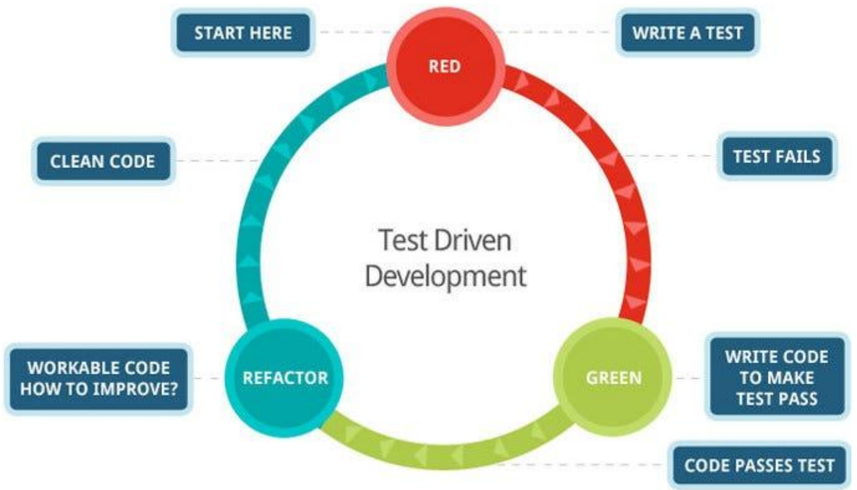
Assignment 2: Difference Between TDD and BDD Methodologies

(With visuals, examples, benefits, and suitability)

Introduction

TDD (Test-Driven Development) and BDD (Behaviour-Driven Development) are two popular development methodologies that improve software quality.
Both rely on testing before coding, but their **focus, workflow, and communication style** are different.

What is TDD (Test-Driven Development)?



Definition

TDD is a development practice where **tests are written before the actual code**.
The goal is to ensure the code works exactly as expected and remains maintainable.

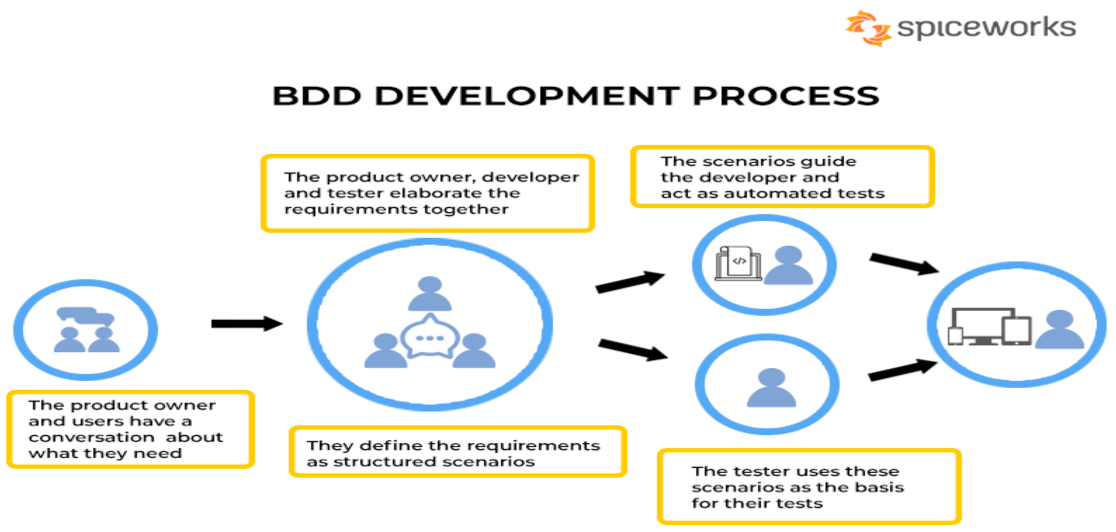
TDD Workflow (Red → Green → Refactor)

1. **Red** – Write a failing test case.
2. **Green** – Write the simplest code to make the test pass.
3. **Refactor** – Improve the code while keeping tests green.

When to Use TDD

✓ Small modules
✓ Logic-heavy systems
✓ Projects demanding high reliability
✓ When clean, bug-free code is the priority

What is BDD (Behaviour-Driven Development)?


spiceworks
BDD DEVELOPMENT PROCESS
The product owner and users have a conversation about what they need
The product owner, developer and tester elaborate the requirements together
They define the requirements as structured scenarios
The scenarios guide the developer and act as automated tests
The tester uses these scenarios as the basis for their tests

Definition

BDD focuses on **system behaviour** from the user's perspective.
Tests are written in **plain English**, making them understandable to developers, testers, and business stakeholders.

Key Characteristic

BDD uses the **Gherkin** syntax:

Given (precondition)

When (action)

Then (expected outcome)

Example Using Gherkin

Feature: Login

Scenario: Successful login

Given user is on login page

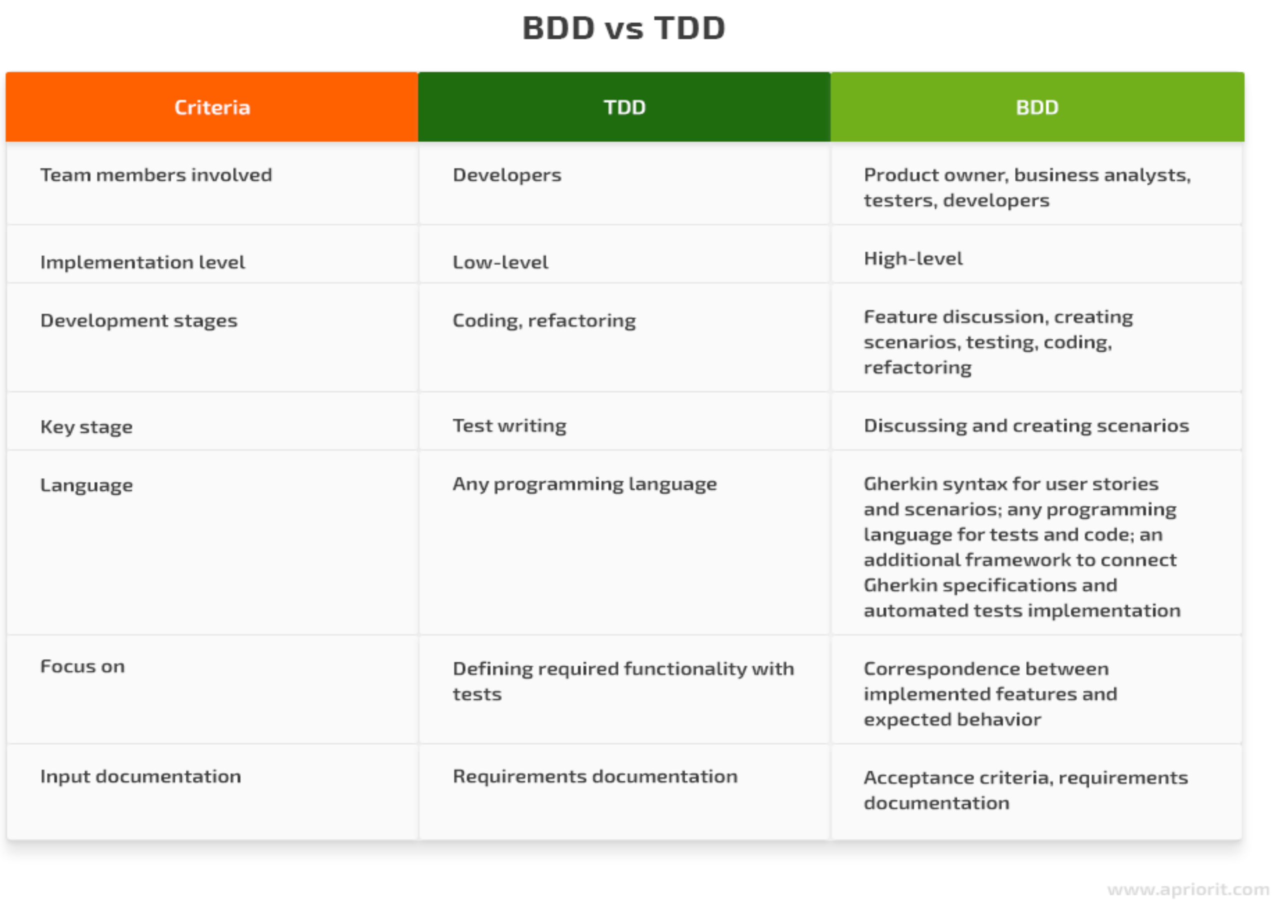
When user enters valid username and password

Then user should be redirected to dashboard

When to Use BDD

✓ Projects with constant client interaction
✓ When clarity of requirements is critical
✓ Large teams with developers + testers + product owners
✓ Systems where user behaviour is the focus

Key Differences Between TDD and BDD


BDD vs TDD

Criteria	TDD	BDD
Team members involved	Developers	Product owner, business analysts, testers, developers
Implementation level	Low-level	High-level
Development stages	Coding, refactoring	Feature discussion, creating scenarios, testing, coding, refactoring
Key stage	Test writing	Discussing and creating scenarios
Language	Any programming language	Gherkin syntax for user stories and scenarios; any programming language for tests and code; an additional framework to connect Gherkin specifications and automated tests implementation
Focus on	Defining required functionality with tests	Correspondence between implemented features and expected behavior
Input documentation	Requirements documentation	Acceptance criteria, requirements documentation

Feature	**TDD**	**BDD**
Focus	Code correctness	User behaviour
Language Used	Programming language	Business-friendly English
Tests Written By	Developers	Developers + QA + Business Analysts
Test Type	Unit tests	Acceptance & behavioural tests
Workflow	Red → Green → Refactor	Define behaviour → Write Scenarios → Automate
Goal	Bug-free, clean code	Correct user experience
Best For	Logic-heavy code	User-focused applications

Benefits of Each Method

✓ TDD Benefits

- Reduces bugs early
- Improves code quality
- Encourages modular & clean architecture
- Easier refactoring
- Enhances developer confidence

✓ BDD Benefits

- Improves communication between teams
- Requirements are clearer and testable
- Reflects real user behaviour
- Leads to better user experience
- Helps avoid misunderstandings

Which One Should You Use?

Use TDD When…

You want high-quality, error-free internal logic.

Use BDD When…

You want clarity, stakeholder involvement, and behaviour-focused testing.
