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.

1. Test-Driven Development (TDD)

Approach:

- Write tests before writing the actual code.
- Follow a cycle: Red (write a failing test) -> Green (write the minimum code to pass the test) -> Refactor (improve the code while keeping tests green).

Benefits:

- Ensures code quality and reliability.
- Facilitates cleaner, more maintainable code.
- Provides clear documentation through tests.

Suitability:

- Best for projects requiring high reliability.
- Suitable for complex algorithms and business logic.
- Ideal for environments with rigorous regression testing needs.

Visual:

- **Cycle:**
- **Red:** Write Test ->
- **Green:** Implement Code to Pass Test ->
- **Refactor:** Improve Code
- **Benefits:** Code Quality, Clean Code, Documentation

2. Behavior-Driven Development (BDD)

Approach:

- Define behavior in plain English using scenarios.
- Focus on the behavior of the application from the end user's perspective.
- Use Gherkin syntax for writing scenarios: Given, When, Then.

Benefits:

- Improves communication among stakeholders.
- Ensures understanding of requirements before development.
- Facilitates automated acceptance testing.

Suitability:

- Ideal for projects with non-technical stakeholders.
- Suitable for complex business requirements.
- Best for enhancing collaboration between developers, testers, and business analysts.

Visual:

Flow:

- **Given:** Initial context
- **When:** Action or event
- **Then:** Expected outcome
- **Benefits:** Improved Communication, Requirement Clarity, Automated Acceptance Testing

3. Feature-Driven Development (FDD)

Approach:

- Develops software based on features.
- Features are small, client-valued functions.

- Follows a five-step process: Develop Overall Model, Build Feature List, Plan by Feature, Design by Feature, Build by Feature.

Benefits:

- Focuses on delivering tangible, working software regularly.
- Scales well for larger teams and projects.
- Encourages detailed planning and design.

Suitability:

- Best for large-scale projects with clear feature sets.
- Suitable for teams needing regular, feature-based progress.
- Ideal for environments where detailed documentation and planning are necessary.

Visual:

Process:

1: Develop Overall Model

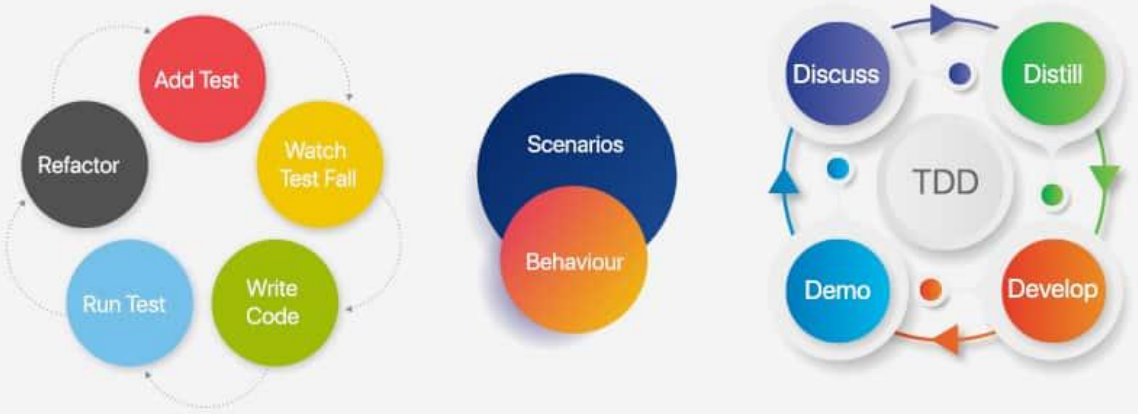
2: Build Feature List

3: Plan by Feature

4: Design by Feature

5: Build by Feature

Difference between TDD BDD and ATDD



TDD BDD ATDD

Benefits: Regular Deliverables, Scalable, Detailed Planning