**Terraform Implementation**

This repository contains scripts and Terraform files to deploy application infrastructure on AWS for Django App. Below is an overview of the architecture and services utilized in this deployment.

**Architecture Overview**

The deployment is based on a microservices architecture, leveraging various AWS services for scalability, reliability, and security

**Services Utilized**

* Virtual Private Cloud (VPC): Configured with public and private subnets spanning two availability zones for enhanced availability and fault tolerance.
* Internet Gateway: Enables connectivity between VPC instances and the wider internet.
* Security Groups: Acts as a network firewall mechanism to control traffic to and from instances.
* Availability Zones: Utilized for increased system reliability and fault tolerance.
* Public Subnets: Host infrastructure components like the NAT Gateway and Application Load Balancer.
* EC2 Instance Connect Endpoint: Facilitates secure connections to assets within both public and private subnets.
* Identity and Access Management (IAM): Used to create roles allowing EC2 instances to access and communicate with data files stored in S3.
* Amazon Relational Database Service (RDS): Provides database service for the application.
* Private Subnets: Hosts web servers (EC2 instances) and databases (RDS instances) for enhanced security.
* NAT Gateway: Allows instances in private subnets to access the internet.
* Application Load Balancer (ALB): Distributes web traffic evenly to an Auto Scaling Group of EC2 instances across multiple Availability Zones.
* Auto Scaling Group: Automatically manages EC2 instances for availability, scalability, fault tolerance, and elasticity.
* Simple Notification Service (SNS): Configured to alert about activities within the Auto Scaling Group.
* Amazon S3: Stores application code, migration scripts, and logs

**Deployment steps**

1. Access the modules in their folders with their filenames indicated.
2. Create terraform.tfvars with sensitive values (see example below):

db\_password = "SuperSecret!"

public\_key\_path = "~/.ssh/id\_rsa.pub" # optional

s3\_bucket\_name = "my-django-app-static-logs-2025"

1. Initialize & apply:

terraform init

terraform plan -out plan.tfplan

terraform apply "plan.tfplan"