

Test Scenario: Ungrouped Cognito Users

Finanzas SD – Architecture, Flows & SOPs

Arquitectura, Flujos y Procedimientos

January 8, 2026

1 Test Scenario: Ungrouped Cognito Users

1.1 Issue Description

After PR #577 (tighten RBAC), ungrouped Cognito users were unable to access the UI and received the error:

Application Error
DEFAULT_ROLE is not defined

1.2 Root Cause

The AuthProvider referenced an undefined DEFAULT_ROLE constant in 9 places. When PR #577 removed the default EXEC_RO assignment for ungrouped users, these references caused the application to crash.

1.3 Fix Applied

All DEFAULT_ROLE references have been removed and replaced with proper null handling.

1.4 Test Scenarios

1.4.1 Scenario 1: User with No Cognito Groups

Setup: - User authenticates successfully with Cognito - User has NO groups assigned (e.g., cognito:groups claim is empty array)

Expected Behavior: 1. ☐ User authenticates successfully (isAuthenticated = true)
2. ☐ availableRoles array is empty 3. ☐ currentRole is set to null 4. ☐ App.tsx checks availableRoles.length === 0 5. ☐ NoAccess component is rendered with message: - “Sin permisos asignados” (No permissions assigned) - Instructions to contact system administrator - Sign out button

Previous Behavior: - ☐ Application crashed with “DEFAULT_ROLE is not defined” error - ☐ User saw error page instead of helpful message

1.4.2 Scenario 2: User with Only Unrecognized Groups

Setup: - User has groups like ["ikusi-acta-ui"] or ["UNKNOWN_GROUP"] - These groups don't map to any application role

Expected Behavior: 1. ☐ mapGroupsToRoles returns empty array 2. ☐ Same flow as Scenario 1 - NoAccess component shown

1.4.3 Scenario 3: User with Valid Groups (Regression Test)

Setup: - User has groups like ["PM0"], ["SDMT"], ["EXEC_R0"]

Expected Behavior: 1. mapGroupsToRoles returns corresponding roles 2. availableRoles contains the roles 3. currentRole is set to first available role 4. User can access the application normally 5. Navigation and role switching work correctly

1.4.4 Scenario 4: User Logout (Regression Test)

Setup: - Authenticated user clicks logout

Expected Behavior: 1. currentRole is set to null (not DEFAULT_ROLE) 2. availableRoles is cleared 3. No errors during logout process 4. User redirected to login page

1.5 Code Changes Summary

1.5.1 AuthProvider.tsx Changes:

1. **Line 210-219:** Added null check before calling getRoutesForRole

```
[] if (!currentRole) { setRouteConfigMissing(true); console.warn("[Router] No role assigned - user has no access", {...}); return; }
```

2. **Line 244-252:** Added check to ensure currentRole stays null for users without roles

```
[] if (userRoles.length === 0) { if (currentRole !== null) { setCurrentRole(null); } return; }
```

3. **Line 328, 338, 467:** Changed setCurrentRole(DEFAULT_ROLE) to setCurrentRole(null)

4. **Line 516-522:** Added null check in checkRouteAccess

```
[] if (!currentRole) { return false; }
```

5. **Line 524-530:** Added null check in checkActionPermission

```
[] if (!currentRole) { return false; }
```

6. **Line 548, 572:** Removed || DEFAULT_ROLE fallback in session and context value

1.6 Verification Steps

1.6.1 Manual Testing

1. Create a Cognito user with no groups
2. Authenticate with that user
3. Verify NoAccess screen is shown (not error page)
4. Verify sign out button works

1.6.2 Automated Testing

- JWT tests pass (no default EXEC_RO for ungrouped)
- Auth routes tests pass

- Code review completed with no issues
- CodeQL security scan passed with 0 alerts

1.7 Security Impact

This fix maintains the security tightening from PR #577: - Users without recognized groups have NO access (not even read-only) - No implicit EXEC_RO role assignment - Clear messaging about lack of permissions - Proper audit trail (console warnings logged)