Ejercicio Avanzado para Pasantes de Backend en Spring Boot

Descripción:

Construir una API REST en Spring Boot para un sistema de gestión de tareas colaborativo, donde se utilicen enums para manejar todos los textos del sistema. Todos los mensajes, títulos, y estados deben ser configurables a través de un archivo centralizado. Además, se añadirá un nivel avanzado de validación, lógica de negocio y documentación obligatoria con Swagger. Se debe generar un reporte en JasperReports con detalles de las tareas.

Requerimientos del Sistema:

1. Todos los textos y mensajes del sistema deben ser gestionados desde un enum centralizado.
2. Los usuarios deben tener atributos como ID, nombre, correo (único), rol y fecha de registro.
3. Los equipos deben incluir ID, nombre único, fecha de creación, creador, miembros y estado.
4. Las tareas deben incluir atributos como ID, título, descripción, estado, fecha límite, equipo, y usuario asignado.
5. Debe haber una colección de auditoría para registrar acciones importantes con atributos como usuario, acción, y timestamp.
6. Validar todas las entradas y manejar errores mediante un controlador global.
7. Documentar completamente la API utilizando Swagger.
8. Generar un reporte en JasperReports que incluya lo siguiente:
   - Un reporte individual por equipo con una tabla que liste todas las tareas asociadas.
   - La tabla debe incluir columnas para el título de la tarea, descripción, estado, fecha límite, y usuario asignado.

Requerimientos Técnicos:

1. Usar MongoDB como base de datos, con colecciones para usuarios, equipos, tareas y auditoría.
2. Crear un enum centralizado para gestionar textos, estados y mensajes de error.

3. Estructurar el proyecto en capas: Controller, Service, Repository y Utils.

4. Implementar generación de reportes dinámicos en JasperReports con datos de MongoDB.

5. Documentar la API utilizando Swagger para describir endpoints, solicitudes y respuestas.

Entrega Esperada:

1. Código fuente bien organizado, utilizando enums para todos los textos.

2. Documentación completa de la API generada con Swagger.

3. Implementación de reportes en JasperReports que permitan filtrar tareas por equipo.

4. Instrucciones claras para ejecutar el proyecto, incluidas en un archivo README.md.

5. Pruebas unitarias e integración cubriendo la funcionalidad del sistema.

Criterios de Evaluación:

1. Uso adecuado de enums para gestionar textos y estados.

2. Implementación de reportes dinámicos en JasperReports con datos precisos.

3. Manejo eficiente de validaciones y errores.

4. Modularidad y escalabilidad del código.

5. Cobertura de pruebas unitarias e integración.

6. Documentación detallada de la API en Swagger.

Extensiones Opcionales:

1. Implementar exportación de datos en formatos CSV o Excel.

2. Crear funcionalidad para cambiar dinámicamente el idioma del sistema.

3. Implementar paginación y filtros avanzados para la búsqueda de tareas.

4. Generar reportes adicionales como estadísticas de productividad por usuario.