**Library Management System API Documentation**

1. **Prerequisites**

* Java 11 or higher
* Maven 3.6.0 or higher

## Getting Started

* Clone the Repository : git clone [orabi55555/maids (github.com)](https://github.com/orabi55555/maids).git
* cd library-management-system
* Build the Project : mvn clean install
* Run the Application : mvn spring-boot:run

## API Documentation

### Swagger UI

<http://localhost:8080/swagger-ui.html>

## Interacting with the API

* **Books**
  + POST api/books - Create a new book
  + PUT api/books/{id} - Update a book by ID
  + GET api/books - Get all books
  + GET api/books/{id} - Get a book by ID
  + DELETE api/books/{id} - Delete a book by ID
* **Patrons**
  + POST api/patrons - Create a new patron
  + PUT api/patrons/{id} - Update a patron by ID
  + GET api/patrons - Get all patrons
  + GET api/patrons/{id} - Get a patron by ID
  + DELETE api/patrons/{id} - Delete a patron by ID
* **Borrowing Records**
  + POST api/borrow/{bookId}/patron/{patronId} - Borrow a book
  + PUT api/return/{bookId}/patron/{patronId} - Return a borrowed book

## Request and Response Formats

All endpoints accept and return JSON data. Below are example request and response formats for key operations:

**Create a Book**:

**Request**: {

"title": "Clean Code",

"author": "Robert C. Martin",

"isbn": "123456789x",

"publicationYear": 2008,

"description": “this is a book about clean code”

}

**Response: {**

"id": 1,

"title": "Clean Code",

"author": "Robert C. Martin",

"isbn": "123456789x",

"publicationYear": 2008,

"description": “this is a book about clean code”

}

**Create a patron**

**Request:**{

"name": "John Doe",

"email": "john@email.com",

"phoneNumber": "01111111111",

}

**Response:**{

"id": 1

"name": "John Doe",

"email": "john@email.com",

"phoneNumber": "01111111111",

}

**Borrow a Book**:

**Request**: No request body needed.

**Response**: {

"id": 1,

"book": {

"id": 1,

"title": "Clean Code",

"author": "Robert C. Martin",

"isbn": "123456789x",

"publicationYear": 2008,

"description": “this is a book about clean code”

},

"patron": {

"id": 1,

"name": "John Doe",

"email": "john@email.com",

"phoneNumber": "01111111111",

},

"borrowDate": "2024-08-01T10:15:30",

"returnDate": null

}

## Error Handling

The API uses custom exception handling to return meaningful error messages. Common error responses include:

* **400 Bad Request**: Returned for validation errors.
* **404 Not Found**: Returned when a requested resource (e.g., book or patron) does not exist.
* **409 Conflict**: Returned for data integrity violations (e.g., duplicate entries).

### Example Error Response

{

"timestamp": "2024-08-01T10:15:30",

"error": "Not Found",

"message": "Book not found",

"path": "/books/999"

}

## Testing

To run the tests, use the following command: mvn test

The tests are located in the src/test/java directory and cover various scenarios to ensure the API behaves as expected