



**ADITYA UNIVERSITY**

# Library Management System

**Presented by Team 9**

G.Joshitha Rani(24B11CS119)

G.Yamini(24B11CS146)

P.Likhitha(24B11CS351)

Y.Gladis Esther(24B11CS496)

# Introduction

- The objective of this project is to design and develop a console-based Library Management System using Core Java and JDBC that helps in efficient management of library book records.
- The system allows the user to add, view, search, update, and delete book details such as book ID, title, author, price, and quantity, which are stored in a relational database.
- This project aims to provide practical understanding of **Java programming concepts**, **JDBC connectivity**, and **SQL-based CRUD** operations.
- It also focuses on implementing basic validation and exception handling to ensure data accuracy and reliability, without using any frontend technologies.

# Problem Statement

- Libraries require a reliable and organized way to manage book records such as book name, author, price, and quantity. As the number of books increases, maintaining and updating these details without a proper system becomes inefficient and time-consuming.
- In the absence of a structured application, tasks like adding new books, updating book information, deleting outdated records, and searching for books become difficult and prone to errors. There is also no centralized way to store and retrieve book data securely and accurately.
- Therefore, there is a need for a **simple Java-based application** that connects to a database and supports **basic CRUD operations** (Create, Read, Update, Delete) to manage library book records effectively. The system should ensure easy data handling, better organization, and improved accuracy without using any frontend technologies.

# Modules

## **1. Book Management Module**

- Allows adding new book records
- Allows updating book details
- Allows deleting book records

## **2. Search Module**

- Search books using:
  - Book ID
  - Book Name
  - Author Name

### **3. Database Connectivity Module**

- Establishes connection between Java application and database
- Executes SQL queries using JDBC

### **4. Validation Module**

- Validates book price
- Validates book quantity
- Prevents incorrect data entry

### **5. Exception Handling Module**

- Handles invalid book ID
- Handles database and input errors



## ADMIN DASHBOARD



2

Books Listed



6

Times Book Issued



3

Times Books Returned



6

Registered Users



2

Authors Listed



6

Listed Categories

# Role Contributions

## ➤ G.Yamini -**Book Management Developer**

- Works on adding, updating, and deleting book records
- Handles book-related logic in Java

## ➤ P.Likhitha- **Search & Validation Developer**

- Implements book search by ID, name, and author
- Handles price and quantity validation

## ➤ G.Joshitha Rani- **Database & JDBC Developer**

- Designs the books table
- Manages JDBC connection and SQL queries

## ➤ Y.Gladis Esther- **Testing & Documentation Lead**

- Tests all CRUD operations
- Prepares PPT, documentation, and review content



# Software Requirements

**Operating System:** Windows / Linux

**Programming Language:** Java (J2SE)

**JDK Version:** JDK 8 or above

**Database:** MySQL / Oracle / SQLite

**Database Connectivity:** JDBC (Java Database Connectivity)

**IDE:** VS Code / Eclipse / IntelliJ / NetBeans



# Conclusion

- The Library Management System is designed to provide a simple and effective solution for managing library book records such as book name, author, price, and quantity. The project addresses the difficulties faced by libraries in maintaining and updating book information without a proper system.
- By using Core Java and JDBC, the proposed system allows book records to be stored securely in a database and managed through basic CRUD operations. The system helps in organizing data, reducing manual effort, and minimizing errors while handling library records.

- This project provides practical exposure to Core Java, JDBC connectivity, SQL operations, and exception handling. Since the system is console-based and backend-focused, it strengthens core programming concepts and builds a strong foundation for future enhancements, such as adding a user interface or extending library functionalities.