1. Layout Designer (HTML Developer)

Responsibilities: Hermela Tedese

Create the app structure using semantic HTML.

Include the following key sections:

Header: Display the app title (<header> tag).

Main Content: Add two sections:

Add Book Section: A form to input book details (title and author) and add them to the library.

Book List Section: A search bar and a container to display books dynamically.

Footer: Include basic information like copyright (<footer> tag).

Instructions:

Use HTML form elements for input fields and buttons (e.g., <input>, <button>).

Add unique IDs to elements (e.g., id="addBookForm", id="bookList") so that JavaScript can access them.

Ensure semantic tags (<header>, <section>, <footer>) are used to make the structure clean and readable.

2. Stylist (CSS Developer)

Responsibilities: Sami Siraj

Style the app to make it visually appealing and user-friendly.

Ensure responsiveness (the layout adjusts well to different screen sizes).

Apply consistent styles to buttons, input fields, and sections.

Instructions:

Style the Header:

Add a background color, center-align the text, and use a bold font.

Style the Add Book Section:

Add space between input fields and the button.

Ensure the button changes color on hover.

Style the Book List Section:

Use a flexbox or grid layout for displaying book entries.

Add margins between book cards and borders for clarity.

Style the Footer:

Add smaller font size and align text to the center.

Make the design responsive using @media queries for screen width changes.

3. JavaScript Functionalities Developer 1 (Book Management) : Maraki Ayalneh

**Responsibilities:**

Handle book-related actions:

Adding books: Allow users to input book details and add them to a list.

Displaying books: Dynamically display the list of books added.

Removing books: Add functionality to delete books from the list.

**Instructions:**

Create an array (bookList) to store books with their details (title, author).

Write a function (renderBookList) to display the books in the #bookList container dynamically.

Add an event listener to the "Add Book" button:

Validate the input fields to ensure they're not empty.

Push new book objects ({title, author}) into the bookList array.

Call renderBookList to update the display.

Write a function (removeBook(index)) to delete a book from the array and update the display.

4. JavaScript Functionalities Developer 2 (Search and Filter) : Sahlesilase Asmamaw

**Responsibilities:**

Implement search functionality for books.

Filter books based on user input in the search bar.

**Instructions:**

Add an event listener to the search bar input field:

Use the input event to capture real-time user input.

Write a function to filter books in the bookList array:

Compare the search term with book titles and authors (case-insensitive).

Use the renderBookList function to display only the filtered results.

Ensure that when the search bar is cleared, the full book list reappears.

5. Coordinator & Tester

Responsibilities:Lelisa Hailu

Integrate the HTML, CSS, and JavaScript components into a single working app.

Test the app to ensure all functionalities work seamlessly.

Instructions:

**Integrate:**

Link the CSS file in the <head> and the JS file before the closing <body> tag in the HTML.

Ensure all IDs and class names in the JavaScript match the HTML structure.

**Test:**

Add test cases for each functionality:

Add a book with valid details and check the display.

Remove a book and ensure the list updates correctly.

Search for a book and verify that only matching results are shown.

Test responsiveness by resizing the browser window and ensuring the layout adapts well.

Check for any console errors or unexpected behavior.