AnsibleCI-CD

Description

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.

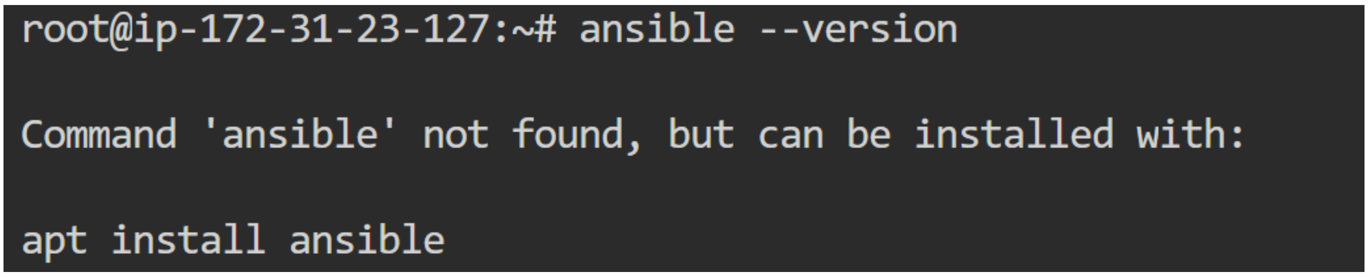
Steps to Perform:

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

Solution

1) Check if Ansible is installed

```
ansible --version
```

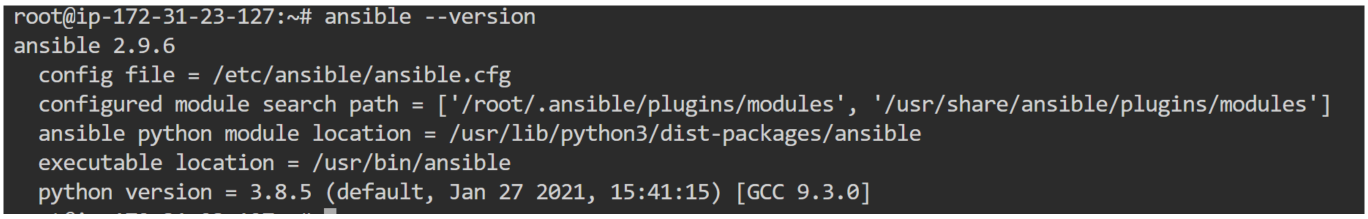


```
root@ip-172-31-23-127:~# ansible --version

Command 'ansible' not found, but can be installed with:

apt install ansible
```

We then install it.

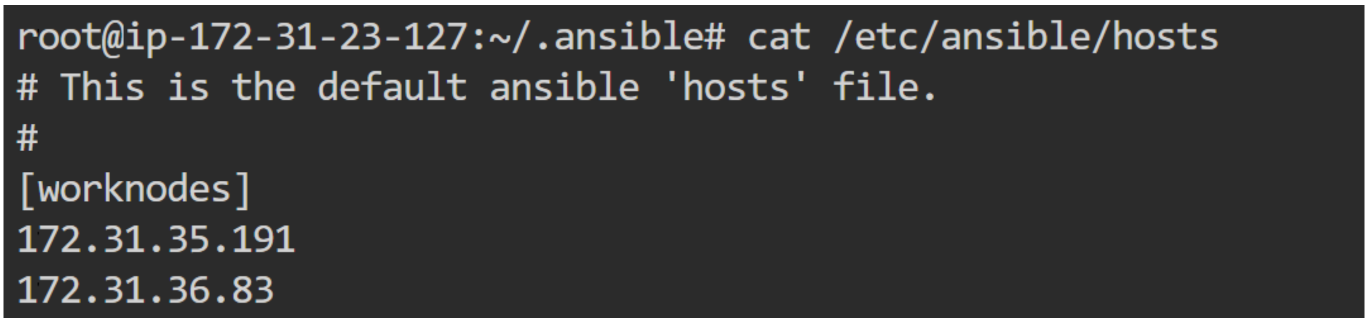
```
sudo apt install ansible
```

Then we check installation



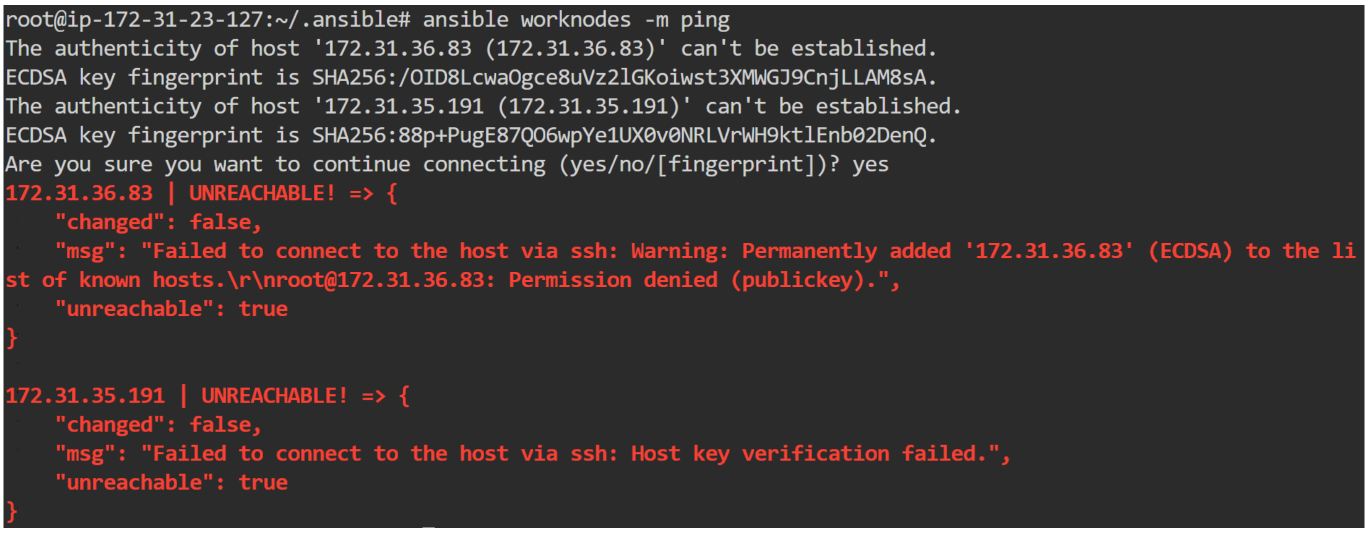
```
root@ip-172-31-23-127:~# ansible --version
ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.5 (default, Jan 27 2021, 15:41:15) [GCC 9.3.0]
```

Then we can modify our `/etc/ansible/hosts` file in order to create a group of worknodes



```
root@ip-172-31-23-127:~/.ansible# cat /etc/ansible/hosts
# This is the default ansible 'hosts' file.
#
[worknodes]
172.31.35.191
172.31.36.83
```

After configuring our inventory file we check if it is working and if we have connectivity with our worker nodes.



```
root@ip-172-31-23-127:~/.ansible# ansible worknodes -m ping
The authenticity of host '172.31.36.83 (172.31.36.83)' can't be established.
ECDSA key fingerprint is SHA256:/OID8LcwaOgce8uVz2lGKoiwst3XMWGJ9CnjLLAM8sA.
The authenticity of host '172.31.35.191 (172.31.35.191)' can't be established.
ECDSA key fingerprint is SHA256:88p+PugE87QO6wpYe1UX0v0NRLVrWH9ktlEnb02DenQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
172.31.36.83 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: Warning: Permanently added '172.31.36.83' (ECDSA) to the li
st of known hosts.\r\nroot@172.31.36.83: Permission denied (publickey).",
    "unreachable": true
}

172.31.35.191 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: Host key verification failed.",
    "unreachable": true
}
```

> Since both failed it means we need to generate a new ssh key and we will use it through a new ansisuser.

We will do the same three steps on each Node.

1) Add ansiuser

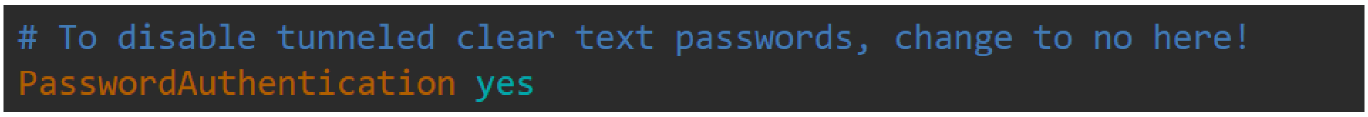
We will create ansiuser, the user which ansible will use.

```
sudo su - #Use root perms

adduser ansiuser
```

2) Modify /etc/ssh/sshd_config

In order to connect without needing password we will need to edit sshd daemon config files as well as the sudoers file.

```
sudo vim /etc/ssh/sshd_config
```



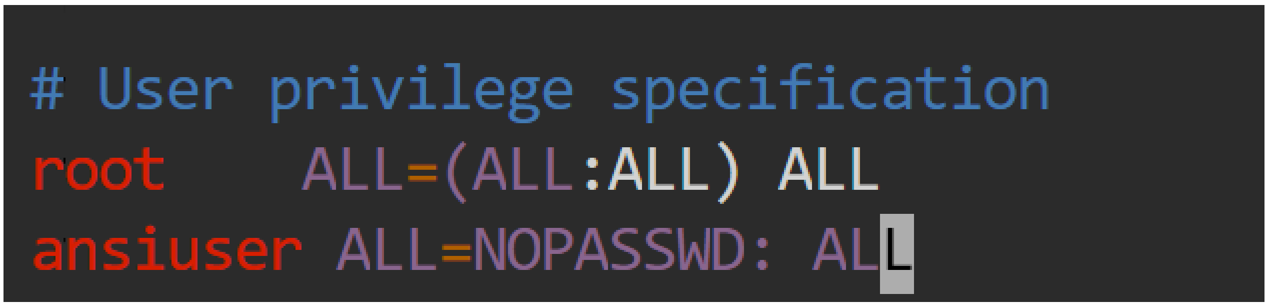
```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
```

Then we restart the service

```
sudo service sshd restart
```

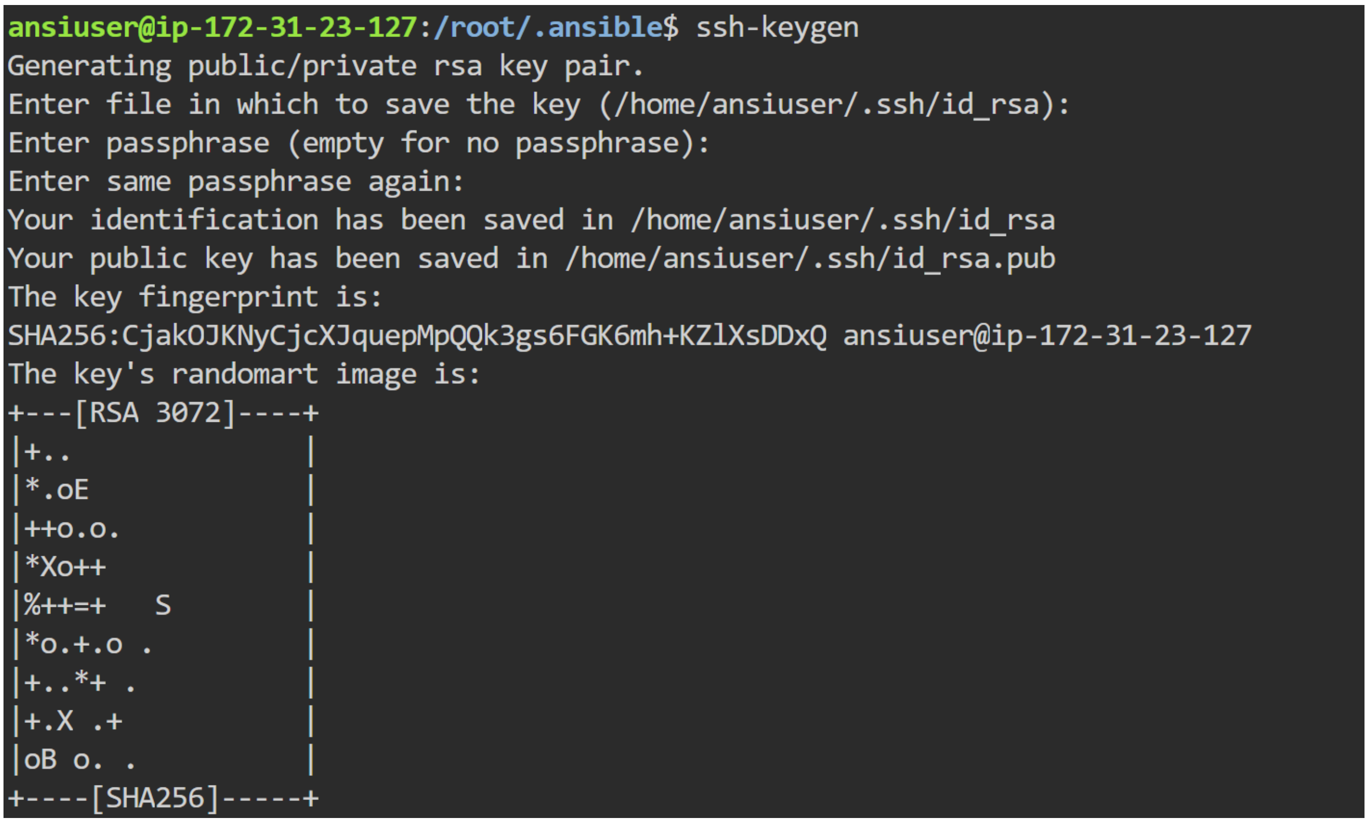
3) Finally add perms on the sudoers

```
sudo vim /etc/sudoers
```



```
# User privilege specification
root     ALL=(ALL:ALL) ALL
ansiuser ALL=NOPASSWD: ALL
```

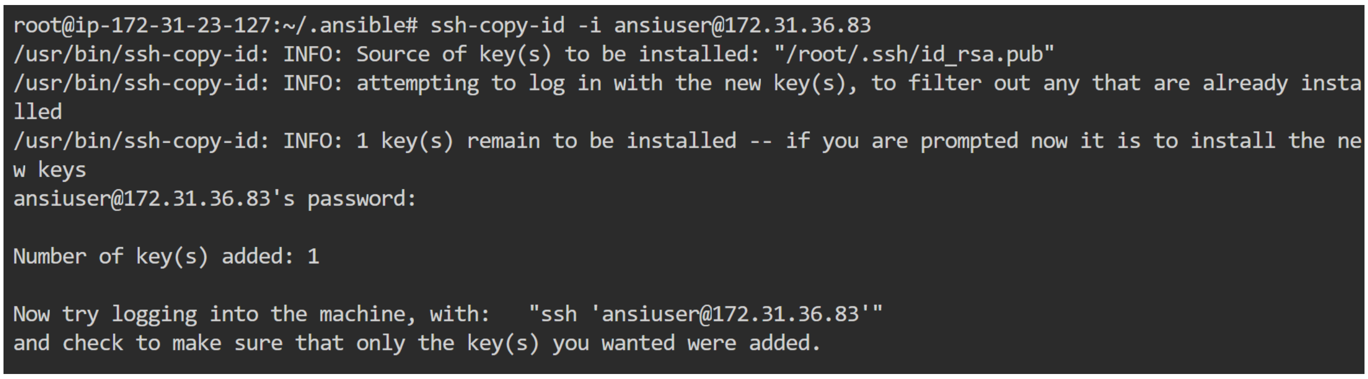
After adding it to the worker nodes we will create and copy a new ssh key from the AMC.



```
ansiuser@ip-172-31-23-127:/root/.ansible$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ansiuser/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansiuser/.ssh/id_rsa
Your public key has been saved in /home/ansiuser/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:CjakOJKNyCjcXJquepMpQQk3gs6FGK6mh+KZlXsDDxQ ansiuser@ip-172-31-23-127
The key's randomart image is:
+---[RSA 3072]----+
|+..              |
|*.oE             |
|++o.o.           |
|*Xo++            |
|%++=+   S        |
|*o.+.o .         |
|+..*+ .          |
|+.X .+           |
|oB o.  .         |
+----[SHA256]-----+
```

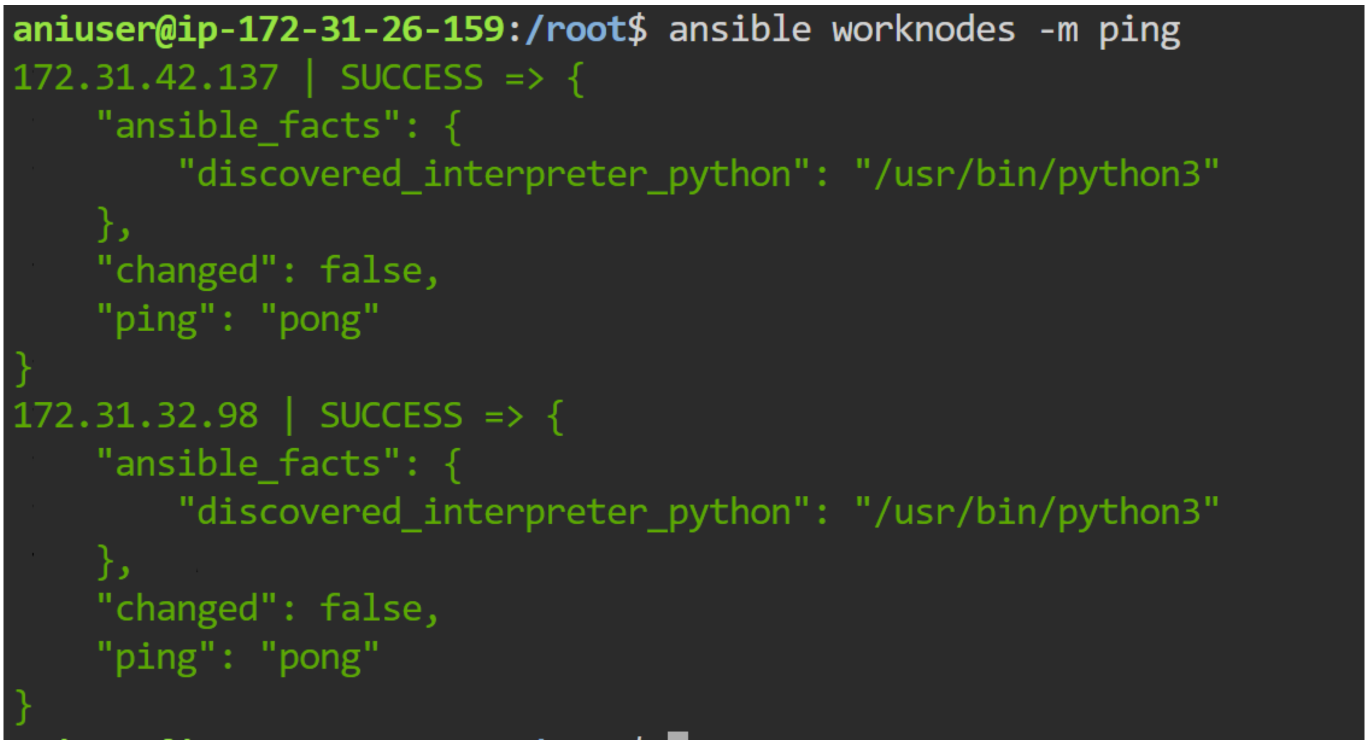
We then copy it to the worker nodes

```
ssh-copy-id -i ansiuser@hostip
```



```
root@ip-172-31-23-127:~/.ansible# ssh-copy-id -i ansiuser@172.31.36.83
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already insta
lled
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the ne
w keys
ansiuser@172.31.36.83's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'ansiuser@172.31.36.83'"
and check to make sure that only the key(s) you wanted were added.
```

Then we can finally check for the connection on our **worknodes**.



```
aniuser@ip-172-31-26-159:/root$ ansible worknodes -m ping
172.31.42.137 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
172.31.32.98 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

> We hard reseted lab. Ips may have changed.

PlayBook Creation

First we can create a PlayBook to install all needed dependencies and build our **War** file. We will need to install.

- Git
- Maven

We can create a simple playbook that checks for each dependency and it's version. Then, we store the output in a variable and install the dependency if needed.

Example of checking and installing a dependency:

```
  - name: Check if Git is installed
      command: git --version
      register: git_check
```

```
        ignore_errors: yes
   - name: Install Git
        package:
          name: git
          state: present
        when: git_check.rc != 0
```

Addiontonally, we need to add to our PlayBook.

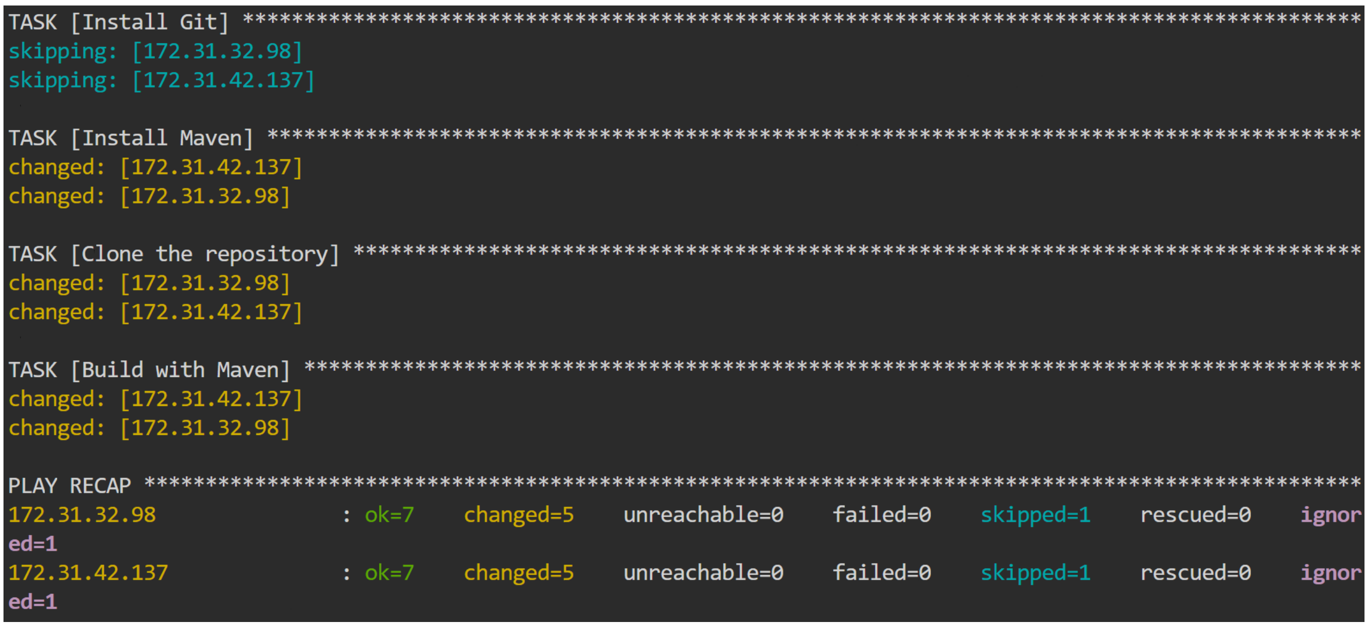
- Ability to clone the repository
- Build our War file with Maven

Our Java WebApp is part of ***Sonals repo***. You may find the full Java WebApp here.

```
---
- name: Install Dependencies
  hosts: worknodes
  become: true
  vars:
  tasks:
    - name: Update Repository
      command: sudo apt-update

    - name: Check if Git is installed
      command: git --version
      register: git_check
      ignore_errors: yes
    - name: Check if Maven is installed
      command: mvn --version
      register: maven_check
      ignore_errors: yes

    - name: Install Git
      package:
        name: git
        state: present
      when: git_check.rc != 0

    - name: Install Maven
      package:
        name: maven
        state: present
      when: maven_check.rc != 0

    - name: Clone the repository
      git: repo=https://github.com/Sonal0409/DevOpsCodeDemo.git dest=/tmp/code

    - name: Build with Maven
      command: chdir=/tmp/code mvn package
```

Before deploying it via Jenkins. We should try it.

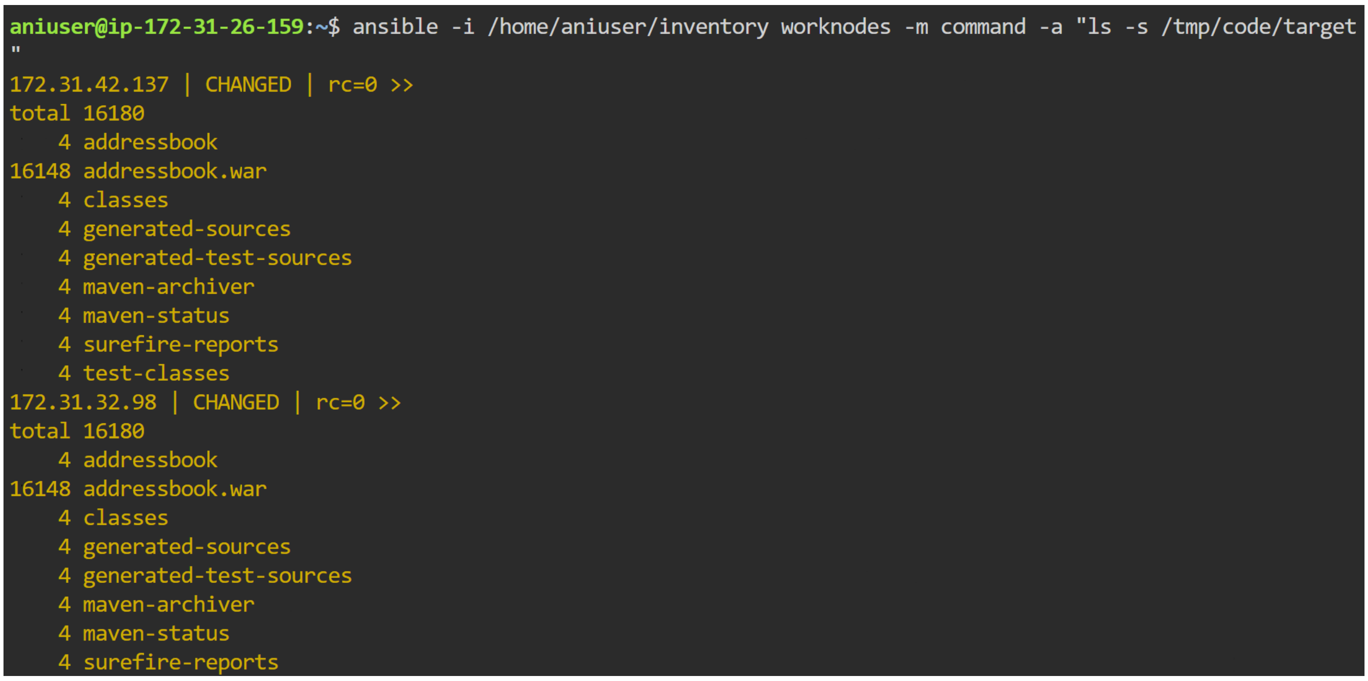
```
ansible-playbook -i /home/aniuser/inventory InstallationPlayBook.yml
```



```
TASK [Install Git] ***************************************************************************************
skipping: [172.31.32.98]
skipping: [172.31.42.137]

TASK [Install Maven] *************************************************************************************
changed: [172.31.42.137]
changed: [172.31.32.98]

TASK [Clone the repository] ******************************************************************************
changed: [172.31.32.98]
changed: [172.31.42.137]

TASK [Build with Maven] **********************************************************************************
changed: [172.31.42.137]
changed: [172.31.32.98]

PLAY RECAP ***********************************************************************************************
172.31.32.98               : ok=7    changed=5    unreachable=0    failed=0    skipped=1    rescued=0    ignor
ed=1
172.31.42.137              : ok=7    changed=5    unreachable=0    failed=0    skipped=1    rescued=0    ignor
ed=1
```

We can check as well if the files were created on the node.

```
ansible -i /home/aniuser/inventory worknodes -m command -a "ls -s
/tmp/code/target"
```



```
aniuser@ip-172-31-26-159:~$ ansible -i /home/aniuser/inventory worknodes -m command -a "ls -s /tmp/code/target
"
172.31.42.137 | CHANGED | rc=0 >>
total 16180
    4 addressbook
16148 addressbook.war
    4 classes
    4 generated-sources
    4 generated-test-sources
    4 maven-archiver
    4 maven-status
    4 surefire-reports
    4 test-classes
172.31.32.98 | CHANGED | rc=0 >>
total 16180
    4 addressbook
16148 addressbook.war
    4 classes
    4 generated-sources
    4 generated-test-sources
    4 maven-archiver
    4 maven-status
    4 surefire-reports
```

Since we have our CI part with a PlayBook. We can create another playbook to copy the files needed and building a docker image. After it, it will deploy it.

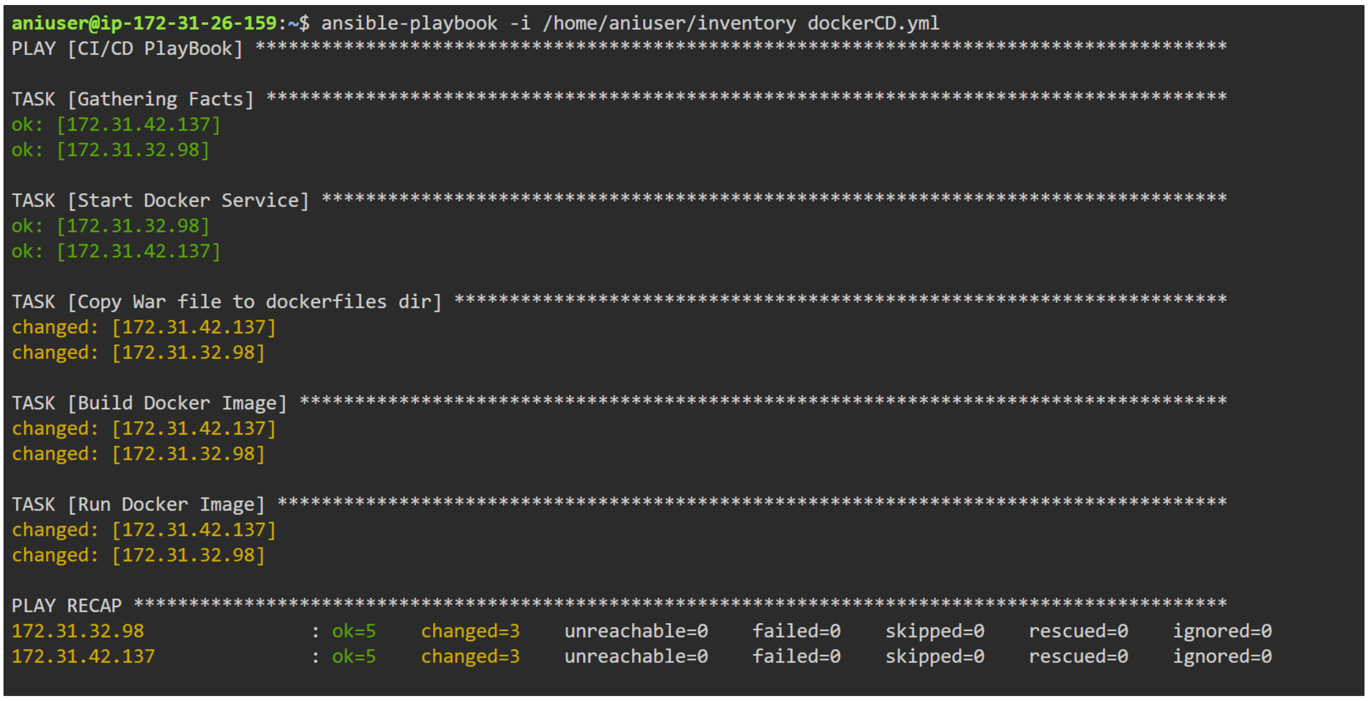
We will use the same dockerfile as last time.

```
FROM tomcat:9
ADD addressbook.war /usr/local/tomcat/webapps
CMD ["catalina.sh","run"]
EXPOSE 8080
```

The PlayBook looks like this.

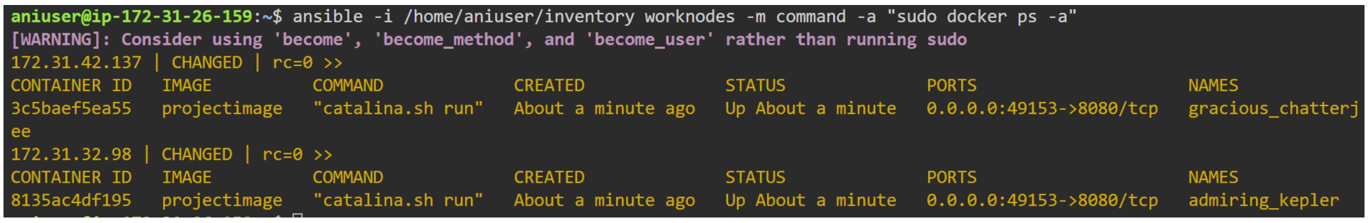
```
---
- name: CI/CD PlayBook
  hosts: worknodes
  vars:
  become: true
  tasks:
    - name: Start Docker Service
      service: name=docker state=started
    - name: Copy War file to dockerfiles dir
      copy: src=/tmp/code/target/addressbook.war dest=/tmp/code remote_src=yes
    - name: Build Docker Image
      command: chdir=/tmp/code docker build -t projectimage .
    - name: Run Docker Image
      command: docker run -d -P projectimage
```

Execution functions perfectly



```
aniuser@ip-172-31-26-159:~$ ansible-playbook -i /home/aniuser/inventory dockerCD.yml
PLAY [CI/CD PlayBook] ************************************************************************************

TASK [Gathering Facts] ***********************************************************************************
ok: [172.31.42.137]
ok: [172.31.32.98]

TASK [Start Docker Service] ******************************************************************************
ok: [172.31.32.98]
ok: [172.31.42.137]

TASK [Copy War file to dockerfiles dir] ******************************************************************
changed: [172.31.42.137]
changed: [172.31.32.98]

TASK [Build Docker Image] ********************************************************************************
changed: [172.31.42.137]
changed: [172.31.32.98]

TASK [Run Docker Image] **********************************************************************************
changed: [172.31.42.137]
changed: [172.31.32.98]

PLAY RECAP ***********************************************************************************************
172.31.32.98               : ok=5    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.42.137              : ok=5    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

When checking both containers and their runnign status we see they are working.



```
aniuser@ip-172-31-26-159:~$ ansible -i /home/aniuser/inventory worknodes -m command -a "sudo docker ps -a"
[WARNING]: Consider using 'become', 'become_method', and 'become_user' rather than running sudo
172.31.42.137 | CHANGED | rc=0 >>
CONTAINER ID   IMAGE          COMMAND             CREATED              STATUS              PORTS                     NAMES
3c5baef5ea55   projectimage   "catalina.sh run"   About a minute ago   Up About a minute   0.0.0.0:49153->8080/tcp   gracious_chatterj
ee
172.31.32.98 | CHANGED | rc=0 >>
CONTAINER ID   IMAGE          COMMAND             CREATED              STATUS              PORTS                     NAMES
8135ac4df195   projectimage   "catalina.sh run"   About a minute ago   Up About a minute   0.0.0.0:49153->8080/tcp   admiring_kepler
```

Jenkins PlayBook Setup

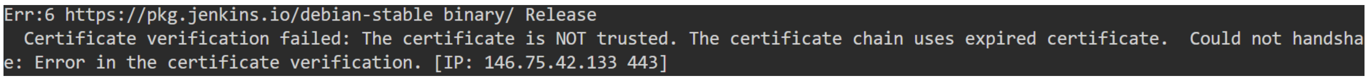
Firstly we check for Java and if it is install

```
java -version
```

Since we do not have it installed we can install it with

```
apt install default-jre
```

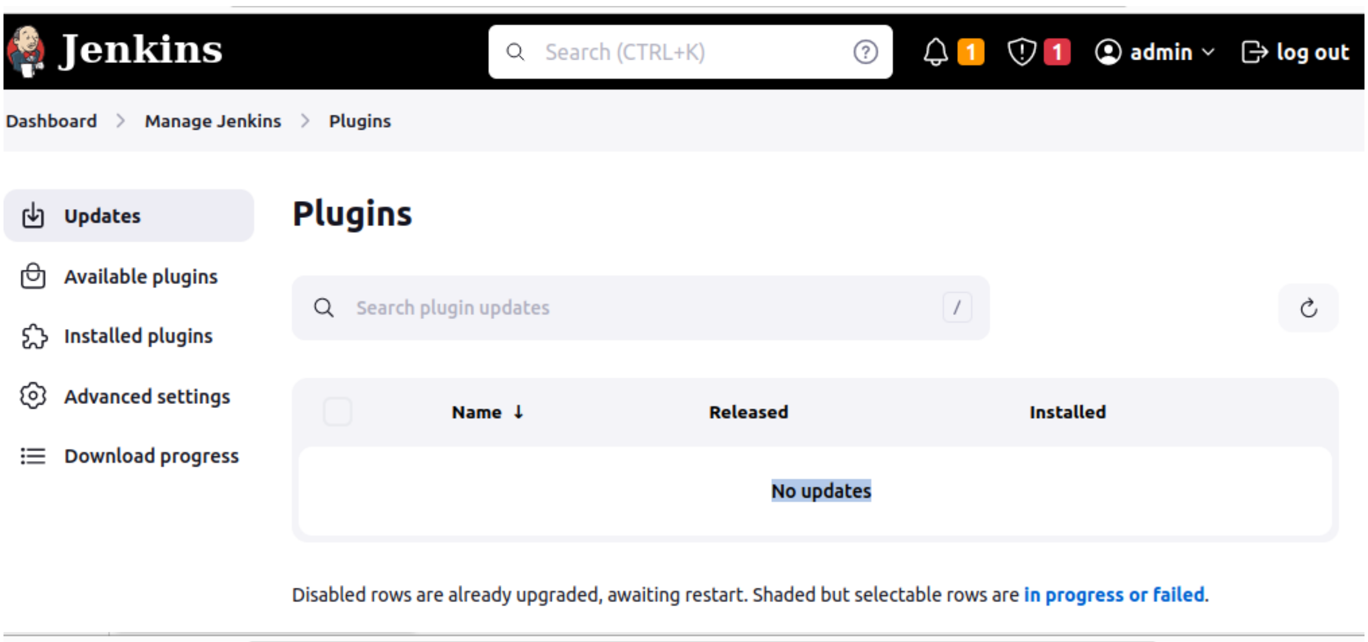
> Since we were having troubles installing jenkins. We decided to install another version on a virtual machine.



```
Err:6 https://pkg.jenkins.io/debian-stable binary/ Release
  Certificate verification failed: The certificate is NOT trusted. The certificate chain uses expired certificate.  Could not handsha
e: Error in the certificate verification. [IP: 146.75.42.133 443]
```

Without a handshake verification is not posible to install jenkins.

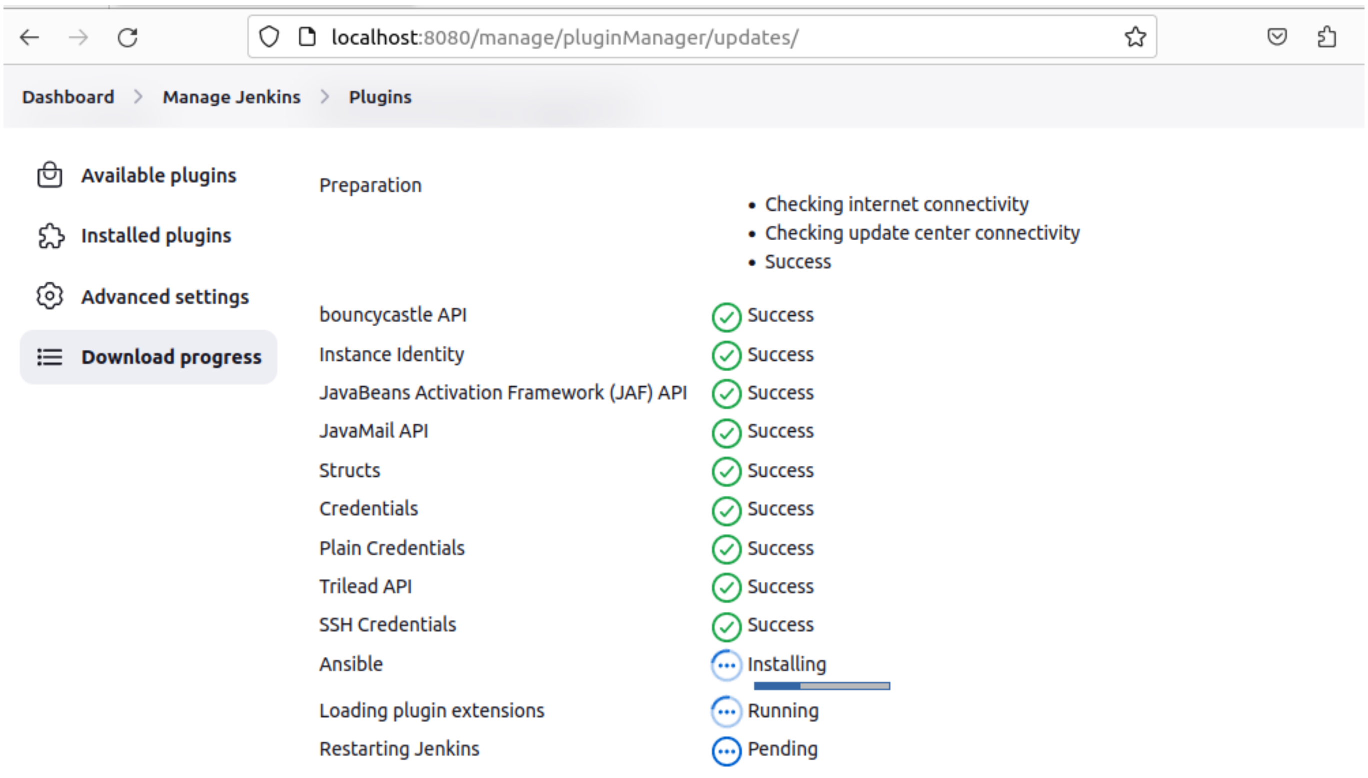
We can then continue with the virtual machine one. We will need to install the needed plugins.



Plugins

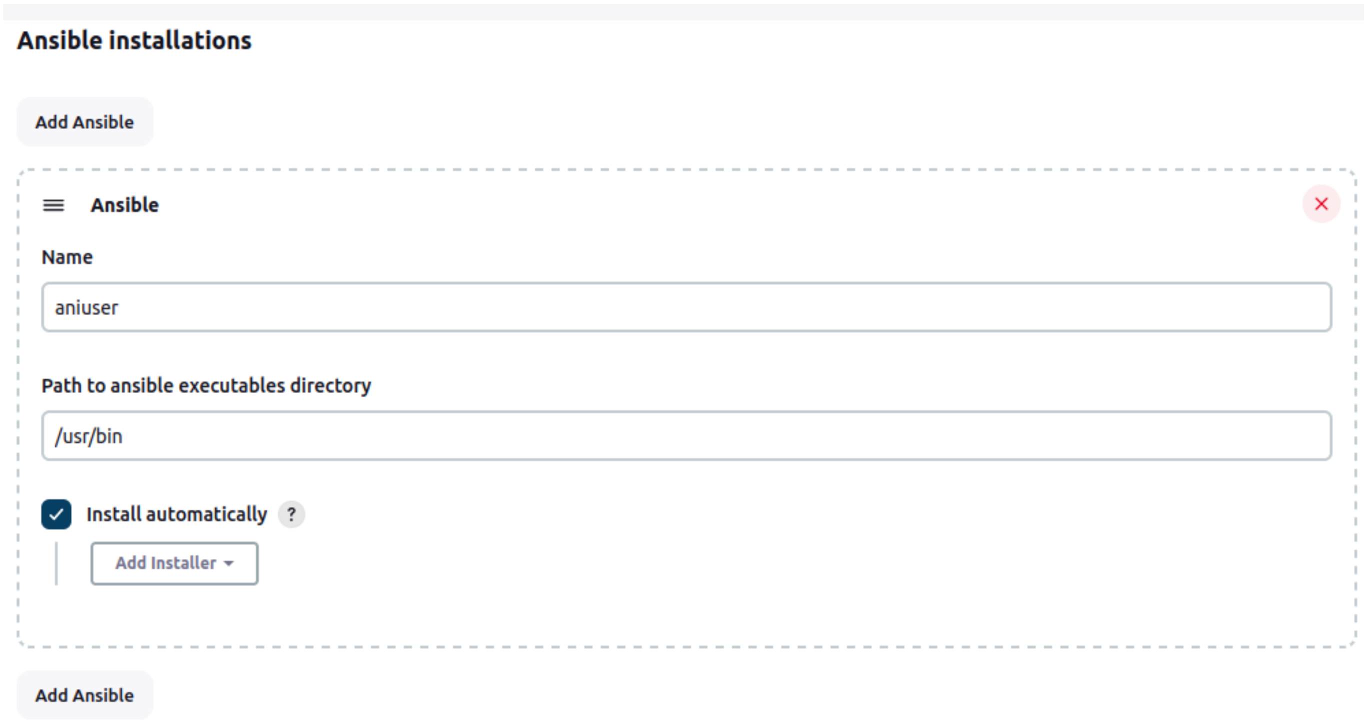
Name ↓	Released	Installed
No updates		

Disabled rows are already upgraded, awaiting restart. Shaded but selectable rows are in progress or failed.


localhost:8080/manage/pluginManager/updates/
Dashboard > Manage Jenkins > Plugins
Available plugins
Installed plugins
Advanced settings
Download progress

Preparation	• Checking internet connectivity • Checking update center connectivity • Success
bouncycastle API	✓ Success
Instance Identity	✓ Success
JavaBeans Activation Framework (JAF) API	✓ Success
JavaMail API	✓ Success
Structs	✓ Success
Credentials	✓ Success
Plain Credentials	✓ Success
Trilead API	✓ Success
SSH Credentials	✓ Success
Ansible	Installing
Loading plugin extensions	Running
Restarting Jenkins	Pending

Once installed we configure the tool.


Ansible installations

Add Ansible

Ansible

Name

aniuser

Path to ansible executables directory

/usr/bin

[✓] Install automatically ?

Add Installer ▾

Add Ansible

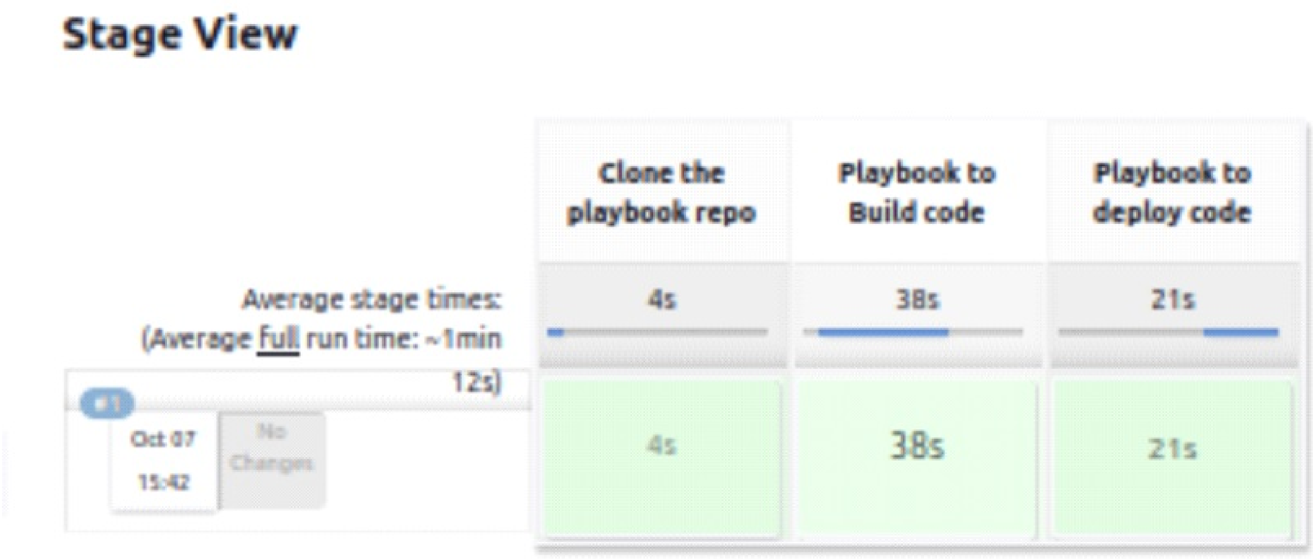
Create Ansible Job

Create a new Job

We create a pipeline job with then next syntax:

```
pipeline{

    agent any

    stages{

    stage('Clone the playbook repo')
    {
    steps{
        git branch: 'main', url: 'https://github.com/fpedrazav02/AnsibleCI-CD.git'
    }
    }
    stage('Playbook to Build code')
    {

    steps{
        ansiblePlaybook credentialsId: 'ansiblecredentials',
disableHostKeyChecking: true, installation: 'myansible', inventory: 'dev.inv',
playbook: 'InstallationPlayBook.yml'

    }

    }

    stage('Playbook to deploy code')
    {

    steps{
        ansiblePlaybook credentialsId: 'ansiblecredentials',
disableHostKeyChecking: true, installation: 'myansible', inventory: 'dev.inv',
playbook: 'dockerCD.yml'
    }
        }
    }
}
```

Finally, we can run the Job.



Stage View

	Clone the playbook repo	Playbook to Build code	Playbook to deploy code
Average stage times: (Average full run time: ~1min 12s)	4s	38s	21s
#1 Oct 07 15:42 No Changes	4s	38s	21s