IITM Modern Application Development - 1

Library Management System

Thanus Kumaar A - 22f3000726

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:

Entity Identification:

The following entities and their respective attributes were identified for the given problem statement,

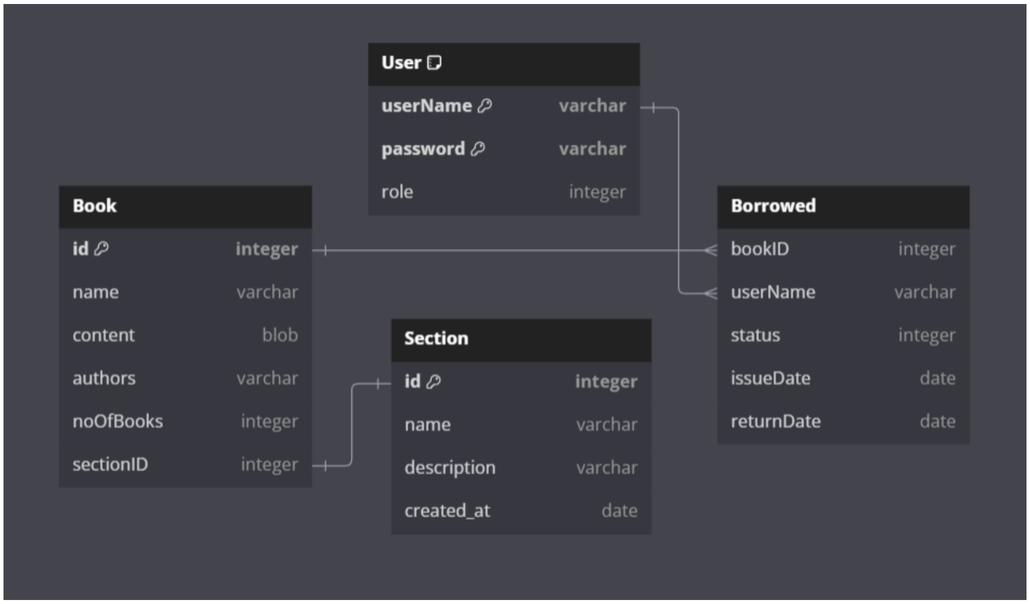
- Book – ID, Name, Author(s), Content, NoOfBooks
- Section – ID, Name, Description, CreationDate
- User – UserName, Password, Role

The following relationships were identified,

- Book is present in Section
- Users can borrow books
- Users can return books

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.
- **SQLite –** SQLite is a embedded database engine written in C. It's self-contained, i.e. it doesn't require a separate server process.
- **Jinja 2 –** Jinja is a fast, expressive, extensible templating engine. Special placeholders in the template allow writing code similar to Python syntax. Then the template is passed data to render the final document.

API Design:

- **GET** request – For rendering html pages and return data from sqlite server.
- **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.