Name: Planta, Calvin Earl L.	**Date Performed: Sep 3, 2025**
Course/Section: CPE 212 - CPE31S2	**Date Submitted: Sep 5, 2025**
Instructor: Engr. Robin Valenzuela	**Semester and SY: 1st Sem SY 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 issued this command. Did something happen? Why?

```
vbearl@workstation:~/CPE212_Planta$ git pull
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 7 (delta 1), reused 6 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (7/7), 787 bytes | 787.00 KiB/s, done.
From github.com:Calvin-Earl/CPE212_Planta
   c844df1..53a64e8  main       -> origin/main
Updating c844df1..53a64e8
error: The following untracked working tree files would be overwritten by merge:
        ansible.cfg
        install_apache.yml
        inventory.yaml
Please move or remove them before you merge.
Aborting
```

- The git pull command unpacks all the files that we have previously pushed in our github

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."

```
TASK [install apache2 package] ************************************************
changed: [192.168.56.108]

TASK [add PHP support for apache2] ********************************************
changed: [192.168.56.108]

PLAY RECAP ********************************************************************
192.168.56.106             : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.107             : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.108             : ok=4    changed=3    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.117             : ok=1    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
```

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.

```
 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 apache2
      apt:
        name: libapache2-mod-php
      when: ansible_distribution == "Ubuntu"
```

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

```
vbearl@workstation:~/CPE212_Planta$ ansible-playbook --ask-become-pass install_a
pache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]
fatal: [192.168.56.106]: UNREACHABLE! => {"changed": false, "msg": "Failed to co
nnect to the host via ssh: ssh: connect to host 192.168.56.106 port 22: No route
 to host", "unreachable": true}
fatal: [192.168.56.107]: UNREACHABLE! => {"changed": false, "msg": "Failed to co
nnect to the host via ssh: ssh: connect to host 192.168.56.107 port 22: No route
 to host", "unreachable": true}
ok: [192.168.56.117]

TASK [update repository index] *************************************************
skipping: [192.168.56.117]
changed: [192.168.56.108]

TASK [install apache2 package] *************************************************
skipping: [192.168.56.117]
ok: [192.168.56.108]
```

```
TASK [install apache2 package] *************************************************
skipping: [192.168.56.117]
ok: [192.168.56.108]

TASK [add PHP support for apache2] *********************************************
skipping: [192.168.56.117]
ok: [192.168.56.108]

PLAY RECAP *********************************************************************
192.168.56.106             : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.107             : ok=0    changed=0    unreachable=1    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.108             : ok=4    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.117             : ok=1    changed=0    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
```

- As observed in the result, the task was skipped for the CentOS remote server. This is because we specified in the playbook that the distribution should be Ubuntu for the task to be executed.

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

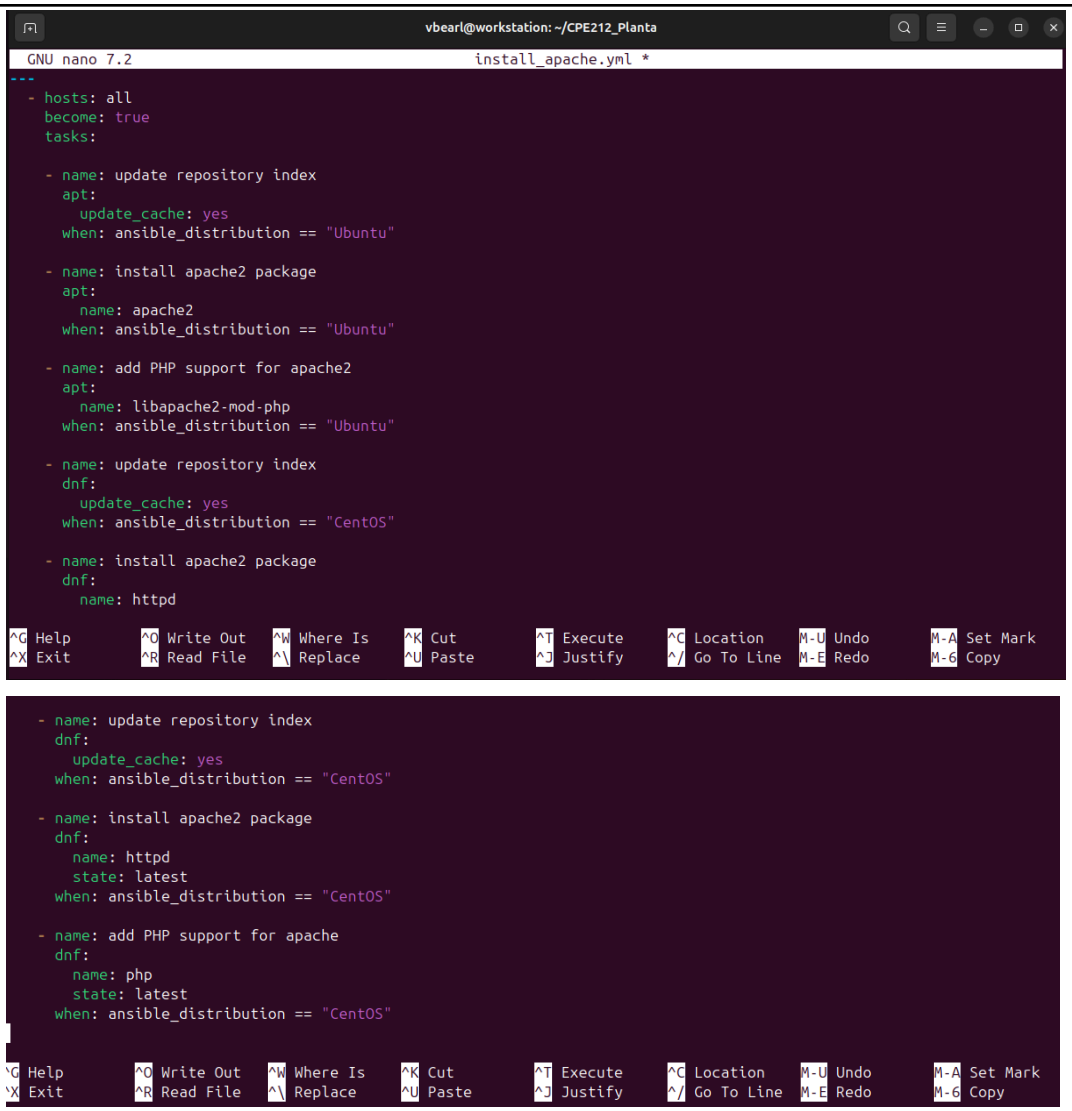
```
- name: update repository index
  apt:
    update_cache: yes
  when: ansible_distribution in [“Debian”, “Ubuntu]
```

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.



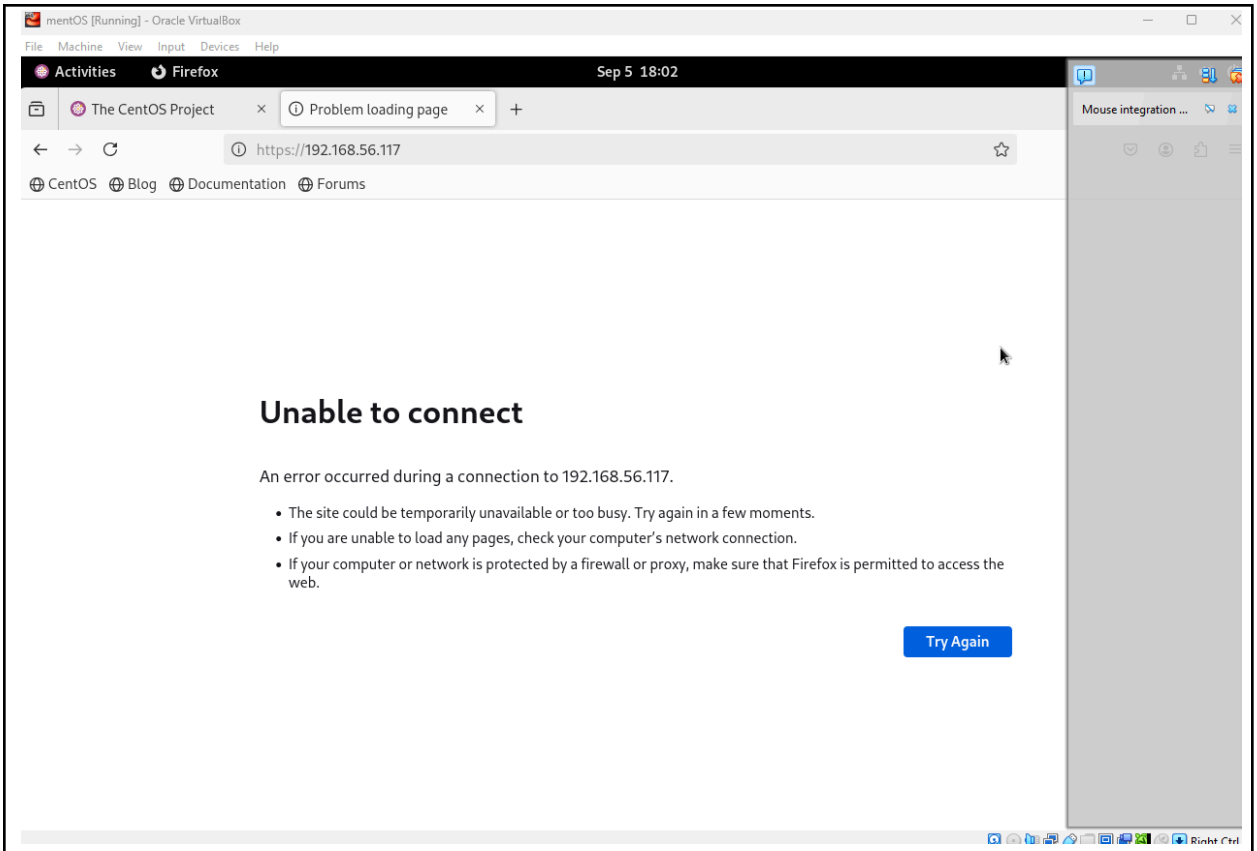
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.

```
vbearl@workstation:~/CPE212_Planta$ ansible-playbook --ask-become-pass install_a
pache.yml
BECOME password:

PLAY [ubuntu_centos] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]
ok: [192.168.56.117]

TASK [install apache2 package] *************************************************
skipping: [192.168.56.117]
ok: [192.168.56.108]

TASK [add PHP support for apache2] *********************************************
skipping: [192.168.56.117]
ok: [192.168.56.108]

TASK [install apache2 package] *************************************************
skipping: [192.168.56.108]
ok: [192.168.56.117]

TASK [add PHP support for apache] **********************************************
skipping: [192.168.56.108]
```

```
TASK [add PHP support for apache] **********************************************
skipping: [192.168.56.108]
ok: [192.168.56.117]

PLAY RECAP *********************************************************************
192.168.56.108             : ok=3    changed=0    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.56.117             : ok=3    changed=0    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0

vbearl@workstation:~/CPE212_Planta$
```

- After we created more tasks targeted to centOS, If the target of the task is Ubuntu, it skips the hosts that use centOS distribution, and vice versa.

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 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

```
[vbearl@centOS ~]$ systemctl status httpd
o 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)
```

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

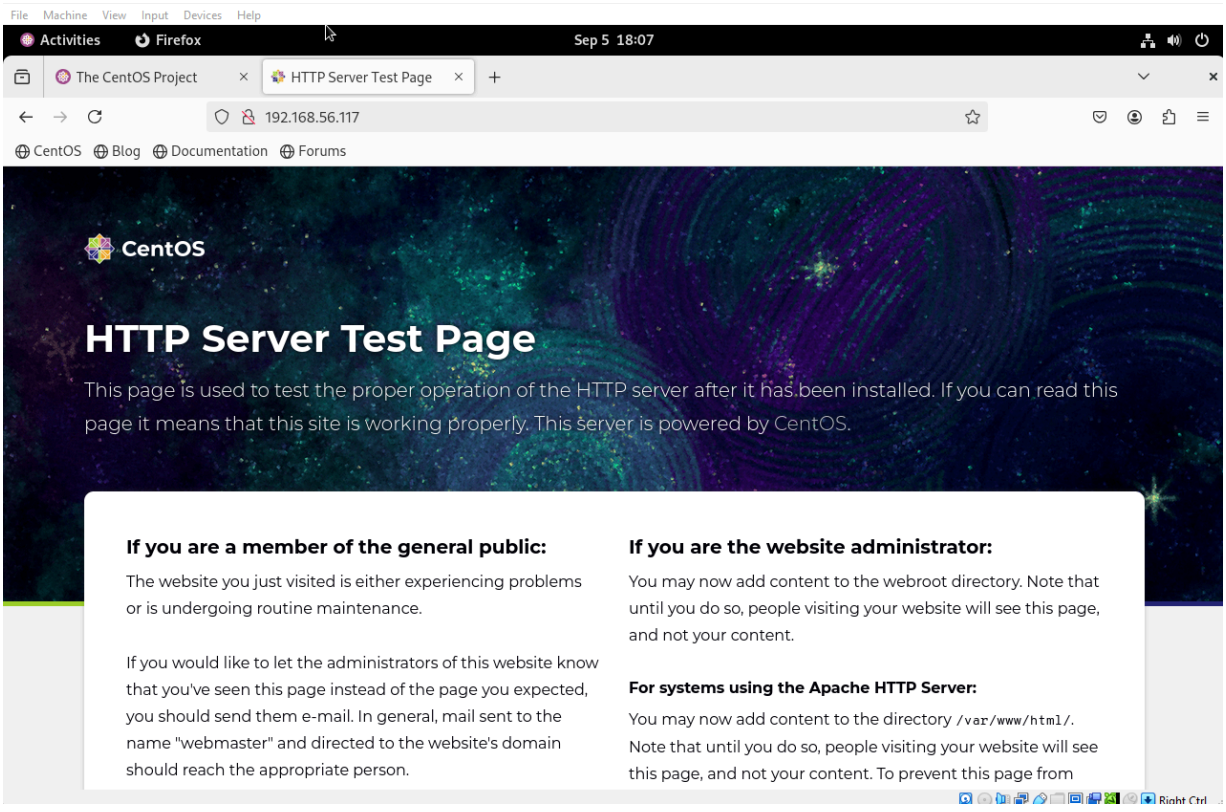
```
[vbearl@centOS ~]$ sudo systemctl start httpd
[sudo] password for vbearl:
```

(When prompted, enter the sudo password)
sudo firewall-cmd --add-port=80/tcp

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

(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)



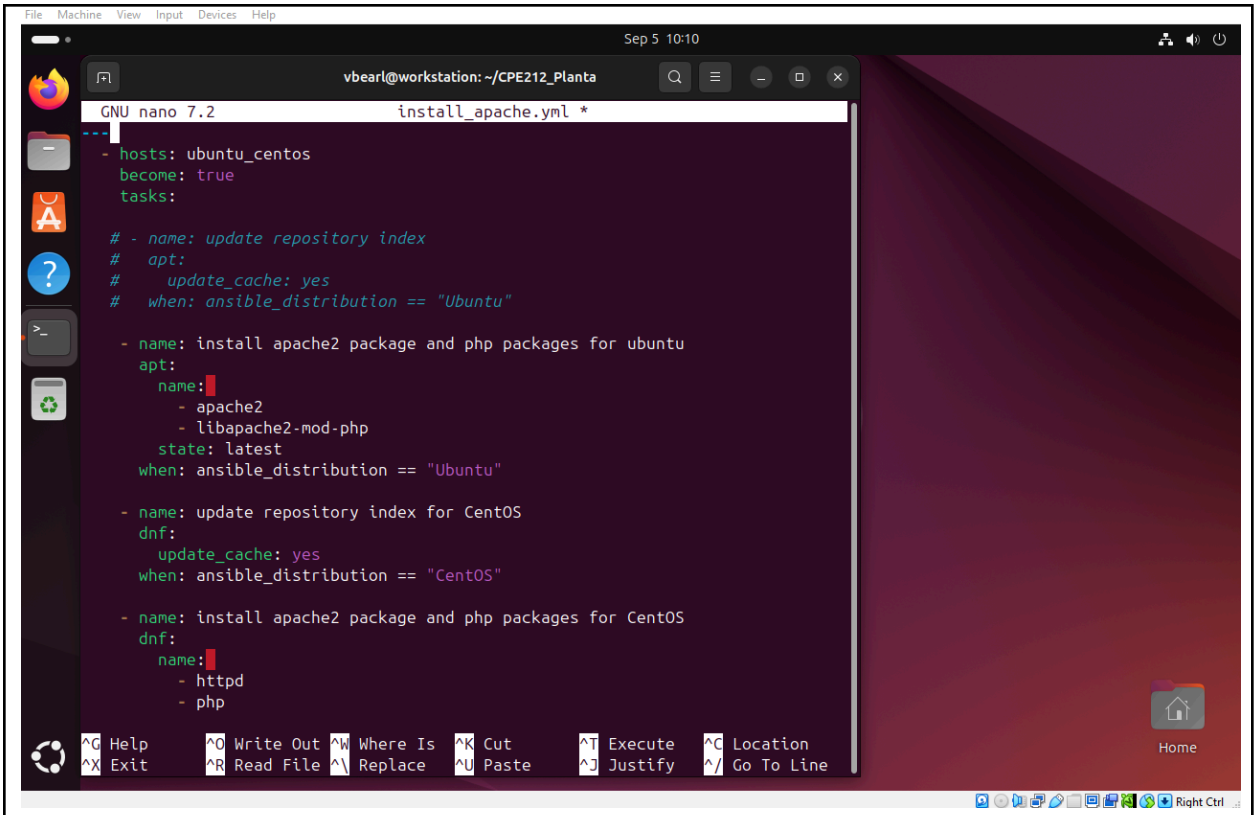
Task 2: Refactoring playbook

This time, we want to make sure that our playbook is efficient and that the codes are easier to read. This will also make 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.



Note: I skipped the update repository task for now due to ubuntu servers having an error.

```
- name: install apache2 package and php packages for CentOS
  dnf:
    name:
      - httpd
      - php
    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
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

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

```
PLAY [ubuntu_centos] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]
ok: [192.168.56.117]

TASK [install apache2 package and php packages for ubuntu] *********************
skipping: [192.168.56.117]
ok: [192.168.56.108]

TASK [update repository index for CentOS] **************************************
skipping: [192.168.56.108]
ok: [192.168.56.117]

TASK [install apache2 package and php packages for CentOS] *********************
skipping: [192.168.56.108]
ok: [192.168.56.117]

TASK [add PHP support for apache] **********************************************
skipping: [192.168.56.108]
ok: [192.168.56.117]

PLAY RECAP *********************************************************************
192.168.56.108             : ok=2    changed=0    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
192.168.56.117             : ok=4    changed=0    unreachable=0    failed=0    s
kipped=1    rescued=0    ignored=0
```

2. Edit the playbook *install_apache.yml* again. In task 2.1, we consolidated the plays into one play. This time we can actually consolidate 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.

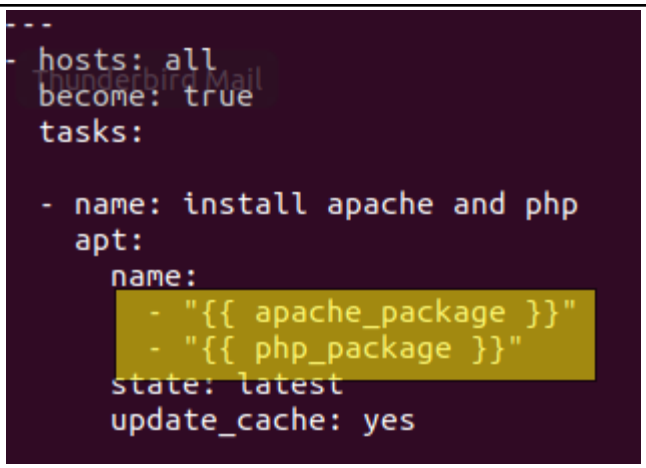
```
  GNU nano 7.2                    install_apache.yml *
---
  - hosts: ubuntu_centos
    become: true
    tasks:

    - 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 apache2 package and php packages for CentOS
      dnf:
        name:
          - httpd
          - php
        state: latest
        update_cache: yes
      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
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

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

```
vbearl@workstation:~/CPE212_Planta$ ansible-playbook --ask-become-pass install_a
pache.yml
BECOME password:

PLAY [ubuntu_centos] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]
ok: [192.168.56.117]

TASK [install apache2 package and php packages for ubuntu] *********************
skipping: [192.168.56.117]
ok: [192.168.56.108]

TASK [install apache2 package and php packages for CentOS] *********************
skipping: [192.168.56.108]
ok: [192.168.56.117]

TASK [add PHP support for apache] **********************************************
skipping: [192.168.56.108]
ok: [192.168.56.117]

PLAY RECAP *********************************************************************
192.168.56.108             : ok=2    changed=0    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.56.117             : ok=3    changed=0    unreachable=0    failed=0    s
kipped=1    rescued=0    ignored=0
```

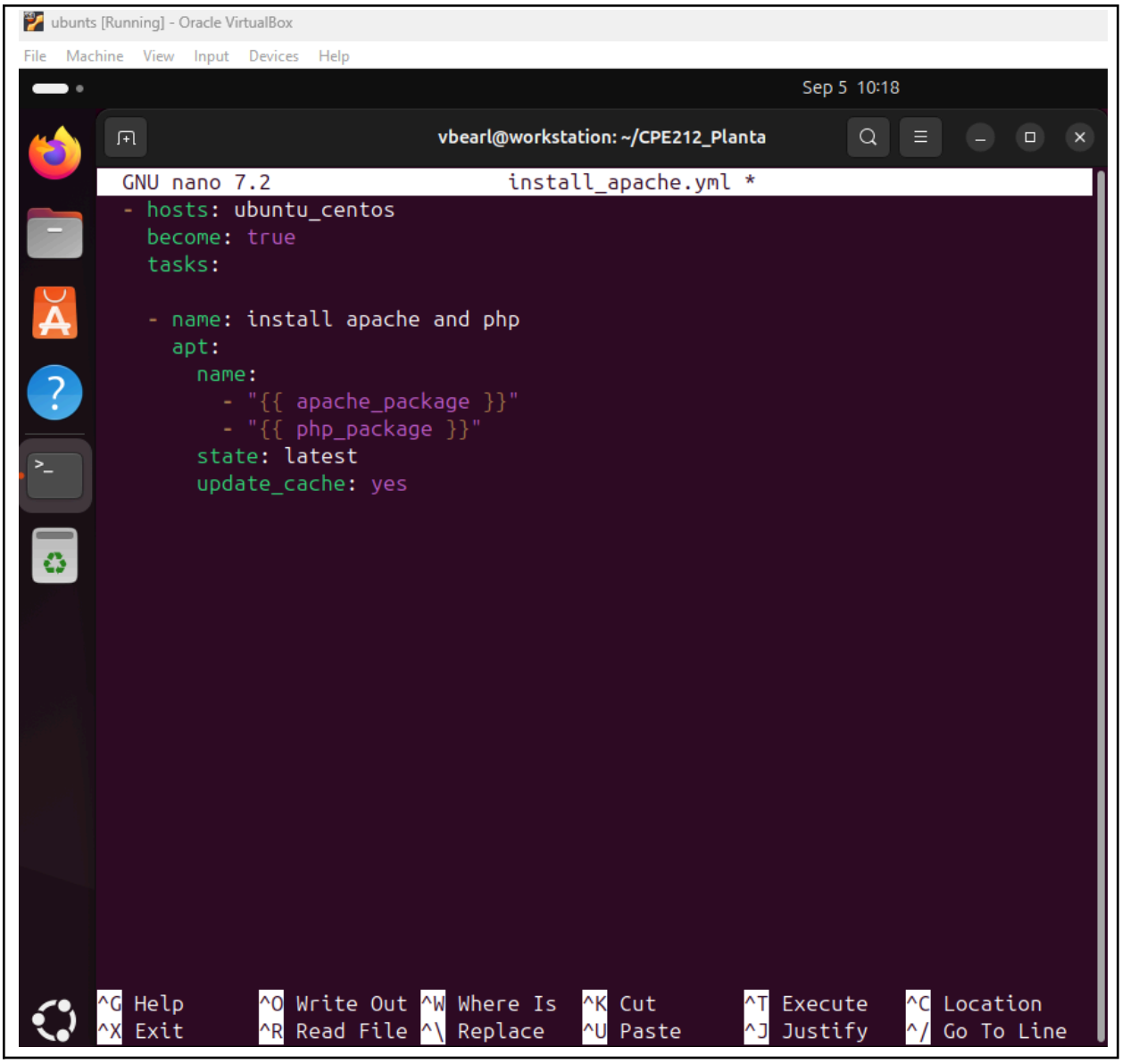
- The result stays the same as the previous one. The difference is that we were able to simplify our playbook by combining the update cache along with the installation tasks, reducing the playbook to 2 plays only.

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.


ubunts [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Sep 5 10:18
vbearl@workstation: ~/CPE212_Planta
GNU nano 7.2 install_apache.yml *
- hosts: ubuntu_centos
become: true
tasks:
- name: install apache and php
apt:
name:
- "{{ apache_package }}"
- "{{ php_package }}"
state: latest
update_cache: yes
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line

```
vbearl@workstation:~/CPE212_Planta$ ansible-playbook --ask-become-pass install_a
pache.yml
BECOME password:

PLAY [ubuntu_centos] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]
ok: [192.168.56.117]

TASK [install apache and php] **************************************************
fatal: [192.168.56.108]: FAILED! => {"msg": "The task includes an option with an
 undefined variable. The error was: 'apache_package' is undefined. 'apache_packa
ge' is undefined\n\nThe error appears to be in '/home/vbearl/CPE212_Planta/insta
ll_apache.yml': line 6, column 7, but may\nbe elsewhere in the file depending on
 the exact syntax problem.\n\nThe offending line appears to be:\n\n\n    - name:
 install apache and php\n      ^ here\n"}
fatal: [192.168.56.117]: FAILED! => {"msg": "The task includes an option with an
 undefined variable. The error was: 'apache_package' is undefined. 'apache_packa
ge' is undefined\n\nThe error appears to be in '/home/vbearl/CPE212_Planta/insta
ll_apache.yml': line 6, column 7, but may\nbe elsewhere in the file depending on
 the exact syntax problem.\n\nThe offending line appears to be:\n\n\n    - name:
 install apache and php\n      ^ here\n"}

PLAY RECAP *********************************************************************
192.168.56.108             : ok=1    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
192.168.56.117             : ok=1    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=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.

```
[ubuntu_centos]
192.168.56.108 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.117 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]

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

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

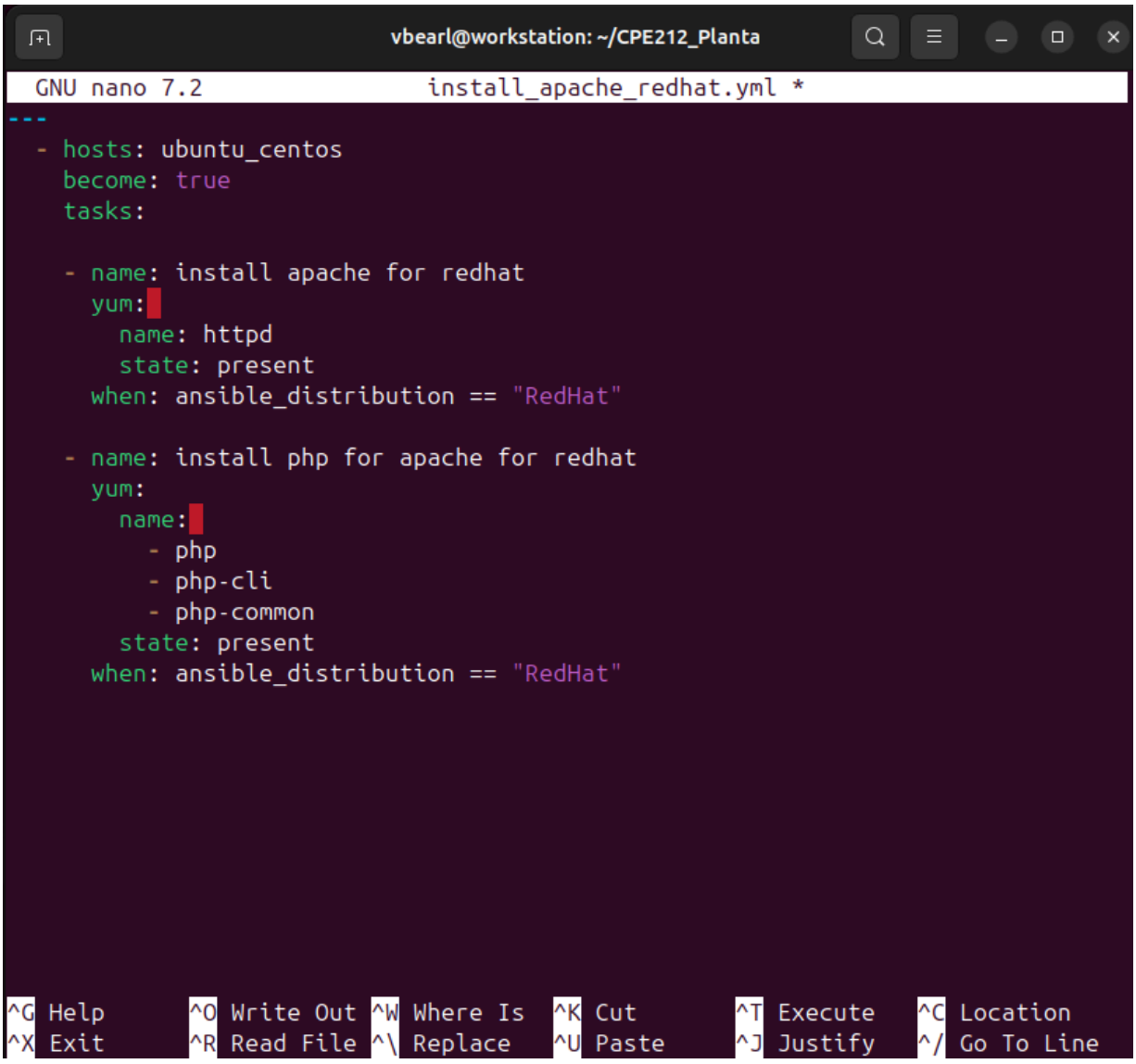
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.

```
vbearl@workstation:~/CPE212_Planta$ ansible-playbook --ask-become-pass install_a
pache.yml
BECOME password:

PLAY [ubuntu_centos] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]
ok: [192.168.56.117]

TASK [install apache and php] **************************************************
ok: [192.168.56.117]
ok: [192.168.56.108]

PLAY RECAP *********************************************************************
192.168.56.108             : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.117             : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

vbearl@workstation:~/CPE212_Planta$
```

- The output is still the same as the previous versions of the playbooks we had. The core difference is that we were able to reduce the codes into one but still does the same.

Supplementary Activity:

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



This playbook does the same as what the previous playbooks did, but this one is for hosts with the Red Hat distribution. Red Hat uses yum as its package, and the state is defined as ‘present’ instead of ‘latest’. To ensure that the playbook targets the said distribution, I included when: ansible_distribution == “RedHat”.

```
vbearl@workstation:~/CPE212_Planta$ ansible-playbook --ask-become-pass install_a
pache_redhat.yml
BECOME password:

PLAY [ubuntu_centos] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.117]
ok: [192.168.56.108]

TASK [install apache for redhat] ***********************************************
skipping: [192.168.56.108]
skipping: [192.168.56.117]

TASK [install php for apache for redhat] ***************************************
skipping: [192.168.56.108]
skipping: [192.168.56.117]

PLAY RECAP *********************************************************************
192.168.56.108             : ok=1    changed=0    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
192.168.56.117             : ok=1    changed=0    unreachable=0    failed=0    s
kipped=2    rescued=0    ignored=0
```

- As seen in the result, all tasks were skipped for the hosts, which uses Ubuntu and CentOS distributions since they do not satisfy the requirement of using the redhat distribution.

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?
   - Refactoring playbook codes can be important when managing ansibles for servers, especially when your playbook is complex and long. Simplifying your playbook by merging tasks into one promotes simplicity and organization, which makes it less harder to read and debug, and allows you to code tasks efficiently. As seen in the output, we started off with individual blocks of codes that performed one task each, and by the end of the procedure, we were able to compress all the tasks in one block of code, which is more simple and efficient.
2. When do we use the “when” command in the playbook?
   - We use the “when” command when we want a specific task in the playbook to only be directed into hosts with specific distributions. As demonstrated in the procedure, we used the when distribution to update our ubuntu node, which uses the apt package. This wont work for the node that uses CentOS, since it

uses a different package (yum/dnf). Since we wouldn't want errors to occur, we use the when command so that it targets ubuntu only and skips centOS.