Sistema de busqueda de libros: Gestión de Usuarios - Documentación Técnica

1. Descripción General del Sistema

El sistema es una aplicación web de gestión de usuarios desarrollada con Spring Boot, que ofrece funcionalidades de registro, autenticación, y administración de usuarios con diferentes roles.

2. Endpoints Implementados

2.1 Endpoint de Registro (/registro)

Detalles Técnicos

- **Método HTTP:** POST
- **Controlador:** RegistroController
- **Ruta:** /registro

Parámetros de Entrada:

- usuario: Nombre de usuario (único)
- email: Correo electrónico del usuario
- password: Contraseña
- confirmPassword: Confirmación de contraseña

Proceso de Registro:

1. Validación de coincidencia de contraseñas
2. Verificación de unicidad del nombre de usuario
3. Encriptación de contraseña con BCrypt
4. Asignación del rol predeterminado (ROLE_USER)
5. Persistencia del usuario en base de datos

Respuestas:

- Éxito: Redirección a página de login
- Error: Retorno a página de registro con mensaje de error

2.2 Endpoint de Login (/login)

Detalles Técnicos

- **Método HTTP:** POST
- **Controlador:** Manejado por Spring Security
- **Ruta:** /procesar_login

Proceso de Autenticación:

1. Validación de credenciales contra base de datos
2. Verificación de contraseña encriptada
3. Generación de contexto de seguridad
4. Asignación de roles de usuario

Flujo de Autenticación:

- Éxito: Redirección a /home
- Fallo: Redirección a /login?error=true
- **2.3 Endpoint de Home (/home)**

Detalles Técnicos

- **Método HTTP:** GET
- **Controlador:** HomeController
- **Ruta:** /home

Características:

- Requiere autenticación
- Muestra opciones según el rol del usuario
- Atributos del modelo:
    - isAdmin: Indica si el usuario tiene rol de administrador
    - isUser: Indica si el usuario tiene rol de usuario

Opciones Disponibles:

- Ver Perfil (para todos los usuarios)
- Gestión de Sistema (solo para administradores)
- Cerrar Sesión

2.4 Endpoint de Gestión de Usuarios (/admin/gestion-usuarios)

Detalles Técnicos

- **Método HTTP:** GET, POST
- **Controlador:** AdminController
- **Ruta:** /admin/gestion-usuarios
- **Rol Requerido:** ROLE_ADMIN

Funcionalidades:

1. **Listar Usuarios**

- Método: GET
- Muestra todos los usuarios (excepto el usuario actual)
- Muestra información: ID, nombre, email, rol

2. **Actualizar Usuario**
    - Método: POST
    - Ruta: /admin/actualizar-usuario
    - Permite modificar:
        - Nombre de usuario
        - Email
        - Rol (ROLE_USER o ROLE_ADMIN)
3. **Eliminar Usuario**
    - Método: POST
    - Ruta: /admin/eliminar-usuario
    - Permite eliminar usuarios por ID

2.5 Endpoint de Perfil (/perfil)

Detalles Técnicos

- **Método HTTP:** GET, POST
- **Controlador:** PerfilController
- **Ruta:** /perfil
- **Requiere Autenticación**

Funcionalidades:

1. **Ver Perfil**
    - Método: GET
    - Muestra información del usuario:
        - Nombre de usuario
        - Email
        - Rol
        - Contraseña enmascarada
2. **Actualizar Perfil**
    - Método: POST

- Ruta: /perfil/actualizar
- Permite modificar:
    - Nombre de usuario
    - Email
- Si cambia el nombre de usuario, cierra la sesión

3. Replicación del Código en Entorno Local

Requisitos Previos

- Java Development Kit (JDK) 21
- Maven 3.8+
- MySQL 8.0+

Pasos de Instalación

1. **Clonar Repositorio**
2. git clone <URL_DEL_REPOSITORIO>
3. cd HOLASPRING6CV3
4. **Configurar Base de Datos**
    - Crear base de datos MySQL
    - CREATE DATABASE tarea2;
    - CREATE USER 'admin'@'localhost' IDENTIFIED BY 'admin';
    - GRANT ALL PRIVILEGES ON tarea2.* TO 'admin'@'localhost';
5. **Configurar Archivo de Propiedades** Verificar src/main/resources/application.properties:
6. spring.datasource.url=jdbc:mysql://localhost:3306/tarea2
7. spring.datasource.username=admin
8. spring.datasource.password=admin
9. **Compilar Proyecto**
10. mvn clean install
11. **Ejecutar Aplicación**
12. mvn spring-boot:run

4. Manejo de Sesión Segura

Mecanismo de Autenticación

- **Implementación:** Spring Security
- **Tipo de Autenticación:** Basada en Sesión

Componentes de Seguridad

1. **Autenticación**
    - CustomUserDetailsService: Carga detalles de usuario
    - PasswordEncoder (BCryptPasswordEncoder): Encriptación de contraseñas
2. **Configuración de Seguridad** (SecurityConfig)
    - Configuración de rutas protegidas
    - Manejo de login y logout
    - Control de acceso por roles

Flujo de Sesión

1. Usuario se autentica
2. Spring Security genera una sesión
3. Se almacena Authentication en SecurityContextHolder
4. Tokens de sesión gestionados internamente
5. Cierre de sesión invalida el contexto de seguridad

Características de Seguridad

- Protección contra CSRF (deshabilitado en este ejemplo)
- Control de acceso basado en roles
- Rutas protegidas según privilegios
- Encriptación de credenciales
- Gestión de sesiones segura

Endpoints Protegidos

- /home: Requiere autenticación
- /admin/**: Solo accesible para ROLE_ADMIN
- /user/**: Solo accesible para ROLE_USER

5. Consideraciones de Seguridad

- Uso de encriptación de contraseñas
- Validación de entrada de usuario
- Control de acceso por roles

- Manejo de errores controlado

6. Conclusiones

El sistema proporciona una implementación robusta de gestión de usuarios con Spring Boot, ofreciendo funcionalidades de registro, autenticación y control de acceso seguro.