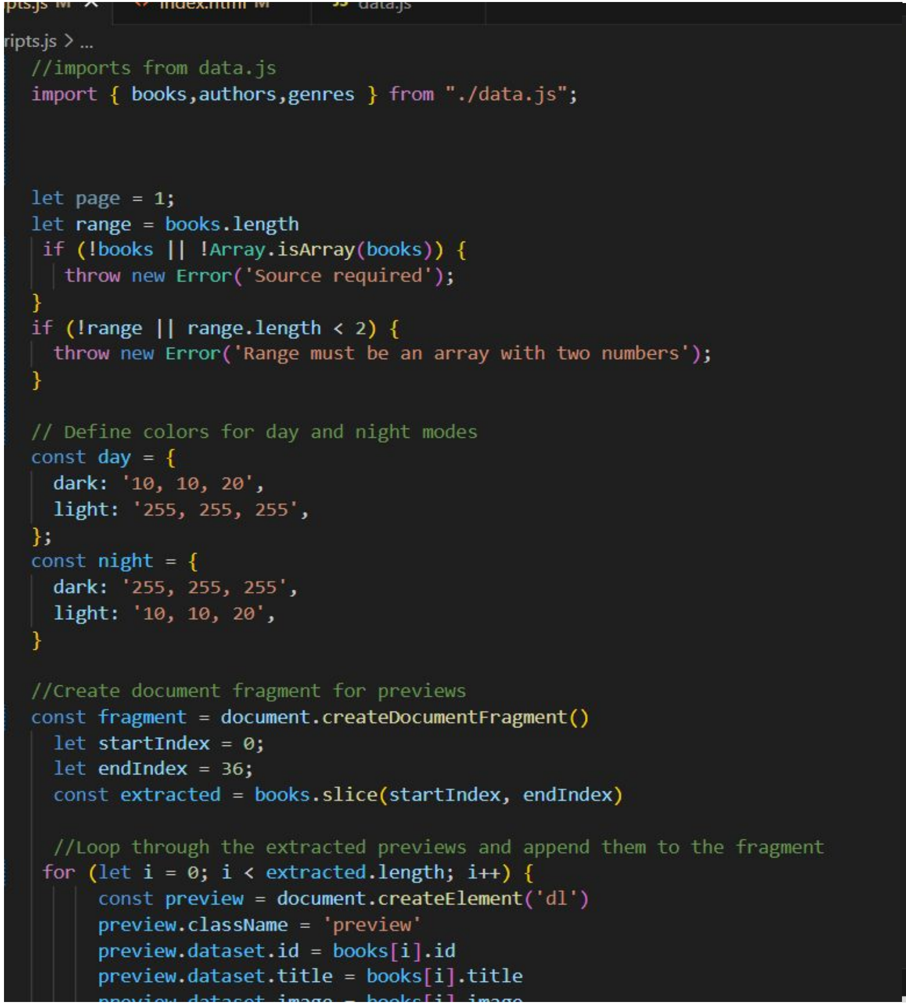
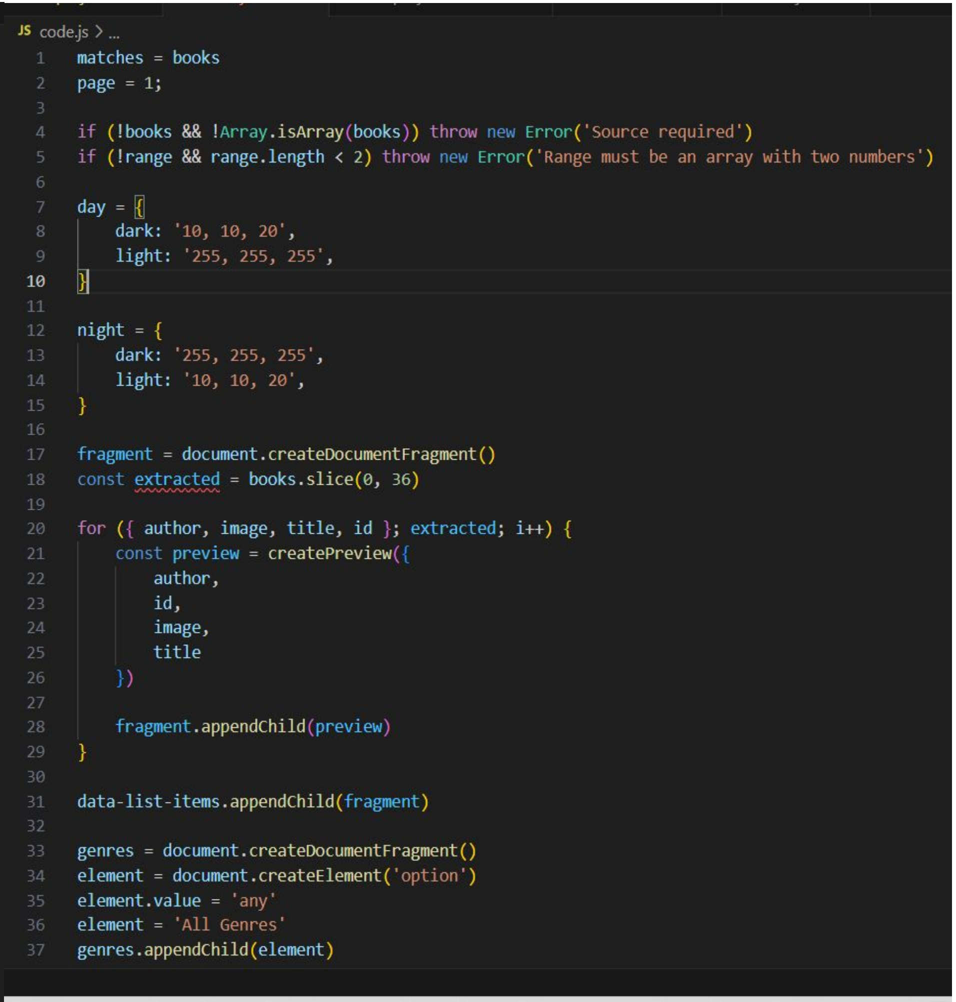
Book-Connect

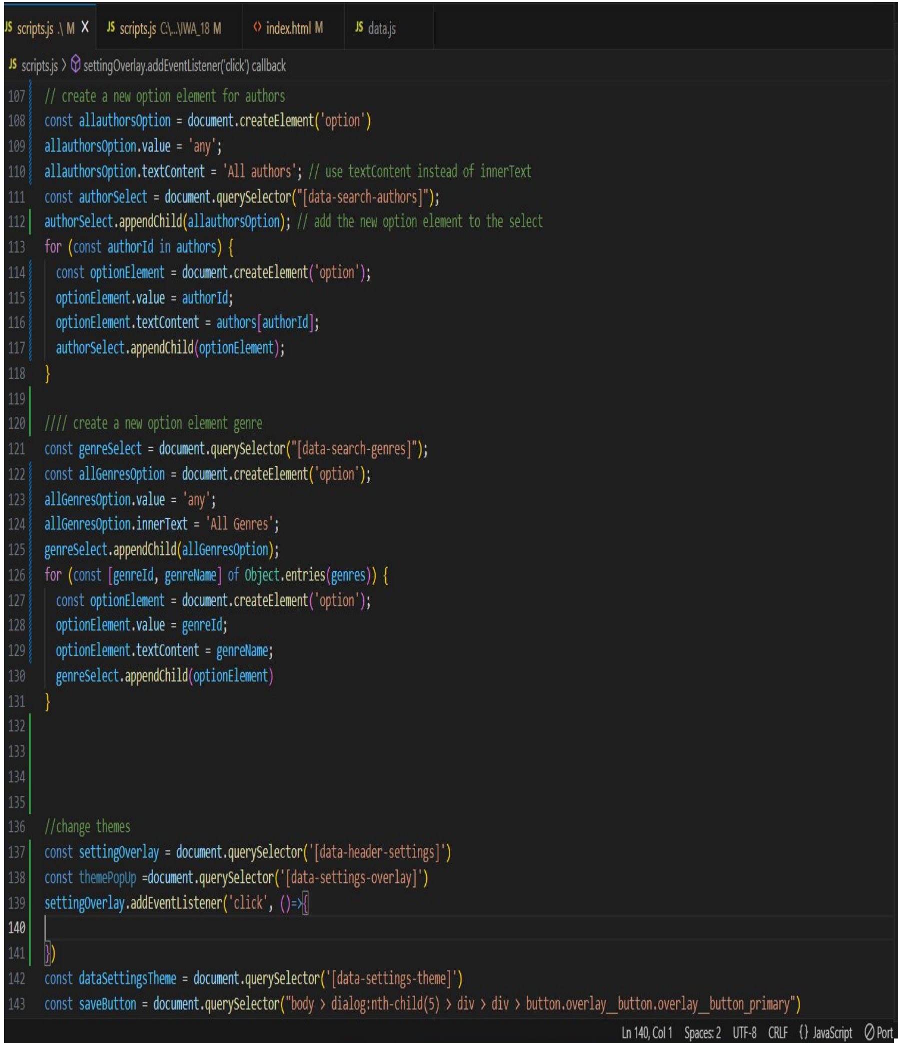
IWA19 Presentation



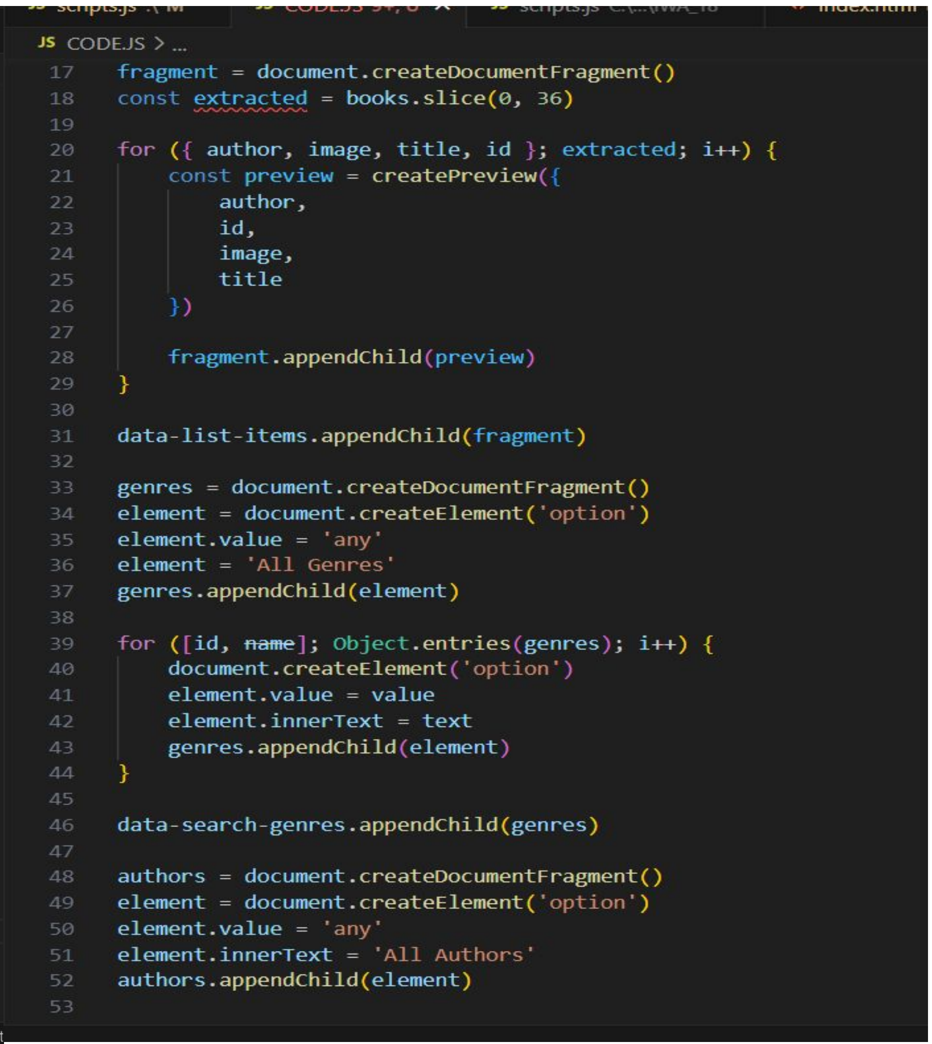
```
//imports from data.js
import { books,authors,genres } from "./data.js";



let page = 1;
let range = books.length
 if (!books || !Array.isArray(books)) {
   throw new Error('Source required');
}
if (!range || range.length < 2) {
  throw new Error('Range must be an array with two numbers');
}

// Define colors for day and night modes
const day = {
  dark: '10, 10, 20',
  light: '255, 255, 255',
};
const night = {
  dark: '255, 255, 255',
  light: '10, 10, 20',
}

//Create document fragment for previews
const fragment = document.createDocumentFragment()
  let startIndex = 0;
  let endIndex = 36;
  const extracted = books.slice(startIndex, endIndex)

  //Loop through the extracted previews and append them to the fragment
 for (let i = 0; i < extracted.length; i++) {
      const preview = document.createElement('dl')
      preview.className = 'preview'
      preview.dataset.id = books[i].id
      preview.dataset.title = books[i].title
      preview.dataset.image = books[i].image
```



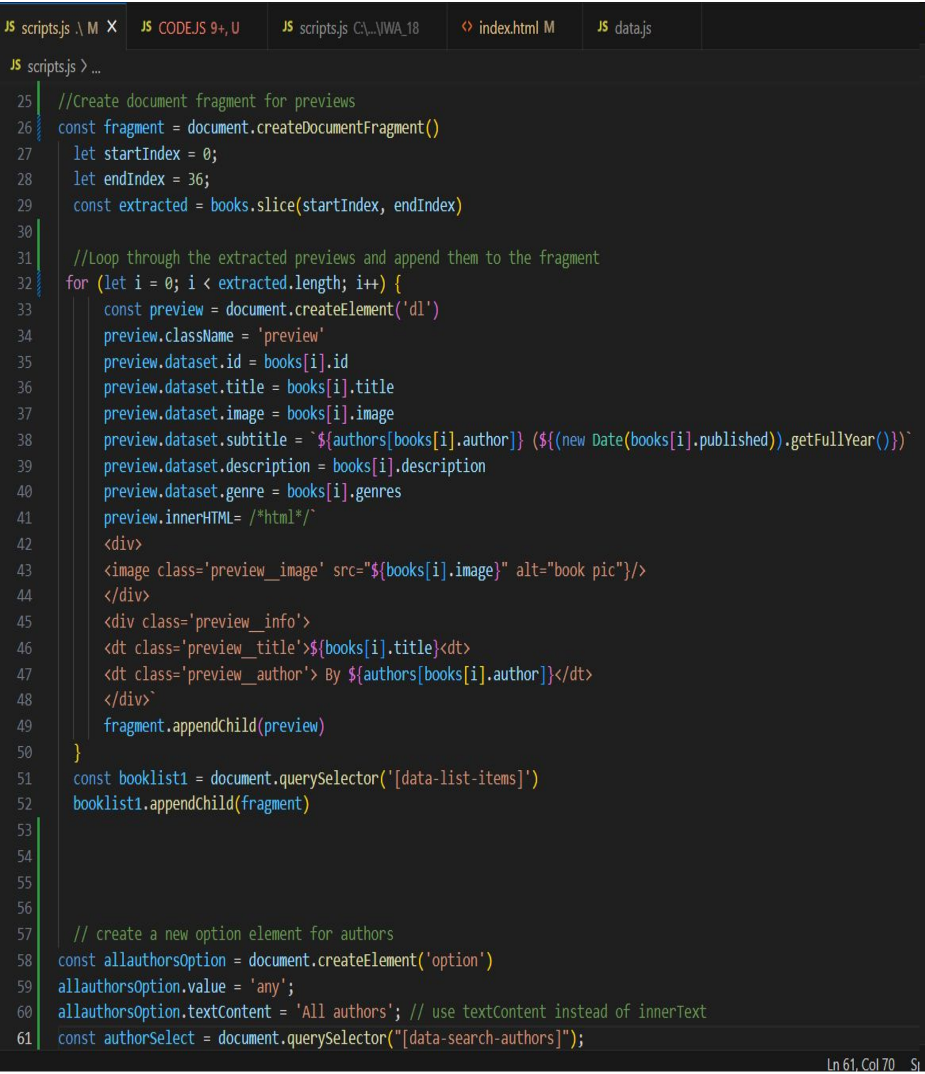
```
matches = books
page = 1;

if (!books && !Array.isArray(books)) throw new Error('Source required')
if (!range && range.length < 2) throw new Error('Range must be an array with two numbers')

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

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

fragment = document.createDocumentFragment()
const extracted = books.slice(0, 36)

for ({ author, image, title, id }; extracted; i++) {
    const preview = createPreview({
        author,
        id,
        image,
        title
    })

    fragment.appendChild(preview)
}

data-list-items.appendChild(fragment)

genres = document.createDocumentFragment()
element = document.createElement('option')
element.value = 'any'
element = 'All Genres'
genres.appendChild(element)
```


scripts.js .\ M X
scripts.js C:\...\IWA_18 M
index.html M
data.js
scripts.js > settingOverlay.addEventListener('click') callback
Ln 140, Col 1   Spaces: 2   UTF-8   CRLF   {} JavaScript   Port

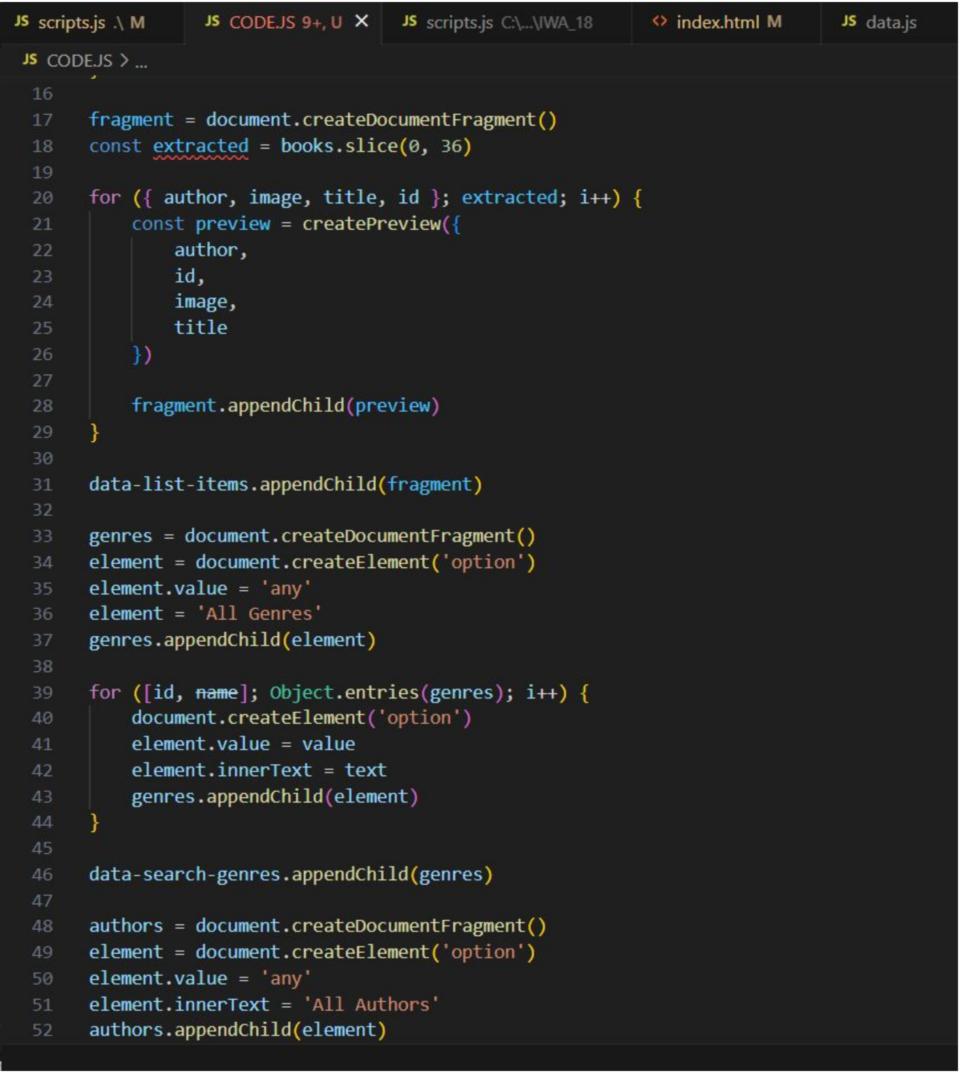
```
// create a new option element for authors
const allauthorsOption = document.createElement('option')
allauthorsOption.value = 'any';
allauthorsOption.textContent = 'All authors'; // use textContent instead of innerText
const authorSelect = document.querySelector("[data-search-authors]");
authorSelect.appendChild(allauthorsOption); // add the new option element to the select
for (const authorId in authors) {
  const optionElement = document.createElement('option');
  optionElement.value = authorId;
  optionElement.textContent = authors[authorId];
  authorSelect.appendChild(optionElement);
}

//// create a new option element genre
const genreSelect = document.querySelector("[data-search-genres]");
const allGenresOption = document.createElement('option');
allGenresOption.value = 'any';
allGenresOption.innerText = 'All Genres';
genreSelect.appendChild(allGenresOption);
for (const [genreId, genreName] of Object.entries(genres)) {
  const optionElement = document.createElement('option');
  optionElement.value = genreId;
  optionElement.textContent = genreName;
  genreSelect.appendChild(optionElement)
}




//change themes
const settingOverlay = document.querySelector('[data-header-settings]')
const themePopUp =document.querySelector('[data-settings-overlay]')
settingOverlay.addEventListener('click', ()=>{

})
const dataSettingsTheme = document.querySelector('[data-settings-theme]')
const saveButton = document.querySelector("body > dialog:nth-child(5) > div > div > button.overlay__button.overlay__button_primary")
```



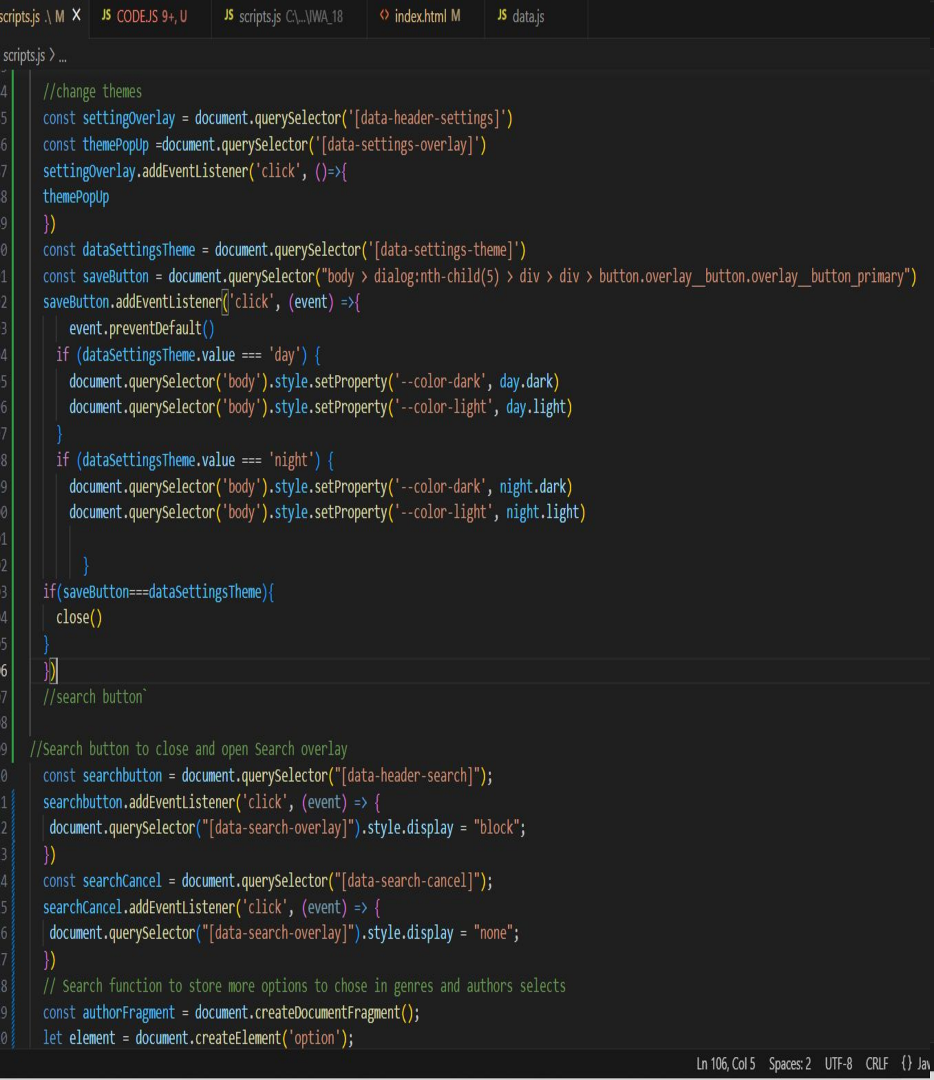
```
fragment = document.createDocumentFragment()
const extracted = books.slice(0, 36)

for ({ author, image, title, id }; extracted; i++) {
    const preview = createPreview({
        author,
        id,
        image,
        title
    })

    fragment.appendChild(preview)
}

data-list-items.appendChild(fragment)

genres = document.createDocumentFragment()
element = document.createElement('option')
element.value = 'any'
element = 'All Genres'
genres.appendChild(element)

for ([id, name]; Object.entries(genres); i++) {
    document.createElement('option')
    element.value = value
    element.innerText = text
    genres.appendChild(element)
}

data-search-genres.appendChild(genres)

authors = document.createDocumentFragment()
element = document.createElement('option')
element.value = 'any'
element.innerText = 'All Authors'
authors.appendChild(element)

```



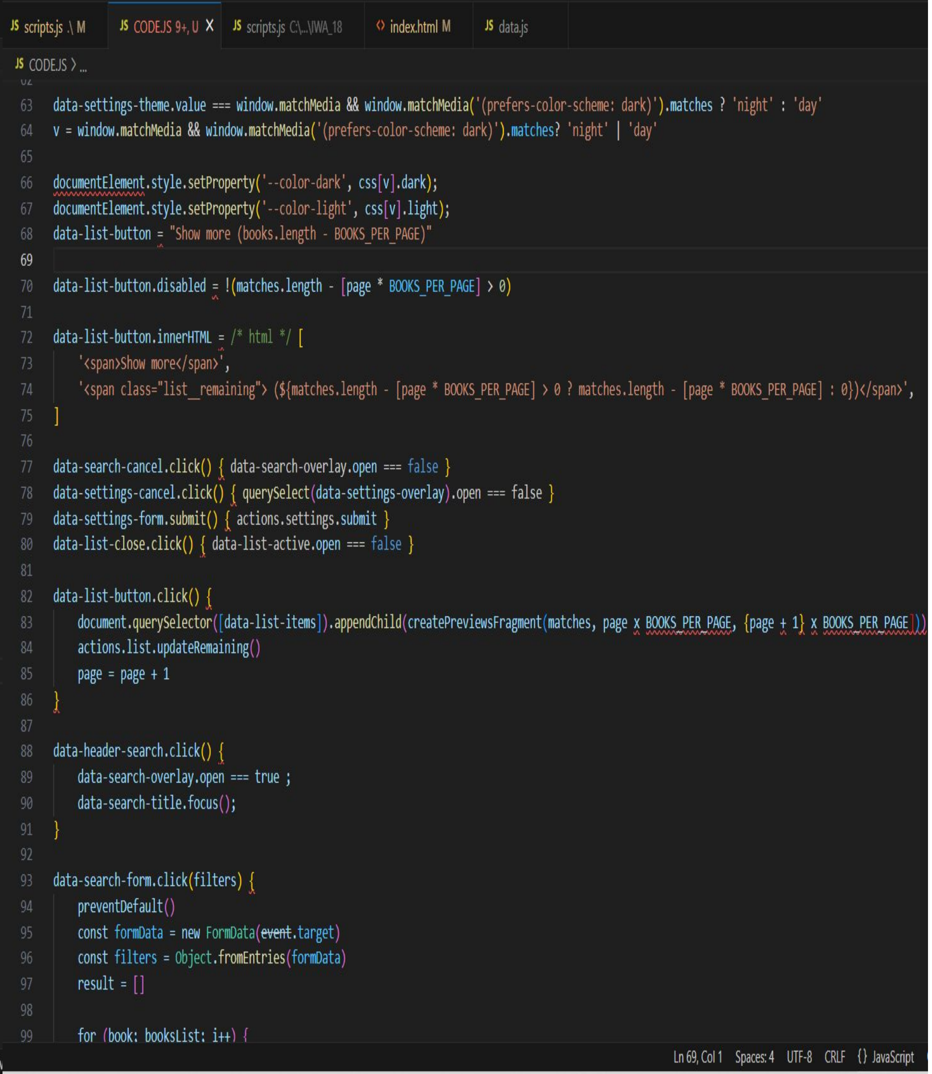
```
//Create document fragment for previews
const fragment = document.createDocumentFragment()
  let startIndex = 0;
  let endIndex = 36;
  const extracted = books.slice(startIndex, endIndex)

  //Loop through the extracted previews and append them to the fragment
 for (let i = 0; i < extracted.length; i++) {
      const preview = document.createElement('dl')
      preview.className = 'preview'
      preview.dataset.id = books[i].id
      preview.dataset.title = books[i].title
      preview.dataset.image = books[i].image
      preview.dataset.subtitle = `${authors[books[i].author]} (${(new Date(books[i].published)).getFullYear()})`
      preview.dataset.description = books[i].description
      preview.dataset.genre = books[i].genres
      preview.innerHTML= /*html*/`
      <div>
      <image class='preview__image' src="${books[i].image}" alt="book pic"}/>
      </div>
      <div class='preview__info'>
      <dt class='preview__title'>${books[i].title}<dt>
      <dt class='preview__author'> By ${authors[books[i].author]}</dt>
      </div>`
      fragment.appendChild(preview)
  }
  const booklist1 = document.querySelector('[data-list-items]')
  booklist1.appendChild(fragment)




  // create a new option element for authors
const allauthorsOption = document.createElement('option')
allauthorsOption.value = 'any';
allauthorsOption.textContent = 'All authors'; // use textContent instead of innerText
const authorSelect = document.querySelector("[data-search-authors]");
```



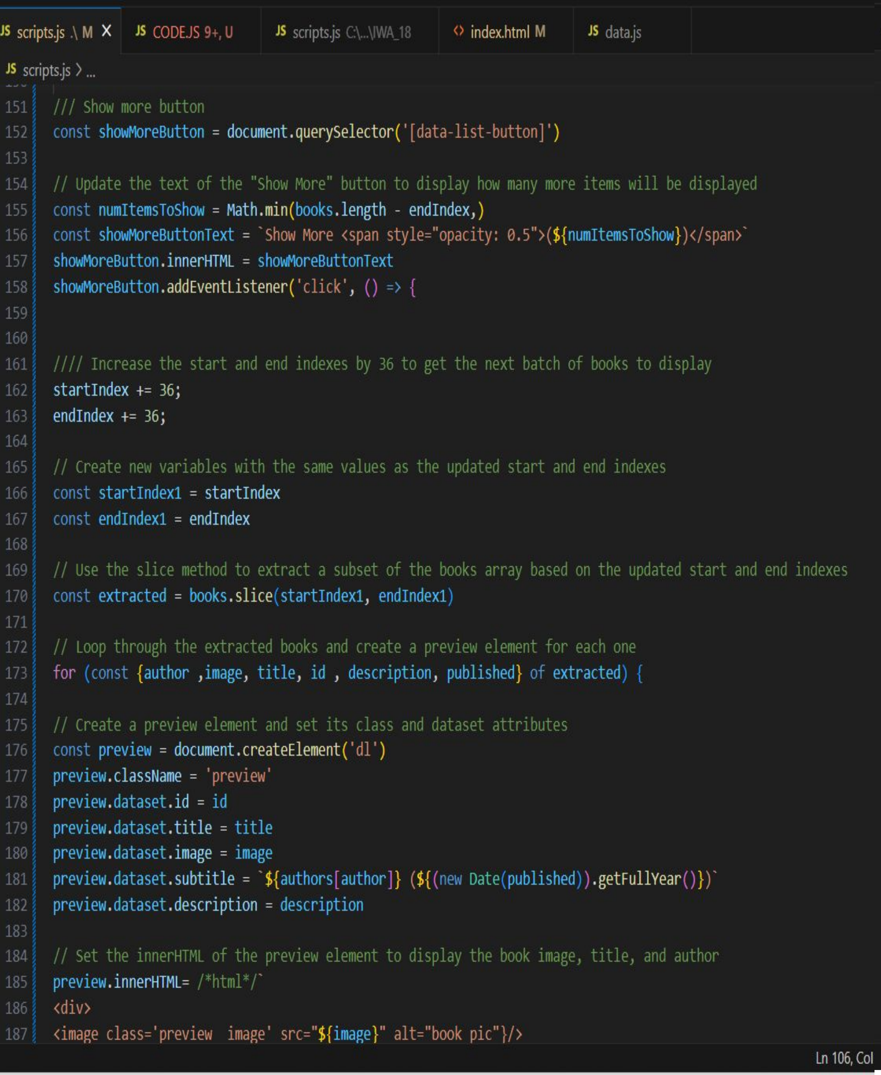
```

fragment = document.createDocumentFragment()
const extracted = books.slice(0, 36)

for ({ author, image, title, id }; extracted; i++) {
    const preview = createPreview({
        author,
        id,
        image,
        title
    })

    fragment.appendChild(preview)
}

data-list-items.appendChild(fragment)

genres = document.createDocumentFragment()
element = document.createElement('option')
element.value = 'any'
element = 'All Genres'
genres.appendChild(element)

for ([id, ~~name~~]; Object.entries(genres); i++) {
    document.createElement('option')
    element.value = value
    element.innerText = text
    genres.appendChild(element)
}

data-search-genres.appendChild(genres)

authors = document.createDocumentFragment()
element = document.createElement('option')
element.value = 'any'
element.innerText = 'All Authors'
authors.appendChild(element)
```



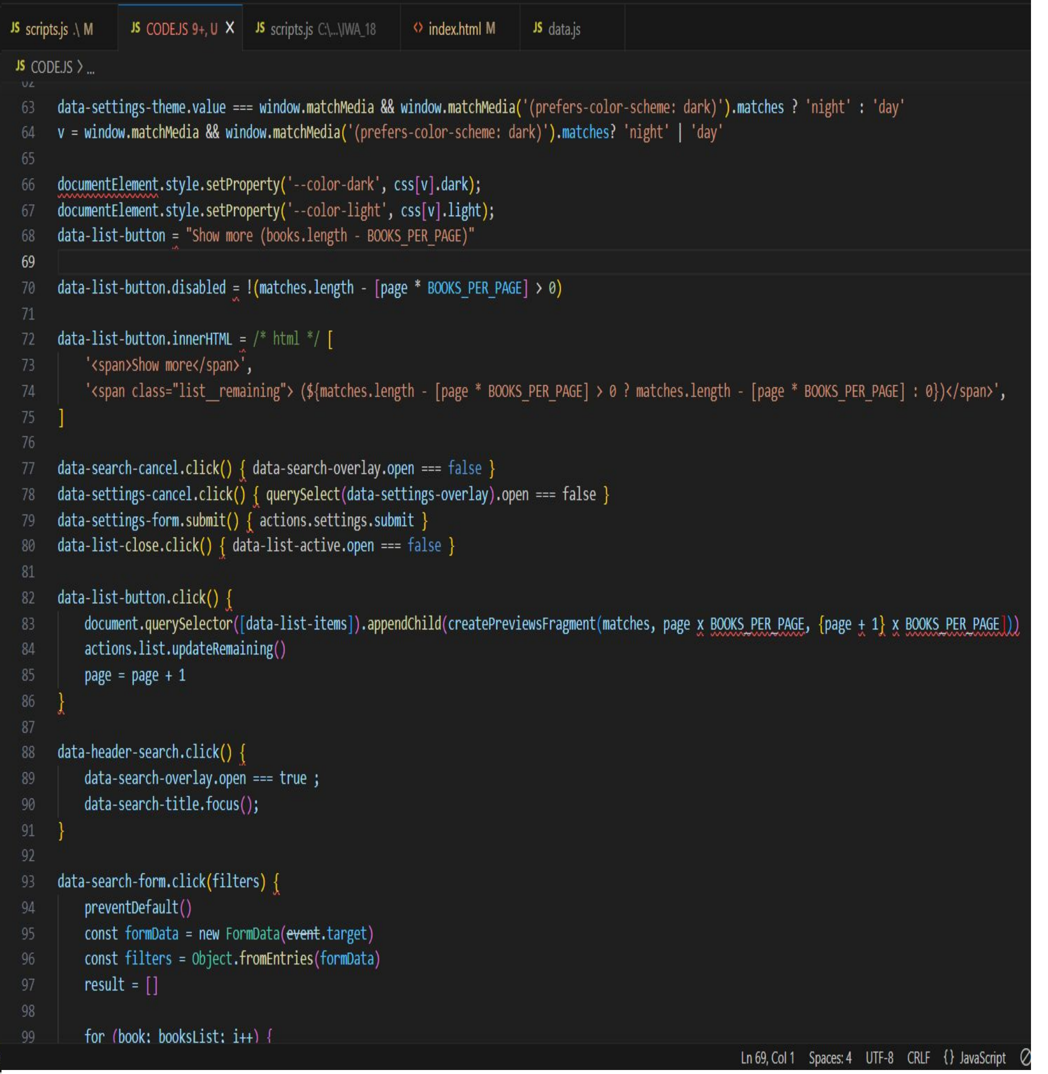
```
 //change themes
 const settingOverlay = document.querySelector('[data-header-settings]')
 const themePopUp =document.querySelector('[data-settings-overlay]')
 settingOverlay.addEventListener('click', ()=>{
 themePopUp
 })
 const dataSettingsTheme = document.querySelector('[data-settings-theme]')
 const saveButton = document.querySelector("body > dialog:nth-child(5) > div > div > button.overlay__button.overlay__button_primary")
 saveButton.addEventListener('click', (event) =>{
     event.preventDefault()
   if (dataSettingsTheme.value === 'day') {
     document.querySelector('body').style.setProperty('--color-dark', day.dark)
     document.querySelector('body').style.setProperty('--color-light', day.light)
   }
   if (dataSettingsTheme.value === 'night') {
     document.querySelector('body').style.setProperty('--color-dark', night.dark)
     document.querySelector('body').style.setProperty('--color-light', night.light)


       }
 if(saveButton===dataSettingsTheme){
   close()
 }
 })
 //search button`

//Search button to close and open Search overlay
 const searchbutton = document.querySelector("[data-header-search]");
 searchbutton.addEventListener('click', (event) => {
  document.querySelector("[data-search-overlay]").style.display = "block";
 })
 const searchCancel = document.querySelector("[data-search-cancel]");
 searchCancel.addEventListener('click', (event) => {
  document.querySelector("[data-search-overlay]").style.display = "none";
 })
 // Search function to store more options to chose in genres and authors selects
 const authorFragment = document.createDocumentFragment();
 let element = document.createElement('option');
```



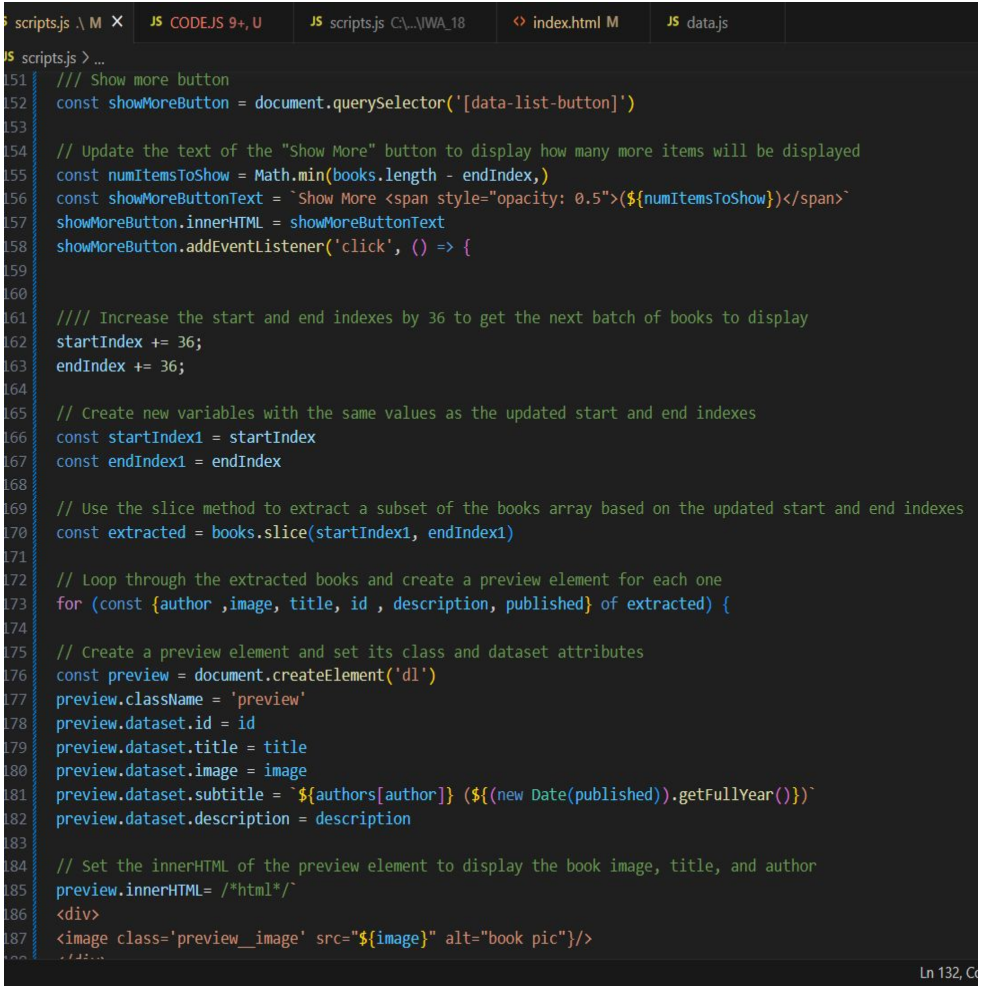
```
data-settings-theme.value === window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches ? 'night' : 'day'
v = window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches? 'night' | 'day'

documentElement.style.setProperty('--color-dark', css[v].dark);
documentElement.style.setProperty('--color-light', css[v].light);
data-list-button = "Show more (books.length - BOOKS_PER_PAGE)"

data-list-button.disabled = !(matches.length - [page * BOOKS_PER_PAGE] > 0)

data-list-button.innerHTML = /* html */ [
    '<span>Show more</span>',
    '<span class="list__remaining"> (${matches.length - [page * BOOKS_PER_PAGE] > 0 ? matches.length - [page * BOOKS_PER_PAGE] : 0})</span>',
]

data-search-cancel.click() { data-search-overlay.open === false }
data-settings-cancel.click() { querySelect(data-settings-overlay).open === false }
data-settings-form.submit() { actions.settings.submit }
data-list-close.click() { data-list-active.open === false }

data-list-button.click() {
    document.querySelector([data-list-items]).appendChild(createPreviewsFragment(matches, page x BOOKS_PER_PAGE, {page + 1} x BOOKS_PER_PAGE]))
    actions.list.updateRemaining()
    page = page + 1
}

data-header-search.click() {
    data-search-overlay.open === true ;
    data-search-title.focus();
}

data-search-form.click(filters) {
    preventDefault()
    const formData = new FormData(event.target)
    const filters = Object.fromEntries(formData)
    result = []

    for (book; booksList; i++) {
```



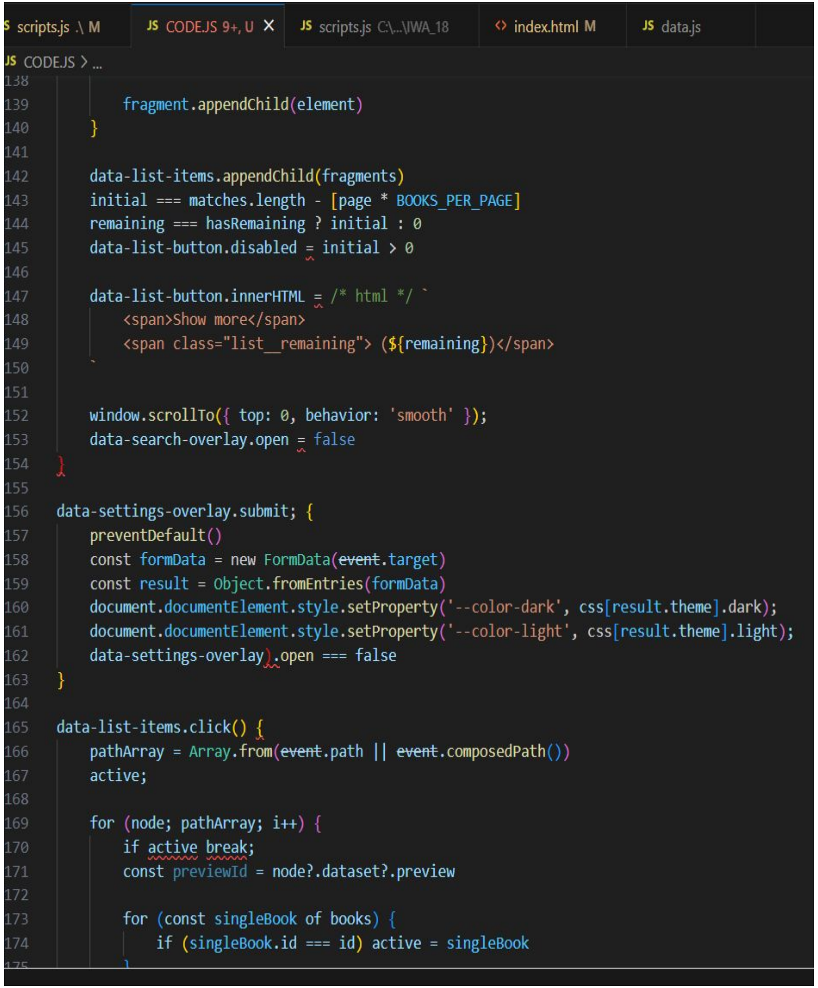
```
/// Show more button
const showMoreButton = document.querySelector('[data-list-button]')

// Update the text of the "Show More" button to display how many more items will be displayed
const numItemsToShow = Math.min(books.length - endIndex,)
const showMoreButtonText = `Show More <span style="opacity: 0.5">(${numItemsToShow})</span>`
showMoreButton.innerHTML = showMoreButtonText
showMoreButton.addEventListener('click', () => {


//// Increase the start and end indexes by 36 to get the next batch of books to display
startIndex += 36;
endIndex += 36;

// Create new variables with the same values as the updated start and end indexes
const startIndex1 = startIndex
const endIndex1 = endIndex

// Use the slice method to extract a subset of the books array based on the updated start and end indexes
const extracted = books.slice(startIndex1, endIndex1)

// Loop through the extracted books and create a preview element for each one
for (const {author ,image, title, id , description, published} of extracted) {

// Create a preview element and set its class and dataset attributes
const preview = document.createElement('dl')
preview.className = 'preview'
preview.dataset.id = id
preview.dataset.title = title
preview.dataset.image = image
preview.dataset.subtitle = `${authors[author]} (${(new Date(published)).getFullYear()})`
preview.dataset.description = description

// Set the innerHTML of the preview element to display the book image, title, and author
preview.innerHTML= /*html*/`
<div>
<image class='preview__image' src="${image}" alt="book pic"}/>
```



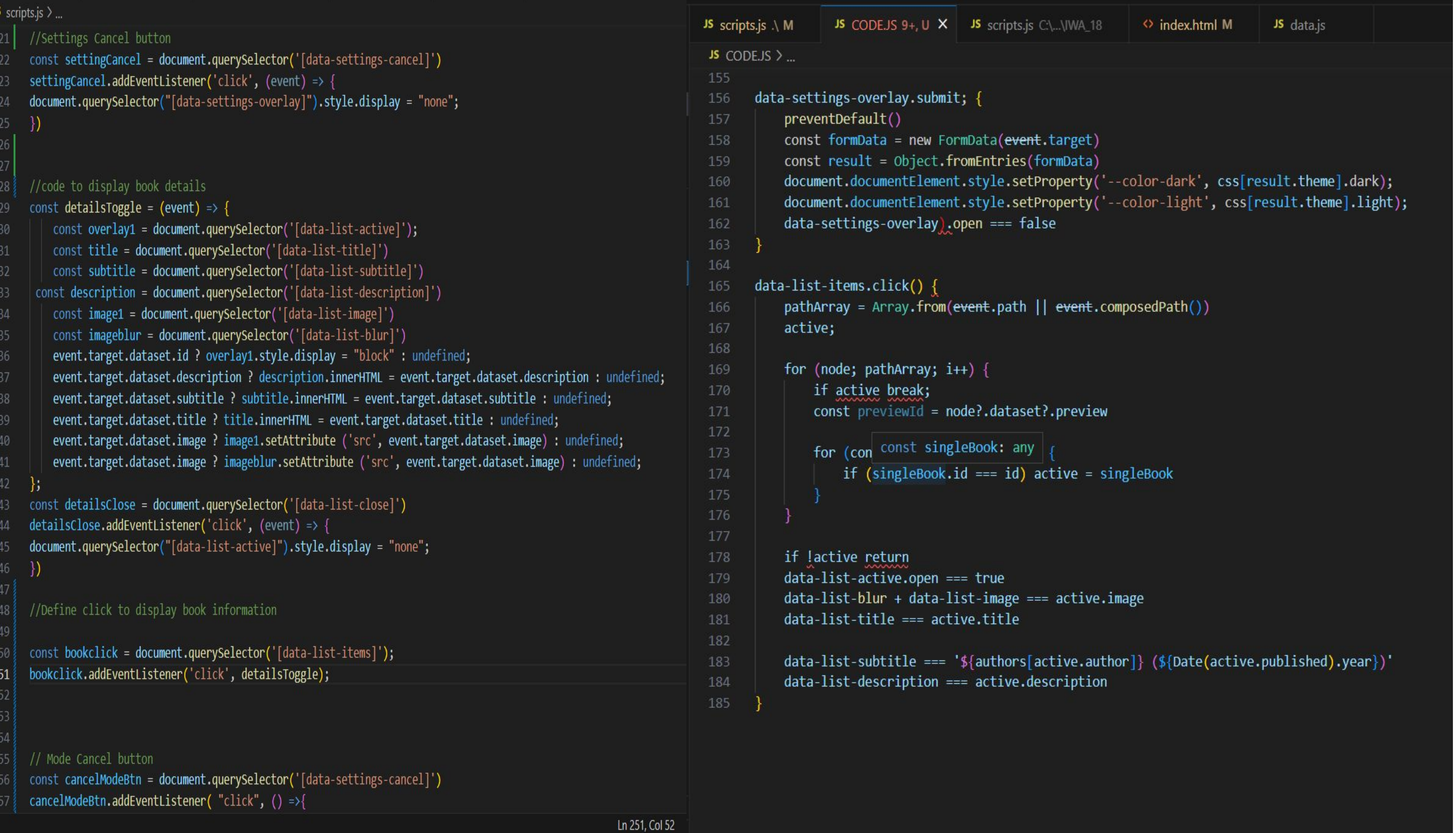
```
data-settings-theme.value === window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches ? 'night' : 'day'
v = window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches? 'night' | 'day'

documentElement.style.setProperty('--color-dark', css[v].dark);
documentElement.style.setProperty('--color-light', css[v].light);
data-list-button = "Show more (books.length - BOOKS_PER_PAGE)"

data-list-button.disabled = !(matches.length - [page * BOOKS_PER_PAGE] > 0)

data-list-button.innerHTML = /* html */ [
    '<span>Show more</span>',
    '<span class="list__remaining"> (${matches.length - [page * BOOKS_PER_PAGE] > 0 ? matches.length - [page * BOOKS_PER_PAGE] : 0})</span>',
]

data-search-cancel.click() { data-search-overlay.open === false }
data-settings-cancel.click() { querySelect(data-settings-overlay).open === false }
data-settings-form.submit() { actions.settings.submit }
data-list-close.click() { data-list-active.open === false }

data-list-button.click() {
    document.querySelector([data-list-items]).appendChild(createPreviewsFragment(matches, page x BOOKS_PER_PAGE, {page + 1} x BOOKS_PER_PAGE]))
    actions.list.updateRemaining()
    page = page + 1
}

data-header-search.click() {
    data-search-overlay.open === true ;
    data-search-title.focus();
}

data-search-form.click(filters) {
    preventDefault()
    const formData = new FormData(~~event~~.target)
    const filters = Object.fromEntries(formData)
    result = []

    for (book: booksList: i++) {
```



```
/// Show more button
const showMoreButton = document.querySelector('[data-list-button]')

// Update the text of the "Show More" button to display how many more items will be displayed
const numItemsToShow = Math.min(books.length - endIndex,)
const showMoreButtonText = `Show More <span style="opacity: 0.5">(${numItemsToShow})</span>`
showMoreButton.innerHTML = showMoreButtonText
showMoreButton.addEventListener('click', () => {


//// Increase the start and end indexes by 36 to get the next batch of books to display
startIndex += 36;
endIndex += 36;

// Create new variables with the same values as the updated start and end indexes
const startIndex1 = startIndex
const endIndex1 = endIndex

// Use the slice method to extract a subset of the books array based on the updated start and end indexes
const extracted = books.slice(startIndex1, endIndex1)

// Loop through the extracted books and create a preview element for each one
for (const {author ,image, title, id , description, published} of extracted) {

// Create a preview element and set its class and dataset attributes
const preview = document.createElement('dl')
preview.className = 'preview'
preview.dataset.id = id
preview.dataset.title = title
preview.dataset.image = image
preview.dataset.subtitle = `${authors[author]} (${(new Date(published)).getFullYear()})`
preview.dataset.description = description

// Set the innerHTML of the preview element to display the book image, title, and author
preview.innerHTML= /*html*/`
<div>
<image class='preview__image' src="${image}" alt="book pic"}/>
```



```
        fragment.appendChild(element)
    }

    data-list-items.appendChild(fragments)
    initial === matches.length - [page * BOOKS_PER_PAGE]
    remaining === hasRemaining ? initial : 0
    data-list-button.disabled = initial > 0

    data-list-button.innerHTML = /* html */ `
        <span>Show more</span>
        <span class="list__remaining"> (${remaining})</span>
    `

    window.scrollTo({ top: 0, behavior: 'smooth' });
    data-search-overlay.open = false
}

data-settings-overlay.submit; {
    preventDefault()
    const formData = new FormData(event.target)
    const result = Object.fromEntries(formData)
    document.documentElement.style.setProperty('--color-dark', css[result.theme].dark);
    document.documentElement.style.setProperty('--color-light', css[result.theme].light);
    data-settings-overlay).open === false
}

data-list-items.click() {
    pathArray = Array.from(event.path || event.composedPath())
    active;

    for (node; pathArray; i++) {
        if active break;
        const previewId = node?.dataset?.preview

        for (const singleBook of books) {
            if (singleBook.id === id) active = singleBook
```



```
//Settings Cancel button
const settingCancel = document.querySelector('[data-settings-cancel]')
settingCancel.addEventListener('click', (event) => {
document.querySelector("[data-settings-overlay]").style.display = "none";
})


//code to display book details
const detailsToggle = (event) => {
    const overlay1 = document.querySelector('[data-list-active]');
    const title = document.querySelector('[data-list-title]')
    const subtitle = document.querySelector('[data-list-subtitle]')
 const description = document.querySelector('[data-list-description]')
    const image1 = document.querySelector('[data-list-image]')
    const imageblur = document.querySelector('[data-list-blur]')
    event.target.dataset.id ? overlay1.style.display = "block" : undefined;
    event.target.dataset.description ? description.innerHTML = event.target.dataset.description : undefined;
    event.target.dataset.subtitle ? subtitle.innerHTML = event.target.dataset.subtitle : undefined;
    event.target.dataset.title ? title.innerHTML = event.target.dataset.title : undefined;
    event.target.dataset.image ? image1.setAttribute ('src', event.target.dataset.image) : undefined;
    event.target.dataset.image ? imageblur.setAttribute ('src', event.target.dataset.image) : undefined;
};
const detailsClose = document.querySelector('[data-list-close]')
detailsClose.addEventListener('click', (event) => {
document.querySelector("[data-list-active]").style.display = "none";
})

//Define click to display book information

const bookclick = document.querySelector('[data-list-items]');
bookclick.addEventListener('click', detailsToggle);



// Mode Cancel button
const cancelModeBtn = document.querySelector('[data-settings-cancel]')
cancelModeBtn.addEventListener( "click", () =>{
```

```

data-settings-overlay.submit; {
    preventDefault()
    const formData = new FormData(event.target)
    const result = Object.fromEntries(formData)
    document.documentElement.style.setProperty('--color-dark', css[result.theme].dark);
    document.documentElement.style.setProperty('--color-light', css[result.theme].light);
    data-settings-overlay).open === false
}

data-list-items.click() {
    pathArray = Array.from(event.path || event.composedPath())
    active;

    for (node; pathArray; i++) {
        if active break;
        const previewId = node?.dataset?.preview

        for (con const singleBook: any {
            if (singleBook.id === id) active = singleBook
        }
    }

    if !active return
    data-list-active.open === true
    data-list-blur + data-list-image === active.image
    data-list-title === active.title

    data-list-subtitle === '${authors[active.author]} (${Date(active.published).year})'
    data-list-description === active.description
}
```