**OOPS WITH JAVA**

“ LIBRARY MANAGEMENT SYSTEM USING JDBC CONNECTIVITY ”

**ROHINI S 231401085 CSBS ‘B’**

**Code – using JDBC and SQL :**

# SQL :

- - Users Table

CREATE TABLE Users (

UserID INT AUTO\_INCREMENT PRIMARY KEY, Name VARCHAR(100) NOT NULL,

Email VARCHAR(100) UNIQUE NOT NULL, Phone VARCHAR(15),

Role ENUM('Student', 'Faculty', 'Admin') NOT NULL, PasswordHash VARCHAR(255) NOT NULL

);

-- Books Table

CREATE TABLE Books (

BookID INT AUTO\_INCREMENT PRIMARY KEY, Title VARCHAR(200) NOT NULL,

Author VARCHAR(100) NOT NULL,

ISBN VARCHAR(13) UNIQUE NOT NULL,

Publisher VARCHAR(100), YearPublished YEAR, CopiesAvailable INT DEFAULT 0

);

-- Transactions Table

CREATE TABLE Transactions (

TransactionID INT AUTO\_INCREMENT PRIMARY KEY, UserID INT NOT NULL,

BookID INT NOT NULL,

IssueDate DATE NOT NULL, DueDate DATE NOT NULL,

ReturnDate DATE,

FineAmount DECIMAL(10, 2) DEFAULT 0.00,

Status ENUM('Issued', 'Returned') DEFAULT 'Issued', FOREIGN KEY (UserID) REFERENCES Users(UserID), FOREIGN KEY (BookID) REFERENCES Books(BookID)

);

-- Notifications Table

CREATE TABLE Notifications (

NotificationID INT AUTO\_INCREMENT PRIMARY KEY, UserID INT NOT NULL,

Message TEXT NOT NULL,

SentDate TIMESTAMP DEFAULT CURRENT\_TIMESTAMP, IsRead BOOLEAN DEFAULT FALSE,

FOREIGN KEY (UserID) REFERENCES Users(UserID)

);

-- Book Requests Table

CREATE TABLE BookRequests (

RequestID INT AUTO\_INCREMENT PRIMARY KEY, UserID INT NOT NULL,

BookID INT NOT NULL,

RequestDate TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

Status ENUM('Pending', 'Approved', 'Rejected') DEFAULT 'Pending', FOREIGN KEY (UserID) REFERENCES Users(UserID), FOREIGN KEY (BookID) REFERENCES Books(BookID)

);

# Java :

import javax.swing.\*; import java.awt.\*;

import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import java.sql.\*;

import java.util.ArrayList;

public class LibraryManagementSystem {

// Database connection details

private static final String DB\_URL = "jdbc:mysql://localhost:3306/library\_db"; // Replace with your database URL

private static final String DB\_USER = "your\_username"; // Replace with your database username

private static final String DB\_PASSWORD = "your\_password"; // Replace with your database password

public static void main(String[] args) { SwingUtilities.invokeLater(() -> {

JFrame frame = new JFrame("Smart Library Management System"); frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); frame.setSize(600, 400);

// Main Panel

JPanel panel = new JPanel(new BorderLayout()); frame.add(panel);

// Menu

String[] menuOptions = {"View Books", "Add Book"}; JComboBox<String> menu = new JComboBox<>(menuOptions); panel.add(menu, BorderLayout.NORTH);

// Content Panel

JPanel contentPanel = new JPanel(); panel.add(contentPanel, BorderLayout.CENTER);

menu.addActionListener(e -> {

String selectedOption = (String) menu.getSelectedItem(); contentPanel.removeAll();

if ("View Books".equals(selectedOption)) { showBooks(contentPanel);

} else if ("Add Book".equals(selectedOption)) { addBook(contentPanel);

}

contentPanel.revalidate(); contentPanel.repaint();

});

// Initial View showBooks(contentPanel);

frame.setVisible(true);

});

}

// Database Connection

private static Connection createConnection() { try {

return DriverManager.getConnection(DB\_URL, DB\_USER, DB\_PASSWORD);

} catch (SQLException e) {

JOptionPane.showMessageDialog(null, "Error connecting to database: " + e.getMessage());

return null;

}

}

// View Books

private static void showBooks(JPanel panel) { panel.setLayout(new BorderLayout());

JTextArea textArea = new JTextArea(); textArea.setEditable(false);

panel.add(new JScrollPane(textArea), BorderLayout.CENTER);

Connection connection = createConnection(); if (connection != null) {

try (Statement stmt = connection.createStatement()) {

ResultSet rs = stmt.executeQuery("SELECT \* FROM Books;"); StringBuilder books = new StringBuilder("Available Books:\n\n"); while (rs.next()) {

books.append("Title: ").append(rs.getString("Title"))

.append(" | Author: ").append(rs.getString("Author"))

.append(" | Copies Available: ").append(rs.getInt("CopiesAvailable"))

.append("\n");

}

textArea.setText(books.toString()); connection.close();

} catch (SQLException e) {

textArea.setText("Error fetching books: " + e.getMessage());

}

} else {

textArea.setText("No books found!");

}

}

// Add Book

private static void addBook(JPanel panel) { panel.setLayout(new GridLayout(7, 2));

JTextField titleField = new JTextField(); JTextField authorField = new JTextField(); JTextField isbnField = new JTextField(); JTextField publisherField = new JTextField(); JTextField yearField = new JTextField(); JTextField copiesField = new JTextField();

panel.add(new JLabel("Book Title:")); panel.add(titleField);

panel.add(new JLabel("Author:")); panel.add(authorField);

panel.add(new JLabel("ISBN:")); panel.add(isbnField);

panel.add(new JLabel("Publisher:")); panel.add(publisherField);

panel.add(new JLabel("Year Published:")); panel.add(yearField);

panel.add(new JLabel("Copies Available:")); panel.add(copiesField);

JButton addButton = new JButton("Add Book"); panel.add(new JLabel());

panel.add(addButton);

addButton.addActionListener(new ActionListener() { @Override

public void actionPerformed(ActionEvent e) { String title = titleField.getText();

String author = authorField.getText(); String isbn = isbnField.getText();

String publisher = publisherField.getText();

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

int copies = Integer.parseInt(copiesField.getText());

Connection connection = createConnection(); if (connection != null) {

try (PreparedStatement pstmt = connection.prepareStatement(

"INSERT INTO Books (Title, Author, ISBN, Publisher, YearPublished, CopiesAvailable) " +

"VALUES (?, ?, ?, ?, ?, ?)")) {

pstmt.setString(1, title); pstmt.setString(2, author); pstmt.setString(3, isbn); pstmt.setString(4, publisher); pstmt.setInt(5, year); pstmt.setInt(6, copies);

pstmt.executeUpdate();

JOptionPane.showMessageDialog(null, "Book added successfully!"); connection.close();

} catch (SQLException ex) {

JOptionPane.showMessageDialog(null, "Error adding book: " + ex.getMessage());

}

}

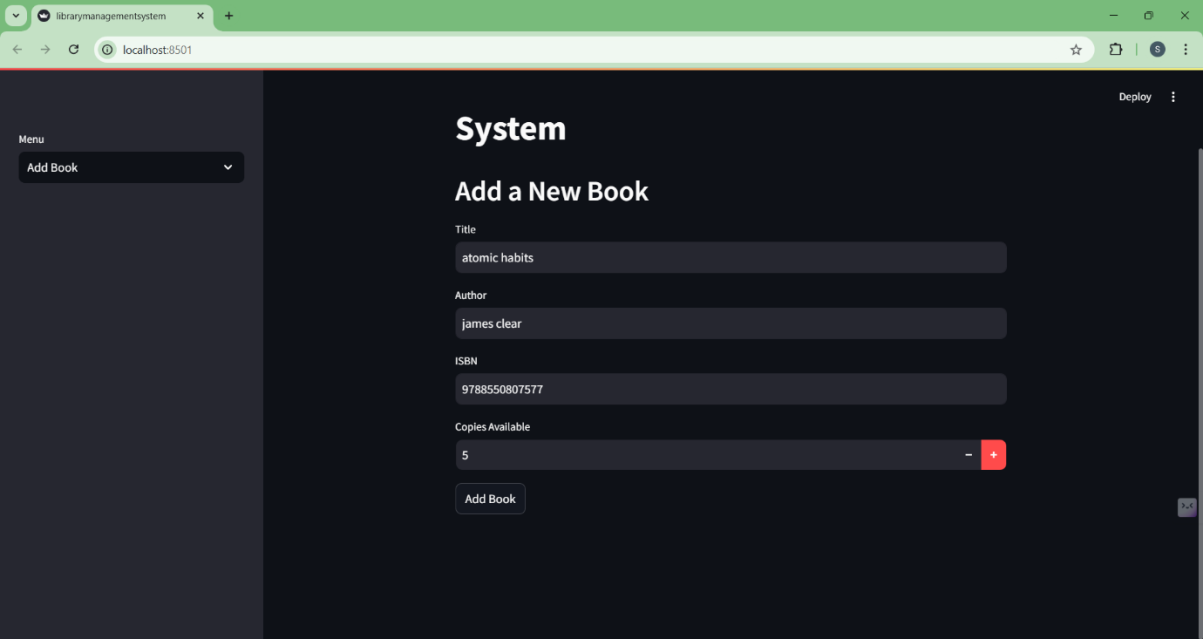
}

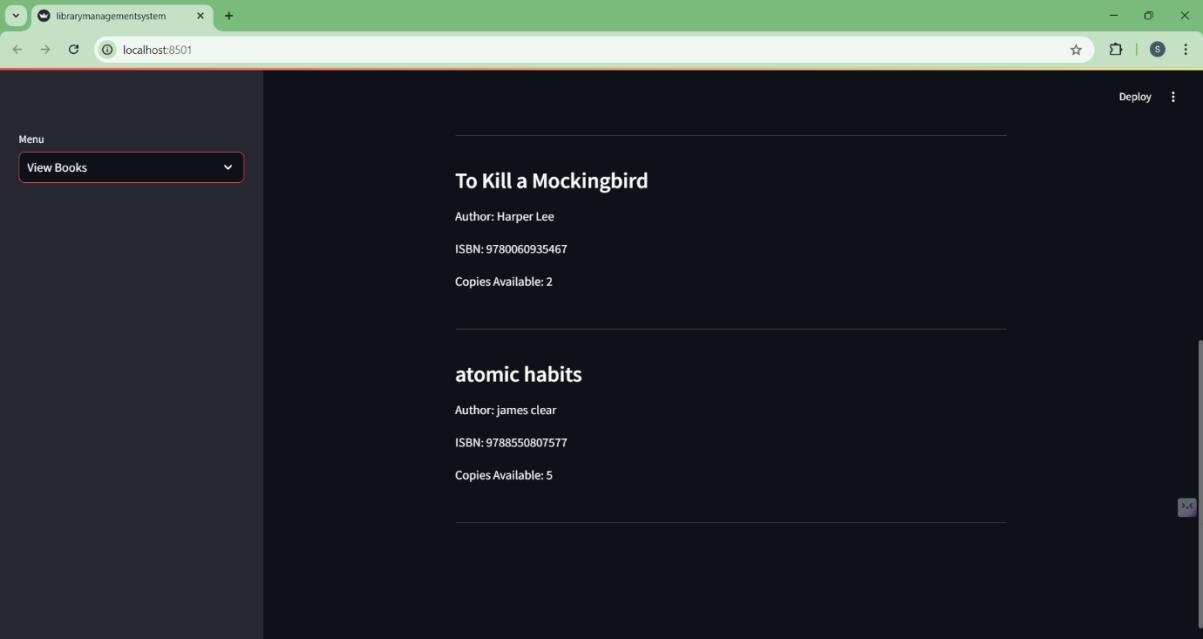
});

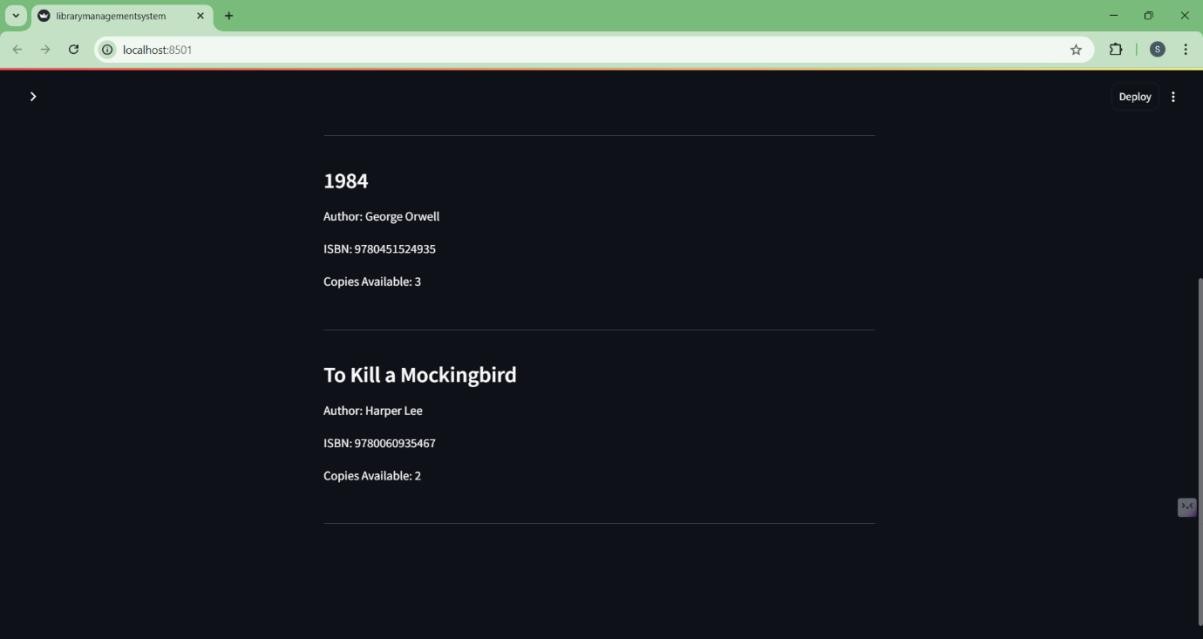
}

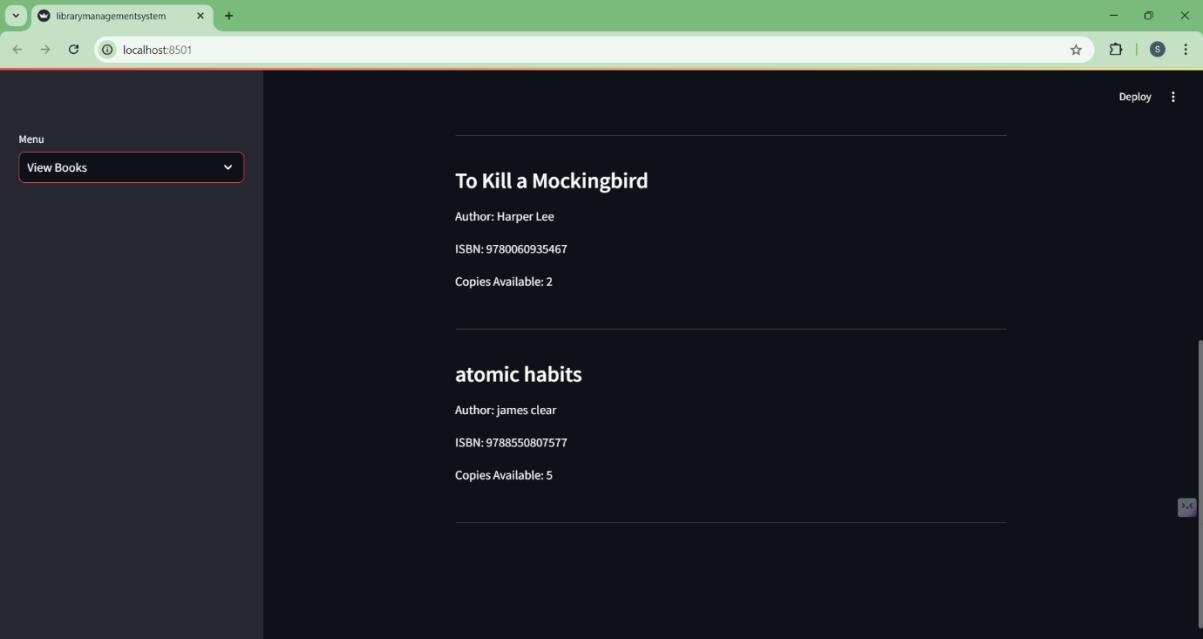
}

# Project output screenshots :









**CONCLUSION:**

The Library Management System streamlines library operations by providing a robust database design that supports key functionalities such as book borrowing, user management, and fine calculation. The implementation ensures data consistency, reduces manual workload, and enhances user satisfaction. By integrating relational database technology, the system can handle a large volume of data efficiently and adapt to future requirements.