DevOps Hackathon Challenge: Containerized Microservices Deployment

Overview

Welcome to the DevOps Hackathon Challenge! In this hackathon, you will demonstrate your skills in containerization, Infrastructure as Code (IaC), CI/CD, and cloud deployment using AWS services. You will be working with a simple healthcare application consisting of two microservices.

Challenge Structure

This hackathon offers three deployment tracks. Each team should choose one track to focus on:

1. Amazon EKS Deployment Track
2. AWS Fargate Deployment Track
3. AWS Lambda Container Deployment Track

Common Requirements

Regardless of the track you choose, you will be working with the following common elements:

1. **Microservices**: You will be provided with two Node.js microservices - a Patient Service and an Appointment Service. The code for these services can be found in the Sample Microservices Code file.

2. **Containerization**: You need to containerize these microservices using Docker.

3. **Infrastructure as Code (Terraform)**:

    - Set up a Terraform project structure supporting multiple environments (dev, staging, prod).
    - Provision the following AWS resources:
        - VPC with public and private subnets across two availability zones
        - IAM roles and security groups
        - S3 bucket for Terraform state storage
        - DynamoDB table for state locking
        - (Other resources specific to your chosen track)

4. **Terraform State Management**:

    - Implement remote state storage using S3
    - Set up state locking with DynamoDB
    - Configure workspace separation for different environments

5. **GitHub Actions for IaC**:

    - Create workflows for:
        - Terraform fmt and validate on all PRs
        - Terraform plan on pull requests

    - Terraform apply on merges to main branch

6. **CI/CD**: Implement a CI/CD pipeline using GitHub Actions for your application code.
7. **Monitoring and Logging**: Set up basic monitoring and logging using AWS CloudWatch.

Time Allocation

You will have 5 hours to complete this challenge. Budget your time wisely across planning, development, deployment, and documentation.

Evaluation Criteria

While specific criteria vary by track, you will generally be evaluated on:

1. Correct implementation of the chosen deployment platform
2. Quality and security of the IaC implementation
3. Effectiveness of the CI/CD pipeline
4. Containerization best practices
5. Monitoring and logging setup
6. Documentation quality
7. Overall architecture and security considerations
8. Proper implementation of Terraform state management and collaboration features

Getting Started

1. Review the common requirements and evaluation criteria.
2. Choose your deployment track: EKS, Fargate, or Lambda Container.
3. Follow the specific instructions for your chosen track.
4. Use the provided microservices code as a starting point for your application.

Good luck, and happy coding!