Informe: Implementación de un Sistema de Autenticación basado en JWT para los servicios REST

Integrantes:

- Silem Nabib Villa Contreras
- María Alejandra Mantilla Coral
- Juan Sebastián Libreros García
- Andrés David Parra García

1. Contexto General

El objetivo de esta entrega fue implementar un sistema de autenticación basado en JWT para los servicios REST utilizando Spring Security, Mappers y DTOs. Este informe detalla lo que se logró, lo que quedó pendiente y las dificultades encontradas durante el desarrollo.

2. Integración de Controladores REST:

La implementación de los controladores REST sigue las especificaciones RESTful para garantizar una comunicación eficiente y estandarizada entre el cliente y el servidor. Cada controlador se encarga de gestionar un conjunto específico de operaciones relacionadas con una entidad del dominio de la aplicación. Además, todos utilizan servicios para realizar la lógica de negocio y mappers para transformar las entidades en DTOs (Data Transfer Objects) y viceversa. Los controladores realizados fueron los siguientes:

CommentsController: Gestiona las operaciones CRUD para los comentarios en las recetas.
RecipesController: Maneja la creación, actualización, eliminación y búsqueda de recetas.
MealPlannerController: Administra la planificación de comidas, incluyendo la creación, actualización, eliminación y búsqueda de planes de comidas.
IngredientController: Gestiona las operaciones CRUD para los ingredientes.
TagController: Maneja la creación, actualización, eliminación y búsqueda de etiquetas.
UserFollowController: Administra las operaciones de seguimiento de usuarios.
NotificationController: Gestiono la actualización del estado y búsqueda de las notificaciones de los usuarios.
StepController: Se encarga del CRUD para los pasos de una receta.

3. DTOs:
Se utilizaron DTOs (Data Transfer Objects) para el manejo de los datos enviados y recibidos. Estos se dividieron en dos carpetas: request y response. Los DTOs de **request** se encargan de encapsular los datos que llegan desde el cliente, mientras que los DTOs de **response** encapsulan los datos que se envían de vuelta al cliente.

4. Mappers
Se usaron mappers para transformar entre entidades y DTOs. Estos se pueden encontrar en la carpeta Mapper, algunos de ellos son: CommentMapper, IngredientMapper, MealPlannerMaper, RecipeMapper, TagMapper, UserFollowMapper.

5. Manejo de Excepciones
Se implementó el manejo adecuado de excepciones y códigos de error HTTP. Se utilizaron excepciones y códigos de error predeterminados como IllegalArgumentException para manejar errores comunes. Además, se crearon excepciones personalizadas para manejar casos específicos de la aplicación, las cuales están ubicadas en la carpeta exception. Algunas de estas excepciones personalizadas incluyen: EmailNotVerifiedException, InvalidResourceException, UserAlreadyExistsException, AlreadyFollowingException, UnauthorizedException, etc.

6. Integración de JWT y Security

6.1 Configuración de JWT:
En el proyecto, se implementó la configuración de JWT para manejar la autenticación y autorización de los usuarios. La clase **JwtTokenUtils** se encarga de generar, validar y extraer información de los tokens JWT.

6.2 SecurityFilter:
Para gestionar el acceso a los servicios públicos y restringidos mediante autenticación por Token JWT, se configuró un filtro de seguridad en la clase **JwtAuthenticationFilter**. Este filtro extiende **OncePerRequestFilter** y se encarga de interceptar cada solicitud HTTP para realizar la autenticación basada en JWT.

7. Creación de Pruebas usando Postman:
Se desarrollaron pruebas en Postman para verificar el correcto funcionamiento de todos los controladores REST especificados en el proyecto. Dichas pruebas fueron

implementadas mediante el uso de la funcionalidad post-response para validar las respuestas a las peticiones de forma automatizada.

Cada solicitud incluye la URL del endpoint, el método HTTP correspondiente (GET, POST, PUT, DELETE), y los headers necesarios, como el token JWT para la autenticación.

Se configuraron variables de entorno en Postman para almacenar valores dinámicos como el token JWT, URLs base, y otros parámetros necesarios para las pruebas.
Estas variables permiten reutilizar valores en múltiples solicitudes y facilitan la actualización de datos sin necesidad de modificar cada solicitud individualmente.

8. Despliegue del Backend:
El despliegue del backend se llevó a cabo en un servidor Tomcat ubicado en el PC xhgrid2 de la institución. A continuación, se describen los pasos y consideraciones tomadas durante el proceso:

8.1 Despliegue en el servidor Tomcat

Para el despliegue, se generó un archivo .war utilizando Gradle. Este archivo contiene la aplicación empaquetada y lista para ser desplegada en el servidor Tomcat. El archivo .war se copió directamente en la carpeta `webapps` del servidor Tomcat. Tomcat detectó automáticamente el nuevo archivo y desplegó la aplicación.

8.2 Configuración de variables de entorno

Debido a que el PC de despliegue es compartido, se decidió embeber las variables de entorno directamente en el código. Esto incluyó las credenciales de la base de datos y las credenciales de Google Cloud, con el fin de evitar conflictos con otros proyectos que pudieran estar en el mismo servidor.

9. Dificultades Encontradas:

- **Despliegue con Tomcat:** No sabíamos cómo especificar las variables de entorno sin ponerlas directamente en el código, pero el monitor nos resolvió la duda.
- **Tests de Postman:** Tuvimos problemas con la configuración de las variables de entorno en Postman, lo que causó fallos en algunas pruebas automatizadas.

- **Configuración Inicial de Spring Security:** La configuración inicial de Spring Security fue compleja debido a la necesidad de personalizar la autenticación y autorización.
- **Integración de JWT:** La integración de JWT presentó desafíos, especialmente en la gestión de la expiración de tokens y la renovación automática.

10. Reflexiones y Conclusiones

Esta entrega permitió profundizar en el uso de Spring Security y JWT para la autenticación en servicios REST, proporcionando un sistema de autenticación robusto y seguro. La integración de JWT presentó algunos retos, especialmente en la configuración de los tokens y su expiración, pero el conocimiento adquirido sobre estas tecnologías fue significativo. Las pruebas en Postman facilitaron la verificación de la funcionalidad de los endpoints y el control de errores. Además, el manejo de excepciones personalizadas y el uso de DTOs y mappers contribuyó a mantener la separación de responsabilidades y una estructura más ordenada en el proyecto. Aunque la configuración inicial fue desafiante, la experiencia obtenida nos ayudará a abordar futuros proyectos con mayor seguridad y conocimiento de las mejores prácticas en autenticación y autorización de servicios RESTful.