IITM Modern Application Development - 2

Library Management System V2

Thanus Kumaar A - 22f3000726

Project statement:

It's a multi-user app that is used to issue, grant/revoke and maintain e-books across various sections like an online library

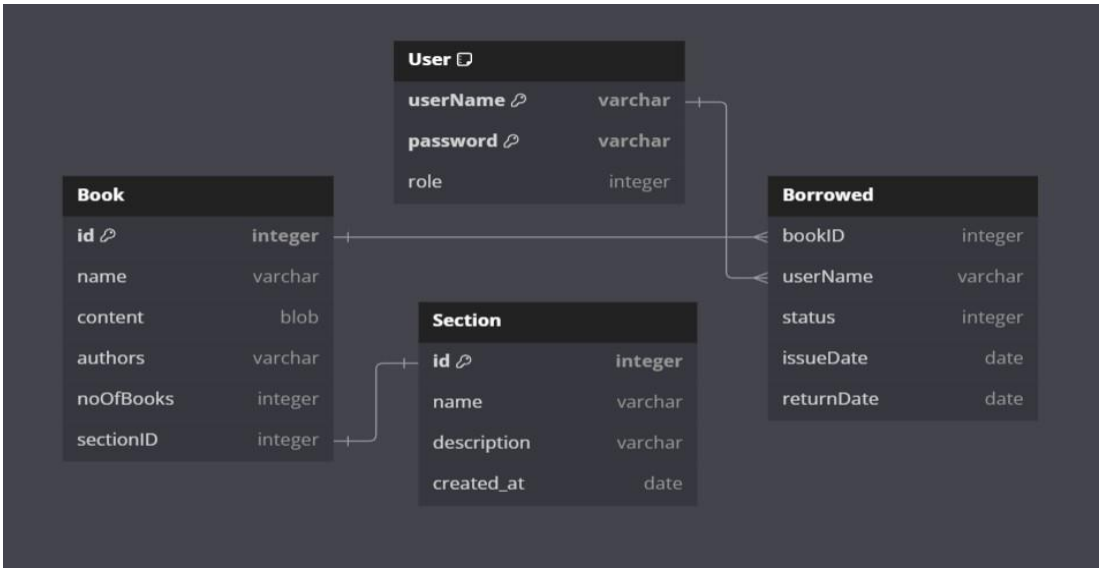
Description:

This is a library management system for multiple users. Both librarians and users can sign up and login in the application. Librarians and users will be redirected to their respective home pages. The admis will have the options to manage Sections (Add, Edit, Delete sections), manage Books (Add, Edit, Delete books), view signed up users, search for books based on authors or book name and issue or revoke books which are requested by users. All the books are e-books which are present with the system for a specific number of copies. Librarians can revoke any book from any user at any given time. The users can search books, request for e-books so that it can be issued for their usage, read issued books and return them back. The simple design and straight forward user interface makes it easier for the users to navigate around the pages.

Database Design:

Schema:

Considering the above entity model, schema diagram has been constructed,



Tech Stack:

- **Flask** – Flask is a backend framework for python. It includes flexible core functionality and an extensive ecosystem of supported modules like Flask-SQL Alchemy for database access, Flask-Login for session management, and Flask-RESTful for API development. Flask-cache for caching data that will be used frequently by the clients.
- **SQLite** – SQLite is a embedded database engine written in C. It's self-contained, i.e. it doesn't require a separate server process.
- **Vue js** – Vue.js is a progressive JavaScript framework for building user interfaces. It provides a reactive and component-based architecture that simplifies the development of interactive web applications. Vue.js includes a core library focused on the view layer, along with an ecosystem of tools and libraries such as Vue Router for navigation.

API Design:

- **GET** request – Return data from sqlite server or simple cache to the frontend client as json.
- **POST** request – For add and edit operations on the database. Used in adding users, books, sections, editing books, sections etc...
- **DELETE** request – For deleting operations performed on the database.

All APIs send appropriate messages and data as response from the backend with respective status codes.