

FASE II — AUTH CON RBAC

ISS-12: Features RoleUsers y ResourceRoles

La Matriz RBAC: Asignaciones de Roles y Concesiones de Permisos Deterministas

Propósito del ISS-12: Materialización de la Matriz RBAC

De Catálogos Inertes a Autorización Real

Catálogos Previos (ISS-11):

Role y Resource nacieron como catálogos aislados. Crear un rol como ADMIN o SELLER no concede ningún permiso por sí solo.

El Eslabón Perdido (ISS-12):

Construye las dos tablas pivote (role_users y resource_roles) que relacionan usuarios con roles y roles con recursos.

Principio 'No Entity Permission':

No existe una tabla de permisos. El permiso es la fila activa en resource_roles que une un rol a un recurso.

Lógica de Pivotes N:M y Borrado Lógico

Tablas Pivote N:M:

role_users mapea User ↔ Role. resource_roles mapea Role ↔ Resource.

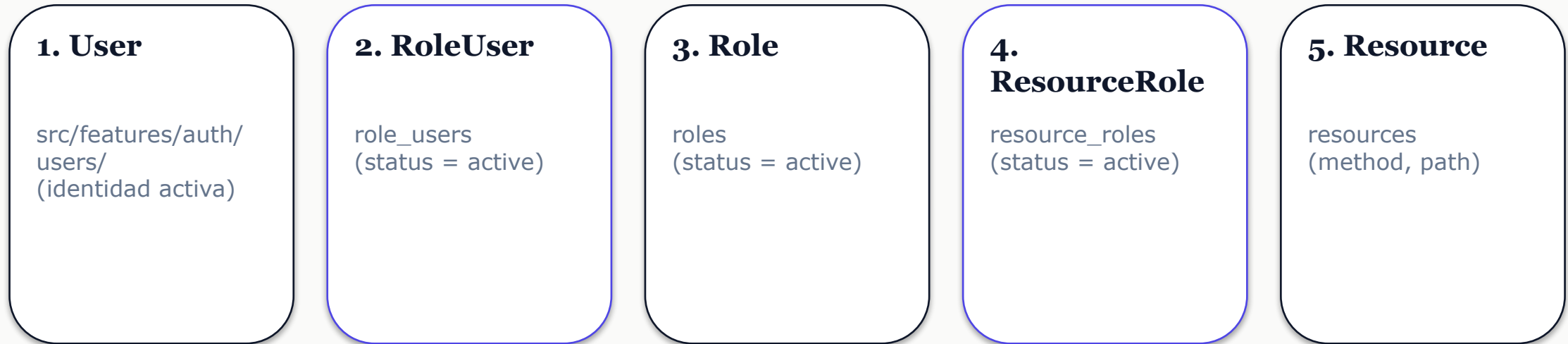
Estado 'active / inactive':

Todas las filas de pivote incluyen columna status. El borrado es siempre LÓGICO (/deactivate), preservando auditoría.

Comportamiento Upsert Lógico:

Asignar o conceder una relación previa inactiva la REACTIVA automáticamente sin crear duplicados (evita error 500).

La Cadena de Autorización en 4 Eslabones Activos



Regla Estricta del Círculo de Seguridad

- **findEffectiveForUser(userId) exige que los CUATRO eslabones intermedia estén en estado 'active'.**
- Si un usuario se desactiva, o se le retira el rol, o el rol se inhabilita, o se desactiva la concesión en resource_roles, la cadena se rompe de inmediato y el middleware devolverá 403 Forbidden.

Feature RoleUsers: Asignaciones Usuario ↔ Rol

Capa de Servicio e Idempotencia

POST /api/asignaciones-rol:

Recibe { user_id, role_id }. Verifica que ambos existan y estén activos.

Gestión de Duplicados:

Si la relación ya está activa, lanza `AppError(409, 'Role already assigned')`.

Reactivación Transparente:

Si la relación existía con `status='inactive'`, se actualiza a `'active'` (HTTP 201) conservando el ID original.

Borrado Lógico (/deactivate):

`PATCH /api/asignaciones-rol/:id/deactivate` rompe el eslabón de asignación de inmediato.

Endpoints OpenAPI (/api/asignaciones-rol)

GET /api/asignaciones-rol

Lista asignaciones activas con include de user (sin password) y role.

GET /api/asignaciones-rol/:id

Obtiene el detalle de una asignación específica por su PK.

PATCH /api/asignaciones-rol/:id/deactivate

Desactiva la asignación. El usuario pierde todos los permisos de ese rol.

PATCH /api/asignaciones-rol/:id/reactivate

Restaura la asignación desactivada devolviéndole su rol.

Feature ResourceRoles: Concesión y Gestión de Permisos

Gestión de Concesiones (/api/concesiones-rol)

POST /api/concesiones-rol:

Concede un recurso (method, path) a un rol mediante { role_id, resource_id }.

Filtros Auditables:

GET /api/concesiones-rol admite ?role_id=N ('¿qué permisos tiene este rol?') y ?resource_id=M ('¿qué roles tienen este recurso?').

Efecto Inmediato sin Reinicio:

La concesión modifica directamente la BD. La siguiente petición HTTP evalúa el permiso actualizado en tiempo real.

La Consulta de Permisos Efectivos

findEffectiveForUser(userId):

Método estelar en ResourceRolesRepository que ejecuta un JOIN secuencial sobre las 4 tablas activas.

Retorno DTO (EffectivePermissionDto):

Devuelve una lista de objetos { method, path, description, role_id, role_name }.

Consumo en Middleware (ISS-13):

Esta consulta es la fuente exacta utilizada por el middleware authorize para permitir (200) o denegar (403).

El Reconciliador Determinista: reconcileRole()

ResourceRolesService.reconcileRole(roleId, targetResourceIds)

1. Transacción ACID Única (withTransaction):

Ejecuta todo el proceso dentro de una transacción para evitar inconsistencias mid-execution.

2. Lectura del Estado Actual del Rol:

Obtiene todas las filas existentes en resource_rols para el roleId especificado.

3. Matriz de Decisiones Automáticas:

- Recurso en target + no existía en BD —> Inserción con status='active'.
- Recurso en target + estaba 'inactive' —> Reactivación con status='active'.
- Recurso NO en target + estaba 'active' —> Borrado lógico con status='inactive'.

4. Resultado de la Reconciliación (ReconcileResult):

Retorna { total_active, activated, deactivated }, garantizando que el rol tenga EXACTAMENTE los permisos indicados.

Seeders Deterministas y Matriz de Permisos Semilla

Seeder de Asignaciones (role-users.seeder.ts)

SEED_ROLE_USERS:

Mapea los usuarios canónicos a sus roles de referencia:

- admin —> ADMIN
- seller —> SELLER

Idempotencia Garantizada:

Utiliza findOrCreate por (user_id, role_id) y reactiva si estuviera inactivo.

Conteo en Base de Datos:

SELECT COUNT(*) FROM role_users devuelve exactamente 2 filas activas.

Seeder de Concesiones (resource-roles.seeder.ts)

Reparto de Concesiones:

Delegación directa en reconcileRole:

- ADMIN —> Recibe los 58 recursos del catálogo.
- SELLER —> Recibe los 7 recursos de operación marcados con seller: true.

Total de la Matriz RBAC:

58 (ADMIN) + 7 (SELLER) = EXACTAMENTE 65 concesiones activas en resource_roles.

Cero Acumulación Basura:

Reejecutar npm run db:seed reconcilia el estado exacto sin duplicar filas.

Criterios de Cierre GATE (DoD del ISS-12)

[x] Verificación de Tipos: `npx tsc --noEmit` finaliza con código 0 y sin errores de compilación.

[x] Conteos Deterministas de la BD: `SELECT COUNT(*) FROM role_users = 2` y `SELECT COUNT(*) FROM resource_roles = 65`.

[x] Idempotencia Probada: Reejecutar `npm run db:seed` mantiene exactamente 2 asignaciones y 65 concesiones activas.

[x] Comportamiento Upsert Lógico: `POST /api/asignaciones-rol` sobre pareja activa responde 409 Conflict; sobre inactiva la reactiva (201).

[x] Pruebas HTTP e Integración: Archivos `.http` verifican asignaciones, concesiones, filtro por rol (`?role_id=`) y efecto en tiempo real.

[x] Habilitación del ISS-13: La matriz RBAC queda completamente construida y lista para ser consumida por los middlewares `authenticate` y `authorize`.