Library Management System
(AppDev-1 Project)

Name: Harikrishnan k
Roll No: 23f1000774
Email: 23f1000774@ds.study.iitm.ac.in

Introduction:

This is a web app designed to streamline the operations of a library.
This platform can be used for reading/managing books.
Web app provides user-friendly interface for users for efficient book management.

Technologies Used:

- **Flask**: A lightweight Python web framework used for building the backend and handling HTTP requests and responses.
- **Jinja2**: A templating engine for Flask, used for rendering dynamic HTML templates on the server-side.
- **SQLAlchemy**: A Python SQL toolkit and Object-Relational Mapping (ORM) library for interacting with the database.
- **Bootstrap**: A popular front-end framework for building responsive and visually appealing user interfaces.
- **Flask-RESTful API**: An extension for Flask that simplifies the process of building RESTful APIs.
- **SQLite3**: To create the lightweight database for the app.
- **Chart js**: A JavaScript library for creating charts.
- **PDF js**: For reading pdf in browser

System Architecture:

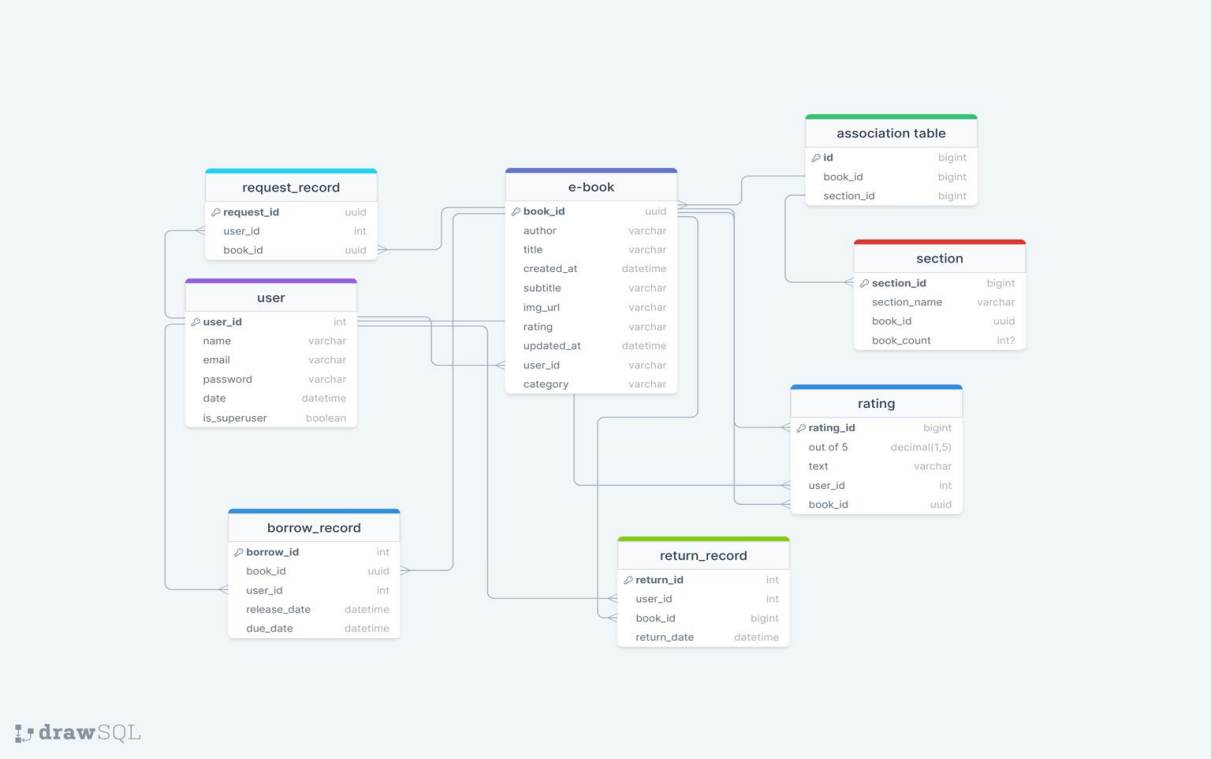
The Library Management System follows a MVC architecture, with the front-end built using HTML, CSS, and JavaScript, and the back-end developed using Flask and SQLAlchemy. SQLAlchemy is used for interacting with the database. Bootstrap ensures a responsive and visually appealing user interface.

Features Implemented:

- Secure user login and registration.
- Search functionality for easy book retrieval.
- User profile management.
- View borrowing history, pending requests, returned books.
- Proper validation for all forms.
- Admin can add, edit, and delete books.
- Admin can manage sections.
- Admin dashboard.
- API’s for book and section management.
- API authentication using JWT
- CRUD using API’s on books and sections

Database Design:

https://drawsql.app/teams/lms-22/diagrams/lms



RESTful API Endpoints:

https://documenter.getpostman.com/view/16763912/2sA2xb7G9N

- Login - GET http://127.0.0.1:5000/api/login
- Books - GET http://127.0.0.1:5000/api/books
- Book by id - GET http://127.0.0.1:5000//api/book/{book_id}
- Add book - POST http://127.0.0.1:5000//api/add-book
- Update book - PUT http://127.0.0.1:5000//api/update-book/{book_id}
- Delete book - DELETE http://127.0.0.1:5000//api/delete-book/{book_id}
- Sections - GET http://127.0.0.1:5000/api/sections
- Section by id - GET http://127.0.0.1:5000/api/sections/{section_id}
- Add section - POST http://127.0.0.1:5000/api/add-section
- Update section - PUT http://127.0.0.1:5000/api/update-section/{section_id}
- Delete section - DELETE http://127.0.0.1:5000/api/delete-section/{section_id}

To run the app:

Unzip the project folder, create a virtual environment, activate it and install all the dependencies from requirements.txt and run the main.py file.

Presentation video:

https://drive.google.com/file/d/16n7c0YUGga_5cKgSd_2xRNWBkt6yjEDS/view?usp=sharing