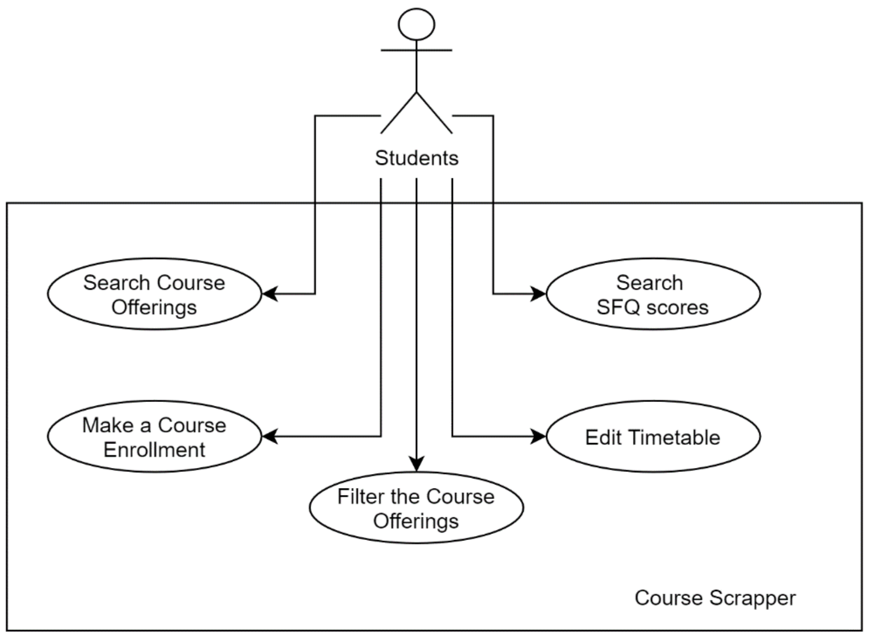


System: Course Scrapper
User: Users

- Search the specific course offerings
- Display the courses information in the console

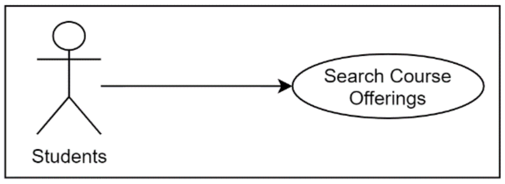
Search all the subjects
Display Scrapped Course information
- Click all subject search
- Obtain the list of all subjects
- Print SUBJECT IS DONE on the system console after one course scrapped
- Update the progress bar by a fraction
- Display # courses fetched
- Update the info in filter list backend

<u>**Use Case: Search Course Offerings**</u>

Brief Description

This use case describes how a student search for courses with or without available slots.

Use-case Diagram



Basic Flow

1. The use case begins when the Student actor chooses to search the courses.
2. The system displays the interface for searching courses.

{Enter Base URL, Terms, Subject}

3. The student indicates website to search on, terms and subject of a course he would like to search.
4. While the student has an activity to perform

    4.1. If the **Search** activity is selected

    {Display course information of specific subject}

    4.1.1. The system retrieves and displays the available course information for the given URL, terms and subject.

    4.2. If the **All Subject Search** activity is selected

    4.2.1 The system retrieves and displays total number of subjects based on the given URL and term

    4.2.2. If the **All Subject Search** activity is selected again

    {Display course information of all subject}

    4.2.2.1. The system notifies the Student after each subject is scraped and the total number of courses when all scraping is done

    4.2.2.2. The system retrieves and displays course information for the given URL and term

    4.2.2.3. The system update information in other tabs (*Filter, List, Backend)*

5. The use case ends.

Alternative Flows

A1: Invalid URL

At {Display course information of specific subject} or {Display course information of all subject} if the entered Base URL is invalid,

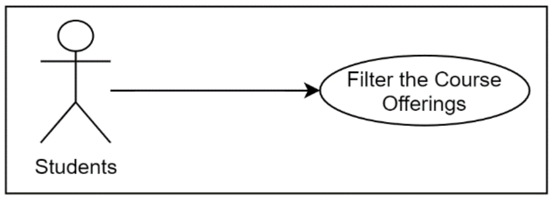
1. The system informs the student that the page is not found (Error 404)
2. The flow of events is resumed at {Enter Base URL, Terms, Subject}

Use Case: Filter the Courses Offerings

Brief Description

This use case describes how a student filter the courses offerings based on certain criteria

Use-case Diagram



Basic Flow

1. The use case begins when the student chooses to filter the courses based on some criteria

{Display the Filters}

2. The system displays all the filter options for filtering the courses to view.
3. The console displays all the filtered courses.

{Select the Filter(s)}

4. While the student has an activity to perform
    - 4.1. If the **Select All** is selected
        - {Begin to Find Courses Fulfilled the Filters}
        - 4.1.1. The text button changes its display to De-Select All.
        - 4.1.2. All the filters on this tab are selected.
        - 4.1.3. Select all sections of courses that fulfill all the boxes.
        - {Display the Result}
        - 4.1.3. Clear the console.
        - 4.1.4. The console displays all the courses fulfilled all the filters.
        - {End to Find Courses Fulfilled the Filters}
    - 4.2. If the **De-Select All** is selected
        - {Begin to Find Courses Fulfilled the Filter}
        - 4.2.1. The text button changes its display to Select All.
        - 4.2.2. All the filters on this tab are de-selected.
        - {Display the Result}
        - 4.2.3. Clear the console.
        - 4.2.4. The console displays all the scrapped courses without filtering.
        - {End to Find Courses Fulfilled the Filter}
    - 4.3. If the **AM / PM** is selected
        - 4.3.1. If option **AM** selected
            - {Begin to Find Courses Fulfilled the Filter(s)}
            - 4.3.1.1. Select all sections of courses that has a slot in AM.
            - 4.3.1.2. Filter the selected courses that fulfill other filters selected, if any.

```
        {Display the Result}
        4.3.1.3. Clear the console and print out the selected courses in the console.
        {End to Find Courses Fulfilled the Filters}
    4.3.1. If option PM selected
        {Begin to Find Courses Fulfilled the Filter(s)}
        4.3.1.1. Select all sections of courses that has a slot in PM
        4.3.1.2. Filter the selected courses that fulfill other filters selected, if any.
        {Display the Result}
        4.3.1.3. Clear the console and print out the selected courses in the console.
        {End to Find Courses Fulfilled the Filter(s)}
    4.3.2. If both AM and PM selected
        {Begin to Find Courses Fulfilled the Filter(s)}
        4.3.2.1. Select all sections of courses that has a slot starts at AM and ends at
        PM or a section has a slot in AM and another slot in PM
        4.3.2.2. Filter the selected courses that fulfill other filters selected, if any.
        {Display the Result}
        4.3.2.3. Clear the console and print out the selected courses in the console.
        {End to Find Courses Fulfilled the Filter(s)}
4.4. If the any days of the week (Monday, Tuesday …… ) is selected
    4.4.1. If only one day selected
        {Begin to Find Courses Fulfilled the Filter(s)}
        4.4.2.1. Select all sections of courses that has a slot on the selected day.
        4.4.2.2. Filter the selected courses that fulfill other filters selected, if any.
        {Display the Result}
        4.4.2.3. Clear the console and print out the selected courses in the console.
        {End to Find Courses Fulfilled the Filter(s)}
    4.4.2. If more than one day selected
        {Begin to Find Courses Fulfilled the Filter(s)}
        4.4.2.1. Select all sections of courses that has slots on the selected days.
        4.4.2.2. Filter the selected courses that fulfill other filters selected, if any.
        {Display the Result}
        4.4.2.3. Clear the console and print out the selected courses in the console.
        {End to Find Courses Fulfilled the Filter(s)}
4.5. If the Common Core is selected
    {Begin to Find Courses Fulfilled the Filter(s)}
    4.5.1. Select all sections of courses that are 4Y CC.
    4.5.2. Filter the selected courses that fulfill other filters selected, if any.
    {Display the Result}
    4.5.3. Clear the console and print out the selected courses in the console.
    {End to Find Courses Fulfilled the Filter(s)}
4.6. If the No-Exclusion is selected
    {Begin to Find Courses Fulfilled the Filter(s)}
    4.6.1. Select all sections of courses that does not define exclusion.
    4.6.2. Filter the selected courses that fulfill other filters selected, if any.
```

{Display the Result}
4.6.3. Clear the console and print out the selected courses in the console.
{End to Find Courses Fulfilled the Filter(s)}

4.7. If **With Labs or Tutorial** is selected
{Begin to Find Courses Fulfilled the Filter(s)}
4.7.1. Select all sections of courses that has labs or tutorials.
4.7.2. Filter the selected courses that fulfill other filters selected, if any.
{Display the Result}
4.7.3. Clear the console and print out the selected courses in the console.
{End to Find Courses Fulfilled the Filter(s)}

5. Update the lists of courses display in List tab based on the selected courses.
6. The use case ends

Alternative Flow

A1: First time running

At {Display the Filters} if it is first time to open the tab,

1. The system will print out nothing in the console.
2. The flow of events is resumed at {Select the Filter(s)}

A2: Second or above time running

At {Display the Filters} if it is not the first time to open the tab,

1. The system will remain the previous console result without clearing.
2. Keep all the filters checked intact
3. The flow of events is resumed at {Select the Filter(s)}

A3: No fulfilled result(s)

At {Display the Result} if none of the courses fulfilled the filters

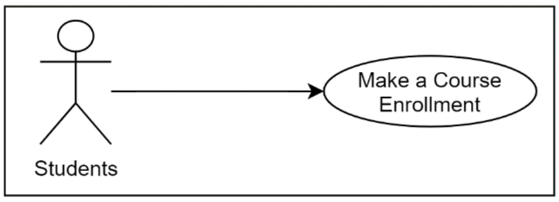
1. Clear the console.
2. The flow of events is resumed at {End to Find Courses Fulfilled the Filter(s)}

<u>Use Case: Make a Course Enrollment</u>

Brief Description

This use case describes how a student enroll the courses.

Use-case Diagram



Basic Flow

1. The use case begins when the student chooses to enroll the courses after filtering.

{Display the Courses List}

2. The system displays all the courses filtered with the basic course information.

{Status Change}

3. While the student has an activity to perform.
   - 3.1. If the status checkbox in **Enroll** is selected.
     - 3.1.1. The console clear.
     - 3.1.2. Print out all the filtered course in the console.

     {Display the enrolled course(s)}
     - 3.1.3. Display the statement “The following sections are enrolled: ” on the console, followed by the list of enrolled sections.
   - 3.2. If the status checkbox in **Enroll** is de-selected.
     - 3.2.1. The console clear.
     - 3.2.2. Print out all the filtered course in the console.

     {Display the enrolled course(s)}
     - 3.2.3. Display the statement “The following sections are enrolled: ” on the console, followed by the list of enrolled sections.
4. The use case ends

Alternative Flow

A1: Failed Filter implementation

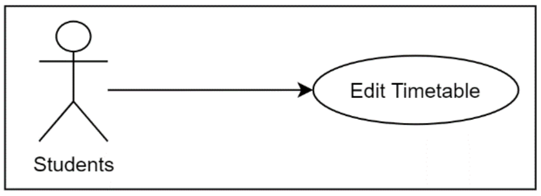
At {Display the Courses List} if the Filtering cannot be correctly implemented,

1. Show all the results scrapped.
2. The flow of events is resumed at {Status Change}

<u>Use Case: Edit the timetable</u>

Brief Description

Use-case Diagram



Basic Flow

1. The use case begins when the student actor performs search or all subject search.

{Enrolment}

2. The system accesses the updated course sections added in enrolment when the student actor adds courses into enrollment by clicking the checkbox in the list

{Edit Timetable}

3. The system adds the enrolled courses on the timetable in the form of blocks, with blocks at the position in accordance to the section time, relevant course code as well as section code on the blocks and the blocks representing the same course sharing the same background color.

4. If there are slots of sections overlapped,

    4.1. The overlapped section of the block would be represented by another color different to colors of the both blocks.

5. The system displays the timetable to the user

6. The use case ends.

Alternative Flow

A1: Failed enrollment implementation

At {Enrolment} if the Enrolment cannot be correctly implemented in List,

1. If there are less than five courses in the timetable,

    1.1. If there are more than or equal to five sections in the scrapped data,

        1.1 Enroll the first 5 sections of the scrapped data.

    1.2. If there are less than five sections in the scrapped data,

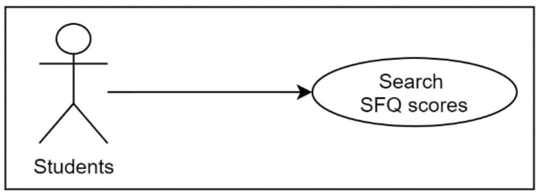
        1.2.1. Enroll to all sections of the scrapped data

2. The flow of events is resumed at {Edit Timetable}.

Use Case: Search SFQ scores

Brief Description

Use-case Diagram



Basic Flow

1. The use case begins when Search or All Subject Search is clicked.

{Scrape score}

2. Scrape SFQ data from SFQ URL.

{Process data}

3. If Find SFQ with my enrolled courses is clicked,

{Begin Find SFQ with enrolled courses}
3.1 The system accesses the course code and section code in the enrolment.
3.2 Calculate the average score of the courses in enrolment among sections.
3.3 Print out the average score on the console.
{End Find SFQ with enrolled courses}

4. If List instructors' average SFQ is clicked,
    4.1 Scrape SFQ data from SFQ URL.
    4.2 Match instructor and sections taught by the relevant instructor, together with SFQ score for every section.
    4.3 Calculate the average score of the sections taught by each instructor, and assign the average score to the relevant instructor.
    4.4 Print out all the instructors name, aligned with its SFQ score, on the console.
5. The use case ends.

Alternative Flow

A1: Failed enrollment implementation
At {Begin Find SFQ with enrolled courses} if the Enrolment cannot be correctly implemented in List,
1. If there are less than five courses in the timetable,
    1.1. If there are more than or equal to five sections in the scrapped data,
        1.1 Enroll the first 5 sections of the scrapped data.
    1.2. If there are less than five sections in the scrapped data,
        1.2.1. Enroll to all sections of the scrapped data
2. The flow of events is resumed at {Scrape score}.

A2: Invalid URL
At {Scrape score} if the URL is invalid,
1. The system informs the students that the URL is invalid.
2. The flow of events is resumed at {Process data}.