Sisimpur Library Management System

Entry Point: Starter Code

Sisimpur Library has been operating for quite some time. Due to its small size and nature, the library has been operating in traditional pen and paper. Recently it saw a spike in users and activity. Due to this sudden spike maintaining the system has been a headache. On top of that, the librarian is suspecting that someone resembling a fox is not returning the books at all. The management decided to maintain the library system using software and decided to ask for your help. They have already done the requirement analysis and also provided some skeleton project structure. You have to start from there.

Requirement Analysis

Entities

The system will comprise of 3 entities

- User
- Author
- Book

Book: Books are the heart and soul of a library. Admin will add, edit, delete the book entries from the system
Author: The books are logically bound with an author. Author management is also done from the admin.
User: The users can be registered in the system. They can lend and return books. For now, these operations are done by the Admin.

Role Management:

As mentioned earlier, all kinds of modifying operations (Insert, Update, Delete) will be done from the admin. Thus, we do not need any role management for now. Users can do some read only operations. However, these are not critical and thus can be done without any issue.

Functions

- Admin can create, edit, fetch, delete books
- Admin can create, edit, fetch, delete authors
- Admin can create, edit, fetch, delete users
- Admin can assign or unassign book(s) to a user (meaning user have lent the book(s))
- Users can search and filter books based on author name, book name, category, publishing year, availability

Extra

- Try to maintain best practices as much as you can
- Use proper error handling
-

Starting with the starter

A starter project has been given to you. The folder will have

- Spring boot project
- Database setup in docker
- API documentation in bruno docs

Note: you might need to add more tables or columns with the existing definitions

Installing Docker

We understand that not all of you are familiar with docker, but this will make your life a bit easier. Follow this guide to install docker in your system:
https://github.com/WCSCourses/index/blob/main/Docker_guide.md

Setting up the Database

There is a `ddl.sql` file in the `db` folder. If you want to add more tables or columns or want to add predefined data, you can do it there. Alternatively, you can set up flyway database migration if you want, but it is not required.

Notice that there is a `docker-compose.yaml` file present in the root directory. Run `docker compose up` command in the root directory.

Make sure the docker desktop (if you are on windows or mac) is running. You should see a new container spawn in the docker desktop UI.

Running Spring Boot Project

We have set up the project with the database connection configured. Run `./gradlew bootrun` to start the spring boot server. To check if everything is working, go to any browser (or postman or any other tool), and use `localhost:8080/api/v1/health`. You should see a greetings message.

Testing the Endpoints

We have provided some API documentation as bruno files in the `bruno` folder in root. Install Bruno from https://www.usebruno.com/downloads. From Bruno chose `collection > open collection` and chose the bruno folder in the project root. Alternatively, you can use the bruno extension from VS Code.

If you add new endpoints, then add them in the bruno docs.