INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE COMPUTO

INGENIERÍA DE SOFTWARE

Práctica 1: parte individual del proyecto

Alumno: Omar Redondo González

Grupo: 6CV3

Fecha: 10/03/2025

Requerimientos para Sistema de Autenticación con Roles

1. Descripción General

Este documento detalla los requerimientos para un sistema de autenticación robusto con roles de administrador y usuario. El sistema permitirá a los usuarios estándar acceder únicamente a sus propios datos, mientras que los administradores podrán realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) sobre otros usuarios y administradores.

2. Especificaciones Funcionales

2.1 Sistema de Autenticación

2.1.1 Registro de Usuarios

- El sistema debe proporcionar endpoints para el registro de nuevos usuarios.
- Durante el registro se capturarán como mínimo:
    - Nombre completo
    - Correo electrónico (único en el sistema)
    - Contraseña (debe cumplir con políticas de seguridad)
    - Información de contacto básica
- Por defecto, los nuevos registros tendrán el rol de "Usuario estándar".
- Solo los administradores podrán asignar el rol de "Administrador" a otros usuarios.

2.1.2 Inicio de Sesión

- El sistema debe proporcionar endpoints para autenticación de usuarios.
- La autenticación se realizará mediante correo electrónico y contraseña.
- El sistema generará un JWT (JSON Web Token) tras la autenticación exitosa.
- El token deberá incluir información sobre el rol del usuario y su identificador.
- Se implementará un mecanismo de expiración de token adecuado.

2.2 Gestión de Roles y Permisos

2.2.1 Roles del Sistema

- **Administrador**:
    - Acceso completo a operaciones CRUD sobre todos los usuarios
    - Capacidad para asignar y revocar roles de usuarios
    - Acceso a sus propios datos personales
    - Acceso a estadísticas y logs del sistema

- **Usuario estándar**:
    - Acceso de lectura únicamente a sus propios datos
    - Sin acceso a datos de otros usuarios
    - Sin permisos para crear, actualizar o eliminar otros usuarios
    - Capacidad para actualizar su propia información personal

2.2.2 Control de Acceso

- Implementación de autorización basada en roles a nivel de controlador.
- Uso de anotaciones de seguridad (@PreAuthorize, @Secured) para proteger endpoints.
- Verificación de identidad para asegurar que un usuario solo acceda a sus propios recursos.

2.3 Operaciones CRUD

2.3.1 Operaciones para Administradores

- **Crear**:
    - Registrar nuevos usuarios
    - Asignar roles a usuarios
- **Leer**:
    - Consultar información de cualquier usuario
    - Listar todos los usuarios con capacidad de filtrado y paginación
- **Actualizar**:
    - Modificar información de cualquier usuario
    - Cambiar roles de usuarios
- **Eliminar**:
    - Desactivar o eliminar cuentas de usuario

2.3.2 Operaciones para Usuarios Estándar

- **Leer**:
    - Consultar únicamente su propia información personal
- **Actualizar**:
    - Modificar únicamente su propia información personal
    - Actualizar su contraseña siguiendo protocolos de seguridad

3. Especificaciones Técnicas

3.1 Arquitectura

- Aplicación basada en Spring Boot con arquitectura de capas:
    - Capa de presentación (controladores REST)
    - Capa de servicios (lógica de negocio)
    - Capa de acceso a datos (repositorios JPA)
    - Capa de seguridad (configuración Spring Security)

3.2 Seguridad

3.2.1 Autenticación

- Implementación mediante Spring Security
- Uso de JWT para gestión de sesiones
- Configuración de filtros de autenticación personalizados

3.2.2 Gestión de Contraseñas

- Encriptación mediante algoritmo BCrypt
- Implementación de políticas de contraseñas seguras
- Mecanismo de restablecimiento de contraseñas

3.3 Persistencia de Datos

3.3.1 Base de Datos

- MySQL como sistema gestor de base de datos
- Uso de Spring Data JPA para la capa de acceso a datos
- Diseño de esquema relacional para usuarios y roles

3.3.2 Modelo de Datos

- **Usuario**:
    - ID (clave primaria)
    - Nombre
    - Correo electrónico (único)
    - Contraseña (encriptada)
    - Estado (activo/inactivo)
    - Fecha de creación

    - Fecha de última modificación
- **Rol**:
    - ID (clave primaria)
    - Nombre (ROLE_ADMIN, ROLE_USER)
    - Descripción
- **Usuario_Rol** (tabla de relación):
    - ID_Usuario (clave foránea)
    - ID_Rol (clave foránea)

3.4 API REST

3.4.1 Endpoints de Autenticación

- POST /api/auth/register: Registro de nuevos usuarios
- POST /api/auth/login: Autenticación de usuario y generación de token JWT

3.4.2 Endpoints de Usuarios

- GET /api/users: Lista de usuarios (solo admin)
- GET /api/users/{id}: Obtener detalles de un usuario específico
- POST /api/users: Crear nuevo usuario (solo admin)
- PUT /api/users/{id}: Actualizar usuario existente
- DELETE /api/users/{id}: Eliminar usuario (solo admin)
- GET /api/users/profile: Obtener perfil del usuario actual

4. Configuración del Entorno

4.1 Dependencias Principales

- Spring Boot Starter Web
- Spring Boot Starter Security
- Spring Boot Starter Data JPA
- MySQL Connector
- jjwt (para manejo de JWT)
- BCrypt (para encriptación de contraseñas)

4.2 Configuración de la Base de Datos

Configuración en application.properties:

Properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/practica2?useSSL=false&ser
verTimezone=UTC
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

5. Pruebas y Validación

5.1 Pruebas de Seguridad

- Verificación de encriptación de contraseñas
- Validación de token JWT
- Pruebas de acceso no autorizado a endpoints protegidos

5.2 Pruebas Funcionales

- Registro exitoso de usuarios
- Inicio de sesión correcto
- Acceso a recursos según rol
- Operaciones CRUD según permisos de rol

FURPS+ para Sistema de Autenticación con Roles

F - Funcionalidad

Autenticación y Autorización

- **Registro de usuarios**
    - El sistema debe permitir el registro de nuevos usuarios capturando información básica (nombre, correo electrónico, contraseña).
    - Las contraseñas deben ser validadas según políticas de seguridad establecidas (longitud mínima, caracteres especiales, etc.).
    - Verificación de correo electrónico único en el sistema.
- **Inicio de sesión**
    - Autenticación mediante correo electrónico y contraseña.
    - Generación de tokens JWT con tiempo de expiración configurado.
    - Manejo de sesiones activas e inactivas.
    - Bloqueo temporal de cuenta tras intentos fallidos repetidos.
- **Gestión de roles**
    - Implementación de roles de administrador y usuario estándar.
    - Asignación de permisos específicos asociados a cada rol.
    - Verificación de permisos en cada solicitud a endpoints protegidos.
- **Operaciones CRUD**
    - Administradores:
        - Crear, leer, actualizar y eliminar cualquier usuario del sistema.
        - Asignar y modificar roles de otros usuarios.
        - Acceder a listado completo de usuarios con capacidad de filtrado.
    - Usuarios estándar:
        - Leer únicamente su propia información personal.
        - Actualizar sus propios datos personales.
        - Sin acceso a información de otros usuarios.
- **Auditoría de acciones**
    - Registro de actividades críticas (inicio de sesión, cambios de contraseña, modificación de roles).

    - Almacenamiento de fecha, hora, IP y acción realizada para cada actividad.

sistema, como:

U - Usabilidad

Interfaz de usuario

- **Formularios de autenticación**
    - Diseño intuitivo y limpio de formularios de registro y login.
    - Validación en tiempo real de campos con mensajes descriptivos de error.
    - Opción visible para recuperación de contraseña en pantalla de login.
- **Navegación según rol**
    - Menú de navegación adaptado a los permisos del usuario actual.
    - Ocultamiento de opciones no disponibles según el rol asignado.
    - Indicador visible del rol actual y sesión activa.
- **Gestión de perfiles**
    - Interfaz amigable para actualización de datos personales.
    - Proceso de cambio de contraseña con confirmación requerida.
    - Visualización clara de datos personales almacenados.

Experiencia de usuario

- **Mensajes informativos**
    - Retroalimentación clara sobre éxito o error en operaciones.
    - Notificaciones específicas para acciones prohibidas según rol.
    - Confirmación antes de realizar acciones destructivas (eliminar).
- **Accesibilidad**
    - Compatibilidad con lectores de pantalla.
    - Cumplimiento de contrastes de color adecuados.
    - Navegación posible mediante teclado.
- **Responsive design**
    - Adaptación a diferentes tamaños de pantalla (móviles, tablets, escritorio).
    - Experiencia de autenticación optimizada para dispositivos móviles.

R - Confiabilidad (Reliability)

Seguridad de datos

- **Protección de credenciales**
    - Encriptación de contraseñas mediante algoritmo BCrypt.
    - Almacenamiento seguro sin posibilidad de recuperación en texto plano.
    - Transmisión de credenciales únicamente bajo HTTPS.
- **Protección contra ataques**
    - Implementación de protección contra ataques CSRF.
    - Protección contra inyección SQL usando JPA/Hibernate.
    - Prevención de ataques de fuerza bruta mediante bloqueo temporal.

Disponibilidad del sistema

- **Manejo de errores**
    - Respuestas adecuadas y descriptivas para errores de autenticación.
    - Registro detallado de excepciones para facilitar diagnóstico.
    - Manejo de caídas de conexión a base de datos.
- **Tiempo de actividad**
    - Disponibilidad del servicio de autenticación 24/7.
    - Tiempo de inactividad programado mínimo para mantenimiento.
    - Plan de recuperación ante fallos.

Integridad de datos

- **Validación de entradas**
    - Verificación de formato y contenido en todos los campos.
    - Sanitización de entradas para prevenir XSS.
    - Validación en backend independiente de controles de frontend.
- **Consistencia de datos**
    - Transacciones atómicas para operaciones que afectan múltiples entidades.
    - Constraints a nivel de base de datos para integridad referencial.
    - Validación de estado consistente tras operaciones CRUD.

P - Rendimiento (Performance)

Tiempos de respuesta

- **Autenticación rápida**
    - Tiempo máximo de respuesta para login: 1 segundo bajo carga normal.
    - Tiempo máximo para operaciones CRUD: 2 segundos.
    - Optimización de consultas a base de datos para listas de usuarios.
- **Escalabilidad**
    - Soporte para 1000 usuarios concurrentes sin degradación significativa.
    - Capacidad de escalado horizontal mediante replicación de servicios.
    - Caché para tokens JWT validados frecuentemente.

Gestión de recursos

- **Uso eficiente de memoria**
    - Optimización de objetos de sesión.
    - Liberación adecuada de recursos después de operaciones.
    - Monitoreo de uso de memoria en periodos de alta carga.
- **Gestión de conexiones**
    - Pool de conexiones a base de datos dimensionado adecuadamente.
    - Cierre adecuado de conexiones tras completar operaciones.
    - Timeout configurado para evitar conexiones huérfanas.

S - Soporte

Mantenibilidad

- **Documentación**
    - Documentación detallada de API mediante Swagger/OpenAPI.
    - Manual técnico para desarrolladores con diagramas de clases.
    - Manual de usuario final para operaciones de autenticación.
- **Estructura de código**
    - Arquitectura limpia con separación clara de responsabilidades.
    - Convenciones de nombrado consistentes.
    - Comentarios descriptivos en secciones críticas del código.

Integración con herramientas

- **Gestión de proyectos**
    - Integración con ClickUp para seguimiento de tareas de mantenimiento.
    - Enlace desde errores de sistema a tickets en ClickUp.
    - Dashboard de estado de servicio integrado.
- **Monitoreo y logs**
    - Implementación de logging estructurado para facilitar análisis.
    - Integración con herramientas de monitoreo de rendimiento.
    - Alertas automatizadas para situaciones críticas.

Localización e internacionalización

- **Soporte multiidioma**
    - Mensajes de error y éxito traducibles.
    - Adaptación a formatos regionales (fechas, números).
    - Configuración de zona horaria para registro de actividades.

+ (Otros requisitos)

Requisitos de diseño

- **Patrones arquitectónicos**
    - Implementación de arquitectura de capas clara (controlador, servicio, repositorio).
    - Uso de patrón DTO para transferencia segura de datos.
    - Implementación de interfaces para permitir cambios en implementaciones futuras.

Requisitos de implementación

- **Tecnologías específicas**
    - Spring Boot como framework principal.
    - Spring Security para manejo de autenticación.
    - Spring Data JPA para acceso a datos.
    - Base de datos MySQL para persistencia.

Requisitos de interfaz

- **Comunicación con otros sistemas**
    - API REST documentada para interoperabilidad.

    - Formatos estándar para intercambio de datos (JSON).
    - Headers HTTP estandarizados para autorización.

Requisitos físicos

- **Entorno de despliegue**
    - Compatible con despliegue en entornos Linux, Windows y macOS.
    - Soporte para contenedores Docker.
    - Requisitos mínimos de hardware claramente definidos.

Restricciones legales

- **Conformidad con regulaciones**
    - Cumplimiento de estándares básicos GDPR para datos personales.
    - Política de privacidad accesible desde interfaces de registro.
    - Mecanismo de consentimiento para almacenamiento de datos.