Name: Jessie Robert Lazo	Date Performed: 9/24/2024
Course/Section:CPE 212-CPE31S2	Date Submitted:9/26/2024
Instructor: Engr. Robin Valenzuela	Semester and SY:

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?

It means that there is no changes and the git pull command is successful.

```
jessie@jessie:~/CPE232_jessie$ git pull
Current branch master is up to date.
jessie@jessie:~/CPE232_jessie$
```

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 [Gathering Facts] ***************************************************
ok: [node2]
ok: [node1]
ok: [node3]

TASK [install apache2 package] *******************************************
[WARNING]: Updating cache and auto-installing missing dependency: python3-ap
fatal: [node3]: FAILED! => {"changed": false, "cmd": "apt-get update", "msg"
Errno 2] No such file or directory: b'apt-get'", "rc": 2}
ok: [node1]
ok: [node2]

PLAY RECAP ***************************************************************
node1                      : ok=2    changed=0    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0
node2                      : ok=2    changed=0    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0
node3                      : ok=1    changed=0    unreachable=0    failed=1
kipped=0    rescued=0    ignored=0

jessie@jessie:~/CPE232_jessie$
```

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

```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
    when: ansible_distribution == "Ubuntu"

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

jessie@jessie: ~/CPE232_jessie

GNU nano 6.2 install_apache.yml

```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
    when: ansible_distribution == "Ubuntu"

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

Make sure to save the file and exit.

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

```
jessie@jessie:~/CPE232_jessie$ ansible-playbook --ask-become-pass install_apa
_yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [node2]
ok: [node1]
ok: [node3]

TASK [update repository index] **************************************************
skipping: [node3]
changed: [node2]
changed: [node1]

TASK [install apache2 package] **************************************************
skipping: [node3]
ok: [node1]
ok: [node2]

TASK [add PHP support for apache] ***********************************************
skipping: [node3]
changed: [node2]
changed: [node1]

PLAY RECAP *********************************************************************
node1                      : ok=4    changed=2    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0
node2                      : ok=4    changed=2    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0
node3                      : ok=1    changed=0    unreachable=0    failed=0
kipped=3    rescued=0    ignored=0

jessie@jessie:~/CPE232_jessie$
```

centos node was skipped since the tasks are only for ubuntu servers.

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"
```

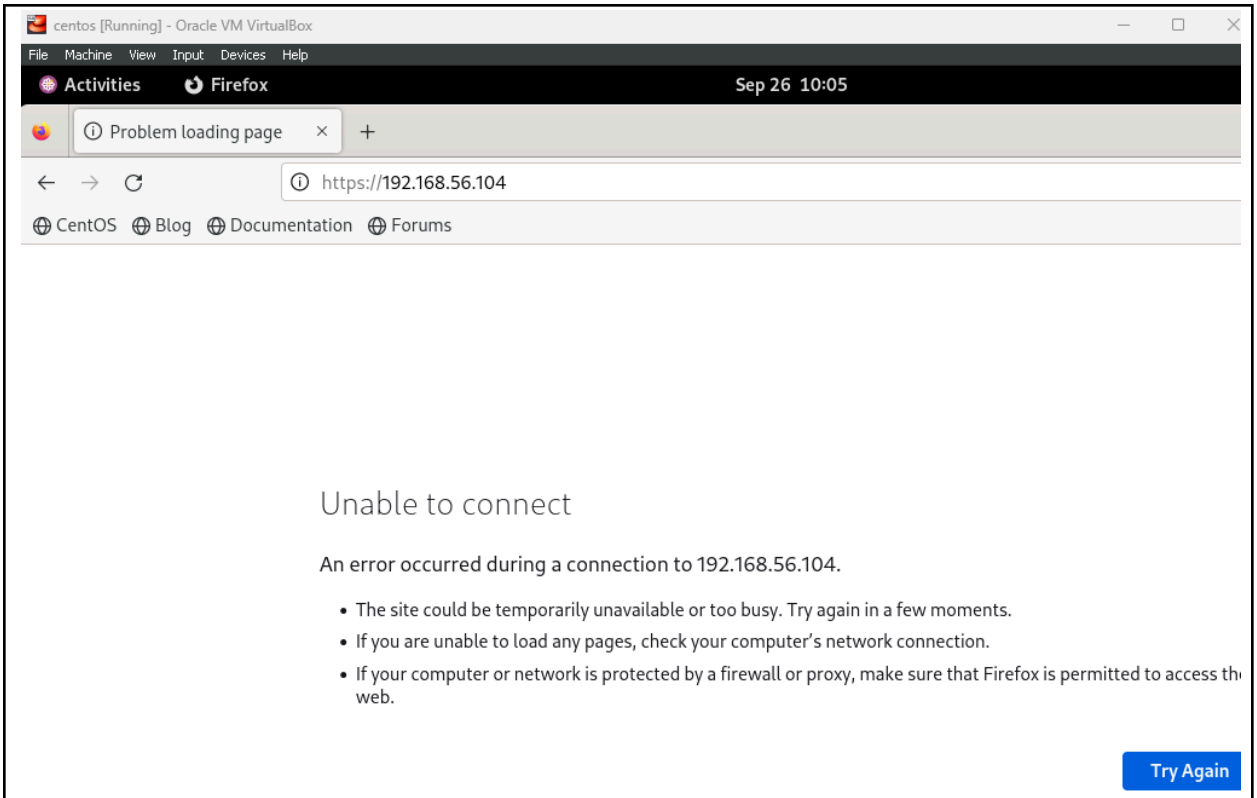
Make sure to save and exit.

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

```
TASK [install apache2 package] *******************************************
skipping: [node3]
ok: [node2]
ok: [node1]

TASK [add PHP support for apache] ****************************************
skipping: [node3]
ok: [node1]
ok: [node2]

TASK [update repository index] ********************************************
skipping: [node1]
skipping: [node2]
changed: [node3]

TASK [install apache2 package] *******************************************
skipping: [node1]
skipping: [node2]
changed: [node3]

TASK [add PHP support for apache] ****************************************
skipping: [node1]
skipping: [node2]
changed: [node3]

PLAY RECAP ****************************************************************
node1                      : ok=4    changed=1    unreachable=0    failed=0
kipped=3    rescued=0    ignored=0
node2                      : ok=4    changed=1    unreachable=0    failed=0
kipped=3    rescued=0    ignored=0
node3                      : ok=4    changed=3    unreachable=0    failed=0
kipped=3    rescued=0    ignored=0

jessie@jessie:~/CPE232_jessie$
```

the result is now successful for both ubuntu and centos

5. To verify the installations, go to CentOS VM and type its IP address on the browser. Was it successful? The answer is no. It's because the httpd service or the Apache HTTP server in the CentOS is not yet active. Thus, you need to activate it first.



5.1 To activate, go to the CentOS VM terminal and enter the following:

systemctl status httpd

The result of this command tells you that the service is inactive.

```
[jlazo@localhost yum.repos.d]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset
    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)

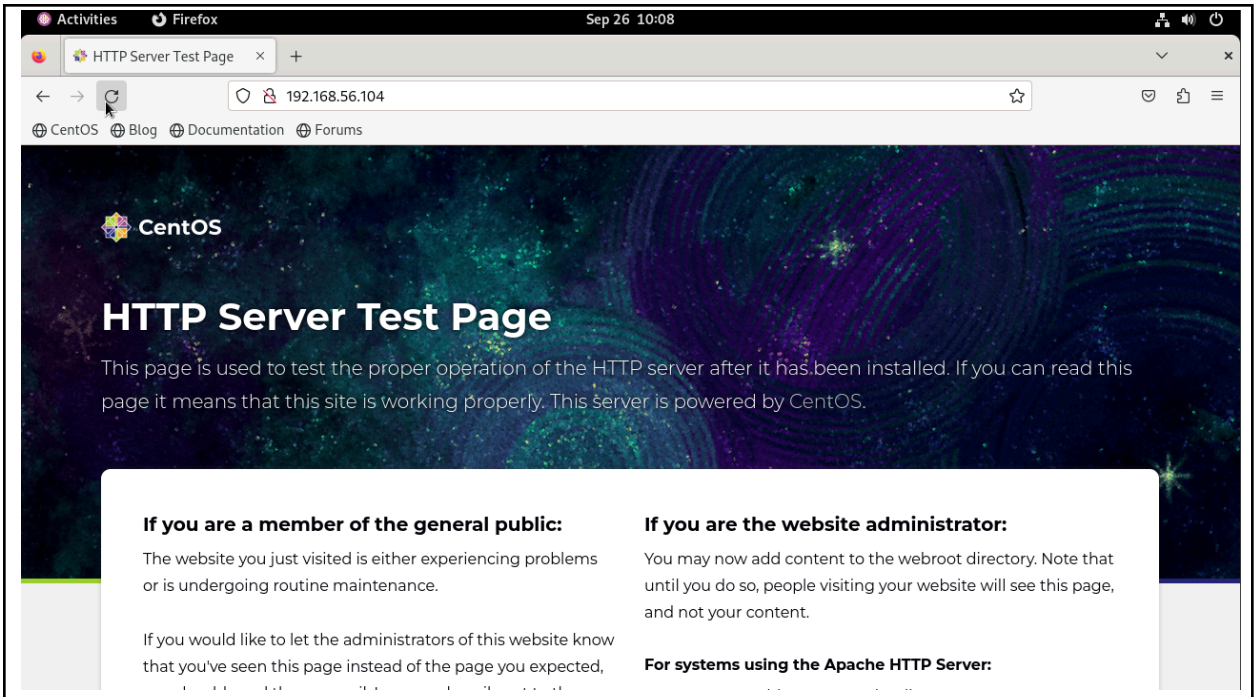
sudo firewall-cmd --add-port=80/tcp

(The result should be a success)

```
[jlazo@localhost ~]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; pre
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: active (running) since Thu 2024-09-26 10:06:36 PST; 8s ago
       Docs: man:httpd.service(8)
   Main PID: 80710 (httpd)
     Status: "Started, listening on: port 80"
      Tasks: 177 (limit: 10963)
     Memory: 20.2M
        CPU: 125ms
     CGroup: /system.slice/httpd.service
             ├─80710 /usr/sbin/httpd -DFOREGROUND
             ├─80717 /usr/sbin/httpd -DFOREGROUND
             ├─80718 /usr/sbin/httpd -DFOREGROUND
             ├─80719 /usr/sbin/httpd -DFOREGROUND
             └─80720 /usr/sbin/httpd -DFOREGROUND

Sep 26 10:06:36 localhost.localdomain systemd[1]: Starting The Apache HTT
Sep 26 10:06:36 localhost.localdomain httpd[80710]: AH00558: httpd: Coulc
Sep 26 10:06:36 localhost.localdomain httpd[80710]: Server configured, li
Sep 26 10:06:36 localhost.localdomain systemd[1]: Started The Apache HTTP
lines 1-22/22 (END)
[jlazo@localhost ~]$
[jlazo@localhost ~]$ sudo firewall-cmd --add-port=80/tcp
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)

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.

jessie@jessie: ~/CPE232_jessie

GNU nano 6.2 install_apache.yml

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

  - name: update repository index
    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.

```
jessie@jessie: ~/CPE232_jessie
jessie@jessie:~/CPE232_jessie$ ansible-playbook --ask-become-pass install_apa
_yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [node2]
ok: [node1]
ok: [node3]

TASK [update repository index] *************************************************
skipping: [node3]
changed: [node2]
changed: [node1]

TASK [install apache2 and php packages for Ubuntu] *****************************
skipping: [node3]
ok: [node1]
ok: [node2]

TASK [update repository index for CentOS] **************************************
skipping: [node1]
skipping: [node2]
ok: [node3]

TASK [install apache and php packages for CentOS] ******************************
skipping: [node1]
skipping: [node2]
ok: [node3]

PLAY RECAP *********************************************************************
node1                      : ok=3    changed=1    unreachable=0    failed=0
kipped=2    rescued=0    ignored=0
node2                      : ok=3    changed=1    unreachable=0    failed=0
kipped=2    rescued=0    ignored=0
node3                      : ok=3    changed=0    unreachable=0    failed=0
kipped=2    rescued=0    ignored=0

jessie@jessie:~/CPE232_jessie$
```

the installation of apache and php packages are merged into 1 which lessen the tasks. I think this is much faster compared to the initial installation.

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.

```
jessie@jessie: ~/CPE232_jessie
GNU nano 6.2                    install_apache.yml
---
- hosts: all
  become: true
  tasks:

    - name: update repository index, install apache2 and php packages for Ub
      apt:
        name:
          - apache2
          - libapache2-mod-php
        state: latest
        update_cache: yes
      when: ansible_distribution == "Ubuntu"

    - name: update repository index, install apache and php packages for Cen
      dnf:
        name:
          - httpd
          - php
        state: latest
        update_cache: yes
      when: ansible_distribution == "CentOS"

                                [ Read 23 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 Li
```

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

```
jessie@jessie:~/CPE232_jessie$ ansible-playbook --ask-become-pass install_a
_yml
BECOME password:

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

TASK [Gathering Facts] ***********************************************
ok: [node2]
ok: [node1]
ok: [node3]

TASK [update repository index, install apache2 and php packages for Ubuntu]
skipping: [node3]
ok: [node2]
ok: [node1]

TASK [update repository index, install apache and php packages for CentOS]
skipping: [node1]
skipping: [node2]
ok: [node3]

PLAY RECAP ***********************************************************
node1                      : ok=2    changed=0    unreachable=0    failed=0
kipped=1    rescued=0    ignored=0
node2                      : ok=2    changed=0    unreachable=0    failed=0
kipped=1    rescued=0    ignored=0
node3                      : ok=2    changed=0    unreachable=0    failed=0
kipped=1    rescued=0    ignored=0
```

The result is much simpler with less tasks and much faster results.

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
```

```
jessie@jessie: ~/CPE232_jessie
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

                          [ Wrote 12 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.

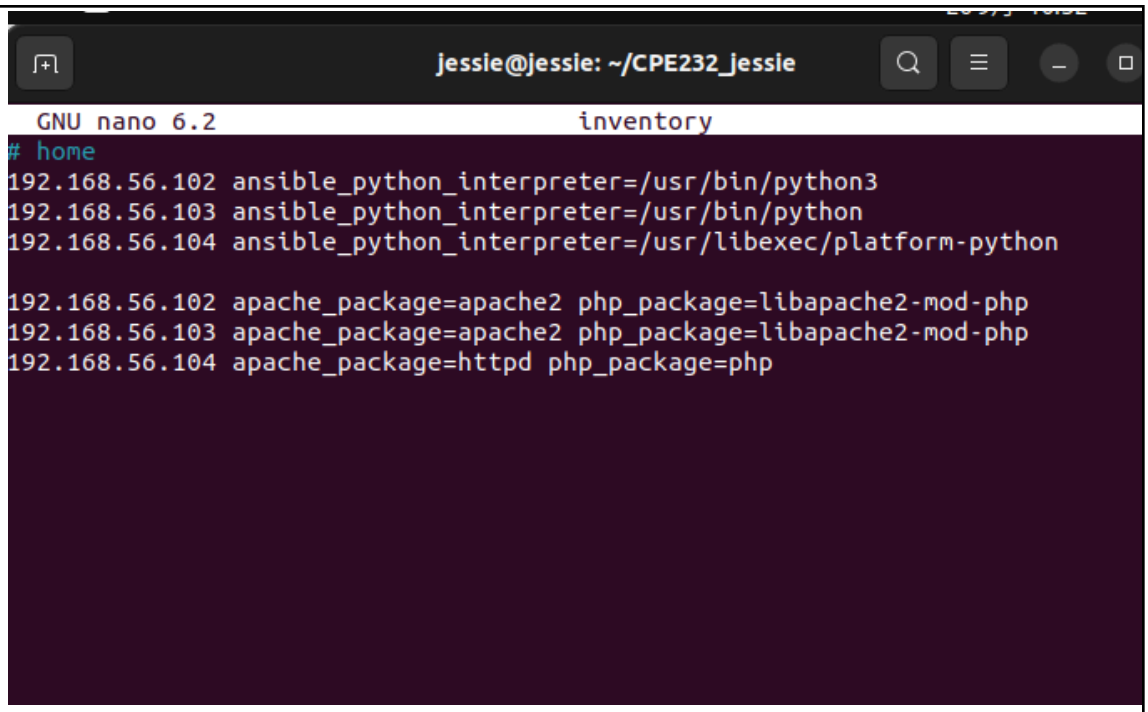
```
jessie@jessie:~/CPE232_jessie$ ansible-playbook --ask-become-pass install_a
_yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [node2]
ok: [node1]
ok: [node3]

TASK [install apache and php] **************************************************
fatal: [node1]: FAILED! => {"msg": "The task includes an option with an unc
d variable. The error was: 'apache_package' is undefined\n\nThe error appea
 be in '/home/jessie/CPE232_jessie/install_apache_yml': line 6, column 7, b
y\nbe elsewhere in the file depending on the exact syntax problem.\n\nThe c
ing line appears to be:\n\n\n    - name: install apache and php\n      ^ he

fatal: [node2]: FAILED! => {"msg": "The task includes an option with an unc
d variable. The error was: 'apache_package' is undefined\n\nThe error appea
 be in '/home/jessie/CPE232_jessie/install_apache_yml': line 6, column 7, b
y\nbe elsewhere in the file depending on the exact syntax problem.\n\nThe c
ing line appears to be:\n\n\n    - name: install apache and php\n      ^ he

fatal: [node3]: FAILED! => {"msg": "The task includes an option with an unc
d variable. The error was: 'apache_package' is undefined\n\nThe error appea
 be in '/home/jessie/CPE232_jessie/install_apache_yml': line 6, column 7, b
y\nbe elsewhere in the file depending on the exact syntax problem.\n\nThe c
ing line appears to be:\n\n\n    - name: install apache and php\n      ^ he

PLAY RECAP *********************************************************************
node1                      : ok=1    changed=0    unreachable=0    failed=1
skipped=0    rescued=0    ignored=0
node2                      : ok=1    changed=0    unreachable=0    failed=1
skipped=0    rescued=0    ignored=0
node3                      : ok=1    changed=0    unreachable=0    failed=1
skipped=0    rescued=0    ignored=0

jessie@jessie:~/CPE232_jessie$
```

failed because of an undefined variable.

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.

```
11:12
jessie@jessie: ~/CPE232_jessie
 install_apache_yml
BECOME password:

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

TASK [Gathering Facts] ***************************************
***************
ok: [node2]
ok: [node1]
ok: [node3]

TASK [Install Apache and PHP on Ubuntu] **********************
***************
skipping: [node3]
ok: [node2]
ok: [node1]

TASK [Install Apache and PHP on CentOS] **********************
***************
skipping: [node1]
skipping: [node2]
ok: [node3]

TASK [Ensure Apache is running] ******************************
***************
ok: [node1]
ok: [node2]
changed: [node3]

PLAY RECAP ***************************************************
***************
node1                      : ok=3    changed=0    unreachable=0
  failed=0    skipped=1    rescued=0    ignored=0
node2                      : ok=3    changed=0    unreachable=0
  failed=0    skipped=1    rescued=0    ignored=0
node3                      : ok=3    changed=1    unreachable=0
  failed=0    skipped=1    rescued=0    ignored=0

jessie@jessie:~/CPE232_jessie$
```

Supplementary Activity:

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

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?

   -Refactoring improves the clarity of the code, making it easier for others to understand the playbook's intent and structure.

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

   - we use “when” command to adapt to various scenarios effectively. For example in this activity, “when” command checks the OS type of the hosts to ensure that the appropriate package manager and package names are used for each OS.