Step 1: Setting Up the Basic Structure

- **Goal:** Create the basic HTML structure for the application.
- **Tasks:**
  - Create a form with input fields for the book title and author.
  - Add a button to submit the form.
  - Add a container (e.g., a `div`) to display the list of books.
- **Practice:**
  - Write basic HTML with IDs like `bookForm`, `bookTitle`, `bookAuthor`, and `bookList`.
  - Include the JavaScript file in your HTML.

Step 2: Initialize the Application

- **Goal:** Set up the basic structure of the JavaScript object `libraryManagement`.
- **Tasks:**
  - Define the `books` array in the `libraryManagement` object.
  - Write a method `renderBooks` to display a placeholder or empty state when there are no books.
- **Practice:**
  - Test rendering a simple message like "No books available."

Step 3: Adding a Book

- **Goal:** Implement the functionality to add a book.
- **Tasks:**
  - Write the `addBook` method.
  - Ensure that when a book is added, it appears in the list.
  - Add a form submission handler for adding books.
- **Practice:**
  - Test adding multiple books and ensure they are displayed correctly.

Step 4: Displaying Books

- **Goal:** Render books dynamically in the DOM.
- **Tasks:**
  - Update the `renderBooks` method to loop through the `books` array and display each book.

    - Include book details like title, author, and availability status.
- **Practice:**
    - Style the book display using classes or CSS.
    - Test displaying various combinations of books.

Step 5: Borrowing a Book

- **Goal:** Implement the functionality to borrow a book.
- **Tasks:**
    - Write the `borrowBook` method to mark a book as unavailable.
    - Add a "Borrow" button for available books.
    - Ensure the button works correctly by updating the book's status.
- **Practice:**
    - Test borrowing a book and ensure the list updates dynamically.

Step 6: Returning a Book

- **Goal:** Implement the functionality to return a borrowed book.
- **Tasks:**
    - Write the `returnBook` method to mark a book as available.
    - Add a "Return" button for borrowed books.
    - Ensure the button works correctly by updating the book's status.
- **Practice:**
    - Test returning books and check that the list updates dynamically.

Step 7: Filtering Books

- **Goal:** Filter and display only available books.
- **Tasks:**
    - Write the `filterAvailableBooks` method.
    - Add a button or feature to view available books.
    - Display only books that are available for borrowing.
- **Practice:**
    - Test the filter with various scenarios.

Step 8: Sorting Books

- **Goal:** Sort books alphabetically by title.

- **Tasks:**
    - Write the `sortBooks` method.
    - Add a button or feature to sort books by title.
    - Ensure the sorting works dynamically.
- **Practice:**
    - Test sorting with books in various orders.

Step 9: Resetting the Form

- **Goal:** Reset the form after adding a book.
- **Tasks:**
    - Write the `resetForm` method to clear the input fields.
    - Ensure the form is reset every time a book is added.
- **Practice:**
    - Test adding multiple books to check that the form resets properly.

Step 10: Error Handling

- **Goal:** Handle edge cases and provide user feedback.
- **Tasks:**
    - Handle errors when trying to borrow an unavailable book.
    - Handle errors when trying to return an already available book.
    - Display appropriate messages using alerts or notifications.
- **Practice:**
    - Test various edge cases to ensure the application is robust.

Step 11: Polishing the UI

- **Goal:** Improve the visual design and usability.
- **Tasks:**
    - Add CSS for styling the book list and buttons.
    - Use responsive design techniques to make the application mobile-friendly.
- **Practice:**
    - Enhance the user interface for better experience.

Bonus Steps

1. **Add a Search Feature:**
    - Allow users to search for books by title or author.
    - Highlight matching results dynamically.
2. **Add Pagination:**
    - If the list of books grows, divide it into pages.
    - Display navigation buttons to move between pages.
3. **Save Data Locally:**
    - Use `localStorage` to save the book list and retrieve it on page load.
4. **Enhance Error Handling:**
    - Provide more detailed error messages or modals.
5. **Create a Dark Mode:**
    - Add a toggle for light and dark themes.

Practicing Each Step Implement and test each step independently to build confidence before moving to the next. Once all steps are completed, integrate everything to create a fully functional Library Management System.