REQUIREMENT ANALYSIS DOCUMENT

1.Introduction

1.1 Vision

In the registration process of the simulation, system will check whether student can take available courses in current semester. If there is a situation that prevents course taking, the system handles this situation as an error and adds to transcript file of student. For instance, if the capacity of elective course is full, system does not approve the registration of this course. If students can meet the conditions necessary for the course, they may register for the course. In the end of the semester the grades of courses will be generated and GPA will be created. After finishing the process, system provides transcript files of students. Transcript files includes summary of registration process. This process will continue until end of 8 semesters.

1.2 Scope

The purpose of the second iteration of the project is tracking registration process of the students according to department regulations as a simulation. In the simulation, user can observe how distance education period will be handled. Also, project can be used to test if the system works properly.

Our project's aims are as follows:

- To develop a method that will expedite students' course registration and reduce the challenges they face during registration.
- To create a better system that allows students to enroll in university courses fairly.
- To develop a system that can track all the events that occur during the course selection process. University administrators are required to be directly aware of any problems that occurred during the course registration

1.3 System Constraints

- It is necessary to have devices that can execute the Python 3.
- JSON format is required for student transcripts.
- Because of program will work offline, it will not be able to synchronize or run-on different systems, therefore only one user will be able to use it at a time.

2. Functional Requirements

- System requires an input file to start simulation.
- System creates instructors.
- System creates courses.
- System assigns courses to instructors.
- System creates 70 students per semester.
- System provides available courses of the current semester.
- Students register courses according to regulations.
- System creates transcript files for each student.

3. Nonfunctional Requirements

- System creates a Json file for each student.
- System do not use databases
- System should be acceptable for the response time
- System must have an infrastructure that will not crash due to overload during the simulation from start to finish

4.Glossary of Terms:

Course: A lecture that can be taken from the schedule covered in the curriculum.

Student: A person who has a system of registration to take classes properly.

Instructor: The person who will be a teacher in the registration system for per courses

Advisor: The person who will approve and enroll for the students who have taken courses.

Prerequisite: A system that determines all prerequisite courses from Json file.

Transcript: It is the output that shows the history of the courses taken by the student and is created separately for each student.

Id: A unique number assigned to each student by the registration system to recognize them from received student Json file.

Technical alert course: An elective lecture that can be taken if the total credit earned exceeds 155, however it cannot be taken in semesters 2 or 3, and the schedules do not conflict.

Non-technical elective course: An elective lecture that can be taken if the quota is not filled, no credit is required, and the scheduling does not conflict.

5.Stakeholders

Customer:

- Murat Can Ganiz

Developers:

- Muhammed Eren Atala
- Hüseyin Kerem Mican
- Ahmet Faruk Güzel
- Hakan Kenar
- Ahsen Yağmur Kahyaoğlu
- Emre Demir

6.Use Cases

6.1 Use Case - Course Registration

Actor: Student

1) System starts registration for the students in current semester.

2) System selects students failed courses one by one.

3) System selects non-taken courses of students one by one.

4) System checks whether student meets necessary conditions to take the course.

5) After the courses added to the course basket, system sends basket to advisor.

6) Approved courses are going to be added to active course list of the student by the system.

7) Courses that could not passed the approver, gets added to non-taken course list of the student by system.

8) The active courses of the student and errors that explains why student could not take the courses will be written in the transcript.

Alternative Flow: System Control

4a. System checks the capacity of the selected courses.

4b. System selects random elective course for students.

6.2 Use Case - Course Approval

Actors: Advisor, System

1) System checks whether there is a prerequisite course for the relevant course.
2) System checks for the past courses if the elective courses have already been taken.
3) Advisor checks whether there are more than two technical courses in the student's basket.
4) Advisor checks for the student's graduation term is Fall season or not. If it is Fall students are unable to take FTE courses.
5) Advisor controls the students' credits for the graduation project.
6) Advisor controls the students' credits for the technical elective courses.
7) Advisor checks the collisions of the courses.
8) Advisor approves courses in student's course basket.

Alternative Flow: Prerequisite course of the related course has not passed by the student.

1a. If student has not taken or passed the prerequisite course of the related course, system will not allow him/her to take the course.

Alternative Flow: Student tries to take more than two technical courses.

3a. If there are more than two technical courses in the student's course basket, advisor will not allow student to take more than two courses. Advisor reduces the number of taken courses to two.

Alternative Flow: Student is not in the graduation term in the Fall season and tries to take FTE course.

4a. If the problem occurs student cannot take the FTE courses.

Alternative Flow: Student does not have enough credits to take graduation project.

5a. If student's completed credits are less than 165 credits, student cannot take the graduation project.

Alternative Flow: Student does not have enough credits to take technical elective courses.

6a. If student's completed credits are less than 155 credits, student will be unable to take the course.

Alternative Flow: Collision occurs during the registration process.

7a. The course which has higher priority will be registered to student's schedule.

Alternative Flow: Approve Student Basket

8a. If course basket passes all approvement process, advisor approves student's course basket.

6.3 Use Case – Simulation Process

Actor: System

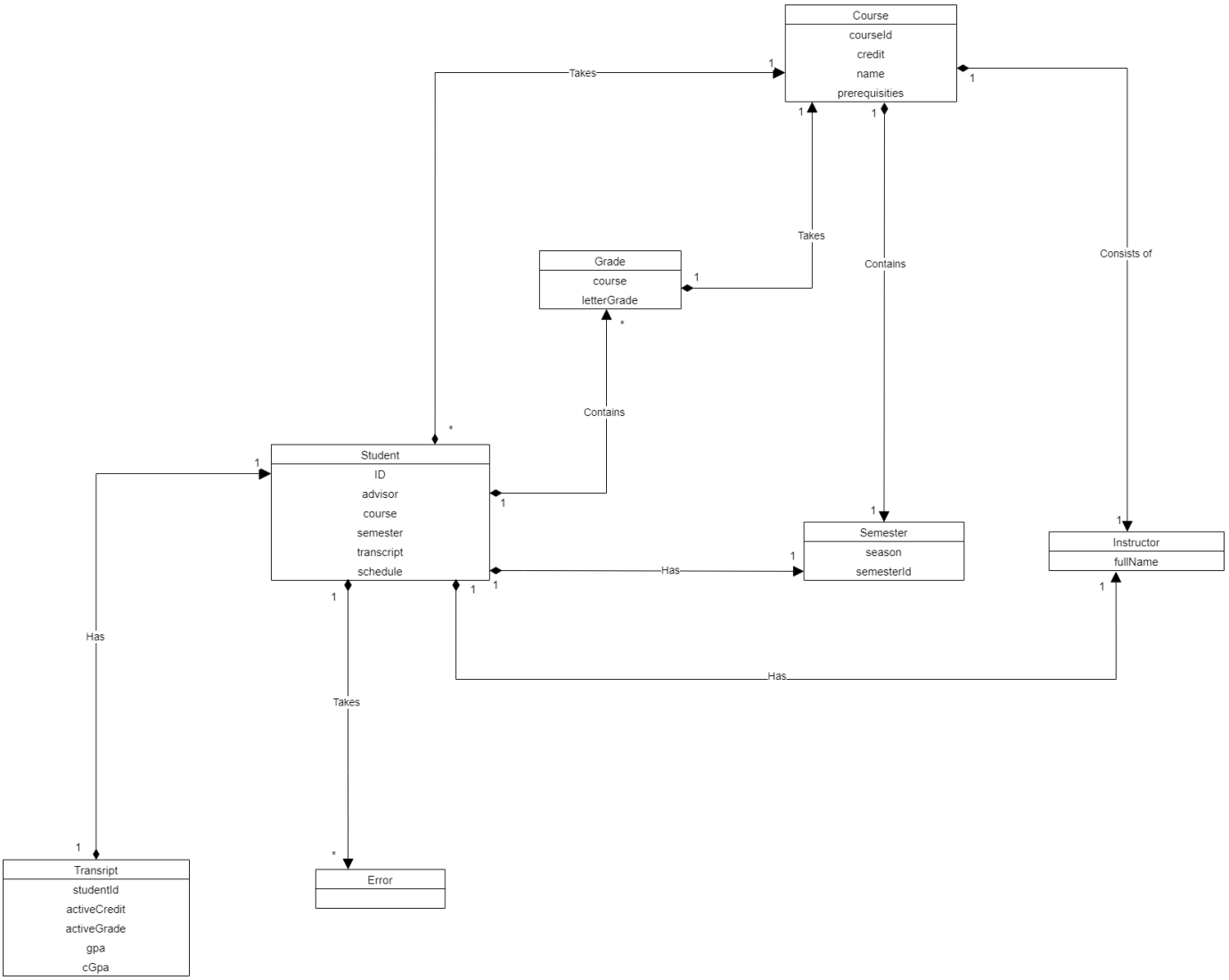
1) System reads instructor information.
2) System reads course information.
3) System reads prerequisite tree information.
4) System assigns an advisor to each student.
5) System reads season and generation parameters.
6) System initializes students.
7) System simulates semester.
8) System begins registration process.
9) System assigns random grades for students.
10) System assigns students from current semester to next semester.
11) System writes transcripts.
12) System repeats 6 to 10 steps until all semesters simulated.
13) System prepares department output which includes errors occurred during registration.

Alternative Flow: Initializing students

6a. Student will be created in the beginning of simulation. Every “Fall” semester system creates new students. Students will get into the simulation for each semester. System repeats same actions until simulation finishes.

6b. If given parameter is “Spring” students’ information will be read from transcript files.

7.Domain Model



8.System Sequence Diagram

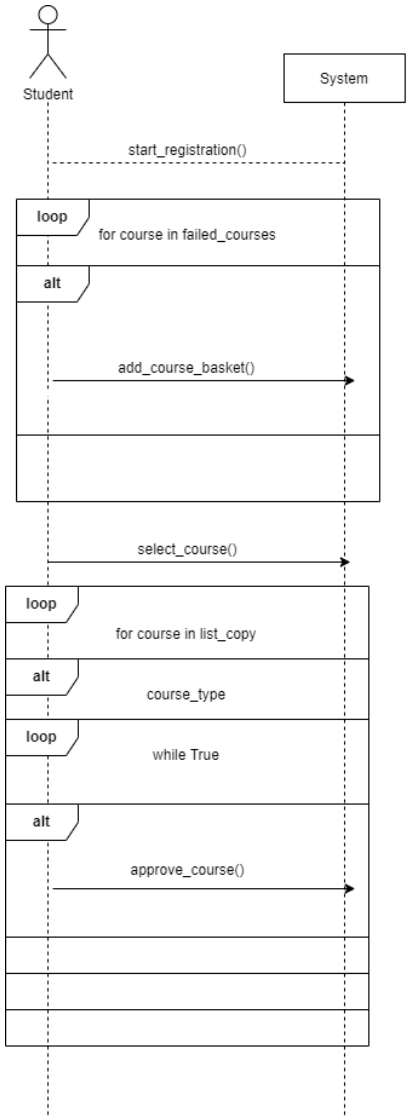


Figure 1: System Sequence Diagram of Use Case 6.1

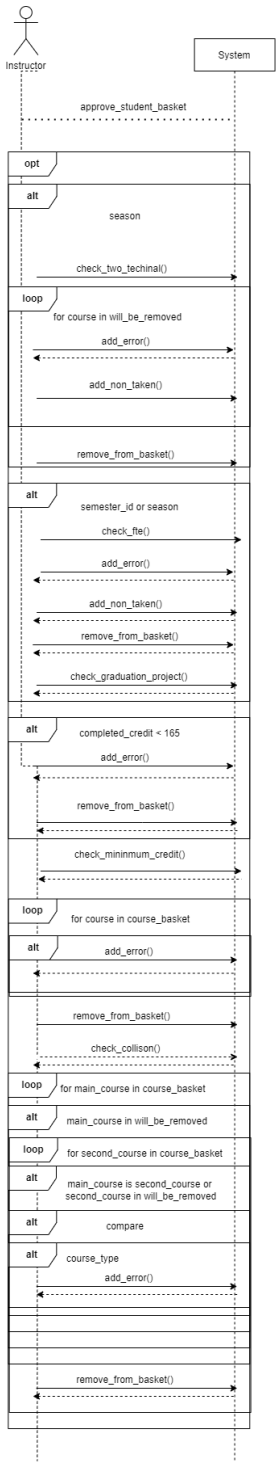


Figure 2: System Sequence Diagram of Use Case 6.2

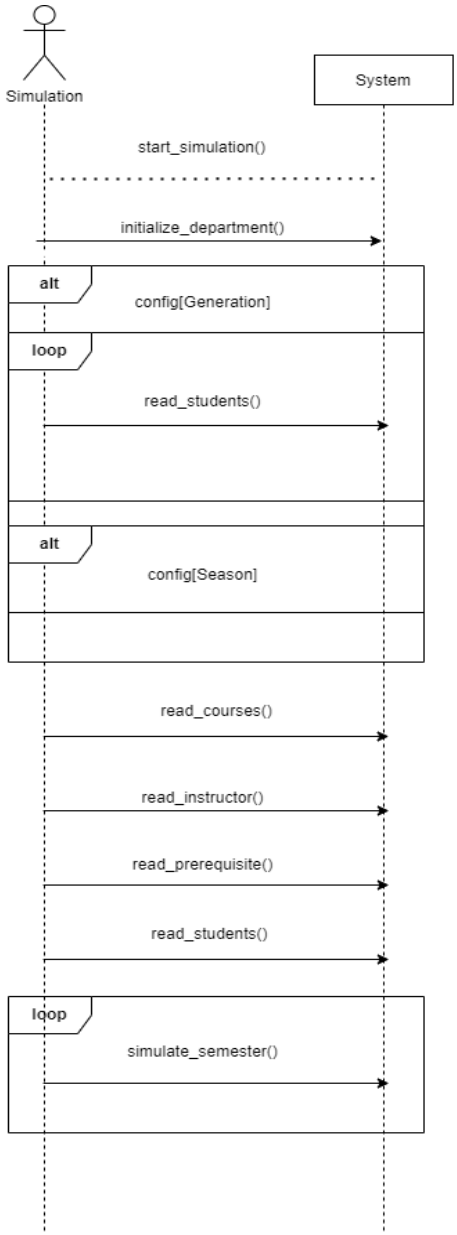


Figure 3: System Sequence Diagram of Use Case 6.3