Name: Baltazar, Paul Eimar R.	**Date Performed:** Oct 7, 2024
Course/Section: CPE212-CPE31S2	**Date Submitted:** Oct 9, 2024
Instructor: Engr. Robin Valenzuela	**Semester and SY:** 1st Sem [2024-2025]

Activity 7: Managing Files and Creating Roles in Ansible

1. **Objectives:**

1.1 Manage files in remote servers

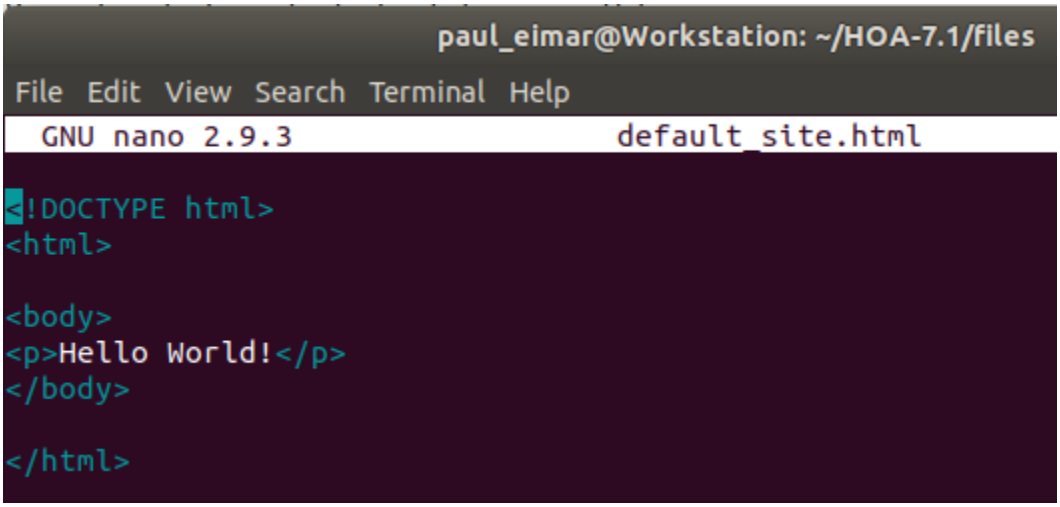
1.2 Implement roles in ansible

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: [192.168.56.106]
```

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.

```
paul_eimar@Server1:~$ cat /var/www/html/index.html
<!DOCTYPE html>
<html>

<body>
<p>Hello World!</p>
</body>

</html>
```

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

```
paul_eimar@Workstation:~/HOA-7.1$ git add ansible.cfg files inventory site.yml
paul_eimar@Workstation:~/HOA-7.1$ git commit "Baltazar HOA 7.1"
error: pathspec 'Baltazar HOA 7.1' did not match any file(s) known to git.
paul_eimar@Workstation:~/HOA-7.1$ git commit -m "Baltazar HOA 7.1"
[main 6617709] Baltazar HOA 7.1
 4 files changed, 114 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 files/default_site.html
 create mode 100644 inventory
 create mode 100644 site.yml
paul_eimar@Workstation:~/HOA-7.1$ git push
Counting objects: 7, done.
Delta compression using up to 6 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 1.15 KiB | 1.15 MiB/s, done.
Total 7 (delta 0), reused 0 (delta 0)
To github.com:rpldpaul/HOA-7.1.git
   bd090ae..6617709  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

```
- 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.

```
[workstation]
192.168.56.106

[web_servers]
192.168.56.106

[db_servers]
192.168.56.107
192.168.56.109 ansible_user=pbaltazar

[file_servers]
192.168.56.110 ansible_user=pbaltazar
```

3. Run the playbook. Describe the output.

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

TASK [Gathering Facts] *********************
ok: [192.168.56.106]

TASK [install unzip] ***********************
ok: [192.168.56.106]

TASK [install terraform] *******************
changed: [192.168.56.106]
```

It executed properly

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

```
paul_eimar@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
    validate           Validates the Terraform files
    version            Prints the Terraform version
    workspace          Workspace management

All other commands:
    0.12upgrade        Rewrites pre-0.12 module source code for v0.12
    debug              Debug output management (experimental)
    force-unlock       Manually unlock the terraform state
    push               Obsolete command for Terraform Enterprise legacy (v1)
    state              Advanced state management
```

terraform installed successfully in ubuntu server

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.

```
---
- 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
```

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.

```
paul_eimar@Workstation:~/HOA-7.1/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.

```
paul_eimar@Workstation: ~/HOA-7.1/roles/base/tasks
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                    main.yml

---

  - name: update repository index (CentOS)
    tags: always
    yum:
      name: "*"
      state: latest
    when: ansible_distribution == "CentOS"

  - name: install updates(Ubuntu)
    tags: always
    apt:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "Ubuntu"
```

base

Open

main.yml
~/HOA-7.1/roles/db_servers/tasks

```yaml
---

- 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,marialdb,ubuntu
  apt:
    name: mariadb-server
    state: latest
  when: ansible_distribution == "Ubuntu"
```

db_servers

Open

main.yml
~/HOA-7.1/roles/file_servers/tasks

main.yml

```yaml
---

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

file_servers

Activities Text Editor Wed 09:07

Open

main.yml
~/HOA-7.1/roles/web_servers/tasks

main.yml main.yml

```yaml
---

- 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

- 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
  yum:
    name:
      - httpd
      - php
    state: latest
  when: ansible_distribution == "CentOS"

- name: start httpd (CentOS)
  tags: apache,centos,httpd
  service:
    name: httpd
    state: started
    enable: true
  when: ansible_distribution == "CentOS"
```

web_servers

```yaml
---

- 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
```

workstations

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

```
TASK [workstations : install terraform] ********************************************************
ok: [192.168.56.106]

PLAY [web_servers] *****************************************************************************

TASK [Gathering Facts] *************************************************************************
ok: [192.168.56.106]

TASK [web_servers : copy default html file for site] ******************************************
ok: [192.168.56.106]

TASK [web_servers : install apache and php for Ubuntu servers] *********************************
ok: [192.168.56.106]

TASK [web_servers : install apache and php for CentOS servers] *********************************
skipping: [192.168.56.106]

TASK [web_servers : start httpd (CentOS)] ******************************************************
skipping: [192.168.56.106]

PLAY [db_servers] ******************************************************************************

TASK [Gathering Facts] *************************************************************************
ok: [192.168.56.107]
ok: [192.168.56.109]

TASK [db_servers : install mariadb package (CentOS)] *******************************************
skipping: [192.168.56.107]
ok: [192.168.56.109]

TASK [db_servers : Mariadb- Restarting/Enabling] ***********************************************
changed: [192.168.56.107]
changed: [192.168.56.109]

TASK [db_servers : install mariadb package (Ubuntu)] *******************************************
skipping: [192.168.56.109]
ok: [192.168.56.107]

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

TASK [Gathering Facts] *************************************************************************
ok: [192.168.56.110]

TASK [file_servers : install samba package] ****************************************************
ok: [192.168.56.110]

PLAY RECAP *************************************************************************************
192.168.56.106             : ok=8    changed=0    unreachable=0    failed=0
192.168.56.107             : ok=5    changed=1    unreachable=0    failed=0
192.168.56.109             : ok=5    changed=1    unreachable=0    failed=0
192.168.56.110             : ok=4    changed=0    unreachable=0    failed=0

paul_eimar@Workstation:~/HOA-7.1$
```

Every task ran successfully

Reflections:

Answer the following:

1. What is the importance of creating roles?

   – The importance of creating roles is to clean up the commands that are used in the playbook. By having roles, you can break down large playbooks into smaller and reusable components. it also allows for easier maintenance of your automation code.

2. What is the importance of managing files?

   – It is important to manage your files since many services rely on configuration files. If these files are not properly managed, it can lead to confusion and might be the cause of difficulty in debugging. Managing your files can also be helpful in system rollback, especially if a corruption occurred, then it will be easier to make a recovery in your system.