Name: John Renzo L. Mendoza	**Date Performed:** September 28, 2023
Course/Section: CPE31S5	**Date Submitted:** October 01, 2023
Instructor: Engr. Roman Richard	**Semester and SY:** First Semester, 2023- 2024

Prelim Skills Examination

Tools Needed:

1. Control Node (CN) - 1
2. Manage Node (MN) - 1 Ubuntu
3. Manage Node (MN) - 1 CentOS

Procedure:

1. Note: You are required to create a document report of the steps you will do for this exam. All screenshots should be labeled and explained properly.

2. Create a repository in your GitHub account and label it as Surname_PrelimExam

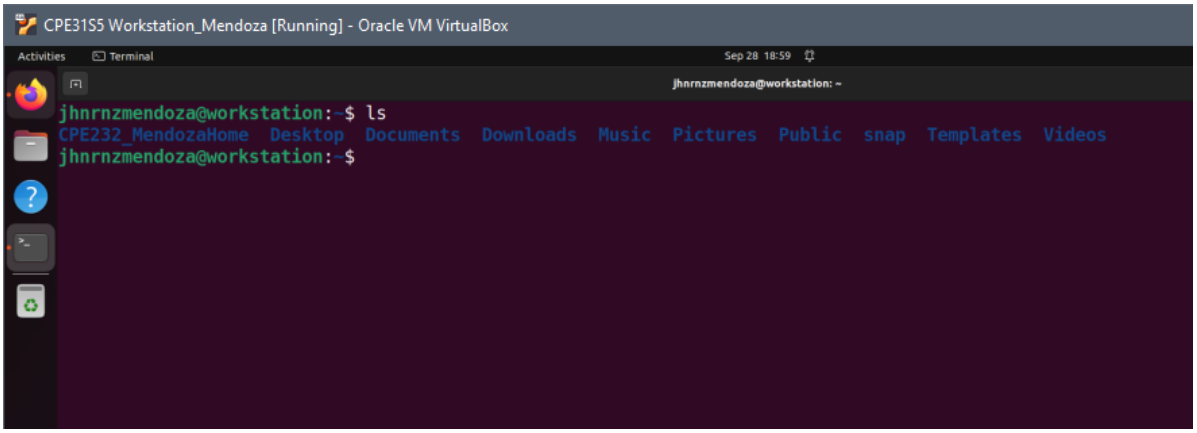

Image 2.1. Initial screenshot of the home directory contents.

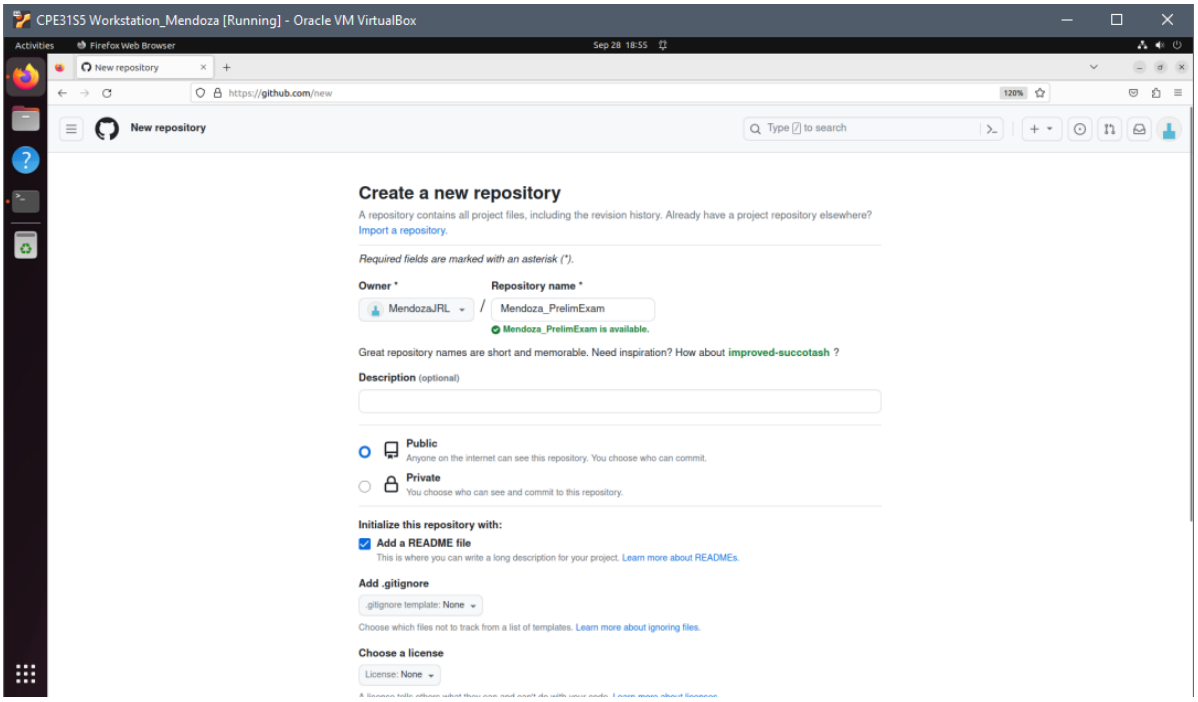

Image 2.2. Creation of a new repository on GitHub with the proper naming convention

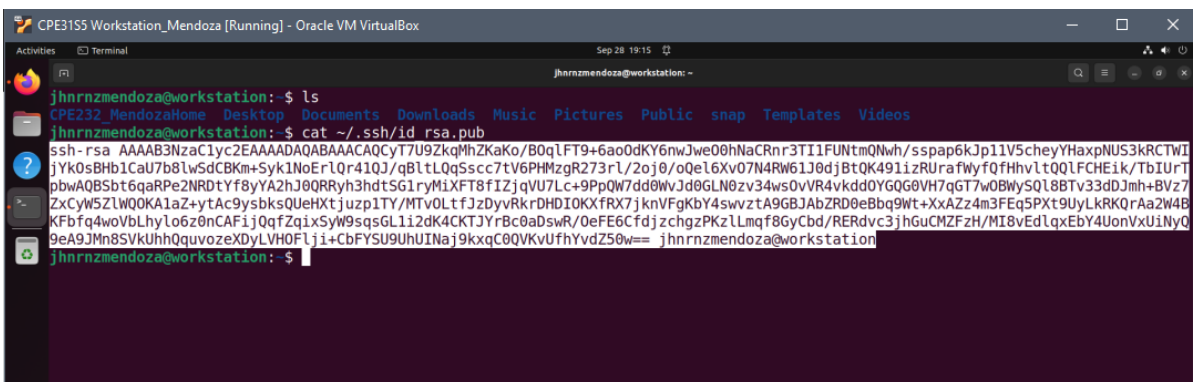

Image 2.3. Optional. Copying the public key generated on the control node.

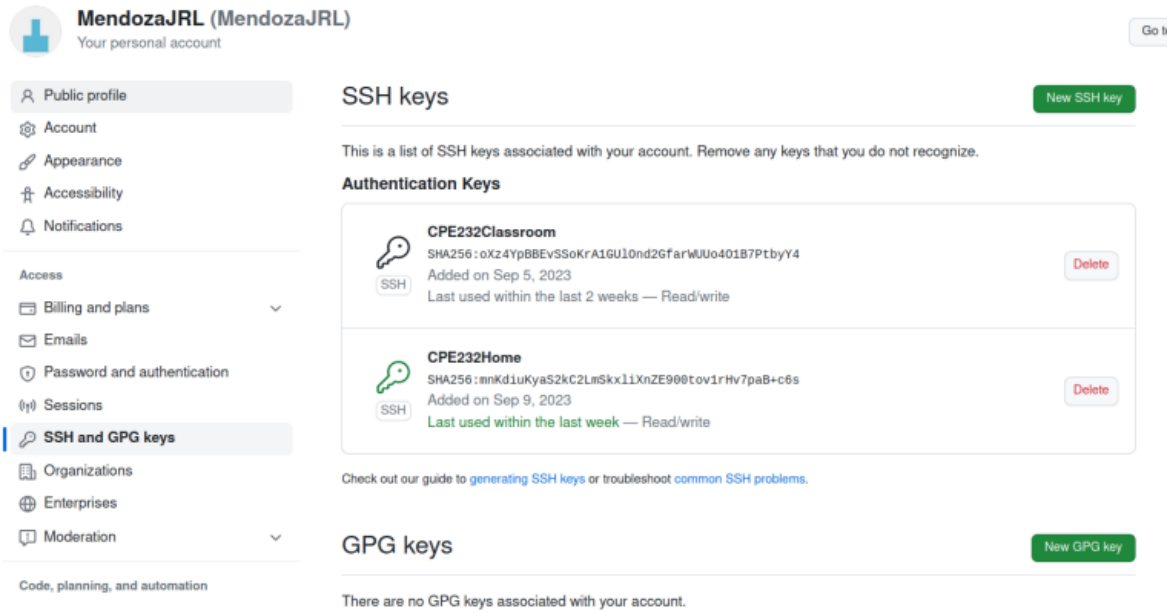

Image 2.4. Optional. Creation of SSH Key on GitHub, and pasting the copied generated key.

Observation:

Assuming an SSH key was already generated and the managed nodes are already accessible by the control node through SSH. The Images 2.3 and 2.4 are optional because my Virtual Machines already use an ssh key which is named as "`CPE232Home`" which is also the same key that I used on previous experiments.

3. Clone your new repository in your CN.

```
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ git --version
git version 2.34.1
```

Image 3.1. Installed `git` command on the local terminal of the control node.

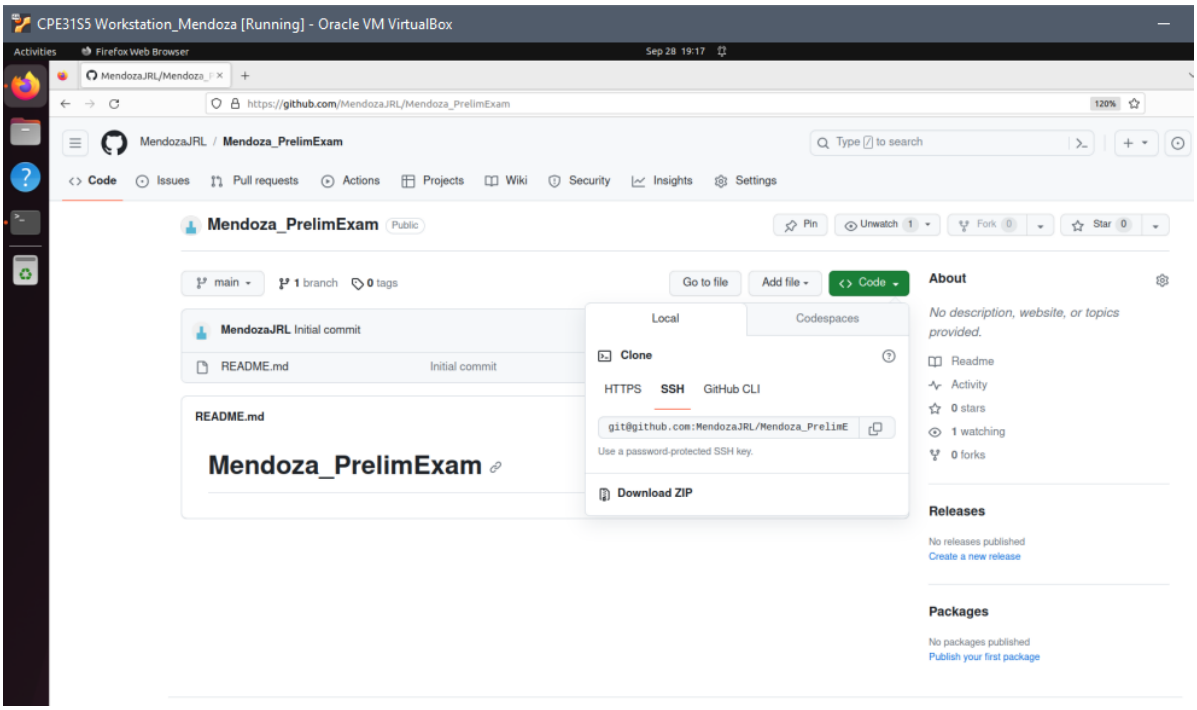

Image 3.2. Copying the SSH link on GitHub repository to use the repository on the control node.

```
jhnrnzmendoza@workstation:~$ git clone git@github.com:MendozaJRL/Mendoza_PrelimExam.git
Cloning into 'Mendoza_PrelimExam'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

Image 3.3. Using `git clone` command to clone the cloud repository on the local terminal.

```
jhnrnzmendoza@workstation:~$ ls
CPE232_MendozaHome  Desktop  Documents  Downloads  Mendoza_PrelimExam  Music  Pictures  Public  snap  Templates  Videos
```

Image 3.4. Screenshot of the repository present in the home directory of the control node.

Observation:

In this procedure, I have used the `git clone` command to clone the cloud repository to the local terminal of the control node. To make this command work, the prerequisite is that you must have installed `Git` on the terminal first. I have not shown the installation of `Git` as it has been already installed as seen on the screenshots.

4. In your CN, create an inventory file and ansible.cfg files.

```
jhnrnzmendoza@workstation:~$ cd Mendoza_PrelimExam
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ ls
README.md
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ sudo nano myNodes
[sudo] password for jhnrnzmendoza:
```

Image 4.1. Changing to the repository directory, and creating the inventory file "`myNodes`".

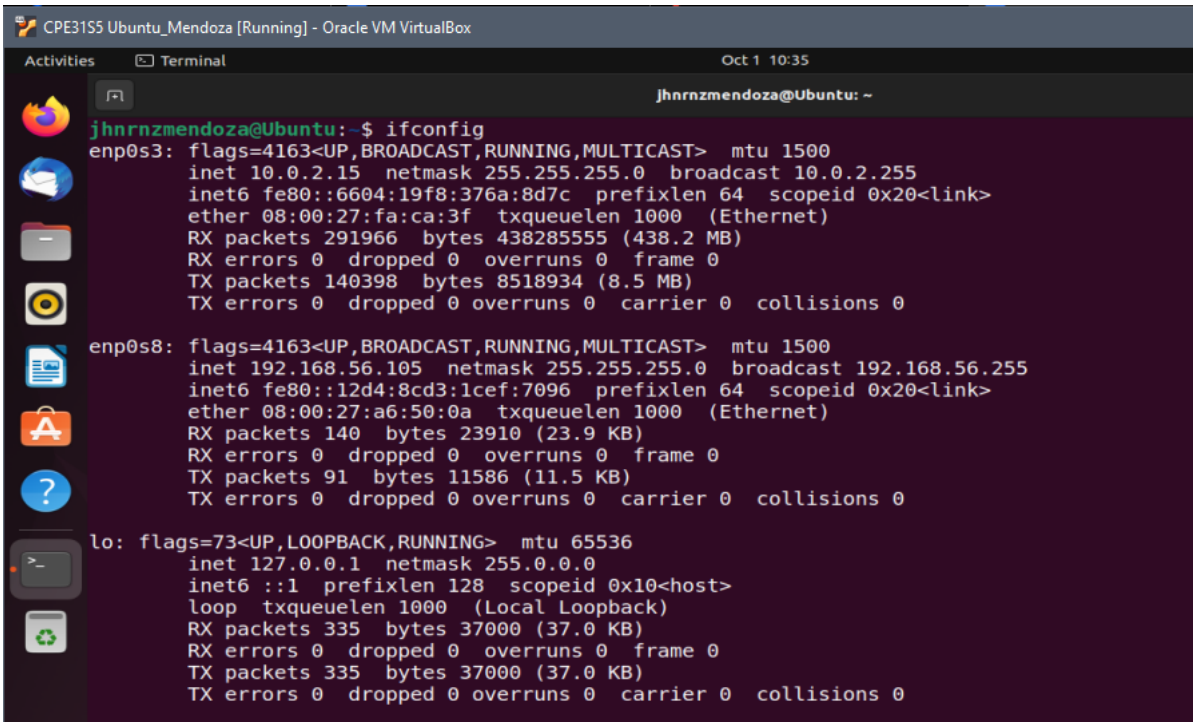

Image 4.2. Issuing the `ifconfig` command to check the IP address Ubuntu managed node.

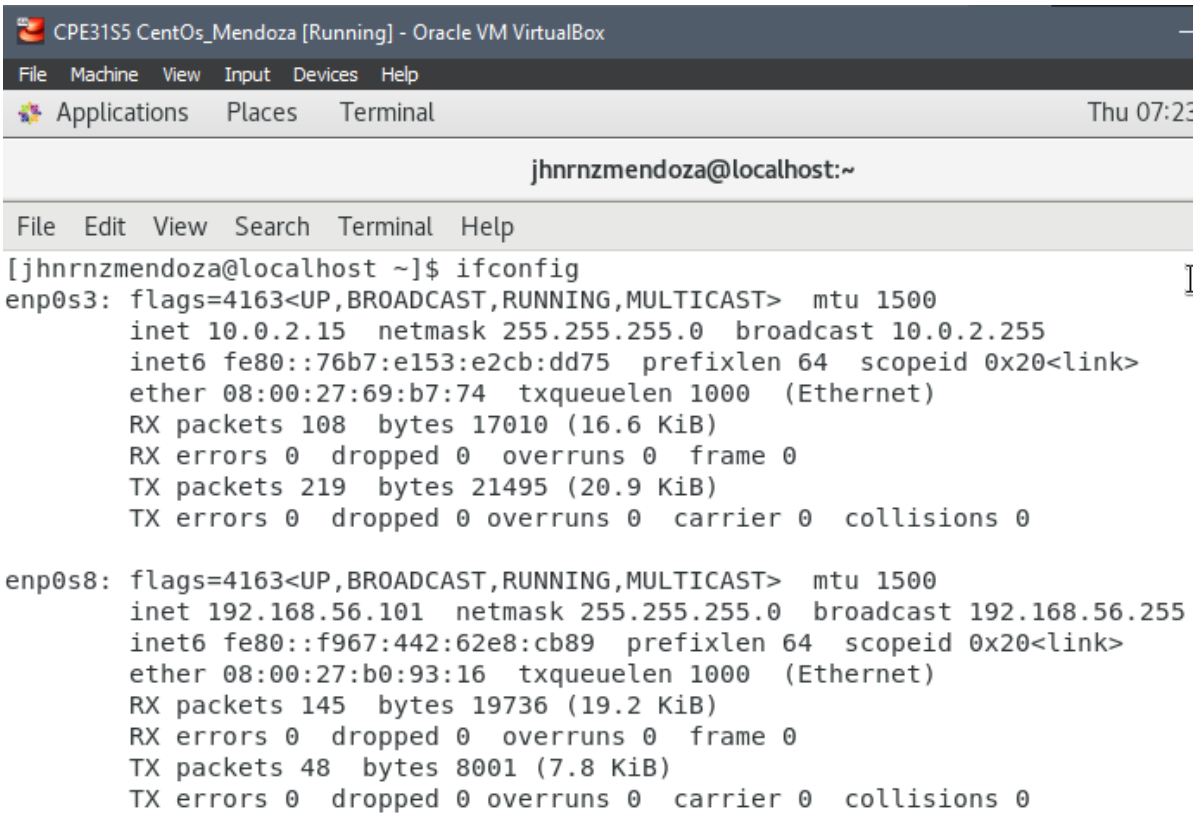

Image 4.3. Running the `ifconfig` command to check the IP address CentOS managed node.

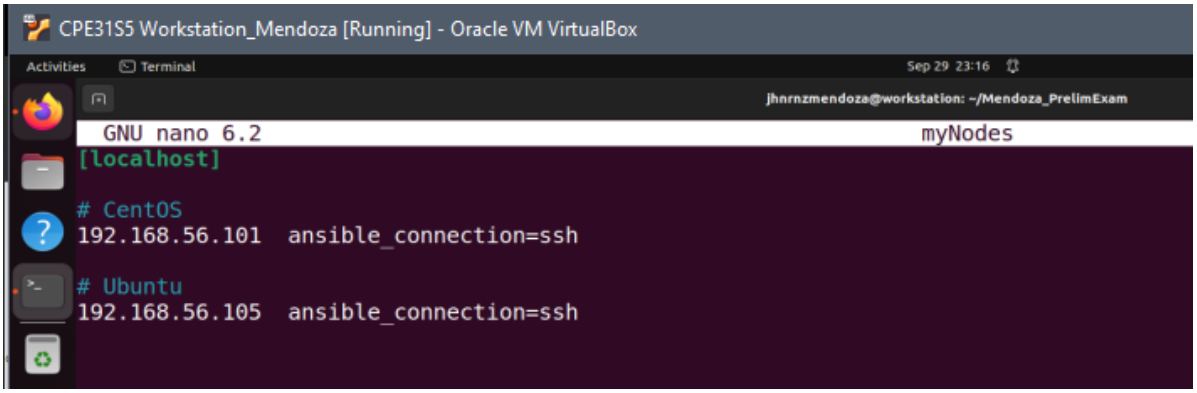

Image 4.4. Adding the IP addresses of the managed nodes to the inventory file.

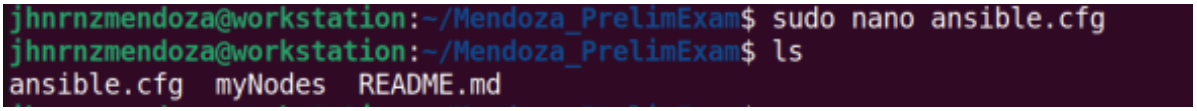

Image 4.5. Creating the `ansible.cfg` file that stores the configurations for Ansible.

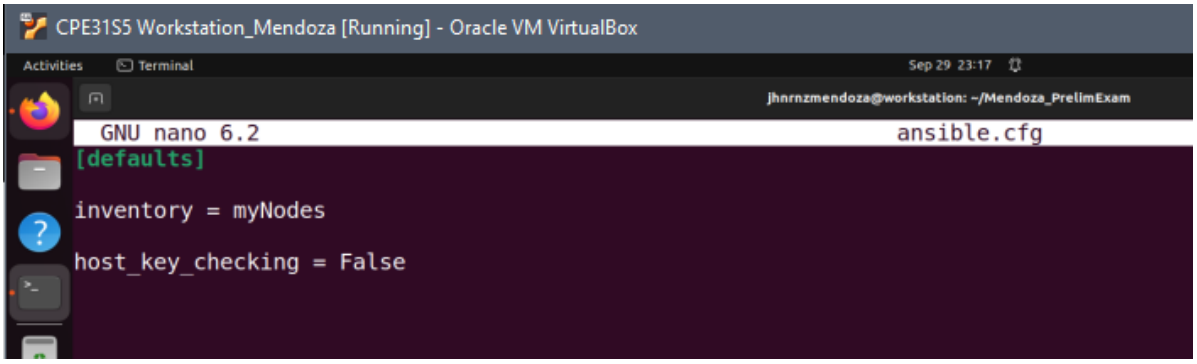

Image 4.6. Contents of the `ansible.cfg` file.

Observation:

In creating the inventory file and `ansible.cfg` file, I have used the `ansible_connection=ssh` since both of the managed nodes are accessed through `ssh` connection. The `myNodes` file will be the parsed inventory for ansible as defined in our `ansible.cfg`.

5. Create an Ansible playbook that does the following with an input of a config.yaml file for both Manage Nodes.

```
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ sudo nano config.yaml
```

Image 5.1. Creation of the playbook file.

- Installs the latest python3 and pip3

```
#Installation Tasks
- name: Installing python3, pip3, and Java open-jdk
  package:
    name:
      - "{{ python_package }}"
      - "{{ pip_package }}"
      - "{{ java_package }}"
    state: latest
    update_cache: yes
```

Image 5.1. Task to install the latest `python3`, `pip3`, and `Java open-jdk`.

```
python_package=python3          pip_package=python3-pip
```

Image 5.2. Specific variables defined on the inventory file for this task.

Observation:

Instead of manually having the task separately, I have made the playbook generalized wherein the system administrator would only need to change the variables on the inventory file. Since these variables such as `python_package`, `pip_package`, and `java_package` are defined on the inventory file.

- use pip3 as default pip

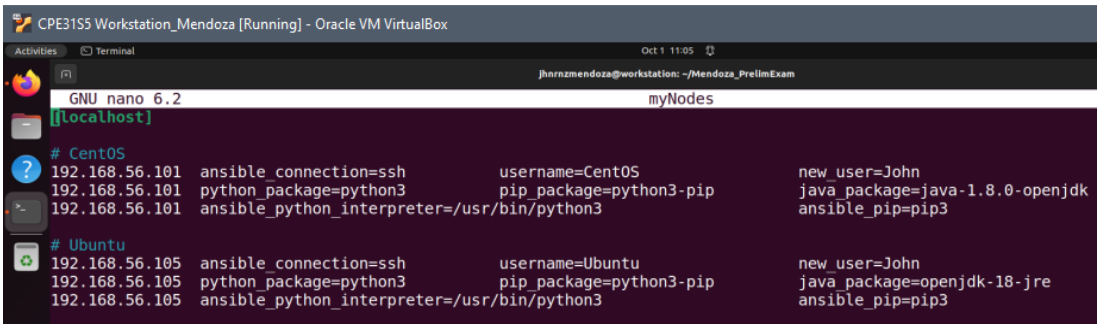

Image 5.3. `Pip3` as default pip defined on inventory file.

Observation:

The specific configuration for this task is on the variable `ansible_pip=pip3`.

- use python3 as default python

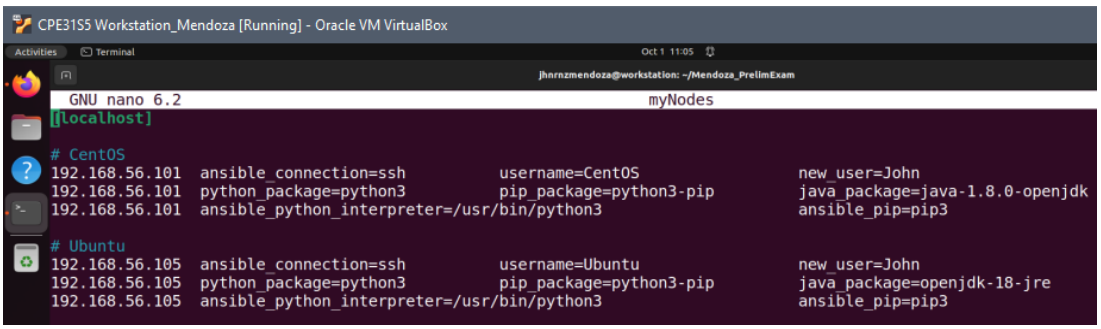

Image 5.4. `Python3` as default python defined on inventory file.

Observation:

The specific configuration for this task is `ansible_python_interpreter=/usr/bin/python3` because this path is the location of the installed python.

- Install Java open-jdk

```
#Installation Tasks
- name: Installing python3, pip3, and Java open-jdk
  package:
    name:
      - "{{ python_package }}"
      - "{{ java_package }}"
    state: latest
    update_cache: yes
```

Image 5.5. Task to install the latest `python3`, `pip3`, and `Java open-jdk`.

```
java_package=java-1.8.0-openjdk
```

Image 5.6. Specific variables defined on the inventory file for `CentOs`.

```
java_package=openjdk-18-jre
```

Image 5.7. Specific variables defined on the inventory file for `Ubuntu`.

Observation:

Similarly, the installation for `Java` has its variables assigned on the inventory file as shown on Images 5.6 and 5.7. There are differences on the package since the managed nodes have different distributions. These are the appropriate commands for both.

- Create Motd containing the text defined by a variable defined in `config.yaml` file and if there is no variable input the default motd is "Ansible Managed node by (your user name)"

```
#Message of the Day
- name: Constructing Message of the Day
  shell: 'echo "Message of the Day"; echo "Ansible Managed Node by {{ username }} "'
  register: motd

- debug:
    var: motd.stdout_lines
```

Image 5.8. Task for creating a message of the day.

Observation:

Using the `shell` command, we are able to run terminal queries. The `register` command is responsible for capturing the output of the previous `shell` command. I have used the variable `motd` to store the output. Lastly, the `debug` command is used to display the output of the command, but since we have stored it on the `motd` variable. I have used the `.stdout_lines` to display the contents.

Essentially, since our inventory file "myNodes", passes a variable which is the user on the managed node, the content of that variable is then used instead of the defaults.

```
TASK [Constructing Message of the Day] ***********************************************
changed: [192.168.56.105]
changed: [192.168.56.101]

TASK [debug] *************************************************************************
ok: [192.168.56.101] => {
    "motd.stdout_lines": [
        "Message of the Day",
        "Ansible Managed Node by CentOS "
    ]
}
ok: [192.168.56.105] => {
    "motd.stdout_lines": [
        "Message of the Day",
        "Ansible Managed Node by Ubuntu "
    ]
}
```

Image 5.9. Part of the Playbook's output, specifically MOTD output.

```
username=CentOS
```

```
username=Ubuntu
```

Observation:

Since we have defined the variable username on the inventory files, the playbook will use these variables as an input.

- Create a user with a variable defined in config.yaml

```
#User Creation
- name: Creating a user
  shell: 'sudo useradd {{ new_user }} | cat /etc/passwd | grep {{ new_user }}'
  register: output

- debug:
    var: output.stdout_lines
```

Image 5.10. Task for creating a new user.

```
new_user=John
```

Observation:

In this task, I have used the `shell` command again to run a query. The query involves the `useradd` command that adds a new user to the `/etc/passwd` file. I have used a `pipe` to filter the output, specifically yung `grep` to find a pattern on the entries having the `new_user` added.

The task also takes a defined variable. I have defined the variable on the inventory file as `new_user=John`, which means that the created user should be "`John`" for both managed nodes.

The `register` and `debug` commands are used to simply display the output on the playbook's execution.

```
TASK [Creating a user] ***********************************************************
changed: [192.168.56.105]
changed: [192.168.56.101]

TASK [debug] *********************************************************************
ok: [192.168.56.101] => {
    "output.stdout_lines": [
        "jhnrnzmendoza:x:1000:1000:John Renzo L. Mendoza:/home/jhnrnzmendoza:/bin/bash",
        "John:x:1001:1001::/home/John:/bin/bash"
    ]
}
ok: [192.168.56.105] => {
    "output.stdout_lines": [
        "John:x:1001:1001::/home/John:/bin/sh"
    ]
}
```

Image 5.10. Part of the Playbook's output, specifically user creation output.

Inventory File

Edited the Inventory file by including the needed variables to be used by the playbook.

```
jhnrnzmendoza@workstation: ~/Mendoza_PrelimExam
GNU nano 6.2                                                    myNodes
[localhost]

# CentOS
192.168.56.101  ansible_connection=ssh          username=CentOS                 new_user=John
192.168.56.101  python_package=python3          pip_package=python3-pip         java_package=java-1.8.0-openjdk
192.168.56.101  ansible_python_interpreter=/usr/bin/python3                     ansible_pip=pip3

# Ubuntu
192.168.56.105  ansible_connection=ssh          username=Ubuntu                 new_user=John
192.168.56.105  python_package=python3          pip_package=python3-pip         java_package=openjdk-18-jre
192.168.56.105  ansible_python_interpreter=/usr/bin/python3                     ansible_pip=pip3
```

Image 5.11. Overall configurations on the inventory file.

Executing the Playbook

```
CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox
Activities   Terminal                                   Sep 29 23:21
jhnrnzmendoza@workstation: ~/Mendoza_PrelimExam
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ ansible-playbook --ask-become-pass config.yaml
BECOME password:

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

TASK [Gathering Facts] *********************************************************************
ok: [192.168.56.105]
ok: [192.168.56.101]

TASK [Constructing Message of the Day] *****************************************************
changed: [192.168.56.105]
changed: [192.168.56.101]

TASK [debug] *******************************************************************************
ok: [192.168.56.105] => {
    "motd.stdout_lines": [
        "Message of the Day",
        "Ansible Managed Node by Ubuntu "
    ]
}
ok: [192.168.56.101] => {
    "motd.stdout_lines": [
        "Message of the Day",
        "Ansible Managed Node by CentOS "
    ]
}

TASK [Installing python3, pip3, and Java open-jdk] *****************************************
```

Image 5.12. Part 1. Playbook Output.

```
TASK [Installing python3, pip3, and Java open-jdk] ***********************************************************
ok: [192.168.56.105]
ok: [192.168.56.101]

TASK [Creating a user] ***************************************************************************************
changed: [192.168.56.105]
changed: [192.168.56.101]

TASK [debug] *************************************************************************************************
ok: [192.168.56.101] => {
    "output.stdout_lines": [
        "jhnrnzmendoza:x:1000:1000:John Renzo L. Mendoza:/home/jhnrnzmendoza:/bin/bash",
        "John:x:1001:1001::/home/John:/bin/bash"
    ]
}
ok: [192.168.56.105] => {
    "output.stdout_lines": [
        "John:x:1001:1001::/home/John:/bin/sh"
    ]
}

PLAY RECAP ***************************************************************************************************
192.168.56.101             : ok=6    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.105             : ok=6    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Image 5.13. Part 2. Playbook Output.

Checking the Ubuntu and CentOS Virtual Machines if Successful.

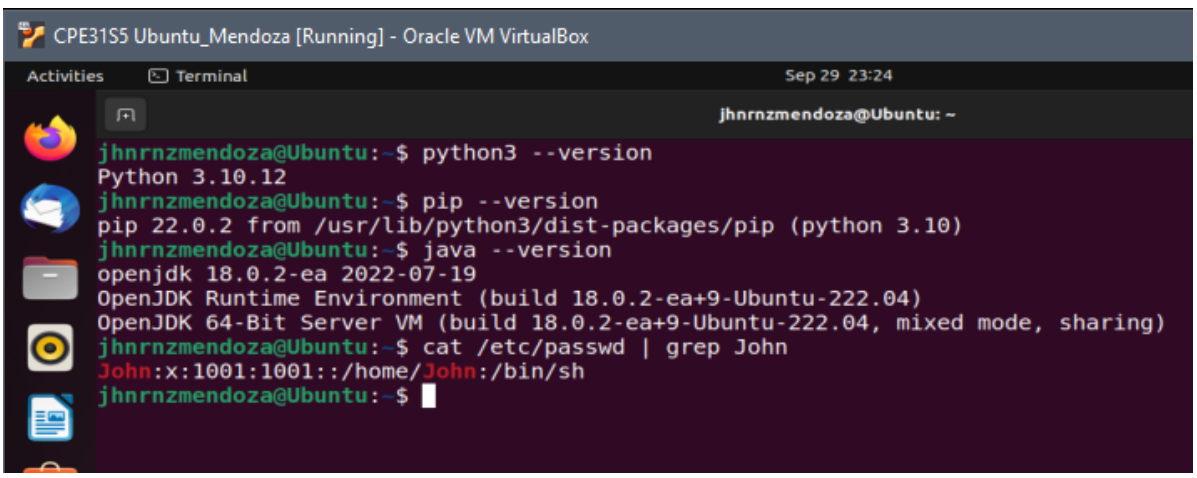

Image 5.14. Checking if python3, pip3, and java open-jdk was installed as well as the creation of the new user “John” on Ubuntu.

```
CPE31S5 CentOs_Mendoza [Running] - Oracle VM VirtualBox
File  Machine  View  Input  Devices  Help
Applications  Places  Terminal                                   Fri 09:58
                          jhnrnzmendoza@localhost:~
File  Edit  View  Search  Terminal  Help
[jhnrnzmendoza@localhost ~]$ python3 --version
Python 3.6.8
[jhnrnzmendoza@localhost ~]$ pip3 --version
pip 9.0.3 from /usr/lib/python3.6/site-packages (python 3.6)
[jhnrnzmendoza@localhost ~]$ java -version
openjdk version "1.8.0_382"
OpenJDK Runtime Environment (build 1.8.0_382-b05)
OpenJDK 64-Bit Server VM (build 25.382-b05, mixed mode)
[jhnrnzmendoza@localhost ~]$ cat /etc/passwd | grep John
jhnrnzmendoza:x:1000:1000:John Renzo L. Mendoza:/home/jhnrnzmendoza:/bin/bash
John:x:1001:1001::/home/John:/bin/bash
[jhnrnzmendoza@localhost ~]$
```

Image 5.15. Checking if python3, pip3, and java open-jdk was installed as well as the creation of the new user "John" on CentOS.

Observation:

As we can observe on the managed nodes, the changes done from the playbook of the control node were successfully executed. The Images 5.14 and 5.15 shows the queries to test each task of the playbook if done successfully.

6. PUSH and COMMIT your PrelimExam in your GitHub repo

```
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ansible.cfg
        config.yaml
        myNodes

nothing added to commit but untracked files present (use "git add" to track)
```

Image 5.16. Using the `git status` command to initially check the changes.

```
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ git add ~/Mendoza_PrelimExam
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ 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:   config.yaml
        new file:   myNodes
```

Image 5.17. Using the `git add` command to add the changes to the GitHub repository.

```
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ git commit -m "CPE 232 Prelim Exam"
[main 5504204] CPE 232 Prelim Exam
 3 files changed, 44 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 config.yaml
 create mode 100644 myNodes
jhnrnzmendoza@workstation:~/Mendoza_PrelimExam$ git push origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 2 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 929 bytes | 929.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:MendozaJRL/Mendoza_PrelimExam.git
   ce6a317..5504204  main -> main
```

Image 5.18. Using the `git commit` and `git push` command to export the changes to the GitHub Repository

Observation:

Using the `git` commands, I am able to save my progress to the GitHub Repository and access it on the cloud.

The `git status` is used to check whether the local repository has new changes compared from the last version. The `git add` is used to add the initial changes on a queue to be committed or to be exported to the cloud repository. The `git commit` is used to commit the changes and add a tag on it as a label (optional). Lastly, the `git push` is used to push the changes to the cloud repository.

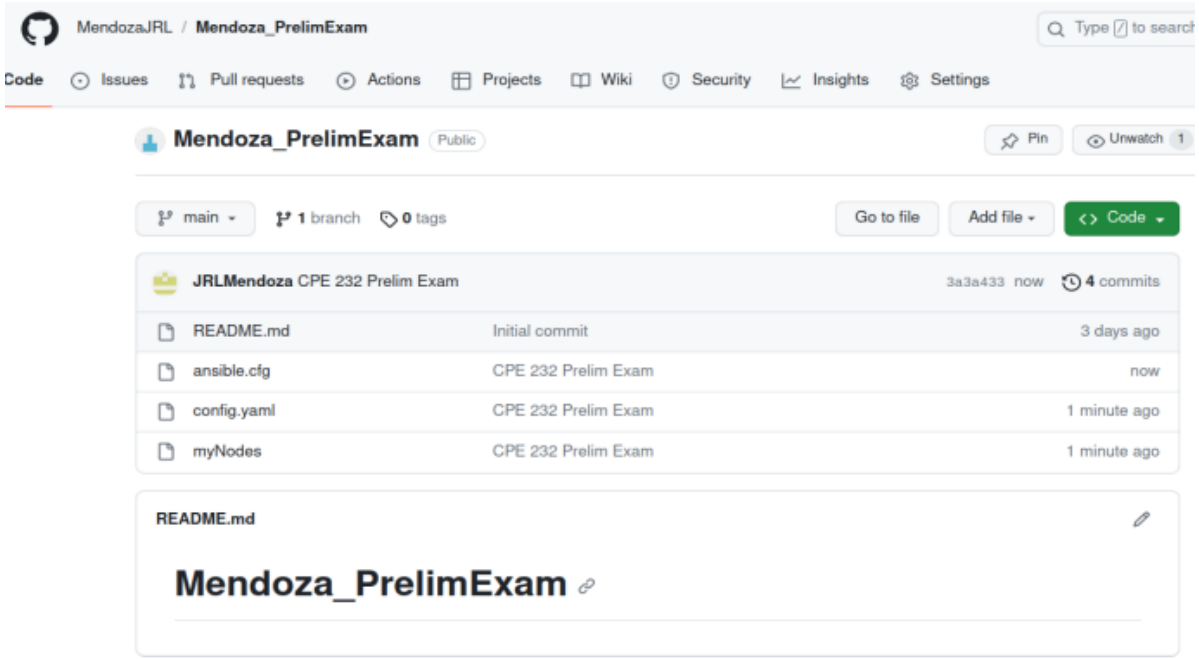

Image 5.19. Checking the changes added to the GitHub repository.

7. Your document report should be submitted here.

8. For your prelim exam to be counted, please paste your repository link here.
   https://github.com/MendozaJRL/Mendoza_PrelimExam.git