CaseGuard Studio - DevOps Architecture & Documentation

1. Architecture Overview

This infrastructure solution leverages the AWS Cloud Development Kit (CDK) in C# to provision a scalable and secure multi-region deployment pipeline for a containerized API. The architecture is modular, resilient, and enables continuous delivery with production safeguards.

The solution adheres to the principles of infrastructure-as-code (IaC), decoupling resources into isolated stacks across networking, compute, and data layers, and ensuring DRY and reusable constructs.

2. High-Level Infrastructure

- VPC: Two isolated VPCs per region (USEast, USWest), provisioned using default VPCs to bootstrap quickly.

- Subnets: Public subnets for ECS Fargate and RDS to maintain connectivity during demonstration. In production, RDS should be moved to private subnets.

- ECS (Fargate): Stateless microservice hosted behind an Application Load Balancer, automatically provisioned with scalable compute.

- Load Balancer: Public-facing ALB configured for ECS service ingress with listener rules.

- RDS (PostgreSQL): Managed DB instance with SecretsManager-based credentials, encryption at rest, and VPC subnet group.

- CDK: All infrastructure is deployed via CDK constructs in C#, with Stack separation (NetworkStack, ComputeStack, DatabaseStack).

3. CI/CD Pipeline Strategy

CI/CD leverages GitHub Actions. The pipeline is triggered on `push` to `main` and performs:

- CDK bootstrap (once)

- Dependency installation (Node.js 20+, .NET 8.0 SDK)

- CDK deployment to `staging` in USEast region automatically
- Manual approval step before deployment to `production` in USWest region using `environment` with reviewer protection.

Secrets used for deployment (stored in GitHub):

- AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, AWS_REGION

4. Region & Environment Configuration

Each environment is deployed to a distinct AWS region to simulate real-world regional isolation:

- Staging: us-east-1
- Production: us-west-2

Environment-specific stacks are created in App.cs and named accordingly (StagingComputeStack, ProdComputeStack, etc.).

This ensures deterministic, isolated deployments across lifecycle stages.

5. Security and Observability

- Secrets Manager handles credentials for RDS.
- Public IPs are used in staging; for production, restrict access via security groups and use private subnets.
- Logging is handled by CloudWatch via ECS task definition. Further observability can be enhanced using AWS X-Ray and structured application logs.

6. Limitations and Recommendations

- Node.js 18 is deprecated: Upgrade to Node.js 20 or 22 in GitHub Actions.

- No monitoring alerts (e.g., CloudWatch alarms or SNS notifications).
- No centralized config management (e.g., SSM Parameter Store).
- Further security hardening (e.g., WAF, Shield Advanced, IAM boundary policies).
- CI/CD enhancements: Blue/Green or Canary deployment strategies.

7. Conclusion

This implementation presents a production-ready baseline for secure, scalable multi-region infrastructure using AWS CDK and GitHub Actions. It is modular, highly reusable, and extensible for larger enterprise-scale workloads.

The architecture follows industry best practices in IaC, CI/CD, environment isolation, and security. It can be evolved further with service mesh, cost optimization layers, and improved governance using AWS Config or Control Tower.