**Library management system**

**Document contains:**

* **Summary**
* **Business**
* **End points**
* **Next step**

**Summary:**

Library management system contain 3 entities book , borrowing and patron , this system allow librarian to manage books , patron and borrowing by using spring boot and mysql database.

**Business:**

We used spring boot to make our project . our project contain 3 entities:

Book: id , title , auther , publication year and ISBN.

Patron: id , name , and contact.

Borrowing: bookId , patronId , borrowingDate and returnDate.

We made some of apis to read , create , update and delete data , all of this end points working well . we handled exception just in borrowing service and controller.

We used MySQL database to store and manage our data.

**End Points:**

 Book management endpoints:

● GET /api/books: Retrieve a list of all books.

● GET /api/books/{id}: Retrieve details of a specific book by ID.

● POST /api/books: Add a new book to the library.

● PUT /api/books/{id}: Update an existing book's information.

● DELETE /api/books/{id}: Remove a book from the library.

 Patron management endpoints:

● GET /api/patrons: Retrieve a list of all patrons.

● GET /api/patrons/{id}: Retrieve details of a specific patron by ID.

● POST /api/patrons: Add a new patron to the system.

● PUT /api/patrons/{id}: Update an existing patron's information.

● DELETE /api/patrons/{id}: Remove a patron from the system.

Borrowing endpoints:

● POST /api/borrow/{bookId}/patron/{patronId}: Allow a patron to

borrow a book.

● PUT /api/return/{bookId}/patron/{patronId}: Record the return of a borrowed book by a patron.

**Next Step:**

1. Make unit and integration testing to test our code .
2. Make security to secure our app by using jwt and spring security.
3. Make cashing
4. Handle all exception in all end points and services