Library Management System

# Abstract

The Library Management System is a simple web-based project designed to manage the issuing and returning of books. It demonstrates the use of JavaScript data structures such as queues and stacks to handle book records efficiently. The system provides a user-friendly interface for issuing and returning books, making it suitable for small-scale libraries or educational demonstrations.

# Introduction

A library management system is essential for maintaining a record of books issued and returned. This project aims to create a minimal and interactive version using HTML, CSS, and JavaScript. It provides two core functionalities: issuing books using a queue structure and returning books using a stack structure. The interface is clean, visually appealing, and responsive, ensuring ease of use.

# Existing System

In traditional systems, book issuing and returning processes are often done manually using registers or spreadsheets. This leads to inefficiency, data loss, and difficulty in tracking book flow. Existing software solutions are complex and not ideal for educational or small-scale purposes.

# Proposed System

The proposed Library Management System automates the process of book issue and return operations. It uses a queue structure for issued books to maintain the order in which books are issued and a stack structure for returned books to maintain the latest return order. This makes the system efficient and provides a better understanding of fundamental data structures in real-life applications.

# Hardware Requirements

• Processor: Intel Core i3 or higher  
• RAM: Minimum 2 GB  
• Storage: 100 MB free disk space  
• Display: 1024x768 resolution or above

# Software Requirements

• Operating System: Windows / Linux / macOS  
• Browser: Google Chrome / Microsoft Edge / Firefox  
• Tools: Any text editor (VS Code, Sublime Text, Notepad++)  
• Technologies Used: HTML, CSS, JavaScript

# Source Code

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8" />  
 <meta name="viewport" content="width=device-width, initial-scale=1.0" />  
 <title>Library Management System</title>  
 <style>  
 body {  
 font-family: Arial, sans-serif;  
 background: linear-gradient(120deg, #d6eaf8, #aed6f1);  
 margin: 0;  
 padding: 0;  
 display: flex;  
 justify-content: center;  
 align-items: center;  
 height: 100vh;  
 }  
 .container {  
 background: white;  
 border-radius: 15px;  
 box-shadow: 0 0 15px rgba(0,0,0,0.2);  
 width: 400px;  
 padding: 25px;  
 }  
 h1 {  
 text-align: center;  
 color: #21618c;  
 }  
 input[type="text"] {  
 width: 90%;  
 padding: 8px;  
 margin: 10px 0;  
 border-radius: 6px;  
 border: 1px solid #ccc;  
 }  
 button {  
 background-color: #3498db;  
 color: white;  
 border: none;  
 padding: 8px 14px;  
 border-radius: 6px;  
 cursor: pointer;  
 margin: 5px;  
 }  
 button:hover {  
 background-color: #2e86c1;  
 }  
 .section {  
 margin-top: 15px;  
 }  
 ul {  
 list-style-type: none;  
 padding: 0;  
 max-height: 100px;  
 overflow-y: auto;  
 }  
 li {  
 background: #f8f9f9;  
 margin: 4px 0;  
 padding: 6px;  
 border-radius: 6px;  
 }  
 </style>  
</head>  
<body>  
 <div class="container">  
 <h1>Library System</h1>  
 <input type="text" id="bookTitle" placeholder="Enter book title" />  
 <div>  
 <button onclick="issueBook()">Issue Book</button>  
 <button onclick="returnBook()">Return Book</button>  
 </div>  
  
 <div class="section">  
 <h3>📚 Issued Books (Queue)</h3>  
 <ul id="issuedList"></ul>  
 </div>  
  
 <div class="section">  
 <h3>📕 Returned Books (Stack)</h3>  
 <ul id="returnedList"></ul>  
 </div>  
 </div>  
  
 <script>  
 let issuedQueue = [];  
 let returnedStack = [];  
  
 function issueBook() {  
 const title = document.getElementById("bookTitle").value.trim();  
 if (title === "") {  
 alert("Enter a book title first!");  
 return;  
 }  
 issuedQueue.push(title);  
 document.getElementById("bookTitle").value = "";  
 displayIssued();  
 }  
  
 function returnBook() {  
 const title = document.getElementById("bookTitle").value.trim();  
 if (title === "") {  
 alert("Enter a book title first!");  
 return;  
 }  
 returnedStack.push(title);  
 document.getElementById("bookTitle").value = "";  
 displayReturned();  
 }  
  
 function displayIssued() {  
 const list = document.getElementById("issuedList");  
 list.innerHTML = "";  
 if (issuedQueue.length === 0) {  
 list.innerHTML = "<li>No issued books.</li>";  
 } else {  
 issuedQueue.forEach((book, i) => {  
 const li = document.createElement("li");  
 li.textContent = (i + 1) + ". " + book;  
 list.appendChild(li);  
 });  
 }  
 }  
  
 function displayReturned() {  
 const list = document.getElementById("returnedList");  
 list.innerHTML = "";  
 if (returnedStack.length === 0) {  
 list.innerHTML = "<li>No returned books.</li>";  
 } else {  
 for (let i = returnedStack.length - 1, n = 1; i >= 0; i--, n++) {  
 const li = document.createElement("li");  
 li.textContent = n + ". " + returnedStack[i];  
 list.appendChild(li);  
 }  
 }  
 }  
 </script>  
</body>  
</html>

# Output

When the project is executed in a browser, it displays a modern and attractive interface with a gradient background. Users can enter the title of a book and choose to either issue or return it. Issued books are listed under the 'Issued Books (Queue)' section, while returned books are displayed under 'Returned Books (Stack)'. The application visually demonstrates how queues and stacks function through book management operations.

# Conclusion

The Library Management System effectively implements core data structures in a practical web environment. It is simple, interactive, and helps in understanding queue and stack concepts. The project can be extended further by adding features like database integration, search functionality, and user authentication.