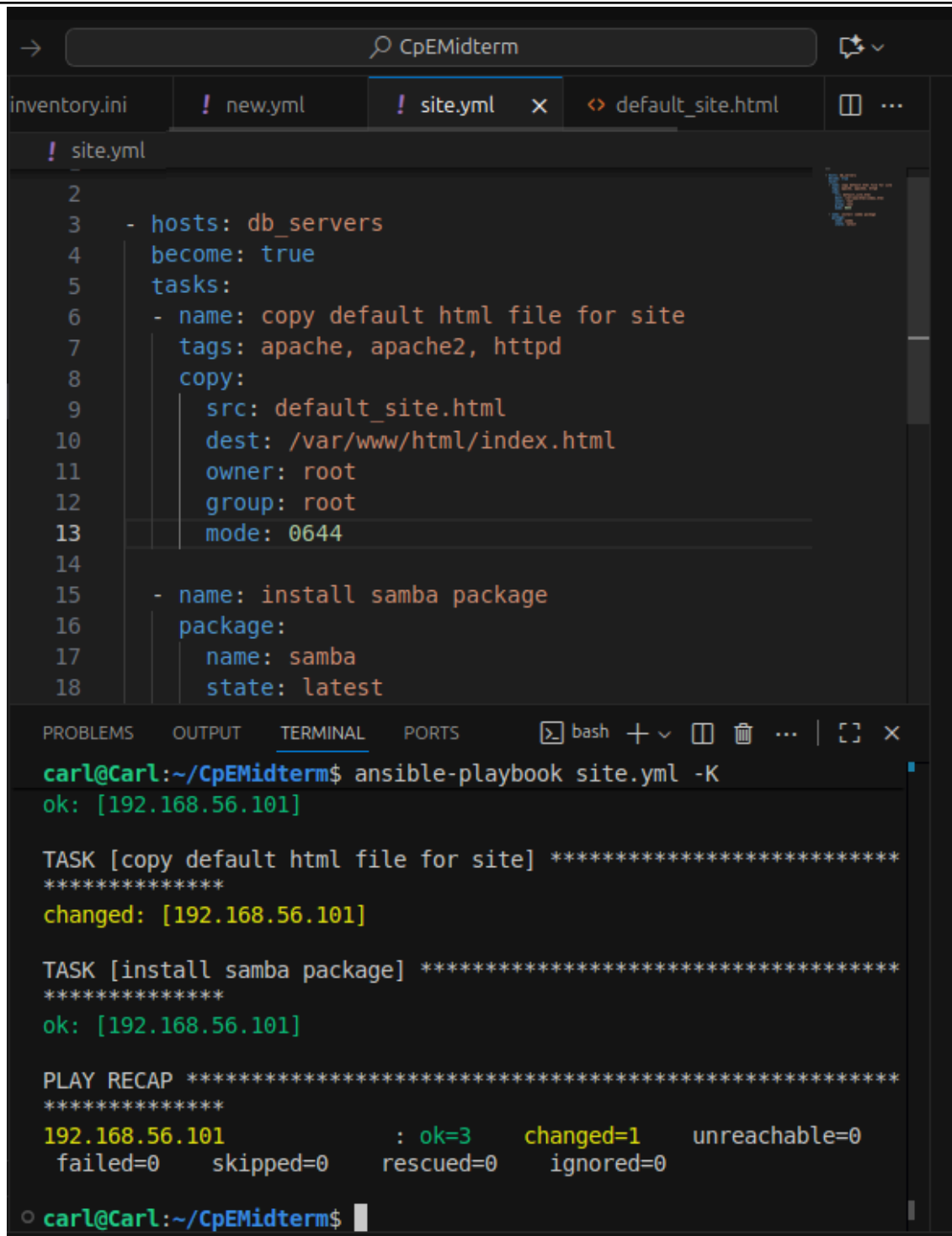


<b>Name: Carag, Carl Jervie B.</b>	<b>Date Performed: 12/10/25</b>
<b>Course/Section: CPE31S2</b>	<b>Date Submitted: 15/10/25</b>
<b>Instructor: Engr. Robin Valenzuela</b>	<b>Semester and SY: 1st Sem 2025-2026</b>
<b>Activity 7: Managing Files and Creating Roles in Ansible</b>	
<b>1. Objectives:</b> 1.1 Manage files in remote servers 1.2 Implement roles in ansible	
<b>2. Discussion:</b>  <p>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.</p>	
<b>Task 1: Create a file and copy it to remote servers</b>  <ol style="list-style-type: none"> <li>Using the previous directory we created, create a directory, and named it "<b>files</b>." Create a file inside that directory and name it "<b>default_site.html</b>." 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.</li> <li>Edit the <b>site.yml</b> file and just below the <b>web_servers</b> play, create a new file to copy the default html file for site: <ul style="list-style-type: none"> <li>name: copy default html file for site</li> <li>tags: apache, apache2, httpd</li> <li>copy: <ul style="list-style-type: none"> <li>src: default_site.html</li> <li>dest: /var/www/html/index.html</li> <li>owner: root</li> <li>group: root</li> <li>mode: 0644</li> </ul> </li> </ul> </li> <li>Run the playbook <b>site.yml</b>. Describe the changes.</li> </ol>	



The screenshot shows a code editor with a file named `site.yml` open. The file contains an Ansible playbook with two tasks: copying a default HTML file to a web server and installing the Samba package. Below the editor, a terminal window shows the execution of the playbook on a host named `192.168.56.101`. The output indicates that the first task was successful and resulted in a change, while the second task was successful without any changes.

```
inventory.ini  ! new.yml  ! site.yml  x  <> 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 + v [] [] ... | [] x

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

o 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](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.

The screenshot shows a code editor with a file named `site.yml` containing an Ansible playbook. The playbook has two hosts: `db_servers` and `workstations`. The `workstations` host has a `become: true` directive and a `tasks` list. The tasks include installing `unzip` and `terraform`. The `terraform` task includes a `unarchive` sub-task with `src` pointing to the Terraform releases page, `dest` set to `/usr/local/bin`, `remote_src` set to `yes`, `mode` set to `0755`, and `owner` set to `root`.

Below the editor is a terminal window titled `bash`. It shows the command `ansible-playbook site.yml -K` being executed. The output shows the following tasks and their results:

- `TASK [Gathering Facts]`: `ok: [192.168.56.101]`
- `TASK [install unzip]`: `ok: [192.168.56.101]`
- `TASK [install terraform]`: `changed: [192.168.56.101]`
- `TASK [install samba package]`: `ok: [192.168.56.101]`

**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.

```
state Advanced state manage  
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

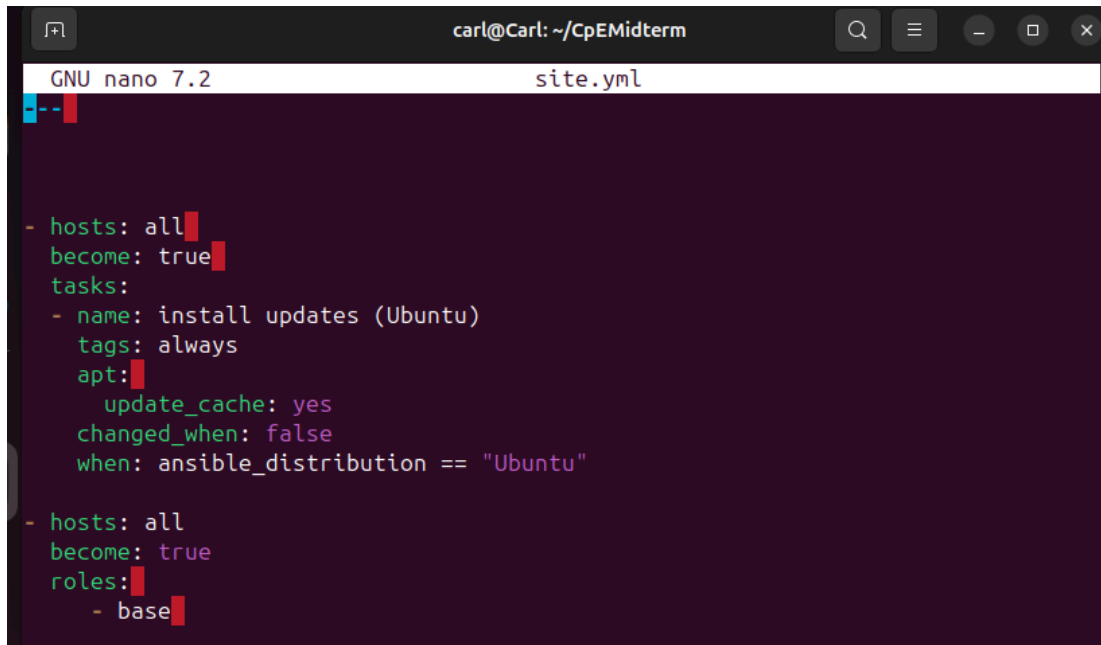
- hosts: file_servers
  become: true
  roles:
    - file_servers

```

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.

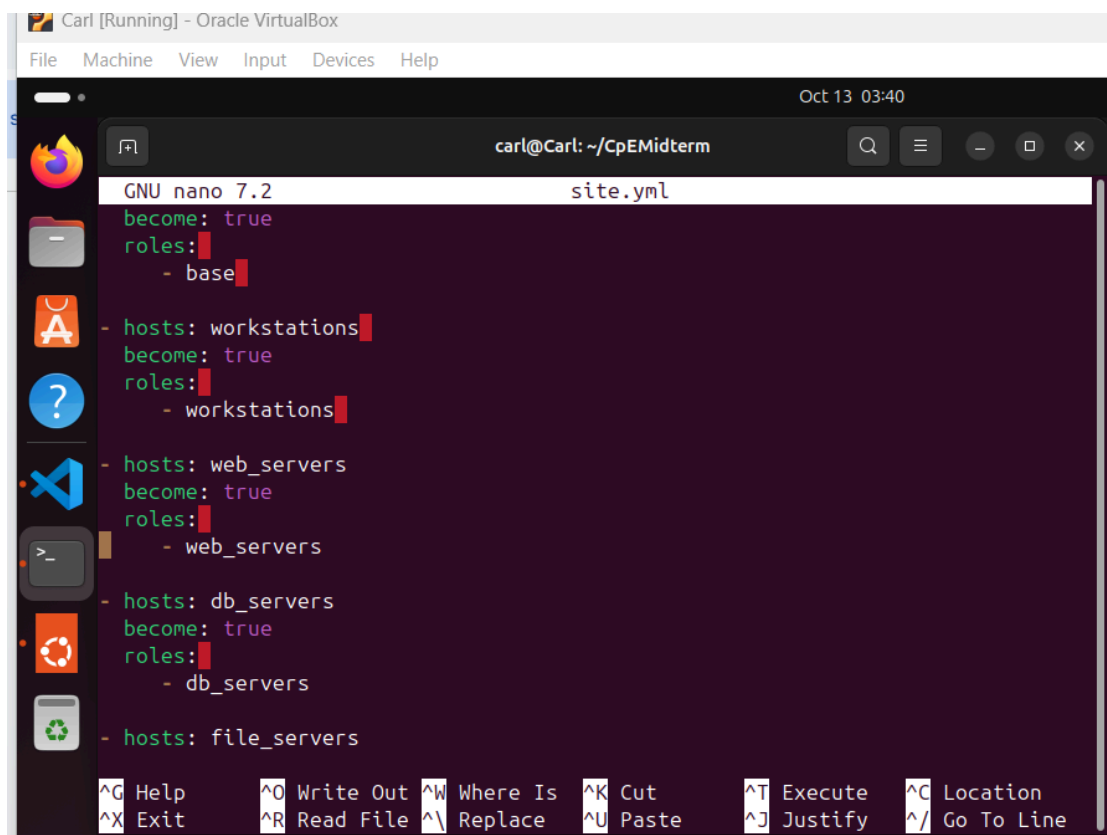


```
carl@Carl: ~/CpEMidterm
GNU nano 7.2 site.yml

--

- hosts: all
  become: true
  tasks:
    - name: install updates (Ubuntu)
      tags: always
      apt:
        update_cache: yes
        changed_when: false
        when: ansible_distribution == "Ubuntu"

- hosts: all
  become: true
  roles:
    - base
```



```
Carl [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Oct 13 03:40

carl@Carl: ~/CpEMidterm
GNU nano 7.2 site.yml

become: true
roles:
  - base

- hosts: workstations
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
    - web_servers

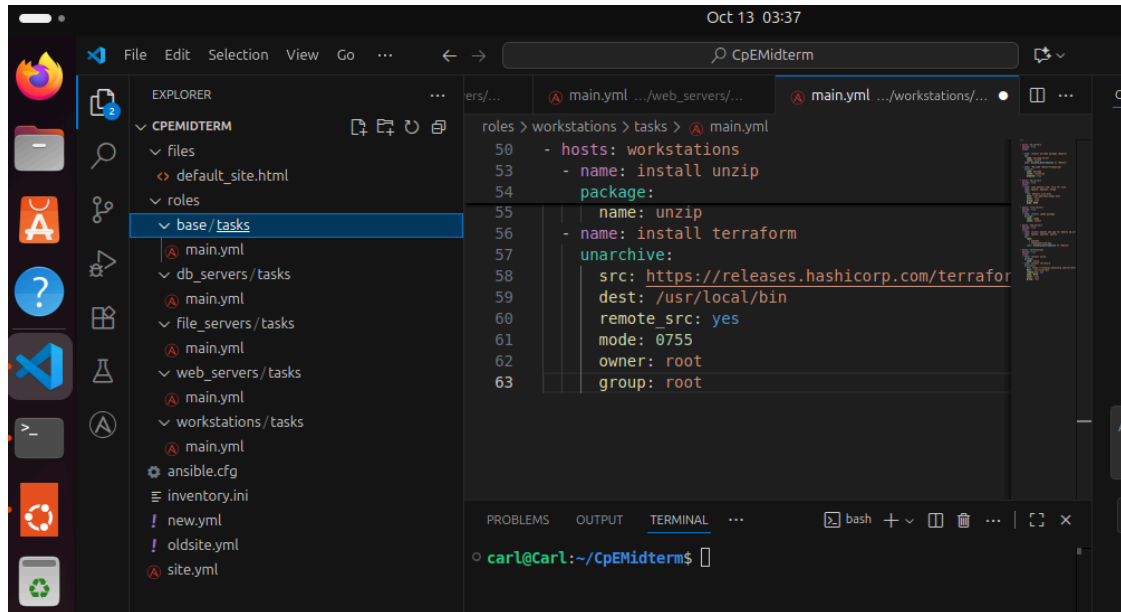
- hosts: db_servers
  become: true
  roles:
    - db_servers

- hosts: file_servers
```

^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

```
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.



