Universidad Alfonso X el Sabio

Programación Concurrente con Hibernate y JPA en un Sistema de Gestión de Bibliotecas

Diego Fernandez

NP: 138167

November 2023

Contents

1 Introducción

1.1 Problema

En una biblioteca pública grande, existen miles de libros y cientos de lectores que buscan pedir prestados, devolver y renovar estos libros. Además, los bibliotecarios deben ser capaces de agregar nuevos libros al sistema, eliminar libros obsoletos o dañados, y realizar un seguimiento de los préstamos de libros. Para manejar estas tareas de forma eficiente y segura, necesitamos desarrollar un Sistema de Gestión de Bibliotecas (LMS por sus siglas en inglés) que use Hibernate y JPA para interactuar con una base de datos SQL y que pueda manejar solicitudes concurrentes de manera segura.

1.2 Requisitos Tecnicos

- Diseñar e implementar un modelo de datos para la biblioteca. Esto debe incluir clases para libros, lectores, préstamos, y cualquier otra entidad que considere necesaria.
- Utilice Hibernate y JPA para mapear sus clases de dominio a las tablas de la base de datos.
- Proporcione una API que permita a los clientes (bibliotecarios y lectores) realizar las operaciones básicas de la biblioteca, como buscar libros, pedir prestados libros, devolver libros, renovar préstamos, agregar nuevos libros y eliminar libros obsoletos. (https://www.nigmacode.com/java/crear-api-rest-con-spring/)
- Implemente el control de concurrencia para evitar condiciones de carrera, por ejemplo, dos lectores que intentan pedir prestado el mismo libro al mismo tiempo.
- Implemente auditoría y control de versiones para realizar un seguimiento de quién hace qué y cuándo en el sistema.
- Utilice una caché para mejorar el rendimiento de las operaciones comunes, como buscar libros.
- Utilice pruebas unitarias e integración para verificar el correcto funcionamiento de su aplicación.

2 Desarrollo

2.1 Programas utilizados

- **Bootify.io:** Para la creación de la estructura utilizamos el programa en línea Bootify.io. Una herramienta que permite generar fácilmente estructuras básicas de proyectos utilizando Spring Boot, una tecnología comúnmente utilizada en el desarrollo de aplicaciones Java. Facilita la creación de aplicaciones con configuraciones predefinidas y plantillas listas para ser utilizadas, lo que agiliza el proceso de inicio de un nuevo proyecto.
- **IntelliJ:**Es un entorno de desarrollo integrado (IDE) que proporciona numerosas herramientas y características para la programación en Java y otras tecnologías. Es altamente reconocido por su soporte a Java, sus capacidades de refactorización, autocompletado inteligente, depuración avanzada y una amplia gama de complementos y características que facilitan el desarrollo de aplicaciones Java de manera eficiente y productiva.

- **PostgreSQL:** Es un sistema de gestión de bases de datos relacional de código abierto y altamente potente. Conocido por su fiabilidad, robustez y funcionalidades avanzadas, PostgreSQL ha ganado reconocimiento en el ámbito de las bases de datos empresariales y de aplicaciones web. Este sistema se destaca por su capacidad para manejar grandes cantidades de datos, ofreciendo soporte para consultas complejas, transacciones ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad), y una amplia gama de tipos de datos nativos y extensiones especializadas. Además, su naturaleza extensible permite a los usuarios crear funciones personalizadas, tipos de datos y extensiones para adaptarse a casos de uso específicos. Con una comunidad activa y comprometida, PostgreSQL se ha consolidado como una opción fiable y escalable para una variedad de aplicaciones y entornos empresariales.

- **Docker:** Es una plataforma de código abierto que permite crear y ejecutar aplicaciones en contenedores. Estos contenedores son unidades estandarizadas que incluyen el software y sus dependencias, facilitando su distribución y ejecución en diferentes entornos de manera consistente y eficiente.

2.2 Tecnologías y Herramientas

- **Java:** Es el lenguaje de programación principal utilizado en el desarrollo del sistema de gestión de bibliotecas. Java es ampliamente utilizado en aplicaciones empresariales y ofrece una sintaxis orientada a objetos, portabilidad y una gran comunidad de desarrolladores.

- **Spring Boot:** Se trata de un framework de desarrollo que simplifica la creación de aplicaciones Java. Proporciona configuraciones predefinidas, facilitando la configuración de nuevas aplicaciones y reduciendo la cantidad de código que debe escribirse manualmente.

- **Hibernate:** Es una biblioteca de mapeo objeto-relacional (ORM) en Java. Permite el mapeo de objetos Java a tablas de bases de datos relacionales, proporcionando una capa de abstracción para interactuar con la base de datos de manera orientada a objetos.

- **JPA (Java Persistence API):** Es una API estándar de Java para el mapeo objeto-relacional. Proporciona una interfaz común para trabajar con datos persistentes en aplicaciones Java. JPA define un conjunto de anotaciones que se utilizan para mapear clases Java a tablas de bases de datos relacionales.

2.3 Diseño de la Base de Datos

La base de datos del sistema de gestión de bibliotecas cuenta con varias entidades, que se encuentran relacionadas entre ellas

- **Libro:** Representa los libros almacenados en la biblioteca, con atributos como ID de libro, título, autor, género, etc.
- **Lector:** Describe a los usuarios de la biblioteca que pueden pedir prestados libros, con información como ID de lector, nombre, dirección, contacto, etc.
- **Préstamo:** Entidad intermedia entre Libro y Lector, permitiendo que un Lector pueda tomar prestados uno o varios Libros en un período determinado. Esta relación se representa mediante una relación muchos a uno (Many-to-One), donde un Libro puede estar asociado con múltiples Préstamos, pero en un momento dado, está en un único Préstamo. Un Lector puede tener varios Préstamos asociados a diferentes Libros.
- **Libro Prestado:** Esta entidad actúa como una relación entre Libro y Lector, almacenando información específica sobre cada préstamo, como la fecha de inicio del préstamo, la fecha de devolución esperada y el estado actual del libro durante el préstamo. Facilita el seguimiento de la información relacionada con los libros y los lectores durante el período activo de préstamo.

Figure 1: Entidades y sus relaciones

2.4 Despliegue del programa

Primero debemos de tener instalados los programas necesarios. Tras configurar de forma correcta los puertos a los que el IDE se tiene que conectar (5432), crearemos una base de datos que su nombre coincida con el de la escrita en el propio codigo, este mismo creara las entidades y las relaciones en la propia base de datos

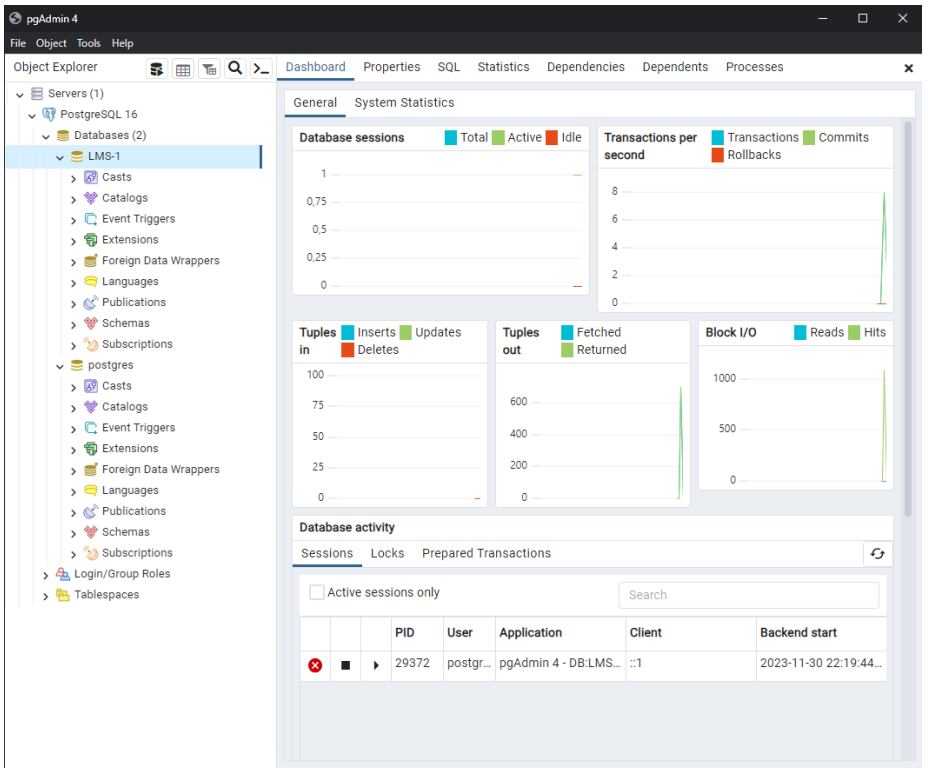

Figure 2: pgAdmin, utilizado para gestionar la base de datos

Tras esto en la terminal del IDE pondremos los siguientes comandos:

- **npm install:** Es un comando de Node Package Manager (npm) que se utiliza para instalar las dependencias de un proyecto "Node.js". Al ejecutar este comando en la raíz de un proyecto, npm busca en el archivo package.json las dependencias listadas y las descarga desde el registro npm, instalándolas localmente en la carpeta del proyecto dentro de la carpeta nodemodules.

- **npm run devserver:** Es un comando usado para iniciar un servidor de desarrollo. Este comando ejecuta un script definido en el archivo package.json, permitiendo iniciar y trabajar en un entorno de desarrollo específico para una aplicación web.

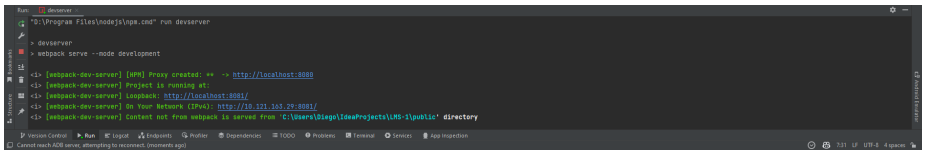

Figure 3: Resultados tras introducir los comandos

Posteriormente iniciaremos el propio programa en la clase Lms1Application, que iniciara el programa en el docker y nos permitirá usarlo en un navegador yendo a la dirección http://localhost:8080/

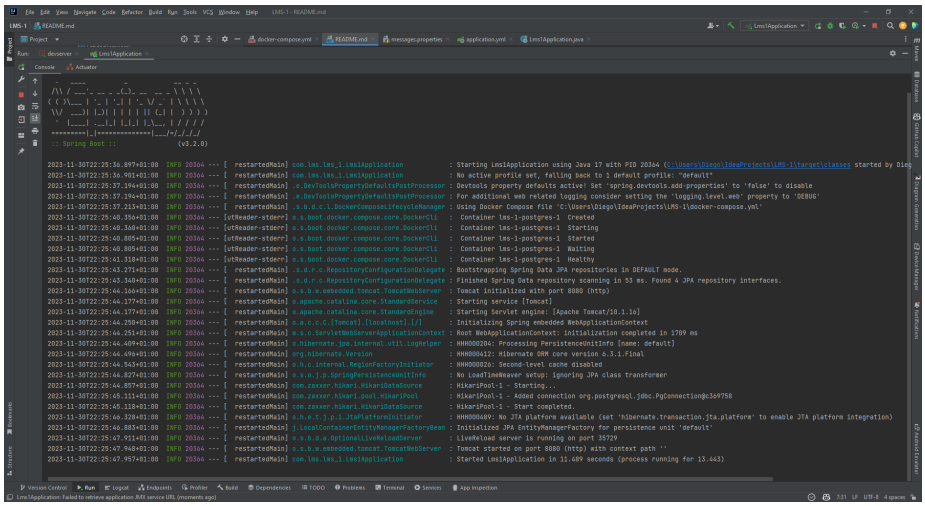

Figure 4: Resultados tras iniciar la aplicación

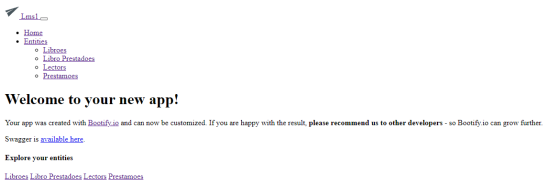

Figure 5: Pagina de inicio

3 Código

3.1 Dependencias

El codigo para el programa necesita varias dependencias para funcionar correctamente, estas son:

- **spring-boot-starter-web**
- **spring-boot-starter-validation**
- **spring-boot-starter-data-jpa**
- **postgresql**
- **spring-boot-starter-thymeleaf**
- **thymeleaf-layout-dialect**
- **springdoc-openapi-starter-webmvc-ui**
- **jakarta.persistence-api**
- **lombok** (opcional)
- **spring-boot-devtools** (alcance de ejecución y opcional)
- **spring-boot-docker-compose** (alcance de ejecución y opcional)
- **spring-boot-starter-test** (alcance de prueba)

3.2 Estructura

4 Conclusiones

El desarrollo del Sistema de Gestión de Bibliotecas (LMS) empleó tecnologías fundamentales como Spring Boot, Hibernate y PostgreSQL. Estas herramientas permitieron gestionar eficazmente el flujo de miles de libros y cientos de lectores.

La implementación de controles de concurrencia y pruebas exhaustivas aseguraron la prevención de conflictos comunes en bibliotecas.

La documentación clara y detallada se convirtió en una valiosa guía para los usuarios, facilitando la comprensión del sistema.

Este proyecto resaltó la importancia de herramientas robustas y una documentación precisa para solucionar los desafíos en la realización de proyectos de gran tamaño y complejidad

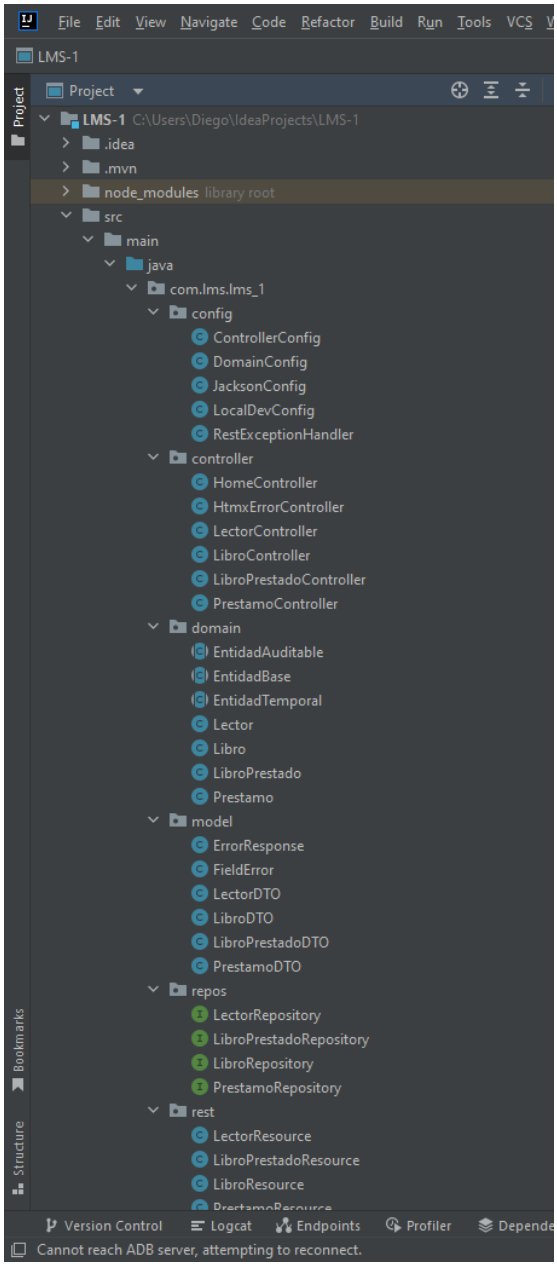

Figure 6: Pagina de inicio

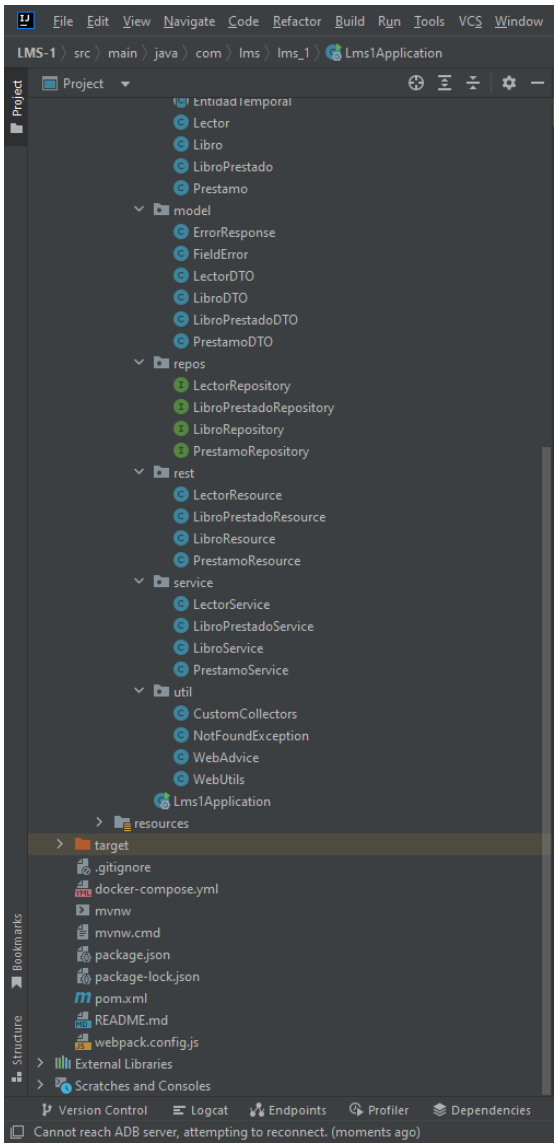

Figure 7: Pagina de inicio