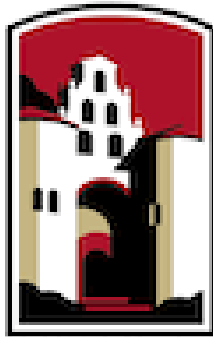


Software Design Specification  
Class Roster Software  
Version 1.0



SAN DIEGO STATE  
UNIVERSITY

Team  
Amiel Nava  
Anoodnya Sangam  
Shane Moro

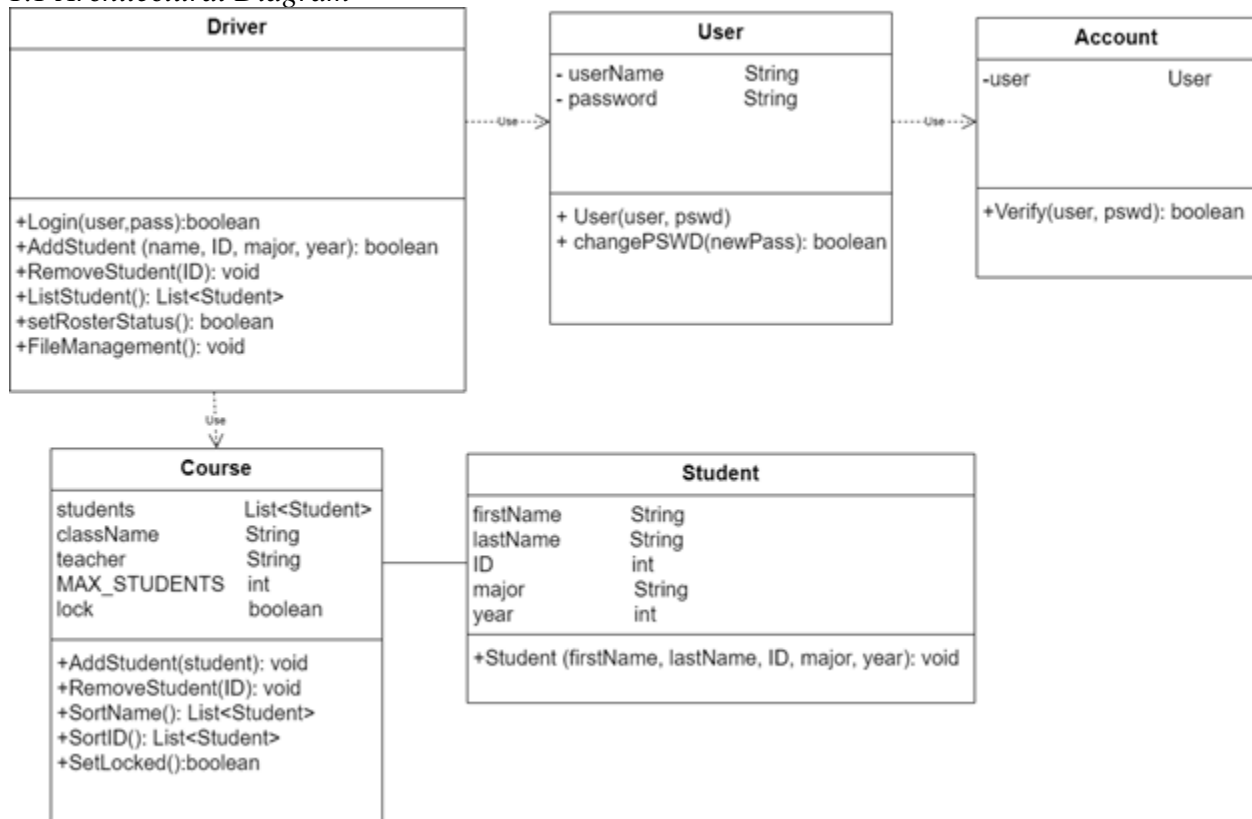
March 12, 2021  
CS-496 Intro Software Systems

## Table of Contents

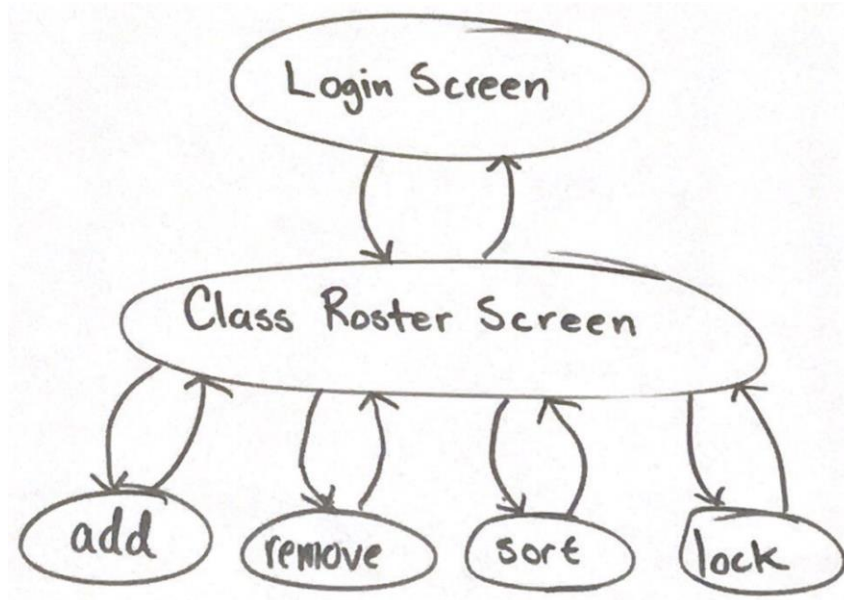
<b>1.0 Software Architecture Overview .....</b>	<b>3</b>
<i>1.1 Architectural Diagram .....</i>	<i>3</i>
<i>1.2 Top Level Design .....</i>	<i>4</i>
<i>1.3 Descriptions of Classes.....</i>	<i>5</i>
<i>1.3.1 Driver .....</i>	<i>5</i>
<i>1.3.2 User .....</i>	<i>6</i>
<i>1.3.3 Account .....</i>	<i>7</i>
<i>1.3.4 Student .....</i>	<i>8</i>
<b>2.0 Development Plan and Timeline .....</b>	<b>9</b>
<i>2.2 Partitioning of Tasks .....</i>	<i>9</i>
<i>2.3 Team Member Responsibilities .....</i>	<i>9</i>

## 1.0 Software Architecture Overview

### 1.1 Architectural Diagram



## 1.2 Top Level Description



The flow of the program is described on the above state diagram. First, the user will be prompted with a login screen when they start up the application. The login screen will prompt for a username and password. After a successful login, the application will go directly to the class roster screen which will show the roster with the default sort view along with a menu on the bottom or on the top listing the available functions to the user. Those functions are adding a new student, removing an existing student, sorting the class roster by name or id, and locking the class roster. Each of these functions will display instructions and will have the option to cancel the function and go back to viewing the class roster. Also, each of these functions would display an error if the user entered an incorrect input. Lastly the class roster screen or in our case the display of the class roster, will have the option to exit the program and go to the login screen.

The driver class will be in charge for the flow of the program using the other classes such as user student, course, and account. These classes will be used to use to let the user edit their class roster and display the class roster.

## *1.3 Description of Classes*

### 1.3.1 Driver

- Attributes:
  - o N/A.
- Operations:
  - o Login: This method takes in a User object, and calls the verify method in the Account class, and returns a boolean true if the login was successful, and false otherwise.
  - o addStudent: This method takes in identifying information of the student, and calls the addStudent method in the Course class, to add the student to the class roster. It returns true if successful, false if not.
  - o removeStudent: This method takes in the ID of an existing student, and calls the removeStudent method from the Course class to remove the student from the class roster.
  - o listStudent: This method has no parameters, but will prompt user to choose if they want to sort by name or by ID, and then return a List of Student objects.
  - o setRosterStatus: This method calls the setLocked method in the Course class, which allows the user to lock and unlock the roster. Returns true if successful, false otherwise.
  - o fileManagement: This method will save the current class roster into a file, so that it will be accessible in between sessions.
- Interactions:

- o Driver class is dependent on User class and Course class. User class contains user info and passwords that Driver uses during login. Course class contains the list of students that Driver class interacts with by add/remove students, list students, or save to a file.

### 1.3.2. User

- Attributes:
  - o userName: This string type is used to identify a user who has an account to use the system.
  - o password: This string type is used to validate a user into a session in the system to view and edit a class roster.
- Operations:
  - o User: This is a constructor for the User class, that creates an account with a user and password that allows the user to login in to the system.
  - o changePSWD: This method takes in a string, their new password, and changes their current password to the new string provided.
- Interactions:
  - o User class interacts with Account class and Driver class. User class is dependent on Account class. Driver class is dependent on User class. Driver class uses User class to store usernames and passwords.

### 1.3.3. Account

- Attributes:

- o user: A User class type that is used to specify a particular user to an account.
- Operations:
  - o verify: This method takes in strings, the username and password that the user entered, and compares it with the current password on their Account. If it matches, it returns a true, and returns false otherwise.
- Interactions:
  - o This class interacts with the User class. The User class is dependent on the Account class to verify username and password.

#### 1.3.4. Course

- Attributes:
  - o students: A list of student class types. This list is used to specify the list of students in a particular course.
  - o className: A string type containing the name of the current course or class name.
  - o teacher: A string type containing the name of the professor or teacher of the course.
  - o MAX\_STUDENTS: A constant integer type that will determine the maximum number of students allowed in the current course. This constant will be useful to automatically lock the class roster.
  - o lock: A boolean type used to determine if the class roster is lock (true) or not (false).
- Operations:

- o addStudent: This method takes in a Student object, and adds them to the course roster.
- o removeStudent: This method takes in the ID of the student and removes them from the course roster.
- o sortName: This method sorts the students in the class roster (which is a List<Student>), and returns a List<Student> that is sorted alphabetically by last name.
- o sortID: This method sorts the students in the class roster (which is a List<Student>), and returns a List<Student> that is sorted numerically by ID number.
- o setLock: This method locks the roster so that no more students can be added to it. Returns true if successful, false if not.
- Interactions:
  - o Course class interacts with Driver class and Student class. Driver class is dependent on Course class for course information and student list. Course class is associated with Student class. Course class contains list of Students.

### 1.3.5 Student

- Attributes
  - o firstName: A string type that has the first name of a student.
  - o lastName: A string type that has the last name of a student.



- o ID: An integer type that identifies a unique student. It is used to identify a student even if the student has the same name as another student. The unique id is assigned by the application.
  - o major: A string type that will contain the student's major.
  - o year: An integer type that will contain the student's expected graduation year.
- Operations:
  - o Student: This is the constructor for the Student class. It takes in a string first name, string last name, int ID, string major, and int year, and creates a Student object that represents a physical student.
- Interactions:
  - o Student class is associated with the Course class. The Course class contains list of students.

## **2.0 Development Plan and Timeline**

### *2.2 Partitioning of Tasks*

- Anoodnya: Will oversee displaying the class roster along with the available functions to the user.
- Amiel: Will oversee implementing the functions that will edit a class roster and read and write the state of the class roster in a csv file. Amiel will also make sure the information is persisted between sessions.
- Shane: In charge of the user and account classes that will validate a user. Those classes will let a user access the class roster. Shane will be setting up the other classes.

### *2.3 Team Member Responsibilities*

- Shane will be responsible for setting up group meetings and getting feedback from the stakeholder (office hours).
- Amiel will be responsible for debugging the system and reviewing the software or documentation.
- Anoodnya will be responsible for integrating each developers' assigned class into project.