Name: Clarence B. Chavez	**Date Performed:** September 23, 2024
Course/Section: CPE212-CPE31S2	**Date Submitted:** September 23, 2024
Instructor: Engr. Robin Valenzuela	**Semester and SY:** 1st Sem, 2024-2025

Activity 5: Consolidating Playbook plays

1. **Objectives:**

1.1 Use **when** command in playbook for different OS distributions

1.2 Apply refactoring techniques in cleaning up the playbook codes

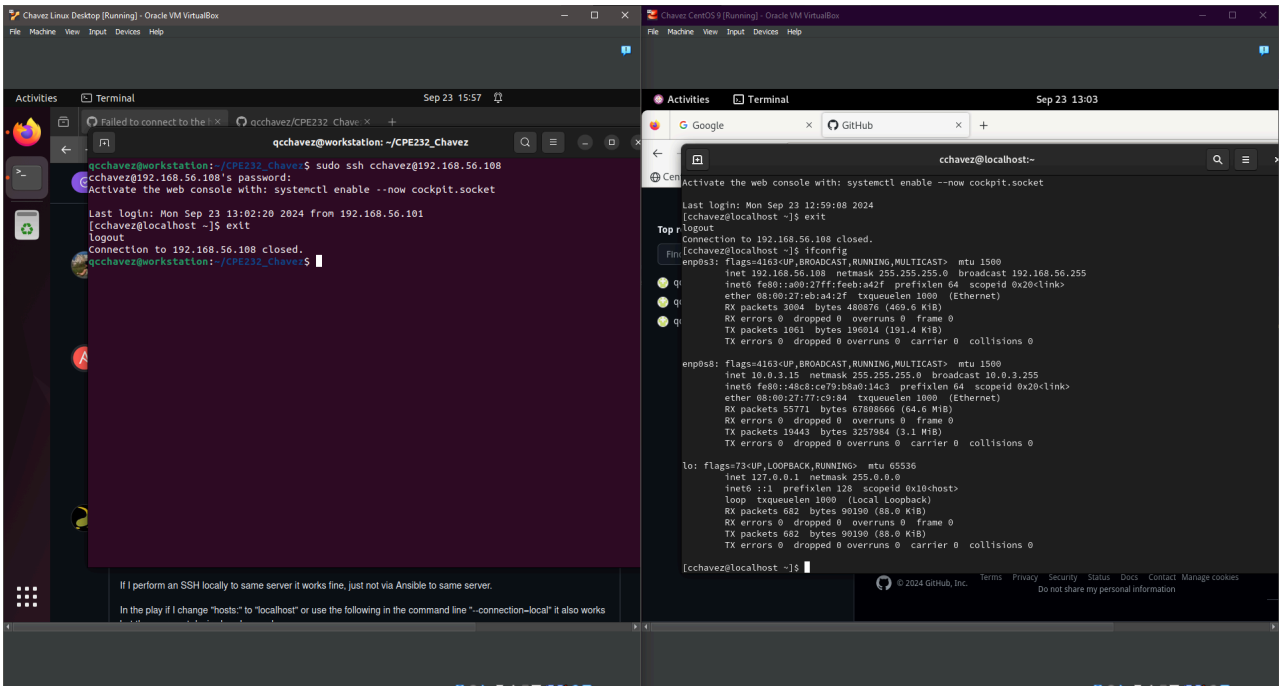
2. **Discussion**:

We are going to look at a way that we can differentiate a playbook by a host in terms of which distribution the host is running. It's very common in most Linux shops to run multiple distributions, for example, Ubuntu shop or Debian shop and you need a different distribution for a one off-case or perhaps you want to run plays only on certain distributions.

It is a best practice in ansible when you are working in a collaborative environment to use the command git pull. git pull is a Git command used to update the local version of a repository from a remote. By default, git pull does two things. Updates the current local working branch (currently checked out branch) and updates the remote-tracking branches for all other branches. git pull essentially pulls down any changes that may have happened since the last time you worked on the repository.

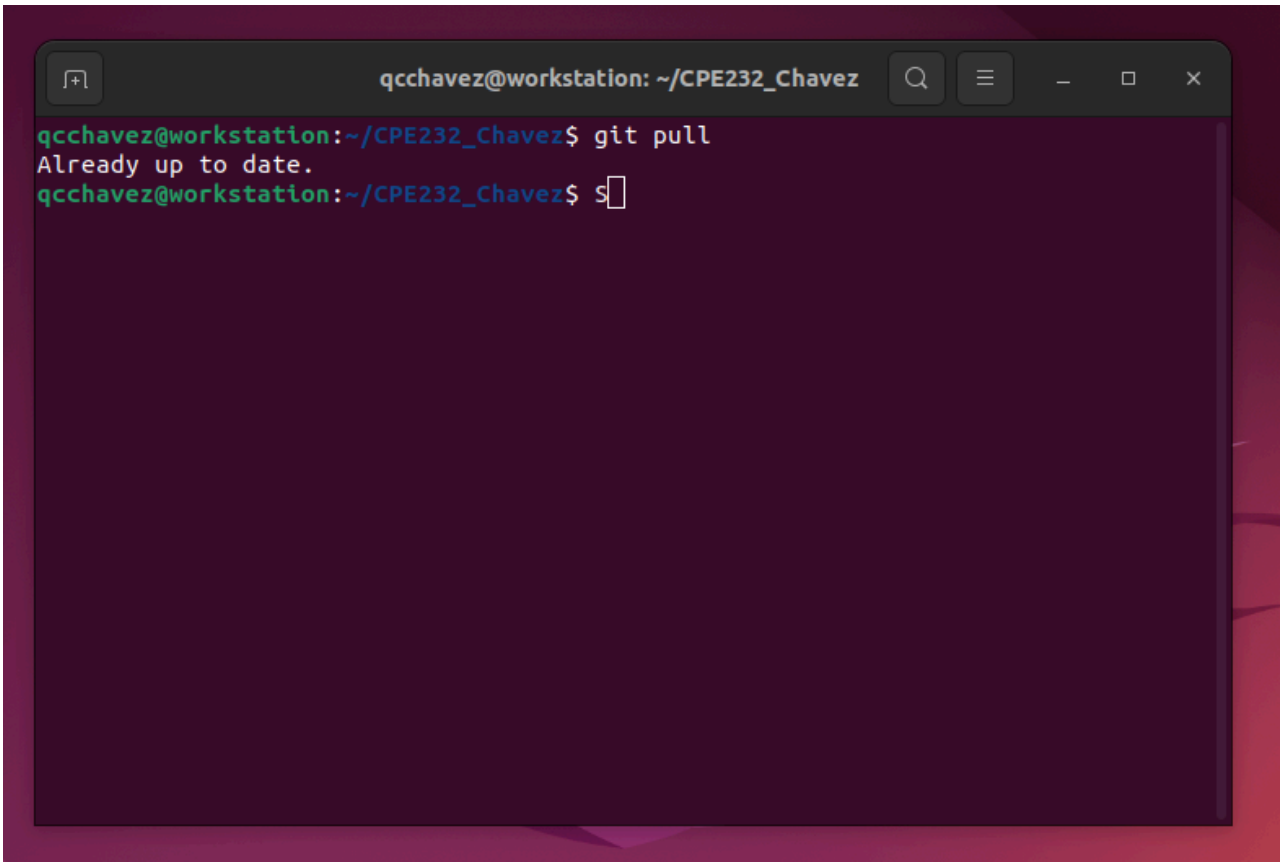
Requirement:

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



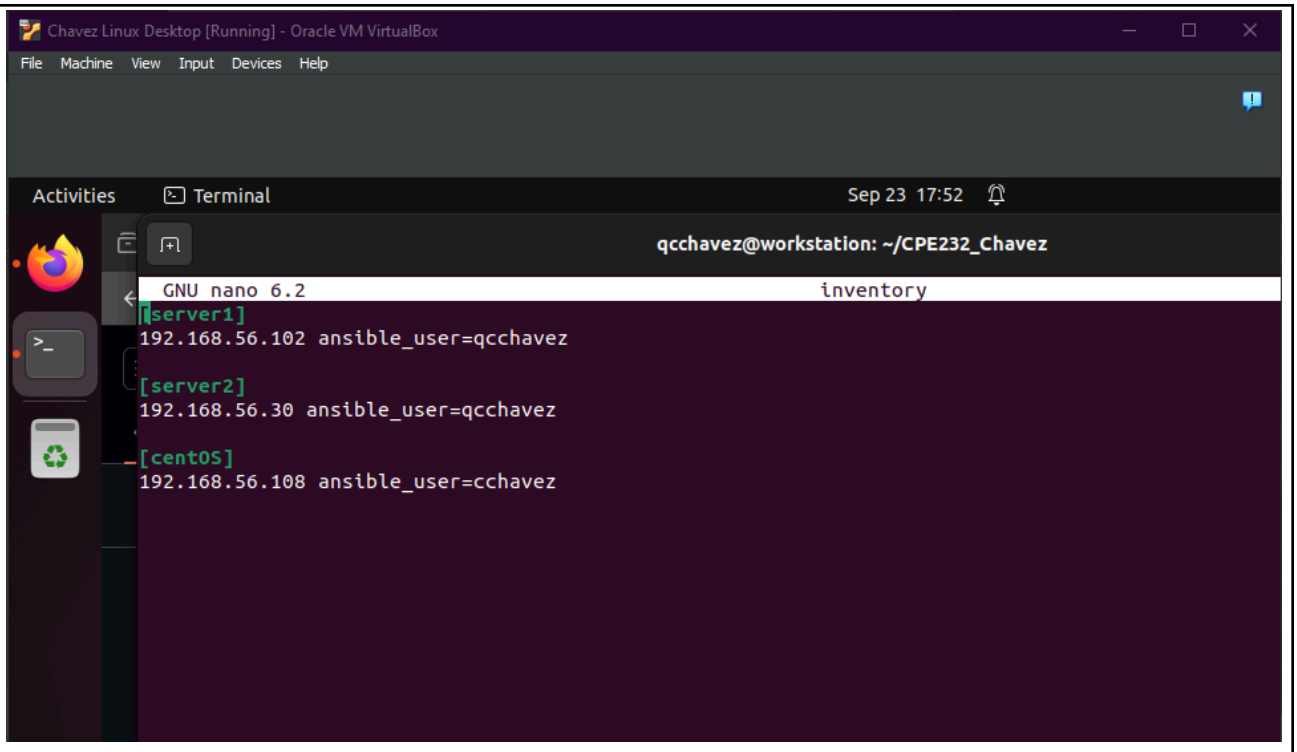
Task 1: Use when command for different distributions

1. In the local machine, make sure you are in the local repository directory (*CPE232_yourname*). Issue the command git pull. When prompted, enter the correct passphrase or password. Describe what happened when you issue this command. Did something happen? Why?

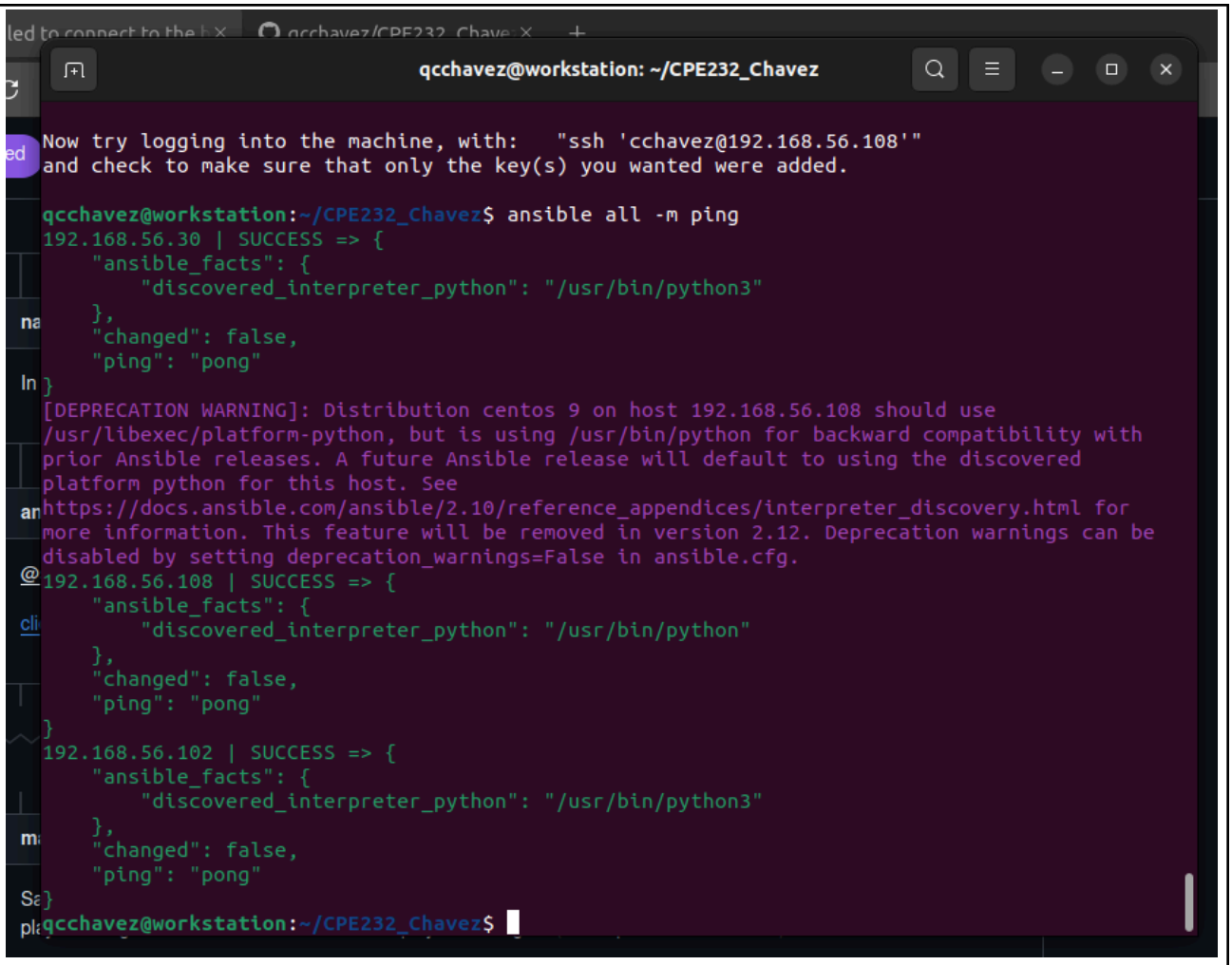


- Based on the screenshot, after changing my directory to **CPE232_Chavez** and prompting the command **git pull**, I received an “already up to date” message from the terminal. This means that the repo is already up to date.

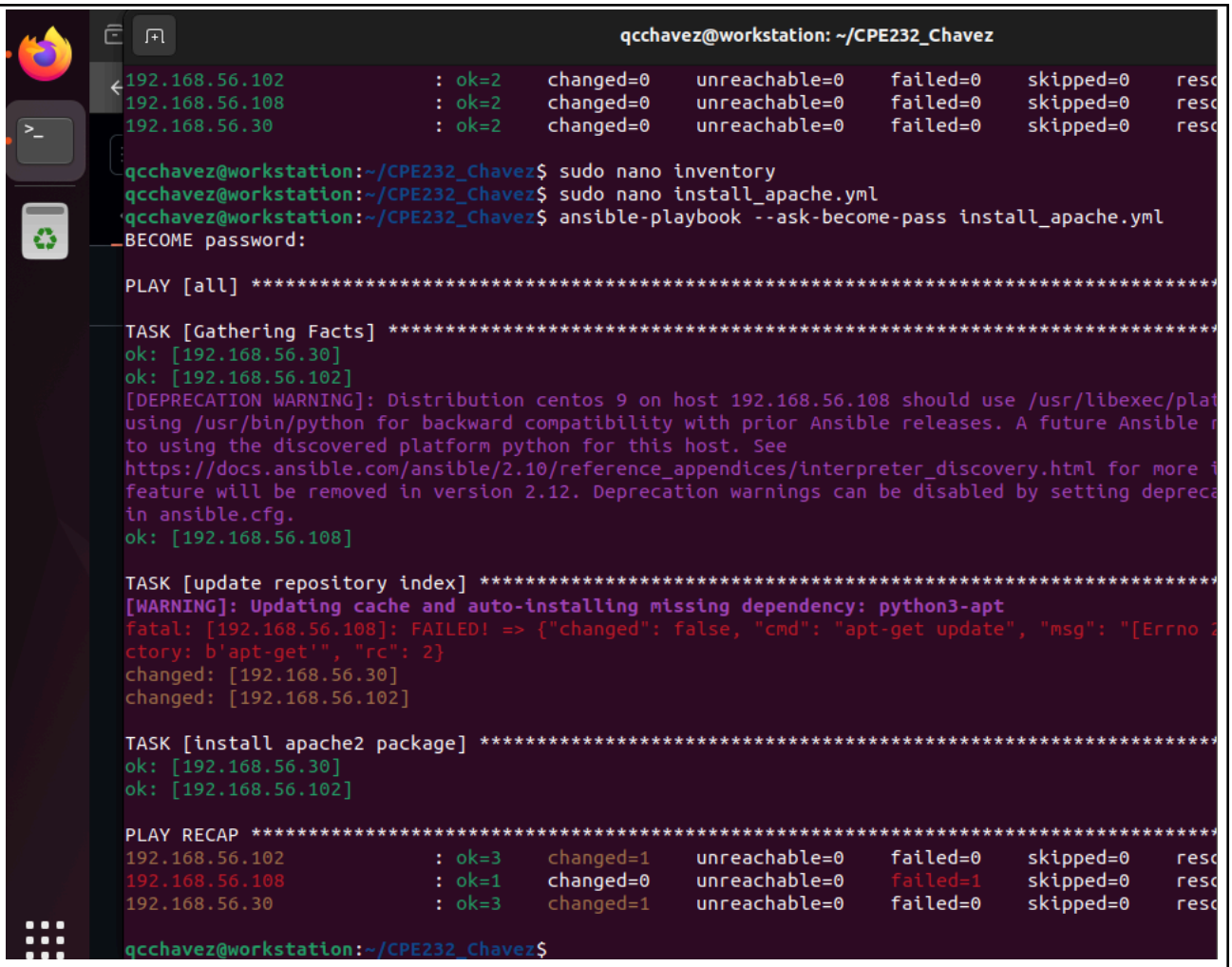
2. Edit the inventory file and add the IP address of the Centos VM. Issue the command we used to execute the playbook (the one we used in the last activity): *ansible-playbook --ask-become-pass install_apache.yml*. After executing this command, you may notice that it did not become successful in the Centos VM. You can see that the Centos VM has failed=1. Only the two remote servers have been changed. The reason is that Centos VM does not support "apt" as the package manager. The default package manager for Centos is "yum."



- In this screenshot, I've added the IP address of my CentOS alongside of my server1 and server2 IP addresses in the inventory file.



- In this screenshot, I've prompted a ping to make sure that the three IP addresses that are inside of the inventory file, are pinging successfully.



- In this screenshot, I've prompted *ansible-playbook --ask-become-pass install_apache.yml* command to check what output will show if I will prompt the command, and the **"failed=1"** was found in the IP address from the 192.168.56.108, which is the IP address of the CentOS. The remaining remote servers had an **changed** and **ok** status.

3. Edit the *install_apache.yml* file and insert the lines shown below.

```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

  - name: install apache2 package
    apt:
      name: apache2
    when: ansible_distribution == "Ubuntu"

  - name: add PHP support for apache
    apt:
      name: libapache2-mod-php
    when: ansible_distribution == "Ubuntu"
```

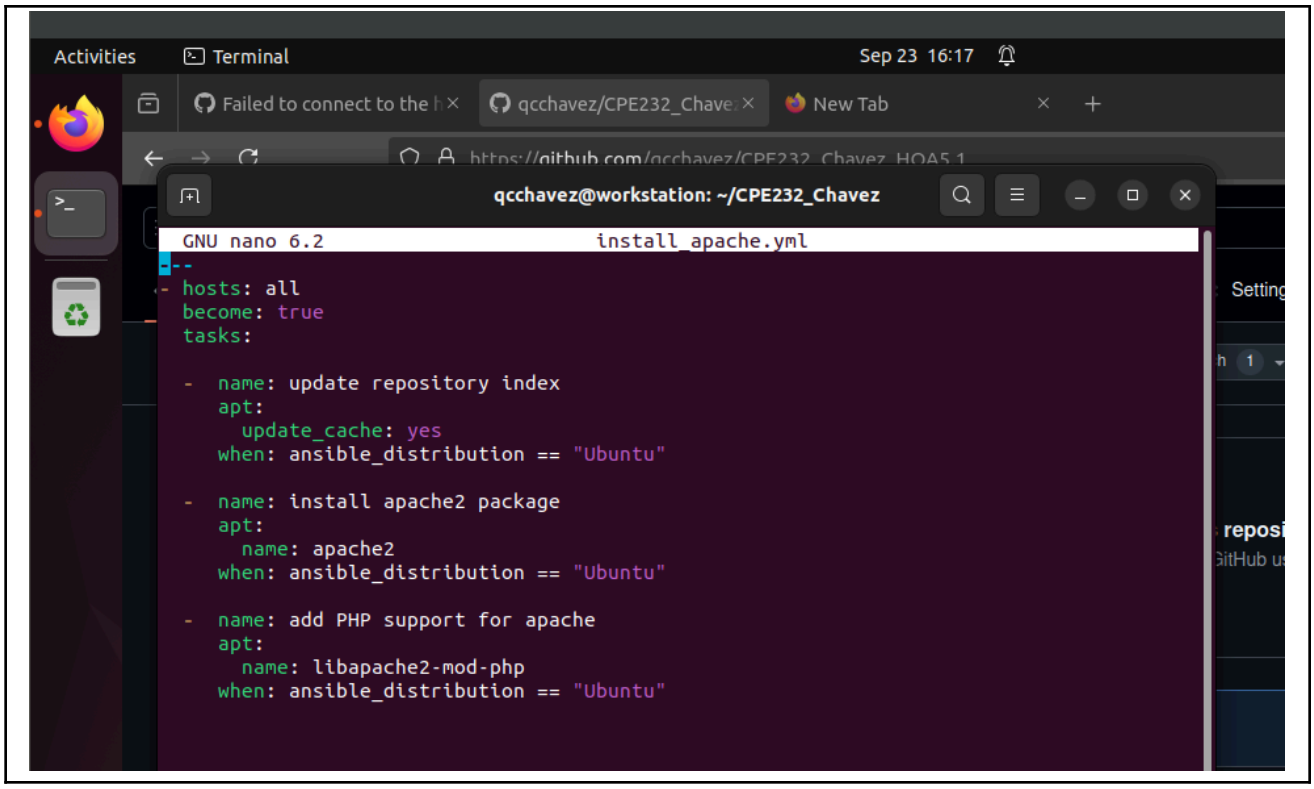
Make sure to save the file and exit.

Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.

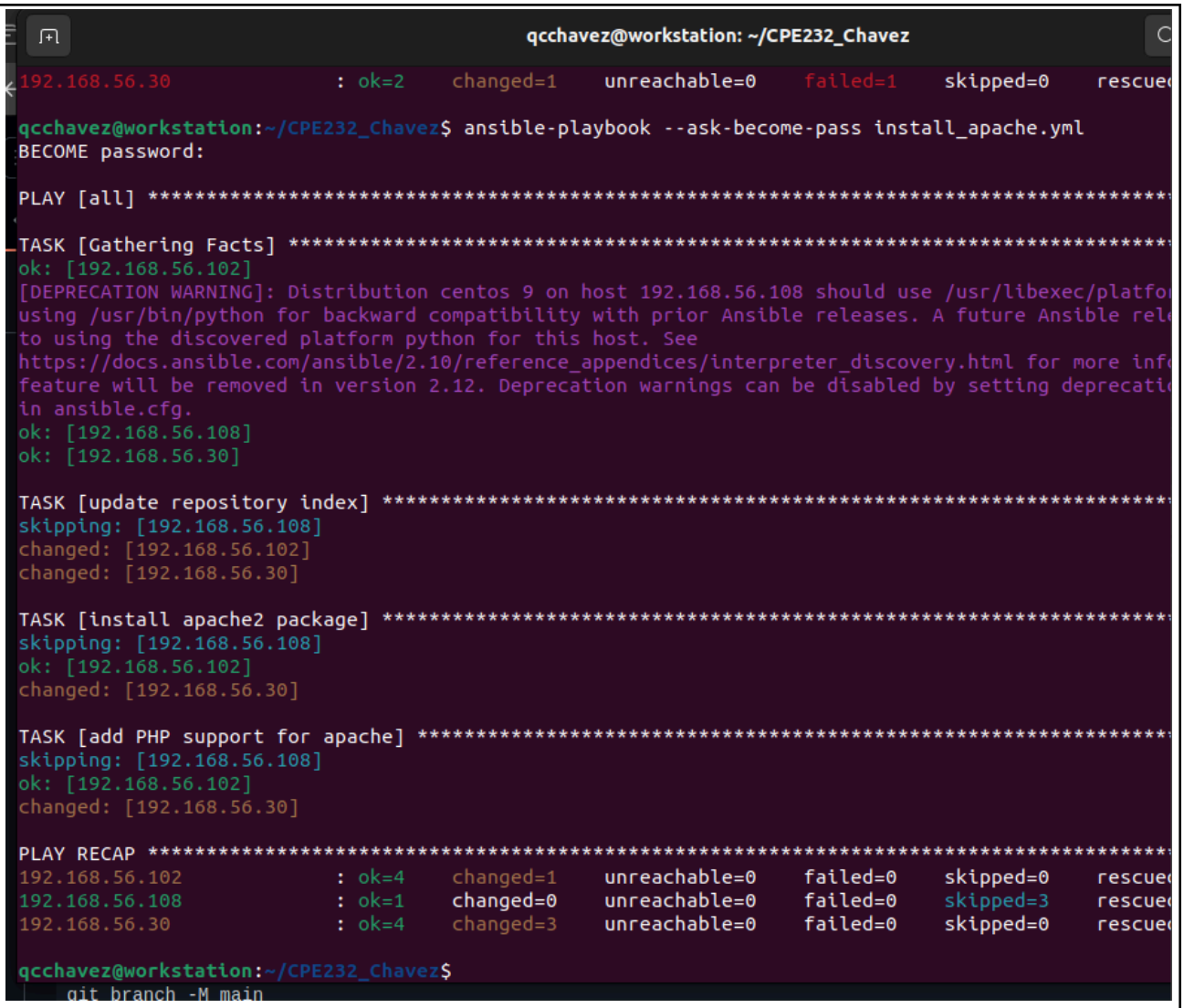
If you have a mix of Debian and Ubuntu servers, you can change the configuration of your playbook like this.

```
- name: update repository index
  apt:
    update_cache: yes
  when: ansible_distribution in [“Debian”, “Ubuntu]
```

Note: This will work also if you try. Notice the changes are highlighted.


Activities
Terminal
Sep 23 16:17
Failed to connect to the h
qcchavez/CPE232_Chave
New Tab
qcchavez@workstation: ~/CPE232_Chavez
GNU nano 6.2
install_apache.yml

- hosts: all
become: true
tasks:
- name: update repository index
apt:
update_cache: yes
when: ansible_distribution == "Ubuntu"
- name: install apache2 package
apt:
name: apache2
when: ansible_distribution == "Ubuntu"
- name: add PHP support for apache
apt:
name: libapache2-mod-php
when: ansible_distribution == "Ubuntu"



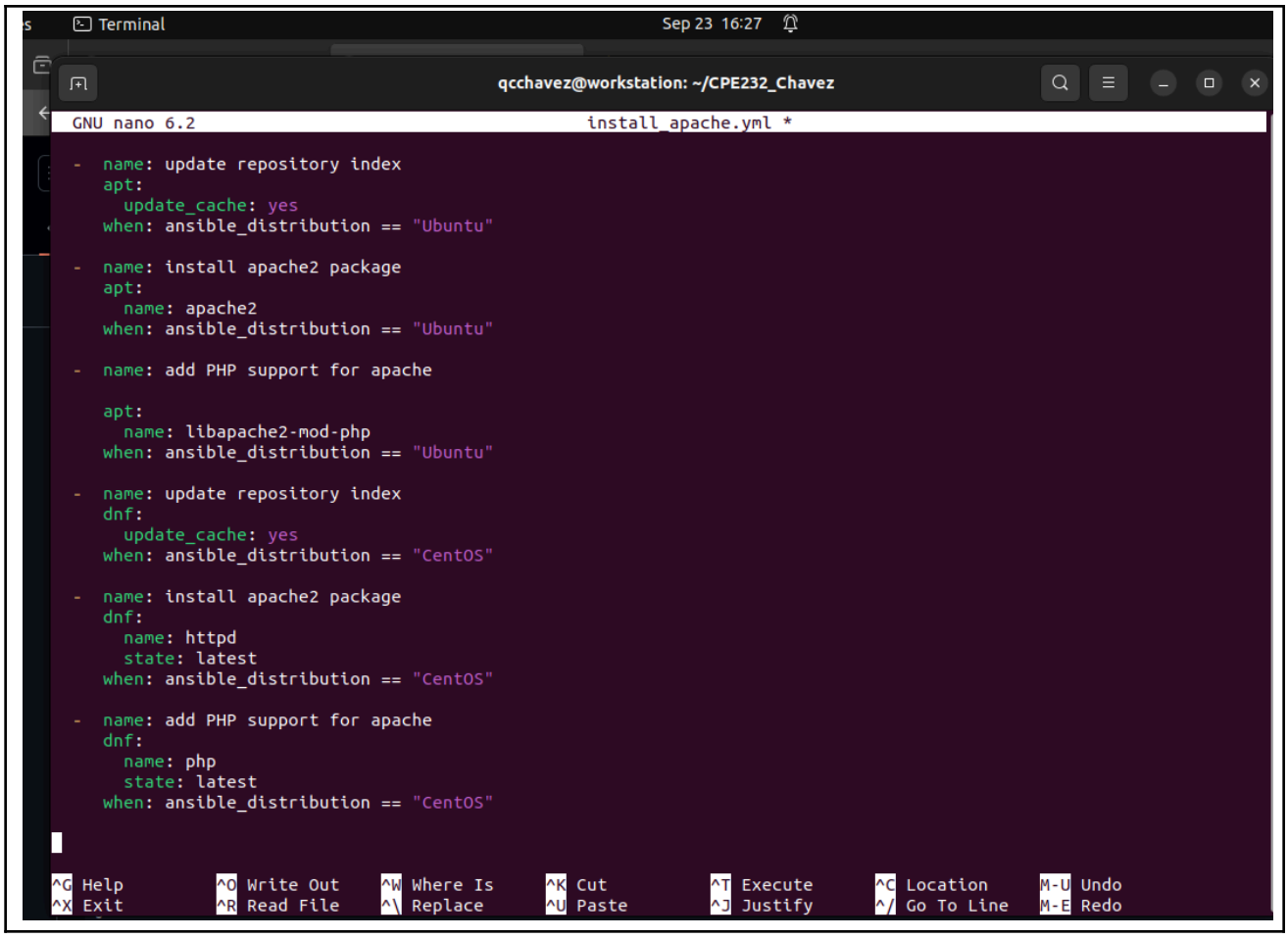
- In these screenshots, I've modified the **install_apache.yml** file and followed the instructions in this step. I've added the **update repository index, install apache2 package, and the adding of PHP support for apache** for Ubuntu only. After modifying the .yml file, I've prompted the command for ansible-playbook, and the tasks were done. The server2's tasks are all in changed state. The tasks for the CentOS were skipped, and the tasks for the server1 were changed and applied.

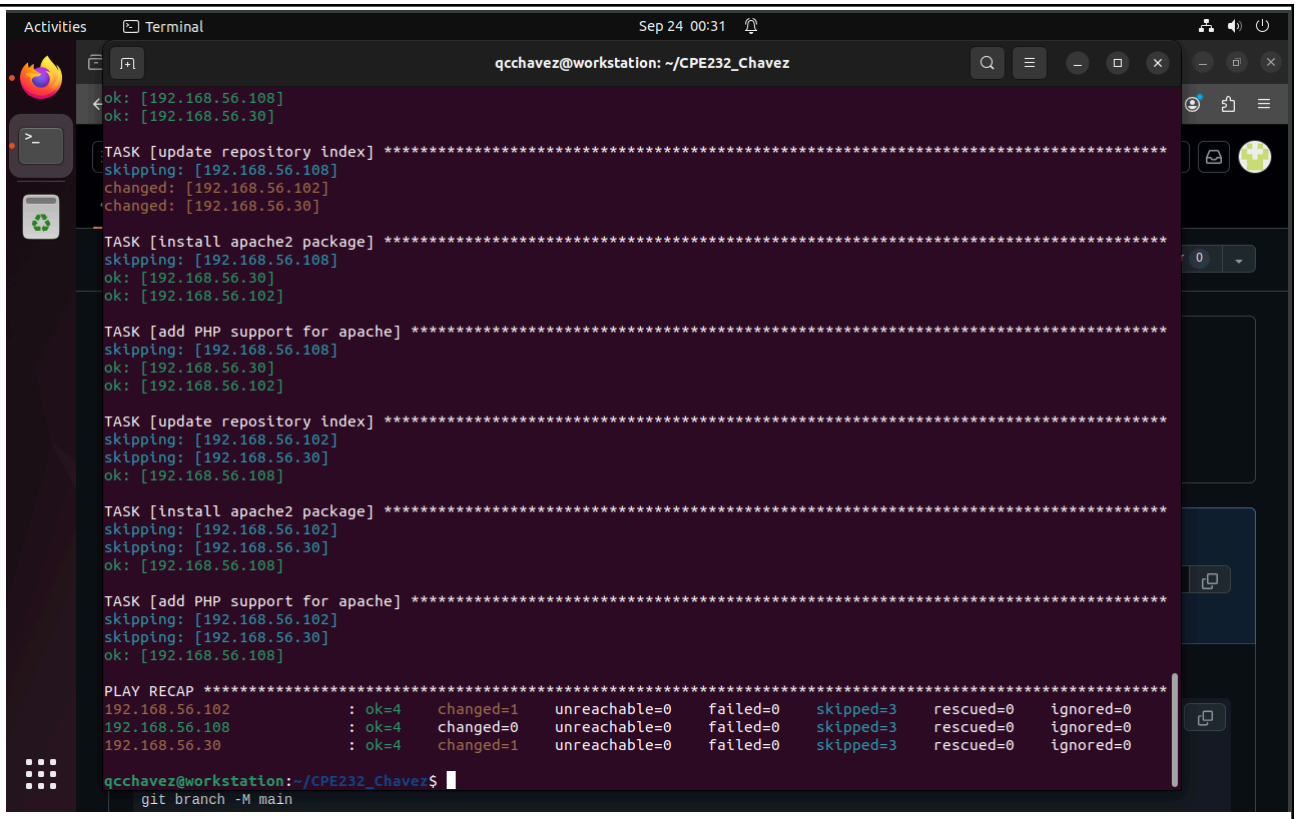
4. Edit the *install_apache.yml* file and insert the lines shown below.

```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

  - name: install apache2 package
    apt:
      name: apache2
      stae: latest
    when: ansible_distribution == "Ubuntu"

  - name: add PHP support for apache
    apt:
      name: libapache2-mod-php
      state: latest
    when: ansible_distribution == "Ubuntu"

  - name: update repository index
    dnf:
      update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: install apache2 package
    dnf:
      name: httpd
      state: latest
    when: ansible_distribution == "CentOS"

  - name: add PHP support for apache
    dnf:
      name: php
      state: latest
    when: ansible_distribution == "CentOS"
```

Make sure to save and exit.

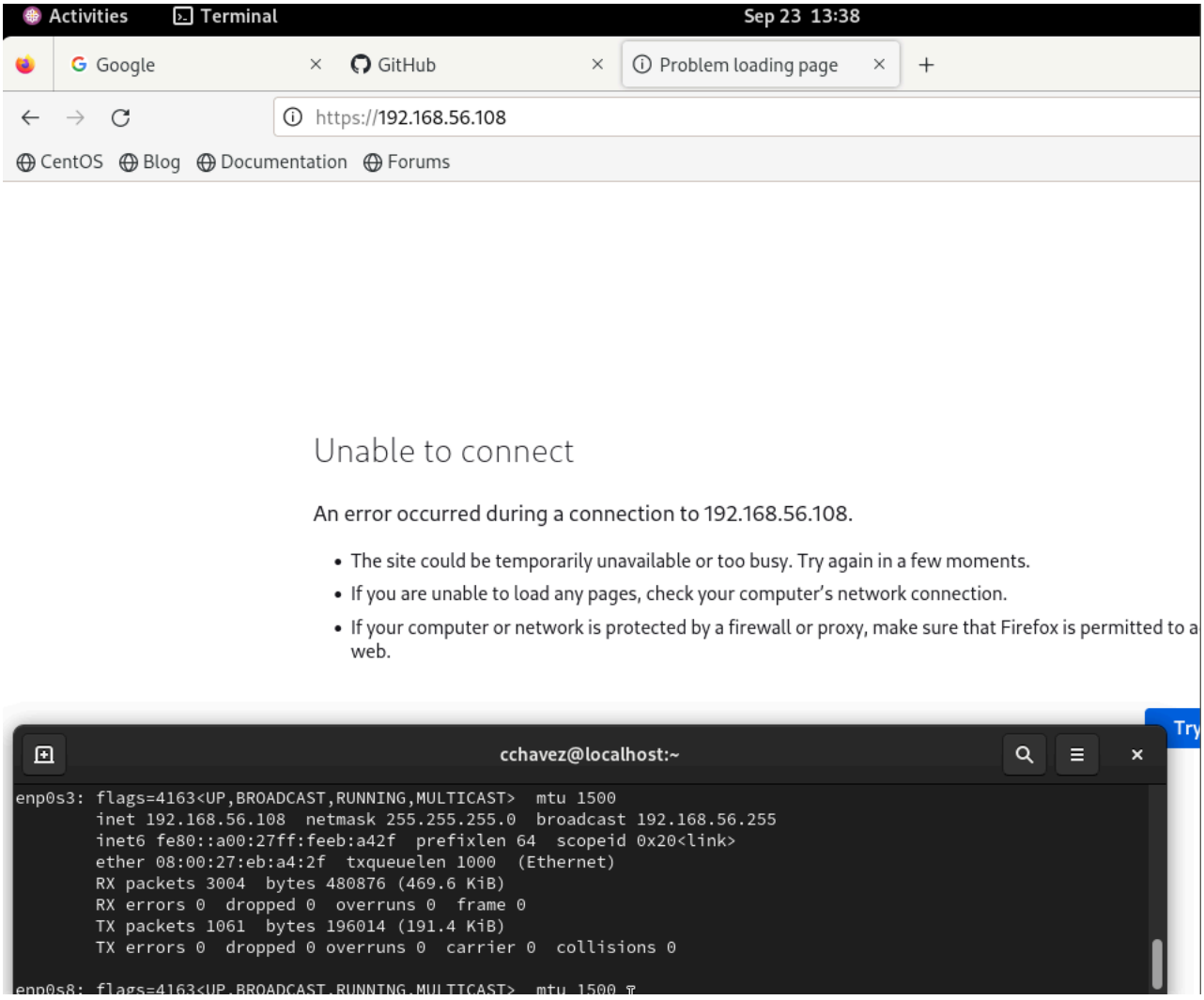
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.


Terminal
Sep 23 16:27
qcchavez@workstation: ~/CPE232_Chavez
GNU nano 6.2
install_apache.yml *
- name: update repository index
apt:
update_cache: yes
when: ansible_distribution == "Ubuntu"
- name: install apache2 package
apt:
name: apache2
when: ansible_distribution == "Ubuntu"
- name: add PHP support for apache
apt:
name: libapache2-mod-php
when: ansible_distribution == "Ubuntu"
- name: update repository index
dnf:
update_cache: yes
when: ansible_distribution == "CentOS"
- name: install apache2 package
dnf:
name: httpd
state: latest
when: ansible_distribution == "CentOS"
- name: add PHP support for apache
dnf:
name: 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



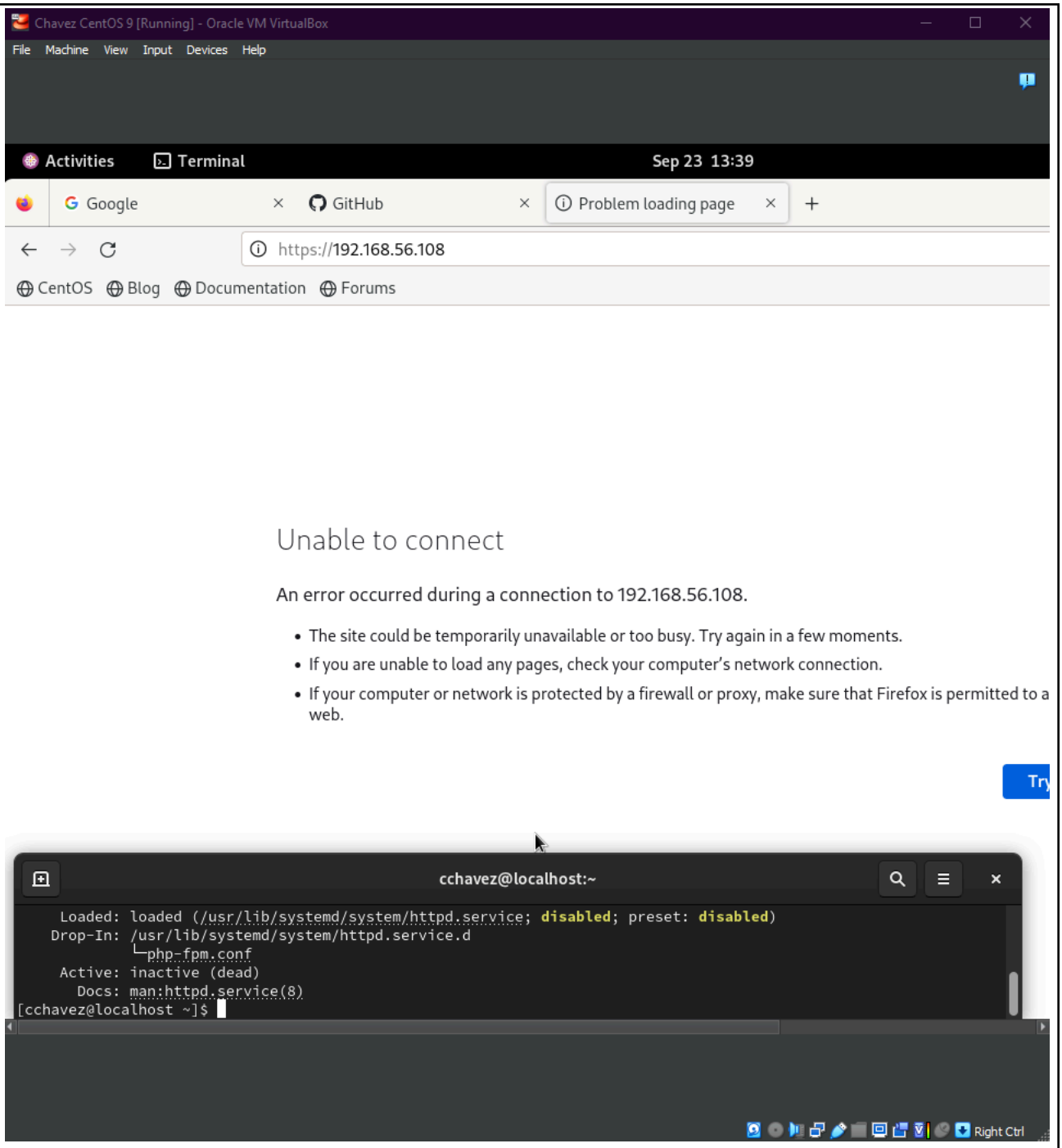
- In these screenshots, I've modified the **install_apache.yml** file again and followed the instructions in this step. I've added the **update repository index, install apache2 package, and added PHP support for apache** for Ubuntu and CentOS. After modifying the .yml file, I've prompted the command for ansible-playbook, and the tasks were done. Some of CentOS'tasks were either skipped, changed or made. This is also the same with Server1. For Server2, update repository index was in a **changed** state, and the rest were good.

5. To verify the installations, go to CentOS VM and type its IP address on the browser. Was it successful? The answer is no. It's because the httpd service or the Apache HTTP server in the CentOS is not yet active. Thus, you need to activate it first.



- In this screenshot, I've used a browser to test if the httpd was installed, I found out that it was installed, but it wasn't activated just yet.

5.1 To activate, go to the CentOS VM terminal and enter the following:
systemctl status httpd
The result of this command tells you that the service is inactive.



- In this screenshot, I've checked the status of the httpd of CentOS, and found out that it was indeed disabled.
-

5.2 Issue the following command to start the service:

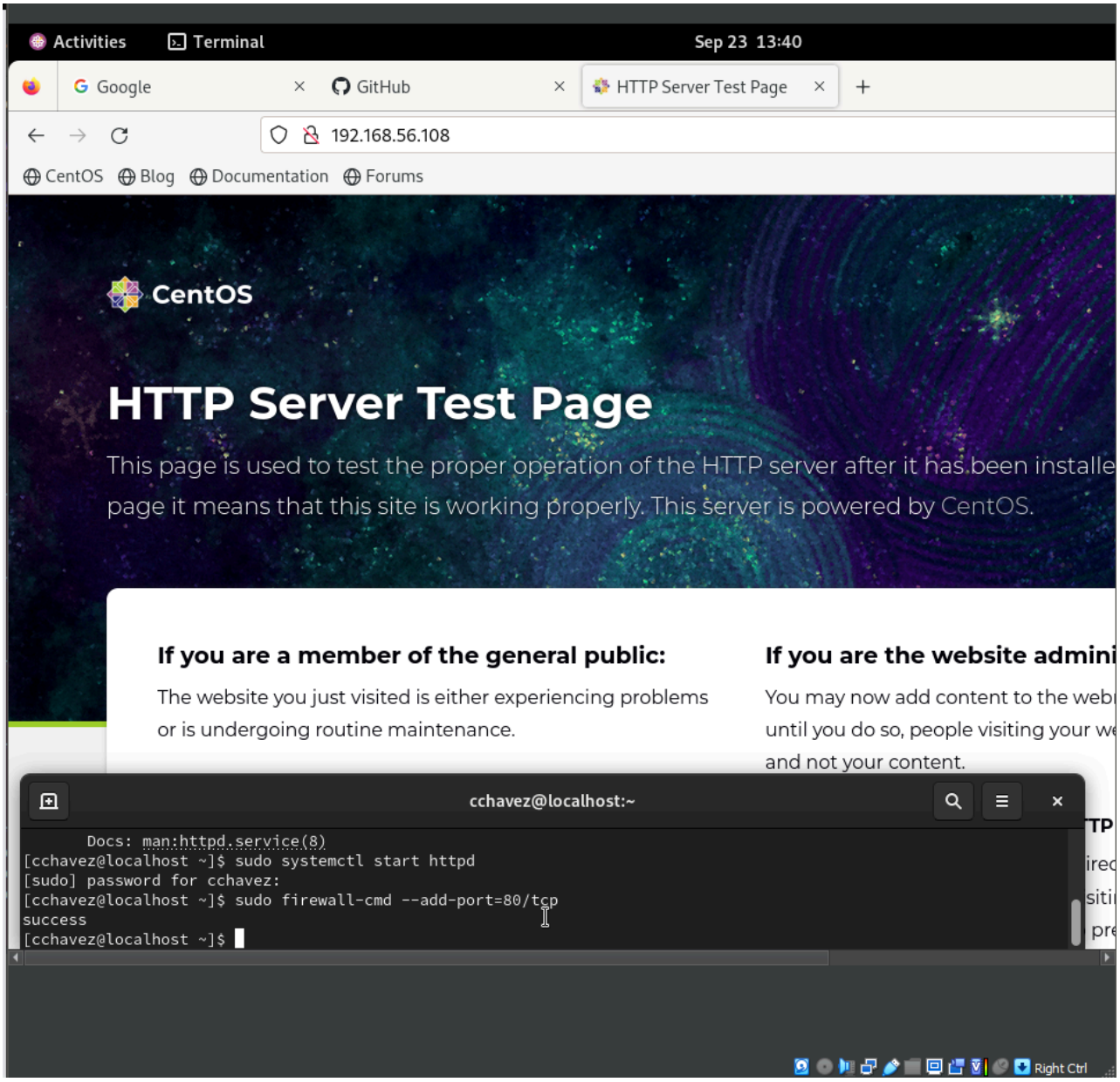
sudo systemctl start httpd

(When prompted, enter the sudo password)

sudo firewall-cmd --add-port=80/tcp

(The result should be a success)

5.3 To verify the service is already running, go to CentOS VM and type its IP address on the browser. Was it successful? (Screenshot the browser)



- In this screenshot, I've activated the httpd of CentOS with the use of the terminal. And I've also checked again in the browser if the httpd now works by entering the CentOS's own IP address on the search bar of the browser.

Task 2: Refactoring playbook

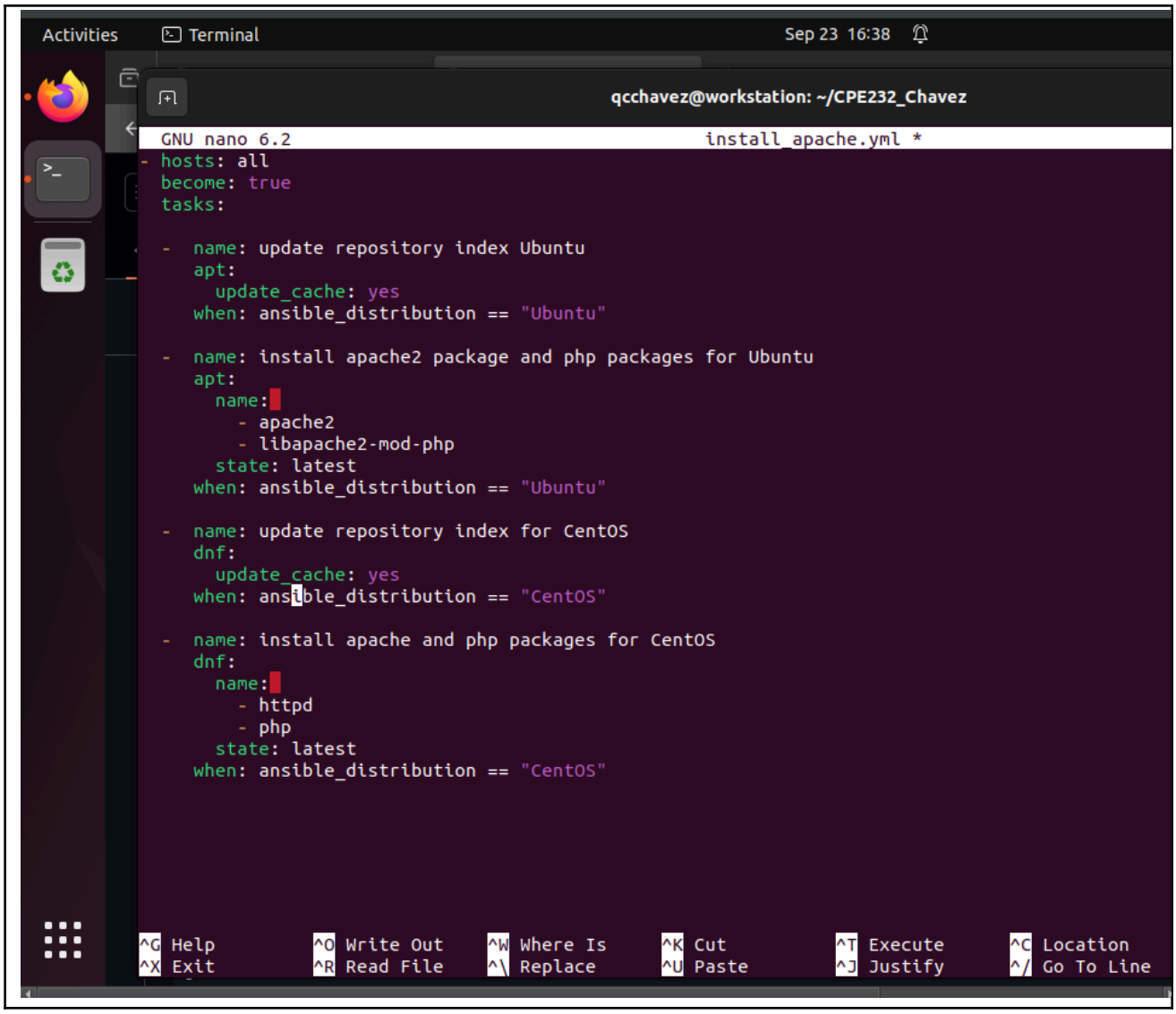
This time, we want to make sure that our playbook is efficient and that the codes are easier to read. This will also makes run ansible more quickly if it has to execute fewer tasks to do the same thing.

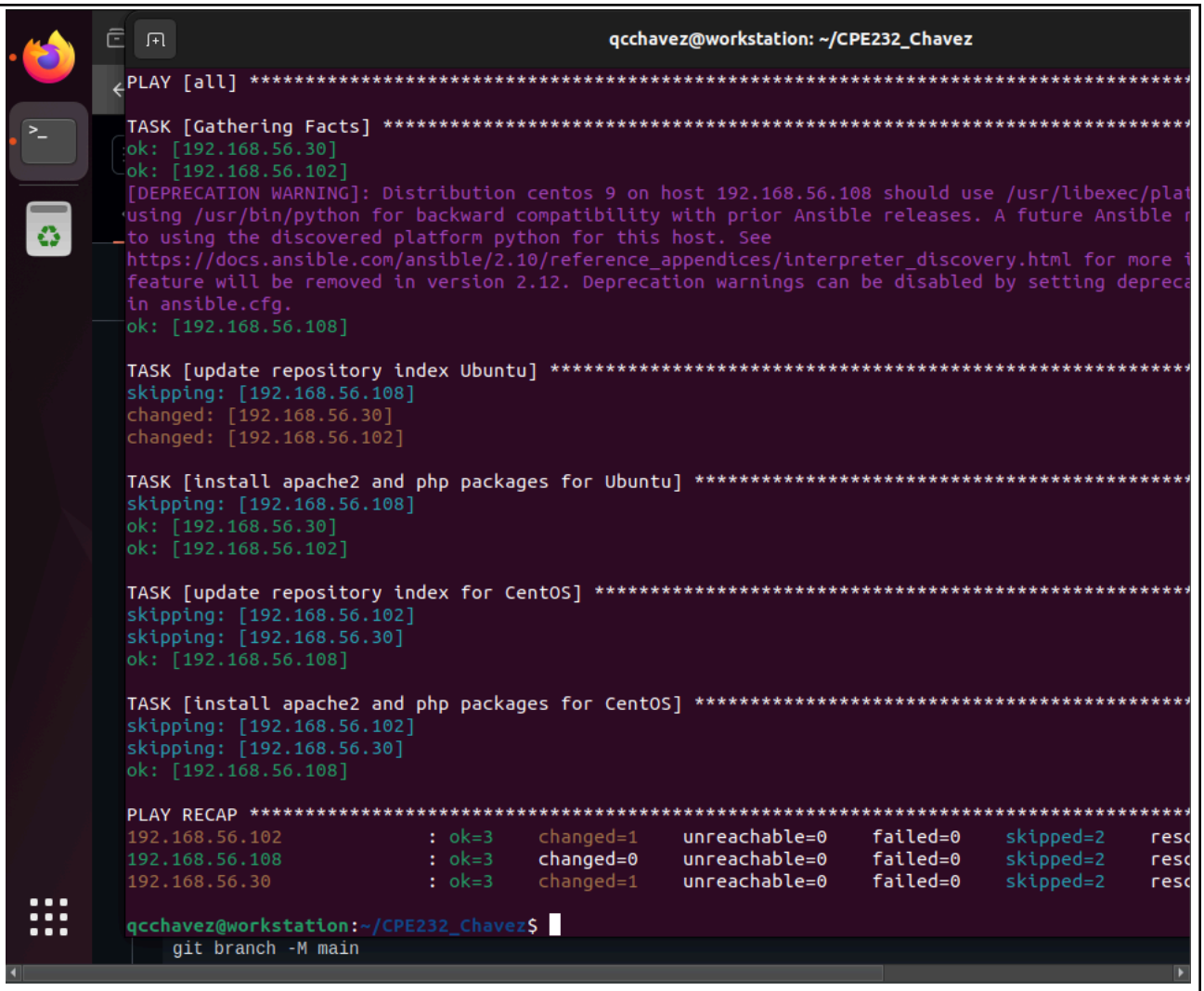
1. Edit the playbook *install_apache.yml*. Currently, we have three tasks targeting our Ubuntu machines and 3 tasks targeting our CentOS machine. Right now, we try to consolidate some tasks that are typically the same. For example, we can consolidate two plays that install packages. We can do that by creating a list of installation packages as shown below:

```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index Ubuntu
    apt:
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

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

  - name: update repository index for CentOS
    dnf:
      update_cache: yes
    when: ansible_distribution == "CentOS"

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

   Make sure to save the file and exit.

   Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.


Activities
Terminal
Sep 23 16:38
qcchavez@workstation: ~/CPE232_Chavez
GNU nano 6.2
install_apache.yml *
- hosts: all
become: true
tasks:
- name: update repository index Ubuntu
apt:
update_cache: yes
when: ansible_distribution == "Ubuntu"
- name: install apache2 package and php packages for Ubuntu
apt:
name:
- apache2
- libapache2-mod-php
state: latest
when: ansible_distribution == "Ubuntu"
- name: update repository index for CentOS
dnf:
update_cache: yes
when: ansible_distribution == "CentOS"
- name: install apache and php packages for CentOS
dnf:
name:
- httpd
- php
state: latest
when: ansible_distribution == "CentOS"
^G Help
^X Exit
^O Write Out
^R Read File
^W Where Is
^\ Replace
^K Cut
^U Paste
^T Execute
^J Justify
^C Location
^/ Go To Line



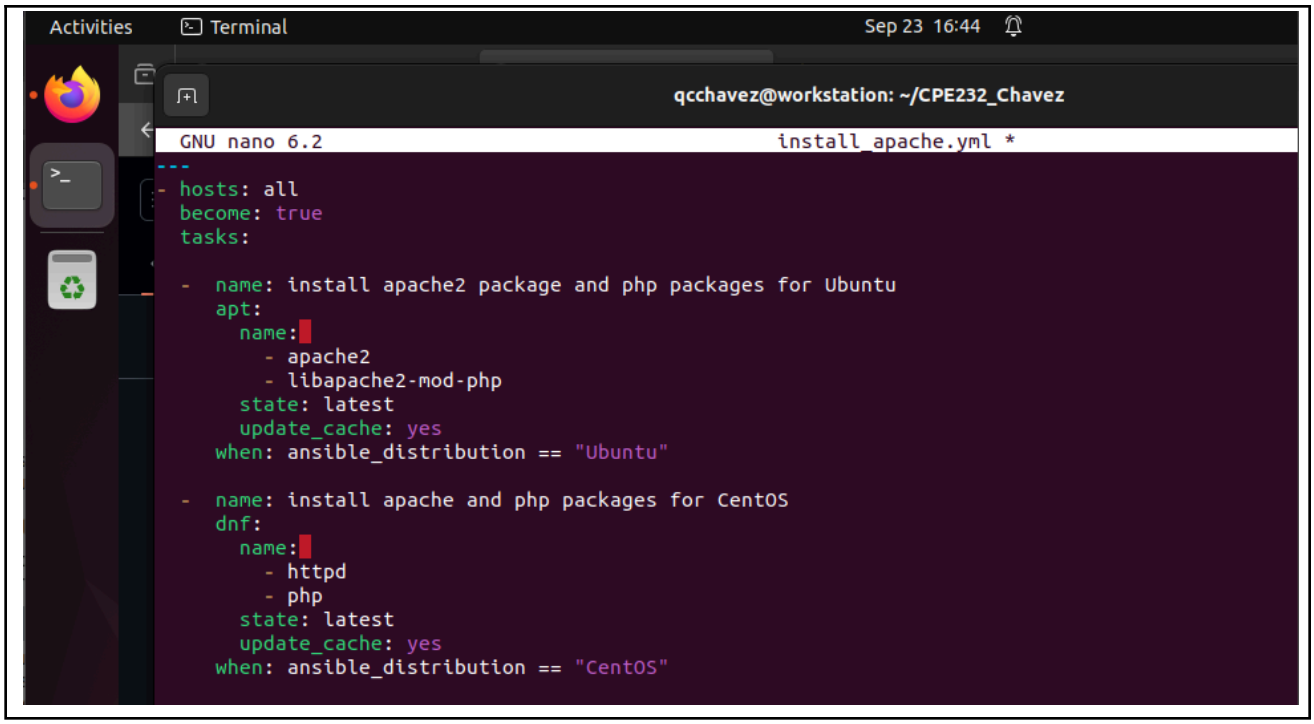
- In these screenshots, I've modified the **install_apache.yml** file again, but this time, the content of the file has less lines compared to earlier. The package names are now in one place, rather than being separately. After modifying, I've prompted the command for activating the playbook. Server1 skipped 2 tasks, and changed 1. The CentOS server skipped 2 tasks, and Server2, just like the Server1, changed 1 task, and also skipped 2 tasks.

2. Edit the playbook *install_apache.yml* again. In task 2.1, we consolidated the plays into one play. This time we can actually consolidated everything in just 2 plays. This can be done by removing the update repository play and putting the command *update_cache: yes* below the command *state: latest*. See below for reference:

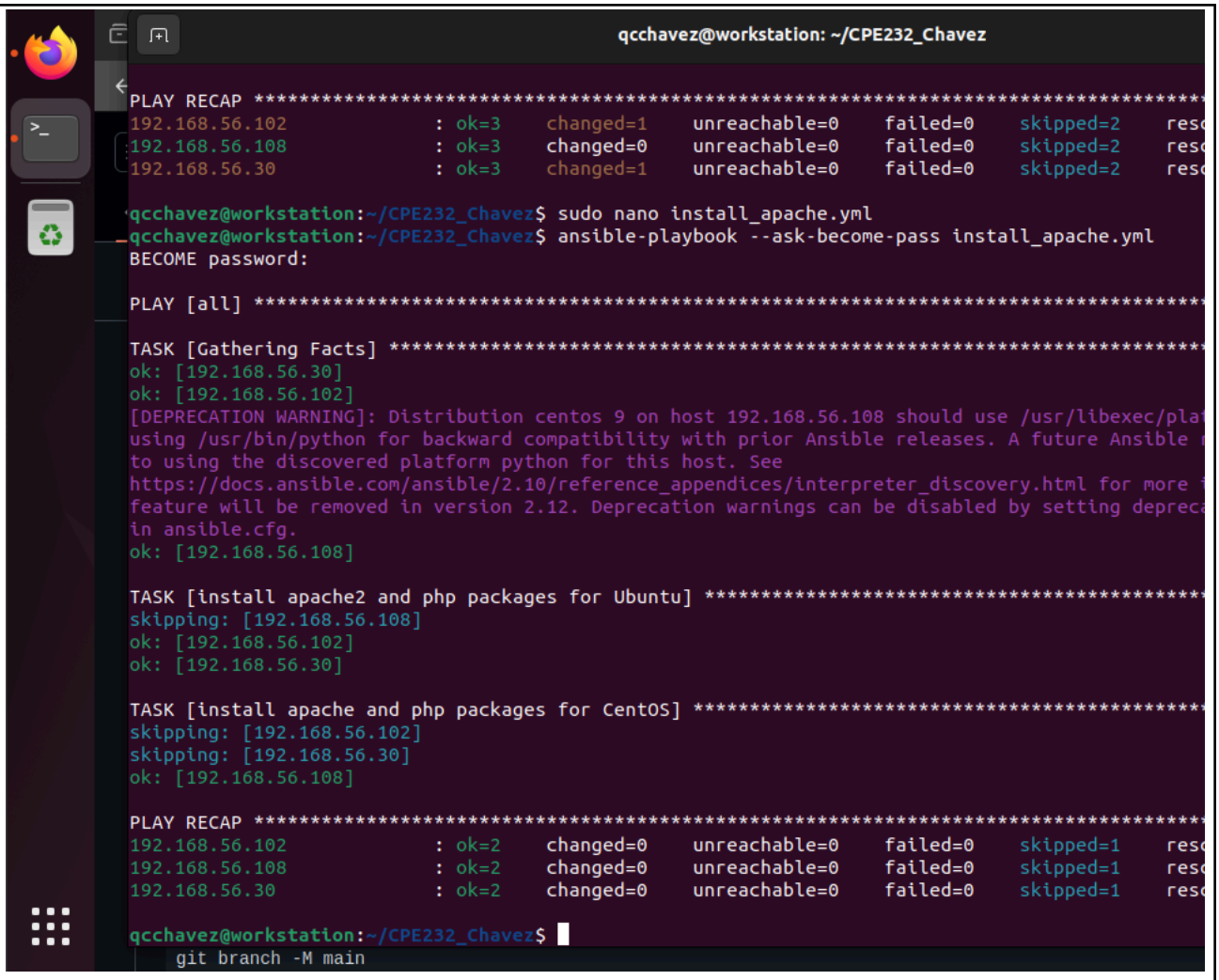
```
---
- hosts: all
  become: true
  tasks:

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

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

Make sure to save the file and exit.

Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.


Activities
Terminal
Sep 23 16:44
qcchavez@workstation: ~/CPE232_Chavez
GNU nano 6.2
install_apache.yml *

- hosts: all
become: true
tasks:
- name: install apache2 package and php packages for Ubuntu
apt:
name:
- apache2
- libapache2-mod-php
state: latest
update_cache: yes
when: ansible_distribution == "Ubuntu"
- name: install apache and php packages for CentOS
dnf:
name:
- httpd
- php
state: latest
update_cache: yes
when: ansible_distribution == "CentOS"

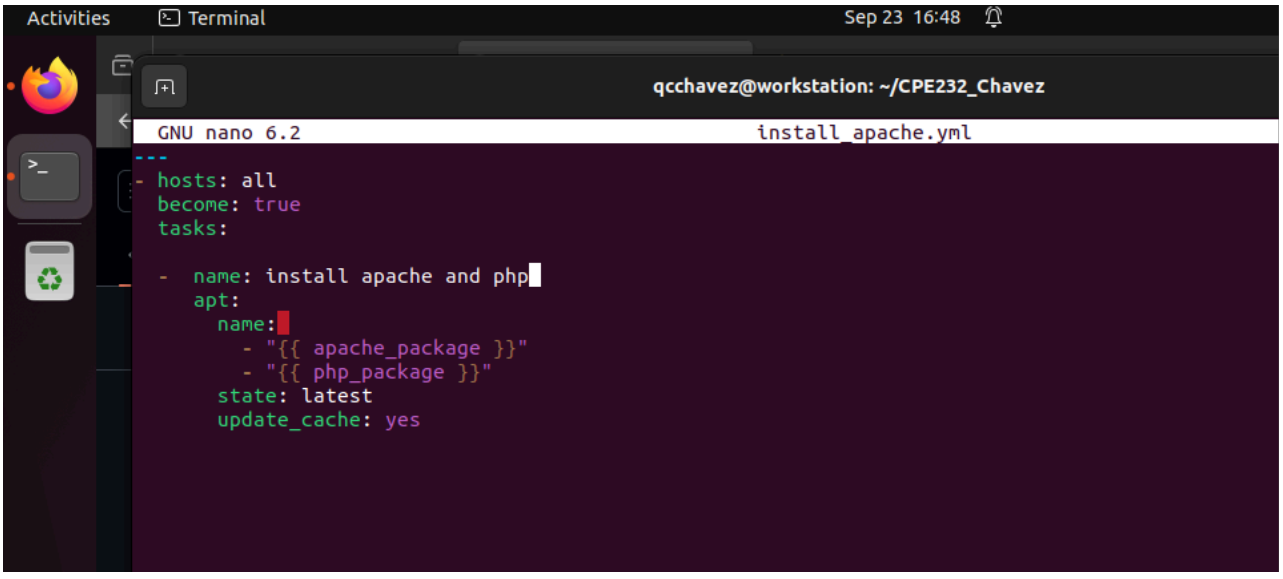


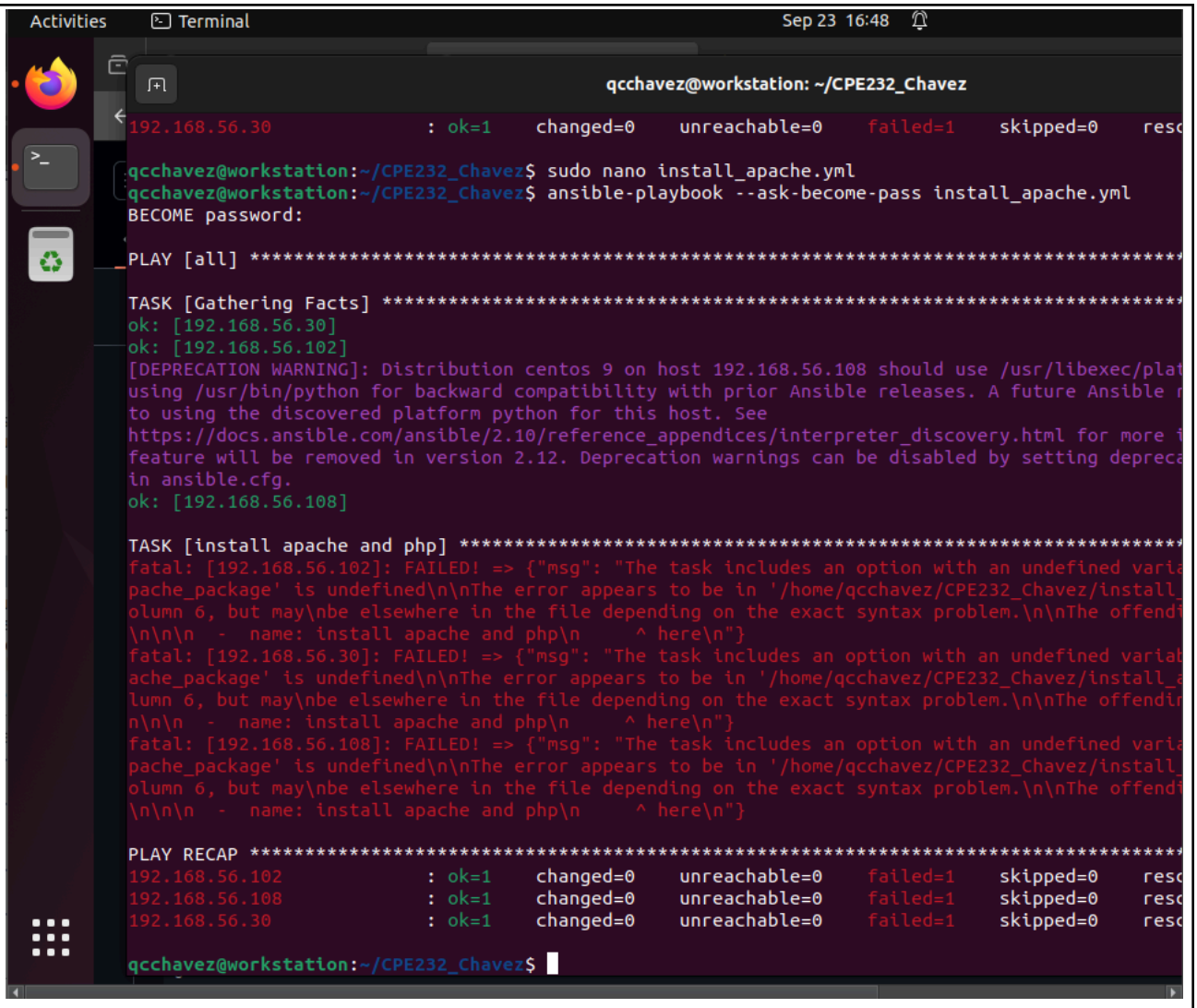
- In these screenshots, I've modified the **install_apache.yml** file again, but this time, reduced more lines to make it simpler. After modifying, I've prompted the command for activating the playbook. All remote servers (Server1, Server2, and CentOS) did the tasks completely, and skipped 1 task.

3. Finally, we can consolidate these 2 plays in just 1 play. This can be done by declaring variables that will represent the packages that we want to install. Basically, the apache_package and php_package are variables. The names are arbitrary, which means we can choose different names. We also take out the line when: ansible_distribution. Edit the playbook *install_apache.yml* again and make sure to follow the below image. Make sure to save the file and exit.

```
---
- hosts: all
  become: true
  tasks:

  - name: install apache and php
    apt:
      name:
        - "{{ apache_package }}"
        - "{{ php_package }}"
      state: latest
      update_cache: yes
```

Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.





- In this screenshot, I've modified the **install_apache.yml** file in its last required form, The packages are automatically detected. After modifying, I've prompted the command for activating the playbook. The three remote hosts failed on the task.

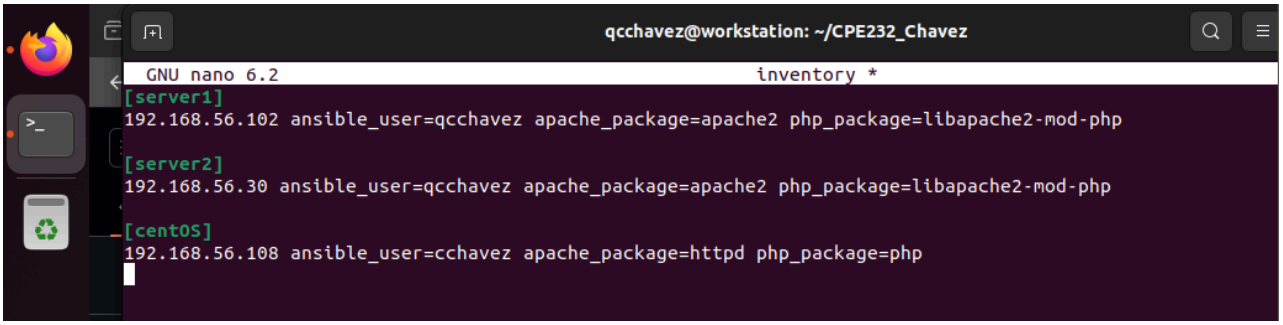
4. Unfortunately, task 2.3 was not successful. It's because we need to change something in the inventory file so that the variables we declared will be in place. Edit the *inventory* file and follow the below configuration:

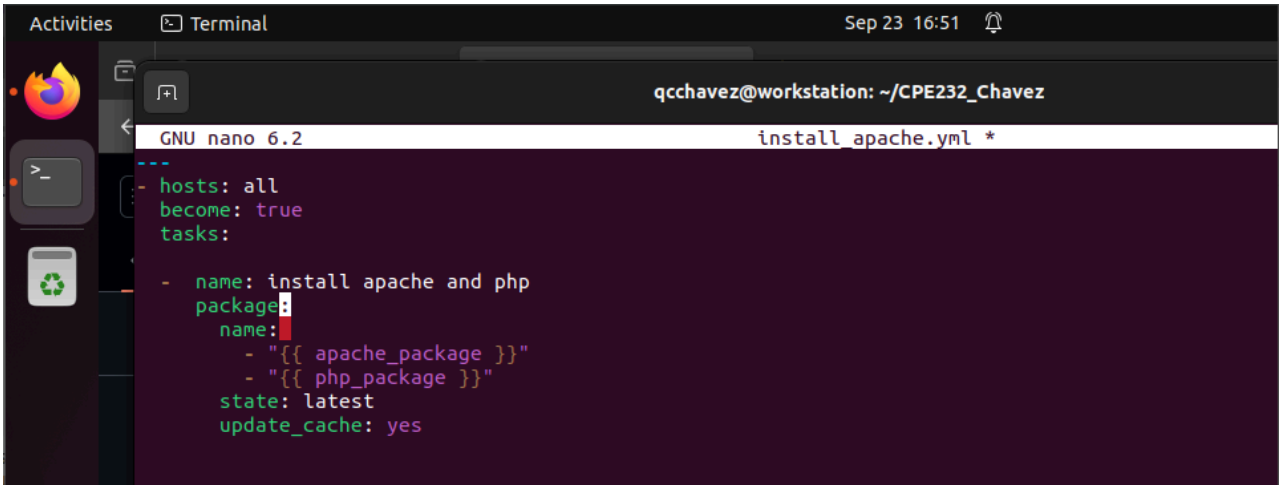
```
192.168.56.120 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.121 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.122 apache_package=httpd php_package=php
```

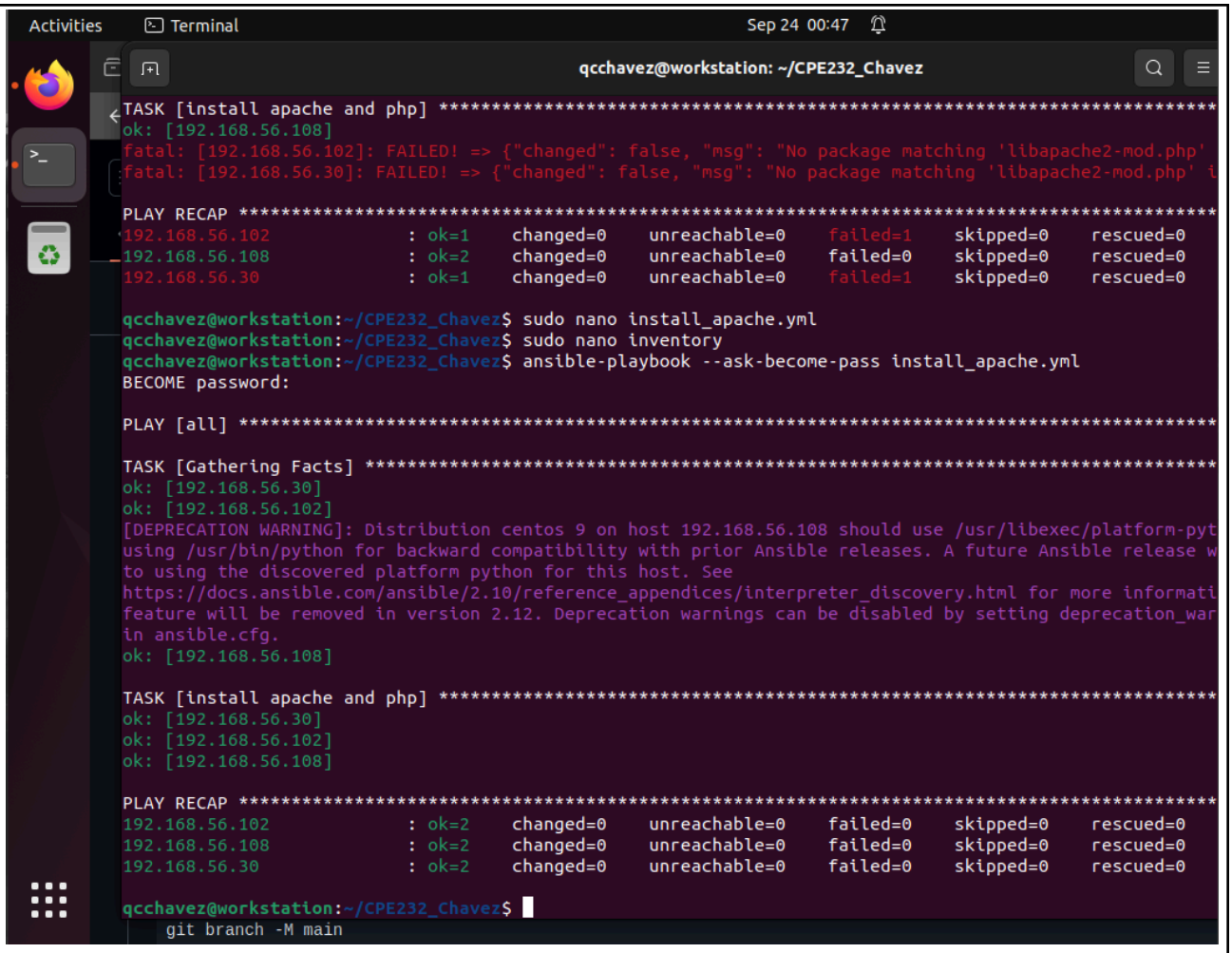
Make sure to save the *inventory* file and exit.

Finally, we still have one more thing to change in our *install_apache.yml* file. In task 2.3, you may notice that the package is assign as apt, which will not run in CentOS. Replace the *apt* with *package*. Package is a module in ansible that is generic, which is going to use whatever package manager the underlying host or the target server uses. For Ubuntu it will automatically use *apt*, and for CentOS it will automatically use *dnf*. Make sure to save the file and exit. For more details about the ansible package, you may refer to this documentation: [ansible.builtin.package – Generic OS package manager — Ansible Documentation]

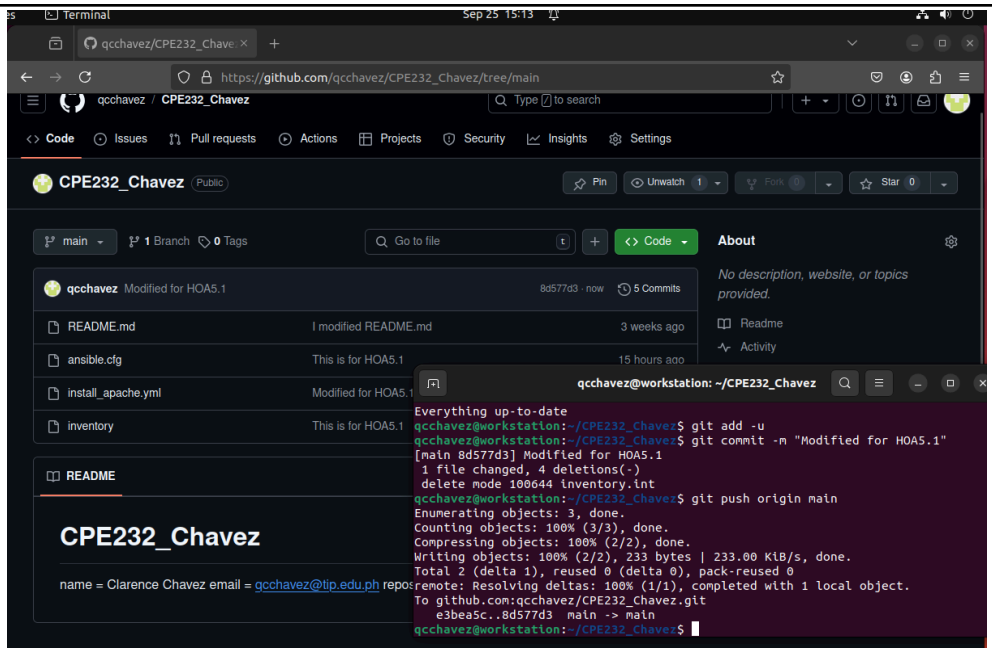
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.







- In these screenshots, I've modified two files such as: **install_apache.yml** file and **inventory** file, The packages are automatically detected based on the syntaxes I've inserted inside the **inventory** file. After modifying, I've prompted the command for activating the playbook. The three remote hosts (Server1, Server2, and CentOS) completed the tasks

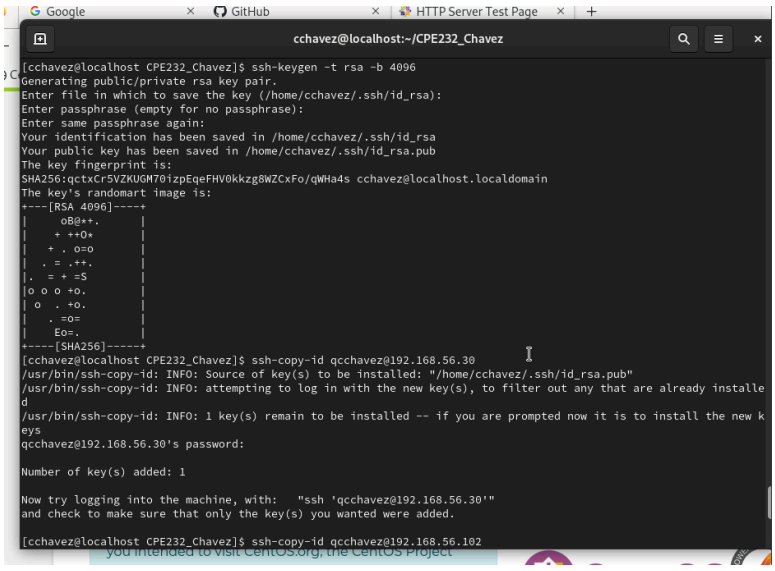


- In this screenshot, I've made a commitment to Github with the repository.

Supplementary Activity:

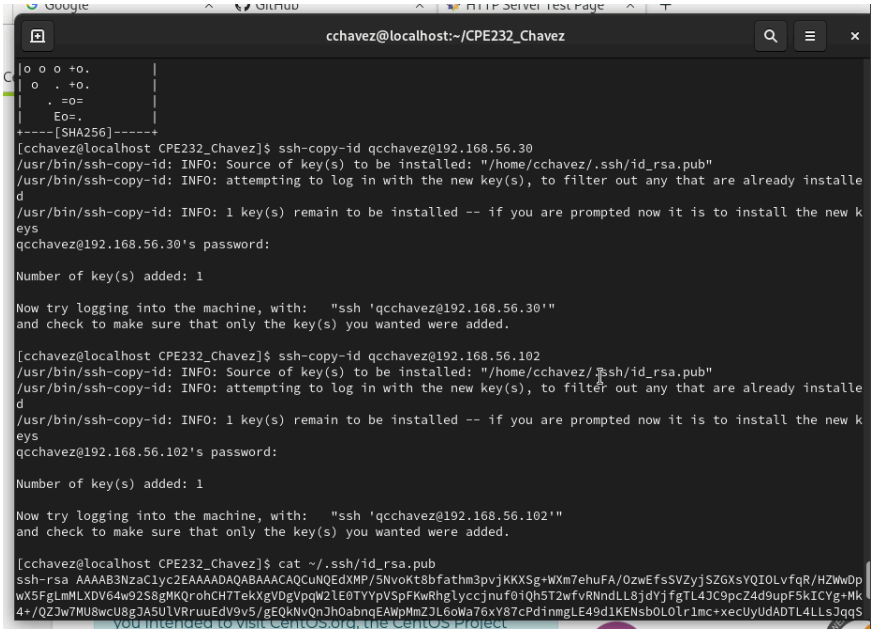
1. Create a playbook that could do the previous tasks in Red Hat OS.

Generating ssh public key for Red Hat OS



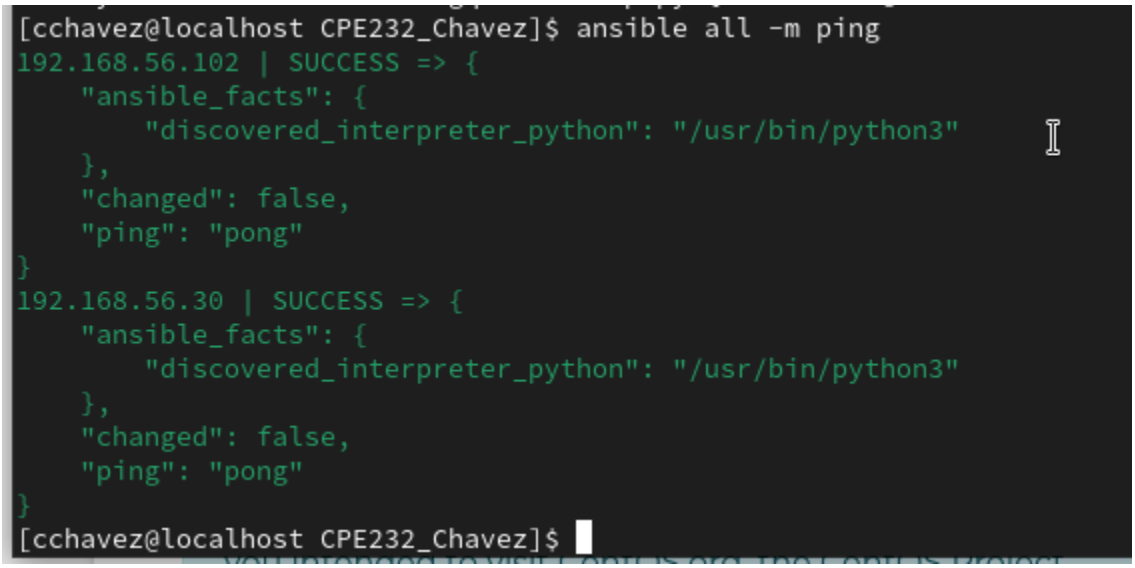
- In this screenshot, I've generated an ssh public key in order for me to make the ssh connections to the remote servers (Server1 and Server2) possible.

Copying ssh public keys from Red Hat OS to Server1 and Server2



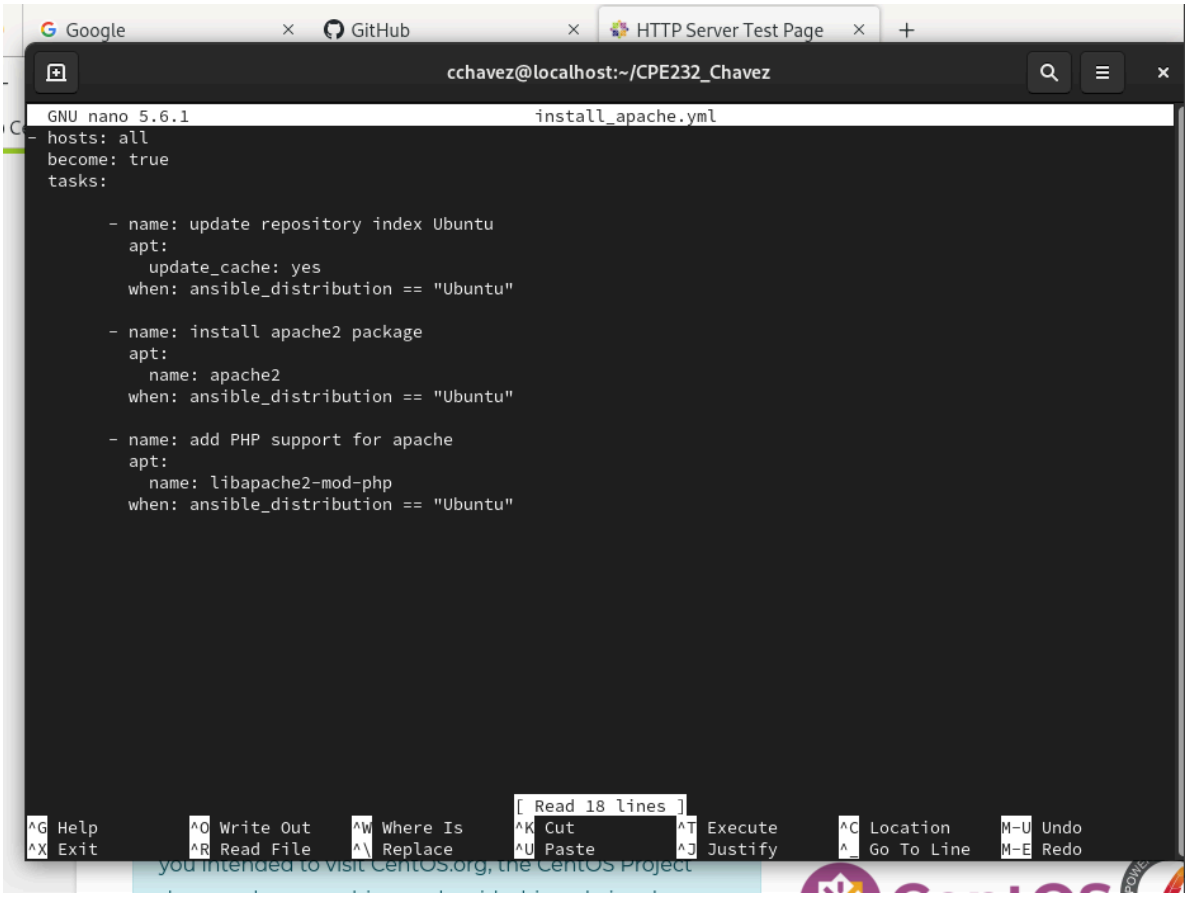
- In this screenshot, I've copied the Red Hat OS' ssh public key in order, so that it can access the servers remotely (Server1 and Server2).

Testing if ansible ping will work on both Server1 and Server2



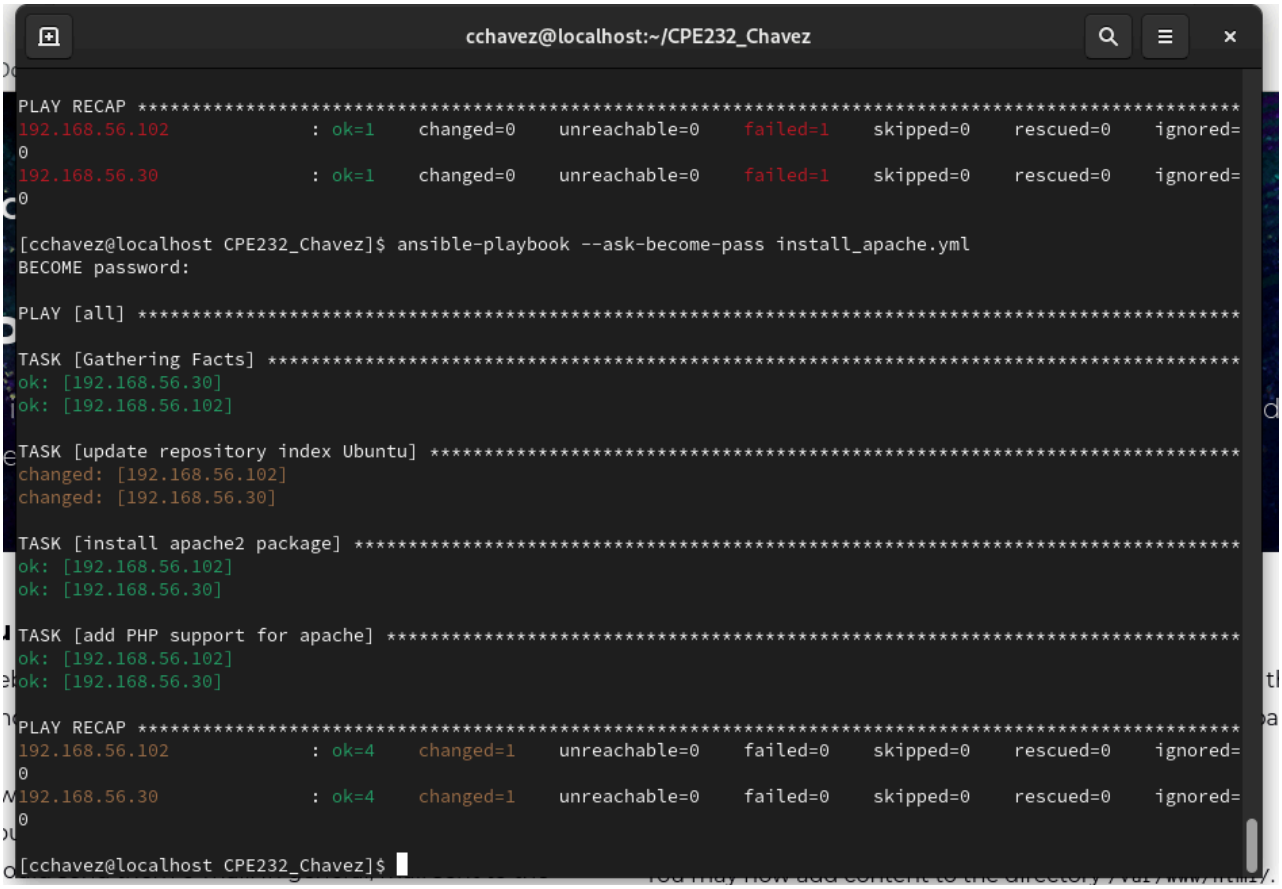
- In this screenshot, I've pinged the two servers remotely (Server1 and Server2) in Red Hat using ansible to check if it is working properly.

Ansible Playbook in CentOS for Ubuntu Servers



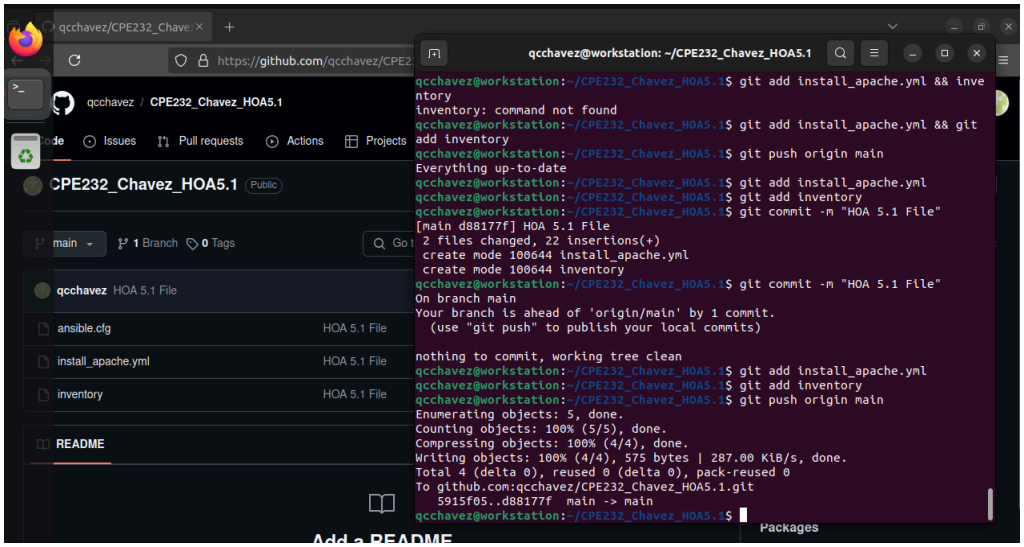
- In this screenshot, I have configured the install_apache.yml file for the two remote servers (Server1 and Server2).

Running the Ansible Playbook in Red Hat OS



- In this screenshot, it says that the repository index in the two Ubuntu servers (Server1 and Server2) was changed. The rest of the tasks were successfully done.

Committing the repository to Github



- In this screenshot, I've made a commitment to Github with the repository. Originally, the files were created at CPE232 directory to comply with the tasks above, but in order for me to determine which hands-on-activity that I'm working on, or I've already done, I've made a separate Github repository which is named as CPE232_Chavez_HOA5.1, after creating that separate repo, I've committed that repo to complete the whole Hands-on-Activity.

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?
   - Refactoring of playbook codes is important because it improves the readability as it is more concise. Since it is concise, that means it is very maintainable or less prone to errors.

2. When do we use the "when" command in the playbook?
   - In the playbook, we can use the **when** command to perform such conditions. It can also be used to check if a file exists or not, before performing any certain tasks under those conditions. This is also good for error handling since it implements error recovery mechanisms by executing alternate tasks when certain errors happen.