**MINI PROJECT**

**Library Management System**

Course Code: 23ITI402

Course Title: Database Management System

Name: M.Sruthi

Roll No: 727623BIT011

Dept & Sec: IT-A

**Library Management System**

**Problem Statement:**

Managing books manually in small libraries can lead to inefficiencies such as

* + Misplaced records
  + Data redundancy
  + Lack of quick access to information

This can result in poor user experience and difficulty in maintaining accurate and up-to-date book inventories.

**Proposed Solution:**

To address the inefficiencies and limitations of manual book management in small libraries, we propose a **Library Management System** developed as a desktop application using **Java Swing** for the frontend and **MySQL** for the backend. This system simplifies the process of adding, viewing, updating, and deleting book records through a graphical user interface.

The application establishes a connection between the Java program and the MySQL database using **JDBC (Java Database Connectivity)**, ensuring smooth data transactions. Each book record includes an **auto-generated ID**, **title**, **author**, and **year of publication**.

This solution is scalable and can be further extended with additional functionalities like user login, book search, issue tracking, or reporting tools, making it suitable for real-time use in small libraries or personal collections.

**Key benefits:**

* Efficient Book Management
* User-Friendly Interface
* Fast and Accurate Data Handling
* Centralized Data Storage
* Easy Maintenance and Updates
* Improves Productivity

**Features:**

**1. Add Book**

* Takes input from the user (Title, Author, Year).
* Inserts the data into the books table in the MySQL database.
* Displays a confirmation dialog upon successful insertion.

**2. View Books**

* Fetches all records from the books table.
* Displays all book entries in a popup message using JOptionPane.

**3. Update Book**

* Prompts the user to enter the Book ID to update.
* Updates the book’s title, author, and year based on new input fields.
* Shows a success or failure message based on the result.

**4. Delete Book**

* Prompts the user for the Book ID to delete.
* Deletes the corresponding entry from the database.
* Displays confirmation or error messages.

**Technical Specification:**

**Technologies used:**

* **Frontend (UI):** Java Swing (AWT & Swing libraries)
* **Backend:** Java
* **Database:** MySQL
* **Database Connectivity:** JDBC (Java Database Connectivity)

**Database schema:**

**Table: books**

* **ID** (INT, PRIMARY KEY, AUTO\_INCREMENT)
* **Title** (VARCHAR)
* **Author** (VARCHAR)
* **Year** (INT)

**Implementation Details**

**Core Classes and Methods**

1. **Main Class: LibraryApp**

* **main(String[] args):** Launches the application window and sets up GUI components.
* **connectToDatabase():** Establishes connection to the MySQL database.

1. **Book Operations (CRUD)**

* **addBook(JFrame frame, String title, String author, int year):**Inserts a new book into the database based on user input.
* **viewBooks(JFrame frame):**  
  Retrieves and displays all book entries from the database using JOptionPane.
* **updateBook(JFrame frame, int id, String title, String author, int year):**  
  Updates a book's details based on the provided Book ID.
* **deleteBook(JFrame frame, int id):**  
  Deletes a book entry corresponding to the specified Book ID.

**GUI Components and Event Handling**

* **Java Swing components used:** JFrame, JPanel, JLabel, JTextField, JButton, JOptionPane.
* **Event handlers trigger database operations based on button clicks:**
* **addButton:** Calls addBook()
* **viewButton:** Calls viewBooks()
* **updateButton:** Prompts for Book ID and calls updateBook()
* **deleteButton:** Prompts for Book ID and calls deleteBook()

**Java Code:**

package com.mycompany.libraryapp;

import java.awt.\*;

import java.awt.event.\*;

import java.sql.\*;

import javax.swing.\*;

public class Libraryapp

{

// Connect to the MySQL database

private static Connection connectToDatabase()

{

Connection connection = null;

try

{

String url = "jdbc:mysql://localhost:3306/library\_db";

String user = "root";

String password = "Sruthi";

connection = DriverManager.getConnection(url, user, password);

System.out.println("Connected to the database successfully.");

}

catch (SQLException e)

{

System.out.println("Database connection failed: " + e.getMessage());

}

return connection;

}

// Add new book to the database

private static void addBook(JFrame frame, String title, String author, int year)

{

try (Connection connection = connectToDatabase();

PreparedStatement preparedStatement = connection.prepareStatement(

"INSERT INTO books (title, author, year) VALUES (?, ?, ?)")) {

preparedStatement.setString(1, title);

preparedStatement.setString(2, author);

preparedStatement.setInt(3, year);

preparedStatement.executeUpdate();

JOptionPane.showMessageDialog(frame, "Book added successfully!");

}

catch (SQLException e)

{

JOptionPane.showMessageDialog(frame, "Error adding book: " + e.getMessage());

}

}

// View all books in the database

private static void viewBooks(JFrame frame)

{

StringBuilder booksList = new StringBuilder();

try (Connection connection = connectToDatabase();

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery("SELECT \* FROM books")) {

while (resultSet.next())

{

int id = resultSet.getInt("id");

String title = resultSet.getString("title");

String author = resultSet.getString("author");

int year = resultSet.getInt("year");

booksList.append("ID: ").append(id)

.append(", Title: ").append(title)

.append(", Author: ").append(author)

.append(", Year: ").append(year)

.append("\n");

}

JOptionPane.showMessageDialog(frame, booksList.toString());

}

catch (SQLException e)

{

JOptionPane.showMessageDialog(frame, "Error retrieving books: " + e.getMessage());

}

}

// Update book details in the database

private static void updateBook(JFrame frame, int id, String title, String author, int year)

{

try (Connection connection = connectToDatabase();

PreparedStatement preparedStatement = connection.prepareStatement(

"UPDATE books SET title = ?, author = ?, year = ? WHERE id = ?")) {

preparedStatement.setString(1, title);

preparedStatement.setString(2, author);

preparedStatement.setInt(3, year);

preparedStatement.setInt(4, id);

int rowsUpdated = preparedStatement.executeUpdate();

if (rowsUpdated > 0)

{

JOptionPane.showMessageDialog(frame, "Book updated successfully!");

}

else

{

JOptionPane.showMessageDialog(frame, "Book not found.");

}

}

catch (SQLException e)

{

JOptionPane.showMessageDialog(frame, "Error updating book: " + e.getMessage());

}

}

// Delete book from the database

private static void deleteBook(JFrame frame, int id)

{

try (Connection connection = connectToDatabase();

PreparedStatement preparedStatement = connection.prepareStatement(

"DELETE FROM books WHERE id = ?"))

{

preparedStatement.setInt(1, id);

int rowsDeleted = preparedStatement.executeUpdate();

if (rowsDeleted > 0)

{

JOptionPane.showMessageDialog(frame, "Book deleted successfully!");

}

else

{

JOptionPane.showMessageDialog(frame, "Book not found.");

}

}

catch (SQLException e)

{

JOptionPane.showMessageDialog(frame, "Error deleting book: " + e.getMessage());

}

}

public static void main(String[] args)

{

// Create the main frame

JFrame frame = new JFrame("Library Management System");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(500, 400);

// Create the panel

JPanel panel = new JPanel(new GridLayout(6, 2));

JLabel titleLabel = new JLabel("Title:");

JTextField titleField = new JTextField(20);

JLabel authorLabel = new JLabel("Author:");

JTextField authorField = new JTextField(20);

JLabel yearLabel = new JLabel("Year:");

JTextField yearField = new JTextField(4);

JButton addButton = new JButton("Add Book");

JButton viewButton = new JButton("View Books");

JButton updateButton = new JButton("Update Book");

JButton deleteButton = new JButton("Delete Book");

panel.add(titleLabel);

panel.add(titleField);

panel.add(authorLabel);

panel.add(authorField);

panel.add(yearLabel);

panel.add(yearField);

panel.add(addButton);

panel.add(viewButton);

panel.add(updateButton);

panel.add(deleteButton);

frame.add(panel);

frame.setVisible(true);

// Button actions

addButton.addActionListener((ActionEvent e) -> {

String title = titleField.getText();

String author = authorField.getText();

int year = Integer.parseInt(yearField.getText());

addBook(frame, title, author, year);

});

viewButton.addActionListener((ActionEvent e) -> {

viewBooks(frame);

});

updateButton.addActionListener((ActionEvent e) -> {

try {

String idStr = JOptionPane.showInputDialog(frame, "Enter Book ID to Update:");

if (idStr == null || idStr.trim().isEmpty()) {

JOptionPane.showMessageDialog(frame, "Book ID is required.");

return;

}

int id = Integer.parseInt(idStr.trim());

String title = titleField.getText().trim();

String author = authorField.getText().trim();

String yearStr = yearField.getText().trim();

if (title.isEmpty() || author.isEmpty() || yearStr.isEmpty()) {

JOptionPane.showMessageDialog(frame, "Please fill in all the fields (Title, Author, Year).");

return;

}

int year = Integer.parseInt(yearStr);

updateBook(frame, id, title, author, year);

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(frame, "Please enter valid numbers for Book ID and Year.");

} catch (Exception ex) {

JOptionPane.showMessageDialog(frame, "An error occurred: " + ex.getMessage());

}

});

deleteButton.addActionListener((ActionEvent e) -> {

String idStr = JOptionPane.showInputDialog(frame, "Enter Book ID to Delete:");

int id = Integer.parseInt(idStr);

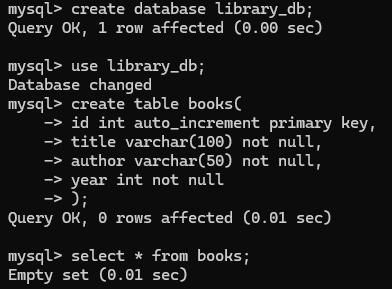
deleteBook(frame, id);

});

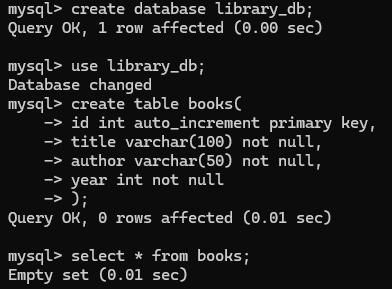
}

}

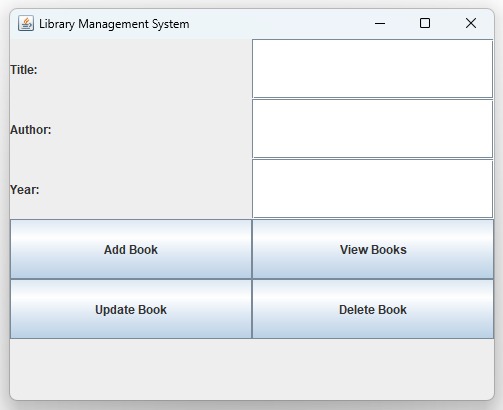
**Table Creation in MySQL:**



**Empty table after creation:**

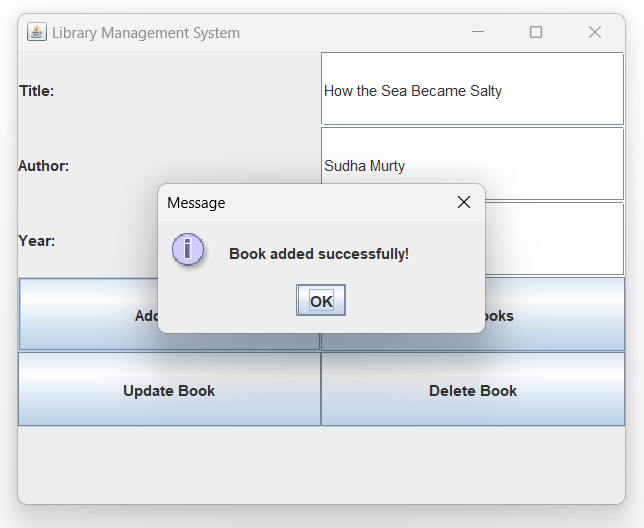


**Main GUI window:**



**Add book:**

**GUI window:**

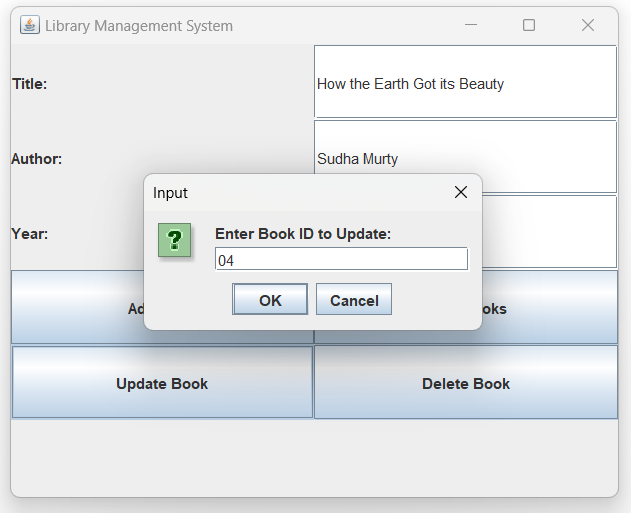
****

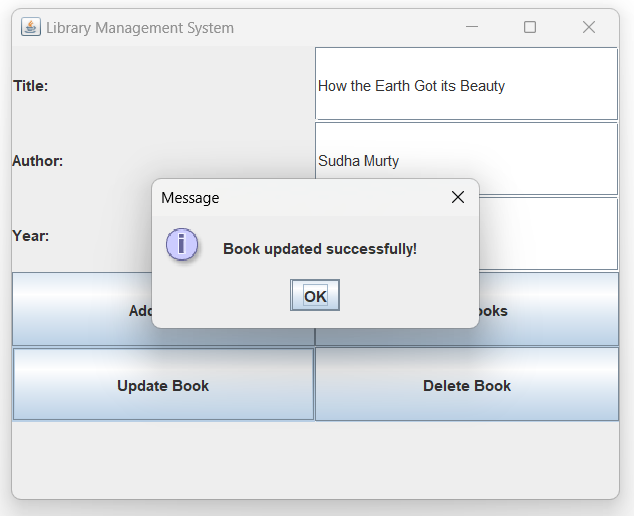
**SQL table:**

****

**Update book:**

**GUI window:**

****

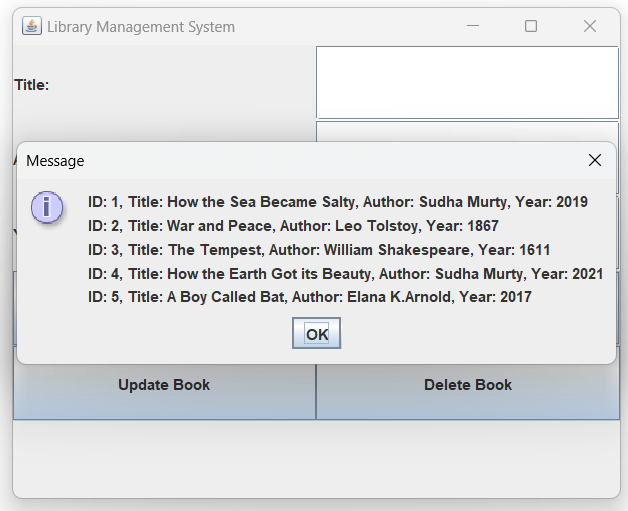
****

**SQL table:**



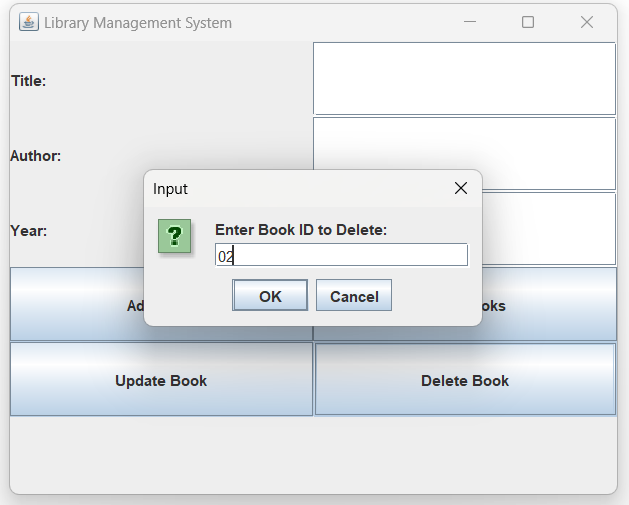
The year for the book id 4 changed from 2020 to 2021

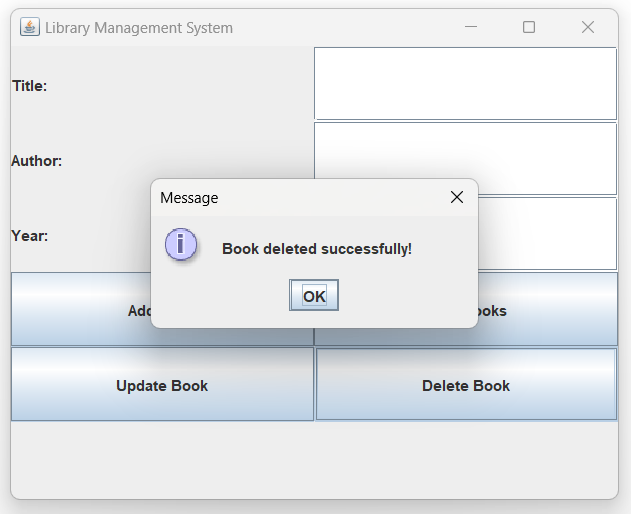
**View books:**

****

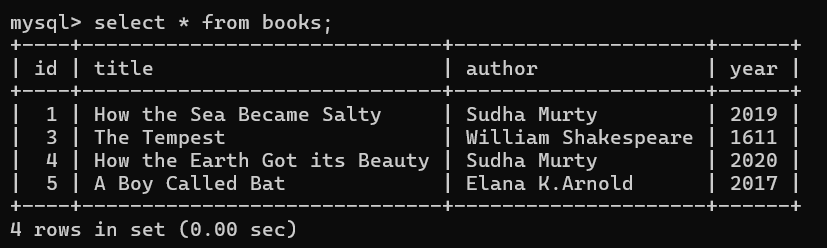
**Delete book:**

**GUI window:**





**SQL table:**

****

**Conclusion:**

The Library Management System project successfully demonstrates the practical application of Java Swing for GUI development, MySQL for data storage, and JDBC for database connectivity. The system offers a user-friendly interface to manage books efficiently, enabling operations such as adding, viewing, updating, and deleting book records. This project not only streamlines library operations but also provides hands-on experience in integrating front-end and back-end technologies. It serves as a solid foundation for future enhancements, such as adding user authentication, search filters, and book issuance modules, making it a scalable and extendable application.