Author

Deepansh Garg
22f2000876
22f2000876@ds.study.iitm.ac.in
I am currently pursuing Btech in Netaji Subhash University of Technology in Electronics and communication specialisation in AI and ML. I will be graduating in 2026

Description

The project involves developing a library management system using Flask framework, SQLite for database management, and HTML/CSS for frontend design. Tasks include creating database schemas, implementing add, remove, update operations for books and users, designing user interfaces for browsing and searching books.

Technologies used

1) Flask: Micro web framework for building the backend of the application, providing routing, request handling, and templating.
2) Jinja2: Templating engine for generating dynamic HTML content.
3) HTML/CSS: Frontend markup and styling languages used for creating user interfaces for browsing books, managing user accounts, and interacting with the system.
4) Flask-SQLAlchemy: Flask extension for ORM (Object-Relational Mapping), facilitating interaction with the SQLite database through Python objects, simplifying database operations.

DB Schema Design

The database schema is designed to efficiently manage users, books, book requests, ratings, and feedback in a library management system. Here's the breakdown of the structure, columns, and constraints:

>Users:
Store user details like username, email, and password.
>Admins:
Manage admin accounts with username and password.
>Books:
Track book information such as title, author, content, and PDF path.
Includes details on availability, quantity, issue and return dates.
>Sections:
Organise books into different sections with names and descriptions.
>Book Requests:
Record user requests for books, including status and timestamps.
>Rating & Feedback:
Collect user ratings and feedback for books.

Constraints ensure data integrity:
Primary Keys for unique identification.
Unique Constraints prevent duplication.
Foreign Keys maintain relationships.
Not Null Constraints ensure essential fields are populated.
Default Values set initial values for certain fields.

Architecture and Features

The project follows the typical structure of a Flask application with the Model-View-Controller (MVC) architectural pattern:

>Models (User, Admin, Book, BookIssued, Section, BookRequest, RatingFeedback) define the data structure.
>Views (HTML templates in templates/) present the user interface.
>Controllers (@app.route functions) handle logic and interact with models and views.

Controllers are organised in the routes/ directory, each file representing a different functionality area like user management or book handling. Templates are stored in templates/, while static assets are in static/.

As for features implemented, the project includes default features common to a library management system, such as user registration and authentication, book listing, searching, and borrowing. Additional features may include:

Admin Panel: Implemented using Flask-Admin to provide administrators with a web interface for managing users, books, and sections efficiently.
Book Requests: Implemented endpoints and logic for users to request books, with administrators having the authority to approve or reject requests.
Ratings and Feedback: Implemented functionality for users to rate books and provide feedback, enhancing user engagement and facilitating book recommendations.
Section/Book Management: Implemented CRUD operations for managing sections/categories of books, allowing administrators to organise books effectively.

Video

Video Link