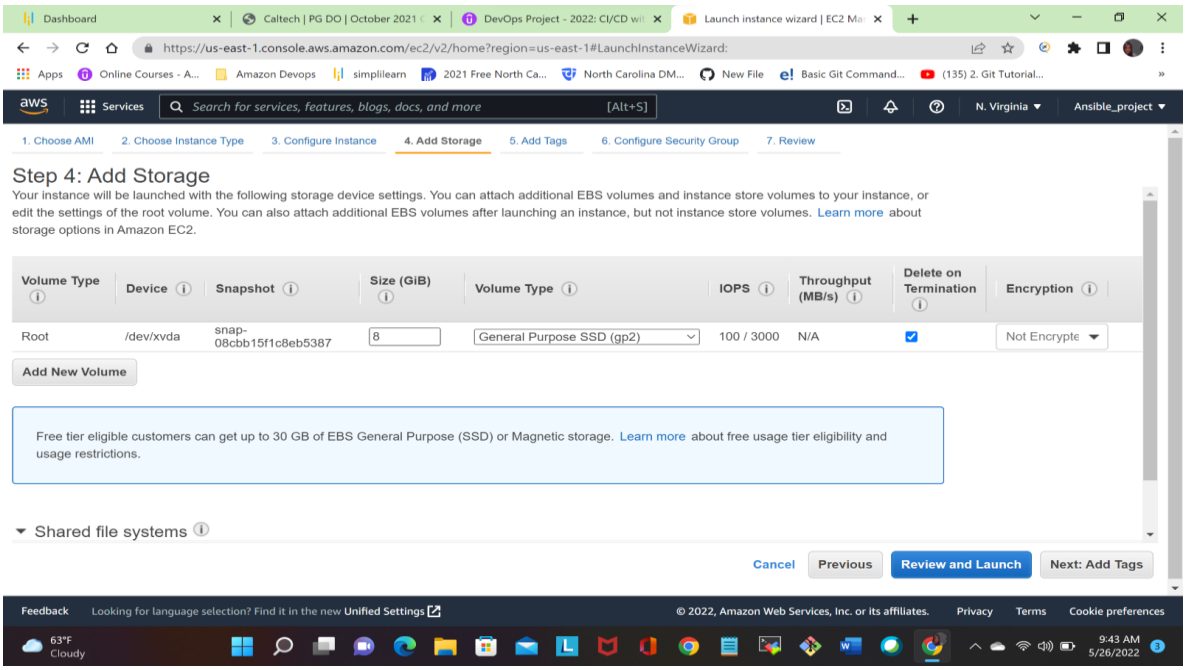


Step 5: Configure Instance Detail:



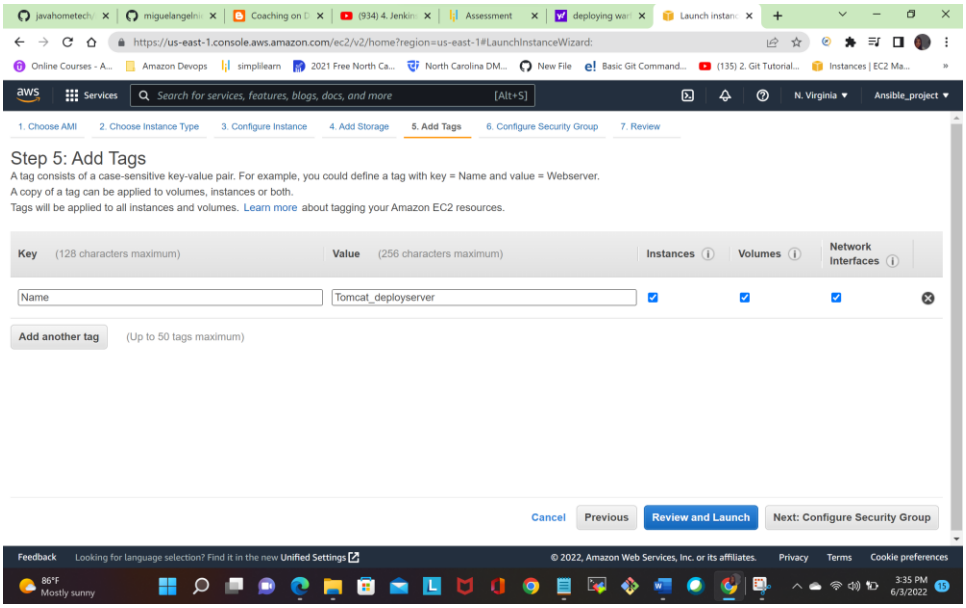
Step 6: Add Storage of Ec2 Instance

In this step, I chose storage. By default, ec2 t2-micro provide 8gb ssd. Shown in the below picture:



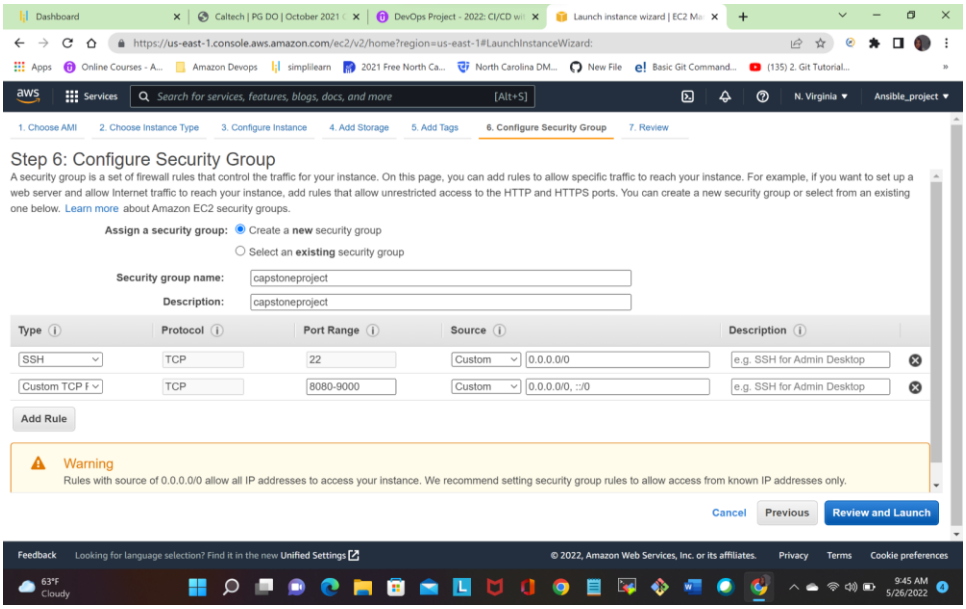
Step 7: Tag Instance of Ec2 Instance

In this step, I added tag of instance with a key-value pair



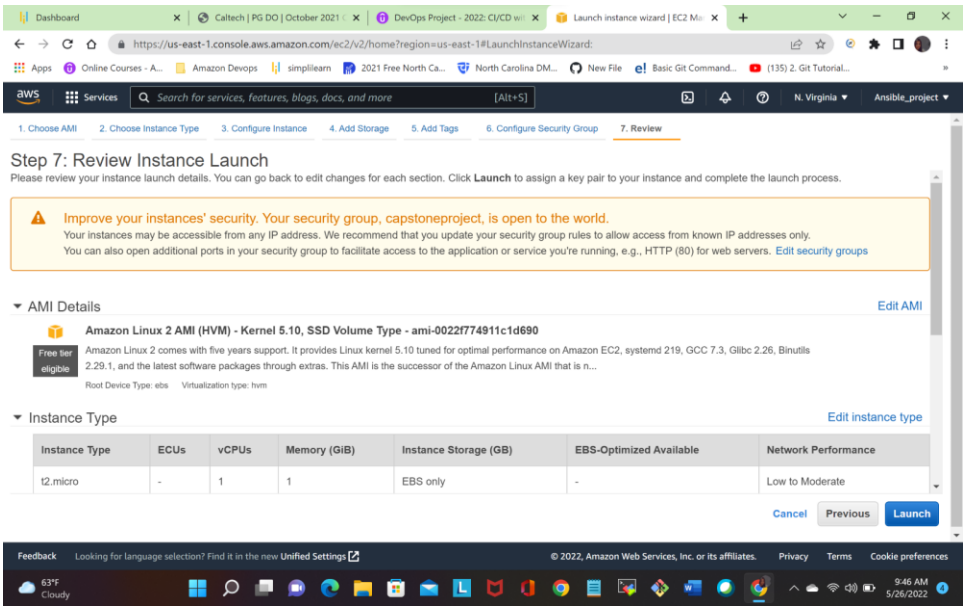
Step 8: Configure Security Groups

In this next step of configuring Security Groups



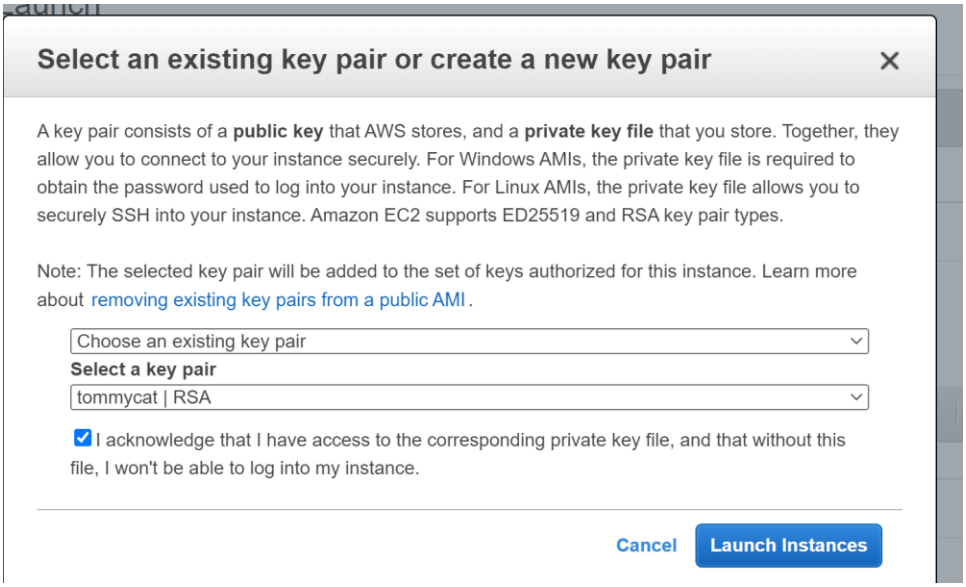
I created security group called capstone project and Defining protocols which i want enabled on my instance

Step 9: Once, the firewall rules are set- Review and launch



Step 10: Create Key-Pair for Instance Access:

Creating a keypair for ec2 instance, (I used existing keypair) create new key pair and add the name of this key. Then download and save it in your secured folder. Shown in below fig.

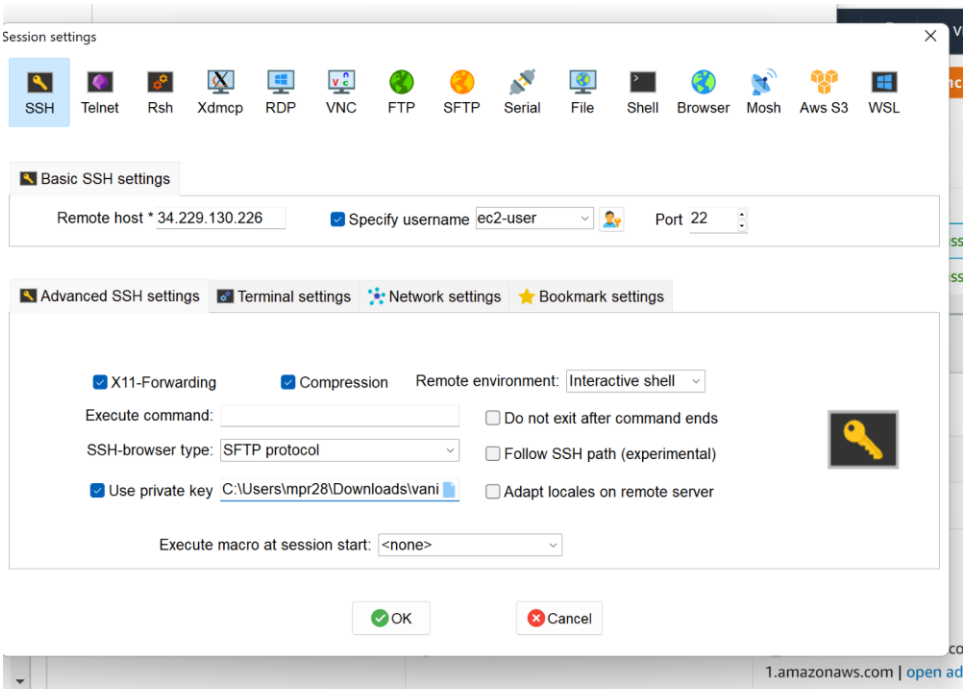


Use pem file in MobaXterm to connect ec2 machine in AWS: -

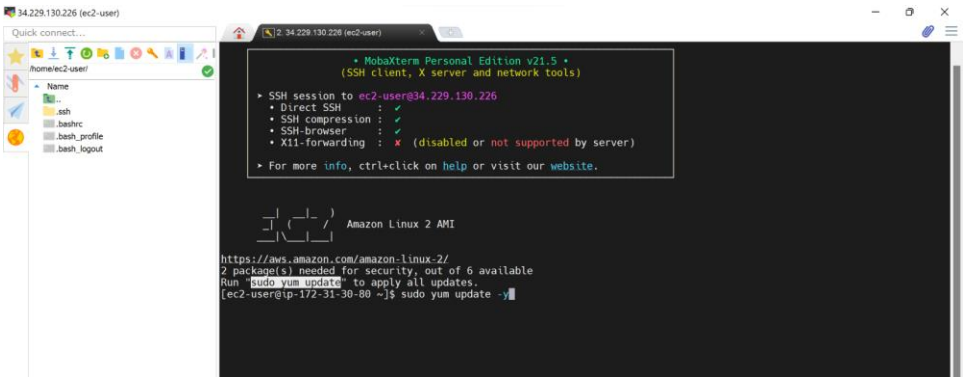
I used MobaXterm to connect ec2 machine. I downloaded mobaxterm .go to the session settings to create a new session, fill up the remote host and username.

And click advance setting and select use private key, select your private key

As shown in the below fig



After clicking ok it taking into page shown in below fig



I created two ec2 instance machine (Jenkins and Tomcat_deployserver)

Install Git on Jenkins server

Using following command

```
yum install git -y
```

```
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-19-44 ~]$ sudo su -
Last login: Fri Jun  3 15:31:16 UTC 2022 on pts/0
[root@ip-172-31-19-44 ~]# git --version
git version 2.32.0
[root@ip-172-31-19-44 ~]#
```

Install maven

Download maven packages https://maven.apache.org/download.cgi onto Jenkins server. In this case, I am using /opt/maven as my installation directory

```
[root@ip-172-31-19-44 ~]# mvn -v
Apache Maven 3.8.5 (3599d3414f046de2324203b78ddcf9b5e4388aa0)
Maven home: /opt/apache-maven-3.8.5
Java version: 1.8.0_312, vendor: Red Hat, Inc., runtime: /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-1.amzn2.0.2.x86_64/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.10.112-108.499.amzn2.x86_64", arch: "amd64", family: "unix"
[root@ip-172-31-19-44 ~]#
```

And I used following command to move maven to opt directory

```
cp –R apache-maven-3.8.5 /opt
```

I used wget command to download maven (wget is used to download files from an HTTP/HTTPS or FTP server)

wget https://dlcdn.apache.org/maven/maven-3/3.8.5/binaries/apache-maven-3.8.5-bin.tar.gz

Next step is to extract the archive using following command

```
tar xzvf apache-maven-3.8.5-bin.tar.gz
```

Maven home path in bash profile

JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-1.amzn2.0.2.x86_64

MAVEN_HOME=/opt/apache-maven-3.8.5

M2=/opt/apache-maven-3.8.5/bin

PATH=$PATH:$HOME/bin:$JAVA_HOME:$MAVEN_HOME:$M2

Java

http://openjdk.java.net/install/

Install Java in Linux

yum install `java-1.8.0-openjdk-devel`

Java is installed in usr/lib directory

Set Java Home Path in Bash profile

JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-1.amzn2.0.2.x86_64

PATH=$PATH:$HOME/bin:$JAVA_HOME:

```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup program

JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-1.amzn2.0.2.x86_64

MAVEN_HOME=/opt/apache-maven-3.8.5
M2=/opt/apache-maven-3.8.5/bin
PATH=$PATH:$HOME/bin:$JAVA_HOME:$MAVEN_HOME:$M2

export PATH
~
~
~
~
~
~
~
```

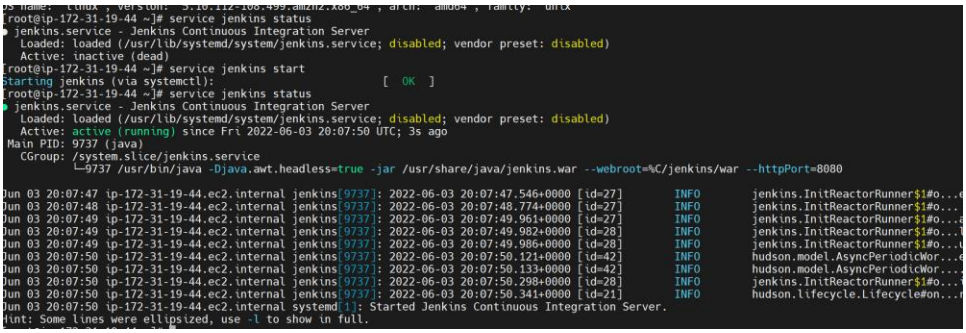
Install Jenkins:-

install Jenkins using the rpm or by setting up the repo

I downloaded the latest Version of Jenkins form https://pkg.jenkins.io/redhat-stable/ and install

`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.key`
`amazon-linux-extras install epel`
`after downloading Start the Jenkins services using following command`

`Service jenkins start,,shown in following fig`



Accessing Jenkins

By default, jenkins runs at port `8080`, You can access jenkins at

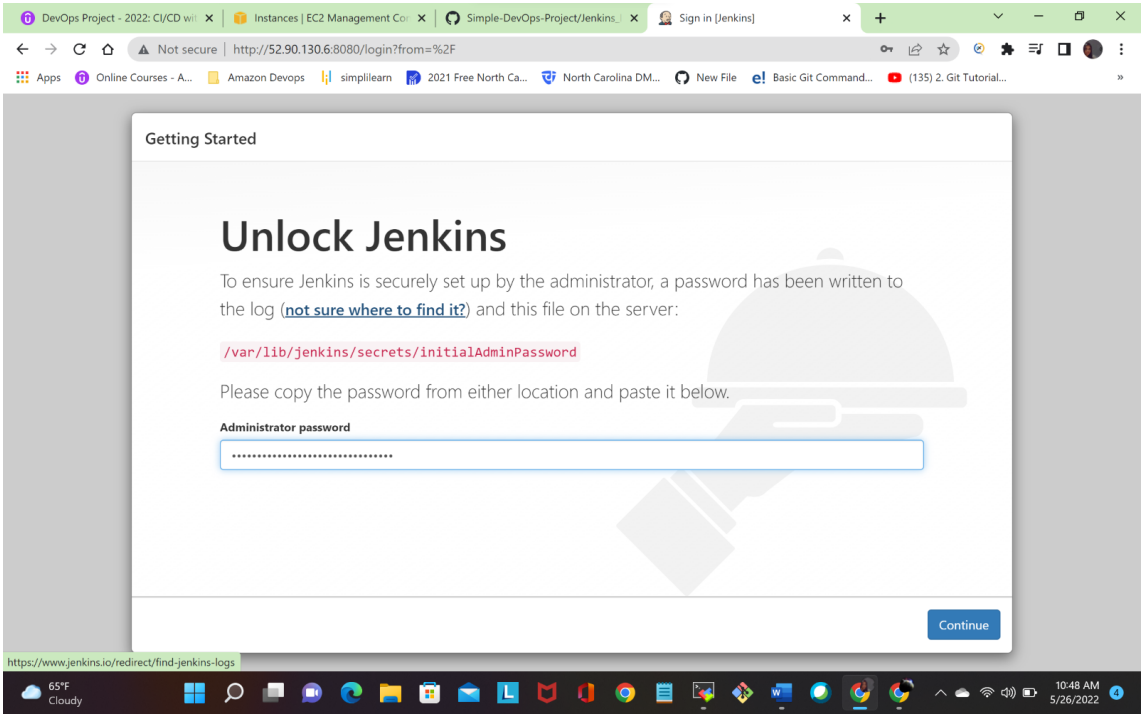
`http://YOUR-SERVER-PUBLIC-IP:8080`

Configure Jenkins:

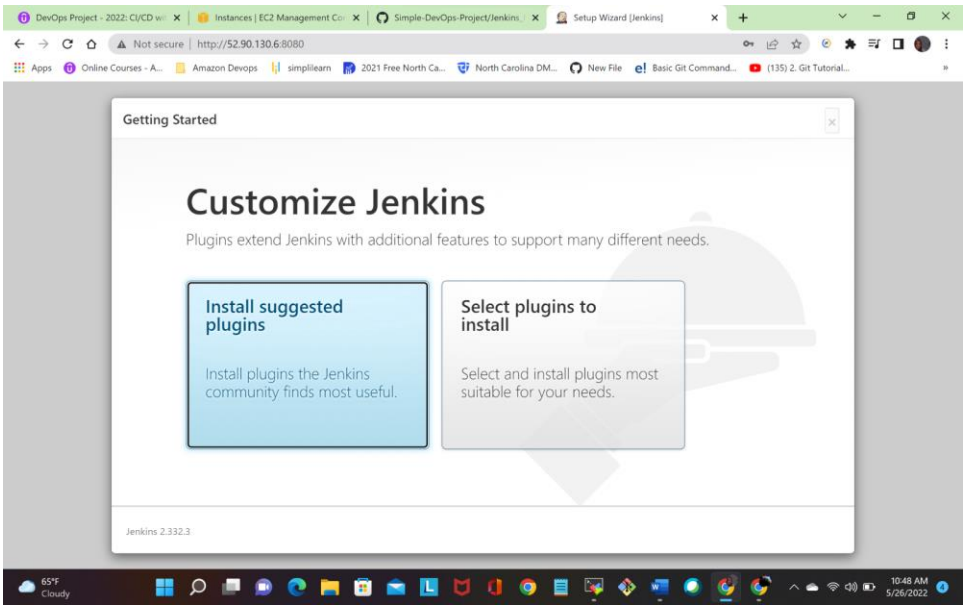
Grab the default password

Password Location:/var/lib/jenkins/secrets/initialAdminPassword

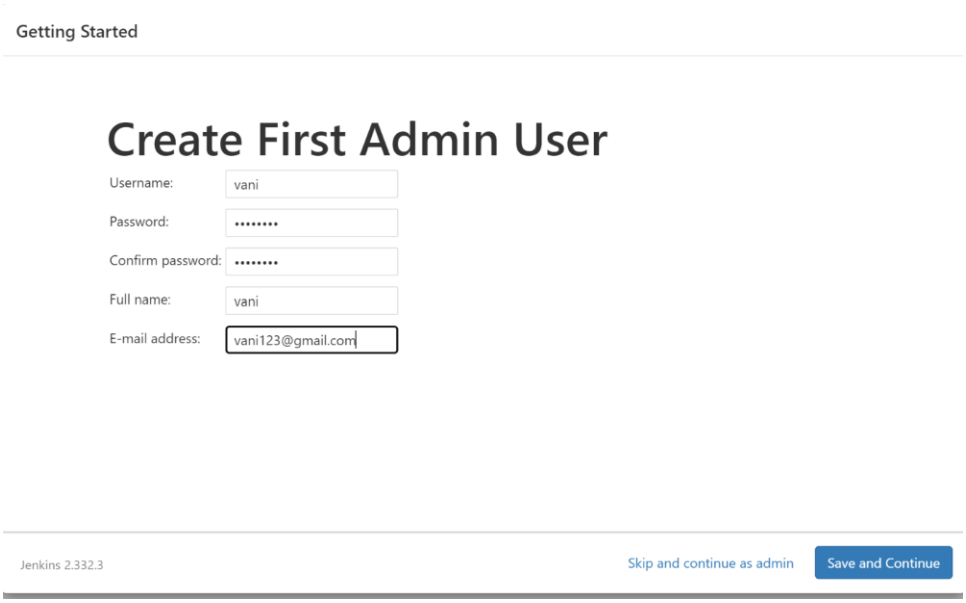
Paste it. Shown in below fig



Choose suggested plugins. Shown in below fig

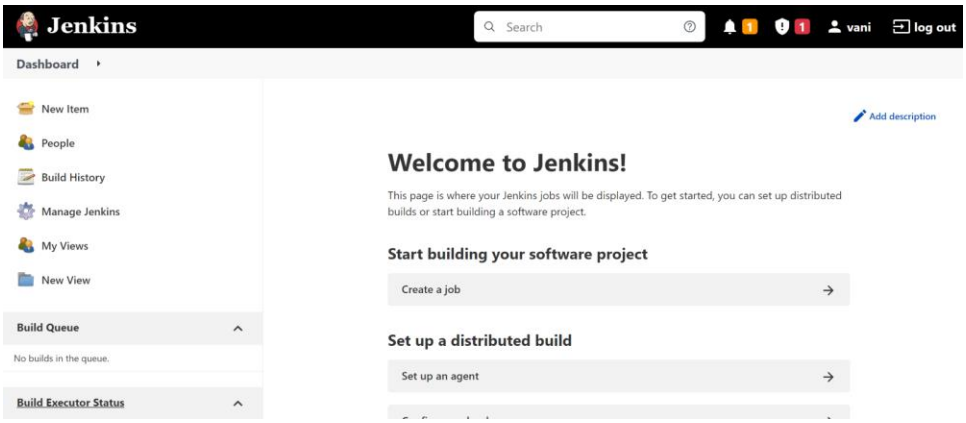


And enter username, password email address shown in below fig



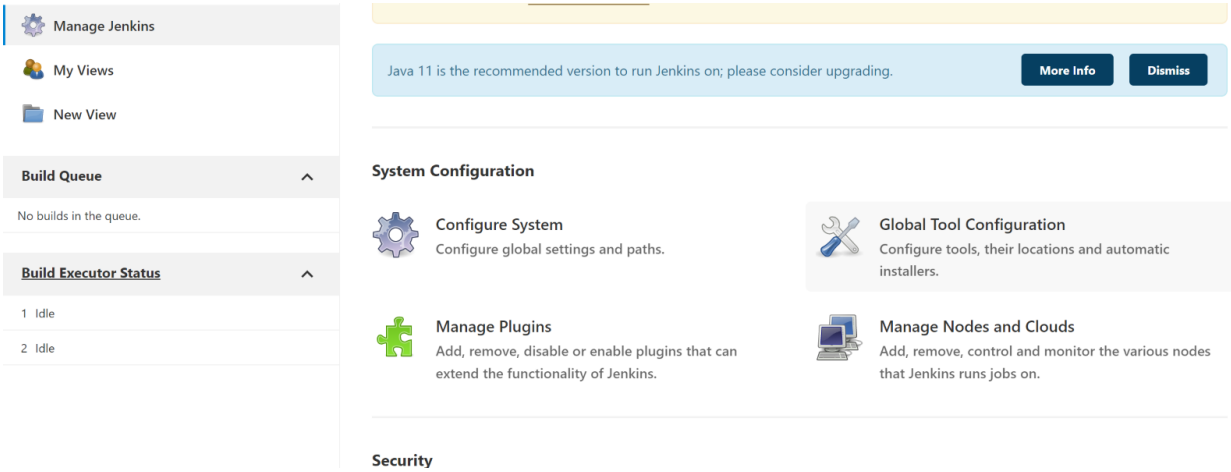
Click save and continue.it taken you into jenkins dashboard

Jenkins Dashboard: -

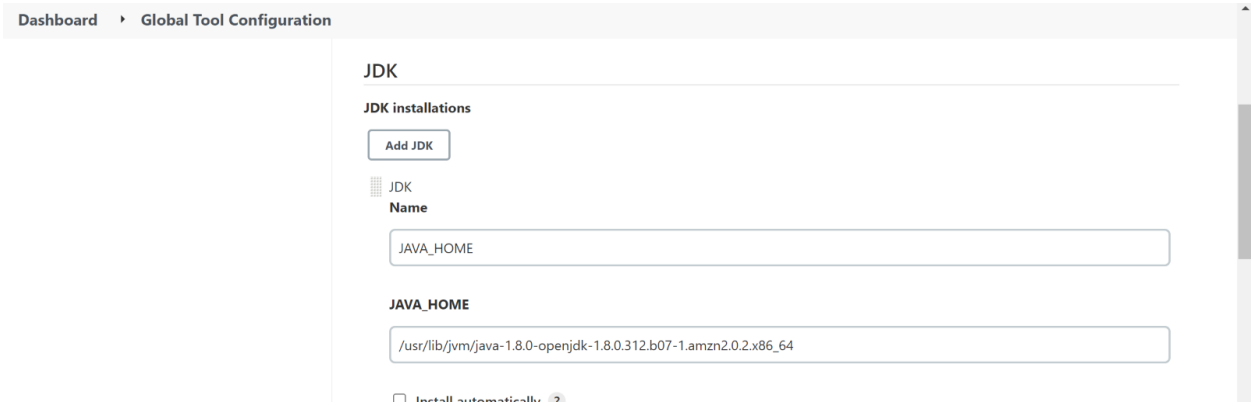


Configure java and maven path

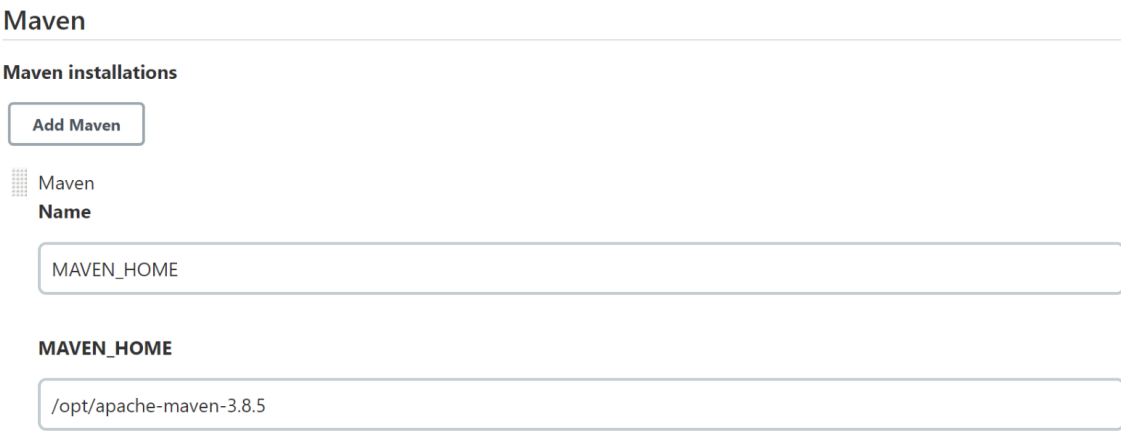
Manage Jenkins > Global Tool Configuration > JDK



Click add JDK and give JDK name as JAVA_HOME and path of JDK

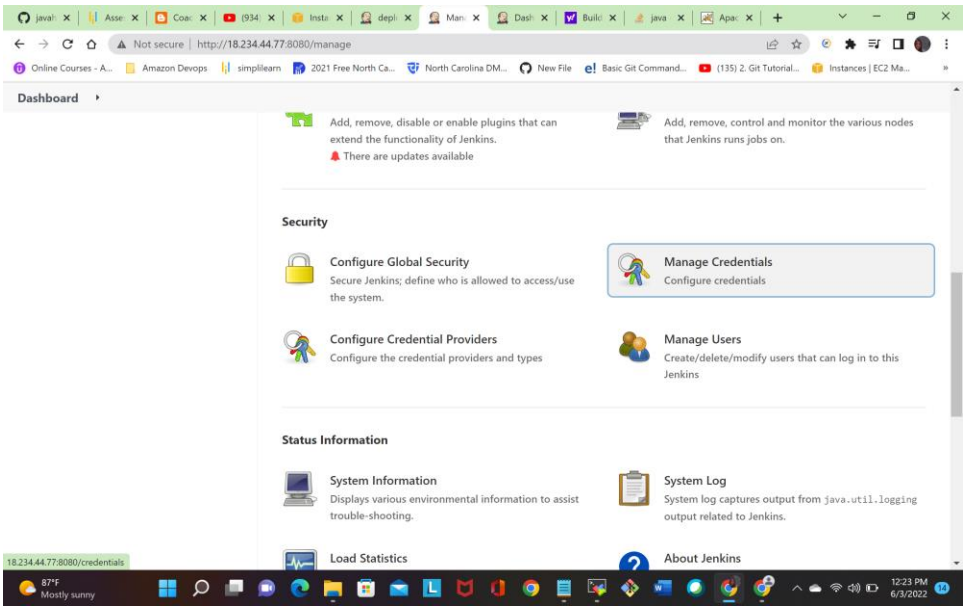


Next configure maven path, click maven give name for maven as MAVEN_HOME And path of maven



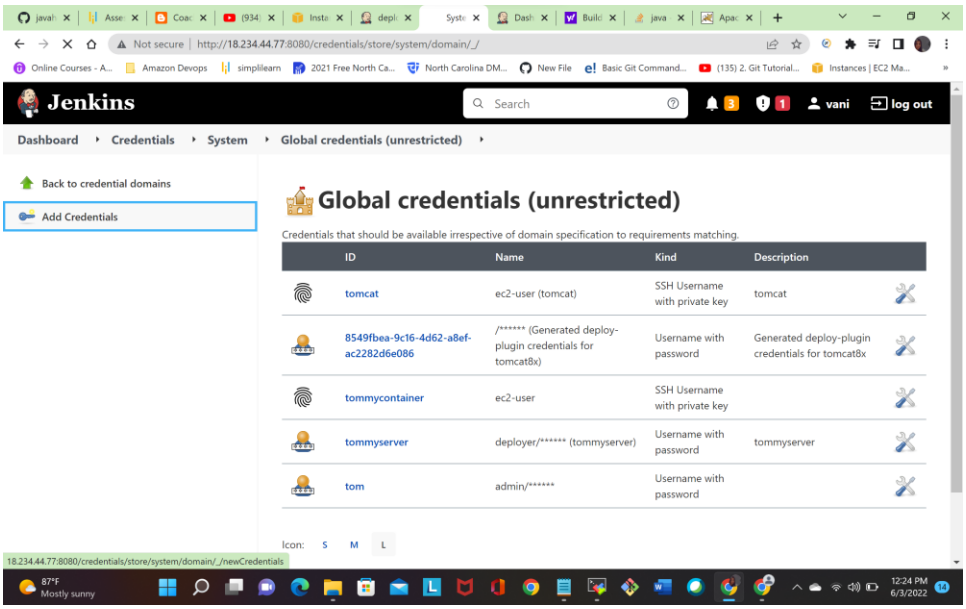
Configure Credentials

Next, I configured Ssh Agent(click--->manage jenkins-->manage credentials to adding private key to connect between tomcatdeployserver and Jenkins server) shown in following fig.

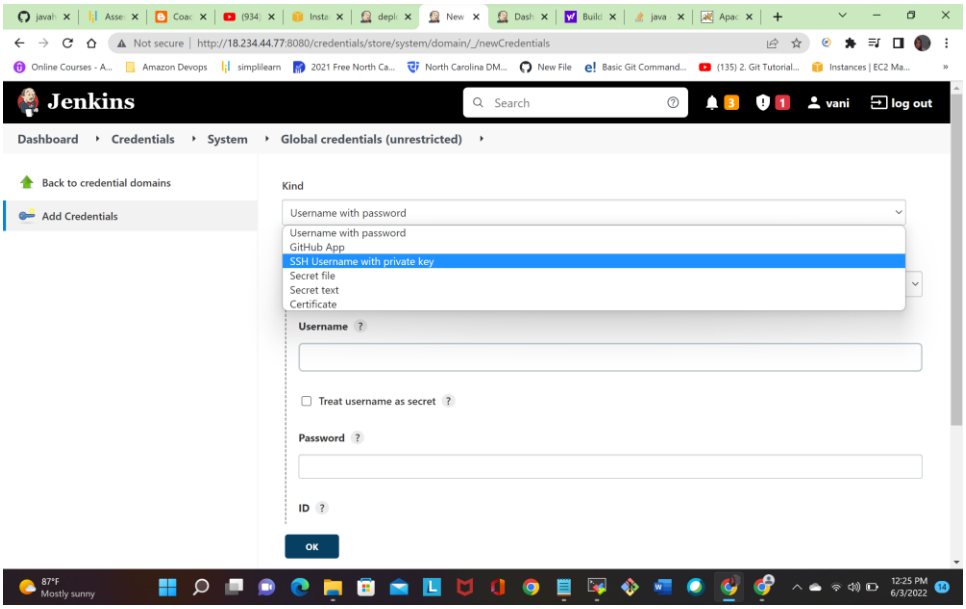


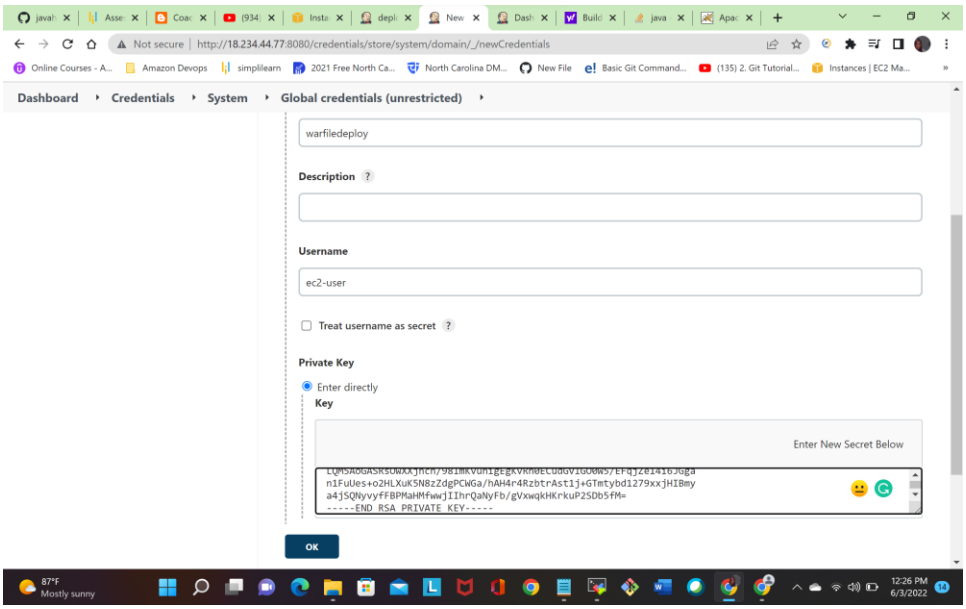
Then click Add credentials to add key

Shown in following fig



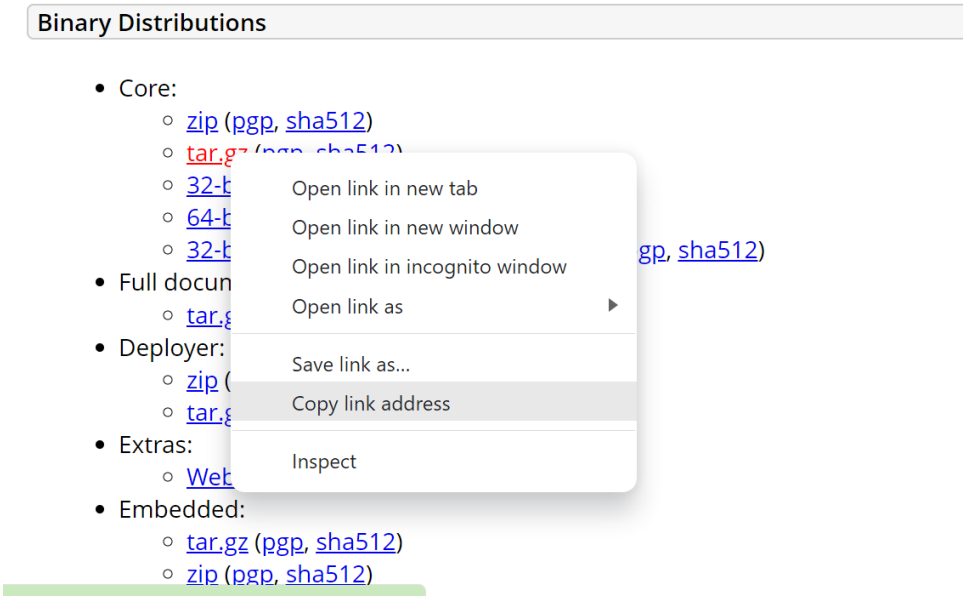
Select SSH USERNAME with private key and fill the details which required





Once I finished jenkins configuration setting I moved to Tomcatdeployserver to install tomcat8 in ec2-instance

Go to browser –> https://tomcat.apache.org/download-90.cgi (to download tomcat9) –> Copy ***tar.gz*** from core section.

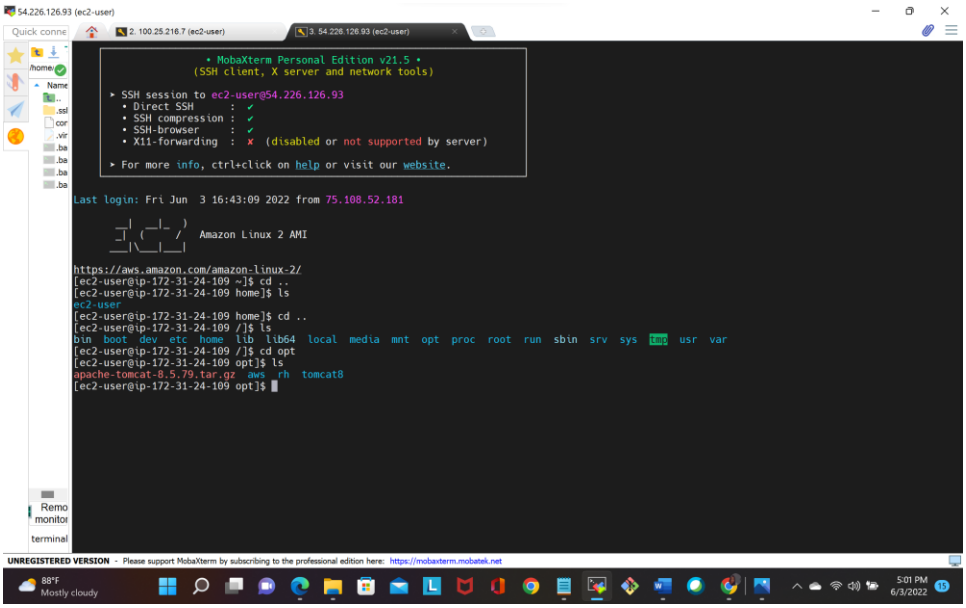


Once I copied binary, I extract .i downloaded tomcat8 in opt directory and I renamed as tomcat8

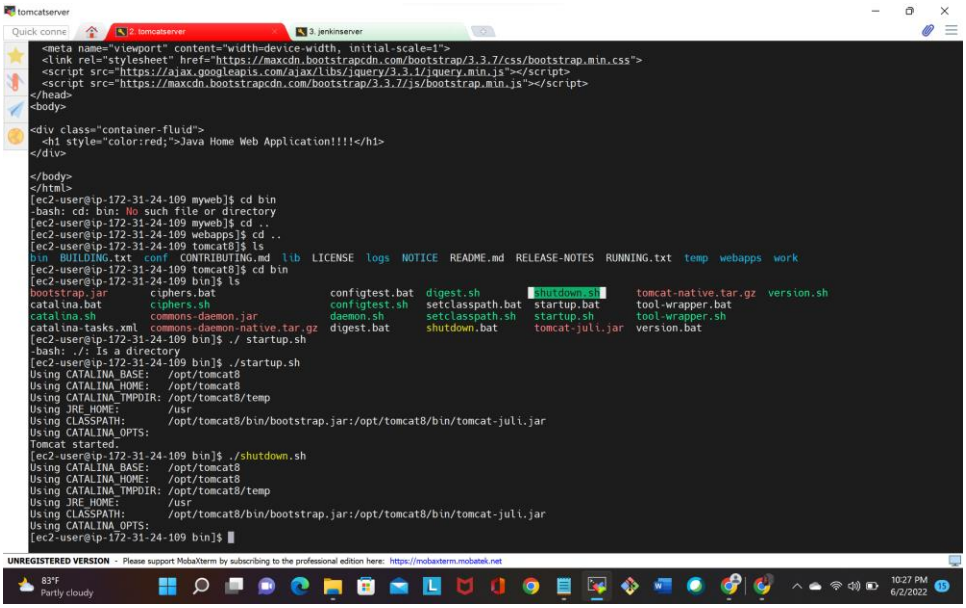
I gave full ownership to ec2-user for tomcat8 folder using following command

Chown –R ec2-user:ec2user tomcat8/

```
[ec2-user@ip-172-31-24-109 tomcat8]$ cd ..
[ec2-user@ip-172-31-24-109 opt]$ ll -ll
total 10344
-rw-r--r-- 1 root     root     10592120 May 16 15:49 apache-tomcat-8.5.79.tar.gz
drwxr-xr-x 4 root     root            33 Apr 28 19:54 aws
drwxr-xr-x 2 root     root             6 Aug 16  2018 rh
drwxr-xr-x 9 ec2-user ec2-user       220 Jun  1 14:45 tomcat8
[ec2-user@ip-172-31-24-109 opt]$
```



Starting tomcat using ./startup.sh



I finished all configuration, then I navigate to jenkins dashboard to deploy war file in tomcat

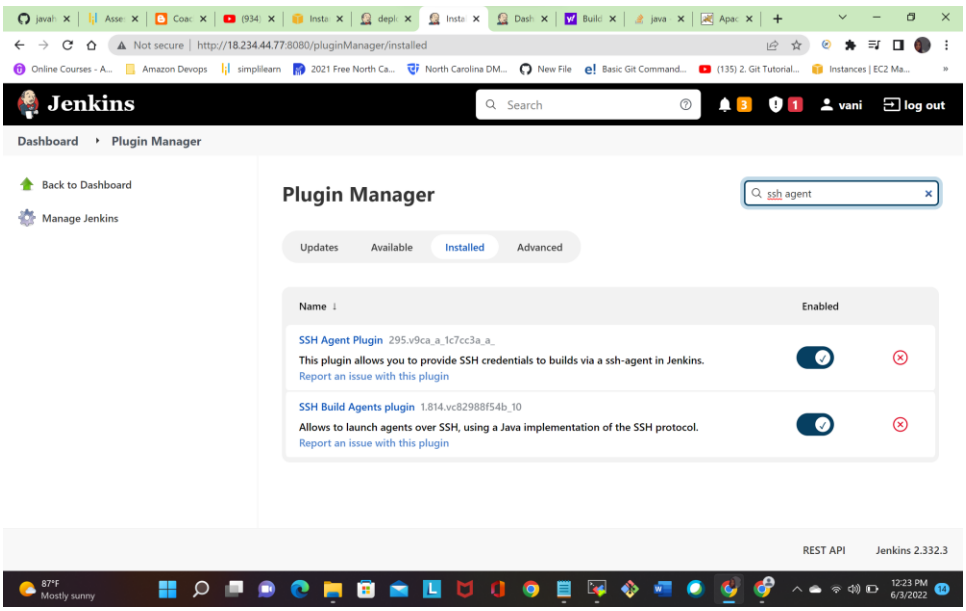
Integration with Jenkins Plugins need to install in jenkins:

Plugins need to install in jenkins:

I installed following plugins

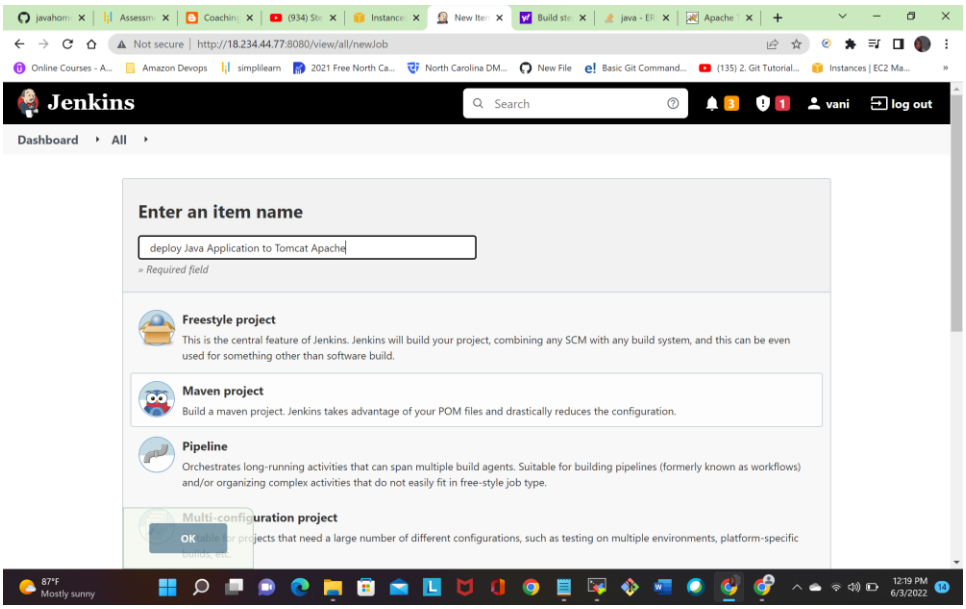
Go to manage jenkins--->plugin manager-->available section and search for ssh agent plugin and Pipeline: Declarative

-->click install without restart

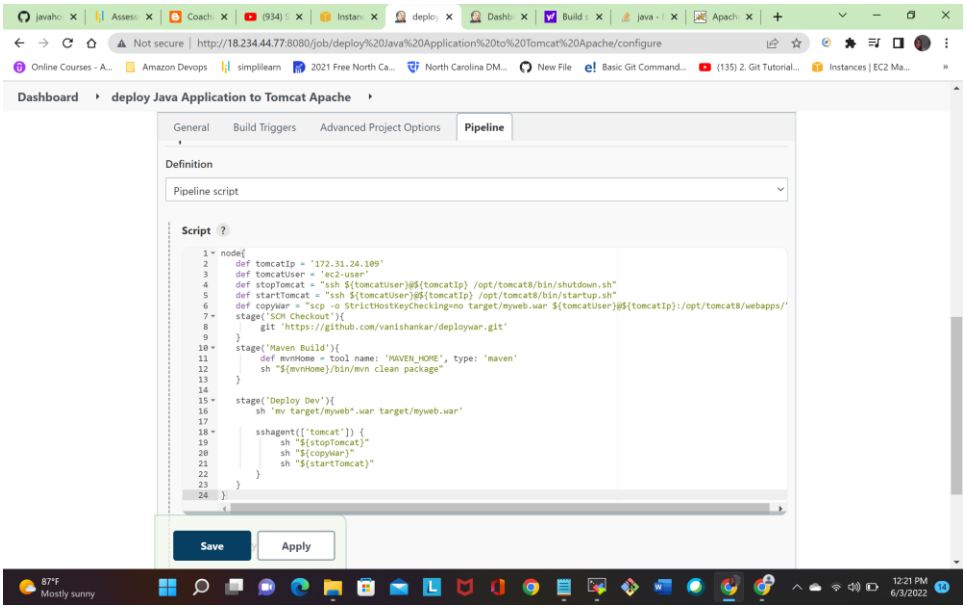


Next, I created a new job for deploying java application in tomcat server

Navigate to jenkins Dashboard click -->New Item-->give name of the job (I gave deploy Java Application on Tomcat Apache Using Pipeline). Then select pipeline and click ok



Then click pipeline tab and select pipeline script in dropdown box



Write the pipeline script using pipeline syntax shown in foll fig

Steps

Sample Step

sshagent: SSH Agent

sshagent ?

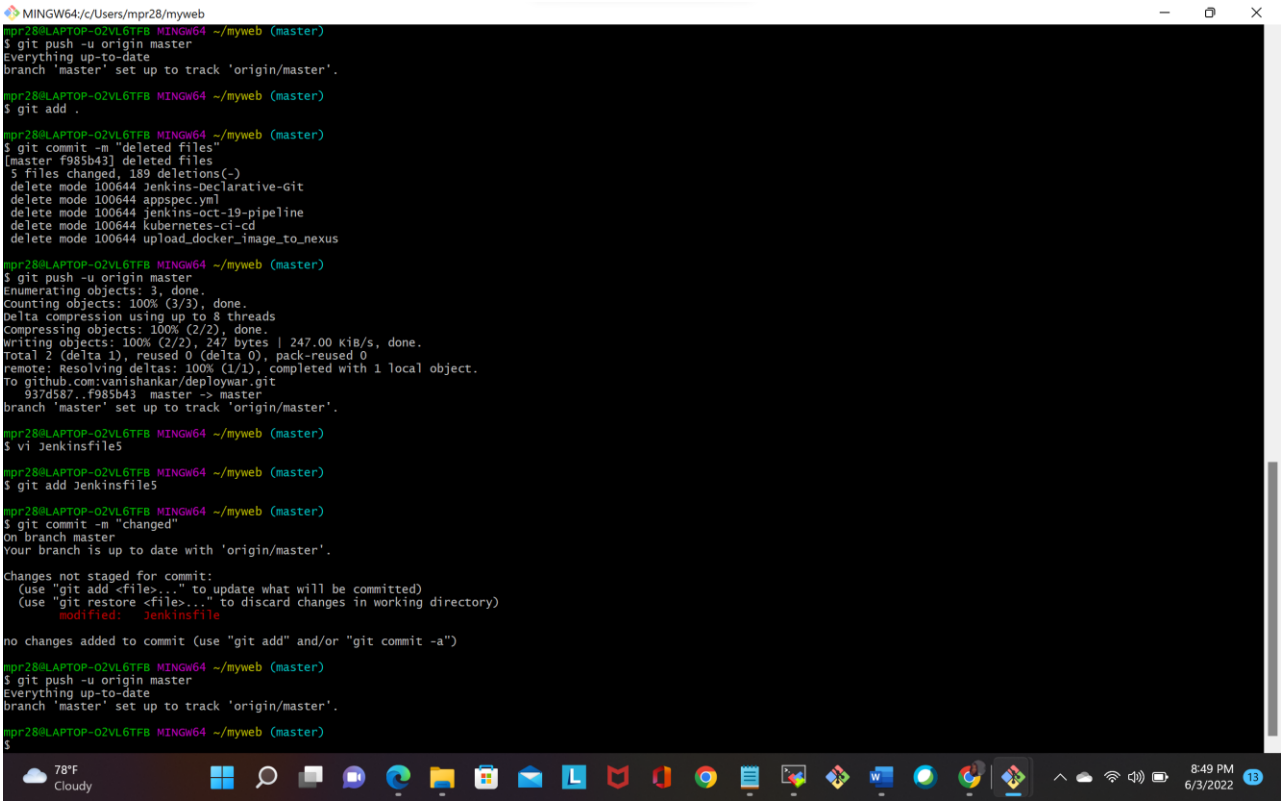
ec2-user Add ?

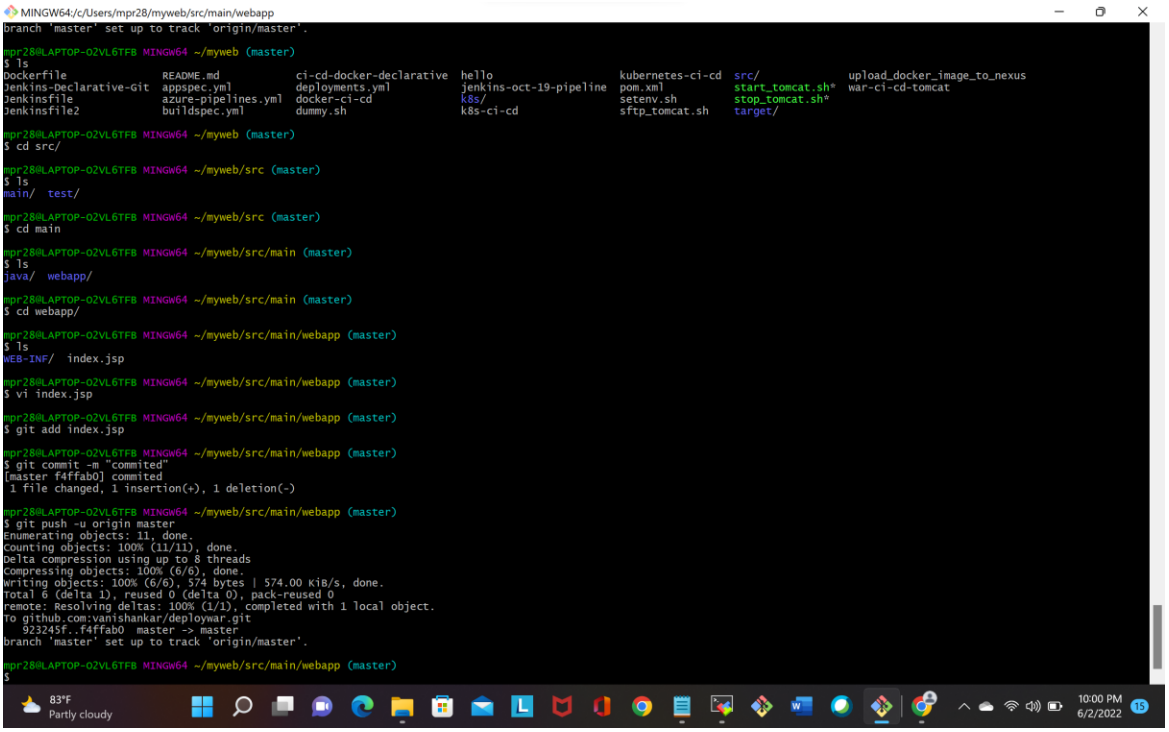
Ignore missing credentials ?

Generate Pipeline Script

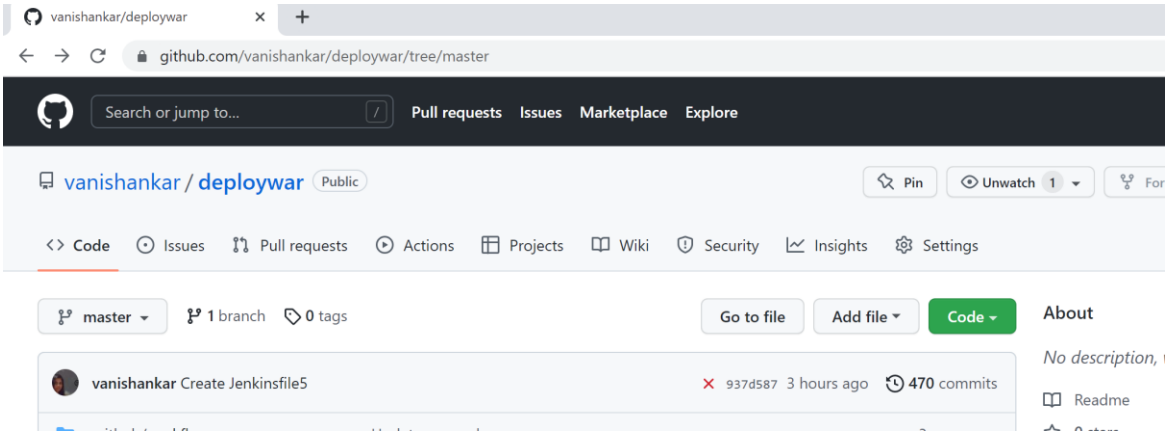
```
sshagent(['tommycontainer']) {
   // some block
}
```

After creating jenkins file I pushed into GitHub





GitHub repo



Create a JenkinsFile

The Jenkinsfile in the project will do the much of the work inside

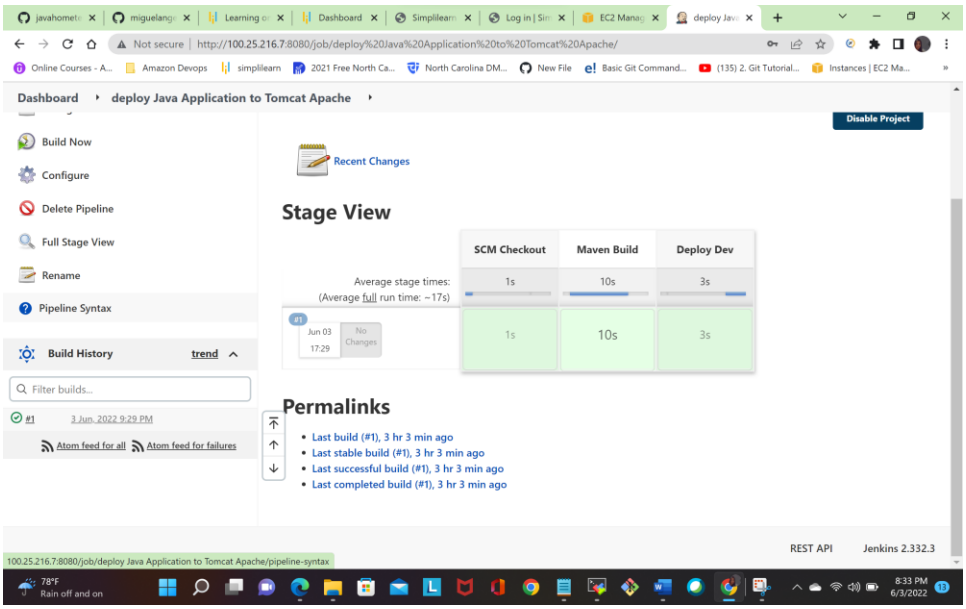
1. Checkout from Git
2. Execute maven and packages the WAR
3. Runs test cases and presents a report
4. Deploy warfile,and start and stop the tomcat server

This below script is used to deploy war file into tomcat

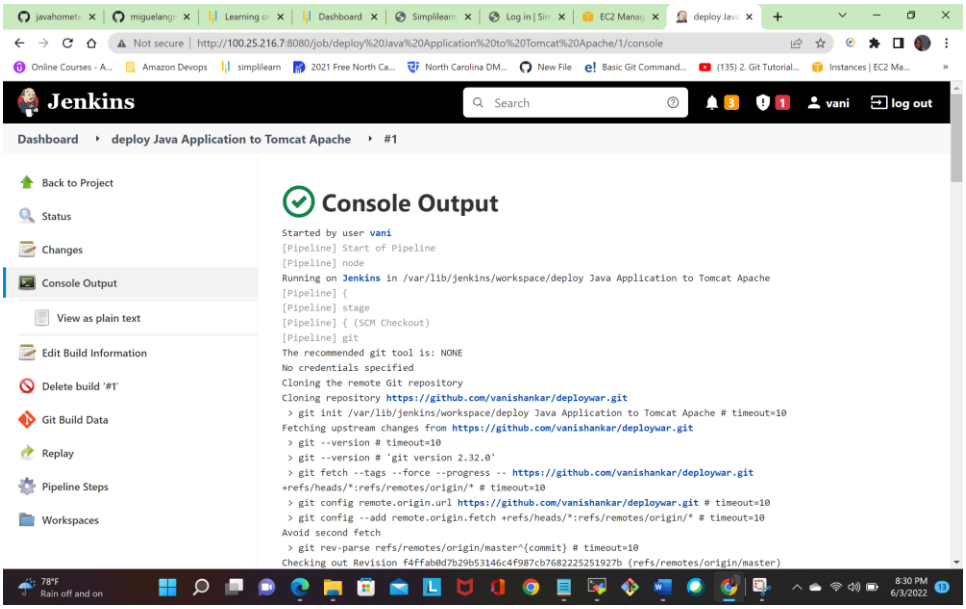
```
node {
  def tomcatIp = '172.31.24.109'
  def tomcatUser = 'ec2-user'
  def stopTomcat = "ssh ${tomcatUser}@${tomcatIp} /opt/tomcat8/bin/shutdown.sh"
  def startTomcat = "ssh ${tomcatUser}@${tomcatIp} /opt/tomcat8/bin/startup.sh"
  def copyWar = "scp -o StrictHostKeyChecking=no target/myweb.war ${tomcatUser}@${tomcatIp}:/opt/tomcat8/webapps/"
  stage ('SCM Checkout'){
      git 'https://github.com/vanishankar/deploywar.git'
  }
  stage('Maven Build'){
      def mvnHome = tool name: 'MAVEN_HOME', type: 'maven'
sh "${mvnHome}/bin/mvn clean package"
  }

  stage('Deploy Dev'){
  sh 'mv target/myweb*.war target/myweb.war'

       sshagent(['tomcat']) {
sh "${stopTomcat}"
sh "${copyWar}"
sh "${startTomcat}"
  }
  }
}
```

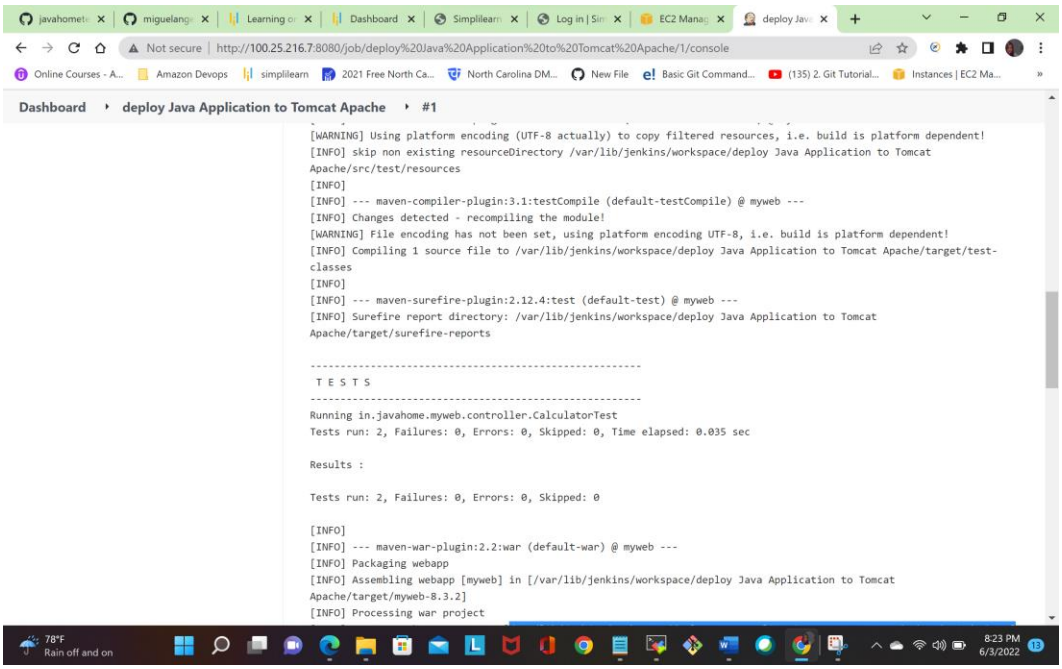
Click Save & build now.

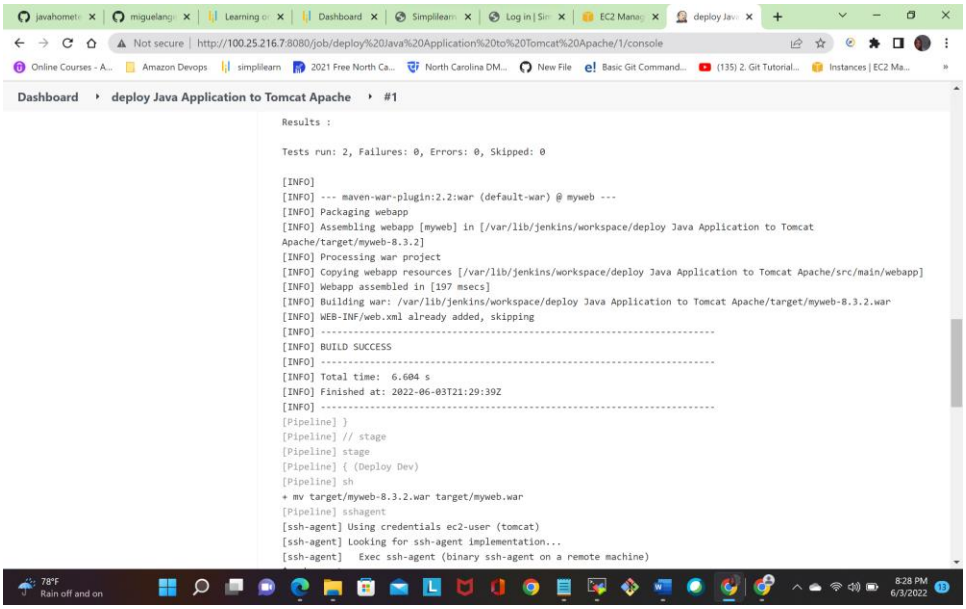
Build Now fig. shown below



Console Output Result



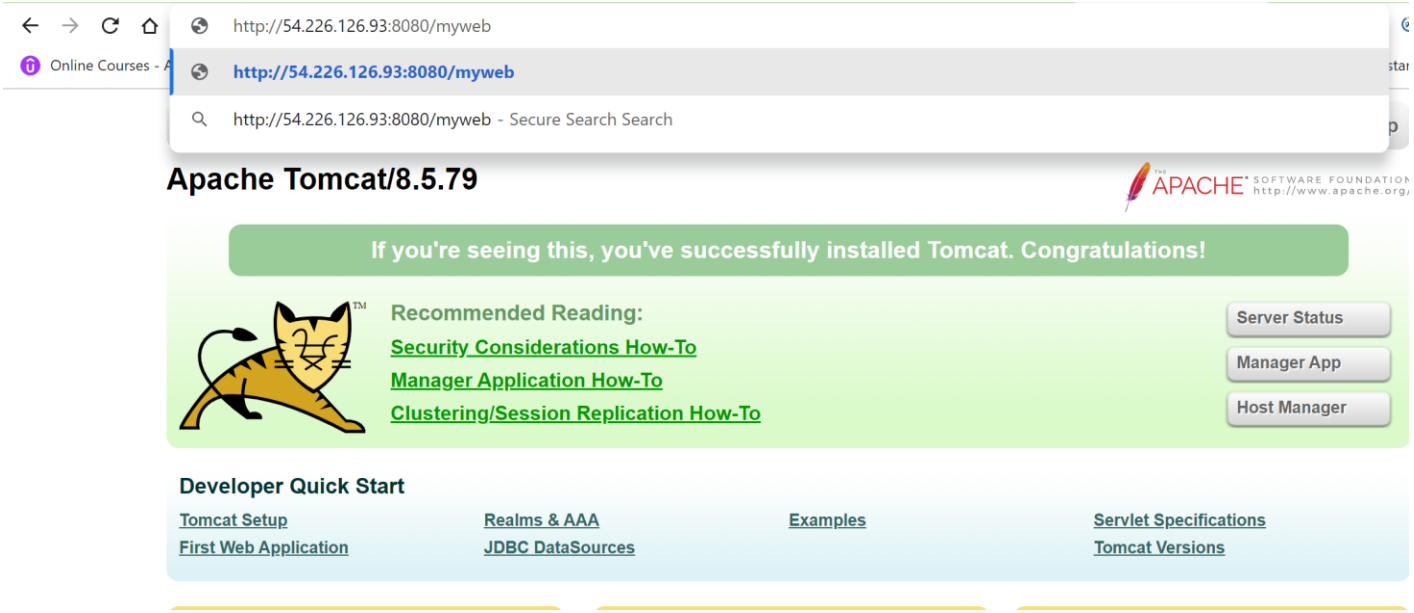

Not secure | http://100.25.216.7:8080/job/deploy%20Java%20Application%20to%20Tomcat%20Apache/1/console
Dashboard ▸ deploy Java Application to Tomcat Apache ▸ #1
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ myweb ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache/target/test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ myweb ---
[INFO] Surefire report directory: /var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache/target/surefire-reports
T E S T S
Running in.javahome.myweb.controller.CalculatorTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.035 sec
Results :
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] --- maven-war-plugin:2.2:war (default-war) @ myweb ---
[INFO] Packaging webapp
[INFO] Assembling webapp [myweb] in [/var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache/target/myweb-8.3.2]
[INFO] Processing war project
78°F
Rain off and on
8:23 PM
6/3/2022


Not secure | http://100.25.216.7:8080/job/deploy%20Java%20Application%20to%20Tomcat%20Apache/1/console
Dashboard ▸ deploy Java Application to Tomcat Apache ▸ #1
Results :
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] --- maven-war-plugin:2.2:war (default-war) @ myweb ---
[INFO] Packaging webapp
[INFO] Assembling webapp [myweb] in [/var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache/target/myweb-8.3.2]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache/src/main/webapp]
[INFO] Webapp assembled in [197 msecs]
[INFO] Building war: /var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache/target/myweb-8.3.2.war
[INFO] WEB-INF/web.xml already added, skipping
[INFO] BUILD SUCCESS
[INFO] Total time: 6.604 s
[INFO] Finished at: 2022-06-03T21:29:39Z
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy Dev)
[Pipeline] sh
+ mv target/myweb-8.3.2.war target/myweb.war
[Pipeline] sshagent
[ssh-agent] Using credentials ec2-user (tomcat)
[ssh-agent] Looking for ssh-agent implementation...
[ssh-agent] Exec ssh-agent (binary ssh-agent on a remote machine)
78°F
Rain off and on
8:28 PM
6/3/2022

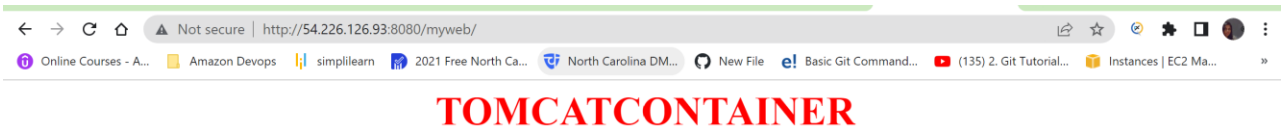
```
[Pipeline] sh
+ scp -o StrictHostKeyChecking=no target/myweb.war ec2-user@172.31.24.109:/opt/tomcat8/webapps/
[Pipeline] sh
+ ssh ec2-user@172.31.24.109 /opt/tomcat8/bin/startup.sh
Tomcat started.
[Pipeline] }
$ ssh-agent -k
unset SSH_AUTH_SOCK;
unset SSH_AGENT_PID;
echo Agent pid 23073 killed;
[ssh-agent] Stopped.
[Pipeline] // sshagent
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Accessing from Browser

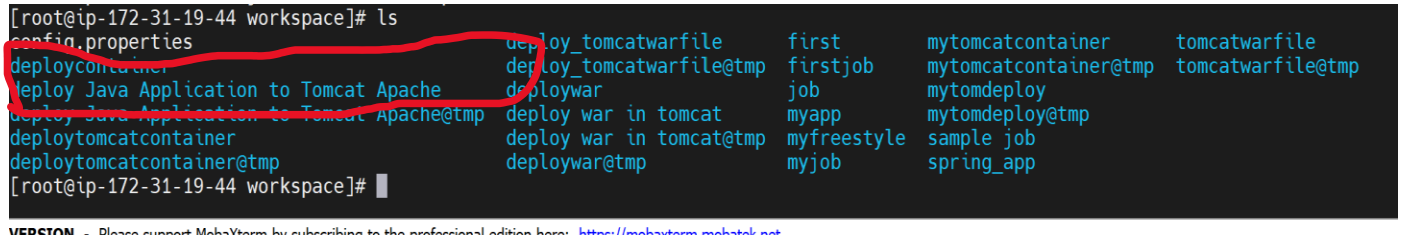
Go to browser: **Error! Hyperlink reference not valid.**



Finally deployed java application in tomcat server



And we can see in jenkins workspace in our jenkins project



Conclusion:

Jenkins pipeline is implemented as a code which allows several users to edit and execute the pipeline process. Jenkins Pipelines support big projects. We can run many jobs, and even use pipelines in a loop.so it is a best way to deploy application using jenkins pipeline

==========