Name: Seruelas, Ronn Kristoper H.	**Date Performed:** 09-18-2023
Course/Section: CPE31S4	**Date Submitted:** 09-18-2023
Instructor: Dr. Jonathan V. Taylar	**Semester and SY:** 2nd Sem. - 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.

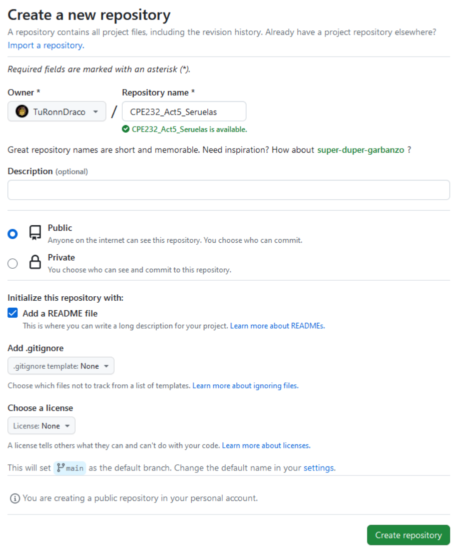
It is a best practice in ansible when you are working in a collaborative environment to use the command git pull. git pull is a Git command used to update the local version of a repository from a remote. By default, git pull does two things. Updates the current local working branch (currently checked 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

Preparation: Creation of new repository for this activity.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * TuRonnDraco / Repository name * CPE232_Act5_Seruelas

CPE232_Act5_Seruelas is available.

Great repository names are short and memorable. Need inspiration? How about super-duper-garbanzo ?

Description (optional)

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore
.gitignore template: None
Choose which files not to track from a list of templates. Learn more about ignoring files.

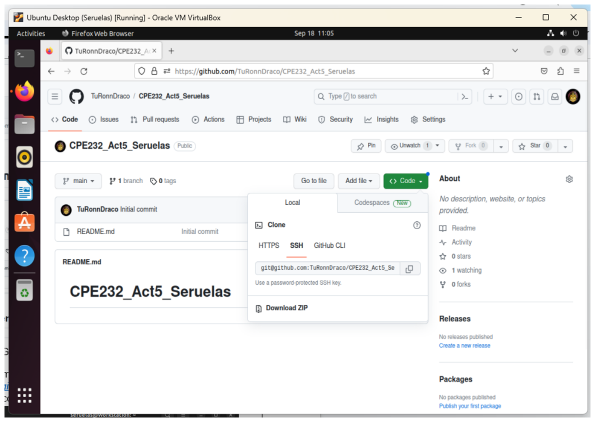
Choose a license
License: None
A license tells others what they can and can't do with your code. Learn more about licenses.

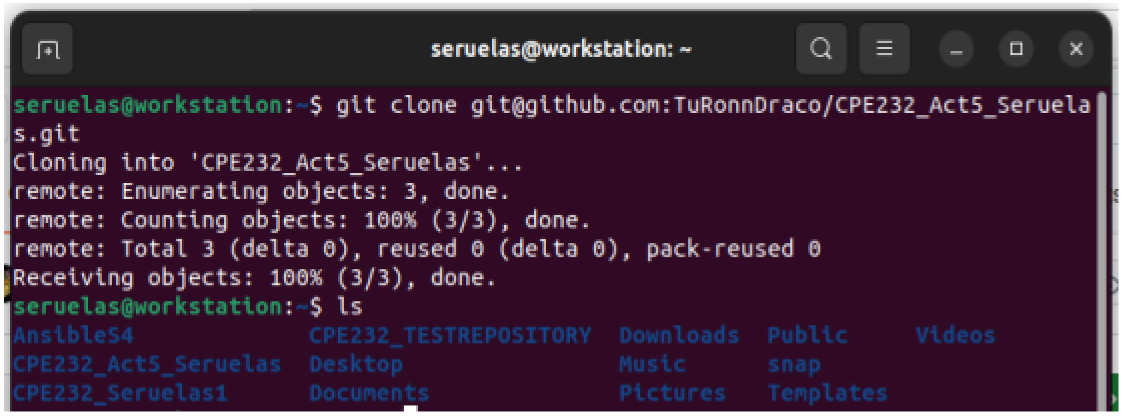
This will set main as the default branch. Change the default name in your settings.

You are creating a public repository in your personal account.

Create repository

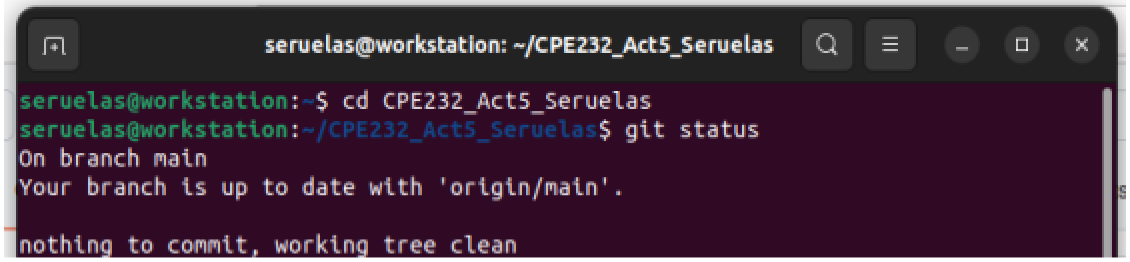
Figure 0.1 - Creation of new repository.





```
seruelas@workstation:~$ git clone git@github.com:TuRonnDraco/CPE232_Act5_Seruela
s.git
Cloning into 'CPE232_Act5_Seruelas'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
seruelas@workstation:~$ ls
Ansible54             CPE232_TESTREPOSITORY  Downloads  Public     Videos
CPE232_Act5_Seruelas  Desktop                Music      snap
CPE232_Seruelas1      Documents              Pictures   Templates
```

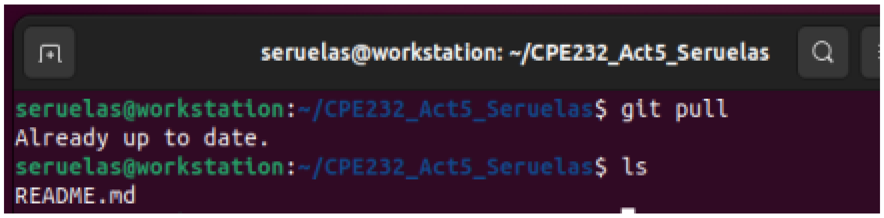
Figure 0.2-0.3 - Connection of new repository to manage/control node.



```
seruelas@workstation:~$ cd CPE232_Act5_Seruelas
seruelas@workstation:~/CPE232_Act5_Seruelas$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

Figure 0.4 - Successful creation of a new repository.

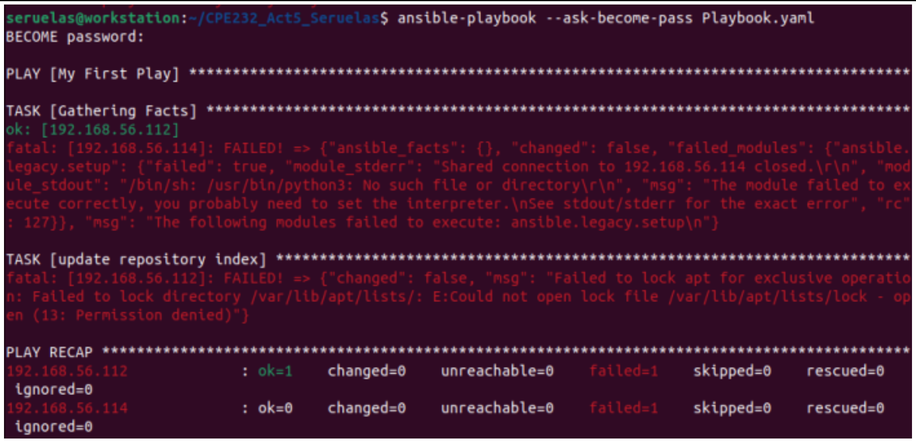
1. In the local machine, make sure you are in the local repository directory (*CPE232_yourname*). Issue the command git pull. When prompted, enter the correct passphrase or password. Describe what happened when you issue this command. Did something happen? Why?



```
seruelas@workstation:~/CPE232_Act5_Seruelas$ git pull
Already up to date.
seruelas@workstation:~/CPE232_Act5_Seruelas$ ls
README.md
```

Figure 1.1 - Execution of the git pull command. When the command is executed, it was prompted that the local repository was already up to date. The git pull command's purpose is to fetch and download all the files that are present in the repository, and to update the files in the local repository located at the control or manage node.

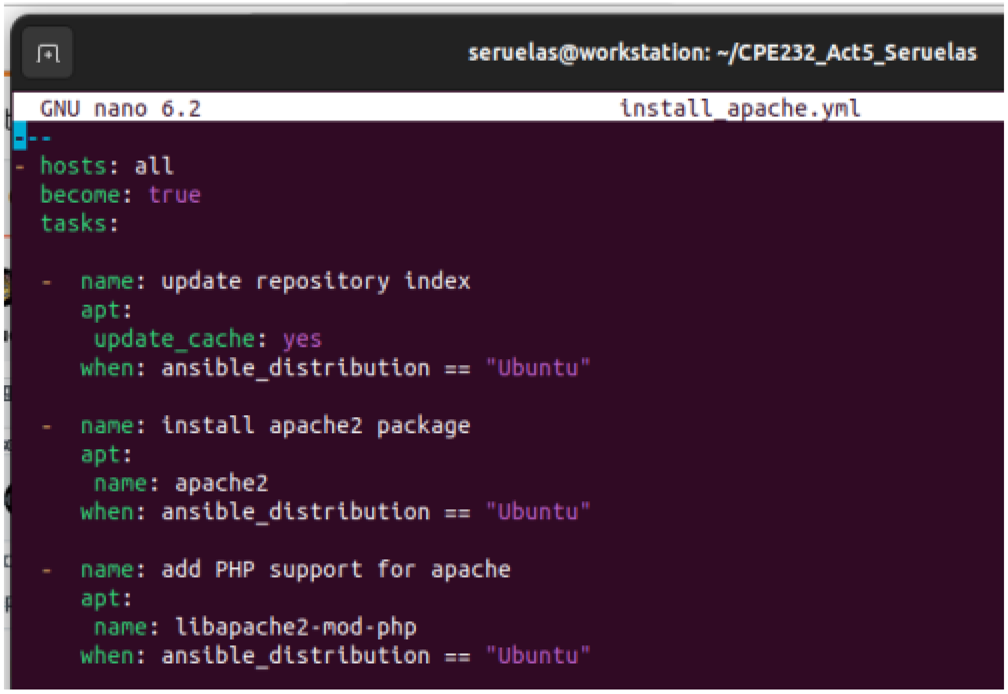
2. Edit the inventory file and add the IP address of the Centos VM. Issue the command we used to execute the playbook (the one we used in the last activity): *ansible-playbook --ask-become-pass install_apache.yml*. After executing this command, you may notice that it did not become successful in the 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."



```
seruelas@workstation:~/CPE232_Act5_Seruelas$ ansible-playbook --ask-become-pass Playbook.yaml
BECOME password:

PLAY [My First Play] *****************************************************************************

TASK [Gathering Facts] ***************************************************************************
ok: [192.168.56.112]
fatal: [192.168.56.114]: FAILED! => {"ansible_facts": {}, "changed": false, "failed_modules": {"ansible.
legacy.setup": {"failed": true, "module_stderr": "Shared connection to 192.168.56.114 closed.\r\n", "mod
ule_stdout": "/bin/sh: /usr/bin/python3: No such file or directory\r\n", "msg": "The module failed to ex
ecute correctly, you probably need to set the interpreter.\nSee stdout/stderr for the exact error", "rc"
: 127}}, "msg": "The following modules failed to execute: ansible.legacy.setup\n"}

TASK [update repository index] *******************************************************************
fatal: [192.168.56.112]: FAILED! => {"changed": false, "msg": "Failed to lock apt for exclusive operatio
n: Failed to lock directory /var/lib/apt/lists/: E:Could not open lock file /var/lib/apt/lists/lock - op
en (13: Permission denied)"}

PLAY RECAP ***************************************************************************************
192.168.56.112             : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0
 ignored=0
192.168.56.114             : ok=0    changed=0    unreachable=0    failed=1    skipped=0    rescued=0
 ignored=0
```

Figure 1.2 - Execution of playbook for installation of apache2, with the failure of the CentOS VM installation or run.

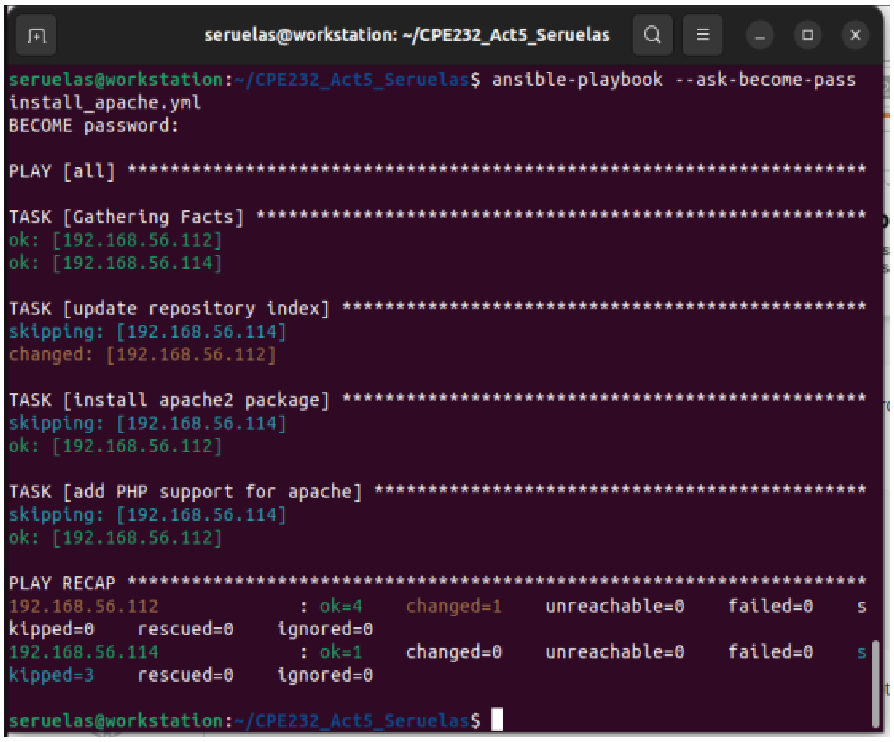
3. Edit the *install_apache.yml* file and insert the lines shown below.



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

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

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

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

Figure 1.3.1 - Editing the install_apache.yml file and inserting the new lines provided. The new lines provided focused on adding a specific ansible distribution, specified for Ubuntu.

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



```
seruelas@workstation:~/CPE232_Act5_Seruelas$ ansible-playbook --ask-become-pass
install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ***********************************************************
ok: [192.168.56.112]
ok: [192.168.56.114]

TASK [update repository index] ***************************************************
skipping: [192.168.56.114]
changed: [192.168.56.112]

TASK [install apache2 package] ***************************************************
skipping: [192.168.56.114]
ok: [192.168.56.112]

TASK [add PHP support for apache] ************************************************
skipping: [192.168.56.114]
ok: [192.168.56.112]

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

seruelas@workstation:~/CPE232_Act5_Seruelas$
```

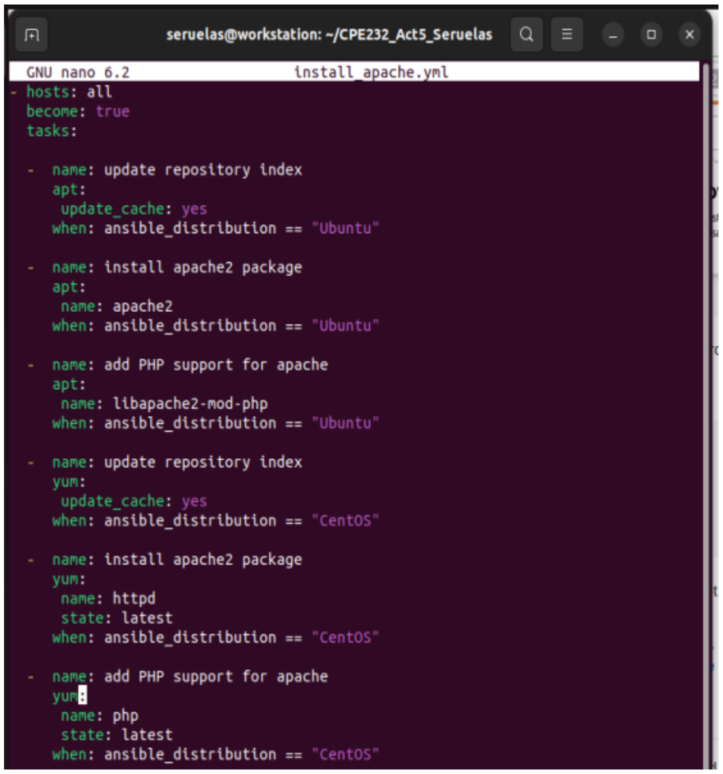
Figure 1.3.2 - Executing the modified install_apache.yml playbook. Once executed, it skipped the CentOS server and has only executed its list of commands to the Ubuntu based ansible systems.

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.

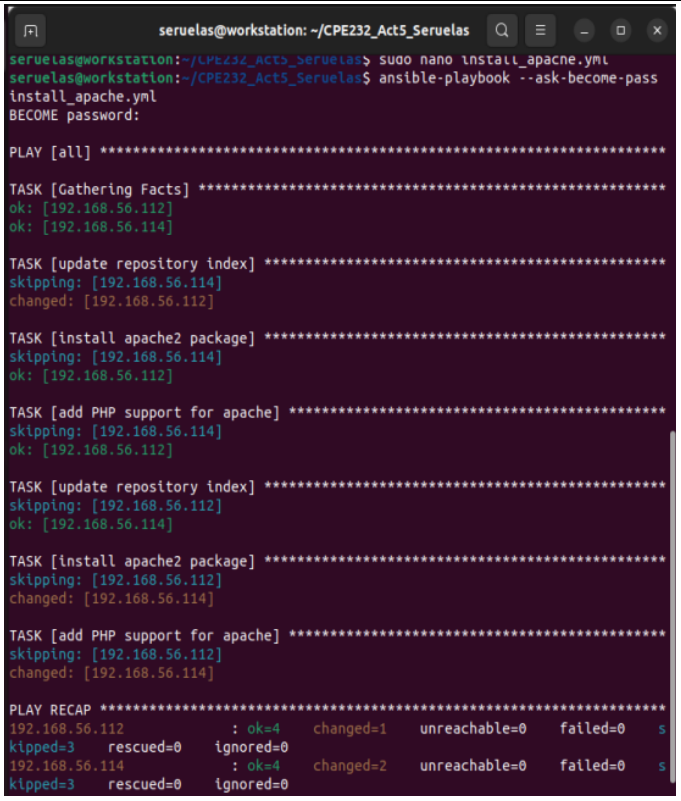


```
seruelas@workstation: ~/CPE232_Act5_Seruelas
GNU nano 6.2                    install_apache.yml
- hosts: all
  become: true
  tasks:

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

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

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

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

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

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

Figure 1.4.1 - Modification of the install_apache.yml playbook. Modification included the ansible distribution for CentOs.

Make sure to save and exit.

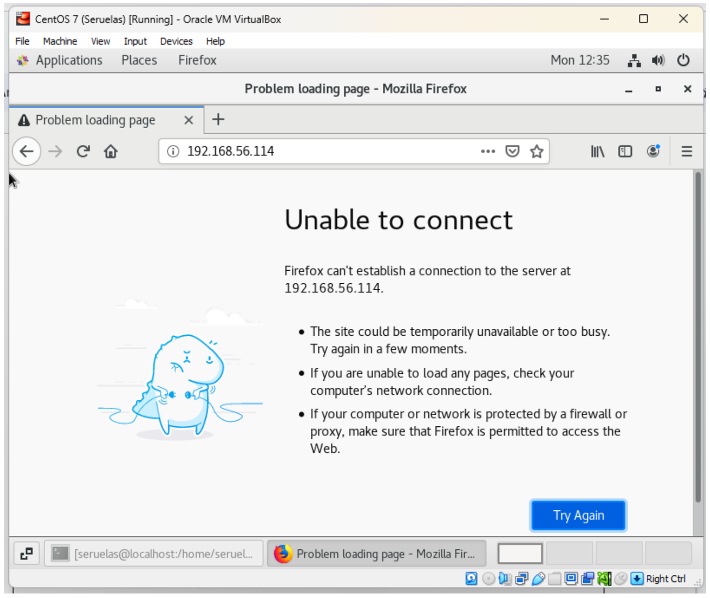
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
seruelas@workstation: ~/CPE232_Act5_Seruelas
seruelas@workstation:~/CPE232_Act5_Seruelas$ sudo nano install_apache.yml
seruelas@workstation:~/CPE232_Act5_Seruelas$ ansible-playbook --ask-become-pass
install_apache.yml
BECOME password:

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

TASK [Gathering Facts] ***********************************************************
ok: [192.168.56.112]
ok: [192.168.56.114]

TASK [update repository index] ***************************************************
skipping: [192.168.56.114]
changed: [192.168.56.112]

TASK [install apache2 package] ***************************************************
skipping: [192.168.56.114]
ok: [192.168.56.112]

TASK [add PHP support for apache] ************************************************
skipping: [192.168.56.114]
ok: [192.168.56.112]

TASK [update repository index] ***************************************************
skipping: [192.168.56.112]
ok: [192.168.56.114]

TASK [install apache2 package] ***************************************************
skipping: [192.168.56.112]
changed: [192.168.56.114]

TASK [add PHP support for apache] ************************************************
skipping: [192.168.56.112]
changed: [192.168.56.114]

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

Figure 1.4.2 - Execution of the modified playbook. It was able to install the apache2 and php to both ansible distributions, as we were able to specify where to install.

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.


CentOS 7 (Seruelas) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Firefox
Mon 12:35
Problem loading page - Mozilla Firefox
Problem loading page
192.168.56.114
Try Again
[seruelas@localhost:/home/seruel...
Problem loading page - Mozilla Fir...
Right Ctrl

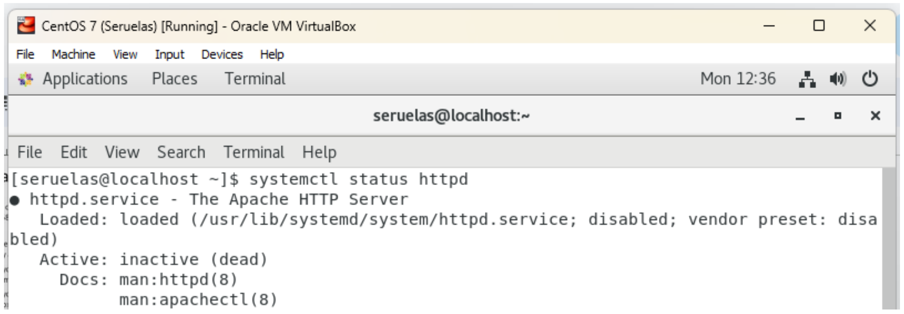
Unable to connect

Firefox can't establish a connection to the server at 192.168.56.114.

- 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.

Figure 1.5.1 - Verifying the installation of apache and php in CentOS. It was unsuccessful as CentOS settings are not configured yet.

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.


CentOS 7 (Seruelas) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Terminal
Mon 12:36
seruelas@localhost:~
File Edit View Search Terminal Help

```
[seruelas@localhost ~]$ systemctl status httpd
● 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)
```

Figure 1.5.2 - Checking of the system status of httpd.

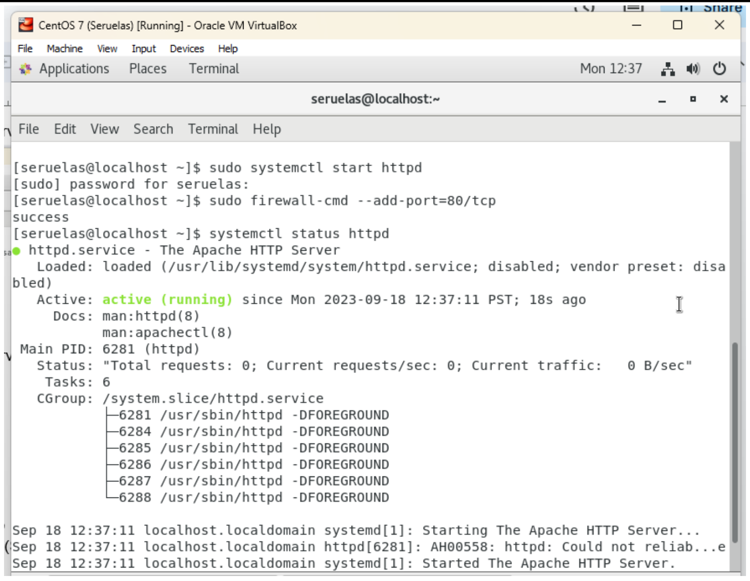
5.2 Issue the following command to start the service:

sudo systemctl start httpd

(When prompted, enter the sudo password)

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

(The result should be a success)

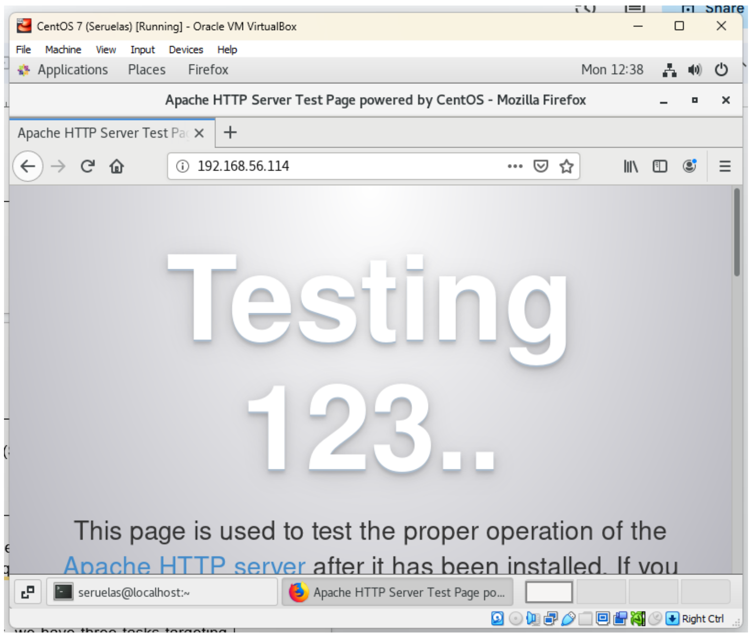


```
[seruelas@localhost ~]$ sudo systemctl start httpd
[sudo] password for seruelas:
[seruelas@localhost ~]$ sudo firewall-cmd --add-port=80/tcp
success
[seruelas@localhost ~]$ systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disa
bled)
   Active: active (running) since Mon 2023-09-18 12:37:11 PST; 18s ago
     Docs: man:httpd(8)
           man:apachectl(8)
 Main PID: 6281 (httpd)
   Status: "Total requests: 0; Current requests/sec: 0; Current traffic:   0 B/sec"
    Tasks: 6
   CGroup: /system.slice/httpd.service
           ├─6281 /usr/sbin/httpd -DFOREGROUND
           ├─6284 /usr/sbin/httpd -DFOREGROUND
           ├─6285 /usr/sbin/httpd -DFOREGROUND
           ├─6286 /usr/sbin/httpd -DFOREGROUND
           ├─6287 /usr/sbin/httpd -DFOREGROUND
           └─6288 /usr/sbin/httpd -DFOREGROUND

Sep 18 12:37:11 localhost.localdomain systemd[1]: Starting The Apache HTTP Server...
Sep 18 12:37:11 localhost.localdomain httpd[6281]: AH00558: httpd: Could not reliab...e
Sep 18 12:37:11 localhost.localdomain systemd[1]: Started The Apache HTTP Server.
```

CentOS 7 (Seruelas) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Terminal
Mon 12:37
seruelas@localhost:~
File Edit View Search Terminal Help

Figure 1.5.3 - Execution of the following commands allowed the system httpd to be enabled and to run at the background of the server.

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)



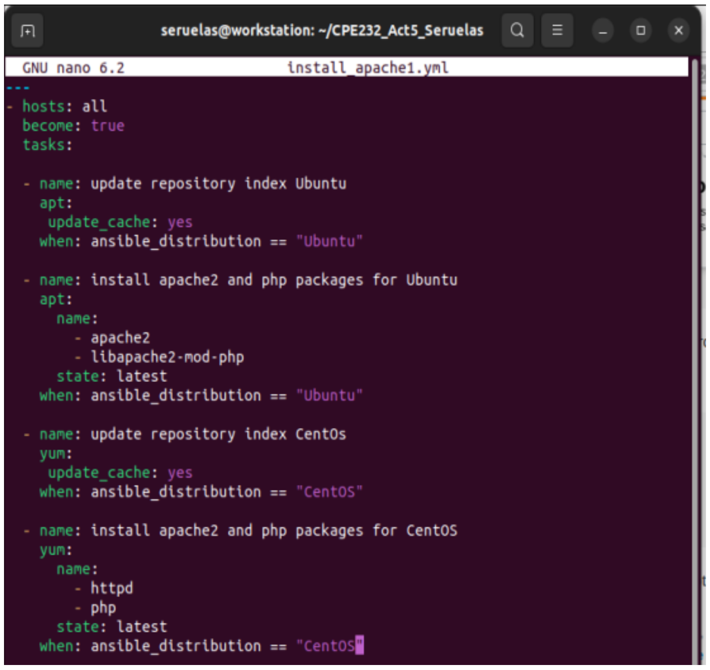
CentOS 7 (Seruelas) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Firefox
Mon 12:38
Apache HTTP Server Test Page powered by CentOS - Mozilla Firefox
Apache HTTP Server Test Pa
192.168.56.114
Testing 123..
This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you
seruelas@localhost:~
Apache HTTP Server Test Page po...
Right Ctrl

Figure 1.5.4 - Verification of the service through browser search. We can see that the httpd service is running as we have successfully accessed the httpd through 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:

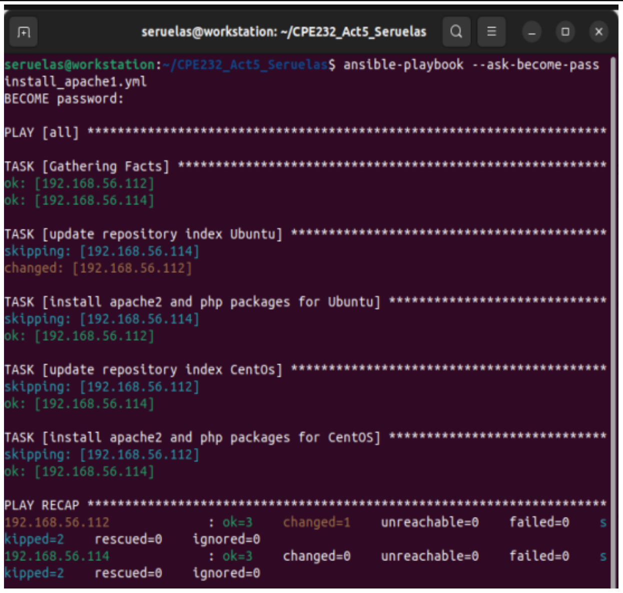


```
GNU nano 6.2                    install_apache1.yml
---
- hosts: all
  become: true
  tasks:

  - 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 CentOs
    yum:
     update_cache: yes
    when: ansible_distribution == "CentOS"

  - name: install apache2 and php packages for CentOS
    yum:
      name:
        - httpd
        - php
      state: latest
    when: ansible_distribution == "CentOS"
```

Figure 2.1.1 - Modifying the playbook, in order for it to be able to execute its three tasks over two ansible distributions, the Ubuntu and the CentOS.

Make sure to save the file and exit.

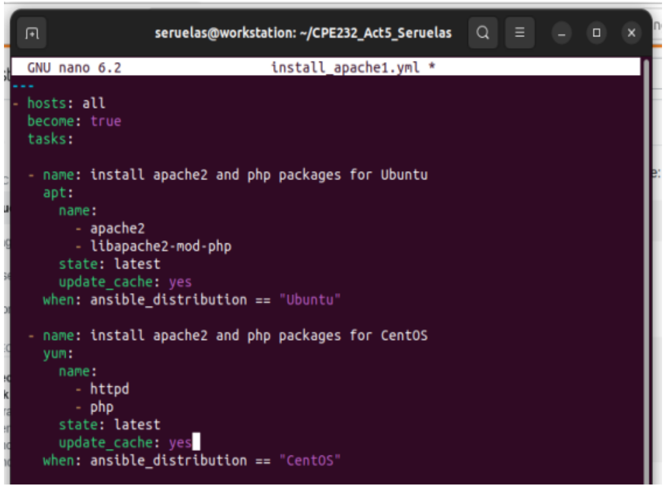
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
seruelas@workstation:~/CPE232_Act5_Seruelas$ ansible-playbook --ask-become-pass
install_apache1.yml
BECOME password:

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

TASK [Gathering Facts] ***********************************************************
ok: [192.168.56.112]
ok: [192.168.56.114]

TASK [update repository index Ubuntu] ********************************************
skipping: [192.168.56.114]
changed: [192.168.56.112]

TASK [install apache2 and php packages for Ubuntu] *******************************
skipping: [192.168.56.114]
ok: [192.168.56.112]

TASK [update repository index CentOs] ********************************************
skipping: [192.168.56.112]
ok: [192.168.56.114]

TASK [install apache2 and php packages for CentOS] *******************************
skipping: [192.168.56.112]
ok: [192.168.56.114]

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

Figure 2.1.2 - Execution of the modified playbook. In this execution, we can see that it successfully installed the apache and the php over the two ansible distributions after the consolidation of the playbook.

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:

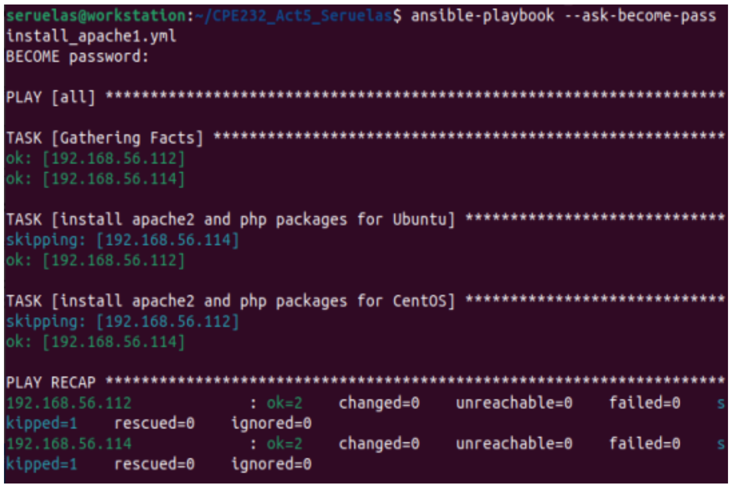


```
GNU nano 6.2                    install_apache1.yml *
---
- hosts: all
  become: true
  tasks:

  - 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
    yum:
      name:
        - httpd
        - php
      state: latest
      update_cache: yes
    when: ansible_distribution == "CentOS"
```

Figure 2.2.1 - Modification of the playbook where every task has been consolidated into one, allowing to not only install apache and php, but also to update the local repository's cache.

Make sure to save the file and exit.

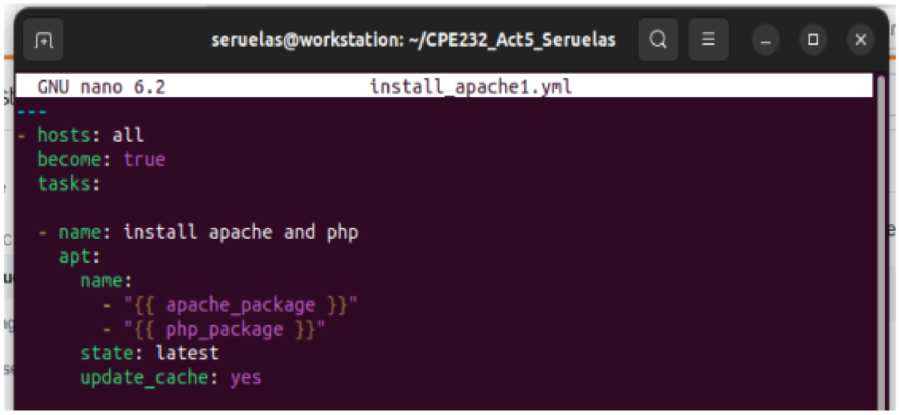
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
seruelas@workstation:~/CPE232_Act5_Seruelas$ ansible-playbook --ask-become-pass
install_apache1.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.112]
ok: [192.168.56.114]

TASK [install apache2 and php packages for Ubuntu] *****************************
skipping: [192.168.56.114]
ok: [192.168.56.112]

TASK [install apache2 and php packages for CentOS] *****************************
skipping: [192.168.56.112]
ok: [192.168.56.114]

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

Figure 2.2.2 - Executing the modified playbook. By executing the modified playbook, it successfully executed its tasks from installation of apache and php, and also updating the repository cache, even after the consolidation or simplification of the playbook.

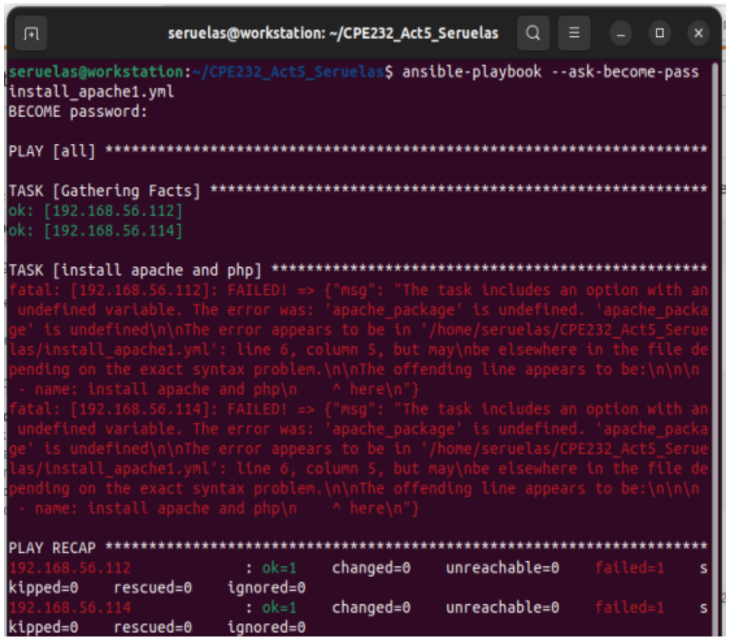
3. Finally, we can consolidate these 2 plays in just 1 play. This can be done by declaring variables that will represent the packages that we want to install. Basically, the apache_package and php_package are variables. The names are arbitrary, which means we can choose different names. We also take 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.



```
seruelas@workstation: ~/CPE232_Act5_Seruelas
  GNU nano 6.2                    install_apache1.yml
---
- hosts: all
  become: true
  tasks:

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

Figure 2.3.1 - Modification of the playbook, by consolidating the playbook's tasks into its most simplified form, where it is efficient, and also minimal for the playbook's memory.

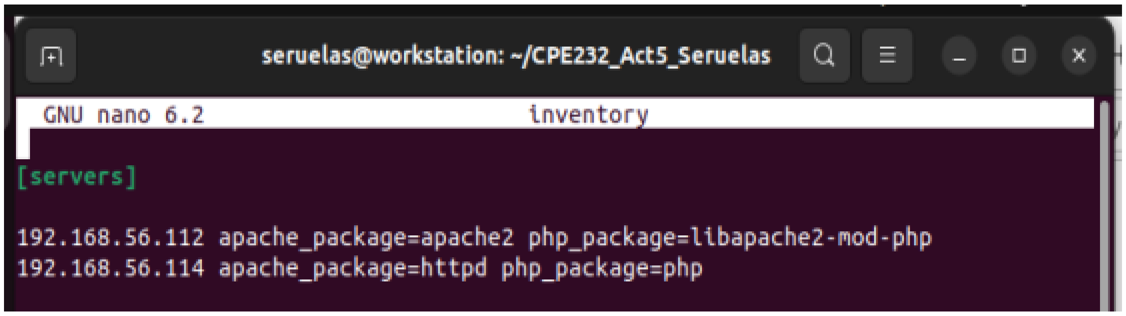
Run *ansible-playbook --ask-become-pass install_apache.yml* and describe the result.



```
seruelas@workstation: ~/CPE232_Act5_Seruelas
seruelas@workstation:~/CPE232_Act5_Seruelas$ ansible-playbook --ask-become-pass
install_apache1.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.112]
ok: [192.168.56.114]

TASK [install apache and php] **************************************************
fatal: [192.168.56.112]: FAILED! => {"msg": "The task includes an option with an
 undefined variable. The error was: 'apache_package' is undefined. 'apache_packa
ge' is undefined\n\nThe error appears to be in '/home/seruelas/CPE232_Act5_Serue
las/install_apache1.yml': line 6, column 5, but may\nbe elsewhere in the file de
pending on the exact syntax problem.\n\nThe offending line appears to be:\n\n\n
 - name: install apache and php\n    ^ here\n"}
fatal: [192.168.56.114]: FAILED! => {"msg": "The task includes an option with an
 undefined variable. The error was: 'apache_package' is undefined. 'apache_packa
ge' is undefined\n\nThe error appears to be in '/home/seruelas/CPE232_Act5_Serue
las/install_apache1.yml': line 6, column 5, but may\nbe elsewhere in the file de
pending on the exact syntax problem.\n\nThe offending line appears to be:\n\n\n
 - name: install apache and php\n    ^ here\n"}

PLAY RECAP *********************************************************************
192.168.56.112             : ok=1    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
192.168.56.114             : ok=1    changed=0    unreachable=0    failed=1    s
kipped=0    rescued=0    ignored=0
```

Figure 2.3.1 - Execution of the modified playbook. The result of the execution was a failure, as there was a syntax problem or 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:

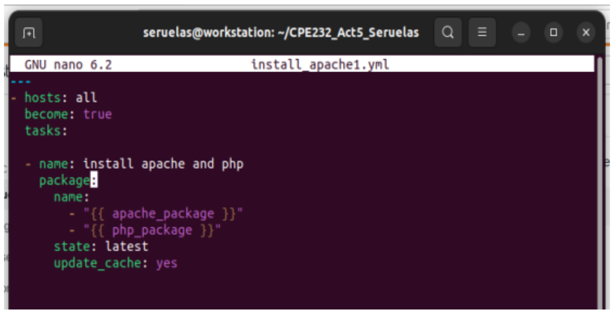


```
seruelas@workstation: ~/CPE232_Act5_Seruelas
  GNU nano 6.2                        inventory

[servers]

192.168.56.112 apache_package=apache2 php_package=libapache2-mod-php
192.168.56.114 apache_package=httpd php_package=php
```

Figure 2.4.1 - Modification of the inventory file, where we have corrected the proper variables for the hosts, allowing them to execute the new playbook.

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

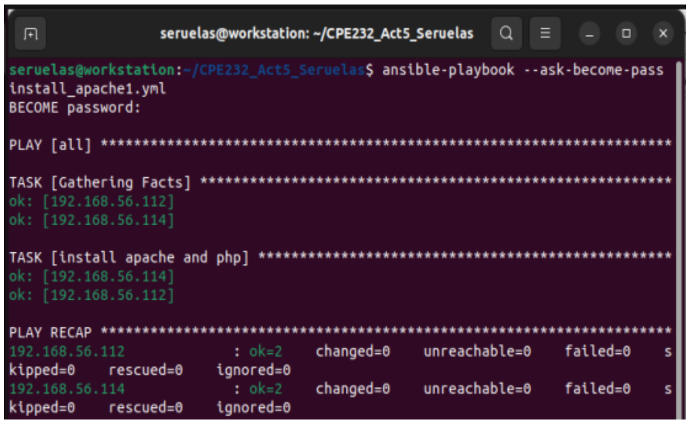
Finally, we still have one more thing to change in our *install_apache.yml* file. In task 2.3, you may notice that the package is assign as apt, which will not run in CentOS. Replace the *apt* with *package*. Package is a module in ansible that is generic, which is going to use whatever package manager 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]()



```
seruelas@workstation: ~/CPE232_Act5_Seruelas
GNU nano 6.2                    install_apache1.yml
---
- hosts: all
  become: true
  tasks:

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

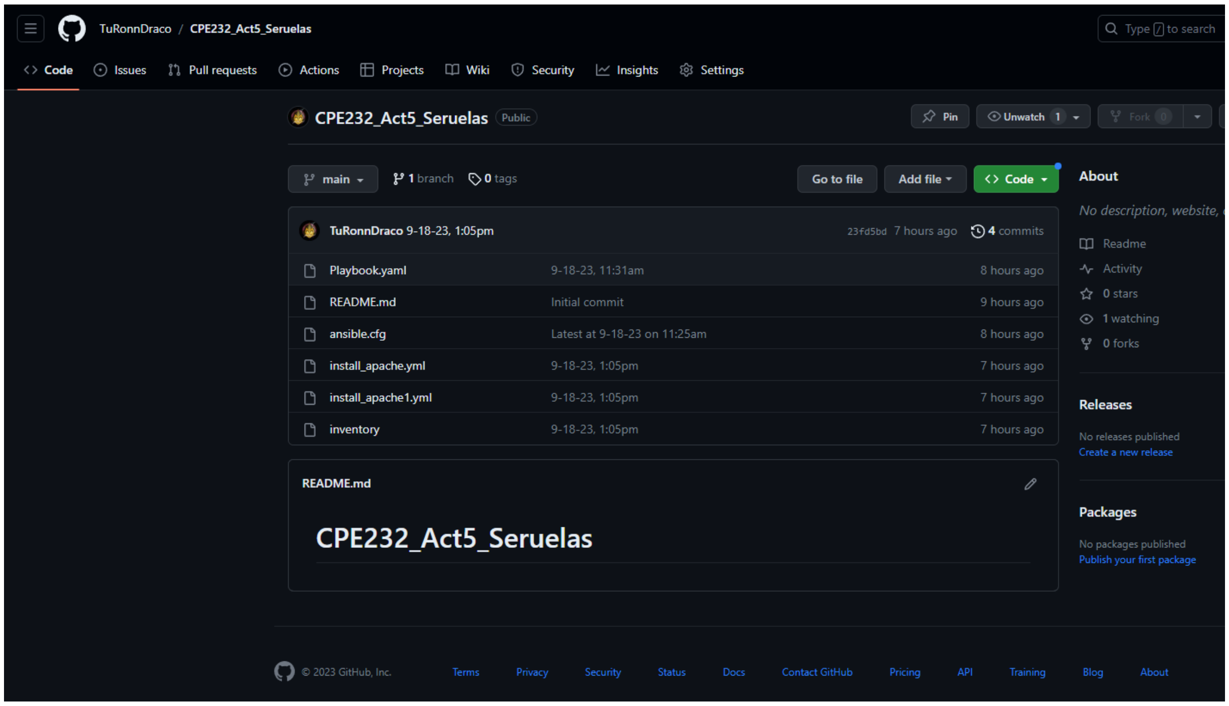
Figure 2.4.2 - Modification of the playbook, where we are to replace the syntax *apt* into *package*, allowing any of the ansible distributions to execute and be able to install the packages.

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



```
seruelas@workstation: ~/CPE232_Act5_Seruelas
seruelas@workstation:~/CPE232_Act5_Seruelas$ ansible-playbook --ask-become-pass
install_apache1.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.112]
ok: [192.168.56.114]

TASK [install apache and php] **************************************************
ok: [192.168.56.114]
ok: [192.168.56.112]

PLAY RECAP *********************************************************************
192.168.56.112             : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
192.168.56.114             : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
```

Figure 2.4.3 - Execution of the modified playbook. With the new modification, we are able to successfully execute the playbook, allowing us to not only consolidate the contents and the tasks of the playbook, but also making it executable for other ansible distributions.



TuRonnDraco 9-18-23, 1:05pm 23fd5bd 7 hours ago 4 commits

Playbook.yaml	9-18-23, 11:31am	8 hours ago
README.md	Initial commit	9 hours ago
ansible.cfg	Latest at 9-18-23 on 11:25am	8 hours ago
install_apache.yml	9-18-23, 1:05pm	7 hours ago
install_apache1.yml	9-18-23, 1:05pm	7 hours ago
inventory	9-18-23, 1:05pm	7 hours ago

README.md

CPE232_Act5_Seruelas

Figure 2.5 - Verification of the changes and the tasks done in the repository, located at github.

https://github.com/TuRonnDraco/CPE232_Act5_Seruelas

Reflections:

Answer the following:

1. Why do you think refactoring of playbook codes is important?
   - Refactoring of playbook codes or content is important as it allows to save more memory or space inside of the playbook. It also allows the playbook to be executed more efficiently without using too much memory. Consolidation or refactoring the playbook codes also allows us to make a more simpler playbook that is more understandable and readable to human eyes.

2. When do we use the "when" command in a playbook?
   - We can use the command "when" in a playbook when we want to write a code that needs a conditional statement, in which it will only execute the task at hand when it meets the condition set for the task. It can also be used as to organize a set of conditions in a playbook, in which it can only be executed when the condition of the task is met.

Conclusion:

In conclusion, The students have educated and learned about consolidation and the refactoring of the playbooks and what benefits it can give to a system administrator. In this activity, the students were able to learn on how one can consolidate or refactor a playbook by simplification of codes, and also how the students can set conditional statements that the playbook can meet. By finishing the activity, not only that the students have learned or educated themselves on the benefits of consolidating or refactoring a playbook, in which it creates a more efficient and memory saving play, but also they were able to apply what they have learned by creating a consolidated playbook, that was to install apache and php, and to also update the local repository cache, that was to install these packages into a certain ansible distribution. Finally, The students are able to say that they are able to learn and to apply the consolidation and the conditional statements in creating their own playbooks.

"I affirm that I have not received or given any unauthorized help and that all work is my own."