CPSC 5011 - HW3 Design / CRC cards:

Group 3: Divya Arisetty, Grant Bishop, Tong Ding, Yi Niu, Neha Sahu

0. Problem statement

The key nouns (to potentially be classes) and **verbs** (part of use cases) in the problem statement where identified:

On SU Online, students are able to **search and register** for classes. In order to **add** a course, students need to meet the prerequisites and the class may not be full. Students may also **drop** a class they decide not to take before the registration deadline. Students can only take a maximum of 18 credits if undergrad or 12 credits if grad student.
Students must be **logged into** SU Online before attempting to add or drop, but can search for classes either logged in or not. If the class is full, students must **send** an email to their advisor with their name, SUID, and the course they want to take; students will be put on a waitlist for the requested course. If space allows, students will be automatically **enrolled** in the order they are listed on the waitlist.

1. Use Cases

The following flowchart provides an end-users perspective of all available actions:

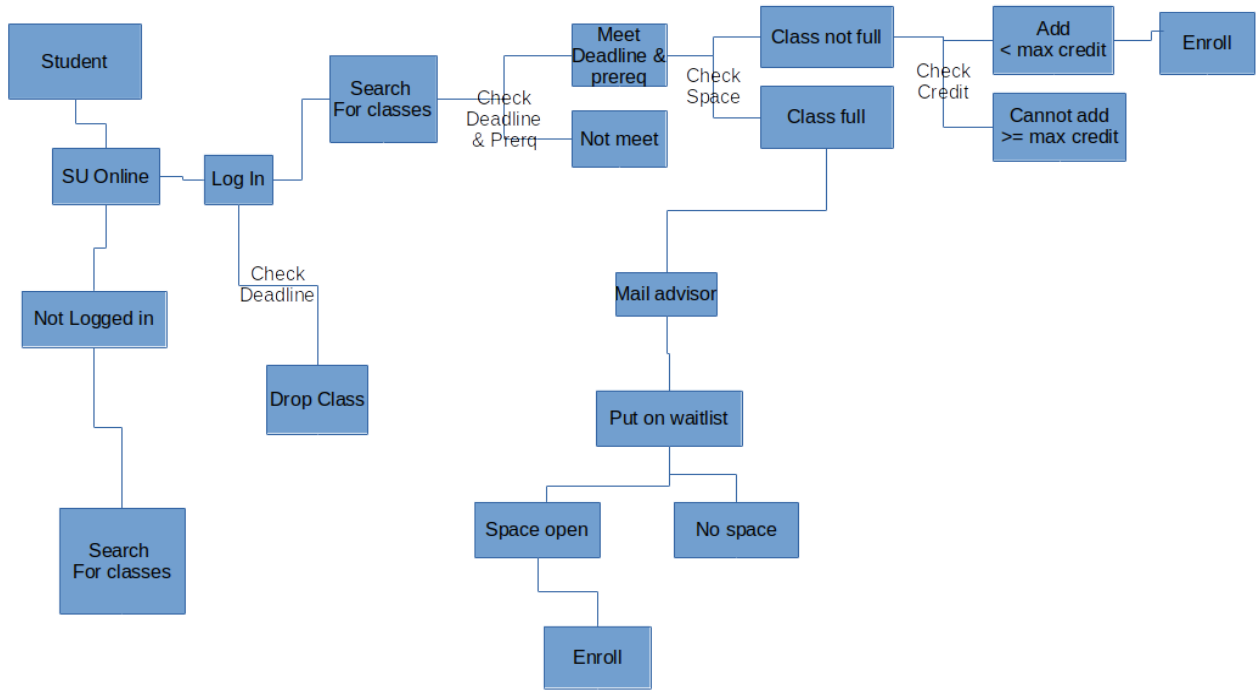

Figure 1. The flowchart indicating possible use cases. Detailed word explanations are followed.

- A student searches a course on SU Online without logging in.
- A student logs in on SU Online, searches for a course but cannot add/drop it because the registration deadline has been passed.
- A student logs in on SU Online to drop a course before the registration deadline.
- A student logs in on SU Online, searches for a course to add but cannot add it because the student hasn't taken all prerequisites.
- A student logs in on SU Online, searches for a course to add. The course is not full, but the student cannot add it because that will exceed his/her maximum credits.
- A student logs in on SU Online, searches for a course to add, and adds the course.
- A student logs in on SU Online, searches for a course to add. The course is full, so the student emails his/her advisor to be put on the waitlist. Eventually, the student may or may not be enrolled depending on whether or not spots open up.

2. Class identification

After Identifying all the important nouns in (0. Problem statement), potential classes were listed:

Probably	Possibly	Rejected
SU Online Student Course	Waitlist - a FIFO queue Advisor - joins student and waitlist Registration - allows a student to add/drop classes after he/she logged in SU Online Class - a course may contain several classes (different sessions). In this case, we simplified this scenario by using the course instead.	Prerequisites: the property of the class Name: the property of the student SUID: the property of the student Space: the property of the class Deadline: the property of the registration system or class

Table 1. Identified "probably", "possibly", and "rejected" class candidates.

Class candidates: SU Online, Student, Course, Registration, Waitlist, Advisor

Decisions making in more detail:

SUOnline / 'Registration System': There need to be one or more classes that deal with the 'front-end,' providing actions to the users. These may be analogous to the Driver and Menu classes in HW 2.

SUOnline: allows the user to search for classes. Contains actions apply to all types of users. Stores schedule (course) information -> class

Registration: allows the *logged in user* to register for classes (or drop or add to the waitlist) if conditions are met. -> class

Student: There is a lot of student-specific data to store (name, GPA, previous classes taken, &c). There are undergraduate and graduate student types. -> class, subclasses

Class/Course: As the student, there are a lot of important attributes to store (title, number of credits, prerequisites), and most importantly a list of enrolled students. A class has a list of enrolled students and a waitlist. A course also may have multiple sections -> class, subclasses

Name: a property of student -> not a class

SUID: a property of student -> not a class

Deadline: a property of the class (if different for each class) or of registration (if the same for all classes) -> not a class (existing Date class?)

Waitlist: the waitlist is a queue of students. Each class will have a waitlist. Students will be added to the waitlist, and may be removed when enrolled -> class, specialized queue (inherits from / composed of?)

Advisor: Receives emails and adds students to the waitlist. May have other functionality? (Creating / editing courses?) -> class

Prerequisites: a course has a list of prerequisite classes, and a student has a list of classes that they have already taken. -> not a class

Space (number of spaces in a class): just an int value or number of indices in a list -> not a class

3. Relationships between classes

Dependency:
The ‘collaborators’ as seen in the CRC cards depend on each other - the diagram in (4.) shows the connections between the classes.

Inheritance:
An [undergraduate/graduate] student is a Student
A CourseSection is a Course

Aggregation and composition:
Class has-a waitlist
Student has-a for many classes
(and the inverse, classes have many students)

Other:
Waitlist could either inherit from a queue or have a queue. (inheritance vs. composition)

4. CRC cards

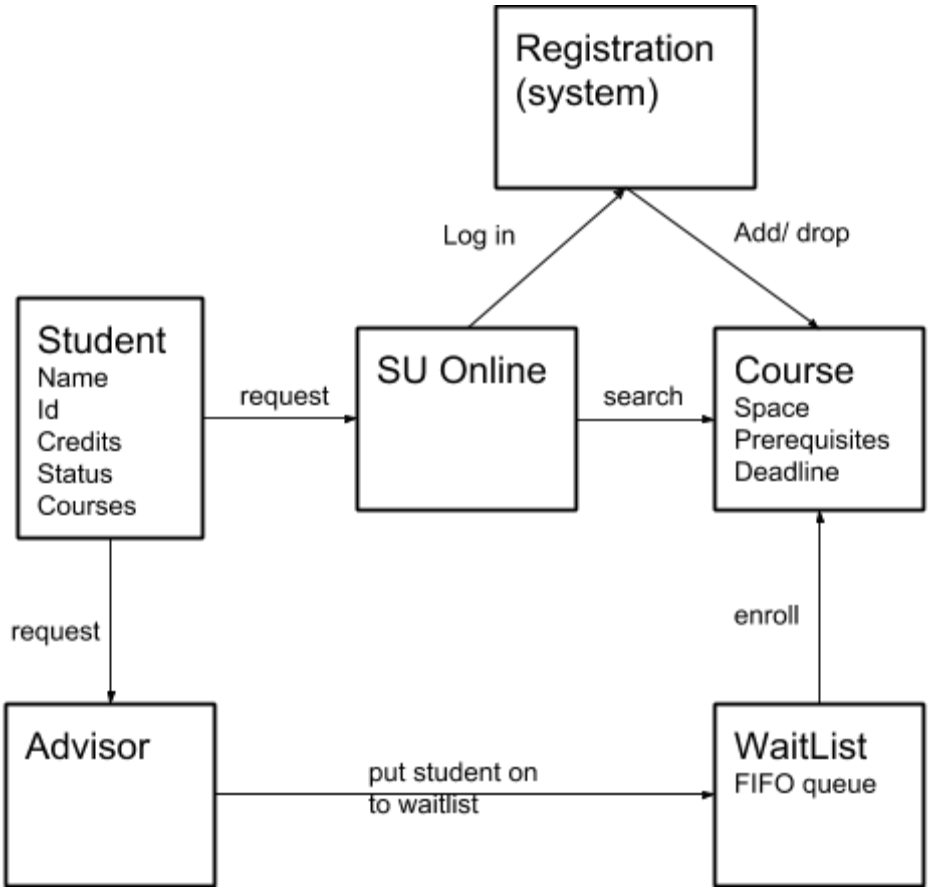

Figure 2. Proposed relationships between identified class candidates. The responsibilities of each class are not showed here but listed in the following CRC cards.

SUOnline	
• Connect user to registration system, take input and keep records • Search for classes with/without log-in • Keep student accounts and detailed info	• Registration • Course

Registration	
• Check deadline for adding/dropping classes • Add/drop course	• - • Course

Student	Graduate/Undergraduate
• Check prerequisites before registration • Check current credits below maximum (Graduate/Undergraduate) • Check if waitlist application needed and apply for waitlist	• Course • Course • Course

Course	CourseSection
• Check if waitlist is needed when exceeding capacity • Register student from waitlist in order when someone drops • Keep records of student including info and grades • Keep list of prerequisites, credit hours, schedule and location	• Waitlist • Waitlist • Student

WaitList	
• Maintain the queue of waiting students (FIFO order) • Accept student from student advisor email • Remove student from the queue (when space exists)	

Advisor	
• Take application of waitlist from student • Add student in waitlist	• Student • WaitList