

<b>Name:</b> Mark Andrei Ponayo	<b>Date Performed:</b> October 14, 2023
<b>Course/Section:</b> BSCPE31S5	<b>Date Submitted:</b> October 14, 2023
<b>Instructor:</b> Engr. Roman Richard	<b>Semester and SY:</b> 1 <sup>st</sup> sem 2023 - 2024
<b>Activity 6: Targeting Specific Nodes and Managing Services</b>	
<b>1. Objectives:</b> <ul style="list-style-type: none"> <li>1.1 Individualize hosts</li> <li>1.2 Apply tags in selecting plays to run</li> <li>1.3 Managing Services from remote servers using playbooks</li> </ul>	
<b>2. Discussion:</b> <p>In this activity, we try to individualize hosts. For example, we don't want apache on all our servers, or maybe only one of our servers is a web server, or maybe we have different servers like database or file servers running different things on different categories of servers and that is what we are going to take a look at in this activity.</p> <p>We also try to manage services that do not automatically run using the automations in playbook. For example, when we install web servers or httpd for CentOS, we notice that the service did not start automatically.</p> <p><b>Requirement:</b></p> <p>In this activity, you will need to create another Ubuntu VM and name it Server 3. Likewise, you need to activate the second adapter to a host-only adapter after the installations. Take note of the IP address of the Server 3. Make sure to use the command <i>ssh-copy-id</i> to copy the public key to Server 3. Verify if you can successfully SSH to Server 3.</p>	
<b>Task 1: Targeting Specific Nodes</b>	
<ul style="list-style-type: none"> <li>1. Create a new playbook and named it site.yml. Follow the commands as shown in the image below. Make sure to save the file and exit.</li> </ul>	

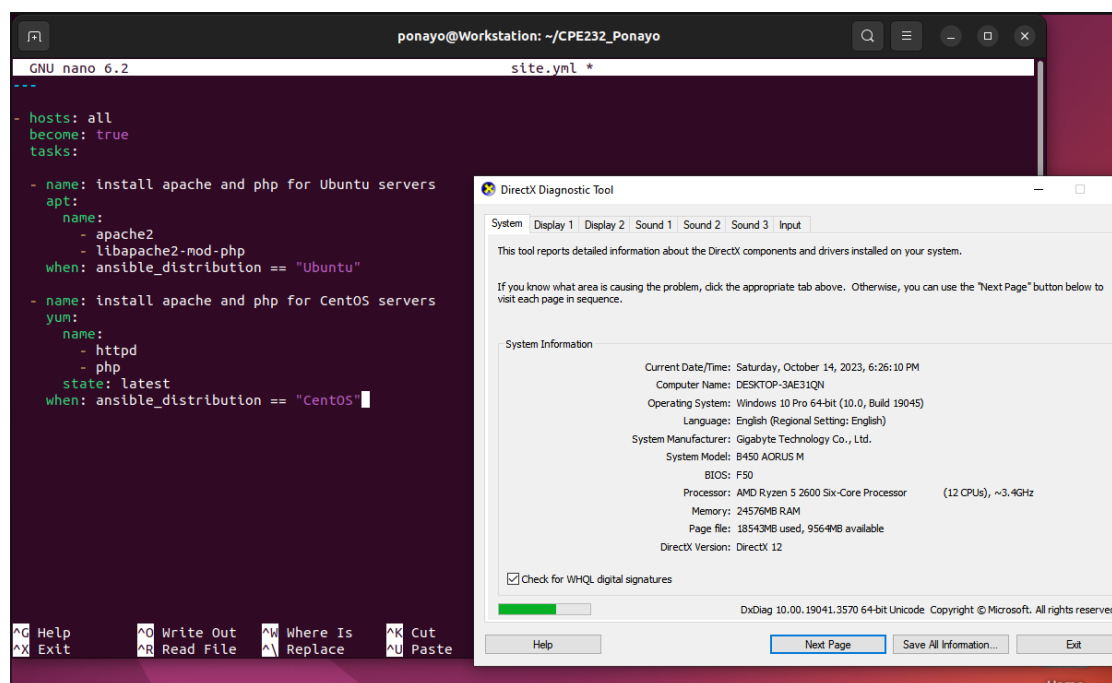
```

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

    - name: install apache and php for Ubuntu servers
      apt:
        name:
          - apache2
          - libapache2-mod-php
        state: latest
        update_cache: yes
        when: ansible_distribution == "Ubuntu"

    - name: install apache and php for CentOS servers
      dnf:
        name:
          - httpd
          - php
        state: latest
        when: ansible_distribution == "CentOS"

```



2. Edit the inventory file. Remove the variables we put in our last activity and group according to the image shown below:

```
[web_servers]
192.168.56.120
192.168.56.121

[db_servers]
192.168.56.122

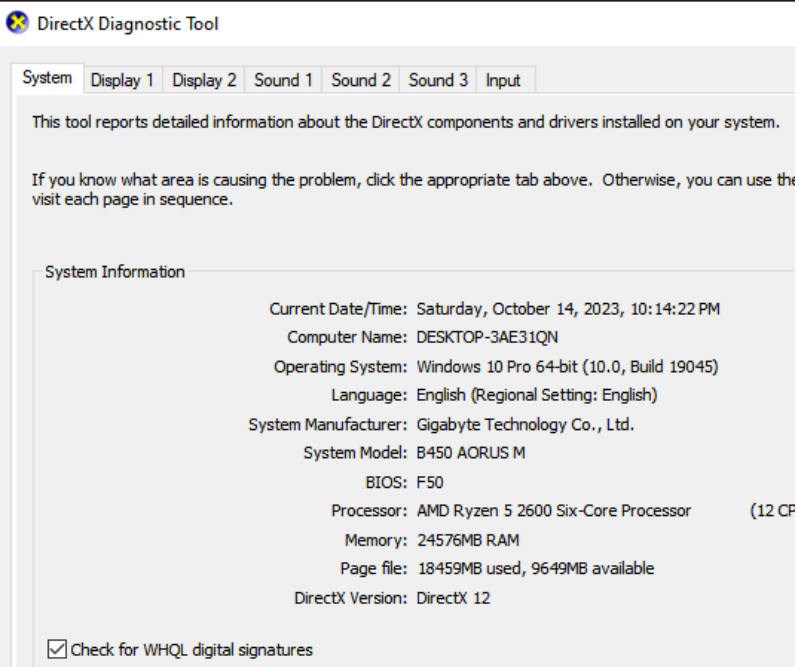
[file_servers]
192.168.56.123
```

Make sure to save the file and exit.

```
[web_servers]
192.168.56.110
192.168.56.112

[db_servers]
192.168.56.110
192.168.56.112

[file_servers]
192.168.56.112
```



The screenshot shows the DirectX Diagnostic Tool window. The 'System' tab is selected. The window displays the following system information:

- Current Date/Time: Saturday, October 14, 2023, 10:14:22 PM
- Computer Name: DESKTOP-3AE31QN
- Operating System: Windows 10 Pro 64-bit (10.0, Build 19045)
- Language: English (Regional Setting: English)
- System Manufacturer: Gigabyte Technology Co., Ltd.
- System Model: B450 AORUS M
- BIOS: F50
- Processor: AMD Ryzen 5 2600 Six-Core Processor (12 CPUs)
- Memory: 24576MB RAM
- Page file: 18459MB used, 9649MB available
- DirectX Version: DirectX 12

There is a checkbox labeled 'Check for WHQL digital signatures' which is checked.

Right now, we have created groups in our inventory file and put each server in its own group. In other cases, you can have a server be a member of multiple groups, for example you have a test server that is also a web server.

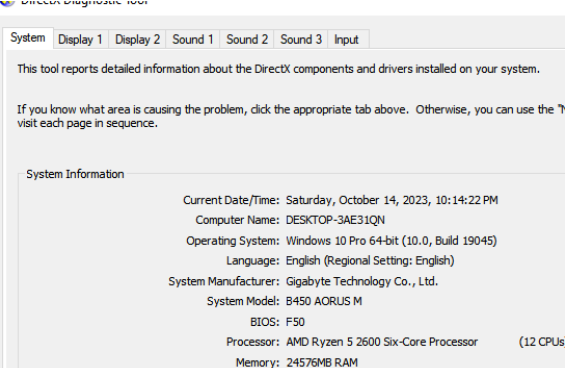
```
ponayo@Workstation:~/CPE232_Ponayo$ sudo nano inventory
ponayo@Workstation:~/CPE232_Ponayo$ sudo nano inventory
ponayo@Workstation:~/CPE232_Ponayo$ ansible-playbook -i
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.110]
ok: [192.168.56.112]

TASK [install updates (CentOS)] *****
skipping: [192.168.56.112]
ok: [192.168.56.110]

TASK [install updates (Ubuntu)] *****
skipping: [192.168.56.110]
ok: [192.168.56.112]
```



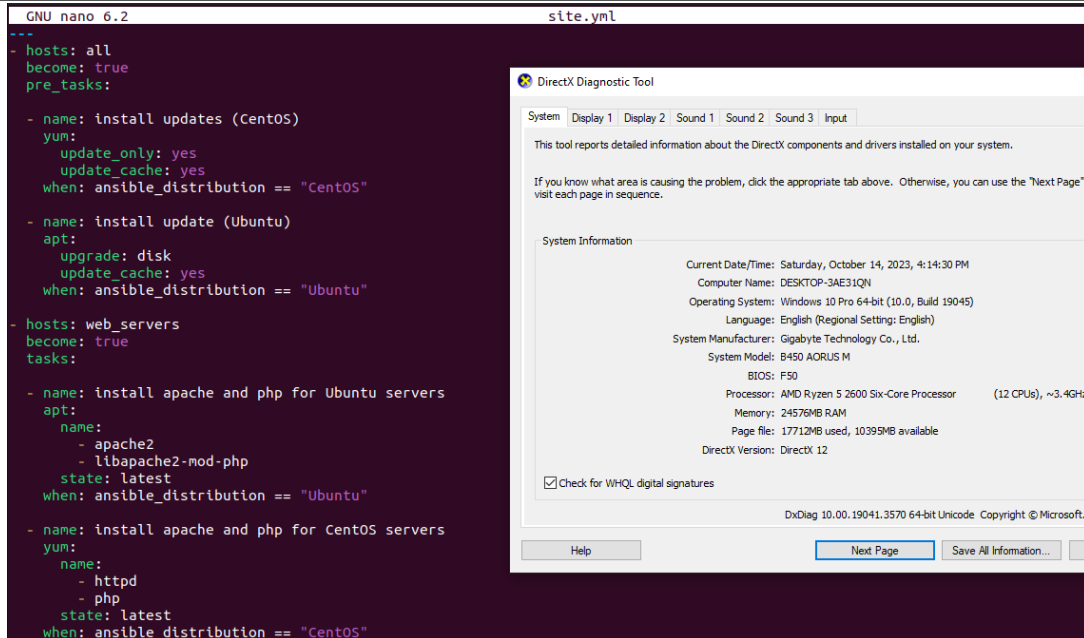
This is a duplicate of the screenshot shown in the previous block, displaying the same system information from the DirectX Diagnostic Tool.

3. Edit the *site.yml* by following the image below:

```
---
- hosts: all
  become: true
  pre_tasks:
    - name: install updates (CentOS)
      dnf:
        update_only: yes
        update_cache: yes
        when: ansible_distribution == "CentOS"
    - name: install updates (Ubuntu)
      apt:
        upgrade: dist
        update_cache: yes
        when: ansible_distribution == "Ubuntu"

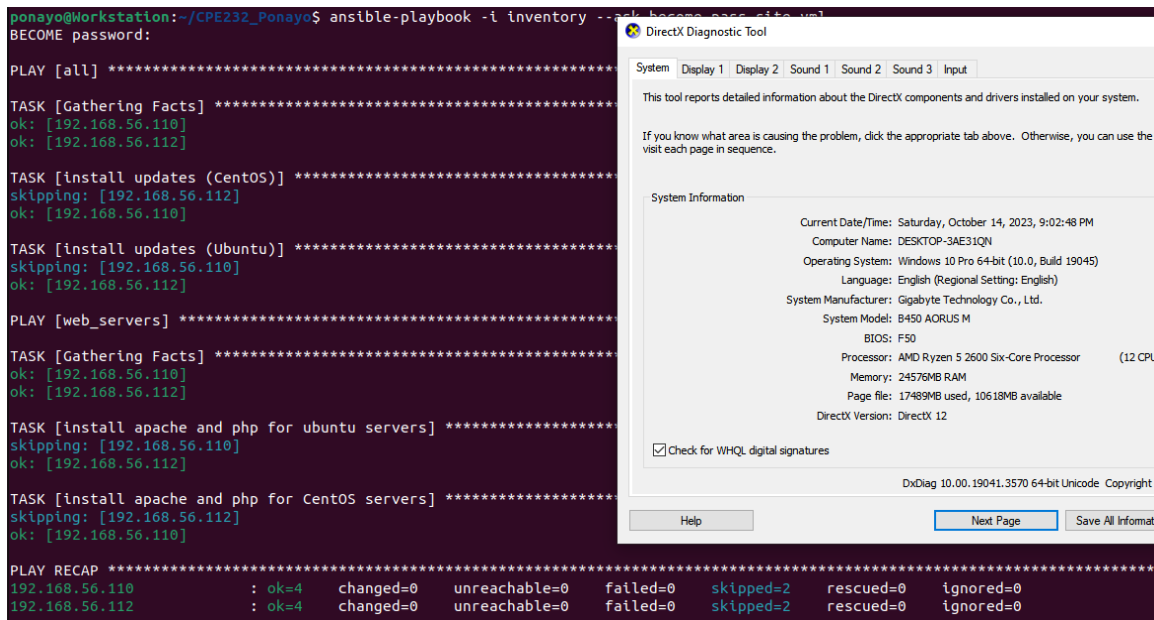
- hosts: web_servers
  become: true
  tasks:
    - name: install apache and php for Ubuntu servers
      apt:
        name:
          - apache2
          - libapache2-mod-php
        state: latest
        when: ansible_distribution == "Ubuntu"
    - name: install apache and php for CentOS servers
      dnf:
        name:
          - httpd
          - php
        state: latest
        when: ansible_distribution == "CentOS"
```

Make sure to save the file and exit.



The *pre-tasks* command tells the ansible to run it before any other thing. In the *pre-tasks*, CentOS will install updates while Ubuntu will upgrade its distribution package. This will run before running the second play, which is targeted at *web\_servers*. In the second play, apache and php will be installed on both Ubuntu servers and CentOS servers.

Run the *site.yml* file and describe the result.



We can install the and update the apache and php by running the `ansible-playbook --ask-become-pass site.yml`. But, i added the “-i inventory” on the command since my hosts are under the “inventory” file.

4. Let's try to edit again the *site.yml* file. This time, we are going to add plays targeting the other servers. This time we target the *db\_servers* by adding it on the current *site.yml*. Below is an example: (Note add this at the end of the playbooks from task 1.3).

```
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    yum:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

  - name: "Mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true

  - name: install mariadb package (Ubuntu)
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"
```

Make sure to save the file and exit.

```
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    yum:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

  - name: install mariadb package (Ubuntu)
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"

  - name: "Mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true
```

Run the *site.yml* file and describe the result.

```
skipping: [192.168.56.110]
ok: [192.168.56.112]

PLAY [web_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.110]
ok: [192.168.56.112]

TASK [install apache and php for ubuntu servers] *****
skipping: [192.168.56.110]
ok: [192.168.56.112]

TASK [install apache and php for CentOS servers] *****
skipping: [192.168.56.110]
ok: [192.168.56.112]

PLAY [db_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.110]
ok: [192.168.56.112]

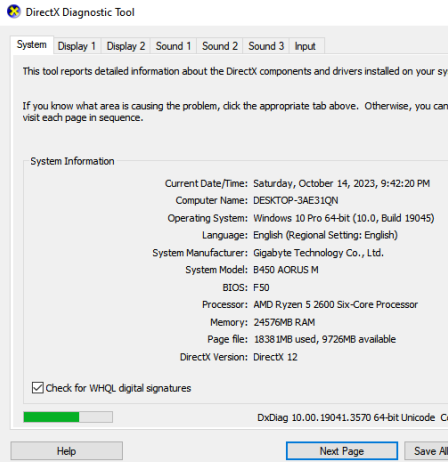
TASK [install mariadb package (CentOS)] *****
skipping: [192.168.56.110]
ok: [192.168.56.112]

TASK [install mariadb package (Ubuntu)] *****
skipping: [192.168.56.110]
changed: [192.168.56.112]

TASK [Mariadb- Restarting/Enabling] *****
changed: [192.168.56.112]
changed: [192.168.56.110]

PLAY RECAP *****
192.168.56.110 : ok=7 changed=1 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0
192.168.56.112 : ok=7 changed=2 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0

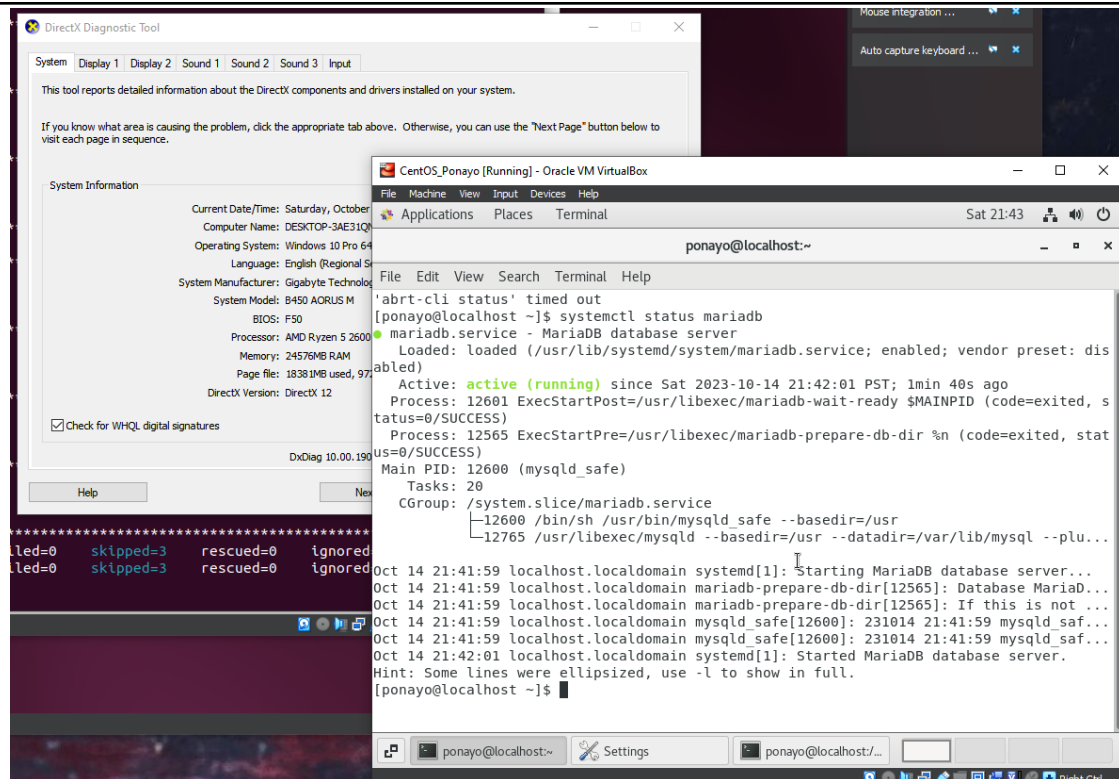
ponayo@Workstation:~/CPE232_Ponayo$
```



The screenshot shows the DirectX Diagnostic Tool window. It has tabs for System, Display 1, Display 2, Sound 1, Sound 2, Sound 3, and Input. The System tab is selected. The window displays system information including: Current Date/Time: Saturday, October 14, 2023, 9:42:20 PM; Computer Name: DESKTOP-3AE31QN; Operating System: Windows 10 Pro 64-bit (10.0, Build 19045); Language: English (Regional Setting: English); System Manufacturer: Gigabyte Technology Co., Ltd.; System Model: B450 AORUS M; BIOS: F50; Processor: AMD Ryzen 5 2600 Six-Core Processor; Memory: 24576MB RAM; Page file: 1838 MB used, 9726MB available; DirectX Version: DirectX 12. There is a checkbox for 'Check for WHQL digital signatures' which is checked. At the bottom, there are buttons for 'Help', 'Next Page', and 'Save All'.

As running the ansible-playbook -i inventory -ask-become-pass site.yml. I was able to successfully update the apache php and also install the mariadb package in both Ubuntu and CentOS. And I was also able to restart or enable the mariadb in the last task.

5. Go to the remote server (Ubuntu) terminal that belongs to the db\_servers group and check the status for mariadb installation using the command: *systemctl status mariadb*. Do this on the CentOS server also.



Describe the output.

After inputting the `sudo systemctl mariadb` in the CentOS server. We can see the MariadB database server and we can also see if the mariadb is active (running) or not.

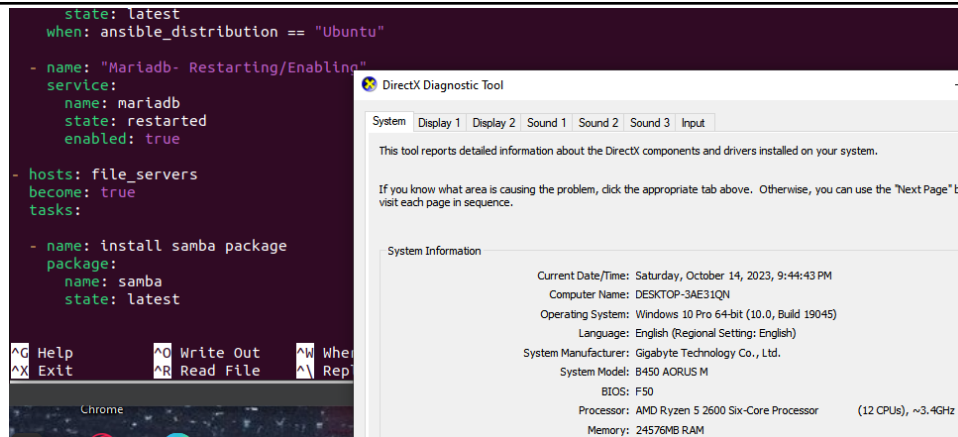
6. Edit the `site.yml` again. This time we will append the code to configure installation on the `file_servers` group. We can add the following on our file.

```
- hosts: file_servers
  become: true
  tasks:

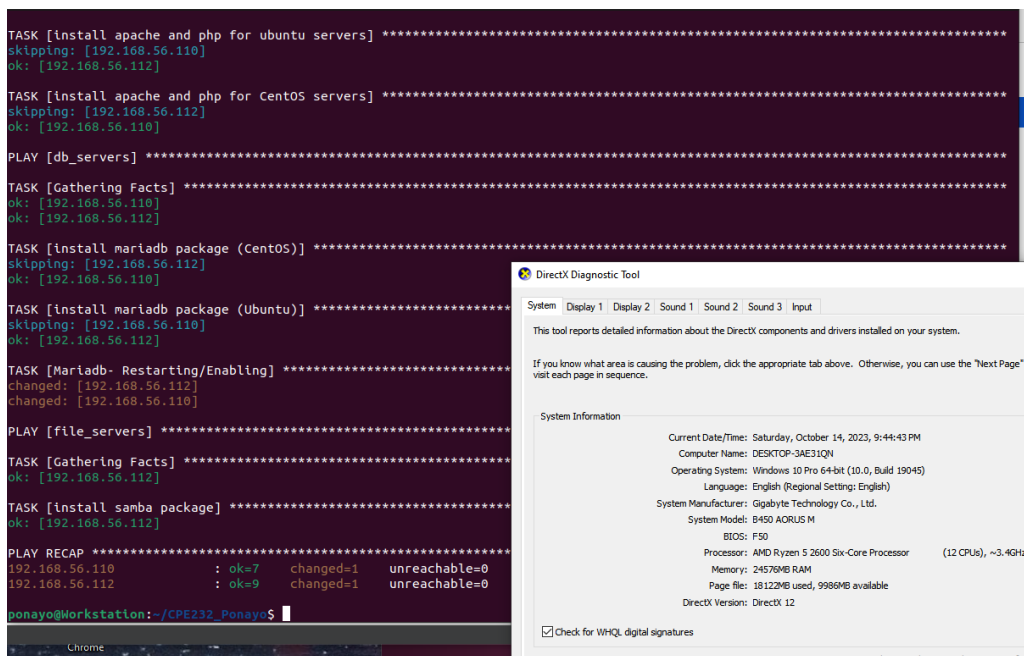
  - name: install samba package
    package:
      name: samba
      state: latest
```

Make sure to save the file and exit.





Run the *site.yml* file and describe the result.



As running the command. I successfully run the script by adding the *file\_servers* group and installing the samba package under the *site.yml* file.

The testing of the *file\_servers* is beyond the scope of this activity, and as well as our topics and objectives. However, in this activity we were able to show that we can target hosts or servers using grouping in ansible playbooks.

## Task 2: Using Tags in running playbooks

In this task, our goal is to add metadata to our plays so that we can only run the plays that we want to run, and not all the plays in our playbook.

1. Edit the *site.yml* file. Add tags to the playbook. After the name, we can place the tags: *name\_of\_tag*. This is an arbitrary command, which means you can use any name for a tag.

```
---  
  
- hosts: all  
  become: true  
  pre_tasks:  
  
    - name: install updates (CentOS)  
      tags: always  
      dnf:  
        update_only: yes  
        update_cache: yes  
        when: ansible_distribution == "CentOS"  
  
    - name: install updates (Ubuntu)  
      tags: always  
      apt:  
        upgrade: dist  
        update_cache: yes  
        when: ansible_distribution == "Ubuntu"
```

```
- hosts: web_servers  
  become: true  
  tasks:  
  
    - name: install apache and php for Ubuntu servers  
      tags: apache,apache2,ubuntu  
      apt:  
        name:  
          - apache2  
          - libapache2-mod-php  
        state: latest  
        when: ansible_distribution == "Ubuntu"  
  
    - name: install apache and php for CentOS servers  
      tags: apache,centos,httpd  
      dnf:  
        name:  
          - httpd  
          - php  
        state: latest  
        when: ansible_distribution == "CentOS"
```

```
- hosts: db_servers
  become: true
  tasks:

    - name: install mariadb package (CentOS)
      tags: centos, db, mariadb
      dnf:
        name: mariadb-server
        state: latest
      when: ansible_distribution == "CentOS"

    - name: "Mariadb- Restarting/Enabling"
      service:
        name: mariadb
        state: restarted
        enabled: true

    - name: install mariadb package (Ubuntu)
      tags: db, mariadb, ubuntu
      apt:
        name: mariadb-server
        state: latest
      when: ansible_distribution == "Ubuntu"

- hosts: file_servers
  become: true
  tasks:

    - name: install samba package
      tags: samba
      package:
        name: samba
        state: latest
```

Make sure to save the file and exit.

```

- hosts: all
  become: true
  pre_tasks:

  - name: install updates (CentOS)
    tags: always
    yum:
      update_only: yes
      update_cache: yes
    when: ansible_distribution == "CentOS"

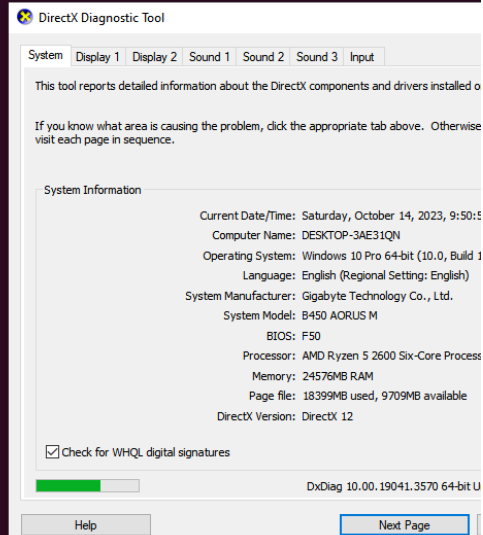
  - name: install updates (Ubuntu)
    tags: always
    apt:
      upgrade: dist
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

- hosts: web_servers
  become: true
  tasks:

  - name: install apache and php for ubuntu servers
    tags: apache,apache2,ubuntu
    apt:
      name:
        - apache2
        - libapache2-mod-php
      state: latest
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

  - name: install apache and php for CentOS servers
    tags: apache,centos,httpd
    yum:
      name:

```



```

GNU nano 6.2 site.yml *
- name: install apache and php for CentOS servers
  tags: apache,centos,httpd
  yum:
    name:
      - httpd
      - php
    state: latest
  when: ansible_distribution == "CentOS"

- hosts: db_servers
  become: true
  tasks:

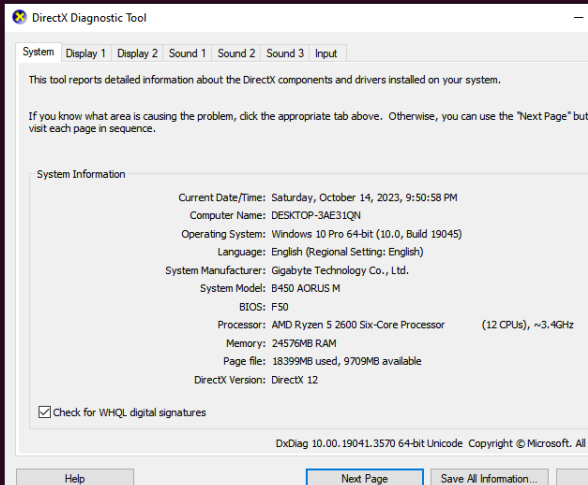
  - name: install mariadb package (CentOS)
    tags: centos,db,mariadb
    yum:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

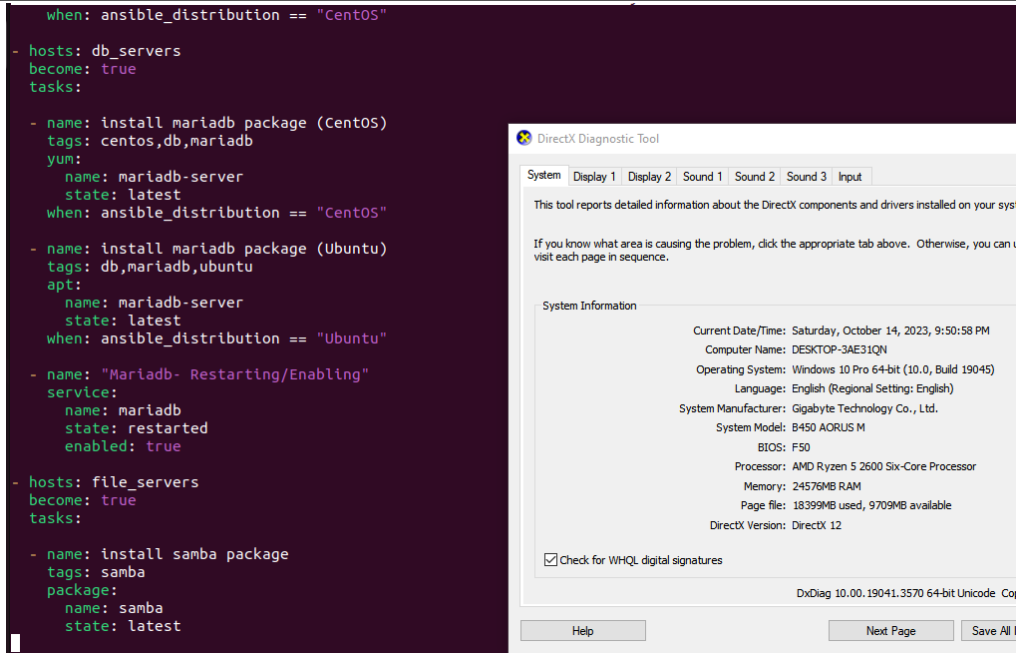
  - name: install mariadb package (Ubuntu)
    tags: db,mariadb,ubuntu
    apt:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "Ubuntu"

  - name: "Mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true

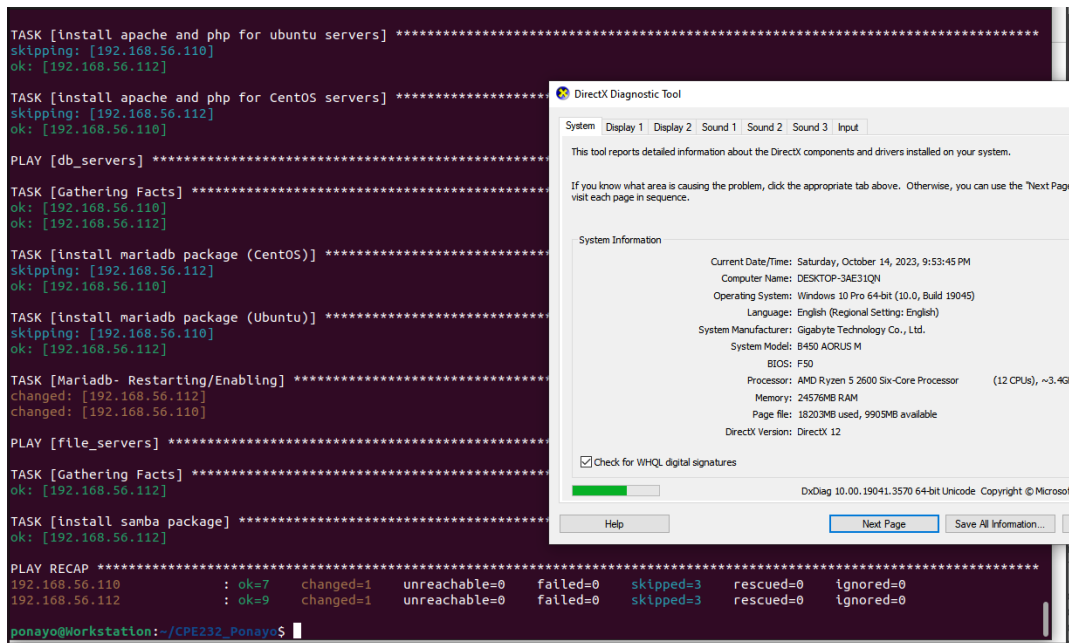
- hosts: file_servers
  become: true
  tasks:

```





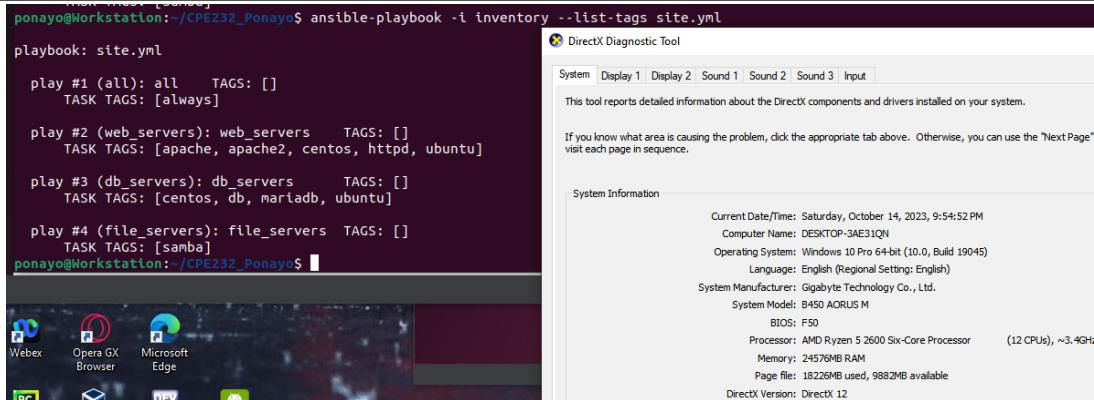
Run the *site.yml* file and describe the result.



After adding the “tags” for each task and running it. We can successfully added a arbitrary command for each task .

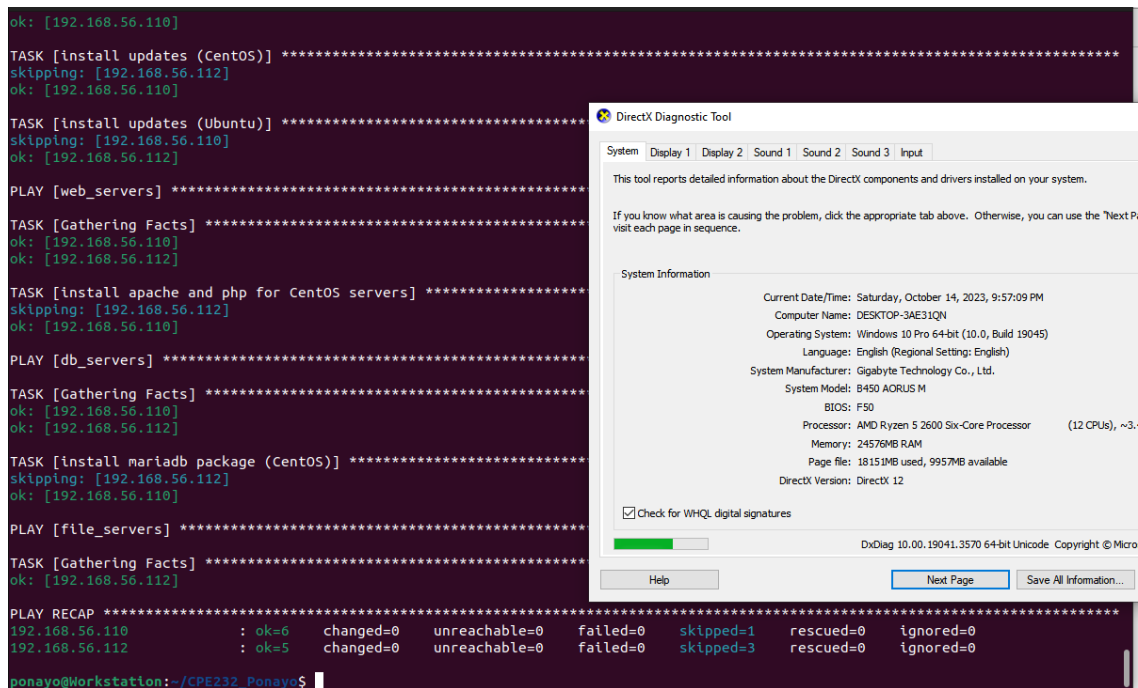
2. On the local machine, try to issue the following commands and describe each result:

*2.1 ansible-playbook --list-tags site.yml*



For this example, by adding the "--list-tags" on the command it will show the lists of tags that we added on the scripts.

## 2.2 ansible-playbook --tags centos --ask-become-pass site.yml



For this example, by adding the "--tags-centos" on the command. It will show the tasks that have the "centos" tags.

## 2.3 ansible-playbook --tags db --ask-become-pass site.yml

```
ok: [192.168.56.112]

TASK [install updates (CentOS)] *****
skipping: [192.168.56.112]
ok: [192.168.56.110]

TASK [install updates (Ubuntu)] *****
skipping: [192.168.56.110]
ok: [192.168.56.112]

PLAY [web_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.110]
ok: [192.168.56.112]

PLAY [db_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.112]
ok: [192.168.56.110]

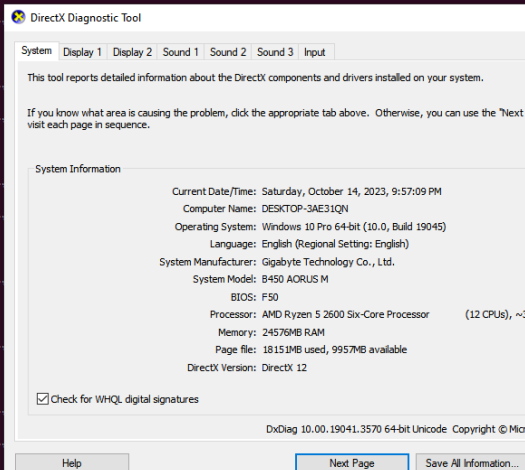
TASK [install mariadb package (CentOS)] *****
skipping: [192.168.56.112]
ok: [192.168.56.110]

TASK [install mariadb package (Ubuntu)] *****
skipping: [192.168.56.110]
ok: [192.168.56.112]

PLAY [file_servers] *****

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

PLAY RECAP *****
192.168.56.110 : ok=5 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignored=0
192.168.56.112 : ok=6 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignored=0
```



For this example, by adding the “--tags db” on the command. It will show the tasks that have the db tags.

## 2.4 ansible-playbook --tags apache --ask-become-pass site.yml

```
ok: [192.168.56.112]

TASK [install updates (CentOS)] *****
skipping: [192.168.56.112]
ok: [192.168.56.110]

TASK [install updates (Ubuntu)] *****
skipping: [192.168.56.110]
ok: [192.168.56.112]

PLAY [web_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.110]
ok: [192.168.56.112]

TASK [install apache and php for ubuntu servers] *****
skipping: [192.168.56.110]
ok: [192.168.56.112]

TASK [install apache and php for CentOS servers] *****
skipping: [192.168.56.112]
ok: [192.168.56.110]

PLAY [db_servers] *****

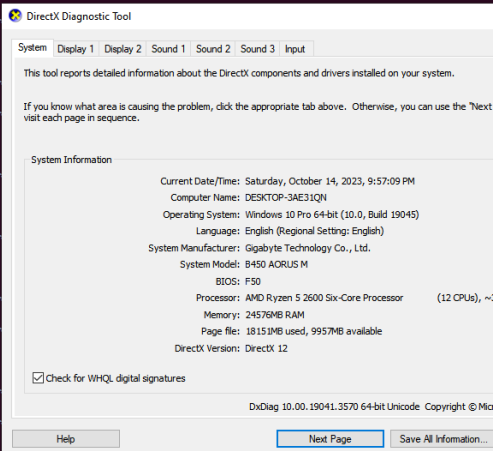
TASK [Gathering Facts] *****
ok: [192.168.56.110]
ok: [192.168.56.112]

PLAY [file_servers] *****

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

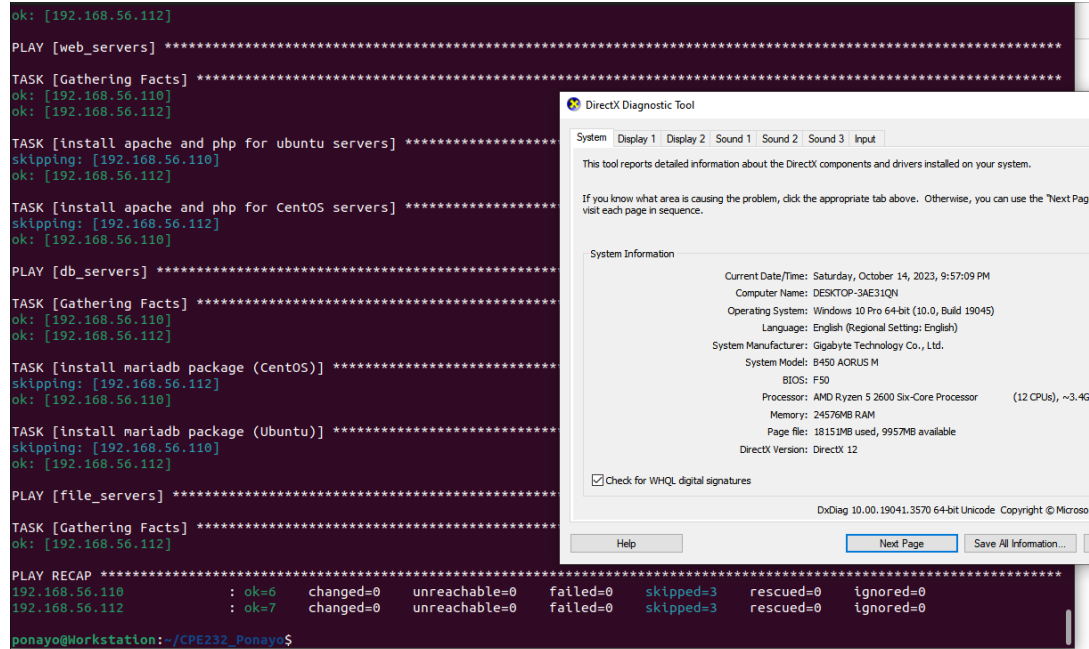
PLAY RECAP *****
192.168.56.110 : ok=5 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignored=0
192.168.56.112 : ok=6 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignored=0

ponayo@Workstation: ~/CPE232_Ponayo$
```



For this example, by adding the “--tags apache” on the command. It will show the tasks that have the apache tags.

## 2.5 ansible-playbook --tags “apache,db” --ask-become-pass site.yml



For this example, by adding the “--tags “apache,db”” on the command. It will show the tasks that have the “apache,db” tags.

### Task 3: Managing Services

1. Edit the file site.yml and add a play that will automatically start the httpd on CentOS server.

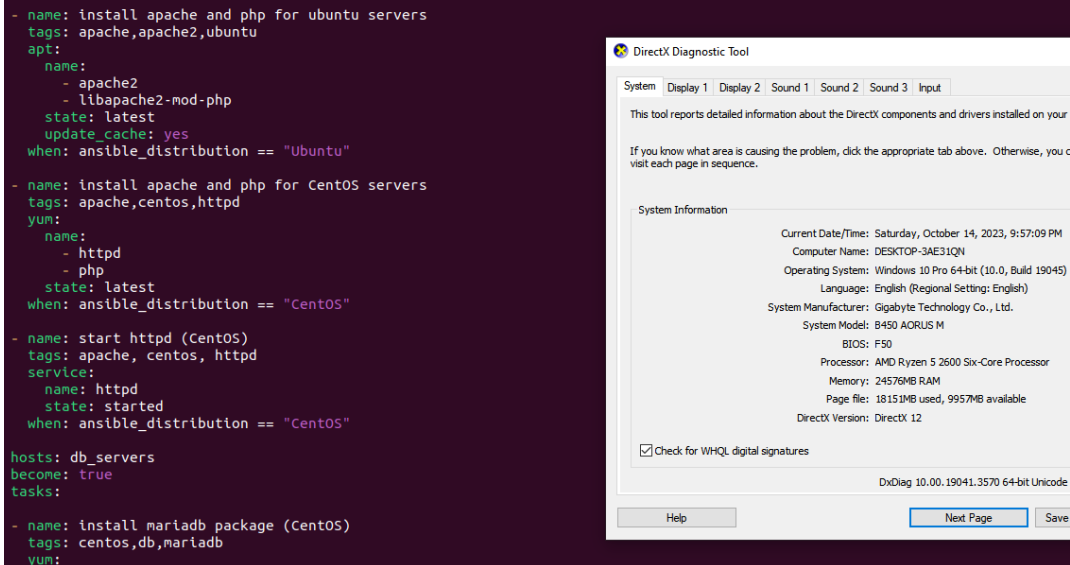
```
- name: install apache and php for CentOS servers
  tags: apache,centos,httpd
  dnf:
    name:
      - httpd
      - php
    state: latest
    when: ansible_distribution == "CentOS"

- name: start httpd (CentOS)
  tags: apache, centos,httpd
  service:
    name: httpd
    state: started
    when: ansible_distribution == "CentOS"
```

Figure 3.1.1

Make sure to save the file and exit.





You would also notice from our previous activity that we already created a module that runs a service.

```
- hosts: db_servers
  become: true
  tasks:

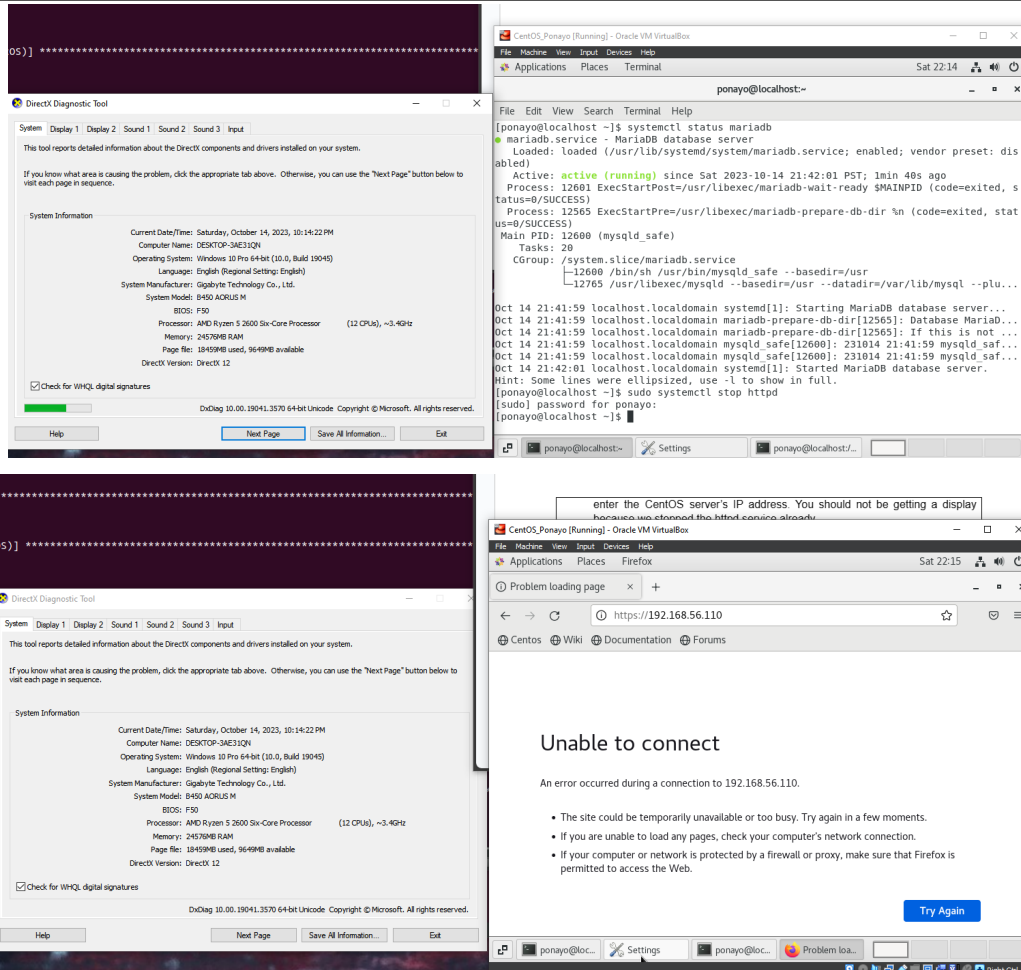
    - name: install mariadb package (CentOS)
      tags: centos, db, mariadb
      dnf:
        name: mariadb-server
        state: latest
      when: ansible_distribution == "CentOS"

    - name: "Mariadb- Restarting/Enabling"
      service:
        name: mariadb
        state: restarted
        enabled: true
```

Figure 3.1.2

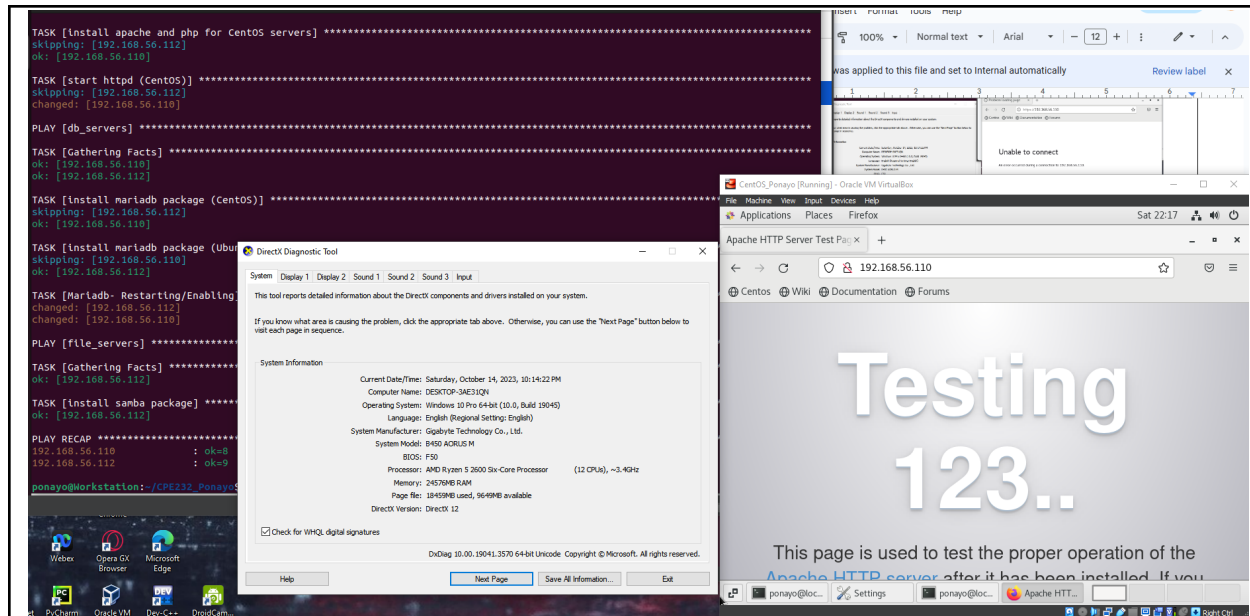
This is because in CentOS, installed packages' services are not run automatically. Thus, we need to create the module to run it automatically.

2. To test it, before you run the saved playbook, go to the CentOS server and stop the currently running httpd using the command ***sudo systemctl stop httpd***. When prompted, enter the sudo password. After that, open the browser and enter the CentOS server's IP address. You should not be getting a display because we stopped the httpd service already.



- Go to the local machine and this time, run the *site.yml* file. Then after running the file, go again to the CentOS server and enter its IP address on the browser. Describe the result.

To automatically enable the service every time we run the playbook, use the command *enabled: true* similar to Figure 7.1.2 and save the playbook.



## Reflections:

Answer the following:

1. What is the importance of putting our remote servers into groups?
  - By putting our remote servers into groups, we can improve our efficiency where it allows us to perform administrative tasks to multiple servers all at once. And it was also too easy to troubleshoot since we can use monitoring tools to quickly identify the root cause of the problem and fix them.
2. What is the importance of tags in playbooks?
  - The importance of tags in playbooks is that it allows us to selectively run or skip the tasks based on their tags by calling the name of their tags. Just like on this activity, I was able to divide the tasks on the script to and show them separately.
3. Why do think some services need to be managed automatically in playbooks?
  - I think by managing automatically the services in the playbook can improve security by ensuring that services are always configured and patched correctly. There are servers like Web servers, database servers, application servers, etc. that manage automatically their servers in playbooks because it is consistent.

