

LIBRARY MANAGEMENT SYSTEM

CS23333 – Object Oriented Programming Using JAVA Project Report

Submitted by

DHANUSH M - 231001033

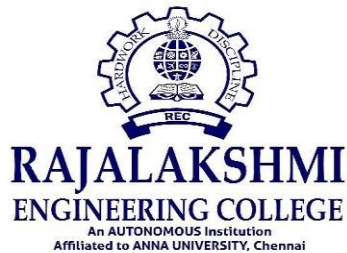
DINESH KUMAR B - 231001039

Of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY



DEPARTMENT OF INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE

NOVEMBER-2024

BONAFIDE CERTIFICATE

Certified that this project titled “Library Management System” is the bonafide work of **DHANUSH(231001033),DINESH KUMAR B(231001039)** who carried out the project work under my supervision.

SIGNATURE

Dr.P.Valarmathie

HEAD OF THE DEPARTMENT

SIGNATURE

Mrs.Usha S

COURSE INCHARGE

Assistant Professor(S.G)

Department of Information Technology
Rajalakshmi Engineering College

Department of Information Technology
Rajalakshmi Engineering College

This project is submitted for CS23333 – Object Oriented Programming Using
JAVA held on _____

INTERNAL EXAMINER

EXTERNAL EXAMINER

Table of Contents:

CHAPTER NO.	TITLE	PAGE NO.
1	1.1 ABSTRACT	7
	1.2INTRODUCTION	7
	1.3 PURPOSE	7
	1.4 SCOPE OF PROJECT	8
	1.5 SOFTWARE REQUIREMENT SPECIFICATION	8
2	SYSTEM FLOW DIAGRAM	12
	2.1 USE CASE DIAGRAM	12
	2.ENTITY RELATIONSHIP DIAGRAM	13
	2.3 DATA FLOW DIAGRAM	14

3	MODULE DESCRIPTION	15
4	4.1 DESIGN	16
	4.2 DATABASE DESIGN	19
	4.3 IMPLEMENTATIONS (CODE)	21
5	CONCLUSION	30
6	REFERENCE	30

LIST OF FIGURES

Figure Numbers	Figure Captions	Pg.no
2.1.1	Use Case Diagram	9
2.2.1	Entity Relation Diagram	10
2.3.1	Data Flow Diagram	11
4.1.1	Login Page	13
4.1.2	Signup Page	13
4.1.3	Home Page	14
4.1.4	Issue Book	14
4.1.5	Manage Books	15
4.1.6	Return Book	15

LIST OF TABLES

Table Numbers	Table Caption	Pg.no
4.2.1	Database design	16
4.2.2	Database design	16
4.2.3	Books Table	17
4.2.4	Students Table	17

1.1 Abstract

The Library Management System (LMS) is a digital platform designed to streamline library operations, including book inventory management, student record maintenance, and tracking issued and returned books. It features a user-friendly dashboard displaying real-time statistics, such as the number of books, students, and defaulters. By automating library tasks, the LMS enhances efficiency, reduces manual errors, and ensures easy access to library resources, making it an essential tool for modern educational institutions.

1.2 Introduction

The Library Management System (LMS) is a comprehensive software solution developed to modernize and simplify the operations of a library. In educational institutions, efficient library management is crucial to ensure students and staff can access resources seamlessly.

This project addresses these challenges by offering a digital platform that automates key library operations, such as managing books, tracking issued and returned books, maintaining student records, and generating defaulter lists. The system's intuitive dashboard provides a snapshot of essential data, including the total number of books, students, and borrowed items, ensuring quick decision-making and streamlined workflows. By implementing this LMS, libraries can save time, enhance resource accessibility, and improve overall efficiency, making it an indispensable tool for modern library management.

1.3 Purpose

The Library Management System (LMS), built using JAVA NetBeans, MySQL, and phpMyAdmin, aims to provide a digital solution for managing library operations efficiently. It facilitates the automation of tasks such as book inventory management, student record maintenance, and book issuance and returns. The project leverages these technologies to ensure a user-friendly interface, robust data storage, and easy database management, reducing manual effort and enhancing operational accuracy and efficiency.

1.4 Scope

The Library Management System (LMS), built with NetBeans, MySQL, and phpMyAdmin, streamlines library operations by enabling efficient management of books, student records, and borrowing activities. It provides features such as book issuance and returns, inventory tracking, defaulter list generation, and a user-friendly dashboard for real-time insights. The system supports role-based access for administrators, librarians, and students, ensuring secure and organized library management.

1.5 Software Requirement Specification

Introduction:

The Software Requirement Specification (SRS) for the Library Management System (LMS) defines the system's functional and non-functional requirements, providing a clear blueprint for development. It ensures all stakeholders have a unified understanding of the system's goals, features, and constraints. Built with NetBeans, MySQL, and phpMyAdmin, the SRS focuses on automating library operations, improving efficiency, and delivering a user-friendly experience.

Document Purpose:

The purpose of this Software Requirement Specification (SRS) document is to define the requirements for developing the Library Management System (LMS). It provides a detailed description of the system's functionality, performance, and design constraints, ensuring clarity for developers, stakeholders, and users. This document serves as a guide throughout the development process, ensuring the project meets its objectives of automating library operations, improving efficiency, and delivering a seamless user experience.

Product Scope:

The Movie Reservation System is a GUI-based application developed using Java and MySQL, integrated with Visual Studio Code for development. The system enables users to log in, view available movies, select schedules, reserve tickets, make payments through mobile numbers, and receive payment confirmations. The system simplifies ticket booking processes, reduces manual intervention, and enhances user convenience.

References and Acknowledgments:

- <https://netbeans.apache.org/tutorial/main/kb/>
- <https://www.cdata.com/kb/tech/sql-jdbc-netbeans.rst>

Product Perspective:

The Library Management System (LMS) is a standalone application built with NetBeans, MySQL, and phpMyAdmin. It automates library tasks like book tracking, student management, and book issuance/returns, offering a scalable and secure solution for educational institutions and organizations.

Product Functionality:

Product Functionality (Library Management System - LMS)

- a) Admin Register: Allows the registration of new administrators to manage the system.
- b) Admin Login: Enables existing administrators to securely log in to the system.
- c) Add Book: Facilitates the addition of new book details, including title, author, and quantity.
- d) View Book: Allows librarians and administrators to view and update existing book information.
- e) Delete Book: Permits the removal of books that are no longer available or needed from the system.
- f) Issue Book: Supports the issuance of books to students, recording issue date and due date.
- g) Return Book: Allows students to return books and updates their borrowing records accordingly.
- h) Defaulter List: Generates a list of overdue books and defaulters, helping to track overdue items.
- i) Add Student: Enables the addition of new student records with relevant details like name, course, and branch.
- j) Update Student: Allows modification of student information, such as course changes or contact details.

Users and Characteristics:

- Qualification: Users should have at least basic educational qualifications, such as matriculation, and be comfortable with English.
- Experience: Familiarity with the university registration process is advantageous.
- Technical Experience: Users are expected to have elementary knowledge of computers for optimal system interaction.

Functional Requirements

1. Login Module (LM)
 - Users (admins) access the Login Module to authenticate.
 - LM supports user login with a username and password.
 - Passwords are masked for security.
 - Successful login requires verification by the database administrator.
2. Registered Users Module (RUM)
 - After a successful login, users (admins, librarians) can navigate through the application.
 - Users can view detailed information about books, students, and transactions.
 - Users can update and maintain book details, including modifying availability, issue status, and due dates.
3. Administrator Module (AM)
 - Upon successful login, the system displays administrative functions.
 - Functions include adding and updating book details, managing student records, and issuing/returning books.
 - The "Add" function allows administrators to input new book details and remove outdated entries.
 - The "Update" function enables administrators to modify existing book information in the database.
 - All add, update, or delete requests trigger communication with the Server Module (SM) for necessary database changes.
4. Server Module (SM)
 - SM acts as an intermediary between the frontend modules and the database.
 - It receives requests from various modules and formats pages for display.
 - SM validates and executes requests, ensuring proper communication with the database.
 - Handles communication with MySQL to ensure data consistency and integrity, especially regarding book details, student records, and transactions.

Non-Functional Requirements:

- Performance:
The system must efficiently handle real-time book issue and return requests, ensuring a response time of less than 2 seconds for record retrieval and updates.
- Scalability:
A robust security mechanism must be implemented on the server side to prevent unauthorized access to sensitive data, such as student records, transaction history, and book inventory.
- Security:
User privacy must be ensured by securely storing and managing personal details (e.g., names, contact information) and login credentials.
Data encryption should be applied to sensitive information, and passwords must be hashed for secure storage.
- Reliability:
The system is critical for managing library functions. In case of system failures or downtime, immediate actions must be taken to resolve issues and restore normal operations quickly.
- Usability:
Provide an intuitive and accessible user interface.

2.SYSTEM FLOW DIAGRAM

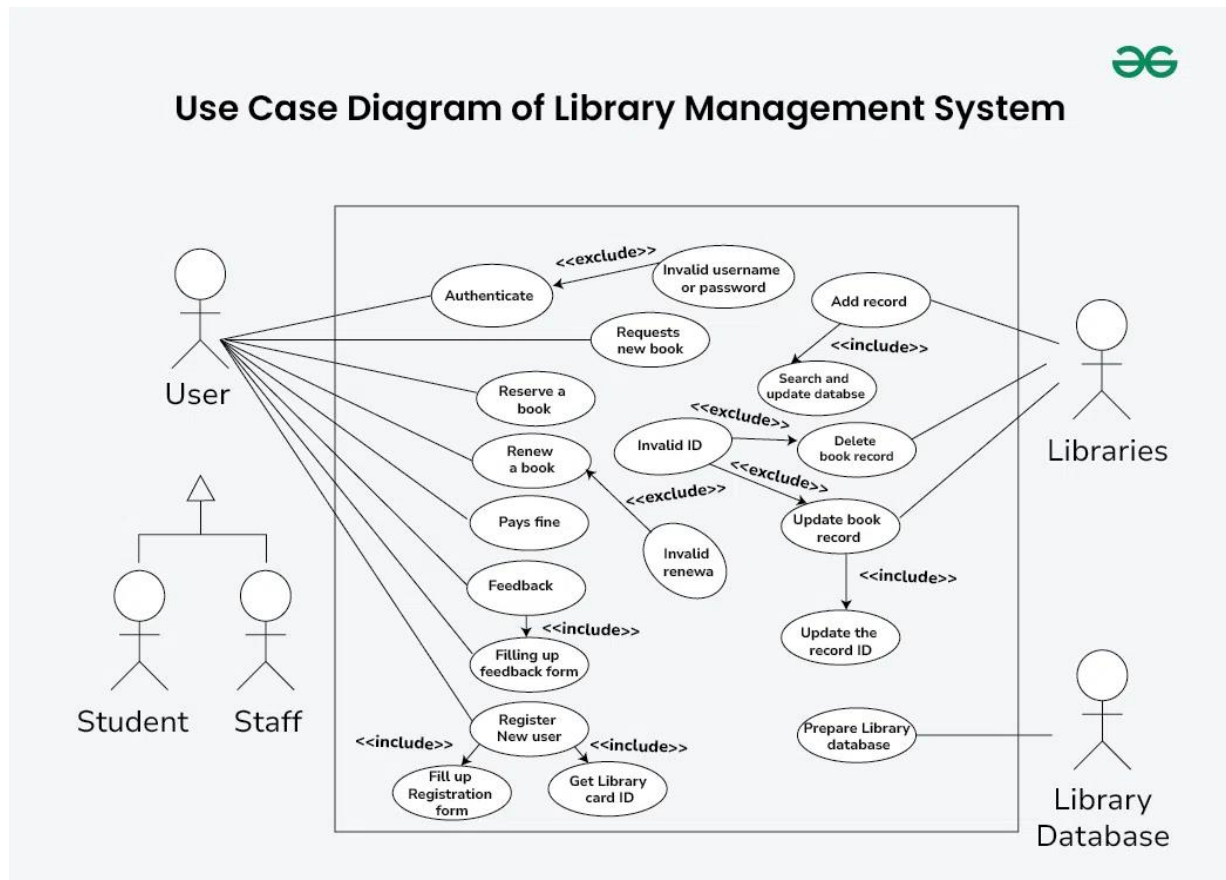


Figure 2.1.1 Use Case Diagram

E-RDiagram of Library Management System

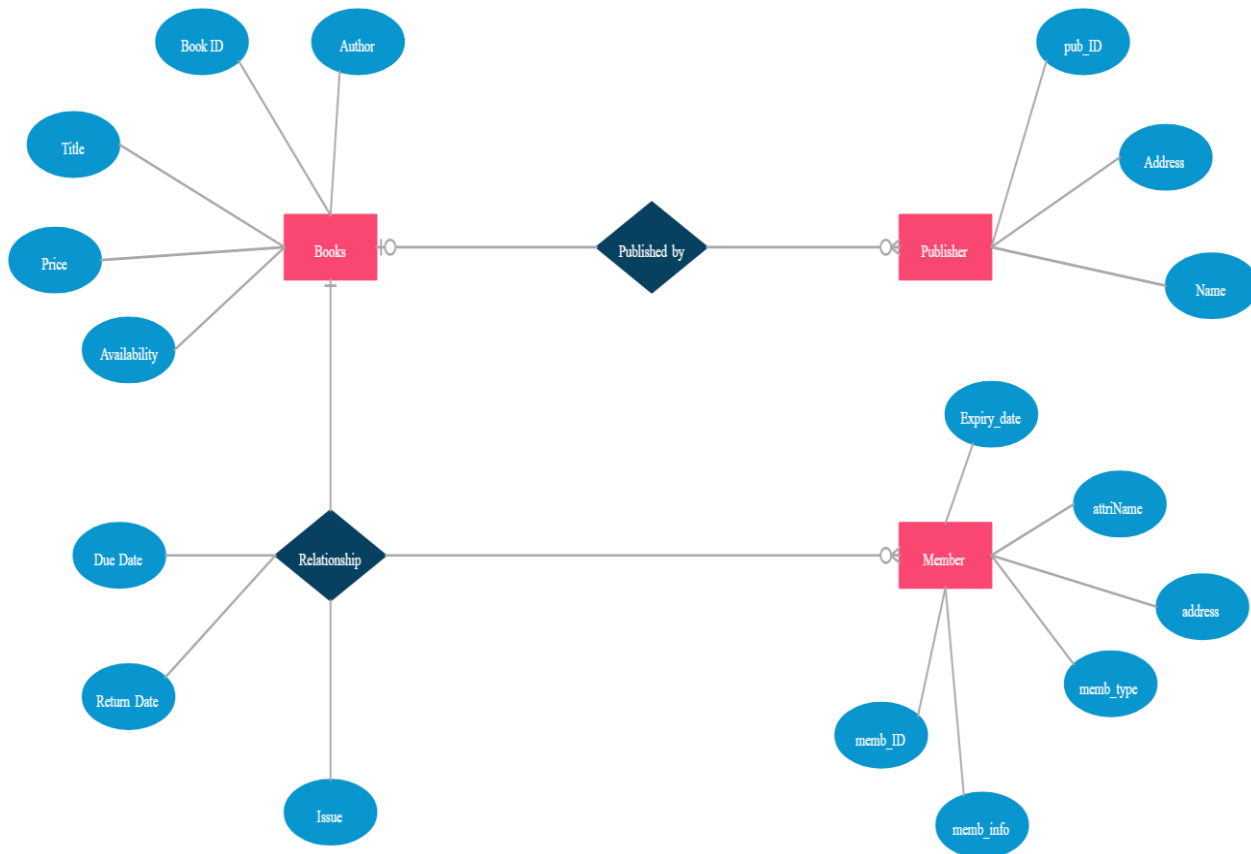


Figure 2.2.1 Entity Relation Diagram

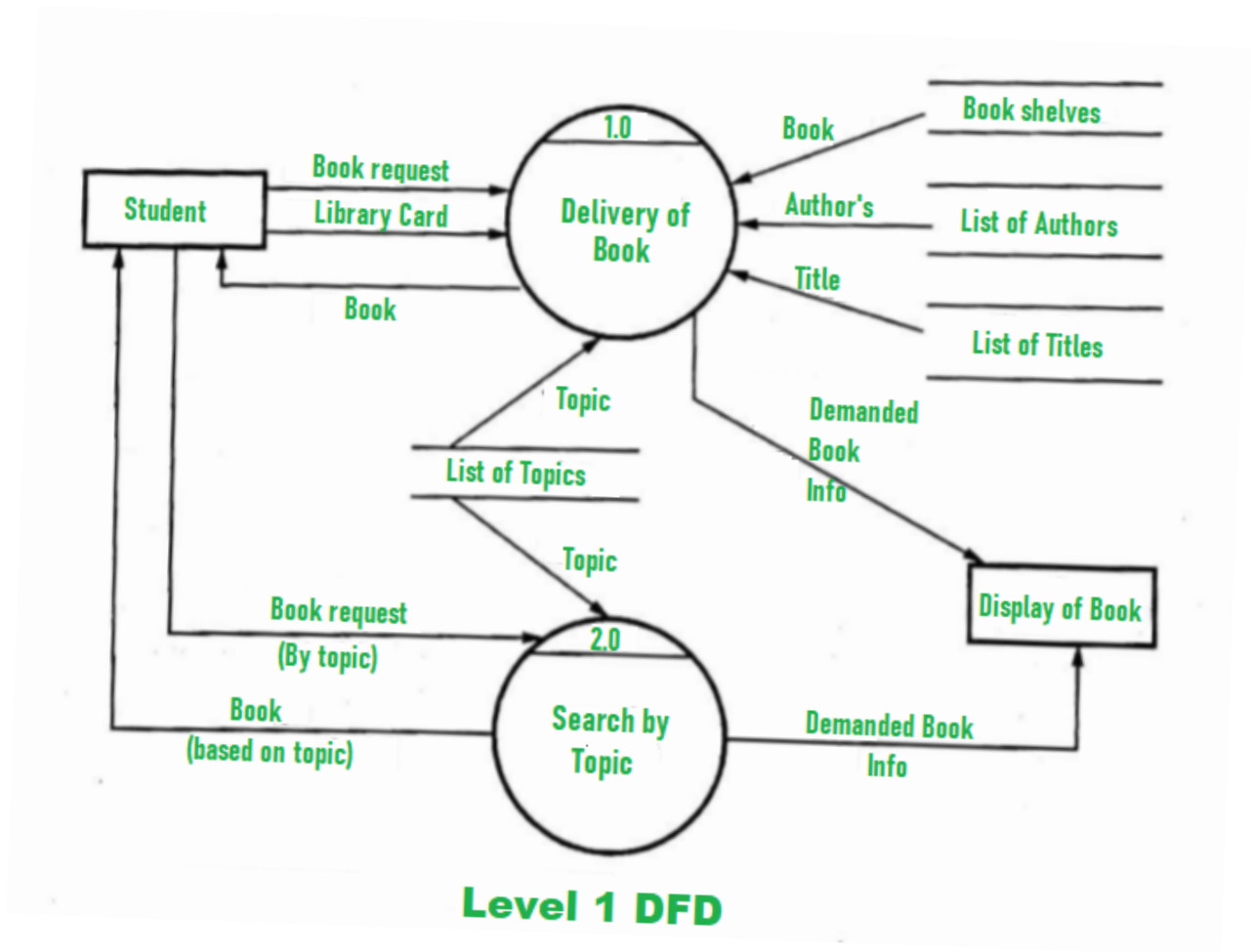


Figure 2.3.1 Data Flow Diagram

3.Module Description:

Admin Module

1. Register
 - Admin can register by providing a username and password to create an admin account in the system.
2. Login
 - Admin can log in using the username and password to access the system.
3. After Login
 - Add Book:
 - The admin can add new book details, including title, author, publisher, quantity, and category to the system.
 - View Book:
 - The admin can view and update existing book details such as title, author, availability, and issue status.
 - Delete Book:
 - The admin can remove books from the system by deleting their details if they are no longer needed or available.
 - Issue Book:
 - The admin can issue books to students, updating the book's availability and recording the transaction details (issue date, due date, etc.).
 - Return Book:
 - The admin can update the system when a book is returned, ensuring the return date is recorded and availability is updated.
 - Defaulter List:
 - The admin can view the defaulter list showing students with overdue books and take necessary actions, such as sending reminders or blocking future issues.

4.1 Design:

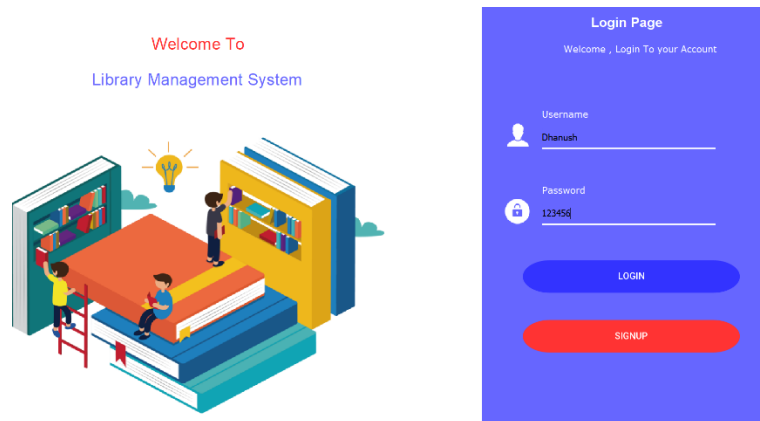


Figure 4.1.1 Login Page

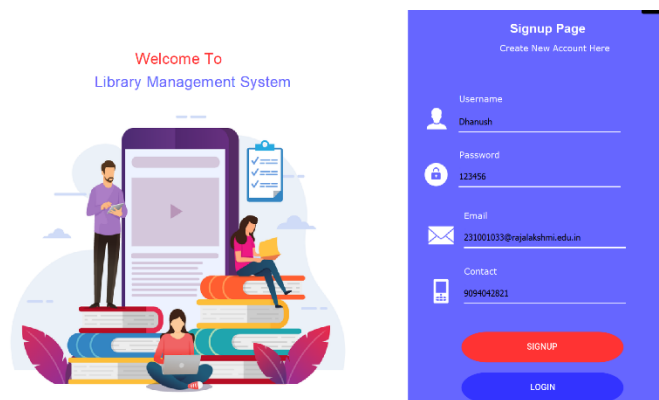


Figure 4.1.2 Signup Page

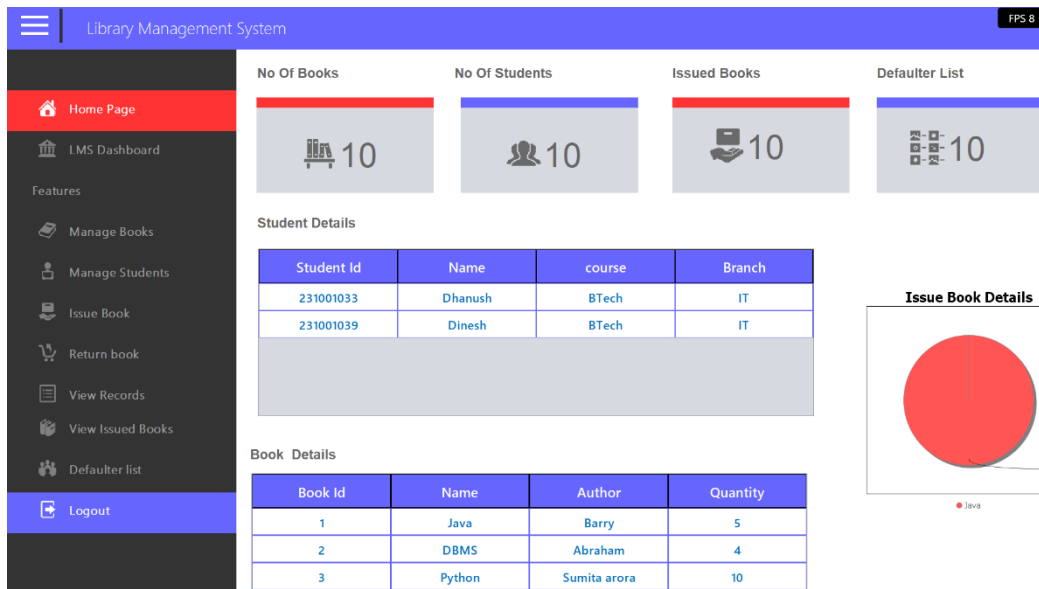


Figure 4.1.3 Home page

Back

Book Details

Book Id : 1

Book name : Java

Author : Barry

Quantity : 4

Student Details

Student Id : 231001033

Student name : Dhanush

Course : BTech

Branch : IT

Issue Book

Book Id : 1


Student Id : 231001033


Issue Date : 2024/11/19

Due Date : 2024/11/28

ISSUE BOOK


Figure 4.1.4 Issue Book

 Back




Enter Book Id

Enter Book Id ...




Enter Book Name :

Enter Book Name : ...



Author Name

Author Name ...



Quantity

Quantity ...

ADD

UPDATE

DELETE

Book Id	Name	Author	Quantity
1	Java	Barry	4
2	DBMS	Abraham	4
3	Python	Sumita arora	10
4	Psychology of money	leo adthi	20

Figure 4.1.5 Manage Books

Figure 4.1.6 Return Book

4.2 Database Design for Library Management System:

Database Design for Learning Management System (LMS)

The LMS database is designed in MySQL to store interrelated data efficiently with minimal redundancy. Normalization ensures data consistency, reduced storage needs, and optimized updates. Relationships between entities are defined using primary and foreign keys to enable quick and flexible data access.

Library Management System which contains 4 MySQL tables :

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> book_details	★ Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> issue_book_details	★ Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> student_details	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> users	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
4 tables	Sum	11	InnoDB	utf8mb4_general_ci	64.0 KiB	0 B

Table 4.2.1 Database design

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> book_details	★ Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> issue_book_details	★ Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> student_details	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> users	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
4 tables	Sum	11	InnoDB	utf8mb4_general_ci	64.0 KiB	0 B

Table 4.2.2 Database design














		book_id	book_name	author	quantity
<input type="checkbox"/>	 Edit  Copy  Delete	1	Java	Barry	5
<input type="checkbox"/>	 Edit  Copy  Delete	2	DBMS	Abraham	4
<input type="checkbox"/>	 Edit  Copy  Delete	3	Python	Sumita arora	10
<input type="checkbox"/>	 Edit  Copy  Delete	4	Psychology of money	leo adthi	20

Table 4.2.3 Books Table











		id	book_id	book_name	student_id	student_name	issue_date	due_date	status
<input type="checkbox"/>	 Edit  Copy  Delete	1	1	Java	231001033	Dhanush	2024-11-11	2024-11-25	returned
<input type="checkbox"/>	 Edit  Copy  Delete	2	1	Java	231001039	Dinesh	2024-11-11	2024-11-20	returned
<input type="checkbox"/>	 Edit  Copy  Delete	3	1	Java	231001033	Dhanush	2024-11-19	2024-11-28	returned

Table 4.2.4 Students Table

4.3 Implementations (CODE):

```
package jframe;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.swing.JOptionPane;
import static jframe.DBConnection.con;

/**
 *
 * @author sunil
 */
public class LoginPage extends javax.swing.JFrame {

    /**
     * Creates new form SignupPage
     */
    public LoginPage() {
        initComponents();
    }

    //validation
    public boolean validateLogin() {
        String name = txt_username.getText();
        String pwd = txt_password.getText();

        if (name.equals("")) {
            JOptionPane.showMessageDialog(this, "please enter username");
            return false;
        }
        if (pwd.equals("")) {
```

```

        JOptionPane.showMessageDialog(this, "please enter password");
        return false;
    }

    return true;
}

//verify creds
public void login() {
    String name = txt_username.getText();
    String pwd = txt_password.getText();

    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/library_ms", "root",
""");
        PreparedStatement pst = con.prepareStatement("select * from users where name = ? and
password = ?");

        pst.setString(1, name);
        pst.setString(2, pwd);

        ResultSet rs = pst.executeQuery();
        if (rs.next()) {
            JOptionPane.showMessageDialog(this, "login successful");
            HomePage home = new HomePage();
            home.setVisible(true);
            this.dispose();

        } else {
            JOptionPane.showMessageDialog(this, "incorrect username or password");
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

```

    }

}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jColorChooser1 = new javax.swing.JColorChooser();
    jPanel1 = new javax.swing.JPanel();
    jLabel5 = new javax.swing.JLabel();
    jLabel6 = new javax.swing.JLabel();
    jLabel2 = new javax.swing.JLabel();
    jPanel2 = new javax.swing.JPanel();
    jLabel4 = new javax.swing.JLabel();
    jLabel7 = new javax.swing.JLabel();
    jLabel8 = new javax.swing.JLabel();
    txt_username = new app.bolivia.swing.JTextField();
    jLabel9 = new javax.swing.JLabel();
    jLabel10 = new javax.swing.JLabel();
    jLabel11 = new javax.swing.JLabel();
    txt_password = new app.bolivia.swing.JTextField();
    rSMaterialButtonCircle1 = new necesario.RSMaterialButtonCircle();
    rSMaterialButtonCircle2 = new necesario.RSMaterialButtonCircle();
    jLabel16 = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    setUndecorated(true);

```

```

getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

jPanel1.setBackground(new java.awt.Color(255, 255, 255));
jPanel1.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

jLabel5.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/library-3.png.png"))); // NOI18N
jPanel1.add(jLabel5, new org.netbeans.lib.awtextra.AbsoluteConstraints(60, 210, 850, 620));

jLabel6.setFont(new java.awt.Font("Sitka Display", 0, 30)); // NOI18N
jLabel6.setForeground(new java.awt.Color(255, 51, 51));
jLabel6.setText("Welcome To");
jPanel1.add(jLabel6, new org.netbeans.lib.awtextra.AbsoluteConstraints(380, 90, 180, 30));

jLabel2.setFont(new java.awt.Font("Sitka Display", 0, 30)); // NOI18N
jLabel2.setForeground(new java.awt.Color(102, 102, 255));
jLabel2.setText("Library Management System");
jPanel1.add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(270, 150, 430, 40));

getContentPane().add(jPanel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(0, 0, 990, 830));

jPanel2.setBackground(new java.awt.Color(102, 102, 255));
jPanel2.setForeground(new java.awt.Color(51, 0, 153));
jPanel2.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

jLabel4.setFont(new java.awt.Font("Verdana", 0, 17)); // NOI18N
jLabel4.setForeground(new java.awt.Color(255, 255, 255));
jLabel4.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/icons/icons8_Account_50px.png"))); // NOI18N
jPanel2.add(jLabel4, new org.netbeans.lib.awtextra.AbsoluteConstraints(40, 250, 60, 50));

jLabel7.setFont(new java.awt.Font("Swis721 WGL4 BT", 1, 30)); // NOI18N
jLabel7.setForeground(new java.awt.Color(255, 255, 255));
jLabel7.setText("X");

```



```

jLabel7.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jLabel7MouseClicked(evt);
    }
});
jPanel2.add(jLabel7, new org.netbeans.lib.awtextra.AbsoluteConstraints(470, 10, 50, 30));

jLabel8.setFont(new java.awt.Font("Verdana", 0, 17)); // NOI18N
jLabel8.setForeground(new java.awt.Color(255, 255, 255));
jLabel8.setText("Welcome , Login To your Account");
jPanel2.add(jLabel8, new org.netbeans.lib.awtextra.AbsoluteConstraints(150, 100, 310, 30));

txt_username.setBackground(new java.awt.Color(102, 102, 255));
txt_username.setBorder(javax.swing.BorderFactory.createMatteBorder(0, 0, 2, 0, new
java.awt.Color(255, 255, 255)));
txt_username.setFont(new java.awt.Font("Tahoma", 0, 17)); // NOI18N
txt_username.setPlaceholder("Enter Username ....");
txt_username.addFocusListener(new java.awt.event.FocusAdapter() {
    public void focusLost(java.awt.event.FocusEvent evt) {
        txt_usernameFocusLost(evt);
    }
});
jPanel2.add(txt_username, new org.netbeans.lib.awtextra.AbsoluteConstraints(110, 260, 320, 40));

jLabel9.setFont(new java.awt.Font("Verdana", 0, 17)); // NOI18N
jLabel9.setForeground(new java.awt.Color(255, 255, 255));
jLabel9.setText("Username ");
jPanel2.add(jLabel9, new org.netbeans.lib.awtextra.AbsoluteConstraints(110, 220, 310, 30));

jLabel10.setFont(new java.awt.Font("Verdana", 0, 17)); // NOI18N
jLabel10.setForeground(new java.awt.Color(255, 255, 255));
jLabel10.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/icons/icons8_Secure_50px.png"))); // NOI18N
jPanel2.add(jLabel10, new org.netbeans.lib.awtextra.AbsoluteConstraints(40, 390, 60, 50));

```

```

jLabel11.setFont(new java.awt.Font("Verdana", 0, 17)); // NOI18N
jLabel11.setForeground(new java.awt.Color(255, 255, 255));
jLabel11.setText("Password");
jPanel2.add(jLabel11, new org.netbeans.lib.awtextra.AbsoluteConstraints(110, 360, 310, 30));

txt_password.setBackground(new java.awt.Color(102, 102, 255));
txt_password.setBorder(javax.swing.BorderFactory.createMatteBorder(0, 0, 2, 0, new
java.awt.Color(255, 255, 255)));
txt_password.setFont(new java.awt.Font("Tahoma", 0, 17)); // NOI18N
txt_password.setPlaceholder("Enter Password ...");
jPanel2.add(txt_password, new org.netbeans.lib.awtextra.AbsoluteConstraints(110, 400, 320, 40));

rSMaterialButtonCircle1.setBackground(new java.awt.Color(51, 51, 255));
rSMaterialButtonCircle1.setText("Login");
rSMaterialButtonCircle1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        rSMaterialButtonCircle1ActionPerformed(evt);
    }
});
jPanel2.add(rSMaterialButtonCircle1, new org.netbeans.lib.awtextra.AbsoluteConstraints(70, 500,
410, 70));

rSMaterialButtonCircle2.setBackground(new java.awt.Color(255, 51, 51));
rSMaterialButtonCircle2.setText("Signup");
rSMaterialButtonCircle2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        rSMaterialButtonCircle2ActionPerformed(evt);
    }
});
jPanel2.add(rSMaterialButtonCircle2, new org.netbeans.lib.awtextra.AbsoluteConstraints(70, 610,
410, 70));

jLabel16.setFont(new java.awt.Font("Swis721 LtEx BT", 1, 25)); // NOI18N

```

```

jLabel16.setForeground(new java.awt.Color(255, 255, 255));
jLabel16.setText("Login Page");
jPanel2.add(jLabel16, new org.netbeans.lib.awtextra.AbsoluteConstraints(200, 50, 180, 30));

getContentPane().add(jPanel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(990, 0, 540,
830));

setSize(new java.awt.Dimension(1523, 828));
setLocationRelativeTo(null);
} // </editor-fold>

private void rSMaterialButtonCircle2ActionPerformed(java.awt.event.ActionEvent evt) {

}

private void jLabel7MouseClicked(java.awt.event.MouseEvent evt) {
    System.exit(0);
}

private void txt_usernameFocusLost(java.awt.event.FocusEvent evt) {

}

private void rSMaterialButtonCircle1ActionPerformed(java.awt.event.ActionEvent evt) {
    if (validateLogin()) {
        login();
    }
}

/**
 * @param args the command line arguments

```

```

*/
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
    * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(LoginPage.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(LoginPage.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(LoginPage.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(LoginPage.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
    }
    //</editor-fold>
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new LoginPage().setVisible(true);

```

```

    }
    });
}

// Variables declaration - do not modify
private javax.swing.JColorChooser jColorChooser1;
private javax.swing.JLabel jLabel10;
private javax.swing.JLabel jLabel11;
private javax.swing.JLabel jLabel16;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JLabel jLabel9;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private necesario.RSMaterialButtonCircle rSMaterialButtonCircle1;
private necesario.RSMaterialButtonCircle rSMaterialButtonCircle2;
private app.bolivia.swing.JTextField txt_password;
private app.bolivia.swing.JTextField txt_username;
// End of variables declaration
}

```

5.Conclusion:

The Learning Management System (LMS) database design ensures efficient and reliable data storage and retrieval. By leveraging MySQL and applying normalization principles, the system minimizes redundancy, ensures data consistency, and optimizes performance. This robust design supports the LMS's objective of delivering a seamless user experience for managing and accessing educational resources.

6.REFERENCE :

- [1] <https://www.javatpoint.com/java-awt>
- [2] <https://www.javatpoint.com/java-swing>