Name: Dao Kha Tuan (s3877347)

Shape

Description automatically generated with medium confidence

I DECLARE THAT IN SUBMITTING ALL WORK FOR THIS ASSESSMENT I HAVE READ, UNDERSTOOD AND AGREE TO THE CONTENT AND EXPECTATIONS OF THE ASSESSMENT DETAILS.

Assignment 1  
cosc2440 – further programming

Technical report

Table of Contents

[I. Overview 2](#_Toc99311888)

[II. Flow of interaction 2](#_Toc99311889)

[1. Data Menu 2](#_Toc99311890)

[2. Main Menu 3](#_Toc99311891)

[3. Enrollment Menu 3](#_Toc99311892)

[4. Get Report Menu 3](#_Toc99311893)

[5. Update Enrollment Menu 4](#_Toc99311894)

[III. Table of methods 5](#_Toc99311895)

[IV. Diagrams 7](#_Toc99311896)

[1. Class diagram 7](#_Toc99311897)

[2. Use case diagram 8](#_Toc99311898)

[V. Appendix 8](#_Toc99311899)

# Overview:

This system is a Java console application that allows academic assistants to add, delete, update, and view all student enrollments of an educational institution. It also contains functionalities to print out all courses of a student in one semester, all students of a course in one semester, and all courses offered in one semester as well as save these reports to CSV files.

# Flow of interaction:

The system’s user interface consists of five different menus: Data Menu, Main Menu, Enrollment Menu, Get Report Menu and Update Enrollment Menu. Users will be prompted to select a CRUD operation on the student enrollments by inputting a number corresponding to the option from each menu. If the users select an option that is not on the menu, the menu will reappear for them to select. After users have selected an option, the program will ask users to fulfill the relevant input fields to complete the task, such as student ID, course ID, and semester. When the user misses an input field that is essential for the task to be completed, the system will prompt the user to fulfill the input field until all input fields are completed. Users can return to the Main Menu or stop the system by selecting "Back" or "Quit" from each menu.

# **Data Menu:**

When the user initially launches the program at Main.java, which is located at folder “src”, the system will display a Data Menu (see Appendix Figure 1: Data Menu), asking the user which method they want to use to populate the system’s data.

The user has two options for populating the data: using the user’s sample CSV file or using the system’s default CSV file. If the user chooses to use the sample CSV file, the system will prompt them to provide the file’s name, which must be in the “src/files/data” directory. The user does not need to input the file name with the file extension “.csv”, and if the file is not available in the directory “src/files/data”, the system will notify the user and re – display the Data Menu.

When the user enters the correct file name in the “src/files/data” directory, the system verifies to see if the CSV file is not empty, contains no duplicate or empty enrollment records, has enough attribute fields in each record and valid semester format before populating the data. If the CSV file is invalid, the system will display the Data Menu again to prompt the user to select a legitimate CSV file or use the system’s default CSV file to populate the data. The user will be redirected to the Main Menu once the data –loading process is completed.

# **Main Menu:**

When the data is populated successfully, the user will be redirected to Main Menu (see Appendix Figure 2: Main Menu).

The user can select between 2 options in the Main Menu: “Manage enrollments” and “Get report”. The Enrollment Menu will display if the user selects the "Manage enrollments" option. A Get Report Menu will display if the user selects the “Get Report” option. Otherwise, the user has the option of exiting the program.

# **Enrollment Menu:**

In Enrollment Menu (see Appendix Figure 3: Enrollment Menu), the user can select between 3 options to perform CRUD operation on student enrollments: view, add and delete enrollments.

When the user chooses to view all enrollments, the system will display the list of all enrollments in a table format (see Appendix Figure 4: List of all enrollments).

If the user chooses to add or delete an enrolment, the system will ask the user to provide the student ID, course ID, and semester of the enrollment user wants to add or remove from the enrollment list. When the user enters an invalid semester, adds a new enrollment that already exists, or deletes an enrollment that is no longer available, the system displays a warning and returns the user to the Enrollment Menu. If the operation is accomplished, the system will display a success message and redirect the user back to Enrollment Menu.

# **Get Report Menu:**

The system provides 3 options in Get Report Menu (see Appendix Figure 5: Get Report Menu): print all courses of a student in one semester, print all students of a course in one semester, and print all courses offered in one semester. The user can also save these reports to CSV files, which can be accessed in the “src/files/courses” directory for course lists and the “src/files/students” directory for student lists.

When user selects to view all courses of a student in one semester, the system will prompt user to input the student ID and semester required to get the report. If the course list is empty or the input fields are not valid, the system will print a warning message and return user back to Get Report Menu. Otherwise, a list of courses will be displayed to user (see Appendix Figure 6: An example of course list).

The system also asks if the user decides to update the enrollments of a student for one semester, which is the course list displayed above. If the user demands to add or remove enrollments of a student in one semester, they will be redirected to the Update Enrollment Menu. Otherwise, the program will ask the user whether to save the course list to a CSV file before returning to the Get Report Menu.

When the user selects the option of viewing all students of a course in one semester, the functionality is quite similar as the functionality of getting a student’s list of courses for one semester. However, the system will prompt the user to enter the course ID instead of student ID, and there is no Update Enrollment Menu to add or delete students from the list. The system also displays the list of students in table format (see Appendix Figure 7: An example of student list) and ask the user whether to save it to a CSV file.

The functionality will be the same as the two options above if the user wants to view all courses available in one semester. However, the system will require the user to provide input for one input field only, which is the semester they wish to view all courses, and they will also be able to save the report to a CSV file.

# **Update Enrollment Menu:**

If the user requests to update the enrollments of a student for one semester, the Update Enrollment Menu (see Appendix Figure 8: An example of Update Enrollment Menu) will appear. It will give the user two options: to add or remove courses from the course list of a student for one semester.

If the user wants to add or remove a course from the course list, the system will ask for the ID of the course they wish to add or remove. The system will display a warning message and return the user to the Update Enrollment Menu if the user adds a course that already exists or removes a course that is not present in the list. After the user has updated the course list and selected to return to Get Report Menu, the system will display the modified course list and ask the user whether they want to save it to a CSV file before returning to Get Report Menu.

# Table of methods:

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **List of parameters** | **Description** |
| populateData | Boolean | None | This method is used to populate the data of the system. |
| getStudentById | Student | * studentID: String | This method is used to get a Student object from the system by student ID. |
| getCourseById | Course | * courseID: String | This method is used to get a Course object from the system by course ID. |
| add | Boolean | * studentID: String * courseID: String * semester: String | This method is used to add a new enrollment to the system with student ID, course ID and semester. |
| delete | Boolean | * studentID: String * courseID: String * semester: String | This method is used to delete an enrollment from the system with student ID, course ID and semester. |
| getOne | Enrollment | * studentID: String * courseID: String * semester: String | This method is used to get an Enrollment object from the system with student ID, course ID and semester. |
| getAll | ArrayList<Enrollment> | None | This method is used to get an ArrayList of all enrollments from the system. |

# Diagrams:

# Class diagram: (the PDF format of class diagram is included in zip file)

Diagram

Description automatically generated

# Use case diagram: (the PDF format of use case diagram is included in zip file)

Diagram

Description automatically generated

# Appendix:

1. Link to my GitHub repository of the project: <https://github.com/TuanDao2002/s3877347_Assignment1_COSC2440>.
2. How to run the program: run the Main.java file located at the folder named “src”.
3. Unit tests are located the directory “src/test”.

*Figure 1: Data Menu*

Text

Description automatically generated

*Figure 2: Main Menu*

Text

Description automatically generated

*Figure 3: Enrollment Menu*

Text

Description automatically generated

*Figure 4: List of all enrollments*

Text

Description automatically generated

*Figure 5: Get Report Menu*

Text

Description automatically generated

*Figure 6: An example of course list*

Text

Description automatically generated

*Figure 7: An example of student list*

Text

Description automatically generated

*Figure 8: An example of Update Enrollment Menu*

Text

Description automatically generated