Basic Software Requirements Specification

Class Roster

Written by: Amiel Nava

February 17, 2021

Table of Contents

1 Introduction and Overview 2
1.1 Purpose 2
1.2 Scope 2
1.3 Definitions, acronyms, and abbreviations 3
1.4 Overview 3
2 Overall description 3
2.1 Product perspective 3
2.1.1 System interfaces 4
2.1.2 User interfaces 4
2.1.3 Hardware interfaces 4
2.1.4 Software interfaces 4
2.1.5 Communications interfaces 4
2.1.6 Operations 4
2.1.7 Site adaptation requirements 4
2.2 Product functions 5
2.3 User characteristics 5
2.4 Constraints 5
2.5 Assumptions and dependencies 5
3 System Requirements 5
3.1 Functional Requirements 5
3.1.1 Use Case Model 5
3.1.2 Use Case Diagram 5
3.1.3 Use case diagram description 6

3.2 Non-functional Requirements 7
4 Other 7
4.1 Potential Risks 7
4.2 Potential Future Changes 7

1 Introduction and Overview

This section will introduce the Class Roster Application and will give a quick overview of this document.

1.1 Purpose

The purpose of this document is to set a concrete specification on what the *Class Roster* application will do and how will the software interact with the user. This document will also address the *Class Roster*'s attributes and its design constraints. The intended audience for this document is for the stakeholder and the developing team to have a concrete understanding on what the application will and will not do.

1.2 Scope

The *Class Roster* application will be used to display the roster of a particular class. The roster will display student's information in the roster. The user of the application will have the availability to sort the students in the roster. The user will also be able to add a student or remove a student from their class roster. The *Class Roster* will not have the availability to edit a student's information and will not contain any faculty information. Also, the *Class Roster* will need to be able to run with no web connectivity.

The goal for this application is to have a professor the availability to view the students in their class and add or remove students before the registration deadline. This will give the professor the power to remove students who do not meet requirements or have not attended the class and add students who are in the waitlist or crashing the class.

1.3 Definitions, acronyms, and abbreviations

Term	Definition
Admin	Short for administrator. The administrator is a person that will have the highest privileges in our system.
CSV file	A comma separated value file that will be used to read and write database information.
Database	A collection of data structured in a particular way
IT department	Information technology department that oversees the school's technology such as the school's server.
Java	A programming language to build our application
Query	An instruction given to the database

1.4 Overview

The rest of this document will be structured as follows:

Section 2 will talk about the interactions of all entities in our system and what functions will they be able to achieve. This section will also talk about who will be using our system and the constrains of our system.

Section 3 will go more in detail what the functions of our system will be available to users and the qualities that our system will require. Or in other words, what our system will be capable of doing.

Section 4 will address potential risks and potential updates to our system.

2 Overall description

This section will describe factors that will affect the *Class Roster* and its requirements.

2.1 Product perspective

This application will be similar programs online that display a roster of people such as public salaries any other public database. The following subsections will describe the interfaces that this application will use.

2.1.1 System interfaces

The *Class Roster* will interact with the school's database in order to add and remove any student in the class. Although it will use the computer's network to do so, it is not necessary for the application to have web connectivity in order to view and add or remove students in the roster.

2.1.2 User interfaces

The *Class Roster* application will be a console-based user interface that is run from the school's computer. The application will display the available class rosters for that user. Each roster will have the class date and time and the course title. The application will have key inputs to select a roster and add or remove a student from the roster.

The application will require a user to enter a username and a password. When a user would like to add or remove a student, the application will just need the student's ID number and will only accept positive integers as input for the student' ID.

2.1.3 Hardware interfaces

The software will run on the school's computer and will require a keyboard in order to interact with the application. The console-based user interface will only accept keyboard keystrokes as inputs.

2.1.4 Software interfaces

The *Class Roster* application will either run on Windows or in Mac since the school has the option for its computers to boot in either one. The software will require the computer to have Java installed.

2.1.5 Communications interfaces

The application could be connected to the school's database in order to retrieve a student's information or update the database, but it is not needed after the application runs for the first time in the current semester so it can retrieve the class roster. The class roster can also be obtained by the IT department and the csv file can be placed in the user's computer.

2.1.6 Operations

The *Class Roster* application will automatically log out after five minutes if there is no activity in the application in order to protect sensitive information of students. The software will prompt for username and password to login and use the software.

2.1.7 Site adaptation requirements

The school's computer has a separate software that will update and retrieve the school's database and is independent of the *Class Roster* application. That software will create a csv file where our *Class Roster* program will read the roster and will update it. Then that

separate software will use the csv file to update the school's database. The csv file will need to be located in a particular directory in order for the school's software and our program will read and write in that csv file.

2.2 Product functions

The *Class Roster* application will be used to display the roster of a particular class. The roster will display the following student's information: ID number, first and last name, major, and year of expected graduation. The user of the application will have the availability to sort the students by name or by student ID and will have the availability to add or remove a student before the registration deadline. The add function will be disable if the maximum class size has already been reached for that class.

2.3 User characteristics

The intended user for this product is for a college professor with a high level of education and having basic computer skills.

2.4 Constraints

The software will not have any mouse support; therefore, an option of keyboard inputs must be always displayed to help the user interact with the software. This software will also not have the most updated class roster if is not connected to the internet/network, hence it must display a warning message that the roster is not updated because is not connected to the network.

2.5 Assumptions and dependencies

This application assumes that the school's computer is a Windows or Mac operated system. The software may need to make some changes if it would be used in a Chrome OS or Linux based OS, but since the software will be made in Java, the changes will be minimal or none due to Java's portability.

3 System Requirements

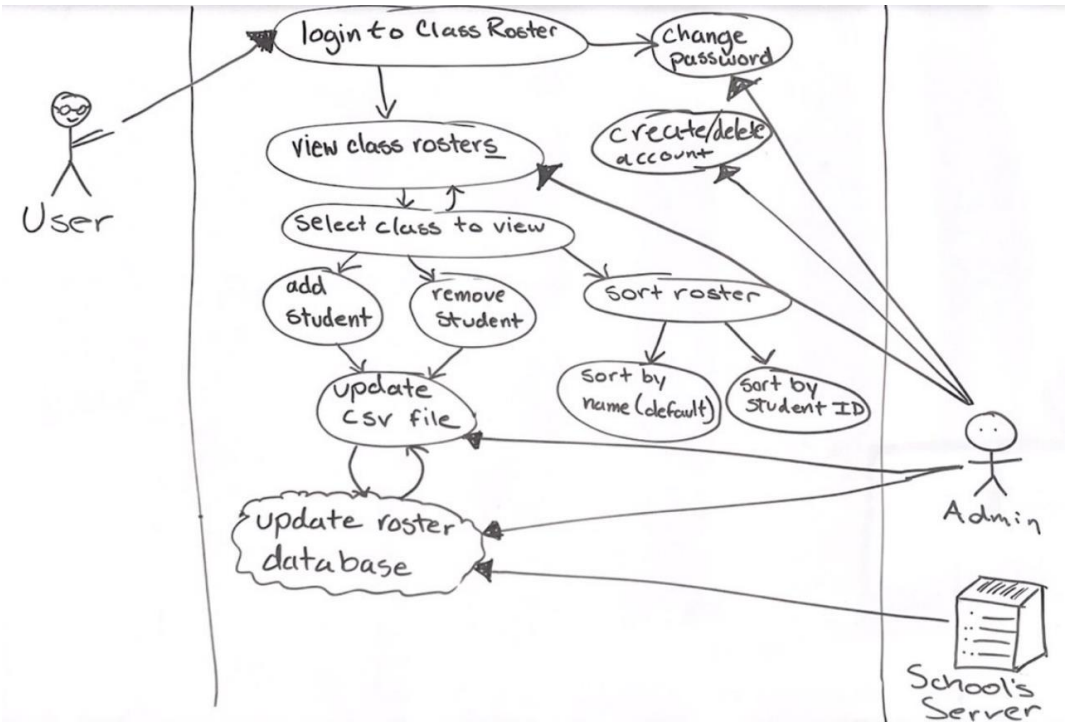
3.1 Functional Requirements

The application functions will be entering username and password, change password, add/remove a student, display class roster, sort roster by name/ID, enter the available rosters menu, and select a roster to view.

3.1.1 Use Case Model

The following diagram will be used to visually describe the user's and other entities in our application.

3.1.2 Use Case Diagram



3.1.3 Use case diagram description

User:
The main functionality of the system is to have a school faculty view a class roster that they have access to view whether is a professor looking at their class or a counselor looking for open spots. The users will have the availability to add a student to their class or remove them. The user will also have the availability to sort the view of the class roster either by student's ID number or by student's name which will be the default setting. In order to achieve the above, the user must login to the system using their username and password which will be provided by the school's IT department. The user does not to be connected to the internet in order to achieve these functions.

Admin:
The school's administrators will be in charge in setting up users to use the system by initializing an account. The admin will have the availability to reset the password of a user. The database for the information containing the class rosters will be available to the admin to locally install the rosters in a computer with no web connectivity. The admin team will also have the availability of all functions the user has and will have the availability to update the class roster database.

School's Server:
The school's server will be the entity that will upkeep the database containing the student's information and check the validation of an update. The server will just update the csv file that the application will need to function properly.

3.2 Non-functional Requirements

This section will describe the desired qualities of the system.

The first desired quality is that the system must have a maximum size of a particular class and users can only add or remove a student before the registration date. After the registration date, the user will not be able to use the add or remove functions. Another desired quality is having users with different level of controls like being able to view all the available class rosters from every class available in the school. With users with such privileges, the software will recommend having an automatically logging out function when the application has been inactive for more than 5 minutes in order to protect sensitive student data. Another security concern is someone with no access granted will add or remove a student, hence a password must be asked when such function is to be used. Also, since the csv file will have the stored student's information it will be needed to have the application read and writing rights to the directory containing the file.

The display qualities must also be addressing such as having the user to be able to always see the key input options like the available options whether they are in the select class roster menu or in the view class roster. Another desired display is to have a warning display to the user that they are not connected to the internet and the classroom roster may not be up to date and any updates will not be made in the database until connected to a network.

4 Other

4.1 Potential Risks

Is up to the IT department to revoke privileges to users who are no longer faculty members of the school. The system will also not be responsible if a certain student is in the system that will be decided by the school's database.

4.2 Potential Future Changes

The system could be updated in order to let the user know that a particular student ID is not valid, or a student does not meet the prerequisites to be enrolled in the class. Another potential change will be to view the students waitlisted in class roster and automatically add student by rank when a student is removed or a class maximum size is extended.