# DWA\_08 Discussion Questions

In this module you will continue with your “Book Connect” codebase, and further iterate on your abstractions. You will be required to create an encapsulated abstraction of the book preview by means of a single factory function. If you are up for it you can also encapsulate other aspects of the app into their own abstractions.   
  
To prepare for your session with your coach, please answer the following questions. Then download this document as a PDF and include it in the repository with your code.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. What parts of encapsulating your logic were easy?

**Creating functions simplifies code reuse and isolation of specific tasks. Using Objects to group related data and methods, enhancing organization and encapsulation. Using modules provides a clear separation of concerns and improves code organization and maintainability.**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. What parts of encapsulating your logic were hard?

Understanding scope, context, and asynchronous code management. Maintaining encapsulation, modularity, and applying object-oriented principles. Choosing and implementing appropriate design patterns.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Is abstracting the book preview a good or bad idea? Why?

Abstracting the book preview can be a good idea, depending on the context and requirements of your application. By abstracting the book preview functionality into a separate component or function, you can reuse it throughout your application wherever book previews are needed. This reduces code duplication and promotes consistency. Abstracting allows you to encapsulate the logic and presentation of the book preview. This makes your codebase more modular and easier to maintain, as changes or updates to the preview can be isolated without affecting other parts of the application. It helps separate the concerns of displaying a book preview from other functionalities. This separation enhances code clarity and makes it easier to understand and modify in the future. Abstracting the book preview prepares your codebase for future enhancements or additions. As your application grows, having modular components makes it easier to scale and extend functionality without introducing complexity.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_