****

1. **Introduction**

The Library Management System (LMS) is a computer program intended to automate a library’s operation, benefiting both the administrator and user alike. LMS enables effective book, member, and transaction management with user-friendly interface and stringent security controls. Implemented in Python with Tkinter (augmented by ttkbootstrap) and MySQL, the LMS offers role-based access, real-time book search, transaction management, fine calculation, and data export capabilities.

The document contains the objectives of the project, system design, implementation, and usage instructions, a comprehensive outline of the LMS.

1. **Project Objectives**

The main goals of the Library Management System are:

* **Library Operations:** Automated issuance and return of books and member management to minimize manual labor.
* **Role-Based Access:** Offer different interfaces for admins (complete control) and users (read-only access to search and list issued books).
* **Data Integrity and Security:** Adopt input validation, password hashing, and parameterized queries to provide sage and dependable data handling.
* **User-Friendly Interface:** Provide a contemporary, clean GUI with ttkbootstrap’s **‘”flatly”** theme for improved user experience.
* **Reporting Capabilities:** Support data export to CSV for admin reporting and analytics.
* **Fine Management:** Compute fines for late return at a customized rate ($1/day)

1. **Team Contribution**

The Library management system was developed through the collaborative efforts of three team members each contributing specialized skills to ensure a high quality deliverable.

* **Muhammad Usman:**
* Designed and implemented the MYSQL database schema, ensuring efficient data organization and relationships.
* Developed Backend Logic for book, member and transaction management, including optimized SQL queries.
* Configured database connectivity and ensure robust error handling for MYSQL operations.
* **Asma Channa**
* Designed and developed the graphical user interface using Tkinter and ttkbootstrap, focusing on modern and responsive design.
* Implemented a real-time search and filtering features for the dashboard interfaces.
* Ensured cross-platform compatibility and a consistent user experience across different screen sizes.
* **Soyam Kapoor**
* Implemented Secure user authentication using bcrypt for password hashing.
* Developed input validation mechanisms to ensure data integrity and prevent errors.
* Created the report generation and CSV export features, enabling administrators to analyze library data.

1. **System Design**

**4.1 System Architecture**

The LMS uses a modular, object-oriented design with explicit separation of concerns. The architecture consists of:

* **Frontend:** Tkinter using ttkbootstrap for a responsive and modern GUI.
* **Backend:** MySQL database for storing data persistently, with python scripts managing business logic.
* **Utilities:** Security modules (bcrypt for password encryption), input validation modules, and database connection modules.

**4.2 File Structure**

Project structure is maintained in a clear directory hierarchy to be easily maintained:

* main.py: Application entry point, creating the Tkinter GUI and login window.
* database/: Database config (db\_config.py) and setup script (db\_setup.sql)
* auth/: User authentication management (login.py, register.py)
* dashboard/: Admin (admin\_dashboard.py) and user (user\_dashboard.py) interface implementation
* modules/: Core functionality implementation (books.py, members.py, transaction.py)
* utils/: Security (security.py) and validators (validators.py) helper function
* assets/: Icon and other static assets
* requirements.txt: Dependency List
* README.md: Gives setup and usage directions

**4.3 Database Schema**

The MYSQL database has the following tables:

* Users: Saves user credentials and role (admin/user)
* Books: Save book information (ID, title, author, quantity, etc.)
* Members: Saves member data (ID, name, email, phone, etc.)
* Transactions: Saves record of book issue and return, including fine computation

Primary keys, foreign keys, and unique fields provide constraints to maintain data integrity.

**4.4 Key Features**

* **Admin Features:**
* Add, update, delete, and search members and books
* Issue and return books, with automatic fine generation for late return
* Export transaction and member information to CSV with pandas
* **Features for users:**
* Search books by title (partial matching)
* View issued books and their status
* **Security:**
* Password hashed with bcrypt
* Parameterized SQL queries to avoid injection attacks.
* **UI/UX:**
* Real-time search with Treeview widgets for tabular data
* Confirmation dialogs and error messages through messagebox
* Modern UI “flatly” theme from ttkbootstrap

1. **Implementation Details**

**5.1 Technologies Used**

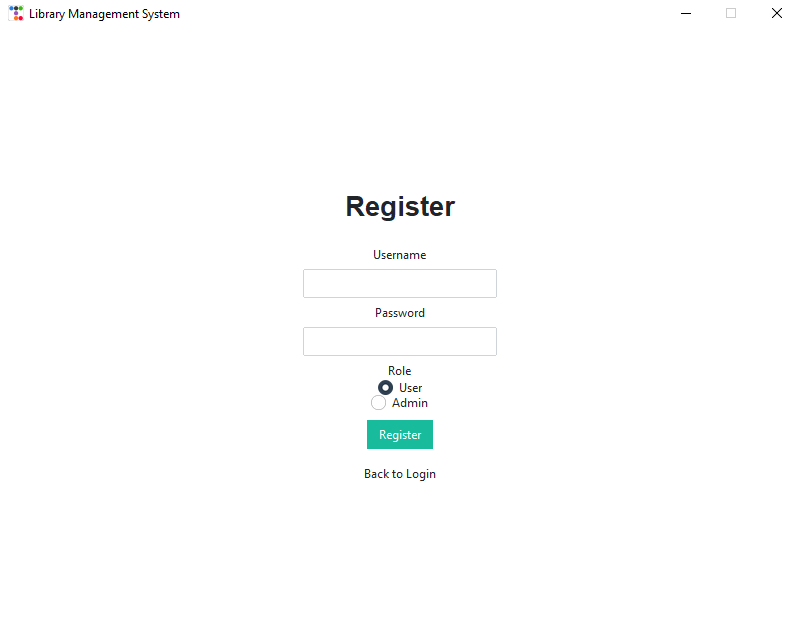
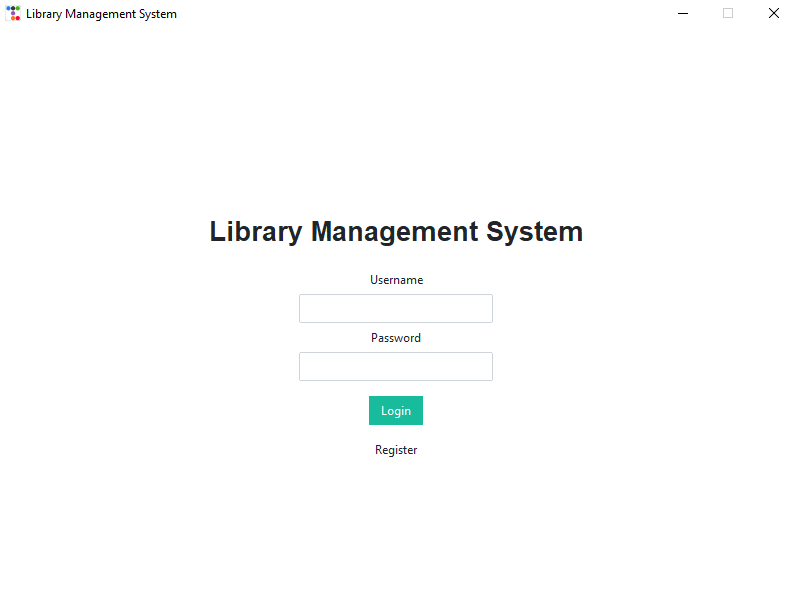
* **Python 3.8+:** Core programming language
* **Tkinter with ttkbootstrap:** For GUI Creation with trendy look
* **MySQL:** Relational database for storing data
* **bcrypt:** for secure password hashing
* **pandas:** For data export to CSV
* **mysql-connect-python:** For database connectivity

**5.2 Development Approach**

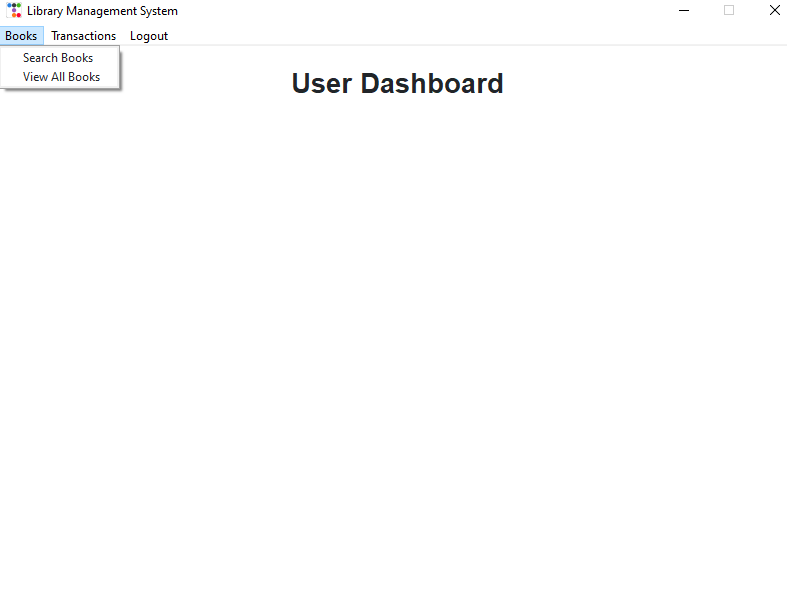
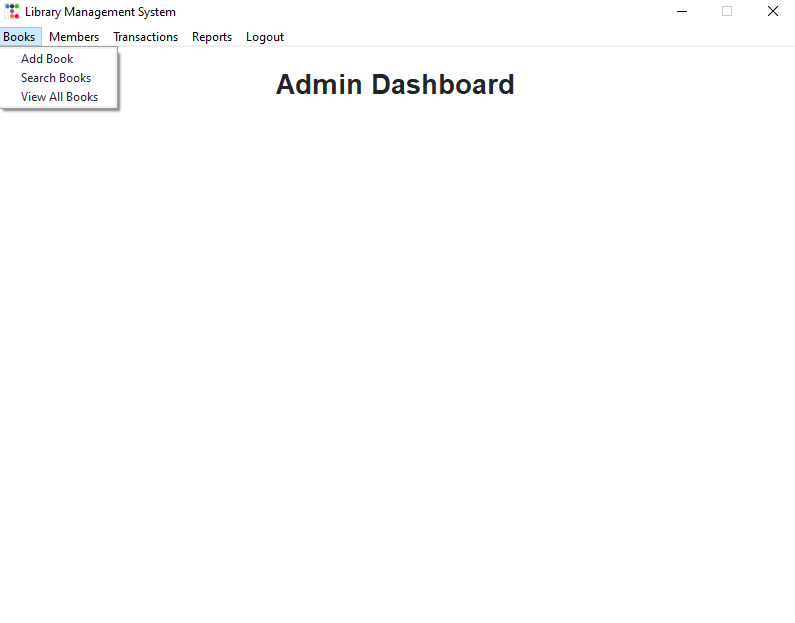
* **Modularity:** Every module (books, members, transactions) is contained in its own class for promoting reusability and maintainability
* **Error Handling:** Try-except blocks catch database errors, while validators provide proper input formats (e.g., email, phone numbers)
* **UI Design:** The GUI employs ttkbootstrap’s “flatly” theme to provide a clean and professional look. Treeview widgets are used to display information, and real-time search is implemented for better usability.
* **Security:** Passwords are hashed, and SQL queries are parameterized to avoid vulnerabilities.
  1. **Key Modules**
* **main.py:** Initializes the Tkinter application and launches the login window.
* **db\_config.py:** Connects to MySQL with error handling
* **security.py:** Provides password hashing and checking
* **validators.py:** Checks inputs (e.g., email format, numeric fields)
* **login.py/register.py:** user authentication and registration
* **admin\_dashboard.py:** Presents admin features such as book/members control and reporting
* **user\_dashboard.py:** Presents users with a simple interface for searching and displaying issued books.
* **books.py/members.py/transaction.py:** Controls main library activities with CRUD functionality and fine calculations.

1. **GUI Screenshots**

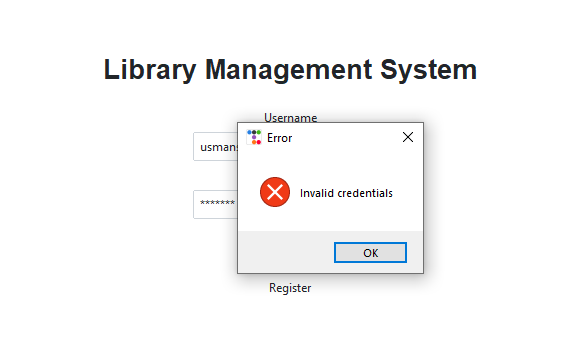
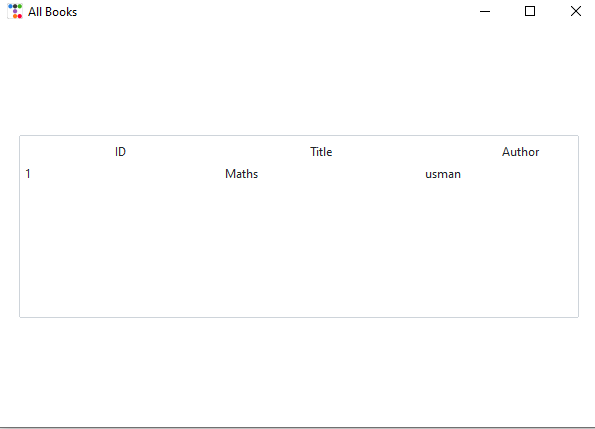
Here are Screenshots of graphical user interface of Project:



<*Fig. 01 Login Window*> <*Fig. 02 Registration Window*>



<*Fig. 03 Admin Dashboard*> <*Fig. 04 User Dashboard*>



<*Fig. 05 All Books in database*> <*Fig. 06 Error for Wrong Credentials*>

1. **Setup and Usage Instructions**

**6.1 Prerequisites**

* MySQL server installed (version 8.0 or higher).
* A working version of Python 3.8 or higher.
* Additional python packages as defined in requirements.txt.
  1. **Setup Steps**

**1. Install MySQL:** Install a MySQL server instance and a library\_db database.

**2. Execute Database Script:** Execute all scripts in database/db\_setup.sql to build tables and set up the database.

**3. Install other packages:**

**pip install -r requirements.txt**

**4. Adjust Configuration for Database:** Modify the database/db\_config.py file with your MySQL information (host, user, password, database).

**5. Start Application:**

**python main.py**

**6. Authentication:** Use pre-assigned credentials to sign in:

Username: admin

Password: admin123

Or create a new account.

**6.3 Usage**

* **Admins:**
  + Sign in using admin account.
  + Control book records (addition, updating, deletion, searching).
  + Control members (addition, updating, and deletion).
  + Execute book issuing/returns and analyze transaction history.
  + Report generation to CSV files.

**Regular users:**

* + Log in using users account.
  + Search books using their titles.
  + Check the status of their issued books and the return dates.

1. **Conclusion**

All requirements have been fulfilled and there are no issues concerning the implementation of functions. The provided solution is safe, simple, and functions well for library management. Its modular construction allows for easy maintenance, the GUI enhanced with ttkbootstrap is pleasant to work with, and the system is highly user friendly. Admins and regular users have improved system interaction through features such as live searches, automatic fine calculations, and exporting data into CSV files. Some new developments that could be made in the system include support for report files in other formats (e.g., PDF), advanced search filters, and the option to include barcode scanning functionality.

1. **Future Enhancements**
   1. **Advanced Reporting**: Integration of charts and other visual presentation formats to provide graphical matplotlib reports.
   2. **Email Notifications:** Automated reminders for overdue dates and accumulating due fines sent via emails.
   3. **Multilingual:** Integration of other languages into the system to support users from different regions.
   4. **Barcode Integration:** Integration of barcode scanning functionalities for easier book issuance and returning processes.
   5. **References**

**[1]** Rifaudeen, M.: ‘*Information management in libraries and its impact on sustainable library and information services with special reference to university libraries of Sri Lanka’, 2015*

**[2]** Rizzo, J.R.: ‘*Management for librarians’ (Greenwood Press, 1980. 1980)*

**[3]** Kumar, P., Kumar, R., Singh, R., & Singh, V. P. (2014). Library management system. *Kochi(unpublished dissertation submitted to the school of computer engineering, Cochin university of science & technology Kochi) p*, *7*.

**[4]** Araya, T. W., & Mengsteab, A. (2020). Designing Web-based Library Management System. *International Journal of Engineering Research & Technology*, *9*(10), 272-277.

**[5]** Abdulrazaq, M. B., & Mustafa, O. M. (2017). Designing and Implementing of An Online Library Managment System. *Science Journal of University of Zakho*, *5*(3), 278-284.

**[6]** Gibbons, S. (2005). Integration of libraries and course-management systems. *Library Technology Reports*, *41*(3), 12-20.

**[7]** Li, L. (2013, December). Designing and implementation of university library automatic management system based on the Internet of Things. In *Joint International Conference on Pervasive Computing and the Networked World* (pp. 241-247). Cham: Springer International Publishing.