REQUIREMENT ANALYSIS DOCUMENT

Description of the project:

Our aim in this project is registration and control of the courses in online education. Students in the system will be able to register to all the courses they have in their own semesters if they meet the requirements. In every semester there will be 70 students. At the beginning, there will be chosen the current semester that is spring or fall. The system will take inputs from a JSON file. In this file there will be a curriculum . along with a pool of University Elective (UE), Technical Elective (TE), Non-Technical Elective (NTE), and Faculty Technical Electives (FTE) courses, information of all courses, the prerequisite tree and also the semester that we will choose fall or spring. Students that will be randomly created will fail from some courses and they will complete their semesters according to the prerequisite tree of the university.There will be a student that will be controlled by an admin that uses console. While other students are randomly created, the student there will be controlled and will also register to courses. Courses will have sections that will be different hours. If there is a collision of two hours between courses, the advisor will not approve. While registering the courses, there will be some criterias. If a student's registration doesn't meet the requirements, the advisor will deny the course request of the student. For example if a student fails a prerequisite lesson, the advisor will not approve the selected course's registration. When the student finishes the semester, the json file created for him includes his notes and the transcript of the courses he/she took. In this transcript, there is also information about why the student could not take the courses. At the end of the program there will be an output JSON file that includes department output. In the department output we should be able to see general statistics about the problems for the given semester. For instance, the system will count the number of students who failed CSE1242 and write this to the JSON file.

Functional Requirements:

The system supports three types of actors: Students, Lecturers, System.

Students:

- ??Students will have randomly created student IDs.??
- Students can register to a course section of a specific course that is in the student's semester course pool.
- Students can not take a course if he/she failed the prerequisite course.
- The students will be graded randomly for the chosen courses.
- Students will register to the ?elective? courses randomly from a pool of courses.
- Students will have transcripts which contain their failed, completed and active courses and grades of the respective courses.

Lecturers:

- Advisors from lecturers may approve or deny student's registration requests.
- Lecturers may have one or more lessons.
- Lecturers may be advisors, if so,they will be advising a certain number of students.

System:

- The system will maintain the lecturers, students, courses, course sections, registrations and connections between them.
- Students will have their own JSON file (150129649.json),which is created by the system, that contains their transcript and summary of the problems about registration.

Nonfunctional Requirements:

- We need to show each step on the console one by one.
- There will be a json file for each student.
- There will be no databases.
- We need to use java an Object Oriented Manner
- The program will be running on the IDE console
- The program should have an infrastructure that will not collapse due to overload between the time period of course selection.
- The response time of the system should be acceptable.

Use Case: Registration to a Course

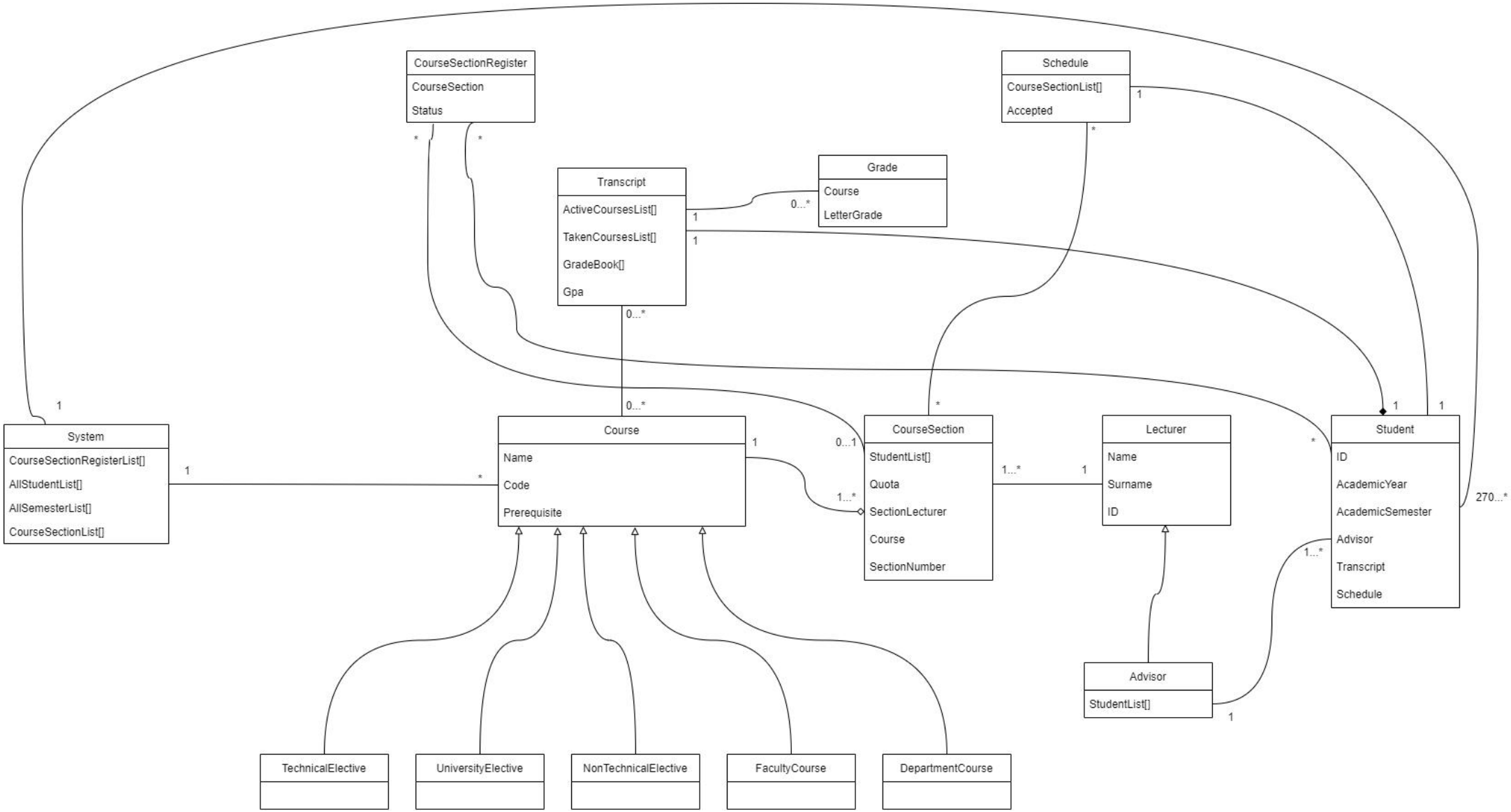
Actors: Student, System, Advisor

1. The system displays the course sections that the student can choose.
2. Student selects the course sections that he/she wants to register for in the system.
3. The system sends the registration request of the student to his/her advisor.
4. The advisor confirms the request of the student.
5. The system sends a confirmation message to the student.

Alternatives:

4a. At step 4, the advisor denies the inappropriate registration request.

Allow the student to re-select the courses he/she wants to register for.


CourseSectionRegister
CourseSection
Status
Schedule
CourseSectionList[]
Accepted
Transcript
ActiveCoursesList[]
TakenCoursesList[]
GradeBook[]
Gpa
Grade
Course
LetterGrade
System
CourseSectionRegisterList[]
AllStudentList[]
AllSemesterList[]
CourseSectionList[]
Course
Name
Code
Prerequisite
CourseSection
StudentList[]
Quota
SectionLecturer
Course
SectionNumber
Lecturer
Name
Surname
ID
Student
ID
AcademicYear
AcademicSemester
Advisor
Transcript
Schedule
Advisor
StudentList[]
TechnicalElective
UniversityElective
NonTechnicalElective
FacultyCourse
DepartmentCourse
1
*
0...*
0...1
1...*
270...*