Name: Andreu John L. Salvador	Date Performed: 17/10/2023
Course/Section: CPE31S5	Date Submitted: 17/10/2023
Instructor: Engr. Roman Richard	Semester and SY: 1st Sem 2023-2024

Activity 7: Managing Files and Creating Roles in Ansible

1. **Objectives:**

1.1 Manage files in remote servers

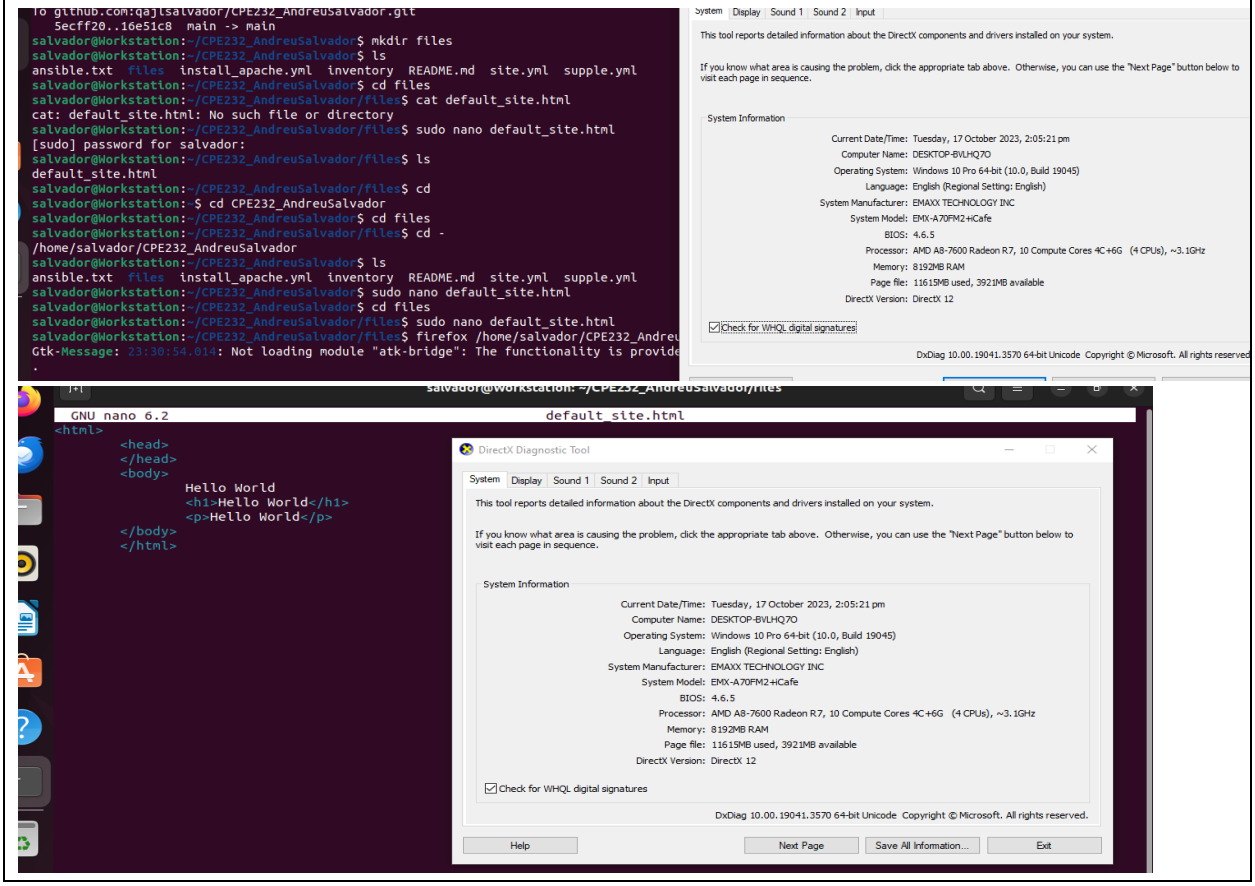
1.2 Implement roles in ansible

2. **Discussion**:

In this activity, we look at the concept of copying a file to a server. We are going to create a file into our git repository and use Ansible to grab that file and put it into a particular place so that we could do things like customize a default website, or maybe install a default configuration file. We will also implement roles to consolidate plays.

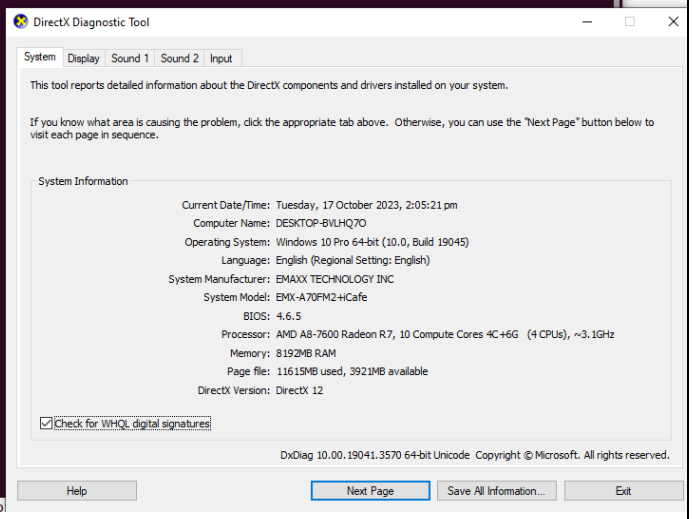
Task 1: Create a file and copy it to remote servers

1. Using the previous directory we created, create a directory, and named it "*files*." Create a file inside that directory and name it "*default_site.html*." Edit the file and put basic HTML syntax. Any content will do, as long as it will display text later. Save the file and exit.

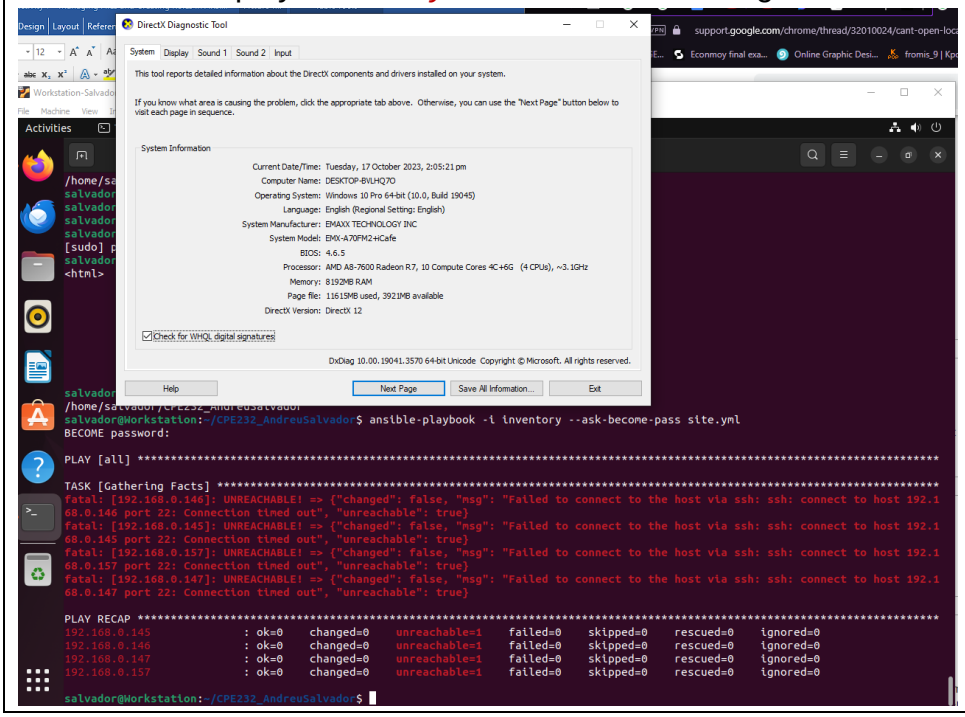


2. Edit the *site.yml* file and just below the *web_servers* play, create a new file to copy the default html file for site:
   - name: copy default html file for site
     tags: apache, apache2, httpd
     copy:
       src: default_site.html
       dest: /var/www/html/index.html
       owner: root
       group: root
       mode: 0644

```
    tags: always
    apt:
      upgrade: dist
      update_cache: yes
    when: ansible_distribution == "Ubuntu"

- hosts: web_servers
  become: true
  tasks:

  - name: copy default html file for site
    tags: apache, apache2, httpd
    copy:
      src: default_site.html
      dest: /var/www/html/index.html
      owner: root
      group: root
      mode: 0644

  - 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
```



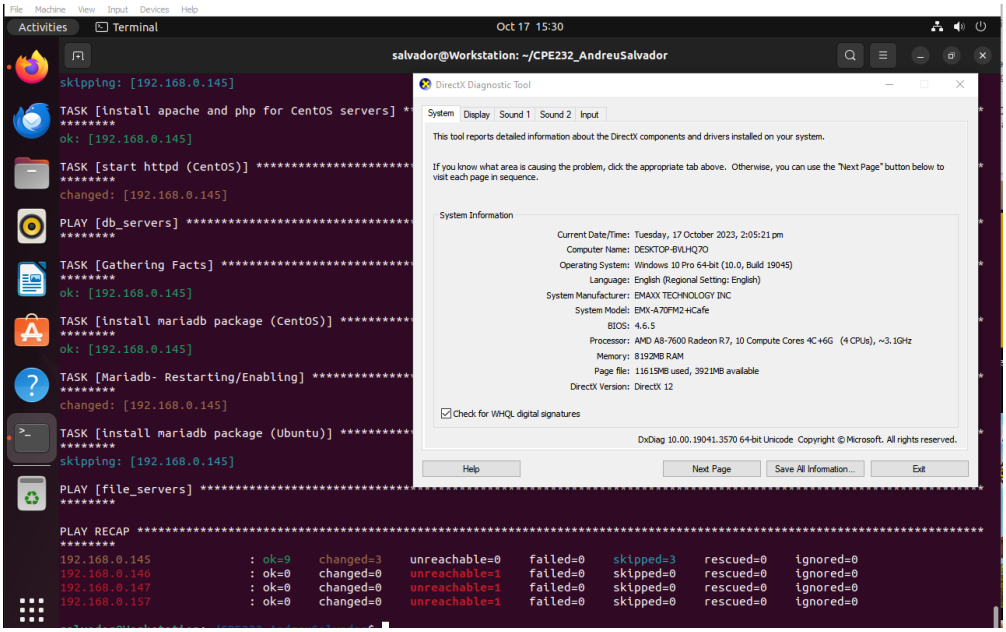
3. Run the playbook *site.yml*. Describe the changes.

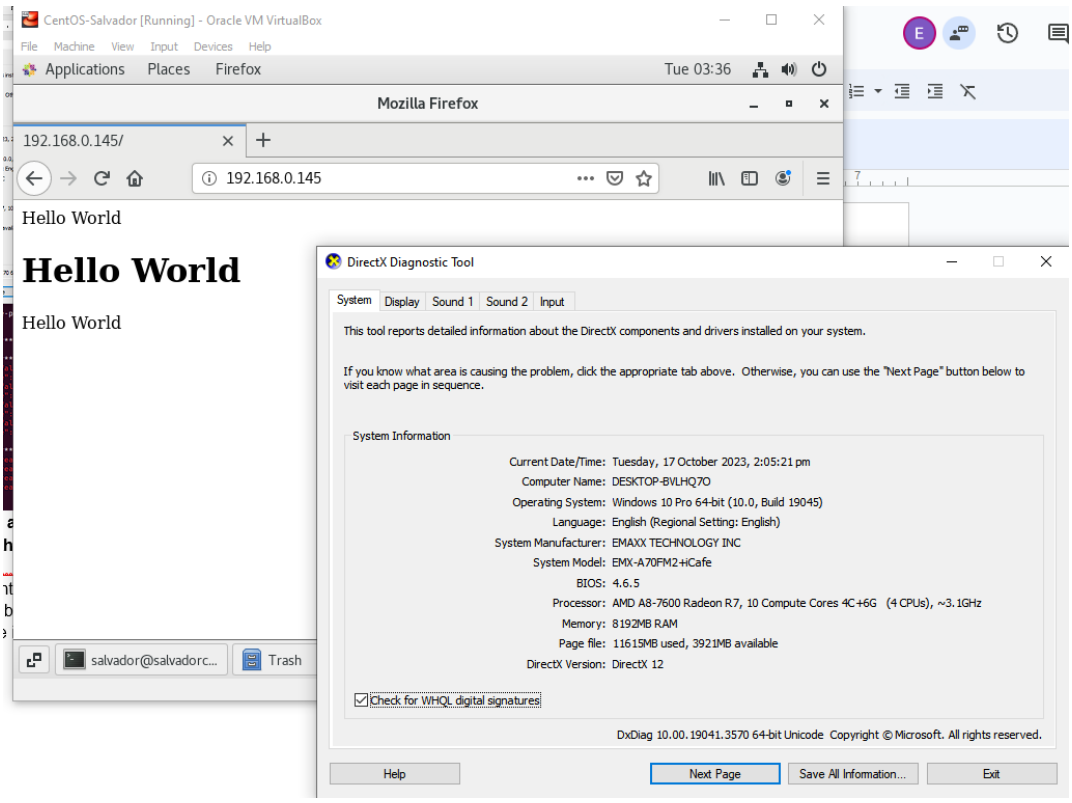


No changes occurred, since there are no server to reach due to the servers not being open prior into running the ansible playbook.

4. Go to the remote servers (*web_servers*) listed in your inventory. Use cat command to check if the index.html is the same as the local repository file (*default_site.html*). Do both for Ubuntu and CentOS servers. On the CentOS server, go to the browser and type its IP address. Describe the output.

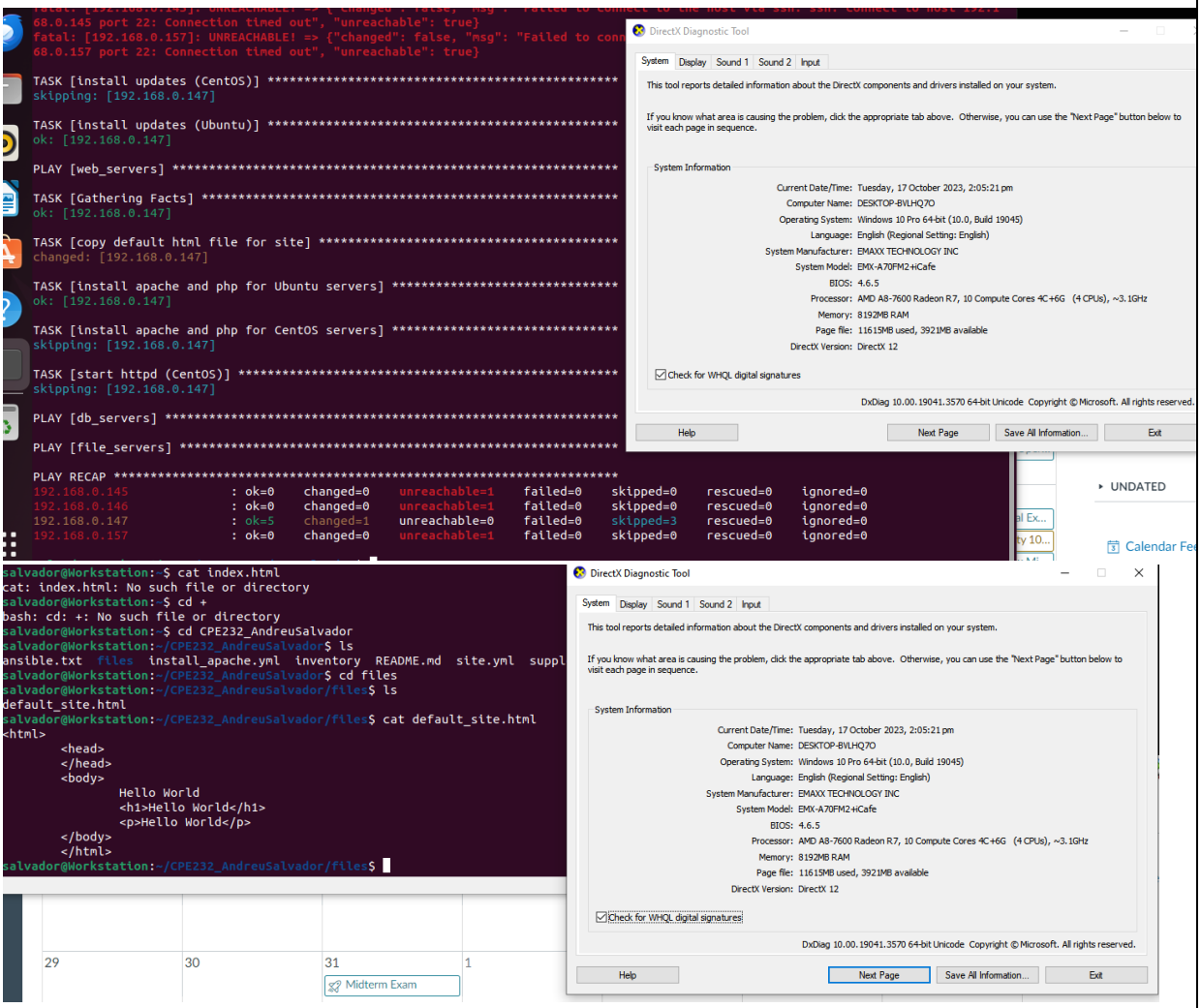
CentOS



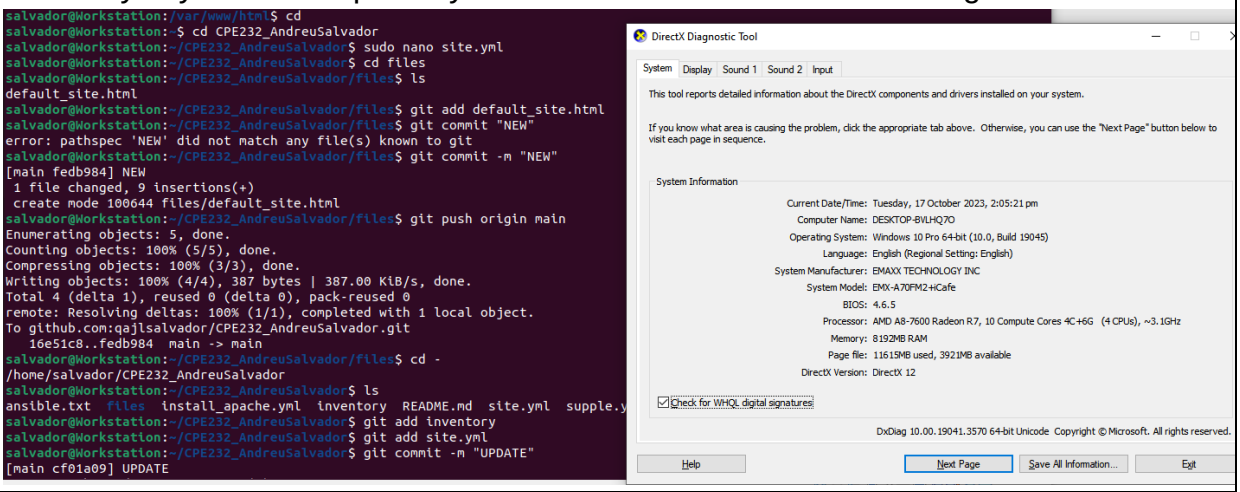


I wrote a basic html code wherein text would be printed in the body and header of the html. After the playbook initiate the tasks written inside, I accessed html file that was copied in the CentOS server via its ip address. In the screenshot above, it shows that what was written inside the default_site.html file was the same as what appeared when I accessed the site through the CentOS server.

Server2:192.168.0.147



5. Sync your local repository with GitHub and describe the changes.



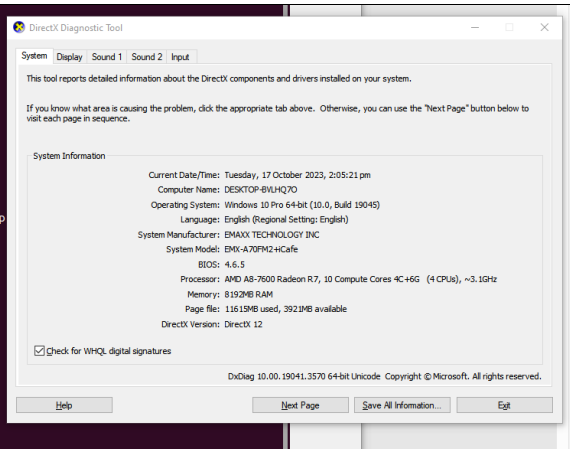
Task 2: Download a file and extract it to a remote server

1. Edit the site.yml. Just before the web_servers play, create a new play:
   - hosts: workstations
     become: true
     tasks:

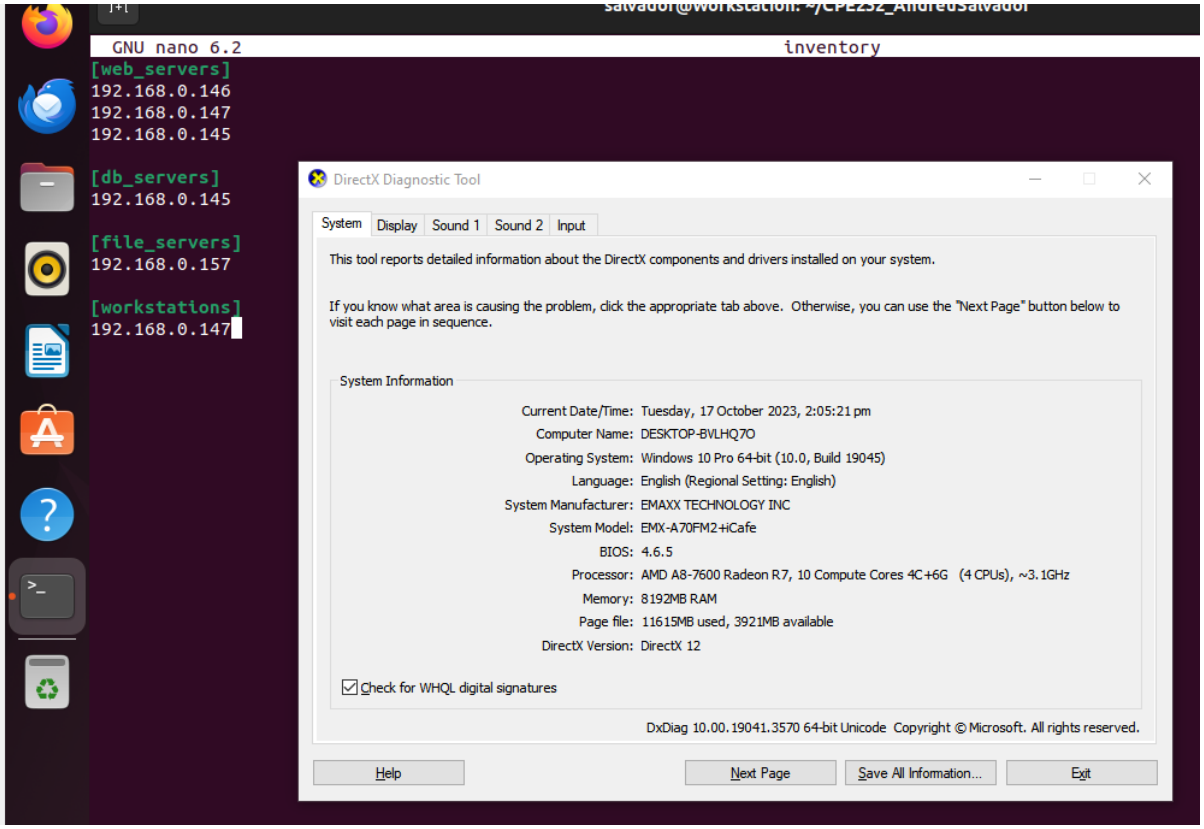
     - name: install unzip
       package:
         name: unzip_

     - name: install terraform
       unarchive:
         src: https://releases.hashicorp.com/terraform/0.12.28/terraform_0.12.28_linux_amd64.zip
         dest: /usr/local/bin
         remote_src: yes
         mode: 0755
         owner: root
         group: root

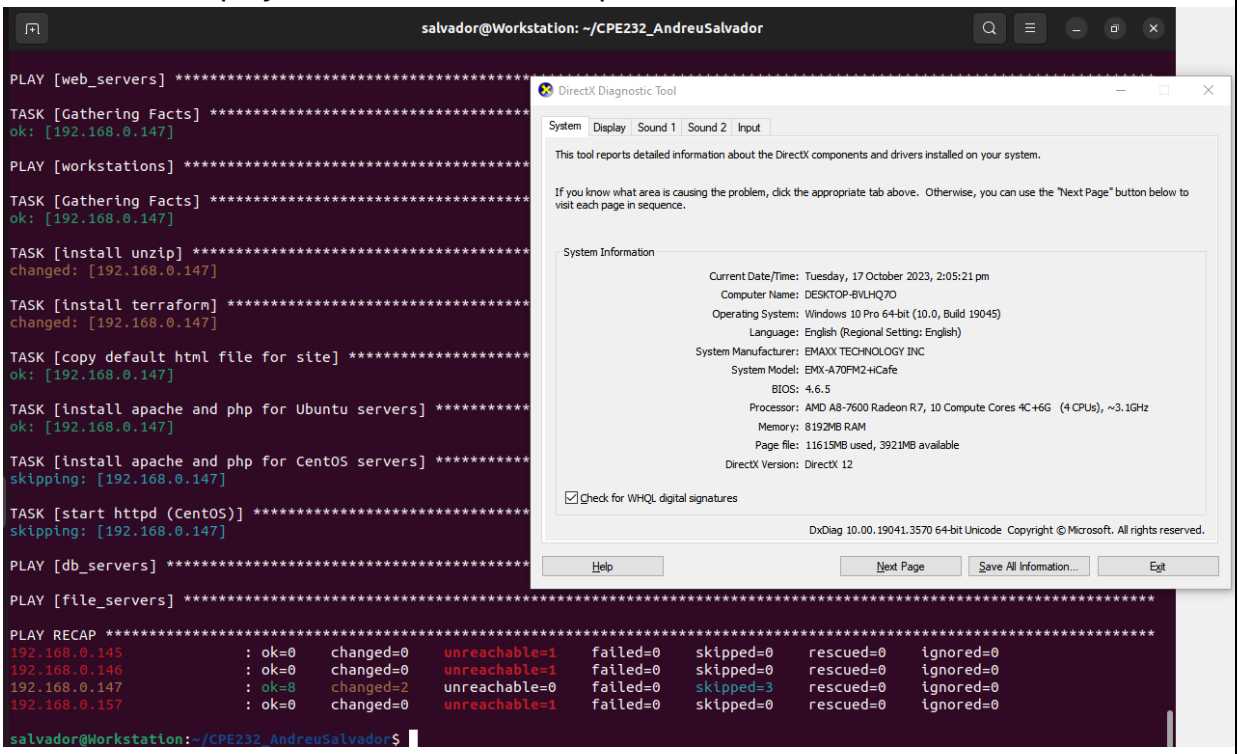
```
- hosts: web_servers
  become: true
  tasks:

- hosts: workstations
  become: true
  tasks:

  - name: install unzip
    package:
      name: unzip

  - name: install terraform
    unarchive:
      src: https://releases.hashicorp.com/terraform/0.12.28/terraform_0.12.28_linux_amd64.zip
      dest: /usr/local/bin
      remote_src: yes
      mode: 0755
      owner: root
      group: root

  - name: copy default html file for site
    tags: apache, apache2, httpd
    copy:
      src: default_site.html
      dest: /var/www/html/index.html
      owner: root
      group: root
      mode: 0644

  - name: install apache and php for Ubuntu servers
```



2. Edit the inventory file and add workstations group. Add any Ubuntu remote server. Make sure to remember the IP address.

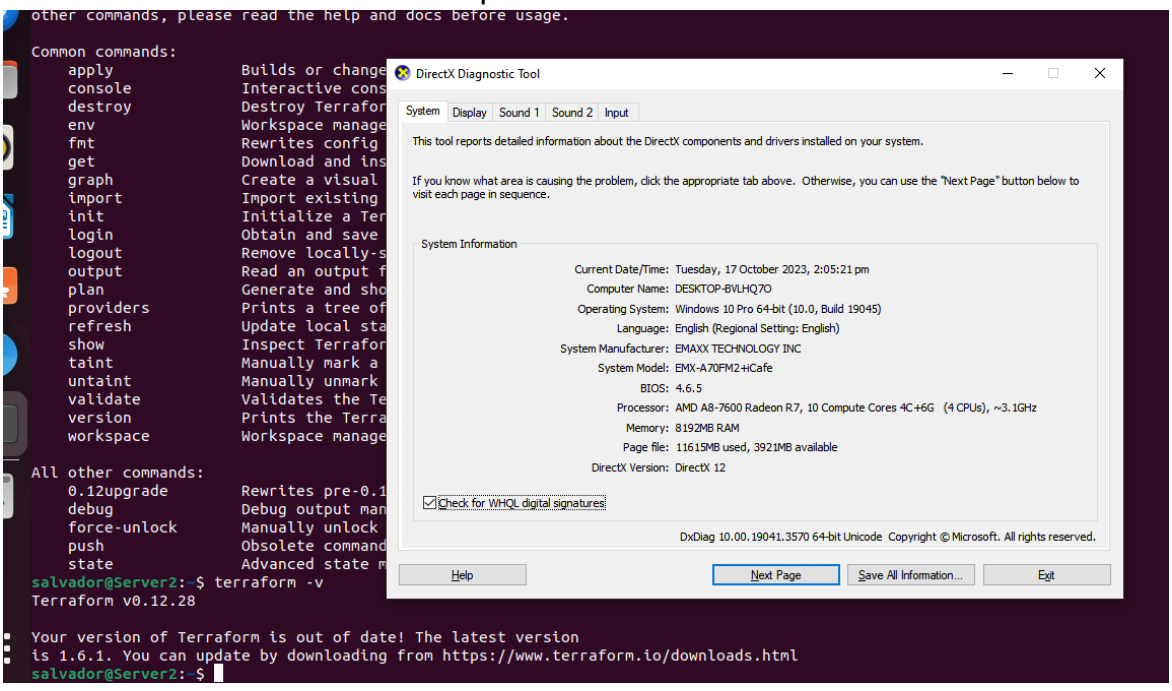


3. Run the playbook. Describe the output.



The unzipping or unarchiving of the zip file was successful as the report suggests that there are 2 changes that occurred after playbook was played.

4. On the Ubuntu remote workstation, type terraform to verify installation of terraform. Describe the output.



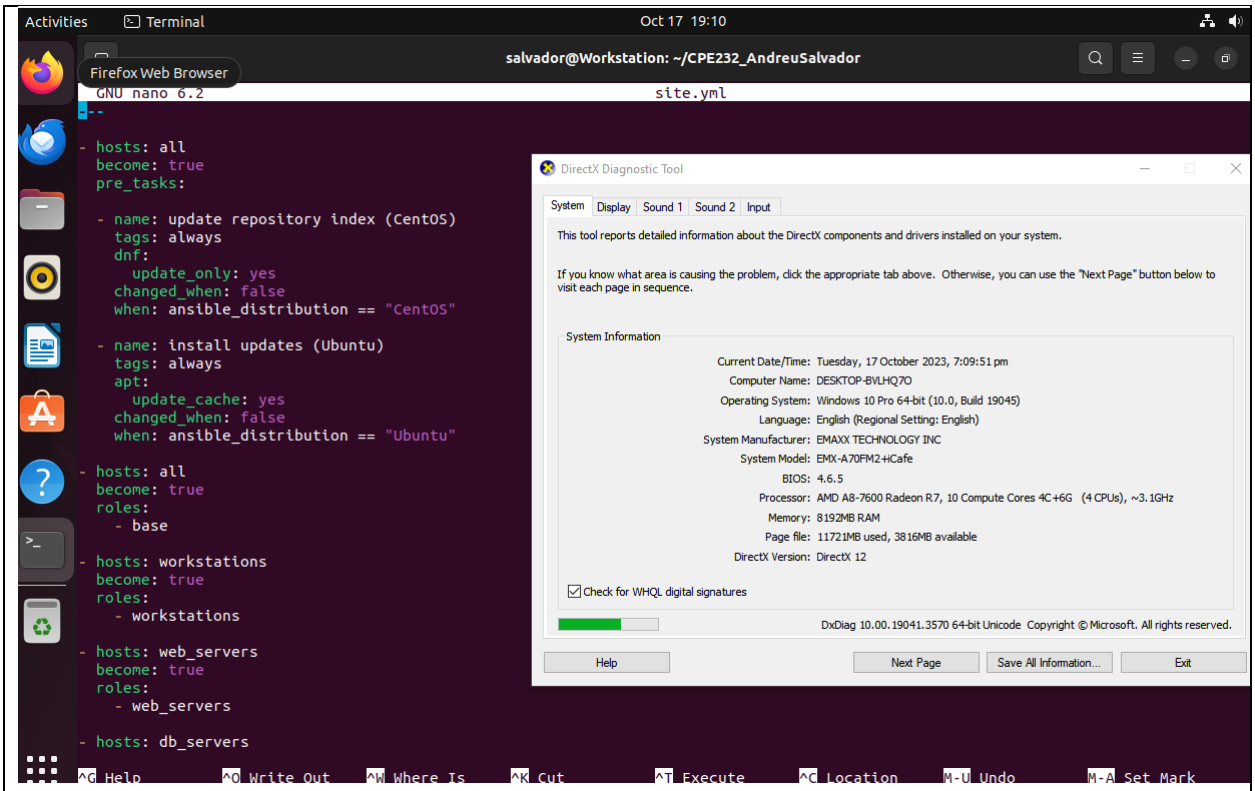
Terraform was installed but needed an update as it says that the terraform installed was out of date.

Task 3: Create roles

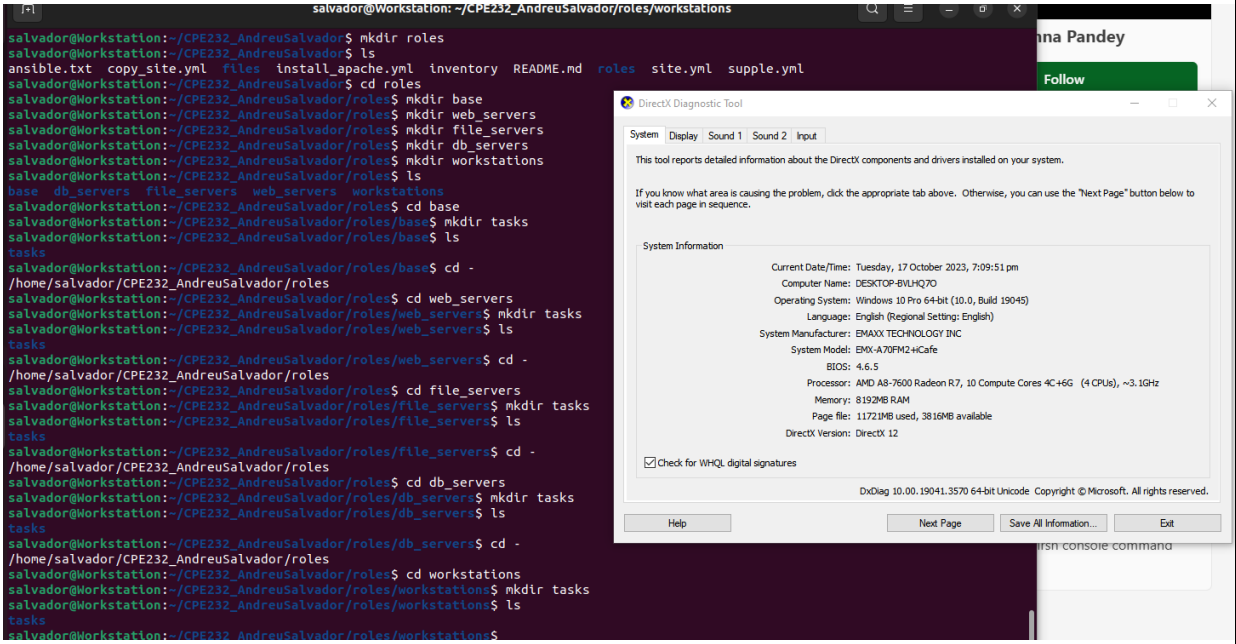
1. Edit the site.yml. Configure roles as follows: (make sure to create a copy of the old site.yml file because you will be copying the specific plays for all groups)

```
---
- hosts: all
  become: true
  pre_tasks:

  - name: update repository index (CentOS)
    tags: always
    dnf:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "CentOS"
  - name: install updates (Ubuntu)
    tags: always
    apt:
      update_cache: yes
    changed_when: false
    when: ansible_distribution == "Ubuntu"

- hosts: all
  become: true
  roles:
    -  base

- hosts: workstations
  become: true
  roles:
    - workstations

- hosts: web_servers
  become: true
  roles:
    - web_servers

- hosts: db_servers
  become: true
  roles:
    - db_servers

- hosts: file_servers
  become: true
  roles:
    - file_servers
```

Save the file and exit.

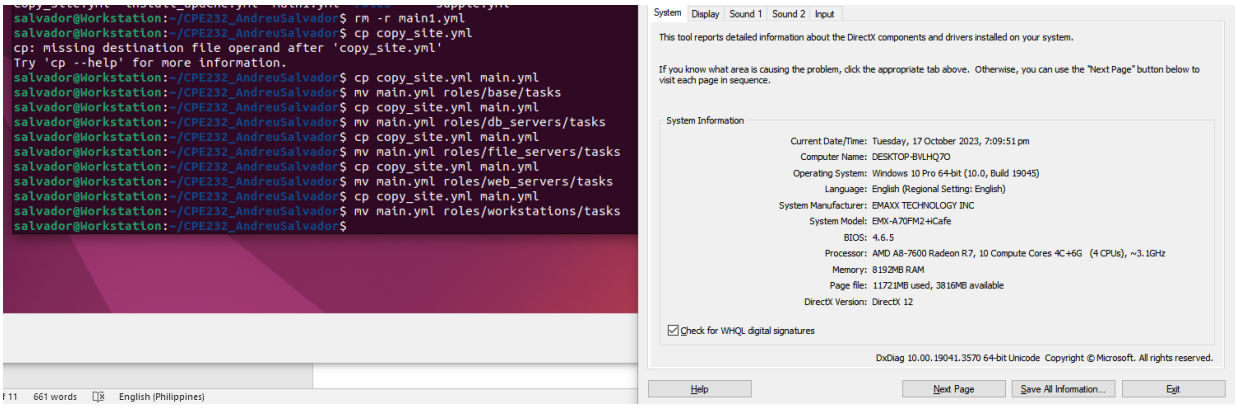


2. Under the same directory, create a new directory and name it roles. Enter the roles directory and create new directories: base, web_servers, file_servers, db_servers and workstations. For each directory, create a directory and name it tasks.

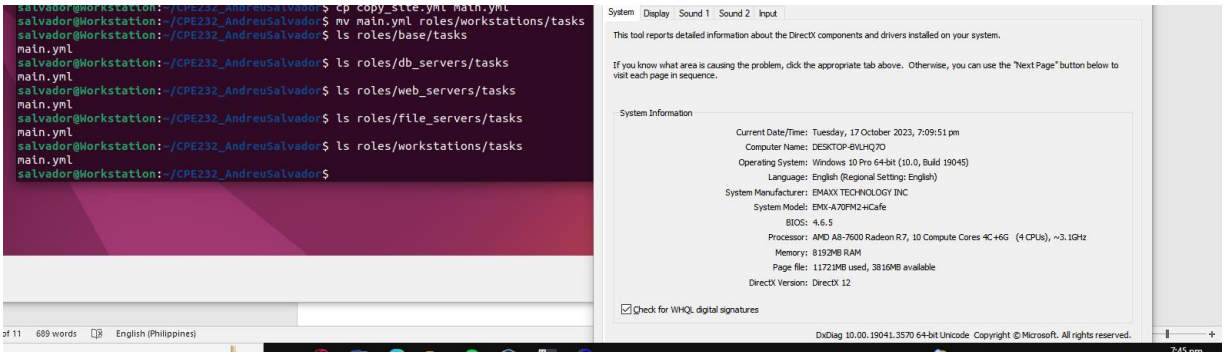


3. Go to tasks for all directory and create a file. Name it main.yml. In each of the tasks for all directories, copy and paste the code from the old site.yml file. Show all contents of main.yml files for all tasks.

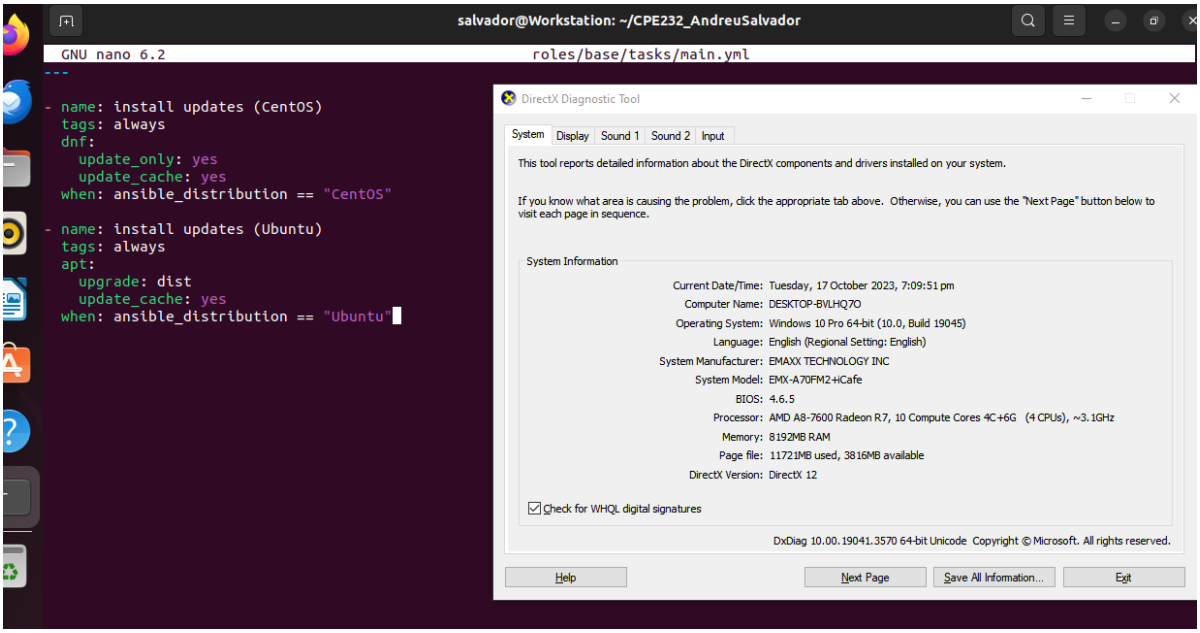
For less time to consume, I used the copy and move function of Ubuntu. I copy the site yml (the old one) and then move it to each directory



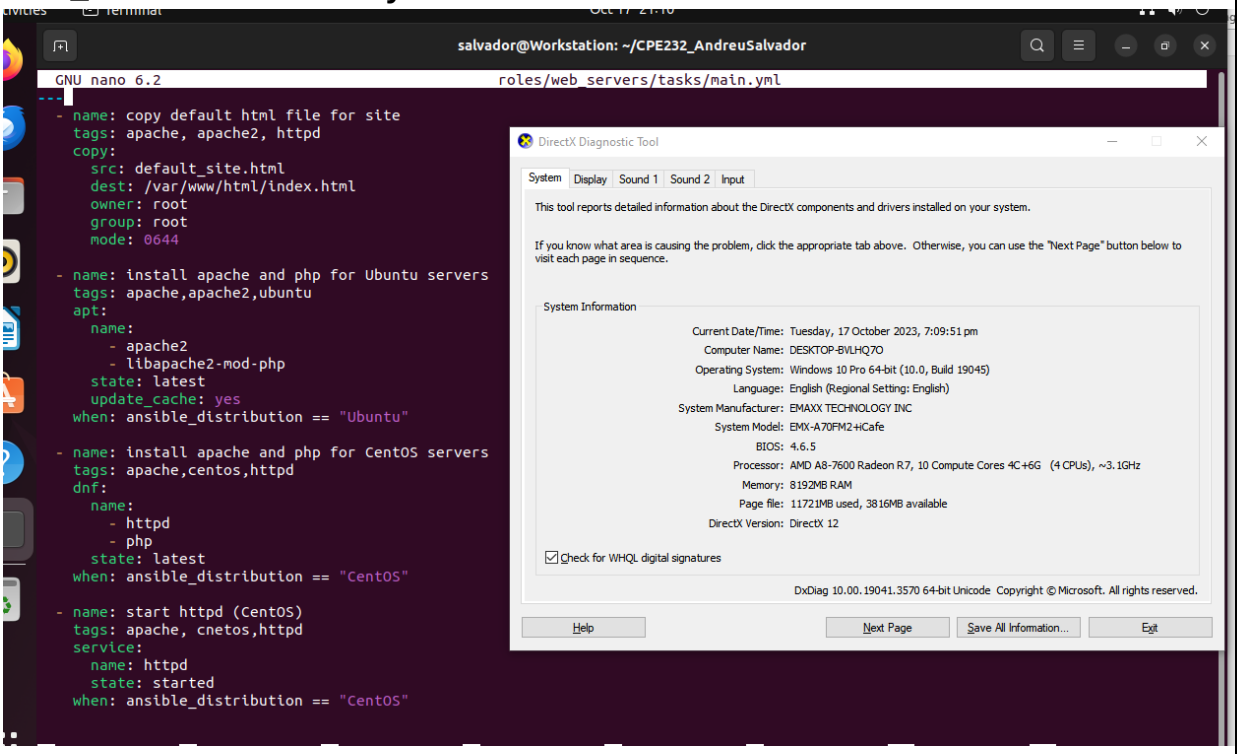
The directories now has the copy of the old site.yml as main.yml



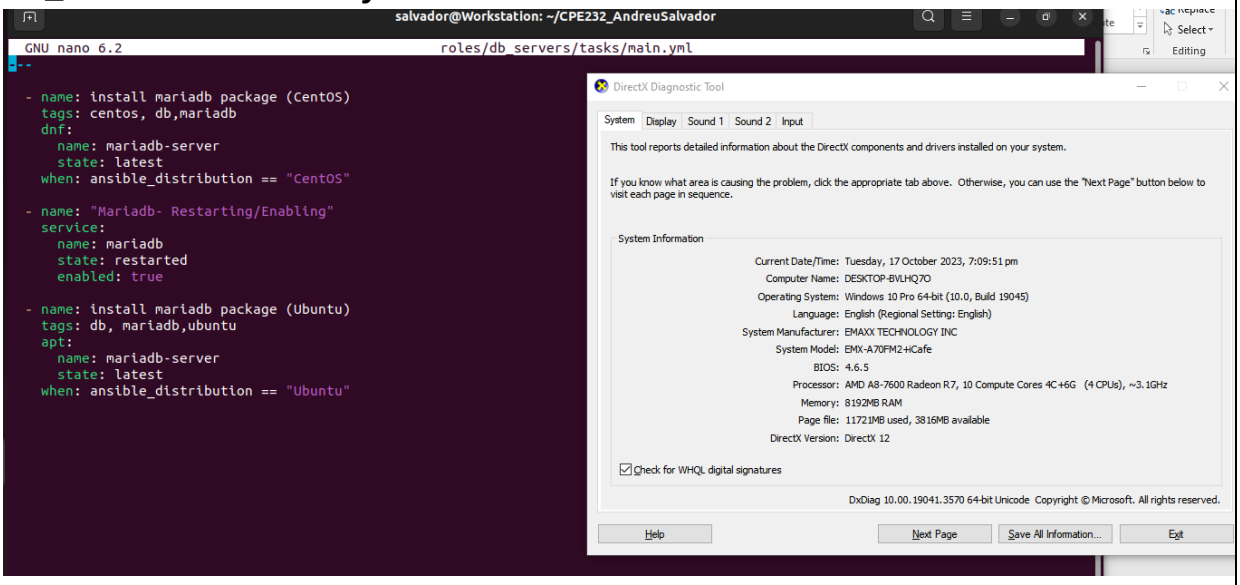
Base tasks main.yml file:



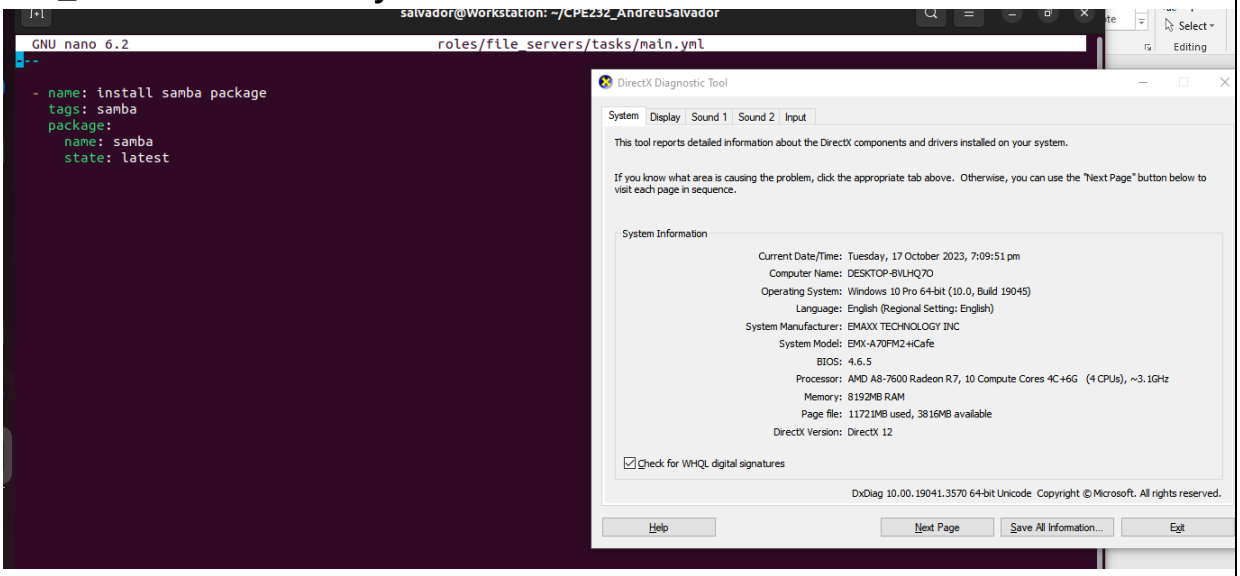
Web_servers tasks main.yml file:



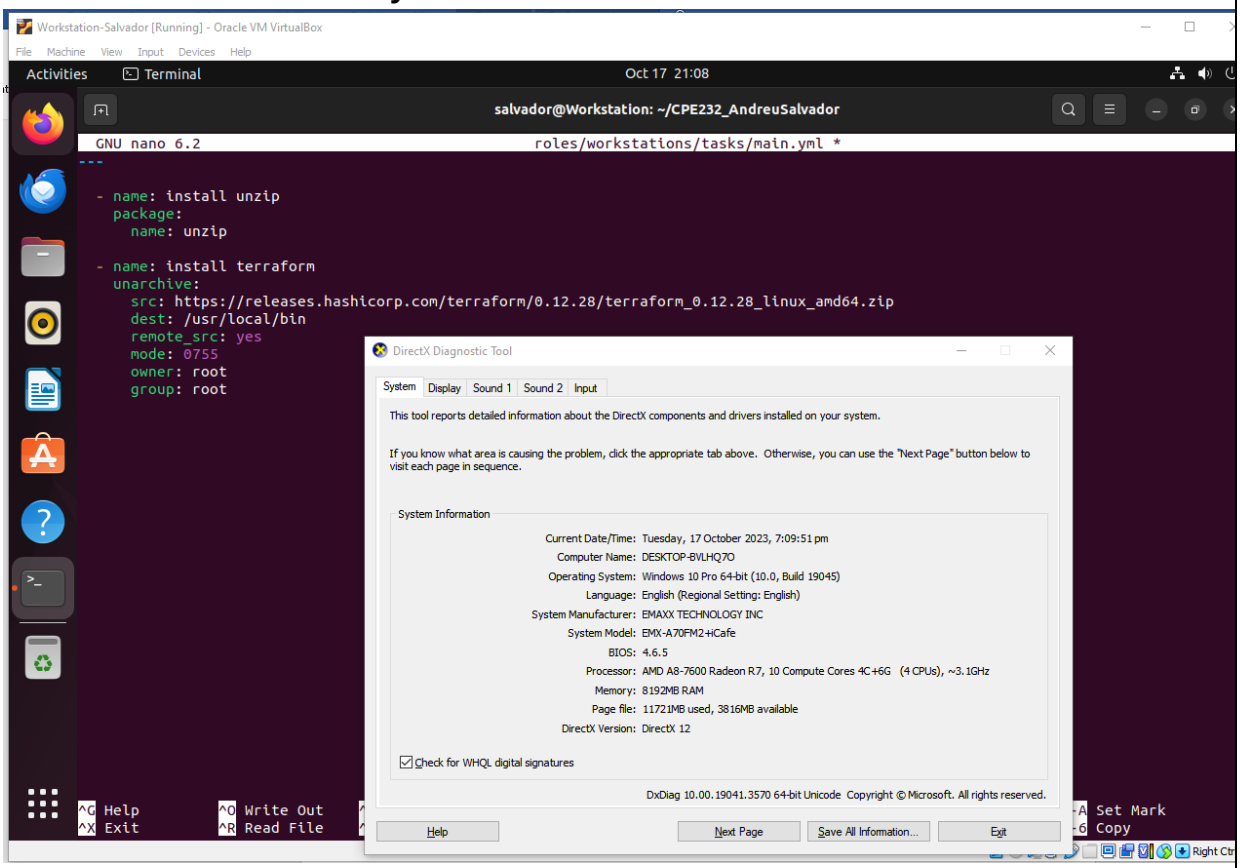
Db_servers tasks main.yml file:



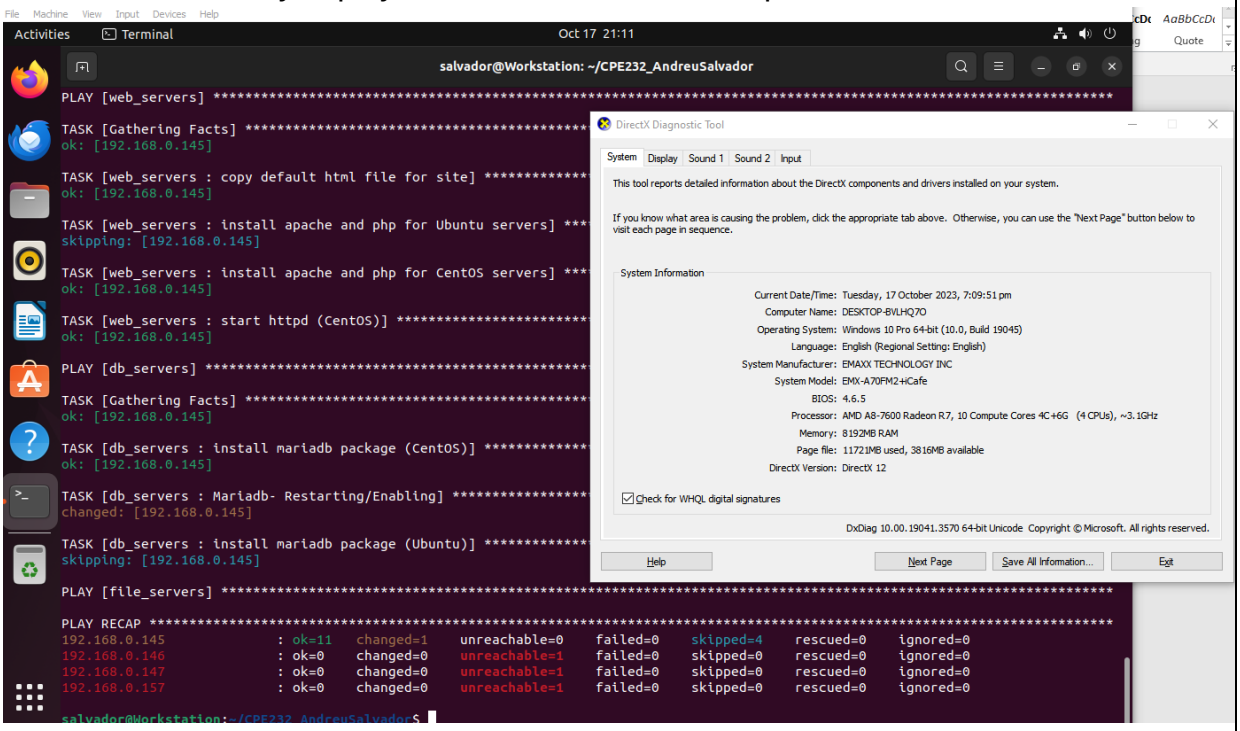
File_servers tasks main.yml file:



Workstations tasks main.yml file:



4. Run the site.yml playbook and describe the output.



The output was successful no failed occurred in running the playbook, changes occurred on the server. It ran every roles that was written on the site.yml file that's why there are a lot of occurrence that happened since I put the ip address of the server on different groups in the inventory.

Reflections:

Answer the following:

1. What is the importance of creating roles?
    - It can specify the tasks on groups that you want that tasks to be played on. It also lessen the line of codes for that specific group that you want those tasks for. Another benefit of creating roles is that you can easily debug an error since the tasks are group-specific. For a programmer/system admin a playbook that can be easily read and understood is an advantage that is really fatal when working on projects and server distribution.
2. What is the importance of managing files?
    - A chaotic workplace would always be hard to deal with. By managing our files using directories and grouping tasks together base on its roles is an advantage that we should be practicing. Our files is very important since it is where the information is stored to, so managing it so that we can easily access it from time to time and lessen the confusion is the most important thing that could happen.