

### GRADO EN INGENIERÍA INFORMÁTICA

#### Sistemas de Información de Gestión y Business Intelligence

## MEMORIA

**Autor:** Redon Kokaj

**Profesor:** Enrique López González

**Fecha:** 10/01/2023

## ÍNDICE

[INTRODUCCIÓN 3](#_TOC_250007)

[DESCRIPCIÓN DEL PROBLEMA 4](#_TOC_250006)

[HERRAMIENTAS 6](#_TOC_250005)

[APLICACIÓN 7](#_TOC_250004)

[ALGORITMO DE RECOMENDACIÓN 15](#_TOC_250003)

[DAFO 16](#_TOC_250002)

[LÍNEAS DE FUTURO 17](#_TOC_250001)

[LECCIONES APRENDIDAS 18](#_TOC_250000)

BIBLIOGRAFÍA Y ENLACES 19

# INTRODUCCIÓN

A lo largo de este trabajo analizaremos un problema concreto de recomendación y lo resolveremos mediante un algoritmo de recomendación. Para ello he realizado un programa donde se expone el problema.

El problema concreto es recomendar una serie de libros, especificando la categoría o un libro en sí mismo.

# DESCRIPCIÓN DEL PROBLEMA

El problema al que me he enfrentado ha sido realizar un programa de recomendación de libros. Cuando empecé a plantearme las diferentes opciones que tenía lo primero que pensé es de qué forma lo iba a hacer, si iba a recomendar en función del género, del autor, puntuación, ….

Después de pensarlo un poco, decidí que lo mejor era darte dos opciones: seleccionar directamente las categorías que te interesan o seleccionar un libro y recibir recomendaciones sobre libros similares.

Ante un sitio de recomendación de libros, creo que es muy importante que cualquiera pueda utilizarlo, desde el niño que busca un libro interesante para leer en su tiempo libre, hasta la persona mayor que busca un libro para leer en su tiempo libre.

Lo más difícil de todo esto, creo, es hacerlo sencillo y comprensible, porque aunque al final lo más importante es que la aplicación haga su trabajo.

Así que me esforzado en que además de cumplir con su función sea atractiva tanto para niños como para adultos.

# HERRAMIENTAS

La verdad es que al principio de este trabajo me encontré con varias dificultades, muchas de ellas relacionadas con las herramientas que iba a utilizar. Una vez elegido el tema sobre el que trabajar, tuve que encontrar un conjunto de datos que se ajustara a lo que quería conseguir. Después de consultar un montón de datos a través de kaggle, conseguí encontrar una base de datos que, aunque no era especialmente completa, podría cumplir la tarea para lo que estaba intentando hacer.

Así que el primer problema era asegurarme de que mi conjunto de datos estaba ordenado y limpio. Después procedí a limpiarlo (eliminando redundancias o incluso registros vacíos) y a eliminar columnas inservibles o simplemente inútiles.

La principal herramienta que tuve que utilizar fue NEO4J, que era totalmente desconocida para mí y debo decir que al principio era muy reacio a utilizarla. Me llevó un tiempo adaptarme, hice un par de cursos sobre Neo4J y al final conseguí cargar mi conjunto de datos.

Decidí tener un único informe para facilitar el filtrado vinculando los libros del conjunto de datos a sus géneros.

Inicialmente mi idea era hacer un sitio web con javascript. Esta idea se descartó al surgir problemas durante el desarrollo, como mi inexperiencia con el lenguaje y errores a los que no encontraba solución ni siquiera en línea.

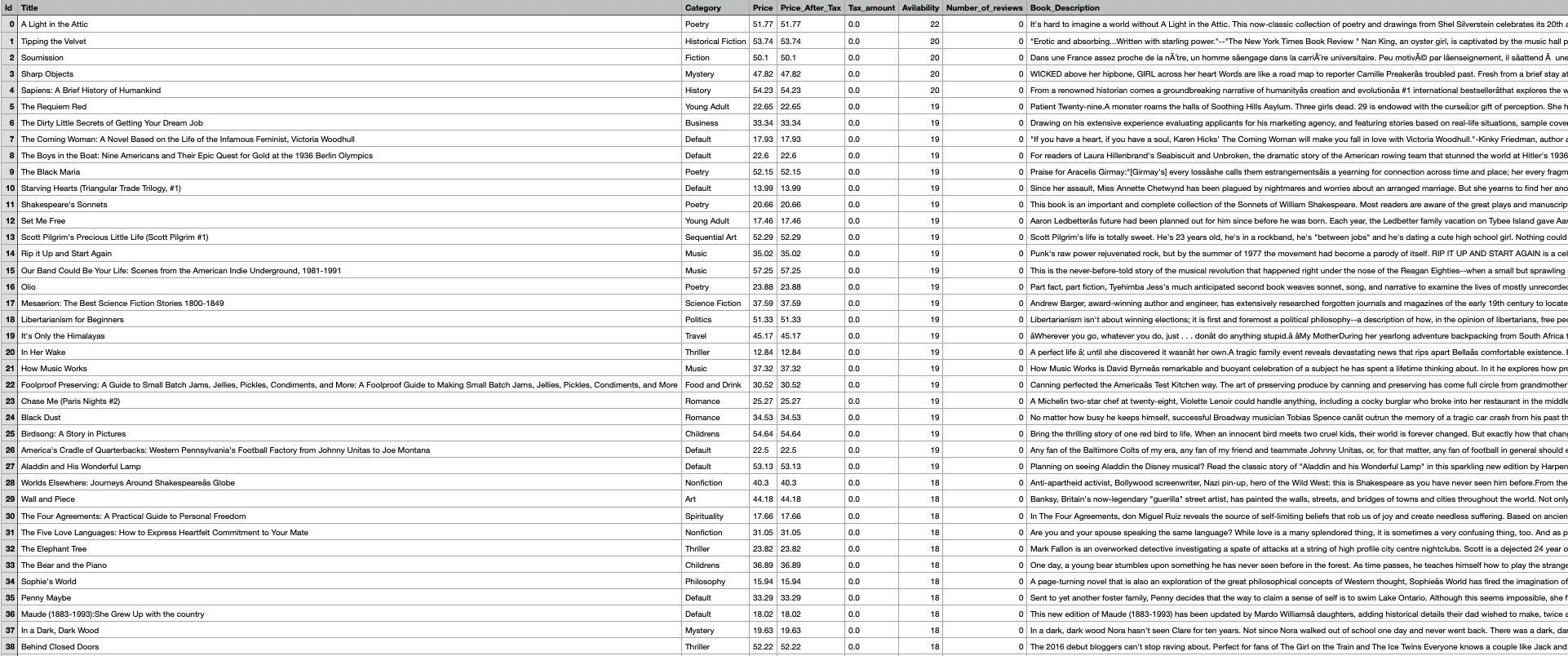
Utilicé Visual Studio para desarrollar la aplicación, ya que lo había utilizado antes y me sentía cómodo con él. Utilicé Java como lenguaje de desarrollo.

Una aplicación centrada en el usuario se divide en dos partes: el frontend, que es la parte centrada en el desarrollo de lo que verá el usuario, y el backend, que se centra en las conexiones entre la aplicación y la base de datos. Para el frontend, decidí usar la librería swing, ya que la había usado antes.

Para el backend he utilizado y controlador Neo4J para establecer la conexión con Neo4J.

# APLICACIÓN

Mi aplicación comienza con la creación de un conjunto de datos que luego utilizaré para crear la base de datos. Empecé a buscar en kaggle un conjunto de datos que se ajustara a lo que tenía en mente, hasta que encontré uno adecuado.



*(Pequeña muestra del conjunto de datos)*

Como puedes ver en el pequeño ejemplo, he decidido que cada fila del documento será un libro en la base de datos, y dentro de la fila tengo varios atributos, incluyendo el título del libro, la categoría, el precio, una breve descripción,...

También me aseguré de que hubiera una forma de evaluación con respecto al libro, que más tarde utilizaría para recomendar libros basándome también en esto.

Una vez que tuve el conjunto de datos, empecé a crear la base de datos, para lo cual utilicé Neo4J. Fue bastante fácil empezar a usarlo, hice dos cursos sobre cypher y afortunadamente no se desviaba demasiado del lenguaje sql que había estudiado antes.

Sin embargo, mirando hacia atrás y viendo los resultados, debo decir que en comparación con otras plataformas que ya había utilizado, como MySQL, Neo4j es mucho más intuitiva y visible una vez que ya has creado la base de datos.

He de decir que estoy muy contento de haber descubierto Neo4j y haber podido trabajar con él, ya que me permitió hacer todos los cambios que quería y trastear mucho con el conjunto de datos sin que eso me llevara horas de trabajo para incluir los cambios en la base de datos.



*(Base de datos)*

Aquí muestro mi base de datos. Los círculos que aparecen en naranja son los libros, es decir, los atributos que componen la base de datos. Los círculos morados son las categorías de los libros.

El único problema que tuve con la base de datos fue la fiabilidad de mi conjunto de datos básicos. Como dije antes, todo lo que tenía que hacer era eliminar redundancias y filas vacías, lo que pude hacer directamente con neo4j gracias a Cypher.

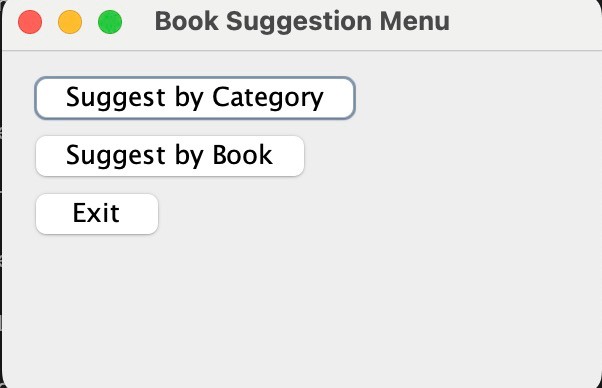
Una vez que todo estaba configurado en Neo4j, era el momento de crear lo que sería mi aplicación. Para ello, tuve que dividir el proyecto en dos partes: el frontend y el backend. El backend es la parte de la clase QueryDB, mientras que el frontend está en la clase Main.

La parte de la conexión de datos debo decir que me costó un poco y la verdad es que estuve varias horas investigando el uso de la librería neo4j en Java.

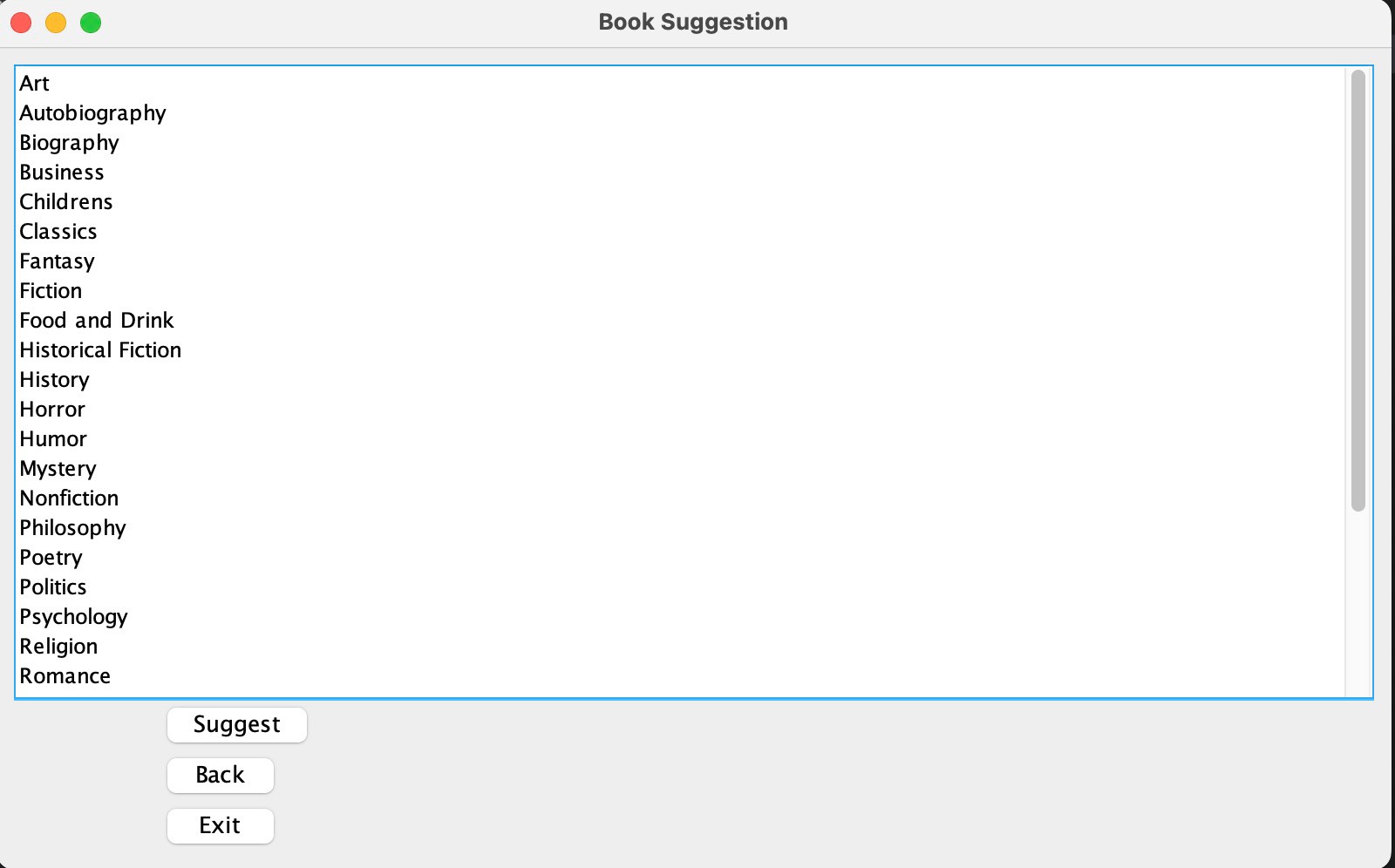
El funcionamiento del backend no es muy complicado, también debo decir que tengo un algoritmo de recomendación simple. Desde el frontend paso un libro o una categoría al backend y mediante una consulta obtengo todos los libros de una determinada categoría. Explicaré más en la sección sobre el algoritmo de recomendación.

Una vez que tengo todos los libros, los guardo en una lista y la envío a fronted, donde los expondré.

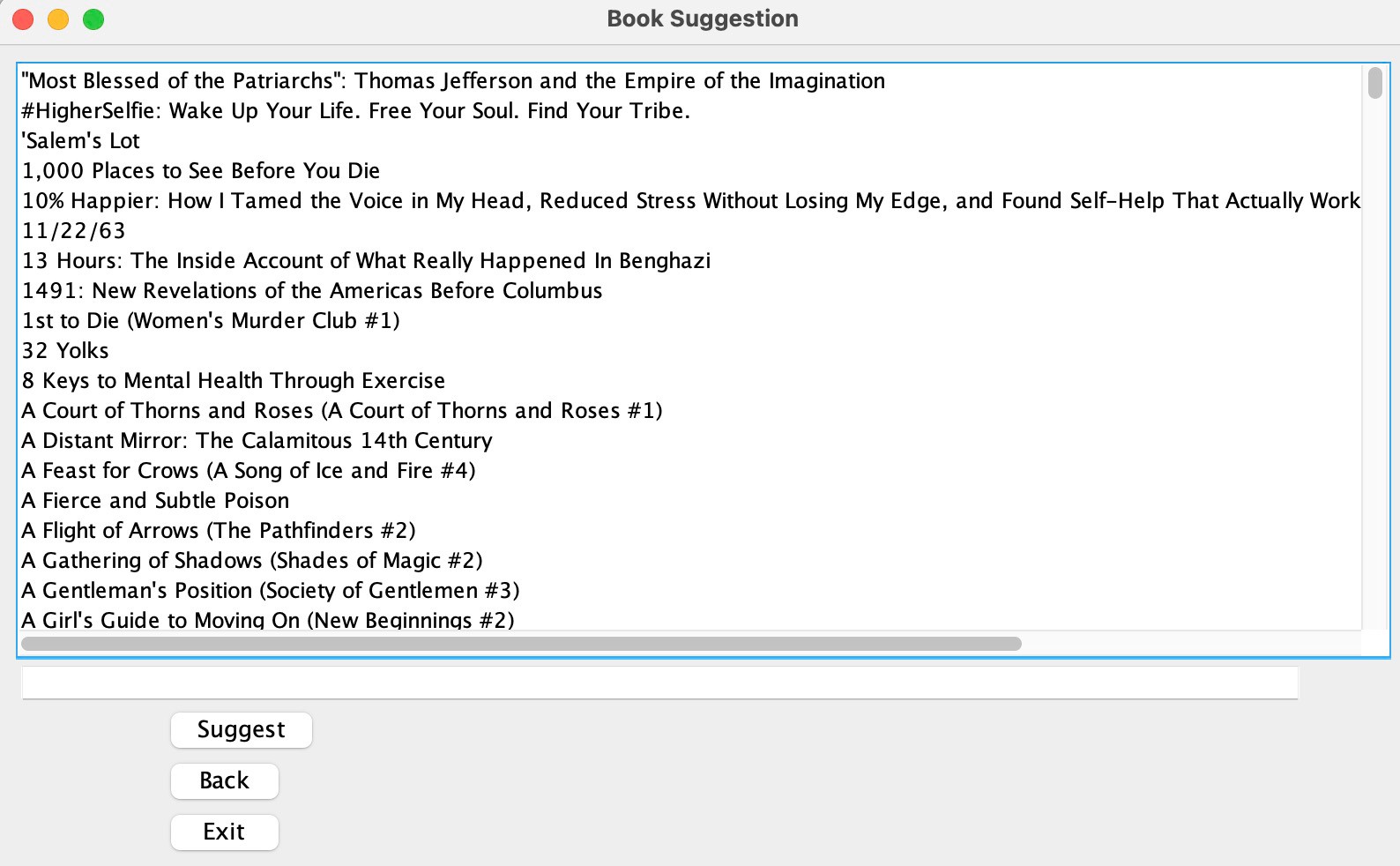
Pasemos al frontend. En un principio, mi idea era mostrar las categorías de libros inmediatamente y seguir a partir de ahí. Más tarde, discutiendo con mis colegas, pensé que era mejor ofrecer la posibilidad de hacer una sugerencia a partir de una categoría o de un libro, desarrollando así un frontend que servirá de menú principal, donde el usuario podrá seleccionar el tipo de sugerencia que más le guste.



(MENÚ PRINCIPAL)



(Sugerencia según categoría)

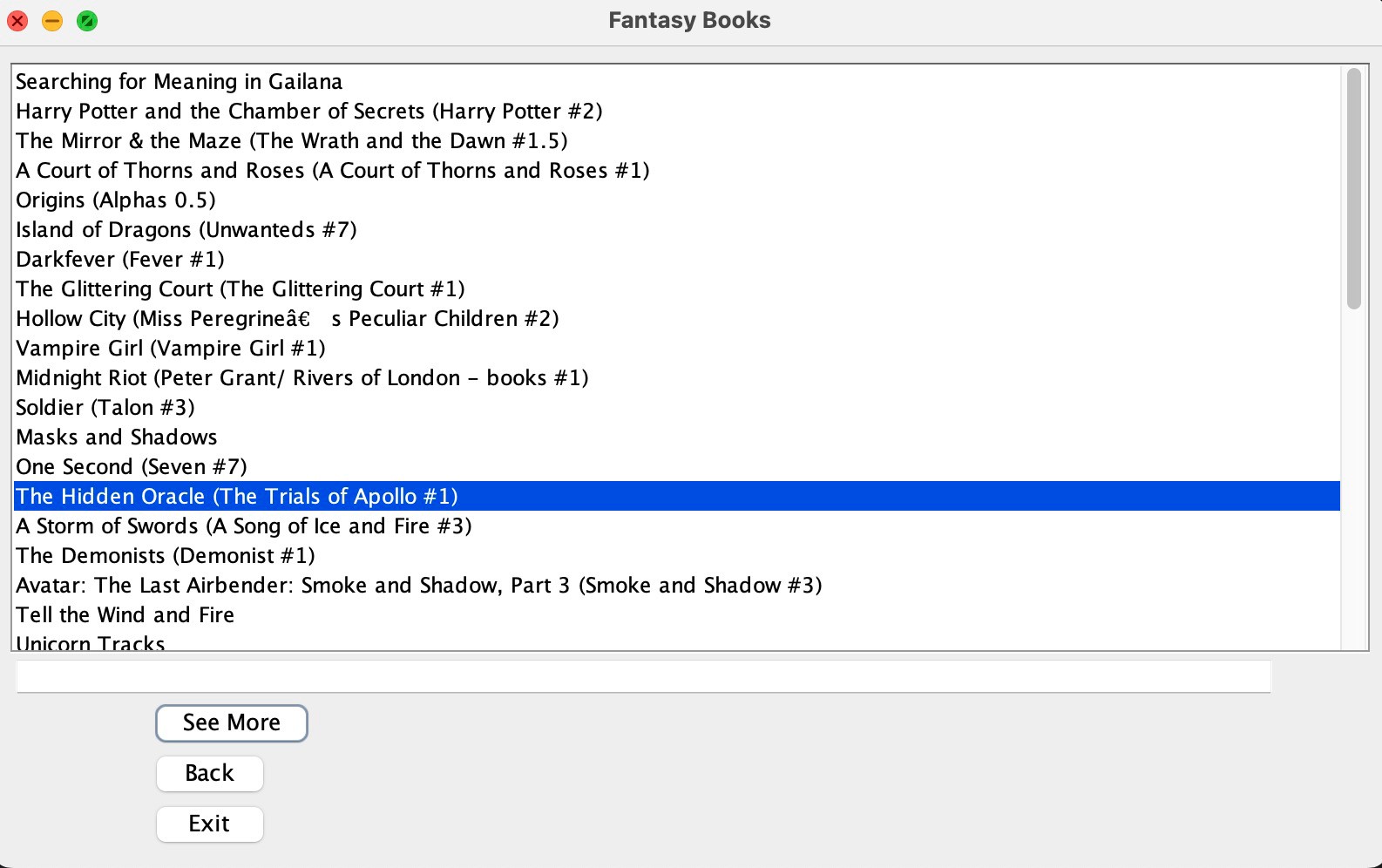


(Sugerencia según libro)

Ambas páginas disponen de 3 botones:

* Sugerir: una vez seleccionado el libro o la categoría, lo enviará al backend que realizará una consulta
* Atrás: para volver a la página anterior
* Salir: que cerrará el programa

Al mostrar los libros, por comodidad, decidí añadir una barra de búsqueda que actualiza lo que se muestra en tiempo real.



(Libros recomendados)

El título de la ventana cambiará según la sugerencia. Los libros mostrados están ordenados según la calificación recibida.

Se cambia un botón:

* Ver más: este botón servirá para mostrar la descripción del libro seleccionado



(Descripción del libro)

# ALGORITMO DE RECOMENDACIÓN

Mi algoritmo de recomendación no es muy complicado, básicamente consiste en un filtrado bastante sencillo: una vez que recibo el libro (o la categoría), deduzco la categoría y muestro los libros de ese tipo. Los ordeno por puntuación para recibir primero los resultados más populares de otros usuarios.

El frontend pasaría el libro o la categoría al backend y éste haría la consulta a la base de datos. La comunicación entre el frontend y el backend tiene lugar entre 'Main.java' y 'QueryDB.java'.

El backend envía la lista al frontend, y en esta ventana introducimos cada libro contenido, que ya ha sido previamente ordenado.

# DAFO

En esta sección estudiaré las limitaciones y ventajas de mi proyecto. Sé muy bien que mi aplicación es eficaz porque cumple su función, pero también sé que podría haber hecho mucho más.

La ventaja de mi aplicación es que no hay casos malos, siempre se obtendrá un resultado, independientemente de la aplicación, porque aunque mi conjunto de datos no es muy grande, con sólo 700 libros, es lo suficientemente completo como para obtener un resultado siempre.

También he de decir que me esforcé mucho en que fuera eficiente, que aunque el trabajo que realiza puede llegar a resultar simple, sea un buen trabajo y no de ningún error.

Una cosa que me hubiera gustado incluir es la posibilidad de que te guste un libro y guardarlo en la base de datos para tenerlo siempre a un clic.

Una ventaja es que tiene una capacidad de crecimiento muy buena: funciona igual de bien con doscientos libros que con diez mil, lo cual es muy ventajoso, porque nos permite garantizar que, a medida que crece la base de datos, no decae el rendimiento de la aplicación.

Me gustaría en un futuro poder hacer otra ventana con favoritos, y al poner un usuario y una contraseña, que te salgan los libros que te gustan o que te han parecido interesantes.

# LÍNEAS DE FUTURO

La verdad es que cuando empecé con este trabajo no sabía a donde llegaría ni que conseguiría. Una vez que he realizado el trabajo la verdad es que he visto muchas opciones de crecimiento.

En el futuro, me gustaría profundizar y perfeccionar la búsqueda, quizá incluso teniendo acceso a los gustos afines de terceros usuarios.

La verdad es que aunque estoy bastante contento con el trabajo realizado, cuanto más hacía más veía que podía hacer. Por ejemplo me hubiera gustado ahora al final, poder haber hecho un filtrado por género además de por edad, ya que si observa mi base datos ya está hecha la relación porque tenía esa idea desde el principio.

Otra cosa que me hubiera gustado es tener también los autores de los libros en la base de datos, para que también se puedan hacer recomendaciones a través de ella.

# LECCIONES APRENDIDAS

En primer lugar, quiero decir que estoy satisfecho con el trabajo que he realizado, porque es fruto de mi propio trabajo. Podría haber hecho mucho más, pero tengo que decir que tardé más de lo esperado en conseguir un conjunto de datos limpio y completo sobre el que trabajar.

La verdad es que aprendí mucho haciendo este trabajo, porque tuve que investigar mucho por mi cuenta, en el sentido de que no sabía cómo funcionaba Neo4j, y tuve que esforzarme mucho en aprender Cypher.

Aunque el proyecto inicial en javascript resultó más difícil de lo esperado y me desanimó, volver sobre mis pasos y desarrollarlo en un lenguaje familiar como Java me ayudó mucho.

La próxima vez, antes de lanzarme a un proyecto tan desafiante en un idioma desconocido para mí, me aseguraré de practicar primero para evitar quedarme atascado en trivialidades.

# BIBLIOGRAFÍA

* + Página de Neo4j:

https://neo4j.com/graphacademy/training-intro-40/02-neo4j-graph- platform/

* + Kaggle: https://[www.kaggle.com](http://www.kaggle.com/)