BOOK CONNECT WEBSITE

INTRODUCTION TO THE CODE AND ITS PURPOSE

CODE STRUCTURE

- The code starts by importing necessary data and constants from the ‘data.js’ file.(‘BOOKS_PER_PAGE’, ‘authors’ , ‘books’, and ‘genres’).
- The code includes data validity using conditional statements. Error is shown if ‘books’ data is not an array, and it the range provided is invalid
- The code then initializes variables and objects needed for the application, such as the ‘matches’ array to store filtered books, ‘page’ for pagination, and ‘day’ and ‘night’ objects defining color schemes.
- CSS variables using the ‘css’ object, which contains the ‘day’ and ‘night’ color schemes.
- The ‘dataListButton’ and ‘dataListItems’ elements are retrieved from the DOM for further manipulation.
- The code creates a document fragment, sets initial start and end index values for book display, and defines the ‘loadBooks()’ function to handle loading and appending books to the DOM.
- ‘updateRemaining()’ function calculates and updates the remaining book count and manages the state of the "Show more" button based on the available books.

BOOK FILTERING

- The ‘loadBooks()’ function is responsible for loading and displaying books on the page. Check if the end index is greater than or equal to the total number of books and hides “Show more” button if all books.
- The function extracts a subset of books based on the current start and end indices using the ‘slice()’ method.
- Book information is retrieved and book previews are created using ‘createPreview()’.
- Book previews are appended to a document fragment for efficiency.
- The document fragment is then appended to the ‘dataListItems’ element.
- Start and end indices are updated for the next batch of books.
- ‘updateRemaining()’ calculates remaining book count and updates the button.
- The "Show more" button triggers ‘handleShowMore()’ to load more books.

BOOK DISPLAY

- The book display is created using the ‘createPreview()’ function.
- This function takes in parameters such as author, ID, image, and title to generate a book preview element.
- Inside the function, a ‘ ’ element with the class "book-preview" is created to hold the preview content.
- The author's UUID is used to retrieve the corresponding author name from the ‘authors’ object.
- The book preview content is generated using template literals, incorporating the retrieved author name, book image, title, and ID.
- The generated content is then assigned to the ‘innerHTML’ property of the preview element.
- the completed preview element is returned.
- The returned preview element is appended to the book display container, resulting in the visual representation of the book with its details on the page.

SEARCH FUNCTIONALITY

- The application incorporates a search functionality to allow users to filter books based on their preferences.
- The ‘searchBooks()’ function is responsible for filtering the books based on the user's search criteria.
- The function takes in an object ‘filters’ that contains properties such as ‘title’, ‘author’, and ‘genre’.
- A loop iterates over the matches array, which contains all the books.
- The ‘titleMatch’ variable checks if the book's title includes the provided search title.
- The ‘authorMatch’ variable verifies if the book's author matches the selected author in the search filters.
- The ‘genreMatch’ variable determines if the book's genres include the chosen genre in the search filters.
- If a book satisfies all the search criteria, it is added to the ‘results’ array.
- The function returns the ‘results’ array containing the filtered books.
- The ‘updateList()’ function is called with the filtered results to update the book display based on the search filters.

USER INTERFACE

- The CSS custom properties ‘--color-dark’ and ‘--color-light’ are used to define the color scheme for the application.
- By changing the values of these properties, the theme can be switched between a light (day) and dark (night) mode.
- The theme is set based on the user's preferred color scheme using the window.matchMedia function.
- User interactions with the theme are handled by the ‘dataSettingTheme’ element and the ‘dataSettingsForm’ event listener.
- The search overlay allows users to input search criteria and refine the displayed books based on their preferences.
- The settings overlay enables users to adjust the application's theme between day and night modes.
- When users submit the settings form or cancel the overlay, the changes are applied, and the overlay is closed.

CONCLUSION

- The main features and functionalities include:Loading and displaying books with the ability to show more books as the user scrolls.

I. Filtering books based on search criteria such as title, author, and genre.

II. Interactive user interface with the option to switch between day and night themes.

III. Smooth handling of user interactions with search and settings overlays.

- The code utilizes JavaScript, CSS, and the Document Object Model (DOM) to create a seamless and engaging user experience.
- The code can be further enhanced and customized based on specific requirements and additional features.
- Overall, this book filtering and display application demonstrates the power of JavaScript in creating dynamic and interactive web applications for various purposes.

CODE ISSUES

- Overall, this book filtering and display application demonstrates the power of JavaScript in creating dynamic and interactive web applications for various purposes.(like the list updates itself after clicking the datalistbutton.
- Another issues is I can’t seem to be able to use the search button to display searched items
- The code would benefit from better code documentation, including comments and clear naming conventions, to enhance its readability and understandability for future maintenance and collaboration.