SE3070 – Case Studies in Software Engineering

Assignment 02 – Implementation of a software solution based on a case study design

Marks out of 100.

Assignment weightage – 30%

Released on – 24/09/2025

Submission Deadline – 11.59 PM, 17/10/2025

This assignment involves both group and individual components. Refer to the case study assigned to you and,

- As a group:
    - Create **one report** critiquing and suggesting justified improvements to the received case study design to make it better aligned with the assigned case study (covering the 4 substantial business use cases in the original design is sufficient).
        - The critique should assess how faithfully the original design fulfills the assigned case study requirements, its logical soundness, and the correctness of UML usage, noting both strengths and weaknesses. For interaction design, the evaluation should cover usability, logical flow, and adherence to HCI principles.
        - Suggested improvements must clearly align with the critique and show **well-justified** adjustments to the use case diagram, class diagram, sequence diagrams, use case scenarios and the user interfaces.
- Individually each group member will:
    - Implement one substantial use case covering all the relevant use case scenarios under each use case.
        - Ensure that the suggested changes in the report are followed exactly in the implementation.

- Ensure that high quality maintainable code is written and that the best practices are followed.
- Ensure that there is excellent and meaningful unit test coverage for the code that they write.

Instructions

- You are expected to preserve the original use cases and design as much as possible, but you are free to introduce **well-justified** changes to broaden the scope of the design.
- Replacing a use case in the original design is only acceptable if the original use case is too simple and cannot be reasonably modified to broaden the scope. Any replacement must be a domain-specific business use case aligned with the given case study.
- Do not change the original design for the sake of changing, without reasonable justification. This will not be acceptable.
- The report must be designed collaboratively, with all members contributing to completeness and accuracy.
- Each member of the group should implement one use case.
- Avoid implementing common functions such as login, logout, or administrative privilege granting. These will not be considered for grading.
- For each implementation, producing high-quality code that meets the selected use case requirements is expected over writing low-quality code that attempts to cover a broader scope.
- Each unit within the selected use case must be meaningfully tested. A good coverage expectation is 80% of functionality.
- User Interfaces should remain consistent with the storyboard/ wireframes in the group report.
- Detailed instructions on each component are found in the marking rubric below.

Marking Rubric

Criteria	Excellent [25-22]	Good [21-18]	Satisfactory [17-14]	Needs Improvement [13-10]	Poor [09-00]
[Group] Critique the design and suggest justified improvements (30 marks)					
Critique the original design Weight: Functional Design 90%, Interaction Design 10% (20 marks)	Comprehensive, precise critique of functional and interaction designs. Demonstrates excellent evaluation of requirement coverage, logical soundness, UML correctness, and usability. Interaction critique is meaningful and justified.	Solid critique with mostly correct evaluation of requirements, logic, UML usage, and usability. Interaction critique is present but may lack depth.	Identifies some issues but misses key aspects. Partial evaluation of requirements, logic, UML usage, or usability. Interaction critique is basic.	Minimal critique with weak reasoning. Major gaps in evaluating requirements, logic, UML correctness, or usability.	Little or no meaningful critique. Fails to evaluate requirements, logic, UML usage, or interaction design.
Proposed changes to the original design (10 marks)	Clear, accurate, and well-drawn diagrams illustrating suggested improvements. All changes are directly tied to critique and justified with strong reasoning.	Mostly correct diagrams showing improvements, with reasonable justification. Some links to critique may be indirect.	Basic illustrations of changes with limited detail. Some changes are weakly justified or loosely tied to the critique.	Minimal or unclear illustrations. Weak alignment with critique and poor justification.	No meaningful illustrations or irrelevant diagrams.
[Individual] Implementation of the selected use case (50 marks)					
Implementation Accuracy & Scope	Fully and correctly implemented end-to-end functionality. All	Mostly correct implementation with minor omissions or	Basic functionality works but contains gaps, errors, or	Partially implemented functionality with significant	Functionality is incomplete,

compared to the design (30 marks)	aspects of the design are covered. Alignment with use cases, use case scenarios and sequence diagrams is clear. UX is excellent. Enhancements are well integrated.	inaccuracies. Alignment with design is generally consistent. UX is good. Enhancements are reasonable.	inconsistencies. UX is reasonable. Enhancements are limited or unclear.	inconsistencies. UX is average. Enhancements are poorly integrated or irrelevant.	incorrect, or irrelevant.
Code Quality & Best Practices (20 marks)	Code is clean, well-structured, and documented. Strong adherence to coding conventions, SOLID principles, and avoidance of code smells. Appropriate design patterns are applied correctly.	Code is mostly clean and structured with meaningful comments. Minor deviations from conventions or SOLID. Design patterns are used appropriately in most cases.	Code is functional but lacks structure, readability, or comments. Partial adherence to conventions or SOLID. Design pattern use is limited or weak. Some recurring code smells.	Code is disorganized, poorly documented, and shows little adherence to conventions or SOLID. Frequent code smells. Design patterns misapplied or missing.	Code is of very poor quality, undocumented, disorganized, and fails to follow best practices.
[Individual] Unit Testing (20 marks)					
Unit Testing Quality (20 marks)	Comprehensive, meaningful tests with >80% coverage. Covers positive, negative, edge, and error cases. Meaningful assertions in unit tests. Tests are well-structured and readable.	Solid tests with 60 - 80% coverage. Covers positive, negative and some edge/error cases. Meaningful assertions in unit tests. Tests are generally well-structured.	Basic tests with 40 - 60% coverage. Covers only positive and/ or negative cases. Mostly meaningful assertions are used. Coverage of edge cases is minimal.	Minimal tests with 20 - 40% coverage. Poorly structured, few assertions, weak understanding of testing principles.	No meaningful tests or coverage < 20%. Tests are irrelevant or unusable.