**Middleware Technologies**

**CSIWZG524**

**Assignment 1.2**

Name: HARI SHANKAR  
Student ID: 2021WA86905

**📚 Books Inventory Management System**

A responsive and user-friendly web application to manage and search book inventory with real-time availability status.

* **GitHub Repository Link:**  
  [*https://github.com/2021wa86905/Books-Inventory-Management-System.git*](https://github.com/2021wa86905/Books-Inventory-Management-System.git)
* **Live Application Link:**  
  [*https://2021wa86905.github.io/Books-Inventory-Management-System/*](https://2021wa86905.github.io/Books-Inventory-Management-System/)

**✨ Features**

* 🔍 **Search by Title**: Instantly filter books by typing in the search bar.
* 📦 **Inventory Display**: Shows quantity and availability status for each book.
* ✅ **Availability Indicator**: Clearly marks books as "Available" or "Unavailable".
* 📱 **Responsive Design**: Optimized for desktops, tablets, and mobile devices.
* 🚫 **No Results Handling**: Displays a friendly message when no books match the search.

**🛠️ Technologies Used**

* **Frontend**: HTML5, CSS3
* **Styling**: Bootstrap 4, custom CSS
* **Programming**: JavaScript (ES6+)

**🚀 How to Run the Project**

**Open the Project**

* Navigate to the project folder.
* Open index.html in your browser.

**Customize Inventory**

* Modify the inventory array in the <script> section of index.html to add or update books.

**🌟 How It Works**

1. Type a book title in the search input.
2. The app filters and displays matching books.
3. Each result shows:
   * 📘 **Title**
   * 🔢 **Quantity**
   * ✅/❌ **Availability Status**
4. If no match is found, a message is shown.

**📂 Project Structure**

📁 books-inventory/

├── index.html # Main HTML structure and JavaScript logic

├── style.css # Custom styling for layout and components

└── README.md # Project documentation

**📖 Learning Points**

* 🔄 **Dynamic DOM Manipulation**: Used JavaScript to update the UI based on user input.
* 🧠 **Conditional Rendering**: Displayed availability status based on quantity.
* 🎨 **Responsive UI**: Combined Bootstrap and custom CSS for a clean layout.

**🔮 Future Enhancements**

* 📝 Add functionality to edit or delete books.
* 📥 Connect to a backend (e.g., Firebase or Node.js) for persistent storage.
* 📊 Add analytics for most searched or out-of-stock books.

**🌐 Web Services Architecture Overview**

* **Frontend**: HTML/CSS/JS runs in the browser.
* **Backend (optional)**: Can be added for persistent inventory storage.
* **Data Source**: Currently hardcoded in JavaScript; can be replaced with API/database.

**📈 Scalability Considerations**

| **Aspect** | **Current State** | **Suggested Enhancements** |
| --- | --- | --- |
| Scalability | Static data in JS | Add backend with database (e.g., MongoDB, Firebase) |
| Fault Tolerance | Basic error handling | Add input validation, fallback UI for data load failures |
| Performance | Lightweight, fast loading | Lazy load results, optimize DOM updates |

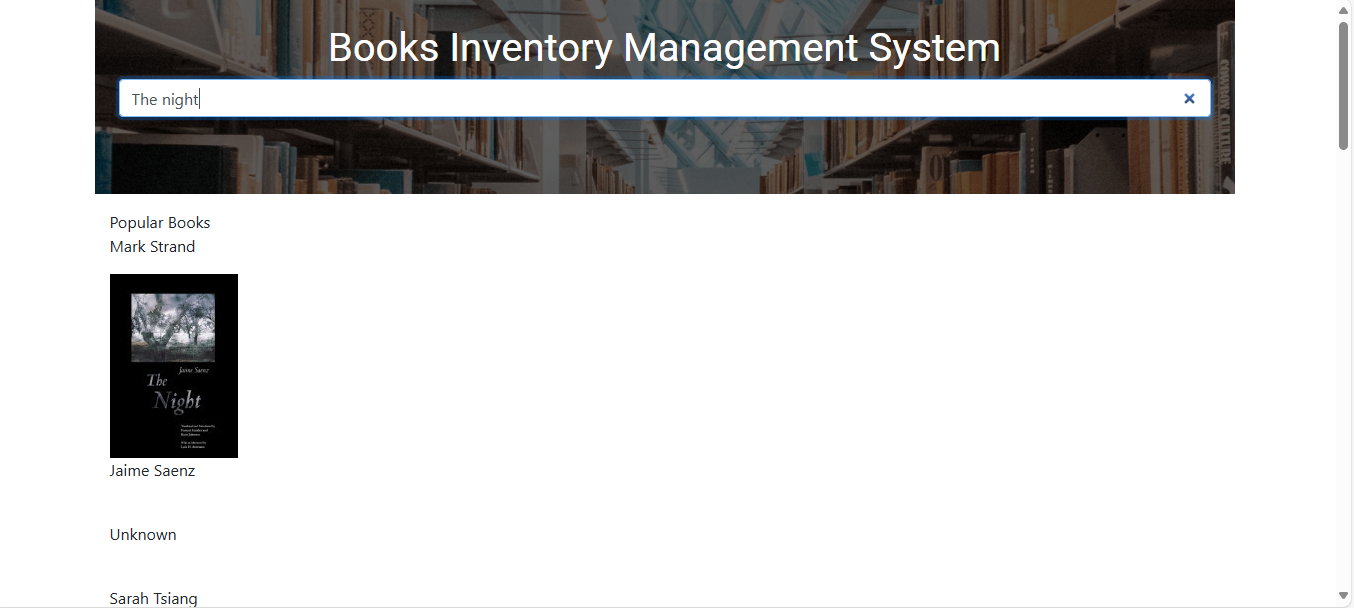
**🛡️ Fault Tolerance Considerations**

* **Current**: Basic handling for no search results.
* **Improvements**:
  + Add input validation.
  + Show fallback message if data fails to load.
  + Use localStorage to cache last known inventory.

**⚡ Performance Considerations**

* **Current**: Fast due to minimal dependencies.
* **Improvements**:
  + Minify CSS/JS.
  + Use service workers for offline support.
  + Optimize DOM updates using virtual DOM or frameworks like React.

**Application Snips:-**



**INDEX code –[HTML]**

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"

integrity="sha384-JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0Z" crossorigin="anonymous" />

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"

integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"

crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"

integrity="sha384-9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN"

crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"

integrity="sha384-B4gt1jrGC7Jh4AgTPSdUtOBvfO8shuf57BaghqFfPlYxofvL8/KUEfYiJOMMV+rV"

crossorigin="anonymous"></script>

<link rel="stylesheet" href="style.css">

</head>

<body>

<div class="container">

<div class="row">

<div class="searchContainer col-12 p-4">

<h1 class="heading">Books Inventory Management System</h1>

<input type="search" id="searchInput" class="search w-90 form-control" placeholder="Type book title">

</div>

<div class="col-12">

<div class="mt-3 text-center">

<div id="spinner" class="d-none spinner-border text-dark" role="status">

<span class="sr-only">Loading...</span>

</div>

<p id="message"></p>

</div>

</div>

<div class="col-12" id="searchResults"></div>

</div>

</div>

<script src="script.js"></script>

</body>

</html>

**CSS code [Java script]**

@import url("https://fonts.googleapis.com/css2?family=Bree+Serif&family=Caveat:wght@400;700&family=Lobster&family=Monoton&family=Open+Sans:ital,wght@0,400;0,700;1,400;1,700&family=Playfair+Display+SC:ital,wght@0,400;0,700;1,700&family=Playfair+Display:ital,wght@0,400;0,700;1,700&family=Roboto:ital,wght@0,400;0,700;1,400;1,700&family=Source+Sans+Pro:ital,wght@0,400;0,700;1,700&family=Work+Sans:ital,wght@0,400;0,700;1,700&display=swap");

.searchContainer {

background-image: url('https://assets.ccbp.in/frontend/dynamic-webapps/library-management-bg.png');

background-size: cover;

height: 32vh;

width: 100%;

text-align: center;

}

.heading {

font-family: 'Roboto';

font-size: 40px;

color: #ffffff;

text-align: center;

}