Name: FRIAS, ABEGAIL L.	Date Performed: September 27, 2024
Course/Section: CPE212 - CPE31S21	Date Submitted:September 29, 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

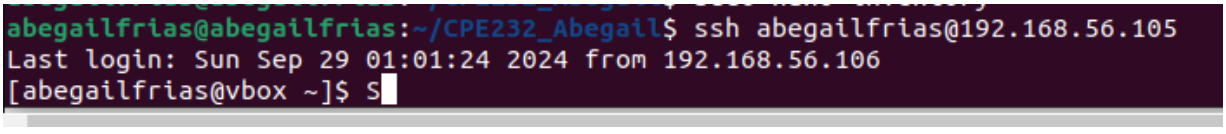
2. **Discussion**:

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.



```
[abegailfrias@vbox CPE232_Abegail]$ ssh abegailfrias@192.168.56.106
abegailfrias@192.168.56.106's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.5.0-18-generic x86_64)

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

Expanded Security Maintenance for Applications is not enabled.

282 updates can be applied immediately.
213 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Sep 28 23:20:48 2024 from 192.168.56.105
abegailfrias@abegailfrias:~$
```

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?

- The command updated the local repository with the content it had downloaded and retrieved from a remote repository. The command updated the local repository with the content it had downloaded and retrieved from a remote repository.

```
[abegailfrias@localhost ~]$ git clone git@github.com:wonbe/CPE232_Abegail.git
Cloning into 'CPE232_Abegail'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 6 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (6/6), done.
[abegailfrias@localhost ~]$ ls
CPE232_Abegail  Documents  Music     Public     Videos
Desktop         Downloads  Pictures  Templates
[abegailfrias@localhost ~]$ cd CPE232_Abegail
[abegailfrias@localhost CPE232_Abegail]$ git pull
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."

```
[abegailfrias@localhost CPE232_Abegail]$ ansible all -m apt -a update_cache=true
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'
```

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.

- Each given condition is followed by the command to complete the task.
  If you have a mix of Debian and Ubuntu servers, you can change the configuration of your playbook like this.

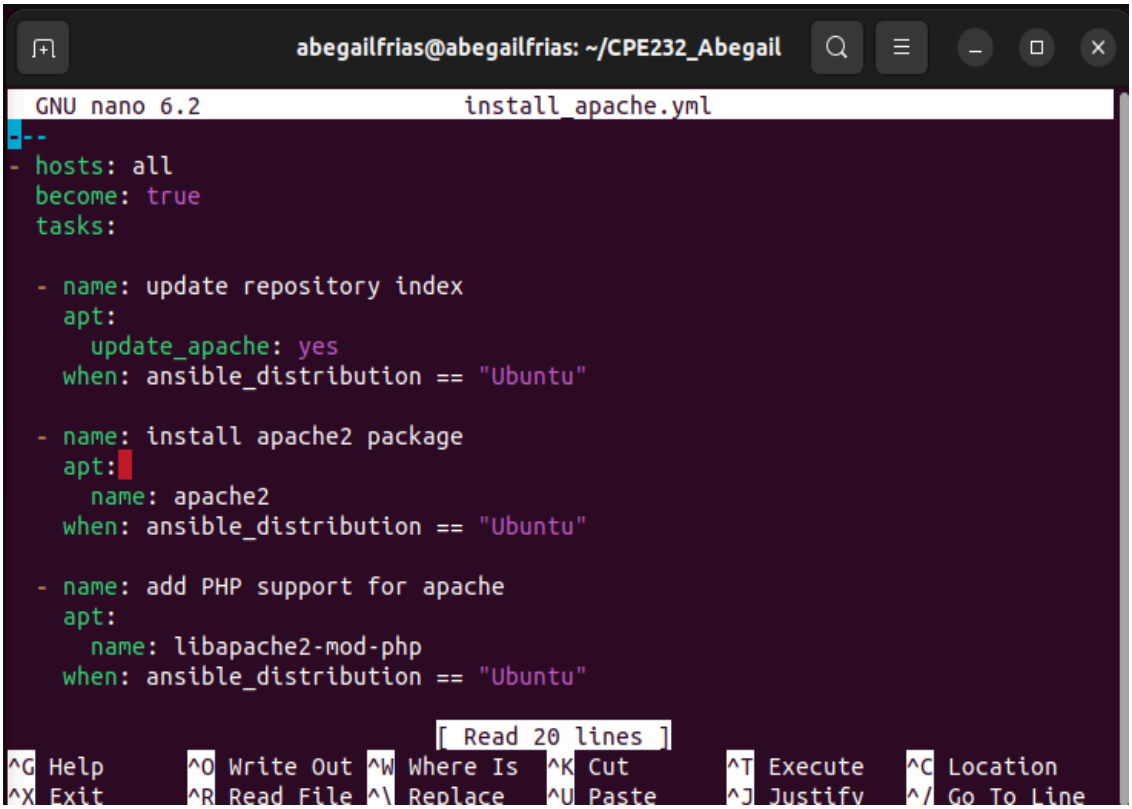
  - name: update repository index

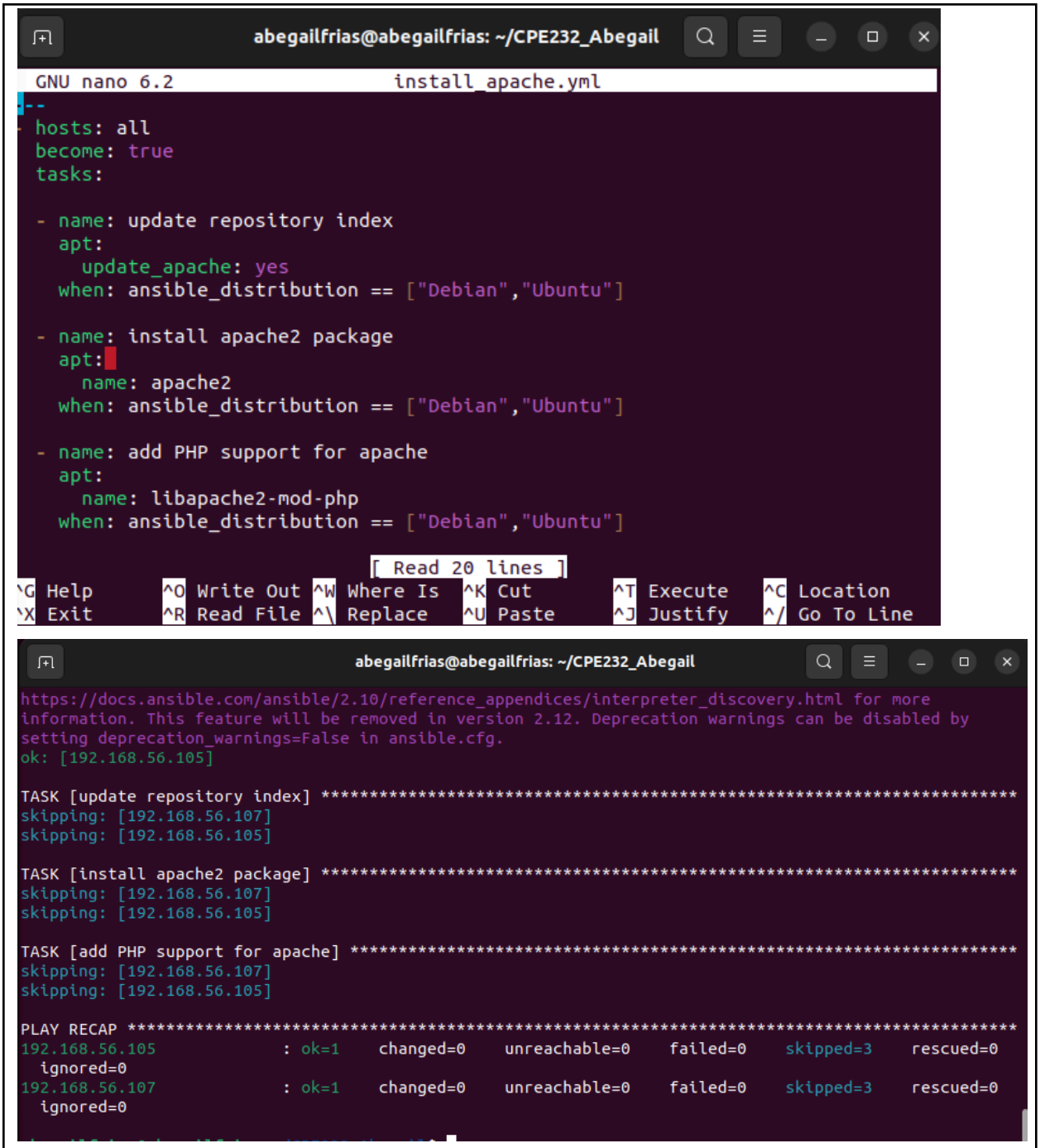
apt:

update_cache: yes

when: ansible_distribution in [“Debian”, “Ubuntu]

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





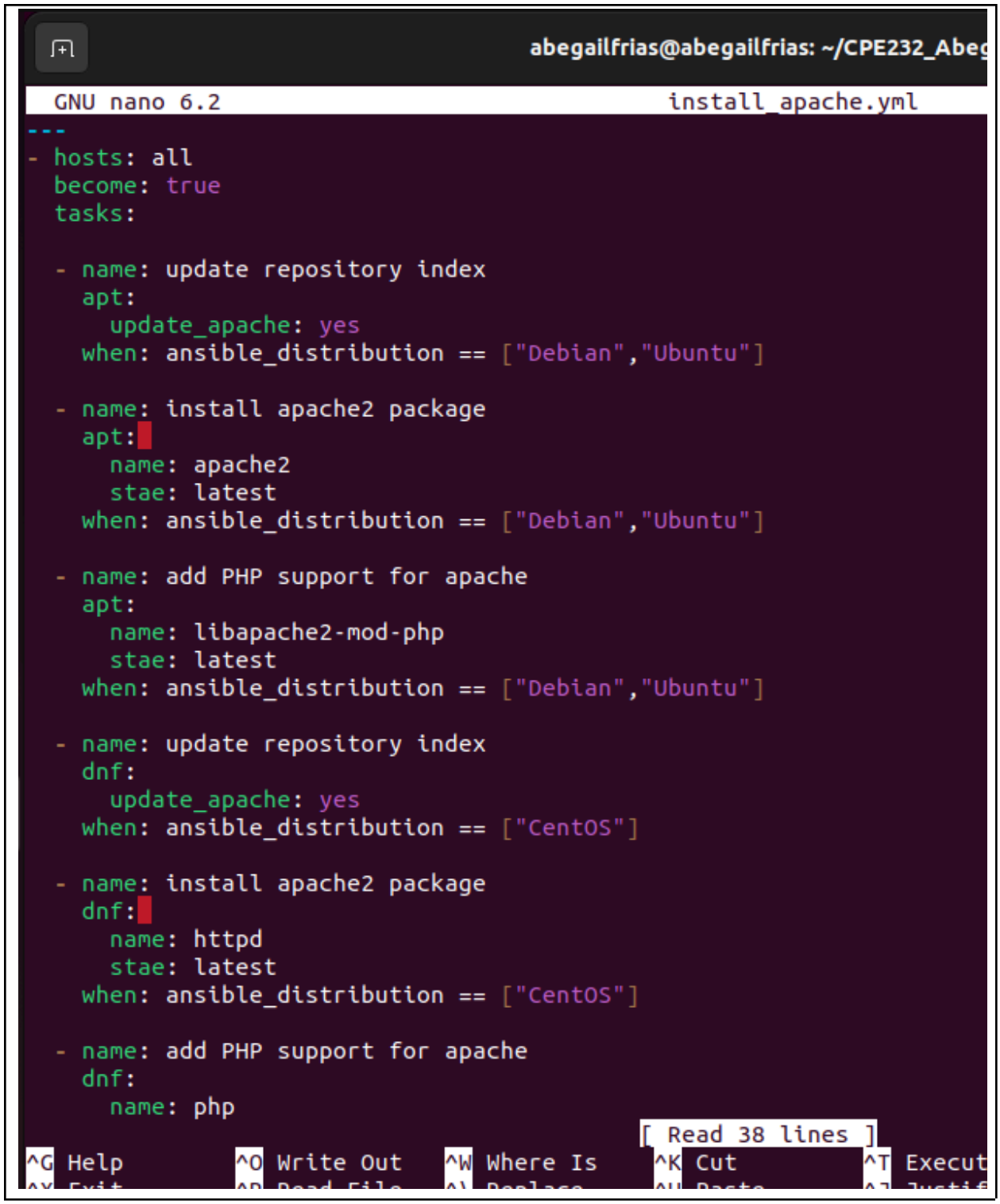
4. Edit the *install_apache.yml* file and insert the lines shown below.

```
---
- 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"
```

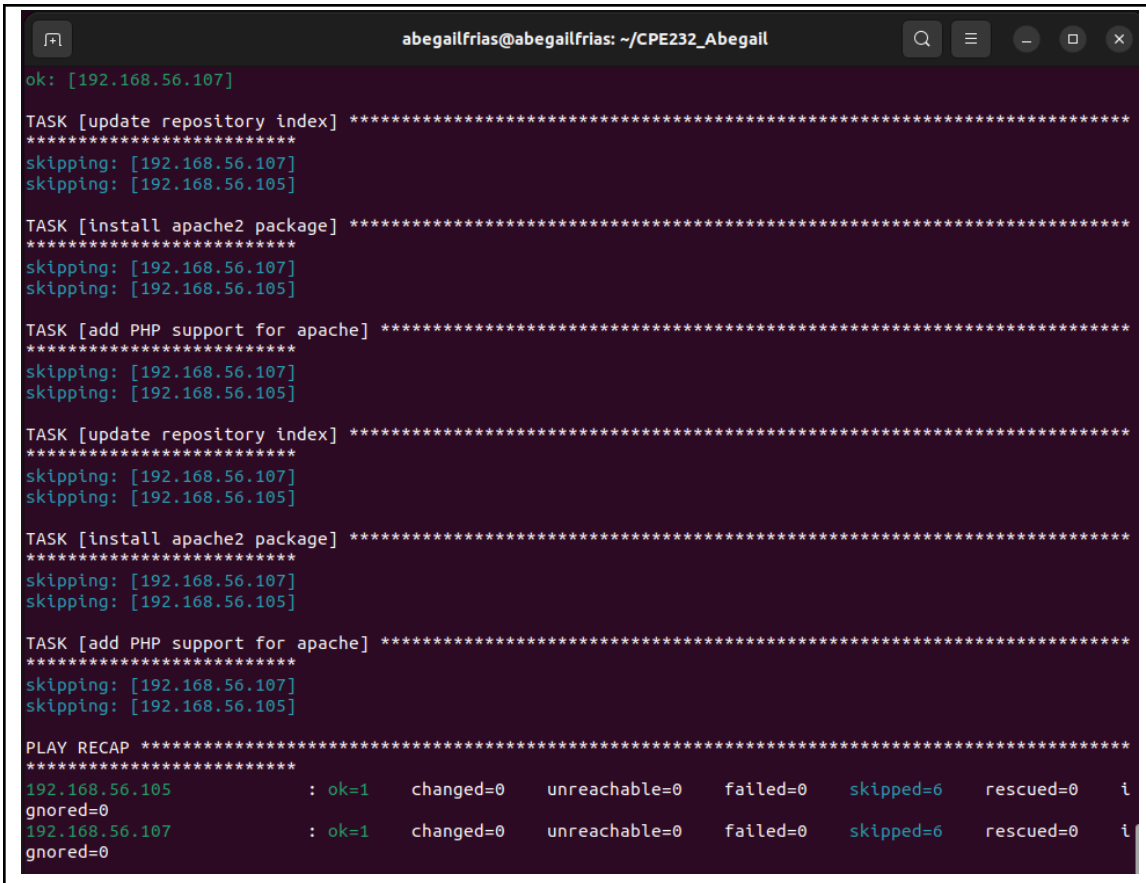
Make sure to save and exit.

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

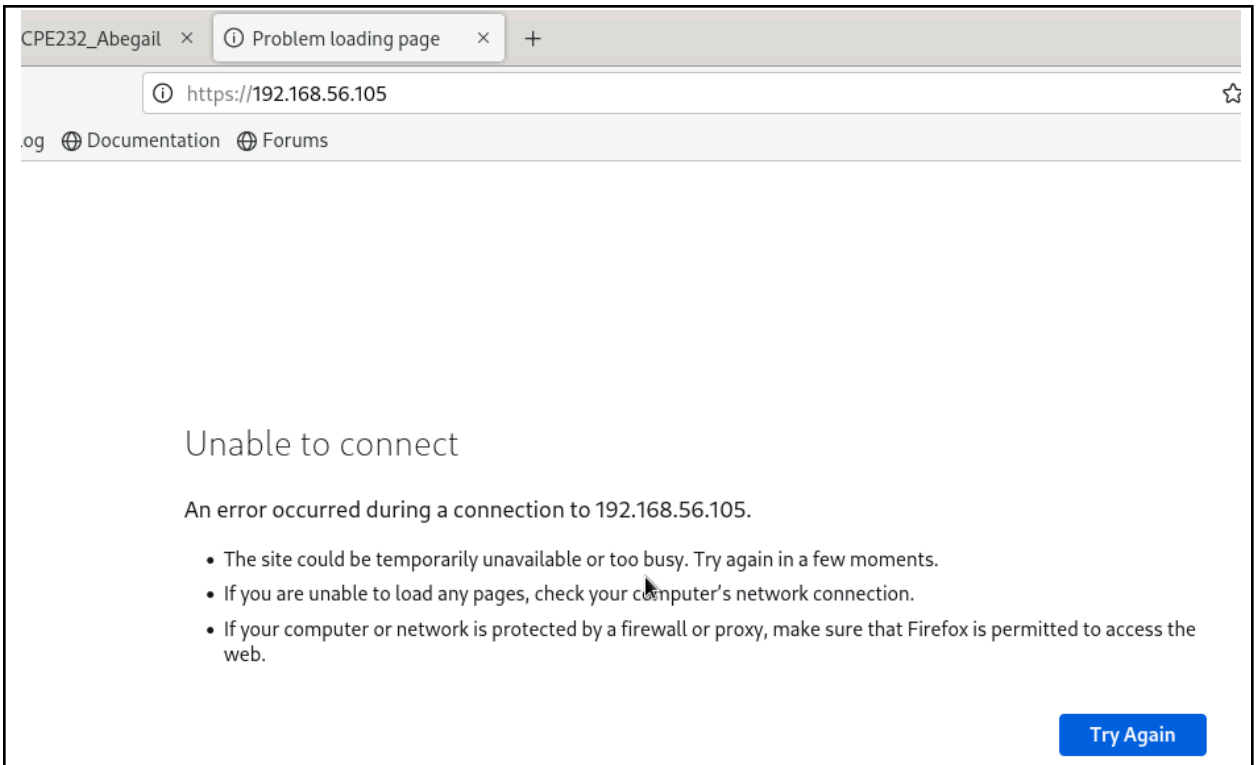
- Each given condition is followed by the command to complete the task.


abegailfrias@abegailfrias: ~/CPE232_Abeg
GNU nano 6.2 install_apache.yml

- hosts: all
become: true
tasks:
- name: update repository index
apt:
update_apache: yes
when: ansible_distribution == ["Debian","Ubuntu"]
- name: install apache2 package
apt:
name: apache2
stae: latest
when: ansible_distribution == ["Debian","Ubuntu"]
- name: add PHP support for apache
apt:
name: libapache2-mod-php
stae: latest
when: ansible_distribution == ["Debian","Ubuntu"]
- name: update repository index
dnf:
update_apache: yes
when: ansible_distribution == ["CentOS"]
- name: install apache2 package
dnf:
name: httpd
stae: latest
when: ansible_distribution == ["CentOS"]
- name: add PHP support for apache
dnf:
name: php
[Read 38 lines]
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execut



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.

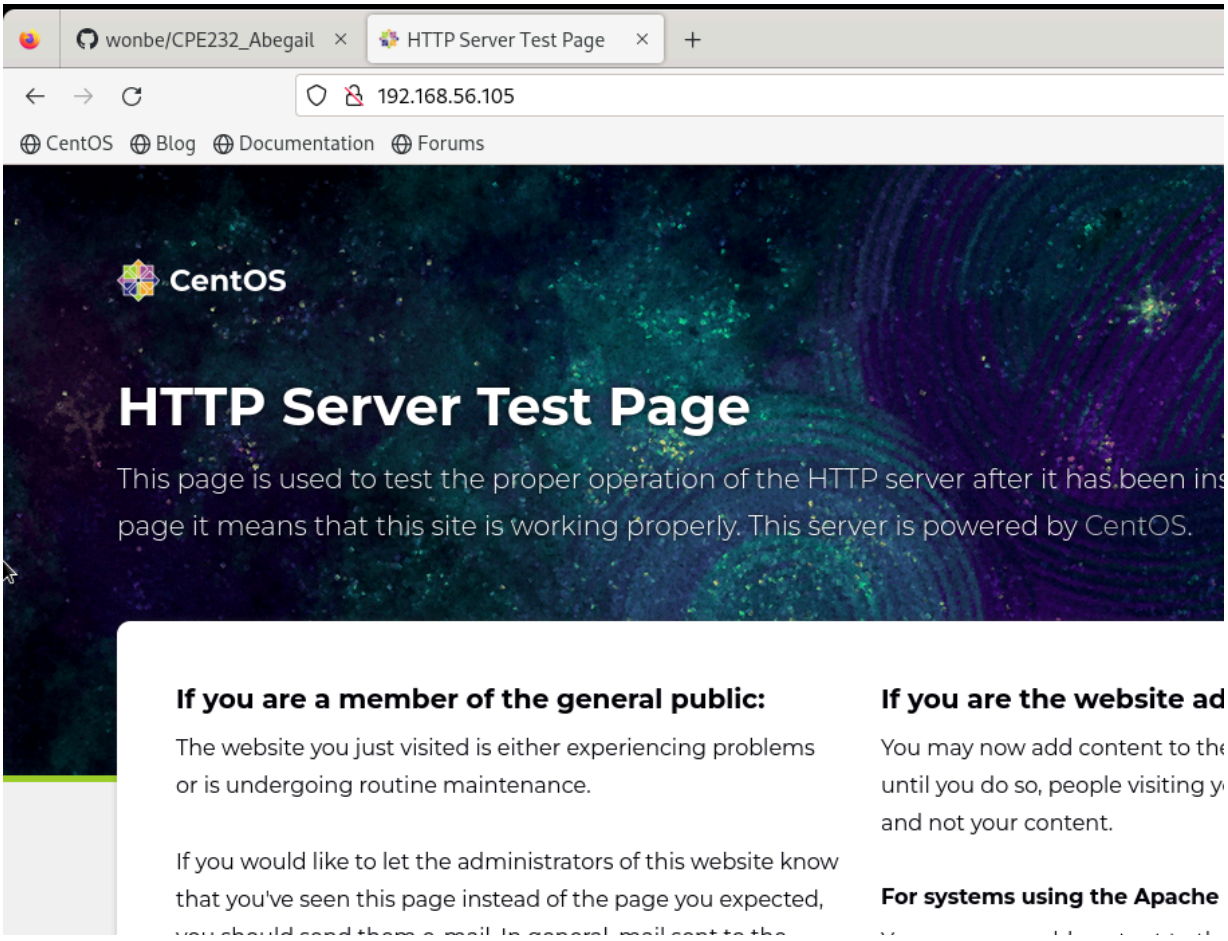


5.1 To activate, go to the CentOS VM terminal and enter the following:
systemctl status httpd
The result of this command tells you that the service is inactive.

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)

```
[abegailfrias@vbox CPE232_Abegail]$ sudo systemctl start httpd
[abegailfrias@vbox CPE232_Abegail]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
     Active: active (running) since Sat 2024-09-28 23:56:34 PST; 13s ago
       Docs: man:httpd.service(8)
   Main PID: 23110 (httpd)
     Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec:   0 >
      Tasks: 177 (limit: 10950)
     Memory: 22.1M
        CPU: 77ms
     CGroup: /system.slice/httpd.service
             ├─23110 /usr/sbin/httpd -DFOREGROUND
             ├─26087 /usr/sbin/httpd -DFOREGROUND
             ├─26088 /usr/sbin/httpd -DFOREGROUND
             ├─26099 /usr/sbin/httpd -DFOREGROUND
             └─26109 /usr/sbin/httpd -DFOREGROUND
lines 1-15/15 (END)
[abegailfrias@vbox CPE232_Abegail]$
```

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)



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.

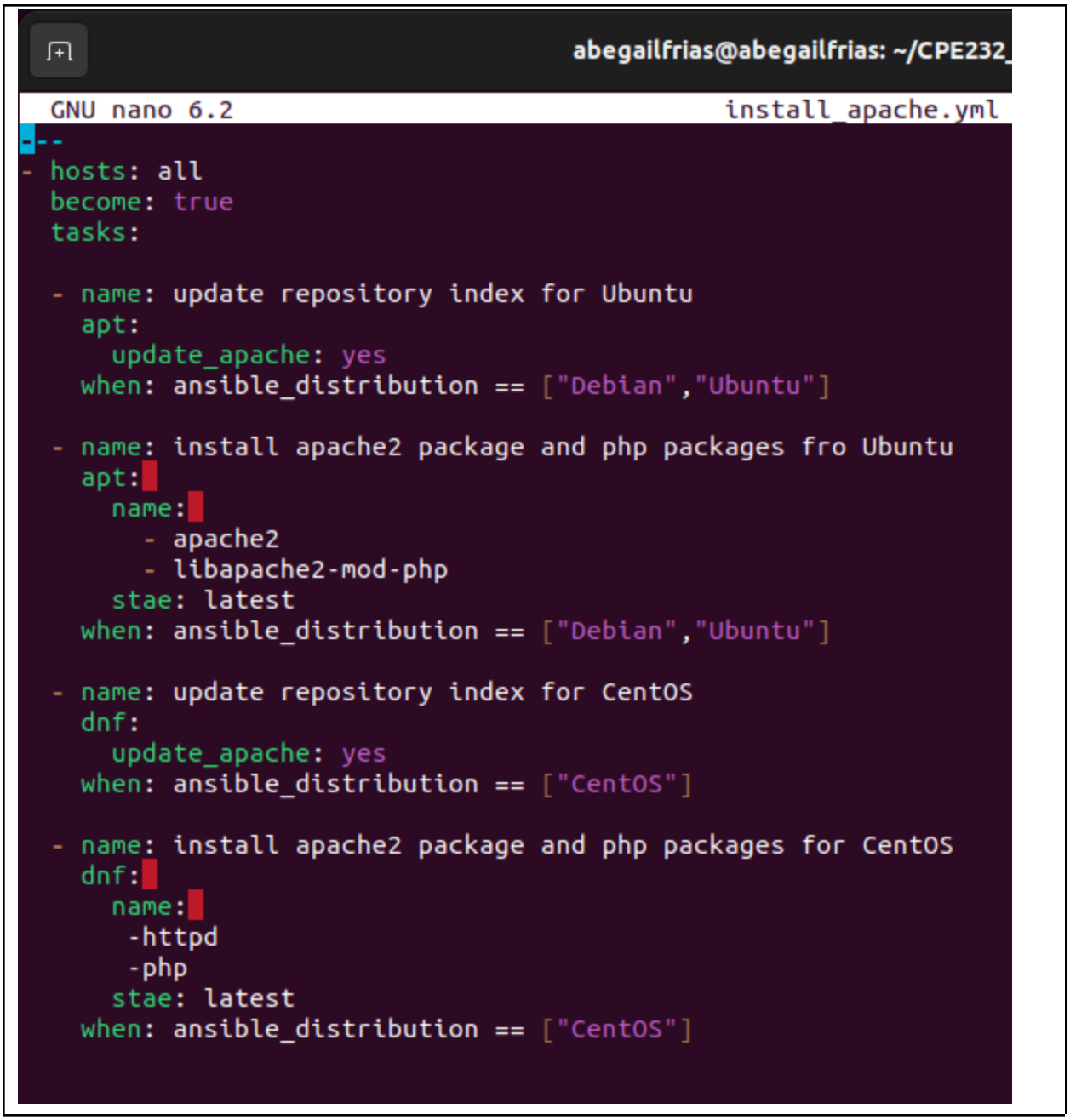
1. Edit the playbook *install_apache.yml*. Currently, we have three tasks targeting our Ubuntu machines and 3 tasks targeting our CentOS machine. Right now, we try to consolidate some tasks that are typically the same. For example, we can consolidate two plays that install packages. We can do that by 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"
```

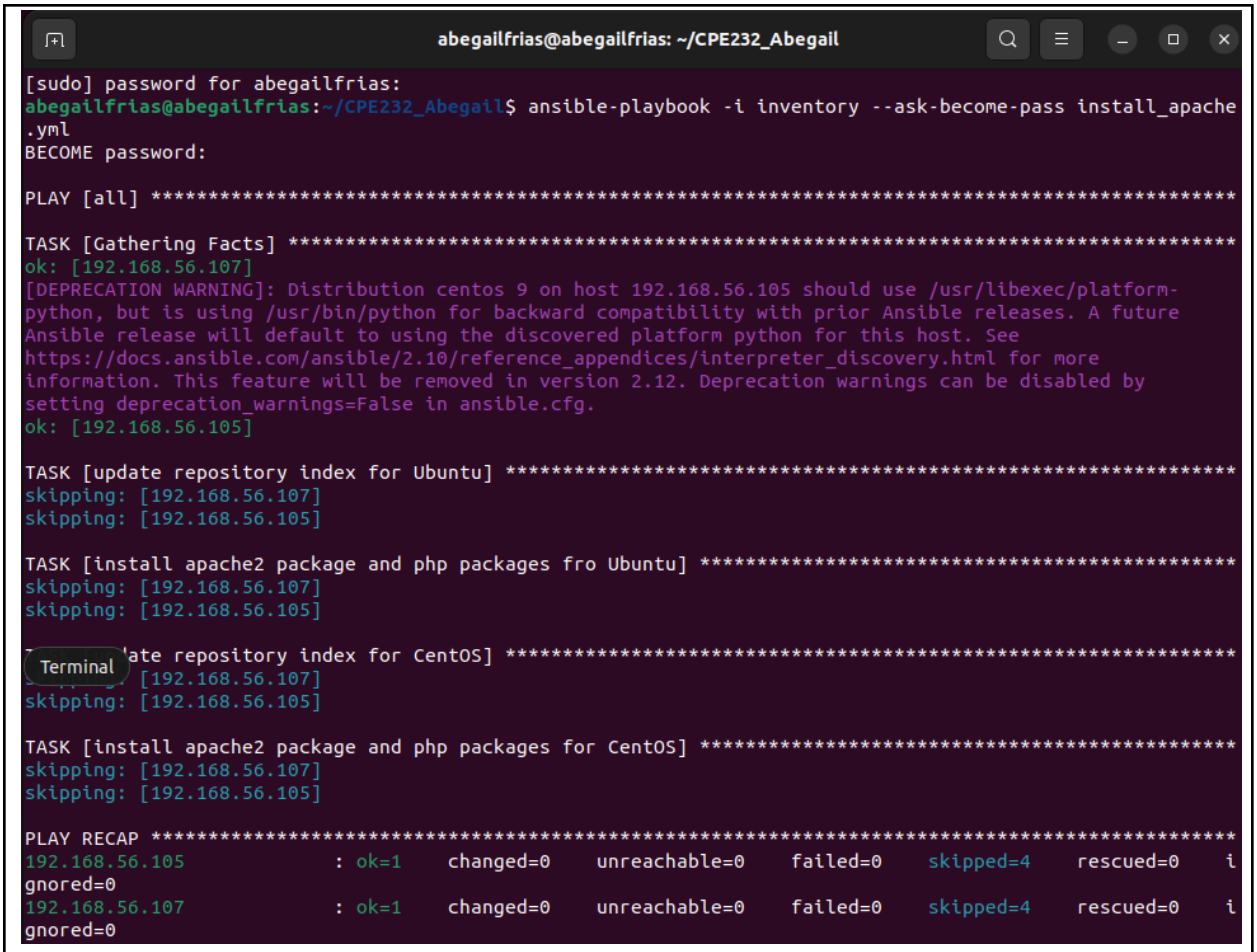
Make sure to save the file and exit.

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

- The same task is included, but it has been simplified to run more quickly.


abegailfrias@abegailfrias: ~/CPE232_
GNU nano 6.2
install_apache.yml

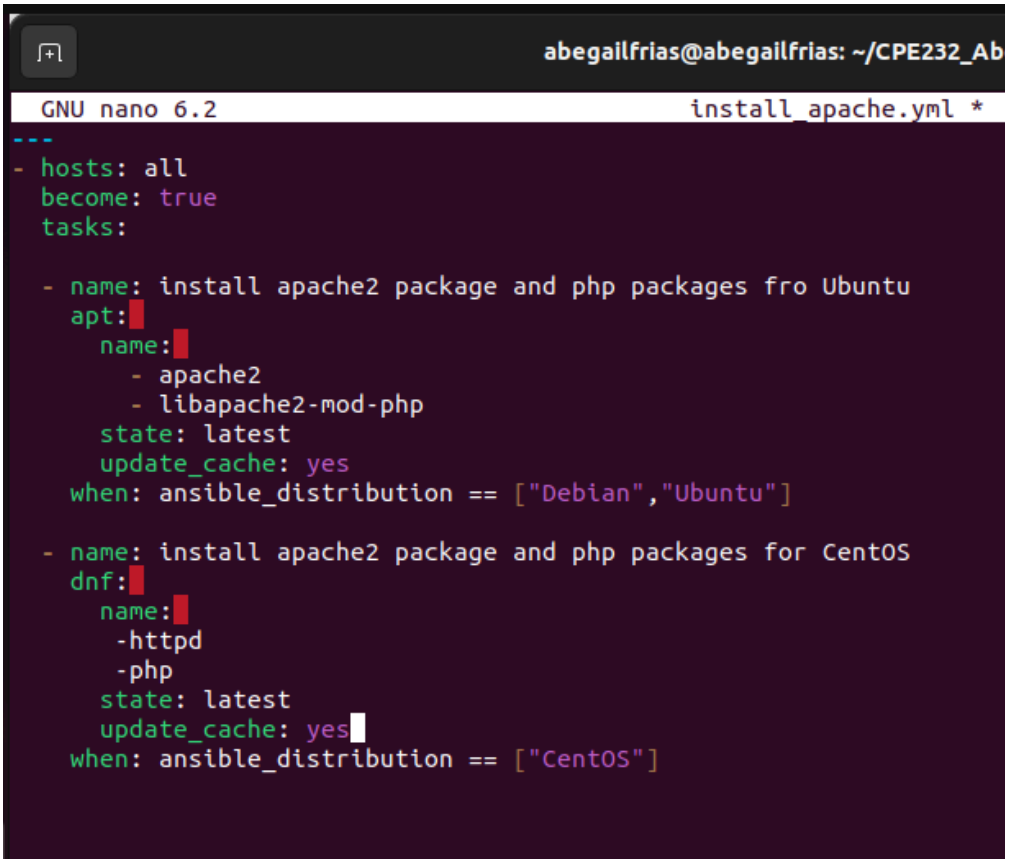
- hosts: all
become: true
tasks:
- name: update repository index for Ubuntu
apt:
update_apache: yes
when: ansible_distribution == ["Debian","Ubuntu"]
- name: install apache2 package and php packages fro Ubuntu
apt:
name:
- apache2
- libapache2-mod-php
stae: latest
when: ansible_distribution == ["Debian","Ubuntu"]
- name: update repository index for CentOS
dnf:
update_apache: yes
when: ansible_distribution == ["CentOS"]
- name: install apache2 package and php packages for CentOS
dnf:
name:
-httpd
-php
stae: latest
when: ansible_distribution == ["CentOS"]



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.

- Because there are now only two executable commands for each task, the code is much more readable and straightforward.

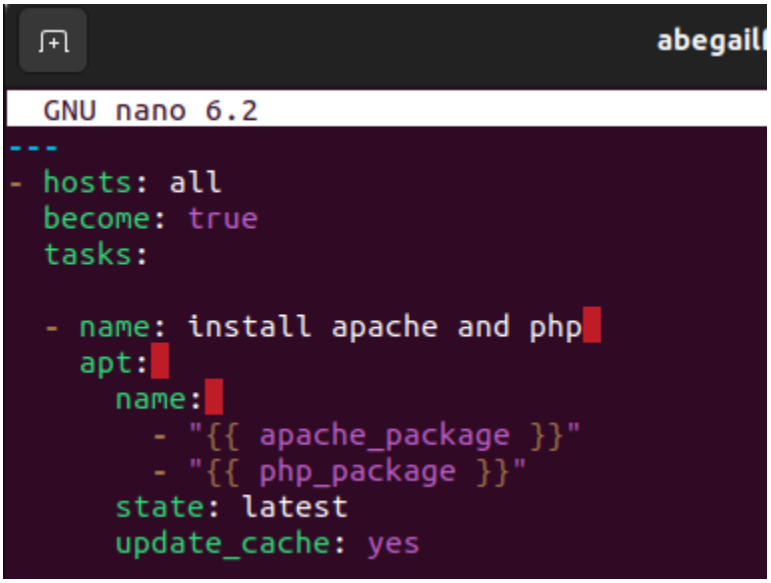
```
abegailfrias@abegailfrias:~/CPE232_Abegail$ ansible-playbook -i inventory --ask-become-pass install_apache
.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************************************
ok: [192.168.56.107]
[DEPRECATION WARNING]: Distribution centos 9 on host 192.168.56.105 should use /usr/libexec/platform-
python, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future
Ansible release will default to using the discovered platform python for this host. See
https://docs.ansible.com/ansible/2.10/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
ok: [192.168.56.105]

TASK [install apache2 package and php packages fro Ubuntu] *********************************************
skipping: [192.168.56.107]
skipping: [192.168.56.105]

TASK [install apache2 package and php packages for CentOS] *********************************************
skipping: [192.168.56.107]
skipping: [192.168.56.105]

PLAY RECAP *********************************************************************************************
192.168.56.105             : ok=1    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    i
gnored=0
192.168.56.107             : ok=1    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    i
gnored=0
```

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.

- The variable has no significant meaning, even though the command failed.



```
abegailfrias@abegailfrias:~/CPE232_Abegail$ ansible-playbook -i inventory --ask-become-pass install_apache
.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************************************
ok: [192.168.56.107]
[DEPRECATION WARNING]: Distribution centos 9 on host 192.168.56.105 should use /usr/libexec/platform-
python, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future
Ansible release will default to using the discovered platform python for this host. See
https://docs.ansible.com/ansible/2.10/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
ok: [192.168.56.105]

TASK [install apache and php] **************************************************************************
fatal: [192.168.56.107]: FAILED! => {"msg": "The task includes an option with an undefined variable. The e
rror was: 'apache_package' is undefined\n\nThe error appears to be in '/home/abegailfrias/CPE232_Abegail/i
nstall_apache.yml': line 6, column 5, but may\nbe elsewhere in the file depending on the exact syntax prob
lem.\n\nThe offending line appears to be:\n\n\n  - name: install apache and php\n    ^ here\n"}
fatal: [192.168.56.105]: FAILED! => {"msg": "The task includes an option with an undefined variable. The e
rror was: 'apache_package' is undefined\n\nThe error appears to be in '/home/abegailfrias/CPE232_Abegail/i
nstall_apache.yml': line 6, column 5, but may\nbe elsewhere in the file depending on the exact syntax prob
lem.\n\nThe offending line appears to be:\n\n\n  - name: install apache and php\n    ^ here\n"}

PLAY RECAP *********************************************************************************************
192.168.56.105             : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    i
gnored=0
192.168.56.107             : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    i
gnored=0
```

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
```

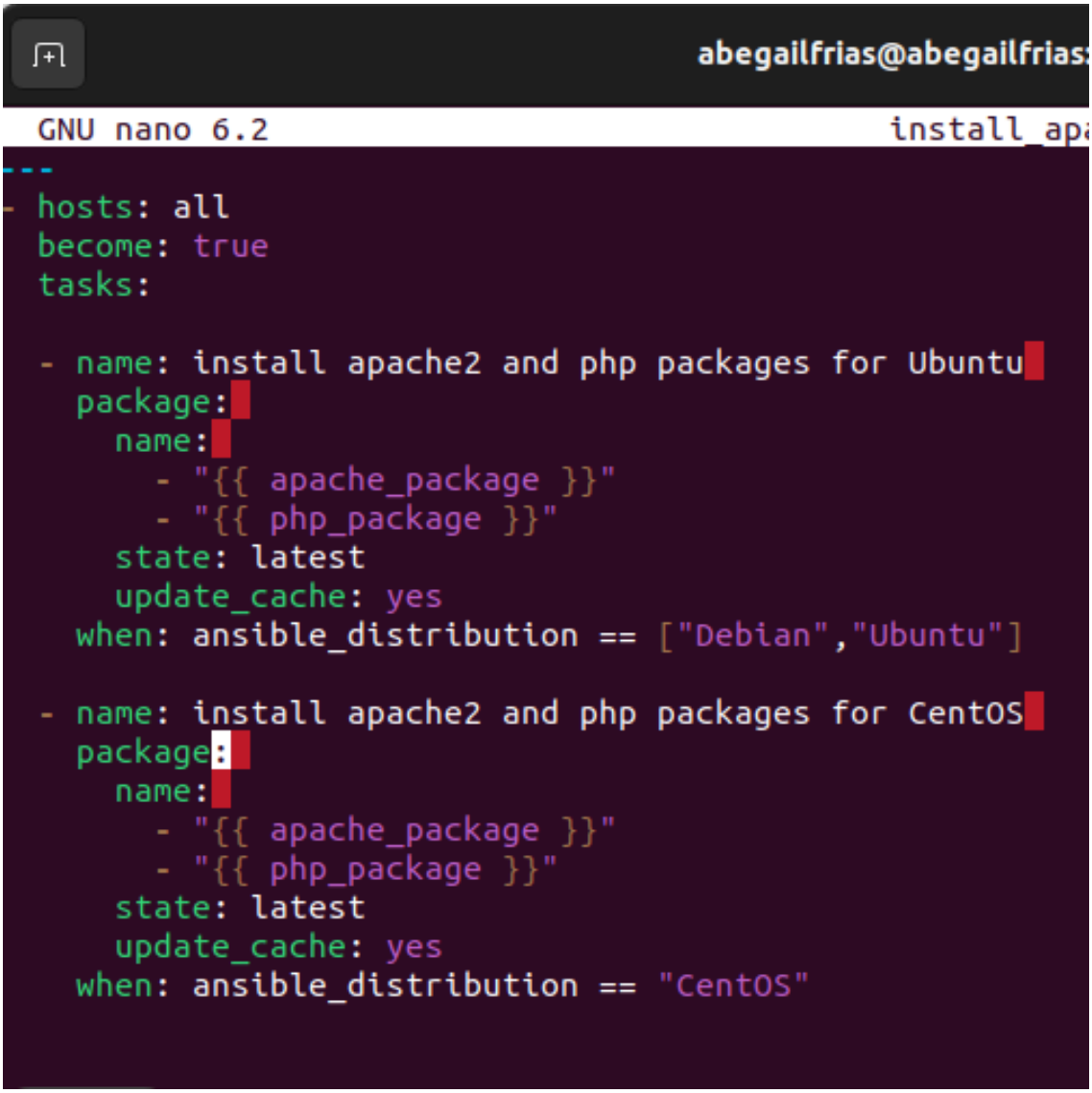
Make sure to save the *inventory* file and exit.

```
abegailfrias@abegailfrias: ~/CPE232_Abegail

  GNU nano 6.2                                    inventory
[webservers]
192.168.56.107 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.105 apache_package=apache2 php_package=libapache2-mod-php
```

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.



```
abegailfrias@abegailfrias:~/CPE232_Abegail$ ansible-playbook -i inventory --ask-become-pass install_apache
.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************************************
ok: [192.168.56.107]
[DEPRECATION WARNING]: Distribution centos 9 on host 192.168.56.105 should use /usr/libexec/platform-
python, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future
Ansible release will default to using the discovered platform python for this host. See
https://docs.ansible.com/ansible/2.10/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
ok: [192.168.56.105]

TASK [install apache2 and php packages for Ubuntu] *****************************************************
skipping: [192.168.56.107]
skipping: [192.168.56.105]

TASK [install apache2 and php packages for CentOS] *****************************************************
skipping: [192.168.56.107]
fatal: [192.168.56.105]: FAILED! => {"changed": false, "failures": ["No package apache2 available.", "No p
ackage libapache2-mod-php available."], "msg": "Failed to install some of the specified packages", "rc": 1
, "results": []}

PLAY RECAP *********************************************************************************************
192.168.56.105             : ok=1    changed=0    unreachable=0    failed=1    skipped=1    rescued=0    i
gnored=0
192.168.56.107             : ok=1    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    i
gnored=0
```

Supplementary Activity:

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

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?
   - Refactoring playbook codes improves maintainability and efficiency, which makes them crucial. Playbooks make complex automation processes easier to manage and expedite by combining several tasks into a single command. This reduces the possibility of mistakes and expedites processes, particularly in extensive implementations.
2. When do we use the “when” command in playbook?
   - Ansible playbooks use the "when" command to conditionally execute tasks based on predefined criteria, like the managed nodes' operating system. This is especially helpful when working with various distributions, such as CentOS and Ubuntu, as they might call for various package management commands. An automated process can run more smoothly if the "when" command is used to make sure that the right packages and syntax are applied depending on the target system.