Which were the three best abstractions, and why?

Data Object - for the DOM Elements it organises the main headings for each section of the code and makes it easier to access and simplifies the code.

```
 * @type {data}
 */
const data = {
    header: {
        search: document.querySelector("[data-header-search]"),
        settings: document.querySelector("[data-header-settings]"),
    },

    list: {
        items:document.querySelector("[data-list-items]"),
        message:document.querySelector("[data-list-message]"),
        button:document.querySelector("[data-list-button]"),
        active: document.querySelector("[data-list-active]"),
        blur: document.querySelector("[data-list-blur]"),
        image:document.querySelector("[data-list-image]"),
        title: document.querySelector("[data-list-title]"),
        subtitle: document.querySelector("[data-list-subtitle]"),
        description:document.querySelector("[data-list-description]"),
        close:document.querySelector("[data-list-close]"),
    },

    search: {
        overlay:document.querySelector("[data-search-overlay]"),
        form:document.querySelector("[data-search-form]"),
        title:document.querySelector("[data-search-title]"),
        genres: document.querySelector("[data-search-genres]"),
        authors: document.querySelector("[data-search-authors]"),
        cancel: document.querySelector("[data-search-cancel]"),

    },

    settings: {
        overlay: document.querySelector("[data-settings-overlay]"),
        form:document.querySelector("[data-settings-form]"),
        theme:document.querySelector("[data-settings-theme]"),
        cancel: document.querySelector("[data-settings-cancel]"),
    }
}
```

CSS Object - it stores the CSS colour values for the day and night themes.the code can easily switch between day and night themes by accessing the appropriate colour values from the css object.

```
/**css object has with 2 properties for night and day theme*/
const css = {
    day: ["255, 255, 255", "10, 10, 20"],
    night: ["10, 10, 20", "255, 255, 255"],
};
```

Which were the three worst abstractions, and why?

Used duplicate key-values

```
    // }
    option.theme = {
        night:document.documentElement.style.setProperty("--color-light",css[option.theme][0]),
        night:document.documentElement.style.setProperty("--color-dark", css[option.theme][1]) ,

        day: document.documentElement.style.setProperty( "--color-light",css[option.theme][0]),
        day: document.documentElement.style.setProperty( "--color-dark",css[option.theme][1]),
    }

    data.settings.overlay.close();
});
```

Used more than one fragment that serves the sames purpose

```
/**
 * create a dropdown list of the genres options
 */
const bookGenre = document.createDocumentFragment();
const theGenres = document.createElement("option");
theGenres.value = "any";
theGenres.textContent = "All Genres";
bookGenre.appendChild(theGenres);

const genreArray = Object.entries(genres);
for (let i = 0; i < genreArray.length; i++) {
    const [id, name] = genreArray[i];
    const genreOp = document.createElement("option");
    genreOp.value = id;
    genreOp.textContent = name;
    bookGenre.appendChild(genreOp);
}
data.search.genres.appendChild(bookGenre);

/**
 * creates a dropdown list of the author names options
 */
const bookAuthors = document.createDocumentFragment();
const theAuthors = document.createElement("option");
theAuthors.value = "any";
theAuthors.innerText = "All Authors";
bookAuthors.appendChild(theAuthors);

const authorArray = Object.entries(authors);
for (let i = 0; i < authorArray.length; i++) {
    const [id, name] = authorArray[i];
    const authOp = document.createElement("option");
    authOp.value = id;
    authOp.textContent = name;
    bookAuthors.appendChild(authOp);
}
data.search.authors.appendChild(bookAuthors);
```

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

Duplicate key values - using the **SRP**(Each module or class should be responsible for a single functionality or feature, and should not be coupled with other functionalities.) by separating the responsibility of setting theme colours from the conditional logic.The code still uses the conditional check to determine the theme, but now it calls the setThemeColors function with the theme and the css object

```
69
70    function setThemeColors(theme, css) {
71        document.documentElement.style.setProperty("--color-light", css[theme][0]);
72        document.documentElement.style.setProperty("--color-dark", css[theme][1]);
73      }
74
75      if (option.theme === "night") {
76        setThemeColors(option.theme, css);
77      } else {
78        setThemeColors(option.theme, css);
79      }
80
81    data.settings.overlay.close();
```

Duplicated fragments- Using the LSP we can take one function that could be created in order to generate a fragment with default content, and callbacks can be added where needed with different inputs to give the required outputs.