

Kalaivani Ravishankar
Vani212008@gmail.com

Deployment of Java Application on Tomcat Apache Using Jenkins Pipeline

TOPICS	PAGE NO
Introduction	2
Application	2
Tools used	2
Steps to create ec2-instance:	3
MobaXterm	8
Git	9
maven	9
Java	10
Home Path in Bash profile	10
Install Jenkins	10
Configure java and maven path	13
Configure Credentials	14
Install tomcat8	16
Integration with Jenkins	18

Plugins need to install in Jenkins	18
JenkinsFile	21
Console output	25
Accessing from Browser	25
Jenkins workspace	26
Conclusion	26

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:

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	dockerhost	i-0db1a49b2c5d0c481	Stopped	t2.micro	-	No alarms +
<input type="checkbox"/>	jenkins_server	i-069428ecbdfc42c1e	Stopped	t2.micro	-	No alarms +
<input type="checkbox"/>	jenkinsdockers...	i-04c9378c7da7b6057	Stopped	t2.micro	-	No alarms +

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

The screenshot shows the AWS Launch Instance Wizard interface. At the top, there is a message: "We are replacing this launch experience with a new launch experience, which we will continue to improve based on your feedback. Opt-in to the new experience by selecting the button on the right and give us feedback. For now you can still opt out once you have tried it." Below this is a "Opt-in to the new experience" button.

The main area is titled "Step 1: Choose an Amazon Machine Image (AMI)". It says: "An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace, or you can select one of your own AMIs." There is a search bar with placeholder text "Search for an AMI by entering a search term e.g. "Windows"" and a "Search by Systems Manager parameter" button.

The "Quick Start" section lists three categories: "My AMIs", "AWS Marketplace", and "Community AMIs". Under "My AMIs", there is a single entry for "Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type - ami-0022f774911c1d690 (64-bit x86) / ami-0e449176cecc3e577 (64-bit Arm)". It includes a "Select" button, a radio button for "64-bit (x86)" (which is selected), and another for "64-bit (Arm)". Below the entry, it says "Amazon Linux 2 comes with five years support. It provides Linux kernel 5.10 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is now under maintenance only mode and has been removed from this wizard." It also mentions "Root device type: ebs" and "Virtualization type: hvm".

At the bottom of the wizard, there are tabs: "1. Choose AMI", "2. Choose Instance Type", "3. Configure Instance", "4. Add Storage", "5. Add Tags", "6. Configure Security Group", "7. Review", "Cancel and Exit", and "Feedback". The status bar at the bottom indicates "Looking for language selection? Find it in the new Unified Settings" and shows the date and time as "© 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 9:42 AM 5/26/2022".

Step 4: Choose EC2 Instance Types

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

The screenshot shows the AWS Launch Instance Wizard interface, specifically Step 2: Choose an Instance Type. It says: "Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs." Below this, there is a table of instance types.

The table has columns: Family, Type, vCPUs, Memory (GiB), Instance Storage (GB), EBS-Optimized Available, Network Performance, and IPv6 Support. The table shows the following data:

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	t2	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.large	2	8	EBS only	-	Low to Moderate	Yes

Below the table, there are buttons: "Cancel", "Previous", "Review and Launch", and "Next: Configure Instance Details". The status bar at the bottom indicates "Looking for language selection? Find it in the new Unified Settings" and shows the date and time as "© 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 9:34 PM 6/1/2022".

Step 5: Configure Instance Detail:

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot Instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances: 2

Purchasing option: Request Spot instances

Network: vpc-0059da187d97fad66 (default)

Subnet: No preference (default subnet in any Availability Zone)

Auto-assign Public IP: Use subnet setting (Enable)

Hostname type: Use subnet setting (IP name)

DNS Hostname:

- Enable IP name IPv4 (A record) DNS requests
- Enable resource-based IPv4 (A record) DNS requests
- Enable resource-based IPv6 (AAAA record) DNS requests

Placement group: Add instance to placement group

Review and Launch

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:

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/xvda	snap-08cbb15f1c8eb5387	8	General Purpose SSD (gp2)	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypt

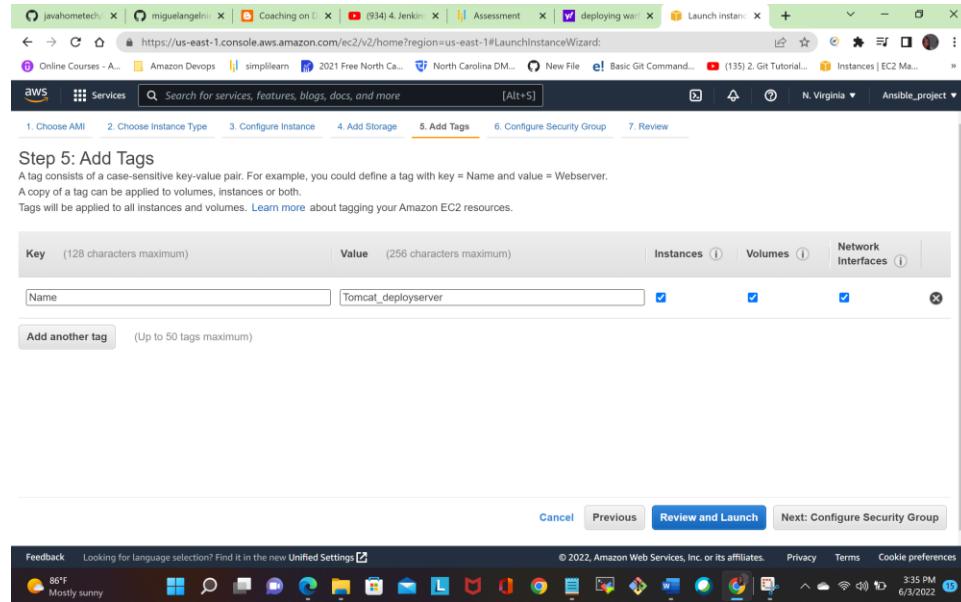
Add New Volume

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

Review and Launch

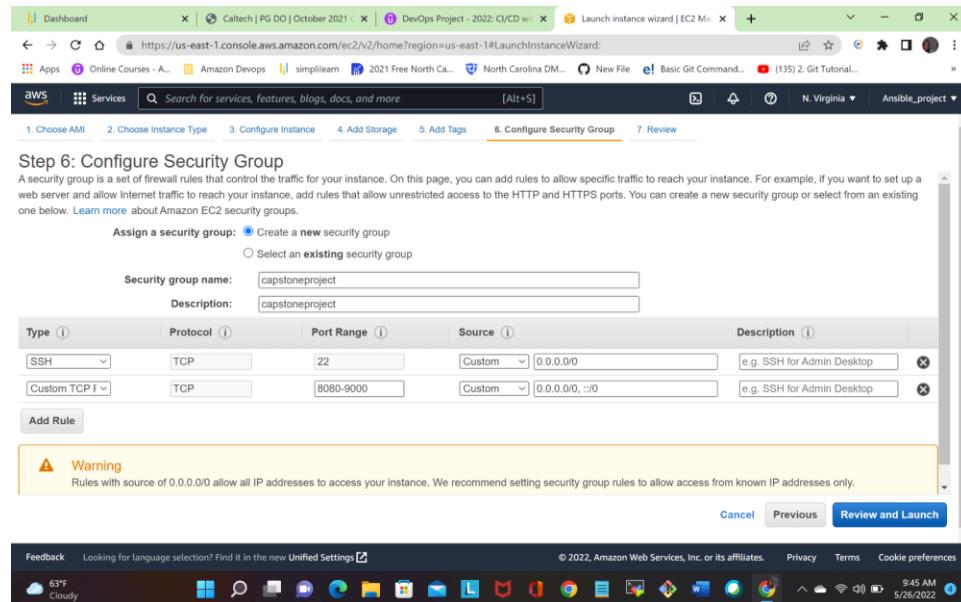
Step 7: Tag Instance of Ec2 Instance

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



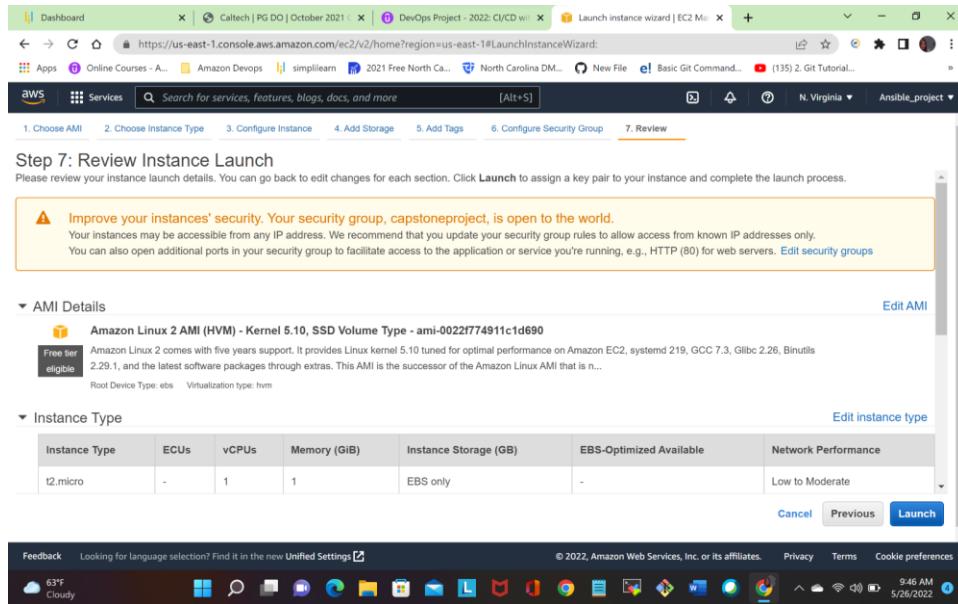
Step 8: Configure Security Groups

In this next step of configuring Security Groups



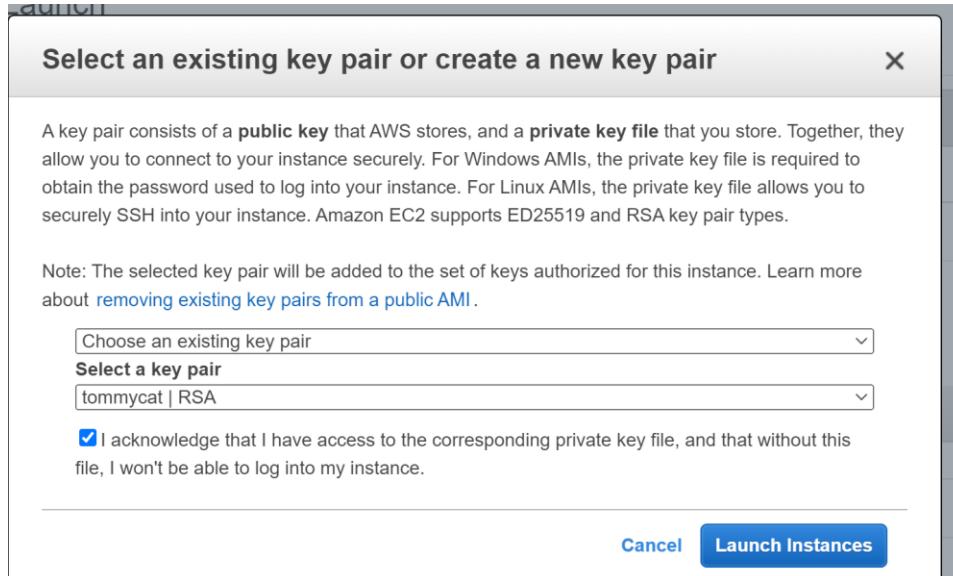
I created security group called capstone project and Defining protocols which i want enabled on my instance

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.

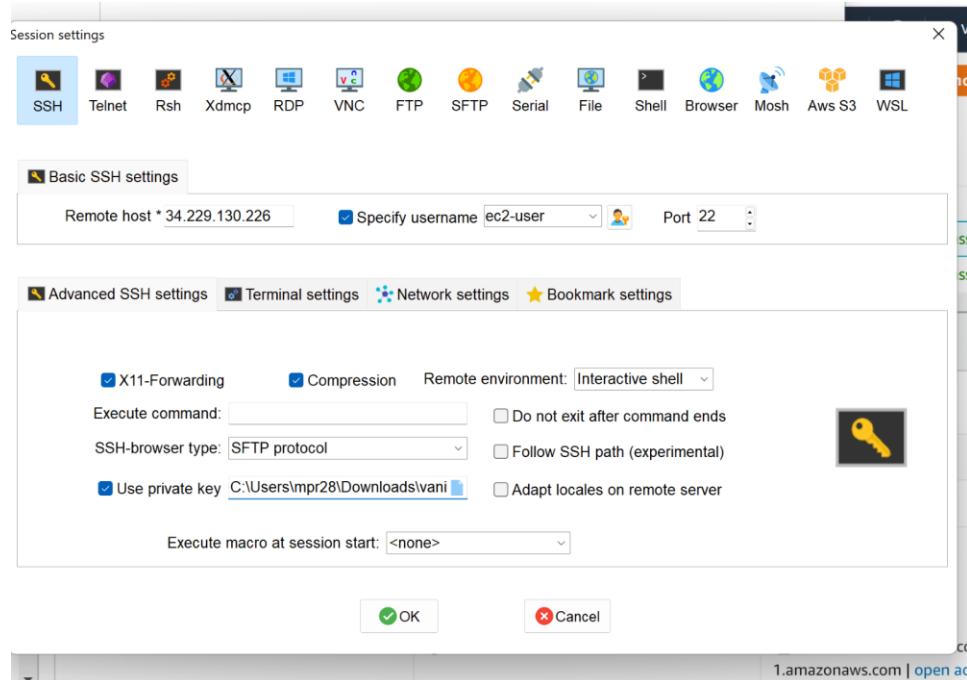


Use pem file in MobaXterm to connect ec2 machine in AWS: -

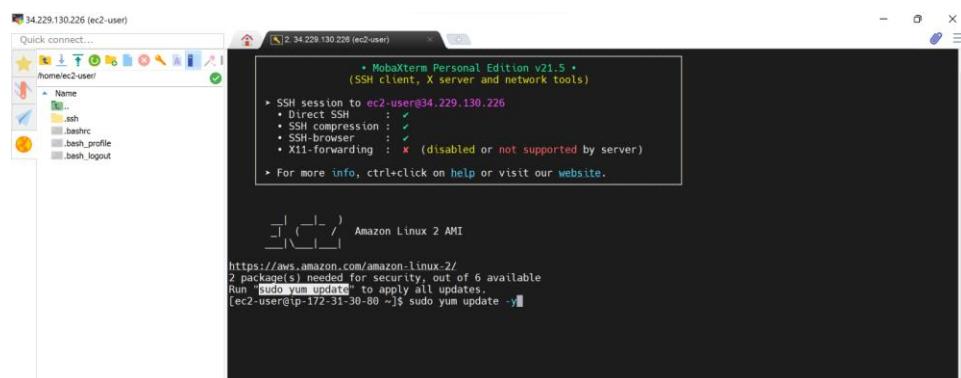
I used MobaXterm to connect ec2 machine. I downloaded mobaxterm .go to the session settings to create a new session, fill up the remote host and username.

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`

```

OS name: Linux , version: 5.10.71-106.499.amzn2.x86_64 , arch: aarch64 , family: Linux
root@ip-172-31-19-44 ~]# service jenkins status
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/usr/lib/systemd/system/jenkins.service; disabled; vendor preset: disabled)
    Active: inactive (dead)
      Docs: man:systemctl(1)
[root@ip-172-31-19-44 ~]# service jenkins start
Starting Jenkins via systemctl: [ OK ]
root@ip-172-31-19-44 ~]# service jenkins status
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/usr/lib/systemd/system/jenkins.service; disabled; vendor preset: disabled)
    Active: active (running) since Fri 2022-06-03 20:07:50 UTC; 3s ago
      Main PID: 9737 (java)
     CGroup: /system.slice/jenkins.service
             └─9737 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080

Jun 03 20:07:47 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:47.546+0000 [id:27] INFO jenkins.InitReactorRunner$1#0... Jun 03 20:07:48 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:48.774+0000 [id:27] INFO jenkins.InitReactorRunner$1#0...
Jun 03 20:07:49 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:49.961+0000 [id:27] INFO jenkins.InitReactorRunner$1#0...
Jun 03 20:07:49 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:49.982+0000 [id:28] INFO jenkins.InitReactorRunner$1#0...
Jun 03 20:07:49 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:49.986+0000 [id:28] INFO jenkins.InitReactorRunner$1#0...
Jun 03 20:07:50 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:50.121+0000 [id:42] INFO hudson.model.AsyncPeriodicWork...
Jun 03 20:07:50 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:50.133+0000 [id:42] INFO hudson.model.AsyncPeriodicWork...
Jun 03 20:07:50 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:50.298+0000 [id:28] INFO jenkins.InitReactorRunner$1#0...
Jun 03 20:07:50 ip-172-31-19-44.ec2.internal jenkins[9737]: 2022-06-03 20:07:50.341+0000 [id:21] INFO hudson.lifecycle.Lifecycle#on...
Jun 03 20:07:50 ip-172-31-19-44.ec2.internal systemd[1]: Started Jenkins Continuous Integration Server.

[Truncated]

```

Accessing Jenkins

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

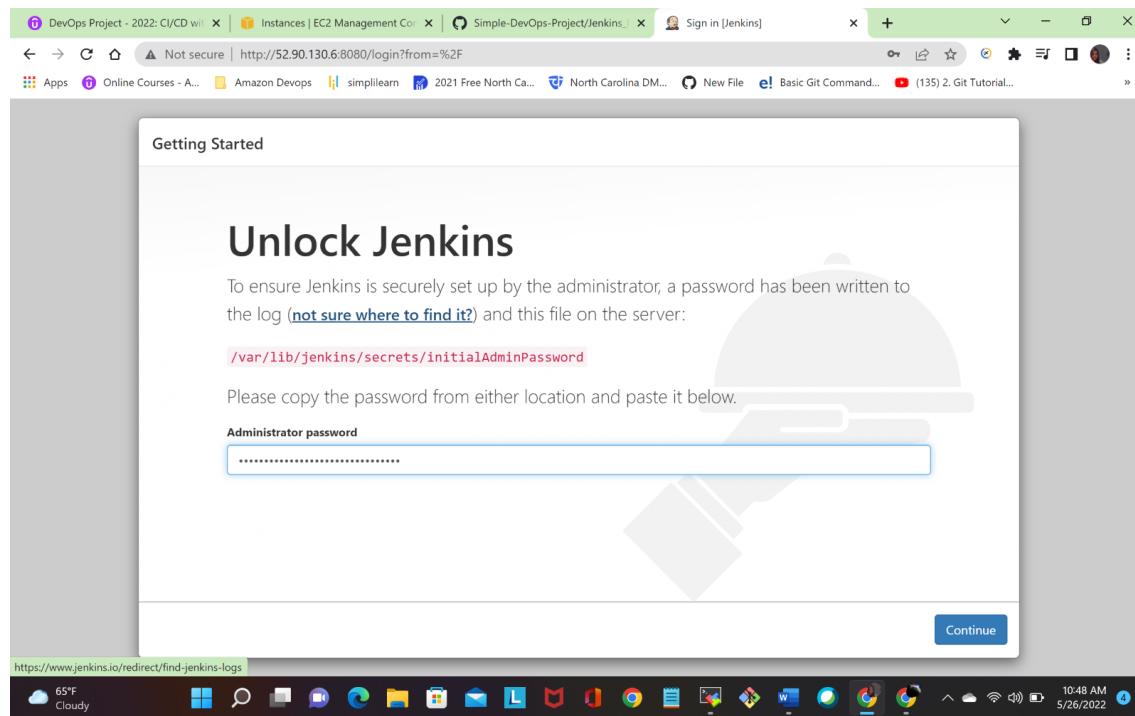
<http://YOUR-SERVER-PUBLIC-IP:8080>

Configure Jenkins:

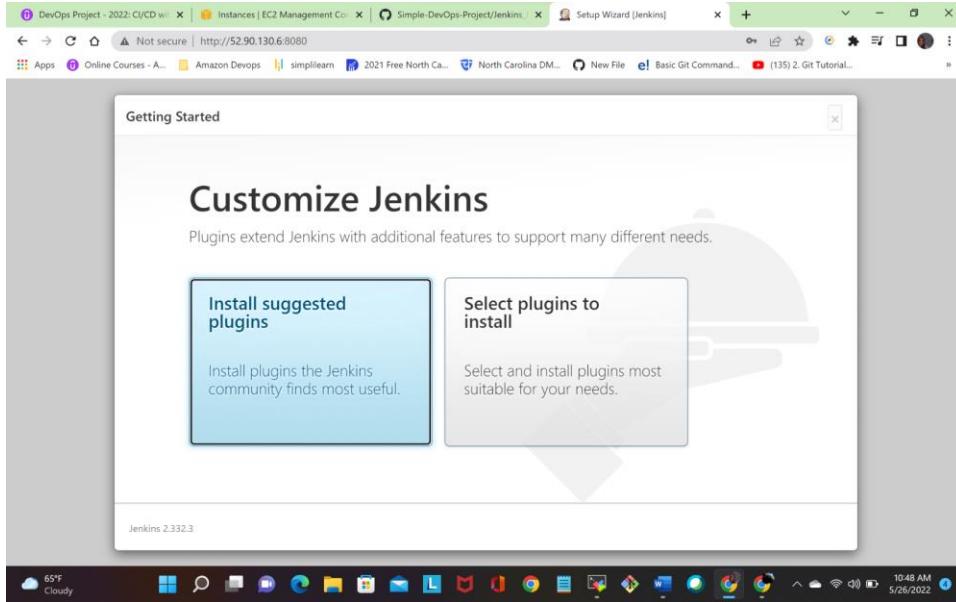
Grab the default password

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

A screenshot of the Jenkins "Create First Admin User" setup step. The title is "Create First Admin User". There are five input fields: "Username" (vani), "Password" (redacted), "Confirm password" (redacted), "Full name" (vani), and "E-mail address" (vani123@gmail.com). At the bottom right are two buttons: "Skip and continue as admin" and "Save and Continue" (highlighted in blue).

Click save and continue.it taken you into jenkins dashboard

Jenkins Dashboard: -

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job →

Set up a distributed build

Set up an agent →

Configure java and maven path

Manage Jenkins > Global Tool Configuration > JDK

Java 11 is the recommended version to run Jenkins on; please consider upgrading.

System Configuration

Configure System Configure global settings and paths.

Global Tool Configuration Configure tools, their locations and automatic installers.

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

Manage Nodes and Clouds
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Security

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

JDK

JDK installations

Add JDK

JDK Name

JAVA_HOME

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

Install automatically

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

Maven

Maven installations

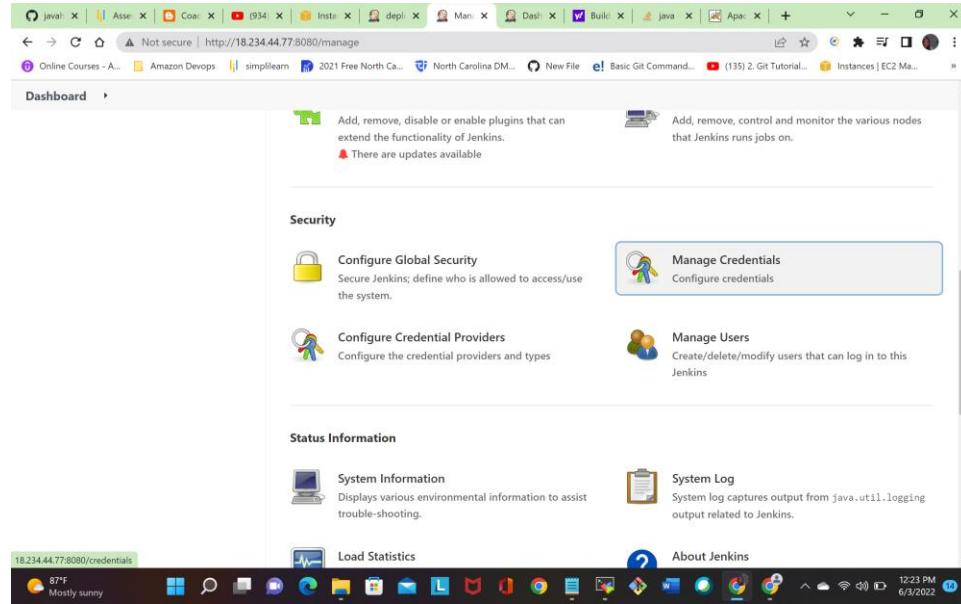
Add Maven

Maven
Name
MAVEN_HOME

MAVEN_HOME
/opt/apache-maven-3.8.5

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

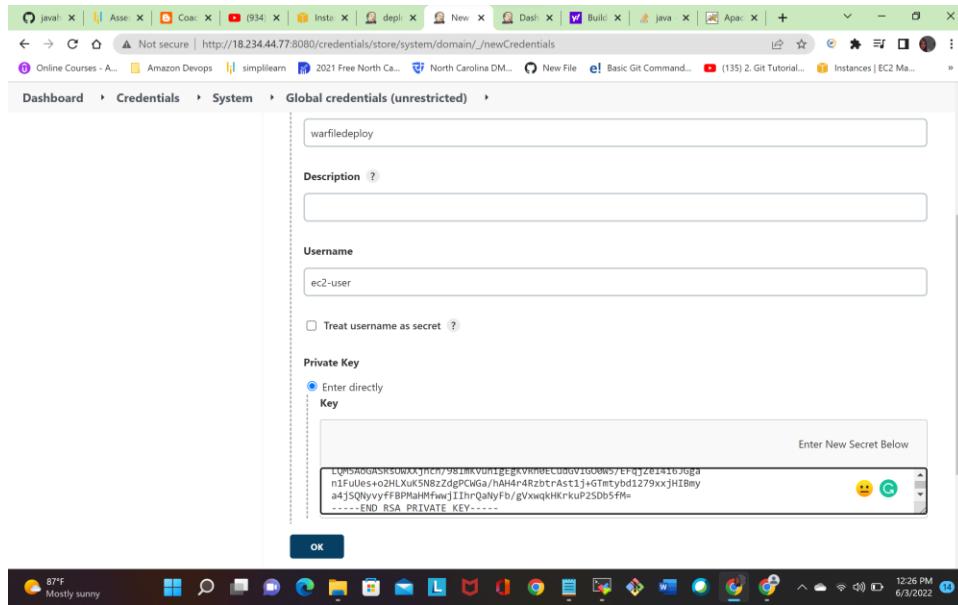
The screenshot shows the Jenkins Global credentials (unrestricted) page. At the top, there is a search bar and a navigation menu with links like 'Dashboard', 'Credentials', 'System', and 'Global credentials (unrestricted)'. Below the header, there is a section titled 'Global credentials (unrestricted)' with a sub-section header 'Generated that should be available irrespective of domain specification to requirements matching'. A table lists five credentials:

ID	Name	Kind	Description
	tomcat	SSH Username with private key	tomcat
	8549fbea-9c16-4d62-a8ef-ac2282d6e086	Username with password	Generated deploy-plugin credentials for tomcat8x
	tommycontainer	SSH Username with private key	
	tommyserver	Username with password	tommyserver
	tom	Username with password	admin*****

At the bottom of the page, there is a toolbar with icons for 'Icon: S M L' and a status bar showing '18.234.44.77:8080/credentials/store/system/domain/_/newCredentials' and the date '6/3/2022'.

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

The screenshot shows the 'Add Credentials' dialog box. The 'Kind' dropdown menu is open, showing several options: 'Username with password', 'Username with password GitHub App', 'SSH Username with private key' (which is selected), 'Secret file', 'Secret text', and 'Certificate'. Below the dropdown, there are fields for 'Username', 'Password', and 'ID'. There is also a checkbox labeled 'Treat username as secret'. At the bottom right of the dialog is an 'OK' button.



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.

Binary Distributions

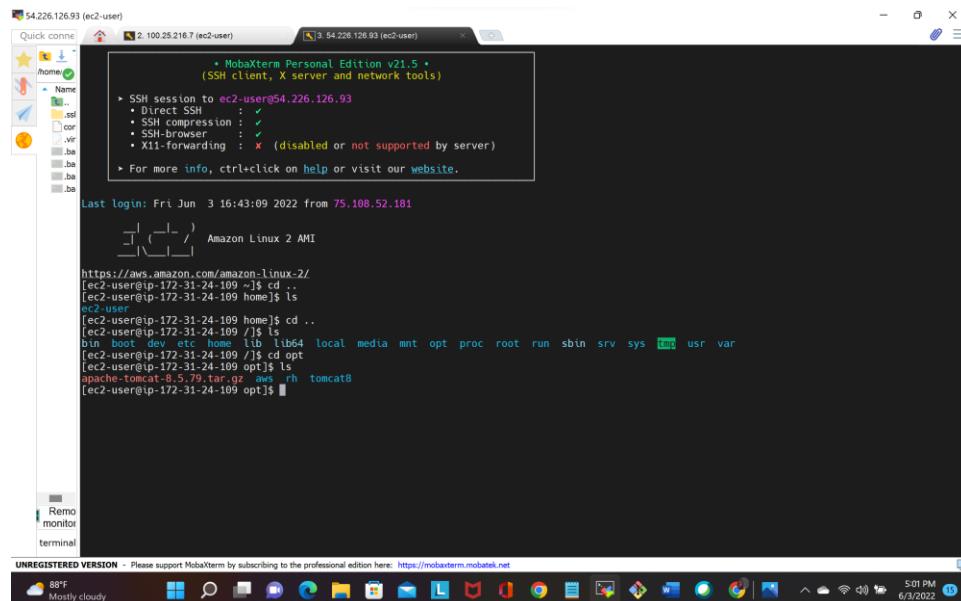
- Core:
 - [zip \(pgp, sha512\)](#)
 - [tar.gz \(pgp, sha512\)](#)
 - [32-bit](#) Open link in new tab
 - [64-bit](#) Open link in new window
 - [32-bit](#) Open link in incognito window
- Full document
- [tar.gz](#) Open link as
- Deployer:
- [zip \(pgp, sha512\)](#) Save link as...
- [tar.gz \(pgp, sha512\)](#) Copy link address
- Extras:
- [Web](#) Inspect
- Embedded:
- [tar.gz \(pgp, sha512\)](#)
- [zip \(pgp, sha512\)](#)

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

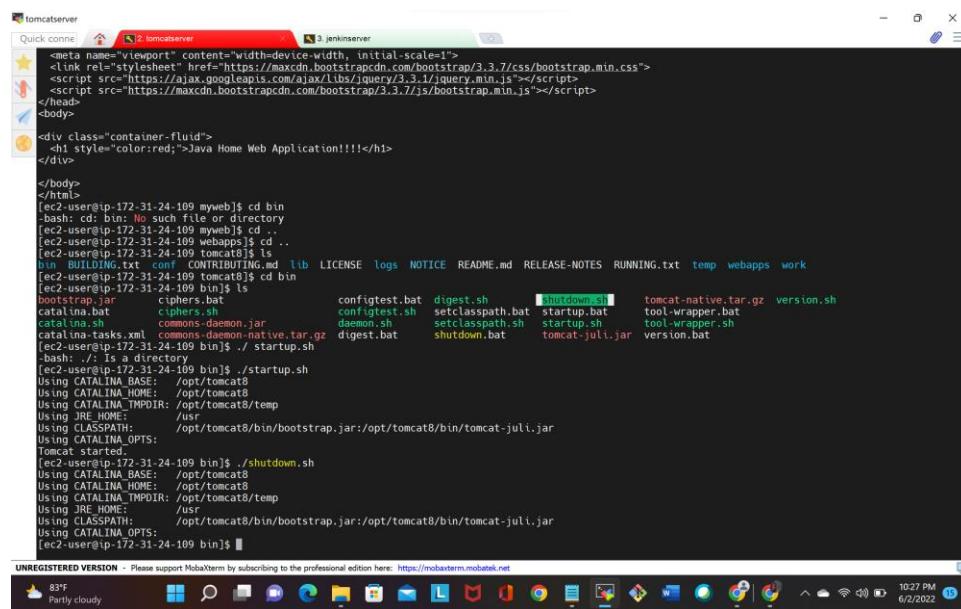
I gave full ownership to ec2-user for tomcat8 folder using following command

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

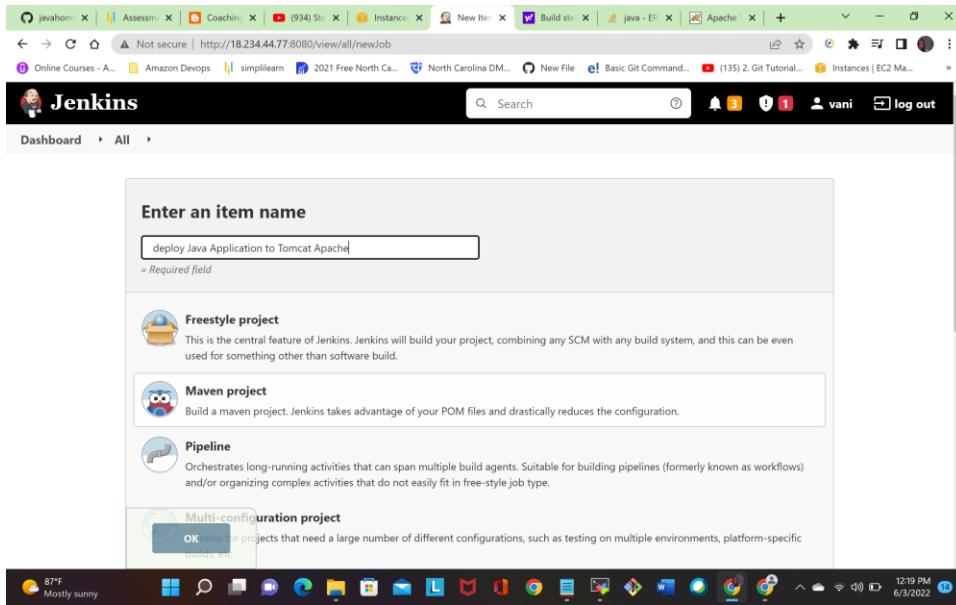
**Go to manage jenkins-->plugin manager-->available section and search for ssh agent plugin and Pipeline: Declarative
-->click install without restart**

The screenshot shows the Jenkins Plugin Manager interface. At the top, there's a search bar with the text 'ssh agent'. Below it, there are tabs for 'Updates', 'Available', 'Installed' (which is selected), and 'Advanced'. Under the 'Available' tab, two plugins are listed:

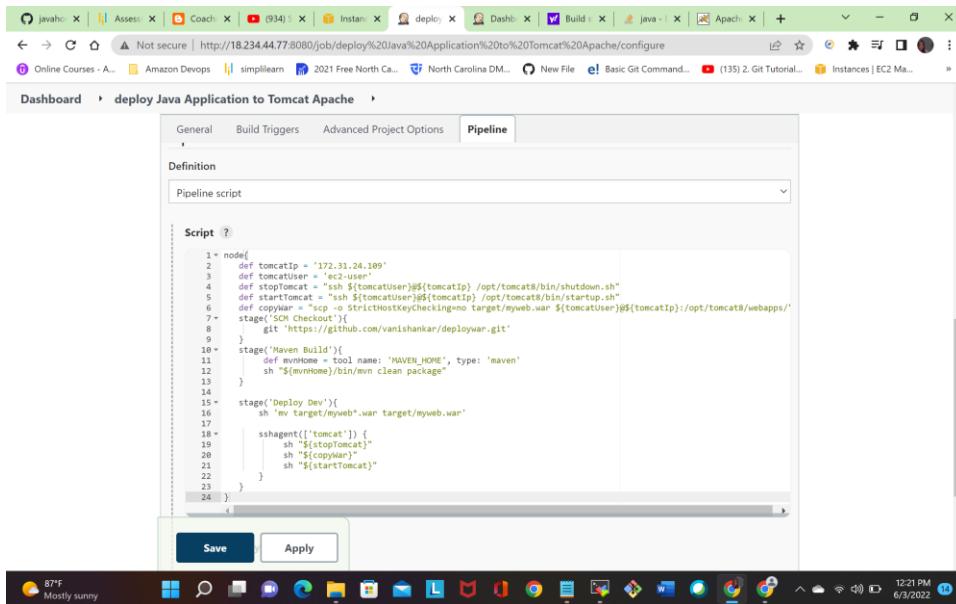
- SSH Agent Plugin** (version 295.v9ca_a_1c7cc3a_a) - Description: This plugin allows you to provide SSH credentials to builds via a ssh-agent in Jenkins. Status: Enabled.
- SSH Build Agents plugin** (version 1.814.vc82988f54b_10) - Description: Allows to launch agents over SSH, using a Java implementation of the SSH protocol. Status: Enabled.

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 ?

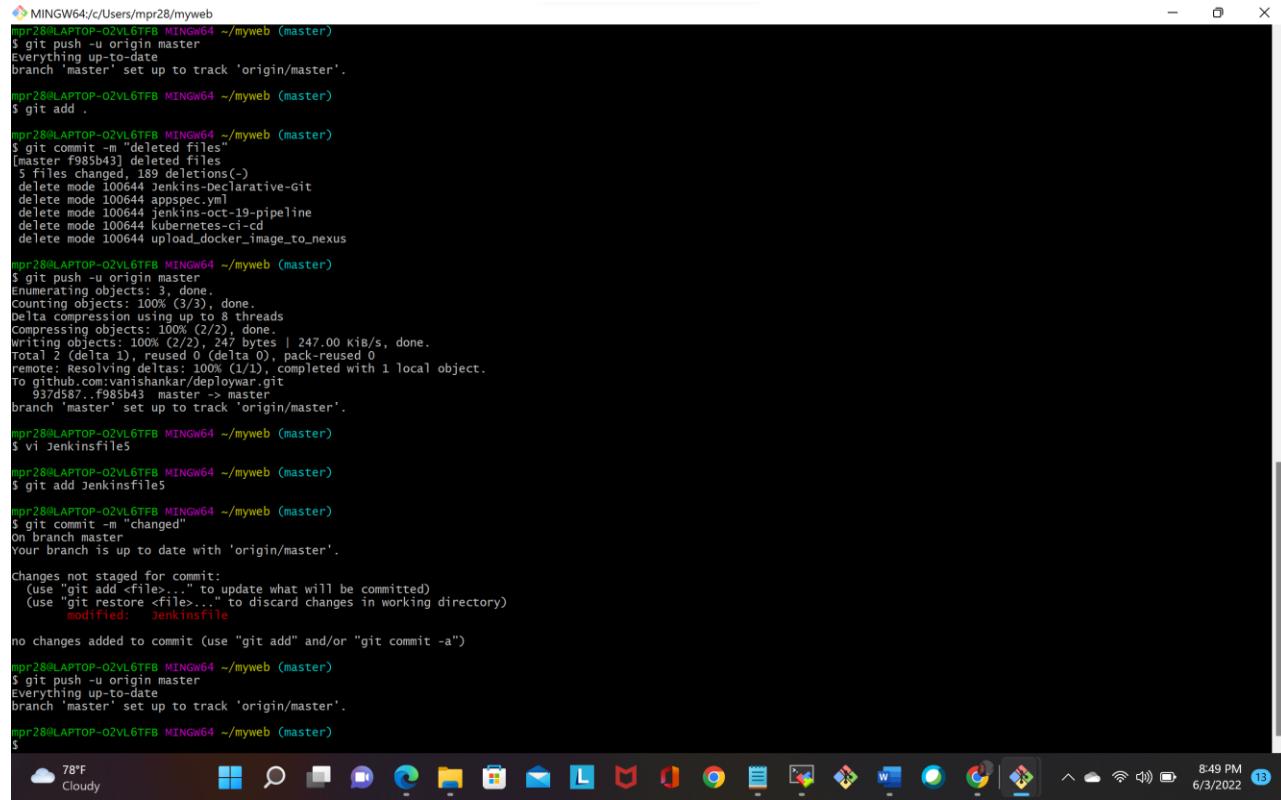
ec2-user Add ?

Ignore missing credentials ?

Generate Pipeline Script

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

After creating jenkins file I pushed into GitHub



```

MINGW64:/c/Users/mpnpr28/myweb
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$ git push -u origin master
Everything up-to-date
branch 'master' set up to track 'origin/master'.
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$ git add .
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$ git commit -m "deleted files"
[master f985b43] deleted files
 5 files changed, 189 deletions(-)
   delete mode 100644 jenkins-declarative-git
   delete mode 100644 appspec.yml
   delete mode 100644 jenkins-oct-19-pipeline
   delete mode 100644 kubernetes-ci-ci-d
   delete mode 100644 upload_docker_image_to_nexus
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 3 (delta 0), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 247 bytes | 247.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:vanishankar/deploywar.git
   937d587..f985b43 master -> master
branch 'master' set up to track 'origin/master'.
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$ vi Jenkinsfile
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$ git add Jenkinsfile
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$ git commit -m "changed"
On branch master
Your branch is up to date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified: Jenkinsfile

no changes added to commit (use "git add" and/or "git commit -a")
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$ git push -u origin master
Everything up-to-date
branch 'master' set up to track 'origin/master'.
mpnpr28@LAPTOP-OZVL6TFB MINGW64 ~/myweb (master)
$
```

Cloudy 8:49 PM 6/3/2022

```

MINGW64:/c/Users/mpr28/myweb/src/main/webapp
branch 'master' set up to track 'origin/master'.
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb (master)
$ ls
Dockerfile          README.md      ci-cd-docker-declarative hello      kubernetes-ci-cd  src/
Jenkinsfile         appspec.yml   deployments.yml    jenkins-oct-19-pipeline pom.xml
Jenkinsfile2        azure-pipelines.yml docker-ci-cd      k8s/      start_tomcat.sh* upload_docker_image_to_nexus
Jenkinsfile2        buildspec.yml  dummy.sh       sftp_tomcat.sh
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb (master)
$ cd src/
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src (master)
$ ls
main/ test/
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src (master)
$ cd main
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src/main (master)
$ ls
java/ webapp/
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src/main (master)
$ cd webapp/
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src/main/webapp (master)
$ ls
WEB-INF/ index.jsp
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src/main/webapp (master)
$ git add index.jsp
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src/main/webapp (master)
$ git commit -m "committed"
[master f4ffab0] committed
 1 file changed, 1 insertion(+), 1 deletion(-)
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src/main/webapp (master)
$ git push -u origin master
Enumerating objects: 11, done.
Counting objects: 1000 (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6) | 574 bytes | 574.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0
Remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:vanishankar/deploywar.git
 923245f..f4ffab0 master -> master
branch 'master' set up to track 'origin/master'.
mpr28@LAPTOP-02VL6TFB MINGW64 ~/myweb/src/main/webapp (master)
$ 

```

83°F Partly cloudy 10:00 PM 6/2/2022

GitHub repo

The screenshot shows a GitHub repository page for the user 'vanishankar' named 'deploywar'. The repository is public. At the top, there are links for Pull requests, Issues, Marketplace, and Explore. Below that, there's a search bar and a 'Code' button. The main area displays the repository structure with a 'master' branch selected. A recent commit by 'vanishankar' titled 'Create Jenkinsfile' is shown, dated 3 hours ago with 470 commits. There are also links for Readme and other repository details.

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

Stage View

SCM Checkout	Maven Build	Deploy Dev
1s	10s	3s

Average stage times:
(Average full run time: ~17s)

Recent Changes

Jun 03
17:29
No Changes

1s 10s 3s

Permalinks

- Last build (#1), 3 hr 3 min ago
- Last stable build (#1), 3 hr 3 min ago
- Last successful build (#1), 3 hr 3 min ago
- Last completed build (#1), 3 hr 3 min ago

Console Output Result

Console Output

Started by user vani
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache
[Pipeline] {
[Pipeline] stage
[Pipeline] { (SCM Checkout)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/vanishankar/deploywar.git
> git init /var/lib/jenkins/workspace/deploy Java Application to Tomcat Apache # timeout=10
Fetching upstream changes from https://github.com/vanishankar/deploywar.git
> git --version # timeout=10
> git --version # git version 2.32.0
> git fetch --tags --force --progress -- https://github.com/vanishankar/deploywar.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/vanishankar/deploywar.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision f4ffab0d7b29b53146c4f987cb7682225251927b (refs/remotes/origin/master)

```
[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
```

```
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 msec]
[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] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 6.604 s
[INFO] Finished at: 2022-06-03T21:29:39Z
[INFO] -----
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] {
  [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)
```

```
[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.**

The screenshot shows a web browser window with the following details:

- Address Bar:** http://54.226.126.93:8080/myweb
- Error Message:** Error! Hyperlink reference not valid.
- Tomcat Welcome Page:** The page displays the Apache Tomcat 8.5.79 logo and a green banner stating "If you're seeing this, you've successfully installed Tomcat. Congratulations!".
- Tomcat Features:** A cartoon cat icon is shown next to a list of recommended readings: Security Considerations How-To, Manager Application How-To, and Clustering/Session Replication How-To.
- Navigation Links:** Server Status, Manager App, and Host Manager buttons are visible on the right side.
- Developer Quick Start:** Links to Tomcat Setup, First Web Application, Realms & AAA, JDBC DataSources, Examples, Servlet Specifications, and Tomcat Versions.

Finally deployed java application in tomcat server



And we can see in jenkins workspace in our jenkins project

```
[root@ip-172-31-19-44 workspace]# ls
config.properties
deploytomcat
deploy Java Application to Tomcat Apache
deploy Java Application to Tomcat Apache@tmp
deploytomcatcontainer
deploytomcatcontainer@tmp
[root@ip-172-31-19-44 workspace]#
```

Mahayarn - Please support Mahayarn by subscribing to the professional edition here: <https://mahayarn.mahatalk.net>

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

=====