Name: Ballesteros, John Erwin S.	**Date Performed:** 10/7/2024
Course/Section: Cpe 31s2	**Date Submitted:** 10/9/2024
Instructor: Engr. Robin Valenzuela	**Semester and SY:** 1st Sem, '24-'25

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:

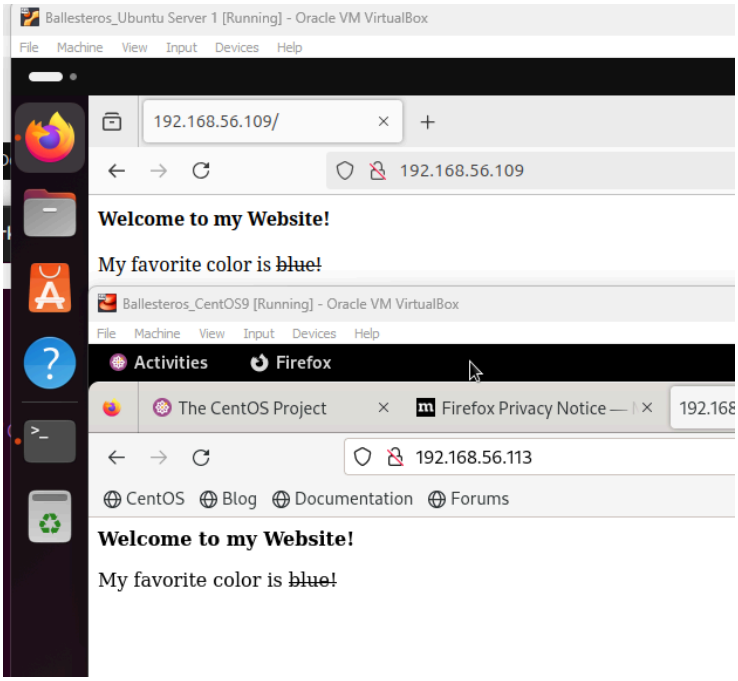
```
- name: copy default html file for site

  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.
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.

```
TASK [copy default html file for site] ****************
changed: [192.168.56.109]

TASK [install apache and php for Ubuntu Servers] ******
```

```
TASK [copy default html file for site] **********
ok: [192.168.56.109]
changed: [192.168.56.113]
```

```
erwin@Server1:~$ cd /var/www/html
erwin@Server1:/var/www/html$ cat index.html
<b> Welcome to my Website! </b>

<p> My favorite color is <del>blue!</del> </p>
```



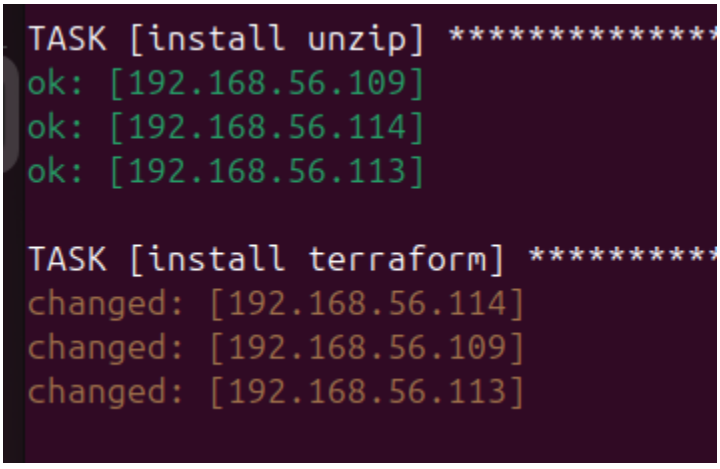
The output shows that the index.html that we have created appears in the file path that we entered

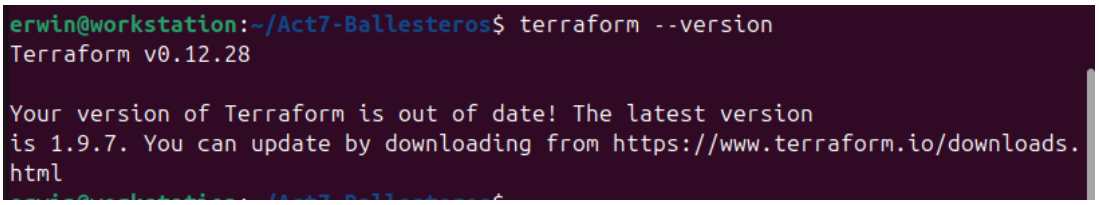
Task 2: Download a file and extract it to a remote server

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





```
[erwin@centos9 ~]$ terraform --version
Terraform v0.12.28

Your version of Terraform is out of date! The latest version
is 1.9.7. You can update by downloading from https://www.terraform.io/downloads.
html
[erwin@centos9 ~]$
```

```
erwin@Server1:~$ terraform --version
Terraform v0.12.28

Your version of Terraform is out of date! The latest version
is 1.9.7. You can update by downloading from https://www.terraform.io/downloads.
html
erwin@Server1:~$
```

The output shows that terraform has been updated and that we are still able update it to the latest version if we desire.

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)

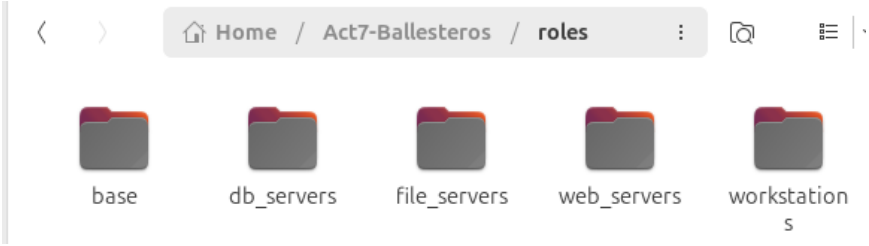
```
---
- hosts: all
  become: true
  pre_tasks:

  - 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.

```yaml
---
- 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"
```

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 eacch directory, create a directory and name it tasks.

```
erwin@workstation:~/Act7-Ballesteros$ mkdir roles
erwin@workstation:~/Act7-Ballesteros$ ls
files  README.md  roles  site.yml  site.yml.bak
erwin@workstation:~/Act7-Ballesteros$ cd roles
erwin@workstation:~/Act7-Ballesteros/roles$ mkdir base
erwin@workstation:~/Act7-Ballesteros/roles$ mkdir web_servers
erwin@workstation:~/Act7-Ballesteros/roles$ mkdir file_servers
erwin@workstation:~/Act7-Ballesteros/roles$ mkdir db_servers
erwin@workstation:~/Act7-Ballesteros/roles$ mkdir workstations
erwin@workstation:~/Act7-Ballesteros/roles$ ls
base  db_servers  file_servers  web_servers  workstations
erwin@workstation:~/Act7-Ballesteros/roles$
```



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.

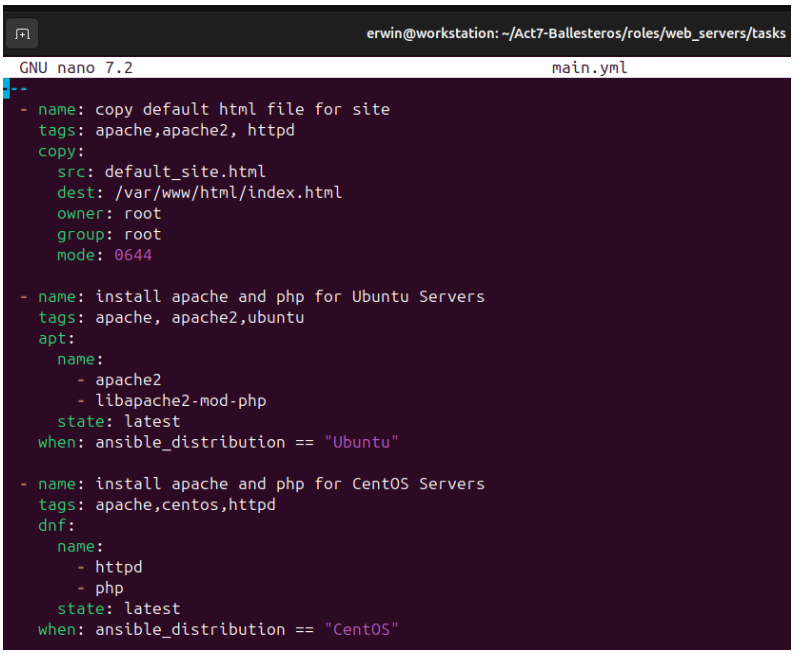
base main.yml

```yaml
---
  - name: install updates Ubuntu
    tags: always
    apt:
      upgrade: dist
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

  - name: install updates Centos
    tags: always
    dnf:
      update_only: yes
      update_cache: yes
    when: ansible_distribution == "CentOS"
```

workstation main.yml

```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
```

web_server main.yml



db_server main.yml

```
 GNU nano 7.2                                                main.yml
---
  - name: install mariadb package ubuntu
    tags: db, mariadb,ubuntu
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"

  - name: install mariadb for CentOS Servers
    tags: db. mariadb,centos
    yum:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

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

file_server main.yml

```
                                  erwin@workstation: ~/Act7-Ballesteros/roles/file_servers/tasks
 GNU nano 7.2                                                main.yml
--
 - name: install samba package
   tags: samba
   package:
     name: samba
     state: latest
```

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

```
TASK [web_servers : install apache and php for CentOS Servers] *
skipping: [192.168.56.109]
ok: [192.168.56.113]

TASK [web_servers : start httpd (centos)] ***********************
skipping: [192.168.56.109]
changed: [192.168.56.113]

TASK [web_servers : start apache2 (ubuntu)] *********************
skipping: [192.168.56.113]
ok: [192.168.56.109]

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

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

TASK [db_servers : install mariadb package ubuntu] **************
ok: [192.168.56.110]

TASK [db_servers : install mariadb for CentOS Servers] **********
```

```
PLAY RECAP ********************************************************************************
192.168.56.109             : ok=11   changed=0    unreachable=0    failed=0    skipped=4    r
192.168.56.110             : ok=7    changed=1    unreachable=0    failed=0    skipped=3    r
192.168.56.113             : ok=13   changed=1    unreachable=0    failed=0    skipped=4    r
192.168.56.114             : ok=7    changed=0    unreachable=0    failed=0    skipped=2    r

erwin@workstation:~/Act7-Ballesteros$
```

On the output of the site.yml it has changed significantly because instead of all the tasks being in one playbook they have been distributed across multiple playbooks where they are separated in each role with different tasks. It checks first for the roles and available remote server. It then proceeds to perform the playbook.

Reflections:

Answer the following:

1. **What is the importance of creating roles?**
   Creating roles in Ansible is one of the best ways to be able to break down simple tasks for different organization or groups. Doing this helps us to declutter our main ansible playbook while maintaining to allow us to expand it in the future by making it easier to understand and reusable.
2. **What is the importance of managing files?**

   Managing files is essential for automating tasks also as it helps us to be organized such as creating a own folder for tasks of the ansible playbook for that role. Managing backup for old files allows us to reuse it in the future if needed.

   Github Link: https://github.com/Moznaim/Act7-Ballesteros