Task 2 – Reflection

Group L

Topic 1: Requirements Engineering

Lecture Reference: Slides 2-6, L03_Requirements_Engineering

How it Helped in Software Development

Requirements engineering (RE) is essential in defining what the software needs to do. The lecture provided a framework for creating clear and detailed Software Requirements Specifications (SRS).

Challenges and Impact:

Ambiguous Requirements: Initially, we faced ambiguity in stakeholder requirements. The lecture emphasized focusing on "what the system is supposed to do" rather than "how it should do it" (Slide 6). This helped us refine our SRS, making the requirements clear and actionable.

Evolving Requirements: Managing changing requirements was challenging. The lecture's focus on handling evolving needs (Slide 9) guided us in implementing iterative review sessions, keeping our development aligned with stakeholder expectations.

Incorporating these principles reduced misunderstandings and rework, ensuring we built the right system.

Topic 2: Implementation and Testing

Lecture Reference: Slides 4-8, L07_Implementation + Testing pt. 1

How it Helped in Software Development

The lectures on implementation and testing were crucial for ensuring software quality and functionality.

Challenges and Impact:

Ensuring Code Quality: Ensuring high-quality code was challenging. The lecture discussed unit, integration, and system testing (Slides 4-6), which guided us in implementing comprehensive test cases. This early defect identification significantly improved our software's quality.

Handling Complex Test Scenarios: Managing complex test scenarios and edge cases was another challenge. The importance of writing detailed test cases (Slides 7-8) led us to develop thorough test plans, covering typical and edge cases, ensuring robustness and reliability.

These testing strategies saved time and resources, ensuring our final product met user expectations.

Missing Aspects in the Lectures

1. User Interface (UI) Design:

For beginners, understanding the basics of UI design is crucial. Good UI design enhances user experience and usability. Including a lecture on basic UI design principles, such as layout, colour theory, and user interaction, would help students create more user-friendly applications**.**

2. Software Security Basics:

Security is a critical aspect of software development. A lecture on basic software security principles, including common vulnerabilities, secure coding methods, and basic encryption approaches, would prepare students to design more secure programs.