Library Management System

Object Oriented Programming in Java- HNDIT 3012

KAN/IT/2022/F/0068

Department of Information Technology

Advanced Technological Institute

Kandy

2024

Table of Contents

Library Management System Documentation

1. Introduction

The **Library Management System (LMS)** is an automated platform designed to streamline library functions such as book issuance, returns, member management, and database handling. This system uses Java Swing for its graphical user interface (GUI), MySQL for data persistence, and follows the **MVC (Model-View-Controller)** architectural pattern for separation of concerns.

---

2. System Objectives

- **Automation of Library Functions**: Efficiently manage operations like book issuance, returns, member management, and more.

- **User-Friendly Interface**: Simplify the user experience for both library members and administrators.
- **Data Integrity**: Maintain accurate records for books, members, and transactions

---

3. Project Scope

The Library Management System covers the following functionalities:

1. **Book Management:**
    - **Add, Update, Remove, Search** books..
    - View books and track their availability (Issued/Returned status).
2. **Member Management:**
    - Add new members, deactivate existing ones.
    - View member details such as contact information and borrowed details.
3. **Login System:**
    - Secure login system for both library administrators and regular members.
4. **Database Integration:**
    - All system data (books, members, transactions) is stored in a **MySQL database** for persistent storage.

4. System Design

4.1 Architecture

The system is based on the **MVC architecture**, which consists of Three main components:

- **Model:** Represents the data and logic. It includes:
- Book class (manages book details)
- Member class (manages member data)
- Transaction class (manages book borrowing and returning details)
- **View:** Represents the user interface using Java Swing:
- GUI Components like LoginForm, BookManagementPanel, MemberDetailsPanel.
- **Controller or DAO:** Contains the business logic and database interaction:
- BookDAO (handles operations related to books)
- MemberDAO (handles member-related operations)
- IssueDAO & ReturnDAO (manages borrow/return operations)

4.2 Packages

1. **Model Package:**

- **Classes:** Admin, Book, Member, MembershipCard, User, IssueBook, ReturnBook.

Represents Constructors and Variable declaration .

- 2. **View Package:**

- **JFrames**: Welcome, LoginView, Menu, AddMembers, AddBooks, MemberManagement, BookManagement, Issuebooks, Returnbooks, Statistics.

- Provides graphical user interface (GUI) for the system.

3. **Controller or DAO Package:**

- **Classes**: BookDAO, MemberDAO, IssueBookDAO, ReturnDAO, StatisticsDAO LoginController.
- Manages database operations and authentication logic.

5. Database Design

Table	Action	Rows	Type	Collation	Size	Overhead
books	Browse Structure Search Insert Empty Drop	11	InnoDB	utf8mb4_general_ci	48.0 KiB	-
issuedbooks	Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_general_ci	16.0 KiB	-
members	Browse Structure Search Insert Empty Drop	9	InnoDB	utf8mb4_general_ci	32.0 KiB	-
return_book	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_general_ci	16.0 KiB	-
users	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
5 tables	Sum	37	InnoDB	utf8mb4_general_ci	128.0 KiB	0 B

Server: 127.0.0.1 » Database: librarydatabase » Table: users

Browse | Structure | SQL | Search | Insert | E:

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default
1	userId	varchar(50)	utf8mb4_general_ci		No	None
2	name	varchar(255)	utf8mb4_general_ci		No	None
3	password	varchar(255)	utf8mb4_general_ci		No	None
4	role	varchar(255)	utf8mb4_general_ci		No	None
5	email	varchar(255)	utf8mb4_general_ci		No	None
6	contactNo	varchar(255)	utf8mb4_general_ci		No	None
7	cardNumber	varchar(255)	utf8mb4_general_ci		No	None
8	expiryDate	date			No	None

Server: 127.0.0.1 » Database: librarydatabase » Table: books

Browse | Structure | SQL | Search | Insert | Expo

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default
1	bookID	varchar(50)	utf8mb4_general_ci		No	None
2	title	varchar(255)	utf8mb4_general_ci		No	None
3	author	varchar(255)	utf8mb4_general_ci		No	None
4	yearPublished	int(4)			No	None
5	genre	varchar(100)	utf8mb4_general_ci		No	None
6	isActive	tinyint(1)			No	None

Server: 127.0.0.1 » Database: librarydatabase » Table: members

Browse | Structure | SQL | Search | Insert | Export

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default
1	memberId	varchar(50)	utf8mb4_general_ci		No	None
2	name	varchar(100)	utf8mb4_general_ci		No	None
3	contactNo	varchar(20)	utf8mb4_general_ci		No	None
4	email	varchar(100)	utf8mb4_general_ci		No	None
5	membershipCardNo	varchar(50)	utf8mb4_general_ci		No	None
6	expiryDate	date			No	None
7	active	tinyint(1)			No	None
8	isActive	tinyint(1)			Yes	1

Server: 127.0.0.1 » Database: librarydatabase » Table: issuedbooks

Browse | Structure | SQL | Search | Insert | Expor

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comm
1	issueid	int(11)			No	None	
2	memberid	varchar(11)	utf8mb4_general_ci		No	None	
3	bookid	varchar(11)	utf8mb4_general_ci		No	None	
4	quantity	int(11)			No	None	
5	getdate	date			No	None	
6	returndate	date			No	None	

Server: 127.0.0.1 » Database: librarydatabase » Table: return_book

Browse | Structure | SQL | Search | Insert | Expor

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Com
1	returnid	int(11)			No	None	
2	memberid	varchar(255)	utf8mb4_general_ci		Yes	NULL	
3	bookid	varchar(11)	utf8mb4_general_ci		No	None	
4	quantity	int(11)			No	None	
5	returndate	date			No	None	

6. Testing

.1. Unit Testing:

Ensure the individual functionalities such as adding books, issuing books, etc., work as intended.

.2. Integration Testing:

Verify that the database operations (CRUD) interact seamlessly with the GUI components.

.3. Performance Testing:

Monitor the system for speed, especially during queries or user actions, ensuring efficient performance.

OOP Concepts Applied

The Library Management System (LMS) is designed using key Object-Oriented Programming (OOP) principles, ensuring modularity, scalability, and maintainability. These principles are applied as follows:

1. **Encapsulation**
   - The system's classes (e.g., Book, Member, Transaction) encapsulate their internal logic and data, exposing only the necessary methods and properties.
   - For instance, getter and setter methods are used to access and modify private fields, ensuring data integrity.
2. **Inheritance**
   - The LMS leverages inheritance to promote code reuse and hierarchical relationships.
   - The Admin and Member classes inherit common properties and methods (e.g., username, password, and login()) from the base User class.
   - Specialized methods unique to Admin (e.g., addBook()) and Member (e.g., borrowBook()) are implemented, showcasing polymorphism.
3. **Polymorphism**
   - Method overloading is used for handling multiple operations with the same method name but different parameters, such as in the addBook() method for handling different book types.
   - Method overriding is demonstrated in the Admin and Member classes, where the login() method is customized based on the user's role.
4. **Abstraction**
   - Abstract classes and interfaces are used to define core functionalities without exposing implementation details.
   - For example, an abstract Transaction class defines operations like issueBook() and returnBook(), which are implemented in specific subclasses.
5. **Composition**
   - The MembershipCard class is composed within the Member class, establishing a "has-a" relationship.

- This ensures that a membership card cannot exist independently without an associated member.

6. **Aggregation**
   - The Library class aggregates Book and Member objects, signifying a whole-part relationship where the existence of the Library is independent of individual books or members.

By applying these OOP principles, the LMS achieves a flexible and robust design, making it easy to extend, maintain, and adapt for future development.

---

7. Results

The Library Management System successfully provides:

1 **User-Friendly Interface**:

The interface is designed for ease of use, allowing administrators to manage books, members, and transactions effortlessly.

2 **Data Persistence**:

All data is stored securely in the MySQL database.

3 **Security**: User authentication ensures only authorized access to sensitive operations.

.

---

8. Future Enhancements

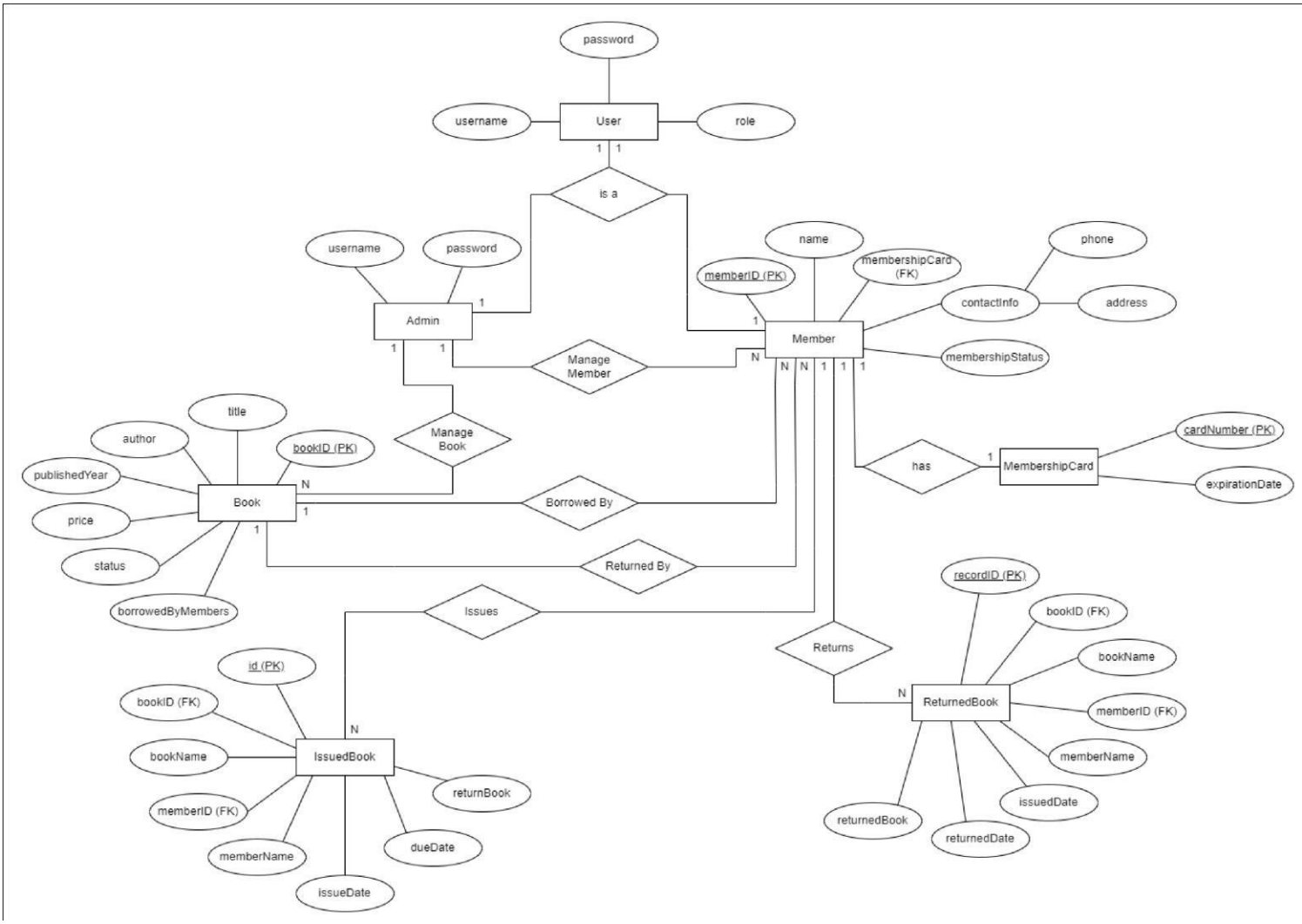
- **Search Features**: Add the ability to search books and members by various criteria (e.g., title, name, category)..
- **Reports**: Generate reports such as overdue books, transaction histories, and active members.

Advanced UI: Transition from Java Swing to JavaFX for enhanced styling and responsiveness.

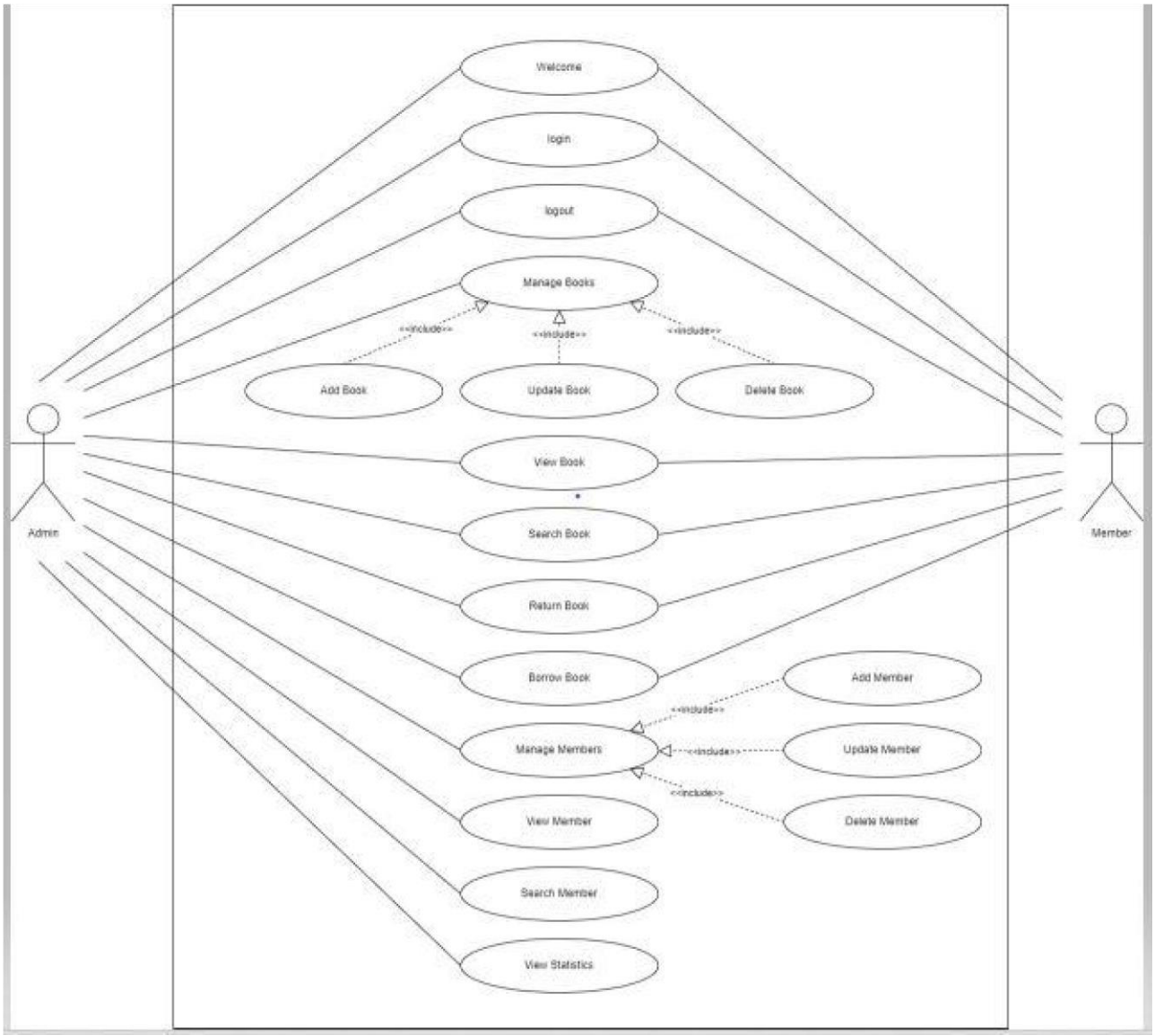
---

9. Diagrams

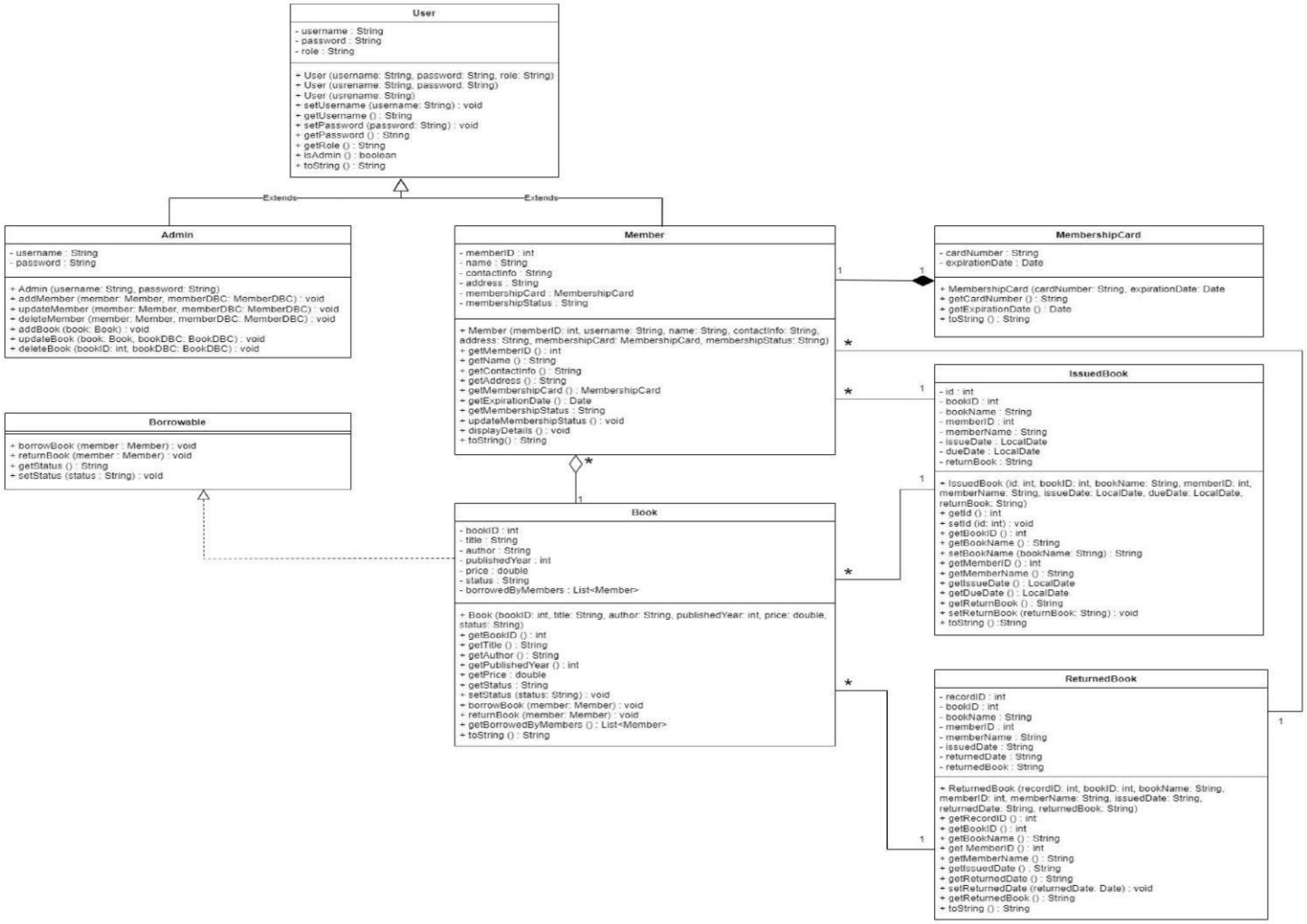
9.1. ER Diagram



9.2. Use Case Diagram



❖ **Class Diagram**



10. Conclusion

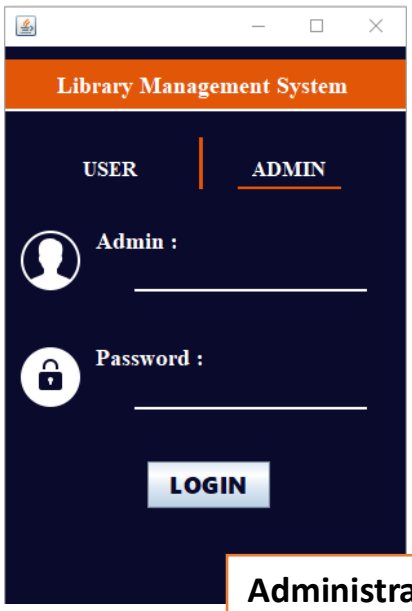
The **Library Management System (LMS)** developed in this project is a feature-rich application that effectively applies object-oriented programming principles. Its modular architecture, with a clear separation of concerns between the Controller, Model, and View components, ensures both maintainability and scalability.

Key components include the **Management class** for coordinating system operations, the **Database class** for handling database logic, and the **Constructor and Variable Declaration class** for managing data objects. The **Graphical User Interface (GUI)** provides an intuitive and user-friendly experience.

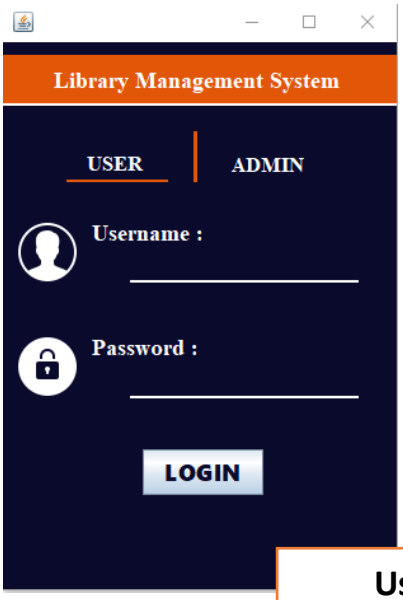
By leveraging OOP concepts such as **encapsulation**, **inheritance**, and **composition** the LMS achieves high modularity and flexibility, making it adaptable for future development. This report offers a comprehensive overview of the system's architecture, providing valuable insights for understanding and maintaining the LMS.

11. GUI of the System

❖ Login

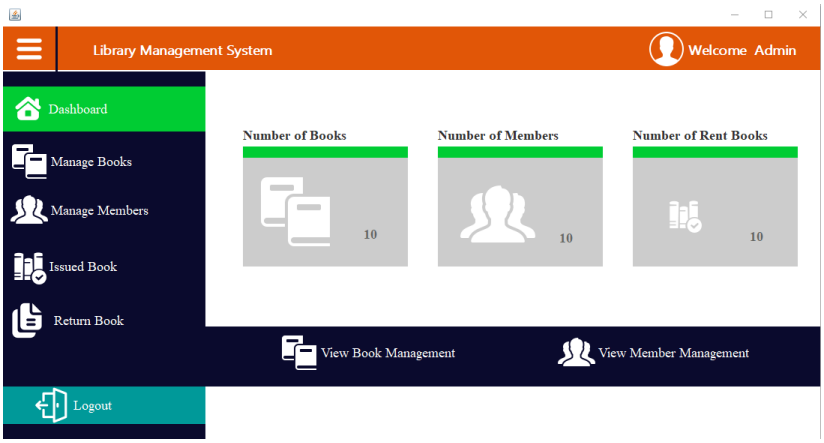


Administration Login

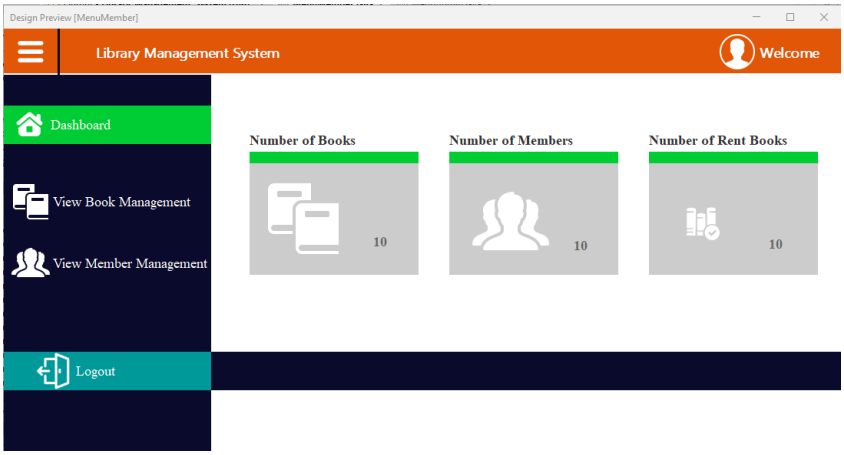


User Login

❖ Dashboard

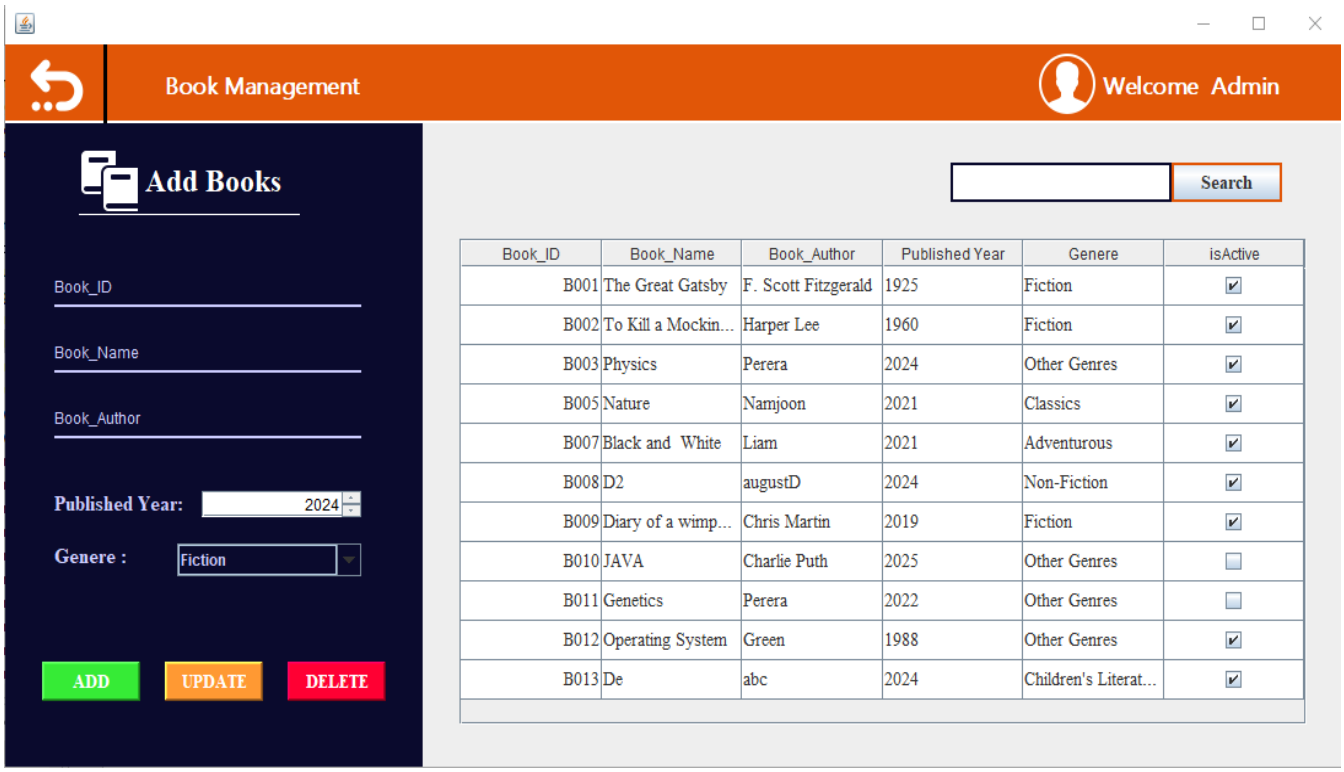


Admin Dashboard

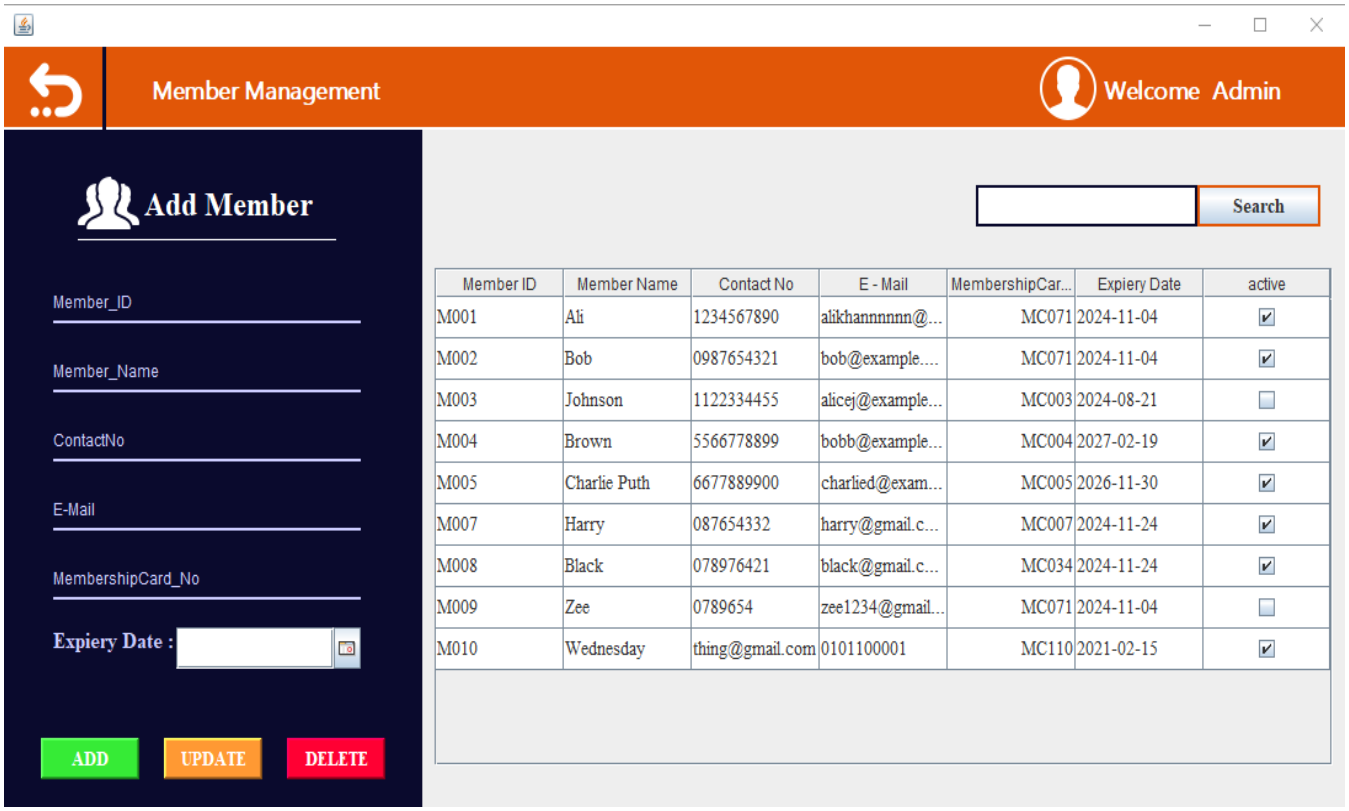


User Dashboard

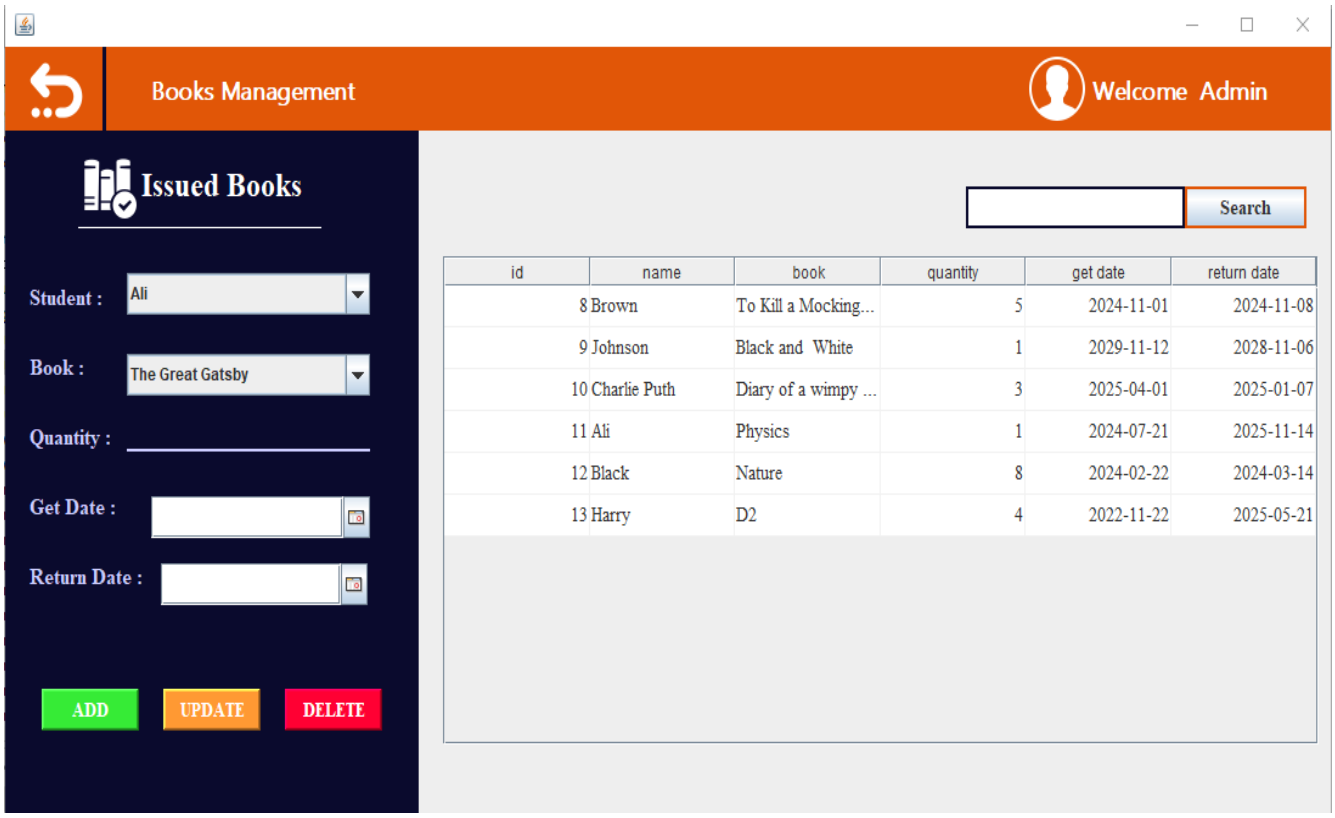
❖ Add Books



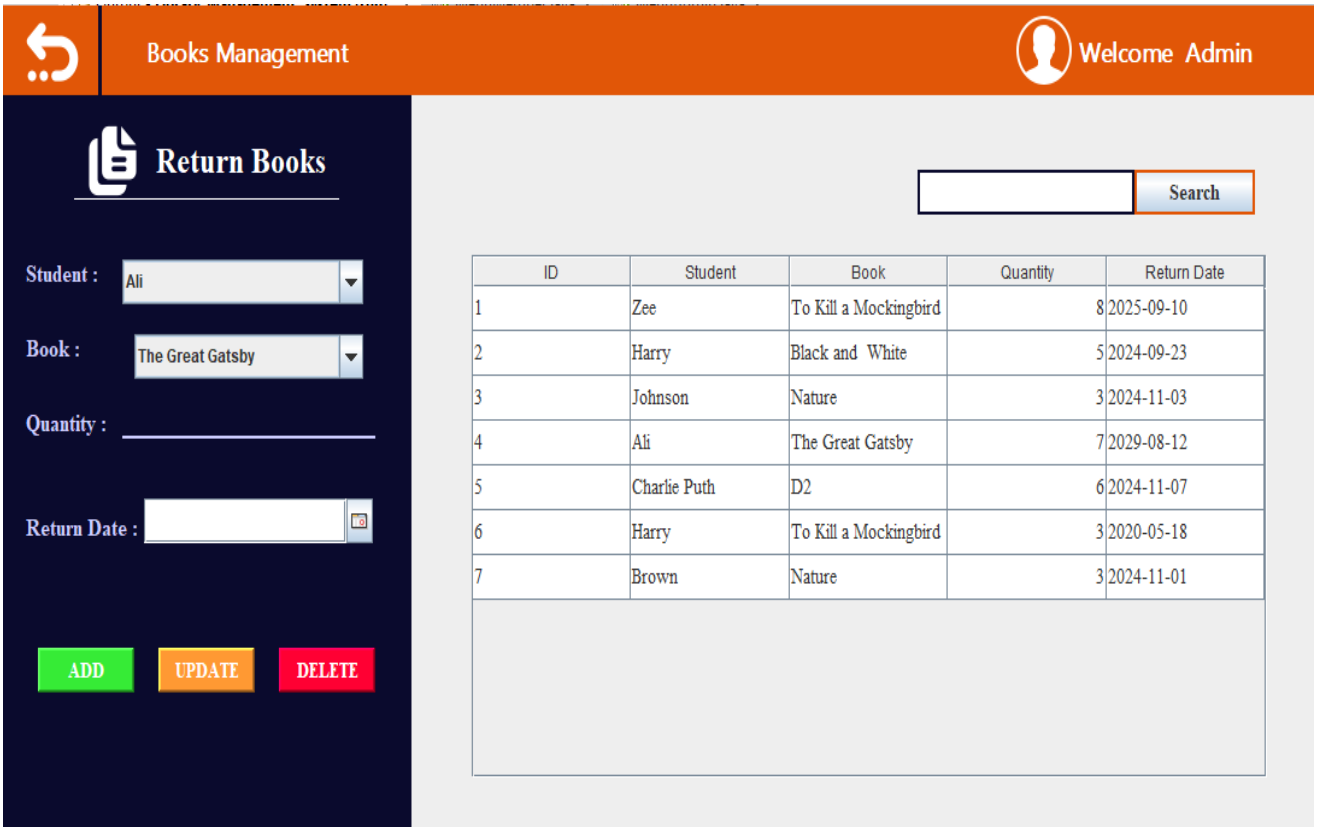
❖ Add Members



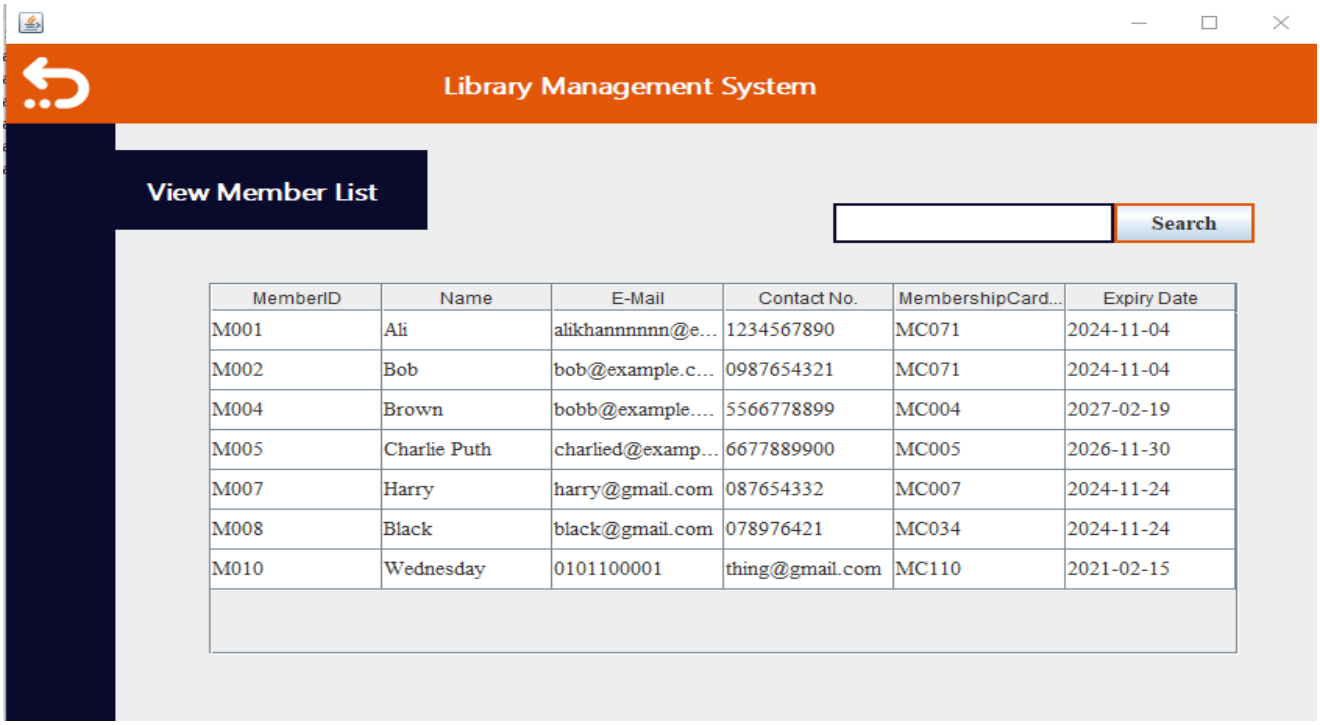
❖ Issued Books



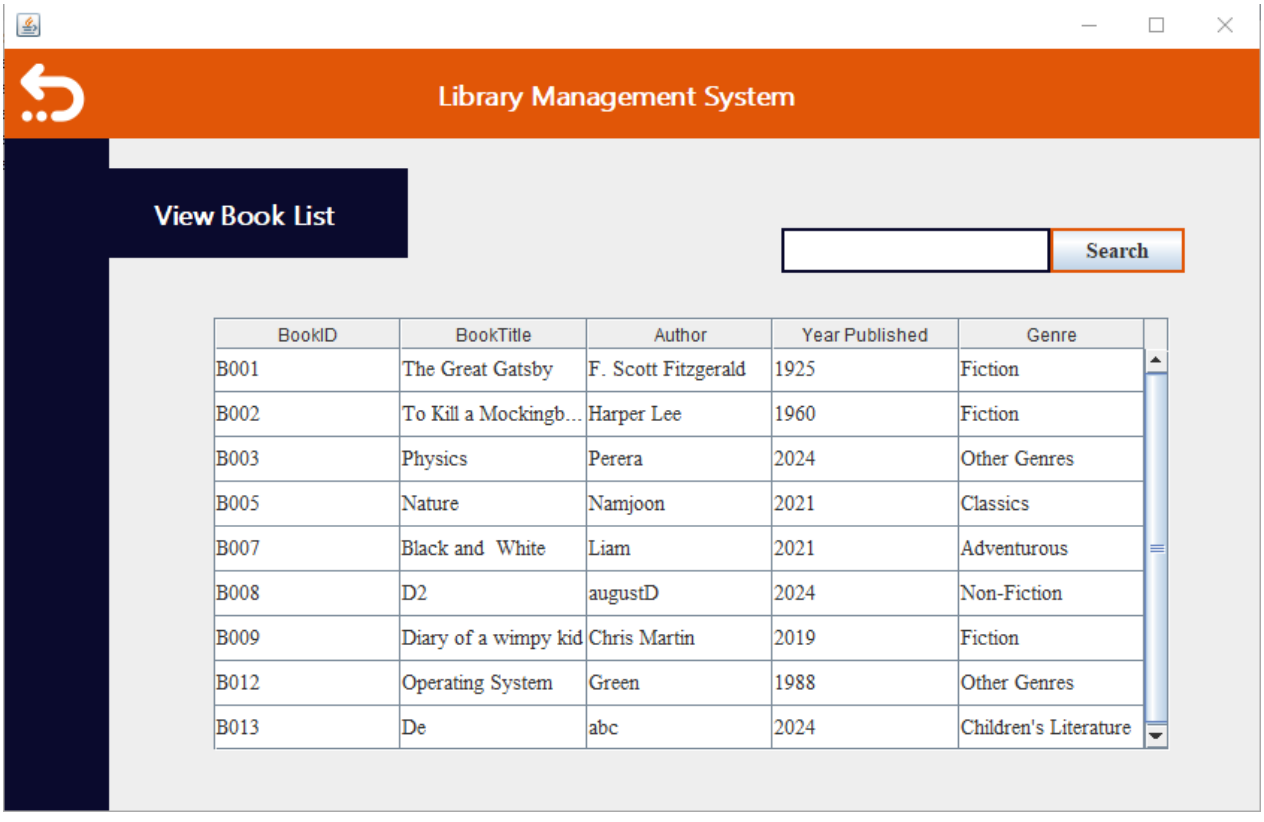
❖ Return Books



❖ View Member Management



❖ View Book Management



12. References

- Java: The Complete Reference" by Herbert Schildt
- Oracle Java Documentation
- ▪ Oracle Java Docs
- W3 School
- Kadhem Tech youtube channel