Vision:

We designed a simulation that deals with the rules and methods that a student must follow during the course registration process.

Detailed Description:

This is a student course registration controller simulation for CSE Department of Marmara University. Students, advisors and the courses are created. (Courses created according to the curriculum and prerequisite tree.) After initializing students and advisors, every student assigned to an advisor. After all, registration controller randomly marks each student's courses which before the simulated semester as passed or failed with probability declared in input.json. Reason to fail some courses is to simulate the prerequisite and required credit conditions. Then registration controller lists the offered courses for students. Offered courses includes previously failed courses and the courses of the current semester. After that those lists are sent to advisors to either approve or disapprove the schedule.

Advisors approve or disapprove conditions:

- There should not be more than one hour of collision in the weekly schedule of student.
- There must be empty seats in the section.
- The prerequisite of the course must be passed.
- The required credit amount for the course must be completed.

If a course disapproved student have no chance to register to that course in the simulated semester, so they move on with the next course in their offered course list.

When the registration finished the registration controller shows every student's transcript and the statistics about the overall registration process:

- Number of students failed to register because of a collision
- Number of students failed to register because of the course section being full
- Number of students failed to register because of the required credits not provided

Simulation ends. Statistics can be reviewed, and necessary inferences and improvements can be made.

Use Cases: Course Registration

1) System lists the semester's courses to student.

2) Student adds this course to his/her new course list.

3) An advisor is assigned to the student by the system.

4) Student sends his/her course list to the advisor.

5) Advisor checks collision status of courses, quotas of courses, if the course has a prerequisite course, and the student's total passed credits.

6) If all the conditions listed above are true, advisor approves this course/courses.

7) These approved courses are added to the student's schedule.

8) The system shows student's transcript.

9) According to the past transcript, system determines and shows the GPA of student.

10) The system keeps the statistics of the students who could not be registered for the courses and shows it after the process.

Alternative Flow: Prerequisite course is not passed

5a) Student cannot take this course and moves other courses on the list.

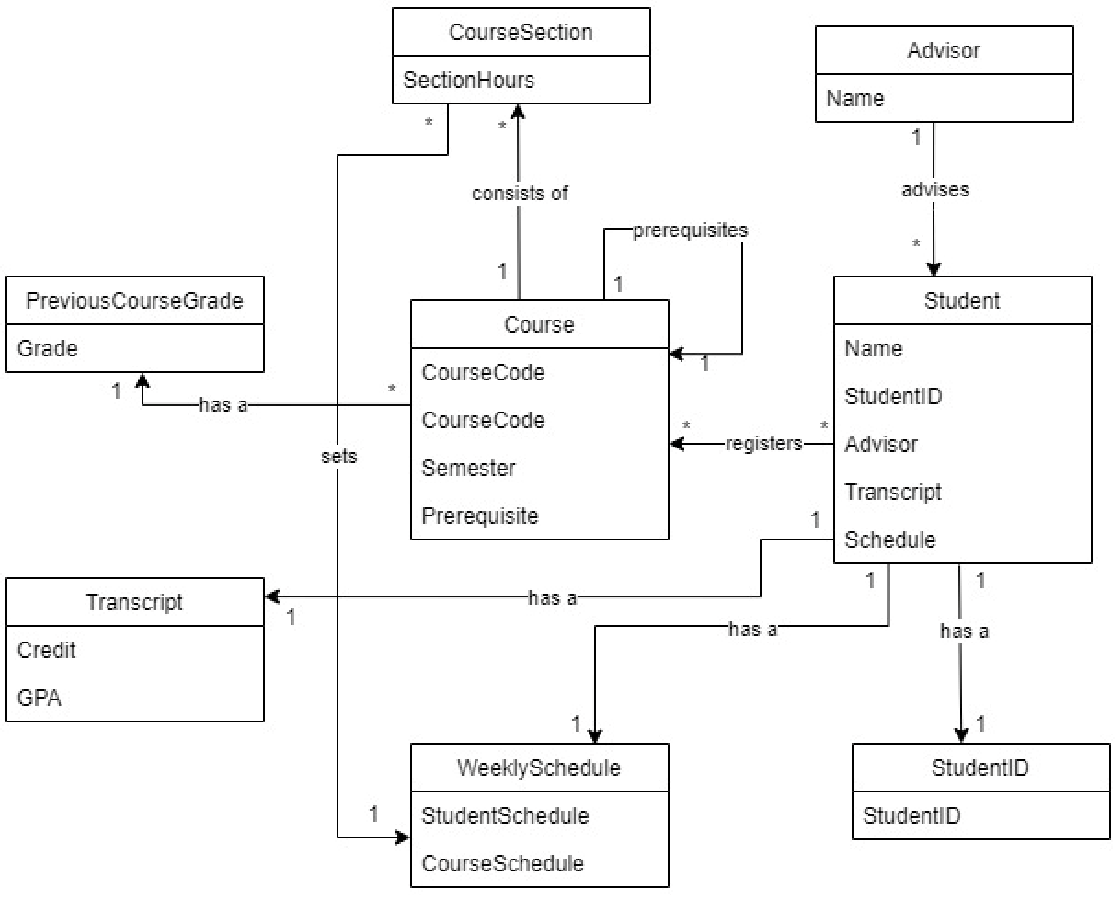
Alternative Flow: Course section is full

5a) Student checks to see if there is an available course section.

5b) Student cannot find any available section.

5c) Student cannot take this course and moves other courses on the list.

Domain Class Diagram:



REQUIREMENT SPECIFICATIONS

2.1 FUNCTIONAL REQUIREMENTS

2.1.1 MUST HAVE REQUIREMENTS

- The courses should have included prerequisite course and further business logic and regulation information.
- System will simulate random students with all 8 semesters with equal probability.
- The randomly created student will have random course success rate.
- System should implement prerequisite tree starting from 3rd semester.
- The students and their transcripts should be stored is JSON format.
- Each student will have their own Json file. Course registration process will be printed to student's Json file.
- Program will be controlled via command prompt.
- The student will register the elective courses from a random course pool and each step will be printed to command line.
- With a certain probability an elective course's quota may be full.
- The program will include a batch process to simulate a student's whole courses starting from first to last semester.

2.1.2 OPTIONAL REQUIREMENTS

- There can be a course adviser program for better success rate.

2.2 NON-FUNCTIONAL REQUIREMENTS

- Security
- Portability
- Capacity
- Disaster Recovery

Stakeholders:

Murat Can Ganiz (Professor)

Ahmet Can Bağırgan

Emir Said Haliloğlu