Documentation: The Library Hub Application

Group 2

Table of Contents

LibraryHub Documentation

Overview

This document provides an in-depth explanation of the LibraryHub system, including its structure, key components, and interactions. It serves as a reference for developers and stakeholders to understand the system's functionality, implementation, and architecture. Project is built with Gradle Wrapper

---

System Architecture

The system comprises the following key components:

- Views: Responsible for the user interface and capturing user inputs, implemented using the Java Swing framework.
- Controllers: Handle user interactions, validate inputs, and bridge views with data models.
- Models: Represent the underlying data structure and logic, including database operations, implemented using Java. The backend interacts with a PostgreSQL database via PostgreSQL JDBC drivers.

Key Features

1. **User Authentication**
    - **Description:** Allows users to register and log in securely.
    - **Workflow:**
        - Users register by providing their first name, last name, username, and password.
        - Credentials are stored in the database securely.
        - LoginView validates credentials via the LoginController and redirects users upon successful authentication.
    - **Collaborators:**
        - LoginView, LoginController, UserService, UserRepository
2. **Book Search**
    - **Description:** Enables users to search for books by title.
    - **Workflow:**
        - Users input book titles in the search bar.
        - Application interacts with the DatabaseManager to fetch results.

        - Matching results are displayed in the interface.
    - **Collaborators:**
        - Application, DatabaseManager, BookRepository
3. **Inventory Management (Librarian/Admin POV)**
    - **Description:** Allows librarians to view and manage the library's book inventory.
    - **Workflow:**
        - Displays a list of all books, their availability status, current borrowers, and due dates.
        - Librarians can access these details via the Application interface.
    - **Collaborators:**
        - Application, DatabaseManager

Class Descriptions

Views

1. **LoginView**
    - Presents the login interface.
    - Redirects users to registration or the home page.
2. **RegisterView**
    - Collects user registration data and sends it to the controller for validation.
3. **Application**
    - Provides a dashboard for searching books, checkout etc.
4. ApplicationAdmin
    - Provides more functionality than application for admin, such as return books, show inventory etc.
5. Main
    - This is our main class which runs the program.

Controllers

1. **LoginController**
    - Handles user login and registration.
    - Validates credentials and redirects users appropriately.

2. **UserService**
    - Acts as an adapter to interact with the database for user-related queries.
3. DatabaseManager
    - Creates connection with database, and updates entries on database.
    - Uses preparedStatements to pass sql commands to update database.

Models

1. **UserRepository**
    - Manages user-related and admin-related database operations.
    - Validates credentials and stores user/admin data.
2. User
    - Handles user data and provides access to information for other classes.

Database Integration

1. **Tables:**
    - **User Table:** Stores user details (first name, last name, username, password, class).
    - **Book Table:** Stores book details (name, isbn_number, checked_out, due_date, checked_out_date, current_book_user).
2. **Operations:**
    - User registration: Adds user details to the User table.
    - Book search: Fetches book records matching user input.
    - Inventory management: Retrieves complete inventory details, including statuses and borrower information.
    - User login: two classes; admin and users, have different functions available to them
    - Return book: return checked out books after logging in as admin. Admins can return any book without needing the user who checked it out to login.
    - Add to cart: multiple books can be added to cart
    - Checkout : multiple books can be checked out after adding to the cart
    - Database integration: all information is saved in postgresql databases
    - Database updating: all information in database can be updated from the UI

Development Best Practices

- **Documentation:** Accompany all code with relevant comments to ensure smooth onboarding and knowledge sharing.
- **Testing:** Perform rigorous unit testing for each component to ensure reliability.
- **Optimization:** Optimize database queries to improve system performance, especially for book search and inventory management.
