TB_Mothibi DWA_6

- **Which were the three best abstractions, and why?**
  The first: Because it’s only responsible for the setting overlay. I mainly could found making functions for event listeners for buttons easy

```
      /**
       * Sets the settings button event listener.
       */
      function setSettingsButton() {
document.querySelector('[data-settings-overlay]').open = true;
      }
      document.querySelector('[data-header-settings]').addEventListener('click', setSettingsButton)
```

  The second: It didn’t have many complications

```
/**
 * Creates a preview element for a book.
 * @param {Object} book - The book object containing author, id, image, and title.
 * @returns {HTMLElement} The preview element.
 */
function createPreviewElement({ author, id, image, title }) {
    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>
    `;
    return element;
}
```

- **Which were the three worst abstractions, and why?**

  The first one: It’s a bit complex for me

```
/**
 * Populates the genres select element.
 * @returns {DocumentFragment} The document fragment containing the
genre options.
 */
function populateGenres() {
    const genreHtml = document.createDocumentFragment();
    const firstGenreElement = document.createElement('option');
    firstGenreElement.value = 'any';
    firstGenreElement.innerText = 'All Genres';
    genreHtml.appendChild(firstGenreElement);
    for (const [id, name] of Object.entries(genres)) {
        const element = document.createElement('option');
        element.value = id;
        element.innerText = name;
        genreHtml.appendChild(element);
    }
    return genreHtml;
}
/**
 * Populates the authors select element.
 * @returns {DocumentFragment} The document fragment containing the
author options.
 */
function populateAuthors() {
    const authorsHtml = document.createDocumentFragment();
    const firstAuthorElement = document.createElement('option');
    firstAuthorElement.value = 'any';
    firstAuthorElement.innerText = 'All Authors';
    authorsHtml.appendChild(firstAuthorElement);
    for (const [id, name] of Object.entries(authors)) {
        const element = document.createElement('option');
        element.value = id;
        element.innerText = name;
        authorsHtml.appendChild(element);
    }
    return authorsHtml;
}
```

TB_Mothibi DWA_6

The second one: But I did win at the end

```
      /**
function setThemeColors(theme) {
    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');
    }
  }


// Calls the setThemeColors
  if (window.matchMedia && window.matchMedia('(prefers-color-scheme:
dark)').matches) {
    document.querySelector('[data-settings-theme]').value = 'night';
    setThemeColors('night');
  } else {
    document.querySelector('[data-settings-theme]').value = 'day';
    setThemeColors('day');
  }



/**
 * Sets the settings form event listener.
 */
function setSettingsForm(event) {
    event.preventDefault();
    const formData = new FormData(event.target);
    const { theme } = Object.fromEntries(formData);
    setThemeColors(theme);
    document.querySelector('[data-settings-overlay]').open = false;
}
document.querySelector('[data-settings-form]').addEventListener('submit
', setSettingsForm)
```

TB_Mothibi DWA_6

The third: Show more works but I failed to display the showmore title when the page loads on the button and also the number of list remaining

```
/**
 * Updates the remaining count in the list button.
 */
function updateListButtonRemaining() {
    const remaining = Math.max(matches.length - (page *
BOOKS_PER_PAGE), 0);
    document.querySelector('[data-list-button]').innerHTML = `
        <span>Show more</span>
        <span class="list__remaining"> (${remaining})</span>
    `;
}
```

- **How can the three worst abstractions be improved via SOLID principles?**

  The first one: I used SRP because it's the only one i understand better

```
/**
 * Creates option elements for a dropdown menu.
 * @param {data} defaultValue - The default value of the dropdown.
 * @param {defaultOptionText} options - The options for the dropdown.
 */
function generateOptions(data, defaultOptionText) {
  const fragment = document.createDocumentFragment();
  const defaultOption = document.createElement('option');
  defaultOption.value = 'any';
  defaultOption.innerText = defaultOptionText;
  fragment.appendChild(defaultOption);

  for (const [id, name] of Object.entries(data)) {
    const option = document.createElement('option');
    option.value = id;
    option.innerText = name;
    fragment.appendChild(option);
  }

  return fragment;
}

// Appends options to the given container element
function appendOptions(container, options) {
  container.appendChild(options);
}

// Generates genre options
const genreHtml = generateOptions(genres, 'All Genres');
// Generates author options
const authorHtml = generateOptions(authors, 'All Authors');

// Appends genre options to the DOM
const genreContainer = document.querySelector('[data-search-genres]');
appendOptions(genreContainer, genreHtml);

// Appends author options to the DOM
const authorContainer =
document.querySelector('[data-search-authors]');
appendOptions(authorContainer, authorHtml);
```

The second one:

```
/**
 * Sets the theme colors in the DOM.
 * @param {string} theme - The theme to set ('day' or 'night').
 */
function setThemeColors(theme) {
  const isDarkMode = theme === 'night';
  const darkColor = isDarkMode ? '255, 255, 255' : '10, 10, 20';
  const lightColor = isDarkMode ? '10, 10, 20' : '255, 255, 255';
  document.documentElement.style.setProperty('--color-dark',
darkColor);
  document.documentElement.style.setProperty('--color-light',
lightColor);
}

/**
 * Sets the settings form event listener.
 * @param {Event} event - The form submission event.
 */
function setSettingsForm(event) {
  event.preventDefault();
  const formData = new FormData(event.target);
  const { theme } = Object.fromEntries(formData);
  setThemeColors(theme);
  document.querySelector('[data-settings-overlay]').open = false;
}

// Calls the setThemeColors based on the preferred color scheme
if (window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches) {
  document.querySelector('[data-settings-theme]').value = 'night';
  setThemeColors('night');
} else {
  document.querySelector('[data-settings-theme]').value = 'day';
  setThemeColors('day');
}

// Adds event listener to the settings form
document.querySelector('[data-settings-form]').addEventListener('submit', setSettingsForm);
```