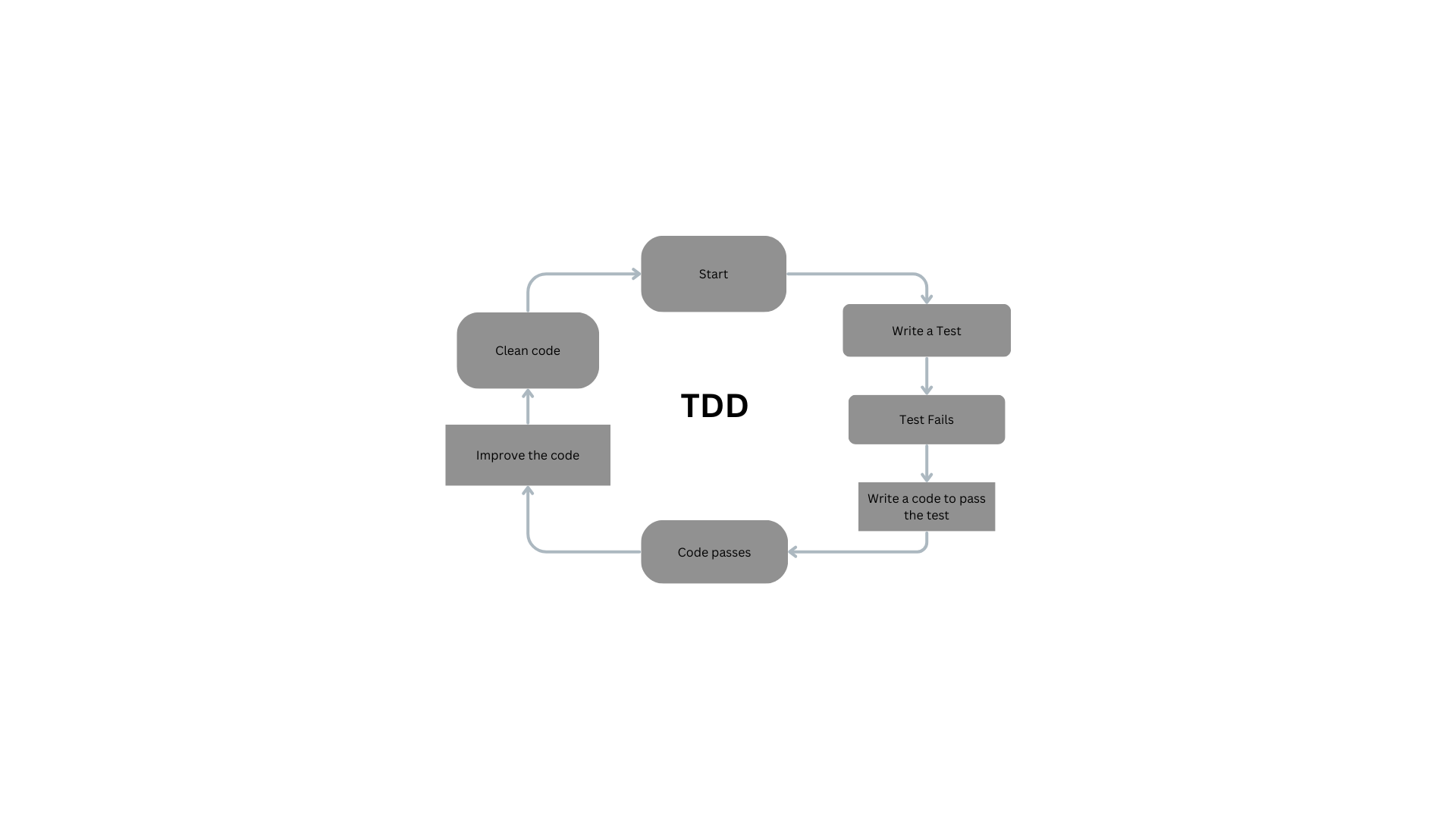
****Assignment 1: Create an infographic illustrating the Test-Driven Development (TDD) process. Highlight steps like writing tests before code, benefits such as bug reduction, and how it fosters software reliability

**Test-Driven Development (TDD) Process**

1. **Write Test:**
   * Create a failing test case based on a specific requirement or functionality.
2. **Run Test:**
   * Execute the test case to confirm it fails, indicating the absence of the intended functionality.
3. **Write Code:**
   * Develop the minimal code required to pass the failing test.
4. **Run Test Again:**
   * Execute the test suite to verify the newly implemented code.
5. **Refactor Code:**
   * Improve code quality and maintainability without altering its functionality.
6. **Run Test Once More:**
   * Re-run the test suite to ensure the code changes haven't introduced regressions.

**Benefits of Test-Driven Development (TDD)**

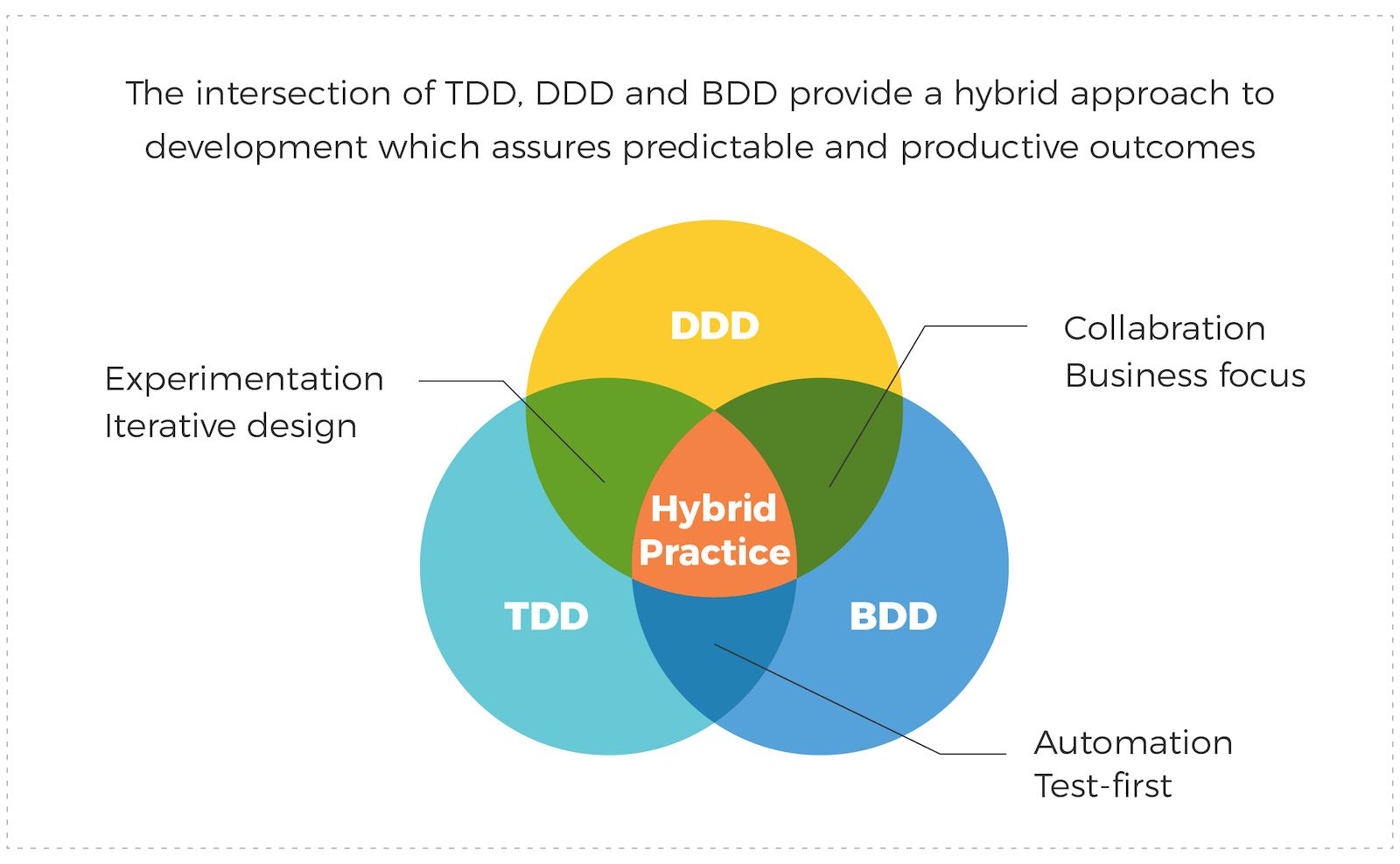
* **Bug Reduction:**
  + Early detection and resolution of defects lead to fewer bugs in the final product.
* **Improved Code Quality:**
  + Ensures code is well-structured, modular, and adheres to specifications.
* **Faster Development Cycles:**
  + Reduces debugging time and speeds up development iterations.
* **Enhanced Software Reliability:**
  + Continuous testing fosters confidence in the software's functionality and performance.

**TDD Software Reliability**

* **Incremental Development:**
  + Builds software in small, manageable increments, reducing the risk of introducing large-scale defects.
* **Continuous Testing:**
  + Ensures each code change is thoroughly validated, maintaining software integrity throughout development.
* **Feedback Loop:**
  + Provides immediate feedback on code changes, facilitating early identification and resolution of issues.

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.

**TDD, BDD and FDD Methodologies**



**Introduction**

* **Brief Overview**: Introduction to TDD, BDD, and FDD, and their importance in software development.

**2. Methodologies**

**Test-Driven Development (TDD)**

* **Approach**: Write tests before code.
* **Process**: Write a test -> Run the test -> Write code -> Run the test -> Refactor -> Repeat.
* **Visual**: Cycle diagram showing the TDD steps.

**Behaviour-Driven Development (BDD)**

* **Approach**: Write specifications based on behaviour.
* **Process**: Define behaviour in plain language -> Write tests based on behaviour -> Develop code -> Run tests -> Refactor -> Repeat.
* **Visual**: Flowchart showing steps from behaviour definition to testing and refactoring.

**Feature-Driven Development (FDD)**

* **Approach**: Develop by features.
* **Process**: Develop overall model -> Build feature list -> Plan by feature -> Design by feature -> Build by feature.
* **Visual**: Step-by-step diagram illustrating the FDD process.

**3. Benefits**

**TDD Benefits**

* **Bug Reduction**: Early detection and fixing.
* **Reliable Codebase**: Ensures code works as intended.
* **Facilitates Refactoring**: Encourages clean, maintainable code.
* **Visual**: Bug icon with a slash, shield with a checkmark, broom icon.

**BDD Benefits**

* **Improved Communication**: Clearer requirements through collaboration.
* **Better Alignment with Business Goals**: Focuses on user behavior.
* **Enhanced Test Coverage**: Comprehensive testing through behavior scenarios.
* **Visual**: Dialogue icon, target icon, checklist icon.

**FDD Benefits**

* **Scalable**: Works well for large projects.
* **Efficient Development**: Focuses on delivering features.
* **High Visibility**: Clear tracking of progress.
* **Visual**: Growth chart icon, feature flag icon, visibility icon.

**4. Suitability**

**TDD Suitability**

* **Best For**: Small to medium projects, code-centric development, projects requiring high reliability.
* **Visual**: Small to medium project icons, code symbol.

**BDD Suitability**

* **Best For**: Projects needing strong collaboration, complex requirements, customer-driven development.
* **Visual**: Team icon, complex flow diagram.

**FDD Suitability**

* **Best For**: Large projects, feature-centric development, projects needing clear progress tracking.
* **Visual**: Large project icon, roadmap symbol.