# 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 1: Write a Test

Write a test for a specific piece of functionality

Test should be independent of implementation details

Step 2: Run the Test and Fail

Run the test and see it fail (since no code has been written yet)

Step 3: Write the Code

Write the minimal amount of code necessary to pass the test

Focus on making the test pass, not on writing perfect code

Step 4: Run the Test and Pass

Run the test and see it pass

Step 5: Refactor the Code

Refactor the code to make it maintainable, efficient, and easy to understand

Keep the test passing during refactoring

Benefits of TDD:

- Reduced Bugs: Writing tests before code ensures that the code is testable and reduces the likelihood of bugs.

- Faster Development: Writing tests first helps developers focus on the required functionality, reducing development time.

- Improved Quality: TDD ensures that the code is reliable, stable, and meets the required standards.

- Confidence in Code: TDD gives developers confidence in their code, reducing fear of changing or refactoring it.

Best Practices

- Keep Tests Simple: Keep tests simple, focused, and independent.

- Write Tests Before Code: Write tests before writing code.

- Refactor Mercilessly: Refactor code regularly to keep it maintainable.

# 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.

TDD (Test-Driven Development)

Approach: Write tests before writing code

- Benefits:

- Fewer bugs

- Faster development

- Better quality code

- Confidence in code

- Suitable for:

- Small to medium-sized projects

- Projects with well-defined requirements

- Teams with experienced developers

BDD (Behavior-Driven Development)

Approach: Write tests in natural language style

- Benefits:

- Improved communication between developers and non-technical stakeholders

- Faster development

- Higher quality code

- Better alignment with business needs

- Suitable for:

- Projects with complex business logic

- Projects with multiple stakeholders

- Teams with varying levels of experience

FDD (Feature-Driven Development)

Approach: Deliver functional features to the end-user

- Benefits:

- Improved collaboration between developers and stakeholders

- Faster time-to-market

- Higher quality code

- Better alignment with business needs

- Suitable for:

Large and complex projects

- Projects with rapidly changing requirements

- Teams with diverse skill sets

Comparison

- TDD focuses on unit testing

- BDD focuses on acceptance testing

- FDD focuses on delivering functional features

- TDD and BDD are iterative processes

- FDD is an iterative and incremental process

# Assignment 3

Agile Project Planning - Create a one-page project plan for a new software feature using Agile planning techniques. Include backlog items with estimated story points and a prioritized list of user stories

Project Name: New Payment Gateway

Project Goal: Implement a new payment gateway that allows users to securely make payments online.

Backlog Items:

- User Story 1: As a user, I want to be able to enter my payment information securely. (Estimated Story Points: 4)

- User Story 2: As a user, I want to be able to select from various payment methods (e.g. credit card, PayPal). (Estimated Story Points: 3)

- User Story 3: As a user, I want to be able to view a confirmation of my payment. (Estimated Story Points: 2)

- User Story 4: As a user, I want to be able to receive a receipt via email. (Estimated Story Points: 1)

- User Story 5: As a user, I want to be able to store my payment information for future use. (Estimated Story Points: 5)

Prioritized List of User Stories:

1. User Story 1: Secure payment information

2. User Story 5: Store payment information

3. User Story 2: Select payment method

4. User Story 3: Payment confirmation

5. User Story 4: Email receipt

Sprint Goals:

Sprint 1: Complete User Story 1 and User Story 5

- Sprint 2: Complete User Story 2 and User Story 3

- Sprint 3: Complete User Story 4 and conduct thorough testing

Estimated Timeline: 3 Sprints, each lasting 2 weeks

Team Members:

Maria (Developer)

- John (QA Engineer)

- Emily (Product Owner)

# Assignment 4

Daily Standup Simulation - Write a script for a Daily Standup meeting for a development team working on the software feature from Assignment 1. Address a common challenge and incorporate a solution into the communication flow.

Scrum Master: Good morning, team. Let's start our Daily Standup. Remember to share what you worked on yesterday, what you're working on today, and any obstacles you're facing.

Developer 3: Yesterday, I worked on implementing the payment form UI. Today, I'm working on integrating it with the payment gateway API.

Scrum Master: Great, Developer 3. Any obstacles?

Developer 3: Actually, I'm struggling with some compatibility issues between the API and our application. I'm not sure how to resolve them.

Scrum Master: Okay, let's flag that as a blocker. Has anyone else worked on API integrations before?

Developer 4: Yes, I have. I can help Developer 3 out.

Scrum Master: Excellent. Developer 3 and Developer 4, can you work together to resolve the compatibility issue?

Developer 3: Thanks, Developer 4.

Scrum Master: Moving on... QA Engineer, what's your update?

QA Engineer: Yesterday, I tested the payment processing workflow. Today, I'm testing the payment form UI. So far, everything looks good.

Scrum Master: Great to hear! Product Owner, any updates from your side?

Product Owner: Yeah. I've been reviewing the feature requirements, and I think we need to add some additional security measures to ensure secure payment processing.

Scrum Master: Alright. Let's discuss that further after the standup. Any other updates or obstacles?

Team: No updates or obstacles.

Scrum Master: Great! Let's summarize: Developer 3 and Developer 4 will work on the compatibility issue, QA Engineer will continue testing, and Product Owner will discuss the additional security measures with the team. Let's all keep each other updated and have a great day!