Maddula Rupa Sri Manohar
maddularupasrimanohar2001@gmail.com

Assignment-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.**

Answer:

Test Driven Development is the process in which test cases are written before the code that validatesthose cases. It depends on repetition of a very short development cycle. Test driven Development is a technique in which automated Unit test are used to drive the design and free decoupling of dependencies.

The following sequence of steps is generally followed:

Test-Driven Development (TDD) Process Infographic

1. **Write Test:**

- Developers write automated tests for a small piece of functionality before writing the corresponding production code.
- Tests are written to define the desired behavior and functionality of the code.

2. **Run Test:**

- Automated test suite is executed to validate the code.
- Initial test will fail as no code has been written yet.

3. **Write Code:**

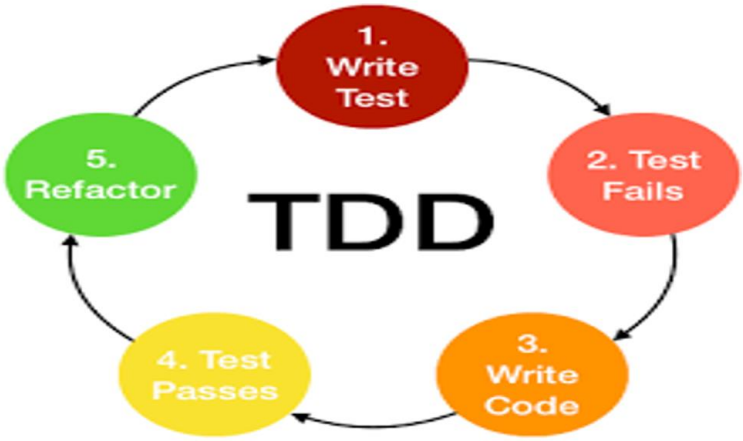
- Developers write the minimum amount of code necessary to make the failing test pass.
- Focus is on writing only what's needed to fulfill the test requirements.

4. **Run Test Again**:

- Automated test suite is executed again to verify that the newly written code passes the test.
- Test should now pass, indicating successful implementation of the functionality.

5. Refactor Code:

- Developers refactor the code to improve readability, maintainability, and performance.
- Refactoring is done without changing the behavior of the code as verified by the tests.



Benefits of TDD:

- Bug Reduction: By writing tests first, developers catch bugs early in the development process, reducing the likelihood of defects in the final product.
- Improved Code Quality: TDD encourages developers to write clean, modular, and well- structured code that is easier to maintain and extend.
- Increased Reliability: With a comprehensive suite of automated tests, developers can confidently make changes to the codebase without fear of introducing regressions.

How TDD Fosters Software Reliability:

- Continuous Testing: TDD promotes a culture of continuous testing, where every code change is validated against a suite of automated tests.
- Regression Prevention: Automated tests act as a safety net, catching

regressions and ensuring that existing functionality remains intact.

- Early Feedback: TDD provides immediate feedback on the correctness of code, allowing developers to quickly identify and fix issues.

Challenges and Best Practices :

- Initial learning curve and mindset shift
- Writing good tests (FIRST principles)
- Test code organization and maintenance
- Balancing TDD with other development approaches

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.

Answer:

1. Test-Driven Development (TDD):

Test Driven Development is the process in which test cases are written before the code that validatesthose cases. It depends on repetition of a very short development cycle. Test driven Development is a technique in which automated Unit test are used to drive the design and free decoupling of dependencies.

The following sequence of steps is generally followed:

1. Add a test – Write a test case that describe the function completely. In order to make the test casesthe developer must understand the features and requirements using user stories and use cases.

2. Run all the test cases and make sure that the

new test case fails.3.Write the code that passes

the test case

4. Run the test cases

5.Refactor code – This is done to remove duplication of code.

6.Repeat the above mentioned steps again and again

Approach:

- Developers write tests before writing production code.
- Focuses on writing small, incremental tests to drive the development process.

Benefits:

- Unit test provides constant feedback about the functions.
- Quality of design increases which further helps in proper maintenance.
- Test driven development act as a safety net against the bugs.
- TDD ensures that your application actually meets requirements defined for it.
- TDD have very short development lifecycle.

Suitability:

- Well-suited for Agile development environments.
- Ideal for projects with clearly defined requirements and a focus on code quality andreliability.

2. Behavior-Driven Development (BDD):

Behavioral Driven Development is a good approach in Automated Testing as it is more focused on thebehavior of the system rather than the implementation of the code. BDD is facilitated through natural language to express the behavior of the system and the expected outcomes from the system. In BDDall parties are involved like a customer, developer, tester, and stakeholder for a collaborated conversation and illustration of the system's behavior.

1. Describe behavior –This includes the flow and features of the product means the main vision.
2. Define requirements –Modeled requirements with business rules for a shared understanding.
3. Run and fail the tests –Develop and run the test cases.
4. Apply code update –Refactor it according to the requirement.

Run and pass the tests –Run the updated code and pass the test cases.

Approach:

- Focuses on behavior and outcomes rather than implementation details.
- Uses natural language specifications (e.g., Given-When-Then) to define tests.

Benefits:

- Greater clarity on business goals and customer requirements.
- Reaches a larger customer set as it uses non-technical languages.
- Helps in defining acceptance criteria before development.
- Focuses on the system's behavior from the client's and developer's point of view.
- Helps in avoiding unnecessary features and includes important features.
- Reduces effort for post-modification and post-deployment defects.

- Avoids misinterpretations during the development process.
- Collaboration between team members is promoted through BDD, involving all stakeholdersin the development process to ensure a shared understanding of objectives.

Suitability:

- Suitable for projects with complex business logic and a need for collaboration betweentechnical and non-technical stakeholders.
- Ideal for Agile teams focused on delivering value to end-users.

3. Feature-Driven Development (FDD):

FDD stands for Feature-Driven Development. It is an agile iterative and incremental model that focuses on progressing the features of the developing software. The main motive of feature-driven development is to provide timely updated and working software to the client. In FDD, reporting andprogress tracking is necessary at all levels.

FDD Lifecycle :

- Build overall model
- Build feature list
- Plan by feature
- Design by feature
- Build by feature

Approach:

- Focuses on building features incrementally based on client priorities.
- Emphasizes short iterations and frequent client feedback.

Benefits:

- Reporting at all levels leads to easier progress tracking.
- FDD provides continuous success for larger size of teams and projects.
- Reduction in risks is observed as whole model and design is build in smaller segments.

- FDD provides greater accuracy in cost estimation of the project due to feature segmentation.

Suitability:

- Well-suited for large-scale projects with multiple development teams and diverse requirements.
- Ideal for projects where features can be prioritized and developed independently.

Conclusion: Each methodology offers a unique approach to software development, catering to different project requirements and team dynamics. Whether it's the test-driven approach of TDD, the collaborative nature of BDD, or the feature-centric approach of FDD, choosing the right methodology depends on factors such as project size, complexity, and stakeholder involvement.