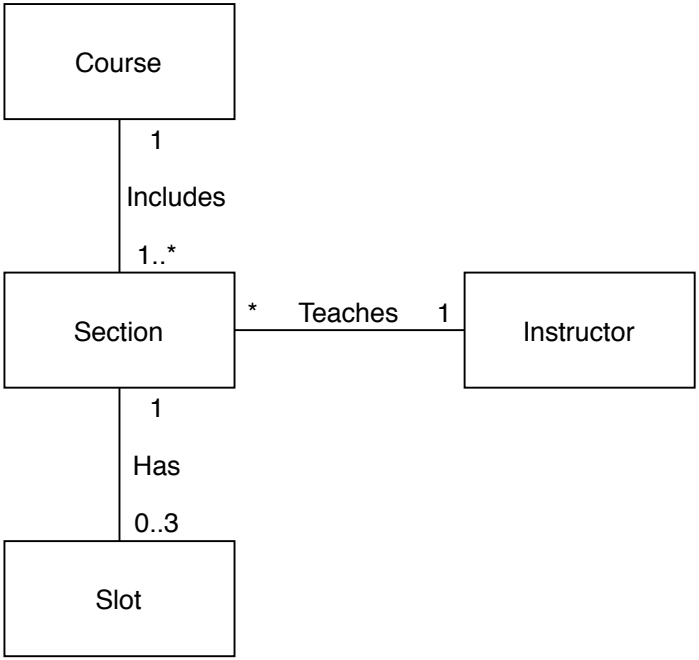
1. Team work

i. Data model diagram



Attribute Table:

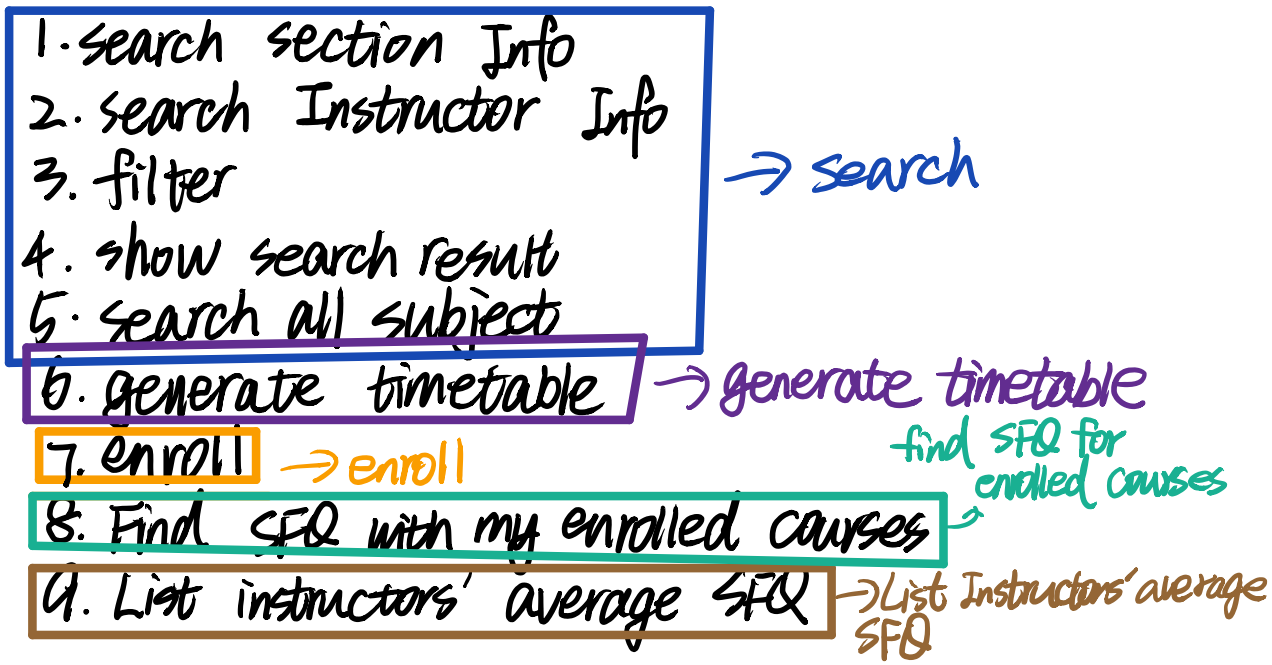
Course:
① title ② description ③ exclusion ④ numslots

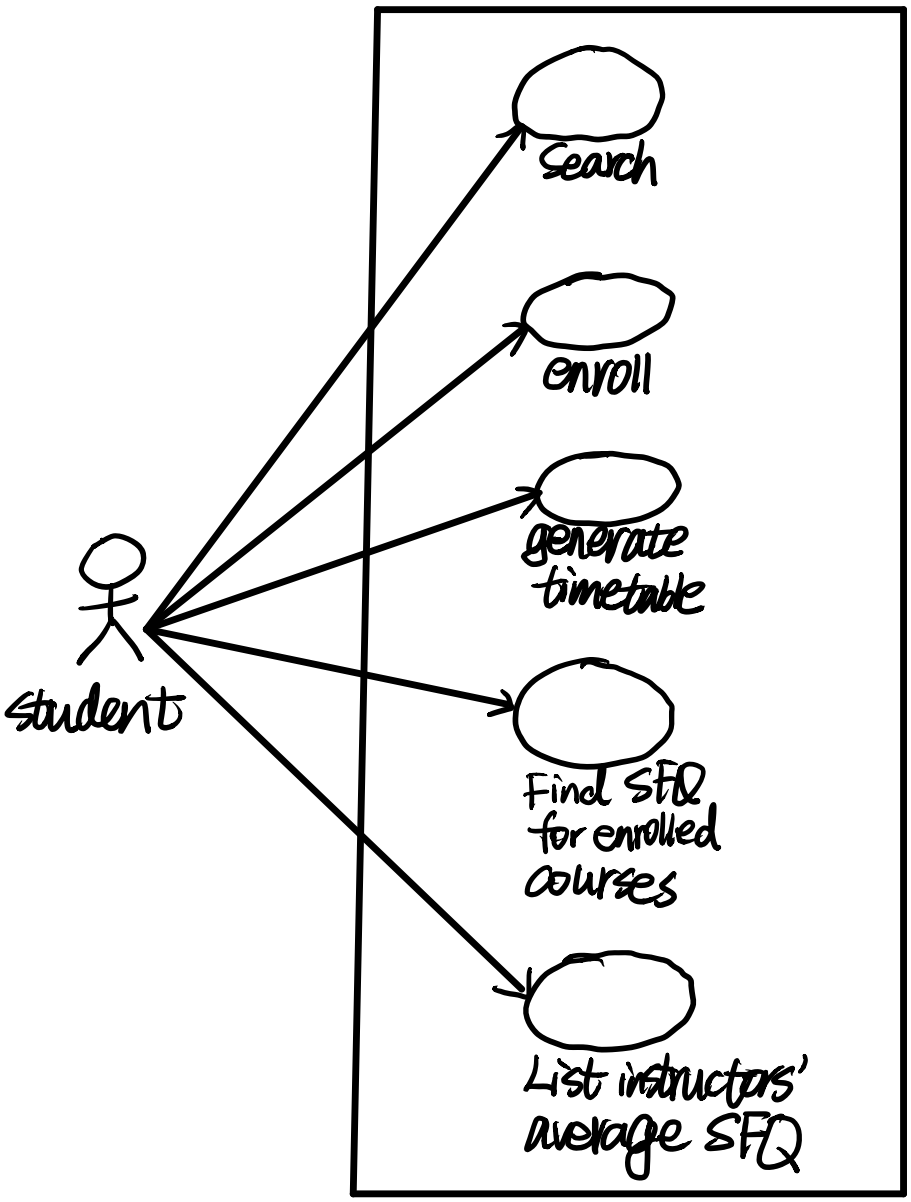
Slot:
① start ② end ③ venue ④ day

Section:
① code ② enrollStatus

Instructor:
① name

ii. Use-case diagram




Search
enroll
generate timetable
Find SFQ for enrolled courses
List instructors' average SFQ
student

iii. Workload distribution:

1. SHENG Ruoheng: Task1 & Task4
2. CHEN Ziwei: Task2 & Task3
3. TIAN Xiangan: Task5 & Task6

iv. GitHub Link:

https://github.com/tommytim0515/COMP3111-Project-2020s.git

2. Individual Work (Use-case specification)

Task 1:

Search: Student can use this system to search all kind of information. The information includes: all the courses, sections and slots available given term and subject; total number of different sections in this search; total number of courses in this search; Instructors who teach in this term and have no class at Tu 3:10pm.

Basic Flow:

1. The system displays the searching interface

{Enter URL}

2. Student indicates the URL he/she wants to search on.

{Enter Terms}

3. Student specifies the term he/she cares about.

{Enter Subject}

4. Student enters the subject he/she aims to find
5. Show all the searching results that mentioned above in console
6. The use case ends.

Alternative Flow:

A1: 404 not found for the URL entered

At {Enter Subject} if the entered URL is invalid

1. display an appropriate message on the screen to notify the users and not throwing error on the system console.
2. The flow of events is resumed at {Enter URL}

Task 2:

Basic Flow:

{Select Filter}

1. The student enter the Filter Page

{change selected boxes}

2.While the student chooses to change (check or uncheck) some selected boxes.

2.1 If AM (or PM) is checked, the system displays only all sections of the courses which has a slot in AM (or in PM).

2.2 If boxes of days of the week are checked, the system displays only all sections of the courses that has slots on the selected boxes.

2.3 If Common Core is checked, the system displays only all sections of the courses that are 4Y CC.

2.4 If No Exclusion is checked, the system displays only all sections of the courses that does not define exclusion.

2.5 If with Labs or Tutorial is checked, the system displays only all sections of the courses that has labs or tutorials.

2.6 If multiple boxes are checked, the system displays only all sections satisfies the requirements.

2.7 If a box is unchecked, the system displays only all sections satisfies the requirements of the remaining checked boxes.

3. The use case ends.

Alternative Flow:

A1: Select All and De-select all

At any point after {Select Filter}, if student click Select All button,

1. All boxes are checked.

2. The text of the button becomes De-select All

2.1 if De-select All is clicked

2.1.1 The text of the button becomes Select All

2.1.2 All boxes are unchecked.

3.The flow of events is resumed at {change selected boxes}

Task 3:

{Select List}

1. The student enter the List page.

2. The system fills the table correctly with the result of the Filter.

{change status of checkbox}

3. While the student changes the status of the checkbox on this tab, the system console displays filtered information.

3.1 The system displays the following sections are enrolled: on the console, and the list of the enrolled sections.

3.1.1The system erases enrolment status when the checkbox is unchecked.

Alternative flow:

A1: Filter not implemented:

At {select List} if the Filter is correctly implemented,

1. the system shows all result scrapped.

2.The flow of events is resumed at {change status of checkbox}.

Task 4:

Generate timetable: The system can generate a timetable automatically for the students. All the sections enrolled will be displayed in the table. And in each block, course code and section code will be shown.

Basic Flow:

1. The student enter the List page.
2. The system fills the table correctly with the result of the Filter.

{change status of checkbox}

3. The student changes the status of the checkbox on this tab to enroll

{Select Timetable}

4. The student enters the Timetable page
5. The system displays the updated timetable
6. Use case ends.

Alternative Flow:

A1: Enrolment not implemented

At {change status of checkbox} if enrolment is not implemented correctly

1. While the system processes the course information
    1.1 If there are less than 5 sections in the scrapped data, enroll to all sections.
    1.2 If there are equal to or more than 5 sections in scrapped data, enroll the first 5 sections.
2. The flow of events is resumed at {Select Timetable}

Task 5:

All subject Search: this is part of the user case, Search. It can search all the subject at a time.

Basic Flow:

1. The use case begins when the student wants to search for the subjects in a specific term according to the URL.
2. The application display the main page for inputting the information.

{Enter URL}

3. The student inputs base_url for web scrapping.
4. The student inputs term for searching.

{Go to All Subject Search}

5. The student switch to “All Subject Search” column by clicking “All Subject Search” tab.
6. The student clicks “All Subject Search” button.

{All subject Search}

7. The application print Total Number of Categories/Code Prefix: ALL_SUBJECT_COUNT in console where ALL_SUBJECT_COUNT is the size of the list.
8. If the student clicks the “All Subject Search” button again.
    8.1 Search all the subjects
    8.2 While Searching all the subject.
        8.2.1 After one subject is scraped. Print the SUBJECT is done on the system console.

8.3 Print Total Number of Courses fetched:

TOTAL_NUMBER_OF_COURSES when all subjects scraped.

9. The use case ends.

Alternative Flow:

A1. Invalid URL (404)

At {All Subject Search}, if the entered URL is invalid, e.g. the searching result is 404 not found,

1. Display an appropriate message on the screen to notify the users and not throwing error on the system console.
2. The flow of events is resumed at {Enter URL}

A2. Clicking Search Button

At {Go to All Subject Search}, if the student stays in the main page,

1. The student click the search button
2. The flow of events is resumed at {All Subject Search}

Task 6:

There are two use cases, Find SFQ for Enrolled Courses, and List Instructors' Average SFQ.

Find SFQ for Enrolled Courses: The student can use it to find the SFQ of his/her enrolled courses.

Basic flow:

1. The user case begins when the student wants to find SFQ for his/her enrolled courses.
2. While button Search or All Subject Search haven't been clicked.
   2.1 Keep Find SFQ with my enrolled courses disabled.
3. If Search or All Subject Search Button is clicked.
   3.1 Enable Find SFQ with my enrolled courses.

{Enter URL}

4. The student input SFQ URL for scrapping.
5. The student click Find SFQ with my enrolled courses disabled.

{Scrape Data}

7. The application scrapes data from SFQ URL.
8. Get the enrolled course from the previous search result.

8. Print unadjusted SFQ data (not the data inside brackets) of the enrolled courses on console.

9. If multiple sections are available for a course.

    8.1 Take the average unadjusted SFQ data and print it.

10. The user case ends.

Alternative Flow:

A1. Invalid SFQ URL

At {Scrape Data}, if the entered URL is invalid, e.g. 404 not found,

1. Display an appropriate message on the screen to notify the users and not throwing error on the system console.
2. The flow of events is resumed at {Enter URL}

List Instructors' Average SFQ: the student can check the average SFQ for instructors.

Basic Flow:

1. The user case begins when the student wants to find the average SFQ for instructors.

{Enter URL}

2. The student enters the SFQ URL.
3. The student clicks List Instructor's Average SFQ.

{Scrape Data}

4. The application scrapes data from SFQ URL.
5. If an instructor has taught more than one sections/courses.
    5.1 all unadjusted SFQ score of the sections taught by him/her will be added and divided by number of sections.
6. print all instructors' name and their unadjusted SFQ score on console.
7. The user case ends.

Alternative Flow:

A1. Invalid SFQ URL

At {Scrape Data}, if the entered URL is invalid, e.g. 404 not found,

1. Display an appropriate message on the screen to notify the users and not throwing error on the system console.
2. The flow of events is resumed at {Enter URL}