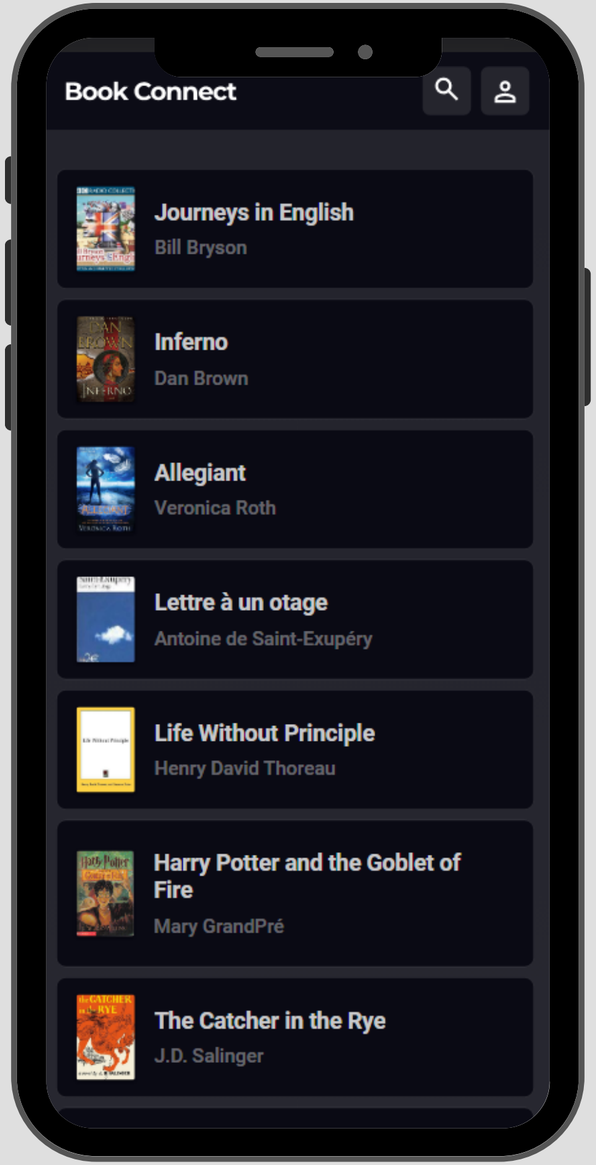
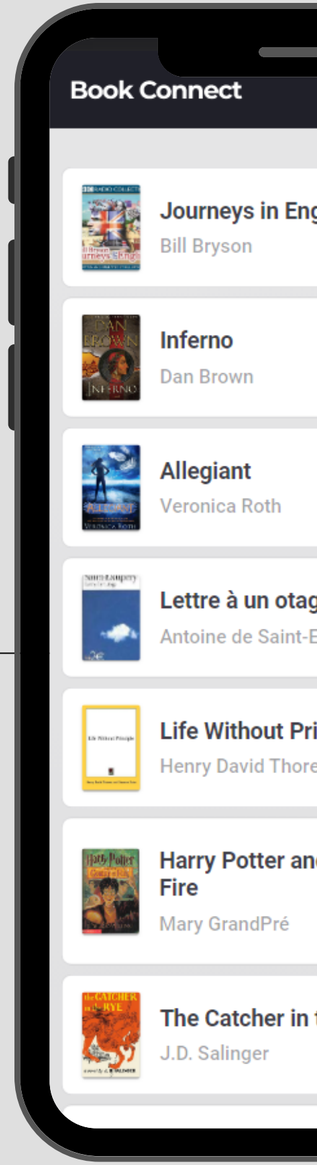


BOOK CONNECT

HTML, CSS, JAVASCRIPT

LETHABO MATHABATHA



https://github.com/lethabomathabatha/LETMAT077_FTO2301_GroupB_LethaboMathabatha_IWA19

01 - problem

02 - quick fixes

03 - app test

The app currently does not meet any of the user story requirements and functionality.
The underlying code requires debugging to make the app viable for launch/use.

USER STORIES TO ACHIEVE

1 - As a user, I want to view a list of book previews, by title and author, so that I can discover new books to read.

2- As a user, I want an image associated with all book previews so that I can recognize a book by the cover even if I forgot the name.

3- As a user, I want to have the option of reading a summary of the book so that I can decide whether I want to read it.

4- As a user, I want to have the option of seeing the date that a book was published so that I can determine how easy it is to obtain second-hand.

6- As a user, I want to find books based on specific text phrases so that I don't need to remember the entire title of a book.

7- As a user, I want to filter books by author so that I can find books to read by authors that I enjoy.

8- As a user, I want to filter books by genre so that I can find books to read in genres that I enjoy.

9- As a user, I want to toggle between dark and light modes so that I can use the app comfortably at night.

QUICK FIXES

Exporting existing data from seperate files, and linking them to the main script via **importing** is a significant starting point.

```
export const BOOKS_PER_PAGE = 36;

export const authors = {···
}

export const genres = {···
}

export const books = [···
]
```

data.js

```
import { BOOKS_PER_PAGE, authors, genres, books } from "./data.js";
```

scripts.js

QUICK FIXES

```
//Global query selectors:
// settings query selectors
const settings = document.querySelector('[data-header-settings]');
const settingsOverlay = document.querySelector('[data-settings-overlay]');
const settingsForm = document.querySelector('[data-settings-form]');
const cancelSettings = document.querySelector('[data-settings-cancel]');

// search query selectors
const search = document.querySelector(".header__button");
const searchOverlay = document.querySelector('[data-search-overlay]');
const searchForm = document.querySelector('[data-search-form]');
const searchCancel = document.querySelector('[data-search-cancel]');
const authorMatch = document.querySelector('[data-search-authors]');

// list and overlay query selectors
const list = document.querySelector("[data-list-items]");
const loadMore = document.querySelector("[data-list-button]");
const previewOverlay = document.querySelector("[data-list-active]");
const closeBtn = document.querySelector("[data-list-close]")
const titleOverlay = previewOverlay.querySelector(".overlay__title");
const dataOverlay = previewOverlay.querySelector(".overlay__data");
const overlayBlur = previewOverlay.querySelector(".overlay__blur");
const overlayImage = previewOverlay.querySelector(".overlay__image");
const infoOverlay = previewOverlay.querySelector("[data-list-description]");
```

Selecting elements from the entire HTML/document to allow interaction with the Javascript code.

scripts.js

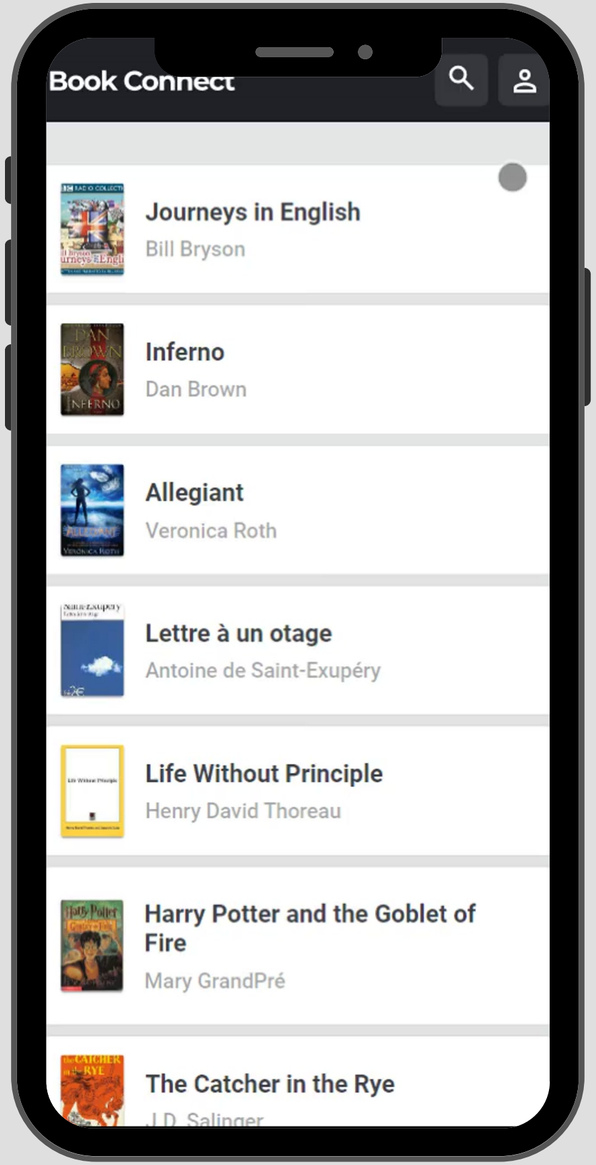
QUICK FIXES

```
// search functionality:

/**
* @param {Event} event handles the event when the search icon is clicked,
*/
// event listener for when the search icon is clicked, it should display a search menu
// overlay with options to search for books

search.addEventListener("click", () => {

    searchOverlay.show();
    e.preventDefault();
    searchForm.classList.toggle('hidden');
});

// event listener for when the cancel button is clicked, it should close the search menu overlay
searchCancel.addEventListener('click', () => {
    searchOverlay.close();
    e.preventDefault();

});
```

scripts.js

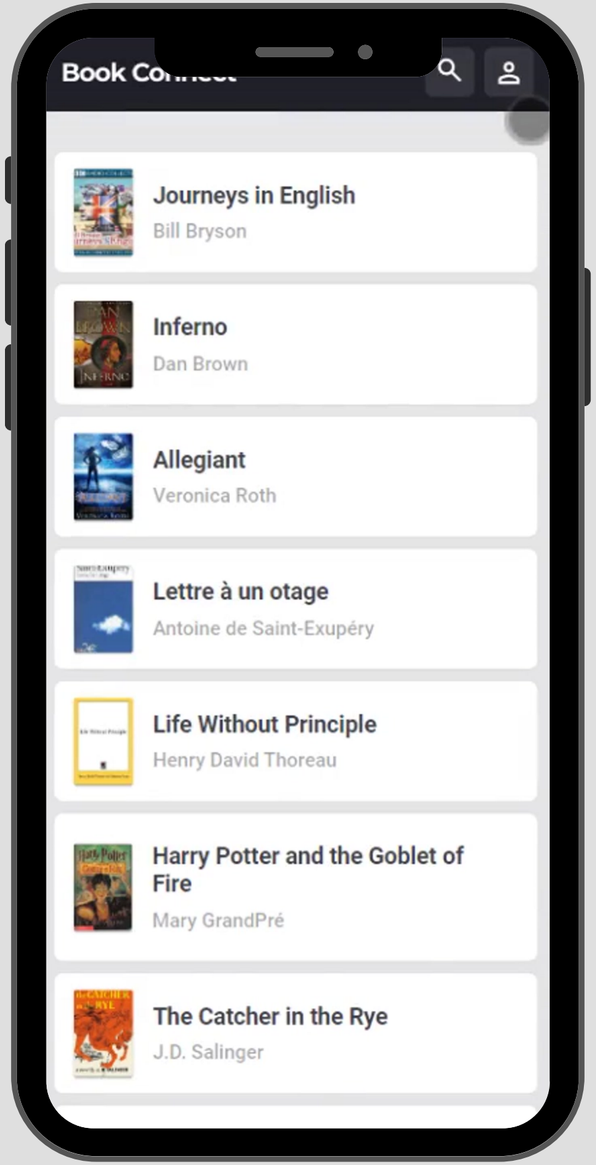
Adding **comments and JSDoc** adds readability and makes the existing code understandable for future code maintainence, as well as make it easier for other developers to use.

Well-commented code is also generally easier to debug.



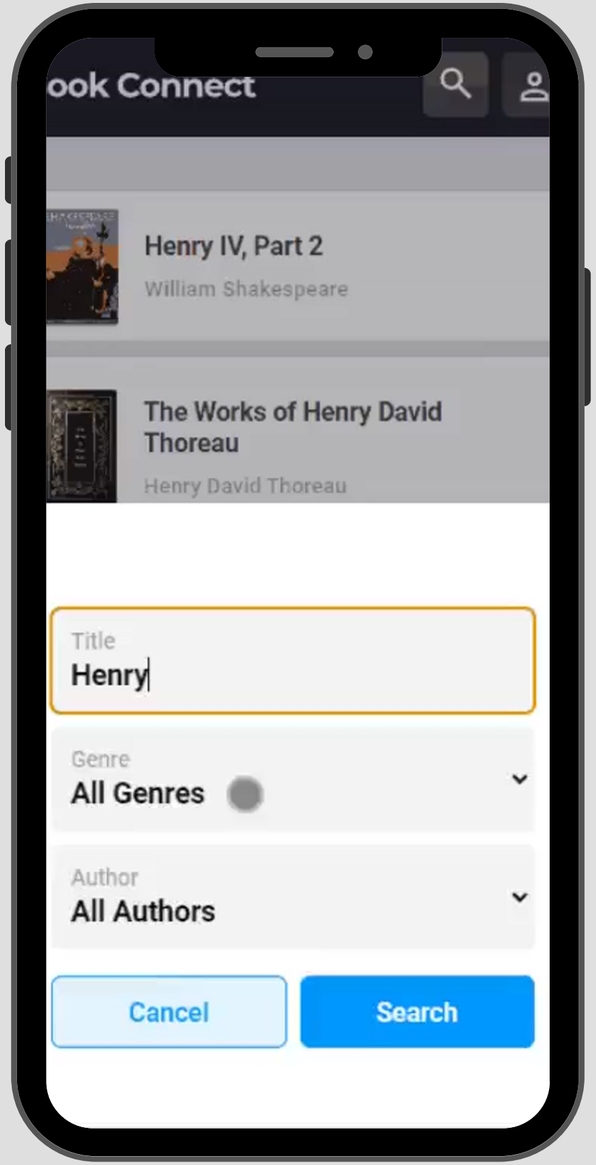

```
// book preview functionality
const innerHTML = ( book, index ) => {
  const booksElement = document.createElement("div");
  booksElement.className = "preview";
  booksElement.dataset.index = `${index}`;
  booksElement.id = book.id;

  booksElement.innerHTML = ` <img src = ${book.image}
  class = 'preview__image'  alt="${book.title} book image"></img>
  <div class="preview__info">
    <h3 class="preview__title">${book.title}</h3>
    <div class="preview__author">${authors[book.author]}</div>
  </div>`;

  return booksElement;
};


for (let i = 0; i < BOOKS_PER_PAGE; i++) {
  list.appendChild(innerHTML(books[i], i));
}
let loaded = 0;

loadMore.innerHTML = `<span>Show More</span>
<span class = "list__remaining">(
    ${books.length - BOOKS_PER_PAGE - loaded}
    )</span>`;

const moreBooks = (e) => {
  loaded += BOOKS_PER_PAGE;
  let booksLeft = books.length - BOOKS_PER_PAGE - loaded;
  let moreBtn = booksLeft > 0 ? booksLeft : 0;
  loadMore.innerHTML = `
  <span>Show more</span>
  <span class = "list__remaining">(${moreBtn})</span>`;

  let booksLoaded = BOOKS_PER_PAGE + loaded;
```

BOOK PREVIEW FUNCTIONALITY



DAY-NIGHT TOGGLE

```
// Theme settings functionality
settings.addEventListener("click", () => {
    settingsOverlay.show();
    settingsForm.classList.toggle('hidden');
    document.querySelector('[data-settings-overlay]').classList.toggle('hidden');
});

const theme = {
    day: {
        dark: '10, 10, 20',
        light: '255, 255, 255',
    },

    night :{
        dark: '255, 255, 255',
        light: '10, 10, 20',
    }
    }

   settingsForm.addEventListener('submit', (e) => {
        e.preventDefault();
        const formData = new FormData(e.target);
        const result = Object.fromEntries(formData);

        document.documentElement.style.setProperty('--color-light', theme[result.theme].light);
        document.documentElement.style.setProperty('--color-dark', theme[result.theme].dark);
        settingsOverlay.close();
    });

// Cancel settings selection functionality
cancelSettings.addEventListener("click", () => {
    settingsOverlay.close();
    settingsForm.classList.toggle('hidden');
    document.querySelector('[data-settings-overlay]').classList.toggle('hidden');
});
```




```
// Genre-author filter functionality
// when user selects specific genre, display books under that genre
    genreSelect.addEventListener('change', (e) => {
        const selectedGenre = e.target.value;
        let booksByGenre = [];

        // If the user selects 'All Genres', display all the books, else, filter the books by the selected genre
        if (selectedGenre === 'all') {
        booksByGenre = bookData;
        } else {
        bookData.forEach(book => {
            if (book.genres.includes(selectedGenre)) {
            booksByGenre.push(book);
            };
        });

        // if no books are found for the selected genre, display a message to the user
        if (booksByGenre.length === 0) {
            const errorMessage = document.querySelector('[data-list-message]');
            errorMessage.textContent = `No books found for the genre`;
        }
    };

// Clear the current book list and display the new list of books
    const bookList = document.querySelector('[data-list-items]');
    bookList.innerHTML = '';

    booksByGenre.forEach(book => {
       const bookCard = document.createElement('div');
       bookCard.innerHTML = `
         <h3>${book.title}</h3>
        <p>${book.description}</p>
         <img src="${book.image}" alt="${book.title}">
       `;
       bookList.appendChild(bookCard);
     });
   });
```

GENRE-AUTHOR SEARCH FUNCTIONALITY