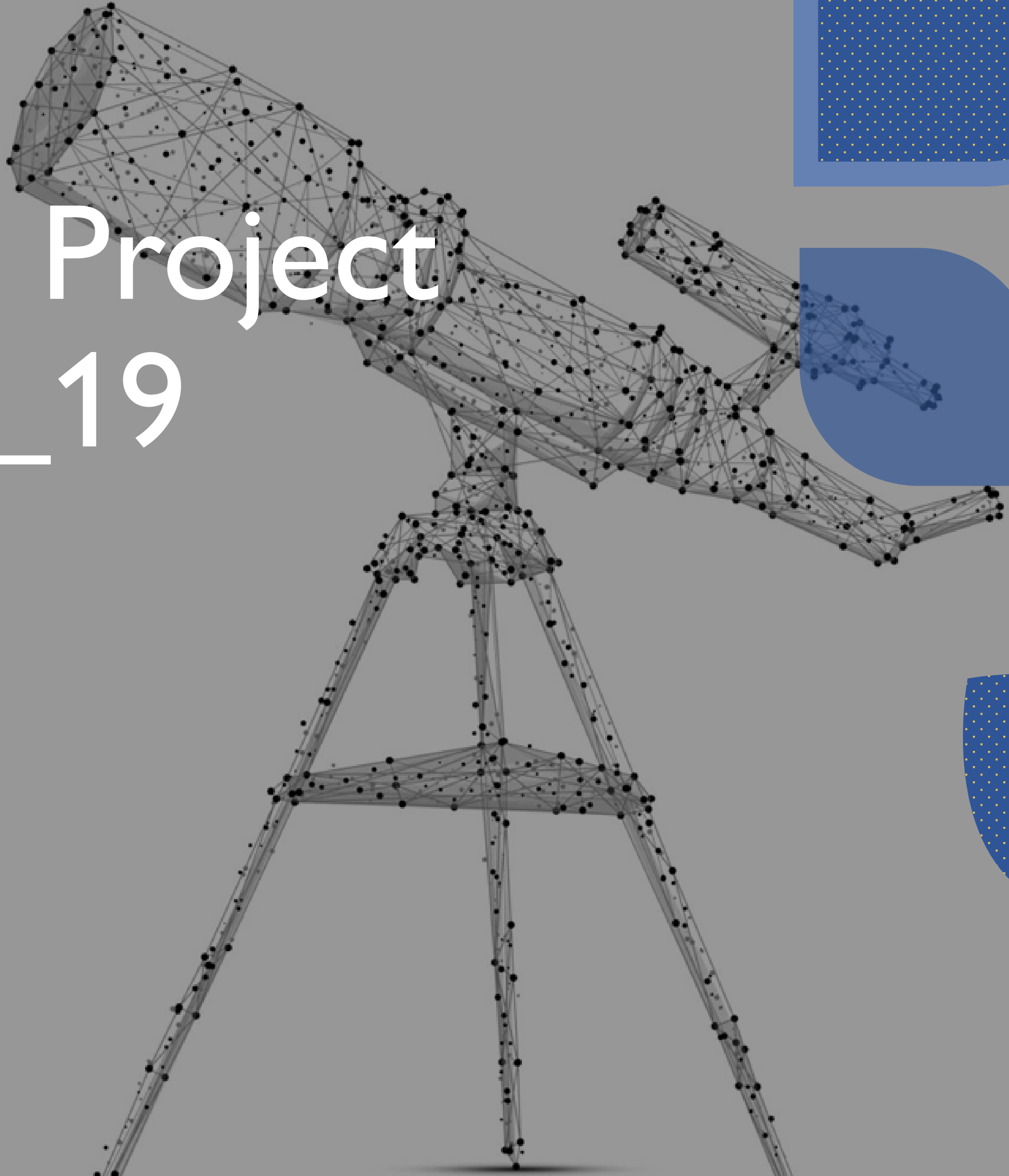
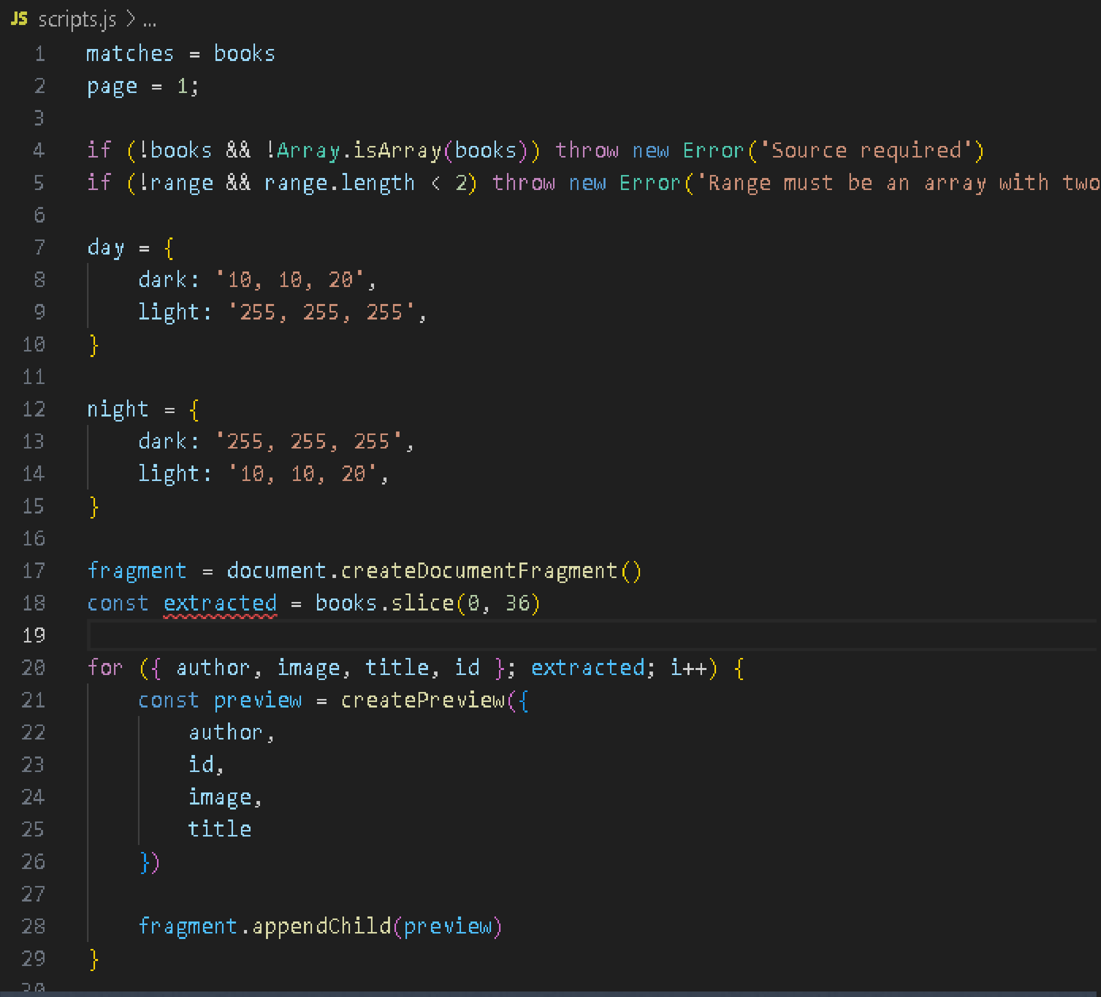
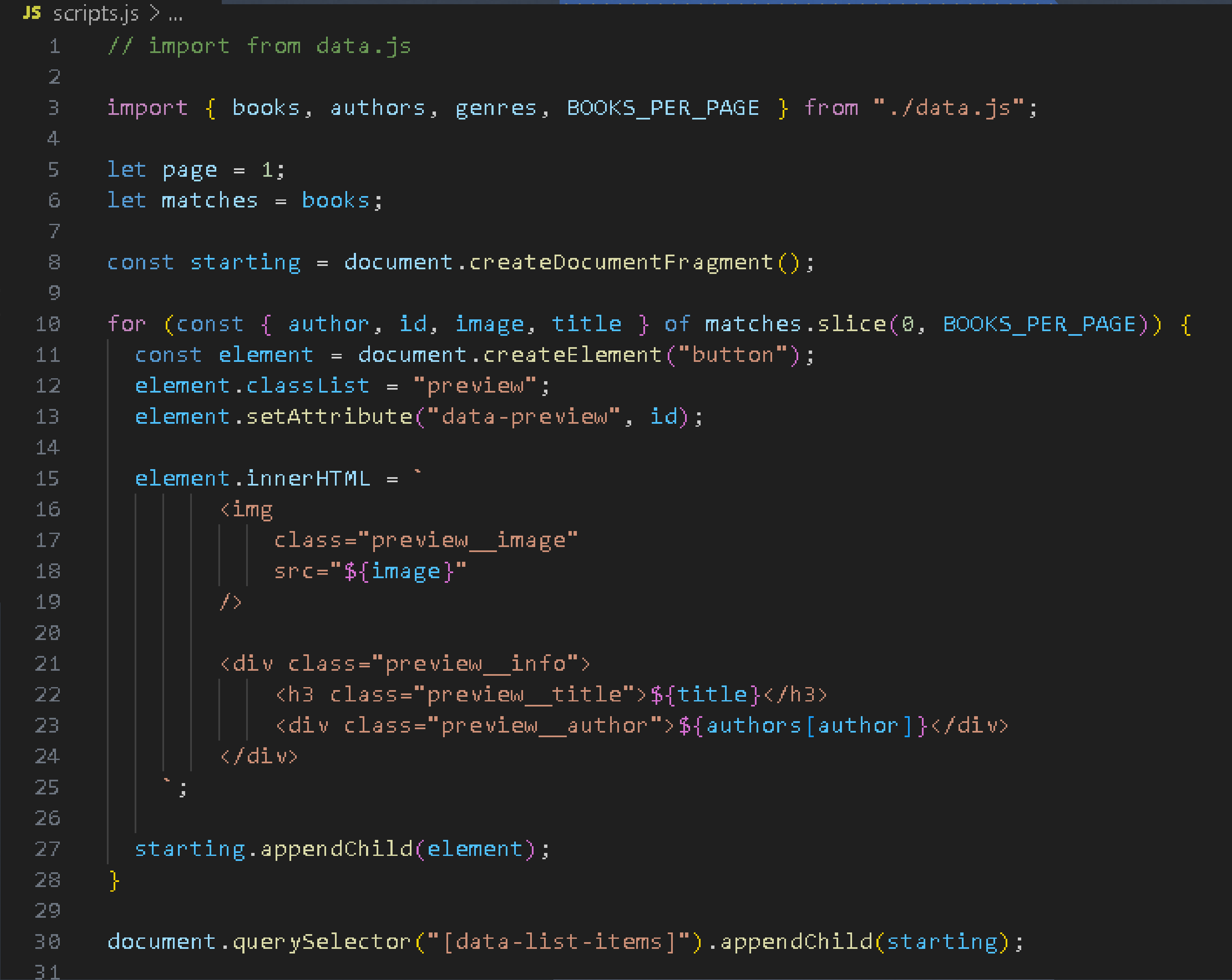
Capstone Project
IWA_19





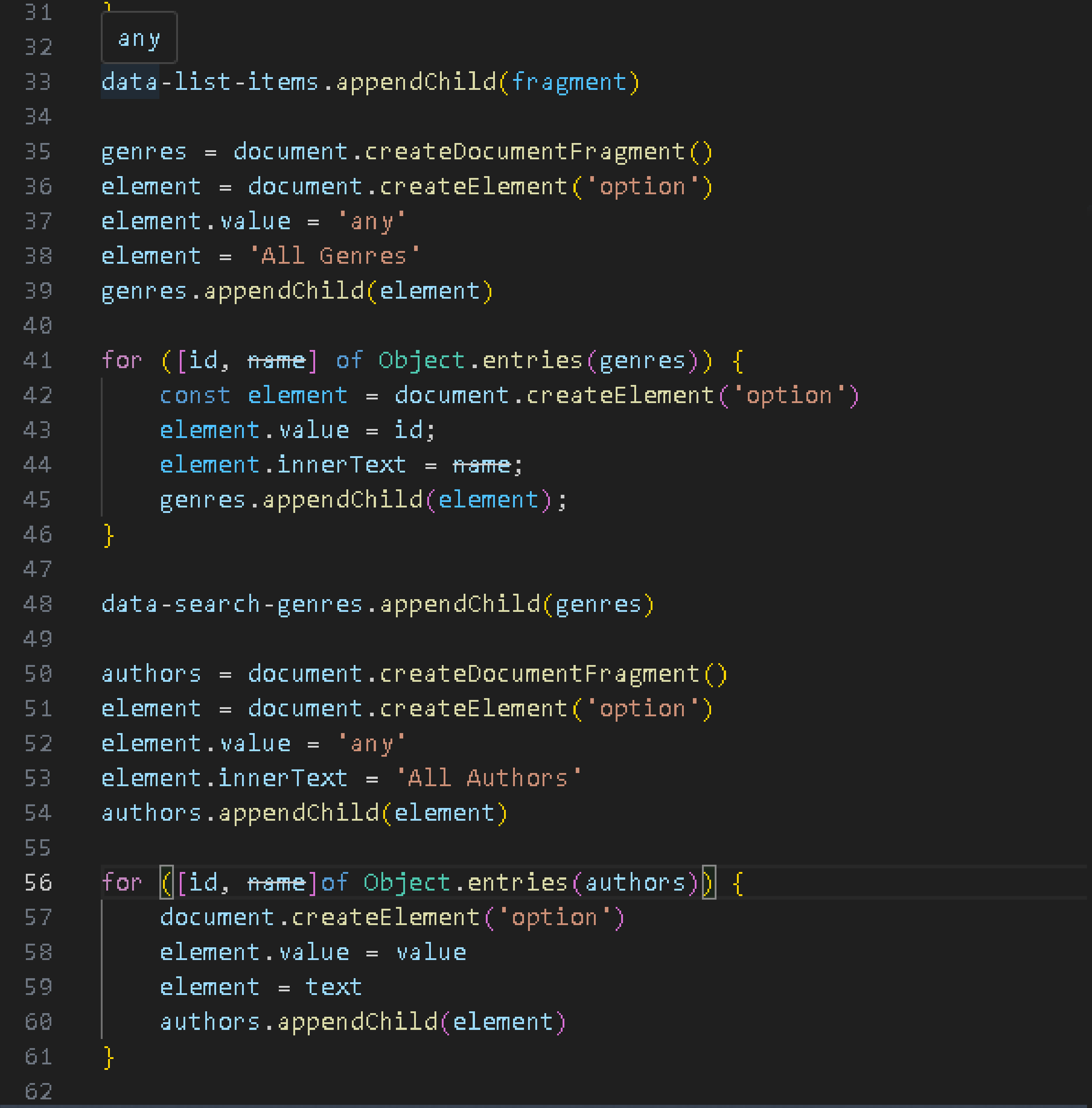
JS scripts.js > ...

```
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)
}
```

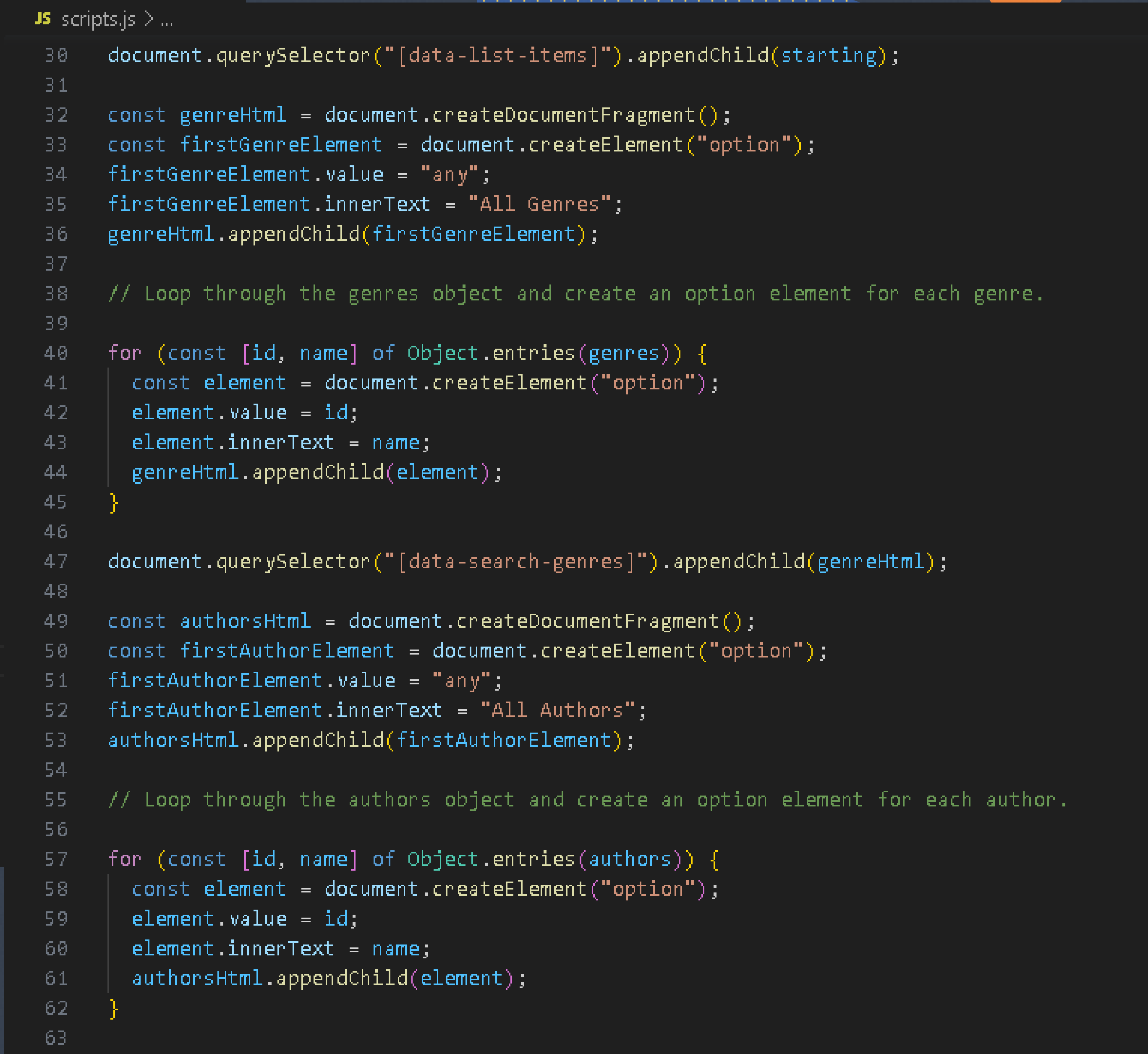


JS scripts.js > ...

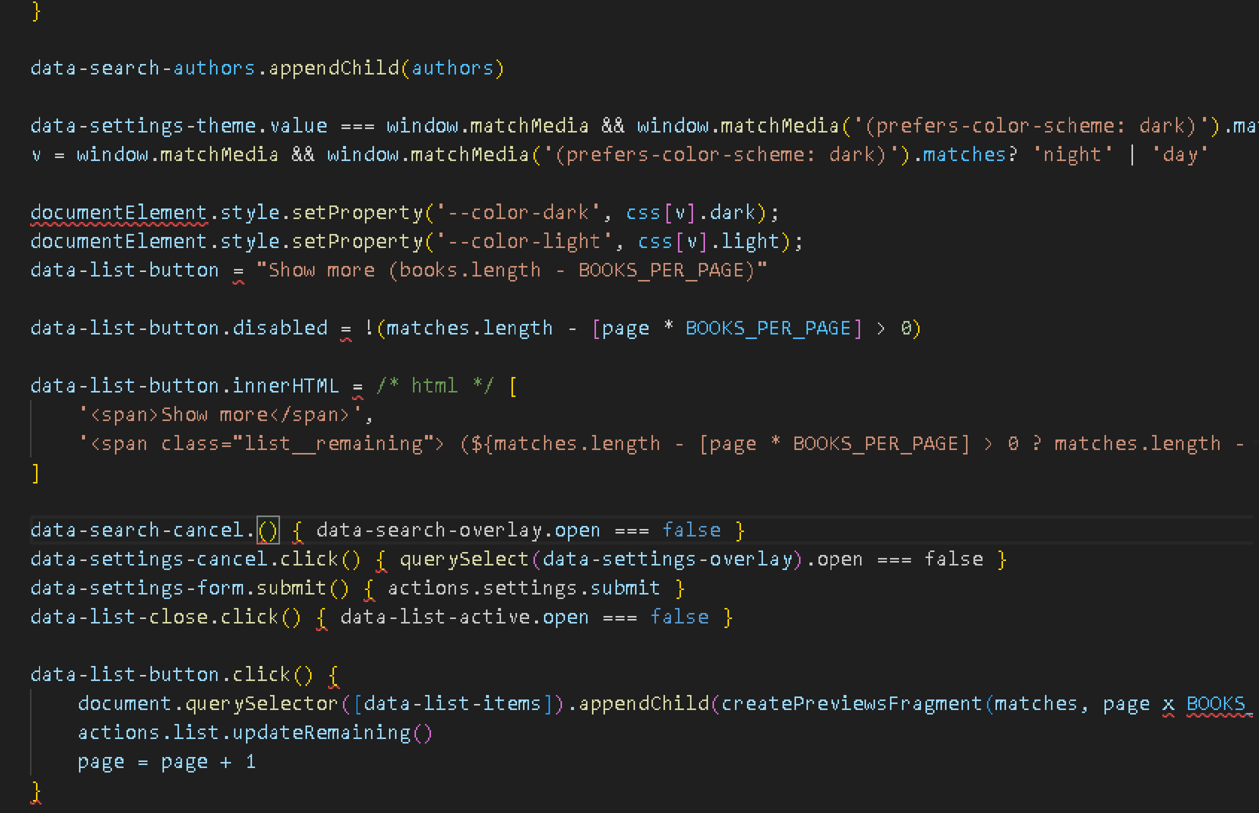
```
// import from data.js

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

let page = 1;
let matches = books;

const starting = document.createDocumentFragment();

for (const { author, id, image, title } of matches.slice(0, BOOKS_PER_PAGE)) {
  const element = document.createElement("button");
  element.classList = "preview";
  element.setAttribute("data-preview", id);

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

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

  starting.appendChild(element);
}

document.querySelector("[data-list-items]").appendChild(starting);
```



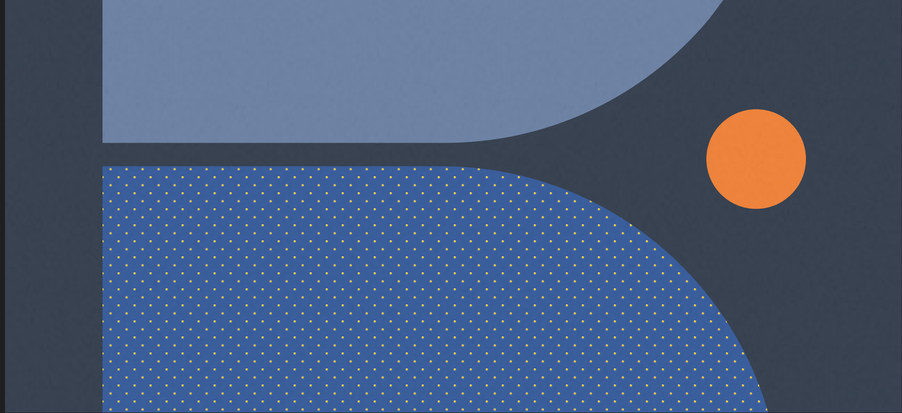
```
}

data-list-items.appendChild(fragment)

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

for ([id, ~~name~~] of Object.entries(genres)) {
    const element = document.createElement('option')
    element.value = id;
    element.innerText = ~~name~~;
    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)

for ([id, ~~name~~]of Object.entries(authors)) {
    document.createElement('option')
    element.value = value
    element = text
    authors.appendChild(element)
}
```

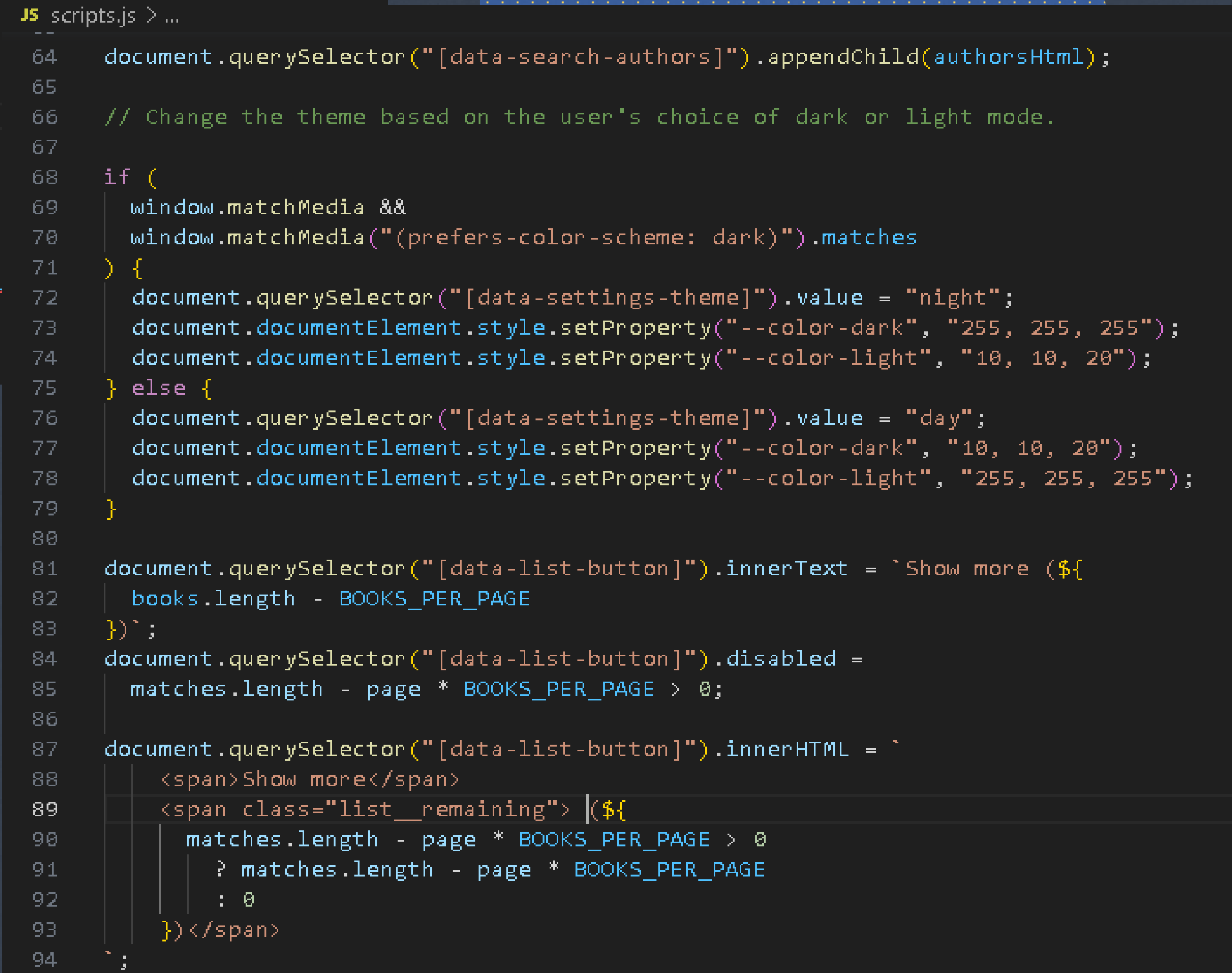


```
document.querySelector("[data-list-items]").appendChild(starting);

const genreHtml = document.createDocumentFragment();
const firstGenreElement = document.createElement("option");
firstGenreElement.value = "any";
firstGenreElement.innerText = "All Genres";
genreHtml.appendChild(firstGenreElement);

// Loop through the genres object and create an option element for each genre.

for (const [id, name] of Object.entries(genres)) {
  const element = document.createElement("option");
  element.value = id;
  element.innerText = name;
  genreHtml.appendChild(element);
}

document.querySelector("[data-search-genres]").appendChild(genreHtml);

const authorsHtml = document.createDocumentFragment();
const firstAuthorElement = document.createElement("option");
firstAuthorElement.value = "any";
firstAuthorElement.innerText = "All Authors";
authorsHtml.appendChild(firstAuthorElement);

// Loop through the authors object and create an option element for each author.

for (const [id, name] of Object.entries(authors)) {
  const element = document.createElement("option");
  element.value = id;
  element.innerText = name;
  authorsHtml.appendChild(element);
}
```

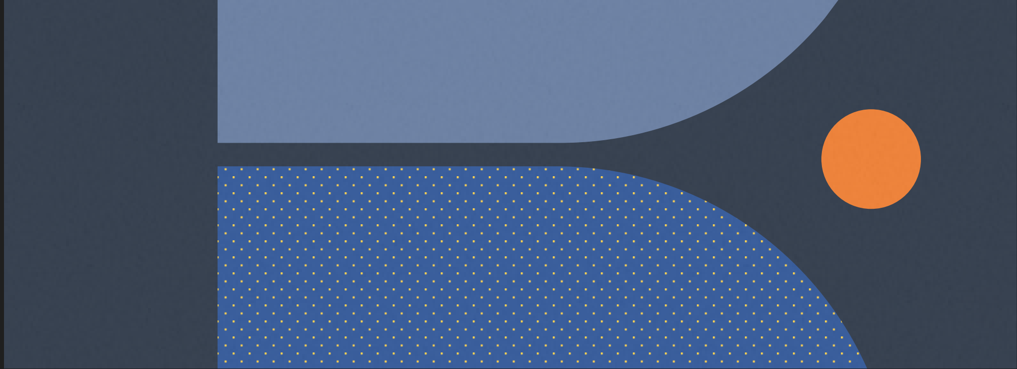


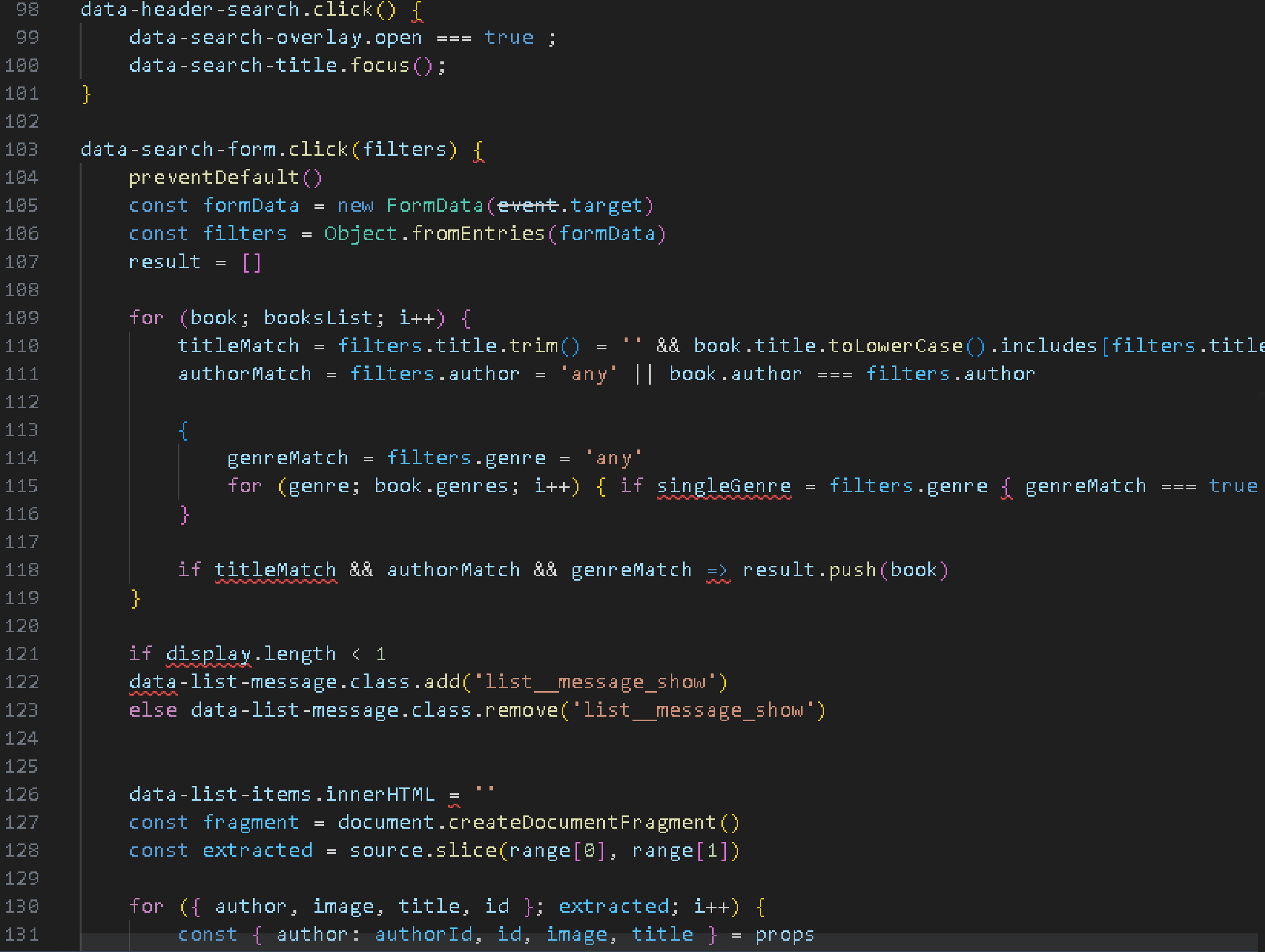
```
}

data-search-authors.appendChild(authors)

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 -
]

data-search-cancel.() { 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_
    actions.list.updateRemaining()
    page = page + 1
}
```



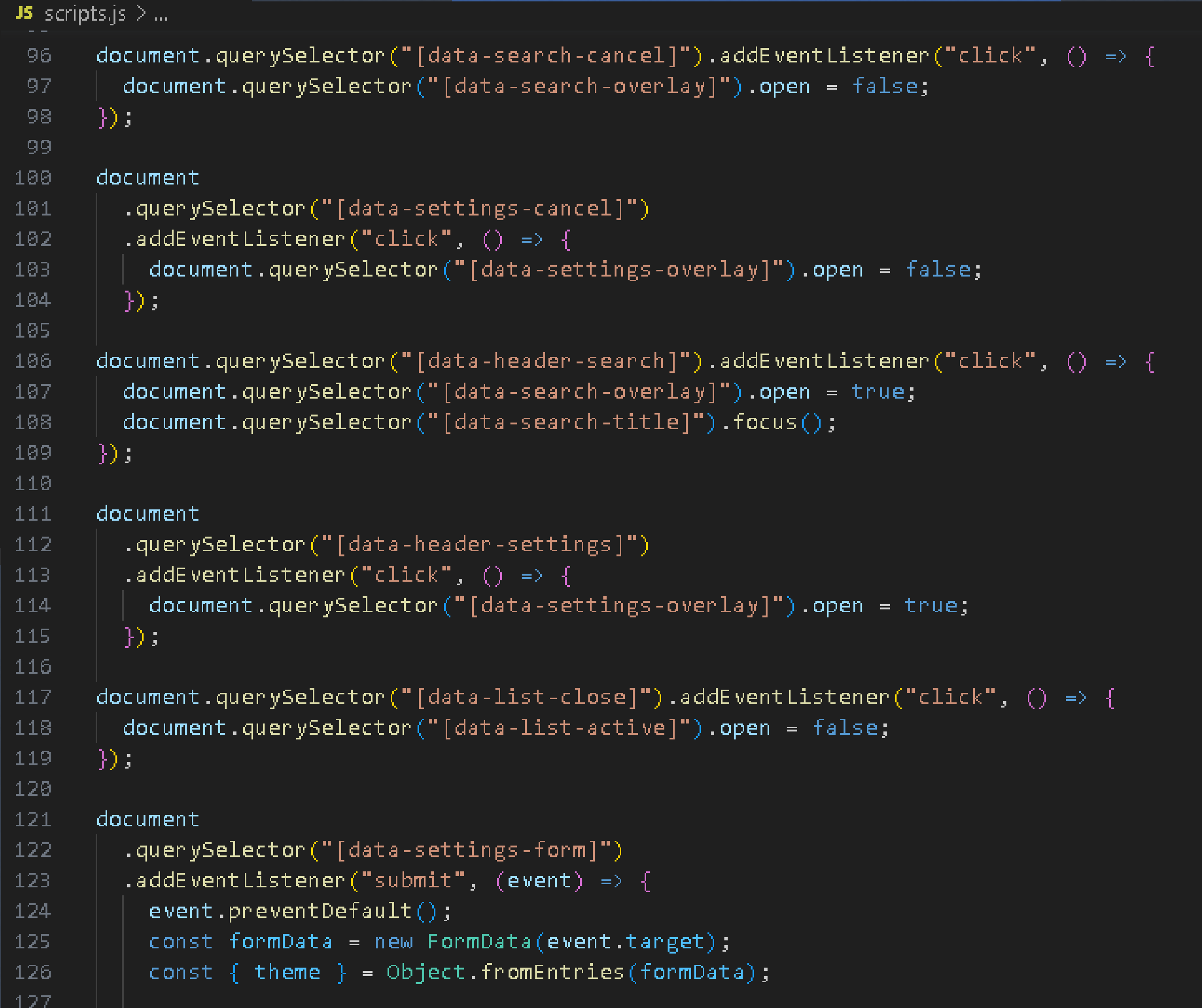


```
document.querySelector("[data-search-authors]").appendChild(authorsHtml);

// Change the theme based on the user's choice of dark or light mode.

if (
  window.matchMedia &&
  window.matchMedia("(prefers-color-scheme: dark)").matches
) {
  document.querySelector("[data-settings-theme]").value = "night";
  document.documentElement.style.setProperty("--color-dark", "255, 255, 255");
  document.documentElement.style.setProperty("--color-light", "10, 10, 20");
} else {
  document.querySelector("[data-settings-theme]").value = "day";
  document.documentElement.style.setProperty("--color-dark", "10, 10, 20");
  document.documentElement.style.setProperty("--color-light", "255, 255, 255");
}

document.querySelector("[data-list-button]").innerText = `Show more (${
  books.length - BOOKS_PER_PAGE
})`;
document.querySelector("[data-list-button]").disabled =
  matches.length - page * BOOKS_PER_PAGE > 0;

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

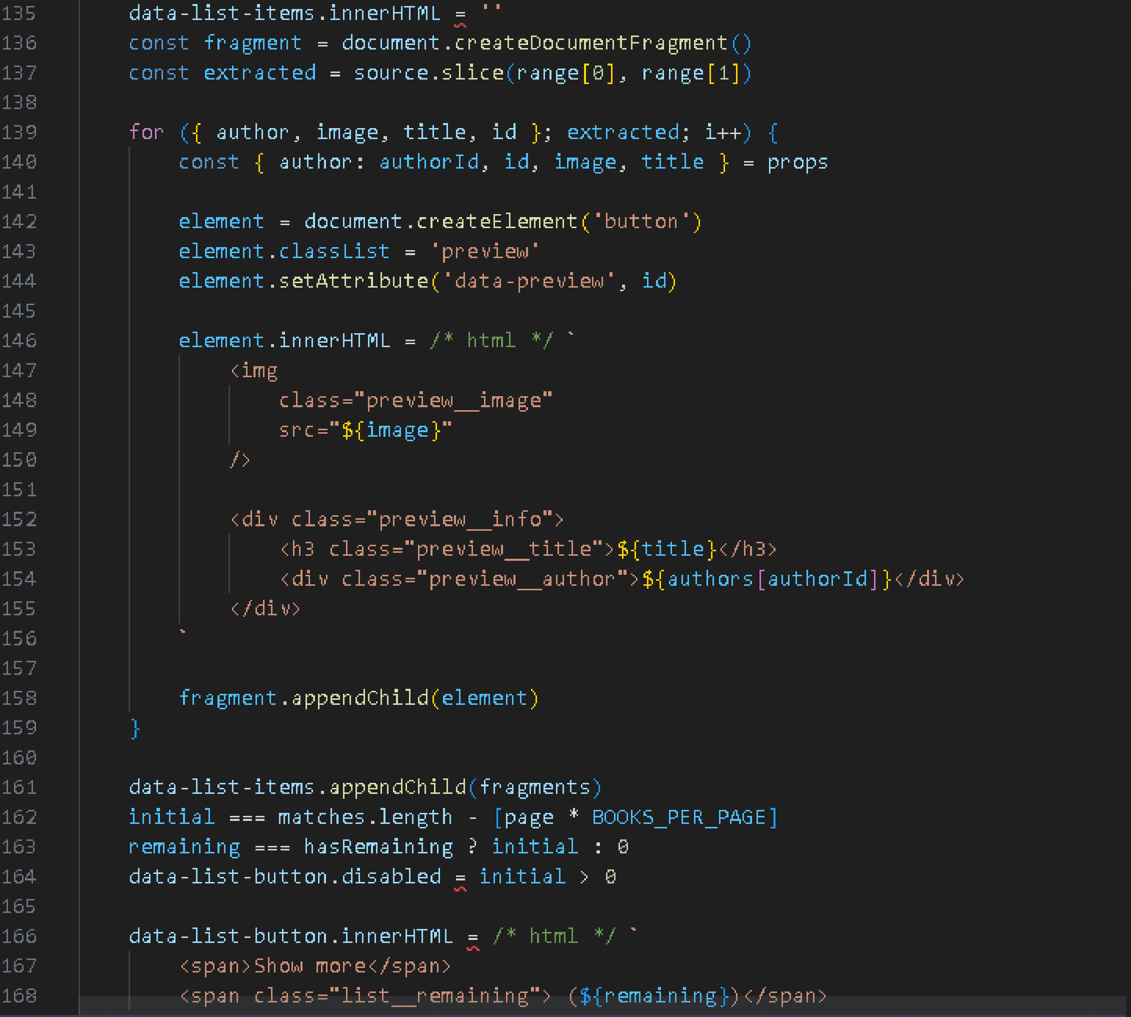




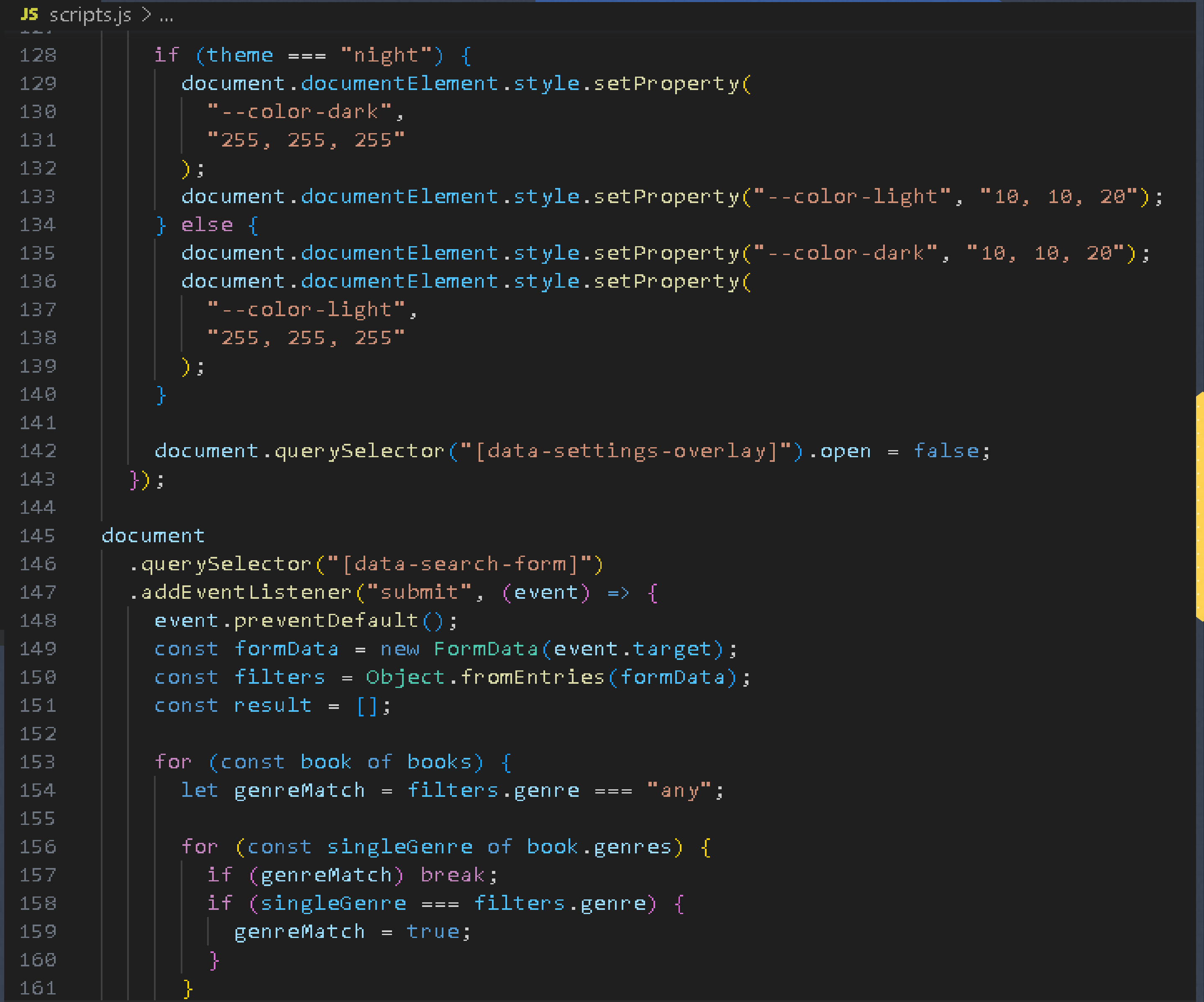
```
data-header-search.click() {
    data-search-overlay.open === true ;
    data-search-title.focus();
}

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

    for (book; booksList; i++) {
        titleMatch = filters.title.trim() = '' && book.title.toLowerCase().includes[filters.title.toLowerCase()]
        authorMatch = filters.author = 'any' || book.author === filters.author

        {
            genreMatch = filters.genre = 'any'
            for (genre; book.genres; i++) { if singleGenre = filters.genre { genreMatch === true
        }

        if titleMatch && authorMatch && genreMatch => result.push(book)
    }

    if display.length < 1
    data-list-message.class.add('list__message_show')
    else data-list-message.class.remove('list__message_show')


    data-list-items.innerHTML = ''
    const fragment = document.createDocumentFragment()
    const extracted = source.slice(range[0], range[1])

    for ({ author, image, title, id }; extracted; i++) {
        const { author: authorId, id, image, title } = props
```



```
document.querySelector("[data-search-cancel]").addEventListener("click", () => {
  document.querySelector("[data-search-overlay]").open = false;
});

document
  .querySelector("[data-settings-cancel]")
  .addEventListener("click", () => {
    document.querySelector("[data-settings-overlay]").open = false;
  });

document.querySelector("[data-header-search]").addEventListener("click", () => {
  document.querySelector("[data-search-overlay]").open = true;
  document.querySelector("[data-search-title]").focus();
});

document
  .querySelector("[data-header-settings]")
  .addEventListener("click", () => {
    document.querySelector("[data-settings-overlay]").open = true;
  });

document.querySelector("[data-list-close]").addEventListener("click", () => {
  document.querySelector("[data-list-active]").open = false;
});

document
  .querySelector("[data-settings-form]")
  .addEventListener("submit", (event) => {
    event.preventDefault();
    const formData = new FormData(event.target);
    const { theme } = Object.fromEntries(formData);
```

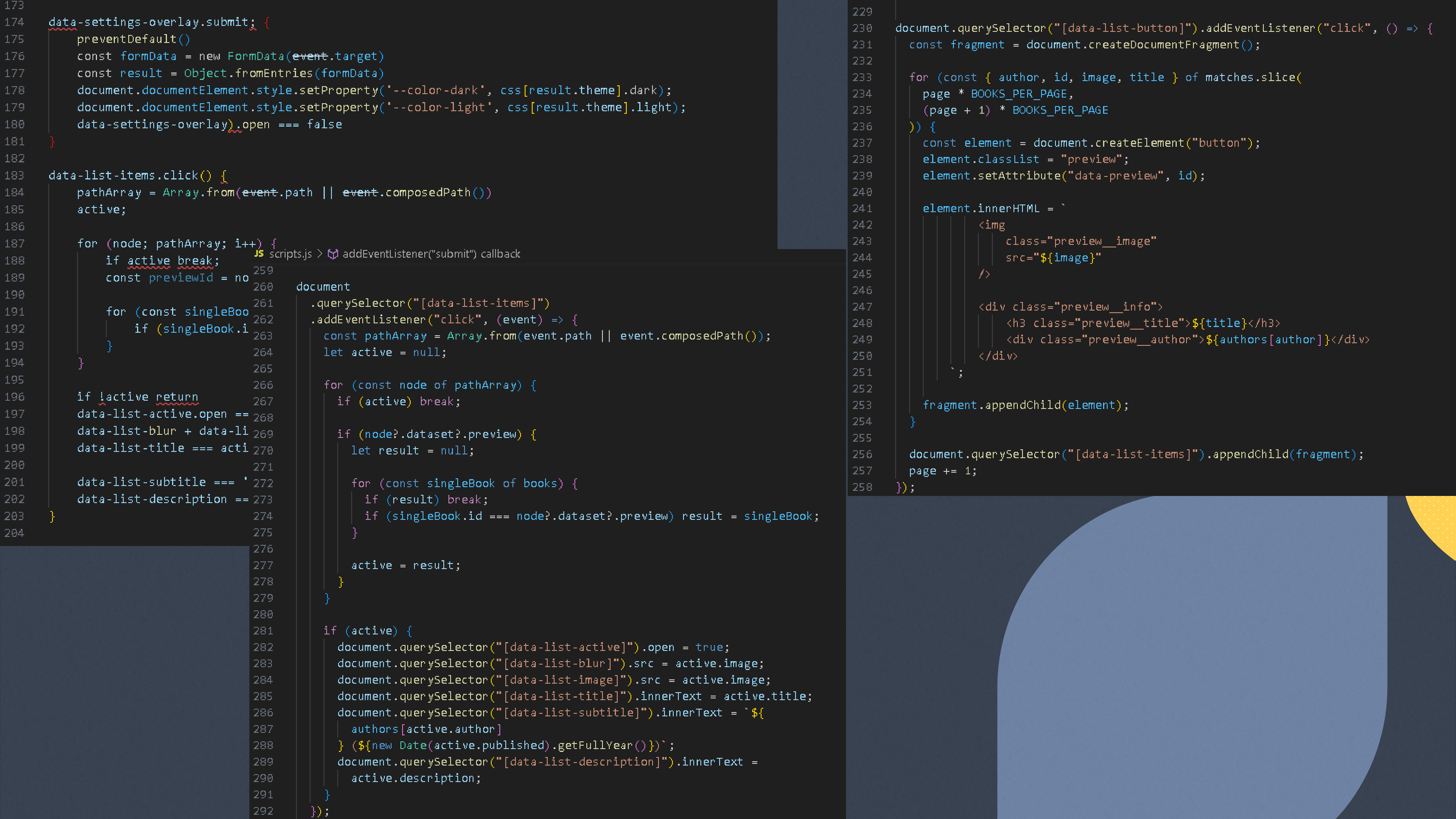


```
data-list-items.innerHTML = ''
const fragment = document.createDocumentFragment()
const extracted = source.slice(range[0], range[1])

for ({ author, image, title, id }; extracted; i++) {
    const { author: authorId, id, image, title } = props

    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[authorId]}</div>
        </div>
    `

    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>
```



JS scripts.js > ...

```
    if (theme === "night") {
      document.documentElement.style.setProperty(
        "--color-dark",
        "255, 255, 255"
      );
      document.documentElement.style.setProperty("--color-light", "10, 10, 20");
    } else {
      document.documentElement.style.setProperty("--color-dark", "10, 10, 20");
      document.documentElement.style.setProperty(
        "--color-light",
        "255, 255, 255"
      );
    }

    document.querySelector("[data-settings-overlay]").open = false;
  });

document
  .querySelector("[data-search-form]")
  .addEventListener("submit", (event) => {
    event.preventDefault();
    const formData = new FormData(event.target);
    const filters = Object.fromEntries(formData);
    const result = [];

    for (const book of books) {
      let genreMatch = filters.genre === "any";

      for (const singleGenre of book.genres) {
        if (genreMatch) break;
        if (singleGenre === filters.genre) {
          genreMatch = true;
        }
      }
```



```
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 = no

        for (const singleBoo
            if (singleBook.i
        }
    }

    if !active return
    data-list-active.open ==
    data-list-blur + data-li
    data-list-title === acti

    data-list-subtitle === '
    data-list-description ==
}
```

JS scripts.js > addEventListener("submit") callback

```
document
  .querySelector("[data-list-items]")
  .addEventListener("click", (event) => {
    const pathArray = Array.from(event.path || event.composedPath());
    let active = null;

    for (const node of pathArray) {
      if (active) break;

      if (node?.dataset?.preview) {
        let result = null;

        for (const singleBook of books) {
          if (result) break;
          if (singleBook.id === node?.dataset?.preview) result = singleBook;
        }

        active = result;
      }
    }

    if (active) {
      document.querySelector("[data-list-active]").open = true;
      document.querySelector("[data-list-blur]").src = active.image;
      document.querySelector("[data-list-image]").src = active.image;
      document.querySelector("[data-list-title]").innerText = active.title;
      document.querySelector("[data-list-subtitle]").innerText = `${
        authors[active.author]
      } (${new Date(active.published).getFullYear()})`;
      document.querySelector("[data-list-description]").innerText =
        active.description;
    }
  });
```

```

document.querySelector("[data-list-button]").addEventListener("click", () => {
  const fragment = document.createDocumentFragment();

  for (const { author, id, image, title } of matches.slice(
    page * BOOKS_PER_PAGE,
    (page + 1) * BOOKS_PER_PAGE
  )) {
    const element = document.createElement("button");
    element.classList = "preview";
    element.setAttribute("data-preview", id);

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

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

    fragment.appendChild(element);
  }

  document.querySelector("[data-list-items]").appendChild(fragment);
  page += 1;
});
```