INDEPENDENT HANDS-ON PROJECT

Highly Available Three-Tier AWS Platform

HTTPS ingress, private compute, Auto Scaling and isolated PostgreSQL

Portfolio track	Independent Hands-on Engineering
Evidence level	Terraform fmt, init and validate passed
Author	Saurabh Dindokar

Evidence boundary: This is a public-safe personal implementation lab. It is not client production work, and no live deployment is claimed.

Overview

A Terraform design for an HTTPS Application Load Balancer, private EC2 Auto Scaling application tier and private PostgreSQL RDS database tier distributed across two Availability Zones.

Engineering Problem

A traditional web workload needs a clear public entry point without exposing application servers or the database. The design must support replacement of unhealthy instances, controlled instance refresh, encrypted storage, private administration and environment-specific database protection.

Architecture

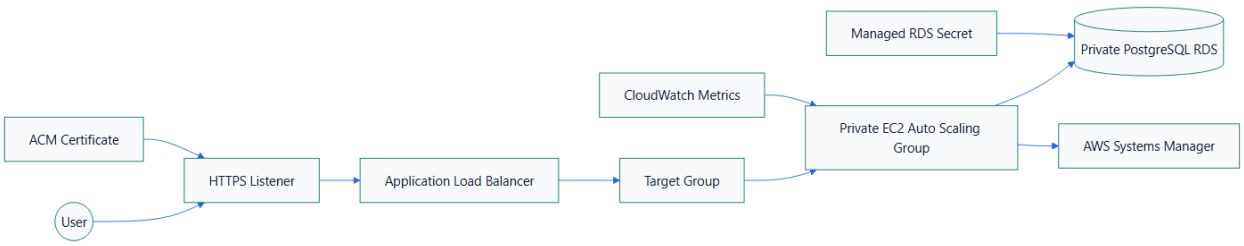

Figure 1. Highly Available Three-Tier AWS Platform architecture

What I Implemented

- Placed the internet-facing ALB in public subnets and application instances in private subnets.
- Created an EC2 launch template with IMDSv2, encrypted storage, Systems Manager permissions and no public address.
- Configured an Auto Scaling Group with ELB health checks, target tracking and rolling instance refresh.
- Provisioned private, encrypted PostgreSQL RDS with managed master credentials and environment-specific backup and deletion controls.
- Restricted database access to the application security group instead of a CIDR-wide rule.

Important Technical Decisions

- The ALB is the only public application entry point.
- HTTPS requires an ACM certificate input; the design avoids presenting HTTP as the production endpoint.
- Instances use Systems Manager instead of SSH ingress.
- The bootstrap page tests ALB-to-instance routing but deliberately does not pretend to be a database-backed application.

Security Controls

- IMDSv2 is required.
- RDS is not publicly accessible and accepts PostgreSQL only from the application security group.
- RDS storage is encrypted and credentials are managed by AWS rather than embedded in user data.

Reliability and Operations

- ALB and Auto Scaling span two Availability Zones.
- ELB health checks replace unhealthy instances.
- Rolling instance refresh preserves healthy capacity during launch-template updates.

Cost and Cleanup Guardrails

- ALB, NAT Gateway, RDS and public IPv4 usage are explicitly called out before apply.
- Development and production examples use different sizing and protection values.
- The teardown checklist includes snapshots, ALB, NAT Gateways, EIPs and ENIs.

Validation Evidence

The following local checks passed on 2026-08-13:

```
terraform fmt -check -recursive
terraform init -backend=false
terraform validate
```

Scope boundary: Format, initialization and validation passed on 2026-08-13. No plan, deployment, ALB health check or RDS connection is claimed. The sample bootstrap application does not connect to the database.

Delivery and Verification Runbook

- 1. Validate network and certificate inputs
- 2. Review ALB, NAT and RDS cost
- 3. Review Terraform plan
- 4. Test target health and SSM access
- 5. Confirm the database is private
- 6. Test instance refresh and rollback
- 7. Destroy and inspect residual resources

Technology Stack

Terraform | Amazon EC2 | Auto Scaling | Application Load Balancer | Amazon RDS | PostgreSQL | IAM | Systems Manager | CloudWatch

Key Learnings

- Infrastructure evidence must distinguish code validation, plan review and live deployment.
- Security, reliability, cost and cleanup decisions should be documented before apply.
- A useful platform lab includes verification and rollback thinking, not only resource declarations.