Name: Calderon Ricardo B.	**Date Performed:03/11/2024**
Course/Section: CPE 232-CPE31S1	**Date Submitted:03/12/2024**
Instructor: Dr. Jonathan Taylar	**Semester and SY: 1st sem/2023-2024**

Activity 7: Managing Files and Creating Roles in Ansible

1. **Objectives:**

1.1 Manage files in remote servers

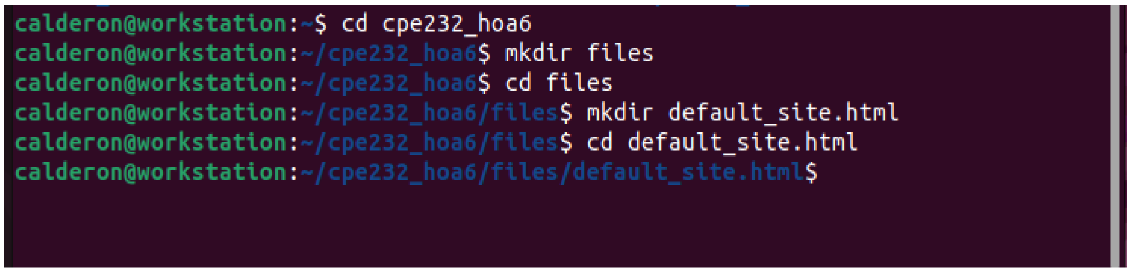
1.2 Implement roles in ansible

2. **Discussion**:

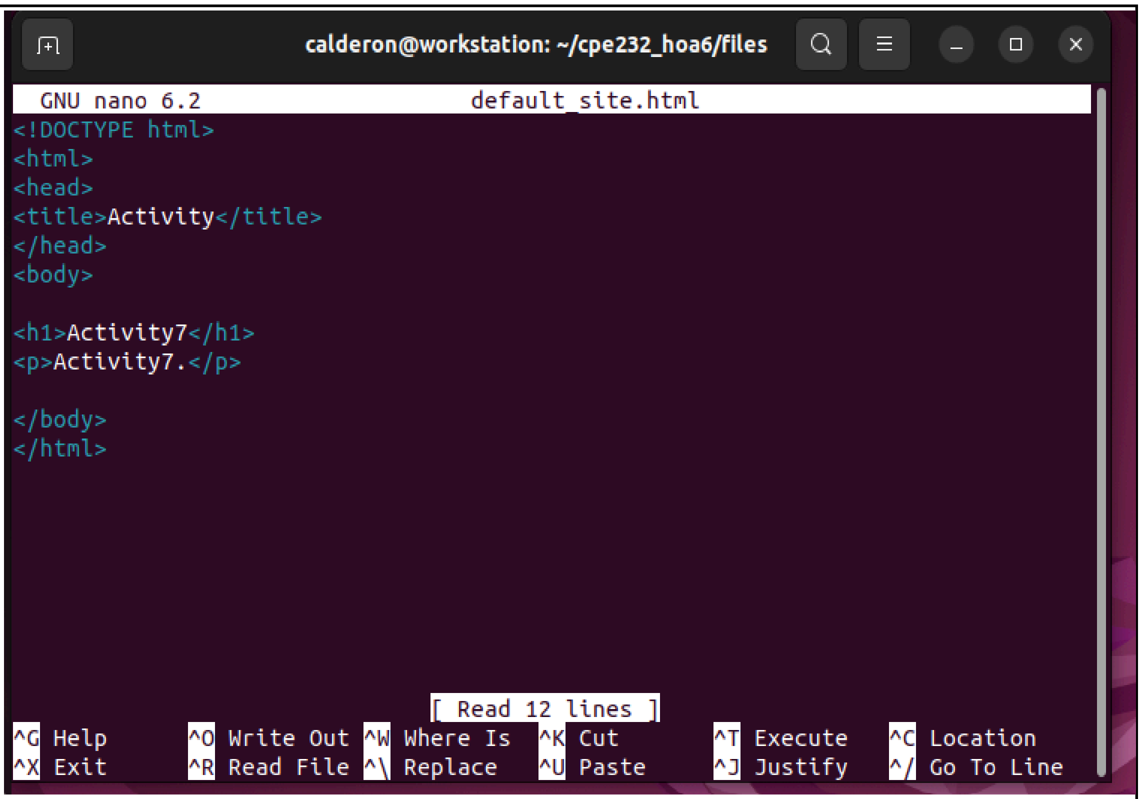
In this activity, we look at the concept of copying a file to a server. We are going to create a file into our git repository and use Ansible to grab that file and put it into a particular place so that we could do things like customize a default website, or maybe install a default configuration file.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



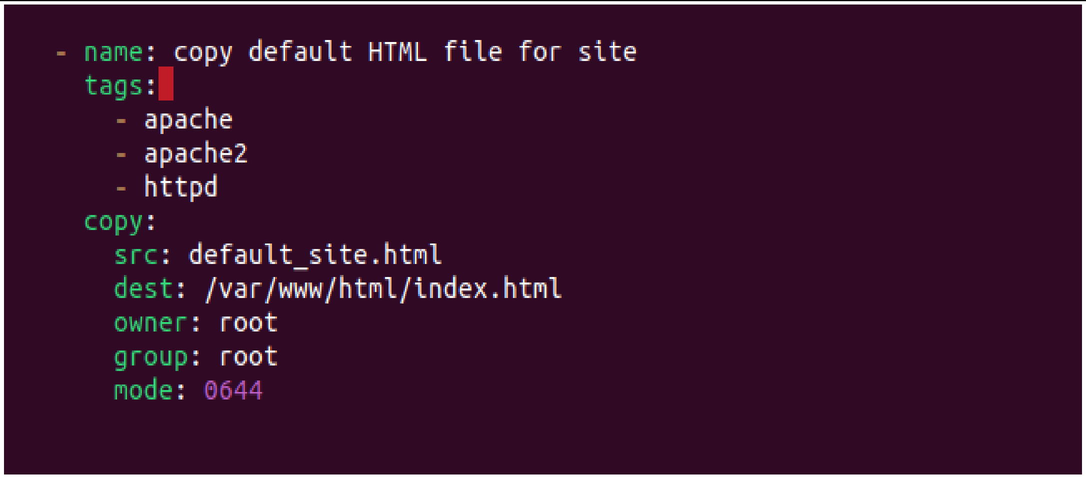
```
calderon@workstation:~$ cd cpe232_hoa6
calderon@workstation:~/cpe232_hoa6$ mkdir files
calderon@workstation:~/cpe232_hoa6$ cd files
calderon@workstation:~/cpe232_hoa6/files$ mkdir default_site.html
calderon@workstation:~/cpe232_hoa6/files$ cd default_site.html
calderon@workstation:~/cpe232_hoa6/files/default_site.html$
```



```
  GNU nano 6.2                          default_site.html
<!DOCTYPE html>
<html>
<head>
<title>Activity</title>
</head>
<body>

<h1>Activity7</h1>
<p>Activity7.</p>

</body>
</html>








                                [ Read 12 lines ]
^G Help       ^O Write Out ^W Where Is  ^K Cut        ^T Execute    ^C Location
^X Exit       ^R Read File ^\ Replace   ^U Paste      ^J Justify    ^/ Go To Line
```

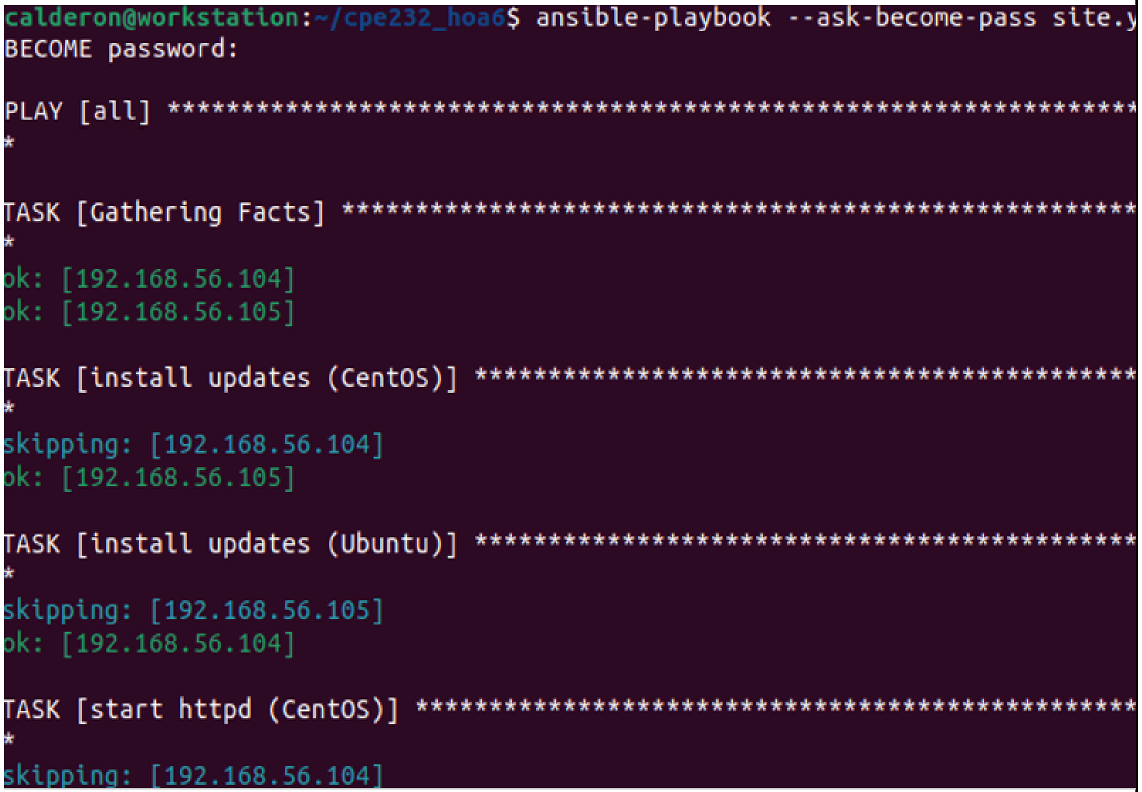
2. Edit the *site.yml* file and just below the *web_servers* play, create a new file to copy the default html file for site:

```
- 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
```

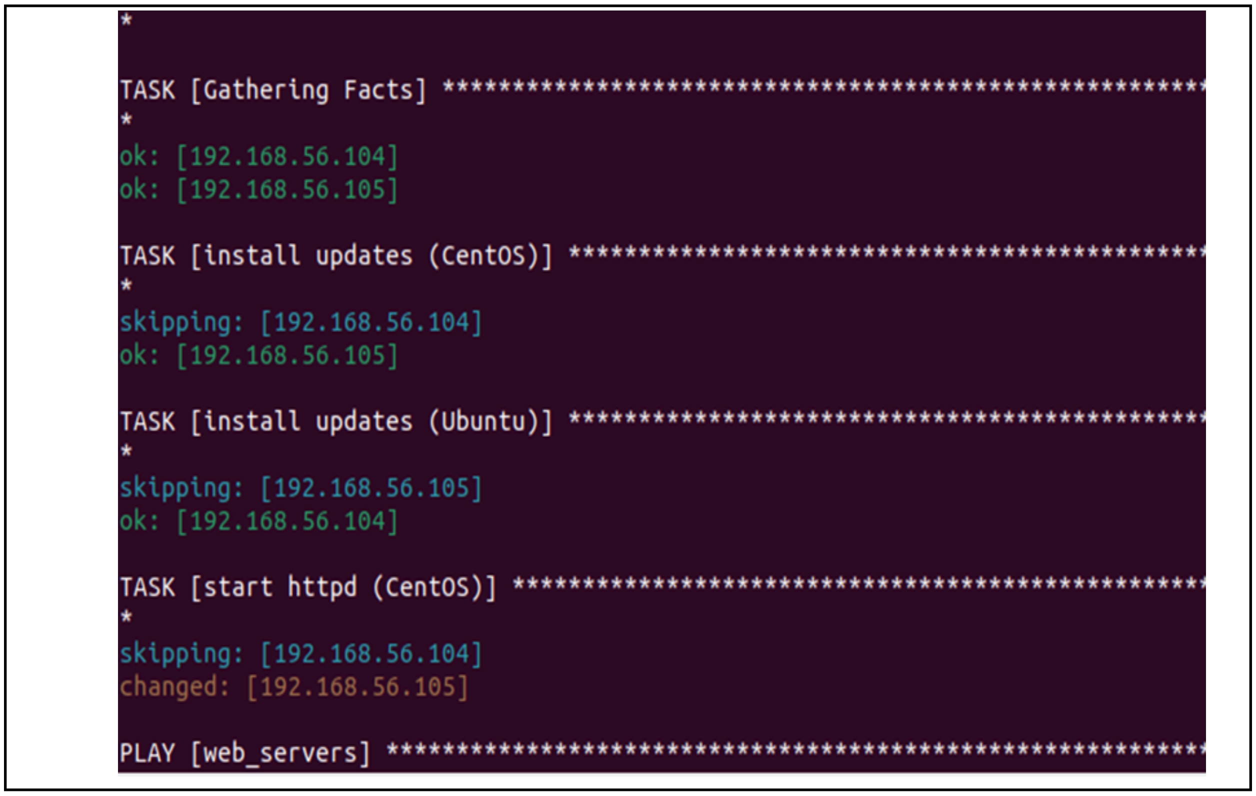


```
  - name: copy default HTML file for site
    tags:
      - apache
      - apache2
      - httpd
    copy:
      src: default_site.html
      dest: /var/www/html/index.html
      owner: root
      group: root
      mode: 0644
```

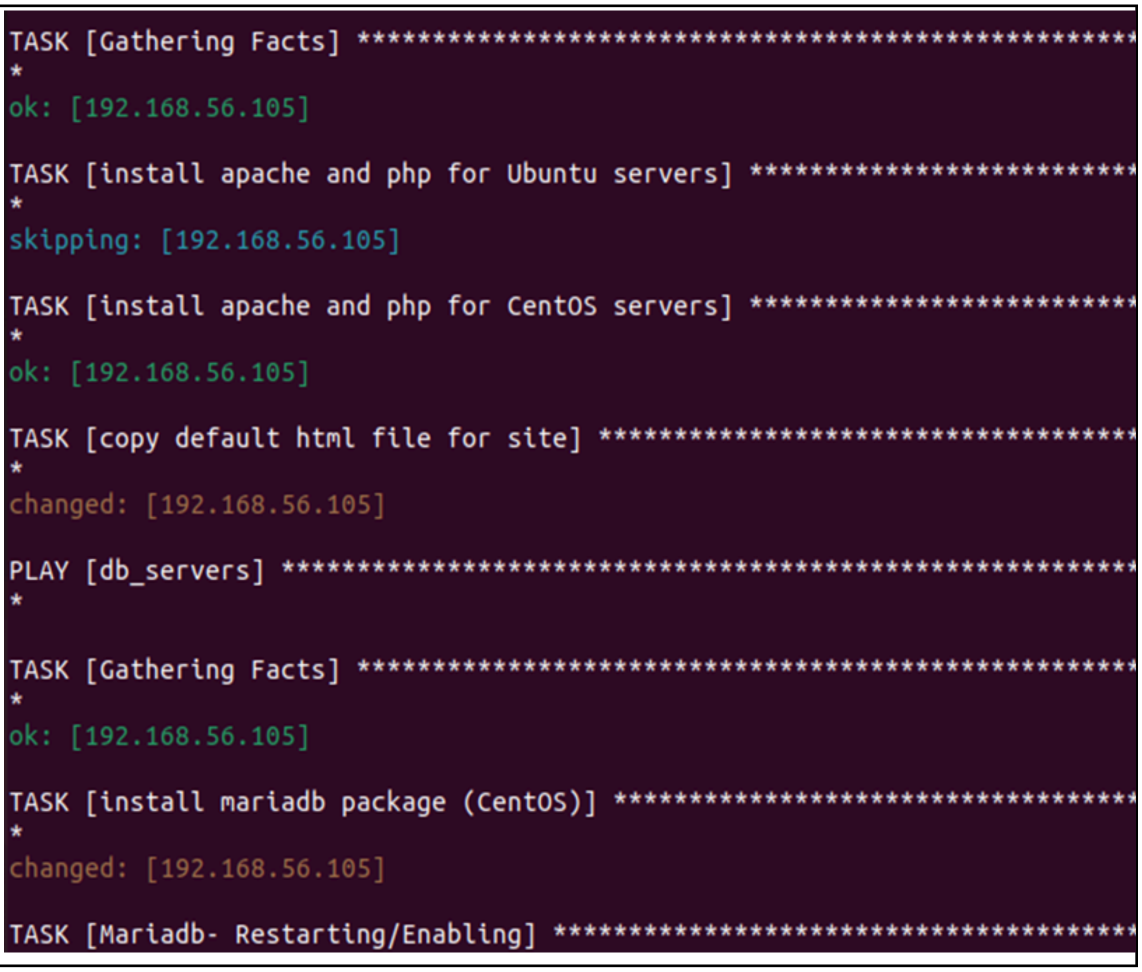
3. Run the playbook *site.yml*. Describe the changes.



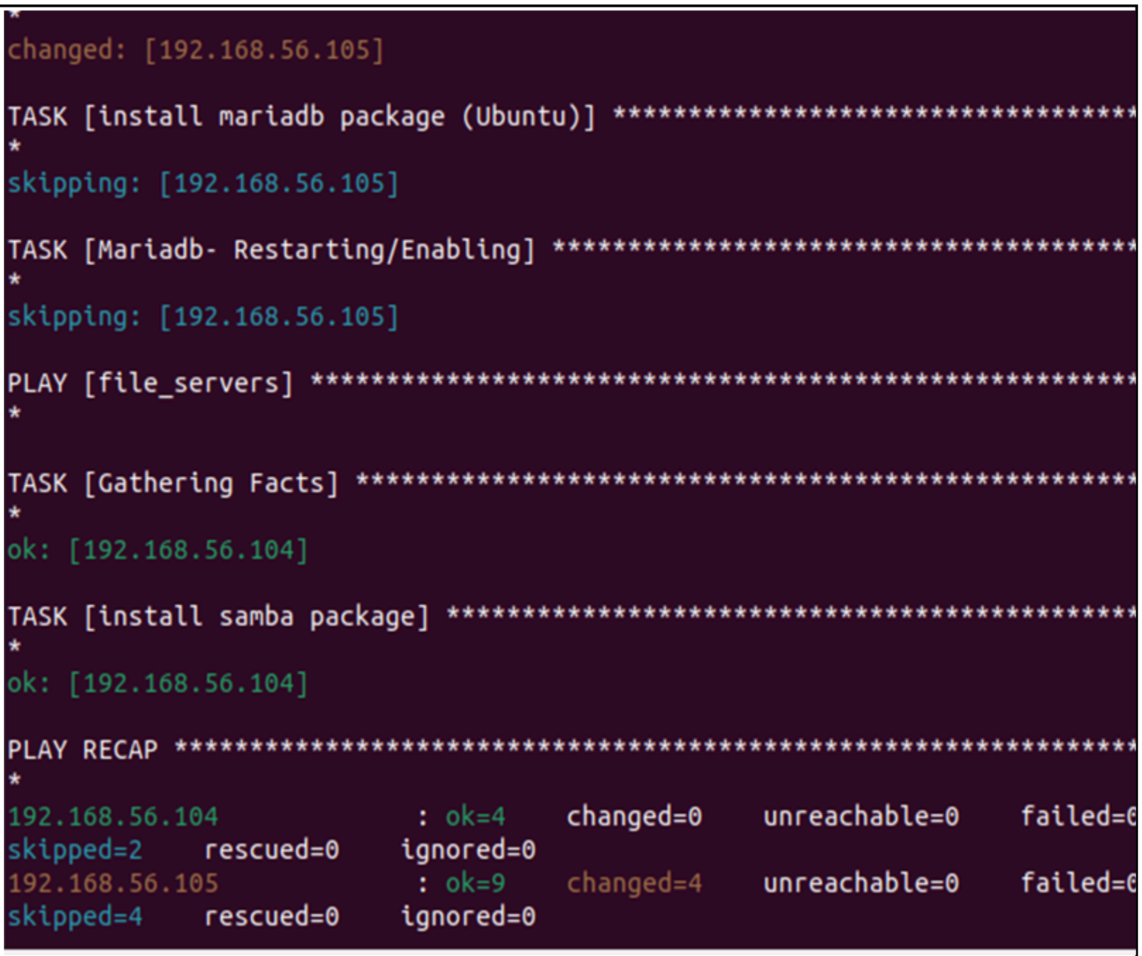
```
calderon@workstation:~/cpe232_hoa6$ ansible-playbook --ask-become-pass site.y
BECOME password:

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

TASK [Gathering Facts] *******************************************************
*
ok: [192.168.56.104]
ok: [192.168.56.105]

TASK [install updates (CentOS)] **********************************************
*
skipping: [192.168.56.104]
ok: [192.168.56.105]

TASK [install updates (Ubuntu)] **********************************************
*
skipping: [192.168.56.105]
ok: [192.168.56.104]

TASK [start httpd (CentOS)] **************************************************
*
skipping: [192.168.56.104]
```



```
*

TASK [Gathering Facts] ******************************************************
*
ok: [192.168.56.104]
ok: [192.168.56.105]

TASK [install updates (CentOS)] *********************************************
*
skipping: [192.168.56.104]
ok: [192.168.56.105]

TASK [install updates (Ubuntu)] *********************************************
*
skipping: [192.168.56.105]
ok: [192.168.56.104]

TASK [start httpd (CentOS)] *************************************************
*
skipping: [192.168.56.104]
changed: [192.168.56.105]

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



```
TASK [Gathering Facts] ****************************************************
*
ok: [192.168.56.105]

TASK [install apache and php for Ubuntu servers] **************************
*
skipping: [192.168.56.105]

TASK [install apache and php for CentOS servers] **************************
*
ok: [192.168.56.105]

TASK [copy default html file for site] ************************************
*
changed: [192.168.56.105]

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

TASK [Gathering Facts] ****************************************************
*
ok: [192.168.56.105]

TASK [install mariadb package (CentOS)] ***********************************
*
changed: [192.168.56.105]

TASK [Mariadb- Restarting/Enabling] ***************************************
```



```
*
changed: [192.168.56.105]

TASK [install mariadb package (Ubuntu)] ***********************************
*
skipping: [192.168.56.105]

TASK [Mariadb- Restarting/Enabling] ***************************************
*
skipping: [192.168.56.105]

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

TASK [Gathering Facts] ****************************************************
*
ok: [192.168.56.104]

TASK [install samba package] **********************************************
*
ok: [192.168.56.104]

PLAY RECAP ****************************************************************
*
192.168.56.104             : ok=4    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.105             : ok=9    changed=4    unreachable=0    failed=0
skipped=4    rescued=0    ignored=0
```

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

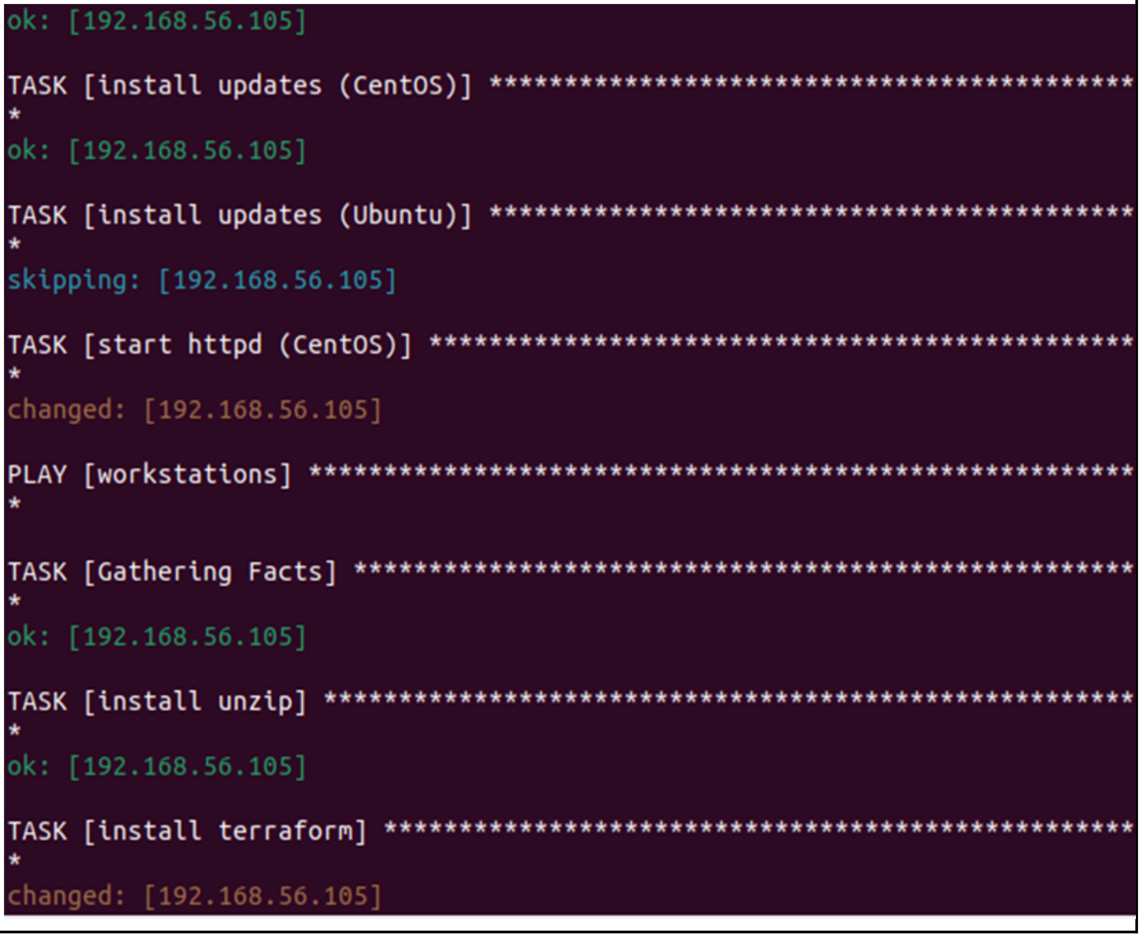
Task 2: Download a file and extract it to a remote server

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

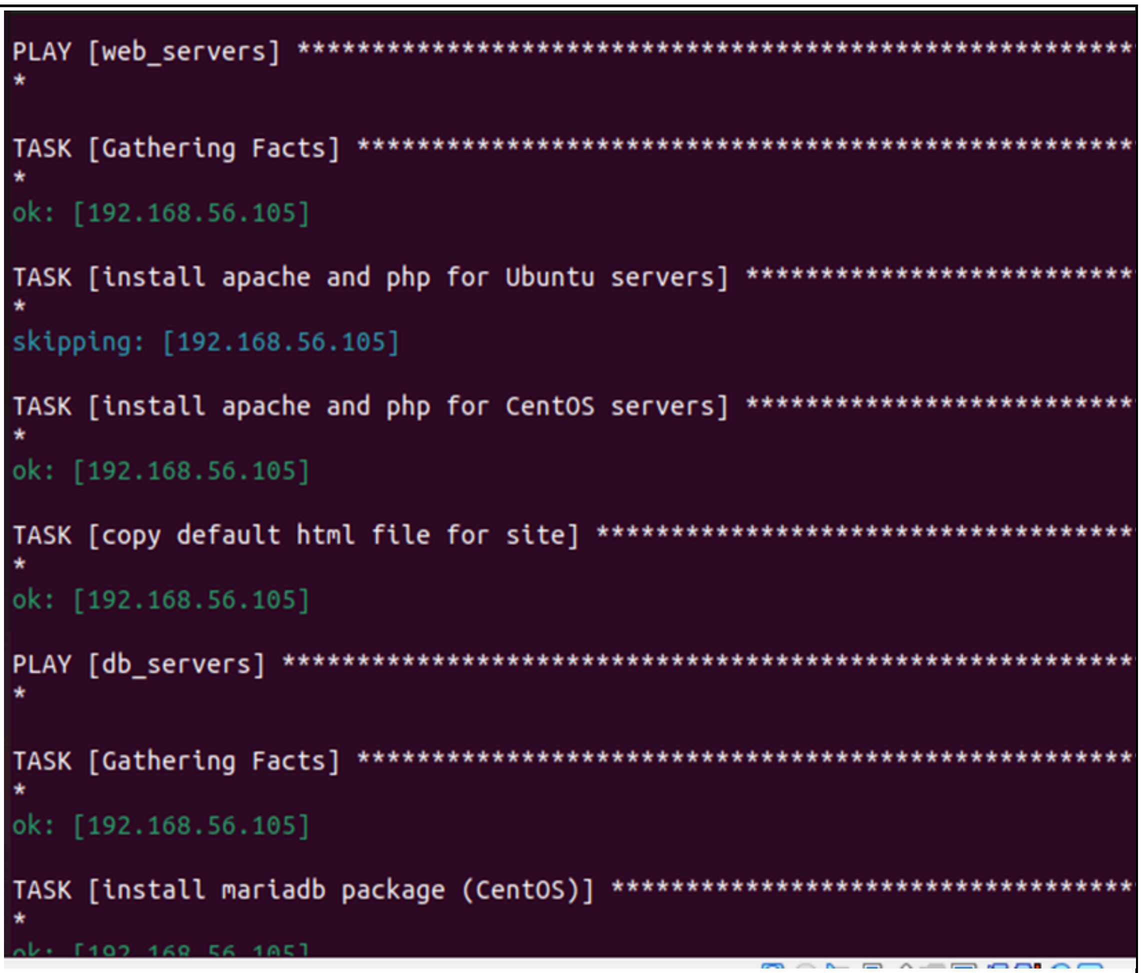
```
- hosts: workstations
  become: true
  tasks:

  - 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
```

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.



```
ok: [192.168.56.105]

TASK [install updates (CentOS)] ********************************************
*
ok: [192.168.56.105]

TASK [install updates (Ubuntu)] ********************************************
*
skipping: [192.168.56.105]

TASK [start httpd (CentOS)] ************************************************
*
changed: [192.168.56.105]

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

TASK [Gathering Facts] *****************************************************
*
ok: [192.168.56.105]

TASK [install unzip] *******************************************************
*
ok: [192.168.56.105]

TASK [install terraform] ***************************************************
*
changed: [192.168.56.105]
```

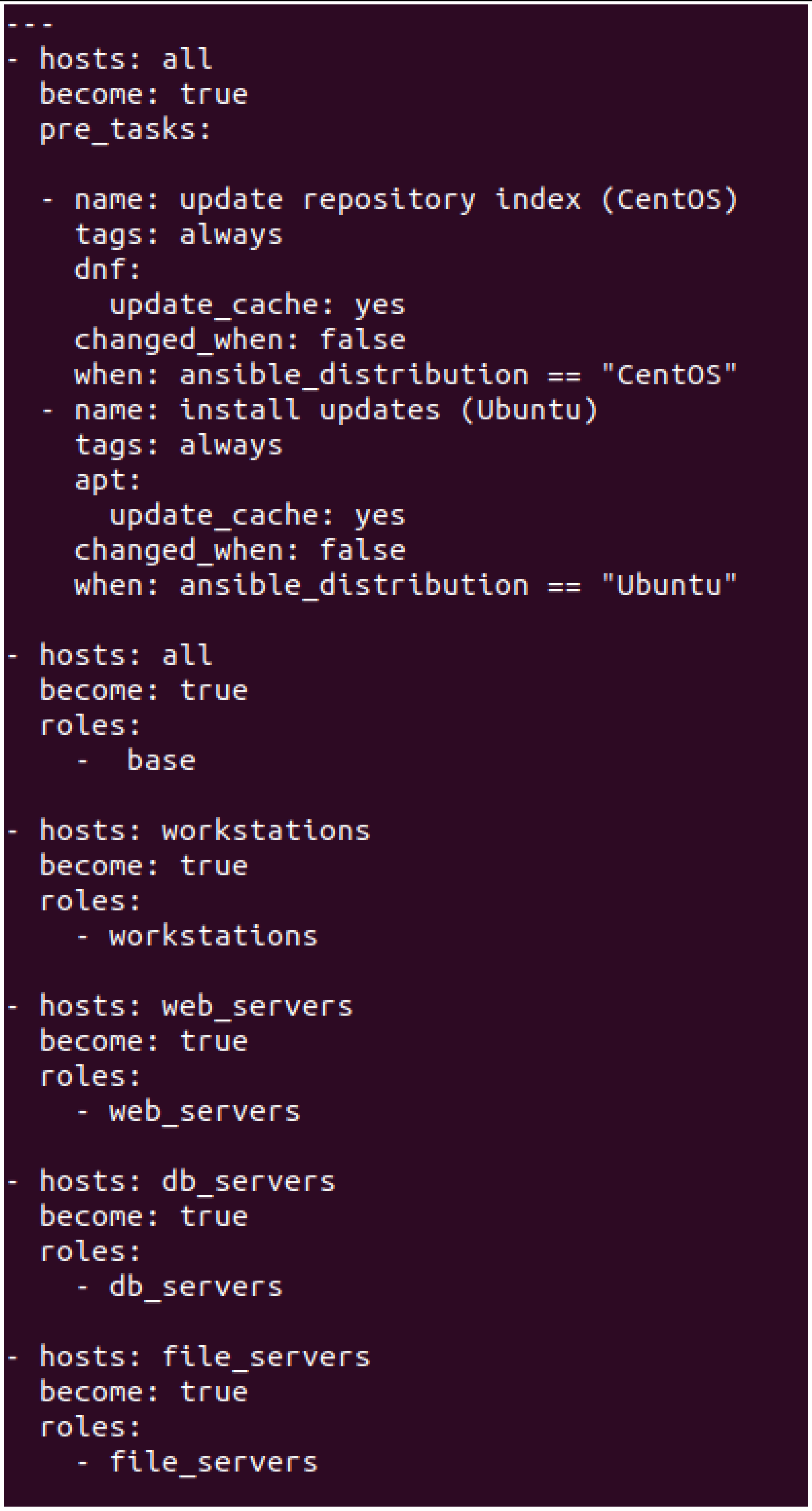


```
PLAY [web_servers] ***********************************************************
*

TASK [Gathering Facts] *******************************************************
*
ok: [192.168.56.105]

TASK [install apache and php for Ubuntu servers] *****************************
*
skipping: [192.168.56.105]

TASK [install apache and php for CentOS servers] *****************************
*
ok: [192.168.56.105]

TASK [copy default html file for site] ***************************************
*
ok: [192.168.56.105]

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

TASK [Gathering Facts] *******************************************************
*
ok: [192.168.56.105]

TASK [install mariadb package (CentOS)] **************************************
*
ok: [192.168.56.105]
```

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

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)

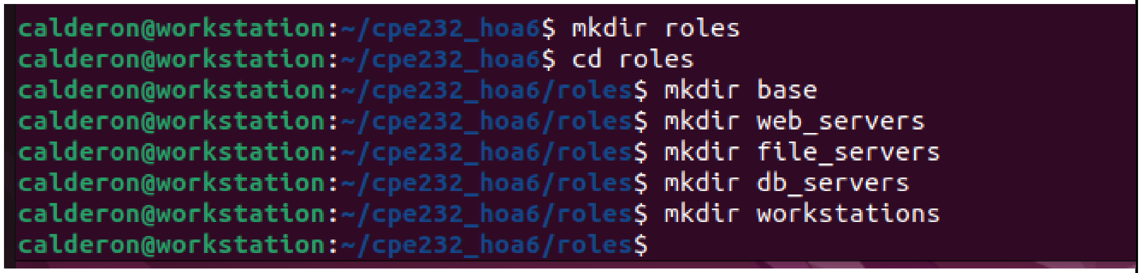


```
---
- hosts: all
  become: true
  pre_tasks:

  - 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.

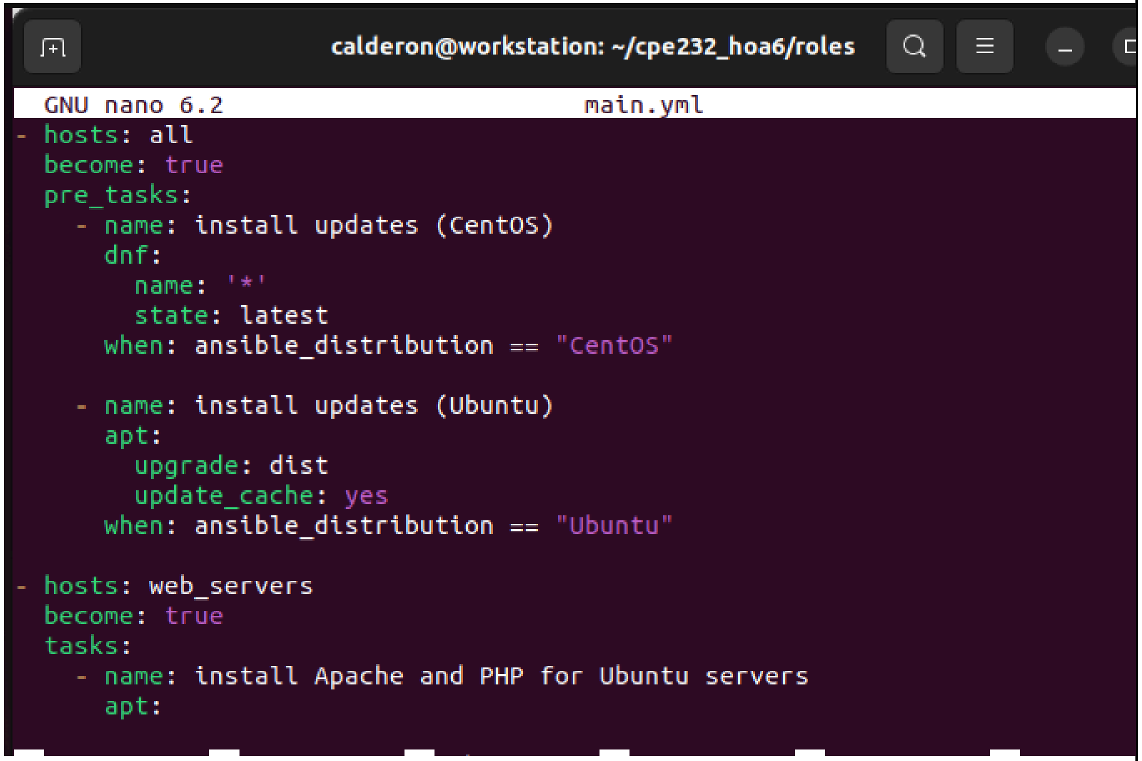
2. Under the same directory, create a new directory and name it roles. Enter the roles directory and create new directories: base, web_servers, file_servers,

db_servers and workstations. For each directory, create a directory and name it tasks.



```
calderon@workstation:~/cpe232_hoa6$ mkdir roles
calderon@workstation:~/cpe232_hoa6$ cd roles
calderon@workstation:~/cpe232_hoa6/roles$ mkdir base
calderon@workstation:~/cpe232_hoa6/roles$ mkdir web_servers
calderon@workstation:~/cpe232_hoa6/roles$ mkdir file_servers
calderon@workstation:~/cpe232_hoa6/roles$ mkdir db_servers
calderon@workstation:~/cpe232_hoa6/roles$ mkdir workstations
calderon@workstation:~/cpe232_hoa6/roles$
```

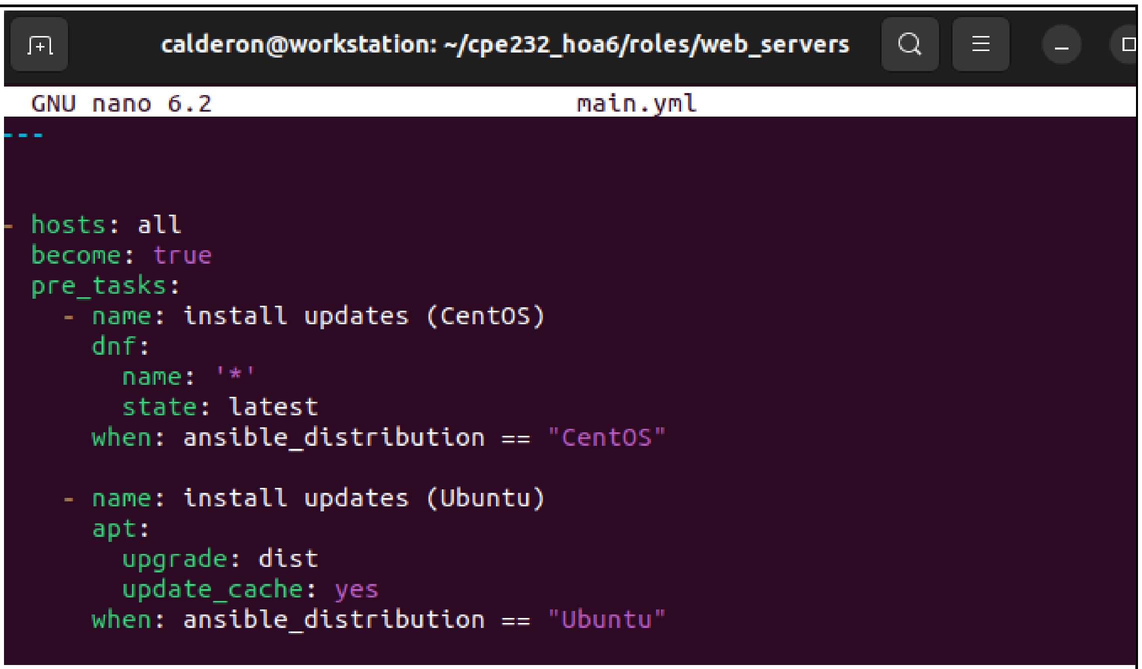
3. Go to tasks for all directory and create a file. Name it main.yml. In each of the tasks for all directories, copy and paste the code from the old site.yml file. Show all contents of main.yml files for all tasks.



calderon@workstation: ~/cpe232_hoa6/roles

```
  GNU nano 6.2                          main.yml
- hosts: all
  become: true
  pre_tasks:
    - name: install updates (CentOS)
      dnf:
        name: '*'
        state: latest
      when: ansible_distribution == "CentOS"

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

- hosts: web_servers
  become: true
  tasks:
    - name: install Apache and PHP for Ubuntu servers
      apt:
```

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



```
---


- hosts: all
  become: true
  pre_tasks:
    - name: install updates (CentOS)
      dnf:
        name: '*'
        state: latest
      when: ansible_distribution == "CentOS"

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

Reflections:

Answer the following:

1. What is the importance of creating roles?

   Creating roles is important in Ansible because it enables you to organize and share your code more effectively. Roles are a way of logically grouping related tasks, files, and variables, making it easier to manage and maintain your Ansible code. Roles also help in simplifying the deployment process by allowing you to reuse code across different projects and environments.

2. What is the importance of managing files?

   Managing files is important in any system, including Ansible, for several reasons:

   1. Maintain consistency: Managing files allows you to maintain consistency across different systems.

   2. Increase efficiency: With file management, you can automate tasks that would otherwise need manual intervention.

   3. Enhance security: Managing files can help to enhance the security of your system. You can use file permissions and access controls to ensure that only authorized users have access to specific files and folders.

Conclusion

Ansible roles organize and manage code into reusable and shareable components for streamlined automation. By using roles, you can improve efficiency and consistency while simplifying collaboration. Additionally, Ansible's file management features allow for automated file-related tasks to enhance the security, efficiency, and disaster recovery capabilities of your systems. In summary, these are important tools for effective automation and systems management.