ASHWIN N HEBBAR
Roll Number:
21f1003155

Email: 21f1003155@student.onlinedegree.iitm.ac.in

I'm a masters student hailing from Bengaluru, I am passionate about the union of full-stack development, Data Science and Machine Learning. In my free time, I usually ponder life's important questions, geopolitics and also critique music. I've poured a week's worth of hard work into this project and I hope you like it.

eLib Online Library Project Report

A place where knowledge cultivates.

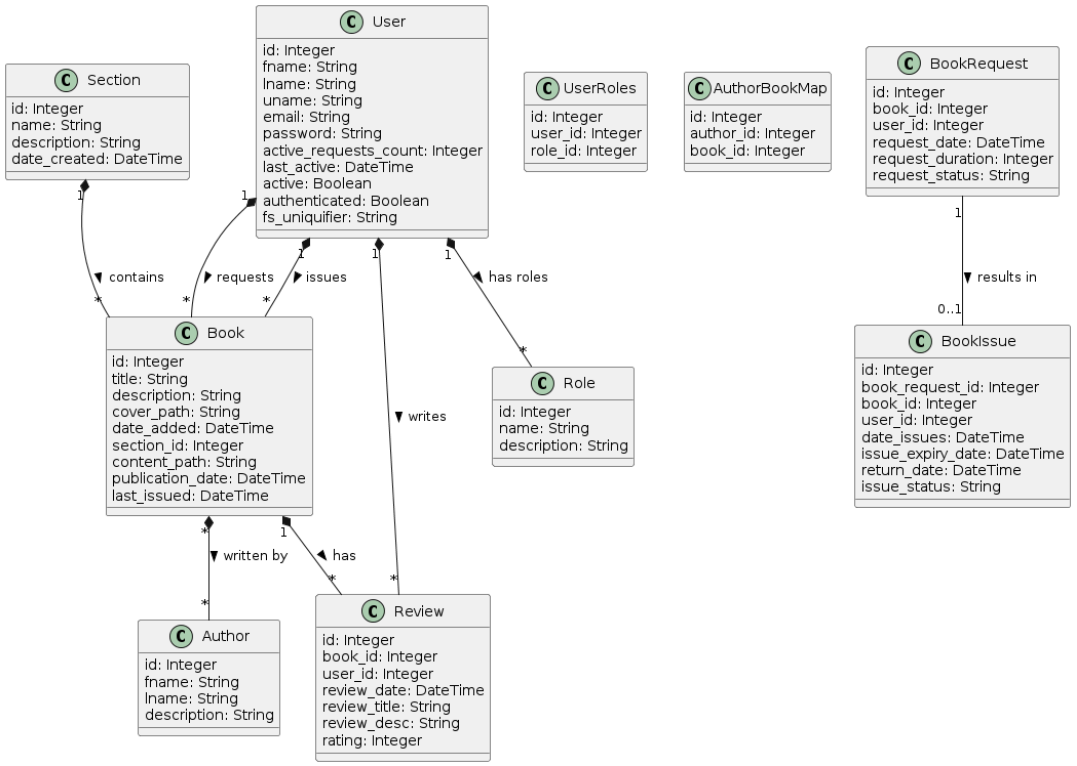
DESCRIPTION

eLib is an Online Library service and application. It enables its readers to access the world's knowledge through it's treasure trove of resources. It is also highly performant, with caching implemented. It has all the functionality and potential to take on giants in the same field.

TECHNOLOGIES USED

- Flask - Web Application Suite
- Flask SQLAlchemy - ORM for SqLite3 DB
- ApexChart - Admin Dashboard Graphs
- VueJS - Frontend
- Vue Router - routing
- Vuex - State Management
- Redis - Database for Celery and Caching of endpoints
- Mailhog - SMTP server

DB SCHEMA DESIGN



API DESIGN

The API is designed based on 5 objects from the database:

1. User
2. Book
3. BookRequest
4. BookIssue
5. Review

The endpoints are kept as meaningful as possible for convenience and intuitiveness.

ARCHITECTURE AND FEATURES

The Application is organized and well documented using comments to enable anybody to get up to speed on what's happening in the backend application. The frontend is also easily understood thanks to the extensive usage of components wherever possible. This also means it has a modular codebase. Which enables easy modification of code.

- Description: Files and Folders
  - Flle:**backend/app.py**
    - **The main application file for the backend**
  - Flle: **backend/appln/views.py**
    - Contains all the routes. Every route is well documented for easy traversal and understanding.
  - Directory: **appln**
    - Contains all the backend code. ./models.py contains the db models. ./worker.py contains the jobs handled by celery. Including periodic and triggered async jobs.
    - The python env is not included in the submission
  - Directory: **frontend**
    - Contains all the files required to run the frontend through vue.js 3. The node dependencies are not included in the submission.
    - Run "`npm run dev`" from this folder to start the frontend application.
  - The celery application needs to have redis installed and running. After that, run using `celery -A app:celery worker -B` To start the async triggered and periodic tasks.
- Features:
  - Full CRUD Functionality for User, Book and Book Sections.
  - Responsive UI using Bootstrap 5.3
  - Auto return book through celery tasks
  - Admin Dashboards.
  - Triggered Async jobs to download user and book data.
  - Fully Functional RBAC
  - Reminder and User Reports

VIDEO LINK:

https://drive.google.com/file/d/1qNN7a0qwgwxPTQ_-QOoZrAyVVO2LITTq/view?usp=drive_link