Student Information Management System

(SIMS) Project Report

**Pofessor:**

Yuchou Chang

**Class:**

Software Engineering, CS 4320

**Date:**

4/20/2018

Table of Contents

[Introduction 3](#_Toc481354328)

[Requirements 3](#_Toc481354329)

[Analysis 3](#_Toc481354330)

[System Development Life Cycle 4](#_Toc481354331)

[Team Work 4](#_Toc481354332)

[Version Control 4](#_Toc481354333)

[Apply UML Method 5](#_Toc481354334)

[Student 6](#_Toc481354335)

[Administrator 6](#_Toc481354336)

[Database 6](#_Toc481354337)

[Graphical User Interface 7](#_Toc481354338)

[Team Model 9](#_Toc481354339)

# Introduction

There has been a request by a University IT Department to create and implement a computerized student information management system. A group of 5 Computer Science under graduate students got together and created the Student Information Management System (SIMS). This is intended to help the IT department track and manage the students schedule and information.

# Requirements

Here is a list of requirements listed by the university, to show what they seek in the system.

* Stores and retrieves student information (first/last name, ID, GPA, courses enrolled, etc.)
* Allows administrator privileges (search student ID, search CRN, view semester, edit grades and GPA)
* GPA Calculation (2 exams and final), with other class grade information
* Two login domains, student and administrator

# Analysis

The purpose of this project is to help current students in a University to view their grading information, course schedule and GPA. It is also intended for administrators to manage and edit student information and course schedule and course grades. The software created is named Student Information Management System (SIMS). The key point of this system is to produce a complete requirements specification for SIMS and analyze the requirement specification of the possible new student or current student information module for the school administrators. The choice of this system development method will heavily affect the quality of the system being produced. SIMS will provide capabilities for viewing courses, grading, results of student tests, course grade and calculate GPA. Therefore, choosing suitable system development method is important to create trouble free system.

The SIMS software will also allow students to view their grades for current or previous semesters. If the students have grades, the grades will reflect their GPA which will be calculated by the software and input into the database. Student will only be allowed to view their information and not allowed to add or update any information. Administrators can view, add and update student grades. Administrators will also be allowed to modify students grades and class schedule.

The information required to complete this project could be split into three parts.

* Appropriate planning of how to approach with beginning this project with knowing the requirements.
* To design and implementation the software.
* Test in order to make sure if everything is working as per client’s requirements.

To fulfill the all the parts of the SIMS software, several discussions with team members were formulated to plan on how to approach the completion of the SIMS software with keeping in mind the client’s requirements. The design and implementation of SIMS were done using the IDE’s: MySQL for the database and Visual Studio using Visual Basic programming language. UML Diagrams were designed to help visualize the structure of SIMS and the database. Final step of testing, use of different methods approach done to received accurate result as of client requirements.

# System Development Life Cycle

The life cycle model chosen is the Waterfall Life Cycle Model because it is document driven throughout the entire software development process. The Waterfall Life Cycle Model also works extremely well for small projects and each phase is completed and reviewed one at a time.

# Team Work

The members of our Software Engineering group are Kirt Registe, Rohan Thaliachery, Jon Viesca, Jake Sceroler, and Devindrasinh Varachhia. Rohan completed the database information inserting which included all student, admin, and course information. Her contributions to the final report consist of the introduction of the team members and the analysis report. Brooke ensured that all of the client’s requirements were fulfilled and constructed the life models and UML diagrams to ensure that the clients were insured we would meet or exceed their requirements. Jon designed the GUI and the description of the group’s teamwork and team model for the final report. Jake completed the version control description. Kirt and Devin both finalized and created the Visual Studio code to connect the GUI and the database, which was created using MySQL Workbench 5.7. Abdurrahman compiled and submitted all the finalized artifacts of the project such as the GUI, Database files and power point presentation for the entire group.

# Version Control

Github was used as the version control. Please visit <https://github.com/rivaside/Software-Engineering-Project.git> to see all the coding and artifacts that lead to the final project. Please look for the folder “Final Project,” to see the final parts of the project.

# Apply UML Method

Below are the UML diagrams that describe the database and SIMS software structure.

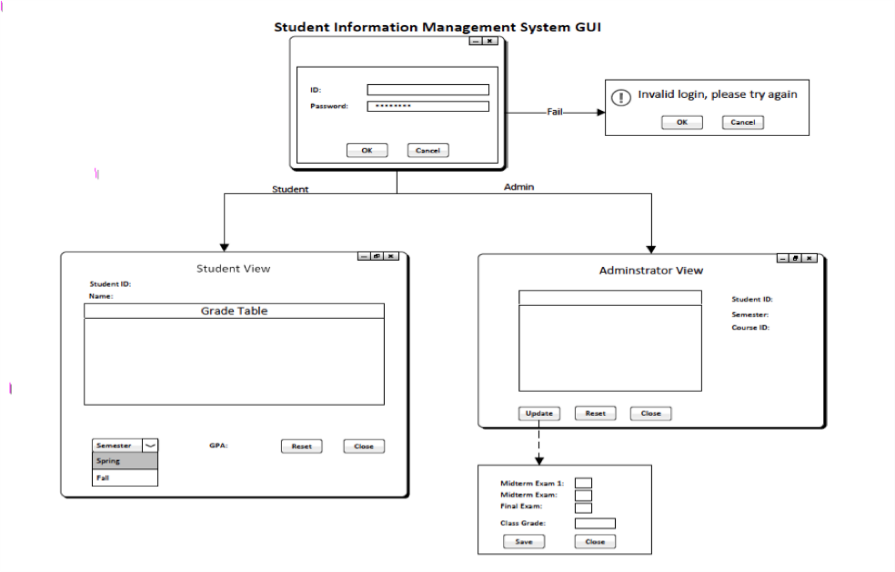


Figure 1: Architecture Diagram

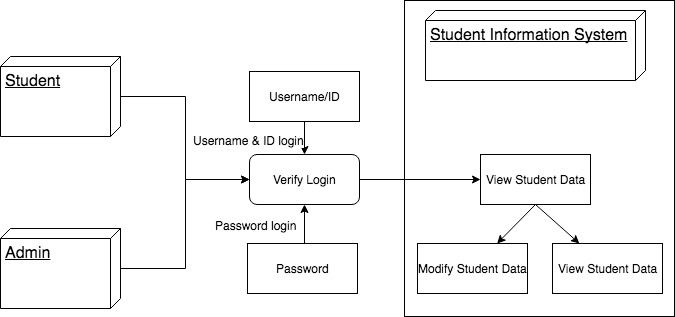


Figure 2: Data Flow Diagram

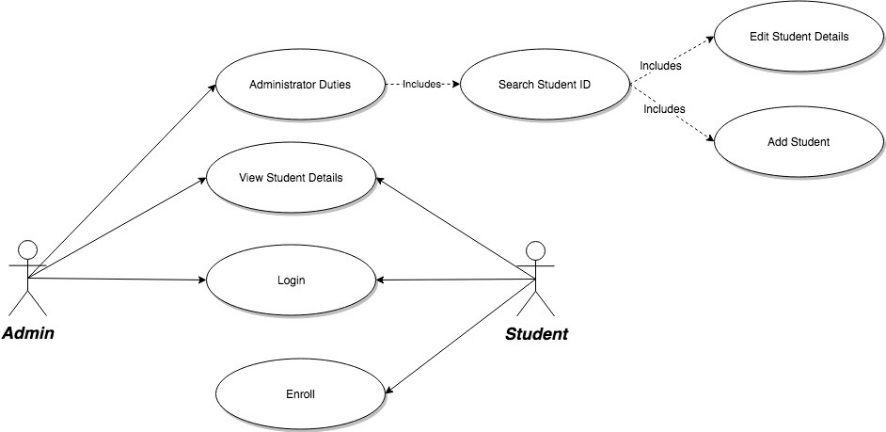


Figure 3: Use Case Diagram

Below are the user privileges that were requested by the clients, which are fully fulfilled in SIMS.

## Student

* Login using student ID and password
* View Student Details
  + Classes currently enrolled
  + Grades
  + GPA
  + View by semester

## Administrator

* Login using admin ID and password
* View Student Details
* Classes currently enrolled
* Grades
* GPA
* View by semester
* Edit Student Details
* Classes currently enrolled
* Grades
* GPA
* View by semester
* Search Student Details
* Semester
* CRN Number
* Student ID

# Database

Database is a gathering of information that is organized so that is can be effortlessly accessed, managed and updated. Database use for three type of transactions: Insert, Modify, and delete. The important step structure the information to be store in the database. A database use to reflect information by using entity-relationship model.

The SIMS database has been created using MySQL Workbench 5.7. In the SIMS database there are six tables:

1. Student personal information
2. Scheduling information
3. Course information
4. Grading information
5. Student GPA information
6. Admin access information

Whenever user as student or administration access, MySQL verify their user access and password to allow in to software. MySQL and Visual Basic are connect to provide all information for user. If any changes are made thru Visual Basic it will also update in the SIMS database.

# Graphical User Interface

Below are the different forms for the SIMS software.

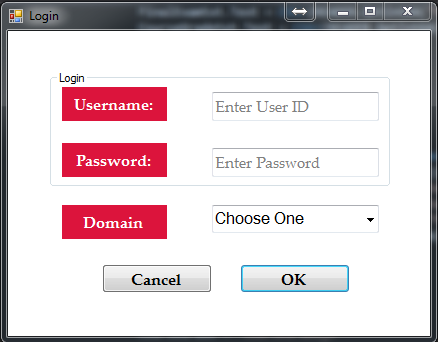


Figure 4: Login Form (Student and Admin)

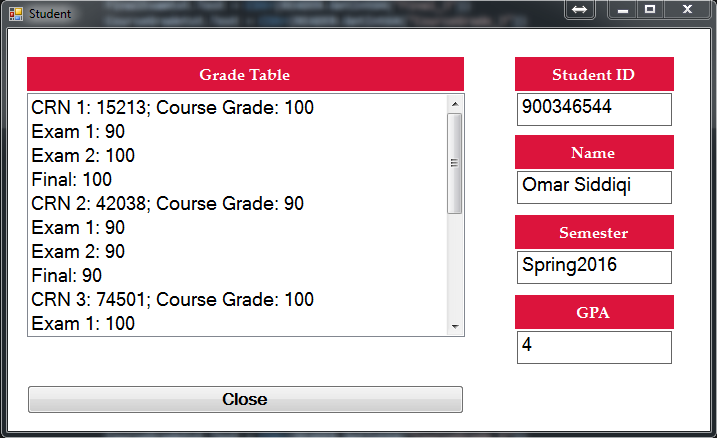


Figure 5: Student View

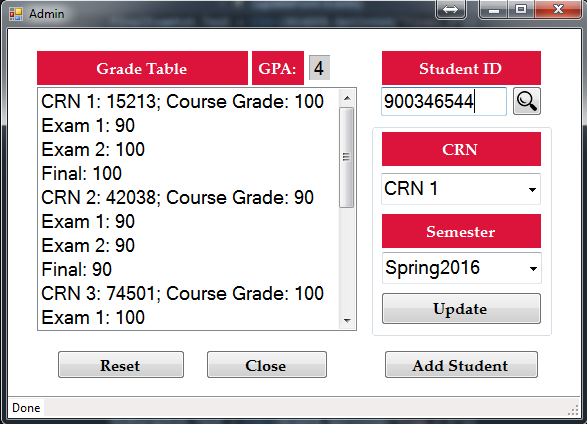


Figure 6: Admin View

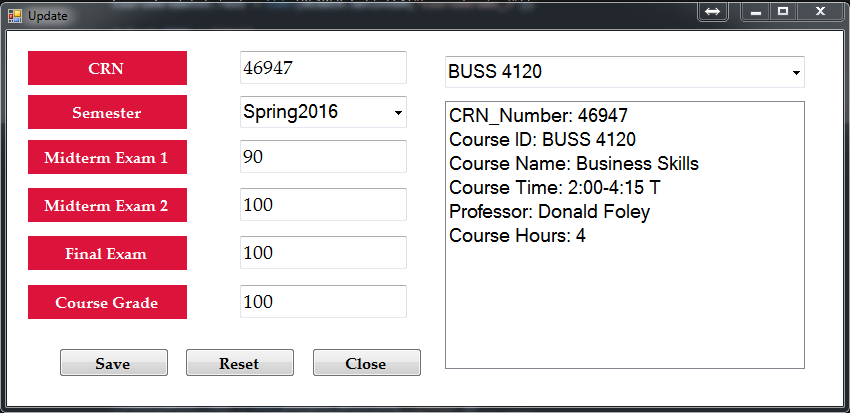


Figure 7: Admin Update Form

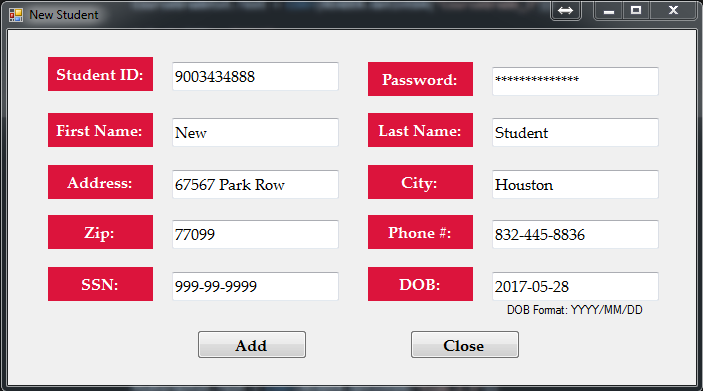


Figure 8: Admin Add Student Form

# Team Model

For our Software Engineering project, we chose the modern hierarchical approach. Each team member was assigned a role. Devindrasinh Varacchia was assigned the role of the team leader programmer and Jake Sceroler was the backup programmer, which together used Visual Studio. Jake Sceroler created all of the UML Diagrams and presented them to the clients and was the programming secretary. Rohan Thaliachery created the database in MySQL. Kiyunna Flourius designed and programmed the GUI. The highly skilled team manager, Jon Viesca was responsible for creating the critical section of code that connected the database and GUI. He did the architectural design, allocated the coding among the team members, and reviewed the work of other team members. Devin was as competent as the team leader and he was extremely knowledgeable about the program. He completed all black-box test case planning and participated in the designing process.

# Implementation

Once our team put together the initial requirements for our project. We began work on refining the initial requirements to pull out the features needed to create the program while encompassing as many features as possible. Once we completed the design phase Jake and Kirk begin working on implementing our designs. They started first with implementing the login feature so that the user could log into the system. Our login screen was our first unit to be implemented as it was our initial interaction with our system. Our system had to be able to read the information entered and check it against our database. Rohan and Devin worked to create the necessary code to be able to connect our MySQL server with VisualBasic(VB) using Visual Studio. Next we implemented the second unit which was the Student view. If a student was to log in they could view their grades, G.P.A, and list of courses they are enrolled in for the current semester. Thirdly they worked to implement the last unit in our system, the Administrator login screen. From here the administrator could view grades, add grades, alter grades, and add students. Then the last unit for implementation was the G.P.A. This function was something that was to be calculated through our system based on the information that Administrator enters and what semester the student is looking at.

# Testing

For the testing phase we worked together to try to test each individual unit by going through our different test cases established in the Requirements phase of our workflow. So, we began running test cases on our login screen. Using a mix of username and passwords that where not in our database to make sure that someone who is not in the database does not have access. It was an initial success until we started using information contained in our database. We ran into issues with our login screen not checking the information against our database. Which led to issues of no one being able to log in. The fault was documented and Omar rectified the issue by changing the access commands to the database. Our group re-tested the login screen to make sure we could log in when providing the correct information and were denied access when we provided the wrong information. Then we tested the login unit to make sure that if a user entered in credentials for a student they were shown only the student view and if the user entered in administrator credentials they were shown only the administrator view.

Once unit testing of the login screen was completed we moved on to testing the second unit, Student View. We logged into several student accounts to verify that the correct student’s information was pulling across correctly and showing the proper grades as well as courses in our database. The Student Unit was rarely straight forward as our group did not have any issues with this unit.

Next our group began working on the Administrator Unit to make sure all the necessary features were added properly. First, we started with making sure we could add information to our database through the administrator view. We were able to add information however the information was making the necessary changes in our database. The fault was recorded and Kirk was able to come up with a solution to the problem. Edrick made changes to the sql command being used in the code to make changes to the database to fix the issue. Afterwards we began adding students to our database then checking to verify they were added using the administrator view. With that feature completed our group moved on to implementing the next feature which would allow us to change a students information based on our use cases. If a the administrator accidentally entered in the wrong grade they would be able to change it or if they need to add another class to the student’s profile they could.

Last but not least, probably the hardest part of our system was making sure that whenever the administrator makes changes to the Student’s information the G.P.A would reflect that change no matter what it was. If the administrator had to update, delete, or add a grade to the student’s profile the G.P.A would reflect those changes. Omar and Edrick worked to fix this issue was our G.P.A did not work as implemented. They made several changes to the code until they were able to get the G.P.A to work as intended based on the requirements. Initially the G.P.A was calculated within the database but that caused issues with our database trying to accurately calculate it. Eventually they resolved to code the G.P.A into our system and referencing the information in our database than adding the calculated G.P.A to our database.