TASKS

Task 1.

Two Ubuntu instances were created.

- One as ansible controller and the other ansible node.
- On the controller, ansible was installed via apt-get
- The hosts file was edited with these

  ```
  [node_servers]
  node1 ansible_host=52.91.184.97

  [all:vars]
  ansible_python_interpreter=/usr/bin/python3
  ansible_ssh_private_key_file=/root/doctorly.pem
  ```

- Then a playbook was created to install the basics for docker on the node.

```
---
- hosts: all
  become: true
  tasks:
    - name: Install aptitude
      apt:
        name: aptitude
        state: latest
        update_cache: true

    - name: Install required system packages
      apt:
        pkg:
          - apt-transport-https
          - ca-certificates
          - curl
          - software-properties-common
          - python3-pip
          - virtualenv
          - python3-setuptools
        state: latest
        update_cache: true
```

```
    - name: Add Docker GPG apt Key
      apt_key:
        url: https://download.docker.com/linux/ubuntu/gpg
        state: present

    - name: Add Docker Repository
      apt_repository:
        repo: deb https://download.docker.com/linux/ubuntu focal stable
        state: present

    - name: Update apt and install docker-ce
      apt:
        name: docker-ce
        state: latest
        update_cache: true

    - name: Install Docker Module for Python
      pip:
        name: docker
    - name: Install Docker Compose
      apt:
        name: docker-compose-plugin
        state: latest
        update-cache: true
```

- Dockerfile was Created for the asp.net application
- A compose file was also created for the whole stack. Using MySQL too.

TASK 2:

> How would you structure your Terraform project if you have multiple environments
> and use different cloud providers?

- This is the point where terraform becomes an orchestrator. using the same workflow to manage multiple providers.

b) Terraform file attached (***main.tf***)

TASK 3:

First step in monitoring production resources is to establish threshold/baseline. I.e defining what normal looks like and critical and important services that make up the system and if services are dependent on each other.

This will inform what % of CPU, disk io, and/or memory utilisation is optimal. Then triggers can be set for these services and metrics using any monitoring tool the team is comfortable or familiar with. E.g Cloudwatch, netdata, nagios, etc.

Files Attached in Git.

- Dockerfile
- Docker-compose
- Playbook
- Terraform.