Name:Calderon Ricardo B.	**Date Performed:Mar 31, 2024**
Course/Section:CPE232 - CPE31S1	**Date Submitted:april 01, 2024**
Instructor: Dr. Jonathan Taylar	**Semester and SY: 2nd Sem / 2023-2024**

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 tool.

Prometheus

Prometheus fundamentally stores all data as timeseries: 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

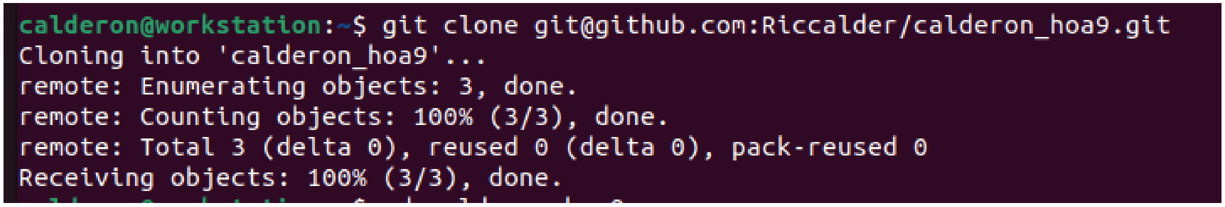
Cacti is a complete network graphing solution designed to harness the power of RRDTool's data storage and graphing functionality. Cacti provides a fast poller, advanced graph templating, multiple data acquisition methods, and user management features out of the box. All of this is wrapped in an intuitive, 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.

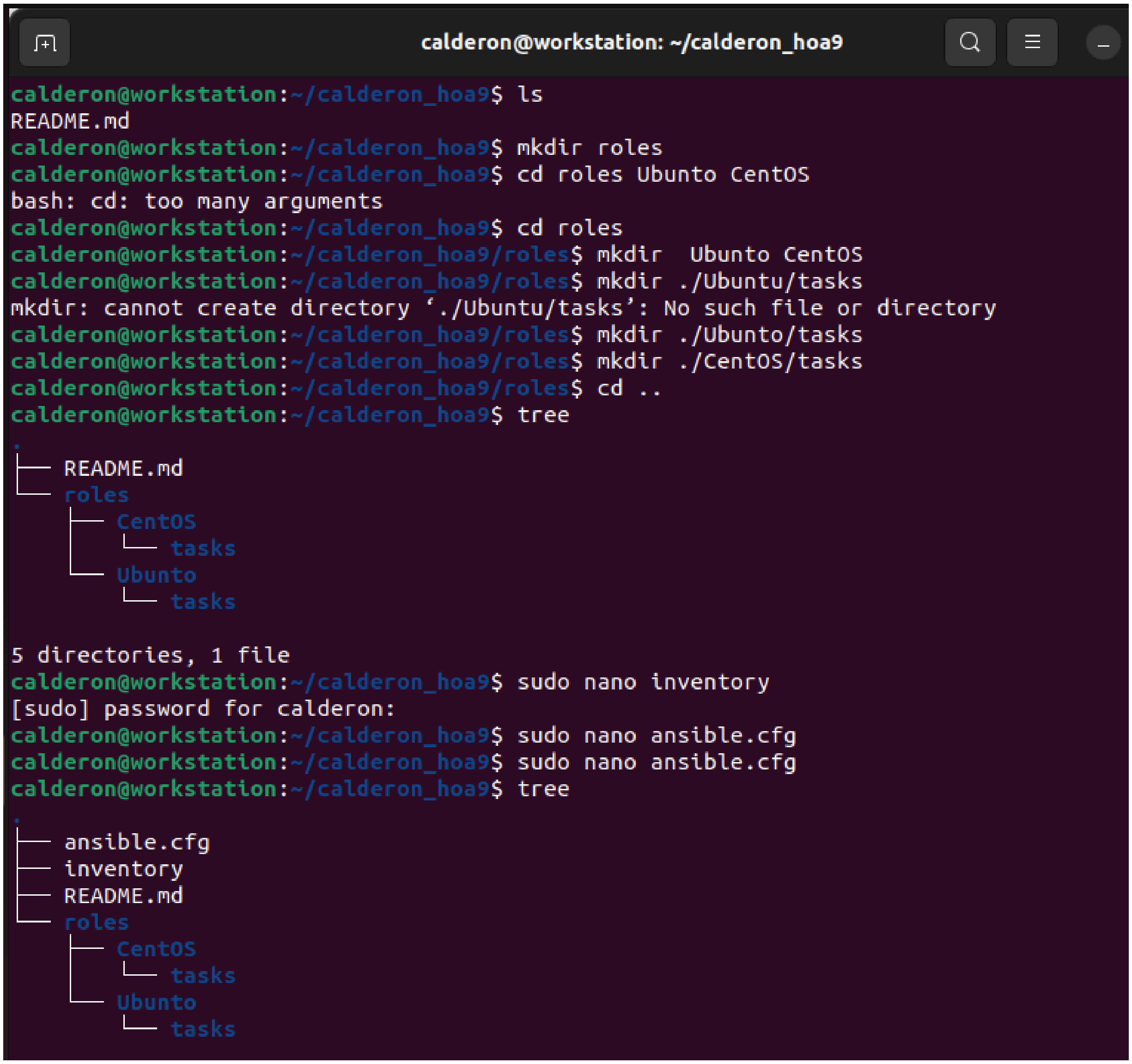
4. Output (screenshots and explanations)

This command clones a Git repository hosted on GitHub with the URL `git@github.com:Riccalder/calderon_hoa9.git` into a local directory named `calderon_hoa9.`



```
calderon@workstation:~$ git clone git@github.com:Riccalder/calderon_hoa9.git
Cloning into 'calderon_hoa9'...
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.
```

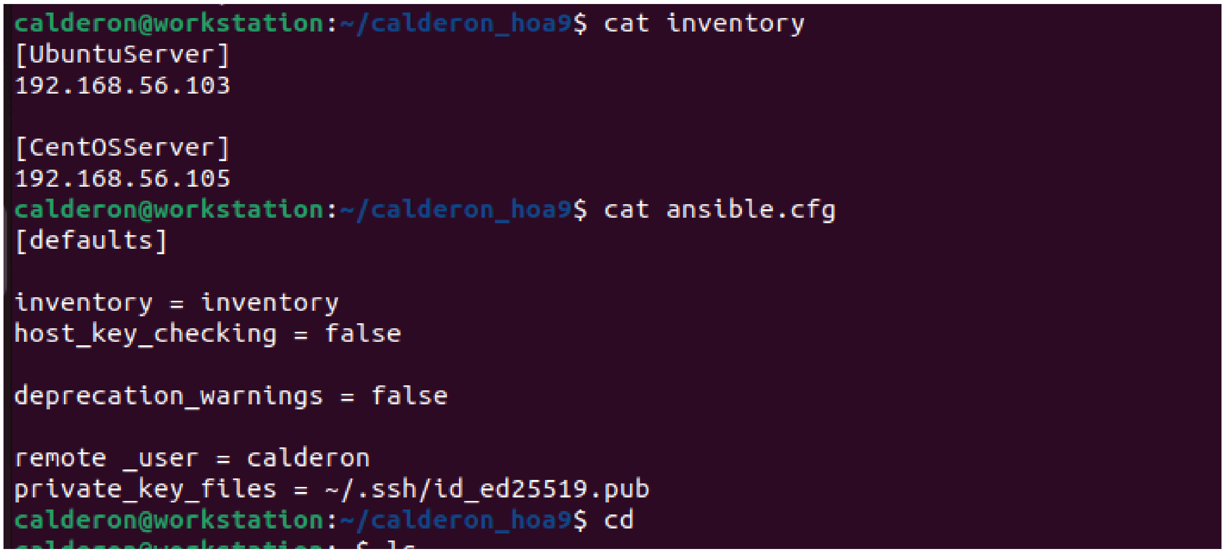
<u>Creating a ‘Roles’ directory and their required contents</u>



```
calderon@workstation:~/calderon_hoa9$ ls
README.md
calderon@workstation:~/calderon_hoa9$ mkdir roles
calderon@workstation:~/calderon_hoa9$ cd roles Ubunto CentOS
bash: cd: too many arguments
calderon@workstation:~/calderon_hoa9$ cd roles
calderon@workstation:~/calderon_hoa9/roles$ mkdir  Ubunto CentOS
calderon@workstation:~/calderon_hoa9/roles$ mkdir ./Ubuntu/tasks
mkdir: cannot create directory ‘./Ubuntu/tasks’: No such file or directory
calderon@workstation:~/calderon_hoa9/roles$ mkdir ./Ubunto/tasks
calderon@workstation:~/calderon_hoa9/roles$ mkdir ./CentOS/tasks
calderon@workstation:~/calderon_hoa9/roles$ cd ..
calderon@workstation:~/calderon_hoa9$ tree
.
├── README.md
└── roles
    ├── CentOS
    │   └── tasks
    └── Ubunto
        └── tasks

5 directories, 1 file
calderon@workstation:~/calderon_hoa9$ sudo nano inventory
[sudo] password for calderon:
calderon@workstation:~/calderon_hoa9$ sudo nano ansible.cfg
calderon@workstation:~/calderon_hoa9$ sudo nano ansible.cfg
calderon@workstation:~/calderon_hoa9$ tree
.
├── ansible.cfg
├── inventory
├── README.md
└── roles
    ├── CentOS
    │   └── tasks
    └── Ubunto
        └── tasks
```

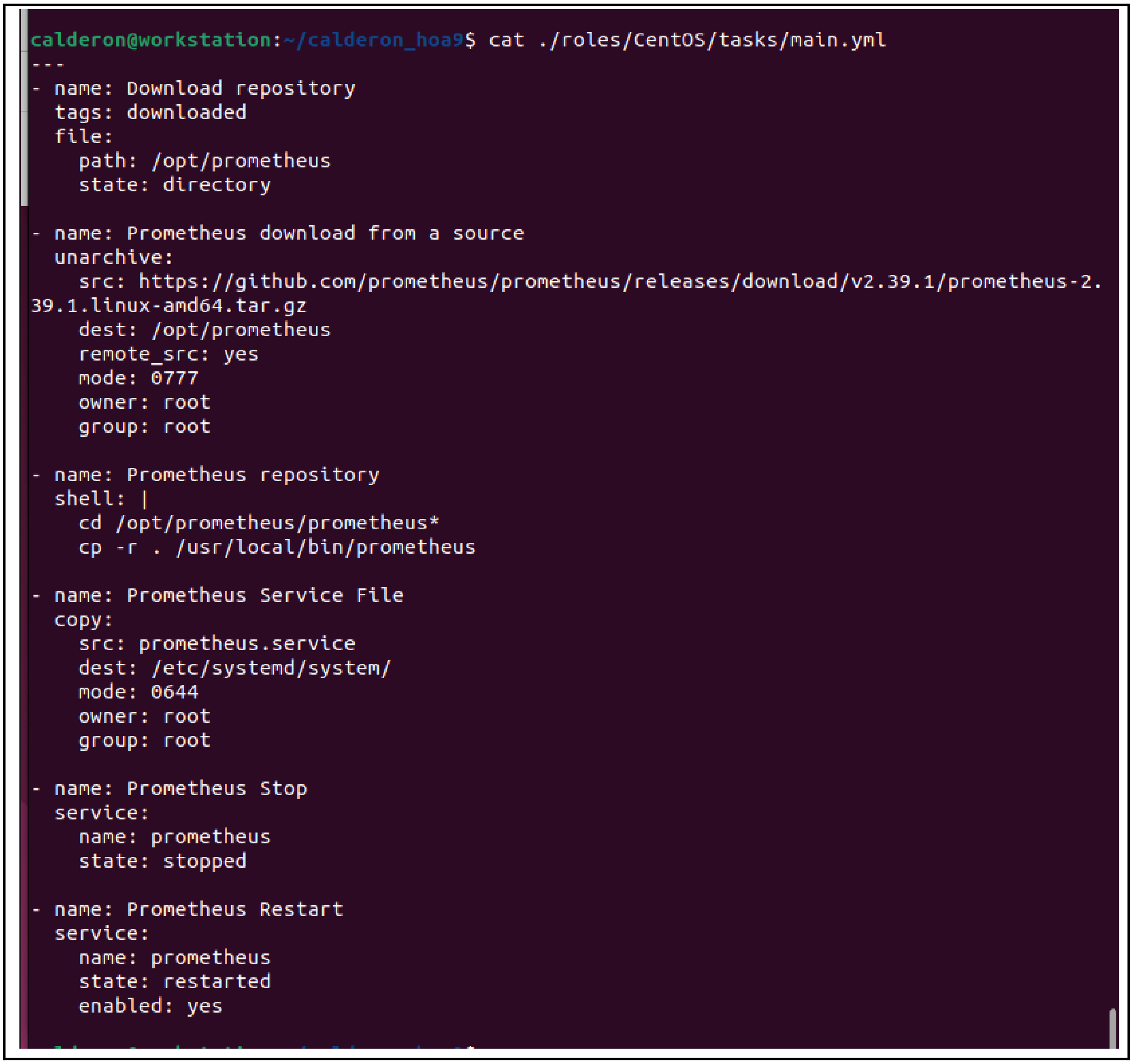
<u>I created a directory named *roles* that contains 2 roles: *CentOS* and *Ubuntu*</u>

I created inventory and ansible.cfg file in the directory where you'll be running Ansible commands or playbooks.

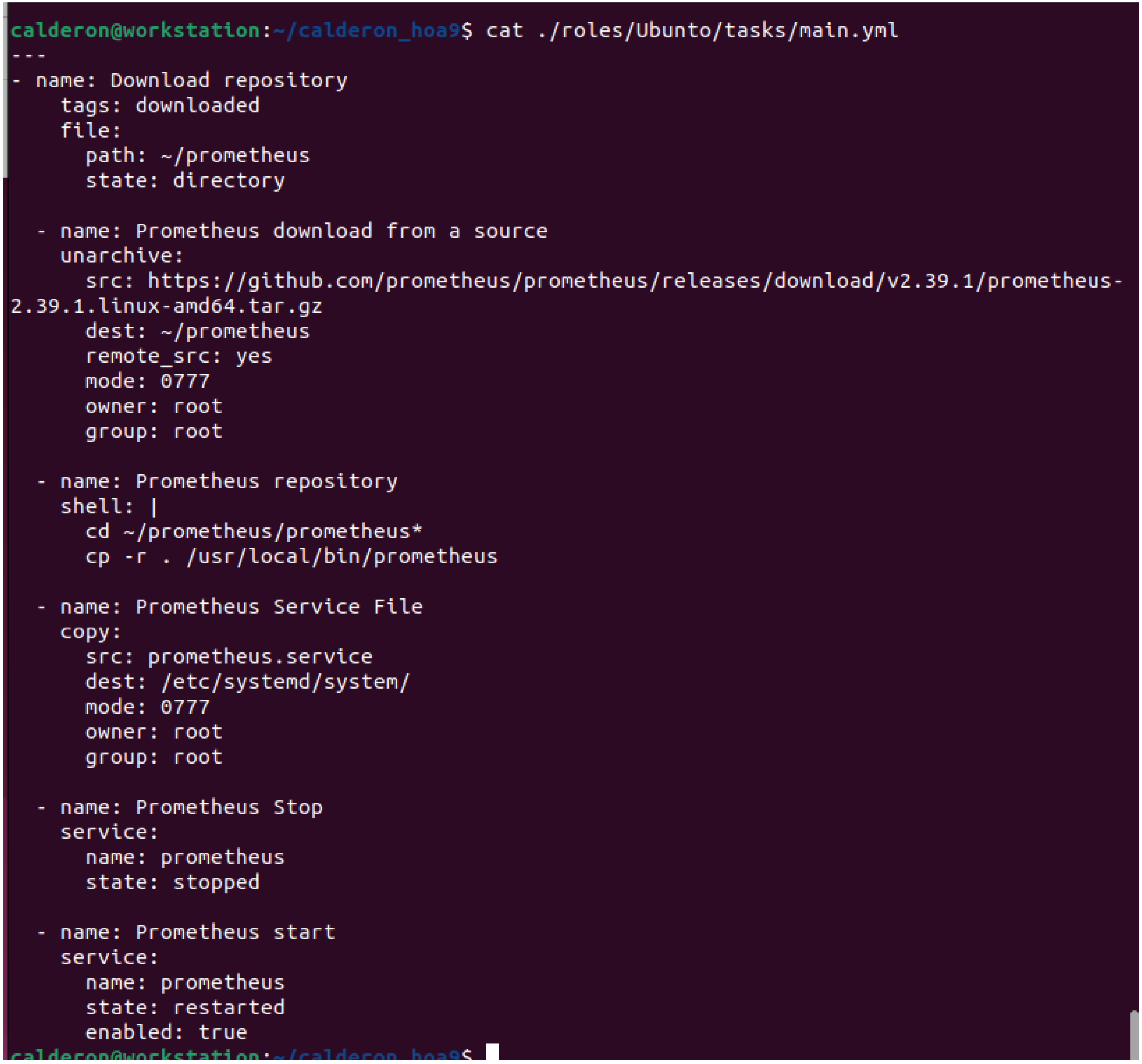


```
calderon@workstation:~/calderon_hoa9$ cat inventory
[UbuntuServer]
192.168.56.103

[CentOSServer]
192.168.56.105
calderon@workstation:~/calderon_hoa9$ cat ansible.cfg
[defaults]

inventory = inventory
host_key_checking = false

deprecation_warnings = false

remote _user = calderon
private_key_files = ~/.ssh/id_ed25519.pub
calderon@workstation:~/calderon_hoa9$ cd
calderon@workstation:~$ ls
```

I created tasks for both CentOS and Ubuntu, these tasks ensure that Prometheus is installed on CentOS and Ubuntu systems using the appropriate package management commands specific to each distribution.

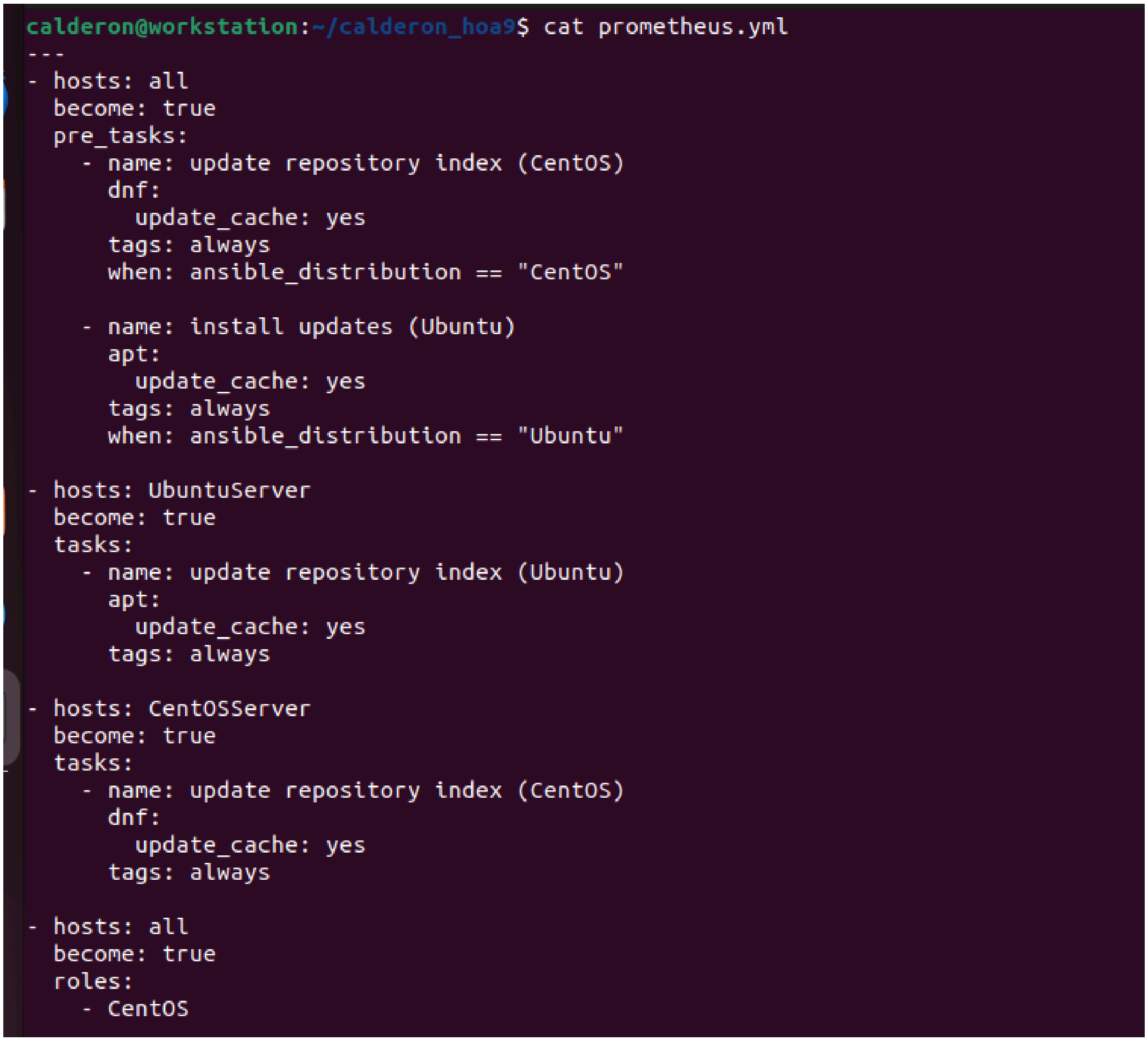


```
calderon@workstation:~/calderon_hoa9$ cat ./roles/CentOS/tasks/main.yml
---
- name: Download repository
  tags: downloaded
  file:
    path: /opt/prometheus
    state: directory

- name: Prometheus download from a source
  unarchive:
    src: https://github.com/prometheus/prometheus/releases/download/v2.39.1/prometheus-2.
39.1.linux-amd64.tar.gz
    dest: /opt/prometheus
    remote_src: yes
    mode: 0777
    owner: root
    group: root

- name: Prometheus repository
  shell: |
    cd /opt/prometheus/prometheus*
    cp -r . /usr/local/bin/prometheus

- name: Prometheus Service File
  copy:
    src: prometheus.service
    dest: /etc/systemd/system/
    mode: 0644
    owner: root
    group: root

- name: Prometheus Stop
  service:
    name: prometheus
    state: stopped

- name: Prometheus Restart
  service:
    name: prometheus
    state: restarted
    enabled: yes
```



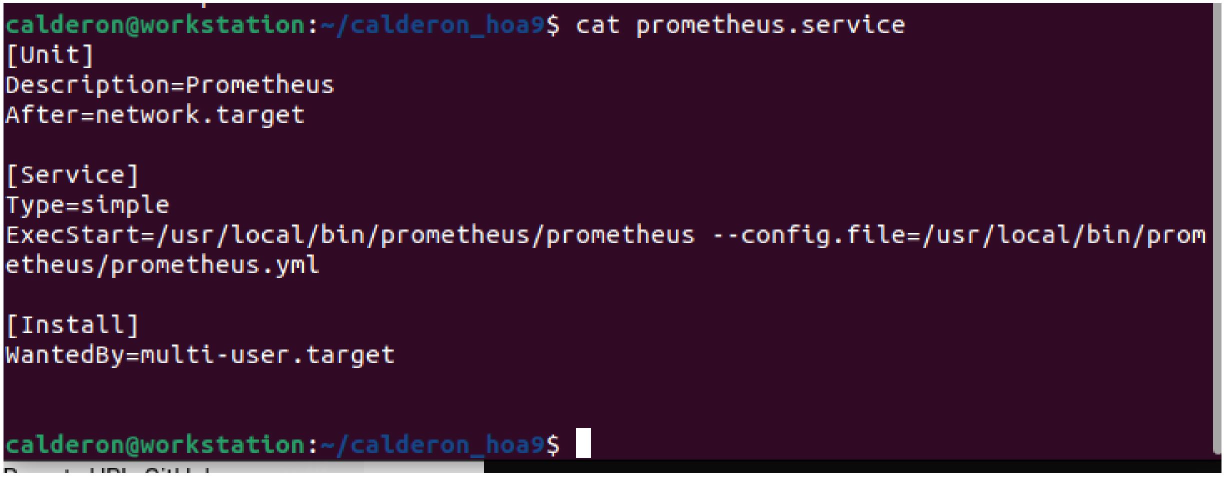
```
calderon@workstation:~/calderon_hoa9$ cat ./roles/Ubunto/tasks/main.yml
---
- name: Download repository
    tags: downloaded
    file:
      path: ~/prometheus
      state: directory

  - name: Prometheus download from a source
    unarchive:
      src: https://github.com/prometheus/prometheus/releases/download/v2.39.1/prometheus-2.39.1.linux-amd64.tar.gz
      dest: ~/prometheus
      remote_src: yes
      mode: 0777
      owner: root
      group: root

  - name: Prometheus repository
    shell: |
      cd ~/prometheus/prometheus*
      cp -r . /usr/local/bin/prometheus

  - name: Prometheus Service File
    copy:
      src: prometheus.service
      dest: /etc/systemd/system/
      mode: 0777
      owner: root
      group: root

  - name: Prometheus Stop
    service:
      name: prometheus
      state: stopped

  - name: Prometheus start
    service:
      name: prometheus
      state: restarted
      enabled: true
calderon@workstation:~/calderon_hoa9$
```

I created prometheus.yml. This Ansible playbook, `prometheus.yml`, is designed to update the repository index and install updates on both Ubuntu and CentOS servers.



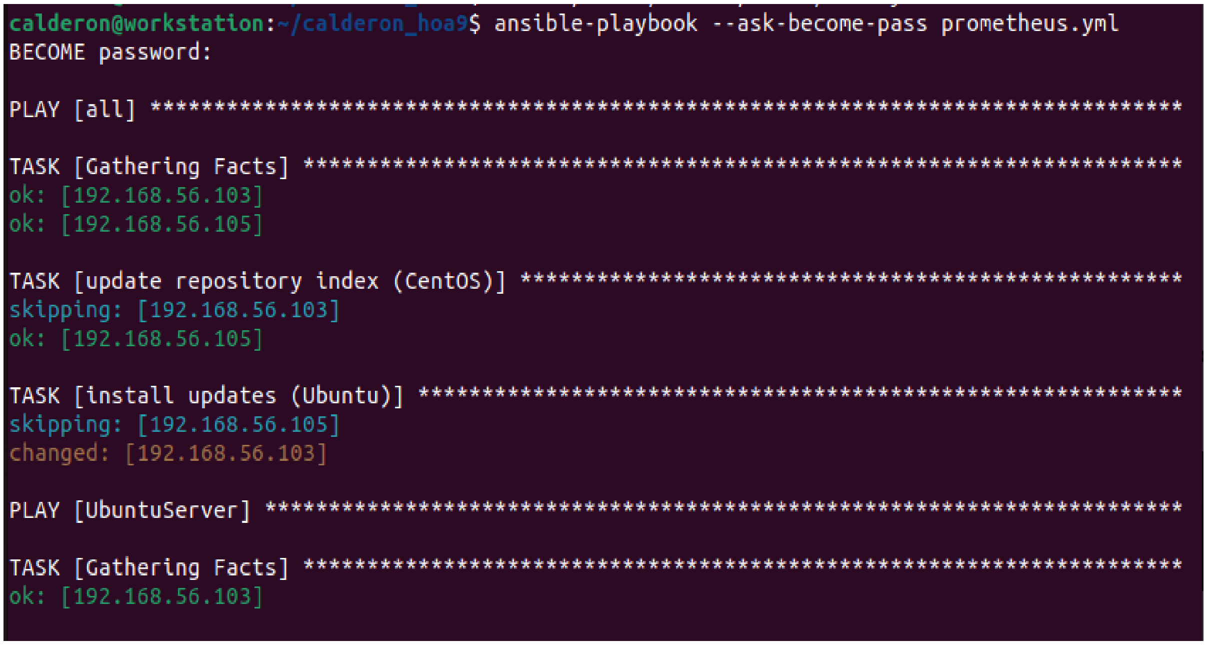
```
calderon@workstation:~/calderon_hoa9$ cat prometheus.yml
---
- hosts: all
  become: true
  pre_tasks:
    - name: update repository index (CentOS)
      dnf:
        update_cache: yes
      tags: always
      when: ansible_distribution == "CentOS"

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

- hosts: UbuntuServer
  become: true
  tasks:
    - name: update repository index (Ubuntu)
      apt:
        update_cache: yes
      tags: always

- hosts: CentOSServer
  become: true
  tasks:
    - name: update repository index (CentOS)
      dnf:
        update_cache: yes
      tags: always

- hosts: all
  become: true
  roles:
    - CentOS
```

I created a new directory named *'files'* inside would be the *service file* that is needed to run *Prometheus* service in every boot-up*

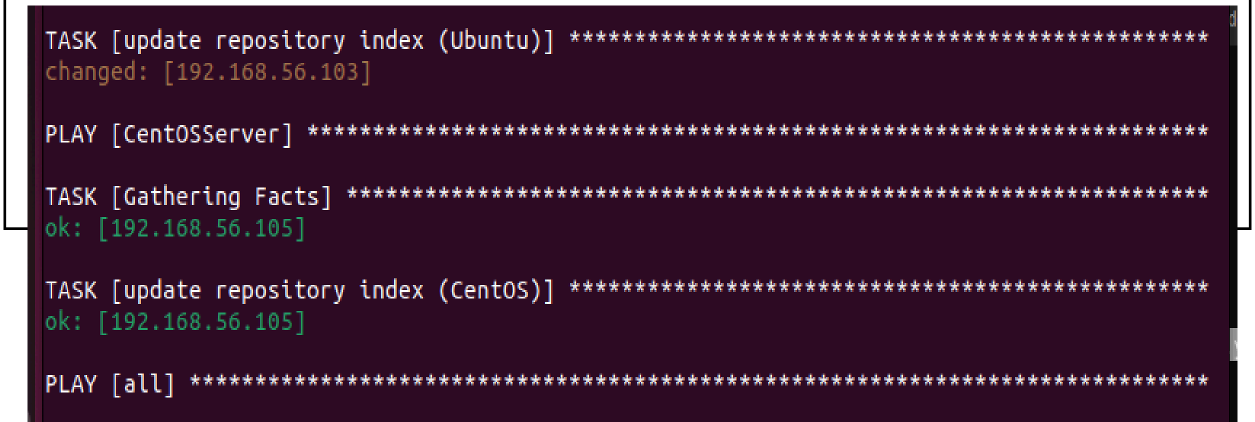


```
calderon@workstation:~/calderon_hoa9$ cat prometheus.service
[Unit]
Description=Prometheus
After=network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/prometheus/prometheus --config.file=/usr/local/bin/prom
etheus/prometheus.yml

[Install]
WantedBy=multi-user.target


calderon@workstation:~/calderon_hoa9$ 
```

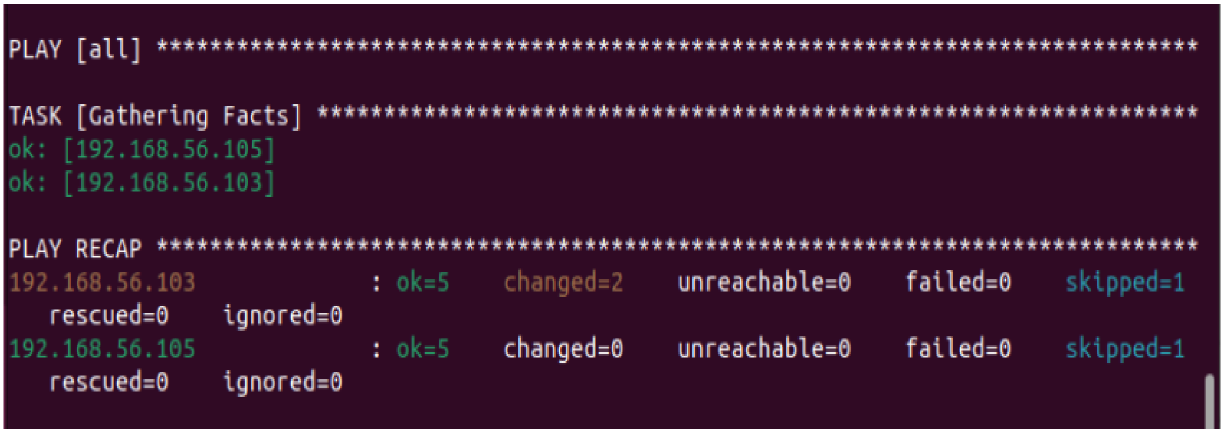
I run this command ansible-playbook --ask-become-pass prometheus.yml. By running this command, Ansible will execute the tasks defined in the `prometheus.yml` playbook, prompting the user for the privilege escalation password when necessary to perform privileged operations on remote hosts. let's see what happens.



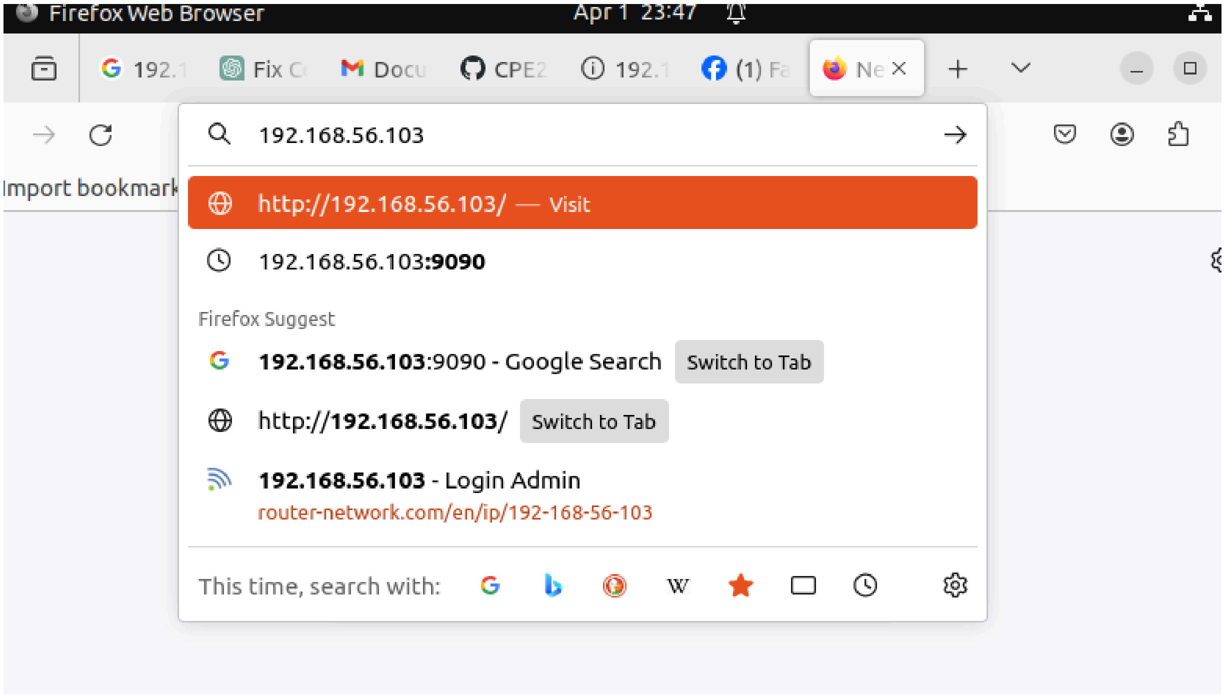
```
calderon@workstation:~/calderon_hoa9$ ansible-playbook --ask-become-pass prometheus.yml
BECOME password:

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

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.103]
ok: [192.168.56.105]

TASK [update repository index (CentOS)] ****************************************
skipping: [192.168.56.103]
ok: [192.168.56.105]

TASK [install updates (Ubuntu)] ************************************************
skipping: [192.168.56.105]
changed: [192.168.56.103]

PLAY [UbuntuServer] ************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.103]
```



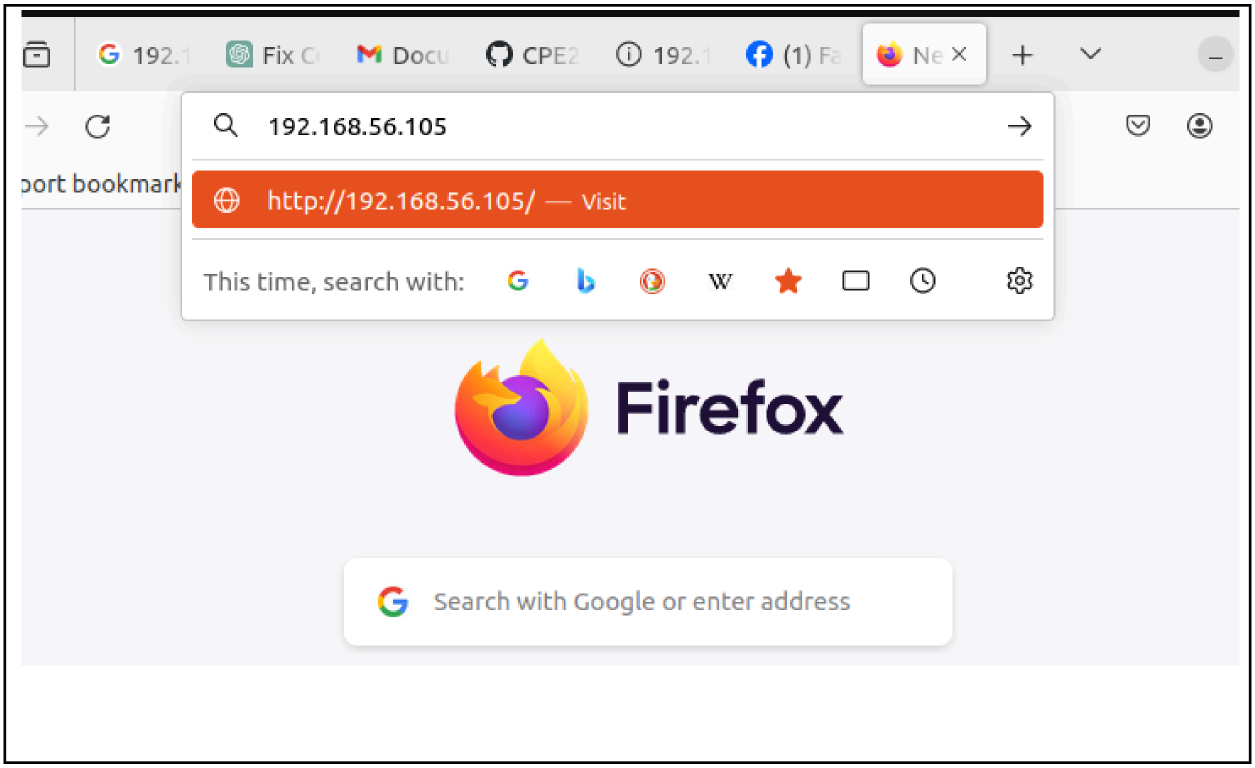
```
TASK [update repository index (Ubuntu)] ****************************************
changed: [192.168.56.103]

PLAY [CentOSServer] ************************************************************

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

TASK [update repository index (CentOS)] ****************************************
ok: [192.168.56.105]

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



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

TASK [Gathering Facts] *************************************************************
ok: [192.168.56.105]
ok: [192.168.56.103]

PLAY RECAP *************************************************************************
192.168.56.103             : ok=5    changed=2    unreachable=0    failed=0    skipped=1
   rescued=0    ignored=0
192.168.56.105             : ok=5    changed=0    unreachable=0    failed=0    skipped=1
   rescued=0    ignored=0
```



192.168.56.103

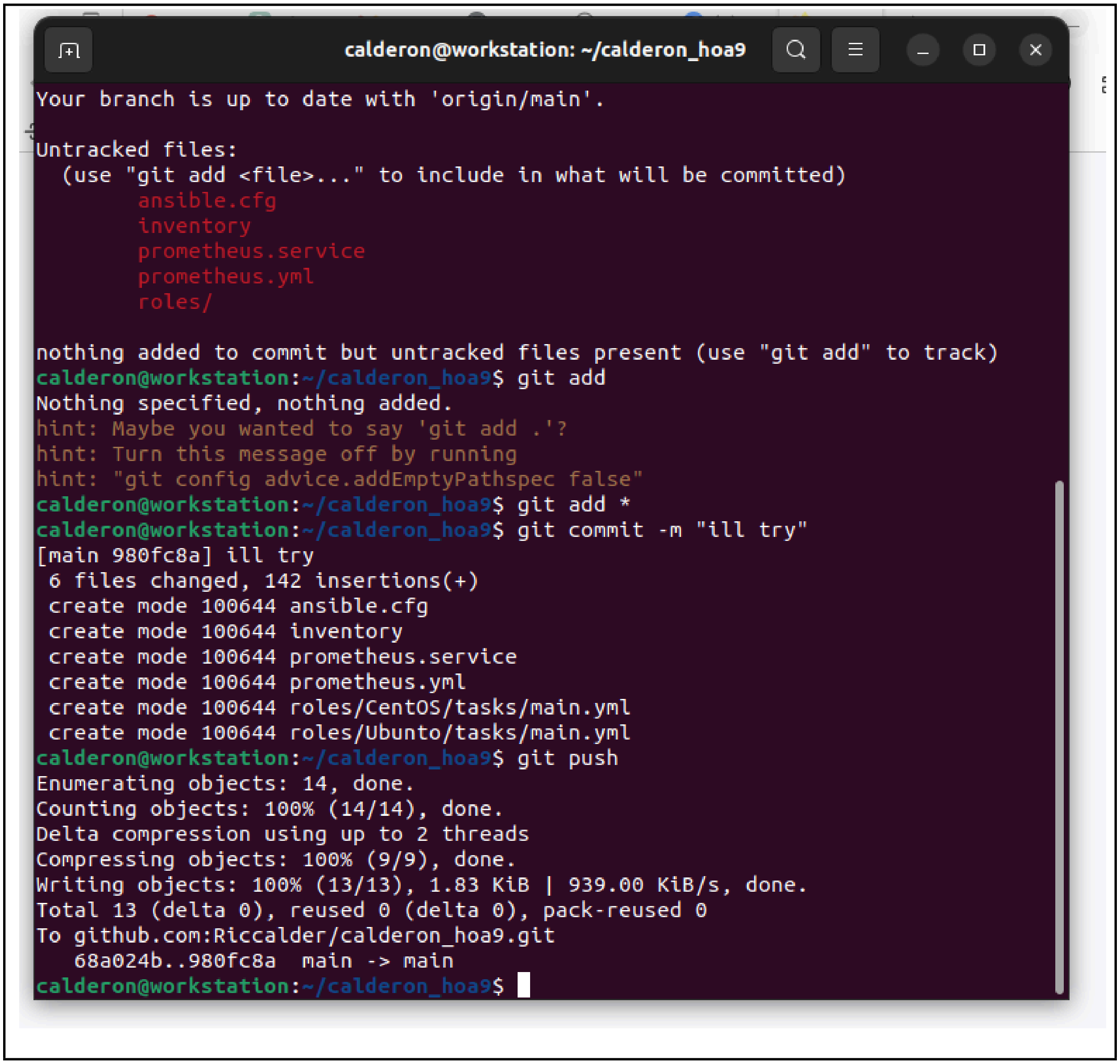


192.168.56.105

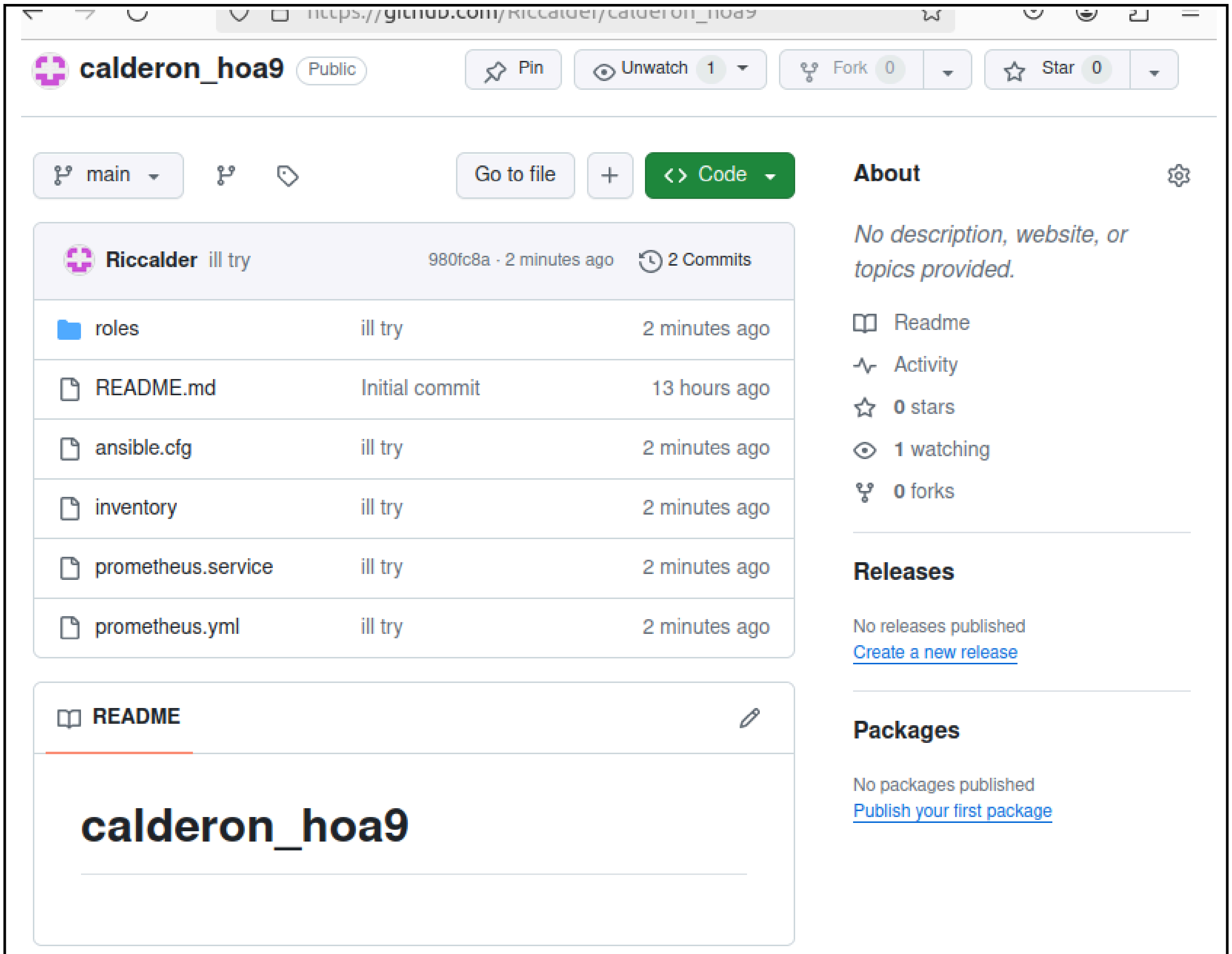
http://192.168.56.105/ — Visit

This time, search with:

Search with Google or enter address



```
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ansible.cfg
        inventory
        prometheus.service
        prometheus.yml
        roles/

nothing added to commit but untracked files present (use "git add" to track)
calderon@workstation:~/calderon_hoa9$ git add
Nothing specified, nothing added.
hint: Maybe you wanted to say 'git add .'?
hint: Turn this message off by running
hint: "git config advice.addEmptyPathspec false"
calderon@workstation:~/calderon_hoa9$ git add *
calderon@workstation:~/calderon_hoa9$ git commit -m "ill try"
[main 980fc8a] ill try
 6 files changed, 142 insertions(+)
 create mode 100644 ansible.cfg
 create mode 100644 inventory
 create mode 100644 prometheus.service
 create mode 100644 prometheus.yml
 create mode 100644 roles/CentOS/tasks/main.yml
 create mode 100644 roles/Ubunto/tasks/main.yml
calderon@workstation:~/calderon_hoa9$ git push
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 2 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (13/13), 1.83 KiB | 939.00 KiB/s, done.
Total 13 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:Riccalder/calderon_hoa9.git
   68a024b..980fc8a  main -> main
calderon@workstation:~/calderon_hoa9$
```



Riccalder ill try 980fc8a · 2 minutes ago 2 Commits

roles	ill try	2 minutes ago
README.md	Initial commit	13 hours ago
ansible.cfg	ill try	2 minutes ago
inventory	ill try	2 minutes ago
prometheus.service	ill try	2 minutes ago
prometheus.yml	ill try	2 minutes ago

calderon_hoa9

No description, website, or topics provided.

Readme
Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

GitHub Repository link: https://github.com/Riccalder/calderon_hoa9

Reflections:

Answer the following:

1. What are the benefits of having a performance monitoring tool?

The benefits of having a performance monitoring tool are undeniable. From proactive issue detection and capacity planning to troubleshooting, optimization, and risk management, these tools play a crucial role in ensuring the reliability, efficiency, and scalability of modern IT infrastructure.

Conclusions:

Performance monitoring tools are essential for maintaining system reliability, efficiency, and scalability. They enable proactive management, rapid issue resolution, and effective resource optimization, ensuring seamless operations and user satisfaction.