MINI PROJECT

LIBRARY MANAGEMENT SYSTEM

AIM:

The aim of this project is to design a simple Library Management System using Java and MySQL to efficiently manage and display book records in a database.

ALGORITHM:

1) Start.
2) Establish a connection to the MySQL database using JDBC.
3) Show the main menu with options to add a book, display books, or exit.
4) Prompt the user for book details (title, author, genre, and price).
5) Insert the entered details into the Books table in the database.
6) Retrieve all records from the Books table.
7) Format and display the book details (ID, title, author, genre, price).
8) Loop back to the main menu until the user chooses to exit.
9) Based on user input, call the corresponding function (add or display books).
10) Close the database connection and terminate the program.
11) Stop.

PROGRAM:

SQL CODE:

```
CREATE DATABASE LibraryDB;



USE LibraryDB;



CREATE TABLE Books (

   id INT AUTO_INCREMENT PRIMARY KEY,

   title VARCHAR(100) NOT NULL,

   author VARCHAR(100) NOT NULL,

   genre VARCHAR(50),

   price DECIMAL(10, 2)

);
```

JAVA CODE:

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

public class LibraryManagementSystem {
    static final String DB_URL = "jdbc:mysql://localhost:3306/LibraryDB";
    static final String USER = "root"; // Update with your MySQL username
    static final String PASS = "password"; // Update with your MySQL password

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        try (Connection conn = DriverManager.getConnection(DB_URL, USER, PASS)) {
            System.out.println("Connected to the database.");

            while (true) {
                System.out.println("\nLibrary Management System:");
                System.out.println("1. Add Book");
                System.out.println("2. Display Books");
                System.out.println("3. Exit");
                System.out.print("Enter your choice: ");
                int choice = scanner.nextInt();
                scanner.nextLine(); // Consume newline

                switch (choice) {
                    case 1:
```

```
                addBook(conn, scanner);
                break;
            case 2:
                displayBooks(conn);
                break;
            case 3:
                System.out.println("Exiting... Goodbye!");
                return;
            default:
                System.out.println("Invalid choice. Please try again.");
        }
      }
    } catch (SQLException e) {
      System.out.println("Database error: " + e.getMessage());
    }
}

private static void addBook(Connection conn, Scanner scanner) {
    try {
      System.out.print("Enter book title: ");
      String title = scanner.nextLine();

      System.out.print("Enter book author: ");
      String author = scanner.nextLine();

      System.out.print("Enter book genre: ");
```

```
            String genre = scanner.nextLine();

            System.out.print("Enter book price: ");
            double price = scanner.nextDouble();

            String query = "INSERT INTO Books (title, author, genre, price) VALUES (?, ?, ?, ?)";
            try (PreparedStatement pstmt = conn.prepareStatement(query)) {
                pstmt.setString(1, title);
                pstmt.setString(2, author);
                pstmt.setString(3, genre);
                pstmt.setDouble(4, price);
                pstmt.executeUpdate();
                System.out.println("Book added successfully.");
            }
        } catch (SQLException e) {
            System.out.println("Error adding book: " + e.getMessage());
        }
    }

    private static void displayBooks(Connection conn) {
        String query = "SELECT * FROM Books";
        try (Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            System.out.println("\nBooks in the Library:");
            System.out.printf("%-5s %-20s %-20s %-15s %-10s\n", "ID", "Title", "Author", "Genre",
"Price");
```

```
            System.out.println("----------------------------------------------------------------");
            while (rs.next()) {
                System.out.printf("%-5d %-20s %-20s %-15s %-10.2f\n",
                        rs.getInt("id"),
                        rs.getString("title"),
                        rs.getString("author"),
                        rs.getString("genre"),
                        rs.getDouble("price"));
            }
        } catch (SQLException e) {
            System.out.println("Error displaying books: " + e.getMessage());
        }
    }
}
```

OUTPUT:

```
Connected to the database.

Library Management System:
1. Add Book
2. Display Books
3. Exit
Enter your choice: 1

Enter book title: Clean Code
Enter book author: Robert C. Martin
Enter book genre: Programming
Enter book price: 40.50
Book added successfully.
```

```
Library Management System:
1. Add Book
2. Display Books
3. Exit
Enter your choice: 1

Enter book title: Atomic Habits
Enter book author: James Clear
Enter book genre: Self-Help
Enter book price: 25.00
Book added successfully.

Library Management System:
1. Add Book
2. Display Books
3. Exit
Enter your choice: 2

Books in the Library:
ID    Title              Author            Genre         Price
-----------------------------------------------------------------
1     Clean Code         Robert C. Martin  Programming   40.50
2     Atomic Habits      James Clear       Self-Help     25.00
```

RESULT:

The Library Management System efficiently manages and displays book records using Java and MySQL, meeting the aim of creating a simple and functional tool.