metin, yazı tipi, logo, simge, sembol içeren bir resim

Açıklama otomatik olarak oluşturuldusimge, sembol, logo, yazı tipi, ticari marka içeren bir resim

Açıklama otomatik olarak oluşturuldu**CSE 3063 PROJECT**

**REQUIREMENT ANALYSIS DOCUMENT (RAD)**

**Iteration 2**

**Group 2**

*150120031 - Şükrü Can Mayda  
150121055 - Alp Büyükköse  
150121065 - Turgut Köroğlu  
150120995 - Seyyid Ali Koldaş  
150121545 - Nurbetül Çakır*

**Project Description**

In this project, we will be creating a course registration system for our department. On the first iteration of the project, the system will accept two types of users: Student and Advisor. Additional roles may be added in next iterations.

When students login to the system, they will be able to choose the courses they will be taking for their current semester. Then, they will send their selection to their assigned advisor to be confirmed. The advisor will be able to see all courses each student has chose and they will approve or disapprove these courses. In the case of approval, the registration process will end successfully. In the case of disapproval, the student will be choosing their courses again based on the limitations.

Furthermore, the students will be able to view their past grades in their transcript, check their current term schedule and learn about the prerequisites. Advisors will also be displaying the students they are held accountable for and get in touch with them via the contact information in their profiles.

Our main goal is to make a clear and understandable, easy-to-use registration system for students and advisors to smooth the academic processes with security and reliability involved.

**Glossary**

**Advisor:** Teacher who is responsible for helping a particular student group, including adding/dropping courses and providing assistance on other subjects.

**Advisor Approval:** The advisor confirmation process of courses selected to be taken by a student.

**Authentication:** The process of confirming a user's identity before allowing access to the system.

**Choosable Course:** A course that students can choose during the registration process, as long as they meet the required prerequisites and there is space available in the course.

**Conflict Detection:** The process of identifying and resolving scheduling issues, like overlapping course times or unmet prerequisites, during course registration.

**Course:** The collected information of a specific subject.

**Course Section:** A specific course offering that includes a set time, classroom location, and maximum number of participants.

**Course Session:** A single time slot on a specific day for a course section that is available during a particular term or semester.

**Credit:** A unit representing the workload of a course.

**Cumulative Grade Point Average (CGPA**): The success measure of a student’s past terms calculated by the cumulative earned grades in their courses.

**Department Head:** A senior faculty member who oversees the academic and administrative functions of the department, including the approval of course schedules, policies, and regulations.

**Department Scheduler:** A role in the system that assigns time slots and classrooms to course sections.

**Earned Credit:** The total credit of courses a student enrolled in the current term.

**Enrollment:** Registration to a course.

**Grade Point Average (GPA):** The success measure of a student’s current term calculated by the earned grades in their courses.

**Lecturer:** Person who is responsible for giving lectures to students.

**Login:** The process of signing in to a website by a user.

**Password:** The key of the student's login on the registration system.

**Prerequisite Course:** The course that is mandatory for a specific course.

**Quota:** The maximum number of students allowed to enroll in a particular course or course section.

**Received Credit:** The total credit of courses a student enrolled and successfully completed in the current term.

**Schedule Management:** The feature that enables students to view, modify, and manage their weekly class schedules.

**Student:** Person who is taking courses and is responsible for these courses.

**Student Affairs:** A department responsible for handling administrative tasks related to students, including enrollment, transcripts, and responding to student inquiries and concerns within the system.

**Transcript:** Summary of a student's academic performance and progress to date.

**Waitlist:** A feature of the course registration system that enables students to join a waiting list for courses that are already full.

**Functional Requirements**

**Login:**

* Login page is the first screen when a user wants to enter the system.
* There will be unique IDs assigned to all users.
* Any user can login to the portal by entering their identification number and password.
* If the credential matches with the database’s, the user can display the main page.

**Schedule Management:**

* Students will be able to see which courses they are taking for the current term, their weekly occurrences and classrooms.
* Each individual schedule will be arranged according to the advisor’s approval of courses.

**Student Interface:**

* Students will be able to see their courses, schedule, transcript and advisor information.
* There will be a section for changing personal information.
* Students will be able to register for courses.

**Advisor Interface:**

* Advisors will be able to see their all student’s information except their password.
* There will be a section for changing personal information.
* Advisors will be able to approve or deny requests of students in the registration system.

**Course Registration:**

* Students can view and register for available courses based on the rules of prerequisite and senior year courses.
* Advisors can update the courses students can take for each term.
* During registration period, students can add and remove courses for the current semester as long as the courses are compatible with the requirements and quota.
* After the selection is done, students can submit the courses for approval to their advisors.
* Advisors can approve a request to finalize the registration or deny for the student to change its selection.
* If the request is denied, the selection will have to be made again by the student according to the rules.

**Course Compliance Check:**

* A thorough check is made before a student can list all the courses they can take if they comply with the prerequisites being passed with at least the letter grade, FD, the obligation of seniority to take the senior year courses and the obligation to take the course in the supposed semester.

**Conflict Detection:**

* Time conflicts in a student’s schedule of chosen courses are detected after the selection is made.
* The student is notified for the necessary changes to be made.
* If there exist minimal acceptable conflicts, chosen courses can be sent for advisor approval.

**Capacity Management:**

* The maximum number of students allowed in each course or course section is tracked.
* If a course or course section reaches full capacity, the system prevents students from enrolling in said courses.

**Waitlist Management:**

* Students are allowed to join a waitlist when a course or a course section’s capacity is full.
* First-come, first-served priority is applied in the queue.
* If any space becomes available, students are automatically enrolled based on the priority rule.
* Students are notified when their waiting status is updated as enrolled.

**Transcript View:**

* Students can display the list of their completed courses, grades and earned credits for each term.

**Notification System:**

* Students are notified when:
* Their waitlist status changes.
* Their course registrations are approved or rejected by their advisor.
* A course is canceled or rescheduled.
* Advisors are notified when a student sends their selected courses for approval.
* Notifications are stored and displayed until they are seen by the user.

**Department Scheduler Role:**

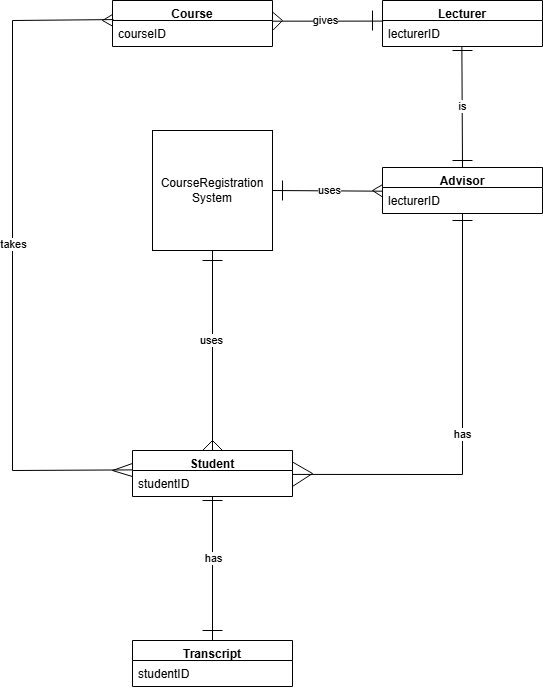
* The department scheduler has the ability to allocate time slots and classrooms to course sections while avoiding any scheduling conflicts.
* They can modify and refine course schedules to meet the needs of the department.
* They can assign larger classrooms to high-demand courses and smaller classrooms to specialized courses.

**Non-Functional Requirements**

System attributes like security, performance, usability, scalability, availability, maintainability, compatibility, compliance and data integrity.

* The system must respond promptly to user requests during logins, database updates, and logouts.
* Any changes to data, like course enrollments and schedule updates, should be accurately reflected throughout all relevant sections of the system.
* Consistent coding standards and practices should be maintained to ensure readability and reduce errors.
* While a command-line interface will be used instead of a graphical user interface, it should still be user-friendly with clear text and order.
* The system needs to be compatible with both Java and Python environments.
* The codebase should be clean, well-documented, and organized to make maintenance and future updates easier.
* Required data will be stored in separate JSON files.
* The CLI should be intuitive and easy to navigate, featuring appropriate help commands and feedback messages for errors or necessary actions.
* Sensitive information, such as student grades and course registrations, must be safeguarded against unauthorized access.
* Compatibility with various JSON formats for data storage should be ensured.
* The system should implement robust authentication to guarantee that only authorized users (students, advisors, schedulers) can access their respective accounts and data.
* The system should be designed to accommodate future growth in the number of courses, students, and users.
* Adding new features, such as additional roles, should be straightforward in future updates.
* Notifications should be clear and provide concise instructions or information for users.
* The system must be capable of handling multiple users simultaneously, especially during the course registration period.

**Domain Model**



**Use Cases**

**Use Case 1: Student Login**

Primary Actor: Student

Stakeholders and Interests:

* Students: Quick and user-friendly access to their accounts to display courses, view schedules or make any other changes
* University IT Department: Guaranteeing that the login system works smoothly and database integrity and security is protected
* Student Affairs’ Office: Managing the content of the system so that students can access the system to check their information
* University Management: Providing a reliable and accessible system backing up the institution’s reputation.

Precondition(s): Student has been registered in the system beforehand and has valid username and password.

Main Flow:

1. Login page interface opens.
2. Student enters their username and password into the blanks.
3. System checks the validity of the info.
4. If the info is correct, student is logged in and directed to the home page.
5. System’s home page is displayed.

Alternative Flow:

* System displays a message indicating some login information is incorrect.
* Student can retry logging in.

Postcondition(s):

* Student is logged in and directed to the home page.

**Use Case 2: Advisor Confirmation**

Primary Actor: Advisor

Stakeholders and Interests:

* Students: Receiving feedback from advisors on their requests and planning their academic year further
* Advisors: Quick and user-friendly access to their accounts to display students and their requests
* Departments: Ensuring that students register for the correct courses based on the requirements
* Student Affairs’ Office: Monitoring course registrations and confirmations in case any troubles come up
* University Management: Maintaining an efficient registration period backing up the institution’s reputation

Precondition(s):

* Advisor has been registered in the system beforehand and has valid username and password.
* There are waiting registration requests from students.

Main Flow:

1. Advisor logs into the system.
2. Advisor clicks the “Pending Registration Requests” section on the main page.
3. Advisor selects a student to view their request.
4. System displays the request details such as student information, course information and prerequisite condition.
5. Advisor reviews the information and decides whether to approve or reject the request.
6. Advisor clicks the “Approve” button or the “Deny” button.
7. System updates the request’s status accordingly.

Alternative Flow:

1. Advisor checks the request and chooses to deny it.
2. Systems requests a reason of denial from the advisor.
3. Advisor enters a reason in plain text.
4. System updates the status as “Denied” and includes the reason with it.

Postcondition(s):

* The student’s registration request is approved or denied.

**Use Case 3: Department Scheduler**

Primary Actor: Department Scheduler

Stakeholders and Interests:

* Students: Ensure a conflict-free schedule with properly assigned class times and rooms.
* Advisors: Allow students to enroll in courses with a well-organized schedule.
* University Management: Ensure efficient and systematic course scheduling.

Precondition(s):

* Courses, time slots, and classroom capacities must be predefined in the system.
* The Department Scheduler must be logged into the system.

Main Flow:

1. The Department Scheduler logs into the system.
2. The scheduler navigates to the "Manage Course Schedule" section.
3. The system displays a list of all courses with their current time slots and room assignments, along with an option to Create a New Course.
4. The scheduler chooses one of the following options:
5. Edit an Existing Course:
   1. The scheduler selects a course and clicks the "Edit" button to modify its time or room assignment.
   2. The system shows the availability of classrooms and highlights potential conflicts.
   3. The scheduler assigns a new time slot or classroom and clicks "Save."
   4. The system verifies that there are no conflicts and updates the schedule.
6. Create a New Course:
   1. The scheduler clicks the "Create New Course" button.
   2. The system prompts the scheduler to enter course details, including course name, course code, time slot, and room assignment.
   3. The system verifies that there are no conflicts with existing schedules.
   4. If no conflicts are found, the system saves the new course to the database.
7. Once the scheduler finishes adjustments for all courses, the system updates and shares the new schedule with all users.

Alternative Flow:

If the scheduler tries to assign a time or room that conflicts with another course:

* The system displays a warning message indicating the conflict.
* The scheduler selects a new time slot or room.
* The system rechecks the new assignment and confirms there are no conflicts.
* The system saves the changes or the new course details.

Postcondition(s):

* The updated course schedule is saved in the system.
* All users (students and advisors) can view the new schedule.

**Use Case 4: Advisor-Student Communication**

Primary Actor: Student

Secondary Actor: Advisor

Stakeholders and Interests:

* Students: Be notified when their course requests are approved or rejected.
* Advisors: Receive notifications about course approval requests from students.
* Departments: Ensure that communication between students and advisors is efficient and supports the academic process.
* University Management: Maintain a reliable and organized course registration system.

Precondition(s):

* The student and advisor are registered in the system.
* The student has submitted a course approval request.

Main Flow:

1. The student selects courses and submits them for approval.
2. The system sends a notification to the advisor about the new course approval request.
3. The advisor logs into the system and reviews the course approval request.
4. The advisor either approves or rejects the request.
5. If the advisor approves the request:
6. The system sends a "Course Approved" notification to the student.
7. If the advisor rejects the request:
8. The system sends a "Course Rejected" notification to the student, including the reason for rejection.

Alternative Flow:

* If the student decides to cancel the request before the advisor reviews it:
* The student logs into the system and withdraws the course request.
* The system removes the notification from the advisor's pending requests.

Postcondition(s):

* The student receives a notification about the approval or rejection of their course request.
* The advisor's pending request list is updated accordingly.

**Use Case 5: Notification Management**

Primary Actor: System

Secondary Actors: Student, Advisor

Stakeholders and Interests:

* Students: Want to receive accurate and timely notifications from the system.
* Advisors: Require efficient notification delivery regarding student requests.
* University Management: Ensures that the notification system supports smooth communication within the platform.

Precondition(s):

* The Notification System must be integrated and functional within the platform.
* The users (students and advisors) must be registered in the system.

Main Flow:

1. The system triggers a notification when an event occurs (e.g., a request is submitted or approved).
2. The system creates the notification content and forwards it to the Notification System.
3. The Notification System:
4. Saves the notification in the database.
5. Sends the notification to the intended recipient (student or advisor).
6. When the user logs into the system, they can view unread notifications.
7. The user clicks on a notification to read it and marks it as "read."

Alternative Flow:

* If the user decides to mark all notifications as "read":
* The user selects the "Mark All as Read" option.
* The system updates all notifications to "read" status in the database.

Postcondition(s):

* Notifications are accurately created, stored, and delivered to users.
* Users can manage and organize their notifications effectively.

**Use Case 6: Waitlist Management**

Primary Actor: Student

Stakeholders and Interests:

* Students: Ensure they are notified about their position on the waitlist and automatically enrolled if a slot becomes available.
* Advisors: Ensure students are properly registered in courses without exceeding capacity.
* Departments: Manage course capacities efficiently and ensure fair registration processes.
* Student Affairs’ Office: Oversee the waitlist process and resolve potential disputes.
* University Management: Maintain a fair and effective course registration system.

Precondition(s):

* The student has valid login credentials and is logged into the system.
* The course has reached its maximum capacity.

Main Flow:

1. Student selects a course to register for.
2. System checks the course's capacity.
3. If the course is full, the system adds the student to the waitlist.
4. System monitors the course for available slots.
5. When a slot becomes available, the system automatically enrolls the student.
6. The system notifies the student about the enrollment.

Alternative Flow:

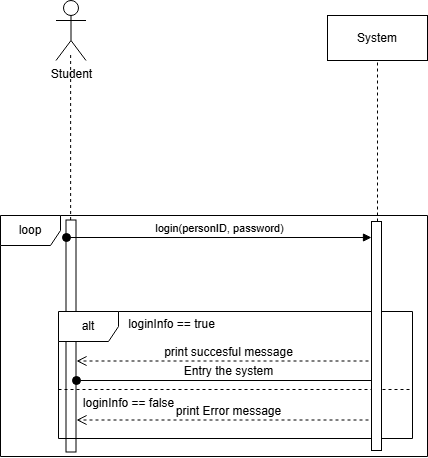
1. If the student decides to leave the waitlist:
2. Student selects the "Leave Waitlist" option.
3. System removes the student from the waitlist and notifies them of the change.

Postcondition(s):

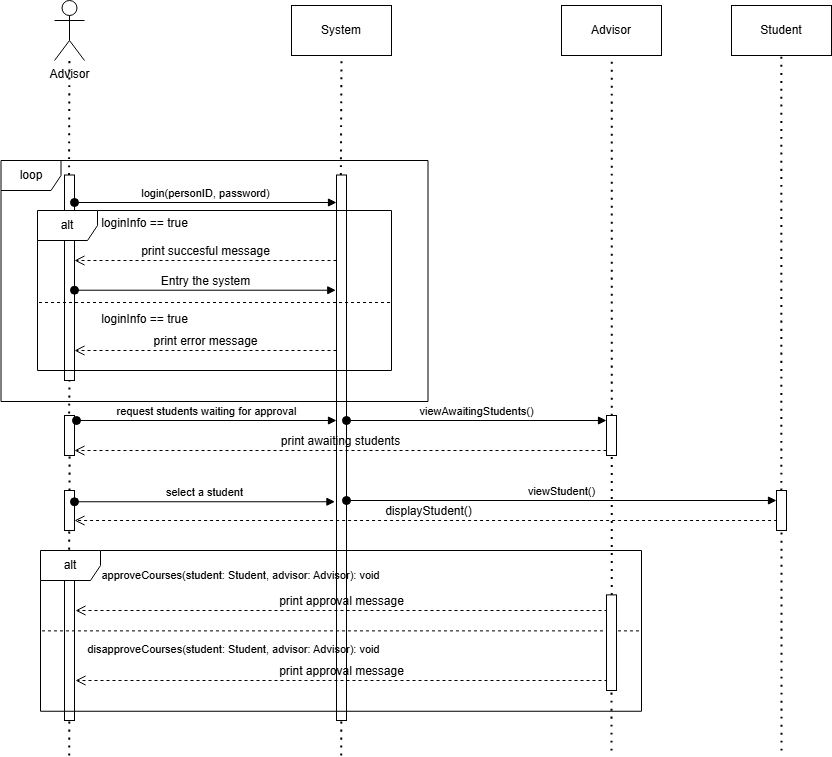
* The student is either enrolled in the course or remains on the waitlist if no slots are available.
* The system logs the action and updates the student's registration status.

**System Sequence Diagrams (SSDs)**

**Login Scenario:**

****

**Advisor Confirmation:**

****