

Software Analysis to Design

Team Actimel

Go team Actimel.

Department of Engineering

School of Informatics & Engineering

Institute of Technology, Blanchardstown

Dublin 15.

Software Design and Quality

Submission Date

# Table of Contents

[Table of Contents i](#_Toc468972995)

[Table of Figures ii](#_Toc468972996)

[Introduction 1](#_Toc468972997)

[Problem Statement 2](#_Toc468972998)

[Software Analysis 3](#_Toc468972999)

[Use Case Analysis 3](#_Toc468973000)

[Fully Dressed Use Case Model 1 3](#_Toc468973001)

[Fully Dressed Use Case Model 2 4](#_Toc468973002)

[Conceptual Model 6](#_Toc468973003)

[UML Class Conceptual Model 6](#_Toc468973004)

[Modelling Assumptions 6](#_Toc468973005)

[Behaviour Analysis 7](#_Toc468973006)

[High-level sequence diagram 7](#_Toc468973007)

[High-level Activity Diagram 8](#_Toc468973008)

[High-level State Diagram 9](#_Toc468973009)

[Software Design 10](#_Toc468973010)

[Interaction Analysis 10](#_Toc468973011)

[Detailed Sequence Diagrams 10](#_Toc468973012)

[Refined Class Diagram 11](#_Toc468973013)

[Implementation Recommendations & Notes 11](#_Toc468973014)

# Table of Figures

[Figure 1 - Fully Dressed Use Case Model 1 3](#_Toc468972916)

[Figure 2 - Fully Dressed Use Case Model 2 4](#_Toc468972917)

[Figure 3 - UML Class Conceptual Model 6](#_Toc468972918)

[Figure 4 - High-level sequence diagram 7](#_Toc468972919)

[Figure 5 - High-level Activity Diagram 8](#_Toc468972920)

[Figure 6 - High-level State Diagram 9](#_Toc468972921)

[Figure 7 - Detailed Student Sequence Diagram 10](#_Toc468972922)

[Figure 8 - Detailed Admin Sequence Diagram 10](#_Toc468972923)

[Figure 9 - Refined Class Diagram 11](#_Toc468972924)

# Introduction

In this assignment, teams of three are tasked with completing a project in tandem. The team must show an ability to work together, setting out a clear plan of action, using knowledge gained in labs and lectures in relation to Analysis and the production of diagrams to support and aid in the planning and implementation process. The team must show an ability to follow the Analysis documents produced in order to complete a successful final product.

The use of ‘Github’ in this assignment is crucial in order to collaborate collectively and productively. Teams are to show a number of Analysis diagrams and Design documentation, which will be used in the production of the code (using tinyXML) to produce a ‘multiple choice quiz’.

The product is a ‘Multiple Choice quiz’ with ten random questions taken from the database of Twenty Question’s. The answer is selected, from a list of four possible answers. After the quiz is completed, the results are generated, and returned to the user, the user has the option to view the report and or restart the quiz, with ten attempts available to each user.

The administrator has clearance/permission to change and add to the txt document, which is referred to as the question bank, this holds the list of questions and answers. They must also have access to all of the reports.

In order to produce this quiz, implementation of the information compiled in the Analysis and design documentation should be followed, as is good practice. In other words, adherence to the Class Diagram is advisable, coding documentation using ‘Doxygen’ will also be provided.

When the team has completed the coding portion of the project, progression to ‘Whitebox testing’ is the next phase in completing the product. In order to do this there is White box testing Design, Branch/Path coverage Analysis for major “Units” and “Google-Test” test harness designed and defined.

Documentations in relation to SRS to complete the Project, will also be produced as part of the final documentation, this is to ensure if there is any future problems the team can go back over the SRS documentation to see if the problem was in the planning, design or implementation process and use this information for future projects.

# Problem Statement

“Multiple Choice Quiz application will be produced using C++ (tinyXML) to aid students taking the module “Software Design & Quality” understand agile software development methodologies (specifically Kanban & SCRUM) and hence study for their January exams.

Users will be presented with a randomly chosen set of 10 questions from a question bank and be presented with a choice of solutions. The user must then choose a correct answer and move to the next question. The application will generate a report of how well the student has done, after the quiz has been completed. The student’s result is saved to their profile. The application can manage many student profiles and can generate an overall class report.

An admin user may administrate the quiz. An admin user has full access to all student’s attempts, but a student user can only see their own attempts as such Student’s must login to their profile before starting the quiz. Students are allowed to have multiple attempts, which are all persistently saved against their individual profile. Questions and solutions are saved in an XML file, which is read by the quiz application. You will need to research Agile developmental lifecycles to define your questions and you will need to investigate how to parse XML using C++.”

# 

# Software Analysis

## Use Case Analysis

### Fully Dressed Use Case Model 1

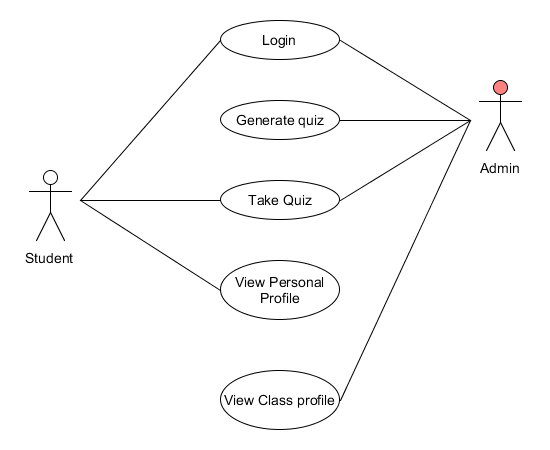


Figure 1 - Fully Dressed Use Case Model 1

#### Fully Dressed Description

Primary Actor: Student

Goal in context: Goal

Level: User Level

Stake Holders and Interests:

Student: wants to login and take a quiz

Student: wants to view their profile

Preconditions: System ready & waiting for the user to login

Minimum guarantee: User is able to login to the system

Success guarantees: User logins to the system, takes quiz, views result

Trigger: User enters credentials

Main success scenario:

1. User enters credentials
2. Presses ‘take quiz’ button
3. Generates quiz
4. Review Quiz

Extensions:

Frequency of use: Daily

Priority: 1

Implementation status: …

Open issues: …

### Fully Dressed Use Case Model 2

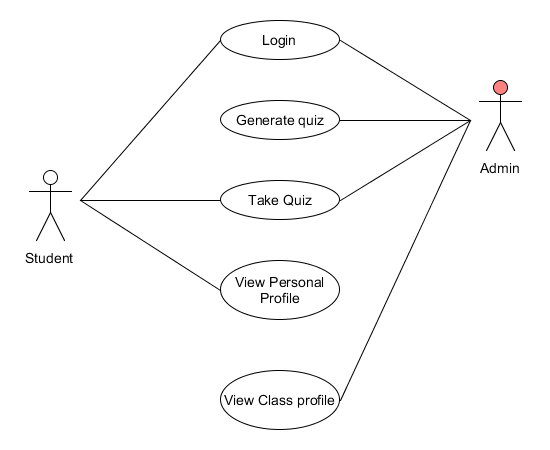


Figure 2 - Fully Dressed Use Case Model 2

#### 

#### Fully Dressed Description

Primary Actor:

Goal in context:

Level:

Stake Holders and Interests:

Preconditions:

Success Guarantees:

Trigger:

Main success scenario:

Extensions:

Frequency of use:

Priority:

Implementation status:

Open issues:

## Conceptual Model

### UML Class Conceptual Model

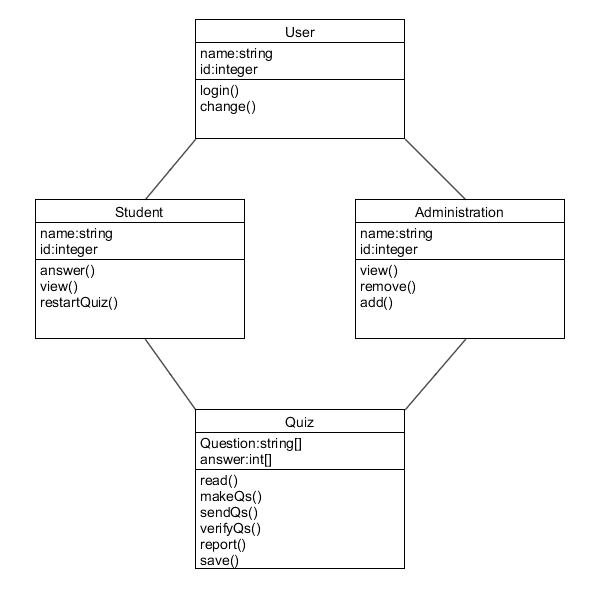


Figure 3 - UML Class Conceptual Model

### Modelling Assumptions

The normal assumptions where that there was to be two types of users in the modelling structure. Firstly, the student, who can login, start the quiz and or view the profile with results and the Administrator who has the permission and clearance to add or change the data within the text file namely the question bank. The administrator can view all class profiles and report’s, they can also take the quiz.

Considerations taken into account for the fact that the ‘Quiz’, will only be generated ten times per student.

## Behaviour Analysis

### High-level sequence diagram

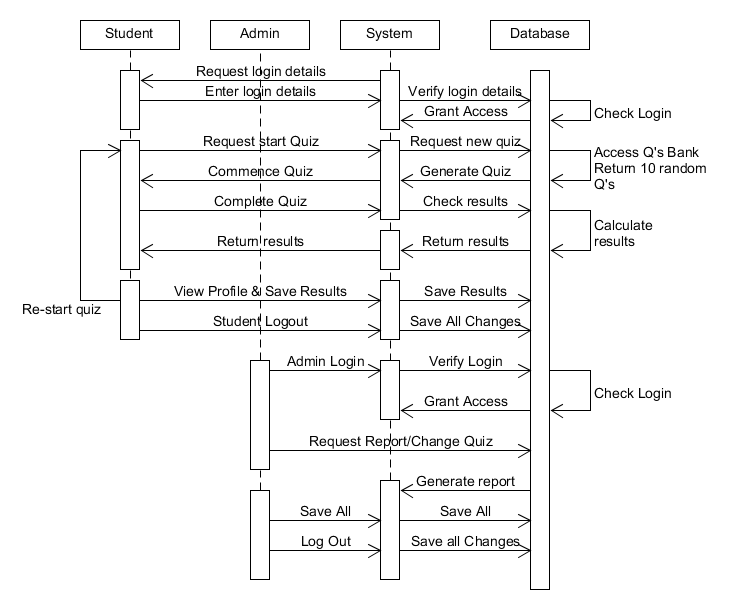


Figure 4 - High-level sequence diagram

### 

### High-level Activity Diagram

### 

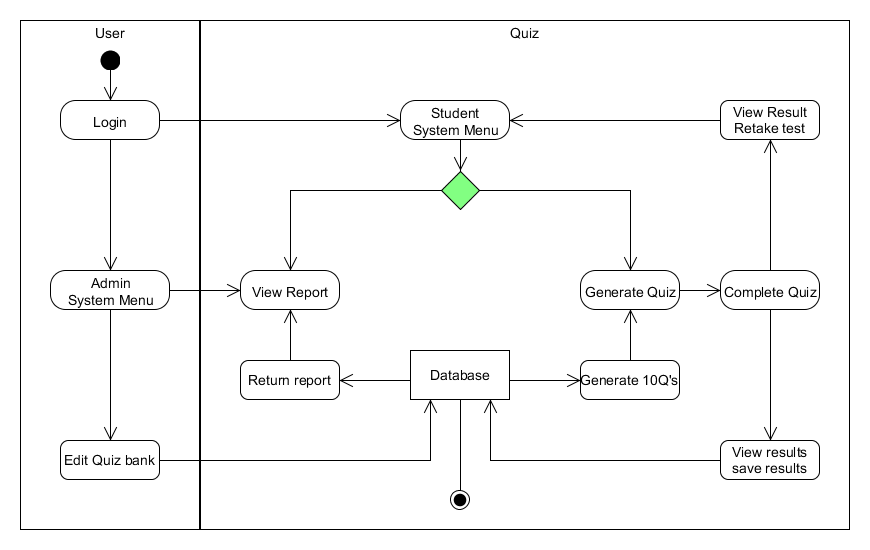


Figure 5 - High-level Activity Diagram

### High-level State Diagram

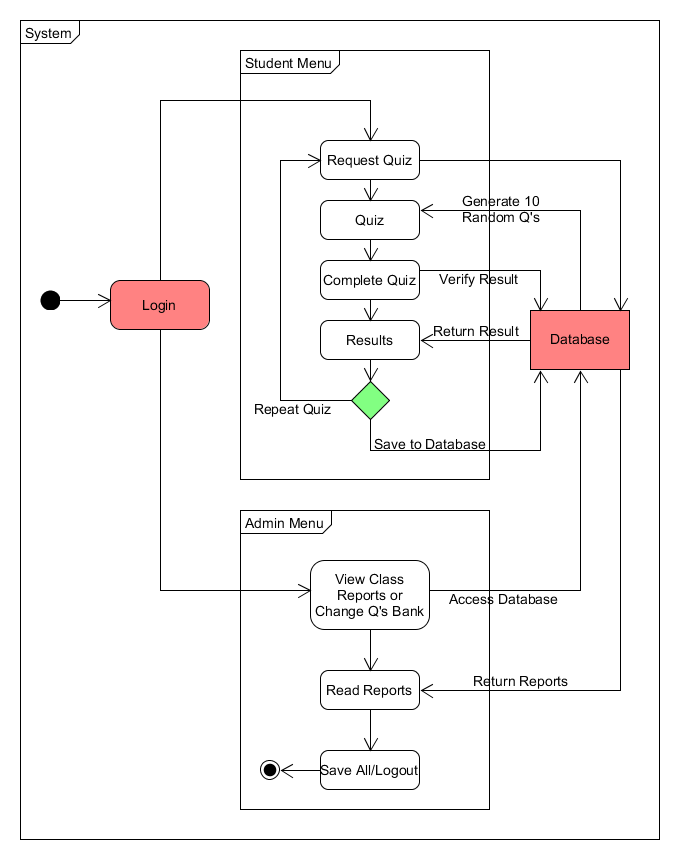


Figure 6 - High-level State Diagram

# Software Design

## Interaction Analysis

### Detailed Sequence Diagrams

#### Sequence 1

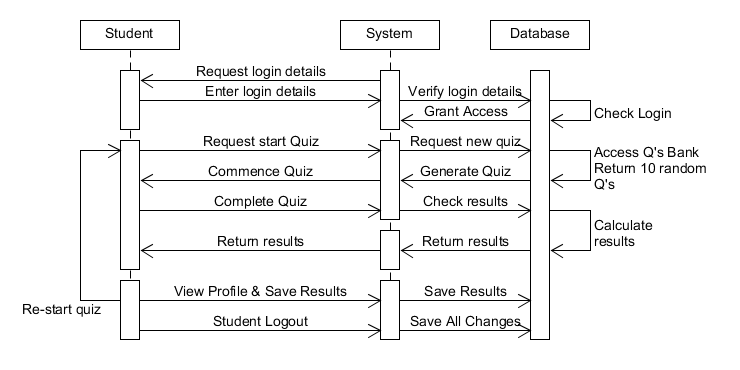


Figure 7 - Detailed Student Sequence Diagram

Sequence 2

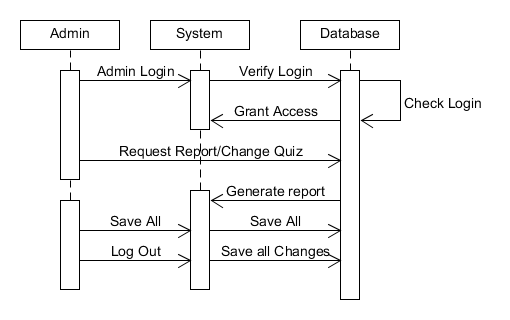


Figure 8 - Detailed Admin Sequence Diagram

### Refined Class Diagram

Figure 9 - Refined Class Diagram

### Implementation Recommendations & Notes