Name: Juanson, Aliya Dane P.	**Date Performed: September 12, 2024**
Course/Section: CpE31S2	**Date Submitted: September 13, 2024**
Instructor: Sir Robin Valenzuela	**Semester and SY: 2024-2025**

Activity 4: Running Elevated Ad hoc Commands

1. **Objectives:**

1.1 Use commands that makes changes to remote machines

1.2 Use playbook in automating ansible commands

2. **Discussion:**

Provide screenshots for each task.

Elevated Ad hoc commands

So far, we have not performed ansible commands that makes changes to the remote servers. We manage to gather facts and connect to the remote machines, but we still did not make changes on those machines. In this activity, we will learn to use commands that would install, update, and upgrade packages in the remote machines. We will also create a playbook that will be used for automations.

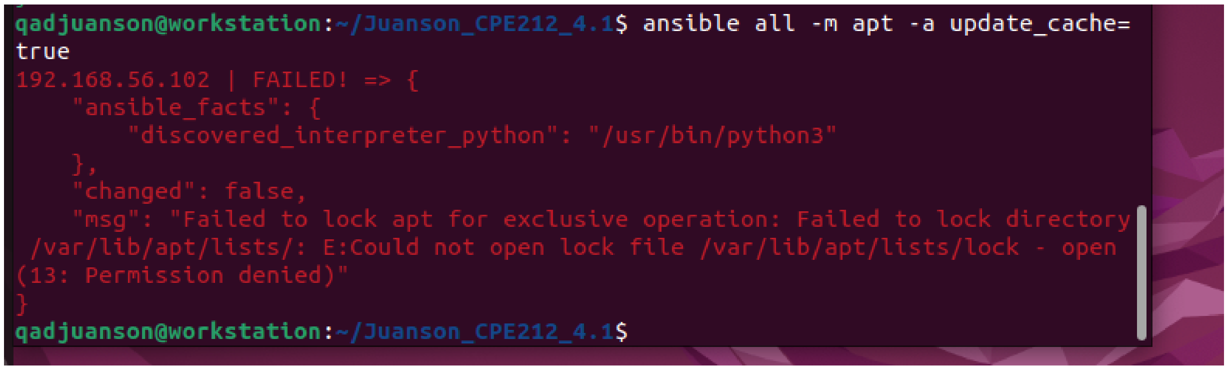
Playbooks record and execute **Ansible**'s configuration, deployment, and orchestration functions. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. If Ansible modules are the tools in your workshop, playbooks are your instruction manuals, and your inventory of hosts are your raw material. At a basic level, playbooks can be used to manage configurations of and deployments to remote machines. At a more advanced level, they can sequence multi-tier rollouts involving rolling updates, and can delegate actions to other hosts, interacting with monitoring servers and load balancers along the way. You can check this documentation if you want to learn more about playbooks. Working with playbooks — Ansible Documentation

Task 1: Run elevated ad hoc commands

1. Locally, we use the command *sudo apt update* when we want to download package information from all configured resources. The sources often defined in /etc/apt/sources.list file and other files located in /etc/apt/sources.list.d/ directory. So, when you run update command, it downloads the package information from the Internet. It is useful to get info on an updated version of packages or their dependencies. We can only run

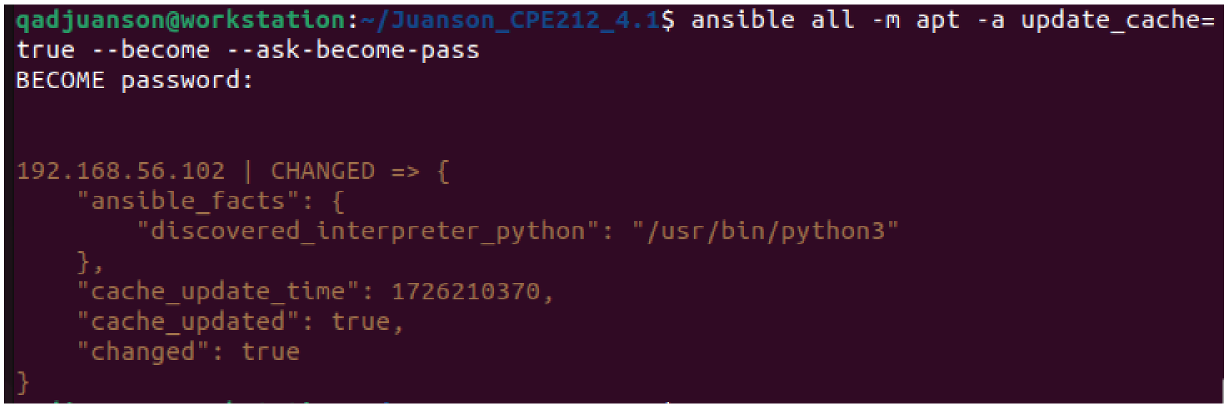
an apt update command in a remote machine. Issue the following command:

ansible all -m apt -a update_cache=true
What is the result of the command? Is it successful?



```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible all -m apt -a update_cache=
true
192.168.56.102 | FAILED! => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "msg": "Failed to lock apt for exclusive operation: Failed to lock directory
 /var/lib/apt/lists/: E:Could not open lock file /var/lib/apt/lists/lock - open
(13: Permission denied)"
}
qadjuanson@workstation:~/Juanson_CPE212_4.1$
```

Try editing the command and add something that would elevate the privilege. Issue the command *ansible all -m apt -a update_cache=true --become --ask-become-pass.* Enter the sudo password when prompted. You will notice now that the output of this command is a success. The *update_cache=true* is the same thing as running *sudo apt update*. The --become command elevate the privileges and the *--ask-become-pass* asks for the password. For now, even if we only have changed the packaged index, we were able to change something on the remote server.

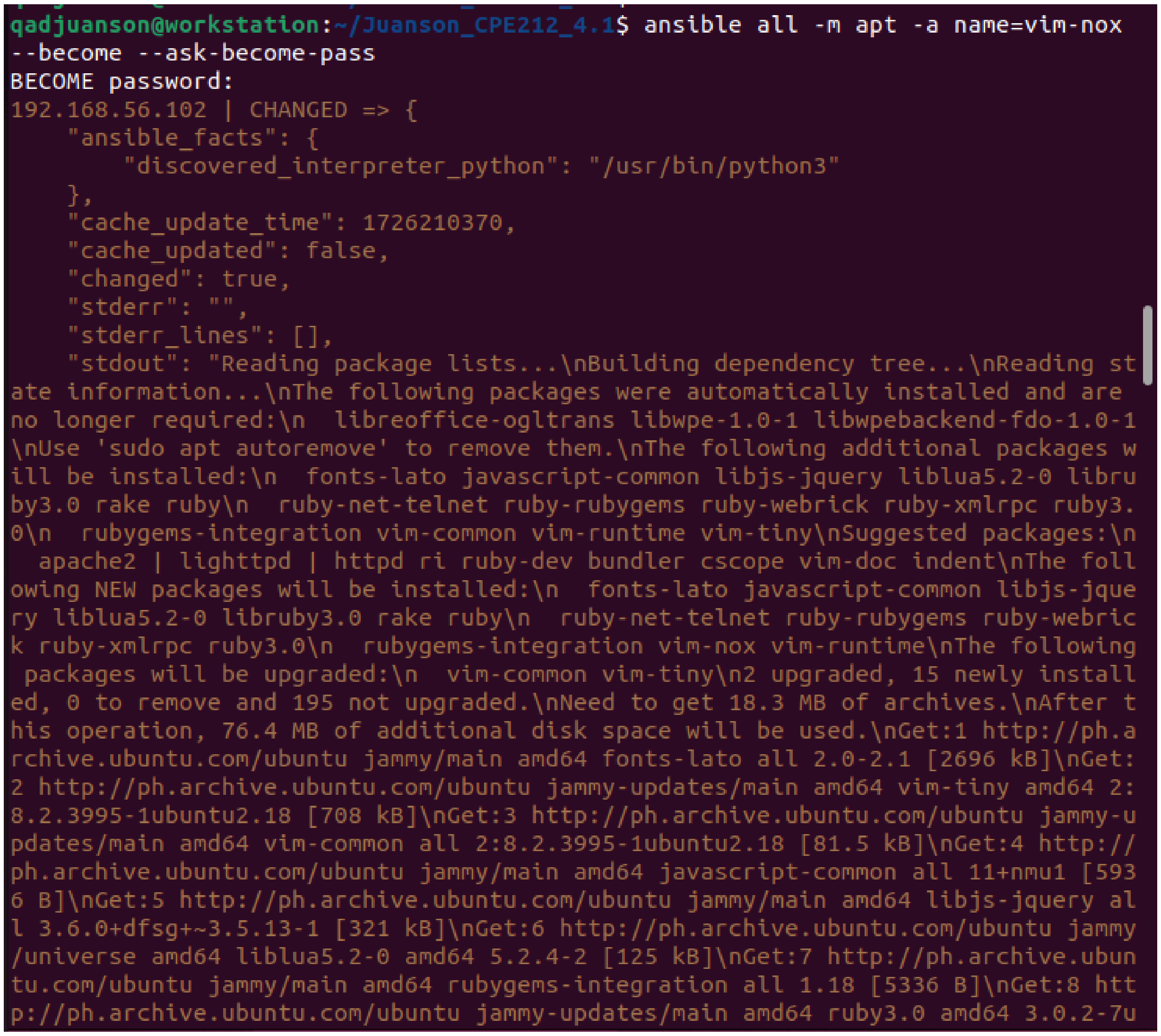


```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible all -m apt -a update_cache=
true --become --ask-become-pass
BECOME password:


192.168.56.102 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1726210370,
    "cache_updated": true,
    "changed": true
}
```

You may notice after the second command was executed, the status is CHANGED compared to the first command, which is FAILED.

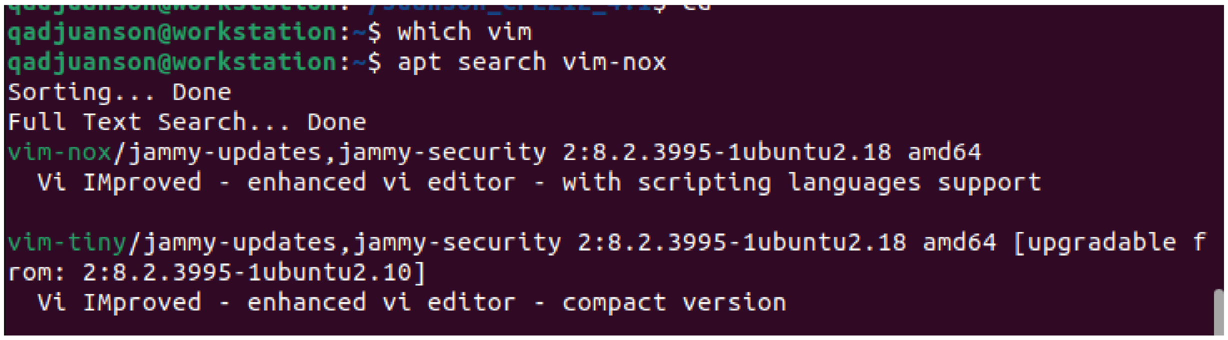
2. Let's try to install VIM, which is an almost compatible version of the UNIX editor Vi. To do this, we will just changed the module part in 1.1 instruction. Here is the command: *ansible all -m apt -a name=vim-nox --become --ask-become-pass.* The command would take some time after typing the

password because the local machine instructed the remote servers to actually install the package.



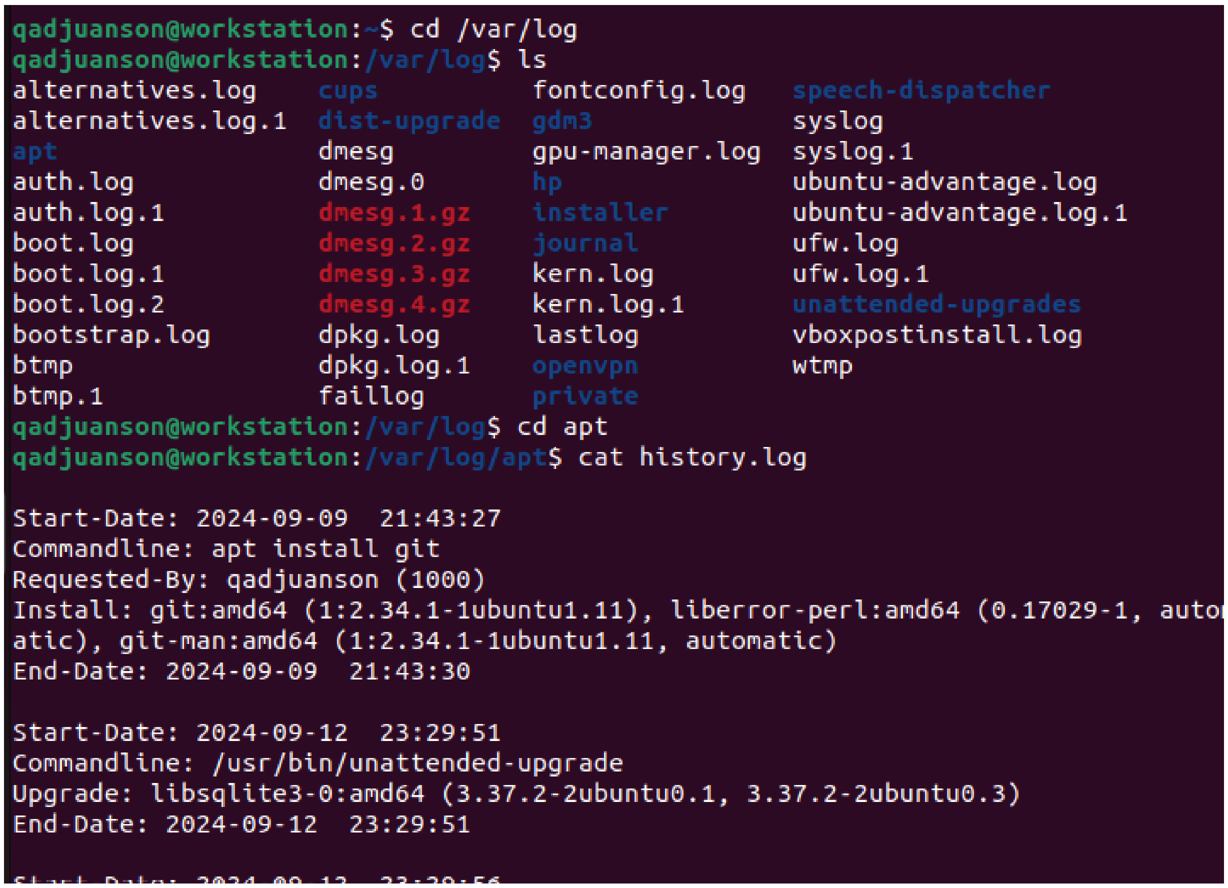
```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible all -m apt -a name=vim-nox
--become --ask-become-pass
BECOME password:
192.168.56.102 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1726210370,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading st
ate information...\nThe following packages were automatically installed and are
no longer required:\n  libreoffice-ogltrans libwpe-1.0-1 libwpebackend-fdo-1.0-1
\nUse 'sudo apt autoremove' to remove them.\nThe following additional packages w
ill be installed:\n  fonts-lato javascript-common libjs-jquery liblua5.2-0 libru
by3.0 rake ruby\n  ruby-net-telnet ruby-rubygems ruby-webrick ruby-xmlrpc ruby3.
0\n  rubygems-integration vim-common vim-runtime vim-tiny\nSuggested packages:\n
  apache2 | lighttpd | httpd ri ruby-dev bundler cscope vim-doc indent\nThe foll
owing NEW packages will be installed:\n  fonts-lato javascript-common libjs-jque
ry liblua5.2-0 libruby3.0 rake ruby\n  ruby-net-telnet ruby-rubygems ruby-webric
k ruby-xmlrpc ruby3.0\n  rubygems-integration vim-nox vim-runtime\nThe following
 packages will be upgraded:\n  vim-common vim-tiny\n2 upgraded, 15 newly install
ed, 0 to remove and 195 not upgraded.\nNeed to get 18.3 MB of archives.\nAfter t
his operation, 76.4 MB of additional disk space will be used.\nGet:1 http://ph.a
rchive.ubuntu.com/ubuntu jammy/main amd64 fonts-lato all 2.0-2.1 [2696 kB]\nGet:
2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 vim-tiny amd64 2:
8.2.3995-1ubuntu2.18 [708 kB]\nGet:3 http://ph.archive.ubuntu.com/ubuntu jammy-u
pdates/main amd64 vim-common all 2:8.2.3995-1ubuntu2.18 [81.5 kB]\nGet:4 http://
ph.archive.ubuntu.com/ubuntu jammy/main amd64 javascript-common all 11+nmu1 [593
6 B]\nGet:5 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 libjs-jquery al
l 3.6.0+dfsg+~3.5.13-1 [321 kB]\nGet:6 http://ph.archive.ubuntu.com/ubuntu jammy
/universe amd64 liblua5.2-0 amd64 5.2.4-2 [125 kB]\nGet:7 http://ph.archive.ubun
tu.com/ubuntu jammy/main amd64 rubygems-integration all 1.18 [5336 B]\nGet:8 htt
p://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby3.0 amd64 3.0.2-7u
```

2.1 Verify that you have installed the package in the remote servers. Issue the command *which vim* and the command *apt search vim-nox* respectively. Was the command successful?



```
qadjuanson@workstation:~$ which vim
qadjuanson@workstation:~$ apt search vim-nox
Sorting... Done
Full Text Search... Done
vim-nox/jammy-updates,jammy-security 2:8.2.3995-1ubuntu2.18 amd64
  Vi IMproved - enhanced vi editor - with scripting languages support

vim-tiny/jammy-updates,jammy-security 2:8.2.3995-1ubuntu2.18 amd64 [upgradable f
rom: 2:8.2.3995-1ubuntu2.10]
  Vi IMproved - enhanced vi editor - compact version
```

2.2 Check the logs in the servers using the following commands: *cd /var/log*. After this, issue the command *ls,* go to the folder *apt* and open history.log. Describe what you see in the history.log.

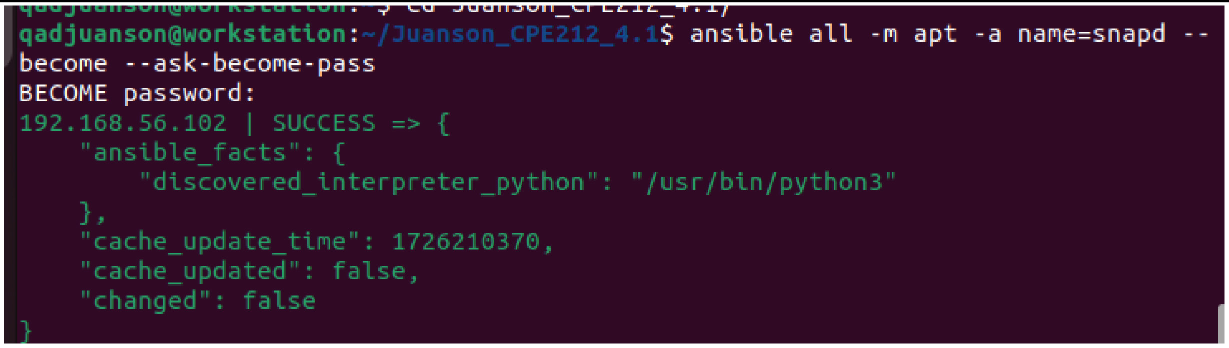


```
qadjuanson@workstation:~$ cd /var/log
qadjuanson@workstation:/var/log$ ls
alternatives.log    cups          fontconfig.log  speech-dispatcher
alternatives.log.1  dist-upgrade  gdm3            syslog
apt                 dmesg         gpu-manager.log syslog.1
auth.log            dmesg.0       hp              ubuntu-advantage.log
auth.log.1          dmesg.1.gz    installer       ubuntu-advantage.log.1
boot.log            dmesg.2.gz    journal         ufw.log
boot.log.1          dmesg.3.gz    kern.log        ufw.log.1
boot.log.2          dmesg.4.gz    kern.log.1      unattended-upgrades
bootstrap.log       dpkg.log      lastlog         vboxpostinstall.log
btmp                dpkg.log.1    openvpn         wtmp
btmp.1              faillog       private
qadjuanson@workstation:/var/log$ cd apt
qadjuanson@workstation:/var/log/apt$ cat history.log

Start-Date: 2024-09-09  21:43:27
Commandline: apt install git
Requested-By: qadjuanson (1000)
Install: git:amd64 (1:2.34.1-1ubuntu1.11), liberror-perl:amd64 (0.17029-1, autom
atic), git-man:amd64 (1:2.34.1-1ubuntu1.11, automatic)
End-Date: 2024-09-09  21:43:30

Start-Date: 2024-09-12  23:29:51
Commandline: /usr/bin/unattended-upgrade
Upgrade: libsqlite3-0:amd64 (3.37.2-2ubuntu0.1, 3.37.2-2ubuntu0.3)
End-Date: 2024-09-12  23:29:51
```

3. This time, we will install a package called snapd. Snap is pre-installed in Ubuntu system. However, our goal is to create a command that checks for the latest installation package.

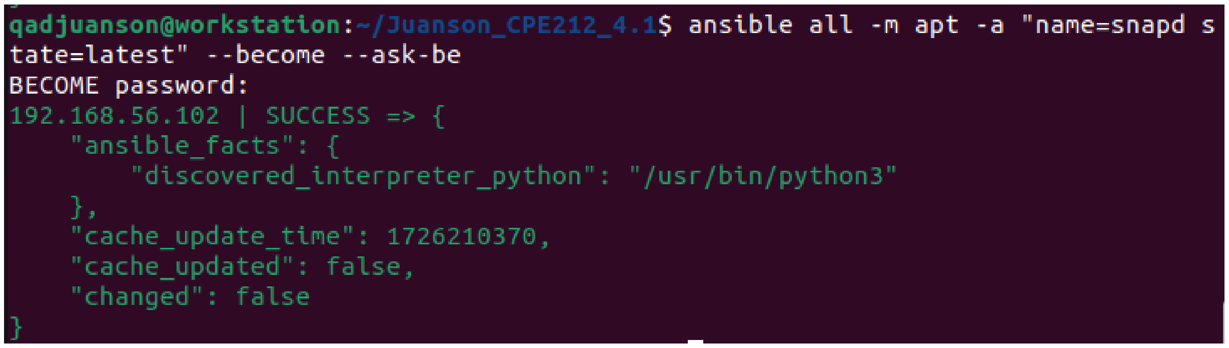
   3.1 Issue the command: *ansible all -m apt -a name=snapd --become --ask-become-pass*
   Can you describe the result of this command? Is it a success? Did it change anything in the remote servers?

- It changes the update time and the command shows a successful ping in server1.



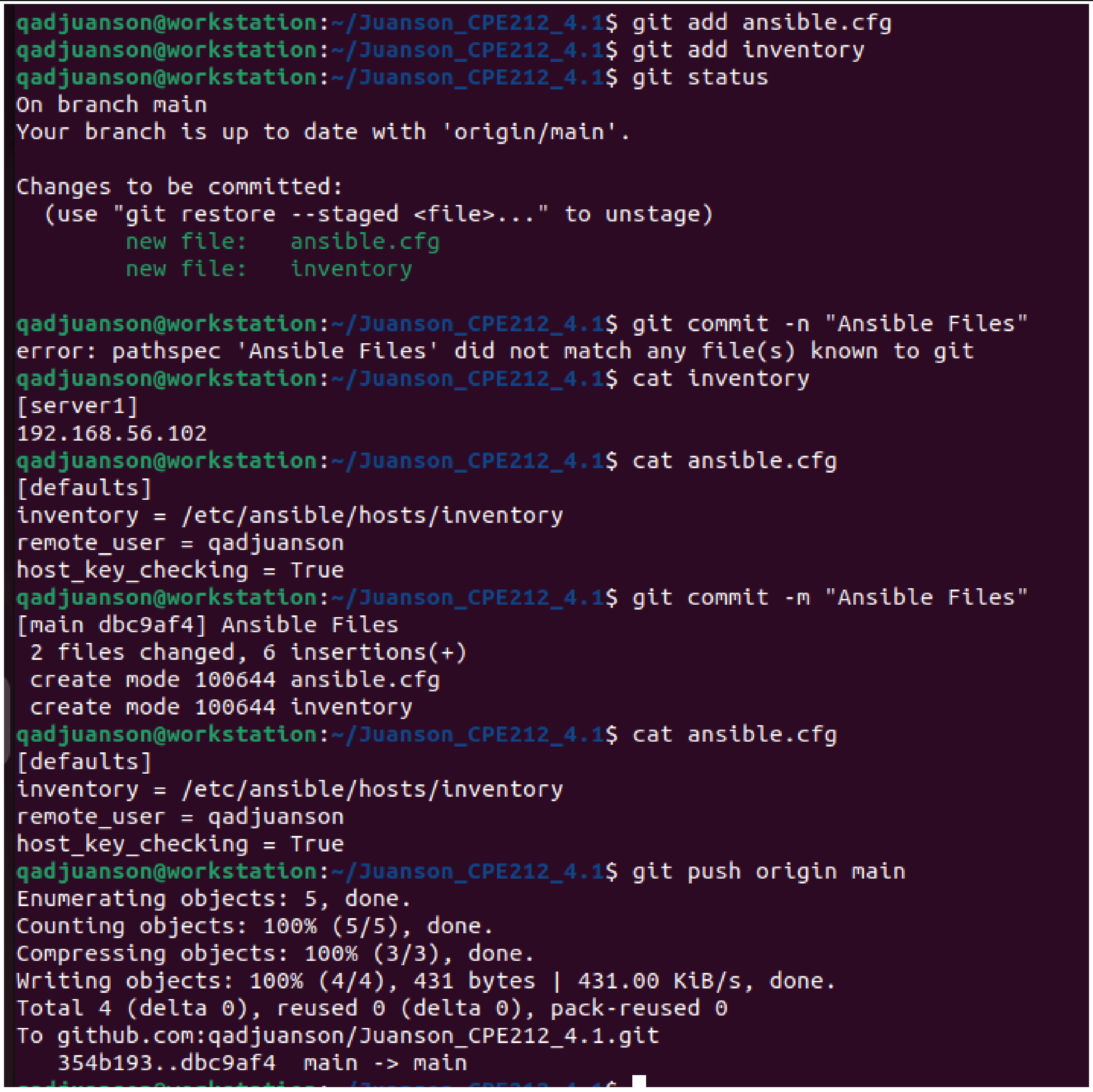
```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible all -m apt -a name=snapd --
become --ask-become-pass
BECOME password:
192.168.56.102 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1726210370,
    "cache_updated": false,
    "changed": false
}
```

3.2 Now, try to issue this command: *ansible all -m apt -a "name=snapd state=latest" --become --ask-become-pass*
Describe the output of this command. Notice how we added the command *state=latest* and placed them in double quotations.



```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible all -m apt -a "name=snapd s
tate=latest" --become --ask-be
BECOME password:
192.168.56.102 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1726210370,
    "cache_updated": false,
    "changed": false
}
```

4. At this point, make sure to commit all changes to GitHub.



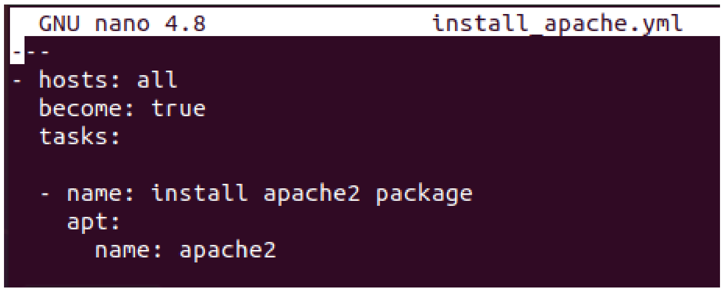
```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ git add ansible.cfg
qadjuanson@workstation:~/Juanson_CPE212_4.1$ git add inventory
qadjuanson@workstation:~/Juanson_CPE212_4.1$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   ansible.cfg
        new file:   inventory

qadjuanson@workstation:~/Juanson_CPE212_4.1$ git commit -n "Ansible Files"
error: pathspec 'Ansible Files' did not match any file(s) known to git
qadjuanson@workstation:~/Juanson_CPE212_4.1$ cat inventory
[server1]
192.168.56.102
qadjuanson@workstation:~/Juanson_CPE212_4.1$ cat ansible.cfg
[defaults]
inventory = /etc/ansible/hosts/inventory
remote_user = qadjuanson
host_key_checking = True
qadjuanson@workstation:~/Juanson_CPE212_4.1$ git commit -m "Ansible Files"
[main dbc9af4] Ansible Files
 2 files changed, 6 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 inventory
qadjuanson@workstation:~/Juanson_CPE212_4.1$ cat ansible.cfg
[defaults]
inventory = /etc/ansible/hosts/inventory
remote_user = qadjuanson
host_key_checking = True
qadjuanson@workstation:~/Juanson_CPE212_4.1$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 431 bytes | 431.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:qadjuanson/Juanson_CPE212_4.1.git
   354b193..dbc9af4  main -> main
```

Task 2: Writing our First Playbook

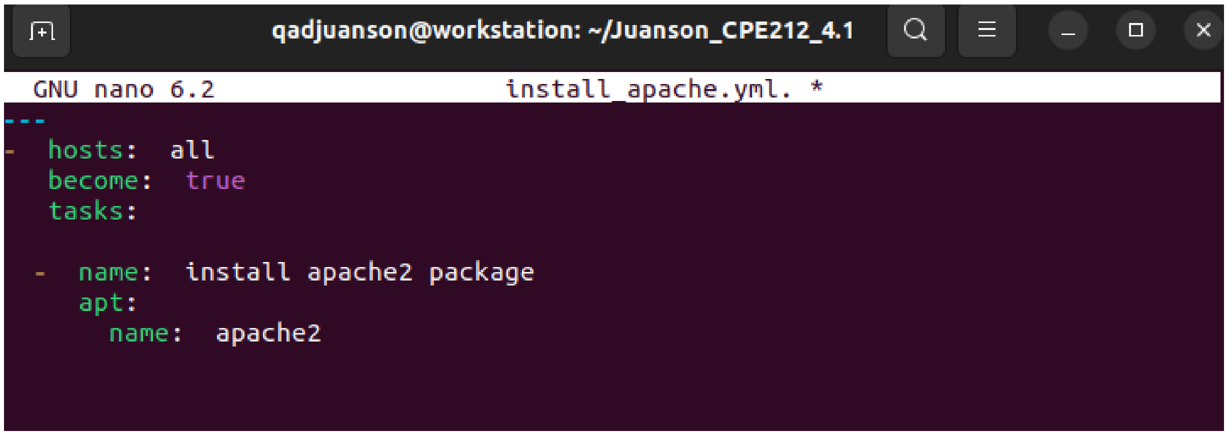
1. With ad hoc commands, we can simplify the administration of remote servers. For example, we can install updates, packages, and applications, etc. However, the real strength of ansible comes from its playbooks. When we write a playbook, we can define the state that we want our servers to be in and the place or commands that ansible will carry out to bring to that state. You can use an editor to create a playbook. Before we proceed, make sure that you are in the directory of the repository that we use in the previous activities (*CPE232_yourname*). Issue the command *nano*

install_apache.yml. This will create a playbook file called *install_apache.yml*. The .yml is the basic standard extension for playbook files.

When the editor appears, type the following:



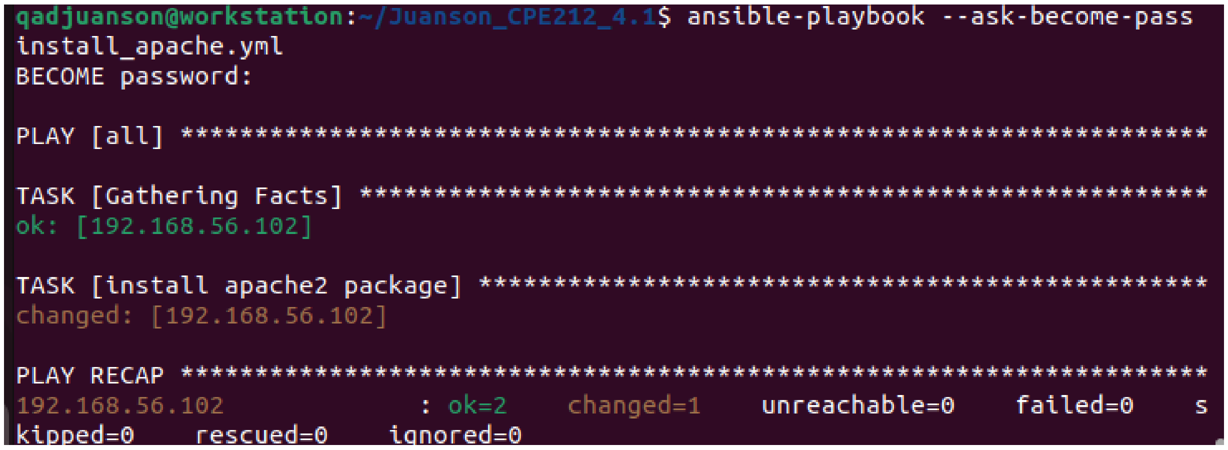
```
  GNU nano 4.8                    install_apache.yml
---
- hosts: all
  become: true
  tasks:

  - name: install apache2 package
    apt:
      name: apache2
```



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

  - name:  install apache2 package
    apt:
      name:  apache2
```

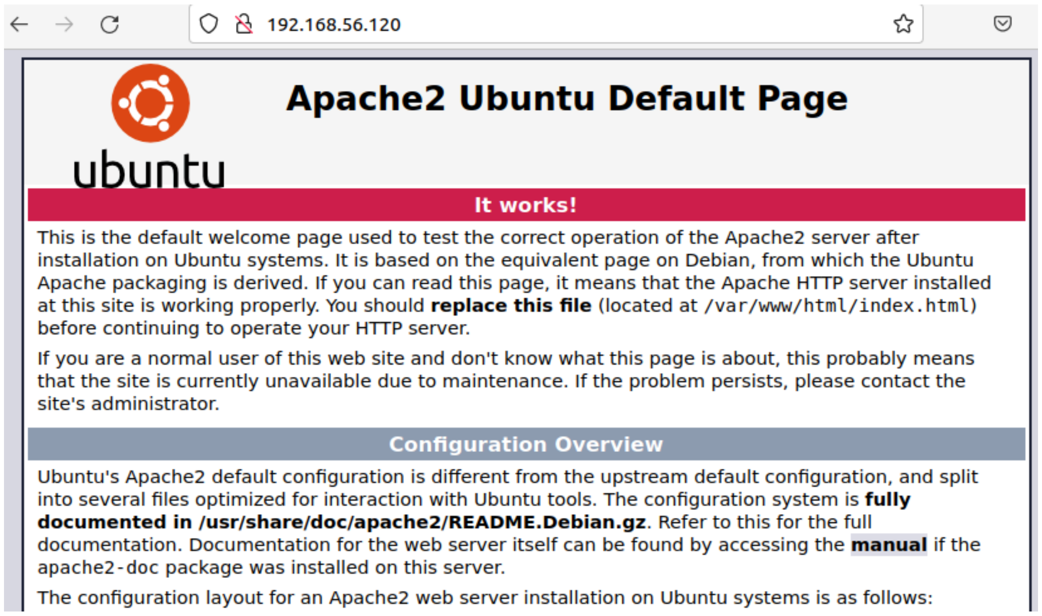
Make sure to save the file. Take note also of the alignments of the texts.

2. Run the yml file using the command: *ansible-playbook --ask-become-pass install_apache.yml.* Describe the result of this command.



```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible-playbook --ask-become-pass
install_apache.yml
BECOME password:

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

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

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

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

3. To verify that apache2 was installed automatically in the remote servers, go to the web browsers on each server and type its IP address. You should see something like this.



Apache2 Ubuntu Default Page

It works!

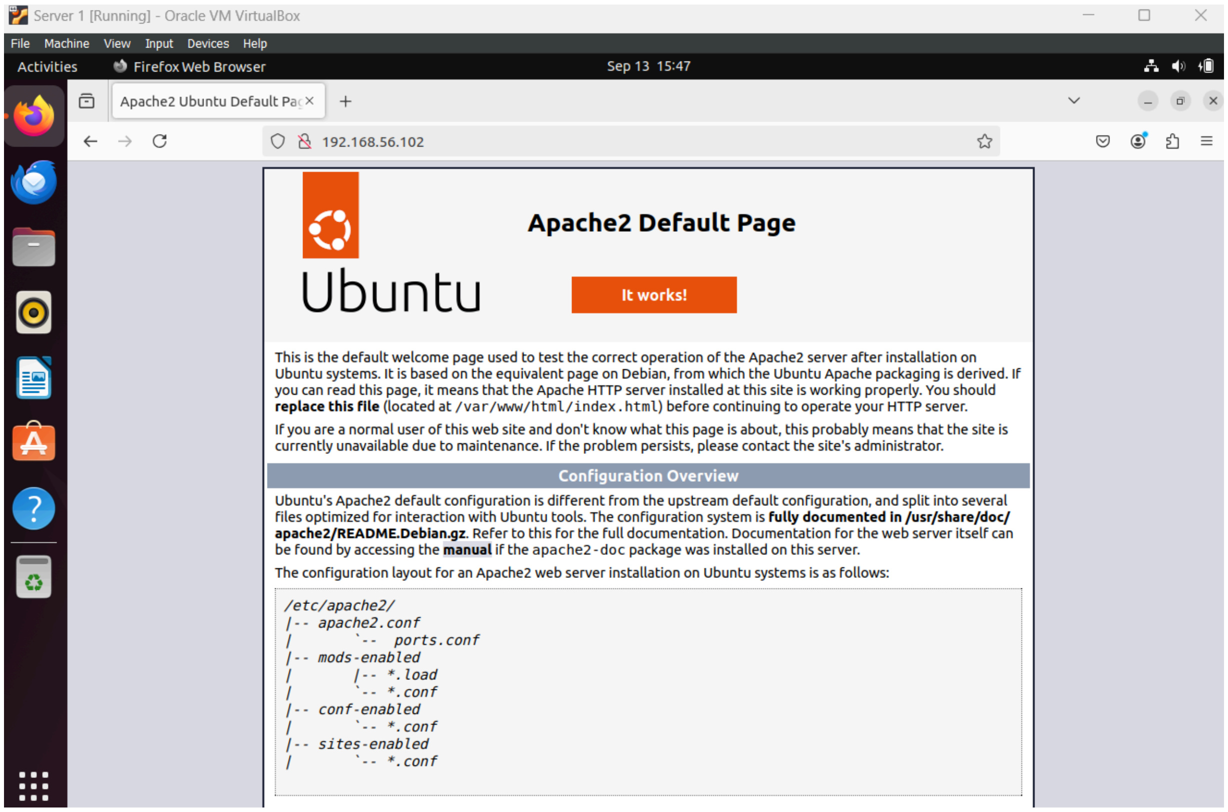
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:


Server 1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Firefox Web Browser
Sep 13 15:47
Apache2 Ubuntu Default Pa
192.168.56.102
Ubuntu

Apache2 Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

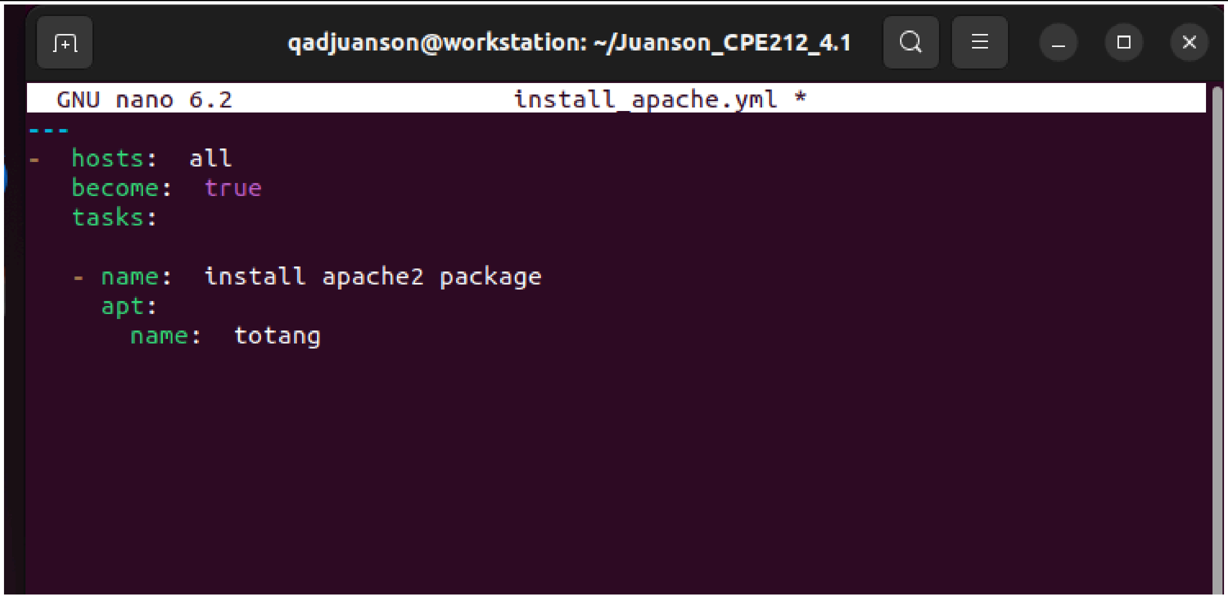
Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

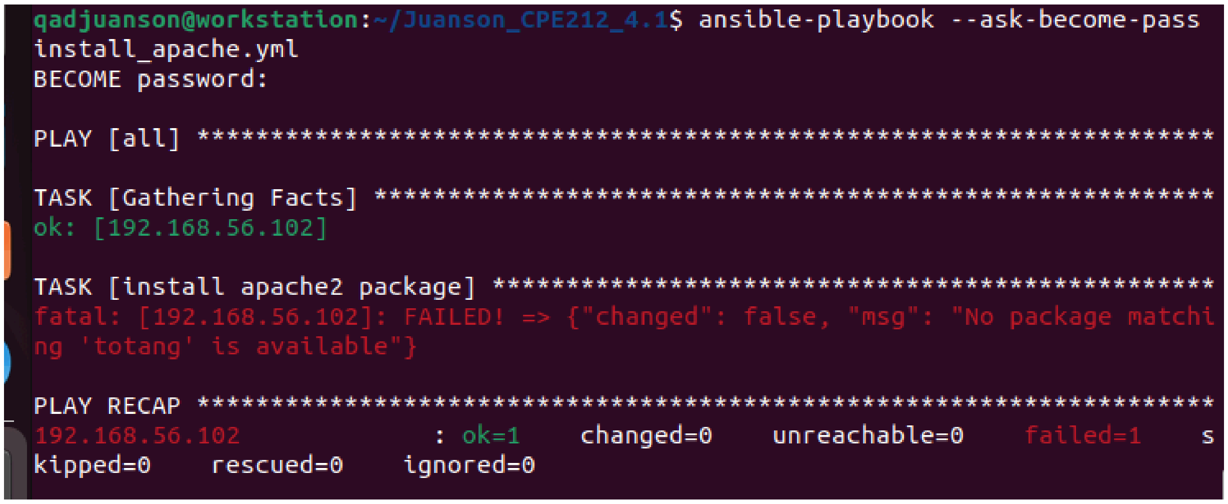
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

4. Try to edit the *install_apache.yml* and change the name of the package to any name that will not be recognized. What is the output?

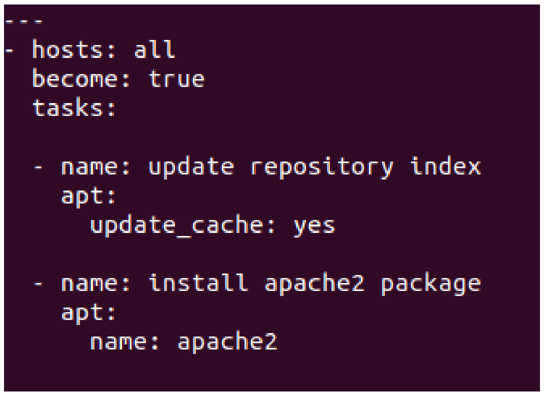


```
  GNU nano 6.2                        install_apache.yml *
---
- hosts:  all
  become:  true
  tasks:

  - name:  install apache2 package
    apt:
      name:  totang
```



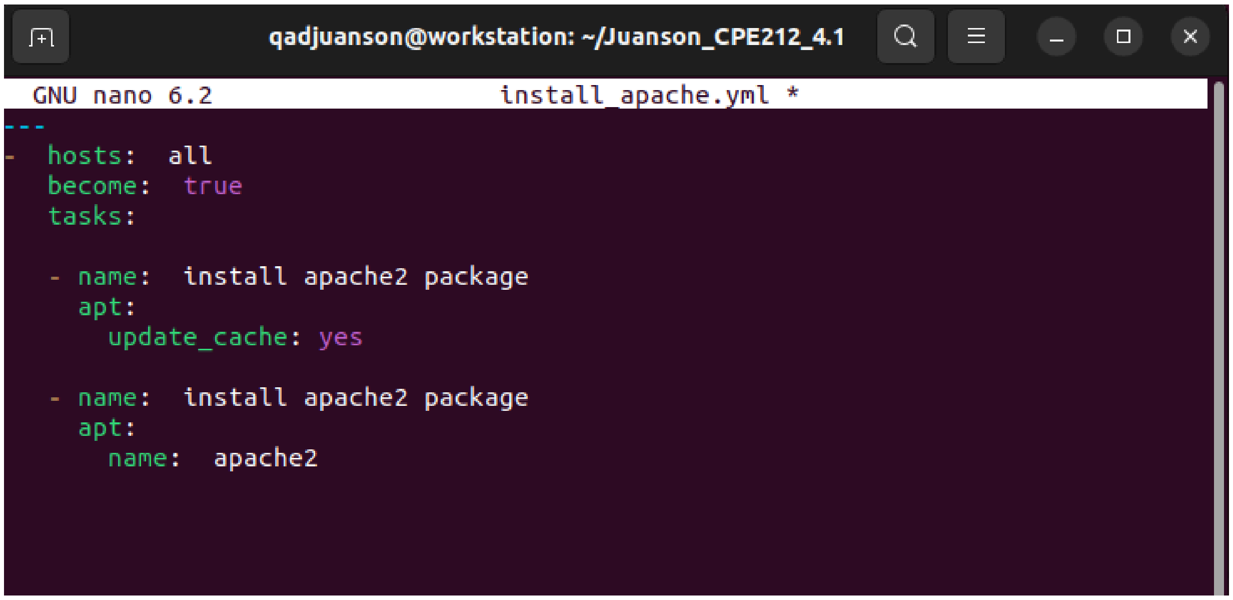
```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible-playbook --ask-become-pass
install_apache.yml
BECOME password:

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

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

TASK [install apache2 package] **************************************************
fatal: [192.168.56.102]: FAILED! => {"changed": false, "msg": "No package matchi
ng 'totang' is available"}

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

5. This time, we are going to put additional task to our playbook. Edit the *install_apache.yml*. As you can see, we are now adding an additional command, which is the *update_cache.* This command updates existing package-indexes on a supporting distro but not upgrading installed-packages (utilities) that were being installed.

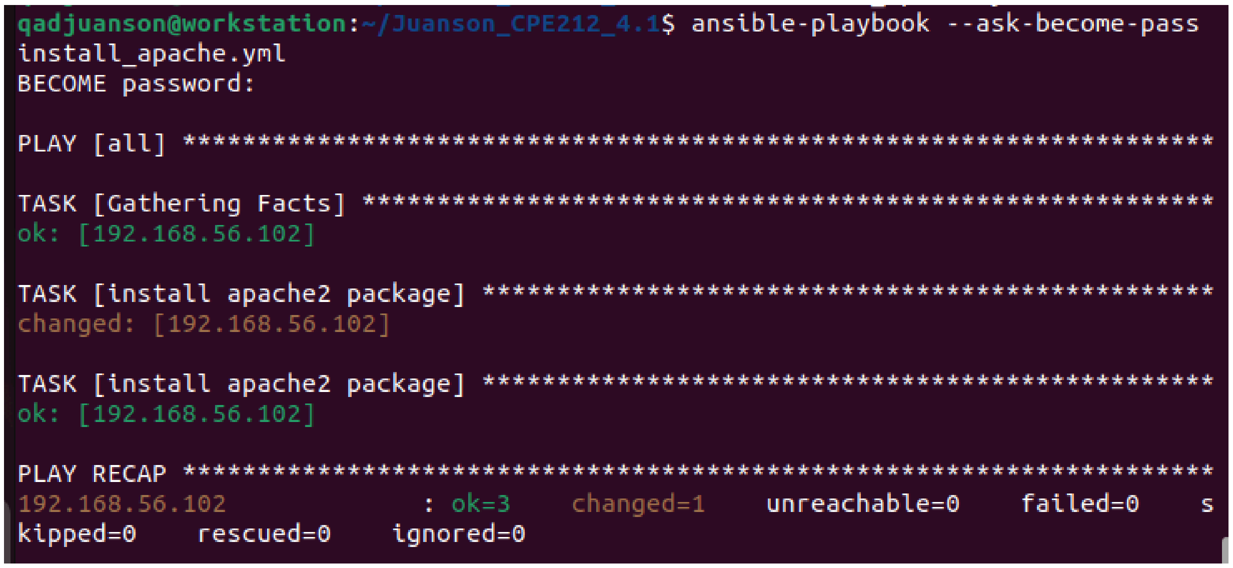


```
---
- hosts: all
  become: true
  tasks:

  - name: update repository index
    apt:
      update_cache: yes

  - name: install apache2 package
    apt:
      name: apache2
```

Save the changes to this file and exit.



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

   - name:  install apache2 package
     apt:
       update_cache: yes

   - name:  install apache2 package
     apt:
       name:  apache2
```

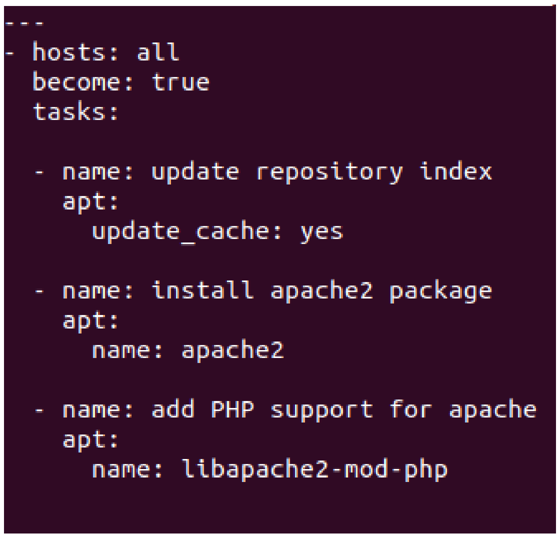


```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible-playbook --ask-become-pass
install_apache.yml
BECOME password:

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

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

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

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

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

6. Run the playbook and describe the output. Did the new command change anything on the remote servers?

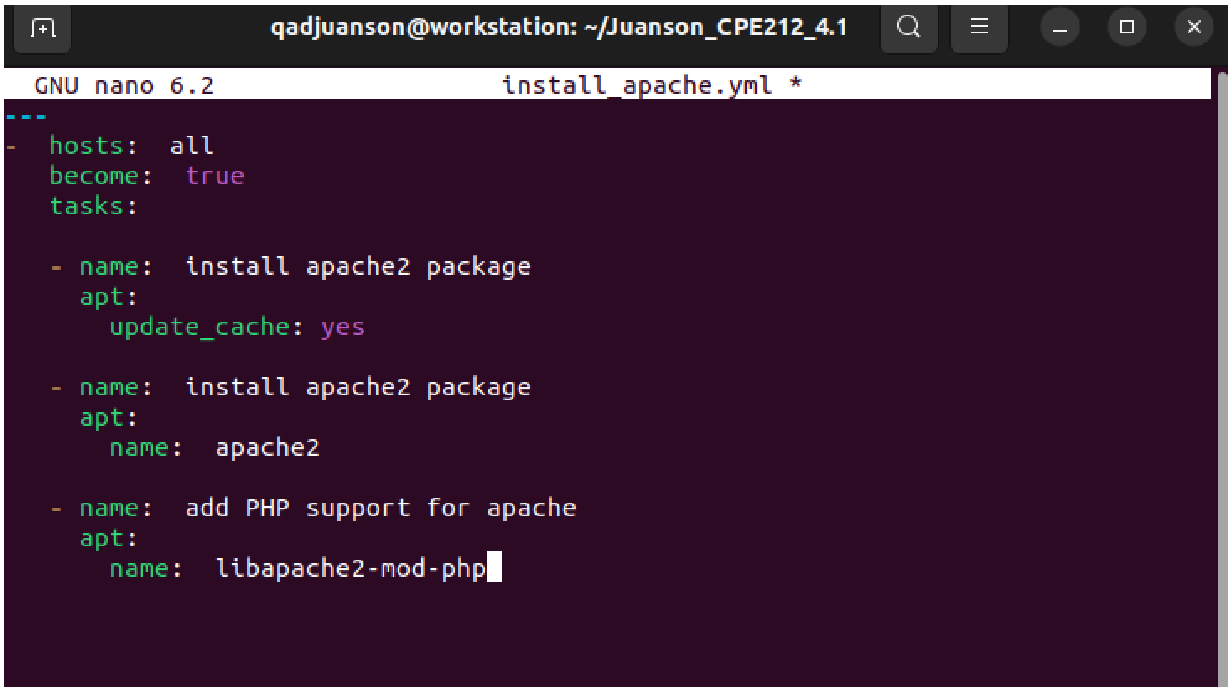
- Yes, the new command updates the package cache.

7. Edit again the *install_apache.yml*. This time, we are going to add a PHP support for the apache package we installed earlier.



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

  - name: update repository index
    apt:
      update_cache: yes

  - name: install apache2 package
    apt:
      name: apache2

  - name: add PHP support for apache
    apt:
      name: libapache2-mod-php
```

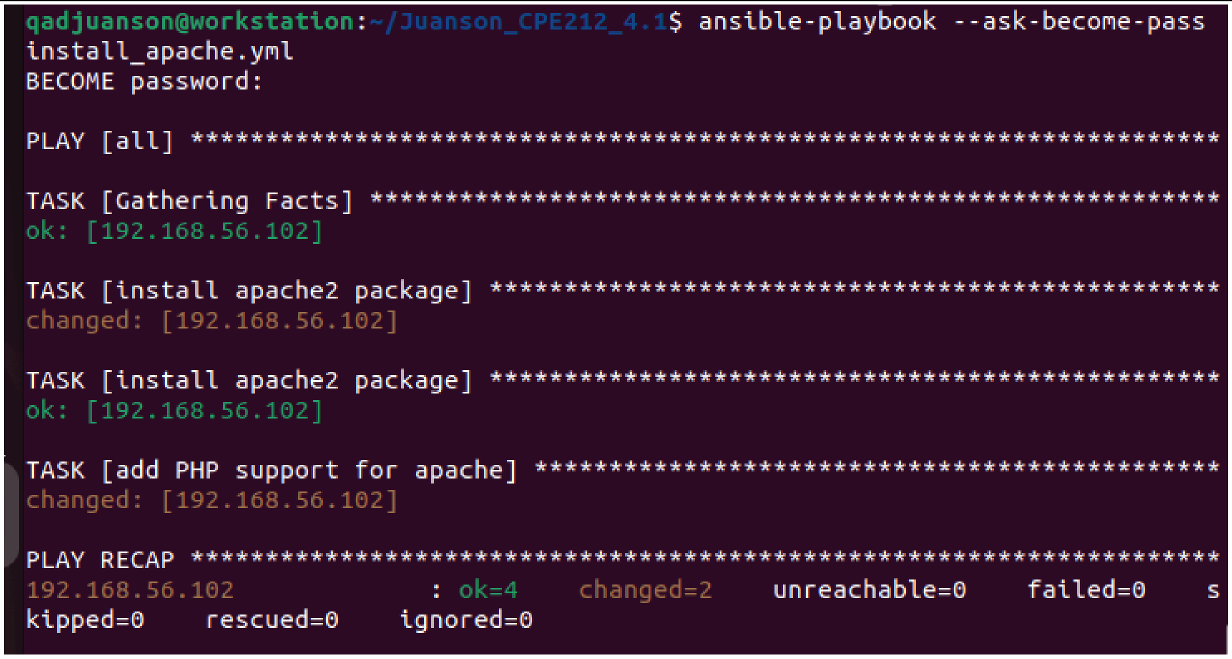
Save the changes to this file and exit.



qadjuanson@workstation: ~/Juanson_CPE212_4.1

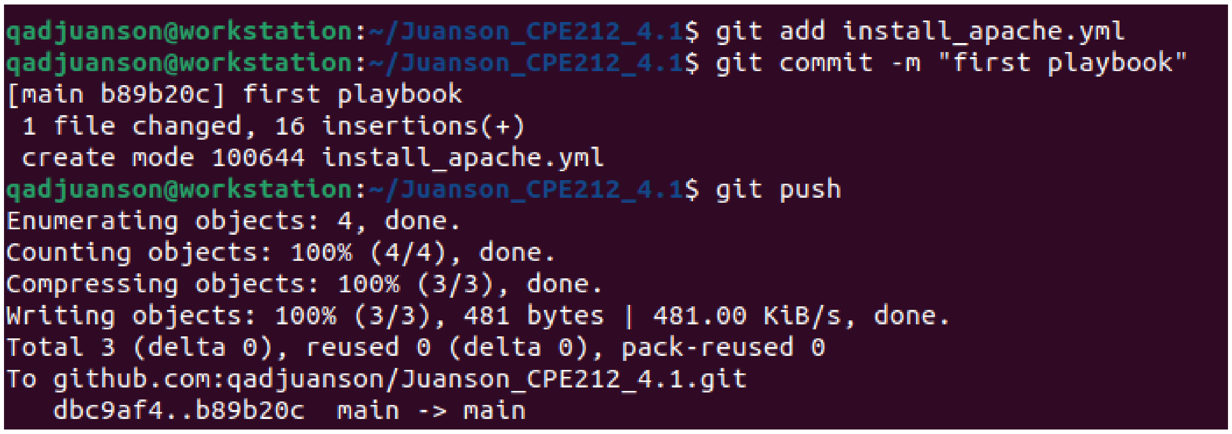
```
GNU nano 6.2                        install_apache.yml *
---
- hosts:  all
  become:  true
  tasks:

  - name:  install apache2 package
    apt:
      update_cache: yes

  - name:  install apache2 package
    apt:
      name:  apache2

  - name:  add PHP support for apache
    apt:
      name:  libapache2-mod-php
```

8. Run the playbook and describe the output. Did the new command change anything on the remote servers?



```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ ansible-playbook --ask-become-pass
install_apache.yml
BECOME password:

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

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

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

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

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

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

9. Finally, make sure that we are in sync with GitHub. Provide the link of your GitHub repository.



```
qadjuanson@workstation:~/Juanson_CPE212_4.1$ git add install_apache.yml
qadjuanson@workstation:~/Juanson_CPE212_4.1$ git commit -m "first playbook"
[main b89b20c] first playbook
 1 file changed, 16 insertions(+)
 create mode 100644 install_apache.yml
qadjuanson@workstation:~/Juanson_CPE212_4.1$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 481 bytes | 481.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:qadjuanson/Juanson_CPE212_4.1.git
   dbc9af4..b89b20c  main -> main
```

Reflections:

Answer the following:

1. What is the importance of using a playbook?
   - A playbook ensures tasks are done consistently, speeds up processes, and reduces errors by providing clear steps. It simplifies training, clarifies responsibilities, and helps teams maintain quality while improving over time.

2. Summarize what we have done on this activity.

- There’s a lot of things that I’ve done in this activity; By installing ansible, installing VIM, creating/cloning a repository and creating a playbook. This activity is quite challenging for me because there’s a lot of errors that I’ve encountered while doing this activity.