Name: Juanson, Aliya Dane P.	Date Performed: October 7, 2024
Course/Section: CPE31S2	Date Submitted:October 9, 2024
Instructor: Sir Robin Valenzuela	Semester and SY: 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

1. Using the previous directory we created, create a directory, and named it "*files*." Create a file inside that di rectory 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.

```
qadjuanson@workstation:~/HOA7/HOA7$ cd files
qadjuanson@workstation:~/HOA7/HOA7/files$ ls
default_site.html
qadjuanson@workstation:~/HOA7/HOA7/files$ cat default_site.html
<html>
<body>

<h1>Aliya</h1>
<h2>Dane</h2>
<h3>Pacia</h3>
<h4>Juanson</h4>

</body>
</html>
```

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
```

```
GNU nano 2.9.3                          site.yml

    name:
      - apache2
      - libapache2-mod-php
    state: latest
  when: ansible_distribution == "Ubuntu"

- 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 CentOS servers
  dnf:
    name:
      - httpd
      - php
    state: latest
  when: ansible_distribution == "CentOS"
```

3. Run the playbook *site.yml*. Describe the changes.

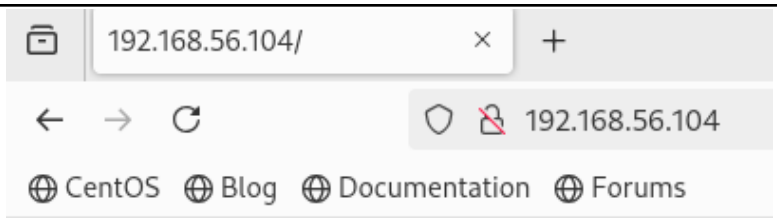
```
TASK [web_servers : install updates (CentOS)] ************************************************
skipping: [192.168.56.102]
ok: [192.168.56.104]

TASK [web_servers : install updates (Ubuntu)] ************************************************
skipping: [192.168.56.104]
ok: [192.168.56.102]

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

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

TASK [web_servers : start httpd (CentOS)] ****************************************************
skipping: [192.168.56.102]
ok: [192.168.56.104]

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

PLAY RECAP ***********************************************************************************
192.168.56.102             : ok=4    changed=0    unreachable=0    failed=0    skipped=3    res
cued=0    ignored=0
192.168.56.104             : ok=5    changed=0    unreachable=0    failed=0    skipped=2    res
cued=0    ignored=0

qadjuanson@workstation:~/HOA7/Activity_7$
```

- The playbook will copy the default_site.html to the web_servers.

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.

```
[qadjuanson@localhost ~]$ cd /var/www/html
[qadjuanson@localhost html]$ ls
index.html
[qadjuanson@localhost html]$
```

```
qadjuanson@server1:~$ cd /var/www/html
qadjuanson@server1:/var/www/html$ ls
index.html
qadjuanson@server1:/var/www/html$
```



Aliya

Dane

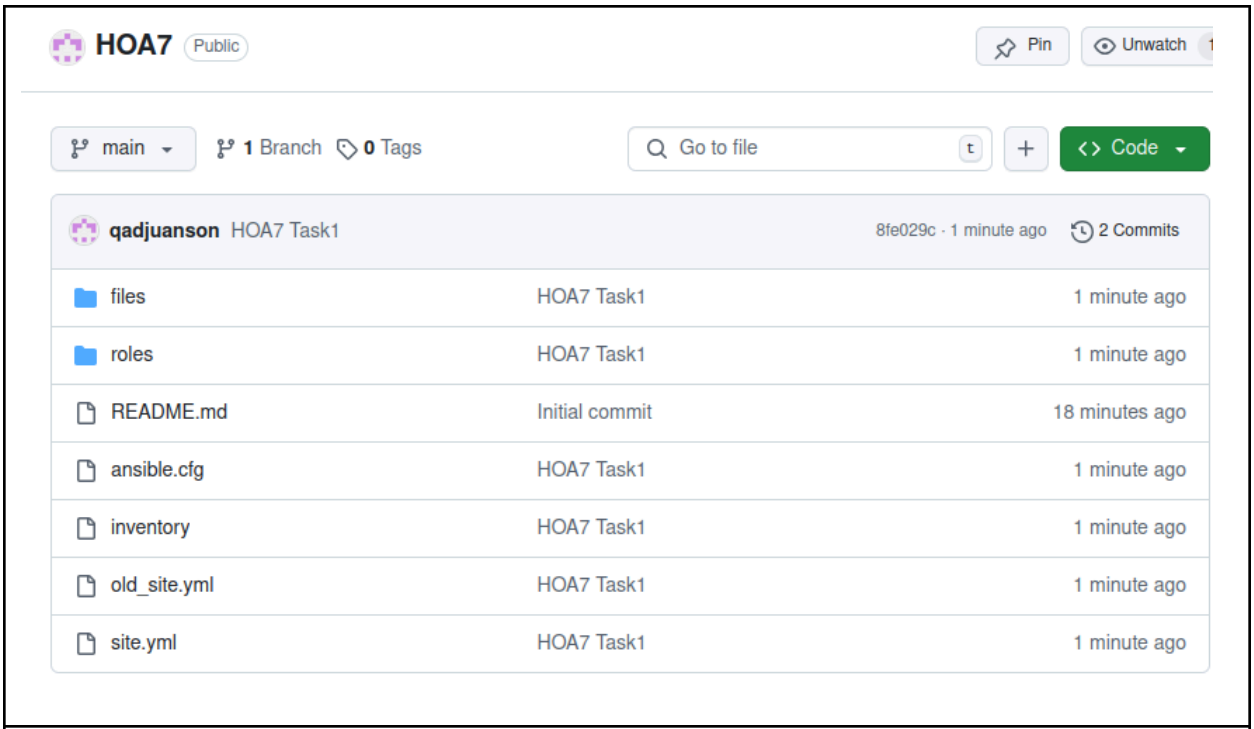
Pacia

Juanson

- It shows the content of default_site.html.

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

```
qadjuanson@workstation:~/HOA7/HOA7$ git add ansible.cfg
qadjuanson@workstation:~/HOA7/HOA7$ git add files
qadjuanson@workstation:~/HOA7/HOA7$ git add inventory
qadjuanson@workstation:~/HOA7/HOA7$ git add old_site.yml
qadjuanson@workstation:~/HOA7/HOA7$ git add site.yml
qadjuanson@workstation:~/HOA7/HOA7$ git add README.md
qadjuanson@workstation:~/HOA7/HOA7$ git add roles
qadjuanson@workstation:~/HOA7/HOA7$ git commit -m "HOA7 Task1"
[main 8fe029c] HOA7 Task1
 12 files changed, 323 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 files/default_site.html
 create mode 100644 inventory
 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/ansible.cfg
 create mode 100644 roles/web_servers/tasks/inventory
 create mode 100644 roles/web_servers/tasks/main.yml
 create mode 100644 roles/workstations/tasks/main.yml
 create mode 100644 site.yml
qadjuanson@workstation:~/HOA7/HOA7$ git push origin main
Enumerating objects: 27, done.
Counting objects: 100% (27/27), done.
Compressing objects: 100% (15/15), done.
Writing objects: 100% (26/26), 3.13 KiB | 3.13 MiB/s, done.
Total 26 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), done.
To github.com:qadjuanson/HOA7.git
   f459e30..8fe029c  main -> main
qadjuanson@workstation:~/HOA7/HOA7$
```



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
```

```
- 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.

```
BECOME password:

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

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

TASK [workstations : install updates (CentOS)] ********************************************
skipping: [192.168.56.102]

TASK [workstations : install updates (Ubuntu)] ********************************************
ok: [192.168.56.102]

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

TASK [workstations : install terraform] ***************************************************
changed: [192.168.56.102]

PLAY RECAP ********************************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0    skipped=1    res
cued=0    ignored=0
```

- It installs the unzip package and downloads the terraform.

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

```
qadjuanson@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
```

- The terraform is now successfully installed.

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.

```
  GNU nano 6.2                                    site.yml
---
- hosts: all
  become: true
  pre_tasks:

  - name: update respository index (CentOS)
    tags: always
    yum:
      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:
```

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.

```
qadjuanson@workstation:~/HOA7/HOA7$ cd roles
qadjuanson@workstation:~/HOA7/HOA7/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.

```
qadjuanson@workstation:~/HOA7/HOA7/roles/web_servers/tasks$ ls
main.yml
qadjuanson@workstation:~/HOA7/HOA7/roles/web_servers/tasks$ sudo nano main.ym
qadjuanson@workstation:~/HOA7/HOA7/roles/web_servers/tasks$ cat main.yml
---

- 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"

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

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

- name: start httpd (CentOS)
```

```
qadjuanson@workstation:~/HOA7/HOA7/roles/db_servers/tasks$ cat main.yml
---
- name: install updates (CentOS)
  tags: always
  yum:
    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"

- 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"
```

```
qadjuanson@workstation:~/HOA7/HOA7/roles/file_servers/tasks$ cat main.yml
---
- name: install updates (CentOS)
  tags: always
  yum:
    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"

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

```
qadjuanson@workstation:~/HOA7/HOA7/roles/workstations/tasks$ cat main.yml
---
- name: install updates (CentOS)
  tags: always
  yum:
    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"

- 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
```

```
qadjuanson@workstation:~/HOA7/HOA7/roles/base/tasks$ cat main.yml
---
- name: install updates (CentOS)
  tags: always
  yum:
    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"
```

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

```
TASK [db_servers : install updates (Ubuntu)] ************************************************
skipping: [192.168.56.104]
ok: [192.168.56.102]

TASK [db_servers : install mariadb package (CentOS)] ****************************************
skipping: [192.168.56.102]
ok: [192.168.56.104]

TASK [db_servers : MariaDB - Restarting/Enabling] *******************************************
changed: [192.168.56.102]
changed: [192.168.56.104]

TASK [db_servers : install mariadb package (Ubuntu)] ****************************************
skipping: [192.168.56.104]
ok: [192.168.56.102]

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

TASK [Gathering Facts] **********************************************************************
ok: [192.168.56.101]

TASK [file_servers : install updates (CentOS)] **********************************************
skipping: [192.168.56.101]

TASK [file_servers : install updates (Ubuntu)] **********************************************
ok: [192.168.56.101]

TASK [file_servers : install samba package] *************************************************
ok: [192.168.56.101]

PLAY RECAP ******************************************************************************************
192.168.56.101             : ok=7    changed=0    unreachable=0    failed=0    skipped=3    res
cued=0    ignored=0
192.168.56.102             : ok=16   changed=1    unreachable=0    failed=0    skipped=8    res
cued=0    ignored=0
192.168.56.104             : ok=13   changed=1    unreachable=0    failed=0    skipped=6    res
cued=0    ignored=0
```

- Running the playbook shows all the configurations I made without errors.

Reflections:

Answer the following:

1. What is the importance of creating roles?
   - Creating roles in Ansible is important because it helps organize and simplify complex automation tasks. Roles break down large playbooks into smaller, reusable parts, each handling a specific function, like setting up a web server or configuring a database.
2. What is the importance of managing files?
   - Managing files in Ansible is essential for effective automation, as it allows you to handle configurations, templates, and scripts on remote servers. Ansible provides various modules for file management.