

Name: VILLASENOR, HANS RAINIER A.	Date Performed: 10/15/25
Course/Section: CPE31S2	Date Submitted: 10/15/25
Instructor: Engr. Robin Valenzuela	Semester and SY: 1st sem 2025-26
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: <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>	
Task 1: Create a file and copy it to remote servers <ol style="list-style-type: none"> 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. Edit the site.yml file and just below the web_servers play, create a new file to copy the default html file for site: <ul style="list-style-type: none"> name: copy default html file for site tags: apache, apache2, httpd copy: <ul style="list-style-type: none"> src: default_site.html dest: /var/www/html/index.html owner: root group: root mode: 0644 Run the playbook site.yml. Describe the changes. 	

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
HANS@LocalMachine:~/CPE31S1_VILLASENOR$ ansible-playbook --ask-become-pass site.yaml
TASK [Gathering Facts] *****
ok: [192.168.56.102]
ok: [192.168.56.104]

TASK [copy default html file for site] *****
changed: [192.168.56.102]
changed: [192.168.56.104]

TASK [install apache and php for Ubuntu servers] *****
skipping: [192.168.56.104]
ok: [192.168.56.102]

TASK [install apache and php for CentOS servers] *****
skipping: [192.168.56.102]

Ln 83, Col 1 Spaces: 2 UTF-8 LF {} YAML
```

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.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
HANS@LocalMachine:~/CPE31S1_VILLASENOR$ ssh HANS@192.168.56.102
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Wed Oct 15 02:17:42 2025 from 192.168.56.101
HANS@Server1:~$ -m -a cat /var/www/html/index.html
-m: command not found
HANS@Server1:~$ -m command -a cat /var/www/html/index.html
-m: command not found
HANS@Server1:~$ cat /var/www/html/index.html
<h1> Hello TIP </h1>

<p> Hello World </p>
HANS@Server1:~$

HANS@LocalMachine:~/CPE31S1_VILLASENOR$ ssh hans@192.168.56.104
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Wed Oct 15 10:13:33 2025 from 192.168.56.101
[hans@localhost ~]$ cat /var/www/html/index.html
<h1> Hello TIP </h1>

<p> Hello World </p>
[hans@localhost ~]$

Ln 84, Col 1 Spaces: 2 UTF-8 LF {} YAML
```

exi

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 [workstations] *****
TASK [Gathering Facts] *****
ok: [192.168.56.102]

TASK [install unzip] *****
ok: [192.168.56.102]

TASK [instal terraform] *****
changed: [192.168.56.102]
```

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

```
HANS@Server1:~$ terraform
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

⌕ Not Committed Yet  Ln 33, Col 68  Spaces: 2  UTF-8  LF  {} YAML
```

tas

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

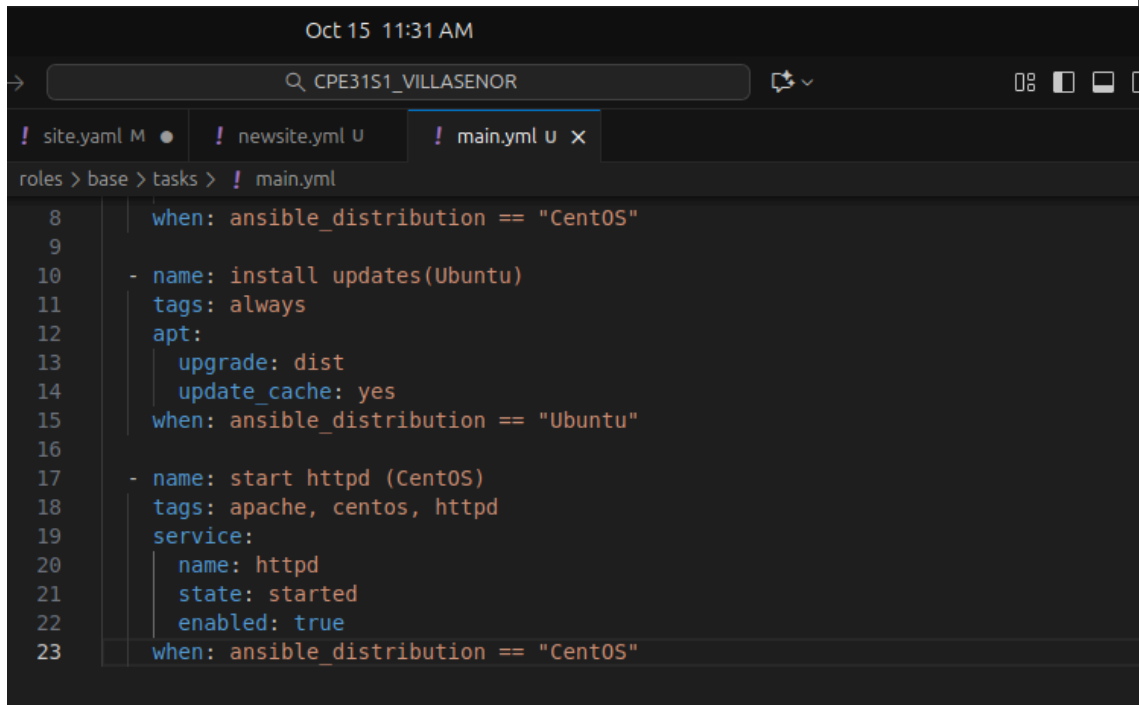
```

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.

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.



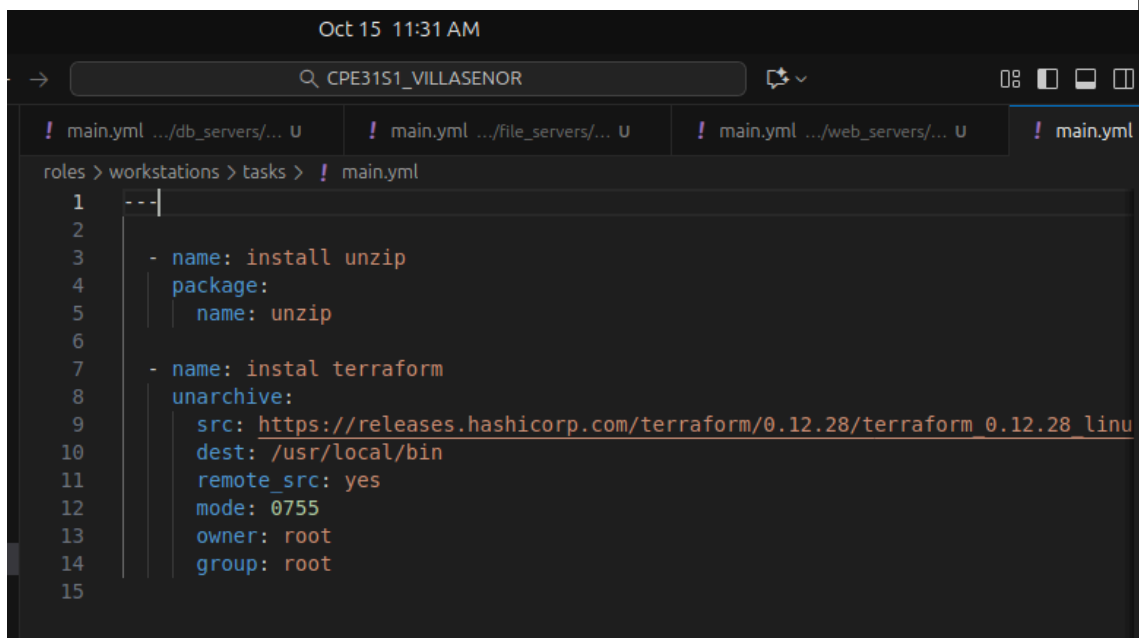
Oct 15 11:31 AM

→ CPE31S1_VILLASENOR

! site.yml M • ! newsite.yml U ! main.yml U x

roles > base > tasks > ! main.yml

```
8   when: ansible_distribution == "CentOS"
9
10  - name: install updates(Ubuntu)
11    tags: always
12    apt:
13      upgrade: dist
14      update_cache: yes
15    when: ansible_distribution == "Ubuntu"
16
17  - name: start httpd (CentOS)
18    tags: apache, centos, httpd
19    service:
20      name: httpd
21      state: started
22      enabled: true
23  when: ansible_distribution == "CentOS"
```



Oct 15 11:31 AM

→ CPE31S1_VILLASENOR

! main.yml .../db_servers/... U ! main.yml .../file_servers/... U ! main.yml .../web_servers/... U ! main.yml

roles > workstations > tasks > ! main.yml

```
1  --|
2
3  - name: install unzip
4    package:
5      name: unzip
6
7  - name: instal terraform
8    unarchive:
9      src: https://releases.hashicorp.com/terraform/0.12.28/terraform_0.12.28_linux
10     dest: /usr/local/bin
11     remote_src: yes
12     mode: 0755
13     owner: root
14     group: root
15
```

Oct 15 11:32 AM

→ CPE3151_VILLASENOR

! main.yml .../db_servers/... U ! main.yml .../file_servers/... U ! main.yml .../web_servers/... U X

roles > web_servers > tasks > ! main.yml

```
1 ---
2
3 - name: copy default html file for site
4   tags: apache, apache2, httpd
5   copy:
6     src: default_site.html
7     dest: /var/www/html/index.html
8     owner: root
9     group: root
10    mode: 0644
11
12 - name: install apache and php for Ubuntu servers
13   apt:
14     name:
15       - apache2
16       - libapache2-mod-php
17     state: latest
18     update_cache: yes
19     when: ansible_distribution == "Ubuntu"
20
21 - name: install apache and php for CentOS servers
22   dnf:
23     name:
24       - httpd
25       - php
26     state: latest
27     when: ansible_os_famil
```

Navigate to Command
Scroll to Previous Command (Ctrl+UpArrow)

Oct 15 11:32 AM

← → CPE3151_VILLASENOR

! site.yml M ! newsite.yml U ! main.yml .../db_servers/... U ! main.yml .../file_servers/... U X

roles > file_servers > tasks > ! main.yml

```
1 ---
2
3 - name: install samba package
4   package:
5     name: samba
6     state: latest
7
```

The screenshot shows a code editor with a dark theme. At the top, the system clock reads 'Oct 15 11:32 AM'. Below the clock is a search bar containing 'CPE3151_VILLASENOR'. There are three tabs open: 'site.yml M', 'newsite.yml U', and 'main.yml U x'. The 'main.yml' tab is active, showing a breadcrumb trail: 'roles > db_servers > tasks > ! main.yml'. The code is an Ansible task list with line numbers 1 through 19 on the left. The tasks are: 1. A separator '---'. 2. A task to install mariadb on CentOs using yum, with name 'mariadb-server', state 'latest', and a when condition 'ansible_os_family == "RedHat"'. 3. A task to restart/enable mariadb as a service, with name 'mariadb', state 'restarted', and enabled 'true'. 4. A task to install mariadb on Ubuntu using apt, with name 'mariadb-server', state 'latest', and a when condition 'ansible_distribution == "Ubuntu"'. The code is as follows:

```
1 ---
2
3 - name: install mariadb package(CentOs)
4   yum:
5     name: mariadb-server
6     state: latest
7   when: ansible_os_family == "RedHat"
8
9 - name: "Mariadb- Restarting/Enabling"
10  service:
11    name: mariadb
12    state: restarted
13    enabled: true
14
15 - name: install mariadb package(Ubuntu)
16  apt:
17    name: mariadb-server
18    state: latest
19  when: ansible_distribution == "Ubuntu"
```

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

Reflections:

Answer the following:

1. What is the importance of creating roles?
Roles help keep things organized especially in playbook.
2. What is the importance of managing files?
Files control how your system works.