Name: Zamora, Angelo E.	**Date Performed:** 10-07-2024
Course/Section: CPE31S2	**Date Submitted:** 10-09-2024
Instructor: Engr. Robin Valenzuela	**Semester and SY:** 1st Semester 2024 - 2025

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.

```
zamora_admin@workstation:~/TIP_HOA-7.1_ZAMORA_Angelo$ mkdir files
zamora_admin@workstation:~/TIP_HOA-7.1_ZAMORA_Angelo$ ls
ansible.cfg  files  inventory  README.md  site.yml
zamora_admin@workstation:~/TIP_HOA-7.1_ZAMORA_Angelo$
```

```
zamora_admin@workstation: ~/TIP_HOA-7.1_ZAMORA_Angelo/files
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                default_site.html                Modified

<h1>Hello TIP</h1>

<p>Hello! I am Angelo Zamora, a CPE Student in TIP.</p>
```

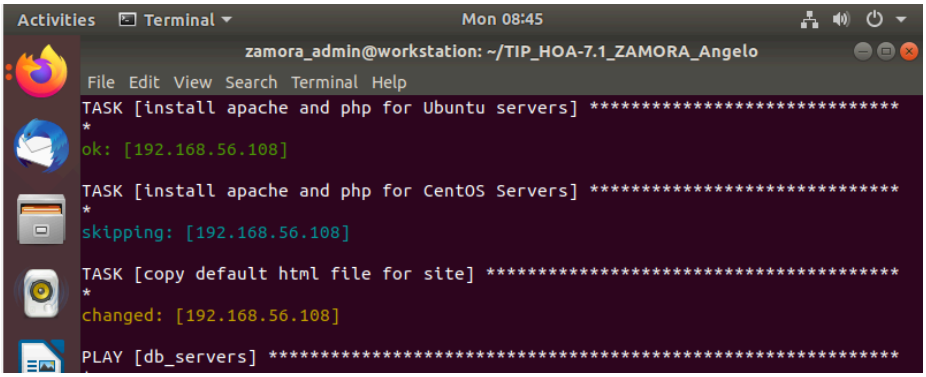
```
zamora_admin@workstation:~/TIP_HOA-7.1_ZAMORA_Angelo/files$ ls
default_site.html
zamora_admin@workstation:~/TIP_HOA-7.1_ZAMORA_Angelo/files$
```

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.



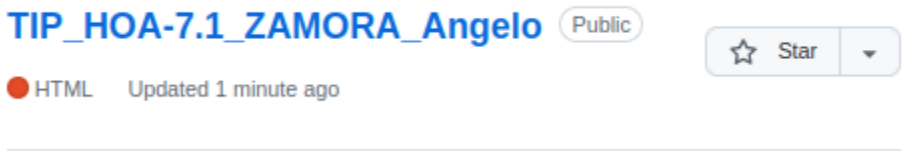
   - The playbook runs successfully in copying the html file to the remote servers.

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.

```
zamora_admin@server1:~$ cat /var/www/html/index.html
<h1>Hello TIP</h1>

<p>Hello! I am Angelo Zamora, a CPE Student in TIP.</p>
zamora_admin@server1:~$
```

- When you've ran the playbook it copies the file you've created in the control node files directory and directly copies to the remote server. We've also set the ownership and permission of the file as well.

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



- **When we've sync the local repo to our VM's it outputs an HTML tag. It means it contains an HTML file on the repo.**

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

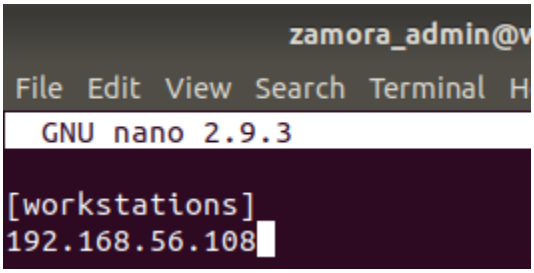
1. Edit the site.yml. Just before the web_servers play, create a new play:

```
- 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.$
         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.



3. Run the playbook. Describe the output.

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

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

TASK [install unzip] *********
*
ok: [192.168.56.108]

TASK [install terraform] *****
*
changed: [192.168.56.108]
```

- This time we've added a new play this time for workstations and install terraform and unzip packages.

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

```
zamora_admin@server1:~$ terraform -version
Terraform v0.12.28
```

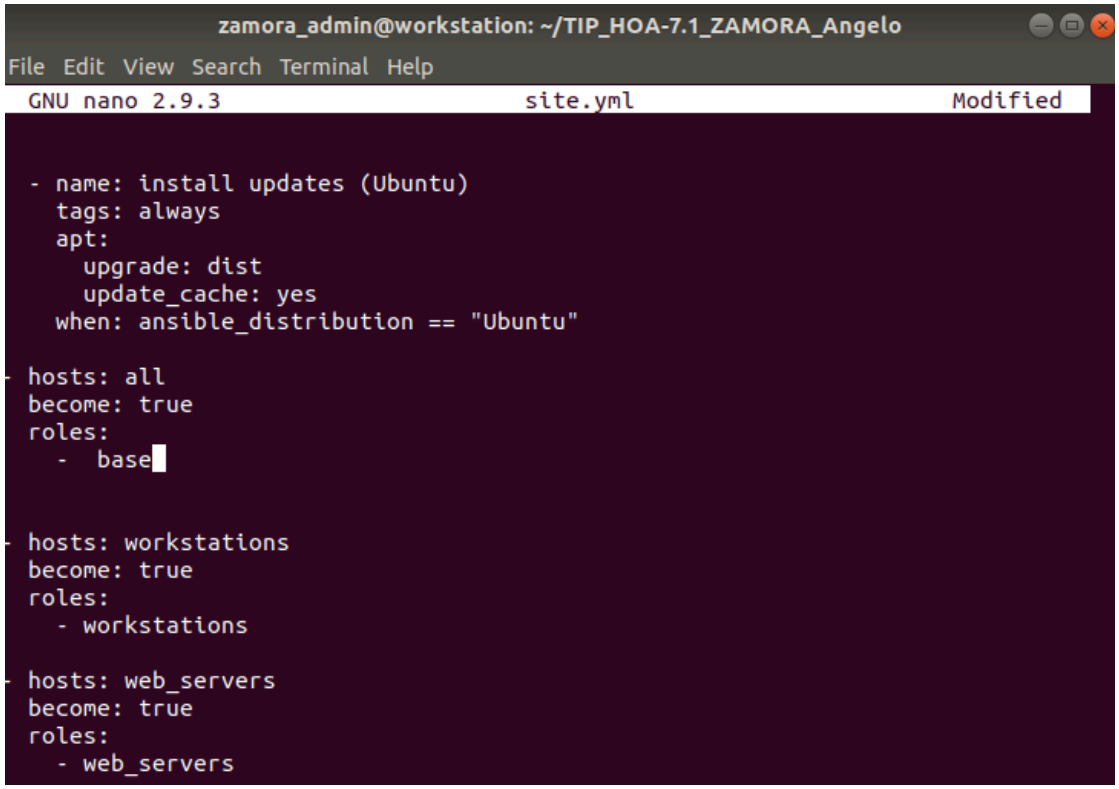
- This command verifies if we have terraform inside our remote server.

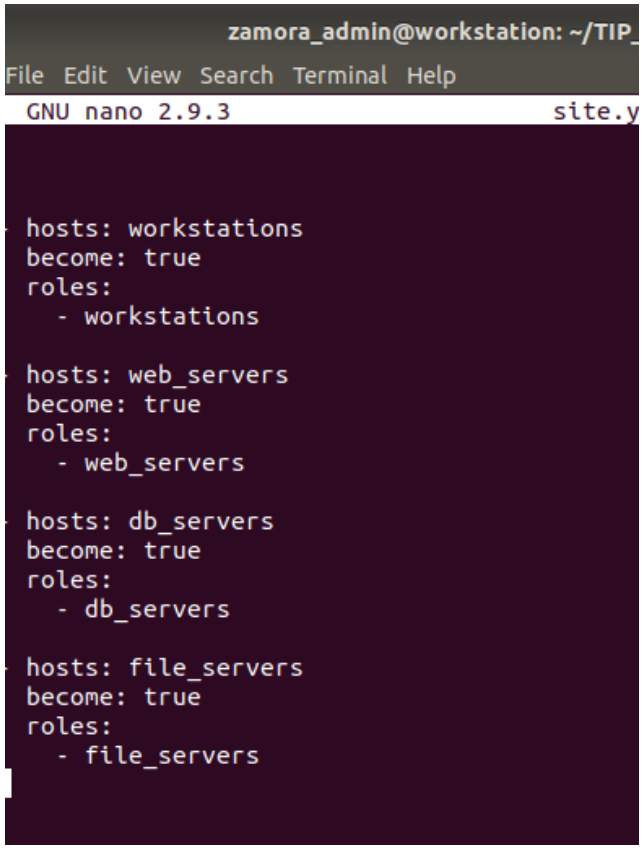
Task 3: Create roles

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.





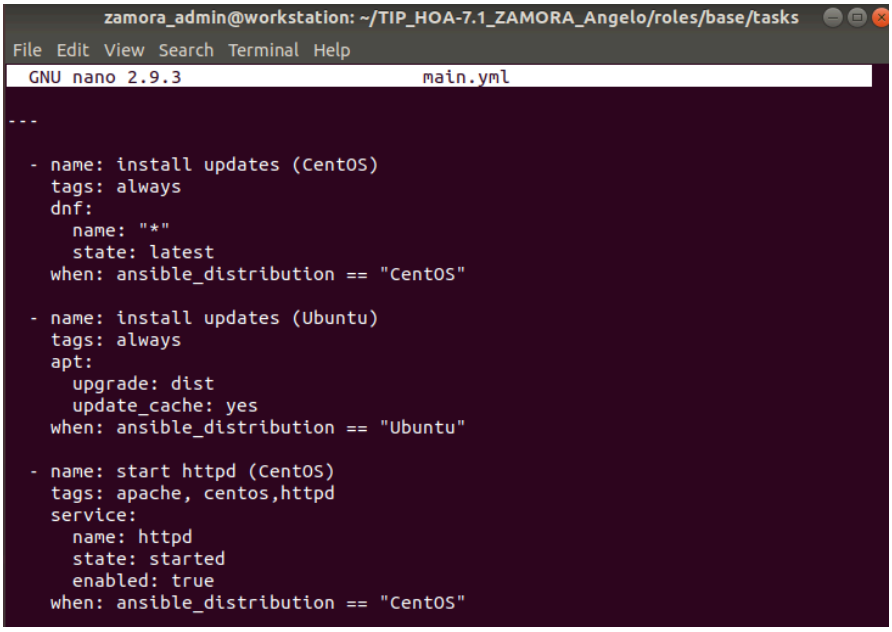
2. Under the same directory, create a new directory and name it roles. Enter the roles directory and create new directories: base, web_servers, file_servers, db_servers and workstations. For each directory, create a directory and name it tasks.

```
zamora_admin@workstation:~/TIP_HOA-7.1_ZAMORA_Angelo/roles$ tree
.
├── base
│   └── tasks
│       └── main.yml
├── db_servers
│   └── tasks
│       └── main.yml
├── file_servers
│   └── tasks
│       └── main.yml
├── web_servers
│   └── tasks
│       └── main.yml
└── workstations
    └── tasks
        └── main.yml
```

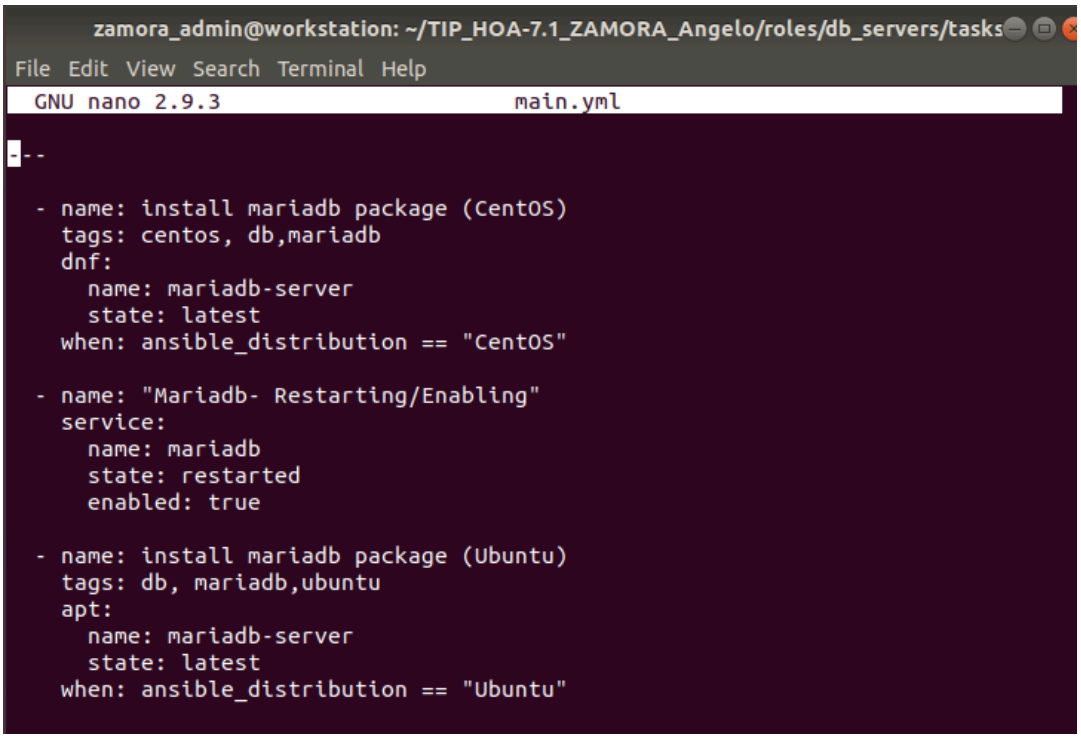
3. Go to tasks for all directory and create a file. Name it main.yml. In each of the tasks for all directories, copy and paste the code from the old site.yml file. Show all contents of main.yml files for all tasks.

```
zamora_admin@workstation:~/TIP_HOA-7.1_ZAMORA_Angelo/roles$ tree
.
├── base
│   └── tasks
│       └── main.yml
├── db_servers
│   └── tasks
│       └── main.yml
├── file_servers
│   └── tasks
│       └── main.yml
├── web_servers
│   └── tasks
│       └── main.yml
└── workstations
    └── tasks
        └── main.yml
```

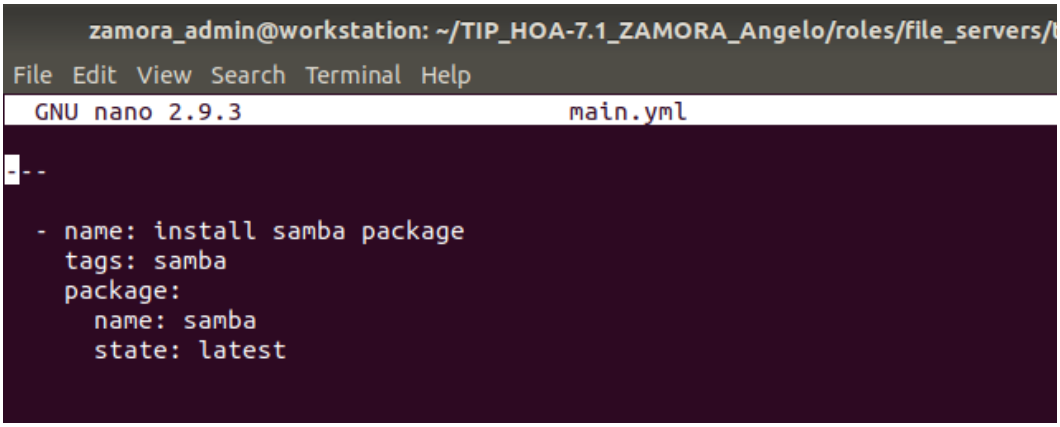
Base:



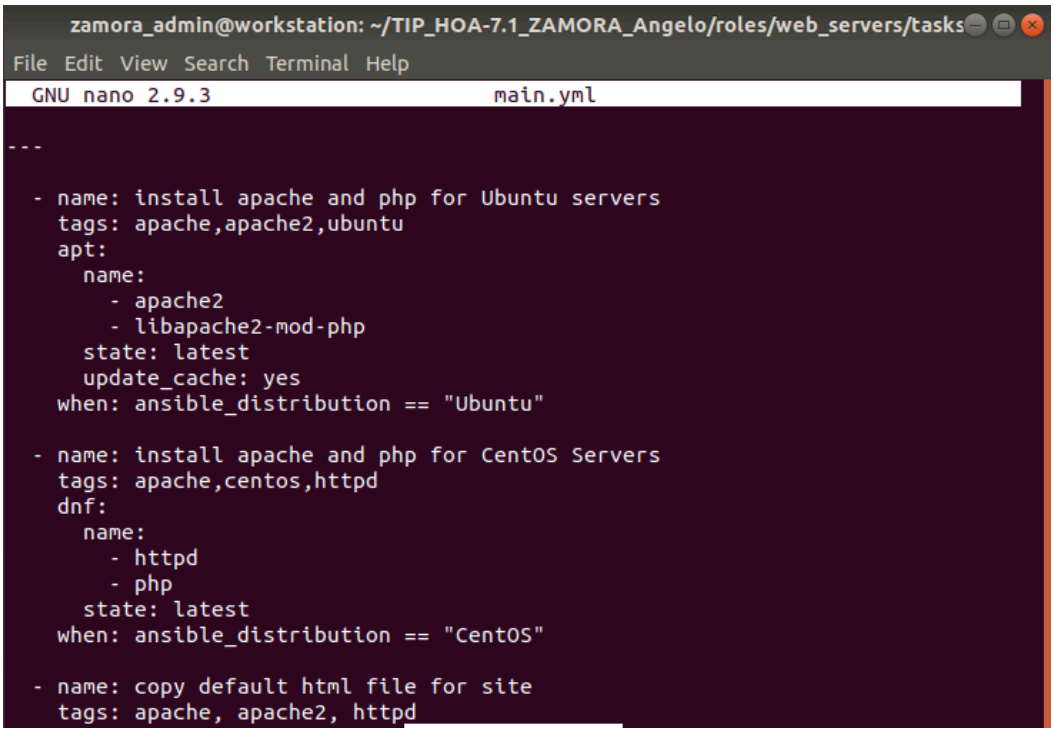
db_servers:



file_servers:



web_servers:



workstations:

```
zamora_admin@workstation: ~/TIP_HOA-7.1_ZAMORA_Angelo/roles/workstations/tasks
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                        main.yml

---

  -  name: install unzip
     package:
         name: unzip

  -  name: install terraform
     unarchive:
         src: https://releases.hashicorp.com/terraform/0.12.28/terraform_0.12.$
         dest: /usr/local/bin
         remote_src: yes
         mode: 0755
         owner: root
         group: root

                              [ Read 16 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell
```

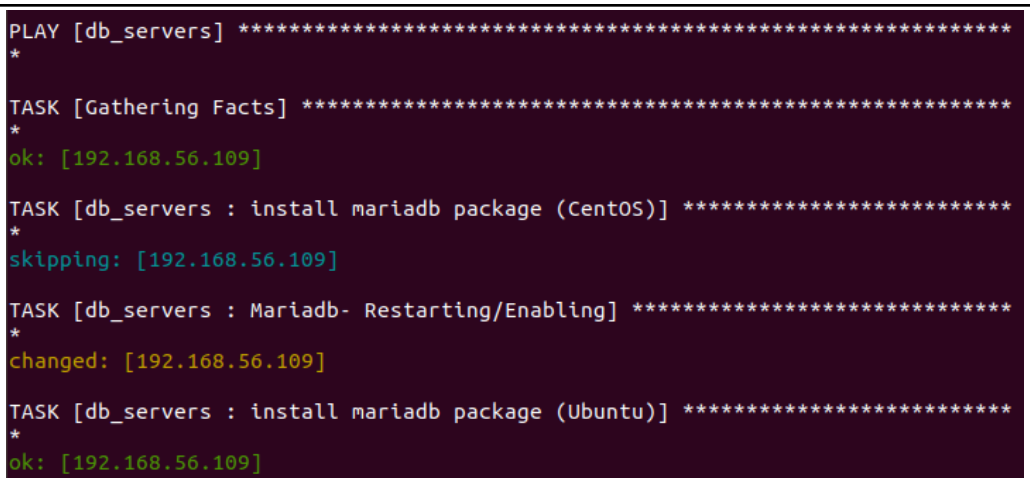
4. Run the site.yml playbook and describe the output.

```
zamora_admin@workstation:~/TIP_HOA-7.1_ZAMORA_Angelo$ ansible-playbook --ask-be
come-pass site.yml
SUDO password:

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

TASK [Gathering Facts] ********************************************************
*
ok: [192.168.56.108]
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [install updates (CentOS)] ***********************************************
*
skipping: [192.168.56.108]
skipping: [192.168.56.109]
ok: [192.168.56.110]

TASK [install updates (Ubuntu)] ***********************************************
*
skipping: [192.168.56.110]
ok: [192.168.56.109]
ok: [192.168.56.108]

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

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

TASK [Gathering Facts] ********************************************************
*
ok: [192.168.56.108]
ok: [192.168.56.109]
ok: [192.168.56.110]

TASK [base : install updates (CentOS)] ****************************************
*
skipping: [192.168.56.108]
skipping: [192.168.56.109]
ok: [192.168.56.110]

TASK [base : install updates (Ubuntu)] ****************************************
*
skipping: [192.168.56.110]
ok: [192.168.56.109]
ok: [192.168.56.108]

TASK [base : start httpd (CentOS)] ********************************************
*
skipping: [192.168.56.108]
skipping: [192.168.56.109]
ok: [192.168.56.110]
```

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

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

TASK [workstations : install unzip] *******************************************
*
ok: [192.168.56.108]

TASK [workstations : install terraform] ***************************************
*
ok: [192.168.56.108]
```

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

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

TASK [web_servers : install apache and php for Ubuntu servers] ****************
*
ok: [192.168.56.108]

TASK [web_servers : install apache and php for CentOS Servers] ****************
*
skipping: [192.168.56.108]

TASK [web_servers : copy default html file for site] **************************
*
ok: [192.168.56.108]
```



- **In this result, you'll see that it's like before but more manageable in a way that you create multiple playbooks for different hosts and create the main playbook to run all the playbooks per role. This shows to me that you can run a playbook's playbook simultaneously. As we run the playbook, it goes through the roles folder and runs the specific tasks playbook per role.**

Reflections:

Answer the following:

1. What is the importance of creating roles? The importance of roles is like playing in a team sport. We have different roles and responsibilities that we need to do in our system. This is just that, with roles in ansible we can organize each task by simply creating a playbook per role. As I've learned that an Ansible Playbook can run a playbook as well that will run the specific play per role and task within the assigned playbook per role. This ensures efficiency and organization in different roles. Additionally, it is reusable and shareable to other users as well.

2. What is the importance of managing files?
   Managing files is one of the most essential tasks to do and somehow we are too lazy to do this task. The playbook has the power to create, manage, and secure files. With this ability we can create, manage, and secure a file in one go to multiple computers simultaneously. File Management Automation is present here in being able to perform file related tasks and it is important to have that ability to not lose the file in the system.

   Github Link: https://github.com/GeloaceRT/TIP_HOA-7.1_ZAMORA_Angelo