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.

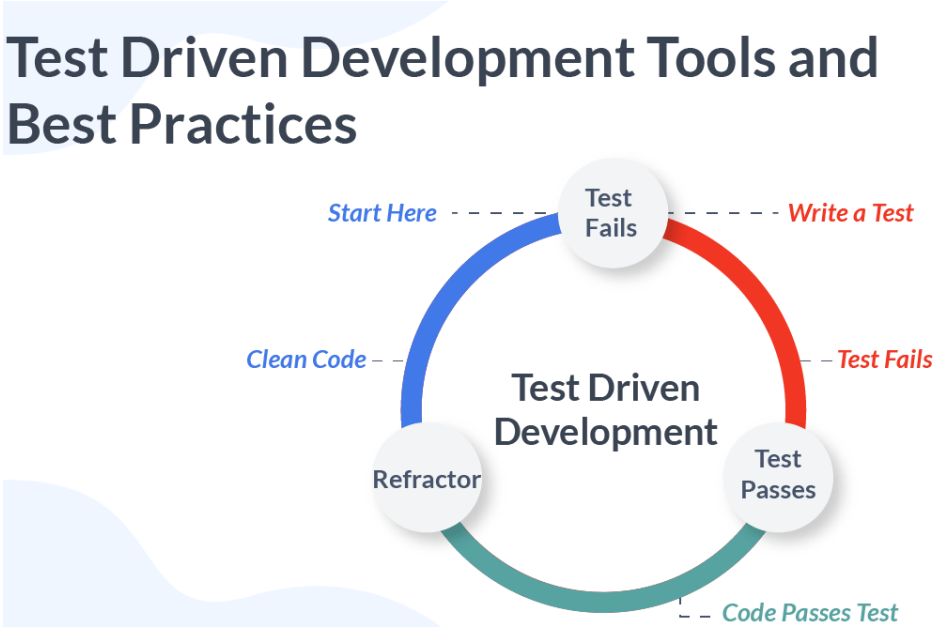
Title: Test-Driven Development (TDD) Process

1. Introduction to TDD

- Definition: TDD is a software development approach where tests are written before the code is implemented.
- Objective: Ensure code reliability, reduce bugs, and streamline the development process.

2. Key Steps of TDD

- Write Test: Begin by writing a test defining desired functionality.
- Run Test: Execute the test to ensure it fails initially.
- Write Code: Develop code to pass the test.
- Run Test Again: Re-run the test suite to confirm the functionality.
- Refactor Code: Optimize code without changing its functionality.
- Repeat: Continue the cycle for new features and refactor as needed.



3. Benefits of TDD

- Bug Reduction: Early bug detection leads to fewer bugs in the final product.
- Improved Code Quality: Promotes modular, well-structured code for easier maintenance.
- Faster Development: Reduces debugging time, speeding up overall development.
- Enhanced Software Reliability: Thorough testing ensures software behaves as expected.

4. Conclusion

TDD is a powerful approach for building high-quality software through iterative testing, implementation, and refinement.

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.

Comparative Infographic: TDD vs BDD vs FDD Methodologies

1. Test-Driven Development (TDD)

- Approach:

  Tests are written before the code is implemented.

  Focuses on writing small, incremental tests to drive development.

- Benefits:

  Early bug detection leads to higher code reliability.

  Promotes modular and well-structured code.

  Reduces debugging time and speeds up development.

- Suitability:

  Ideal for projects with evolving requirements and where code quality is crucial.

2. Behavior-Driven Development (BDD)

- Approach:

  Focuses on defining system behavior using scenarios or examples.

  Encourages collaboration between developers, testers, and stakeholders.

- Benefits:

  Ensures alignment with stakeholder expectations.

  Promotes a shared understanding of system behavior.

  Improves communication and collaboration among team members.

- Suitability:

  Best suited for projects with complex business logic and extensive stakeholder involvement.

3. Feature-Driven Development (FDD)

- Approach:

  Breaks down development into manageable feature sets.

  Emphasizes iterative and incremental delivery of features.

- Benefits:

  Enhances project visibility and control through feature-based planning.

  Facilitates early identification and resolution of issues.

  Promotes a disciplined and structured approach to development.

- Suitability:

  Well-suited for large-scale projects with a focus on feature delivery and incremental development.

Comparison between TDD, BDD, FDD:

Aspect	TDD	BDD	FDD
Approach	Tests written before code.	Behavior-driven scenarios or examples.	Development based on feature sets.
Focus	Code implementation details.	System behavior and stakeholder alignment.	Feature delivery and incremental development.
Benefits	Early bug detection, code reliability.	Stakeholder alignment, improved communication.	Enhanced project visibility, iterative delivery.
Suitability	Evolving requirements, code quality.	Complex business logic, stakeholder involvement.	Large-scale projects, feature-based planning.