Requirement Analysis Document

Student Lecture Registration System

	Name	Surname	Number
1.	Ahmet Abdullah	Gültekin	150121025
2.	Ömer	Deligöz	150120035
3.	Melik	Özdemir	150120004
4.	Hamza Ali	İslamoğlu	150120063
5.	Enes Haksoy	Öztürk	150120024
6.	Mehrin	Kaya	150123824
7.	Ayşe Gülsüm	Eren	150120005

~~Deleted parts~~

Added parts

1) Introduction

In this project, our aim is to create a student course registration system. This system will be like the course registration system used by our school. Students will interact with their advisors according to specific rules to successfully complete or not complete the course registration process.

1.1) Purpose of the System

Our system's purpose is to enable students to effectively track their course statuses and enhance communication with student advisors.

1.2) Scope of the System

The system scope includes interactions related to course registrations, encompassing activities such as course enrollment, approval/rejection, and viewing student information.

1.3) Objectives and Success Criteria of the Project

a) Objectives of the Project:

- Developing a simple Command Line Interface for students to manage their course registrations and academic information.
- Making the course registration process more efficient and accessible for students.
- Creating interaction between students and advisors
- Enabling the advisor to access student requests and course statuses.

b) Success Criteria of the Project:

- The system should be accessible accurately and promptly by both students and advisors.
- The system should process login information accurately and present correct data to the users.
- It should accurately reflect user requests, course statuses, and added data in a current and precise manner.
- Ensuring accurate processing of added data and maintaining consistency within the system.

- The system should be user-friendly and effective based on user feedback.
- The system should effectively reflect student and advisor interactions through feedback.

2) Current System

In our current system, there are two primary users: the student and the advisor. These two users are the fundamental actors interacting with and utilizing the system. When students meet the requirements of class year and the limitation on the number of courses, they have the ability to submit a draft to their advisors as a request. Rules defined during the course addition process are processed by the system, presenting the student with measures such as specific course restrictions, or preventing the addition of more courses.

For each student's draft that reaches the advisor, a request is generated, and approval or rejection is based on the advisor's decision. The courses within an approved draft are added to the student's transcript. If a draft is not approved, the student can edit and resubmit it after receiving the rejection. If the student already has a pending request awaiting the advisor's response, the student cannot submit a new request until the current one is addressed. This structure ensures a more organized and effective communication and workflow between students and advisors within the system.

Despite the two mentioned constraints, there is another restriction related to the enrollment in courses, which is the prerequisite course requirement. For a student to enroll in a course, if the course has a prerequisite, the student must have successfully completed the prerequisite course with a passing grade (AA, BA, BB, CB, CC, DC, DD).

Furthermore, it's implemented that some courses is composed of various course sections, each with its unique attributes and details.

3) Proposed System

~~A system operating in accordance with the principles of Marmara University course registration system.~~

In the recently proposed system, several modifications have been implemented. Here's a detailed breakdown of each update:

1. **Course section implementation were removed; instead, lab sections that could be used were added:**
    - Course section was removed, and in its place, laboratory (lab) sections were added. This can make course organization more specific and appropriate.
2. **T.A's were added to the project:**
    - Teaching Assistants (T.A's) have been included in the project. This is a significant update to provide more support and guidance to students.
3. **Conflict control was introduced for the course registration system:**
    - Conflict control was implemented to check for conflicts during course addition and removal. This can contribute to a more organized process and help prevent conflicts.
4. **Notification feature was added:**
    - Notifications were added to inform students and advisors about the status after sending a course. This can enhance communication and make the process more transparent.
5. **Draft system was updated:**
    - The draft system from the previous version, which added courses to the student's transcript after advisor approval, has been updated. Now, students can add or remove courses in a draft, and only after advisor approval are they added to the transcript. During the add-drop week, students can still remove newly added courses.

6. **Increased user and course diversity:**
    - Lecturer, advisor, student, and course diversity have been increased. This allows for more data and improved usability by a variety of users.
7. **Logging system was added:**
    - A logging system was introduced to track processed data. This is an important feature for data security and tracking.
8. **Coloring feature was added:**
    - With the increase in data and functionality, coloring was added to the command line interface for a more user-friendly experience.
9. **University structure was added:**
    - A university structure was implemented to hold all data and facilitate future department additions. This can ease expansion and management within the system.

With these updates, our system appears to be more user-friendly, flexible, and expandable.

3.1) Overview

In the project, there are two types of users: advisor and student. Each user type has its own viewing menus with distinct displays. A student can view his/her requests, course registrations, transcripts, advisor information, and check whether the registration is successful or not. Similarly, advisors can view the information and request of students and provide feedback such as approval or rejection of requests.

3.2) Requirements

a) Functional Requirements

1. Student Requirements

- Student must have an advisor.
- Student must have student ID.
- Student must have transcript.
- ~~Student must have class year.~~
- Student must have semester number
- Student may have schedule.
- Student may have request.
- Student must have username to login.
- Student must have password to login.
- Student may have get notification from the system.

2. Advisor Requirements

- Advisor must have an ID.
- Advisor must have username to login.
- Advisor must have password to login.
- Advisor must have students that they advise.
- Advisor can have list request(s).
- Advisor may have get notification from the system.

3. Registration Requirements

- Registration system includes a student and student's desired courses.
- Registration system sends the list of selected courses to the advisor.

4. Course Requirements

- Course must have course name.
- Course must have course code.
- Course must have credit score.

- Course must have semester number.
- Course may have lab sections.
- Course must have a lecturer who teaches.
- Course must have the current enrollment number.
- Course must have student capacity number.
- Course must have hour.
- Course must have day.
- Course may have pre-requisite course(s).

5. **Notification Requirements**
    - Notification must contain message.
    - Notification must belong to a recipient.

b) Non-Functional Requirements

1. **Obtaining Student Information from JSON Input:**
    - Student information should be obtained from JSON files as input.
2. **Secure Storage and Transmission of User Data:**
    - User data should be stored securely to prevent unauthorized access or data breaches.
3. **User Interface:**
    - The user interface should be easily understandable and user-friendly.
    - The layout should be carefully selected to enable users to access information and interact with the system effortlessly.
4. **Implementation of Registration System Program with JAVA:**
    - The registration system program must be implemented using the JAVA programming language.

- The latest stable version of JAVA and appropriate libraries should be utilized to ensure compatibility and security.

5. **All Steps Should Be Performed on the Command Line:**
    - All registration and other processes should be carried out via the Command Line interface.
    - This ensures that users can interact with the system through command-line instructions for efficient management purposes.
6. **Error Handling:**
    - Meaningful Error Messages: The program should provide clear and meaningful error messages in case of issues during execution.
    - Logging: Log files should capture and store information about errors for later analysis and troubleshooting.
    - Graceful Continuation: The user should be able to continue with the program in its normal flow, even when errors occur, to maintain a smooth user experience.

4) Use Cases

a) Student Use Case-1

Scope: Course Registration System

Level: User Goal

Primary Actor: Student

~~**Description:** This use case describes the process by which a student registers for a course using the Course Registration System.~~

Description: This use case describes the process by which a student registers for a Mandatory/Non-Technical Elective courses using the Course Registration System.

StakeHolders and Interests:

-Student: requires proper, fast course operations as it affects his/her student career.

-**Advisor:** Accepts or rejects students' chosen courses in a way that is most productive for students.

Preconditions:

-Students' and Advisors' information is identified and authenticated.

Success Guarantee:

- The courses that are chosen by students are accepted. The students are added to course. The transcripts are updated.

Main Success Senario:

1.Student logs in the system with username and password.

Repeat 1 until username and password are correct.

~~2. Student lists the courses that can be chosen.~~

2.System checks the prerequisite of courses if there is any.

3.System checks the probation condition for student.

4.Student lists the Mandatory/Non-Technical Elective courses that can be chosen.

5.Student selects a course

6.Student adds the selected course.

Repeat 5 until the course adding process are done.

7.Student saves the draft.

8.Student sends draft to Advisor

9.The courses that are chosen and sent to the Advisor are accepted.

10.Student logs out of the system.

Alternative Senarios:

1.Student can exit from the system.

2.Student can return to main menu.

3.Student can view the his/her own transcript.

~~4.Student can list the courses that were chosen by his/herself.~~

4.Student can list the Mandatory/Non-Technical Elective courses that were chosen by his/herself.

5.Student can list the status of request.

6.Student can not take new semester courses and must take failed or DD courses if probation condition is accurate.

b) Student Use Case-2

Scope: Course Registration System

Level: User Goal

Primary Actor: Student

Description: This use case describes the process by which a student registers for a Technical Elective-Faculty-Technical Elective courses using the Course Registration System.

StakeHolders and Interests:

-**Student:** requires proper, fast course operations as it affects his/her student career.

-**Advisor:** Accepts or rejects students' chosen courses in a way that is most productive for students.

Preconditions:

-Students' and Advisors' information is identified and authenticated.

Success Guarantee:

- The courses that are chosen by students are accepted. The students are added to course. The transcripts are updated.

Main Success Senario:

1.Student logs in the system with username and password.

Repeat 1 until username and password are correct.

2.System checks the 155 credit prerequisite for courses.

3.Student lists the Technical Elective-Faculty-Technical Elective courses that can be chosen.

4.Student selects a course.

5.Student adds the selected course.

Repeat 4 until the course adding process are done.

6.Student saves the draft.

7.Student sends draft to Advisor

8.The courses that are chosen and sent to the Advisor are accepted.

9.Student logs out of the system.

Alternative Senarios:

1.Student can exit from the system.

2.Student can return to main menu.

3.Student can view the his/her own transcript.

4.Student can list the Technical Elective-Faculty-Technical Elective courses that were chosen by his/herself.

5.Student can list the status of request.

6.Student can not take TE-FTE courses if complete credit is less than 155.

c) Student Use Case-3

Scope: Course Registration System

Level: User Goal

Primary Actor: Student

Description: This use case describes the process by which a student drops courses using the Course Registration System.

StakeHolders and Interests:

-**Student:** requires proper, fast course operations as it affects his/her student career.

-**Advisor:** Accepts or rejects students' chosen courses in a way that is most productive for students.

Preconditions:

-Students' and Advisors' information is identified and authenticated.

Success Guarantee:

- The courses that are chosen by students are dropped. The transcripts are updated.

Main Success Senario:

1.Student logs in the system with username and password.

Repeat 1 until username and password are correct.

2.Student opens the drop course menu and lists droppable courses.

3.System takes the transcript database and look for the student's ongoing courses.

4.Student selects a course.

5.Student drops the selected course and add to the draft for deleting.

Repeat 4 until the course dropping process are done.

6.Student saves the draft.

7.Student sends draft to Advisor.

8.The courses that are chosen to drop and sent to the Advisor are accepted.

9.System delete the courses from transcript and send them to the available courses.

9.Student logs out of the system.

Alternative Senarios:

1.Student can exit from the system.

2.Student can return to main menu.

3.Student can view the his/her own transcript.

4.Student can list the droppable courses that were chosen by his/herself.

5.Student can list the status of request.

6.Student can take the dropped courses again.

7.Student can delete courses from draft.

a-Student choose the deletable courses in draft.

b-Repeat a until deleting from draft process are done.

c-Return to the adding operation.

d) Advisor Use Case

Scope: Course Registration System

Level: User goal

Primary Actor: Advisor

~~**Description:** This use case describes the process by which a student registers for a course using the Course Registration System.~~

Description:This use case describes the process by which a advisor operations for courses that are chosen by student in course registration system.

StakeHolders and Interests:

-Student: requires proper, fast course operations as it affects his/her student career.

-**Advisor:** Accepts or rejects students' chosen courses in a way that is most productive for students.

Preconditions:

-Students' and Advisors' information is identified and authenticated

Success Guarantee:

-The courses that are chosen by the student are accepted.

Main Success Senario:

1.Advisor logs in to the system by entering the username and password.

Repeat 1 until the username and password are correct.

2.Advisor checks the requests.

3.Advisor accepts the course that are chosen by student.

Repeat 3 until the there are no request left to process.

4.Advisor logs out of the system.

Alternative Senarios:

1.If the advisor identifies issues with the selected course, advisor rejects the request.

2.If the advisor wants to check the requests of student that is chosen by advisor, he/she can do this operation.

3.Advisor accepts the courses that student wants to drop.

4.Advisor can exit from the system.

5.Advisor can return to main menu.

5) Domain Rules

1. **Student Domain Rules:**
    - A student cannot register for a course without having an assigned advisor.
    - A student can only log in with a valid username and password.
    - Each student must have a unique student ID.
    - A student's transcript should be updated promptly after successful course registration.
    - Only students of a certain class year can register for specific courses.
    - A student wishes to enroll in a course, they must have been successful in the prerequisite courses for that specific course.

      a) Mandatory Courses:
    - For the student to take the course, they need to be in the semester or higher.
    - If there is a prerequisite for the course, the student needs to satisfy that prerequisite.

      b) NTE (Specialization Courses):
    - For the student to take the course, they need to be in the semester or higher.

      c) TE-FTE (Total Credit Requirement):
    - The student needs to complete a total of 155 credits, excluding 4th-year courses and NTE courses.

      d) Engineering Project:
    - The student needs to complete a total of 165 credits, excluding 4th-year courses and NTE courses.
    - If a student wishes to take a course as an advanced standing, their GPA (Grade Point Average) must be 3 or higher.
    - If a student wishes to enroll in two courses that are scheduled for the same time, there should not be a requirement for attendance in at least one of them; otherwise, the student cannot submit a request to the advisor, system rearranges add courses menu.
    - Once a student adds a course through the course registration add course menu and after that when it gets processed onto the transcript following advisor approval, the student can still perform add-drop actions for that particular course on transcript.
    - If a student is enrolled in a course that includes lab sections, they are required to choose one of the available lab sections.
2. **Advisor Domain Rules:**
    - An advisor must have a unique ID.
    - An advisor can only log in with a valid username and password.
    - Advisors can only manage course requests for students they advise.
    - Advisors should provide timely feedback on course requests.

3. **Course Domain Rules:**
    - Each course must have a unique course code.

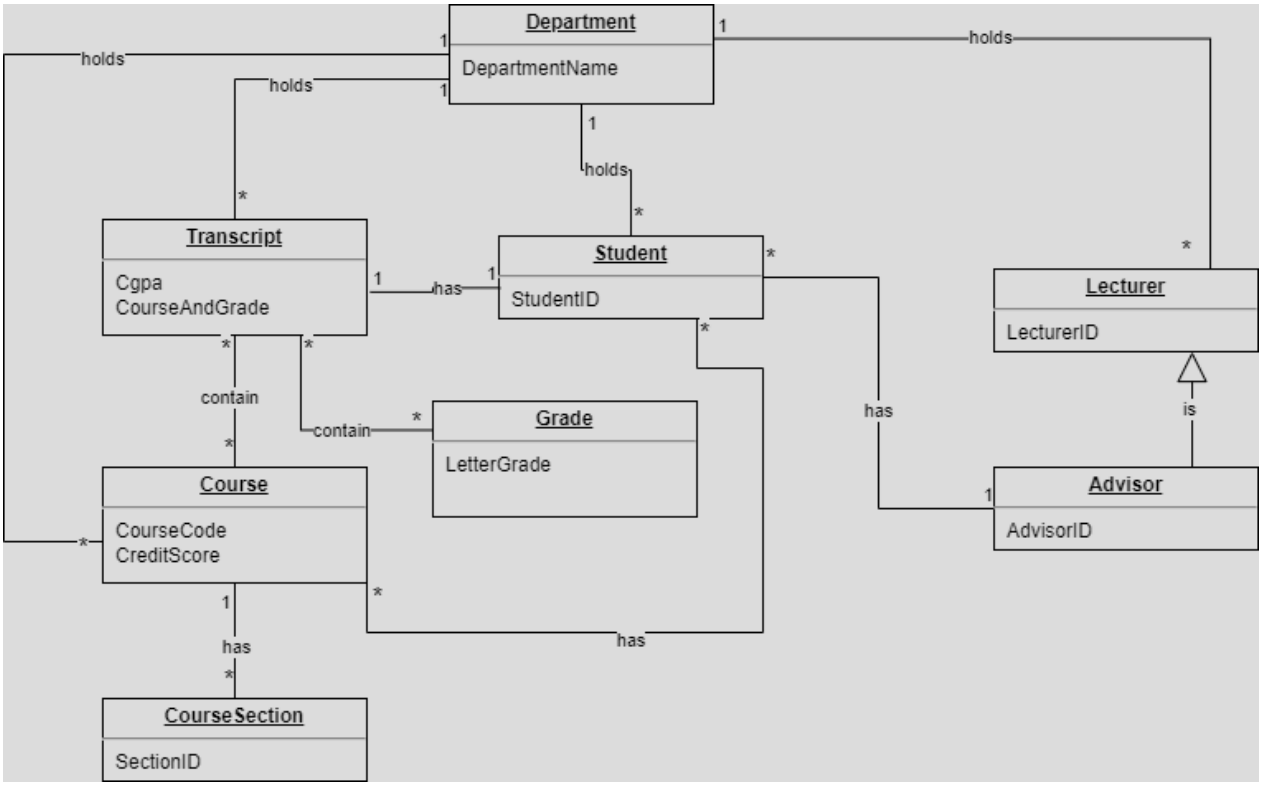
- Pre-requisite courses must be clearly defined and enforced during the registration process.
- Courses are categorized into four types: mandatory, non-technical elective, technical elective, faculty technical.
- Course may have several lab sessions.

4. **User Interface Domain Rules:**
   - The user interface should follow accessibility standards.
   - Error messages should be clear and informative.

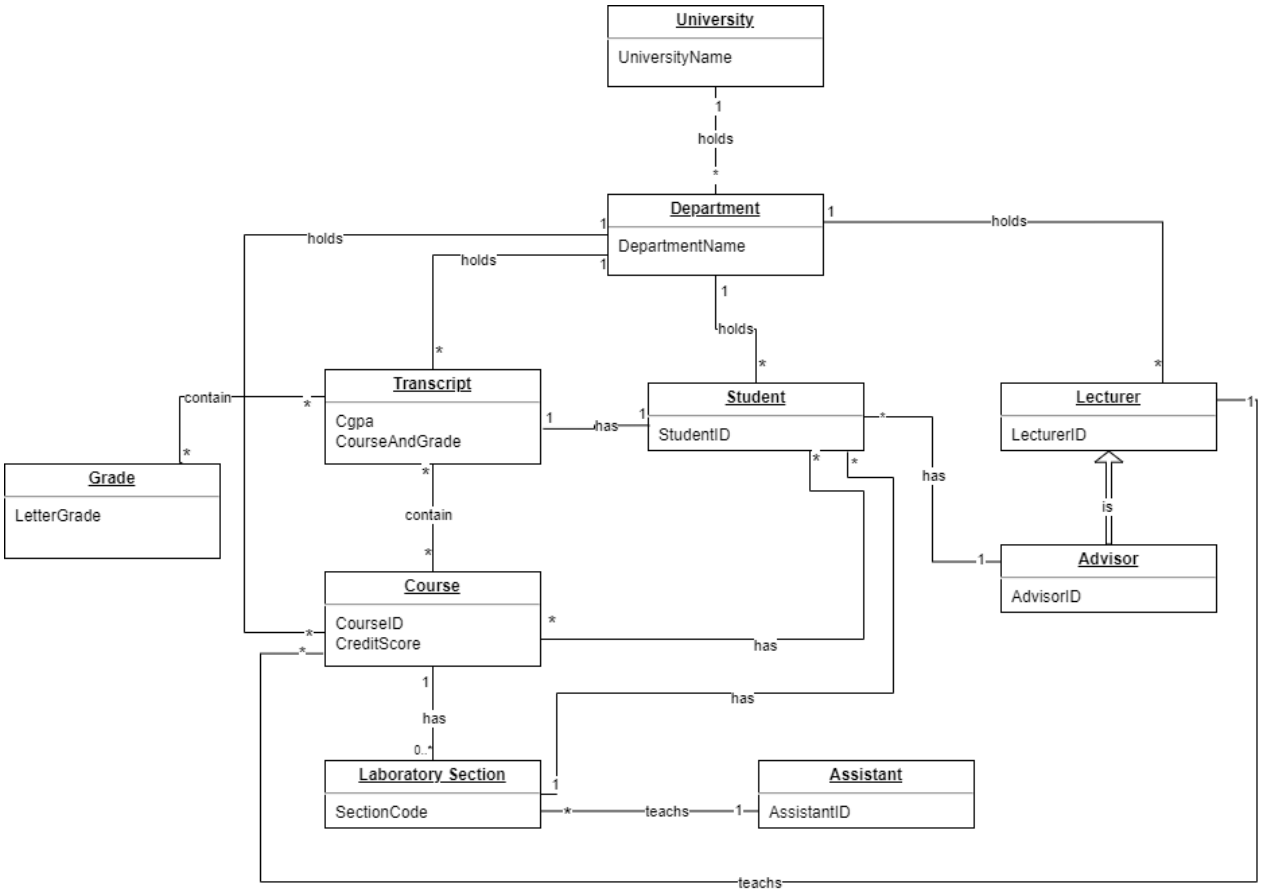
5. **JSON Input Domain Rules:**
   - The JSON input format for student information should adhere to a specific schema.
   - Data integrity checks should be performed when reading information from JSON files.

6. **Logging Domain Rules:**
   - All necessary system flow should be logged in a single file for each execution.

~~6) Domain Model~~

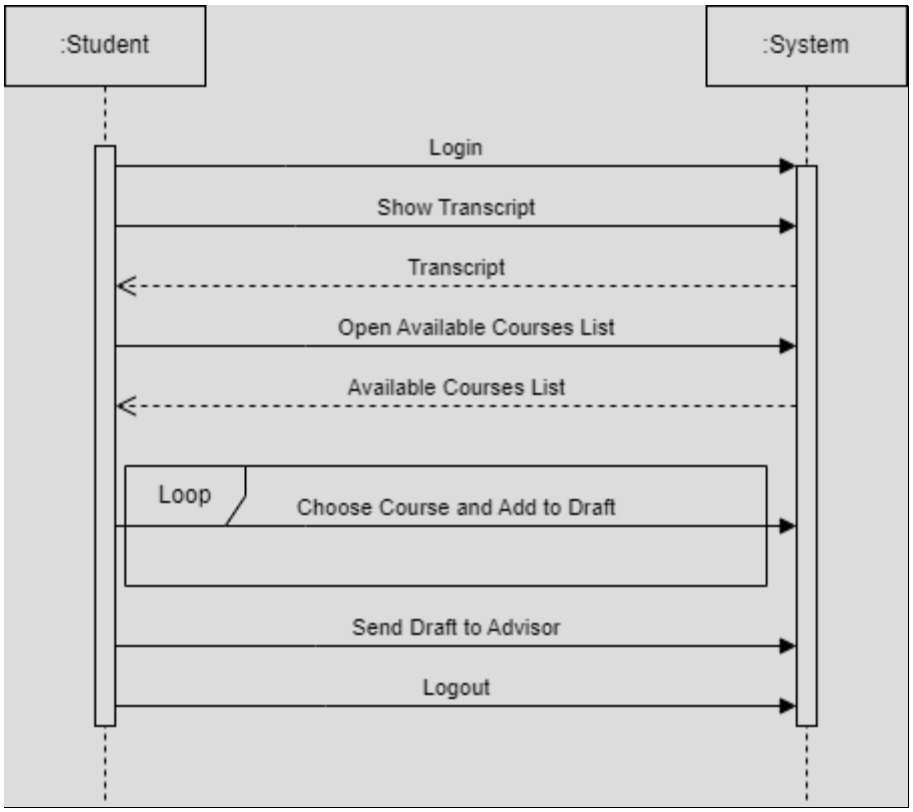


6) Domain Model

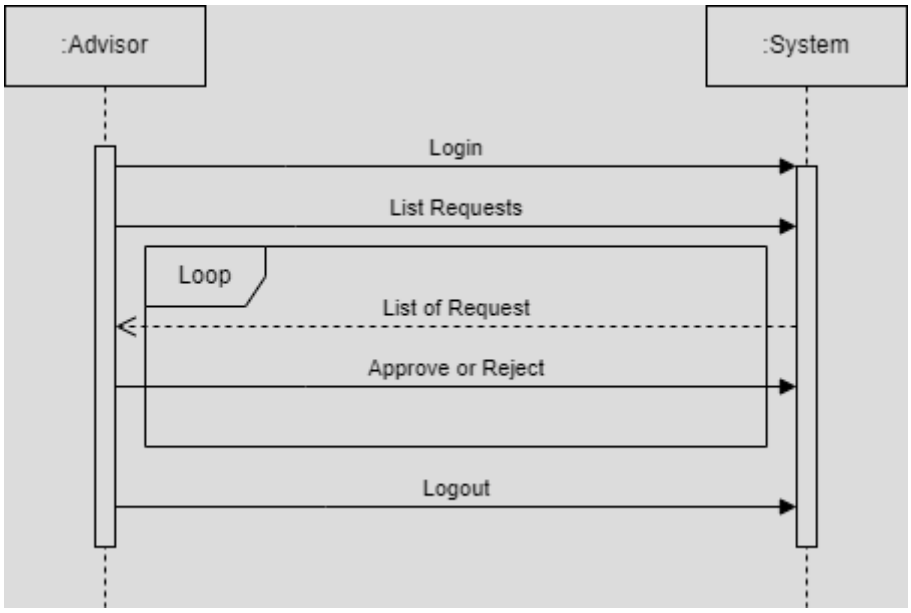


7) System Sequence Diagram (SSD)

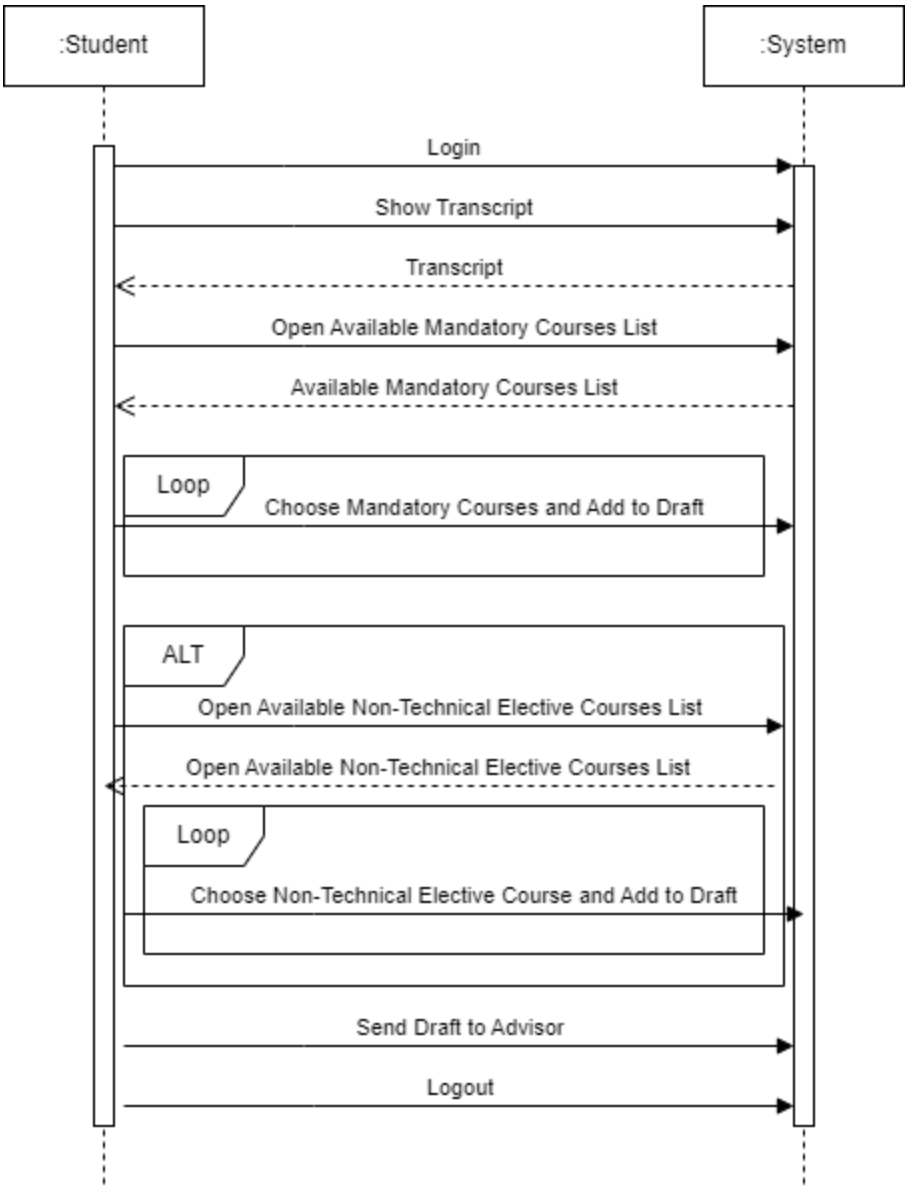
~~a) Student SSD~~



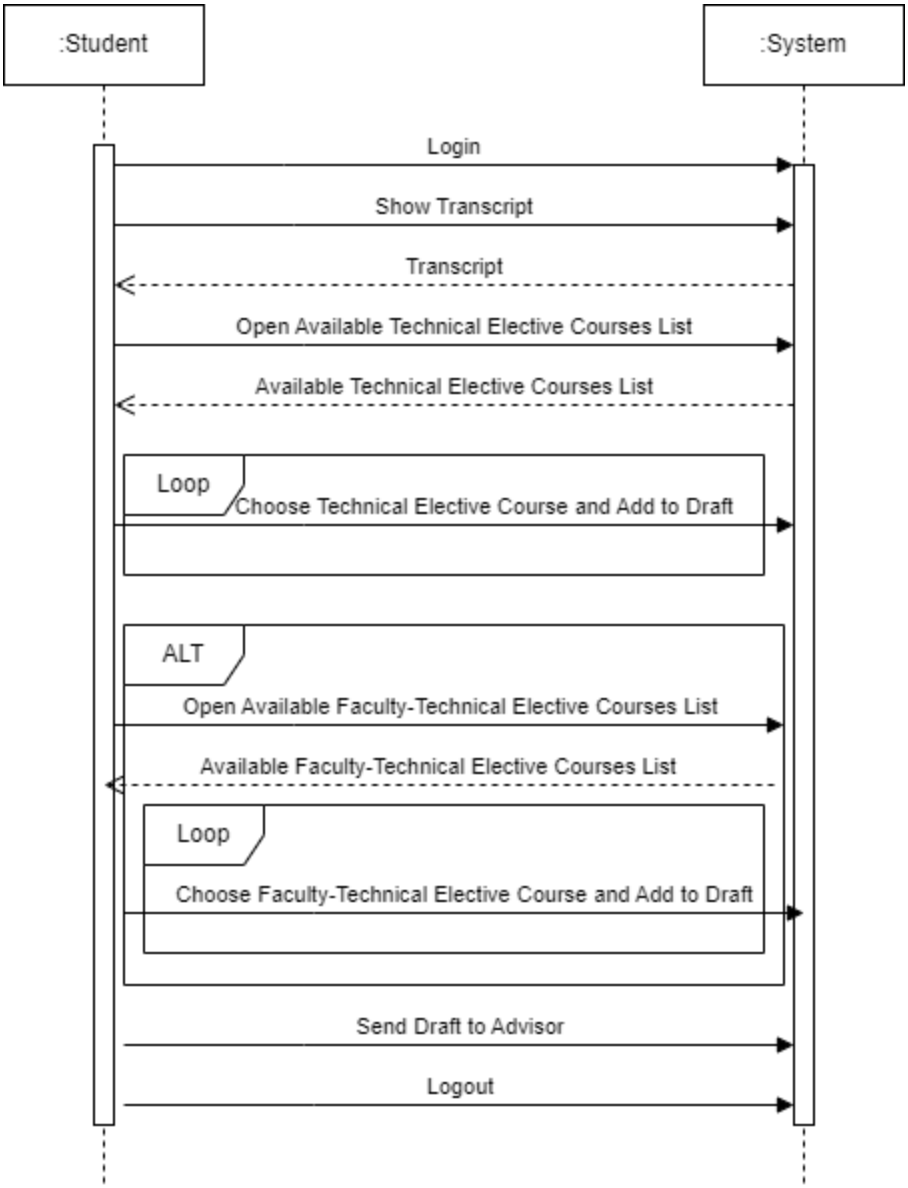
~~b) Advisor SSD~~



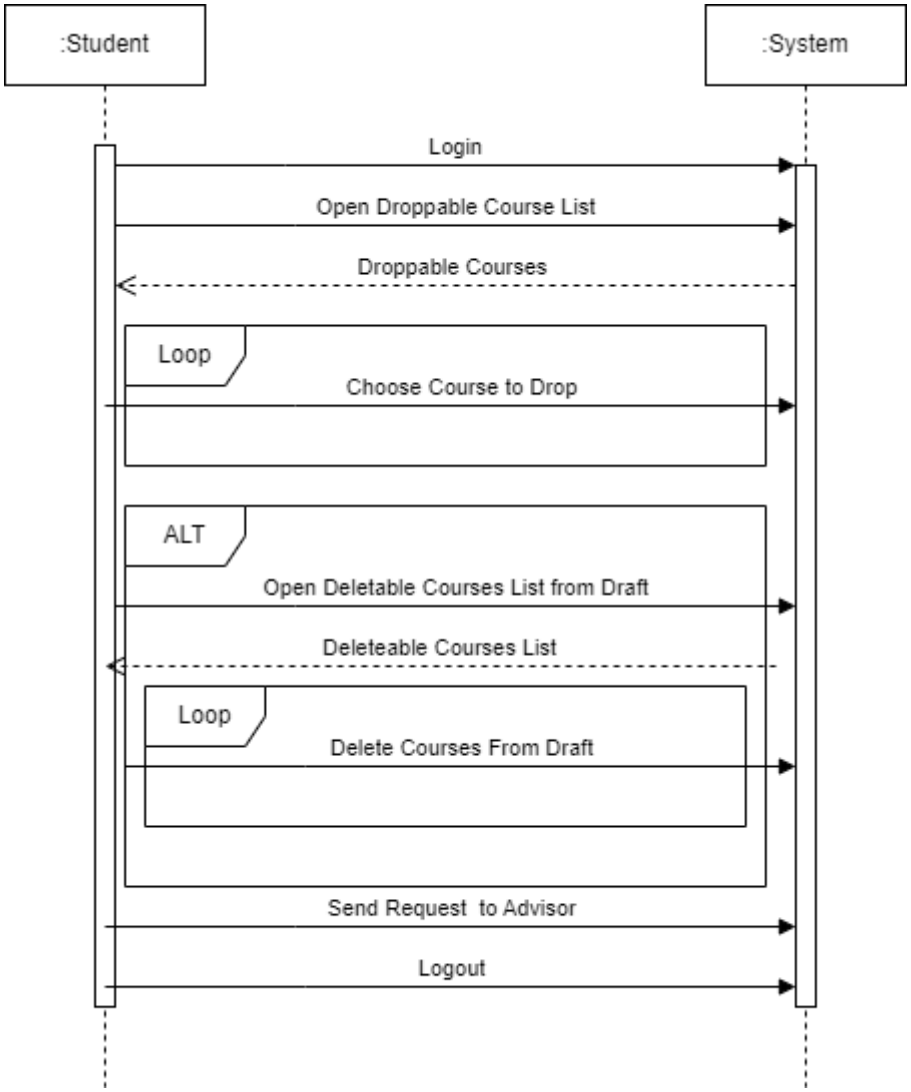
a) Student SSD – 1



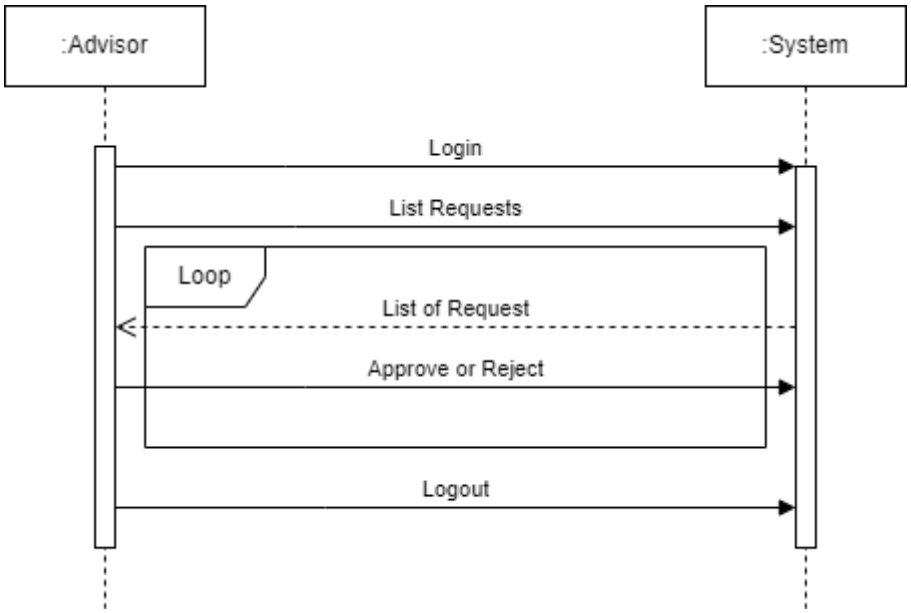
b) Student SSD – 2



c) Student SSD – 3



d) Advisor SSD – 1



8) Glossary

- **Student Lecture Registration System:**
  A software system designed to facilitate the course registration process for students in an educational institution.
- **Advisor:**
  An academic or administrative staff member responsible for guiding and assisting students in academic matters, including course selection and registration.
- **Command Line Interface (CLI):**
  A text-based user interface where users interact with the software by typing commands into a terminal or command prompt.
- **JSON (JavaScript Object Notation):**
  A lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate.
- **Functional Requirements:**
  Specifications describing the functions and capabilities that a system must possess, including both user and system interactions.
- **Non-Functional Requirements:**

Specifications that describe the qualities and characteristics the system must have, such as security, performance, and usability.

- **JAVA:**

  A high-level, class-based, object-oriented programming language designed for creating robust and secure applications.

- **Use Case:**

  A description of how a system will be used, typically outlining interactions between users (actors) and the system to achieve a specific goal.

- **Transcript:**

  A document that provides a record of a student's academic performance, including grades and courses completed.

- **Course Code:**

  A unique identifier assigned to a specific academic course.

- **Credit Score:**

  A numerical representation of the academic credit value assigned to a course, indicating the workload and difficulty level.

- **Pre-requisite Course(s):**

  Courses that must be successfully completed before a student is allowed to enroll in a particular course.

- **Main Success Scenario:**

  The expected sequence of actions and events leading to the successful accomplishment of a use case.

- **Alternative Scenario:**

  Variations in the sequence of actions and events that may occur in response to different conditions or inputs during the execution of a use case.

- **Logging:**

  Recording the events that occur during the operation of a computer program is the process of.

• Mandatory:

Mandatory courses are those that students must complete, typically covering fundamental knowledge and skills.

• Non-Technical Elective:

Non-technical elective courses provide students with the freedom to choose from non-technical subjects, aiming to offer a broad perspective.

• Technical Elective:

Technical elective courses offer students the opportunity to deepen their knowledge in a specific specialization or area of interest.

• Faculty Technical:

Faculty technical courses are technical courses designed for students within a specific department or faculty, often covering topics that require expertise.

• Engineering Project:

Engineering project courses involve hands-on application of engineering principles and skills to solve real-world problems, providing students with practical experience in project development and implementation.

• Teaching Assistants (T.A's):

Teaching Assistants (T.A's) are helpers who work with teachers, typically in universities or other educational institutions.