NAME : PIYUSH SAHA

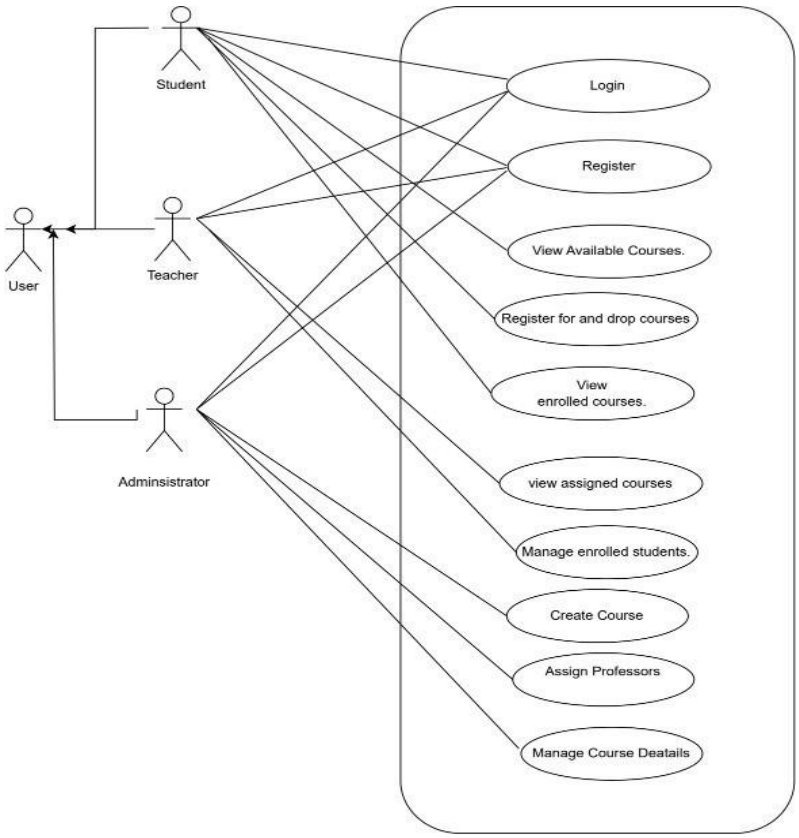
ROLL : 2263053

DEPT : CSE(IOT,CS)

DAY - 1

1. Introduction

This document describes the functional and behavioral requirements of the *University Course Registration System* using **Use Case Modeling**. It includes the **Use Case Diagram**, key assumptions, and detailed use case descriptions for all system functionalities. The system automates course registration for students, professors, and administrators, ensuring efficient course management and registration within defined periods.



2. Assumptions Document

Example assumptions (you can add/remove as needed):

1. Each student, professor, and administrator has a unique login ID.
2. The system operates only within the official university registration period.

3. Course capacity limits are predefined and enforced by the system.
4. Students can enroll in multiple courses, subject to prerequisites.
5. Professors can only manage enrollments for their assigned courses.
6. Only administrators can create and modify course schedules.
7. Internet connectivity is required to access the system.
8. All interactions require successful authentication.

3. Actors

- **Student:** Registers, drops, and views courses.
- **Professor:** Views assigned courses and manages student enrollments.
- **Administrator:** Creates courses, assigns professors, and manages course details.

4. Use Cases Overview

Actor	Use Case
Student	Register for Course, Drop Course, View Courses, View Enrolled Courses, Log In
Professor	View Assigned Courses, Manage Enrolled Students, Log In
Administrator	Create Course, Schedule Classes, Assign Professors, Update Course Details, Log In

5. Detailed Use Case Descriptions

5.1. Log In (Common to all actors)

- **Preconditions:** User must have valid credentials.
- **Main Flow:**
  - User enters username and password.
  - System verifies credentials.
  - Access granted to relevant dashboard.
- **Alternate Flow:**
  - If credentials are invalid, an error message is displayed.
- **Postconditions:** User is logged into the system.
- **Exceptions:** Account locked after 3 failed attempts.

5.2. Register for a Course (Student)

- **Preconditions:** Student is logged in, registration period is active.

- **Main Flow:**
    - Student views list of available courses.
    - Student selects a course.
    - System checks availability and prerequisites.
    - Student confirms registration.
    - System updates enrollment records.
- **Alternate Flow:**
    - If course is full, error message is displayed.
    - If prerequisites not met, registration is denied.
- **Postconditions:** Student is enrolled in the course.
- **Exceptions:** Duplicate registration triggers error.

5.3. Drop a Course (Student)

- **Preconditions:** Student is logged in, drop period is active.
- **Main Flow:**
    - Student selects course to drop.
    - System confirms action.
    - System updates enrollment records.
- **Alternate Flow:**
    - Dropping after deadline not allowed.
- **Postconditions:** Course removed from student's enrollment list.

5.4. View Enrolled Courses (Student)

- **Preconditions:** Student is logged in.
- **Main Flow:**
    1. Student requests list of enrolled courses.
    2. System retrieves and displays data.
- **Postconditions:** Student views current course list.

5.5. Manage Enrolled Students (Professor)

- **Preconditions:** Professor is logged in.
- **Main Flow:**

1. Professor selects course from list of assigned courses.
2. System displays enrolled students.
3. Professor updates or manages records (e.g., marking attendance).

- **Postconditions:** Updated student list is saved.

5.6. Create Course (Administrator)

- **Preconditions:** Administrator is logged in.
- **Main Flow:**
  1. Administrator selects "Create Course" option.
  2. Enters course details (name, code, capacity, prerequisites).
  3. Saves course in system.
- **Postconditions:** Course is added to the course catalog.

5.7. Assign Professors (Administrator)

- **Preconditions:** Administrator is logged in, courses exist.
- **Main Flow:**
  1. Administrator selects a course.
  2. Selects a professor from staff list.
  3. System updates course assignment.
- **Postconditions:** Professor assigned to course.

5.8. Update Course Details (Administrator)

- **Preconditions:** Administrator is logged in, registration period active.
- **Main Flow:**
  1. Administrator selects a course.
  2. Updates details (schedule, location, capacity).
  3. Saves changes.
- **Postconditions:** Updated course details are stored.

6. Conclusion

The *University Course Registration System* streamlines enrollment, course management, and scheduling for all stakeholders. By clearly defining actors, use cases, and flows, this model ensures system requirements are fully understood before development. The structured approach reduces errors, improves communication, and supports further stages of software design and testing .

DAY - 2

1. Introduction

This document provides a detailed overview of the key user-system interactions within the University Course Registration System. It includes the business rules and assumptions governing the system's behavior, step-by-step descriptions of the interaction flows, and a comprehensive UML Sequence Diagram illustrating these processes. The diagram models four primary scenarios: student course registration, student course withdrawal, professor enrollment management, and administrator course creation.

2. Consolidated Assumptions

The following assumptions and business rules apply across the relevant scenarios:

- **Authentication:** All users (Student, Professor, Administrator) must have a valid account and be securely authenticated before accessing system functionalities.
- **Data Integrity:** The course database (CourseDB) is considered the single source of truth and is assumed to be accurate and up-to-date regarding course details, prerequisites, schedules, seat availability, and enrollment records.
- **User Permissions:** Users have distinct roles and permissions. Administrators have the highest level of access for managing system data, while professors and students have access limited to their specific roles.
- **Prerequisites:** Students are required to meet all specified prerequisites before they can successfully enroll in a course.
- **Concurrency:** The system is designed to handle multiple concurrent requests from different users without data conflicts.
- **Add/Drop Period:** It is assumed there is a defined academic period during which students can add or drop courses, though the enforcement of this deadline is outside the scope of these specific diagrams.

3. Sequence Flow Descriptions

A. Student Course Registration

1. **Login:** The **Student** logs into the **System**.
2. **Search:** The **Student** searches for available courses. The **System** queries the CourseDB and displays the results.
3. **Request:** The **Student** selects a course and requests to register.
4. **Verification:** The **System** checks the CourseDB for prerequisite fulfillment and seat availability.
5. **Outcome:**
    - **Success:** If conditions are met, the **System** enrolls the student, updates records in the CourseDB, and sends a success confirmation.
    - **Failure:** If conditions are not met, the **System** displays an error message.

B. Student Dropping a Course

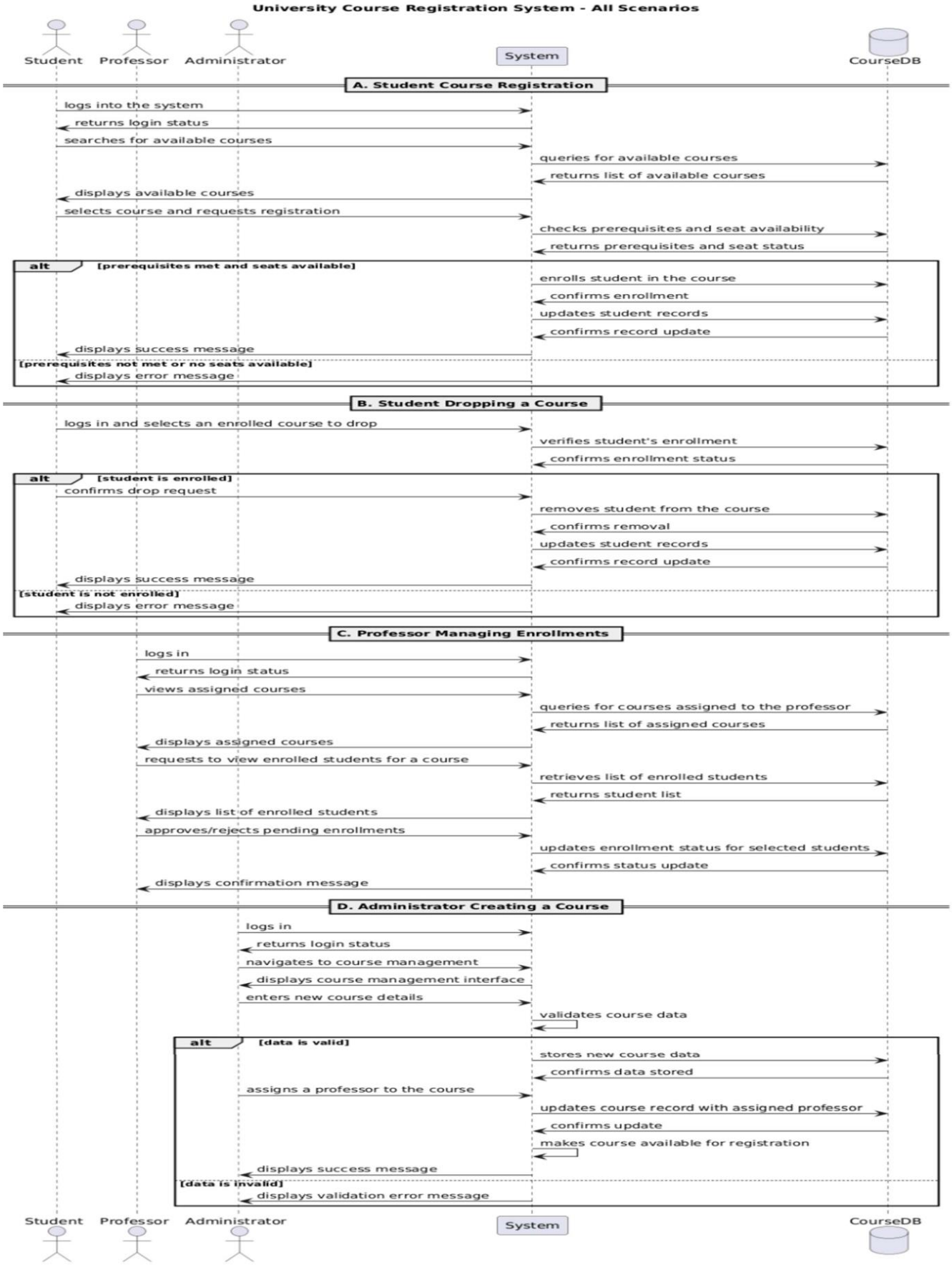
1. **Login s Select:** The **Student** logs in and selects an enrolled course to drop.
2. **Verify:** The **System** confirms with the CourseDB that the student is currently enrolled.
3. **Confirm:** The **Student** confirms the drop request.
4. **Update:** The **System** removes the student from the course in the CourseDB, updates their records, and frees up the seat.
5. **Confirmation:** The **System** displays a success message.

C. **Professor Managing Enrollments**

1. **Login s View:** The **Professor** logs in and views their list of assigned courses, which the **System** retrieves from the CourseDB.
2. **Request Roster:** The **Professor** selects a course to view its student roster.
3. **Display Roster:** The **System** fetches and displays the list of enrolled and pending students from the CourseDB.
4. **Manage:** The **Professor** approves or rejects any students with a "pending" enrollment status.
5. **Update:** The **System** updates the enrollment status in the CourseDB and sends a confirmation to the **Professor**.

D. **Administrator Creating a Course**

1. **Login s Navigate:** The **Administrator** logs in and navigates to the course management interface.
2. **Enter Details:** The **Administrator** provides the details for the new course.
3. **Validate s Store:** The **System** validates the input. If valid, it stores the new course in the CourseDB. If invalid, it shows an error.
4. **Assign Professor:** The **Administrator** assigns a professor to the newly created course.
5. **Finalize:** The **System** updates the course record in the CourseDB with the professor assignment and makes the course available for registration, then displays a success message.


University Course Registration System - All Scenarios
Student
Professor
Administrator
System
CourseDB
A. Student Course Registration
logs into the system
returns login status
searches for available courses
queries for available courses
returns list of available courses
displays available courses
selects course and requests registration
checks prerequisites and seat availability
returns prerequisites and seat status
alt
[prerequisites met and seats available]
enrolls student in the course
confirms enrollment
updates student records
confirms record update
displays success message
[prerequisites not met or no seats available]
displays error message
B. Student Dropping a Course
logs in and selects an enrolled course to drop
verifies student's enrollment
confirms enrollment status
alt
[student is enrolled]
confirms drop request
removes student from the course
confirms removal
updates student records
confirms record update
displays success message
[student is not enrolled]
displays error message
C. Professor Managing Enrollments
logs in
returns login status
views assigned courses
queries for courses assigned to the professor
returns list of assigned courses
displays assigned courses
requests to view enrolled students for a course
retrieves list of enrolled students
returns student list
displays list of enrolled students
approves/rejects pending enrollments
updates enrollment status for selected students
confirms status update
displays confirmation message
D. Administrator Creating a Course
logs in
returns login status
navigates to course management
displays course management interface
enters new course details
validates course data
alt
[data is valid]
stores new course data
confirms data stored
assigns a professor to the course
updates course record with assigned professor
confirms update
makes course available for registration
displays success message
[data is invalid]
displays validation error message