| Name: Bueno, Joshua C. | Date Performed: 10/03/25 |
|---|---|
| Course/Section: CPE31S2 | Date Submitted:10/15/25 |
| 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

```
- hosts: web_servers
  become: true
  tasks:

  - 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.
   **-It copies the default_site.html file from the local files directory to /var/www/html/index.html**
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.

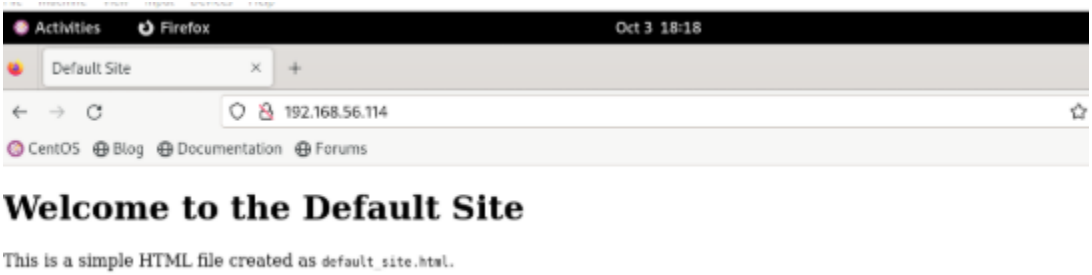
```
TASK [copy default html file for site] ***********
changed: [192.168.56.111]
changed: [192.168.56.112]
```

```
ok: [192.168.56.114]

TASK [copy default html file for site] ***************************************
changed: [192.168.56.114]

TASK [install apache and php for Ubuntu servers] *****************************
skipping: [192.168.56.114]
```

**Ubuntu:**

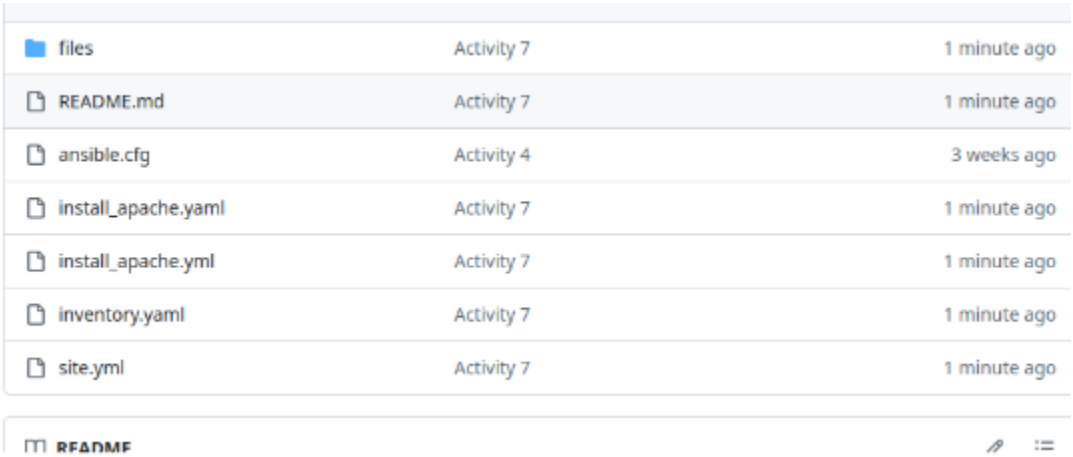
```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width,initial-scale=1" />
  <title>Default Site</title>
</head>
<body>
  <h1>Welcome to the Default Site</h1>
  <p>This is a simple HTML file created as <code>default_site.html</code>.</p
</body>
</html>
```

**Cent:**




**Based on the changes I made. The output**

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



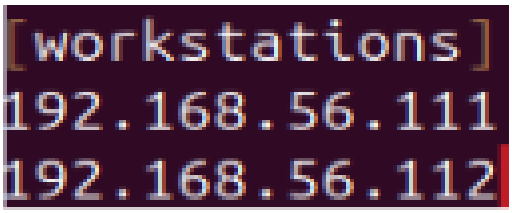

**-The output displays the content of the default_site.html file, showing the text and HTML elements that I created.**

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

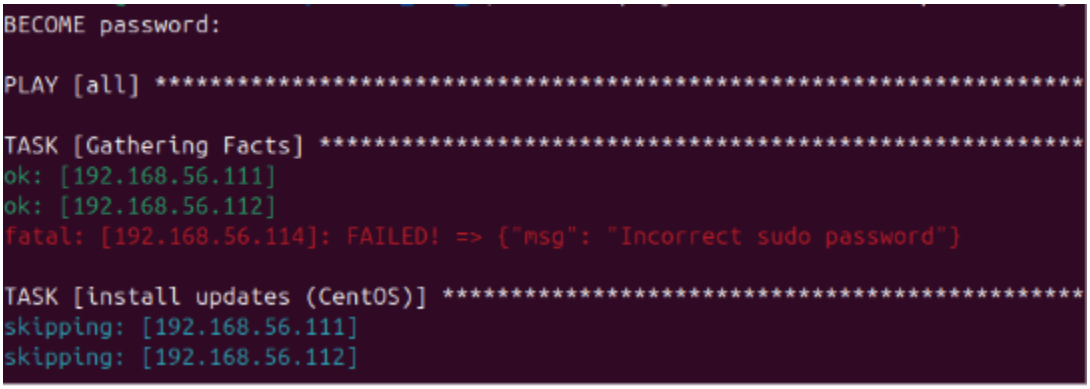
1. Edit the site.yml. Just before the web_servers play, create a new play:

```
- 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 [file_servers] ************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.112]
ok: [192.168.56.111]

TASK [install samba package] ***************************************************
ok: [192.168.56.112]
ok: [192.168.56.111]

PLAY RECAP *********************************************************************
192.168.56.111             : ok=13   changed=2    unreachable=0    failed=0
kipped=4    rescued=0    ignored=0
192.168.56.112             : ok=13   changed=2    unreachable=0    failed=0
kipped=4    rescued=0    ignored=0
192.168.56.114             : ok=0    changed=0    unreachable=0    failed=1
kipped=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
```

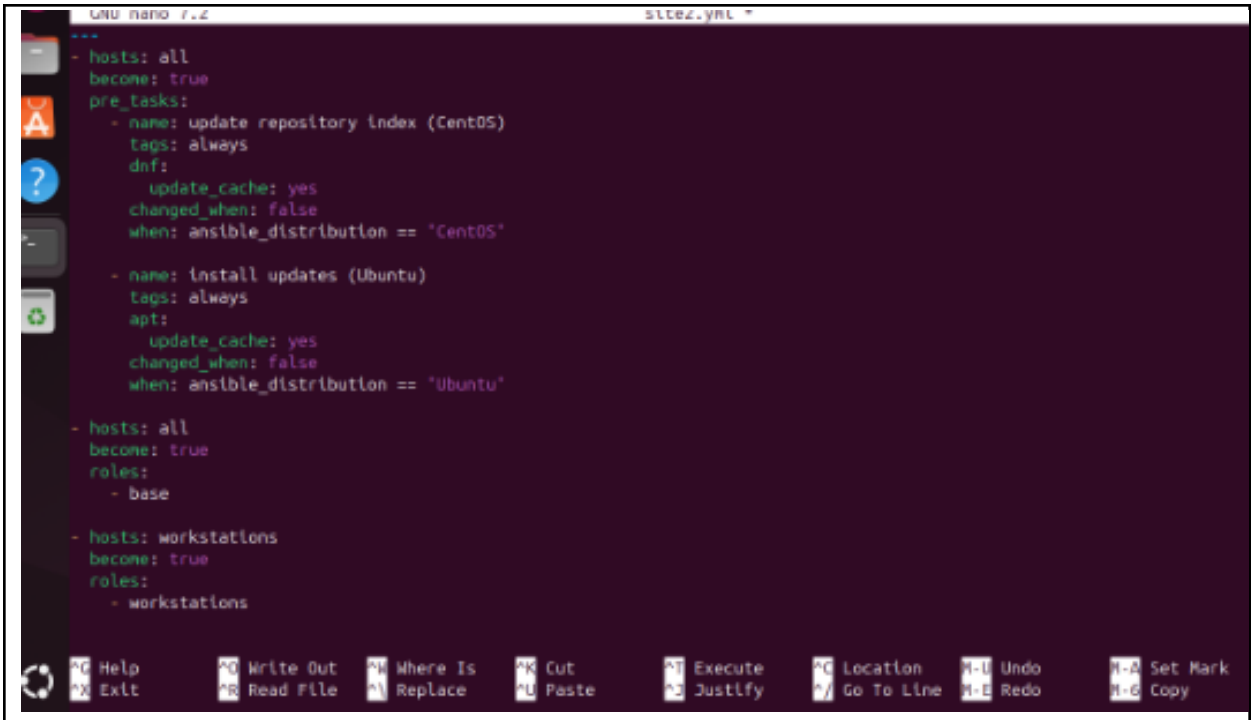
**- The ansible installs the unzip package and downloads the Terraform zip file to the Ubuntu workstation, then extracts it to /usr/local/bin**

**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.

```
er@workstation:~/CPE212_LIM_4$ ls
e.cfg  files  install_apache.yaml  install_apache.yml  inventory.yaml  README.md  roles  site2.yml  site.yml
er@workstation:~/CPE212_LIM_4$ cd roles
er@workstation:~/CPE212_LIM_4/roles$ ls
er@workstation:~/CPE212_LIM_4/roles$ mkdir -p {base,web_servers,file_servers,db_servers,workstations}/tasks
er@workstation:~/CPE212_LIM_4/roles$ ls
db_servers  file_servers  web_servers  workstations
er@workstation:~/CPE212_LIM_4/roles$ ls base
```

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/roles/base$ cd tasks
_4/roles/base/tasks$ ls
```

```
main.yml
```

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

**Ubuntu:**

```
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.111]
fatal: [192.168.56.114]: FAILED! => {"msg": "Incorrect sudo password"}
ok: [192.168.56.112]

TASK [update repository index (CentOS)] ****************************************
skipping: [192.168.56.111]
skipping: [192.168.56.112]

TASK [install updates (Ubuntu)] ************************************************
ok: [192.168.56.112]
```

```
TASK [db_servers : Mariadb- Restarting/Enabling] ********************************
changed: [192.168.56.111]
changed: [192.168.56.112]

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.111]
ok: [192.168.56.112]

TASK [file_servers : install samba package] ************************************
ok: [192.168.56.111]
ok: [192.168.56.112]

PLAY RECAP *********************************************************************
192.168.56.111             : ok=15   changed=1    unreachable=0    failed=0    s
kipped=5    rescued=0    ignored=0
192.168.56.112             : ok=15   changed=1    unreachable=0    failed=0    s
kipped=5    rescued=0    ignored=0
192.168.56.114             : ok=0    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
```

**Cent:**

```
ok: [192.168.56.112]

TASK [file_servers : install samba package] ***********************************
ok: [192.168.56.111]
ok: [192.168.56.112]

PLAY RECAP ********************************************************************
192.168.56.111             : ok=15   changed=1    unreachable=0    failed=0    s
kipped=5    rescued=0    ignored=0
192.168.56.112             : ok=15   changed=1    unreachable=0    failed=0    s
kipped=5    rescued=0    ignored=0
192.168.56.114             : ok=0    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0

vboxuser@workstation:~/CPE212_LIM_4$ ansible-playbook --ask-become-pass site2.ym
l
BECOME password:

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

TASK [Gathering Facts] ********************************************************
ok: [192.168.56.114]
```

```
TASK [db_servers : install mariadb package (Ubuntu)] **************************
skipping: [192.168.56.114]

TASK [db_servers : Mariadb- Restarting/Enabling] ******************************
changed: [192.168.56.114]

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

PLAY RECAP ********************************************************************
192.168.56.111             : ok=0    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
192.168.56.112             : ok=0    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
192.168.56.114             : ok=11   changed=1    unreachable=0    failed=0    s
kipped=4    rescued=0    ignored=0
```

**-It executes each role (base, web_servers, file_servers, db_servers, and workstations) in order, applying their respective tasks from the main.yml files to the assigned hosts.**

**Reflections:**

Answer the following:

1. What is the importance of creating roles?

**-Creating roles is important because it helps define clear responsibilities and access levels for each user, ensuring that tasks are organized and security is maintained. It allows for better collaboration and accountability within a system or organization.**

2. What is the importance of managing files?

**-Managing files is essential to keep data organized, secure, and easily accessible when needed. Proper file management also prevents data loss, duplication, and confusion, which improves efficiency and productivity.**