DWA_01.3 Knowledge Check_DWA1

---

1. Why is it important to manage complexity in Software?

Managing complexity in software is crucial because it helps improve development efficiency, maintainability, and overall system reliability. By reducing complexity, software becomes easier to understand, test, debug, and modify, leading to fewer errors and better overall performance

---

2. What are the factors that create complexity in Software?

1.Size and scope: Large-scale projects with numerous features and extensive codebases tend to be more complex.
2. Dependencies: Complex software often relies on external libraries, frameworks, and APIs, increasing the intricacy of integration and maintenance.
3.Interactions and communication: Systems with multiple components or modules that need to interact and exchange data introduce complexity.
Business requirements: Ambiguous or changing requirements can lead to complex software design and implementation.

---

3. What are ways in which complexity can be managed in JavaScript?

Abstraction and encapsulation: Using object-oriented programming principles, such as encapsulation and abstraction, allows hiding complex implementation details and exposing simpler interfaces.

---

4. Are there implications of not managing complexity on a small scale?

Code readability and maintainability suffer: Complex code becomes difficult to understand and modify, making it harder for developers to work with it. This leads to increased development time and potential introduction of bugs.

Increased risk of errors: Complexity often leads to a higher likelihood of introducing errors or bugs. Without proper management, it becomes challenging to identify and fix these issues, resulting in reduced software reliability.

---

5. List a couple of codified style guide rules, and explain them in detail.

Commenting and documentation:

Rule: Include comments to explain complex code logic, clarify assumptions, and provide high-level explanations of functions or classes.
Explanation: Comments and documentation help other developers understand the purpose and behavior of code sections. They provide insights into the reasoning behind specific design choices or algorithms. Well-documented code promotes maintainability, simplifies debugging, and makes it easier to onboard new team members.
Code organization:

Rule: Group related code together and maintain a logical structure within files and directories.
Explanation: Organizing code in a structured manner improves code maintainability and navigability. Grouping related code together makes it easier to find specific functionality

and reduces the likelihood of duplicated code. Logical file and directory structures help developers locate and modify code components efficiently.
Error handling:

Rule: Properly handle exceptions and errors, including logging and appropriate error messages.
Explanation: Implementing robust error handling improves software reliability and user experience. Catching and handling errors gracefully prevents application crashes and provides meaningful feedback to users or developers when errors occur. Proper logging assists in identifying and debugging issues, aiding troubleshooting efforts.

---

6. To date, what bug has taken you the longest to fix - why did it take so long?

It was a bug where I thought I installed the npm package but turn out I installed the wrong version and was using react icons but the icon I wanted to use was outdated .I thought that the computer was wrong I guess .
