

Day 03

Assignment 1: Create an infographic illustrating the Test-Driven Development (TDD) process. Highlight steps like writing tests before code, benefits such as bug reduction, and how it fosters software reliability.

Solution :- Test-Driven Development (TDD) is a software development approach where tests are written before the code itself. This iterative process ensures high-quality, well-tested software.

The TDD Cycle:

1. **Red:** Write a failing test that defines the desired functionality. This test initially fails because the code to fulfil that functionality hasn't been written yet.
2. **Green:** Write the minimum amount of code to make the failing test pass. This ensures the code directly addresses the functionality you're testing.
3. **Refactor:** Without changing functionality, improve the code's readability, maintainability, and efficiency. This keeps your code clean and easy to understand in the long run.

Benefits of TDD:

- **Reduced Bugs:** Writing tests upfront helps identify and fix potential issues early in the development process, leading to more robust software.
- **Clear Requirements:** The act of writing tests clarifies the desired behaviour of the code, promoting better understanding of requirements.
- **Confidence in Changes:** Existing tests act as a safety net, giving you confidence to refactor code or add new features without introducing regressions (introducing bugs in existing functionality).
- **Improved Design:** TDD encourages writing smaller, more focused code units, leading to a well-designed and maintainable codebase.
- **Faster Debugging:** Failing tests pinpoint exactly where the issue lies in your code, saving time and frustration during debugging.

Building Reliability:

By continuously writing failing tests and then making them pass with clean code, TDD fosters a software development process that emphasises quality from the very beginning. This iterative cycle helps create a foundation of reliable and well-tested software, ready for future enhancements and user needs.

Assignment 2: Produce a comparative infographic of TDD, BDD, and FDD methodologies. Illustrate their unique approaches, benefits, and suitability for different software development contexts. Use visuals to enhance understanding.

Solution :-

Development Approach	TDD	BDD	FDD
Visual Metaphor	Writing failing tests to guide development.	Collaboration on user stories defining desired behavior	Breaking down features into manageable tasks
Focus	Code quality and functionality.	User-centric development and collaboration	Project planning and efficient development cycles
Strengths	<ul style="list-style-type: none"> * Reduced bugs * Clear requirements * Improved design * Faster debugging 	<ul style="list-style-type: none"> Early user feedback * Improved communication * Stakeholder alignment * Focus on business value 	<ul style="list-style-type: none"> Efficient task management * Reduced rework * Improved team communication * Predictable delivery

Difference between TDD, BDD and FDD