| | |
|---|---|
| **Name: Planta, Calvin Earl L.** | **Date Performed: 9/19/25** |
| **Course/Section: CPE 212 - CPE31S2** | **Date Submitted: 10/3/25** |
| **Instructor: Engr. Robin Valenzuela** | **Semester and SY: 1st Sem SY 25-26** |

# Activity 7: Managing Files and Creating Roles in Ansible

1. **Objectives:**

1.1 Manage files in remote servers

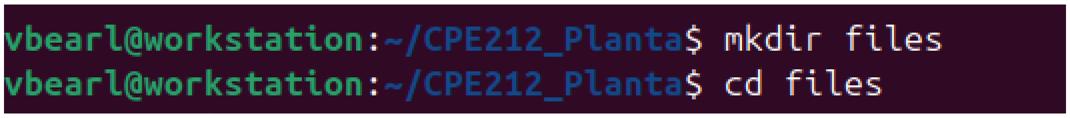
1.2 Implement roles in ansible

2. **Discussion**:

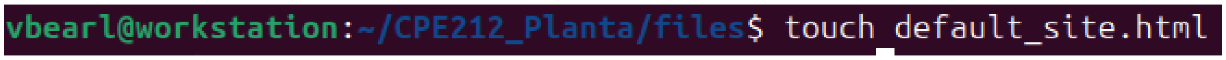
In this activity, we look at the concept of copying a file to a server. We are going to create a file into our git repository and use Ansible to grab that file and put it into a particular place so that we could do things like customize a default website, or maybe install a default configuration file.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

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

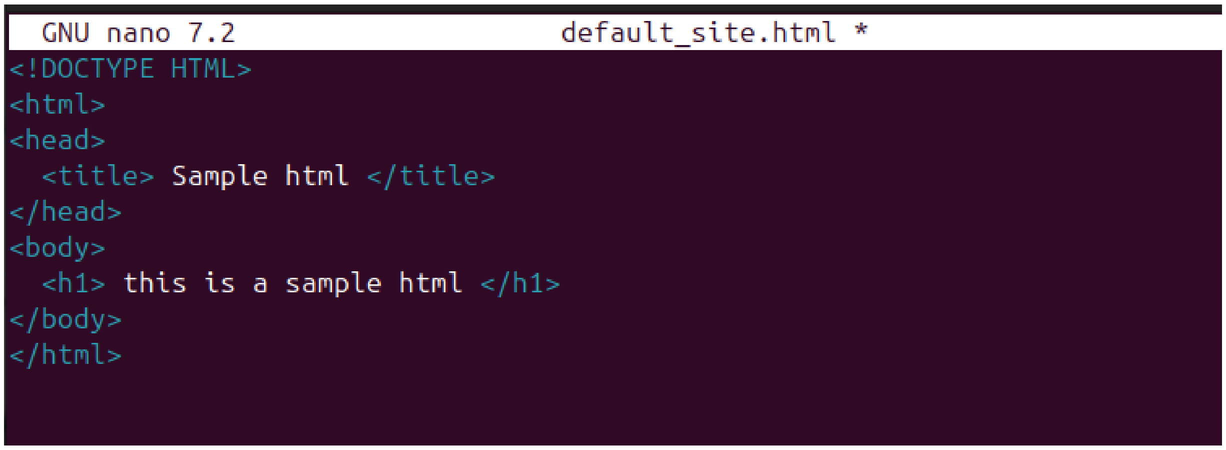
1. Using the previous directory we created, create a directory, and name it "*files*." Create a file inside that directory and name it "*default_site.html*." Edit the file and put basic HTML syntax. Any content will do, as long as it will display text later. Save the file and exit.



```
vbearl@workstation:~/CPE212_Planta$ mkdir files
vbearl@workstation:~/CPE212_Planta$ cd files
```



```
vbearl@workstation:~/CPE212_Planta/files$ touch default_site.html
```



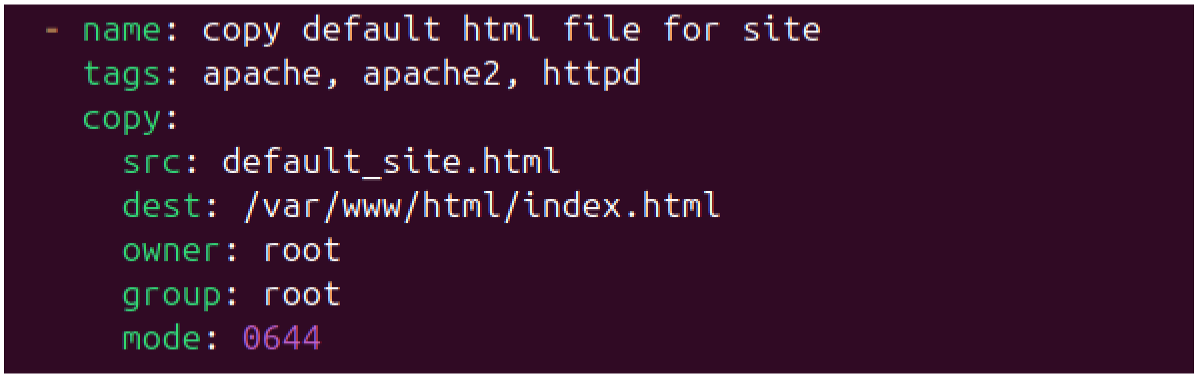
```
  GNU nano 7.2                          default_site.html *
<!DOCTYPE HTML>
<html>
<head>
  <title> Sample html </title>
</head>
<body>
  <h1> this is a sample html </h1>
</body>
</html>
```

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

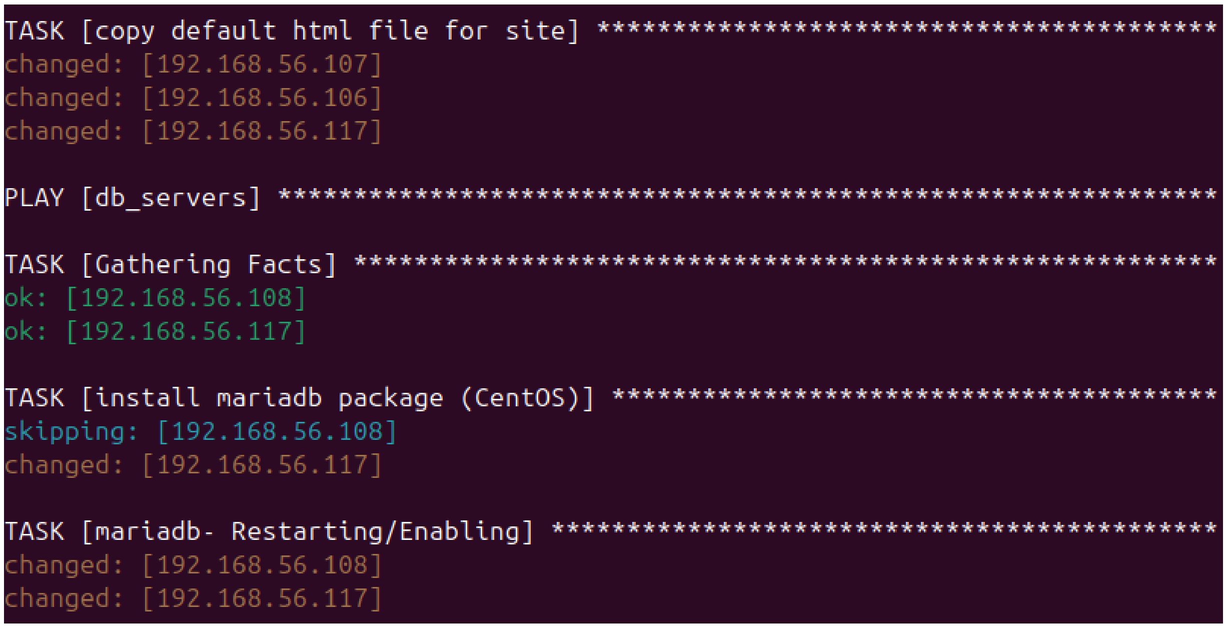
     tags: apache, apache2, httpd

```
copy:
  src: default_site.html
  dest: /var/www/html/index.html
  owner: root
  group: root
  mode: 0644
```

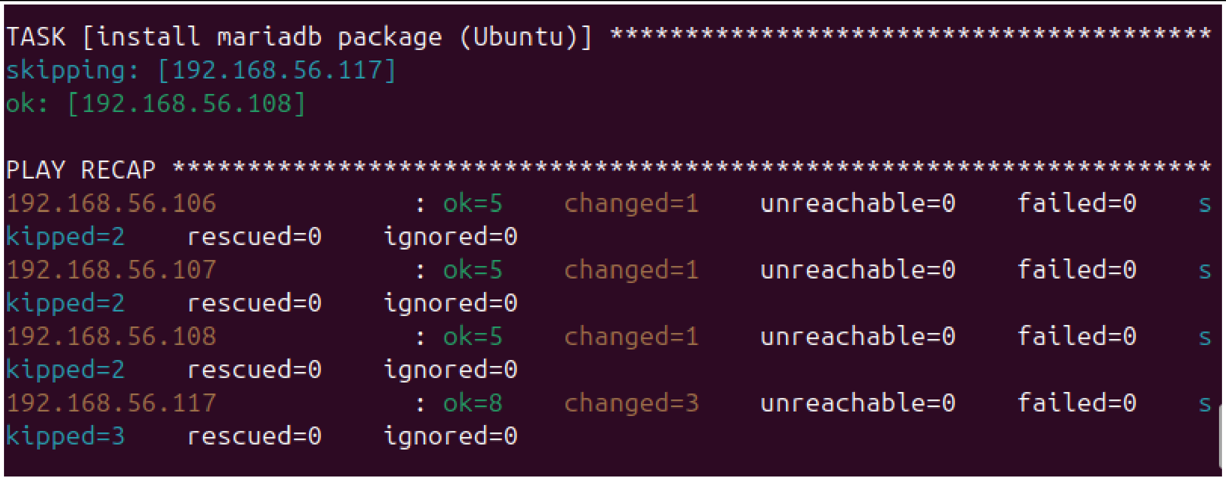


```
- name: copy default html file for site
  tags: apache, apache2, httpd
  copy:
    src: default_site.html
    dest: /var/www/html/index.html
    owner: root
    group: root
    mode: 0644
```

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



```
TASK [copy default html file for site] *****************************************
changed: [192.168.56.107]
changed: [192.168.56.106]
changed: [192.168.56.117]

PLAY [db_servers] **************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]
ok: [192.168.56.117]

TASK [install mariadb package (CentOS)] ****************************************
skipping: [192.168.56.108]
changed: [192.168.56.117]

TASK [mariadb- Restarting/Enabling] ********************************************
changed: [192.168.56.108]
changed: [192.168.56.117]
```

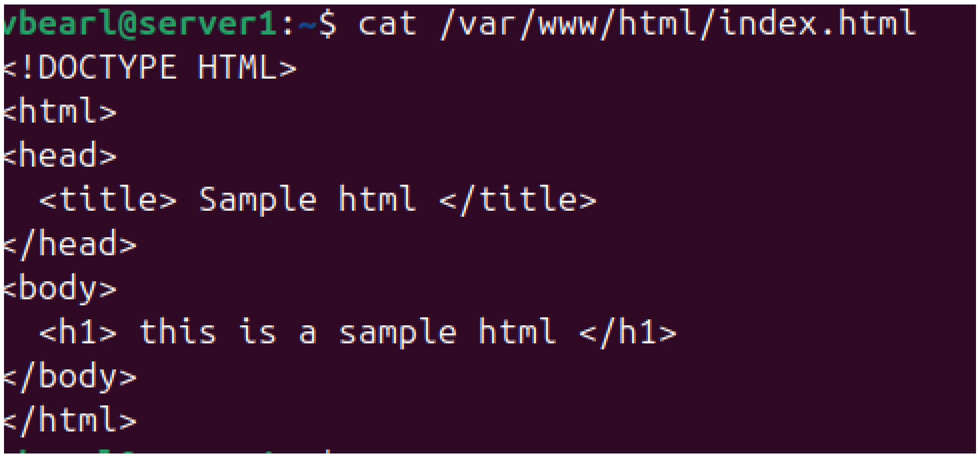


```
TASK [install mariadb package (Ubuntu)] ****************************************
skipping: [192.168.56.117]
ok: [192.168.56.108]

PLAY RECAP *********************************************************************
192.168.56.106             : ok=5    changed=1    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.56.107             : ok=5    changed=1    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.56.108             : ok=5    changed=1    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.56.117             : ok=8    changed=3    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
```

- The play copied the html file that I created into the remote servers, in the /var/www/html/index.html path.

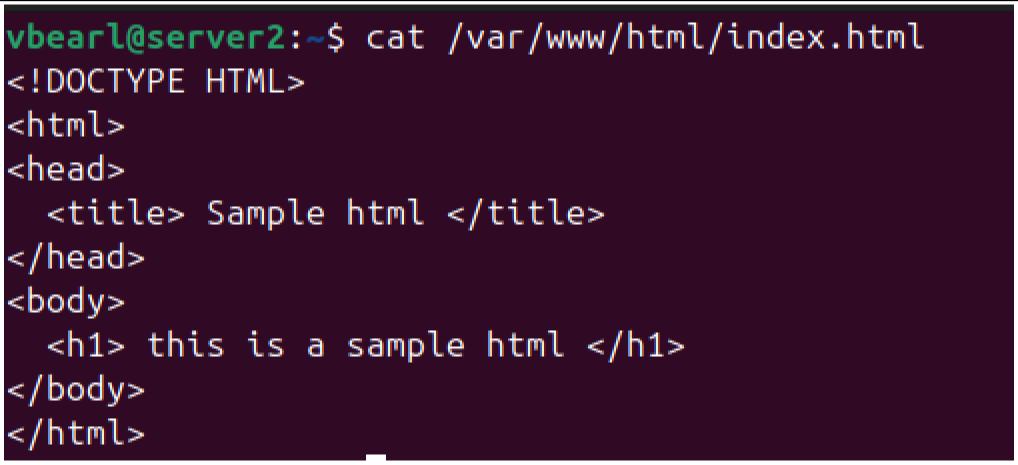
4. Go to the remote servers (*web_servers*) listed in your inventory. Use cat command to check if the index.html is the same as the local repository file (*default_site.html*). Do both for Ubuntu and CentOS servers. On the CentOS server, go to the browser and type its IP address. Describe the output.

**Server 1**



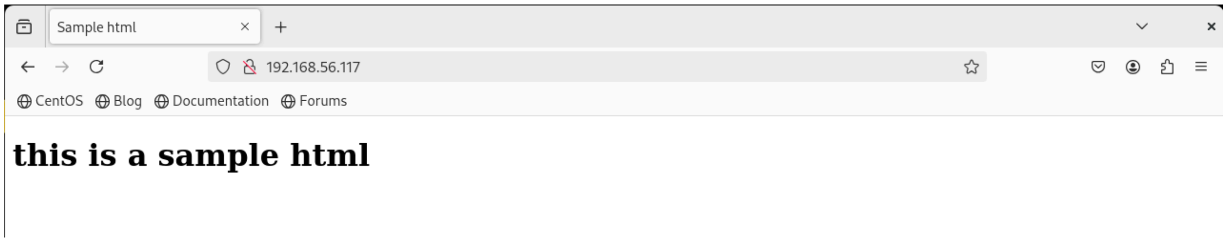
```
vbearl@server1:~$ cat /var/www/html/index.html
<!DOCTYPE HTML>
<html>
<head>
  <title> Sample html </title>
</head>
<body>
  <h1> this is a sample html </h1>
</body>
</html>
```

**Server 2**



```
vbearl@server2:~$ cat /var/www/html/index.html
<!DOCTYPE HTML>
<html>
<head>
  <title> Sample html </title>
</head>
<body>
  <h1> this is a sample html </h1>
</body>
</html>
```

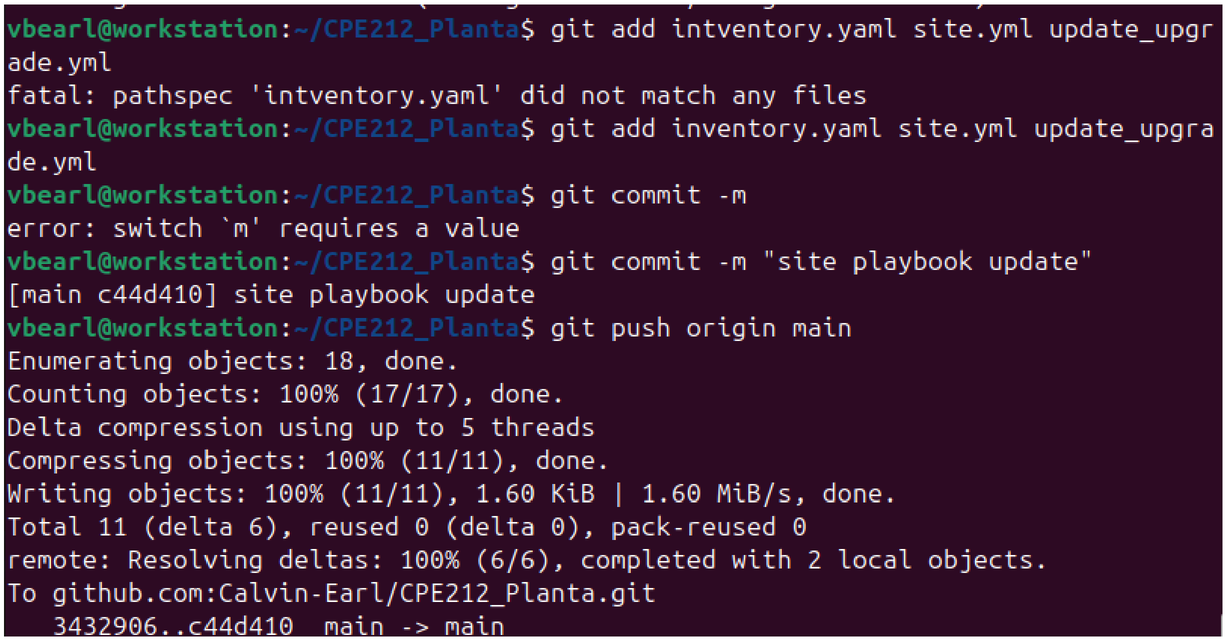
**CentOS Server**



this is a sample html

- Upon typing the server's IP address, it displayed the contents of the default_site.html that I had previously created in my local machine.

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



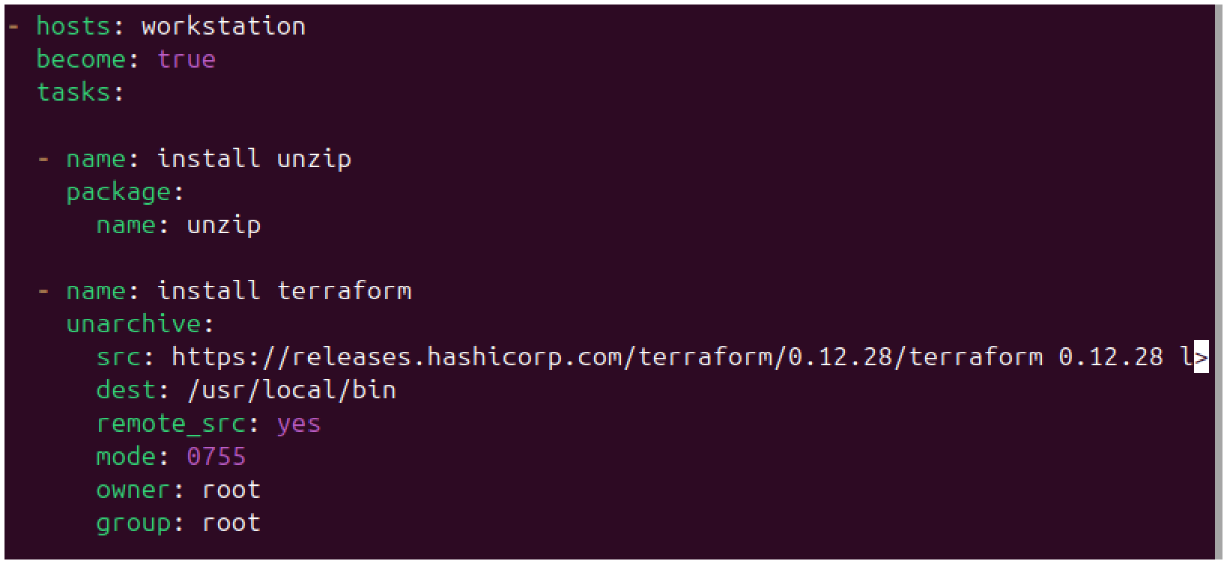
```
vbearl@workstation:~/CPE212_Planta$ git add intventory.yaml site.yml update_upgr
ade.yml
fatal: pathspec 'intventory.yaml' did not match any files
vbearl@workstation:~/CPE212_Planta$ git add inventory.yaml site.yml update_upgra
de.yml
vbearl@workstation:~/CPE212_Planta$ git commit -m
error: switch `m' requires a value
vbearl@workstation:~/CPE212_Planta$ git commit -m "site playbook update"
[main c44d410] site playbook update
vbearl@workstation:~/CPE212_Planta$ git push origin main
Enumerating objects: 18, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 5 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 1.60 KiB | 1.60 MiB/s, done.
Total 11 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (6/6), completed with 2 local objects.
To github.com:Calvin-Earl/CPE212_Planta.git
   3432906..c44d410  main -> main
```

- The changes I made to site.yml was simply adding an additional task that copies a file (default_site.html) into a certain destination (/var/www/html/index.html). This also changes the default http page of CentOS into the html file that was created and copied.

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

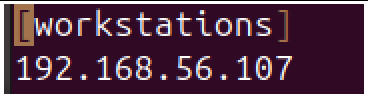
1. Edit the site.yml. Just before the web_servers play, create a new play:

```
- 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
```



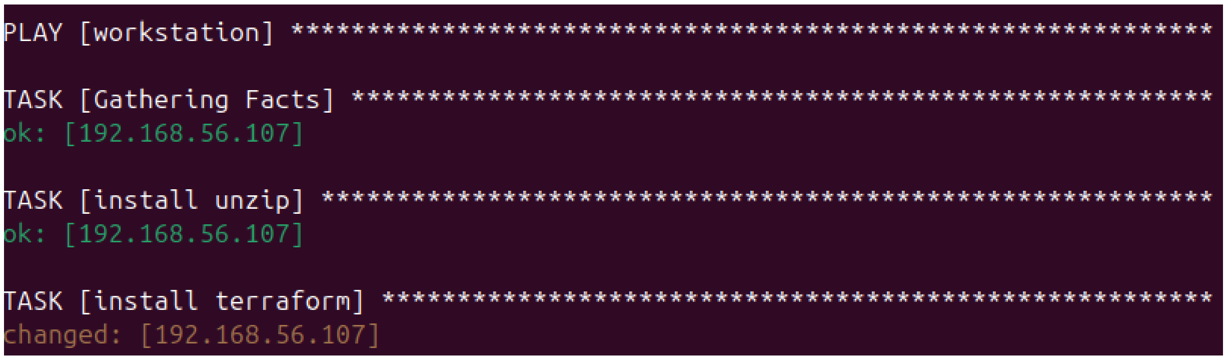
```
- hosts: workstation
  become: true
  tasks:

  - name: install unzip
    package:
      name: unzip

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

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



```
[workstations]
192.168.56.107
```

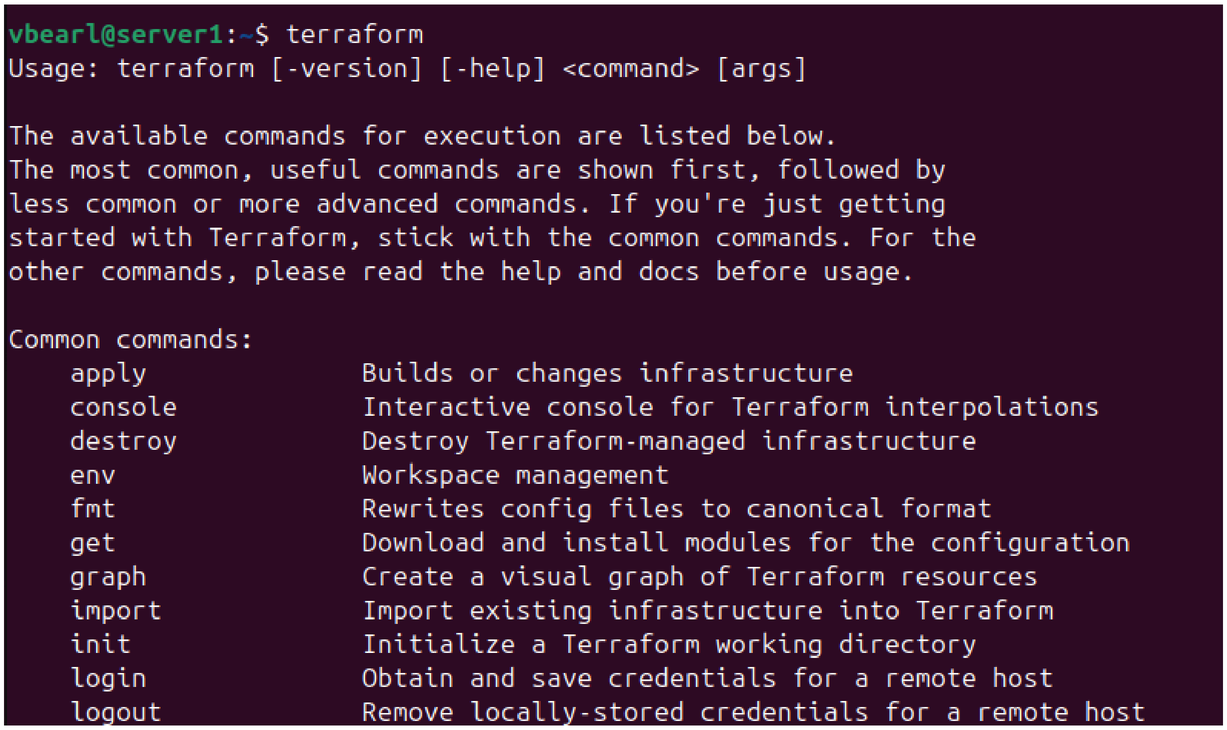
3. Run the playbook. Describe the output.



```
PLAY [workstation] *************************************************************

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

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

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

- The play installs the unzip package, which allows file extractions, then extracts the specified file (terraform) into /usr/local/bin from the targeted ubuntu remote server.

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

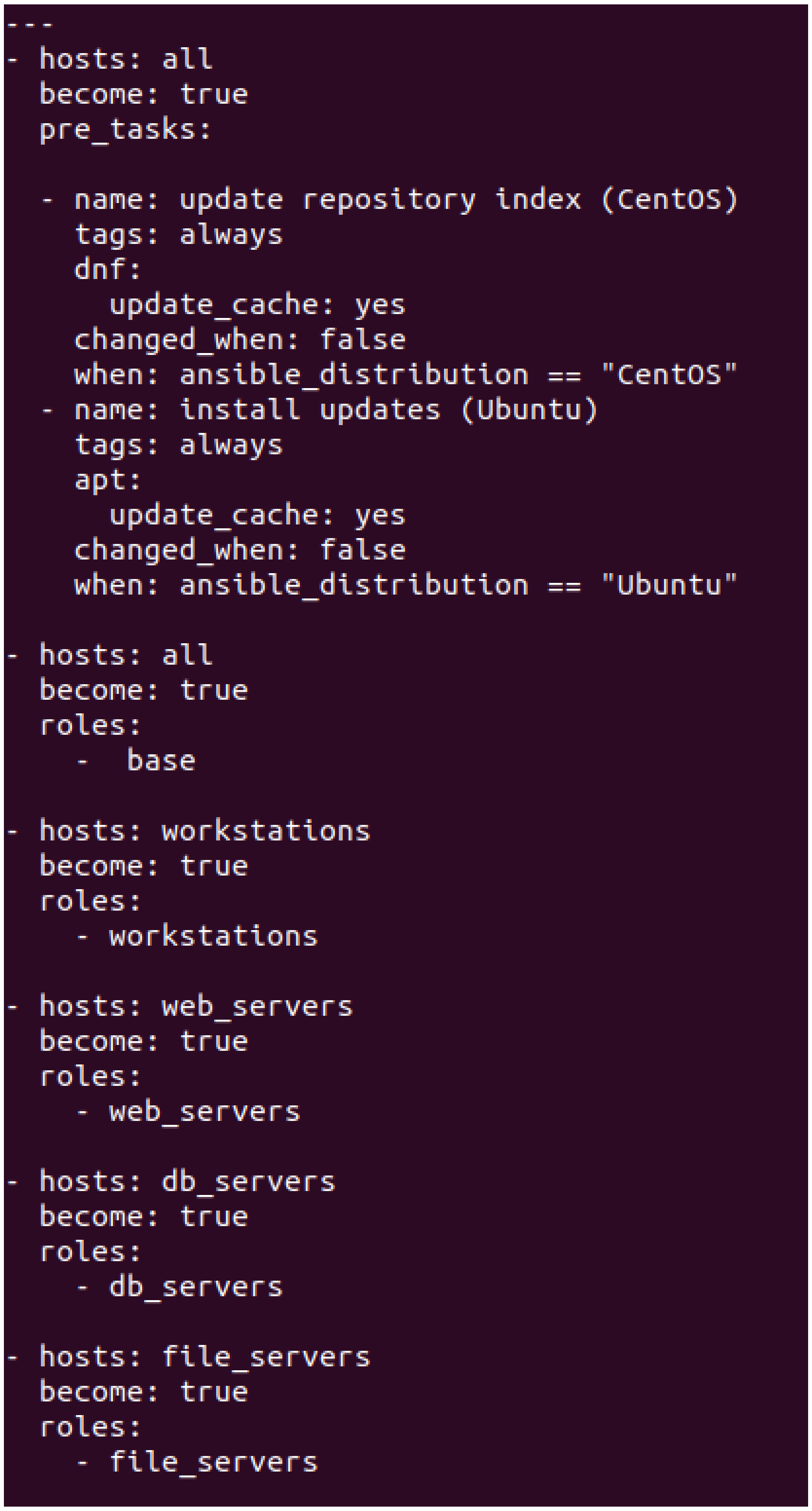


```
vbearl@server1:~$ terraform
Usage: terraform [-version] [-help] <command> [args]

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
    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
```

- When I typed “terraform”, it showed a list of all available commands that terraform supports, confirming that the package has been successfully installed into the remote server.

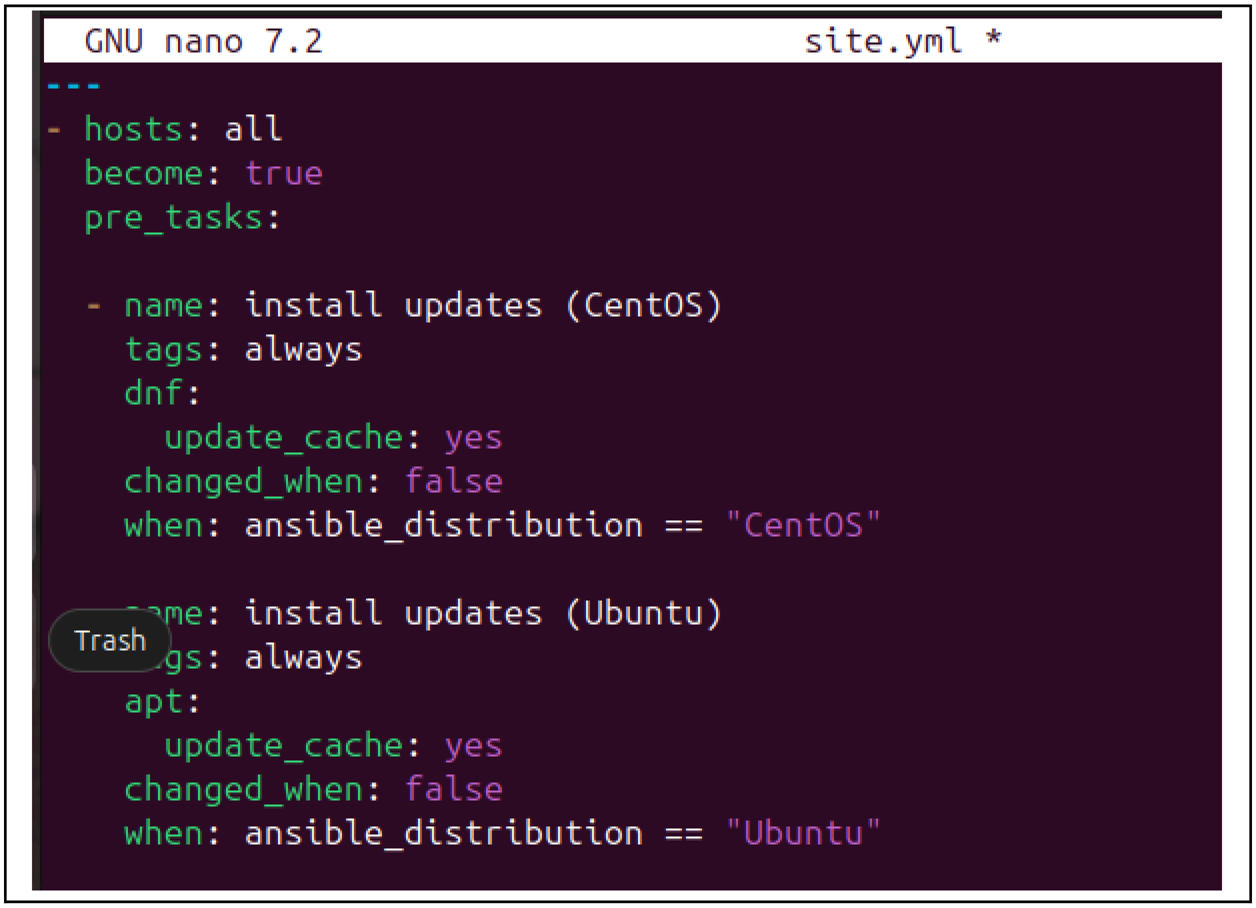
**Task 3: Create roles**

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)

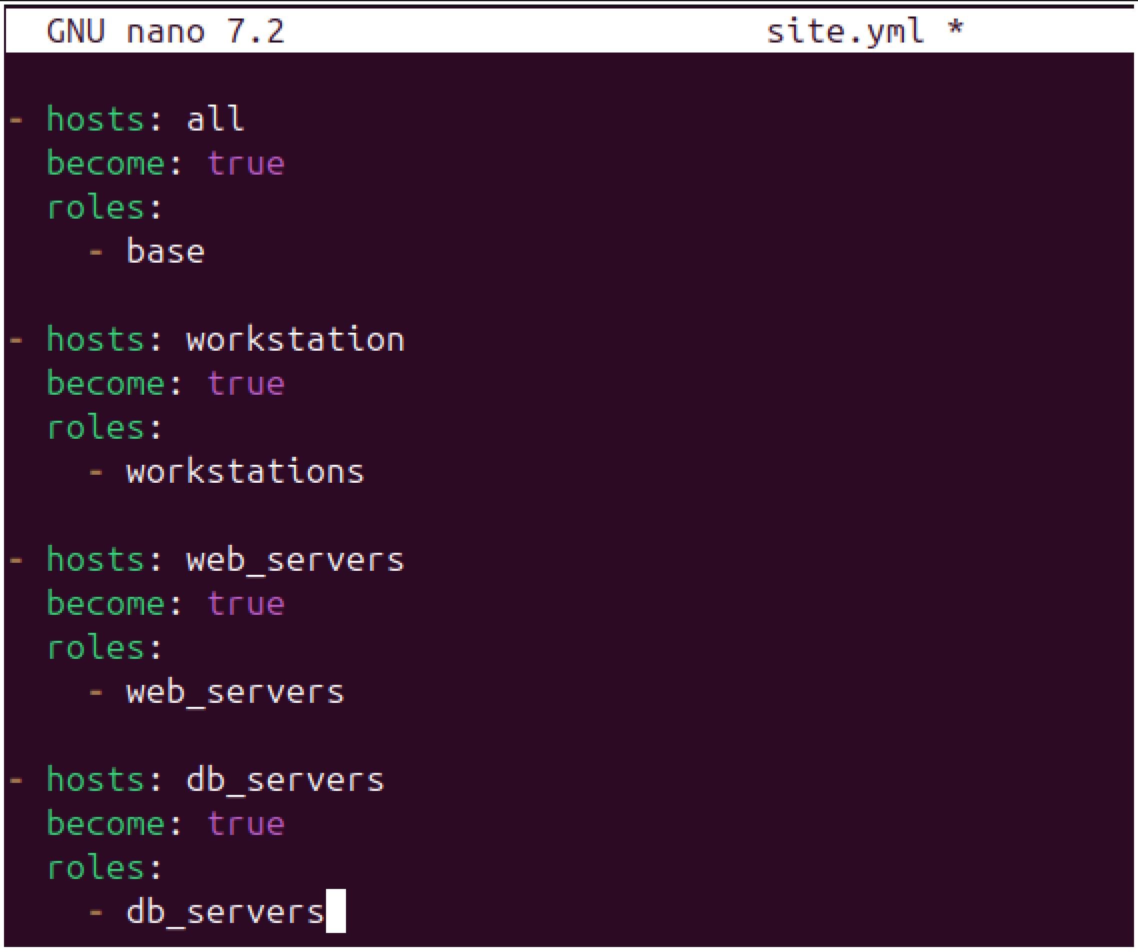


```
---
- hosts: all
  become: true
  pre_tasks:

  - 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
```

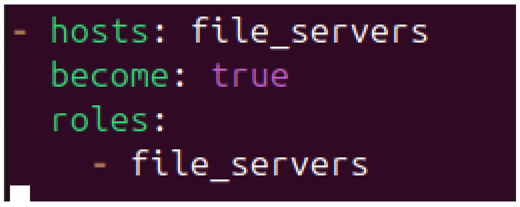
Save the file and exit.



```
  GNU nano 7.2                          site.yml *
---
- hosts: all
  become: true
  pre_tasks:

  - name: install updates (CentOS)
    tags: always
    dnf:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "CentOS"

     ame: install updates (Ubuntu)
      gs: always
    apt:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "Ubuntu"
```

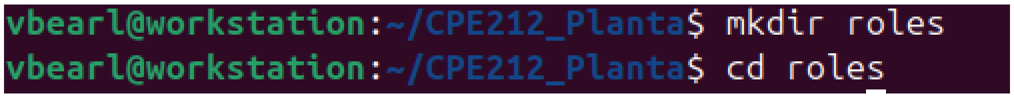


```
  GNU nano 7.2                          site.yml *

- hosts: all
  become: true
  roles:
    - base

- hosts: workstation
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
    - web_servers

- hosts: db_servers
  become: true
  roles:
    - db_servers
```

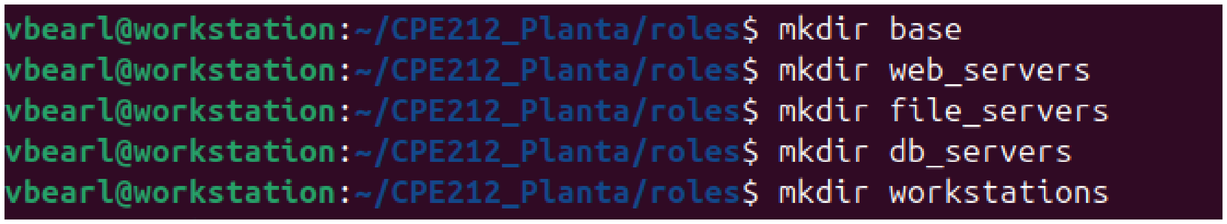


```
- hosts: file_servers
  become: true
  roles:
    - file_servers
```

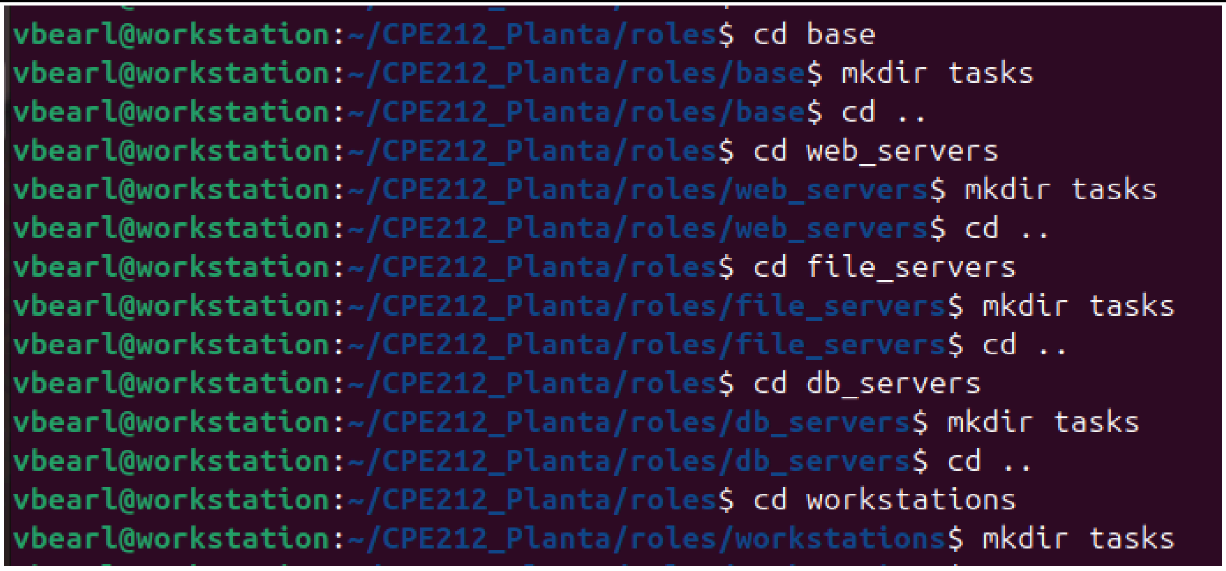
2. Under the same directory, create a new directory and name it roles. Enter the roles directory and create new directories: base, web_servers, file_servers, db_servers and workstations. For each directory, create a directory and name it tasks.



```
vbearl@workstation:~/CPE212_Planta$ mkdir roles
vbearl@workstation:~/CPE212_Planta$ cd roles
```



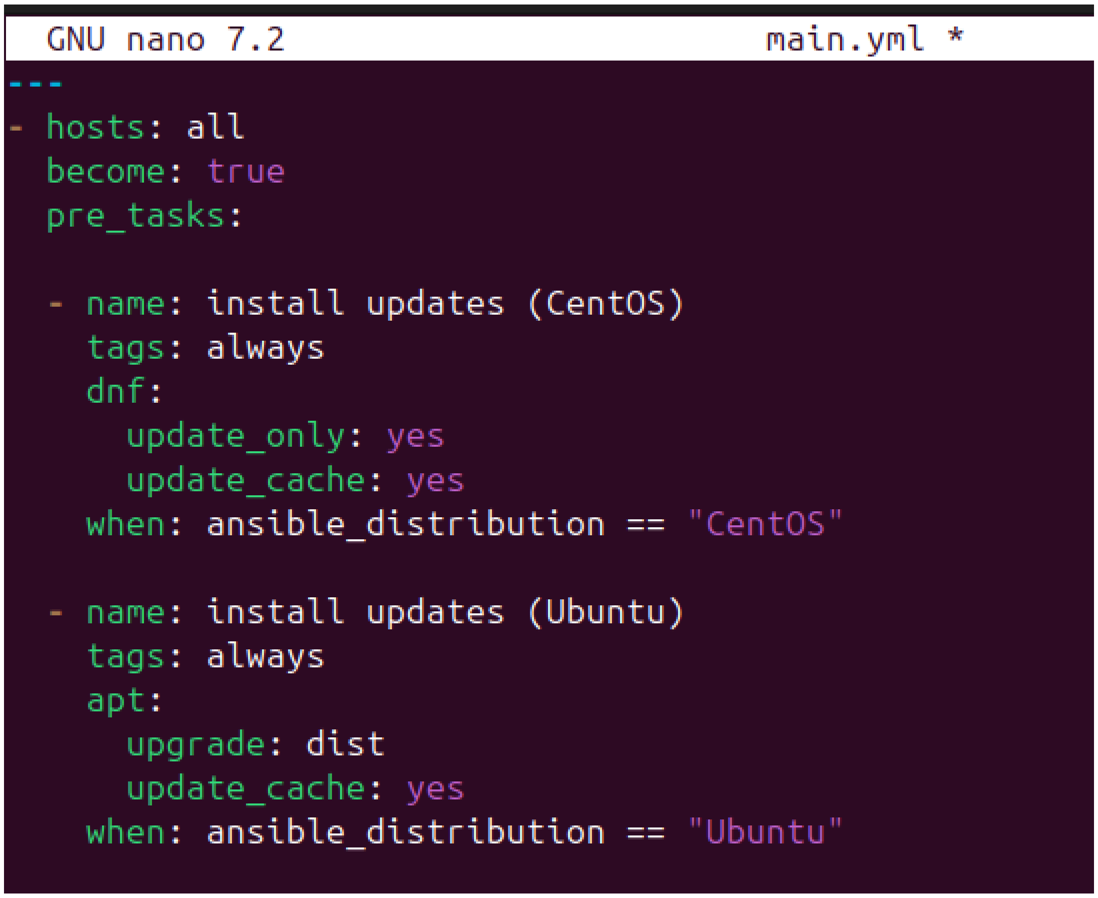
```
vbearl@workstation:~/CPE212_Planta/roles$ mkdir base
vbearl@workstation:~/CPE212_Planta/roles$ mkdir web_servers
vbearl@workstation:~/CPE212_Planta/roles$ mkdir file_servers
vbearl@workstation:~/CPE212_Planta/roles$ mkdir db_servers
vbearl@workstation:~/CPE212_Planta/roles$ mkdir workstations
```



```
vbearl@workstation:~/CPE212_Planta/roles$ cd base
vbearl@workstation:~/CPE212_Planta/roles/base$ mkdir tasks
vbearl@workstation:~/CPE212_Planta/roles/base$ cd ..
vbearl@workstation:~/CPE212_Planta/roles$ cd web_servers
vbearl@workstation:~/CPE212_Planta/roles/web_servers$ mkdir tasks
vbearl@workstation:~/CPE212_Planta/roles/web_servers$ cd ..
vbearl@workstation:~/CPE212_Planta/roles$ cd file_servers
vbearl@workstation:~/CPE212_Planta/roles/file_servers$ mkdir tasks
vbearl@workstation:~/CPE212_Planta/roles/file_servers$ cd ..
vbearl@workstation:~/CPE212_Planta/roles$ cd db_servers
vbearl@workstation:~/CPE212_Planta/roles/db_servers$ mkdir tasks
vbearl@workstation:~/CPE212_Planta/roles/db_servers$ cd ..
vbearl@workstation:~/CPE212_Planta/roles$ cd workstations
vbearl@workstation:~/CPE212_Planta/roles/workstations$ mkdir tasks
```

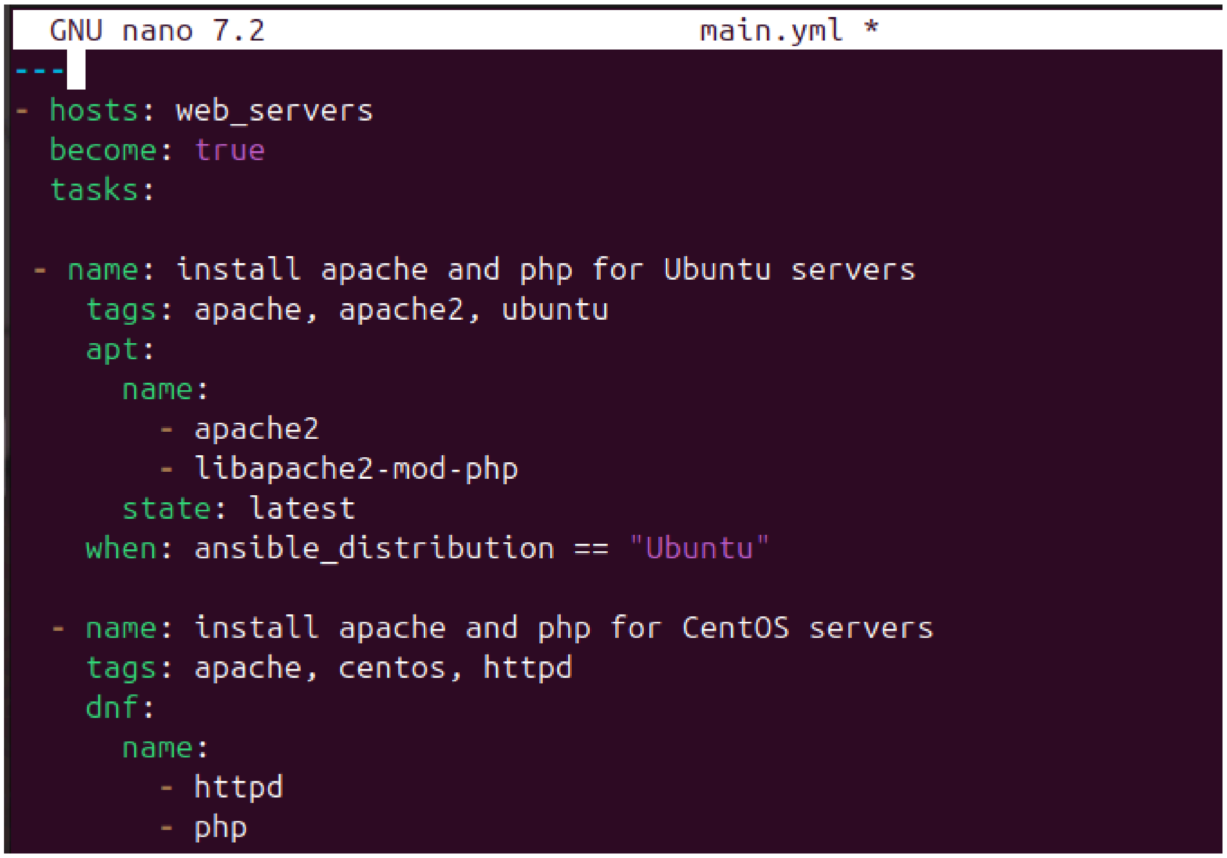
3. Go to tasks for all directory and create a file. Name it main.yml. In each of the tasks for all directories, copy and paste the code from the old site.yml file. Show all contents of main.yml files for all tasks.

**base role**

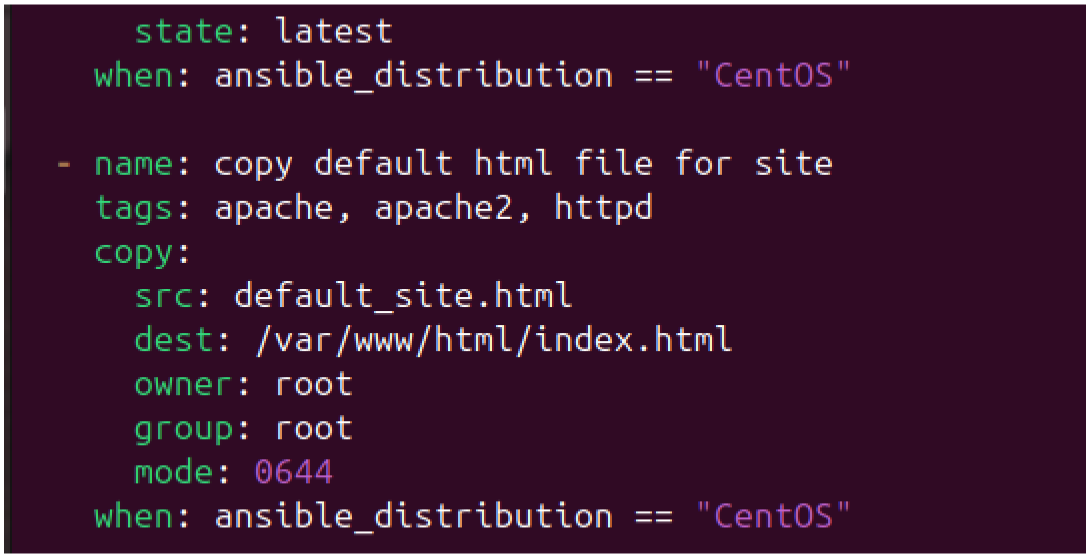


```
  GNU nano 7.2                          main.yml *
---
- hosts: all
  become: true
  pre_tasks:

  - name: install updates (CentOS)
    tags: always
    dnf:
      update_only: yes
      update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: install updates (Ubuntu)
    tags: always
    apt:
      upgrade: dist
      update_cache: yes
    when: ansible_distribution == "Ubuntu"
```

**web_server role**

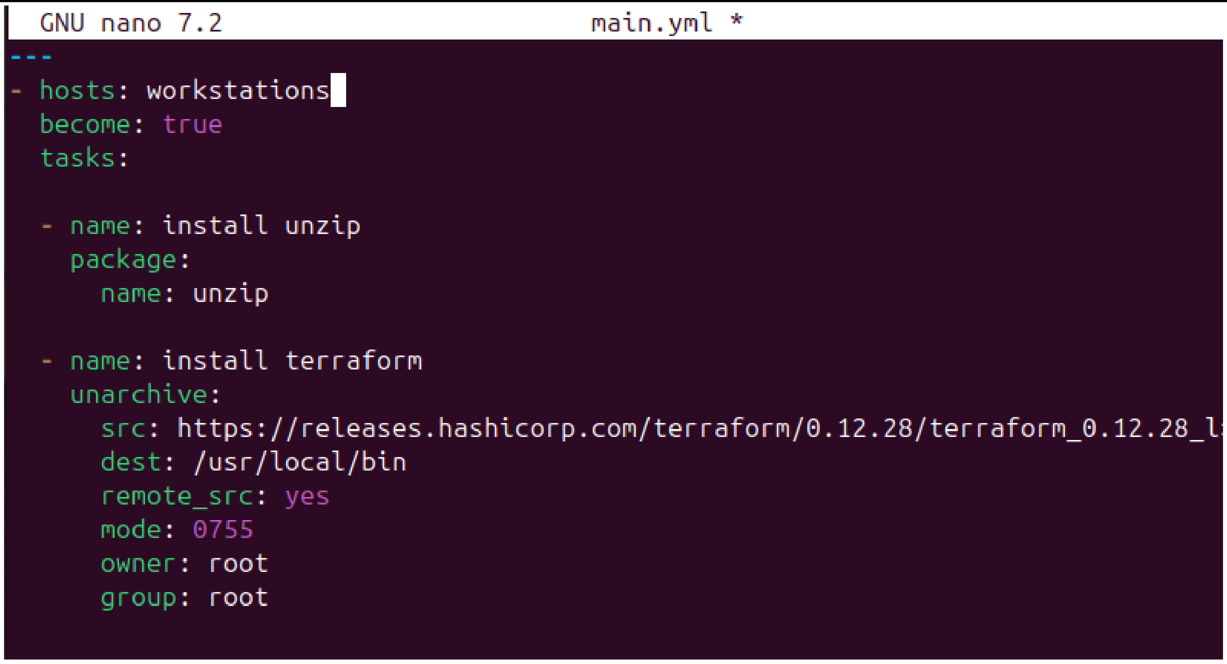


```
  GNU nano 7.2                                   main.yml *
---
- hosts: web_servers
  become: true
  tasks:

 - name: install apache and php for Ubuntu servers
   tags: apache, apache2, ubuntu
   apt:
     name:
       - apache2
       - libapache2-mod-php
     state: latest
   when: ansible_distribution == "Ubuntu"

  - name: install apache and php for CentOS servers
   tags: apache, centos, httpd
   dnf:
     name:
       - httpd
       - php
```



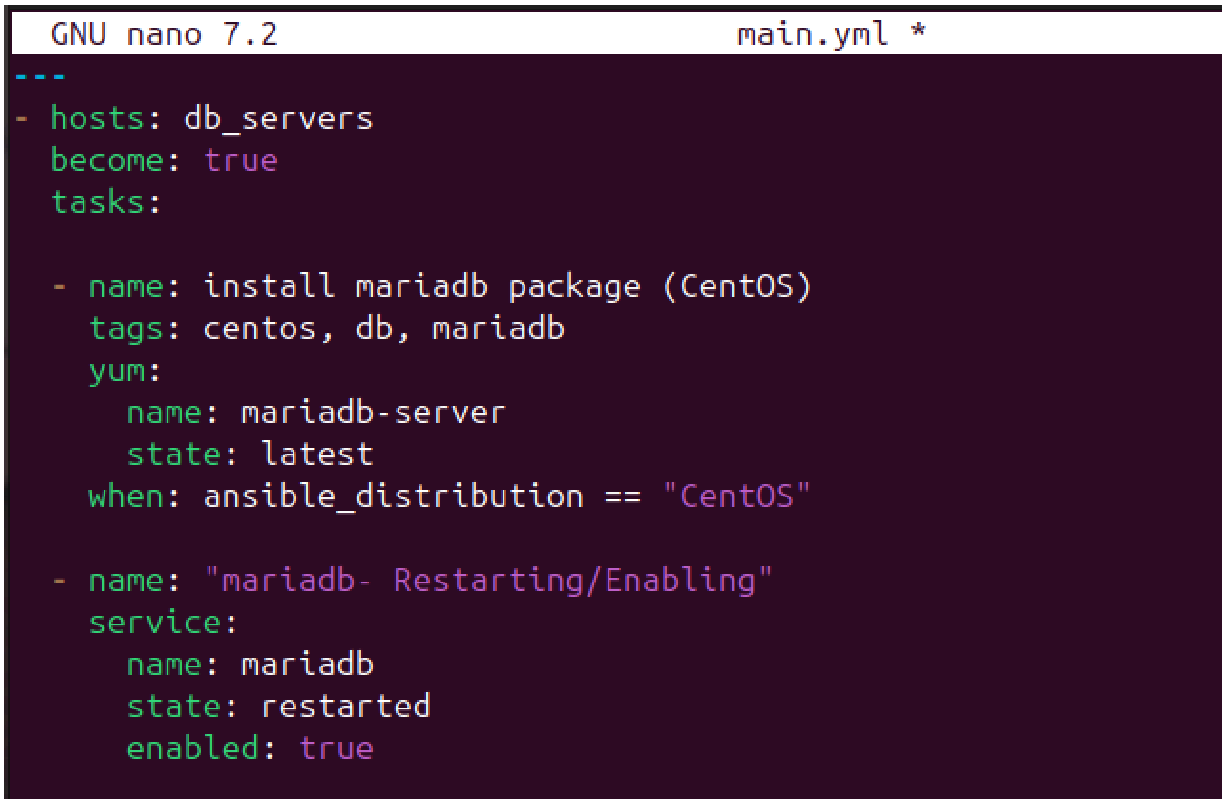
```
     state: latest
   when: ansible_distribution == "CentOS"

 - 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
   when: ansible_distribution == "CentOS"
```

**workstations role**

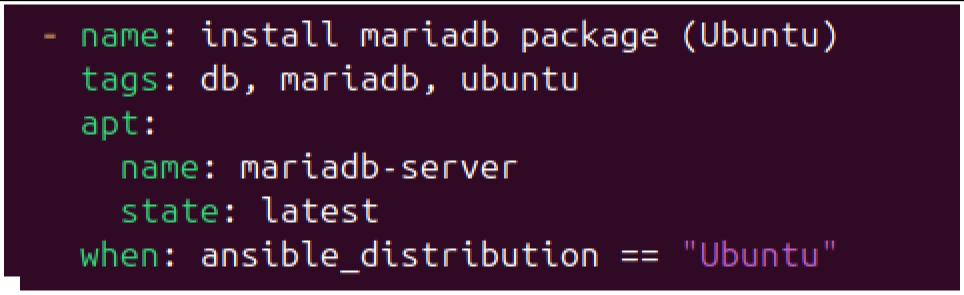


```
 GNU nano 7.2                          main.yml *
---
- hosts: workstations
  become: true
  tasks:

  - name: install unzip
    package:
      name: unzip

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

**db_servers role**

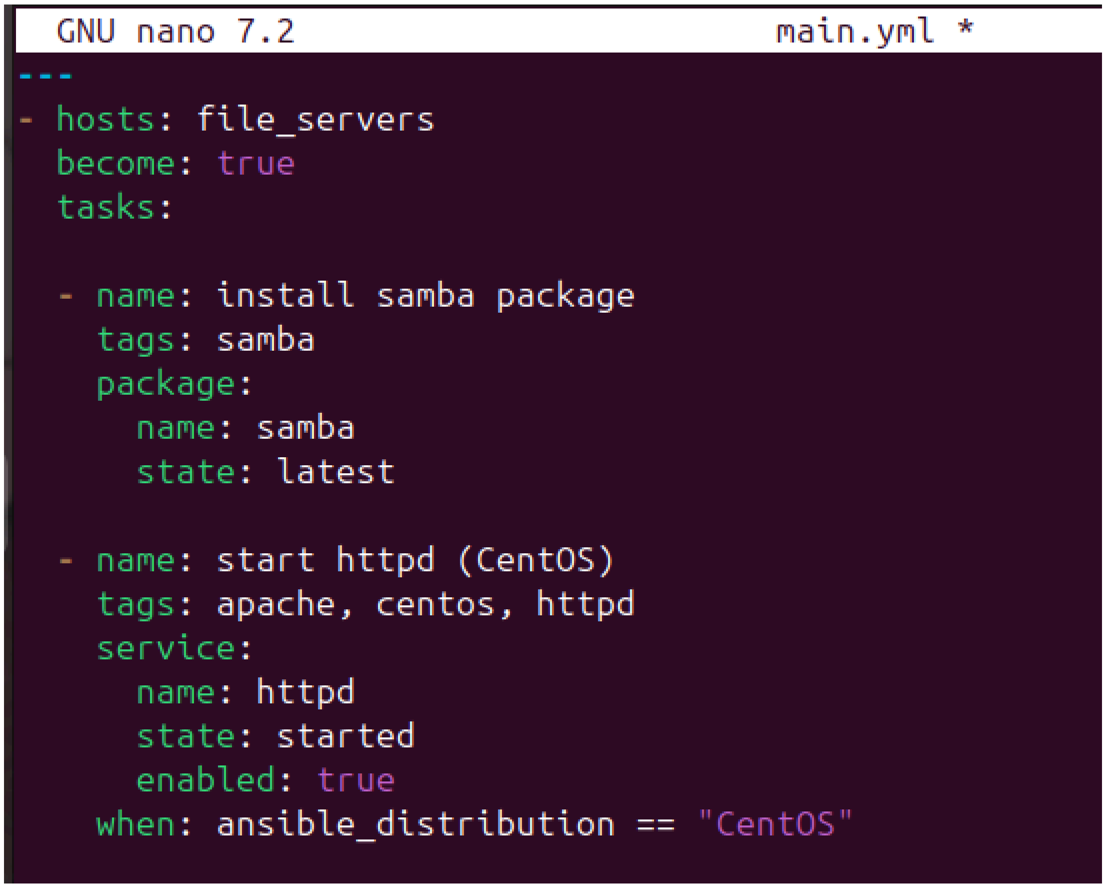


```
 GNU nano 7.2                          main.yml *
---
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    tags: centos, db, mariadb
    yum:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

  - name: "mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true
```



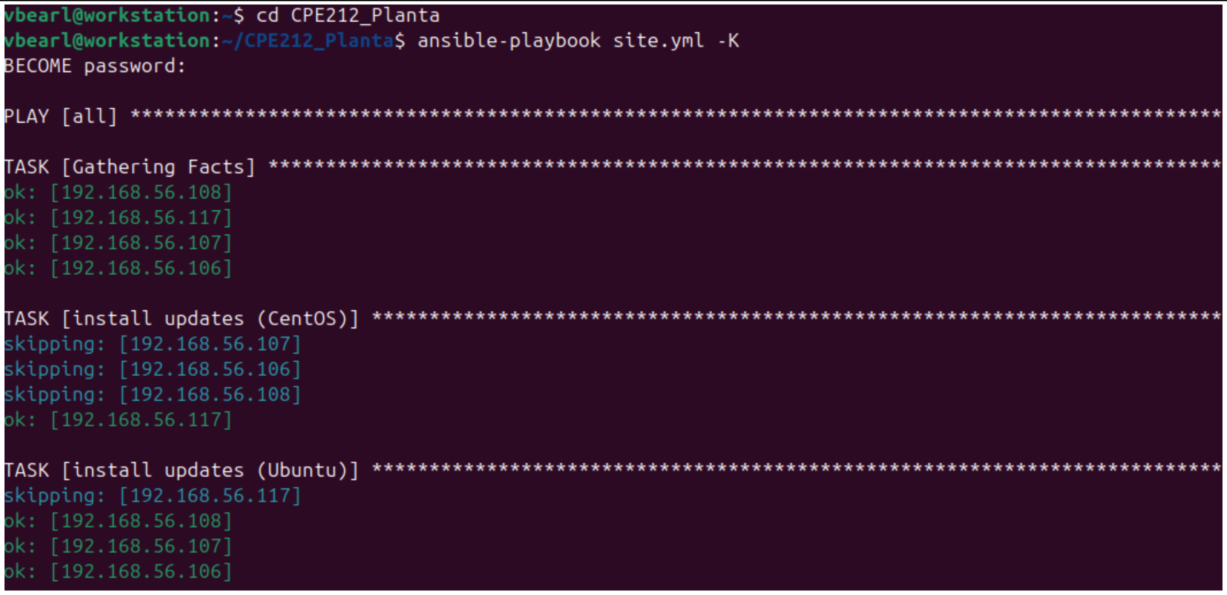
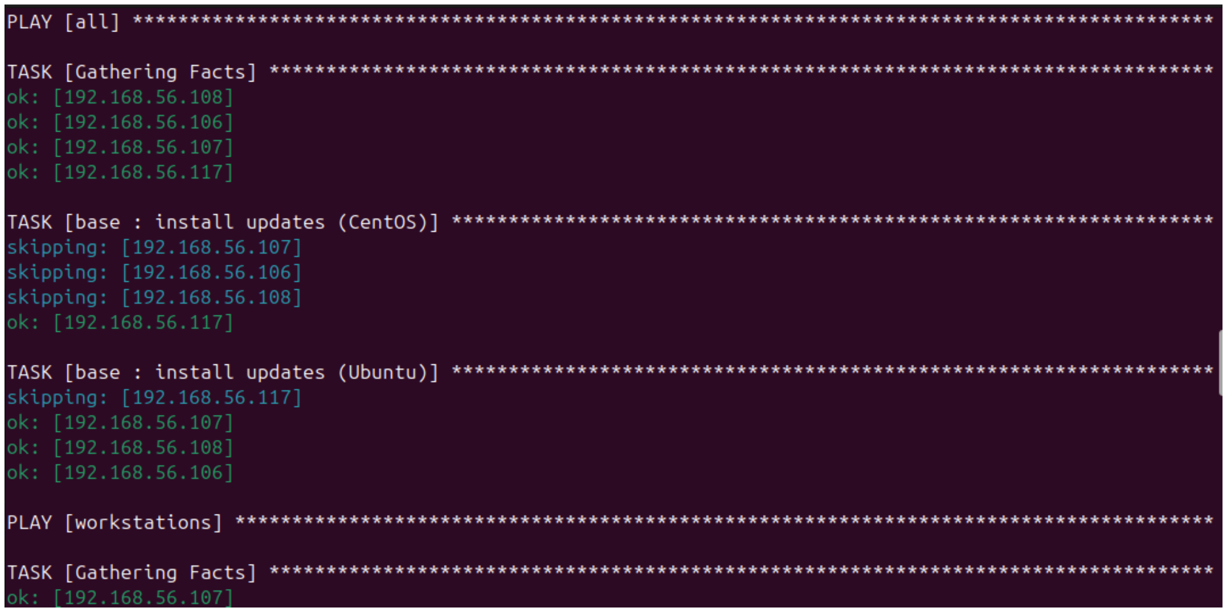
```
  - name: install mariadb package (Ubuntu)
    tags: db, mariadb, ubuntu
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"
```

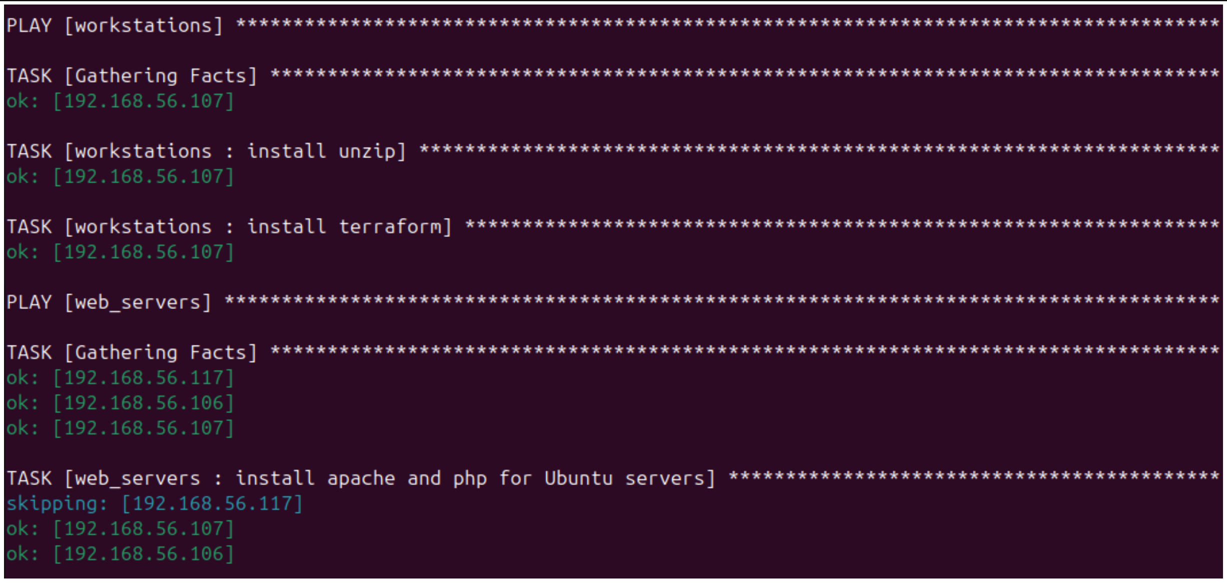
**file_servers role**



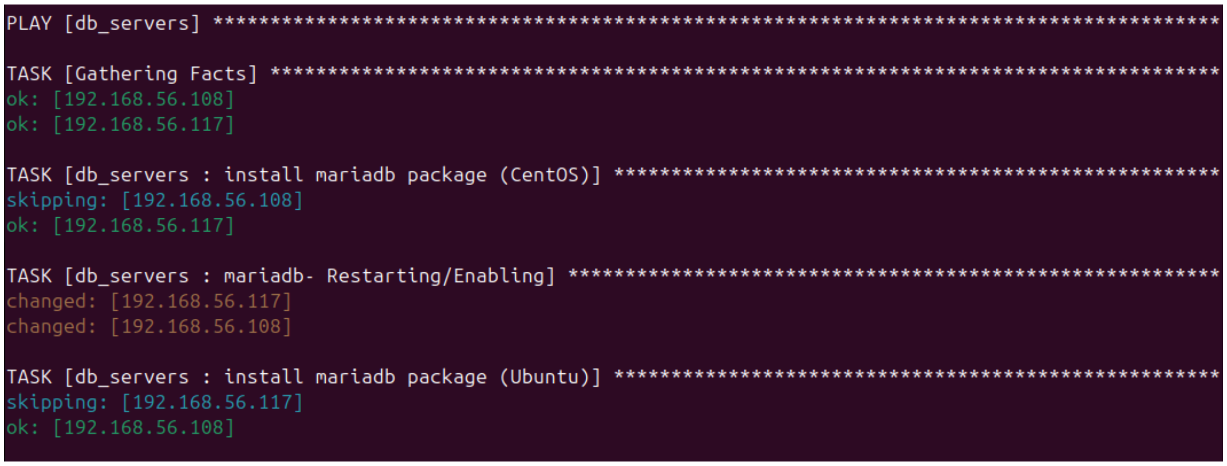
```
GNU nano 7.2                              main.yml *
---
- hosts: file_servers
  become: true
  tasks:

  - name: install samba package
    tags: samba
    package:
      name: samba
      state: latest

  - name: start httpd (CentOS)
    tags: apache, centos, httpd
    service:
      name: httpd
      state: started
      enabled: true
    when: ansible_distribution == "CentOS"
```

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

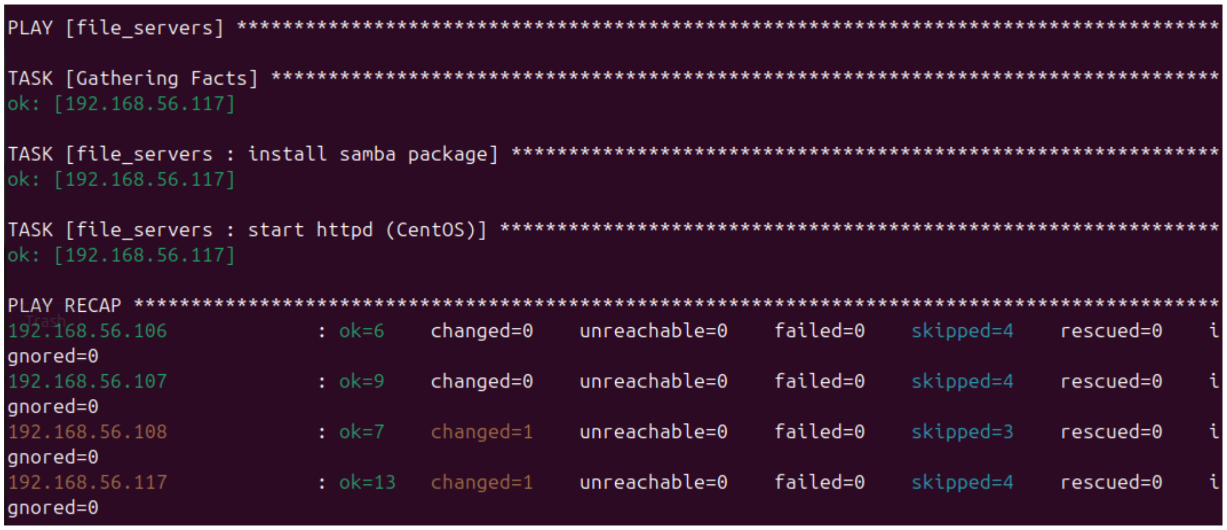

```
vbearl@workstation:~$ cd CPE212_Planta
vbearl@workstation:~/CPE212_Planta$ ansible-playbook site.yml -K
BECOME password:

PLAY [all] *****************************************************************************************

TASK [Gathering Facts] *****************************************************************************
ok: [192.168.56.108]
ok: [192.168.56.117]
ok: [192.168.56.107]
ok: [192.168.56.106]

TASK [install updates (CentOS)] ********************************************************************
skipping: [192.168.56.107]
skipping: [192.168.56.106]
skipping: [192.168.56.108]
ok: [192.168.56.117]

TASK [install updates (Ubuntu)] ********************************************************************
skipping: [192.168.56.117]
ok: [192.168.56.108]
ok: [192.168.56.107]
ok: [192.168.56.106]
```

```
PLAY [all] *****************************************************************************************

TASK [Gathering Facts] *****************************************************************************
ok: [192.168.56.108]
ok: [192.168.56.106]
ok: [192.168.56.107]
ok: [192.168.56.117]

TASK [base : install updates (CentOS)] *************************************************************
skipping: [192.168.56.107]
skipping: [192.168.56.106]
skipping: [192.168.56.108]
ok: [192.168.56.117]

TASK [base : install updates (Ubuntu)] *************************************************************
skipping: [192.168.56.117]
ok: [192.168.56.107]
ok: [192.168.56.108]
ok: [192.168.56.106]

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

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



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

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

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

TASK [workstations : install terraform] *************************************************
ok: [192.168.56.107]

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

TASK [Gathering Facts] ******************************************************************
ok: [192.168.56.117]
ok: [192.168.56.106]
ok: [192.168.56.107]

TASK [web_servers : install apache and php for Ubuntu servers] **************************
skipping: [192.168.56.117]
ok: [192.168.56.107]
ok: [192.168.56.106]
```



```
PLAY [db_servers] ***********************************************************************

TASK [Gathering Facts] ******************************************************************
ok: [192.168.56.108]
ok: [192.168.56.117]

TASK [db_servers : install mariadb package (CentOS)] ************************************
skipping: [192.168.56.108]
ok: [192.168.56.117]

TASK [db_servers : mariadb- Restarting/Enabling] ****************************************
changed: [192.168.56.117]
changed: [192.168.56.108]

TASK [db_servers : install mariadb package (Ubuntu)] ************************************
skipping: [192.168.56.117]
ok: [192.168.56.108]
```



```
PLAY [file_servers] *********************************************************************

TASK [Gathering Facts] ******************************************************************
ok: [192.168.56.117]

TASK [file_servers : install samba package] *********************************************
ok: [192.168.56.117]

TASK [file_servers : start httpd (CentOS)] **********************************************
ok: [192.168.56.117]

PLAY RECAP ******************************************************************************
192.168.56.106             : ok=6    changed=0    unreachable=0    failed=0    skipped=4    rescued=0    i
gnored=0
192.168.56.107             : ok=9    changed=0    unreachable=0    failed=0    skipped=4    rescued=0    i
gnored=0
192.168.56.108             : ok=7    changed=1    unreachable=0    failed=0    skipped=3    rescued=0    i
gnored=0
192.168.56.117             : ok=13   changed=1    unreachable=0    failed=0    skipped=4    rescued=0    i
gnored=0
```

- The revised playbook runs the same output as the previous version of the playbook before. The difference is we split the tasks and separated each task into a new yml file in the directories of their respective targeted roles. The site.yml now only calls the name of the roles, which will run their respective main.yml files through the site.yml file.

**Reflections:**

Answer the following:

1. What is the importance of creating roles?
    - Creating roles for ansible playbooks promotes modularity, resulting in a much easier script management and improved selective play execution. With roles, it's easier to change what plays we want to run in a playbook, by simply specifying the role name rather than manually editing numerous lines of codes, which is generally harder and more time-consuming.

2. What is the importance of managing files?
    - Managing files in ansible improves file organization and management, as it allows us to specify where we want a file to be located. When we extract files using ansible playbooks, we can specify its file destination using “dest:”. With this, we can manage file locations easier and avoid file misplacement errors.