Terraform and Ansible Multi-Environment Setup Documentation

Project Overview

This project demonstrates how to set up a multi-environment infrastructure using Terraform for provisioning and Ansible for configuration management. It covers the creation of infrastructure for development, staging, and production environments, focusing on automation, scalability, and best practices.

Installation Steps

- 1. Installing Terraform on Ubuntu -

Update the Package List:

```
bash
     sudo apt-get update
```

- Install Dependencies:

```
bash
     sudo apt-get install -y gnupg software-properties-common
```

- Add HashiCorp's GPG Key:

```
bash
     curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/hashicorp-archive-keyring.gpg
```

- Add the HashiCorp Repository:

```
bash
     echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee
/etc/apt/sources.list.d/hashicorp.list
```

- Install Terraform:

```
bash
     sudo apt-get update && sudo apt-get install terraform
```

- Verify the Installation:

```
bash
     terraform --version
```

- 2. Installing Ansible on Ubuntu

- Add the Ansible PPA:

```
bash
     sudo apt-add-repository ppa:ansible/ansible
```

- Update the Package List:

```
bash
     sudo apt update
```

- Install Ansible:

```
bash
     sudo apt install ansible
```

- Verify the Installation:

```
bash
     ansible --version
```

Directory Structure

- Create directories for Terraform and Ansible:

```
bash
  mkdir  && cd
  mkdir terraform
  mkdir ansible
```

- Verify the Directory Structure:

```
/
  ├── terraform/
  └── ansible/
```

Setting Up Infrastructure with Terraform
- 1. Create the Infrastructure Directory -

Navigate to the Terraform Directory:

```
bash
     cd terraform
```

- Create the infra Directory:

```
bash
     mkdir infra && cd infra
```

- 2. Create and Populate Terraform Files

- Create the following files: - `bucket.tf` (S3 Bucket Configuration) - `dynamodb.tf` (DynamoDB Table for State Locking) - `ec2.tf` (EC2 Instance Configuration) - `output.tf` (Output Definitions) - `variable.tf` (Variable Declarations)

- 3. Final Directory Structure

```
infra/
   ├── bucket.tf
   ├── dynamodb.tf
   ├── ec2.tf
   ├── output.tf
   └── variable.tf
```

Running Terraform Commands

- Initialize Terraform:

```
bash
  terraform init
```

- Review the plan:

```
bash
  terraform plan
```

- Apply the changes:

```
bash
  terraform apply
```

Setting Up Ansible

- 1. Creating Dynamic Inventories
- Navigate to the Ansible Directory:

```
bash
     cd ../ansible
```

- Create the Inventories Directory:

```
bash
     mkdir -p inventories/dev inventories/prod inventories/stg
```

- 2. Creating Playbook for Installing Nginx

- Create the playbooks Directory:

```
bash
     mkdir playbooks
```

- Create the `install_nginx_playbook.yml` file.

- 3. Initializing Roles for Nginx -

Navigate to the playbooks Directory:

```
bash
     cd playbooks
```

- Initialize the nginx-role:

```
bash
     ansible-galaxy role init nginx-role
```

Script for Updating Inventories

- Create the `update_inventories.sh` script to dynamically update inventory files based on Terraform outputs.

Final Notes

- Ensure to secure your private key:

```
bash
  chmod 400 devops-key
```

- SSH into EC2 instances:

```
bash


  ssh -i devops-key ubuntu@
```

Thank You

mahirdasare@gmail.com