Library Management System Report

Name: Aman Garg
Roll Number: 21f1004958
Student Email: 21F1004958@ds.study.iitm.ac.in

Introduction

The library management system described in this report is implemented with a Flask backend,relying on Redis cache for speed, and celery workers for async processing. Vue.js is used for the front end interface. The system covers the functionality list in the department of user authentication, book & section management, functioning of issue tracking and feedback collection. Furthermore, it involves scheduling tasks for overdue books, log out notification, and monthly report generation. This paper seeks to analyze the concept, functionality, and practical implementation of the system, detailing its influence on library's activities and the growth of user orientation.

Main Objectives:

- Library Management System V2: Create the new and upgraded Library Management System (LMS) which is based on Flask and VueJS framework and have RBAC Access Control.
- Problem Definition: Describe the responsibilities of managing library resources, such as user access, built-in libraries, and reporting facilities, through the introduction of advanced functions and the streamlining of the process.
- Core Functionality: Implement the basic elements like authorization, section build and e-book management, search capability as well as schedule jobs for reminders as well as reports. These features give the framework a backbone, which guarantees its uninterrupted operation, and enables the library staff to successfully manage its resources.

Technologies Used:

- Flask SQL Alchemy: SQLAlchemy is a component of the Flask framework which is used for database management. This component has the ability to manage database operations and interactions in an efficient and effective way.
- Flask: Flask is the backend framework which is used to develop API endpoints and handle server-side logic. It offers a very lightweight and flexible solution to website development.
- Flask Security: The Flask-Security extension improves security in the Flask framework with the features like authentication, authorization, and password hashing which are responsible for the user management system functions.
- Vue CLI: Vue CLI is employed for frontend development, which provides us with the possibility of setting up an application in Vue.js - a dynamic and responsive JavaScript framework.
- Redis: The Redis caching within application is used for performance enhancement by keeping the most often accessed data in memory for the fast retrieval.

- MailHog: The primary purpose of MailHog is to act as a mail server within the application responsible for the implementation of functionalities related to emails and provide a foundation for debugging and testing email features during development.
- Cache: Data retrieval and processing are done by caching mechanisms which makes the application more responsive and faster.
- Celery: An instance of celery is used for task queue management and asynchronous job processing, which enables processing of the background tasks such as periodic jobs, email sending, and other operations as separate threads.

Implementation Details:

- Flask Setup: The Flask framework is initiated, along with essential setup like the database initialization and the API generation.
- API Endpoints: The Flask-RESTful API endpoints for user authentication, data section, book management, issue tracking, and other features are defined.
- Vue: Vue.js was used for frontend development combined with Vue Router for serving separate views and components through .vue files.
- Celery Tasks: Celery-based asynchronous tasks are defined for these tasks such as checking due dates of borrowed books, sending login reminders, and generating monthly reports.
- Scheduled Jobs: Celery utilizes a scheduling system to run tasks on a daily basis, such as date checking, login reminders and a monthly report which sends out on every first day of each month.
- Backend Operations: Database queries are performed to retrieve information that may involes total books, sections, users, and feedback for the purpose of generating monthly reports.
- Email Functionality: The monthly reports are sent to the admin email address by the mail service.
- Caching: Redis is incorporated for caching to optimize data retrieval and processing.

Database Schema

- The database schema consists of a number of tables that are aimed at enabling effective management of library resources. It contains the tables for user, authentication and authorization, book requests, sections, books, issued books, feedback and user activity tracking.
- User table holds user information such as email, username, password, and login details, while the Role table specifies user roles and permissions.
- The BookRequest table deals with the requests that have been made by the users for particular books, and the Section and Book tables sort and hold the data on the library sections and books separately.
- The Issue table includes issued books, issue dates, due dates, and statuses, and the Feedback table has user feedback on issued books.
- The UserActivity table, lastly, keeps track of user activities within the system for purposes of monitoring and analysis.

- This scheme is a solid basis for development of a powerful library management system with user authentication, resource organization, issue tracking, and user feedback functions.

Conclusion

The developed library management system is a modernized solution based on Flask, Vue.js, Redis, and Celery for effective backend processing, dynamic frontend interface, caching, and asynchronous task management. The system covers all users and resource management operations, issue tracking and scheduled tasks, and as a result, the system is quite effective in solving library management challenges. The database schema guarantees well-ordered storage of the data and user activity tracking that underpins a smooth and user-oriented library experience. This report reveals the architecture, features, and implementation details of the system, demonstrating its success in improving library operations and user satisfaction.