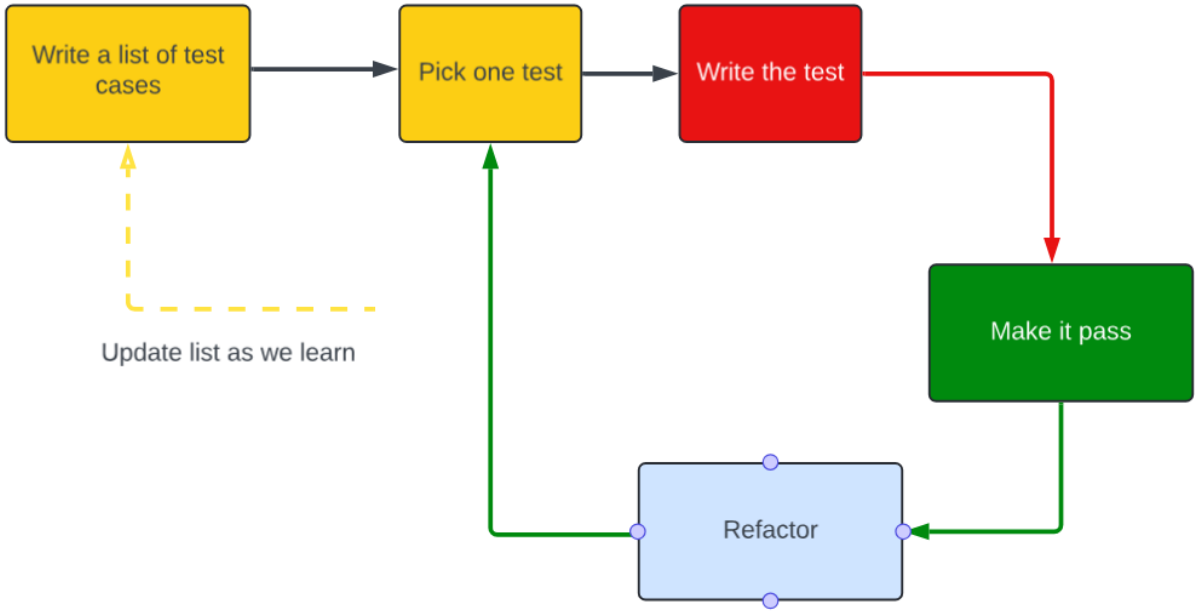
hariharane01@gmail.com

SDLC-Assignment 02

Question 01: 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.

Answer:

Test-Driven Development (TDD) Process Infographic



1. Write Test:

- Write a failing test based on the desired functionality.
- Test should be written before any code is implemented.

2. Run Test:

- Execute the failing test.
- Test should fail since no code has been implemented yet.

3. Write Code:

- Implement the minimum amount of code to pass the failing test.
- Keep the code simple and focused on passing the test.

4. Run Test Again:

- Execute the test suite again to validate the implemented code.
- Test should pass if the code has been implemented correctly.

5. Refactor Code:

- Refactor the code to improve readability, performance, or maintainability.
- Ensure all tests still pass after refactoring.

Benefits of TDD:

- Bug Reduction: TDD helps identify and fix bugs early in the development process, reducing the likelihood of bugs in the final product.
- Improved Software Reliability: By writing tests before code, TDD ensures that the codebase is thoroughly tested, leading to more reliable software.

Question 02: 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.

Answer:

1. Test-Driven Development (TDD):

Approach:

- Write tests before writing code.
- Focus on small, incremental steps.
- Red, Green, Refactor cycle.

Benefits:

- Early bug detection and prevention.
- Improved code quality and design.
- Fast feedback loop.

Suitability:

- Suitable for projects with well-defined requirements.
- Ideal for developers who prefer a bottom-up approach.

2. Behavior-Driven Development (BDD):

Approach:

- Focus on behavior and outcomes.
- Use natural language specifications (Given-When-Then).
- Collaboration between developers, testers, and stakeholders.

Benefits:

- Improved communication between stakeholders.
- Encourages shared understanding of requirements.
- Drives development from a user's perspective.

Suitability:

- Suitable for projects with complex business logic.
- Ideal for teams with diverse stakeholders and roles.

3. **Feature-Driven Development (FDD):**

Approach:

- Divide project into manageable features.
- Develop features iteratively.
- Emphasize design and architecture.

Benefits:

- Focus on delivering tangible, working features.
- Encourages frequent integration and testing.
- Scalable for large and complex projects.

Suitability:

- Suitable for large-scale projects with evolving requirements.
- Ideal for teams with strong technical expertise.