MÓDULO DE SEGURIDAD – ASSIP‑ERP (Versión 1.1)

Empresa: ERP ASSIP SOLIDARIA Y FINANCIERA

Autor: Carlos González Pérez — Dirección de Tecnología

Fecha: 23/10/2025

Versión: 1.1

# 1. Introducción

El módulo de Seguridad de ASSIP‑ERP garantiza la autenticación, autorización y auditoría centralizada en todo el ecosistema del ERP. Está diseñado para proteger los datos y operaciones bajo un esquema robusto basado en JWT (JSON Web Token), roles, permisos y registros de auditoría.

# 2. Arquitectura General

El sistema está desarrollado en Spring Boot con una arquitectura modular en capas: domain, repository, service, web y config. El esquema de seguridad se apoya en JWT y en la base de datos PostgreSQL bajo el esquema 'seguridad'.  
  
Estructura de paquetes:

* co.assip.erp.seguridad  
   ├── config  
   ├── domain  
   ├── repository  
   ├── service  
   └── web

Componentes principales: SecurityConfig, JwtAuthenticationFilter, JwtService, AccessValidator, AuthService, LogEventoService y los controladores REST.

# 3. Autenticación con JWT

El proceso de autenticación se realiza a través del endpoint /api/v1/auth/login. El usuario envía sus credenciales y, si son válidas, el servidor genera un token JWT firmado con una clave secreta. Cada solicitud posterior a los endpoints protegidos debe incluir el token en el encabezado 'Authorization: Bearer <token>'. El token es verificado por JwtAuthenticationFilter.

# 4. Auditoría y Log de Eventos

El módulo registra automáticamente los eventos críticos (inicios de sesión, creación, actualización o eliminación de usuarios) en la tabla seguridad.log\_evento.  
Campos principales:  
• id\_usuario  
• módulo  
• acción  
• descripción  
• ip\_origen  
• user\_agent  
• fecha\_evento

Ejemplo de registro:  
1 | LOGIN | Inicio de sesión exitoso | 2025‑10‑22 08:05:39  
4 | CREAR\_USUARIO | Se creó el usuario prueba1 | 2025‑10‑22 08:15:35

# 5. Control de Acceso Granular (AccessValidator)

El componente AccessValidator valida en tiempo de ejecución si el rol de un usuario tiene permisos para ejecutar una acción. Internamente consulta la tabla seguridad.rol\_permisos. Si el permiso no existe, lanza una excepción controlada.  
  
Ejemplo de uso:

* accessValidator.validarAcceso(usuarioActual, "USUARIOS\_EDIT");

# 6. Estructura de Base de Datos (Esquema 'seguridad')

Tablas principales:  
• usuarios  
• roles  
• permisos  
• rol\_permisos  
• log\_evento  
  
Relaciones:  
Usuario → Rol (N:1)  
Rol → Permisos (N:M mediante rol\_permisos).

# 7. Flujo de Autenticación

1️⃣ Cliente envía username/password a /auth/login  
2️⃣ AuthService valida credenciales en base de datos  
3️⃣ JwtService genera token JWT firmado  
4️⃣ LogEventoService registra LOGIN exitoso  
5️⃣ Cliente usa Authorization: Bearer <token> en cada request  
6️⃣ JwtAuthenticationFilter valida token en cada solicitud

# 8. Endpoints Principales

/api/v1/auth/login – Autenticación y generación de token JWT  
/api/v1/auth/me – Obtener usuario autenticado  
/api/v1/auth/register – Registro de nuevos usuarios  
/api/v1/seguridad/usuarios – CRUD de usuarios  
/api/v1/seguridad/roles – CRUD de roles  
/api/v1/seguridad/permisos – CRUD de permisos  
/api/v1/seguridad/rol-permisos – Asignación de permisos a roles

# 9. Pruebas Reales (PowerShell)

1️⃣ Login y obtención de token:

Invoke-RestMethod -Uri "http://localhost:8080/api/v1/auth/login" `  
 -Method POST `  
 -ContentType 'application/json' `  
 -Body '{"username":"admin","password":"1234"}'

Respuesta esperada:  
{  
 "token": "eyJhbGciOiJIUzI1NiJ9...",  
 "username": "admin",  
 "rol": "ADMIN"  
}

2️⃣ Validar usuario autenticado:

$token = "<TOKEN>"  
Invoke-RestMethod -Uri "http://localhost:8080/api/v1/auth/me" `  
 -Method GET `  
 -Headers @{ Authorization = "Bearer $token" }

# 10. Estado Actual y Próximos Pasos

✅ Login funcional con JWT  
✅ Auditoría en seguridad.log\_evento  
✅ AccessValidator activo  
✅ CRUDs de usuarios, roles, permisos y rol\_permisos  
✅ Context path '/api/v1' establecido  
⚙️ Pendiente: Aplicar AccessValidator a los módulos de negocio (Hoja de Vida, Contabilidad, etc.) e integrar el frontend Angular con login y guardias de sesión.