CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

UNIVERSITY INSTITUTE OF COMPUTING

PROJECT REPORT

ON

Library management system

Program Name: BCA

Subject Name/Code: Java programming lab

(22CAP-352)

Submitted by:

Name: Robinpreet Singh
UID: 22BCA10966
Section: 22BCA -9(A)

Submitted to:

Name: Mr.Rishabh
Designation: Asst. Prof.

ABSTRACT

The Library Management System (LMS) is a software application designed to streamline the management of books and user interactions within a library environment. The primary objective of this project is to create an intuitive and efficient system that allows librarians and users to manage book inventories, track user information, and perform essential operations such as adding, updating, retrieving, and deleting book records.

The application is developed using Java, leveraging the AWT/Swing framework for the graphical user interface (GUI) to provide a user-friendly experience. The GUI is designed to facilitate easy navigation and interaction, allowing users to input book details, view existing records, and manage the library's collection seamlessly.

Introduction

In today's fast-paced world, libraries play a crucial role in providing access to information and knowledge. However, managing a library's collection of books and user interactions can be a complex and timeconsuming task. Traditional manual methods of tracking books and users often lead to inefficiencies, errors, and difficulties in retrieving information. To address these challenges, the Library Management System (LMS) project aims to develop a comprehensive software solution that automates and streamlines library operations.

Technique

The Library Management System (LMS) project employs a variety of techniques and technologies to ensure efficient functionality, user-friendly interaction, and robust data management. Below are the key techniques utilized in the

Development of this project:

1. Object-Oriented Programming (OOP):

- The project is designed using OOP principles, which promote code reusability, modularity, and maintainability. Classes such as Book, BookDAO, and DatabaseConnection encapsulate related data and behaviors, making the codebase easier to manage and extend.

2. Graphical User Interface (GUI) Design:

- The application uses Java's AWT (Abstract Window Toolkit) and Swing libraries to create a graphical user interface. This approach allows for the development of a visually appealing and interactive interface that enhances user experience.

Components such as buttons, text fields, and text areas are utilized to facilitate user input and display information.

3. Database Management:

- The project employs MySQL as the relational database management system (RDBMS) to store and manage book and user data. The use of SQL (Structured Query Language) enables efficient data manipulation and retrieval through well-defined queries.

4. Java Database Connectivity (JDBC):

- JDBC is used to establish a connection between the Java application and the MySQL database. This technique allows for executing SQL statements and managing database transactions directly from the Java code, enabling seamless integration of database operations within the application.

5. CRUD Operations:

- The project implements CRUD (Create, Read, Update, Delete) operations to manage book records effectively. Each operation is encapsulated within methods in the BookDAO class, allowing for clear separation of concerns and easy maintenance.

Code:-

```
import java.sql.Connection; import
java.sql.DriverManager; import
java.sql.SQLException;

public class DatabaseConnection {     private static final String URL =
"jdbc:mysql://localhost:3306/library_db";     private static final String USER = "root";
// Change as per your MySQL setup     private static final String PASSWORD =
"password"; // Change as per your MySQL setup

   public static Connection getConnection() {
Connection connection = null;        try {
          connection = DriverManager.getConnection(URL, USER, PASSWORD);
} catch (SQLException e) {           e.printStackTrace();
      }
      return connection;
   }
}
2. Book Class
java Run
Copy code public class Book
{     private int id;     private
String title;     private String
author;     private int
publishedYear;

   public Book(int id, String title, String author, int publishedYear) {
      this.id = id;        this.title = title;
this.author = author;
this.publishedYear = publishedYear;
   }

   // Getters and Setters     public int getId() { return id; }     public void setId(int id) { this.id
= id; }     public String getTitle() { return title; }     public void setTitle(String title) { this.title
= title; }     public String getAuthor() { return author; }     public void setAuthor(String
author) { this.author = author; }     public int getPublishedYear() { return publishedYear; }
public void setPublishedYear(int publishedYear) { this.publishedYear = publishedYear; }
}
```

3. BookDAO Class
java Run
Copy code

```
import java.sql.*; import
java.util.ArrayList;
import java.util.List;

public class BookDAO {
   public void addBook(Book book) {
      String query = "INSERT INTO books (title, author, published_year) VALUES (?, ?, ?)";
try (Connection connection = DatabaseConnection.getConnection();
PreparedStatement statement = connection.prepareStatement(query)) {
statement.setString(1, book.getTitle());          statement.setString(2, book.getAuthor());
statement.setInt(3, book.getPublishedYear());          statement.executeUpdate();
      } catch (SQLException e) {
         e.printStackTrace();
      }
   }

   public List<Book> getAllBooks() {
      List<Book> books = new ArrayList<>();
String query = "SELECT * FROM books";
      try (Connection connection = DatabaseConnection.getConnection();
          Statement statement = connection.createStatement();
ResultSet resultSet = statement.executeQuery(query)) {          while
(resultSet.next()) {
             Book book = new Book(resultSet.getInt("id"),
resultSet.getString("title"),                 resultSet.getString("author"),
resultSet.getInt("published_year"));             books.add(book);
          }
      } catch (SQLException e) {
e.printStackTrace();
      }
      return books;
   }

   public void updateBook(Book book) {
      String query = "UPDATE books SET title = ?, author = ?, published_year = ? WHERE id =
?";
```

```
    try (Connection connection = DatabaseConnection.getConnection();
PreparedStatement statement = connection.prepareStatement(query)) {
statement.setString(1, book.getTitle());          statement.setString(2,
book.getAuthor());          statement.setInt(3, book.getPublishedYear());
statement.setInt(4, book.getId());          statement.executeUpdate();       } catch
(SQLException e) {          e.printStackTrace();
    }
  }

  public void deleteBook(int id) {
    String query = "DELETE FROM books WHERE id = ?";
    try (Connection connection = DatabaseConnection.getConnection();
PreparedStatement statement = connection.prepareStatement(query)) {
statement.setInt(1, id);          statement.executeUpdate();       } catch
(SQLException e) {          e.printStackTrace();
    }
  }
}
```

4. Library Management System GUI

java Run

Copy code import javax.swing.*;
import java.awt.*; import
java.awt.event.ActionEvent; import
java.awt.event.ActionListener; import
java.util.List;

```
public class LibraryManagementSystem extends JFrame {
private JTextField titleField, authorField, yearField;
private JTextArea displayArea;    private BookDAO
bookDAO;

  public LibraryManagementSystem() {          bookDAO =
new BookDAO();              setTitle("Library Management
System");                   setSize(400,    400);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(new FlowLayout());

    titleField = new JTextField(20);
authorField = new JTextField(20);       yearField
= new JTextField(4);       displayArea = new
```

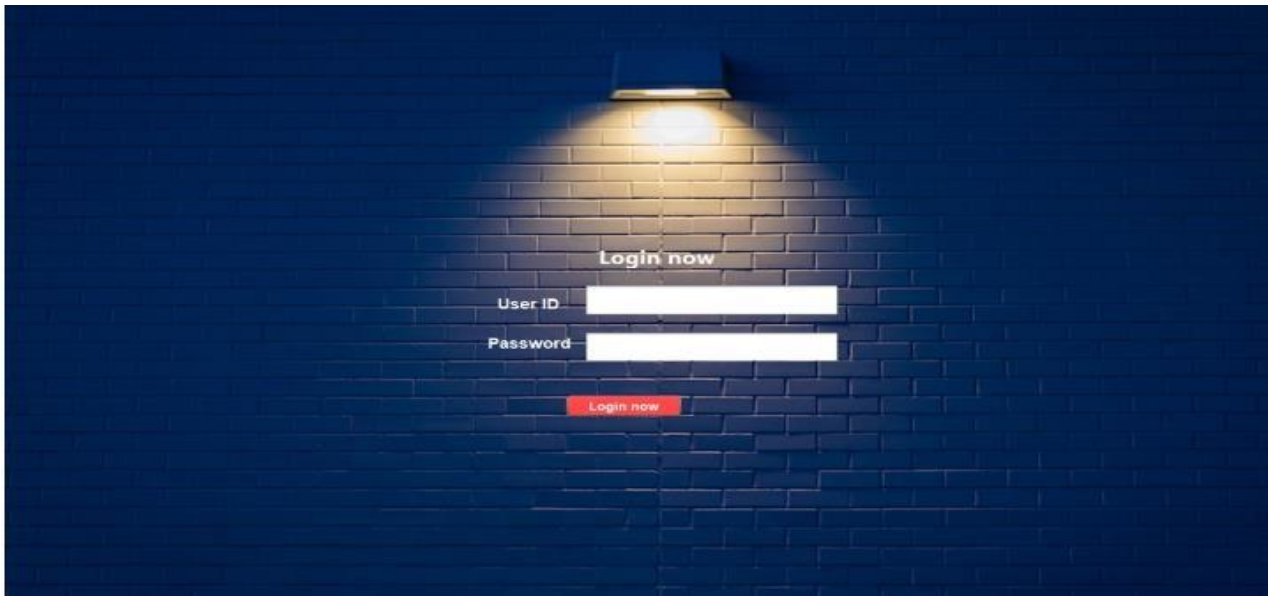
CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Visual Representation Of Project

A library management system is software that is designed to manage all the functions of a library. It helps librarian to maintain the database of new books and the books that are borrowed by members along with their due dates.

First Outcome:- You can see The dashboard like this photo.

2nd Outcome:-In next step you can see the login page, This will help you to login .



3rd Outcome:-In third outcome u can see the Book details page,You have to fill all the Column .And then you can access the book details.

4th Outcome:-You can see the Student or User Registration page,Again you have to fill all the details and then you can easily use the Book.

Last Outcome:-

Result

The Library Management System (LMS) project, upon successful implementation, provides a functional application that allows users to manage books in a library efficiently. Below are the key results and outcomes of the project:

1. User -Friendly Interface:

- The application features a well-designed graphical user interface (GUI) that is intuitive and easy to navigate. Users can quickly access various functionalities such as adding, updating, deleting, and viewing books.

2.CRUD Functionality:

- The system successfully implements CRUD (Create, Read, Update, Delete) operations:

- Create: Users can add new books to the library by entering details such as title, author, and published year.

- Read: Users can view a list of all books in the library, which is displayed in a text area.

- Update: Users can modify the details of existing books by selecting a book and updating its information.

- Delete: Users can remove books from the library by specifying the book ID.

3. Database Integration:

- The application is connected to a MySQL database, allowing for persistent storage of book records. This ensures that data is retained even after the application is closed, and users can retrieve it at any time.

4. Data Validation:

- Basic data validation is implemented to ensure that user inputs are accurate. For example, the application can check for empty fields or invalid data types before performing database operations.

5. **Error Handling:**

- The application includes exception handling mechanisms to manage potential errors during database operations. This ensures that the application can handle unexpected situations gracefully, providing feedback to users without crashing.

6. **Modular Design:**

- The project is structured in a modular fashion, with separate classes for database connection, book management, and GUI. This modularity enhances code readability and maintainability, making it easier to update or extend the application in the future.

7. **Demonstration of Software Development Skills:**

- The project serves as a practical demonstration of various software development skills, including object-oriented programming, GUI design, database management, and event-driven programming.

8. Potential for Future Enhancements:

- The foundation laid by this project allows for future enhancements, such as:
- User authentication and role management (e.g., librarian vs. user).
- Advanced search functionality (e.g., searching by genre, ISBN).
- Integration with online book databases or APIs.
- Reporting features to track borrowing history and popular books.

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

SUMMARY

The Library Management System project successfully demonstrates the integration of GUI design with database interaction, providing a practical tool for library management. It enhances the efficiency of library operations, making it easier for librarians to manage their collections and for users to access information about available books. The project serves as a valuable learning experience in software development, showcasing skills in object-oriented programming, database management, and user interface design.

Future Enhancements

The foundation of the LMS allows for potential future enhancements, such as user authentication, advanced search capabilities, integration with online book databases, and reporting features to track borrowing history and popular books. In summary, the Library Management System project is a comprehensive solution that addresses the challenges of managing library operations, ultimately contributing to a better experience for both librarians and library patrons.