CI/CD Deployment Using Ansible CM Tool

by
Anjali Lalwani

Problem Statement

You are a DevOps engineer at XYZ Ltd. Your company is working on a Java application and wants to automate WAR file artifact deployment so that they don't have to perform WAR deployment on Tomcat/Jetty web containers. Automate Ansible integration with Jenkins CI server so that we can run and execute playbooks to deploy custom WAR files to a web container and then perform restart for the web container.

Steps to Perform:

1. Configure Jenkins server as Ansible provisioning machine
2. Install Ansible plugins in Jenkins CI server
3. Prepare Ansible playbook to run Maven build on Jenkins CI server
4. Prepare Ansible playbook to execute deployment steps on the remote web container with restart of the web container post deployment

Goal

The goal of the project is to implement a CI/CD pipeline for a Java application using Ansible and Jenkins. The project aims to automate the deployment process, specifically for a WAR to a web container i.e TomCat.

The subsequent sections in this document highlight the steps required to achieve this objective.

Prerequisites:

Make sure that both Jenkins and Ansible are installed on the same server. Install Jenkins/Ansible on the control server if not already installed.

.

STEPS:

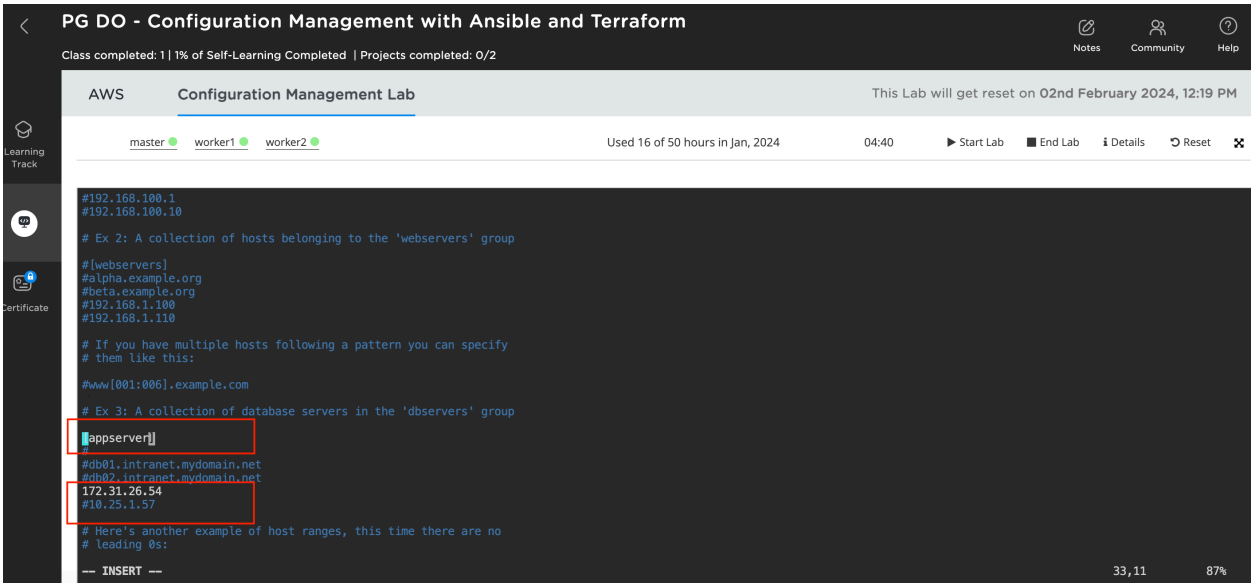
Step 1: Establish test connection with the node machine from Jenkins/Ansible controller machine.

Add the ip of the node machine to the hosts file.

sudo vi /etc/ansible/hosts

Add -

[appserver]
172.31.26.54



Run the below command to check the connectivity with node machine:

ansible -m ping appserver

```
labsuser@ip-172-31-19-180:~$ ansible -m ping appserver
172.31.26.54 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
labsuser@ip-172-31-19-180:~$
```

Step 2: Install Maven on the control server.

Use the following command to install maven on the control server.

sudo apt-get install maven

```
labsuser@ip-172-31-19-180:~$ sudo apt-get install maven
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm11
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java libcdi-api-java libcommons-cli-java libcommons-io-java libcommons-lang3-java libcommons-parent-java
  libgeronimo-annotation-1.3-spec-java libgeronimo-interceptor-3.0-spec-java libguava-java libguice-java libhawtjni-runtime-java libjansi-java libjansi-native-java libjsr305-java
  libmaven-parent-java libmaven-resolver-java libmaven-shared-utils-java libmaven3-core-java libplexus-cipher-java libplexus-classworlds-java libplexus-component-annotations-java
  libplexus-interpolation-java libplexus-sec-dispatcher-java libplexus-utils2-java libsisu-inject-java libsisu-plexus-java libslf4j-java libwagon-file-java libwagon-http-shaded-java
  libwagon-provider-api-java
Suggested packages:
  libaopalliance-java-doc libatinject-jsr330-api-java-doc libservlet3.1-java libcommons-io-java-doc libcommons-lang3-java-doc libasm-java libcglib-java libjsr305-java-doc
  libmaven-shared-utils-java-doc liblogback-java libplexus-cipher-java-doc libplexus-classworlds-java-doc libplexus-sec-dispatcher-java-doc libplexus-utils2-java-doc junit4 testng
  libcommons-logging-java liblog4j1.2-java
The following NEW packages will be installed:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java libcdi-api-java libcommons-cli-java libcommons-io-java libcommons-lang3-java libcommons-parent-java
  libgeronimo-annotation-1.3-spec-java libgeronimo-interceptor-3.0-spec-java libguava-java libguice-java libhawtjni-runtime-java libjansi-java libjansi-native-java libjsr305-java
  libmaven-parent-java libmaven-resolver-java libmaven-shared-utils-java libmaven3-core-java libplexus-cipher-java libplexus-classworlds-java libplexus-component-annotations-java
  libplexus-interpolation-java libplexus-sec-dispatcher-java libplexus-utils2-java libsisu-inject-java libsisu-plexus-java libslf4j-java libwagon-file-java libwagon-http-shaded-java
  libwagon-provider-api-java maven
0 upgraded, 33 newly installed, 0 to remove and 314 not upgraded.
Need to get 9657 kB of archives.
After this operation, 12.6 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libapache-pom-java all 18-1 [4720 B]
Get:2 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libatinject-jsr330-api-java all 1.0+ds1-5 [5348 B]
Get:3 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libgeronimo-interceptor-3.0-spec-java all 1.0.1-4fakesync [8616 B]
Get:4 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libcdi-api-java all 1.2-2 [54.5 kB]
Get:5 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libcommons-cli-java all 1.4-1 [53.8 kB]
Get:6 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libcommons-parent-java all 43-1 [10.8 kB]
Get:7 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal-updates/universe amd64 libcommons-io-java all 2.6-2ubuntu0.20.04.1 [198 kB]
Get:8 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libcommons-lang3-java all 3.8-2 [458 kB]
Get:9 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libgeronimo-annotation-1.3-spec-java all 1.0-1 [10.7 kB]
Get:10 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libjsr305-java all 0.1~+svn49-11 [27.0 kB]
Get:11 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libguava-java all 19.0-1 [2028 kB]
Get:12 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libaopalliance-java all 20070526-6 [9084 B]
Get:13 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal-updates/universe amd64 libguice-java all 4.2.1-1ubuntu0.2 [1410 kB]
Get:14 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libhawtjni-runtime-java all 1.17-1 [28.8 kB]
Get:15 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libjansi-native-java all 1.8-1 [23.8 kB]
Get:16 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libjansi-java all 1.18-1 [56.8 kB]
Get:17 http://us-west-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libmaven-parent-java all 31-2 [5140 B]
```

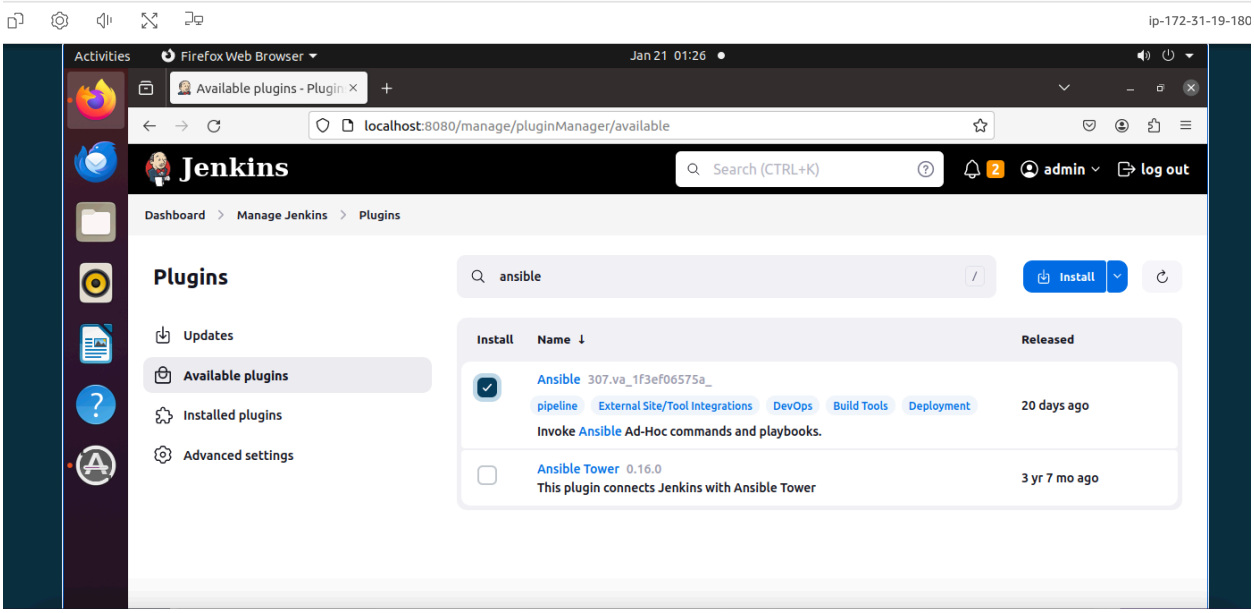
Check if maven is installed.

mvn -v

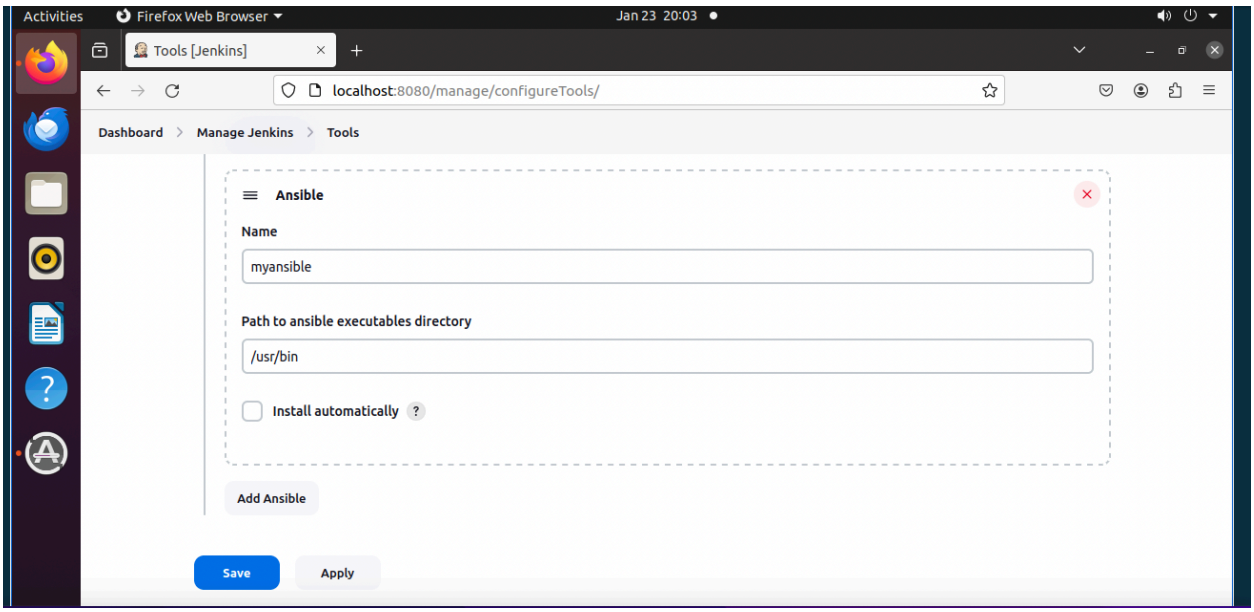
```
labsuser@ip-172-31-19-180:~$ mvn -v
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.21, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "5.15.0-1051-aws", arch: "amd64", family: "unix"
labsuser@ip-172-31-19-180:~$
```

Step 2: Install Ansible Plugins in Jenkins CI Server.

- Open Jenkins and navigate to "Manage Jenkins" > "Manage Plugins."In the "Available" tab, search for "Ansible" and install the "Ansible" plugin.



- Navigate to "Manage Jenkins" > “Tools” and under the “Ansible” section make the following configuration.



- Enable sudo access for Jenkins user ID so that it can connect with Ansible to trigger the playbook.

sudo bash -c 'echo "jenkins ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers'

Step 3: Prepare Ansible Playbook to Run Maven Build on Jenkins CI Server:

Create an Ansible playbook build.yaml.

```
---
- name: Build the war file using Maven
  hosts: localhost
  become: true
  tasks:
```

```
  - name: Clone Git repository
    git:
      repo: https://github.com/AnjaliLalwaniGit/hello-world-war.git
      dest: /mytmp/config_mng_proj1/
  - name:  Maven to build the code
    command: mvn chdir=/mytmp/config_mng_proj1/ clean install
```

Step 4: Prepare Ansible Playbook to Execute Deployment Steps on the Remote Web Container along with Tomcat Installation.

Create another Ansible playbook deploy_war.yml for deploying the WAR file to the web container and restarting it. This playbook includes tasks to install Java and Tomcat on the web container.

```
---

- hosts: appserver
  vars:
    - http_port: 8080
    - tomcat_version: 9.0.50
  become: true
  tasks:
    - name: Install OpenJDK
      apt: name=openjdk-8-jdk state=latest

    - name: add group "tomcat"
```

```
      group: name=tomcat

    - name: add user "tomcat"

      user: name=tomcat group=tomcat createhome=yes

    - name: Download Tomcat archive

      get_url:

        url: "https://archive.apache.org/dist/tomcat/tomcat-9/
v{{ tomcat_version }}/bin/apache-tomcat-{{ tomcat_version }}.tar.gz"

        dest: "/opt/"

        mode: 0755

      become: true

    - name: Extract Tomcat archive

      command: tar zxvf /opt/apache-tomcat-{{ tomcat_version }}.tar.gz
-C /opt/ creates=/opt/apache-tomcat-{{ tomcat_version }}

    - name: Change ownership of Tomcat installation

      file: path=/opt/apache-tomcat-{{ tomcat_version }} owner=tomcat
group=tomcat state=directory recurse=yes

      register: result

    - name: Change the working directory to Tomcat Apache before
running Tomcat Apache

      shell: ./bin/startup.sh

      args:

        chdir: /opt/apache-tomcat-{{ tomcat_version }}

      become: true
```

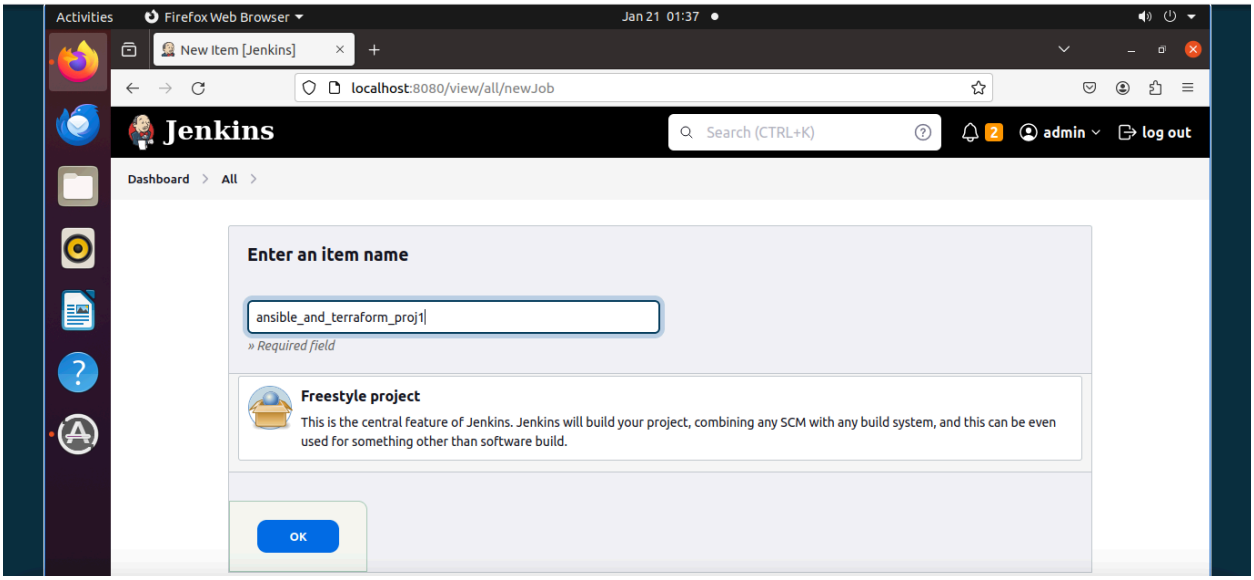
```
    become_user: tomcat
  - name: Copy Tomcat systemd service file
    copy:
      content: |
        [Unit]
        Description=Apache Tomcat 9.0.50 Web Application Container
        After=network.target
        [Service]
        Type=forking
        Environment=JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
        Environment=CATALINA_PID=/opt/apache-tomcat-9.0.50/temp/
tomcat.pid
        Environment=CATALINA_HOME=/opt/apache-tomcat-9.0.50
        Environment=CATALINA_BASE=/opt/apache-tomcat-9.0.50
        Environment='CATALINA_OPTS=-Xms512M -Xmx1024M -server
-XX:+UseParallelGC'
        Environment='JAVA_OPTS=-Djava.awt.headless=true
-Djava.security.egd=file:/dev/./urandom'
        ExecStart=/opt/apache-tomcat-9.0.50/bin/startup.sh
        ExecStop=/opt/apache-tomcat-9.0.50/bin/shutdown.sh
         User=tomcat
        Group=tomcat
```

```
      UMask=0007

      RestartSec=10

      Restart=always

      [Install]

      WantedBy=multi-user.target
    dest: /etc/systemd/system/tomcat.service
  notify:
    - Reload SystemD
- name: Start Tomcat service
  systemd:
    name: tomcat
    state: started

- name: Copy WAR file to web container
  copy:
    src: /mytmp/config_mng_proj1/target/hello-world-war-1.0.0.war
    dest: "/opt/apache-tomcat-{{ tomcat_version }}/webapps/"
- name: Restart Tomcat service
  service:
    name: tomcat
    state: restarted
```

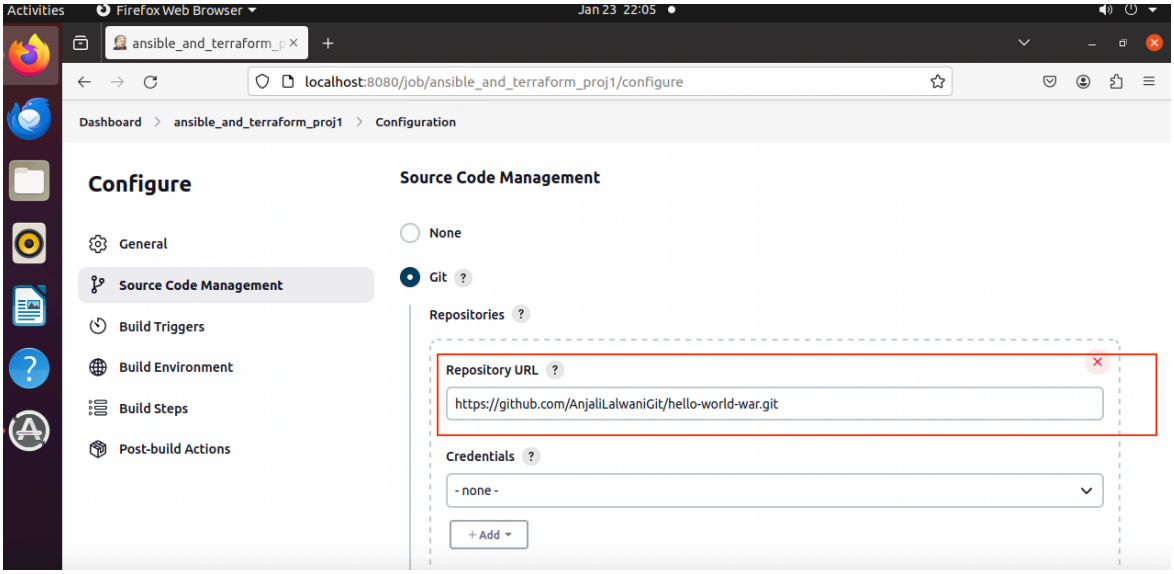
```
handlers:
  - name: Reload SystemD
    systemd:
      name: tomcat
      state: reloaded
```

Step 4: Create a new Jenkins job for your CI/CD pipeline.

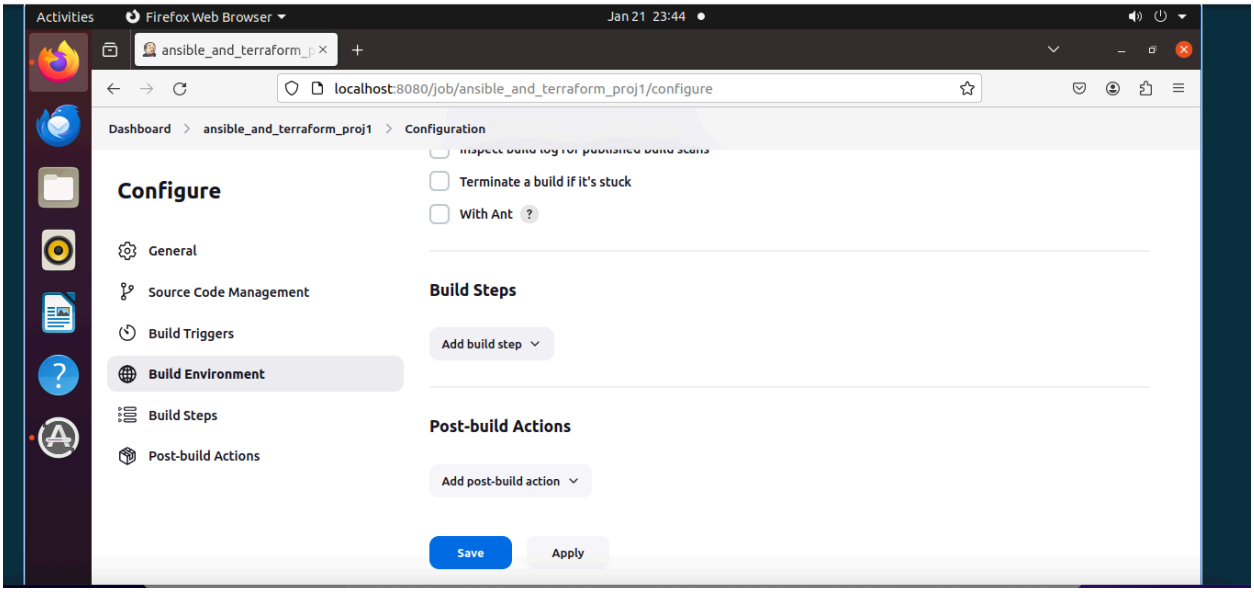
Create a new freestyle Jenkins project “ansible_and_terraform_proj1”.

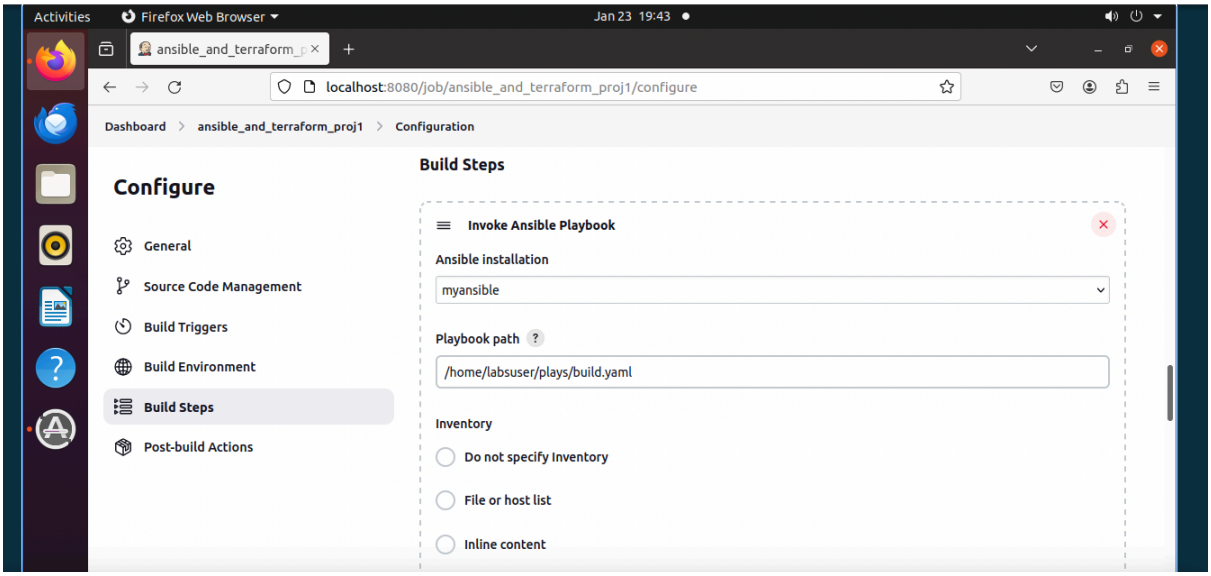


Configure the job to use Git and add the path to your repo which contains the source for building the war file.

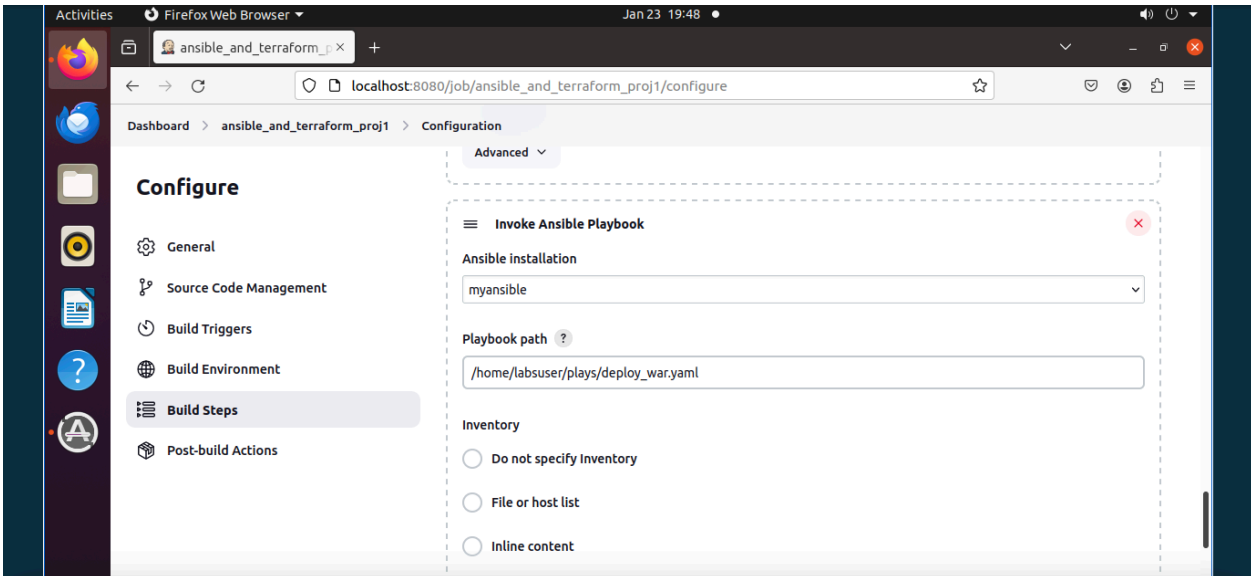


Add a build step to run the ansible playbook “build.yaml”.



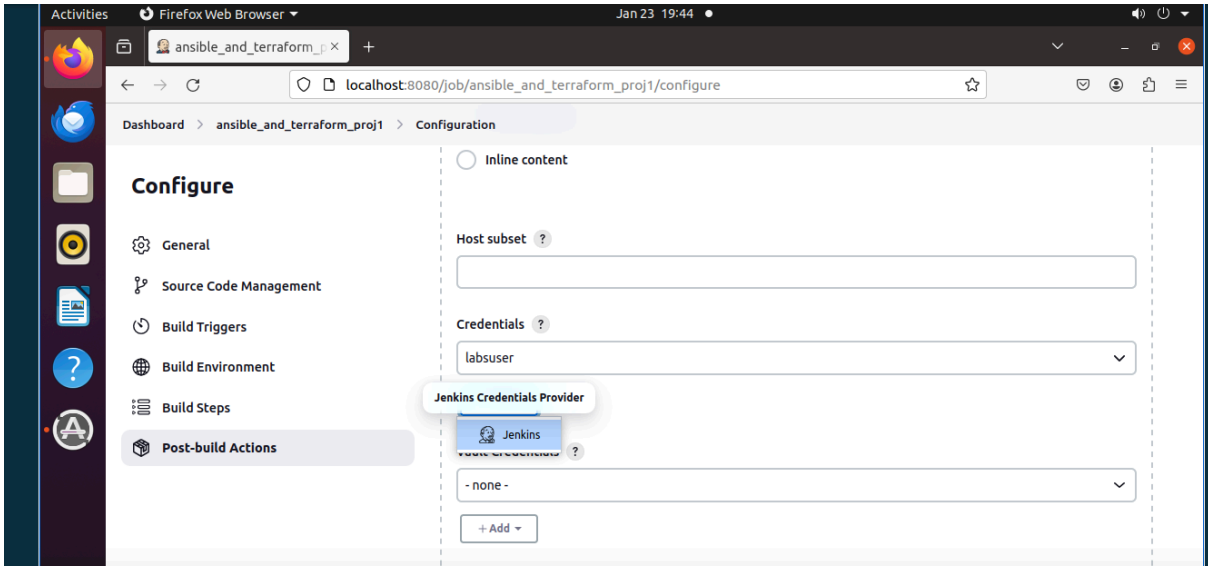


Add another build step to run the ansible playbook “deploy_war.yaml”. This playbook will install tomcat on the web container, copy the war file to the web container and restart the tomcat service.

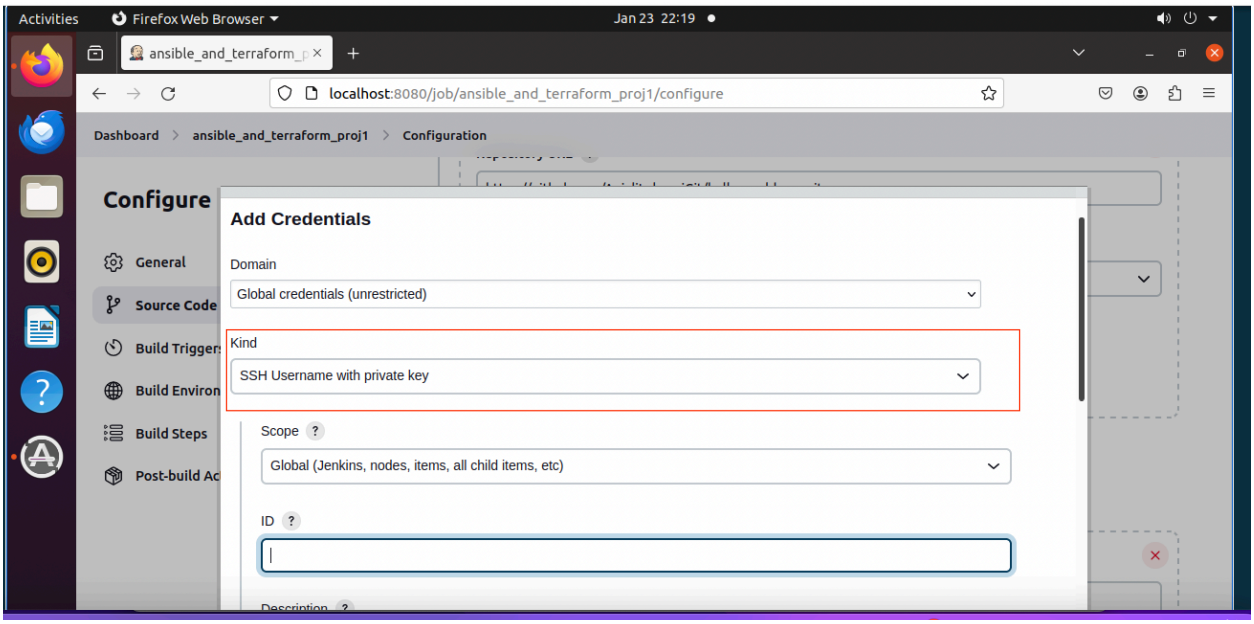


To allow the “Jenkins” user to install and start the tomcat service on the remote web container, add the following ssh configuration.

Under ‘Invoke Ansible Playbook’, select Credentials.



Select SSH Username with private key. Enter the necessary details, including the username associated with the SSH key and the private key (generated via ssh-keygen).

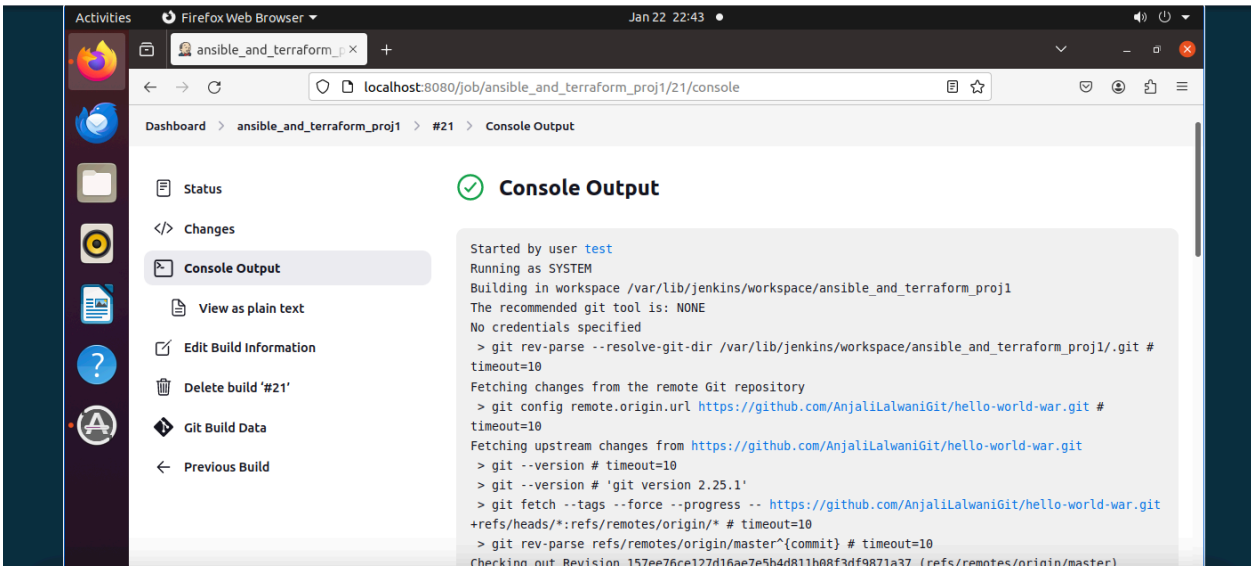


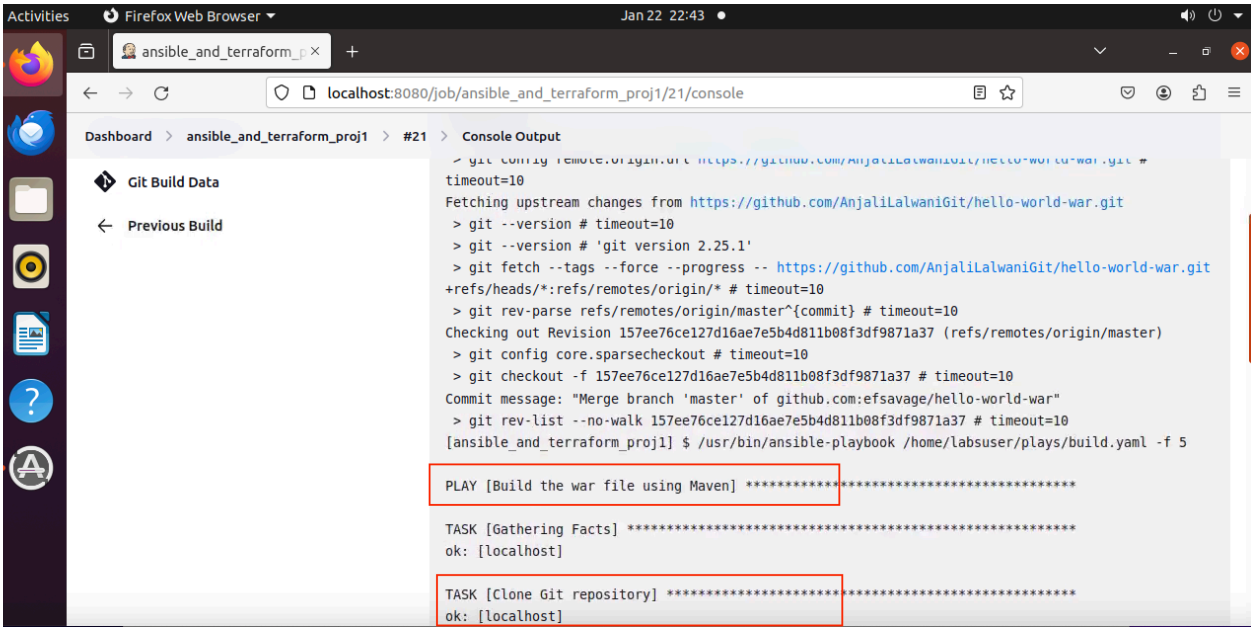
Save the configuration after providing the required information.

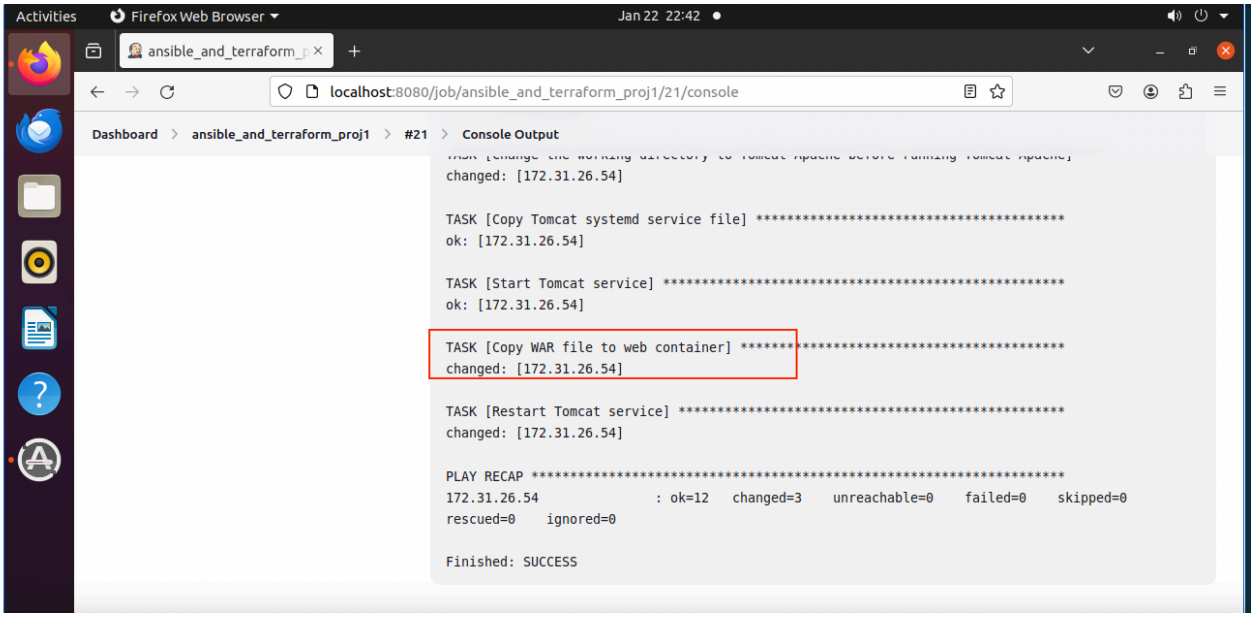
Save all the changes made to the job configuration.

Step 5: Run the Jenkins Job.

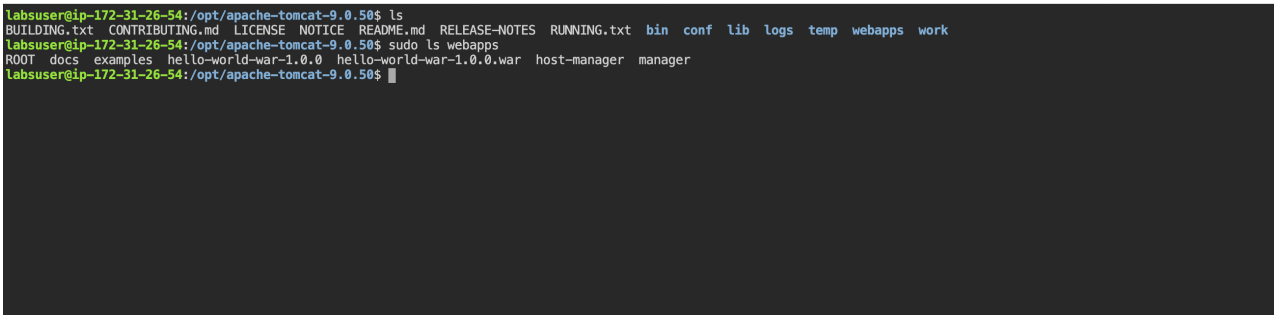
Build the job and observe the console output.







Verfiy that the war file is copied to the webapps directiory of the tomcat installation location on the remote web container.



CONCLUSION

In this project, we have successfully automated the deployment process for a Java application i.e the deployment of WAR file to a remote web container. By integrating Ansible and Jenkins integration, we have established an efficient and reliable pipeline for deploying WAR files to remote web containers.