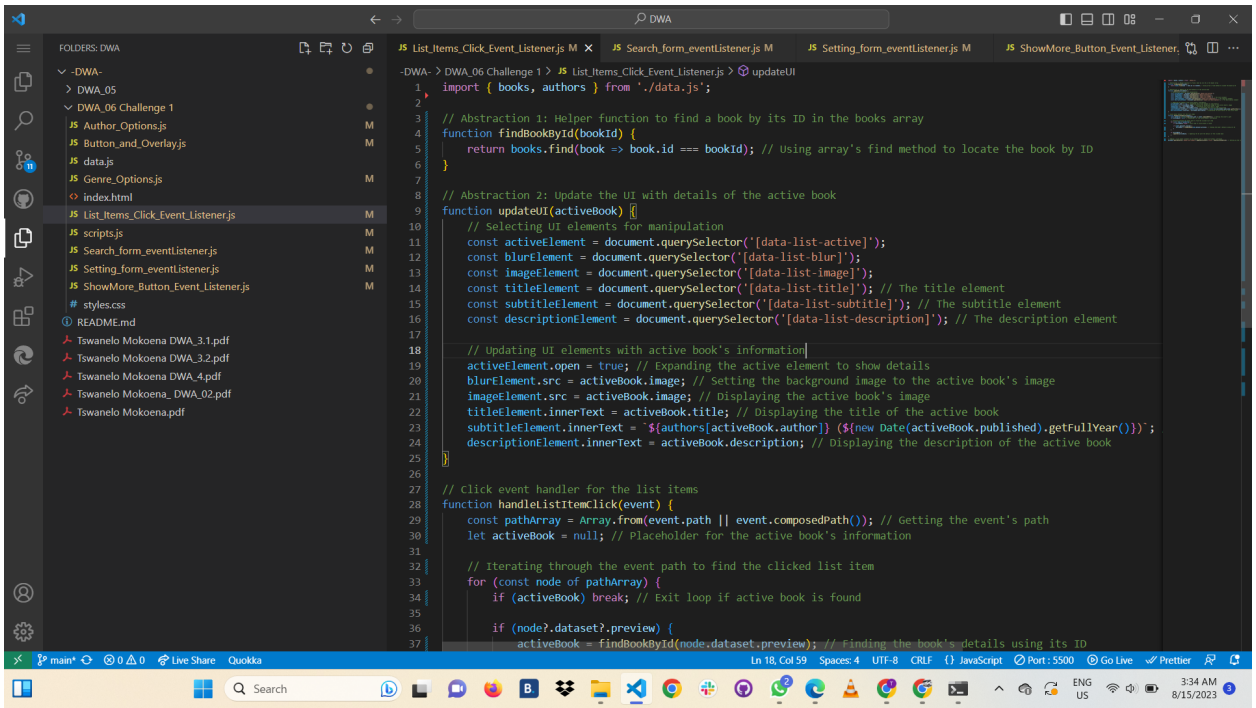
# DWA_07.4 Knowledge Check_DWA7

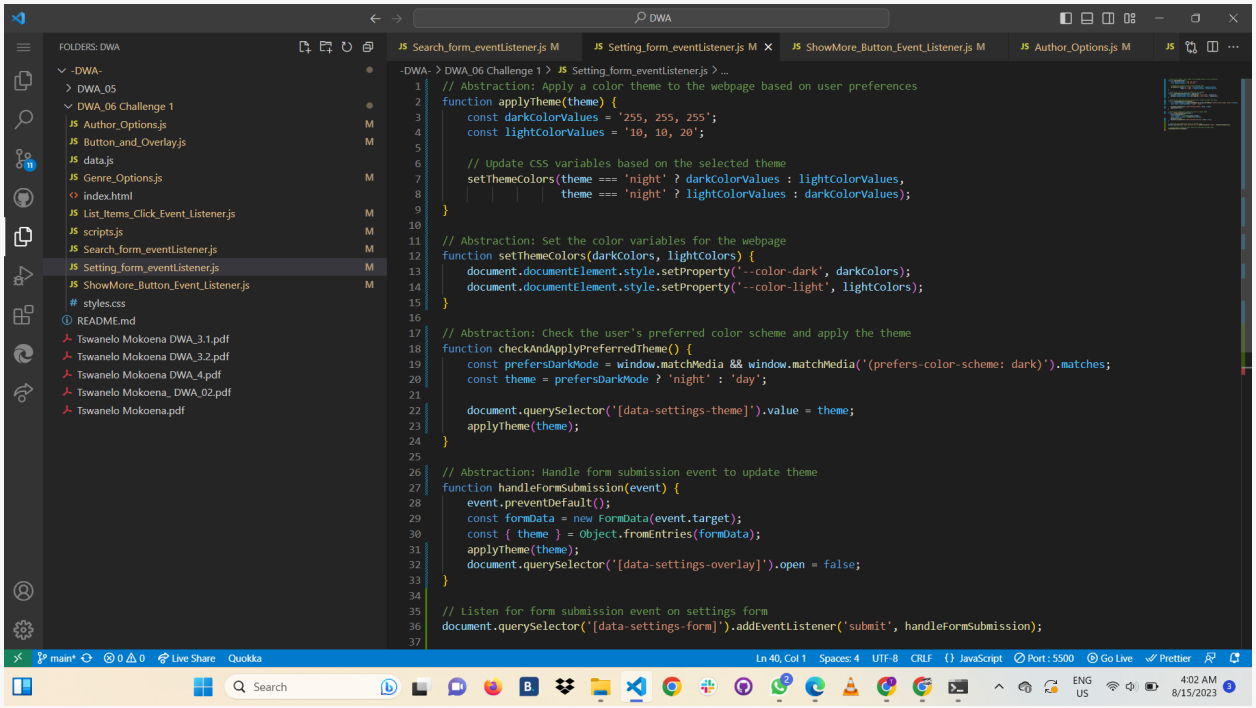
---

1. Which were the three best abstractions, and why?




Abstraction 1: The "**findBookById**" function does something special. It helps to find a book using its ID in the group of books. This special function makes the main part of the code look nicer and easier to understand. It also divides the different jobs of the code, making it easier to work on and keep in good shape.

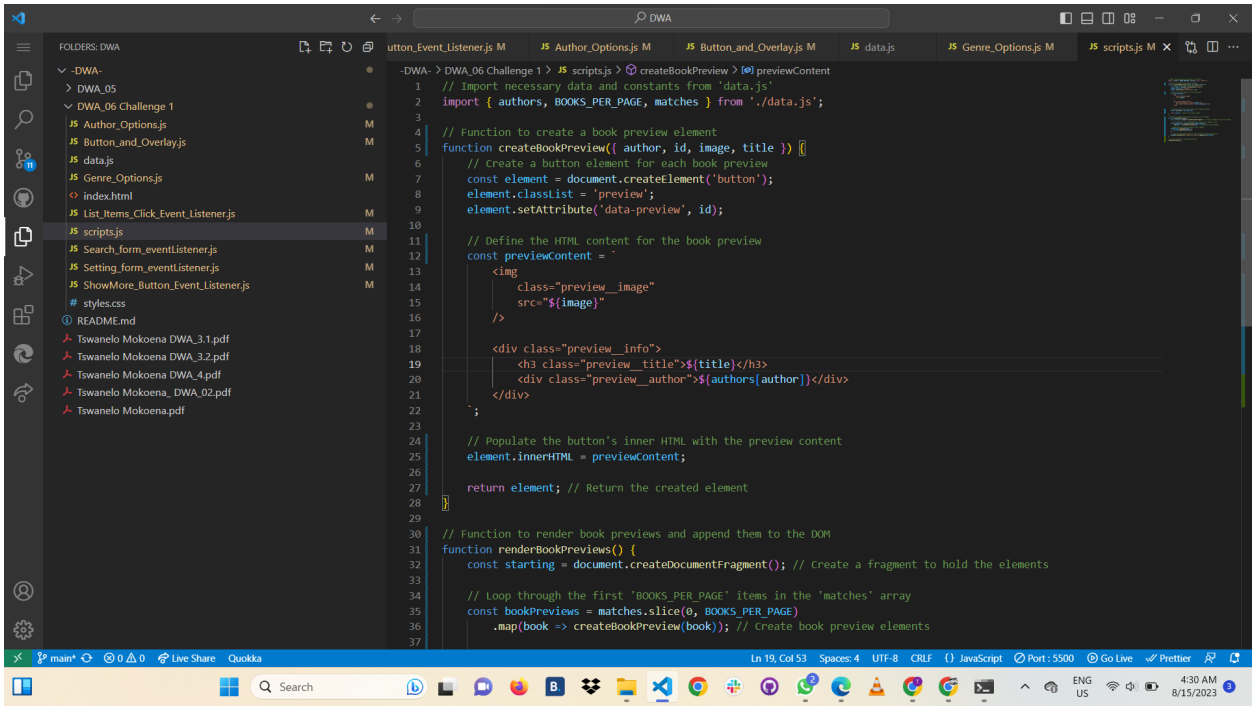
Abstraction 2: The **"updateUI"** function does something important. It puts information about a book that's being used right now into the parts you see on the screen. By putting this action into its own special function, the main set of instructions looks neater and is simpler to figure out. This also helps use the same code again later if we need to show similar updates in different parts of the app.




Using functions to group similar tasks together in the code makes it easier to understand. This way, the code is easier to read and maintain, and it helps to keep different tasks separate from each other.

---

2. Which were the three worst abstractions, and why?




**Making Code Easier**: Simplifying DOM Changes The function renderBookPreviews does two things: it creates book preview pieces and adds them to the webpage. This can be clearer if we split these tasks. If we do this, the code can be simpler to understand and test.

**Fixed Selection in Code:** Be Careful In the function renderBookPreviews, we have a fixed selector **[data-list-items]**. This finds where the book previews should go on the page. But if the webpage's structure changes, this code might not work. To avoid future problems, it's better to give the selector as a special word when using the function.

**Hidden Link to 'data.js' Module:** The code directly takes certain things (like authors, BOOKS_PER_PAGE, matches) from the 'data.js' module. This is fine for small projects, but it makes the code very connected to the data. If the data changes, the code might

stop working. It could be better to hide the data behind a special tool that the code can talk to. This would also help with testing and let us switch data sources more easily.

---

3. How can The three worst abstractions be improved via SOLID principles.

> **Single Responsibility Principle (SRP)**: The function "createBookPreview" does two things: it makes the structure of a web page and adds details about a book to it. It's better to do these things separately.
>
> **Open/Closed Principle (OCP):** The current code can't easily add new features without changing existing parts. We can make it easier to add new things by adding a special layer.
>
> **Dependency Inversion Principle (DIP)**: The current code relies directly on information about authors and a fixed number for books per page. It's better to give this information to the code from the outside, instead of using fixed values.

These changes use SOLID ideas to make my code easier to work with. SOLID is like advice to make my code more organized and flexible. It's not a strict rule, and I can decide how much to use based on my program.

---