Library Management System Documentation

GitHub : https://github.com/yoyokabo/SpringBoot-Library-RestAPI

- ## Requirements
  - Java 22
  - Maven
- ## Dependencies
  - Spring-Boot 3.1.10
  - Lombok
  - Spring Web
  - H2 Database
  - Spring Data JPA

Tables

Books

- ID: Long (Auto-generated)
- Title: String ****Required**
- Author: String ****Required**
- Publication Year: Integer
- ISBN: String ****Required**

Patrons

- ID: Long (Auto-generated)
- Name: String ****Required**
- Address : String
- Phone Number : String ****Required**

Borrowing

- ID: Serialized Composite Primary Key from Book ID and Patron ID
- Book ID: Long (Foreign Key to Book entity) ****Required**
- Patron ID: Long (Foreign Key to Patron entity) ****Required**
- Borrow Date: Date ****Required**
- Due Date: Date
- Return Date: Date (nullable)

API Endpoints

Book Endpoints

- **GET /api/books**
  - Description: Retrieve a list of all books.
  - Response: List of Book objects.
- **GET /api/books/{id}**
  - Description: Retrieve details of a specific book by ID.
  - Parameters: id (Integer)
  - Response: Book object.
- **POST /api/books**
  - Description: Add a new book to the library.
  - Request Body: Book object (excluding ID).
  - Response: Created Book object and HTTP Status 201.
- **PUT /api/books/{id}**
  - Description: Update an existing book's information.
    - Parameters: id (Integer)
    - Request Body: Updated Book object.
    - Response: Updated Book object and HTTP Status 200.
- **DELETE /api/books/{id}**
  - Description: Remove a book from the library.
  - Parameters: id (Integer)
  - Response: HTTP Status 204 (No Content).

Patron Endpoints

- **GET /api/patrons**
  - Description: Retrieve a list of all patrons.
  - Response: List of Patron objects.
- **GET /api/patrons/{id}**
  - Description: Retrieve details of a specific patron by ID.
  - Parameters: id (Integer)
  - Response: Patron object.
- **POST /api/patrons**
  - Description: Add a new patron to the system.
  - Request Body: Patron object (excluding ID).
  - Response: Created Patron object.
- **PUT /api/patrons/{id}**
  - Description: Update an existing patron's information.

    - Parameters: id (Integer)
    - Request Body: Updated Patron object.
    - Response: Updated Patron object.
- **DELETE /api/patrons/{id}**
    - Description: Remove a patron from the system.
    - Parameters: id (Integer)
    - Response: HTTP Status 204 (No Content) if successful.

Borrowing Endpoints

- **POST /api/borrow/{bookId}/patron/{patronId}**
    - Description: Allow a patron to borrow a book.
    - Parameters: bookId (Integer), patronId (Integer)
    - Request Body: Borrowing Objects (excluding return date , patron id , and book id)
    - Response: HTTP Status 201 if successful
- **PUT /api/return/{bookId}/patron/{patronId}**
    - Description: Record the return of a borrowed book by a patron.
    - Parameters: bookId (Integer), patronId (Integer)
    - Response: HTTP Status 200 (OK) if successful.

Data Storage

Data is stored in H2 and is validated with Spring Boot's @Valid annotation.

Data integrity is maintained using @Transactional annotation.

Input Validation is handled using @Valid annotation with a custom exception handler that returns proper information for bad requests. ex. Name Cannot be empty.

Unit and Integration Tests

Unit Tests for all of the services including (Books, Patrons , Borrowing) Services.

Integration Tests using live test environment for all controller including (Books, Patrons , Borrowing) Services.

Deployment tested using Postman to insure all behavior is as expected.