Name: BONIFACIO, REDJ GUILLIAN F.	**Date Performed:** October 16, 2025
Course/Section: CPE31S4	**Date Submitted:** October 16, 2025
Instructor: Engr. VALENZUELA, ROBIN	**Semester and SY:** 2nd Sem. SY 2025 - 2026

Activity 7: Managing Files and Creating Roles in Ansible

1. Objectives:

1.1 Manage files in remote servers

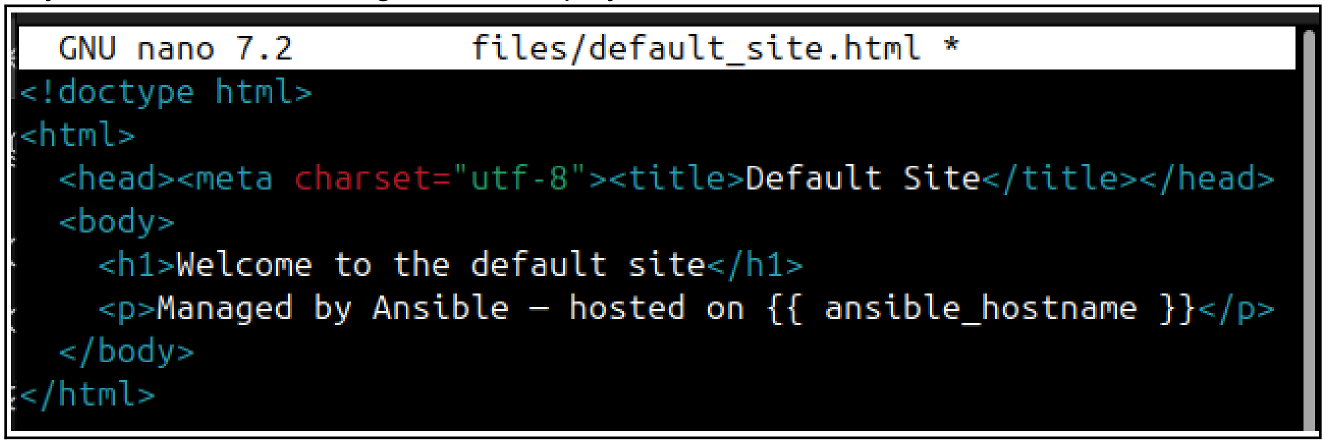
1.2 Implement roles in ansible

2. Discussion:

In this activity, we look at the concept of copying a file to a server. We are going to create a file into our git repository and use Ansible to grab that file and put it into a particular place so that we could do things like customize a default website, or maybe install a default configuration file. We will also implement roles to consolidate plays.

Task 1: Create a file and copy it to remote servers

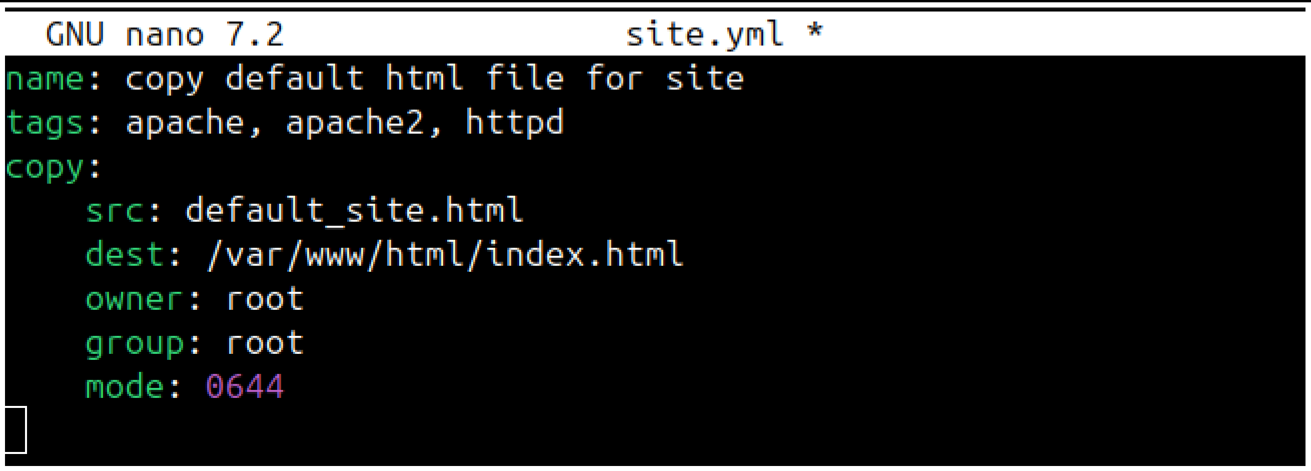
1. Using the previous directory we created, create a directory, and named it "*files*." Create a file inside that directory and name it "*default_site.html*." Edit the file and put basic HTML syntax. Any content will do, as long as it will display text later. Save the file and exit.



```
  GNU nano 7.2          files/default_site.html *
<!doctype html>
<html>
  <head><meta charset="utf-8"><title>Default Site</title></head>
  <body>
    <h1>Welcome to the default site</h1>
    <p>Managed by Ansible — hosted on {{ ansible_hostname }}</p>
  </body>
</html>
```

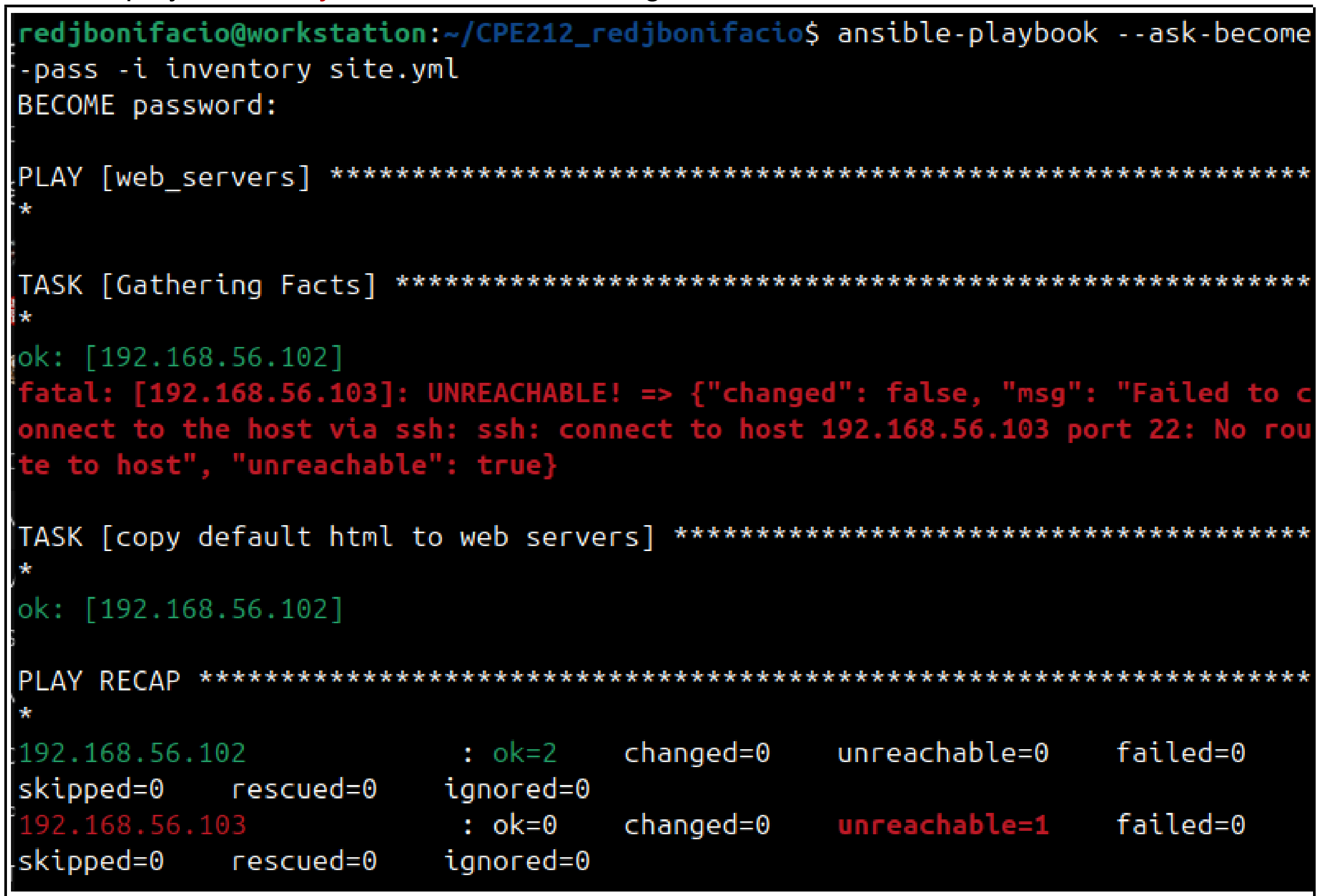
2. Edit the *site.yml* file and just below the *web_servers* play, create a new file to copy the default html file for site:

```
- name: copy default html file for site

  tags: apache, apache2, httpd
  copy:
    src: default_site.html
    dest: /var/www/html/index.html
    owner: root
    group: root
    mode: 0644
```



```
  GNU nano 7.2                          site.yml *
name: copy default html file for site
tags: apache, apache2, httpd
copy:
    src: default_site.html
    dest: /var/www/html/index.html
    owner: root
    group: root
    mode: 0644
```

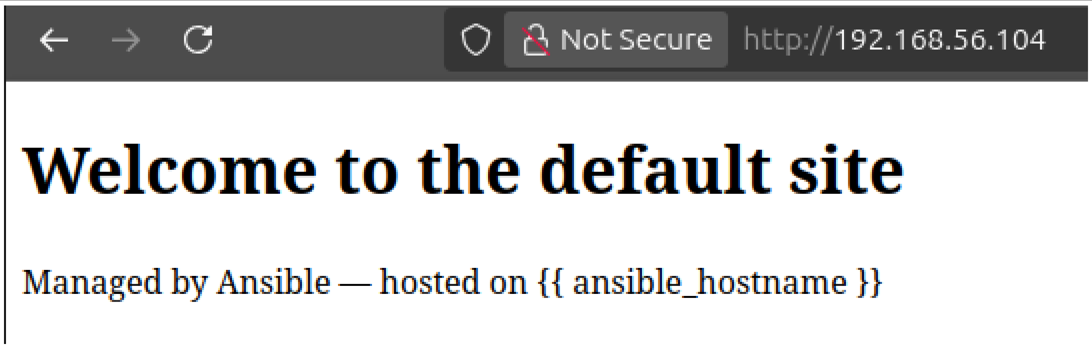
3. Run the playbook *site.yml*. Describe the changes.



```
redjbonifacio@workstation:~/CPE212_redjbonifacio$ ansible-playbook --ask-become
-pass -i inventory site.yml
BECOME password:

PLAY [web_servers] *************************************************************
*

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.102]
fatal: [192.168.56.103]: UNREACHABLE! => {"changed": false, "msg": "Failed to c
onnect to the host via ssh: ssh: connect to host 192.168.56.103 port 22: No rou
te to host", "unreachable": true}

TASK [copy default html to web servers] ****************************************
*
ok: [192.168.56.102]

PLAY RECAP *********************************************************************
*
192.168.56.102             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103             : ok=0    changed=0    unreachable=1    failed=0
skipped=0    rescued=0    ignored=0
```

Running the playbook copied the default HTML file to the web servers, showing "changed" on the first run and "ok" on later runs since the file was already updated.

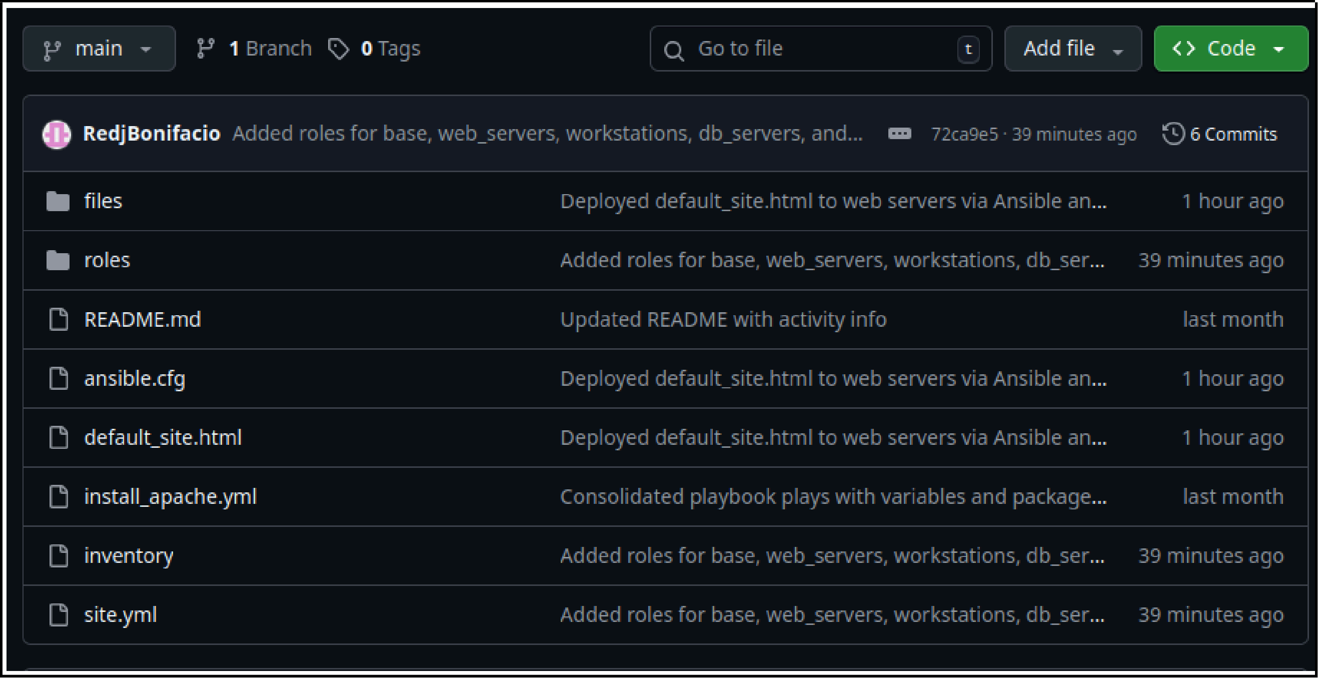
4. Go to the remote servers (*web_servers*) listed in your inventory. Use cat command to check if the index.html is the same as the local repository file (*default_site.html*). Do both for Ubuntu and CentOS servers. On the CentOS server, go to the browser and type its IP address. Describe the output.


Not Secure http://192.168.56.104

Welcome to the default site

Managed by Ansible — hosted on {{ ansible_hostname }}

5. Sync your local repository with GitHub and describe the changes.


main · 1 Branch · 0 Tags · Go to file · Add file · Code

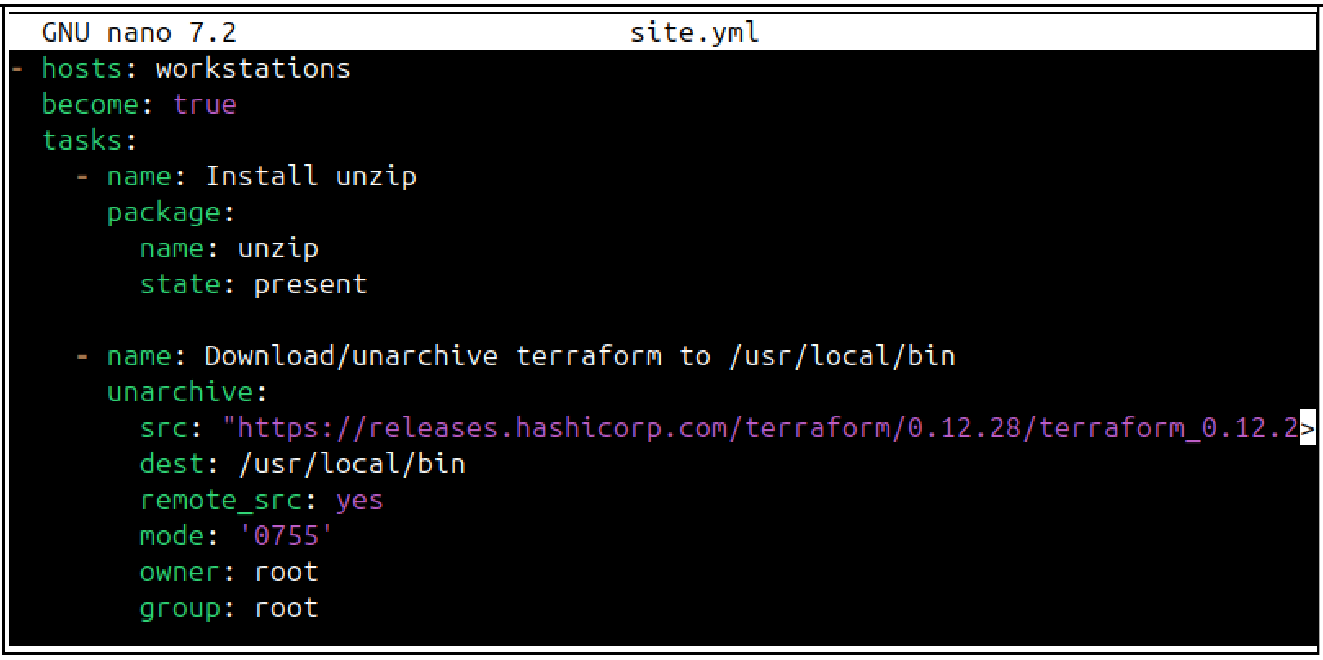
RedjBonifacio Added roles for base, web_servers, workstations, db_servers, and... 72ca9e5 · 39 minutes ago · 6 Commits

files	Deployed default_site.html to web servers via Ansible an...	1 hour ago
roles	Added roles for base, web_servers, workstations, db_ser...	39 minutes ago
README.md	Updated README with activity info	last month
ansible.cfg	Deployed default_site.html to web servers via Ansible an...	1 hour ago
default_site.html	Deployed default_site.html to web servers via Ansible an...	1 hour ago
install_apache.yml	Consolidated playbook plays with variables and package...	last month
inventory	Added roles for base, web_servers, workstations, db_ser...	39 minutes ago
site.yml	Added roles for base, web_servers, workstations, db_ser...	39 minutes ago

Task 2: Download a file and extract it to a remote server

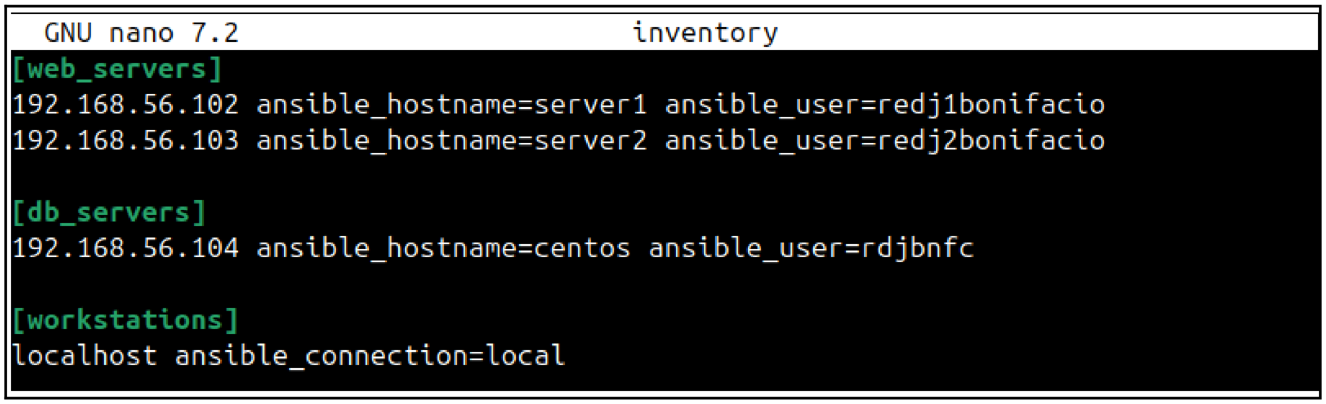
1. Edit the site.yml. Just before the web_servers play, create a new play:

```
- hosts: workstations
  become: true
  tasks:

  - name: install unzip
    package:
      name: unzip

  - name: install terraform
    unarchive:
                                                                        src:
https://releases.hashicorp.com/terraform/0.12.28/terraform_0.12.28_linux_amd64.zip
      dest: /usr/local/bin
      remote_src: yes
      mode: 0755
      owner: root
      group: root
```



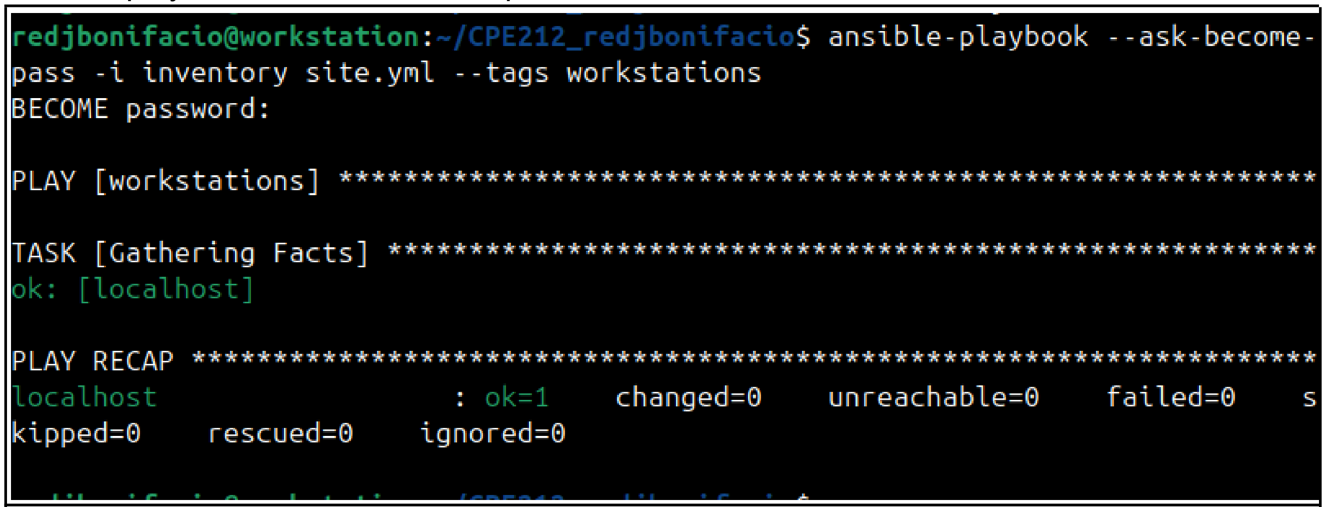
```
  GNU nano 7.2                         site.yml
- hosts: workstations
  become: true
  tasks:
    - name: Install unzip
      package:
        name: unzip
        state: present

    - name: Download/unarchive terraform to /usr/local/bin
      unarchive:
        src: "https://releases.hashicorp.com/terraform/0.12.28/terraform_0.12.2>
        dest: /usr/local/bin
        remote_src: yes
        mode: '0755'
        owner: root
        group: root
```

2. Edit the inventory file and add workstations group. Add any Ubuntu remote server. Make sure to remember the IP address.



```
  GNU nano 7.2                         inventory
[web_servers]
192.168.56.102 ansible_hostname=server1 ansible_user=redj1bonifacio
192.168.56.103 ansible_hostname=server2 ansible_user=redj2bonifacio

[db_servers]
192.168.56.104 ansible_hostname=centos ansible_user=rdjbnfc

[workstations]
localhost ansible_connection=local
```

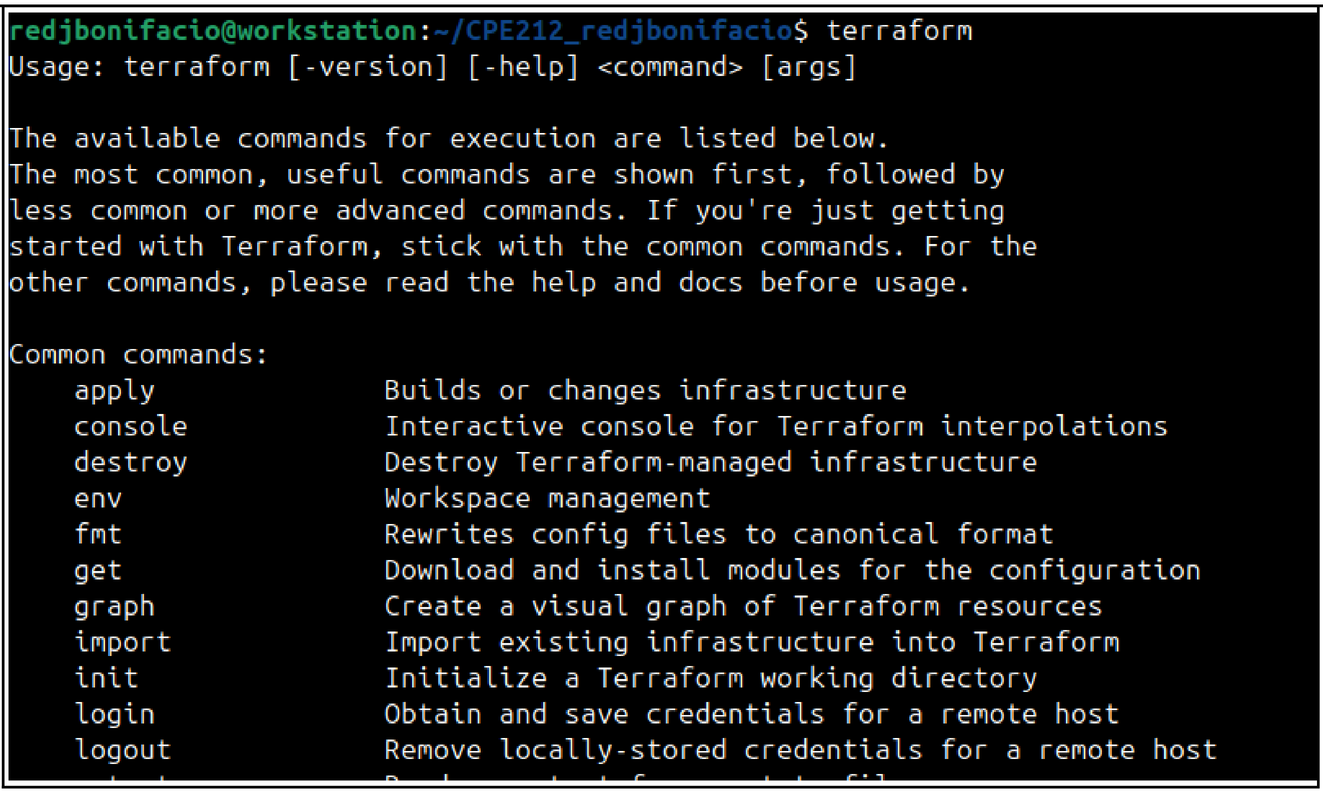
3. Run the playbook. Describe the output.



```
redjbonifacio@workstation:~/CPE212_redjbonifacio$ ansible-playbook --ask-become-
pass -i inventory site.yml --tags workstations
BECOME password:

PLAY [workstations] *************************************************************

TASK [Gathering Facts] **********************************************************
ok: [localhost]

PLAY RECAP **********************************************************************
localhost                  : ok=1    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
```

Running the playbook installed Unzip and downloaded Terraform on the workstation, showing "changed" when first installed and "ok" afterward once everything was already set up.

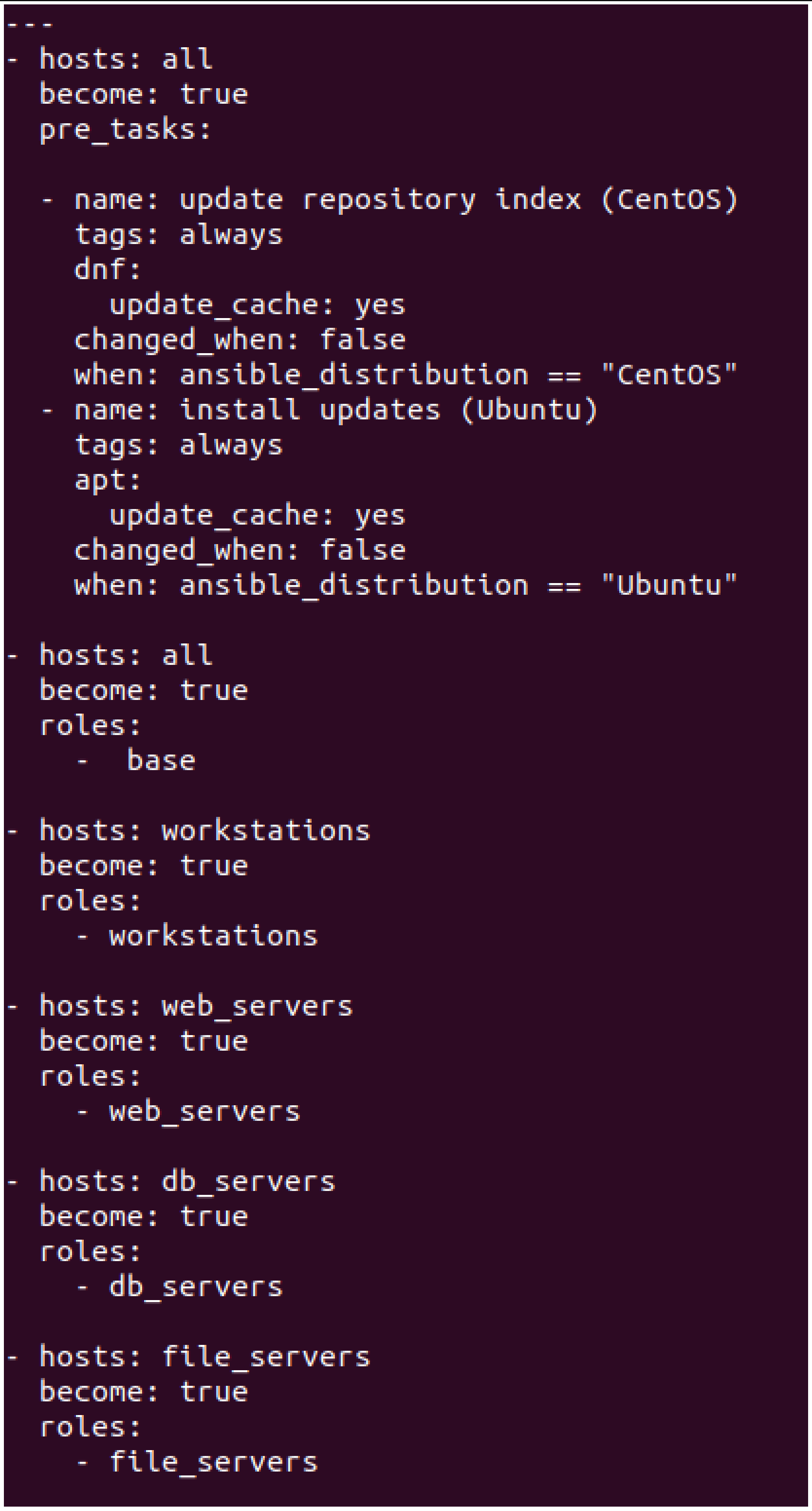
4. On the Ubuntu remote workstation, type terraform to verify installation of terraform. Describe the output.



```
redjbonifacio@workstation:~/CPE212_redjbonifacio$ terraform
Usage: terraform [-version] [-help] <command> [args]

The available commands for execution are listed below.
The most common, useful commands are shown first, followed by
less common or more advanced commands. If you're just getting
started with Terraform, stick with the common commands. For the
other commands, please read the help and docs before usage.

Common commands:
    apply              Builds or changes infrastructure
    console            Interactive console for Terraform interpolations
    destroy            Destroy Terraform-managed infrastructure
    env                Workspace management
    fmt                Rewrites config files to canonical format
    get                Download and install modules for the configuration
    graph              Create a visual graph of Terraform resources
    import             Import existing infrastructure into Terraform
    init               Initialize a Terraform working directory
    login              Obtain and save credentials for a remote host
    logout             Remove locally-stored credentials for a remote host
```

Task 3: Create roles

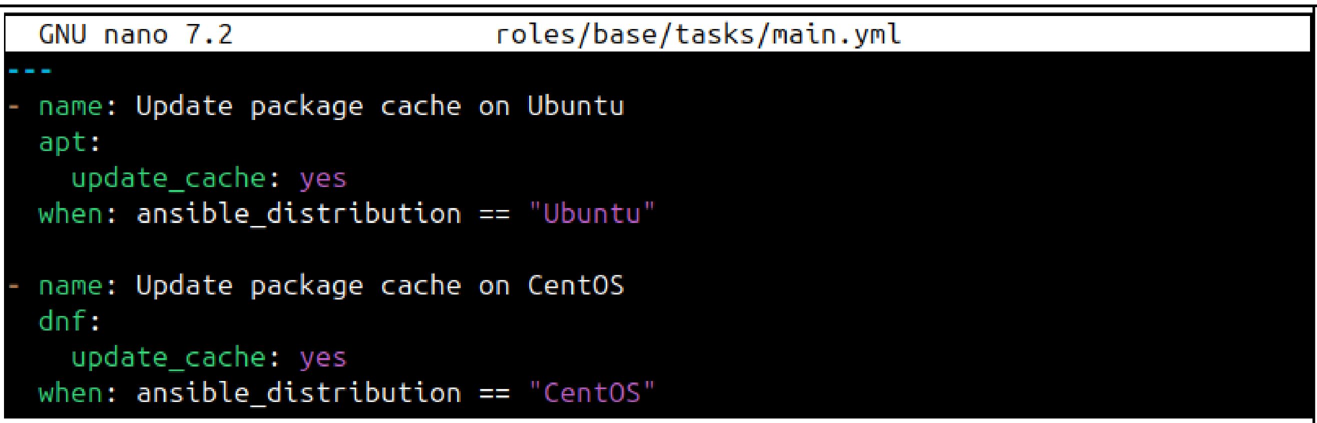
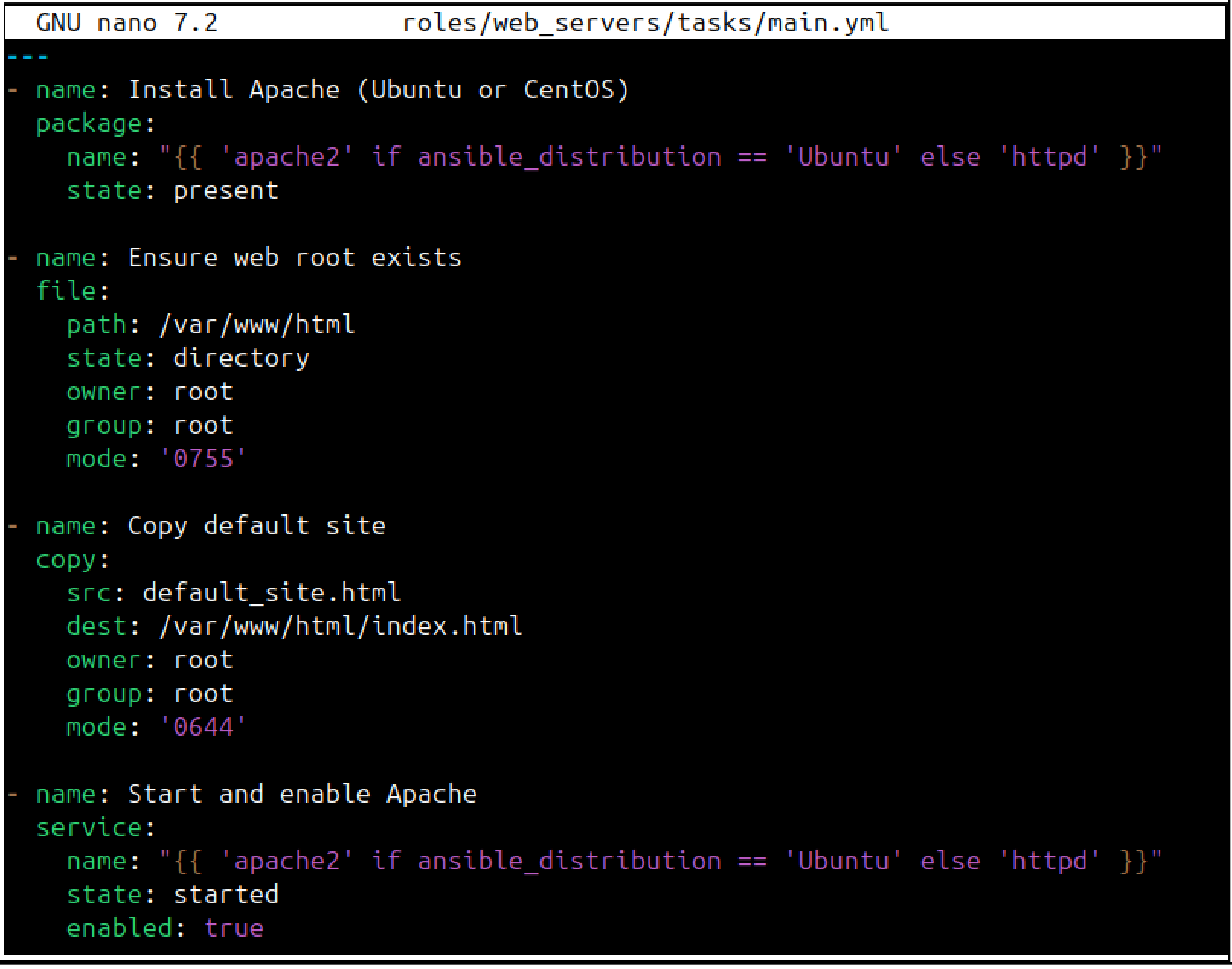
1. Edit the site.yml. Configure roles as follows: (make sure to create a copy of the old site.yml file because you will be copying the specific plays for all groups)



```
---
- hosts: all
  become: true
  pre_tasks:

  - name: update repository index (CentOS)
    tags: always
    dnf:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "CentOS"
  - name: install updates (Ubuntu)
    tags: always
    apt:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "Ubuntu"

- hosts: all
  become: true
  roles:
    -  base

- hosts: workstations
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
    - web_servers

- hosts: db_servers
  become: true
  roles:
    - db_servers

- hosts: file_servers
  become: true
  roles:
    - file_servers
```

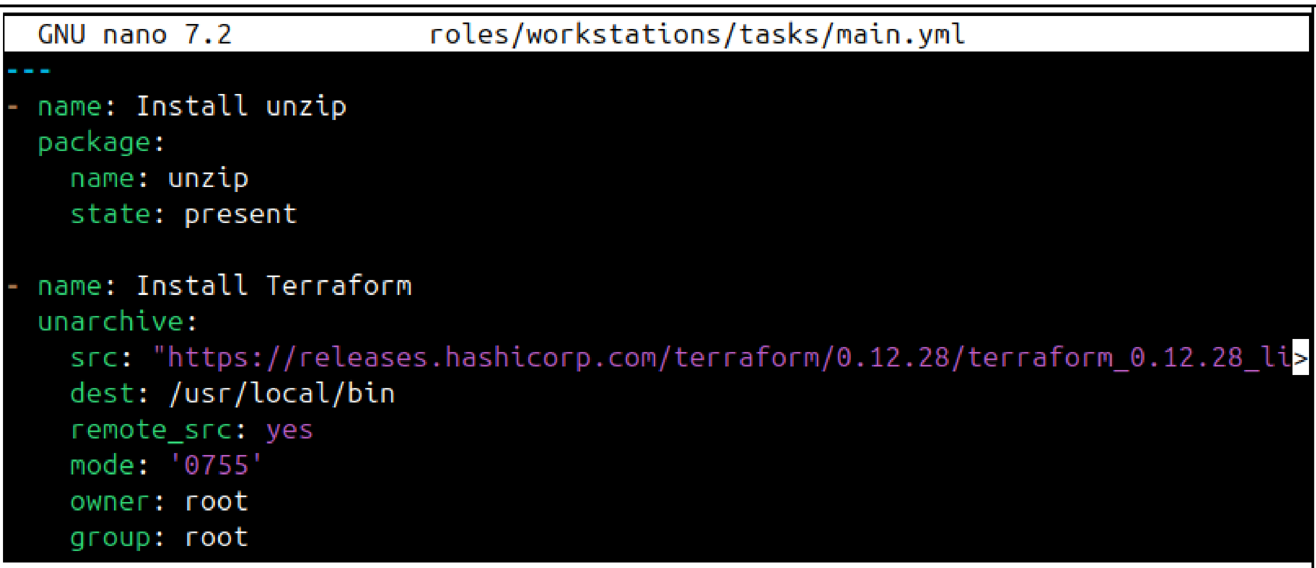
Save the file and exit.

2. Under the same directory, create a new directory and name it roles. Enter the roles directory and create new directories: base, web_servers, file_servers, db_servers and workstations. For each directory, create a directory and name it tasks.

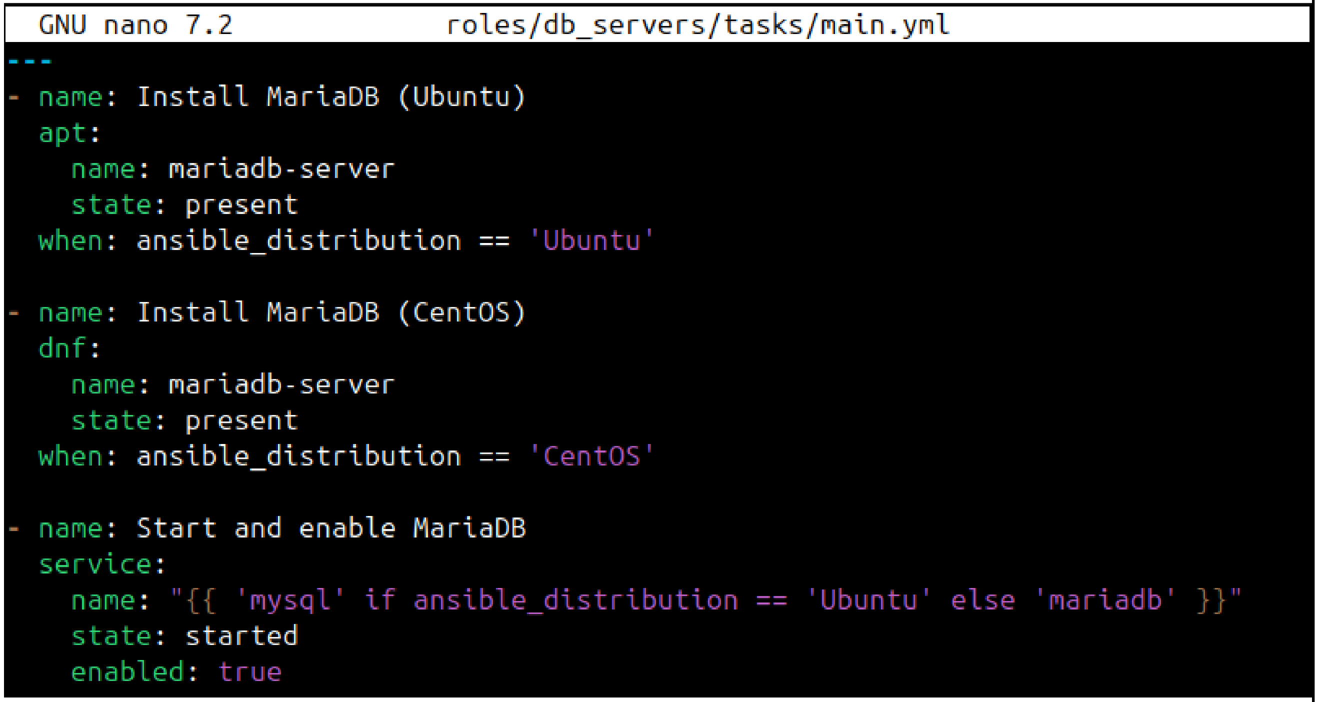
	 

```
  GNU nano 7.2                roles/base/tasks/main.yml
---
- name: Update package cache on Ubuntu
  apt:
    update_cache: yes
  when: ansible_distribution == "Ubuntu"

- name: Update package cache on CentOS
  dnf:
    update_cache: yes
  when: ansible_distribution == "CentOS"
```

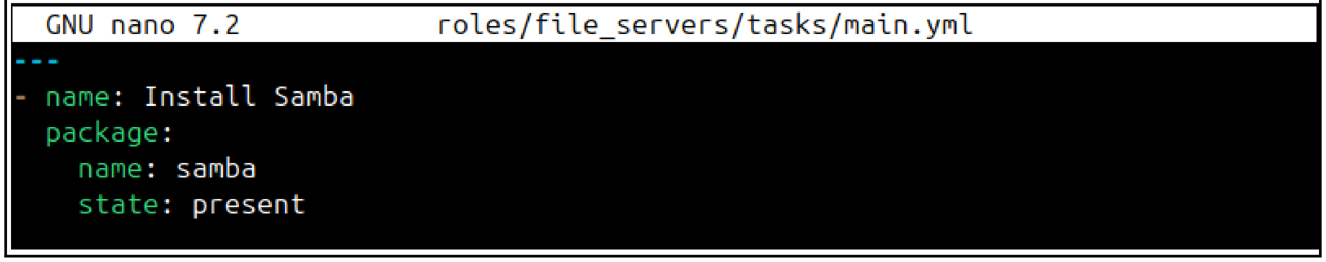
```
  GNU nano 7.2            roles/web_servers/tasks/main.yml
---
- name: Install Apache (Ubuntu or CentOS)
  package:
    name: "{{ 'apache2' if ansible_distribution == 'Ubuntu' else 'httpd' }}"
    state: present

- name: Ensure web root exists
  file:
    path: /var/www/html
    state: directory
    owner: root
    group: root
    mode: '0755'

- name: Copy default site
  copy:
    src: default_site.html
    dest: /var/www/html/index.html
    owner: root
    group: root
    mode: '0644'

- name: Start and enable Apache
  service:
    name: "{{ 'apache2' if ansible_distribution == 'Ubuntu' else 'httpd' }}"
    state: started
    enabled: true
```



```
  GNU nano 7.2                roles/workstations/tasks/main.yml
---
- name: Install unzip
  package:
    name: unzip
    state: present

- name: Install Terraform
  unarchive:
    src: "https://releases.hashicorp.com/terraform/0.12.28/terraform_0.12.28_li>
    dest: /usr/local/bin
    remote_src: yes
    mode: '0755'
    owner: root
    group: root
```

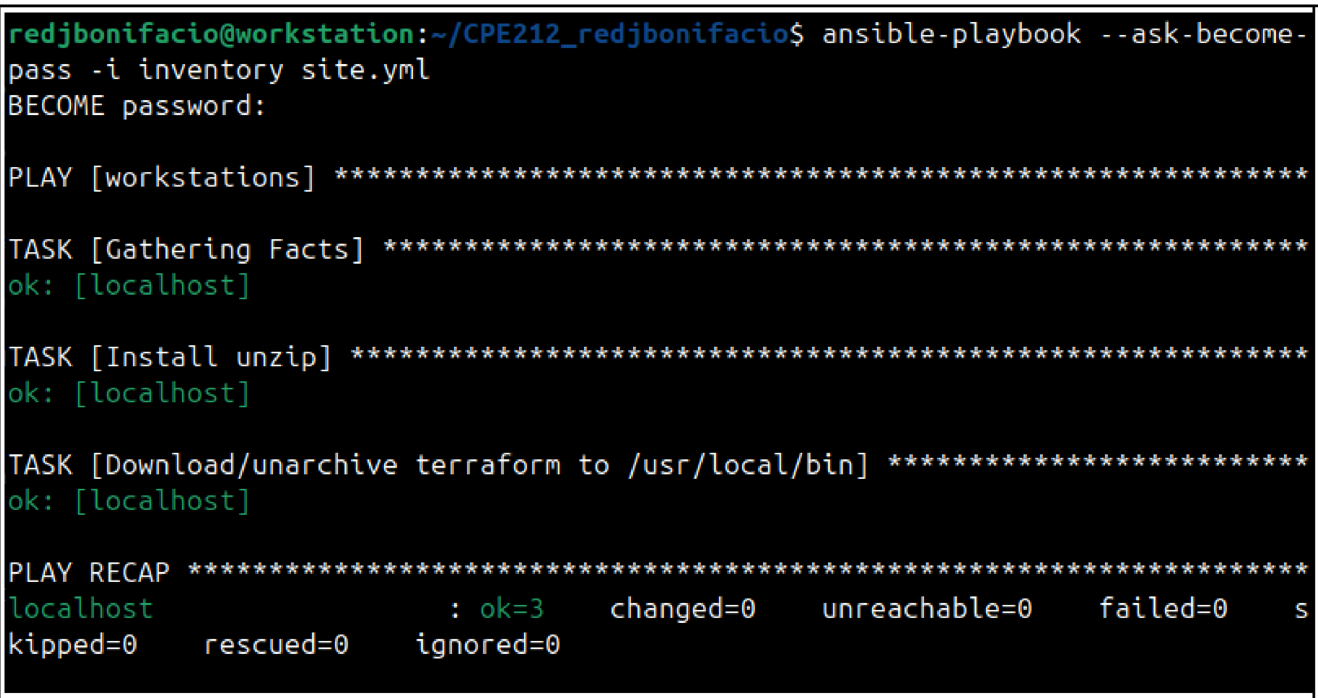


```
  GNU nano 7.2                roles/db_servers/tasks/main.yml
---
- name: Install MariaDB (Ubuntu)
  apt:
    name: mariadb-server
    state: present
  when: ansible_distribution == 'Ubuntu'

- name: Install MariaDB (CentOS)
  dnf:
    name: mariadb-server
    state: present
  when: ansible_distribution == 'CentOS'

- name: Start and enable MariaDB
  service:
    name: "{{ 'mysql' if ansible_distribution == 'Ubuntu' else 'mariadb' }}"
    state: started
    enabled: true
```



```
  GNU nano 7.2                roles/file_servers/tasks/main.yml
---
- name: Install Samba
  package:
    name: samba
    state: present
```

3. Go to tasks for all directory and create a file. Name it main.yml. In each of the tasks for all directories, copy and paste the code from the old site.yml file. Show all contents of main.yml files for all tasks.
4. Run the site.yml playbook and describe the output.



```
redjbonifacio@workstation:~/CPE212_redjbonifacio$ ansible-playbook --ask-become-
pass -i inventory site.yml
BECOME password:

PLAY [workstations] ************************************************************

TASK [Gathering Facts] *********************************************************
ok: [localhost]

TASK [Install unzip] ***********************************************************
ok: [localhost]

TASK [Download/unarchive terraform to /usr/local/bin] **************************
ok: [localhost]

PLAY RECAP *********************************************************************
localhost                  : ok=3    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
```

Running the playbook applied all created roles, automatically installing and configuring packages for the base, web, workstation, database, and file servers without any errors.

Reflections:

Answer the following:

1. What is the importance of creating roles?
   - Creating roles makes playbooks organized, reusable, and easier to manage. It separates tasks by purpose, so updates or changes can be done without affecting other parts. Roles also help automate complex setups across multiple servers efficiently.
2. What is the importance of managing files?
   - Managing files ensures that all servers have the correct and updated configurations or content. It saves time by automatically deploying files instead of doing it manually. This helps maintain consistency and reliability across different systems.