#### **Project 01**

#### **Problem Statement**

You are tasked with deploying a three-tier web application (frontend, backend, and database) using Ansible roles. The frontend is an Nginx web server, the backend is a Node.js application, and the database is a MySQL server. Your solution should use Ansible Galaxy roles where applicable and define appropriate role dependencies. The deployment should be automated to ensure that all components are configured correctly and can communicate with each other.

#### **Steps and Deliverables**

1. **Define Project Structure**
   * Create a new Ansible project with a suitable directory structure to organize roles, playbooks, and inventory files.
2. **Role Selection and Creation**
   * Select appropriate roles from Ansible Galaxy for each tier of the application:
     + Nginx for the frontend.

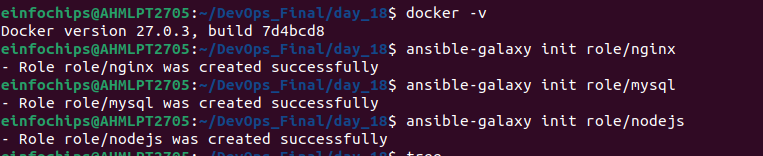
ansible-galaxy init role/nginx

* + - Node.js for the backend.

ansible-galaxy init role/nodejs

* + - MySQL for the database.

ansible-galaxy init role/mysql



1. **Dependencies Management**
   * Define dependencies for each role in the meta/main.yml file.

dependencies:

- role: mysql

when: ansible\_os\_family == "Debian"

# List your role dependencies here, one per line. Be sure to remove the '[]' above,

# if you add dependencies to this list.

* + Ensure that the roles have appropriate dependencies, such as ensuring the database is set up before deploying the backend.

1. **Inventory Configuration**
   * Create an inventory file that defines the groups of hosts for each tier (frontend, backend, database).
   * Ensure proper group definitions and host variables as needed.

[frontend]

web\_host ansible\_host=18.117\*\*\*\*\*ansible\_user=ubuntu ansible\_ssh\_private\_key\_file=/home/einfochips/.ssh/ansible-worker.pem

[backend]

app\_host ansible\_host=18.117.\*\*\*\*\*ansible\_user=ubuntu ansible\_ssh\_private\_key\_file=/home/einfochips/.ssh/ansible-worker.pem

[database]

db\_host ansible\_host=18.117\*\*\*\*\*ansible\_user=ubuntu ansible\_ssh\_private\_key\_file=/home/einfochips/.ssh/ansible-worker.pem

1. **Playbook Creation**
   * Create a playbook (deploy.yml) that includes and orchestrates the roles for deploying the application.
   * Ensure the playbook handles the deployment order and variable passing between roles

/home/einfochips/DevOps\_Final/project-directory/playbooks.

---

- hosts: database

become: yes

roles:

- database

- hosts: backend

become: yes

roles:

- appserver

- hosts: frontend

become: yes

roles:

- webserver

/home/einfochips/DevOps\_Final/project-directory/roles/appserver/tasks/main.yml

---

# tasks file for roles/appserver

- name: Install Node.js

apt:

name: nodejs

state: present

update\_cache: yes

- name: Ensure npm is installed

apt:

name: npm

state: present

/home/einfochips/DevOps\_Final/project-directory/roles/database/tasks/main.yml

---

# tasks file for roles/database

- name: Install MySQL server

apt:

update\_cache: yes

name: ['mysql-server','mysql-client','python3-mysqldb','libmysqlclient-dev']

state: present

become: yes

- name: Start MySQL service

service:

name: mysql

state: started

enabled: true

become: yes

- name: Create a MySQL user

mysql\_user:

name: "{{msql\_user}}"

password: "{{msql\_password}}"

priv: '\*.\*:ALL'

host: '%'

state: present

/home/einfochips/DevOps\_Final/project-directory/roles/webserver/tasks

---

# tasks file for roles/database

- name: Install MySQL server

apt:

update\_cache: yes

name: ['mysql-server','mysql-client','python3-mysqldb','libmysqlclient-dev']

state: present

become: yes

- name: Start MySQL service

service:

name: mysql

state: started

enabled: true

become: yes

- name: Create a MySQL user

mysql\_user:

name: "{{msql\_user}}"

password: "{{msql\_password}}"

priv: '\*.\*:ALL'

host: '%'

state: present

1. **Role Customization and Variable Definition**
   * Customize the roles by defining the necessary variables in group\_vars or host\_vars as needed for the environment.
   * Ensure sensitive data like database credentials are managed securely.

group\_vars

msql\_user: "myuser"

msql\_password: "mypassword"

1. **Testing and Validation**
   * Create a separate playbook for testing the deployment (test.yml) that verifies each tier is functioning correctly and can communicate with the other tiers.
   * Use Ansible modules and tasks to check the status of services and applications.
2. **Documentation**
   * Document the setup process, including any prerequisites, role dependencies, and how to run the playbooks.
   * Include a README.md file that provides an overview of the project and instructions for use.