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 1**

**Group 2**

*150120031 - Şükrü Can Mayda  
150121055 - Alp Büyükköse  
150122610 - Yunus Emre Ocak  
150121065 - Turgut Köroğlu  
150120995 - Seyyid Ali Koldaş  
150121545 - Nurbetül Çakır*

**Project Description**

In this project, we will create a course registration system for our department. On the first iteration of the project, the system will accept three types of users; Student, Advisor, Department Schedular. Additional roles may be added in next iterations.

When students logged into the system, they will choose given courses according to their grade, prerequisete courses and transcript. Then, they will send these course choices to advisor to confirm.

Advisor will be able to see who has chosen what courses and he/she will approve completely or approve partly or disapprove these courses.

Our main goal is to make a clear registration system for students and advisors. Everyone should use this system easily.

**Glossary**

* Advisor: Teacher who is responsible for help particular student group which includes adding/dropping courses and help on other subjects.
* Student: Person who is taking courses and responsible with these courses.
* Transcript: Summary of a student's academic performance and progress to date.
* Password: The key of the student's login on registration system.
* Prerequisite Course: The course that is mandatory for specific course.
* Course: The collected information of specific subject.

**Functional Requirements**

**Login:**

* Login page is the first screen when an user wants to enter the system.
* Users must have an username and a password to login. If they don’t have these, theremust be log up to the system with their student numbers.
* If the password is not correct, system will let the user try for 4 more passwords with the screen that warns the user “Password is wrong, try again.”. Then, system will block the user for 1 minutes.

**Course List:**

* Students will be able to see which courses will be given in that term and which courses they can take.
* Advisors are responsible for course list for any update or change.

**Student Interface:**

* Students will be able to see only their information and sections that open for every student.
* Each student will have a unique e-mail address and they can send or take messages from teachers, their advisor and mails that everyone takes.
* There will be a section for changing student information such as phone number, password etc.

**Lecturer Interface:**

* Lecturers will be able to see their all student’s information except their password.
* Each lecturer will have a unique e-mail address. They can send and take messages from students or other lecturers.
* There will be a section for changing personal information.
* Lecturers will be able to add files, notes according to their classes.

**E-Mail:**

* There will be e-mail section for all users.
* All users will have an unique email address.
* There will be no need for extra password e-mails. Users will be able to enter system with their user name or e-mail address as well.
* There will be database for e-mail system.

**Non-Functional Requirements**

System attributes like security, performance, and usability.

* When user enters the system, this information of entering should send to the system as database or file.
* When user exits from the system, changes that user made should processed in the database.
* Every time user entered some area on system, that screen must have been come to the user’s perspective.
* Graphical user interface will not be done, however the interface should be understandable with texts and information.
* On the first and second iteration of project Java will be used; on the third iteration, Python will be used.
* There has to be JSON or another file that keeps the database.

**Domain Model**

UML class diagram showing real-world objects, concepts, their relationships, and features.

??????????????????

**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 or clicks “Forget Password”.

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.

**System Sequence Diagrams (SSDs)**

One SSD for each use case.

Turgut and Alp