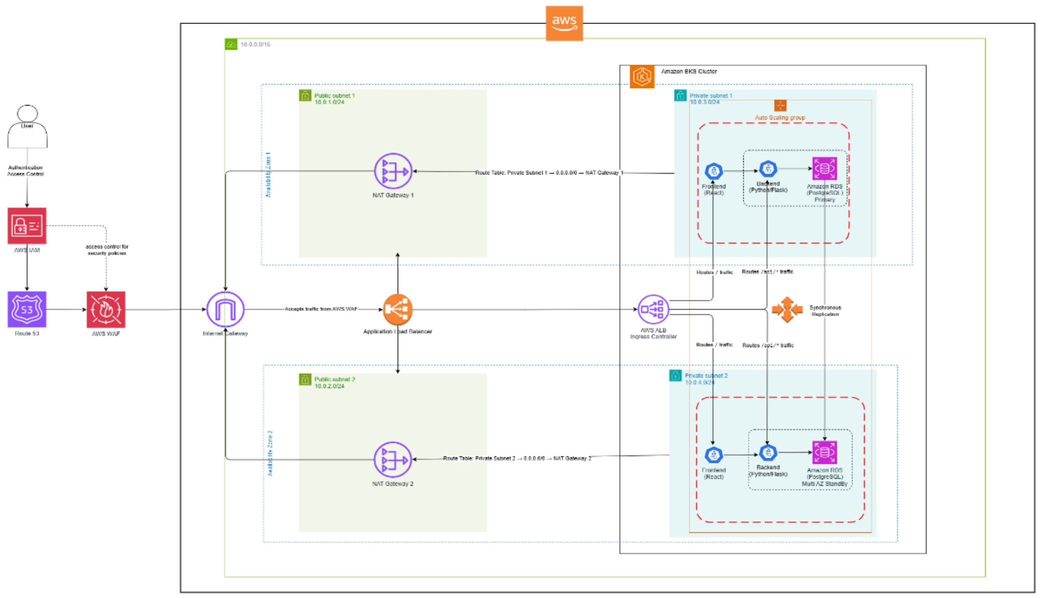
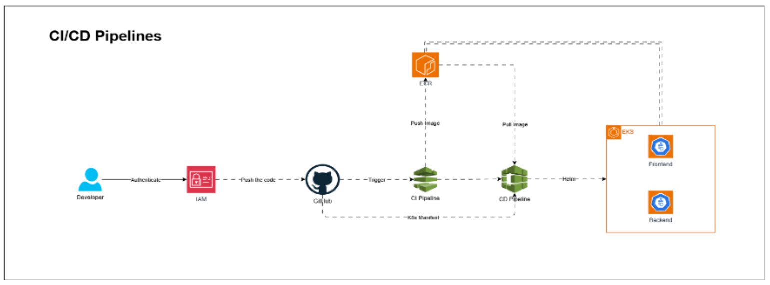
Innovate Inc. AWS Cloud Infrastructure Design Document





1. Introduction

Innovate Inc. is deploying a web application using AWS-managed Kubernetes (EKS) following best practices for scalability, security, and cost-effectiveness. This document outlines the architectural design, including cloud environment structure, network design, compute platform, database configuration, and CI/CD integration.

2. Cloud Environment Structure

AWS Accounts and Management

To follow best practices for isolation, billing, and security, we recommend using three AWS accounts:

- **Development Account** – Used for testing and staging environments to prevent impact on production.
- **Production Account** – Hosts the live web application with strict security controls.

- **Management Account** – Handles centralized logging, billing, and monitoring.

AWS Organizations and IAM roles are utilized for cross-account access management and governance.

3. Network Design

VPC Architecture

- **AWS VPC CIDR:** 10.0.0.0/16
- **Subnets:**
  - **Public Subnets:** Two public subnets in different availability zones for ALB and NAT Gateways.
  - **Private Subnets:** Two private subnets in different availability zones for EKS worker nodes and RDS.

Security Measures

- **AWS Web Application Firewall (WAF)** is integrated with **AWS ALB** to mitigate attacks.
- **Security Groups and Network ACLs:**
  - **ALB:** Only allows HTTP/HTTPS traffic from WAF.
  - **EKS Worker Nodes:** Only accept traffic from ALB.
  - **RDS PostgreSQL:** Only accepts traffic from Flask backend on port 5432.
- **NAT Gateways** (one per public subnet) provide controlled outbound access for private resources.
- **VPC Flow Logs** monitor and analyze network traffic for security and troubleshooting.

4. Compute Platform

Amazon EKS Cluster

- EKS deployed in private subnets for security.
- Auto Scaling enabled for dynamic resource allocation.
- Node Groups:
  - **On-Demand Nodes** for critical workloads.

    - **Spot Instances** for cost optimization in non-critical workloads.

AWS ALB Ingress Controller

- Manages traffic routing within EKS.
- Routes:
    - / → Frontend (React SPA pods)
    - /api/* → Backend (Flask API pods)
- Ensures secure traffic flow from ALB to EKS backend services.

Containerization Strategy

- Dockerized services (Frontend and Backend).
- Images stored in AWS Elastic Container Registry (ECR).
- Helm used for Kubernetes deployments.
- GitHub Actions automates CI/CD deployments.

5. Database Configuration

Amazon RDS for PostgreSQL

- **Multi-AZ Deployment:** Ensures high availability with automatic failover.
- **Security:**
    - Data at rest encryption with AWS KMS.
    - Data in transit encryption using TLS.
    - Access restricted to backend (Flask API) in private subnets.
- Automated Backups & Point-in-Time Recovery (PITR) enabled.
- Read replicas for scalability.
- CloudWatch Logs for database monitoring.

6. CI/CD Pipeline

- GitHub Actions for CI/CD.
- Workflow:

    - Code push triggers image build → Image pushed to ECR → Deploys to EKS using Helm.
- **Monitoring and Logging:**
    - Amazon CloudWatch for logs.
    - Prometheus & Grafana for metrics.

7. Conclusion

This AWS architecture ensures high availability, security, and scalability while leveraging AWS-managed services. The use of EKS, RDS, and CI/CD automation allows Innovate Inc. to efficiently deploy, manage, and scale their application while keeping operational overhead minimal.