Name: Carag, Carl Jervie B.	Date Performed: 12/10/25
Course/Section: CPE31S2	Date Submitted: 15/10/25
Instructor: Engr. Robin Valenzuela	Semester and SY: 1st Sem 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.
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.

CpEMidterm

inventory.ini | new.yml | site.yml | default_site.html

site.yml

```
2
3   - hosts: db_servers
4     become: true
5     tasks:
6     - name: copy default html file for site
7       tags: apache, apache2, httpd
8       copy:
9         src: default_site.html
10        dest: /var/www/html/index.html
11        owner: root
12        group: root
13        mode: 0644
14
15    - name: install samba package
16      package:
17        name: samba
18        state: latest
```

PROBLEMS OUTPUT TERMINAL PORTS bash

```
carl@Carl:~/CpEMidterm$ ansible-playbook site.yml -K
ok: [192.168.56.101]

TASK [copy default html file for site] ****************************
**************
changed: [192.168.56.101]

TASK [install samba package] **************************************
**************
ok: [192.168.56.101]

PLAY RECAP ********************************************************
**************
192.168.56.101             : ok=3    changed=1    unreachable=0
 failed=0    skipped=0    rescued=0    ignored=0

carl@Carl:~/CpEMidterm$
```

Output: The copying of the html file for site works and there is changes that occurred.

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

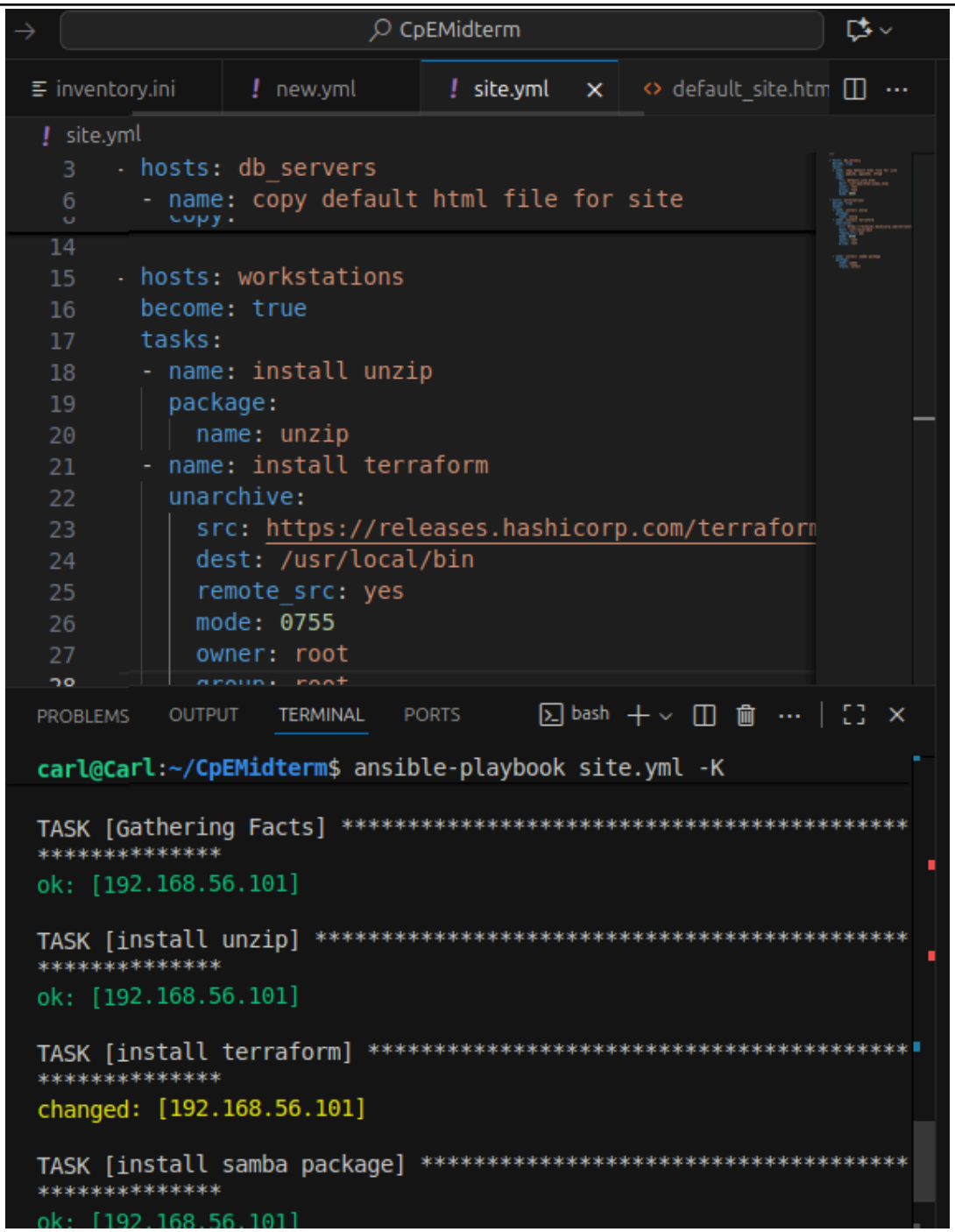
5. Sync your local repository with GitHub and describe the changes.

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.



Observation: It successfully installed the terraform indicating changes in the ip address of my manage node.

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

```
carl@Carl:~$ terraform --version
Terraform v0.12.28
carl@Carl:~$
```

Output: It shows that terraform was successfully downloaded and we can see it once we reviewed the version of the terraform.

Task 3: Create roles

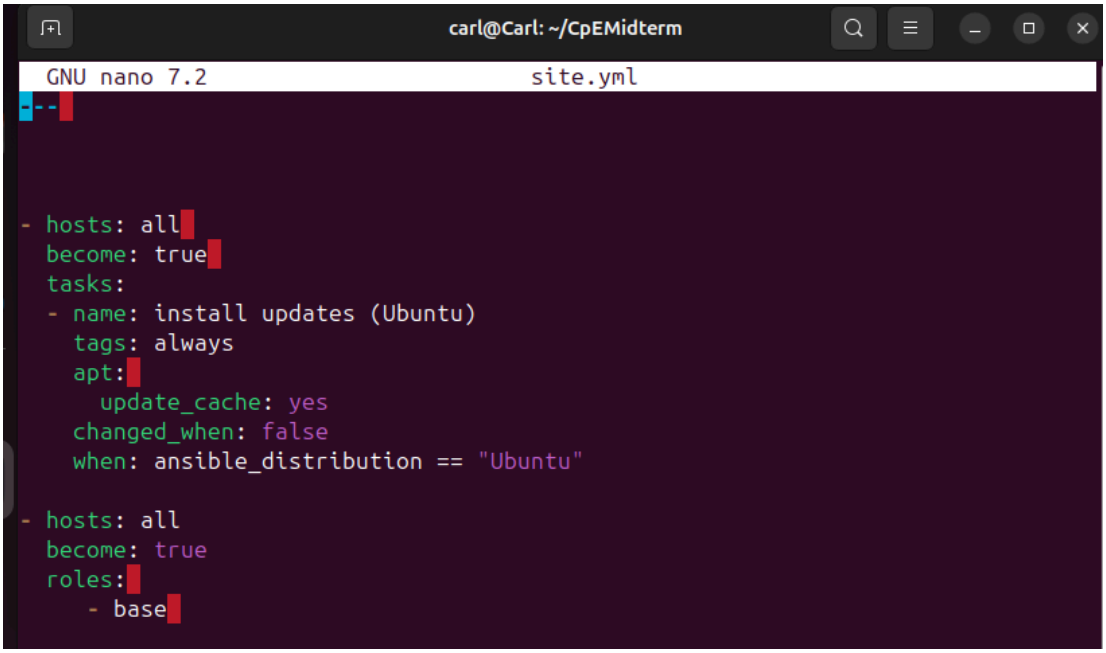
1. Edit the site.yml. Configure roles as follows: (make sure to create a copy of the old site.yml file because you will be copying the specific plays for all groups)

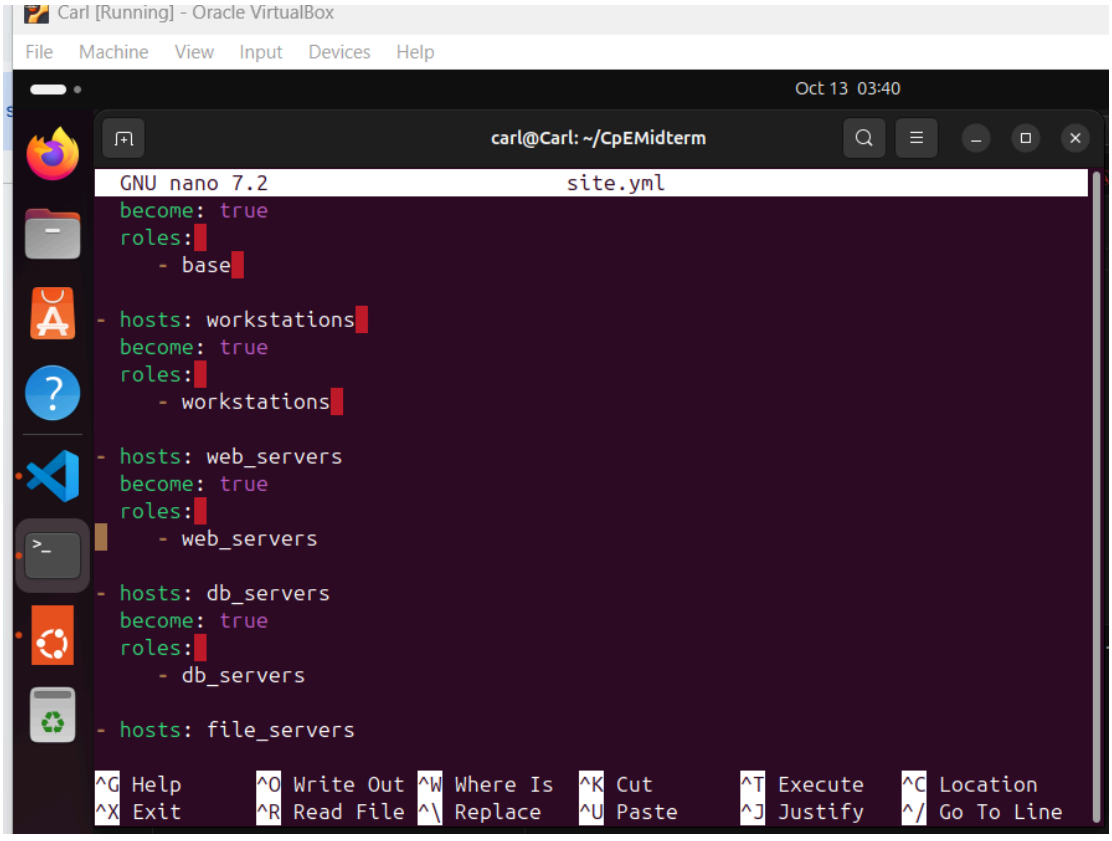
```
---
- hosts: all
  become: true
  pre_tasks:

  - 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,

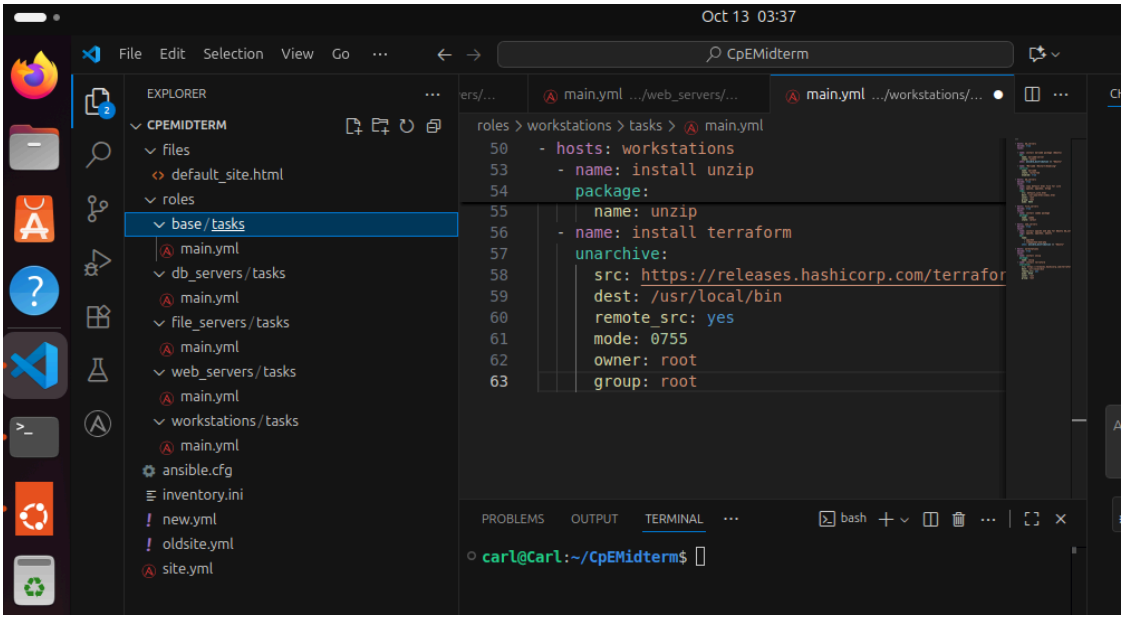
db_servers and workstations. For each directory, create a directory and name it tasks.





```
carl@Carl:~/CpEMidterm/roles$ ls
base  db_servers  file_servers  web_servers  workstations
carl@Carl:~/CpEMidterm/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.



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

Reflections:

Answer the following:

1. What is the importance of creating roles?
   - It allows specific installation of the files.
2. What is the importance of managing files?
   - Managing files, both physical and digital, is crucial because it directly impacts **productivity, data security, collaboration, and efficiency** for individuals and organizations.