Book Connect

Kgotso Makhalimele

CREATE A BOOKSTORE THAT CAN:

Show Users a preview of books by title and author

Show an image associated with all book previews

Show a summary of the book

Let the user see when a book was published

Allow users to search for books by using certain phrases

Filter books by author or genre

Toggle between light and dark mode

Overview

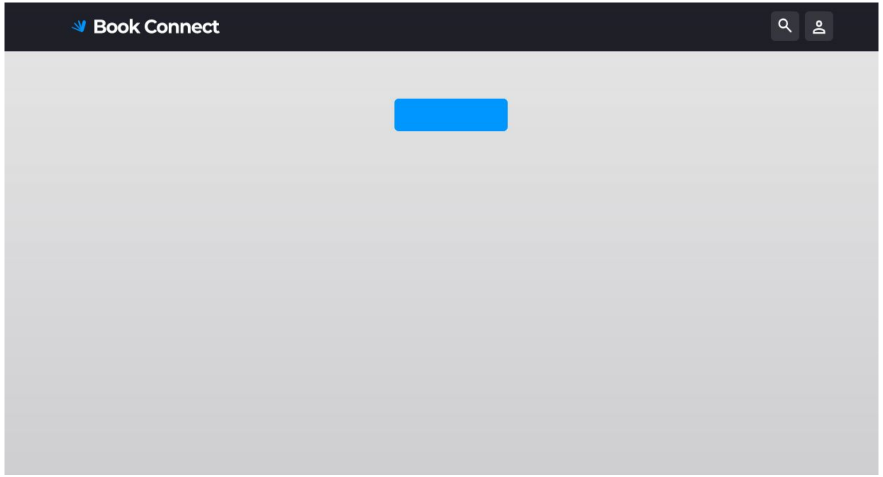
The Book Connect online book store allows users to search for their favourite books by either specifying the book title, author , genre or description into the search functionality. This will show a summary of the results of the search. Other additions are the dark mode or light mode that changes the theme of the online store. We also allow users to search for books by using a phrase that will be matched with the description or title.

Problems to solve

1 There was no interactive functionality on the book-connect web application

2 The scripts.js and data.js files had a few errors and missing components

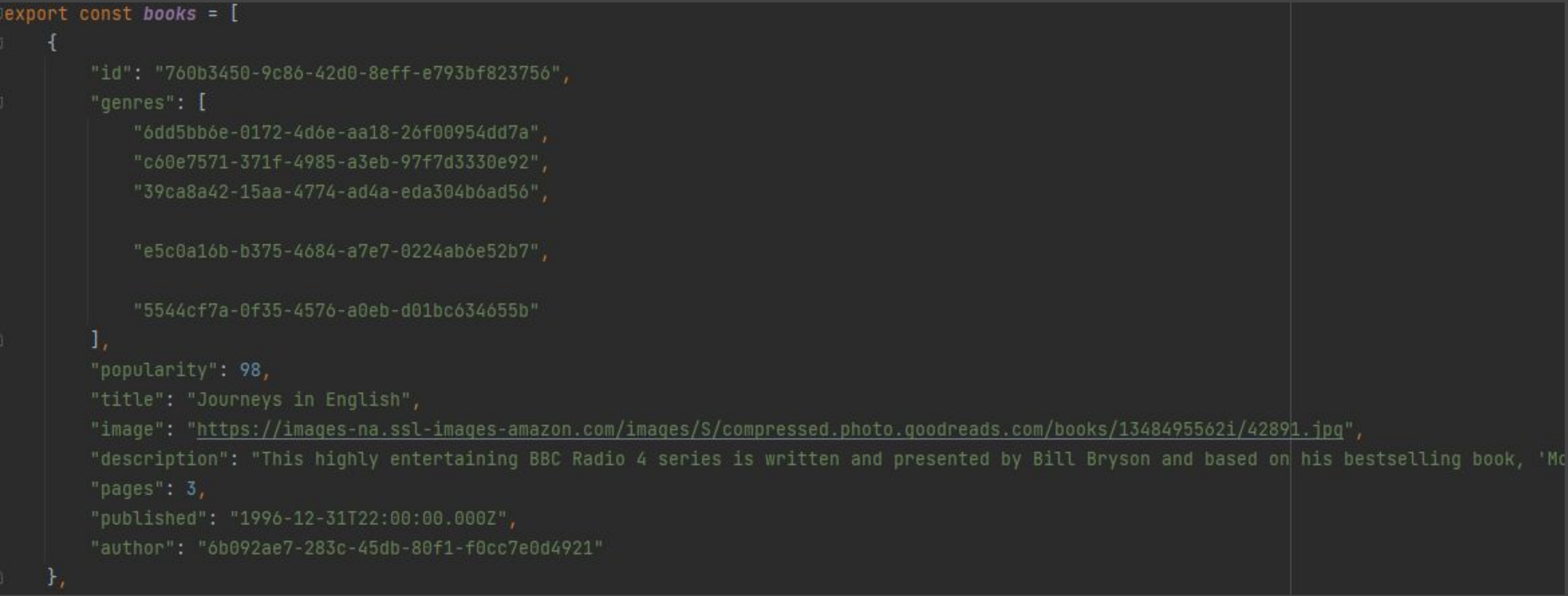
3 No books were displayed on the landing page of book-connect



Data.js Debugging

- Added “export” keyword before each of the objects in the data.js file
- Objects: authors, genres and books
- Constant: BOOKS_PER_PAGE

Books Object

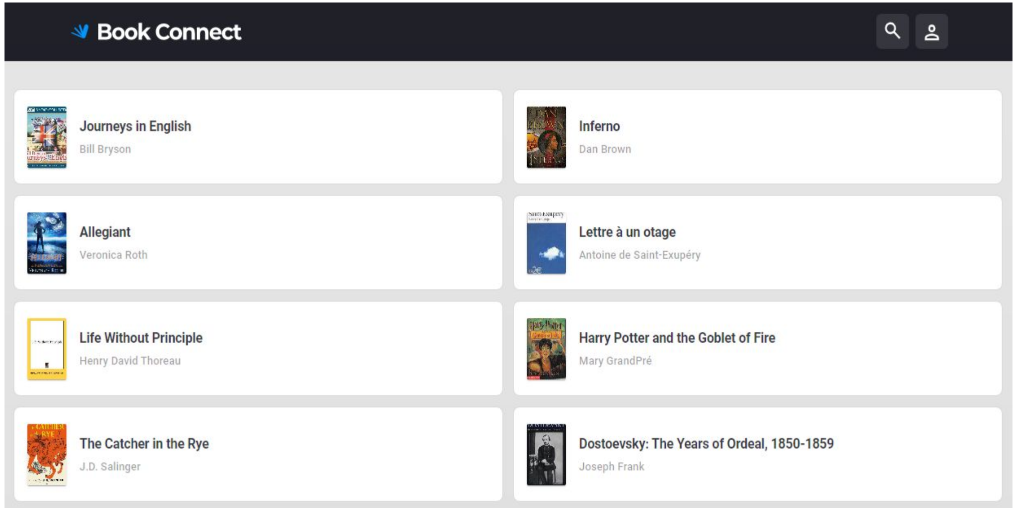


```
export const books = [
    {
        "id": "760b3450-9c86-42d0-8eff-e793bf823756",
        "genres": [
            "6dd5bb6e-0172-4d6e-aa18-26f00954dd7a",
            "c60e7571-371f-4985-a3eb-97f7d3330e92",
            "39ca8a42-15aa-4774-ad4a-eda304b6ad56",

            "e5c0a16b-b375-4684-a7e7-0224ab6e52b7",

            "5544cf7a-0f35-4576-a0eb-d01bc634655b"
        ],
        "popularity": 98,
        "title": "Journeys in English",
        "image": "https://images-na.ssl-images-amazon.com/images/S/compressed.photo.goodreads.com/books/1348495562i/42891.jpg",
        "description": "This highly entertaining BBC Radio 4 series is written and presented by Bill Bryson and based on his bestselling book, 'Mo
        "pages": 3,
        "published": "1996-12-31T22:00:00.000Z",
        "author": "6b092ae7-283c-45db-80f1-f0cc7e0d4921"
    },
```

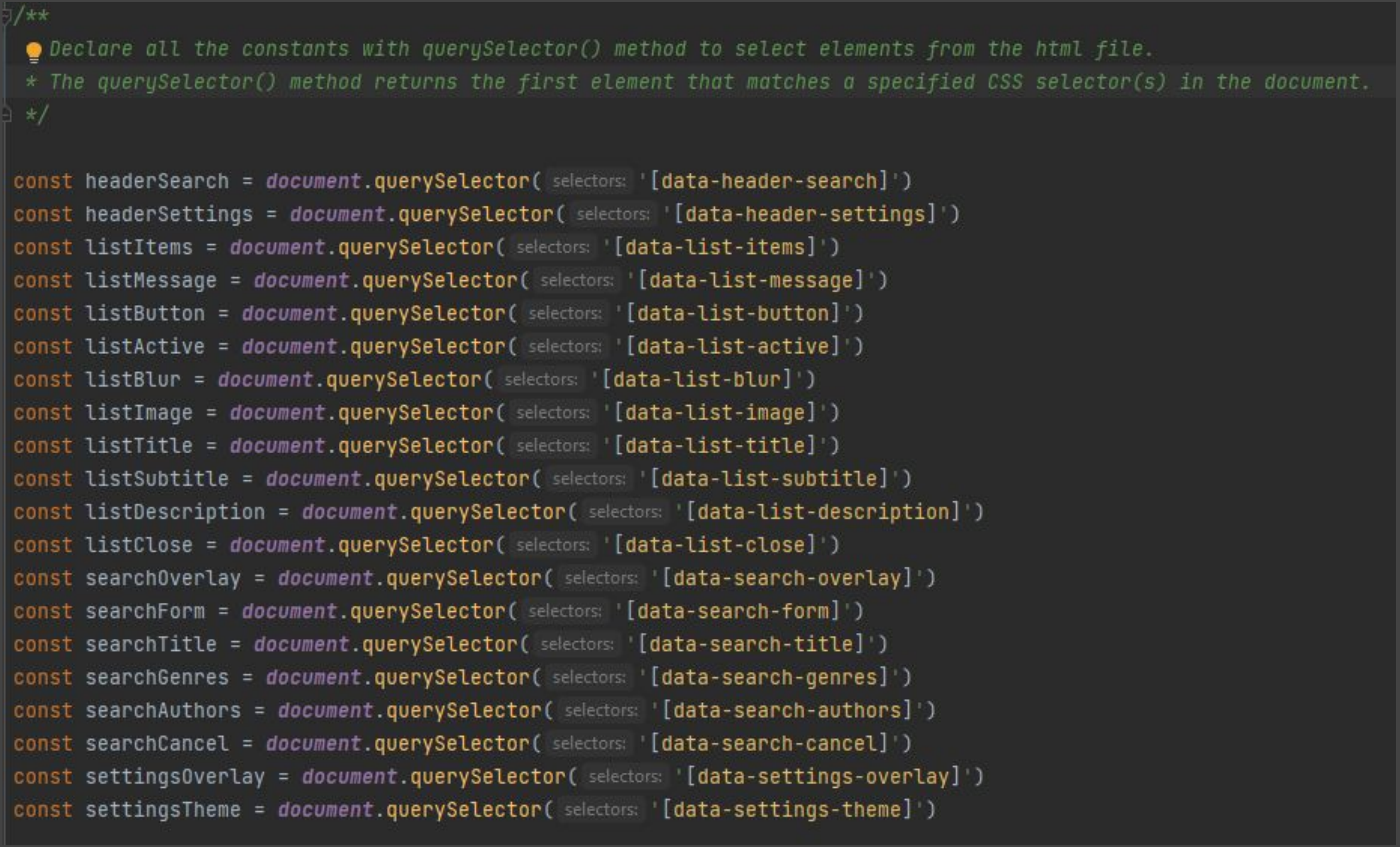
Debugging scripts.js

Methodology

- Principles of the Document Object Model (DOM)
- Used Query Selectors for interactive behaviour
- Created functions that serve as underlying logic



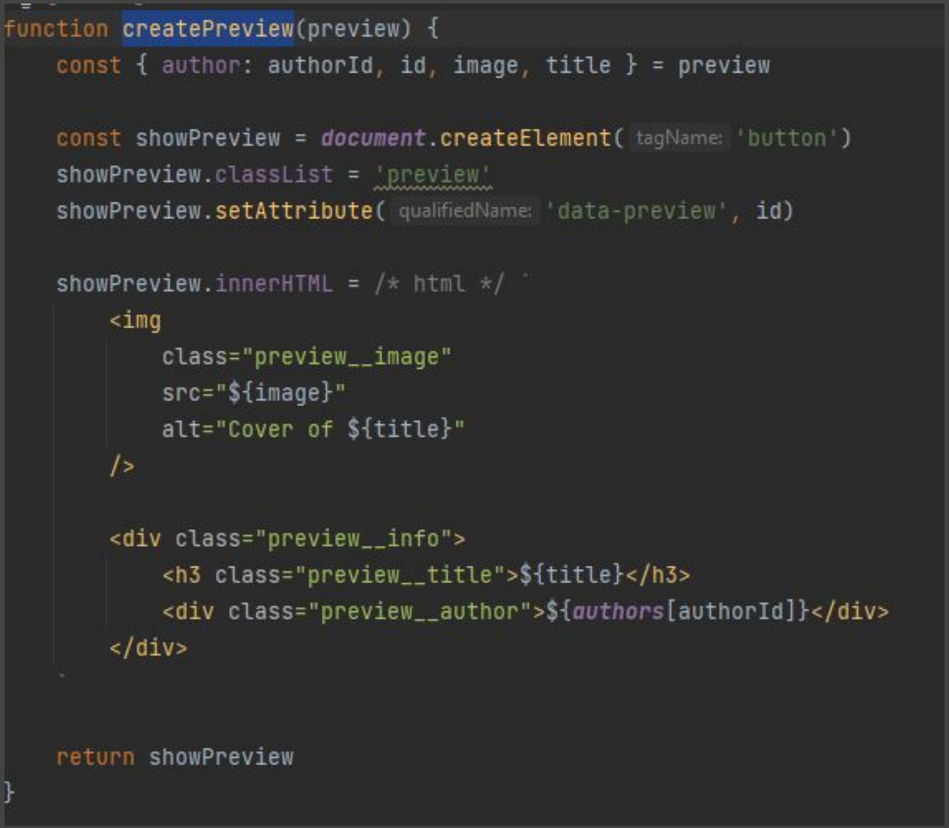
Journeys in English Bill Bryson	Inferno Dan Brown
Allegiant Veronica Roth	Lettre à un otage Antoine de Saint-Exupéry
Life Without Principle Henry David Thoreau	Harry Potter and the Goblet of Fire Mary GrandPré
The Catcher in the Rye J.D. Salinger	Dostoevsky: The Years of Ordeal, 1850-1859 Joseph Frank



```javascript
/**
 Declare all the constants with querySelector() method to select elements from the html file.
 * The querySelector() method returns the first element that matches a specified CSS selector(s) in the document.
 */

const headerSearch = document.querySelector( selectors: '[data-header-search]')
const headerSettings = document.querySelector( selectors: '[data-header-settings]')
const listItems = document.querySelector( selectors: '[data-list-items]')
const listMessage = document.querySelector( selectors: '[data-list-message]')
const listButton = document.querySelector( selectors: '[data-list-button]')
const listActive = document.querySelector( selectors: '[data-list-active]')
const listBlur = document.querySelector( selectors: '[data-list-blur]')
const listImage = document.querySelector( selectors: '[data-list-image]')
const listTitle = document.querySelector( selectors: '[data-list-title]')
const listSubtitle = document.querySelector( selectors: '[data-list-subtitle]')
const listDescription = document.querySelector( selectors: '[data-list-description]')
const listClose = document.querySelector( selectors: '[data-list-close]')
const searchOverlay = document.querySelector( selectors: '[data-search-overlay]')
const searchForm = document.querySelector( selectors: '[data-search-form]')
const searchTitle = document.querySelector( selectors: '[data-search-title]')
const searchGenres = document.querySelector( selectors: '[data-search-genres]')
const searchAuthors = document.querySelector( selectors: '[data-search-authors]')
const searchCancel = document.querySelector( selectors: '[data-search-cancel]')
const settingsOverlay = document.querySelector( selectors: '[data-settings-overlay]')
const settingsTheme = document.querySelector( selectors: '[data-settings-theme]')
```

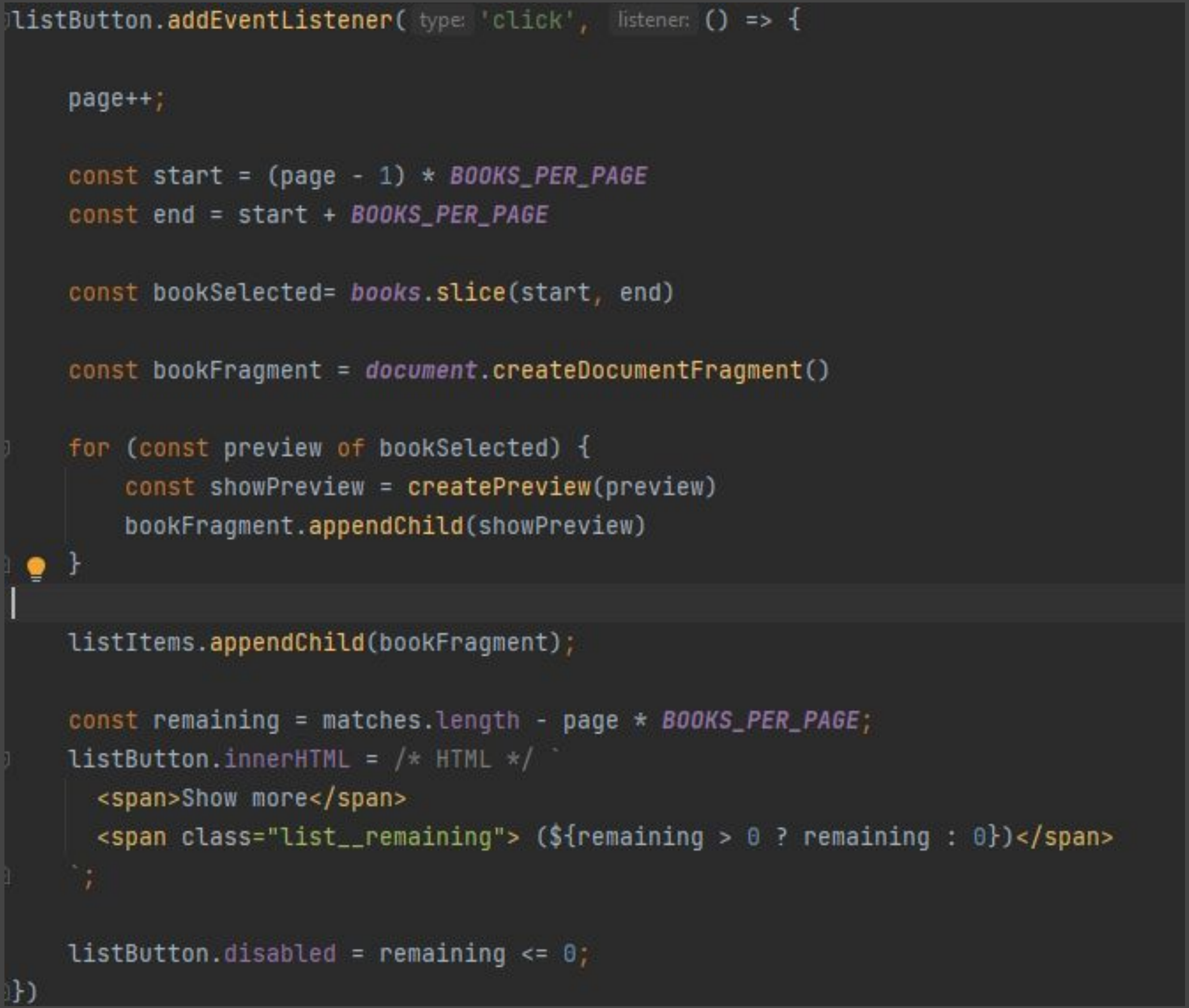
Proposed Functions & Event Listeners

The createPreview function takes a book preview object and returns a button element containing the book details.



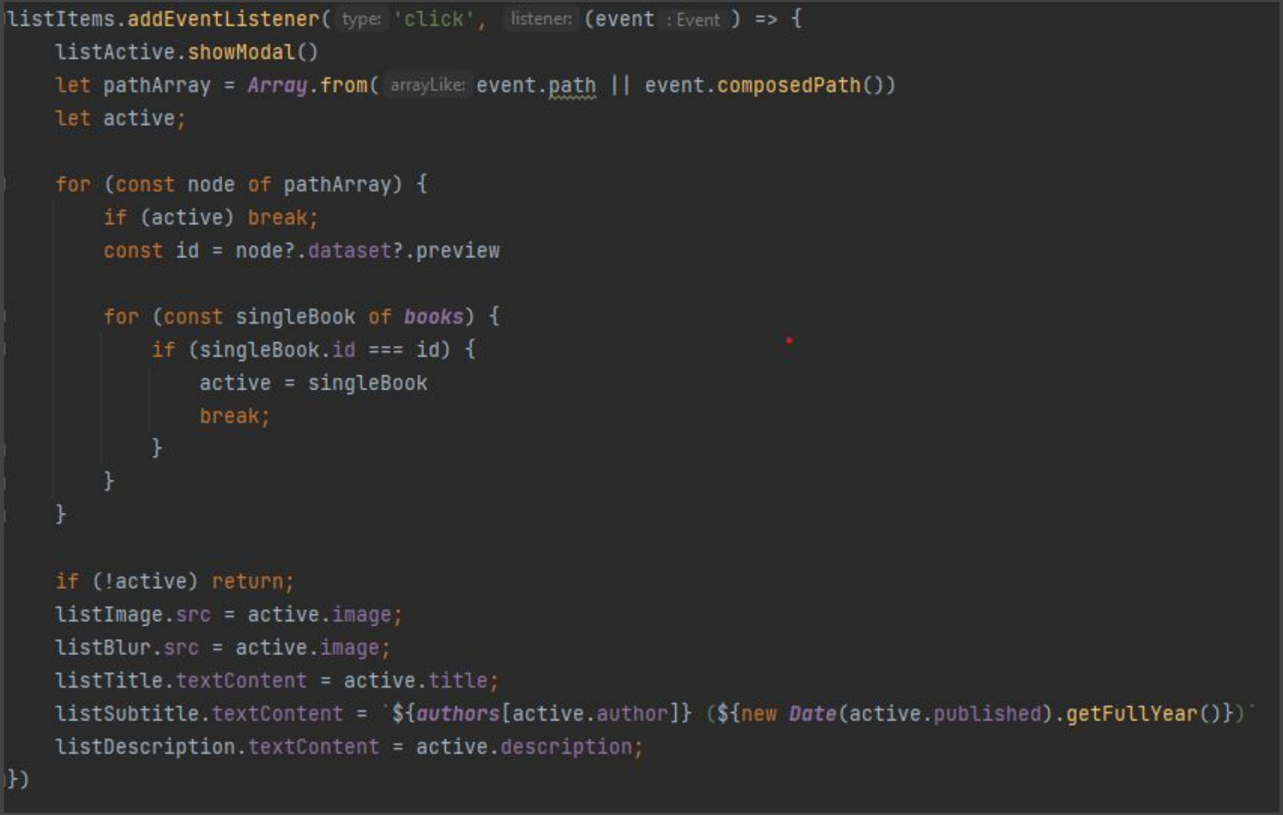
```
function createPreview(preview) {
    const { author: authorId, id, image, title } = preview

    const showPreview = document.createElement( tagName: 'button')
    showPreview.classList = 'preview'
    showPreview.setAttribute( qualifiedName: 'data-preview', id)

    showPreview.innerHTML = /* html */ `
        <img
            class="preview__image"
            src="${image}"
            alt="Cover of ${title}"
        />

        <div class="preview__info">
            <h3 class="preview__title">${title}</h3>
            <div class="preview__author">${authors[authorId]}</div>
        </div>
    `

    return showPreview
}
```

Allow user to view more books when the maximum 36 books per page is reached



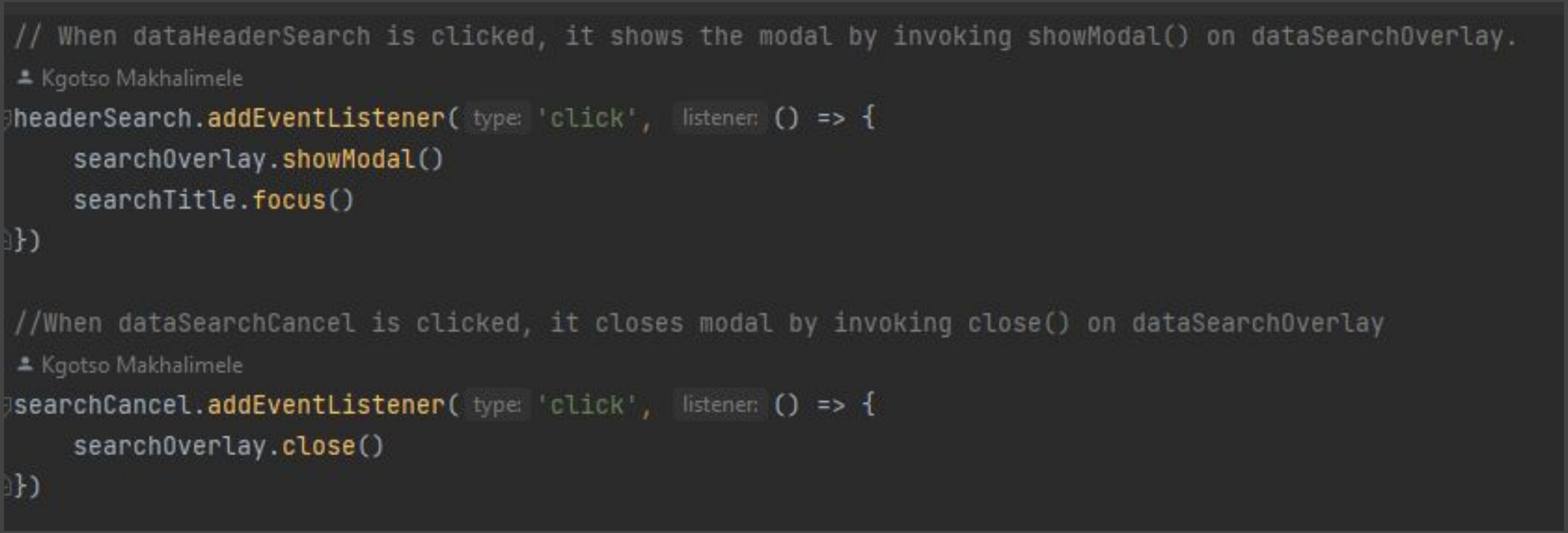
```
listButton.addEventListener( type: 'click',  listener: () => {

    page++;

    const start = (page - 1) * BOOKS_PER_PAGE
    const end = start + BOOKS_PER_PAGE

    const bookSelected= books.slice(start, end)

    const bookFragment = document.createDocumentFragment()

    for (const preview of bookSelected) {
        const showPreview = createPreview(preview)
        bookFragment.appendChild(showPreview)
    }

    listItems.appendChild(bookFragment);

    const remaining = matches.length - page * BOOKS_PER_PAGE;
    listButton.innerHTML = /* HTML */ `
      <span>Show more</span>
      <span class="list__remaining"> (${remaining > 0 ? remaining : 0})</span>
    `;

    listButton.disabled = remaining <= 0;
})
```

When the user clicks on a book, they are shown a preview with the book cover, title, author and the year of publication



```
listItems.addEventListener( type: 'click',  listener: (event : Event ) => {
    listActive.showModal()
    let pathArray = Array.from( arrayLike: event.path || event.composedPath())
    let active;

    for (const node of pathArray) {
        if (active) break;
        const id = node?.dataset?.preview

        for (const singleBook of books) {
            if (singleBook.id === id) {
                active = singleBook
                break;
            }
        }
    }

    if (!active) return;
    listImage.src = active.image;
    listBlur.src = active.image;
    listTitle.textContent = active.title;
    listSubtitle.textContent = `${authors[active.author]} (${new Date(active.published).getFullYear()})`
    listDescription.textContent = active.description;
})
```

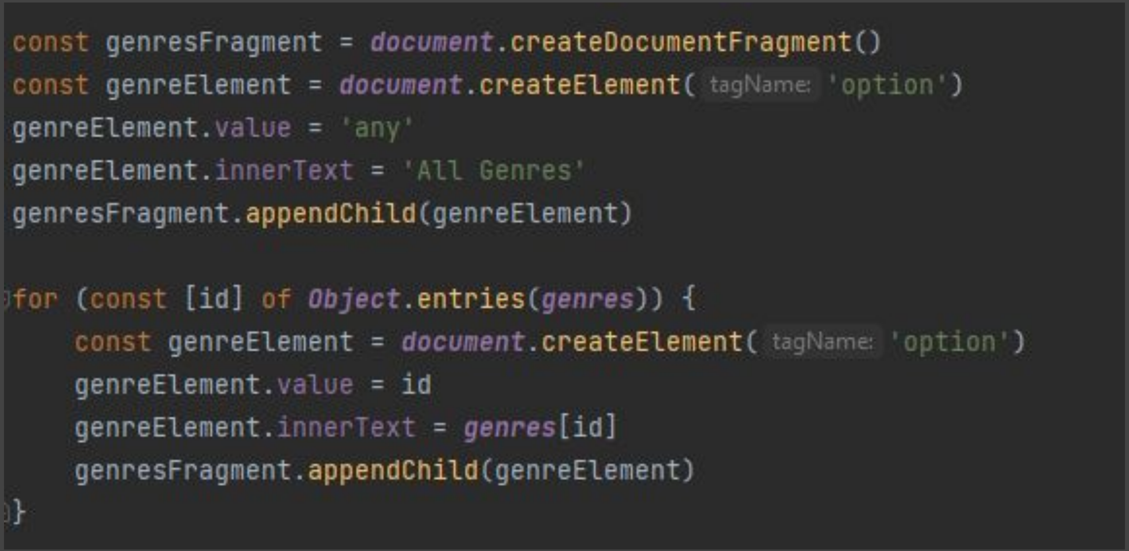
Search Functionality



```
// When dataHeaderSearch is clicked, it shows the modal by invoking showModal() on dataSearchOverlay.
Kgotso Makhalimele
headerSearch.addEventListener( type: 'click', listener: () => {
    searchOverlay.showModal()
    searchTitle.focus()
})

//When dataSearchCancel is clicked, it closes modal by invoking close() on dataSearchOverlay
Kgotso Makhalimele
searchCancel.addEventListener( type: 'click', listener: () => {
    searchOverlay.close()
})
```

Search by genre

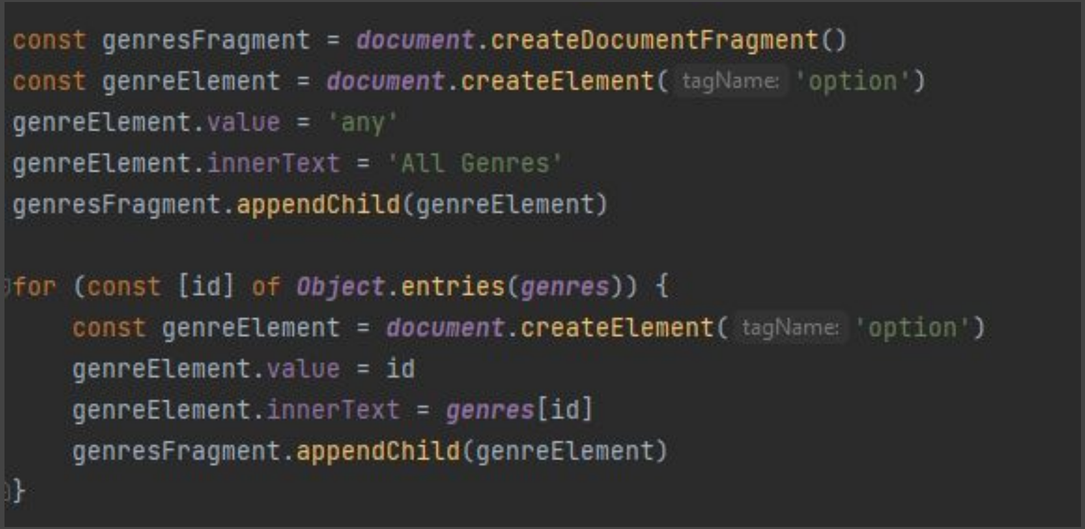
Access the genres and create a fragment listing all genres on the site



```
const genresFragment = document.createDocumentFragment()
const genreElement = document.createElement( tagName: 'option')
genreElement.value = 'any'
genreElement.innerText = 'All Genres'
genresFragment.appendChild(genreElement)

for (const [id] of Object.entries(genres)) {
    const genreElement = document.createElement( tagName: 'option')
    genreElement.value = id
    genreElement.innerText = genres[id]
    genresFragment.appendChild(genreElement)
}
```

Search by author

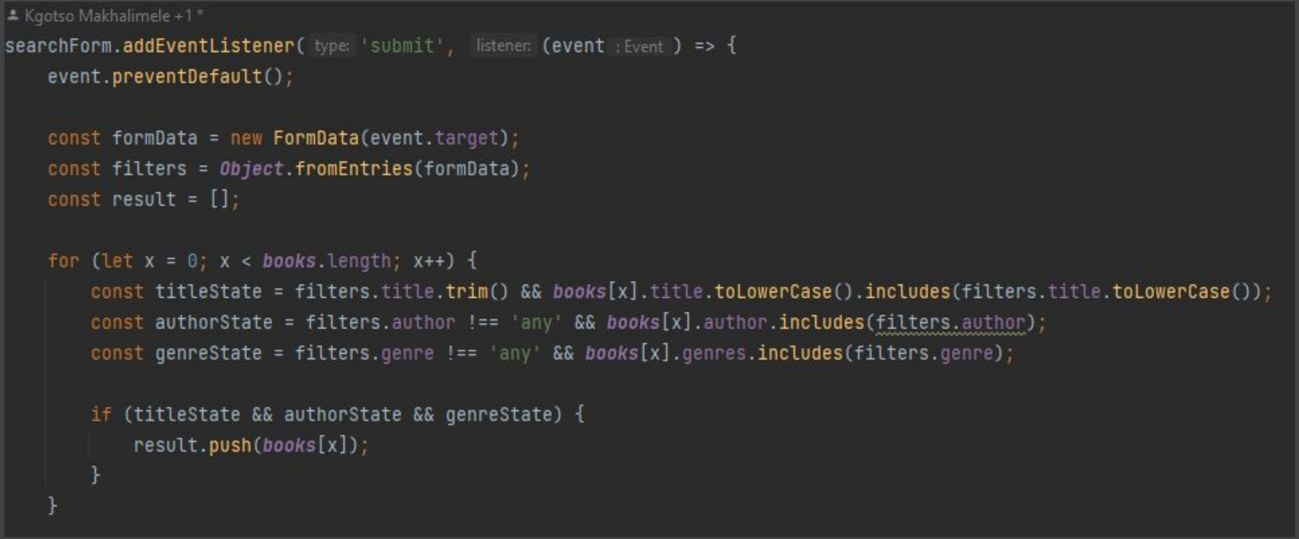
Access the authors object and create a link to the html options



```
const genresFragment = document.createDocumentFragment()
const genreElement = document.createElement( tagName: 'option')
genreElement.value = 'any'
genreElement.innerText = 'All Genres'
genresFragment.appendChild(genreElement)

for (const [id] of Object.entries(genres)) {
    const genreElement = document.createElement( tagName: 'option')
    genreElement.value = id
    genreElement.innerText = genres[id]
    genresFragment.appendChild(genreElement)
}
```

Submitting the search

Set an empty array to store search results and perform the search through an array of books to show relevant results

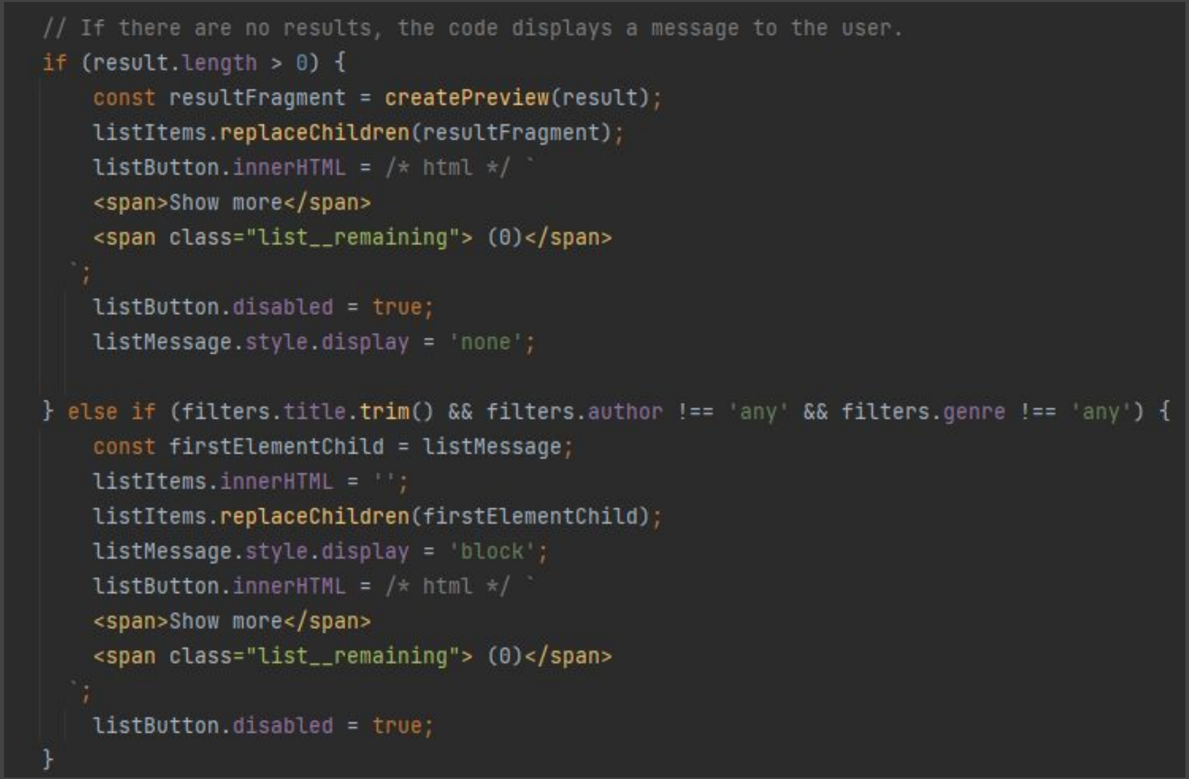


Kgotso Makhalimele +1 *

```
searchForm.addEventListener( type: 'submit',  listener: (event : Event ) => {
    event.preventDefault();

    const formData = new FormData(event.target);
    const filters = Object.fromEntries(formData);
    const result = [];

    for (let x = 0; x < books.length; x++) {
        const titleState = filters.title.trim() && books[x].title.toLowerCase().includes(filters.title.toLowerCase());
        const authorState = filters.author !== 'any' && books[x].author.includes(filters.author);
        const genreState = filters.genre !== 'any' && books[x].genres.includes(filters.genre);

        if (titleState && authorState && genreState) {
            result.push(books[x]);
        }
    }
```

If there are no results, return a message alerting the user.

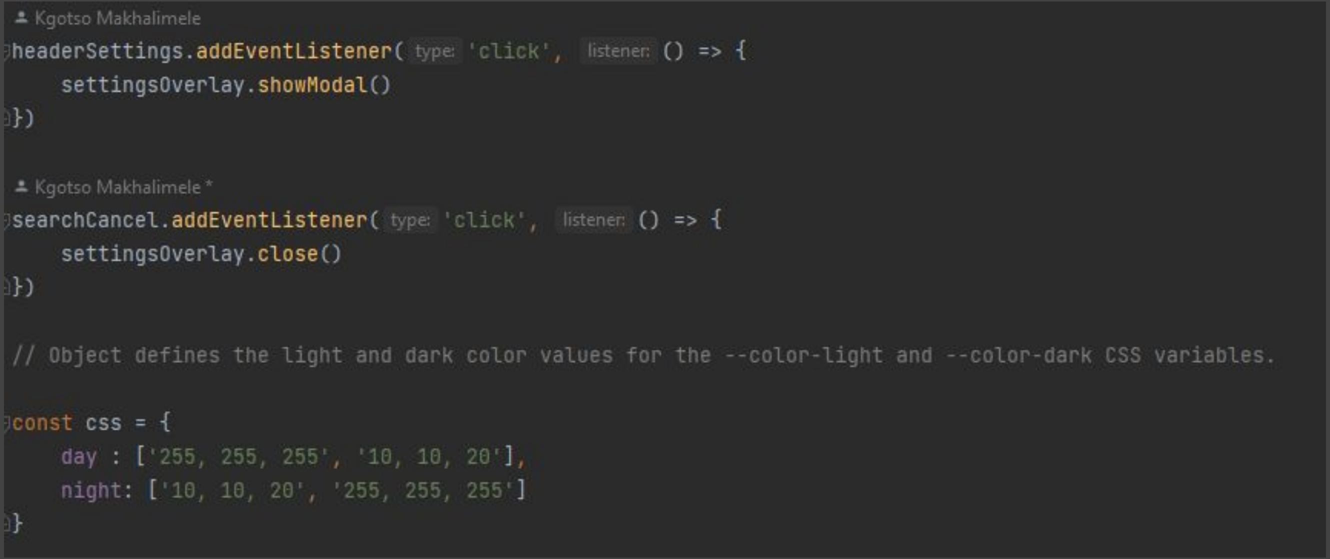
If the search returns results, display them to the user



```
// If there are no results, the code displays a message to the user.
if (result.length > 0) {
    const resultFragment = createPreview(result);
    listItems.replaceChildren(resultFragment);
    listButton.innerHTML = /* html */ `
    <span>Show more</span>
    <span class="list__remaining"> (0)</span>
  `;
    listButton.disabled = true;
    listMessage.style.display = 'none';

} else if (filters.title.trim() && filters.author !== 'any' && filters.genre !== 'any') {
    const firstElementChild = listMessage;
    listItems.innerHTML = '';
    listItems.replaceChildren(firstElementChild);
    listMessage.style.display = 'block';
    listButton.innerHTML = /* html */ `
    <span>Show more</span>
    <span class="list__remaining"> (0)</span>
  `;
    listButton.disabled = true;
}
```

Selecting a theme

Users have a choice between the day and night themes



Kgotso Makhalimele

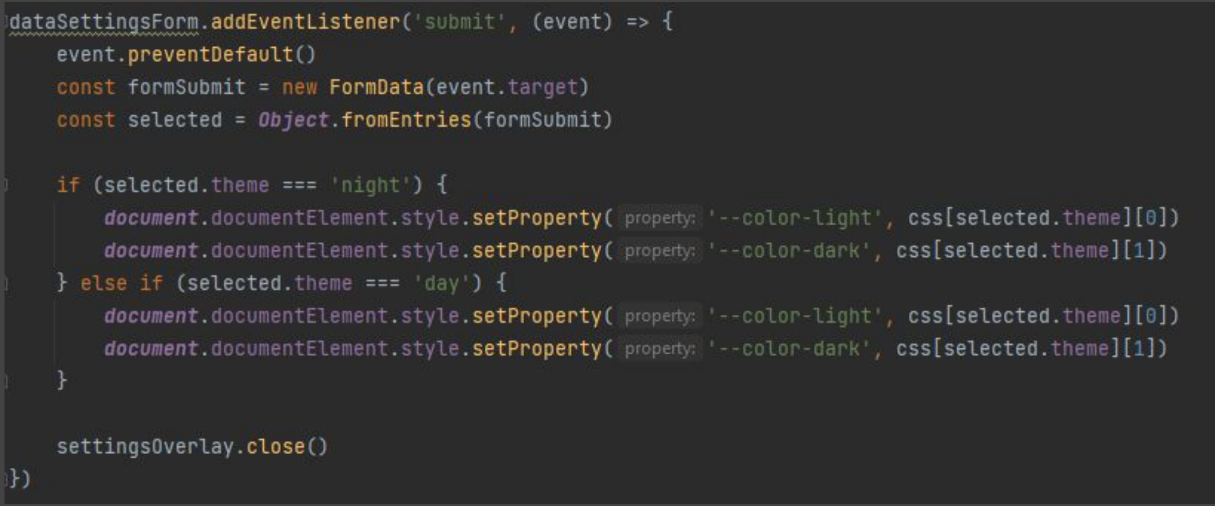
```
headerSettings.addEventListener( type: 'click',  listener: () => {
    settingsOverlay.showModal()
})
```

Kgotso Makhalimele *

```
searchCancel.addEventListener( type: 'click',  listener: () => {
    settingsOverlay.close()
})

// Object defines the light and dark color values for the --color-light and --color-dark CSS variables.

const css = {
    day : ['255, 255, 255', '10, 10, 20'],
    night: ['10, 10, 20', '255, 255, 255']
}
```

Actioning the theme selection

Event listener to change theme when the submit button is clicked.



```
dataSettingsForm.addEventListener('submit', (event) => {
    event.preventDefault()
    const formSubmit = new FormData(event.target)
    const selected = Object.fromEntries(formSubmit)

    if (selected.theme === 'night') {
        document.documentElement.style.setProperty( property: '--color-light', css[selected.theme][0])
        document.documentElement.style.setProperty( property: '--color-dark', css[selected.theme][1])
    } else if (selected.theme === 'day') {
        document.documentElement.style.setProperty( property: '--color-light', css[selected.theme][0])
        document.documentElement.style.setProperty( property: '--color-dark', css[selected.theme][1])
    }

    settingsOverlay.close()
})
```

Thank you.