# Library Management System with basic functionalities:

**User Functionalities**

1. User registration
2. User login
3. User can see list of books
4. User can see specific book
5. User can borrow a book
6. User can return a book

**Admin Functionalities**

1. Add new book
2. Update a book
3. Manage units of book
4. Fetch all the books
5. Fetch specific book
6. Delete a book
7. Delete all books
8. Get which book which user have

**Entities**

1. **Users**

* **ID**: A unique identifier for each user.
* **Name**: The name of the user.
* **Email**: The email address of the user.
* **Password**: A hashed version of the user’s password.
* **Role**: The role of the user (e.g., 'admin', 'member').

1. **Books**

* **ID**: A unique identifier for each book.
* **Title**: The title of the book.
* **Author**: The author of the book.
* **ISBN**: A unique International Standard Book Number for the book.
* **Copies Available**: The number of copies of the book that are available for borrowing.
* **Total\_Copies**: The number of copies of the book that are available for borrowing.

1. **Loans**

The represent represents the borrowing transactions within the library system. This entity tracks which books are borrowed by which users, and when. Key attributes include:

* **ID**: A unique identifier for each loan transaction.
* **Book ID**: The ID of the book being borrowed.
* **User ID**: The ID of the user borrowing the book.
* **Borrowed Date**: The date when the book was borrowed.
* **Due Date**: The date when the book is due to be returned.
* **Returned Date**: The date when the book was actually returned (if applicable).

**API Endpoints:**

User Endpoints:

1. User Registration

* Route: POST/users/register
* Description: Register a new user.
* Request Body: in Json

{

"name": "Iram Abid",

"email": "Iram.abid@example.com",

"password": "password123"

}

* Response: Json

{

"id": 1,

"name": "Iram Abid",

"email": "iram.abid@example.com"

}

1. User login

* Route: POST/domain/users/login
* Description: user login into system.
* Request Body: in Json

{

"email": "Iram.abid@example.com",

"password": "password123"

}

* Response: Json

{

"session\_id": 1,

"jwt\_token":"jwt\_token"

}

1. See List of Books b

* Route: GET /user/user\_id/books
* Description: Retrieve a list of all books.
* Response: Json

[

{

"id": 1,

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"isbn": "9780743273565",

"copies\_available": 3

},

...

]

1. See Specific Book

* Route: GET /user/user\_id/books/{book\_id}
* Description: Retrieve details of a specific book.
* Response: Json

{

"id": 1,

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"isbn": "9780743273565",

"copies\_available": 3

}

1. Borrow a Book

* POST: /user/user\_id /loans/borrow/book\_id
* Description: Borrow a book.
  + Request Body: Json

{

"book\_id": 1,

"user\_id": 1,

}

* + Response: Json

{

"id": 1,

"book\_id": 1,

"user\_id": 1,

"borrowed\_date": "2023-01-01",

"due\_date": "2023-01-15"

}

1. Return a Book

* Route: POST user/user\_id/loans/return/book\_id
* Description: Return a borrowed book.
* Request Body: Json

{

"loan\_id": 1

}

* Response: Json

{

" message": "Book returned successfully."

}

Admin Endpoints:

* 1. Add New Book
* Route: POST admin/books/add
* Description: Add a book
* Request Body: in Json

{

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"isbn": "9780743273565",

"copies\_available": 3

}

{

"id": 1,

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"isbn": "9780743273565",

"copies\_available": 3

}

* Response: Json
  1. Update a Book
* Route: PUT admin/books/id
* Description: Update a book
* Request Body: Json

{

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"isbn": "9780743273565",

"copies\_available": 2

}

* Response: Json

{

"id": 1,

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"isbn": "9780743273565",

"copies\_available": 2

"total\_copies": 5}

1. Manage book units

* Route: PUT admin/books/{book\_id}/copies
* Description: to update book units
* Request: Json

{

"copies\_available": 5, "total\_copies": 5

}

* Response: Json

{

"id": 1,

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"isbn": "9780743273565",

"copies\_available": 5

"total\_copies": 5}

1. See Specific Book

* Route: GET admin/books/{book\_id}
* Description: Retrieve details of a specific book.
* Response: Json

{

"id": 1,

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"isbn": "9780743273565",

"copies\_available": 5

"total\_copies": 5}

1. Delete a Book

* Route: DELETE admin/books/{id}
* Description: Delete a book.
  + Response: Json

{

"message": "Book deleted successfully."

}

1. Delete All Books

* Route: DELETE admin/books/
* Description: Delete a book.
* Response: Json

{

"message": "All books deleted successfully."

}

1. Get Which Book Which User Have
   * Route: GET admin/loans
   * Description: Retrieve a list of all loans, showing which book each user has borrowed.
   * Response: Json

[

{

"id": 1,

"book\_id": 1,

"book\_title": "The Great Gatsby",

"user\_id": 1,

"user\_name": "Iram Abid",

"borrowed\_date": "2023-01-01",

"due\_date": "2023-01-15"

},

...

]

SCHEMA:

USERS TABLE:

ID: SERIAL Primary key

NAME: Name of User

EMAIL: email of user (unique)

PASSWORD\_HASH: password of user NOT NULL

ROLE: role of user (member or admin)

BOOKS TABLE:

ID: Primary key

NAME: Name of User

AUTHOR: email of user (unique)

ISBN: ISBN number of books NOT NULL

COPIES\_AVAILABLE: copies of books available to borrow

TOTAL\_COPIES: total copies of books

LOANS TABLE:

ID SERIAL PRIMARY KEY,

BOOK\_ID INT books(id),

USER\_ID INT users(id),

BORROWED\_DATE date borrow on,

DUE\_DATE DATE date for returning,

RETURNED\_DATE date book returned on