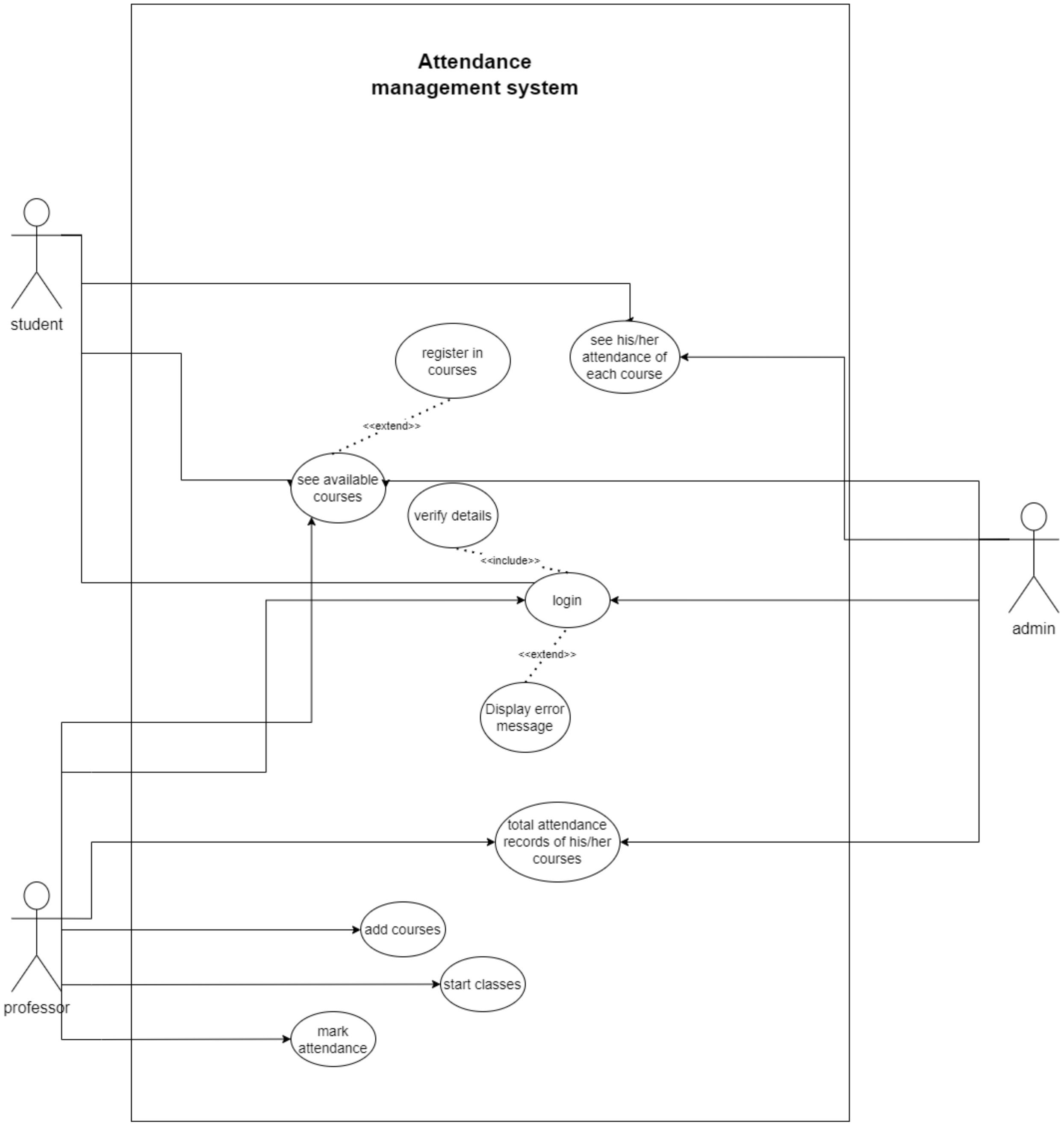
Group - 10
Project id - 82
Project name: Attendance Management System
Lab-3 - Specifying Functional and Non-Functional Requirements.

Use-Case Diagram



Use case Templates

1. Login
   a. Name :
      - Log-in
   b. Actors:
      - All Users
   c. Goals:
      - To log-in in system using user credentials and for security
   d. Description:
      - This use case describes how a user (student,faculty,admin) logs into the Attendance Management System.
      - Show error when log-in credentials are wrong.
   e. Summary:
      - On logging in the system, The user reaches the interface of the system displaying courses according to the type of user.
   f. Post-condition:
      - If the use case was successful, the actor is now logged into the system. If not, the system state is unchanged.
   g. Basic Flow

      This use case starts when an actor wishes to log into the Attendance Management System.

      - The system requests that the actor enter his/her name and password.
      - The actor enters his/her name and password.
      - The system validates the entered name and password and logs the actor into the system.

2. See available courses:

   a. Name:
      - See available courses

   b. Actors:
      - All users

   c. Goals:
      - Students can register.
      - Check attendance in each course.

   d. Description:
      - This use case describes how users can see available courses.

   e. summary:
      - After this student will be able to see available courses according to the student's year and branch.
      - Then he/she will be able to register in courses

   f. Post condition:
      - Students will be able to register
      - Admin can change if needed
      - Professor can alter if needed

   g. Basic flow:
      - System checks what type of user it is
      - provides basic functionalities given what type of user that is using the system.

3.Add courses

a. Name:
   - Add course

b. Actors:
   - Professor
   - admin

c. Goals:
   - User(admin,faculty) can add course

d. Preconditions:
   - User(admin,faculty) should be logged in the system

e. Postconditions:
   - If the use case was successful users will be able to add selected courses.

f. Description:
   - Users (faculty,admin) can add courses in the section of see courses.
   - Users (student, faculty, admin) should be able to see the updated courses.

g. Summary:
   - Users (student, faculty, admin) should be able to see the updated courses.

h. Basic flow:
   - User will login the system
   - System checks which type of user it is
   - user(faculty,admin) can choose what course to add and drop
   - System handles the user input and gives required output
   - System will update add or drop courses in the available courses section.

4. See his/ her attendance of each course

i. Name:
   - See attendance

j. Actors:
   - Students
   - admin

k. Goals:
   - Students can keep track of his attendance.

l. Preconditions:
   - When the user selects the course from the interface after log-in.
   - Students can see his/her own attendance.

m. Description:
   - Users(Students,Professors and Admin) can check the attendance of each student as well as attendance of overall attendance of class.

n. Summary:
   - The user can check their attendance as well as overall attendance of the whole class.

o. Basic flow:
   - Once User has logged in the system, He checks the courses in his interface to see the attendance.
   - If the user is a student, it shows attendance of the student in the course.
   - If the user is Admin or a professor, the system shows overall attendance in the course and the students enrolled in the course upon selecting.

5. Start classes

A. Name:
   - Start classes

B. Actors:

- Professors

C. Goal:

- Professors can start classes to mark attendance

D. Preconditions:

- User should be logged in the system
- System checks for type of user
- Professor should select the class that is to be taught.
- Then there will be an option available to mark attendance.

E. Description:

- Faculty can start class by initiating attendance for students in the system.
- Faculty can update students attendance if required
- Admin can change attendance of student for that particular course

F. Summary:

- User can start class and start initializing the attendance process

G. Postconditions:

- After the professor has taken attendance, the absentees should get a notification and the result of attendance should be updated in the system.

H. Basic flow:

- The system checks for the type of user(It should be the professor teaching the course)
- The User selects the option to take attendance after selecting the course he will be teaching.
- The attendance is updated in the system.
- The absentees get the notification about their absence in the class.

6. Mark Attendance:

A. Name:

- Mark attendance

B. Actor:
   - Faculty
   - Admin

C. Goal:
   - To mark the attendance of student

D. Pre condition:
   - User should be logged in
   - User should be faculty or admin
   - User should be familiar with system
   - User has started the start classes option before this.

E. Post condition:
   - Attendance updated in the system.
   - The absentees get notified.

F. Description:
   - Users will be able to mark their attendance after entering/starting the class.

G. Summary:
   - After starting/entering the class system will mark attendance and then it will update it to datasheet and overall attendance.

H. Basic flow:
   - user logs in to the system.
   - Then the user will join an ongoing class.
   - And he /she is able to mark attendance for ongoing classes.

7. total attendance record of his/her courses

I. Name:
   - total attendance record of his/her courses

J. Actors:
   - Professors
   - Admin

K. Goals:

- Students can keep track of his attendance.

L. Preconditions:

- When the user selects the course from the interface after log-in.
- Students can see his/her own attendance.

M. Description:

- Users(Students and Admin) can check the attendance of each student as well as attendance of overall attendance of class.

N. Summary:

- The user can check their attendance as well as overall attendance of the whole class.

O. Basic flow:

- Once User has logged in the system, He checks the courses in his interface to see the attendance.
- If the user is a student, it shows attendance of the student in the course.
- If the user is Admin or a professor, the system shows overall attendance in the course and the students enrolled in the course upon selecting.

Non-functional requirements

Security:

- The attendance of each student is secured and can be only accessed by the student, professors and admin staff. No one except them will be able to check the attendance of a particular student.

Usability

- System should be simple to use and understand with a basic interface.

Reliability:

- The attendance of a particular student should be marked correctly for a particular course on a particular day.
- There should be no mismatch between the attendance of students.

Performance:

- The attendance of the student is stored in the system and can be accessed whenever needed.
- The system is able to manage the attendance of all the students and is fast enough to produce the output.

Scalability:

- System should be able to scale over a thousand students and efficiently manage the data.