Finanzas Verification Scripts – Quick Reference

Finanzas SD – Architecture, Flows & SOPs

Arquitectura, Flujos y Procedimientos

November 11, 2025

1 Finanzas Verification Scripts - Quick Reference

1.1 Overview

This guide provides quick instructions for running the Finanzas deployment verification scripts and interpreting their results.

1.2 Prerequisites

- AWS CLI configured with appropriate credentials
- Access to the repository scripts directory
- For E2E tests: Valid Cognito user credentials

1.3 Available Scripts

1.3.1 1. Basic Deployment Verification

Script: `verify-deployment.sh`
Location: `/scripts/verify-deployment.sh`
Purpose: Quick infrastructure and accessibility check

```
[] cd /path/to/repository ./scripts/verify-deployment.sh
```

Usage

What It Checks □ CloudFront Distribution - Distribution is accessible - `/finanzas/*` behavior exists - Distribution configuration is correct

□ S3 Bucket Content - Bucket is accessible - PMO files exist at root (index.html) - Finanzas files exist at /finanzas/ (finanzas/index.html) - File timestamps are recent

□ Web Accessibility - PMO Portal returns HTTP 200 - Finanzas Portal returns HTTP 200 - Finanzas Catalog page returns HTTP 200

Expected Output

```
==========================================
□ Deployment Verification - PMO + Finanzas
==========================================

Configuration:
  CloudFront Distribution: EPQU7PVDLQXUA
  CloudFront Domain: d7t9x3j66yd8k.cloudfront.net
  S3 Bucket: ukusi-ui-finanzas-prod
```

```
1  Checking CloudFront Behaviors...

 CloudFront distribution accessible
 /finanzas/* behavior found

2  Checking S3 Bucket Content...

 S3 bucket accessible: s3://ukusi-ui-finanzas-prod/
 PMO Portal: index.html found at root
 Finanzas Portal: index.html found
   Last modified: 2025-11-10T12:34:56.000Z

3  Testing Web Accessibility...

Testing PMO Portal: https://d7t9x3j66yd8k.cloudfront.net/
 PMO Portal: HTTP 200
Testing Finanzas Portal: https://d7t9x3j66yd8k.cloudfront.net/finanzas/
 Finanzas Portal: HTTP 200
Testing Finanzas Catalog: https://d7t9x3j66yd8k.cloudfront.net/finanzas/catalog/ru
 Finanzas Catalog: HTTP 200

================================================
Summary
================================================

 Deployment appears correct!
```

Troubleshooting **CloudFront distribution not accessible** - Check AWS credentials - Verify you have permissions to read CloudFront configurations - Confirm distribution ID is correct

⚠ **/finanzas/* behavior NOT found** - Log into AWS Console → CloudFront → Distributions → EPQU7PVDLQXUA - Check Behaviors tab for `/finanzas/*` entry - If missing, add the behavior (see FINANZAS_NEXT_STEPS.md)

 Finanzas Portal: index.html NOT found - Check S3 deployment: `aws s3 ls s3://ukusi-ui-finanzas-prod/finanzas/` - Re-run deployment: `npm run build:finanzas` and upload to S3 - Verify build output directory: `dist-finanzas/`

 Finanzas Portal: HTTP 403 or 404 - Verify CloudFront behavior for `/finanzas/*` exists - Check custom error responses (403/404 → /finanzas/index.html) - Create Cloud-

Front invalidation: `aws cloudfront create-invalidation --distribution-id EPQU7PVDLQXUA --paths '/finanzas/*'`

1.3.2 2. End-to-End Smoke Test

Script: `finanzas-e2e-smoke.sh`
Location: `/scripts/finanzas-e2e-smoke.sh`
Purpose: Complete API → Lambda → DynamoDB verification

```
[] cd /path/to/repository
# Set credentials as environment variables export USERNAME="your-test-user@example.com"
export PASSWORD="your-test-password"
# Run the script ./scripts/finanzas-e2e-smoke.sh
```

Usage

What It Tests **Section 1: Cognito Authentication** - Obtains ID token from Cognito - Validates token claims (aud, iss)

Section 2: API Health & Public Endpoints - Tests `/health` endpoint (public, no auth) - Tests `/catalog/rubros` endpoint (public) - Tests `/allocation-rules` endpoint (protected)

Section 3: Lambda → DynamoDB Write Test - Creates test adjustment via POST `/adjustments` - Verifies Lambda execution - Confirms API returns 201 Created

Section 4: DynamoDB Verification - Queries DynamoDB directly for created record - Confirms data persistence - Validates table structure

Section 5: Audit Log Scan - Scans `finz_audit_log` for recent entries - Verifies audit trail is working

Expected Output

```
╔═══════════════════════════════════════════════════════════╗
║   Finanzas SDT End-to-End Smoke Test                      ║
║   API → Lambda → DynamoDB Verification                    ║
╚═══════════════════════════════════════════════════════════╝

 Section 1: Cognito Authentication
──────────────────────────────────────

  Getting ID token for: christian.valencia@ikusi.com
```

```
 IdToken obtained
 Token aud matches AppClientId: dshos5iou44tuach7ta3ici5m

 Section 2: API Health & Public Endpoints
──────────────────────────────────────────────
Testing: GET /health (public, no auth required)
 GET /health → 200
   Response: {"status":"ok"}

Testing: GET /catalog/rubros (public, no auth required)
 GET /catalog/rubros → 200
   Rubros count: 71

Testing: GET /allocation-rules (protected, requires Bearer token)
 GET /allocation-rules → 200
   Rules count: 2

 Section 3: Lambda → DynamoDB Write Test
──────────────────────────────────────────────
Testing: POST /adjustments (protected, writes to DynamoDB)
   Creating test adjustment: ADJ-E2E-1699632000
 POST /adjustments → 201 (created)
   Response: {"adjustment_id":"ADJ-E2E-1699632000","status":"created"}

 Section 4: DynamoDB Verification
──────────────────────────────────────
   Checking DynamoDB table: finz_adjustments
    Looking for record: adjustment_id = ADJ-E2E-1699632000
 Record found in finz_adjustments
   Item data:
     {
       "adjustment_id": {"S": "ADJ-E2E-1699632000"},
       "project_id": {"S": "PRJ-E2E-TEST"},
       "amount": {"N": "12345"},
       "currency": {"S": "COP"}
     }

 Section 5: Audit Log Scan (Optional)
──────────────────────────────────────────
```

```
🔍 Scanning finz_audit_log for recent entries...
📊 Found 5 recent audit entries

╔═══════════════════════════════════════════════════════════════╗
║                    ✅ SMOKE TESTS COMPLETE                    ║
╚═══════════════════════════════════════════════════════════════╝

📋 EVIDENCE SUMMARY:
  ✅ Auth: IdToken obtained from Cognito
  ✅ API Health: Responding
  ✅ Catalog: GET /catalog/rubros → 200 (count: 71)
  ✅ Rules: GET /allocation-rules → 200 (count: 2)
  ✅ Lambda: POST /adjustments → 201
  ✅ DynamoDB: Record persisted in finz_adjustments

🎯 WHAT THIS PROVES:
  1. Cognito authentication working (correct IdToken)
  2. API Gateway routing requests to Lambda
  3. Lambda handler executing successfully
  4. DynamoDB write operations persisting data
  5. End-to-end wiring: UI → API → Lambda → DynamoDB ✅
```

Troubleshooting ❌ **FAILED: Could not obtain IdToken** - Verify USERNAME and PASSWORD environment variables are set correctly - Check user exists in Cognito User Pool - Verify user is not locked or requires password reset - Confirm App Client ID is correct

❌ **GET /catalog/rubros → 401** - This endpoint should be public - check Lambda authorizer configuration - Verify API Gateway resource configuration - Check if endpoint requires authentication (it shouldn't)

❌ **GET /allocation-rules → 401** - Verify Bearer token is being sent correctly - Check token is not expired - Verify JWT authorizer configuration in API Gateway - Confirm token aud/iss claims match expected values

❌ **POST /adjustments → 500** - Check Lambda function logs: `aws logs tail /aws/lambda/fina --follow` - Verify DynamoDB table exists and Lambda has write permissions - Check IAM role attached to Lambda function - Review Lambda environment variables

❌ **Record NOT found in finz_adjustments** - Verify table name is correct - Check AWS region: us-east-2 - Confirm Lambda successfully wrote to DynamoDB - Query table directly: `aws dynamodb scan --table-name finz_adjustments --limit 10`

⚠ **Rubros count: ?** - Indicates DynamoDB scan failed or returned invalid data -

Seed the rubros table: Run seed scripts in `/scripts/ts-seeds/` - Verify table structure matches expected schema

---

1.4 Quick Verification Workflow

1.4.1 For Daily Checks

[] *# 1. Quick health check* ./scripts/verify-deployment.sh
 # 2. If all □, deployment is healthy # 3. If any □, investigate and fix

1.4.2 For Full Verification (After Deployment)

[] *# 1. Basic infrastructure check* ./scripts/verify-deployment.sh
 # 2. Full E2E test export USERNAME="your-test-user@example.com" export PASSWORD="your-test-password" ./scripts/finanzas-e2e-smoke.sh
 # 3. Review outputs - ALL should be □ # 4. Collect evidence (save terminal output)

1.4.3 Manual UI Testing

After scripts pass, perform manual UI tests:

1. **Navigate to Finanzas:**
   - Open: https://d7t9x3j66yd8k.cloudfront.net/finanzas/
   - Verify page loads without errors
2. **Test Login:**
   - Click "Sign In"
   - Enter credentials
   - Verify redirect to /finanzas/ after login
3. **Test Navigation:**
   - Click "Rubros" → verify data loads (71 entries)
   - Click "Rules" → verify data loads (2+ entries)
   - Check browser console for errors
4. **Verify Assets:**
   - Open DevTools → Network tab
   - Refresh page

- Confirm no 404 errors on JS/CSS files
- Confirm API calls return 200

1.5 Interpreting Results

1.5.1 All Green (□)

Status: □ System is healthy
Action: No action needed

1.5.2 Some Warnings (⚠)

Status: □ Minor issues detected
Action: Review warnings, may not be critical

1.5.3 Any Failures (□)

Status: □ Critical issues detected
Action: 1. Note which checks failed 2. Review troubleshooting section 3. Fix issues 4. Re-run scripts 5. Do not proceed to production until all □

1.6 Evidence Collection

1.6.1 For Compliance/Audit

After running scripts, collect:

1. **Terminal Output:**
   - Save complete output of both scripts
   - Timestamp the verification
2. **Screenshots:**
   - CloudFront behaviors configuration
   - S3 bucket structure
   - Finanzas UI pages (home, rubros, rules)
   - Browser DevTools showing API calls
3. **API Responses:**
   - Sample response from /catalog/rubros
   - Sample response from /allocation-rules

- API health check response

4. **Documentation:**
   - Update DEPLOYMENT_VERIFICATION_CHECKLIST.md
   - Note any issues and resolutions
   - Document deployment date/time

1.7 Support

1.7.1 Need Help?

1. **Review Documentation:**
   - FINANZAS-DEPLOYMENT-COMPLETE.md
   - FINANZAS_NEXT_STEPS.md
   - FINANZAS_DEPLOYMENT_VERIFICATION.md
2. **Check CloudFront & S3:**
   - Verify configurations in AWS Console
   - Create cache invalidations if needed
3. **Review API Logs:**
   - Check Lambda CloudWatch logs
   - Review API Gateway access logs
   - Check DynamoDB metrics
4. **Contact DevOps:**
   - Provide script outputs
   - Share error messages
   - Include timestamps

Last Updated: 2025-11-10
Version: 1.0
Maintained by: DevOps Team