

INSURANCE DOMAIN

Submission by : Pallavi
Submitted to : Vikul
Date of submission: 29-12-24

Life Insurance Project

The project involves creating a fully automated pipeline with Terraform, Jenkins, Docker, Ansible, Prometheus, and Grafana for monitoring.

- ✓ Git - For version control for tracking changes in the code files
- ✓ Jenkins - For continuous integration and continuous deployment
- ✓ Docker - For deploying containerized applications
- ✓ Ansible - Configuration management tools
- ✓ Selenium - For automating tests on the deployed web application
- ✓ AWS : For creating ec2 machines as servers and deploy the web application.

Insure Me project code is attached below.

<https://github.com/CSPPallavi/star-agile-insurance-project>

Step 1: Create an EC2 Instance

Step 1: Launch two Ubuntu EC2 Instance on AWS as ansible controller and ansible worker node.

1. Log in to AWS Management Console:

- o Access the AWS Management Console using below link and sign in with your credentials.
- o <https://aws.amazon.com/>

2. Go to the EC2 Dashboard:

From the Services menu, search for "EC2" and select it to open the EC2 Dashboard.

3. Start Instance Launch Process:

- o Click on the instances on left side and "Launch Instance" button

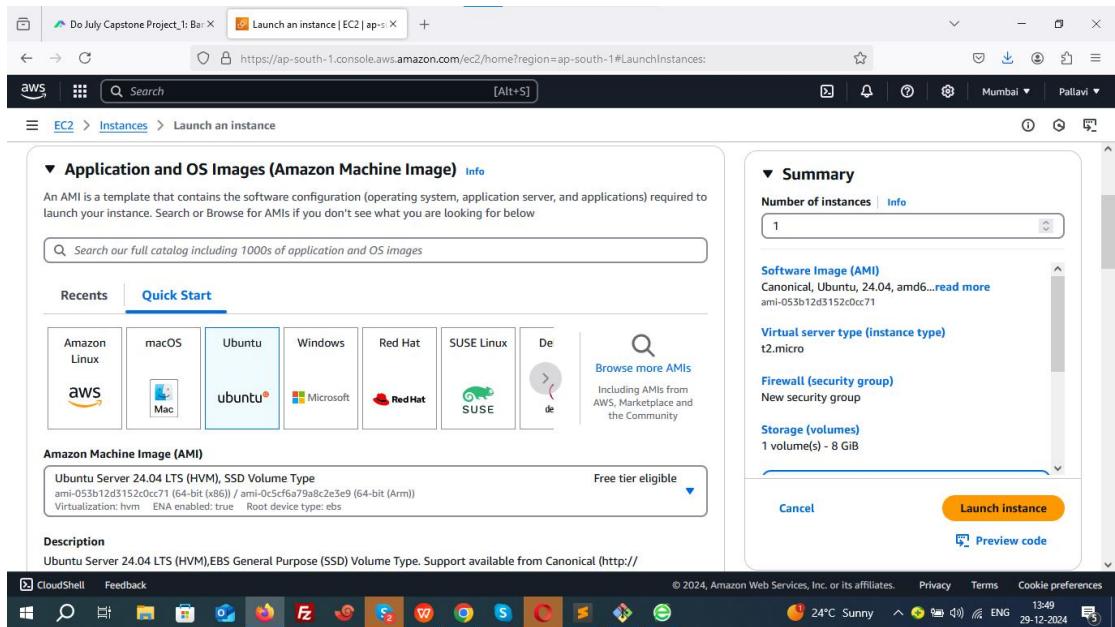
The screenshot shows the AWS EC2 Instances page. The left sidebar includes options like Dashboard, EC2 Global View, Events, Instances (with sub-options Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations), Images (AMIs, AMI Catalog), and Elastic Block Store (Volumes). The main content area displays 'Instances (2) Info' with a table showing instance details. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. Two instances are listed: 'Jenkins+Grafana+Prom...' (i-0b863563a66dcc5b5) and 'Finance_Project' (i-00678f747e2d88f5e), both marked as 'Running'. Below the table is a section titled 'Select an instance'.

o **Name Your Instance:** Enter a name for the instance.

Insure_me_controller

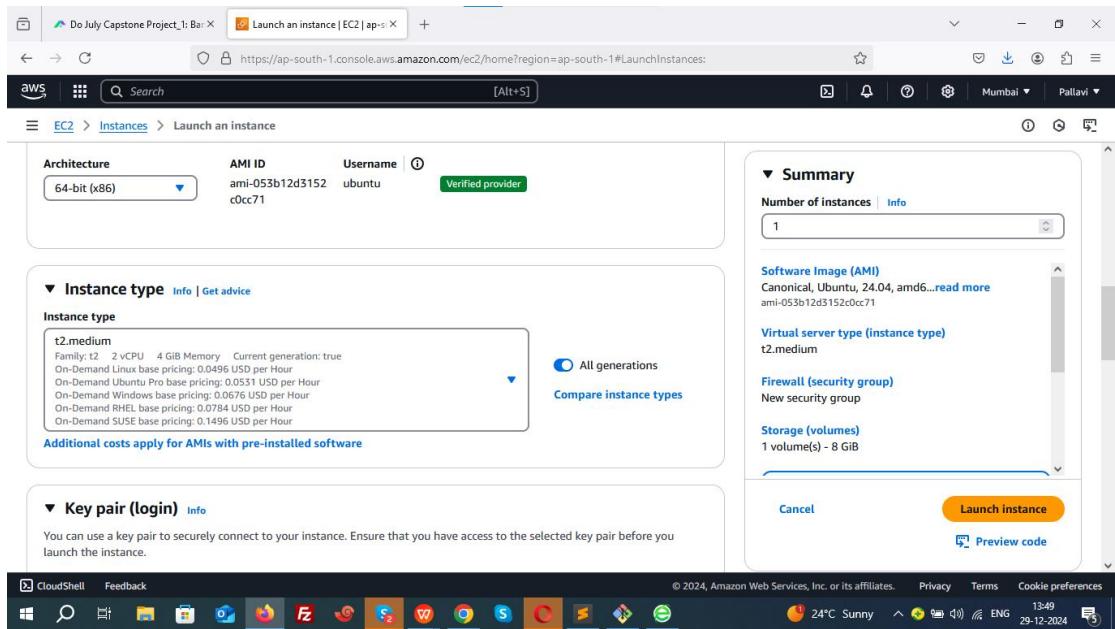
The screenshot shows the 'Launch an instance' wizard. The top navigation bar includes links for CloudShell, Feedback, and various browser tabs. The main content area is titled 'Launch an instance' with a sub-section 'Name and tags'. It shows a text input field with 'Insure_me_controller' and a button 'Add additional tags'. Below this is a section 'Application and OS Images (Amazon Machine Image)'. It features a search bar 'Search our full catalog including 1000s of application and OS images' and a 'Quick Start' tab. Underneath are buttons for 'Amazon Linux', 'macOS', 'Ubuntu' (which is selected), 'Windows', 'Red Hat', 'SUSE Linux', and 'Debian'. A 'Browse more AMIs' link is also present. To the right, a summary panel shows 'Number of instances' set to 1, 'Software Image (AMI)' as Canonical, Ubuntu, 24.04, amd64, 'Virtual server type (instance type)' as t2.micro, and 'Storage (volumes)' as 1 volume(s) - 8 GiB. At the bottom are 'Cancel', 'Launch instance' (in orange), and 'Preview code' buttons.

o **Choose an Amazon Machine Image (AMI):** Select an Ubuntu Server AMI



o Select Instance Type: Choose an appropriate instance type

t2.medium



o Key Pair: Choose an existing key pair or create a new one
 Click on create new key pair. Provide key pair name, select key pair type as RSA and private key format as .pem and click on create key pair.

The screenshot shows the 'Launch an instance' wizard on the AWS EC2 console. The 'Network settings' section is open, showing a VPC (vpc-0e78ba0969c51beb6) and a subnet (No preference). The 'Auto-assign public IP' option is set to 'Enable'. Other visible fields include 'Software Image (AMI)', 'Virtual server type (instance type)', 'Firewall (security group)', and 'Storage (volumes)'. A summary panel on the right indicates 1 instance will be launched. Buttons for 'Launch instance' and 'Preview code' are at the bottom.

o Configure Network Settings:

Click on edit.

Use the default VPC settings.
Choose any one subnet

Ensure "Auto-assign Public IP" is enabled.

The screenshot shows the 'Launch an instance' wizard on the AWS EC2 console. The 'Network settings' section is open, showing a VPC (vpc-0e78ba0969c51beb6) and a subnet (No preference). The 'Auto-assign public IP' option is set to 'Enable'. In the 'Firewall (security groups)' section, the 'Select existing security group' button is selected. Other visible fields include 'Software Image (AMI)', 'Virtual server type (instance type)', and 'Storage (volumes)'. A summary panel on the right indicates 1 instance will be launched. Buttons for 'Launch instance' and 'Preview code' are at the bottom.

o Create a New Security Group:



Option Selection: Select the option "Create security group."

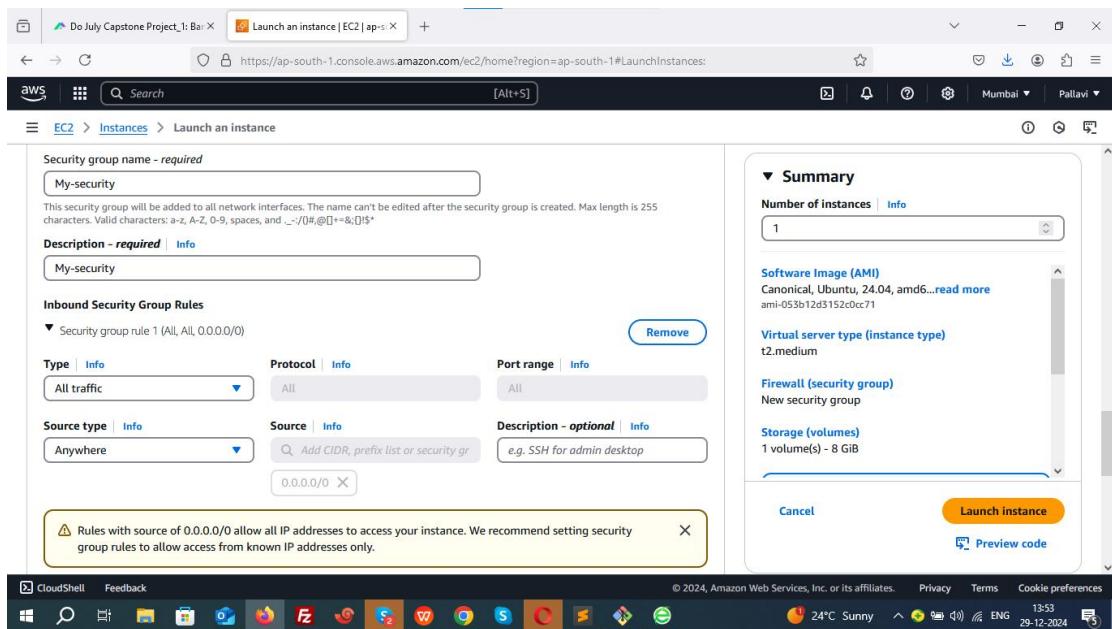
Security Group Name: Enter a name for your security group in the "Security group name" field

Description: Provide a brief description of the security group

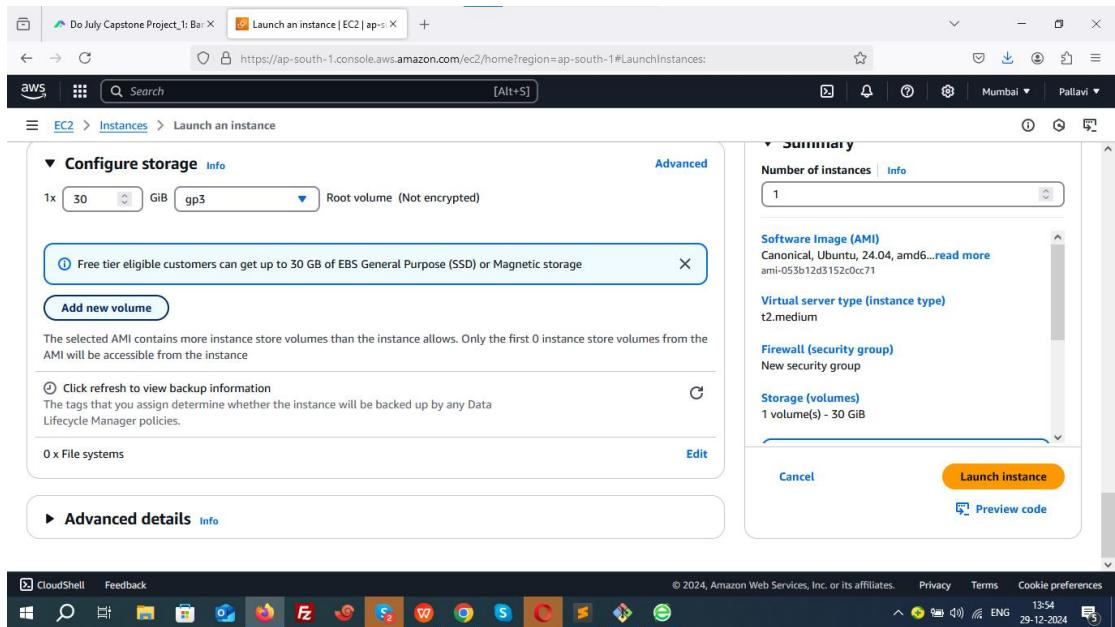
Configure Inbound Security Group Rules:

Rule 1

- o **Type:** Select All traffic from the drop down menu.
- o **Source Type:** Choose Anywhere to allow SSH access from any IP address.



- o **Set Storage Options:** Adjust the storage size if necessary.

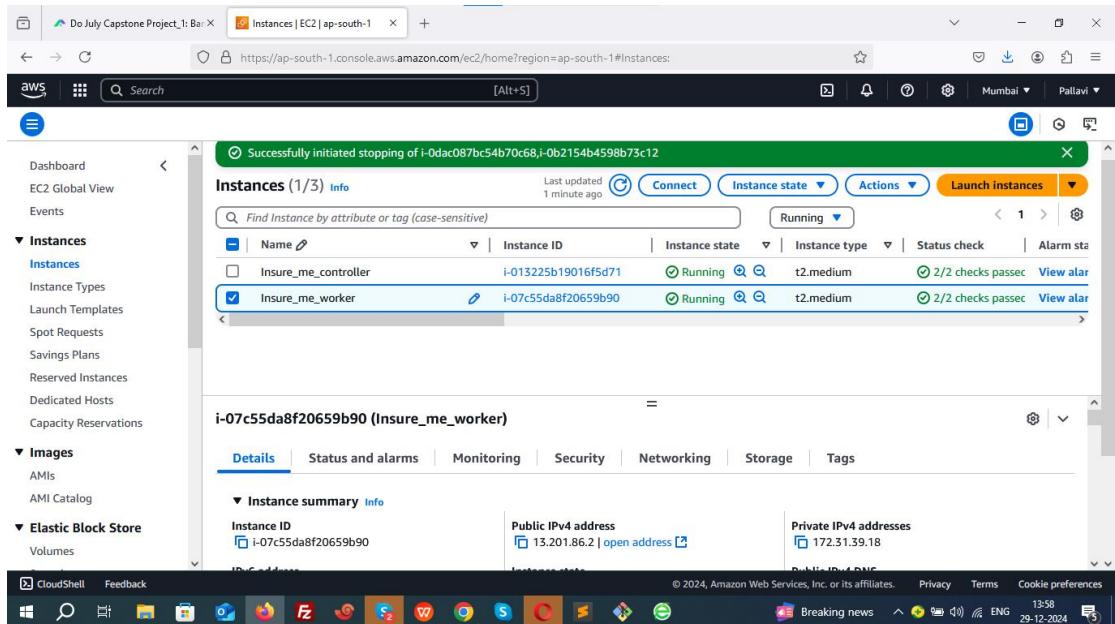


Review and Launch: Click "Launch Instance."

o Wait for Instance Initialization:

Monitor the instance until it's in a "Running" state and passes the status checks.

Similary launch another sever called Insure_me_worker

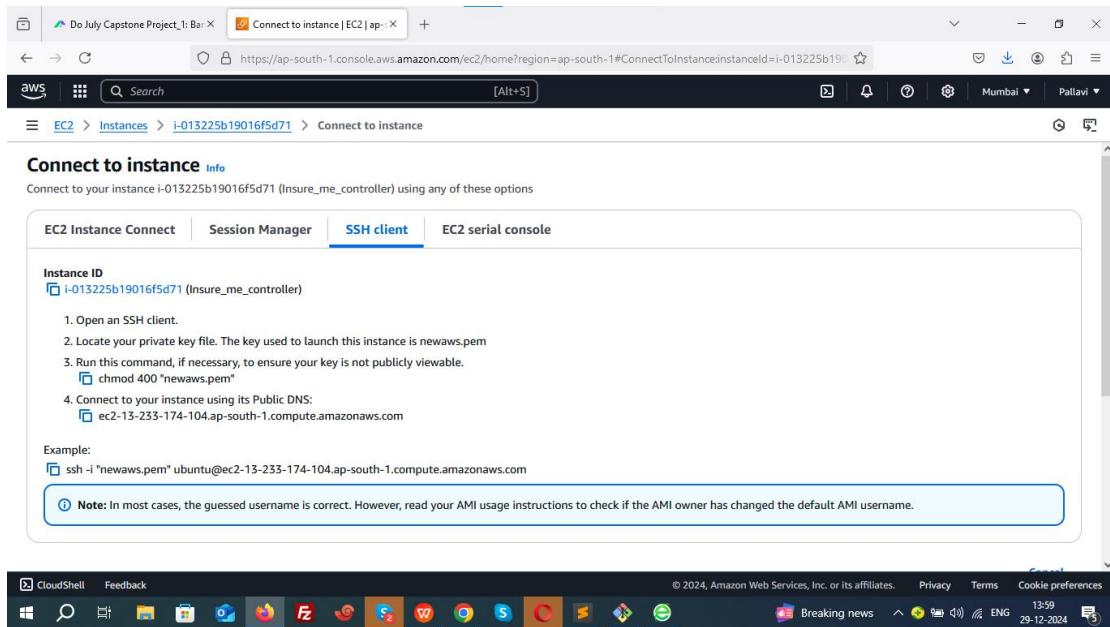


o Connect to Your controller using git bash or any other.

Select the instance and click on connect

Select the tab SSH client and copy ssh command.

```
ssh -i "insure-me.pem" ubuntu@ec2-3-110-193-218.ap-south-1.compute.amazonaws.com
```



Open git bash and go to the location where your pem key is stored.

Paste the command

```
ubuntu@ip-172-31-37-208: ~
12754@NITL-009506 MINGW64 ~/Downloads
$ ssh -i "insure-me.pem" ubuntu@ec2-3-110-193-218.ap-south-1.compute.amazonaws.com
welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1018-aws x86_64)

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

System information as of Fri Dec 6 04:29:22 UTC 2024
System load: 0.0          Processes:           123
Usage of /: 18.9% of 28.02GB Users logged in: 0
Memory usage: 38%          IPV4 address for enx0: 172.31.37.208
Swap usage: 0%

Expanded security Maintenance for Applications is not enabled.

44 updates can be applied immediately.
29 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Fri Dec 6 02:40:48 2024 from 125.63.79.174
ubuntu@ip-172-31-37-208:~$ |
```

Step 2: Configure Ansible on the controller EC2 Instance

1. Connect to the Newly Created controller Instance:

Use SSH with the generated private key.

2. Ansible Installation

--> Install required dependencies

[sudo apt-get install -y software-properties-common](#)

--> Add the Ansible repository

[sudo apt-add-repository ppa:ansible/ansible](#)

--> Press [ENTER] when prompted to continue

--> Update the package index

[sudo apt-get update](#)

--> Install Ansible

[sudo apt-get install -y ansible](#)

--> Verify installation

[ansible --version](#)

```

root@ip-172-31-42-32:/home/ubuntu
Preparing to unpack .../10-sshpss_1.09-1_amd64.deb ...
Unpacking sshpass (1.09-1) ...
Setting up python3-ntlm-auth (1.5.0-1) ...
Setting up python3-resolvelib (1.0.1-1)
Setting up python3-kerberos (1.1.14-3.1build9) ...
Setting up ansible-core (2.17.7-1ppa-noble) ...
Setting up sshpass (1.09-1) ...
Setting up python3-xmmitodict (0.13.0-1) ...
Setting up ansible (10.7.0-1ppa-noble) ...
Setting up python3-nacl (1.5.0-4build1)
Setting up python3-requests_ntlm (1.1.0-3) ...
Setting up python3-winrm (0.4.3-2) ...
Setting up python3-paramiko (2.12.0-2ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-42-32:/home/ubuntu# ansible --version
ansible [core 2.17.7]
  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
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Sep 11 2024, 14:17:37) [GCC 13.2.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
root@ip-172-31-42-32:/home/ubuntu#

```

2. Become a root user and create a New User (ansiuser)

--> Add a new user named ansiuser
 sudo adduser ansiuser

```

ansiuser@ip-172-31-91-124: ~
ansible python module location = /usr/lib/python3/dist-packages/ansible
ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
executable location = /usr/bin/ansible
python version = 3.12.3 (main, Sep 11 2024, 14:17:37) [GCC 13.2.0] (/usr/bin/python3)
jinja version = 3.1.2
libyaml = True
root@ip-172-31-84-10:/home/ubuntu# adduser ansiuser
info: Adding user 'ansiuser' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group 'ansiuser' (1001) ...
info: Adding new user 'ansiuser' (1001) with group 'ansiuser (1001)' ...
info: Creating home directory '/home/ansiuser' ...
info: Copying files from '/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ansiuser
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n]
info: Adding new user 'ansiuser' to supplemental / extra groups 'users' ...
info: Adding user 'ansiuser' to group 'users' ...
root@ip-172-31-84-10:/home/ubuntu#
root@ip-172-31-84-10:/home/ubuntu# vim /etc/sudoers
root@ip-172-31-84-10:/home/ubuntu# su - ansiuser
ansiuser@ip-172-31-84-10:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ansiuser/.ssh/id_ed25519):
Created directory '/home/ansiuser/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:

```

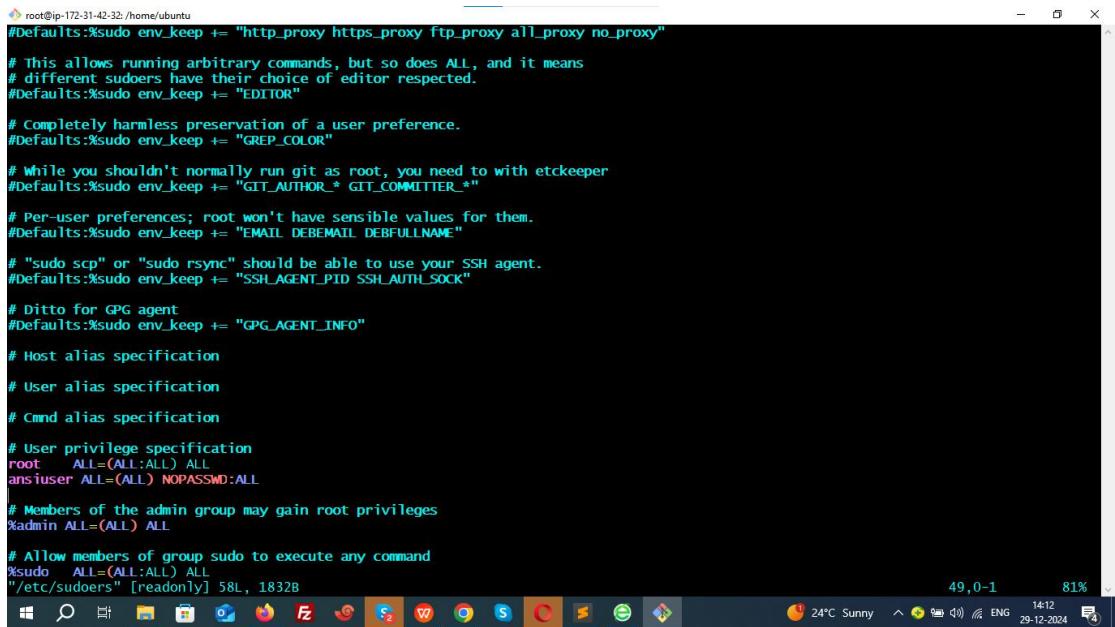
3. Grant Root Privileges to ansiuser

--> Edit the sudoers file
 sudo vim /etc/sudoers

Add the following line at the end of the file:

plaintext

ansiuser ALL=(ALL) NOPASSWD:ALL



```
# Defaults: %sudo env_keep += "http_proxy https_proxy ftp_proxy all_proxy no_proxy"
# This allows running arbitrary commands, but so does ALL, and it means
# different sudoers have their choice of editor respected.
#Defaults: %sudo env_keep += "EDITOR"

# Completely harmless preservation of a user preference.
#Defaults: %sudo env_keep += "GREP_COLOR"

# While you shouldn't normally run git as root, you need to with etckeeper
#Defaults: %sudo env_keep += "GIT_AUTHOR_* GIT_COMMITTER_"

# Per-user preferences; root won't have sensible values for them.
#Defaults: %sudo env_keep += "EMAIL DEBEMAIL DEBFULLNAME"

# "sudo scp" or "sudo rsync" should be able to use your SSH agent.
#Defaults: %sudo env_keep += "SSH_AGENT_PID SSH_AUTH_SOCK"

# Ditto for GPG agent
#Defaults: %sudo env_keep += "GPG_AGENT_INFO"

# Host alias specification
# User alias specification
# Cmd alias specification

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

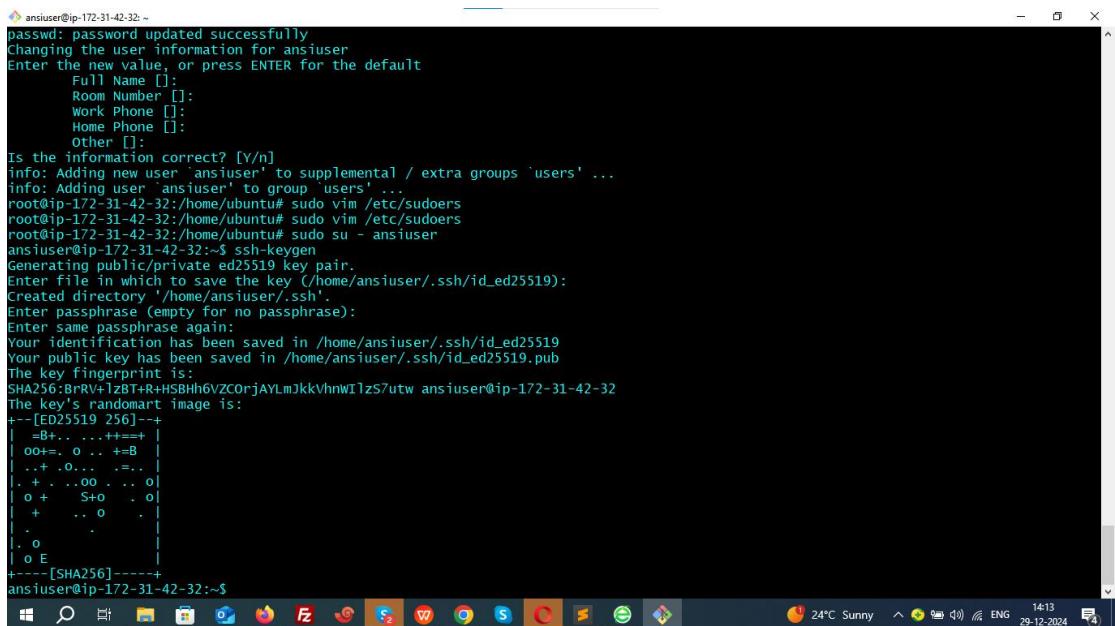
# Members of the admin group may gain root privileges
%admin  ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
"/etc/sudoers" [readonly] 58L, 1832B
```

4. Switch to ansiuser and Generate SSH Key

--> Switch to the new user
sudo su - ansiuser

--> Generate an SSH key pair
ssh-keygen--> Press Enter for all prompts to create the key with default settings

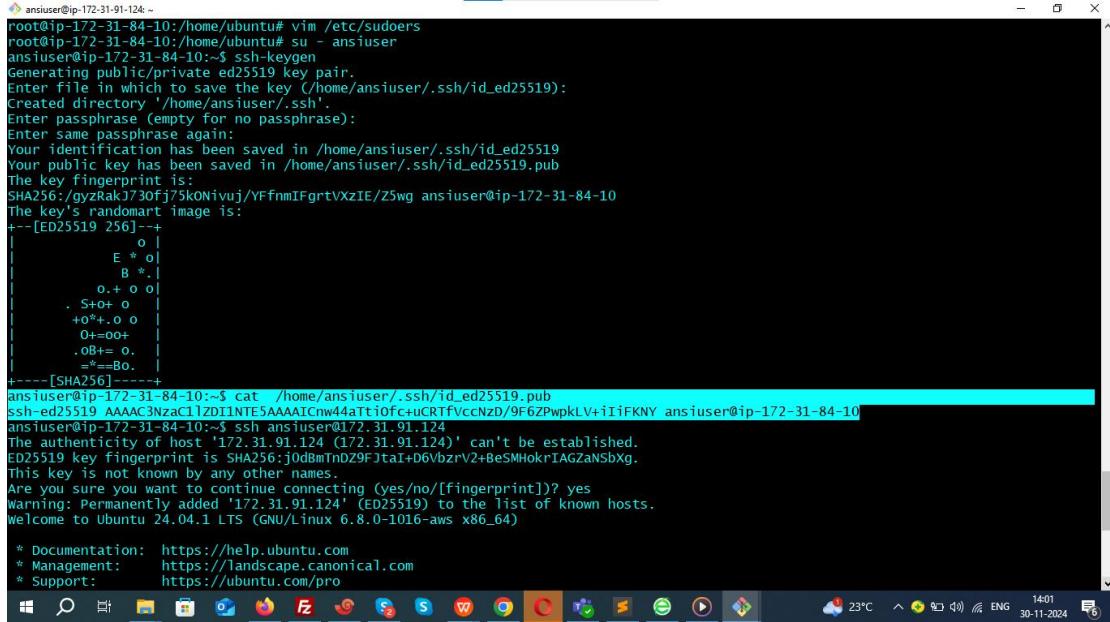


```
ansiuuser@ip-172-31-42-32: ~
passwd: password updated successfully
Changing the user information for ansiuser
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n]
info: Adding new user 'ansiuser' to supplemental / extra groups 'users' ...
info: Adding user 'ansiuser' to group 'users' ...
root@ip-172-31-42-32:/home/ubuntu# sudo vim /etc/sudoers
root@ip-172-31-42-32:/home/ubuntu# sudo vim /etc/sudoers
root@ip-172-31-42-32:/home/ubuntu# sudo su - ansiuser
ansiuser@ip-172-31-42-32:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ansiuser/.ssh/id_ed25519):
Created directory '/home/ansiuser/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansiuser/.ssh/id_ed25519
Your public key has been saved in /home/ansiuser/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:BRv1zBTR+HSBHN6VZCOrjAYLmJkkVhnW1zS7utw ansiuser@ip-172-31-42-32
The key's randomart image is:
++[ED25519 256]++
| =B+... . .+==+
| oo+=. o ... +B |
| .+ .o . . . . |
| . + .oo . . o |
| o + . S+o . o |
| + . . o . |
| . . . |
| o E |
+---[SHA256]---+
ansiuser@ip-172-31-42-32:~$
```

--> Display the public key

```
cat /home/ansiuser/.ssh/id_ed25519.pub
```

Copy the output of the public key for use on the managed node.



The screenshot shows a terminal window on a Windows desktop. The command `ssh-keygen` is being run to generate an SSH key pair. The terminal output includes:

```
ansiuuser@ip-172-31-91-124: ~
root@ip-172-31-84-10:/home/ubuntu# vim /etc/sudoers
root@ip-172-31-84-10:/home/ubuntu# su - ansiuser
ansiuser@ip-172-31-84-10:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ansiuser/.ssh/id_ed25519):
Created directory '/home/ansiuser/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansiuser/.ssh/id_ed25519
Your public key has been saved in /home/ansiuser/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:/gyzRakj730fj75kOnivuj/YFfnmIFgryVxIE/Z5wg ansiuser@ip-172-31-84-10
The key's randomart image is:
+--[ED25519 256]--+
|          o
|         E * o
|         B *
|         o . o
|        o.+ o o
|       . S+o+ o
|      +o*+o o o
|      O++o+
|     .O8+= o.
|    =*=B0.
+---[SHA256]---+
ansiuser@ip-172-31-84-10:~$ cat /home/ansiuser/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1ZD1lNTE5AAAAICnw44aTti0fc+uCRTFVccNzD/9F6ZPwpkLV+jIiFKNY ansiuser@ip-172-31-84-10
ansiuser@ip-172-31-84-10:~$ ssh ansiuser@172.31.91.124
The authenticity of host '172.31.91.124' (172.31.91.124) can't be established.
ED25519 key fingerprint is SHA256:j0d8mTnD29FJtAI+D6vbzrV2+fSMHokrIAGZaNSbXg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.31.91.124' (ED25519) to the list of known hosts.
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

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

 23°C ^ ⚡ ⚡ ENG 14:01 30-11-2024
```

On the worker node

1. Create the User (ansiuser)

--> Add a new user named ansiuser
sudo adduser ansiuser

2. Grant Root Privileges to ansiuser

--> Edit the sudoers file
sudo vim /etc/sudoers

Add the following line at the end of the file:

plaintext

ansiuser ALL=(ALL) NOPASSWD:ALL

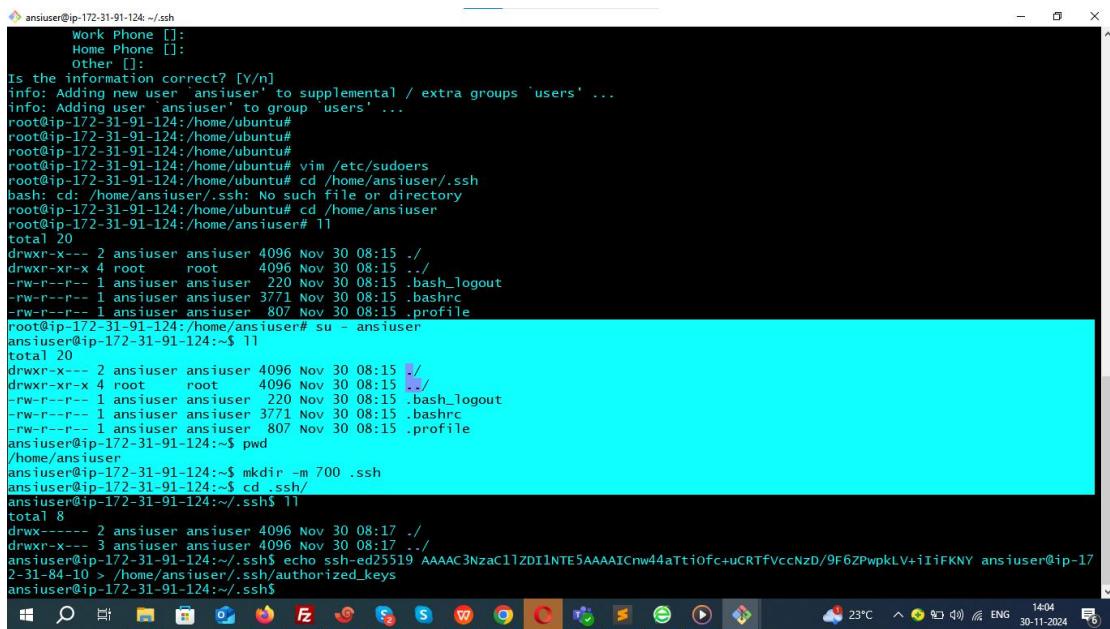
3. Create .ssh Directory and Authorize the Public Key

--> Switch to the new user

```
sudo su - ansiuser
```

--> Create the .ssh directory with appropriate permissions

```
mkdir -m 700 /home/ansiuser/.ssh
```



The screenshot shows a Windows terminal window with a black background and white text. It displays the command history and output of the user's actions:

```
ansiuser@ip-172-31-91-124:~/.ssh
work Phone []
Home Phone []
Other []

Is the information correct? [Y/n]
info: Adding new user 'ansiuser' to supplemental / extra groups 'users' ...
info: Adding user 'ansiuser' to group 'users' ...
root@ip-172-31-91-124:/home/ubuntu#
root@ip-172-31-91-124:/home/ubuntu#
root@ip-172-31-91-124:/home/ubuntu#
root@ip-172-31-91-124:/home/ubuntu# vim /etc/sudoers
root@ip-172-31-91-124:/home/ubuntu# cd /home/ansiuser/.ssh
bash: cd: /home/ansiuser/.ssh: No such file or directory
root@ip-172-31-91-124:/home/ubuntu# cd /home/ansiuser
root@ip-172-31-91-124:/home/ansiuser# ll
total 20
drwxr-x--- 2 ansiuser ansiuser 4096 Nov 30 08:15 ./
drwxr-xr-x 4 root    root   4096 Nov 30 08:15 ../
-rw-r--r-- 1 ansiuser ansiuser 220 Nov 30 08:15 .bash_logout
-rw-r--r-- 1 ansiuser ansiuser 3771 Nov 30 08:15 .bashrc
-rw-r--r-- 1 ansiuser ansiuser 807 Nov 30 08:15 .profile
root@ip-172-31-91-124:/home/ansiuser# su - ansiuser
ansiuser@ip-172-31-91-124:~$ ll
total 20
drwxr-x--- 2 ansiuser ansiuser 4096 Nov 30 08:15 ./
drwxr-xr-x 4 root    root   4096 Nov 30 08:15 ../
-rw-r--r-- 1 ansiuser ansiuser 220 Nov 30 08:15 .bash_logout
-rw-r--r-- 1 ansiuser ansiuser 3771 Nov 30 08:15 .bashrc
-rw-r--r-- 1 ansiuser ansiuser 807 Nov 30 08:15 .profile
ansiuser@ip-172-31-91-124:~$ pwd
/home/ansiuser
ansiuser@ip-172-31-91-124:~$ mkdir -m 700 .ssh
ansiuser@ip-172-31-91-124:~$ cd .ssh/
ansiuser@ip-172-31-91-124:~/ssh$ ll
total 8
drwx----- 2 ansiuser ansiuser 4096 Nov 30 08:17 ./
drwxr-x--- 3 ansiuser ansiuser 4096 Nov 30 08:17 ../
ansiuser@ip-172-31-91-124:~/ssh$ echo ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAICnw44aTtiOfc+uCRTfVccNzD/9F6ZPwpkLV+iIiFKNY ansiuser@ip-172-31-84-10 > /home/ansiuser/.ssh/authorized_keys
ansiuser@ip-172-31-91-124:~/ssh$
```

--> Create the authorized_keys file and paste the public key

```
echo "<public_key>" > /home/<username>/ssh/authorized_keys
```

```
echo "ssh-ed25519
```

```
AAAAC3NzaC1lZDI1NTE5AAAAICnw44aTtiOfc+uCRTfVccNzD/9F6Z
```

```
PwpkLV+iIiFKNY ansiuser@ip-172-31-84-10" >
```

```
/home/ansiuser/.ssh/authorized_keys
```

--> Set proper permissions for the authorized_keys file

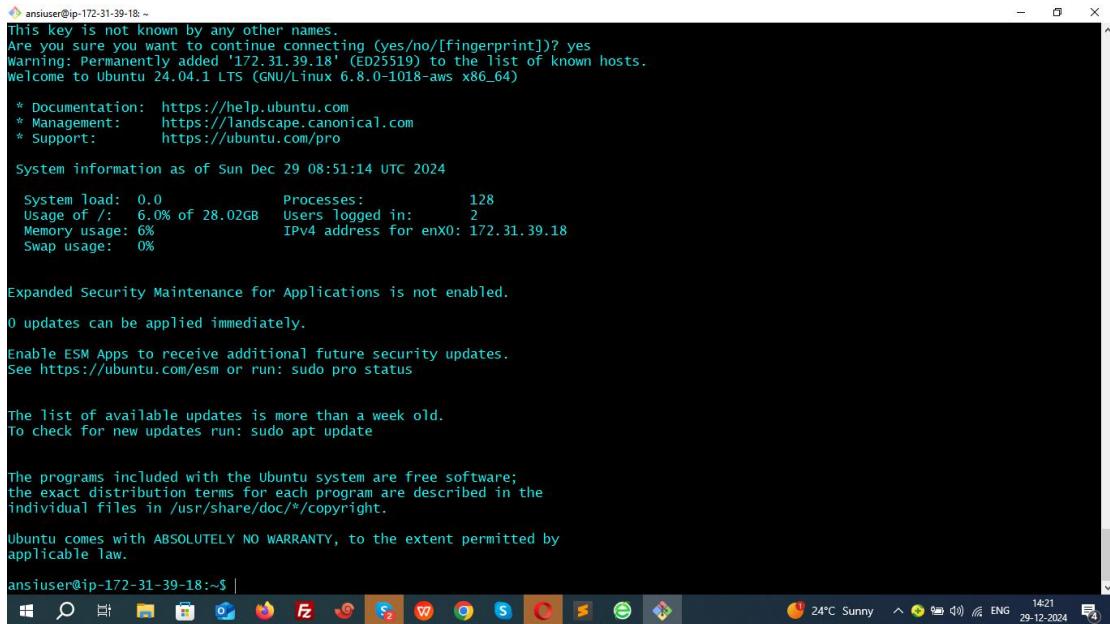
```
chmod 600 /home/ansiuser/.ssh/authorized_keys
```

Verify SSH Access

From the **controller node**, test SSH access to the managed node as ansiuser:

```
ssh ansiuser@<host_private_IP>
```

If configured correctly, you will be able to log in without a password.



```
ansiu...@ip-172-31-39-18: ~
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.31.39.18' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1018-aws x86_64)

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

System information as of Sun Dec 29 08:51:14 UTC 2024

System load: 0.0      Processes:          128
Usage of /: 6.0% of 28.02GB   Users logged in:    2
Memory usage: 6%
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

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.

ansiuser@ip-172-31-39-18:~$ |
```

Additional Steps for Ansible Inventory and Ping Test

By default the inventory file for ansible will be available at /etc/ansible/hosts directory

Edit this file as below.

```
# vim hosts
```

Press i to insert below data

```
[webserver]
```

```
172.31.39.18
```

Save the file.

Test Ansible Connection:

Execute below Ad Hoc command

```
ansible -i <inventory_file_name> <target_host_group_name> -m  
<module_name>  
ansible -i my_inventory webserver -m ping
```

```
ansiuser@ip-172-31-42-32: ~
ansiuser@ip-172-31-42-32:/etc/ansible$ ansible -i hosts webserver -m ping
[WARNING]: Platform linux on host 172.31.39.18 is using the discovered Python interpreter at
/usr/bin/python3.12, but future installation of another Python interpreter could change the meaning of
that path. See https://docs.ansible.com/ansible-
core/2.17/reference_appendices/interpreter_discovery.html for more information.
172.31.39.18 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.12"
    },
    "changed": false,
    "ping": "pong"
}
ansiuser@ip-172-31-42-32:/etc/ansible$ sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
--2024-12-29 09:05:10-- https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 151.101.154.133, 2a04:4e42:24::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|151.101.154.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3175 (3.1k) [application/pgp-keys]
Saving to: '/usr/share/keyrings/jenkins-keyring.asc'

/usr/share/keyrings/jenki 100%[=====] 3.10K --.-KB/s   in 0s
2024-12-29 09:05:10 (42.5 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' saved [3175/3175]

ansiuser@ip-172-31-42-32:/etc/ansible$ echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

- SUCCESS: Indicates the host was reachable.
 - ping: "pong": Confirms successful communication.

Install Java , Jenkins, git and docker Using Ansible Playbook:

Add Jenkins key and repo manually.

ADD GPG KEY:

```
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
```

ADD Jenkins repository:

```
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
\https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

Create an Ansible playbook:

```
vim insurance_me.yml
```

Add the following:

```
- name: Install and set up devops tools
  hosts: localhost
  become: true
  tasks:
    - name: Update the apt repo
      command: apt-get update
    - name: Install multiple packages
      package: name={{item}} state=present
      loop:
        - git
        - docker.io
        - openjdk-17-jdk
    - name: update apt-repo
      command: sudo apt-get update
```

```

- name: install jenkins
  command: sudo apt-get install jenkins -y
- name: start Jenkins and docker service
  service: name={{item}} state=started
loop:
- jenkins
- docker

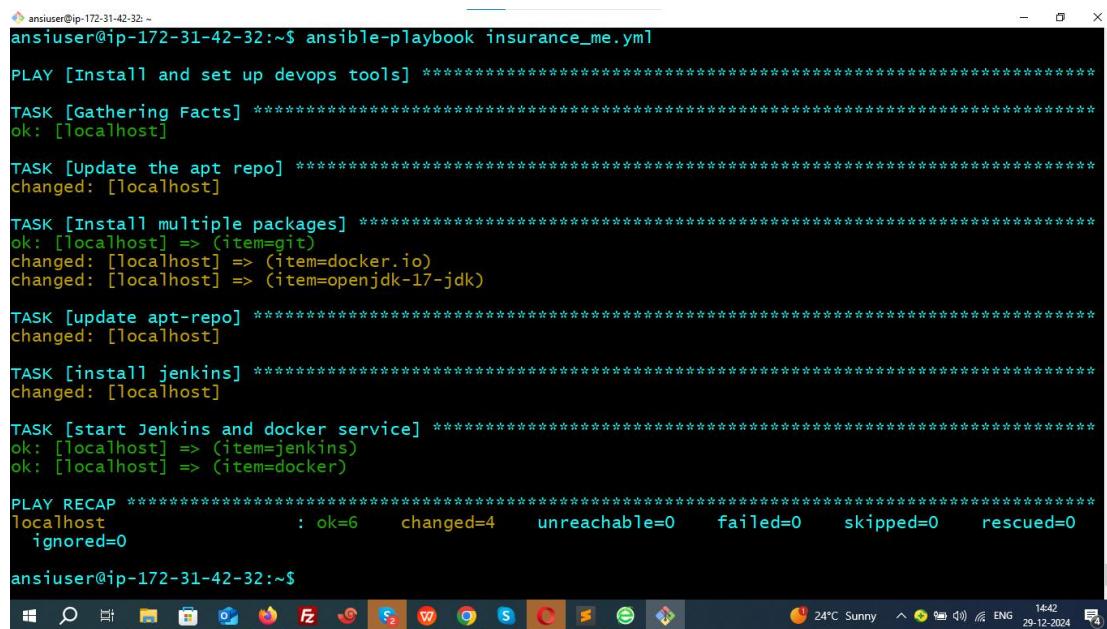
```

Run the playbook:

[ansible-playbook insurance_me.yml](#)

Run the playbook:

[ansible-playbook finance_me.yml](#)



```

ansiuser@ip-172-31-42-32:~$ ansible-playbook insurance_me.yml
PLAY [Install and set up devops tools] ****
TASK [Gathering Facts] ****
ok: [localhost]
TASK [Update the apt repo] ****
changed: [localhost]
TASK [Install multiple packages] ****
ok: [localhost] => (item=git)
changed: [localhost] => (item=docker.io)
changed: [localhost] => (item=openjdk-17-jdk)
TASK [update apt-repo] ****
changed: [localhost]
TASK [install jenkins] ****
changed: [localhost]
TASK [start Jenkins and docker service] ****
ok: [localhost] => (item=jenkins)
ok: [localhost] => (item=docker)

PLAY RECAP ****
localhost                  : ok=6    changed=4    unreachable=0    failed=0    skipped=0    rescued=0
ignored=0

ansiuser@ip-172-31-42-32:~$
```

Verify installations.

```

ansiuser@ip-172-31-42-32:~$ java --version
openjdk 17.0.13 2024-10-15
OpenJDK Runtime Environment (build 17.0.13+11-Ubuntu-2ubuntu124.04)
OpenJDK 64-Bit Server VM (build 17.0.13+11-Ubuntu-2ubuntu124.04, mixed mode, sharing)
ansiuser@ip-172-31-42-32:~$ jenkins --version
2.479.2
ansiuser@ip-172-31-42-32:~$ docker --version
Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
ansiuser@ip-172-31-42-32:~$ git --version
git version 2.43.0
ansiuser@ip-172-31-42-32:~$ |

```

24°C Sunny 14:43 29-12-2024

Verify:

`sudo systemctl status jenkins`

```

ansiuser@ip-172-31-42-32:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
   Active: active (running) since Sun 2024-12-29 09:12:16 UTC; 1min 38s ago
     Main PID: 6461 (java)
        Tasks: 45 (limit: 4676)
      Memory: 537.5M (peak: 545.3M)
        CPU: 19.730s
       CGroup: /system.slice/jenkins.service
              └─6461 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=
```

```

Dec 29 09:11:57 ip-172-31-42-32 jenkins[6461]: 6d7ae235b7694b7cb3f039dc12a1dee7
Dec 29 09:11:57 ip-172-31-42-32 jenkins[6461]: This may also be found at: /var/lib/jenkins/secrets/ini
Dec 29 09:11:57 ip-172-31-42-32 jenkins[6461]: ****
Dec 29 09:12:16 ip-172-31-42-32 jenkins[6461]: 2024-12-29 09:12:16.181+0000 [id=33]           INFO
Dec 29 09:12:16 ip-172-31-42-32 jenkins[6461]: 2024-12-29 09:12:16.200+0000 [id=23]           INFO
Dec 29 09:12:16 ip-172-31-42-32 systemd[1]: Started jenkins.service - Jenkins Continuous Integration S
Dec 29 09:12:17 ip-172-31-42-32 jenkins[6461]: 2024-12-29 09:12:17.829+0000 [id=48]           INFO
Dec 29 09:12:17 ip-172-31-42-32 jenkins[6461]: 2024-12-29 09:12:17.829+0000 [id=48]           INFO
lines 1-20/20 (END)

```

24°C Sunny 14:43 29-12-2024

3. Now access to the jenkins dashboard using below url:

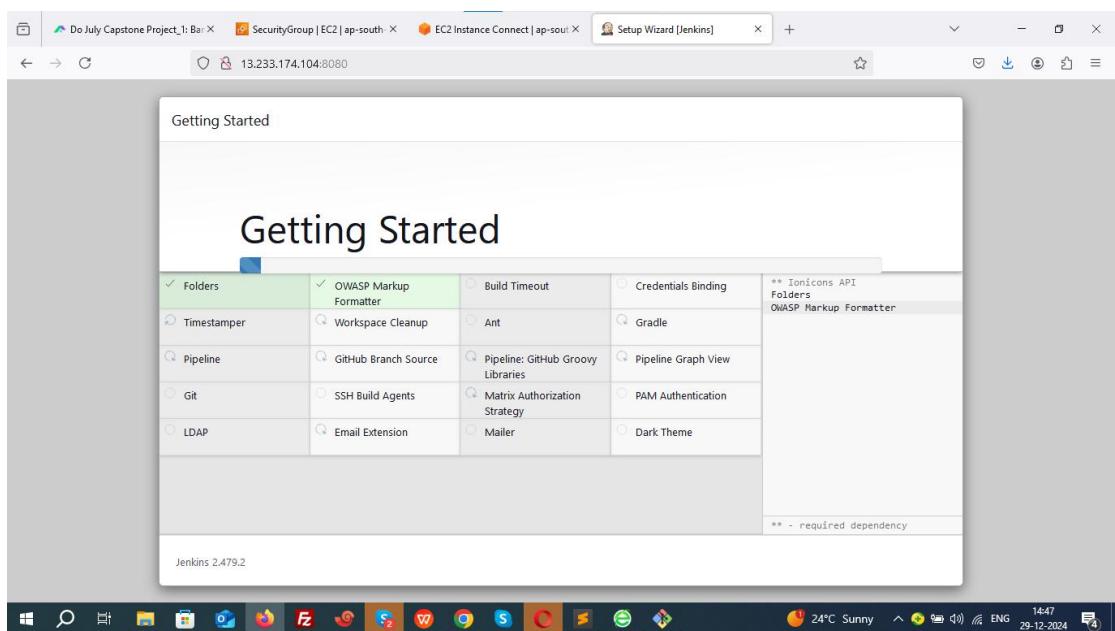
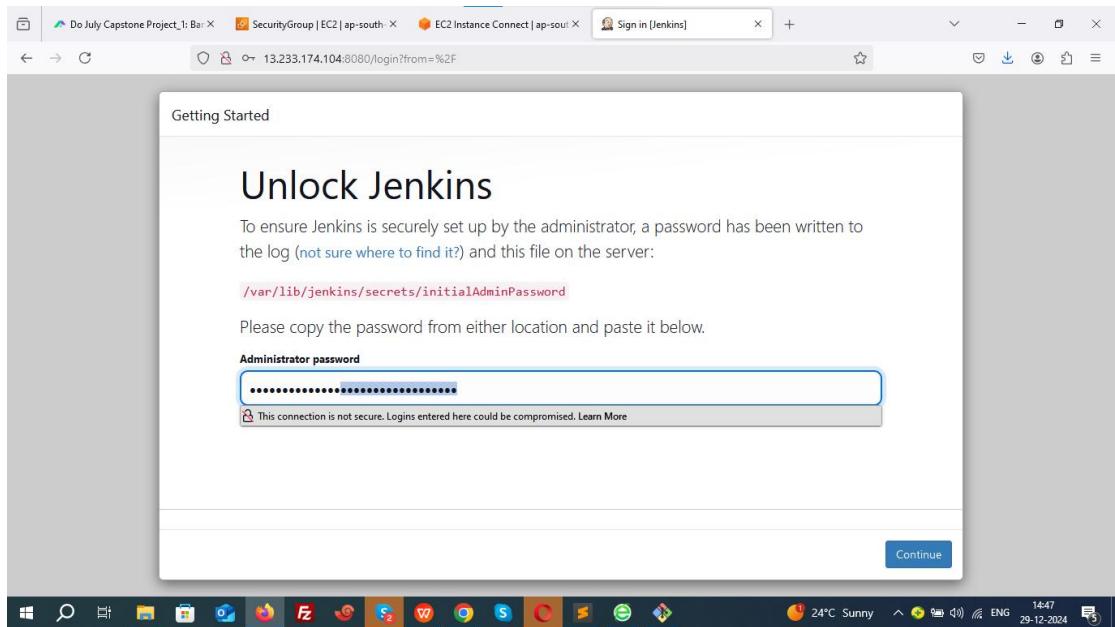
Enable port number 8080 on your controller

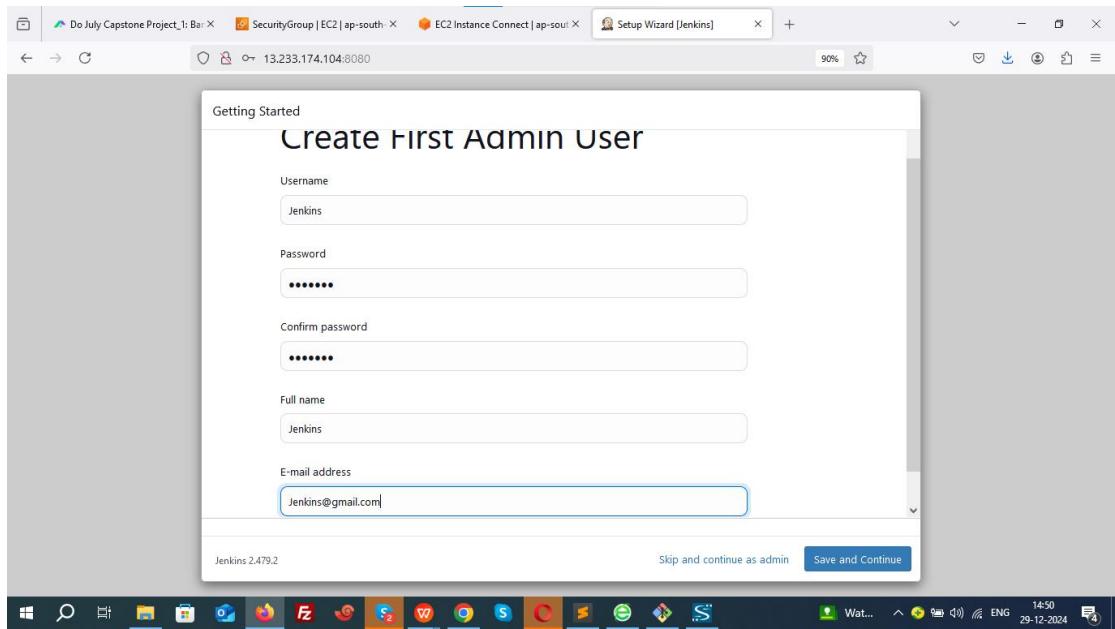
<http://13.233.174.104:8080/>

Default port: 8080

4. Set up plugins and admin user.

`cat /var/lib/jenkins/secrets/initialAdminPassword`

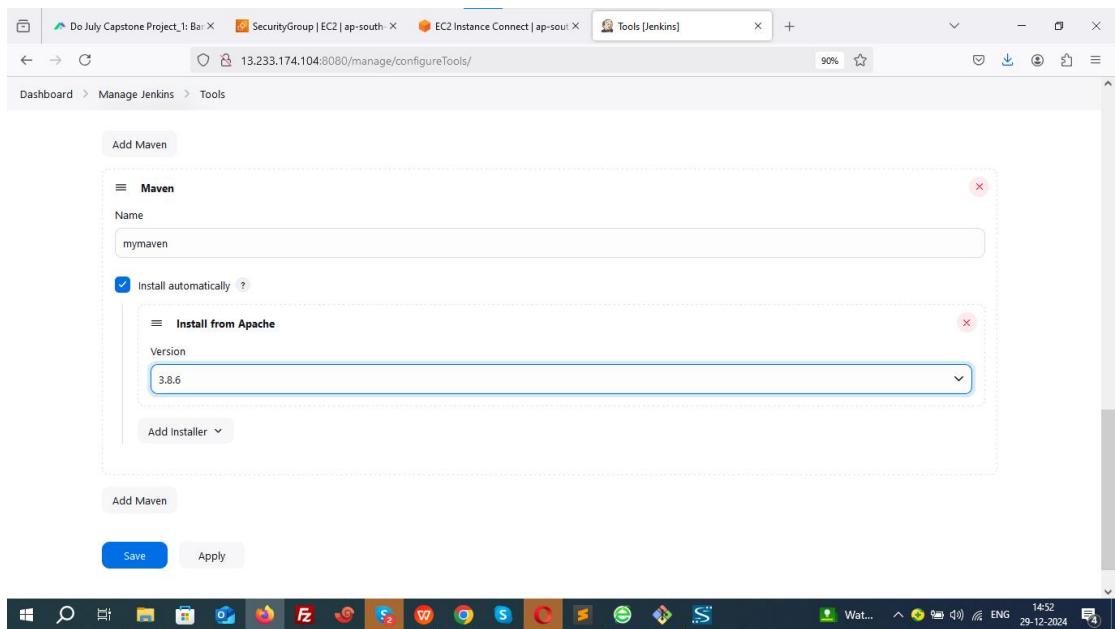




5. Configure Maven:

Go to **Manage Jenkins > Global Tool Configuration**.

Add Maven and set it up.



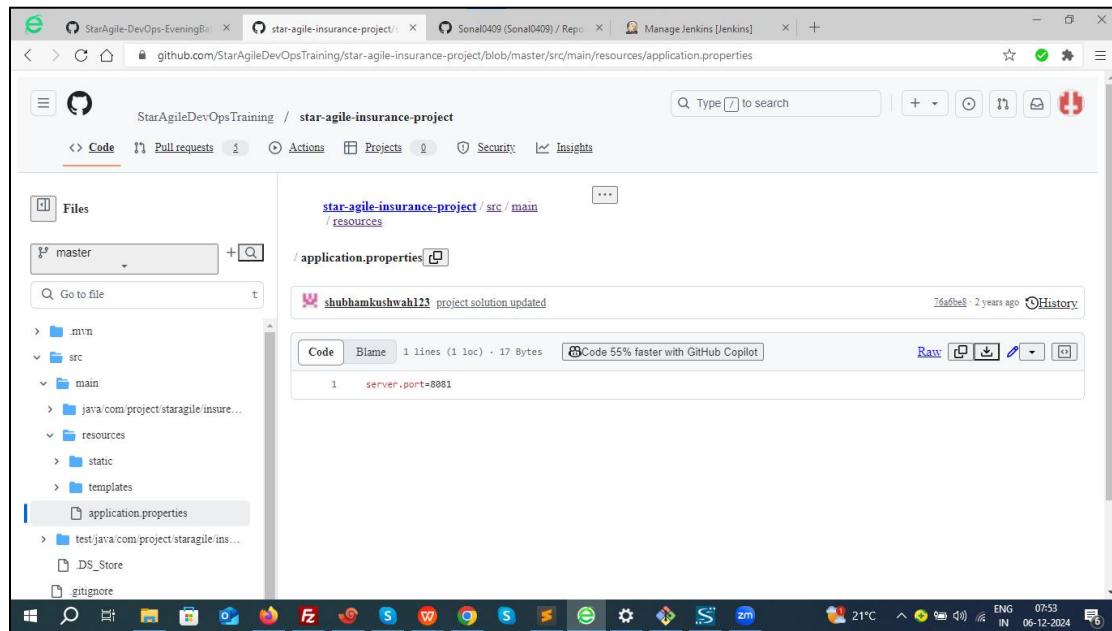
Make sure to give same name for maven in pipeline.

6. Create Pipeline:

Points to be noted:

In source code go to src/main/resources/application.properties

Check server port

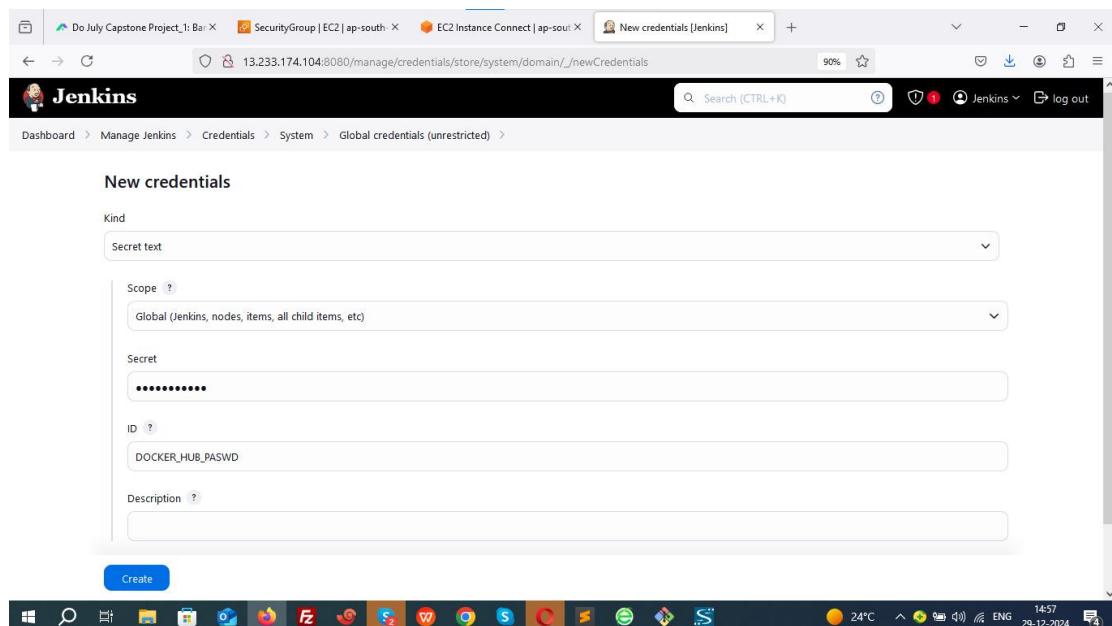


Map the internal container port (8081) to an external port on your host machine.

docker run -d -p 8081:8081 insurance_me:\$BUILD_NUMBER

Enable 8081 in your ec2 security groups.

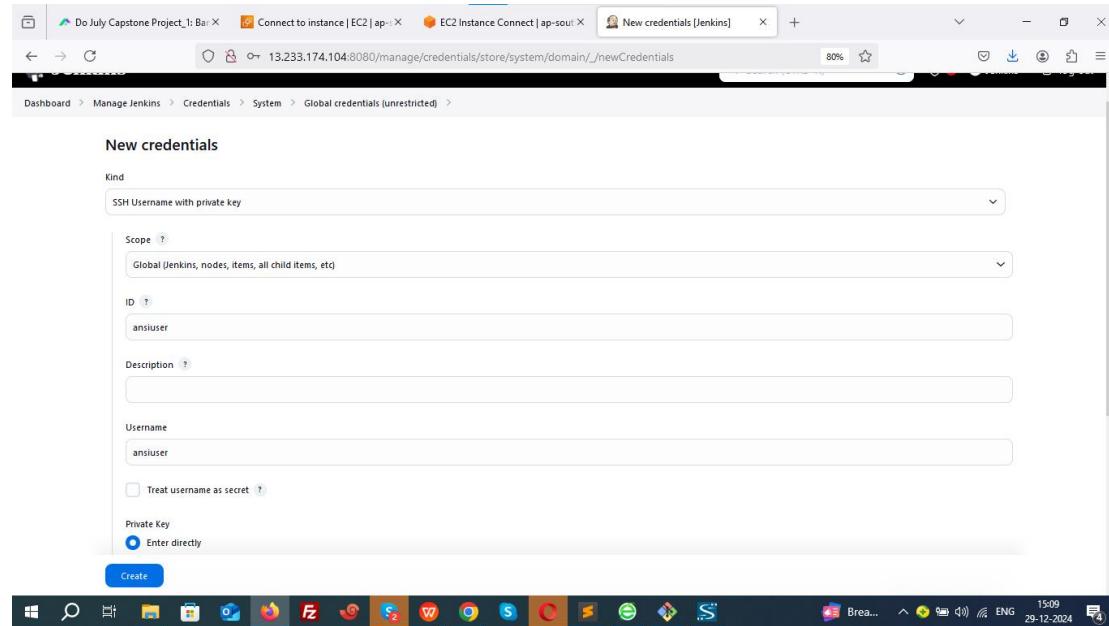
1. Check and Configure Docker Hub Credentials in Jenkins



2. Install Ansible plugin.

3. Add ansiuuser credential In Jenkins

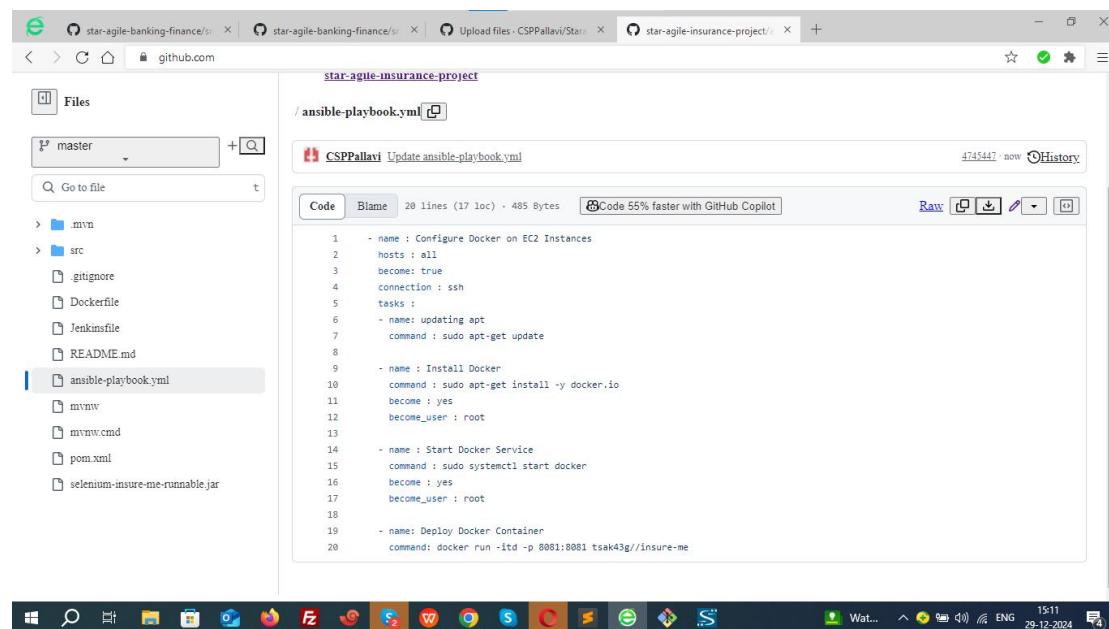
(In Jenkins we have created ssh key for ansiuuser)



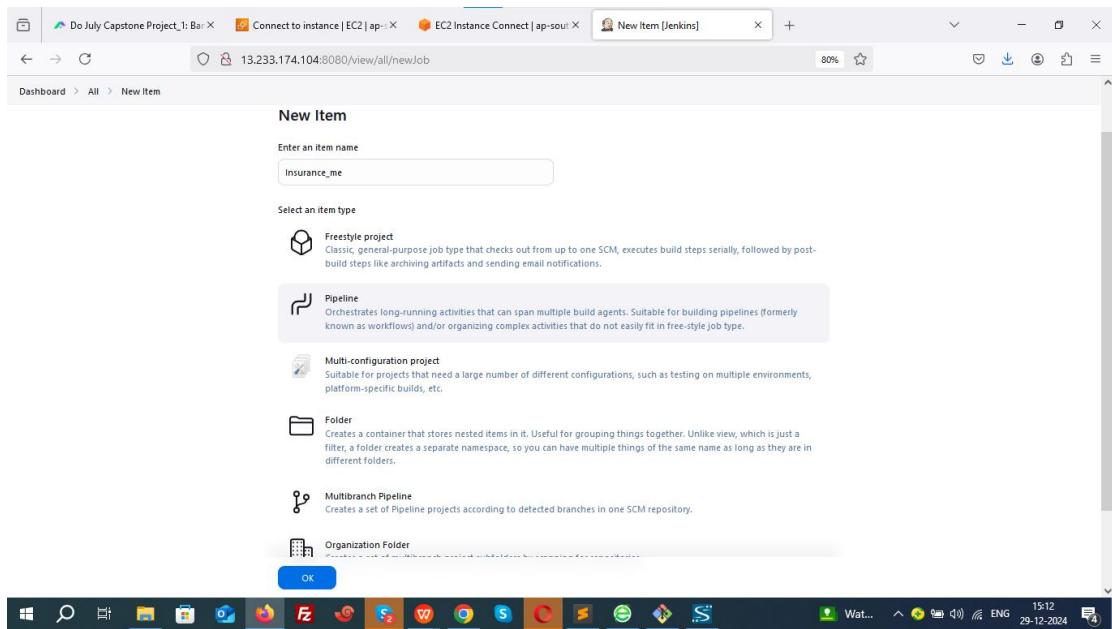
4. We are going to run playbook which is already available on git hub.

ansible-playbook.yml

Just edit image name



Navigate to New Item > Pipeline.



Add below pipeline

```
pipeline {  
    agent any  
  
    tools {  
        maven 'mymaven'  
    }  
  
    stages {  
        stage('Clone Repo') {  
            steps {  
                git 'https://github.com/CSPPallavi/star-agile-insurance-project.git'  
            }  
        }  
    }  
}
```

```
stage('Test Code') {  
    steps {  
        sh 'mvn test'  
    }  
}  
  
stage('Build Code') {  
    steps {  
        sh 'mvn package'  
    }  
}  
  
stage('Build Image') {  
    steps {  
        sh 'docker build -t banking_finance:$BUILD_NUMBER.'  
    }  
}  
  
stage('Push the Image to DockerHub') {  
    steps {  
        withCredentials([string(credentialsId:  
        'DOCKER_HUB_PASWD', variable: 'DOCKER_HUB_PASWD')]) {  
            sh 'docker login -u tsak43g -p  
            ${DOCKER_HUB_PASWD}'  
        }  
    }  
}
```

```

    sh 'docker tag banking_finance:$BUILD_NUMBER
tsak43g/insure-me:$BUILD_NUMBER'

    sh 'docker push tsak43g/insure-me:$BUILD_NUMBER'

}

stage('Deploy to Worker Server') {

    steps {

        ansiblePlaybook(
            credentialsId: 'ansiuser', // Use the credentials ID for
            the SSH private key required by Ansible

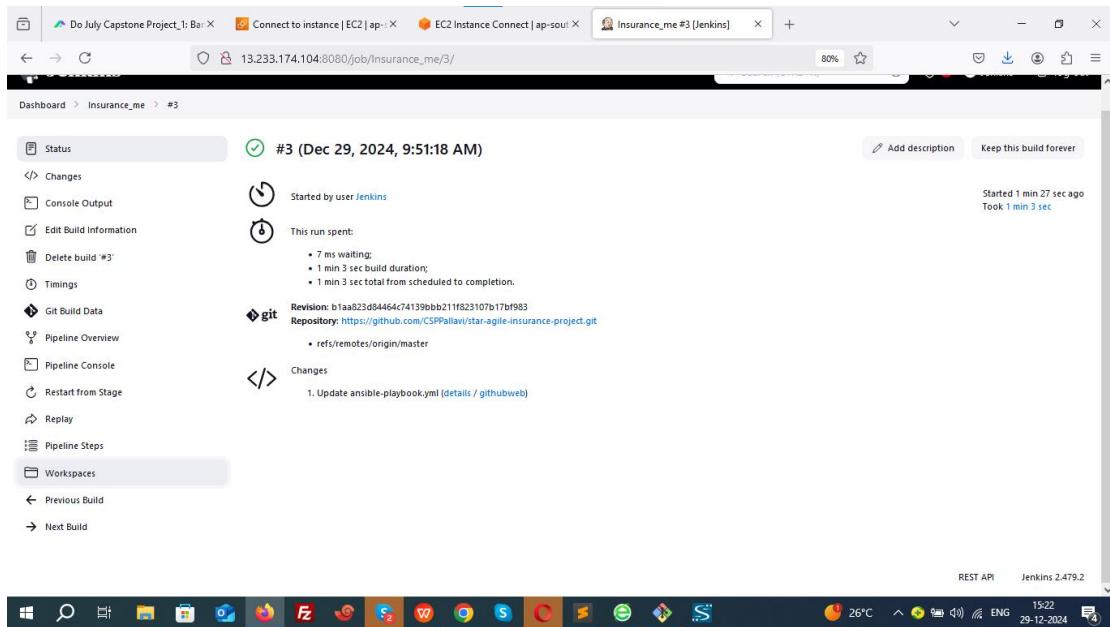
            playbook: 'ansible-playbook.yml',
            inventory: '/etc/ansible/hosts',
            disableHostKeyChecking: true,
            become: true

        )
    }
}
}
```

7. Before building the job allow Jenkins user to Run Docker Commands:

chmod -R 777 /var/run/docker.sock

8. Build the job successfully.



9. Verify docker image and container.

```

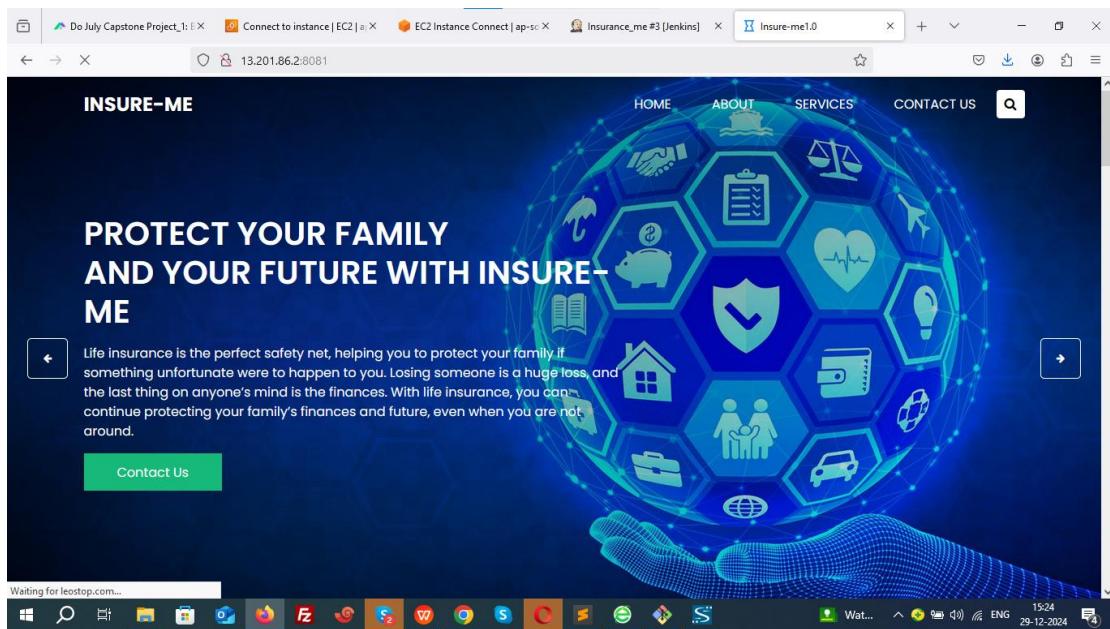
Room Number []:
Work Phone []:
Home Phone []:
Other []:
Is the information correct? [Y/n]
info: Adding new user 'ansiuser' to supplemental / extra groups 'users' ...
info: Adding user 'ansiuser' to group 'users' ...
root@ip-172-31-39-18:/home/ubuntu# sudo vim /etc/ssh/sshd_config
root@ip-172-31-39-18:/home/ubuntu# sudo su - ansiuser
ansiuser@ip-172-31-39-18:~$ mkdir -m 700 /home/ansiuser/.ssh
ansiuser@ip-172-31-39-18:~$ echo "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIGl4cn0mH2lma+55N0L3BL5N20gmGX4CrrBKHWEMZ0s ansiuser@ip-172-31-42-32" > /home/ansiuser/.ssh/authorized_keys
ansiuser@ip-172-31-39-18:~$ chmod 600 /home/ansiuser/.ssh/authorized_keys
ansiuser@ip-172-31-39-18:~$ docker images
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping":
dial unix /var/run/docker.sock: connect: permission denied
ansiuser@ip-172-31-39-18:~$ docker images
REPOSITORY          TAG           IMAGE ID      CREATED        SIZE
tsak43/insure-me   2             f15918579c64   6 minutes ago  695MB
ansiuser@ip-172-31-39-18:~$ sudo docker ps
CONTAINER ID        IMAGE          COMMAND       CREATED        STATUS        PORTS     NAMES
cddc241f4395        tsak43/insure-me:2    "java -jar /app.jar"   About a minute ago   Up 59 seconds   0.0.0.0:8081->8081/tcp, :::8081->8081/tcp   vigilant
ansiuser@ip-172-31-39-18:~$ 
```

i-07c55da8f20659b90 (Insure_me_worker)

PublicIPs: 13.201.86.2 PrivateIPs: 172.31.39.18

Access the container

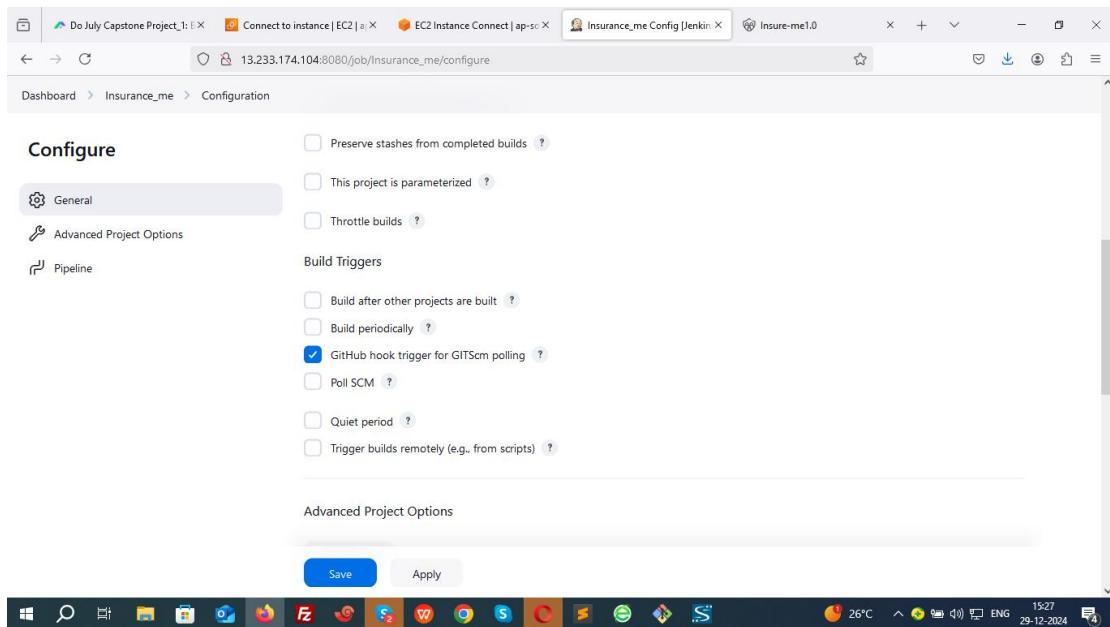
<http://13.201.86.2:8081/>



Application is deployed successfully in docker container.

STEP 5: WEBHOOK CONFIGURATION:

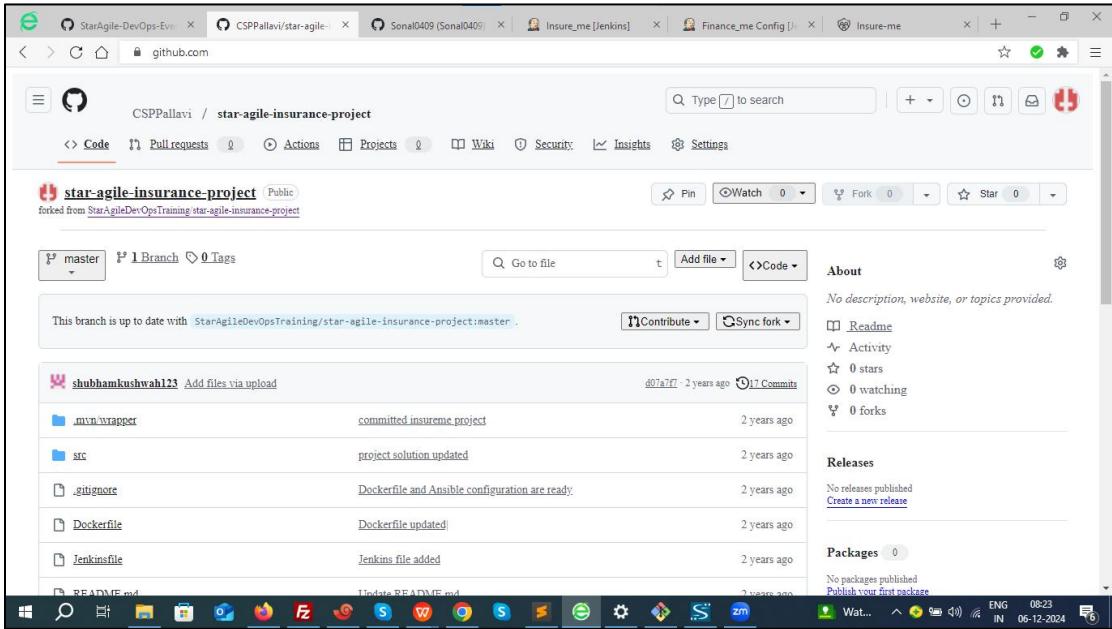
Go to your Jenkins dashboard and add build trigger- **GitHub hook trigger for GITScm polling**



Configuring a webhook in Git is a crucial step to enable automated actions, such as triggering a CI/CD pipeline in Jenkins or notifying an application about repository events.

Step 1: Open Your Repository Settings

1. Navigate to the repository where you want to configure the webhook.



Step 2: Find the Webhooks Section

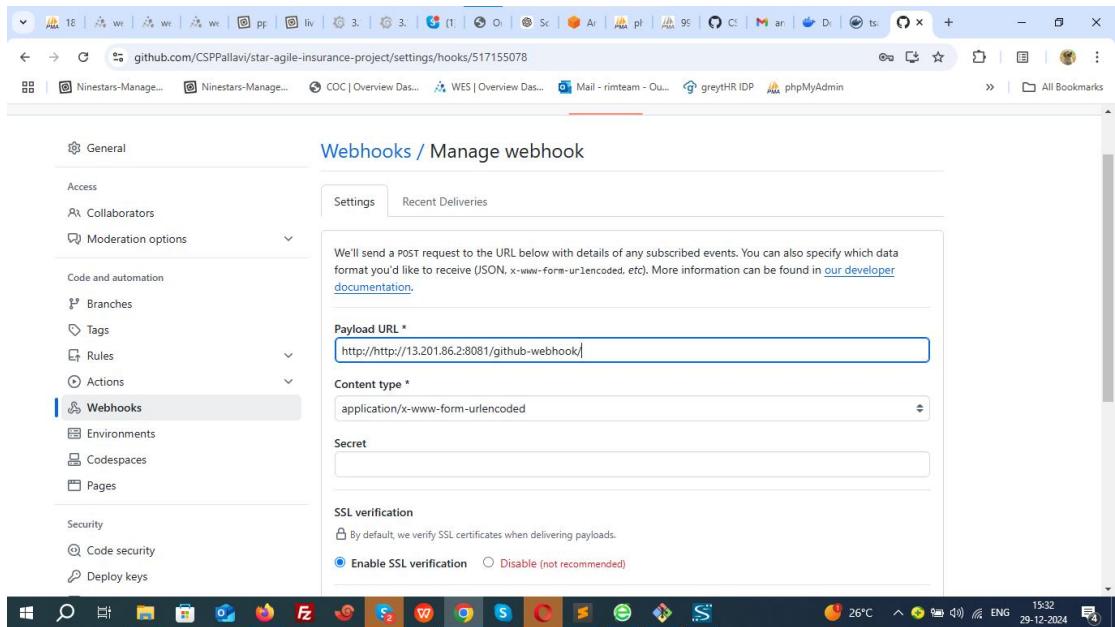
Go to **Repository Settings > Webhooks**.

Step 3: Add a New Webhook

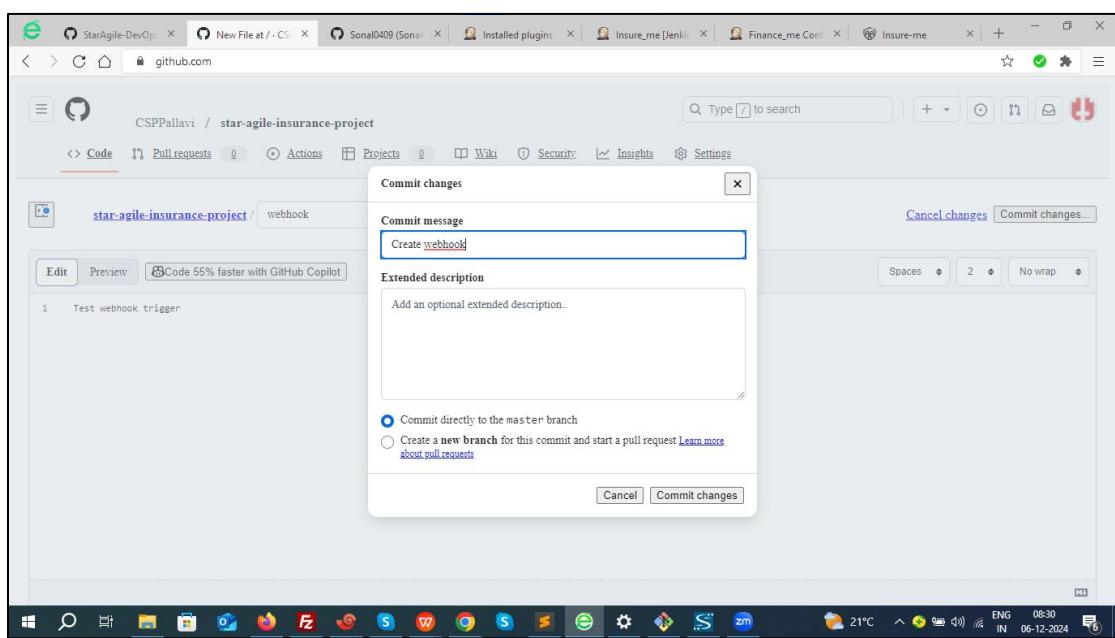
Fill in the webhook details:

Payload URL:

- This is the endpoint that will receive the webhook payload.
 - <http://<Jenkins-URL>/github-webhook/>
- <http://13.233.174.104:8080//github-webhook/>



Step 4 : Commit changes.



Step 5 : Testing webhook.

Lets make changes in the application Index.html page.



Let's make changes in welcome page where it says PROTECT YOUR FAMILY AND YOUR FUTURE.

Let's change this to PROTECT YOUR FAMILY AND YOUR FUTURE WITH INSURE-ME

Go to index.html path in your git hub

star-agile-insurance-project/src/main/resources/static

```

PROTECT
<n>
    Protect Your Family <br>
    and Your Future WITH INSURE-ME
</n>
<p>
    Life insurance is the perfect safety net, helping you to protect your family if something unfortunate were to happen
</p>
<div class="btn-box">
    <a href="#" class="btn1">
        Contact Us
    </a>
</div>
...

```

Now commit the changes.

Whenever changes are done to code, new build will start automatically.