LIBRARY MANAGEMENT AUTOMATED SYSTEMS

MEETING SCHEDULE

- Meeting Days: Tuesday and Thursday

- Meeting Time: 11:00 AM

- Project Due Date: 1st May 2025

- Presentation Date:9th May 2025

TEAM MEMBERS AND ROLES

1. Femi-Olagundoye Oluwaferanmi – Project Manager

- Responsible for overseeing the project timeline, coordinating team efforts, and ensuring deliverables are met.

2. Fajolu Oluwadarsimi Esther – Backend Developer

- Handles server-side logic, database management, API integration, and security implementation.

3. Fajolu Oluwademilade – Product Designer

- Designs the user interface, creates wireframes, and ensures a seamless user experience.

4. Olonade Shayo – Frontend Developer

- Develops the client-side interface, ensures responsiveness, and integrates with the backend.

5. Oluwayimika David - Assistant Front End Developer

Support the lead frontend developer by implementing simpler UI components, handling styling tasks, assisting with testing and debugging.

6. Ebayn Kendic – Technical writer Create clear, accurate, and user-friendly documentation by translating complex technical information into accessible content that meets the needs of the target audience.

INTRODUCTION

The Library Management Automated System (LMAS) is a comprehensive web application designed to revolutionize how students and lecturers at the university access and manage library resources. This system aims to replace traditional, manual library processes with a digital, automated solution that enhances efficiency, security, and user experience.

The LMAS will feature user authentication, book borrowing and returning, a reminder system, and an admin dashboard for managing users and books. By addressing the functional and user requirements outlined in this document, the system will provide a robust, scalable, and user-friendly solution tailored to the needs of Babcock University’s library.

OBJECTIVES

The primary objectives of the Library Management Automated System are:

1. User Accessibility: Provide a secure and intuitive platform for students and lecturers to access library resources.

2. Security: Implement a two-step authentication process to ensure secure logins and protect user data.

3. Efficiency: Streamline book borrowing, returning, and cataloging processes.

4. Admin Control: Equip librarians with a comprehensive admin dashboard for managing users, books, and generating reports.

5. Scalability: Design a system that can accommodate future updates and additional features.

FUNCTIONAL REQUIREMENTS

The functional requirements are divided into key features to ensure the system meets its objectives:

1. **Authentication and User Management**

- The system shall allow users (students and lecturers) to register and log in using their unique credentials (matriculation number for students, staff ID for lecturers).

- The system shall hash and securely store user passwords using bcrypt or a similar library.

- The system shall allow admins to manage user roles (student, lecturer, admin) and permissions.

2. **Book Management**

- The admin shall allow admins to add, update, or remove books from the library catalog.

- The system shall allow users to search for books by title, author, or ISBN.

- The system shall provide filters for genre, availability, and publication year.

- The system shall display book details, including title, author, ISBN, publication year, and availability status.

3. **Borrowing and Returning Books**

- The system shall allow users to borrow books, subject to availability.

- The system shall allow users to return borrowed books.

- The system shall automatically update the availability status of books when borrowed or returned.

**Overdue Book Penalty System**

* **Physical Consequences**

After 3-day borrowing period expires:

* Automatic deduction of 20 penalty points/tickets from user's account
* User must physically meet with librarian to:
* Return overdue book
* Regain borrowing privileges
* Resolve any account restrictions
* **Online Consequences**

When book becomes overdue:

* Book is automatically deactivated/removed from user's digital collection
  + User cannot:
    - Access the book digitally.
* System displays clear warning: "Overdue, Return book to librarian to restore access"
* **Librarian Verification for Book Pickup:**
  + When a user borrows a book online, the system shall mark the book as "Reserved" until the librarian verifies that the book has been physically picked up.
  + The librarian shall have access to a verification interface where they can confirm the pickup by scanning the book's barcode or manually updating the status.
  + Once verified, the book's status shall change from "Reserved" to "Borrowed," and the due date shall be activated.
  + If the book is not picked up within a specified timeframe (e.g., 3 days), the reservation shall be automatically canceled, and the book shall be made available to other users.

5. **Admin Dashboard**

- The system shall provide an admin dashboard for managing users, books, and system settings.

- The system shall generate reports on book usage, overdue books, and user activity.

- The system shall provide analytics on popular books and user engagement.

6. **Security**

- The system shall use HTTPS for secure communication between the frontend and backend.

- The system shall implement CSRF protection and input validation to prevent attacks.

- Deactivate/reactivate accounts for non-compliance.

- The system shall restrict access to sensitive features (e.g., admin dashboard) based on user roles.

7. **Responsiveness and Accessibility**

- Dark/light theme toggle.

- The system shall be responsive and accessible on multiple devices (desktop, tablet, mobile).

- The system shall provide a user-friendly interface for both students and lecturers.

USER REQUIREMENTS

These are divided by user roles:

1. Students

- A simple and secure way to log in using their matriculation number and password.

- The ability to search for books by title, author, or ISBN.

- The ability to borrow and return books easily.

- Reminders for due dates and overdue books.

- A responsive and user-friendly interface that works on multiple devices.

* **Rating System Awareness:**
  + Students need to be aware of their borrowing rating and how it is affected by overdue books.
  + Students should receive notifications when their rating drops and guidance on how to improve it (e.g., returning overdue books).

2. Lecturers

- A simple and secure way to log in using their staff ID and password.

- The ability to search for books by title, author, or ISBN.

- The ability to borrow and return books easily.

- Reminders for due dates and overdue books.

- A responsive and user-friendly interface that works on multiple devices.

* **Rating System Awareness:**
  + Lecturers need to be aware of their borrowing rating and how it is affected by overdue books.
  + Lecturers should receive notifications when their rating drops and guidance on how to improve it.

3. Admins (Librarians)

- A secure way to log in and manage the system.

- The ability to add, update, or remove books from the catalog.

- The ability to manage user roles and permissions.

- The ability to generate reports on book usage, overdue books, and user activity.

- The ability to configure reminder settings (e.g., frequency, delivery method).

* **Book Pickup Verification:**
  + Admins need a verification interface to confirm that a reserved book has been physically picked up by the user.
  + Admins need the ability to cancel reservations if the book is not picked up within the specified timeframe.

4. General User Expectations

- The system should be fast, reliable, and easy to use.

- The system should provide clear error messages and guidance when something goes wrong.

- The system should protect user data and ensure privacy.

- The system should be accessible to users with disabilities (e.g., screen reader compatibility).

TOOLS FOR DEVELOPERS

1. Frontend Developer

- Languages: HTML, CSS, JavaScript.

- Styling: Bootstrap or Tailwind CSS (for styling and responsiveness)

2. Backend Developer

- Database: SQL (for database management)

- Languages: C# (for server-side logic)

- APIs: RESTful APIs (for communication between frontend and backend)

3. Product Designer

- Design Tools: Figma (for wireframing and prototyping), Adobe XD (optional)

- User Flow Diagrams: To map out user interactions and ensure a seamless experience.

ADDITIONAL FEATURES

1. Search and Filter Functionality

- Allow users to search for books by title, author, or ISBN.

- Implement filters for genre, availability, and publication year.

2. Admin Dashboard

- Allow librarians or admins to add, update, or remove books.

- Provide analytics on book usage, popular books, and user activity.

3. Security Measures

- Use HTTPS for secure communication.

- Hash passwords using bcrypt or similar libraries.

4. User Roles and Permissions

- Define different roles (e.g., student, lecturer, admin) with specific permissions.

5. Reporting and Analytics

- Generate reports on book usage, overdue books, and user activity.

**Overdue Penalty System**

* Implement a user rating system that decreases when books are returned late.
* Notify users of their updated rating and provide guidance on how to improve it.
* Restrict borrowing privileges for users with low ratings.

6. Librarian Verification for Book Pickup

- Add a "Reserved" status for books that have been borrowed online but not yet picked up.

-Provide librarians with a verification interface to confirm book pickup.

-Automatically cancel reservations if books are not picked up within a specified timeframe.

7. Auto-Deactivation

- Overdue books disappear from user accounts until returned.

8. Admin Override:

- Manual account reactivation after penalty resolution.

**TECHNICAL IMPLEMENTATION DETAILS**

**Backend Developer**

* **Overdue Penalty System:**
  + Implement a user rating field in the database.
  + Create scheduled task to check for overdue books daily and update user ratings accordingly.
* **Librarian Verification for Book Pickup:**
  + Develop an API endpoint for librarians to update the book status to "Borrowed" after verification.

**Frontend Developer**

* **Overdue Penalty System:**
  + Display the number of given tickets(example 20) on their profile page.
  + Show notifications when the rating drops.

**Product Designer**

* **Overdue Penalty System:**
  + Design a user-friendly rating display and notification system.
* **Librarian Verification for Book Pickup:**
  + Create wireframes for the librarian verification interface, ensuring it is intuitive and efficient.