Name: Emmanuell Dave G. Santos	**Date Performed: 09/19/2025**
Course/Section: CPE212-CPE32S2	**Date Submitted: 09/19/2025**
Instructor: Engr. RObin Valenzuela	**Semester and 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.

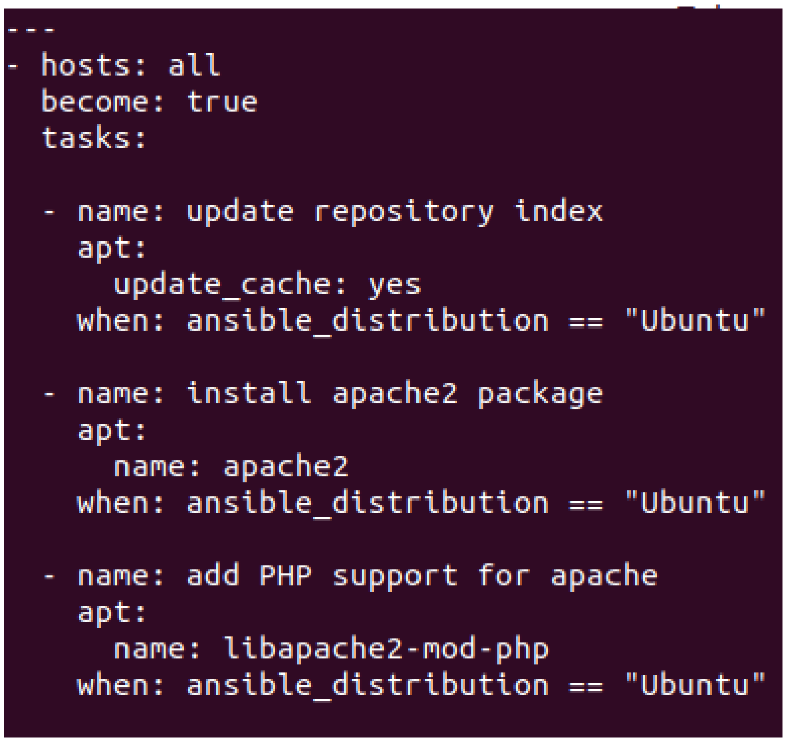
Requirement:
In this activity, you will need to create a CentOS VM. Likewise, you need to activate the second adapter to a host-only adapter after the installations. Take note of the IP address of the CentOS VM. Make sure to use the command *ssh-copy-id* to copy the public key to CentOS. Verify if you 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 issue this command. Did something happen? Why?
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

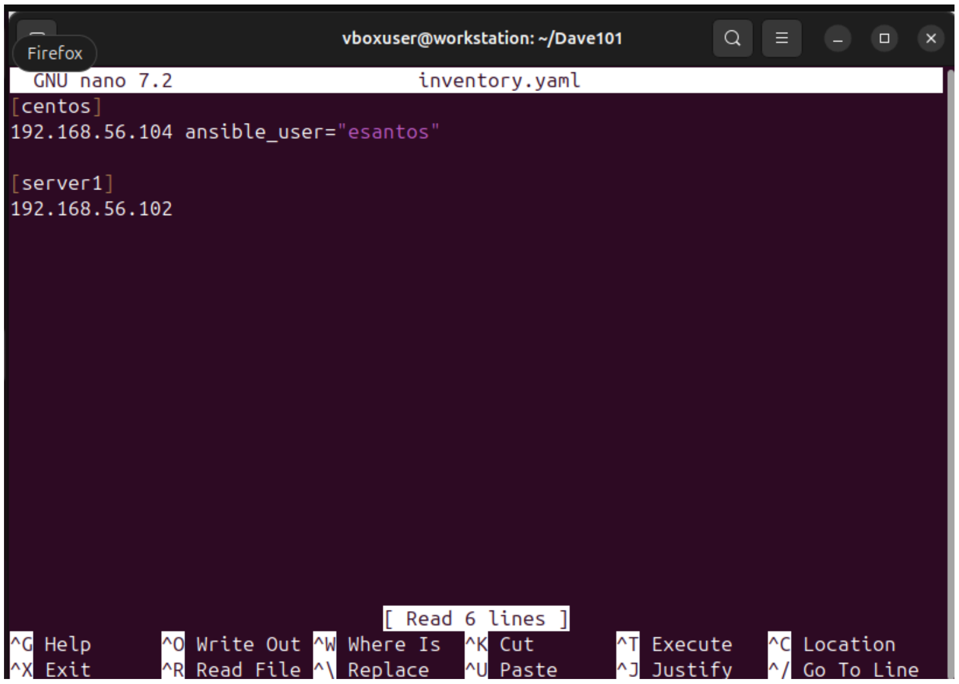
the Centos VM. You can see that the Centos VM has failed=1. Only the two remote servers have been changed. The reason is that Centos VM does not support "apt" as the package manager. The default package manager for Centos is "yum."

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



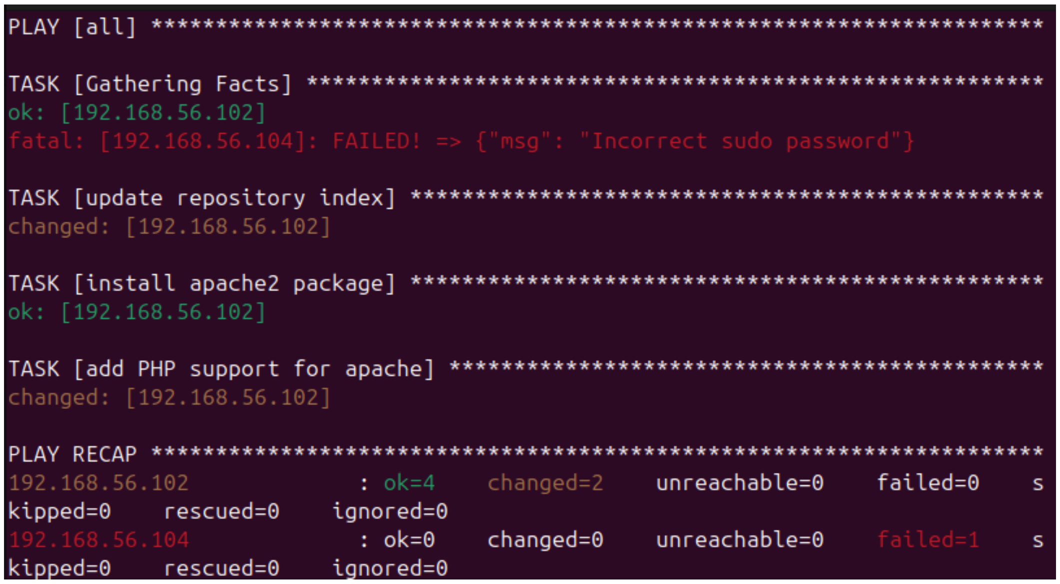
```
---
- hosts: all
  become: true
  tasks:

  - 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                       inventory.yaml
[centos]
192.168.56.104 ansible_user="esantos"

[server1]
192.168.56.102




                              [ Read 6 lines ]
^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.



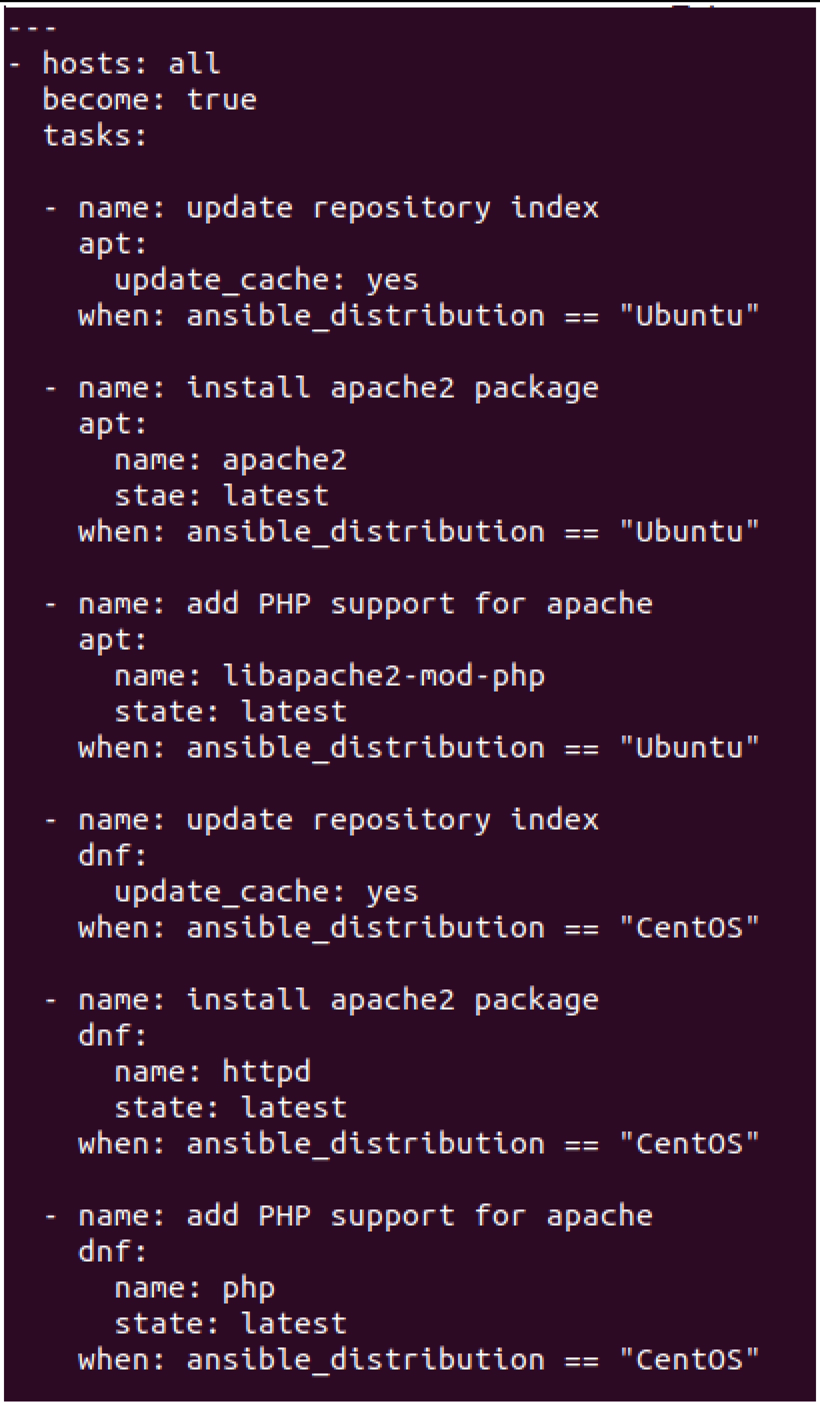
```
PLAY [all] *********************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.102]
fatal: [192.168.56.104]: FAILED! => {"msg": "Incorrect sudo password"}

TASK [update repository index] *************************************************
changed: [192.168.56.102]

TASK [install apache2 package] *************************************************
ok: [192.168.56.102]

TASK [add PHP support for apache] **********************************************
changed: [192.168.56.102]

PLAY RECAP *********************************************************************
192.168.56.102             : ok=4    changed=2    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.104             : ok=0    changed=0    unreachable=0    failed=1    s
kipped=0    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]
```

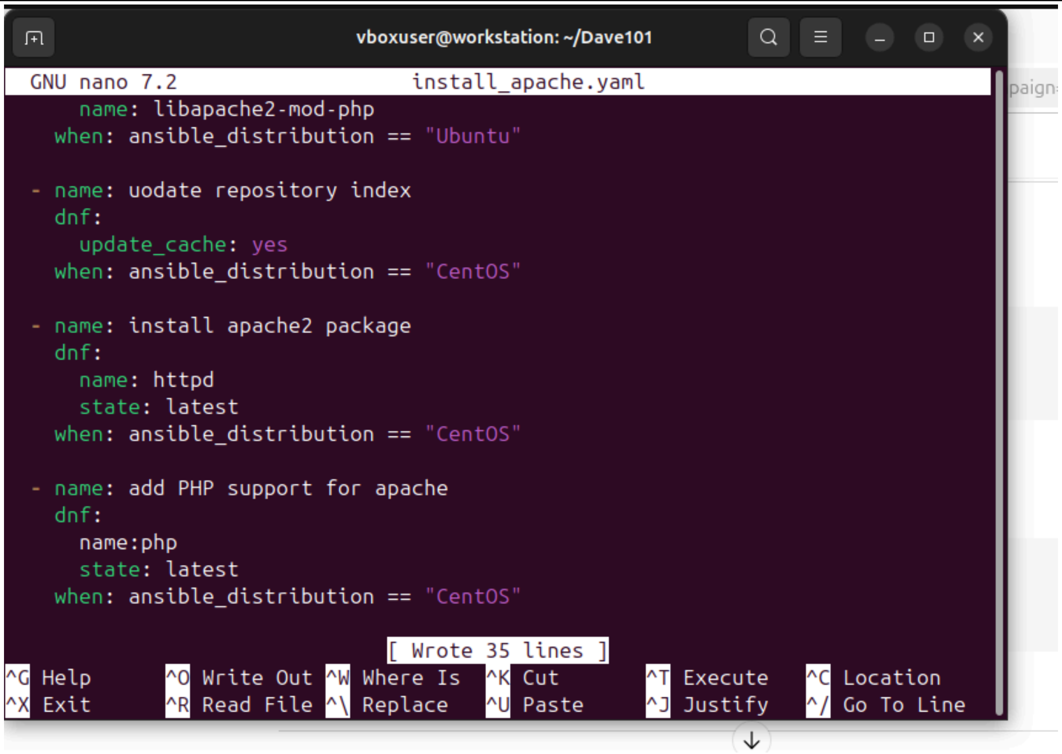
Note: This will work also if you try. Notice the changes are highlighted.

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



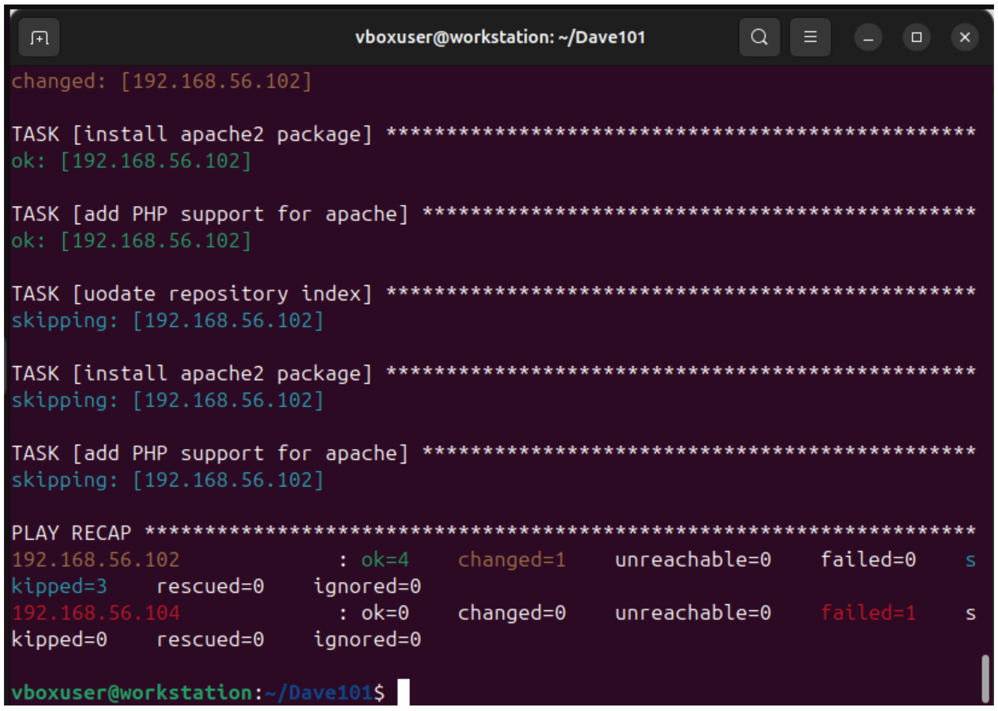
```
---
- hosts: all
  become: true
  tasks:

  - 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.



```
vboxuser@workstation: ~/Dave101
GNU nano 7.2                      install_apache.yaml
      name: libapache2-mod-php
    when: ansible_distribution == "Ubuntu"

  - name: uodate 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"

                             [ Wrote 35 lines ]
^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.



```
vboxuser@workstation: ~/Dave101
changed: [192.168.56.102]

TASK [install apache2 package] ************************************************
ok: [192.168.56.102]

TASK [add PHP support for apache] *********************************************
ok: [192.168.56.102]

TASK [uodate repository index] ************************************************
skipping: [192.168.56.102]

TASK [install apache2 package] ************************************************
skipping: [192.168.56.102]

TASK [add PHP support for apache] *********************************************
skipping: [192.168.56.102]

PLAY RECAP ********************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
192.168.56.104             : ok=0    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0

vboxuser@workstation:~/Dave101$
```

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.

Yes it is successful

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

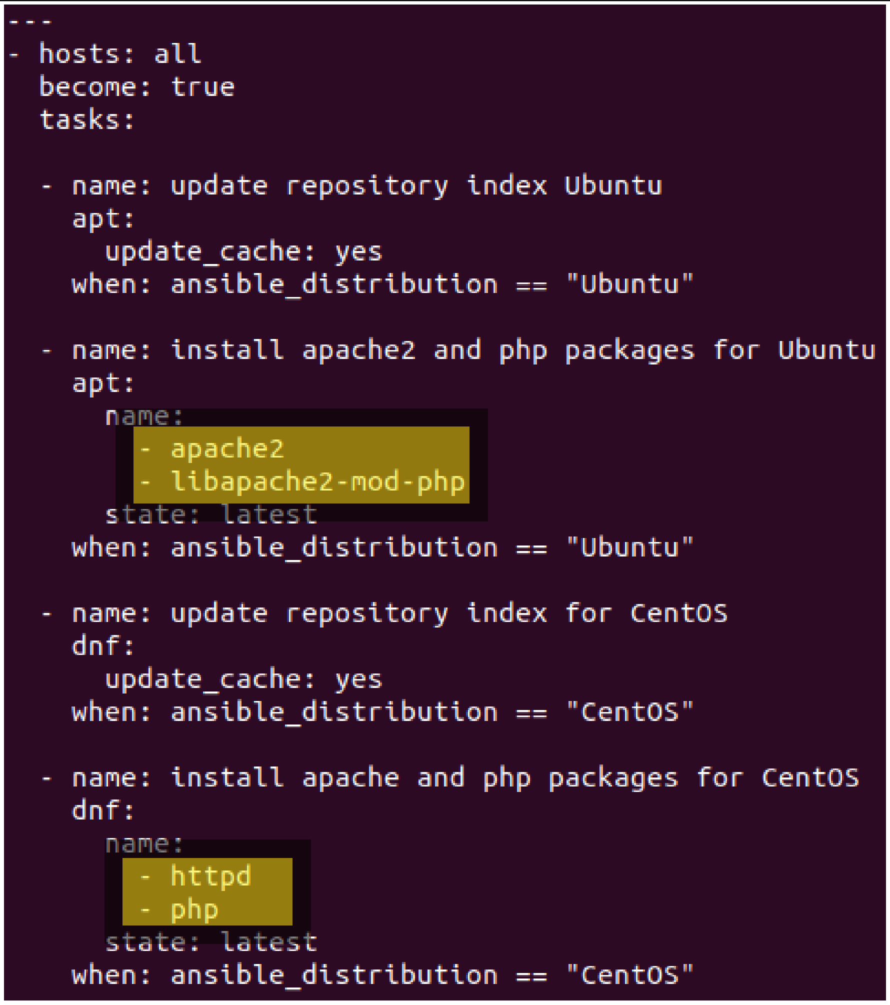
(The result should be a success)

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

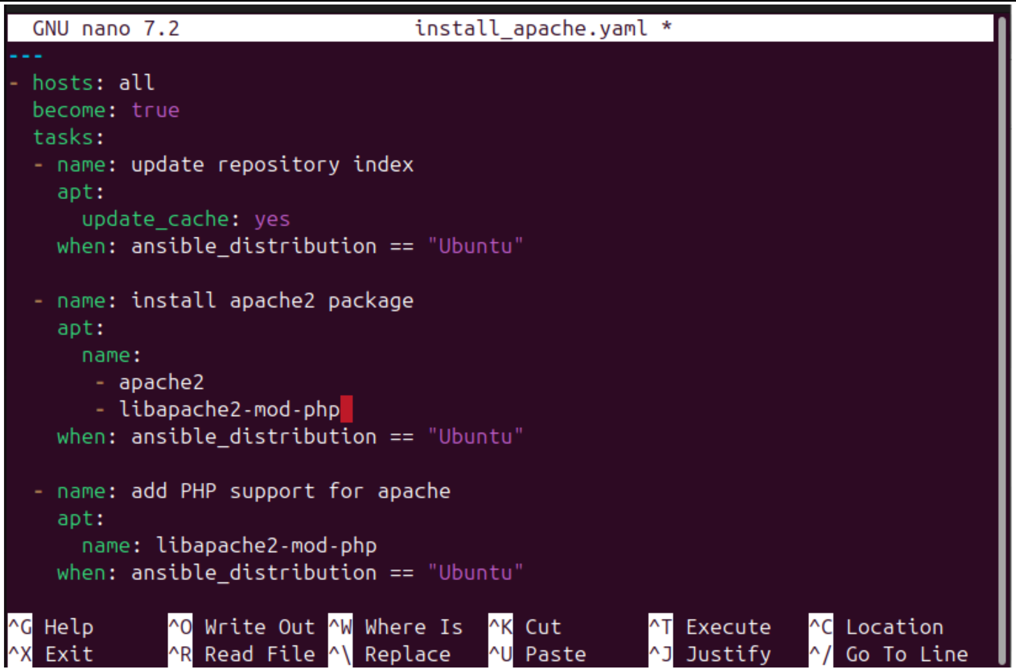
This time, we want to make sure that our playbook is efficient and that the codes are easier to read. This will also makes run ansible more quickly if it has to execute fewer tasks to do the same thing.

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:

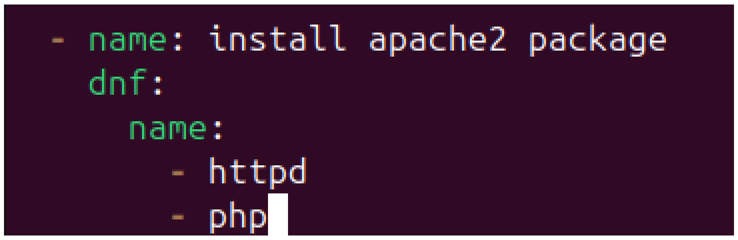


```
---
- hosts: all
  become: true
  tasks:

  - 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.

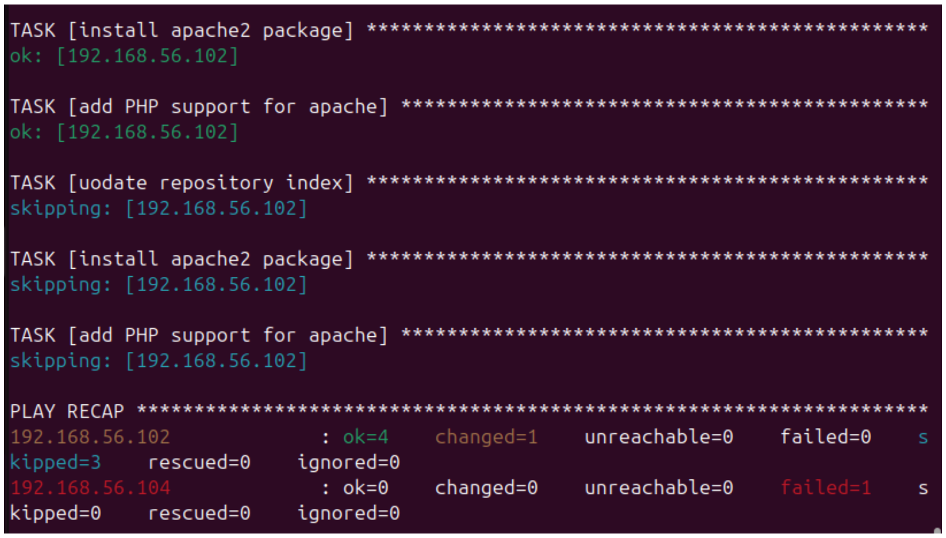


```
GNU nano 7.2                     install_apache.yaml *
---
- hosts: all
  become: true
  tasks:
  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

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

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

^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
```



```
  - name: install apache2 package
    dnf:
      name:
        - httpd
        - php
```

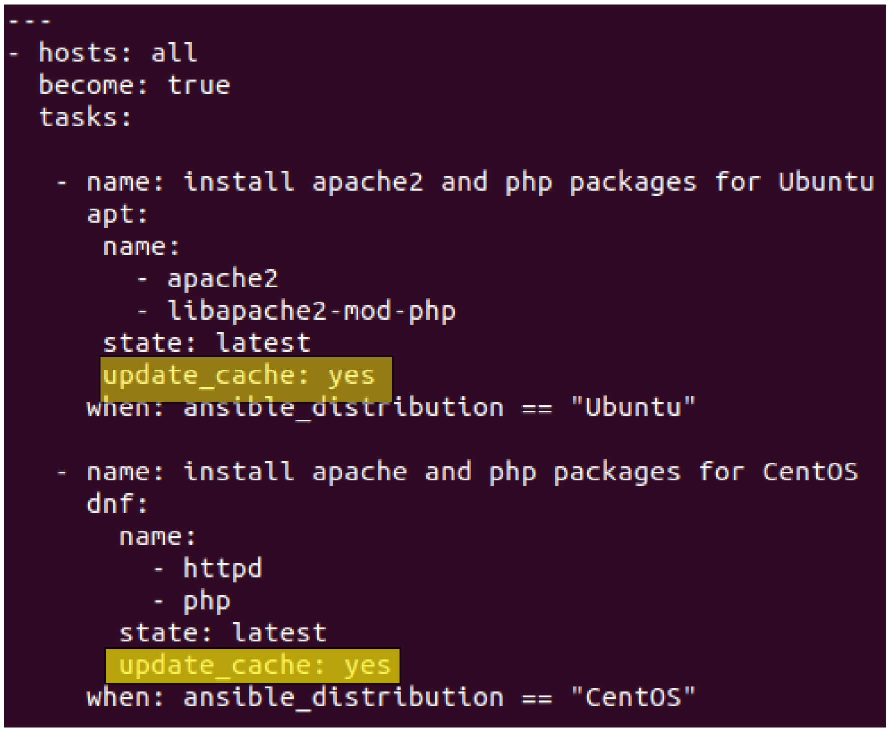
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
TASK [install apache2 package] ***************************************************
ok: [192.168.56.102]

TASK [add PHP support for apache] ************************************************
ok: [192.168.56.102]

TASK [uodate repository index] ***************************************************
skipping: [192.168.56.102]

TASK [install apache2 package] ***************************************************
skipping: [192.168.56.102]

TASK [add PHP support for apache] ************************************************
skipping: [192.168.56.102]

PLAY RECAP ***********************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0    s
kipped=3    rescued=0    ignored=0
192.168.56.104             : ok=0    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
```

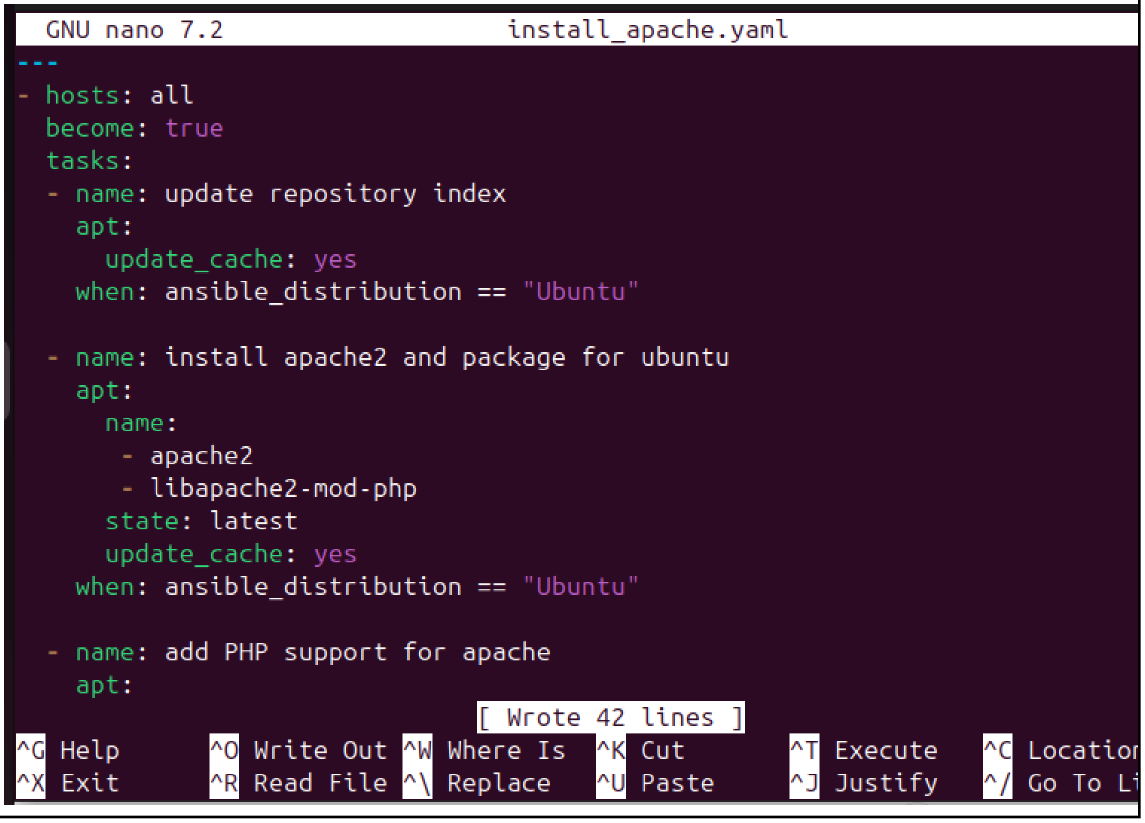
2. Edit the playbook *install_apache.yml* again. In task 2.1, we consolidated the plays into one play. This time we can actually consolidated everything in just 2 plays. This can be done by removing the update repository play and putting the

command *update_cache: yes* below the command *state: latest*. See below for reference:



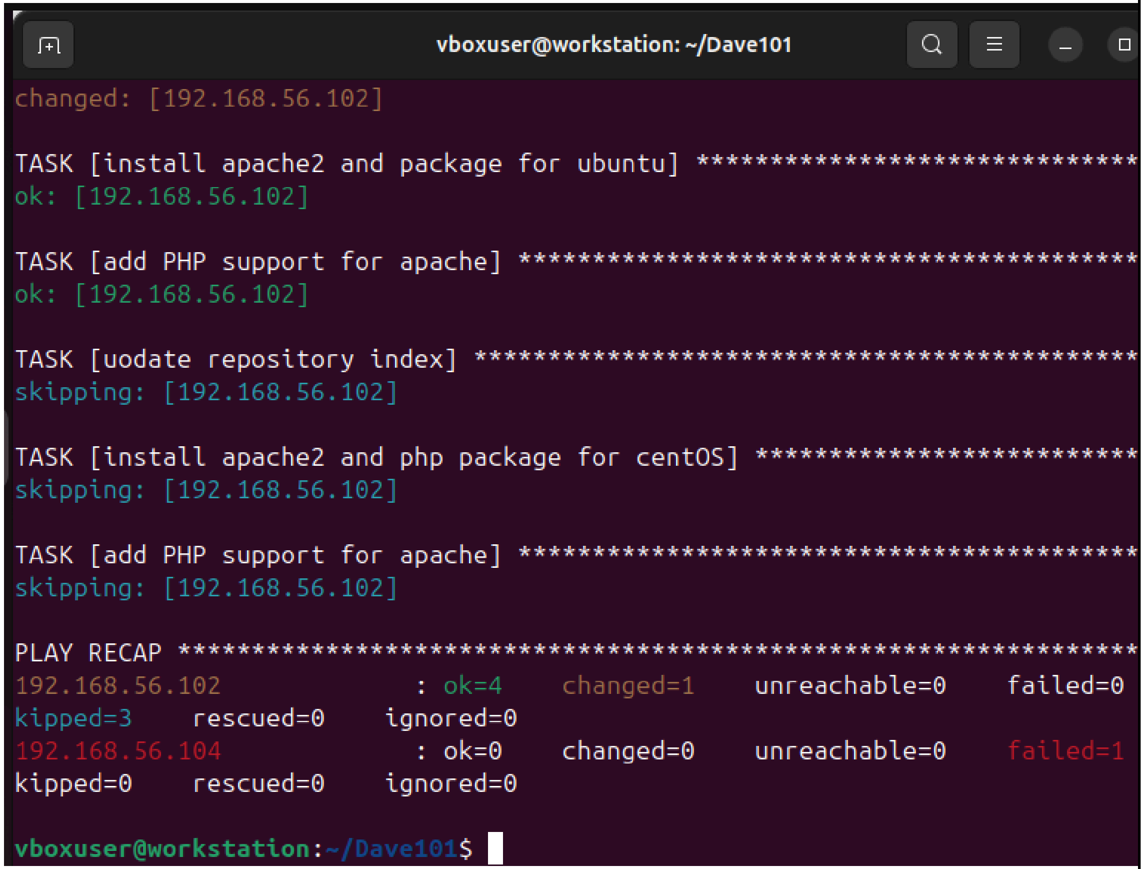
```
---
- hosts: all
  become: true
  tasks:

   - 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.



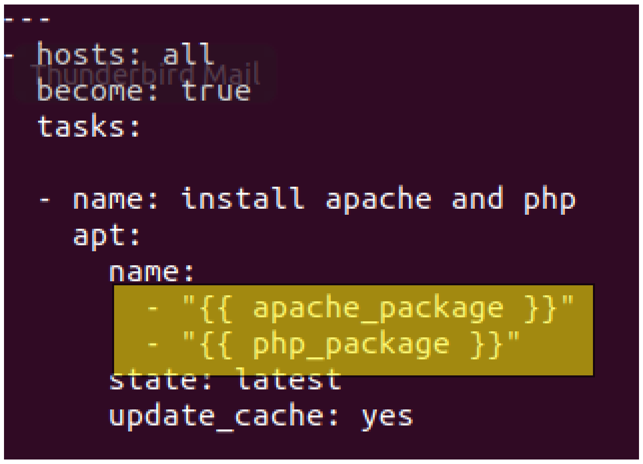
```
  GNU nano 7.2                    install_apache.yaml
---
- hosts: all
  become: true
  tasks:
  - name: update repository index
    apt:
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

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

  - name: add PHP support for apache
    apt:
                               [ Wrote 42 lines ]
^G Help       ^O Write Out ^W Where Is  ^K Cut        ^T Execute    ^C Locatior
^X Exit       ^R Read File ^\ Replace   ^U Paste      ^J Justify    ^/ Go To Li
```

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



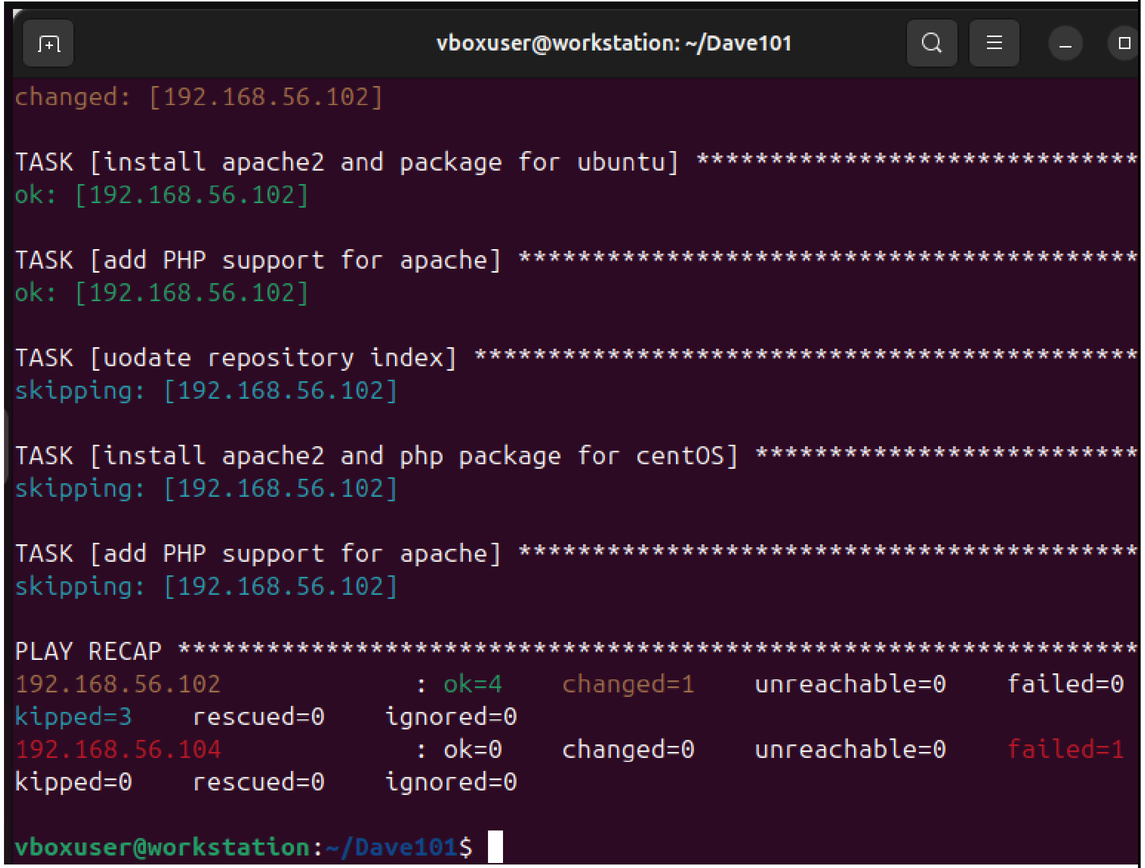
```
changed: [192.168.56.102]

TASK [install apache2 and package for ubuntu] ******************************
ok: [192.168.56.102]

TASK [add PHP support for apache] ******************************************
ok: [192.168.56.102]

TASK [uodate repository index] *********************************************
skipping: [192.168.56.102]

TASK [install apache2 and php package for centOS] **************************
skipping: [192.168.56.102]

TASK [add PHP support for apache] ******************************************
skipping: [192.168.56.102]

PLAY RECAP *****************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0
kipped=3    rescued=0    ignored=0
192.168.56.104             : ok=0    changed=0    unreachable=0    failed=1
kipped=0    rescued=0    ignored=0

vboxuser@workstation:~/Dave101$
```

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.



```
changed: [192.168.56.102]

TASK [install apache2 and package for ubuntu] ******************************
ok: [192.168.56.102]

TASK [add PHP support for apache] ******************************************
ok: [192.168.56.102]

TASK [uodate repository index] *********************************************
skipping: [192.168.56.102]

TASK [install apache2 and php package for centOS] **************************
skipping: [192.168.56.102]

TASK [add PHP support for apache] ******************************************
skipping: [192.168.56.102]

PLAY RECAP *****************************************************************
192.168.56.102             : ok=4    changed=1    unreachable=0    failed=0
kipped=3    rescued=0    ignored=0
192.168.56.104             : ok=0    changed=0    unreachable=0    failed=1
kipped=0    rescued=0    ignored=0

vboxuser@workstation:~/Dave101$
```

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

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

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

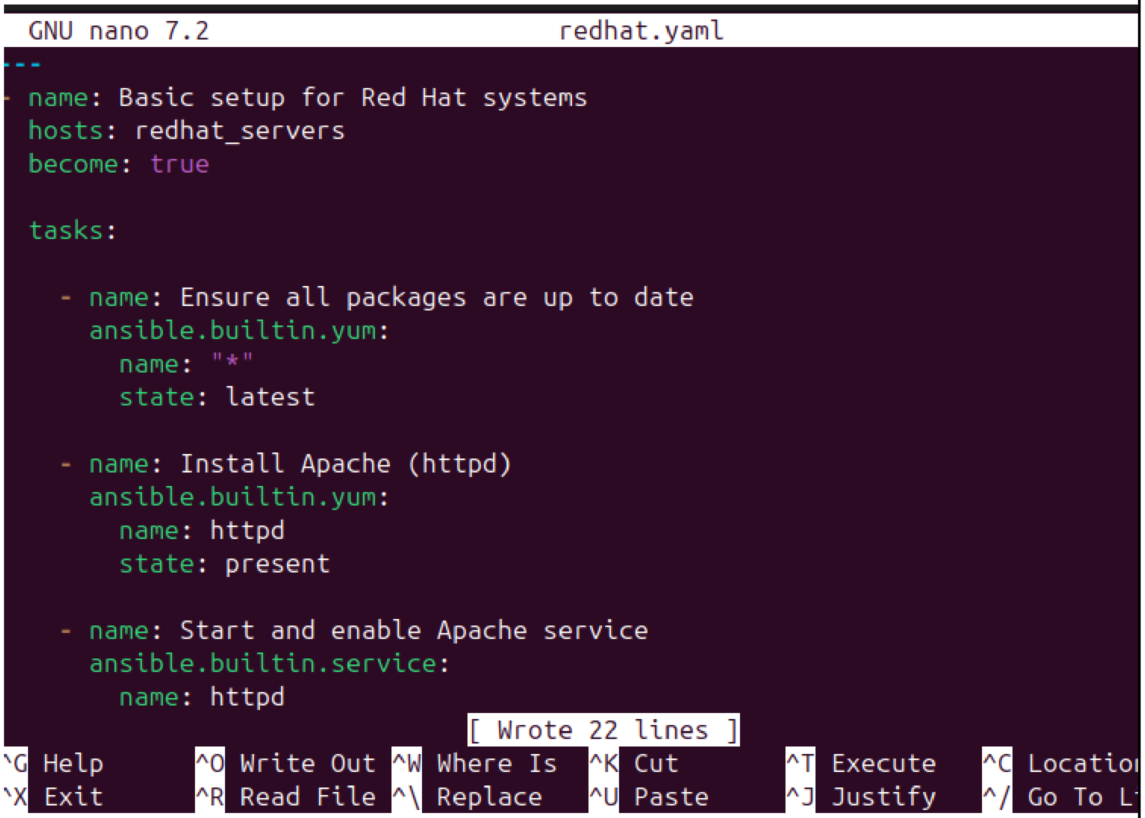
Finally, we still have one more thing to change in our *install_apache.yml* file. In task 2.3, you may notice that the package is assign as apt, which will not run in CentOS. Replace the *apt* with *package*. Package is a module in ansible that is generic, which is going to use whatever package manager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

ansible.builtin.package – Generic OS package manager — Ansible Documentation

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

Supplementary Activity:

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



```
  GNU nano 7.2                          redhat.yaml
---
- name: Basic setup for Red Hat systems
  hosts: redhat_servers
  become: true

  tasks:

    - name: Ensure all packages are up to date
      ansible.builtin.yum:
        name: "*"
        state: latest

    - name: Install Apache (httpd)
      ansible.builtin.yum:
        name: httpd
        state: present

    - name: Start and enable Apache service
      ansible.builtin.service:
        name: httpd
                                  [ Wrote 22 lines ]
^G Help        ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Locatio
^X Exit        ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To L
```

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?
   Yes it is important because it can run all modes in ubuntu and centos
2. When do we use the “when” command in playbook?

   when there is a condition