Name: Seruelas, Ronn Kristoper H.	**Date Performed:** 10-09-2023
Course/Section: CPE 232 - CPE31S4	**Date Submitted:** 10-10-2023
Instructor: Dr. Jonathan V. Taylar	**Semester and SY:** 1st - 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 0: Preparation

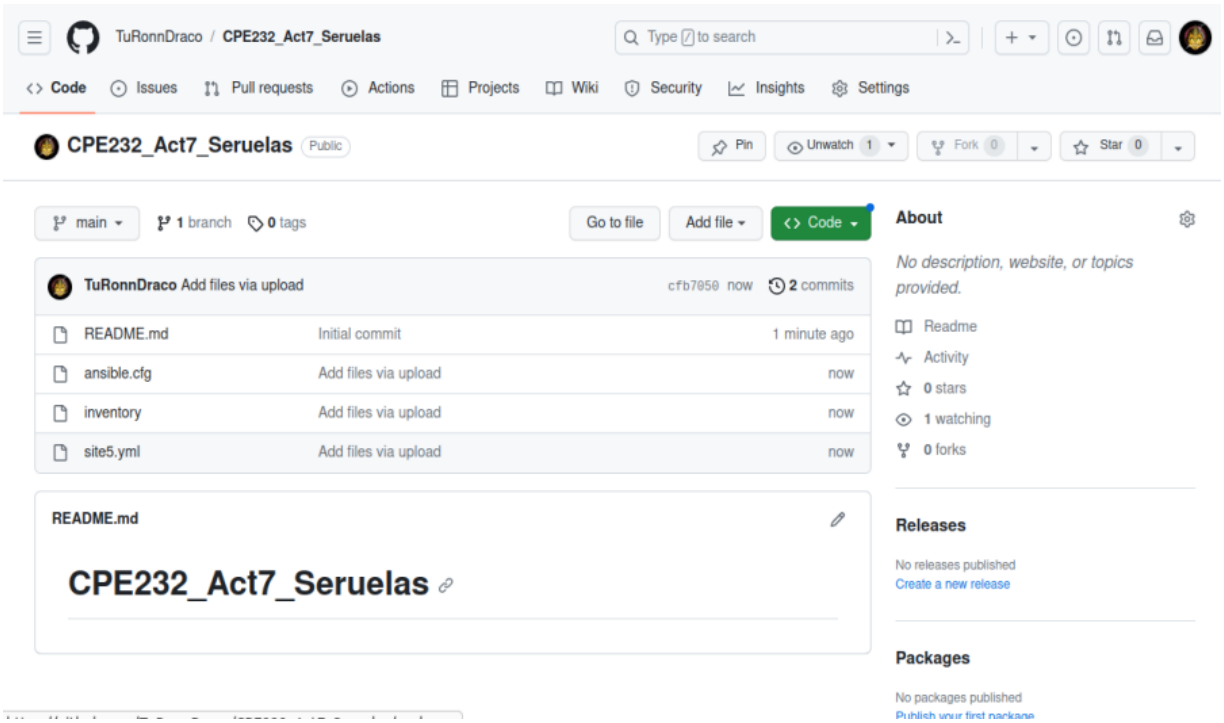

Figure 0.1 - Creation of the new repository, alongside with files from previous activity.

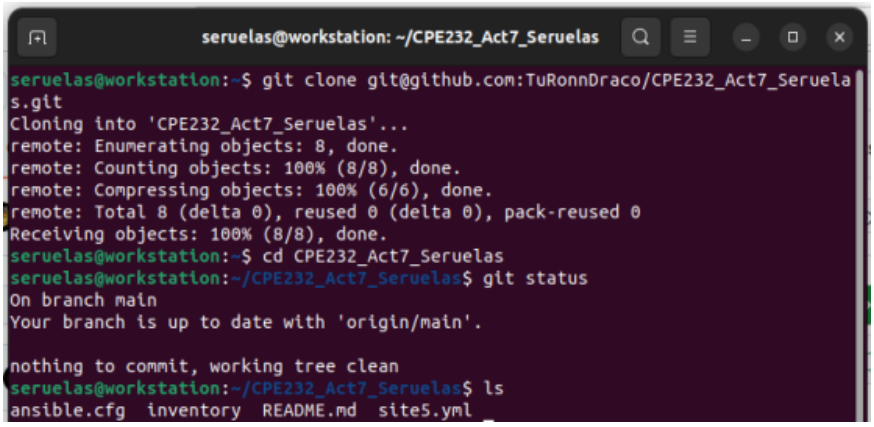

Figure 0.2 - Cloning of new repository to the workstation.

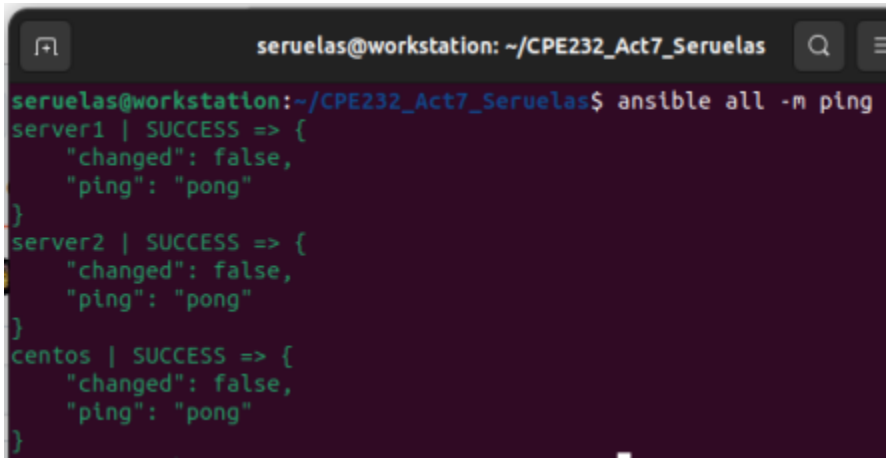

Figure 0.3 - Pinging all machines to verify online status.

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.

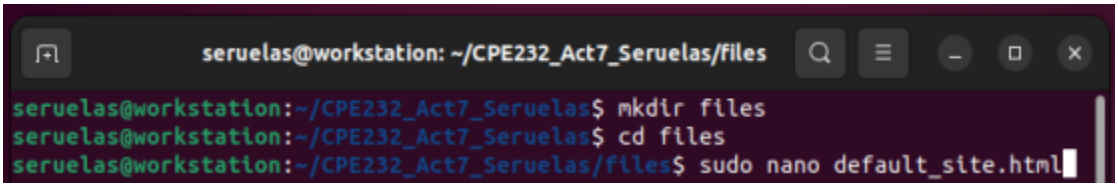

Figure 1.1.1 - Creation of the **files** directory and the **default_site.html**.

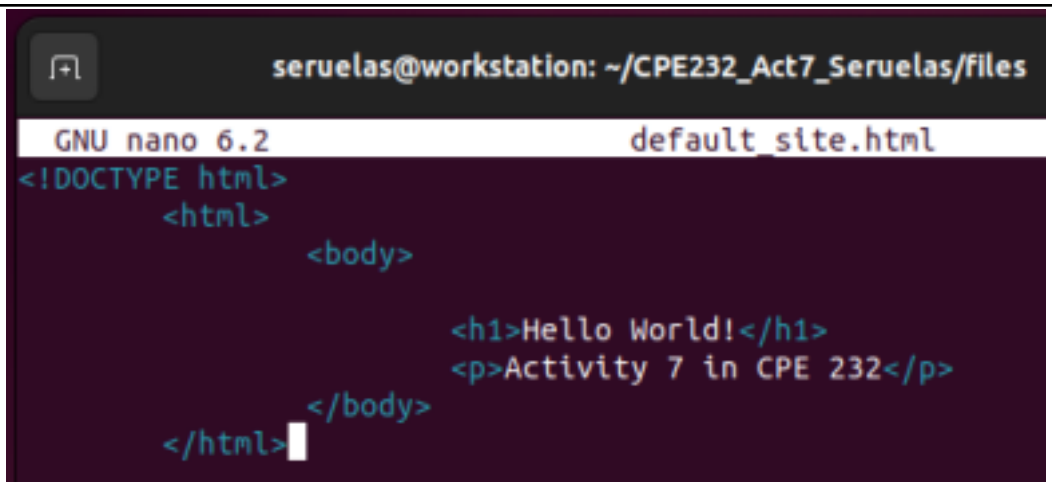

Figure 1.1.2 - Basic syntax inside of default_site.html.

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
```

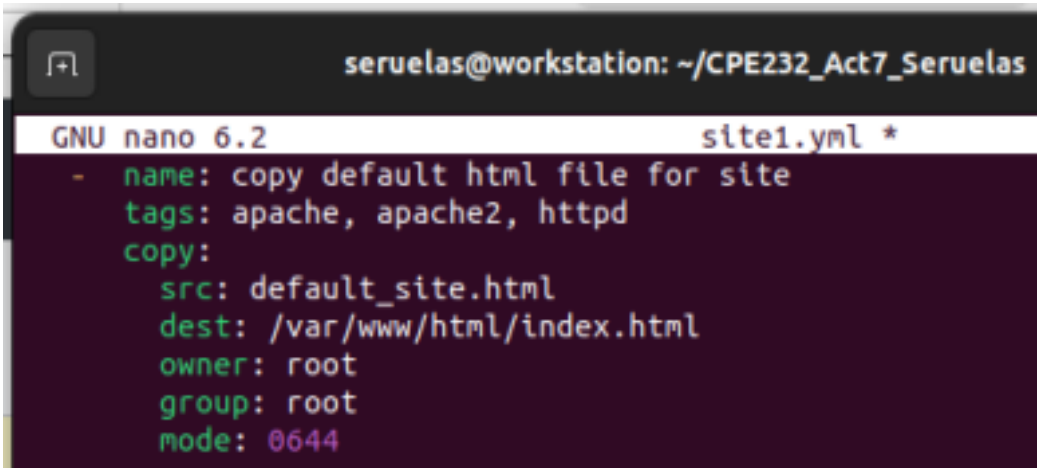

Figure 1.2.1 - Copying new task unto site.yml. (Created new version of site.yml)

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

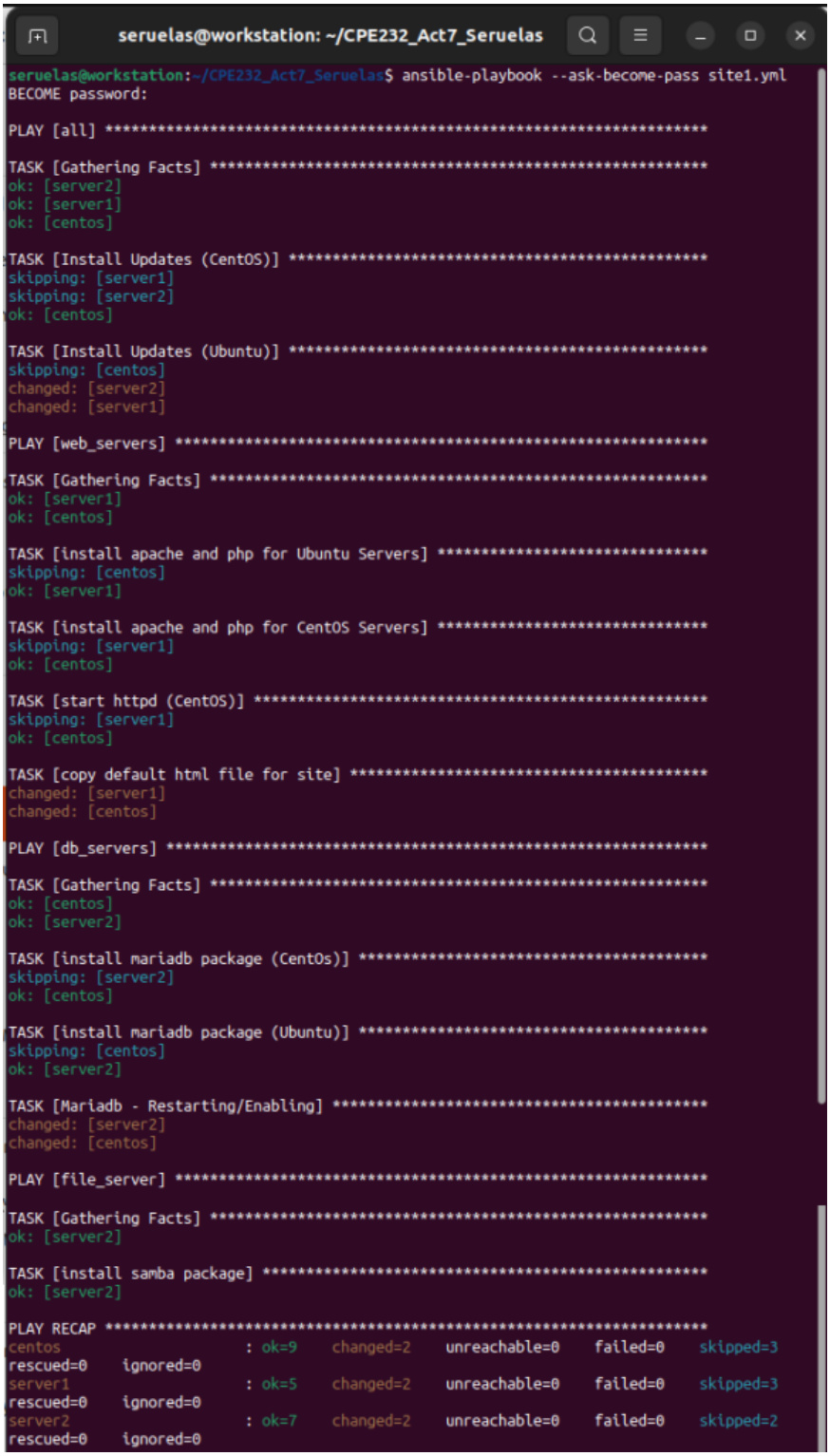

Figure 1.3.1 - Running the playbook added a new task which was executed in the remote group, web_servers.

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.

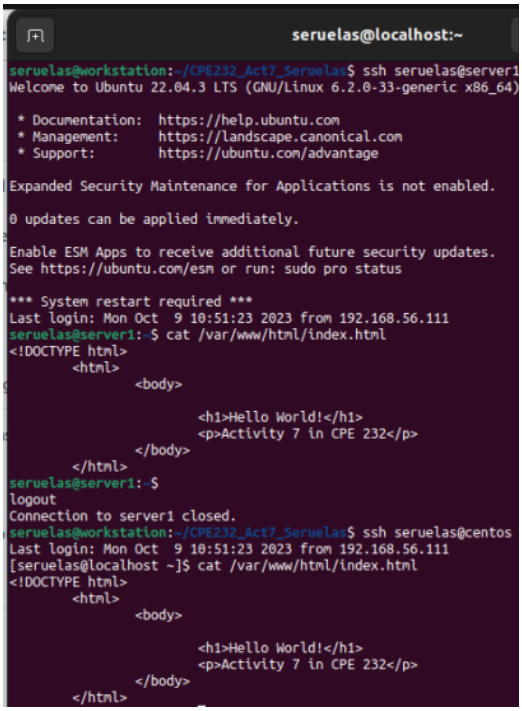

Figure 1.4.1 - Verification of successful copying of default_site.html unto the remote web_servers.

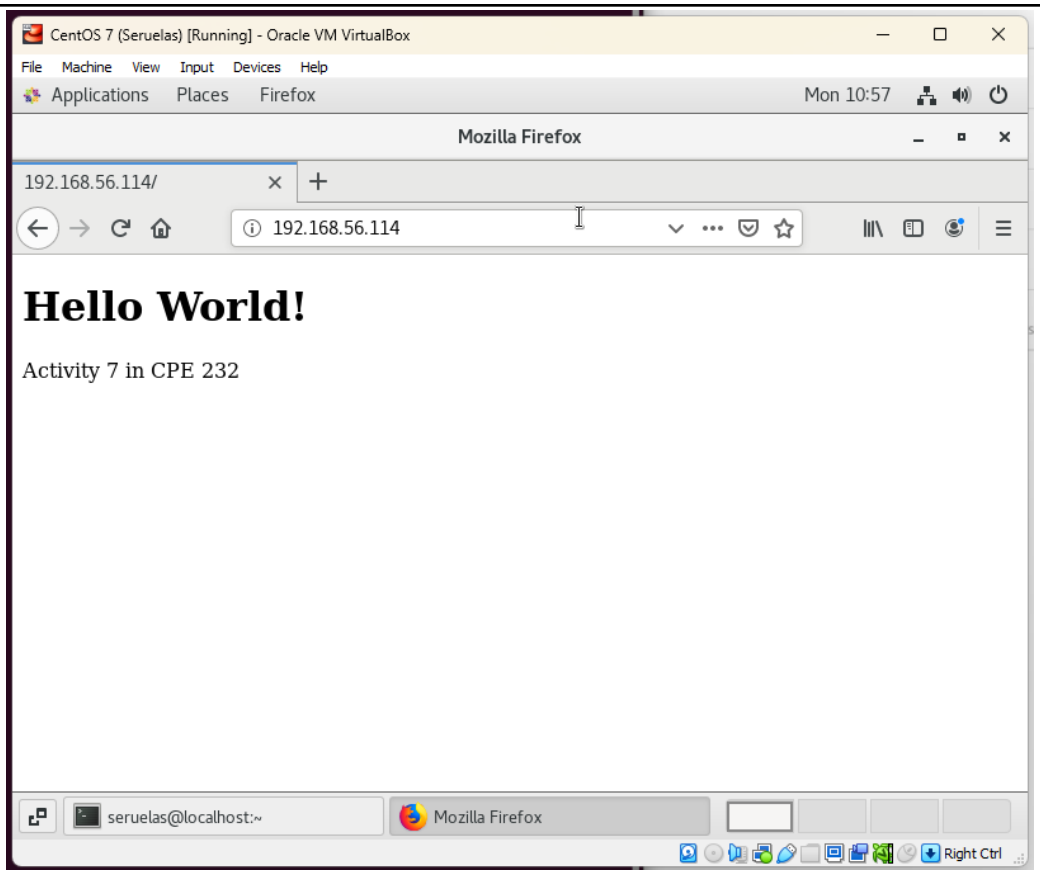

Figure 1.4.2 - Verification of successful transfer of default_site.html in CentOS.

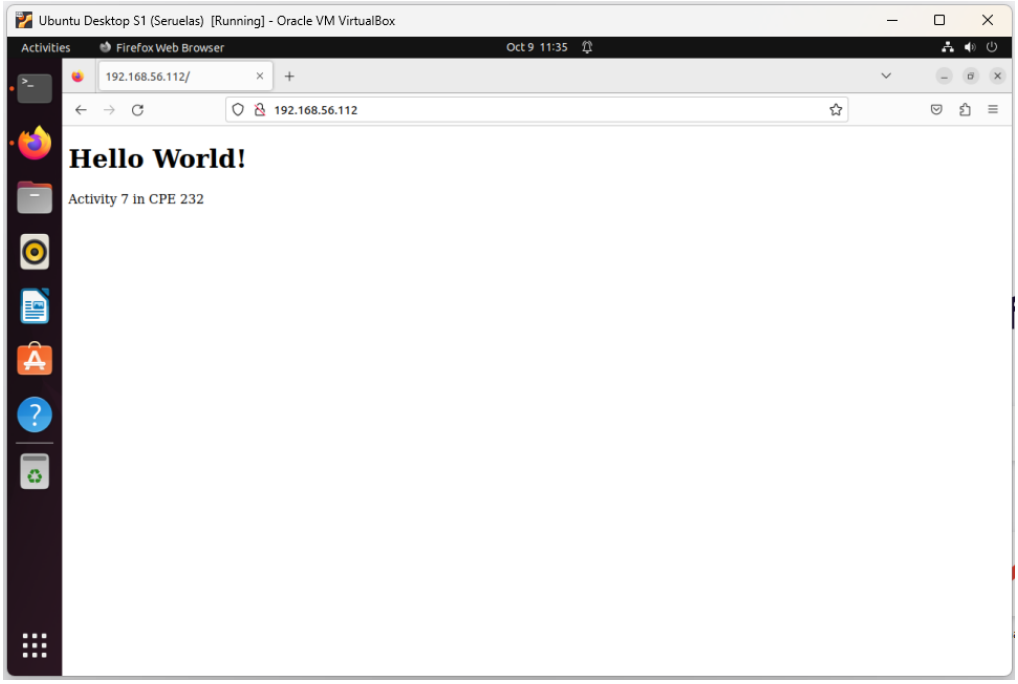

Figure 1.4.3 - Verification of successful transfer of default_site.html in Ubuntu Server1.

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

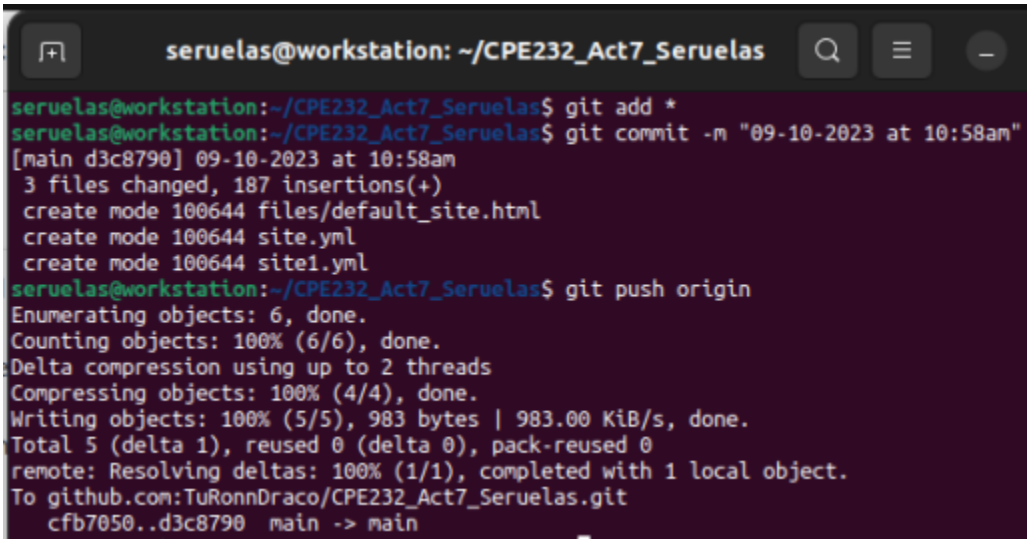

Figure 1.5.1 - Syncing local repository to GitHub repository.

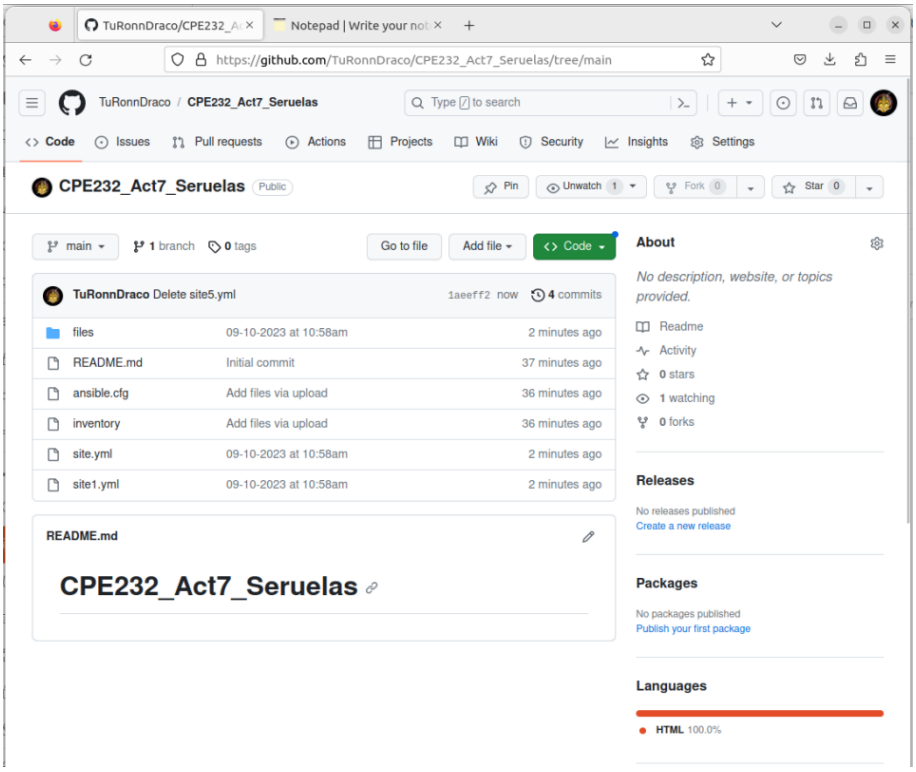

Figure 1.5.2 - Verification of sync in the GitHub repository. A change that can be seen is that there is a new section in GitHub where it displays the languages that is presently used in the repository.

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
```

Figure 2.1.1 - Adding additional tasks for remote group workstations in site.yml (New version of site.yml)

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

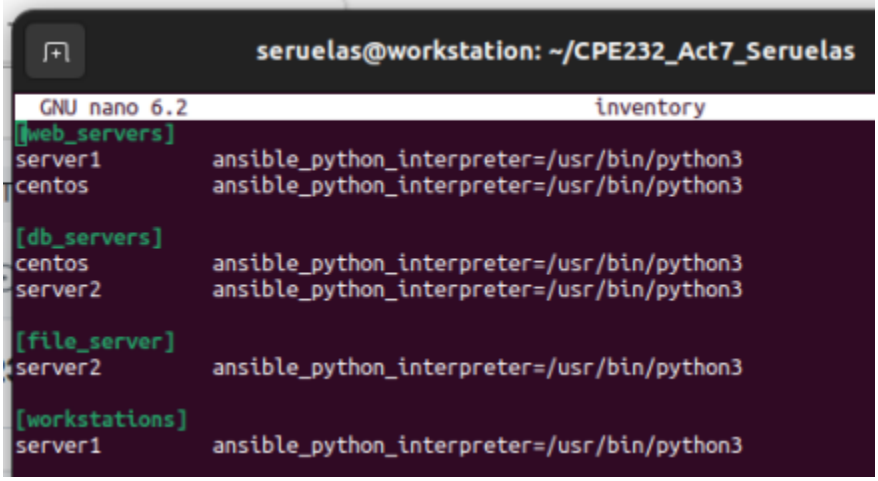

Figure 2.2.1 - Adding a new remote group in inventory, the workstations. (Choosing Ubuntu Server1 as the remote user)

3. Run the playbook. Describe the output.

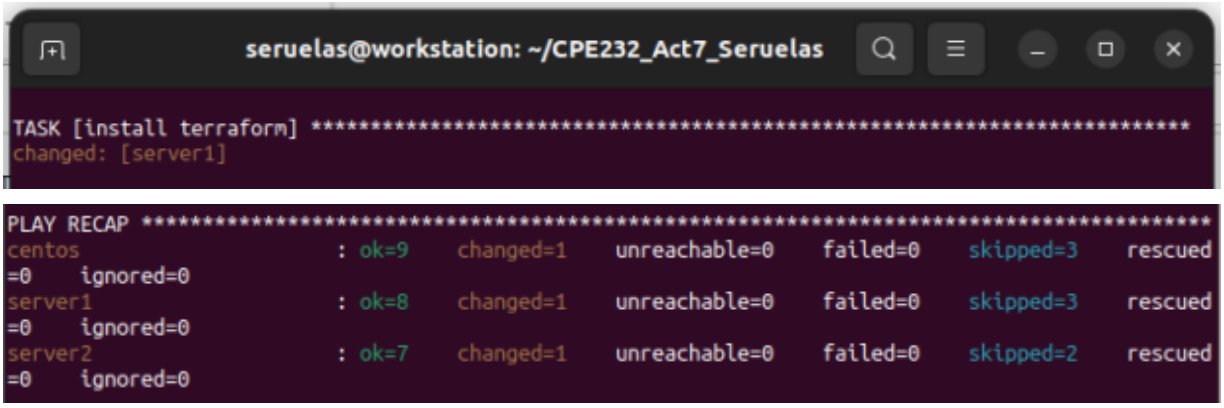

Figure 2.3.1-2.3.2 - Installation of the terraform was successful.

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

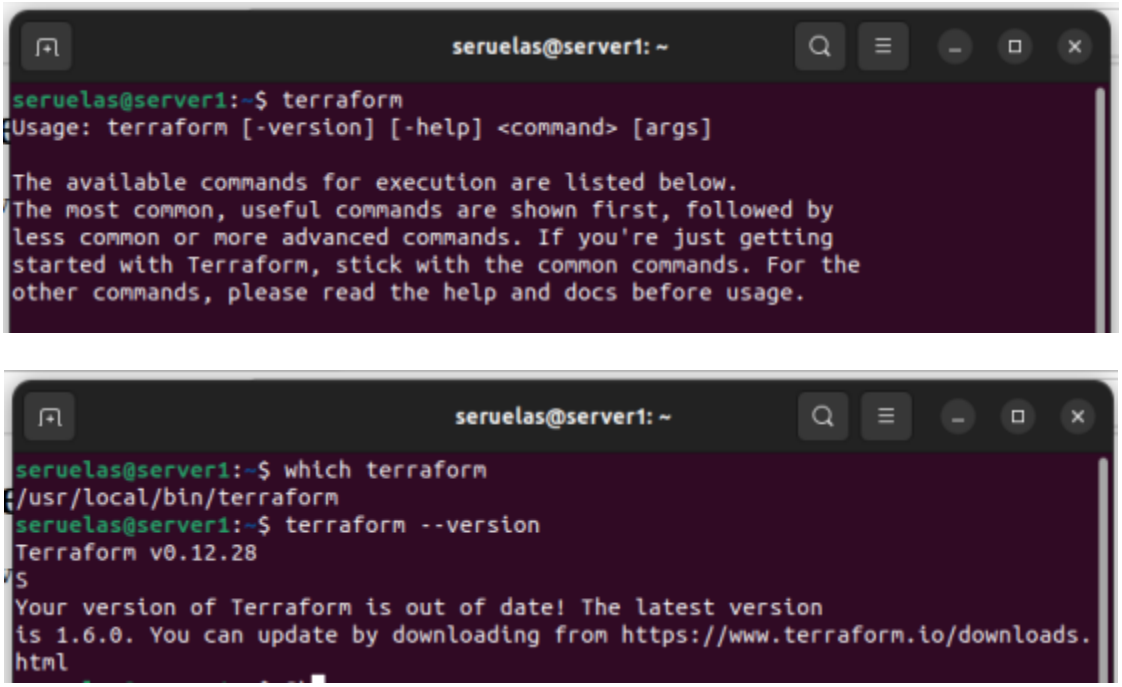

Figure 2.4.1-2.4.2 - Verification of successful installation of terraform in Ubuntu Server1.

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.

Figure 3.1.1 - Creation of new site.yml (New version of site.yml)

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.

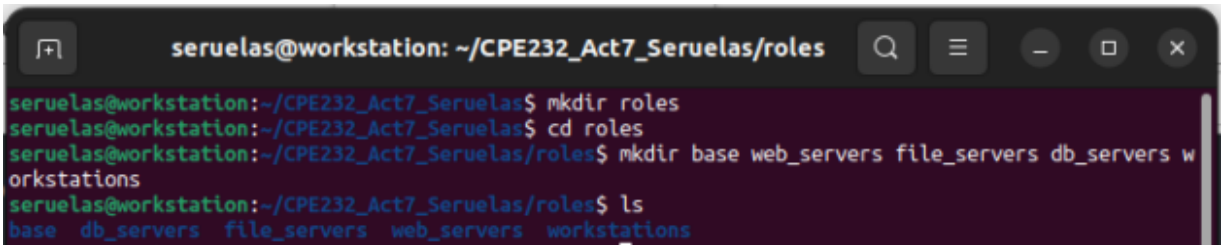

Figure 3.2.1 - Creation of the roles directory, inside of it being the directories regarding each remote group.

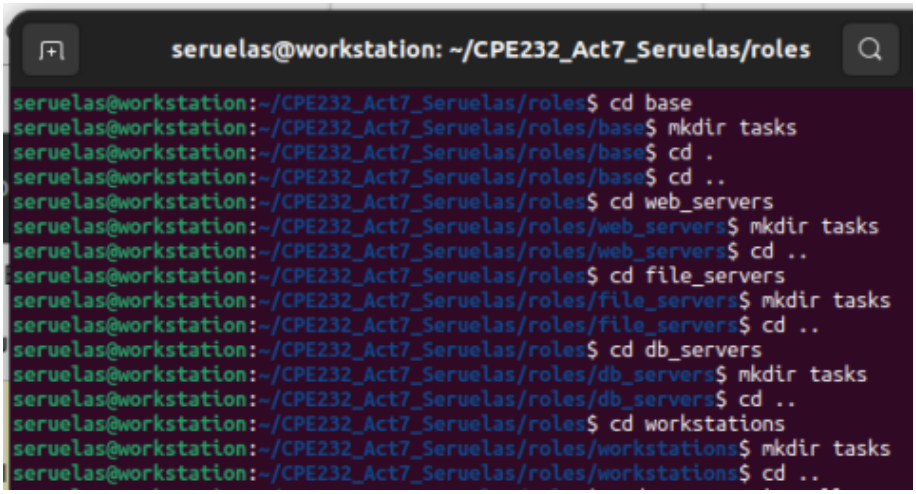

Figure 3.2.2 - Creation of tasks directory in each role directory.

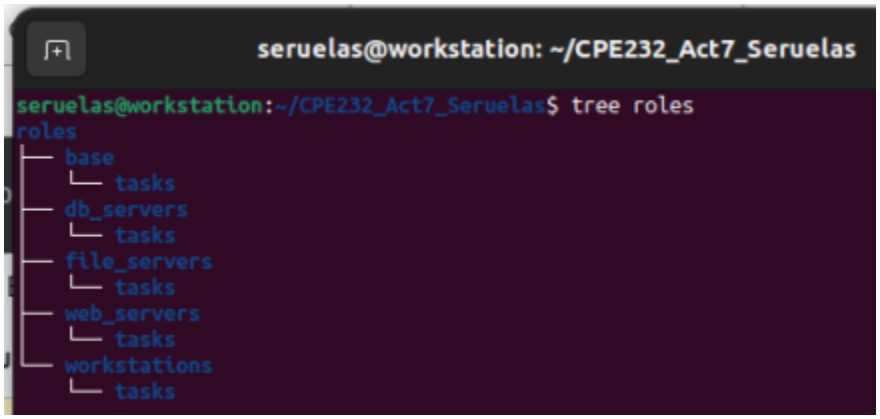

Figure 3.2.3 - Verification of new created directories.

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.

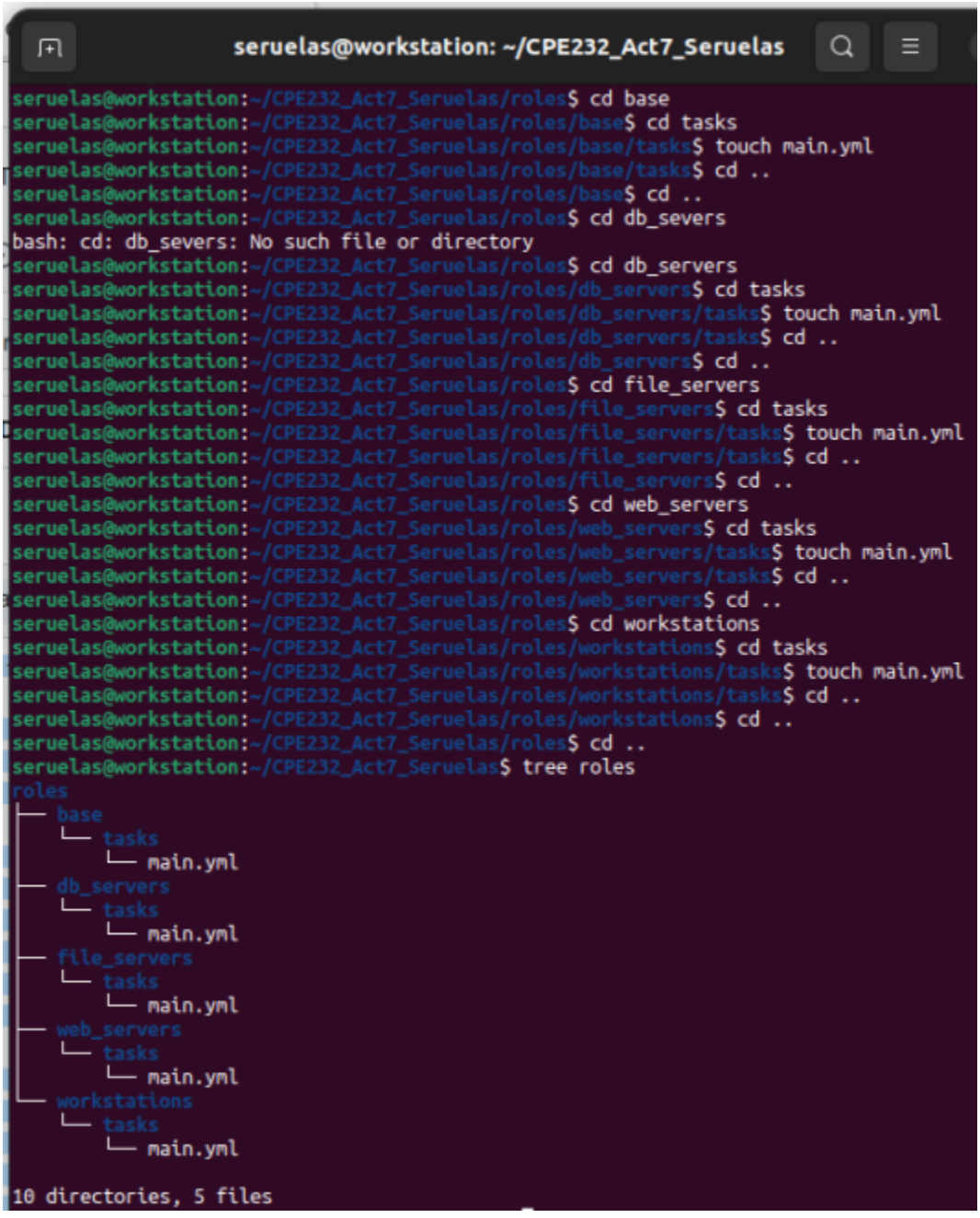

Figure 3.3.1 - Creation of main.yml in each tasks directory of each role directory.

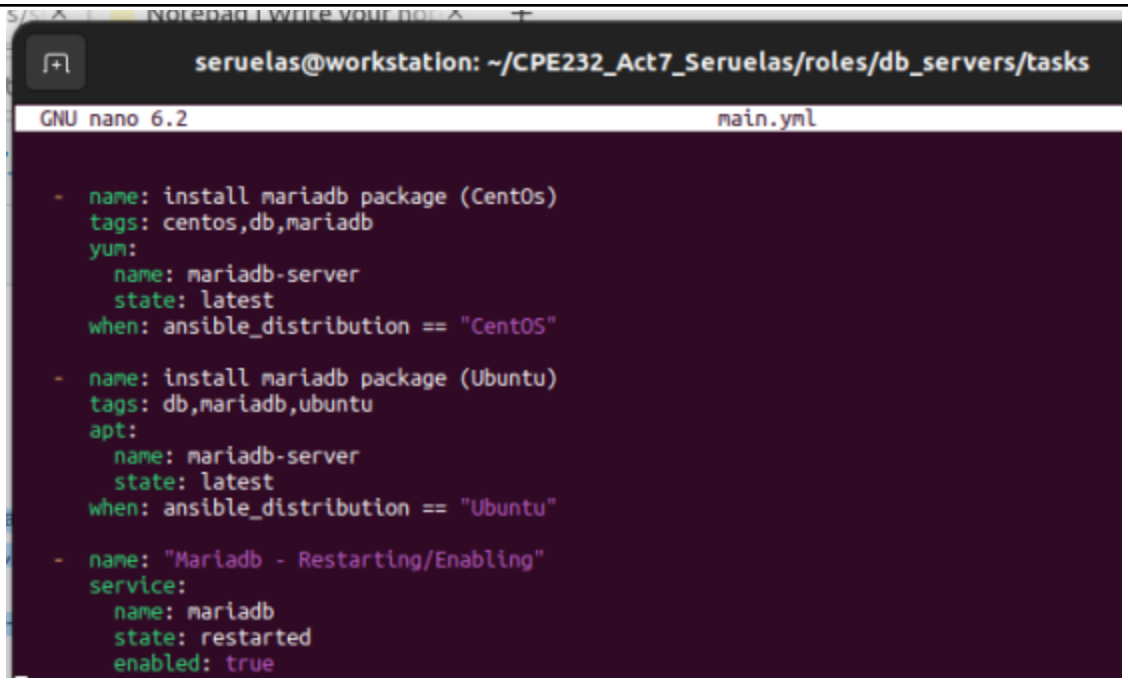

Figure 3.3.2 - main.yml content for role db_servers.

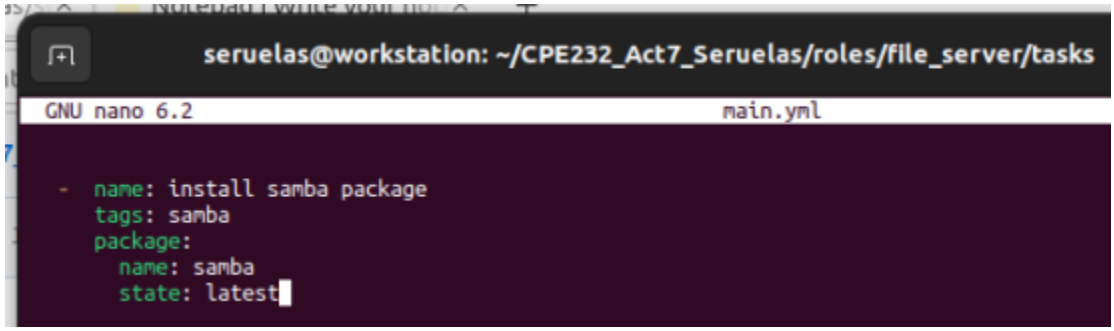

Figure 3.3.3 - main.yml content for role file_server.

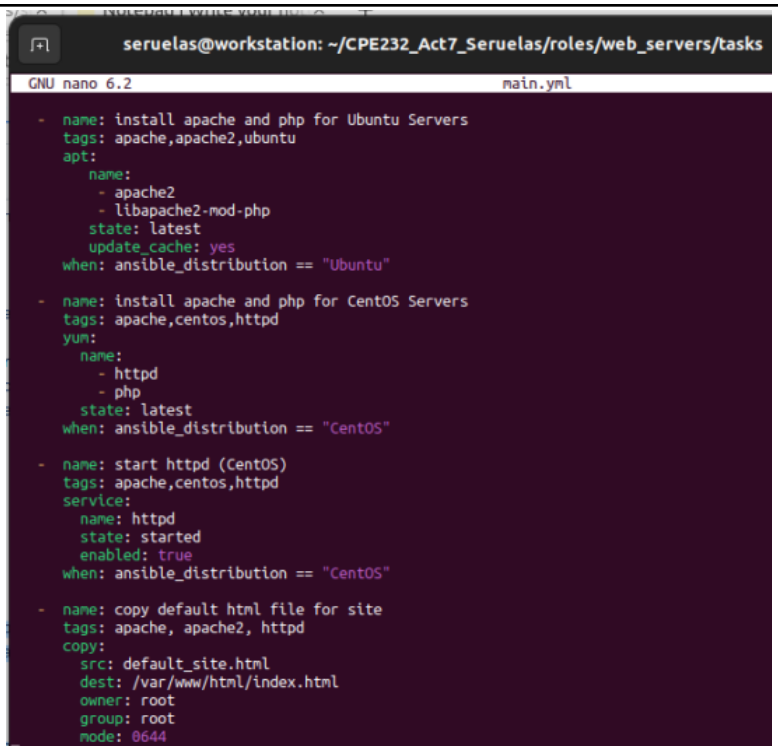

Figure 3.3.4 - main.yml content for role web_servers.

Figure 3.3.4 - main.yml content for role workstations.

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

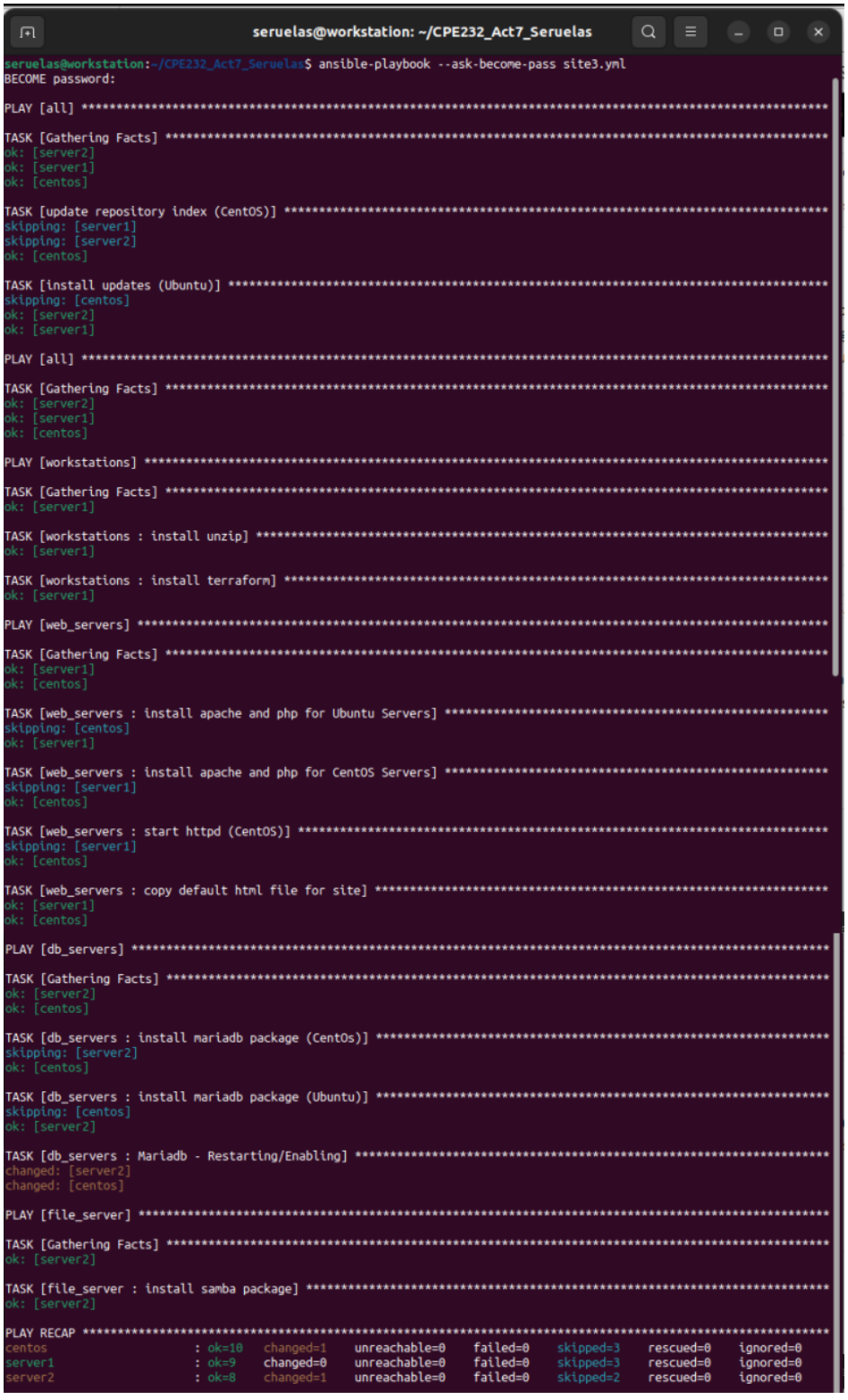

Figure 3.4.1 - Playbook ran shows that each tasks done were done by a specific role.

5. Push and save everything to GitHub.

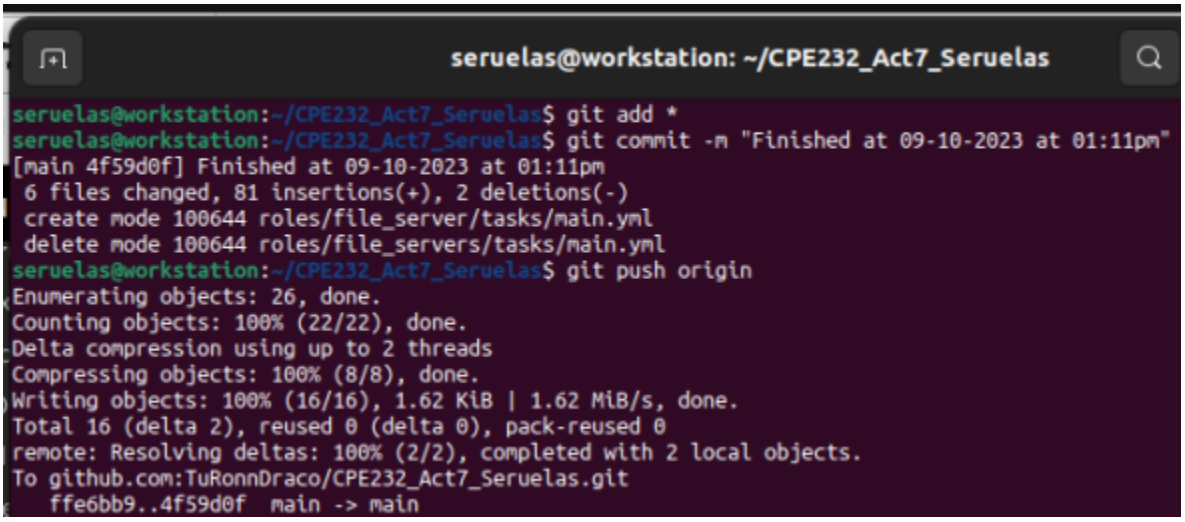

Figure 3.5.1 - Pushing local repository to GitHub.

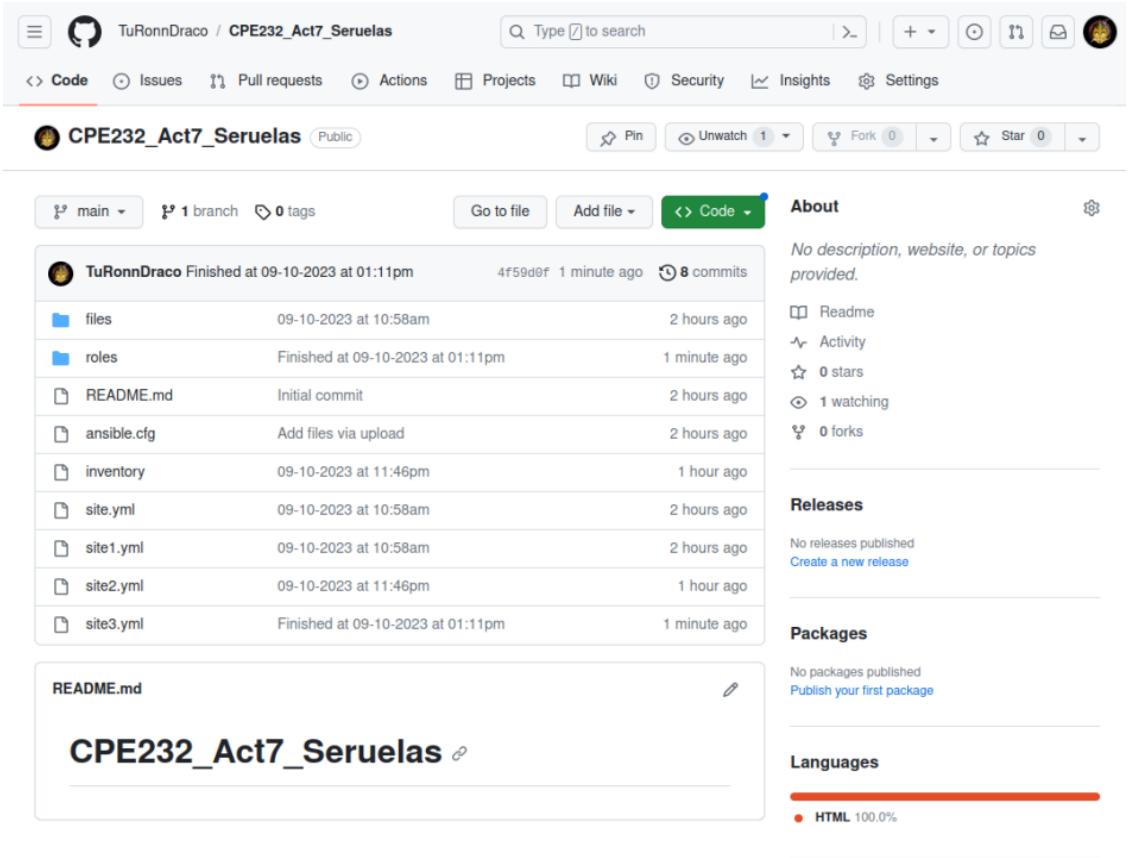

Figure 3.5.2 - Verification of latest commit into GitHub repository.

https://github.com/TuRonnDraco/CPE232_Act7_Seruelas

Reflections:

Answer the following:

1. What is the importance of creating roles?
    - The importance of creating roles in a database is that it allows us to separate and to specify which tasks are to be executed by each role or web group that is given to. The main purpose of roles is to allow the administrators to separate and to organize the tasks of each remote hosts that is grouped by inventory and remote groups, allowing the main.yml to have more memory that executes the tasks of the remote groups separately from the main.yml

2. What is the importance of managing files?
    - The importance of managing files in a database is that it allows the administrator to have more memory and space and organization in their files, allowing more setups and fixated steps for their database, to avoid any conflicts in the files.

Conclusion:

In this activity, we were able to educate ourselves in the importance of managing files and creating roles in ansible. We were able to learn that managing files in our own database or repository allows us to have a more cleaner interface that allows us to create and to specify more setups in our own interface. Creating roles is a huge fundamental in ansible as it also allows us to specify and to preserve more memory in executing our playbooks, allowing us to have better performance and more efficient process in managing enterprise servers. To conclude this activity, we were able to manage our repository by organizing the roles and files needed to execute our playbook.

"I affirm that I have not received or given any unauthorized help on this activity and that all work is my own."