Day 3 – Assignment 2

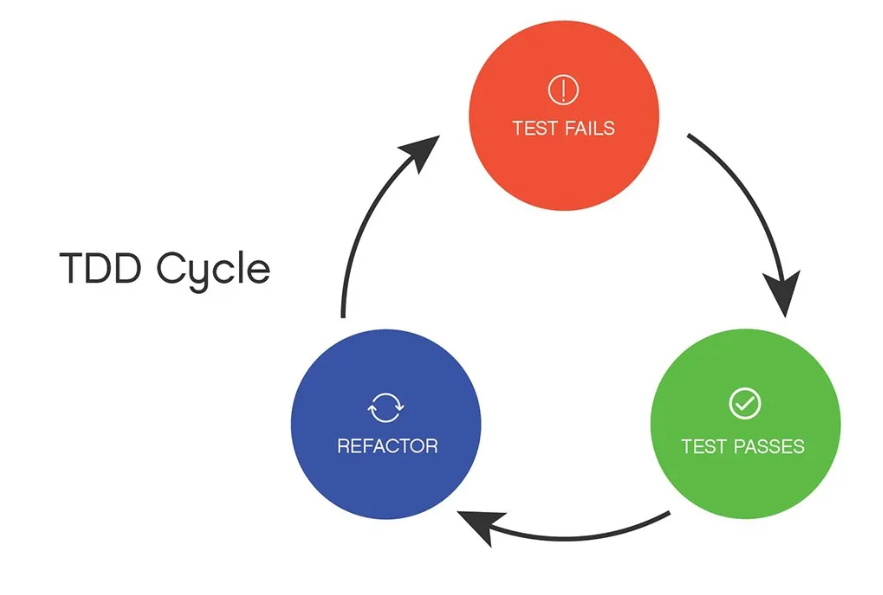
* Produce a comparative infographic of TDD, BDD, and FDD methodologies. Illustrate their unique approaches, benefits, and suitability for different software development contexts. Use visuals to enhance understanding.
* **Software Development Methodologies: TDD vs. BDD vs. FDD**

1. **TDD (Test-Driven Development):**

A software development process where tests are written before the code itself.

* **Steps:**

1. **Write a Test:** Create a test for the next piece of functionality you intend to add.
2. **Run the Test:** Execute the test to ensure it fails (since the functionality isn’t implemented yet).
3. **Write Code:** Write the minimum amount of code required to pass the test.
4. **Run the Test:** Run the test again to see if it passes.
5. **Refactor:** Improve the code without changing its functionality, ensuring all tests still pass.



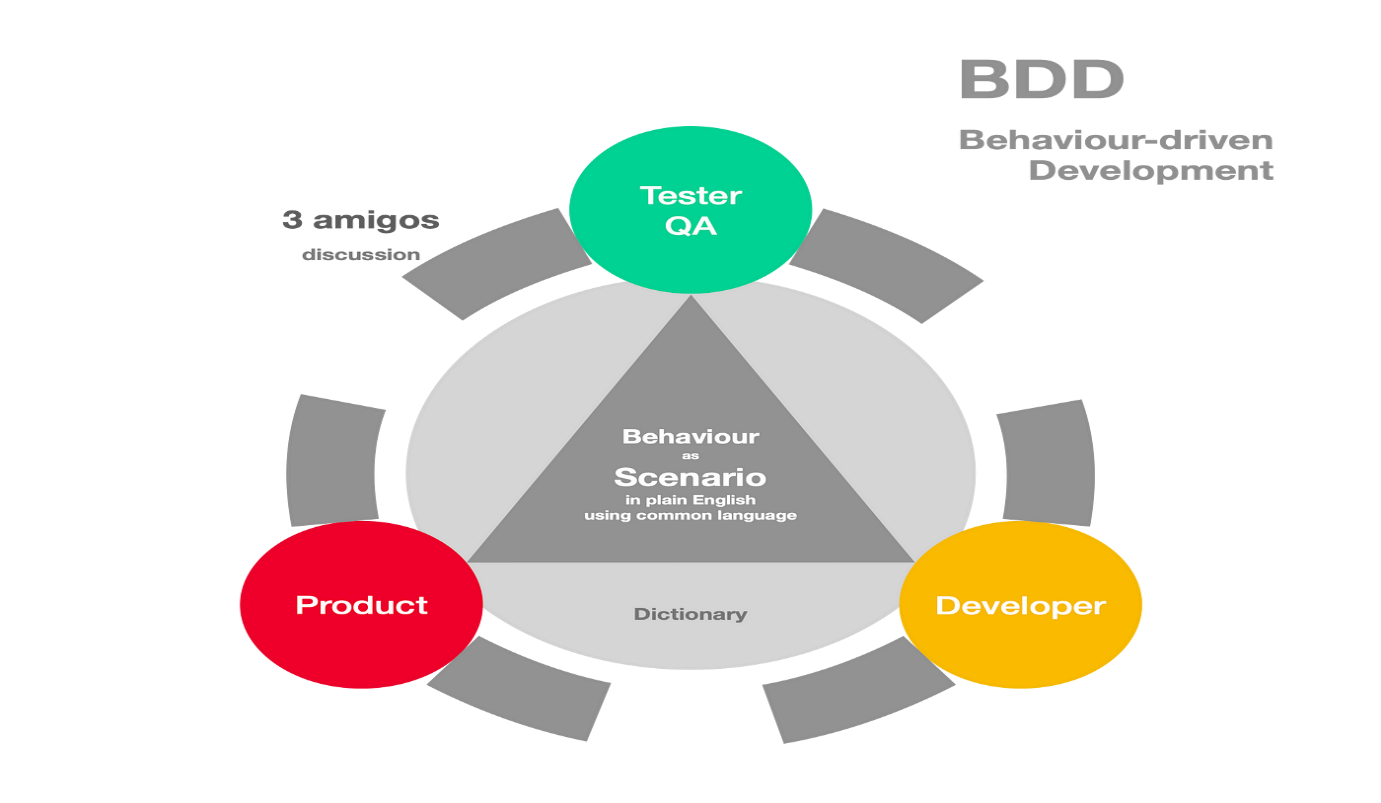
* **Benefits:**
* **Bug Reduction:** Early defect detection and resolution.
* **Software Reliability:** Consistent testing ensures code reliability.
* **Refactoring Confidence:** Safe and easy to refactor with comprehensive tests.
* **Suitability for Different Contexts**
* **Best for:** Projects requiring high reliability and maintainability.
* **Suitable for:** Small to medium-sized projects.

1. **BDD (Behavior-Driven Development):**

An extension of TDD that emphasizes collaboration between developers, testers, and business stakeholders by defining tests in natural language.

* **Steps:**

1. **Write Behavior Specifications:** Define the desired behavior using Gherkin language (Given-When-Then format).
2. **Implement Behavior:** Write the code to fulfill the behavior described in the specifications.
3. **Verify Behavior:** Run tests to verify the behavior meets the specifications.

****

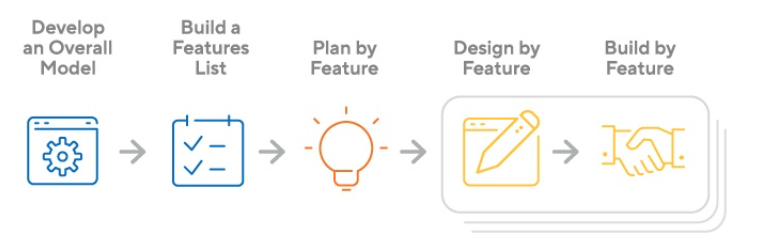
* **Benefits:**
* **Collaboration:** Enhances collaboration among developers, testers, and non-technical stakeholders.
* **Clarity:** Clear understanding of requirements and expected behavior.
* **User-Centric:** Focuses on user behavior and outcomes.
* **Suitability for Different Contexts**
* **Best for:** Projects needing strong collaboration and clear requirements.
* **Suitable for:** Projects with complex user interactions and multiple stakeholders.

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

An iterative and incremental software development methodology focused on delivering features that add business value.

* **Steps:**

1. **Develop an Overall Model:** Create a high-level model of the system.
2. **Build a Feature List:** Identify and prioritize the features to be developed.
3. **Plan by Feature:** Create a plan for developing each feature.
4. **Design by Feature:** Design the architecture and components needed for each feature.
5. **Build by Feature:** Develop and implement each feature.



* **Benefits:**
* **Scalability:** Suitable for large projects with multiple features.
* **Structured:** Provides a clear structure and detailed planning.
* **Feature Focus:** Prioritizes features that deliver business value.
* **Suitability for Different Contexts**
* **Best for:** Large-scale projects with many features.
* **Suitable for:** Projects with a need for detailed planning and structured progress.