

**Practice for Lesson 5:
Understanding of Parallel
Jenkins Jobs and Jenkins
Slave on AWS**

Practices for Lesson 5

Overview

In these practices, you will learn how to Build and Deploy an Application to Webserver using Jenkins Pipeline. Further create a parallel Agent Pipeline Job on Jenkins.

.

Practice 5-1: Build and Deploy an Application to Webserver using Jenkins Pipeline

Overview

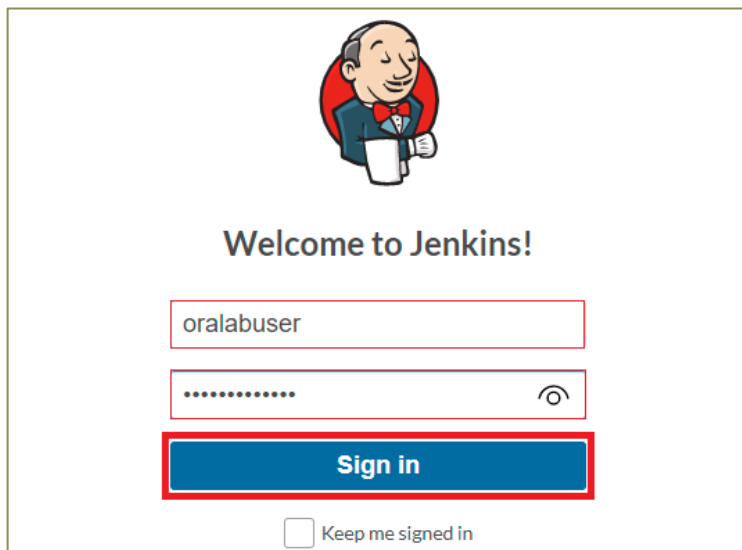
In this practice, you will learn how to Build and Deploy an Application to Webserver using Jenkins Pipeline.

Assumptions

You should have completed the Practice of Lesson 4.

Tasks

1. Sign in to **Jenkins Instance Dashboard**.
 - a. In a browser on your local machine, enter the **Public IP** address of the EC2 instance followed by the IP address to sign in to the Jenkins Dashboard (for example, **<Public-IP>:8080**).



- b. Enter the user name and password provided.
 - c. You will have access to the Jenkins Dashboard.
2. Install Maven on the **Linux instance** (Production server).
 - a. Connect to the **Linux instance** from the **Putty**, run the code to install the **Maven** and **Git** in the server as shown below.

```
[ec2-user@ip-172-31-35-174 ~]$ sudo yum install -y git maven

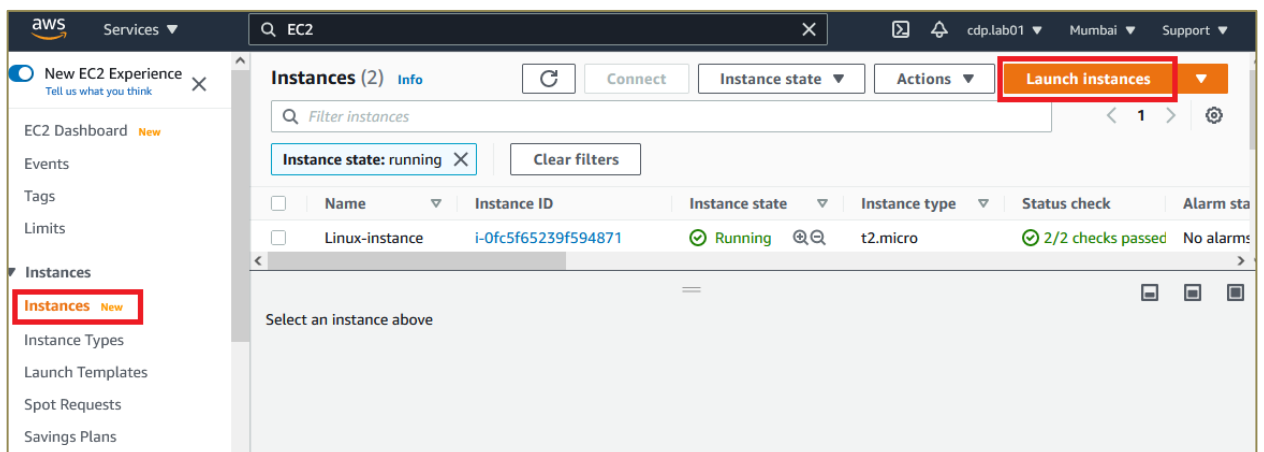
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core | 3.7 kB 00:00
amzn2extra-docker | 3.0 kB 00:00
jenkins | 2.9 kB 00:00
Package git-2.23.4-1.amzn2.0.1.x86_64 already installed and latest version
Resolving Dependencies
--> Running transaction check
---> Package maven.noarch 0:3.0.5-17.amzn2 will be installed
--> Processing Dependency: sisu-inject-plexus for package: maven-3.0.5-17.amzn2.noarch
--> Processing Dependency: sisu-inject-bean for package: maven-3.0.5-17.amzn2.noarch
```

- b. Check the version of **Git** & **Maven** as shown below.

```
[ec2-user@ip-172-31-35-174 ~]$ git --version
git version 2.23.4
[ec2-user@ip-172-31-35-174 ~]$ mvn --version
Apache Maven 3.0.5 (Red Hat 3.0.5-17)
Maven home: /usr/share/maven
Java version: 1.8.0_282, vendor: Red Hat, Inc.
Java home: /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.282.b08-1.amzn2.0.1.x86_64/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "4.14.232-176.381.amzn2.x86_64", arch: "amd64", family: "unix"
[ec2-user@ip-172-31-35-174 ~]$
```

3. Create Webserver Instance in the AWS Console.

- a. Open AWS console and navigate to **EC2** as shown below, select **Launch Instance** to create an instance to the Webserver.



- b. In "Step 1: Choose an Amazon Machine Image (AMI)", select **Ubuntu Server 64-bit (x86)** as shown below.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

[Cancel and Exit](#)

SUSE Linux
Free tier eligible

SUSE Linux Enterprise Server 15 SP2 (HVM), SSD Volume Type - ami-0b3acf3edf2397475 (64-bit x86) / ami-0ab71076ab9b53b0d (64-bit Arm)

SUSE Linux Enterprise Server 15 Service Pack 2 (HVM), EBS General Purpose (SSD) Volume Type. Amazon EC2 AMI Tools preinstalled; Apache 2.2, MySQL 5.5, PHP 5.3, and Ruby 1.8.7 available.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

[Select](#)

☒ 64-bit (x86)
☐ 64-bit (Arm)

Ubuntu
Free tier eligible

Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-0c1a7f89451184c8b (64-bit x86) / ami-0d18acc6e813fd2e0 (64-bit Arm)

Ubuntu Server 20.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

[Select](#)

☒ 64-bit (x86)
☐ 64-bit (Arm)

Windows
Free tier eligible

Microsoft Windows Server 2019 Base - ami-0e5d82cae7458738b

Microsoft Windows 2019 Datacenter edition. [English]

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

[Select](#)

64-bit (x86)

- c. In “Step 2: Choose an Instance Type”, select instance size **t2.micro** and click **Next: Configure Instance Details** as shown below.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 2: Choose an Instance Type

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.

Filter by: All instance families Current generation [Show/Hide Columns](#)

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

| | 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 |

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Instance Details](#)

- d. In “Step 3: Configure instance” accept the default values and click **Next**.
- e. In “Step 4: Add Storage”, accept the default values and click **Next: Add Tags**.
- f. In “Step 5: Add Tags”, click **Add Tag** to provide the **Key** as **Name** and **Value** as **Web-Server** and click **Next: Configure Security Group** as shown below.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

| Key (128 characters maximum) | Value (256 characters maximum) | Instances | Volumes | Network Interfaces |
|------------------------------|--------------------------------|-------------------------------------|-------------------------------------|-------------------------------------|
| Name | Web-Server | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |

[Add another tag](#) (Up to 50 tags maximum)

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Security Group](#)

- g. In “Step 6: Configure Security Group” click **Add Rule** to open the ports as shown below. Click **Review and Launch**.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group
☐ Select an existing security group

Security group name:
Description:

| Type | Protocol | Port Range | Source | Description |
|------------|----------|------------|------------------------|----------------------------|
| SSH | TCP | 22 | Custom 0.0.0.0/0 | e.g. SSH for Admin Desktop |
| Custom TCP | TCP | 8080 | Custom 0.0.0.0/0, ::/0 | e.g. SSH for Admin Desktop |

[Add Rule](#)


[Cancel](#) [Previous](#) [Review and Launch](#)

- h. In “Step 7: Review Instance Launch”, review the details of the instance and click **Launch** as shown below.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 7: Review Instance Launch

▼ AMI Details [Edit AMI](#)

 **Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-0c1a7f89451184c8b**

Free tier eligible Ubuntu Server 20.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Root Device Type: ebs Virtualization type: hvm

▼ Instance Type [Edit instance type](#)

| Instance Type | ECUs | vCPUs | Memory (GiB) | Instance Storage (GB) | EBS-Optimized Available | Network Performance |
|---------------|------|-------|--------------|-----------------------|-------------------------|---------------------|
| t2.micro | - | 1 | 1 | EBS only | - | Low to Moderate |

▼ Security Groups [Edit security groups](#)

Security group name launch-wizard-7

Description launch-wizard-7 created 2021-06-16T16:10:03.914+05:30

[Cancel](#) [Previous](#) [Launch](#)

- i. Select from the drop down **Choose an existing key pair**, select the **Key Pair**. Click **Launch Instances** as shown below.

Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

Choose an existing key pair ▼

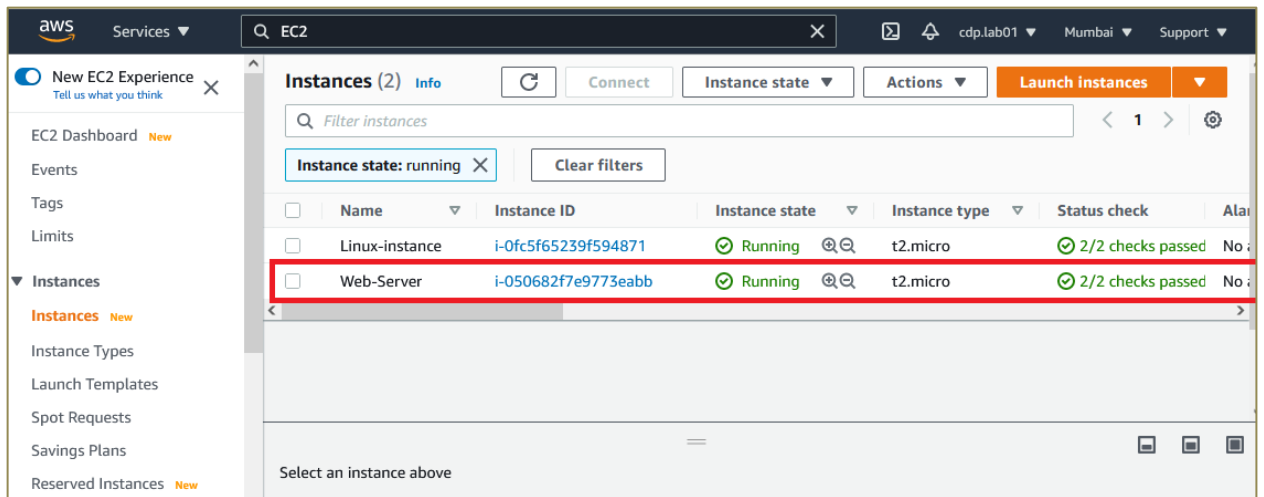
Select a key pair

AWS_user_key ▼

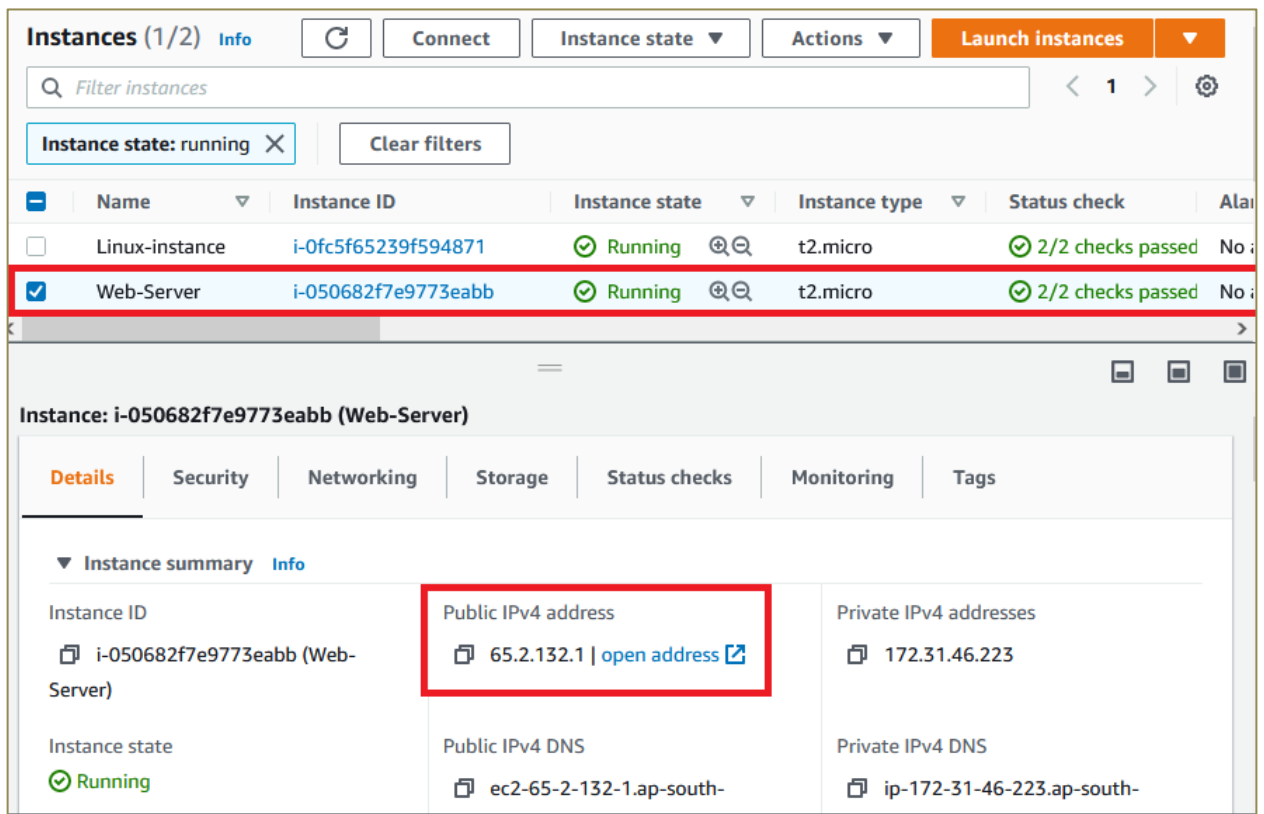
☒ I acknowledge that I have access to the selected private key file (AWS_user_key.pem), and that without this file, I won't be able to log into my instance.

[Cancel](#) [Launch Instances](#)

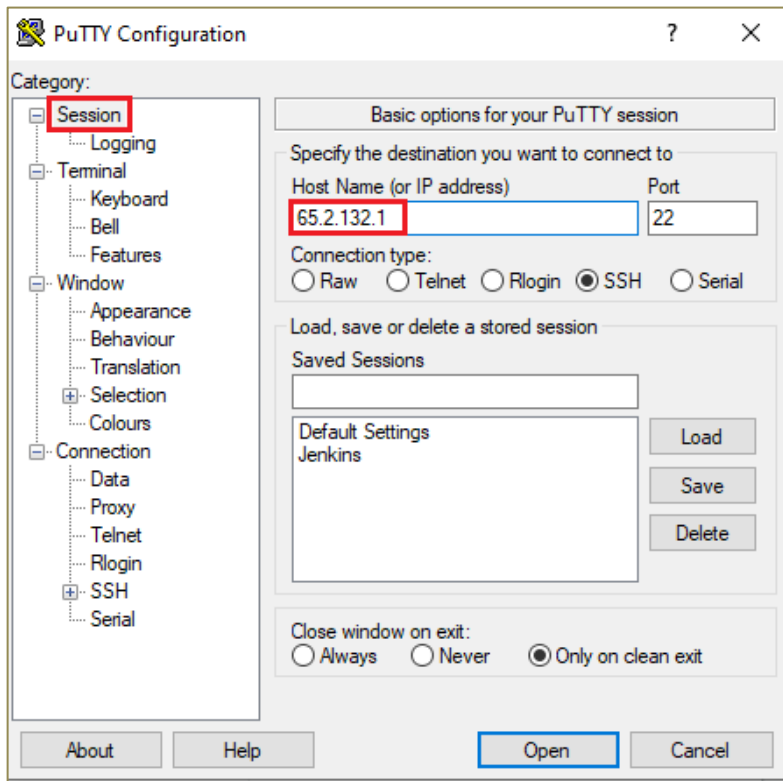
- j. The AWS EC2 **Web-Server** is created successfully and its status is **running** as shown below.



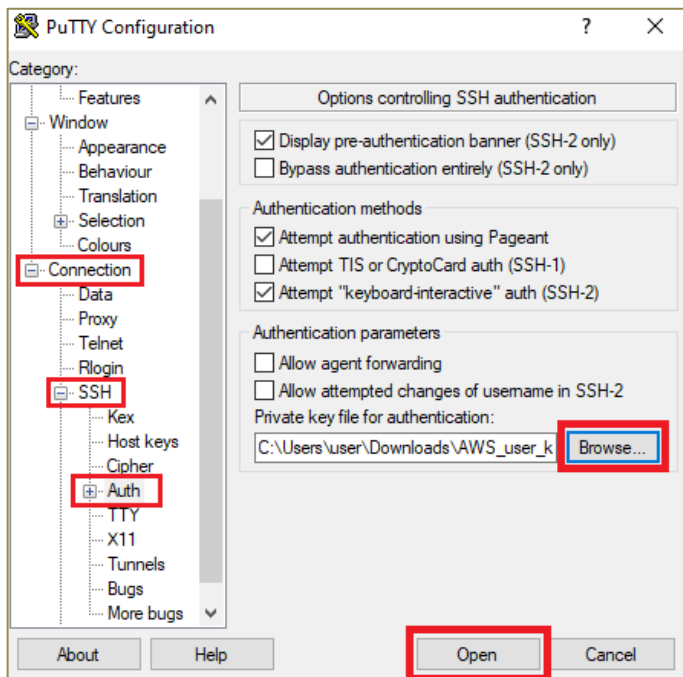
- k. To connect to the Web-Server instance copy the public IP address as shown below.



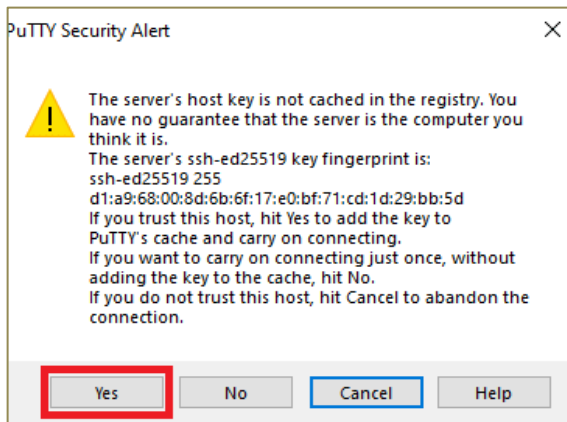
- l. Open Putty Configuration, navigate to **Session** and paste the instance public IP address in the **Host Name** as shown below.



- m. Navigate to **Connection**, go to **SSH** and select **Auth**. Click **Browse** to add .ppk file and click **Open** as shown below.



- n. Putty connects to the AWS **Web_Server** instance and a Putty Security Alert pop-up displays as shown below. Click **Yes**.



- o. In the terminal window, for **login as:** type **ubuntu** and the login is successful as shown below.

```
ubuntu@ip-172-31-46-223: /etc/tomcat9
login as: ubuntu
Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-1045-aws x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

System information as of Wed Jun 16 05:27:34 UTC 2021

System load:  0.14           Processes:            103
Usage of / :  16.4% of 7.69GB Users logged in:          0
Memory usage: 22%           IPv4 address for eth0: 172.31.46.223
Swap usage:   0%

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-46-223:~$
```

- p. Run the command to Update the apt repository as shown below.

```

ubuntu@ip-172-31-46-223:~$ sudo apt-get update
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:2 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [14 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Get:5 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [70 kB]
Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [10 kB]
Get:8 http://security.ubuntu.com/ubuntu focal-security/main amd64 c-n-f Metadata [7780 B]
Get:9 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [251 kB]
Get:10 http://security.ubuntu.com/ubuntu focal-security/restricted Translation-en [36.5 kB]
Get:11 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 c-n-f Metadata [456 B]
Get:12 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [588 kB]
Get:13 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5384 kB]

```

- q. Run the command to Install **Tomcat** in the **Web-Server** as shown below.

```

ubuntu@ip-172-31-46-223:~$ sudo apt-get install -y tomcat9
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java default-jre-headless fontconfig-config fonts-dejavu-core java-common
  libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libecj-jdt-core-j
  libfontconfig1 libgraphite2-3 libharfbuzz0b libjpeg-turbo8 libjpeg8 liblcms2-2 libnspr
  libpcsclite1 libtcnative-1 libtomcat9-java openjdk-11-jre-headless tomcat9-common
Suggested packages:
  default-jre cups-common liblcms2-utils pscd libnss-mdns fonts-dejavu-extra fonts-ipa
  fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei fonts-indic tomcat9-admin
  tomcat9-examples tomcat9-user
The following NEW packages will be installed:
  ca-certificates-java default-jre-headless fontconfig-config fonts-dejavu-core java-common
  libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libecj-jdt-core-j
  libfontconfig1 libgraphite2-3 libharfbuzz0b libjpeg-turbo8 libjpeg8 liblcms2-2 libnspr
  libpcsclite1 libtcnative-1 libtomcat9-java openjdk-11-jre-headless tomcat9 tomcat9-co
0 upgraded, 25 newly installed, 0 to remove and 60 not upgraded.
Need to get 53.2 MB of archives.
After this operation, 197 MB of additional disk space will be used.
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 java-common all
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libavahi-common-

```

- r. Install the one more package for the **Tomcat** Admin as shown below.

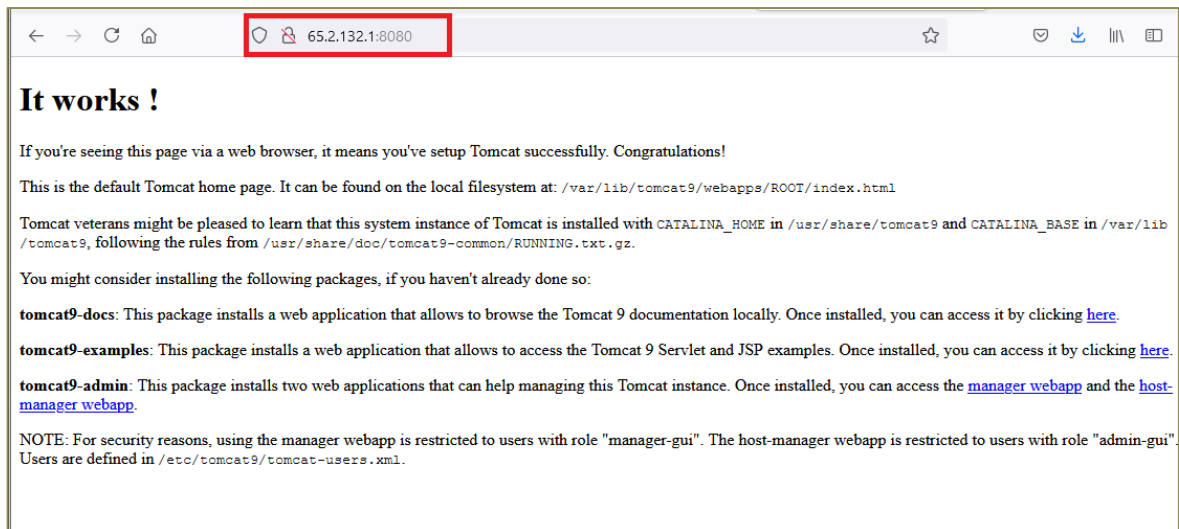
```

ubuntu@ip-172-31-46-223:~$ sudo apt-get install -y tomcat9-admin
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  tomcat9-admin
0 upgraded, 1 newly installed, 0 to remove and 60 not upgraded.
Need to get 24.6 kB of archives.
After this operation, 189 kB of additional disk space will be used.
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/universe
.31-lubuntu0.1 [24.6 kB]
Fetched 24.6 kB in 1s (25.6 kB/s)
Selecting previously unselected package tomcat9-admin.
(Reading database ... 61123 files and directories currently installed.)
Preparing to unpack .../tomcat9-admin_9.0.31-lubuntu0.1_all.deb ...
Unpacking tomcat9-admin (9.0.31-lubuntu0.1) ...
Setting up tomcat9-admin (9.0.31-lubuntu0.1) ...
ubuntu@ip-172-31-46-223:~$

```

- s. To verify the Tomcat installation, copy the Public IP of the Web-Server instance and paste in the browser with the port number 8080 as shown below.

Syntax: **<Public-IP Address>:8080**



4. Setting the path of Tomcat in Jenkins.
 - a. Connect to the **Web-Server** and execute the command to navigate to the Tomcat directory to list the files as shown below.

```

ubuntu@ip-172-31-46-223:~$ cd /etc/tomcat9/
ubuntu@ip-172-31-46-223:/etc/tomcat9$ ls
Catalina          context.xml        logging.properties  server.xml          web.xml
catalina.properties  jaspic-providers.xml  policy.d             tomcat-users.xml
ubuntu@ip-172-31-46-223:/etc/tomcat9$

```

- b. Open the file **tomcat-users.xml** using the vi editor as shown below.

```

ubuntu@ip-172-31-46-223:/etc/tomcat9$ sudo vi tomcat-users.xml

```

- c. Navigate to the end of the file and add the statement given below add the username, password and roles as shown below.

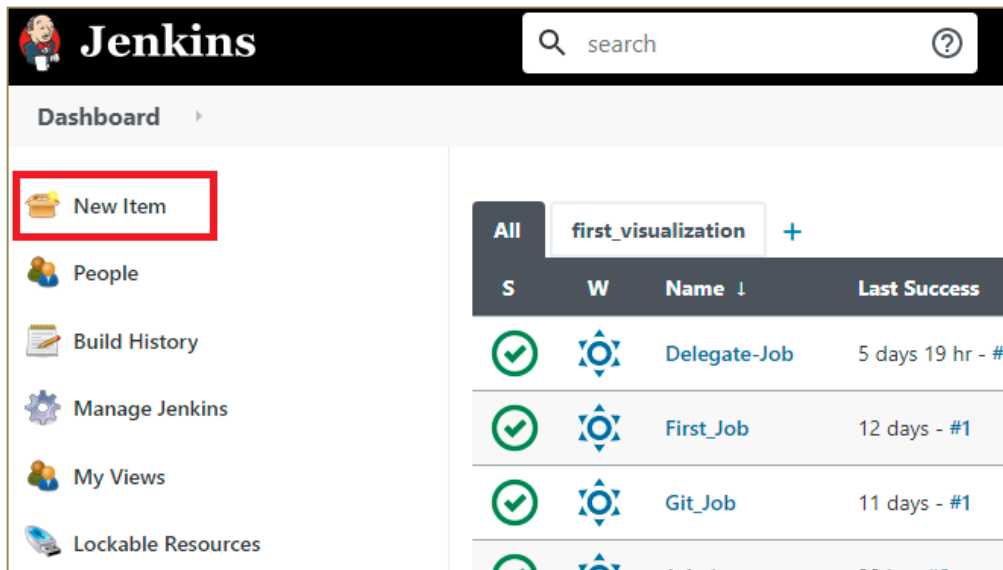
```
<user username="training" password="freefree" roles="manager-script,manager-status,manager-gui"/>
```

```
See the License for the specific language governing permissions and
limitations under the License.
-->
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">
<!--
NOTE: By default, no user is included in the "manager-gui" role required
to operate the "/manager/html" web application. If you wish to use this app,
you must define such a user - the username and password are arbitrary. It is
strongly recommended that you do NOT use one of the users in the commented out
section below since they are intended for use with the examples web
application.
-->
<!--
NOTE: The sample user and role entries below are intended for use with the
examples web application. They are wrapped in a comment and thus are ignored
when reading this file. If you wish to configure these users for use with the
examples web application, do not forget to remove the <!-- ... --> that surrounds
them. You will also need to set the passwords to something appropriate.
-->
<!--
<role rolename="tomcat"/>
<role rolename="role1"/>
<user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
<user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
<user username="role1" password="<must-be-changed>" roles="role1"/>
-->
<user username="training" password="freefree" roles="manager-script,manager-status,manager-gui"/>
</tomcat-users>
-- INSERT --
```

- d. Save and Quit the file by pressing `ecs` key and type `:wq` and press enter.
- e. Run the code to restart the service as shown below.

```
ubuntu@ip-172-31-46-223:/etc/tomcat9$ sudo service tomcat9 restart
ubuntu@ip-172-31-46-223:/etc/tomcat9$
```

5. Continuous Download START CI-CD.
- a. In the Jenkins Dashboard, navigate to main menu and select **New Item** to create a Deployment Job as shown below.



- b. Provide the name for Job, select **Freestyle project** and click **OK**.

The screenshot shows the 'Enter an item name' form. The name field contains 'Deploy_Job'. Below the field, the 'Freestyle project' option is selected and highlighted with a red rectangle. The 'OK' button at the bottom is also highlighted with a red rectangle.

- c. Navigate to **Source Code Management** and select **Git**. Enter the URL of the **GitHub** repository given below.

`https://github.com/oralabuser/WebApp.git`

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

☐ Execute concurrent builds if necessary ?

☐ Restrict where this project can be run ?

Advanced...

Source Code Management

☐ None

☒ **Git** ?

Repositories ?

Repository URL ?

https://github.com/oralabuser/WebApp.git

Credentials ?

- none - Add

- d. Change the **Branch Specifier** has **main** as shown below. Click **Apply** and **Save**.

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

Repository URL ?

https://github.com/oralabuser/WebApp.git

Credentials ?

- none - Add

Advanced...

Add Repository

Branches to build ?

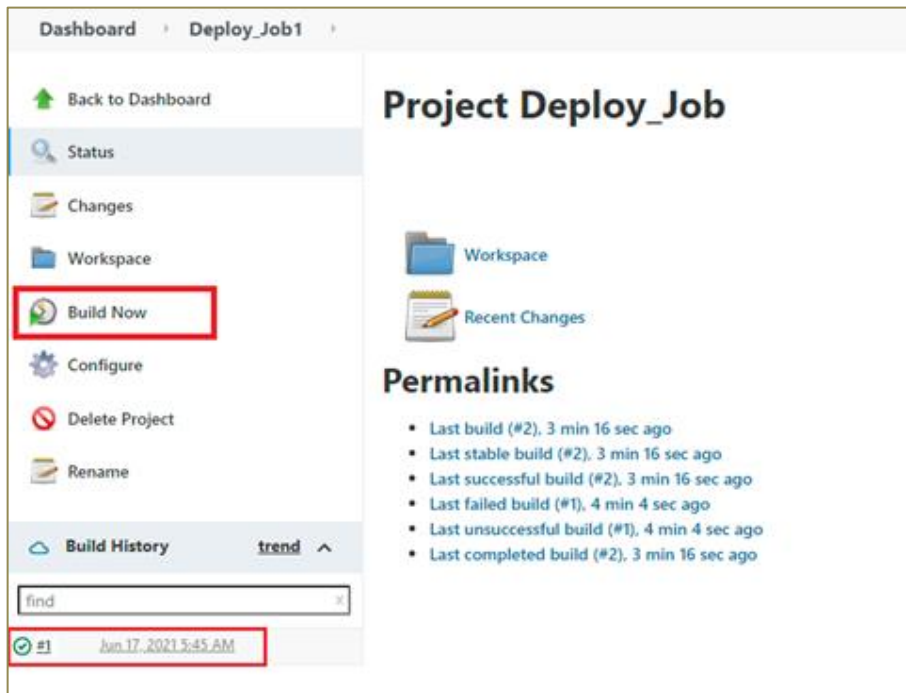
Branch Specifier (blank for 'any') ?

*/main

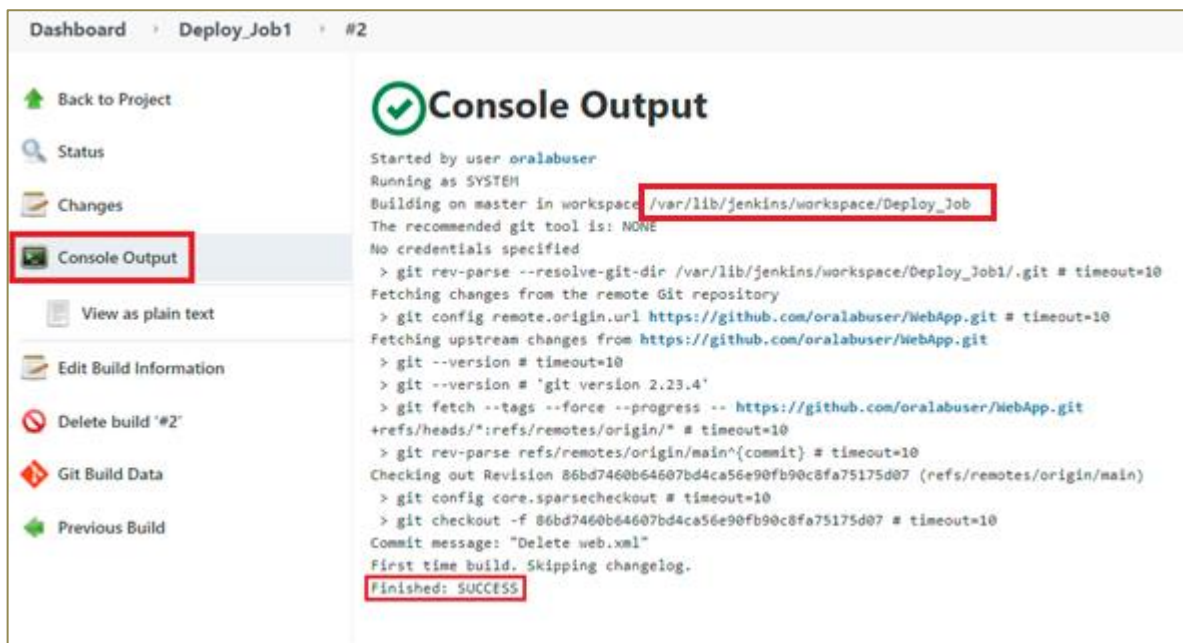
Add Branch

Save Apply

- e. Job is created successfully, click **Build Now** to execute the pipeline. Click on build created under **Build History** as shown below.



- f. In the **Build** page, click **Console Output** to view the output of the job. The path is provided where the code is downloaded.

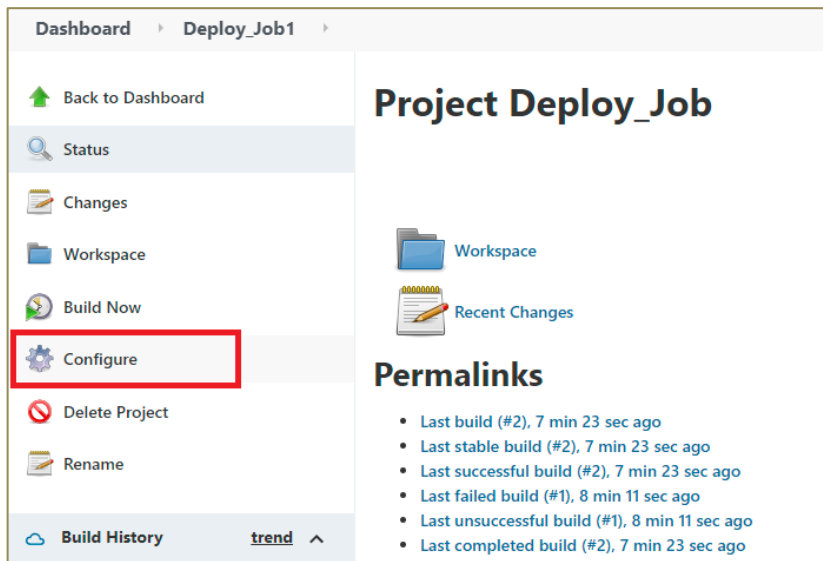


- g. In the terminal navigate to the respective path provided and list the files to verify as shown below.

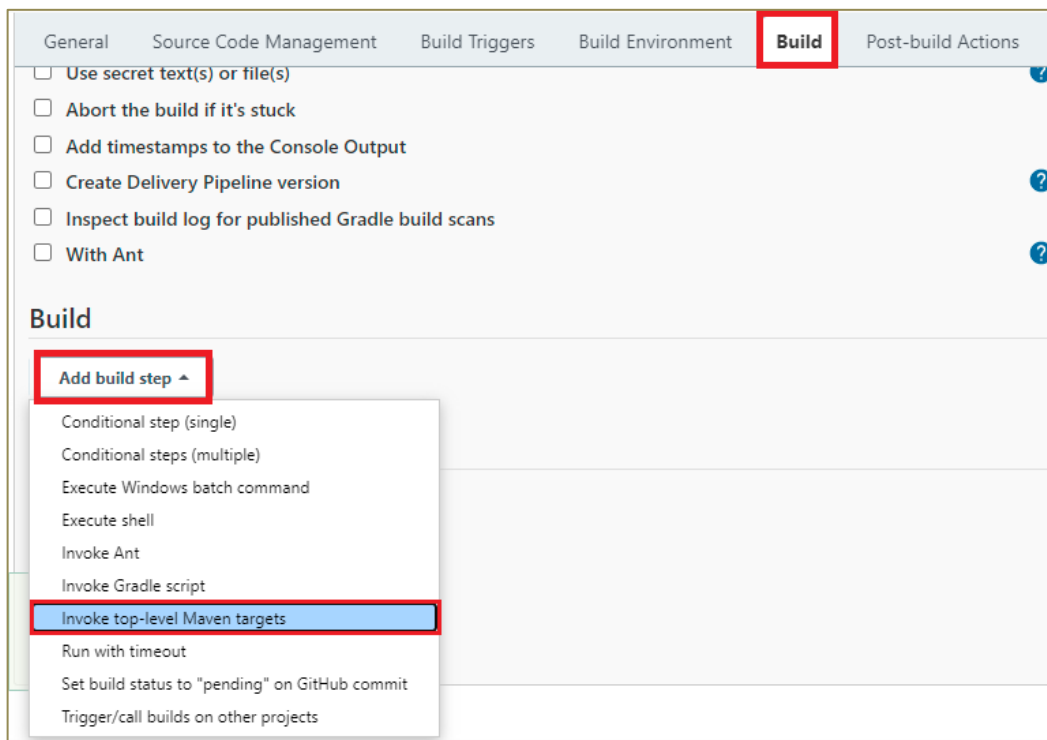
```
[ec2-user@ip-172-31-35-174 ~]$ cd /var/lib/jenkins/workspace/Deploy_Job
[ec2-user@ip-172-31-35-174 Deploy_Job]$ ls
DeclarativePipeline_Jenkinsfile  pom.xml                                server
git2                             README.md                             webapp
Jenkinsfile                     ScriptedPipeline_Jenkinsfile.txt
[ec2-user@ip-172-31-35-174 Deploy_Job]$
```


6. Continuous Build process using the Jenkins Job.

- a. Click on **Configure** of the same job **Deploy_Job** as shown below.



- b. Navigate to **Build** section, click **Add build step** and select **Invoke Top-level Maven targets** as shown below.



- c. Enter the **Goals** as **package**, click **Apply** and **Save** to update the configuration of the job as shown below.

The screenshot shows the Jenkins configuration interface for a build job. The 'Build Environment' tab is active. Under the 'Build' section, the 'Invoke top-level Maven targets' goal is set to 'package'. The 'Post-build Actions' section is empty. At the bottom, the 'Save' and 'Apply' buttons are highlighted with red boxes.

- d. Select **Build Now** to build the job again in Jenkins, click on the new build created under **Build History** as shown below

Jenkins search

Dashboard > Deploy_Job1 >

Back to Dashboard

Status

Changes

Workspace

Build Now

Configure

Delete Project

Rename

Build History trend ^

find

#2 Jun 17, 2021 7:35 AM

#1 Jun 17, 2021 5:45 AM

Project Deploy_Job

Workspace

Recent Changes

Permalinks

- Last build (#9), 21 sec ago
- Last stable build (#9), 21 sec ago
- Last successful build (#9), 21 sec ago
- Last failed build (#8), 4 min 30 sec ago
- Last unsuccessful build (#8), 4 min 30 sec ago
- Last completed build (#9), 21 sec ago

- e. Click **Console Output** to view the output of the job. The Console output is displayed as shown below.

Dashboard > Deploy_Job1 > #9

Back to Project

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#9'

Git Build Data

Previous Build

Console Output

```

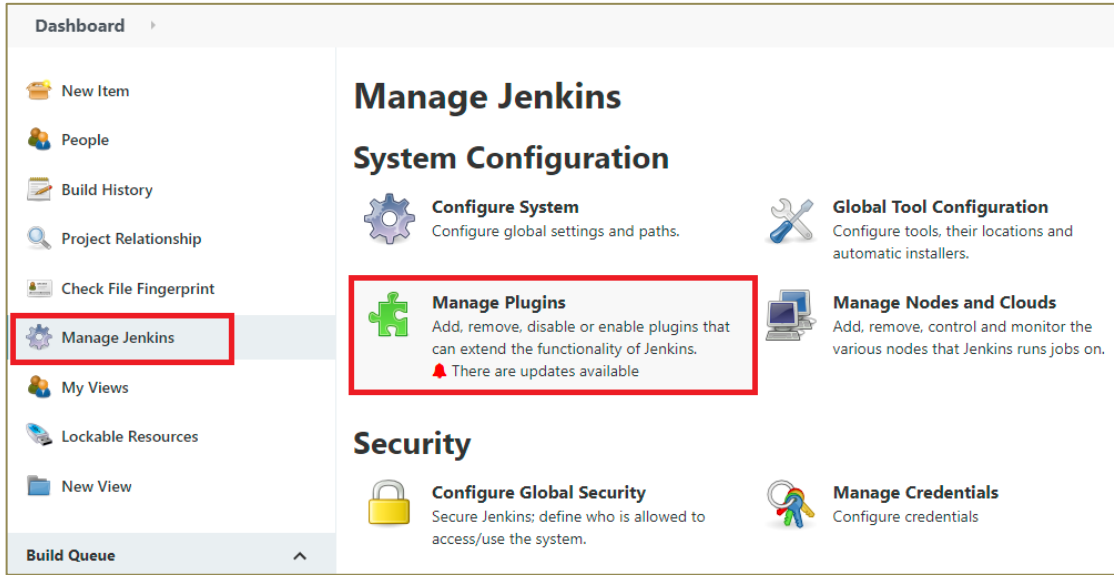
Started by user oralabuser
Running as SYSTEM
Building on master in workspace /var/lib/jenkins/workspace/Deploy_Job1
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Deploy_Job1/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/oralabuser/WebApp.git # timeout=10
Fetching upstream changes from https://github.com/oralabuser/WebApp.git
> git --version # timeout=10
> git --version # 'git version 2.23.4'
> git fetch --tags --force --progress -- https://github.com/oralabuser/WebApp.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 19cf4c2737e42bd91b98c79d9649504532830abb (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 19cf4c2737e42bd91b98c79d9649504532830abb # timeout=10
Commit message: "Add files via upload"
> git rev-list --no-walk 86bd7460b64607bd4ca56e90fb90c8fa75175d07 # timeout=10
[Deploy_Job1] $ mvn package
[INFO] Scanning for projects...
[INFO] -----
[INFO] Reactor Build Order:
  
```

```
Results :

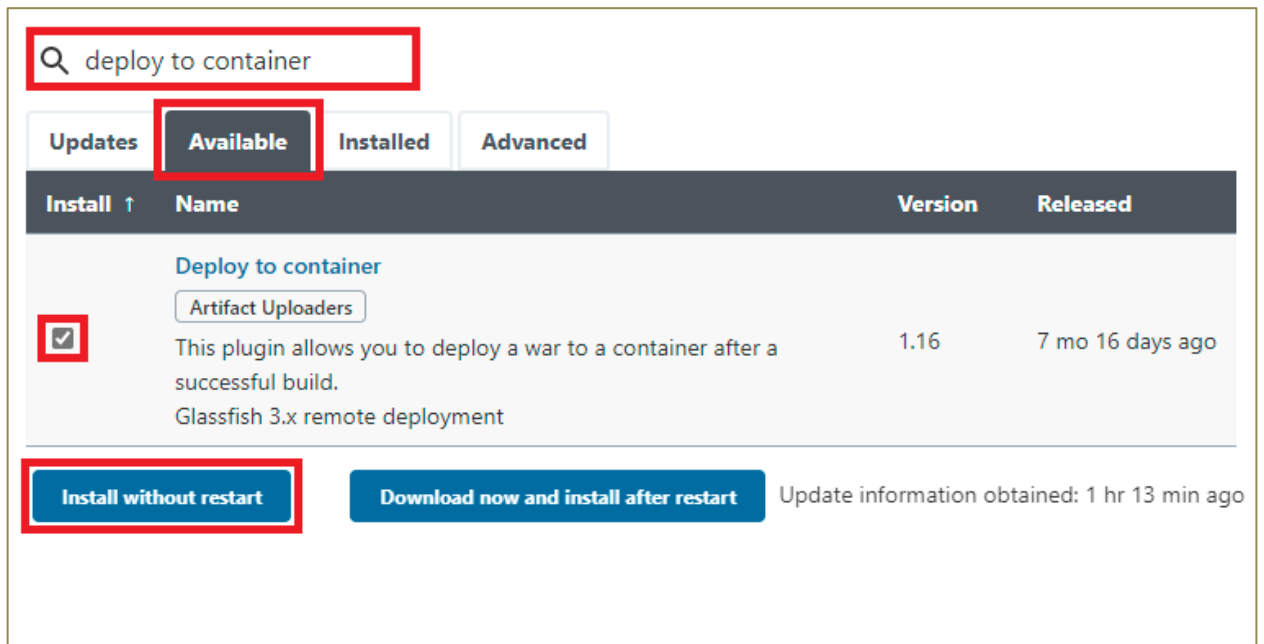
Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-war-plugin:2.1.1:war (default-war) @ webapp ---
[INFO] Packaging webapp
[INFO] Assembling webapp [webapp] in [/var/lib/jenkins/workspace/Deploy_Job1/webapp/target/webapp]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/Deploy_Job1/webapp/src/main/webapp]
[INFO] Webapp assembled in [34 msecs]
[INFO] Building war: /var/lib/jenkins/workspace/Deploy_Job1/webapp/target/webapp.war
[INFO] WEB-INF/web.xml already added, skipping
[INFO] -----
[INFO] Reactor Summary:
[INFO]
[INFO] Maven Project ..... SUCCESS [0.002s]
[INFO] Server ..... SUCCESS [3.585s]
[INFO] Webapp ..... SUCCESS [0.933s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 4.707s
[INFO] Finished at: Thu Jun 17 07:35:56 UTC 2021
[INFO] Final Memory: 15M/38M
[INFO] -----
Finished: SUCCESS
```

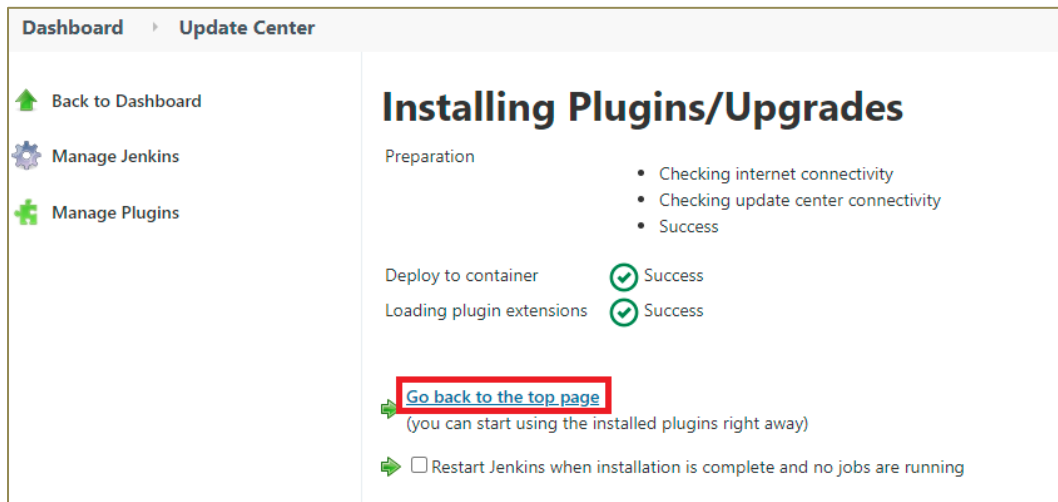
7. Continuous Deployment process using the Jenkins Job.
- a. To deploy the war file into the Web-Server first install the "deploy to container" plugin. Navigate to Dashboard, select **Manage Jenkins** and click **Manage Plugins** as shown below.



- b. Select **Available**, search for “Deploy to container” and select the check box of the plugin. Click **Install without restart**.



- c. The installation process will proceed and Success message will be displayed as shown below.



- d. Navigate to **Dashboard** and select **Deploy_Job** to configure the continues deployment in the job as shown below.

Dashboard

New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Lockable Resources

New View

add description

All

First visualization

+

| S | W | Name ↓ | Last Success | Last Failure | Last Duration | |
|---|---|--------------------------|------------------|------------------|---------------|---|
| ✓ | ☁ | Delegate-Job | 1 day 21 hr - #1 | 1 day 21 hr - #2 | 0.89 sec | 🔄 |
| ✓ | ☁ | Deploy_Job | 2 hr 28 min - #9 | 4 hr 19 min - #1 | 6.8 sec | 🔄 |
| ✓ | ⚙ | First_Job | 1 day 23 hr - #1 | N/A | 0.19 sec | 🔄 |
| ✓ | ⚙ | Git_Job | 1 day 22 hr - #2 | N/A | 1.3 sec | 🔄 |
| ✓ | ⚙ | GitHub_Pipeline_Scripted | 1 day 0 hr - #1 | N/A | 3.6 sec | 🔄 |

- e. Select **Configure** to configure the Job.

Dashboard > Deploy_Job1

Back to Dashboard

Status

Changes

Workspace

Build Now

Configure

Delete Project

Rename

Project Deploy_Job

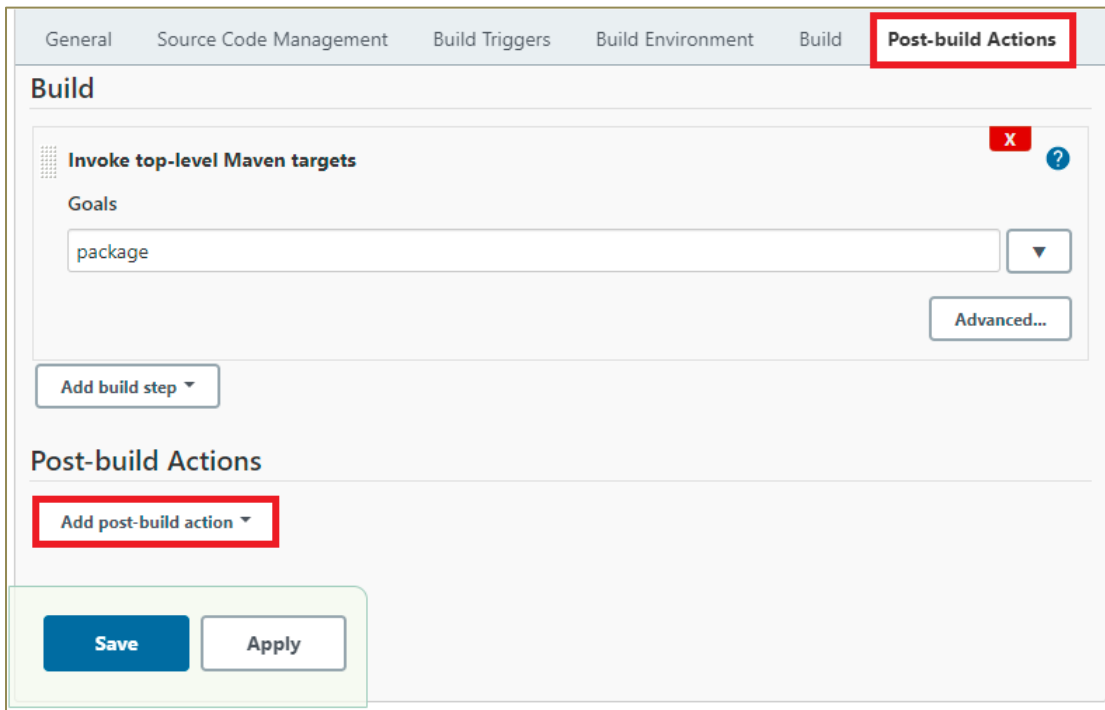
Workspace

Recent Changes

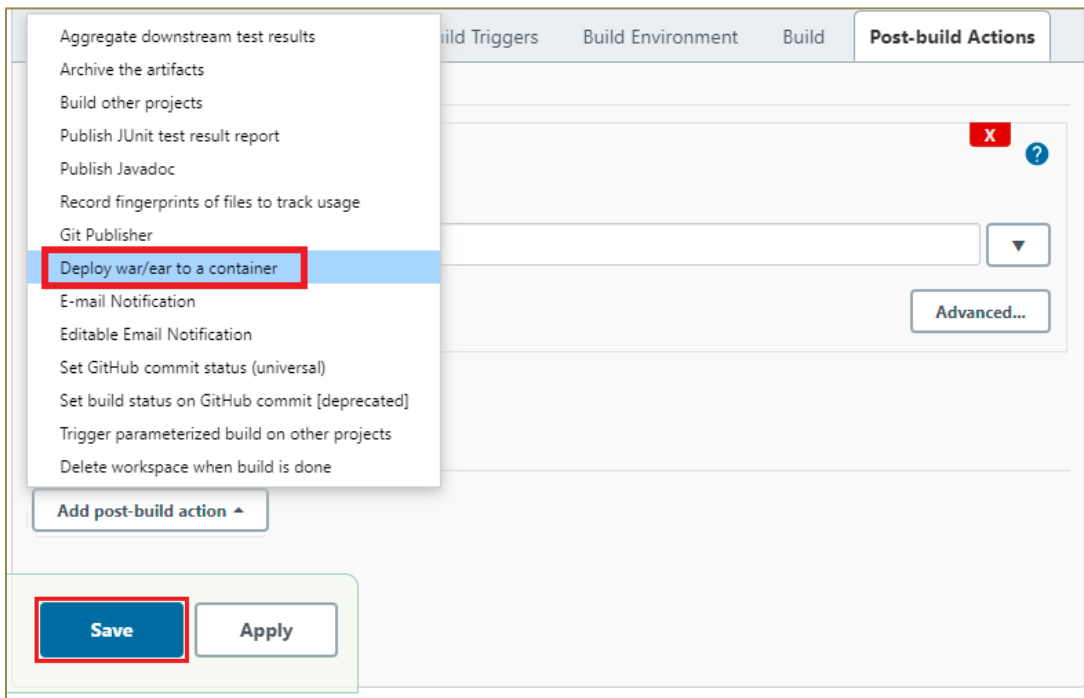
Permalinks

- Last build (#9), 2 hr 30 min ago
- Last stable build (#9), 2 hr 30 min ago
- Last successful build (#9), 2 hr 30 min ago
- Last failed build (#1), 4 hr 21 min ago

- f. Navigate to **Post-build Actions** and select **Add post-build actions** as shown below



- g. Select **Deploy war/ear to container** as shown below.



- h. Enter the path of the war file or provide the ****/*.war** in **war/ear files** and provide the **Context path** as shown below.

The screenshot shows the Jenkins 'Post-build Actions' configuration page. The 'Deploy war/ear to a container' action is selected. The 'WAR/EAR files' field contains '**/*.war'. The 'Context path' field contains 'webserverenv'. The 'Containers' section has an 'Add Container' button. The 'Deploy on failure' checkbox is unchecked. At the bottom, there are 'Save' and 'Apply' buttons.

- i. Select **Containers** as **Tomcat 9** (installed Tomcat version in server).

This screenshot shows the same Jenkins configuration page as above, but with the 'Add Container' dropdown menu open. The menu lists various container options: JBoss AS 3.x, JBoss AS 4.x, JBoss AS 5.x, JBoss AS 6.x, JBoss AS 7.x, Tomcat 4.x Remote, Tomcat 5.x Remote, Tomcat 6.x Remote, Tomcat 7.x Remote, Tomcat 8.x Remote, and Tomcat 9.x Remote. The 'Tomcat 9.x Remote' option is highlighted with a blue background and a red border.

- j. To provide **credentials** click **Add** and select **Jenkins** as shown below.

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

WAR/EAR files ?

**/*.*.war

Context path ?

webserverenv

Containers

Tomcat 9.x Remote X

Credentials

- none - Add

Tomcat URL ? Jenkins

Advanced...

Add Container

Deploy on failure

Save Apply

- k. Provide the Tomcat **Username** and **Password** as shown below. Click **Add**.

Username: training

Password: freefree

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

Scope ?

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

Username ?

training

☐ Treat username as secret ?

Password ?

freefree

ID ?

Description ?

Add Cancel

Deploy on failure

- l. Select **Credentials** provided, give the private IP address of the Web-Server instance has **http://private_ip:8080** as shown below. Click **Apply** and **Save**.

The screenshot shows the Jenkins configuration interface for a 'Tomcat 9.x Remote' build job. The 'General' tab is active. The 'Credentials' dropdown menu is set to 'training/*****' and the 'Tomcat URL' text field contains 'http://172.31.46.223:8080'. Both the dropdown and the URL field are highlighted with red rectangles. Below these fields, there is an 'Add Container' button, a checkbox for 'Deploy on failure' (which is unchecked), and an 'Add post-build action' button. At the bottom of the configuration area, there are two buttons: 'Save' (a blue button) and 'Apply' (a white button with a red border). Both buttons are highlighted with red rectangles. The top of the interface shows tabs for 'General', 'Source Code Management', 'Build Triggers', 'Build Environment', 'Build', and 'Post-build Actions'. A red 'X' icon is visible in the top right corner of the configuration area.

- m. Click **Build Now** to build the job in Jenkins and click on the latest build job link provided under **Build History** to verify the execution as shown below.

Dashboard
Deploy_Job1

Back to Dashboard

Status

Changes

Workspace

Build Now

Configure

Delete Project

Rename

Build History

trend ^

find

#3 Jun 17, 2021 10:34 AM

#2 Jun 17, 2021 7:35 AM

#1 Jun 17, 2021 5:45 AM

Project Deploy_Job

Workspace
Recent Changes

Permalinks

- Last build (#9), 2 hr 59 min ago
- Last stable build (#9), 2 hr 59 min ago
- Last successful build (#9), 2 hr 59 min ago
- Last failed build (#1), 4 hr 49 min ago
- Last unsuccessful build (#1), 4 hr 49 min ago
- Last completed build (#9), 2 hr 59 min ago

- n. Select **Console Output** to view the execution as shown below.

Dashboard
Deploy_Job1
#10

Back to Project
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#10'
Git Build Data
Previous Build

Console Output

Started by user [oralabuser](#)
Running as SYSTEM
Building on master in workspace /var/lib/jenkins/workspace/Deploy_Job1
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Deploy_Job1/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url <https://github.com/oralabuser/WebApp.git> # timeout=10
Fetching upstream changes from <https://github.com/oralabuser/WebApp.git>
> git --version # timeout=10
> git --version # 'git version 2.23.4'
> git fetch --tags --force --progress -- <https://github.com/oralabuser/WebApp.git>
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 19cf4c2737e42bd91b98c79d9649504532830abb (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 19cf4c2737e42bd91b98c79d9649504532830abb # timeout=10
Commit message: "Add files via upload"
> git rev-list --no-walk 19cf4c2737e42bd91b98c79d9649504532830abb # timeout=10
[Deploy_Job1] \$ mvn package
[INFO] Scanning for projects...

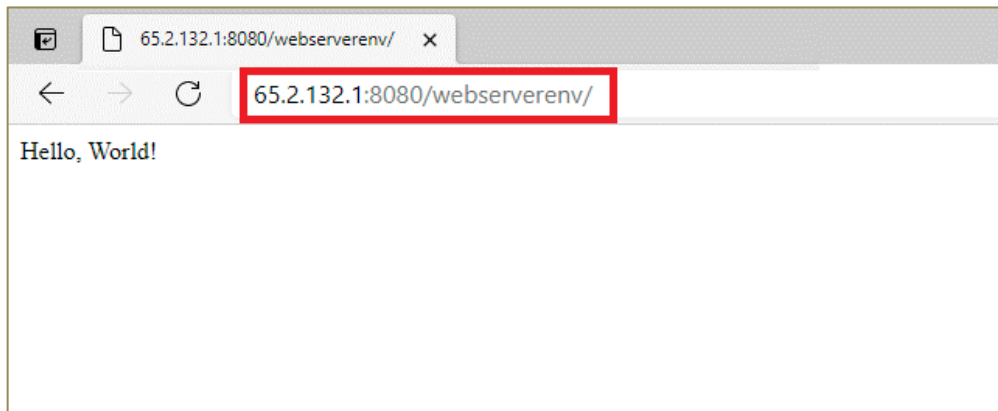
```

[INFO] Maven Project ..... SUCCESS [0.001s]
[INFO] Server ..... SUCCESS [2.141s]
[INFO] Webapp ..... SUCCESS [1.014s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.336s
[INFO] Finished at: Thu Jun 17 10:35:04 UTC 2021
[INFO] Final Memory: 7M/19M
[INFO] -----
[DeployPublisher][INFO] Attempting to deploy 1 war file(s)
[DeployPublisher][INFO] Deploying /var/lib/jenkins/workspace/Deploy_Job1/webapp/target/webapp.war
to container Tomcat 9.x Remote with context webserverenv
  Redeploying [/var/lib/jenkins/workspace/Deploy_Job1/webapp/target/webapp.war]
  Undeploying [/var/lib/jenkins/workspace/Deploy_Job1/webapp/target/webapp.war]
  Deploying [/var/lib/jenkins/workspace/Deploy_Job1/webapp/target/webapp.war]
Finished: SUCCESS

```

- o. To access the home page verify it in the browser by providing the **Public_IP** of **Web-Server** with the port **8080** followed by the **Context path**.

<Public_IP of Web-Server>:8080/webserverenv/



8. Keep the Jenkins Dashboard, Linux-instance terminal and the AWS Management Console open for the next practice.