GANDAKI COLLEGE OF ENGINEERING AND SCIENCE

Lamachaur,Pokhara

LAB REPORT OF

Agile Software Development

LAB – 2

SUBMITTED BY:	**SUBMITTED TO:**
Baradan Parajuli	Er. Rajendra Bdr. Thapa
Roll No: 16	
6th Semester	
BE Software	

LAB 2: Test Driven Development and Behavioral Driven Development

Objective

To explore and compare Test Driven Development (TDD) and Behavioral Driven Development (BDD) methodologies through practical implementation and analysis of their approaches, tools, and effectiveness in software development.

Theory

Test-Driven Development (TDD) is a methodology where developers write tests before writing the actual code. The approach follows a three-step cycle:

1. **Red** – Write a test that fails.
2. **Green** – Write just enough code to make the test pass.
3. **Refactor** – Improve the code structure while ensuring the test still passes.
4.

Behavior-Driven Development (BDD) builds on TDD by focusing more on the expected **behavior** of the system from the end user's perspective. It encourages writing test cases in simple, natural language that both technical and non-technical team members can understand. BDD follows this structure:

- **Given** a specific context or condition
- **When** an action is taken
- **Then** an expected result should occur

BDD helps in improving communication, aligning software features with business goals, and ensuring clarity in requirements.

Methodologies

1. We began by analyzing a simple problem (e.g., a calculator).
2. In the **TDD approach**, we first wrote tests for functions like addition and subtraction before writing the actual code.
3. In the **BDD approach**, we described how the calculator should behave in plain language and then implemented the functionality based on those descriptions.
4. Both approaches were repeated for different features of the application.

Observations

Test-Driven Development (TDD)

Strengths:

- Helped catch errors early in the development process.
- Improved the structure and modularity of the code.
- Tests acted as a form of documentation.
- Increased confidence while making code changes.

Challenges:

- Writing tests before code required a mindset shift.
- Maintaining tests as code changed added extra effort.
- Difficult to mock complex dependencies at times.

Behavior-Driven Development (BDD)

Strengths:

- Improved collaboration between technical and non-technical team members.
- Scenarios written in plain language made requirements clearer.
- Encouraged a stronger focus on user behavior and business needs.
- Test cases served as living documentation.

Challenges:

- More time-consuming to write and maintain behavior scenarios.
- Initial setup and understanding of the BDD format took some effort.
- Translating real-world behavior into testable steps wasn’t always straightforward.

Conclusion

Through this lab, we practiced writing tests before code (TDD) and describing functionality from the user's point of view (BDD). These techniques help developers build better software by reducing bugs, improving design, and ensuring the final product meets user expectations.