Name: Froilan Gayao	**Date Performed: 11/10/24**
Course/Section: CPE31S4	**Date Submitted: 11/10/24**
Instructor: Engr. Robin Valenzuela	**Semester and SY: 1st 24-25**

Activity 7: Managing Files and Creating Roles in Ansible

1. **Objectives:**

1.1 Manage files in remote servers

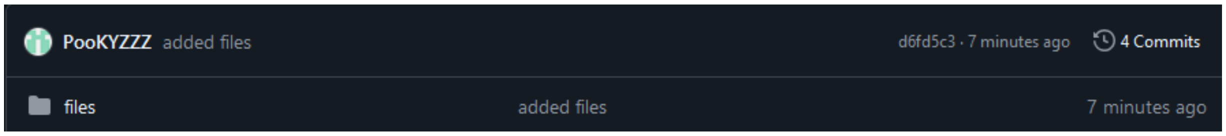
1.2 Implement roles in ansible

2. **Discussion**:

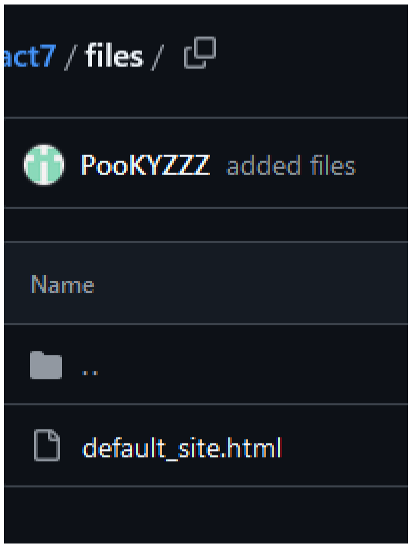
In this activity, we look at the concept of copying a file to a server. We are going to create a file into our git repository and use Ansible to grab that file and put it into a particular place so that we could do things like customize a default website, or maybe install a default configuration file. We will also implement roles to consolidate plays.

Task 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.



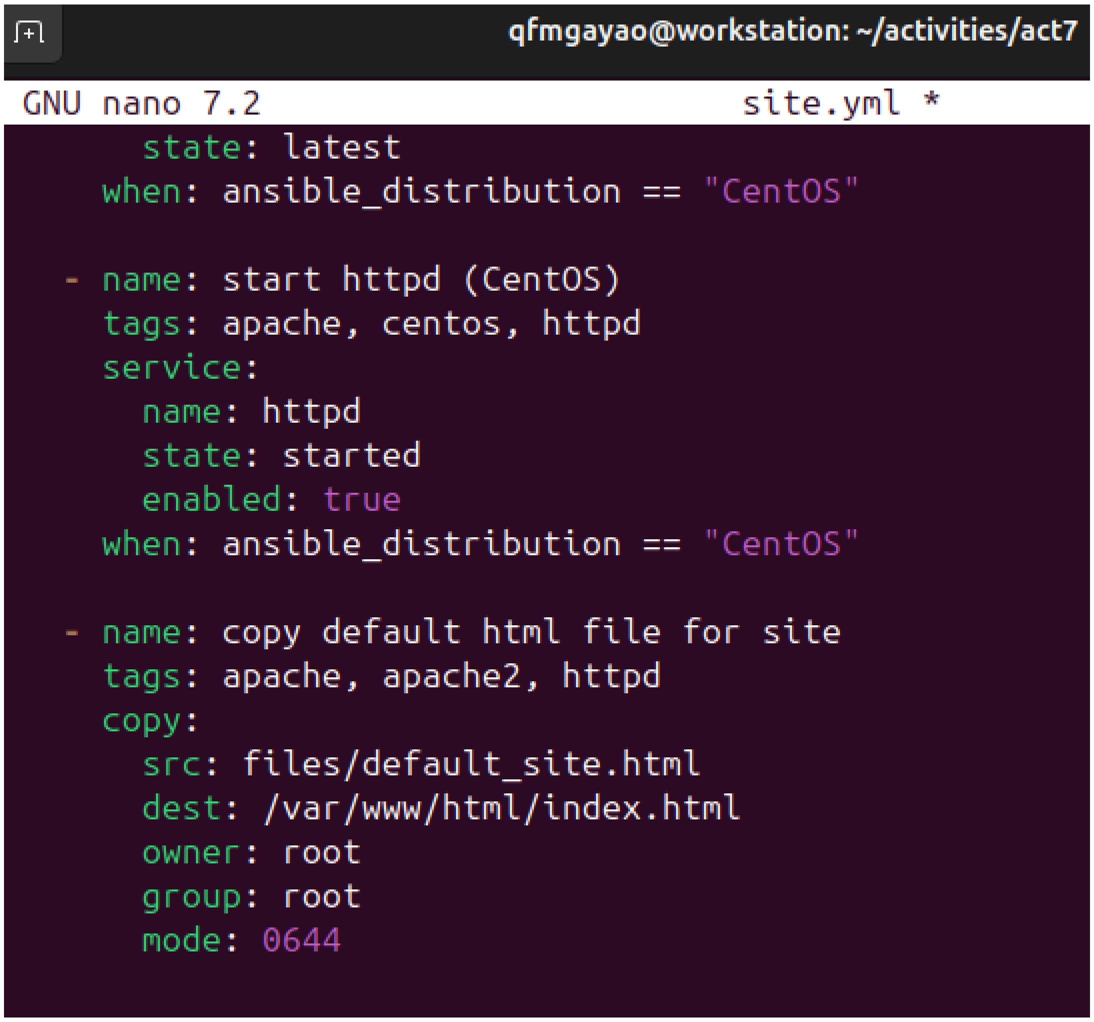
```
PooKYZZZ  added files                                d6fd5c3 · 7 minutes ago   4 Commits
files                         added files                             7 minutes ago
```



```
act7 / files /
PooKYZZZ  added files
Name
..
default_site.html
```

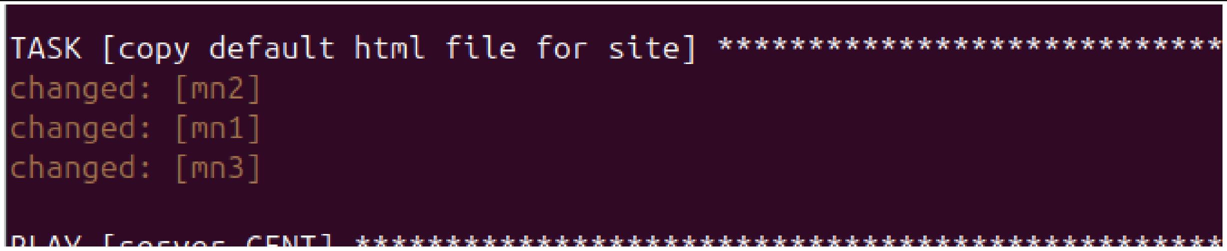
2. Edit the *site.yml* file and just below the *web_servers* play, create a new file to copy the default html file for site:
   - name: copy default html file for site

     tags: apache, apache2, httpd
     copy:
       src: default_site.html
       dest: /var/www/html/index.html
       owner: root
       group: root
       mode: 0644



```
qfmgayao@workstation: ~/activities/act7
GNU nano 7.2                         site.yml *
      state: latest
    when: ansible_distribution == "CentOS"

  - name: start httpd (CentOS)
    tags: apache, centos, httpd
    service:
      name: httpd
      state: started
      enabled: true
    when: ansible_distribution == "CentOS"

  - name: copy default html file for site
    tags: apache, apache2, httpd
    copy:
      src: files/default_site.html
      dest: /var/www/html/index.html
      owner: root
      group: root
      mode: 0644
```

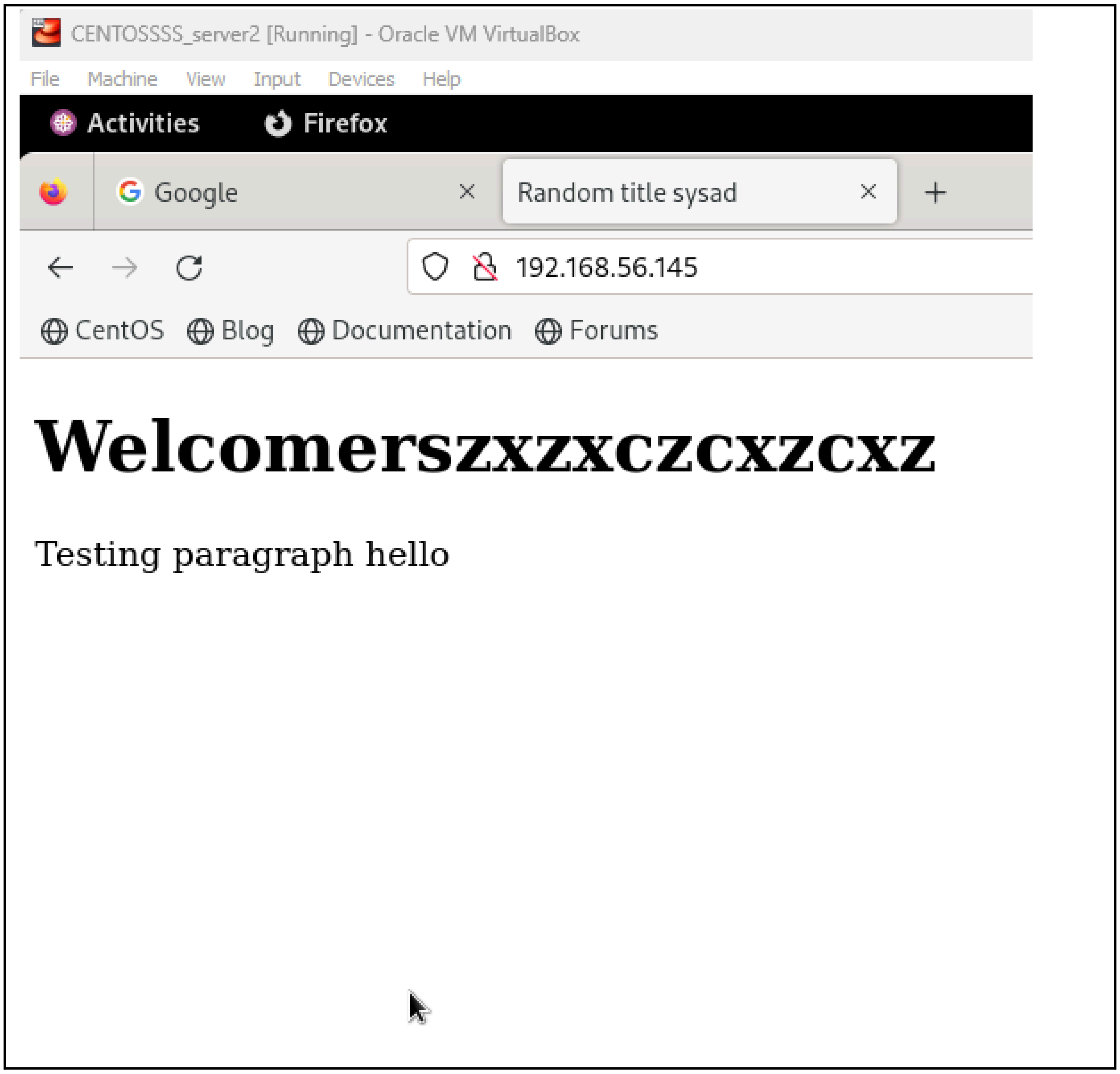
3. Run the playbook *site.yml*. Describe the changes.
   - editing my site.yml it includes that all these tags will be used and it will copy the source of my default html then it will output to all my servers included



```
TASK [copy default html file for site] ****************************
changed: [mn2]
changed: [mn1]
changed: [mn3]

PLAY [server CENT] ************************************************
```

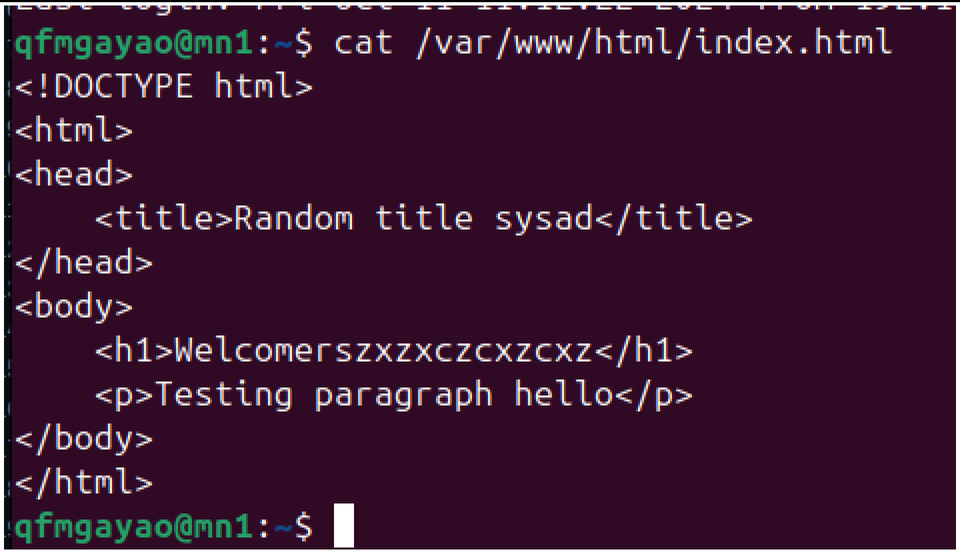
4. Go to the remote servers (*web_servers*) listed in your inventory. Use cat command to check if the index.html is the same as the local repository file (*default_site.html*). Do both for Ubuntu and CentOS servers. On the CentOS server, go to the browser and type its IP address. Describe the output.

- doing a cat command on my ubuntu servers, it shows that it successfully copied the html and inside the centOS it shows the updated html when inputting the IP address in the browser.


CENTOSSSS_server2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Firefox
Google
Random title sysad
192.168.56.145
CentOS Blog Documentation Forums

Welcomerszxzxczcxzcxz

Testing paragraph hello



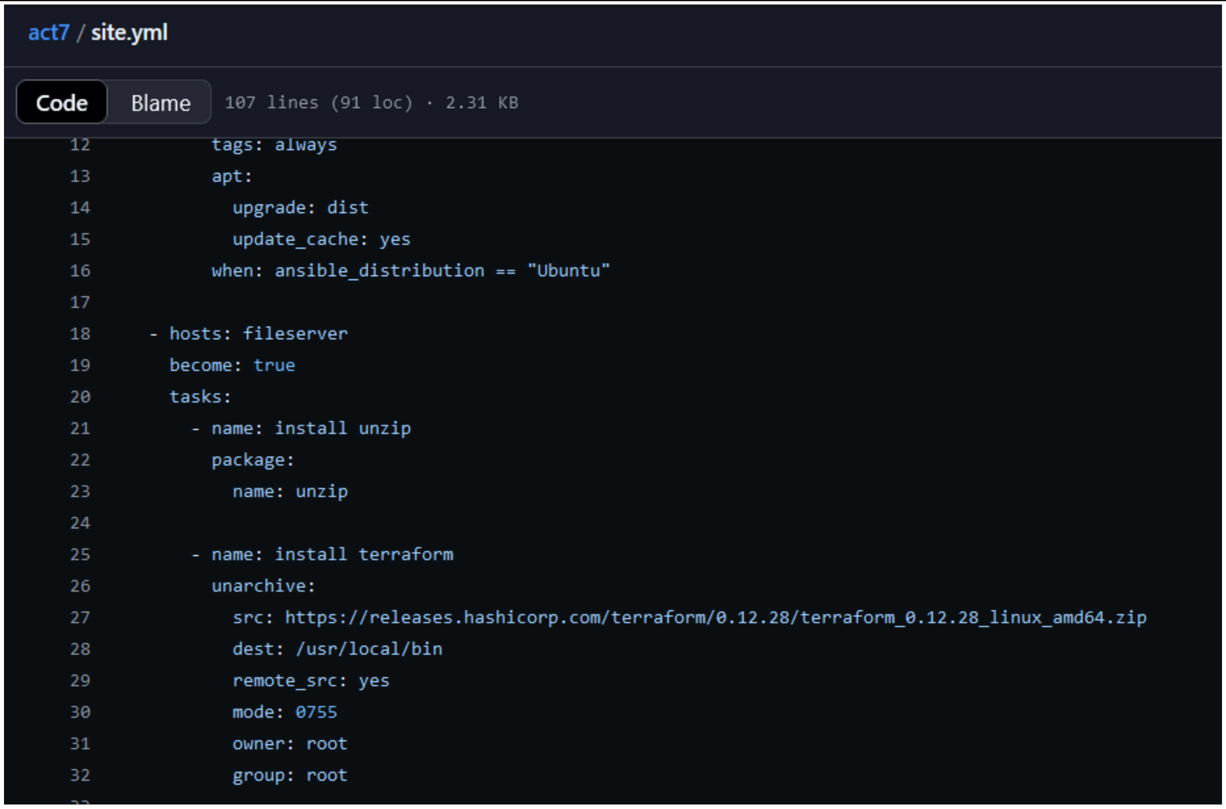
```
qfmgayao@mn1:~$ cat /var/www/html/index.html
<!DOCTYPE html>
<html>
<head>
    <title>Random title sysad</title>
</head>
<body>
    <h1>Welcomerszxzxczcxzcxz</h1>
    <p>Testing paragraph hello</p>
</body>
</html>
qfmgayao@mn1:~$
```

5. Sync your local repository with GitHub and describe the changes.
    - it updates all the sites.yml and my new directory files for my html.

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:

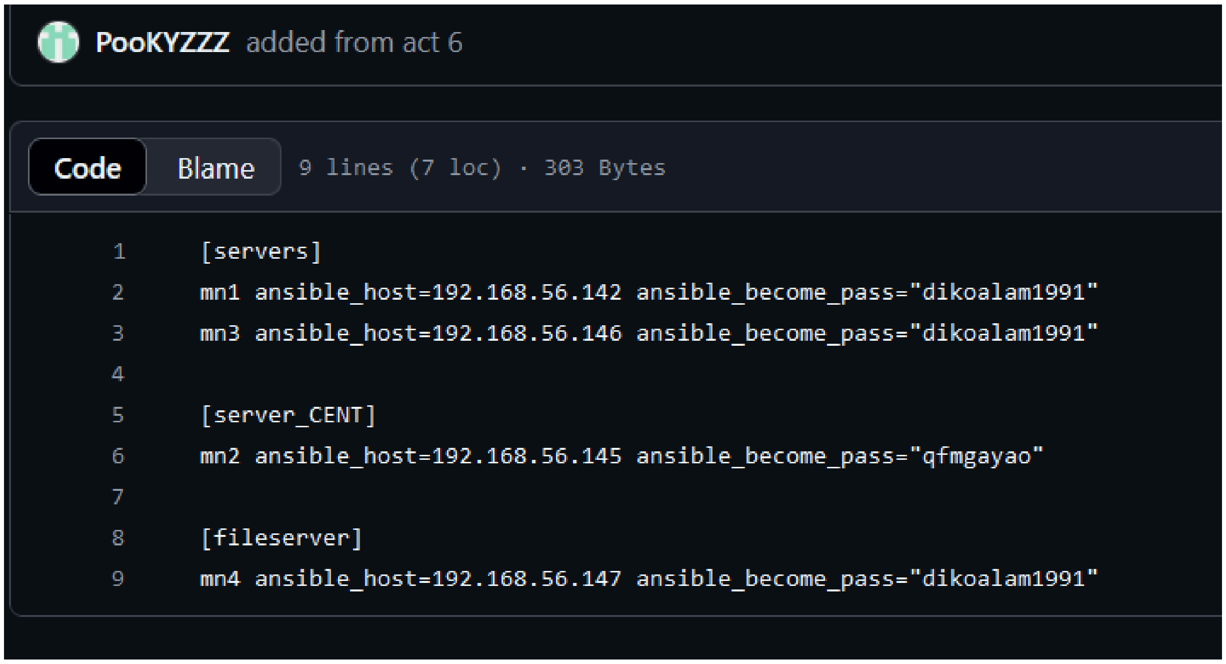
```
- hosts: workstations
  become: true
  tasks:

  - 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
```


act7 / site.yml

Code Blame 107 lines (91 loc) · 2.31 KB

```
        tags: always
        apt:
          upgrade: dist
          update_cache: yes
        when: ansible_distribution == "Ubuntu"

    - hosts: fileserver
      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
```

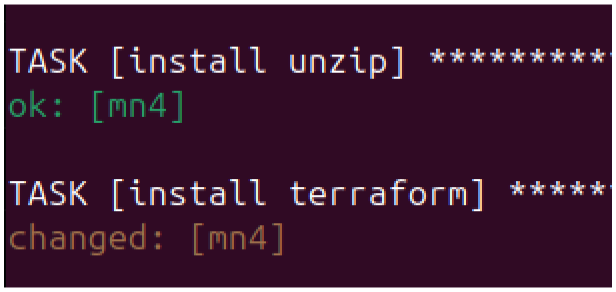
2. Edit the inventory file and add workstations group. Add any Ubuntu remote server. Make sure to remember the IP address.


PooKYZZZ added from act 6

Code Blame 9 lines (7 loc) · 303 Bytes

```
[servers]
mn1 ansible_host=192.168.56.142 ansible_become_pass="dikoalam1991"
mn3 ansible_host=192.168.56.146 ansible_become_pass="dikoalam1991"

[server_CENT]
mn2 ansible_host=192.168.56.145 ansible_become_pass="qfmgayao"

[fileserver]
mn4 ansible_host=192.168.56.147 ansible_become_pass="dikoalam1991"
```

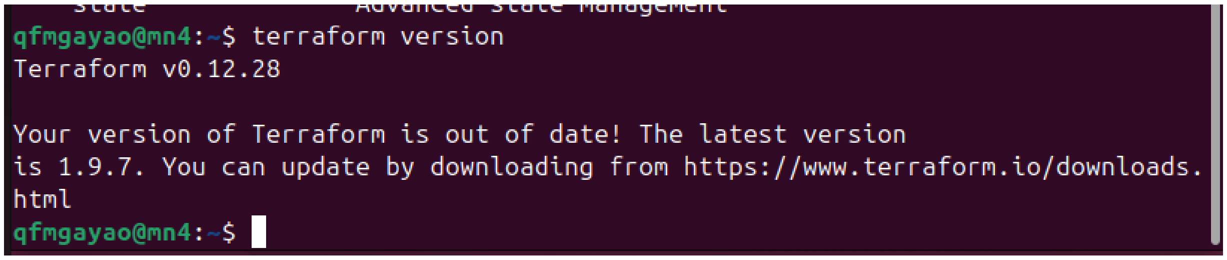
3. Run the playbook. Describe the output.



```
TASK [install unzip] *********
ok: [mn4]

TASK [install terraform] *****
changed: [mn4]
```

- here it shows that I successfully installed the unzip and terraform in my server 4.

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

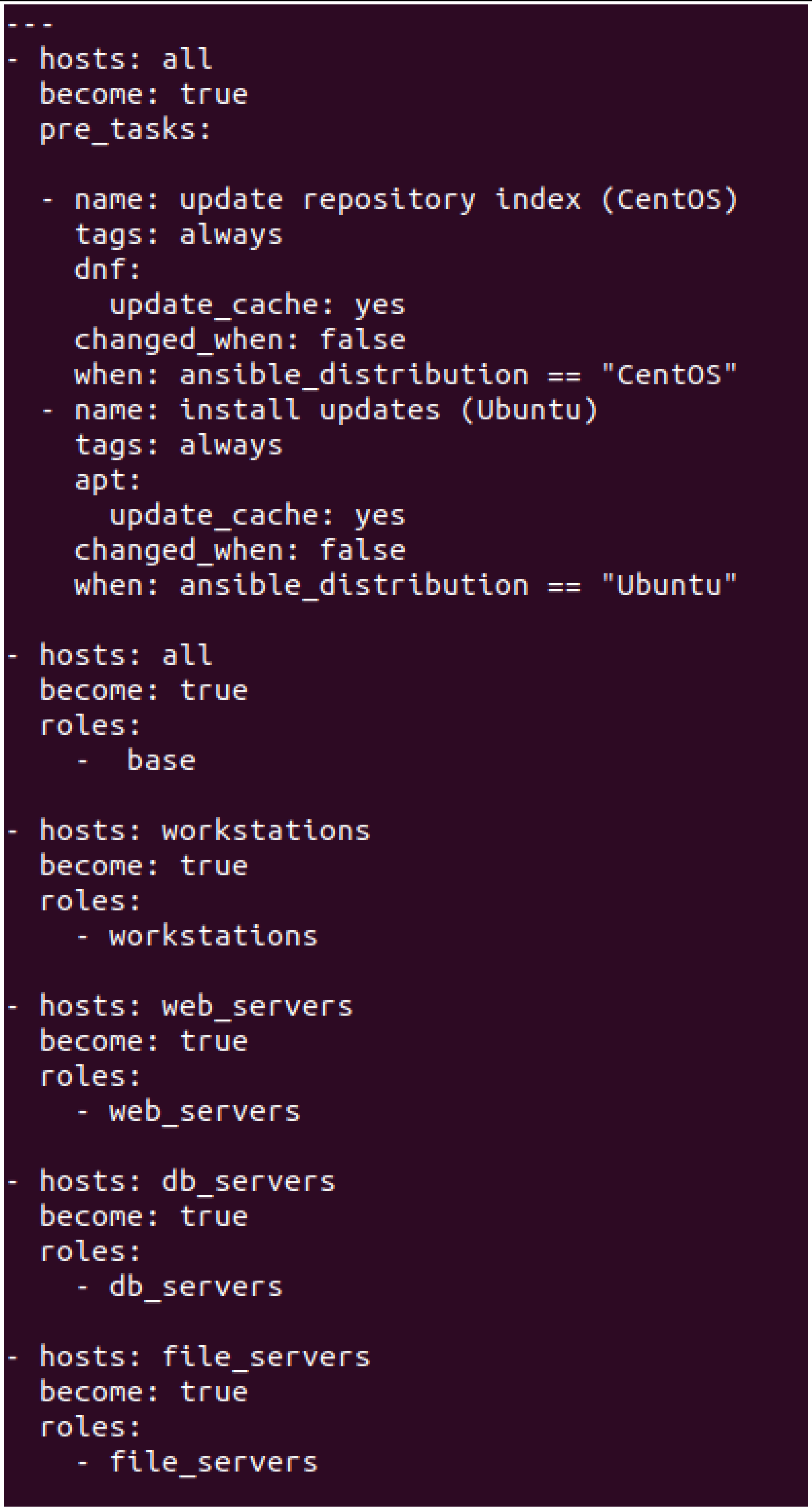


```
qfmgayao@mn4:~$ terraform version
Terraform v0.12.28

Your version of Terraform is out of date! The latest version
is 1.9.7. You can update by downloading from https://www.terraform.io/downloads.
html
qfmgayao@mn4:~$ 
```

- I manually check my server 4 to check whether the terraform is installed and in my output it shows that it's successfully installed but it’s an outdated version.

Task 3: Create roles

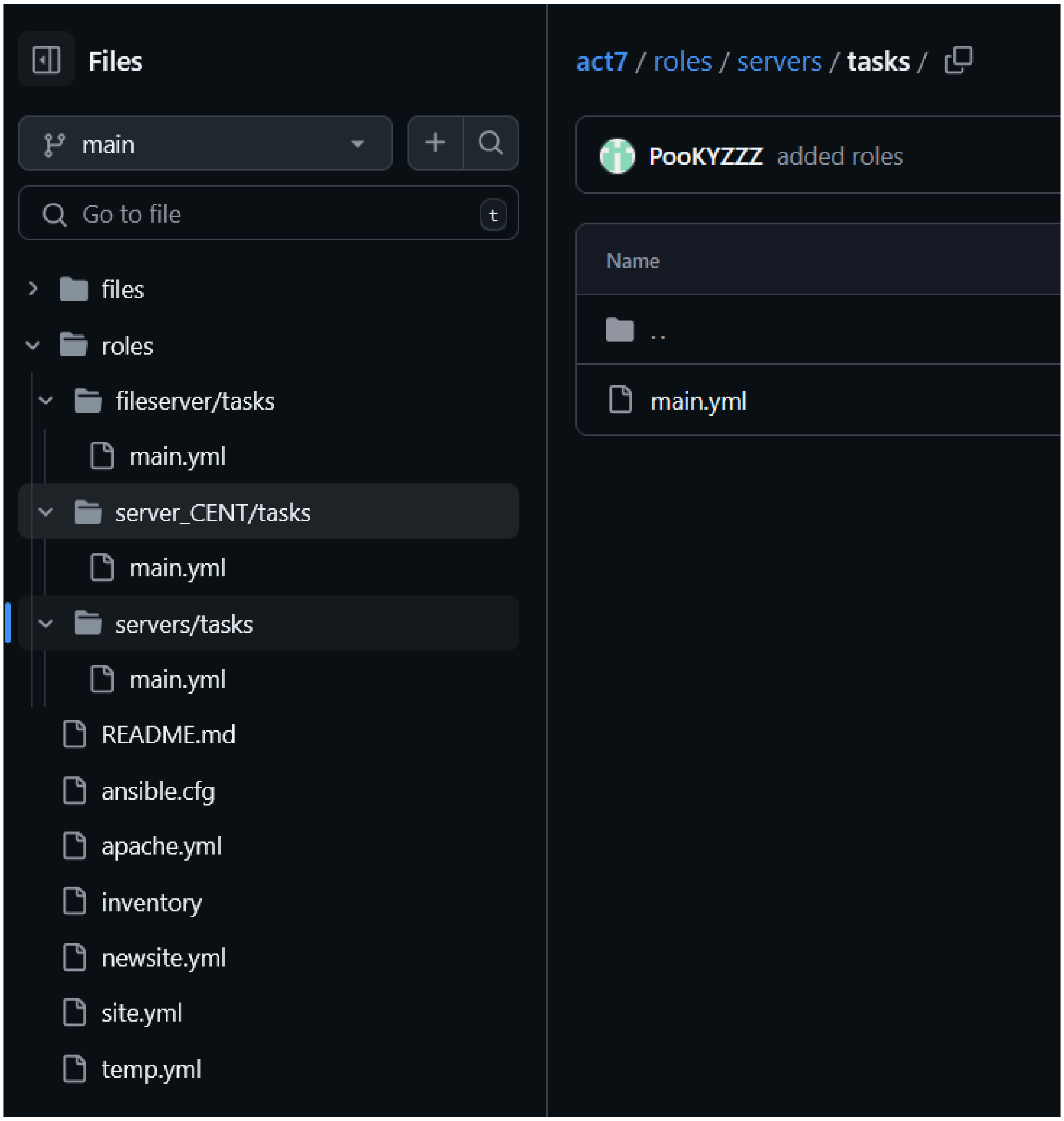
1. Edit the site.yml. Configure roles as follows: (make sure to create a copy of the old site.yml file because you will be copying the specific plays for all groups)



```
---
- 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.

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.

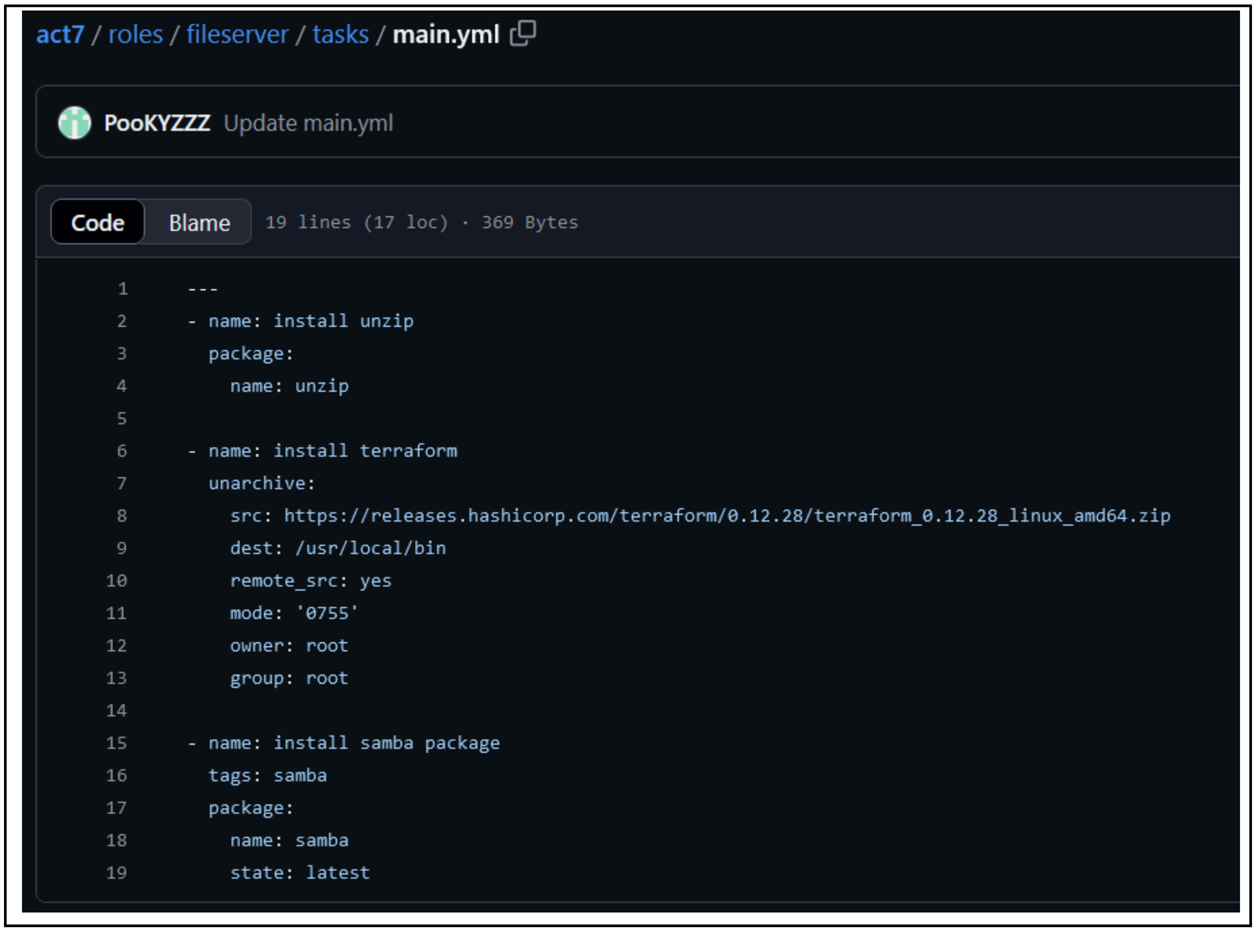


PooKYZZZ added roles

```
files
roles
  fileserver/tasks
    main.yml
  server_CENT/tasks
    main.yml
  servers/tasks
    main.yml
README.md
ansible.cfg
apache.yml
inventory
newsite.yml
site.yml
temp.yml
```

main.yml

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.

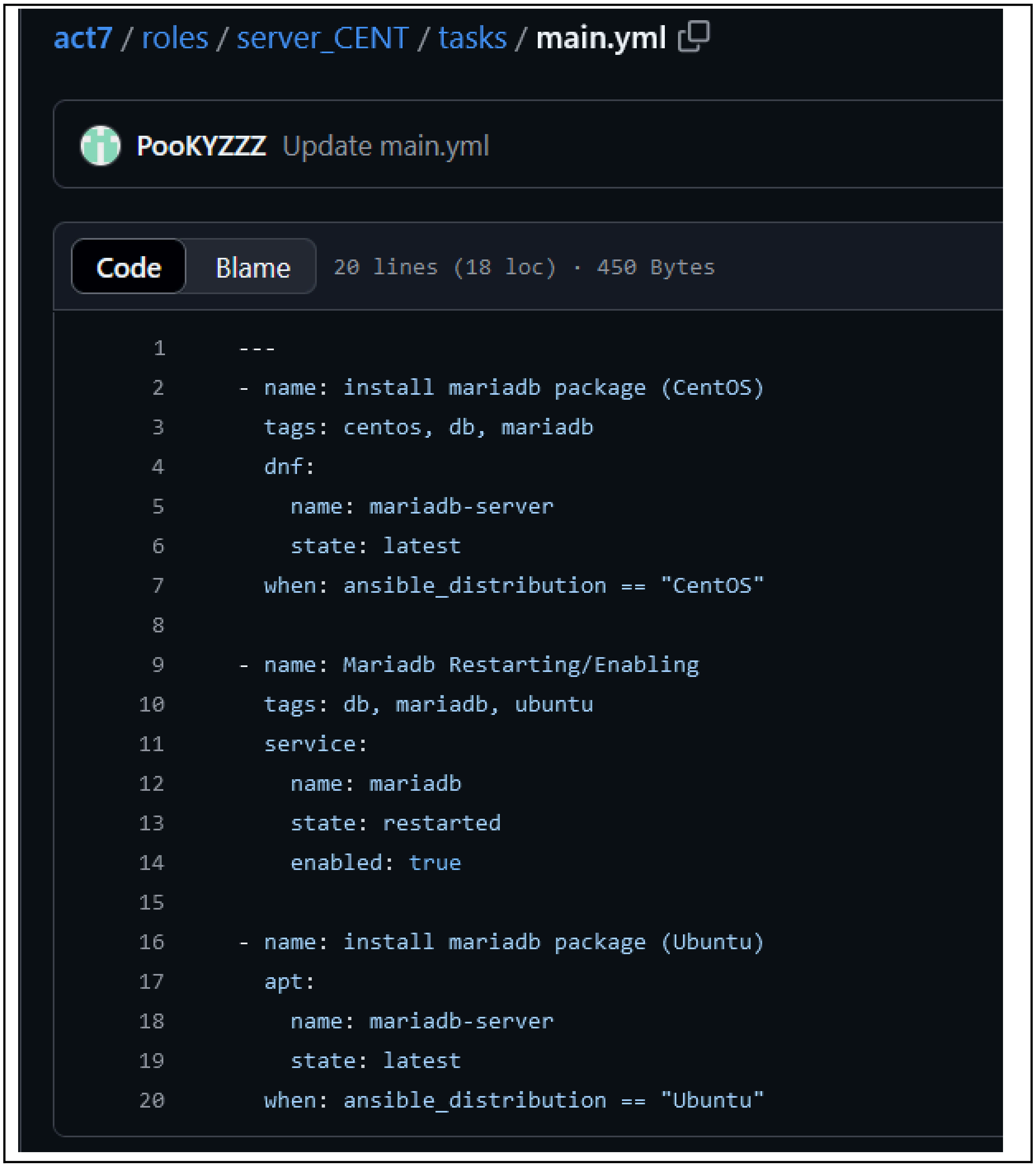


act7 / roles / fileserver / tasks / main.yml

PooKYZZZ Update main.yml

19 lines (17 loc) · 369 Bytes

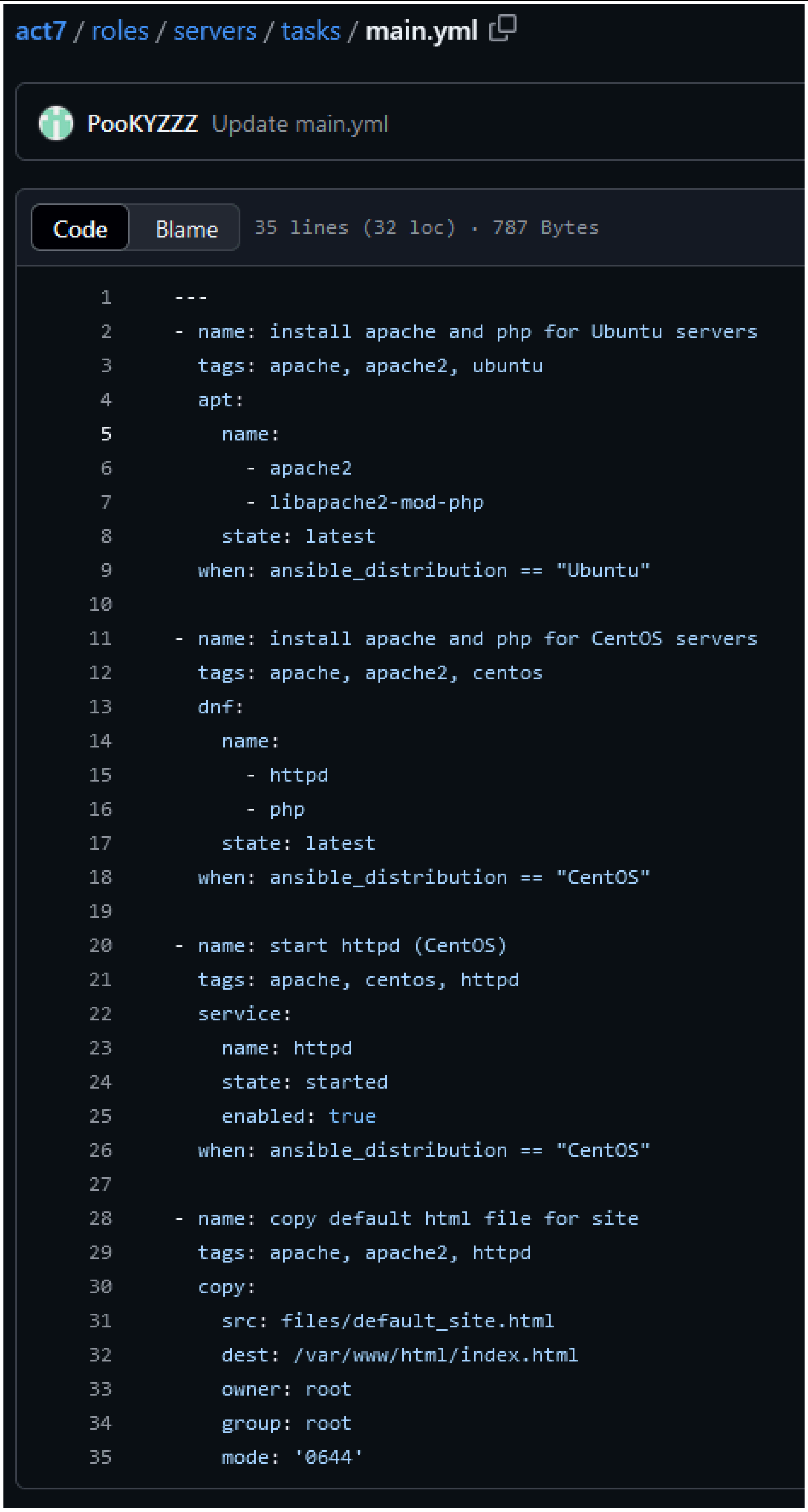
```yaml
---
- name: install unzip
  package:
    name: unzip

- 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

- name: install samba package
  tags: samba
  package:
    name: samba
    state: latest
```


act7 / roles / server_CENT / tasks / main.yml

PooKYZZZ Update main.yml

20 lines (18 loc) · 450 Bytes

```yaml
---
- name: install mariadb package (CentOS)
  tags: centos, db, mariadb
  dnf:
    name: mariadb-server
    state: latest
  when: ansible_distribution == "CentOS"

- name: Mariadb Restarting/Enabling
  tags: db, mariadb, ubuntu
  service:
    name: mariadb
    state: restarted
    enabled: true

- name: install mariadb package (Ubuntu)
  apt:
    name: mariadb-server
    state: latest
  when: ansible_distribution == "Ubuntu"
```

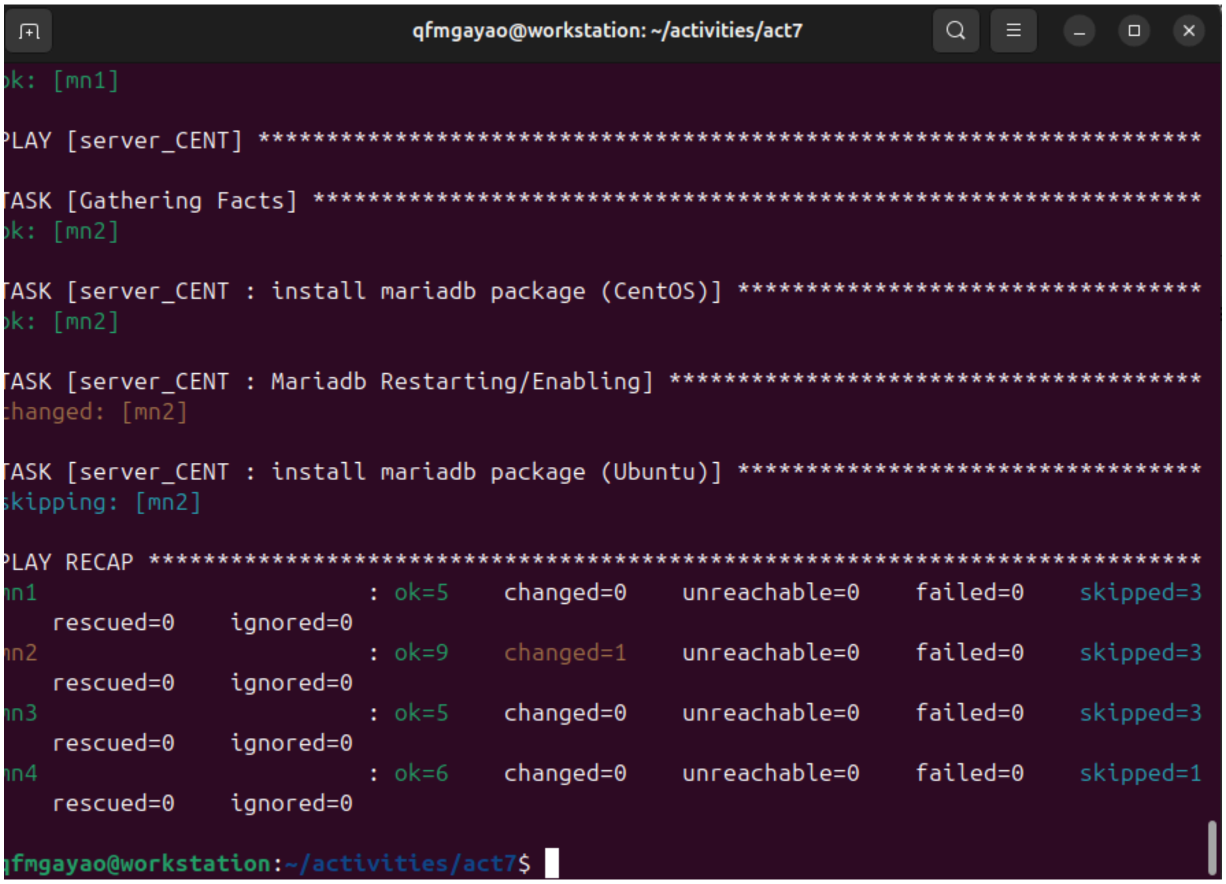

act7 / roles / servers / tasks / main.yml

PooKYZZZ Update main.yml

Code Blame 35 lines (32 loc) · 787 Bytes

```yaml
---
- name: install apache and php for Ubuntu servers
  tags: apache, apache2, ubuntu
  apt:
    name:
      - apache2
      - libapache2-mod-php
    state: latest
  when: ansible_distribution == "Ubuntu"

- name: install apache and php for CentOS servers
  tags: apache, apache2, centos
  dnf:
    name:
      - httpd
      - php
    state: latest
  when: ansible_distribution == "CentOS"

- name: start httpd (CentOS)
  tags: apache, centos, httpd
  service:
    name: httpd
    state: started
    enabled: true
  when: ansible_distribution == "CentOS"

- name: copy default html file for site
  tags: apache, apache2, httpd
  copy:
    src: files/default_site.html
    dest: /var/www/html/index.html
    owner: root
    group: root
    mode: '0644'
```

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



```
ok: [mn1]

PLAY [server_CENT] ***********************************************************************

TASK [Gathering Facts] *******************************************************************
ok: [mn2]

TASK [server_CENT : install mariadb package (CentOS)] ***********************************
ok: [mn2]

TASK [server_CENT : Mariadb Restarting/Enabling] ****************************************
changed: [mn2]

TASK [server_CENT : install mariadb package (Ubuntu)] ***********************************
skipping: [mn2]

PLAY RECAP *******************************************************************************
mn1                        : ok=5    changed=0    unreachable=0    failed=0    skipped=3
   rescued=0    ignored=0
mn2                        : ok=9    changed=1    unreachable=0    failed=0    skipped=3
   rescued=0    ignored=0
mn3                        : ok=5    changed=0    unreachable=0    failed=0    skipped=3
   rescued=0    ignored=0
mn4                        : ok=6    changed=0    unreachable=0    failed=0    skipped=1
   rescued=0    ignored=0

qfmgayao@workstation:~/activities/act7$
```

in this we created a roles that we can just call out for easy access, like if we have multiple servers, we can just assign them roles on what they do and just call out their yml in our playbook which shows that you can make the code much easier to read and easier to see the code errors if we have one.

Reflections:

Answer the following:

1. What is the importance of creating roles?

   - Roles in Ansible make it easier to organize and reuse tasks across different playbooks. Instead of repeating tasks, you just create a role once and apply it to any server group. Roles help keep things clean and modular, especially when managing complex setups with many servers. It's like splitting your tasks into small blocks so it's easier to manage. Plus, roles let you share your work with others and use common practices in different projects.

2. What is the importance of managing files?

   - Managing files in Ansible is important because it lets you control what files go to which server. For example, you can use it to deploy websites or configuration files. This ensures every server gets the right version of the file. Instead of manually copying files, Ansible does it automatically for you, saving time and making sure there are no mistakes.

Answer the following:

3. What is the importance of creating roles?

   - Roles in Ansible makes it easier for us to organize and reuse the tasks across different playbooks. Instead of repeating the tasks, we can just create a role once and make a group where if we have multiple servers and if they do the same tasks, we can just group them in a role. Roles help keep things clean and flexible, especially when managing complex setups with many servers. It's like splitting my tasks into small groups so it's easier to manage. Plus, roles let you share your work with others and use it as a common practice in different projects.

4. What is the importance of managing files?

   - Managing files in Ansible is very important because it lets us control what files can go inside our server. For example, we can use it to deploy our websites or configuration files. This ensures that every server gets the correct and latest version of the file, also instead of manually copying files, Ansible does it automatically which saves us time and making sure there are no mistakes.