Requirement Analysis Document

Student Lecture Registration System

	NAME	SURNAME	NUMBER
1.	Kaan	CAMCI	150119063
2.	Kadir Berk	YAĞAR	150120016
3.	Sena	VATANSEVER	150119755
4.	Fatih	ÖZTÜRK	150119519
5.	Samet	CAN	150120528
6.	Serdar	ALSAN	150120034
~~**7.**~~	~~Erman~~	~~KUNDAKÇIOĞLU~~	~~150122603~~

1.) Vision

Vision is to design and implement simulation of student lecture registration which is used by our department. It will be designed by depending on department rules and regulations for taking lecture. In developing our system, simulation will take input file and arguments such as student, lecture and simulation runs.

2.) Description

This student registration system is the need of all universities like our university. Unless this registration system design exists, it would be exactly harder to carry out process of taking lectures for students. On the other side, there may be many problems while taking lectures by students such as lecture quota, prerequisite courses, collision.

In order to avoid these problems, we design this software which simulates registration of courses. Basically, student takes courses by using registration system from their curriculum and advisor observes their requests by taking care of problems mentioned above.

3.) Requirements

Functional Requirements

a. Lecture requirements

- Lecture must have lecture ID.
- Lecture must have name.
- Lecture must have lecture type.
- Lecture must have credit score.
- Lecture must have quota limit.
- Lecture may have different sessions.
- Lecture may have pre- requisite lecture(s).

b. Student requirements

- Student must have advisor.
- Student must have student ID.
- Student must have transcript.
- Student must have entry date.
- Student must have application registration.
- Student may lecture list. g-student may have debt.

c. Lecture enrollment requirements

- Selected lectures must not have confliction.
- Student must pay education debt.
- Advisor must approve if requirements met.

Non-Functional Requirements

- Student information is taken from JSON file as an input.
- Registration system program has to be implemented with JAVA.
- Every step of program should be performed on Command Line Interface.

4.) Use Cases

Use Case Name: Enrolling To lectures

Summary: In order to get lectures, students must enroll to lectures from system and send request to their advisor.

Subject: Student

Basic Flow: 1- Student opens "Enroll to lectures" tab at site to view lectures. 2- Student needs to select their lectures from a lecture list according to their current curriculum, current semester and their current status of lecture progression. 3- System saves lectures as draft at another tab. 4- Student sends their lecture draft list to their advisor for approval. 5- Student's enrolling to a lecture process has done, lectures from list added to their syllabus.

Alternative Flow: Step 2: if selected lecture has reached their quota capacity system gives warning to student, then use case returns to step 2. Step 4: if student needs to pay education debt and he/she didn't paid it yet, system sends warning to student when he/she sends draft to advisor and student returns to step 2. Step 4: if there is conflict at lecture draft student can't send lecture draft to advisor, student returns to step 2. Step 5: if student sent TE lecture to advisor for approval, even though they have taken 2 TE lectures in fall semester, advisor rejects that TE lecture. Step 5: if student sent TE lecture to advisor for approval, even though they have taken 3 TE lectures in spring semester, advisor rejects that TE lecture. Step 5: if student sent FTE lecture to advisor for approval and if student's graduation is impossible at that semester advisor rejects that FTE lecture.

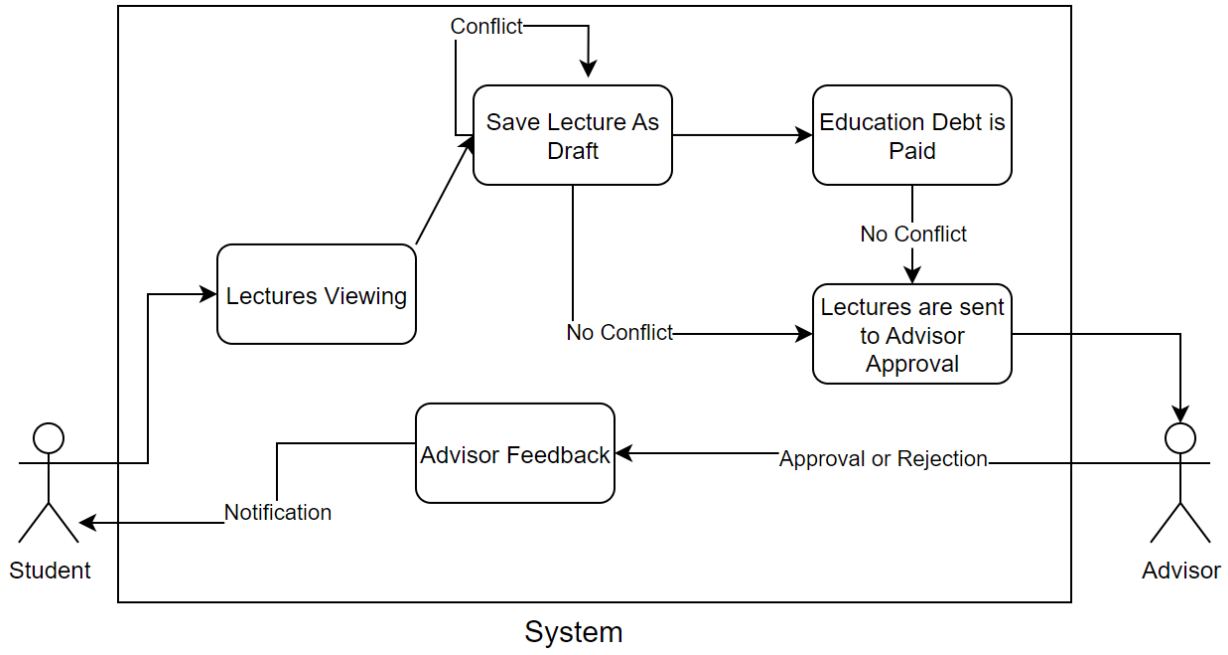
Use Case Name: Approving/Rejecting lecture Requests

Summary: In order to enregister students to lectures, advisor must enter the system and approve or reject coming requests from their students.

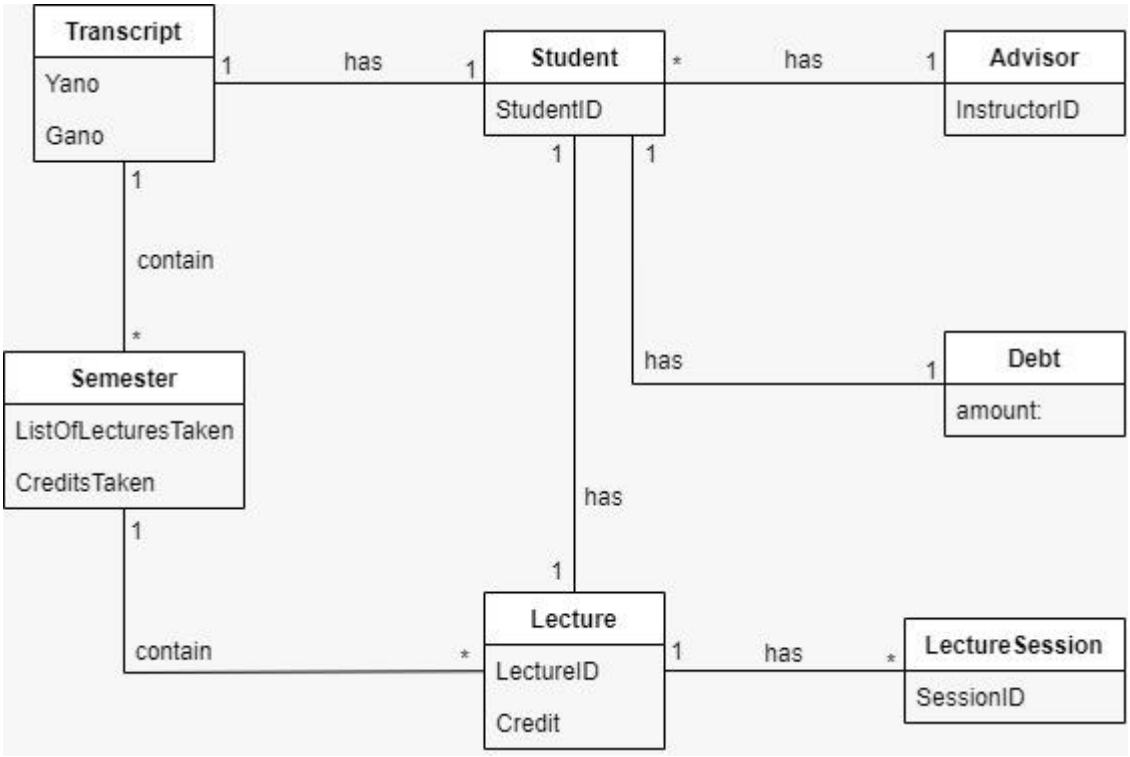
Subject: Advisor

Basic Flow: 1- Advisor enters to systems' site and selects coming requests tab from site for approving or rejecting coming requests. 2- Advisor checks coming lecture requests from student. 3- Based on advisor's feedback system sends notification to student.

Alternative Flow: Step 2: if request has TE lecture even though student took 2 TE lectures in fall semester, advisor needs to reject that TE lecture, after that advisor returns to step 2 for another student. Step 2: if request has TE lecture even though student took 3 TE lectures in spring semester, advisor needs to reject that TE lecture, after that advisor returns to step 2 for another student. Step 2: if request has FTE lecture and if student's graduation is impossible at that semester advisor needs to reject that FTE lecture, after that advisor returns to step 2 for another student.

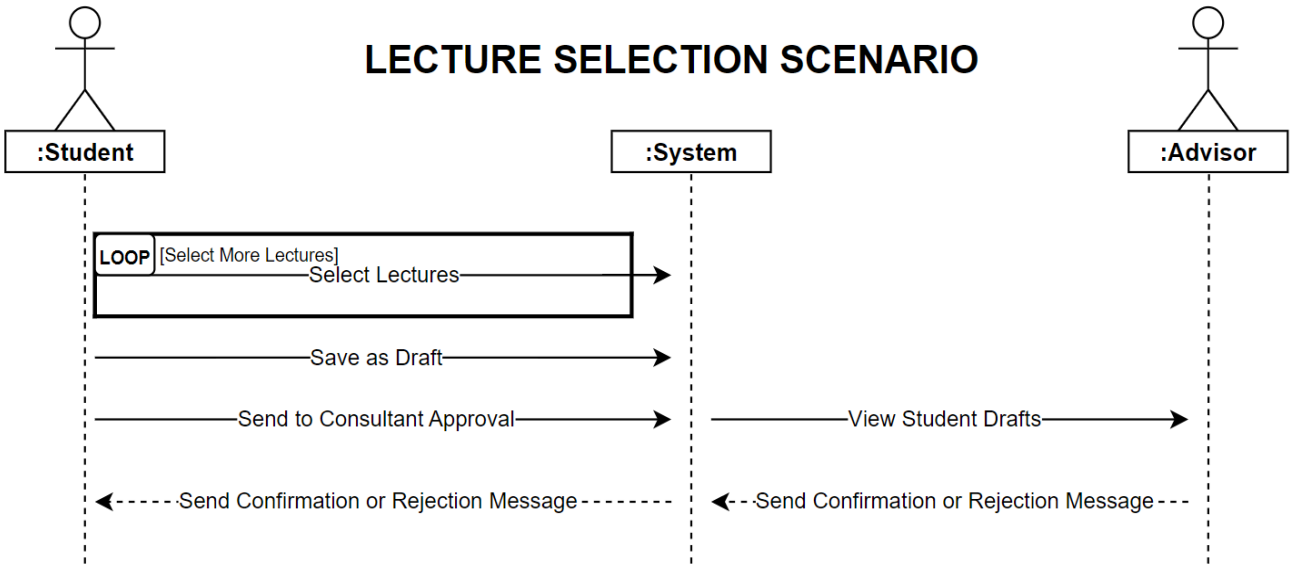


5.) Domain Model



6.) System Sequence Diagram (SSD)

1. Students view the lectures they can take, save the selected lectures as drafts.
2. Students with tuition debts pay their debts. (This is an alternate step)
3. If there is no conflict in the selected lectures, the student sends the selected lectures to the advisor's approval.
4. The advisor checks the lectures chosen by the student.
5. The advisor approves the selected lectures.



7.) Glossary

- **Lecture :** Lessons students take
- **Lecture Session :** given lectures by different lecturers
- **Transcript :** Students' grade of lecture records
- **Advisor:** Actor who approves or rejects requests of students for taking lectures , stakeholder Curriculum : All of the semester's lectures
- **Student :** Actor who takes lecture from system , stakeholder
- **Schedule :** Student's weekly plan
- **Semester :** A half-year term in school, university
- **FTE Lecture :** Faculty Technical Elective Lecture
- **NTE Lecture :** Non Technical Elective Lecture
- **TE Lecture :** Technical Elective Lecture
- **Prerequisite Lecture :** Required lecture must be completed prior to take other Lecture
- **UE Lecture :** University Elective Lecture Credits : Is a way of measuring and impression of particular lecture
- **Java :** Object Oriented Programming Language
- **JSON :** Standard data interchange and file format