Name: Mamaril, Justin Kenneth I.	**Date Performed: 09/**
Course/Section: CPE 212-CPE31S2	**Date Submitted: 09/**
Instructor: Engr. Robin Valenzuela	**Semester and SY: 1st 2025-2026**

Activity 7: Managing Files and Creating Roles in Ansible

1. Objectives:

1.1 Manage files in remote servers

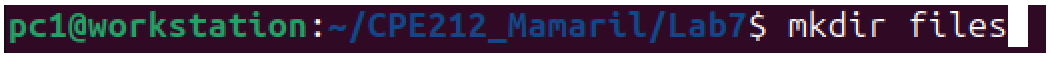
1.2 Implement roles in ansible

2. Discussion:

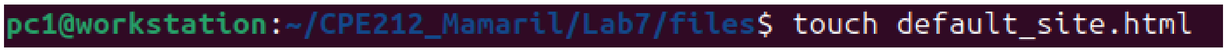
In this activity, we look at the concept of copying a file to a server. We are going to create a file into our git repository and use Ansible to grab that file and put it into a particular place so that we could do things like customize a default website, or maybe install a default configuration file.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

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

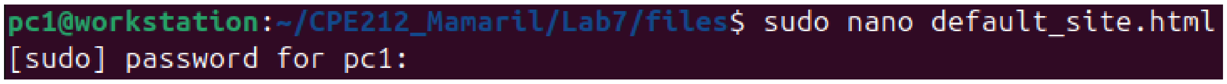
1. Using the previous directory we created, create a directory, and named it "*files*." Create a file inside that directory and name it "*default_site.html*." Edit the file and put basic HTML syntax. Any content will do, as long as it will display text later. Save the file and exit.



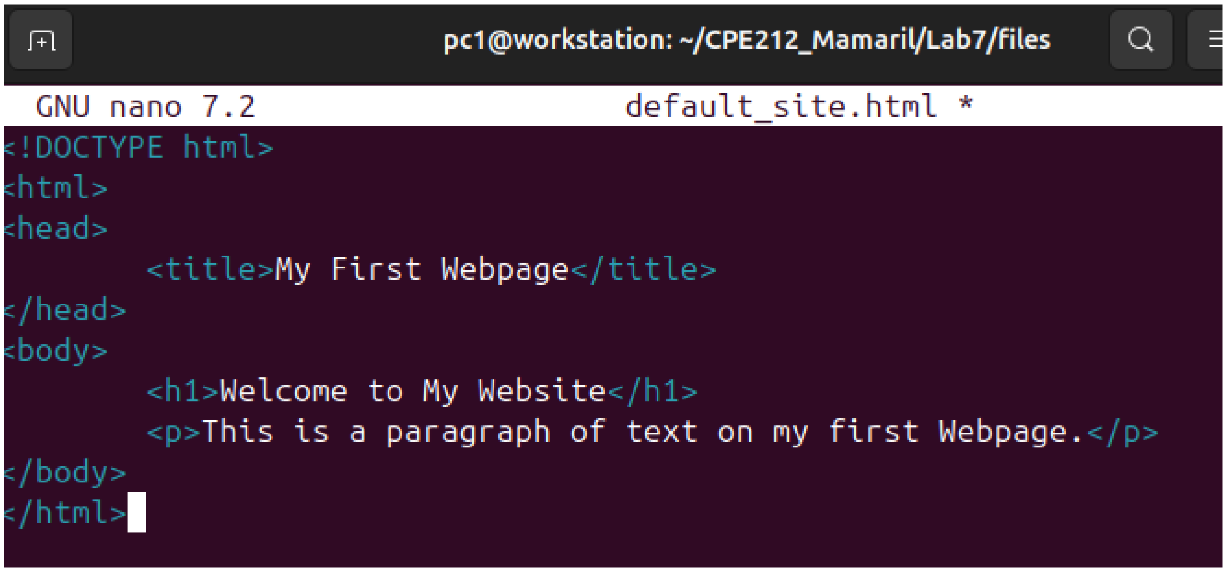
```
pc1@workstation:~/CPE212_Mamaril/Lab7$ mkdir files
```



```
pc1@workstation:~/CPE212_Mamaril/Lab7/files$ touch default_site.html
```



```
pc1@workstation:~/CPE212_Mamaril/Lab7/files$ sudo nano default_site.html
[sudo] password for pc1:
```

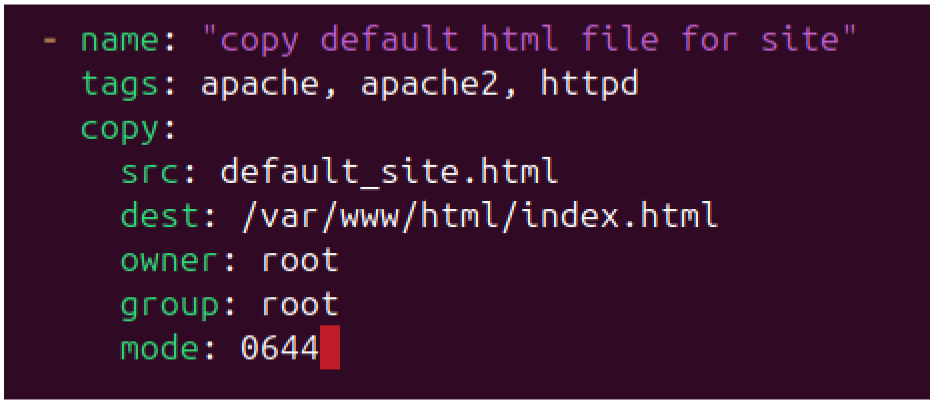


```
pc1@workstation: ~/CPE212_Mamaril/Lab7/files

  GNU nano 7.2                    default_site.html *
<!DOCTYPE html>
<html>
<head>
        <title>My First Webpage</title>
</head>
<body>
        <h1>Welcome to My Website</h1>
        <p>This is a paragraph of text on my first Webpage.</p>
</body>
</html>
```

- I made a directory file using mkdir and then made a file by using the touch command. And then used nano to change the contents of default_site.html. Then proceed to make a simple html code.

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:

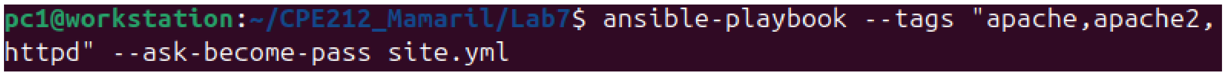
```
- name: copy default html file for site

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



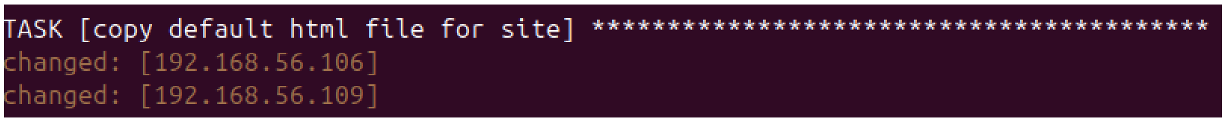
```
- name: "copy default html file for site"
  tags: apache, apache2, httpd
  copy:
    src: default_site.html
    dest: /var/www/html/index.html
    owner: root
    group: root
    mode: 0644
```

- I added below the web_servers play in my site.yml

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



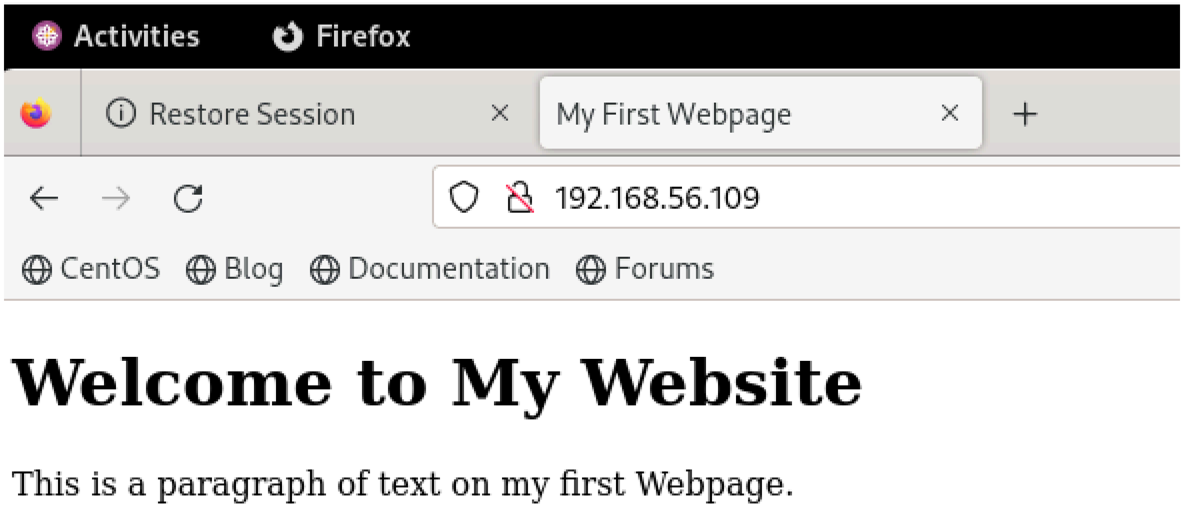
```
pc1@workstation:~/CPE212_Mamaril/Lab7$ ansible-playbook --tags "apache,apache2,
httpd" --ask-become-pass site.yml
```



```
TASK [copy default html file for site] ******************************************
changed: [192.168.56.106]
changed: [192.168.56.109]
```

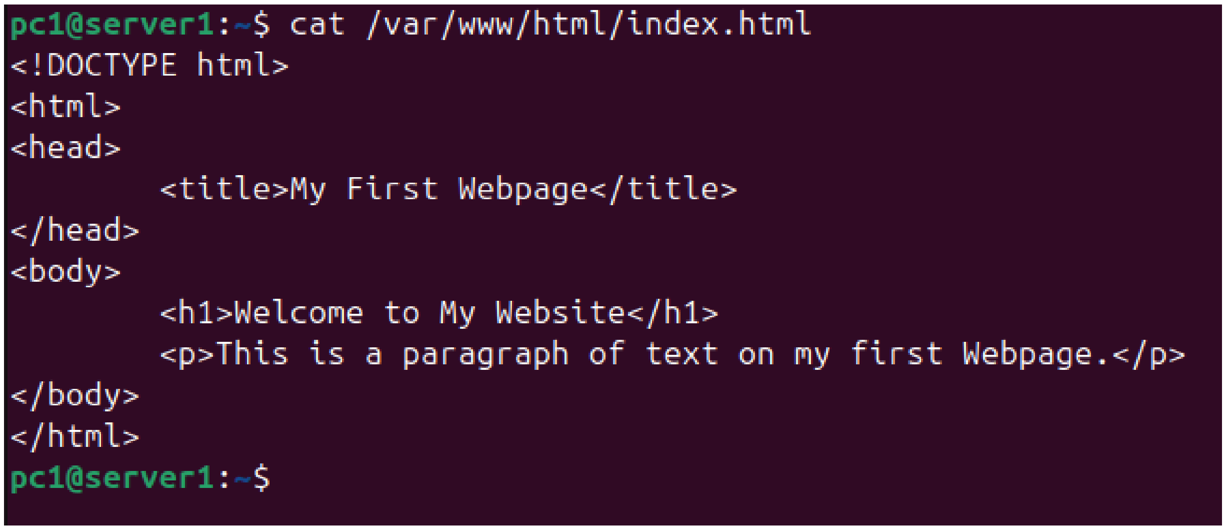
- It runned in both of my remote servers

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.



Welcome to My Website

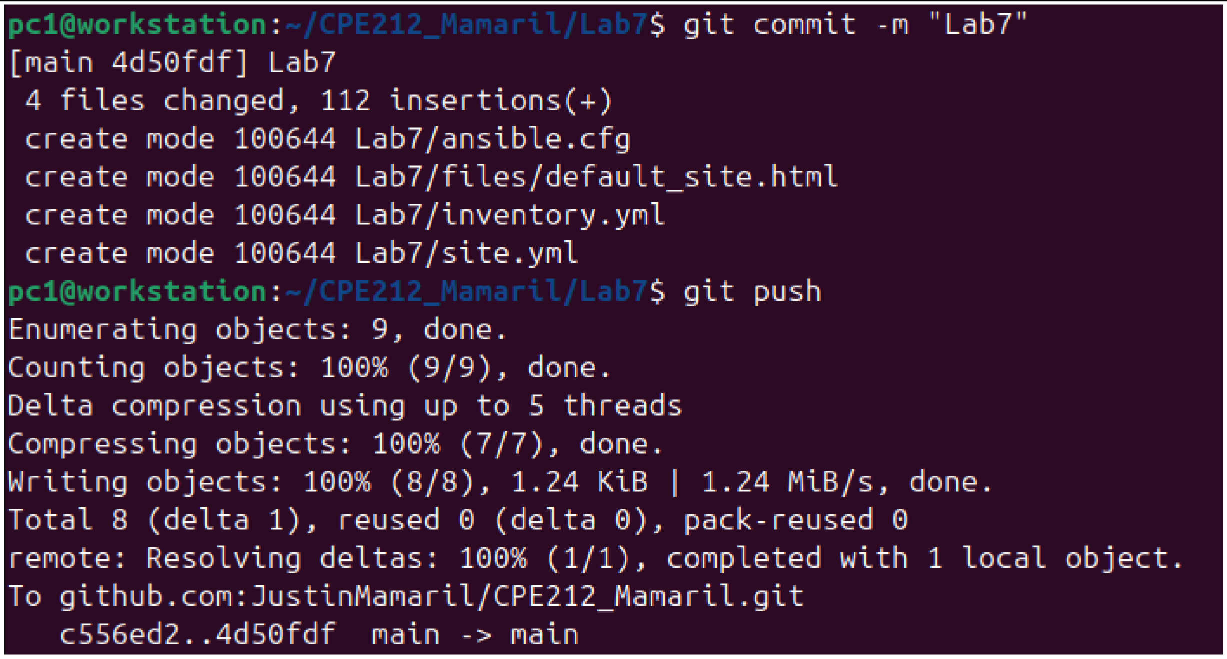
This is a paragraph of text on my first Webpage.



```
pc1@server1:~$ cat /var/www/html/index.html
<!DOCTYPE html>
<html>
<head>
        <title>My First Webpage</title>
</head>
<body>
        <h1>Welcome to My Website</h1>
        <p>This is a paragraph of text on my first Webpage.</p>
</body>
</html>
pc1@server1:~$
```

- It successfully runs on my CentOs server by typing the ip address at the website. And when i cat the dest in my ubuntu server, it shows the html file contents too thats h

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

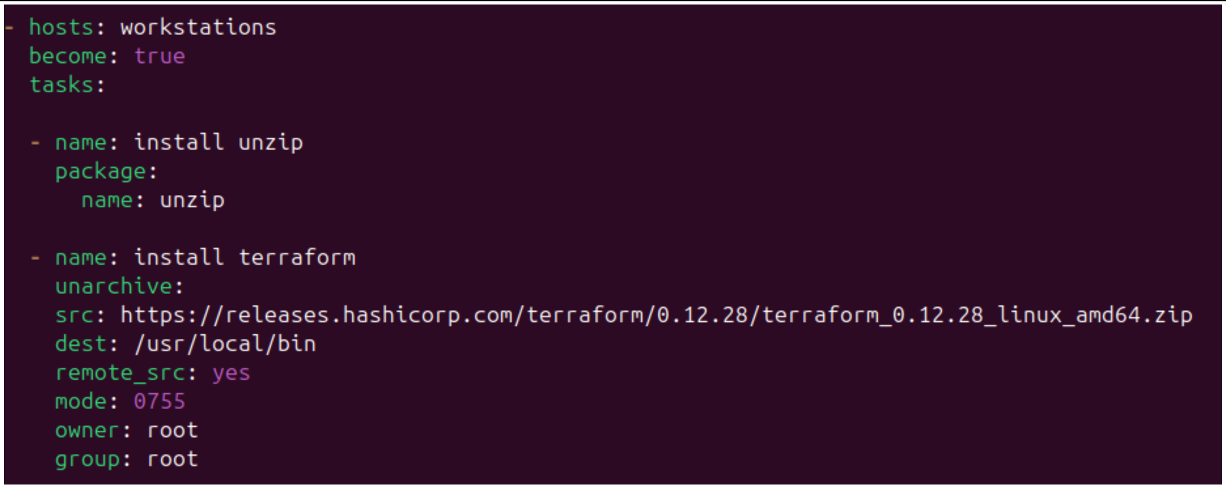


```
pc1@workstation:~/CPE212_Mamaril/Lab7$ git commit -m "Lab7"
[main 4d50fdf] Lab7
 4 files changed, 112 insertions(+)
 create mode 100644 Lab7/ansible.cfg
 create mode 100644 Lab7/files/default_site.html
 create mode 100644 Lab7/inventory.yml
 create mode 100644 Lab7/site.yml
pc1@workstation:~/CPE212_Mamaril/Lab7$ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 5 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 1.24 KiB | 1.24 MiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:JustinMamaril/CPE212_Mamaril.git
   c556ed2..4d50fdf  main -> main
```

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:

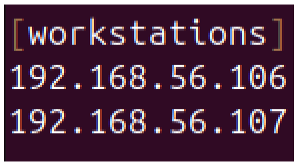
```
- hosts: workstations
  become: true
  tasks:

  - 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
```



```
- 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
```

- I edited the site.yml file and created a new play with host workstations.

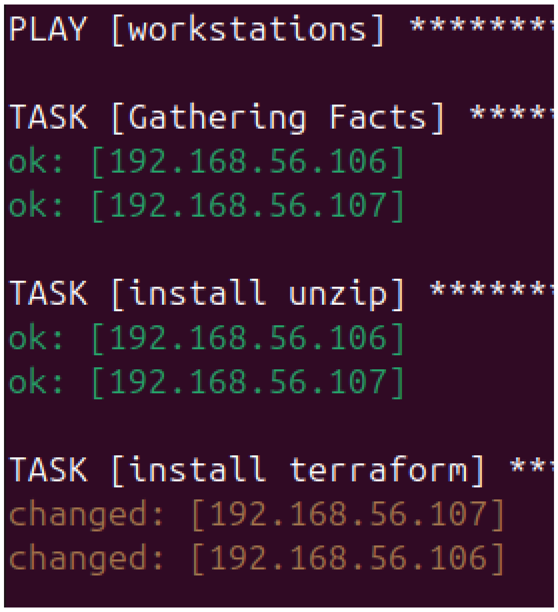
2. Edit the inventory file and add workstations group. Add any Ubuntu remote server. Make sure to remember the IP address.



```
[workstations]
192.168.56.106
192.168.56.107
```

- i added 2 of my remote servers which is server1 and server2

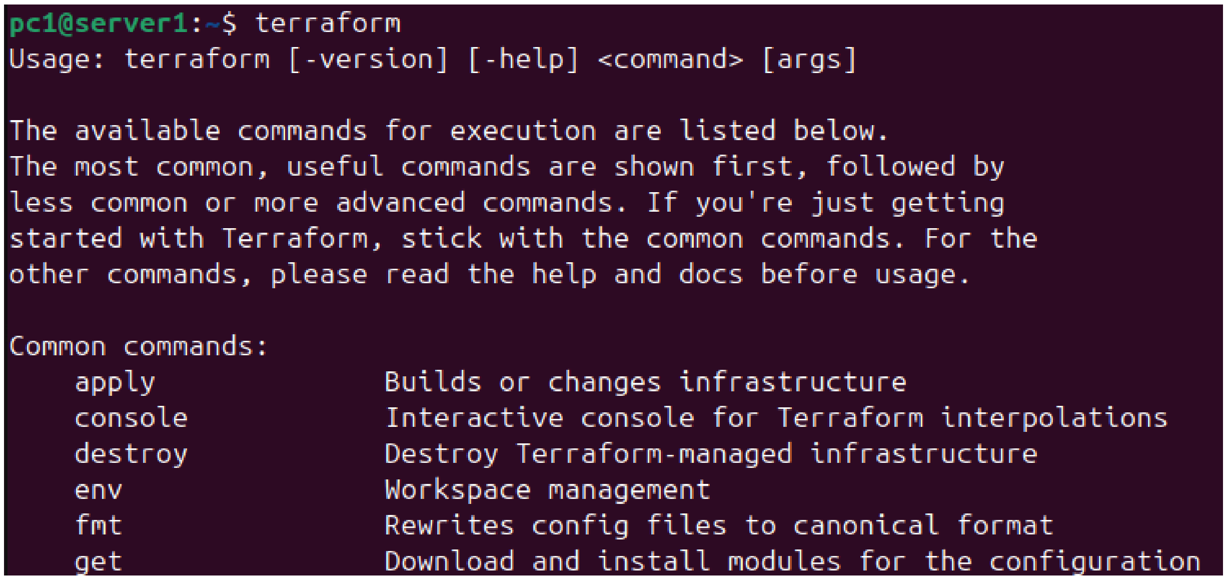
3. Run the playbook. Describe the output.



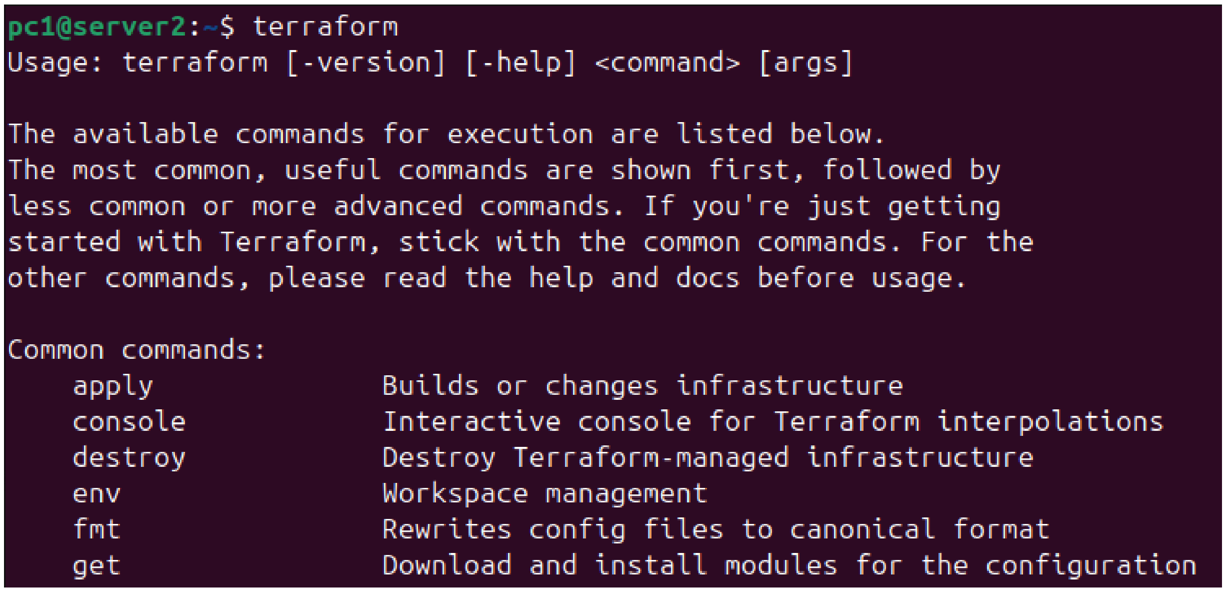
```
PLAY [workstations] ********

TASK [Gathering Facts] *****
ok: [192.168.56.106]
ok: [192.168.56.107]

TASK [install unzip] *******
ok: [192.168.56.106]
ok: [192.168.56.107]

TASK [install terraform] **
changed: [192.168.56.107]
changed: [192.168.56.106]
```

- It successfully installed the unzip and the terraform. as it says it changed in both of the ip addresses in the workstation.

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



```
pc1@server1:~$ 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
```

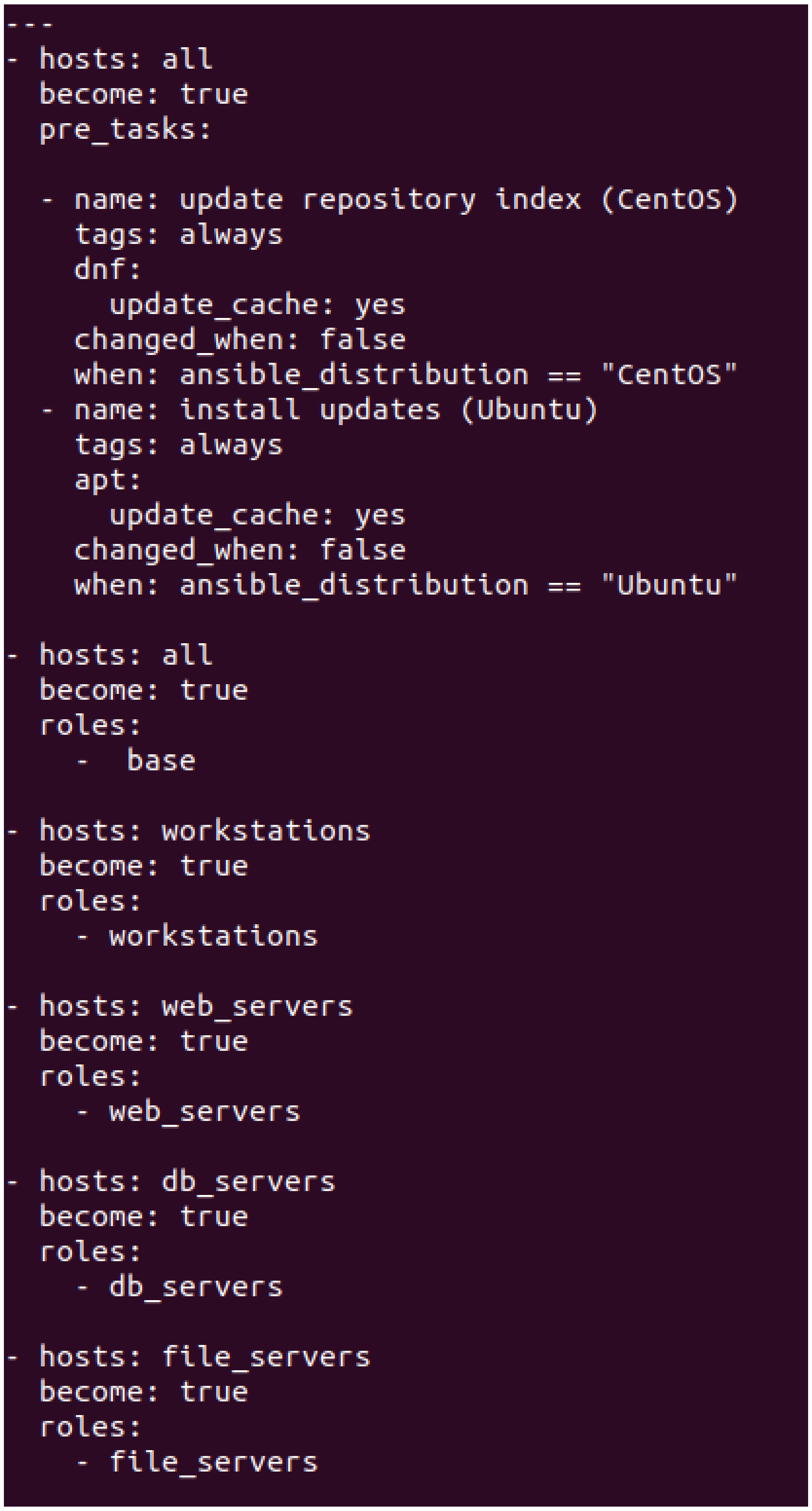


```
pc1@server2:~$ terraform
Usage: terraform [-version] [-help] <command> [args]

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
```

- it successfully installed on both of my remote servers.

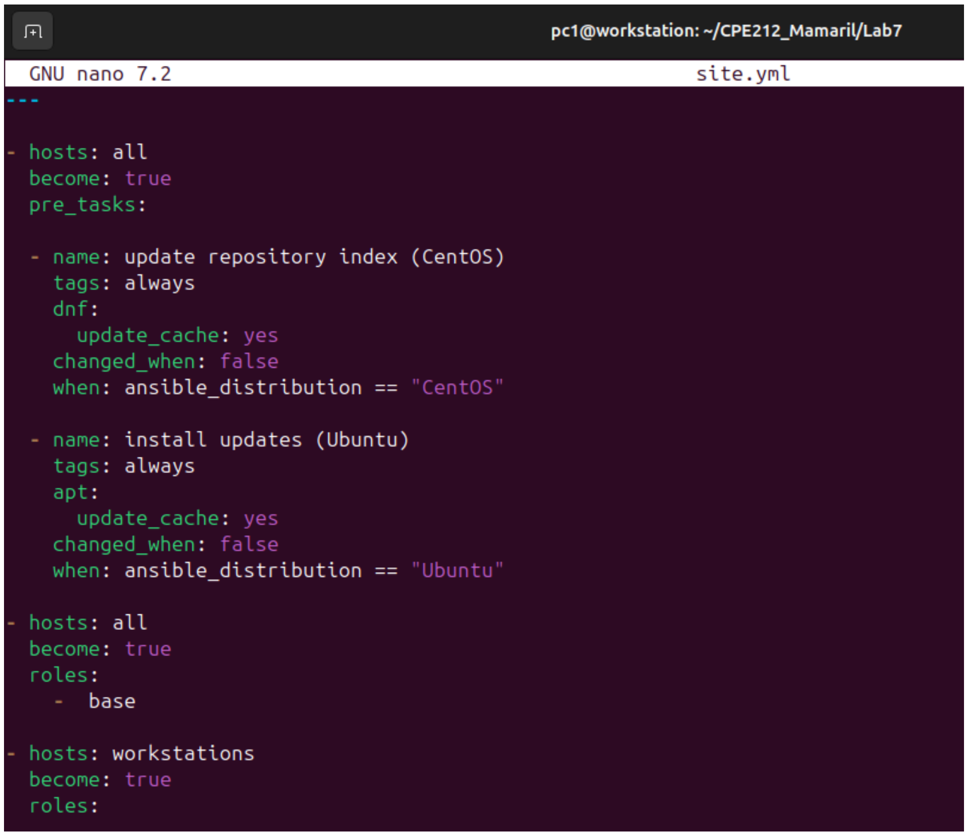
Task 3: Create roles

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)

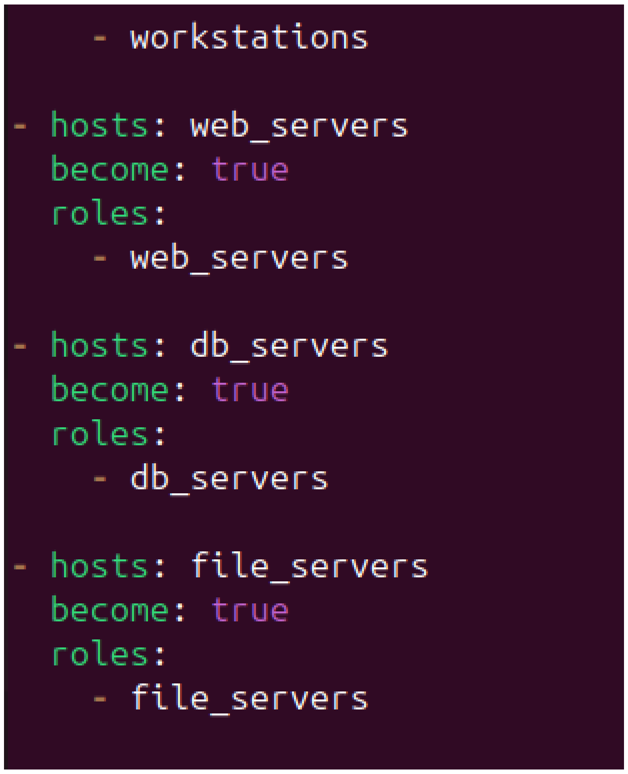


```
---
- hosts: all
  become: true
  pre_tasks:

  - 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.

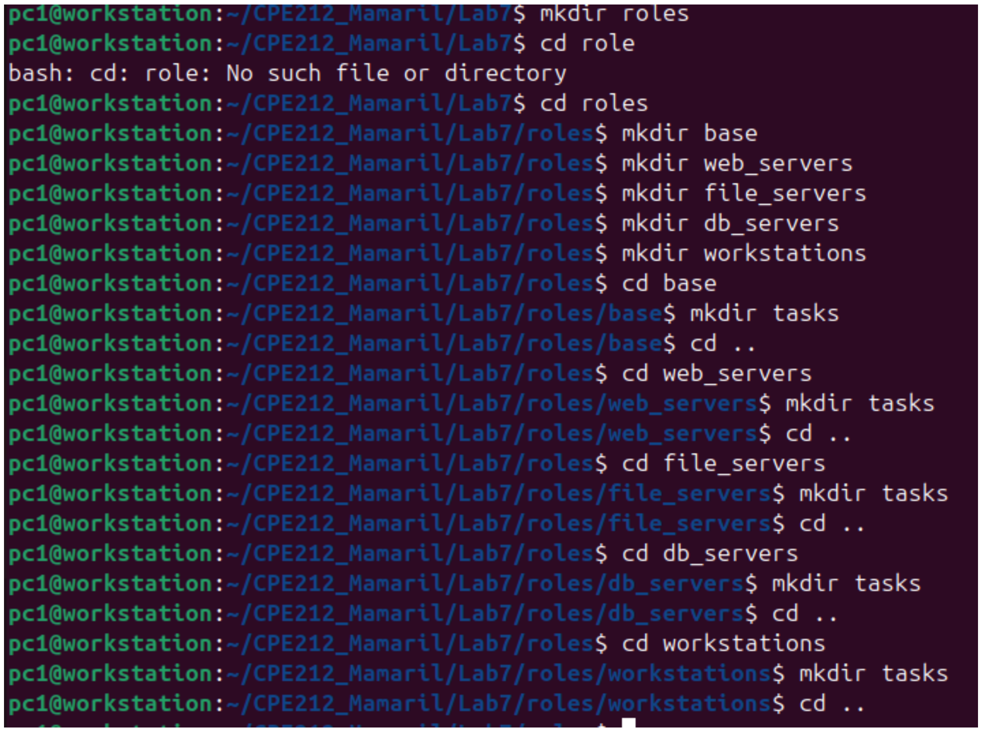


```
pc1@workstation: ~/CPE212_Mamaril/Lab7
GNU nano 7.2                                            site.yml
---

- 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
```

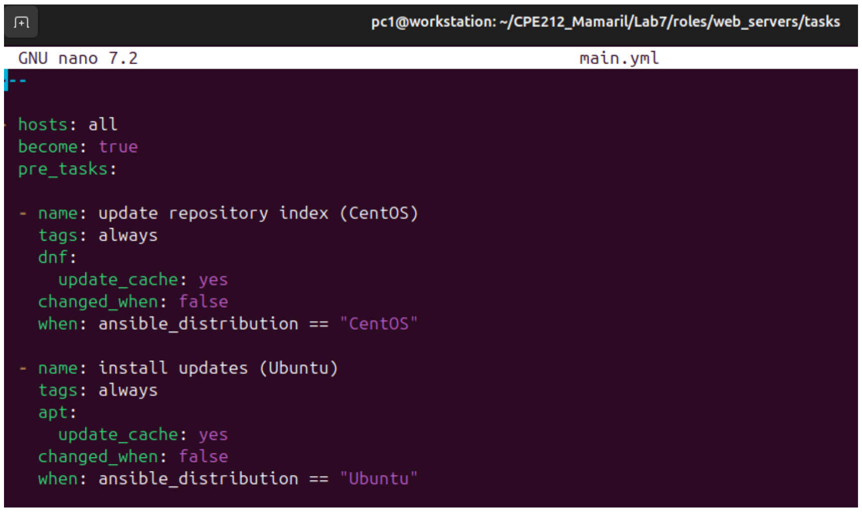
2. Under the same directory, create a new directory and name it roles. Enter the roles directory and create new directories: base, web_servers, file_servers, db_servers and workstations. For each directory, create a directory and name it tasks.



```
pc1@workstation:~/CPE212_Mamaril/Lab7$ mkdir roles
pc1@workstation:~/CPE212_Mamaril/Lab7$ cd role
bash: cd: role: No such file or directory
pc1@workstation:~/CPE212_Mamaril/Lab7$ cd roles
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ mkdir base
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ mkdir web_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ mkdir file_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ mkdir db_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ mkdir workstations
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd base
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/base$ mkdir tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/base$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd web_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers$ mkdir tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd file_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/file_servers$ mkdir tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/file_servers$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd db_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/db_servers$ mkdir tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/db_servers$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd workstations
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/workstations$ mkdir tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/workstations$ cd ..
```

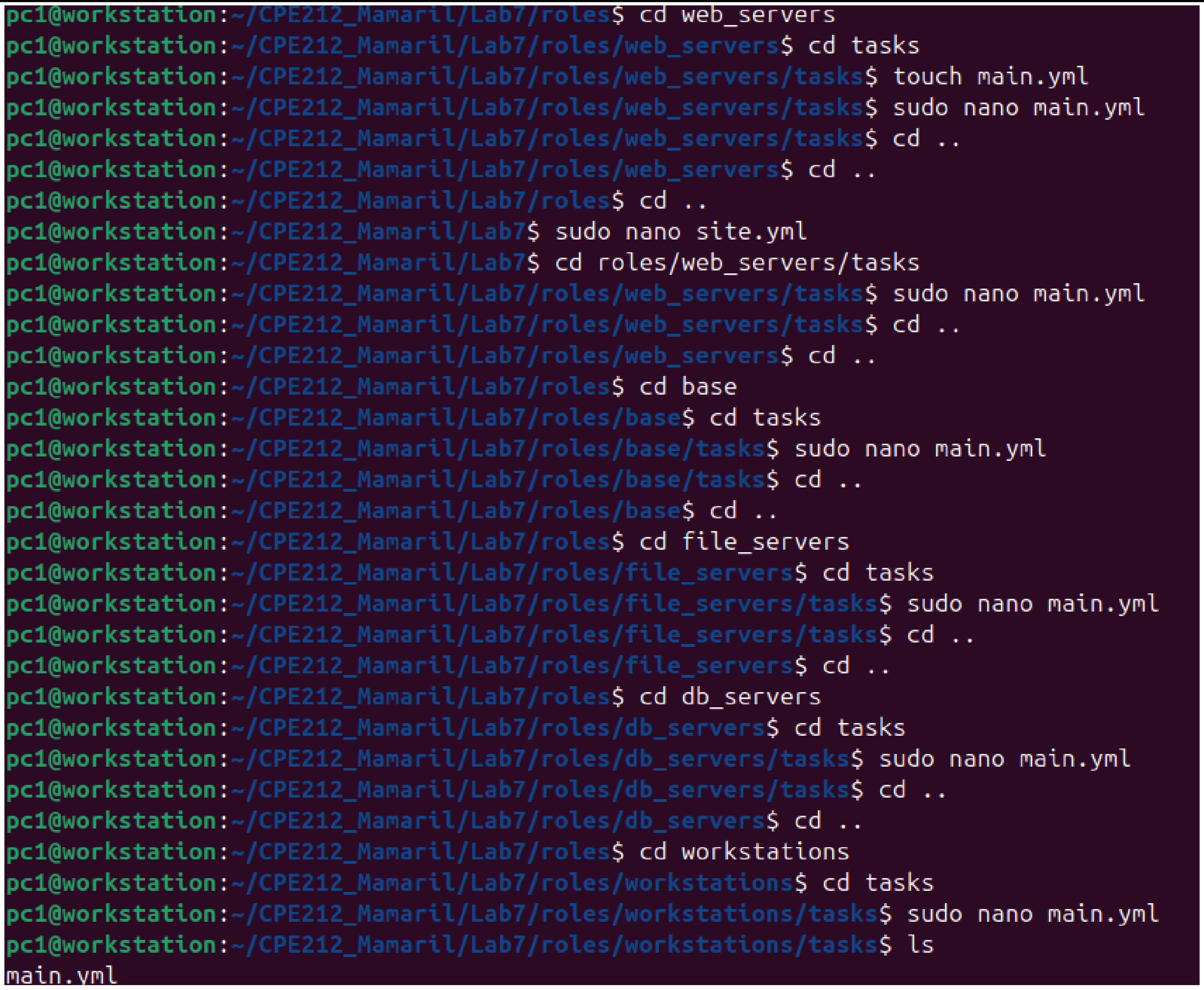
- I created a directory named roles and then i made 5 directories inside roles directory and then in each directory inside the roles, i made another directory named tasks.

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.



pc1@workstation: ~/CPE212_Mamaril/Lab7/roles/web_servers/tasks

```
GNU nano 7.2                                    main.yml
---

- 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"
```



```
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd web_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers$ cd tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers/tasks$ touch main.yml
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers/tasks$ sudo nano main.yml
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers/tasks$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7$ sudo nano site.yml
pc1@workstation:~/CPE212_Mamaril/Lab7$ cd roles/web_servers/tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers/tasks$ sudo nano main.yml
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers/tasks$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/web_servers$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd base
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/base$ cd tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/base/tasks$ sudo nano main.yml
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/base/tasks$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/base$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd file_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/file_servers$ cd tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/file_servers/tasks$ sudo nano main.yml
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/file_servers/tasks$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/file_servers$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd db_servers
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/db_servers$ cd tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/db_servers/tasks$ sudo nano main.yml
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/db_servers/tasks$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/db_servers$ cd ..
pc1@workstation:~/CPE212_Mamaril/Lab7/roles$ cd workstations
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/workstations$ cd tasks
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/workstations/tasks$ sudo nano main.yml
pc1@workstation:~/CPE212_Mamaril/Lab7/roles/workstations/tasks$ ls
main.yml
```

4. Run the site.yml playbook and describe the output.

Reflections:

Answer the following:

1. What is the importance of creating roles? - Creating roles in Ansible is important because it brings modularity and organization to your automation code. Roles break down complex playbooks into smaller, focused components that are easier to manage, maintain, and reuse across different projects. This structure helps teams collaborate more effectively by providing a standardized way to organize tasks, handlers, files, and variables. Additionally, roles support scalability, allowing you to expand or modify your automation without disrupting existing configurations.

2. What is the importance of managing files? - Managing files is crucial for ensuring consistency and reliability across your infrastructure. By automating the

deployment of configuration files, scripts, or website content, you reduce manual errors and speed up the setup process. Proper file management also enables customization of default configurations to suit specific needs, while version control allows you to track changes, audit modifications, and roll back if necessary. Overall, managing files efficiently helps maintain the desired state of systems and supports repeatable, dependable deployments.