Core Design Principles

Simplicity and Accessibility, Complexity should be hidden

- The pages should not be cluttered, but show only key information

Books should be as easy to find as possible

- To facilitate this
    - Search Features should be redundant, so there are multiple ways to find books if the user misses how to use a feature
    - There should be multiple redundant ways to navigate through the website, as the user should be using their time to find books, not stuck navigating.
    - All pages should be individually accessible via the URL without the user needing to redo actions just to get to a previous page
        - i.e., redirects are preferred over a single URL returning different html pages given different inputs
        - i.e., JavaScript modifying content is discouraged

Outright Errors are to be mitigated as much as possible

- Redirecting user to a default page is preferred
- If Errors do occur, it should be as easy as possible to return to a previous page

---

Additional Feature

Introduction

A recommendations pane was added to a few key locations, which recommends books to the user based on their recent book viewing activities.

Key Design Decisions for Additional Feature

- The purpose of the recommendations pane is to find books for the user, which they may or may not like.
- The recommendations pane should not track users which have not logged in
- The recommendations pane will not show up on
    - The detailed book interface, as the user has found a book and is viewed as in a found state.
    - Search Results as both share the core principle: to find books for the user
    - Error pages as they are meant to quickly redirect users back
    - Authentication pages as recommendations will be intrusive

CONTINUED OVERLEAF

- The site recommends books to users based on their 10 previously viewed books,
    - If the user is not logged in, it will randomly choose 10 books for interest
    - If the user is logged in, it will rank books based on
        1. Author Weight 4
        2. Publisher Weight 2
        3. Release Year Weight 1
    - The weighting is cumulative and each of the entries is doubly weighted than the next down, with the following reasoning,

      if they (user) find a book they like,

    - they are likely to enjoy books from the same author,
        - This is weighed the highest as authors tend to stay consistent with the style of books that they write
    - they may also prefer books from the same publisher,
        - as a publisher generally publishes similar genres of books,
        - however large publishers are more likely to diversify their catalogue, so the weighting is kept lower that the author
    - Authors, publishers, and trends generally will change the general content of books with time
        - so, the books published and written around the same year ± 1 year are weighted higher than those not in the same period
        - this will also pull non-related books, so the weighting is kept low
- The site will then recommend the top 10 books which meet this ranking,
- If the recommendations pull less than 10 books from the catalogue, the site will then randomly select the remainder randomly

---

Key Design Decisions

Accessibility

- The palette of the website are pastel colours chosen to be as feel toned down and accessible
    - The palette is from https://lospec.com/palette-list/sobeachy8

      
    - The Website should be WCAG 2.0 Compliant,
        - Exception when a few buttons are hovered over,
            - Regarded as too minor as the user can move the mouse

- The user should not be confused
    - Any button when hovered over should have a slight highlight and/or a colour change of text to indicate
    - The LOGOUT Button only appears if the user is logged in
    - The NEXT PAGE button only appears if there is a next page in the results list
      The PREVIOUS PAGE button has similar behaviour
    - Any clickable item should have the mouse show as a pointer

Application Blueprints (i.e. Cores) (Simplicity)

- The cores of the applications will be
  - Searching
  - Authentication
  - Book Viewing
  - Intermediate Interfaces (Home screens and search forms)

- Everything should be routed to their respective cores
- Everything should be routable to each other without requiring any extra navigation
  - Hence there is a sidebar available which can link to any of the cores

- Although the List is technically both a search interface and a book viewing interface
  - List is placed under the Book viewing as the search component is much larger and it is easier to manage if List is placed so.
- Reviews are on the same page as the books, so they are placed under book viewing.

Quality of Life (Accessibility)

- The application should be as easy to use as possible
- Hence the use of JavaScript for the use of
  - Sorting tables using each table header,
  - If clicked multiple times, it toggles between ascending to descending
    - The description is not sortable as the feature was not seen as helpful

---

Approach to Errors (Complexity should be hidden)

- *Any error page should be avoided*
- The user will be guided to the defaults
  - When page query > number of pages
    - It routes to the highest possible page for that list
  - When page query <= 0,
    - It routes to page =1
- If an error has occurred,
  - the page tries to get the user back to actual content as fast as possible, hence the page text will redirect to the previous history item.

Continued Overleaf

Redundant Design (Accessibility)

This helps with the core principle of ***Accessibility*** as it facilitates users to utilise its features by making multiple unintrusive ways to access them which are normally hidden away which helps with ***Simplicity***

- Features are designed so that there is a redundant way to access each of them, as seen below (Most are under the sidebar)

Feature Category	**Feature/s**	**Ways to Access**
Auth	Login/Logout/Register	Sidebar Button/Home
Search	Search Form	Sidebar Button/Search Home
Search	Search (Query)	List/Book View and Review
Interface	Home	Header Text/ Sidebar Image/ Sidebar Button
Interface	Search Home	Sidebar Button/Home
Books	View Book and Review	Recommendations/Search/List
Books	List	Sidebar Button/Home

Search

This uses the core principle of ***Simplicity***, as it hides all the processing and selection, the advantage is that each of the modules can follow the ***Single Responsibility principle***, which lets additional functionality be added (search queries via URL), and helps with making ***Books should be as easy to find as possible***

- All Search forms are to be serviced via the same HTML document
  - The HTML Document gets passed a $search\_type$ which indicates the labels and form used
  - All data except the dropdown selections are handled through two fields,
    - One for integers
    - One for strings
  - Internally, are then handled by a module checking for validity
    and then passed to independent modules which fetch the data from the $memory\_repository$
    - This is separated as each independent can now service a different route for queries which bypasses the forms completely and can facilitate the linking of search to general data shown in the *Books Core*
  - then the fetched data is passed to another module, for validation and to reduce duplicate code.
    It checks for content and
    - If books found > 1
      - Sends a generic table template
    - If books found = 1
      - Redirects to *Books Core* for handling
    - If books found = 0
      - Sends an error page

Miscellaneous

- The user is automatically rerouted to the book interface with a review form if they are logged in, which slightly defies the core principle that ***Books should be as easy to find as possible***, but it is more likely for a user to review a book if they do not have to do any additional steps, (low friction), so the core principle was decided against.