Report: ER Diagram, Relational Schema Design, and System Implementation

# Introduction

This report outlines the design and implementation of a Library Management System (LMS). The LMS aims to streamline the management of library resources such as books and users, including students and librarians. The system enables features such as book borrowing, return tracking, and student management.

# ER Diagram Design

The Entity-Relationship (ER) Diagram represents the core entities and their relationships within the library system. The diagram defines key entities such as ***Book***, ***Student***, ***Librarian***, and ***BorrowedBook***. Relationships between these entities demonstrate how the system functions.

## Entities:

Book: Contains information about the books available in the library.

Student: Represents students registered in the system.

Librarian: Represents the librarians who manage book inventory and lending.

School: Stores the school of the university e.g. Natural Science, Law etc.

*BorrowedBook*: Keeps track of which student borrowed which book and the due date.

ReservedBook: Keeps track of which student reserved which book.

PastPapers: Keeps a record of the Title and file path of each Past Papers uploaded by librarians

Penalties: Represents paid and unpaid fines for late book returns

## Attributes:

* Book:
* book\_id (Primary Key)
* title
* author
* publication\_year
* isbn
* edition
* genre
* description
* available\_copies
* total\_copies
* location
* Student:
* student\_id (Primary Key)
* first\_name
* last\_name
* email
* password
* school (Foreign Key)
* program
* year\_of\_study
* Librarian:
* librarian\_id (Primary Key)
* first\_name
* last\_name
* email
* password
* BorrowedBook:
* borrow\_id (Primary Key)
* book\_id (Foreign Key)
* student\_id (Foreign Key)
* borrow\_date
* due\_date
* return\_date
* status

## Relationships:

* A *student* can borrow many Books (1-to-many relationship).
* A *librarian* manages the book inventory and lending process.
* The *borrowed\_book* entity tracks which student borrowed which book and includes details like the due date.

# Relational Schema Design

The Relational Schema is derived from the ER Diagram and provides a clear structure for how data will be stored in the database. Each entity is mapped to a table, and foreign keys represent relationships between tables.

# System Implementation

The system was implemented using Fastify for the backend, React for the frontend, and PostgreSQL as the database. The following sections provide a breakdown of each component.

## Backend (Fastify):

* The backend API handles CRUD operations for the entities defined above (e.g., adding books, registering students, lending books). Each operation is exposed as a RESTful endpoint.

### Routes include:

GET /books: Retrieve all books.

POST /books: Add a new book.

PUT /books/:id: Update a book's information.

DELETE /books/:id: Delete a book from the system.

POST /borrow: Record a book borrowing event.

And More

### Session Management:

The backend uses session-based authentication for librarians and students to login.

## Frontend (React):

* The frontend provides a user-friendly interface for librarians and students to interact with the system.
* Forms are used for student lookup, adding books, and managing borrow/return operations.
* A Dashboard is provided for librarians to manage the books and borrowing process.

## Database (PostgreSQL):

* The PostgreSQL database stores all entities and relationships, maintaining the integrity of the data using foreign key constraints.
* Optimizations such as indexing were implemented on key fields like ***book\_id*** and ***student\_id*** to improve query performance.

# System Features

## Book Borrowing:

* Librarians can lend books to students. The system tracks the due date, and a notification system can remind students of upcoming due dates (future work).

## Book Inventory Management:

* Librarians can add, update, or remove books from the system.

## Student Management:

* Students are registered in the system with their details, and librarians can search for students by their student\_id.

## Book Reservation:

* Student can reserve books then go pick it up from the library to make the borrowing process simpler and faster.

# Challenges and Solutions

* Data Integrity: Ensuring data consistency between related entities was a challenge. Foreign key constraints and careful handling of CRUD operations solved this issue.
* Concurrency: To manage concurrent book borrows, locks on certain database operations ensured that a book wasn't borrowed by multiple students simultaneously.

# Conclusion

The Library Management System successfully implements core features of a typical library system, such as managing books, students, and borrowing operations. The ER diagram and relational schema design ensured a robust data model, while the Fastify backend and React frontend provided a seamless user experience.

Future improvements include adding a notification system for overdue books and expanding the system to include book reservations and more granular reports.