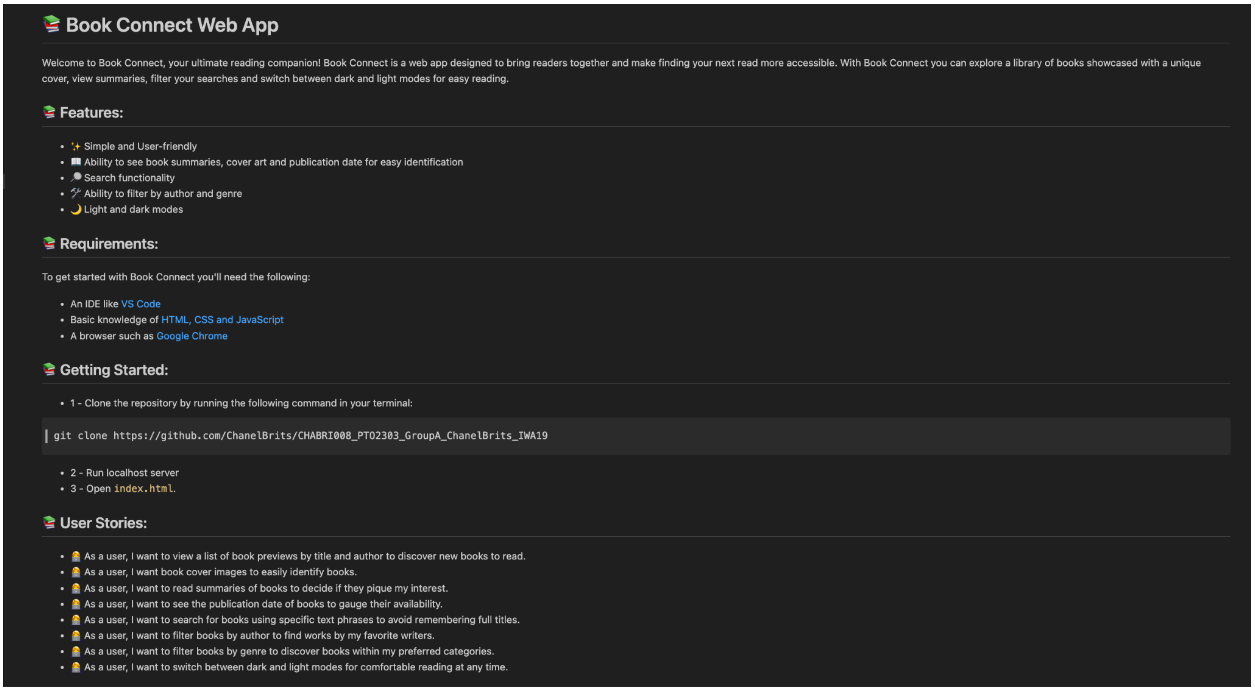
DWA_03.4 Knowledge Check_DWA3.1

1. Please show how you applied a Markdown File to a piece of your code.



📚 Book Connect Web App

Welcome to Book Connect, your ultimate reading companion! Book Connect is a web app designed to bring readers together and make finding your next read more accessible. With Book Connect you can explore a library of books showcased with a unique cover, view summaries, filter your searches and switch between dark and light modes for easy reading.

📚 Features:

- ✨ Simple and User-friendly
- 📖 Ability to see book summaries, cover art and publication date for easy identification
- 🔎 Search functionality
- 🛠️ Ability to filter by author and genre
- 🌙 Light and dark modes

📚 Requirements:

To get started with Book Connect you'll need the following:

- An IDE like VS Code
- Basic knowledge of HTML, CSS and JavaScript
- A browser such as Google Chrome

📚 Getting Started:

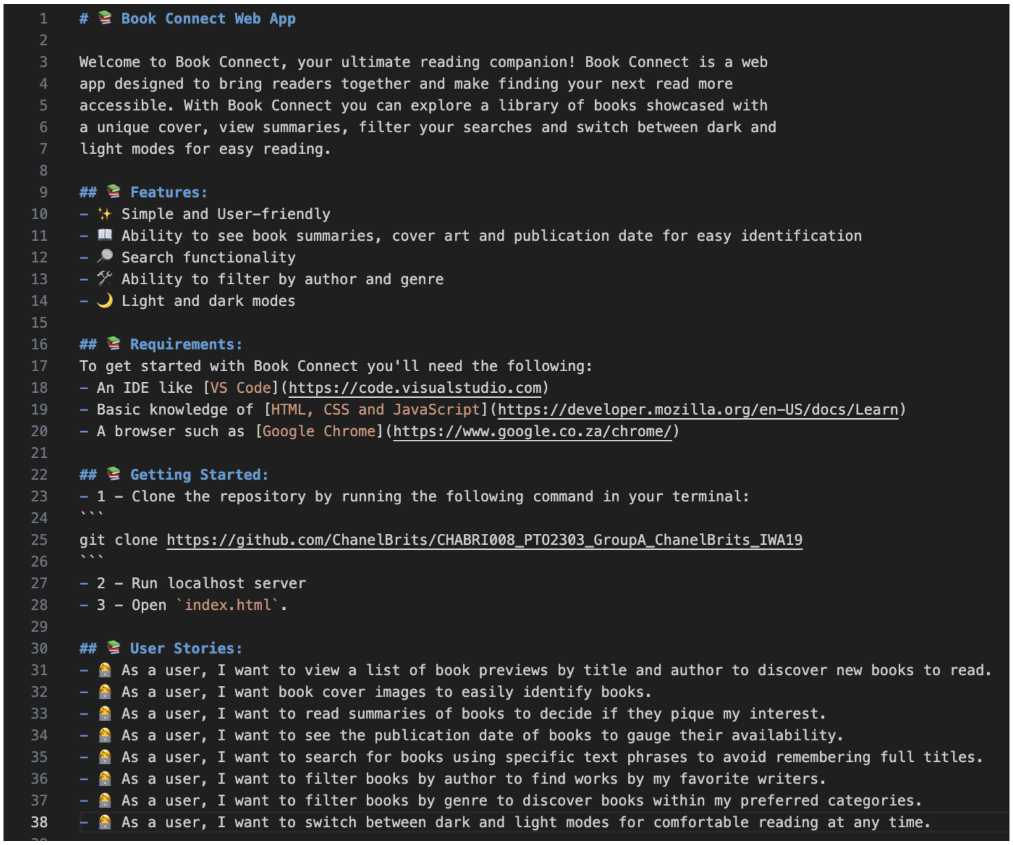
- 1 - Clone the repository by running the following command in your terminal:

```
git clone https://github.com/ChanelBrits/CHABRI008_PTO2303_GroupA_ChanelBrits_IWA19
```

- 2 - Run localhost server
- 3 - Open `index.html`.

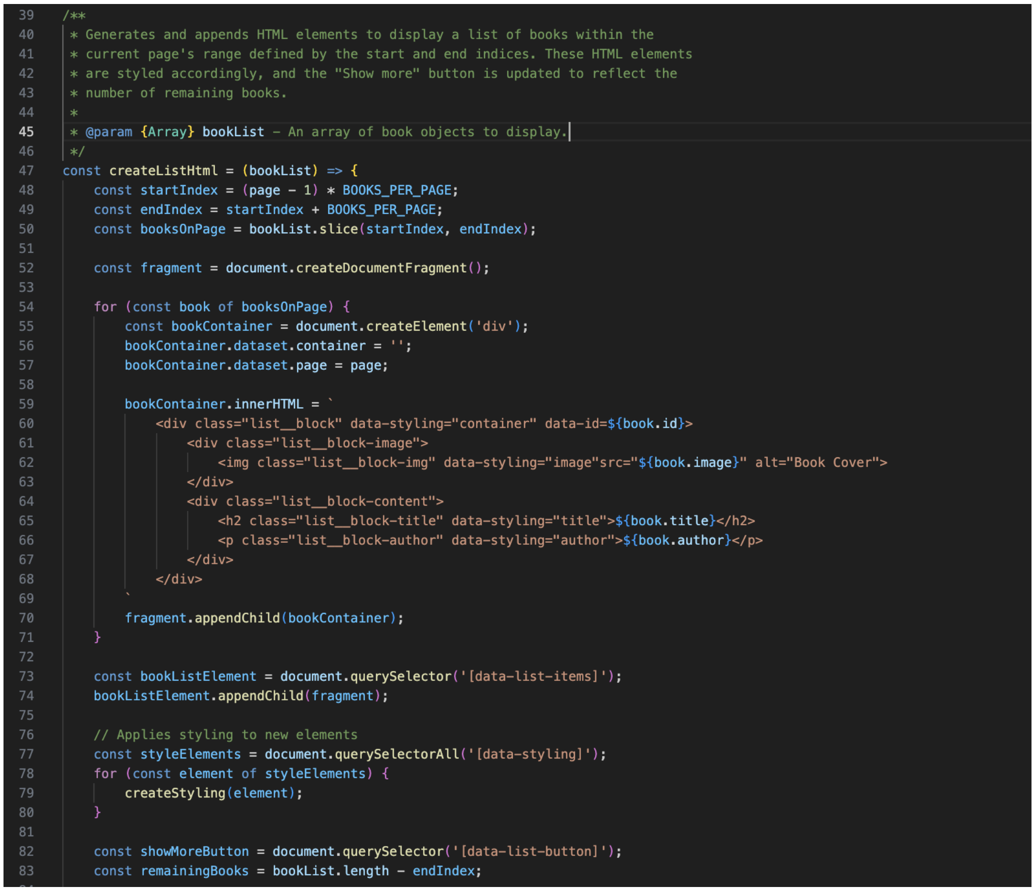
📚 User Stories:

- 👩‍💻 As a user, I want to view a list of book previews by title and author to discover new books to read.
- 👩‍💻 As a user, I want book cover images to easily identify books.
- 👩‍💻 As a user, I want to read summaries of books to decide if they pique my interest.
- 👩‍💻 As a user, I want to see the publication date of books to gauge their availability.
- 👩‍💻 As a user, I want to search for books using specific text phrases to avoid remembering full titles.
- 👩‍💻 As a user, I want to filter books by author to find works by my favorite writers.
- 👩‍💻 As a user, I want to filter books by genre to discover books within my preferred categories.
- 👩‍💻 As a user, I want to switch between dark and light modes for comfortable reading at any time.

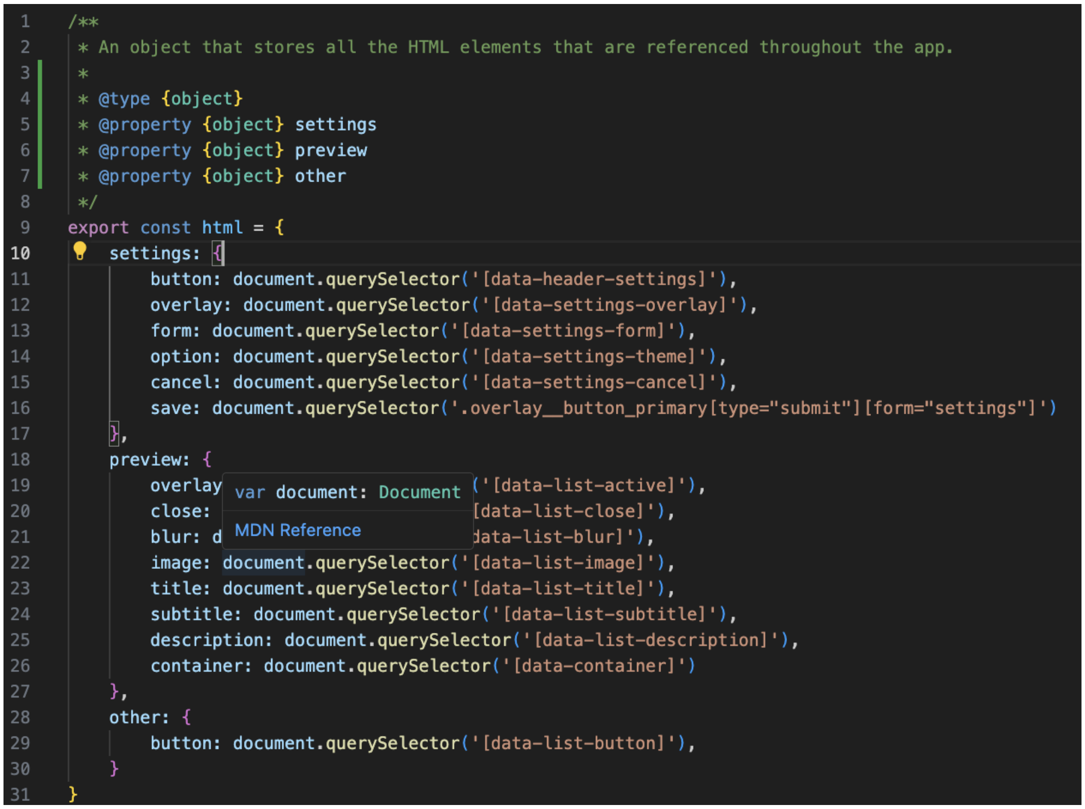


~~~markdown
# 📚 Book Connect Web App

Welcome to Book Connect, your ultimate reading companion! Book Connect is a web
app designed to bring readers together and make finding your next read more
accessible. With Book Connect you can explore a library of books showcased with
a unique cover, view summaries, filter your searches and switch between dark and
light modes for easy reading.

## 📚 Features:
- ✨ Simple and User-friendly
- 📖 Ability to see book summaries, cover art and publication date for easy identification
- 🔎 Search functionality
- 🛠️ Ability to filter by author and genre
- 🌙 Light and dark modes

## 📚 Requirements:
To get started with Book Connect you'll need the following:
- An IDE like [VS Code](https://code.visualstudio.com)
- Basic knowledge of [HTML, CSS and JavaScript](https://developer.mozilla.org/en-US/docs/Learn)
- A browser such as [Google Chrome](https://www.google.co.za/chrome/)

## 📚 Getting Started:
- 1 - Clone the repository by running the following command in your terminal:
```
git clone https://github.com/ChanelBrits/CHABRI008_PTO2303_GroupA_ChanelBrits_IWA19
```
- 2 - Run localhost server
- 3 - Open `index.html`.

## 📚 User Stories:
- 👩‍💻 As a user, I want to view a list of book previews by title and author to discover new books to read.
- 👩‍💻 As a user, I want book cover images to easily identify books.
- 👩‍💻 As a user, I want to read summaries of books to decide if they pique my interest.
- 👩‍💻 As a user, I want to see the publication date of books to gauge their availability.
- 👩‍💻 As a user, I want to search for books using specific text phrases to avoid remembering full titles.
- 👩‍💻 As a user, I want to filter books by author to find works by my favorite writers.
- 👩‍💻 As a user, I want to filter books by genre to discover books within my preferred categories.
- 👩‍💻 As a user, I want to switch between dark and light modes for comfortable reading at any time.
~~~

2. Please show how you applied JSDoc Comments to a piece of your code.

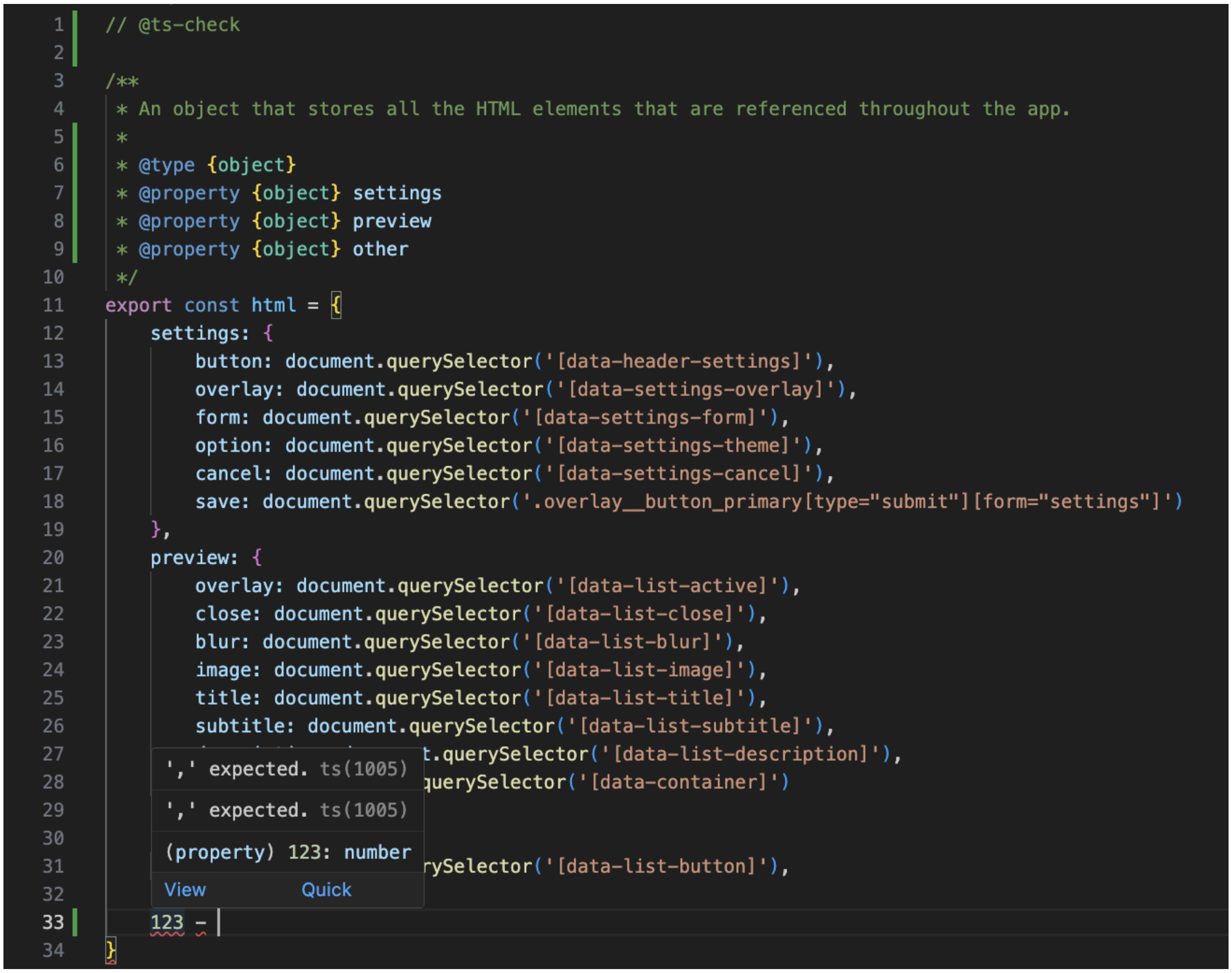


```
/**
 * Generates and appends HTML elements to display a list of books within the
 * current page's range defined by the start and end indices. These HTML elements
 * are styled accordingly, and the "Show more" button is updated to reflect the
 * number of remaining books.
 *
 * @param {Array} bookList - An array of book objects to display.
 */
const createListHtml = (bookList) => {
    const startIndex = (page - 1) * BOOKS_PER_PAGE;
    const endIndex = startIndex + BOOKS_PER_PAGE;
    const booksOnPage = bookList.slice(startIndex, endIndex);

    const fragment = document.createDocumentFragment();

    for (const book of booksOnPage) {
        const bookContainer = document.createElement('div');
        bookContainer.dataset.container = '';
        bookContainer.dataset.page = page;

        bookContainer.innerHTML = `
            <div class="list__block" data-styling="container" data-id=${book.id}>
                <div class="list__block-image">
                    <img class="list__block-img" data-styling="image"src="${book.image}" alt="Book Cover">
                </div>
                <div class="list__block-content">
                    <h2 class="list__block-title" data-styling="title">${book.title}</h2>
                    <p class="list__block-author" data-styling="author">${book.author}</p>
                </div>
            </div>
        `
        fragment.appendChild(bookContainer);
    }

    const bookListElement = document.querySelector('[data-list-items]');
    bookListElement.appendChild(fragment);

    // Applies styling to new elements
    const styleElements = document.querySelectorAll('[data-styling]');
    for (const element of styleElements) {
        createStyling(element);
    }

    const showMoreButton = document.querySelector('[data-list-button]');
    const remainingBooks = bookList.length - endIndex;
```



```
/**
 * An object that stores all the HTML elements that are referenced throughout the app.
 *
 * @type {object}
 * @property {object} settings
 * @property {object} preview
 * @property {object} other
 */
export const html = {
    settings: {
        button: document.querySelector('[data-header-settings]'),
        overlay: document.querySelector('[data-settings-overlay]'),
        form: document.querySelector('[data-settings-form]'),
        option: document.querySelector('[data-settings-theme]'),
        cancel: document.querySelector('[data-settings-cancel]'),
        save: document.querySelector('.overlay__button_primary[type="submit"][form="settings"]')
    },
    preview: {
        overlay                          ('[data-list-active]'),
        close:                           [data-list-close]'),
        blur: d                          data-list-blur]'),
        image: document.querySelector('[data-list-image]'),
        title: document.querySelector('[data-list-title]'),
        subtitle: document.querySelector('[data-list-subtitle]'),
        description: document.querySelector('[data-list-description]'),
        container: document.querySelector('[data-container]')
    },
    other: {
        button: document.querySelector('[data-list-button]'),
    }
}
```

3. Please show how you applied the @ts-check annotation to a piece of your code.



```
// @ts-check

/**
 * An object that stores all the HTML elements that are referenced throughout the app.
 *
 * @type {object}
 * @property {object} settings
 * @property {object} preview
 * @property {object} other
 */
export const html = {
    settings: {
        button: document.querySelector('[data-header-settings]'),
        overlay: document.querySelector('[data-settings-overlay]'),
        form: document.querySelector('[data-settings-form]'),
        option: document.querySelector('[data-settings-theme]'),
        cancel: document.querySelector('[data-settings-cancel]'),
        save: document.querySelector('.overlay__button_primary[type="submit"][form="settings"]')
    },
    preview: {
        overlay: document.querySelector('[data-list-active]'),
        close: document.querySelector('[data-list-close]'),
        blur: document.querySelector('[data-list-blur]'),
        image: document.querySelector('[data-list-image]'),
        title: document.querySelector('[data-list-title]'),
        subtitle: document.querySelector('[data-list-subtitle]'),
                       t.querySelector('[data-list-description]'),
                       querySelector('[data-container]')


                       rySelector('[data-list-button]'),

    123 - 
}
```

```
',' expected. ts(1005)
',' expected. ts(1005)
(property) 123: number
```

4. As a BONUS, please show how you applied any other concept covered in the 'Documentation' module.