Multi-Cloud Auto Deployment using Terraform (AWS Free Tier)

Introduction

This project demonstrates automated infrastructure provisioning using Terraform on AWS Free Tier. The main goal is to deploy a web server that is automatically configured and accessible via a public IP, showcasing Infrastructure as Code (IaC) principles. The setup helps understand how Terraform can manage cloud resources efficiently and consistently with minimal manual intervention.

Abstract

The Multi-Cloud Auto Deployment project focuses on automating the deployment of a web application using Terraform. It provisions cloud resources in AWS and installs NGINX to host a simple web page. The project validates auto-deployment by ensuring that all resources are created and configured with a single Terraform command. This approach simulates a multi-cloud environment using local DNS for route simulation and cloud-based EC2 servers for hosting.

Tools Used

- Terraform – for infrastructure automation and provisioning
- AWS Free Tier – to host EC2 instances for deployment
- NGINX – lightweight web server for hosting a simple web page
- DNSMasq – local DNS simulator for multi-cloud routing (optional)
- Ubuntu Server – base OS for Terraform and AWS instances

Steps Involved in Building the Project

- 1. Launched an Ubuntu EC2 instance to act as the Terraform server.
- 2. Installed Terraform, AWS CLI, and Git for infrastructure management.
- 3. Configured Terraform provider with AWS credentials and region.
- 4. Created Terraform scripts to provision an EC2 instance and install NGINX using user_data.
- 5. Applied Terraform configuration using ‘terraform apply -auto-approve’ to deploy resources.
- 6. Verified deployment by accessing the web page served by NGINX using the public IP address.
- 7. (Optional) Simulated multi-cloud routing using DNSMasq or /etc/hosts configuration.

Conclusion

The project successfully demonstrated automated provisioning and configuration of cloud resources using Terraform. By deploying and validating a live NGINX web server, it proved the efficiency of Infrastructure as Code for managing scalable, repeatable, and consistent cloud deployments. The setup provides a foundation for future multi-cloud integrations and advanced automation scenarios.