Date: 22.07.2025
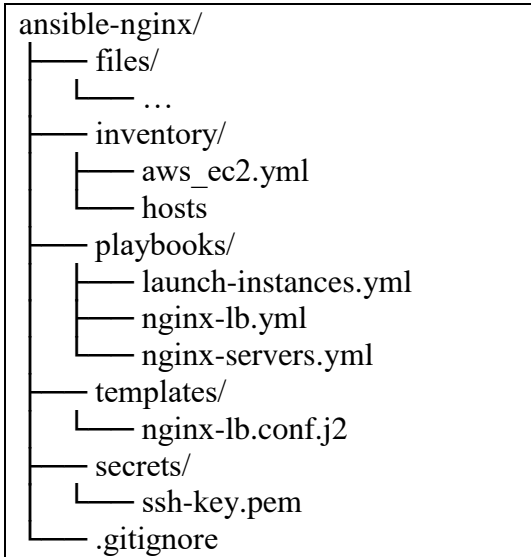
Ansible Case Study: #2

Using Ansible, launch 2 linux instances, configure them with nginx to server website, make the control node as a load balancer for the webservers

Contents

1. Directory Structure

```
ansible-nginx/
├── files/
│   └── …
├── inventory/
│   ├── aws_ec2.yml
│   └── hosts
├── playbooks/
│   ├── launch-instances.yml
│   ├── nginx-lb.yml
│   └── nginx-servers.yml
├── templates/
│   └── nginx-lb.conf.j2
├── secrets/
│   └── ssh-key.pem
└── .gitignore
```

2. Git Repository

This Git repository contains all playbooks and inventory files except ssh-key.pem
aryan-madhavi/ansible-nginx

3. Setting up vite-react-app

$ cd files
$ npx create-vite@latest
Follow through the setup

4. Commands and other notes

Ansible installation and configuration refer 'Ansible Case Study #1'

Create the template for nginx loadbalancer configuration

Run the playbooks:
1. Launch instances: ansible-playbook -i inventory/hosts playbooks/launch-instances.yml
2. Check if ec2 instances are dynamically plugged in properly: ansible-inventory -i inventory/aws_ec2.yml --graph
3. Set up EC2 webservers by installing Nginx, Node.js, Git, clone a React app repo then build and deploy it to Nginx and ensure Nginx is running: ansible-playbook -i inventory/aws_ec2.yml playbooks/nginx-servers.yml
4. Set up an Nginx load balancer on localhost by installing and starting Nginx, create a fallback maintenance page, deploy the load balancer config, and reload Nginx as needed: ansible-playbook -i inventory/hosts playbooks/nginx-lb.yml

Check functionality:
1. Get the DNS of the load balancer machine
2. Open browser, paste the DNS then append the port 8080 to it
3. It should show the default vite-react-app

Verify maintenance fallback page:
1. Terminate the webservers instances via the AWS console
2. Reload the Page in the browser (DNS with port 8080)
3. It should show the message "Oops, we are under maintenance"