Name: Gabriel Soriano	**Date Performed: September 15, 2022**
Course/Section: CPE31S23	**Date Submitted:** **1st submission: September 15, 2022** **2nd submission: September 17, 2022**
Instructor: Engr. Taylar	**Semester and SY: 1st sem - 2022/2023**

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

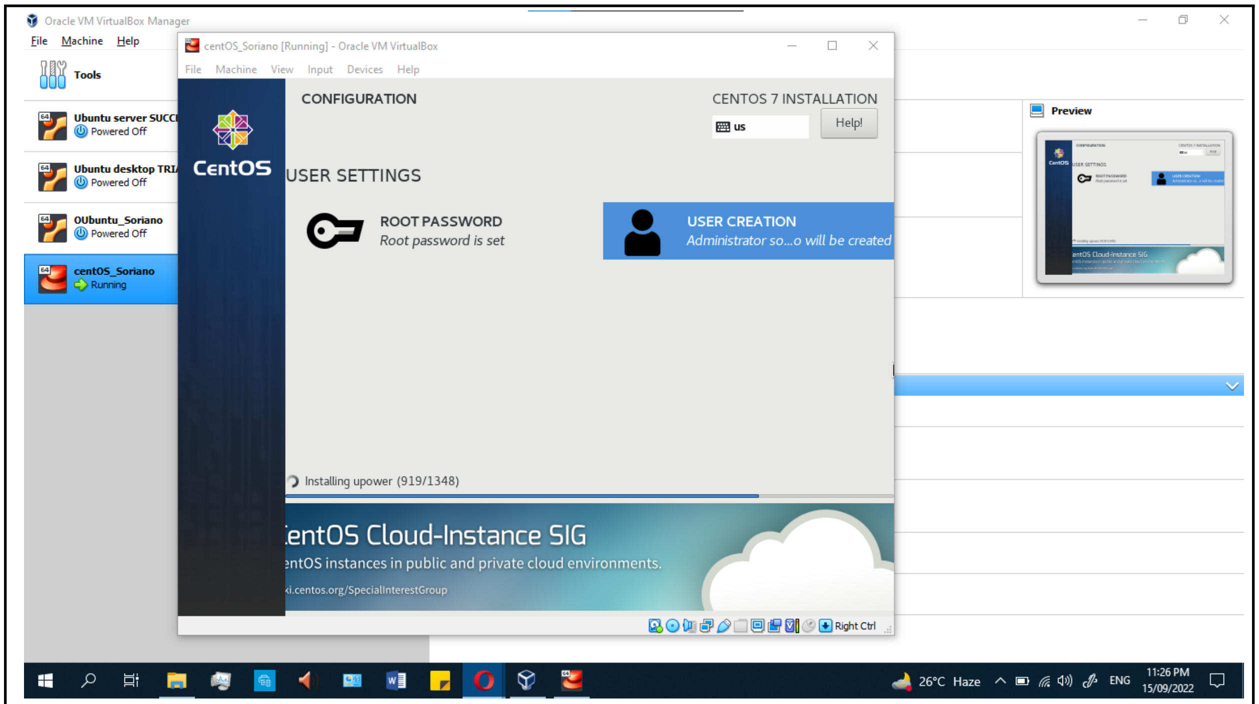
2. **Discussion**:

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.

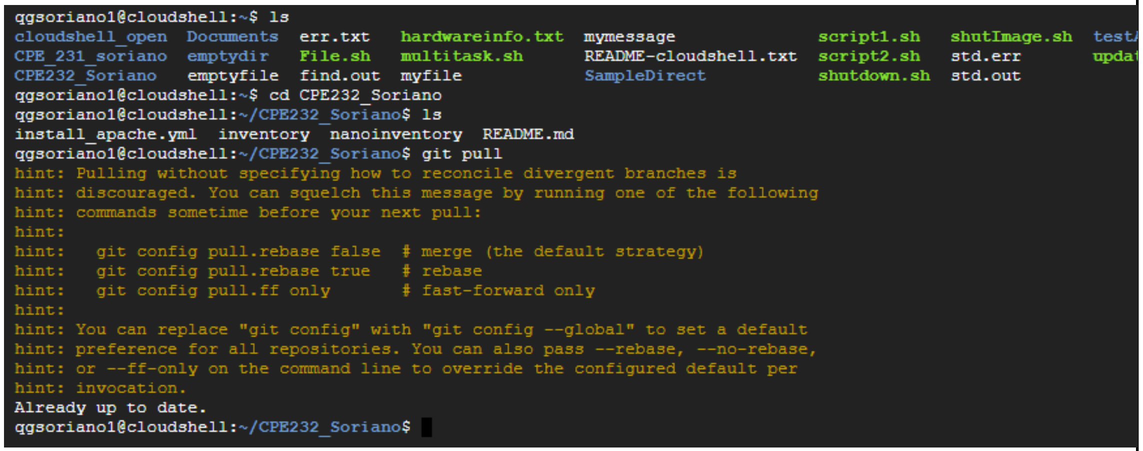
SCREENSHOT:



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?

SCREENSHOT:

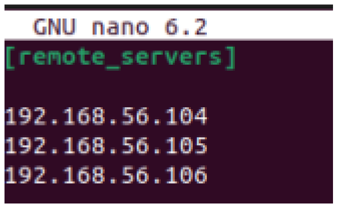


```
qgsoriano1@cloudshell:~$ ls
cloudshell_open  Documents  err.txt  hardwareinfo.txt  mymessage           script1.sh   shutImage.sh  test/
CPE_231_soriano  emptydir   File.sh  multitask.sh      README-cloudshell.txt  script2.sh   std.err       upda
CPE232_Soriano   emptyfile  find.out myfile            SampleDirect        shutdown.sh  std.out
qgsoriano1@cloudshell:~$ cd CPE232_Soriano
qgsoriano1@cloudshell:~/CPE232_Soriano$ ls
install_apache.yml  inventory  nanoinventory  README.md
qgsoriano1@cloudshell:~/CPE232_Soriano$ git pull
hint: Pulling without specifying how to reconcile divergent branches is
hint: discouraged. You can squelch this message by running one of the following
hint: commands sometime before your next pull:
hint:
hint:   git config pull.rebase false  # merge (the default strategy)
hint:   git config pull.rebase true   # rebase
hint:   git config pull.ff only       # fast-forward only
hint:
hint: You can replace "git config" with "git config --global" to set a default
hint: preference for all repositories. You can also pass --rebase, --no-rebase,
hint: or --ff-only on the command line to override the configured default per
hint: invocation.
Already up to date.
qgsoriano1@cloudshell:~/CPE232_Soriano$ 
```

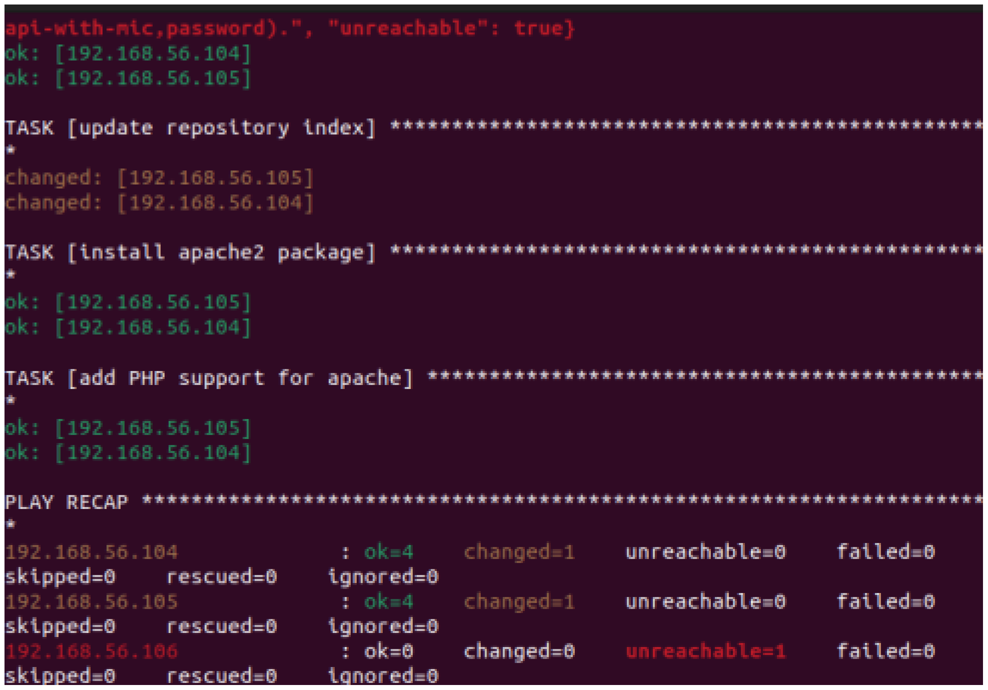
- **I don't have to enter the correct passphrase or password because it was not prompted.**

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."

SCREENSHOTS:

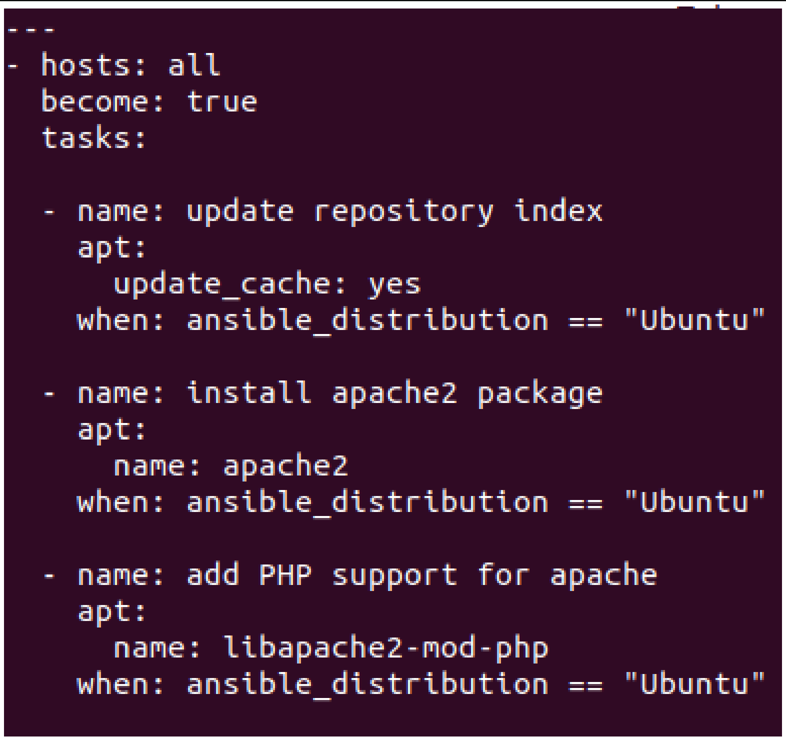


```
  GNU nano 6.2
[remote_servers]

192.168.56.104
192.168.56.105
192.168.56.106
```



```
api-with-mic,password).", "unreachable": true}
ok: [192.168.56.104]
ok: [192.168.56.105]

TASK [update repository index] ************************************************
*
changed: [192.168.56.105]
changed: [192.168.56.104]

TASK [install apache2 package] ************************************************
*
ok: [192.168.56.105]
ok: [192.168.56.104]

TASK [add PHP support for apache] *********************************************
*
ok: [192.168.56.105]
ok: [192.168.56.104]

PLAY RECAP ********************************************************************
*
192.168.56.104             : ok=4    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.105             : ok=4    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.106             : ok=0    changed=0    unreachable=1    failed=0
skipped=0    rescued=0    ignored=0
```

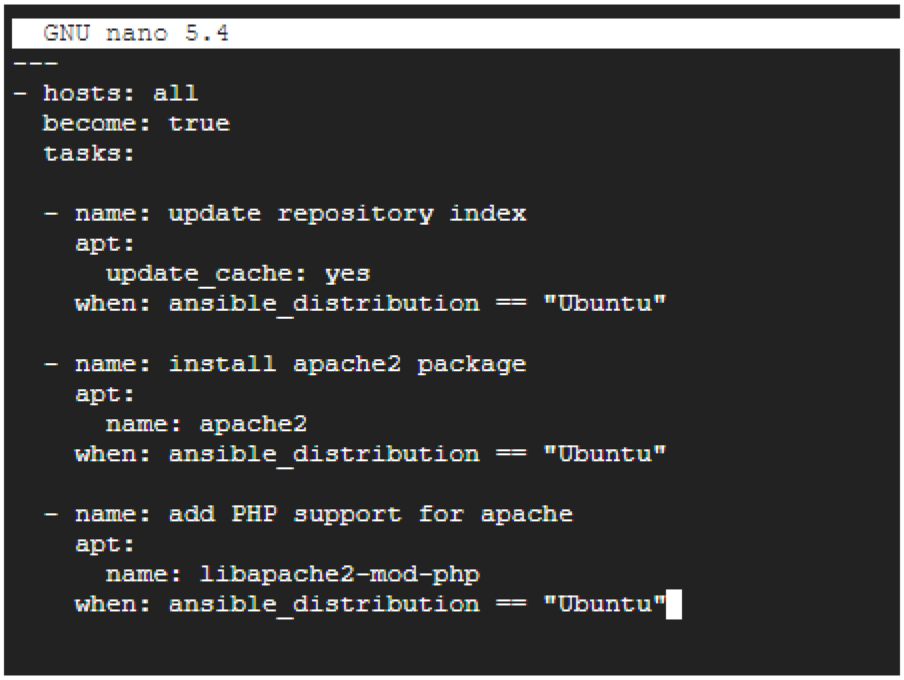
3. Edit the *install_apache.yml* file and insert the lines shown below.



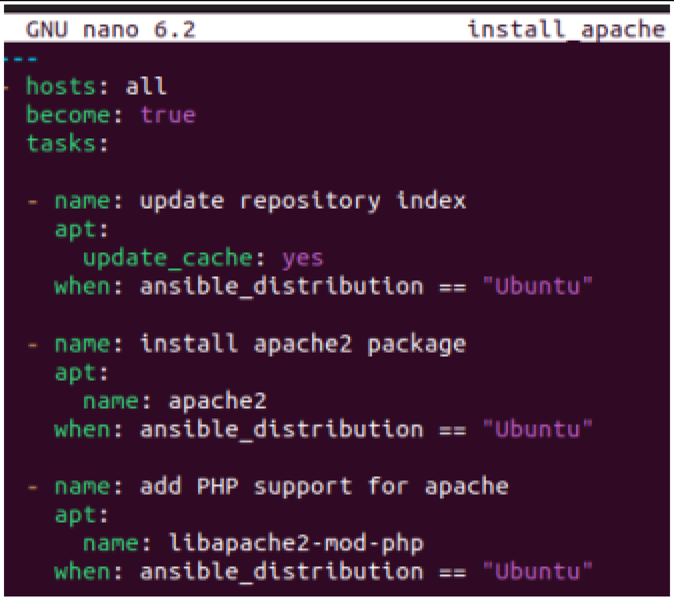
```
---
- hosts: all
  become: true
  tasks:

  - 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.

SCREENSHOT:



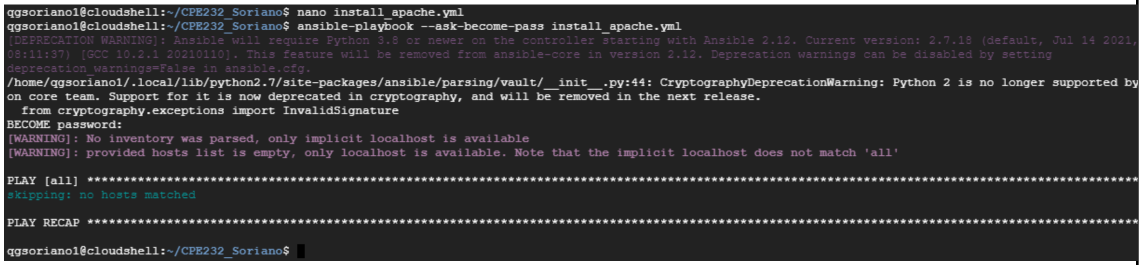
```
  GNU nano 5.4
---
- hosts: all
  become: true
  tasks:

  - 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"
```



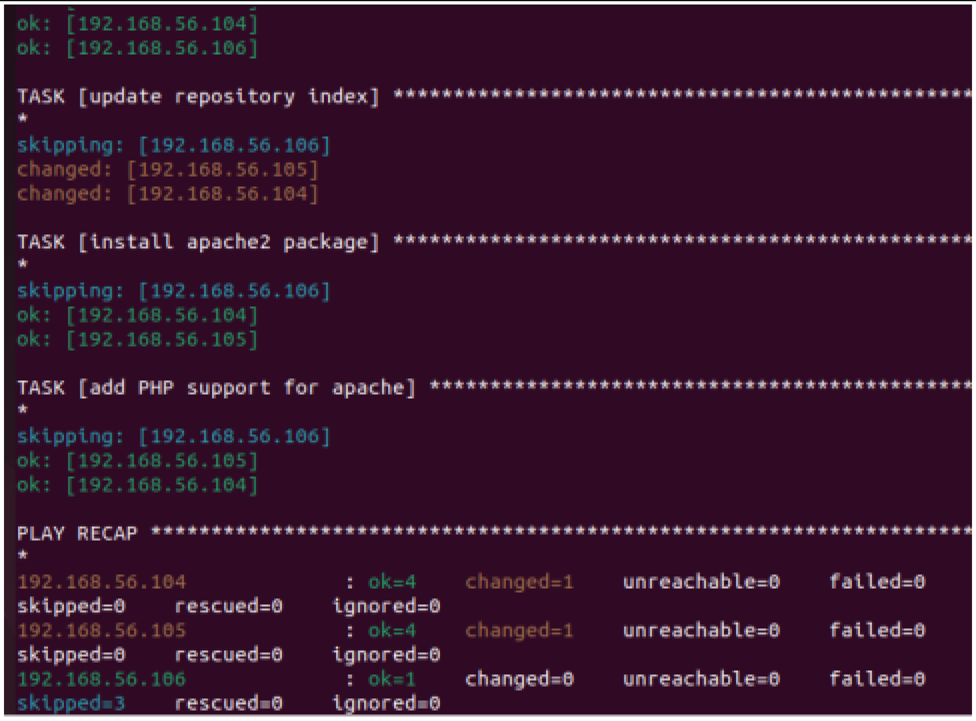
```
GNU nano 6.2                              install_apache
---
- hosts: all
  become: true
  tasks:

  - 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.

SCREENSHOT:



```
qgsoriano1@cloudshell:~/CPE232_Soriano$ nano install_apache.yml
qgsoriano1@cloudshell:~/CPE232_Soriano$ ansible-playbook --ask-become-pass install_apache.yml
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with Ansible 2.12. Current version: 2.7.18 (default, Jul 14 2021,
08:11:37) [GCC 10.2.1 20210110]. This feature will be removed from ansible-core in version 2.12. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.
/home/qgsoriano1/.local/lib/python2.7/site-packages/ansible/parsing/vault/__init__.py:44: CryptographyDeprecationWarning: Python 2 is no longer supported by
on core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
  from cryptography.exceptions import InvalidSignature
BECOME password:
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [all] *******************************************************************************************************************************
skipping: no hosts matched

PLAY RECAP *******************************************************************************************************************************

qgsoriano1@cloudshell:~/CPE232_Soriano$ 
```



```
ok: [192.168.56.104]
ok: [192.168.56.106]

TASK [update repository index] *************************************************
*
skipping: [192.168.56.106]
changed: [192.168.56.105]
changed: [192.168.56.104]

TASK [install apache2 package] *************************************************
*
skipping: [192.168.56.106]
ok: [192.168.56.104]
ok: [192.168.56.105]

TASK [add PHP support for apache] **********************************************
*
skipping: [192.168.56.106]
ok: [192.168.56.105]
ok: [192.168.56.104]

PLAY RECAP *********************************************************************
*
192.168.56.104             : ok=4    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.105             : ok=4    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.106             : ok=1    changed=0    unreachable=0    failed=0
skipped=3    rescued=0    ignored=0
```

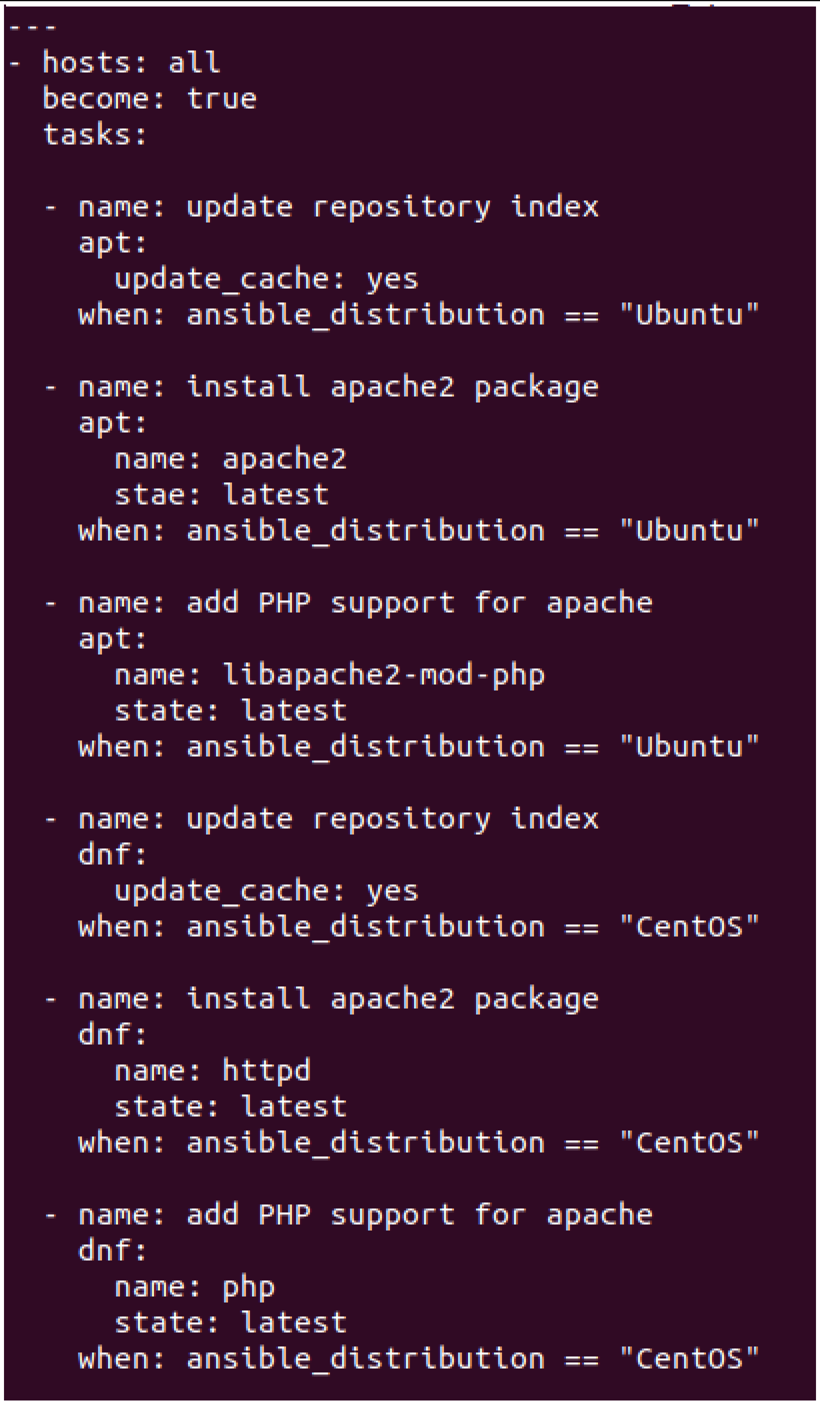
It is successful, and can be reached.

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.

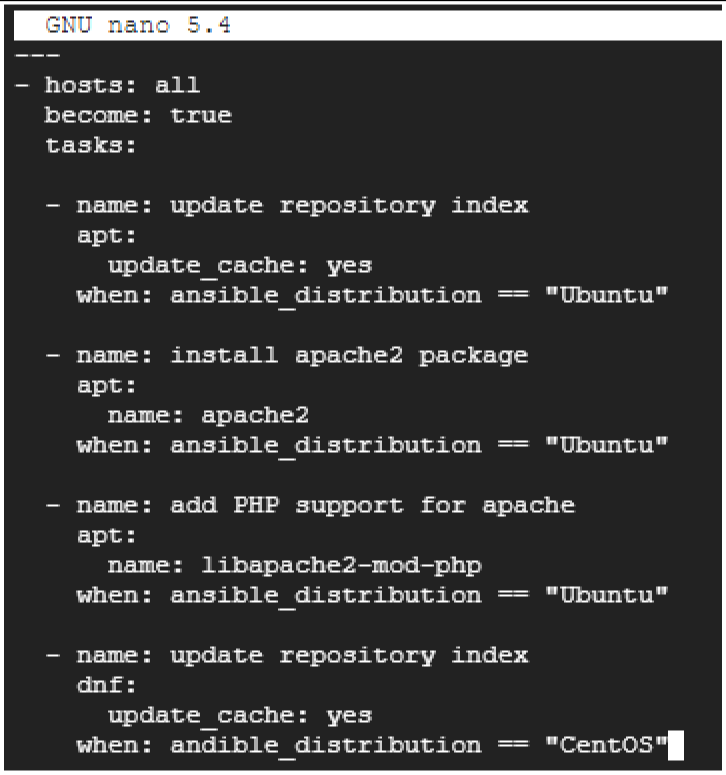
4. Edit the *install_apache.yml* file and insert the lines shown below.



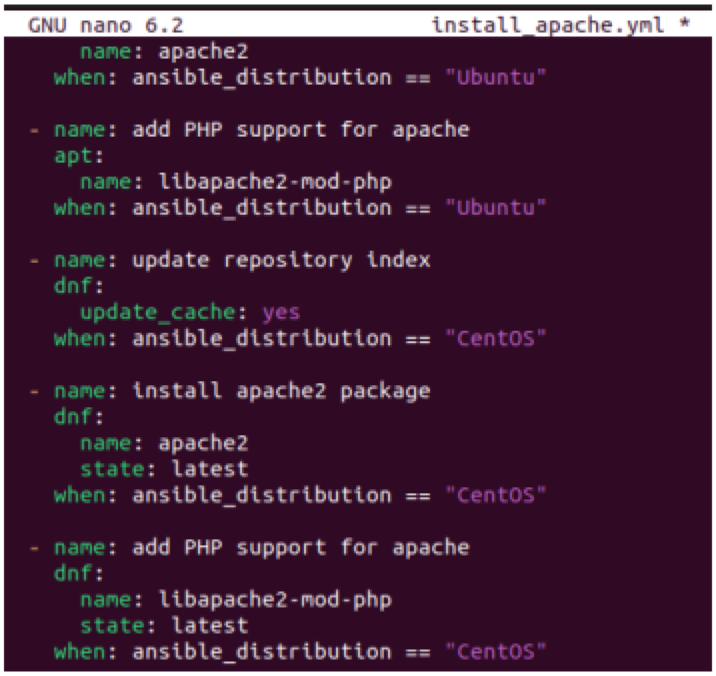
```
---
- hosts: all
  become: true
  tasks:

  - 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.

SCREENSHOT:



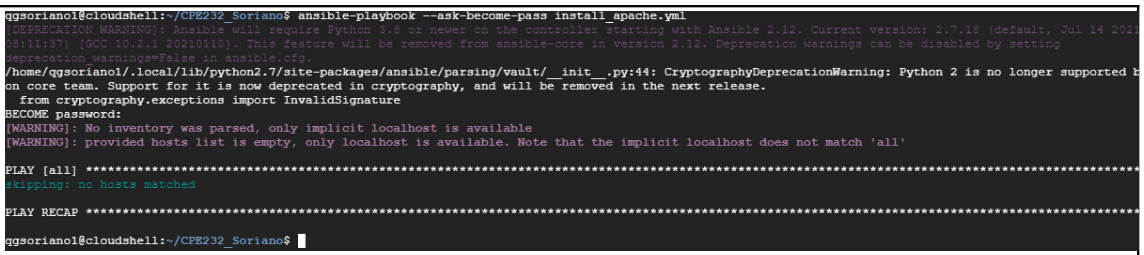
```
  GNU nano 5.4
---
- hosts: all
  become: true
  tasks:

  - 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"

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



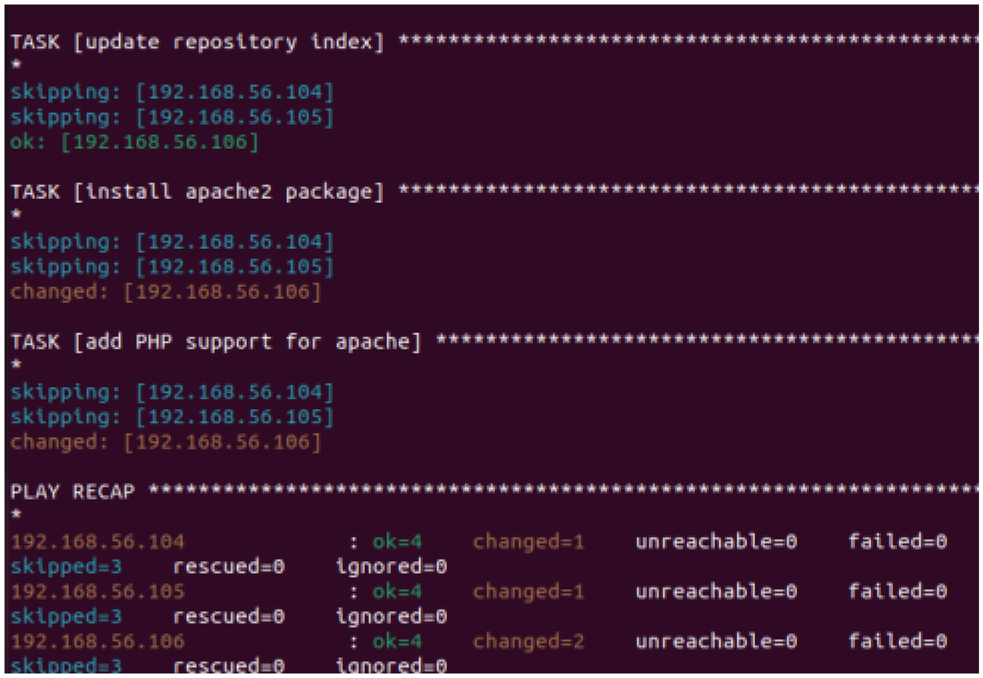
```
  GNU nano 6.2                     install_apache.yml *
      name: apache2
    when: ansible_distribution == "Ubuntu"

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

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

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

  - name: add PHP support for apache
    dnf:
      name: libapache2-mod-php
      state: latest
    when: ansible_distribution == "CentOS"
```

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

SCREENSHOT:



```
qgsoriano1@cloudshell:~/CPE232_Soriano$ ansible-playbook --ask-become-pass install_apache.yml
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with Ansible 2.12. Current version: 2.7.18 (default, Jul 14 2021
08:11:37) [GCC 10.2.1 20210110]. This feature will be removed from ansible-core in version 2.12. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.
/home/qgsoriano1/.local/lib/python2.7/site-packages/ansible/parsing/vault/__init__.py:44: CryptographyDeprecationWarning: Python 2 is no longer supported b
on core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
  from cryptography.exceptions import InvalidSignature
BECOME password:
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [all] ************************************************************************************************************************************************
skipping: no hosts matched

PLAY RECAP ************************************************************************************************************************************************

qgsoriano1@cloudshell:~/CPE232_Soriano$ 
```

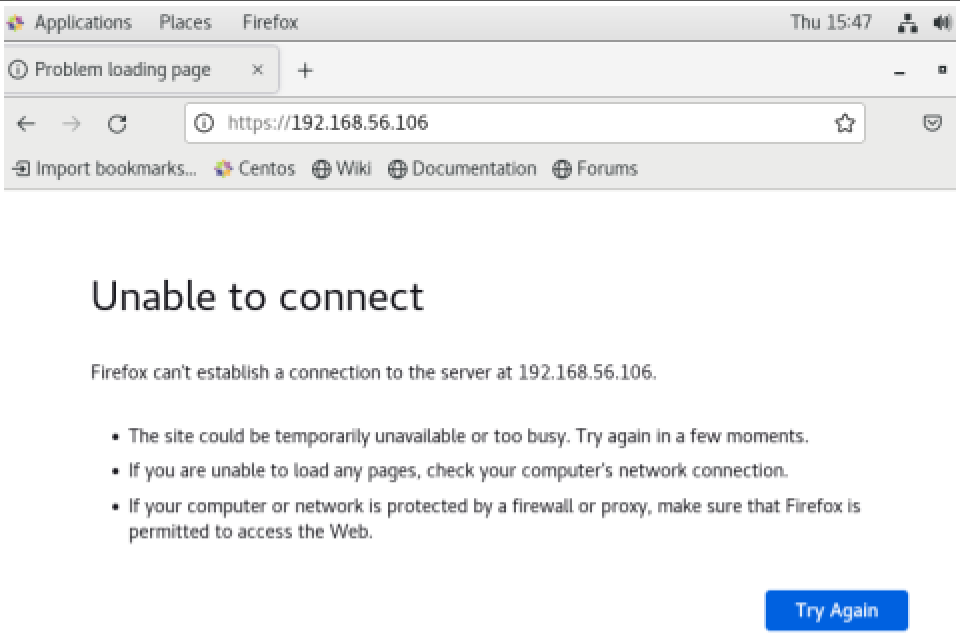


```
TASK [update repository index] ************************************************
*
skipping: [192.168.56.104]
skipping: [192.168.56.105]
ok: [192.168.56.106]

TASK [install apache2 package] ************************************************
*
skipping: [192.168.56.104]
skipping: [192.168.56.105]
changed: [192.168.56.106]

TASK [add PHP support for apache] *********************************************
*
skipping: [192.168.56.104]
skipping: [192.168.56.105]
changed: [192.168.56.106]

PLAY RECAP ********************************************************************
*
192.168.56.104             : ok=4    changed=1    unreachable=0    failed=0
skipped=3    rescued=0    ignored=0
192.168.56.105             : ok=4    changed=1    unreachable=0    failed=0
skipped=3    rescued=0    ignored=0
192.168.56.106             : ok=4    changed=2    unreachable=0    failed=0
skipped=3    rescued=0    ignored=0
```

Also successful

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.

   SCREENSHOT:



Unable to connect

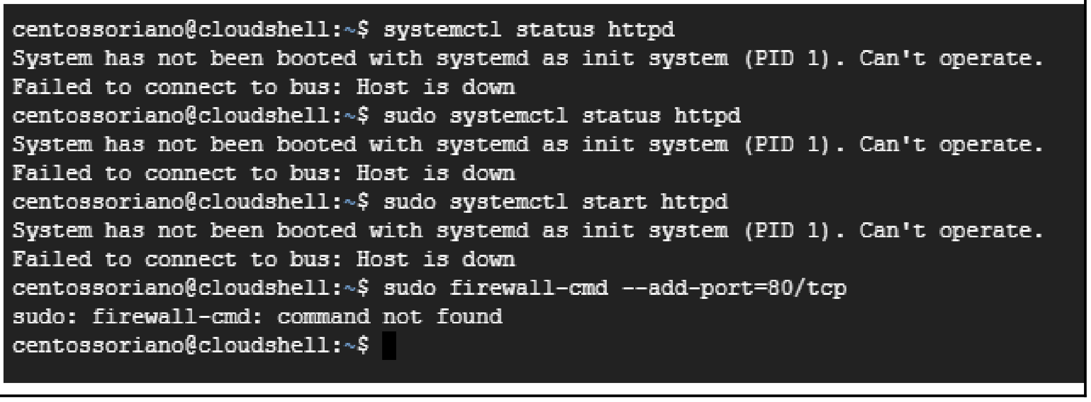
Firefox can't establish a connection to the server at 192.168.56.106.

- 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.

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)

SCREENSHOT:

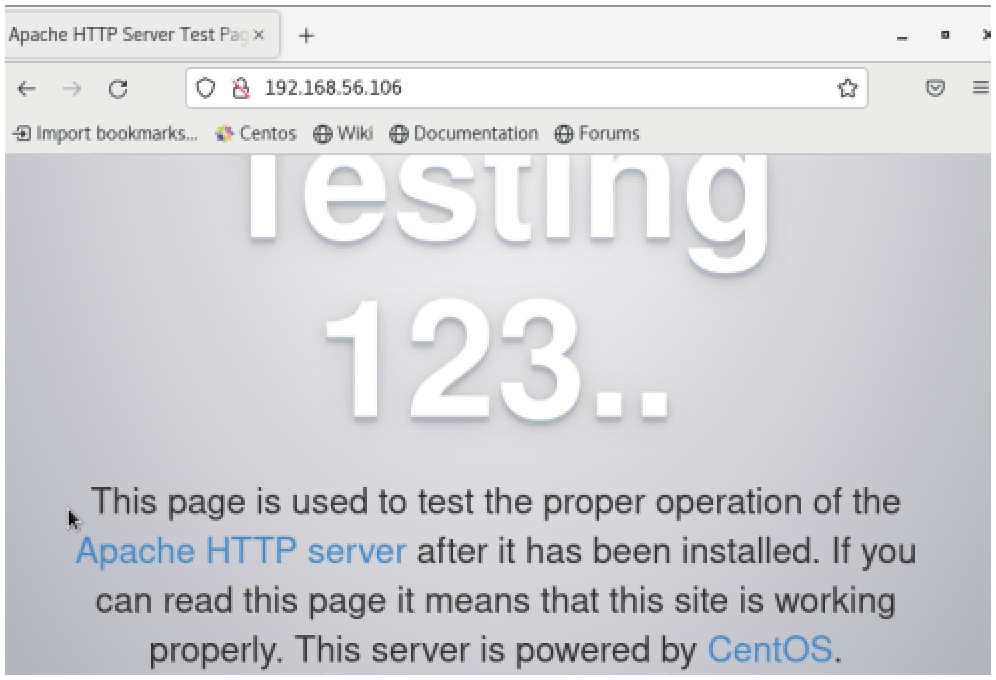


```
centossoriano@cloudshell:~$ systemctl status httpd
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
centossoriano@cloudshell:~$ sudo systemctl status httpd
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
centossoriano@cloudshell:~$ sudo systemctl start httpd
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
centossoriano@cloudshell:~$ sudo firewall-cmd --add-port=80/tcp
sudo: firewall-cmd: command not found
centossoriano@cloudshell:~$
```

```
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disa
bled)
   Active: inactive (dead)
     Docs: man:httpd(8)
           man:apachectl(8)
```

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)

SCREENSHOT:



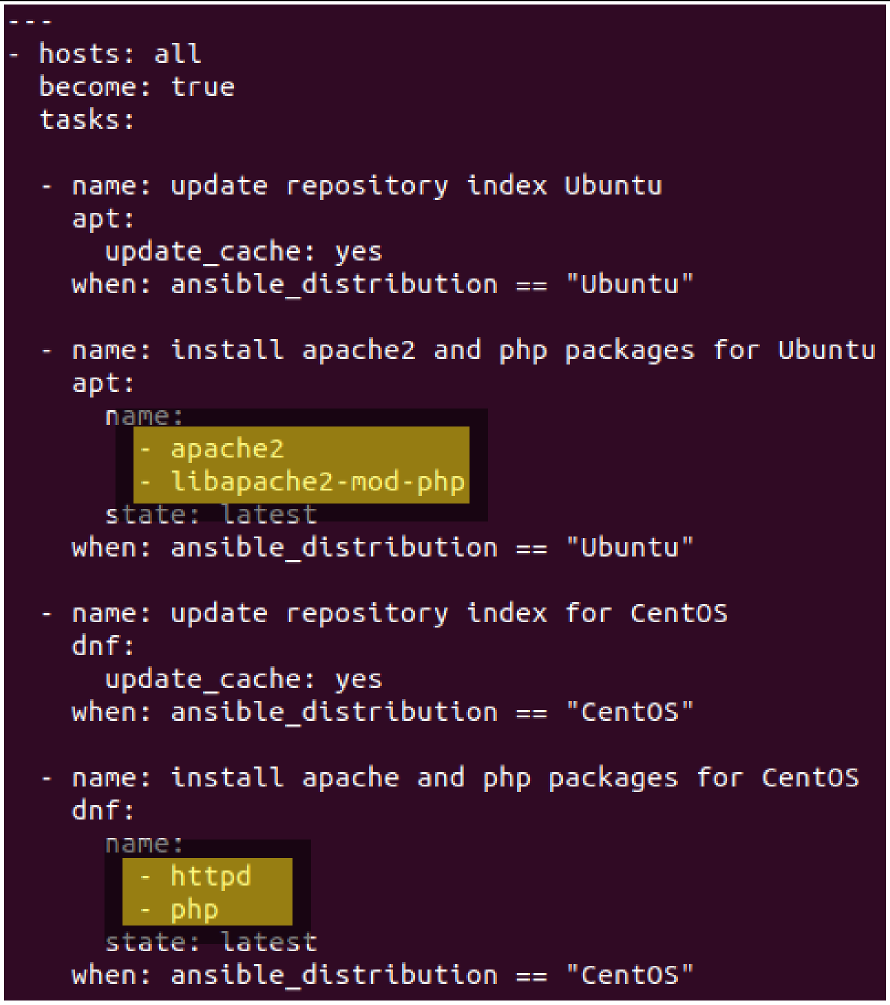
Testing
123..

This page is used to test the proper operation of the Apache 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.

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.

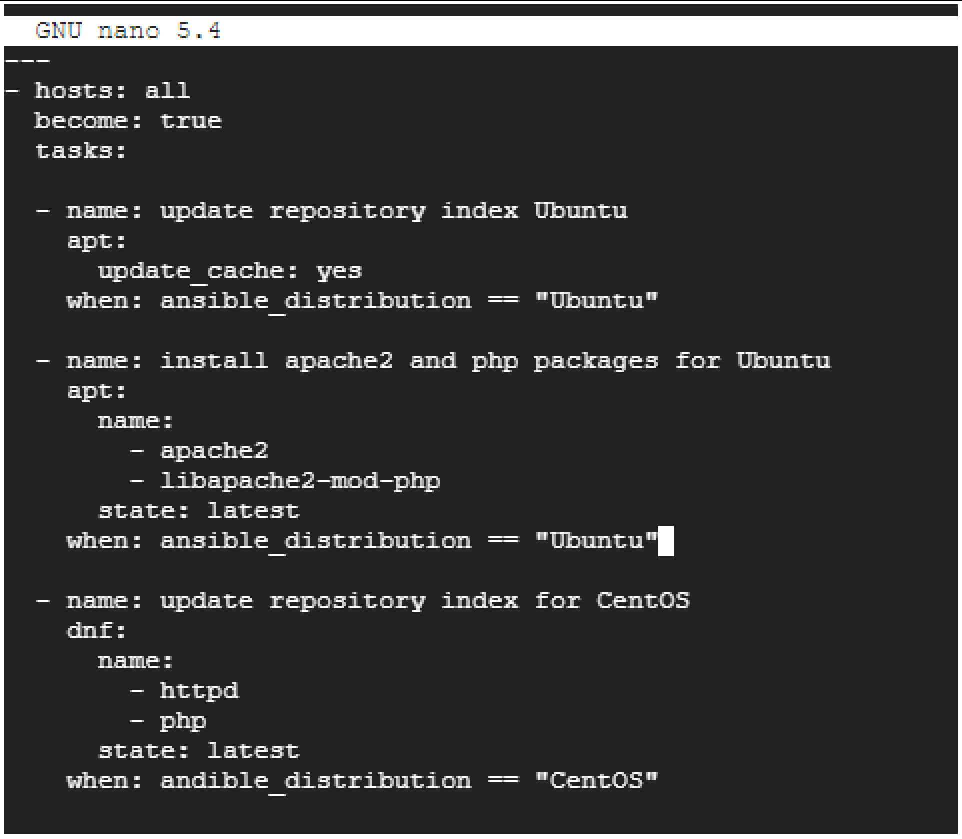
1. Edit the playbook *install_apache.yml*. Currently, we have three tasks targeting our Ubuntu machines and 3 tasks targeting our CentOS machine. Right now, we try to consolidate some tasks that are typically the same. For example, we can consolidate two plays that install packages. We can do that by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



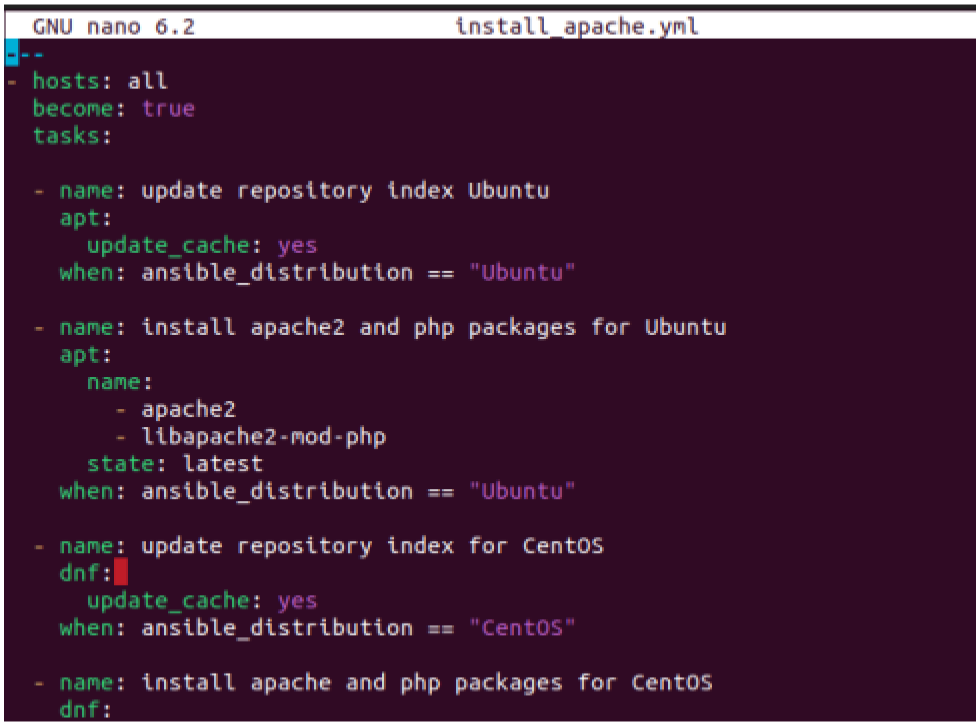
```
---
- hosts: all
  become: true
  tasks:

  - 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.

SCREENSHOT:



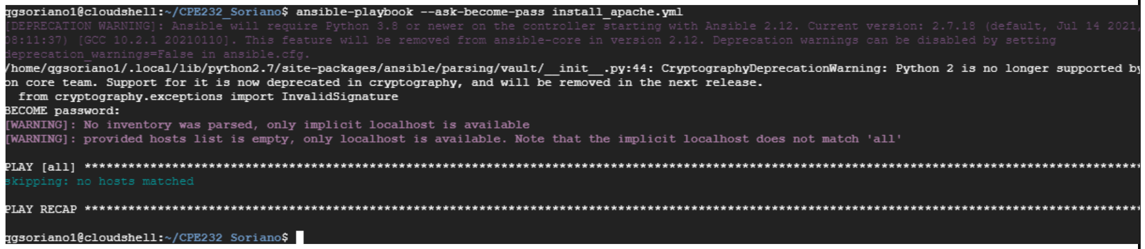
```
  GNU nano 5.4
---
- hosts: all
  become: true
  tasks:

  - 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:
      name:
        - httpd
        - php
      state: latest
    when: andible_distribution == "CentOS"
```



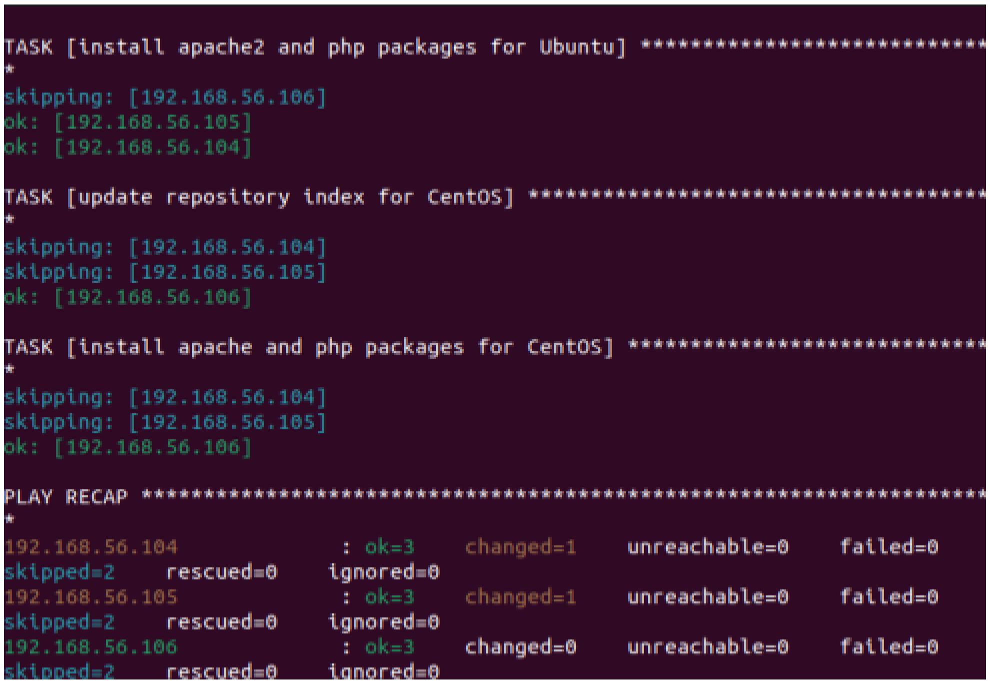
```
  GNU nano 6.2                   install_apache.yml
---
- hosts: all
  become: true
  tasks:

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

  - name: install apache2 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:
```

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

SCREENSHOT:



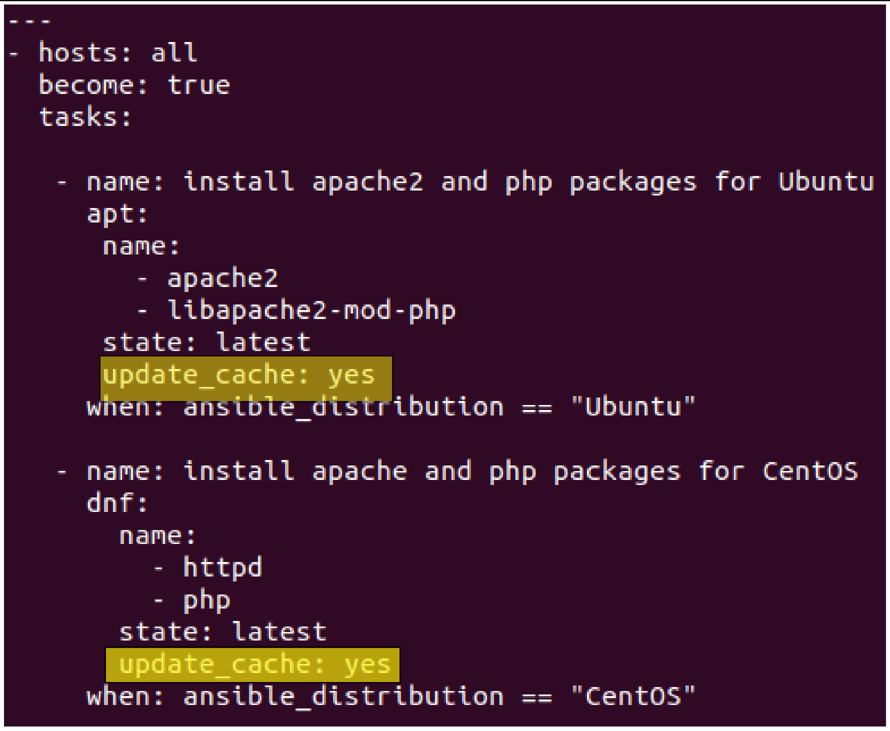
```
ggsoriano1@cloudshell:~/CPE232_Soriano$ ansible-playbook --ask-become-pass install_apache.yml
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with Ansible 2.12. Current version: 2.7.18 (default, Jul 14 2021,
08:11:37) [GCC 10.2.1 20210110]. This feature will be removed from ansible-core in version 2.12. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.
/home/qgsoriano1/.local/lib/python2.7/site-packages/ansible/parsing/vault/__init__.py:44: CryptographyDeprecationWarning: Python 2 is no longer supported by
on core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
  from cryptography.exceptions import InvalidSignature
BECOME password:
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [all] *****************************************************************************************************************
skipping: no hosts matched

PLAY RECAP *****************************************************************************************************************

ggsoriano1@cloudshell:~/CPE232_Soriano$
```



```
TASK [install apache2 and php packages for Ubuntu] ****************************
*
skipping: [192.168.56.106]
ok: [192.168.56.105]
ok: [192.168.56.104]

TASK [update repository index for CentOS] *************************************
*
skipping: [192.168.56.104]
skipping: [192.168.56.105]
ok: [192.168.56.106]

TASK [install apache and php packages for CentOS] *****************************
*
skipping: [192.168.56.104]
skipping: [192.168.56.105]
ok: [192.168.56.106]

PLAY RECAP ********************************************************************
*
192.168.56.104             : ok=3    changed=1    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.105             : ok=3    changed=1    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
192.168.56.106             : ok=3    changed=0    unreachable=0    failed=0
skipped=2    rescued=0    ignored=0
```

The execution of the command is good. Both of the servers have changed into centOS

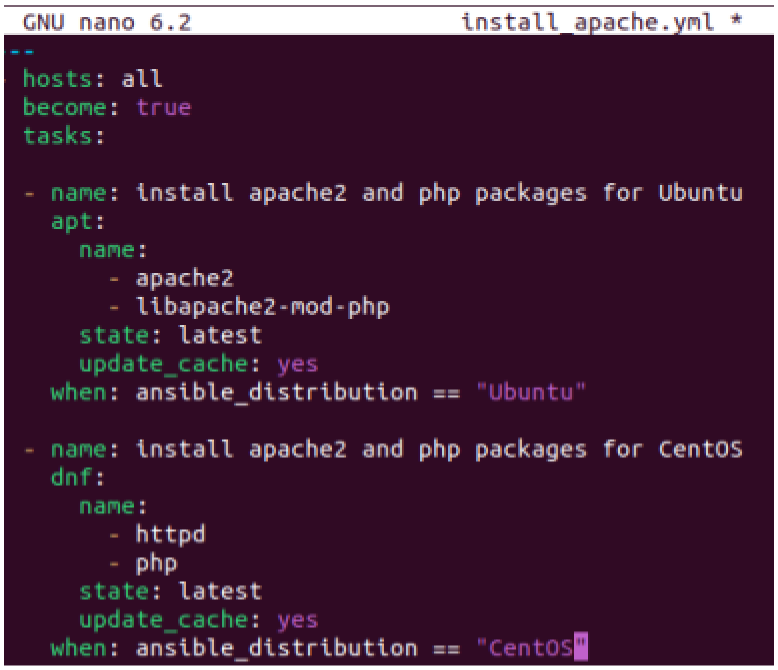
2. Edit the playbook *install_apache.yml* again. In task 2.1, we consolidated the plays into one play. This time we can actually consolidated everything in just 2 plays. This can be done by removing the update repository play and putting the command *update_cache: yes* below the command *state: latest*.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

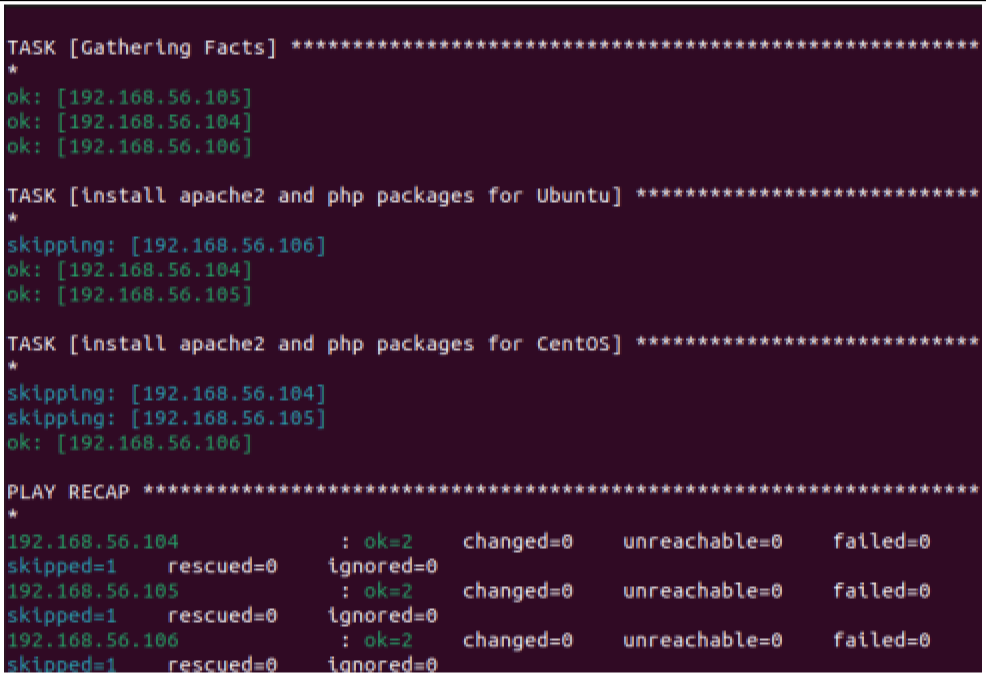
SCREENSHOT:



```
  GNU nano 6.2                   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 apache2 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.

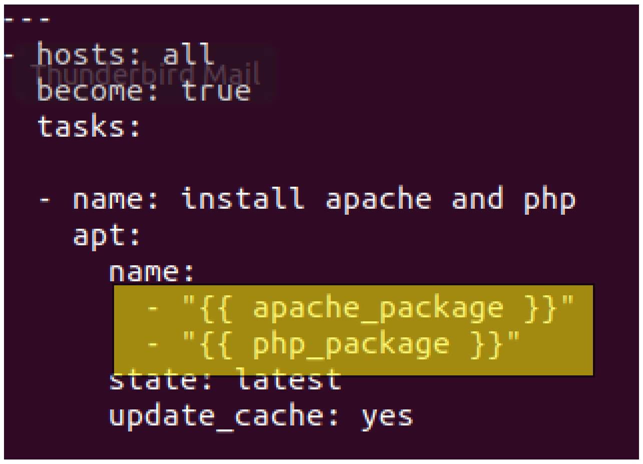
SCREENSHOT:



```
TASK [Gathering Facts] ***************************************************************
*
ok: [192.168.56.105]
ok: [192.168.56.104]
ok: [192.168.56.106]

TASK [install apache2 and php packages for Ubuntu] ***************************
*
skipping: [192.168.56.106]
ok: [192.168.56.104]
ok: [192.168.56.105]

TASK [install apache2 and php packages for CentOS] ***************************
*
skipping: [192.168.56.104]
skipping: [192.168.56.105]
ok: [192.168.56.106]

PLAY RECAP ***************************************************************************
*
192.168.56.104             : ok=2    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
192.168.56.105             : ok=2    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
192.168.56.106             : ok=2    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
```

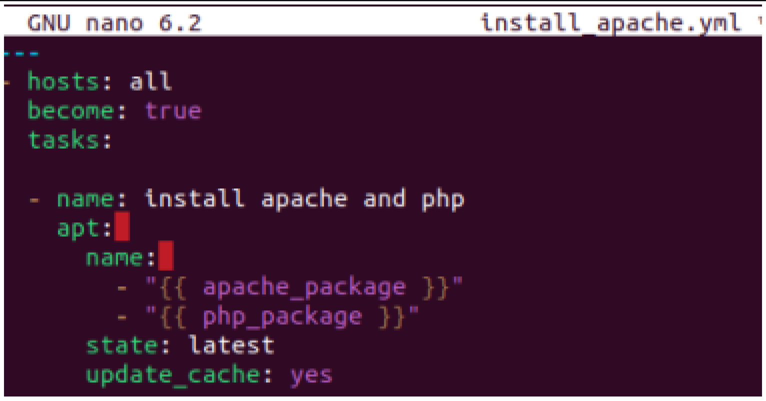
Commands are good and there are no errors encountered.

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
```

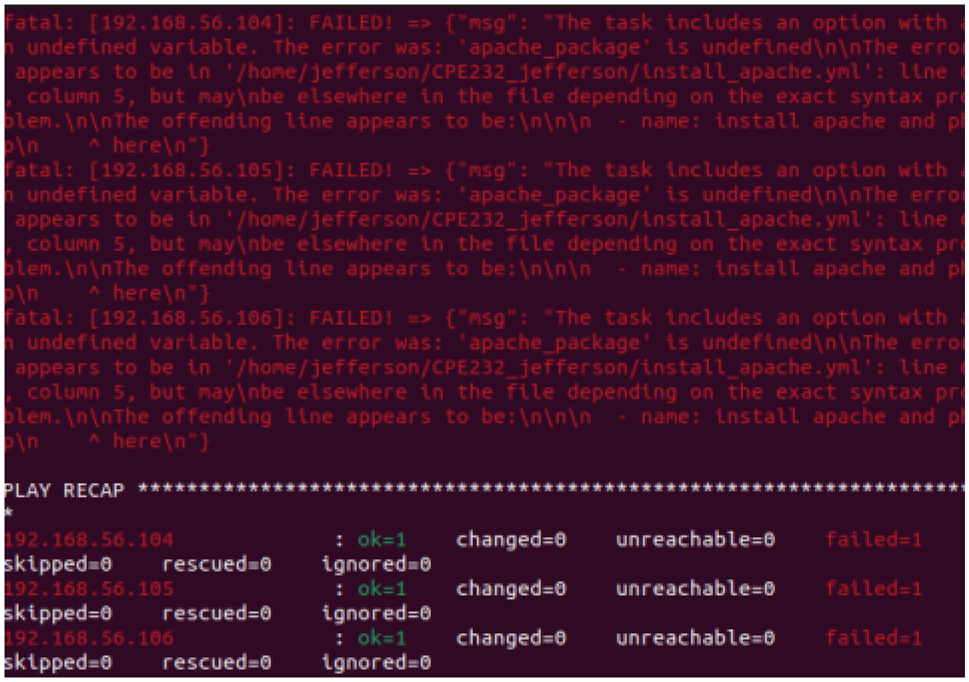
SCREENSHOT:



```
  GNU nano 6.2                          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.

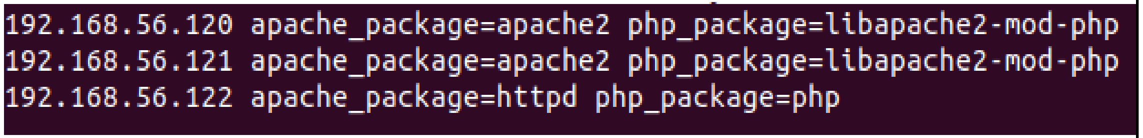
SCREENSHOT:



```
fatal: [192.168.56.104]: FAILED! => {"msg": "The task includes an option with a
n undefined variable. The error was: 'apache_package' is undefined\n\nThe erro
 appears to be in '/home/jefferson/CPE232_jefferson/install_apache.yml': line 
, column 5, but may\nbe elsewhere in the file depending on the exact syntax pr
blem.\n\nThe offending line appears to be:\n\n\n  - name: install apache and p
p\n    ^ here\n"}
fatal: [192.168.56.105]: FAILED! => {"msg": "The task includes an option with a
n undefined variable. The error was: 'apache_package' is undefined\n\nThe erro
 appears to be in '/home/jefferson/CPE232_jefferson/install_apache.yml': line 
, column 5, but may\nbe elsewhere in the file depending on the exact syntax pr
blem.\n\nThe offending line appears to be:\n\n\n  - name: install apache and p
p\n    ^ here\n"}
fatal: [192.168.56.106]: FAILED! => {"msg": "The task includes an option with a
n undefined variable. The error was: 'apache_package' is undefined\n\nThe erro
 appears to be in '/home/jefferson/CPE232_jefferson/install_apache.yml': line 
, column 5, but may\nbe elsewhere in the file depending on the exact syntax pr
blem.\n\nThe offending line appears to be:\n\n\n  - name: install apache and p
p\n    ^ here\n"}

PLAY RECAP ************************************************************************
*
192.168.56.104             : ok=1    changed=0    unreachable=0    failed=1    
skipped=0    rescued=0    ignored=0
192.168.56.105             : ok=1    changed=0    unreachable=0    failed=1    
skipped=0    rescued=0    ignored=0
192.168.56.106             : ok=1    changed=0    unreachable=0    failed=1    
skipped=0    rescued=0    ignored=0
```

There was an error encountered.

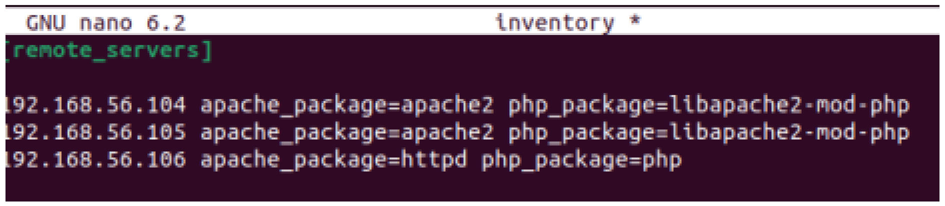
4. Unfortunately, task 2.3 was not successful. It's because we need to change something in the inventory file so that the variables we declared will be in place. Edit the *inventory* file and follow the below configuration:



```
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.

SCREENSHOT:

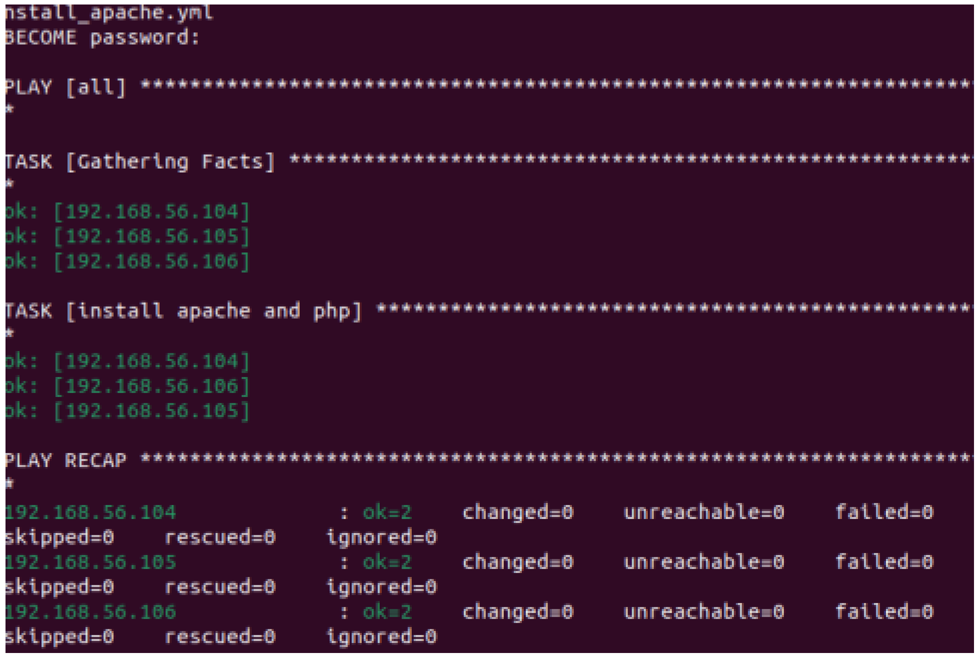


```
  GNU nano 6.2                         inventory *
[remote_servers]

192.168.56.104 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.105 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.106 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.

SCREENSHOT:



```
nstall_apache.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
*
ok: [192.168.56.104]
ok: [192.168.56.105]
ok: [192.168.56.106]

TASK [install apache and php] **************************************************
*
ok: [192.168.56.104]
ok: [192.168.56.106]
ok: [192.168.56.105]

PLAY RECAP *********************************************************************
*
192.168.56.104             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.105             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.106             : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

After some editing in the commands/codes, it is now successful.

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?
    - **Refactoring playbook codes is crucial since it makes it possible to It was shorter and simpler to memorize, which made it easier for the programmer. It additionally aids in accelerating the development of the code.**

2. When do we use the “when” command in playbook?
    - **The playbook's "when" command is used to control a variable's outcome. Additionally, it is utilized to establish connections with any ansible host regardless of the OS type.**

Honor Pledge

“I affirm that I shall not give or receive any unauthorized help on this assignment and that all work is my own.”