**Logicworks DevOps Infrastructure**

**Requirement 1: Infrastructure as Code (IAC)**

**Implementation:**

* Infrastructure as Code implemented using with AWS provider.
* All components (VPC, ECS, ALB, CodePipeline, ECR, CodeCommit, Lambda, CloudWatch) defined in 8 reusable Terraform modules.
* Entire infrastructure deployed within few minutes.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

**Requirement 2: Multi-Region Architecture for High Availability**

**Implementation:**

* Infrastructure deployed in two AWS regions (us-east-1, us-east-2) for high availability and disaster recovery.
* Each region has isolated VPCs, public/private subnets, NAT Gateways, Internet Gateway, Application Load Balancers, ECS Fargate clusters.
* Independent CI/CD pipelines per region for fault tolerance.
* Traffic can be routed to healthy region during failure.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Requirement 3: Container Management System**

**Implementation:**

* Application containerized with Docker (multi-stage builds for optimized size/security).
* Images stored in Amazon ECR (separate repos per region).
* Orchestration via Amazon ECS Fargate (serverless compute).
* Auto-scaling based on CPU/memory (70% threshold), scaling from 2 to 10 tasks.
* Base image: public.ecr.aws/docker/library/node:18-alpine

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Requirement 4: Automated CI/CD Pipeline**

**Implementation:**

* Fully automated CI/CD pipeline using AWS CodePipeline (five stages, independent in both regions).
* Workflow: Source (CodeCommit push via EventBridge) → Build (CodeBuild, Docker image, ECR push) → Deploy Staging → Manual Approval (SNS notification) → Deploy Production.
* Pipeline completes code commit to staging in ~3-5 minutes.
* buildspec.yml defines build instructions.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

**Requirement 5: Source Code Repository Replication**

**Implementation:**

* Automated replication via AWS Lambda , triggered by EventBridge on CodeCommit push.
* Lambda retrieves changed files, commits to secondary region repository, prefixes messages with “[Replicated]”.
* Eliminates cross-region latency; pipelines pull from local repositories.
* Replication completes in ~5-8 seconds for typical commits.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Requirement 6: Continuous Monitoring with Notifications**

**Implementation:**

* Monitoring via Amazon CloudWatch (custom dashboards for both regions).
* CloudWatch Alarms for ECS CPU/memory, ALB 5xx errors, unhealthy targets, response time, application errors.
* Alarms integrated with SNS for email notifications to DevOps team.
* Log Groups capture logs from ECS, CodeBuild, Lambda, VPC Flow A screenshot of a computer

  AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Requirement 7: Approval Gate Before Production**

**Implementation:**

* Manual approval gate in CodePipeline between staging and production.
* SNS notification sent to approvers with AWS Console links for review.
* Approvers validate staging, review artifacts, approve/reject deployment.
* Audit trail maintained for all approval decisions.

**Architecture Summary**

**Multi-Region Architecture:**

* Regions: us-east-1 (primary), us-east-2 (secondary)
* Resources per region: VPC, 2 Public Subnets, 2 Private Subnets, 2 NAT Gateways, 1 Internet Gateway, 1 Application Load Balancer, 1 ECS Cluster, 1 ECR Repository, 1 CodeCommit Repository, 1 CodePipeline, 1 CodeBuild Project, CloudWatch Dashboard/Alarms
* Cross-region: Lambda Replication Function (us-east-1), EventBridge Rules, SNS Topics

**Application Flow:**

1. Developer pushes code to primary CodeCommit (us-east-1)
2. EventBridge triggers Lambda replication to secondary CodeCommit (us-east-2)
3. Both pipelines trigger via EventBridge
4. CodeBuild builds Docker images in parallel
5. Images pushed to respective ECR repositories(docker registry was giving a limit error for unauthenticated requests, could have used my docker registry credentials in secrets manager and use it authenticated request but ECR was more organic choice)
6. ECS deploys to staging in both regions
7. Manual approval via SNS notification
8. Upon approval, production deployment in both regions
9. CloudWatch monitors all components