Name: Bernardo, Christian Emmanuel	**Date Performed: 12/03/24**
Course/Section: CPE232 - CPE31S1	**Date Submitted: 12/03/24**
Instructor: Dr. Jonathan Taylar	**Semester and SY: 2nd. , 2023-2024**

Activity 7: Managing Files and Creating Roles in Ansible

1. **Objectives:**

1.1 Manage files in remote servers

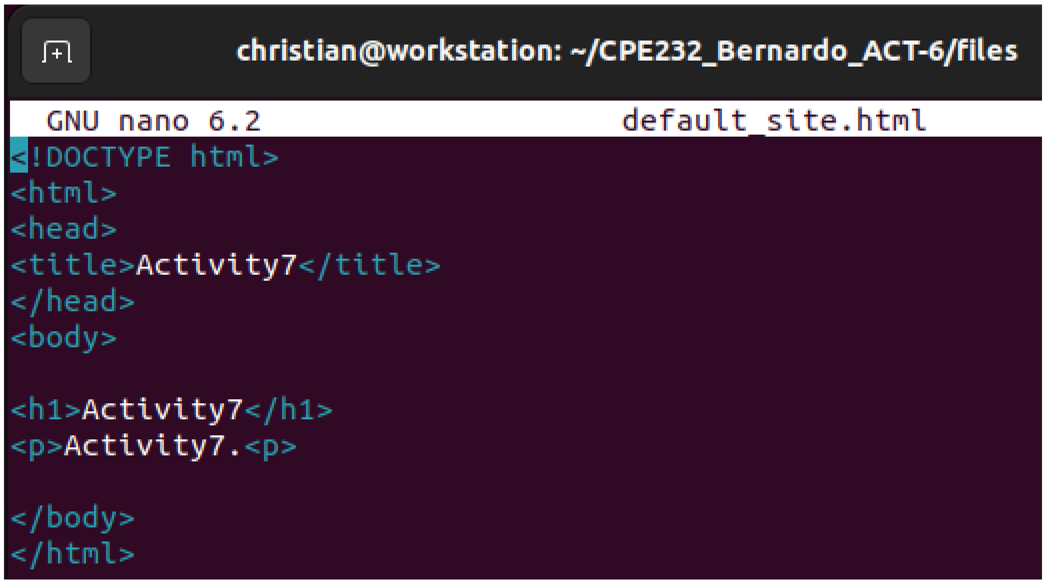
1.2 Implement roles in ansible

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.



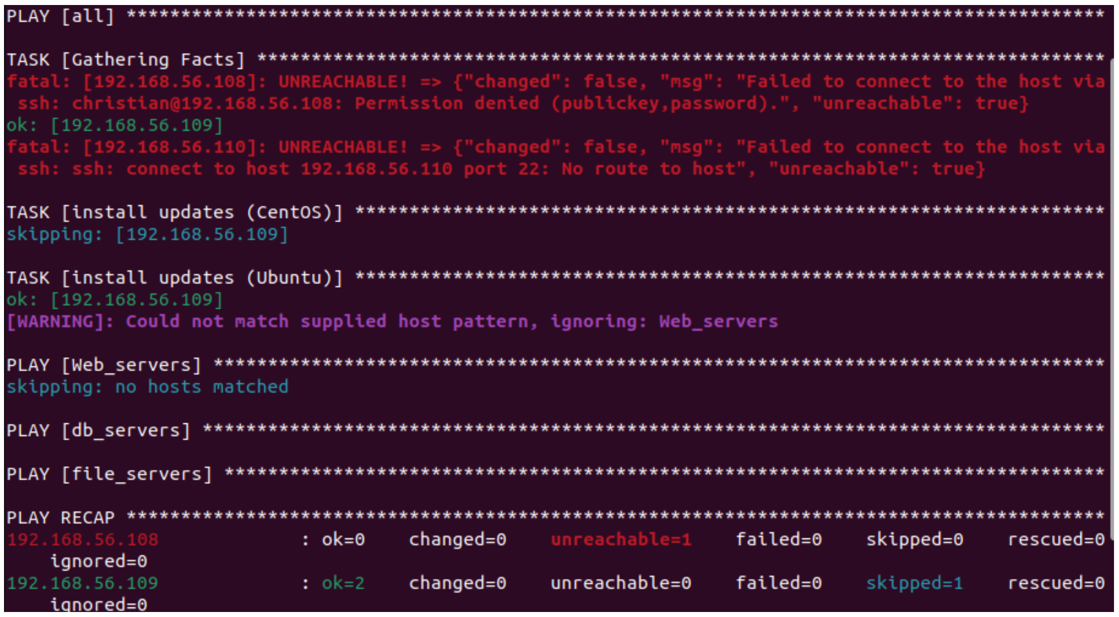
```
christian@workstation: ~/CPE232_Bernardo_ACT-6/files

  GNU nano 6.2                        default_site.html
<!DOCTYPE html>
<html>
<head>
<title>Activity7</title>
</head>
<body>

<h1>Activity7</h1>
<p>Activity7.<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:
```

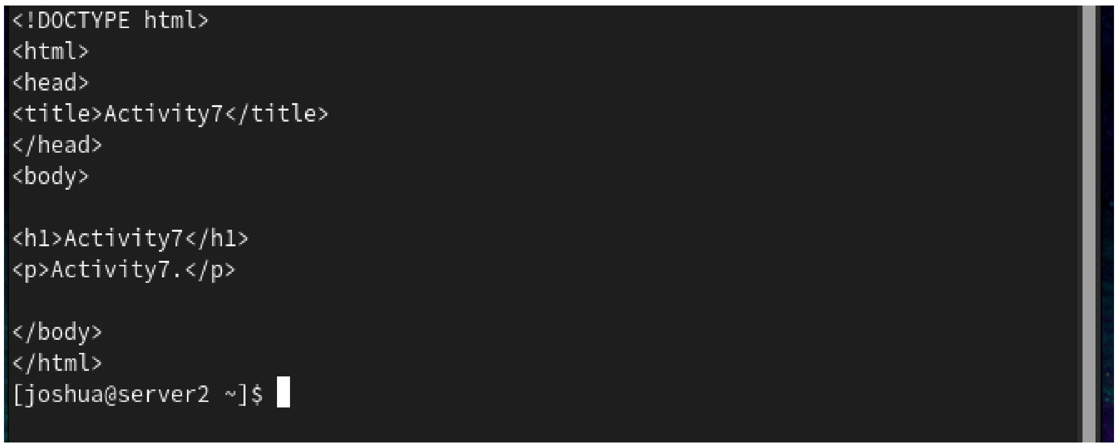
src: default_site.html
dest: /var/www/html/index.html
owner: root
group: root
mode: 0644

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



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

TASK [Gathering Facts] *******************************************************************
fatal: [192.168.56.108]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via
 ssh: christian@192.168.56.108: Permission denied (publickey,password).", "unreachable": true}
ok: [192.168.56.109]
fatal: [192.168.56.110]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via
 ssh: ssh: connect to host 192.168.56.110 port 22: No route to host", "unreachable": true}

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

TASK [install updates (Ubuntu)] **********************************************************
ok: [192.168.56.109]
[WARNING]: Could not match supplied host pattern, ignoring: Web_servers

PLAY [Web_servers] ***********************************************************************
skipping: no hosts matched

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

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

PLAY RECAP *******************************************************************************
192.168.56.108             : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0
    ignored=0
192.168.56.109             : ok=2    changed=0    unreachable=0    failed=0    skipped=1    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.



```
<!DOCTYPE html>
<html>
<head>
<title>Activity7</title>
</head>
<body>

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

</body>
</html>
[joshua@server2 ~]$ 
```

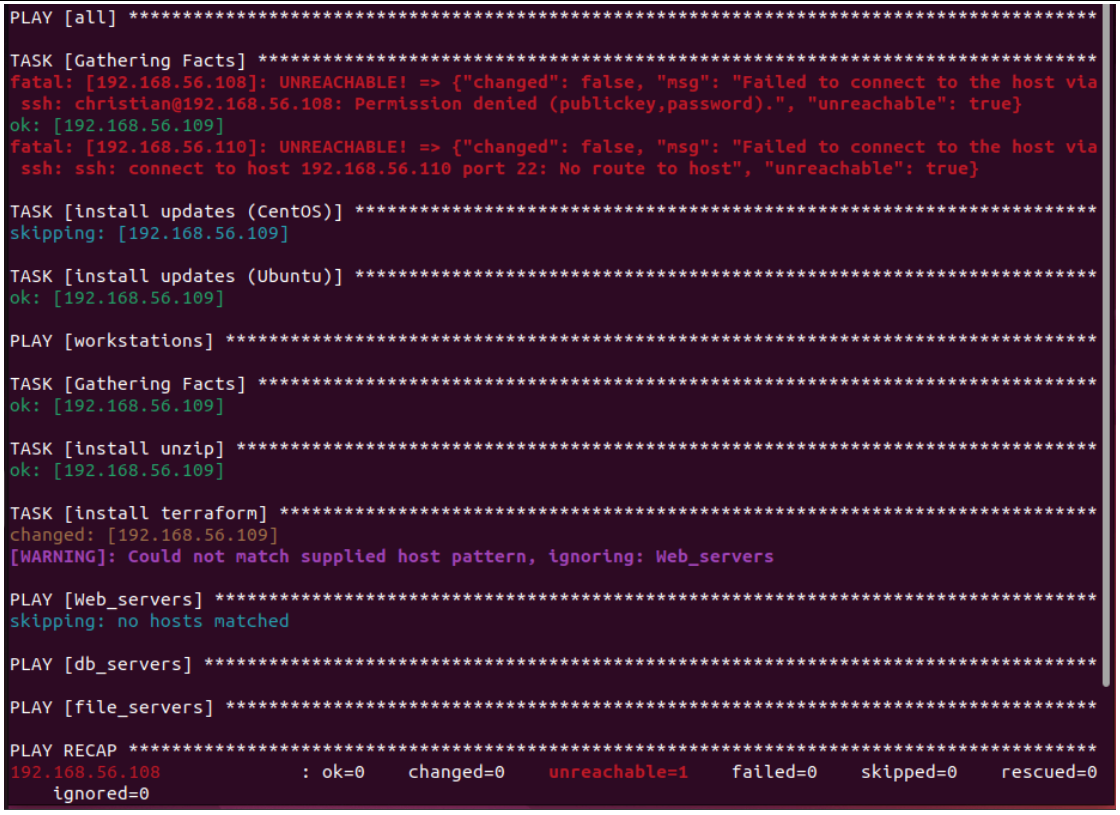
5. Sync your local repository with GitHub and describe the changes.

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:

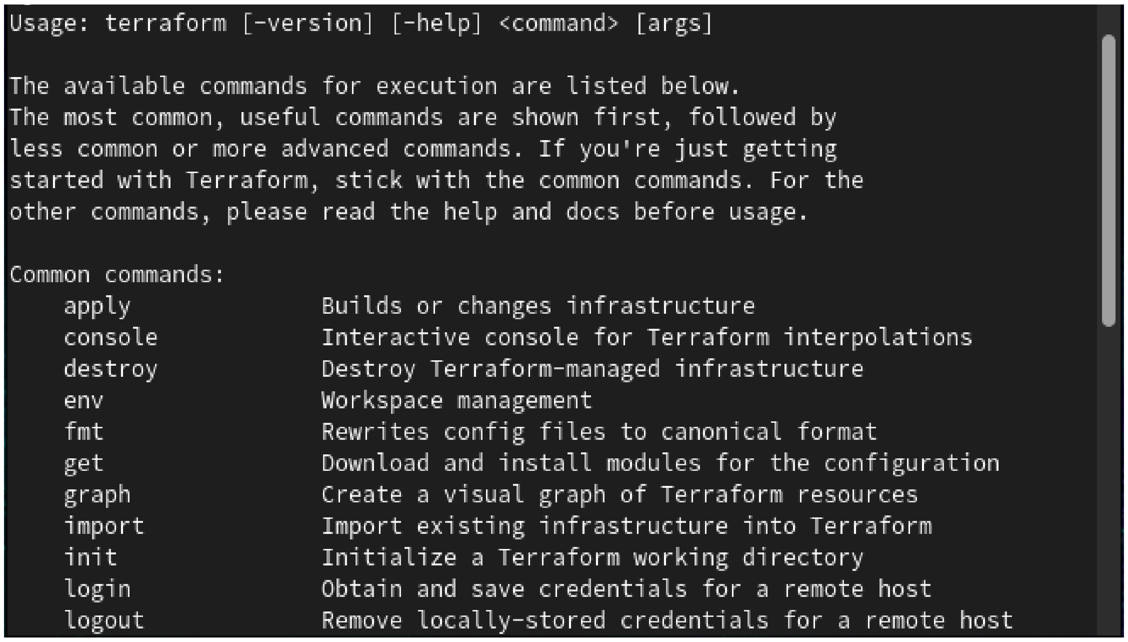
```
- hosts: workstations
  become: true
  tasks:

  - 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.



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

TASK [Gathering Facts] ***********************************************************************
fatal: [192.168.56.108]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via
 ssh: christian@192.168.56.108: Permission denied (publickey,password).", "unreachable": true}
ok: [192.168.56.109]
fatal: [192.168.56.110]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via
 ssh: ssh: connect to host 192.168.56.110 port 22: No route to host", "unreachable": true}

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

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

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

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

TASK [install unzip] *************************************************************************
ok: [192.168.56.109]

TASK [install terraform] *********************************************************************
changed: [192.168.56.109]
[WARNING]: Could not match supplied host pattern, ignoring: Web_servers

PLAY [Web_servers] ***************************************************************************
skipping: no hosts matched

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

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

PLAY RECAP ***********************************************************************************
192.168.56.108             : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0
    ignored=0
```

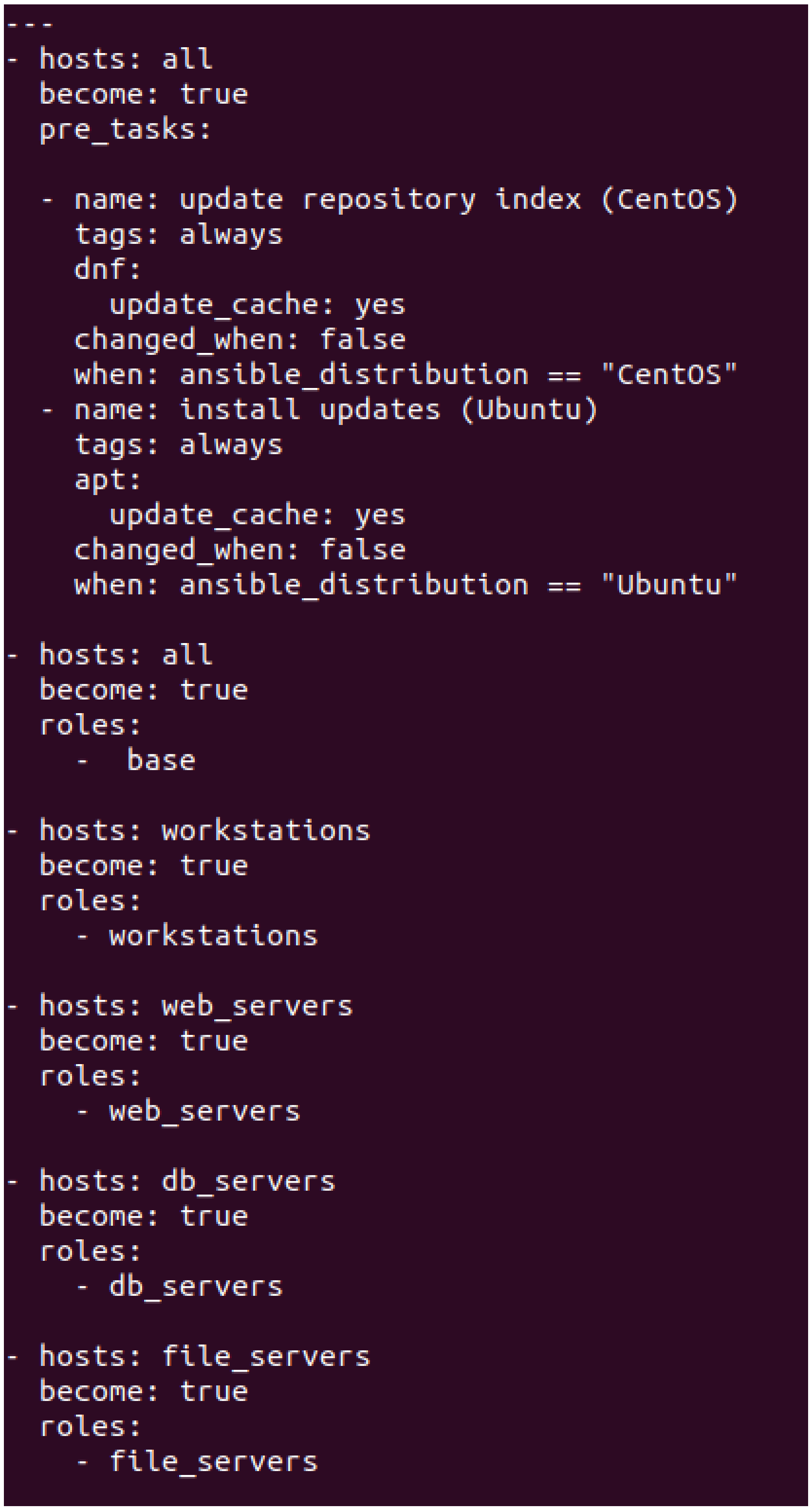
4. On the Ubuntu remote workstation, type terraform to verify installation of terraform. Describe the output.



```
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

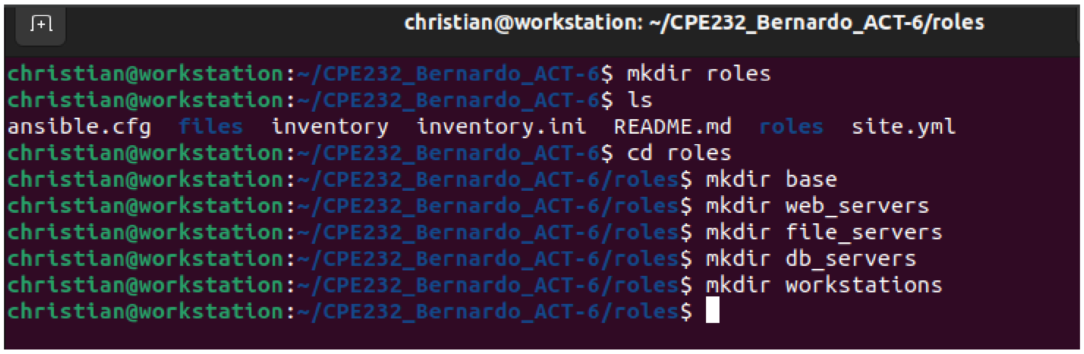
1. Edit the site.yml. Configure roles as follows: (make sure to create a copy of the old site.yml file because you will be copying the specific plays for all groups)



```
---
- 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
```

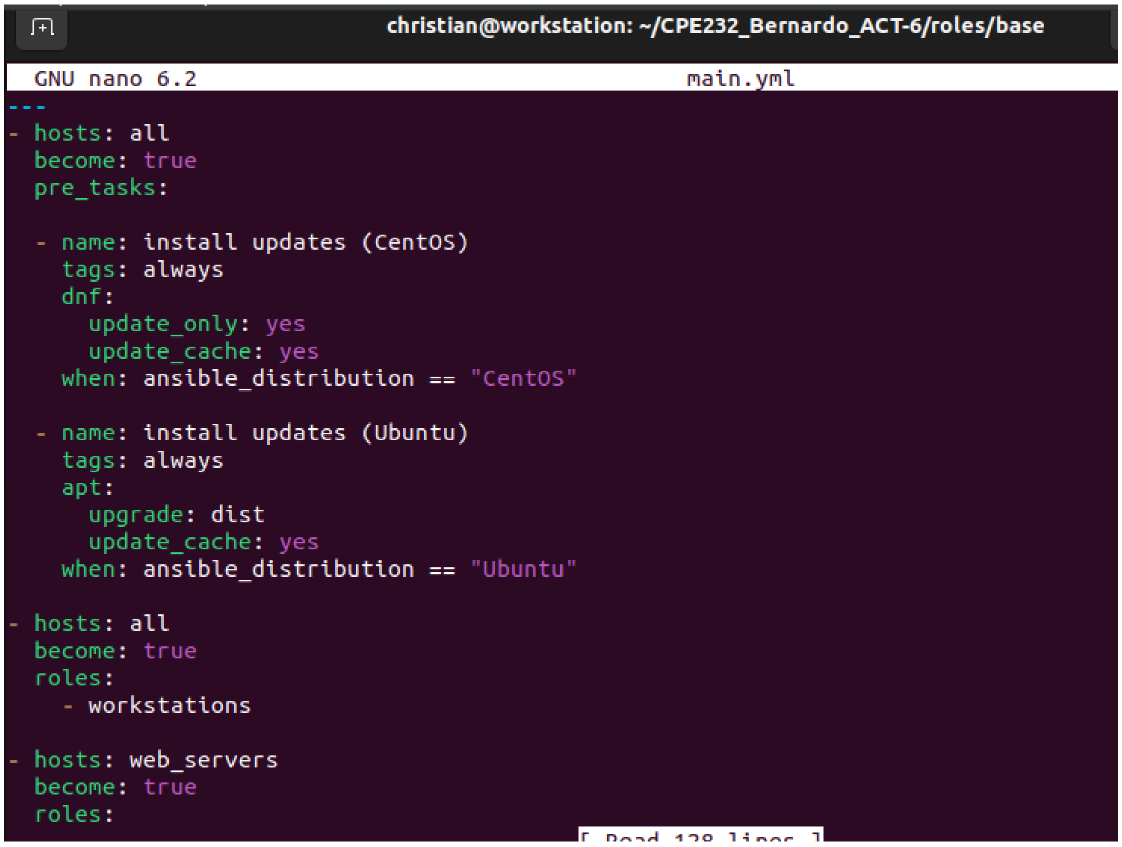
Save the file and exit.

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.

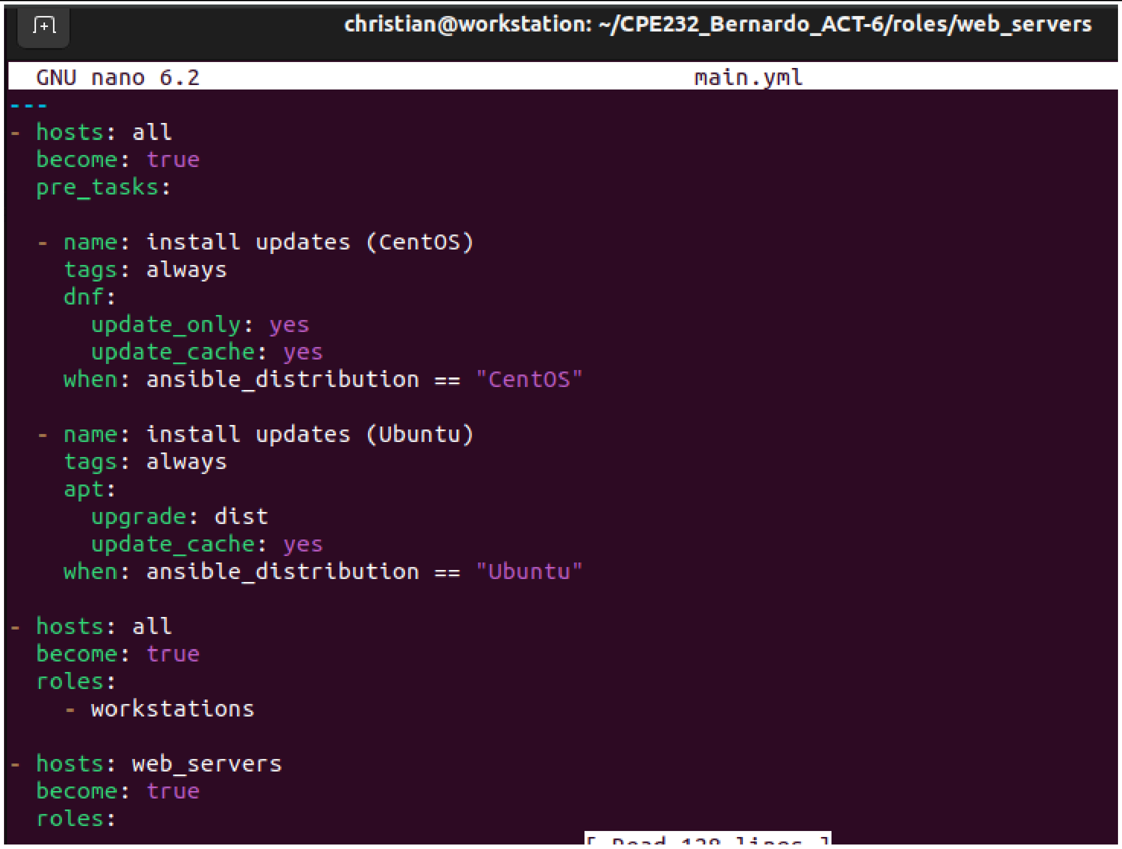


```
christian@workstation: ~/CPE232_Bernardo_ACT-6/roles

christian@workstation:~/CPE232_Bernardo_ACT-6$ mkdir roles
christian@workstation:~/CPE232_Bernardo_ACT-6$ ls
ansible.cfg  files  inventory  inventory.ini  README.md  roles  site.yml
christian@workstation:~/CPE232_Bernardo_ACT-6$ cd roles
christian@workstation:~/CPE232_Bernardo_ACT-6/roles$ mkdir base
christian@workstation:~/CPE232_Bernardo_ACT-6/roles$ mkdir web_servers
christian@workstation:~/CPE232_Bernardo_ACT-6/roles$ mkdir file_servers
christian@workstation:~/CPE232_Bernardo_ACT-6/roles$ mkdir db_servers
christian@workstation:~/CPE232_Bernardo_ACT-6/roles$ mkdir workstations
christian@workstation:~/CPE232_Bernardo_ACT-6/roles$
```

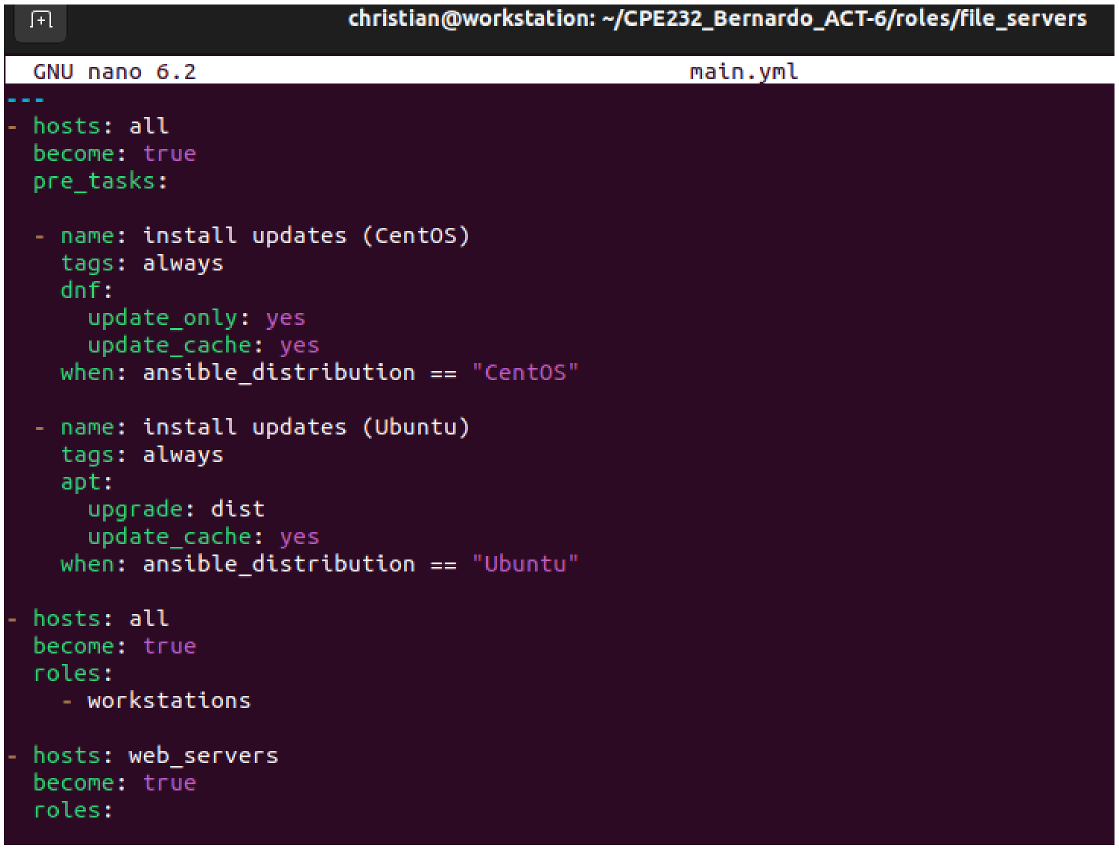
3. Go to tasks for all directory and create a file. Name it main.yml. In each of the tasks for all directories, copy and paste the code from the old site.yml file. Show all contents of main.yml files for all tasks.



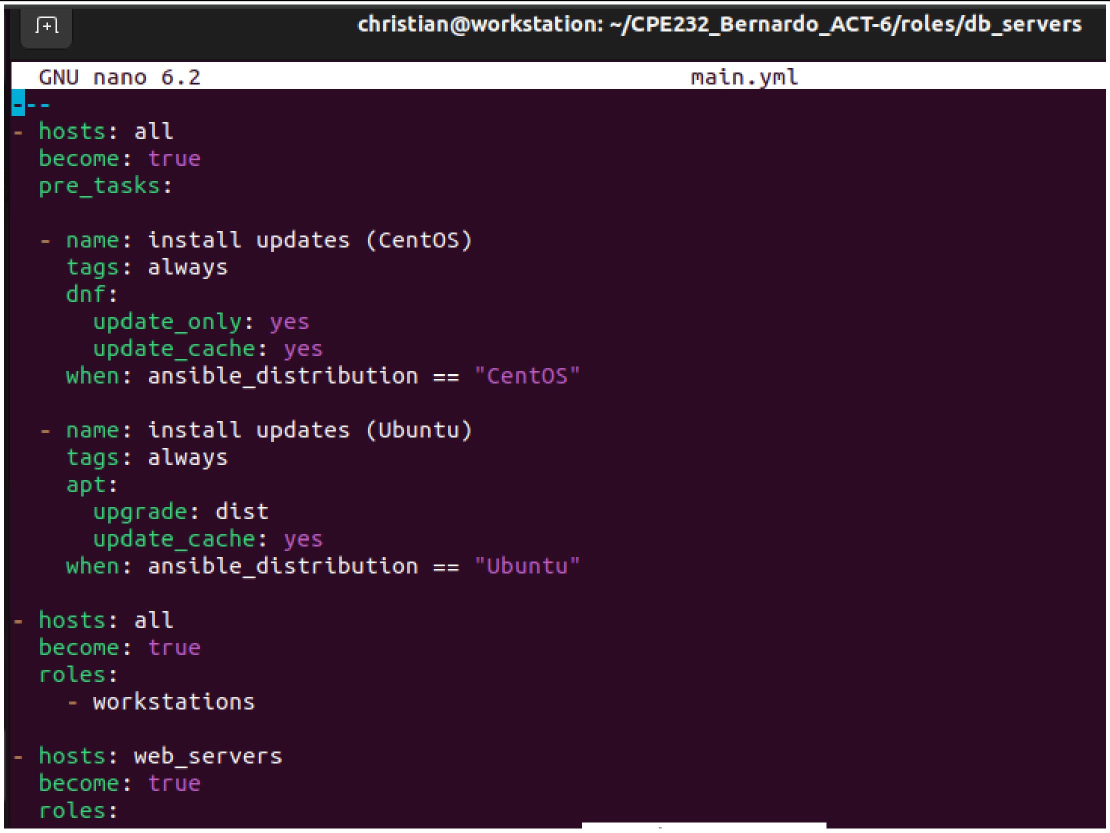
```
christian@workstation: ~/CPE232_Bernardo_ACT-6/roles/base

 GNU nano 6.2                                   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"

- hosts: all
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
                                            [ Read 128 lines ]
```



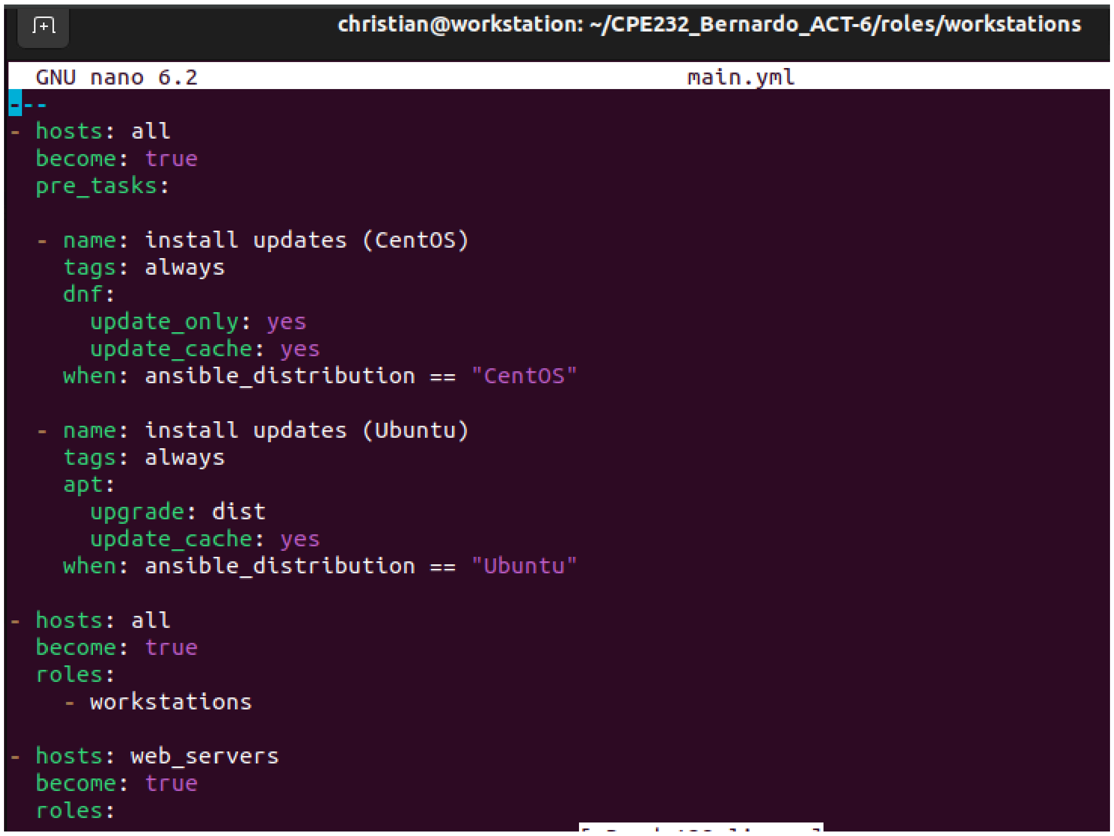
```
christian@workstation: ~/CPE232_Bernardo_ACT-6/roles/web_servers
 GNU nano 6.2                                      main.yml
---
- hosts: all
  become: true
  pre_tasks:

  - 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"

- hosts: all
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
```



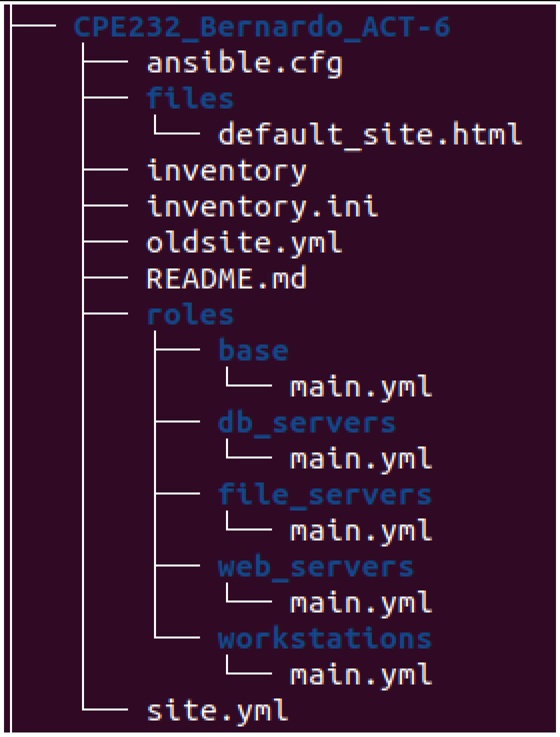
```
christian@workstation: ~/CPE232_Bernardo_ACT-6/roles/file_servers
 GNU nano 6.2                                      main.yml
---
- hosts: all
  become: true
  pre_tasks:

  - 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"

- hosts: all
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
```



```
christian@workstation: ~/CPE232_Bernardo_ACT-6/roles/db_servers
GNU nano 6.2                                              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"

- hosts: all
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
```

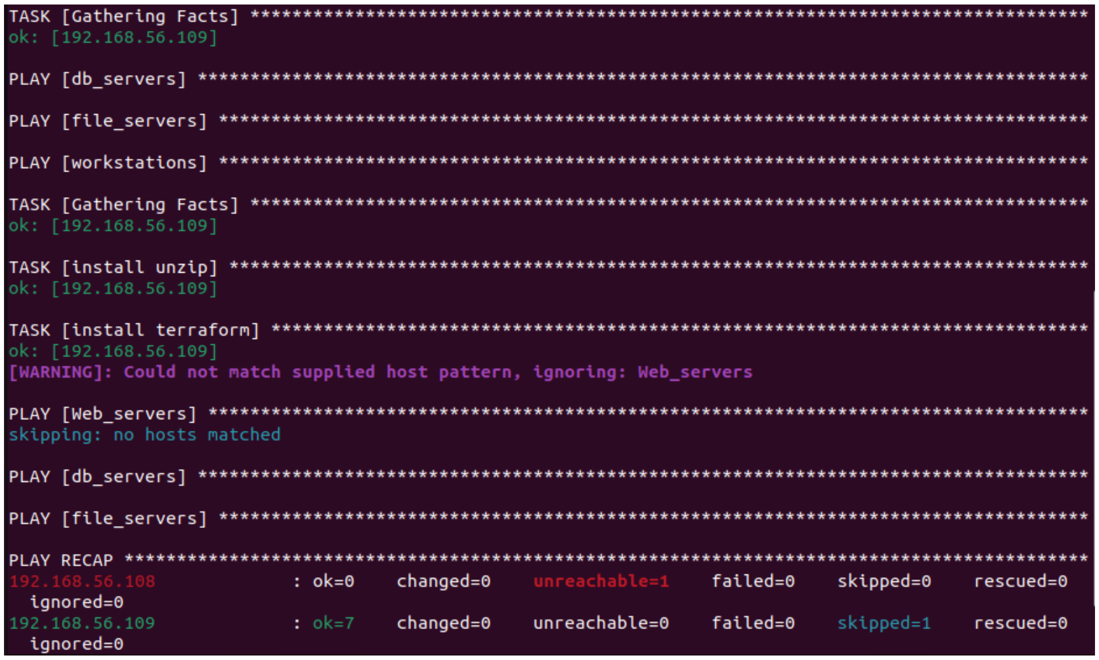


```
christian@workstation: ~/CPE232_Bernardo_ACT-6/roles/workstations
GNU nano 6.2                                              main.yml
---
- hosts: all
  become: true
  pre_tasks:

  - 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"

- hosts: all
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
```



```
├── CPE232_Bernardo_ACT-6
    ├── ansible.cfg
    ├── files
    │   └── default_site.html
    ├── inventory
    ├── inventory.ini
    ├── oldsite.yml
    ├── README.md
    ├── roles
    │   ├── base
    │   │   └── main.yml
    │   ├── db_servers
    │   │   └── main.yml
    │   ├── file_servers
    │   │   └── main.yml
    │   ├── web_servers
    │   │   └── main.yml
    │   └── workstations
    │       └── main.yml
    └── site.yml
```

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



```
TASK [Gathering Facts] ******************************************************************************
ok: [192.168.56.109]

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

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

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

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

TASK [install unzip] ********************************************************************************
ok: [192.168.56.109]

TASK [install terraform] ****************************************************************************
ok: [192.168.56.109]
[WARNING]: Could not match supplied host pattern, ignoring: Web_servers

PLAY [Web_servers] **********************************************************************************
skipping: no hosts matched

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

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

PLAY RECAP ******************************************************************************************
192.168.56.108             : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0
  ignored=0
192.168.56.109             : ok=7    changed=0    unreachable=0    failed=0    skipped=1    rescued=0
  ignored=0
```

Reflections:

Answer the following:

1. What is the importance of creating roles?

- Creating roles is important because it is very reusable and easy to give maintenance when or if the code gets corrupted or gives an error so it will be easier to fix individually than all together at once.

2. What is the importance of managing files?

- It is important for software deployment to install and update many things all at once or one at a time. you can even install specific versions when it is wanted or needed. and the most important is to give privacy very important to make sure that the only people can access to the file is with the right verifications.

Conclusion:

- In this activity I learned how to make roles and make management of files much easier and faster to use. I learned how to make things much easier when I need to manage files and make it more secure to use and manage it if or when it gets corrupted to fix make individually.