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) is a software development practice that focuses on creating unit test cases before developing the actual code. It is an iterative approach combining programming, unit test creation, and refactoring.
- The TDD approach originates from the Agile manifesto principles and Extreme programming.
- As the name suggests, the test process drives software development.
- Moreover, it's a structuring practice that enables developers and testers to obtain optimized code that proves resilient in the long term.

Test Driven Development (TDD) Examples

Note App Function: When building a Note App function, a TDD approach would involve writing a test case for the "add note" function and then writing the code for the process to pass that test. Once the "add note" function is working correctly, additional test cases would be written for other functions such as "delete note", "modify note" .

User Authentication: When building a user authentication system, a TDD approach would involve writing a test case for the user login functionality and then writing the code for the login process to pass that test.

Book Ticket Website: When building an e-commerce website, a TDD approach would involve writing test cases for various features such as Shows Listing, select ticked, and checkout process. Tests would be written to ensure the system works correctly at each process stage, from selecting tickets to the shows to completing the purchase.

Three Phases of Test Driven Development

- Create precise tests: Developers need to create exact unit tests to verify the functionality of specific features. They must ensure that the test compiles so that it can execute. In most cases, the test is bound to fail. This is a meaningful failure as developers create compact tests based on their assumptions of how the feature will behave.

- Correcting the Code: Once a test fails, developers must make the minimal changes required to update the code to run successfully when re-executed.
- Refactor the Code: Once the test runs successfully, check for redundancy or any possible code optimizations to enhance overall performance. Ensure that refactoring does not affect the external behavior of the program.

Benefits of Test Driven Development (TDD)

- Fosters the creation of optimized code.
- It helps developers better analyze and understand client requirements and request clarity when not adequately defined.
- Adding and testing new functionalities become much easier in the latter stages of development.
- Test coverage under TDD is much higher compared to conventional development models. The TDD focuses on creating tests for each functionality right from the beginning.
- It enhances the productivity of the developer and leads to the development of a codebase that is flexible and easy to maintain.

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.

Test Driven Development (TDD) :

- Is a software development practice that focuses on creating unit test cases before developing the actual code. It is an iterative approach combining programming, unit test creation, and refactoring.
- The TDD approach originates from the Agile manifesto principles and Extreme programming.
- As the name suggests, the test process drives software development.
- Moreover, it's a structuring practice that enables developers and testers to obtain optimized code that proves resilient in the long term.

Test Driven Development (TDD) Examples

Note App Function: When building a Note App function, a TDD approach would involve writing a test case for the "add note" function and then writing the code for the process to pass that test. Once the "add note" function is working correctly, additional test cases would be written for other functions such as "delete note", "modify note" .

User Authentication: When building a user authentication system, a TDD approach would involve writing a test case for the user login functionality and then writing the code for the login process to pass that test.

Book Ticket Website: When building an e-commerce website, a TDD approach would involve writing test cases for various features such as Shows Listing, select ticked, and checkout process. Tests would be written to ensure the system works correctly at each process stage, from selecting tickets to the shows to completing the purchase.

Create precise tests: Developers need to create exact unit tests to verify the functionality of specific features. They must ensure that the test compiles so that it can execute. In most cases, the test is bound to fail. This is a meaningful failure as developers create compact tests based on their assumptions of how the feature will behave.

Correcting the Code: Once a test fails, developers must make the minimal changes required to update the code to run successfully when re-executed.

Refactor the Code: Once the test runs successfully, check for redundancy or any possible code optimizations to enhance overall performance. Ensure that refactoring does not affect the external behavior of the program.

Benefits of Test Driven Development (TDD) ? Fosters the creation of optimized code.

- It helps developers better analyze and understand client requirements and request clarity when not adequately defined.
- Adding and testing new functionalities become much easier in the latter stages of development.
- Test coverage under TDD is much higher compared to conventional development models. The TDD focuses on creating tests for each functionality right from the beginning.
- It enhances the productivity of the developer and leads to the development of a codebase that is flexible and easy to maintain.

Behavioral-Driven Development (BDD) :
Behavioral-Driven Development (BDD) is a testing approach derived from the Test-Driven Development (TDD) methodology. In BDD, tests are mainly based on systems behavior. This approach defines various ways to develop a feature based

Given the user has entered valid login credentials

When a user clicks on the login button

Then display the successful validation message

Key benefits of BDD

- Helps reach a wider audience through the usage of non-technical language
- Focuses on how the system should behave from the customer's and the developer's perspective
- BDD is a cost-effective technique
- Reduces efforts needed to verify any post-deployment defects
- Acceptance Test-Driven development

In the Acceptance Test-Driven Development (ATDD) technique, a single acceptance test is written from the user's perspective, mainly focusing on satisfying the system's functional behavior. This technique attempts to answer the question – Is the code working as expected?

This technique enhances collaboration among developers, users, and QAs with a shared focus on defining the acceptance criteria.

The following are some of the key practices in ATDD:

- Analyzing and discussing the real-world scenarios
- Deciding the acceptance criteria for those test scenarios
- Automating the acceptance of test cases
- Focusing on the development of those requirement cases

Benefits of ATDD

- Requirements are very clearly analyzed without any ambiguity
- Encourages collaboration among cross-team members
- The acceptance test serves as a guide for the software development process