Name: Tendencia, Jasmin Raiza S.	**Date Performed:** 10/09/2023
Course/Section: CPE232 - CPE31S4	**Date Submitted:** 10/09/2023
Instructor: Dr. Jonathan Taylar	**Semester and SY:** 1st/2023 - 2024

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.

```
tendencia@workstation:~/HOA7$ mkdir files
tendencia@workstation:~/HOA7$ cd files
tendencia@workstation:~/HOA7/files$ nano default_site.html
```

```
 GNU nano 6.2
<!DOCTYPE html>
<html>
<head>
    <title>Jr Pogi's Site</title>
</head>
<body>
    <h1>Welcome to Jr Pogi's Site</h1>
    <p>This is created for CPE232.</p>
</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
```

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

```
tendencia@workstation:~/HOA7$ ansible-playbook --ask-become-pass site.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.103]
ok: [192.168.56.104]
ok: [192.168.56.102]

TASK [install updates (CentOS)] ************************************************
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.104]

TASK [install updates (Ubuntu)] ************************************************
skipping: [192.168.56.104]
ok: [192.168.56.103]
ok: [192.168.56.102]

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.102]
ok: [192.168.56.104]

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

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

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

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

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.103]
ok: [192.168.56.104]

TASK [install mariadb package (CentOS)] ****************************************
skipping: [192.168.56.103]
ok: [192.168.56.104]

TASK [install mariadb package (Ubuntu)] ****************************************
skipping: [192.168.56.104]
ok: [192.168.56.103]

TASK [Mariadb - Restarting/Enabling] *******************************************
changed: [192.168.56.103]
changed: [192.168.56.104]

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.103]

TASK [install samba package] ***************************************************
ok: [192.168.56.103]

PLAY RECAP *********************************************************************
192.168.56.102             : ok=5    changed=1    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
192.168.56.103             : ok=7    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.104             : ok=9    changed=2    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
```

Observation: The results are either changed or ok for all the tasks intended for centOS, but the tasks intended for Ubuntu servers have been skipped. Also, the added code in the *site.yml*, this Ansible task is responsible for copying the "default_site.html" file from the local machine to the specified destination on the target machine while setting the owner, group, and permissions for the copied file.

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.
   Server 1:

```
tendencia@workstation:~/HOA7$ ssh 'tendencia@server1'
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-33-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

*** System restart required ***
Last login: Mon Oct  9 19:44:47 2023 from 192.168.56.101
tendencia@server1:~$ cat index.html
cat: index.html: No such file or directory
tendencia@server1:~$ cd /var/www/html/
tendencia@server1:/var/www/html$ ls
index.html
tendencia@server1:/var/www/html$ cat index.html
<!DOCTYPE html>
<html>
<head>
    <title>Jr Pogi's Site</title>
</head>
<body>
    <h1>Welcome to Jr Pogi's Site</h1>
    <p>This is created for CPE232.</p>
</body>
</html>
tendencia@server1:/var/www/html$
```

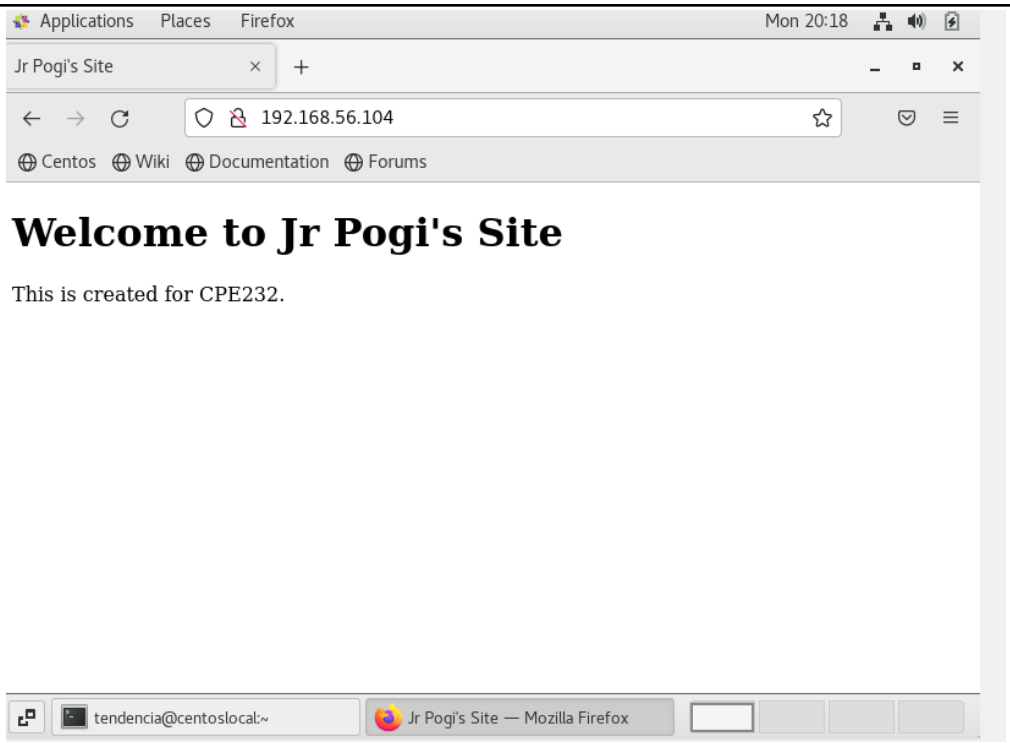
   Server 2:

```
tendencia@workstation:~/HOA7$ ssh 'tendencia@server2'
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-33-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

*** System restart required ***
Last login: Mon Oct  9 20:14:29 2023 from 192.168.56.101
tendencia@server2:~$ cd /var/www/html/
tendencia@server2:/var/www/html$ cat index.html
<!DOCTYPE html>
<html>
<head>
    <title>Jr Pogi's Site</title>
</head>
<body>
    <h1>Welcome to Jr Pogi's Site</h1>
    <p>This is created for CPE232.</p>
</body>
</html>
tendencia@server2:/var/www/html$
```

CentOS:

```
tendencia@workstation:~/HOA7$ ssh 'tendencia@centoslocal'
Last login: Mon Oct  9 19:45:30 2023 from 192.168.56.101
[tendencia@centoslocal ~]$ cd /var/www/html
[tendencia@centoslocal html]$ ls
index.html
[tendencia@centoslocal html]$ cat index.html
<!DOCTYPE html>
<html>
<head>
    <title>Jr Pogi's Site</title>
</head>
<body>
    <h1>Welcome to Jr Pogi's Site</h1>
    <p>This is created for CPE232.</p>
</body>
</html>
[tendencia@centoslocal html]$
```

CentOS browser:

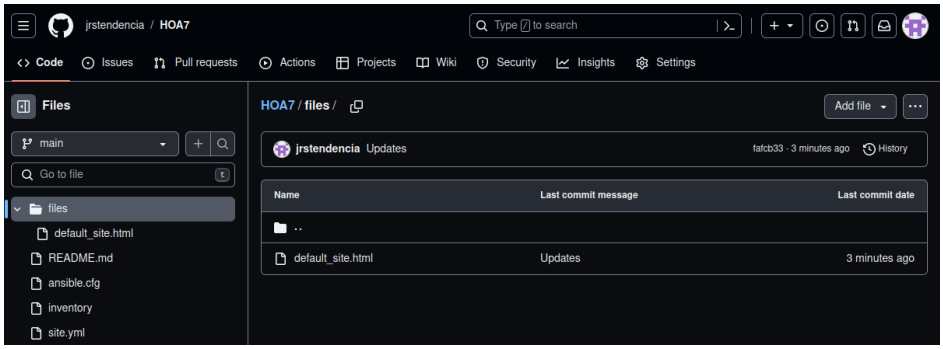


Observation: The content of index.html in each remote servers are the same as the content in the edited default_site.html. Also, the webpage in centOS displayed the same content.

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

```
tendencia@workstation:~/HOA7$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ansible.cfg
        files/
        inventory
        site.yml

nothing added to commit but untracked files present (use "git add" to track)
tendencia@workstation:~/HOA7$ git add ansible.cfg
tendencia@workstation:~/HOA7$ git add files/
tendencia@workstation:~/HOA7$ git add files
tendencia@workstation:~/HOA7$ git add inventory
tendencia@workstation:~/HOA7$ git add site.yml
tendencia@workstation:~/HOA7$ git commit -m "Updates"
[main fafcb33] Updates
 4 files changed, 118 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 files/default_site.html
 create mode 100644 inventory
 create mode 100644 site.yml
tendencia@workstation:~/HOA7$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
tendencia@workstation:~/HOA7$ git push origin main
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 2 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 1.25 KiB | 1.25 MiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:jrstendencia/HOA7.git
   b8d228c..fafcb33  main -> main
tendencia@workstation:~/HOA7$
```

Github repository:



Observation: It is noticeable here that the created directory named *files* is evident and under it, we can see the newly created file which is the *default_site.html*.

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

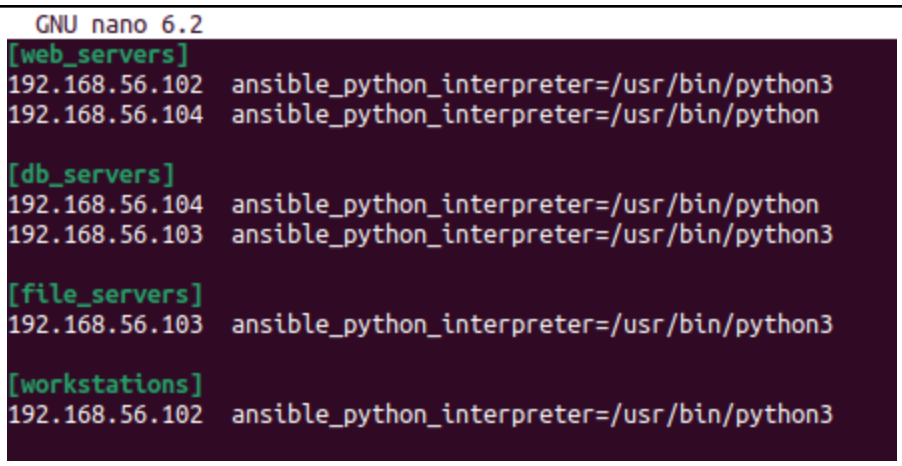
1. Edit the site.yml. Just before the web_servers play, create a new play:
   - hosts: workstations
     become: true
     tasks:

     - 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
```

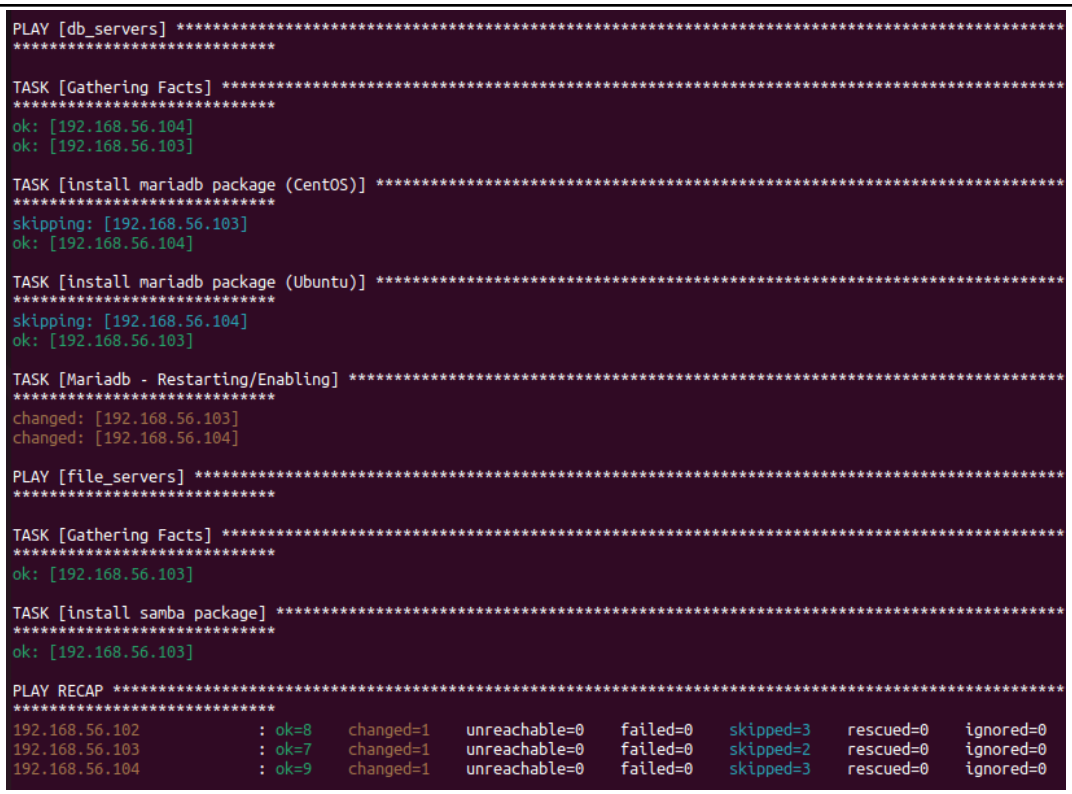
2. Edit the inventory file and add workstations group. Add any Ubuntu remote server. Make sure to remember the IP address.
   I added the server 1 under workstation group:



3. Run the playbook. Describe the output.

```
tendencia@workstation:~/HOA7$ ansible-playbook --ask-become-pass site.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
****************************
ok: [192.168.56.103]
ok: [192.168.56.102]
ok: [192.168.56.104]

TASK [install updates (CentOS)] ************************************************
****************************
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.104]

TASK [install updates (Ubuntu)] ************************************************
****************************
skipping: [192.168.56.104]
ok: [192.168.56.103]
ok: [192.168.56.102]

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

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

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

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

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

TASK [Gathering Facts] *********************************************************
****************************
ok: [192.168.56.102]
ok: [192.168.56.104]

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

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

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

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



Observation: Running the modified code of the playbook include Ansible task execution information, showing the progress of each task on the hosts in the *workstations* group. The tasks complete successfully, there are messages indicating that *unzip* and *Terraform* have been installed on the workstation hosts.

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

```
tendencia@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
```

Observation: It includes a list of available Terraform subcommands, it indicates that Terraform is installed and functioning correctly on my Ubuntu remote workstation. This is a valid and common way to verify the installation of 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)

```
---
- 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"

- 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.

```
tendencia@workstation:~/HOA7$ mkdir roles
```

```
tendencia@workstation:~/HOA7$ cd roles
tendencia@workstation:~/HOA7/roles$ mkdir base web_servers file_servers db_servers workstations
```

```
tendencia@workstation:~/HOA7/roles$ cd base
tendencia@workstation:~/HOA7/roles/base$ mkdir tasks
```

```
tendencia@workstation:~/HOA7/roles$ cd web_servers
tendencia@workstation:~/HOA7/roles/web_servers$ mkdir tasks
```

```
tendencia@workstation:~/HOA7/roles$ cd file_servers
tendencia@workstation:~/HOA7/roles/file_servers$ mkdir tasks
```

```
tendencia@workstation:~/HOA7/roles$ cd db_servers
tendencia@workstation:~/HOA7/roles/db_servers$ mkdir tasks
```

```
tendencia@workstation:~/HOA7/roles$ cd workstations
tendencia@workstation:~/HOA7/roles/workstations$ mkdir tasks
```

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:

```
tendencia@workstation:~/HOA7/roles/base$ nano main.yml
```

```yaml
---

- hosts: all
  become: true
  pre_tasks:

  - 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"
```

   workstations:

```
tendencia@workstation:~/HOA7/roles/workstations$ nano main.yml
```

```yaml
---

- 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
```

   web_servers:

```
tendencia@workstation:~/HOA7/roles/web_servers$ nano main.yml
```

```
---

- 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

  - 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
      enabled: true
    when: ansible_distribution == "CentOS"
```

db_servers:

```
tendencia@workstation:~/HOA7/roles/db_servers$ nano main.yml
```

```
---

- hosts: db_servers
  become: true
  tasks:

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

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

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

file_servers:

```
tendencia@workstation:~/HOA7/roles/file_servers$ nano main.yml
```

```
---

- hosts: file_servers
  become: true
  tasks:

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

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

```
tendencia@workstation:~/HOA7$ ansible-playbook --ask-become-pass site.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.103]
ok: [192.168.56.104]
ok: [192.168.56.102]

TASK [update repository index (CentOS)] ****************************************************************************
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.104]

TASK [install updates (Ubuntu)] ************************************************************************************
skipping: [192.168.56.104]
ok: [192.168.56.103]
ok: [192.168.56.102]

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

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.102]
ok: [192.168.56.103]
ok: [192.168.56.104]

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

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

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

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.104]
ok: [192.168.56.102]

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

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.103]
ok: [192.168.56.104]

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

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.103]

PLAY RECAP *********************************************************************************************************
192.168.56.102             : ok=5    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.103             : ok=5    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.104             : ok=5    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

Observation: The output provide information about which tasks were executed, whether they were successful, and any changes made during the playbook run. It also listed servers for which tasks were skipped if they do not belong to the respective roles.

Reflections:

Answer the following:

1. What is the importance of creating roles?
    - Roles in Ansible play a pivotal role in enhancing automation practices. It introduce modularity and reusability by breaking down complex tasks into reusable components, thereby simplifying playbook maintenance. Roles also provide an organized structure for playbook code, enhancing code readability through the separation of infrastructure aspects into distinct roles. This approach results in more concise playbooks, as roles reference specific

functionality, aiding comprehension and focus. Moreover, roles can be effortlessly shared across projects and the Ansible community, fostering collaboration and expediting automation efforts. Lastly, role dependencies enable structured orchestration of automation tasks, ensuring that roles are executed in the correct order, optimizing the automation workflow.

2. What is the importance of managing files?

    - Effective file management is pivotal in infrastructure automation, encompassing various vital aspects. Configuration management ensures that target servers maintain desired configuration file states, ensuring adherence to infrastructure policies and maintaining consistency. Version control, using systems like Git, permits change tracking, collaborative work, and rollback capabilities, guaranteeing comprehensive documentation and auditability of infrastructure configurations. File management extends to critical data and configuration backups, essential for disaster recovery and rapid service restoration in failure scenarios. Proper file management, including permissions and data security, helps prevent unauthorized access and data breaches. Scalability demands consistent file management across multiple servers, achievable with automation tools like Ansible, which apply uniform configurations, reducing configuration discrepancies. Automation enhances file-related tasks, ensuring speed and accuracy, elevating operational efficiency, and mitigating the risk of human errors.

Conclusion:

In this activity, I delved into the realm of infrastructure automation using Ansible, focusing on managing files and creating roles. These two fundamental aspects play a crucial role in streamlining and optimizing automation processes. I began by creating and copying a file to remote servers, demonstrating my ability to customize configurations and install default files efficiently. This showcased Ansible's power in orchestrating file operations across multiple servers. Next, I explored the concept of roles, a cornerstone of Ansible's organization and modularity. Roles enable me to break down complex automation tasks into reusable components, improving code organization, and enhancing playbook maintainability. By implementing roles, I streamlined my playbooks, making them more concise and focused, ultimately improving my automation workflow.

Throughout the activity, I learned the importance of file management in infrastructure automation. Efficient file management is essential for maintaining configuration consistency, version control, backup and recovery, security, scalability, and overall

operational efficiency. Ansible serves as a valuable tool for automating file-related tasks, ensuring reliability and reducing the likelihood of human errors.