Name: Clarence B. Chavez	**Date Performed:** September 30, 2024
Course/Section: CPE31S2	**Date Submitted:** September 30, 2024
Instructor: Engr. Robin Valenzuela	**Semester and SY:** 1st Sem, 2024-2025

Activity 6: Targeting Specific Nodes and Managing Services

1. **Objectives:**

1.1 Individualize hosts

1.2 Apply tags in selecting plays to run

1.3 Managing Services from remote servers using playbooks

2. **Discussion**:

In this activity, we try to individualize hosts. For example, we don't want apache on all our servers, or maybe only one of our servers is a web server, or maybe we have different servers like database or file servers running different things on different categories of servers and that is what we are going to take a look at in this activity.

We also try to manage services that do not automatically run using the automations in playbook. For example, when we install web servers or httpd for CentOS, we notice that the service did not start automatically.

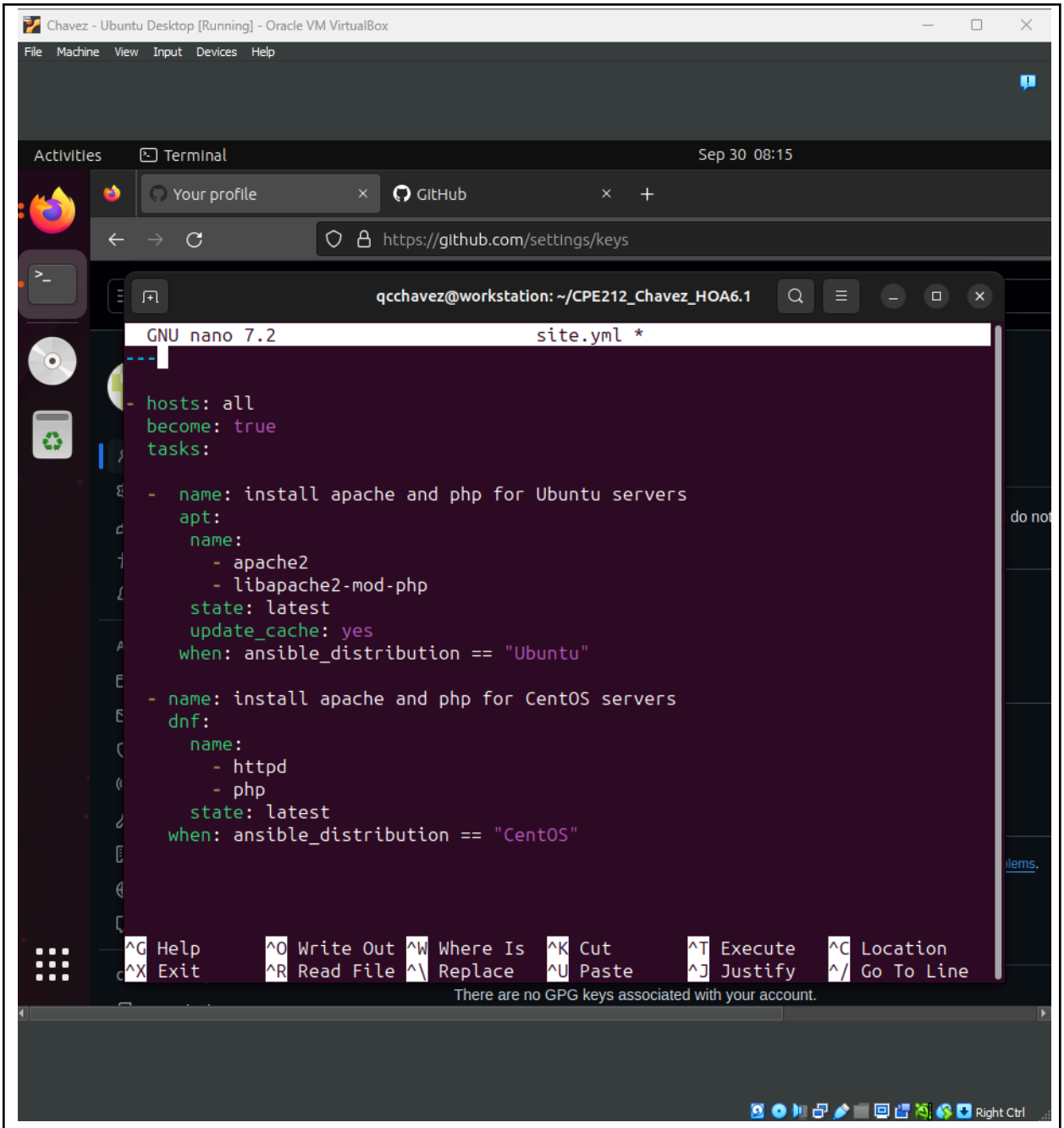
Requirement:

In this activity, you will need to create another Ubuntu VM and name it Server 3. Likewise, you need to activate the second adapter to a host-only adapter after the installations. Take note of the IP address of the Server 3. Make sure to use the command *ssh-copy-id* to copy the public key to Server 3. Verify if you can successfully SSH to Server 3.

Task 1: Targeting Specific Nodes

1. Create a new playbook and name it *site.yml*. Follow the commands as shown in the image below. Make sure to save the file and exit.

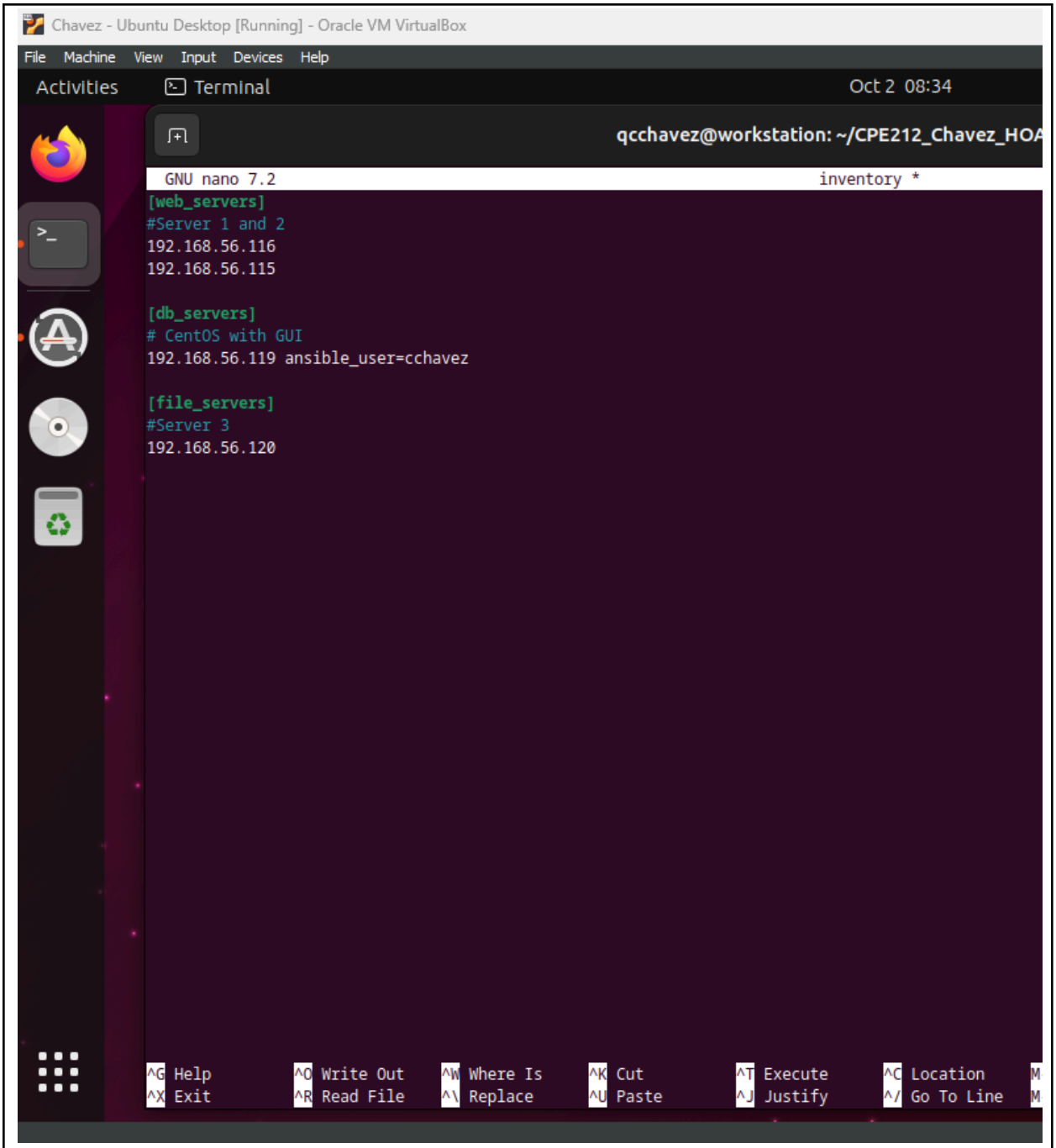
```
---

- hosts: all
  become: true
  tasks:

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

  - name: install apache and php for CentOS servers
    dnf:
      name:
        - httpd
        - php
      state: latest
    when: ansible_distribution == "CentOS"
```



- In this screenshot, I've added the necessary lines for the ansible playbook **site.yml**, the lines of codes that were added was about the installation of apache and php for both Ubuntu and CentOS servers that I have.

2. Edit the inventory file. Remove the variables we put in our last activity and group according to the image shown below:

Make sure to save the file and exit.



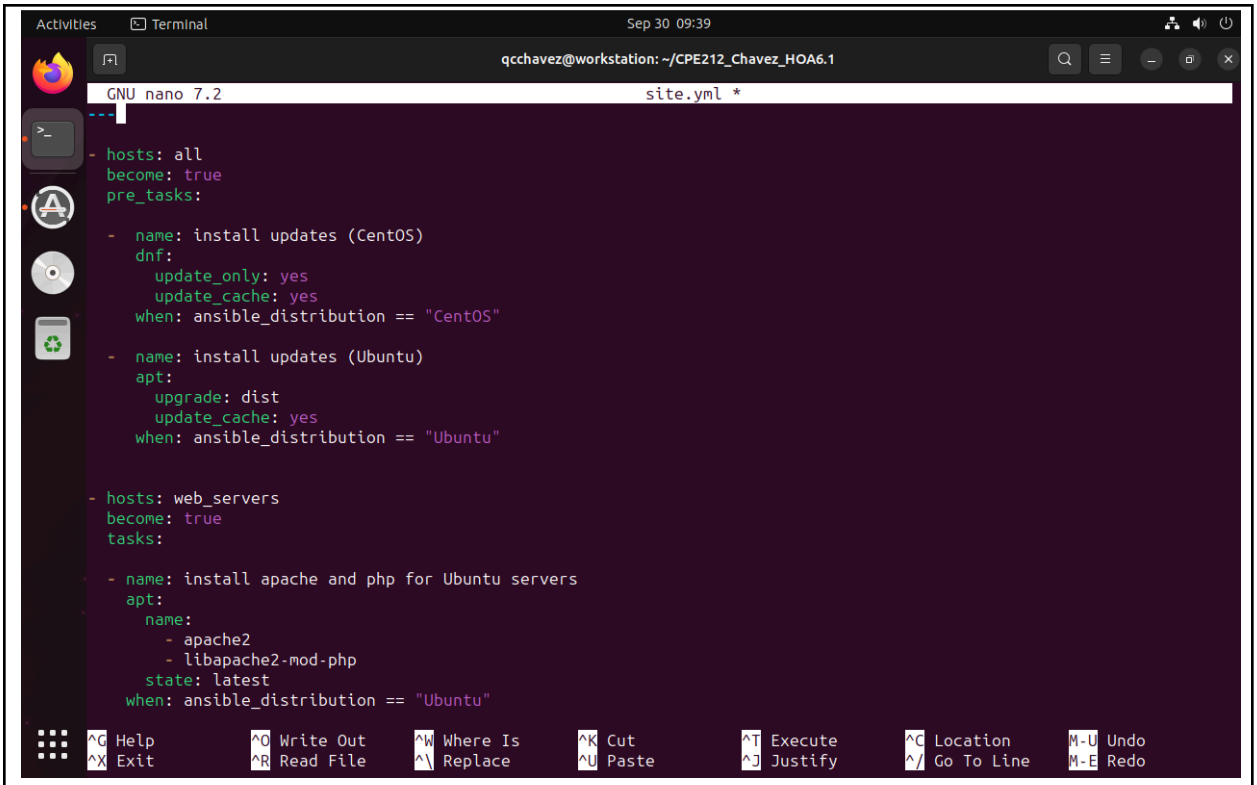
- In this screenshot, I've organized my remote servers' IP addresses to comply with the task, the **web_servers** group contains **Server 1 and 2**, the **db_servers** group contains **CentOS**, and the **file_servers** contains **Server 3**.

  Right now, we have created groups in our inventory file and put each server in its own group. In other cases, you can have a server be a member of multiple groups, for example you have a test server that is also a web server.

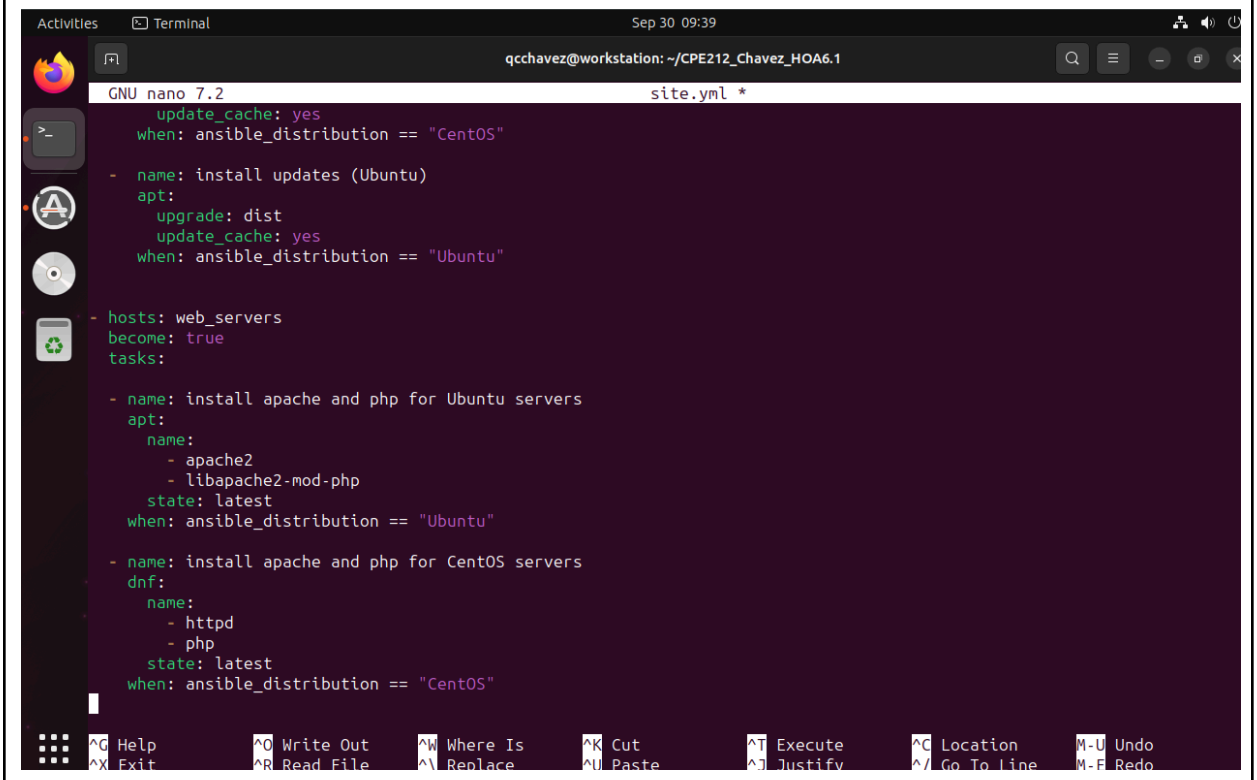
3. Edit the *site.yml* by following the image below:

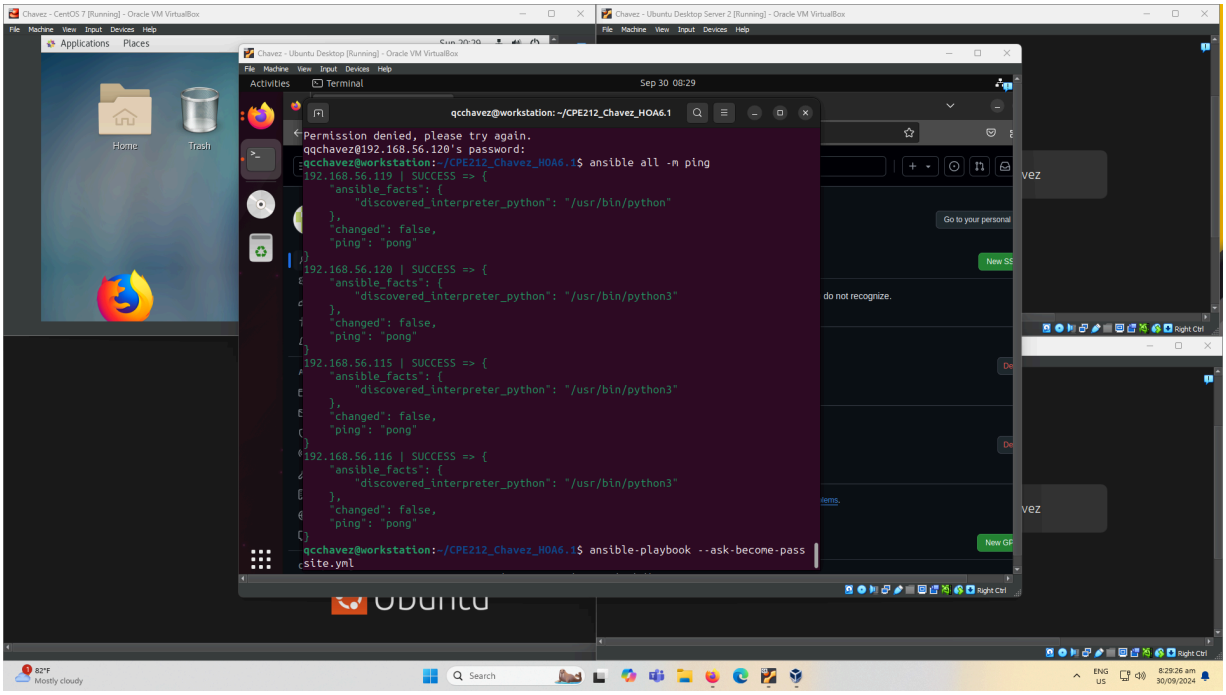
```
---

- hosts: all
  become: true
  pre_tasks:

  - name: install updates (CentOS)
    dnf:
      update_only: yes
      update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: install updates (Ubuntu)
    apt:
      upgrade: dist
      update_cache: yes
    when: ansible_distribution == "Ubuntu"


- hosts: web_servers
  become: true
  tasks:

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

  - name: install apache and php for CentOS servers
    dnf:
      name:
        - httpd
        - php
      state: latest
    when: ansible_distribution == "CentOS"
```

Make sure to save the file and exit.


Activities
Terminal
Sep 30 09:39
qcchavez@workstation: ~/CPE212_Chavez_HOA6.1
GNU nano 7.2
site.yml *

- hosts: all
become: true
pre_tasks:
- name: install updates (CentOS)
dnf:
update_only: yes
update_cache: yes
when: ansible_distribution == "CentOS"
- name: install updates (Ubuntu)
apt:
upgrade: dist
update_cache: yes
when: ansible_distribution == "Ubuntu"
- hosts: web_servers
become: true
tasks:
- name: install apache and php for Ubuntu servers
apt:
name:
- apache2
- libapache2-mod-php
state: latest
when: ansible_distribution == "Ubuntu"
^G Help
^O Write Out
^W Where Is
^K Cut
^T Execute
^C Location
M-U Undo
^X Exit
^R Read File
^\ Replace
^U Paste
^J Justify
^/ Go To Line
M-E Redo

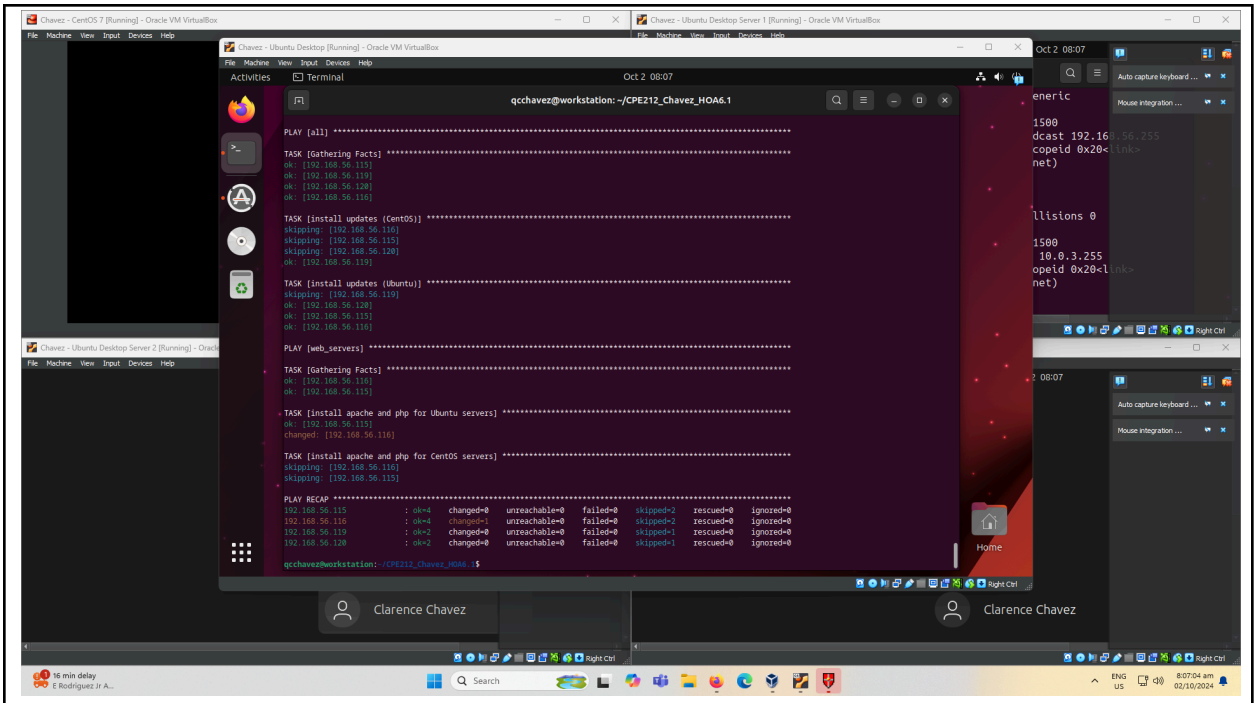

Activities
Terminal
Sep 30 09:39
qcchavez@workstation: ~/CPE212_Chavez_HOA6.1
GNU nano 7.2
site.yml *
update_cache: yes
when: ansible_distribution == "CentOS"
- name: install updates (Ubuntu)
apt:
upgrade: dist
update_cache: yes
when: ansible_distribution == "Ubuntu"
- hosts: web_servers
become: true
tasks:
- name: install apache and php for Ubuntu servers
apt:
name:
- apache2
- libapache2-mod-php
state: latest
when: ansible_distribution == "Ubuntu"
- name: install apache and php for CentOS servers
dnf:
name:
- httpd
- php
state: latest
when: ansible_distribution == "CentOS"
^G Help
^O Write Out
^W Where Is
^K Cut
^T Execute
^C Location
M-U Undo
^X Exit
^R Read File
^\ Replace
^U Paste
^J Justify
^/ Go To Line
M-E Redo



- In these screenshots, I've added the necessary lines again for the ansible playbook **site.yml**, the lines of codes that were added were about making the update of CentOS and Ubuntu as **pre_tasks**, and adding a task for the **web_servers** group. I also made sure that all of the remote servers are working successfully.

  The *pre-tasks* command tells the ansible to run it before any other thing. In the *pre-tasks*, CentOS will install updates while Ubuntu will upgrade its distribution package. This will run before running the second play, which is targeted at *web_servers*. In the second play, apache and php will be installed on both Ubuntu servers and CentOS servers.

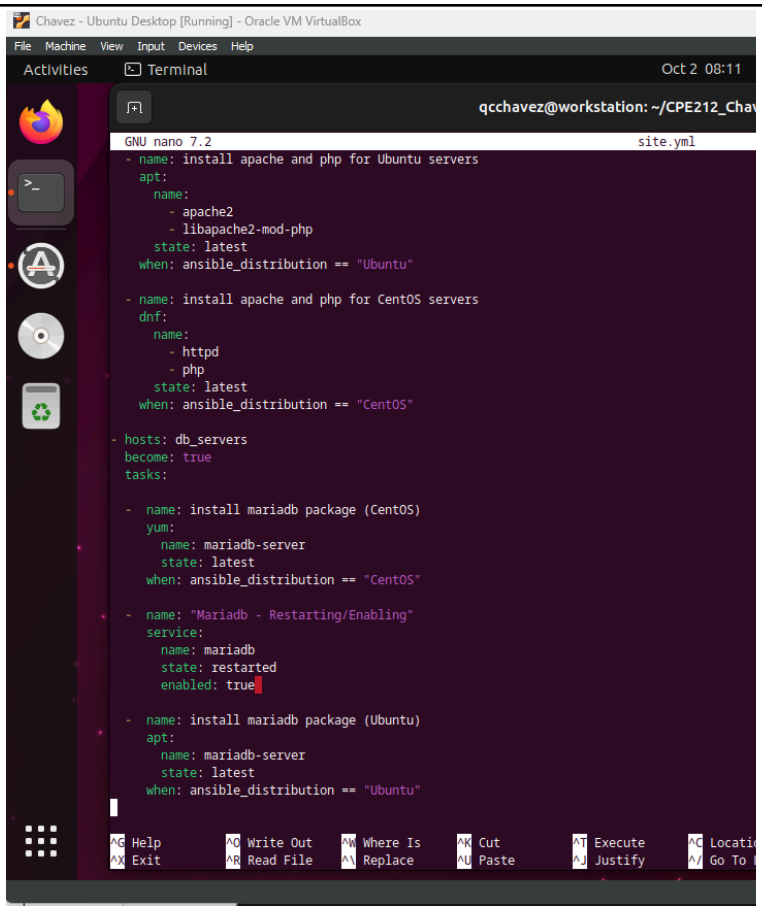
  Run the *site.yml* file and describe the result.



- In this screenshot, I've ran the **site.yml** playbook to comply with the task, it also shows that all of the tasks worked properly and made adjustments to the remote servers.

4. Let's try to edit again the *site.yml* file. This time, we are going to add plays targeting the other servers. This time we target the *db_servers* by adding it on the current *site.yml*. Below is an example: (Note add this at the end of the playbooks from task 1.3.

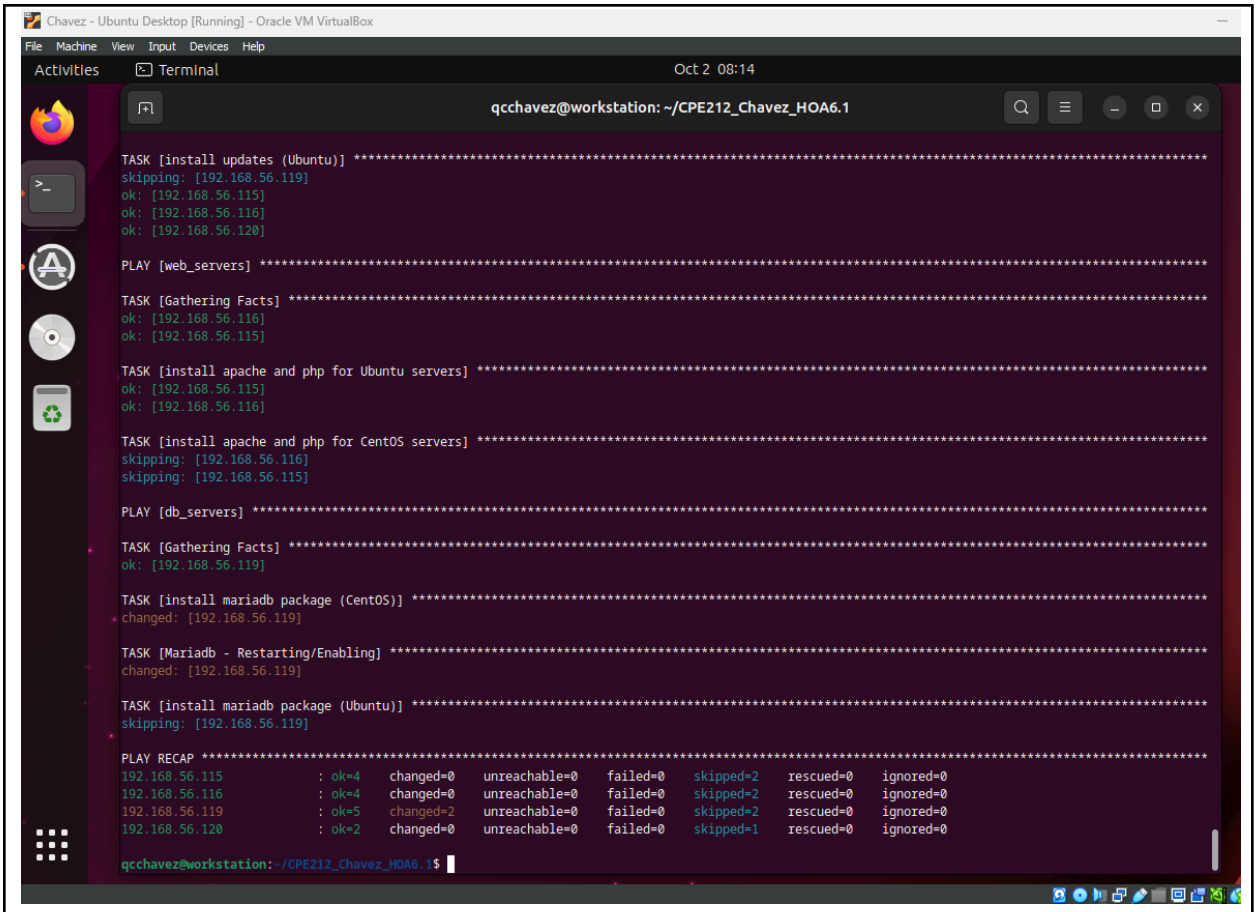
```
- hosts: db_servers
  become: true
  tasks:

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

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

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



- In this screenshot, I've added the tasks specifically for the **db_servers** group in the **site.yml** playbook.

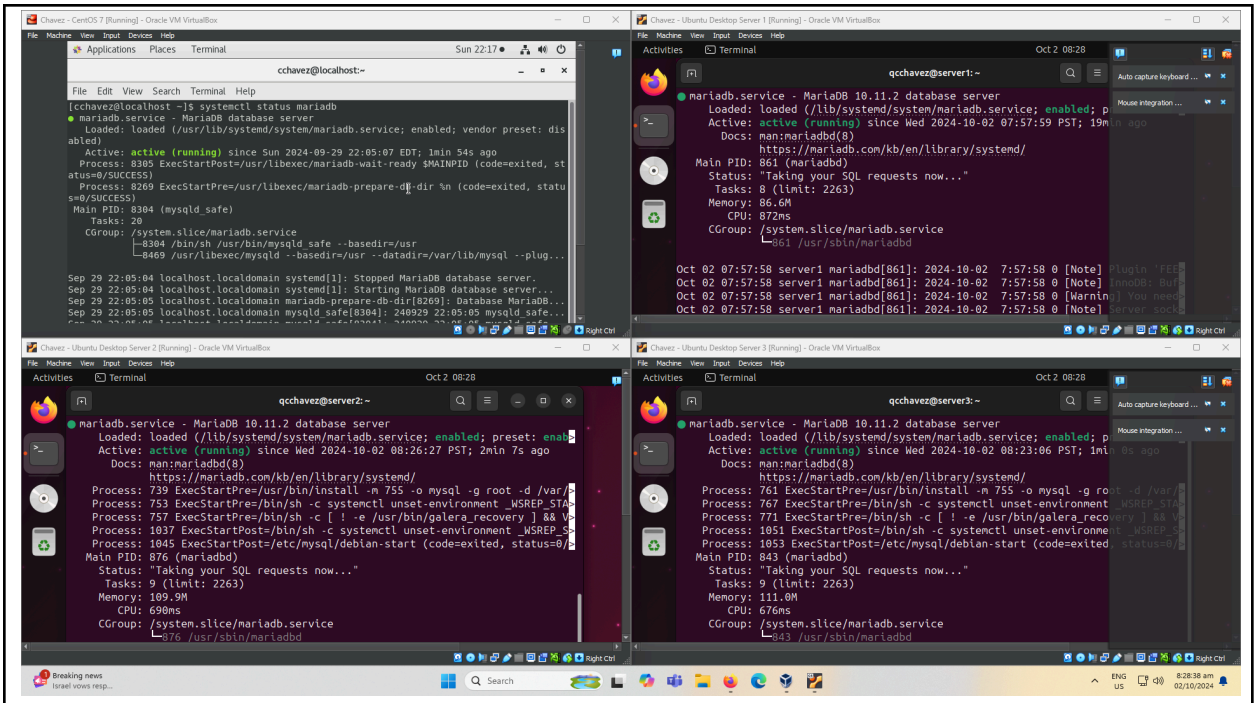
  Make sure to save the file and exit.

  Run the *site.yml* file and describe the result.



- In this screenshot, I've run the **site.yml** playbook to comply with the task, it shows that the tasks exclusively for the **db_servers** group works properly.

5. Go to the remote server (Ubuntu) terminal that belongs to the db_servers group and check the status for mariadb installation using the command: *systemctl status mariadb.* Do this on the CentOS server also.

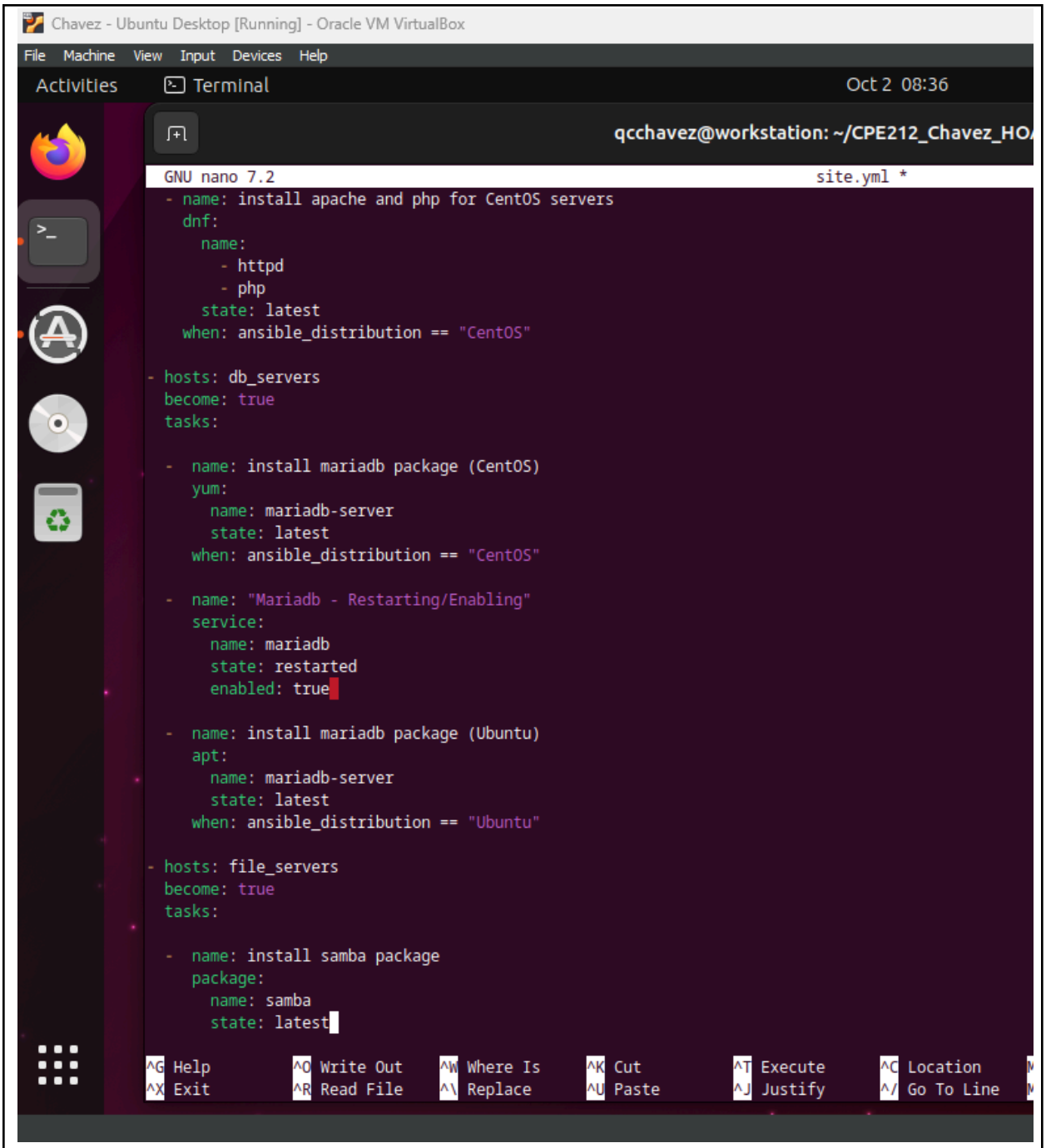


Describe the output.

- In this screenshot, I've prompted the command **sudo systemctl status mariadb** to check if the mariadb in the remote servers works properly.

6. Edit the *site.yml* again. This time we will append the code to configure installation on the *file_servers* group. We can add the following on our file.

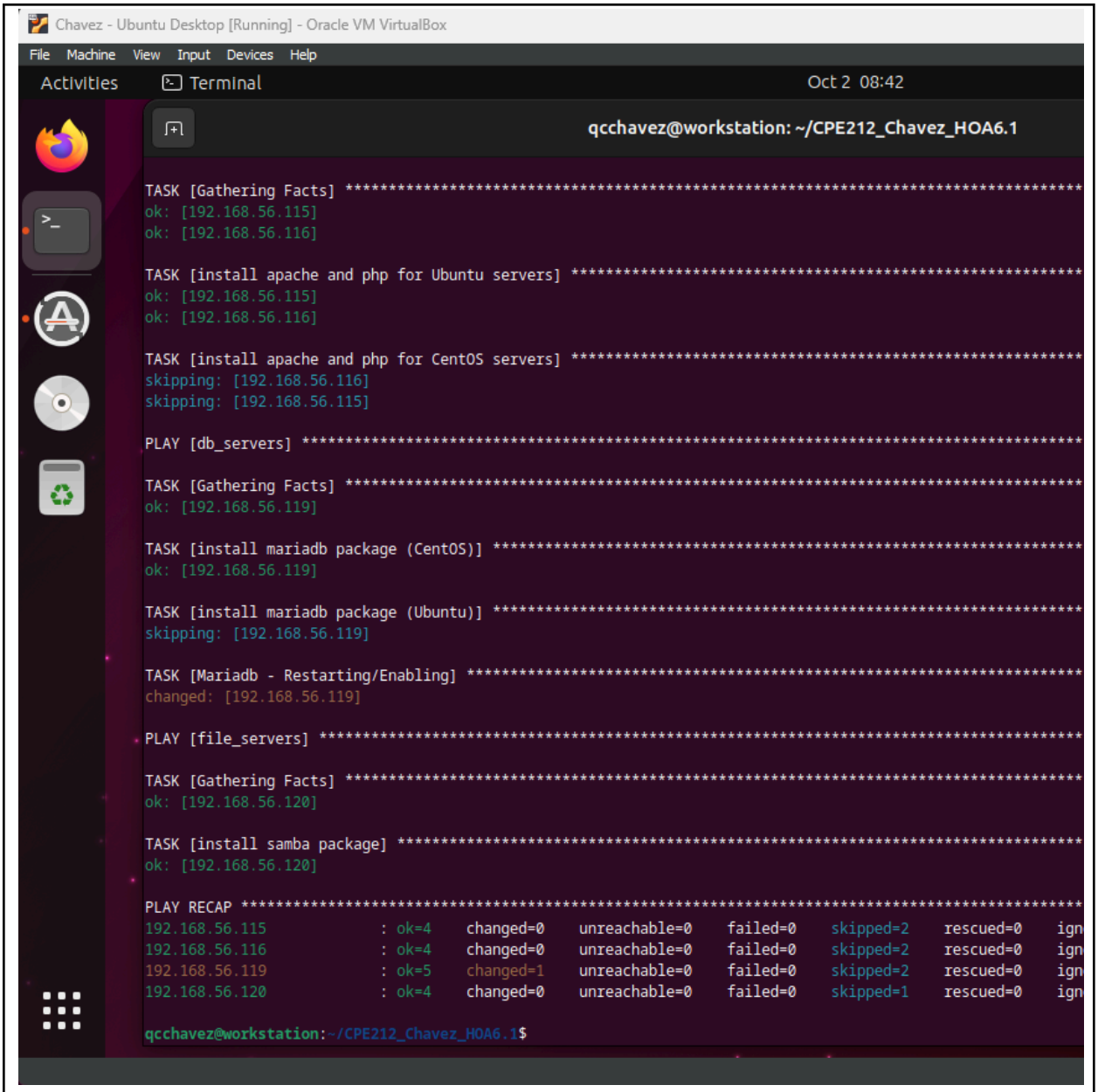
```
- hosts: file_servers
  become: true
  tasks:

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



- In this screenshot, I've added another tasks but this time, it is for the **file_servers** group.

  Make sure to save the file and exit.

  Run the *site.yml* file and describe the result.



- In this screenshot, I've ran the **site.yml** playbook to comply with the task, it also shows that all of the tasks worked properly for the **file_servers** group.

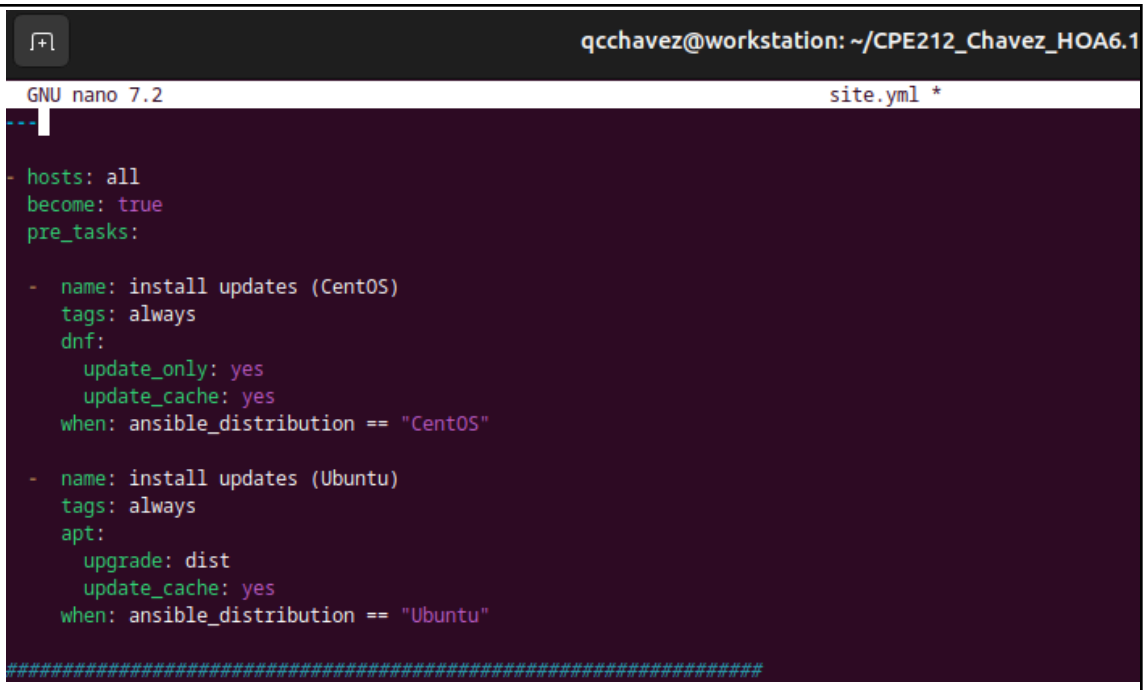
The testing of the *file_servers* is beyond the scope of this activity, and as well as our topics and objectives. However, in this activity we were able to show that we can target hosts or servers using grouping in ansible playbooks.

Task 2: Using Tags in running playbooks

In this task, our goal is to add metadata to our plays so that we can only run the plays that we want to run, and not all the plays in our playbook.

1. Edit the *site.yml* file. Add tags to the playbook. After the name, we can place the tags: *name_of_tag*. This is an arbitrary command, which means you can use any name for a tag.

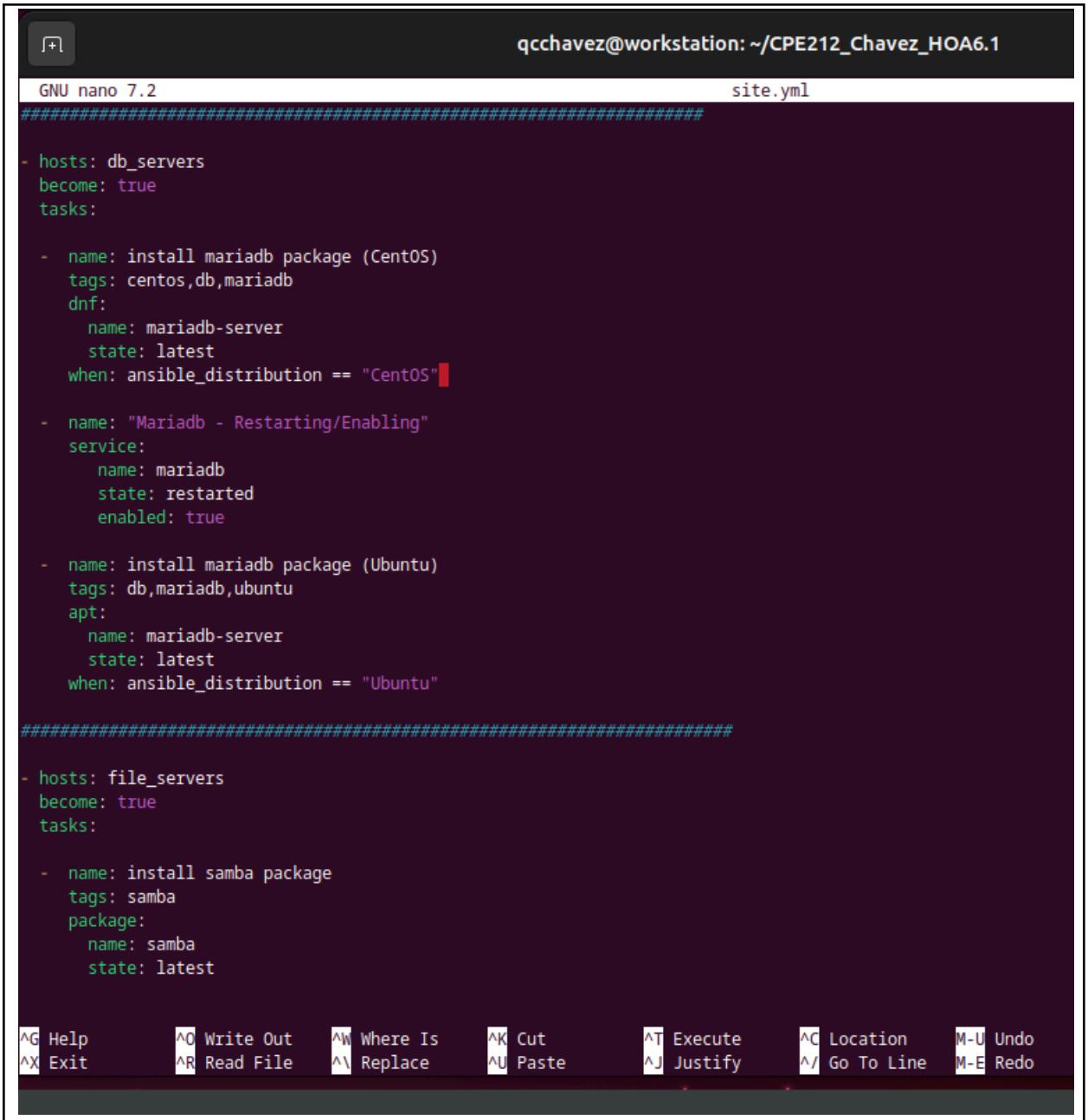
```
---

- hosts: all
  become: true
  pre_tasks:

  - 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"
```




```yaml
- hosts: web_servers
  become: true
  tasks:

  - 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"
```

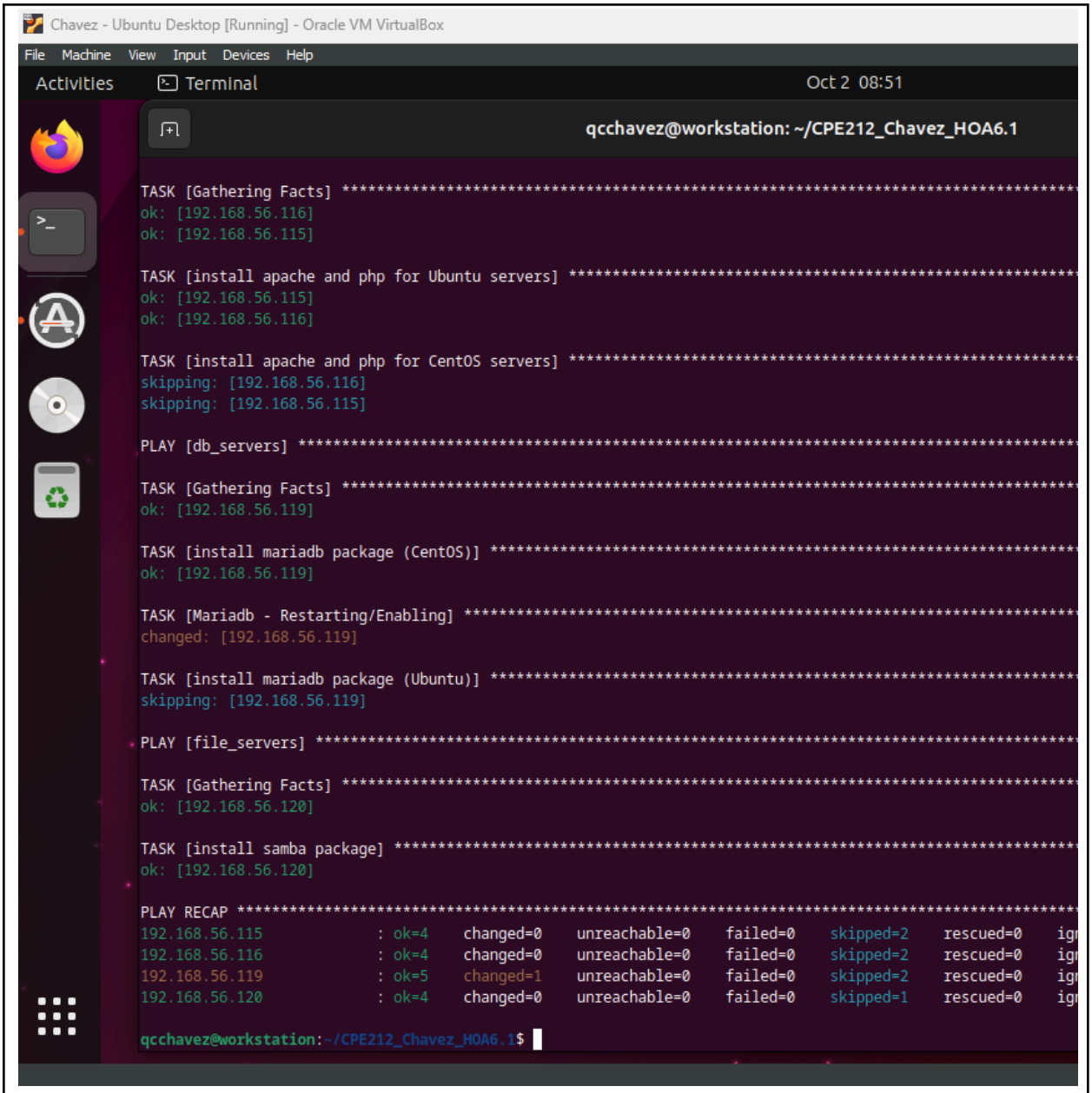
```
#############################################################

- hosts: web_servers
  become: true
  tasks:

  -  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"
```

```
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    tags: centos, db,mariadb
    dnf:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

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

  - name: install mariadb packege (Ubuntu)
    tags: db, mariadb,ubuntu
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"

- hosts: file_servers
  become: true
  tasks:

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



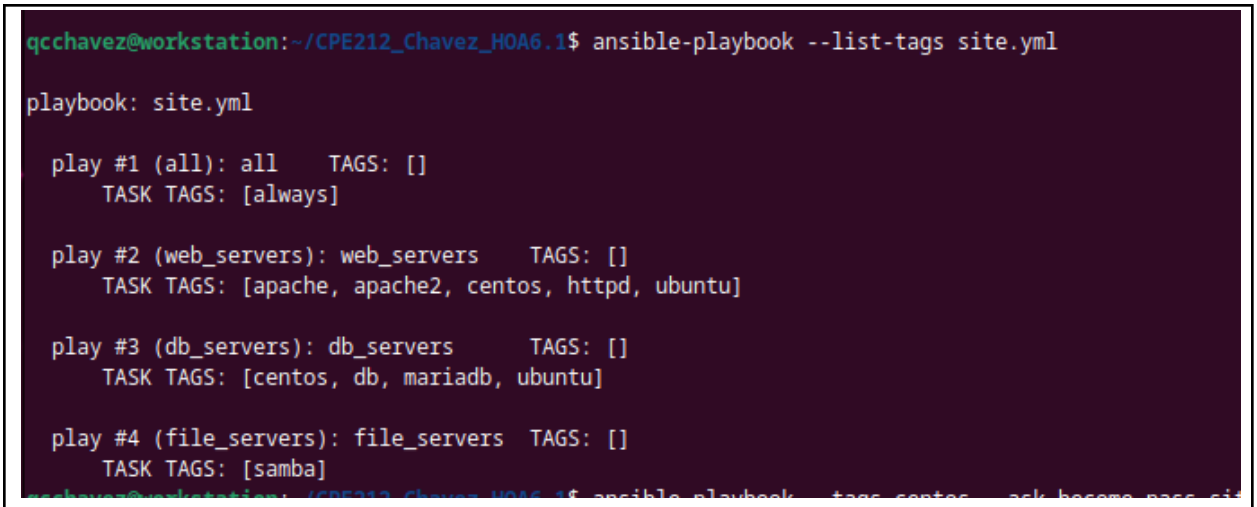
- In this screenshot, I've added tags in every host group in the **site.yml** file that will uniquely identify them, depending on the added tags.

Make sure to save the file and exit.
Run the *site.yml* file and describe the result.



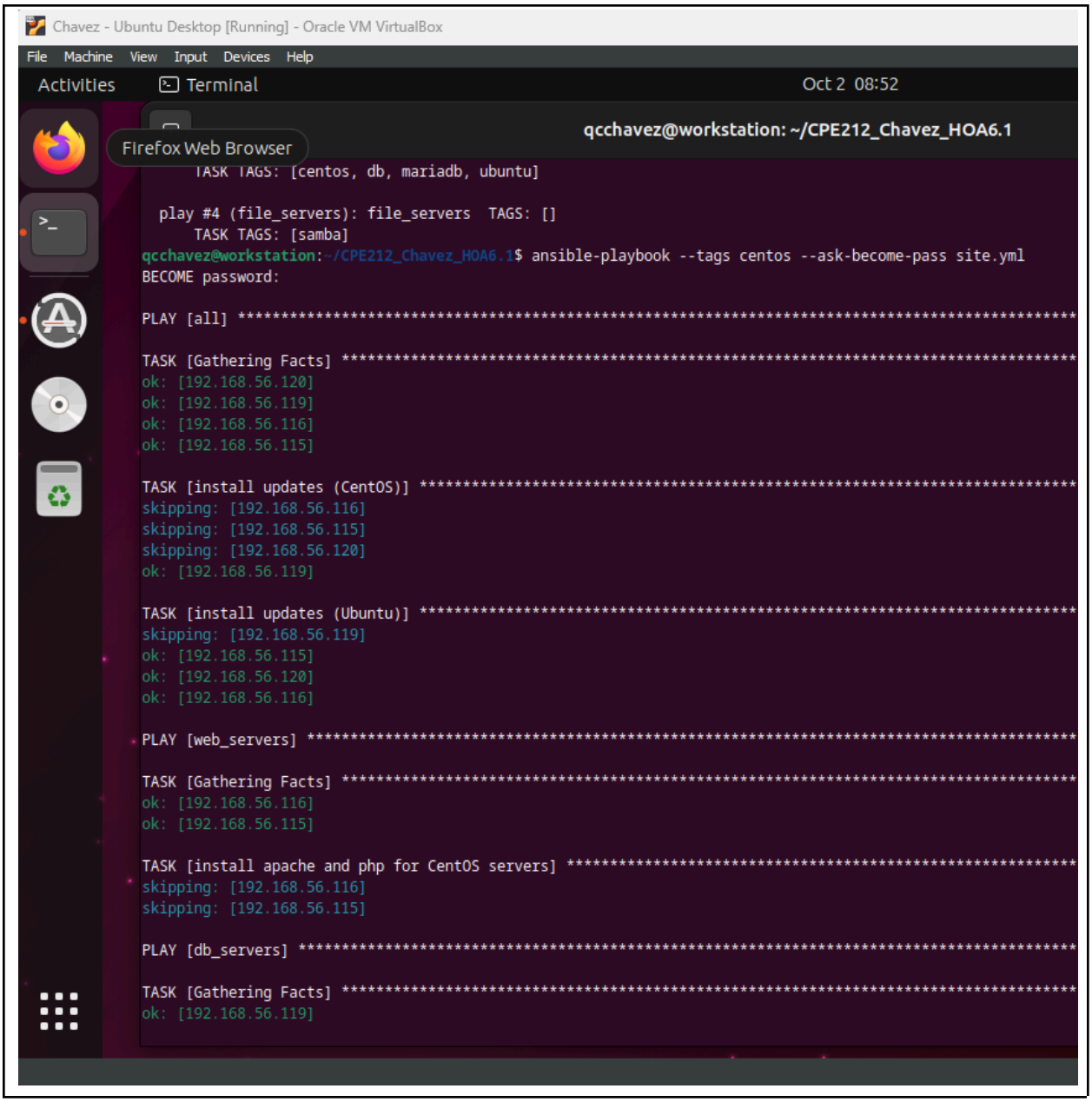
- In these screenshots, I've ran the **site.yml** playbook to comply with the task, the output shows that it installed the samba package on the **file_servers** group.

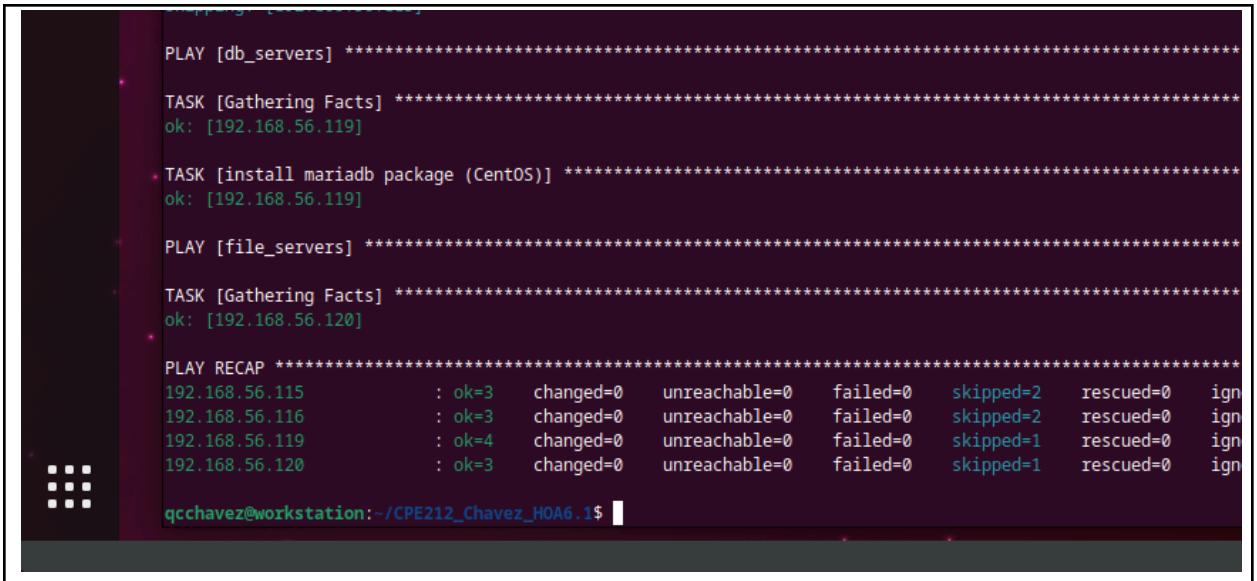
2. On the local machine, try to issue the following commands and describe each result:
   2.1 *ansible-playbook --list-tags site.yml*



- In this screenshot, it shows the tags in every hosts group.

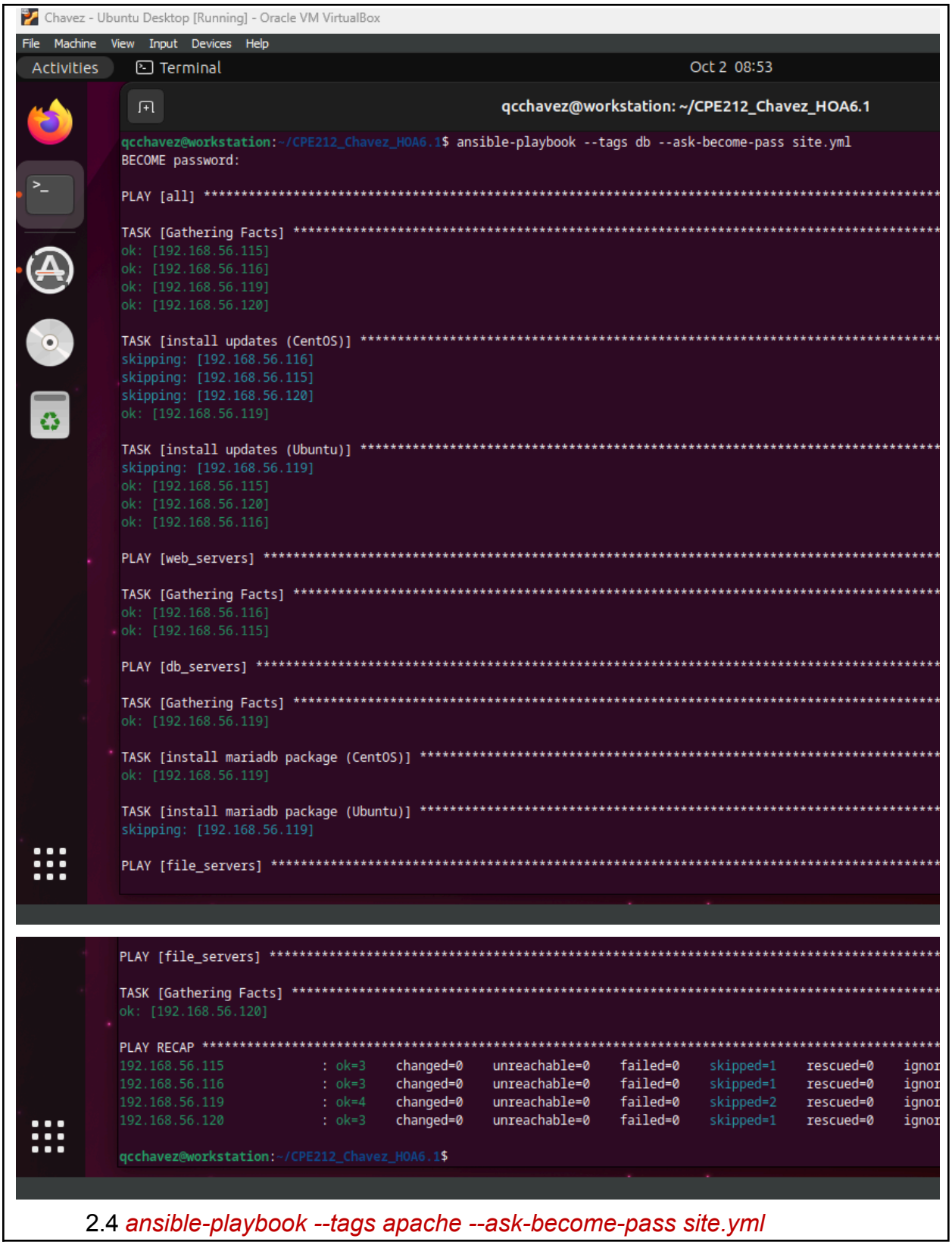
2.2 *ansible-playbook --tags centos --ask-become-pass site.yml*


Chavez - Ubuntu Desktop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Oct 2 08:52
Firefox Web Browser
qcchavez@workstation: ~/CPE212_Chavez_HOA6.1
TASK TAGS: [centos, db, mariadb, ubuntu]
play #4 (file_servers): file_servers TAGS: []
TASK TAGS: [samba]
qcchavez@workstation:~/CPE212_Chavez_HOA6.1$ ansible-playbook --tags centos --ask-become-pass site.yml
BECOME password:
PLAY [all] ***
TASK [Gathering Facts] ***
ok: [192.168.56.120]
ok: [192.168.56.119]
ok: [192.168.56.116]
ok: [192.168.56.115]
TASK [install updates (CentOS)] ***
skipping: [192.168.56.116]
skipping: [192.168.56.115]
skipping: [192.168.56.120]
ok: [192.168.56.119]
TASK [install updates (Ubuntu)] ***
skipping: [192.168.56.119]
ok: [192.168.56.115]
ok: [192.168.56.120]
ok: [192.168.56.116]
PLAY [web_servers] ***
TASK [Gathering Facts] ***
ok: [192.168.56.116]
ok: [192.168.56.115]
TASK [install apache and php for CentOS servers] ***
skipping: [192.168.56.116]
skipping: [192.168.56.115]
PLAY [db_servers] ***
TASK [Gathering Facts] ***
ok: [192.168.56.119]

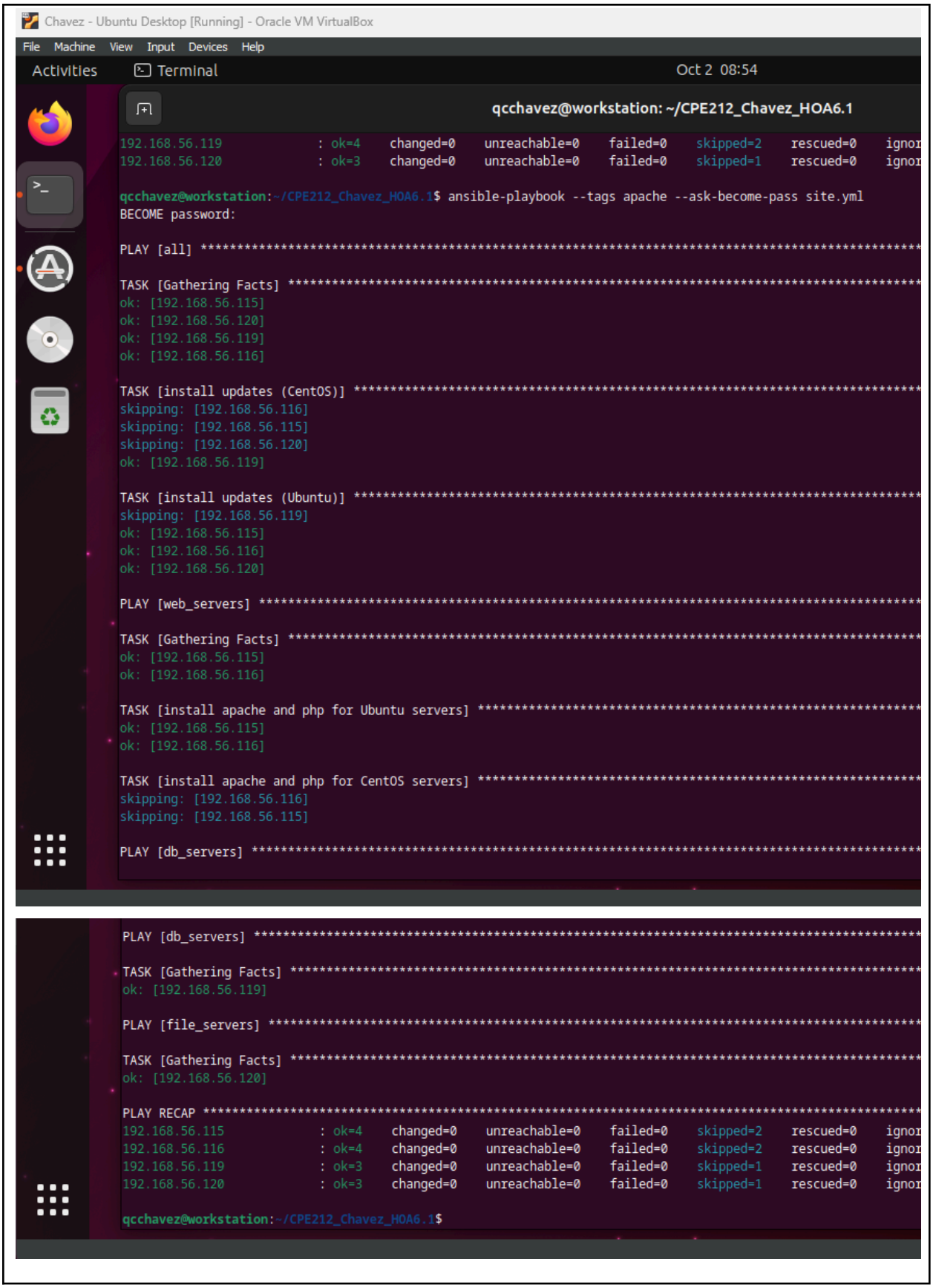


- In these screenshots, I've ran the **site.yml** playbook to comply with the task, the output shows that it only does the tasks where the **centos** tags exists**..**

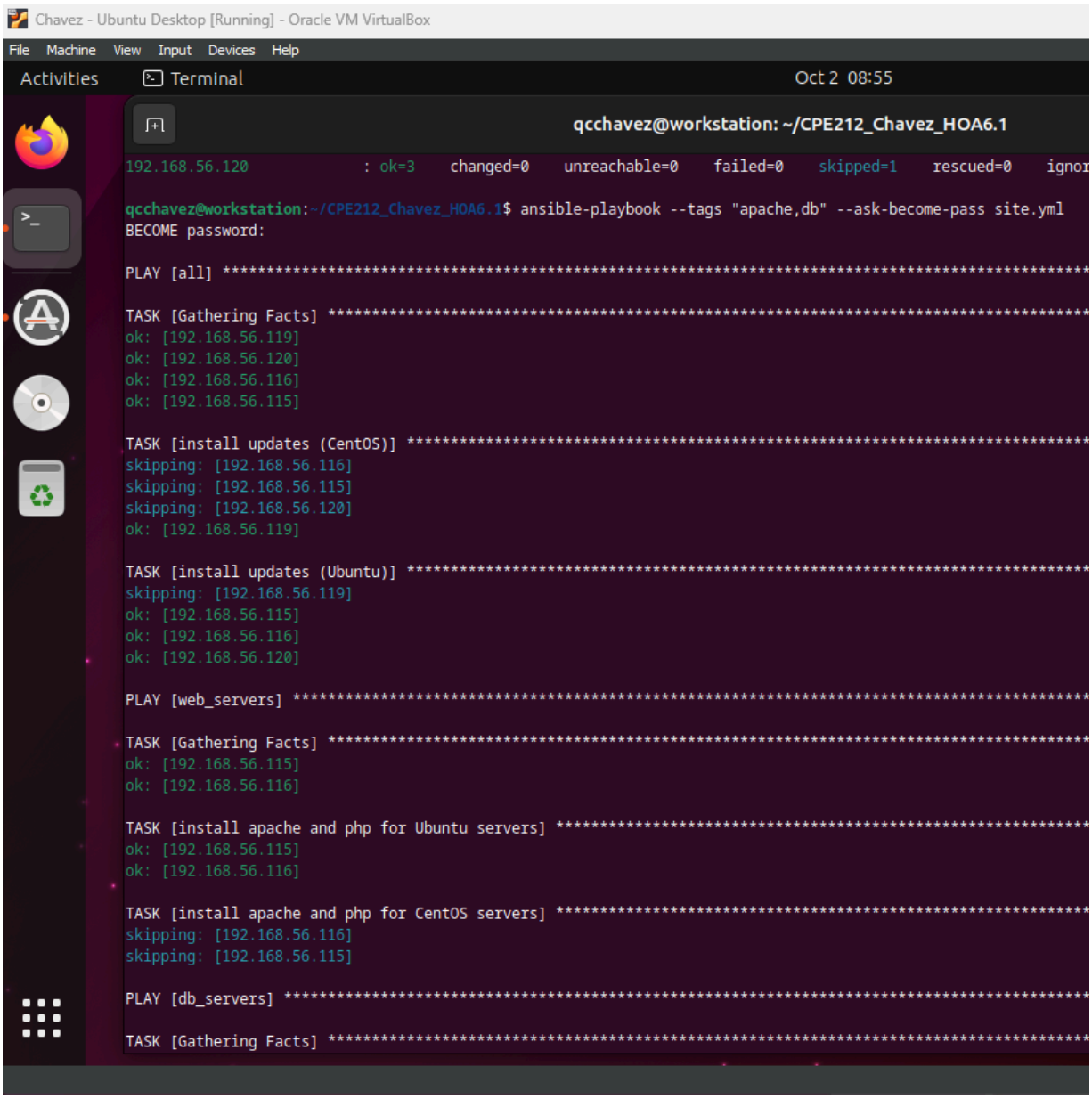
2.3 *ansible-playbook --tags db --ask-become-pass site.yml*

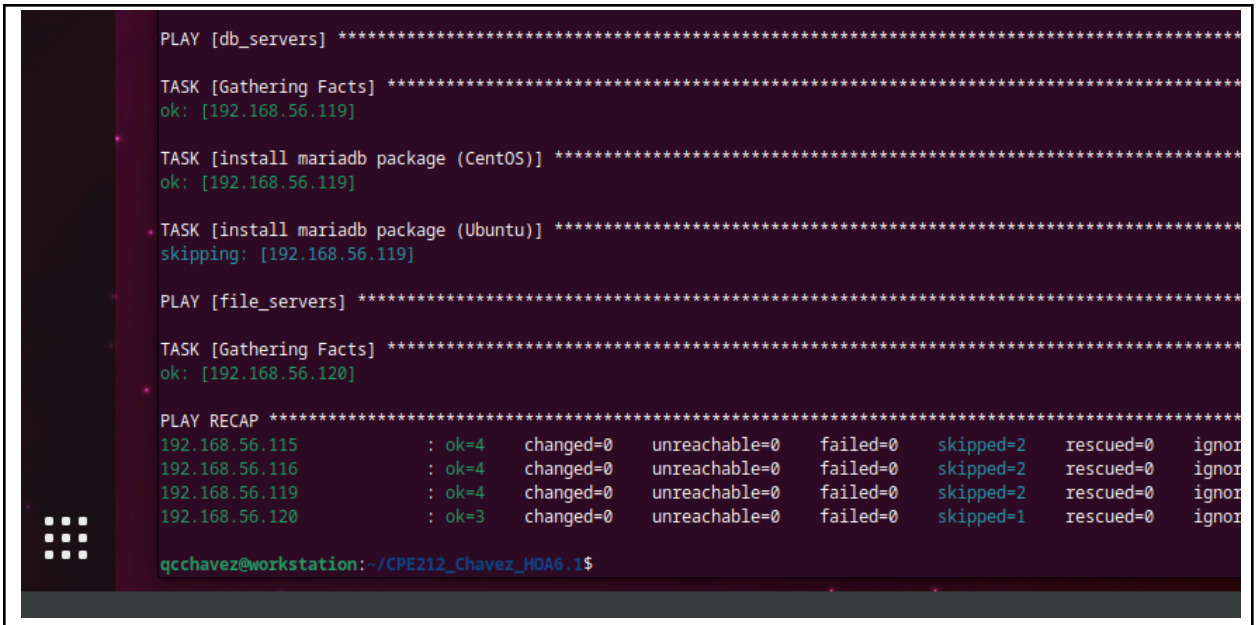


2.4 *ansible-playbook --tags apache --ask-become-pass site.yml*


Chavez - Ubuntu Desktop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Oct 2 08:54
qcchavez@workstation: ~/CPE212_Chavez_HOA6.1
192.168.56.119 : ok=4 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignor
192.168.56.120 : ok=3 changed=0 unreachable=0 failed=0 skipped=1 rescued=0 ignor
qcchavez@workstation:~/CPE212_Chavez_HOA6.1$ ansible-playbook --tags apache --ask-become-pass site.yml
BECOME password:
PLAY [all] ****
TASK [Gathering Facts] ****
ok: [192.168.56.115]
ok: [192.168.56.120]
ok: [192.168.56.119]
ok: [192.168.56.116]
TASK [install updates (CentOS)] ****
skipping: [192.168.56.116]
skipping: [192.168.56.115]
skipping: [192.168.56.120]
ok: [192.168.56.119]
TASK [install updates (Ubuntu)] ****
skipping: [192.168.56.119]
ok: [192.168.56.115]
ok: [192.168.56.116]
ok: [192.168.56.120]
PLAY [web_servers] ****
TASK [Gathering Facts] ****
ok: [192.168.56.115]
ok: [192.168.56.116]
TASK [install apache and php for Ubuntu servers] ****
ok: [192.168.56.115]
ok: [192.168.56.116]
TASK [install apache and php for CentOS servers] ****
skipping: [192.168.56.116]
skipping: [192.168.56.115]
PLAY [db_servers] ****
PLAY [db_servers] ****
TASK [Gathering Facts] ****
ok: [192.168.56.119]
PLAY [file_servers] ****
TASK [Gathering Facts] ****
ok: [192.168.56.120]
PLAY RECAP ****
192.168.56.115 : ok=4 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignor
192.168.56.116 : ok=4 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignor
192.168.56.119 : ok=3 changed=0 unreachable=0 failed=0 skipped=1 rescued=0 ignor
192.168.56.120 : ok=3 changed=0 unreachable=0 failed=0 skipped=1 rescued=0 ignor
qcchavez@workstation:~/CPE212_Chavez_HOA6.1$

2.5 *ansible-playbook --tags “apache,db” --ask-become-pass site.yml*



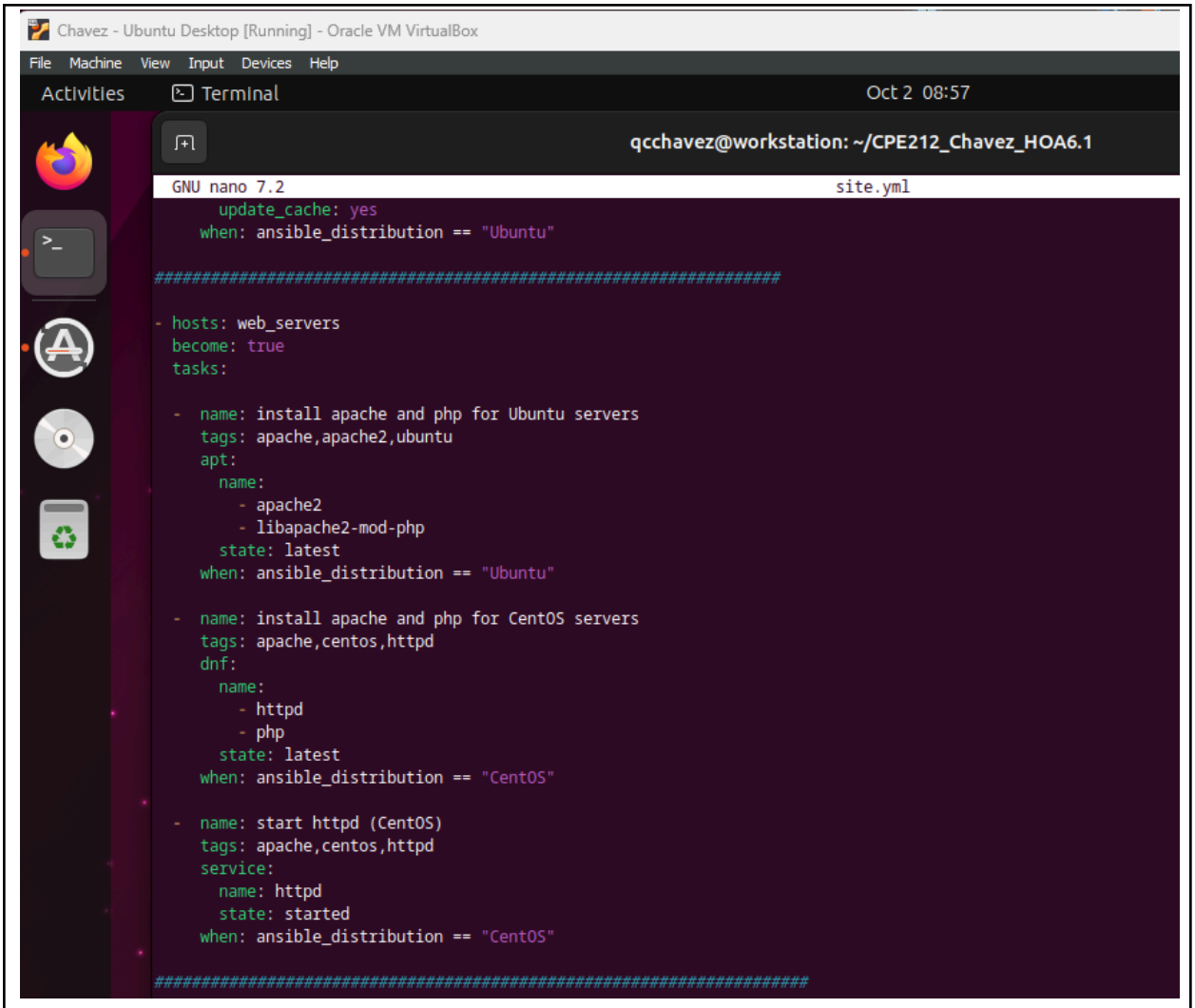


- In these screenshots, I've ran the **site.yml** playbook to comply with the task, the output shows that it only does the tasks where the **apache** tags exists.

Task 3: Managing Services

1. Edit the file *site.yml* and add a play that will automatically start the httpd on CentOS server.

```
- 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: started
  when: ansible_distribution == "CentOS"
```



- In these screenshot, I've added the necessary codes to start the **httpd** package for CentOS.

Figure 3.1.1
Make sure to save the file and exit.

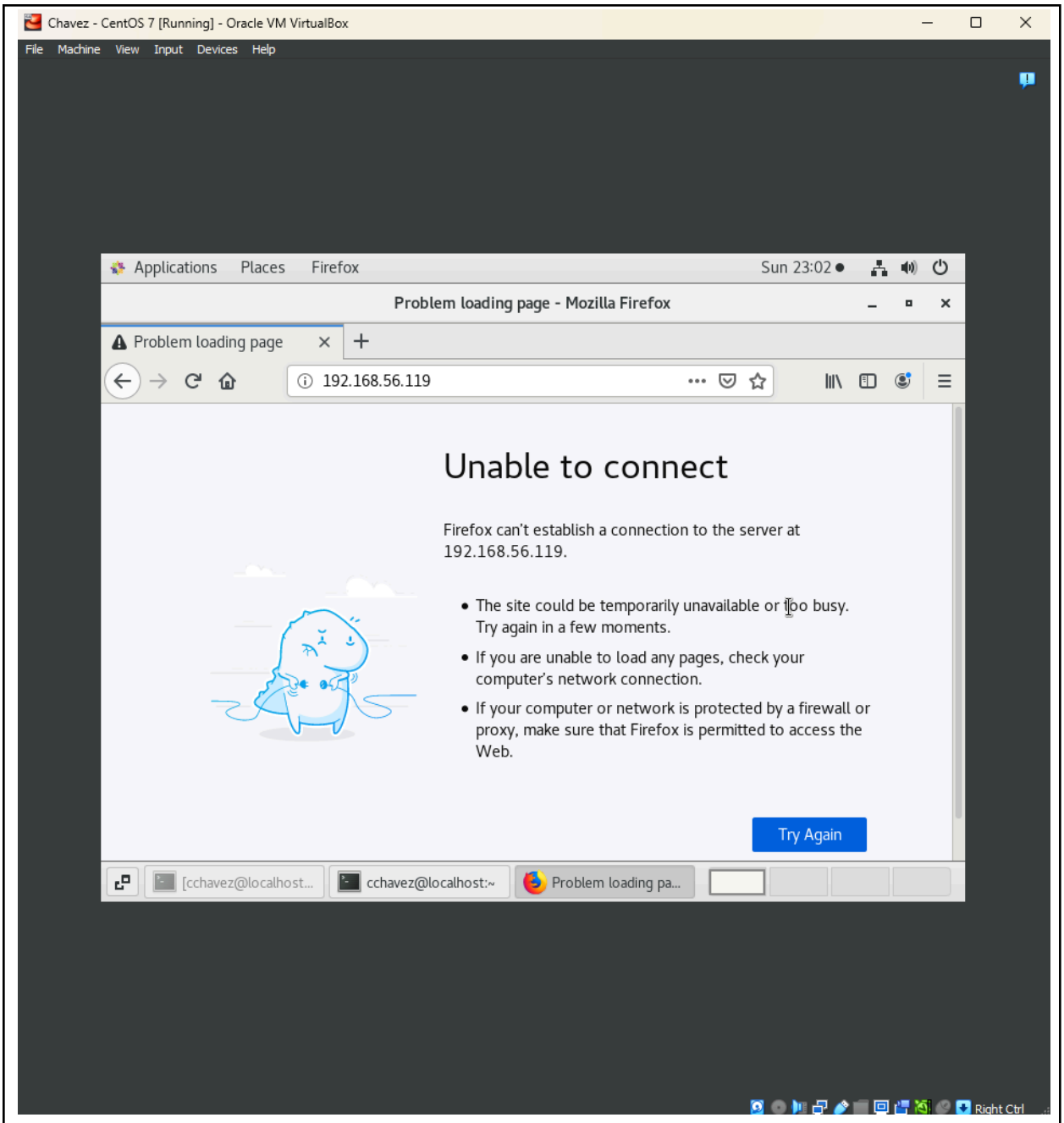
You would also notice from our previous activity that we already created a module that runs a service.

```
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    tags: centos, db,mariadb
    dnf:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

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

Figure 3.1.2

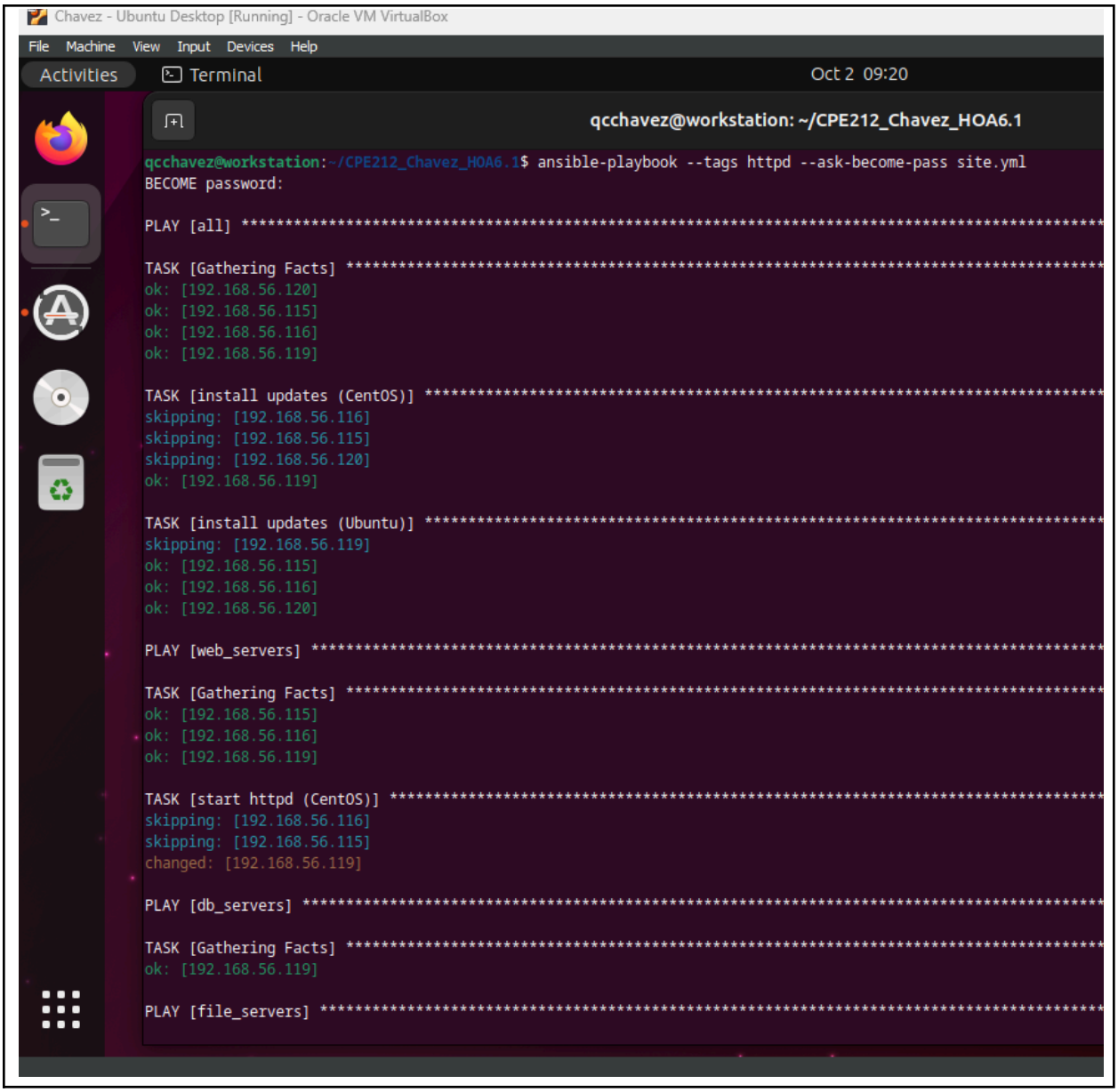
This is because in CentOS, installed packages' services are not run automatically. Thus, we need to create the module to run it automatically.

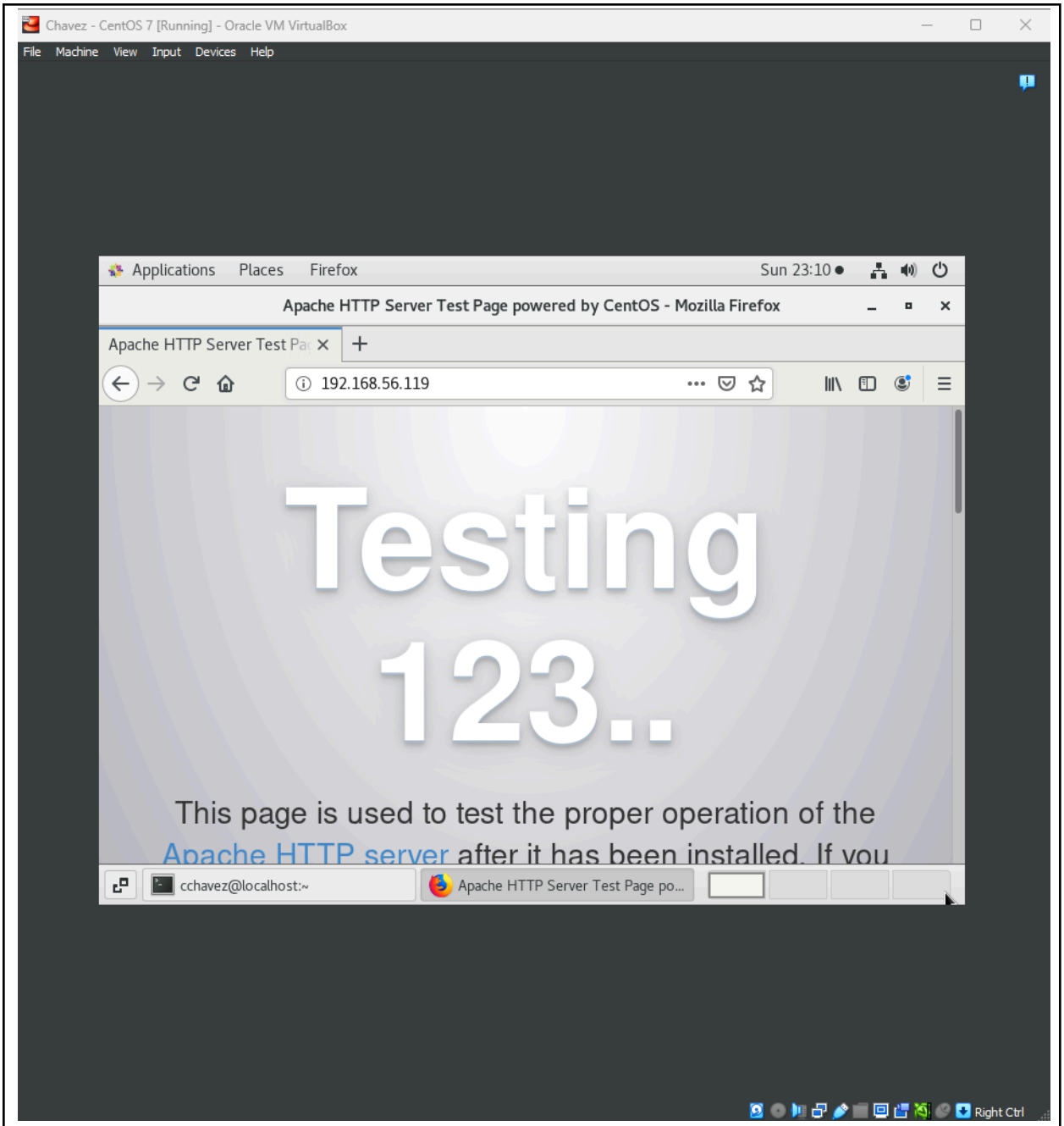
2. To test it, before you run the saved playbook, go to the CentOS server and stop the currently running httpd using the command *sudo systemctl stop httpd*. When prompted, enter the sudo password. After that, open the browser and enter the CentOS server's IP address. You should not be getting a display because we stopped the httpd service already.



- In this screenshot, I've double checked in the firefox if the **httpd** is active by entering the CentOS' IP address in the firefox browser.

3. Go to the local machine and this time, run the *site.yml* file. Then after running the file, go again to the CentOS server and enter its IP address on the browser. Describe the result.


Chavez - Ubuntu Desktop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Oct 2 09:20
qcchavez@workstation: ~/CPE212_Chavez_HOA6.1
qcchavez@workstation:~/CPE212_Chavez_HOA6.1$ ansible-playbook --tags httpd --ask-become-pass site.yml
BECOME password:
PLAY [all] ***
TASK [Gathering Facts] ***
ok: [192.168.56.120]
ok: [192.168.56.115]
ok: [192.168.56.116]
ok: [192.168.56.119]
TASK [install updates (CentOS)] ***
skipping: [192.168.56.116]
skipping: [192.168.56.115]
skipping: [192.168.56.120]
ok: [192.168.56.119]
TASK [install updates (Ubuntu)] ***
skipping: [192.168.56.119]
ok: [192.168.56.115]
ok: [192.168.56.116]
ok: [192.168.56.120]
PLAY [web_servers] ***
TASK [Gathering Facts] ***
ok: [192.168.56.115]
ok: [192.168.56.116]
ok: [192.168.56.119]
TASK [start httpd (CentOS)] ***
skipping: [192.168.56.116]
skipping: [192.168.56.115]
changed: [192.168.56.119]
PLAY [db_servers] ***
TASK [Gathering Facts] ***
ok: [192.168.56.119]
PLAY [file_servers] ***



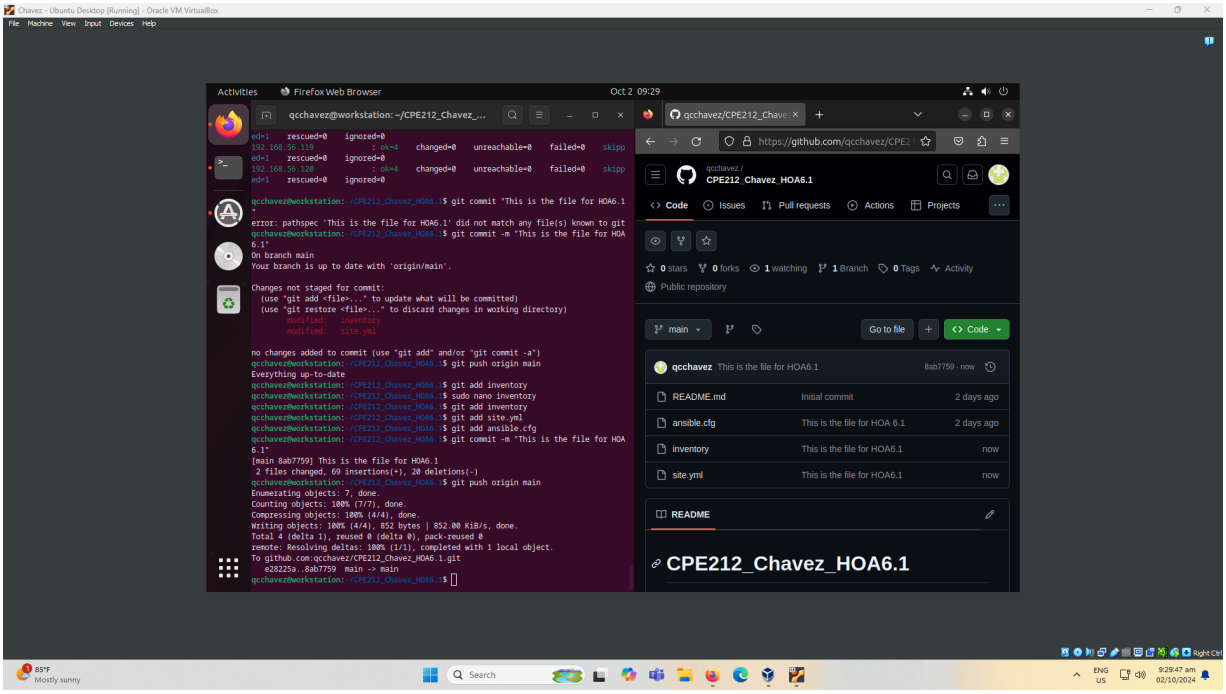
- In these screenshots, I've ran the **site.yml** playbook to comply with the task, the output shows that the **httpd** service for the CentOS remote server activated.

To automatically enable the service every time we run the playbook, use the command *enabled: true* similar to Figure 7.1.2 and save the playbook.

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

##############################################################

- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    tags: centos,db,mariadb
    dnf:
      name: mariadb-server
      state: latest

^G Help        ^O Write Out   ^W Where Is   ^K Cut         ^T Execute     ^C Location    M-U Undo
^X Exit        ^R Read File   ^\ Replace    ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo
```

- In this screenshot, I've adjusted the **service** section by putting a code **enabled: true** to make sure that the service always runs.

Committing repository to Github



- In this screenshot, I've committed the repository to the Github.

Reflections:

Answer the following:

1. What is the importance of putting our remote servers into groups?
    - Putting remote servers into groups is one of the important things when using playbooks as it separates the necessary remote servers based on the tasks that are needed to be applied.

2. What is the importance of tags in playbooks?
    - The importance of tags in playbooks is that you can choose whatever task that can be run so that you don't have to wait for a long time when running the whole playbook, especially when it consists of multiple lines of codes.

3. Why do you think some services need to be managed automatically in playbooks?
    - Services are needed to be managed automatically in playbooks because some services are required for everyday uses. And this practice is good for maintaining the efficiency, reliability, and being also manageable of the remote servers.