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
16th December 2016

Table of Contents

Table of Figures

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

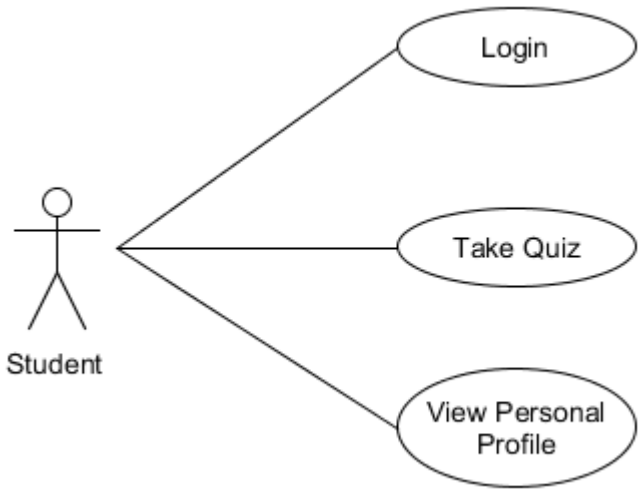

Figure 1 - Fully Dressed Use Case Model 1

Fully Dressed Description

Primary Actor: Student

Goal in context: Student takes quiz

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 to takes quiz and then views result

Trigger: User enters credentials to login

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

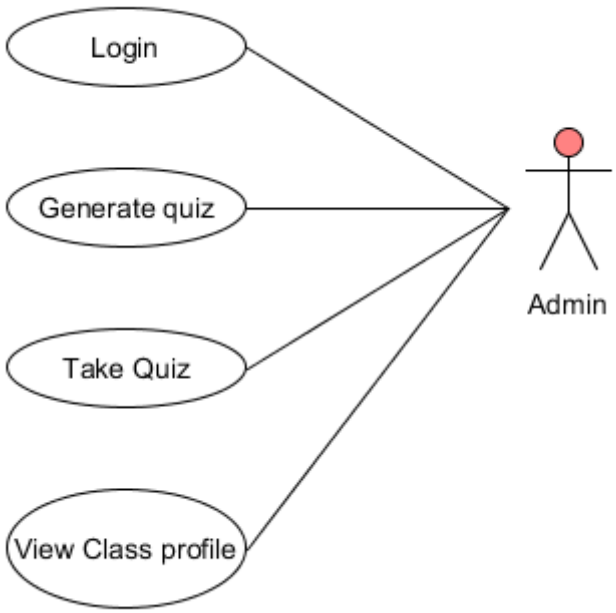

Figure 2 - Fully Dressed Use Case Model 2

Fully Dressed Description

Primary Actor: Admin

Goal in context: Produce quiz

Level: Admin Level

Stake Holders and Interests:

Admin wants to generate quiz

Admin wants to take quiz

Admin wants to view class profile

Preconditions: System ready & waiting for the user to login

Minimum guarantee: Admin is able to login to the system

Success Guarantees: Admin logins to the system to generate a quiz, takes the quiz and then views the result

Trigger: Admin enters credentials to login

Main success scenario:

1. Admin Logs in
2. View class reports
3. Edit Xml Document
4. Take quiz
5. Log out

Extensions:

Frequency of use: Daily

Priority: 1

Implementation status: …

Open issues: …

Conceptual Model

UML Class Conceptual Model

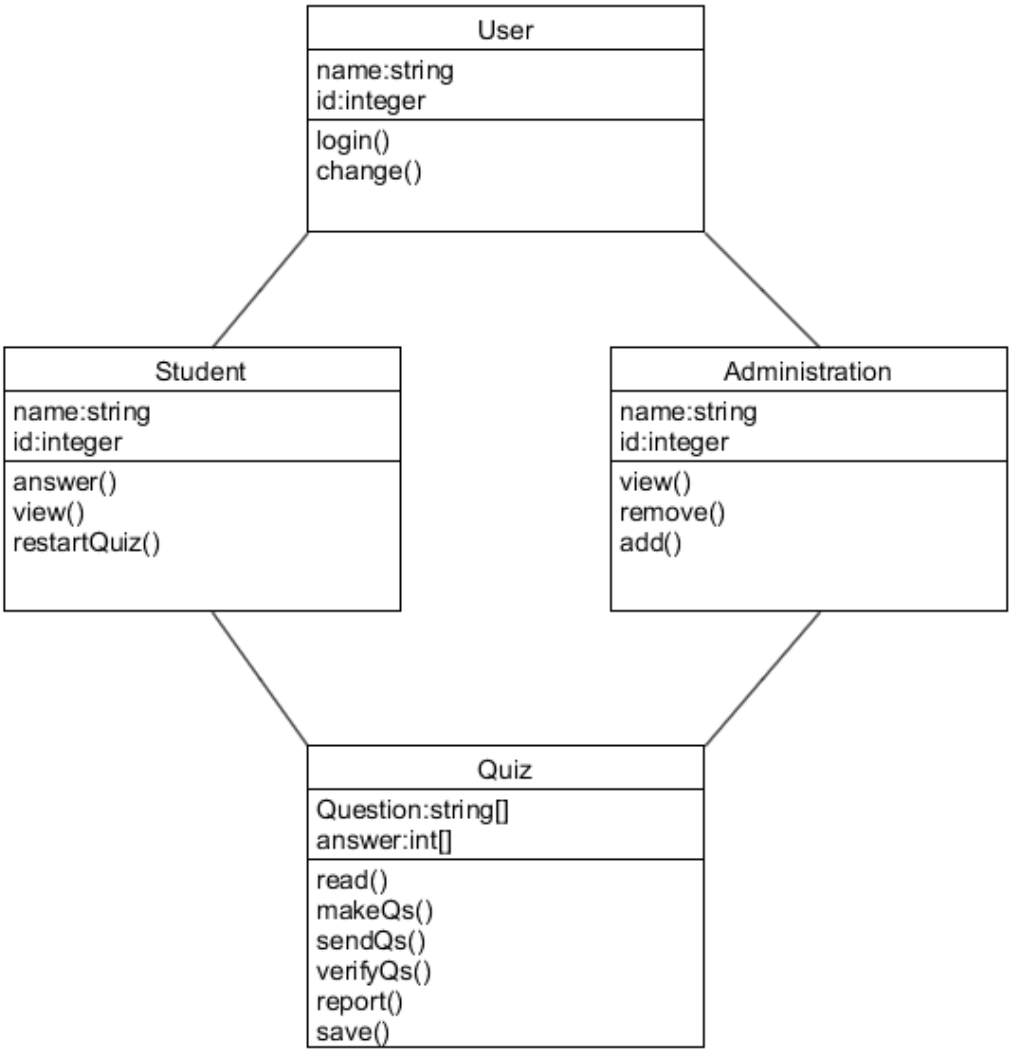

Figure 3 - UML Class Conceptual Model

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

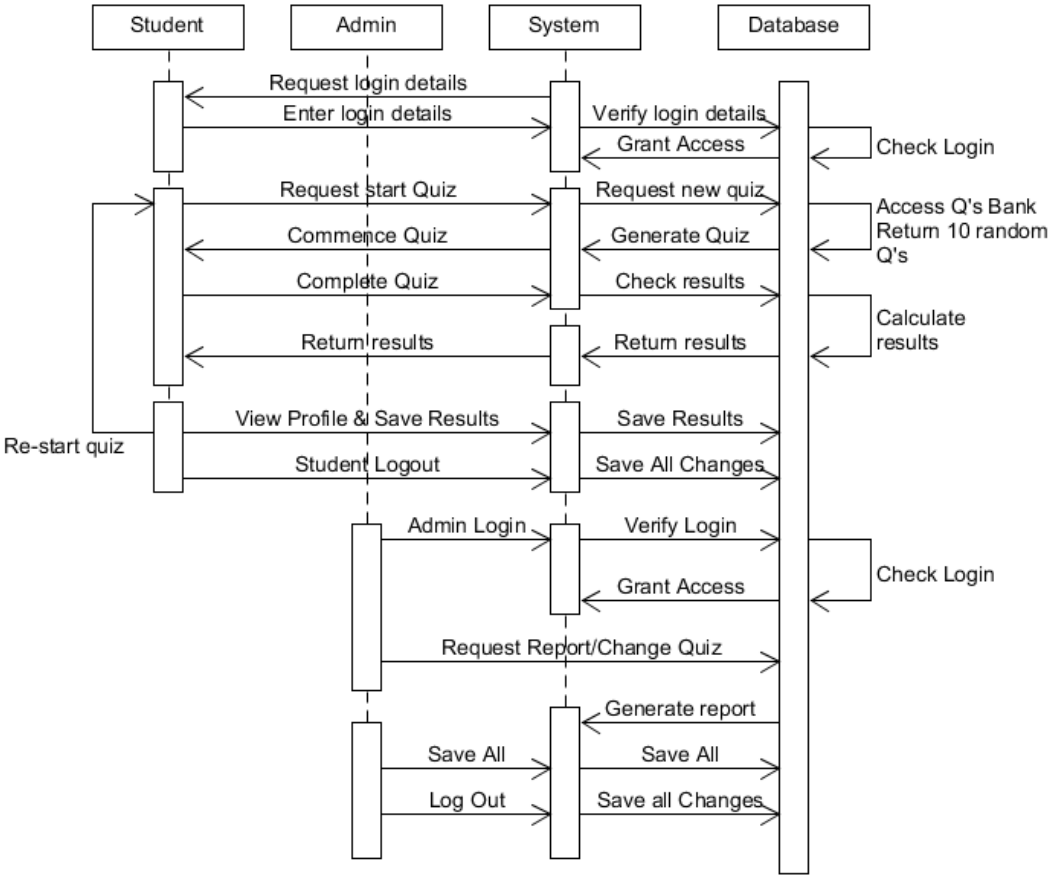

Figure 4 - High-level sequence diagram

High-level Activity Diagram

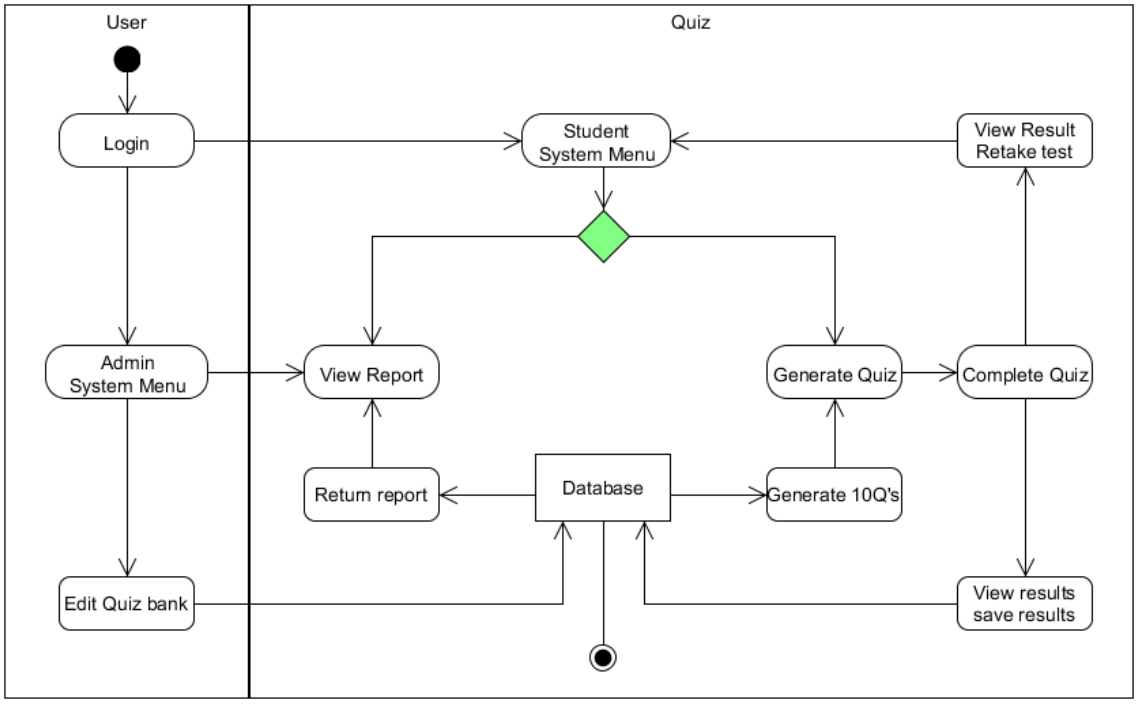

Figure 5 - High-level Activity Diagram

High-level State Diagram

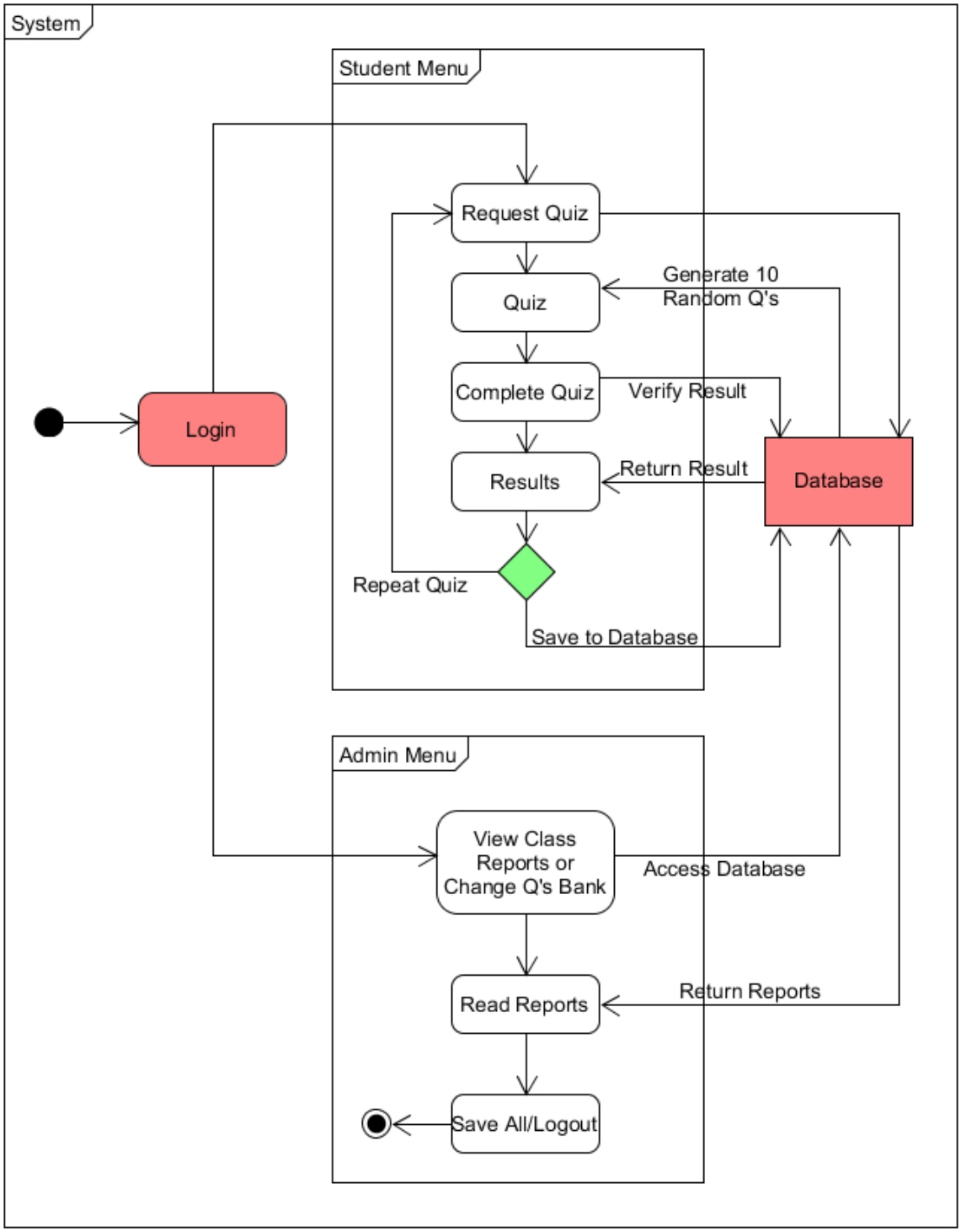

Figure 6 - High-level State Diagram

System Design Observations and Recommendations

During the process of planning and developing the Analysis and Design documentations which mainly consists of Class, Sequence, Case, State and activity diagrams at different levels of detail. The overall visualisation of the product to be completed was easier to imagine and therefore implement.

The more the team went into detailing each phase the more achievable the 'End product' seemed to be. Therefore, when it came to the coding phase, this seemed to be the easiest of all the tasks involved.

The only observation the team would have is the difficulty in producing a productive Refined Class diagram, which was difficult without contemplating the use of the new code that the team had recently been introduced to, but after studying 'tinyXML' further, the Team was able to put together a relevant Refined Class diagram.

The team would recommend a program language be chosen before planning begins and a good understanding of that code in order to productively move through each phase at a timely manner and keep within the outlined time constraints.

Also, keep a details plan and record of all work and saved diagrams even if they are not included in the SRS document or reports as these may be useful. Never discard any work completed, this may be a missing piece for future projects.

Software Design

Interaction Analysis

Detailed Sequence Diagrams

Sequence 1

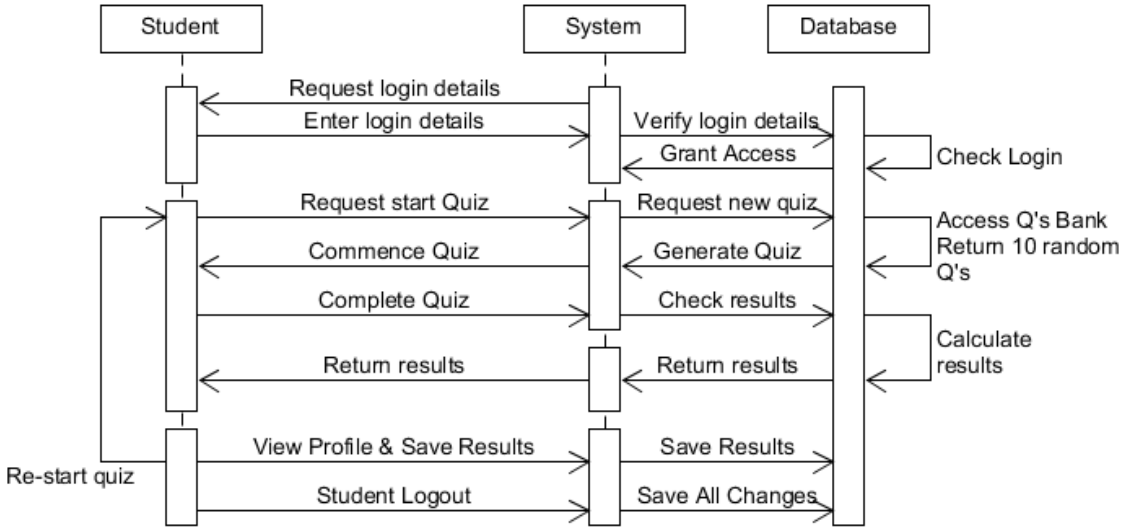

Figure 7 - Detailed Student Sequence Diagram

Sequence 2

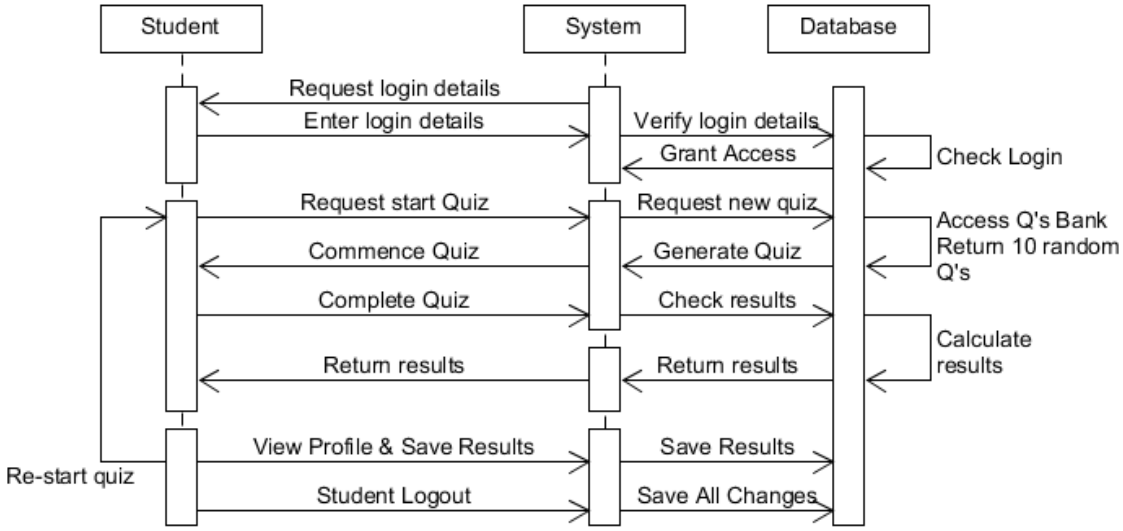

Figure 8 - Detailed Admin Sequence Diagram

Refined Class Diagram

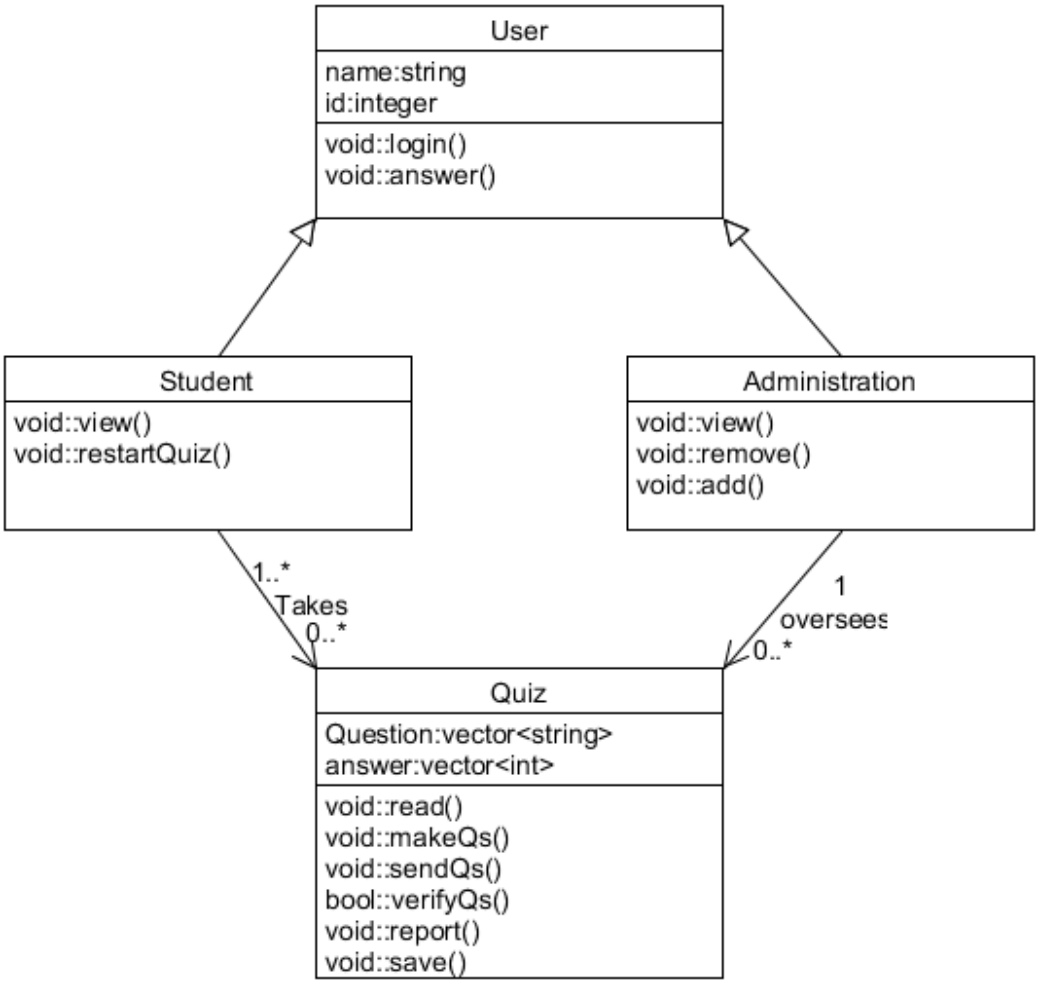

Figure 9 - Refined Class Diagram

Implementation Recommendations & Notes

Use of an xml parser such as 'TinyXML2' is recommended for storing and reading data in to the quiz project. This allows you to have multiple xml documents for different quizzes which can each be read in to the project.