Library Management Systems (AppDev2- Project)

Name : Priyansh Vijay
Roll No. : 21f1004415
Student Email : 21f1004415@ds.study.iitm.ac.in

Description:

This app is a multi-user web app for borrowing and reading books.
To create this app, I followed the app wireframe and the guidelines provided by appdev2 instructors and the following steps:-

● Creating the database schema of the app and tables using flask-sqlalchemy.
● Creating the flask app instance and html pages using css and bootstrap
● Creating all the required routes to link the app to the database. A proper login framework with hashed passwords stored in the database.
● CSS styling to the web pages.
● At last i implemented celery jobs

Frameworks Used:

● Vue.js - The client side/ frontend part of the app is built on Vue.js.
● Flask - The server side/ backend part of the app is built on Flask.
● Redis and Celery are used for scheduled jobs/daily reminders via Google Chat and MailHog.
● Flask security for token based authentication.
● Smtplib and MIMEMultipart to send multipart messages using simple mail transfer protocol.
● Flask - this web application is built on flask.
● Jinja2 - for generating Monthly activity reports at backend
● Bootstrap - for templates of the web pages.
● SQLite3 - to create the database structure for the app.
● Flask-SQLAlchemy - to create and manage the relational database for the app.
● Matplotlib - to plot the app statistics graphs for the librarian dashboard.

Database:

● Database models for the app are created using flask-sqlalchemy.
● There are 5 Tables used in the database: User,Book,Section,Role,user_roles
● Book and Section have many to one relationship.
● User and Book have many to one relationship
● Users are differentiated based on their roles using the RolesUsers table.

System Design:

- This web app follows MVC architecture style:-
    - Model(M) is handled by flask. Flask interacts with the database and manages the data model.
    - View(V) is handled by vue.js. Vue components are responsible for interactive user interface.
    - Controller(C) is handled by flask. Flask routes handle all the business logic at the backend.
- Instance Folder - stores the database of the app.
- Static Folder - stores all the graphs and image files.
- Main.py - the code for the flask app instance, celery app instance and initializing database for the app.
- Sample_data.py - the code of some pre saved data of the app.
- Models.py - the code for creating database tables.
- Worker.py, Celeryconfig.py, Task.py - the code for celery configuration, scheduled jobs and daily reminders.
- Views.py - the code for all the routes and endpoints.

Features Implemented:

- Separate login form for users and librarian
- Proper alert messages for the tasks performed.
- Librarian dashboard with app statistics on users,books,section,book vs rating graph and Section vs Number of books in section graph.
- Librarian can manage books,section and give access to book and revoke the access from users.
- Librarian can create,update and delete books as well as sections
- Librarian can revoke access of books whose due date has passed by going to Particular route
- Librarian/User can search books based on their name/author.
- Users can request books, read content of approved books, like/dislike books,.
- Users can at maximum at any point of time can request 5 books
- Monthly Activity report of creator is sent to creators email on first day of month.
- Daily notifications on google chat to users to visit the app if inactive for 24 hours.

To run the app:

- Run the main.py file.

Presentation Video Link:

https://drive.google.com/file/d/1wqaHIyiUDApMjBuHhRO9oOuVMewnFYGV/view?usp=sharing