



GUVI Project Report

Department:- School of Computer Science and
Engineering

Project Title :- *Library Management System*

Section :- 49

***Created By : Arpit Sahu , Dharmendra Pratap
Singh, Dipongkor Roy (Joy), Angela Sarekar***

LibraryManagementSystem

Library Management System: Take Control of Your Book Collection This Java code provides a foundational framework for a simple library management system. It allows users to:

Create a library: Manage a collection of books with titles, authors, and availability status. Add books: Easily add new books to your library with their details. Borrow books: Simulate the borrowing process, marking books as unavailable. Return books: Track returned books, updating their availability status. List books: View a list of all books in the library, including their availability. This code serves as a starting point for building a more comprehensive library management system. Here are some potential enhancements:

Member Management: Include functionality for adding and managing members, allowing for tracking who borrows which books. Due Dates: Implement a system for assigning due dates to borrowed books and generating overdue notices. Search Functionality: Allow users to search for books by title, author, or keywords. Persistence: Save and load library data to a file or database for persistent storage. Graphical User Interface (GUI): Develop a user-friendly GUI application for easier interaction with the library system. This code demonstrates the basic principles of object-oriented programming and data structures (using ArrayLists) to manage a library collection. It's a valuable tool for practicing these concepts and provides a foundation for building a more robust library management system.

Introduction:-

What is a Library Management System?

A Library Management System is software designed to automate and manage a library's activities such as maintaining book records, issuing and returning books, and managing student records.

Real-life Use:

It reduces paperwork, prevents human error, and enhances the efficiency of managing a library.

Objective of the Project

- To create a user interface using HTML.
- To allow users to add, issue, return, and view books efficiently.
- To learn and implement core software development principles.

Objective

A simple yet efficient library management system with user-friendly features to manage books, users, transactions, and fines.

Features

- **User-Friendly Interface:** Easy-to-use Swing GUI.
- **Book Management:** Add, view, search, and filter books.
- **User Management:** Manage user records with validation.
- **Borrow and Return Books:** Record transactions with fine calculation.
- **Real-Time Search:** Search for books dynamically.
- **Dark Mode Toggle:** UI toggle for light/dark themes.
- **Reporting and Statistics:** Generate summaries for library usage.


Data Validation

- Ensures non-empty input fields.
- Validates email formats using regex.

Code Quality

- Modularized code with reusable components.
- Constants for error messages and queries.

These All Files From GITHUB:-

 namanpal07 Add files via upload		912b423 · 3 weeks ago	 2 Commits
 Code Quality and Innovative Features	Add files via upload		3 weeks ago
 Core Feature Implementation	Add files via upload		3 weeks ago
 Creating CSS and Bootstrap	Add files via upload		3 weeks ago
 DataValidation	Add files via upload		3 weeks ago
 Error Handling and Robustness	Add files via upload		3 weeks ago
 Event Handling and processing	Add files via upload		3 weeks ago
 GUI.java	Add files via upload		3 weeks ago
 HTML templates	Add files via upload		3 weeks ago
 Integration of Components	Add files via upload		3 weeks ago
 JavaScript Validation and Interactivity	Add files via upload		3 weeks ago
 Library.HTML	Add files via upload		3 weeks ago
 LibraryManagementSystem.java	Add files via upload		3 weeks ago
 ProjectDocumentation	Add files via upload		3 weeks ago
 README.md	Add files via upload		3 weeks ago
 Repository Structure	Add files via upload		3 weeks ago
 database.sql	Add files via upload		3 weeks ago
 main.java	Add files via upload		3 weeks ago

This is The Code of Main java File:-

The screenshot displays a web browser window with the address bar showing the URL: `https://github.com/namanpal07/Library_Management_System/blob/main/main.java`. The browser interface includes standard navigation buttons (back, forward, refresh) and a search bar. The page content is divided into two main sections: a left sidebar and a main code area.

Left Sidebar (Files):

- A dropdown menu is set to 'main'.
- A search bar labeled 'Go to file' is present.
- A list of files and folders is shown, including:
 - Code Quality and Innovative Feat...
 - Core Feature Implementation
 - Creating CSS and Bootstrap
 - DataValidation
 - Error Handling and Robustness
 - Event Handling and processing
 - GUI.java
 - HTML templates
 - Integration of Components
 - JavaScript Validation and Interact...
 - Library.HTML
 - LibraryManagementSystem.java
 - ProjectDocumentation
 - README.md
 - Repository Structure
 - database.sql
 - main.java** (highlighted)

Main Code Area:

The code is displayed in a dark-themed editor. At the top, it shows 'Code' and 'Blame' tabs, along with statistics: '33 lines (28 loc) · 1.01 KB'. The code itself is a Java file named `main.java`, containing a `LibraryManagementSystem` class. The code is as follows:

```
1 public class LibraryManagementSystem {
2     private LibraryGUI gui;
3     private LibraryDB database;
4
5     public LibraryManagementSystem() {
6         gui = new LibraryGUI();
7         database = new LibraryDB();
8     }
9
10    public void manageBooks(String action, String title, String author, int year) {
11        if (action.equalsIgnoreCase("add")) {
12            database.addBook(title, author, year);
13        } else {
14            System.out.println("Unsupported book action.");
15        }
16    }
17
18    public void manageUsers(String action, String name, String email) {
19        if (action.equalsIgnoreCase("add")) {
20            database.addUser(name, email);
21        } else {
22            System.out.println("Unsupported user action.");
23        }
24    }
25
26    public static void main(String[] args) {
27        LibraryManagementSystem system = new LibraryManagementSystem();
28
29        // Example usage
30        system.manageBooks("add", "Java Programming", "Author Name", 2023);
31        system.manageUsers("add", "John Doe", "john@example.com");
32    }
33 }
```

The bottom of the image shows a Windows taskbar with various application icons, including the Start button, Search, File Explorer, and several web browsers. The system clock in the bottom right corner indicates the time is 10:26 PM on 09-06-2025.

This the Database Of SQL From GITHUB:-

The image is a screenshot of a web browser displaying a GitHub repository page. The browser's address bar shows the URL: https://github.com/namanpal07/Library_Management_System/blob/main/database.sql. The repository owner is 'namanpal07', and the file is 'database.sql', which is 638 Bytes and 29 lines long. The file content is SQL code for creating a database and tables. The repository structure is visible on the left sidebar, and the Windows taskbar is at the bottom. The SQL code is as follows:

```
1 CREATE DATABASE LibraryDB;
2
3 USE LibraryDB;
4
5 CREATE TABLE Books (
6     BookID INT AUTO_INCREMENT PRIMARY KEY,
7     Title VARCHAR(255),
8     Author VARCHAR(255),
9     Genre VARCHAR(50),
10    Available INT DEFAULT 1
11 );
12
13 CREATE TABLE Users (
14     UserID INT AUTO_INCREMENT PRIMARY KEY,
15     Name VARCHAR(255),
16     Email VARCHAR(255),
17     PhoneNumber VARCHAR(15)
18 );
19
20 CREATE TABLE Transactions (
21     TransactionID INT AUTO_INCREMENT PRIMARY KEY,
22     UserID INT,
23     BookID INT,
24     BorrowDate DATE,
25     ReturnDate DATE,
26     Fine DOUBLE DEFAULT 0,
27     FOREIGN KEY (UserID) REFERENCES Users(UserID),
28     FOREIGN KEY (BookID) REFERENCES Books(BookID)
29 );
```


CONCLUSION:-

The **Library Management System** project provided a practical understanding of how **Object-Oriented Programming (OOP)** principles can be applied to solve real-world problems. By using **Java**, we implemented core concepts such as encapsulation, constructors, static variables, and class-based design. The integration of **HTML** offered a basic user interface to interact with the system.

This project not only improved our **coding skills and logic building**, but also gave insights into **modular software design**, problem-solving, and the importance of **user-friendly interfaces**.

It served as a strong foundation for developing more advanced applications in the future.

FUTURE SCOPE:-

While the current system fulfills basic library operations, several enhancements can significantly increase its effectiveness and usability:

Database Integration: Use of MySQL or MongoDB to store data persistently and securely.

User Authentication: Implement admin and user login systems to manage access control.

Responsive Web Interface: Upgrade UI using HTML, CSS, and JavaScript or modern frameworks like React.

Notifications: Add due-date reminders via email or SMS.

Reporting System: Generate automatic reports for books issued, returned, and overdue.

Mobile App Integration: Develop an Android/iOS app to provide remote access to library services.

Cloud Deployment: Host the system on a web server or cloud platform for real-time online access.