**Lab: Ansible Roles**

**Introduction:**

**What are roles**?

Roles provide a framework for fully **independen**t or **interdependen**t collections of **files**, **tasks**, **templates**, **variables,** and **modules**.

The role is the primary mechanism for **breaking** a playbook into **multiple files**. This simplifies **writing complex** playbooks and makes them **easier to reuse**. The breaking of the playbook allows you to break the playbook into reusable components.

Each role is limited to a **particular functionality** or desired output, with all the necessary steps to provide that result either within the same role itself or in other roles listed as dependencies.

Roles are **not playbooks**. Roles are small **functionality** that can be used within the **playbooks independently**. Roles have **no specific** setting for which **hosts** the role will apply.

1. Login to the **Control node (ansi-master)** as **root** user with **password** as **linux.**
   1. The first step in creating a role is creating its directory structure. In order to create the base directory structure, we’re going to use a tool called **ansible-galaxy**:

# ansible-galaxy init roles/apache --offline

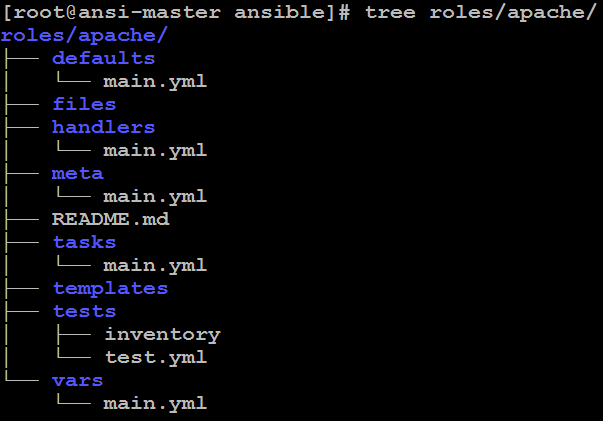
**Output**:



* 1. This command will create an **apache** directory with the following structure lets use the tree command to view

# tree roles/apache/

Output:



Note: A role’s directory structure consists of defaults, vars, files, handlers, meta, tasks, and templates.

* 1. Lets create the **install.yml** which will install the **httpd package** in the task directory

# cat > roles/apache/tasks/install.yml << EOF

---

- name: Install httpd Package

yum:

name: httpd

state: latest

EOF

* 1. Lets create the **configaure.yml** which perform configauration mangement in the task directory

# cat > roles/apache/tasks/configure.yml << EOF

---

- name: Copy httpd configuration file

copy:

src: files/httpd.conf

dest: /etc/httpd/conf/httpd.conf

- name: Copy index.html file

copy:

src: files/index.html

dest: /var/www/html

notify:

- restart apache

EOF

* 1. Lets create the service.yml which enable and start the httpd service in the task directory

# cat > roles/apache/tasks/service.yml << EOF

---

- name: Start and Enable httpd service

service:

name: httpd

state: restarted

enabled: yes

EOF

* 1. Lets break the codes even more as below using “import\_tasks” statements

# cat > roles/apache/tasks/main.yml << EOF

---

- import\_tasks: install.yml

- import\_tasks: configure.yml

- import\_tasks: service.yml

EOF

* 1. Lets copy the required files index.html to the files directory

# cat > roles/apache/files/index.html << EOF

This is a homepage created by Naushad for ansible roles

EOF

* 1. Lets docwnlo the required files (httpd.conf) to the files directory

# wget https://raw.githubusercontent.com/EyesOnCloud/ansible/main/httpd.conf

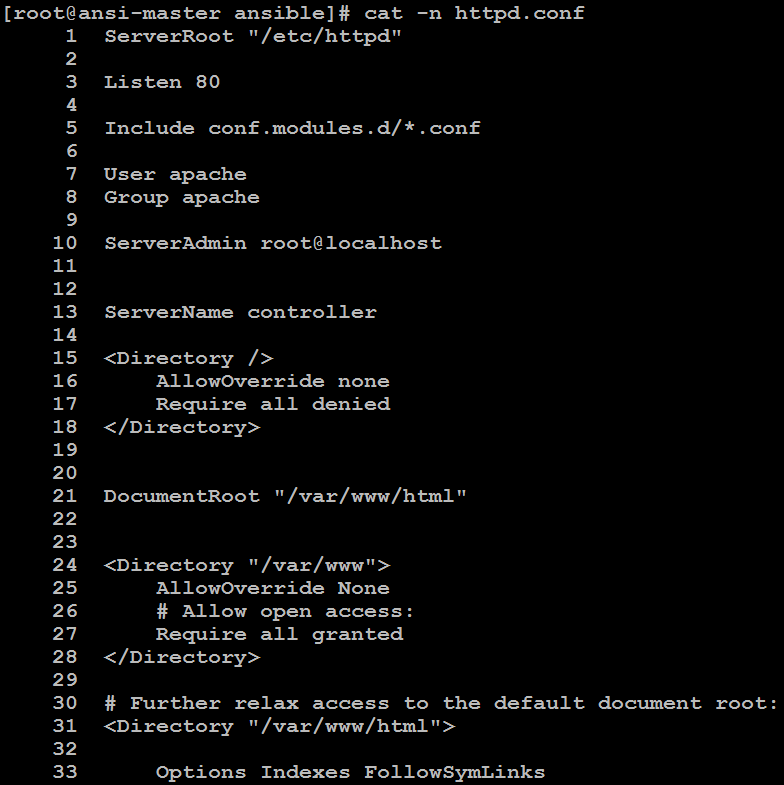
**Output**:



* 1. Lets veiw the https.conf

# cat -n httpd.conf

**Output**:



* 1. Lets move this httpd.conf to files

# mv httpd.conf roles/apache/files/

* 1. Lets edit the main.yaml to restart the server when there is a change. Because we have already defined it in the task with notify option. Use the same name “restart apache ”

# cat > roles/apache/handlers/main.yml << EOF

---

- name: restart apache

service:

name: httpd

state: restarted

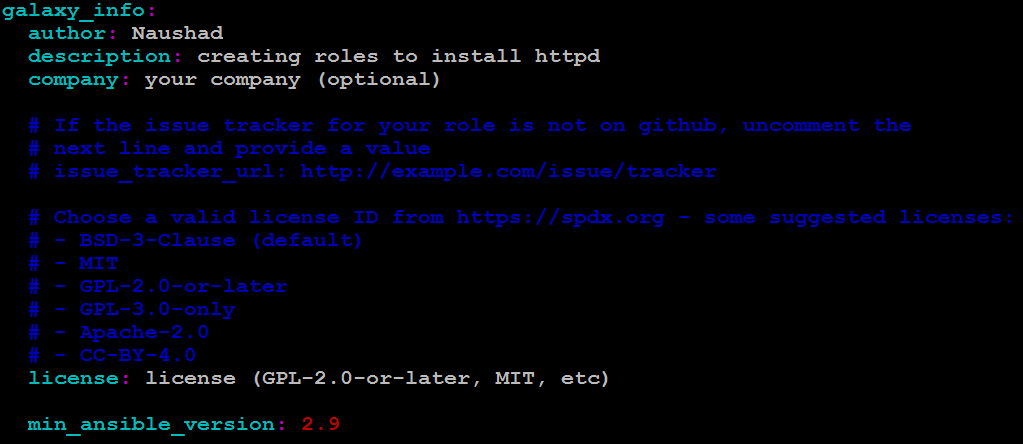
EOF

* 1. Lets edit the meta main.yaml to add the information about the roles like author,desription,license,platform supported

# vim roles/apache/meta/main.yml

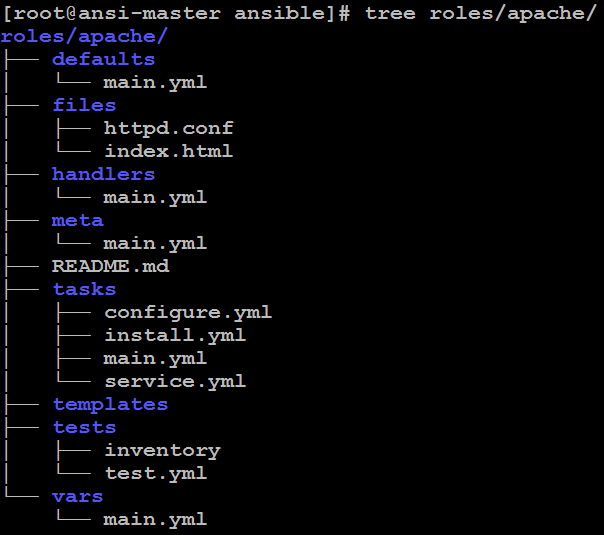
Output:





# tree roles/apache/

Output:



Note: Before further proceeding verify the files created

* 1. We have got the required files for Apache roles. Lets appply this role into the ansible playbook

“runsetup.yaml” as below to deploy it on the client nodes

# cat > runsetup.yml << EOF

---

- hosts: all

roles:

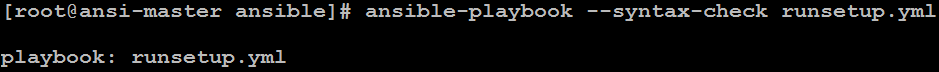
- apache

EOF

* 1. Let’s verify the any syntax error

# ansible-playbook --syntax-check runsetup.yml

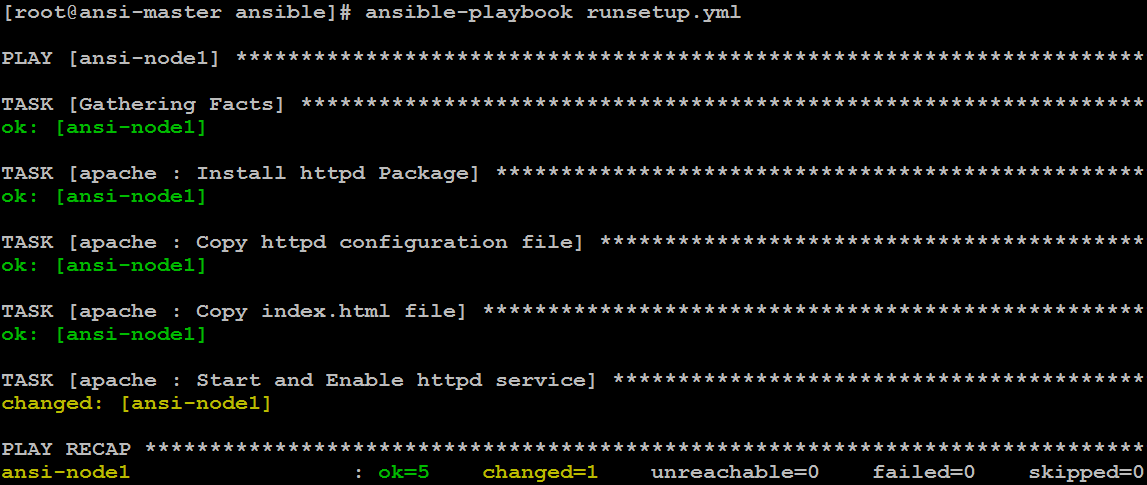
Output:



* 1. Let’s deploy the roles

# ansible-playbook runsetup.yml

Output:



Let’s verify the status httpd on client host(ansi-node1)

# ansible ansi-node1 -m command -a 'systemctl status httpd'

Let’s access the service through browser

# http://192.168.100.151

Output:

