RMIT UNIVERSITY

Student Enrollment Management System

Assignment 1

Author:

S3836387 - Ngô Quang Khải

SYSTEM DESCRIPTION

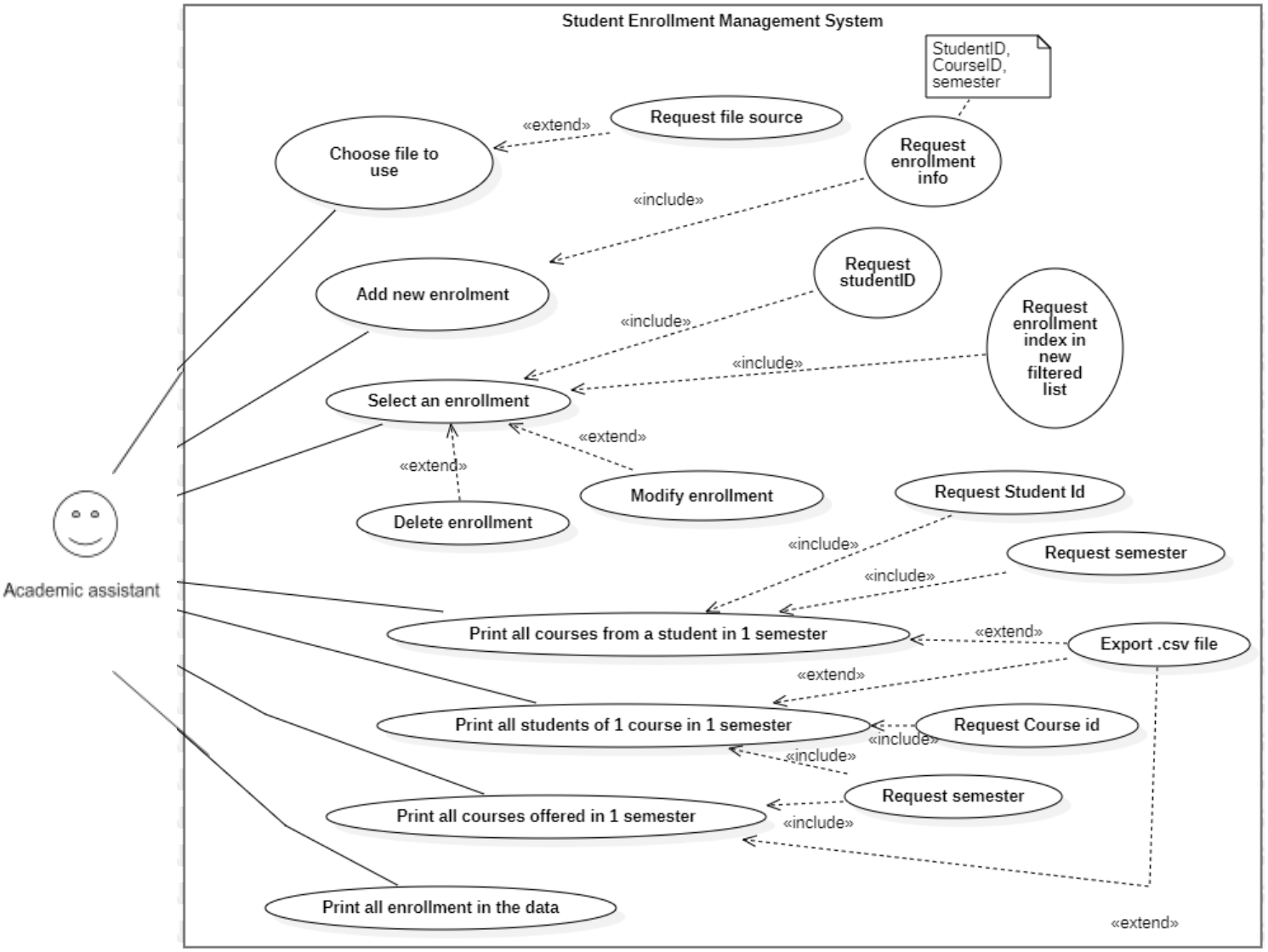
The system will use the class Console to take in all the inputs and validate the input format. It will use regex to check the input format.The student id must have a beginning character “s” or “S” followed by 7 digit numbers. The course id must have 3-4 letters followed by 4 digits. The semester can not be before 2001 (before university is built) and must end with “A”, “B” or “C” case sensitive.

Firstly,the system will ask the user to provide a data file or use default. If the user chooses their own file, the system will use this file as the main file to extract and populate the information with. Whereas, if the user chooses the latter, the system will use the data from three different .csv files (student.csv, course.csv, default.csv) to populate the data. The student.csv and course.csv will act as reference files for comparing with user input. This is to ensure that the user does not input the wrong data for student and course. After selecting the data file, the user will have 6 options: Add enrollment, Select enrollment to modify/ delete, 4 options to print data. The first option will let the user add new data into the existing list and update it into the csv file. The system will not accept student id or course id that is not in the database (from sample data from default.csv or student.csv, course.csv).

The second option, the user will select an enrollment to: modify or delete. The system will ask the user to enter a student id and a semester. Then using that data, the system creates and prints out a sub-list from the main enrollments list that satisfies the criteria. To select an enrollment, the user type in the enrollment index provided in the printed list. This will prevent the user from inputting the wrong data and is more user friendly. The system will then make a copy of that selected object to check for duplication after the user modified the object. If there is duplication after the user modified the new object, the system will not update any changes. When the user selects “UPDATE”, the system will give the user choices to choose which field the user wants to change. After modifying the field, the user must choose “DONE” to save the changes into the system data and default.csv file. On the other hand, if the user chooses “DELETE”, the record will be deleted from the database and default.csv file.

Finally, the 4 print options will print specific data by user input: Print all courses from a student in 1 semester, print all students of 1 course in 1 semester, print all courses offered in 1 semester and print all enrolments. All four options will give the user the option to export a .csv file of the printed data (the file is exported in the root file).

USE CASE DIAGRAM



CLASS DIAGRAM

Console

This class is for taking and checking input from the user. All the methods in this class will take in a String value called “prompt”, the methods will use this to print out the questions for the user to input. All the methods will return a valid format value.

1. **validateInt**(String prompt , int min, int max): This method is used for taking the input for choices and checking it. It will ensure that the input will be in between min and max value (both inclusive). If the value is invalid the method will repeat and ask the user again.
2. **validateStudentId**(String prompt): This method is used for taking input and checking if the student id has the right format. The method used regex to check and will repeat if the input is invalid.
3. **validateCourseId** (String prompt): This method is used for taking input and checking if the course id has the right format. The method also uses regex to check and will repeat until a valid value is entered.
4. **validateSem**(String prompt): This method is used taking input and check input semester is valid. It will check if the year is after 2001 (after the university is built) and check if the last letter is A, B or C. The method will also regex and will repeat until a valid value is entered.

Student Enrolment Manager

This class implements singleton design pattern to ensure that all data is centralized and ensures data integrity.

1. **createFile** (String filePath): return true if a new file is created, else false. This method is used to create new .csv files for the print methods.
2. **writeFile**(List<StudentEnrolment> list, String filePath): This method is used to re-write new data into the existing file. This method for updating the .csv file to synchronize with the system data.
3. **checkFile** (String filePath): return true if file exists, else false. This method is used to check whether the file that the user provided existed. This is used with the populateData methods to ensure that the no Exception is throw.
4. **populateData**(): This method will populate data from three file student.csv into StudentList, course.csv into CourseList and default.csv into StudentEnrolmentList.
5. **populateData**(String filePath): This method will populate data from the file provided by the user. This method will extract data from the file and add it into StudentList, CourseList and StudentEnrolmentList.
6. **getStudent**(String id): This method will take in student id and return a Student object. It will do a search in the StudentList and return a Student object with the matching id, otherwise it will return null. This method is also used to check and prevent duplication in StudentList.
7. **getCourse**(String id): This method will take in course id and return a Course object. It will do a search in the CourseList and return a Course object with the

matching id, otherwise it will return null. Other than getting data from the CourseList, this method is also used to check and prevent duplication in CourseList.

8. **add**(Student student, Course course, String semester): This method will add a new StudentEnrolment object into the StudentEnrolmentList and return true if success. Moreover, it also checks whether the object is a duplication of any object in the list, if so it will not add that object and return a false.
9. **update**(StudentEnrolment newObject,StudentEnrolment oldObject):Using the “newObejct”, the method will search for any duplication, if found, the method will print error message and do nothing. If no duplication is found, this method will change the “oldObject " in the StudenEnrollmentList and set it to the “newObject”.
10. **delete**(StudentEnrolment obj): This method will find the object in the StudentEnrolmentList and delete it. This method is also called after we select the object in the StudentEnrolmentList, we can use the .getIndexOf() to quickly find the object and remove it.
11. **getOne**(String studentId, String courseId, String semester): This method will use the information provided in the parameter to find and return the correct StudentEnrollment object in the StudentEnrolmentList. It will return null if it can not find the object
12. **getAll**(): This method will print out all the StudentEnrolment objects inside the StudentEnrolmentList.
13. **getStudentEnrolmentByStudentIdNSem**(String studentId, String semester): This method will return a new list of StudentEnrolment that show all courses from a student in 1 semester. It will return an empty list if no data is found.
14. **getStudentsEnrolmentByCourseNSem**(String course, String sem): This method will return a new list of StudentEnrolment that show all students of 1 course in 1 semester. Return an empty list if no data is found.
15. **getStudentsEnrolmentBySem**(String semester): This method will return a new list of StudentEnrolment that show all courses offered in 1 semester. Return an empty list if no data is found.