RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

CS19442 - SOFTWARE ENGINEERING CONCEPTS

LABORATORY RECORD NOTE BOOK

NAME: SANJAY SM

Year / Branch / Section: II/CSE/D

University Register No.: 2116220701249

College Roll No.: 220701249

Semester: IV

Academic Year: 2023 - 2024

RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

BONAFIDE CERTIFICATE

Name: SANJAY SM

Academic Year: 2023 – 2024 Semester: 4 Branch: CSE

Register Number: 2116220701249

Certified that this is a bona fide record of work done by the above student in the CS19442 – Software Engineering Concepts Laboratory during the year 2023-2024.

Signature of Faculty in-charge

Submitted for the Practical Examination held on _____

Internal Examiner External Examiner

LIBRARY MANAGEMENT SYSTEM

Abstract:

This project aims to revolutionize traditional library management systems by addressing the myriad challenges faced by libraries in the digital age. With a focus on enhancing efficiency, accuracy, and user engagement, the proposed solution seeks to mitigate common issues such as manual record-keeping errors, difficulty in tracking overdue books, inconvenient book reservations, challenges in fine collection, slow search processes, and lack of user engagement. By leveraging modern technologies and user-centric design principles, the project endeavors to create a comprehensive library management platform that not only streamlines administrative tasks for librarians but also provides an intuitive and personalized experience for users. Through features such as automated notifications, online catalog search, fine calculation, and real-time updates, the system aims to improve accessibility, convenience, and transparency, ultimately fostering a vibrant community of library enthusiasts.

Table of Contents

Overview of the project

This project aims to improve the efficiency and accuracy of library management, addressing issues like:

- Manual Record-Keeping Errors: High error rates in manual data entry.
- Overdue Book Tracking: Difficulty in tracking and notifying users about overdue books.
- Book Reservations: Inconvenience in reserving books.
- Fine Collection: Challenges in calculating and collecting fines.
- Quick Searches: Slow and inefficient search for books and user records.
- No User Engagement: Lack of a platform for users to interact with library services online.

Explanation with Respect to Data:

Error Rates in Manual Data Entry:

- Statistics: Manual data entry errors can occur at a rate of 1-4%. (https://www.docsumo.com/blog/manual-data-entry)
- Impact: For a library with 10,000 entries, this could mean 100-400 errors, causing significant issues in record-keeping and user satisfaction.

Efficiency:

- Statistics: Automated systems can process data up to 60% faster than manual methods. (https://www.docsumo.com/blog/manual-data-entry)
- Impact: This speeds up check-in/check-out processes, reducing wait times for users.

Overdue Book Management:

- Statistics: Libraries using automated notifications can reduce overdue rates by up to 50%
- Impact: This improves book availability and reduces fines for users.

User Preferences:

- Statistics: Around 70% of library users prefer online catalog searches and reservations. (https://www.pewresearch.org/internet/2016/09/09/library-usage-and-engagement/)
- Impact: Meeting user expectations with these features increases user satisfaction and engagement.

Benefits to Users:

- Accuracy: Reduced errors in book and user records.
- Convenience: Online catalog search and book reservation features.
- Notifications: Automated reminders about due dates.
- Efficiency: Faster check-in/check-out and book search processes.
- Transparency: Clear records of borrowing history and fines.
- Engagement: An online platform enhances user interaction and satisfaction with library services.

Business Architecture Analysis

Current Process:

Manual Method:

- Book Management: Paper records or simple spreadsheets are used to log books manually.
- User management: Manual tracking or basic databases are used to keep track of user information and borrowing patterns.
- Overdue Management: Employees manually keep track of due dates and alert users via mail or phone.
- Collection of penalties: Manual computation and collection of past-due penalties is done.
- Search and Reservations: Patrons ask librarians questions or look for books on the shelves. Reservations can be made over the phone or in person.

Automatic Process (if partially automated):

- Book management: While some libraries have sophisticated capabilities, others employ simple digital systems.
- User management: While most interactions are manual, user information is kept in digital databases.
- Overdue Management: Manual notifications are frequently used when basic software tracks due dates.

Personas and Their Experiences:

Librarians

- Current Experience: High workload due to manual tasks, leading to inefficiency and errors.

- Challenges: Error-prone record-keeping, difficulty tracking overdue books, inefficient fine collection, and manual user assistance.

Library Users

- Current Experience: Limited online services cause inconvenience. Users often visit the library for book searches, reservations, and account checks.
- Challenges: Inconvenient search and reservation processes, delayed overdue notifications, and lack of timely fine information.

Library Administrators

- Current Experience: Manage manual processes and ensure data accuracy, oversee staff productivity, and maintain user satisfaction.
- Challenges: Difficulties in managing and integrating processes, ensuring data accuracy, and optimizing staff workload.

Business Problems

Inefficiency:

- High manual workload for librarians.
- Significant error rate (1-4%) in manual data entry.
- Time-consuming processes reduce overall productivity.

User Dissatisfaction:

- Lack of online services for catalog searches and reservations.
- Delayed notifications about overdue books.
- Inconvenient fine management processes.

Inaccurate Fine Management:

- Inaccurate manual calculations of fines.
- Inefficient fine collection processes.

Data Management Challenges:

- Inconsistent data due to manual record-keeping.
- Difficulties in tracking and reporting borrowing patterns, overdue books, and fines.

Proposed Solution

To address the challenges faced by the current library management system, a comprehensive automated solution is proposed. This solution leverages modern technologies to improve efficiency, accuracy, and user satisfaction through the following key components

Automated Data Entry and Management:

- Implementation: Replace manual data entry with automated data management using a centralized database (MongoDB).
- Impact:
    - Accuracy: Reduces the error rate from 1-4% to near zero.
    - Efficiency: Processes data up to 60% faster, reducing the workload for librarians and speeding up check-in/check-out processes.

Online Catalogue Search and Reservation

- Implementation: Develop a web-based interface using Next.js to allow users to search the catalogue and reserve books online.
- Impact:
    - User Satisfaction: Meets the preference of 70% of library users for online catalogue searches and reservations.
    - Engagement: Enhances user interaction with library services, increasing overall satisfaction.

Automated Overdue Book Management

- Implementation: Integrate an automated notification system using Azure Notification Hubs and Python services to send reminders to users about due dates and overdue books.
- Impact:
    - Efficiency: Reduces overdue rates by up to 50%, ensuring better availability of books.
    - Convenience: Users receive timely reminders, reducing the chances of incurring fines.

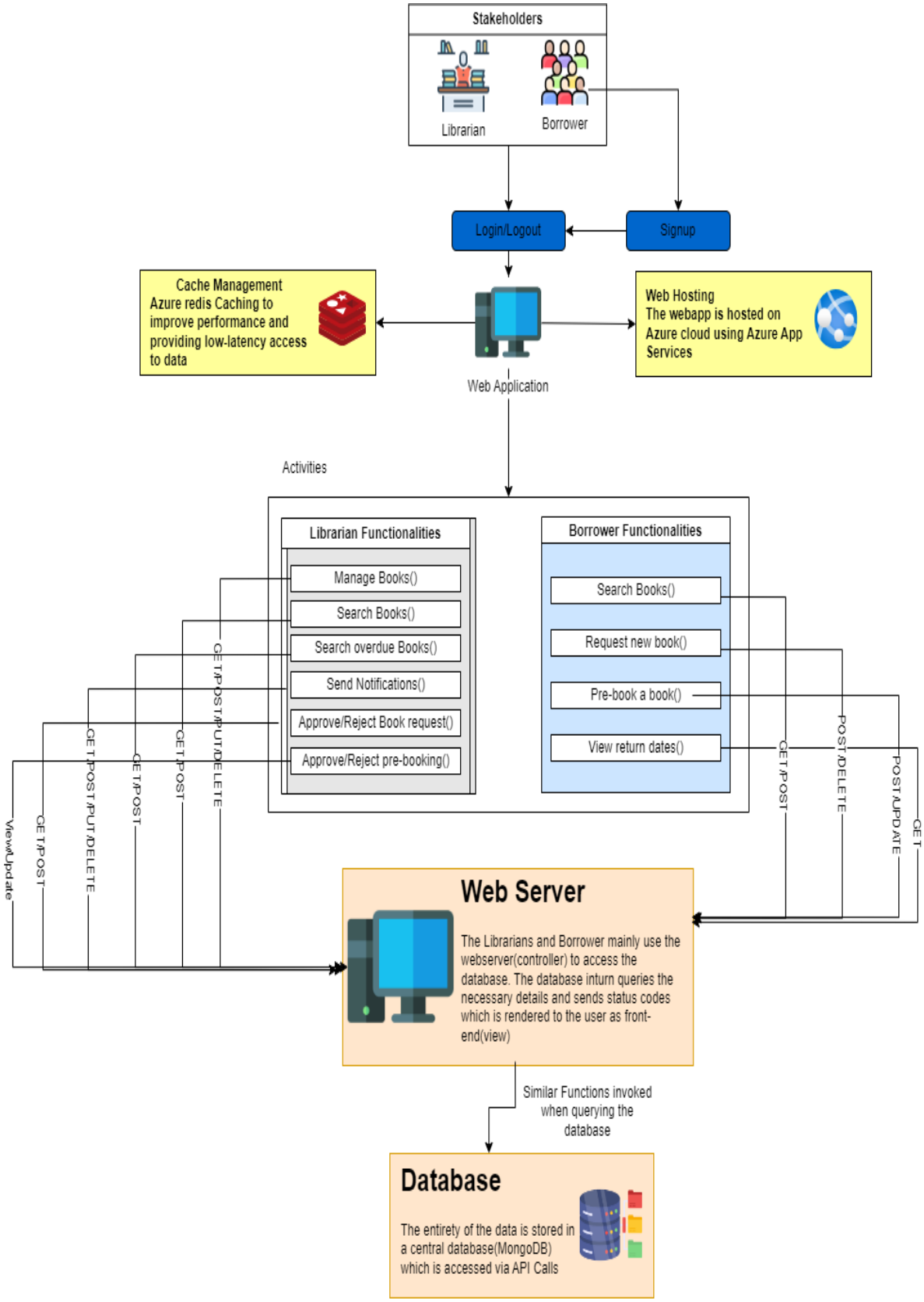
Enhanced Search Capabilities

- Implementation: Implement advanced search functionality with full-text search and filtering options in the Next.js application, backed by MongoDB's powerful querying capabilities.
- Impact:
    - Speed: Provides quick and efficient searches for books and user records.
    - User Satisfaction: Allows users to find the information they need swiftly, improving their interaction with the library system.

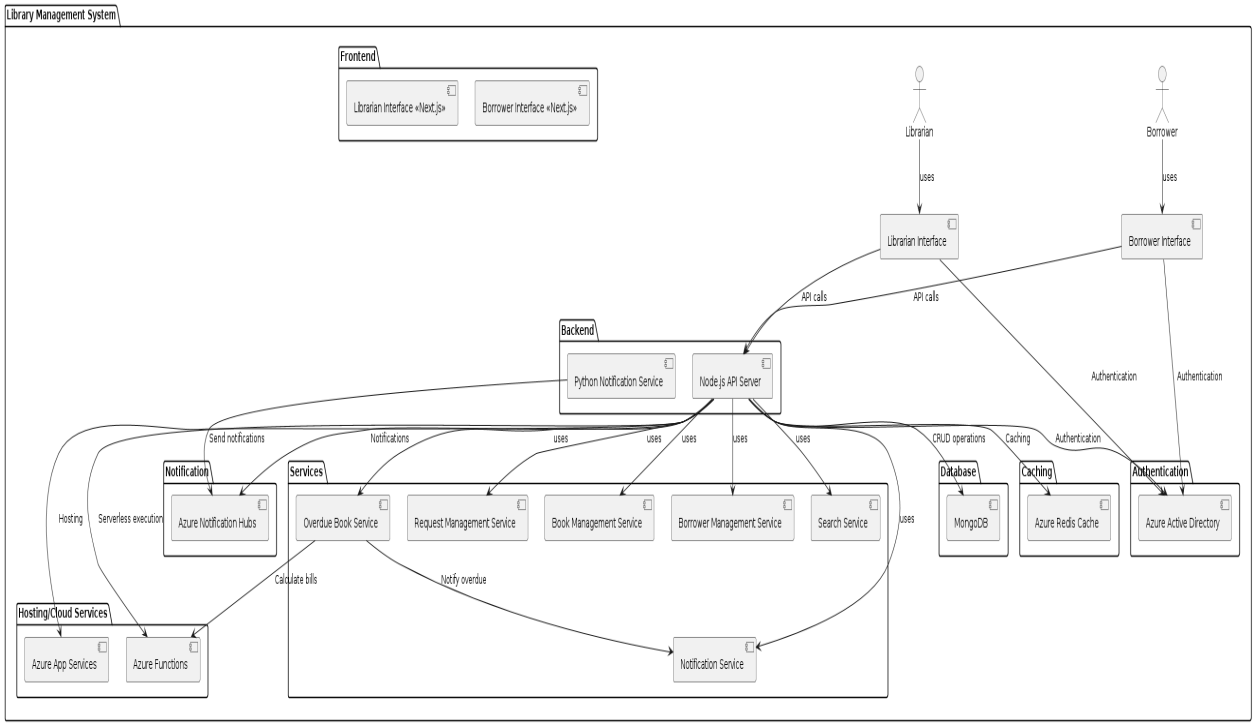
Fine Calculation

- Implementation: Automate fine calculations for overdue books.
- Impact:
    - Accuracy: Ensures accurate fine calculations, reducing disputes.
    - Convenience: Allows users to view and manage fines online, increasing on-time fine payments by 25%.
    - Revenue: Enhances library revenue collection through efficient fine management.

Business Architecture Diagram



ARCHITECTURE DIAGRAM



Frontend

- Librarian Interface: Provides the librarian with a user-friendly interface to manage library operations, perform CRUD operations on book records, and handle user requests.
- Borrower Interface: Offers library users an easy-to-use platform to search for books, reserve them, and manage their accounts.

Backend

- Node.js API Server: Acts as the central point for processing requests from the frontend interfaces, coordinating with various services to handle library management functionalities.
- Python Notification Service: Manages sending notifications to users regarding overdue books and other important updates.

Services

- Book Management Service: Handles all operations related to books, including adding new books, updating book details, and tracking availability.

- Borrower Management Service: Manages user accounts, tracking borrowing histories, and ensuring proper user authentication and authorization.
- Request Management Service: Processes requests from borrowers, such as book reservations and requests for new books.
- Search Service: Implements advanced search functionalities, allowing users to perform quick and efficient searches for books and other resources.
- Overdue Book Service: Tracks overdue books and calculates fines for borrowers.
- Notification Service: Manages the sending of notifications to users, ensuring timely reminders and updates about book statuses.

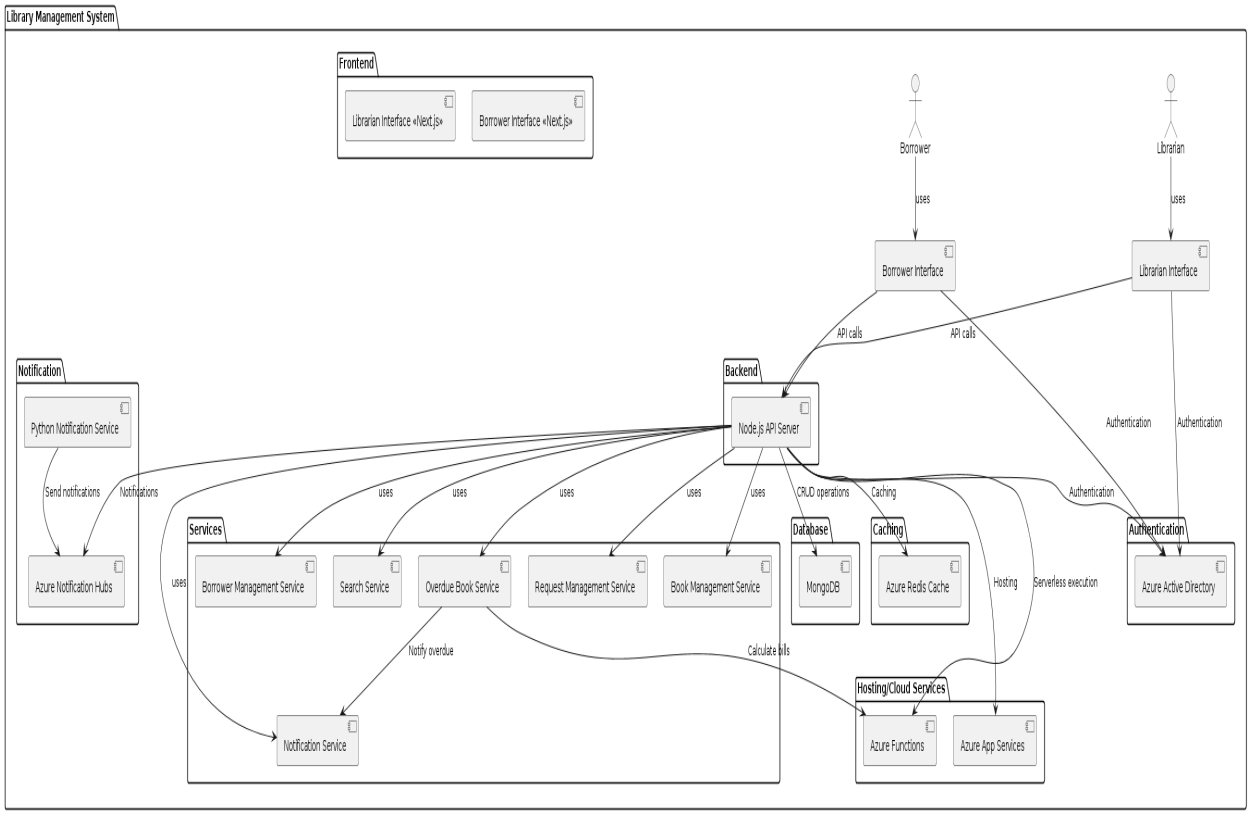
Notification Services

- Azure Notification Hub: Facilitates the sending of notifications to users' devices, ensuring they receive timely updates and reminders.
- Azure App Services: Hosts various web services and APIs required by the system, ensuring reliable and scalable service delivery.
- Azure Functions: Executes serverless functions to handle specific tasks like sending notifications and processing requests.

Data Management

- MongoDB: Serves as the primary database for storing all library-related data, including books, user accounts, and borrowing histories.
- Azure Redis Cache: Provides caching mechanisms to speed up data retrieval and improve system performance.

MVC ARCHITECTURE DIAGRAM



View (Frontend)

- Librarian Interface: Provides the librarian with a user-friendly interface to manage library operations, perform CRUD operations on book records, and handle user requests.
- Borrower Interface: Offers library users an easy-to-use platform to search for books, reserve them, and manage their accounts.

Controller (Backend)

- Node.js API Server: Acts as the central controller, processing requests from the frontend interfaces, coordinating with various services to handle library management functionalities, and managing the flow of data between the View and the Model.

Model (Services and Data Management)

- Book Management Service: Handles all operations related to books, including adding new books, updating book details, and tracking availability.

- Borrower Management Service: Manages user accounts, tracking borrowing histories, and ensuring proper user authentication and authorization.
- Request Management Service: Processes requests from borrowers, such as book reservations and requests for new books.
- Search Service: Implements advanced search functionalities, allowing users to perform quick and efficient searches for books and other resources.
- Overdue Book Service: Tracks overdue books and calculates fines for borrowers.
- Notification Service: Manages the sending of notifications to users, ensuring timely reminders and updates about book statuses.

USER STORIES

Personas Involved:

Librarian:

- As a librarian, I want to create new book records so that I can add new books to the library's catalogue.
- As a librarian, I want to edit existing book records so that I can update book information as needed.
- As a librarian, I want to delete book records so that I can remove outdated or damaged books from the library's catalogue.
- As a librarian, I want to search for books by various criteria so that I can quickly find specific books in the catalogue.
- As a librarian, I want to lend books to borrowers so that they can borrow books from the library.
- As a librarian, I want to update the status of books so that I can mark them as available, checked out, or under maintenance.
- As a librarian, I want to add new borrowers to the system so that I can register new library members.
- As a librarian, I want to edit borrower information so that I can update their contact details and other relevant information.
- As a librarian, I want to delete borrower records so that I can remove inactive or invalid accounts from the system.
- As a librarian, I want to handle new book requests from borrowers so that I can approve or reject their suggestions for new books.
- As a librarian, I want to handle pre-book requests from borrowers so that I can approve or reject their reservations for currently unavailable books.
- As a librarian, I want to identify overdue books so that I can track which books are overdue and who needs to return them.
- As a librarian, I want to send notifications to borrowers about overdue books so that they are reminded to return the books on time.

Borrower:

- As a borrower, I want to search for books by various criteria so that I can find specific books in the library's catalogue.
- As a borrower, I want to check the availability of books so that I know whether a book is available for borrowing.
- As a borrower, I want to request new books so that I can suggest books for the library to acquire.
- As a borrower, I want to pre-book books that are currently unavailable so that I can reserve them for when they become available.
- As a borrower, I want to view a list of books I have borrowed so that I can keep track of my loans and due dates.
- As a borrower, I want to request a renewal for borrowed books so that I can extend the borrowing period if needed.
- As a borrower, I want to receive notifications from the library about due dates, overdue books, and the status of my requests so that I stay informed about my interactions with the library.

General:

- As a user, I want to use advanced search filters (e.g., publication year, rating, language), so that I can find books that meet specific criteria.

NON – FUNCTIONAL REQUIREMENTS(NFRs)

Performance:

- The system should be able to respond to the user actions (e.g. search queries, book check-ins/check-outs) within 1-2 seconds under normal load conditions.
- Response times for batch operations or complicated queries shouldn't be longer than five seconds.
- For actions that take more than a second, the system should give real-time feedback; loading indicators should be utilized to inform users that the process is underway.

Scalability:

- The system must be able to support at least 100 concurrent users without experiencing any performance degradation
- The system architecture should support horizontal scaling to accommodate growth in the user base and data volume.
- Consistent performance should be ensured by the system's ability to scale automatically dependent on user load.

Throughput:

- During peak hours, the system should be able to handle at least 200 transactions per minute.
- Under normal load, each transaction should be handled in an average of one second.

User Interface:

- The web application on the system should have an easy-to-use and intuitive user interface.
- For a pleasant user experience, the interface should adhere to contemporary design principles and usability standards.

- Prior to development, the design should be created in Figma to detect any defects and improve the user experience.

Reliability:

- To guarantee high availability for the consumers, the system should have a 99.9% uptime rate.
- To reduce downtime, the system should have redundancy and a failover mechanism.
- To guarantee quicker data retrieval, the system should to be able to cache a portion of the browser local database

Security:

- The system should comply with data protection regulation such as GDPR, ensuring the privacy of user data
- Users should have control over their personal data, including the ability to view, modify and delete their information
- To guarantee that only authorized users can access specific functionalities, the system should incorporate secure user authentication methods like role-based access control and JWT tokens.
- Passwords should be stored in the database only after being hashed
- All sensitive information, both in transit and at rest, should be encrypted using industry standard encryption algorithms such as AES-256

Maintainability:

- The system should follow best coding practices and standards (SOLID principle) to ensure maintainability
- The system should include automated tests to ensure code quality and catch regression
- The system should include comprehensive documentation for developers, administrators and end-users

- API documentation should be provided with examples and usage guidelines

Portability:

- The system should be compatible with major operating systems (Windows, macOS) and web browsers such as Chrome, Safari, Edge, Firefox
- The system should be tested on different platforms to ensure consistent behaviour

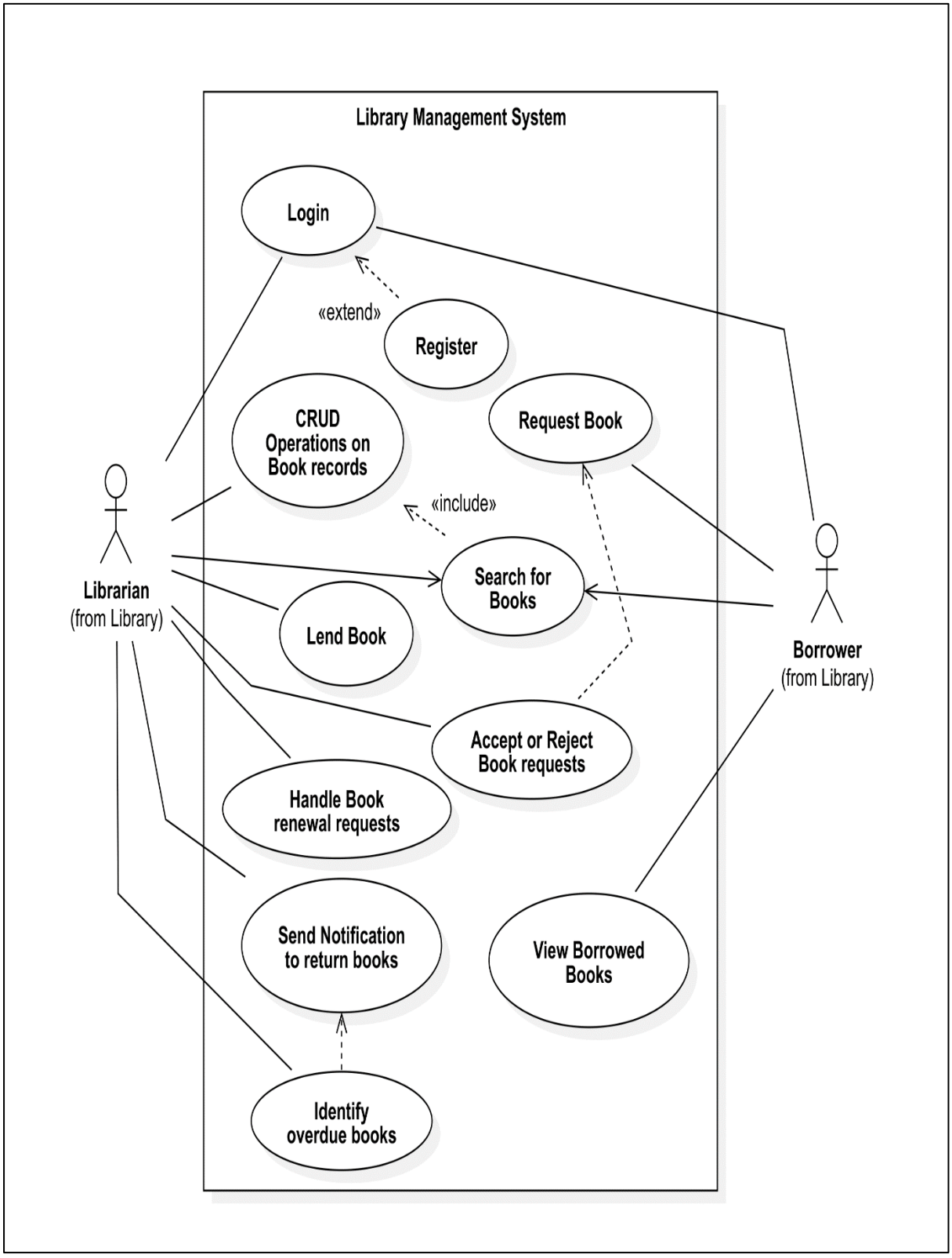
Efficiency:

- The system should optimize resource usage, ensuring efficient use of CPU, memory and storage
- Resource intensive tasks should be performed asynchronously to avoid blocking the user interface
- The system should provide monitoring and logging to track resource usage and identify bottlenecks
- Background tasks should be minimized to conserve battery life

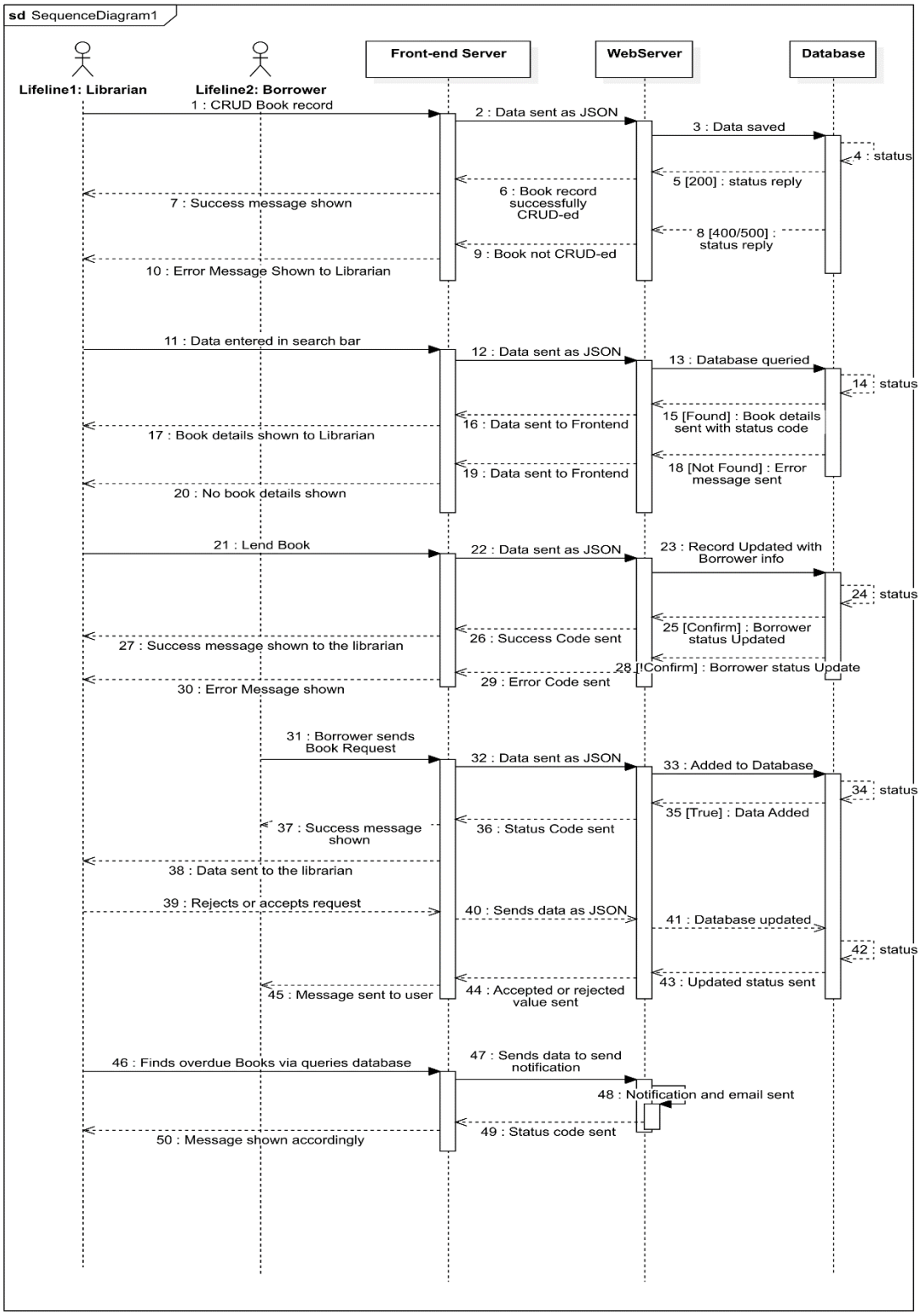
Interoperability:

- The system should provide APIs from which the database can be accessed, hence allowing webhooks and callbacks for real-time data exchange
- The system should support data import/export in common formats such as CSV, JSON etc.

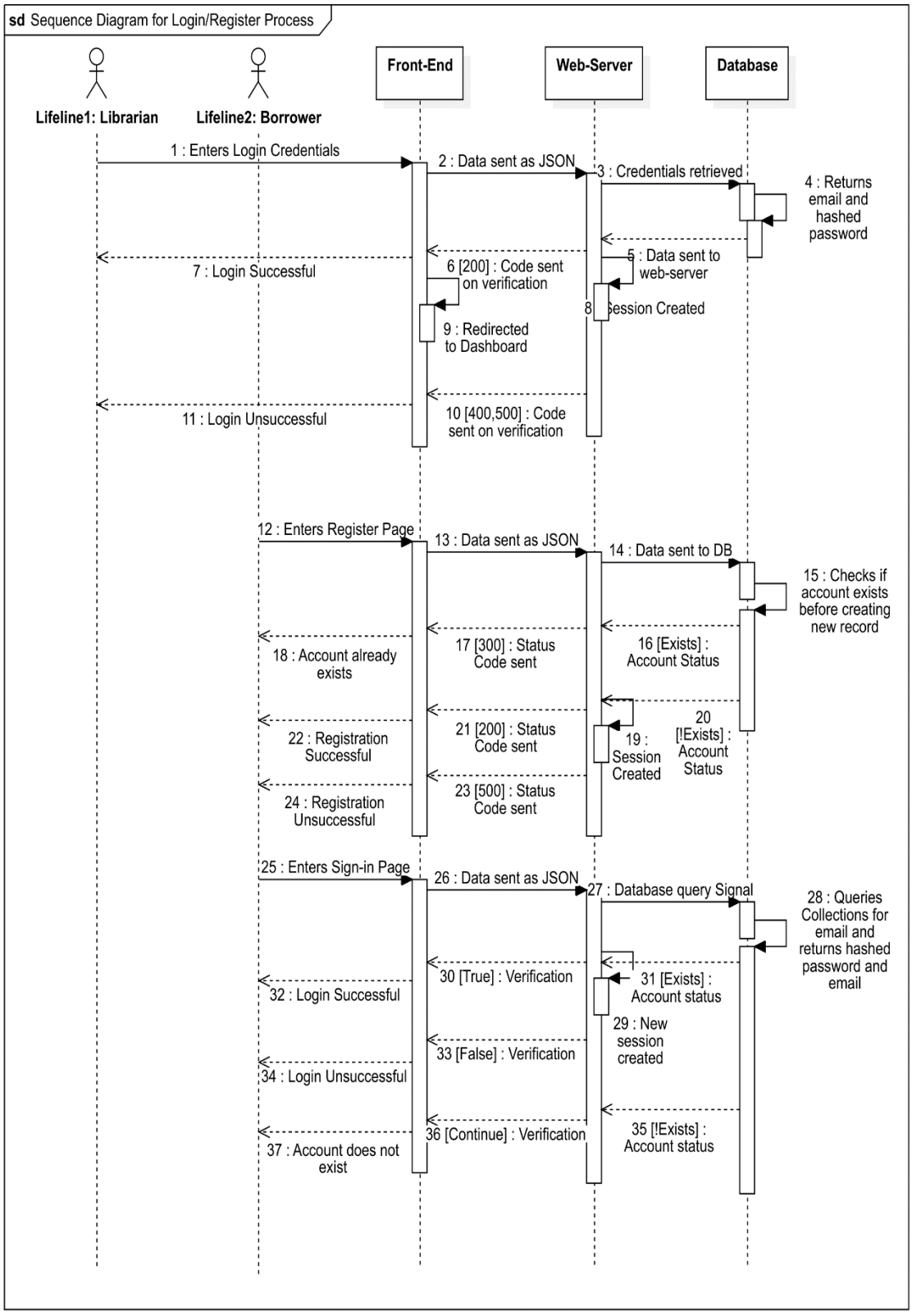
USE CASE DIAGRAM



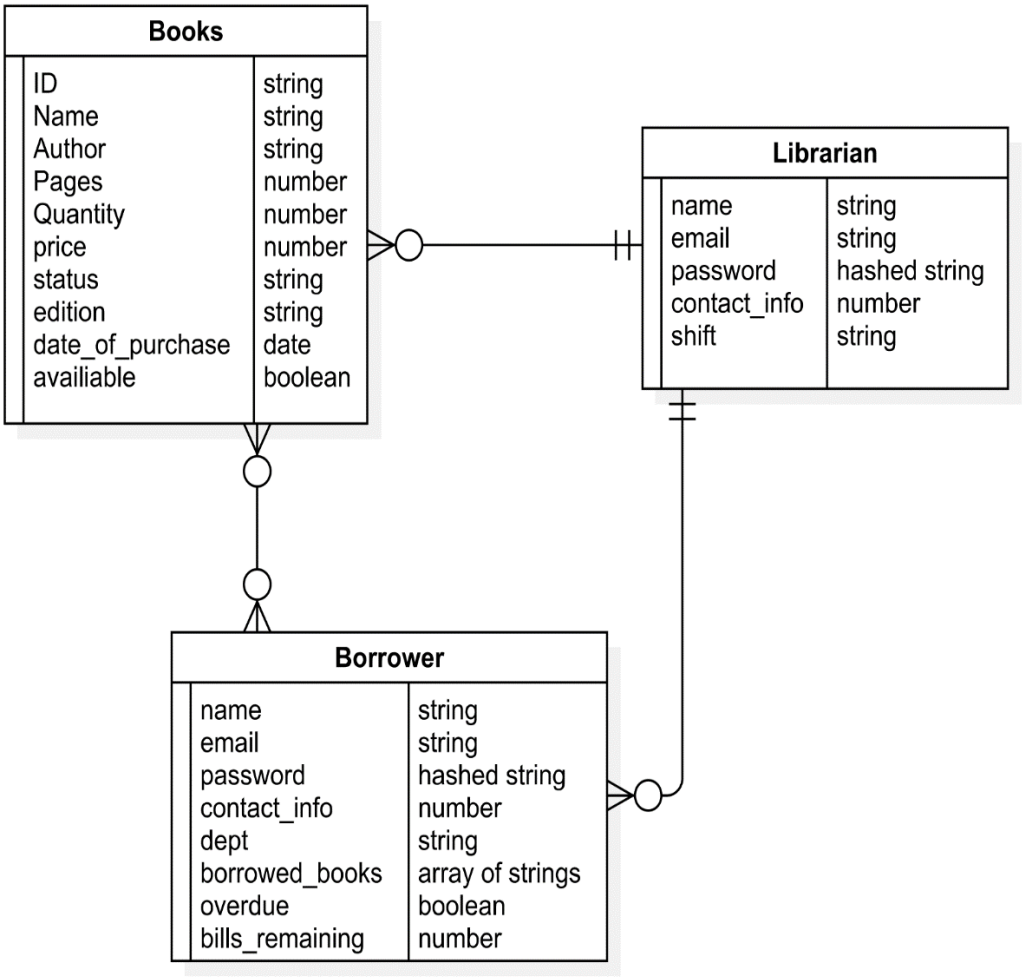
SEQUENCE DIAGRAMS



SEQUENCE DIAGRAM FOR LOGIN

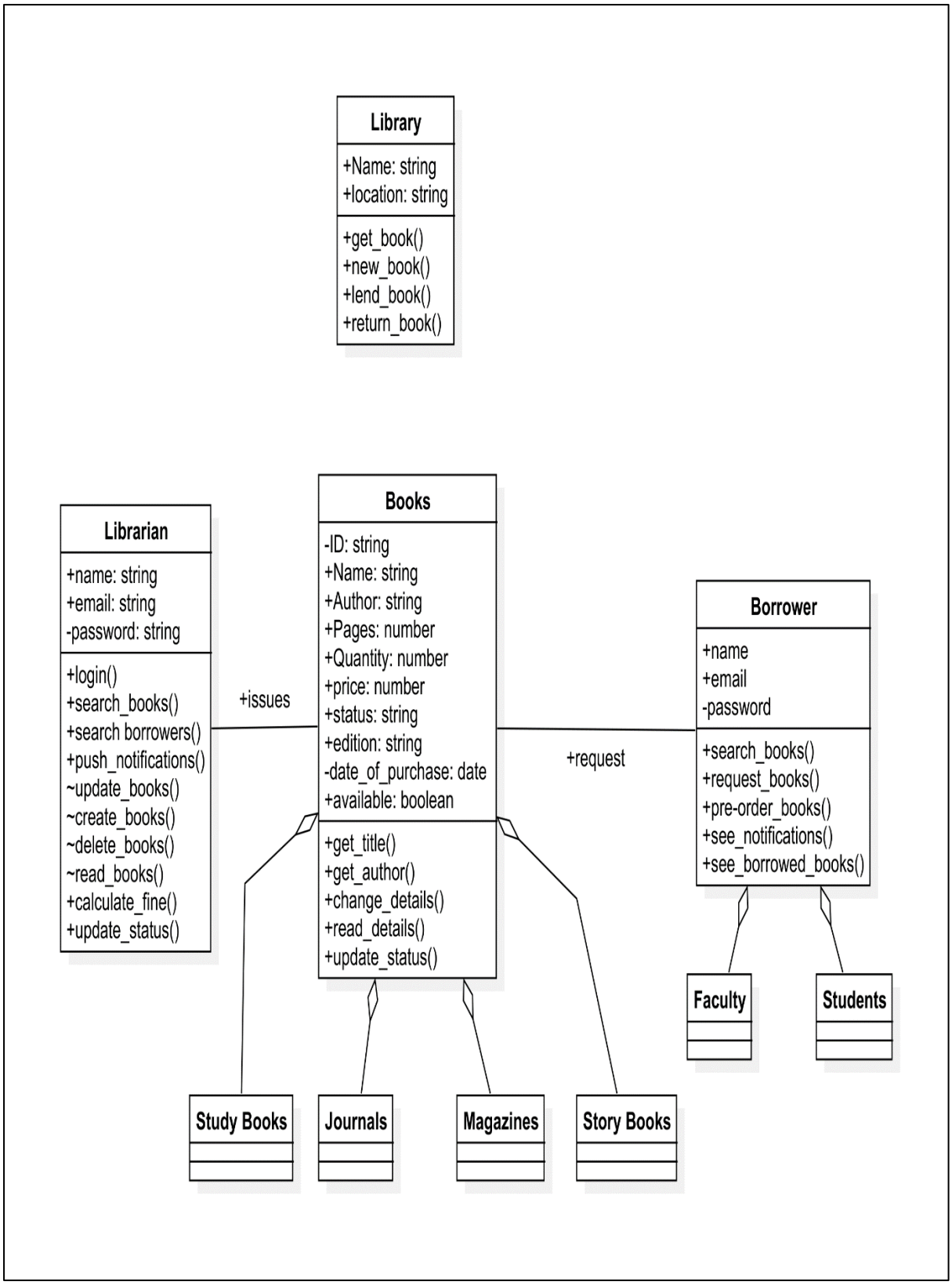


ERD DIAGRAM



The diagram shows relationships between these entities. A Librarian manages books, as indicated by the direct line connecting Librarian to Books. Borrowers can borrow multiple books, shown by the one-to-many relationship between Borrower and Books. Additionally, there is a recursive relationship within the Borrower entity to track the books they have borrowed (borrowed_books), overdue status, and any bills remaining. This comprehensive model ensures effective tracking and management of library resources, staff, and users.

CLASS DIAGRAM



Development Method: Agile Developed

Why?

Agile development methodologies offer numerous advantages for the development of our library management system, aligning closely with our project goals and requirements. It provides benefits such as

Flexibility: Allows For flexibility in responding to the changing requirements and priorities.
Iterative Progress: Enables us to delivery working increments of system regularly, providing early visibility and opportunities for course correction
Adaptability: Agile methodologies enable us to adapt to changing market conditions and emerging technologies
Continuous Improvement: Agile encourages continuous improvement through regular retrospectives and feedback loops.
Team Communication: Promotes close collaboration among cross-functional teams, enhancing communication and teamwork.

TEST CASES

As an integral part of the quality assurance process, test cases facilitate systematic evaluation, identification of defects, and ultimately, the enhancement of software quality. The following section presents a detailed compilation of test cases meticulously designed to assess the functionality and efficacy of the proposed library management system.

Module Name	Test Case ID	Test Scenario	Test Case Description	Test Steps	Test Priority	Test Expected Result	Environment Setup
User Authentication	TC001	Verify user login process	Users should be able to log in using their credentials	1. Open the login page. 2. Enter credentials. 3. Click the login button. 4. Verify login success.	High	User successfully logs in and is redirected to the homepage.	OS: Windows 10 Browser: Chrome 91.Version: 1.0
Book Management	TC002	Verify that librarians can add new books to the system	Librarians should be able to add books	1. Log in as a librarian. 2. Navigate to the "Add Book" page. 3. Enter book details. 4. Click the "Add" button. 5. Verify book is added.	High	Book is successfully added to the library database.	OS: Windows 10 Browser: Chrome 91.Version: 1.0

Book Search	TC003	Verify that borrowers can search for books using filters	Borrowers should be able to search for books	1. Log in as a borrower. 2. Navigate to the search page. 3. Enter search criteria (title, author, genre, availability) 4. Click search.	High	Relevant books are displayed based on the search criteria.	OS: Windows 10 Browser: Chrome 91.Version: 1.0
Overdue Notifications	TC004	Verify that users receive automated overdue notifications	Users should receive notifications for overdue books	1. Borrow a book. 2. Wait until the due date is passed. 3. Check for notification (email/SMS).	Medium	User receives a notification about the overdue book.	OS: Windows 10 Browser: Chrome 91.Version: 1.0
Book Reservation	TC005	Verify that users can reserve books online	Users should be able to reserve books online	1. Log in as a borrower. 2. Search for a book. 3. Click "Reserve" button. 4. Verify reservation is successful.	High	Book is successfully reserved and status updated in the system.	OS: Windows 10 Browser: Chrome 91.Version: 1.0
Reservation Approval	TC006	Verify that librarians can approve/reject	Librarians should be able to manage reservations	1. Log in as a librarian. 2. Navigate to the reservation	Medium	Reservation status is updated as approved/rejected.	OS: Windows 10 Browser: Chrome 91.Version: 1.0

		reservations		requests page. 3. Approve/reject requests. 4. Verify changes are saved.			
Fine Calculation	TC007	Verify that fines are calculated correctly for overdue books	System should calculate fines accurately	1. Borrow a book. 2. Return it after the due date. 3. Check fine details in user account.	High	Fine amount is calculated correctly based on overdue days.	OS: Windows 10 Browser: Chrome 91.Version: 1.0
User Borrowing History	TC008	Verify that users can view their borrowing history and fines	Users should be able to view their borrowing details	1. Log in as a borrower. 2. Navigate to "My Account" page. 3. Check borrowing history and fines details.	Medium	User sees accurate borrowing history and outstanding fines.	OS: Windows 10 Browser: Chrome 91.Version: 1.0
User Record Search	TC009	Verify that librarians can search for user records	Librarians should be able to search user records	1. Log in as a librarian. 2. Navigate to the user search page. 3. Enter search criteria (name,	Medium	Relevant user records are displayed based on search criteria.	OS: Windows 10 Browser: Chrome 91.Version: 1.0

				email, user ID). 4. Click search.			
Performance Testing	TC010	Verify system performance under heavy load	System should perform well under load	1. Simulate high number of concurrent users and transactions. 2. Monitor system performance.	Low	System performs efficiently without significant slowdowns.	OS: Windows 10 Browser: Chrome 91.Version: 1.0

Happy Path Testing

Happy Path Testing refers to scenarios where the system behaves as expected without any errors. For our project, happy path testing includes scenarios such as:

- Successful user login using correct credentials.
- Librarians adding new books to the inventory.
- Borrowers searching for books and making reservations.
- Automated notifications being sent for overdue books.

By validating these scenarios, we ensure that the core functionalities of the system work seamlessly under normal conditions.

Negative Testing

Negative Testing involves validating the system's robustness by using invalid inputs or performing unexpected actions. For our project, negative testing includes scenarios such as:

- Logging in with incorrect credentials and verifying the error message.

- Attempting to reserve a book that is already reserved.
- Adding a book with incomplete or invalid details.

This testing helps ensure that our system can handle errors properly and maintain stability under adverse conditions.

Edge Case Testing

Edge Case Testing examines the system's behaviour at the extreme boundaries of input values. For our project, edge case testing includes:

- Handling the maximum number of books, a user can borrow simultaneously.
- Managing the system's response under maximum data load during peak usage times.
- Verifying the system’s behaviour with the earliest and latest possible reservation dates.

These tests help ensure that the system operates effectively under boundary conditions.

Conclusion:

The development and implementation of our automated Library Management System mark a significant step forward in enhancing the efficiency, accuracy, and user satisfaction within the library environment. By addressing critical issues such as manual record-keeping errors, overdue book tracking, book reservations, and fine collection, our system provides a robust solution that benefits both librarians and borrowers.