12/2/2024

Library Management System

BookHive

Group 13
EECS 3311

Contents

Library Management System: System Design Document

1. High-Level Architecture

The system is designed as a modular, multi-layered application comprising the following components:

- **Frontend**: HTML templates (base.html, home.html, etc.), CSS, and JavaScript files to handle user interaction.
- **Controllers**: Python modules (auth.py, models.py, views.py) to handle routing, requests, and business logic.
- **Database**: SQLite (database.db) for storing data related to users, admin, and book-related data and borrowing records.
- **Backend Framework:** Flask is used to develop the backend, managing routes, views, and server logic.
- **Email Service:** Flask-Mail is configured for sending automated emails, such as password reset links.

2. CRC Cards

2.1 User Authentication (auth.py)

<table>
<tr><td colspan="2">Class Name: auth.py</td></tr>
<tr><td colspan="2">Parent Class: None
Subclasses: None</td></tr>
<tr><td>Responsibilities:
Authenticate users through login.
Handle user registration, account creation, and password reset.
Manage user logout and session.</td><td>Collaborators:
‘models.py’: for user data validation.
‘views.py’: for routing authentication-related views.
Flask-Mail: Send password reset emails.</td></tr>
</table>

2.2 Models (models.py)

Class Name: models.py
Parent Class: None **Subclasses:** None

Responsibilities:	**Collaborators:**
• Manage and query the book inventory. • Store and retrieve user and book borrowing/returning data. • Update book statuses and availability.	• **'auth.py'** (to store user account details). • **'database.db'** (to handle any database related inquires).

2.3 Views (views.py)

Class Name: views.py	
Parent Class: None **Subclasses:** None	
Responsibilities:	**Collaborators:**
• Render templates and display pages to users. • Handle user requests for book management (add, edit, delete, borrow, return). • Route user actions, such as borrowing books, adding/editing/deleting books, and submitting queries. • Validate user permissions (e.g., admin-only access for specific actions). • Send automated emails for receipts and queries using Flask-Mail. • Handle filters for book lists based on author or genre.	• **'models.py'** ➢ Queries and updates the database for books, borrowed books, and users. ➢ Retrieves book and user-related data. (for user-specific routing). • **'auth.py':** ➢ Ensures the user is authenticated. ➢ Validates whether the user is an admin for restricted actions. • **'database.db':** ➢ Stores books, user data, and borrowed book details. ➢ Provides queried data such as book genres, authors, and borrow records.

2.4 App Initialization (__init__.py)

Class Name: __init__.py	
Parent Class: None **Subclasses:** None	
Responsibilities:	**Collaborators:** • **'auth.py':** Registers the authentication blueprint.

• Configure and initialize the Flask app. • Set up database connections (SQLAlchemy). • Configure Flask-Mail for email functionality. • Register blueprints for views and auth. • Create database tables if they don't exist. • Configure the login manager for user authentication.	• **'views.py'**: Registers the views blueprint. • **'database.db':** Initializes and creates database tables.

3. System Interaction with the Environment

- **Programming Language**: Python 3
- **Framework**: Flask
- **Database**: SQLite
- **Frontend Technologies**: HTML, CSS, JavaScript
- **Email Service:** Flask-Mail for sending emails.
- **Operating System**: Platform-independent (Windows, macOS, Linux)

Dependencies:

- **Flask** for backend logic.
- **SQLAlchemy** for database interaction.
- **Flask-Mail** for email functionality.
- **Flask-Login** for session management.

4. System Decomposition

1) User Sign up and Profile Management

- User can sign up with the following details:
- email address
- first name
- last name

- date of birth
- address

After that, they can login with their email address and password.

- Users can:
  - View available books.
  - Search, borrow, and view borrowed books and recommended books.
  - View their personal information
  - Update their address if needed
  - Access contact us form
  - View their messages about their books.

2) Admin Functionalities

- Admins can:
  - Search for books, add books, and view available books.
  - Manage user accounts by activating or deactivating them.
  - Track user activities.
  - See the trends and performance of users in the report.

3) Notifications and Messaging

Users receive messages regarding:

- Confirmation of book borrowing.
- Due dates and return reminders for borrowed books.

4) Account Activation/Deactivation

- Admins can:
  - Deactivate user accounts.
  - Reactivate user accounts.
- When a user's account is deactivated
  - They cannot log in to the system.
  - A flash message is displayed notifying them of the deactivation.

Upon reactivation, users can log in again without any data loss.

5) User Management:

- Handles user authentication, registration, and role management (admin or regular user).

- Includes password reset functionality.

6) Book Management:

- Tracks book inventory, borrowing status, and returns.
- Users can also get recommendations based on their history of borrowed books

7) Borrow Management:

- Links users with borrowed books and tracks due dates.
- The system sends emails and messages to the user about borrowed books and their due dates.

8) Error Handling:

- Validates input on both frontend (JavaScript) and backend (Flask).
- Provides clear error messages for invalid credentials, unregistered users, deactivated accounts or unavailable books.

5. Error Handling Strategy:

The system ensures robust error handling by validating inputs, managing user feedback, and handling exceptions gracefully.

Invalid Input:

- Email validation: display a flash message if the length of the email is less than four characters.
- First name validation: display a flash message if the length of the first name is less than two characters.
- Last name validation: display a flash message if the length of the last name is less than two characters.
- Address validation: display a flash message if the length of the address is less than two characters.
- Password validation: display a flash message if the password is less than seven characters.
- Display error messages for incorrect email or password formats when logging in.

Books errors:

- Display error messages for users if the user attempts to:
    - Borrow a book that is out of stock
    - Borrow a book that they have already borrowed.
    - Borrow more than 5 books.