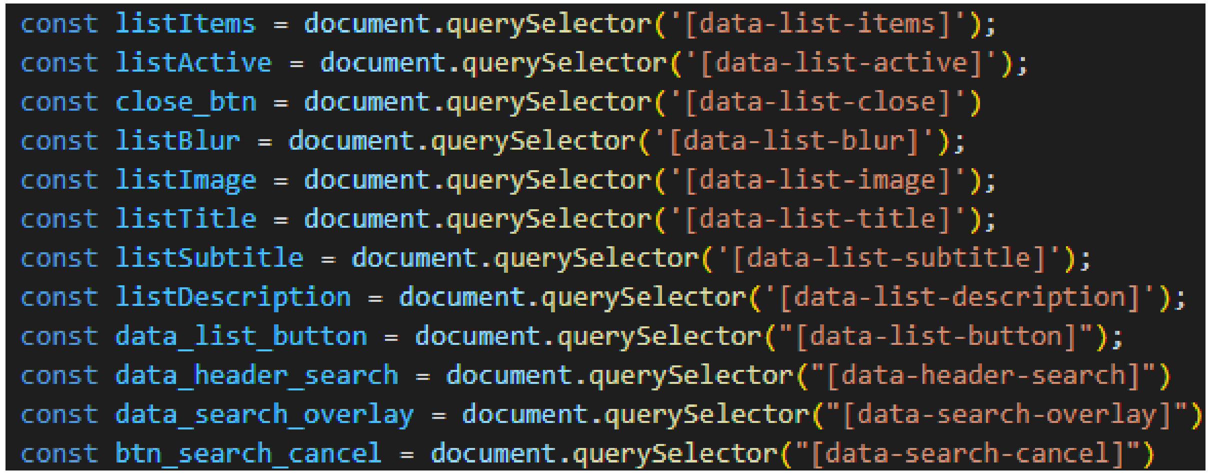
BOOK CONNECT

Code Changes

The first thing I did was to call all the dom elements I will need throughout the codebase

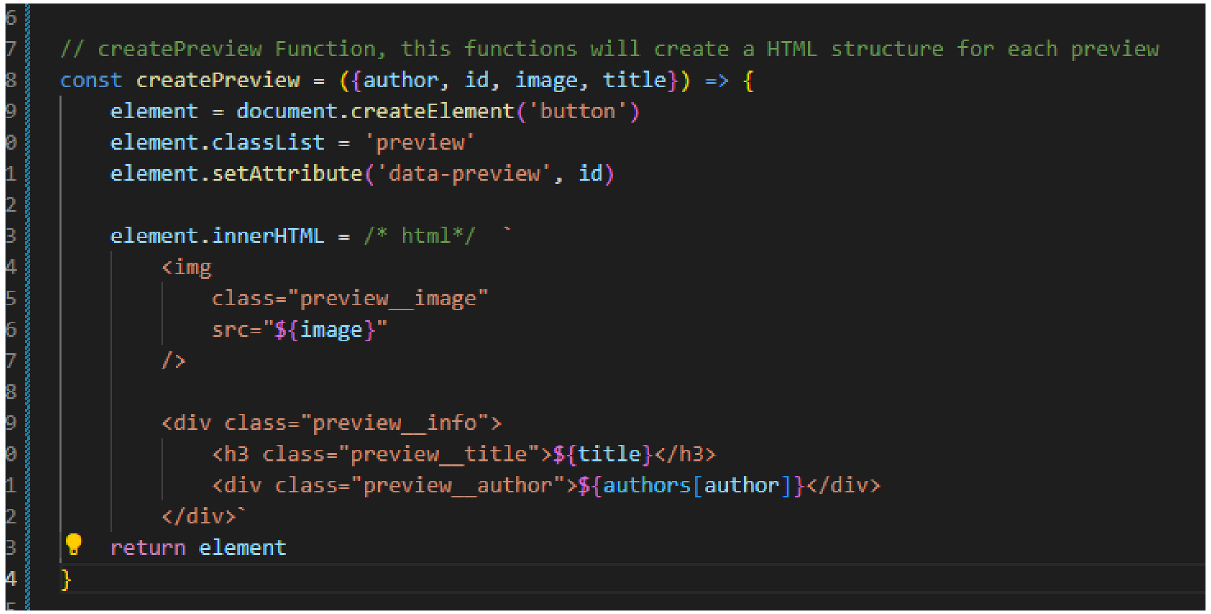
The Reason I did this is because initially they were not called correctly,

Plus we get to call the element once and then we can use it many times instead of calling it from html document whenever we need it

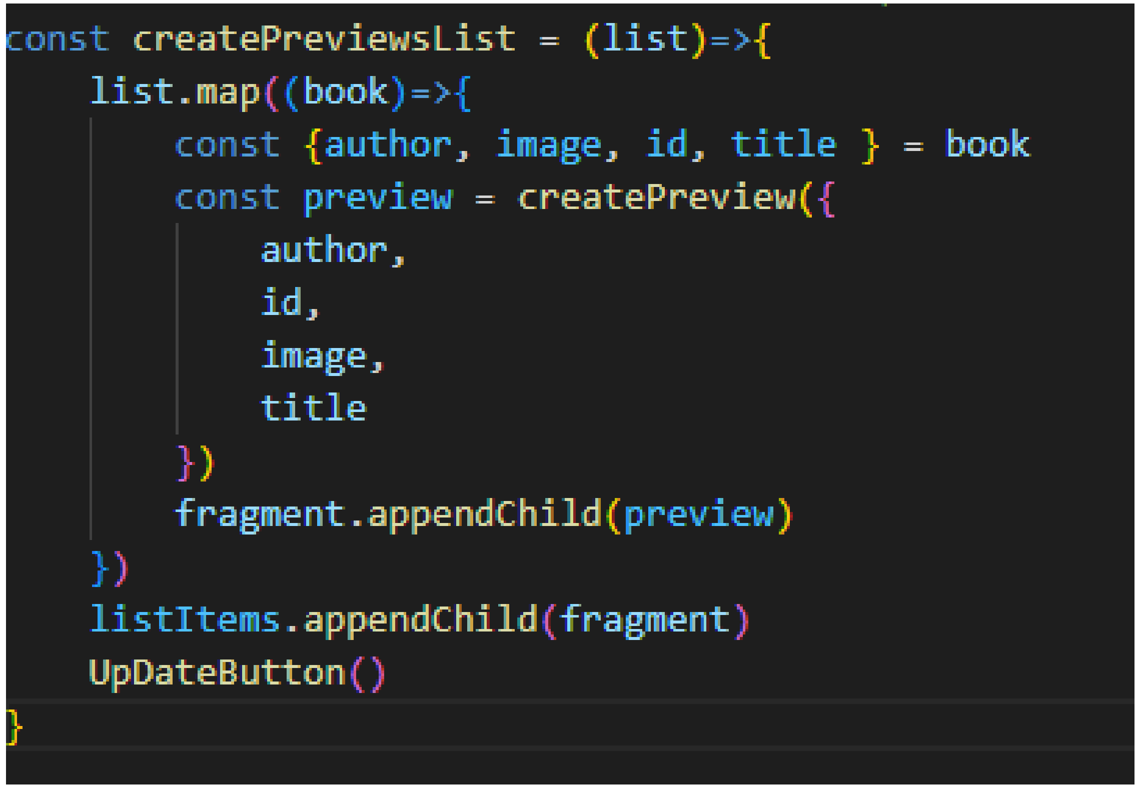


```
const listItems = document.querySelector('[data-list-items]');
const listActive = document.querySelector('[data-list-active]');
const close_btn = document.querySelector('[data-list-close]')
const listBlur = document.querySelector('[data-list-blur]');
const listImage = document.querySelector('[data-list-image]');
const listTitle = document.querySelector('[data-list-title]');
const listSubtitle = document.querySelector('[data-list-subtitle]');
const listDescription = document.querySelector('[data-list-description]');
const data_list_button = document.querySelector("[data-list-button]");
const data_header_search = document.querySelector("[data-header-search]")
const data_search_overlay = document.querySelector("[data-search-overlay]")
const btn_search_cancel = document.querySelector("[data-search-cancel]")
```

1) I decided to add the createPreview Function, that will take a book object as a parameter and create a html element for it.



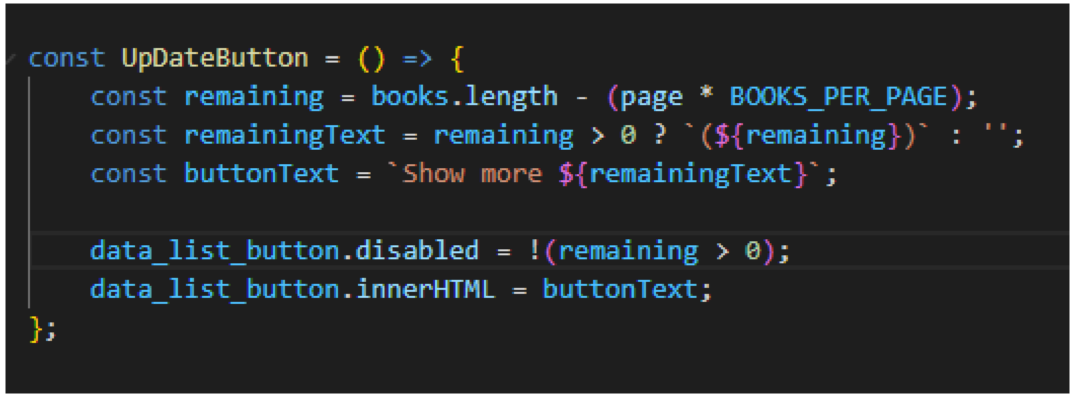
```
// createPreview Function, this functions will create a HTML structure for each preview
const createPreview = ({author, id, image, title}) => {
    element = document.createElement('button')
    element.classList = 'preview'
    element.setAttribute('data-preview', id)

    element.innerHTML = /* html*/  `
        <img
            class="preview__image"
            src="${image}"
        />

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



```
const createPreviewsList = (list)=>{
    list.map((book)=>{
        const {author, image, id, title } = book
        const preview = createPreview({
            author,
            id,
            image,
            title
        })
        fragment.appendChild(preview)
    })
    listItems.appendChild(fragment)
    UpDateButton()
}
```

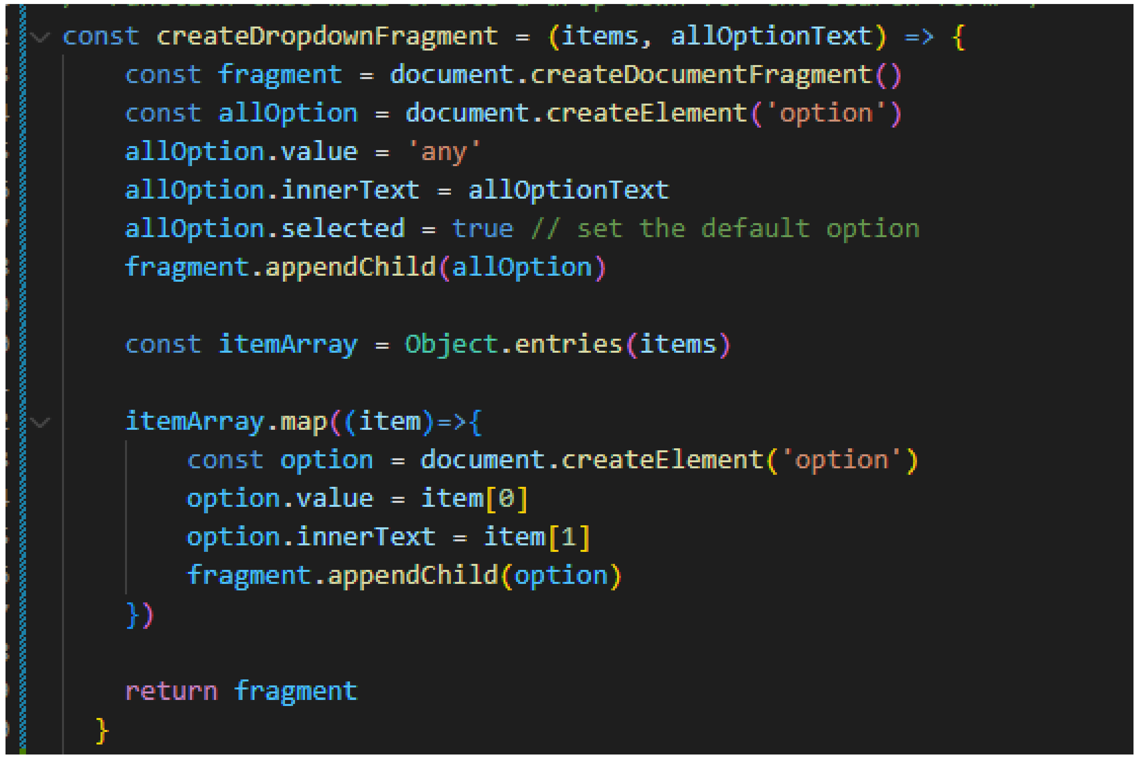
CreatePreviewList function takes on a array of books as an argument.   It calls the createPreview function to create the html element for preview, and after that it appends the element as a child of the listItems element.

Reason split functionalities to different functions is that it makes the code more dynamic and cleaner



```
const UpDateButton = () => {
    const remaining = books.length - (page * BOOKS_PER_PAGE);
    const remainingText = remaining > 0 ? `(${remaining})` : '';
    const buttonText = `Show more ${remainingText}`;

    data_list_button.disabled = !(remaining > 0);
    data_list_button.innerHTML = buttonText;
};
```

This function updates the number of show more the remaining books

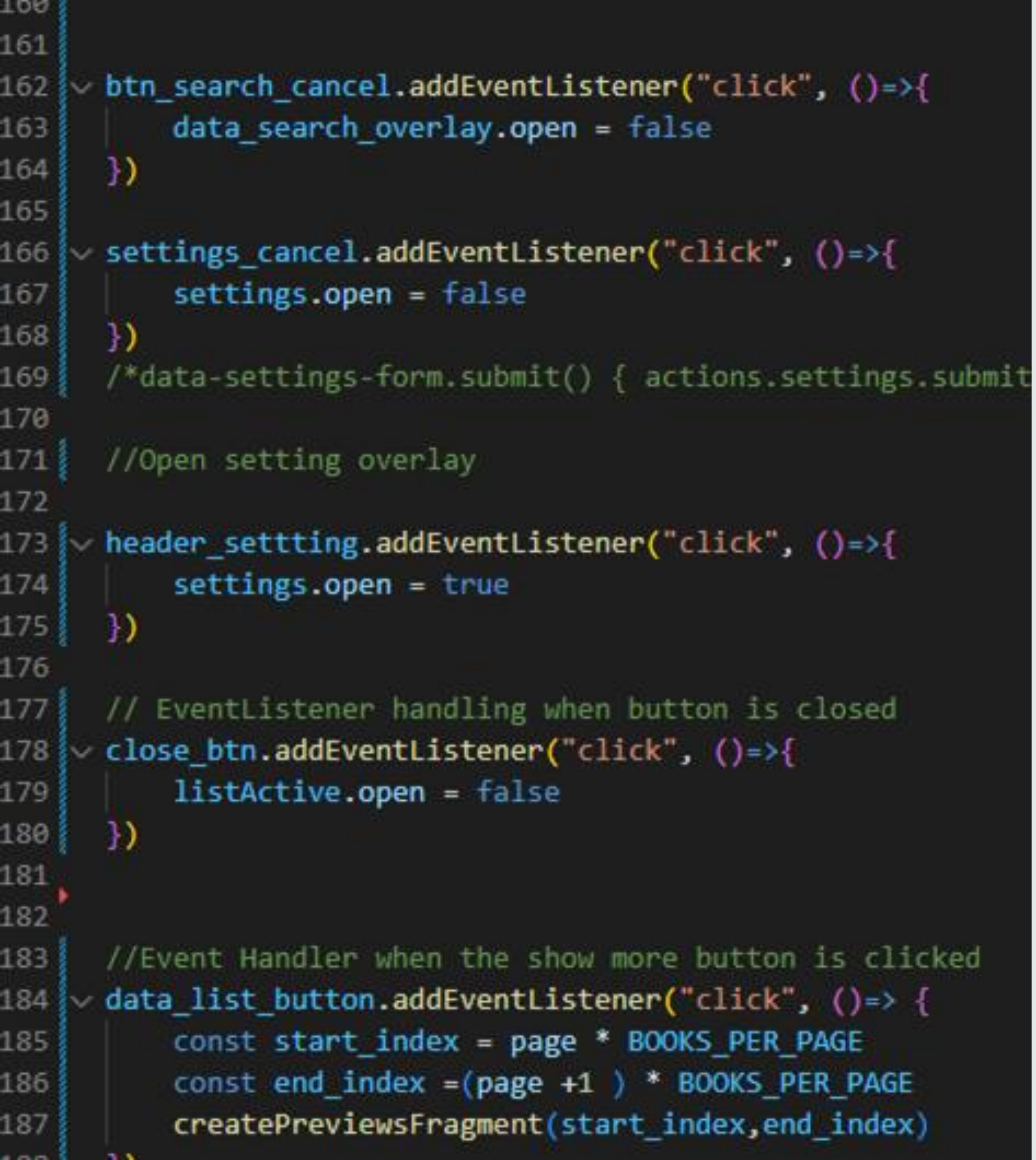


```
const createDropdownFragment = (items, allOptionText) => {
    const fragment = document.createDocumentFragment()
    const allOption = document.createElement('option')
    allOption.value = 'any'
    allOption.innerText = allOptionText
    allOption.selected = true // set the default option
    fragment.appendChild(allOption)

    const itemArray = Object.entries(items)

    itemArray.map((item)=>{
        const option = document.createElement('option')
        option.value = item[0]
        option.innerText = item[1]
        fragment.appendChild(option)
    })

    return fragment
  }
```

This function I created is responsible for creating the options for the dropdown of authors and genres.

It takes on two arguments Items which could be an array of authors or an array of genres, it takes on a string which will tell us we are dealing with genres or authors.

Reason being it makes our code smaller and cleaner this is why I have created a function for both



```
btn_search_cancel.addEventListener("click", ()=>{
    data_search_overlay.open = false
})

settings_cancel.addEventListener("click", ()=>{
    settings.open = false
})
/*data-settings-form.submit() { actions.settings.submit

//Open setting overlay

header_setting.addEventListener("click", ()=>{
    settings.open = true
})

// EventListener handling when button is closed
close_btn.addEventListener("click", ()=>{
    listActive.open = false
})


//Event Handler when the show more button is clicked
data_list_button.addEventListener("click", ()=> {
    const start_index = page * BOOKS_PER_PAGE
    const end_index =(page +1 ) * BOOKS_PER_PAGE
    createPreviewsFragment(start_index,end_index)
```

All the event listeners were created incorrectly and I fixed all of them by implementing as shown above.

In closing Javascript in this file was used incorrectly and the 4loops were not implemented properly, the dom elements were not called from the html . The eventlisteners were not implemented.