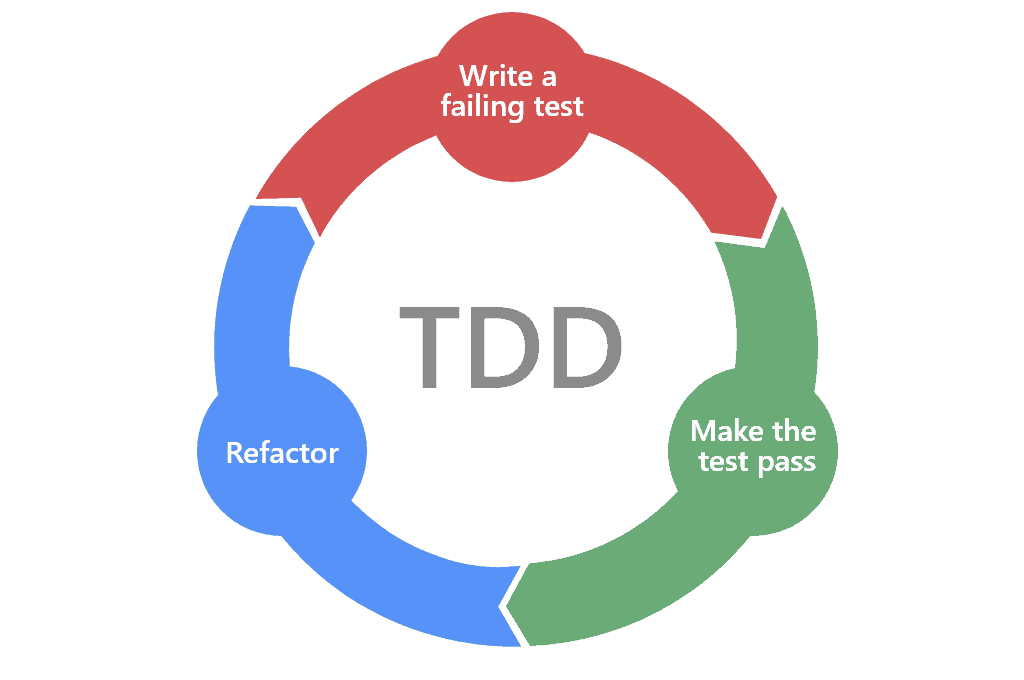
## Days 3

**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.**

****

#### **Step-by-Step Process**

1. Write a test for a new feature or functionality before writing any actual code.
2. Run the test and see it fail. This ensures that the test is valid and the feature does not exist yet.
3. Write the minimum amount of code required to pass the test.
4. Run the test again to see it pass. This confirms that the code meets the requirements specified in the test.
5. Refactor the code to improve its structure and readability without changing its behaviour.
6. Repeat the cycle for each new feature or functionality.

**Benefits of TDD**

1. **Bug Reduction**

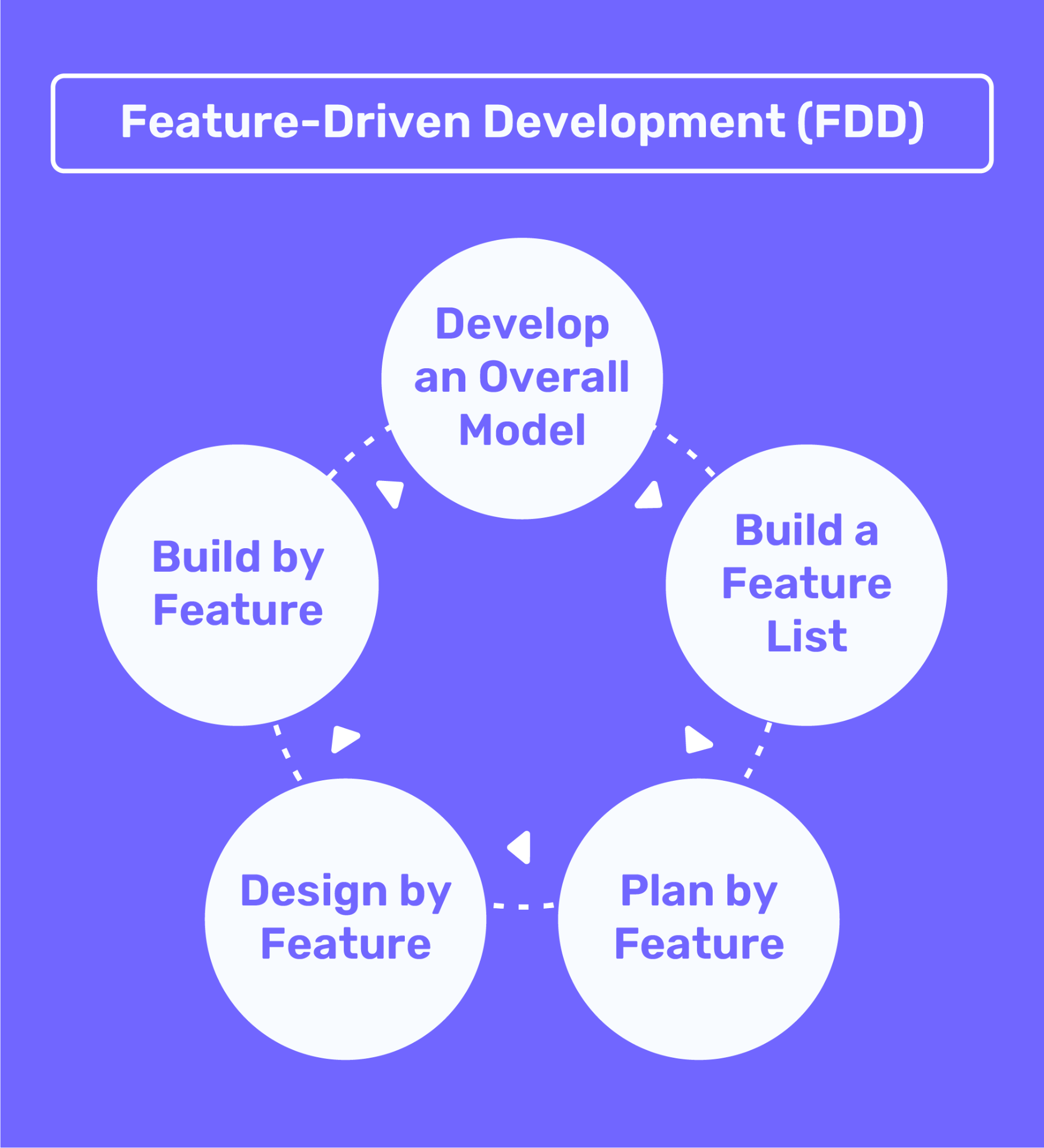
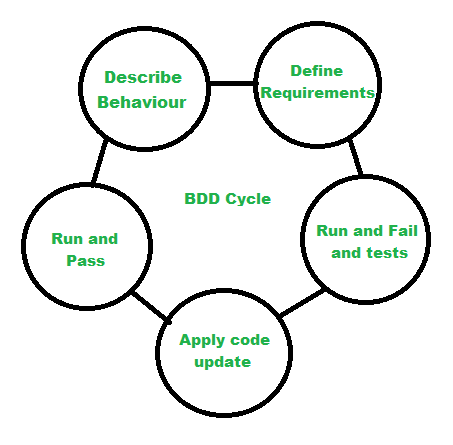
Early detection of bugs through automated tests.

1. **Software Reliability**

Ensures the software behaves as expected, increasing confidence in its reliability.

**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.**

**BDD FDD**

****

|  | **TDD (Test-Driven Development)** | **BDD (Behavior-Driven Development)** | **FDD (Feature-Driven Development)** |
| --- | --- | --- | --- |
| | **Focus** | | --- |  |  | | --- | | | * Testing | | --- |  |  | | --- | | | * Behaviour | | --- | | * Features |
| **Approach** | * Write tests before code | * Write specifications in natural language | * Develop features iteratively |
| | **Process** | | --- |  |  | | --- | | * Write a test * Write code to pass the test * Refactor code | * Write a scenario in natural language * Automate the scenario * Implement functionality to pass the scenario | * Develop an overall model * Build a feature list * Plan by feature * Design by feature * Build by feature |
| **Key Practice** | * Red-Green-Refactor | * Given-When-Then format | | * Iterative feature development | | --- | |
| **Benefits** | | * Ensures code quality. * Reduces bugs and errors. * Facilitates refactoring. | | --- |  |  | | --- | | | * Enhances communication among stakeholders. * Produces clear requirements. * Ensures application meets business needs. | | --- | | | * Manages complexity effectively. * Produces tangible working results. * Suits larger projects with multiple teams. | | --- | |
| **Suitability** | | * Projects requiring high reliability and maintainability. | | --- |  |  | | --- | | | * Projects where clear communication with non-technical stakeholders is crucial. | | --- |  |  | | --- | | | * Large-scale projects with distributed teams. | | --- |  |  | | --- | |

**Assignment 3 : Agile Principle**

* Satisfy the customer through early and continuous delivery of value software
* Welcome changing requirement, even late in development
* Deliver Working Software frequently
* Business People and Developer Must Together/Collaboration
* Build Project Around Motivated individuals
* Promote face-to-face conversation
* Agile process promote sustainable development
* Simplicity
* Working Software
* Regular intervals