SOURCECODE

Create 3 AWS ec2 instances in ubuntu

First instance - Ansible
Second instance - server 1
Third instance - server 2

Login to Ansible ec2 instance and use these command

- Switch as root : sudo su-
- update packages : apt update -y
- run the following command to include the official projects PPA (personal package archive) in your systems list of source:

```
apt-add-repository ppa:ansible/ansible
```

- Next, refresh your systems package index so that it is aware of the packages available in the newly included PPA : apt update
- Following this update, you can install the ansible software with: apt install ansible-y.
- check ansible version : ansible --version
- go to hosts and add your server 1 and server2

```
nano /etc/hosts
Add: 15.168.39.192 server1
     13.208.141.45 server2
```

- Generate ssh key from ansbile server

  ssh-keygen -t rsa (and) press → enter → enter → enter

  you can see ssh keys of public key and private key.
- Copy the public key (id-rsa.pub) and paste it in authorize-key on server1 and server2.

```
cat id-rsa.pub
```

- Go to server1 and server2
- Login server1 and paste this public key in .ssh/authorize/key

nano .ssh/authorize-keys
(save it and come out from the shell)

- Login server2 and paste this public key in .ssh/authorize_key
  nano.ssh/authorize_keys
  (save it and come out from the shell)
- Return to the ansible server and check if ping is working on server1 and server2
  ping server1
  ping server2
- create a directory in the name of ansible
  mkdir ansible
- Get in the ansible directory
  cd ansible
- Create an inventory file and then add these lines hosts
  nano inventory

```
[webservers]
server1
server2
```

(save it and come out from shell)

- create ansible.cfg file and add these lines
  nano ansible.cfg

```
[defaults]
inventory = /root/ansible/inventory
remote_user = ubuntu
ask_pass = false
```

(save and come out of it)

- For testing purpose, we need to install nginx in server1 and apache in server 2 from ansible server
- Create a yml file for install nginx and apache in server 1 and server2.
  nano install_webservers.yml

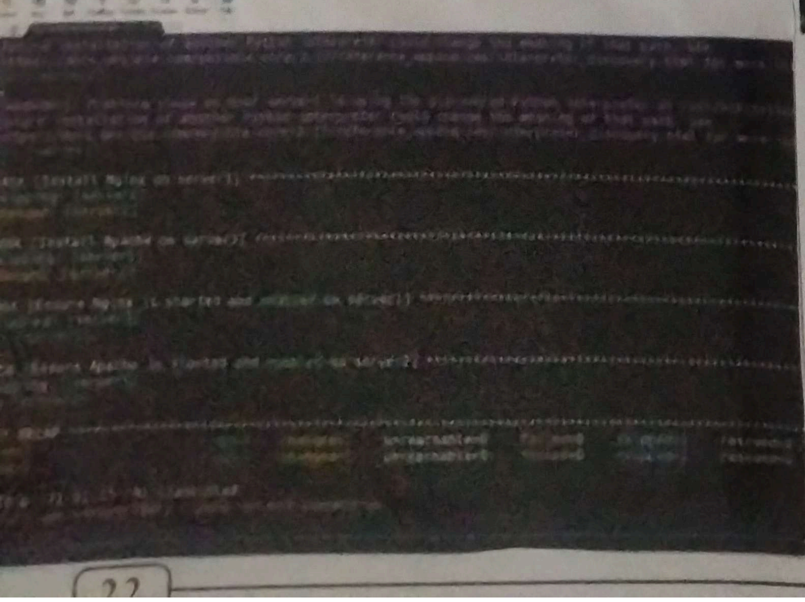
```
---
- name: Install Web Servers
  host: webservers
  become: true
  tasks:
```

OUTPUT

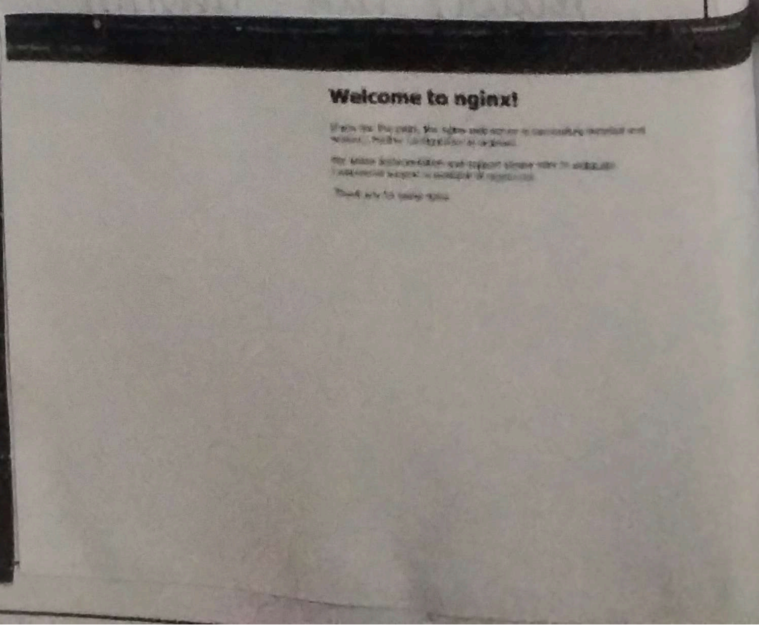
```
- name : Install Nginx on server 1
  apt :
     name : nginx
     state : present
  when : inventory_hostname == 'server1'
- name : Instale Apache on server 2
  apt :
     name : apache2
     state : present
  when : inventory_hostname == 'server2'
- name : Ensure Nginx is started and enables on server1
  service :
     name : nginx
     state : started
     enabled : yes
  when : inventory_hostname == 'server 1'
- name : Ensure Apache is started and enable on server2
  service :
     name : apache2
     state : started
     enabled : yes
  when : inventory_hostname == 'server 2'
```

(save and come out from shell)

- Run ansible yml file using the following command.
ansible-playbook -i /root/ansible/inventory installe-webservers.yml
Here you can see installing nginx and apache each servers and you can test by copy each servers ip and paste it in the browser.



```
[illegible]

TASK [Install Nginx on server1] ************************************************
[illegible]

TASK [Install Apache on server2] ***********************************************
[illegible]

TASK [Ensure Nginx is started and enabled on server1] **************************
[illegible]

TASK [Ensure Apache is started and enabled on server2] *************************
[illegible]

PLAY RECAP *********************************************************************
[illegible]    unreachable=0    failed=0    [illegible]
[illegible]    unreachable=0    failed=0    [illegible]
```

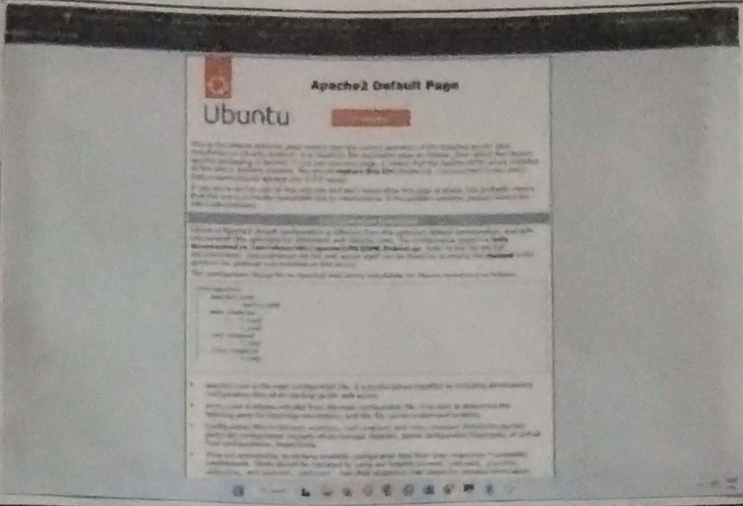


Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.



Apache2 Default Page

Ubuntu

VIVA QUESTIONS

1. What is the difference between a Jenkins server and an Ansible controller?

Ans. Jenkins: automates CI/CD pipelines, managing build, test, deployment.
Ansible: automates infrastructure management and configuration using declarative playbooks.

2. Why is Jenkins often integrated with tools like Ansible in DevOps pipelines?

Ans. To automate infrastructure provisioning, configuration management, and deployment ensuring efficient DevOps flows.

3. How do you create an Ansible controller node on AWS?

Ans. Launch an EC2 instance, install Ansible using yum or apt, configure SSH access, and define inventory and playbooks.

4. How can you test if the Ansible controller can communicate with the managed nodes?

Ans. Run ansible all -m ping to check connectivity between the ansible controller and managed nodes. or using ping.

5. How does Jenkins push Docker images to Docker Hub?

Ans. Jenkins builds image, logs into Docker Hub, tags the image, and pushes it using docker push <repository>:<tag>.