

Name: Ferrer, Joseph Bryan M.	Date Performed: 09/18/2023
Course/Section: CPE31S6	Date Submitted: 09/21/2023
Instructor: Dr. Jonathan V. Taylar	Semester and SY: 1st Semester / 2023-2024

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.

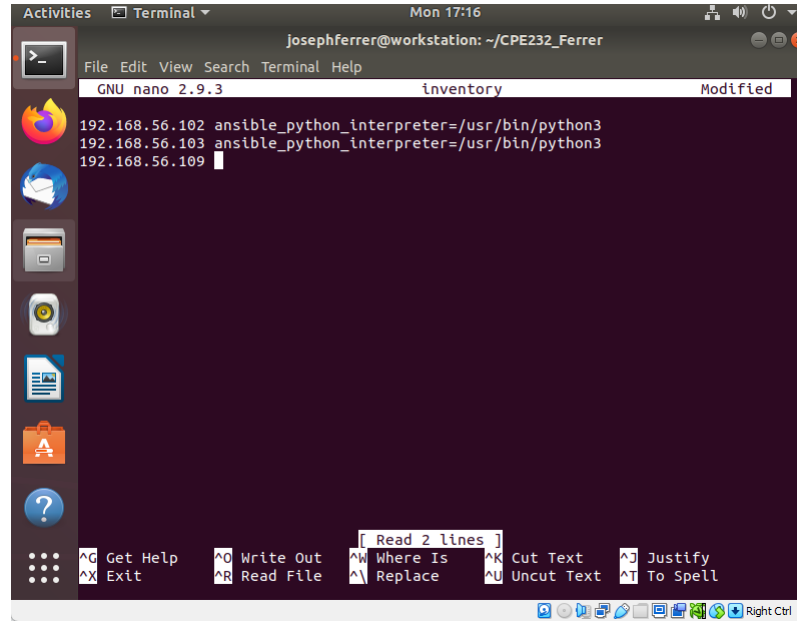
Task 1: Use when command for different distributions

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?

```
josephferrer@workstation:~/CPE232_Ferrer$ git pull
Already up to date.
```

It says that github is already up to date because I didn't do any changes from my Ubuntu terminal.

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."



```
Josephferrer@workstation: ~/CPE232_Ferrer
GNU nano 2.9.3 inventory Modified
192.168.56.102 ansible_python_interpreter=/usr/bin/python3
192.168.56.103 ansible_python_interpreter=/usr/bin/python3
192.168.56.109
```

```
Josephferrer@workstation:~/CPE232_Ferrer$ ansible-playbook --ask-become-pass in
stall_apache.yml
BECOME password:

PLAY [all] *****
*

TASK [Gathering Facts] *****
*
ok: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [update repository index] *****
*
[WARNING]: Updating cache and auto-installing missing dependency: python-apt
fatal: [192.168.56.109]: FAILED! => {"changed": false, "cmd": "apt-get update",
  "msg": "[Errno 2] No such file or directory", "rc": 2, "stderr": "", "stderr_l
  ines": [], "stdout": "", "stdout_lines": []}
changed: [192.168.56.103]
changed: [192.168.56.102]

TASK [install apache2 package] *****
*
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [add PHP support for apache] *****
```

```

*
[WARNING]: Updating cache and auto-installing missing dependency: python-apt
Fatal: [192.168.56.109]: FAILED! => {"changed": false, "cmd": "apt-get update",
  "msg": "[Errno 2] No such file or directory", "rc": 2, "stderr": "", "stderr_l
  nes": [], "stdout": "", "stdout_lines": []}
changed: [192.168.56.103]
changed: [192.168.56.102]

TASK [install apache2 package] *****
*
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [add PHP support for apache] *****
*
ok: [192.168.56.103]
ok: [192.168.56.102]

PLAY RECAP *****
*
192.168.56.102      : ok=4    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103      : ok=4    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.109      : ok=1    changed=0    unreachable=0    failed=1
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.

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

```
josephferrer@workstation: ~/CPE232_Ferrer
File Edit View Search Terminal Help
GNU nano 2.9.3      install_apache.yml      Modified
---
- 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"
```

```
josephferrer@workstation:~/CPE232_Ferrer$ ansible-playbook --ask-become-pass in
stall_apache.yml
BECOME password:

PLAY [all] *****
*

TASK [Gathering Facts] *****
*
ok: [192.168.56.109]
ok: [192.168.56.103]
ok: [192.168.56.102]

TASK [update repository index] *****
*
skipping: [192.168.56.109]
changed: [192.168.56.102]
changed: [192.168.56.103]

TASK [install apache2 package] *****
*
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [add PHP support for apache] *****
*
skipping: [192.168.56.109]
ok: [192.168.56.102]
```

The command will be successful however it will skip the ip address of CentOS since apt is not applicable for CentOS.

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  
        state: latest  
        when: ansible_distribution == "Ubuntu"  
  
    - name: add PHP support for apache  
      apt:  
        name: libapache2-mod-php  
        state: latest  
        when: ansible_distribution == "Ubuntu"  
  
    - name: update repository index  
      dnf:  
        update_cache: yes  
        when: ansible_distribution == "CentOS"  
  
    - name: install apache2 package  
      dnf:  
        name: httpd  
        state: latest  
        when: ansible_distribution == "CentOS"  
  
    - name: add PHP support for apache  
      dnf:  
        name: php  
        state: latest  
        when: ansible_distribution == "CentOS"
```

Make sure to save and exit.

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

```
josephferrer@workstation: ~/CPE232_Ferrer
File Edit View Search Terminal Help
GNU nano 2.9.3 install_apache.yml Modified

- 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: httpd
    state: latest
    when: ansible_distribution == "CentOS"

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

```
josephferrer@workstation:~/CPE232_Ferrer$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [update repository index] *****
skipping: [192.168.56.109]
changed: [192.168.56.103]
changed: [192.168.56.102]

TASK [install apache2 package] *****
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [add PHP support for apache] *****
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [update repository index] *****
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.109]

TASK [install apache2 package] *****
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.109]
```

```

TASK [install apache2 package] *****
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [add PHP support for apache] *****
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [update repository index] *****
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.109]

TASK [install apache2 package] *****
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.109]

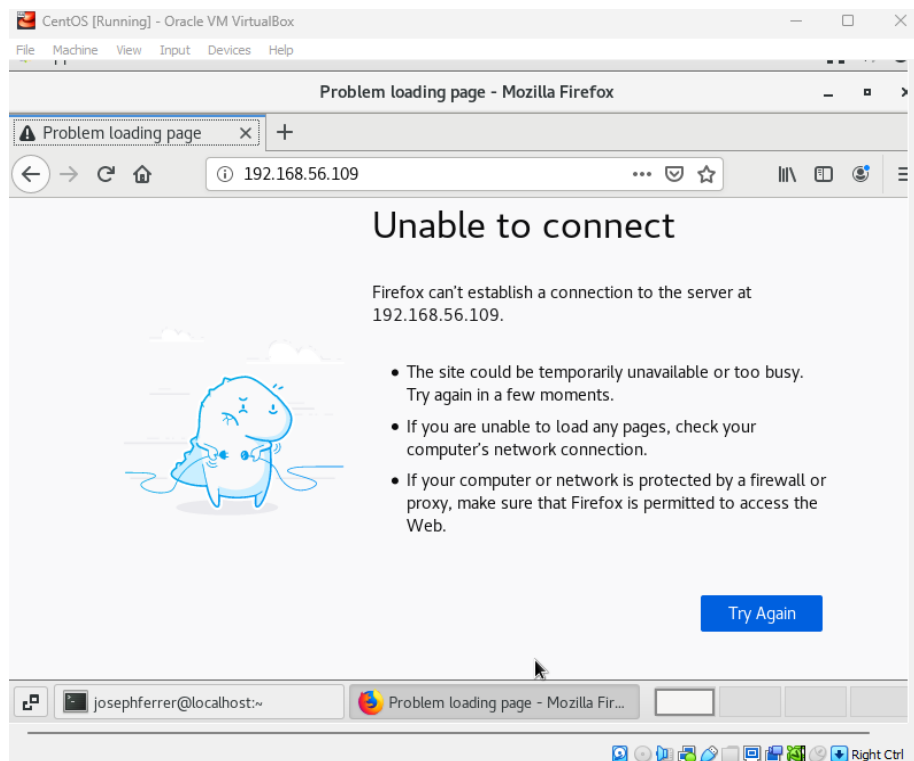
TASK [add PHP support for apache] *****
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.109]

PLAY RECAP *****
192.168.56.102      : ok=4    changed=1    unreachable=0    failed=0    skipped=3    rescued=0
ignored=0
192.168.56.103      : ok=4    changed=1    unreachable=0    failed=0    skipped=3    rescued=0
ignored=0
192.168.56.109      : ok=4    changed=0    unreachable=0    failed=0    skipped=3    rescued=0
ignored=0

```

The command will be successful since we use the dnf package manager in our playbook.

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.

```
[josephferrer@localhost ~]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:httpd(8)
          man:apachectl(8)
```

5.2 Issue the following command to start the service:

sudo systemctl start httpd

(When prompted, enter the sudo password)

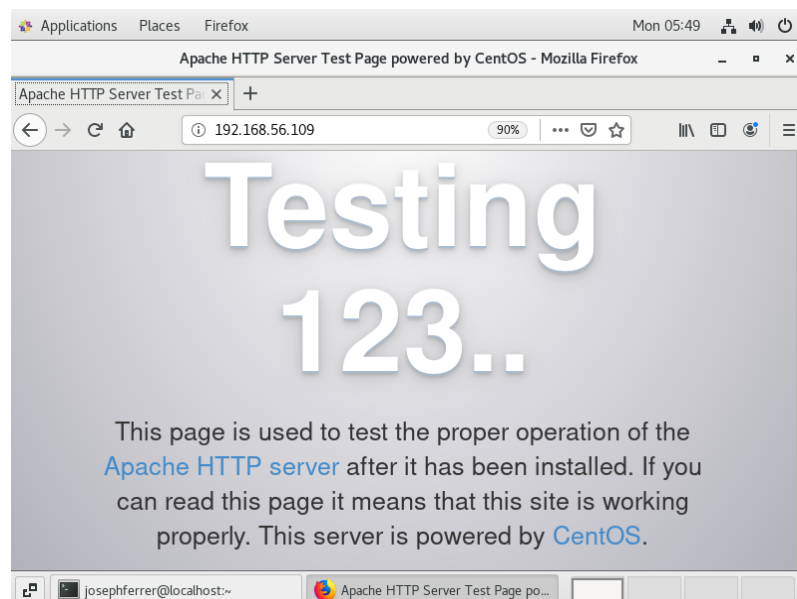
```
[josephferrer@localhost ~]$ sudo systemctl start httpd
[sudo] password for josephferrer:
```

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

(The result should be a success)

```
[josephferrer@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)



Yes it is 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.

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

```
josephferrer@workstation: ~/CPE232_Ferrer
File Edit View Search Terminal Help
GNU nano 2.9.3 install_apache.yml
---
- hosts: all
  become: true
  tasks:
    - name: update repository index
      apt:
        update_cache: yes
      when: ansible_distribution == "Ubuntu"

    - name: install apache2 package 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"
```

```
josephferrer@workstation:~/CPE232_Ferrer$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.103]
ok: [192.168.56.102]

TASK [update repository index] *****
skipping: [192.168.56.109]
changed: [192.168.56.103]
changed: [192.168.56.102]

TASK [install apache2 package and php packages for Ubuntu] *****
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [update repository index for CentOS] *****
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.109]

TASK [install apache and php packages for CentOS] *****
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.109]

PLAY RECAP *****
192.168.56.102      : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.103      : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.109      : ok=3    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```

The tasks are now refactored since we combine the task of installing apache2 and php packages for both Ubuntu and 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.

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

```
Josephferrer@workstation: ~/CPE232_Ferrer
File Edit View Search Terminal Help
GNU nano 2.9.3 install_apache.yml

---
- hosts: all
  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 apache and php packages for CentOS
      dnf:
        name:
          - httpd
          - php
        state: latest
        update_cache: yes
      when: ansible_distribution == "CentOS"
```

```

josephferrer@workstation:~/CPE232_Ferrer$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.102]
ok: [192.168.56.103]
ok: [192.168.56.109]

TASK [install apache2 package and php packages for Ubuntu] *****
skipping: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [install apache and php packages for CentOS] *****
skipping: [192.168.56.102]
skipping: [192.168.56.103]
ok: [192.168.56.109]

PLAY RECAP *****
192.168.56.102      : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0
    ignored=0
192.168.56.103      : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0
    ignored=0
192.168.56.109      : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0
    ignored=0

```

From 5 tasks, we now have 3 tasks since we refactor it again. The playbook runs successfully.

- 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.

```
josephferrer@workstation: ~/CPE232_Ferrer
File Edit View Search Terminal Help
GNU nano 2.9.3 install_apache.yml

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

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

josephferrer@workstation:~/CPE232_Ferrer$ ansible-playbook --ask-become-pass install_apache.yml
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.102]
ok: [192.168.56.103]

TASK [install apache and php] *****
fatal: [192.168.56.102]: FAILED! => {"msg": "The task includes an option with an undefined variable. T
he error was: 'apache_package' is undefined\n\nThe error appears to be in '/home/josephferrer/CPE232_F
errer/install_apache.yml': line 6, column 5, but may\nbe elsewhere in the file depending on the exact
syntax problem.\n\nThe offending line appears to be:\n\n    - name: install apache and php\n      ^ here
\n"}
fatal: [192.168.56.103]: FAILED! => {"msg": "The task includes an option with an undefined variable. T
he error was: 'apache_package' is undefined\n\nThe error appears to be in '/home/josephferrer/CPE232_F
errer/install_apache.yml': line 6, column 5, but may\nbe elsewhere in the file depending on the exact
syntax problem.\n\nThe offending line appears to be:\n\n    - name: install apache and php\n      ^ here
\n"}
fatal: [192.168.56.109]: FAILED! => {"msg": "The task includes an option with an undefined variable. T
he error was: 'apache_package' is undefined\n\nThe error appears to be in '/home/josephferrer/CPE232_F
errer/install_apache.yml': line 6, column 5, but may\nbe elsewhere in the file depending on the exact
syntax problem.\n\nThe offending line appears to be:\n\n    - name: install apache and php\n      ^ here
\n"}

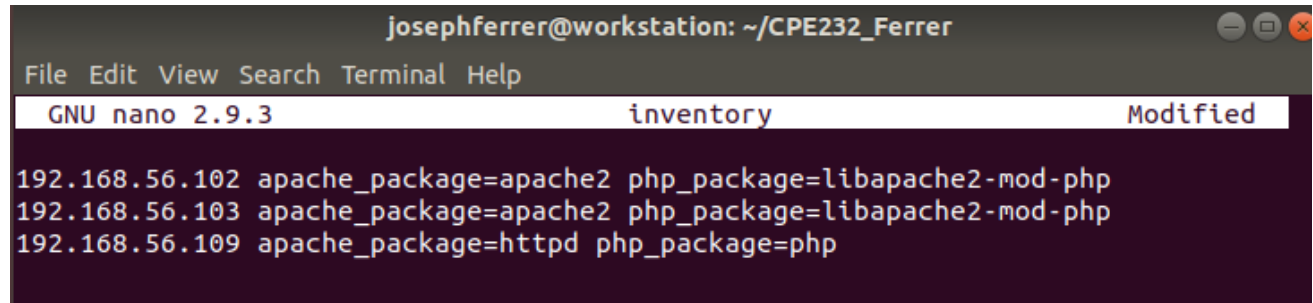
PLAY RECAP *****
192.168.56.102      : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0
ignored=0
192.168.56.103      : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0
ignored=0
192.168.56.109      : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0
ignored=0
```

The result will be failure because the task variable is undefined. We should declare it in our inventory file in order to run the playbook.

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.

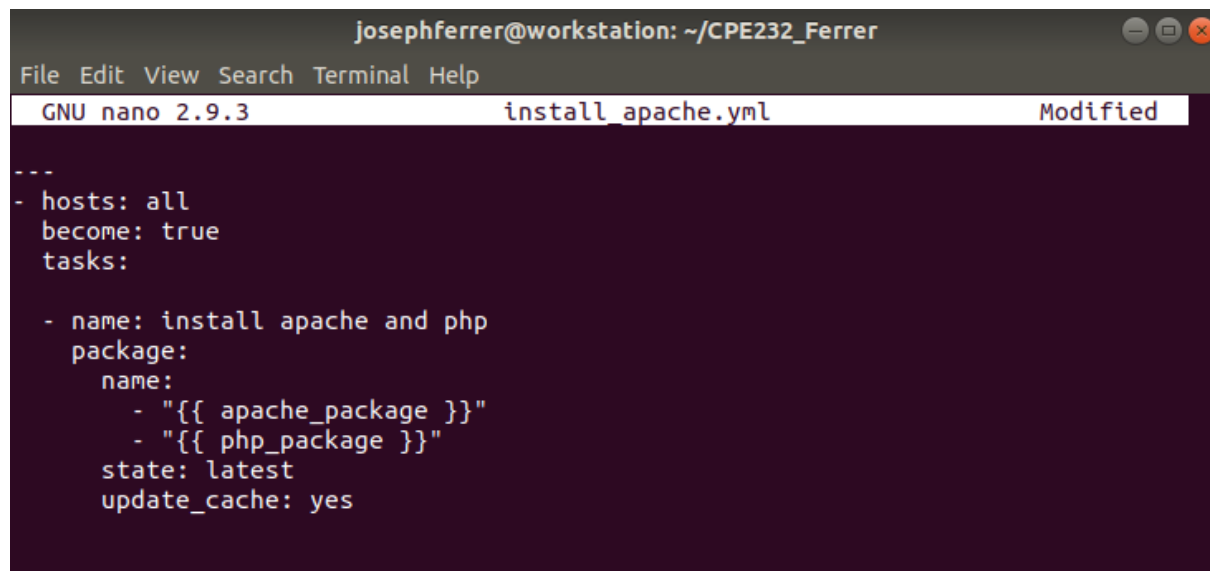


The screenshot shows a terminal window titled 'josephferrer@workstation: ~/CPE232_Ferrer'. Inside, the nano editor is open to a file named 'inventory'. The editor's status bar shows 'GNU nano 2.9.3' and 'Modified'. The content of the file is:

```
192.168.56.102 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.103 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.109 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](https://docs.ansible.com/ansible/latest/modules/package_module.html)

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



The screenshot shows a terminal window titled 'josephferrer@workstation: ~/CPE232_Ferrer'. Inside, the nano editor is open to a file named 'install_apache.yml'. The editor's status bar shows 'GNU nano 2.9.3' and 'Modified'. The content of the file is:

```
---
- hosts: all
  become: true
  tasks:
    - name: install apache and php
      package:
        name:
          - "{{ apache_package }}"
          - "{{ php_package }}"
        state: latest
        update_cache: yes
```

```

josephferrer@workstation:~/CPE232_Ferrer$ ansible-playbook --ask-become-pass in
stall_apache.yml
BECOME password:

PLAY [all] *****
*

TASK [Gathering Facts] *****
*
ok: [192.168.56.109]
ok: [192.168.56.103]
ok: [192.168.56.102]

TASK [install apache and php] *****
*
ok: [192.168.56.109]
ok: [192.168.56.103]
ok: [192.168.56.102]

PLAY RECAP *****
*
192.168.56.102      : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.103      : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
192.168.56.109      : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0

```

The playbook will now be successful from Server 1, Server 2, and CentOS. The tasks are now successfully refactored.

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?
 - There are a number of reasons why it's important to refactor playbook codes. First, it will help to improve readability of the code so that you can understand and maintain it more easily. In addition, it would help to decrease the number of code that has to be coded so as to cut down on time and effort. Thirdly, it can help to improve the code's performance and speed up its operation. In addition, it may help to reduce the number of bugs and errors in a code which can improve its overall quality.

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

- The When command in a playbook is used if you want to perform multiple tasks or objectives, depending on values of facts, variables and results from an earlier task. It will allow you to set up certain conditions for the flow of executions within your playbook. For example, if you want to check whether a task is required before you run it, you can use the "when" command. In order to introduce updates and changes in your infrastructure with a gradual approach, this may be helpful if you want to control when each of these hosts is deployed. You can create more flexible and dynamic playbooks that respond to any scenario or requirement by using conditional options with the 'when' command.

Conclusion:

- **In this activity, I conclude that I have been able to fulfill the objectives that are given. The first objective is to use when command in the playbook for different OS distributions. I have been able to accomplish the first objective by using when command in my own playbook. I used the when command in order to control the flow of execution in my playbook. It is also used for certain conditions such as running the task in Debian hosts only or running it only on CentOS. The second objective is to apply refactoring techniques in cleaning up the playbook codes. By applying refactoring, I have been able to improve the quality of my code, make it easier to understand and maintain, and reduce the amount of time and effort required for development.**