

PROG 8870 -- Final Project: Deploying AWS Infrastructure with Terraform

Title: AWS Infrastructure Automation with Terraform

Overview

In this project, students will utilize Terraform to create a scalable AWS infrastructure. The project involves:

- Creating an S3 bucket for object storage.
- Launching multiple EC2 instances for compute needs.
- Applying best practices for Infrastructure as Code (IaC).

Project Scope

You will be tasked with deploying AWS resources using Terraform, ensuring your infrastructure is modular, reusable, and well-documented. Additionally, you will demonstrate your project and explain your code, infrastructure design, and implementation decisions.

Tasks and Deliverables

1. S3 Bucket Setup

- Create an **S3 bucket** to store your Terraform state file (tfstate).
- Enable **versioning** on the S3 bucket to track changes in your state file over time.
- (Optional, Extra Credit): Add a **DynamoDB table** for state locking to avoid concurrent state updates.

2. VPC and EC2 Instance

- Set up an **EC2 instance** in a VPC:
 - Define a **custom VPC** with at least one **public subnet**.
 - Assign a **public IP** to the EC2 instance.
 - Configure a **security group** to allow:
 - SSH access on port 22.
 - HTTP/HTTPS traffic on ports 80 and 443 (optional for extra credit).
 - Use a variable to specify the **AMI ID** and **instance type** (e.g., t2.micro for free-tier users).

3. Terraform Backend Configuration

- Configure your Terraform backend to use the created **S3 bucket** for storing state files.
- Ensure the backend configuration is modular and easy to manage.

4. Variables in tfvars File

- Use a variables.tf file to define all necessary variables (e.g., region, bucket name, instance type).
- Store the actual values in a terraform.tfvars file to separate configurations from the main codebase.

5. GitHub Repository

- Push your Terraform code to a GitHub repository.
- Your repository must include:
 - main.tf (Terraform configuration)
 - variables.tf (Variables definition)
 - terraform.tfvars (Variables values; sensitive data should not be pushed to GitHub)
 - backend.tf (Backend configuration)
 - README.md (Documentation)
- Share your GitHub repository URL along with the submission document.

6. Presentation/Demo

- Prepare a **5-10 minute presentation** explaining:
 - Your code structure and implementation.
 - The AWS infrastructure you created.
 - Key features or challenges you encountered.
 - How your Terraform code ensures modularity and best practices.
 - **Live Demo:**
 - Run your Terraform configuration (terraform init, terraform plan, terraform apply).
 - Show the resources created in the AWS Management Console (e.g., the S3 bucket, EC2 instance, VPC, etc.).
 - Demonstrate the use of your tfvars file and backend configuration.
-

Submission Details

- **Due Date:** December 06, 2024 11:59pm EST
- **Submission Items: One document containing below items**
 1. GitHub repository link.
 2. Screenshot(s) showing:
 - Running EC2 instance.
 - Terraform apply output in the terminal.
 - S3 bucket showing the tfstate file.
 3. Clear and concise documentation in the README.md (in github repo)

Assessment Criteria (Total: 30 points)

Weightage in Final grade: 35%

1. **Functionality** (10 points)
 - Are all resources deployed correctly?
 - Is the Terraform state file stored in the S3 bucket?
2. **Best Practices** (5 points)
 - Use of variables and tfvars files.
 - Proper backend configuration.
3. **Documentation** (5 points)
 - Is the README.md clear and comprehensive?
 - Can a third party replicate the setup using your documentation?
4. **Presentation/Demo** (10 points)
 - Was the presentation well-structured and informative?
 - Did the demo showcase the infrastructure and code effectively?
 - Were challenges and solutions explained clearly during presentation in the class?

Resources

- **Terraform Documentation:** <https://registry.terraform.io/>
- **AWS Free Tier:** <https://aws.amazon.com/free/>
- **GitHub Guides:** <https://guides.github.com/>