Name: Agpaoa, Ma.Diane J.	**Date Performed:** 10/11/2022
Course/Section: CPE232 - CPE31S22	**Date Submitted:** 10/11/2022
Instructor: Dr. Jonathan Taylar	**Semester and SY:** 1st Sem 2022-2023

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.

```
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible$ ls
ansible.cfg  files  inventory  site.yml
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible$ cd files
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/files$ nano default_s
ite.html
```

Figure 1.1 Creating the files directory

I created the directory and named it files by executing the command mkdir files.

```
  GNU nano 6.2                    default_site.html
<html>
<body>

<h1>Hello World! </h1>

</body>
</html>
```

Figure 1.2 Basic HTML syntax inside the default_site.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
```

```
madiane@workstation: ~/CPE232_Agpaoa-Ma.Diane/hoa7_a...
GNU nano 6.2                                site.yml

- 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
```

Figure 1.3 Adding the code shown in Task 1 Step 2

I edited the site.yml and added the code shown in the image just below the web_servers play.

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

```
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible$ ansible-playbook --tags "httpd" --ask-become-pass site.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************************************************
fatal: [192.168.56.109]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.109 port 22: No route to host", "unreac
hable": true}
fatal: [192.168.56.118]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.118 port 22: No route to host", "unreac
hable": true}
ok: [192.168.56.120]
ok: [192.168.56.103]

TASK [install updates (CentOS)] ************************************************************************************
skipping: [192.168.56.103]
ok: [192.168.56.120]

TASK [install updates (Ubuntu)] ************************************************************************************
skipping: [192.168.56.120]
ok: [192.168.56.103]

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

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.120]
ok: [192.168.56.103]

TASK [install apache and php for CentOS servers] *******************************************************************
skipping: [192.168.56.103]
ok: [192.168.56.120]

TASK [start httpd (CentOS)] ****************************************************************************************
skipping: [192.168.56.103]
changed: [192.168.56.120]

TASK [copy default html file for site] *****************************************************************************
ok: [192.168.56.120]
changed: [192.168.56.103]

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

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

PLAY RECAP *********************************************************************************************************
192.168.56.103             : ok=4    changed=1    unreachable=0    failed=0    skipped=3    rescued=0    ignored=0
192.168.56.109             : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.118             : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.120             : ok=6    changed=1    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

Figure 1.4 Running the playbook site.yml

Based on the results the task of “copy the default_html file for site” is successful, since the state shown was “changed” and ”ok”.

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.

```
madiane@server2:~$ cat /var/www/html/index.html
<html>
<body>

<h1>Hello World! </h1>

</body>
</html>
```

Figure 1.5 Checking the content of index.html in Ubuntu

I checked if the index.html is the same as the local repository file (default_site.html) by executing the command “cat /var/www/html/index.html”.

```
[madiane@localhost ~]$ cat /var/www/html/index.html
<html>
<body>

<h1>Hello World! </h1>

</body>
</html>
```

Figure 1.6 Checking the content of index.html in CentOS

I checked if the index.html is the same as the local repository file (default_site.html) by executing the command “cat /var/www/html/index.html”.

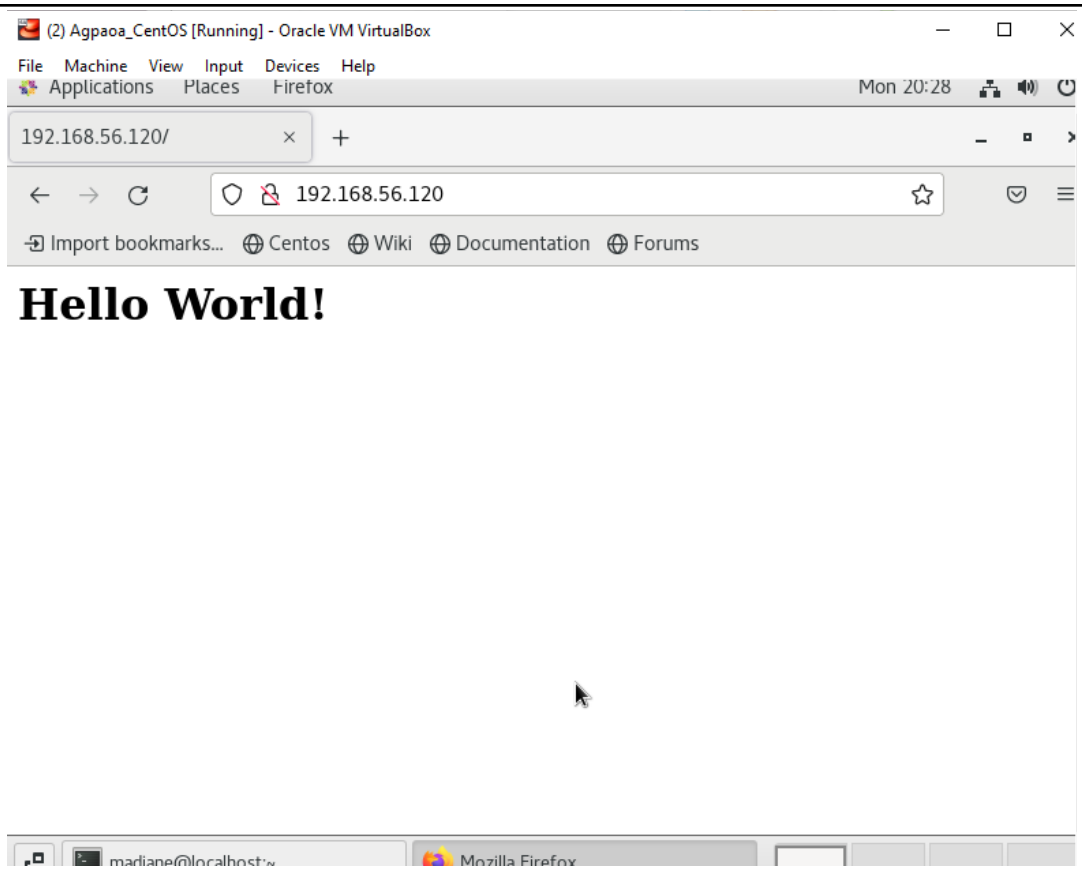


Figure 1.7 Typing the IP address of the CentOS in the browser

Based on the result, the output of the page will show the equivalent result of the syntax inside the index.html.

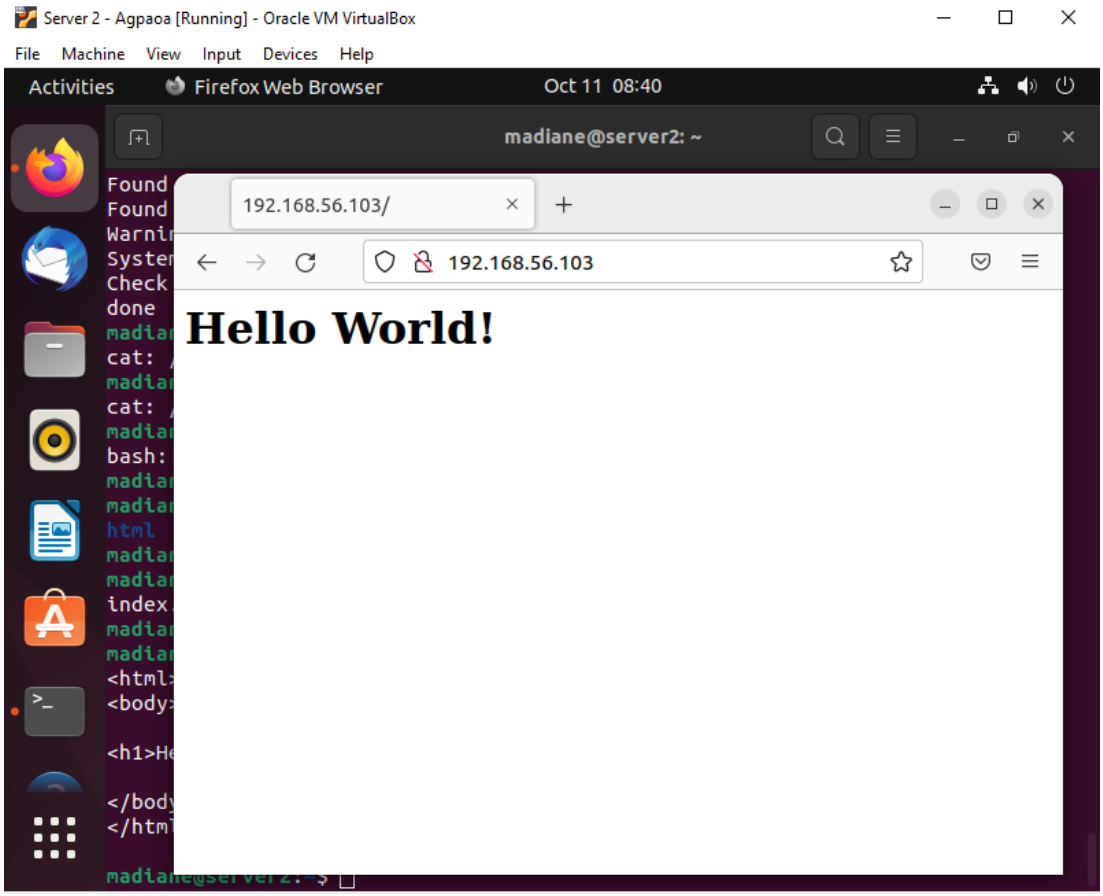


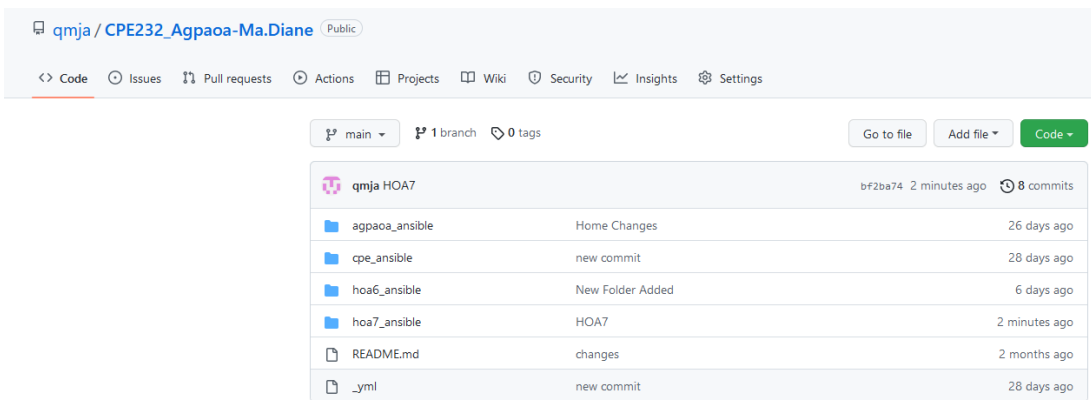
Figure 1.8 Typing the IP address of the Ubuntu in the browser

Based on the result, the output of the page will show the equivalent result of the syntax inside the index.html.

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

```
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane$ git add hoa7_ansible
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane$ git commit -m "HOA7"
[main bf2ba74] HOA7
 4 files changed, 119 insertions(+)
 create mode 100644 hoa7_ansible/ansible.cfg
 create mode 100644 hoa7_ansible/files/default_site.html
 create mode 100644 hoa7_ansible/inventory
 create mode 100644 hoa7_ansible/site.yml
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane$ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 1.18 KiB | 604.00 KiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:qmja/CPE232_Agpaoa-Ma.Diane.git
   c838a3a..bf2ba74  main -> main
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane$
```

Figure 1.9 Adding the new changes to Github repository



After the executing the commands to add the new changes in the repository, I looked into the Github repository and found that the new changes was implemented in the repository.

GitHub Link: https://github.com/qmja/CPE232_Agpaoa-Ma.Diane.git

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

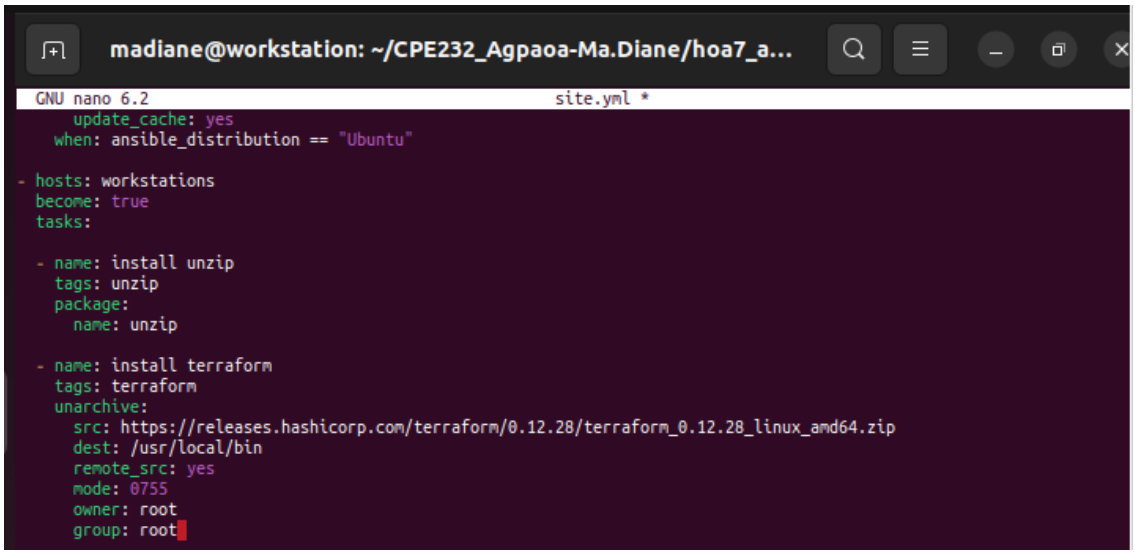


Figure 2.1 Adding the code shown in Task 2 Step 1

I edited the site.yml and added the code shown above just before the web_servers play. I created a new play and named it as workstations, then added the tasks that will install the unzip and terraform. I also added tags so I can specify the tasks that I want to run and avoid running the unnecessary tasks for this procedure.

2. Edit the inventory file and add workstations group. Add any Ubuntu remote server. Make sure to remember the IP address.

Figure 2.2 Creating new group named as workstations

Inside the inventory file I created a new group and named it as workstations then added the IP address of my Ubuntu remote server.

3. Run the playbook. Describe the output.

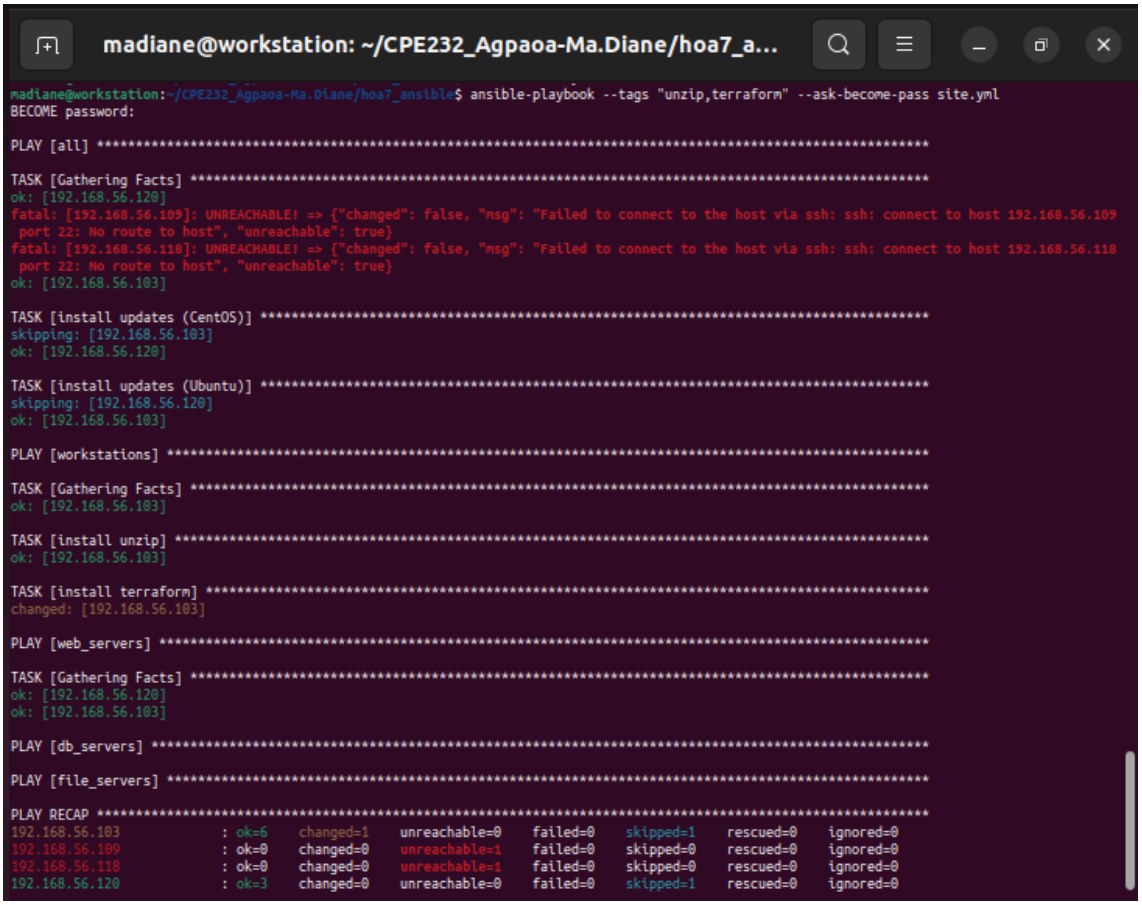


Figure 2.3 Running the playbook site.yml

Based on the results, installing the unzip and terraform is successful. The task for installing unzip has a state “ok”, which means that the unzip is already installed in the remote server. The next task is the installation of terraform that has a state “change” which means that the terraform was installed successfully.

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

```
madiane@server2:~$ 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
```

Figure 2.4 Verifying the installation of terraform

I execute the command “terraform” but since the syntax is not complete, the system shows the proper syntax and commands that can be used in order to use the terraform properly. With this result, we verified that the terraform is successfully installed in the Ubuntu remote 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.

```
madiane@workstation: ~/CPE232_Agpaoa-Ma.Diane/hoa7_a...
GNU nano 6.2                                                    site.yml
  - 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
```

Figure 3.1 Editing the site.yml

I edited the site.yml based on the code shown in Task 3 step 1. In addition, before changing the content of the site.yml, I created a copy as suggested in this procedure.

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.

```
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles$ mkdir workstations
```

```
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles$ mkdir base web_servers file_servers db_servers
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles$ ls
base  db_servers  file_servers  web_servers  workstations
```

```
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/base$ mkdir tasks
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/base$ ls
tasks
```

```
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles$ cd web_servers
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/web_servers$ mkdir tasks
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/web_servers$ cd tasks
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/web_servers/tasks$ nano main.yml
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/web_servers/tasks$ ls
main.yml
```

```
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/web_servers$ cp -r tasks ~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/file_servers
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/web_servers$ cp -r tasks ~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/db_servers
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/web_servers$ cp -r tasks ~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/workstations
madiane@workstation:~/CPE232_Agpaoa-Ma.Diane/hoa7_ansible/roles/web_servers$
```

Figure 3.2 Creating directories and creating the main.yml in advance

First I created a directory and named it as roles. Inside the roles directory, I created another new directories and these are base, web_servers, file_servers, db_servers and workstations. In one of these directories (base and web_servers), I created a new directory and named it as tasks. Inside the directory tasks I created the main.yml file. After that, I copy the task directory to the file_servers, db_servers and workstations directories.

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.

```
madiane@workstation: ~/CPE232_Agpaoa-Ma.Diane/hoa7_a...
GNU nano 6.2                                   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"
```

Figure 3.3 Contents of all tasks in the main.yml of base directory

```
madiane@workstation: ~/CPE232_Agpaoa-Ma.Diane/hoa7_a...
GNU nano 6.2                                        main.yml
- 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
  dnf:
    name:
      - httpd
      - php
    state: latest
  when: ansible_distribution == "CentOS"

- name: start httpd (CentOS)
  tags: apache, centos, httpd
  service:
    name: httpd
    state: restarted
    enabled: true
  when: ansible_distribution == "CentOS"

- 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
```

Figure 3.4 Contents of all tasks in the main.yml of web_servers directory

```
madiane@workstation: ~/CPE232_Agpaoa-Ma.Diane/hoa7_a...
GNU nano 6.2                                        main.yml
- name: install samba package
  tags: samba
  package:
    name: samba
    state: latest
```

Figure 3.5 Contents of all tasks in the main.yml of file_servers directory

```
madiane@workstation: ~/CPE232_Agpaoa-Ma.Diane/hoa7_a..
GNU nano 6.2                                    main.yml *
- 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"
```

Figure 3.6 Contents of all tasks in the main.yml of db_servers directory

```
madiane@workstation: ~/CPE232_Agpaoa-Ma.Diane/hoa7_a...
GNU nano 6.2                                    main.yml
- name: install unzip
  tags: unzip
  package:
    name: unzip

- name: install terraform
  tags: 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
```

Figure 3.7 Contents of all tasks in the main.yml of workstations directory

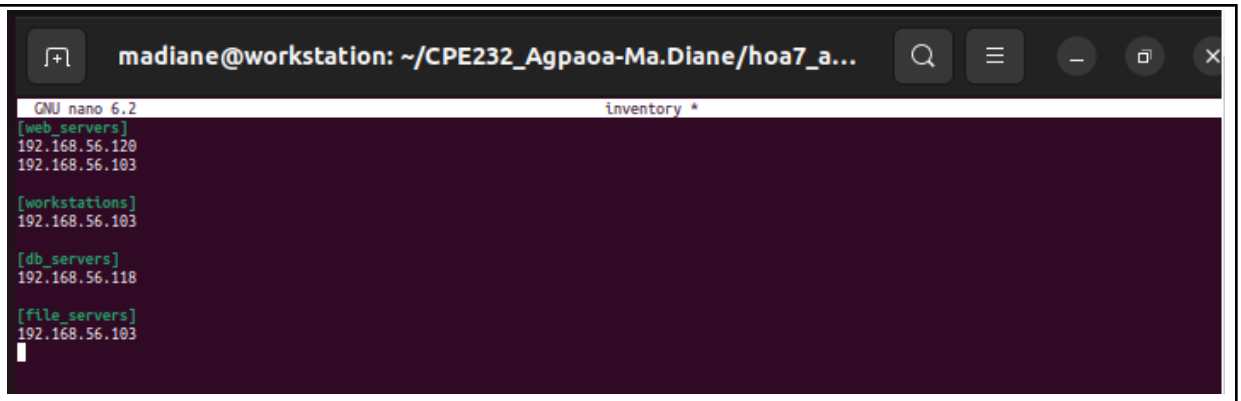


Figure 3.8 Contents of inventory file

Before running the site.yml playbook, I changed the inventory file based on the remote servers that I could use in this procedure.

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

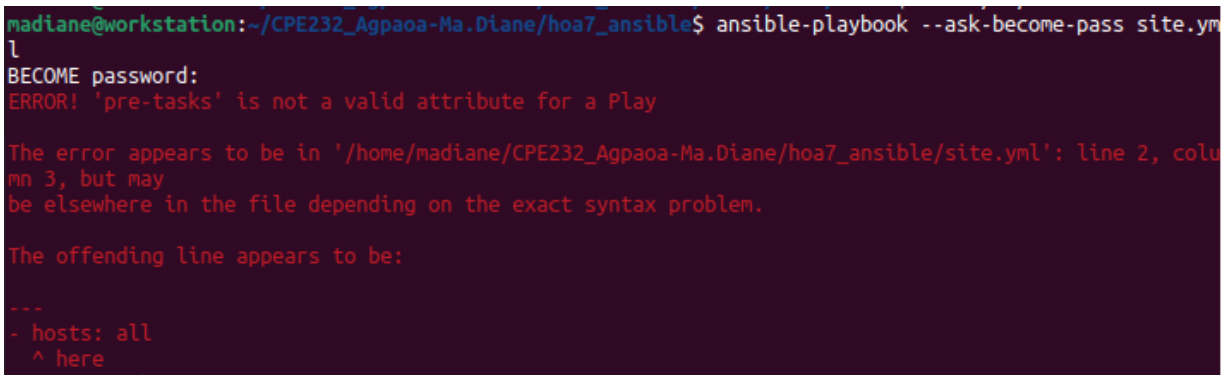


Figure 3.9 Running the site.yml playbook

The output showed an error saying that ‘pre-tasks’ is not a valid attribute for a Play.

Figure 3.10 Changing the recommended code in the Task 3 Step 1

In order to solve the error, I changed the site.yml from the recommended code in the Task 3 Step 1. I deleted the other all play since it can be already included in the play of “all”. The base contains similar tasks from the other play “all”, that is why I removed the first 2 code blocks and the play “all”.

```
madlane@workstation:~/CPE232_Agpaoa-Ma.Orlane/hw7_ansible$ ansible-playbook --ask-become-pass site.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.120]
ok: [192.168.56.103]
ok: [192.168.56.118]

TASK [base : install updates (CentOS)] *****************************************
skipping: [192.168.56.103]
skipping: [192.168.56.118]
ok: [192.168.56.120]

TASK [base : install updates (Ubuntu)] *****************************************
skipping: [192.168.56.120]
ok: [192.168.56.118]
ok: [192.168.56.103]

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

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

TASK [workstations : install unzip] ********************************************
ok: [192.168.56.103]

TASK [workstations : install terraform] ****************************************
ok: [192.168.56.103]

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.120]
ok: [192.168.56.103]

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.118]

TASK [db_servers : install mariadb package (CentOS)] ***************************
skipping: [192.168.56.118]

TASK [db_servers : Mariadb - Restarting/Enabling] ******************************
changed: [192.168.56.118]

TASK [db_servers : install mariadb package (Ubuntu)] ***************************
ok: [192.168.56.118]

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

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

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

PLAY RECAP *********************************************************************
192.168.56.103             : ok=8    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.118             : ok=5    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.120             : ok=3    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

Figure 3.11 Running the site.yml playbook

Based on the results, the tasks in main.yml from each directories inside the roles directories were successfully executed. The result of using the roles is almost the same as using the old site.yml. The advantage of using the roles is it can call out to the .yml files in different directories. In addition, using the roles would be beneficial in shortening the codes which will help with debugging errors easily.

Reflections:

Answer the following:

1. What is the importance of creating roles?

   Creating roles is important because roles can be used to reuse, isolate the codes, template, tasks, handlers, configuration files etc that consolidate our plays. Since roles isolate these sections it would be easier to locate the source of errors which will make the administrator's work faster and efficient.

2. What is the importance of managing files?

Managing files is important because we can modify, upload and retrieve files from specific remote servers. The files in our remote servers would be more organized and by managing files we could make our work faster and easier.

Conclusion:

In this activity, we implement the concept of managing files by copying a file from a local workstation to the remote server. In this activity, I customize the default website by creating a html that contains basic html syntax. Inside a .yml file I learned how to copy a file while setting the permissions. In addition, because of this activity I realized the importance of managing files. In managing files we can modify, upload and retrieve a file from a specific remote server, and these processes could make our work efficient.

In this activity, there's an implementation of the utilization of roles for consolidating the plays we created. In the Task 3 of this activity, I learned how to create roles and the importance of using and creating roles. By using roles, locating the errors would be much easier because we can isolate the codes that we will use to manage the remote servers. In addition, it is easier to share and reuse a specific task by using roles. These factors can make our management in remote servers easier and efficient.

In conclusion, this activity helped me to learn how to manage files in remote servers and implement roles in ansible. I also learned the importance of managing files and creating roles. Lastly, this activity also helped me to improve my skills in creating and debugging ansible playbooks.