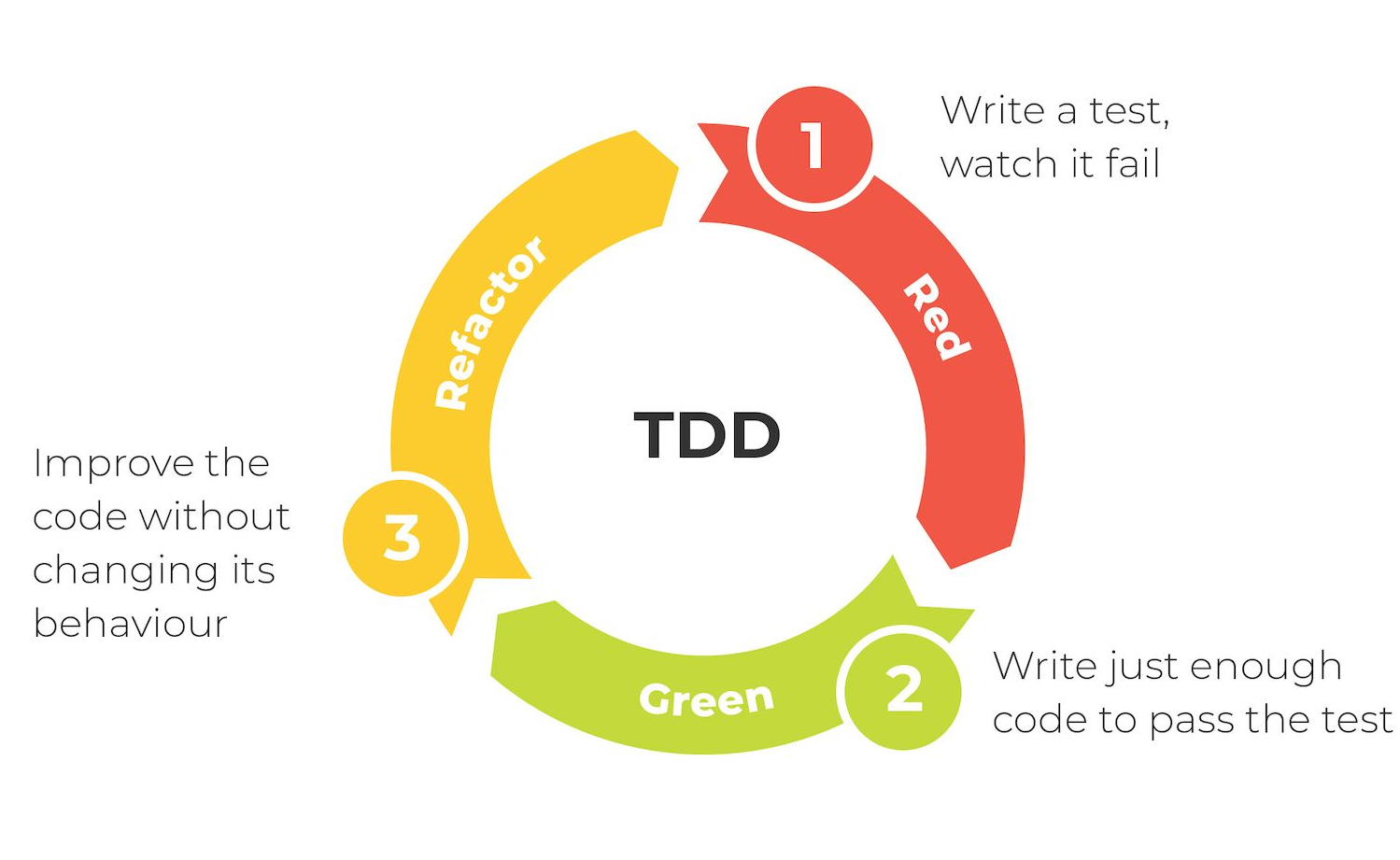
**Noman Khan Day 3 Assignments**

**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**

#### **Section 1: Introduction**

* **Header:** What is TDD?
* **Text:** Test-Driven Development (TDD) is a software development methodology where tests are written before the code. It ensures code reliability and reduces bugs.

#### **Section 2: The TDD Cycle**

#### **1. Write a Test**

* **Icon:** 🧪 (Test tube or checkmark)
* **Text:** Write a test for the next bit of functionality you want to add.
* **Position:** Top of the circle.

#### **2. Run All Tests**

* **Icon:** ▶️ (Running person or play button)
* **Text:** Run all tests. Initially, the new test will fail since the code isn’t written yet.
* **Position:** Top-right segment.

#### **3. Write the Code**

* **Icon:** ✏️ (Code symbol or pencil)
* **Text:** Write the minimum amount of code needed to pass the test.
* **Position:** Bottom-right segment.

#### **4. Run Tests Again**

* **Icon:** 🔄 (Refresh or replay button)
* **Text:** Run all tests again to ensure the new code passes the test.
* **Position:** Bottom segment.

#### **5. Refactor the Code**

* **Icon:** 🔧 (Hammer and wrench)
* **Text:** Refactor the code for optimization and remove redundancies.
* **Position:** Bottom-left segment.

#### **6. Repeat**

* **Icon:** 🔁 (Circular arrow)
* **Text:** Repeat the cycle for the next functionality.
* **Position:** Top-left segment.

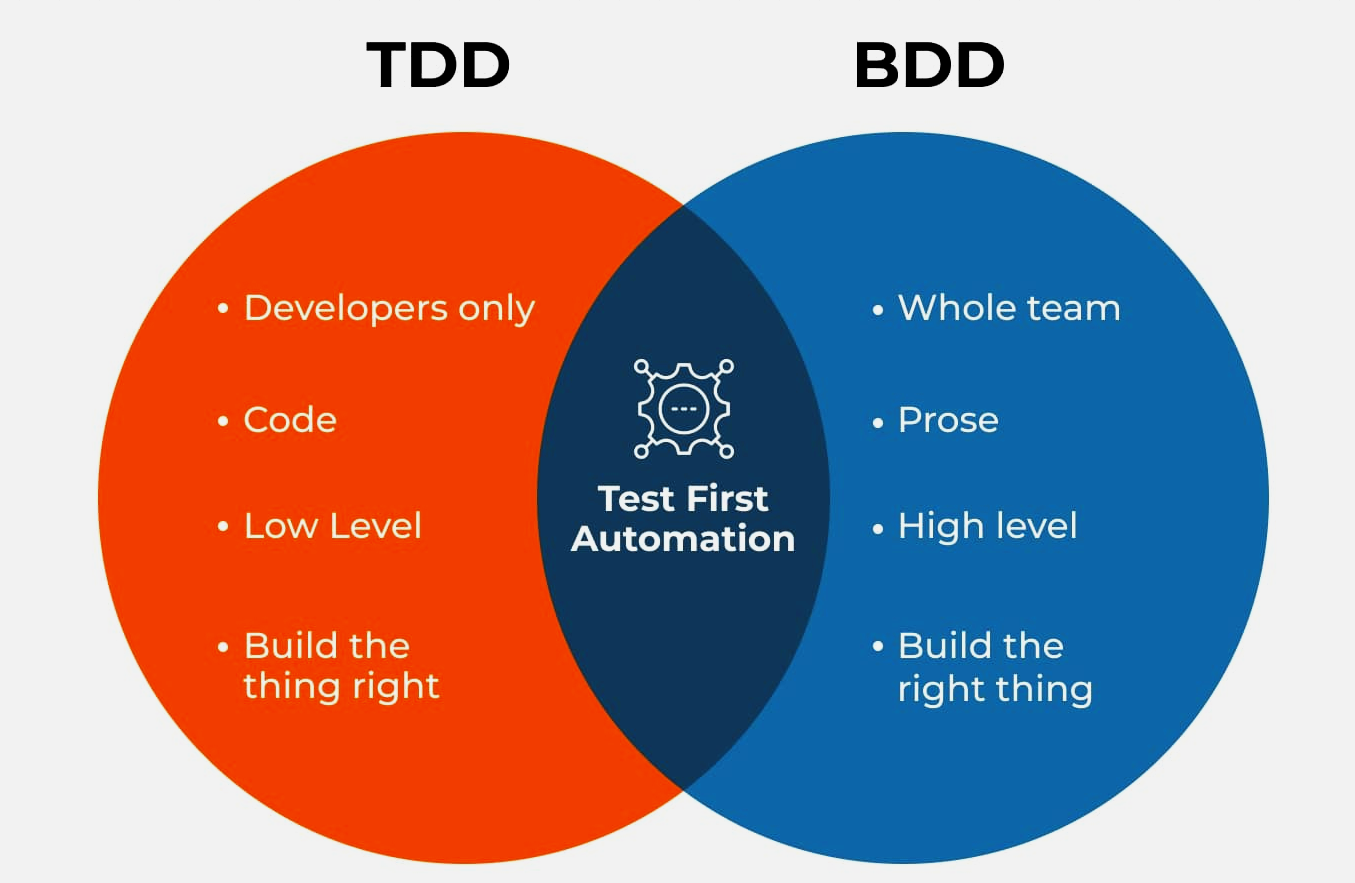
#### 

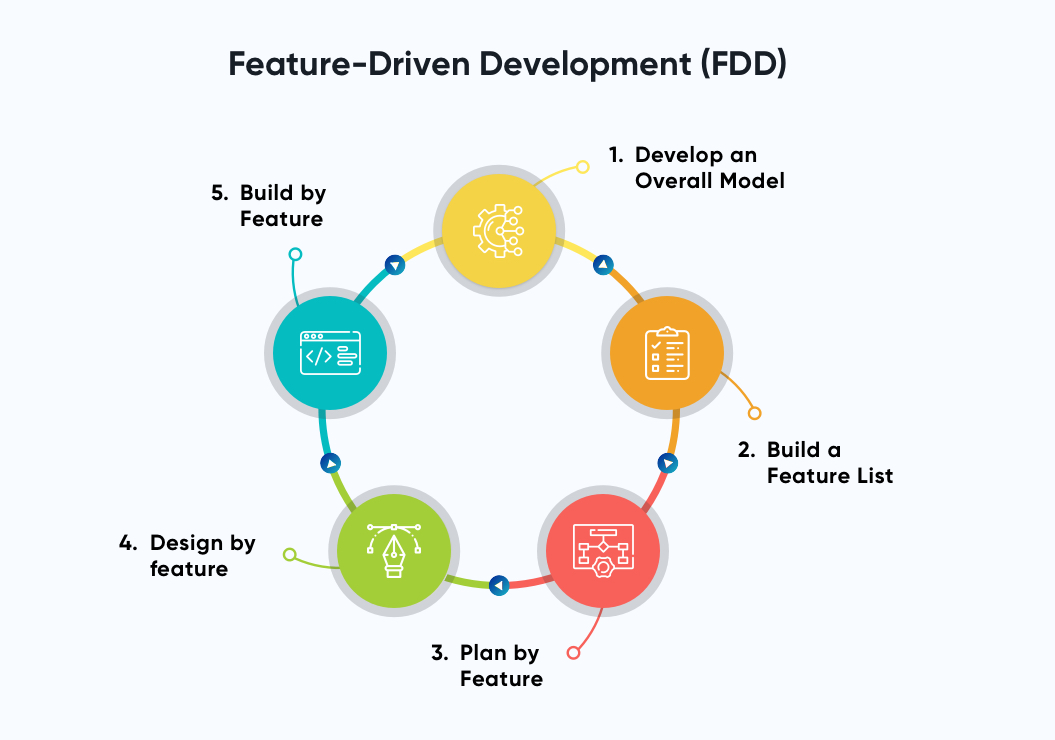
#### 

#### **Section 3: Benefits of TDD**

1. **Bug Reduction**
   * **Icon:** Bug with a cross mark.
   * **Text:** Catch bugs early in the development process.
2. **Reliable Code**
   * **Icon:** Shield or lock.
   * **Text:** Ensures the code works as intended, leading to higher software reliability.
3. **Better Design**
   * **Icon:** Lightbulb or pencil.
   * **Text:** Encourages simpler, more modular, and maintainable code design.
4. **Documentation**
   * **Icon:** Document or book.
   * **Text:** Tests serve as documentation for the codebase.
5. **Confidence to Refactor**
   * **Icon:** Rocket or thumbs up.
   * **Text:** Provides confidence to refactor and improve code without fear of breaking functionality.

**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.





#### **Title: Comparing TDD, BDD, and FDD Methodologies**

### **Section 1: Introduction**

**Header: Understanding Software Development Methodologies**

* **Text:** An overview of Test-Driven Development (TDD), Behavior-Driven Development (BDD), and Feature-Driven Development (FDD). Each methodology has its unique approach, benefits, and suitable contexts.

### **Section 2: Methodology Overview**

1. **Test-Driven Development (TDD)**
   * **Icon:** 🧪
   * **Text:**
     + **Approach:** Write tests before writing the actual code.
     + **Cycle:** Write a test, run all tests, write code, run tests again, refactor, repeat.
2. **Behavior-Driven Development (BDD)**
   * **Icon:** 🤖
   * **Text:**
     + **Approach:** Define the behavior of the software using examples in plain language.
     + **Cycle:** Write scenarios in Given-When-Then format, automate tests, write code, run tests, refactor, repeat.
3. **Feature-Driven Development (FDD)**
   * **Icon:** 🚀
   * **Text:**
     + **Approach:** Develop features based on client-valued functionality.
     + **Cycle:** Identify features, plan by feature, design by feature, build by feature, and inspect code.

### **Section 3: Benefits**

1. **TDD Benefits**
   * **Icon:** ✅
   * **Text:**
     + Reduces bugs early.
     + Ensures code reliability.
     + Provides documentation through tests.
     + Encourages better design and modularity.
2. **BDD Benefits**
   * **Icon:** 📄
   * **Text:**
     + Enhances communication among stakeholders.
     + Provides clear understanding of requirements.
     + Creates living documentation.
     + Facilitates collaboration and reduces misunderstandings.
3. **FDD Benefits**
   * **Icon:** 🌟
   * **Text:**
     + Delivers tangible results quickly.
     + Focuses on client-valued functionality.
     + Ensures frequent and reliable delivery.
     + Supports larger projects with clear feature breakdown.

### **Section 4: Suitability for Different Contexts**

1. **TDD Suitability**
   * **Icon:** 💻
   * **Text:**
     + Ideal for projects requiring high reliability.
     + Suitable for complex codebases.
     + Best for environments emphasizing test automation.
2. **BDD Suitability**
   * **Icon:** 🏢
   * **Text:**
     + Suitable for projects with non-technical stakeholders.
     + Ideal for Agile environments.
     + Great for projects where clear communication and requirements are critical.
3. **FDD Suitability**
   * **Icon:** 🏗️
   * **Text:**
     + Best for large-scale projects.
     + Suitable for teams focusing on incremental and feature-based delivery.
     + Ideal for environments needing frequent and reliable software updates.