**How to Know If Your Code Is Clean?** Clean code should be simple, readable, maintainable, testable, and elegant. It must follow the single responsibility principle and be enjoyable to read, almost like poetry. Here are 16 guidelines to help you write clean code: 1. **Use Names Having a Purpose** – Always use descriptive names for variables, functions, classes, and modules to convey intent clearly. 2. **Create Small Functions That Do a Single Task** – Small, focused functions are easier to read, test, and reuse. 3. **Write Code That People Can Understand** – Format code properly with clear indentation, whitespace, and consistent structure; document it where necessary. 4. **To Make a Legible Code, Use Empty Lines** – Use empty lines to separate logical blocks and enhance readability. 5. **A Function Should Not Have More Than Three Parameters** – Limit parameters to improve clarity; if more are needed, pass an object. 6. **Procedures Should Only Do One Task** – Avoid functions that perform multiple unrelated operations; keep each function focused. 7. **Repetition of Codes** – Eliminate repetitive blocks by abstracting common code into reusable functions. 8. **Capitalize Fixed Values (SNAKE UPPER CASE)** – Constants should be written in uppercase with underscores separating words. 9. **Be Consistent** – Use a consistent coding style and tools like linters to enforce standards (e.g., Standard JS or PEP8). 10. **Encapsulation + Modularization** – Group related functions and properties together; split large codebases into manageable modules. 11. **The DRY Principle** – Don't repeat yourself; extract reusable logic into functions, but avoid over-abstraction. 12. **Avoid Writing Unnecessary Comments** – Prefer self-explanatory code; update or remove outdated comments to avoid confusion. 13. **Write Unit Test to Improve Code Quality** – Unit tests help maintain code quality, ease refactoring, and prevent regressions; consider Test-Driven Development (TDD). 14. **Make Your Project Well Organized** – Structure folders and files logically so others can easily understand and navigate the project. 15. **Single Responsibility Principle (SRP)** – Ensure each function or module has one reason to change by doing only one job. 16. **Avoid Noise Words** – Eliminate unnecessary words like "Info", "Data", "Object", or "Manager" in names; they add no real value.