Runbook: CI/CD Operations - CVDex

Finanzas SD – Architecture, Flows & SOPs

Arquitectura, Flujos y Procedimientos

November 11, 2025

1 Runbook: CI/CD Operations - CVDex

CI/CD Operations Runbook
Runbook de Operaciones CI/CD

1.1 EN: Overview

This runbook provides operational procedures for managing the CI/CD pipeline for Finanzas SD using GitHub Actions with OIDC authentication and multi-stage deployments.

1.2 ES: Descripción General

Este runbook proporciona procedimientos operacionales para gestionar el pipeline CI/CD para Finanzas SD usando GitHub Actions con autenticación OIDC y despliegues multi-etapa.

1.3 EN: Pipeline Architecture

1.3.1 Stages

1. **Development (dev)**: Automatic deployment on merge to main
2. **Staging (stg)**: Manual promotion from dev
3. **Production (prod)**: Manual promotion from stg with evidence pack

1.3.2 Authentication

- **OIDC Only**: No static AWS credentials
- **GitHub OIDC Provider**: Configured in AWS IAM
- **Short-lived Tokens**: Automatically expired after workflow

1.3.3 Deployment Components

- **API** (SAM): Lambda functions, API Gateway, DynamoDB
- **UI** (Vite): React SPA built and deployed to S3/CloudFront
- **Observability**: CloudWatch dashboards, alarms, canaries

1.4 ES: Arquitectura del Pipeline

1.4.1 Etapas

1. **Desarrollo (dev)**: Despliegue automático al fusionar a main
2. **Preparación (stg)**: Promoción manual desde dev
3. **Producción (prod)**: Promoción manual desde stg con paquete de evidencia

1.4.2 Autenticación

- **Solo OIDC**: Sin credenciales estáticas de AWS
- **Proveedor OIDC de GitHub**: Configurado en AWS IAM
- **Tokens de Corta Duración**: Expirados automáticamente después del flujo de trabajo

1.4.3 Componentes de Despliegue

- **API** (SAM): Funciones Lambda, API Gateway, DynamoDB
- **UI** (Vite): SPA React construido y desplegado en S3/CloudFront
- **Observabilidad**: Paneles CloudWatch, alarmas, canaries

1.5 EN: Running a Deployment

1.5.1 Prerequisites Check

Before deployment, verify: - [ ] All unit tests passing - [ ] API contract tests passing (Newman) - [ ] Linting clean - [ ] No security vulnerabilities - [ ] Feature flags configured - [ ] Environment variables set

Command:

[] *# Run all checks* npm run lint npm test npm run api:test

1.5.2 Dev Deployment (Automatic)

Trigger: Merge to `main` branch

Workflow: `.github/workflows/r1-dev-pipeline.yml`

Steps: 1. Checkout code 2. Configure AWS credentials via OIDC 3. Run unit tests 4. Run API contract tests 5. SAM build 6. SAM deploy to dev environment 7. Vite build UI 8. Deploy UI to S3 (dev bucket) 9. Invalidate CloudFront (dev distribution) 10. Deploy observability stack 11. Run smoke tests 12. Generate step summary

Monitoring: - Watch GitHub Actions UI for progress - Check CloudWatch logs for Lambda deployment - Verify API health endpoint: `/health` - Test UI at dev CloudFront URL

Rollback (if needed):

```
[] # Revert to previous commit git revert HEAD git push origin main
# Or deploy specific version sam deploy --stack-name finanzas-sd-api-dev \ --parameter-overrides Version=previous-version
```

1.5.3 Staging Promotion

Trigger: Manual via GitHub Actions UI

Workflow: `.github/workflows/promote-to-staging.yml`

Prerequisites: - Dev deployment successful - Dev smoke tests passing - Dev environment stable for 24+ hours - No critical bugs reported

Steps: 1. Navigate to Actions → "Promote to Staging" 2. Click "Run workflow" 3. Select source commit SHA (from dev) 4. Add deployment notes 5. Click "Run workflow"

Workflow Actions: 1. Checkout specified commit 2. Run full test suite 3. SAM deploy to stg environment 4. Deploy UI to stg 5. Run extended smoke tests 6. Generate evidence pack 7. Post summary to step output

Post-Deployment: - Notify QA team for testing - Monitor staging for 48 hours - Document any issues found - Update evidence pack with results

1.5.4 Production Deployment

Trigger: Manual via GitHub Actions UI

Workflow: `.github/workflows/promote-to-prod.yml`

Prerequisites: - Staging stable for 48+ hours - QA sign-off received - Evidence pack complete and approved - Change management ticket approved - Maintenance window scheduled

Evidence Pack Required: - Test results (unit, integration, E2E) - API contract validation report - Security scan results (CodeQL, dependency check) - Performance test results - Smoke test results from staging - Rollback plan documented

Steps: 1. Verify maintenance window is active 2. Notify stakeholders of deployment start 3. Navigate to Actions → "Deploy to Production" 4. Upload evidence pack 5. Select staging commit SHA 6. Add deployment notes 7. Click "Run workflow"

Workflow Actions: 1. Validate evidence pack 2. Checkout specified commit 3. SAM deploy to prod environment 4. Deploy UI to prod 5. Run production smoke tests 6. Update CloudWatch dashboards 7. Verify canaries passing 8. Generate deployment report

Post-Deployment Monitoring (first 2 hours): - Watch error rates in CloudWatch - Monitor canary success rate - Check API latency metrics - Verify no increase in 4xx/5xx errors - Monitor database performance

Communication: - Post deployment announcement in Slack - Update status page - Send email to stakeholders - Update change management ticket

1.6 ES: Ejecutar un Despliegue

1.6.1 Verificación de Prerrequisitos

Antes del despliegue, verificar: - [ ] Todas las pruebas unitarias pasando - [ ] Pruebas de contrato API pasando (Newman) - [ ] Linting limpio - [ ] Sin vulnerabilidades de seguridad - [ ] Feature flags configurados - [ ] Variables de entorno establecidas

Comando:

```
[] # Ejecutar todas las verificaciones npm run lint npm test npm run api:test
```

1.6.2 Despliegue Dev (Automático)

Disparador: Fusionar a rama `main`

Flujo de Trabajo: `.github/workflows/r1-dev-pipeline.yml`

Pasos: 1. Checkout de código 2. Configurar credenciales AWS vía OIDC 3. Ejecutar pruebas unitarias 4. Ejecutar pruebas de contrato API 5. Construcción SAM 6. Despliegue SAM a ambiente dev 7. Construcción Vite UI 8. Desplegar UI a S3 (bucket dev) 9. Invalidar CloudFront (distribución dev) 10. Desplegar stack de observabilidad 11. Ejecutar smoke tests 12. Generar resumen de pasos

Monitoreo: - Ver UI de GitHub Actions para progreso - Verificar logs CloudWatch para despliegue Lambda - Verificar endpoint de salud API: `/health` - Probar UI en URL CloudFront dev

Rollback (si es necesario):

```
[] # Revertir a commit anterior git revert HEAD git push origin main
# O desplegar versión específica sam deploy --stack-name finanzas-sd-api-dev \
--parameter-overrides Version=versión-anterior
```

1.6.3 Promoción a Staging

Disparador: Manual vía UI GitHub Actions

Flujo de Trabajo: `.github/workflows/promote-to-staging.yml`

Prerrequisitos: - Despliegue dev exitoso - Smoke tests dev pasando - Ambiente dev estable por 24+ horas - Sin bugs críticos reportados

Pasos: 1. Navegar a Actions → “Promote to Staging” 2. Hacer clic en “Run workflow” 3. Seleccionar SHA de commit origen (desde dev) 4. Agregar notas de despliegue 5. Hacer clic en “Run workflow”

Acciones del Flujo de Trabajo: 1. Checkout de commit especificado 2. Ejecutar suite completa de pruebas 3. Despliegue SAM a ambiente stg 4. Desplegar UI a stg 5. Ejecutar smoke tests extendidos 6. Generar paquete de evidencia 7. Publicar resumen en salida de paso

Post-Despliegue: - Notificar al equipo QA para pruebas - Monitorear staging por 48 horas - Documentar problemas encontrados - Actualizar paquete de evidencia con resultados

1.6.4 Despliegue a Producción

Disparador: Manual vía UI GitHub Actions

Flujo de Trabajo: `.github/workflows/promote-to-prod.yml`

Prerrequisitos: - Staging estable por 48+ horas - Aprobación QA recibida - Paquete de evidencia completo y aprobado - Ticket de gestión de cambios aprobado - Ventana de mantenimiento programada

Paquete de Evidencia Requerido: - Resultados de pruebas (unitarias, integración, E2E) - Reporte de validación de contrato API - Resultados de escaneo de seguridad (CodeQL, verificación de dependencias) - Resultados de pruebas de rendimiento - Resultados de smoke tests de staging - Plan de rollback documentado

Pasos: 1. Verificar ventana de mantenimiento activa 2. Notificar a partes interesadas del inicio de despliegue 3. Navegar a Actions → “Deploy to Production” 4. Cargar paquete de evidencia 5. Seleccionar SHA de commit staging 6. Agregar notas de despliegue 7. Hacer clic en “Run workflow”

Acciones del Flujo de Trabajo: 1. Validar paquete de evidencia 2. Checkout de commit especificado 3. Despliegue SAM a ambiente prod 4. Desplegar UI a prod 5. Ejecutar smoke tests de producción 6. Actualizar paneles CloudWatch 7. Verificar canaries pasando 8. Generar reporte de despliegue

Monitoreo Post-Despliegue (primeras 2 horas): - Ver tasas de error en CloudWatch - Monitorear tasa de éxito de canary - Verificar métricas de latencia API - Verificar sin aumento en errores 4xx/5xx - Monitorear rendimiento de base de datos

Comunicación: - Publicar anuncio de despliegue en Slack - Actualizar página de estado - Enviar correo a partes interesadas - Actualizar ticket de gestión de cambios

---

1.7 EN: Troubleshooting

1.7.1 Deployment Fails at SAM Build

Symptoms: Build error in GitHub Actions

Common Causes: - TypeScript compilation errors - Missing dependencies - Syntax errors in template.yaml

Resolution:

```
[] # Test locally cd services/finanzas-api npm ci sam build sam local start-api
```

1.7.2 OIDC Authentication Fails

Symptoms: “Unable to assume role” error

Common Causes: - OIDC provider not configured in AWS - Trust policy incorrect - GitHub repository settings wrong

Resolution: 1. Verify OIDC provider exists in AWS IAM 2. Check trust policy includes correct GitHub repo 3. Verify workflow has correct role ARN 4. Check IAM role permissions

1.7.3 UI Deploy Fails

Symptoms: S3 sync or CloudFront invalidation errors

Common Causes: - Incorrect S3 bucket name - CloudFront distribution not found - Permissions issue

Resolution:

```
[] # Verify bucket exists aws s3 ls s3://your-bucket-name
# Check CloudFront distribution aws cloudfront get-distribution --id DISTRIBUTION_ID
# Test manual upload npm run build aws s3 sync dist/ s3://your-bucket/finanzas/
```

1.7.4 Smoke Tests Fail

Symptoms: Health endpoint returns error

Common Causes: - Lambda not warmed up (cold start) - DynamoDB table not accessible - Environment variables missing

Resolution: 1. Check Lambda logs in CloudWatch 2. Verify DynamoDB table exists 3. Check Lambda environment variables 4. Test endpoint manually with curl 5. Wait 2 minutes and retry

1.8 ES: Solución de Problemas

1.8.1 Falla en Construcción SAM

Síntomas: Error de construcción en GitHub Actions

Causas Comunes: - Errores de compilación TypeScript - Dependencias faltantes - Errores de sintaxis en template.yaml

Resolución:

```
[] # Probar localmente cd services/finanzas-api npm ci sam build sam local start-api
```

1.8.2 Falla Autenticación OIDC

Síntomas: Error “Unable to assume role”

Causas Comunes: - Proveedor OIDC no configurado en AWS - Política de confianza incorrecta - Configuración de repositorio GitHub incorrecta

Resolución: 1. Verificar proveedor OIDC existe en AWS IAM 2. Verificar política de confianza incluye repo GitHub correcto 3. Verificar flujo de trabajo tiene ARN de rol correcto 4. Verificar permisos de rol IAM

1.8.3 Falla Despliegue UI

Síntomas: Errores de sincronización S3 o invalidación CloudFront

Causas Comunes: - Nombre de bucket S3 incorrecto - Distribución CloudFront no encontrada - Problema de permisos

Resolución:

```
[] # Verificar bucket existe aws s3 ls s3://nombre-su-bucket
# Verificar distribución CloudFront aws cloudfront get-distribution --id DISTRIBU-
TION_ID
# Probar carga manual npm run build aws s3 sync dist/ s3://su-bucket/finanzas/
```

1.8.4 Fallan Smoke Tests

Síntomas: Endpoint de salud devuelve error

Causas Comunes: - Lambda no calentado (arranque en frío) - Tabla DynamoDB no accesible - Variables de entorno faltantes

Resolución: 1. Verificar logs Lambda en CloudWatch 2. Verificar tabla DynamoDB existe 3. Verificar variables de entorno Lambda 4. Probar endpoint manualmente con curl 5. Esperar 2 minutos y reintentar

---

Document Version: 1.0
Effective Date: November 2024
Review Date: May 2025
Owner: DevOps Team / Equipo DevOps
Status: Active / Activo