AJEENKYA
D Y PATIL UNIVERSITY
THE INNOVATION UNIVERSITY

School of Engineering

A Mini Project Report on

# "Library Management System"

submitted in partial fulfillment of the requirements

of the degree of

***Bachelor of Technology in Data Science***

By

**Anushka Mhalankar**

**2021-B-01102003B**

Under the Guidance of

**Prof. Sandeep M. Chitalkar**

AJEENKYA
D Y PATIL UNIVERSITY
THE INNOVATION UNIVERSITY

November 2022

School of Engineering, **Ajeenkya D Y Patil University**

AJEENKYA D Y PATIL UNIVERSITY
THE INNOVATION UNIVERSITY
School of Engineering

# CERTIFICATE

This is to certify that the project entitled "Library Management System" is a bonafide work of "Anushka Mhalankar" (URN No. 2021-B-01102003B) submitted to the Ajeenkya D Y Patil University, Pune in partial fulfillment of the requirement for the award of the degree of "*Bachelor of Technology in Data Science*".

**Prof. Sandeep Chitalkar**

Internal-Examiner

# ABSTRACT

Online Library Management System is a system that maintains the information about the books present in the library, their authors, the members of the library to whom the books are issued, library staff, and all. This is very difficult to organize manually. Maintenance of all this information manually is a very complex task. Owing to the advancement of technology, the organization of an Online Library becomes much simpler by using mobile applications. Online Library Management has been designed to computerize and automate the operations performed over the information about the members, book issues and returns, and all other operations. This computerization of the library helps in many instances of its maintenance. It reduces the workload of management as most of the manual work done is reduced.

**KEYWORDS:** workload, oblivious computation, mobile applications.

# TABLE OF CONTENTS

# CHAPTER 1: INTRODUCTION

## 1.1 INTRODUCTION

The project has been made, using JDBC technology, for storing school/college library-related information in a database to reduce the manual workload of the staff and to make library management more easy and accessible to all. The project has a menu at the start from which the operations to conduct can be chosen by the user (librarian or administrator or student). Any student can register in the library free of cost and take advantage of the benefits of an online library which can be used anywhere and anytime and free of cost.

This chapter gives an overview of the background, the scope of work, and the operating environment of the system.

## 1.2 SCOPE OF WORK

E-Library Management System is an application that refers to library systems that are generally small or medium in size. It is used by librarians to manage the library using a computerized system where he/she can add new books, videos, and Page sources. Books and student maintenance modules are also included in this system which would keep track of the students using the library and also a detailed description of the books a library contains. With this computerized system, there will be no loss of book records or member records which generally happens when a non-computerized system is used. All these modules are able to help librarians manage the library with more convenience and in a more efficient way as compared to library systems which are not computerized.

## 1.3 OPERATING ENVIRONMENT

| | |
|---|---|
| • PROCESSOR | INTEL CORE PROCESSOR<br>OR BETTER PERFORMANCE |
| • OPERATING SYSTEM | WINDOWS 10 |
| • DATABASE | MySQL |
| • JAVA COMPILER | IntelliJ IDEA |

## 1.4 DETAIL DESCRIPTION OF TECHNOLOGY USED

➔ *Operating system* - Windows 10 is used as the operating system as it is stable and supports more features and is more user-friendly.

➔ *Database MySQL* - MySQL is used as a database as it is easy to maintain and retrieve records by simple queries which are in the English language which are easy to understand and easy to write.

➔ *Development tools and Programming language*- JAVA is used to write the code for the management system.

➔ *JAVA DATABASE CONNECTIVITY (JDBC)-* Java database connectivity (JDBC) is the JavaSoft specification of a standard application programming interface (API) that allows Java programs to access database management systems. The JDBC API consists of a set of interfaces and classes written in the Java programming language.

# CHAPTER 2: PROPOSED SYSTEM

## 2.1 PROPOSED SYSTEM

To solve the inconveniences mentioned earlier, an Online Library is proposed. The proposed system contains the following features:

- ➔ Individually each member will have his account through which he can access the information he needs.
- ➔ Book details like authors, number of copies totally maintained by the library, presently available number of books, reference books, non-reference books, etc. all this information can be made handy.
- ➔ Administrator can add, and update the books.
- ➔ Time consumption is low which gives accurate results, and reliability can be improved with the help of security.

## 2.2 OBJECTIVES OF THE SYSTEM

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter.

The aims and objectives are as follows:

- ➔ Search for the availability of a book
- ➔ Search books by the book ID in the database
- ➔ Search books by the Author's name in the database
- ➔ Add book details to the database
- ➔ Update book details to the database
- ➔ Register students in the library
- ➔ Update student details in the library

## 2.3 USER REQUIREMENTS

- ➔ Allow the librarian to add and remove new members.
- ➔ Allow the user to search for books based on title, publication date, author, etc., and find their location in the library.
- ➔ Users can request, reserve, or renew a book.
- ➔ Librarian can add and manage the books.
- ➔ Usability is the main non-functional requirement for a library management system.
- ➔ The software should be easily maintainable and adding new features and making changes to the software must be as simple as possible.

# CHAPTER 3: SOURCE CODE

## CREATING CONNECTION AND MENU:

```java
package com.company;

import java.sql.*;
import java.util.Scanner;

public class asliproject implements haha {
    public static void main(String[] args) throws SQLException {
        Scanner sc = new Scanner(System.in);
        Scanner ss = new Scanner(System.in);
        Connection conn = DriverManager.getConnection(url, user, pass);
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery( sql: "select * from book_records");
        ResultSet rss = stmt.executeQuery( sql: "select * from student_details");


        startMenu(conn);

    }

    8 usages
    static void startMenu(Connection conn) throws SQLException {
        Scanner sc = new Scanner(System.in);
        System.out.println("Welcome to X-Library\nPlease choose one of the options.");
        System.out.println("1. Add a book to library.");
        System.out.println("2. Search a Book with its Serial number.");
        System.out.println("3. Search Books With Author Name."); //prob!!!!!!!!!
        System.out.println("4. Show all Books and their related Information.");
        System.out.println("5. Register a Student.");
        System.out.println("6. Show All Registered Students.");
        System.out.println("7. Update book quantity.");
        System.out.println("8. Please enter 0 to EXIT.");
        char ch = sc.next().charAt(0);
        switch (ch) {
            case '1':
                addBook(conn);
                break;
            case '2':
                searchSno(conn);
                break;
            case '3':
                searchAu(conn);
            case '4':
                DispBooks(conn);
            case '5':
                RegistrStd(conn);
            case '6':
                DispStudents(conn);
            case '7':
                updateBook(conn);
            case '8':
                System.out.println("Thank you for using this application");
                System.exit( status: 0);
                break;
        }

    }
```

# ADDING BOOKS:

```
1 usage
static void addBook(Connection conn) throws SQLException {
    Scanner sc = new Scanner(System.in);
    Scanner ss = new Scanner(System.in);
    System.out.println("Enter SNo.: ");
    int b_id = sc.nextInt();
    System.out.println("Enter Book Name");
    String b_name = ss.nextLine();
    System.out.println("Enter Author Name");
    String a_name = ss.nextLine();
    System.out.println("Enter Book Quantity");
    int b_qty = sc.nextInt();

    PreparedStatement pstmt = conn.prepareStatement(insert_query);
    pstmt.setInt( parameterIndex: 1, b_id);
    pstmt.setString( parameterIndex: 2, b_name);
    pstmt.setString( parameterIndex: 3, a_name);
    pstmt.setInt( parameterIndex: 4, b_qty);
    try {
        pstmt.executeUpdate();
        System.out.println("Book successfully added!");
    } catch (SQLException ex) {

        System.out.println("Something went wrong.");
        ex.printStackTrace();
    }

    startMenu(conn);
}

1 usage
```

# SEARCH BY SNo. :

```
1 usage
static void searchSno(Connection conn) throws SQLException {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter Serial No/ Book ID: ");
    int id = sc.nextInt();
    Statement stmt = conn.createStatement();
    ResultSet rs = stmt.executeQuery( sql: "SELECT * FROM book_records where b_id ="+id);
    while(rs.next()){
        String b_name = rs.getString( columnLabel: "b_name");
        String a_name = rs.getString( columnLabel: "a_name");
        int b_qty = rs.getInt( columnLabel: "b_qty");
        System.out.println("Book ID:\t"+id+"\nBook Name:\t"+b_name+"\nAuthor Name:\t"+a_name+"\nBook Quantity:\t"+b_qty);
    }
    startMenu(conn);
}
```

# SEARCH BY AUTHOR'S NAME:

```
1 usage
static void searchAu(Connection conn) throws SQLException{
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter Author's Name: ");
    String a_name = sc.next();
    Statement stmt = conn.createStatement();
    ResultSet rs = stmt.executeQuery( sql: "SELECT * FROM book_records WHERE a_name ='"+a_name+"';");
    while(rs.next()){
        int b_id = rs.getInt( columnLabel: "b_id");
        String b_name = rs.getString( columnLabel: "b_name");
        int b_qty = rs.getInt( columnLabel: "b_qty");
        System.out.println("Book ID:\t"+b_id+"\nBook Name:\t"+b_name+"\nAuthor Name:\t"+a_name+"\nBook Quantity:\t"+b_qty);
    }
    startMenu(conn);
}
```

# DISPLAY A LIST OF BOOKS:

```
1 usage
static void DispBooks(Connection conn) throws SQLException{
    Scanner sc = new Scanner(System.in);
    Statement stmt = conn.createStatement();
    ResultSet rs = stmt.executeQuery( sql: "SELECT * FROM book_records");
    while(rs.next()){
        int b_id = rs.getInt( columnLabel: "b_id");
        String b_name = rs.getString( columnLabel: "b_name");
        String a_name = rs.getString( columnLabel: "a_name");
        int b_qty = rs.getInt( columnLabel: "b_qty");
        System.out.println("Book ID:\t"+b_id+"\nBook Name:\t"+b_name+"\nAuthor Name:\t"+a_name+"\nBook Quantity:\t"+b_qty);
    }
    startMenu(conn);

}
```

# REGISTER A STUDENT:

```
131 @   static void RegistrStd(Connection conn)throws SQLException{
132         Scanner sc = new Scanner(System.in);
133         Scanner ss = new Scanner(System.in);
134         System.out.println("Enter Student ID.: ");
135         int s_id = sc.nextInt();
136         System.out.println("Enter Student Name: ");
137         String s_name = ss.nextLine();
138         System.out.println("Enter Student Phone No.: ");
139         int s_phNo = sc.nextInt();
140         System.out.println("Enter Student Department: ");
141         String s_dept = ss.nextLine();
142
143         PreparedStatement pstmt = conn.prepareStatement(insert_queryy);
144         pstmt.setInt( parameterIndex: 1, s_id);
145         pstmt.setString( parameterIndex: 2, s_name);
146         pstmt.setInt( parameterIndex: 3, s_phNo);
147         pstmt.setString( parameterIndex: 4, s_dept);
148         try {
149             pstmt.executeUpdate();
150             System.out.println("Student Registered Successfully!!");
151         } catch (SQLException ex) {
152
153             System.out.println("Something went wrong.");
154             ex.printStackTrace();
155         }
156
157         startMenu(conn);
158
159     }
```

# DISPLAY STUDENTS:

```
        1 usage
161 @   static void DispStudents(Connection conn)throws SQLException{
162         Scanner sc = new Scanner(System.in);
163         Statement stmt = conn.createStatement();
164         ResultSet rss = stmt.executeQuery( sql: "SELECT * FROM student_details");
165         while(rss.next()){
166             int s_id = rss.getInt( columnLabel: "s_id");
167             String s_name = rss.getString( columnLabel: "s_name");
168             int s_phNo = rss.getInt( columnLabel: "s_phNo");
169             String s_dept = rss.getString( columnLabel: "s_dept");
170             System.out.println("Student ID:\t"+s_id+"\nStudent Name:\t"+s_name+"\nStudent Phone No.:\t"+s_phNo+
171                     "\nStudent Department:\t"+s_dept);
172         }
173         startMenu(conn);
174     }
175
```

# UPDATE BOOK:

```
1 usage
static void updateBook(Connection conn)throws SQLException{
    Scanner sc = new Scanner(System.in);
    Scanner ss = new Scanner(System.in);
    System.out.println("Enter Book ID to update information: ");
    int id = sc.nextInt();
    System.out.println("Enter Book Name");
    String b_name = ss.nextLine();
    System.out.println("Enter Author Name");
    String a_name = ss.nextLine();
    System.out.println("Enter Book Quantity");
    int b_qty = sc.nextInt();

    PreparedStatement pstmt = conn.prepareStatement(insert_queryU );
    pstmt.setInt( parameterIndex: 1,id);
    pstmt.setString( parameterIndex: 2, b_name);
    pstmt.setString( parameterIndex: 3, a_name);
    pstmt.setInt( parameterIndex: 4, b_qty);
    try {
        pstmt.executeUpdate();
        System.out.println("Book Information Updated Successfully!!");
    } catch (SQLException ex) {

        System.out.println("Something went wrong.");
        ex.printStackTrace();
    }
    startMenu(conn);

}
```

# CHAPTER 4: RESULTS AND DISCUSSION

**ADDING A BOOK TO THE DB:**

```
Welcome to X-Library
Please choose one of the options.
1. Add a book to library.
2. Search a Book with its Serial number.
3. Search Books With Author Name.
4. Show all Books and their related Information.
5. Register a Student.
6. Show All Registered Students.
7. Update book quantity.
8. Please enter 0 to EXIT.
1
Enter SNo.:
04
Enter Book Name
ALIBABA AND THE 40 CHOOR
Enter Author Name
XYZ
Enter Book Quantity
5
Book successfully added!
```

**SEARCH A BOOK BY SNo. :**

```
Welcome to X-Library
Please choose one of the options.
1. Add a book to library.
2. Search a Book with its Serial number.
3. Search Books With Author Name.
4. Show all Books and their related Information.
5. Register a Student.
6. Show All Registered Students.
7. Update book quantity.
8. Please enter 0 to EXIT.
2
Enter Serial No/ Book ID:
04
Book ID:    4
Book Name:  ALIBABA AND THE 40 CHOOR
Author Name:    XYZ
Book Quantity:  5
```

**SEARCH BOOK BY AUTHOR'S NAME:**

```
Welcome to X-Library
Please choose one of the options.
1. Add a book to library.
2. Search a Book with its Serial number.
3. Search Books With Author Name.
4. Show all Books and their related Information.
5. Register a Student.
6. Show All Registered Students.
7. Update book quantity.
8. Please enter 0 to EXIT.
3
Enter Author's Name:
XYZ
Book ID:    4
Book Name:  ALIBABA AND THE 40 CHOOR
Author Name:    XYZ
Book Quantity:  5
```

**REGISTER A STUDENT:**

```
Welcome to X-Library
Please choose one of the options.
1. Add a book to library.
2. Search a Book with its Serial number.
3. Search Books With Author Name.
4. Show all Books and their related Information.
5. Register a Student.
6. Show All Registered Students.
7. Update book quantity.
8. Please enter 0 to EXIT.
5
Enter Student ID.:
01
Enter Student Name:
ABABAB
Enter Student Phone No.:
8965327854
Enter Student Department:
Engineering
Student Registered Successfully!!
```

**SHOW ALL BOOKS:**

```
Welcome to X-Library
Please choose one of the options.
1. Add a book to library.
2. Search a Book with its Serial number.
3. Search Books With Author Name.
4. Show all Books and their related Information.
5. Register a Student.
6. Show All Registered Students.
7. Update book quantity.
8. Please enter 0 to EXIT.
4
Book ID:    1
Book Name:
Author Name:    HArrrrryyyy
Book Quantity:  3
Book ID:    2
Book Name:  Harry potter
Author Name:    JK Rolling
Book Quantity:  12
Book ID:    3
Book Name:  Percy Jackson
Author Name:    ABC
Book Quantity:  2
Book ID:    4
Book Name:  ALIBABA AND THE 40 CHOOR
Author Name:    XYZ
Book Quantity:  5
```

**UPDATE BOOK INFORMATION:**

```
Welcome to X-Library
Please choose one of the options.
1. Add a book to library.
2. Search a Book with its Serial number.
3. Search Books With Author Name.
4. Show all Books and their related Information.
5. Register a Student.
6. Show All Registered Students.
7. Update book quantity.
8. Please enter 0 to EXIT.
7
Enter Book ID to update information:
1
Enter Book Name
HARI PUTTAR
Enter Author Name
JKR
Enter Book Quantity
4
Book Information Updated Successfully!!
```

# CHAPTER 5: CONCLUSION

We can improve the traditional method of working in a library because the traditional method includes doing all the things in manual mode which is slow, less efficient, less secure, and challenging to manage. The solution to this is an online library management system that takes care of all the work by automating and digitizing the whole process. Our application is based on Java and is linked to a relational database (SQL). The frontend part has been coded using Java and the backend is supported and connected with database using java, its libraries and APIs. With the increase in the workload of the library, new features can be added to the existing application to make it relevant in the future as well.
This JDBC project provides a computerized version of the library management system which will benefit the students as well as the staff of the library. It makes the entire process online where students can search books, staff can add and update book information and student details.