LIBRARY SYSTEM MANAGEMENT DOCUMENTATION

PROBLEM STATEMENT :

Build a simple library management page where

- Users should be able to see the list of books.
- Instead of loading all the results on the page, perform paging on the list of books – either implement 10 results per page or load on scroll will be a bonus.
- Users should be able to filter the list of books based on Title, Author, Subject and Publish - date
- User should be able to see the count of books in based on each criteria [Title, Author, Subject and publish-date]

APPROACH:

Create a Library management system that starts with the user registration where the user provides the necessary information to create an account after successfully creating an account. User can log in to the system after the authentication process then the system display list of books. The user can apply filters to view the filtered list of books. The user can filter the by either by using book name, author name, subject, or published date.

TOOLS :

- IDE: Visual Studio code
- Language:
    1. HTML-for skeleton
    2. CSS and Bootstrap - for styling
    3. Javascript - for implement functionality

FILES:

- Index.html
- Signup.html
- Userlogin.html
- Loggedpage.html
- Adminlogin.html
- Adminlogged.html
- Preview.html
- Book.html
- Script.js
- Style.css

INDEX.HTML:

Navigation Menu: The webpage includes a navigation menu with three links: Home, User Login, and Admin Login. These links allow users to navigate between different sections or access specific features of the Library Management System.

Carousels: The webpage displays four carousels, each containing a collection of images related to a specific category (such as "MOST READ BOOKS," "CHILDREN'S CORNER," "NOVEL," and "MAGAZINE"). These carousels provide a visual representation of books, newspapers, and magazines that users can browse through by clicking on navigation arrows or using automatic slide transitions.

Book Information: Towards the end of the webpage, there is a footer section that provides information about four books. Each book is displayed in a separate container, including details such as the book's title, author, publication date, and a "PREVIEW" button. The "PREVIEW" button could potentially allow users to access additional information or preview the content of the selected book.

Styling and Layout: The HTML code includes CSS styles applied inline to various elements using the style attribute. These styles control the appearance of the webpage, including the layout, colors, font sizes, and other visual aspects. By defining styles, the webpage achieves a specific look and feel, making it visually appealing and user-friendly.

User and Admin Functionality: Although not explicitly implemented in the provided code snippet, the presence of "User Login" and "Admin Login" links in the navigation menu suggests that there could be separate login systems for users and administrators. Clicking on these links might redirect users to login pages where they can authenticate themselves and access personalized features or administrative functionalities of the Library Management System.

SIGNUP.HTML:

- The signup page allows users to enter their information and sign up for the Library Management System.
- When the user clicks the "SIGN-UP" button, the displayAlert() function is called, which shows an alert message saying "Signedup Successfully."

USERLOGIN.HTML:

- The login form is created using the <form> element.
- Input fields are used to collect the user's username and password.
- Each input field has a corresponding label and is styled using inline CSS.
- The form's action attribute is set to "loggedpage.html," suggesting that upon form submission, the user will be redirected to the "loggedpage.html" page.
- The code includes a "Forgot Password?" link and a "Sign Up" link.
- The "Sign Up" link is styled as white text and points to "signup.html," indicating that clicking the link will redirect the user to the signup page.

ADMINLOGIN.HTML:

- The login form is created using the <form> element. Input fields are used to collect the admin's ID and password.
- The form's action attribute is set to "adminlogged.html," indicating that upon form submission, the user will be redirected to the "adminlogged.html" page.

LOGGEDPAGE.HTML:

Book Filtering:

- The page provides various filtering options for books.
- Input fields are provided for filtering by title, author, subject, and publish date.
- The filters are accompanied by small elements displaying the total count of books that match the respective filter criteria.
- An "Apply Filters" button is provided to trigger the filtering functionality.

Book List:

- A section is dedicated to displaying the book list.
- The book list is rendered as a table using the <table> element.
- The table contains four columns: Title, Author, Subject, and Publish Date.
- Initially, the table is empty, and books will be dynamically populated through JavaScript.

ADMINLOGGED.HTML:

- A section is dedicated to displaying the add book form.
- The form allows users to input book details such as title, author, subject, and publish date.
- Each input field has a corresponding placeholder and is styled using inline CSS.
- A "Add Book" button is provided to submit the form and trigger the addition of the book. The button has an onclick event handler that calls the JavaScript function "addBook()".

PREVIEW.HTML:

- The footer includes a collection of book cards, represented by Bootstrap's card component.
- Each book card contains information such as the book name, author, and published date.
- The book cards are displayed in a grid layout using Bootstrap's grid system and CSS classes.

BOOK.HTML:

Book Preview Images:

- The page displays a series of book images.
- Each book image is represented by an HTML img tag and includes a src attribute pointing to the respective image URL.
- The images are displayed with specific dimensions and margins using inline CSS styling.

SCRIPT.JS:

Book Data: An array named books holds book objects with properties such as title, author, subject, and publishDate. Each book object represents a book in the library.
Pagination: currentPage and booksPerPage variables control the pagination functionality. The displayBooks function receives an array of books to be displayed on the current page. It dynamically populates an HTML table with book data by iterating over the provided books. Pagination is handled by the renderPagination function, which calculates the total number of pages and generates clickable links to navigate between pages. The changePage function updates the current page and calls applyFilters to reapply filters and display the updated book list.

Filtering Books: The applyFilters function retrieves filter values from input fields (title, author, subject, publish date). It filters the books array based on the provided criteria, using the filter method and a callback function. The filtered books are passed to renderPagination to update the pagination and displayBooks to show the filtered books in the table. Additionally, the renderCriteriaCounts function counts the occurrences of each criteria (title, author, subject, publish date) in the filtered books and renders the count values in respective HTML elements.
Rendering Criteria Counts: The renderCriteriaCounts function calculates and displays the counts of each criteria (title, author, subject, publish date) based on the filtered books. It uses the incrementCriteriaCount function to increment the count for each unique value in the criteria object. The renderCount function iterates over the criteria and sums up the counts to display the total count in the corresponding HTML element.
Initialization: The applyFilters function is called initially to apply the filters and display the books based on the default filter values. It also calls renderCriteriaCounts to render the initial counts.

SUMMARY:

HTML Structure:

- The HTML structure defines a header with a navigation menu and links to home, user login, and admin login pages.
- It includes an image tag to display a book preview.

JavaScript Functionality:

- The code defines an array named books that stores book objects with properties such as title, author, subject, and publish date.
- Pagination functionality is implemented with the variables currentPage and booksPerPage. The displayBooks function populates a table with book information based on the current page.
- The code includes a renderPagination function to generate pagination links and a changePage function to update the current page.
- Filtering functionality is provided by the applyFilters function. It retrieves filter values, filters the books array based on the criteria, and updates the table and pagination accordingly.
- The code also includes a renderCriteriaCounts function that calculates and displays the count of each criterion based on the filtered books.
- The applyFilters function is called initially to apply the filters and display the books based on default filter values.