Name: John Renzo L. Mendoza	**Date Performed:** October 24, 2023
Course/Section: CPE31S5	**Date Submitted:** October 25, 2023
Instructor: Engr. Roman Richard	**Semester and SY:**

Activity 9: Install, Configure, and Manage Performance Monitoring tools

1. Objectives

Create and design a workflow that installs, configure and manage enterprise performance tools using Ansible as an Infrastructure as Code (IaC) tool.

2. Discussion

Performance monitoring is a type of monitoring tool that identifies current resource consumption of the workload, in this page we will discuss multiple performance monitoring tools.

Prometheus

Prometheus fundamentally stores all data as time series: streams of timestamped values belonging to the same metric and the same set of labeled dimensions. Besides stored time series, Prometheus may generate temporary derived time series as the result of queries. Source: Prometheus - Monitoring system & time series database

Cacti

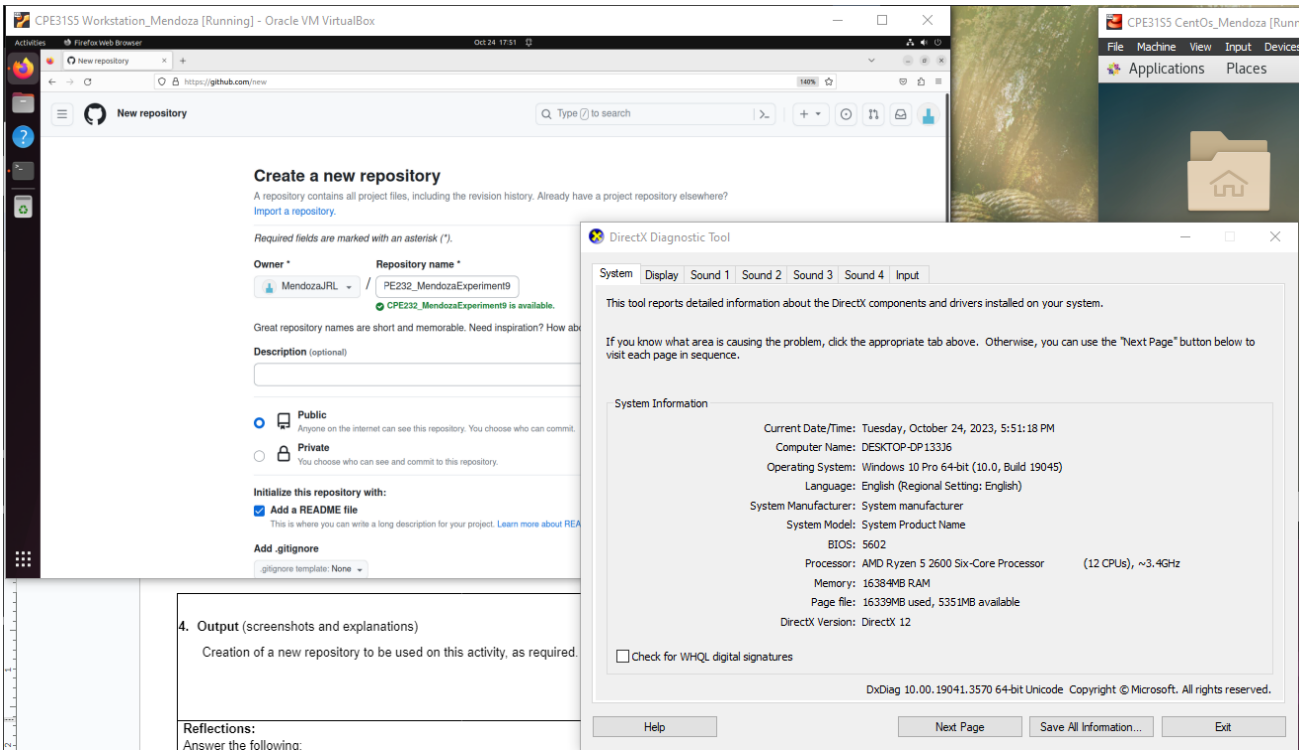
Cacti is a complete network graphing solution designed to harness the power of RRDTool's data storage and graphing functionality. Cacti provides a fast poller, advanced graph templating, multiple data acquisition methods, and user management features out of the box. All of this is wrapped in an intuitive, easy to use interface that makes sense for LAN-sized installations up to complex networks with thousands of devices. Source: Cacti® - The Complete RRDTool-based Graphing Solution

3. Tasks

1. Create a playbook that installs Prometheus in both Ubuntu and CentOS. Apply the concept of creating roles.
2. Describe how you did step 1. (Provide screenshots and explanations in your report. Make your report detailed such that it will look like a manual.)
3. Show an output of the installed Prometheus for both Ubuntu and CentOS.
4. Make sure to create a new repository in GitHub for this activity.

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

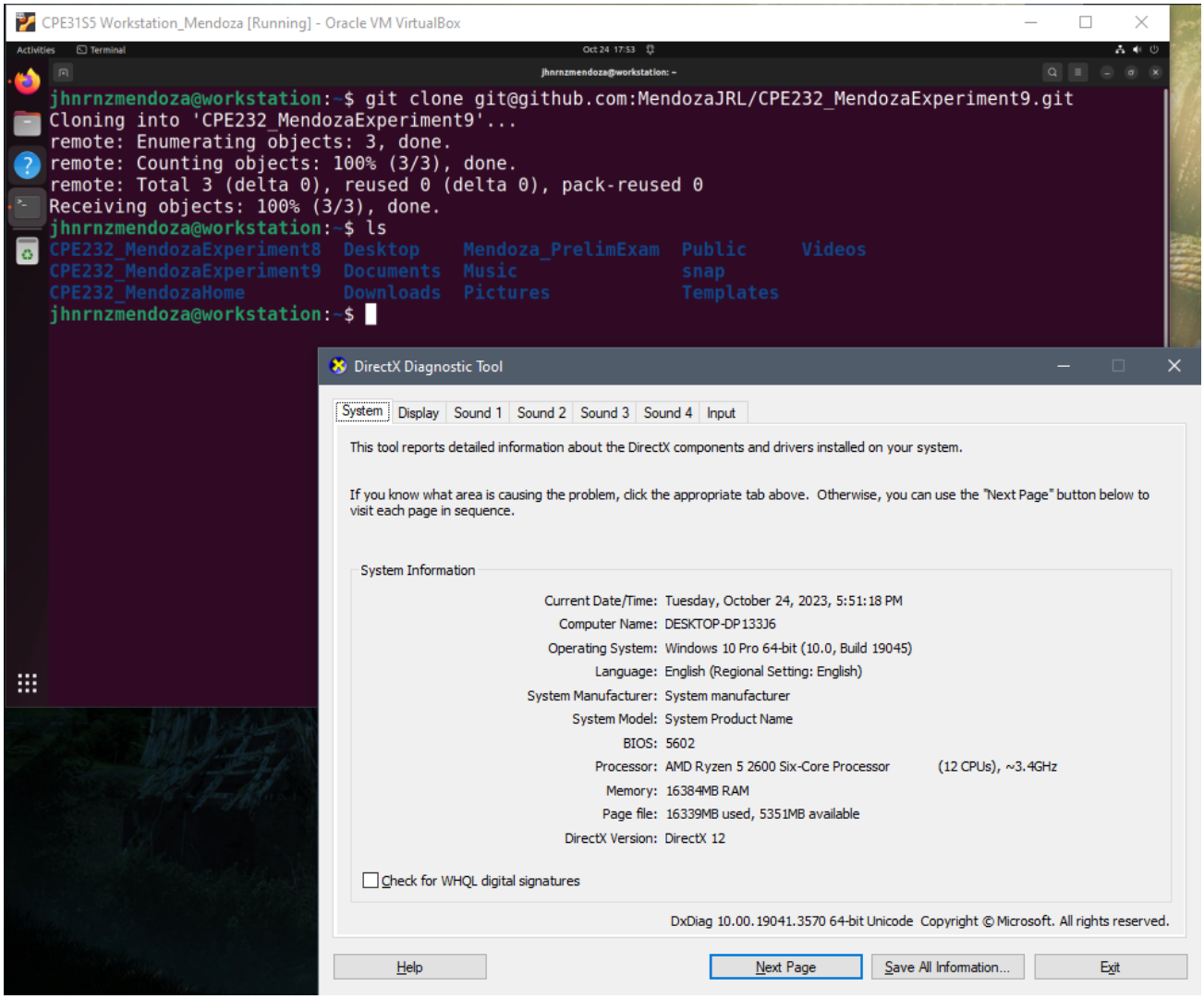
Creation of a new repository to be used on this activity, as required.



Observation:

- I have created a new repository CPE232_MendozaExperiment9 by making one in the GitHub website.

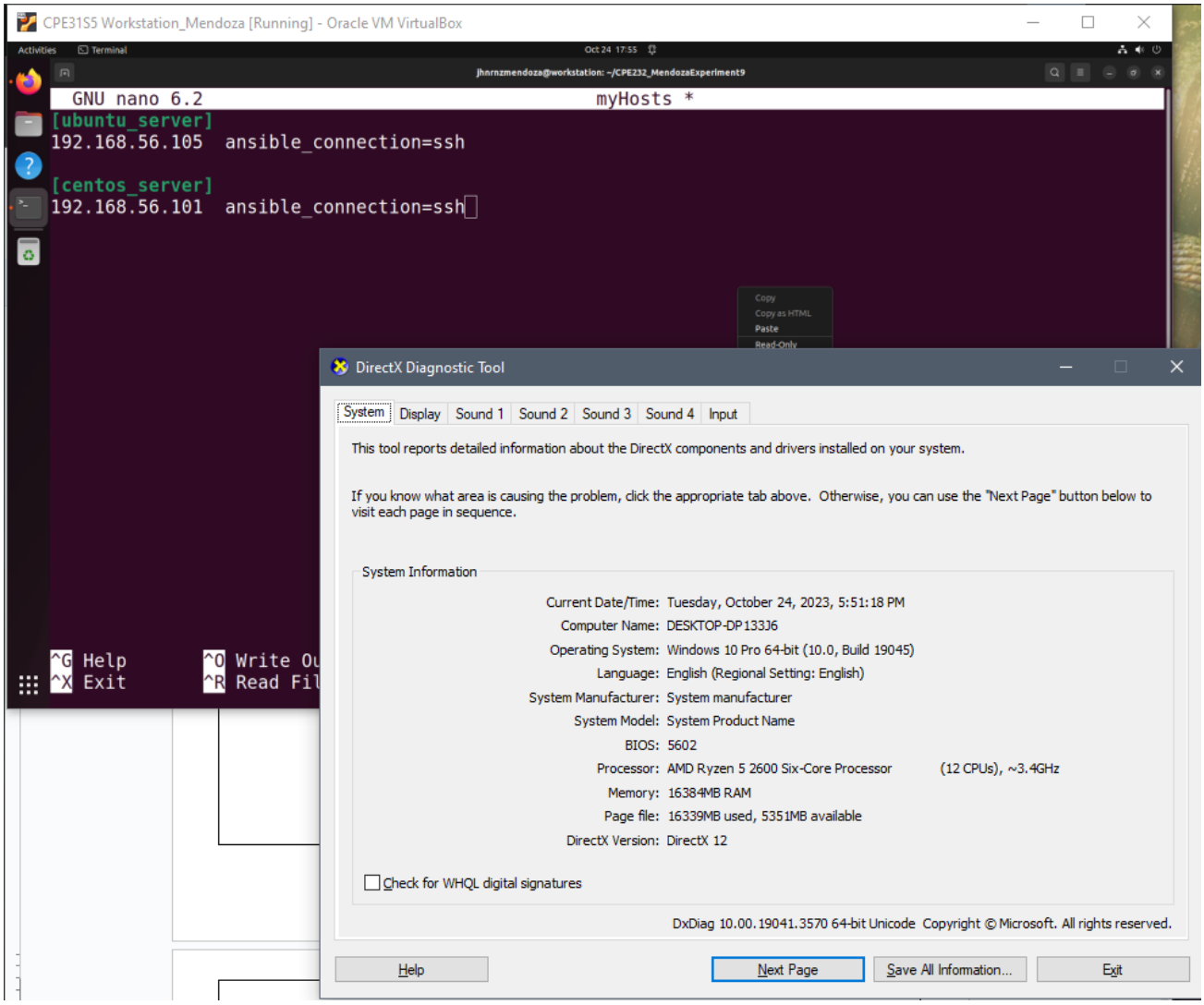
Cloning the created GitHub repository to the Control Node.



Observation:

- Using the `git clone` command, I have imported the created repository to the local control node.

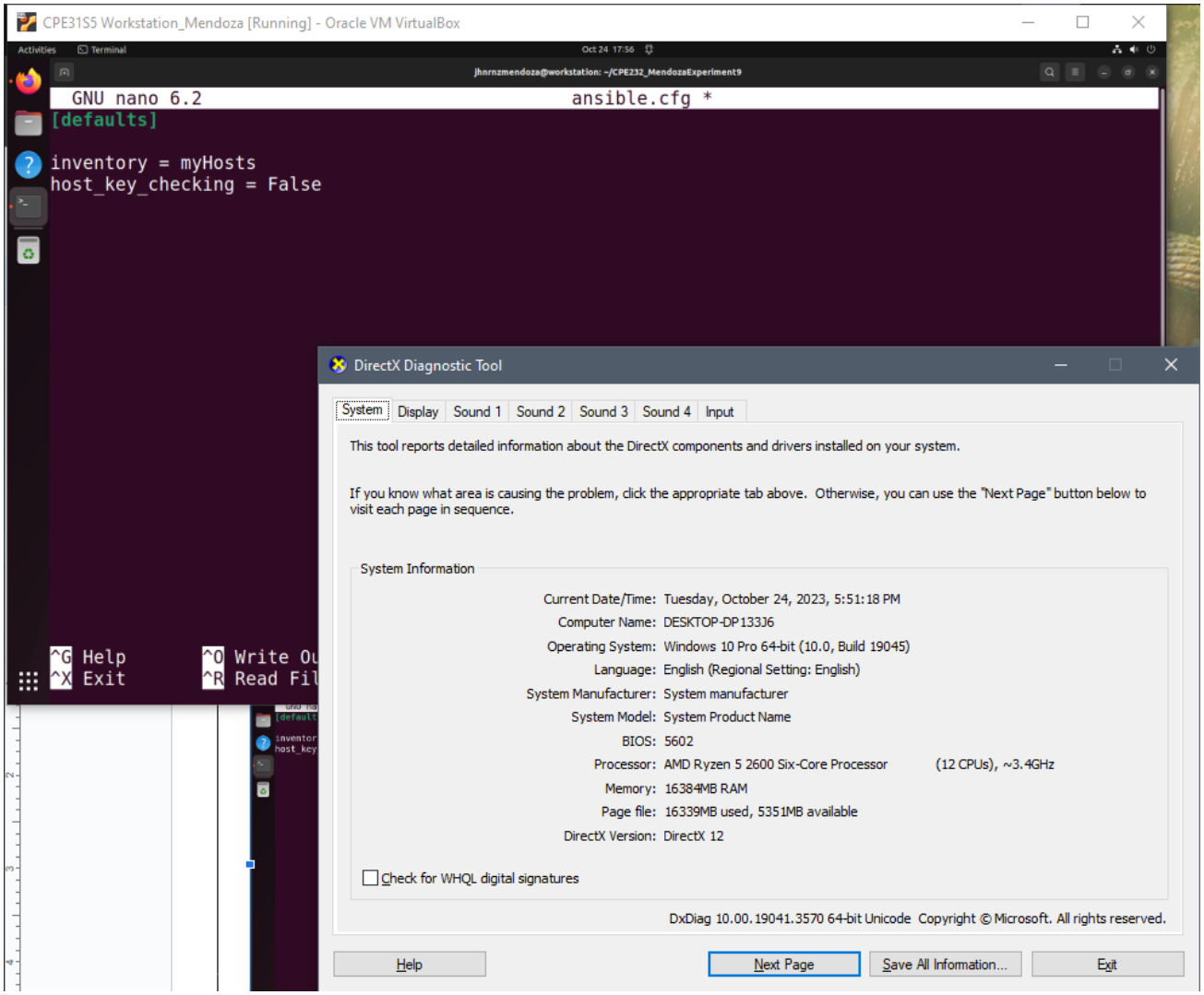
Creation of the inventory file “`myHosts`” under the cloned repository.



Observation:

- I have created `myHosts` as my inventory file which stores the ip addresses of the managed nodes.

Creation of the ansible configuration file `ansible.cfg`.



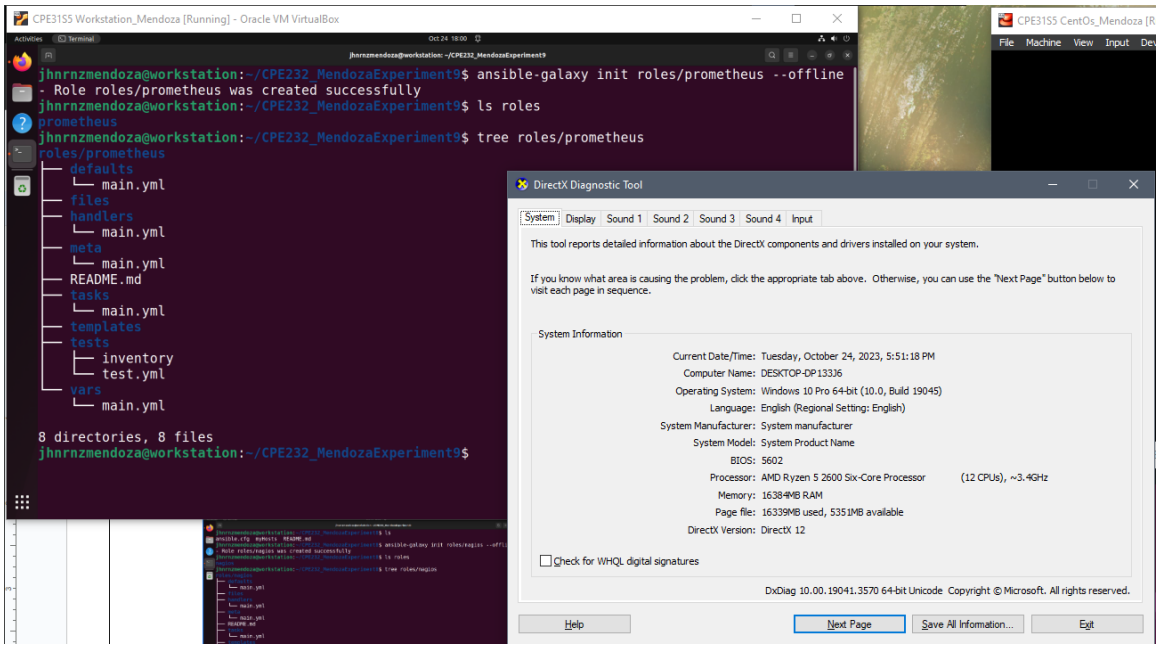
Observation:

- I have also created an ansible configuration file that stores the configuration needed for running ansible.

1. Create a playbook that installs Prometheus in both Ubuntu and CentOS. Apply the concept of creating roles.

2. Describe how you did step 1. (Provide screenshots and explanations in your report. Make your report detailed such that it will look like a manual.)

    a. Creation of ansible roles to be implemented in this experiment using the `ansible_galaxy` command.

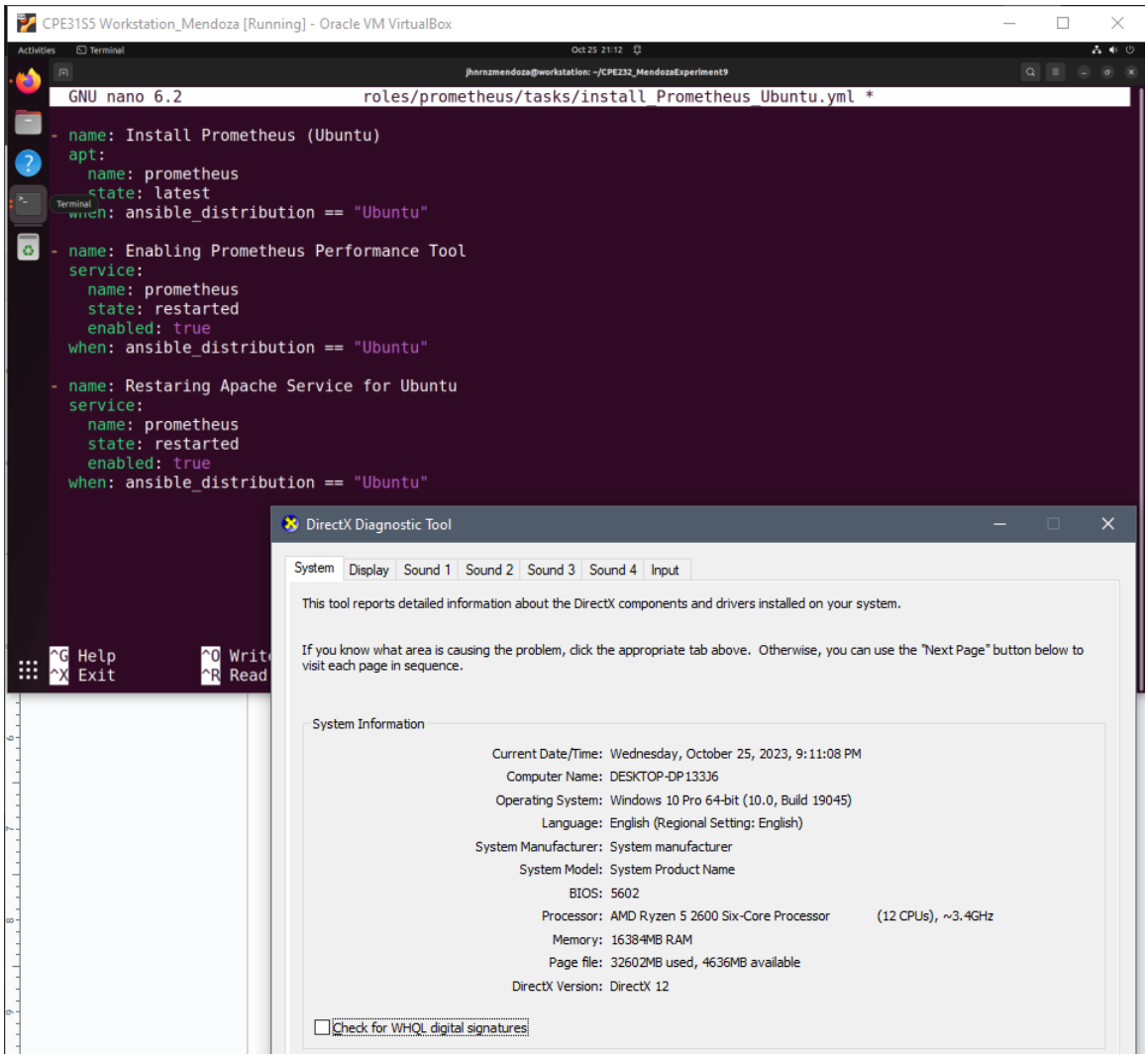


Observation:

- In this step, I have utilized the `ansible-galaxy` command in order to create the role prometheus which will be used to store the commands that are needed for installing prometheus for both Ubuntu and CentOS remote servers.
- The `tree roles` query will let us show the initial created role `prometheus`.

b. Creation of the playbook file that installs the Prometheus on the roles.

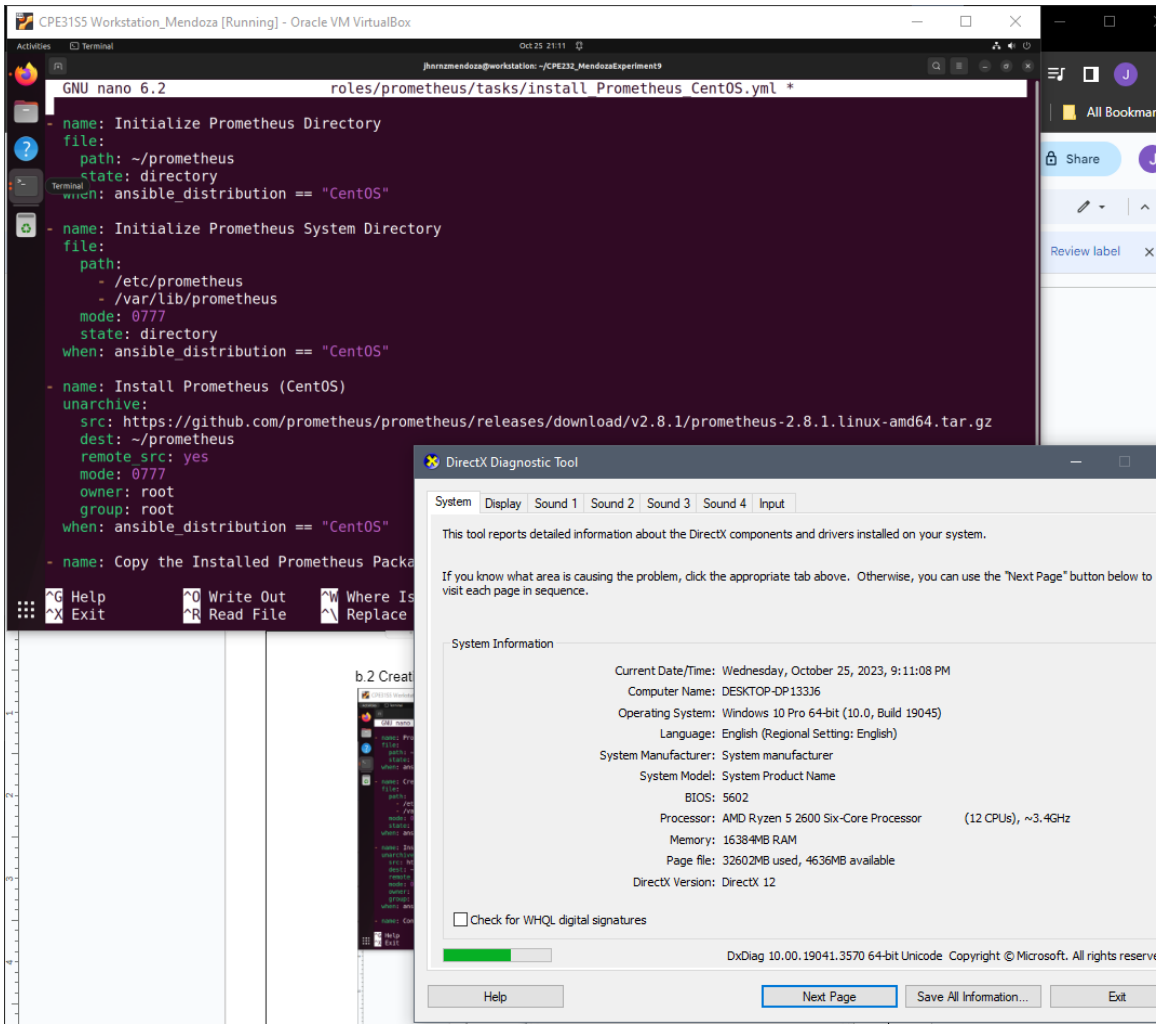
b.1 Creation of the necessary tasks to install Prometheus on Ubuntu.



Observation:

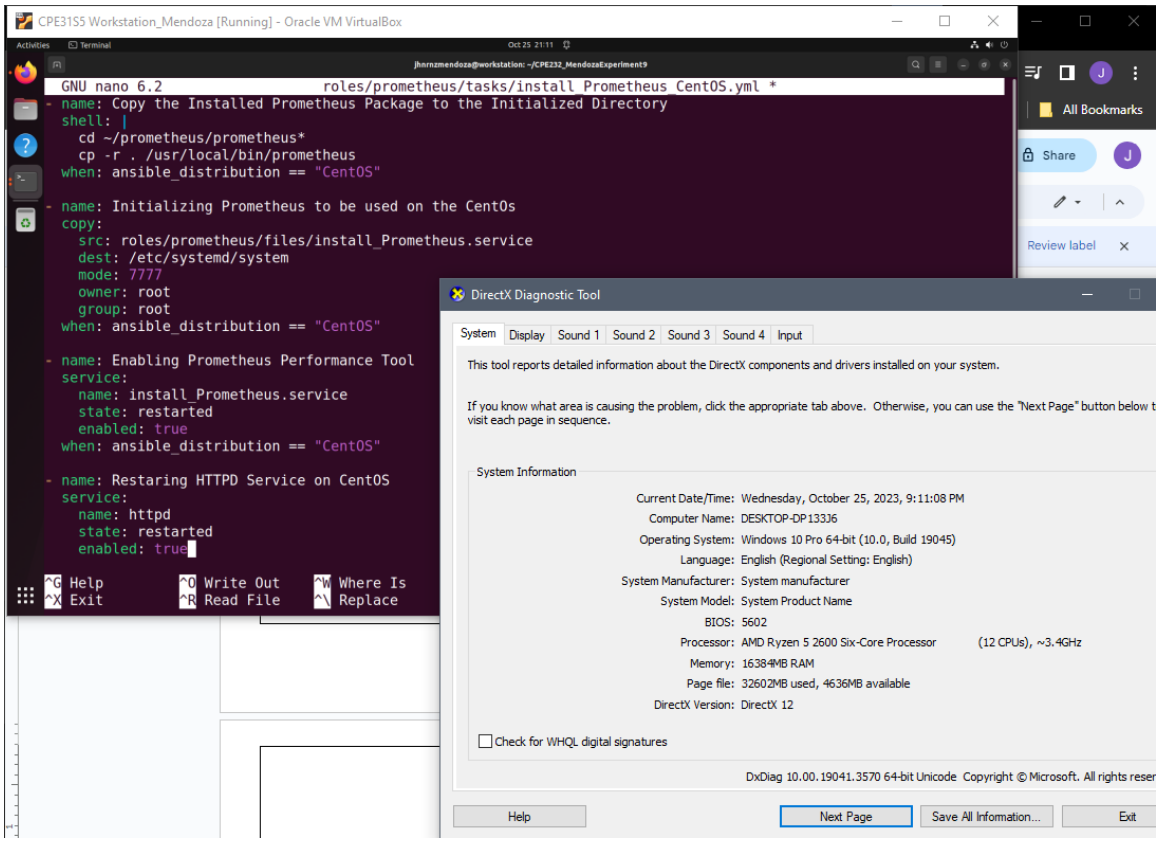
- To install the prometheus for Ubuntu, I have created a yaml file `(install_Prometheus_Ubuntu.yml)` which has the tasks needed for the installation.
- The first task simply installs the prometheus package to the Ubuntu distribution using the apt package manager.
- The second task restarts and enables the installed prometheus service so that the performance monitoring tool would be used.
- The third and last task is to restart the apache service so that we could observe or use the prometheus service on the web browser.

b.2 Creation of the necessary tasks to install Prometheus on CentOS.



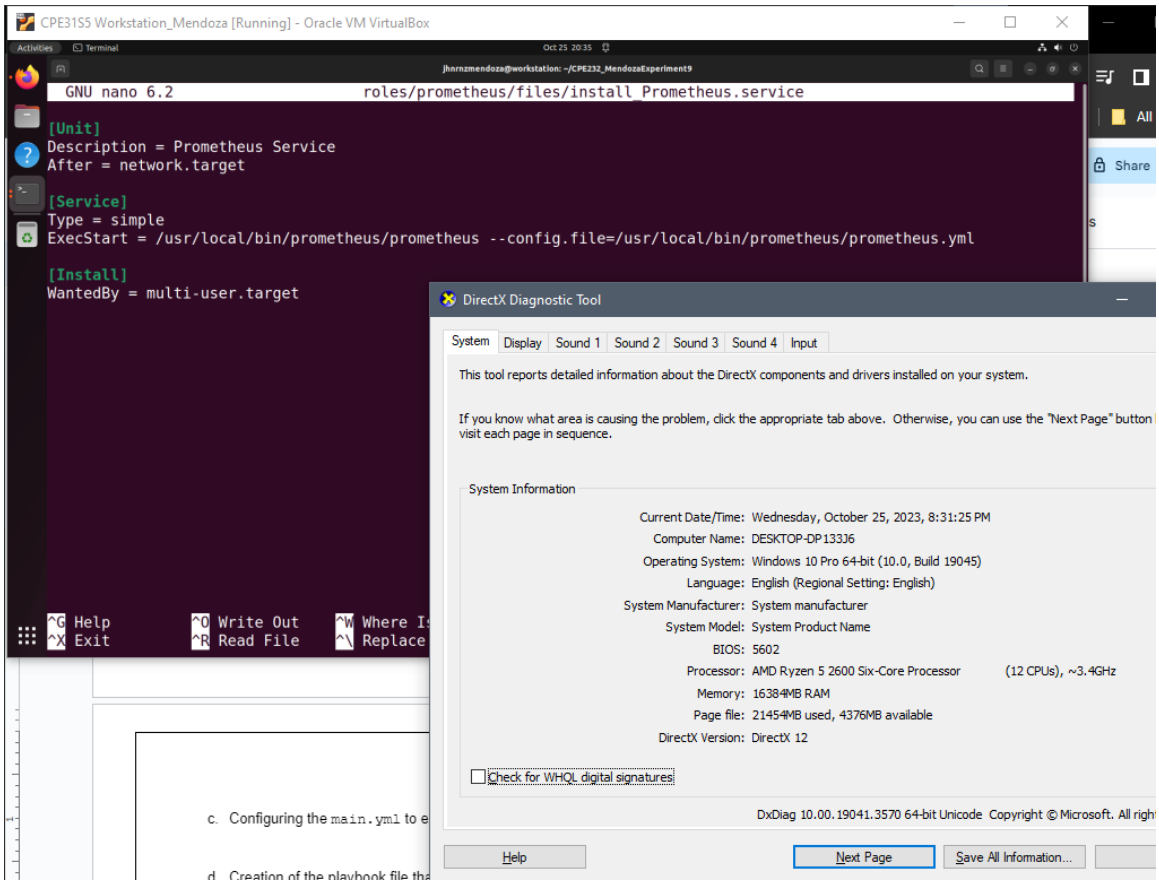
Observation:

- To install the prometheus for CentOS, I have created a yaml file (`install_Prometheus_CentOS.yml`) which has the tasks needed for the installation.
- The first task simply creates a new directory to store the prometheus package that would be installed.
- The second task will also create a new directory to store the prometheus package which is the same with the previous but in a different location since this is for running the service on the web browser.
- The third task is for installing prometheus using a remote link that provides the package. The destination folder would be the created directory on the first task. The owner and group is the root user to give privileges.



Observation:

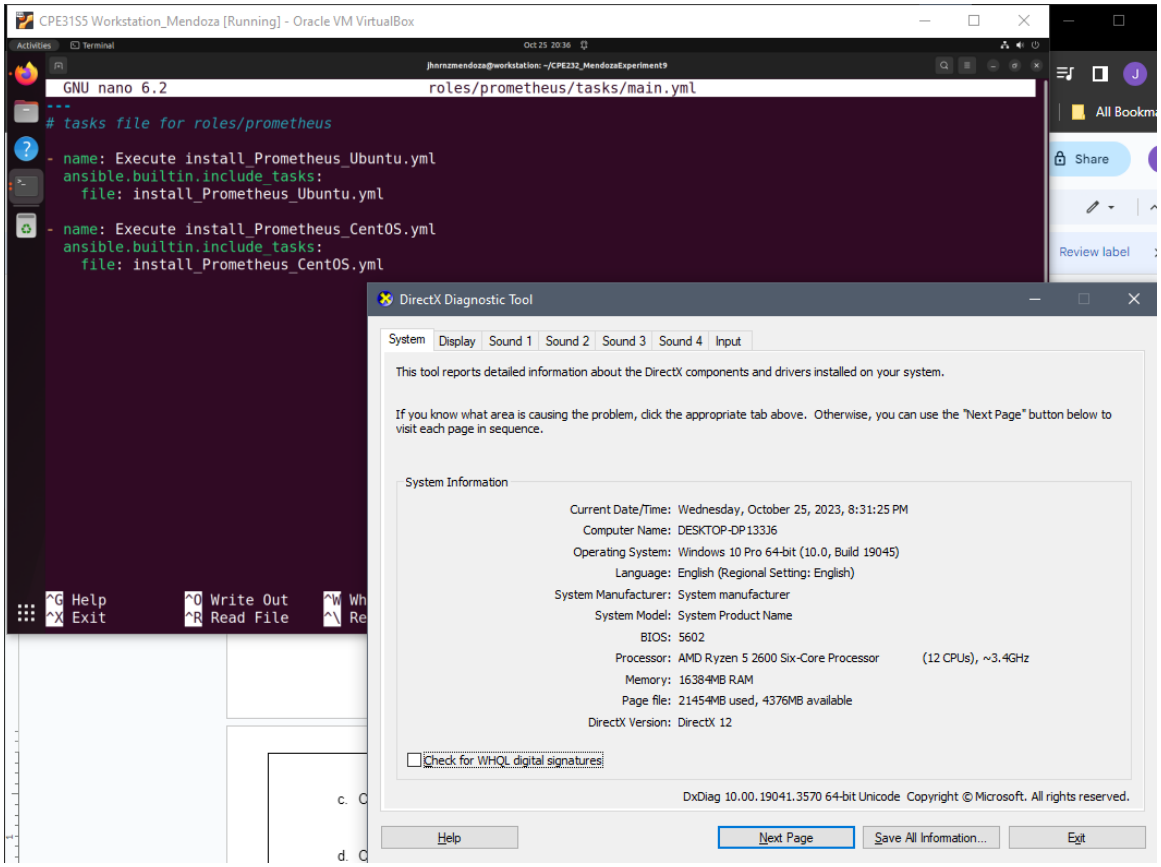
- The fourth task is to copy the package installed to the `usr/local/bin/` directory. This was needed so that the service will be able to be run by other users and so that when the service is updated, a backup of the old running version would not be easily overwritten.
- The fifth task is to initialize the installed package and to configure it in a way that it will be ready to use once the service has started.
- The sixth task is to enable the `prometheus` service on the system.
- The seventh and last task is to restart the `httpd` service so that the new service which is `prometheus` would be able to be used on the system as well as on the web browser.



Observation:

- This file is created on the files directory of the role prometheus. The purpose of this file is to configure the defaults of the prometheus service which will be used by the `install_Prometheus_CentOS.yml`. Specifically, this file was called on the sixth task, to configure the installed service before using it.

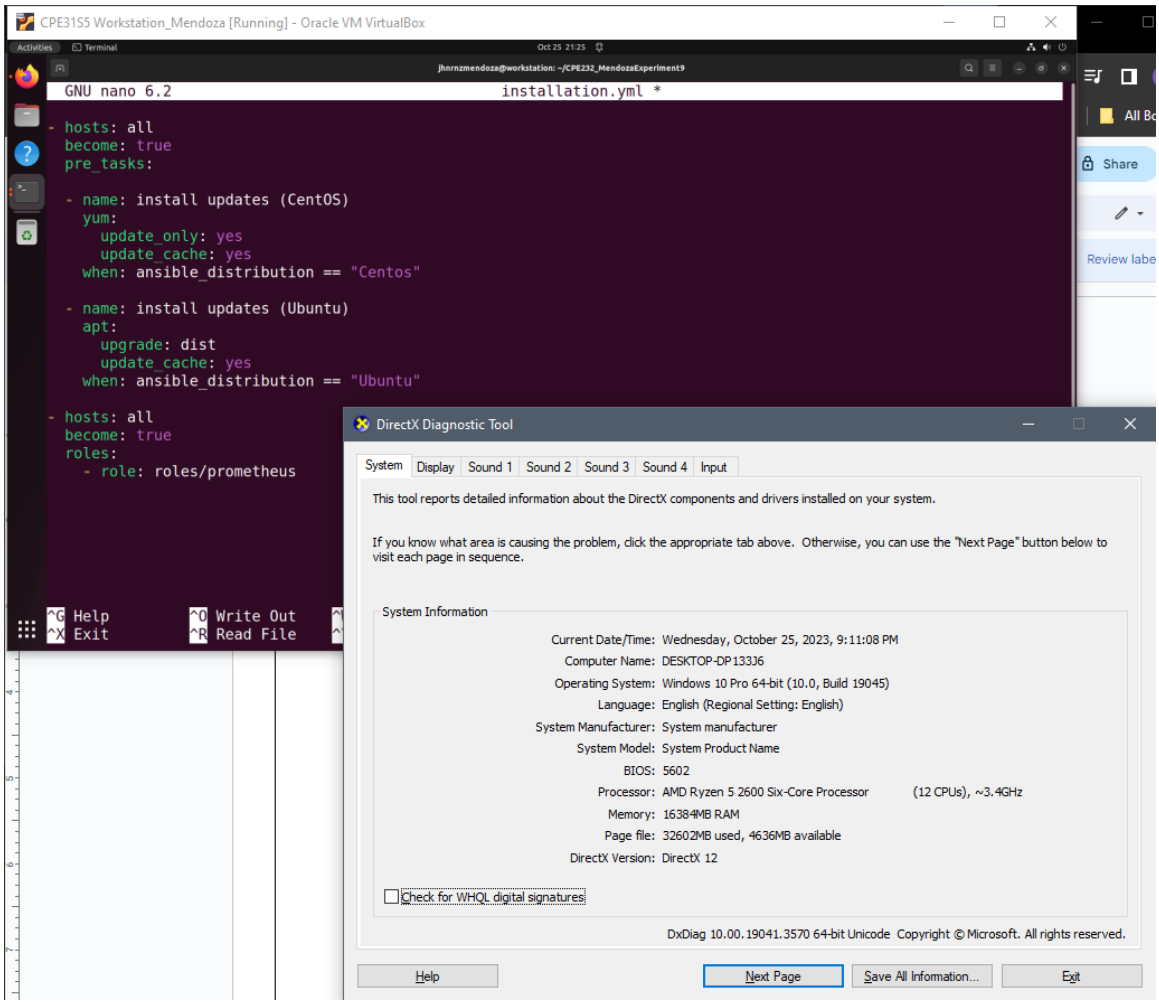
c. Configuring the `main.yml` to execute the created playbook.



Observation:

- In the `roles/prometheus/tasks/` directory, I have edited the main.yml so that it would include the tasks defined on the `install_Prometheus_Ubuntu.yml` and `install_Prometheus_CentOS.yml`. This was needed, so that the predefined tasks on previous steps would be executed by the ansible playbook file which run the role "`prometheus`"

d. Creation of the playbook file that executes the roles.

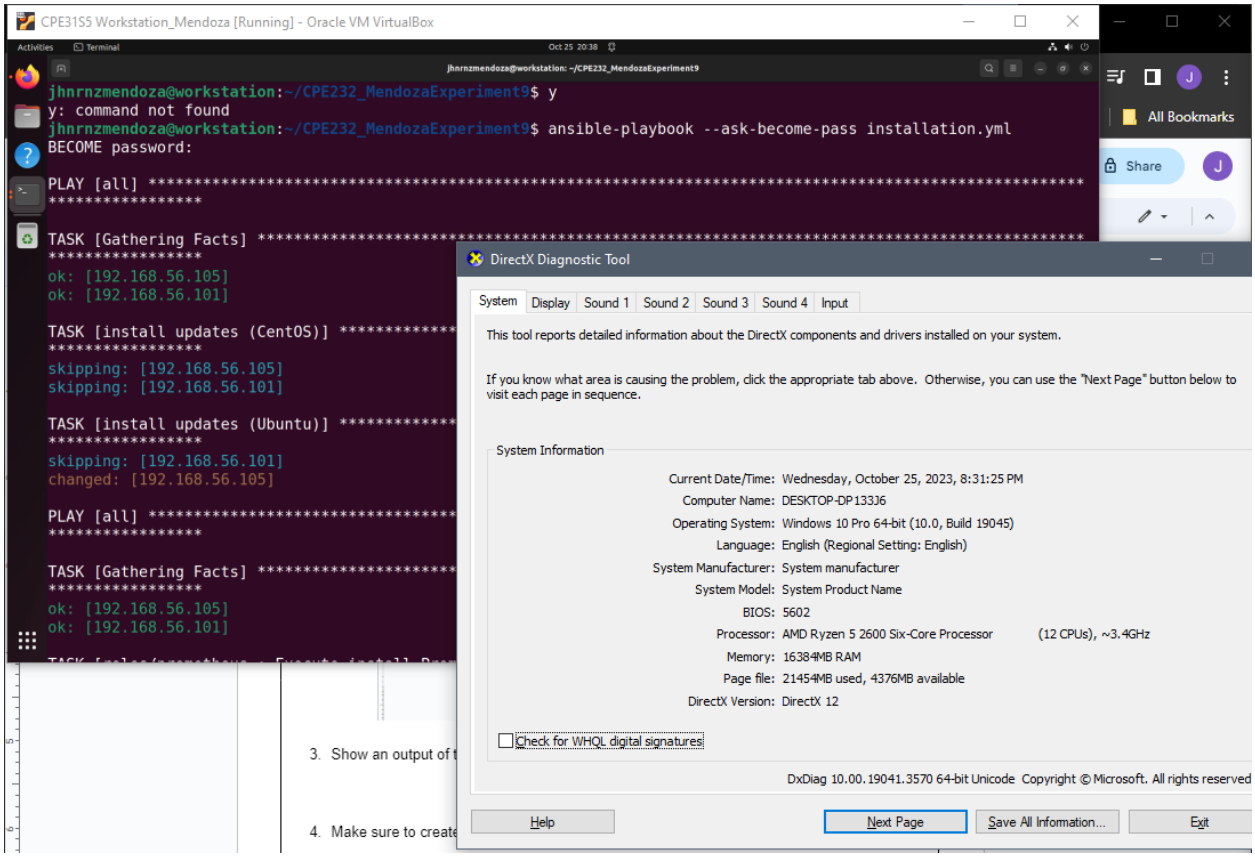


Observation:

- I have created an ansible playbook file that would run the role "`prometheus`". Basically, it will `update` the remote servers as defined on the pre-tasks. Then, it will call the role "prometheus" which will automatically execute the `main.yml` indicated in the respective called role. Finally, the main.yml imports tasks from `install_Prometheus_Ubuntu.yml` and `install_Prometheus_CentOS.yml` which are both used to perform the installation of Prometheus for the remote servers.

3. Show an output of the installed Prometheus for both Ubuntu and CentOS.

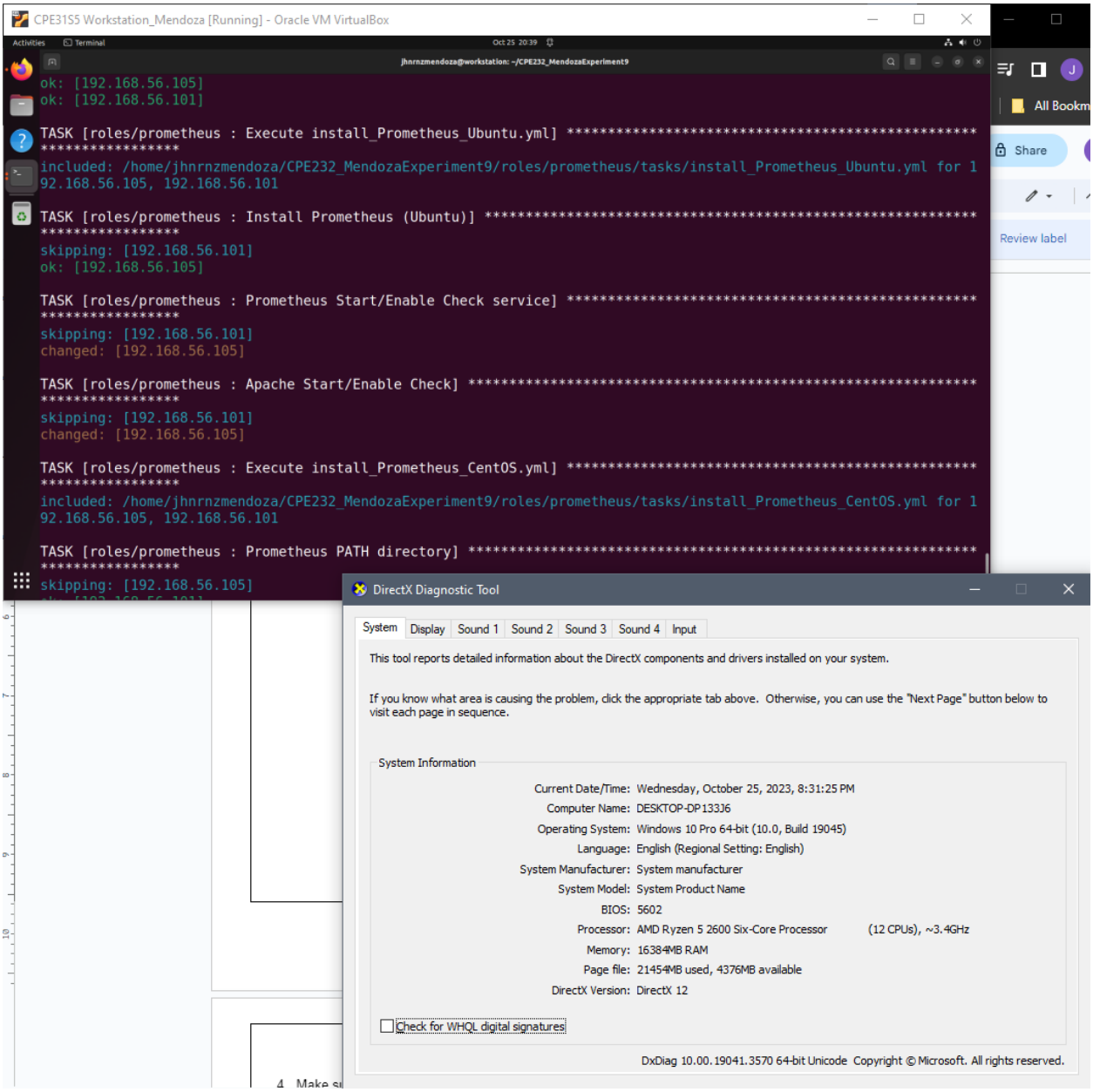
Output: Running the playbook `installation.yml` (Part 1)



Observation:

- As we can see, the playbook run was able to update both of the remote servers as its pre-task.

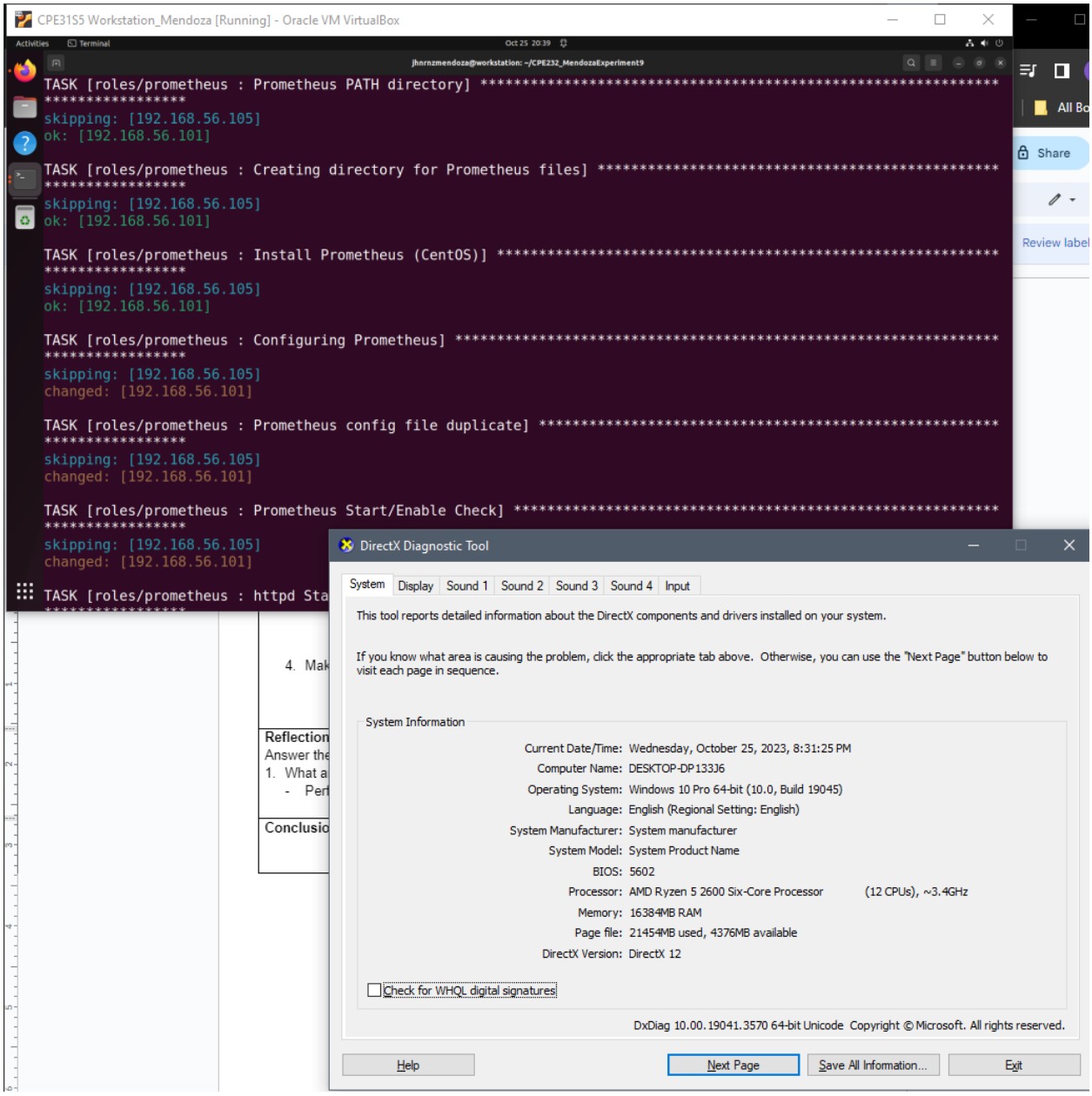
Output: Running the playbook `installation.yml` (Part 2)



Observation:

- The `installation.yml` was able to call the `main.yml` of the role "`prometheus`". The `main.yml` was also able to import tasks from the `install_Prometheus_Ubuntu.yml` and `install_Prometheus_CentOS.yml`. The tasks were able to do their processes while filtering the target remote servers using the conditional statements.

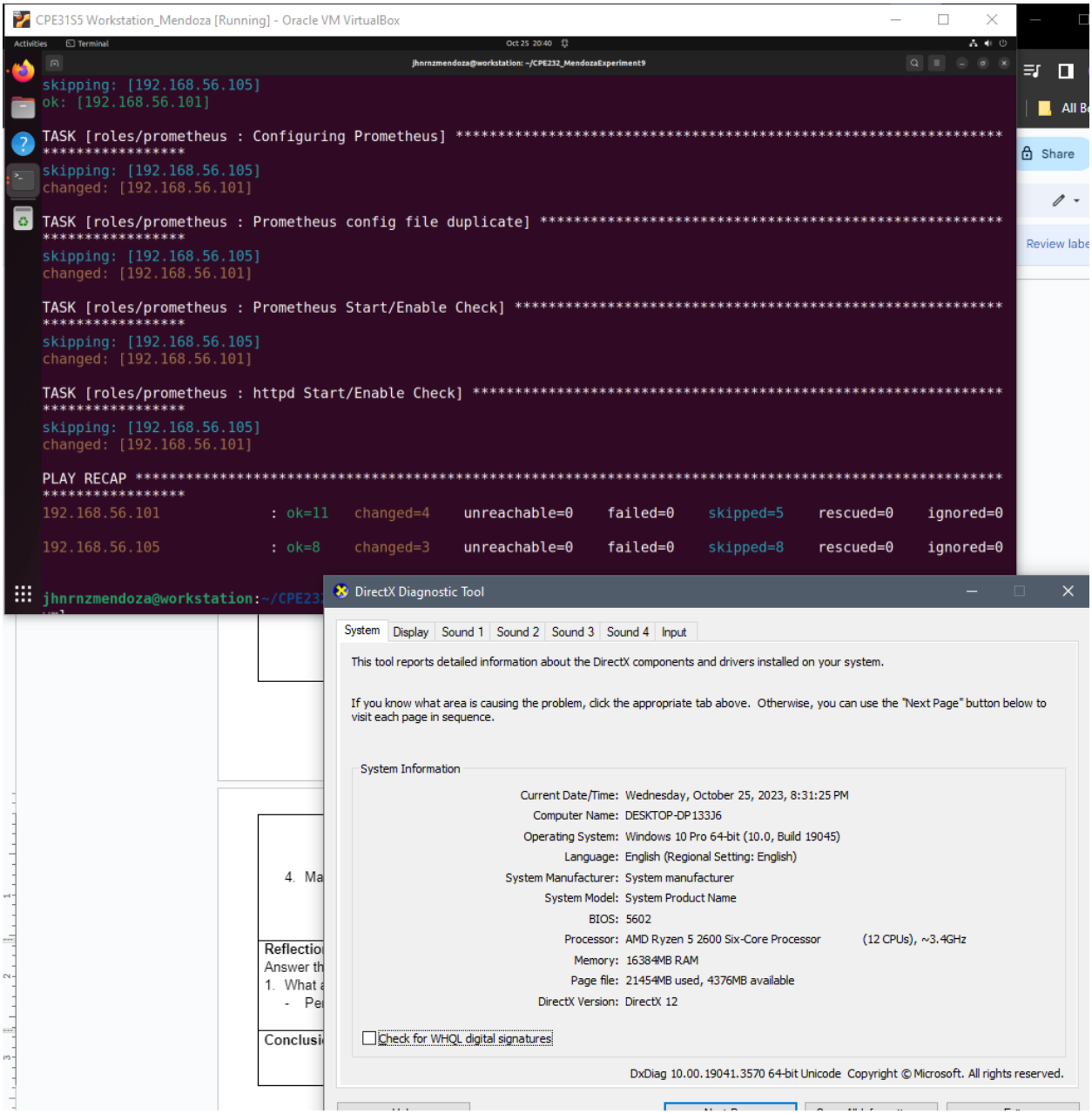
Output: Running the playbook `installation.yml` (Part 3)



Observation:

- Continuation of the output of the playbook. We can observe how each task was executed, and was able to perform the desired processes.

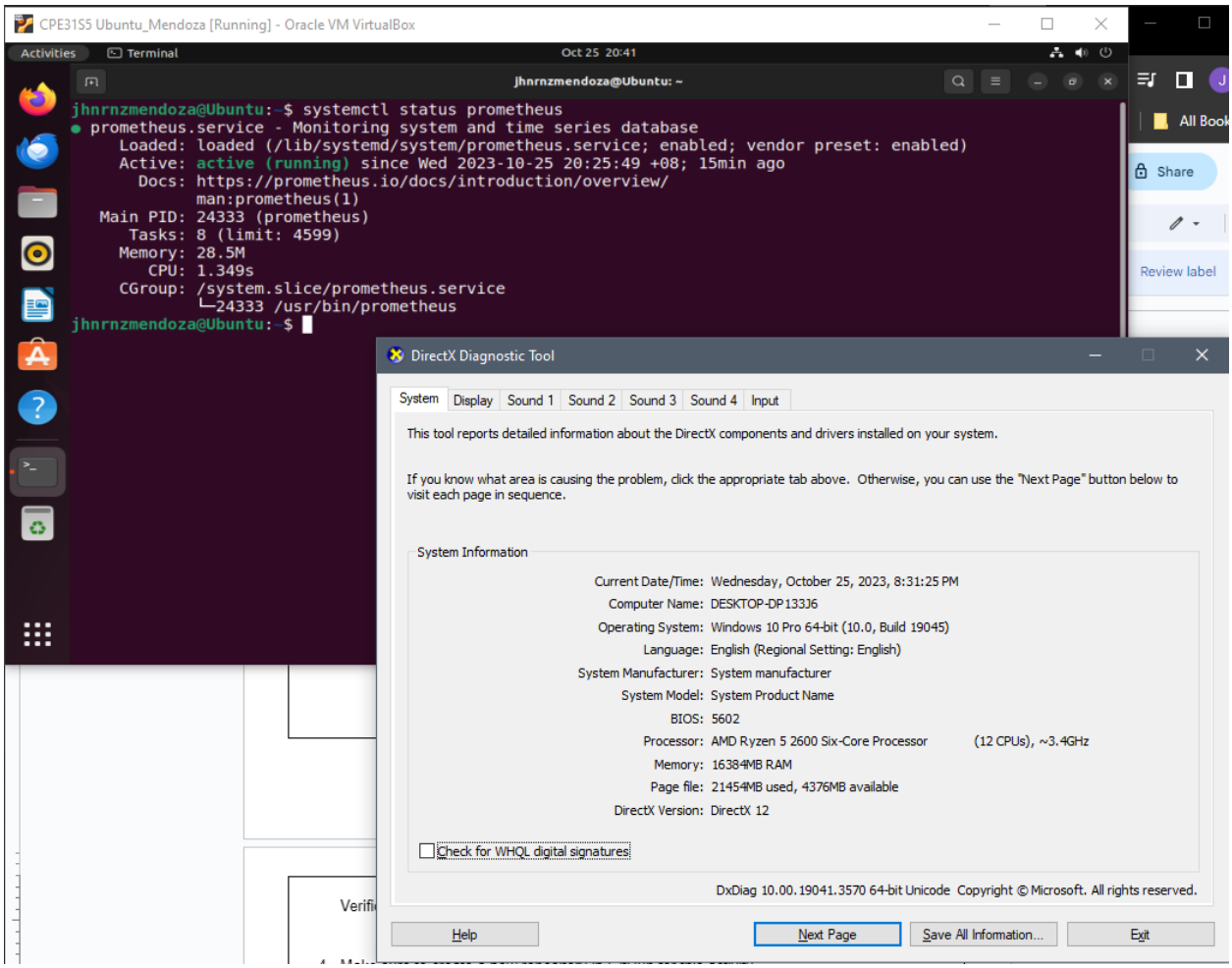
Output: Running the playbook `installation.yml` (Part 4)



Observation:

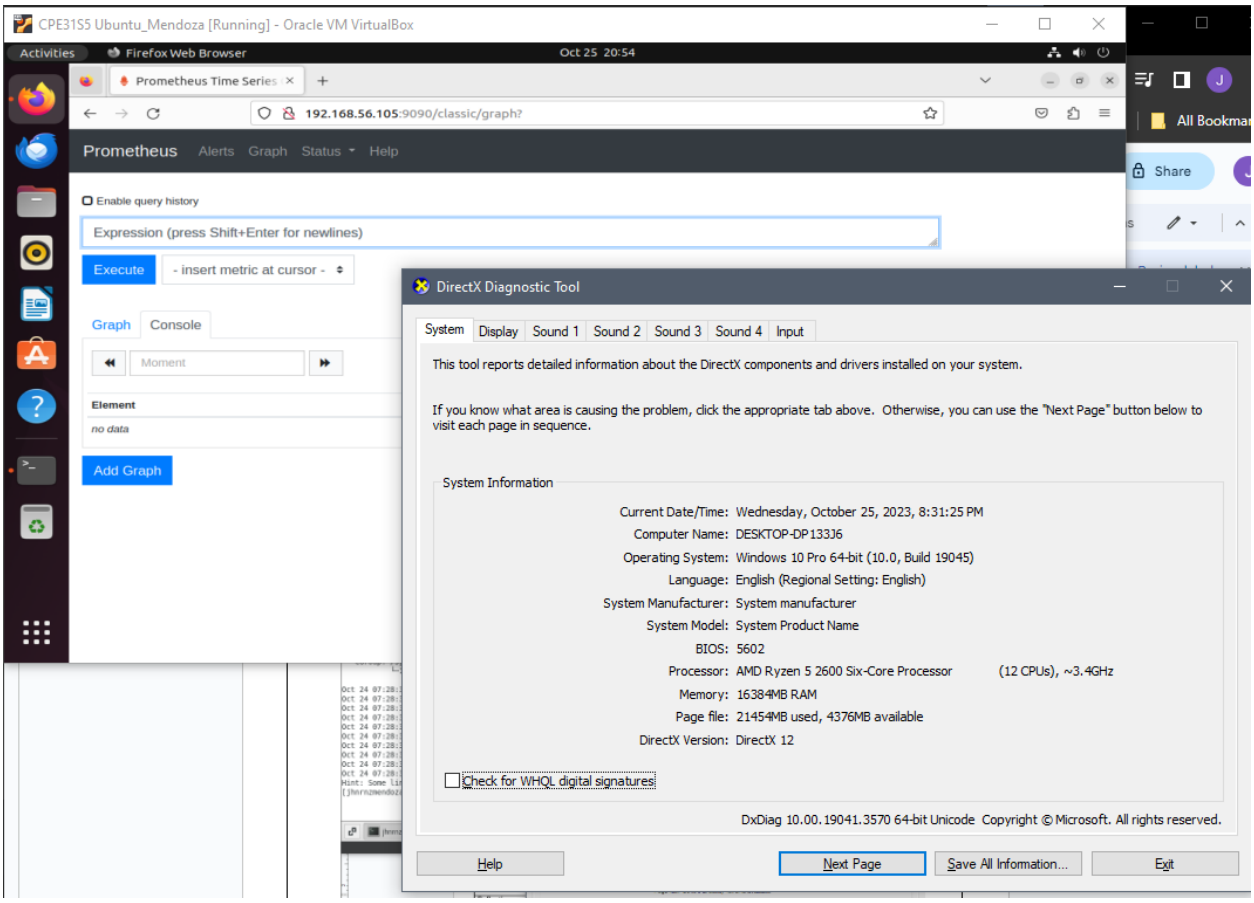
- Finally, we can observe the summary of the accomplished playbook run. Both of the servers has been configured and has no errors.

Verification: Ubuntu Remote Server



Observation:

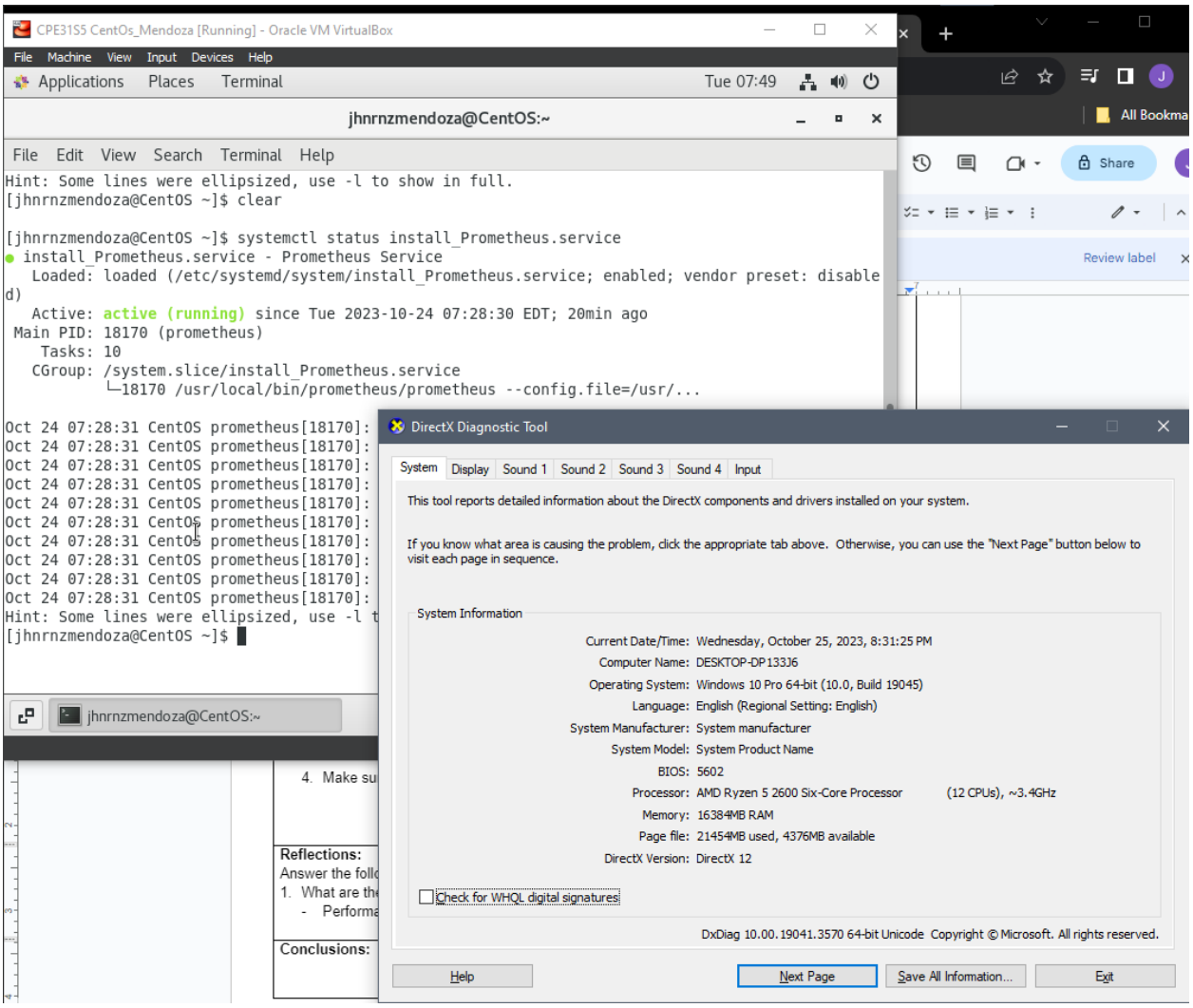
- Using the `systemctl status` command, we are able to observe that the `prometheus` service is successfully installed and is running.



Observation:

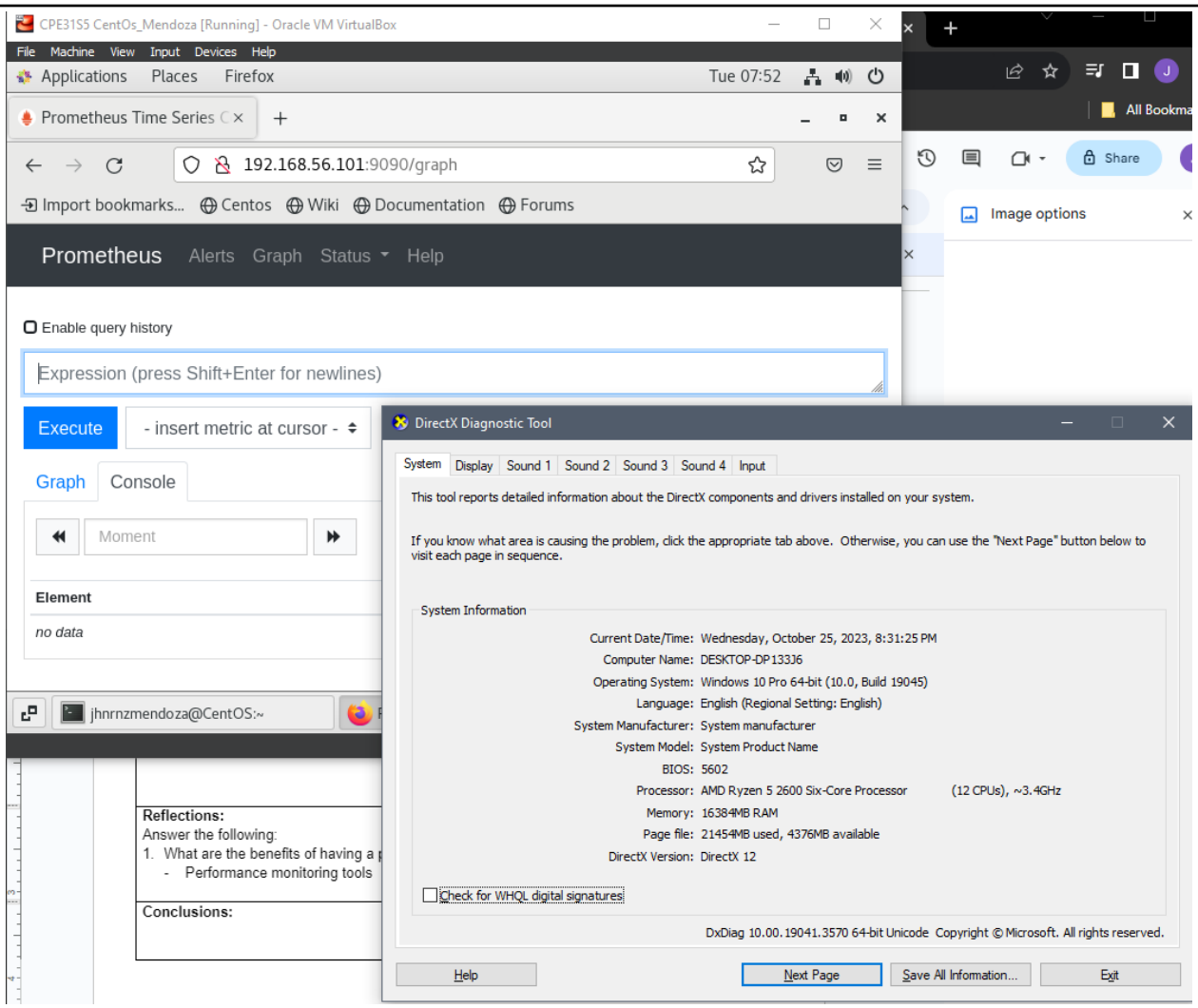
- By going to the web browser and inputting the server's IP together with the port number of Prometheus which is `9090`. We are able to successfully observe the Prometheus GUI browser.

Verification: CentOS Remote Server



Observation:

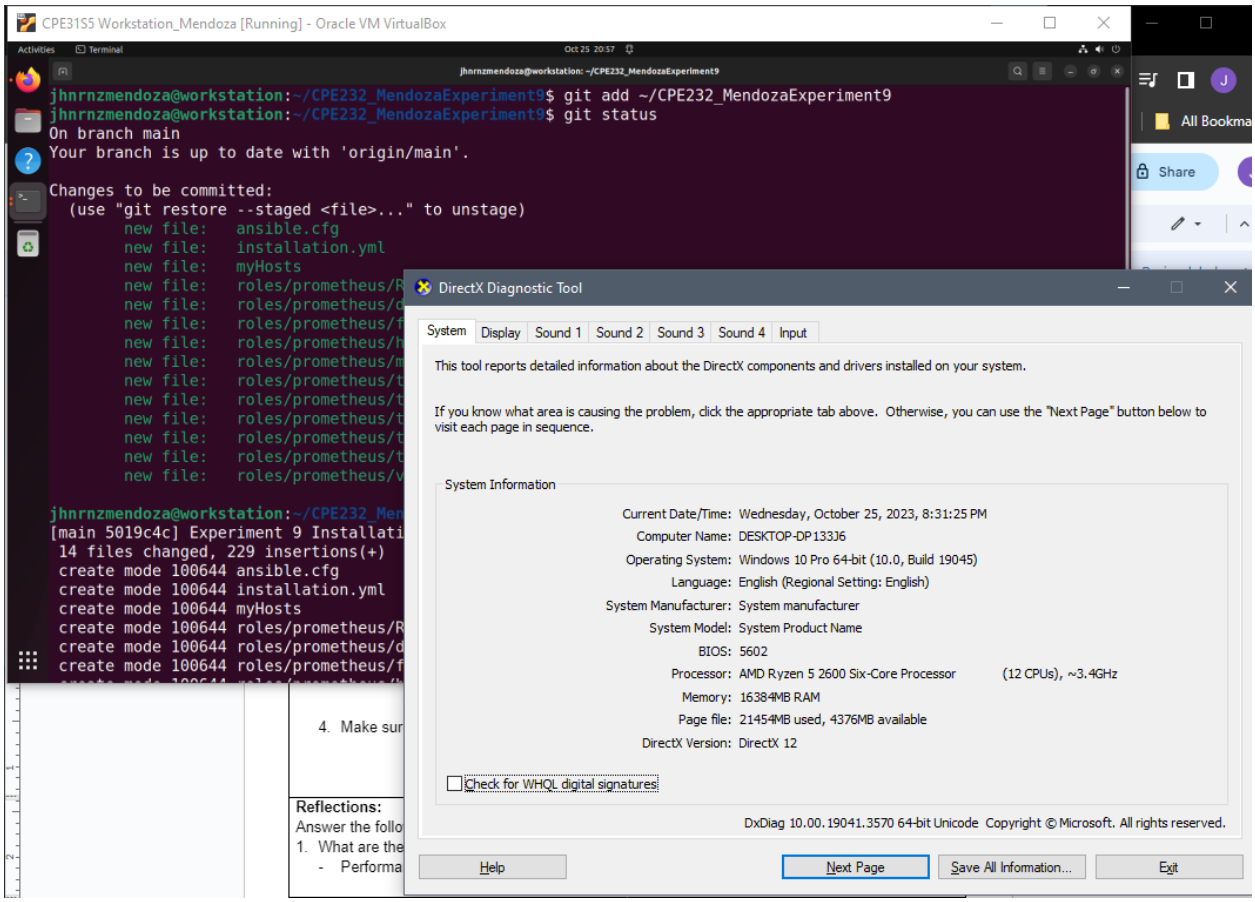
- Using the `systemctl status` command, we are able to observe that the `prometheus` service is successfully installed and is running. Although, we might have to start the service manually if we have encountered a few errors such as restarting the kernel using the s`ystemctl daemon-reload`.



Observation:

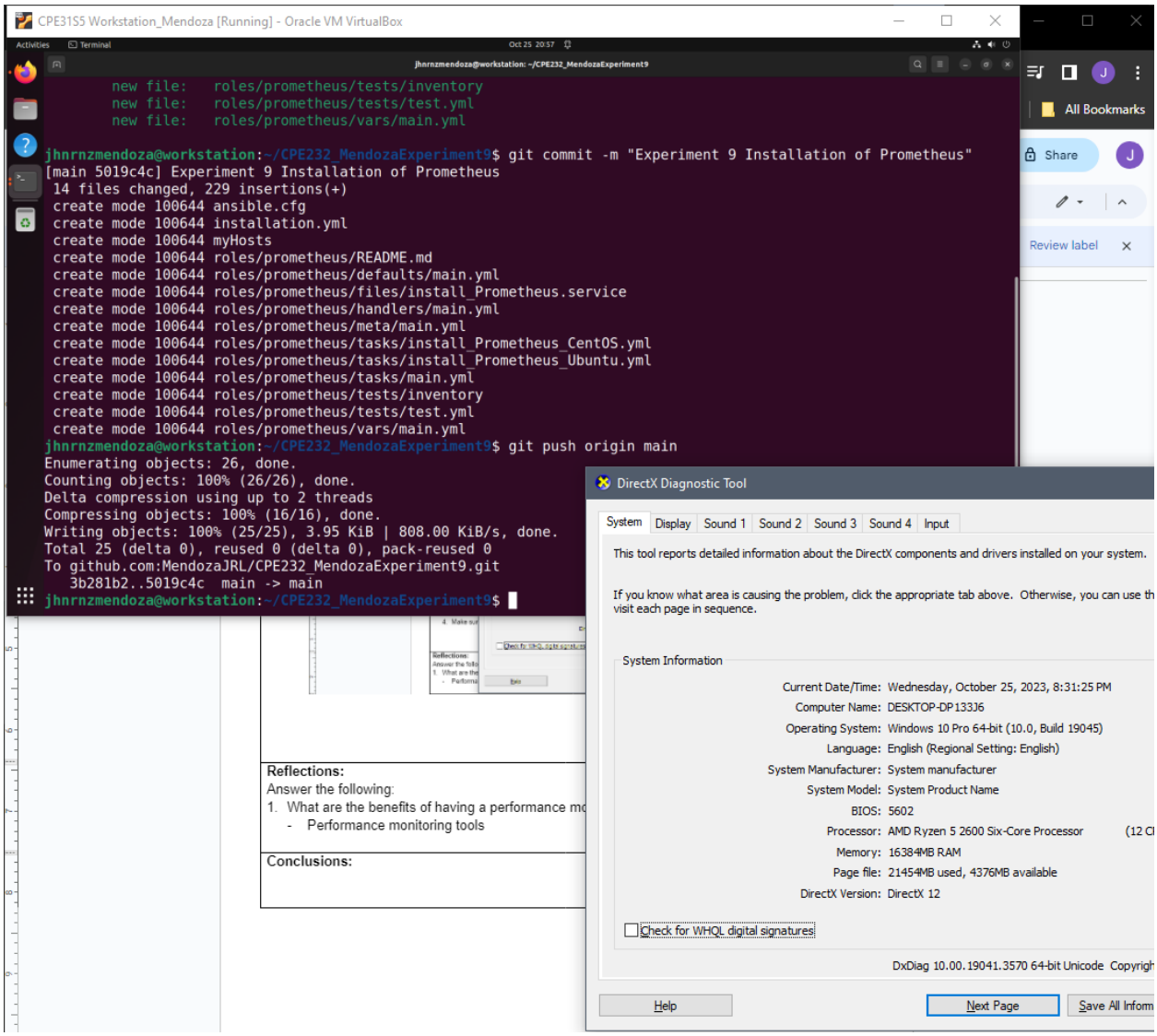
- By going to the web browser and inputting the server's IP together with the port number of Prometheus which is `9090`. We are able to successfully observe the Prometheus GUI browser.

4. Make sure to create a new repository in GitHub for this activity.



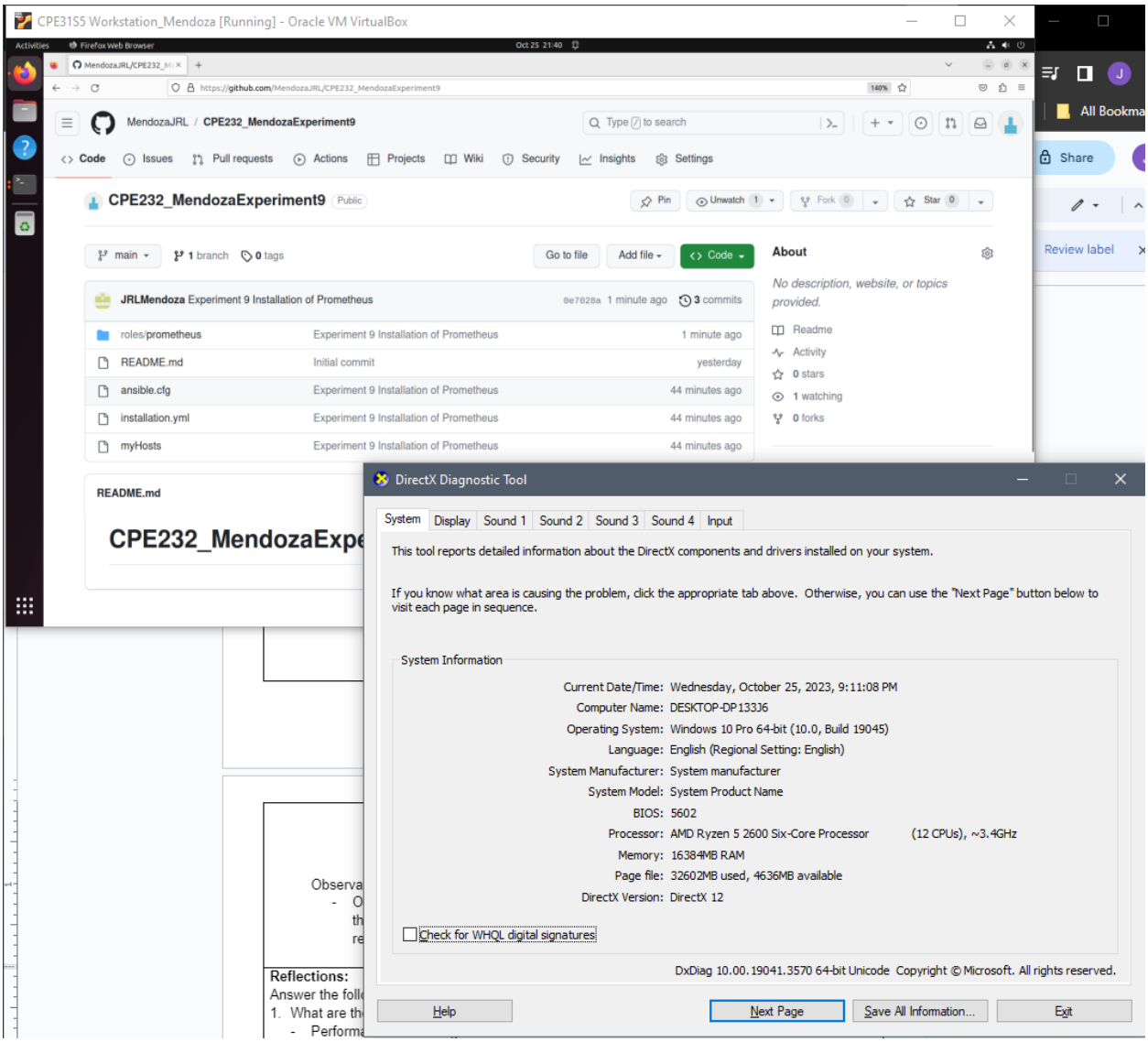
Observation:

- Using the command `git add` to add the target directory to be pushed to the cloud GitHub repository. The command `git status`, is to show the changes we have done to the files.



Observation:

- The command git commit and git push is to push the changes to the cloud repository.



Observation:

- On the cloud GitHub repository, we can observe that the changes were pushed and that the directories and files that we have created were present to the online repository.

GitHub Link: https://github.com/MendozaJRL/CPE232_MendozaExperiment9.git

Reflections:

Answer the following:

1. What are the benefits of having a performance monitoring tool?
    - Performance monitoring tools are essential in order to monitor the current status of our system and even remote servers. Tools such as prometheus, will provide us information about the device's system consumption, memory, and others in a time-stamped format. This information would help the system administrator to be notified on the current status of the remote servers and perform necessary precautions or solutions in order to avoid or solve them. By having performance monitoring tools, the system would prevent having issues and errors which will yield to a better uptime of the system.

Conclusions:

In this experiment, the students have been introduced to performance monitoring tools, specifically prometheus. Performance monitoring tools help the system administrator identify the issues or errors a device would have, and allow the system administrator to fix them ahead of time to avoid downtime. The students were able to implement this performance monitoring tool using the previously learned concepts about ansible playbooks, ansible roles, and git commands. The students were also able to understand how the ansible playbook and ansible roles work since this experiment provides the students how ansible can be used in real applications. Moreover, the ansible roles simple makes the playbook execution much organized in a way that the tasks needed are just loaded to the main playbook file when needed. Furthermore, the students would be able to use the understood concepts covered on this experiment such as the essence of performance monitoring tools in the future experiments