Name: Reyes, Alexzander J.	**Date Performed: 12/09/2025**
Course/Section:CPE 212- CPE31S2	**Date Submitted: 12/09/2025**
Instructor: Engr. Robin Valenzuela	**Semester and SY: 1st Sem & 2025-2026**

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**:

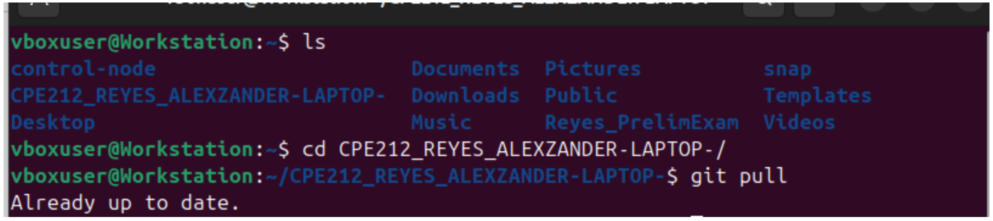
We are going to look at a way that we can differentiate a playbook by a host in terms of which distribution the host is running. It's very common in most Linux shops to run multiple distributions, for example, Ubuntu shop or Debian shop and you need a different distribution for a one off-case or perhaps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

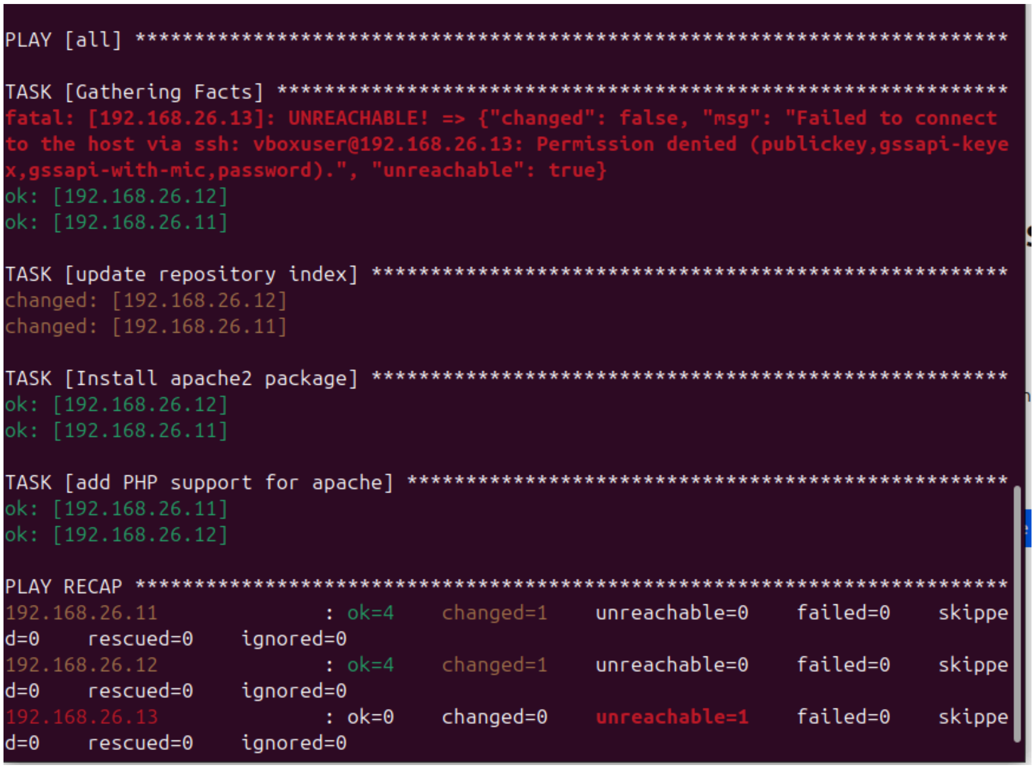
1. In the local machine, make sure you are in the local repository directory (*CPE232_yourname*). Issue the command git pull. When prompted, enter the correct passphrase or password. Describe what happened when you issue this command. Did something happen? Why?



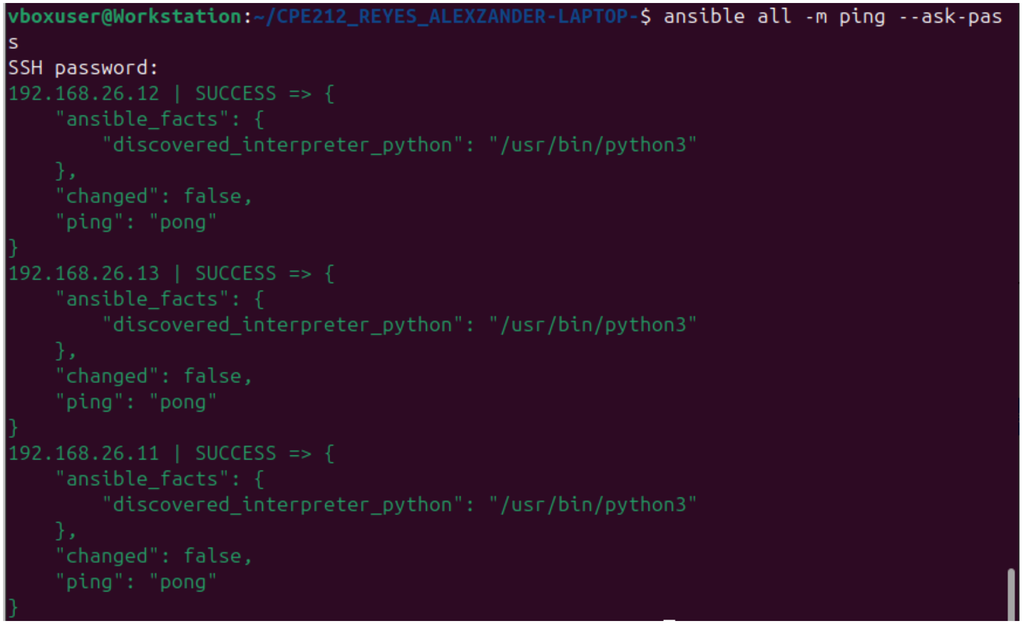
```
vboxuser@Workstation:~$ ls
control-node                    Documents  Pictures          snap
CPE212_REYES_ALEXZANDER-LAPTOP- Downloads  Public            Templates
Desktop                         Music      Reyes_PrelimExam  Videos
vboxuser@Workstation:~$ cd CPE212_REYES_ALEXZANDER-LAPTOP-/
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ git pull
Already up to date.
```

- Already up to date.
- local repository was already in sync with the remote.

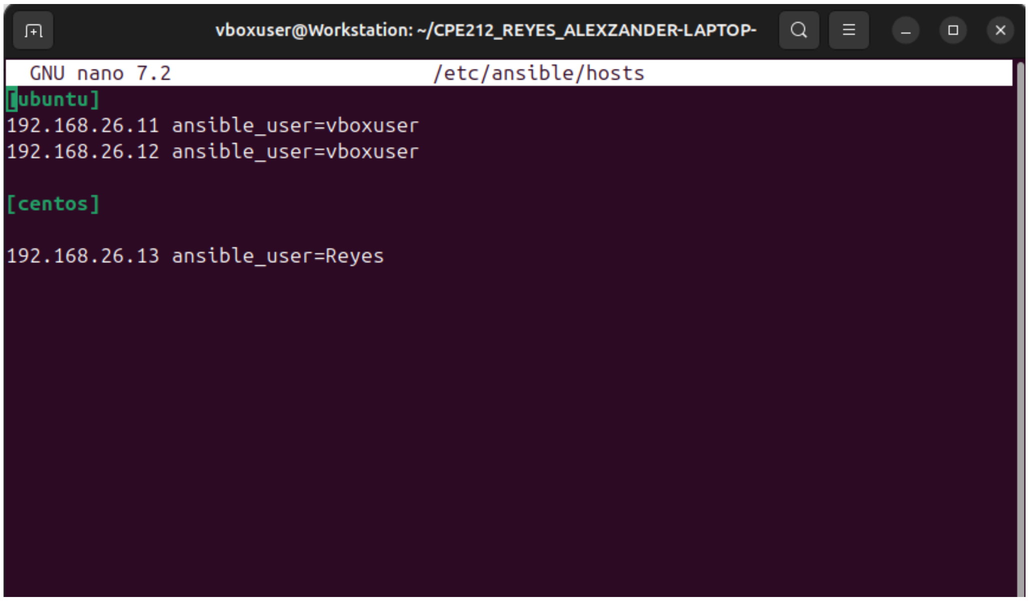
2. Edit the inventory file and add the IP address of the Centos VM. Issue the command we used to execute the playbook (the one we used in the last activity): *ansible-playbook --ask-become-pass install_apache.yml*. After executing this command, you may notice that it did not become successful in the 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."sudo



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

TASK [Gathering Facts] ***********************************************************
fatal: [192.168.26.13]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect
to the host via ssh: vboxuser@192.168.26.13: Permission denied (publickey,gssapi-keye
x,gssapi-with-mic,password).", "unreachable": true}
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [update repository index] ***************************************************
changed: [192.168.26.12]
changed: [192.168.26.11]

TASK [Install apache2 package] ***************************************************
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [add PHP support for apache] ************************************************
ok: [192.168.26.11]
ok: [192.168.26.12]

PLAY RECAP ***********************************************************************
192.168.26.11              : ok=4    changed=1    unreachable=0    failed=0    skippe
d=0    rescued=0    ignored=0
192.168.26.12              : ok=4    changed=1    unreachable=0    failed=0    skippe
d=0    rescued=0    ignored=0
192.168.26.13              : ok=0    changed=0    unreachable=1    failed=0    skippe
d=0    rescued=0    ignored=0
```



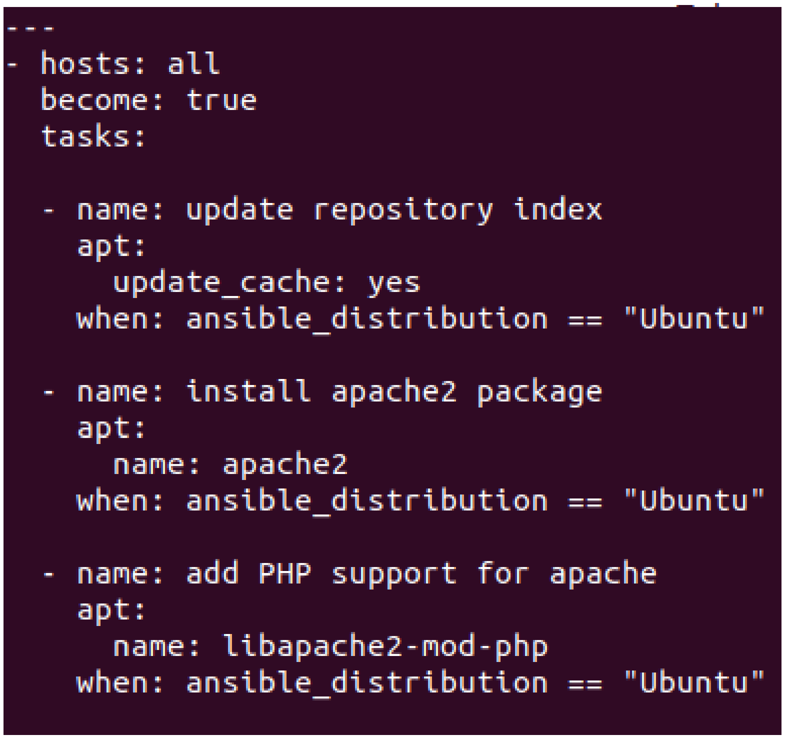
```
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ ansible all -m ping --ask-pas
s
SSH password:
192.168.26.12 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
192.168.26.13 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
192.168.26.11 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```



vboxuser@Workstation: ~/CPE212_REYES_ALEXZANDER-LAPTOP-

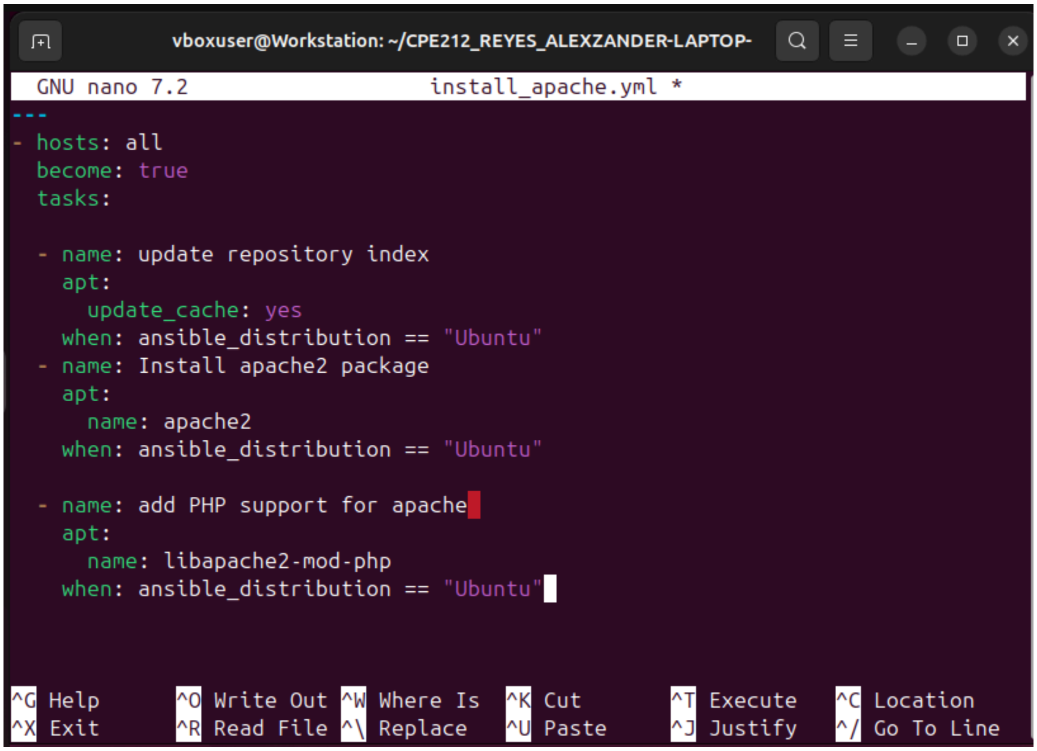
```
  GNU nano 7.2                     /etc/ansible/hosts
[ubuntu]
192.168.26.11 ansible_user=vboxuser
192.168.26.12 ansible_user=vboxuser

[centos]

192.168.26.13 ansible_user=Reyes
```

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"
```

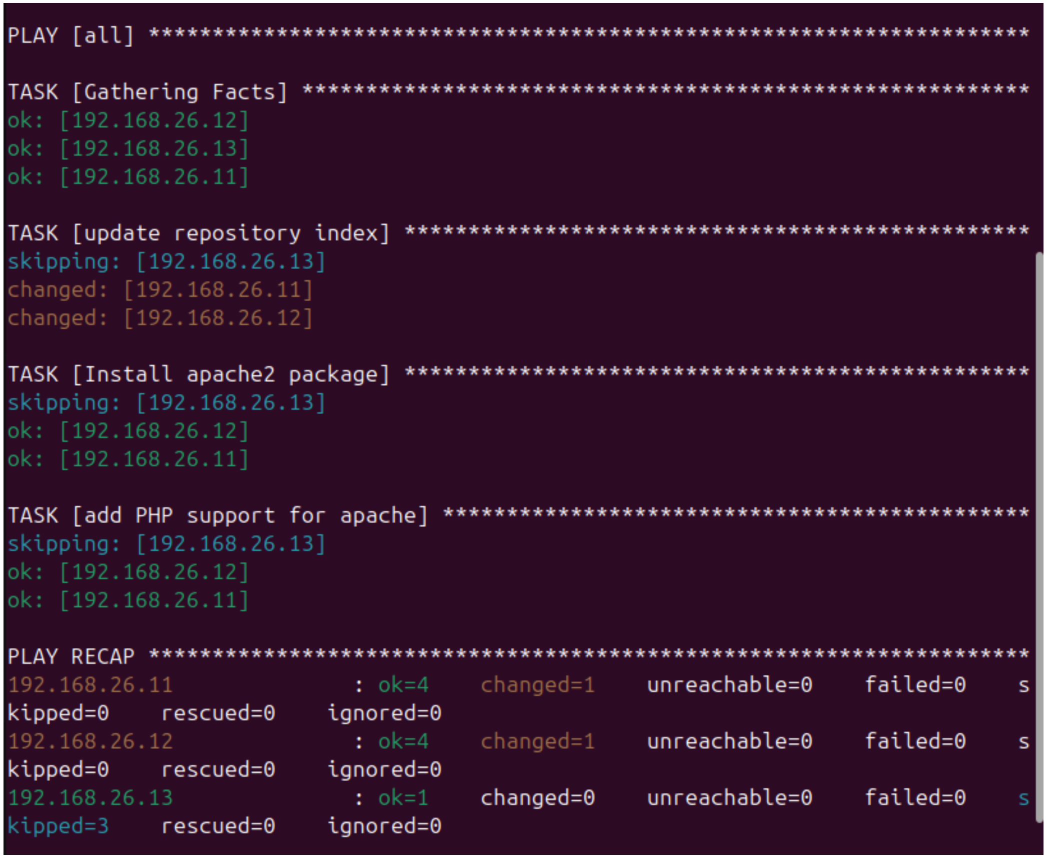
Make sure to save the file and exit.



vboxuser@Workstation: ~/CPE212_REYES_ALEXZANDER-LAPTOP-

```
  GNU nano 7.2                      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"
```

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

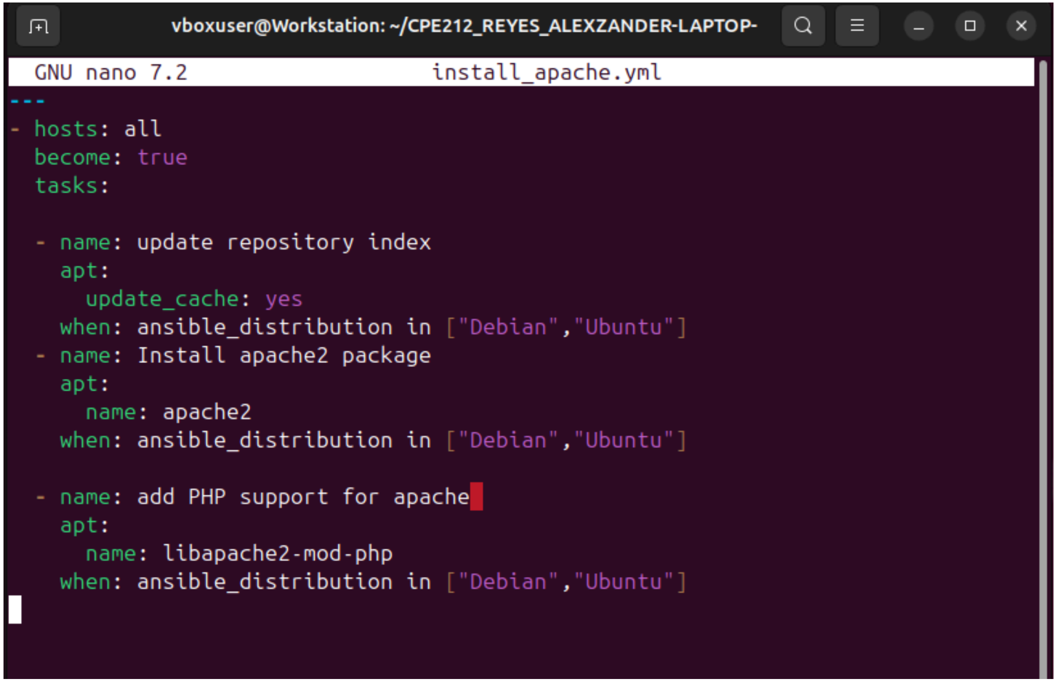


```
PLAY [all] **********************************************************************

TASK [Gathering Facts] **********************************************************
ok: [192.168.26.12]
ok: [192.168.26.13]
ok: [192.168.26.11]

TASK [update repository index] **************************************************
skipping: [192.168.26.13]
changed: [192.168.26.11]
changed: [192.168.26.12]

TASK [Install apache2 package] **************************************************
skipping: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [add PHP support for apache] ***********************************************
skipping: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

PLAY RECAP **********************************************************************
192.168.26.11              : ok=4    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.26.12              : ok=4    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.26.13              : ok=1    changed=0    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
```

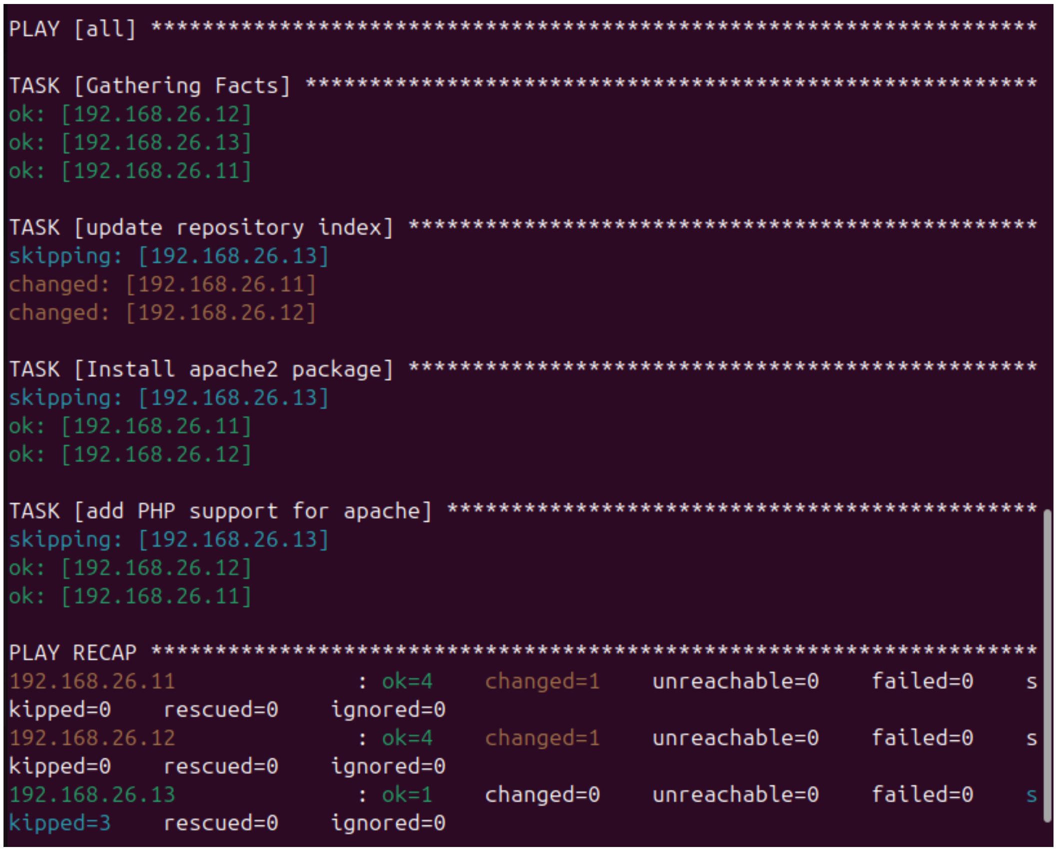
If you have a mix of Debian and Ubuntu servers, you can change the configuration of your playbook like this.

```
-    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.



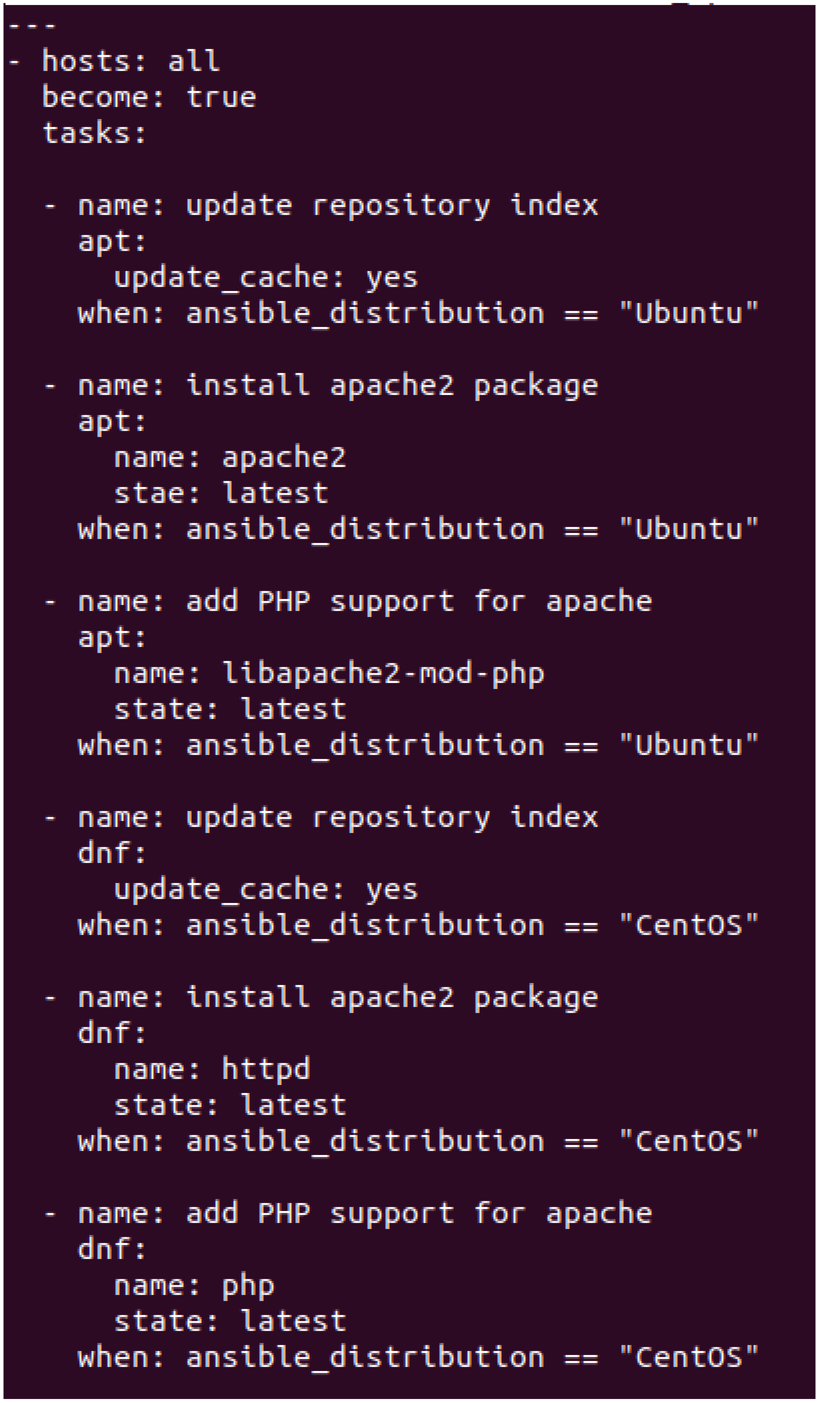
```
  GNU nano 7.2                    install_apache.yml
---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution in ["Debian","Ubuntu"]
  - name: Install apache2 package
    apt:
      name: apache2
    when: ansible_distribution in ["Debian","Ubuntu"]

  - name: add PHP support for apache
    apt:
      name: libapache2-mod-php
    when: ansible_distribution in ["Debian","Ubuntu"]
```



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

TASK [Gathering Facts] **********************************************************
ok: [192.168.26.12]
ok: [192.168.26.13]
ok: [192.168.26.11]

TASK [update repository index] **************************************************
skipping: [192.168.26.13]
changed: [192.168.26.11]
changed: [192.168.26.12]

TASK [Install apache2 package] **************************************************
skipping: [192.168.26.13]
ok: [192.168.26.11]
ok: [192.168.26.12]

TASK [add PHP support for apache] ***********************************************
skipping: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

PLAY RECAP **********************************************************************
192.168.26.11              : ok=4    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.26.12              : ok=4    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.26.13              : ok=1    changed=0    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
```

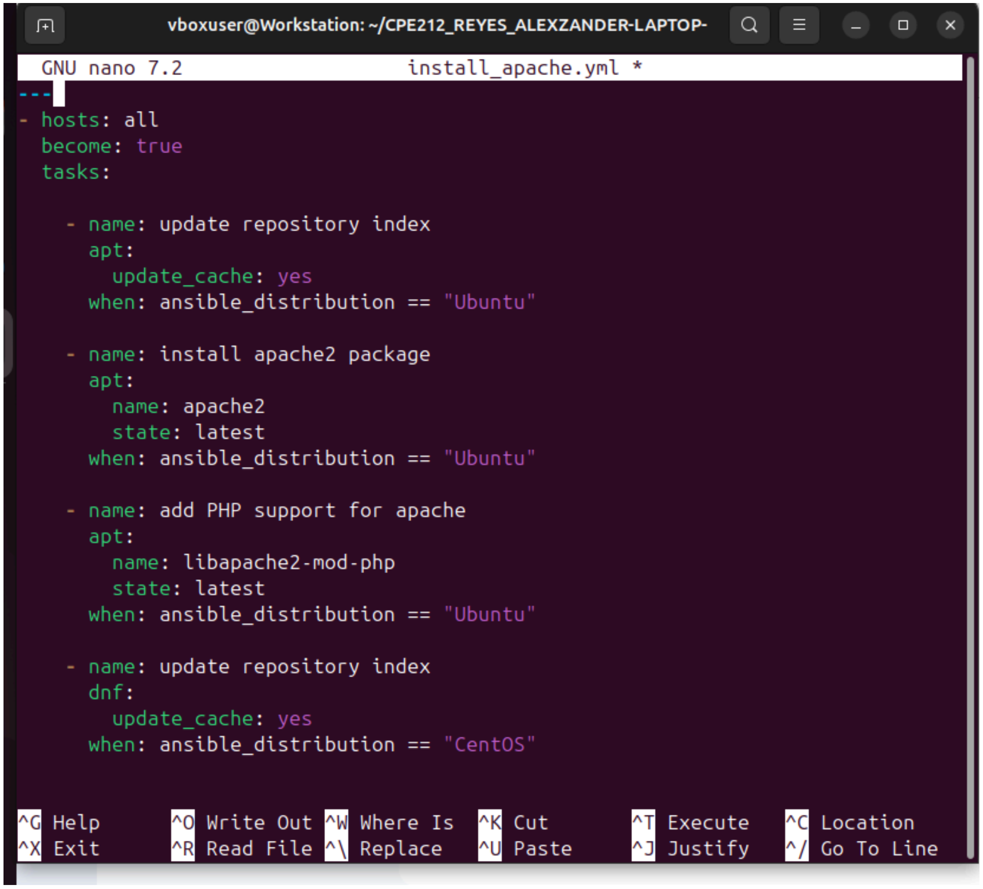
- Ansible will now run the task on both Debian and Ubuntu systems, instead of only Ubuntu.

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

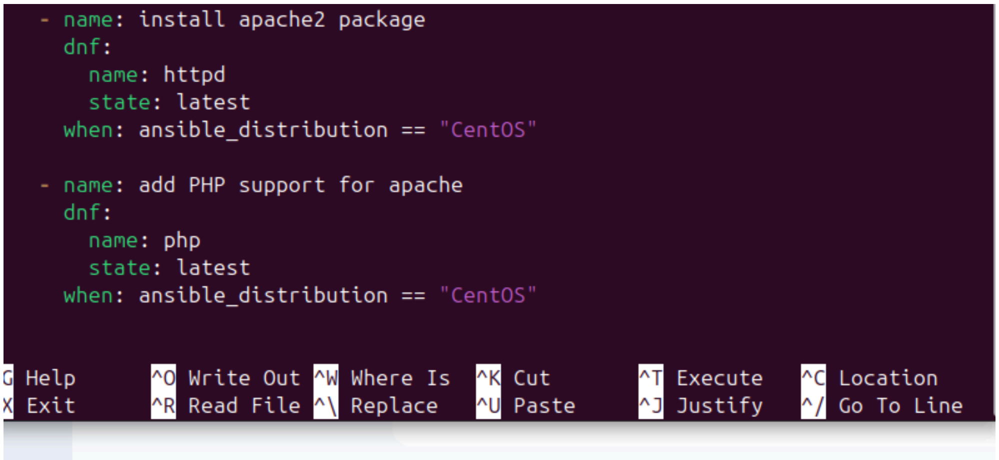


```
---
- hosts: all
  become: true
  tasks:

  - 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.

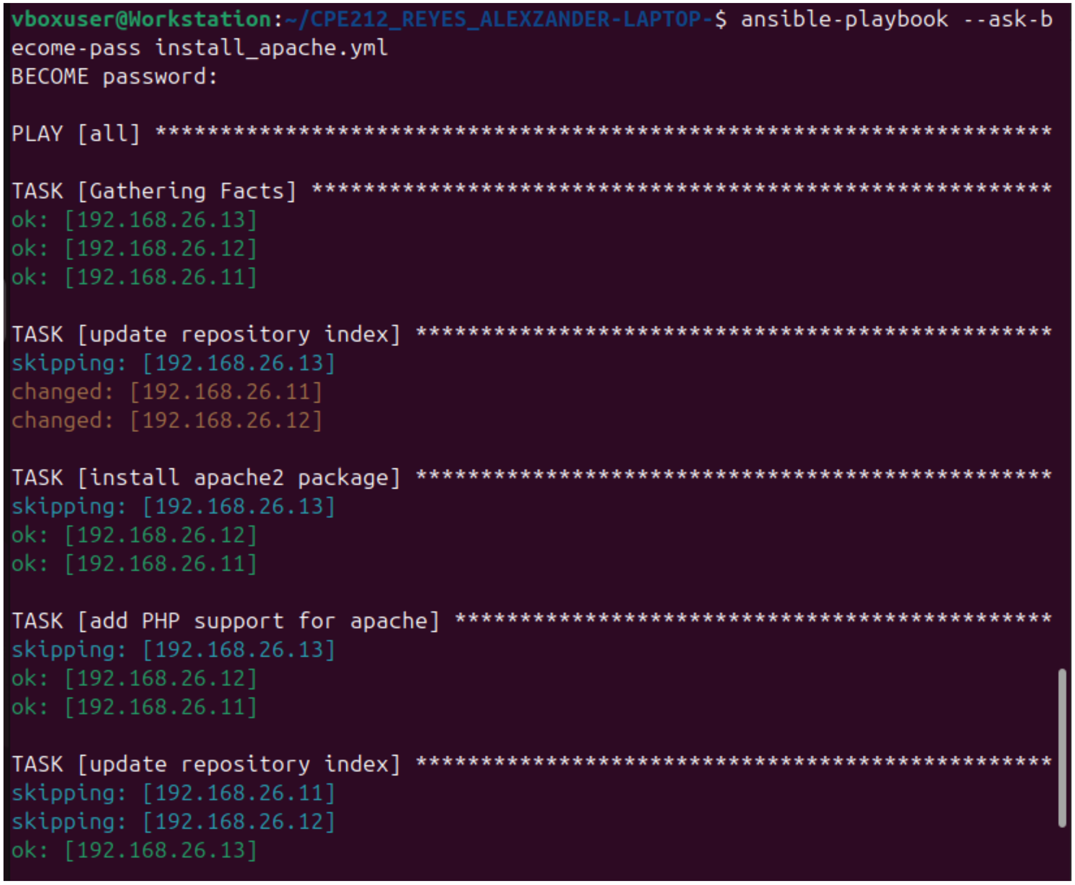


```
---
- hosts: all
  become: true
  tasks:

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

    - name: install apache2 package
      apt:
        name: apache2
        state: 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"
```

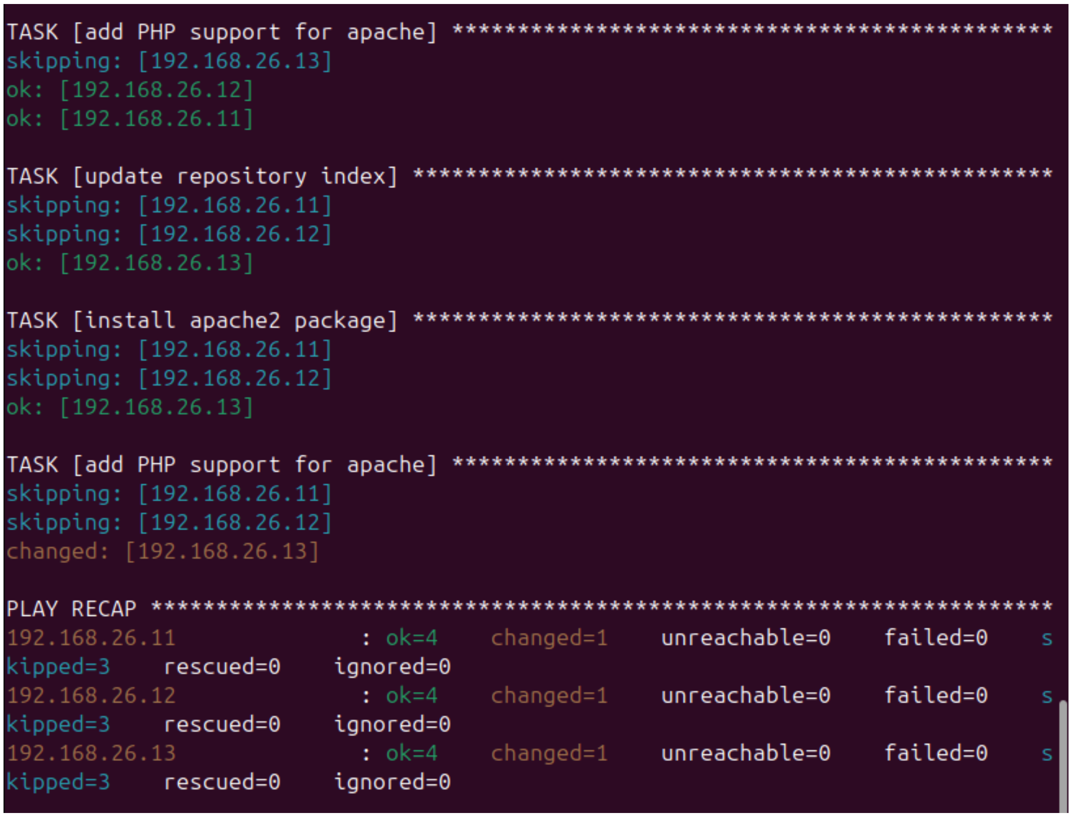


```
  - name: install apache2 package
    dnf:
      name: httpd
      state: latest
    when: ansible_distribution == "CentOS"

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

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



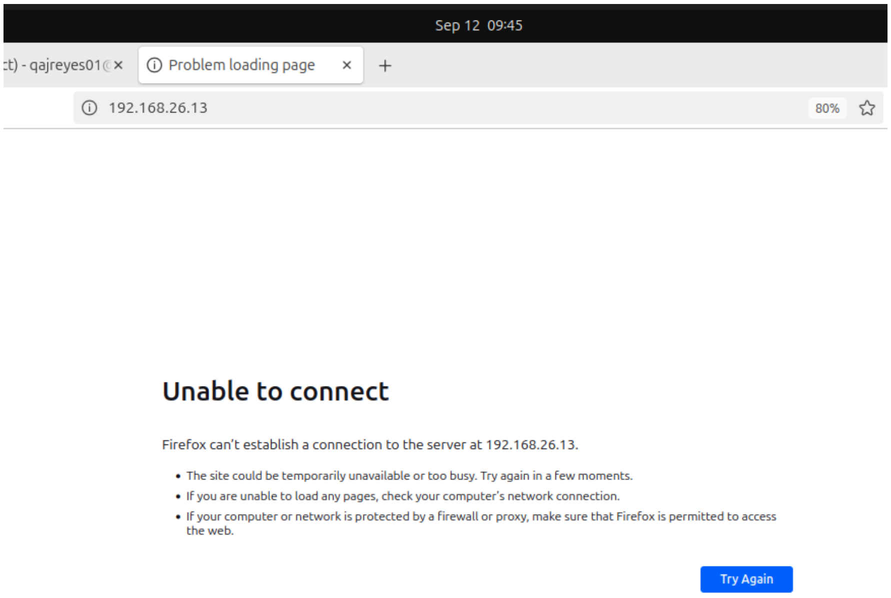
```
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ ansible-playbook --ask-b
ecome-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [update repository index] *************************************************
skipping: [192.168.26.13]
changed: [192.168.26.11]
changed: [192.168.26.12]

TASK [install apache2 package] *************************************************
skipping: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [add PHP support for apache] **********************************************
skipping: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [update repository index] *************************************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
ok: [192.168.26.13]
```



```
TASK [add PHP support for apache] **********************************************
skipping: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [update repository index] *************************************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
ok: [192.168.26.13]

TASK [install apache2 package] *************************************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
ok: [192.168.26.13]

TASK [add PHP support for apache] **********************************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
changed: [192.168.26.13]

PLAY RECAP *********************************************************************
192.168.26.11              : ok=4    changed=1    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
192.168.26.12              : ok=4    changed=1    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
192.168.26.13              : ok=4    changed=1    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
```

- Apache and PHP are installed on both Ubuntu and CentOS hosts.
- Tasks are executed conditionally depending on the system's OS.

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.

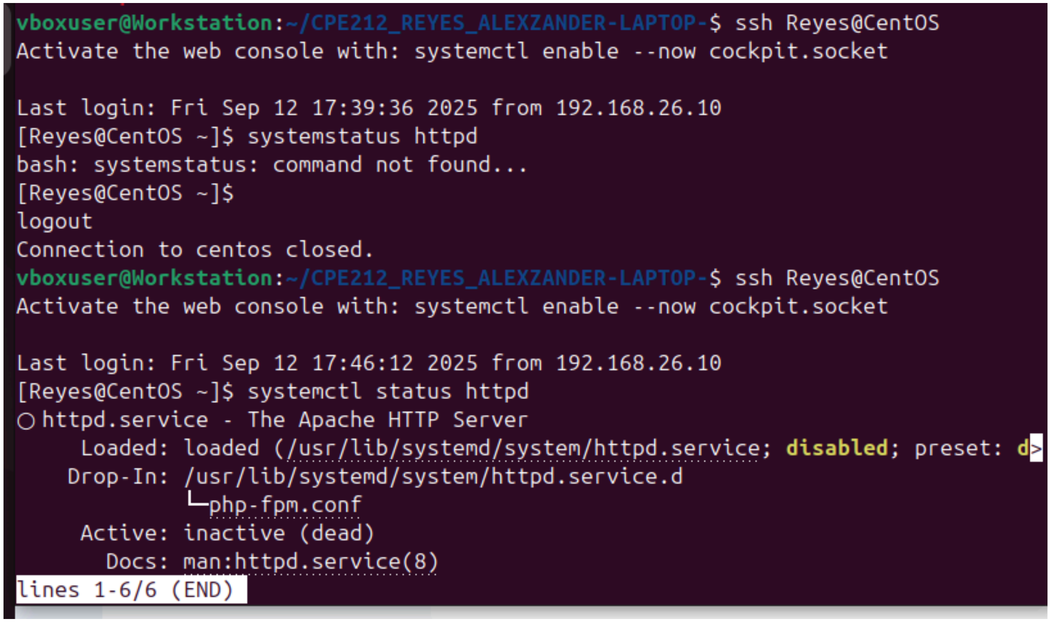


Unable to connect

Firefox can't establish a connection to the server at 192.168.26.13.

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the web.

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.



```
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ ssh Reyes@CentOS
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Fri Sep 12 17:39:36 2025 from 192.168.26.10
[Reyes@CentOS ~]$ systemstatus httpd
bash: systemstatus: command not found...
[Reyes@CentOS ~]$
logout
Connection to centos closed.
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ ssh Reyes@CentOS
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Fri Sep 12 17:46:12 2025 from 192.168.26.10
[Reyes@CentOS ~]$ systemctl status httpd
○ httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: d>
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: inactive (dead)
       Docs: man:httpd.service(8)
lines 1-6/6 (END)
```

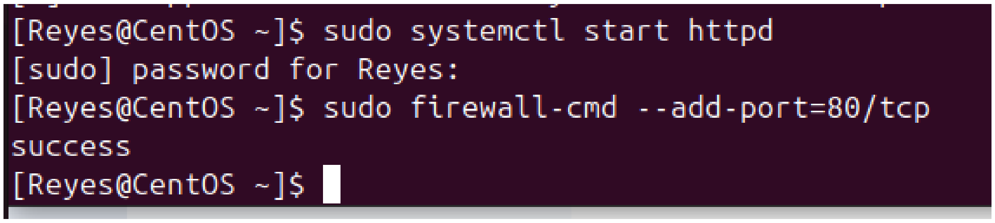
5.2 Issue the following command to start the service:

sudo systemctl start httpd

(When prompted, enter the sudo password)

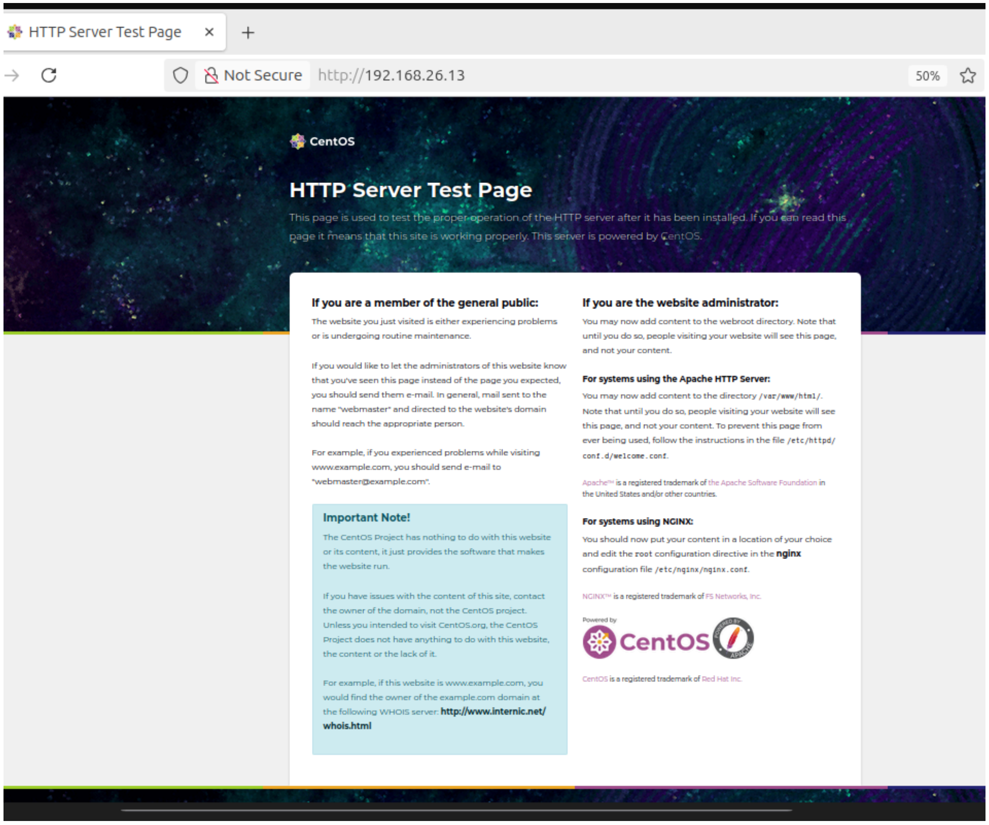
sudo firewall-cmd --add-port=80/tcp

(The result should be a success)



```
[Reyes@CentOS ~]$ sudo systemctl start httpd
[sudo] password for Reyes:
[Reyes@CentOS ~]$ sudo firewall-cmd --add-port=80/tcp
success
[Reyes@CentOS ~]$
```

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)



HTTP Server Test Page

This page is used to test the proper operation of the HTTP server after it has been installed. If you can read this page it means that this site is working properly. This server is powered by CentOS.

If you are a member of the general public:

The website you just visited is either experiencing problems or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

Important Note!

The CentOS Project has nothing to do with this website or its content, it just provides the software that makes the website run.

If you have issues with the content of this site, contact the owner of the domain, not the CentOS project. Unless you intended to visit CentOS.org, the CentOS Project does not have anything to do with this website, the content or the lack of it.

For example, if this website is www.example.com, you would find the owner of the example.com domain at the following WHOIS server: **http://www.internic.net/whois.html**

If you are the website administrator:

You may now add content to the webroot directory. Note that until you do so, people visiting your website will see this page, and not your content.

For systems using the Apache HTTP Server:

You may now add content to the directory `/var/www/html/`. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

Apache™ is a registered trademark of the Apache Software Foundation in the United States and/or other countries.

For systems using NGINX:

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX™ is a registered trademark of F5 Networks, Inc.

Powered by CentOS

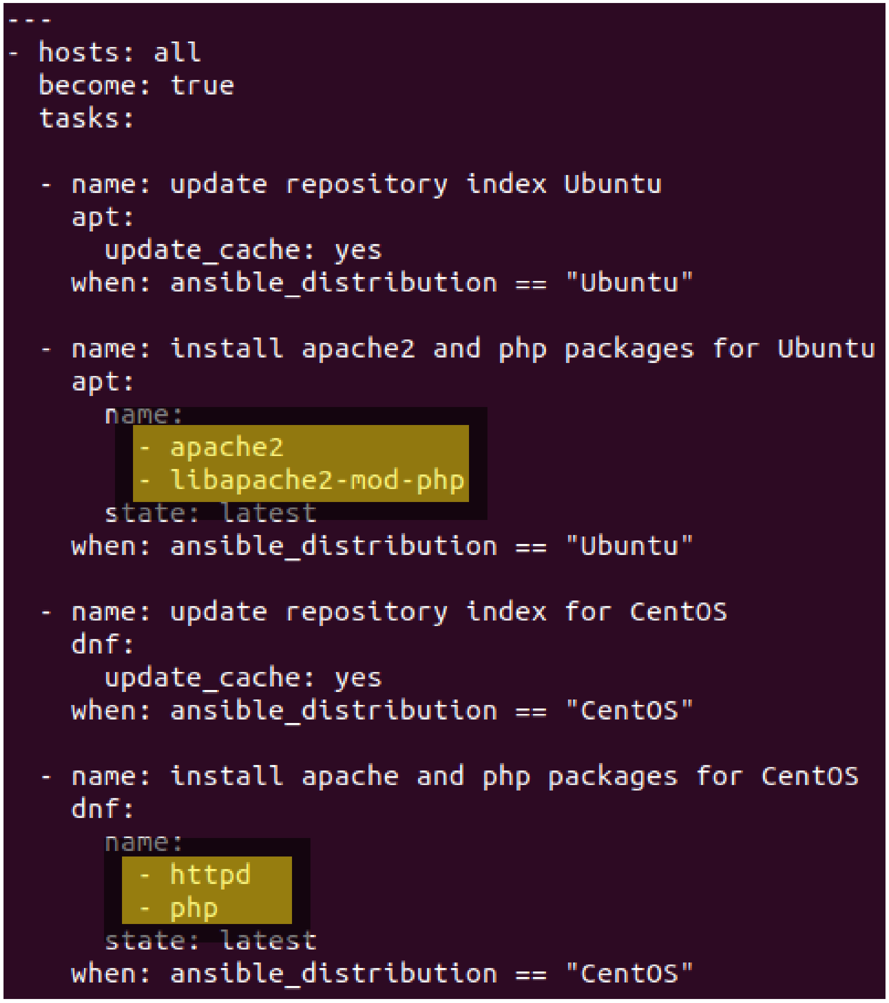
CentOS is a registered trademark of Red Hat Inc.

- It was successful.

Task 2: Refactoring playbook

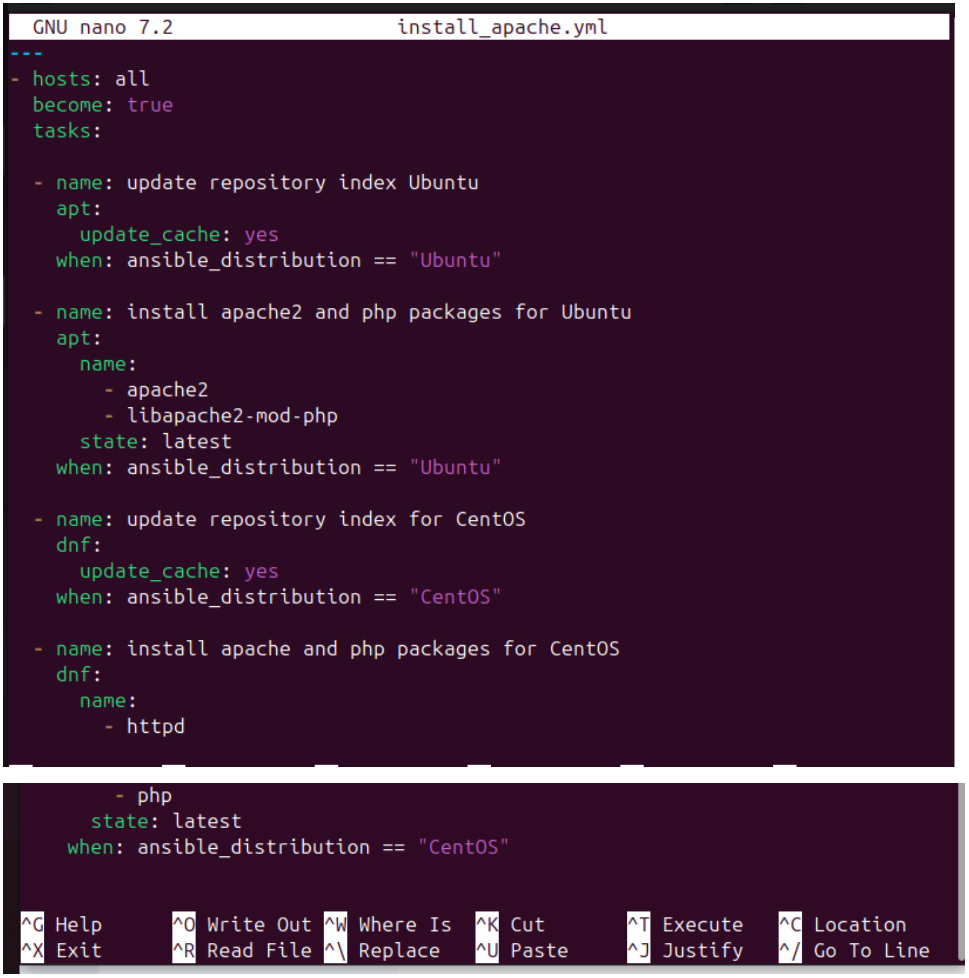
This time, we want to make sure that our playbook is efficient and that the codes are easier to read. This will also makes run ansible more quickly if it has to execute fewer tasks to do the same thing.

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:



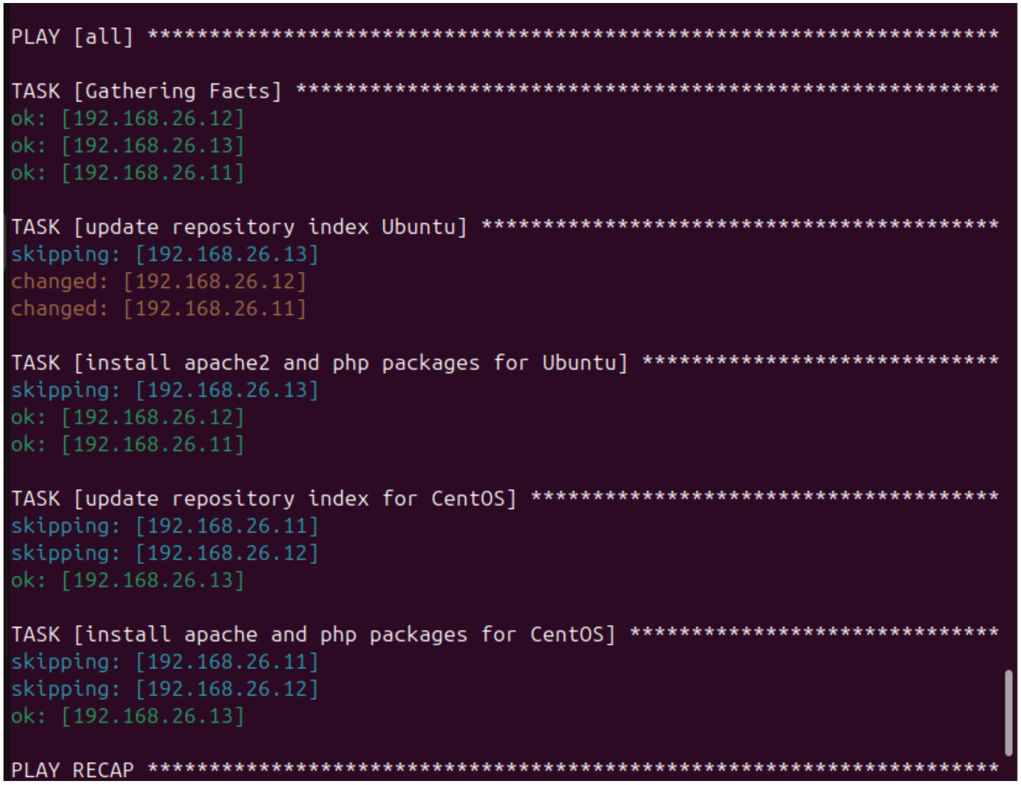
```
---
- hosts: all
  become: true
  tasks:

  - 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.

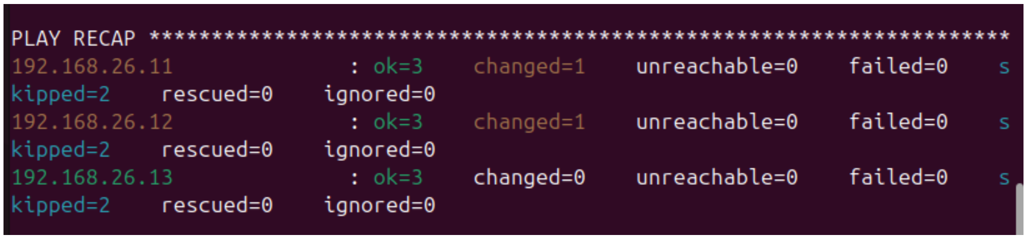


```
---
- hosts: all
  become: true
  tasks:

  - 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"
```

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



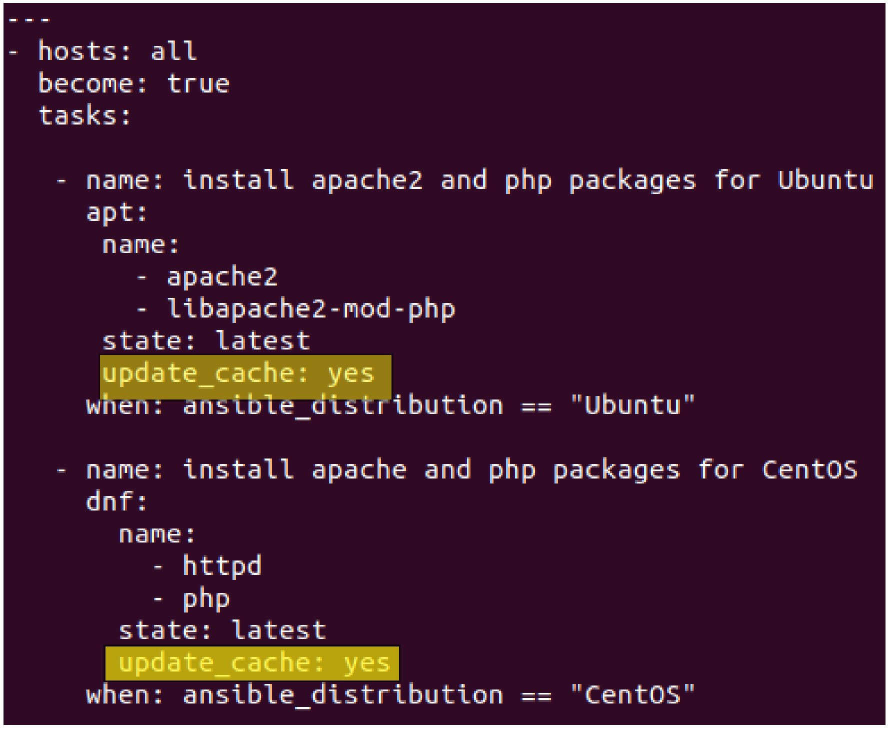
```
PLAY [all] ***********************************************************************

TASK [Gathering Facts] ***********************************************************
ok: [192.168.26.12]
ok: [192.168.26.13]
ok: [192.168.26.11]

TASK [update repository index Ubuntu] ********************************************
skipping: [192.168.26.13]
changed: [192.168.26.12]
changed: [192.168.26.11]

TASK [install apache2 and php packages for Ubuntu] *******************************
skipping: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [update repository index for CentOS] ****************************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
ok: [192.168.26.13]

TASK [install apache and php packages for CentOS] ********************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
ok: [192.168.26.13]

PLAY RECAP ***********************************************************************
```



```
PLAY RECAP ***********************************************************************
192.168.26.11              : ok=3    changed=1    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.26.12              : ok=3    changed=1    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.26.13              : ok=3    changed=0    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
```

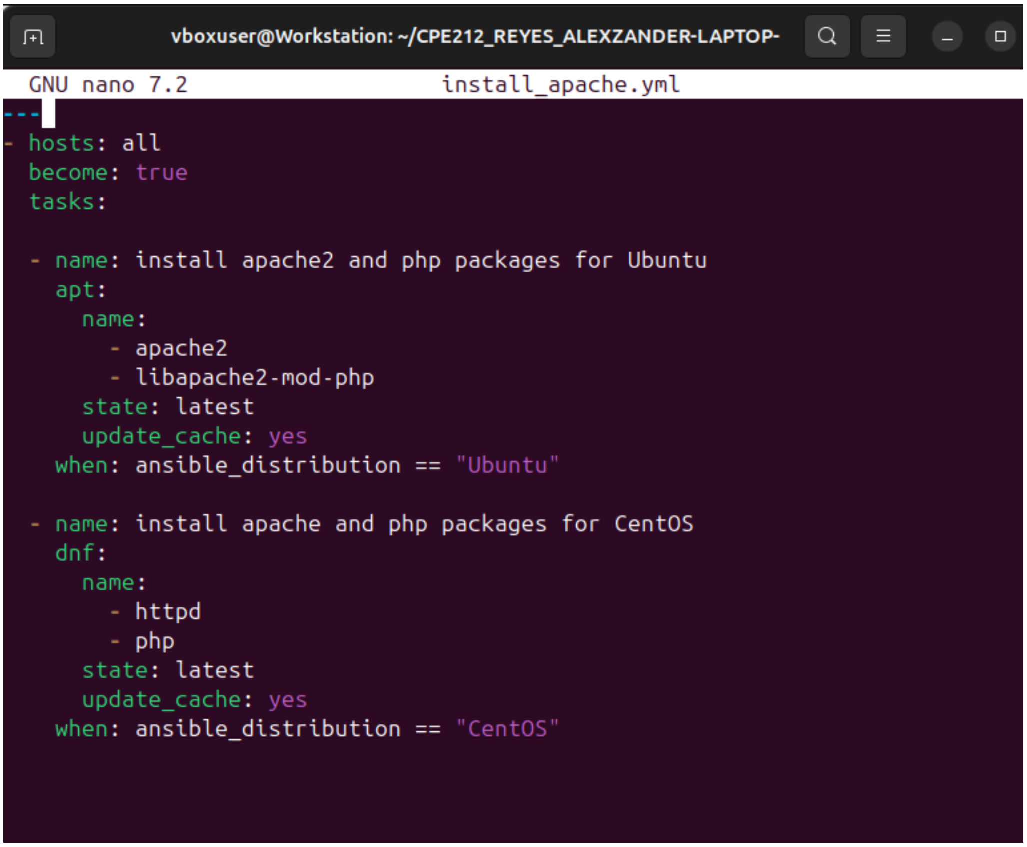
- It then identified the operating system of each target host and conditionally executed tasks based on whether it was Ubuntu or CentOS. For Ubuntu, it updated the APT repository and installed apache2 and libapache2-mod-php; for CentOS, it updated the DNF repository and installed httpd and php. Tasks not relevant to the host's OS were skipped, and the output displayed which tasks were changed, skipped, or already up-to-date, confirming that Apache and PHP were successfully installed based on the system type.

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:



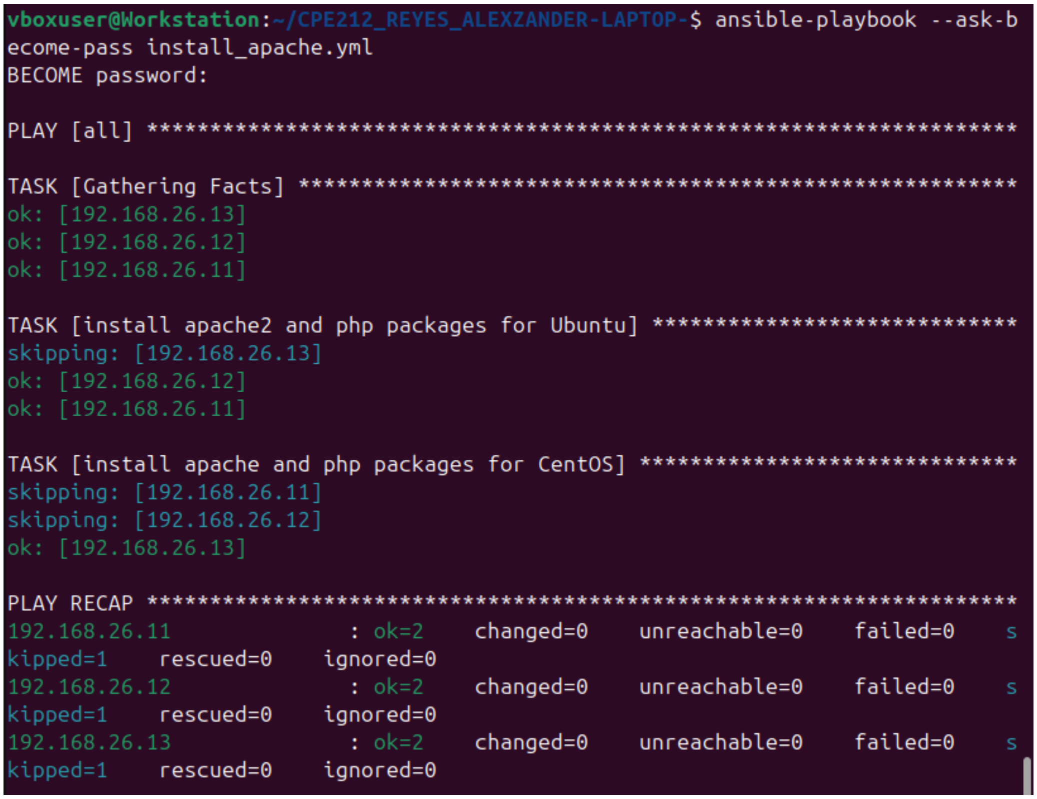
```
---
- hosts: all
  become: true
  tasks:

   - 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.



```
GNU nano 7.2                      install_apache.yml
---
- 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"
```

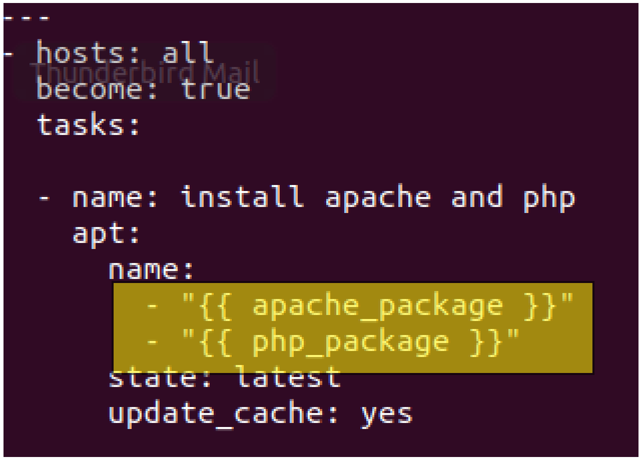
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



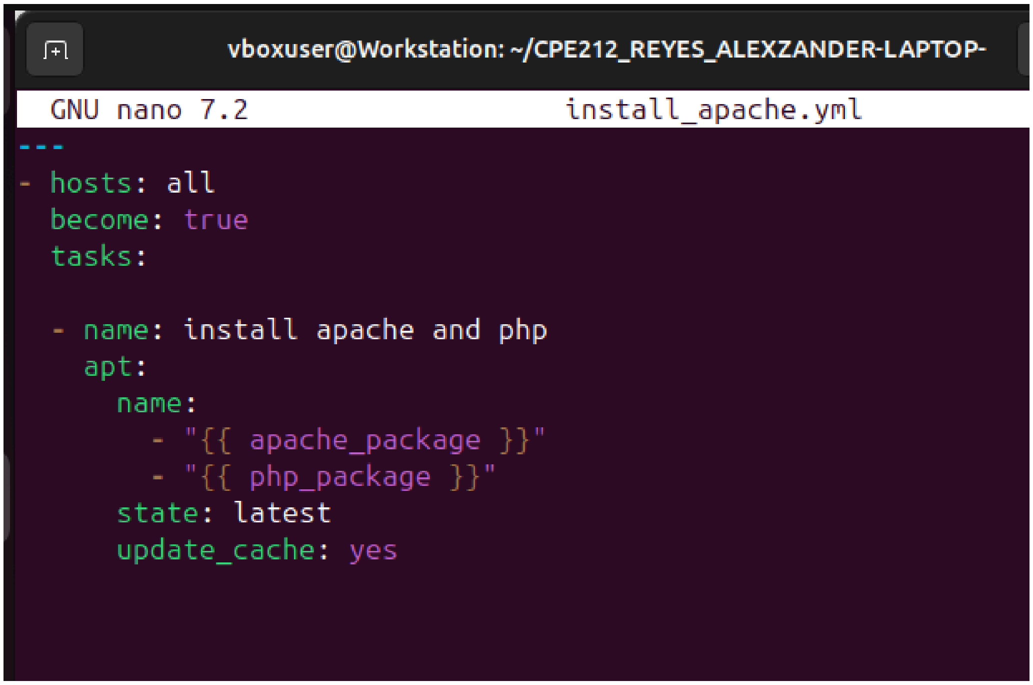
```
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ ansible-playbook --ask-b
ecome-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [install apache2 and php packages for Ubuntu] *****************************
skipping: [192.168.26.13]
ok: [192.168.26.12]
ok: [192.168.26.11]

TASK [install apache and php packages for CentOS] ******************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
ok: [192.168.26.13]

PLAY RECAP *********************************************************************
192.168.26.11              : ok=2    changed=0    unreachable=0    failed=0     s
kipped=1    rescued=0    ignored=0
192.168.26.12              : ok=2    changed=0    unreachable=0    failed=0     s
kipped=1    rescued=0    ignored=0
192.168.26.13              : ok=2    changed=0    unreachable=0    failed=0     s
kipped=1    rescued=0    ignored=0
```

- If the host is running Ubuntu, it updates the APT package cache and installs apache2 and libapache2-mod-php; if the host is running CentOS, it updates the DNF cache and installs httpd and php. Tasks not matching the system's OS are skipped.

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.



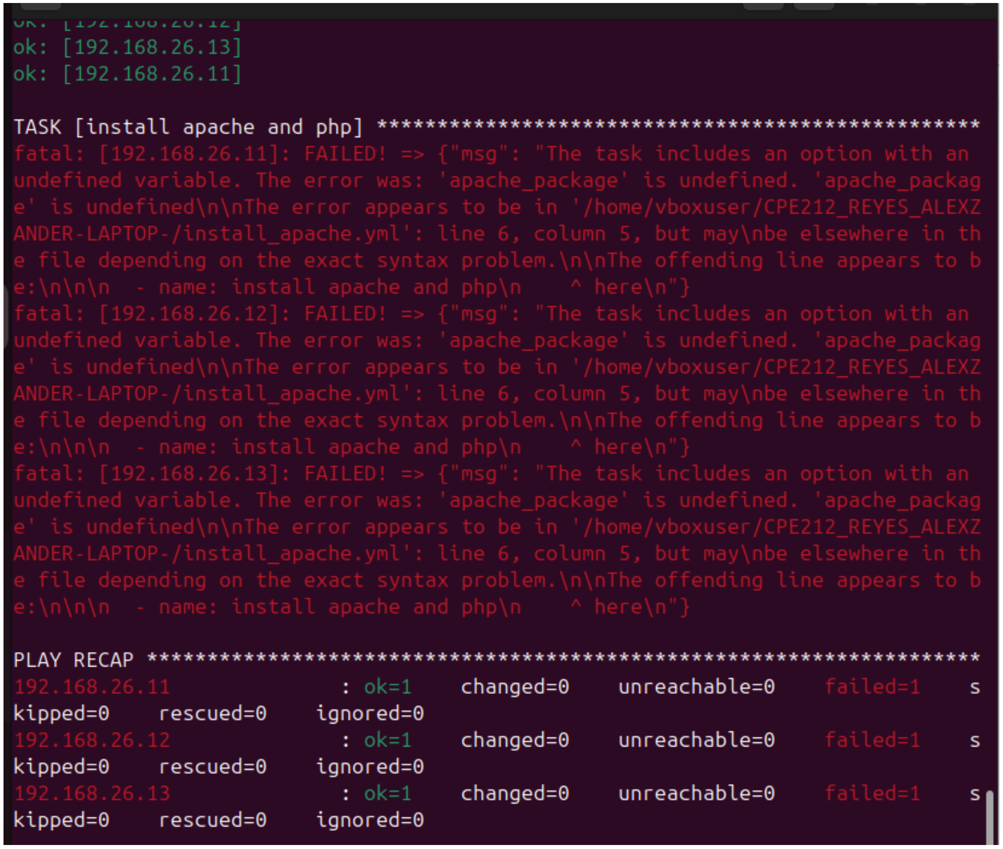
```
---
- hosts: all
  become: true
  tasks:

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



vboxuser@Workstation: ~/CPE212_REYES_ALEXZANDER-LAPTOP-

```
  GNU nano 7.2                               install_apache.yml
---
- hosts: all
  become: true
  tasks:

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

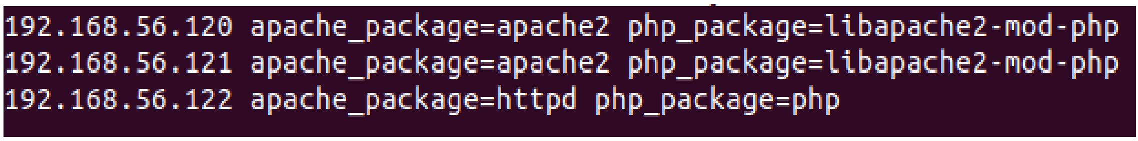
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
ok: [192.168.26.12]
ok: [192.168.26.13]
ok: [192.168.26.11]

TASK [install apache and php] *****************************************************
fatal: [192.168.26.11]: FAILED! => {"msg": "The task includes an option with an
undefined variable. The error was: 'apache_package' is undefined. 'apache_packag
e' is undefined\n\nThe error appears to be in '/home/vboxuser/CPE212_REYES_ALEXZ
ANDER-LAPTOP-/install_apache.yml': line 6, column 5, but may\nbe elsewhere in th
e file depending on the exact syntax problem.\n\nThe offending line appears to b
e:\n\n\n  - name: install apache and php\n    ^ here\n"}
fatal: [192.168.26.12]: FAILED! => {"msg": "The task includes an option with an
undefined variable. The error was: 'apache_package' is undefined. 'apache_packag
e' is undefined\n\nThe error appears to be in '/home/vboxuser/CPE212_REYES_ALEXZ
ANDER-LAPTOP-/install_apache.yml': line 6, column 5, but may\nbe elsewhere in th
e file depending on the exact syntax problem.\n\nThe offending line appears to b
e:\n\n\n  - name: install apache and php\n    ^ here\n"}
fatal: [192.168.26.13]: FAILED! => {"msg": "The task includes an option with an
undefined variable. The error was: 'apache_package' is undefined. 'apache_packag
e' is undefined\n\nThe error appears to be in '/home/vboxuser/CPE212_REYES_ALEXZ
ANDER-LAPTOP-/install_apache.yml': line 6, column 5, but may\nbe elsewhere in th
e file depending on the exact syntax problem.\n\nThe offending line appears to b
e:\n\n\n  - name: install apache and php\n    ^ here\n"}

PLAY RECAP ************************************************************************
192.168.26.11              : ok=1    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
192.168.26.12              : ok=1    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
192.168.26.13              : ok=1    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
```

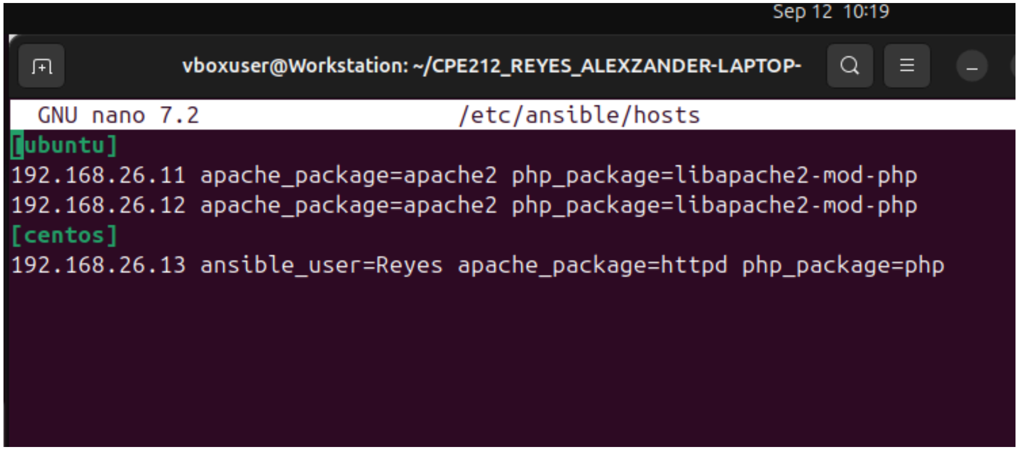
- he variables apache_package and php_package weren't defined, so Ansible didn't know which packages to install. You need to define these variables to fix the error.

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
```

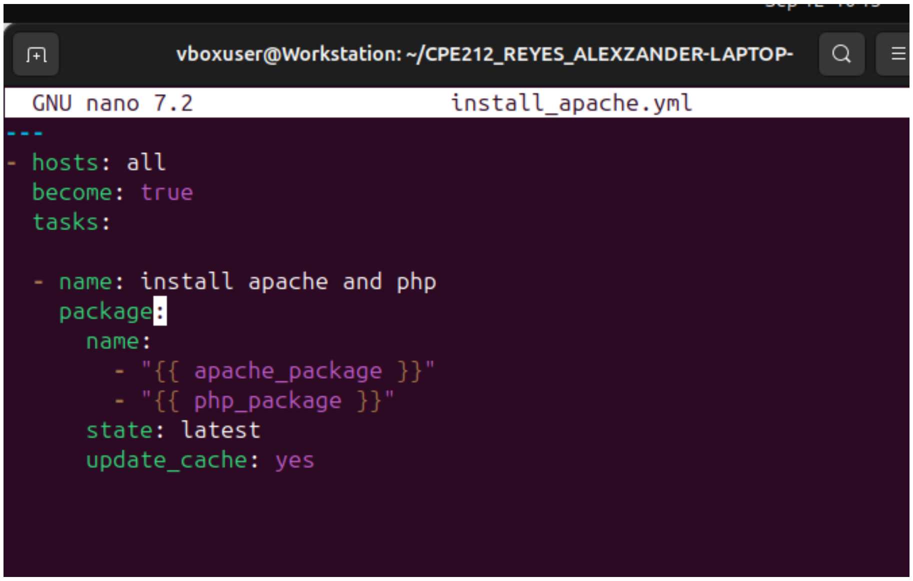
Make sure to save the *inventory* file and exit.



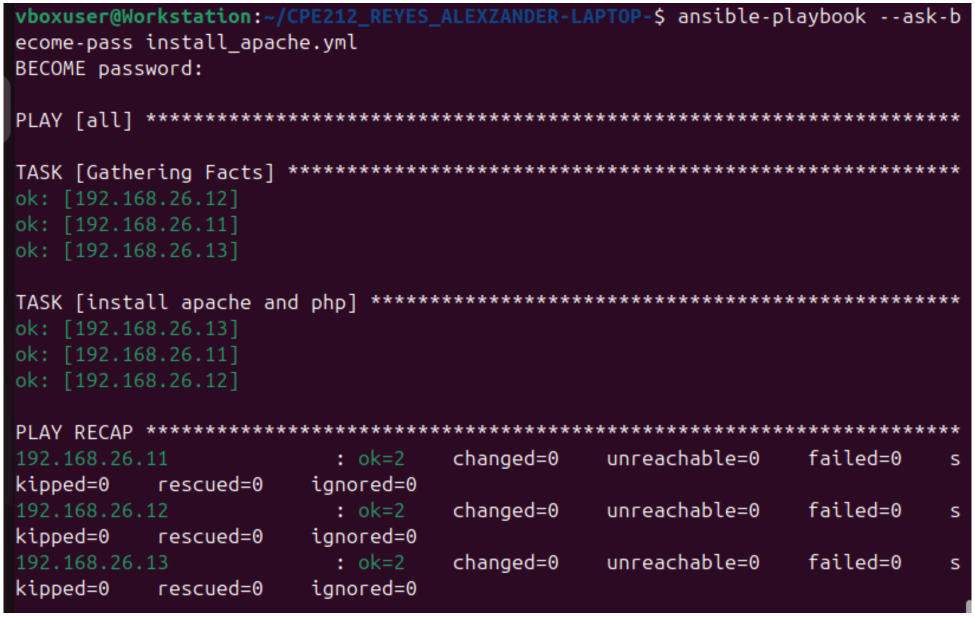
```
  GNU nano 7.2                    /etc/ansible/hosts
[ubuntu]
192.168.26.11 apache_package=apache2 php_package=libapache2-mod-php
192.168.26.12 apache_package=apache2 php_package=libapache2-mod-php
[centos]
192.168.26.13 ansible_user=Reyes apache_package=httpd php_package=php
```

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.



```
  GNU nano 7.2                    install_apache.yml
---
- hosts: all
  become: true
  tasks:

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



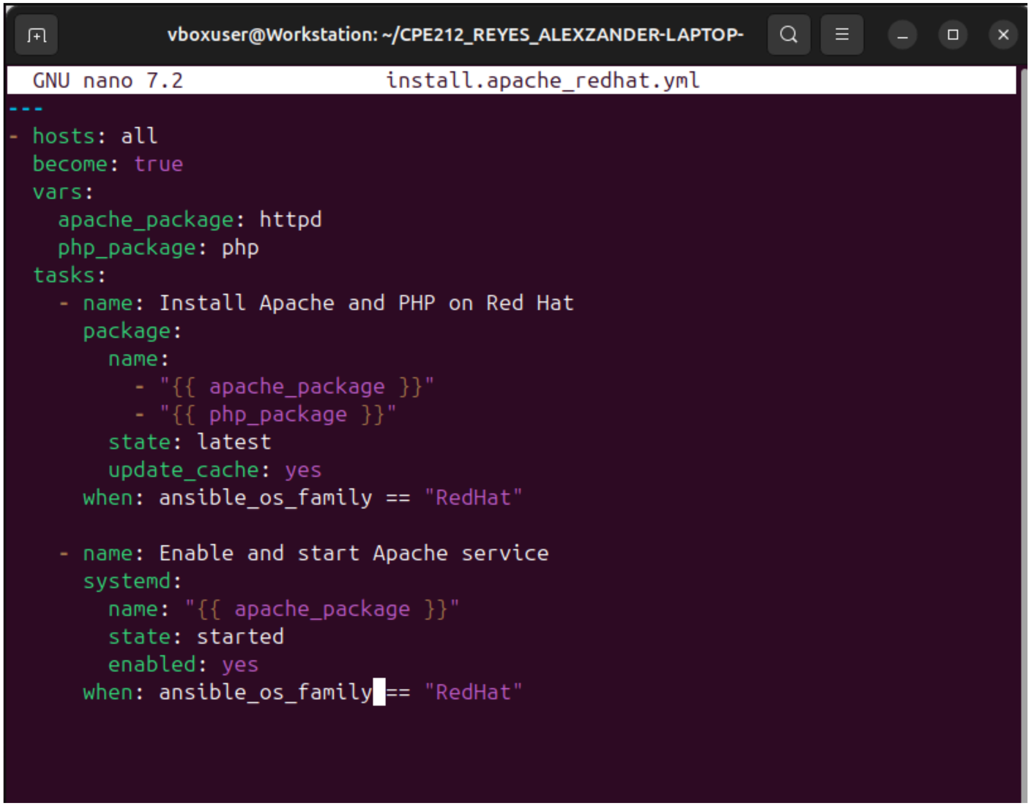
```
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ ansible-playbook --ask-b
ecome-pass install_apache.yml
BECOME password:

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

TASK [Gathering Facts] **********************************************************
ok: [192.168.26.12]
ok: [192.168.26.11]
ok: [192.168.26.13]

TASK [install apache and php] ***************************************************
ok: [192.168.26.13]
ok: [192.168.26.11]
ok: [192.168.26.12]

PLAY RECAP **********************************************************************
192.168.26.11              : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.26.12              : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.26.13              : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
```

- It successfully gathered facts from all target hosts (192.168.26.11, 192.168.26.12, 192.168.26.13). Then, using the generic package module, it installed the Apache and PHP packages on each host according to their respective OS package managers without any errors. The playbook completed successfully with no failed tasks, indicating that Apache and PHP were installed or already present on all the hosts.

Supplementary Activity:

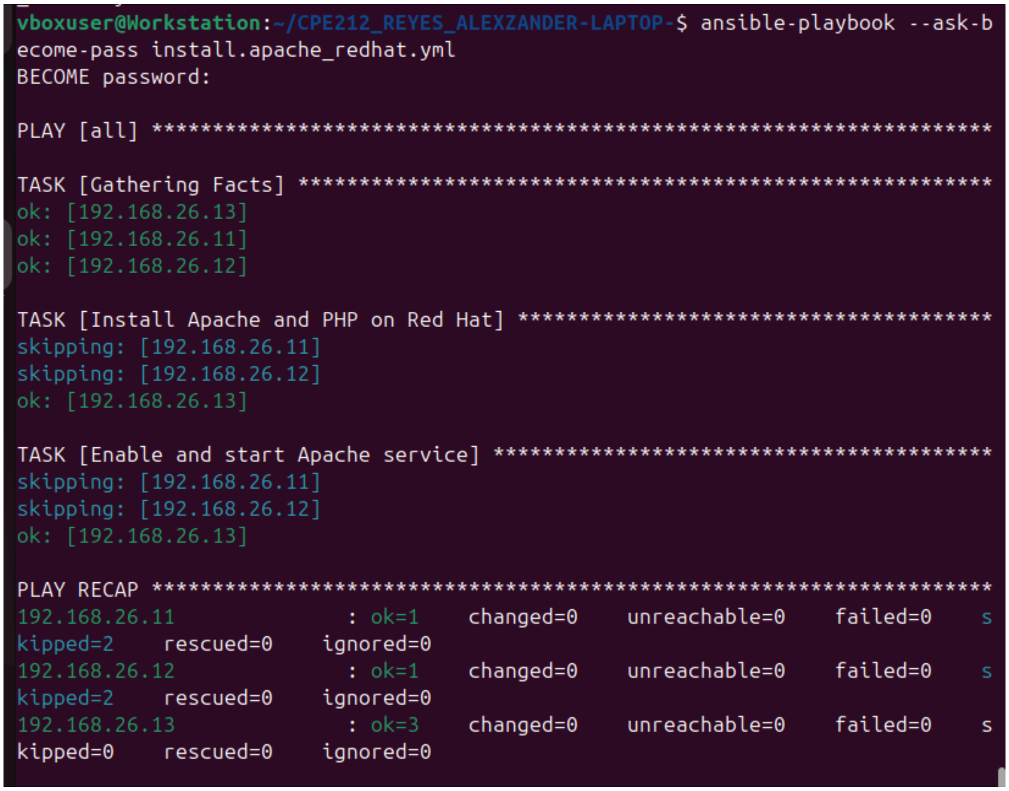
1. Create a playbook that could do the previous tasks in Red Hat OS.

   Playbook:

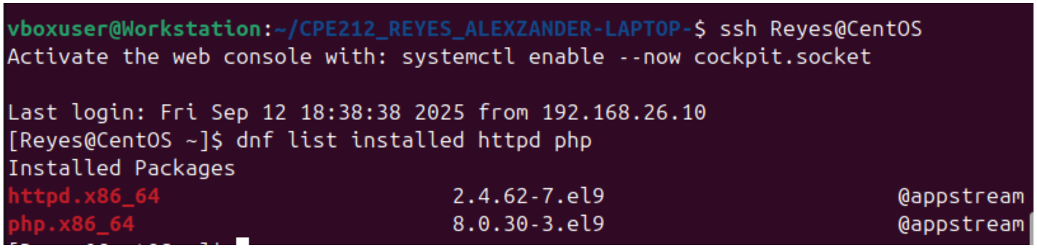


```
GNU nano 7.2                    install.apache_redhat.yml
---
- hosts: all
  become: true
  vars:
    apache_package: httpd
    php_package: php
  tasks:
    - name: Install Apache and PHP on Red Hat
      package:
        name:
          - "{{ apache_package }}"
          - "{{ php_package }}"
        state: latest
        update_cache: yes
      when: ansible_os_family == "RedHat"

    - name: Enable and start Apache service
      systemd:
        name: "{{ apache_package }}"
        state: started
        enabled: yes
      when: ansible_os_family == "RedHat"
```

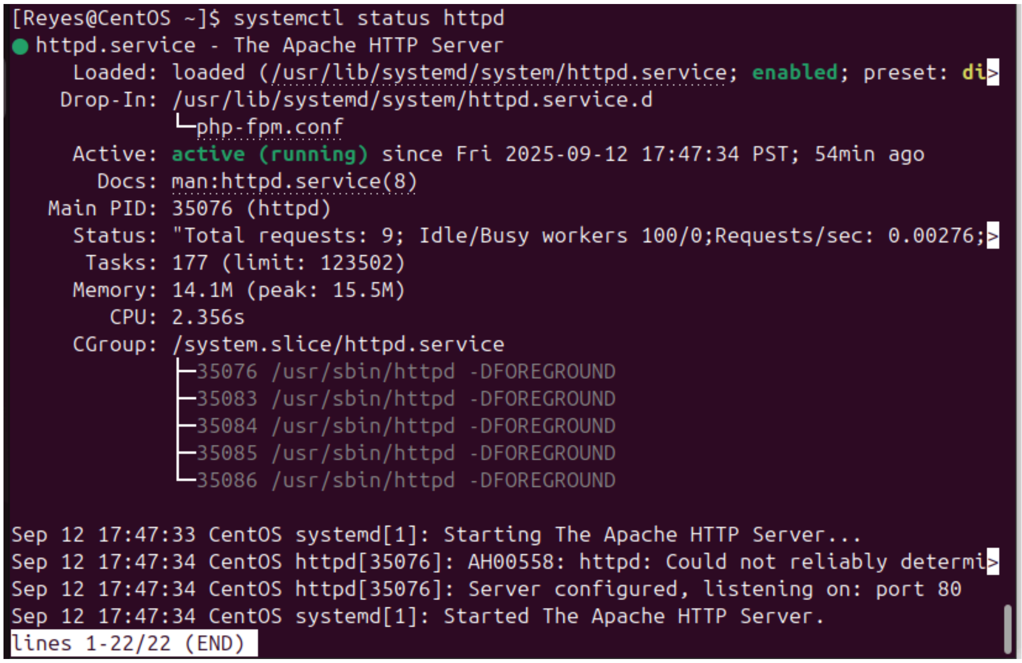
Result:



```
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ ansible-playbook --ask-b
ecome-pass install.apache_redhat.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.26.13]
ok: [192.168.26.11]
ok: [192.168.26.12]

TASK [Install Apache and PHP on Red Hat] ***************************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
ok: [192.168.26.13]

TASK [Enable and start Apache service] *****************************************
skipping: [192.168.26.11]
skipping: [192.168.26.12]
ok: [192.168.26.13]

PLAY RECAP *********************************************************************
192.168.26.11              : ok=1    changed=0    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.26.12              : ok=1    changed=0    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.26.13              : ok=3    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
```



```
vboxuser@Workstation:~/CPE212_REYES_ALEXZANDER-LAPTOP-$ ssh Reyes@CentOS
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Fri Sep 12 18:38:38 2025 from 192.168.26.10
[Reyes@CentOS ~]$ dnf list installed httpd php
Installed Packages
httpd.x86_64                       2.4.62-7.el9                       @appstream
php.x86_64                         8.0.30-3.el9                       @appstream
```



```
[Reyes@CentOS ~]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: di>
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: active (running) since Fri 2025-09-12 17:47:34 PST; 54min ago
       Docs: man:httpd.service(8)
   Main PID: 35076 (httpd)
     Status: "Total requests: 9; Idle/Busy workers 100/0;Requests/sec: 0.00276;>
      Tasks: 177 (limit: 123502)
     Memory: 14.1M (peak: 15.5M)
        CPU: 2.356s
     CGroup: /system.slice/httpd.service
             ├─35076 /usr/sbin/httpd -DFOREGROUND
             ├─35083 /usr/sbin/httpd -DFOREGROUND
             ├─35084 /usr/sbin/httpd -DFOREGROUND
             ├─35085 /usr/sbin/httpd -DFOREGROUND
             └─35086 /usr/sbin/httpd -DFOREGROUND

Sep 12 17:47:33 CentOS systemd[1]: Starting The Apache HTTP Server...
Sep 12 17:47:34 CentOS httpd[35076]: AH00558: httpd: Could not reliably determi>
Sep 12 17:47:34 CentOS httpd[35076]: Server configured, listening on: port 80
Sep 12 17:47:34 CentOS systemd[1]: Started The Apache HTTP Server.
lines 1-22/22 (END)
```

Analysis:

- The playbook only affected the Red Hat machine.
- Ubuntu machines were left untouched, as required.
- The use of ansible_os_family is correct for detecting Red Hat family systems.

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?
   - Writing playbooks in Ansible is relevant as it enables maintaining software well. It has prevented duplication, decreasing the number of errors and increasing the ease of updating or scaling as the infrastructure expands. Well-organized, clean playbooks can be debugged, learned and worked on more easily, in teams. Good practices like use of variables, roles and conditionals are also kept and thus more efficient and reliable automation is ensured by refactoring.

2. When do we use the "when" command in the playbook?

- Ansible playbook uses when command to establish task conditions to be followed. It will only perform a job as long as a given condition is met. This comes in handy, more particularly where you are aiming at different operating systems, providing logic specific to the host or even checking the value of a variable. So, say when: ansible_os_family = some and you could run a task only on Red Hat-based systems or when: my-var = some and you could run a task based on variable input. It aids in increasing the flexibility, efficiency and ease with which playbooks can adapt to various environments.