CREATE CI PIPELINE THAT WILL DEPLOY AND CONFIGURE THESE VMS USING TERRAFORM AND ANSIBLE

→ ACCORDING FOLLOWING REQUIREMENTS

- Deploy 2 virtual machines using terraform.

first vm on Amazon linux, hostname: c8.local

second vm on ubuntu 21.04, hostname: u21.local

- 2. As a result of terraform execution, dynamically create inventory for ansible

c8.local should be in the frontend group

u21.local should be in the backend group

- 3. Create ansible playbook for c8.local and u21.local

for linux OS playbook should apply the following changes

selinux: disable

firewalld: disable

- 4.for frontend playbook group should install and configure nginx

nginx configuration should do proxying from port 80 on port 19999 to the

backend group

- 5.for the backend group, the playbook must install the Netdata application from the

official repositories and run it on port 19999.

→CREATE A ANSIBLE SERVER ON AWS 

→LOGIN TO SERVER THROUGH SSH AND INSTALL ANSIBLE THROUGH VISUAL STUDIO

```
PS C:\Users\syedf\Desktop> ssh -i "ohio.pem" ec2-user@ec2-18-119-103-199.us-east-2.compute.amazonaws.com
```

→INSTALL ANSIBLE AND TERRAFORM ON YOUR EC2 INSTANCE

```
sudo yum update -y
sudo amazon-linux-extras install ansible2 -y
ansible --version
sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo
https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
```

→CHECK FOR VERSION

```
[ec2-user@ip-172-31-11-120 ~]$ terraform --version
Terraform v1.10.2
on linux_amd64
```


```
[ec2-user@ip-172-31-11-120 ~]$ terraform --version
Terraform v1.10.2
on linux_amd64
```


→DOWNLOAD AWS CLI AND CONFIGURE IT

```
[ec2-user@ip-172-31-11-120 ~]$ aws configure
AWS Access Key ID [None]: AKIAVVZPCK2JLYC6PJME
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-2
Default output format [None]: json
```

→ Generate an SSH Key Pair

We'll generate a new SSH key pair to use for your Terraform-managed instances.

ssh-keygen -t rsa -b 2048 -f ~/.ssh/pipeline_key -N ""

→CREATE A TERRAFORM DIRECTORY AND main.tf FILE

```
mkdir  terraform_project
cd  terraform_project
vi main.tf
```

→ADD THE FOLLOWING TEMPLATE WHICH WILL CREATE TWO INSTANCE ONE WITH LINUX IMAGE AND ONE WITH UBUNTU IMAGE AND GIVE THE PUBLIC IP AS OUTPUT

```
provider "aws" {
  region = "us-east-2"
}

resource "aws_key_pair" "pipeline" {
  key_name   = "pipeline"
  public_key = file("~/.ssh/pipeline_key.pub")
}

resource "aws_instance" "c8_linux" {
  ami                  = "ami-088d38b423bff245f"
  instance_type        = "t2.micro"
  key_name             = aws_key_pair.pipeline.key_name
  associate_public_ip_address = true
  tags = {
    Name = "c8_linux"
  }

  user_data = <<-EOF
    #!/bin/bash
    hostnamectl set-hostname c8.local
  EOF
}

resource "aws_instance" "u21_ubuntu" {
  ami                  = "ami-00eb69d236edcfaf8"
  instance_type        = "t2.micro"
  key_name             = aws_key_pair.pipeline.key_name
  associate_public_ip_address = true
  tags = {
    Name = "u21_ubuntu"
  }

  user_data = <<-EOF
    #!/bin/bash
    hostnamectl set-hostname u21.local
  EOF
}

output "amazon_linux_public_ip" {
  value = aws_instance.c8_linux.public_ip
}

output "ubuntu_public_ip" {
  value = aws_instance.u21_ubuntu.public_ip
}
```

→RUN THE FOLLOWING COMMANDS

 # terraform init

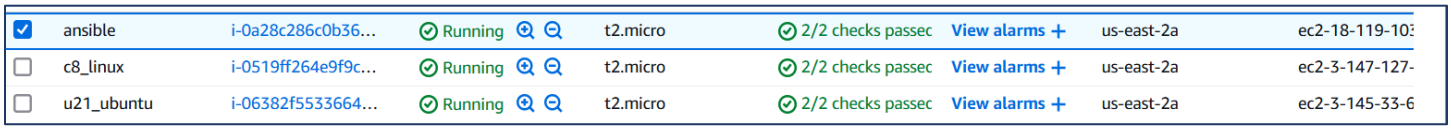
 # terraform plan

 # terraform apply

```
Outputs:

amazon_linux_public_ip = "3.147.127.28"
ubuntu_public_ip = "3.145.33.66"
```

→checking on aws console if instance is ok!



→ NOW GET THE OUTPUT WHICH ARE PUBLIC IP'S OF THE CREATED SERVER AND ADD TO ANSIBLE INVENTORY

terraform output **<copy the ip's>**

cd /etc

sudo chown -R ec2-user:ec2-user ansible

vi /etc/ansible/hosts

```
[frontend]
Public_ip ansible_user=ec2-user ansible_ssh_private_key_file=~/.ssh/pipeline_key

[backend]
Public_ip ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/pipeline_key
```

cat /etc/ansible/hosts

```
[frontend]
3.147.127.28 ansible_user=ec2-user ansible_ssh_private_key_file=~/.ssh/pipeline_key

[backend]
3.145.33.66 ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/pipeline_key
```

→Ensure SELinux and Firewalld are Disabled

→MAKE SURE CONNECTIVITY IS ESTABLISHED

```
[ec2-user@ip-172-31-11-120 ~]$ ansible all -m ping
3.145.33.66 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform linux on host 3.147.127.28 is using the discovered Python interpreter at /usr/bin/python, but future installation of another Python interpreter could
change this. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
3.147.127.28 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

vi playbook.yml

```
---
- name: Configure Amazon Linux and Ubuntu servers
  hosts: all
  become: yes

  tasks:
    - name: Disable SELinux on Amazon Linux
      when: ansible_distribution == "Amazon"
      command: setenforce 0
      ignore_errors: yes

    - name: Disable firewalld on Amazon Linux
      when: ansible_distribution == "Amazon"
      service:
        name: firewalld
        state: stopped
        enabled: no

    - name: Disable firewalld on Ubuntu
      when: ansible_distribution == "Ubuntu"
      service:
        name: ufw
        state: stopped
        enabled: no
```

ansible-playbook playbook.yml –syntax-check

```
[ec2-user@ip-172-31-11-120 ~]$ ansible-playbook playbook.yml --syntax-check

playbook: playbook.yml
```

ansible-playbook playbook.yml

```
PLAY RECAP *********************************************************************************************************************************************
3.145.33.66                : ok=2    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
3.147.127.28               : ok=3    changed=1    unreachable=0    failed=0    skipped=1    rescued=0    ignored=1
```

→Install and Configure Nginx on the Frontend (Amazon Linux)

→CREATE TEMPLATE THAT ENSURES NGINX PROXIES REQUESTS FROM PORT 80 TO PORT 19999 ON THE BACKEND SERVER.

mkdir template

cd template

vi nginx.conf.j2

```
worker_processes 1;

events {
   worker_connections 1024;
}

http {
   upstream backend_group {
      {% for host in groups['backend'] %}
      server {{ host }}:19999;
      {% endfor %}
   }

   server {
      listen 19999;

      location / {
         proxy_pass http://backend_group;
         proxy_set_header Host $host;
         proxy_set_header X-Real-IP $remote_addr;
         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
         proxy_set_header X-Forwarded-Proto $scheme;
      }
   }
}
```

→CREATE A PLAYBOOK TO INSTALL NGINX AND PROXY ITS PORT

vi nginx.yml

```
---
- name: Configure frontend servers (Amazon Linux)
  hosts: frontend
  become: yes

  tasks:
   - name: Add the official Nginx repository
     get_url:
       url: http://nginx.org/packages/centos/7/x86_64/RPMS/nginx-1.22.1-1.el7.ngx.x86_64.rpm
       dest: /opt/nginx.rpm

   - name: Install Nginx from the downloaded RPM
     yum:
       name: /opt/nginx.rpm
       state: present

   - name: Configure nginx proxy
     template:
       src: templates/nginx.conf.j2
       dest: /etc/nginx/nginx.conf

   - name: Start and enable nginx
     service:
       name: nginx
       state: started
       enabled: yes
```

ansible-playbook nginx.yml

→Install and Run Netdata on the Backend (Ubuntu)

vi netdata.yml

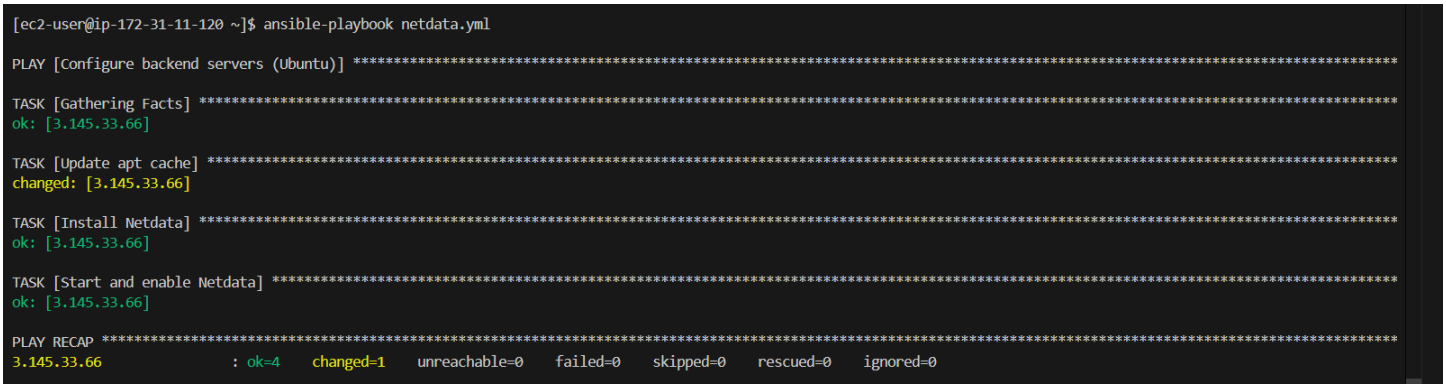
```
---
- name: Configure backend servers (Ubuntu)
  hosts: backend
  become: yes

  tasks:
    - name: Update apt cache
      apt:
        update_cache: yes

    - name: Install Netdata
      apt:
        name: netdata
        state: present

    - name: Start and enable Netdata
      service:
        name: netdata
        state: started
        enabled: yes
```

ansible-playbook netdata.yml



*__Amazon Linux (Frontend Server)__: Nginx is installed, configured, and running to proxy requests from port 80 to port 19999 on the backend server.

__Ubuntu (Backend Server)__: Netdata is installed and running on port 19999.*

----------setup-completed----------

<INSTANCE RESTARTED SO THE PUBLIC IP GOT CHANGED>

→CHECKING ON BROWSER GET THE PUBLIC IP OF NGINX AND SEARCH BY PORT 19999←

Public IPv4 address
3.149.237.241 | open address

< http://3.149.237.241:19999 >