| | |
|---|---|
| **Name:** John Renzo L. Mendoza | **Date Performed:** November 16, 2023 |
| **Course/Section:** CPE31S5 | **Date Submitted:** November 16, 2023 |
| **Instructor:** Engr. Roman Richard | **Semester and SY:** 1st Semester 2023-2024 |

**Activity 11:** Containerization

## 1. Objectives

Create a Dockerfile and form a workflow using Ansible as Infrastructure as Code (IaC) to enable Continuous Delivery process

## 2. Discussion

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

Source: https://docs.docker.com/get-started/overview/

You may also check the difference between containers and virtual machines. Click the link given below.

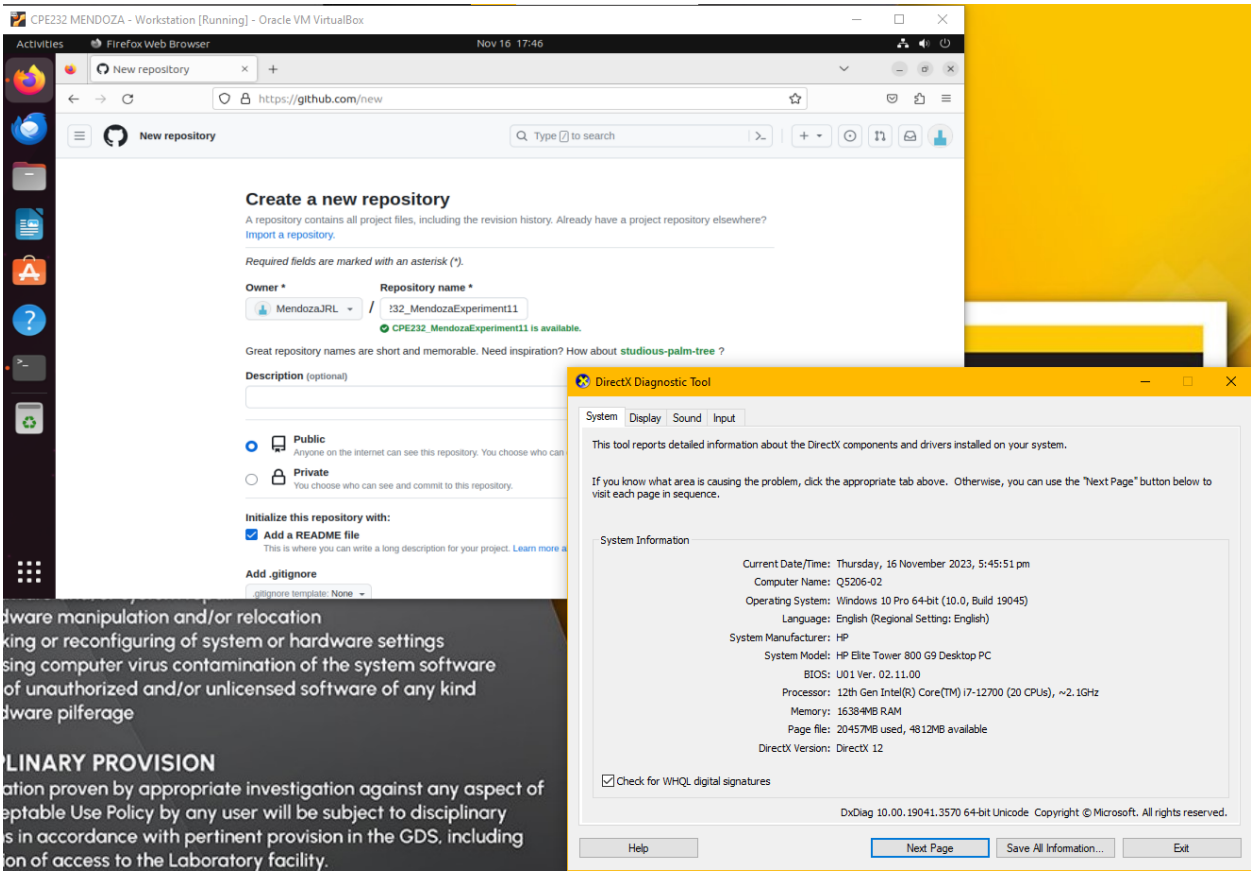
Source: https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/containers-vs-vm

## 3. Tasks

1. Create a new repository for this activity.
2. Install Docker and enable the docker socket.
3. Add a Docker group to your current user.
4. Create a Dockerfile to install web and DB servers.
5. Install and build the Dockerfile using Ansible.
6. Add, commit and push it to your repository.

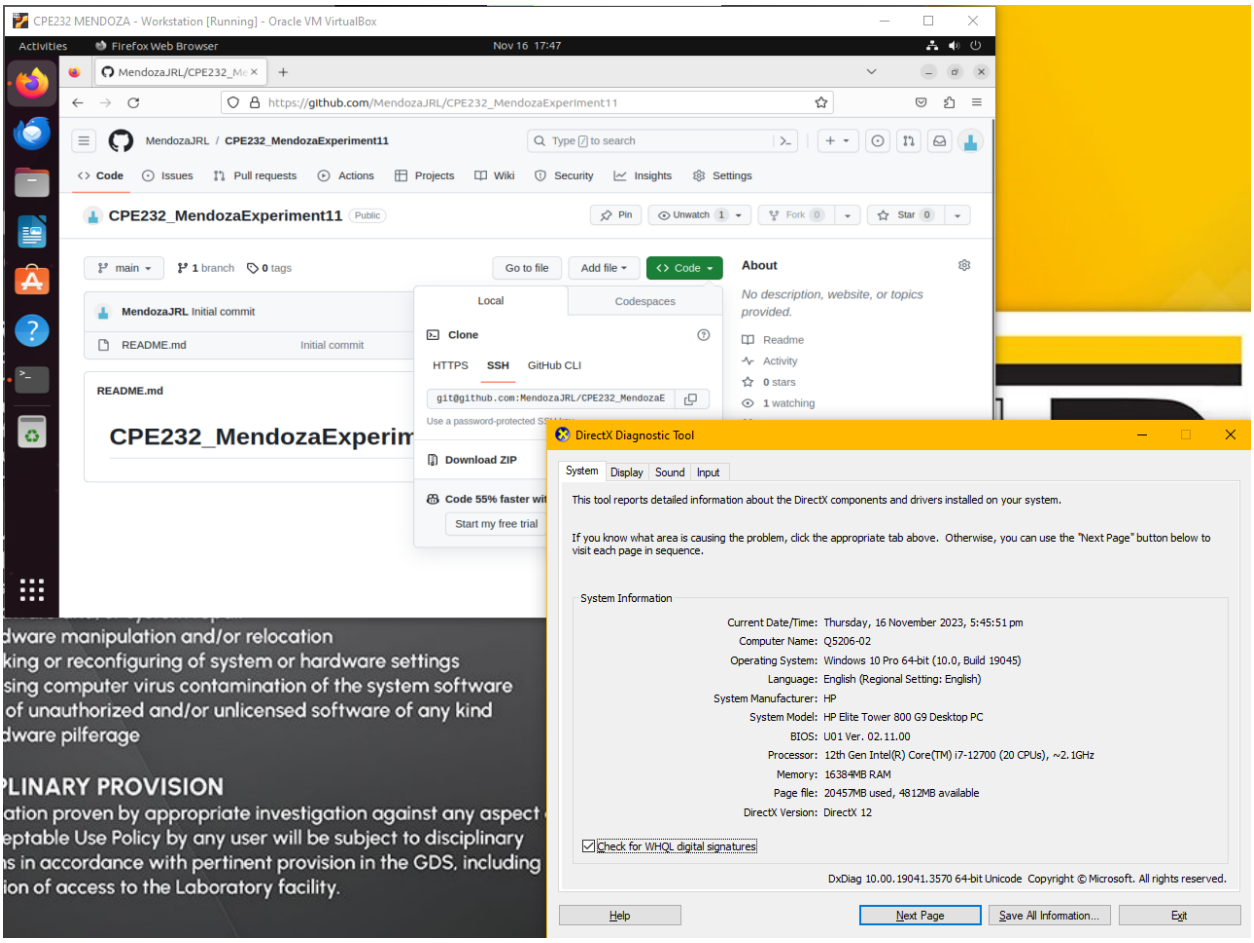
**4. Output** (screenshots and explanations)

1. Create a new repository for this activity.



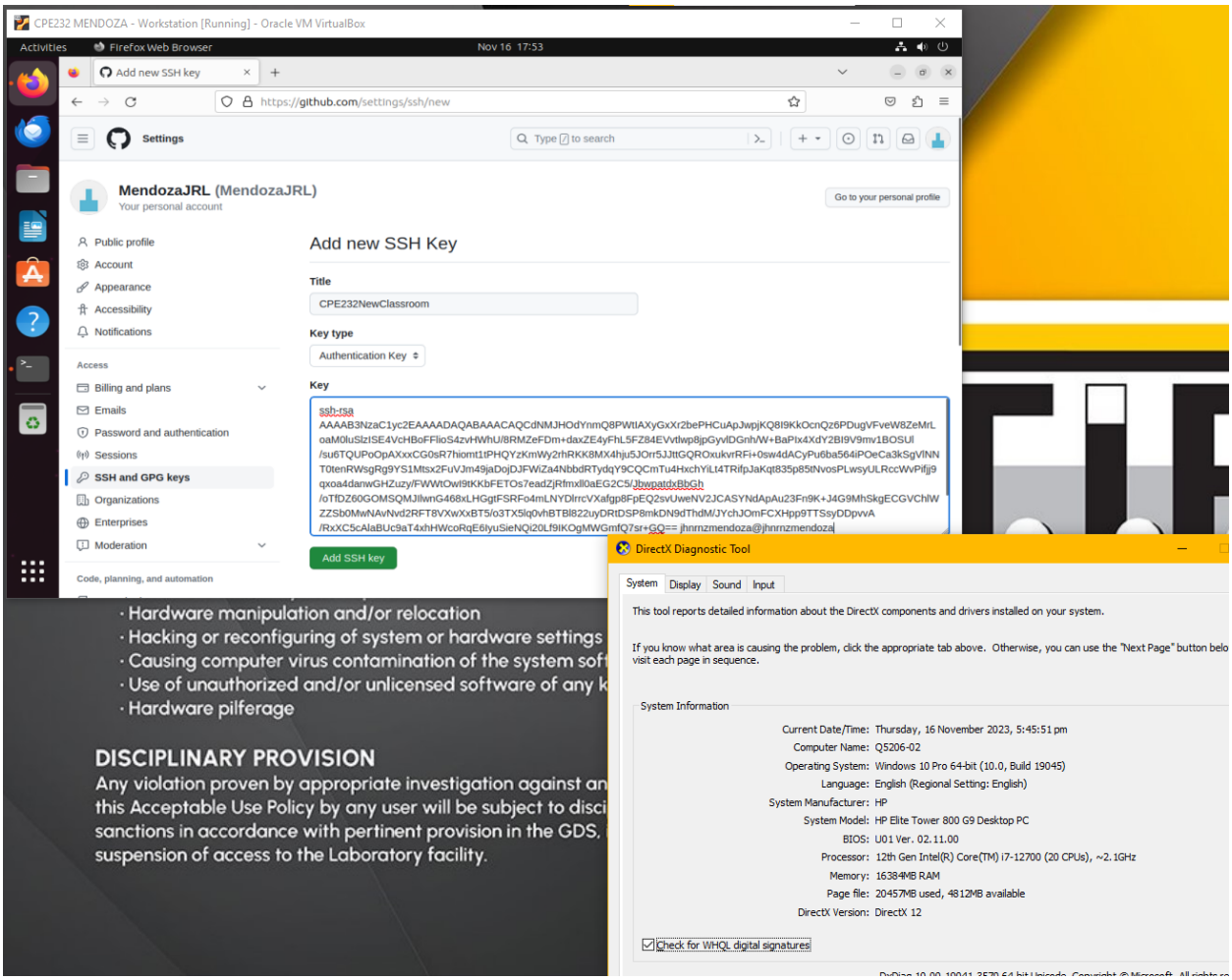

Observation:

- Using the GitHub Website, I have created a new repository for this activity.



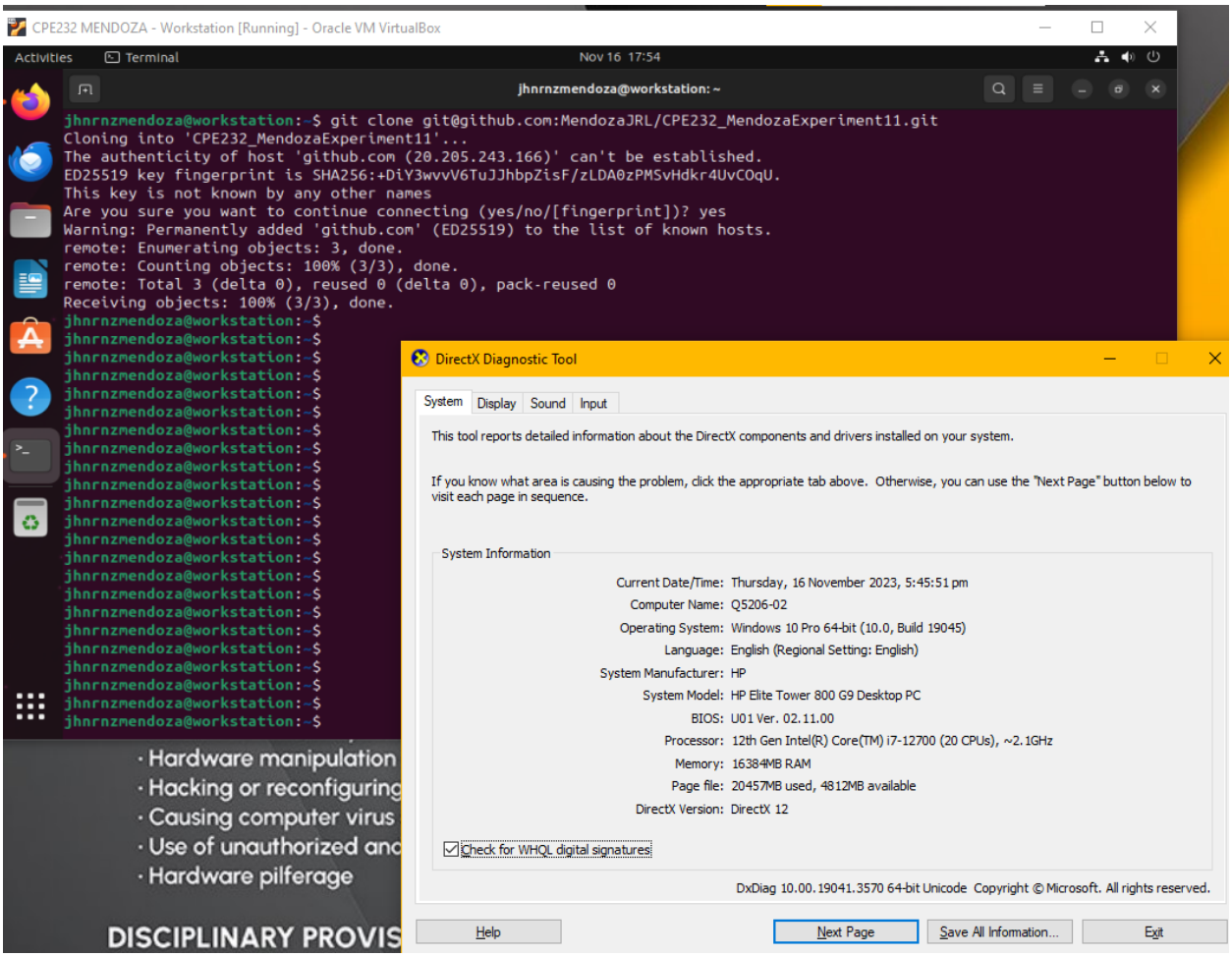

Observation:

- I have copied the SSH link of the created repository which will be used when cloning the repository on the local machine.




Observation:

- Since this activity was done on the new computer laboratory, I have added another SSH key to my GitHub account for the new computer on school.



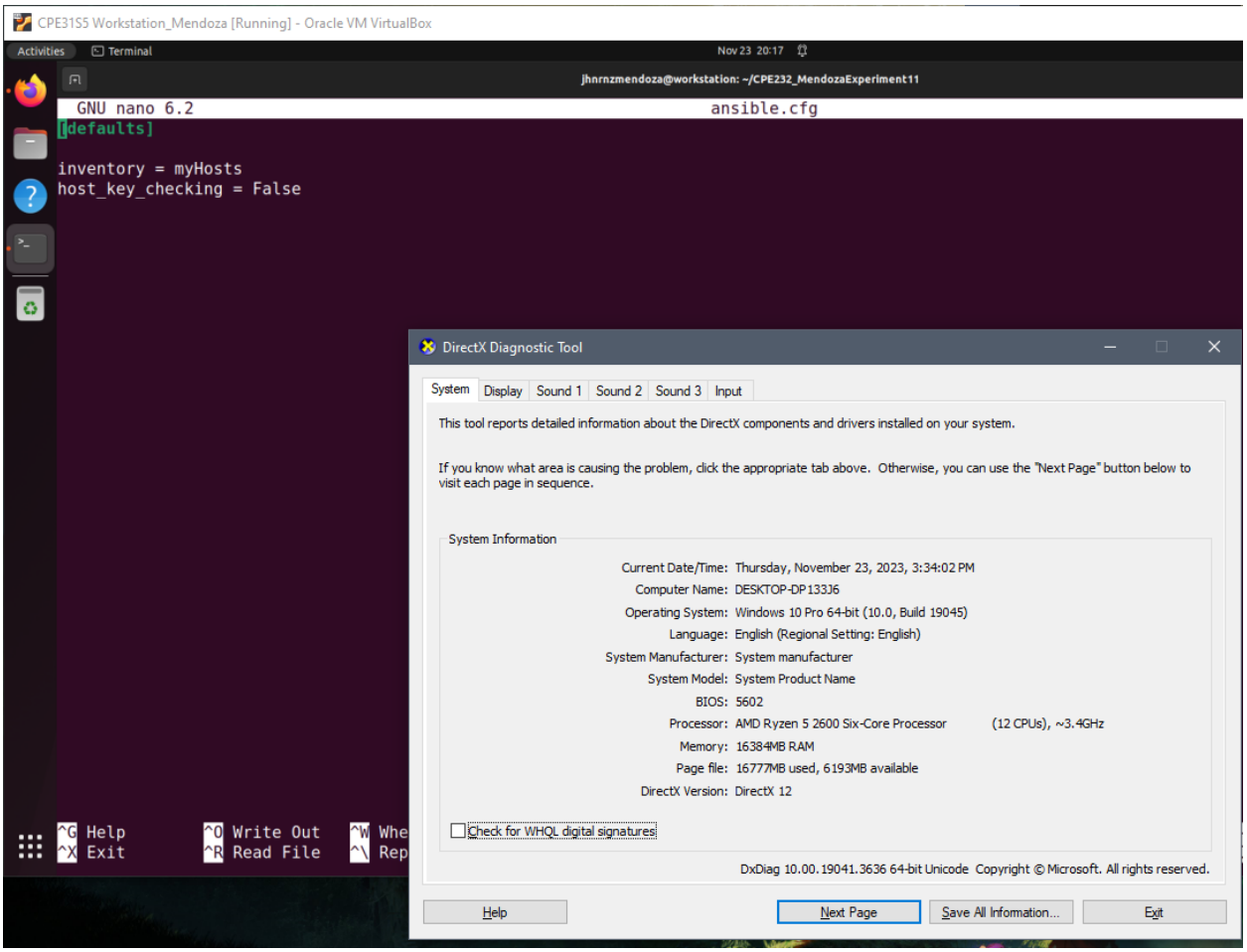

Observation:

- As observed, I have cloned the repository locally using the command “git clone”. This will enable me to push, pull, and create changes to the repository.

2. Install Docker and enable the docker socket.

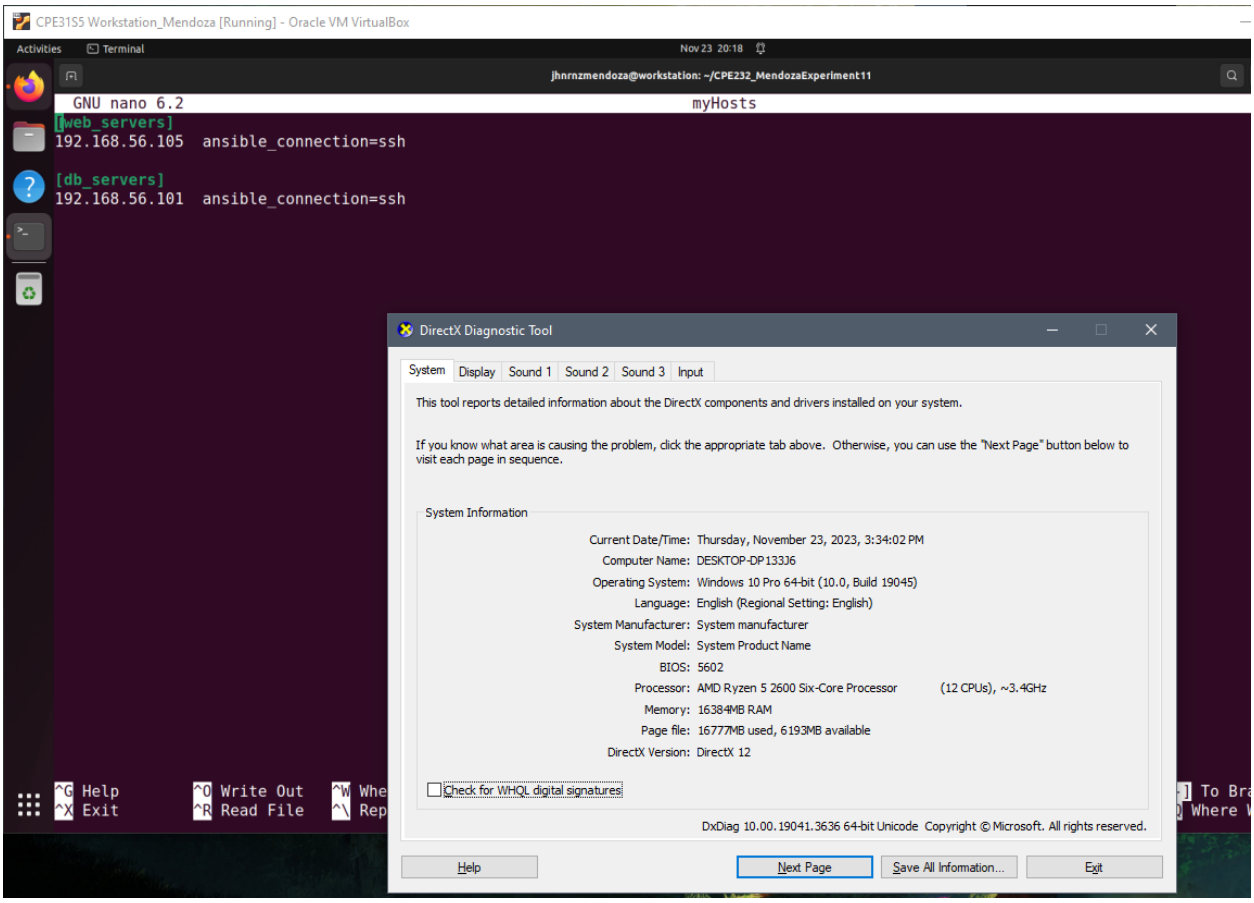
   In order to install Docker with the docker socket, I have referred to this website and translated it into an Ansible oriented installation:
   [How To Install and Use Docker on Ubuntu 22.04 | DigitalOcean]

**Pre-Configuration of Ansible Configuration, Ansible Inventory, and Ansible Hosts.**



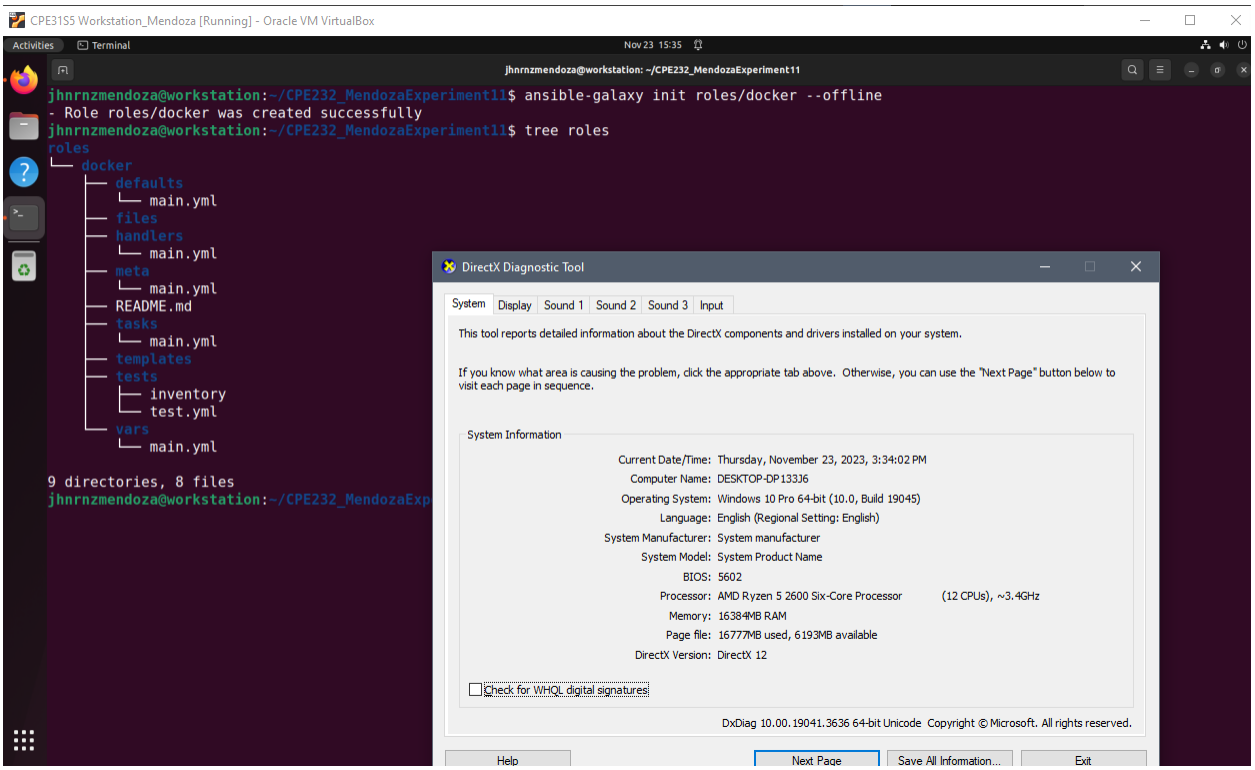

Observation:

- Initially, I have created a ansible.cfg file which stores the default settings of the repository including the inventory.



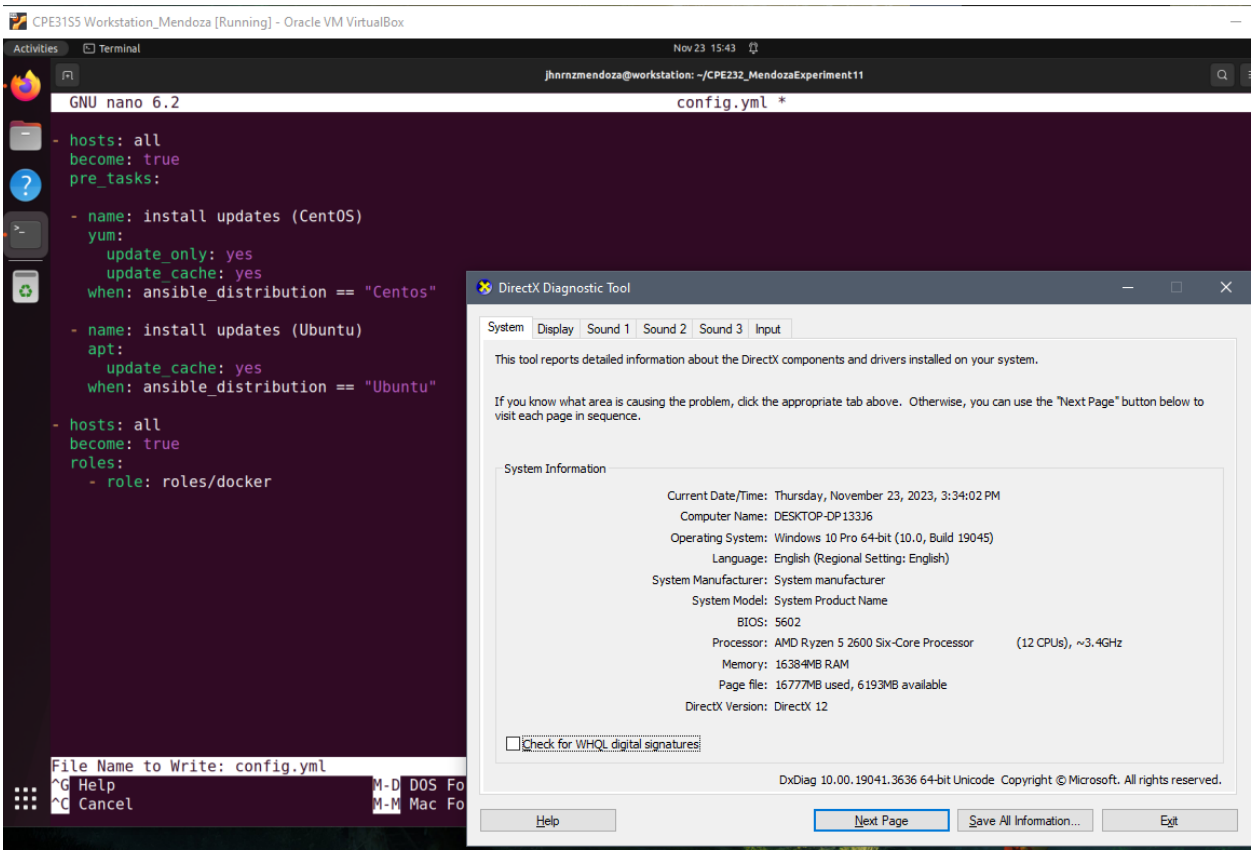

Observation:

- I have also created an inventory file entitled myHosts which stores the target managed nodes which will be managed by the control node. Specifically, one with Ubuntu distro and one for CentOS.



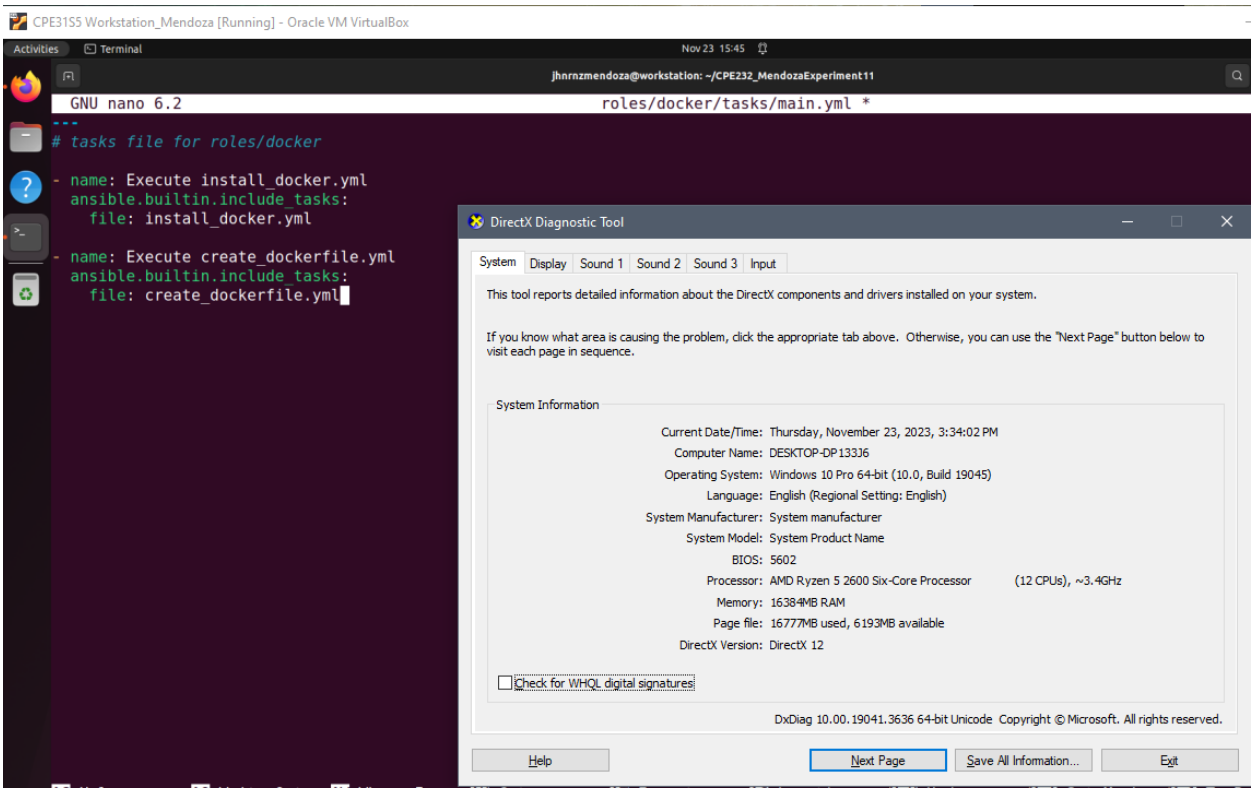

Observation:

- To include ansible roles, I have used the command ansible-galaxy to create a new role entitled docker. This new role will have the playbook tasks needed in performing this activity.




Observation:

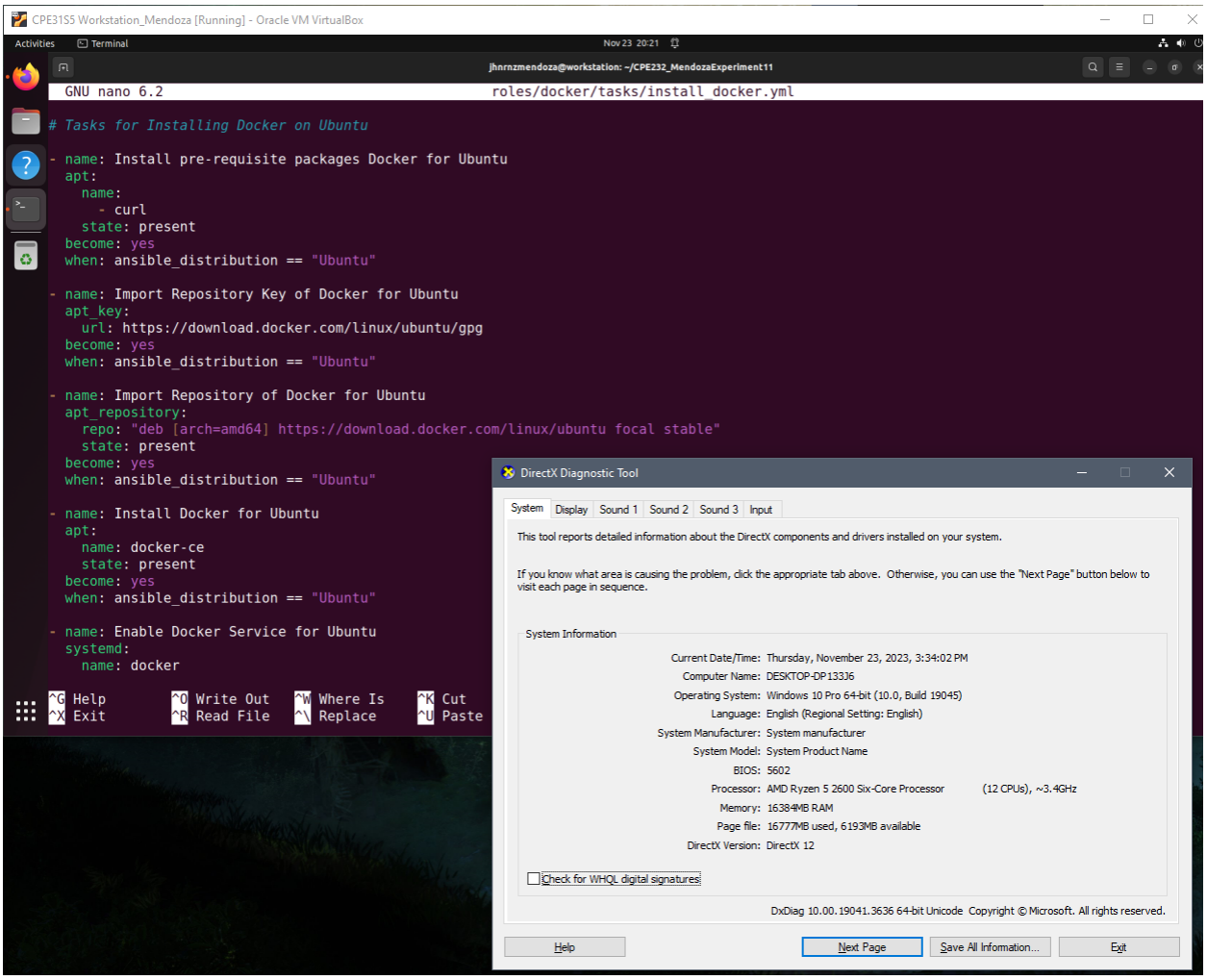
- In the repository, I have created a playbook config.yml which will perform tasks on the managed nodes. It will initially perform pre-tasks such as updating and upgrading the managed nodes to prepare them for modification later.




Observation:

- Under the roles/docker/tasks/ directory, I have modified the main.yml file to import the tasks from the other two yaml files to be created later. This will make the playbook run more organized and easier to identify in terms of its purpose.

Part of the Tasks which is for Installing Docker and Enabling it:




Observation:

- In this playbook file install_docker.yml, the tasks for Ubuntu and CentOS is similar in structure:
- First, it will install the prerequisite packages, which is Curl.
- Second, it will import the GPG key of the docker repository so that the repository can be accessed by the remote server.
- Third, it will import the docker repository and open it using the GPG key from the previous task.
- Fourth, the remote servers will now install docker.
- Fifth, the remote servers will enable the docker service, so that it will be running on their systems.

CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox

Activities Terminal Nov 23 20:21

jhnrnzmendoza@workstation: ~/CPE232_MendozaExperiment11

```
GNU nano 6.2                    roles/docker/tasks/install_docker.yml
- name: Enable Docker Service for Ubuntu
  systemd:
    name: docker
    enabled: yes
    state: started
  become: yes
  when: ansible_distribution == "Ubuntu"

# Tasks for Installing Docker on CentOS

- name: Install pre-requisite packages of Docker for CentOS
  yum:
    name:
      - curl
    state: present
  become: yes
  when: ansible_distribution == "CentOS"

- name: Import Repository Key of Docker for CentOS
  shell: rpm --import https://get.docker.com/ | sh
  when: ansible_distribution == "CentOS"

- name: Install Docker for CentOS
  yum:
    name: docker
    state: present
  become: yes
  when: ansible_distribution == "CentOS"

- name: Enable Docker Service for CentOS
  systemd:
    name: docker
    enabled: yes
    state: started

^G Help      ^O Write Out   ^W Where Is   ^K Cut
^X Exit      ^R Read File   ^\ Replace    ^U Paste
```

DirectX Diagnostic Tool

System | Display | Sound 1 | Sound 2 | Sound 3 | Input

This tool reports detailed information about the DirectX components and drivers installed on your system.

If you know what area is causing the problem, click the appropriate tab above. Otherwise, you can use the "Next Page" button below to visit each page in sequence.

System Information

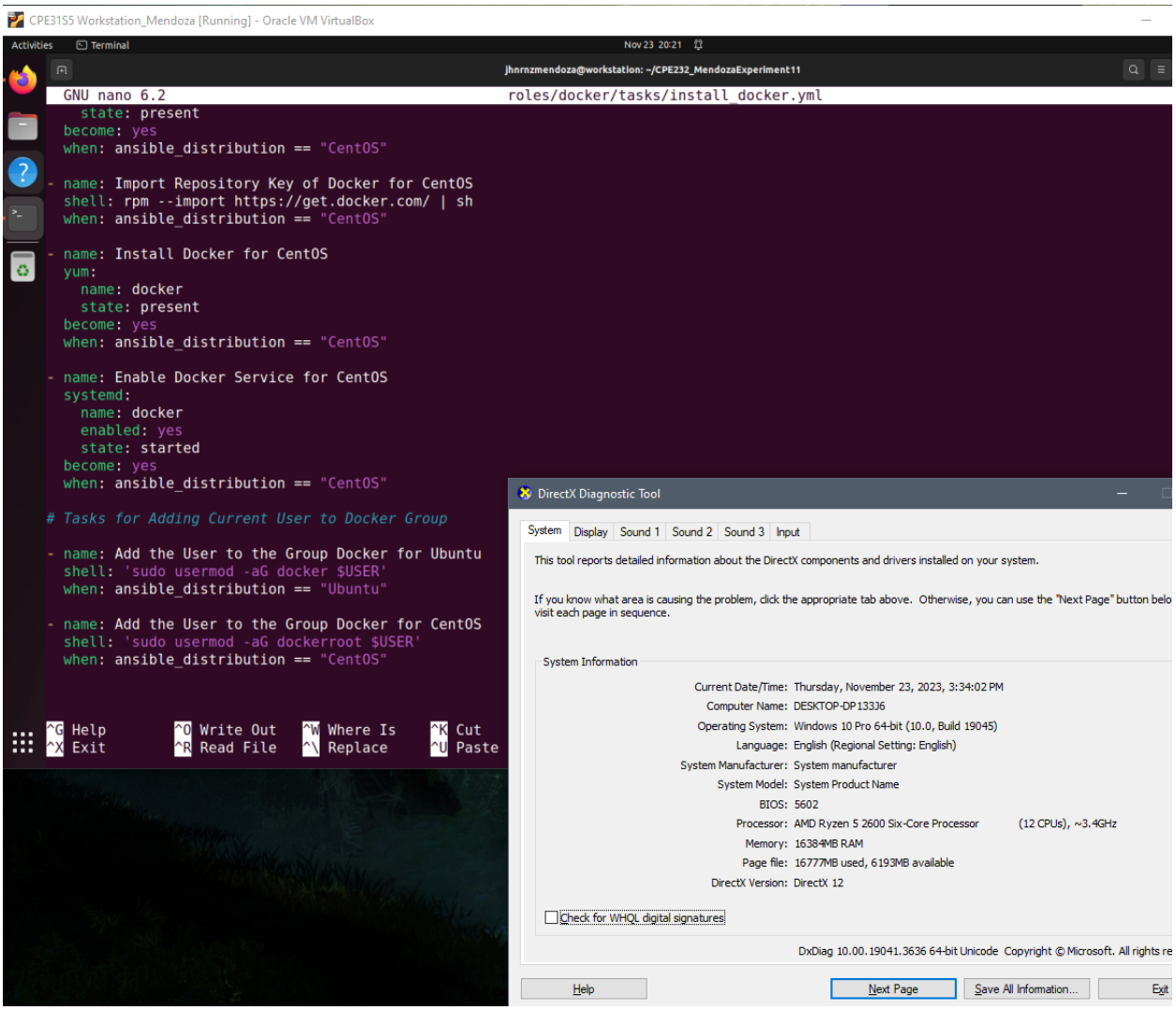
| | |
|---|---|
| Current Date/Time: | Thursday, November 23, 2023, 3:34:02 PM |
| Computer Name: | DESKTOP-DP13336 |
| Operating System: | Windows 10 Pro 64-bit (10.0, Build 19045) |
| Language: | English (Regional Setting: English) |
| System Manufacturer: | System manufacturer |
| System Model: | System Product Name |
| BIOS: | 5602 |
| Processor: | AMD Ryzen 5 2600 Six-Core Processor (12 CPUs), ~3.4GHz |
| Memory: | 16384MB RAM |
| Page file: | 16777MB used, 6193MB available |
| DirectX Version: | DirectX 12 |

☐ Check for WHQL digital signatures

DxDiag 10.00.19041.3636 64-bit Unicode Copyright © Microsoft. All rights reserved.

Help | Next Page | Save All Information... | Exit

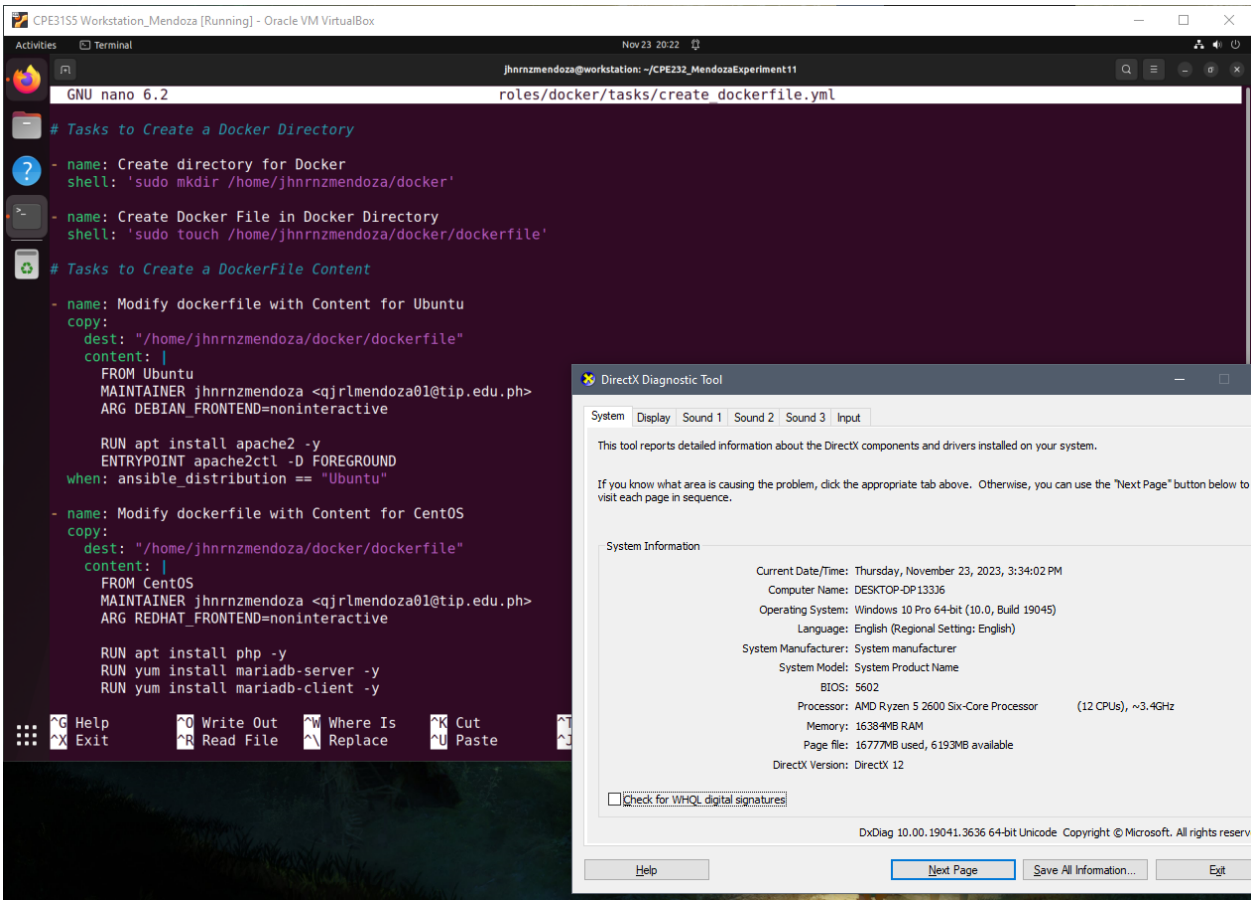
3. Add a Docker group to your current user.




Observation:

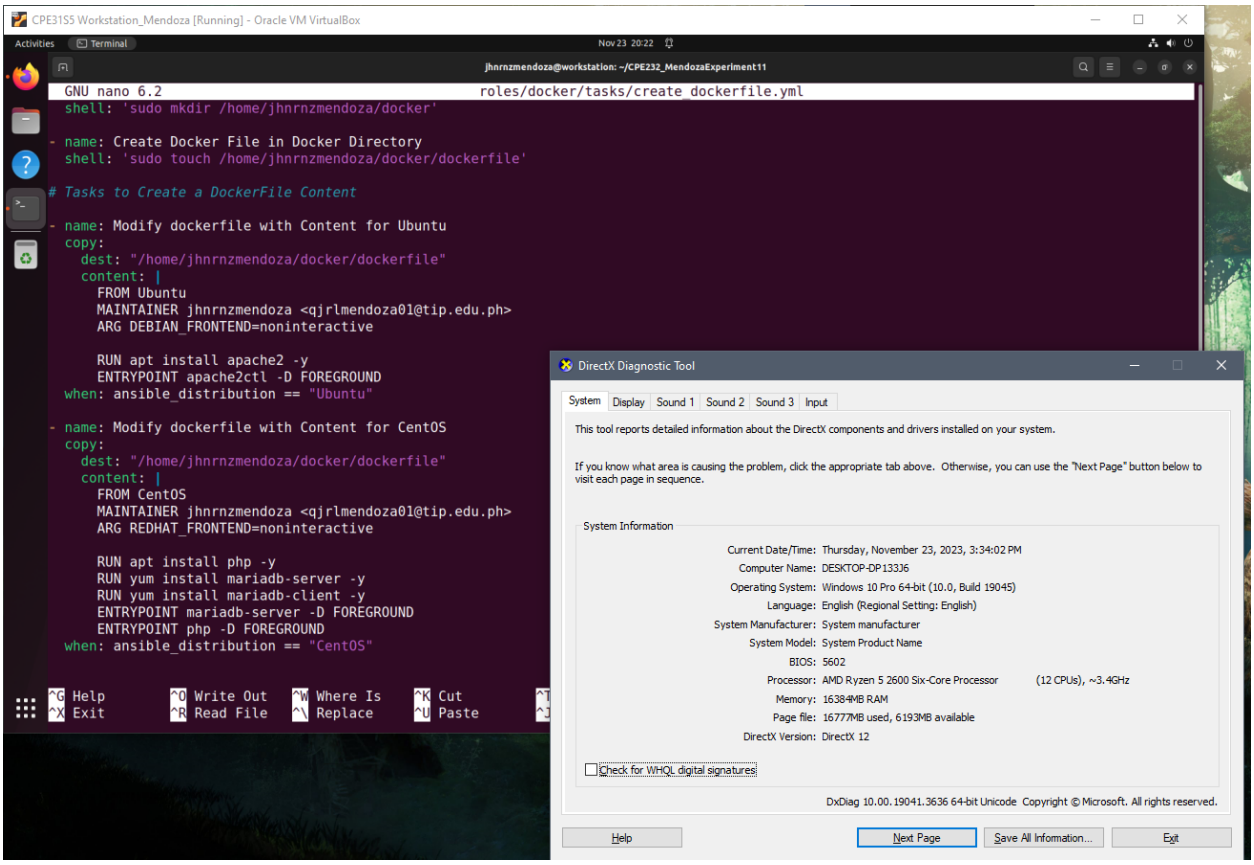
- To create a docker group and add the current user as part of the group, I have used the ansible playbook's shell syntax to perform terminal queries. The commands groupadd may be used to create a docker group, but it is created automatically after the installation, which is why it is neglected in this step. The command usermod can be used to assign the user as a member of the group.

4. Create a Dockerfile to install web and DB servers.



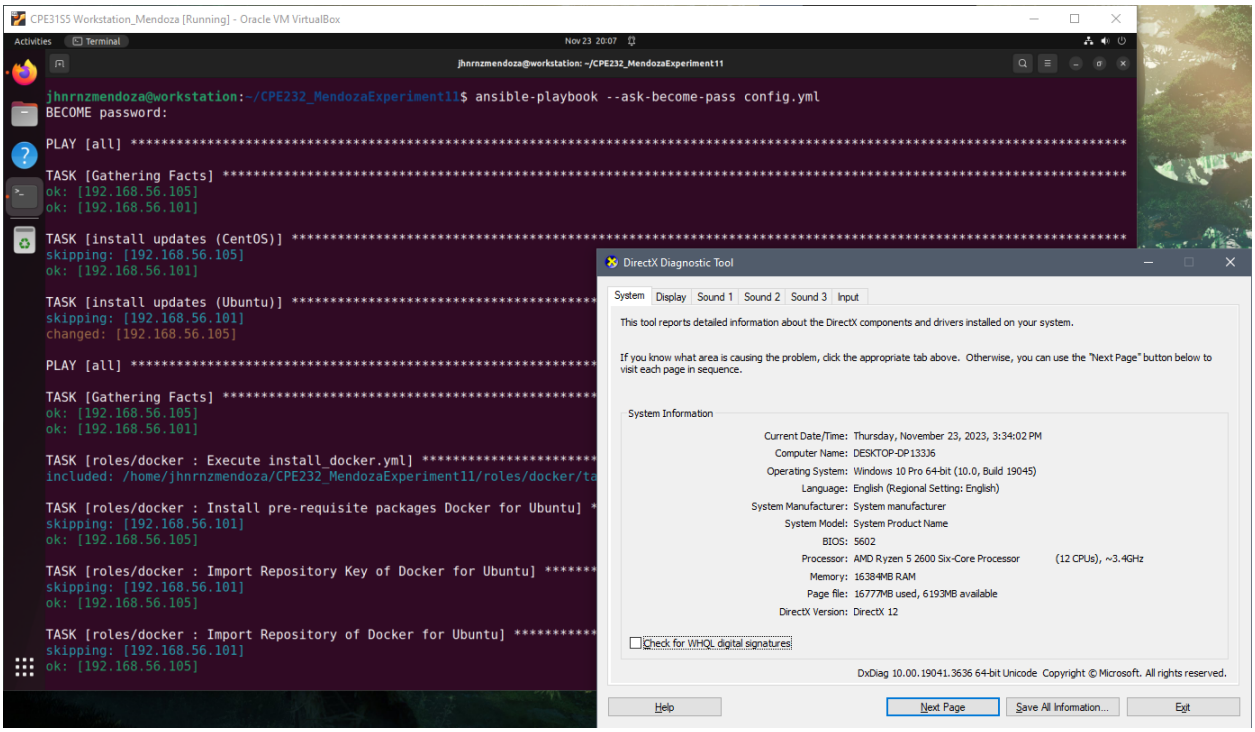

Observation:

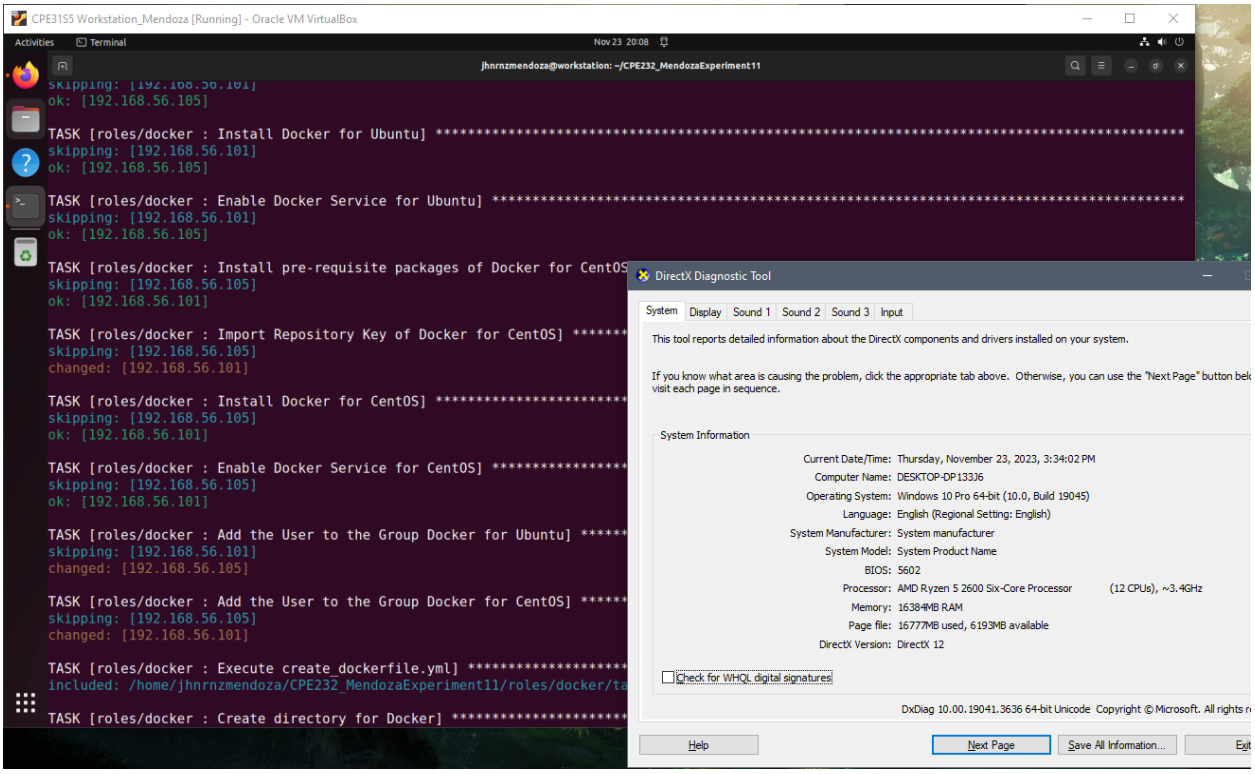
- Initially, I have created a directory and a file to store the dockerfile commands in the succeeding step.
- The docker file will run commands which are to install Web and DB services on their respective remote servers. Apache2 for Ubuntu, while MariaDB and PHP for CentOS.

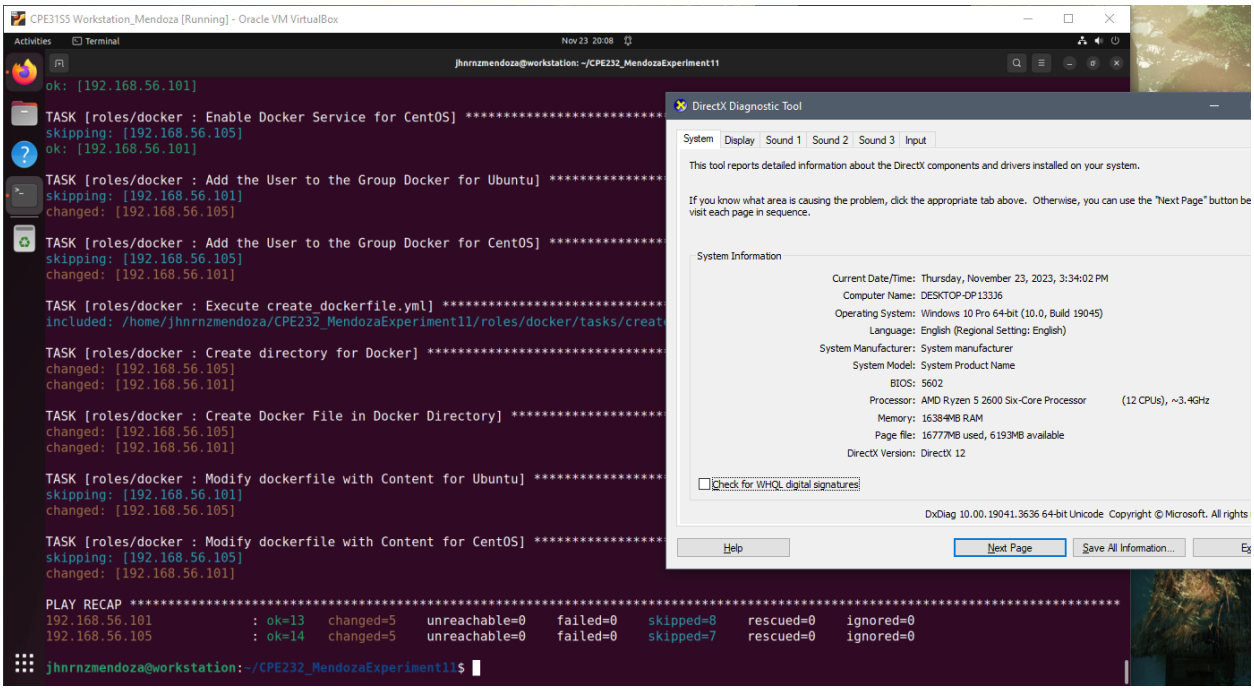

CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox
Activities
Terminal
Nov 23 20:22
jhnrnzmendoza@workstation: ~/CPE232_MendozaExperiment11
GNU nano 6.2
roles/docker/tasks/create dockerfile.yml
shell: 'sudo mkdir /home/jhnrnzmendoza/docker'
- name: Create Docker File in Docker Directory
shell: 'sudo touch /home/jhnrnzmendoza/docker/dockerfile'
# Tasks to Create a DockerFile Content
- name: Modify dockerfile with Content for Ubuntu
copy:
dest: "/home/jhnrnzmendoza/docker/dockerfile"
content: |
FROM Ubuntu
MAINTAINER jhnrnzmendoza <qjrlmendoza01@tip.edu.ph>
ARG DEBIAN_FRONTEND=noninteractive
RUN apt install apache2 -y
ENTRYPOINT apache2ctl -D FOREGROUND
when: ansible_distribution == "Ubuntu"
- name: Modify dockerfile with Content for CentOS
copy:
dest: "/home/jhnrnzmendoza/docker/dockerfile"
content: |
FROM CentOS
MAINTAINER jhnrnzmendoza <qjrlmendoza01@tip.edu.ph>
ARG REDHAT_FRONTEND=noninteractive
RUN apt install php -y
RUN yum install mariadb-server -y
RUN yum install mariadb-client -y
ENTRYPOINT mariadb-server -D FOREGROUND
ENTRYPOINT php -D FOREGROUND
when: ansible_distribution == "CentOS"
^G Help
^X Exit
^O Write Out
^R Read File
^W Where Is
^\ Replace
^K Cut
^U Paste
DirectX Diagnostic Tool
System
Display
Sound 1
Sound 2
Sound 3
Input
This tool reports detailed information about the DirectX components and drivers installed on your system.
If you know what area is causing the problem, click the appropriate tab above. Otherwise, you can use the "Next Page" button below to visit each page in sequence.
System Information
Current Date/Time: Thursday, November 23, 2023, 3:34:02 PM
Computer Name: DESKTOP-DP133J6
Operating System: Windows 10 Pro 64-bit (10.0, Build 19045)
Language: English (Regional Setting: English)
System Manufacturer: System manufacturer
System Model: System Product Name
BIOS: 5602
Processor: AMD Ryzen 5 2600 Six-Core Processor (12 CPUs), ~3.4GHz
Memory: 16384MB RAM
Page file: 16777MB used, 6193MB available
DirectX Version: DirectX 12
Check for WHQL digital signatures
DxDiag 10.00.19041.3636 64-bit Unicode Copyright © Microsoft. All rights reserved.
Help
Next Page
Save All Information...
Exit

5. Install and build the Dockerfile using Ansible.

Ansible Playbook Run:



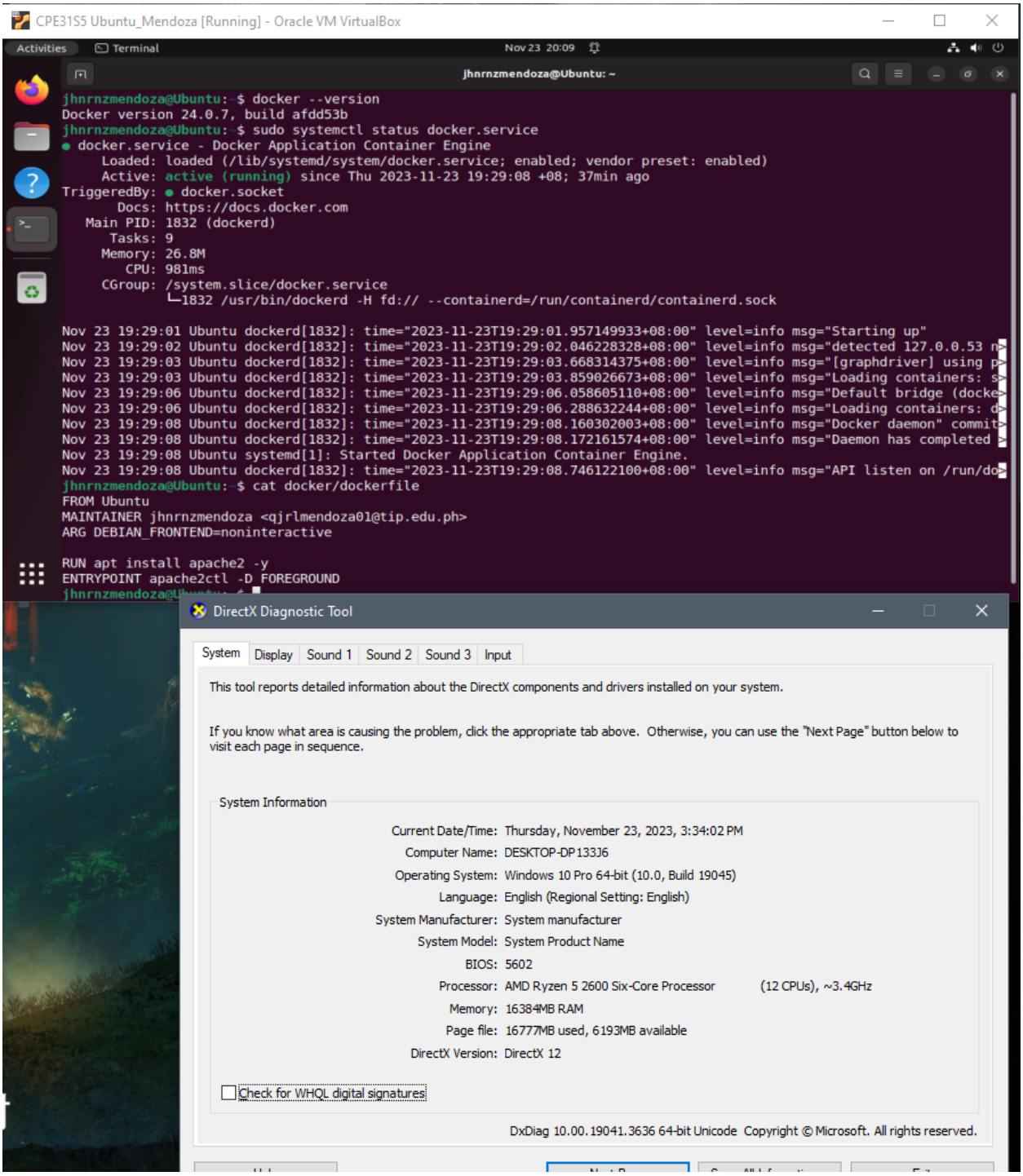



Observation:

- As we can observe, the tasks defined on the .yml files that have been created in this activity were included in the playbook run. To check if the results are valid, we may check the changes manually on the remote virtual machines.

**Proof of Docker Installed and DockerFile created on Remote Servers**

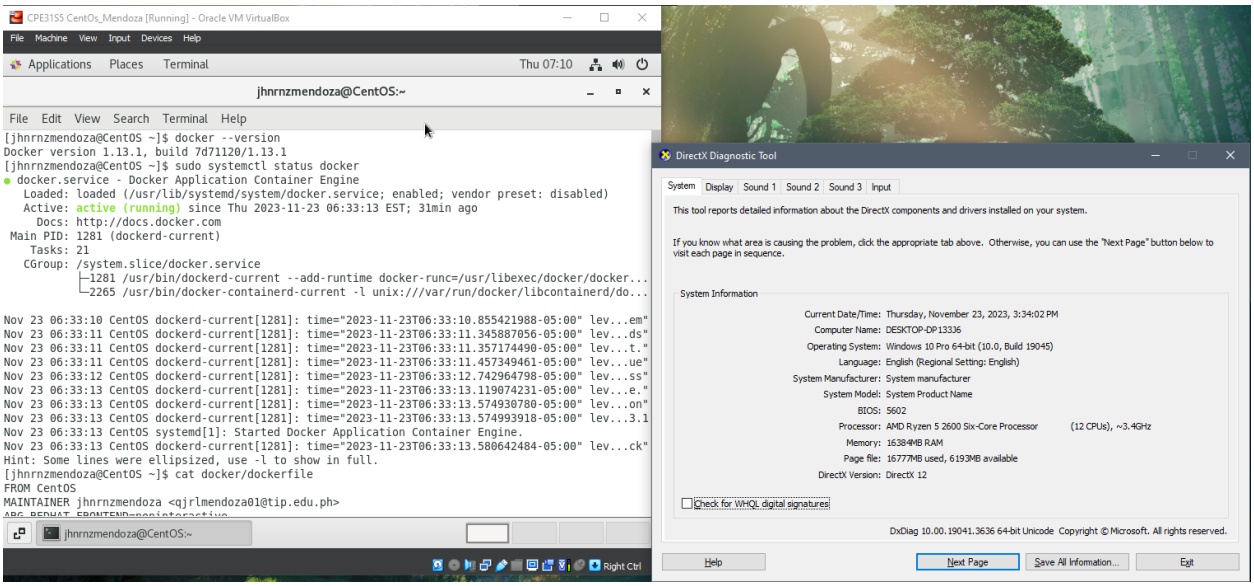
Ubuntu:

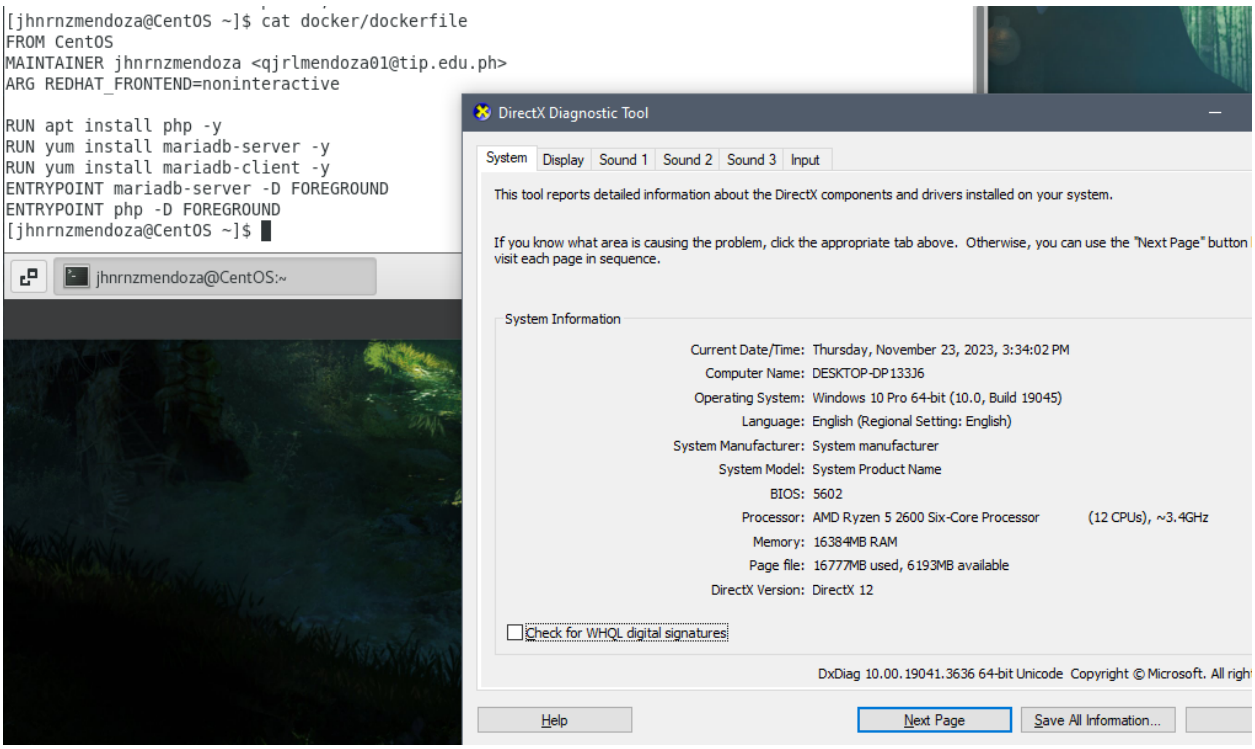



Observation:

- Using the systemctl, and docker –version, we can observe that the docker is successfully enabled and installed. In addition, the dockerfile was successfully added as well.

CentOS:

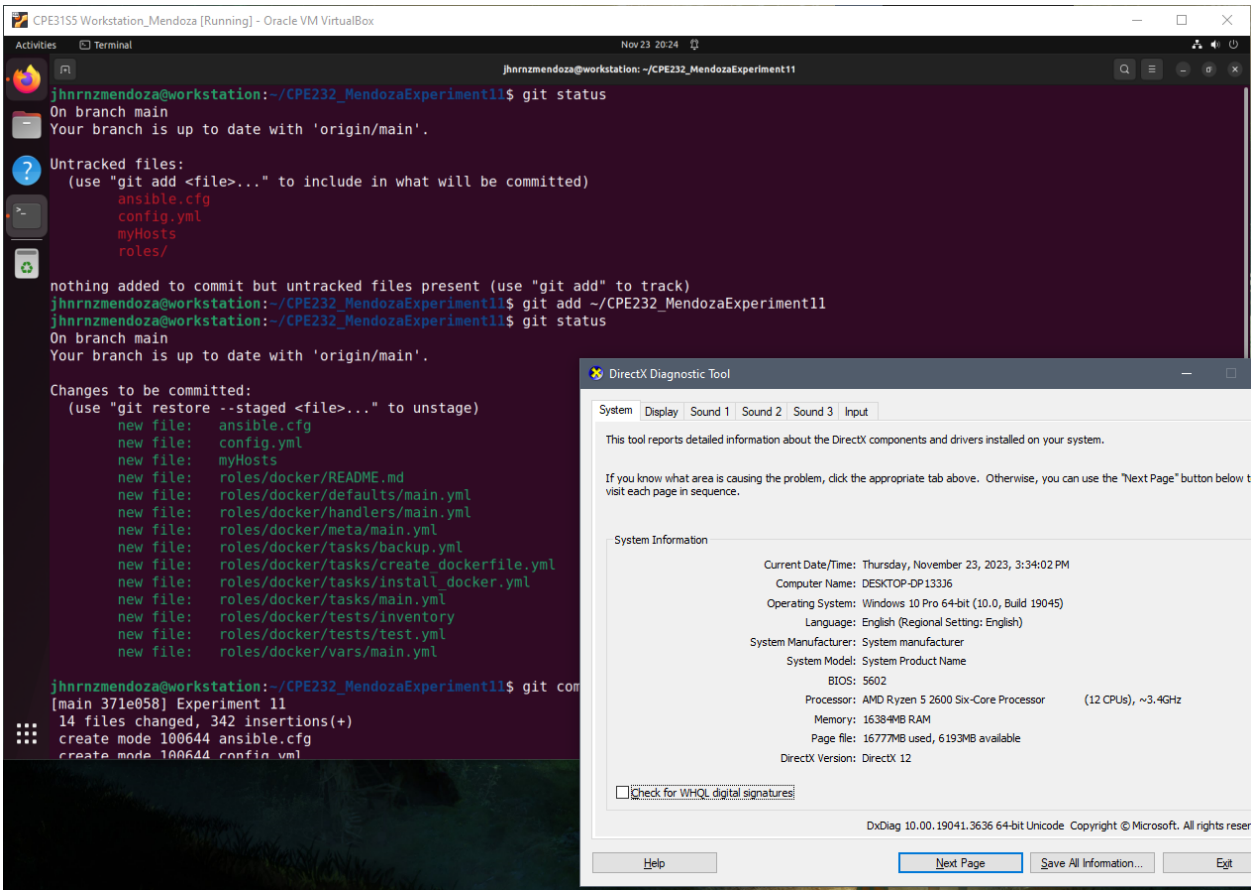


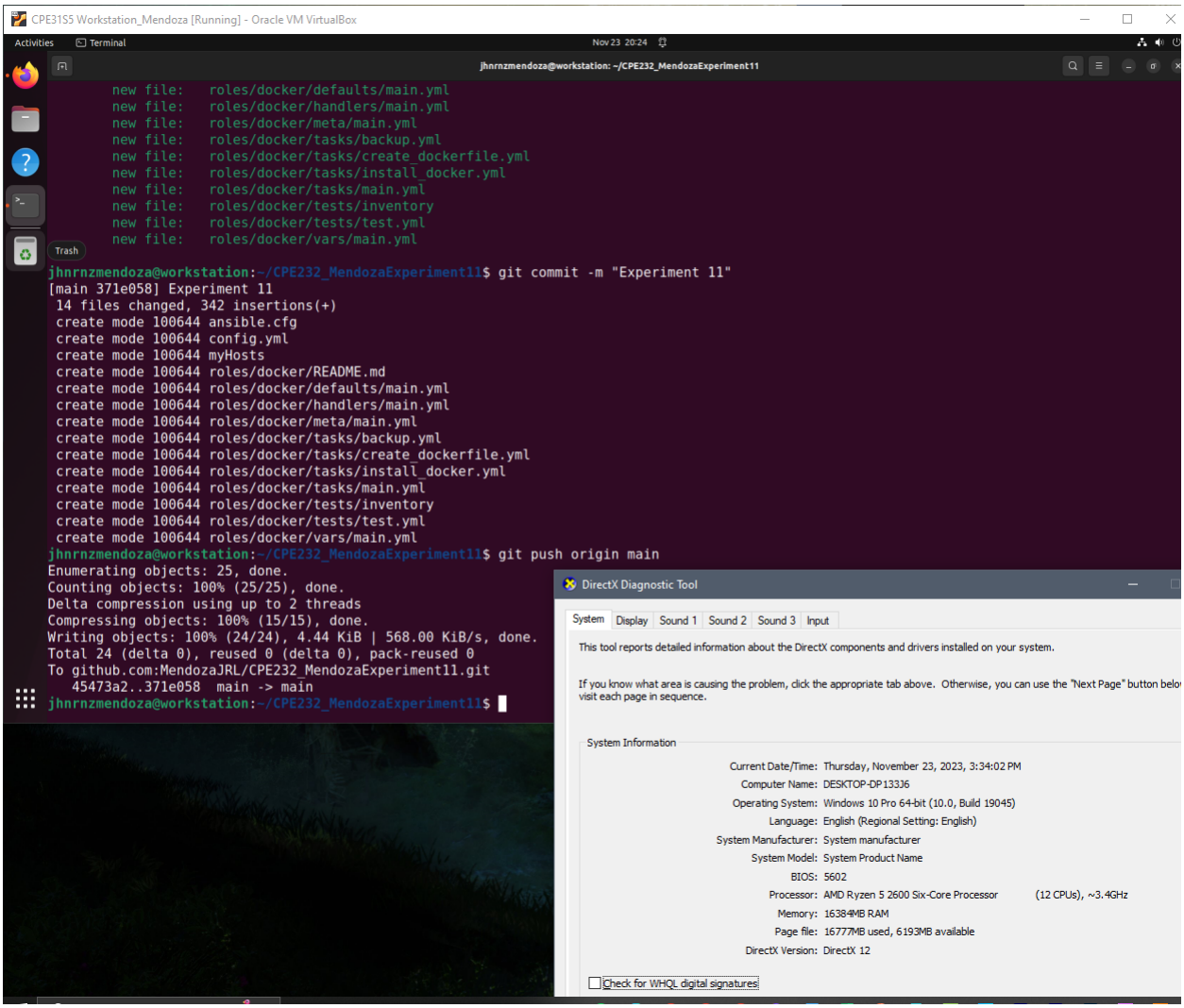



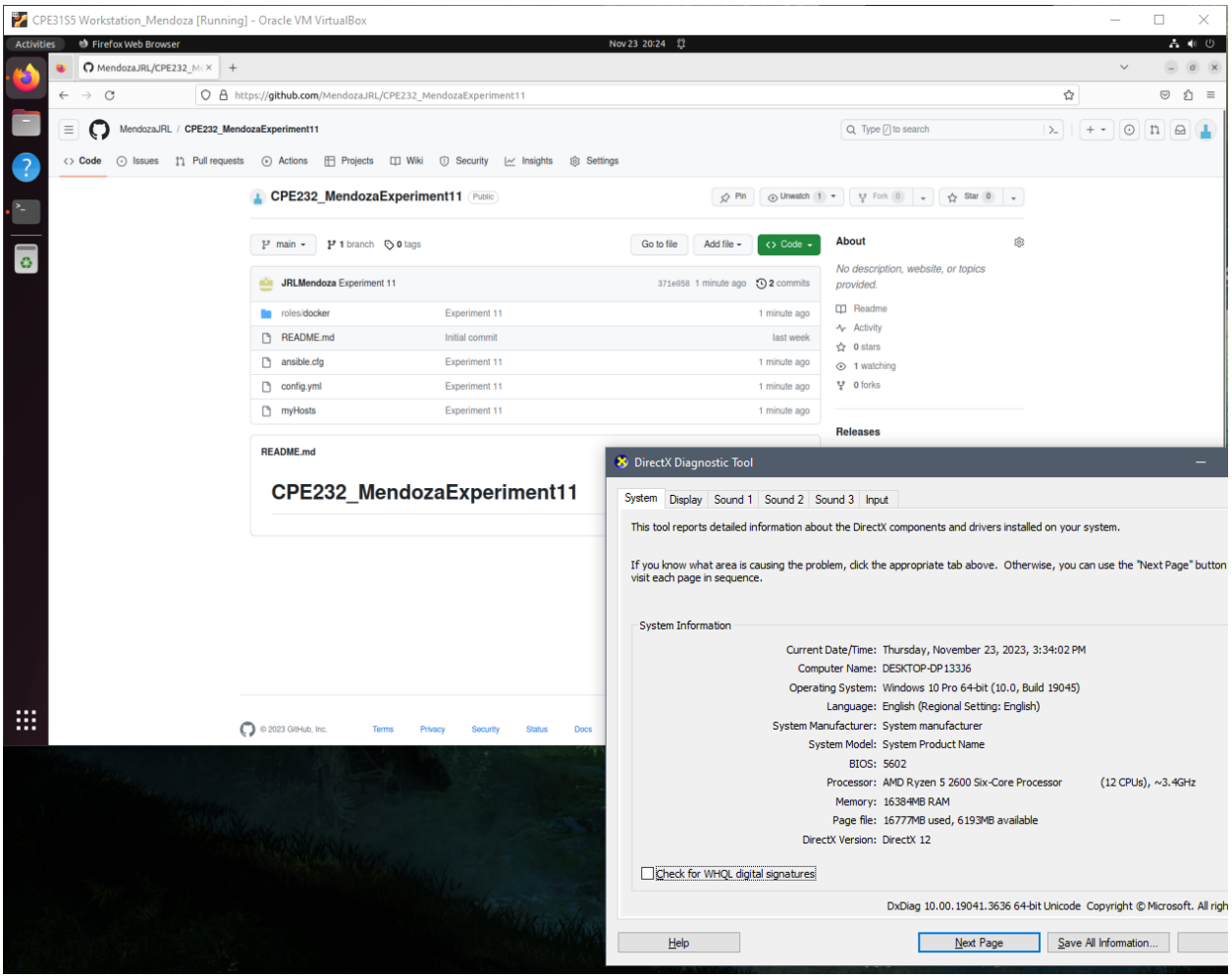

Observation:

- Using the systemctl, and docker –version, we can observe that the docker is successfully enabled and installed. In addition, the dockerfile was successfully added as well.

6. Add, commit and push it to your repository.




CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox
Activities
Terminal
Nov 23 20:24
jhnrnzmendoza@workstation: ~/CPE232_MendozaExperiment11
new file: roles/docker/defaults/main.yml
new file: roles/docker/handlers/main.yml
new file: roles/docker/meta/main.yml
new file: roles/docker/tasks/backup.yml
new file: roles/docker/tasks/create_dockerfile.yml
new file: roles/docker/tasks/install_docker.yml
new file: roles/docker/tasks/main.yml
new file: roles/docker/tests/inventory
new file: roles/docker/tests/test.yml
new file: roles/docker/vars/main.yml
Trash
jhnrnzmendoza@workstation:~/CPE232_MendozaExperiment11$ git commit -m "Experiment 11"
[main 371e058] Experiment 11
14 files changed, 342 insertions(+)
create mode 100644 ansible.cfg
create mode 100644 config.yml
create mode 100644 myHosts
create mode 100644 roles/docker/README.md
create mode 100644 roles/docker/defaults/main.yml
create mode 100644 roles/docker/handlers/main.yml
create mode 100644 roles/docker/meta/main.yml
create mode 100644 roles/docker/tasks/backup.yml
create mode 100644 roles/docker/tasks/create_dockerfile.yml
create mode 100644 roles/docker/tasks/install_docker.yml
create mode 100644 roles/docker/tasks/main.yml
create mode 100644 roles/docker/tests/inventory
create mode 100644 roles/docker/tests/test.yml
create mode 100644 roles/docker/vars/main.yml
jhnrnzmendoza@workstation:~/CPE232_MendozaExperiment11$ git push origin main
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 2 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (24/24), 4.44 KiB | 568.00 KiB/s, done.
Total 24 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:MendozaJRL/CPE232_MendozaExperiment11.git
45473a2..371e058 main -> main
jhnrnzmendoza@workstation:~/CPE232_MendozaExperiment11$
DirectX Diagnostic Tool
System
Display
Sound 1
Sound 2
Sound 3
Input
This tool reports detailed information about the DirectX components and drivers installed on your system.
If you know what area is causing the problem, click the appropriate tab above. Otherwise, you can use the "Next Page" button below
visit each page in sequence.
System Information
Current Date/Time: Thursday, November 23, 2023, 3:34:02 PM
Computer Name: DESKTOP-DP13336
Operating System: Windows 10 Pro 64-bit (10.0, Build 19045)
Language: English (Regional Setting: English)
System Manufacturer: System manufacturer
System Model: System Product Name
BIOS: 5602
Processor: AMD Ryzen 5 2600 Six-Core Processor (12 CPUs), ~3.4GHz
Memory: 16384MB RAM
Page file: 16777MB used, 6193MB available
DirectX Version: DirectX 12
Check for WHQL digital signatures




Observation:

- Using git commands, I have successfully pushed all of the changes to the cloud repository and will be able to see them publicly on my GitHub Link.

  GitHub Link: [https://github.com/MendozaJRL/CPE232_MendozaExperiment11](https://github.com/MendozaJRL/CPE232_MendozaExperiment11)

**Reflections:**

Answer the following:

1. What are the benefits of implementing containerizations?

- Containerization is a process in which we also divide a single computer into different parts. The operating system, its kernel, and protocols are bound to be the same on each created sub-machine. Instead of physically segmenting the computer resources, containerization utilizes the same resources for all separate machines as it only runs in one operating system. It is only divided based on the user.

- The Benefits of Containerization is that it only uses one operating system, in which if it is updated, the other containers would also be updated thus making the application updates as well. With that, the implementation is more portable as it is only concerned with lightweight software and does not consider the capability of the hardware as a main factor in order for it to work. It is also easier to expand, as it does not require high-end hardware in order to be implemented.

**Conclusions:**

In this activity, the students were introduced to containerization, specifically the docker architecture. Containerization has the similar purpose with virtualization but is more capable in the development operations. As the containers do not have their own operating systems, it makes the processes and configuration much easier to implement. In this activity, we have installed docker on the remote servers with the use of ansible. With the use of ansible, the installation is much more efficient and organized, as we will just import the tasks from the playbook files as needed by the remote server. As observed, we have successfully installed docker on the remote servers and are able to create a docker file along with it. A docker file contains a set of commands with its own syntax (similar to shell script) that is run in order to create a docker image. A docker image on the other hand is a prerequisite in order to be executed and create a docker container. Lastly, the docker container is implemented that uses the same operating system but is different on application or on the software level. In addition, the student believes that containers would be very helpful as it would not be too restricted when implementing multiple machines. Unlike with Virtualization which cannot be scaled or expanded easily.