Assessment 1 – Part 1: Requirements Analysis

Due date: Sunday of Week 7
Weight: 15%
Project: Requirements analysis and software design
Collaboration: Groups of 3-4. Group submission, group assessed with peer ratings.

1- Project Brief

A local university wants to develop a new interactive system to allow students to self-enrol in subjects. Students need to register before they can access the system. A student can enrol in a maximum of four (4) subjects. Enrolment is limited to one semester; therefore, multiple semester enrolment is out of the scope.

Students must register by entering their name, an email, and a password. Student emails and passwords should be validated against pre-defined patterns. For example, emails should end with the domain "@university.com", hence firstname.lastname@university.com is a valid email, while firstname.lastname@university is not. A password should meet the following criteria: (i) It starts with an upper-case letter, (ii) It contains at least five (5) letters, (iii) It is followed by three (3) or more digits.

Upon registration, a unique ID ranging from 1 to 999,999 will be automatically generated for each student. If the generated ID has fewer than six digits, prefix it with zeros to ensure it becomes a six-digit number (e.g., 002340 is valid, whereas 2345 is not).

Registered students can then log into the application and perform the following operations: (i) Enrol in a subject, (ii) Remove a subject from their enrolment list, (iii) View the current enrolment list, (iv) Change password. Students do not need to select a specific subject when enrolling, a new subject will be added automatically to their enrolment list upon selecting the enrol command. The system tracks the number of subjects in each student's enrollment list and notifies them if they attempt to enroll in more than four (4) subjects. A subject is identified by a unique 3-digit auto-generated ID (1 <= ID <= 999). Upon enrolment, a random subject mark between 25 and 100 will be autogenerated and assigned. The subject grade will then be automatically calculated based on the mark. According to the UTS grading system: mark < 50 → Z; 50 <= mark < 65 → P; 65 <= mark < 75 → C; 75 <= mark < 85 → D; mark >= 85 → HD.

Data of registered students including their enrolment details should be saved to a file called `students.data`. This file should also be available for Admins to perform student management operations.

Admins, who are existing university staff and do not require registration, have a dedicated subsystem within the application for student management. Admins can view all registered students, organize and view students by grade, and categorize students into PASS/FAIL categories based on their marks (mark < 50 → FAIL; mark >= 50 → PASS). Additionally, admins can remove individual students or clear the entire `students.data` file.

The university requires a CLI application - the CLIUniApp - for the above student and admin operations. CLIUniApp should provide two interactive subsystems, one for students and one for admins. The application will store student data in `students.data`. All CRUD operations in CLIUniApp should use this file.

The university also requires a GUI component - GUIUniApp - at a smaller scale. GUIUniApp is a challenge task of Assessment 1 - Part 2 and is designed only for registered students. The "login window" is the main window of the GUIUniApp. A registered student should be able to log into GUIUniApp and access the "enrolment window" to enrol in a maximum of four (4) subjects. Each time a student is enrolled in a subject, the subject is added to their enrolment list. GUIUniApp should handle possible exceptions for empty login fields and for attempts to enrol in more than four (4) subjects.

You team is expected to develop the application in two phases: Part 1 and Part 2, then demonstrate the result to the stakeholders in Part 3.

In Part 1, you will complete and deliver a comprehensive software requirements analysis report including: (i) Transforming the requirements into user stories and mapping the user stories to a requirements table (backlog); (ii) Creating a UML use case diagram to illustrate the actors, use cases and their relationships; (iii) Creating a UML class diagram and explaining in detail the classes, their properties and their relationships.

In Part 2, you will develop the university application with respect to the specifications and the Part 1 design. The university application is composed of CLIUniApp and GUIUniApp (challenge task). The code and supplementary materials should be submitted by the due date.

Part 3 is the formal showcase. Each team will present their Part 2 working application. All team members should participate equally and collaborate in all assessment parts.

2- User-Story Backlog

You should read the customer (university) requirements thoroughly and transform the requirements into user stories. The user story should be simple and generic so that each story is later translated into a function (or action). Each story will have a unique 3-digit ID. Stories related to the same feature will share the same hundreds digit. For example:

The following stories are all related to the same Login feature. Hence their IDs should start with the same hundreds digit.
Story: match username and password with the ones on file → 101
Story: verify username and password against patterns → 105
Story: show an error message if credentials do not match → 106
Story: take a student to the student subsystem if credentials are correct → 100

The refined user stories should be mapped into a requirements table (backlog) formatted as follows:

ID	User	Action	Result	Function

A unique 3-digit user story ID.	The person or entity taking the action	The action taken by the user	The result or outcome of the action	The action name

3- UML Use Case Diagram

Your team will develop a comprehensive UML use case diagram by identifying the actors, goals, use cases, and their relationships. Provide explanation for each actor, goal, use case and relationship and ensure your diagram is consistent and aligned with the explanations.

4- UML Class Diagram

Your team will develop a comprehensive UML class diagram by identifying the classes, fields, methods, visibility, multiplicity, and their relationships. Provide explanation for each of these and ensure your diagram is consistent and aligned with the explanations.

5- Marking Scheme

Total Assessment 1 - Part 1 mark is 15/70. All team members should contribute equally.

a. User story backlog (5 Marks)

Entity	Criteria	Mark
User stories are specific	User stories are decomposed into simple stories	2
User stories are consistent	User stories align with the project requirements	2
Backlog correctness	User stories are correctly mapped into the backlog	1

b. Use case diagram (5 Marks)

Entity	Criteria	Mark
Entities identification	Goals, use cases, actors, relationships correctly identified	1
Entities description	Entities are correctly explained	1
Actors action	Actors initiate accurate use cases	1
Relationships	Accurate and consistent relationship between actors and use cases and between use cases	1
Labelling	Use of correct relationship labelling	1

c. Class diagram (5 Marks)

Entity	Criteria	Mark
Class	Properly identified and explained	1
Fields	Properly identified with accurate choice of visibility	1
Methods	Correctly name the methods and identify theie types and visibility	1
Relationships	Consistent class relationships	1
Multiplicity	Accurate relationship multiplicities	1

6- Deliverables and Contribution

This assessment requires collaboration and equal contribution from all group members. The individual contributions or parts will be collated into a group submission for assessment. For Assessment 1 - Part 1, each submission will receive a group mark, and you will be asked to rate the contributions of your group members in Sparkplus. The group mark multiplied by the Sparkplus score will be the individual mark.

7- Assessment Submission

Submit Assessment 1 - Part 1 as a **single submission per group** in either .docx or .pdf format.

File naming convention: **group<group-number>-Cmp1<lab-number>.<format>**

Upload your assessment file to Canvas under Task 1: Case Study/Part 1: Analysis and Design by the due date and time specified on Canvas.

8- Special Consideration

Special consideration for deadline extensions must be arranged beforehand with the subject co-ordinator. Please refer to the UTS Special Consideration Procedure (https://www.uts.edu.au/current-students/managing-your-course/classes-and-assessment/special-circumstances/special-consideration).

9- Late Penalty

Please refer to the UTS Late Submission Penalties page (https://www.uts.edu.au/current-students/current-students-information-faculty-law/assessment/late-submission-penalties).

10- Assessment Misconduct

Please see the subject outline for plagiarism and academic integrity in conjunction with UTS policy and procedures for the assessment for coursework subjects.