Name: Bacong, El Cid A.	**Date Performed:** 3/10/2025
Course/Section: CPE212 - CPE31S4	**Date Submitted:** 3/10/2025
Instructor: Engr. Robin Valenzuela	**Semester and SY:** 1st Sem, 25-26

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**:

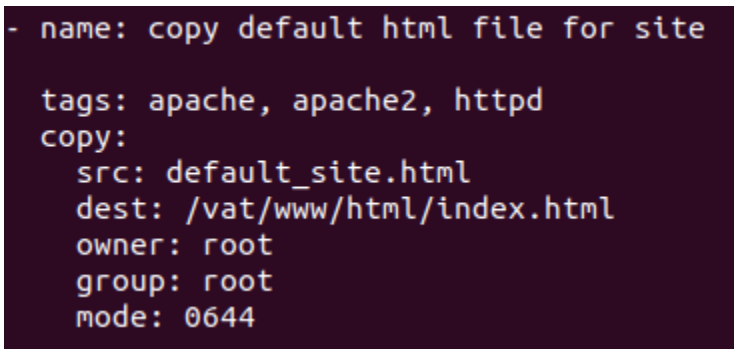
In this activity, we look at the concept of copying a file to a server. We are going to create a file into our git repository and use Ansible to grab that file and put it into a particular place so that we could do things like customize a default website, or maybe install a default configuration file. 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.

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

```
PLAY RECAP *****************************************************************
*
centos                     : ok=6    changed=1    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
server1                    : ok=3    changed=0    unreachable=0    failed=1
skipped=2    rescued=0    ignored=0
server2                    : ok=3    changed=0    unreachable=0    failed=1
skipped=2    rescued=0    ignored=0
server3                    : ok=2    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
```

It copied a custom HTML file from the Ansible workstation to web_server document directory.

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.

```
[cidee@centos ~]$ cat /var/www/html/index.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Default Page</title>
</head>
<body>
    <h1>Welcome to the Default Page</h1>
    <p>This is a simple HTML document.</p>
</body>
</html>[cidee@centos ~]$
```

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

```
cidee@workstation:~/CPE212_elcid$ git add .
cidee@workstation:~/CPE212_elcid$ git commit -m .
[main 3ee442b] .
 9 files changed, 692 insertions(+), 57 deletions(-)
 create mode 100644 files/default_site.html
 create mode 100644 old_site.yml
 create mode 100644 roles/base/tasks/main.yml
 create mode 100644 roles/db_servers/tasks/main.yml
 create mode 100644 roles/file_servers/tasks/main.yml
 create mode 100644 roles/web_servers/tasks/main.yml
 create mode 100644 roles/workstations/tasks/main.yml
cidee@workstation:~/CPE212_elcid$ git push origin main
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (10/10), 1.60 KiB | 818.00 KiB/s, done.
Total 10 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:moussecake22/CPE212_elcid.git
   2ccca5c..3ee442b  main -> main
```

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
```

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

   - name: install unzip
     package:
       name: unzip

   - name: install terraform
     unarchive:

       src: https://releases.hashicorp.com/te
       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.

```
[workstations]
workstation ansible_host=192.168.56.105 ansib
```

3. Run the playbook. Describe the output.

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

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

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

TASK [install terraform] *******************************************************
*
changed: [workstation]
```

This playbook installs Terraform on Linux workstations by downloading the binary zip from HashiCorp's releases, extracting it, and placing it in /usr/local/bin with proper executable permissions.

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

```
cidee@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
    output             Read an output from a state file
    plan               Generate and show an execution plan
    providers          Prints a tree of the providers used in the configuration
    refresh            Update local state file against real resources
    show               Inspect Terraform state or plan
    taint              Manually mark a resource for recreation
    untaint            Manually unmark a resource as tainted
```

It listed the possible commands to use in 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)

```
cidee@workstation:~/CPE212_elcid$ cp site.yml old_site.yml
cidee@workstation:~/CPE212_elcid$ ls
ansible.cfg  install_apache_redhat.yml  inventory.ini  README.md
files        install_apache.yml         old_site.yml   site.yml
```

```
---
- 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.

```
cidee@workstation:~/CPE212_elcid$ mkdir -p roles/{workstations,base,web_servers,
file_servers,db_servers}/tasks
cidee@workstation:~/CPE212_elcid$ ls
ansible.cfg  install_apache_redhat.yml  inventory.ini  README.md  site.yml
files        install_apache.yml         old_site.yml   roles
cidee@workstation:~/CPE212_elcid$ cd roles
cidee@workstation:~/CPE212_elcid/roles$ ls
base  db_servers  file_servers  web_servers  workstations
```

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.

```
cidee@workstation:~/CPE212_elcid$ cat roles/*/tasks/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: 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_amd
64.zip
      dest: /usr/local/bin
      remote_src: yes
      mode: 0755
```

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

  - name: copy default html file for site

    tags: apache, apache2, httpd
    copy:
      src: /home/cidee/CPE212_elcid/files/default_site.html
      dest: /var/www/html/index.html
      owner: root
      group: root
      mode: 0644

  - name: install apache and php for Ubuntu servers
    tags: apache, apache2, ubuntu
    apt:
      name:
        - apache2
        - libapache2-mod-php
      state: latest
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

  - name: install apache and php for CentOS servers
    tags: apache, centos, httpd
    dnf:
      name:
        - httpd
        - php
      state: latest
    when: ansible_distribution == "CentOS"

  - name: start httpd (CentOS)
    tags: apache, centos, httpd
    service:
      name: httpd
```

```
      state: started
    when: ansible_distribution == "CentOS"

- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    tags: centos, db, mariadb
    yum:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

  - name: "Mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true

  - name: install mariadb package (Ubuntu)
    tags: db, mariadb, ubuntu
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"

- hosts: file_servers
  become: true
  tasks:

  - name: install samba package
    tags: samba
    package:
      name: samba
      state: latest
---
- hosts: all
```

```
- 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: 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_amd
64.zip
      dest: /usr/local/bin
      remote_src: yes
      mode: 0755
      owner: root
      group: root

- hosts: web_servers
```

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

```
cidee@workstation:~/CPE212_elcid$ ansible-playbook site.yml -K
BECOME password:
ERROR! conflicting action statements: hosts, pre_tasks

The error appears to be in '/home/cidee/CPE212_elcid/roles/base/tasks/main.yml':
 line 2, column 3, but may
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:


---
- hosts: all
  ^ here
```

Misalignment issues probably. I couldn't pinpoint which one though.

Reflections:

Answer the following:

1. What is the importance of creating roles?
   Roles keep tasks organized, reusable, and easier to manage instead of having one long playbook.
2. What is the importance of managing files?
   Managing files ensures consistency, avoids errors, and makes updates simpler and faster.