DWA_01.3 Knowledge Check_DWA1

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

Software complexity management is essential for increasing system stability, maintainability, and development efficiency. Software can be made simpler to comprehend, test, debug, and modify, which reduces errors and improves performance overall.

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

- Interactions and communication: Complexity is introduced into systems when there are numerous modules or components that must communicate and exchange data. Business needs: Complex software design and execution might result from ambiguous or shifting requirements.

- Size and scope: Complexity is often higher in large-scale projects with plenty of features and large codebases.

- Dependencies: The complexity of integration and maintenance increases when complex software depends on external libraries, frameworks, and APIs.

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

Encapsulation and abstraction: By utilizing object-oriented programming concepts, such as encapsulation and abstraction, it is possible to conceal intricate implementation details and reveal more straightforward interfaces.

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

Code readability and maintainability suffer: Complex code becomes more challenging for developers to work with since it is more difficult to comprehend and alter. This results in longer development times and the possible introduction of errors.

Error risk: Complexity frequently increases the chance of introducing faults or errors. Reducing software reliability is the outcome of finding and fixing these problems, which are more difficult in the absence of adequate management.

---

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

Commenting and documentation:
Rule: Use comments to clarify assumptions, give high-level explanations of methods or classes, and explain complex code logic. An explanation of the function and behaviour of code parts is provided by comments and documentation for other developers. They give light on the rationale behind particular algorithmic or design decisions. Debugging becomes easier, maintainability is encouraged, and onboarding new team members is facilitated by well-documented code.

Code organization:
Rule: Keep files and directories logically organised and group relevant code together. Justification: Maintaining and navigating code is made easier when it is organised. Reducing the possibility of duplicate code and simplifying the search for specific functionality are two benefits of grouping relevant code together. Developers can find and edit code components more quickly and easily when they follow logical file and directory structures.

---

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