LIBRARY MANAGEMENT SYSTEM

Guide Name: Preethi S

TEAM MEMBERS:

D.V.S.S SWAPNITH [AM.EN.U4AIE22016]
BHAVIKA GONDI [AM.EN.U4AIE22013]
SANTHOSH KUMAR [AM.EN.U4AIE22033]

Abstract:

Crafting the Library Management System was a journey to modernize and simplify traditional library operations. The aim was to address the challenges faced by conventional libraries in the fast-paced world and introduce a digital solution that enhances accessibility and efficiency.

The project recognizes the importance of transitioning from manual processes to automation. By embracing contemporary technologies, the system strives to offer a seamless experience for users, bridging the gap between the age-old practices of library management and the demands of the modern era.

Users engaging with this digital platform encounter features such as easy registration, secure logins, uncomplicated book searches, and straightforward borrowing and returning procedures. These functionalities are designed to enhance the user experience, making library interactions more convenient and user-friendly.

The Library Management System serves as a testament to the harmonious integration of technology with a timeless institution. It not only addresses the challenges faced by traditional libraries but also sets a standard for efficient and intuitive library management in the digital age. In the upcoming sections, we'll take a closer look at the details of how this system was created. This includes examining the specific design features, the different parts or sections (modules), and the methods used to turn this vision into reality.

Introduction:

In the realm of libraries, where the essence of knowledge converges with the yearning minds of readers, the Library Management System emerges as a beacon of transformation. With their physical catalogues and manual transactional processes, traditional libraries face challenges in keeping pace with the demands of the digital age. Recognizing the need for a more efficient, automated, and user-centric approach, the Library Management System project sets out to revolutionize how libraries function.

The traditional methods of library management, involving manual book tracking, user registrations, and transaction logs, have become increasingly impractical in a world characterized by speed and accessibility. Patrons of libraries, ranging from avid readers to students and researchers, require a system that not only preserves the sanctity of a library's purpose but also aligns seamlessly with contemporary technological advancements.

The introduction of the Library Management System seeks to address these challenges by ushering in a new era of library administration. At its core, the system aims to simplify and streamline the day-to-day operations of a library, offering a digital haven for both librarians and patrons. With this system, librarians can bid farewell to arduous manual bookkeeping and

administrative tasks, allowing them to focus more on curating enriching collections and providing an elevated library experience.

For library patrons, the introduction of the Library Management System promises a more dynamic and accessible engagement with the library's resources. Gone are the days of tedious book searches and manual record-keeping for borrowed and returned books. The system opens an avenue for users to explore the library's vast collection effortlessly, check their borrowing history, and conduct transactions securely and conveniently.

As we delve deeper into the intricacies of this project, the subsequent pages will unveil the various modules, design strategies, and the collaborative efforts that have gone into shaping the Library Management System. This introduction serves as a prologue to a transformative narrative, where technology and tradition harmoniously coexist, ensuring that the library remains a timeless sanctuary of knowledge in the digital age.

Software Requirements:

❖ System Requirements:

- MySQL Database Server (version 8.0 or higher)
- Web Browser to interact with the Streamlit web application.
- Python programming language installed
- An integrated development environment (IDE) such as Visual Studio Code (VS Code) or any other suitable alternative

❖ Libraries Used:

- **Streamlit**:
  Description: Streamlit was employed for creating the user interface of the Library Management System.

  Usage: Streamlit is used extensively throughout the project to design and structure the web-based user interface, providing a seamless experience for both administrators and library users.

- **MYSQL Connector:**
  Description: MySQL Connector is utilized for establishing a connection with the database, facilitating the storage and retrieval of data related to users, books, issued books, and more.

  Integration: The connector is integrated into various modules of the system, such as user registration, book management, and issuing and returning books, ensuring smooth interaction with the underlying database.

- **Datetime:**
  Description: The Datetime library is incorporated to handle date and time operations within the system.

  Integration: Datetime is employed in the issuance and return of books, ensuring accurate tracking of due dates and maintaining an organized borrowing history.

- **MySQL Database:**
  Description: MySQL serves as the relational database management system (RDBMS) for storing and retrieving structured data efficiently. It supports the organization of user details, book information, and transaction records.

  Integration: MySQL is the backend database for the Library Management System, storing critical information such as user credentials, book details, and transaction history.

Project Plan:

1. Authentication Module:

Description: Manages user authentication through a login page.

2. Circulation Management Module:

Description: Admin page for handling various aspects of the library circulation system.

Submodules:

1. **Manage Books** - Allows adding, updating, and deleting library books.
2. **Manage Students** - Manages student records and details.
3. **Request Management**- Handles book requests from users.
4. **Book Management:**
   a. Issue Book - Manages the process of issuing books to users.
   b. Return Book - Handles book return transactions.
   c. View Issued Books - Provides a list of currently issued books.

3. User Side Management Module:

Description: User page module managing interactions from the user side.

Submodules:

1. **Search Books** - Enables users to search for books in the library.
2. **Request Books**- Allows users to request books for borrowing.
3. **Borrowing History**- Displays the user's history of borrowed books.

Design Strategy:

1. Authentication Module - Login Page:

Design Overview:

The login page boasts a clean and user-friendly design, offering options for both admin and regular users to log in by entering their respective usernames and passwords. If individuals are not yet registered users, they have the option to sign up before logging in. This ensures a seamless and secure authentication process through an intuitive interface

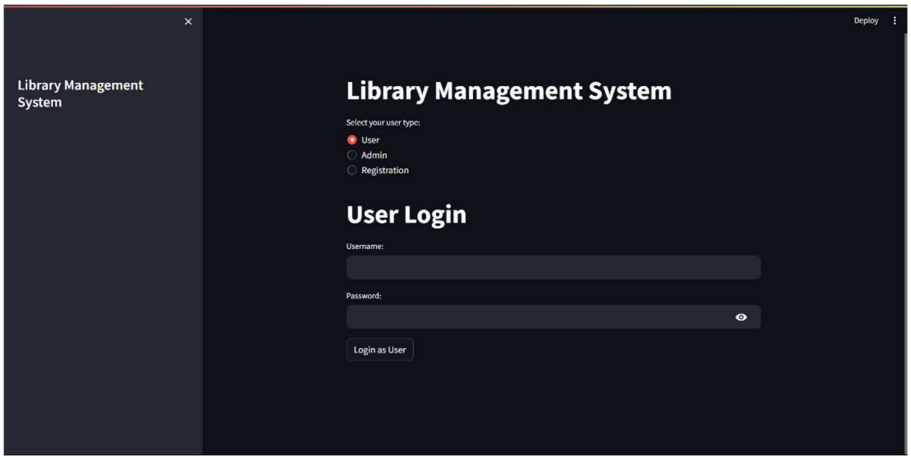


Figure 1: Login Page

2.Circulation Management Module - Admin Page:

Design Overview: The admin dashboard provides a centralized hub for overseeing library functions. It comprises distinct sections:

1. **Manage Books:** Manage books provides options to library manager to add book by adding book information like book title, author, ISBN and Genre. Also provides two

more options like view all books and delete a book by entering its title and ISBN number.

2. **Manage Students:** this submodule allows admin to get access of all the users that are registered
3. **Request Management:** admin can accept a book requested by user to borrow it or admin can reject the request.
4. **Book Management:**
    1. **Issue Book:** issue a book to a user manually then this option allows to do it.
    2. **Return Book:** when a student wants to return a book admin can take the book and update it by entering the username of user, book name which user wants to return and ISBN number of book.
    3. **View Issued Books:** Can view all the books that had been issued to which users as a table by admin

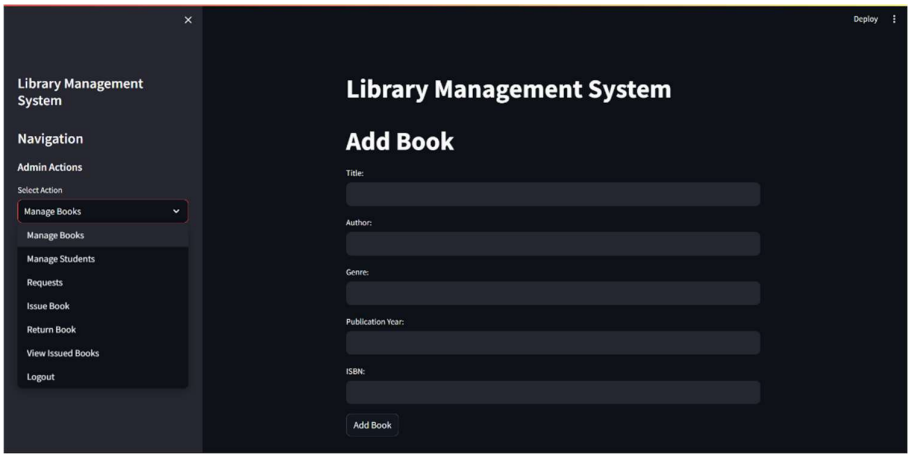


Figure 2: Admin page and admin actions

3.User Side Management Module - User Page:

Design Overview: The user page is designed to be intuitive for seamless user interaction with the library system.

1. Search Books: using this user can search for a book based on book name, author name, genre, ISBN and then select a book. The selected book can be requested to borrow by user by sending a request to admin. User can only request up to 3 books.

2. Request Books: the status of the requested books can be viewing here weather books are accepted or pending.

3. Borrowing History: Here user can view all the books he had borrowed from the past till now whole borrowing history.

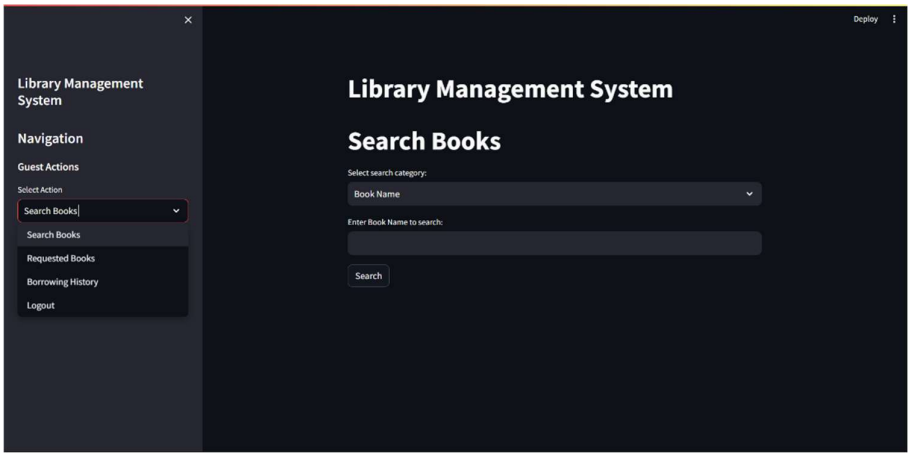


Figure 3: User page and user actions

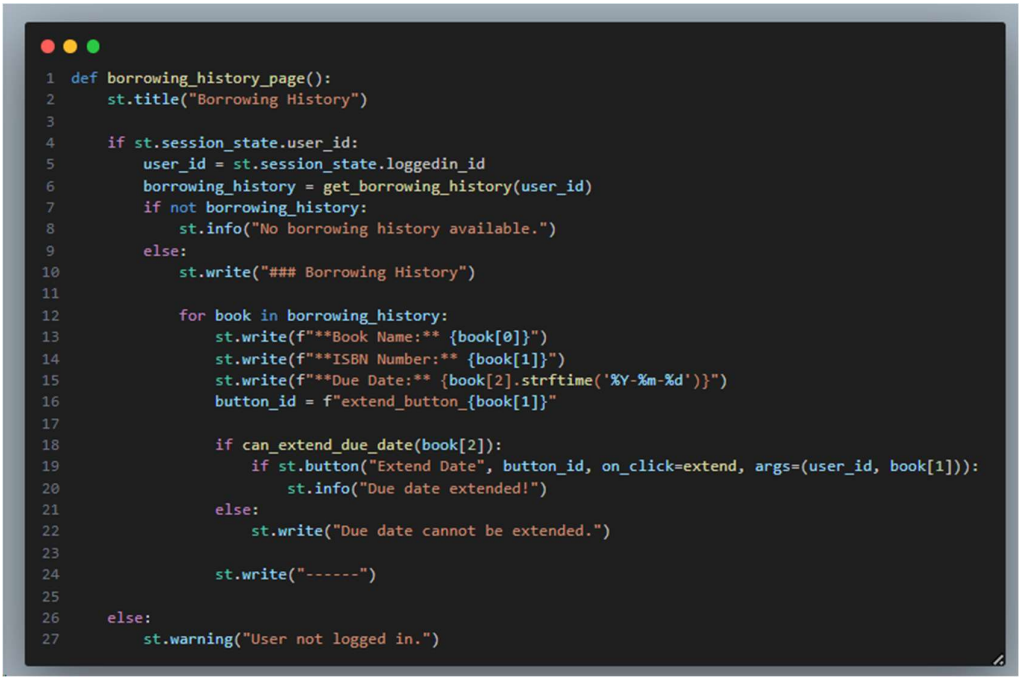
Important code snippets:

```python
def return_book_page():
    st.title("Return Book")
    username_return = st.text_input("Username:")
    book_name_return = st.text_input("Book Name:")
    isbn_number_return = st.text_input("ISBN Number:")

    if st.button("Return Book"):
        if username_return == "" or book_name_return == "" or isbn_number_return == "":
            st.warning("Username, book name, and ISBN number cannot be empty.")
        else:
            issued_books = get_issued_books()
            matching_books = [book for book in issued_books if
                              book[0] == book_name_return and book[1] == isbn_number_return and
                              book[3] == username_return]

            if matching_books:
                return_book(book_name_return,
                            isbn_number_return, username_return, book_id=matching_books[0][4])
                st.success(f"Book '{book_name_return}' returned successfully.")
            else:
                st.error("Book not found or not issued to the specified user.")
```

Figure 4: Code to return the book

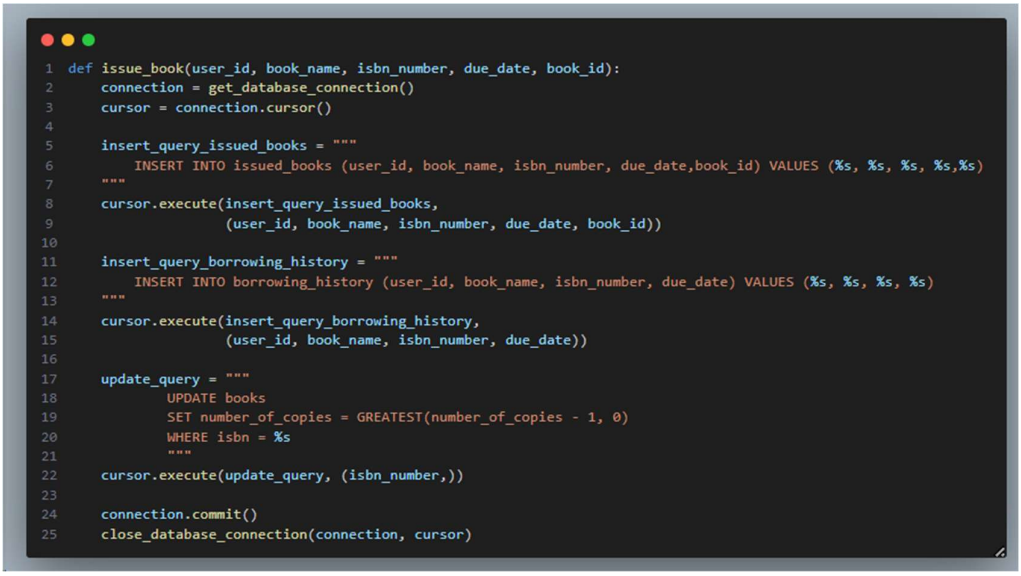
```python
def issue_book_page():
    st.title("Issue Book")
    student_username = st.text_input("Student Username:")
    book_name = st.text_input("Book Name:")
    isbn_number = st.text_input("ISBN Number:")

    min_date = datetime.now().date()
    max_date = min_date + timedelta(days=3650)
    due_date = st.date_input(
        "Due Date", min_value=min_date, max_value=max_date)

    if st.button("Issue Book"):
        if student_username == "" or book_name == "" or isbn_number == "":
            st.warning(
                "Student username, book name, and ISBN number cannot be empty.")
        else:
            student = get_user_details(student_username)
            if student and student[3] == 'guest':
                book_exists = check_book_existence(book_name, isbn_number)
                if book_exists:
                    issue_book(student[0], book_name, isbn_number,
                               due_date, book_id=book_exists[0])
                    st.success(
                        f"Book '{book_name}' issued to {student_username} with due date {due_date}.")
                else:
                    st.error(
                        "Book not found or not available for issuance. Please check the book name and ISBN number.")
            else:
                st.error(
                    f"User '{student_username}' not found or not a guest user.")
```

Figure 5: Issue Book page



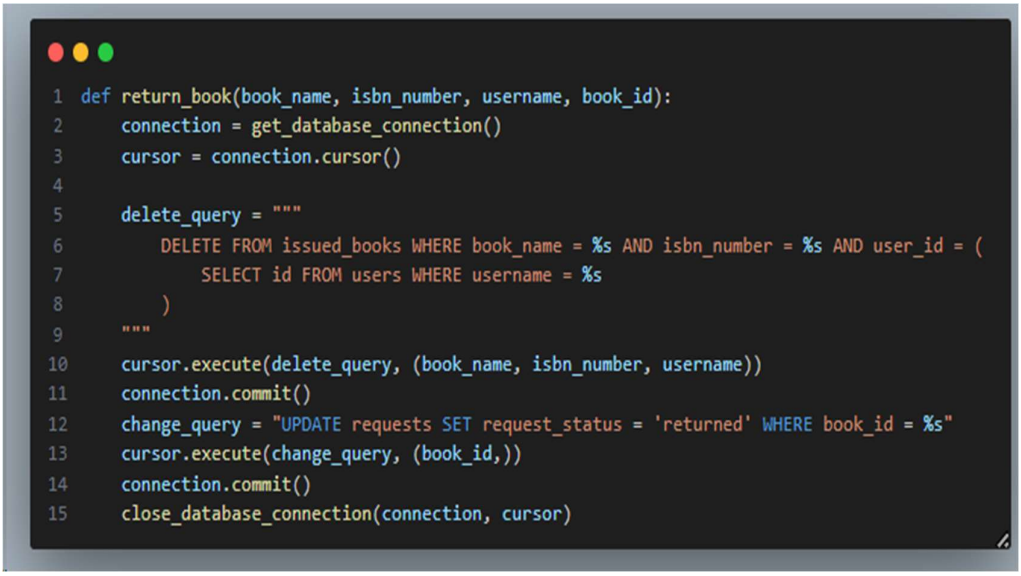
```python
def borrowing_history_page():
    st.title("Borrowing History")

    if st.session_state.user_id:
        user_id = st.session_state.loggedin_id
        borrowing_history = get_borrowing_history(user_id)
        if not borrowing_history:
            st.info("No borrowing history available.")
        else:
            st.write("### Borrowing History")

            for book in borrowing_history:
                st.write(f"**Book Name:** {book[0]}")
                st.write(f"**ISBN Number:** {book[1]}")
                st.write(f"**Due Date:** {book[2].strftime('%Y-%m-%d')}")
                button_id = f"extend_button_{book[1]}"

                if can_extend_due_date(book[2]):
                    if st.button("Extend Date", button_id, on_click=extend, args=(user_id, book[1])):
                        st.info("Due date extended!")
                else:
                    st.write("Due date cannot be extended.")

                st.write("------")

    else:
        st.warning("User not logged in.")
```

Figure 6: Code for the borrowing history



```python
def issue_book(user_id, book_name, isbn_number, due_date, book_id):
    connection = get_database_connection()
    cursor = connection.cursor()

    insert_query_issued_books = """
        INSERT INTO issued_books (user_id, book_name, isbn_number, due_date,book_id) VALUES (%s, %s, %s, %s,%s)
    """
    cursor.execute(insert_query_issued_books,
                   (user_id, book_name, isbn_number, due_date, book_id))

    insert_query_borrowing_history = """
        INSERT INTO borrowing_history (user_id, book_name, isbn_number, due_date) VALUES (%s, %s, %s, %s)
    """
    cursor.execute(insert_query_borrowing_history,
                   (user_id, book_name, isbn_number, due_date))

    update_query = """
            UPDATE books
            SET number_of_copies = GREATEST(number_of_copies - 1, 0)
            WHERE isbn = %s
            """
    cursor.execute(update_query, (isbn_number,))

    connection.commit()
    close_database_connection(connection, cursor)
```

Figure 7: Code for connection and executing queries in MYSQL for issue book



```python
def return_book(book_name, isbn_number, username, book_id):
    connection = get_database_connection()
    cursor = connection.cursor()

    delete_query = """
        DELETE FROM issued_books WHERE book_name = %s AND isbn_number = %s AND user_id = (
            SELECT id FROM users WHERE username = %s
        )
    """
    cursor.execute(delete_query, (book_name, isbn_number, username))
    connection.commit()
    change_query = "UPDATE requests SET request_status = 'returned' WHERE book_id = %s"
    cursor.execute(change_query, (book_id,))
    connection.commit()
    close_database_connection(connection, cursor)
```

Figure 8: Code for connection and executing queries in MYSQL for return book

Summary:

The Library Management System (LMS) project represents a comprehensive solution for effectively managing library operations. From user authentication to book management and library operations, the system's modular design ensures a seamless experience for both administrators and users. The project's success is attributed to its user-friendly interfaces, robust backend processes, and efficient data management.

The User Management module establishes a secure authentication process, while the Book Management module streamlines operations related to book addition, deletion, and searching. The Library Operations module orchestrates complex processes such as book issuance, returns, and request management, offering administrators a centralized control hub.

The design strategy prioritizes a clean and intuitive interface, ensuring accessibility and ease of use. Notable features include secure password hashing, due date tracking for book issuance, and streamlined processes for managing user requests and borrowing history. The project not only meets specified requirements but also anticipates future scalability through its modular architecture.

As a result, the Library Management System project stands as a successful implementation that significantly improves the efficiency of library management. It provides a solid foundation for further enhancements and serves as a testament to the effective integration of technology to streamline traditional library processes.